车规 **AEC-Q100 Grade1 MCU** 设计公司

STC32G144K246
STC32G96K246
STC32G18K64
单片机原理及应用

（初稿框架，逐章校核及详细测试中）

DSP 型 32 位 8051
DSP 型 8 位 1T 8051
2 组独立 12 位 ADC；2 组独立 12 位 DAC
4 组独立运放 / PGA，4 组独立比较器

- 10 个 32 位累加器
- 16 个 16 位累加器
- 16 个 8 位累加器
- 32 位加减指令
- 16 位乘除指令
- 32/64 位 DSP 指令（DSP32）
- 单精度浮点运算+三角/反三角函数（TFPU）
- 32 位算术比较指令
- 所有的 SFR（80H~FFH）均支持位寻址
- ebdata（20H~7FH）全部支持位寻址
- 单时钟 32/16/8 位数据读写（edata）
- 单时钟端口读写
- 堆栈理论深度可达 64K（实际取决于 edata）
- 编译器：Keil C51 / 8 位，Keil C251 / 32 位
- 已移植的 RTOS：uC/OS-II，FreeRTOS
- 已移植的文件系统：FATFS，Petit FATFS
- 已移植的 GUI：uGFX，U8g2
- 工作温度：－40℃ ～ ＋125℃

资料更新日期：2026-06-18

(本文档可直接添加备注和标记)

目录

1 单片机基础概述 1
1.1 什么是单片机/微控制器 1
1.1.1 经典单片机 89C52RC/89C58RD+系列组成框图 1
1.1.2 经典单片机 12C5A60S2 组成框图 3
1.1.3 Intel 80251 内部结构图 4
1.2 数制与编码 5
1.2.1 数制转换 6
1.2.2 原码、反码及补码 9
1.2.3 常用编码 9
1.3 几种常用的逻辑运算及其图形符号 10
2 集成开发环境、ISP 下载软件及实验箱 13
2.1 下载并安装 Keil 集成开发环境 13
2.1.1 下载 Keil 集成开发环境 13
2.1.2 安装 Keil 集成开发环境 15
2.1.3 从哪购买 KEIL，获取无限制版 KEIL 22
2.2 安装 AiCube-ISP 下载/编程/烧录 工具,含强大的辅助开发工具 23
2.2.1 安装 AiCube-ISP 下载/编程/烧录 工具软件，取代 STC-ISP 23
2.2.2 STCAI 单片机上电工作过程 24
2.2.3 STC32G144K246 的 ISP 下载流程图（硬件 USB+串口模式） 25
2.2.4 STC32G144K246 的 USB 直接下载电路及 ISP 下载操作步骤 26
2.3 添加型号和头文件到 Keil 28
2.4 STC 单片机程序中头文件的使用方法 30
2.5 STC32G144K246 实验箱实物图及程序下载流程 32
2.5.1 STC32G144K246 实验箱实物图及与电脑连接实物图 32
2.5.2 硬件 USB 下载用户程序步骤 33
2.5.3 下载用户程序文件并观察实验现象 35
2.6 新建项目和项目设置，在 Keil 中 38
2.6.1 准备工作，见前面几小节 38
2.6.2 新建项目 38
2.6.3 项目的各种基础选项设置 42
2.7 了解 File、Build、Debug 工具栏与仿真相关的常用操作，在 Keil 中 45
2.7.1 File Toolbar（File 工具栏） 46
2.7.2 Build Toolbar（Build 工具栏） 50
2.7.3 Debug Toolbar（Debug 工具栏） 52
2.7.4 观察变量/寄存器/内存（数据存储器、程序存储器）窗口 54
2.8 输入中文出现乱码的解决办法，在 Keil uVision5 的编辑器中 55
2.9 在 Keil uVision5 的编辑器中，输入中文出现乱码的解决办法 56
2.10 Keil 软件中因含 0xFD 编码的汉字而出现乱码的问题 57
2.11 C 语言中 printf()函数打印输出数据时，常用的各种输出格式 58
2.12 扩展 Keil 对中断号数量的支持，中断号大于 31 编译出错的处理 59

2.12.1 如何下载 Keil 中断号拓展工具，及安装 59
2.12.2 不使用扩展中断号工具，则需借用保留中断号进行中转 61
2.13 关于 Keil 的双数据指针编译选项 71
2.14 设置超 64K 程序代码的项目（Source 模式） 72
2.14.1 设置 -- “CPU Mode”选择 Source 模式 72
2.14.2 设置 -- “Memory Model”选择 XSmall 模式 72
2.14.3 设置 -- “Code Rom Size”选择 Large 或者 Huge 模式 75
2.14.4 设置 -- 超 64K 代码的相关设置 76
2.14.5 设置 -- HEX 文件格式设置 77
2.15 程序超 64K 时如何设置保留 EEPROM 空间 78
2.16 嵌入汇编代码在 C 语言中，Keil C251 编译器 80
2.17 对变量、常量、表格数据、函数指定绝对地址，在 C251 中 82
2.17.1 Keil C251 中，变量如何指定绝对地址 82
2.17.2 Keil C251 中，常量如何指定绝对地址 83
2.17.3 Keil C251 中，表格数据如何指定绝对地址 84
2.17.4 Keil C251 中，函数如何指定绝对地址 85
2.18 获取帮助的简单方法，Keil 软件中 87
2.19 建立多文件项目的方法，在 Keil 软件中 90
2.20 外部变量和外部函数的声明和使用，在多文件项目中 92
2.21 变量的声明要求在可执行语句之前，C89 标准，KEIL C 语言 93
2.22 STC32G144K246 仿真 -- SWD 硬件仿真 94
2.22.1 安装 AiCube-ISP 软件，制作仿真芯片 94
2.22.2 Debug 和 Utilities 选项设置--在 Keil 集成环境中 99
2.22.3 进入 Debug 界面，观察实验现象 100
3 STC32G144K246 系列选型简介、特性、价格、管脚图 104
3.1 STC32G144K246-LQFP100 总体介绍，USB 下载，烧录 /仿真 线路图 104
3.1.1 特性及价格 104
3.1.2 管脚图，最小系统（LQFP100/QFN100） 108
3.1.3 多功能管脚切换 110
3.1.4 管脚图，最小系统（LQFP64/QFN64） 111
3.1.5 管脚图，最小系统（LQFP48） 113
3.1.6 管脚图，最小系统（LQFP44） 115
3.1.7 不同封装形式管脚对应功能说明 117
3.1.8 USB-Link1D 对 STC32G144K246 自动停电 / 上电烧录，串口仿真+串口通讯 138
3.1.9 【USB 转双串口@Ai8H2K12U-SOP16】工具自动烧录，串口仿真+串口通讯 143
3.1.10 USB 转双串口芯片全自动停电/上电烧录，串口仿真+串口通讯，5V 145
3.1.11 USB 转双串口芯片全自动烧录，串口仿真+串口通讯，3.3V 原理图 146
3.1.12 USB 转双串口芯片进行自动烧录/仿真+串口通讯，5V/3.3V 跳线选择 147
3.1.13 通用 USB 转串口芯片全自动停电/上电烧录，串口仿真，5V 原理图 148
3.1.14 通用 USB 转串口芯片全自动停电/上电烧录，串口仿真，3.3V 原理图 149
3.1.15 USB 转串口芯片全自动停电/上电烧录/仿真/通信，5V/3.3V 跳线选择 150
3.1.16 USB 转串口芯片进行烧录/串口仿真，手动停电/上电，5V/3.3V 原理图 151
3.1.17 USB 转串口芯片进行烧录，串口仿真，手动停电/上电，3.3V 原理图 152
3.1.18 USB-Link1D 支持 脱机下载 说明 153

3.1.19 USB-Link1D 支持 脱机下载，如何免烧录环节 154
3.1.20 USB-Writer1A 编程器/烧录器 支持 插在 锁紧座上 烧录 156
3.1.21 USB-Writer1A 支持 自动烧录机，通信协议和接口 157
3.2 两组独立 DAC、4 组运放、4 组比较器结构及应用，整理中 159
3.2.1 反相放大 164
3.2.2 同相/正相放大 168
3.2.3 Input BUFFER Mode（输入缓冲模式） 170
3.2.4 DAC BUFFER Mode（DAC 缓冲模式） 171
3.2.5 常见的运放应用电路，不是本器件的内部特定电路 Other mode:(建议使用 GP MODE 来外接电路) 172
4 STC32G96K246-64MHz-LQFP100/64/48 系列提前预告 175
4.1.1 特性及价格 175
4.1.2 管脚图（LQFP100） 178
4.1.3 管脚图（LQFP64） 179
4.1.4 管脚图（LQFP48） 180
4.1.5 多功能管脚切换 181
5 STC32G18K64-64MHz-LQFP48/32，TSSOP20 系列提前预告 182
5.1.1 特性及价格 182
5.1.2 管脚图（LQFP48） 185
5.1.3 管脚图（LQFP32） 186
5.1.4 管脚图（TSSOP20） 186
5.1.5 多功能管脚切换 187
6 封装尺寸图、命名规则、USB-TypeC 封装尺寸图及原理图 188
6.1 SOP8 封装尺寸图 188
6.2 DFN8 封装尺寸图（3mm*3mm） 189
6.3 SOP16 封装尺寸图 190
6.4 SOP20 封装尺寸图 191
6.5 TSSOP20 封装尺寸图 192
6.6 QFN20 封装尺寸图（3mm*3mm） 193
6.7 SOP28 封装尺寸图 194
6.8 TSSOP28 封装尺寸图 195
6.9 SKDIP28 封装尺寸图 196
6.10 LQFP32 封装尺寸图（9mm*9mm）-- 塑封体正面 3 个顶杆印 197
6.11 LQFP32 封装尺寸图（9mm*9mm）-- 塑封体正面 1 个顶杆印 198
6.12 QFN32 封装尺寸图（4mm*4mm） 199
6.13 PDIP40 封装尺寸图 200
6.14 LQFP44/QFP44 封装尺寸图（12mm*12mm） 201
6.15 LQFP48 封装尺寸图（9mm*9mm）-- 塑封体正面 3 个顶杆印 202
6.16 LQFP48 封装尺寸图（9mm*9mm）-- 塑封体正面 1 个顶杆印 203
6.17 QFN48 封装尺寸图（6mm*6mm） 204
6.18 LQFP64 封装尺寸图（12mm*12mm） 205
6.19 QFN64 封装尺寸图（8mm*8mm） 206
6.20 LQFP100 封装尺寸图（16mm*16mm） 207
6.21 QFN100 封装尺寸图（12mm*12mm） 208

6.22 USB-TypeC 封装尺寸图，原理图 209
6.23 单片机命名规则 210
7 i-Cache 应用，如何让高速 CPU 和慢速 Flash 协同 211
7.1 程序读取等待控制寄存器（WTST） 211
7.2 速度配置及程序初始化标准参考例程 212
8 时钟管理，芯片上电工作过程，外部晶振抗干扰 213
8.1 系统时钟控制 213
8.2 芯片上电工作过程 214
8.3 时钟配置寄存器，外设，高频 PLL 时钟及分频，外部晶振抗干扰 215
8.3.1 系统时钟选择寄存器（CLKSEL） 216
8.3.2 时钟分频寄存器（CLKDIV） 216
8.3.3 内部高速高精度 IRC 控制寄存器（HIRCCR） 217
8.3.4 外部振荡器控制寄存器（XOSCCR），外部晶振抗干扰 217
8.3.5 内部低速 IRC 控制寄存器（IRC32KCR） 218
8.3.6 外部 32K 振荡器控制寄存器（X32KCR） 219
8.3.7 内部 48MHz 高速 IRC 控制寄存器（IRC48MCR） 219
8.3.8 高速 PLL 控制寄存器（HPLLCR） 220
8.3.9 高速 PLL 预分频寄存器（HPLLPDIV） 220
8.3.10 高速 PLL2 控制寄存器（HPLL2CR） 221
8.3.11 高速 PLL2 预分频寄存器（HPLL2PDIV） 221
8.3.12 主时钟输出控制寄存器（MCLKOCR） 222
8.3.13 内部高速 IRC 时钟恢复振荡到稳定需要等待的时钟数控制寄存器（IRCDB） 222
8.3.14 SPIn 时钟分频寄存器（SPIn_CLKDIV） 223
8.3.15 PWMn 时钟分频寄存器（PWMn_CLKDIV） 223
8.3.16 TFPU 时钟分频寄存器（TFPU_CLKDIV） 224
8.3.17 I2Sn 时钟分频寄存器（I2Sn_CLKDIV） 224
8.4 STC32G144K246 系列内部高速 IRC 时钟频率调整 225
8.4.1 IRC 频段选择寄存器（IRCBAND） 225
8.4.2 内部 IRC 频率调整寄存器（IRTRIM） 225
8.4.3 内部 48M IRC 频率调整寄存器（IRC48MnTRIM） 225
8.5 外部晶振及外部时钟电路 226
8.5.1 外部晶振输入电路 226
8.5.2 外部时钟输入电路（P1.6 为高阻输入模式，可当输入口使用） 226
9 校准内部 IRC 时钟，利用外部 32768Hz 晶振，降低电磁辐射 227
9.1 相关寄存器 227
9.1.1 CRE 控制寄存器（CRECR） 227
9.1.2 CRE 校准计数值寄存器（CRECNT） 228
9.1.3 CRE 校准误差值寄存器（CRERES） 228
9.2 内部高速 IRC 的 4 个频段的中心频率，供参考 229
10 复位、看门狗、掉电唤醒专用定时器与电源管理 230
10.1 系统复位 230
10.1.1 看门狗控制寄存器（WDT_CONTR） 231
10.1.2 IAP 控制寄存器（IAP_CONTR） 232
10.1.3 复位配置寄存器（RSTCFG） 232

10.1.4 复位标志寄存器（RSTFLAG） 233
10.1.5 复位控制寄存器（RSTCRx） 235
10.1.6 内置专业级复位电路，不需传统的阻容式上电延时电路 236
10.1.7 外部低电平按键复位电路 236
10.1.8 传统 8051 高电平上电复位参考电路 236
10.2 主时钟停振/省电模式，系统电源管理 237
10.2.1 电源控制寄存器（PCON） 237
10.2.2 内部高速 IRC 时钟恢复振荡到稳定需要等待的时钟数控制寄存器（IRCDB） 239
10.3 可以唤醒省电模式/主时钟停振模式的中断资源 240
10.4 省电模式唤醒必须设置的寄存器，注意事项 242
10.5 主时钟停振/省电模式，I/O 口如何设置才省电 243
10.6 掉电唤醒定时器 244
10.6.1 掉电唤醒定时器计数寄存器（WKTCL，WKTCH） 244
10.7 省电模式，I/O 口如何设置才省电 246
11 第三方对 STC32G144K246 下载程序范例及如何选择内部不同时钟频率 247
11.1 电源控制参考电路 247
11.2 通信协议流程图 248
11.3 参考代码（C 语言） 251
11.4 如何在用户程序中设置程序运行时的工作频率 258
11.4.1 在用户程序区用户如何动态选择不同的系统时钟频率 258
11.5 如何在用户程序中设置复位脚、低压检测门槛电压等参数 260
12 i-Cache，存储器（32/16/8 位访问），全球唯一 ID 号 CHIPID 储重要参数 261
12.1 i-Cache 及程序存储器 262
12.1.1 程序读取等待控制寄存器（WTST） 263
12.2 数据存储器（32 位访问，16 位访问，8 位访问） 264
12.2.1 Keil 选项 Memory Model 设置 265
12.2.2 内部 edata-RAM（C 语言声明关键字为 edata） 267
12.2.3 程序状态寄存器（PSW） 268
12.2.4 程序状态寄存器 1（PSW1） 268
12.2.5 内部 xdata-RAM（C 语言声明关键字为 xdata） 269
12.2.6 辅助寄存器（AUXR） 269
12.2.7 片外用户自己扩展的外部 XRAM/xdata 269
12.2.8 片外用户扩展 RAM 的总线速度控制寄存器（BUS_SPEED） 270
12.2.9 片内 MCU 扩展 RAM 数据总线时钟控制寄存器（CKCON） 271
12.2.10 片外扩展 RAM 地址总线扩展寄存器（MXAX） 271
12.2.11 STC32G144K246 系列单片机中可位寻址的数据存储器 272
12.2.12 扩展 SFR 使能寄存器 EAXFR 的使用说明 274
12.3 RAM 中运行程序@120MHz 275
12.3.1 使用说明 275
12.3.2 简单科普 277
12.3.3 范例程序 277
12.4 寄存器堆 283
12.4.1 寄存器堆结构图 283
12.4.2 字节、字以及双字寄存器 284

12.4.3 专用寄存器 284
12.4.4 扩展数据指针，DPX 286
12.4.5 扩展堆栈指针，SPX 286
12.5 只读特殊功能寄存器中存储的唯一 ID 号和重要参数（CHIPID 和 CHIPIDX） 287
12.5.1 CHIPID 相关寄存器 287
12.5.2 CHIPID 之全球唯一 ID 号解读 288
12.5.3 CHIPID 之内部参考信号源解读 288
12.5.4 CHIPID 之内部 32K 的 IRC 振荡频率解读 289
12.5.5 CHIPID 之高精度 IRC 参数解读 290
12.5.6 CHIPID 之测试时间参数解读 290
12.5.7 CHIPID 之芯片封装形式编号解读 291
13 特殊功能寄存器（SFR、XFR），EAXFR 允许访问 XFR 292
13.1 EAXFR 允许访问扩展的 XFR，取指令不减速，访问片内部 XRAM 不减速 292
13.2 STC32G144K246 系列 293
13.3 特殊功能寄存器列表及复位值（SFR：0x80-0xFF） 305
13.4 扩展特殊功能寄存器列表（XFR：0x7EFE00-0x7EFEFF） 308
13.5 扩展特殊功能寄存器列表（XFR：0x7EFD00-0x7EFDFF） 314
13.6 扩展特殊功能寄存器列表（XFR：0x7EFB00-0x7EFBFF） 319
13.7 扩展特殊功能寄存器列表（XFR：0x7EFA00-0x7EFAFF） 322
13.8 扩展特殊功能寄存器列表（XFR：0x7EF900-0x7EF9FF） 327
13.9 扩展特殊功能寄存器列表（XFR：0x7EF800-0x7EF8FF） 331
13.10 扩展特殊功能寄存器列表（XFR：0x7EF700-0x7EF7FF） 336
13.11 扩展特殊功能寄存器列表（XFR：0x7EF600-0x7EF6FF） 341
13.12 扩展特殊功能寄存器列表（XFR：0x7EF500-0x7EF5FF） 346
13.13 扩展特殊功能寄存器列表（XFR：0x7EF400-0x7EF4FF） 349
13.14 扩展特殊功能寄存器列表（XFR：0x7EF300-0x7EF3FF） 355
14 I/O 口 360
14.1 I/O 口相关寄存器 361
14.1.1 端口数据寄存器（Pn） 365
14.1.2 端口模式配置寄存器（PnM0，PnM1） 366
14.1.3 端口上拉电阻控制寄存器（PnPU） 367
14.1.4 端口施密特触发控制寄存器（PnNCS） 367
14.1.5 端口电平转换速度控制寄存器（PnSR） 368
14.1.6 端口驱动电流控制寄存器（PnDR） 368
14.1.7 端口数字信号输入使能控制寄存器（PnIE） 369
14.1.8 端口下拉电阻控制寄存器（PnPD） 369
14.1.9 端口模式用户控制寄存器（PnBP） 370
14.1.10 P5.4 和 P1.2 脚功能切换寄存器（P_SWX1） 370
14.2 配置 I/O 口 371
14.3 I/O 的结构图 372
14.3.1 【无弱上拉 准双向口】+上拉电阻 = 准双向口 372
14.3.2 强推挽输出/强上拉 373
14.3.3 高阻输入(电流既不能流入也不能流出) 373
14.3.4 开漏模式 374

14.3.5 STC32G144K246 内部上拉电阻/下拉电阻，小结 375
14.3.6 如何设置 I/O 口对外输出速度 376
14.3.7 如何设置 I/O 口电流驱动能力 376
14.3.8 如何降低 I/O 口对外辐射 376
14.4 AiCube-ISP | I/O 口配置工具 377
14.4.1 普通配置模式 377
14.4.2 高级配置模式 378
14.5 典型发光二极管控制电路 379
14.6 一种典型三极管控制电路 380
14.7 混合电压供电系统 3V/5V 器件 I/O 口互连 381
14.8 如何让 I/O 口上电复位时为低电平 383
14.9 I/O 口直接驱动 LED 数码管应用线路图 383
14.10 使用 I/O 和 R-2R 电阻分压实现 DAC 的经典线路图 384
14.11 利用 ADC 做按键扫描节省 I/O 口 385
14.12 SPI 实测波形图 386
14.13 QSPI 实测波形图 387
15 中断系统及外部中断 INT0/1/2/3/4 389
15.1 STC32G144K246 系列中断源 389
15.2 STC32G144K246 中断及中断优先级结构图 393
15.3 STC32G144K246 系列中断向量地址及同级中断优先级中断查询次序表 394
15.4 中断相关寄存器 401
15.5 总中断控制 408
15.6 外部中断 409
15.6.1 中断入口地址 409
15.6.2 中断使能 409
15.6.3 中断请求 410
15.6.4 中断优先级 410
15.6.5 外部中断类型 411
15.7 普通 I/O 口中断 412
15.7.1 中断入口地址 412
15.7.2 中断使能 412
15.7.3 中断请求 413
15.7.4 中断优先级 413
15.7.5 中断类型 413
15.8 定时器中断 415
15.8.1 中断入口地址 415
15.8.2 中断使能 415
15.8.3 中断请求 417
15.8.4 中断优先级 418
15.9 串口中断 420
15.9.1 中断入口地址 420
15.9.2 中断使能 420
15.9.3 中断请求 421
15.9.4 中断优先级 422

15.9.5 超时中断使能 424
15.9.6 超时中断请求 425
15.10 SPI 中断 426
15.10.1 中断入口地址 426
15.10.2 中断使能 426
15.10.3 中断请求 426
15.10.4 中断优先级 427
15.10.5 超时中断使能 428
15.10.6 超时中断请求 428
15.11 QSPI 中断 429
15.11.1 中断入口地址 429
15.11.2 中断使能 429
15.11.3 中断请求 429
15.11.4 中断优先级 429
15.12 I2C 中断 430
15.12.1 中断入口地址 430
15.12.2 中断使能 430
15.12.3 中断请求 431
15.12.4 中断优先级 432
15.12.5 超时中断使能 432
15.12.6 超时中断请求 432
15.13 I2S 中断 433
15.13.1 中断入口地址 433
15.13.2 中断使能 433
15.13.3 中断请求 434
15.13.4 中断优先级 434
15.14 RTC 中断 435
15.14.1 中断入口地址 435
15.14.2 中断使能 435
15.14.3 中断请求 436
15.14.4 中断优先级 436
15.15 TFT 彩屏/LCM 中断 437
15.15.1 中断入口地址 437
15.15.2 中断使能 437
15.15.3 中断请求 437
15.15.4 中断优先级 437
15.16 高级 PWM 中断 438
15.16.1 中断入口地址 438
15.16.2 中断使能 438
15.16.3 中断请求 439
15.16.4 中断优先级 439
15.17 USB 中断 440
15.17.1 中断入口地址 440
15.17.2 中断使能 440

15.17.3 中断请求 441
15.17.4 中断优先级 443
15.18 ADC 中断 444
15.18.1 中断入口地址 444
15.18.2 中断使能 444
15.18.3 中断请求 444
15.18.4 中断优先级 445
15.19 DAC 中断 446
15.19.1 中断入口地址 446
15.19.2 中断使能 446
15.19.3 中断请求 446
15.19.4 中断优先级 446
15.20 CMP 比较器中断 447
15.20.1 中断入口地址 447
15.20.2 中断使能 447
15.20.3 中断请求 448
15.20.4 中断优先级 448
15.21 LVD 低压检测中断 449
15.21.1 中断入口地址 449
15.21.2 中断使能 449
15.21.3 中断请求 449
15.21.4 中断优先级 449
15.22 DMA 中断 450
15.22.1 中断入口地址 450
15.22.2 中断使能 451
15.22.3 中断请求 454
15.22.4 中断优先级 456
16 普通 I/O 口均可中断，不是传统外部中断 461
16.1 I/O 口中断相关寄存器 461
16.1.1 端口中断使能寄存器（PnINTE） 463
16.1.2 端口中断标志寄存器（PnINTF） 463
16.1.3 端口中断模式配置寄存器（PnIM0，PxIM1，PxIM2） 464
16.1.4 端口中断优先级控制寄存器（PINnIPL，PINnIPH） 466
16.1.5 端口中断掉电唤醒使能寄存器（PnWKUE） 466
17 14 个 24 位定时器/计数器（16 位自动重装载定时器 + 8 位预分频） 467
17.1 定时器 0/1，16 位自动重装载，+ 8 位预分频就是 24 位定时器 467
17.1.1 定时器相关寄存器 467
17.1.2 定时器 0/1 控制寄存器（TCON） 468
17.1.3 定时器 0/1 模式寄存器（TMOD） 469
17.1.4 定时器 0 模式 0，16 位自动重装载，启动 8 位预分频就是 24 位定时器 470
17.1.5 定时器 0 模式 1（16 位不可重装载模式） 471
17.1.6 定时器 0 模式 2（8 位自动重装载模式） 472
17.1.7 定时器 0 模式 3（不可屏蔽中断 16 位自动重装载，实时操作系统节拍器） 473
17.1.8 定时器 1 模式 0，16 位自动重装载，启动 8 位预分频就是 24 位定时器 474

17.1.9 定时器 1 模式 1（16 位不可重装载模式）.....475
17.1.10 定时器 1 模式 2（8 位自动重装载模式）.....476
17.1.11 定时器 0 计数寄存器（TL0，TH0）.....476
17.1.12 定时器 1 计数寄存器（TL1，TH1）.....476
17.1.13 辅助寄存器 1（AUXR）.....477
17.1.14 中断与时钟输出控制寄存器（INTCLKO）.....477
17.1.15 定时器 0 的 8 位预分频寄存器（TM0PS）.....477
17.1.16 定时器 1 的 8 位预分频寄存器（TM1PS）.....477
17.1.17 定时器 0 计算公式.....478
17.1.18 定时器 1 计算公式.....479
17.2 定时器 2，16 位自动重装载，+ 8 位预分频就是 24 位定时器.....479
17.2.1 定时器相关寄存器.....479
17.2.2 辅助寄存器 1（AUXR）.....480
17.2.3 中断与时钟输出控制寄存器（INTCLKO）.....480
17.2.4 定时器 2 计数寄存器（T2L，T2H）.....480
17.2.5 定时器 2 的 8 位预分频寄存器（TM2PS）.....480
17.2.6 定时器 2 工作模式.....481
17.2.7 定时器 2 计算公式.....481
17.3 定时器 3/4，24 位定时器，8 位预分频+16 位自动重装载定时器.....482
17.3.1 定时器功能脚切换.....482
17.3.2 定时器相关寄存器.....482
17.3.3 定时器 4/3 控制寄存器（T4T3M）.....483
17.3.4 定时器 3 计数寄存器（T3L，T3H）.....484
17.3.5 定时器 4 计数寄存器（T4L，T4H）.....484
17.3.6 定时器 3 的 8 位预分频寄存器（TM3PS）.....484
17.3.7 定时器 4 的 8 位预分频寄存器（TM4PS）.....484
17.3.8 定时器 3 工作模式.....485
17.3.9 定时器 4 工作模式.....486
17.3.10 定时器 3 计算公式.....487
17.3.11 定时器 4 计算公式.....487
17.4 定时器 5/6，24 位定时器，8 位预分频+16 位自动重装载定时器.....487
17.4.1 定时器功能脚切换.....487
17.4.2 定时器相关寄存器.....487
17.4.3 定时器 6/5 控制寄存器（T6T5M）.....488
17.4.4 定时器 6/5 配置寄存器（T6T5CFG）.....489
17.4.5 定时器 5 计数寄存器（T5L，T5H）.....489
17.4.6 定时器 6 计数寄存器（T6L，T6H）.....489
17.4.7 定时器 5 的 8 位预分频寄存器（TM5PS）.....490
17.4.8 定时器 6 的 8 位预分频寄存器（TM6PS）.....490
17.4.9 定时器 5 工作模式.....491
17.4.10 定时器 6 工作模式.....492
17.4.11 定时器 5 计算公式.....493
17.4.12 定时器 6 计算公式.....493
17.5 定时器 7/8，24 位定时器，8 位预分频+16 位自动重装载定时器.....494

17.5.1 定时器功能脚切换 494
17.5.2 定时器相关寄存器 494
17.5.3 定时器 8/7 控制寄存器（T8T7M） 495
17.5.4 定时器 8/7 配置寄存器（T8T7CFG） 496
17.5.5 定时器 7 计数寄存器（T7L，T7H） 496
17.5.6 定时器 8 计数寄存器（T8L，T8H） 496
17.5.7 定时器 7 的 8 位预分频寄存器（TM7PS） 497
17.5.8 定时器 8 的 8 位预分频寄存器（TM8PS） 497
17.5.9 定时器 7 工作模式 497
17.5.10 定时器 8 工作模式 498
17.5.11 定时器 7 计算公式 499
17.5.12 定时器 8 计算公式 499
17.6 定时器 9/10，24 位定时器，8 位预分频+16 位自动重装载定时器 500
17.6.1 定时器功能脚切换 500
17.6.2 定时器相关寄存器 500
17.6.3 定时器 10/9 控制寄存器（T10T9M） 501
17.6.4 定时器 10/9 配置寄存器（T10T9CFG） 502
17.6.5 定时器 9 计数寄存器（T9L，T9H） 502
17.6.6 定时器 10 计数寄存器（T10L，T10H） 502
17.6.7 定时器 9 的 8 位预分频寄存器（TM9PS） 503
17.6.8 定时器 10 的 8 位预分频寄存器（TM10PS） 503
17.6.9 定时器 9 工作模式 503
17.6.10 定时器 10 工作模式 504
17.6.11 定时器 9 计算公式 505
17.6.12 定时器 10 计算公式 505
17.7 定时器 T11（24 位定时器，8 位预分频+16 位自动重装载定时器） 506
17.7.1 定时器功能脚切换 506
17.7.2 定时器相关寄存器 506
17.7.3 定时器 T11 控制寄存器（T11CR） 507
17.7.4 定时器 T11 的 8 位预分频寄存器（T11PS） 508
17.7.5 定时器 T11 计数寄存器（T11L，T11H） 508
17.7.6 定时器 T11 工作模式 509
17.7.7 定时器 T11 计算公式 509
17.8 定时器 17/18，24 位定时器，8 位预分频+16 位自动重装载定时器 510
17.8.1 定时器功能脚切换 510
17.8.2 定时器相关寄存器 510
17.8.3 定时器 18/17 控制寄存器（T18T17M） 511
17.8.4 定时器 18/17 配置寄存器（T18T17CFG） 512
17.8.5 定时器 17 计数寄存器（T17L，T17H） 512
17.8.6 定时器 18 计数寄存器（T18L，T18H） 512
17.8.7 定时器 17 的 8 位预分频寄存器（TM7PS） 513
17.8.8 定时器 18 的 8 位预分频寄存器（TM8PS） 513
17.8.9 定时器 17 工作模式 513
17.8.10 定时器 18 工作模式 514

17.8.11 定时器 17 计算公式 515
17.8.12 定时器 18 计算公式 515
17.9 AiCube-ISP | 定时器计算器工具 516
18 RTC 实时时钟，年/月/日/星期/时/分/秒 517
18.1 RTC 相关的寄存器 518
18.1.1 RTC 控制寄存器（RTCCR） 519
18.1.2 RTC 配置寄存器（RTCCFG） 519
18.1.3 RTC 中断使能寄存器（RTCIEN） 520
18.1.4 RTC 中断请求寄存器（RTCIF） 521
18.1.5 RTC 闹钟设置寄存器 521
18.1.6 RTC 实时时钟初始值设置寄存器 522
18.1.7 RTC 实时时钟计数寄存器 523
18.2 RTC 实战线路图 524
19 8 组 USART-LIN/SPI，同步/异步串口通信，硬件奇偶校验 525
19.1 利用串口 DMA 的发送时间间隔寄存器可实现 1.5bit/2bit 停止位 525
19.2 串口 1（同步/异步串口 USART） 526
19.2.1 串口 1 功能脚切换 526
19.2.2 串口 1 相关寄存器 527
19.2.3 串口 1 控制寄存器（SCON） 528
19.2.4 串口 1 数据寄存器（SBUF） 529
19.2.5 电源管理寄存器（PCON） 529
19.2.6 辅助寄存器 1（AUXR） 529
19.2.7 串口 1 模式 0，模式 0 波特率计算公式 530
19.2.8 串口 1 模式 1，模式 1 波特率计算公式 532
19.2.9 串口 1 模式 2，模式 2 波特率计算公式 534
19.2.10 串口 1 模式 3，模式 3 波特率计算公式 535
19.2.11 自动地址识别，从机地址控制寄存器（SADDR，SADEN） 536
19.2.12 串口 1 同步模式控制寄存器 1（USARTCR1） 537
19.2.13 串口 1 同步模式控制寄存器 2，硬件奇偶校验位设置（USARTCR2） 538
19.2.14 串口 1 同步模式控制寄存器 3（USARTCR3） 538
19.2.15 串口 1 同步模式控制寄存器 4（USARTCR4） 539
19.2.16 串口 1 同步模式控制寄存器 5（USARTCR5） 540
19.2.17 串口 1 同步模式保护时间寄存器（USARTGTR） 541
19.2.18 串口 1 同步模式波特率寄存器（USARTBR） 541
19.2.19 串口 1 接收超时控制寄存器（UR1TOCR） 541
19.2.20 串口 1 超时状态寄存器（UR1TOSR） 542
19.2.21 串口 1 超时长度控制寄存器（UR1TOTE/H/L） 542
19.3 串口 2（同步/异步串口 USART2） 543
19.3.1 串口 2 功能脚切换 543
19.3.2 串口 2 相关寄存器 544
19.3.3 串口 2 控制寄存器（S2CON） 545
19.3.4 串口 2 数据寄存器（S2BUF） 546
19.3.5 串口 2 配置寄存器（S2CFG） 546
19.3.6 串口 2 模式 0，模式 0 波特率计算公式 547

19.3.7 串口 2 模式 1，模式 1 波特率计算公式 549
19.3.8 串口 2 模式 2，模式 2 波特率计算公式 551
19.3.9 串口 2 模式 3，模式 3 波特率计算公式 552
19.3.10 串口 2 自动地址识别，从机地址控制寄存器（S2ADDR，S2ADEN） 553
19.3.11 串口 2 同步模式控制寄存器 1（USART2CR1） 554
19.3.12 串口 2 同步模式控制寄存器 2（USART2CR2） 555
19.3.13 串口 2 同步模式控制寄存器 3（USART2CR3） 556
19.3.14 串口 2 同步模式控制寄存器 4（USART2CR4） 556
19.3.15 串口 2 同步模式控制寄存器 5（USART2CR5） 557
19.3.16 串口 2 同步模式保护时间寄存器（USART2GTR） 558
19.3.17 串口 2 同步模式波特率寄存器（USART2BR） 558
19.3.18 串口 2 接收超时控制寄存器（UR2TOCR） 558
19.3.19 串口 2 超时状态寄存器（UR2TOSR） 559
19.3.20 串口 2 超时长度控制寄存器（UR2TOTE/H/L） 559
19.4 串口 3（同步/异步串口 USART3） 560
19.4.1 串口 3 功能脚切换 560
19.4.2 串口 3 相关寄存器 561
19.4.3 串口 3 控制寄存器（S3CON） 562
19.4.4 串口 3 数据寄存器（S3BUF） 563
19.4.5 串口 3 配置寄存器（S3CFG） 563
19.4.6 串口 3 模式 0，模式 0 波特率计算公式 564
19.4.7 串口 3 模式 1，模式 1 波特率计算公式 566
19.4.8 串口 3 模式 2，模式 2 波特率计算公式 568
19.4.9 串口 3 模式 3，模式 3 波特率计算公式 569
19.4.10 串口 3 自动地址识别，从机地址控制寄存器（S3ADDR，S3ADEN） 570
19.4.11 串口 3 同步模式控制寄存器 1（USART3CR1） 571
19.4.12 串口 3 同步模式控制寄存器 2（USART3CR2） 572
19.4.13 串口 3 同步模式控制寄存器 3（USART3CR3） 573
19.4.14 串口 3 同步模式控制寄存器 4（USART3CR4） 573
19.4.15 串口 3 同步模式控制寄存器 5（USART3CR5） 574
19.4.16 串口 3 同步模式保护时间寄存器（USART3GTR） 575
19.4.17 串口 3 同步模式波特率寄存器（USART3BR） 575
19.4.18 串口 3 接收超时控制寄存器（UR3TOCR） 575
19.4.19 串口 3 超时状态寄存器（UR3TOSR） 576
19.4.20 串口 3 超时长度控制寄存器（UR3TOTE/H/L） 576
19.5 串口 4（同步/异步串口 USART4） 577
19.5.1 串口 4 功能脚切换 577
19.5.2 串口 4 相关寄存器 578
19.5.3 串口 4 控制寄存器（S4CON） 579
19.5.4 串口 4 数据寄存器（S4BUF） 580
19.5.5 串口 4 配置寄存器（S4CFG） 580
19.5.6 串口 4 模式 0，模式 0 波特率计算公式 581
19.5.7 串口 4 模式 1，模式 1 波特率计算公式 583
19.5.8 串口 4 模式 2，模式 2 波特率计算公式 585

19.5.9 串口 4 模式 3，模式 3 波特率计算公式 586
19.5.10 串口 4 自动地址识别，从机地址控制寄存器（S4ADDR，S4ADEN） 587
19.5.11 串口 4 同步模式控制寄存器 1（USART4CR1） 588
19.5.12 串口 4 同步模式控制寄存器 2（USART4CR2） 589
19.5.13 串口 4 同步模式控制寄存器 3（USART4CR3） 590
19.5.14 串口 4 同步模式控制寄存器 4（USART4CR4） 590
19.5.15 串口 4 同步模式控制寄存器 5（USART4CR5） 591
19.5.16 串口 4 同步模式保护时间寄存器（USART4GTR） 592
19.5.17 串口 4 同步模式波特率寄存器（USART4BR） 592
19.5.18 串口 4 接收超时控制寄存器（UR4TOCR） 592
19.5.19 串口 4 超时状态寄存器（UR4TOSR） 593
19.5.20 串口 4 超时长度控制寄存器（UR4TOTE/H/L） 593
19.6 串口 5（同步/异步串口 USART5） 594
19.6.1 串口 5 功能脚切换 594
19.6.2 串口 5 相关寄存器 595
19.6.3 串口 5 控制寄存器（S5CON） 596
19.6.4 串口 5 数据寄存器（S5BUF） 597
19.6.5 串口 5 中断控制寄存器（S5INTR） 597
19.6.6 串口 5 配置寄存器（S5CFG） 597
19.6.7 串口 5 模式 0，模式 0 波特率计算公式 598
19.6.8 串口 5 模式 1，模式 1 波特率计算公式 600
19.6.9 串口 5 模式 2，模式 2 波特率计算公式 602
19.6.10 串口 5 模式 3，模式 3 波特率计算公式 603
19.6.11 串口 5 自动地址识别，从机地址控制寄存器（S5ADDR，S5ADEN） 604
19.6.12 串口 5 同步模式控制寄存器 1（USART5CR1） 605
19.6.13 串口 5 同步模式控制寄存器 2（USART5CR2） 606
19.6.14 串口 5 同步模式控制寄存器 3（USART5CR3） 607
19.6.15 串口 5 同步模式控制寄存器 4（USART5CR4） 607
19.6.16 串口 5 同步模式控制寄存器 5（USART5CR5） 608
19.6.17 串口 5 同步模式保护时间寄存器（USART5GTR） 609
19.6.18 串口 5 同步模式波特率寄存器（USART5BR） 609
19.6.19 串口 5 接收超时控制寄存器（UR5TOCR） 609
19.6.20 串口 5 超时状态寄存器（UR5TOSR） 610
19.6.21 串口 5 超时长度控制寄存器（UR5TOTE/H/L） 610
19.7 串口 6（同步/异步串口 USART6） 611
19.7.1 串口 6 功能脚切换 611
19.7.2 串口 6 相关寄存器 612
19.7.3 串口 6 控制寄存器（S6CON） 613
19.7.4 串口 6 数据寄存器（S6BUF） 614
19.7.5 串口 6 中断控制寄存器（S6INTR） 614
19.7.6 串口 6 配置寄存器（S6CFG） 614
19.7.7 串口 6 模式 0，模式 0 波特率计算公式 615
19.7.8 串口 6 模式 1，模式 1 波特率计算公式 617
19.7.9 串口 6 模式 2，模式 2 波特率计算公式 619

19.7.10 串口 6 模式 3，模式 3 波特率计算公式 620
19.7.11 串口 6 自动地址识别，从机地址控制寄存器（S6ADDR，S6ADEN） 621
19.7.12 串口 6 同步模式控制寄存器 1（USART6CR1） 622
19.7.13 串口 6 同步模式控制寄存器 2（USART6CR2） 623
19.7.14 串口 6 同步模式控制寄存器 3（USART6CR3） 624
19.7.15 串口 6 同步模式控制寄存器 4（USART6CR4） 624
19.7.16 串口 6 同步模式控制寄存器 5（USART6CR5） 625
19.7.17 串口 6 同步模式保护时间寄存器（USART6GTR） 626
19.7.18 串口 6 同步模式波特率寄存器（USART6BR） 626
19.7.19 串口 6 接收超时控制寄存器（UR6TOCR） 626
19.7.20 串口 6 超时状态寄存器（UR6TOSR） 627
19.7.21 串口 6 超时长度控制寄存器（UR6TOTE/H/L） 627
19.8 串口 7（同步/异步串口 USART7） 628
19.8.1 串口 7 功能脚切换 628
19.8.2 串口 7 相关寄存器 629
19.8.3 串口 7 控制寄存器（S7CON） 630
19.8.4 串口 7 数据寄存器（S7BUF） 631
19.8.5 串口 7 中断控制寄存器（S7INTR） 631
19.8.6 串口 7 配置寄存器（S7CFG） 631
19.8.7 串口 7 模式 0，模式 0 波特率计算公式 632
19.8.8 串口 7 模式 1，模式 1 波特率计算公式 634
19.8.9 串口 7 模式 2，模式 2 波特率计算公式 636
19.8.10 串口 7 模式 3，模式 3 波特率计算公式 637
19.8.11 串口 7 自动地址识别，从机地址控制寄存器（S7ADDR，S7ADEN） 638
19.8.12 串口 7 同步模式控制寄存器 1（USART7CR1） 639
19.8.13 串口 7 同步模式控制寄存器 2（USART7CR2） 640
19.8.14 串口 7 同步模式控制寄存器 3（USART7CR3） 641
19.8.15 串口 7 同步模式控制寄存器 4（USART7CR4） 641
19.8.16 串口 7 同步模式控制寄存器 5（USART7CR5） 642
19.8.17 串口 7 同步模式保护时间寄存器（USART7GTR） 643
19.8.18 串口 7 同步模式波特率寄存器（USART7BR） 643
19.8.19 串口 7 接收超时控制寄存器（UR7TOCR） 643
19.8.20 串口 7 超时状态寄存器（UR7TOSR） 644
19.8.21 串口 7 超时长度控制寄存器（UR7TOTE/H/L） 644
19.9 串口 8（同步/异步串口 USART8） 645
19.9.1 串口 8 功能脚切换 645
19.9.2 串口 8 相关寄存器 646
19.9.3 串口 8 控制寄存器（S8CON） 647
19.9.4 串口 8 数据寄存器（S8BUF） 648
19.9.5 串口 8 中断控制寄存器（S8INTR） 648
19.9.6 串口 8 配置寄存器（S8CFG） 648
19.9.7 串口 8 模式 0，模式 0 波特率计算公式 649
19.9.8 串口 8 模式 1，模式 1 波特率计算公式 651
19.9.9 串口 8 模式 2，模式 2 波特率计算公式 653

19.9.10 串口 8 模式 3，模式 3 波特率计算公式 654
19.9.11 串口 8 自动地址识别，从机地址控制寄存器（S8ADDR，S8ADEN） 655
19.9.12 串口 8 同步模式控制寄存器 1（USART8CR1） 656
19.9.13 串口 8 同步模式控制寄存器 2（USART8CR2） 657
19.9.14 串口 8 同步模式控制寄存器 3（USART8CR3） 658
19.9.15 串口 8 同步模式控制寄存器 4（USART8CR4） 658
19.9.16 串口 8 同步模式控制寄存器 5（USART8CR5） 659
19.9.17 串口 8 同步模式保护时间寄存器（USART8GTR） 660
19.9.18 串口 8 同步模式波特率寄存器（USART8BR） 660
19.9.19 串口 8 接收超时控制寄存器（UR8TOCR） 660
19.9.20 串口 8 超时状态寄存器（UR8TOSR） 661
19.9.21 串口 8 超时长度控制寄存器（UR8TOTE/H/L） 661
19.10 AiCube-ISP | 串口波特率计算器工具 662
19.11 AiCube-ISP | 串口助手/USB-CDC 663
20 4 组 6P6N 比较器，掉电检测，内部固定比较电压 667
20.1 比较器 1 667
20.1.1 比较器 1 内部结构图 667
20.1.2 比较器 1 功能脚切换 668
20.1.3 比较器 1 相关的寄存器 668
20.1.4 比较器 1 控制寄存器 1（CMP1_6P6N_CR1） 669
20.1.5 比较器 1 控制寄存器 2（CMP1_6P6N_CR2） 670
20.1.6 比较器 1 控制寄存器 3（CMP1_6P6N_CR3） 671
20.1.7 比较器 1 控制寄存器 4（CMP1_6P6N_CR4） 671
20.2 比较器 2 672
20.2.1 比较器 2 内部结构图 672
20.2.2 比较器 2 功能脚切换 673
20.2.3 比较器 2 相关的寄存器 673
20.2.4 比较器 2 控制寄存器 1（CMP2_6P6N_CR1） 674
20.2.5 比较器 2 控制寄存器 2（CMP2_6P6N_CR2） 675
20.2.6 比较器 2 控制寄存器 3（CMP2_6P6N_CR3） 676
20.2.7 比较器 2 控制寄存器 4（CMP2_6P6N_CR4） 676
20.3 比较器 3 677
20.3.1 比较器 3 内部结构图 677
20.3.2 比较器 3 功能脚切换 678
20.3.3 比较器 3 相关的寄存器 678
20.3.4 比较器 3 控制寄存器 1（CMP3_6P6N_CR1） 679
20.3.5 比较器 3 控制寄存器 2（CMP3_6P6N_CR2） 680
20.3.6 比较器 3 控制寄存器 3（CMP3_6P6N_CR3） 681
20.3.7 比较器 3 控制寄存器 4（CMP3_6P6N_CR4） 681
20.4 比较器 4 682
20.4.1 比较器 4 内部结构图 682
20.4.2 比较器 4 功能脚切换 683
20.4.3 比较器 4 相关的寄存器 683
20.4.4 比较器 4 控制寄存器 1（CMP4_6P6N_CR1） 684

20.4.5 比较器 4 控制寄存器 2（CMP4_6P6N_CR2）.....685
20.4.6 比较器 4 控制寄存器 3（CMP4_6P6N_CR3）.....686
20.4.7 比较器 4 控制寄存器 4（CMP4_6P6N_CR4）.....686
21 IAP/EEPROM，1/4/8/16 字节读写.....687
21.1 EEPROM 操作时间.....687
21.2 关于 EEPROM 操作时是否需要关闭中断的问题.....688
21.3 EEPROM 相关的寄存器.....689
21.3.1 EEPROM 数据寄存器（IAP_DATA）.....690
21.3.2 EEPROM 扩展数据寄存器（IAP_XDATA）.....690
21.3.3 EEPROM 地址寄存器（IAP_ADDR）.....690
21.3.4 EEPROM 命令寄存器（IAP_CMD）.....691
21.3.5 EEPROM 触发寄存器（IAP_TRIG）.....692
21.3.6 EEPROM 控制寄存器（IAP_CONTR）.....693
21.3.7 EEPROM 擦除等待时间控制寄存器（IAP_TPS）.....693
21.4 EEPROM 大小及地址.....694
21.4.1 STC32G144K246 系列 EEPROM 操作.....694
22 两组 12 位模数转换（ADC1/ADC2）.....695
22.1 ADC1.....695
22.1.1 ADC1 内部结构图（ADC 框图）.....695
22.1.2 ADC1 相关的寄存器.....696
22.1.3 ADC1 控制寄存器（ADC_CONTR），PWM 触发 ADC 控制.....697
22.1.4 ADC1 配置寄存器（ADCCFG）.....698
22.1.5 ADC1 转换结果寄存器（ADC_RES，ADC_RESL）.....698
22.1.6 ADC1 时序控制寄存器（ADCTIM）.....699
22.1.7 ADC1 扩展配置寄存器（ADCEXCFG）.....700
22.1.8 ADC1 扩展配置寄存器 2（ADCEXCFG2）.....701
22.2 ADC2.....702
22.2.1 ADC2 内部结构图（ADC2 框图）.....702
22.2.2 ADC2 相关的寄存器.....703
22.2.3 ADC2 控制寄存器（ADC2_CONTR），PWM 触发 ADC2 控制.....704
22.2.4 ADC2 配置寄存器（ADC2CFG）.....705
22.2.5 ADC2 转换结果寄存器（ADC2_RES，ADC2_RESL）.....705
22.2.6 ADC2 时序控制寄存器（ADC2TIM）.....706
22.2.7 ADC2 扩展配置寄存器（ADC2EXCFG）.....707
22.2.8 ADC2 扩展配置寄存器 2（ADC2EXCFG2）.....708
22.2.9 ADC2 中断控制寄存器（ADC2INTR）.....709
22.3 ADC 静态特性.....709
22.4 ADC 相关计算公式.....710
22.4.1 ADC 速度计算公式.....710
22.4.2 ADC 转换结果计算公式.....710
22.4.3 反推 ADC 输入电压计算公式.....710
22.4.4 反推工作电压计算公式.....710
22.5 ADC 应用参考线路图.....711
22.5.1 ADC 参考线路图 1 -- 有外部 专业 ADC 转换参考源 REF3025.....711

22.5.2 ADC 参考线路图 2 -- 有外部 普通 ADC 转换参考源 TL431B 712
22.5.3 ADC 参考线路图 3 -- 无外接 ADC 参考源，直接用内部 1.19V BandGap，反推外部输入电压 713
22.5.4 TL431B/CD431 省电模式正规控制电路 714
22.6 AiCube-ISP | ADC 转换速度计算器工具 715
22.7 使用 I/O 和 R-2R 电阻分压实现 DAC 的经典线路图 716
22.8 利用 PWM 实现 16 位 DAC 的参考线路图 717
22.9 ADC 高级应用 1-使用定时器同时触发两组 ADC 实现两路 ADC 同步转换 718
22.10 ADC 高级应用 2-两组 ADC 同时采样两路信号实现差分 ADC 转换 718
22.11 ADC 高级应用 3-使用定时器交替触发两组 ADC 对同一个信号进行采样,实现 ADC 转换速度加倍的目的 718
23 两组 12 位数模转换（DAC1/DAC2） 719
23.1 主要特性 719
23.2 DAC1 720
23.2.1 DAC1 相关的寄存器 720
23.2.2 DAC1 控制寄存器（DAC1_CR） 720
23.2.3 DAC1 状态寄存器（DAC1_STA） 721
23.2.4 DAC1 数据寄存器（DAC1_DATH/DATL） 721
23.2.5 DAC1 时钟分频寄存器（DAC1_DIVH/DIVL） 721
23.3 DAC2 722
23.3.1 DAC2 相关的寄存器 722
23.3.2 DAC2 控制寄存器（DAC2_CR） 722
23.3.3 DAC2 状态寄存器（DAC2_STA） 723
23.3.4 DAC2 数据寄存器（DAC2_DATH/DATL） 723
23.3.5 DAC2 时钟分频寄存器（DAC2_DIVH/DIVL） 723
23.4 电气特性 724
23.5 范例程序 725
23.5.1 【DAC+OP】应用：【DAC1+OP1；DAC2+OP3；DAC2+OP4】同时输出 3 个正弦波 725
23.5.2 录音/放音，ADC+DAC 应用：STC32G144K246-实验箱, 复读机-A 率-存储在 SRAM 733
23.5.3 STC32G144K246，DAC-DMA 例程及注意事项 747
24 4 组可编程增益放大器（PGA/OPA） 754
24.1 主要特性 754
24.2 PGA1 755
24.2.1 PGA1 内部结构图 755
24.2.2 PGA1 功能脚切换 756
24.2.3 PGA1 相关的寄存器 756
24.2.4 PGA1 控制寄存器 1（PGA1_CR1） 757
24.2.5 PGA1 控制寄存器 2（PGA1_CR2） 758
24.3 PGA2 759
24.3.1 PGA2 内部结构图 759
24.3.2 PGA2 功能脚切换 760
24.3.3 PGA2 相关的寄存器 760

24.3.4 PGA2 控制寄存器 1（PGA2_CR1）.....761
24.3.5 PGA2 控制寄存器 2（PGA2_CR2）.....762
24.4 PGA3.....763
24.4.1 PGA3 内部结构图.....763
24.4.2 PGA3 功能脚切换.....764
24.4.3 PGA3 相关的寄存器.....764
24.4.4 PGA3 控制寄存器 1（PGA3_CR1）.....765
24.4.5 PGA3 控制寄存器 2（PGA3_CR2）.....766
24.5 PGA4.....767
24.5.1 PGA4 内部结构图.....767
24.5.2 PGA4 功能脚切换.....768
24.5.3 PGA4 相关的寄存器.....768
24.5.4 PGA4 控制寄存器 1（PGA4_CR1）.....769
24.5.5 PGA4 控制寄存器 2（PGA4_CR2）.....770
24.6 静态特性（IBS=0）.....771
24.7 放大器的典型应用.....772
24.7.1 反相放大模式.....772
24.7.2 非反相放大模式，同相/正相放大.....776
24.7.3 输入缓冲模式.....778
24.7.4 DAC 缓冲模式.....779
24.7.5 常见的运放应用电路，不是本器件的内部特定电路.....780
24.7.6 有源滤波器应用.....783
24.8 运放应用笔记及注意事项.....785
25 3 组同步串行外设接口（SPI）.....788
25.1 SPI1.....788
25.1.1 SPI 的 MOSI 和 MISO 脚交换控制.....788
25.1.2 SPI 功能脚切换.....788
25.1.3 SPI 相关的寄存器.....789
25.1.4 SPI 状态寄存器（SPSTAT）.....789
25.1.5 SPI 控制寄存器（SPCTL），SPI 速度控制.....790
25.1.6 SPI 数据寄存器（SPDAT）.....790
25.1.7 SPI 从机超时控制寄存器（SPITOCR）.....791
25.1.8 SPI 从机超时状态寄存器（SPITOSR）.....791
25.1.9 SPI 从机超时长度控制寄存器（SPITOTE/H/L）.....791
25.1.10 高速 SPI 配置寄存器（HSSPI_CFG）.....792
25.1.11 高速 SPI 配置寄存器 2（HSSPI_CFG2）.....792
25.1.12 高速 SPI 状态寄存器（HSSPI_STA）.....793
25.1.13 高速 SPI 时钟分频器（HSSPI_PSCR）.....793
25.1.14 SPI 时钟分频寄存器（SPI_CLKDIV）.....793
25.1.15 如何提高 SPI+DMA 的传输效率.....793
25.2 SPI2.....794
25.2.1 SPI2 的 MOSI2 和 MISO2 脚交换控制.....794
25.2.2 SPI2 功能脚切换.....794
25.2.3 SPI2 相关的寄存器.....794

25.2.4 SPI2 状态寄存器（SPI2STAT）.....795
25.2.5 SPI2 控制寄存器（SPI2CTL），SPI2 速度控制.....795
25.2.6 SPI2 数据寄存器（SP2DAT）.....796
25.2.7 SPI2 中断控制寄存器（SPI2_INTR）.....796
25.2.8 SPI2 从机超时控制寄存器（SPI2TOCR）.....796
25.2.9 SPI2 从机超时状态寄存器（SPI2TOSR）.....796
25.2.10 SPI2 从机超时长度控制寄存器（SPI2TOTE/H/L）.....797
25.2.11 高速 SPI2 配置寄存器（HSSPI2_CFG）.....797
25.2.12 高速 SPI2 配置寄存器 2（HSSPI2_CFG2）.....797
25.2.13 高速 SPI2 状态寄存器（HSSPI2_STA）.....798
25.2.14 高速 SPI2 时钟分频器（HSSPI2_PSCR）.....798
25.2.15 SPI2 时钟分频寄存器（SPI2_CLKDIV）.....798
25.3 SPI3.....799
25.3.1 SPI3 的 MOSI3 和 MISO3 脚交换控制.....799
25.3.2 SPI3 功能脚切换.....799
25.3.3 SPI3 相关的寄存器.....799
25.3.4 SPI3 状态寄存器（SPSTAT）.....800
25.3.5 SPI3 控制寄存器（SPCTL），SPI3 速度控制.....800
25.3.6 SPI3 数据寄存器（SPDAT）.....801
25.3.7 SPI3 中断控制寄存器（SPI3_INTR）.....801
25.3.8 SPI3 从机超时控制寄存器（SPI3TOCR）.....801
25.3.9 SPI3 从机超时状态寄存器（SPI3TOSR）.....801
25.3.10 SPI3 从机超时长度控制寄存器（SPI3TOTE/H/L）.....802
25.3.11 高速 SPI3 配置寄存器（HSSPI3_CFG）.....802
25.3.12 高速 SPI3 配置寄存器 2（HSSPI3_CFG2）.....803
25.3.13 高速 SPI3 状态寄存器（HSSPI3_STA）.....803
25.3.14 高速 SPI3 时钟分频器（HSSPI3_PSCR）.....804
25.3.15 SPI3 时钟分频寄存器（SPI3_CLKDIV）.....804
25.4 SPI 通信方式.....805
25.4.1 单主单从.....805
25.4.2 互为主从.....805
25.4.3 单主多从.....806
25.5 配置 SPI.....807
25.6 数据模式.....809
26 四线 SPI（QSPI）.....810
26.1 QSPI 功能脚切换.....810
26.2 QSPI 主要特性.....810
26.3 QSPI 功能说明.....811
26.3.1 QSPI 框图.....811
26.3.2 QSPI 应用注意事项.....811
26.3.3 QSPI 命令序列.....812
26.3.4 QSPI 信号接口协议模式.....813
26.3.5 QSPI 间接模式.....815
26.3.6 QSPI 状态标志轮询模式.....816

26.3.7 QSPI Flash 配置 816
26.3.8 QSPI 延迟数据采样 817
26.3.9 QSPI 配置 817
26.3.10 QSPI 的用法 817
26.3.11 指令仅发送一次 819
26.3.12 QSPI 差错管理 819
26.3.13 QSPI 的繁忙位和中止功能 819
26.3.14 nCS 行为 820
26.4 QSPI 中断 821
26.5 QSPI 相关的寄存器 822
26.5.1 QSPI 控制寄存器 1（QSPI_CR1） 823
26.5.2 QSPI 控制寄存器 2（QSPI_CR2） 823
26.5.3 QSPI 控制寄存器 3（QSPI_CR3） 824
26.5.4 QSPI 控制寄存器 4（QSPI_CR4） 825
26.5.5 QSPI 器件配置寄存器 1（QSPI_DCR1） 825
26.5.6 QSPI 器件配置寄存器 2（QSPI_DCR2） 825
26.5.7 QSPI 状态寄存器 1（QSPI_SR1） 826
26.5.8 QSPI 状态寄存器 2（QSPI_SR2） 826
26.5.9 QSPI 标志清零寄存器（QSPI_FCR） 826
26.5.10 QSPI 数据长度寄存器（QSPI_DLR） 827
26.5.11 QSPI 通信配置寄存器 1（QSPI_CCR1） 827
26.5.12 QSPI 通信配置寄存器 2（QSPI_CCR2） 828
26.5.13 QSPI 通信配置寄存器 3（QSPI_CCR3） 828
26.5.14 QSPI 通信配置寄存器 4（QSPI_CCR4） 829
26.5.15 QSPI 地址寄存器（QSPI_AR） 829
26.5.16 QSPI 交替字节寄存器（QSPI_ABR） 829
26.5.17 QSPI 数据寄存器（QSPI_DR） 830
26.5.18 QSPI 状态屏蔽寄存器（QSPI_PSMKR） 830
26.5.19 QSPI 状态匹配寄存器（QSPI_PSMAR） 830
26.5.20 QSPI 轮询间隔寄存器（QSPI_PIR） 830
27 两组 I2C 总线 831
27.1 I2C1 832
27.1.1 I2C 功能脚切换 832
27.1.2 I2C 相关的寄存器 832
27.1.3 I2C 配置寄存器（I2CCFG），总线速度控制 1 833
27.1.4 I2C 主机时钟分频寄存器（I2CPSCR），总线速度控制 2 833
27.1.5 I2C 主机控制寄存器（I2CMSCR） 834
27.1.6 I2C 主机辅助控制寄存器（I2CMSAUX） 835
27.1.7 I2C 主机状态寄存器（I2CMSST） 835
27.1.8 I2C 从机控制寄存器（I2CSLCR） 836
27.1.9 I2C 从机状态寄存器（I2CSLST） 836
27.1.10 I2C 从机地址寄存器（I2CSLADR） 837
27.1.11 I2C 数据寄存器（I2CTXD，I2CRXD） 838
27.1.12 I2C 从机超时控制寄存器（I2CTOCR） 838

27.1.13 I2C 从机超时状态寄存器（I2CTOSR）.....838
27.1.14 I2C 从机超时长度控制寄存器（I2CTOTE/H/L）.....838
27.2 I2C2.....839
27.2.1 I2C2 功能脚切换.....839
27.2.2 I2C2 相关的寄存器.....839
27.2.3 I2C2 配置寄存器（I2C2CFG），总线速度控制 1.....840
27.2.4 I2C2 主机时钟分频寄存器（I2C2PSCR），总线速度控制 2.....840
27.2.5 I2C2 主机控制寄存器（I2C2MSCR）.....841
27.2.6 I2C2 主机辅助控制寄存器（I2C2MSAUX）.....842
27.2.7 I2C2 主机状态寄存器（I2C2MSST）.....843
27.2.8 I2C2 从机控制寄存器（I2C2SLCR）.....843
27.2.9 I2C2 从机状态寄存器（I2C2SLST）.....844
27.2.10 I2C2 从机地址寄存器（I2C2SLADR）.....845
27.2.11 I2C2 数据寄存器（I2C2TXD，I2C2RXD）.....845
27.2.12 I2C2 从机超时控制寄存器（I2C2TOCR）.....846
27.2.13 I2C2 从机超时状态寄存器（I2C2TOSR）.....846
27.2.14 I2C2 从机超时长度控制寄存器（I2C2TOTE/H/L）.....846
28 六组 16 位高级 PWM 定时器，支持正交编码.....847
28.1 高级 PWMA/PWMC/PWME 定时器内部结构框图.....849
28.2 高级 PWMB/PWMD/PWMF 定时器内部结构框图.....851
28.3 简介.....853
28.4 主要特性.....853
28.5 时基单元.....854
28.5.1 读写 16 位计数器.....855
28.5.2 16 位 PWMA_ARR 寄存器的写操作.....855
28.5.3 预分频器.....855
28.5.4 向上计数模式.....856
28.5.5 向下计数模式.....858
28.5.6 中间对齐模式（向上/向下计数）.....860
28.5.7 重复计数器.....862
28.6 时钟/触发控制器.....863
28.6.1 预分频时钟（CK_PSC）.....863
28.6.2 内部时钟源（fMASTER）.....863
28.6.3 外部时钟源模式 1.....864
28.6.4 外部时钟源模式 2.....865
28.6.5 触发同步.....866
28.6.6 与 PWMB 同步.....868
28.7 捕获/比较通道.....871
28.7.1 16 位 PWMA_CCRi 寄存器的写流程.....873
28.7.2 输入模块.....873
28.7.3 输入捕获模式.....874
28.7.4 输出模块.....876
28.7.5 强制输出模式.....877
28.7.6 输出比较模式.....877

28.7.7 PWM 模式 878
28.7.8 使用刹车功能（PWMFLT） 884
28.7.9 在外部事件发生时清除 OCiREF 信号 886
28.7.10 编码器接口模式 886
28.8 中断 888
28.9 PWM 功能脚切换 890
28.9.1 PWMA 功能脚切换（PWMA_PS） 890
28.9.2 PWMB 功能脚切换（PWMB_PS） 892
28.9.3 PWMC 功能脚切换（PWMC_PS） 894
28.9.4 PWMD 功能脚切换（PWMD_PS） 896
28.9.5 PWME 功能脚切换（PWME_PS） 898
28.9.6 PWMF 功能脚切换（PWMF_PS） 900
28.10 PWM 寄存器描述 902
28.10.1 输出使能寄存器（PWMn_ENO） 902
28.10.2 输出附加使能寄存器（PWMn_IOAUX） 902
28.10.3 控制寄存器 1（PWMn_CR1） 903
28.10.4 控制寄存器 2（PWMn_CR2），及实时触发 ADC 904
28.10.5 从模式控制寄存器(PWMn_SMCR) 907
28.10.6 外部触发寄存器(PWMn_ETR) 910
28.10.7 中断使能寄存器(PWMn_IER) 911
28.10.8 状态寄存器 1(PWMn_SR1) 911
28.10.9 状态寄存器 2(PWMn_SR2) 912
28.10.10 事件产生寄存器（PWMx_EGR） 913
28.10.11 捕获/比较模式寄存器 1（PWMx_CCMR1） 914
28.10.12 捕获/比较模式寄存器 2（PWMx_CCMR2） 918
28.10.13 捕获/比较模式寄存器 3（PWMx_CCMR3） 920
28.10.14 捕获/比较模式寄存器 4（PWMx_CCMR4） 922
28.10.15 捕获/比较使能寄存器 1（PWMx_CCER1） 924
28.10.16 捕获/比较使能寄存器 2（PWMx_CCER2） 925
28.10.17 16 位计数器（PWMx_CNTR） 926
28.10.18 16 位预分频器（PWMx_PSCR），输出频率计算公式 927
28.10.19 16 位自动重装载寄存器（PWMx_ARR） 928
28.10.20 16 位重复计数器寄存器（PWMx_RCR） 928
28.10.21 16 位捕获/比较寄存器 1/5（PWMx_CCR1/5） 929
28.10.22 16 位捕获/比较寄存器 2/6（PWMx_CCR2/6） 929
28.10.23 16 位捕获/比较寄存器 3/7（PWMx_CCR3/7） 930
28.10.24 16 位捕获/比较寄存器 4/8（PWMx_CCR4/8） 930
28.10.25 刹车寄存器（PWMx_BKR） 931
28.10.26 死区寄存器（PWMx_DTR） 932
28.10.27 输出空闲状态寄存器（PWMx_OISR） 933
29 高级 PWM-硬件移相 934
29.1 高级 PWM 硬件移相功能脚切换 934
29.2 寄存器描述 936
29.2.1 输出使能寄存器 2（PWMn_ENO2） 936

29.2.2 输出附加使能寄存器 2（PWMn_IOAUX2）.....936
29.2.3 控制寄存器 3（PWMn_CR3）.....937
29.2.4 状态寄存器 3(PWMn_SR3).....938
29.2.5 捕获/比较使能寄存器 3（PWMn_CCER3）.....938
29.2.6 捕获/比较模式扩展寄存器 1（PWMn_CCMR1X）.....939
29.2.7 捕获/比较模式扩展寄存器 2（PWMn_CCMR2X）.....940
29.2.8 捕获/比较模式扩展寄存器 3（PWMn_CCMR3X）.....941
29.2.9 捕获/比较模式扩展寄存器 4（PWMn_CCMR4X）.....941
29.2.10 捕获/比较模式寄存器 5（PWMn_CCMR5）.....941
29.2.11 捕获/比较模式扩展寄存器 5（PWMn_CCMR5X）.....941
29.2.12 捕获/比较模式寄存器 6（PWMn_CCMR6）.....942
29.2.13 捕获/比较模式扩展寄存器 6（PWMn_CCMR6X）.....942
29.2.14 16 位捕获/比较寄存器 5（PWMn_CCR5）.....942
29.2.15 捕获/比较扩展寄存器 5（PWMn_CCR5X）.....943
29.2.16 16 位捕获/比较寄存器 6（PWMn_CCR6）.....943
29.3 移相 PWM 输出模式.....944
29.3.1 不对称 PWM 模式.....944
29.3.2 组合 PWM 模式.....945
29.3.3 组合三相 PWM 模式.....946
29.4 利用不对称 PWM 实现高速正交编码信号输出 （热心网友冲哥提供）.....947
30 高速高级 PWM（HSPWM），可以使用 PLL 高速时钟作为时钟源.....956
30.1 寄存器描述.....956
30.1.1 HSPWM 配置寄存器（HSPWMn_CFG）.....956
30.1.2 HSPWM 地址寄存器（HSPWMn_ADR）.....957
30.1.3 HSPWM 数据寄存器（HSPWMn_DAT）.....957
30.2 高速高级 PWM + DMA（组合使用）特别注意事项.....958
31 USB 2.0-FS 通用串行总线.....959
31.1 USB 相关的寄存器.....960
31.1.1 USB 控制寄存器（USBCON）.....960
31.1.2 USB 时钟控制寄存器（USBCLK）.....961
31.1.3 USB 间址地址寄存器（USBADR）.....962
31.1.4 USB 间址数据寄存器（USBDAT）.....962
31.2 USB 控制器寄存器（SIE）.....963
31.2.1 USB 功能地址寄存器（FADDR）.....964
31.2.2 USB 电源控制寄存器（POWER）.....965
31.2.3 USB 端点 IN 中断标志位（INTRIN1/2）.....965
31.2.4 USB 端点 OUT 中断标志位（INTROUT1/2）.....966
31.2.5 USB 电源中断标志（INTRUSB）.....966
31.2.6 USB 端点 IN 中断允许寄存器（INTRINnE）.....966
31.2.7 USB 端点 OUT 中断允许寄存器（INTROUT1/2E）.....967
31.2.8 USB 电源中断允许寄存器（INTRUSBE）.....967
31.2.9 USB 数据帧号寄存器（FRAMEn）.....967
31.2.10 USB 端点索引寄存器（INDEX）.....968
31.2.11 IN 端点的最大数据包大小（INMAXP）.....968

31.2.12 USB 端点 0 控制状态寄存器（CSR0）.....969
31.2.13 IN 端点控制状态寄存器 1（INCSR1）.....970
31.2.14 IN 端点控制状态寄存器 2（INCSR2）.....971
31.2.15 OUT 端点的最大数据包大小（OUTMAXP）.....971
31.2.16 OUT 端点控制状态寄存器 1（OUTCSR1）.....972
31.2.17 OUT 端点控制状态寄存器 2（OUTCSR2）.....973
31.2.18 USB 端点 0 的 OUT 长度（COUNT0）.....973
31.2.19 USB 端点的 OUT 长度（OUTCOUNTn）.....973
31.2.20 USB 端点的 FIFO 数据访问寄存器（FIFOn）.....974
31.2.21 USB 跟踪控制寄存器（UTRKCTL）.....975
31.2.22 USB 跟踪状态寄存器（UTRKSTS）.....975
31.3 USB 的 VID / PID，使用我们的 VID，可以向我们申请 PID.....976
31.4 如何软复位到系统区启动【USB 直接下载，不管 P3.2】.....976
31.5 使用 USB 时，CPU 如何跑 100MHz ~ 120MHz.....976
32 TFT 彩屏接口接口（8/16 位 i8080/M6800 接口）.....977
32.1 TFT 彩屏接口功能脚切换.....977
32.2 TFT 彩屏相关的寄存器.....978
32.2.1 TFT 彩屏接口配置寄存器（LCMIFCFG）.....978
32.2.2 TFT 彩屏接口配置寄存器 2（LCMIFCFG2）.....979
32.2.3 TFT 彩屏接口控制寄存器（LCMIFCR）.....979
32.2.4 TFT 彩屏接口状态寄存器（LCMIFSTA）.....980
32.2.5 TFT 彩屏接口数据寄存器（LCMIFDATL，LCMIFDATH）.....980
32.2.6 TFT 彩屏接口时钟预分频（LCMIFPSCR）.....980
32.3 i8080/M6800 模式 TFT 彩屏接口时序图.....981
32.3.1 i8080 模式.....981
32.3.2 M6800 模式.....982
33 DMA，支持解放 CPU 的外设到外设传输（P2P）.....983
33.1 DMA 使用注意事项.....984
33.2 外设到外设（P2P）DMA 矩阵.....985
33.3 DMA 存储器读写地址选择（DMA_ARB_CFG）.....986
33.3.1 DMA 仲裁配置寄存器（DMA_ARB_CFG）.....986
33.4 存储器与存储器之间的数据读写（M2M_DMA）.....987
33.4.1 相关的寄存器.....987
33.4.2 M2M_DMA 配置寄存器（DMA_M2M_CFG）.....987
33.4.3 M2M_DMA 控制寄存器（DMA_M2M_CR）.....988
33.4.4 M2M_DMA 状态寄存器（DMA_M2M_STA）.....988
33.4.5 M2M_DMA 传输总字节寄存器（DMA_M2M_AMT）.....988
33.4.6 M2M_DMA 传输完成字节寄存器（DMA_M2M_DONE）.....988
33.4.7 M2M_DMA 发送地址寄存器（DMA_M2M_TXAx）.....988
33.4.8 M2M_DMA 接收地址寄存器（DMA_M2M_RXAx）.....988
33.5 ADC1/2 数据自动存储（ADCn_DMA）.....989
33.5.1 相关的寄存器.....989
33.5.2 ADCn_DMA 配置寄存器（DMA_ADCn_CFG）.....990
33.5.3 ADCn_DMA 控制寄存器（DMA_ADCn_CR）.....990

33.5.4 ADCn_DMA 状态寄存器（DMA_ADCn_STA）.....990
33.5.5 ADCn_DMA 循环扫描总次数寄存器（DMA_ADCn_AMT）.....991
33.5.6 ADCn_DMA 完成扫描次数寄存器（DMA_ADCn_DONE）.....991
33.5.7 ADCn_DMA 接收地址寄存器（DMA_ADCn_RXAx）.....991
33.5.8 ADCn_DMA 配置寄存器 2（DMA_ADCn_CFG2）.....992
33.5.9 ADCn_DMA 通道使能寄存器（DMA_ADCn_CHSWx）.....992
33.5.10 ADCn_DMA 转换周期（DMA_ADCn_ITVx）.....992
33.5.11 ADC_DMA 的数据存储格式.....993
33.6 SPI1/2/3 与存储器之间的数据交换（SPIn_DMA）.....995
33.6.1 相关的寄存器.....995
33.6.2 SPIn_DMA 配置寄存器（DMA_SPIn_CFG）.....997
33.6.3 SPIn_DMA 控制寄存器（DMA_SPIn_CR）.....998
33.6.4 SPIn_DMA 状态寄存器（DMA_SPIn_STA）.....998
33.6.5 SPIn_DMA 传输总字节寄存器（DMA_SPIn_AMT）.....999
33.6.6 SPIn_DMA 传输完成字节寄存器（DMA_SPIn_DONE）.....999
33.6.7 SPIn_DMA 发送地址寄存器（DMA_SPIn_TXAx）.....999
33.6.8 SPIn_DMA 接收地址寄存器（DMA_SPIn_RXAx）.....1000
33.6.9 SPIn_DMA 配置寄存器 2（DMA_SPIn_CFG2）.....1000
33.6.10 SPIn_DMA 传输周期（DMA_SPIn_ITVx）.....1000
33.6.11 如何提高 SPI+DMA 的传输效率.....1001
33.7 利用串口 DMA 的发送时间间隔寄存器可实现 1.5bit/2bit 停止位.....1002
33.8 串口 1/2/3/4/5/6/7/8 与存储器之间的数据交换（URnT_DMA，URnR_DMA）.....1003
33.8.1 相关的寄存器.....1003
33.8.2 URnT_DMA 配置寄存器（DMA_URnT_CFG）.....1008
33.8.3 URnT_DMA 控制寄存器（DMA_URnT_CR）.....1008
33.8.4 URnT_DMA 状态寄存器（DMA_URnT_STA）.....1009
33.8.5 URnT_DMA 传输总字节寄存器（DMA_URnT_AMT）.....1009
33.8.6 URnT_DMA 传输完成字节寄存器（DMA_URnT_DONE）.....1010
33.8.7 URnT_DMA 发送地址寄存器（DMA_URnT_TXAx）.....1010
33.8.8 URn_DMA 传输周期（DMA_URn_ITVx）.....1011
33.8.9 URnR_DMA 配置寄存器（DMA_URnR_CFG）.....1012
33.8.10 URnR_DMA 控制寄存器（DMA_URnR_CR）.....1012
33.8.11 URnR_DMA 状态寄存器（DMA_URnR_STA）.....1013
33.8.12 URnR_DMA 传输总字节寄存器（DMA_URnR_AMT）.....1013
33.8.13 URnR_DMA 传输完成字节寄存器（DMA_URnR_DONE）.....1014
33.8.14 URnR_DMA 接收地址寄存器（DMA_URnR_RXAx）.....1014
33.9 TFT 彩屏与存储器之间的数据读写（LCM_DMA）.....1015
33.9.1 相关的寄存器.....1015
33.9.2 TFT 彩屏 DMA 配置寄存器（DMA_LCM_CFG）.....1015
33.9.3 TFT 彩屏 DMA 控制寄存器（DMA_LCM_CR）.....1016
33.9.4 TFT 彩屏 DMA 状态寄存器（DMA_LCM_STA）.....1016
33.9.5 TFT 彩屏 DMA 传输总字节寄存器（DMA_LCM_AMT）.....1016
33.9.6 TFT 彩屏 DMA 传输完成字节寄存器（DMA_LCM_DONE）.....1017
33.9.7 TFT 彩屏 DMA 发送地址寄存器（DMA_LCM_TXAx）.....1017

33.9.8 TFT 彩屏 DMA 接收地址寄存器（DMA_LCM_RXAx）.....1017
33.9.9 TFT 彩屏 DMA 传输周期（DMA_LCM_ITVx）.....1017
33.10 I2C1/2 与存储器之间的数据交换（I2CnT_DMA，I2CnR_DMA）.....1018
33.10.1 相关的寄存器.....1018
33.10.2 I2CnT_DMA 配置寄存器（DMA_I2CnT_CFG）.....1019
33.10.3 I2CnT_DMA 控制寄存器（DMA_I2CnT_CR）.....1020
33.10.4 I2CnT_DMA 状态寄存器（DMA_I2CnT_STA）.....1020
33.10.5 I2CnT_DMA 传输总字节寄存器（DMA_I2CnT_AMT）.....1020
33.10.6 I2CnT_DMA 传输完成字节寄存器（DMA_I2CnT_DONE）.....1020
33.10.7 I2CnT_DMA 发送地址寄存器（DMA_I2CnT_TXAx）.....1021
33.10.8 I2Cn_DMA 传输周期（DMA_I2Cn_ITVx）.....1021
33.10.9 I2CnR_DMA 配置寄存器（DMA_I2CnR_CFG）.....1021
33.10.10 I2CnR_DMA 控制寄存器（DMA_I2CnR_CR）.....1022
33.10.11 I2CnR_DMA 状态寄存器（DMA_I2CnR_STA）.....1022
33.10.12 I2CnR_DMA 传输总字节寄存器（DMA_I2CnR_AMT）.....1022
33.10.13 I2CnR_DMA 传输完成字节寄存器（DMA_I2CnR_DONE）.....1023
33.10.14 I2CnR_DMA 接收地址寄存器（DMA_I2CnR_RXAx）.....1023
33.10.15 I2Cn_DMA 控制寄存器（DMA_I2Cn_CR）.....1023
33.10.16 I2Cn_DMA 状态寄存器（DMA_I2Cn_ST）.....1024
33.11 I2S1/2 与存储器之间的数据交换（I2SnT_DMA，I2SnR_DMA）.....1025
33.11.1 相关的寄存器.....1025
33.11.2 I2SnT_DMA 配置寄存器（DMA_I2SnT_CFG）.....1026
33.11.3 I2SnT_DMA 控制寄存器（DMA_I2SnT_CR）.....1026
33.11.4 I2SnT_DMA 状态寄存器（DMA_I2SnT_STA）.....1027
33.11.5 I2SnT_DMA 传输总字节寄存器（DMA_I2SnT_AMT）.....1027
33.11.6 I2SnT_DMA 传输完成字节寄存器（DMA_I2SnT_DONE）.....1027
33.11.7 I2SnT_DMA 发送地址寄存器（DMA_I2SnT_TXAx）.....1027
33.11.8 I2Sn_DMA 传输周期（DMA_I2Sn_ITVx）.....1028
33.11.9 I2SnR_DMA 配置寄存器（DMA_I2SnR_CFG）.....1028
33.11.10 I2SnR_DMA 控制寄存器（DMA_I2SnR_CR）.....1028
33.11.11 I2SnR_DMA 状态寄存器（DMA_I2SnR_STA）.....1029
33.11.12 I2SnR_DMA 传输总字节寄存器（DMA_I2SnR_AMT）.....1029
33.11.13 I2SnR_DMA 传输完成字节寄存器（DMA_I2SnR_DONE）.....1029
33.11.14 I2SnR_DMA 接收地址寄存器（DMA_I2SnR_RXAx）.....1029
33.12 QSPI 与存储器之间的数据交换（QSPI_DMA）.....1030
33.12.1 相关的寄存器.....1030
33.12.2 QSPI_DMA 配置寄存器（DMA_QSPI_CFG）.....1030
33.12.3 QSPI_DMA 控制寄存器（DMA_QSPI_CR）.....1031
33.12.4 QSPI_DMA 状态寄存器（DMA_QSPI_STA）.....1031
33.12.5 QSPI_DMA 传输总字节寄存器（DMA_QSPI_AMT）.....1031
33.12.6 QSPI_DMA 传输完成字节寄存器（DMA_QSPI_DONE）.....1031
33.12.7 QSPI_DMA 发送地址寄存器（DMA_QSPI_TXAx）.....1032
33.12.8 QSPI_DMA 接收地址寄存器（DMA_QSPI_RXAx）.....1032
33.12.9 QSPI_DMA 传输周期（DMA_QSPI_ITVx）.....1032

33.13 PWMA/C/E 与存储器之间的数据交换（PWMnT_DMA，PWMnR_DMA）.....1033
33.13.1 PWM 的 DMA 在操作时的注意事项.....1033
33.13.2 相关的寄存器.....1033
33.13.3 PWMnT_DMA 配置寄存器（DMA_PWMnT_CFG）.....1036
33.13.4 PWMnT_DMA 控制寄存器（DMA_PWMnT_CR）.....1036
33.13.5 PWMnT_DMA 状态寄存器（DMA_PWMnT_STA）.....1037
33.13.6 PWMnT_DMA 传输总字节寄存器（DMA_PWMnT_AMT）.....1037
33.13.7 PWMnT_DMA 传输完成字节寄存器（DMA_PWMnT_DONE）.....1037
33.13.8 PWMnT_DMA 发送地址寄存器（DMA_PWMnT_TXAx）.....1038
33.13.9 PWMn_DMA 传输周期（DMA_PWMn_ITVx）.....1038
33.13.10 PWMnR_DMA 配置寄存器（DMA_PWMnR_CFG）.....1038
33.13.11 PWMnR_DMA 控制寄存器（DMA_PWMnR_CR）.....1039
33.13.12 PWMnR_DMA 状态寄存器（DMA_PWMnR_STA）.....1039
33.13.13 PWMnR_DMA 传输总字节寄存器（DMA_PWMnR_AMT）.....1039
33.13.14 PWMnR_DMA 传输完成字节寄存器（DMA_PWMnR_DONE）.....1040
33.13.15 PWMnR_DMA 接收地址寄存器（DMA_PWMnR_RXAx）.....1040
33.13.16 PWMn_DMA 事件使能寄存器（PWMn_DER）.....1040
33.13.17 PWMn_DMA 基址寄存器（PWMn_DBA）.....1041
33.13.18 PWMn_DMA 突发传输次数寄存器（PWMn_DBL）.....1042
33.13.19 PWMn_DMA 控制寄存器（PWMn_DMACR）.....1043
33.14 DAC1/2 数据自动存储（DACn_DMA）.....1044
33.14.1 相关的寄存器.....1044
33.14.2 DACn_DMA 配置寄存器（DMA_DACn_CFG）.....1045
33.14.3 DACn_DMA 控制寄存器（DMA_DACn_CR）.....1045
33.14.4 DACn_DMA 状态寄存器（DMA_DACn_STA）.....1045
33.14.5 DACn_DMA 循环扫描总次数寄存器（DMA_DACn_AMT）.....1046
33.14.6 DACn_DMA 完成扫描次数寄存器（DMA_DACn_DONE）.....1046
33.14.7 DACn_DMA 发送地址寄存器（DMA_DACn_TXAx）.....1046
33.14.8 DACn_DMA 转换周期（DMA_DACn_ITVx）.....1046
33.15 CAN-FD1/2 与存储器之间的数据交换（CANnT_DMA，CANnR_DMA）.....1047
33.15.1 相关的寄存器.....1047
33.15.2 CANnT_DMA 配置寄存器（DMA_CANnT_CFG）.....1049
33.15.3 CANnT_DMA 控制寄存器（DMA_CANnT_CR）.....1050
33.15.4 CANnT_DMA 状态寄存器（DMA_CANnT_STA）.....1050
33.15.5 CANnT_DMA 传输总字节寄存器（DMA_CANnT_AMT）.....1050
33.15.6 CANnT_DMA 传输完成字节寄存器（DMA_CANnT_DONE）.....1050
33.15.7 CANnT_DMA 发送地址寄存器（DMA_CANnT_TXAx）.....1051
33.15.8 CANnT_DMA 发送帧大小寄存器（DMA_CANnT_FRM）.....1051
33.15.9 CANnT_DMA 发送帧数寄存器（DMA_CANnT_FCNTx）.....1051
33.15.10 CANn_DMA 传输周期（DMA_CANn_ITVx）.....1051
33.15.11 CANnR_DMA 配置寄存器（DMA_CANnR_CFG）.....1052
33.15.12 CANnR_DMA 控制寄存器（DMA_CANnR_CR）.....1053
33.15.13 CANnR_DMA 状态寄存器（DMA_CANnR_STA）.....1053
33.15.14 CANn_DMA 辅助控制寄存器（CANn_AUX_CR）.....1054

33.15.15 CANnR_DMA 传输总字节寄存器（DMA_CANnR_AMT）.....1054
33.15.16 CANnR_DMA 传输完成字节寄存器（DMA_CANnR_DONE）.....1054
33.15.17 CANnR_DMA 接收地址寄存器（DMA_CANnR_RXAx）.....1055
33.15.18 CANnR_DMA 接收帧大小寄存器（DMA_CANnR_FRM）.....1055
33.15.19 CANnR_DMA 接收帧数寄存器（DMA_CANnR_FCNTx）.....1055
33.16 外设与外设之间的数据交换（P2P_DMA）.....1056
33.16.1 P2P 使用注意事项.....1056
33.16.2 相关的寄存器.....1056
33.16.3 P2Pn_DMA 目标外设选择寄存器（DMA_P2Pn_DEST）.....1057
33.16.4 P2Pn_DMA 源外设选择寄存器（DMA_P2Pn_SRC）.....1058
34 两组 I2S 音频总线，可以使用 PLL 高速时钟作为时钟源.....1059
34.1 I2S 功能脚切换.....1059
34.2 寄存器描述.....1060
34.2.1 I2S 控制寄存器（I2SnCR）.....1060
34.2.2 I2S 状态寄存器（I2SnSR）.....1061
34.2.3 I2S 数据寄存器（I2SnDRH、I2SnDRL）.....1061
34.2.4 I2Sn 时钟分频寄存器（I2Sn_CLKDIV）.....1062
34.2.5 I2S 位时钟 BCLK 分频寄存器高字节（I2SnPRH）.....1063
34.2.6 I2S 位时钟 BCLK 分频寄存器低字节（I2SnPRL）.....1063
34.2.7 I2S 配置寄存器高字节（I2SnCFGH）.....1064
34.2.8 I2S 配置寄存器低字节（I2SnCFGL）.....1064
34.2.9 I2S 主时钟 MCLK 分频寄存器（I2SnMCKDIV）.....1065
34.2.10 I2S 输入信号选择寄存器（I2SnINSEL）.....1065
35 TFPU（三角函数+单精度浮点运算器），可以使用 PLL 高速时钟作为时钟源...1066
35.1 TFPU 浮点运算器简介.....1066
35.2 相关的特殊功能寄存器.....1066
35.3 运算执行时间表.....1067
35.4 TFPU 基本算术运算.....1068
35.4.1 浮点数加法（+）.....1068
35.4.2 浮点数减法（－）.....1068
35.4.3 浮点数乘法（×）.....1068
35.4.4 浮点数除法（÷）.....1069
35.4.5 浮点数开方/平方根（sqrt）.....1069
35.4.6 浮点数比较（comp）.....1069
35.4.7 浮点数检测（check）.....1070
35.5 TFPU 三角函数.....1071
35.5.1 正弦函数（sin）.....1071
35.5.2 余弦函数（cos）.....1071
35.5.3 正切函数（tan）.....1071
35.5.4 反正切函数（arctan）.....1071
35.6 TFPU 数据转换操作.....1072
35.6.1 浮点数转 8 位整数（float → char）.....1072
35.6.2 浮点数转 16 位整数（float → short）.....1072
35.6.3 浮点数转 32 位整数（float → long）.....1072

35.6.4 8 位整数转浮点数（char → float）.....1072
35.6.5 16 位整数转浮点数（short → float）.....1073
35.6.6 32 位整数转浮点数（long → float）.....1073
35.7 TFPU 协处理器控制操作.....1074
35.7.1 初始化协处理器.....1074
35.7.2 清除异常.....1074
35.7.3 读状态寄存器.....1074
35.7.4 写状态寄存器.....1074
35.7.5 读控制寄存器.....1074
35.7.6 写控制寄存器.....1074
35.8 如何选择 TFPU 的时钟源.....1075
35.8.1 选择系统时钟（和 CPU 时钟同步）作为 TFPU 时钟源.....1075
35.8.2 选择 PLL 时钟（和 CPU 时钟异步）作为 TFPU 时钟源.....1075
36 DSP32（DSP 指令，32 位及 64 位整数运算器）.....1076
36.1 相关的特殊功能寄存器.....1076
36.2 DSP32 操作寄存器说明.....1077
36.3 运算执行时间表.....1078
36.4 DSP32 指令说明.....1081
36.4.1 配置 DPU（FILL_DPUCFG）.....1081
36.4.2 编码转换.....1081
36.4.3 规格化.....1082
36.4.4 互换运算.....1083
36.4.5 加减乘除运算.....1084
36.4.6 比较运算.....1089
36.4.7 清零/置位运算.....1090
36.4.8 取负数运算.....1092
36.4.9 累加运算.....1093
36.4.10 逻辑运算.....1097
36.4.11 移位运算.....1099
36.4.12 复合运算.....1103
37 硬件 CRC32/16（循环冗余校验）.....1105
37.1 相关的特殊功能寄存器.....1105
37.1.1 CRC 控制寄存器（CRC_CR）.....1106
37.1.2 CRC 数据输入寄存器（CRC_DIN）.....1106
37.1.3 16 位 CRC 计算结果寄存器（CRC_DOH、CRC_DOL）.....1106
37.1.4 32 位 CRC 计算结果寄存器（CRC_DO0/1/2/3）.....1106
38 CAN_FD，兼容 CAN2.0B/A.....1107
38.1 CAN-FD 简介.....1107
38.1.1 为什么开发 CAN-FD.....1107
38.1.2 CAN-FD 帧格式.....1108
38.1.3 CAN-FD 和标准 CAN 的区别.....1115
38.2 CAN-FD 主要特性.....1116
38.3 CAN-FD 功能说明.....1117
38.3.1 操作模式.....1117

38.3.2 波特率设定 1117
38.3.3 发送缓冲器 1119
38.3.4 接收缓冲器 1120
38.3.5 接收筛选寄存器组 1120
38.3.6 数据发送 1121
38.3.7 单次数据发送 1122
38.3.8 取消数据发送 1122
38.3.9 数据接收 1122
38.3.10 错误处理 1123
38.3.11 节点关闭 1123
38.3.12 仲裁失败位置捕捉 1124
38.3.13 回环模式 1124
38.3.14 静默模式 1125
38.3.15 软件复位功能 1125
38.3.16 向上兼容 CAN-FD 功能 1126
38.3.17 时间触发 TTCAN 1127
38.3.18 TDC 和 RDC 1130
38.3.19 中断 1130
38.4 使用 CAN-FD / CAN 时，CPU 如何跑 100MHz ~ 120MHz 1131
38.5 CAN1 功能脚切换 1131
38.6 CAN2 功能脚切换 1131
38.7 CAN-FD 相关的特殊功能寄存器 1132
38.7.1 CAN 接收缓冲区寄存器（CAN_RBUFn） 1132
38.7.2 CAN 发送缓冲区寄存器（CAN_TBUFn） 1134
38.7.3 CAN 传输时间戳寄存器（CAN_TTSn） 1135
38.7.4 CAN 配置状态寄存器（CAN_CFG_STAT） 1135
38.7.5 CAN 命令寄存器（CAN_TCMD） 1136
38.7.6 CAN 发送控制寄存器（CAN_TCTRL） 1136
38.7.7 CAN 接收控制寄存器（CAN_RCTRL） 1137
38.7.8 CAN 中断使能寄存器（CAN_RTIE） 1138
38.7.9 CAN 中断状态寄存器（CAN_RTIF） 1139
38.7.10 CAN 错误中断寄存器（CAN_ERRINT） 1140
38.7.11 CAN 警告限制寄存器（CAN_LIMIT） 1140
38.7.12 CAN 慢速位段 1 时间设定寄存器（CAN_S_SEGn） 1141
38.7.13 CAN 慢速同步补偿时间设定寄存器（CAN_S_SJW） 1141
38.7.14 CAN 慢速预分频寄存器（CAN_S_PRESC） 1141
38.7.15 CAN 快速位段 1 时间设定寄存器（CAN_F_SEGn） 1142
38.7.16 CAN 快速同步补偿时间设定寄存器（CAN_F_SJW） 1142
38.7.17 CAN 快速预分频寄存器（CAN_F_PRESC） 1142
38.7.18 CAN 错误和仲裁丢失捕获寄存器（CAN_EALCAP） 1143
38.7.19 CAN 发送器延迟补偿寄存器（CAN_TDC） 1143
38.7.20 CAN 接收错误计数器（CAN_RECNT） 1143
38.7.21 CAN 发送错误计数器（CAN_TECNT） 1143
38.7.22 CAN 筛选器组控制寄存器（CAN_ACFCTRL） 1144

38.7.23 CAN 时间戳控制寄存器（CAN_TIMECFG）.....1144
38.7.24 CAN 筛选器组使能寄存器（CAN_ACF_ENn）.....1144
38.7.25 CAN 筛选器 code 和 mask 寄存器（CAN_ACFn）.....1145
38.7.26 CAN 版本信息寄存器（CAN_VERn）.....1145
38.7.27 CANTB SLOT 指针寄存器（CAN_TBSLOT）.....1146
38.7.28 CAN 时间触发配置寄存器（CAN_TTCFG）.....1146
38.7.29 CAN 参考消息寄存器（CAN_REF_MSGn）.....1147
38.7.30 CAN 触发配置寄存器（CAN_TRIG_CFGn）.....1148
38.7.31 CAN 触发时间寄存器（CAN_TT_TRIGn）.....1149
38.7.32 CAN 触发看门时间寄存器（CAN_TT_WTRIGn）.....1149
39 LIN 总线.....1150
39.1 LIN 总线时序介绍.....1150
39.2 LIN1 功能脚切换.....1153
39.3 LIN2 功能脚切换.....1153
39.4 LIN 相关的寄存器.....1154
39.4.1 辅助寄存器 2（AUXR2）.....1154
39.4.2 LIN 总线中断控制寄存器（LINICR）.....1155
39.4.3 LIN 总线地址寄存器（LINAR）.....1155
39.4.4 LIN 总线数据寄存器（LINDR）.....1156
39.5 LIN 内部功能寄存器.....1157
39.5.1 LIN 数据寄存器（LBUF）.....1157
39.5.2 LIN 数据地址寄存器（LSEL）.....1157
39.5.3 LIN 帧 ID 寄存器（LID）.....1158
39.5.4 LIN 错误寄存器（LER）.....1158
39.5.5 LIN 中断使能寄存器（LIE）.....1158
39.5.6 LIN 状态寄存器（只读寄存器）（LSR）.....1159
39.5.7 LIN 控制寄存器（只写寄存器）（LCR）.....1159
39.5.8 LIN 波特率寄存器（DLL/DLH）.....1160
39.5.9 LIN 帧头延时计数寄存器（HDRL/HDRH）.....1160
39.5.10 LIN 帧头延时分频寄存器（HDP）.....1160
附录 A 指令集.....1161
A.1 指令集简介.....1161
A.1.1 BINARY 模式和 SOURCE 模式.....1161
A.1.2 指令集标记.....1161
A.1.3 指令表（功能排序）.....1163
A.1.4 指令表（机器码排序）.....1169
A.2 指令详解.....1173
A.3 多级流水线内核的中断响应.....1255
附录 B 重要说明.....1257
附录 C 电气特性.....1258
C.1 I/O 口驱动能力（驱动电流对应的 I/O 上的电压）.....1261
C.2 ADC 静态特性.....1263
C.3 DAC 电气特性.....1263
C.4 PGA 静态特性（IBS=0）.....1264

C.5 PLL500M 电气特性 1265
附录 D 单片机电源系统最简易自我保护电路 1266
附录 E 防电源反接电路，汽车电子常用电路 1267
附录 F 更新记录 1268
附录 G 产品授权书 .... 甲
附录 H 自主产权，生产可控 .... 乙

1 单片机基础概述

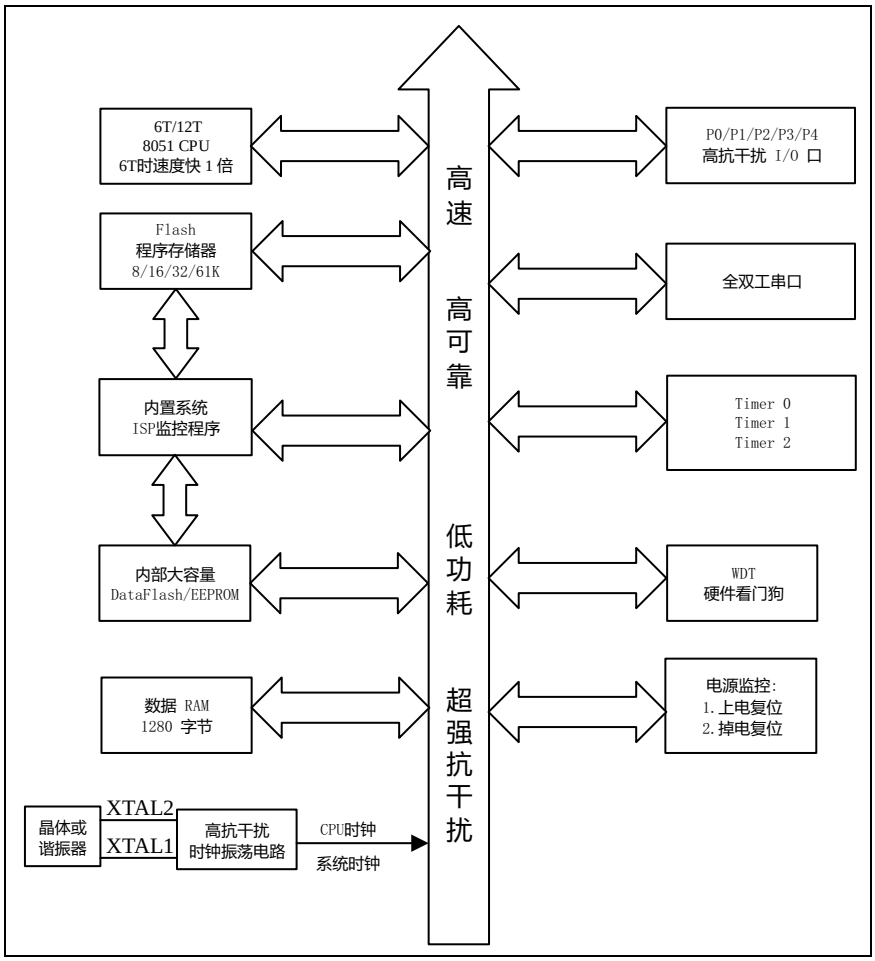
——无微机原理的用户请从本章开始学习

这一章主要讲述的内容有：①在数字设备中进行算术运算的基本知识——数制和编码；②数字电路中一些常用逻辑运算及其图形符号。它们是学习单片机这门课程的基础。对于没有微机原理基础的用户和同学，请从这章开始学习。

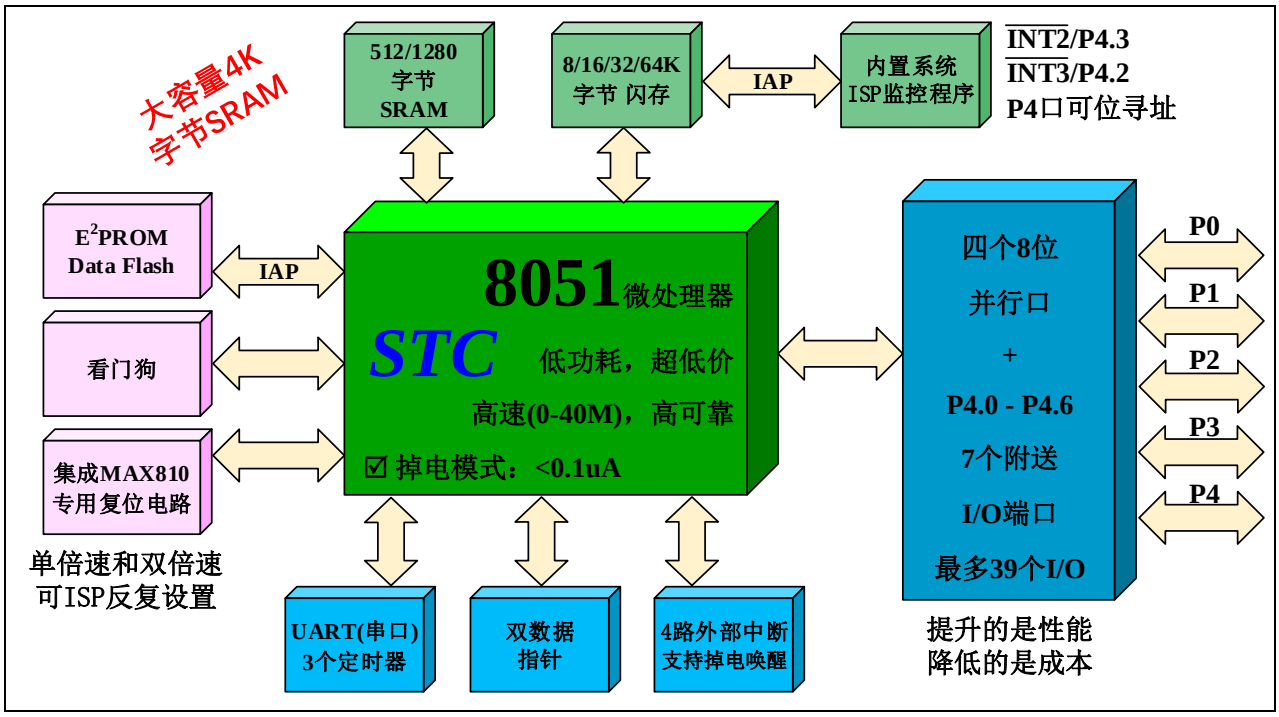
1.1 什么是单片机/微控制器

在一个芯片上实现一个完整的计算机系统就是单片微型计算机，简称单片机，俗称 MCU/微控制器。当然，有些有机械件的外围，由于材料的问题无法集成，如键盘，如显示器。

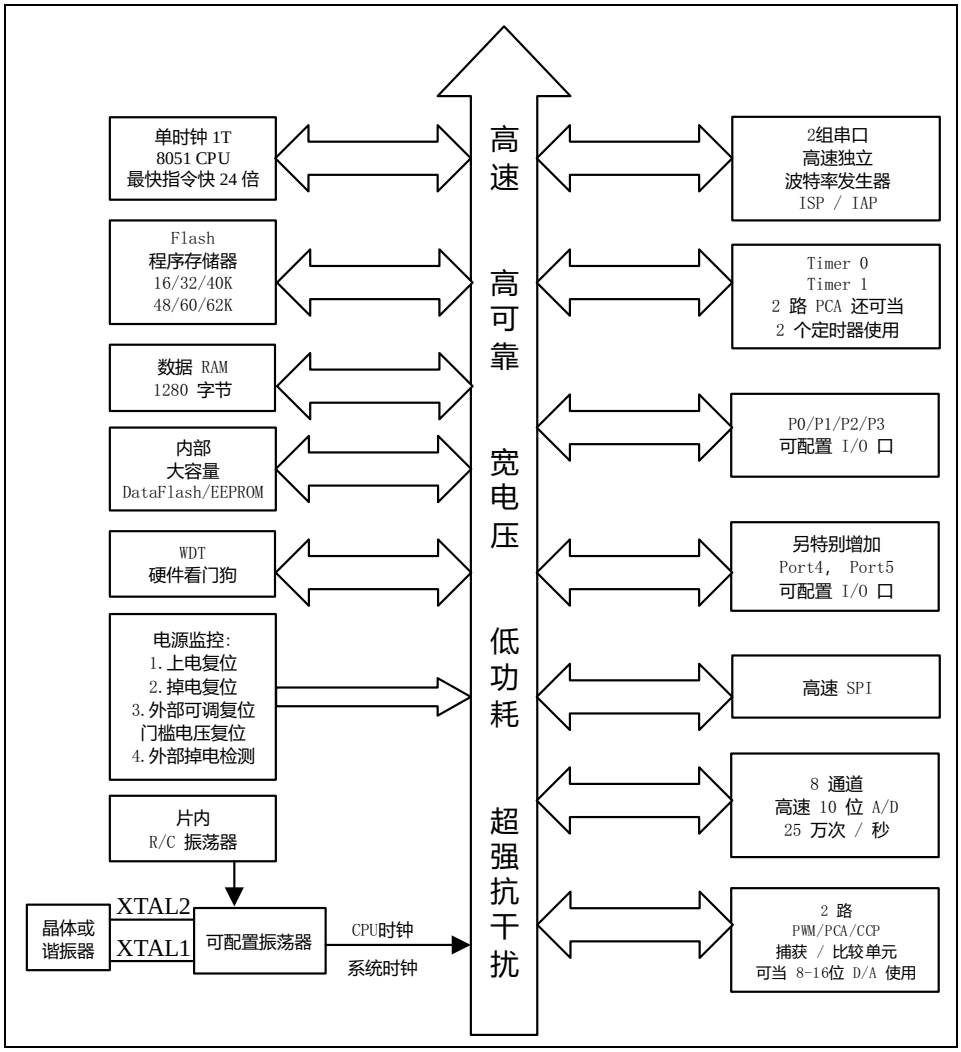
1.1.1 经典单片机 89C52RC/89C58RD+系列组成框图



更通俗的框图：

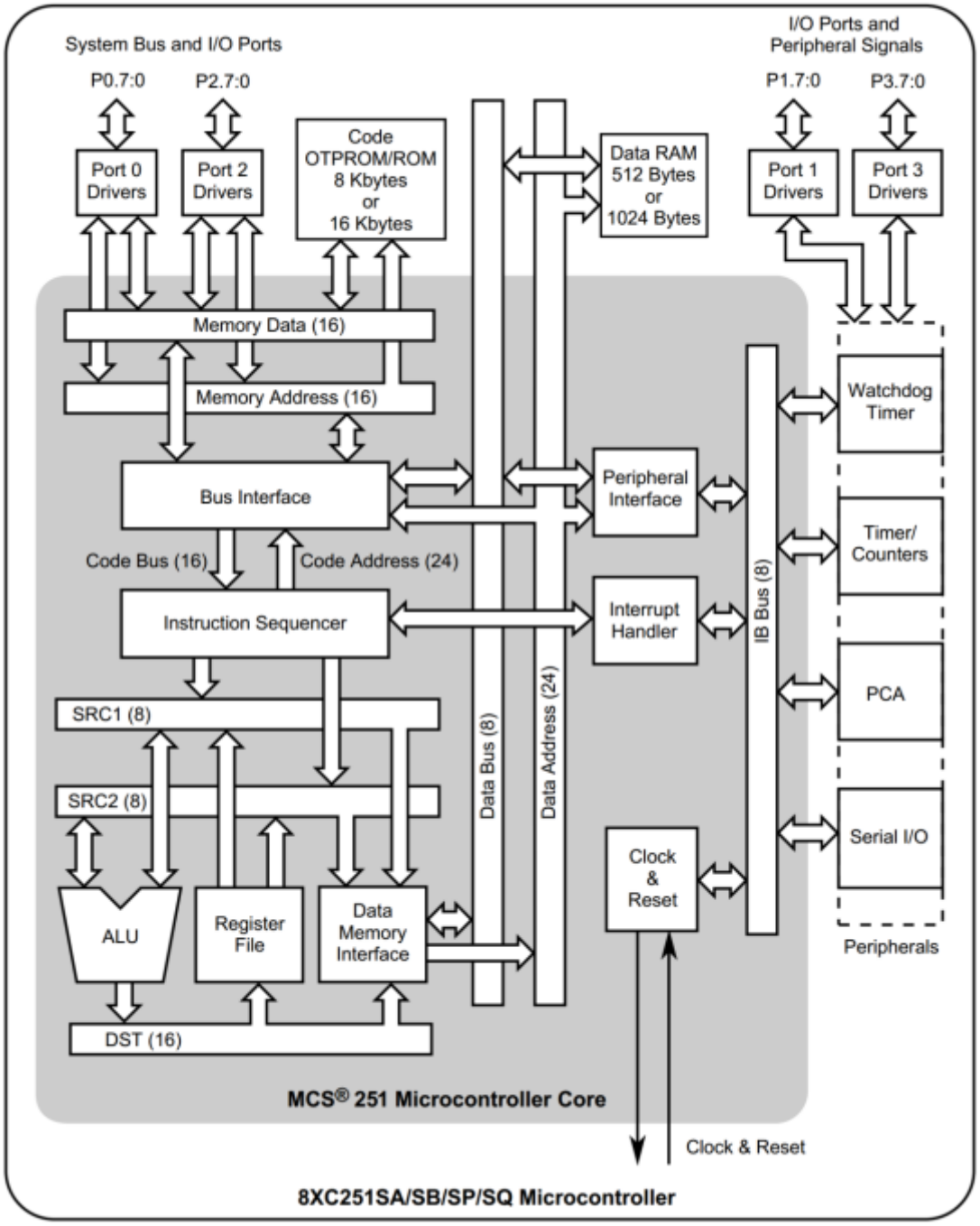

大容量4K
字节SRAM
512/1280
字节
SRAM
8/16/32/64K
字节 闪存
IAP
内置系统
ISP监控程序
INT2/P4.3
INT3/P4.2
P4口可位寻址
E2PROM
Data Flash
IAP
看门狗
集成MAX810
专用复位电路
8051微处理器
STC
低功耗，超低价
高速(0-40M)，高可靠
☑ 掉电模式：<0.1uA
四个8位
并行口
+
P4.0 - P4.6
7个附送
I/O端口
最多39个I/O
P0
P1
P2
P3
P4
单倍速和双倍速
可ISP反复设置
UART(串口)
3个定时器
双数据
指针
4路外部中断
支持掉电唤醒
提升的是性能
降低的是成本

1.1.2 经典单片机 12C5A60S2 组成框图



1.1.3 Intel 80251 内部结构图

Intel 的 80251 芯片内部数据总线宽度是 8 位，但是 32 位指令集，号称 16 位单片机。内部结构图如下：



Intel 80251 读写 32 位 4 个字节的数据，要分 4 次读写 4 个 8 位的字节，效率低。
Intel 80251 读写 16 位 2 个字节的数据，要分 2 次读写 2 个 8 位的字节，效率低。

1.2 数制与编码

数制是人们利用符号进行计数的科学方法。

数制有很多种，常用的数制有：二进制，十进制和十六进制。

进位计数制是把数划分为不同的位数，逐位累加，加到一定数量之后，再从零开始，同时向高位进位。进位计数制有三个要素：数码符号、进位规律和计数基数。下表是各常用数制的总体介绍。

常用的数制	表示符号	数码符号	进制规律	计数基数
二进制	B	0、1	逢二进一	2
十进制	D	0、1、2、3、4、5、6、7、8、9	逢十进一	10
十六进制	H	0、1、2、3、4、5、6、7、8、9、 A、B、C、D、E、F	逢十六进一	16

我们日常生活中计数一般采用十进制。计算机中采用的是二进制，因为二进制具有运算简单，易实现且可靠，为逻辑设计提供了有利的途径、节省设备等优点。为区别于其它进制数，二进制数的书写通常在数的右下方注上基数 2，或加后面加 B 表示。二进制数中每一位仅有 0 和 1 两个可能的数码，所以计数基数为 2。二进制数的加法和乘法运算如下：

0 + 0 = 0　　0 + 1 = 1 + 0 = 1　　1 + 1 = 10

0 × 0 = 0　　0 × 1 = 1 × 0 =0　　1 × 1 = 1

由于二进制数在使用中位数太长,不容易记忆，为了便于描述，又常用十六进制作为二进制的缩写。十六进制通常在表示时用尾部标志 H 或下标 16 以示区别。

1.2.1 数制转换

现在我们来介绍这些常用数制之间的转换。

一：二进制 — 十进制转换

方法：将二进制数按权(如下式)展开，然后将各项的数值按十进制数相加，就得到相应的等值十进制数。

例如：N=(1101.101)B，那么 N 所对应的十进制数时多少呢?

按权展开 $N=1\times2^3+1\times2^2+0\times2^1+1\times2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}=8+4+0+1+0.5+0+0.125=(13.625)D$

二：十进制 — 二进制转换

方法：分两部分进行即整数部分和小数部分。

①整数部分转换(基数除法)：

★ 把我们要转换的数除以二进制的基数(二进制的基数为 2)，把余数作为二进制的最低位；

★ 把上一次得的商在除以二进制基数(即 2)，把余数作为二进制的次低位；

★ 继续上一步,直到最后的商为零,这时的余数就是二进制的最高位.

②小数部分转换(基数乘法)：

★ 把要转换数的小数部分乘以二进制的基数(二进制的基数为 2)，把得到的整数部分作为二进制小数部分的最高位；

★ 把上一步得的小数部分再乘以二进制的基数(即 2)，把整数部分作为二进制小数部分的次高位；

★ 继续上一步，直到小数部分变成零为止。或者达到预定的要求也可以。

例如：将$(213.8125)_{10}$化为二进制数可按如下进行：
先化整数部分：

2	213	余数=1=k_0
2	106	余数=0=k_1
2	53	余数=1=k_2
2	26	余数=0=k_3
2	13	余数=1=k_4
2	6	余数=0=k_5
2	3	余数=1=k_6
2	1	余数=1=k_7
	0	

于是整数部分$(213)_{10}=(11010101)_2$

再化小数部分：

0.8125
× 2
1.6250 ---------- 整数部分=1=k_{-1}

0.6250
× 2
1.2500 ---------- 整数部分=1=k_{-2}

0.2500
× 2
0.5000 ---------- 整数部分=0=k_{-3}

0.5000
× 2
1.0000 ---------- 整数部分=1=k_{-4}

于是小数部分$(0.8125)_{10}=(0.1101)_2$

综上所述，十进制数 $213.8125=(11010101.1101)_2=(11010101.1101)B$

三：二进制 — 十六进制转换

方法：二进制和十六进制之间满足 24 的关系，因此把要转换的二进制从低位到高位每 4 位一组，高位不足时在有效位前面添“0”，然后把每组二进制数转换成十六进制即可。

例如，将(010111011110.10110010)B 转换为十六进制数：

(0101 1101 1110 . 1011 0010)B
↓ ↓ ↓ ↓ ↓
= (5 D E B 2)H

于是，(010111011110.10110010)B=(5DE.B2)H

四：十六进制 — 二进制转换

方法：十六进制转换为二进制时，把上面二进制转换十六进制的过程逆过来，即转换时只需将十六进制的每一位用等值的 4 位二进制代替就行了。

例如：将(C1B.C6)H 转换为二进制数：

(C 1 B . C 6)H
↓ ↓ ↓ ↓ ↓
= (1100 0001 1011 1100 0110)B

于是，(C1B.C6)H=(110000011011.11000110)B

五：十六进制 — 十进制转换

方法：将十六进制数按权(如下式)展开，然后将各项的数值按十进制数相加，就得到相应的等值十进制数。

例如：N=(2A.7F)H，那么 N 所对应的十进制数时多少呢？
按权展开 $N=2\times16^{1}+10\times16^{0}+7\times16^{-1}+15\times16^{-2}=32+10+0.4375+0.05859375$ =(42.49609375)D
于是，(2A.7F)H =(42.49609375)D

六：十进制 — 十六进制转换

方法：将十进制数转换为十六进制数时，可以先将十进制数转换为二进制数，然后再将得到的二进制数转换为等值的十六进制数。

1.2.2 原码、反码及补码

在生活中,数有正负之分,在计算机中是怎样表示数的正负符号呢？

在生活中表示数的时候一般都是把正数前面加一个“+”，负数前面加一个“-”，但是计算机是不认识这些的，通常在二进制数前面增加一位符号位。符号位为“0”表示“+”，符号位为“1”表示“-”。这种形式的二进制数称为原码。如果原码为正数，则原码的反码和补码都与原码相同。如果原码为负数，则将原码(除符号位外)按位取反，所得的新二进制数称为原码的反码，反码加 1 为其补码。

原码、反码、补码这三种形式的总结如下表所示：

	真值	原码	反码	补码
正数	+N	0N	0N	0N
负数	-N	1N	$(2^n-1)+N$	2^n+N

例 1：求+18 和-18 八位原码、反码、补码形式。

真值	原码	反码	补码
+18	00010010	00010010	00010010
-18	10010010	11101101	11101110

1.2.3 常用编码

指定某一组二进制数去代表某一指定的信息，就称为编码。

一：十进制编码

用二进制码表示的十进制数，称为十进制编码。它具有二进制的形式，还具有十进制的特点它可作为人们与数字系统的联系的一种间表示。十进制编码有很多种，最常用的一种是 BCD 码，又称 8421 码。

下面我们用表列出几种常见的十进制编码:

编码种类 / 十进制数	8421 码（BCD 码）	余 3 码	2421 码	5211 码	7321 码
0	0000	0011	0000	0000	0000
1	0001	0100	0001	0001	0001
2	0010	0101	0010	0100	0010
3	0011	0110	0011	0101	0011
4	0100	0111	0100	0111	0101
5	0101	1000	1011	1000	0110
6	0110	1001	1100	1001	0111
7	0111	1010	1101	1100	1000
8	1000	1011	1110	1101	1001
9	1001	1100	1111	1111	1010
权	8421		2421	5211	7321

十进制编码分为有权和无权编码。有权编码码是指每一位十进制数符均用一组四位二进制码来表示，而且二进制码的每一位都有固定权值。无权编码码是指二进制码中每一位都没有固 定的权值。上表中 8421 码(即 BCD 码)、2421 码、5211 码、7321 码都是有权编码，而余 3 码是无权编码。

二：奇偶校验码

在数据的存取、运算和传送过程中，难免会发生错误，把“1”错成“0”或把“0”错成“1”。奇偶校验码是一种能检验这种错误的代码。它分为两部分；信息位和奇偶校验位。有奇数个“1”称为奇校验，有偶数个“1”则称为偶校验。

1.3 几种常用的逻辑运算及其图形符号

逻辑代数中常用的运算有：与(AND)、或(OR)、非(NOT)、与非(NAND)、或非(NOR)、与或非(AND-NOR)、异或(EXCLUSIVE OR)、同或(EXCLUSIVE NOR)等。其中与(AND)、或(OR)、非(NOT)运算时三种最基本的运算。

一：与运算及与门

与运算：决定事件结果的全部条件同时具备时，事件才发生。
逻辑变量 A 和 B 进行与运算时可写成：Y=A・B

真值表		
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

与门：实行与逻辑运算的单元电路。
与门图形符号： 

二：或运算及或门

或运算：决定事件结果的各条件中只要有任何一个满足，事件就会发生。
逻辑变量 A 和 B 进行或运算时可写成：Y=A+B

真值表		
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

或门：实行或逻辑运算的单元电路。
或门图形符号： 

三：非运算及非门

非运算：条件具备时，事件不会发生；条件不具备时，事件才会发生。
逻辑变量 A 进行非运算时可写成：Y=A′

真值表	
A	Y
0	1
1	0

非门：实行非逻辑运算的单元电路。
非门图形符号： 

四：与非运算及与非图形符号

与非运算：先进行与运算，然后将结果求反，最后得到的即为与非运算结果。
逻辑变量 A 和 B 进行与非运算时可写成：Y=(A・B)′

真值表		
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

与非图形符号： 

五：或非运算及或非图形符号

或非运算：先进行或运算，然后将结果求反，最后得到的即为或非运算结果。
逻辑变量 A 和 B 进行或非运算时可写成：Y=(A+B)′

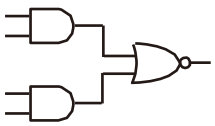
真值表		
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

或非图形符号： 

六：与或非运算及与或非图形符号

与或非运算：在与或非逻辑运算中有 4 个逻辑变量 A、B、C、D。假设 A 和 B 为一组，C 和 D 为一 组，A、B 之间以及 C、D 之间都是与的关系，只要 A、B 或 C、D 任何一组同时为 1， 输出 Y 就是 0。只有当每一组输入都不全是 1 时，输出 Y 才是 1。
逻辑变量 A 和 B 进行或非运算时可写成：Y=(A・B+C・D)′

真值表				
A	B	C	D	Y
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

与或非图形符号： 

七：异或运算及异或图形符号

异或运算：当 A、B 不同时，输出 Y 为 1；而当 A、B 相同时，输出 Y 为 0。 逻辑变量 A 和 B 进行异或运算时可写成：$Y = A \oplus B = (A \cdot B') + (A' \cdot B)$

真值表		
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

异或图形符号： 

八：同或运算及同或图形符号

同或运算：当 A、B 不同时，输出 Y 为 0；而当 A、B 相同时，输出 Y 为 1。 逻辑变量 A 和 B 进行同或运算时可写成：$Y = A \odot B = (A \cdot B)+(A' \cdot B')$

真值表		
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

同或图形符号： 

2 集成开发环境、ISP 下载软件及实验箱

2.1 下载并安装 Keil 集成开发环境

2.1.1 下载 Keil 集成开发环境

登录 Keil 官网，下载最新版的 C251 和 C51 编译器，下载链接如下：

https://www.keil.com/download/product/

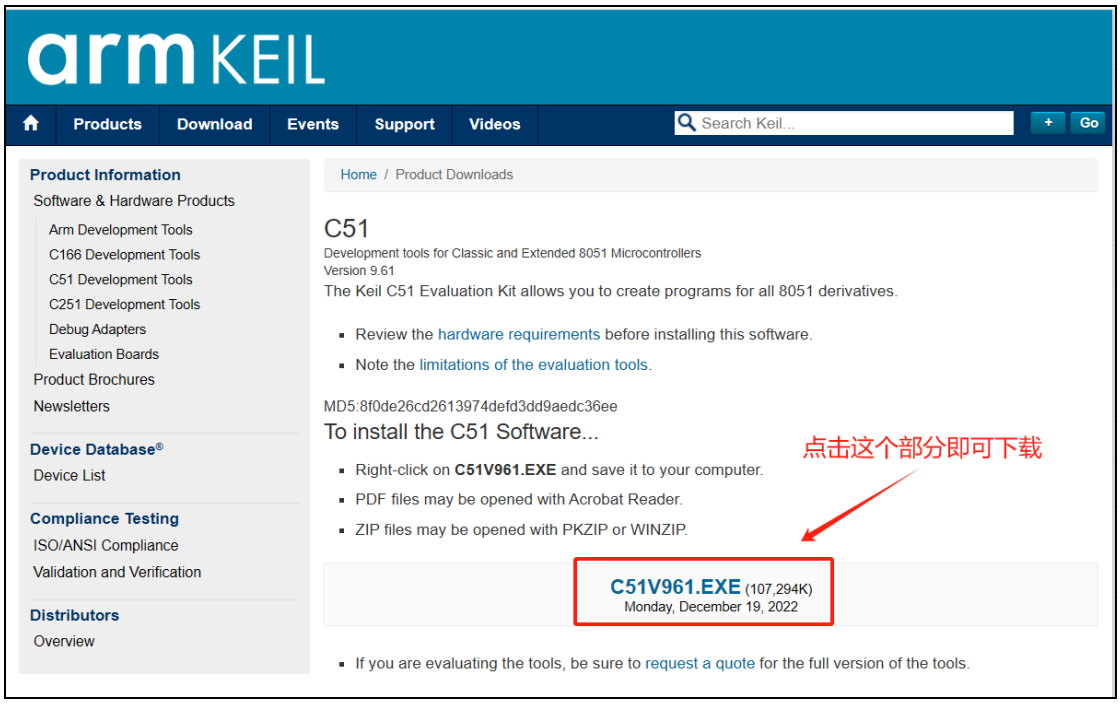
进入下载页面，我们可以看到 C51 和 C251 的下载链接，这里将这两个安装包都下载到桌面上



如下载 C51 编译器，则点击 C51 下载图标，进入如下界面，输入联系信息，注意只有 E-mail 拼写检查较严谨，必须用类似 c@qq.com 这样的格式，其他任意字母填写。然后点击 Submit。



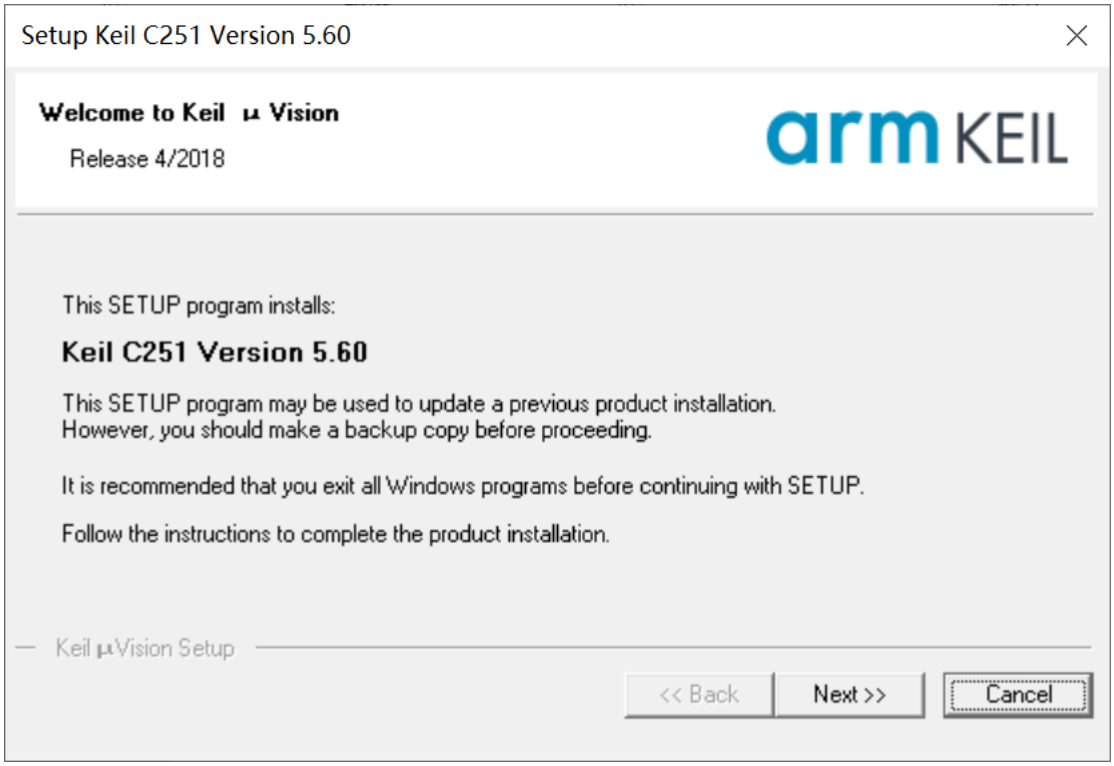
出现下载 C51V961.EXE 的界面/或更高版本的界面，点击粗体蓝色字即可下载 C51 安装包。同样步骤下载 C251 安装包。



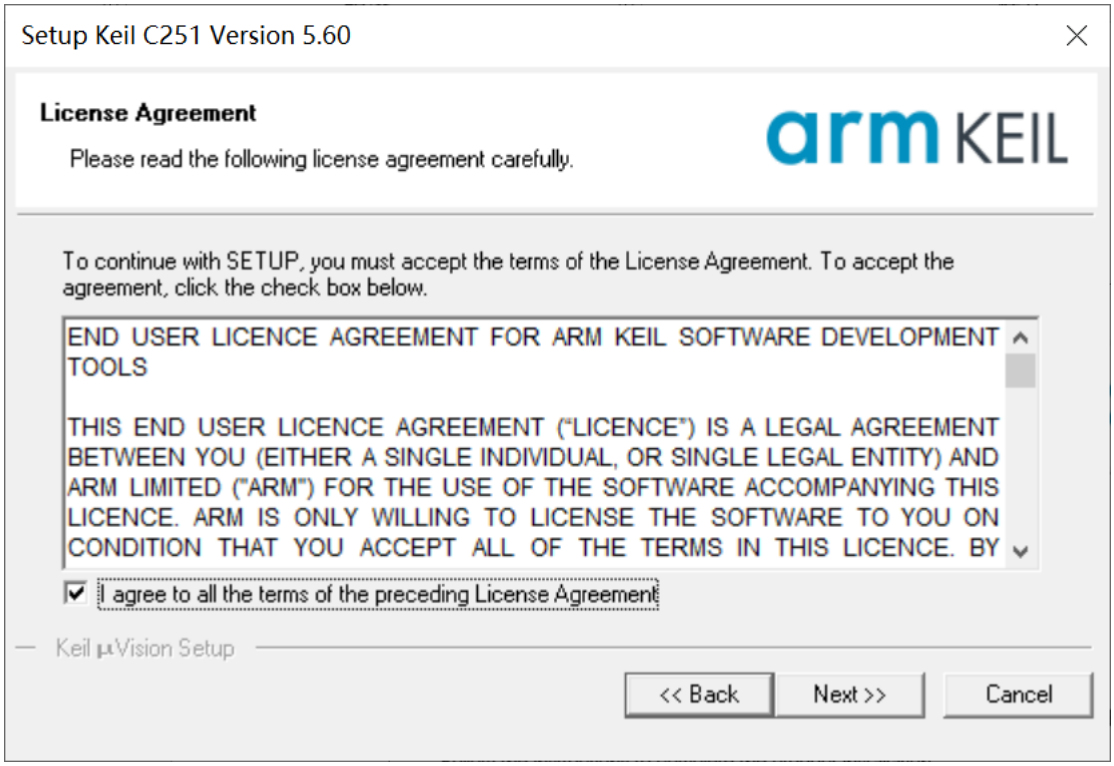
2.1.2 安装 Keil 集成开发环境

2.1.2.1 安装 Keil C251 集成开发环境

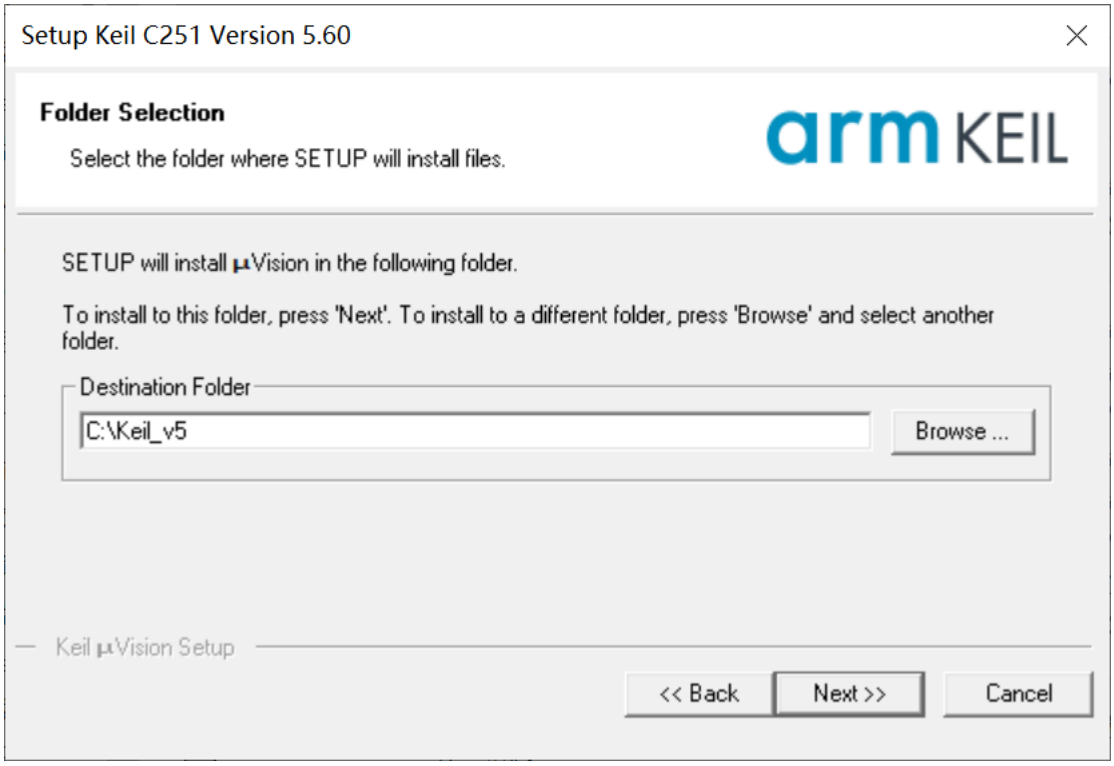
双击下载的安装包开始安装，点击“Next”：



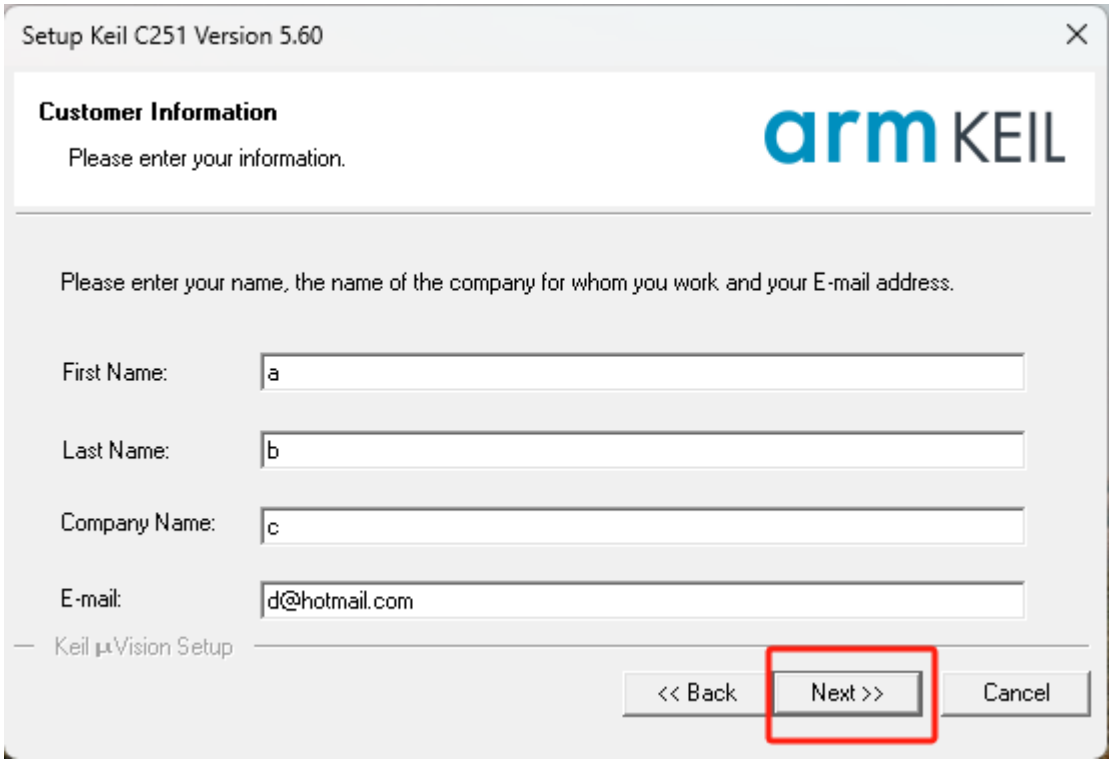
勾选“I agree to all the terms of the preceding License Agreement”,然后点击“Next”：



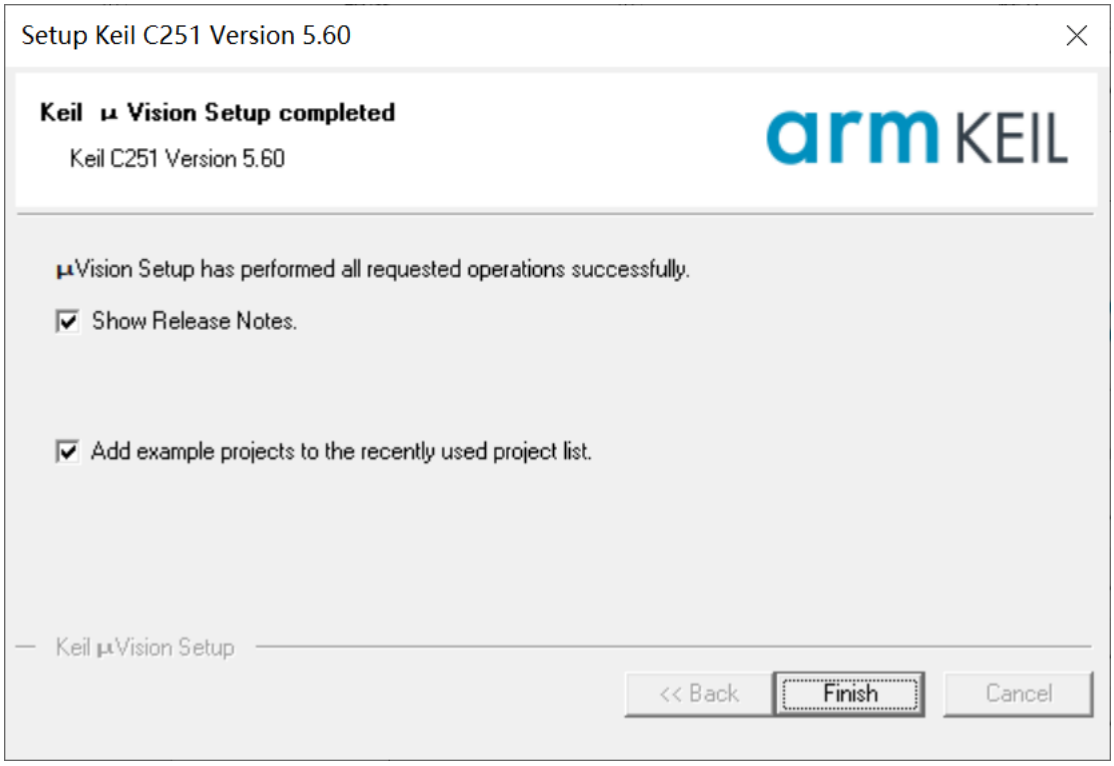
选择安装目录，然后点击“Next”：



填写个人信息，然后点击“Next”：

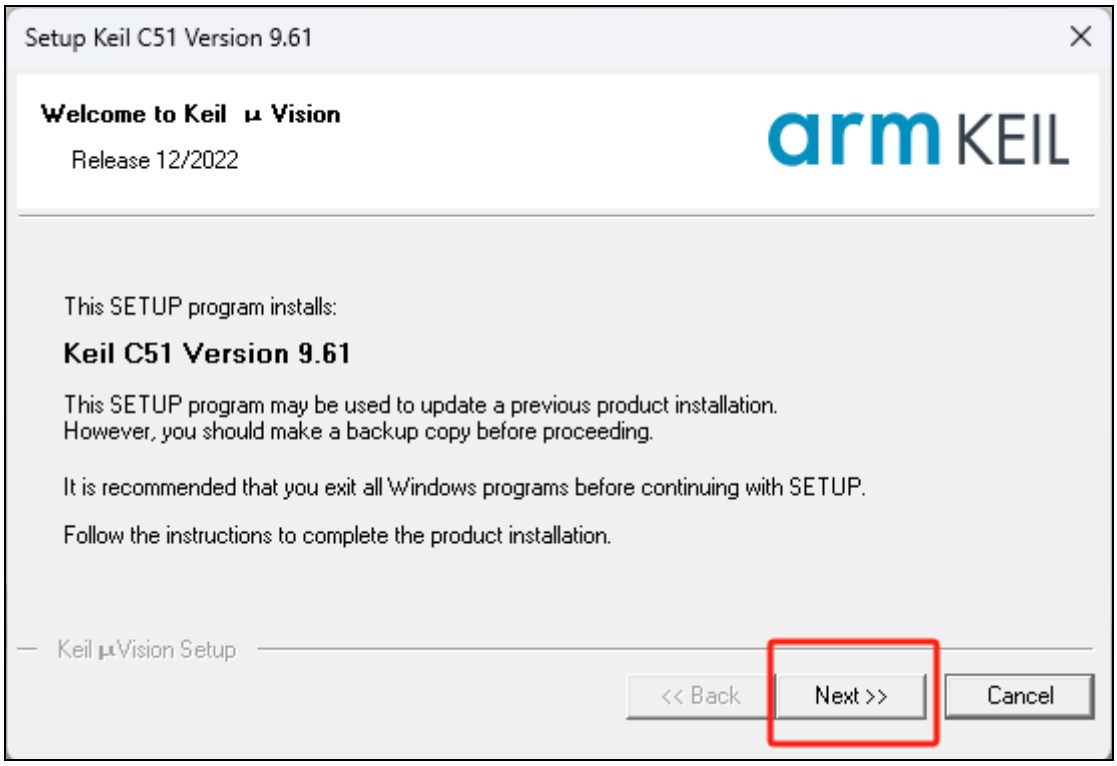


安装完成，点击“Finish”结束。

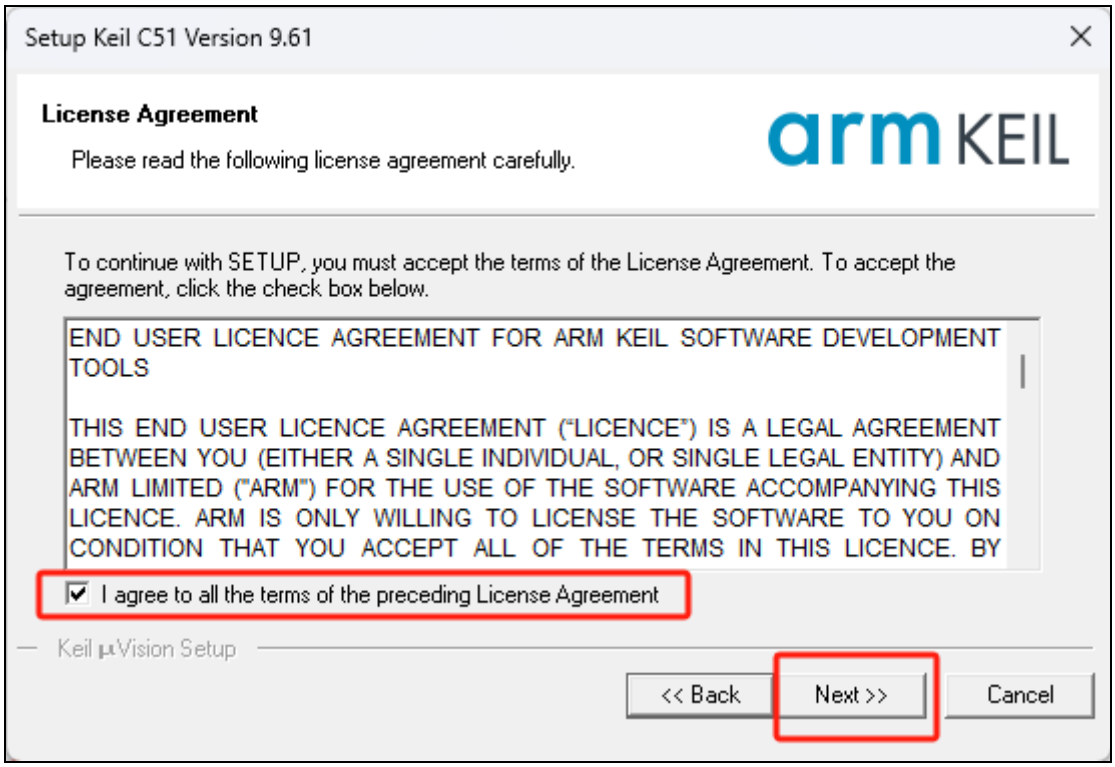


2.1.2.2 安装 Keil C51 集成开发环境

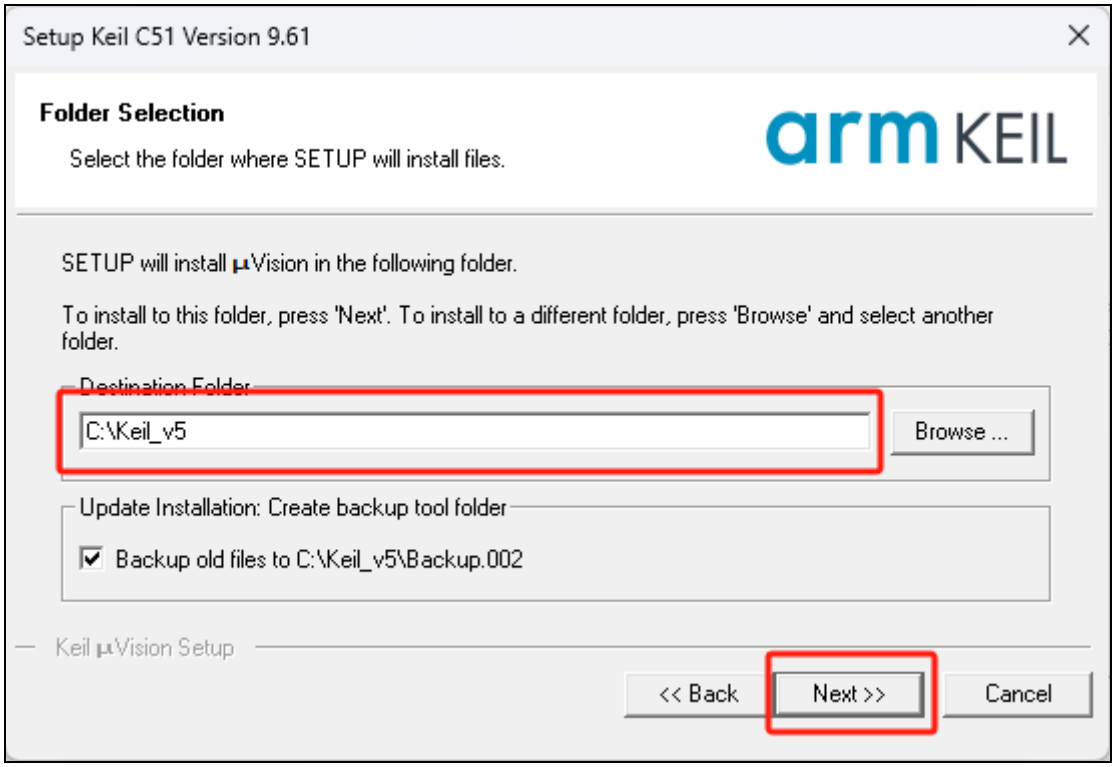
双击下载的安装包开始安装，点击“Next”：



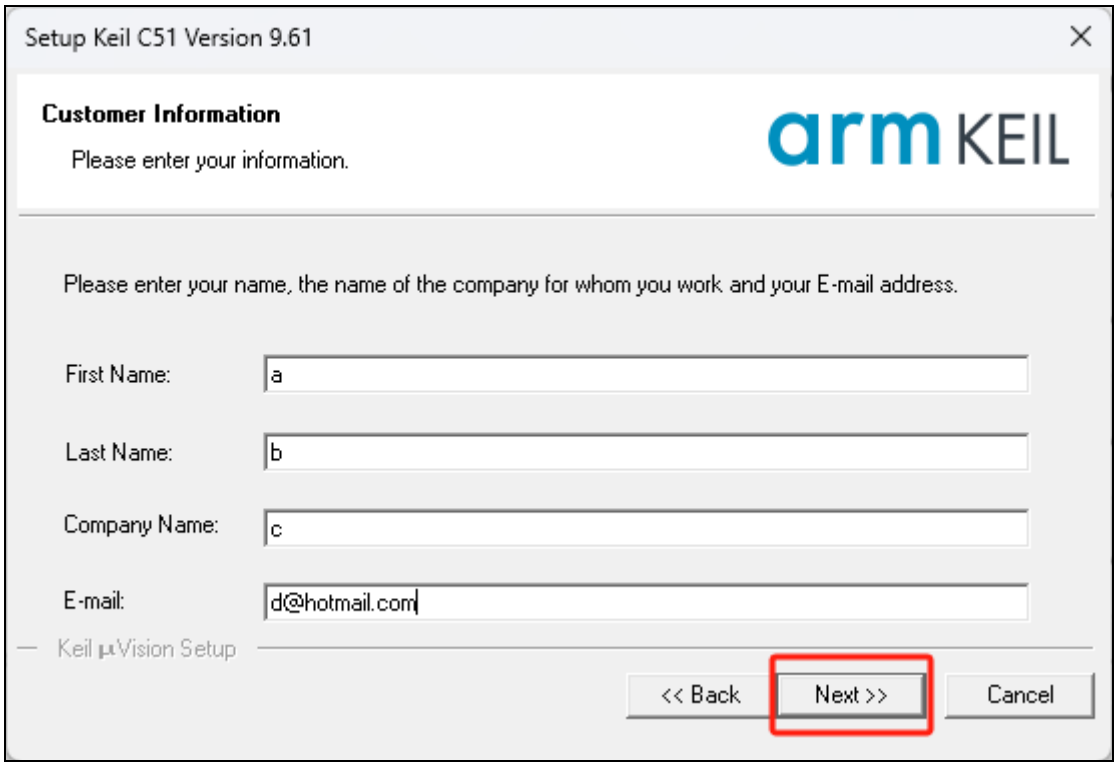
勾选“I agree to all the terms of the preceding License Agreement”,然后点击“Next”：



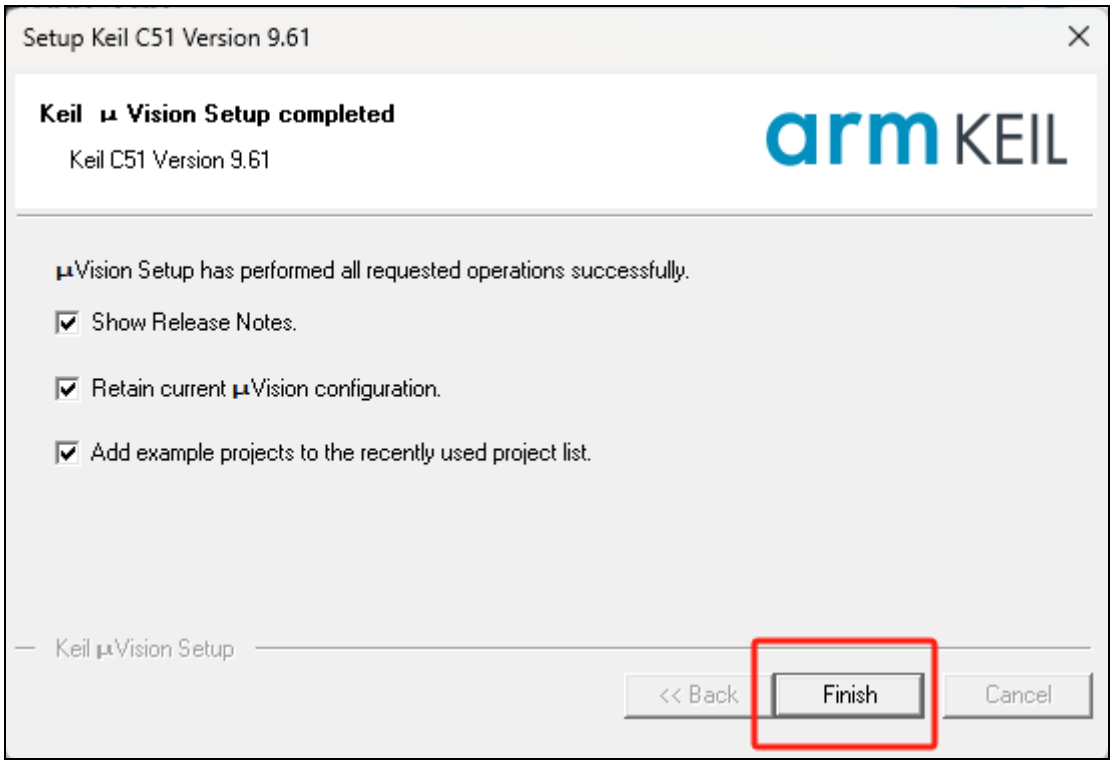
选择安装目录，然后点击“Next”：



填写个人信息，然后点击“Next”：

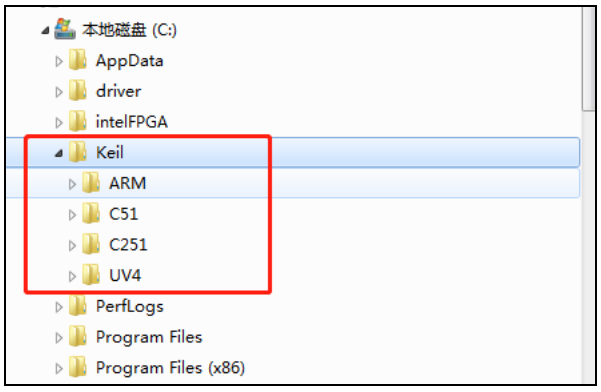


安装完成，点击“Finish”结束。

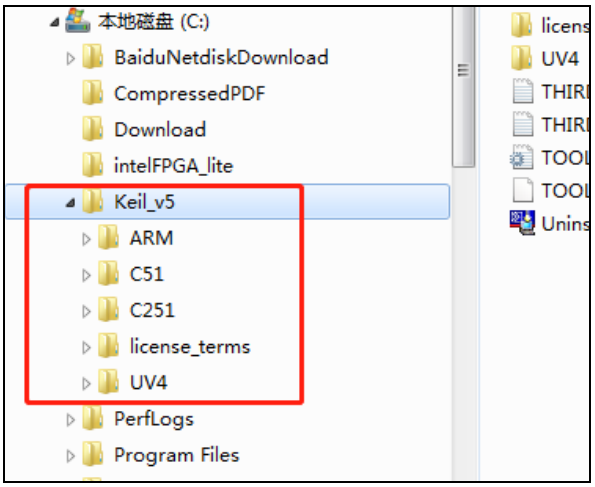


2.1.2.3 Keil 的 C51、C251 和 MDK 可安装在同一台电脑同一个目录

旧版本的 Keil 软件的安装目录默认是 C:\Keil，C51、C251 和 MDK 分别会被安装在 C:\Keil 目录下的 C51、C251 和 ARM 目录中，如下图所示。



新版本的 Keil 软件的安装目录默认是 C:\Keil_v5，C51、C251 和 MDK 分别会被安装在 C:\Keil_v5 目录下的 C51、C251 和 ARM 目录中，如下图所示。



无论是新版本还是旧版本，C51、C251 和 MDK 是安装在不同的目录，并不会有冲突。软件的和谐也是 3 个软件分别进行的，之前已经安装完成并设置好的软件，并不会因为后续有安装新的软件而改变。所以安装时只需要按照默认方式安装即可，Keil 软件会自动处理好。

2.1.3 从哪购买 KEIL，获取无限制版 KEIL

可以 BAIDU:
方法 1, BAIDU：从哪购买 KEIL，获取无限制版 KEIL
方法 2, BAIDU：如何购买 KEIL C251
方法 3, BAIDU：如何购买 KEIL C51
方法 4, BAIDU：如何获取无限制版 KEIL



2.2 安装 AiCube-ISP 下载/编程/烧录 工具,含强大的辅助开发工具

2.2.1 安装 AiCube-ISP 下载/编程/烧录 工具软件，取代 STC-ISP

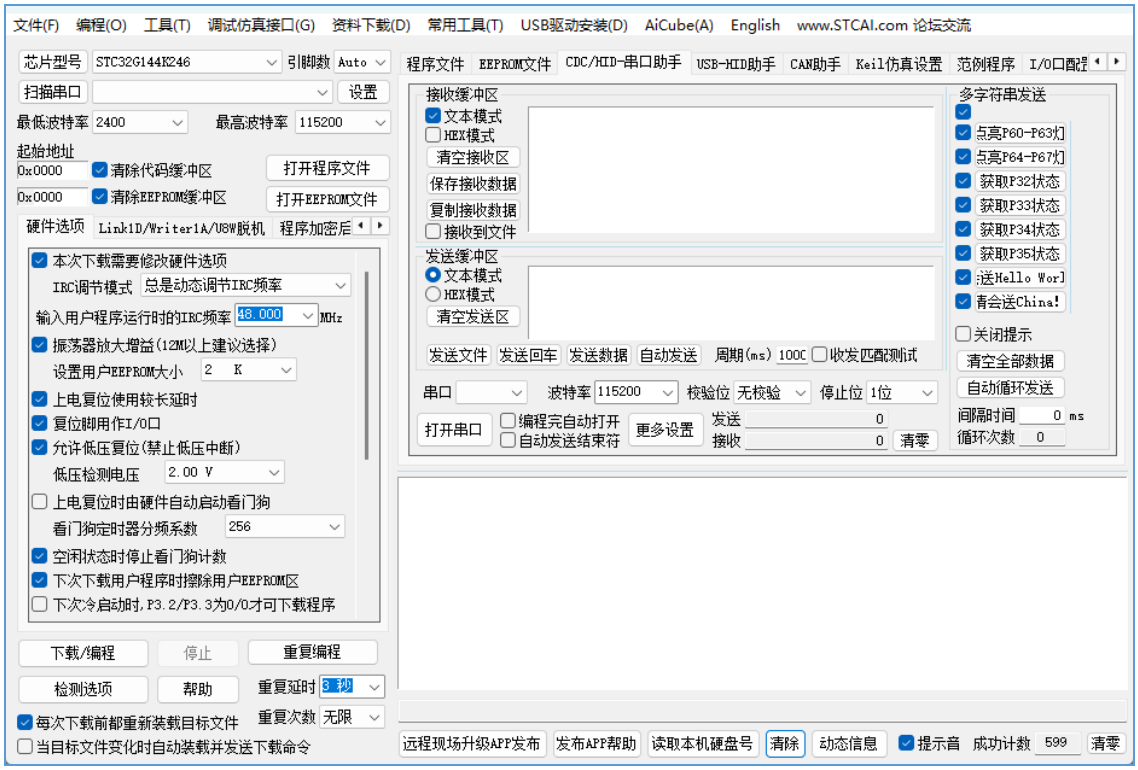
安装 STCAI 公司研发的 AiCube-ISP 下载 / 编程 / 烧录 / 软件，含强大的辅助开发工具包。登录 STCAI 官网，下载最新版的 AiCube-ISP 安装包，下载链接如下：https://www.stcai.com/gjrj
下载 AiCube-ISP 压缩包到桌面：



下载下来的是压缩包文件及解压缩文件如下图：



如果是解压缩到非桌面的其他目录，如 D:\STCAI-ISP，建议创建快捷方式到桌面方便打开。这实际是一个免安装的绿色软件。
双击图标，即可打开 AiCube-ISP 软件界面，即可直接使用，如下图：

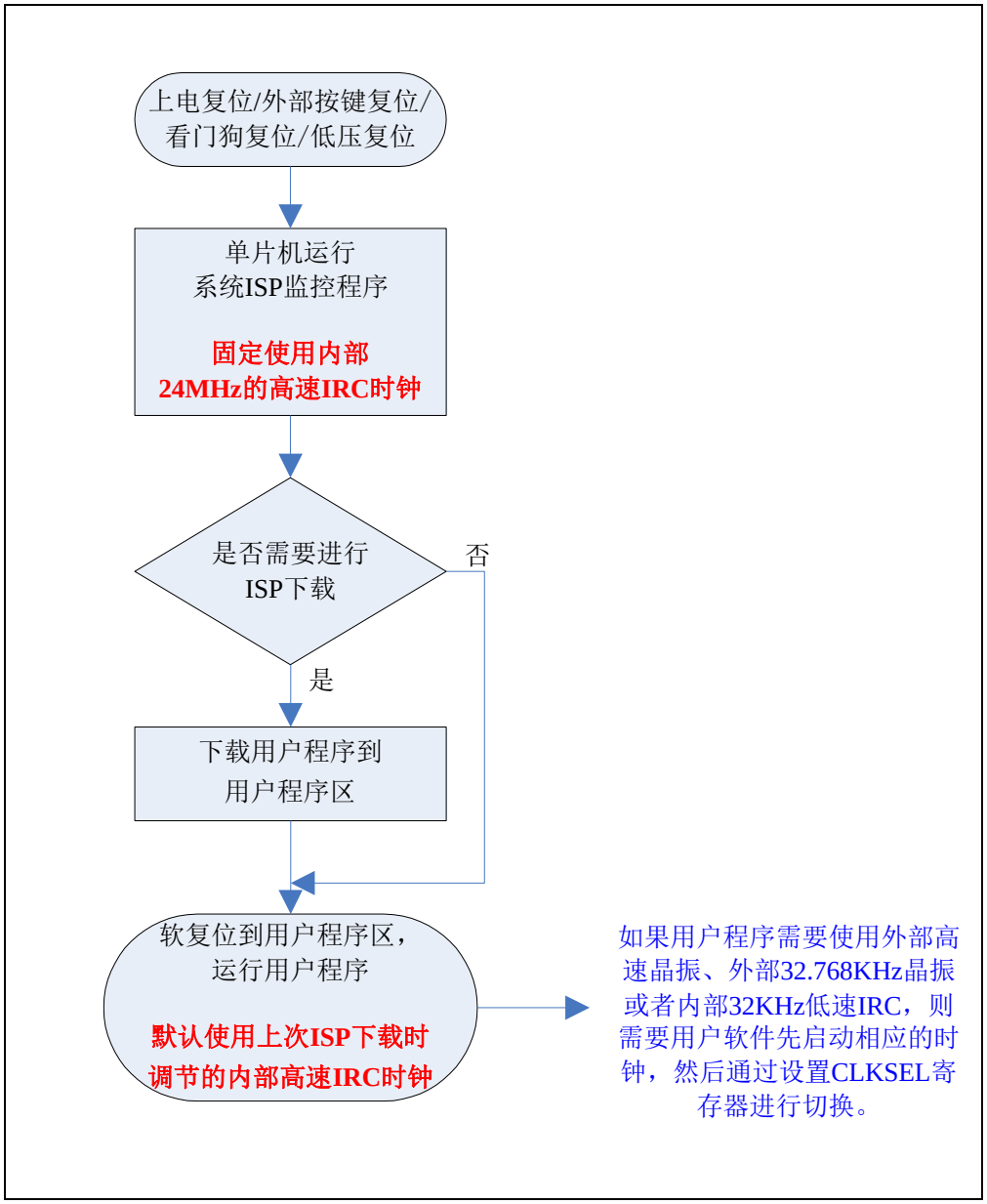


2.2.2 STCAI 单片机上电工作过程

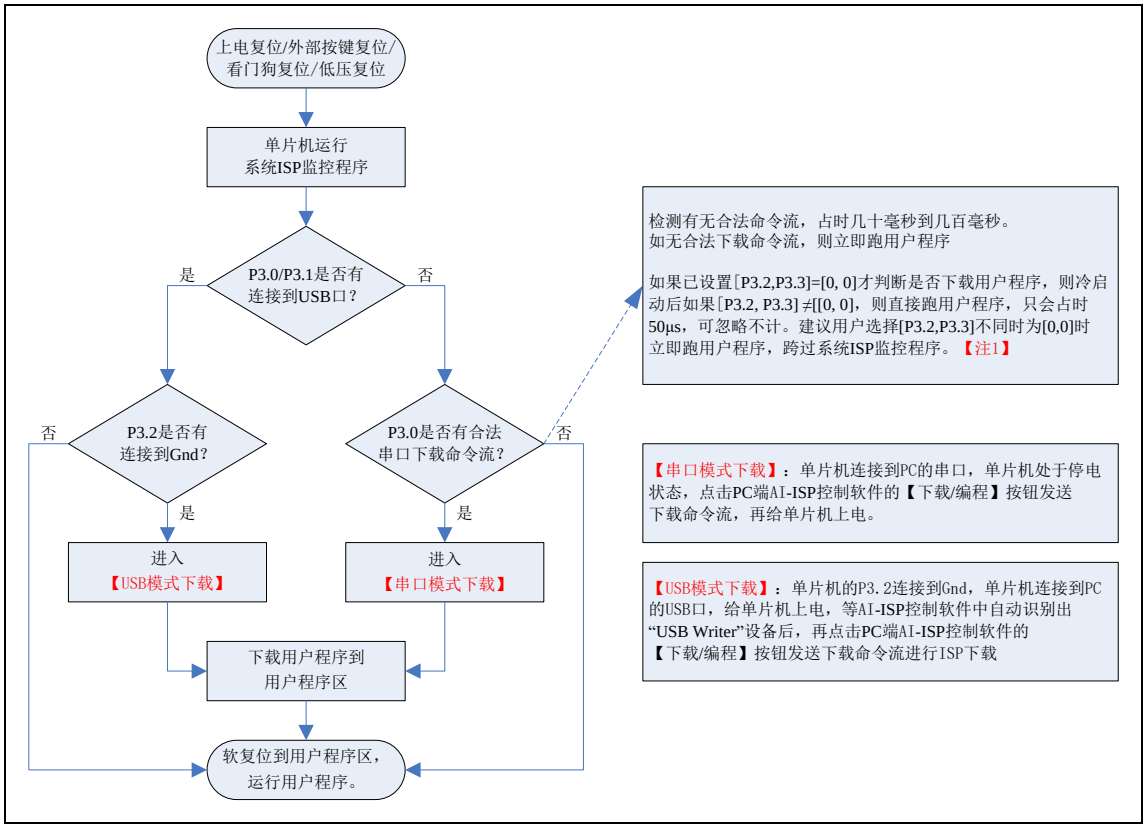
上电复位/复位脚复位/看门狗复位/低压检测复位时，芯片默认从 ISP 系统程序区开始执行系统程序代码，此时固定使用内部 24MHz 的高速 IRC 时钟。

如发现需要下载用户程序且下载完成后复位到用户程序区或者不需要下载直接复位到用户程序区时，默认会使用上次用户下载时所调节的高速 IRC 时钟，如果用户程序需要使用外部高速晶振、外部 32.768KHz 晶振或者内部 32KHz 低速 IRC，则需要用户软件先启动相应的时钟，然后通过设置 CLKSEL 寄存器进行切换。

启动流程如下：



2.2.3 STC32G144K246 的 ISP 下载流程图（硬件 USB+串口模式）



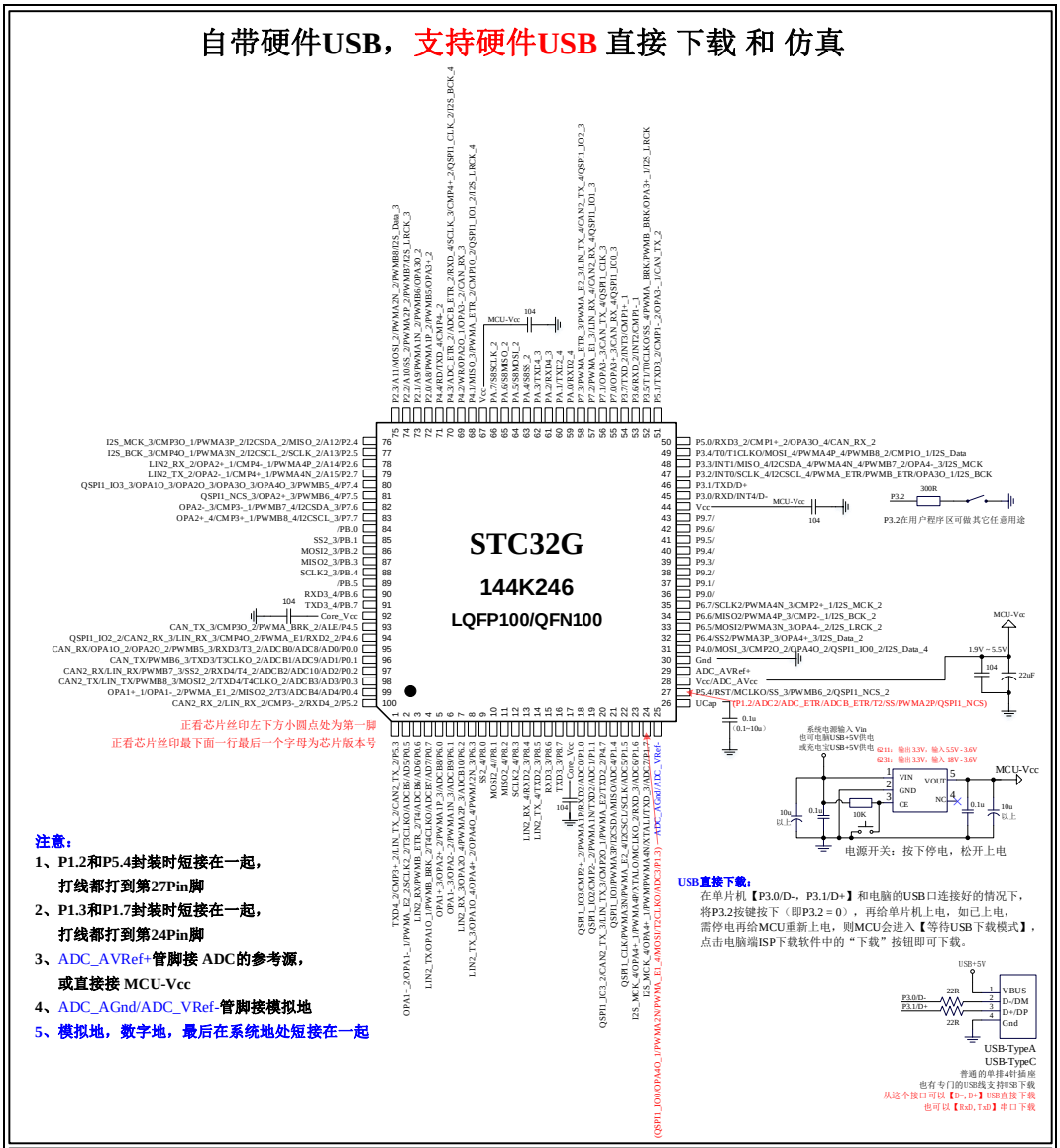
注意：因 [P3.0, P3.1] 作下载/仿真用(下载/仿真接口仅可用 [P3.0, P3.1])，故建议用户将串口 1 放在 P3.6/P3.7 或 P1.6/P1.7 ，若用户不想切换，坚持使用 P3.0/P3.1 工作或作为串口 1 进行通信，则务必在下载程序时，在软件上勾选“下次冷启动时，P3.2/P3.3 为 0/0 时才可以下载程序”。【注 1】

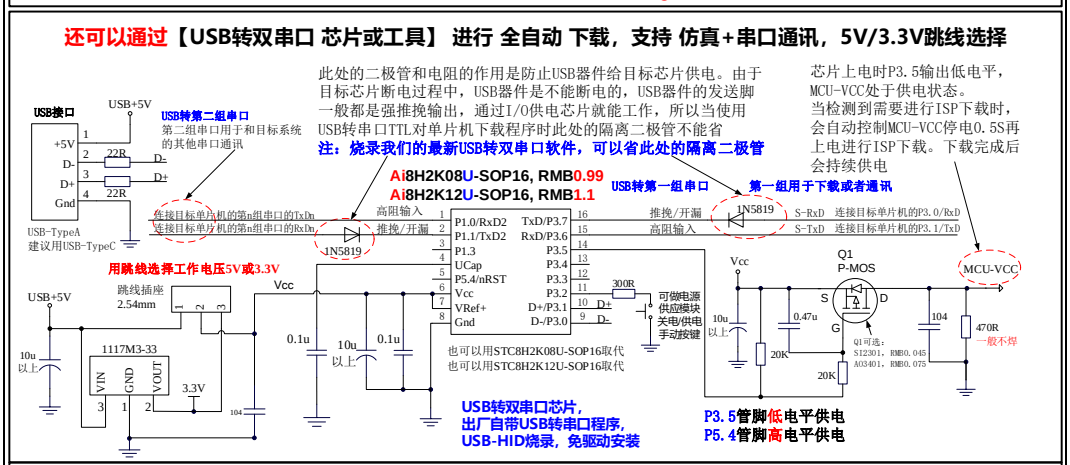
【注 1】：AI15，AI8 系列及以后新出的芯片的烧录保护引脚为 P3.2/P3.3，之前早期芯片的烧录保护引脚为 P1.0/P1.1。

AiCube-ISP 界面如下图所示：



2.2.4 STC32G144K246 的 USB 直接下载电路及 ISP 下载操作步骤





现在带硬件 USB 的 MCU 支持用硬件 USB 下载，因为用的是 USB-HID 通信协议，不需要安装任何驱动。只要 USB 鼠标、USB 键盘能工作，USB-HID 驱动就是好的，不要安装 USB-HID 驱动，免驱。
在 D-/P3.0，D+/P3.1 与 PC-USB 端口连接好的状况下，USB-ISP 下载程序有如下三种模式：

【USB 下载方法一，P3.2 按键，再结合停电上电下载】
1、按下板子上的 P3.2/INT0 按键，就是 P3.2 接地
2、给目标芯片重新上电，不管之前是否已通电。
===电子开关是按下停电后再松开就是上电
等待电脑端 ISP 下载软件中自动识别出“(HID1) USB Writer”后，就与 P3.2 状态无关了，这时可以松开 P3.2 按键（P3.2 在用户程序区可做其它任意用途）
===传统的机械自锁紧开关是按上来停电，按下去是上电
3、点击电脑端下载软件中的【下载/编程】按钮（注意：USB 下载与串口下载的操作顺序不同）
下载进行中，几秒钟后，提示下载成功！

【USB 下载方法二，复位管脚低电平复位下载】
USB 连接好并已上电的情况下，外部按键复位也可进入 USB 下载模式，注意：
P5.4-nRST 出厂时默认是 P5.4-I/O 功能，要改为复位功能，需 ISP 烧录时取消设置复位脚用作 I/O 口，停电一次再上电才生效，程序区中用户程序也可改为复位脚或 I/O，这个立即生效。
1、按下 P5.4-nRST 外接的低电平复位按键复位 MCU，
松开复位键，MCU 从系统程序区启动，判断是否要下载用户程序，
等待电脑端 ISP 下载软件中自动识别出“(HID1) USB Writer”后
2、点击电脑端下载软件中的【下载/编程】按钮
下载进行中，几秒钟后，提示下载成功！
备注：如果目标 MCU 的用户程序区已有 USB 通讯程序（含 CDC、HID 等），则通过复位脚复位进入 USB 下载模式不被支持，只能通过停电上电冷启动下载或者从用户程序区软复位到系统区下载

【USB 下载方法三，从用户程序区软复位到系统区下载】
USB 连接好并已上电的情况下，从用户程序区软复位到系统区也可进入 USB 下载模式
1、在用户程序区运行软复位到系统区的程序，就是 IAP_CONTR 寄存器送 60H
等待电脑端 ISP 下载软件中自动识别出“(HID1) USB Writer”后
2、点击电脑端下载软件中的【下载/编程】按钮
下载进行中，几秒钟后，提示下载成功！

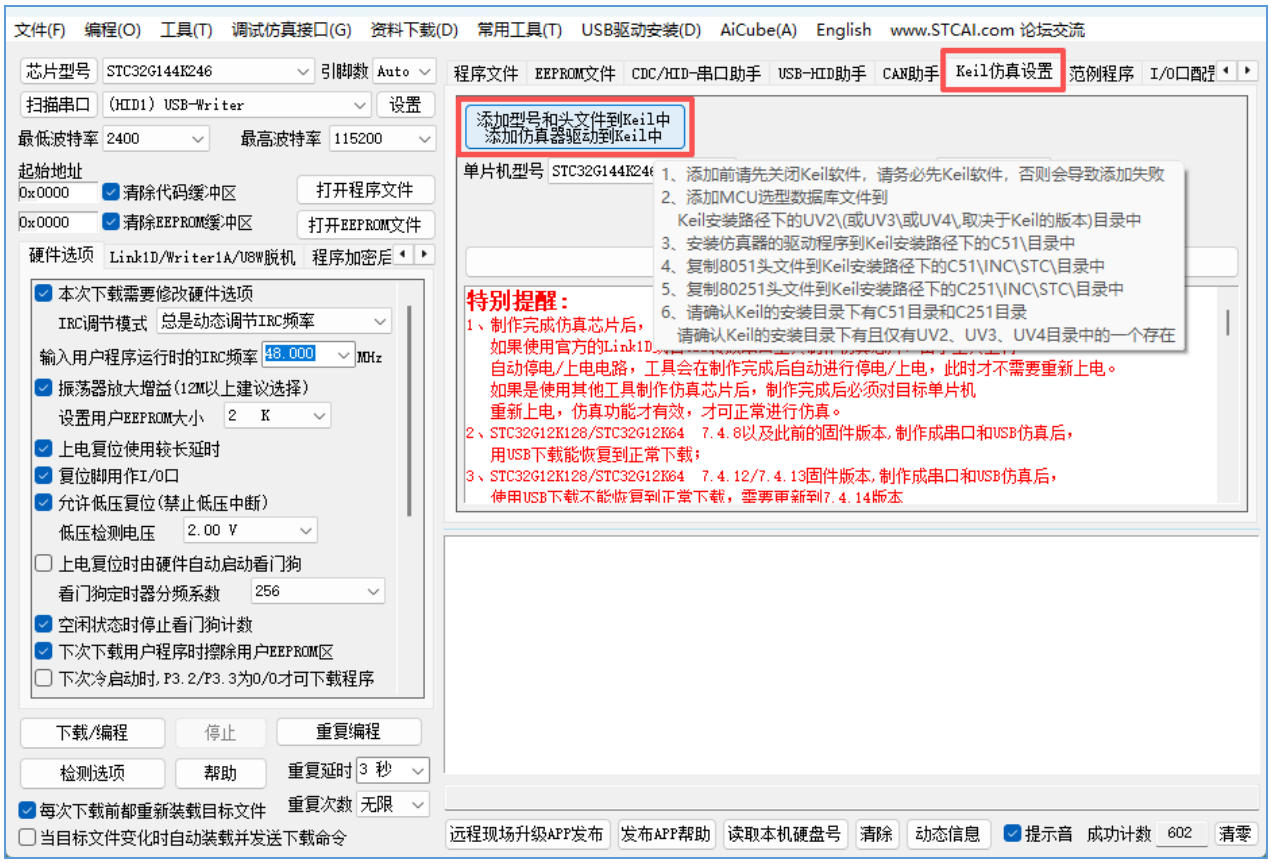
USB 下载 注意事项：

拔插 USB 插头不能代替上面线路图中的电源开关。正常操作步骤如下：
USB 的【Gnd，D+，D-】接好的情况下，按下 P3.2 按键接地，再通过正常的电源开关给 MCU 供电或重新供电，让 MCU 冷启动进入系统程序区，判断是否需要等待电脑端 USB 下载程序。
拔插 USB 插头代替电源开关不能每次都能成功进行 USB 下载的原因：
拔插 USB 插头，如【Gnd，USB+5V】已接触好，已供电，而【D+，D-】有一个甚至两个信号线还没有接触好，MCU 已上电，开始跑系统区程序时，发现 USB 还没接触好，则会从系统区软复位到用户程序区跑用户程序，不再进入等待 USB 下载模式，本次就无法顺利进行 USB 下载。
很多人经过多次插拔 USB，才能碰到 1 次【D+，D-】接触好的情况下，【Gnd，USB+5V】才开始接触好，才开始供电，才能成功进入 USB 下载。插 USB 插头代替供电，不能保证【Gnd，D+，D-，USB+5V】的接触顺序，所以，必须使用正常的电源开关，才能确保每次下载都能成功。

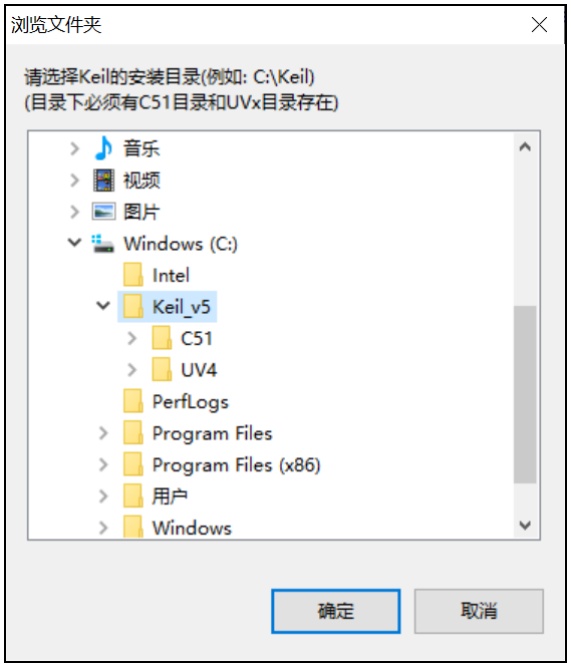
2.3 添加型号和头文件到 Keil

使用 Keil 之前需要先安装 仿真驱动。仿真驱动的安装步骤如下：

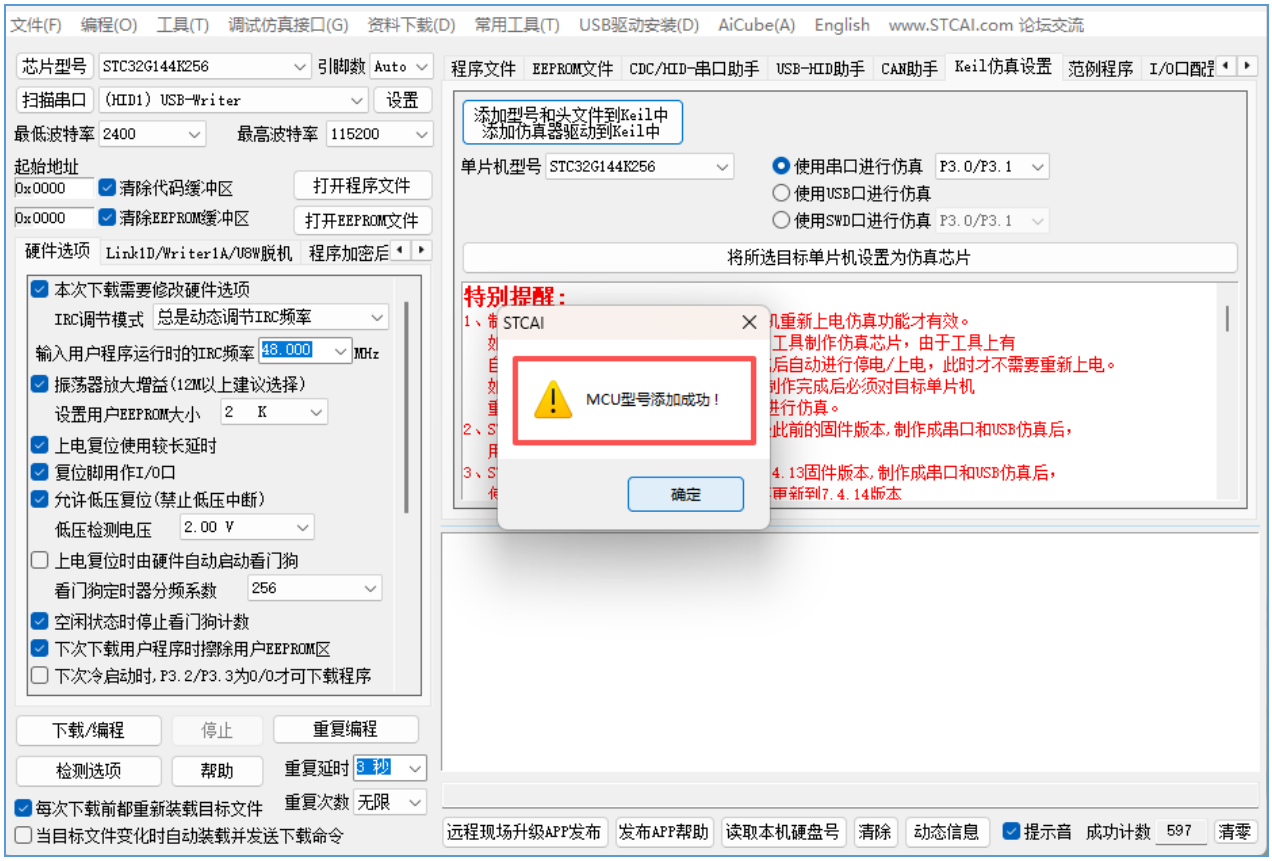
首先打开 ISP 下载软件，然后在软件右边功能区的“Keil 仿真设置”页面中点击“添加型号和头文件到 Keil 中 添加仿真器驱动到 Keil 中”按钮：



按下后会出现如下画面：



将目录定位到 Keil 软件的安装目录，然后确定。安装成功后会弹出如下的提示框：



即表示驱动正确安装了

头文件默认复制到 Keil 安装目录下的“Keil\C51\INC\STC”目录和“Keil\C251\INC\STC”目录中
在 C 代码中使用“#include <STC32G144K246.H>”或者“#include "STC32G144K246.H"”均可正确使用

2.4 STC 单片机程序中头文件的使用方法

c 语言中 include 用法

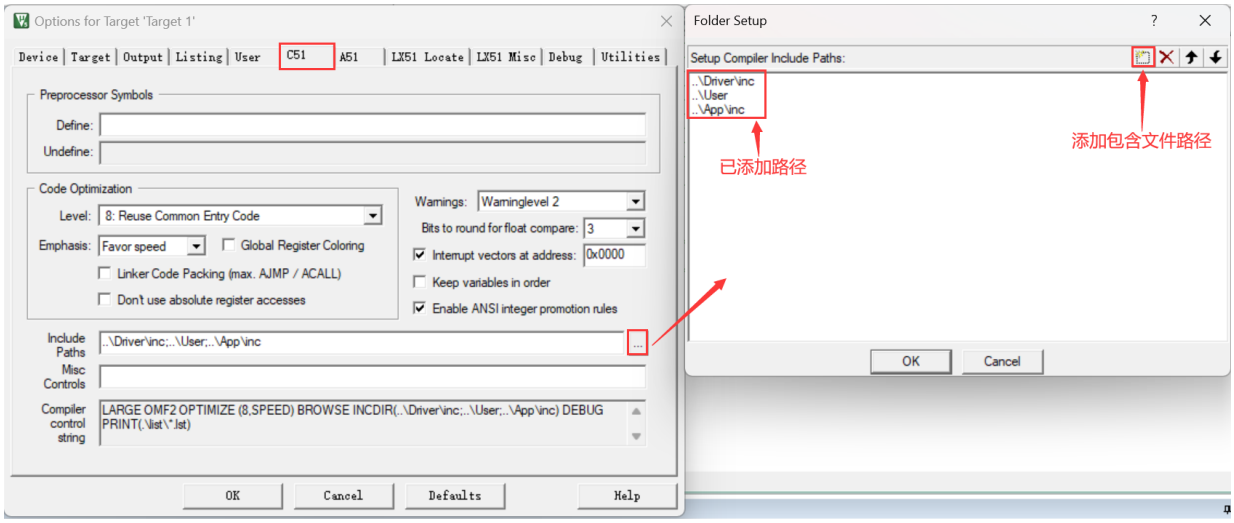
#include 命令是预处理命令的一种，预处理命令可以将别的源代码内容插入到所指定的位置。
有两种方式可以指定插入头文件：

#include <文件名.h>
#include "文件名.h"

使用尖括号< >和双引号" "的区别在于头文件的搜索路径不同：
使用尖括号< >，编译器会到系统路径下查找头文件；
使用双引号" "，编译器首先在当前目录下查找头文件，如果没有找到，再到系统路径下查找。

路径设置方式 1：

通过 keil 设置界面，添加包含文件的路径：



添加后，调用时直接使用 #include "文件名.h" 就可以将需要的文件包含进来，编译器会自动到以上路径下面寻找所包含的文件。
这种情况下，使用双引号" "包含头文件，编译器首先在当前目录下查找头文件，如果没有找到，编译器会到 keil 设置路径查找，还没有的话再到系统路径下查找。（注：系统路径是编译器安装位置存放头文件的目录）

路径设置方式 2：

在包含文件名前添加绝对路径，例如：

#include "E:\xxxx\xxxx\文件名.h"
#include "E:/xxxx/xxxx/文件名.h"

路径设置方式 3：

在包含文件名前添加相对路径，例如：

#include "..\comm\文件名.h"
#include "../comm/文件名.h"

其中 ".."是指上一级目录，以上路径是指包含文件在当前目录的上一级目录的 comm 目录下面。

汇编语言中 include 用法与 c 语言类似，将"#"换成"$"，用小括号()包含文件：

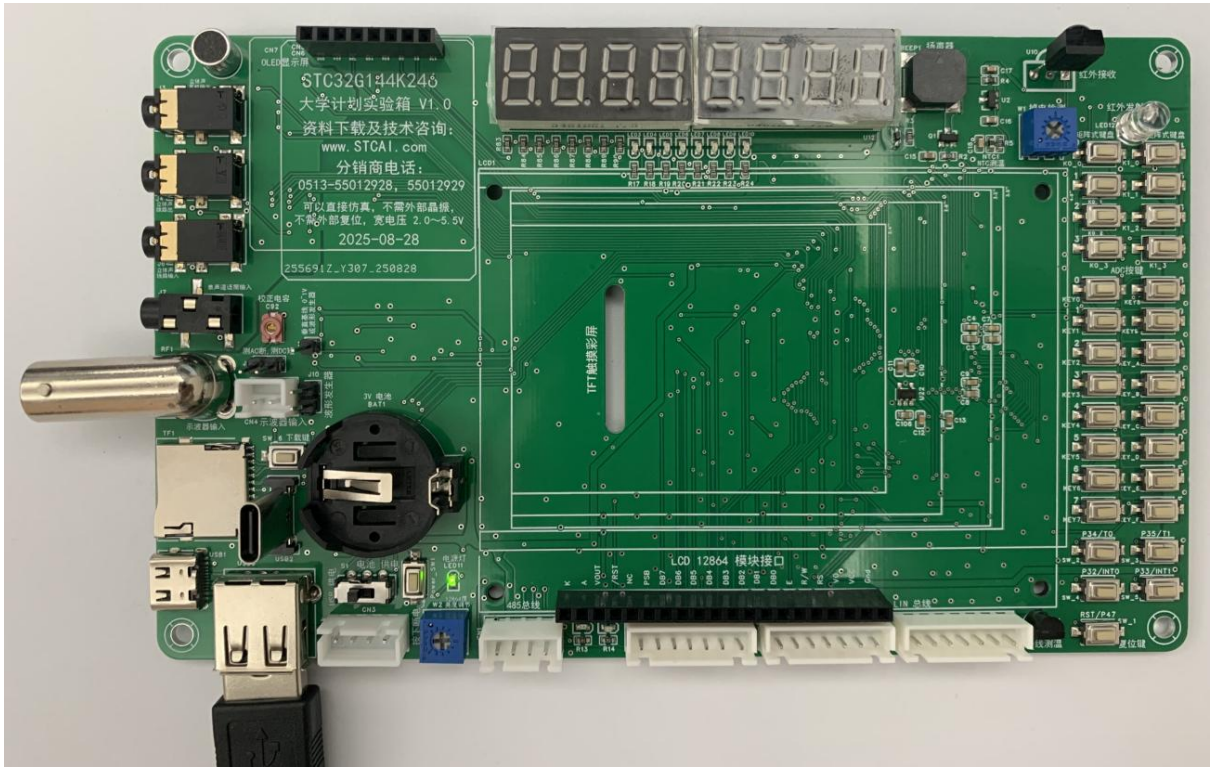
$include (../../comm/ STC32G144K246.INC)

以上指令表示要包含的文件 STC32G144K246.INC，在当前目录的上一级目录的上一级目录的 comm 目录下面。

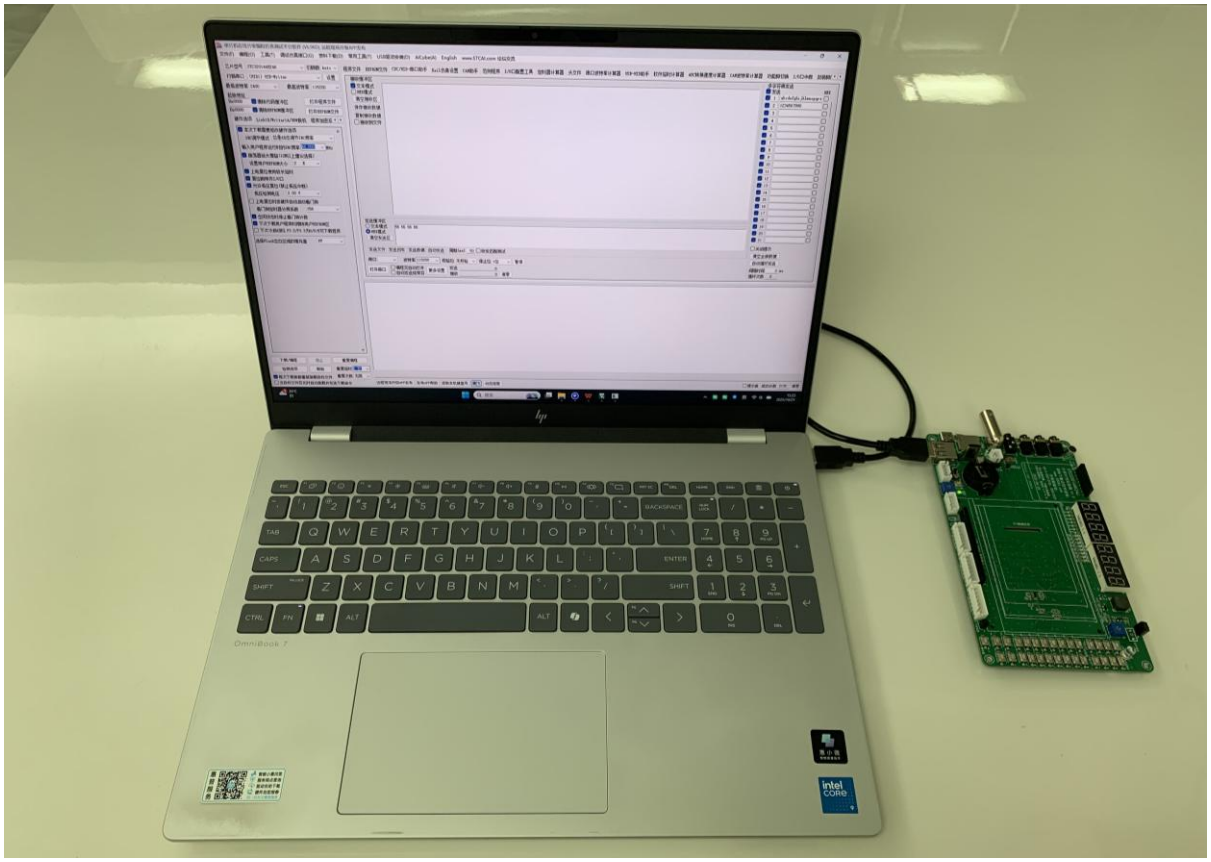
2.5 STC32G144K246 实验箱实物图及程序下载流程

2.5.1 STC32G144K246 实验箱实物图及与电脑连接实物图

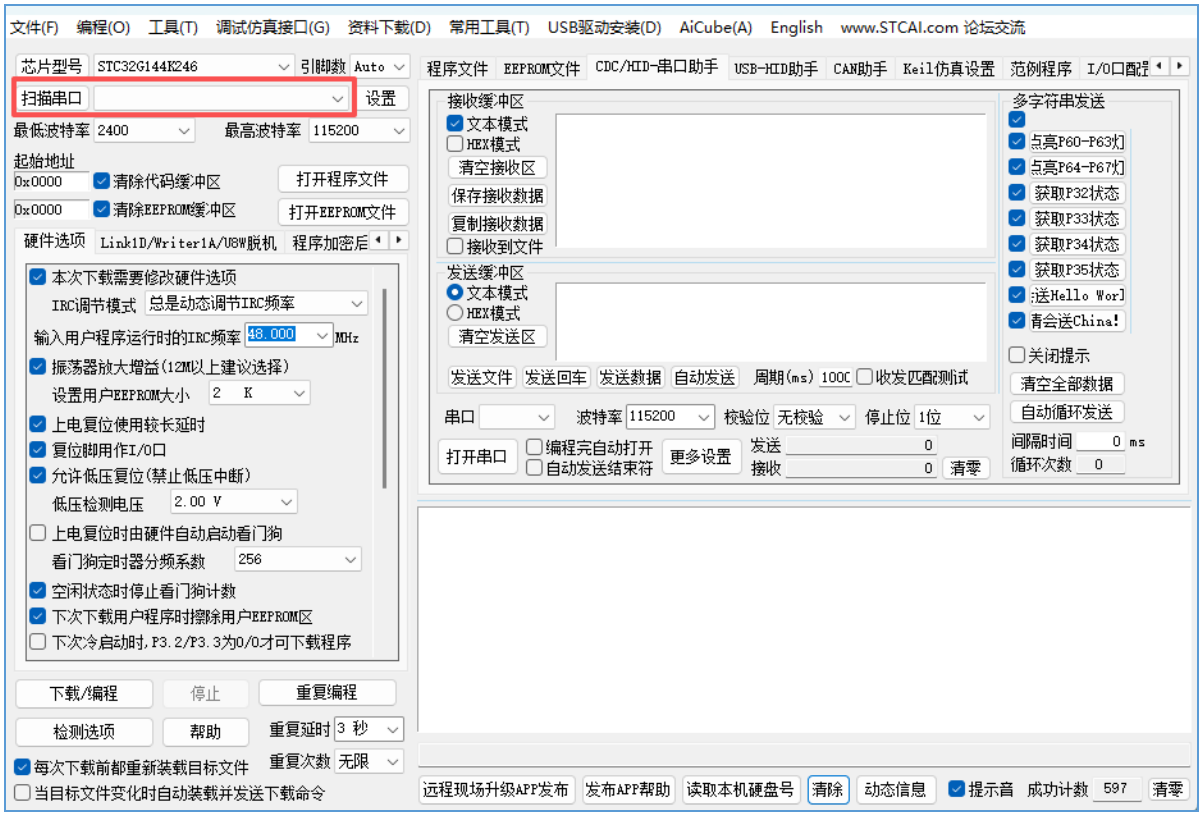
➲ STC32G144K246 实验箱实物图：正面



➲ STC32G144K246 实验箱与电脑连线实物图：USB-TypeA 到 USB-TypeA 线将实验箱与电脑连接。

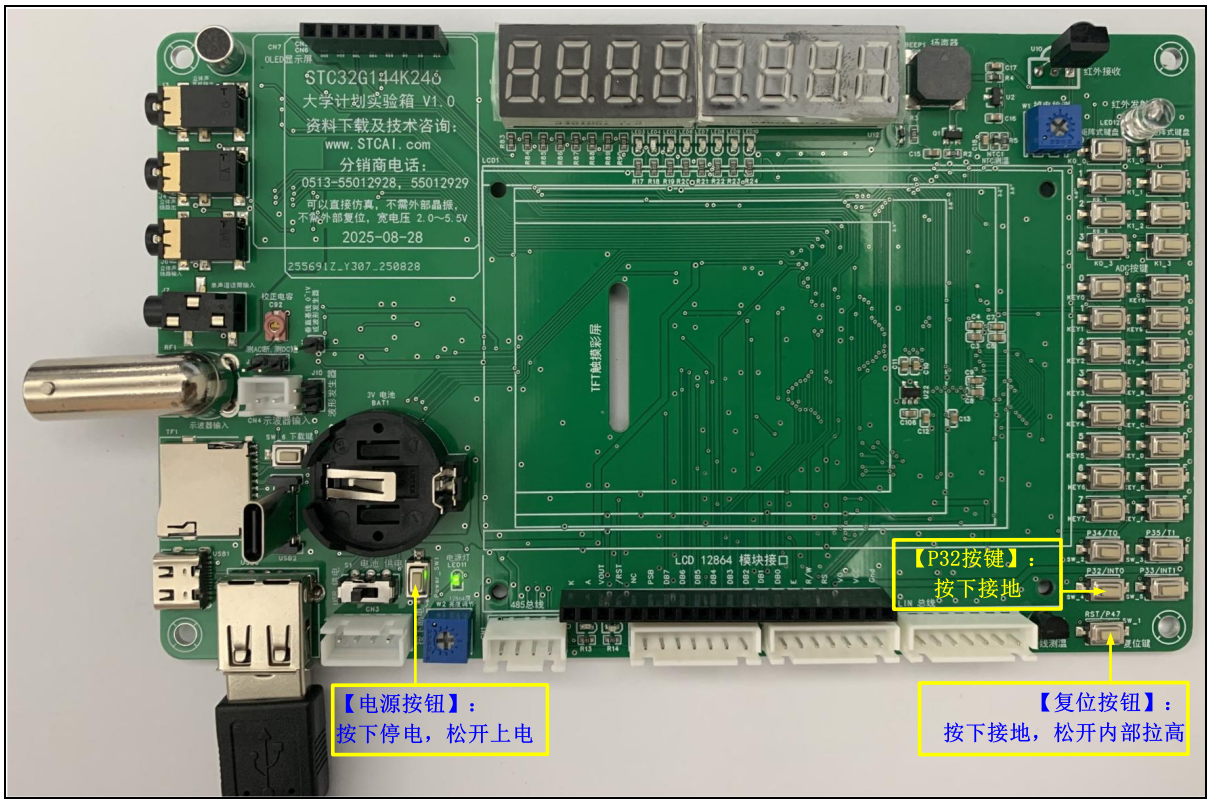


现在运行 AiCube-ISP-v6.96D 或以上版本。选择单片机型号：STC32G144K246，如下图，我们看到“扫描串口”后面的端口列表框不是“（HID1）USB-Writer”，说明还不能下载程序到实验箱。

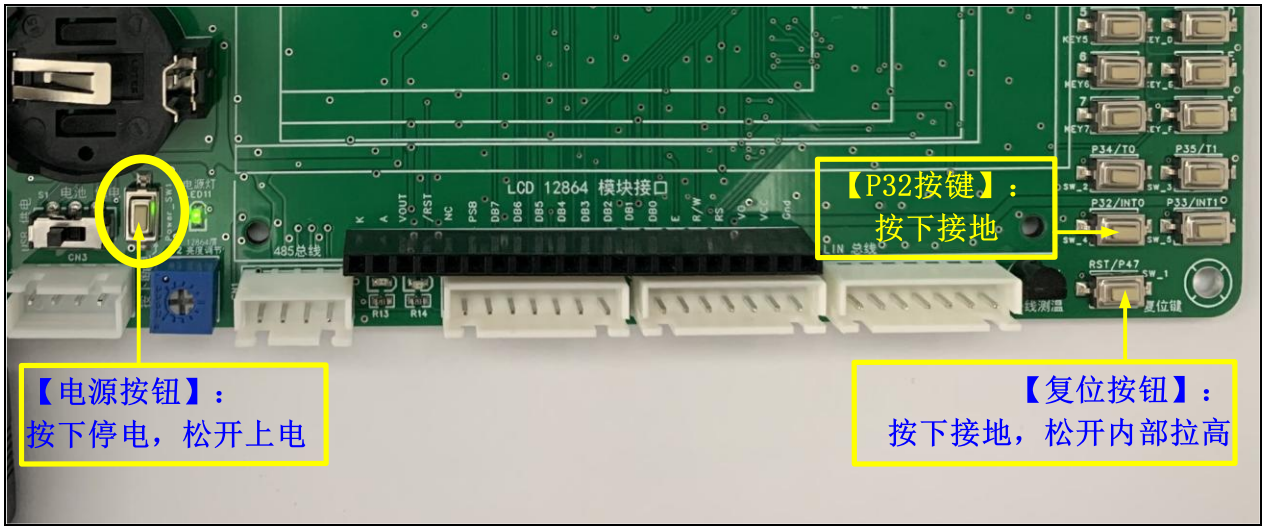


2.5.2 硬件 USB 下载用户程序步骤

下图是 STC32G144K246 实验箱已经和电脑相连的正面高清实物图：



局部放大实验箱图片，找到 P3.2 按键和 Power_SW 按键，如下图：



【方法一，P3.2 按键，再结合停电上电下载】

1、按下板子上的 P3.2 按键，就是 P3.2 接地

2、给目标芯片重新上电，不管之前是否已通电。

===本 STC32G144K246 实验箱是：按下电源按钮停电，再松开电源按钮上电，

这时电脑端 ISP 下载软件会自动识别出“(HID1) USB Writer”，此时就与 P3.2 状态无关了，可以松开 P3.2 按键（P3.2 在用户程序区可做其它任意用途）

3、点击电脑端下载软件中的【下载/编程】按钮（注意：USB 下载与串口下载的操作顺序不同）

电脑端 ISP 下载软件提示：下载进行中…，几秒钟后，提示下载成功！

【方法二：复位管脚低电平复位下载】

USB 连接好并已上电的情况下，外部按键复位也可进入 USB 下载模式，注意：

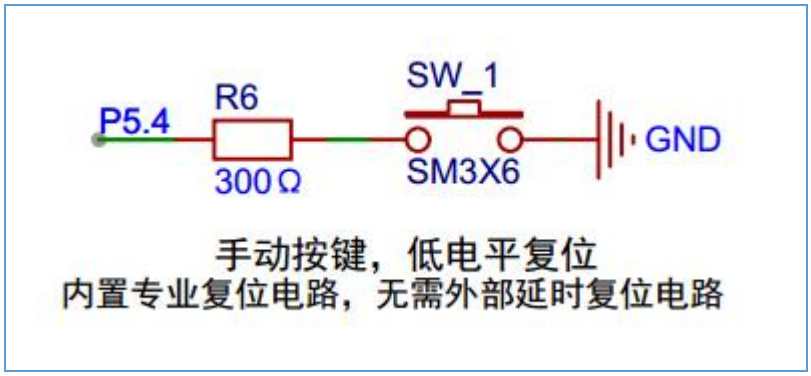
P5.4-nRST 出厂时默认是 P5.4-I/O 功能，要改为复位功能，需 ISP 烧录时取消设置复位脚用作 I/O 口，停电一次再上电才生效，程序区中用户程序也可改为复位脚或 I/O，这个立即生效。

1、按下复位管脚/P5.4-nRST 外接的低电平复位按键，复位 MCU，再松开复位键，MCU 首先从系统程序区启动运行系统程序，判断是否要下载用户程序，

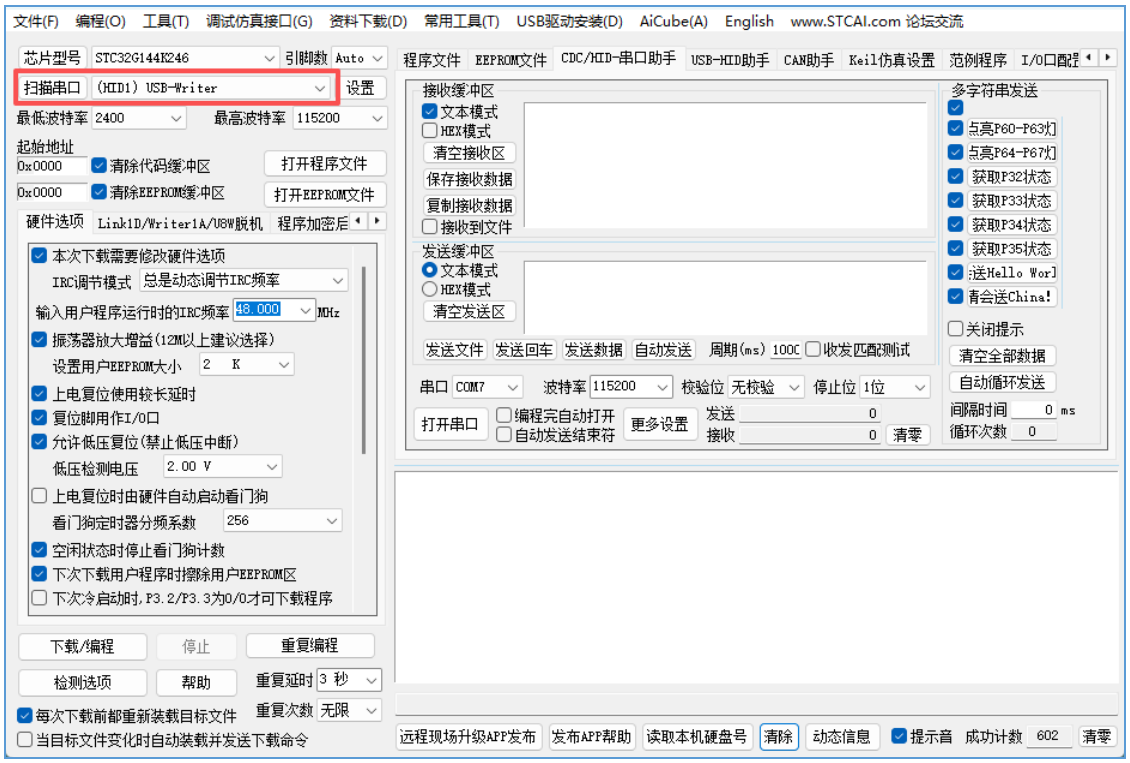
等待电脑端 ISP 下载软件中自动识别出“(HID1) USB Writer”后

2、点击电脑端下载软件中的【下载/编程】按钮

电脑端 ISP 下载软件提示：下载进行中…，下载成功！



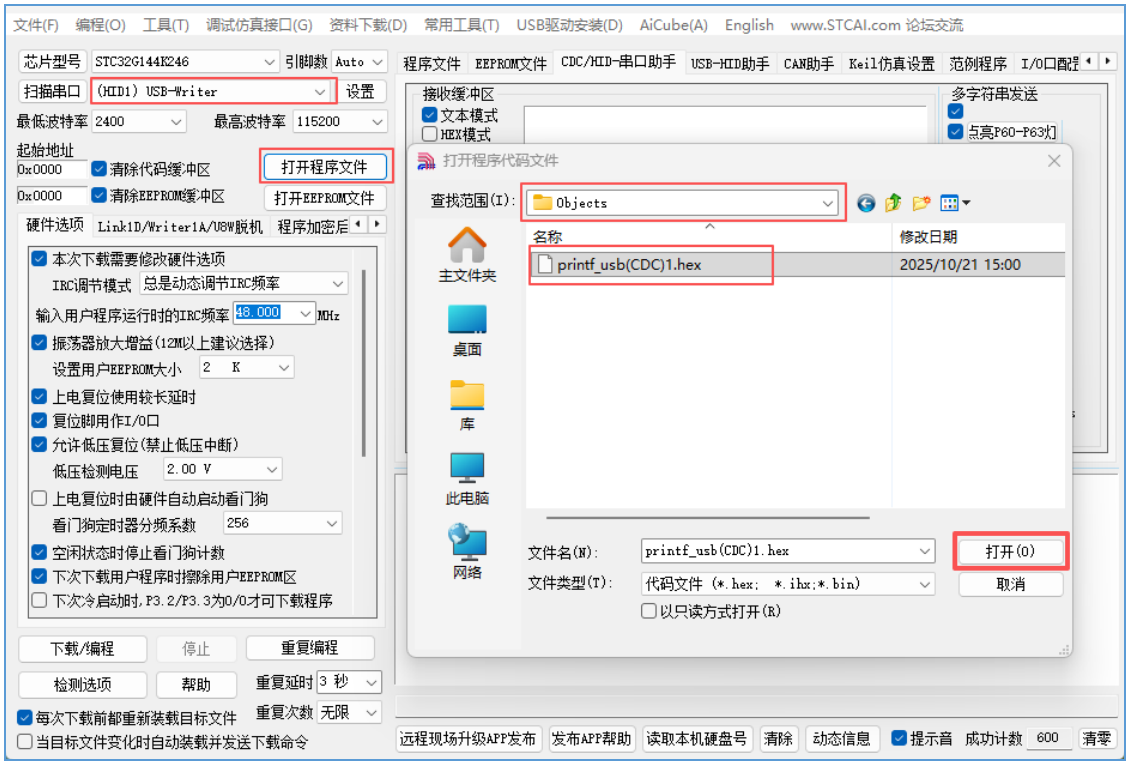
备注：如果目标 MCU 的用户程序区已有 USB 通讯程序（含 CDC、HID 等），则通过复位脚复位进入 USB 下载模式不被支持，只能通过停电上电冷启动下载或者从用户程序区软复位到系统区下载



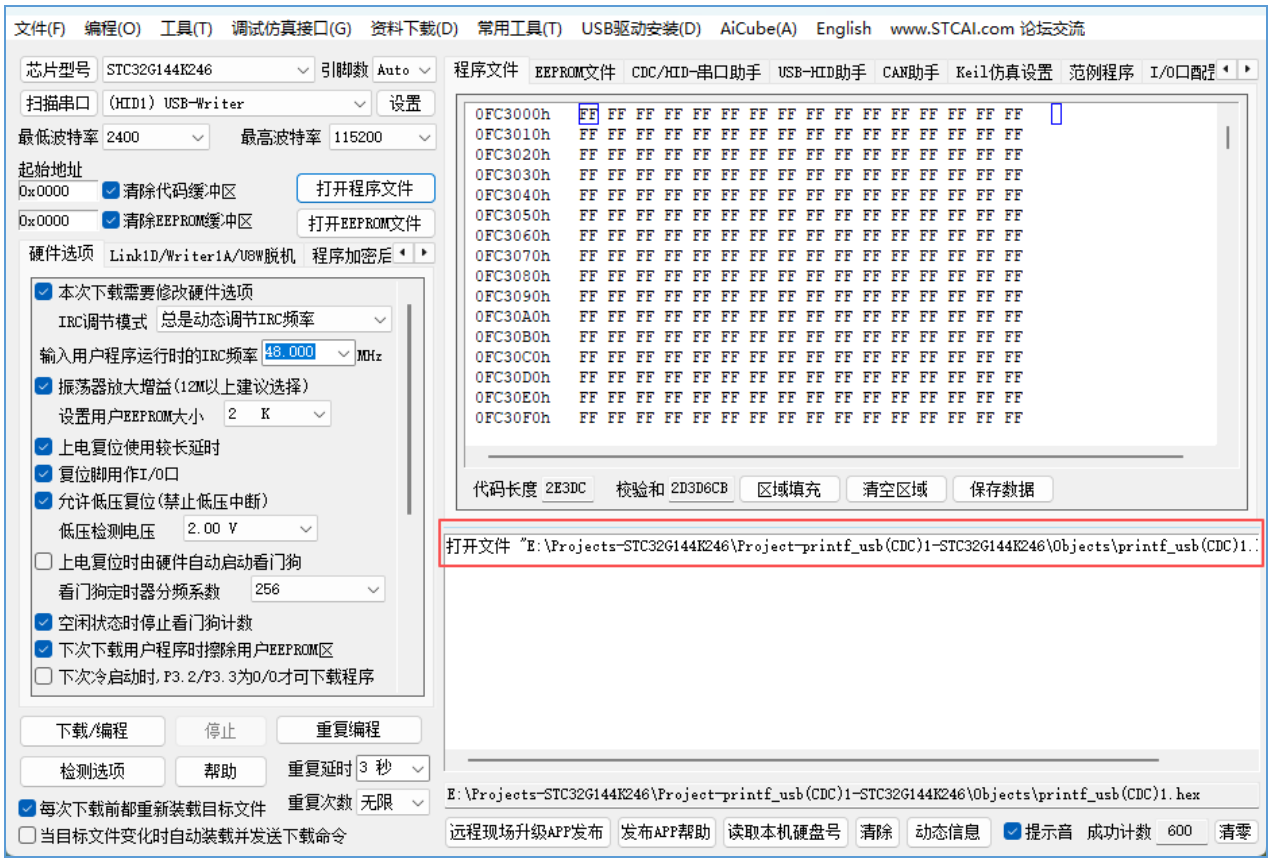
通过上述方法一或者方法二的操作步骤，我们看到 AiCube-ISP 系统软件界面的“扫描串口”的端口列表框显示“（HID1）USB-Writer”，表明 STC32G144K246 实验箱已准备好，可以下载用户程序了。

2.5.3 下载用户程序文件并观察实验现象

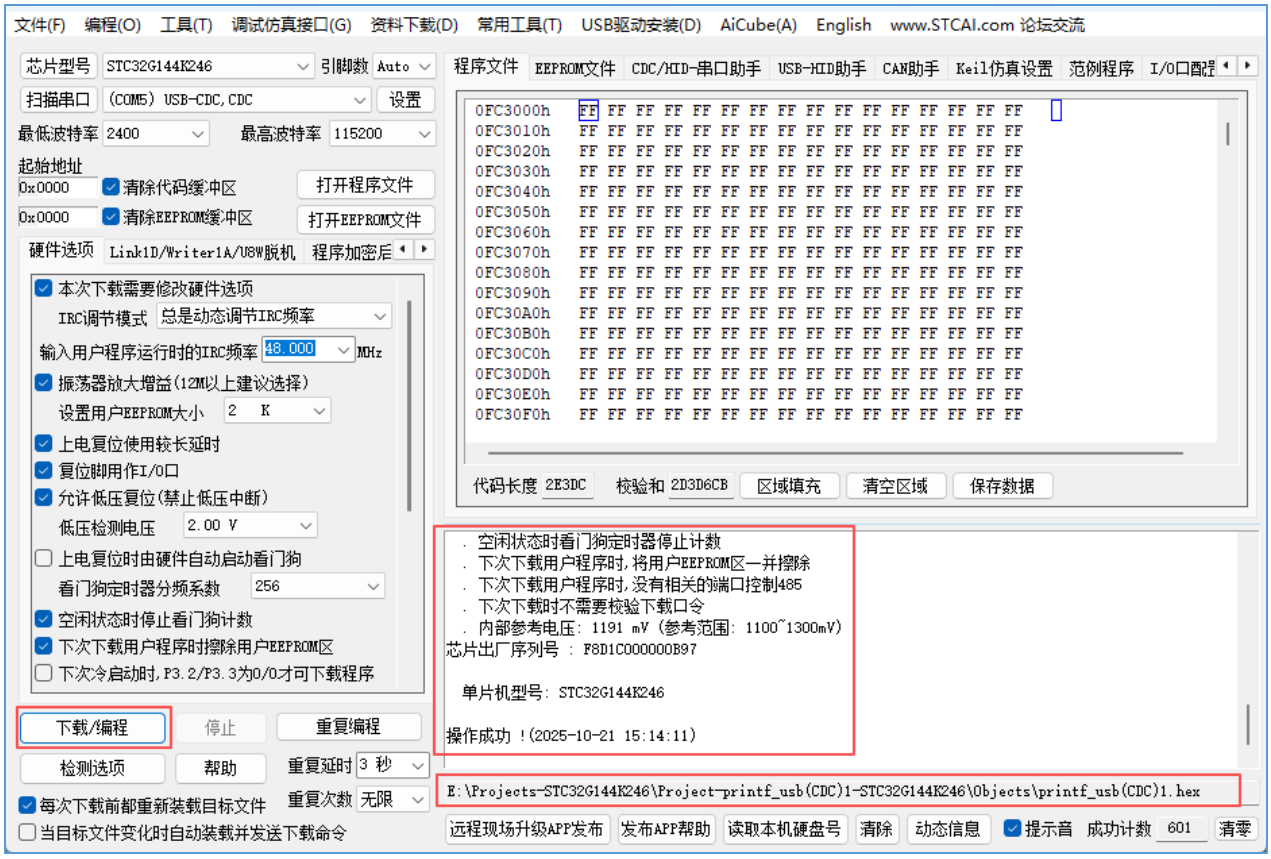
以“实验一：printf_usb("Hello World !\r\n")，第一个完整的 C 语言程序”为例，点击 AiCube-ISP 软件界面上的“打开程序文件”：print_usb(CDC)1.hex（“打开程序文件”打开的是项目程序文件编译无误后生成的 HEX 文件，一般默认存放在该项目的 Objects 目录中）：



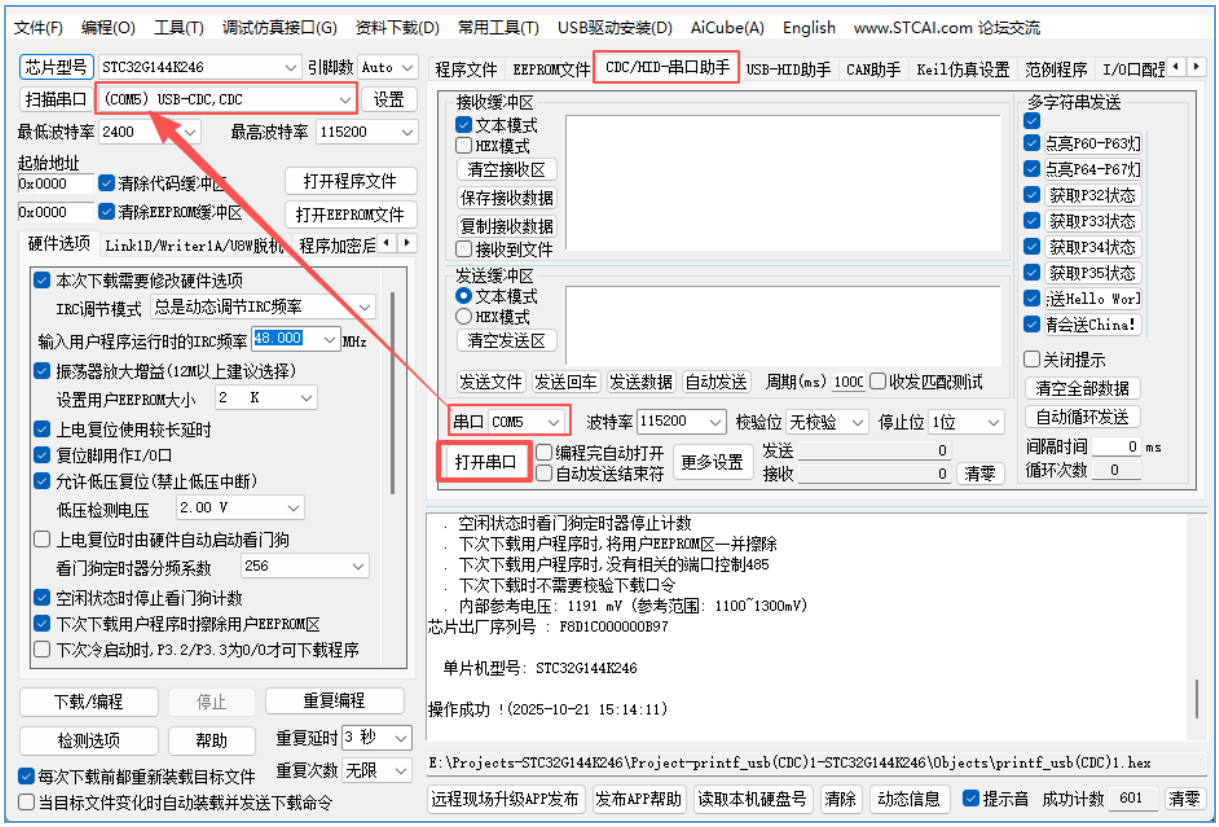
然后，AiCube-ISP 软件界面如下图所示，出现“‘打开文件***.hex’成功！”



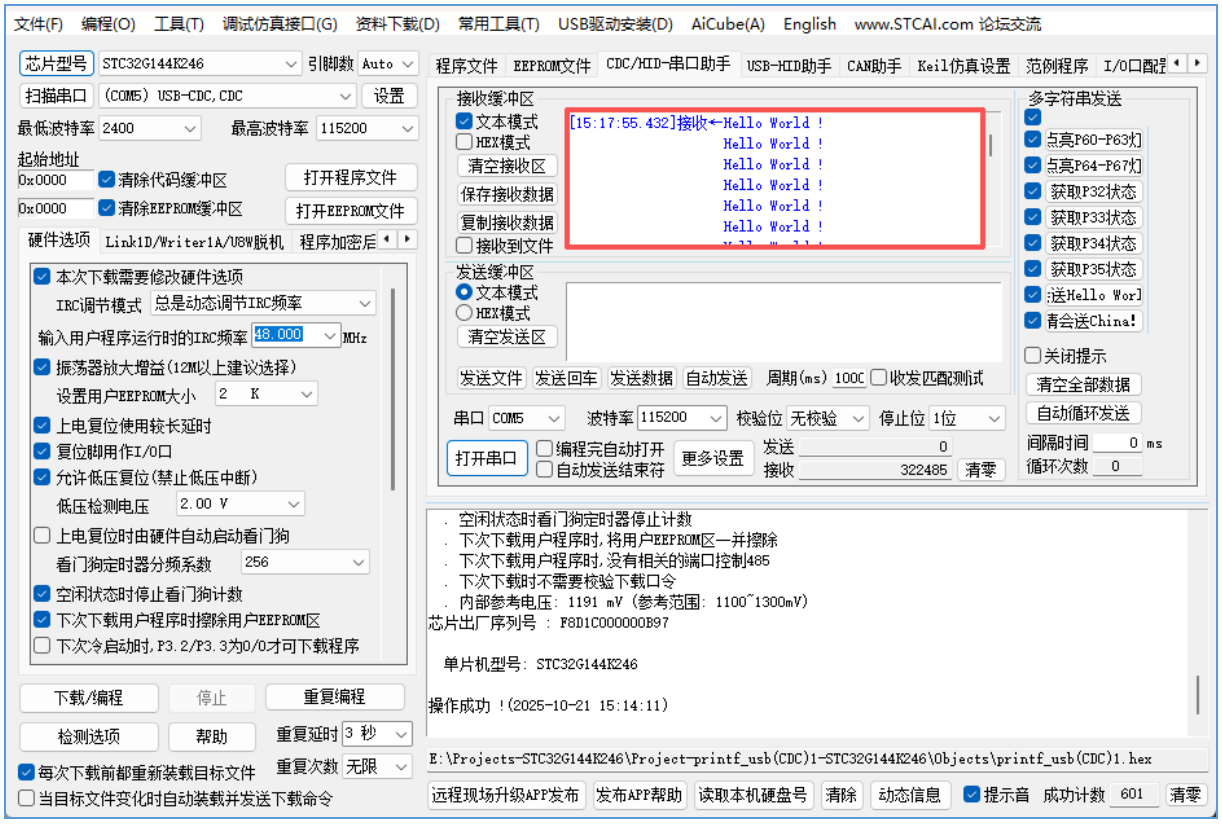
点击“下载/编程”按钮，把 print_usb(CDC)1.hex 烧录到实验箱，AiCube-ISP 软件界面出现：操作成功！恭喜你！



我们看到此时串口显示：（COM5）USB-CDC，CDC，选择 AiCube-ISP 系统软件的“CDC/HID-串口助手”选项卡，找到“打开串口”按钮，如下图：



选择好对应的串口号，点击“打开串口”按钮，CDC/HID-串口助手的“接收缓冲区”，就会不停的显示“Hello World！”，如下图：

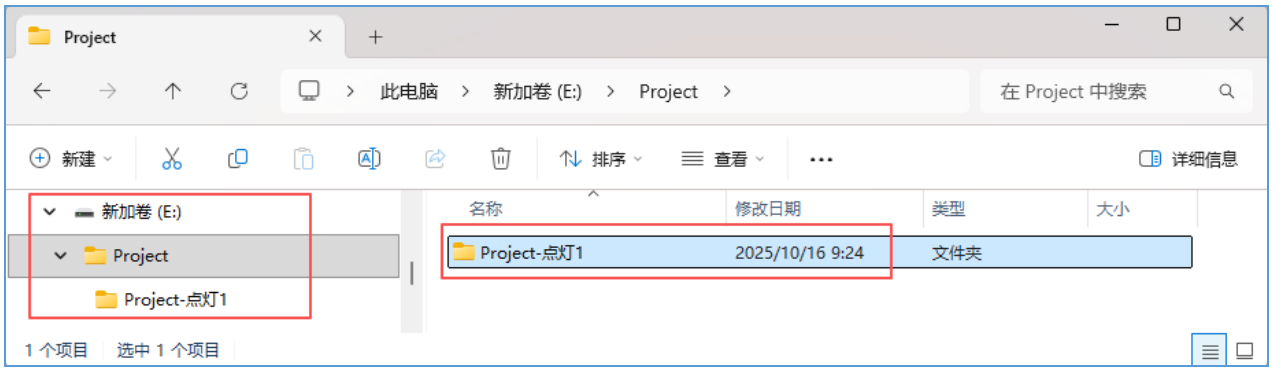


至此，printf_usb()显示“Hello World！”实验成功！

2.6 新建项目和项目设置，在 Keil 中

2.6.1 准备工作，见前面几小节

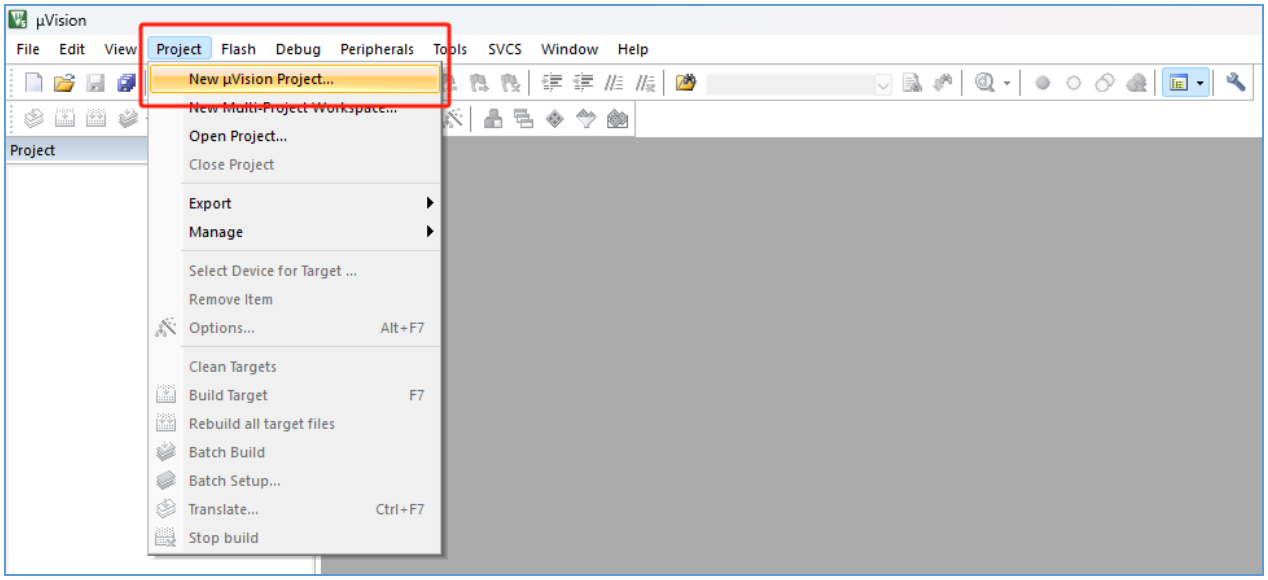
如何能建立自己的 Project，需要做的准备工作：
一、下载、安装 Keil C251/C51，见前面几小节；
二、下载、安装 AiCube-ISP，见前面几小节；
三、添加型号和头文件，见前面几小节；
四、建立一个文件夹存放你的 Project。
在 E 盘创建一个文件夹 Project，用于存放自己的项目文件



2.6.2 新建项目

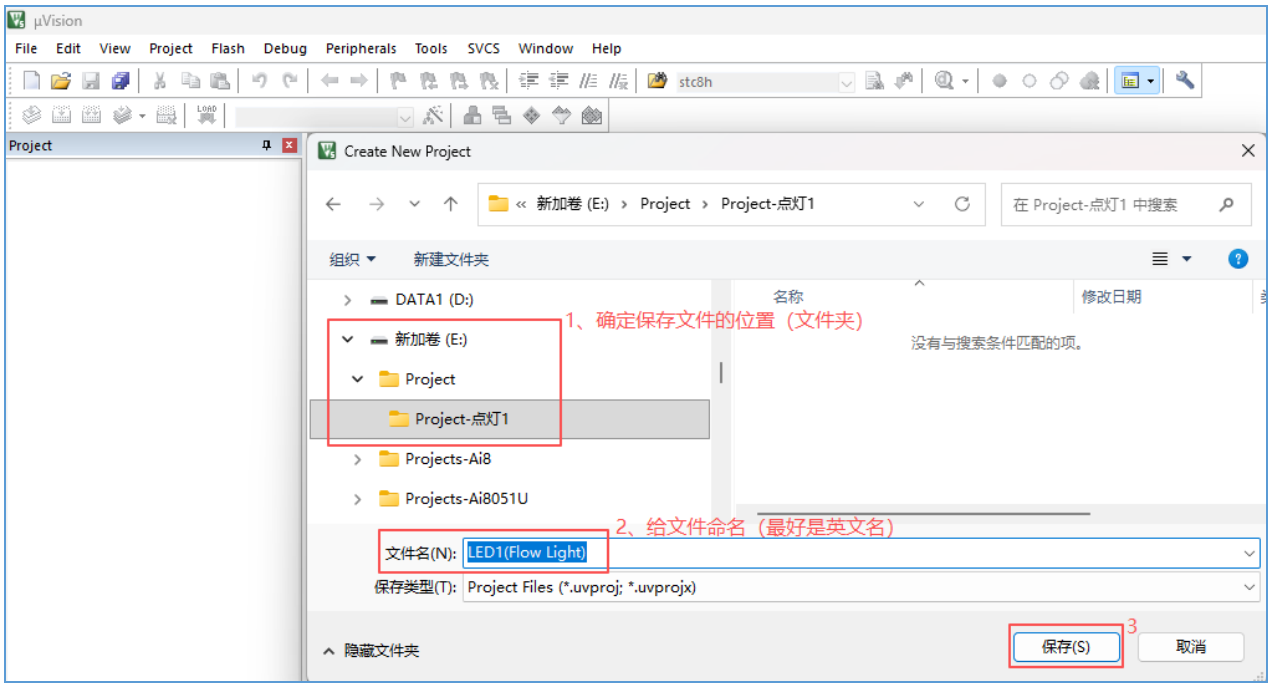
打开 Keil 软件

2.6.2.1 新建工程项目 -- 在菜单栏 Project 标签中，选择“New uVision Project”菜单项



2.6.2.2 设置项目路径和名称--将文件存入建好的文件夹中

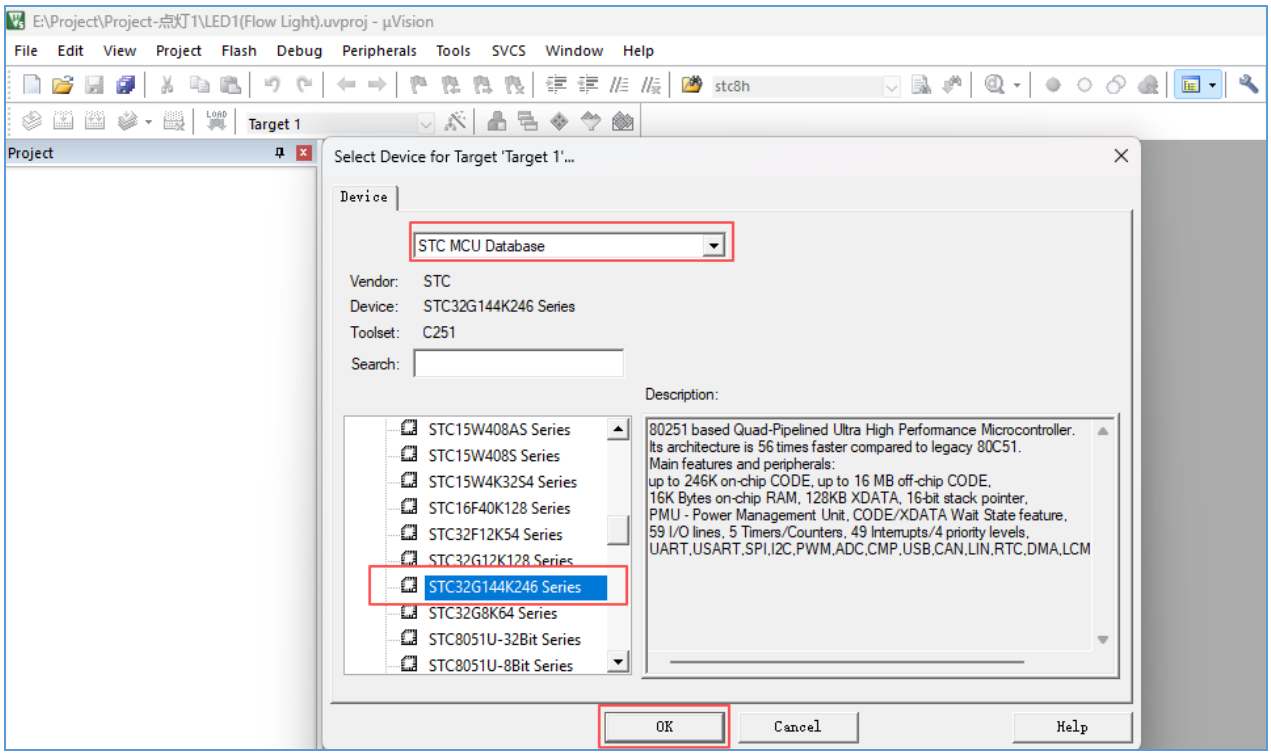
再给项目起个名称（方便查找）（尽量是英文的，如 Test…），点击“保存”。如下图：



2.6.2.3 选择单片机型号

在“Select Device for Target’Target1’”选项卡中：

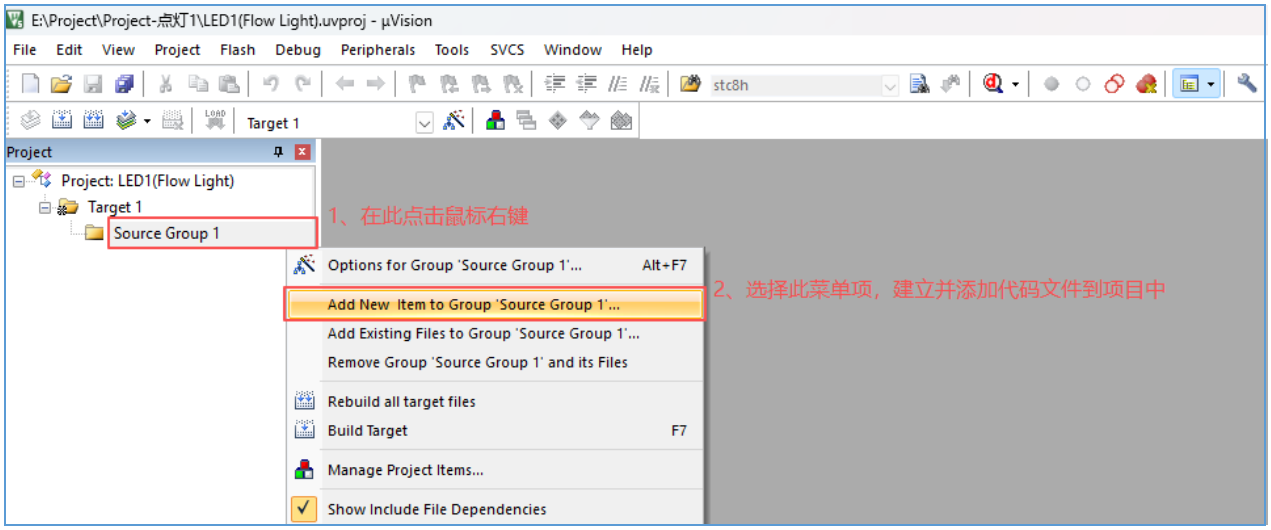
- 选择 STC MCU Database
- 然后展开 STC 选项，选择“STC32G144K246 Series”选项
- 最后点击‘OK’以完成单片机选择。



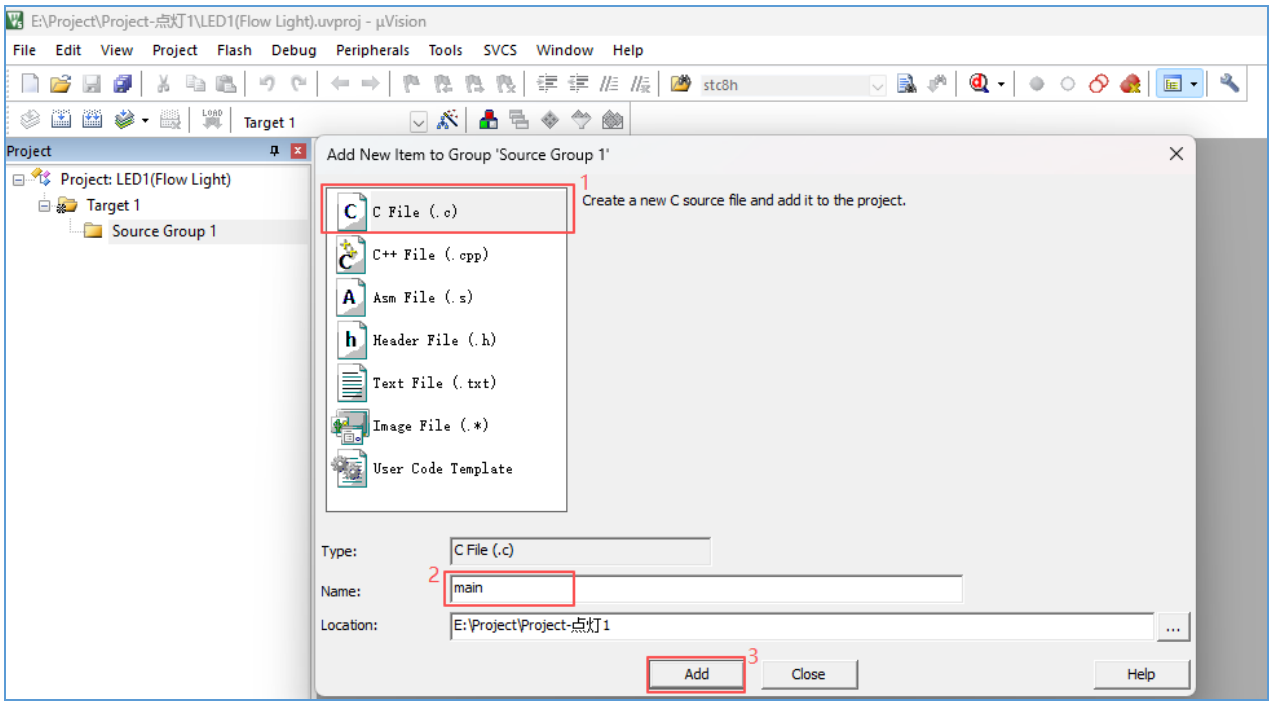
2.6.2.4 添加源代码文件到项目--给工程项目增加一个主程序文件。

选择完单片机后，工程界面还是一片灰白的。这时要给工程项目增加一个主程序文件，可以命名为 main.c 文件。

点击 Project 窗口上 Target1 的+，出现“Source Group1”，右键点击“Source Group1”，出现下图。



再选择快捷菜单中的“Add New Item to Group ‘Source Group 1’”菜单项，出现如下界面提示：

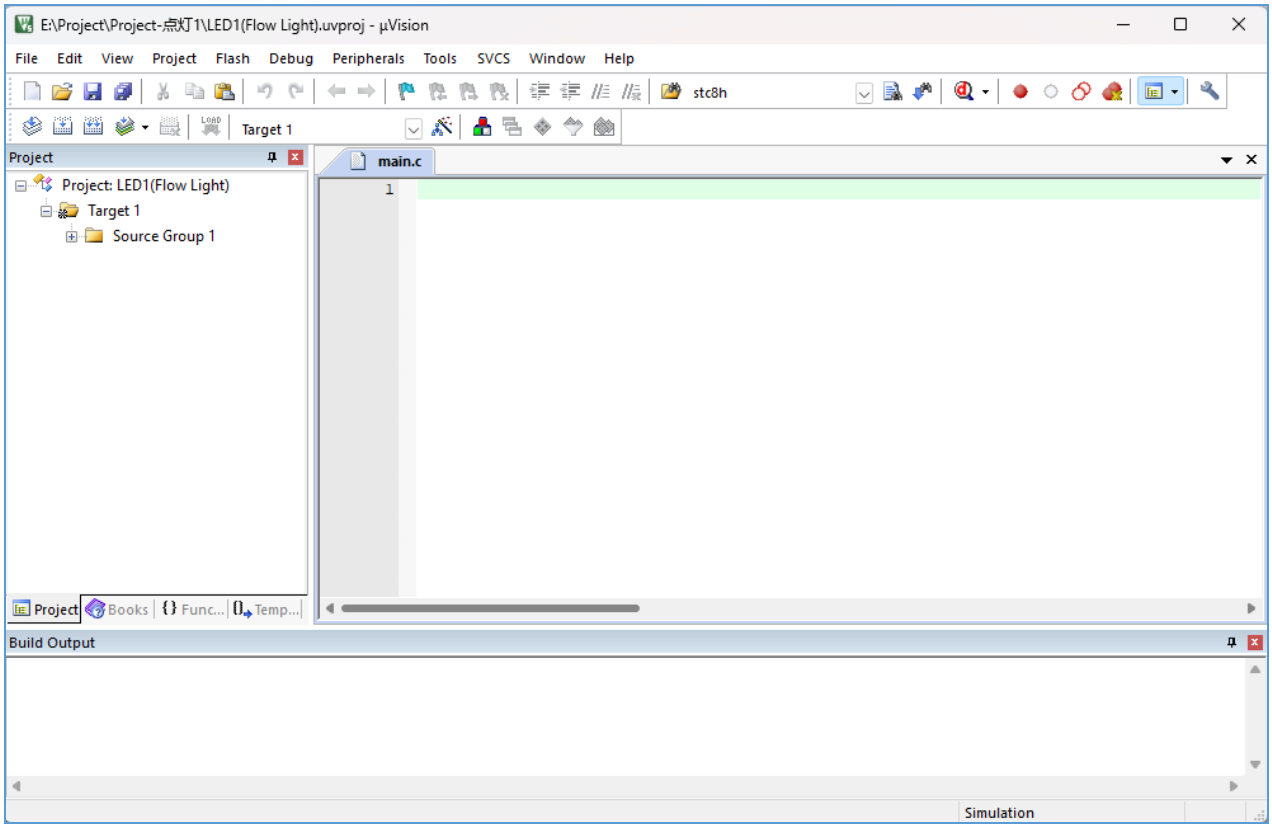


在上图中选择文件类型是：*.c

在 Name 输入框中输入文件名是：main 或其他任意文件名，最好是英文名

在 Location 输入框中选择当前项目的目录

然后点击 Add 按钮，将该新文件加入到当前项目。如下图：



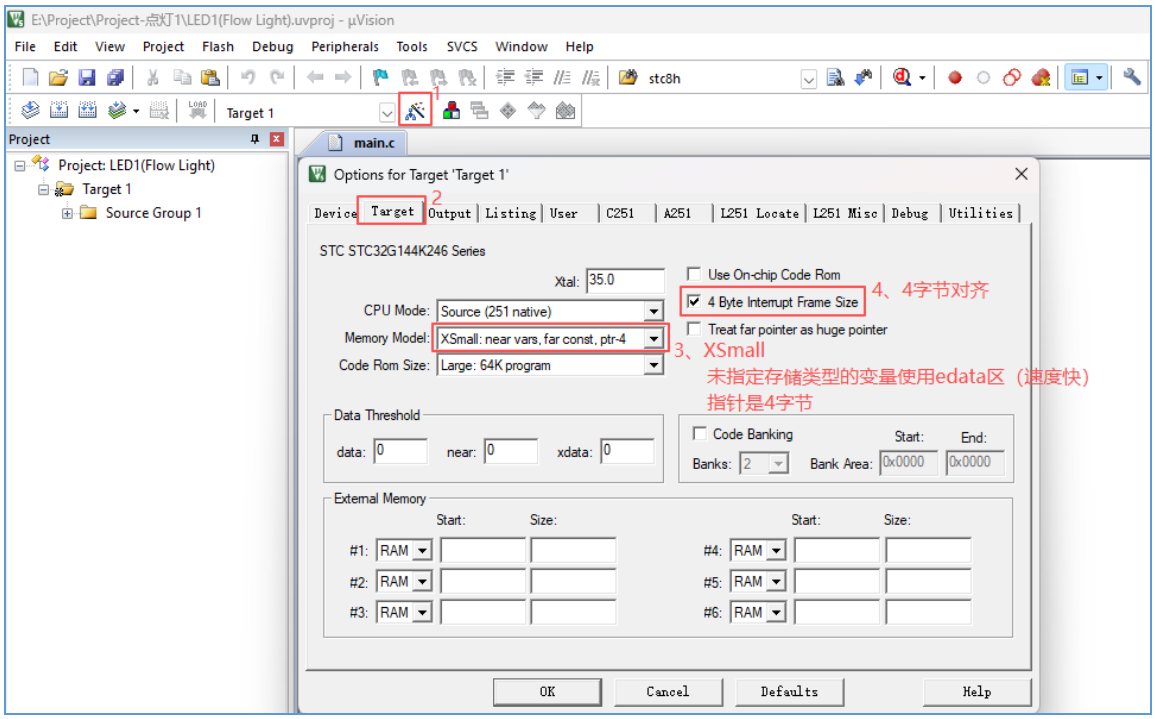
至此，生成了一个空的 main.c 主程序文件并且已经加入到当前项目。可以在该文件中编写代码了。详见后续的实验例程。

当然，在编写代码前，最好先进行 Project 的各项基础设置。

2.6.3 项目的各种基础选项设置

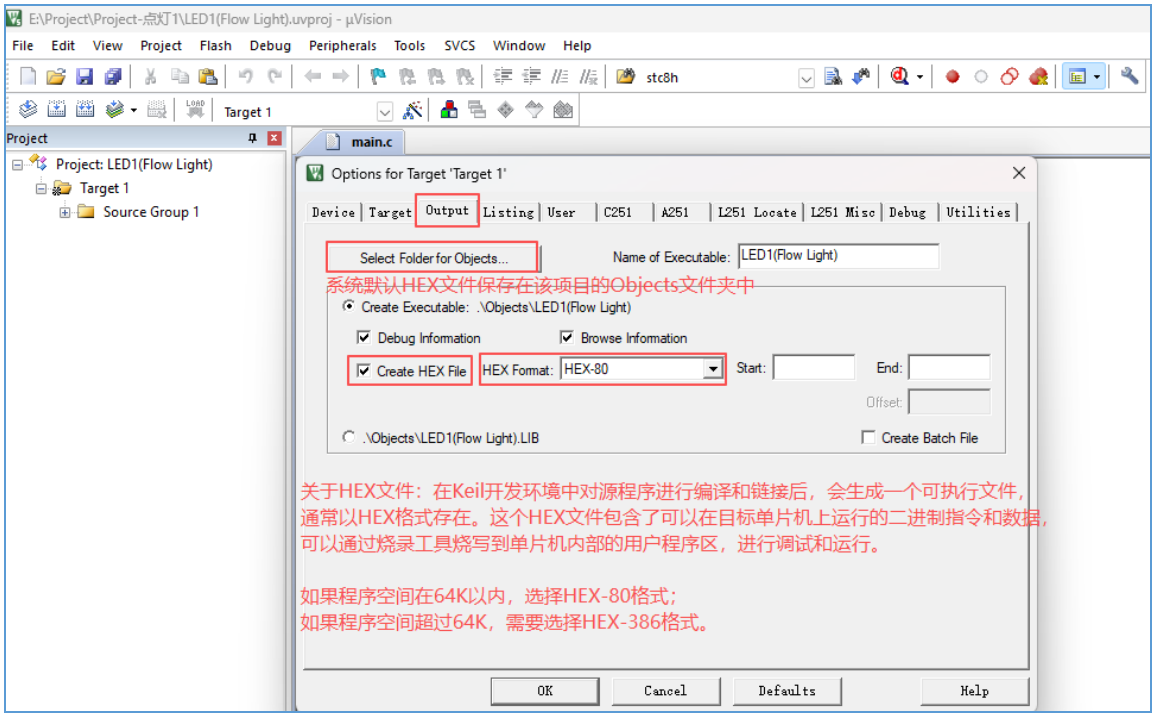
在 Build 工具栏找到一个类似魔术棒的工具，就是任务选项的设置。

2.6.3.1 Target 选项卡--Memory Model 选择 XSmall 模式、勾选 4 字节对齐



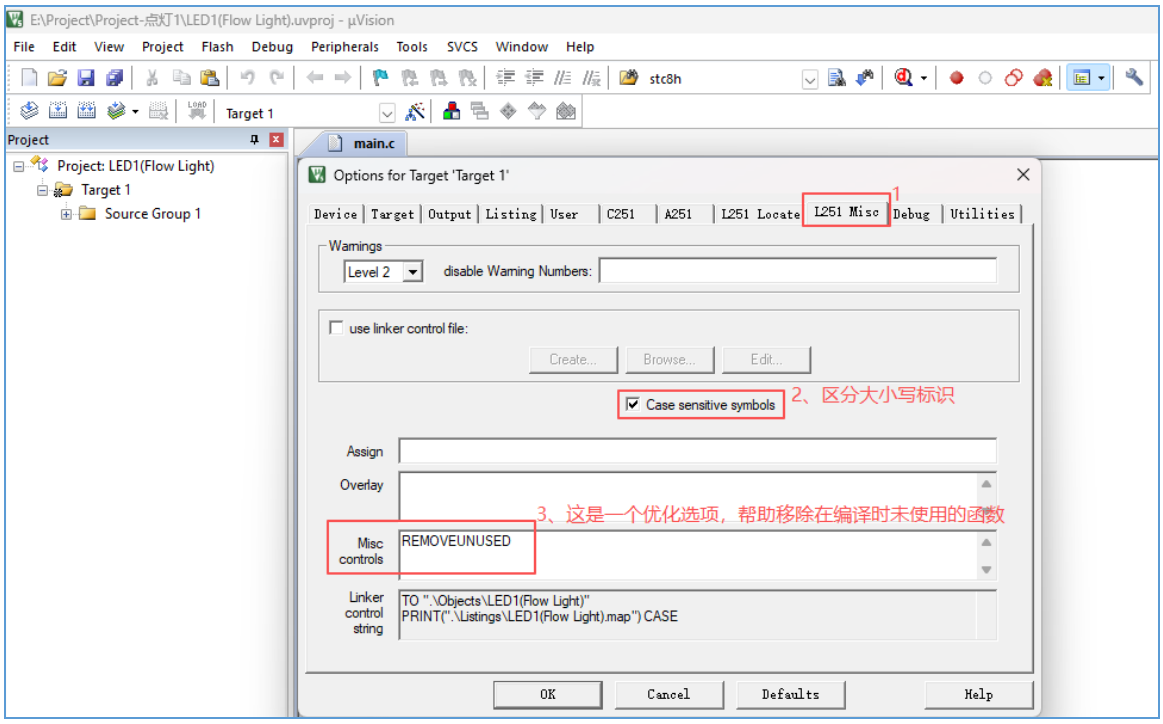
2.6.3.2 Output 选项卡--勾选 Create HEX File，选择 HEX-80

切换到 Output 选项卡，勾选‘Creat HEX File’选项，否则编译后是不会创建 HEX 文件的。HEX 文件是可以通过烧录工具烧录到单片机用户程序区，系统默认 HEX 文件保存在该项目的 Objects 文件夹中。



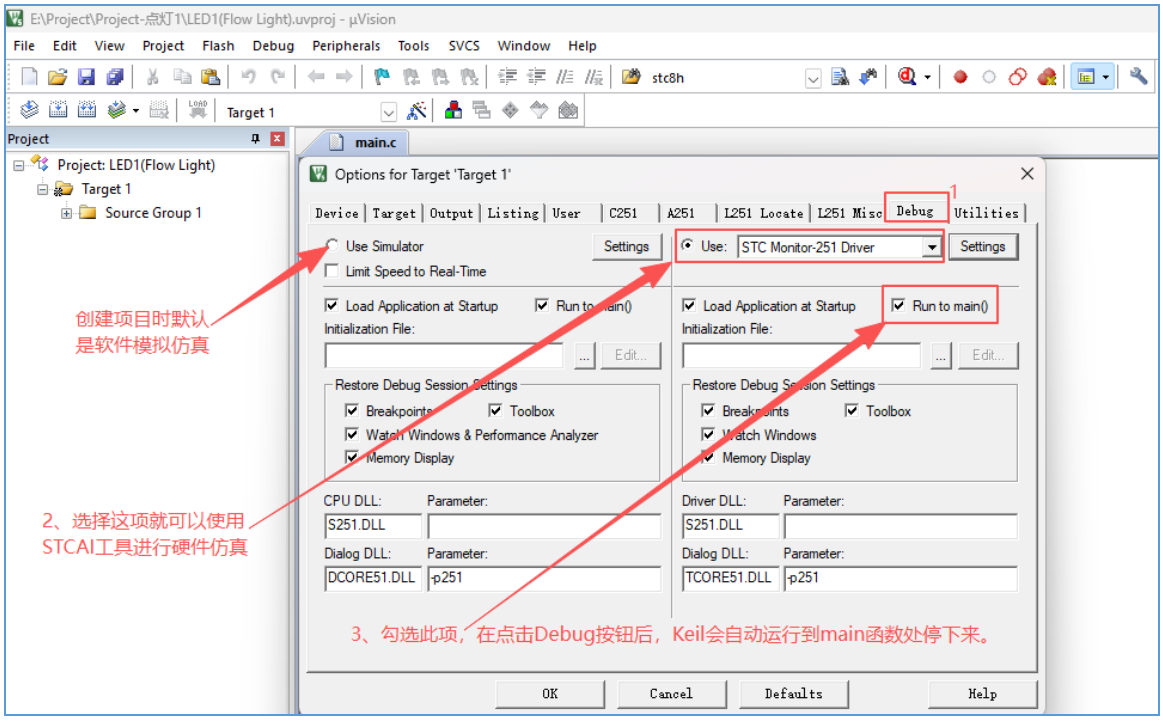
2.6.3.3 L251 Misc 选项卡--在 Misc controls 输入框输入‘REMOVEUNUSED’

切换到 L251 Misc 选项卡，在 Misc controls 输入框中，输入大写的‘REMOVEUNUSED’，移除未使用的函数（如：库函数），以实现不使用的函数不进行链接。

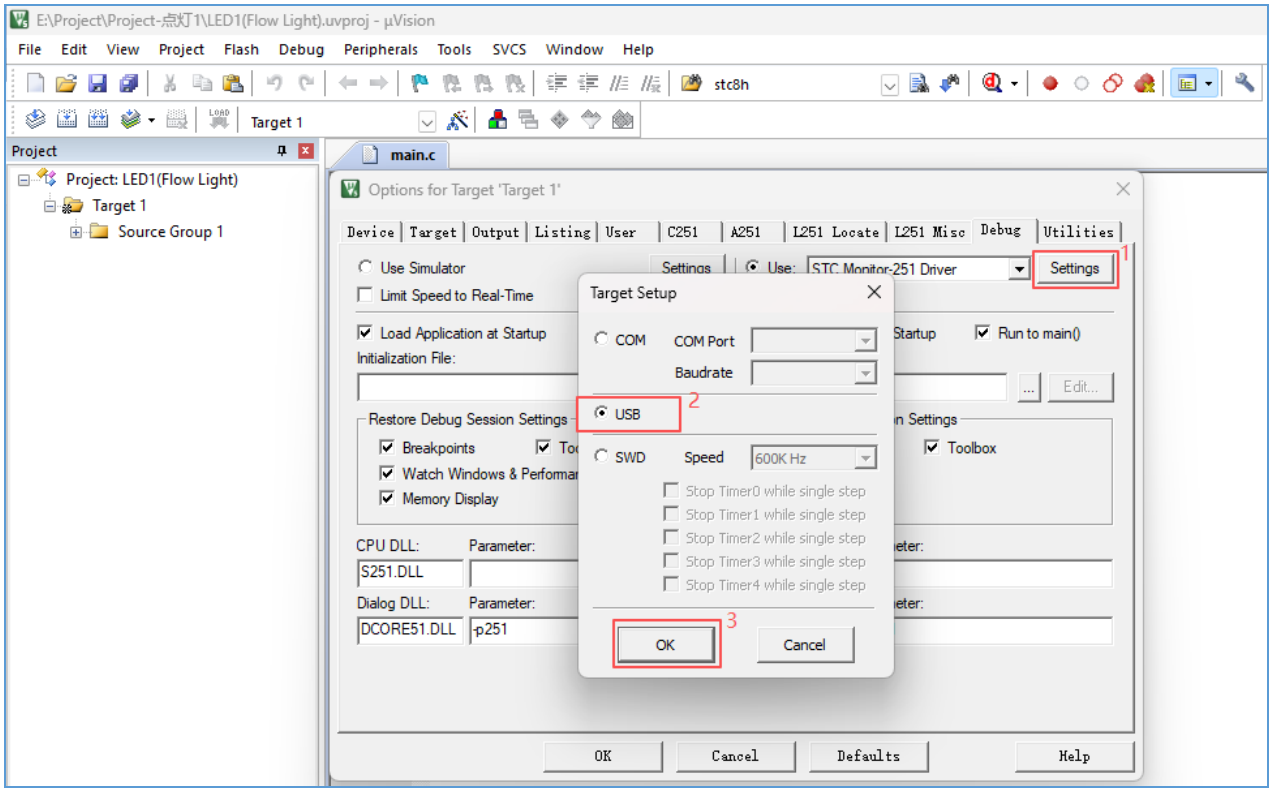


2.6.3.4 Debug 选项卡--仿真设置，使用 STCAI 工具进行硬件仿真设置

切换到 Debug 选项卡，创建项目时，Keil 默认是软件模拟仿真。我们可以借助 STCAI 的工具进行硬件仿真，如下图：

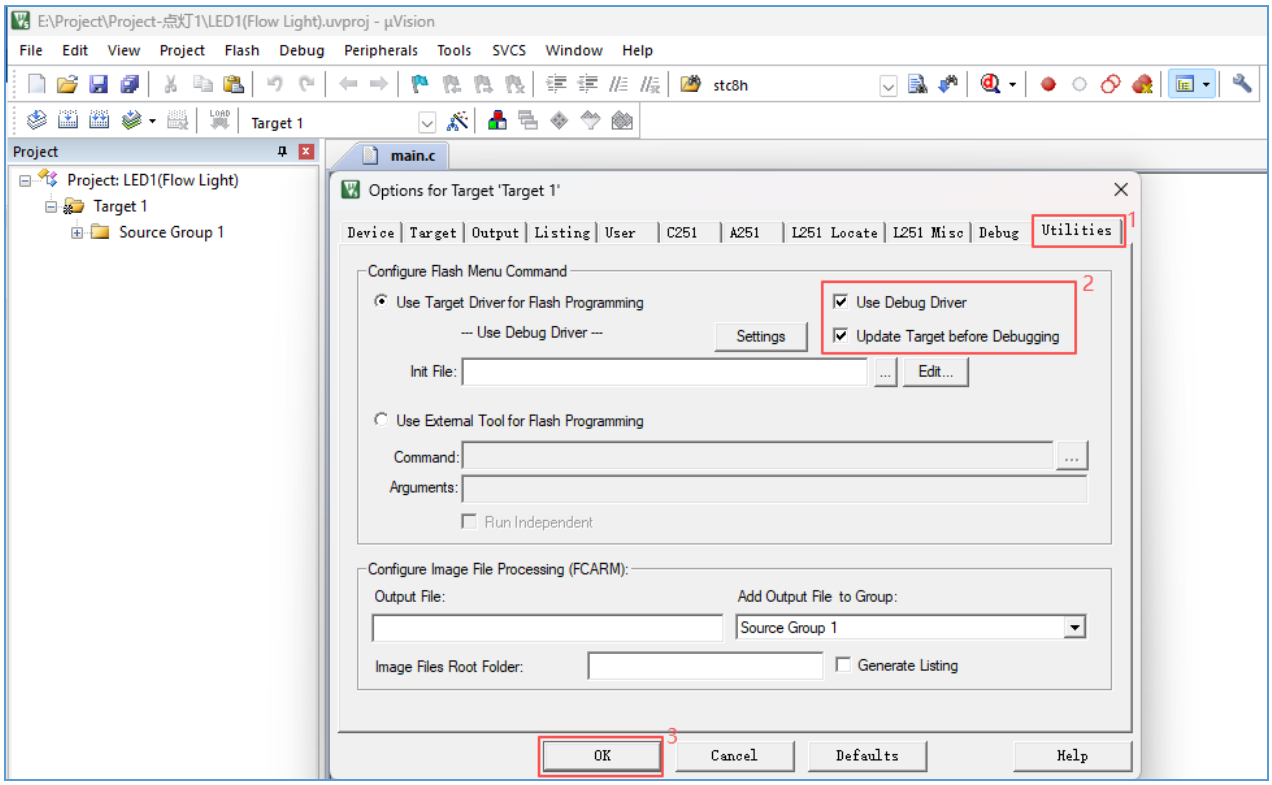


STC32G144K246 是 32 位 MCU，可以 USB 直接通信仿真、COM 串口通信仿真、SWD 仿真。



2.6.3.5 Utilities 选项卡--仿真设置，勾选“Use Debug Driver”

切换到“Utilities”选项卡，勾选“Use Debug Driver”和“Update Target before Debugging”，点击“OK”，完成设置。

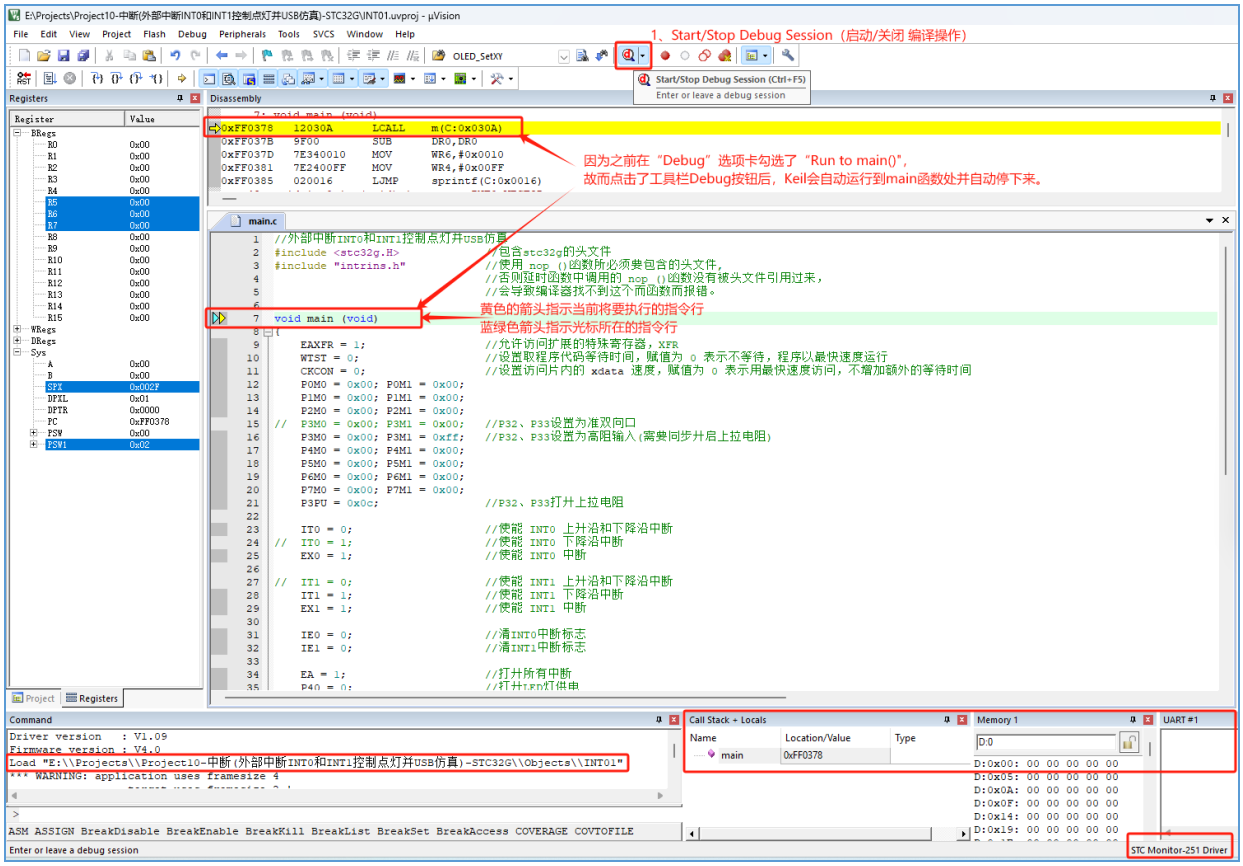


至此，工程项目的各种基础选项设置结束。可以进入下一节，在程序文件中添加实际的代码。

如果不使用仿真，仿真设置可以跳过。

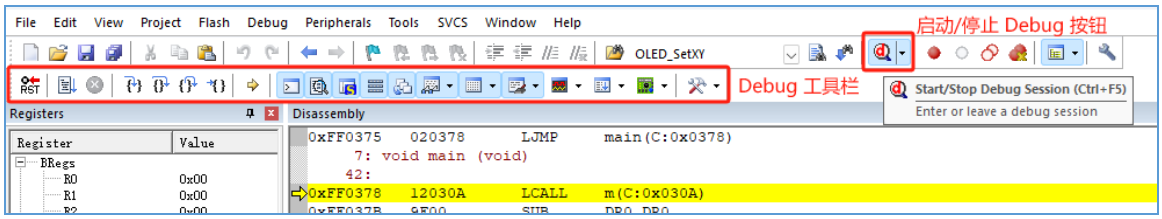
2.7 了解 File、Build、Debug 工具栏与仿真相关的常用操作，在 Keil 中

创建项目并对项目进行前面步骤的设置后，返回 Keil 界面，代码程序编译无误，点击工具栏的“Debug”（启动/关闭编译调试操作）按钮后，Keil 软件进入仿真界面，如下图：

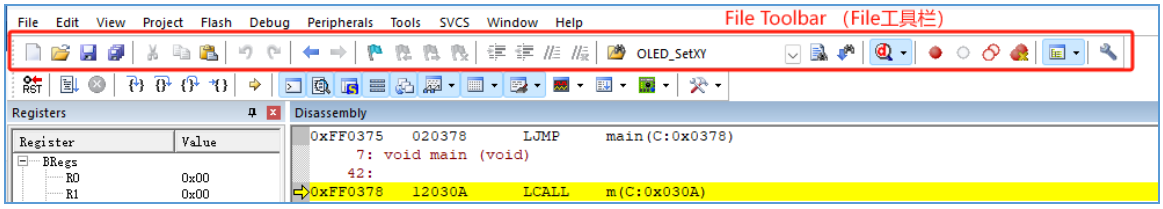


提示：退出 Debug 调试界面，同样点击工具栏的“Debug”按钮。

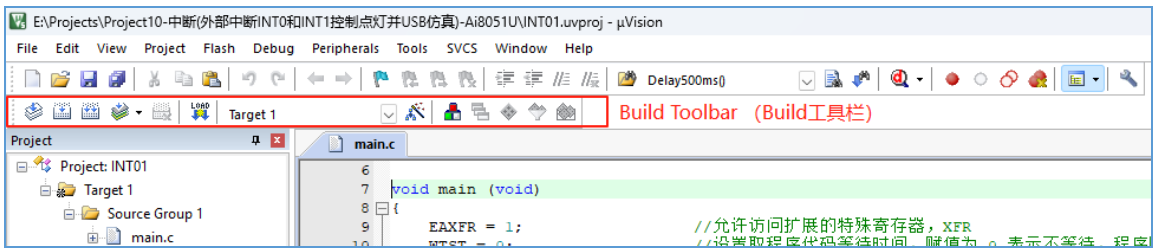
在 Debug 界面，我们要认识一下 **Debug Toolbar（Debug 工具栏）**的常用调试/仿真按钮：



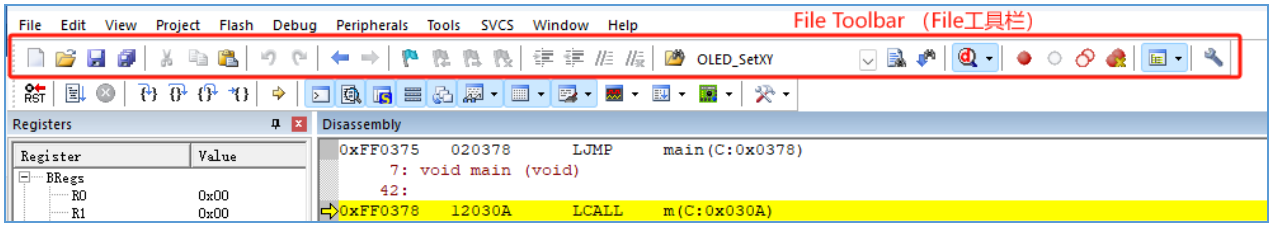
和 **File Toolbar（File 工具栏）**中在调试/仿真时的几个常用按钮：



和 **Build Toolbar（Build 工具栏）**中在调试/仿真时的几个常用按钮：

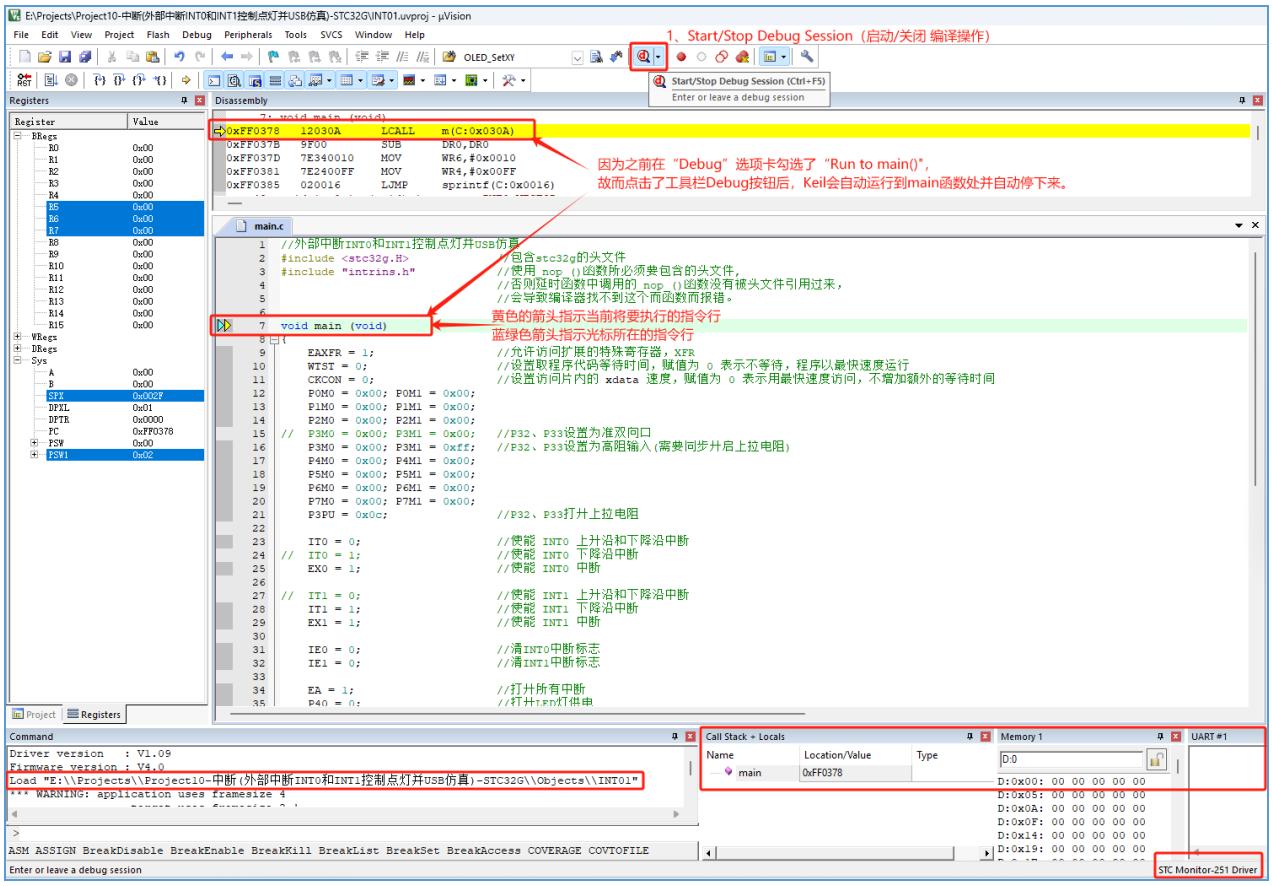


2.7.1 File Toolbar（File 工具栏）



2.7.1.1 Start/Stop Debug Session，File Toolbar

Start/Stop Debug Session（Enter or leave a debug session）按钮：启动/停止 Debug 按钮。在软件开发过程中，调试是不可或缺的一环。Keil 提供了丰富的 Debug 调试工具，帮助我们快速定位和解决程序中的问题。点击此按钮，即刻进入 Debug 界面，如下图。再次点击此按钮，就会退出 Debug 界面。

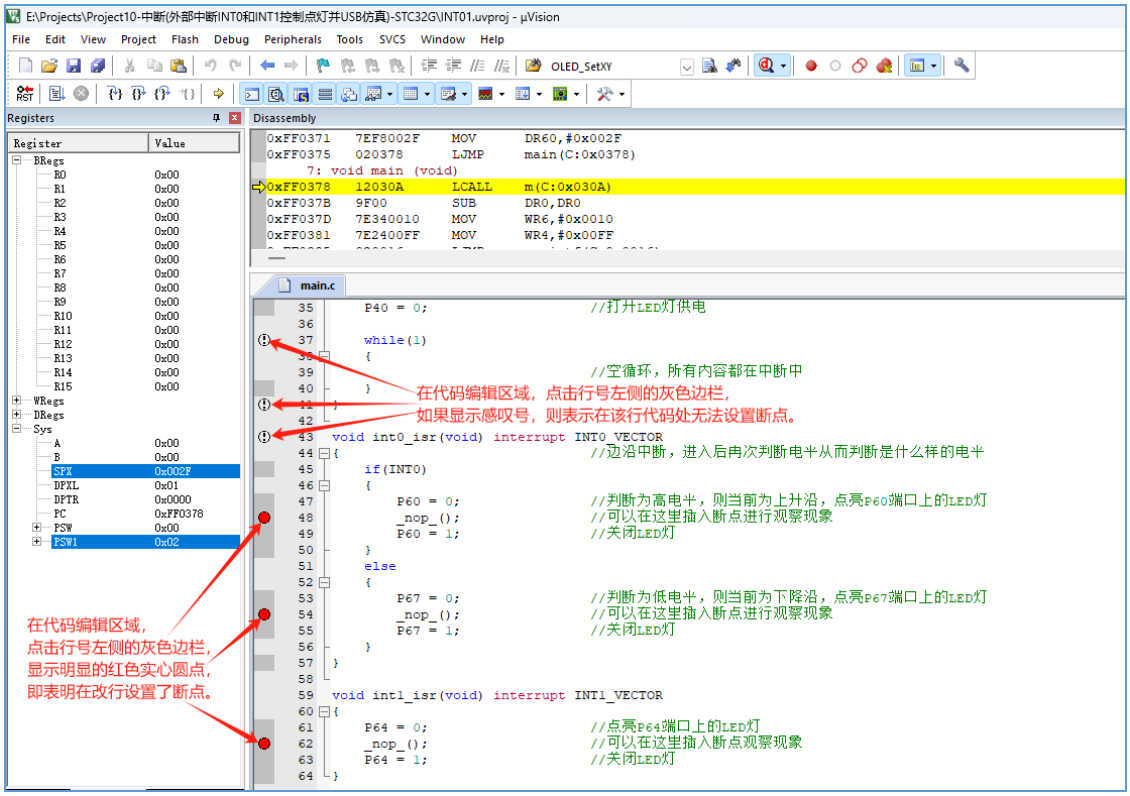


2.7.1.2 Breakpoint，File Toolbar

在调试/仿真程序时，会涉及一个很重要的工具：Breakpoint（断点）。

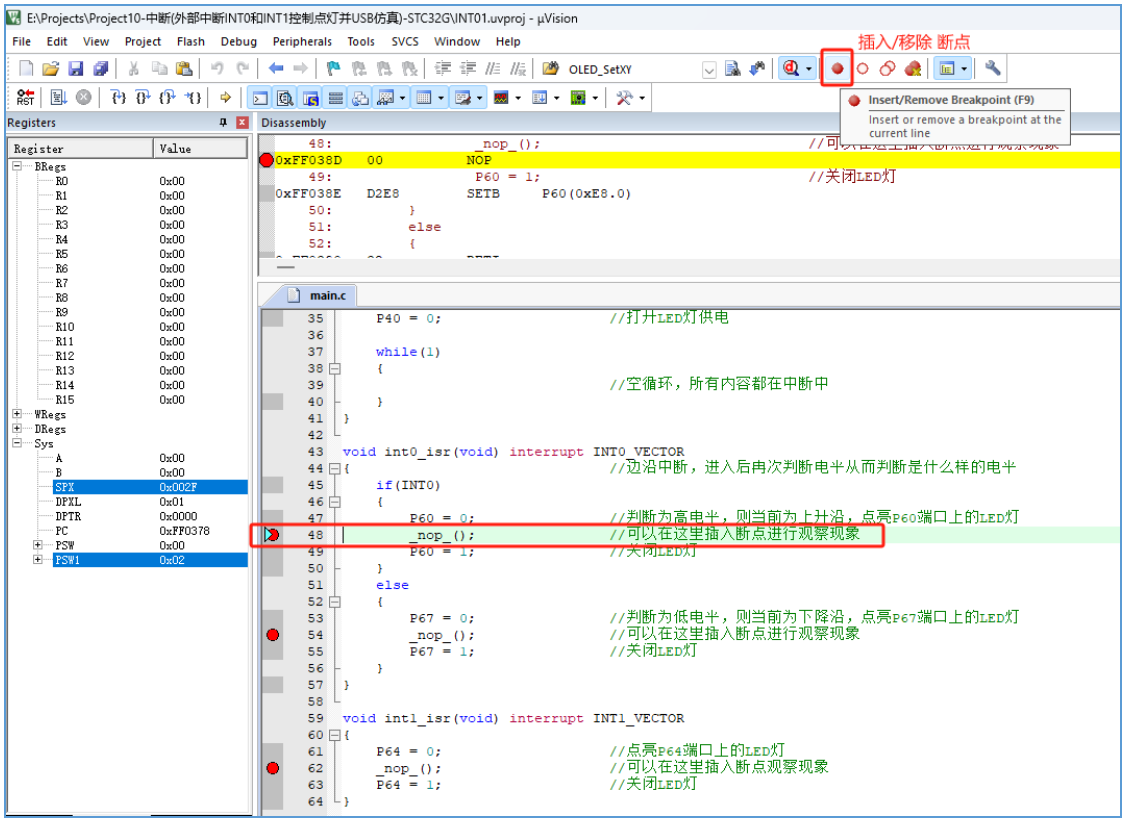
Breakpoint：断点。在 Keil 中，断点是一种强大的调试工具。断点本身是程序中的一个位置标记，当程序执行到断点位置时，程序会暂停执行，从而方便我们观察实验现象，定位、分析和排查程序中的问题等。在 Keil 环境中，设置断点有多种方式，下面我们介绍最简单的 2 种断点设置方法。

1、在代码编辑区域，点击行号左侧的灰色边栏，显示明显的红色实心圆点，即表明在该行代码处插入了断点。如果显示感叹号，则表示在该行代码处无法设置断点。在断点处点击断点（红色实心圆点）即为取消断点。



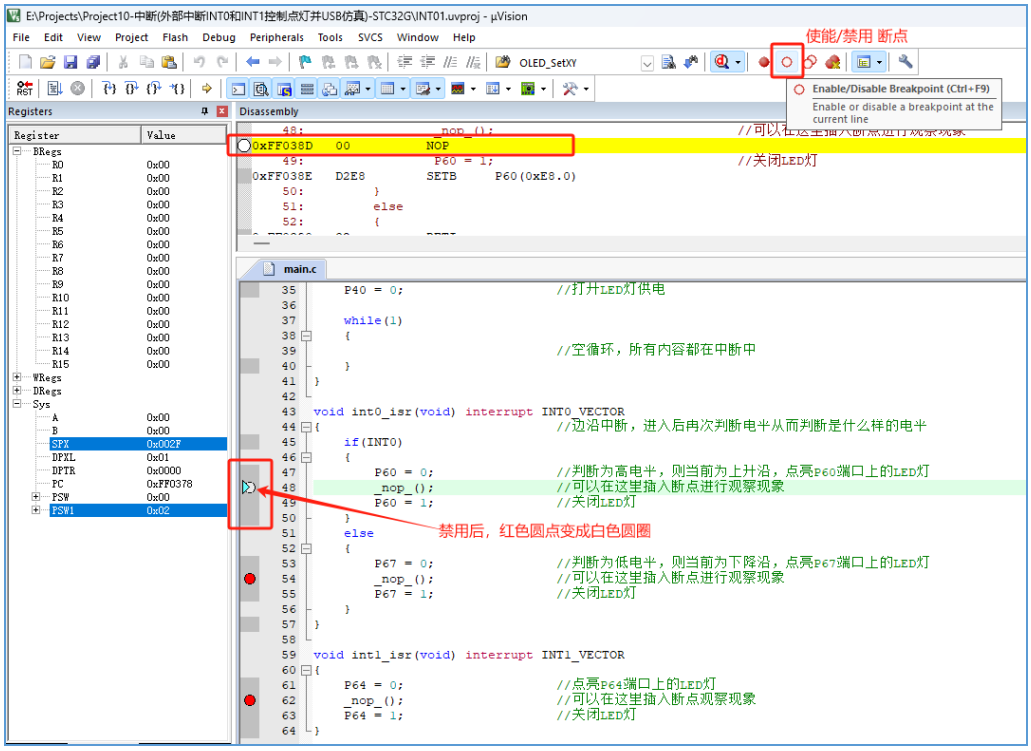
2.7.1.3 Insert/Remove Breakpoint，File Toolbar

2、Insert/Remove Breakpoint（Insert or remove a breakpoint at the current line）按钮：在当前光标所在行插入/移除断点。在当前光标所在行，该按钮如果显示红色，则表示可在该处插入断点。点击该按钮，即在当前行插入断点，当前行左侧就会显示一个红色圆点表示断点位置。该按钮如果显示灰色，则无法在该处插入断点。定位光标在断点所在代码行，点击该按钮，即移除该处断点，红色圆点消失。



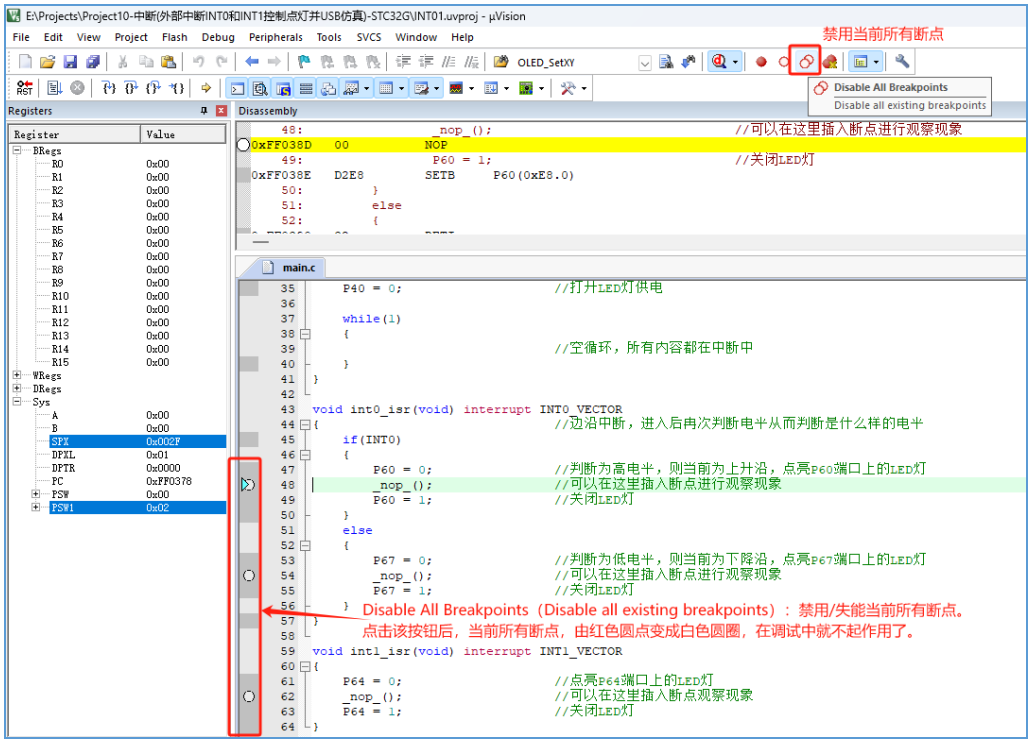
2.7.1.4 Enable/Disable Breakpoint，File Toolbar

Enable/Disable Breakpoint（Enable or disable a breakpoint at the current line）： 使能/禁止断点。启用/禁用当前光标所在行的断点。断点在启用时，红色圆点显示。禁用后，红色圆点变成白色圆圈，在调试中就不起作用了。断点使能/禁止按钮，在设置断点后，你可以使用此功能按钮来启用或禁用断点，从而控制代码的暂停与继续。



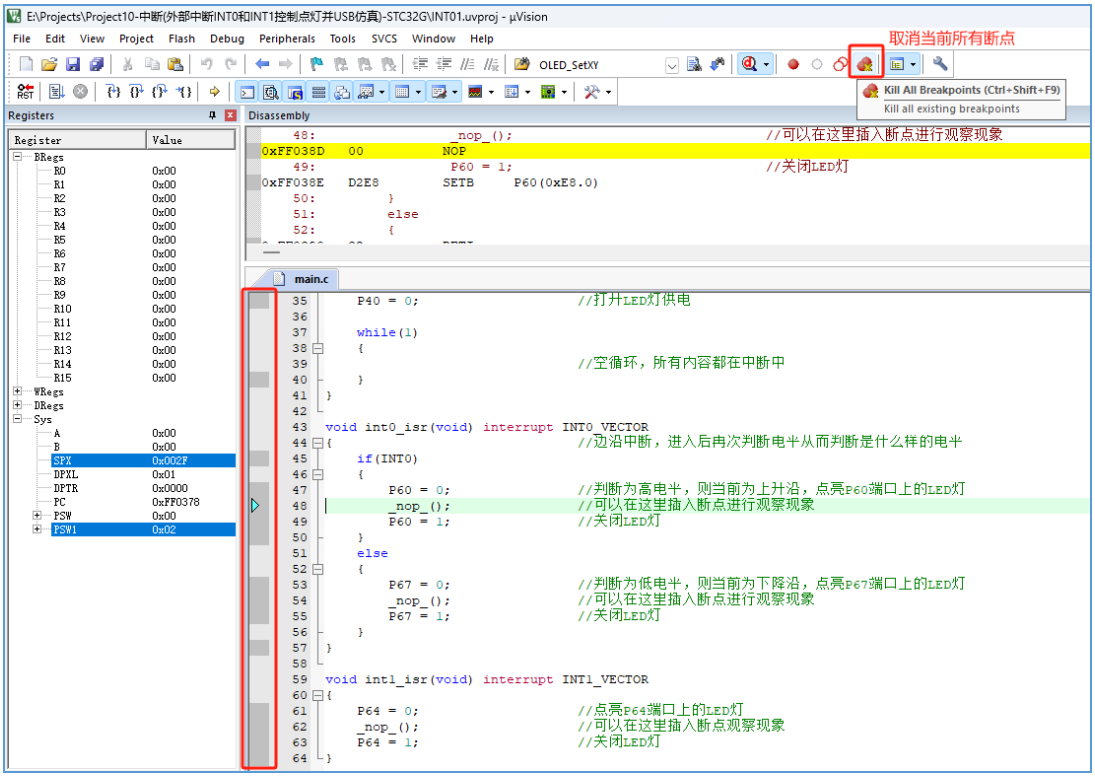
2.7.1.5 Disable All Breakpoints，File Toolbar

Disable All Breakpoints（Disable all existing breakpoints）： 禁用/失能当前所有断点。点击该按钮后，当前所有断点，由红色圆点变成白色圆圈，在调试中就不起作用了。但此按钮不能再次使能所有断点。



2.7.1.6 Kill All Breakpoints，File Toolbar

Kill All Breakpoints（Kill all existing breakpoints）：取消当前所有断点。和 **Disable All Breakpoints** 不同，点击该按钮后，所有断点标记消失。

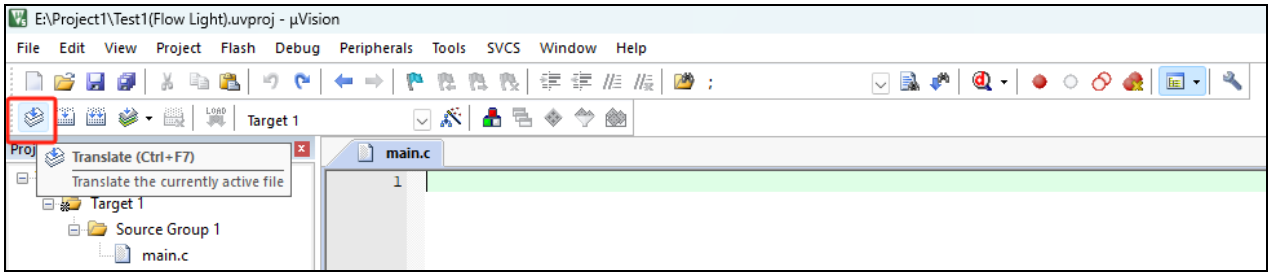


2.7.2 Build Toolbar（Build 工具栏）

Keil 是一款软件开发系统，输入工程项目的程序源代码后，要进行编译（Build）。编译过程是将 C 语言源代码转换为机器可以执行的机器代码（通常是 HEX 文件），这样程序才能在目标硬件上运行。编译过程中，编译器对源代码进行预处理，处理宏定义、条件编译指令等。然后将预处理后的源代码转换成汇编语言。这一步会检查语法错误，并将高级语言转换为低级语言。下面我们认识一下 Build 工具栏的几个重要的功能按钮。

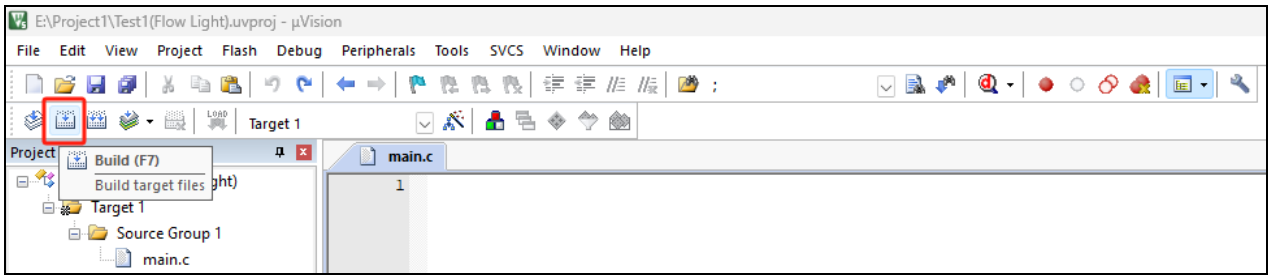
2.7.2.1 Translate（Translate the currently active file），Build Toolbar

Translate（Translate the currently active file）按钮：编译当前正在编辑的源文件(main.c 仅生成 main.o)，在 Translate 过程中会检查源文件中的语法错误，但并不生成最终可执行文件。



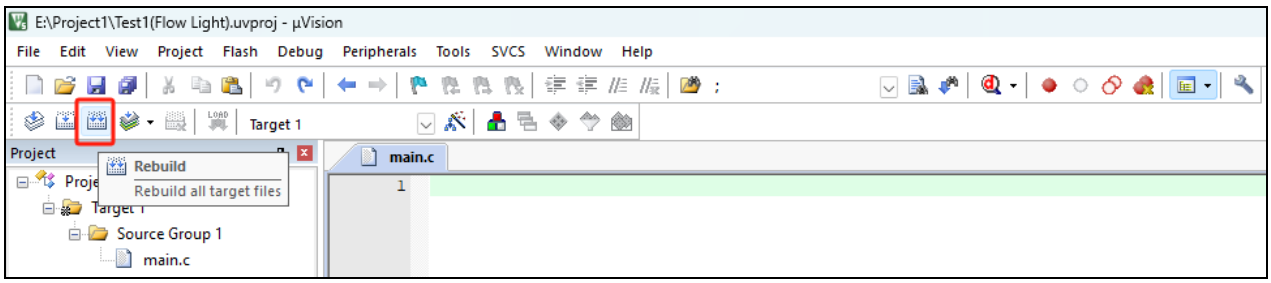
2.7.2.2 Build（Build target files），Build Toolbar

Build（Build target files）按钮：Build 在技术文档中是 increase build，所以理解 Build 按钮是编译工程中当前修改的文件及与此文件相关联的所有文件（模块），检查语法错误，并重新链接生成可执行文件。这样特别在调试大型项目时，因为不需要每次都从头开始编译整个项目，可以节省时间。平时项目开发调试验证过程中，也更多选择用 Build 编译生成可执行文件验证。如果工程是首次编译，会直接调用 Rebuild All。



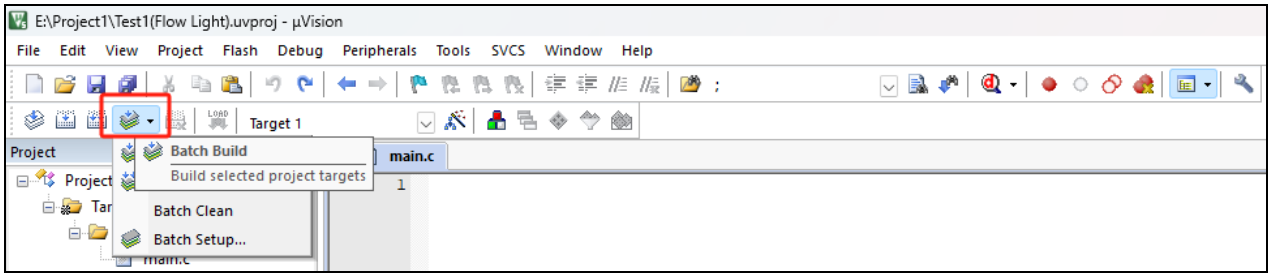
2.7.2.3 Rebuild（Rebuild all target files），Build Toolbar

Rebuild（Rebuild all target files）按钮：编译全部文件，检查语法错误，并重新链接生成可执行文件。



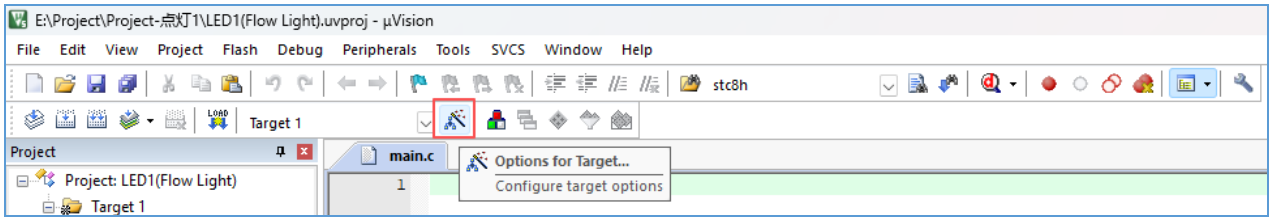
2.7.2.4 Batch Build（Build selected project targets），Build Toolbar

Batch Build（Build selected project targets）按钮：一次编译多个项目。在 Keil 中，可创建多个 projects，并通过 Batch Build 同时对这些 projects 进行编译。

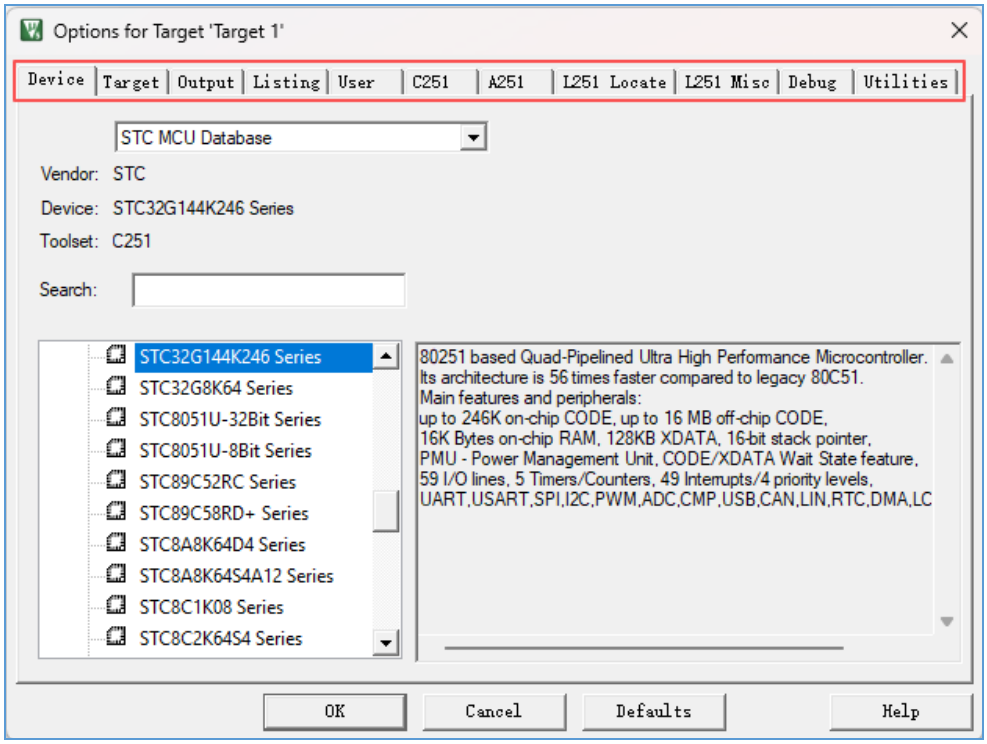


2.7.2.5 Options for Target 'Target 1'（Configure target options），Build Toolbar

Options for Target 'Target 1'（Configure target options）按钮：形似“魔术棒”的一个工具按钮，在 Keil 集成开发环境中，“魔术棒”通常用于快速配置项目设置的工具按钮。这个工具提供了一个图形化界面，让开发者来管理和配置项目的各个方面，如目标设备、输出设置、包含路径、库文件、仿真等。



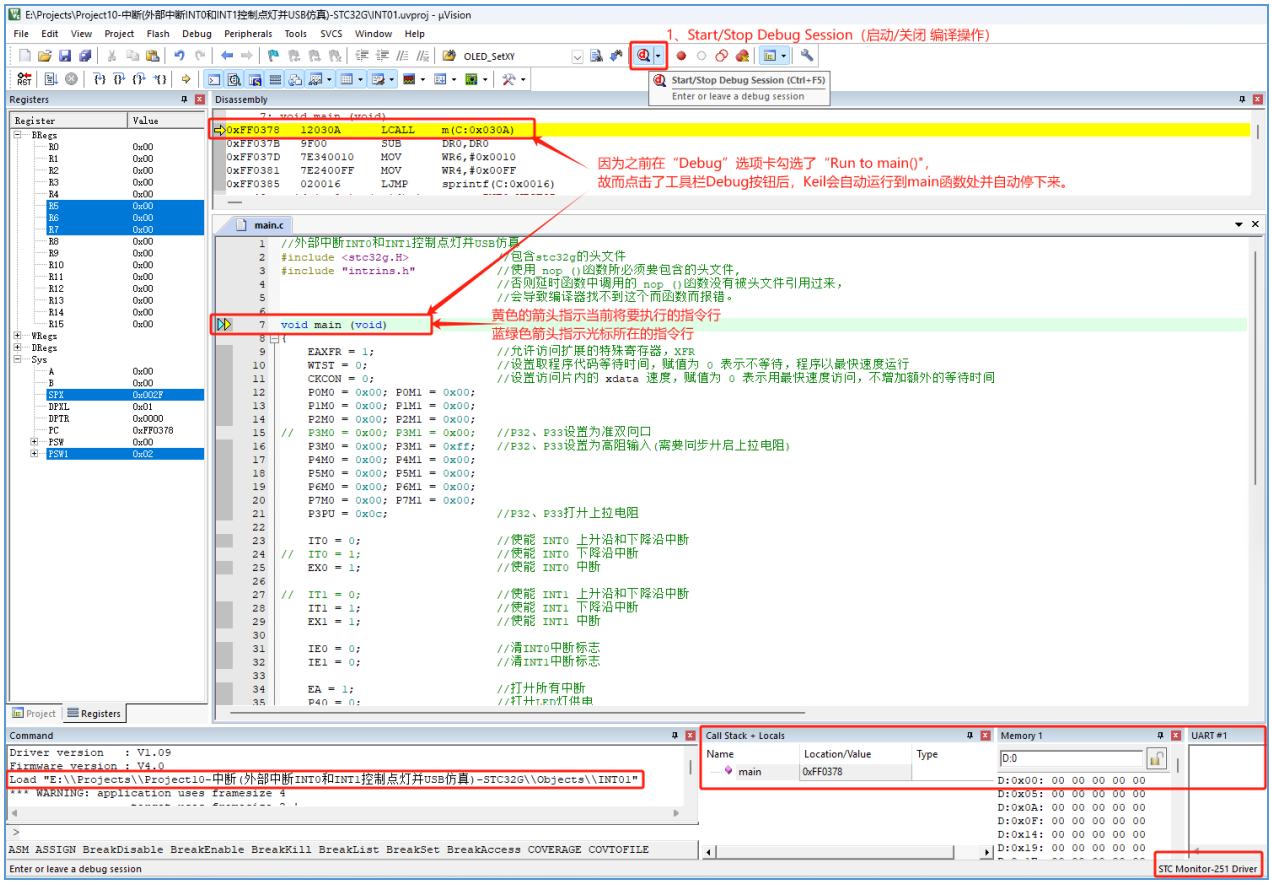
点击“魔术棒”按钮，出现如下浮窗：



开发者可以根据项目的实际情况，进行必要的设置。详见“项目的各种基础选项设置”

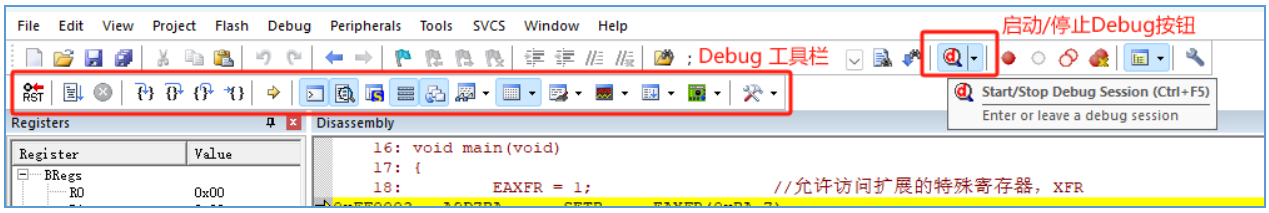
2.7.3 Debug Toolbar（Debug 工具栏）

在 Keil 界面，创建项目并对项目进行前面步骤的仿真设置，代码程序编译无误后，点击 File 工具栏的“Debug”（启动/关闭编译调试操作界面）按钮后，Keil 软件进入仿真界面，如下图：



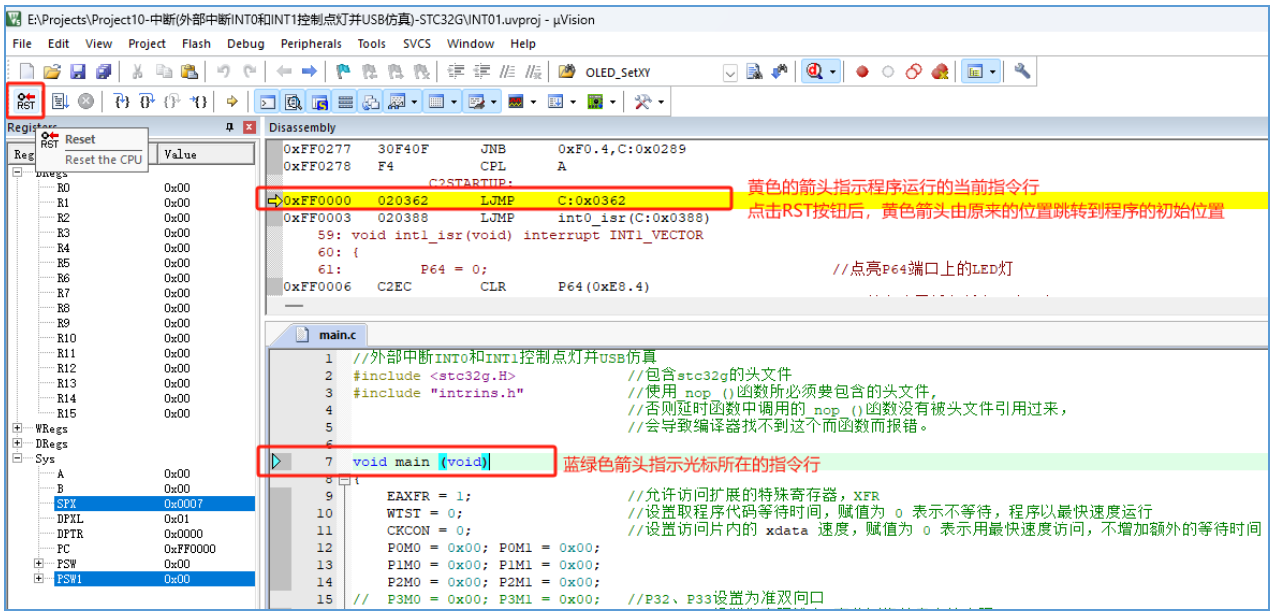
提示：退出 Debug 调试界面，同样点击工具栏的“Debug”按钮。

在 Debug 界面，我们要认识一下 **Debug Toolbar（Debug 工具栏）**的常用调试/仿真按钮：



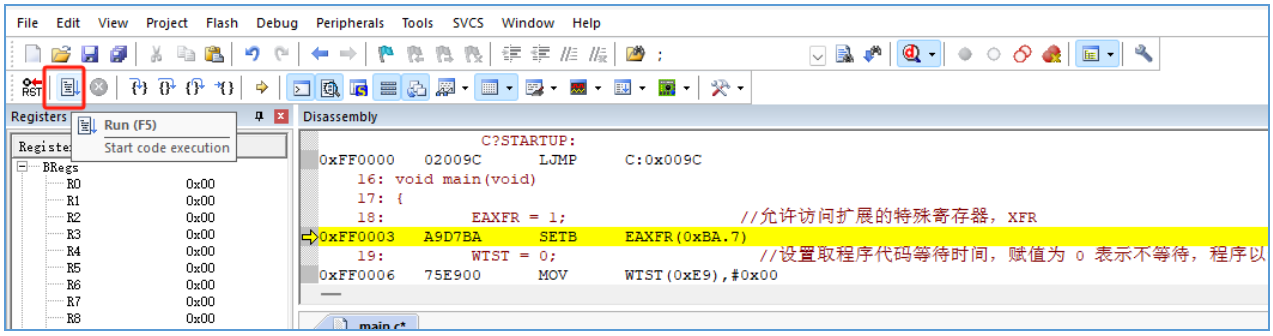
2.7.3.1 RST（Reset the CPU），Debug Toolbar

RST（Reset the CPU）：类似复位按钮，相当于实现了一次硬复位。按下该按钮之后，无论程序代码执行到哪个位置，代码都会重新从头开始执行。



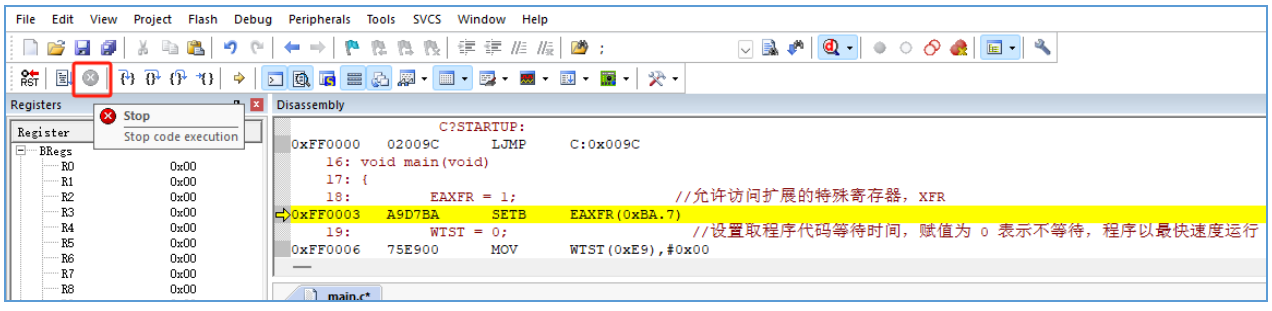
2.7.3.2 Run（Start code execution），Debug Toolbar

Run（Start code execution）：运行按钮。点击该按钮后，程序将一直处于运行状态，或者直接快速运行到断点处（前提是你在查看的地方设置了断点）。有时候你并不需要观看程序的每步是怎么执行的，而是想快速的执行到程序的某个地方看结果，这个按钮就可以实现这样的功能。



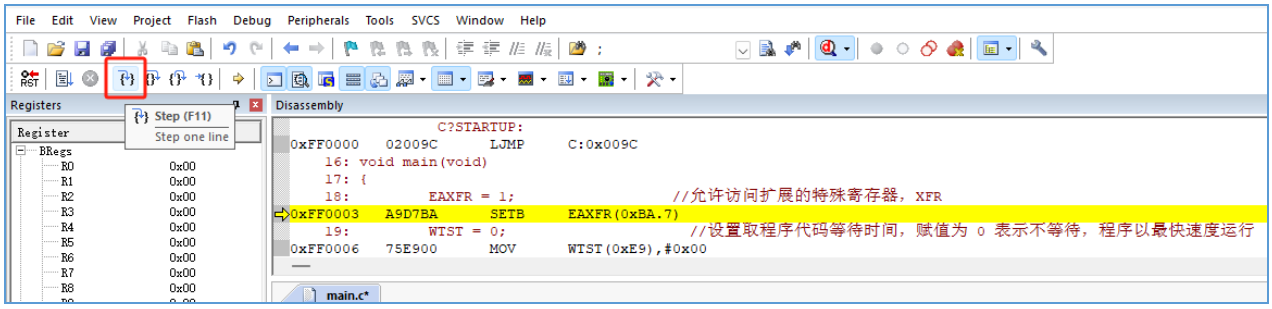
2.7.3.3 Stop（Stop code execution），Debug Toolbar

Stop（Stop code execution）：停止运行按钮。此按钮在程序一直执行的时候会变为有效，通过按该按钮，就可以使程序停止下来，进入到单步调试状态。



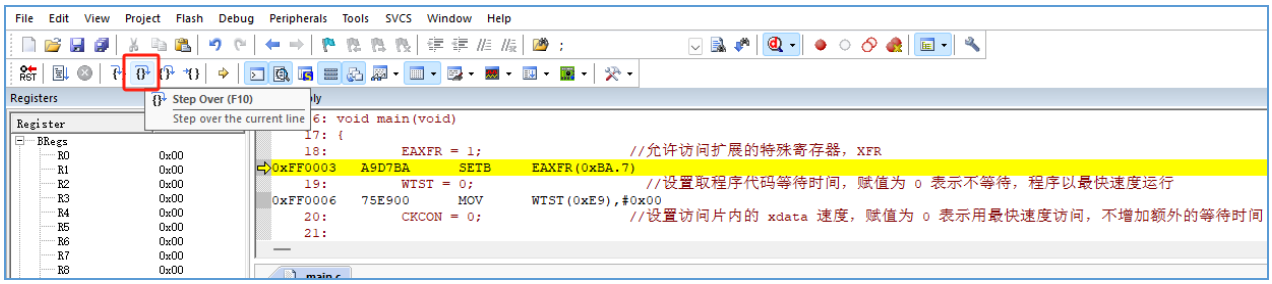
2.7.3.4 Step（Step one line），Debug Toolbar

Step（Step one line）：单步运行按钮。每点击一次该按钮，程序将执行单条语句。如果遇到函数，则会进入函数内部执行。这是调试程序时最常用的功能之一。此按钮在程序停止运行时有效。

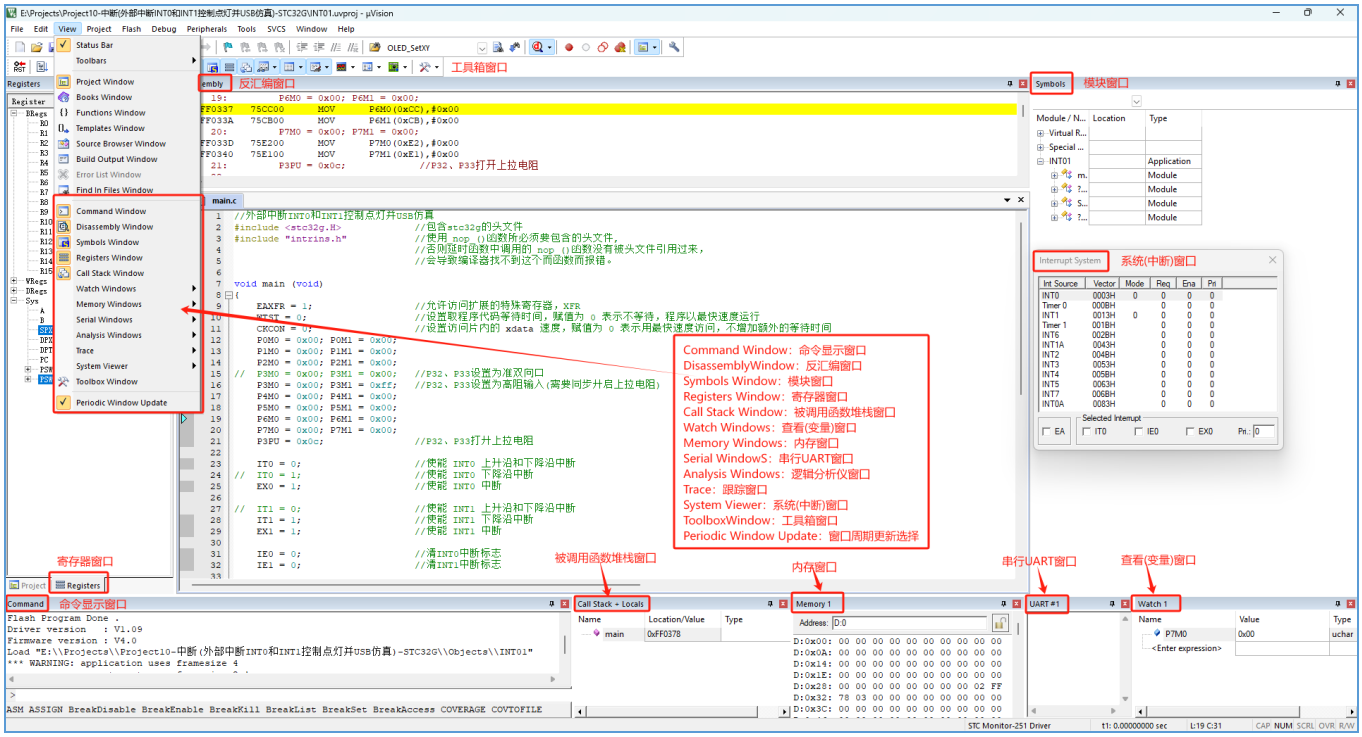


2.7.3.5 Step Over（Step over the current line），Debug Toolbar

Step Over（Step over the current line）：单步跳过按钮，程序运行一行，按单条语句执行。与 Step 按钮不同的是，Step Over 按钮遇到函数不会进入函数内部，而是直接全速运行函数，并跳到下一条语句。在碰到有函数（特别是延时函数）的地方，通过该按钮就可以单步执行过这个函数，而不进入这个函数单步执行。此按钮在程序停止运行时有效。

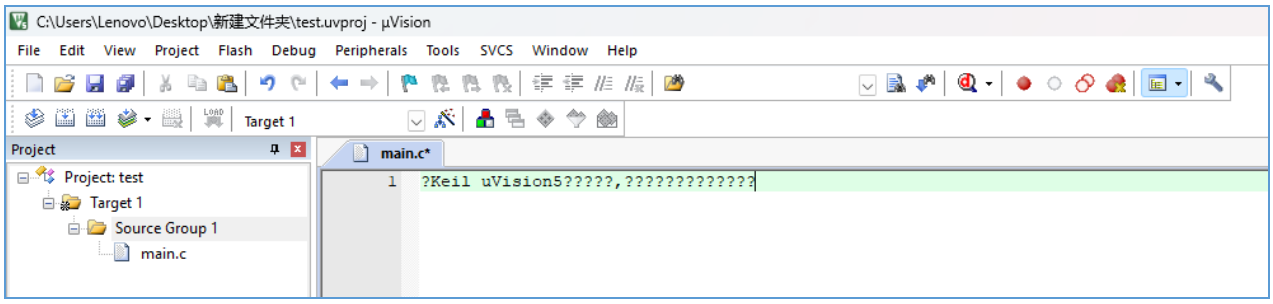


2.7.4 观察变量/寄存器/内存（数据存储器、程序存储器）窗口



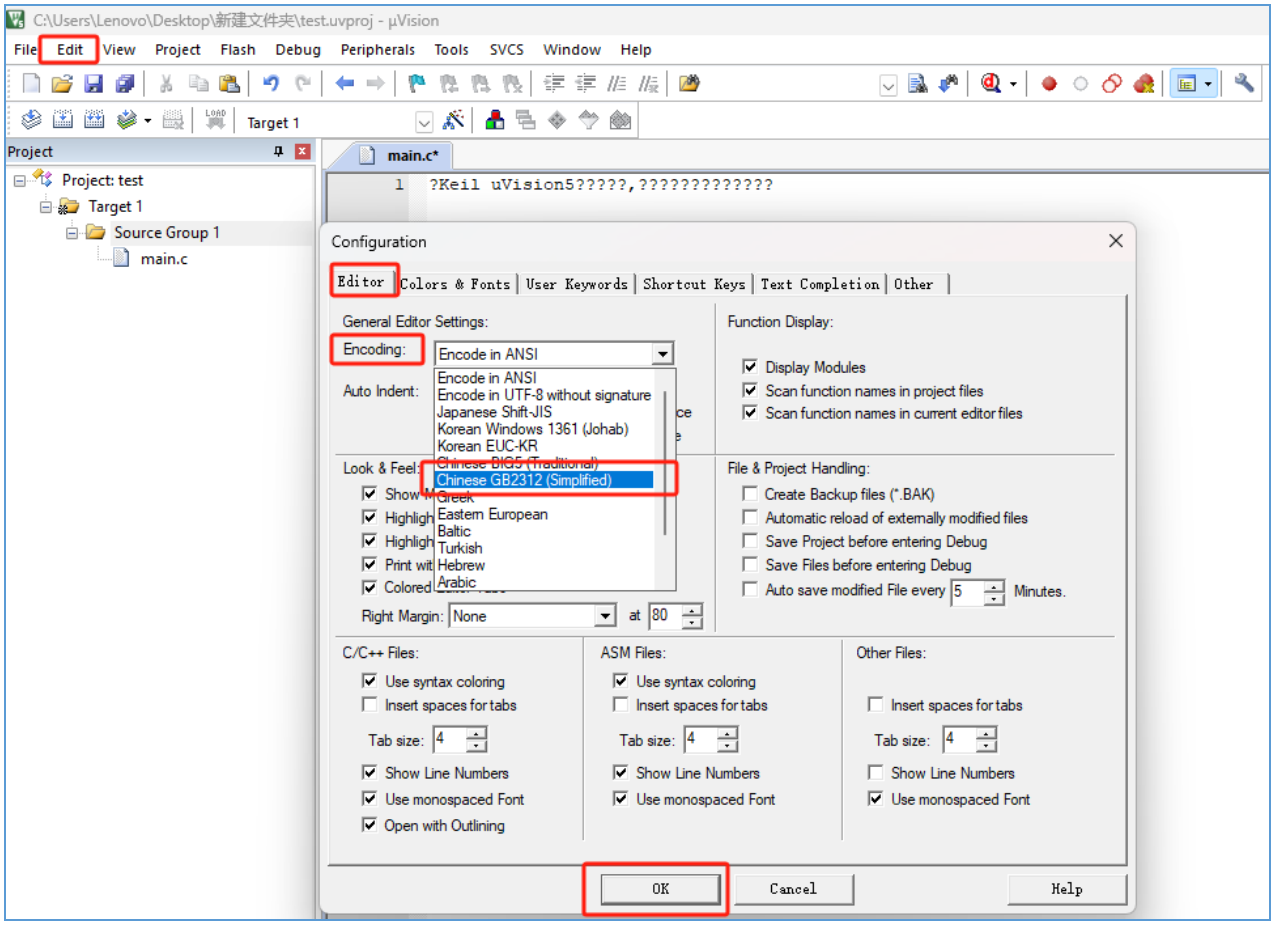
2.8 输入中文出现乱码的解决办法，在 Keil uVision5 的编辑器中

在使用 Keil uVision5 的编辑器中，有输入中文出现乱码的情况，如下图：



这是由于 Keil uVision5 编译器的字符编码设置问题导致的。Keil uVision5 编译器默认情况下使用的是 ANSI 编码，对中文 GB2312 编码不支持，不兼容。按如下步骤更改设置即可恢复正常：

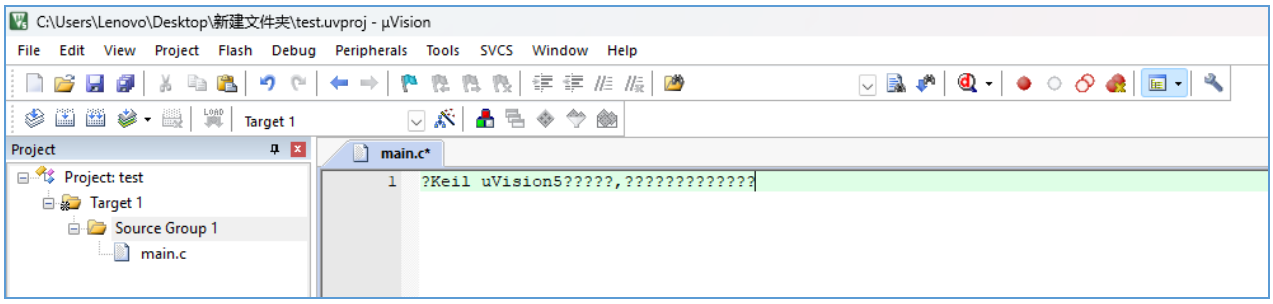
在菜单栏找到并点击 Edit 菜单→选择 Configuration 菜单项→选择 Editor 选项卡→找到 Encoding 的下拉按钮→选择“Chinese GB2312(Simplified)”→点击“OK”，即可。



如果设置后编译器的代码中仍是乱码，则需重新复制粘贴即可正确显示。

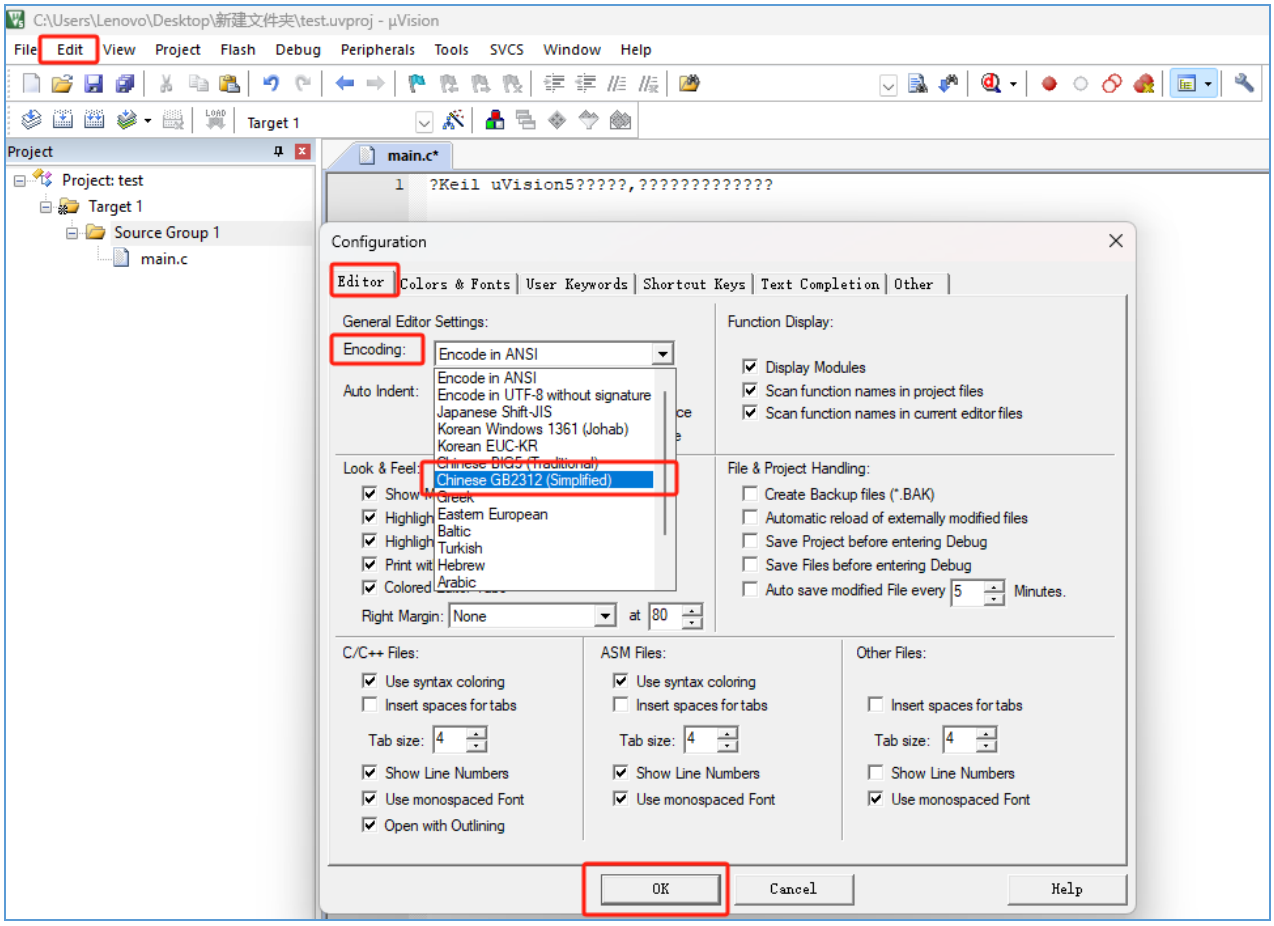
2.9 在 Keil uVision5 的编辑器中，输入中文出现乱码的解决办法

在使用 Keil uVision5 的编辑器中，有输入中文出现乱码的情况，如下图：



这是由于 Keil uVision5 编译器的字符编码设置问题导致的。Keil uVision5 编译器默认情况下使用的是 ANSI 编码，对中文 GB2312 编码不支持，不兼容。按如下步骤更改设置即可恢复正常：

在菜单栏找到并点击 Edit 菜单→选择 Configuration 菜单项→选择 Editor 选项卡→找到 Encoding 的下拉按钮→选择“Chinese GB2312(Simplified)”→点击“OK”，即可。



如果设置后编译器的代码中仍是乱码，则需重新复制粘贴即可正确显示。

2.10 Keil 软件中因含 0xFD 编码的汉字而出现乱码的问题

众所周知，Keil 软件的 8051 和 80251 编译器的所有版本都有一个叫做 0xFD 的问题，主要表现在字符串中不能含有带 0xFD 编码的汉字，否则 Keil 软件在编译时会跳过 0xFD 而出现乱码。

关于这个问题，Keil 官方的回应是：0xfd、0xfe、0xff 这 3 个字符编码被 Keil 编译器内部使用，所以代码中若包含有 0xfd 的字符串时，0xfd 会被编译器自动跳过。

Keil 官方提供的解决方法：在带有 0xfd 编码的汉字后增加一个 0xfd 即可。例如：

```
printf("数学");                    //Keil 编译后打印会显示乱码
printf("数\xfd 学");               //显示正常
```

这里的“\xfd”是标准 C 代码中的转义字符，“\x”表示其后的 1~2 个字符为 16 进制数。“\xfd”表示将 16 进制数 0xfd 插入到字符串中。

由于“数”的汉字编码是 0xCAFD，Keil 在编译时会将 FD 跳过，而只将 CA 编译到目标文件中，后面通过转义字符手动再补一个 0xfd 到目标文件中，就形成完整的 0xCAFD，从而可正常显示。

关于 0xFD 的补丁网上有很多，基本只对旧版本的 Keil 软件有效。打补丁的方法均是在可执行文件中查找关键代码[80 FB FD]，并修改为[80 FB FF]，这种修改方法查找的关键代码过于简单，很容易修改到其它无关的地方，导致编译出来的目标文件运行时出现莫名其妙的问题。所以，代码中的字符串有包含如下的汉字时，建议使用 Keil 官方提供的解决方法进行解决

GB2312 中，包含 0xfd 编码的汉字如下：

褒饼昌除待谍洱俘庚过糊积箭烬君魁
例笼慢谬凝琵讫驱三升数她听妄锡淆
旋妖引育札正铸　佚冽邶埤荤蔟摭啐
帻猃恺泯潺妪纨琮絮辇挛臊忑睚铨稞
瘕颀螨簏酏鲅鳊鼾

另外，Keil 项目路径名的字符中也不能含有带 0xFD 编码的汉字，否则 Keil 软件会无法正确编译此项目。

2.11 C 语言中 printf()函数打印输出数据时，常用的各种输出格式

单片机 C 语言中的 printf_usb()函数和标准 C 语言中 printf()函数格式一致
在%后面根据数据格式需要加关键字：8 位"b", (默认)16 位"h"或不加, 32 位"l"

例如打印有符号十进制数据：
打印 8 位数据：%bd
打印 16 位数据：%hd 或者 %d
打印 32 位数据：%ld

```
printf("cnt8=%bu, ",cnt8++);       //C51 编译器 printf 输出 8 位数据需要使用"%b"
printf("cnt16=%hu, ",cnt16++);     //C51 编译器 printf 输出 16 位数据需要使用"%h"或不加参数
printf("cnt32=%lu\r\n",cnt32++); //C51 编译器 printf 输出 32 位数据需要使用"%l"
```

keil C251 编译器使用 printf 打印数据时，8 位/16 位不用加参数, 32 位加"l"

```
printf("cnt8=%u, ",cnt8++);              //C251 编译器 printf 可直接输出 8 位/16 位数据
printf("cnt16=%u, ",cnt16++);            //C251 编译器 printf 可直接输出 8 位/16 位数据
printf("cnt32=%lu\r\n",cnt32++);       //C251 编译器 printf 输出 32 位数据需要使用"%l"
```

支持数据类型：

Type	Argument Type	Input Format
d	int	Signed decimal number.
u	unsigned int	Unsigned decimal number.
o	unsigned int	Unsigned octal number.
x	unsigned int	Unsigned hexadecimal number using "0123456789abcedf".
X	unsigned int	Unsigned hexadecimal number using "0123456789ABCDEF".
f	float	Floating-point number formatted as <[>-<]>*dddd.dddd*.
e	float	Floating-point number formatted as <[>-<]>*d.ddddd*e<[>-<]>*dd*.
E	float	Floating-point number formatted as <[>-<]>*d.ddddd*E<[>-<]>*dd*.
g	float	Floating-point number using either the e or f format, whichever is more compact for the specified value and precision.
G	float	Floating-point number using either the E or f format, whichever is more compact for the specified value and precision.
c	char	A single character.
s	*	A string of characters terminated by a null character ('\0').
p	*	A generic pointer formatted as *t:aaaa* where *t* is the memory type and *aaaa* is the hexadecimal address.

2.12 扩展 Keil 对中断号数量的支持，中断号大于 31 编译出错的处理

注：Keil C51/C251 编译器只支持 31 以内的中断号（0~31），超过 31 编译会报错。有热心网友提供了一个简单的 Keil 中断号拓展工具，可将中断号拓展到 254。工具界面如下：

但对于目前现有的 Keil 版本，只能使用本章节的方法进行临时解决。

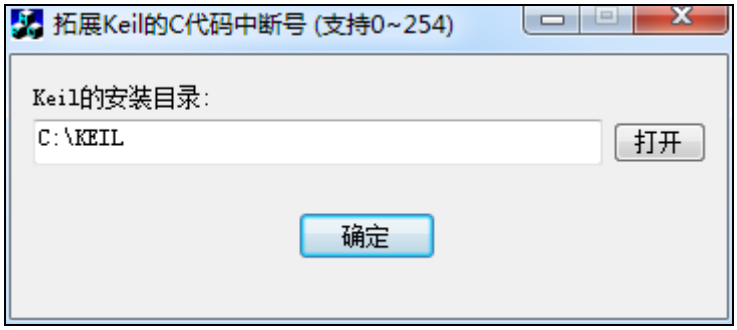
2.12.1 如何下载 Keil 中断号拓展工具，及安装

软件下载地址：https://www.STCAI.com/gjrj



下载完成后，直接解压缩，运行该绿色免安装工具。





点击“打开”按钮，定位到 Keil 的安装目录后，点击“确定”即可。

根据实际的 Keil 安装目录选择地址，如部分用户安装的目录是，C:\Keil_v5

由于 Keil 的版本在不断更新，而早期版本过多，有无法收集齐，这里列举一下已测试通过的 C51.EXE

版本和 C251.EXE 版本

已测试通过的 C51.EXE 版本：

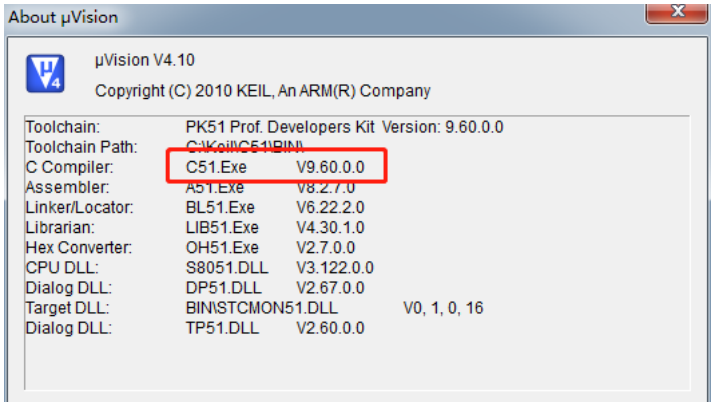
V6.12.0.1
V8.8.0.1
V9.0.0.1
V9.1.0.1
V9.53.0.0
V9.54.0.0
V9.57.0.0
V9.59.0.0
V9.60.0.0

已测试通过的 C251.EXE 版本：

V5.57.0.0
V5.60.0.0

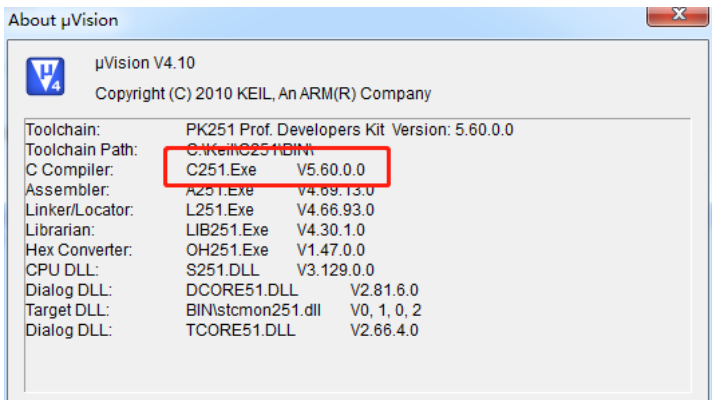
查看 C51.EXE 版本的方法：

在 keil 中打开一个基于 STC8 系列或者 STC15 系列单片机的项目，在 Keil 软件菜单项“Help”中打开“About uVision...”



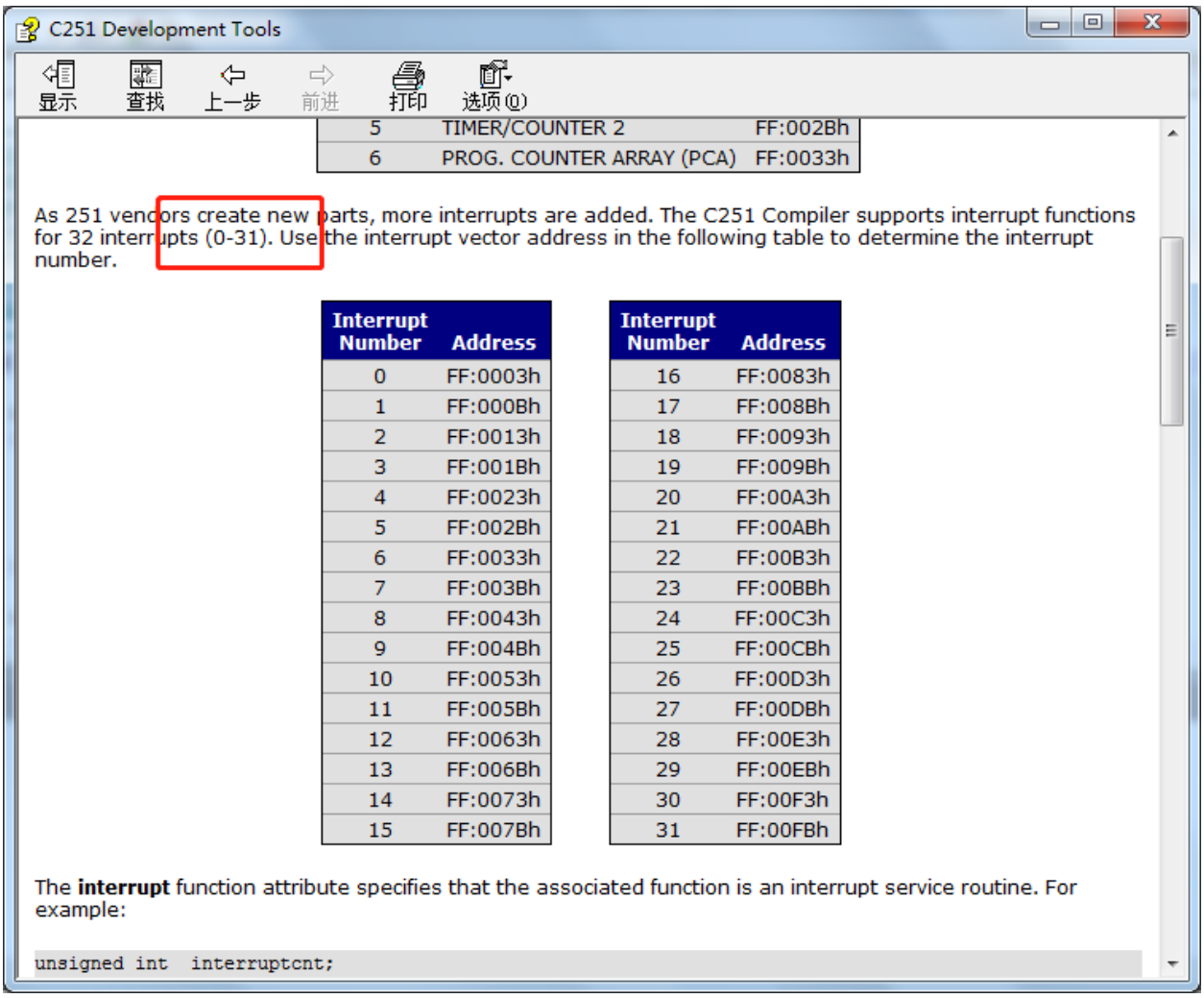
查看 C251.EXE 版本的方法：

在 keil 中打开一个基于 STC32G 系列单片机的项目，在 Keil 软件菜单项“Help”中打开“About uVision...”



2.12.2 不使用扩展中断号工具，则需借用保留中断号进行中转

在 Keil 的 C251 编译环境下，中断号只支持 0~31，即中断向量必须小于 0100H。



下表是 STC 目前所有系列的中断列表：

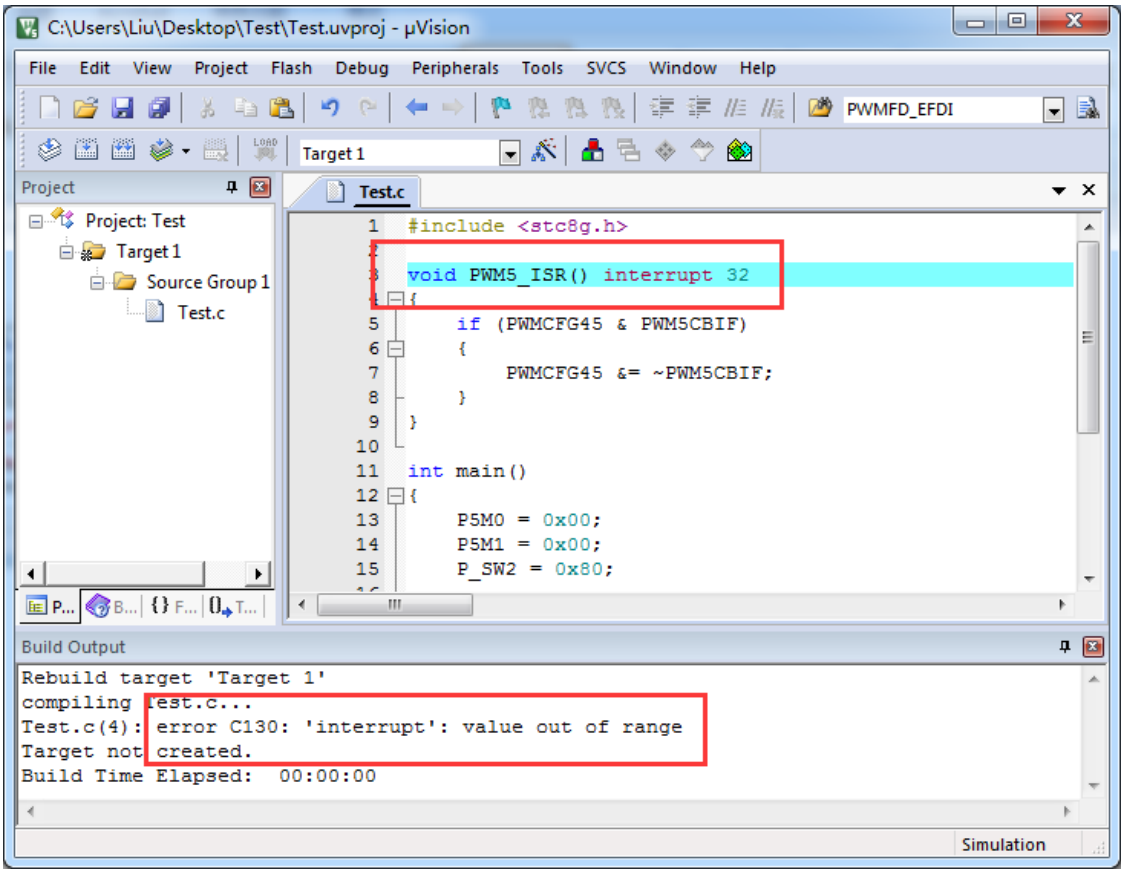
相同优先级内的查询次序	中断向量	中断源
0 （最高）	FF:0003H	INT0/P3.2 中断
1	FF:000BH	Timer0 中断
2	FF:0013H	INT1/P3.3 中断
3	FF:001BH	Timer1 中断
4	FF:0023H	UART1 中断
5	FF:002BH	ADC 中断
6	FF:0033H	LVD 中断
8	FF:0043H	UART2 中断
9	FF:004BH	SPI 中断
10	FF:0053H	INT2/P3.6 中断
11	FF:005BH	INT3/P3.7 中断
12	FF:0063H	Timer2 中断
16	FF:0083H	INT4/P3.0 中断
17	FF:008BH	UART3 中断

相同优先级内的查询次序	中断向量	中断源
18	FF:0093H	UART4 中断
19	FF:009BH	Timer3 中断
20	FF:00A3H	Timer4 中断
21	FF:00ABH	CMP 中断
24	FF:00C3H	I2C 中断
25	FF:00CBH	USB 中断
26	FF:00D3H	PWMA 中断
27	FF:00DBH	PWMB 中断
28	FF:00E3H	CAN1 中断
29	FF:00EBH	CAN2 中断
30	FF:00F3H	LIN1 中断
31	FF:00FBH	LIN2 中断
36	FF:0123H	RTC 中断
37	FF:012BH	P0 中断
38	FF:0133H	P1 中断
39	FF:013BH	P2 中断
40	FF:0143H	P3 中断
41	FF:014BH	P4 中断
42	FF:0153H	P5 中断
43	FF:015BH	P6 中断
44	FF:0163H	P7 中断
45	FF:016BH	P8 中断
46	FF:0173H	P9 中断
47	FF:017BH	DMA_M2M 中断
48	FF:0183H	DMA_ADC 中断
49	FF:018BH	DMA_SPI 中断
50	FF:0193H	DMA_UR1T 中断
51	FF:019BH	DMA_UR1R 中断
52	FF:01A3H	DMA_UR2T 中断
53	FF:01ABH	DMA_UR2R 中断
54	FF:01B3H	DMA_UR3T 中断
55	FF:01BBH	DMA_UR3R 中断
56	FF:01C3H	DMA_UR4T 中断
57	FF:01CBH	DMA_UR4R 中断
58	FF:01D3H	TFT 彩屏 DMA 中断
59	FF:01DBH	TFT 彩屏中断
60	FF:01E3H	DMA_I2CT 中断
61	FF:01EBH	DMA_I2CR 中断
62	FF:01F3H	I2S 中断
63	FF:01FBH	DMA_I2ST 中断
64	FF:0203H	DMA_I2SR 中断
65	FF:020BH	DMA_QSPI 中断
66	FF:0213H	QSPI 中断

相同优先级内的查询次序	中断向量	中断源
67	FF:021BH	Timer11
68	FF:0223H	DMA_I2C2T 中断
69	FF:022BH	DMA_I2C2R 中断
70	FF:0233H	DMA_I2S2T 中断
71	FF:023BH	DMA_I2S2R 中断
72	FF:0243H	DMA_PWMAT 中断
73	FF:024BH	DMA_PWMAR 中断
74	FF:0253H	DMA_PWMCT 中断
75	FF:025BH	DMA_PWMCR 中断
76	FF:0263H	DMA_ADC2 中断
77	FF:026BH	DMA_DAC1 中断
78	FF:0273H	DMA_DAC2 中断
79	FF:027BH	DMA_SPI2 中断
80	FF:0283H	DMA_SPI3 中断
82	FF:028BH	DMA_UR5T 中断
83	FF:0293H	DMA_UR5R 中断
84	FF:029BH	DMA_UR6T 中断
85	FF:02A3H	DMA_UR6R 中断
86	FF:02ABH	DMA_UR7T 中断
87	FF:02B3H	DMA_UR7R 中断
88	FF:02BBH	DMA_UR8T 中断
89	FF:02C3H	DMA_UR8R 中断
90	FF:02CBH	PA 中断
91	FF:02D3H	PB 中断
96	FF:0303H	Timer5 中断
		Timer6 中断
97	FF:030BH	Timer7 中断
		Timer8 中断
98	FF:0313H	Timer9 中断
		Timer10 中断
99	FF:031BH	Timer17 中断
		Timer18 中断
102	FF:0333H	UART5 中断
103	FF:033BH	UART6 中断
104	FF:0343H	UART7 中断
105	FF:034BH	UART8 中断
106	FF:0353H	ADC2 中断
107	FF:035BH	DAC1 中断
108	FF:0363H	DAC2 中断
109	FF:036BH	I2C2 中断
110	FF:0373H	I2S2 中断
111	FF:037BH	SPI2 中断
112	FF:0383H	SPI3 中断

相同优先级内的查询次序	中断向量	中断源
114	FF:0393H	CMP2 中断
115	FF:039BH	CMP3 中断
116	FF:03A3H	CMP4 中断
117	FF:03ABH	DMA_CAN1T 中断
118	FF:03B3H	DMA_CAN1R 中断
119	FF:03BBH	DMA_CAN2T 中断
120	FF:03C3H	DMA_CAN2R 中断
121	FF:03CBH	PWMC 中断
122	FF:03D3H	PWMD 中断
123	FF:03DBH	PWME 中断
124	FF:03E3H	PWMF 中断
125	FF:03EBH	DMA_PWMET 中断
126	FF:03F3H	DMA_PWMER 中断

不难发现，RTC 中断开始，后面所有的中断服务程序，在 keil 中均会编译出错，如下图所示：



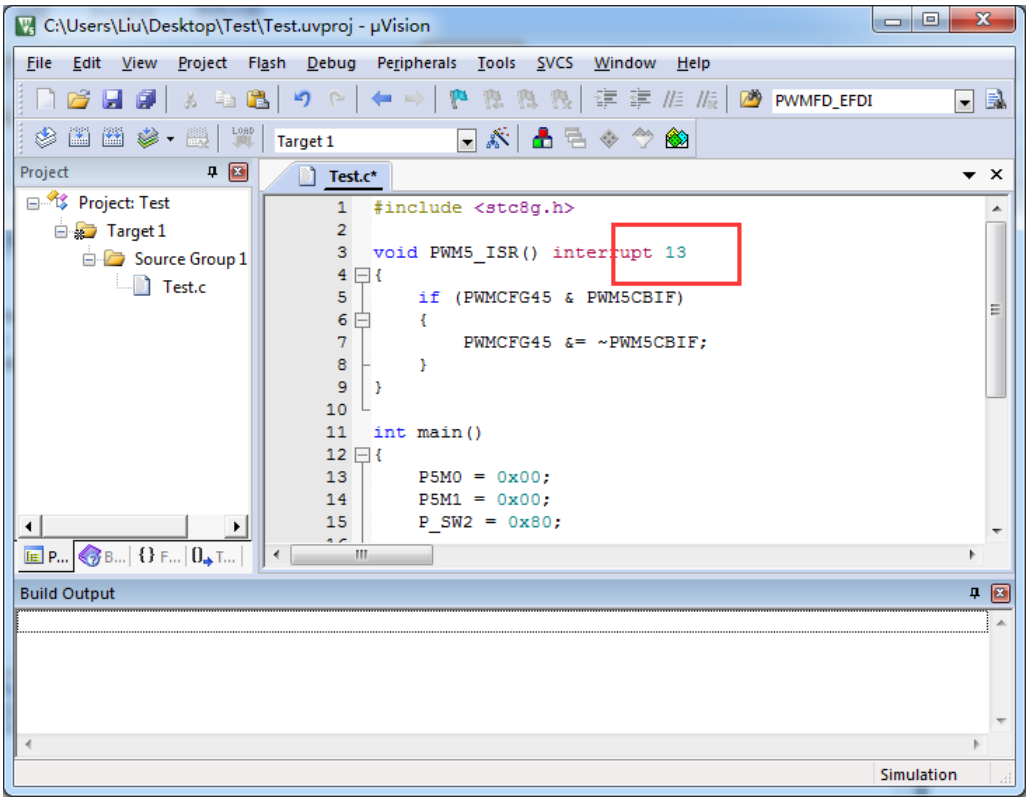
处理这种错误有如下三种方法：（均需要借助于汇编代码，优先推荐使用方法 1）

方法 1：借用 13 号中断向量

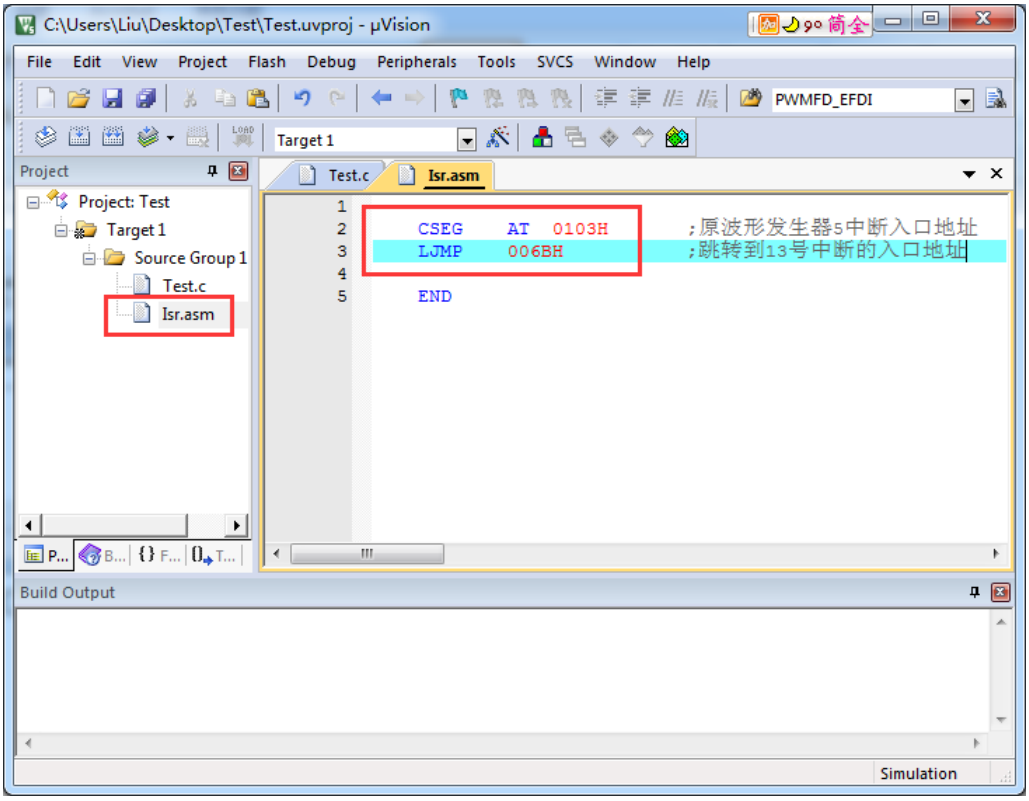
0~31 号中断中，第 13 号是保留中断号，我们可以借用此中断号

操作步骤如下：

1、将我们报错的中断号改为“13”，如下图：

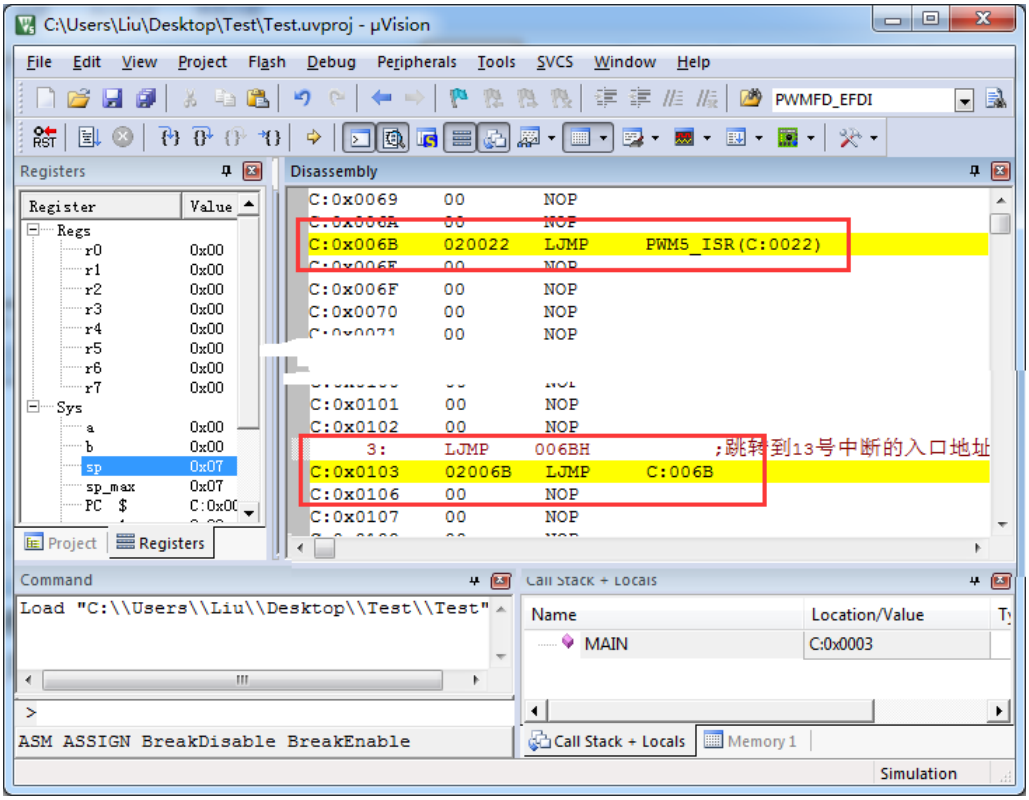


2、新建一个汇编语言文件，比如“isr.asm”，加入到项目，并在地址“0103H”的地方添加一条“LJMP 006BH”，如下图：

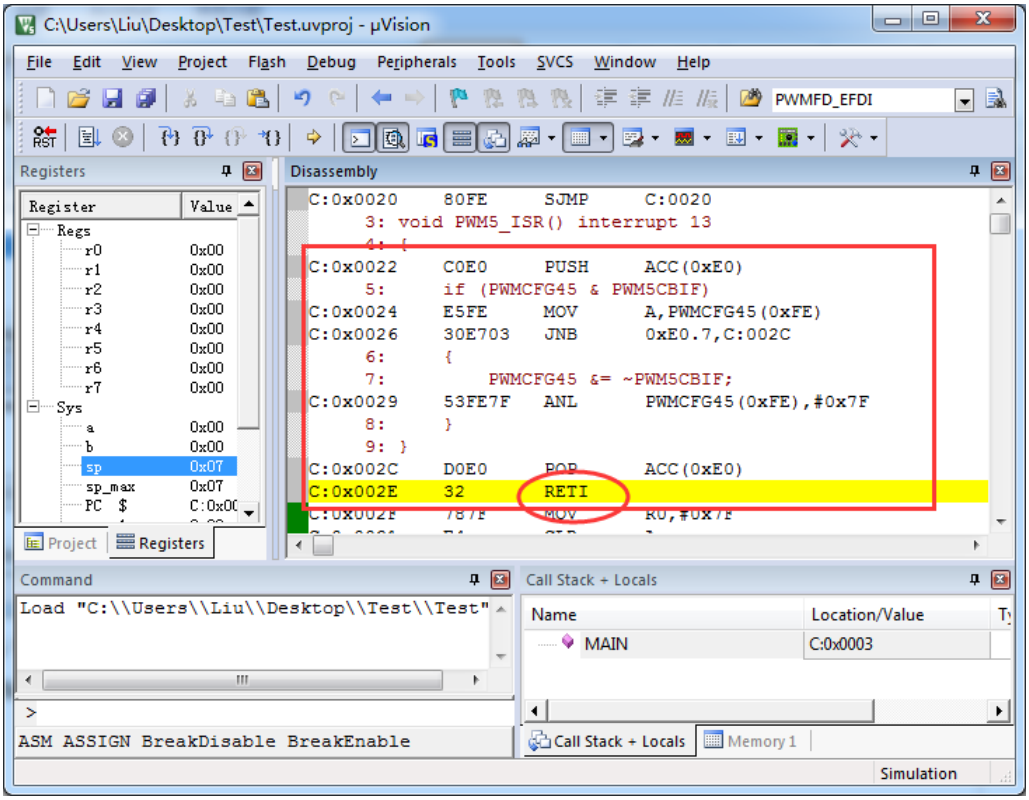


3、编译即可通过。

此时经过 Keil 的 C51 编译器编译后，在 006BH 处有一条“LJMP　　PWM5_ISR”，在 0103H 处有一条“LJMP　　006BH”，如下图：



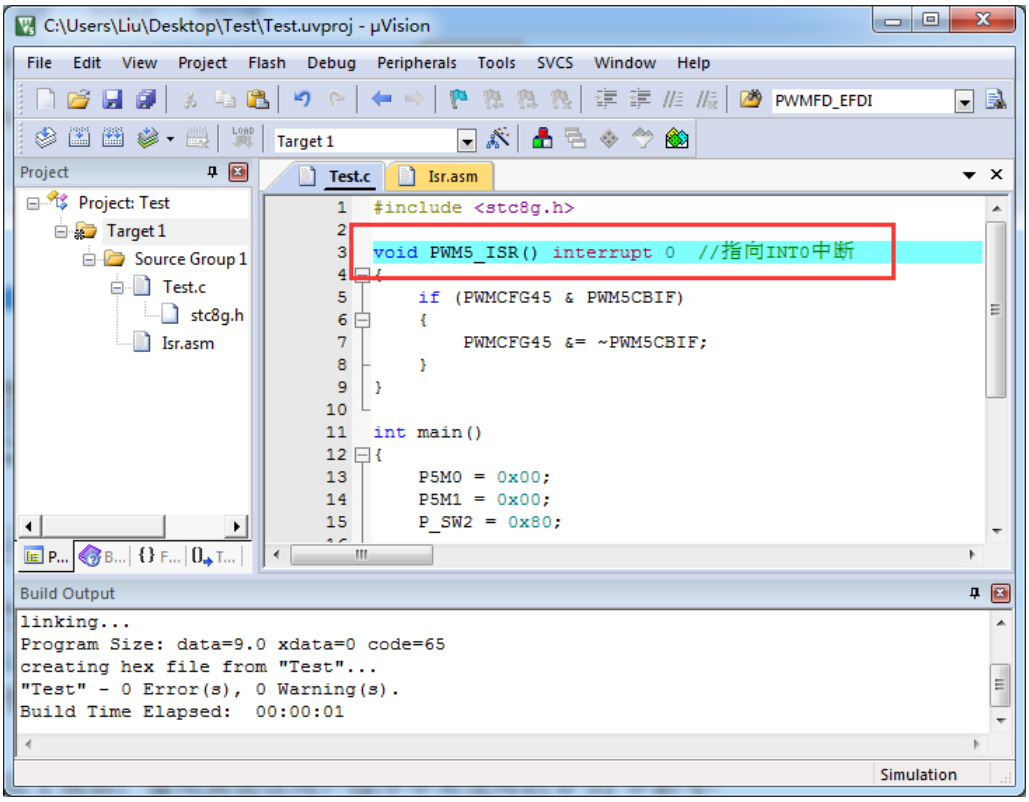
当发生 PWM5 中断时，硬件会自动跳转到 0103H 地址执行“LJMP　　006BH”，然后在 006BH 处再执行“LJMP　　PWM5_ISR”即可跳转到真正的中断服务程序，如下图：

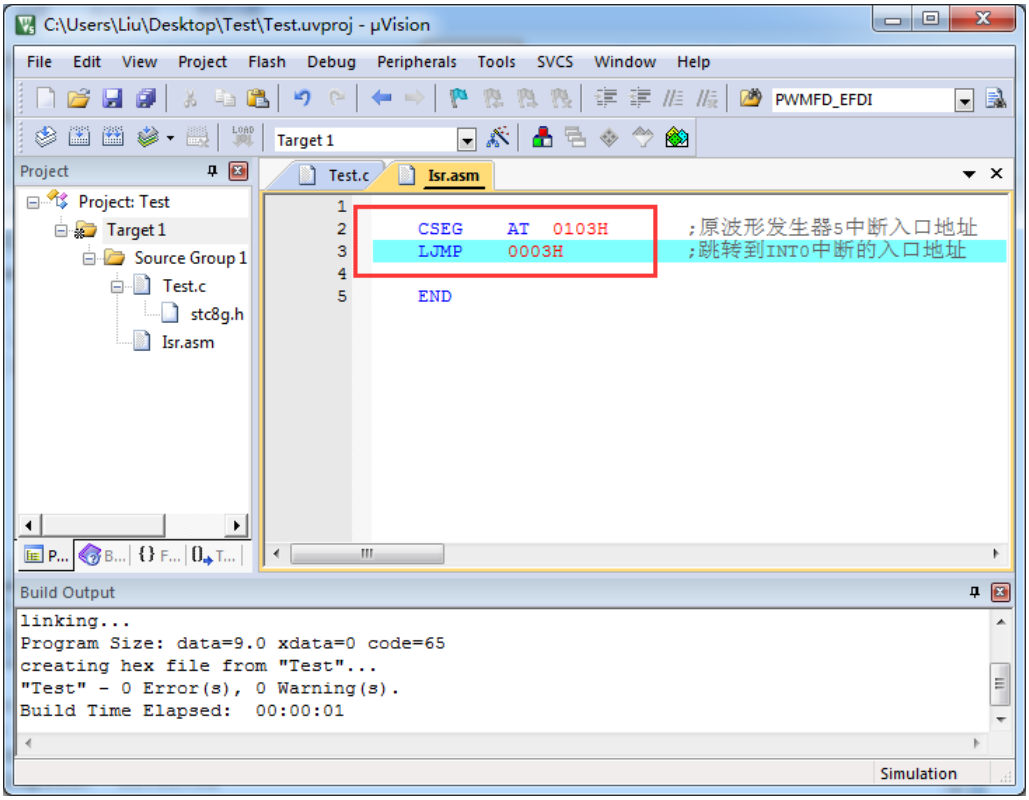


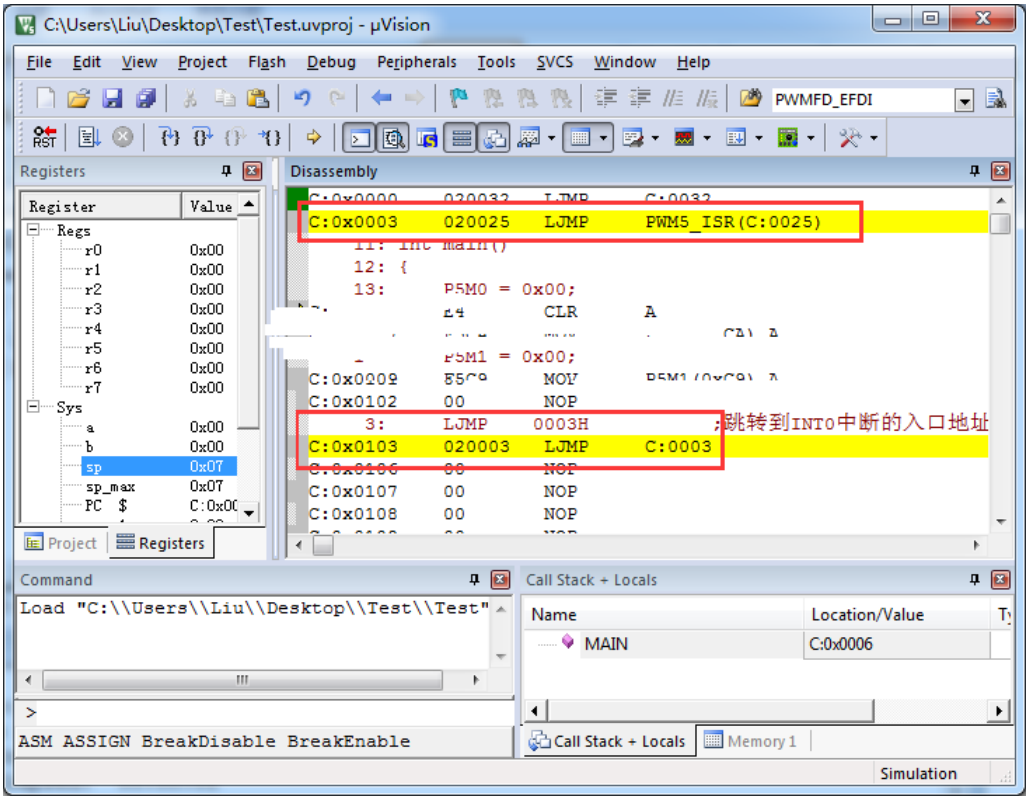
中断服务程序执行完成后，再通过 RETI 指令返回。整个中断响应过程只是多执行了一条 LJMP 语句而已。

方法 2：与方法 1 类似，借用用户程序中未使用的 0~31 的中断号

比如在用户的代码中，没有使用 INT0 中断，则可将上面的代码作类似与方法 1 的修改：





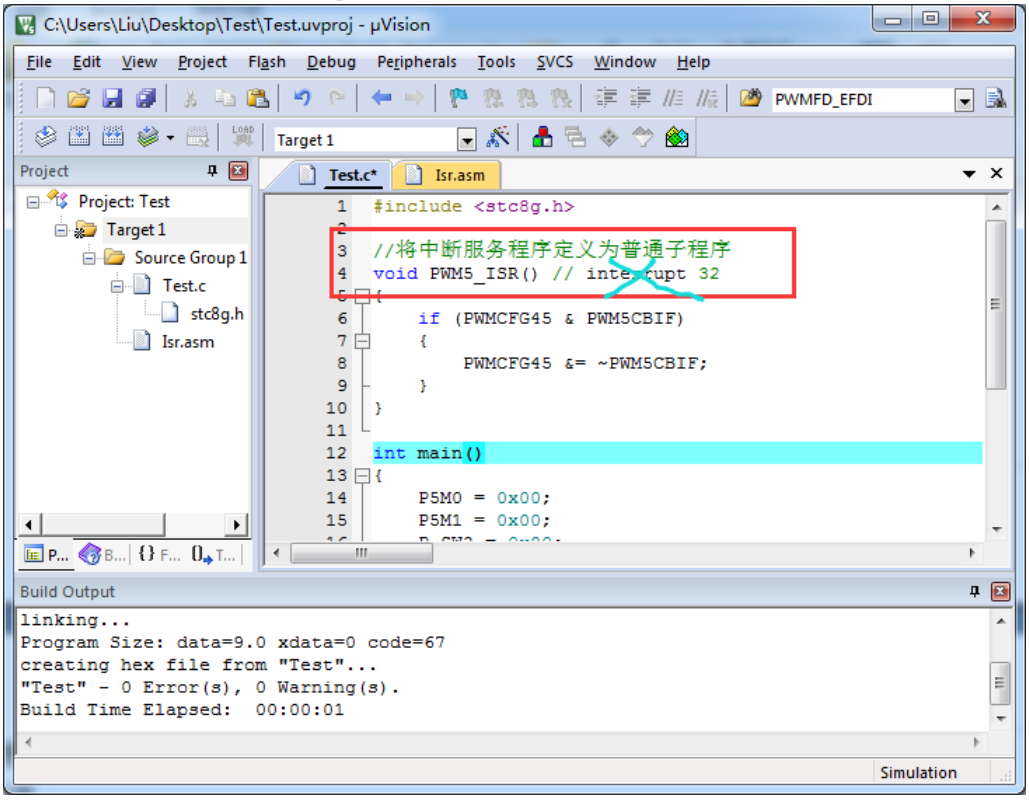


执行效果与方法 1 相同，此方法适用于需要重映射多个中断号大于 31 的情况。

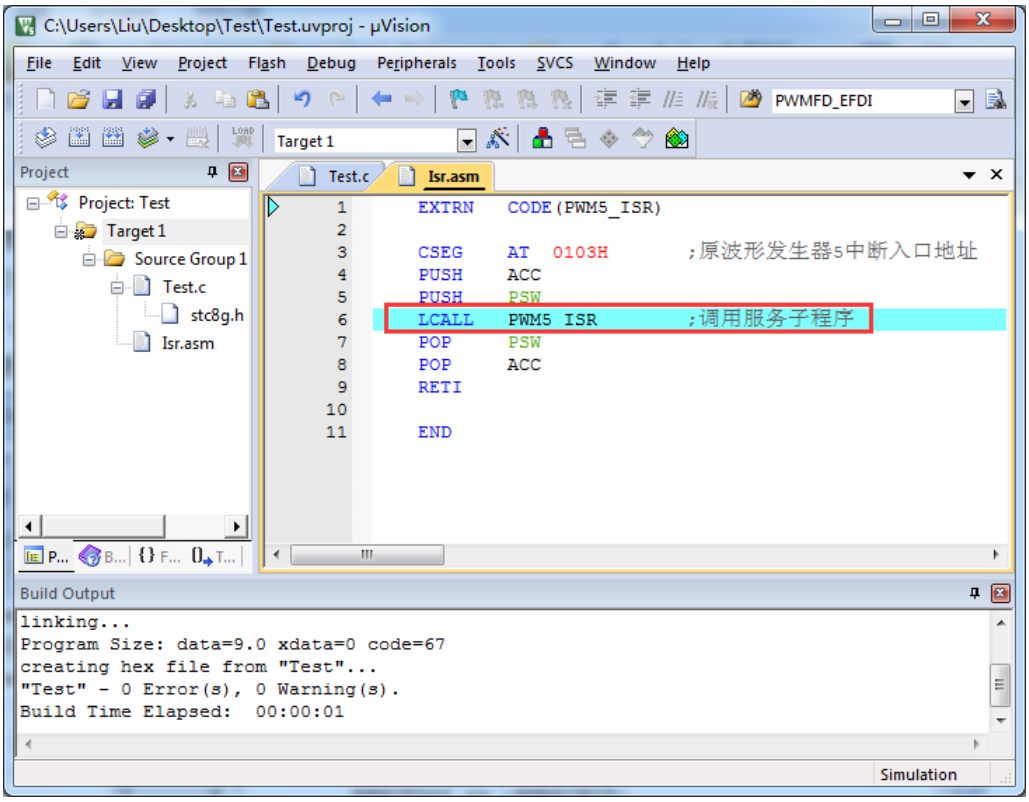
方法 3：将中断服务程序定义成子程序，然后在汇编代码中的中断入口地址中使用 LCALL 指令执行服务程序

操作步骤如下：

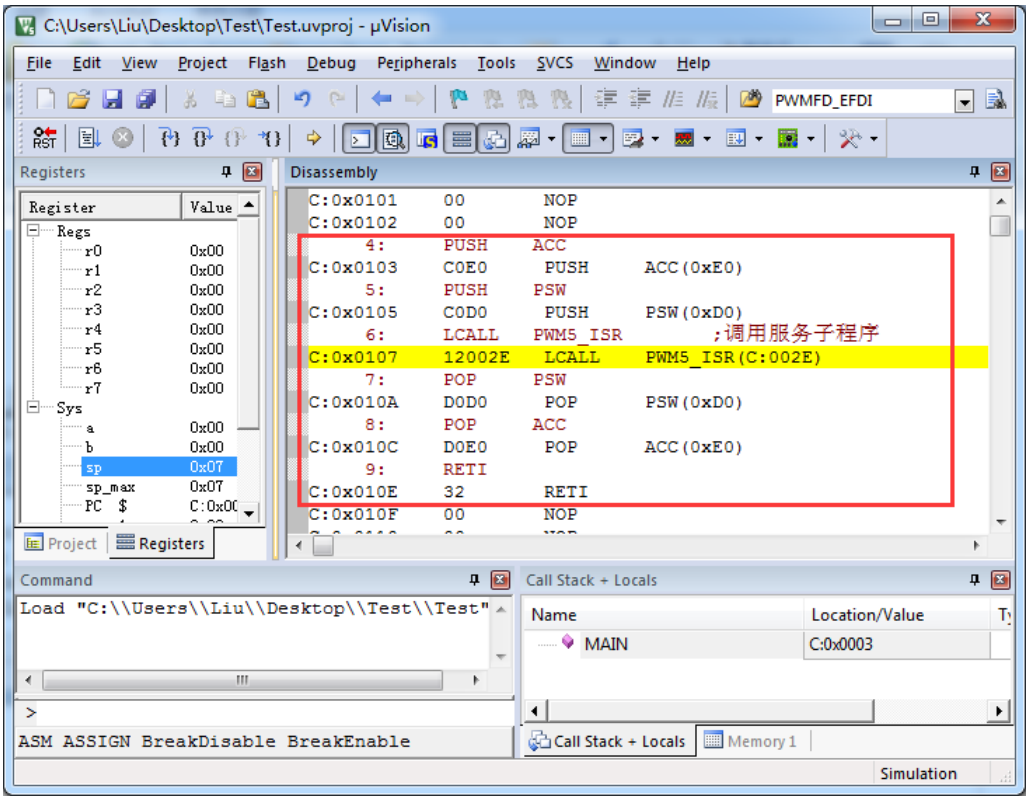
1、首先将中断服务程序去掉“interrupt”属性，定义成普通子程序



2、然后在汇编文件的 0103H 地址输入如下图所示的代码



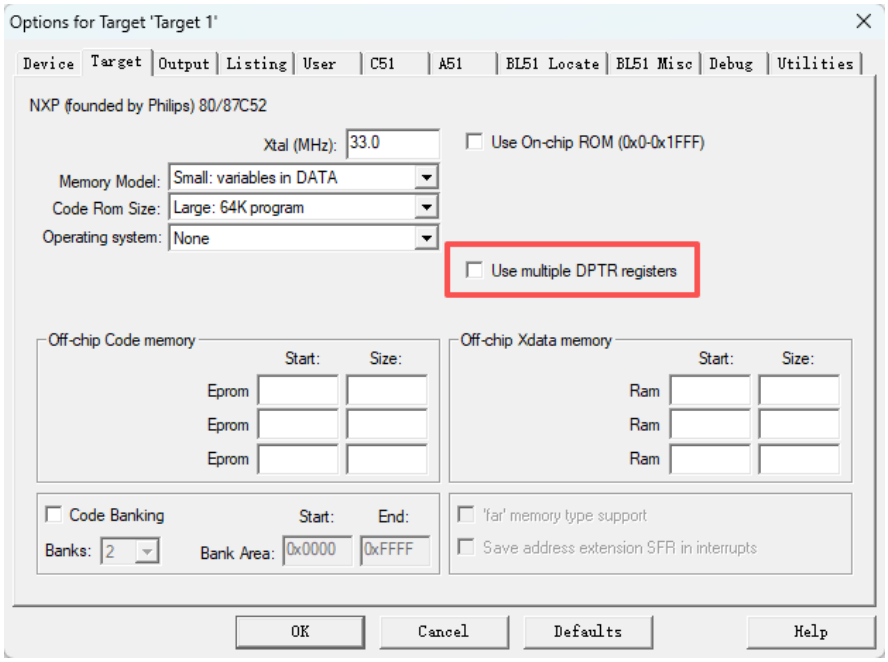
3、编译通过后，即可发现在 0103H 地址的地方即为中断服务程序



此方法不需要重映射中断入口，不过这种方法有一个问题，在汇编文件中具体需要将哪些寄存器压入堆栈，需要用户查看 C 程序的反汇编代码来确定。一般包括 PSW、ACC、B、DPL、DPH 以及 R0~R7。除 PSW 必须压栈外，其他哪些寄存器在用户子程序中有使用，就必须将哪些寄存器压栈。

2.13 关于 Keil 的双数据指针编译选项

Keil 内建的 C51 库函数中，类似 strcpy、memcpy 等数据复制函数均有两个版本，一个是使用单数据指针（DPTR）的版本，另外一个是使用双数据指针（DPTR0 和 DPTR1）的版本，两个版本的选择 Keil 提供了如下的编译开关：



不勾选“Use multiple DPTR registers”选项时，Keil 编译器会选择使用单数据指针（DPTR）的库函数；勾选则编译器则会选择使用双数据指针（DPTR0 和 DPTR1）的库函数。

使用单数据指针的库函数兼容性好，可兼容所有的 8051 内核的单片机，但速度相对较慢；使用双数据指针的库函数速度相对比较快，但兼容性不好，要使用 Keil 默认的数据指针的库函数的前提条件是：双数据指针相关的特殊功能寄存器必须和 Intel 标准的寄存器完全兼容。

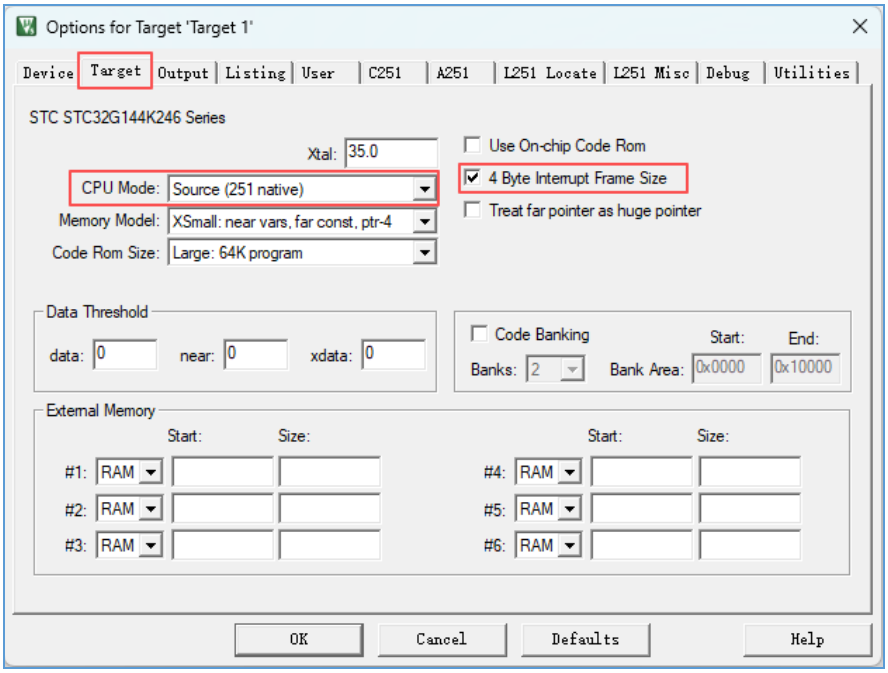
早期的 STC89/STC90/STC10/STC11/STC12C5A/STC15F/STC5W 等型号的双数据指针相关的特殊功能寄存器和 Intel 完全兼容，编译时可以勾选“Use multiple DPTR registers”编译选项。
最近新的 STC8C/STC8H/STC8G/STC12H/STC15H 等 8051 内核的单片机，内部也集成了双数据指针，但寄存器定义和 Intel 不兼容，编译时不能勾选“Use multiple DPTR registers”编译选项，若错误的勾选了这个选项，程序中调用 strcpy、memcpy 等函数会出现数据复制出错。

强烈建议使用官方的 ISP 软件安装 STC 的单片机型号。新建项目时选择正确的单片机型号。官方的型号库中，对支持双数据指针的型号会开放“Use multiple DPTR registers”这个选项。对于无法使用 Keil 的双数据指针函数的型号，则会自动屏蔽“Use multiple DPTR registers”。

2.14 设置超 64K 程序代码的项目（Source 模式）

2.14.1 设置 -- “CPU Mode”选择 Source 模式

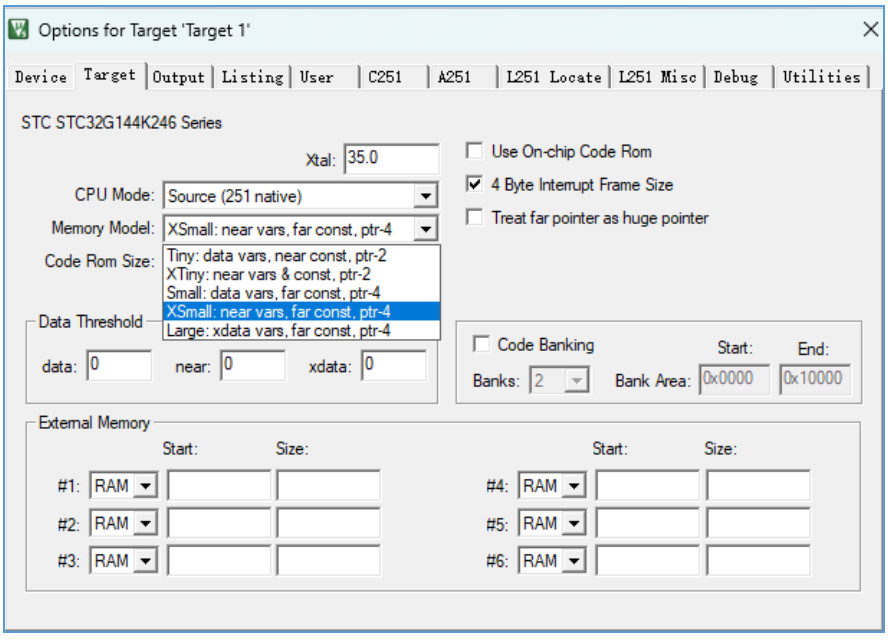
点击 Build 工具栏的“魔术棒”按钮，弹出的“Options for Target ' Target 1'”浮窗，选择“Target”选项页，在“CPU Mode”的下拉选项中选择“Source (251 Native)”（默认）



80251 的指令模式有“Binary”和“Source”两种模式，STC32G144K246 系列目前支持这两种模式。由于 STC32G144K246 系列单片机在中断中的压栈和出栈都是 4 字节模式，建议“4 Byte Interrupt Frame Size”选项也打上钩。

2.14.2 设置 -- “Memory Model”选择 XSmall 模式

在“Memory Model”的下拉选项中选择“XSmall: ...”模式。80251 的存储器模式，在 Keil 环境下有如下图所示的 5 种模式：



各种模式对比如下表：

Memory Model	默认变量类型（数据存储器）	默认常量类型（程序存储器）	默认指针变量	指针访问范围
Tiny 模式	data	near	2 字节	00:0000 ~ 00:FFFF
XTiny 模式	edata	near	2 字节	00:0000 ~ 00:FFFF
Small 模式	data	far	4 字节	00:0000 ~ FF:FFFF
XSmall 模式	**edata**	**far**	**4 字节**	**00:0000 ~ FF:FFFF**
Large 模式	xdata	far	4 字节	00:0000 ~ FF:FFFF

由于 STC32G144K246 的程序逻辑地址为 FC:2800H~FF:FFFFH，需要使用 24 位地址线才能正确访问，默认的常量类型（程序存储器类型）必须使用“far”类型，默认指针变量必须为 4 字节。

不建议使用“Small”“Tiny”和“XTiny”模式，推荐使用“XSmall”模式，这种模式默认将变量定义在内部 RAM(edata)，单时钟存取，访问速度快，且 STC32G144K246 系列芯片有 16K 的 edata 可以使用；使用“Small”模式时，默认将变量定义在内部 RAM(data)，data 默认只有 128 字节，当用户对 RAM 需求超过 128 字节时，Keil 编译器会报错，data 区数量有限，容易报错，所以不建议使用；不推荐使用“Large”模式，虽然该模式也能正确访问 STC32G144K246 的全部 16M 寻址空间，但“Large”模式默认将变量定义在内部扩展 RAM(xdata)里面，存取需要 2~3 个时钟，访问速度慢。

STC32G144K246 系列单片机的 C 语言变量声明建议：

1、 当用户变量需求量较小时，建议不要使用“edata、xdata”等关键字声明变量，而使用如下方式直接声明变量：

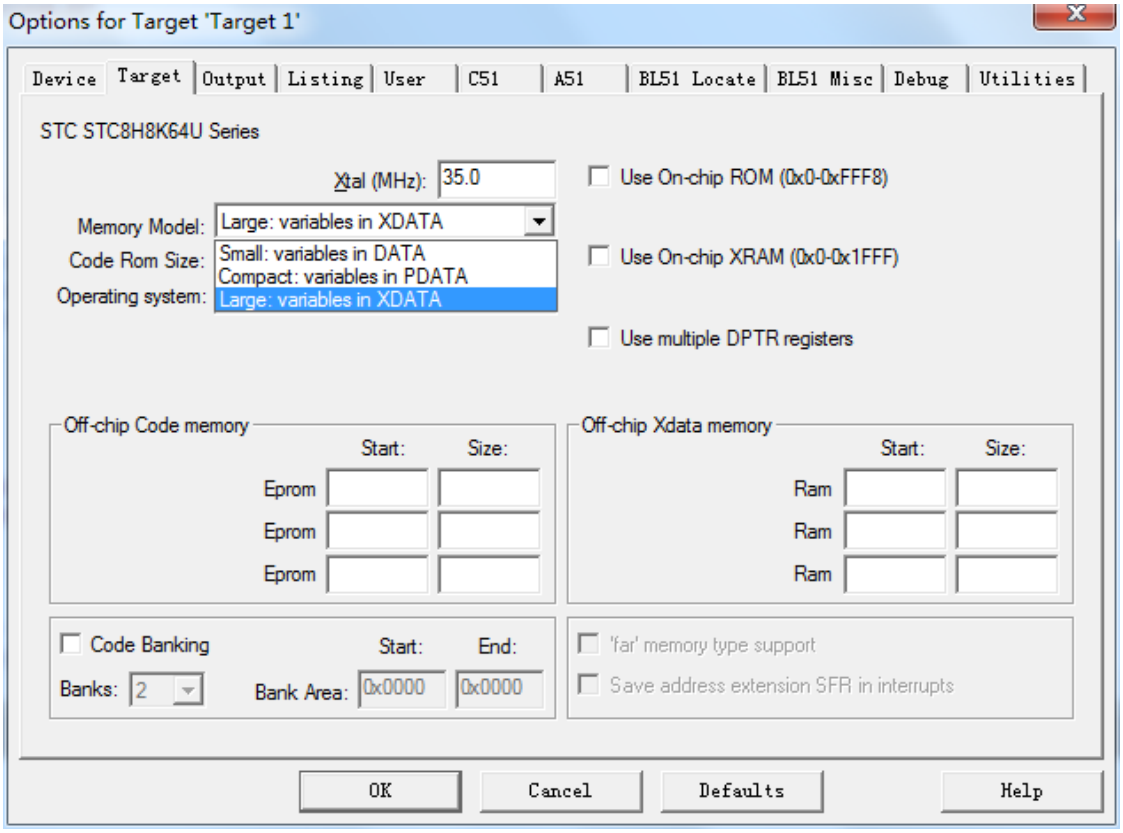
```
char    bCounter = 1;       //声明字节变量
int     wCounter = 100;     //声明双字节变量
long    dwCounter = 0x1234; //声明 4 字节变量
```

然后在项目选项中将“Memory Model”设置为“XSmall”，让编译器自动将声明的变量分配到 edata 区域

2、 当用户变量需求量接近或超过（单片机 edata 容量-1K）的大小时，建议将超出部分使用“xdata”关键字强制分配到 xdata 区域，如下所示：

```
int  xdata  pBuffer = 5;        //使用 xdata 关键字强制分配到 xdata 区域
```

与之相对应的 STC8H8K64U 系列，在 Keil 软件中的“Memory Model”有如下 3 个选择



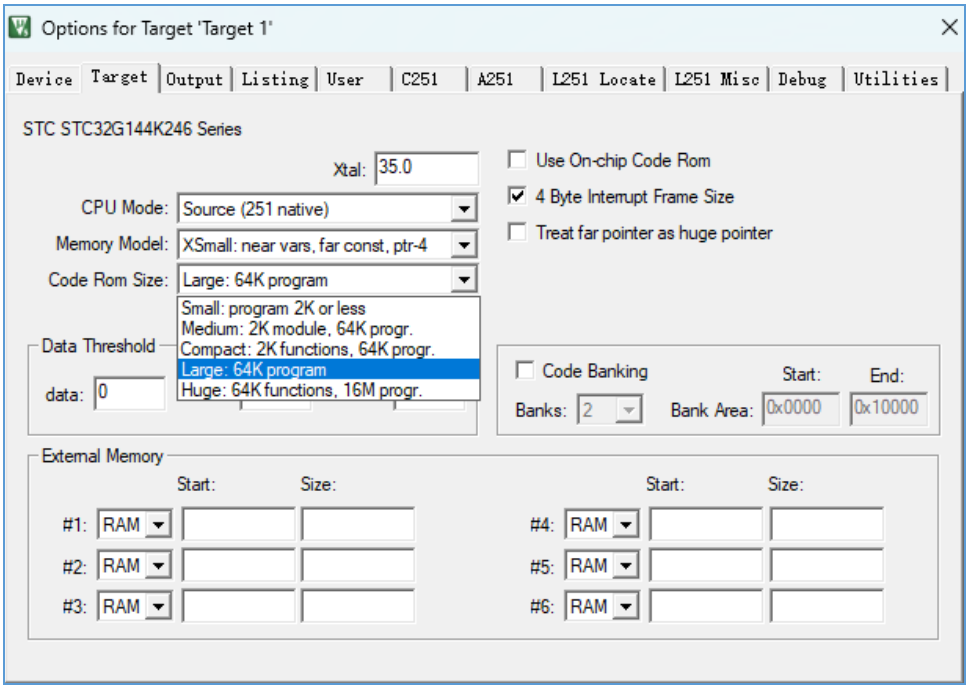
各种模式对比如下表：

Memory Model	默认变量类型（数据存储器）	存储器大小	地址范围
Small 模式	data	128 字节	D:00 ~ D:7F
Compact 模式	pdata	256 字节	X:0000 ~ X:00FF
Large 模式	**xdata**	**64K 字节（理论值）**	**X:0000 ~ X:FFFF**

为了达到比较高的效率，一般建议选择“Small”模式，当编译器出现“error C249: 'DATA': SEGMENT TOO LARGE”错误时，则需要手动将部分比较大的数组通过“xdata”强制分配到 XDATA 区域（例如：char xdata buffer[256];）

2.14.3 设置 -- “Code Rom Size”选择 Large 或者 Huge 模式

在“Code Rom Size”的下拉选项中选择“Large: ...”或者“Huge: ...”模式
80251 的代码大小模式，在 Keil 环境下有如下图所示的 5 种模式：

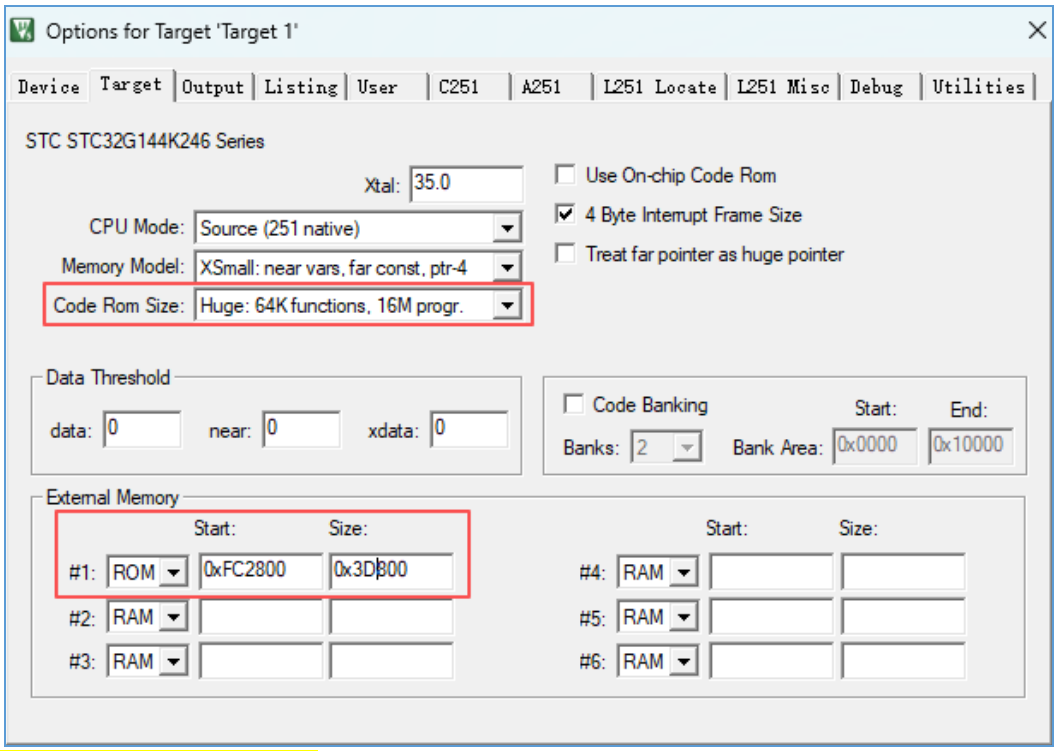


各种模式对比如下表：

Code Rom Size	跳转/调用指令	代码大小限制	
		单个函数/模块/文件的代码大小	总代码大小
Small 模式	AJMP/ACALL	2K	2K
Medium 模式	内部模块代码使用 AJMP/ACALL 外部模块代码使用 LJMP/LCALL	2K	64K
Compact 模式	LCALL/AJMP	2K	64K
Large 模式	LCALL/LJMP	64K	64K
Huge 模式	内部模块代码使用 LJMP/ECALL 外部模块代码使用 EJMP/ECALL （多文件项目中，文件内部的代码为内部模块代码，其他文件的代码为外部模块代码）	64K	16M

2.14.4 设置 -- 超 64K 代码的相关设置

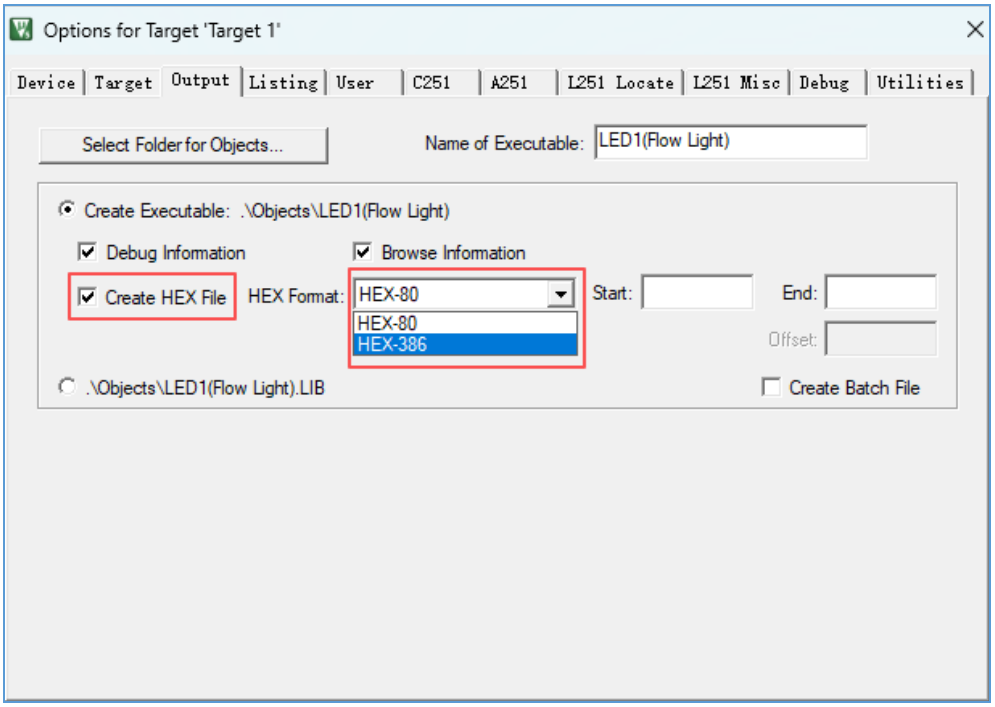
如果代码大小在 64K 以内，选择“Large”模式即可。若代码大小超过 64K，则需要选择“Huge”模式，并需要保证单个函数以及单个文件的代码大小必须在 64K 字节以内，并且单个表格的数据量也必须在 64K 字节以内。同时还需要作如下图所示的设置：



请务必注意：设置的是 ROM 区域

2.14.5 设置 -- HEX 文件格式设置

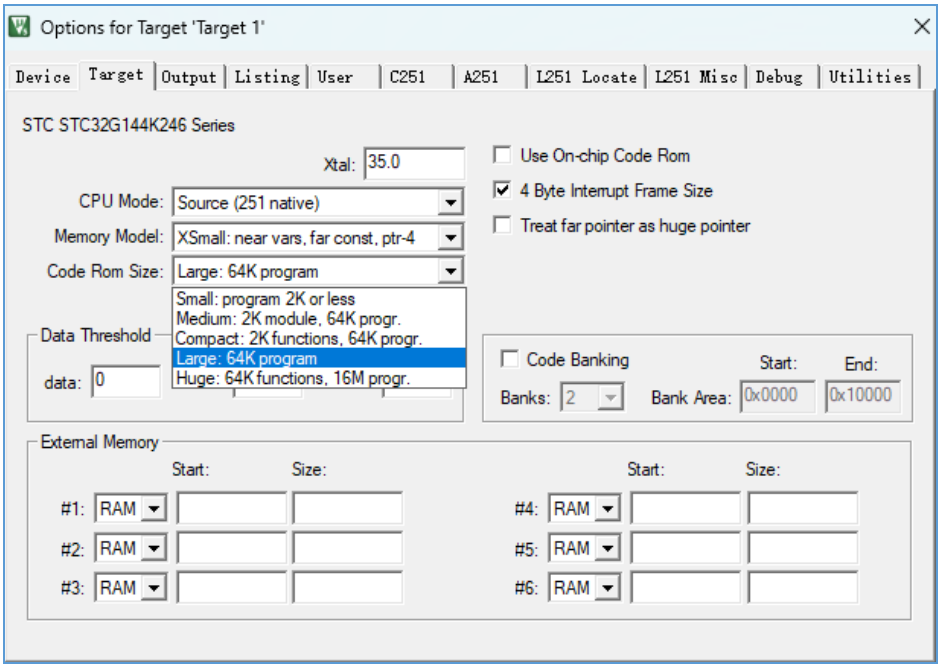
“Options for Target ' Target 1'”浮窗中选择“Output”选项卡，勾选其中的“Create HEX File”选项。若程序空间超过 64K，则“HEX format”必须选择“HEX-386”模式，只有程序空间在 64K 以内，“HEX format”才可选择 “HEX-80”模式；



完成上面的设置后，编译，如果代码没有错误，即可生成 HEX 文件。

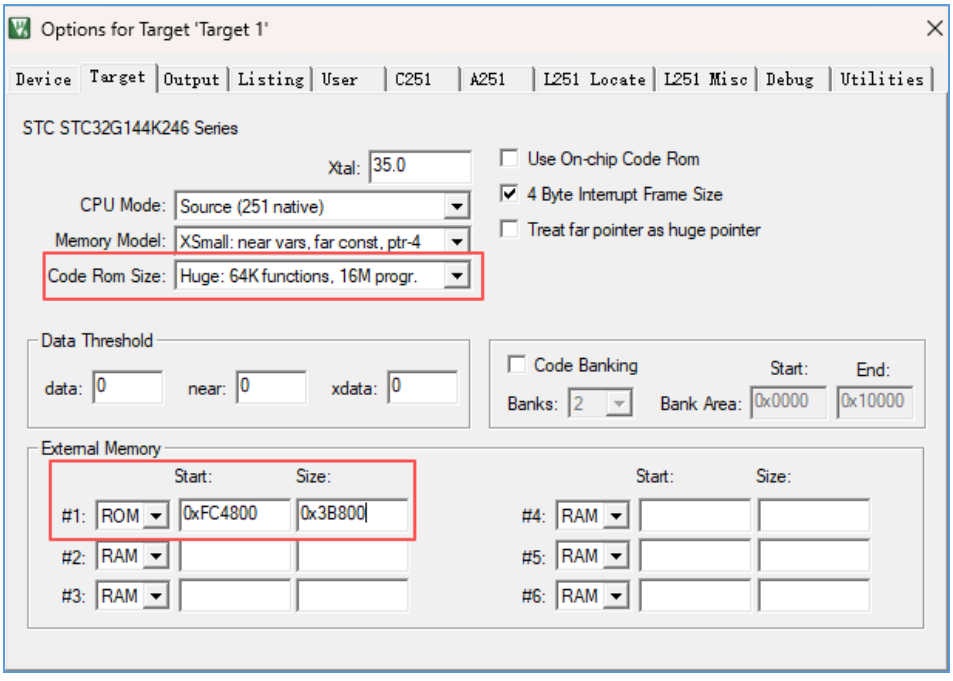
2.15 程序超 64K 时如何设置保留 EEPROM 空间

若用户代码大小在 64K 以内，则可设置“Code Rom Size”为“Large”模式，Keil 编译器在链接代码块时会自动将代码全部链接在 FF:0000~FF:FFFF 的地址范围内。FC:2800~FE:FFFF 的 182K 可根据用户的 EEPROM 大小设置，任意使用。设置如下图：

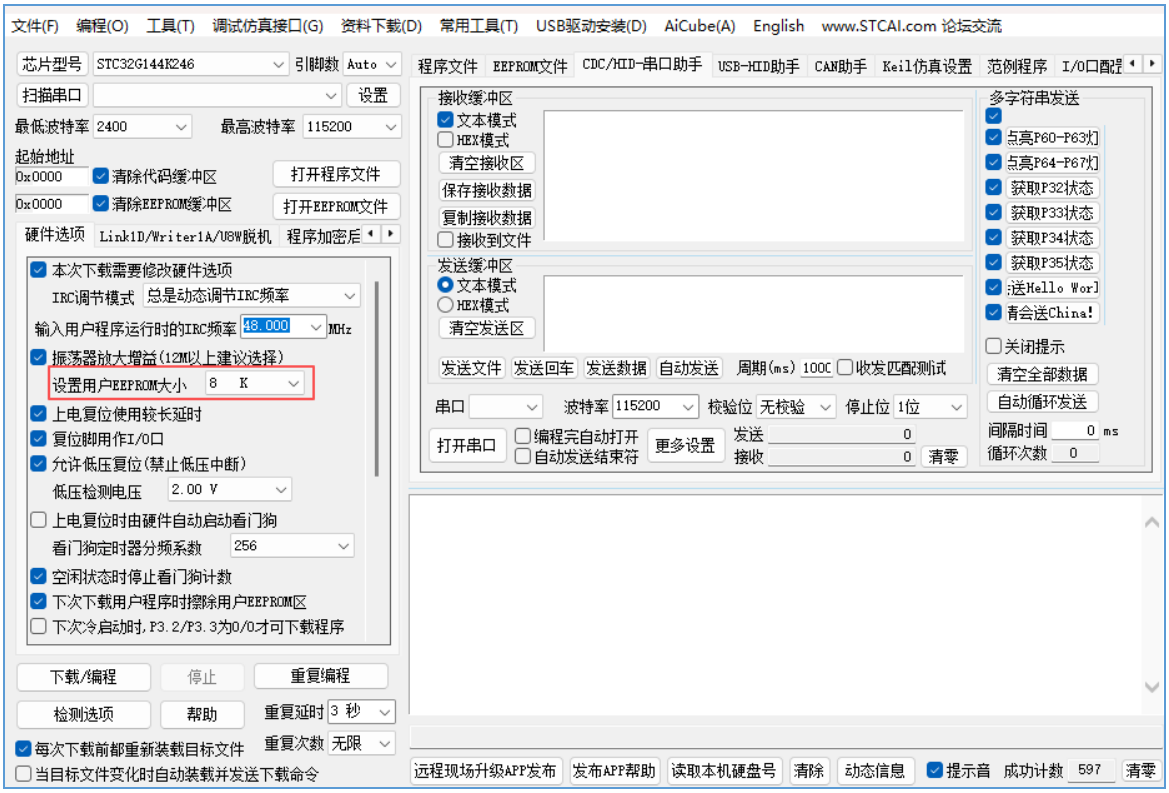


若用户代码大小超过 64K，则“Code Rom Size”必须设置为“Huge”模式，此时 Keil 编译器在链接代码块时，首先会将复位代码和中断向量代码链接在 FF:0000 开始的地址，其他代码块则会自动从用户设置的地址范例开始存放。由于 STC32G144K246 的 EEPROM 在 FLASH 中的地址固定为 FC:2800，所以为了让编译器不要将用户代码放在 EEPROM 区，则必须进行如下的相应设置：

【比如】：用户需要 EEPROM 的大小为 8K，即 FLASH 地址的 FC:2800 ~ FC:47FF 区域为 EEPROM 区域，FLASH 地址的 FC:4800~FF:FFFF 区域为用户代码区。则在 Keil 中需要进行如下设置：



在 AiCube-ISP 下载软件中需要进行如下设置：

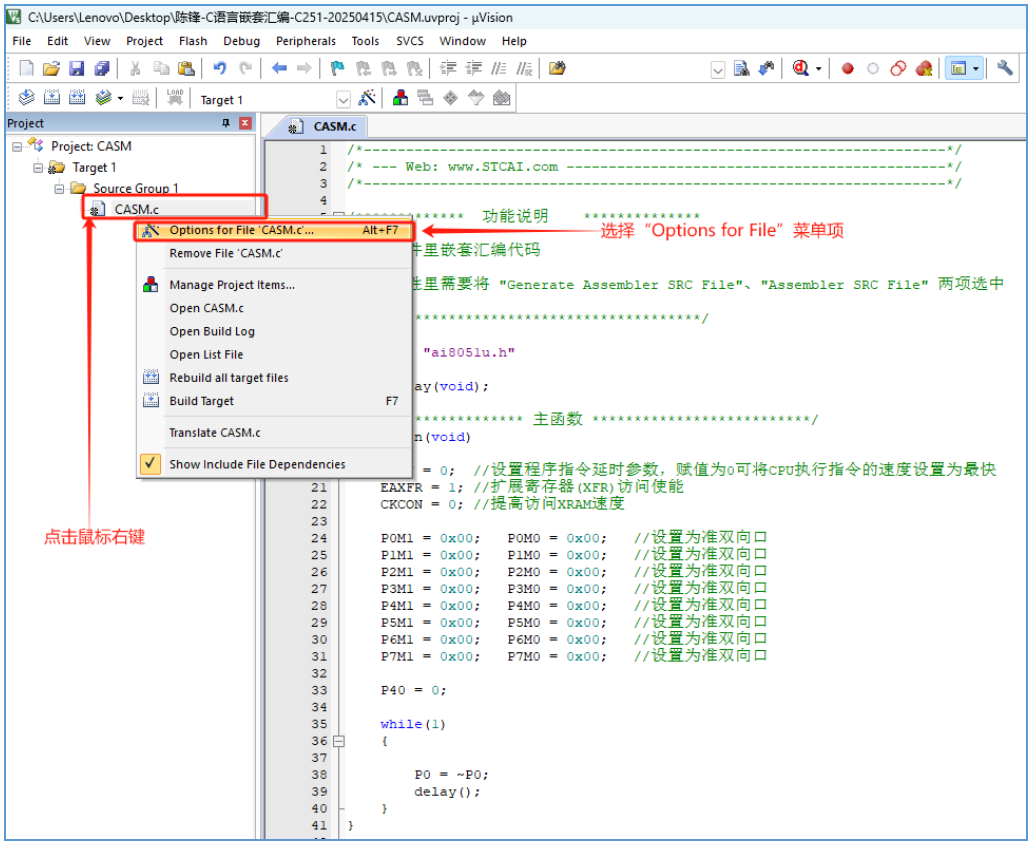


2.16 嵌入汇编代码在 C 语言中，Keil C251 编译器

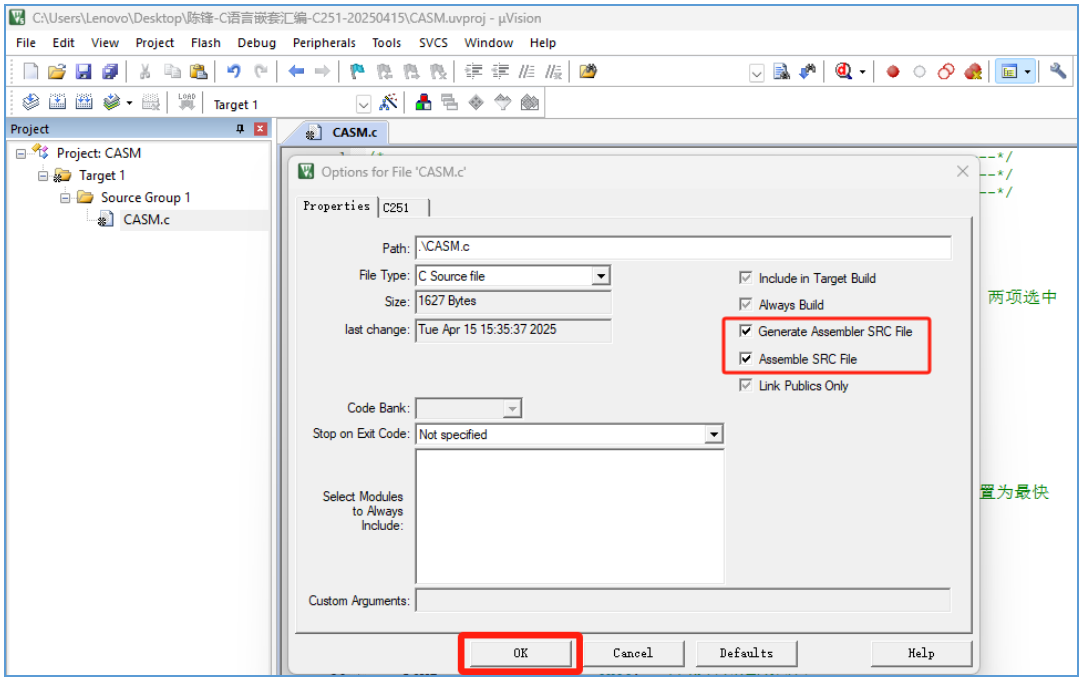
有时为了直接访问硬件资源、优化代码执行速度或精度、实现某些 C 语言本身难以直接表达的算法，需要在 C 语言文件里嵌套汇编代码，以便更直接地利用芯片特性或优化性能。

第 1 步：新建一个工程，添加 C 文件，编译通过

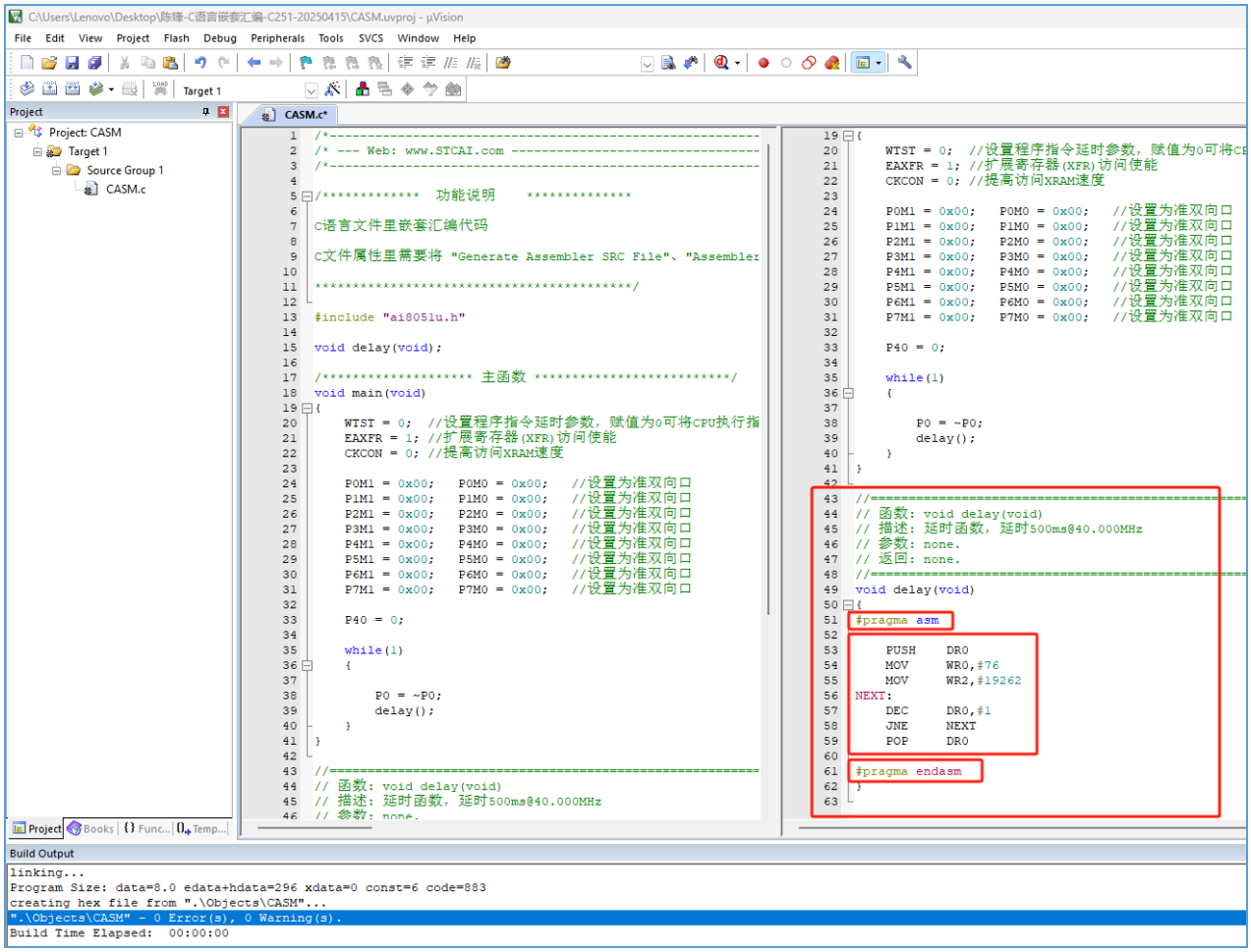
第 2 步：右击 Keil 界面中 Project Workspace 里的 C 文件名，选择‘Option for …’



第 3 步：点击‘Generate Assembler SRC File’、‘Assemble SRC File’两个三态复选框，使它们呈高亮并勾选，点击“OK”



第 4 步：添加汇编代码，以“#pragma asm”为始，以“#pragma endasm”为止。保存、编译，如下图：



连接 Ai8051U 实验箱，打开 AiCube-ISP 系统软件，打开并下载该项目的 HEX 文件，Ai8051U 实验箱上的 LED00~LED07 就闪烁起来了。

2.17 对变量、常量、表格数据、函数指定绝对地址，在 C251 中

2.17.1 Keil C251 中，变量如何指定绝对地址

语法如下：

数据类型 [存储类型] 变量名称 **_at_** 绝对地址;

在 data 区域指定绝对地址变量的范例：

```
int  data  var_data_abs  _at_  0x50;
```

在 edata 区域指定绝对地址变量的范例：

```
int  edata  var_data_abs  _at_  0x200;
```

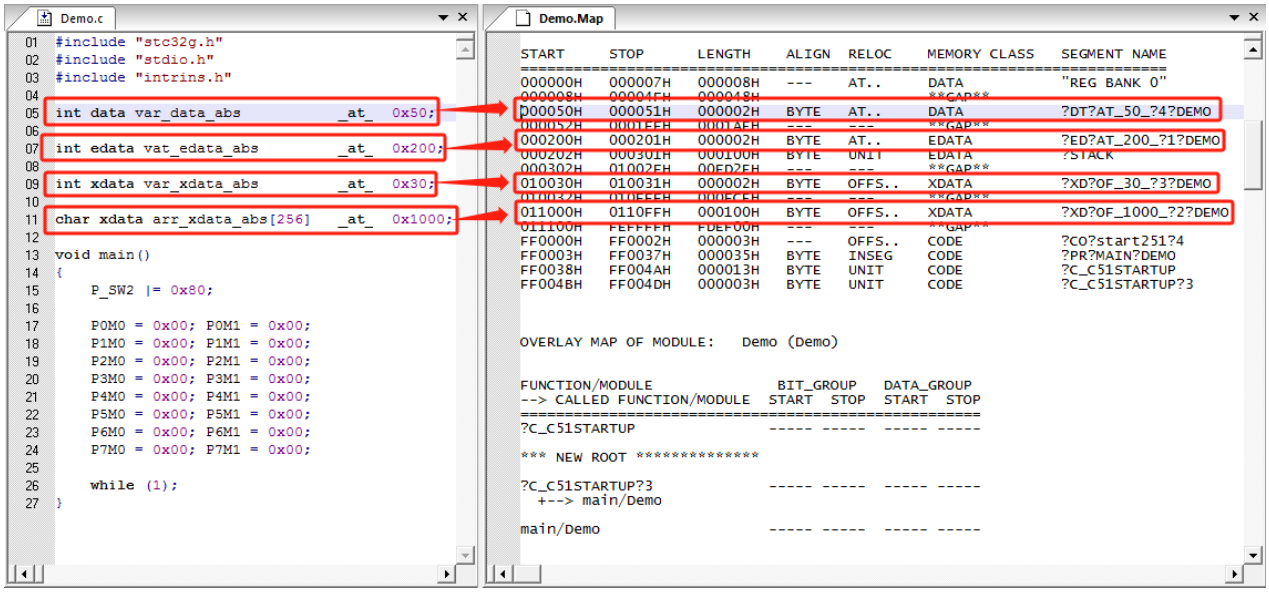
在 xdata 区域指定绝对地址变量的范例：

```
int  xdata  var_xdata_abs  _at_  0x30;
```

在 data 区域指定绝对地址变量的范例：

```
char  xdata  arr_xdata_abs[256]  _at_  0x1000;
```

编译完成后地址分配如下图：



Demo.c

```
#include "stc32g.h"
#include "stdio.h"
#include "intrins.h"

int data var_data_abs          _at_  0x50;

int edata vat_edata_abs        _at_  0x200;

int xdata var_xdata_abs        _at_  0x30;

char xdata arr_xdata_abs[256]  _at_  0x1000;

void main()
{
    P_SW2 |= 0x80;

    P0M0 = 0x00; P0M1 = 0x00;
    P1M0 = 0x00; P1M1 = 0x00;
    P2M0 = 0x00; P2M1 = 0x00;
    P3M0 = 0x00; P3M1 = 0x00;
    P4M0 = 0x00; P4M1 = 0x00;
    P5M0 = 0x00; P5M1 = 0x00;
    P6M0 = 0x00; P6M1 = 0x00;
    P7M0 = 0x00; P7M1 = 0x00;

    while (1);
}
```

Demo.Map

```
START     STOP      LENGTH    ALIGN  RELOC    MEMORY CLASS   SEGMENT NAME
==========================================================================
000000H   000007H   000008H   ---    AT..     DATA           "REG BANK 0"
000008H   00004FH   000048H                   **GAP**
000050H   000051H   000002H   BYTE   AT..     DATA           ?DT?AT_50_?4?DEMO
000052H   0001FFH   0001AEH   ---    ---      **GAP**
000200H   000201H   000002H   BYTE   AT..     EDATA          ?ED?AT_200_?1?DEMO
000202H   000301H   000100H   BYTE   UNIT     EDATA          ?STACK
000302H   01002FH   00FD2EH   ---    ---      **GAP**
010030H   010031H   000002H   BYTE   OFFS..   XDATA          ?XD?OF_30_?3?DEMO
010032H   010FFFH   000FCEH   ---    ---      **GAP**
011000H   0110FFH   000100H   BYTE   OFFS..   XDATA          ?XD?OF_1000_?2?DEMO
011100H   FEFFFFH   FDEF00H   ---    ---      **GAP**
FF0000H   FF0002H   000003H   ---    OFFS..   CODE           ?CO?start251?4
FF0003H   FF0037H   000035H   BYTE   INSEG    CODE           ?PR?MAIN?DEMO
FF0038H   FF004AH   000013H   BYTE   UNIT     CODE           ?C_C51STARTUP
FF004BH   FF004DH   000003H   BYTE   UNIT     CODE           ?C_C51STARTUP?3


OVERLAY MAP OF MODULE:   Demo (Demo)


FUNCTION/MODULE                BIT_GROUP   DATA_GROUP
--> CALLED FUNCTION/MODULE   START  STOP  START  STOP
==========================================================
?C_C51STARTUP                  -----  -----  -----  -----

*** NEW ROOT ***************

?C_C51STARTUP?3                -----  -----  -----  -----
  +--> main/Demo

main/Demo                      -----  -----  -----  -----
```

2.17.2 Keil C251 中，常量如何指定绝对地址

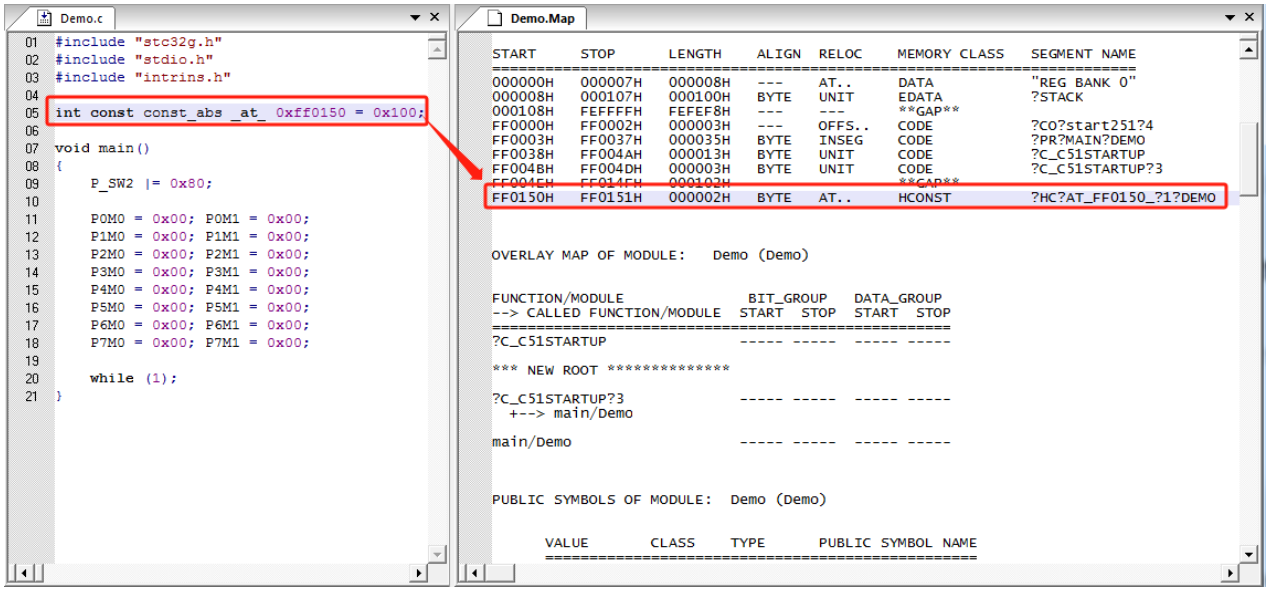
语法如下：

数据类型 **const** 变量名称 **_at_** 绝对地址;

在 code 区域指定绝对地址常量的范例：

```
int const const_abs _at_ 0xff0150 = 0x100;
```

编译完成后地址分配如下图：



Demo.c

```
#include "stc32g.h"
#include "stdio.h"
#include "intrins.h"

int const const_abs _at_ 0xff0150 = 0x100;

void main()
{
    P_SW2 |= 0x80;

    P0M0 = 0x00; P0M1 = 0x00;
    P1M0 = 0x00; P1M1 = 0x00;
    P2M0 = 0x00; P2M1 = 0x00;
    P3M0 = 0x00; P3M1 = 0x00;
    P4M0 = 0x00; P4M1 = 0x00;
    P5M0 = 0x00; P5M1 = 0x00;
    P6M0 = 0x00; P6M1 = 0x00;
    P7M0 = 0x00; P7M1 = 0x00;

    while (1);
}
```

Demo.Map

```
START     STOP      LENGTH    ALIGN  RELOC    MEMORY CLASS   SEGMENT NAME
=============================================================================
000000H   000007H   000008H   ---    AT..     DATA           "REG BANK 0"
000008H   000107H   000100H   BYTE   UNIT     EDATA          ?STACK
000108H   FEFFFFH   FEFEF8H   ---    ---      **GAP**
FF0000H   FF0002H   000003H   ---    OFFS..   CODE           ?CO?start251?4
FF0003H   FF0037H   000035H   BYTE   INSEG    CODE           ?PR?MAIN?DEMO
FF0038H   FF004AH   000013H   BYTE   UNIT     CODE           ?C_C51STARTUP
FF004BH   FF004DH   000003H   BYTE   UNIT     CODE           ?C_C51STARTUP?3
FF004EH   FF014FH   000102H                   **GAP**
FF0150H   FF0151H   000002H   BYTE   AT..     HCONST         ?HC?AT_FF0150_?1?DEMO


OVERLAY MAP OF MODULE:   Demo (Demo)


FUNCTION/MODULE              BIT_GROUP   DATA_GROUP
--> CALLED FUNCTION/MODULE   START  STOP  START  STOP
=================================================
?C_C51STARTUP                ----- -----  ----- -----

*** NEW ROOT **************

?C_C51STARTUP?3              ----- -----  ----- -----
  +--> main/Demo

main/Demo                    ----- -----  ----- -----


PUBLIC SYMBOLS OF MODULE:  Demo (Demo)

     VALUE       CLASS    TYPE      PUBLIC SYMBOL NAME
     ================================================
```

2.17.3 Keil C251 中，表格数据如何指定绝对地址

语法如下：

数据类型 **code** 变量名称 **_at_** 绝对地址 = { 表格数据 };

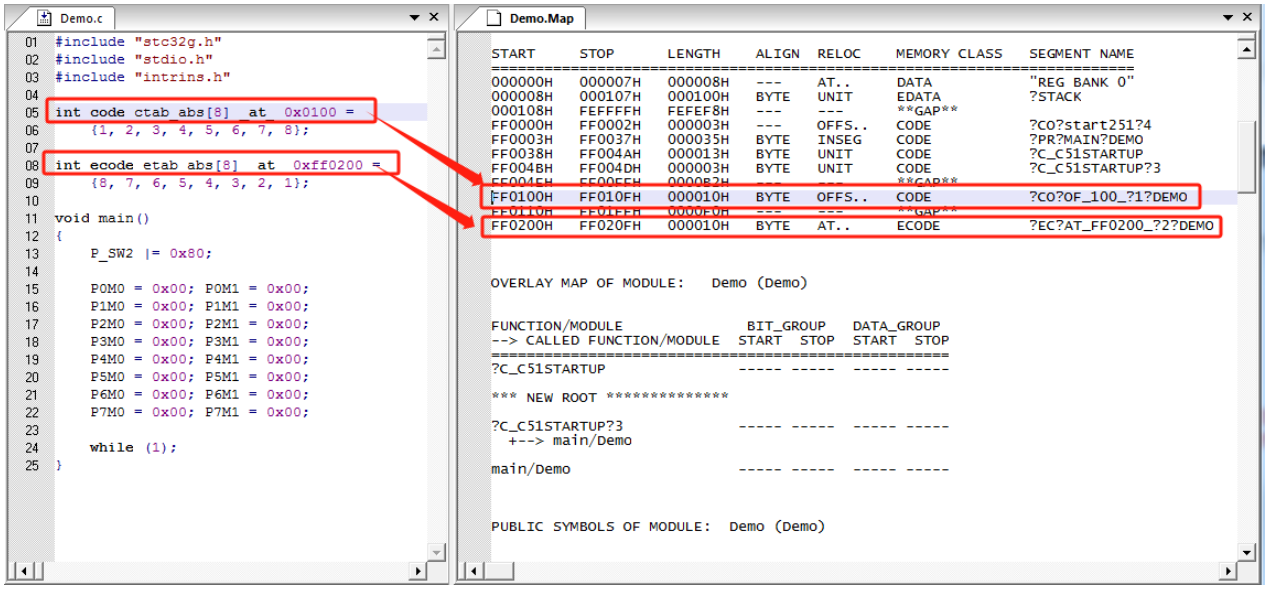
或者

数据类型 **ecode** 变量名称 **_at_** 绝对地址 = { 表格数据 };

注：使用 code 储存类型时，地址范围为 FF:0000H~FF:FFFFH，需要用 0000H~FFFFH 的 16 位地址指定的绝对地址；使用 ecode 储存类型时，地址范围为 80:0000H~FF:FFFFH，需要用 800000H~FFFFFFH 的 24 位地址指定的绝对地址

在 code 区域指定绝对地址表格的范例：

```
int   code   ctab_abs[8]   _at_   0x0100   =   {1, 2, 3, 4, 5, 6, 7, 8};
int   ecode   etab_abs[8]   _at_   0xff0200   =   {8, 7, 6, 5, 4, 3, 2, 1};
```



Demo.c

```
#include "stc32g.h"
#include "stdio.h"
#include "intrins.h"

int code ctab_abs[8]  at  0x0100 =
    {1, 2, 3, 4, 5, 6, 7, 8};

int ecode etab_abs[8]  at  0xff0200 =
    {8, 7, 6, 5, 4, 3, 2, 1};

void main()
{
    P_SW2 |= 0x80;

    P0M0 = 0x00; P0M1 = 0x00;
    P1M0 = 0x00; P1M1 = 0x00;
    P2M0 = 0x00; P2M1 = 0x00;
    P3M0 = 0x00; P3M1 = 0x00;
    P4M0 = 0x00; P4M1 = 0x00;
    P5M0 = 0x00; P5M1 = 0x00;
    P6M0 = 0x00; P6M1 = 0x00;
    P7M0 = 0x00; P7M1 = 0x00;

    while (1);
}
```

Demo.Map

```
START     STOP      LENGTH    ALIGN  RELOC    MEMORY CLASS   SEGMENT NAME
=============================================================================
000000H   000007H   000008H   ---    AT..     DATA           "REG BANK 0"
000008H   000107H   000100H   BYTE   UNIT     EDATA          ?STACK
000108H   FEFFFFH   FEFEF8H   ---    ---      **GAP**
FF0000H   FF0002H   000003H   ---    OFFS..   CODE           ?CO?start251?4
FF0003H   FF0037H   000035H   BYTE   INSEG    CODE           ?PR?MAIN?DEMO
FF0038H   FF004AH   000013H   BYTE   UNIT     CODE           ?C_C51STARTUP
FF004BH   FF004DH   000003H   BYTE   UNIT     CODE           ?C_C51STARTUP?3
FF004EH   FF00FFH   0000B2H   ---    ---      **GAP**
FF0100H   FF010FH   000010H   BYTE   OFFS..   CODE           ?CO?OF_100_?1?DEMO
FF0110H   FF01FFH   0000F0H   ---    ---      **GAP**
FF0200H   FF020FH   000010H   BYTE   AT..     ECODE          ?EC?AT_FF0200_?2?DEMO


OVERLAY MAP OF MODULE:   Demo (Demo)


FUNCTION/MODULE                BIT_GROUP   DATA_GROUP
--> CALLED FUNCTION/MODULE   START  STOP  START  STOP
=============================================================
?C_C51STARTUP                  ----- -----  ----- -----

*** NEW ROOT **************

?C_C51STARTUP?3                ----- -----  ----- -----
  +--> main/Demo

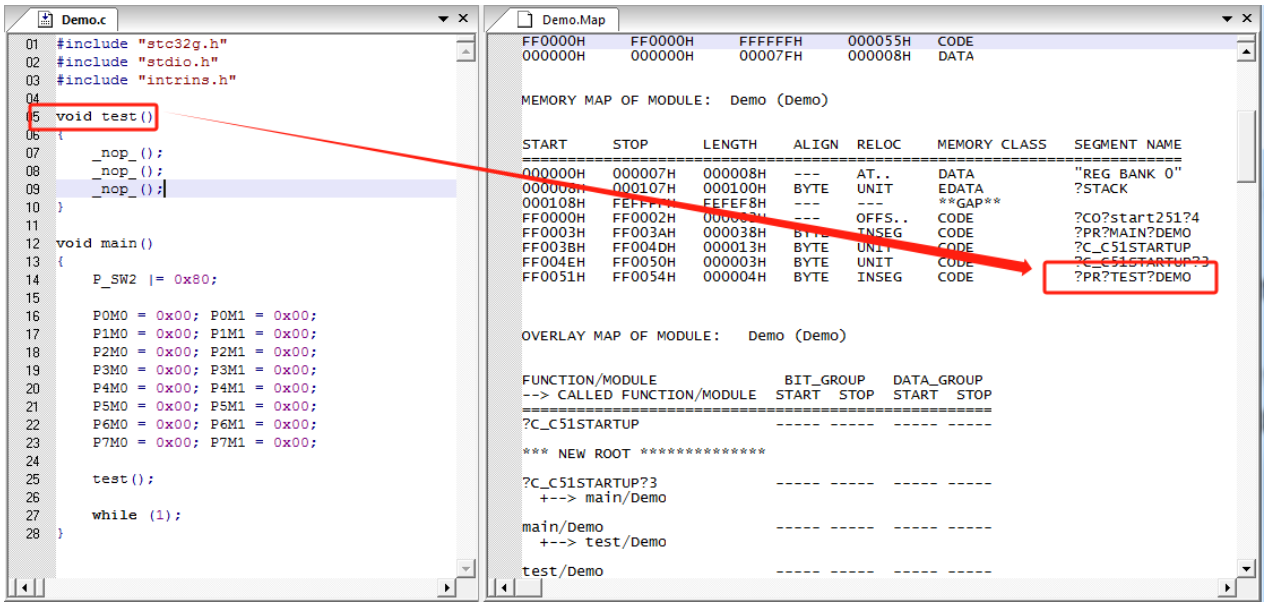
main/Demo                      ----- -----  ----- -----


PUBLIC SYMBOLS OF MODULE:  Demo (Demo)
```

2.17.4 Keil C251 中，函数如何指定绝对地址

C251 中无法直接在程序中指定函数的绝对地址，需要通过如下方法实现

首先在程序编写完成函数代码，编译成功后，获取函数的链接符号（如下图为“?PR?TEST?DEMO”）

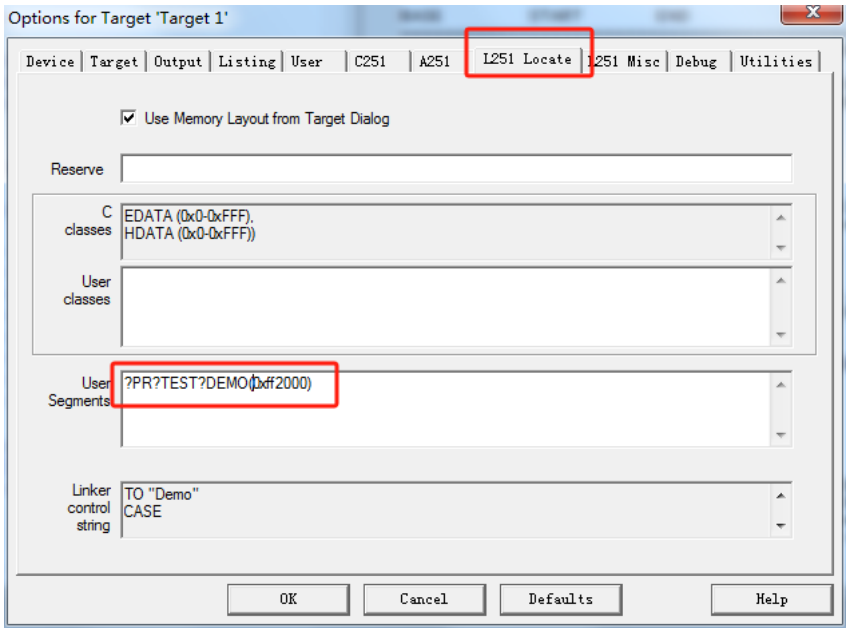


接下来在项目设置项中打开“L251 Locate”设置页面

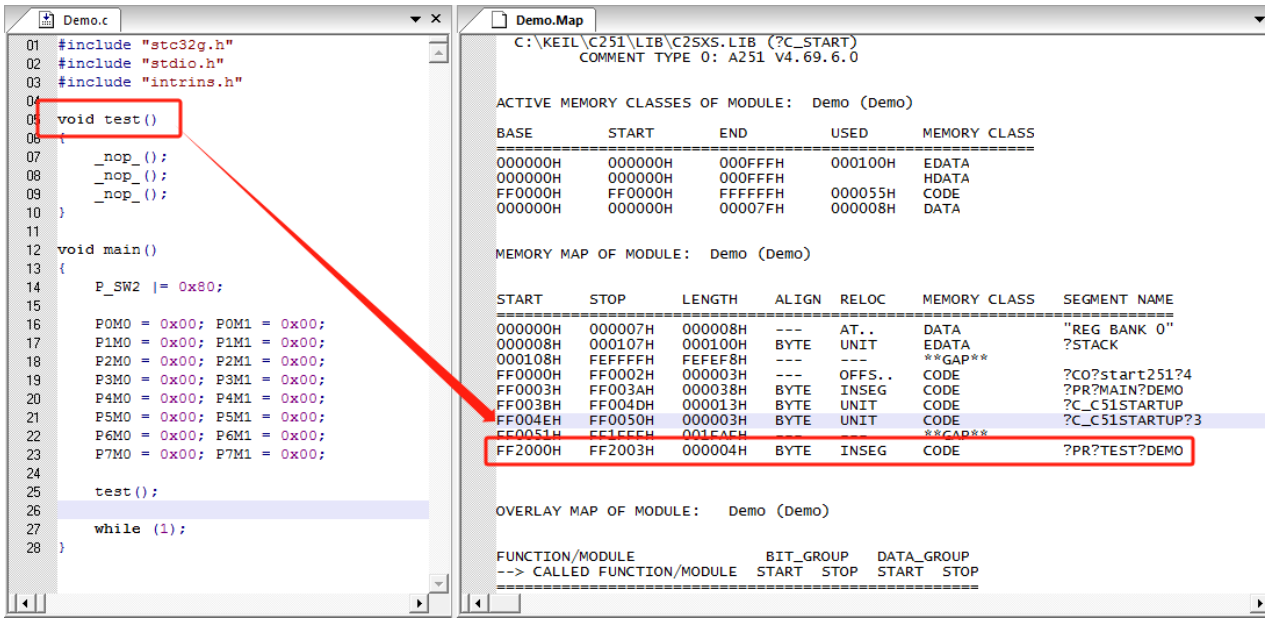
在“User Segments”一栏中按照：链接名称 (链接地址) 的格式，输入绝对地址

注：链接地址需要使用 24 位地址指定

如下图，将函数指定到代码区的 0xff2000 的绝对地址



设置完成后，再次编译，函数即可被链接到指定的绝对地址，如下图：



Demo.c

```
01  #include "stc32g.h"
02  #include "stdio.h"
03  #include "intrins.h"
04
05  void test()
06  {
07      _nop_();
08      _nop_();
09      _nop_();
10  }
11
12  void main()
13  {
14      P_SW2 |= 0x80;
15
16      P0M0 = 0x00; P0M1 = 0x00;
17      P1M0 = 0x00; P1M1 = 0x00;
18      P2M0 = 0x00; P2M1 = 0x00;
19      P3M0 = 0x00; P3M1 = 0x00;
20      P4M0 = 0x00; P4M1 = 0x00;
21      P5M0 = 0x00; P5M1 = 0x00;
22      P6M0 = 0x00; P6M1 = 0x00;
23      P7M0 = 0x00; P7M1 = 0x00;
24
25      test();
26
27      while (1);
28  }
```

Demo.Map

```
  C:\KEIL\C251\LIB\C2SXS.LIB (?C_START)
        COMMENT TYPE 0: A251 V4.69.6.0


ACTIVE MEMORY CLASSES OF MODULE:  Demo (Demo)

BASE         START        END          USED      MEMORY CLASS
=============================================================
000000H      000000H      000FFFH      000100H   EDATA
000000H      000000H      000FFFH                HDATA
FF0000H      FF0000H      FFFFFFH      000055H   CODE
000000H      000000H      00007FH      000008H   DATA


MEMORY MAP OF MODULE:  Demo (Demo)

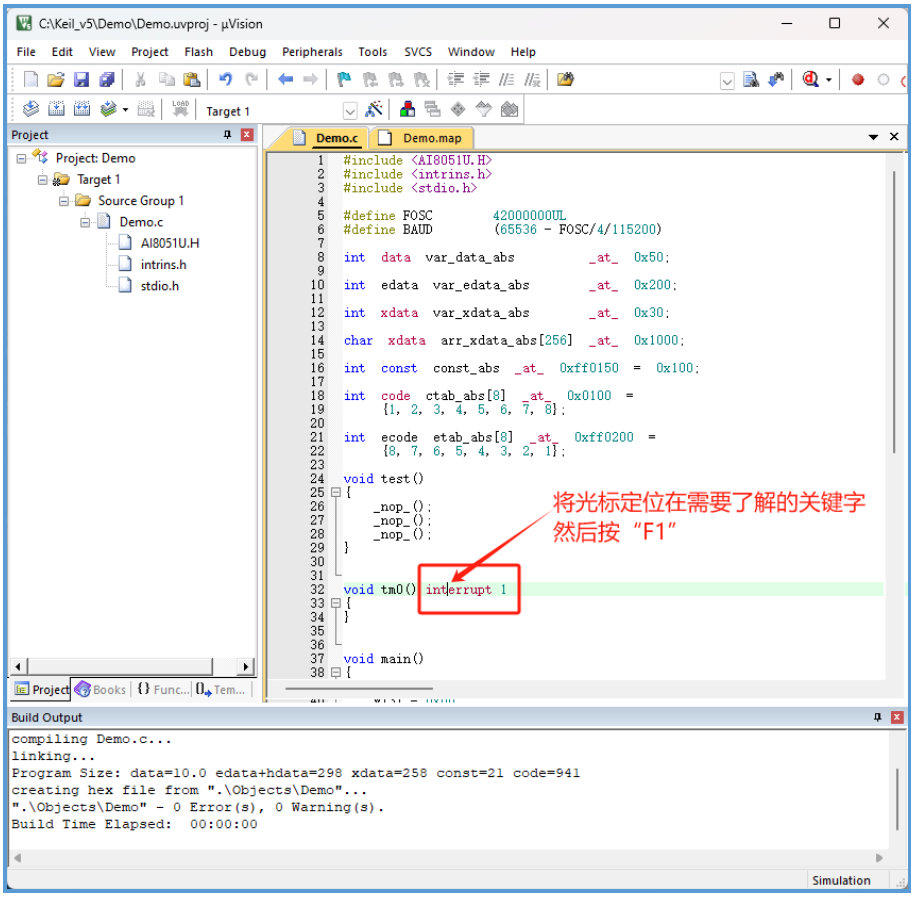

START     STOP      LENGTH    ALIGN  RELOC    MEMORY CLASS   SEGMENT NAME
=============================================================================
000000H   000007H   000008H   ---    AT..     DATA           "REG BANK 0"
000008H   000107H   000100H   BYTE   UNIT     EDATA          ?STACK
000108H   FEFFFFH   FEFEF8H   ---    ---      **GAP**
FF0000H   FF0002H   000003H   ---    OFFS..   CODE           ?CO?start251?4
FF0003H   FF003AH   000038H   BYTE   INSEG    CODE           ?PR?MAIN?DEMO
FF003BH   FF004DH   000013H   BYTE   UNIT     CODE           ?C_C51STARTUP
FF004EH   FF0050H   000003H   BYTE   UNIT     CODE           ?C_C51STARTUP?3
FF0051H   FF1FFFH   001FAFH   ---    ---      **GAP**
FF2000H   FF2003H   000004H   BYTE   INSEG    CODE           ?PR?TEST?DEMO

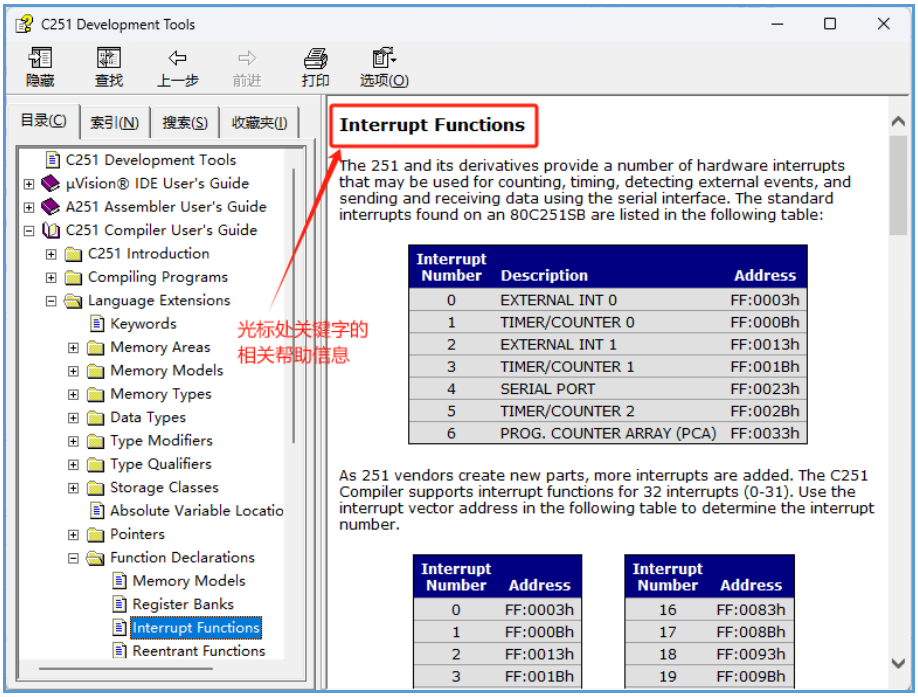

OVERLAY MAP OF MODULE:   Demo (Demo)


FUNCTION/MODULE              BIT_GROUP   DATA_GROUP
--> CALLED FUNCTION/MODULE   START  STOP  START  STOP
=================================================
```

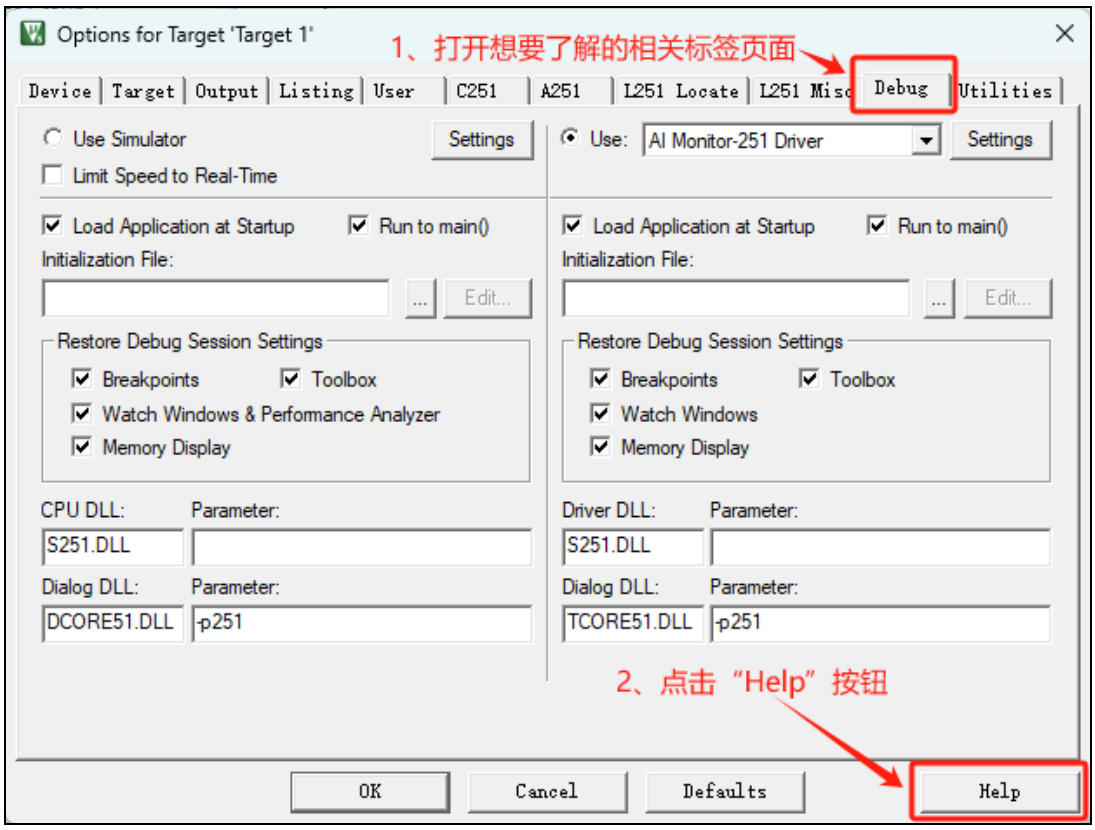
2.18 获取帮助的简单方法，Keil 软件中

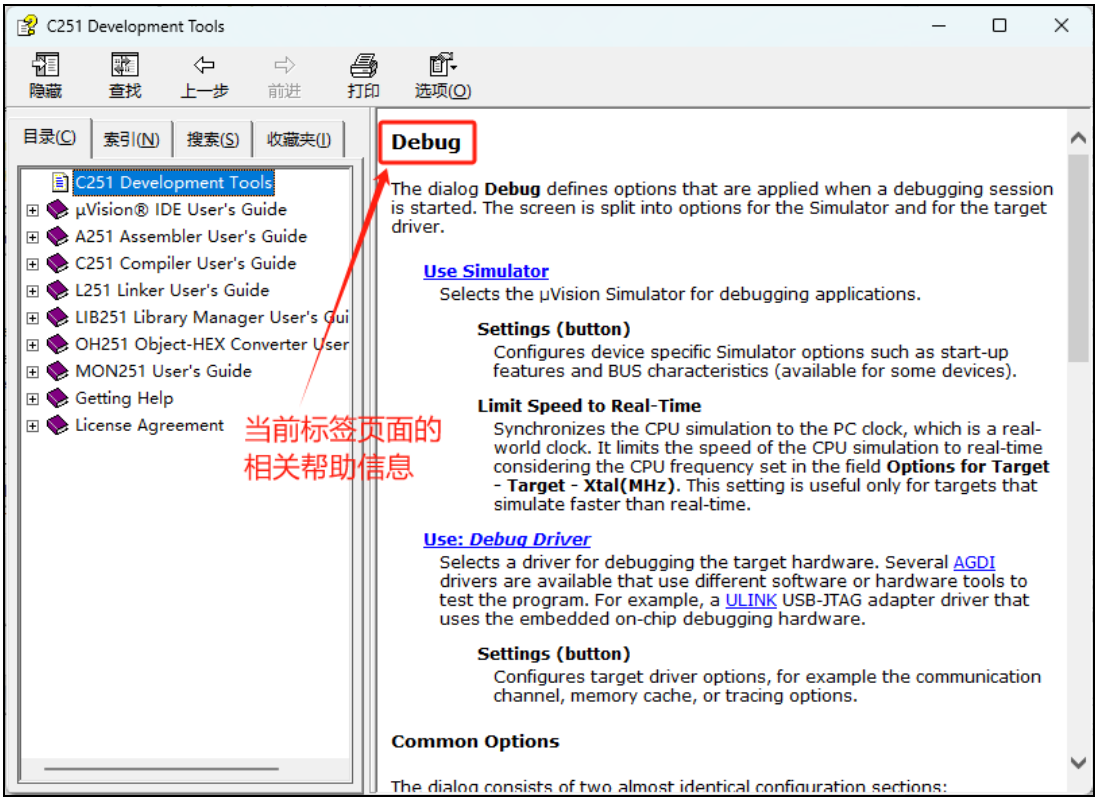
Keil 软件提供了很完整的帮助文件，对于一般的软件使用和编程问题，直接使用 Keil 软件的帮助基本都可以得到解决。如下图：



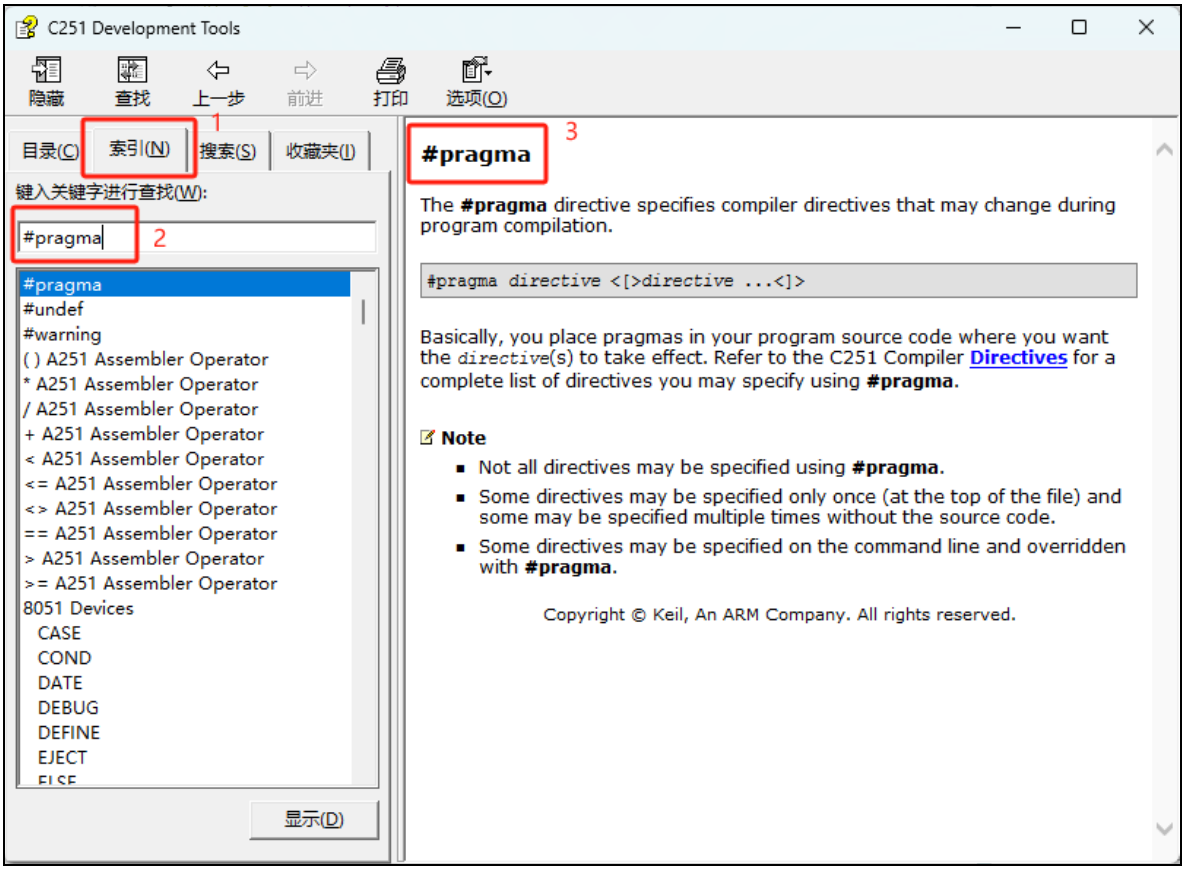


若需要了解项目设置中的相关设置的，可按下图所示的方法获取帮助





另外，也可在帮助窗口中直接输入想了解的内如。比如需要了解如何在程序中设置特殊的编译指示，可按下图所示，在搜索框输入“#pragma”即可

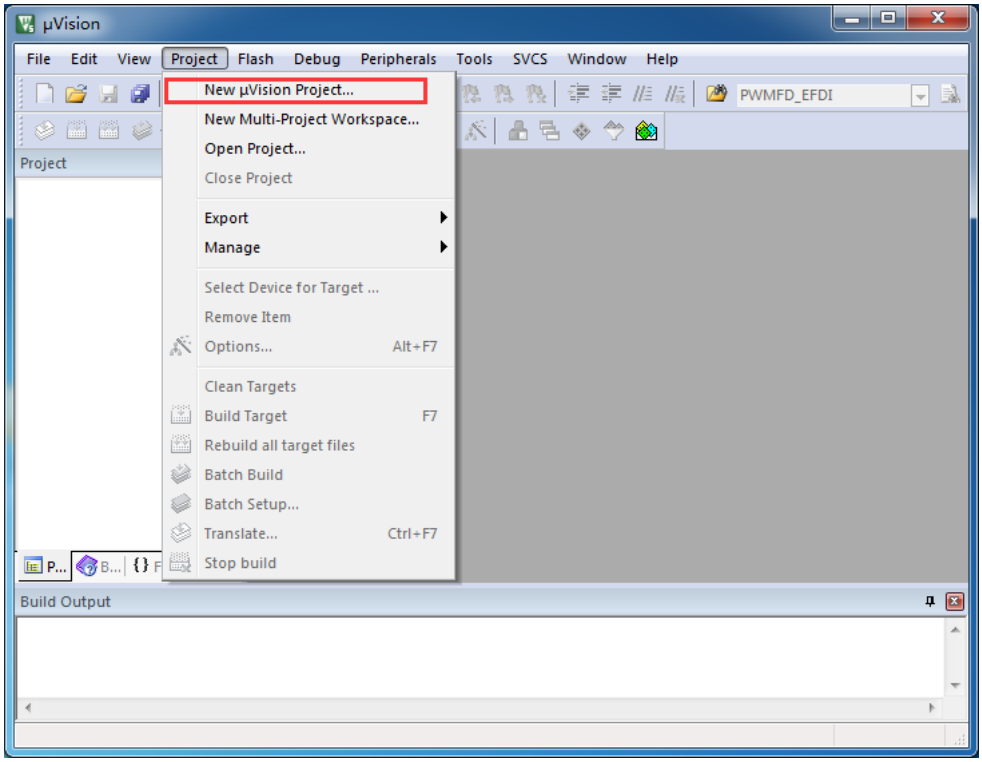


如果需要更详细的帮助详细，可登录 Keil 官网进行查询

2.19 建立多文件项目的方法，在 Keil 软件中

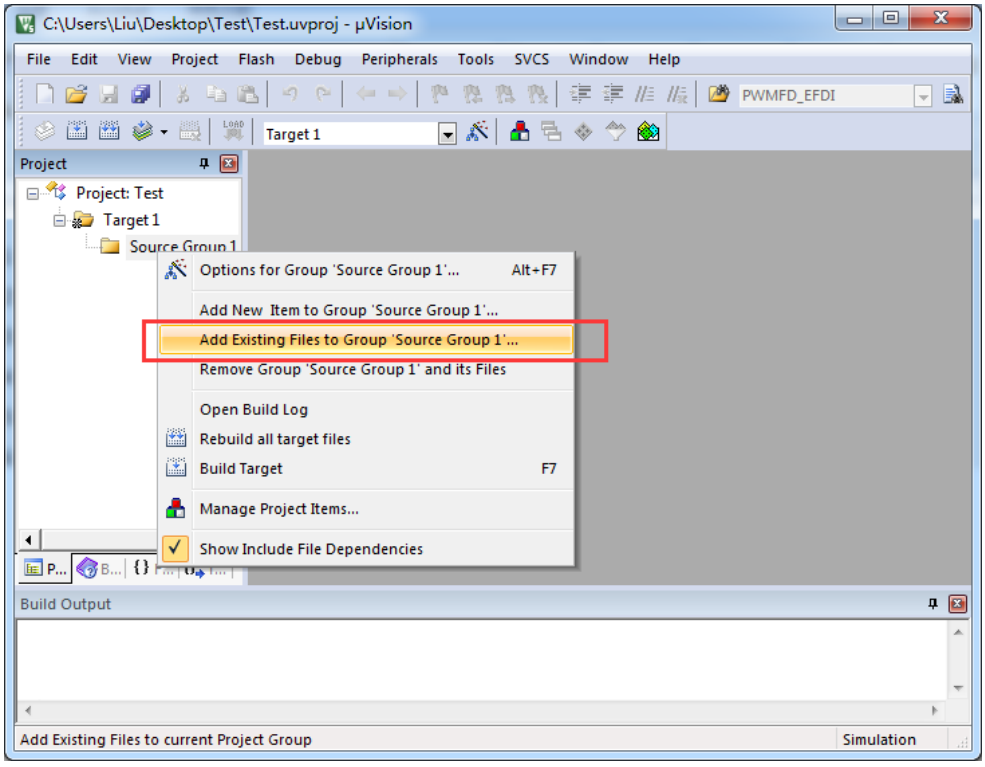
在 Keil 中，一般比较小的项目都只有一个源文件，但对于一些稍微复杂的项目往往需要多个源文件
建立多文件项目的方法如下：

1、首先打开 Keil，在菜单“Project”中选择“New uVision Project ...”

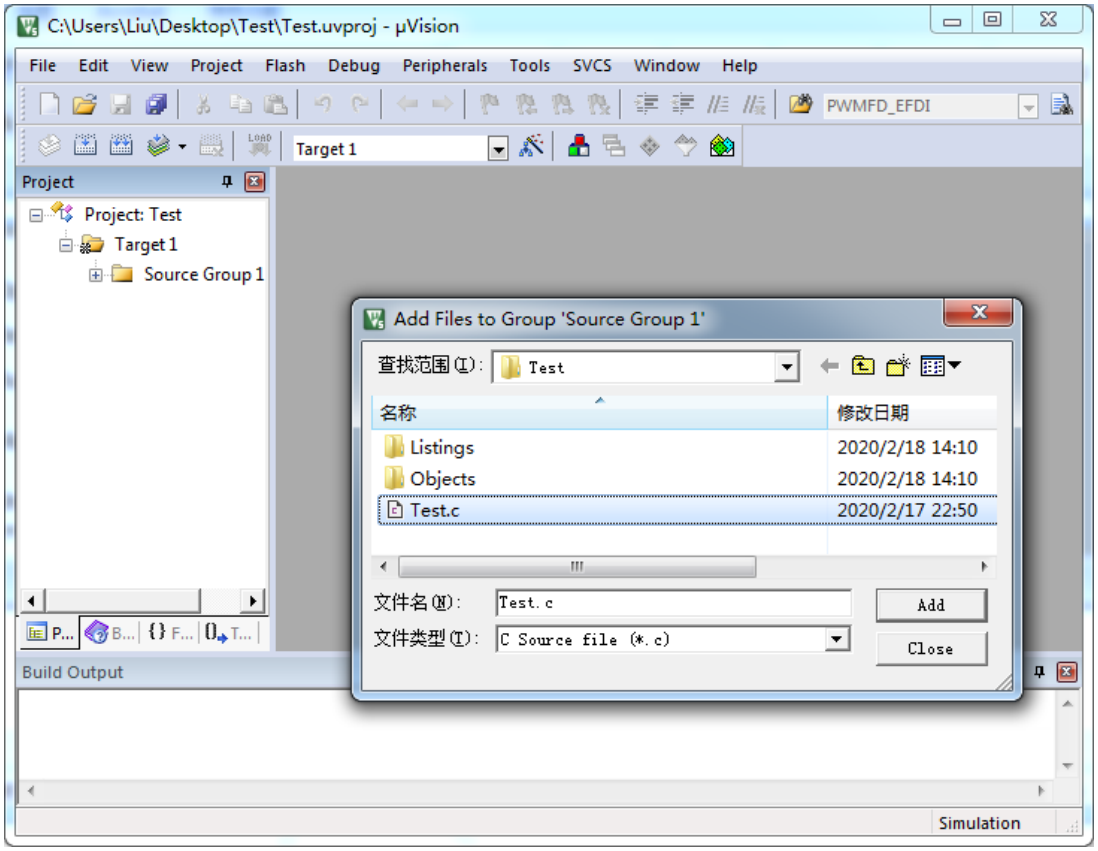


即可完成一个空项目的建立

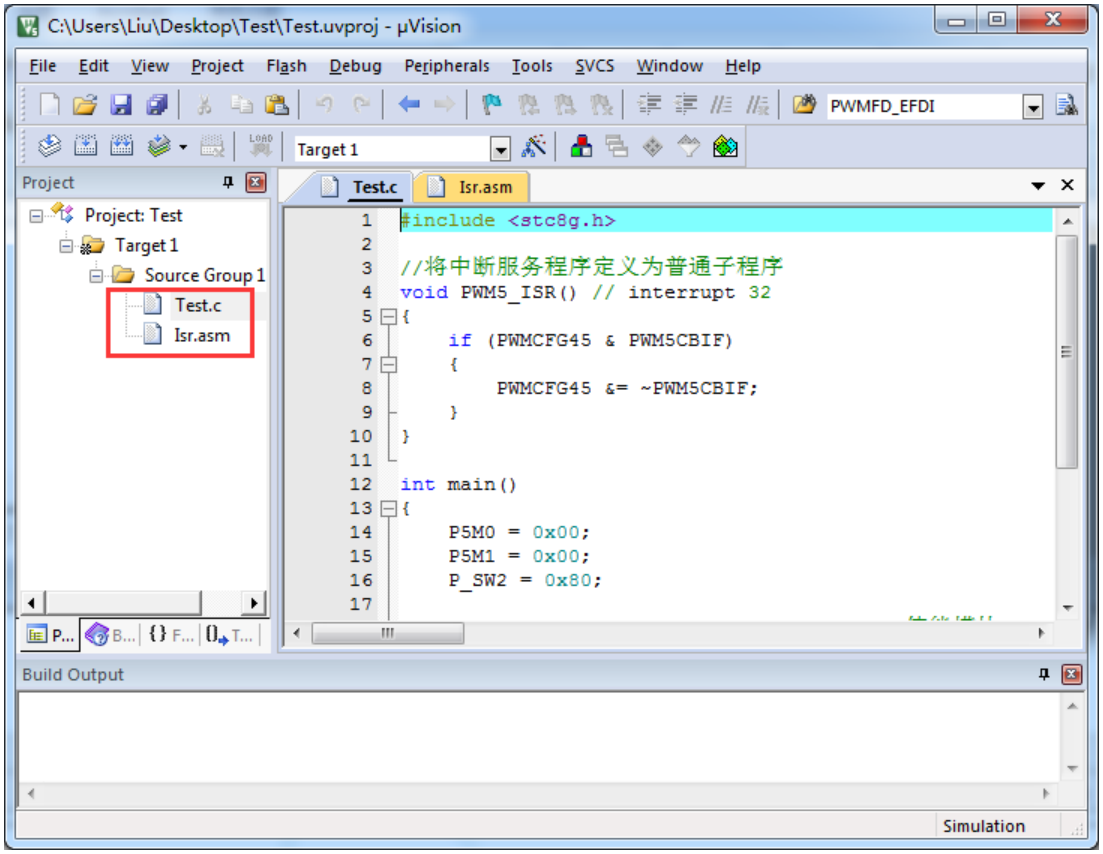
2、在空项目的项目树中，鼠标右键单击“Source Group 1”，并选择右键菜单中的“Add Existing Files to Group "Source Group 1" ...”



3、在弹出的文件对话框中，多次添加源文件



如下图所示即可完成多文件项目的建立



2.20 外部变量和外部函数的声明和使用，在多文件项目中

在 Keil 环境下开发比较大的项目时，往往需要在项目中加入多个源文件。而不同源文件内，可能会有一些变量和函数是开放为其它源文件使用的。但默认情况下，在源文件内部定义的变量和函数只在本源文件可见(可使用)，如果需要跨源文件使用则需要遵循一定的规则进行定义和声明。

对于**需要跨源文件访问变量**：

1、变量必须定义为全局变量
所谓全局变量就是在函数外部定义，在函数内部定义的变量为局部变量
由于局部变量的作用域仅在定义的函数内部，且生命周期在函数返回时就结束了
所以**要使变量能跨源文件访问，变量必须定义为全局变量**

2、变量必须定义为非静态变量
使用 static 关键字定义的变量为静态变量，静态变量作用域仅为本源文件，外部源文件不可访问。
所以要使变量能跨源文件访问，变量必须定义为非静态变量，即定义时不要使用 static 关键字

```
13
14  //定义全局变量
15  /*************************************************
16  非静态的全局变量可以使用extern声明后跨源文件使用
17  不同的源文件不可以定义相同名称的非静态全局变量
18  *************************************************/
19  int g_iVar1;
20
```

3、不同的源文件中不可定义名称相同的全局变量
如果在不同源文件中定义名称相同的全局变量，编译器会如下错误提示

```
linking...
*** ERROR L104: MULTIPLE PUBLIC DEFINITIONS
```

4、在需要使用外部变量的源文件内使用 extern 声明变量

```
06
07  //使用extern声明全局变量,使变量可跨文件访问
08  extern int g_iVar1;
09
```

如果需要在多个源文件中使用同一个外部变量，可将上面的外部声明语句
放在.h 文件中，在需要使用这个变量的源文件中 include 一下这个头文件即可

同样的，若需要**跨源文件调用外部函数**：

1、函数必须定义为非静态函数
使用 static 关键字定义的函数为静态函数，静态函数作用域仅为本源文件，外部源文件不可访问。
所以要使函数能跨源文件访问，必须定义为非静态函数，即定义函数时不要使用 static 关键字。

```
32
33  //定义非静态函数
34  /*************************************************
35  非静态函数可以使用extern声明后跨源文件使用
36  不同的源文件不可以定义相同名称的非静态函数
37  *************************************************/
38  int func1(int var)
39  {
40      check();
41      return (var * factor);
42  }
43
```

2、不同的源文件中不可定义名称相同的非静态函数

如果在不同源文件中定义名称相同的非静态函数时，编译器会如下错误提示

```
linking...
*** ERROR L104: MULTIPLE PUBLIC DEFINITIONS
```

3、在需要调用外部函数的源文件内直接声明一下即可

对于外部函数声明，是否使用 extern 关键字，编译时都可通过

```
03
04  //声明外部函数,使函数可跨文件调用
05  int func1(int var);
06
```

也可以 extern int func1(int var);

如果需要在多个源文件中使用同一个外部函数，可将上面的外部声明语句

放在.h 文件中，在需要调用这个函数的源文件中 include 一下这个头文件即可

2.21 变量的声明要求在可执行语句之前，C89 标准，KEIL C 语言

KEIL 支持的 C89 标准 C 语言，变量的声明必须遵循 C89 的严格作用域规则。

全局变量的声明需要在函数之前声明。

局部变量的声明必须集中在 {} 代码块的起始位置，不能与可执行语句混合，必须在代码块开头（任何可执行语句之前）声明。

循环变量：for 循环的计数器必须在循环外声明（C99 才支持 for(int i=0;...) 语法）。

中间声明：禁止在代码中间声明变量（如 if 语句之间或函数调用之后）。

2.22 STC32G144K246 仿真 -- SWD 硬件仿真

在编写较大的程序时，往往会遇到代码编译没问题，但运行异常的情况，或者要观察项目运行状况，这时就可以借助仿真来查找问题，进行程序调试，或者对项目实时监控，进行逻辑验证。

STC32G144K246 支持仿真功能。STC32G144K246 芯片出厂时默认仿真功能是关闭的，需要使用 AiCube-ISP 软件将目标单片机设置为仿真芯片，才能开启芯片的仿真功能。

STC32G144K246 芯片可支持 USB 仿真和 SWD 硬件仿真。

SWD 硬件仿真，需要使用官方的 USB Link1D 工具作为仿真器才能进行仿真。

下面我们就以“中断实验 1(外部中断 INT0/P3.2，INT1/P3.3 控制点灯并仿真)”实验为例，使用 **STC32G144K246 实验箱仿真 -- SWD 硬件仿真**，并观察实验现象。

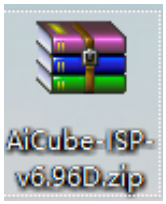
2.22.1 安装 AiCube-ISP 软件，制作仿真芯片

2.22.1.1 安装 AiCube-ISP 下载/编程/烧录 工具软件

安装 STCAI 公司研发的 AiCube-ISP 下载 / 编程 / 烧录 / 软件，含强大的辅助开发工具包。登录 STCAI 官网，下载最新版的 AiCube-ISP 安装包，下载链接：https://www.stcai.com/gjrj



下载下来的压缩包文件图标：

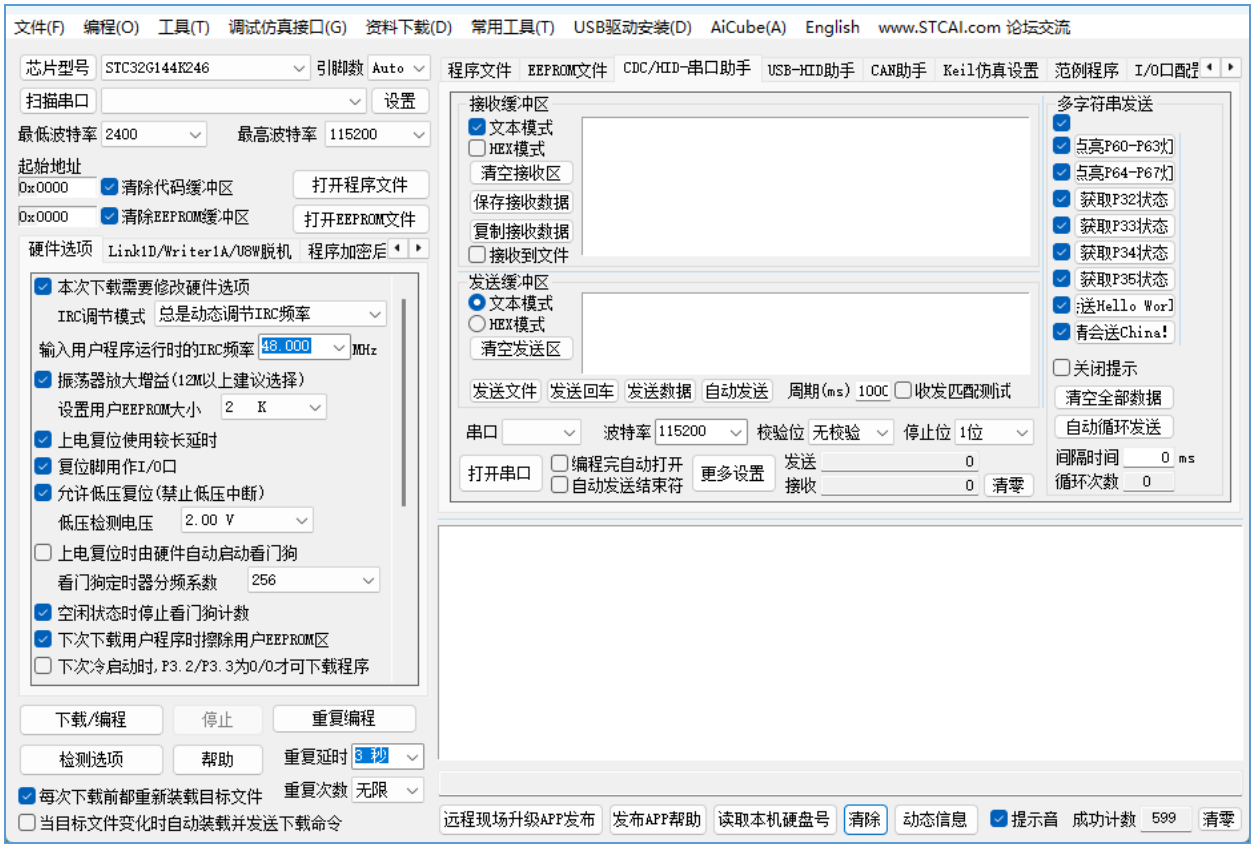


双击解压缩文件，AiCube-ISP 系统软件图标：

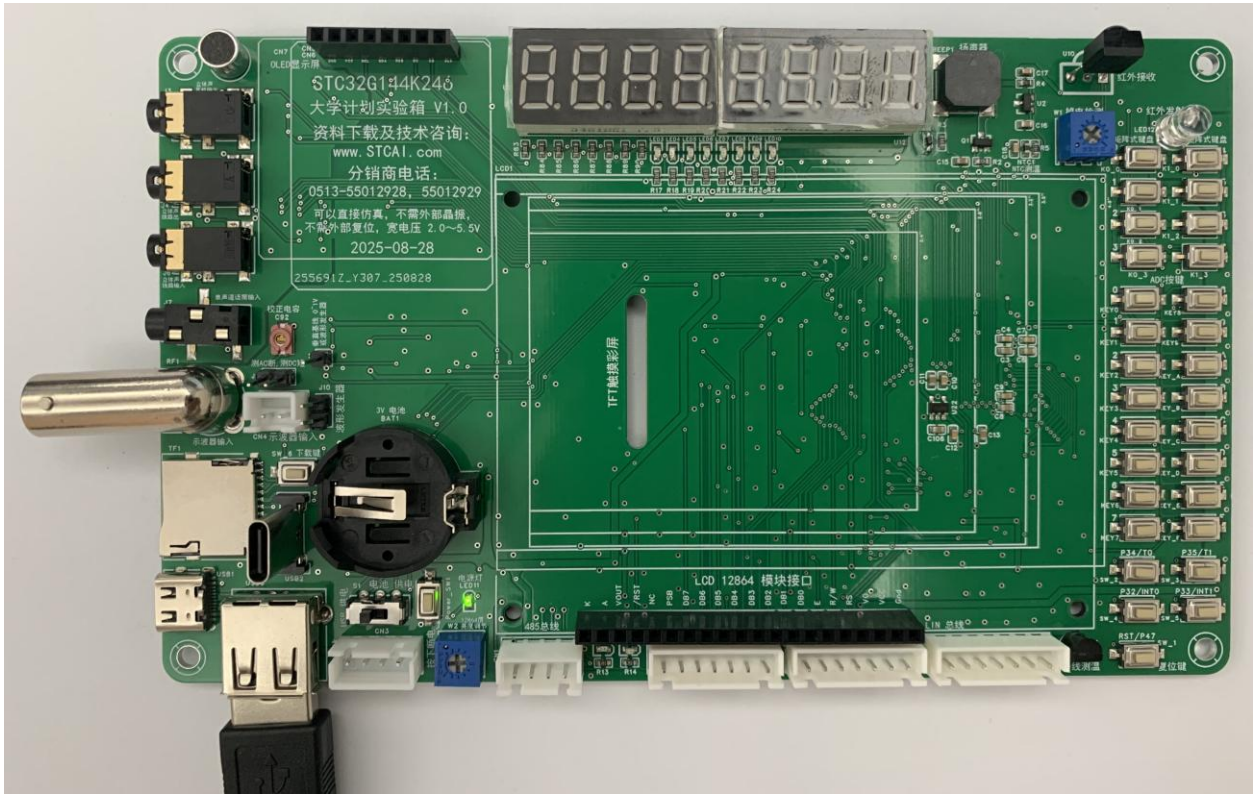


如果是解压缩到非桌面的其他目录，如 D:\STCAI-ISP，建议创建快捷方式到桌面方便打开。这实际是一个免安装的绿色软件。

双击 AiCube-ISP 系统软件图标，打开 AiCube-ISP 软件界面，即可直接使用，如下图：



2.22.1.2 STC32G144K246 实验箱完整实物图（不含外壳）：正面



2.22.1.3 USB Link1D 工具

正面：实物图



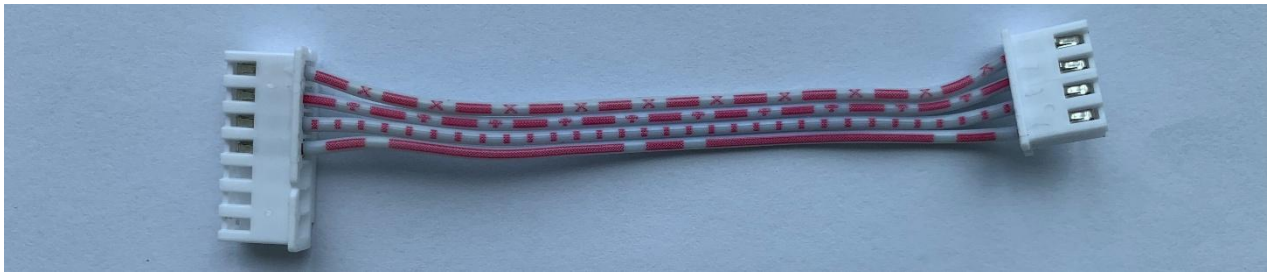
正面（拔掉两端的盖子）



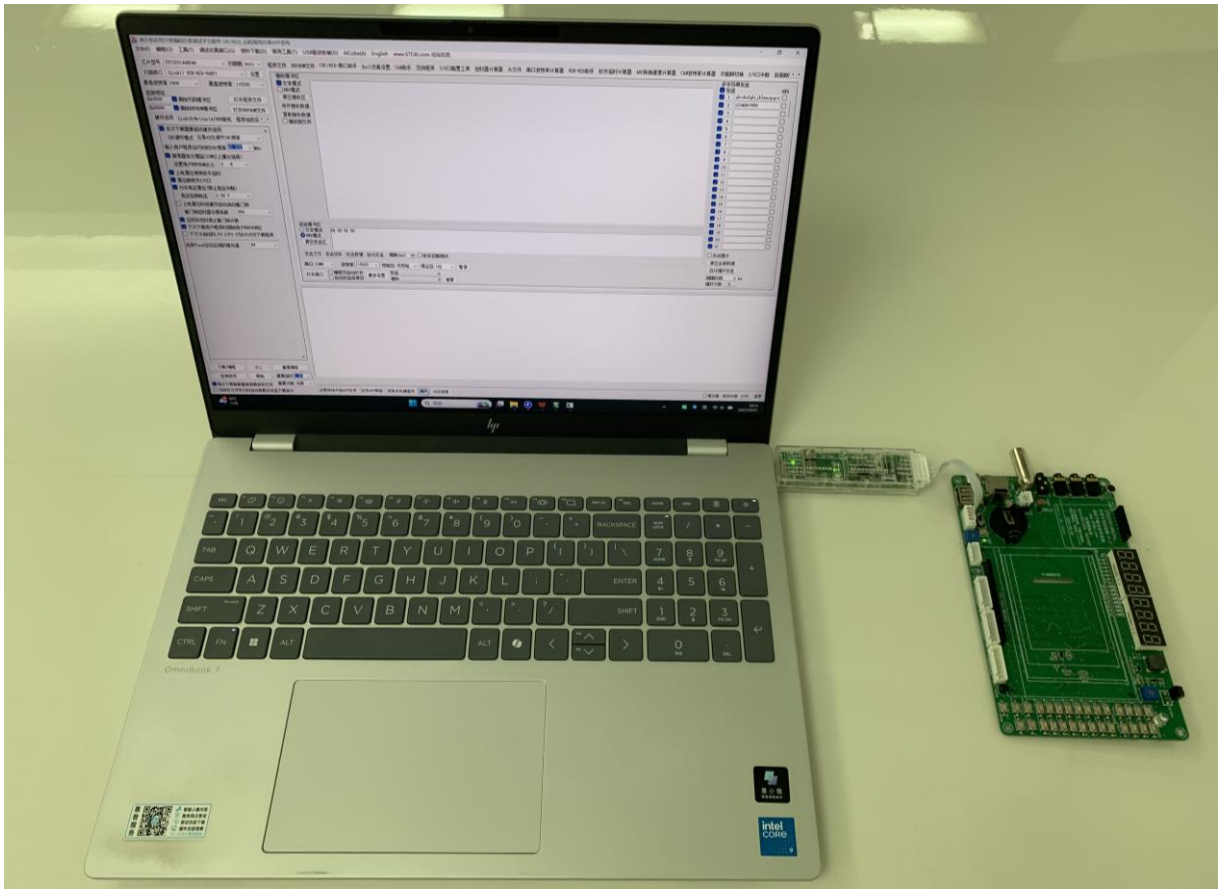
背面（拔掉两端的盖子）



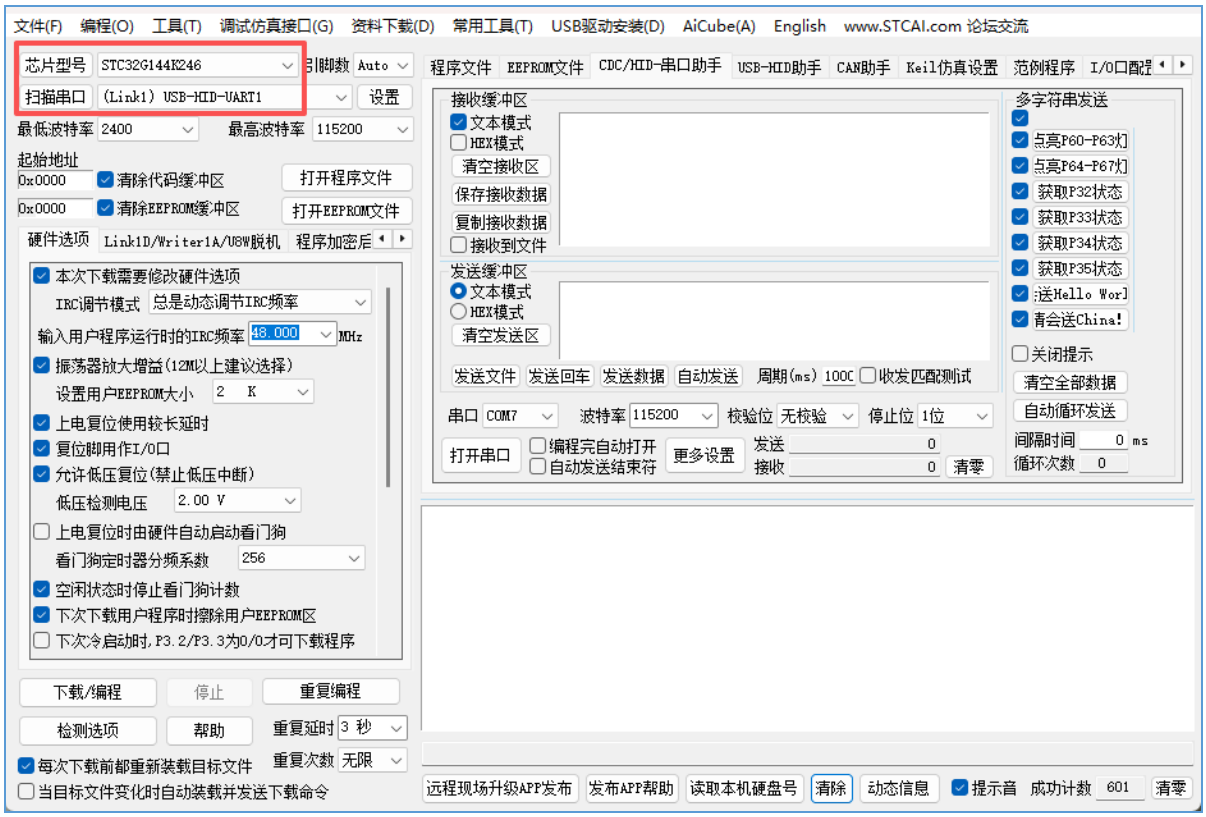
2.22.1.4 USB 转双串口/TTL 连接线：正面



2.22.1.5 STC32G144K246 实验箱通过 USB Link1D 工具与电脑连接实物图



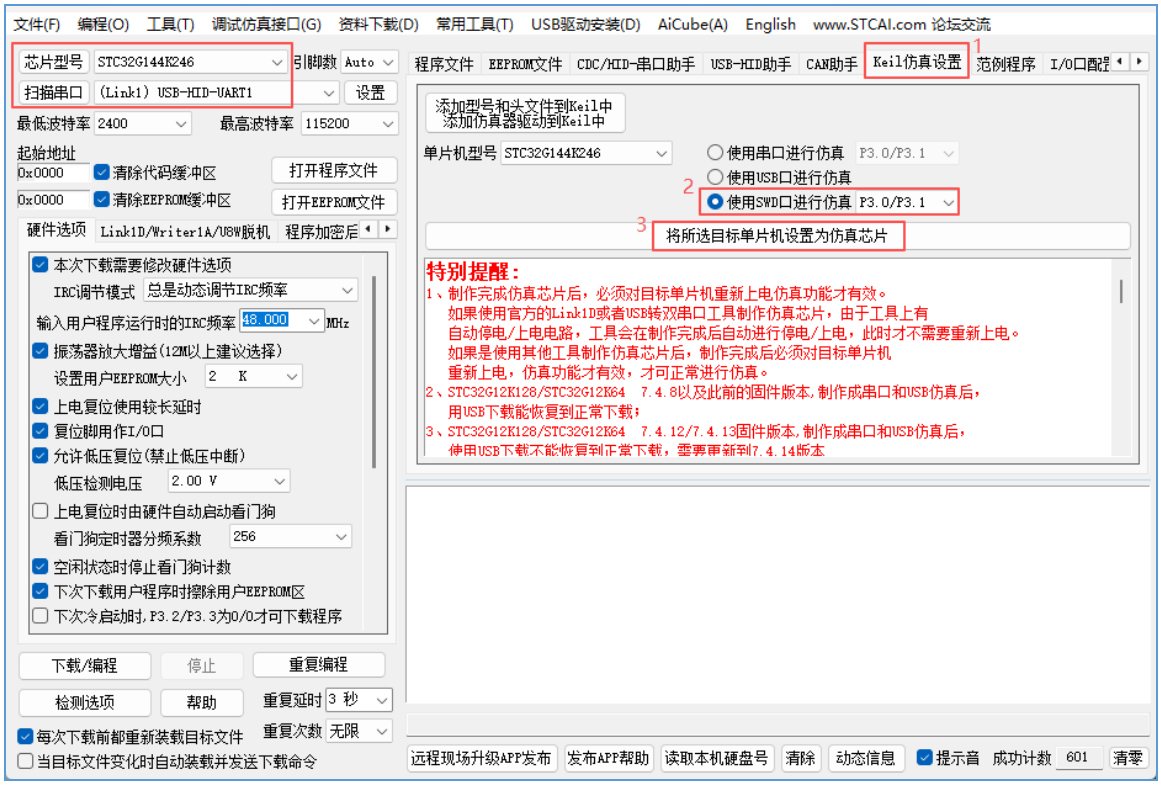
现在运行 AiCube-ISP-v6.96D 或以上版本。选择单片机型号：STC32G144K246，如下图，我们看到“扫描串口”后面的端口列表框显示：（Link1）USB-HID-UART1”，如下图。



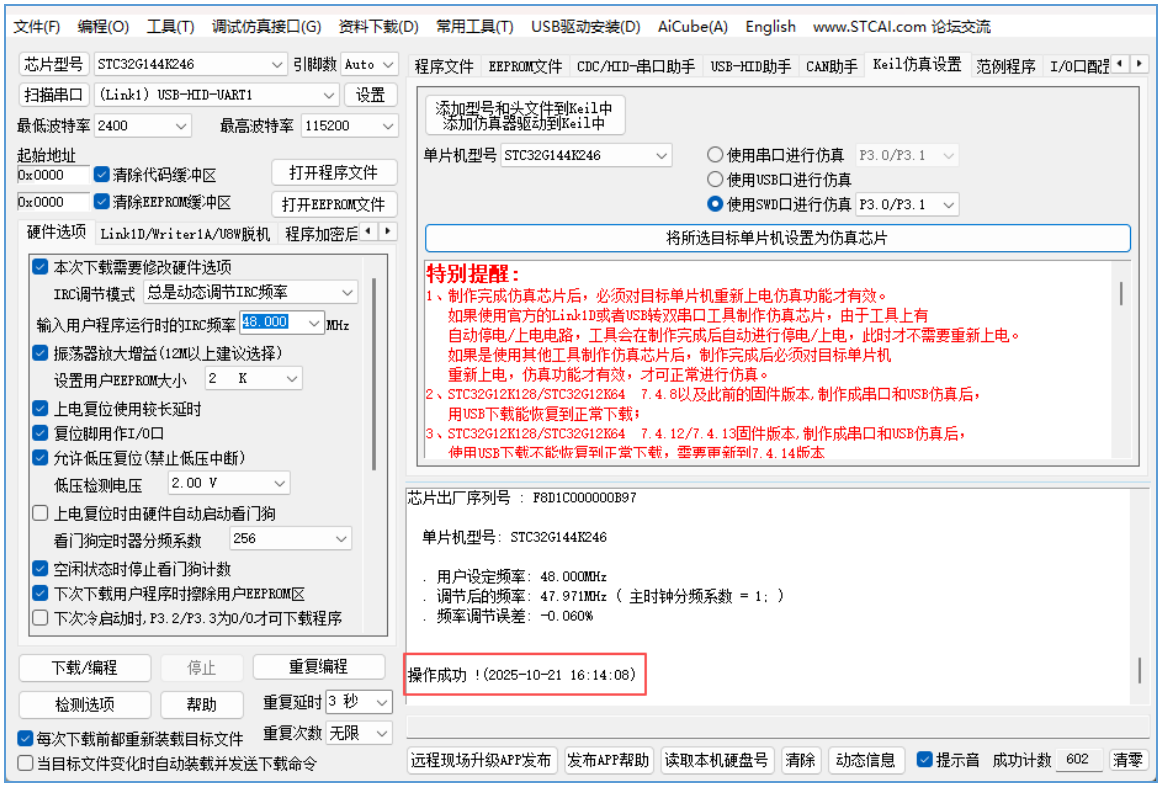
2.22.1.6 制作仿真芯片

在 AiCube-ISP 系统软件界面：

➲ 点击“Keil 仿真设置”标签，在“Keil 仿真设置”选项卡：
选择“使用 SWD 口进行仿真”：默认为“P3.0/P3.1”（那么程序里就不能出现任何占用 P3.0/P3.1 引脚的功能，比如：USB-CDC 功能）
点击“将所选目标单片机设置为仿真芯片”



然后 AiCube-ISP 软件自动进行设置仿真芯片的操作，界面下面的进度条在推进，最后“操作成功！”

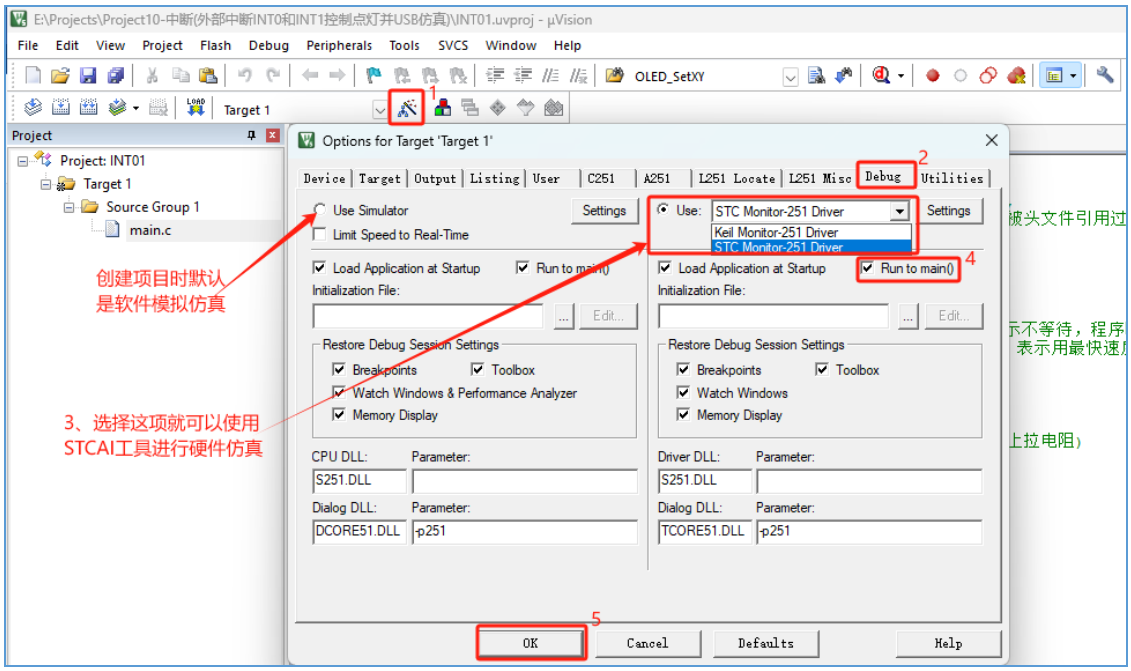


2.22.2 Debug 和 Utilities 选项设置--在 Keil 集成环境中

仿真芯片制作完成后，打开要仿真的项目 Keil 文件，在 Build 工具栏找到一个类似魔术棒的工具，点击打开项目选项设置界面，在 Keil 集成环境中进行与 Debug 相关的必要设置。

2.22.2.1 Debug 选项卡--使用 STCAI 工具进行硬件仿真设置

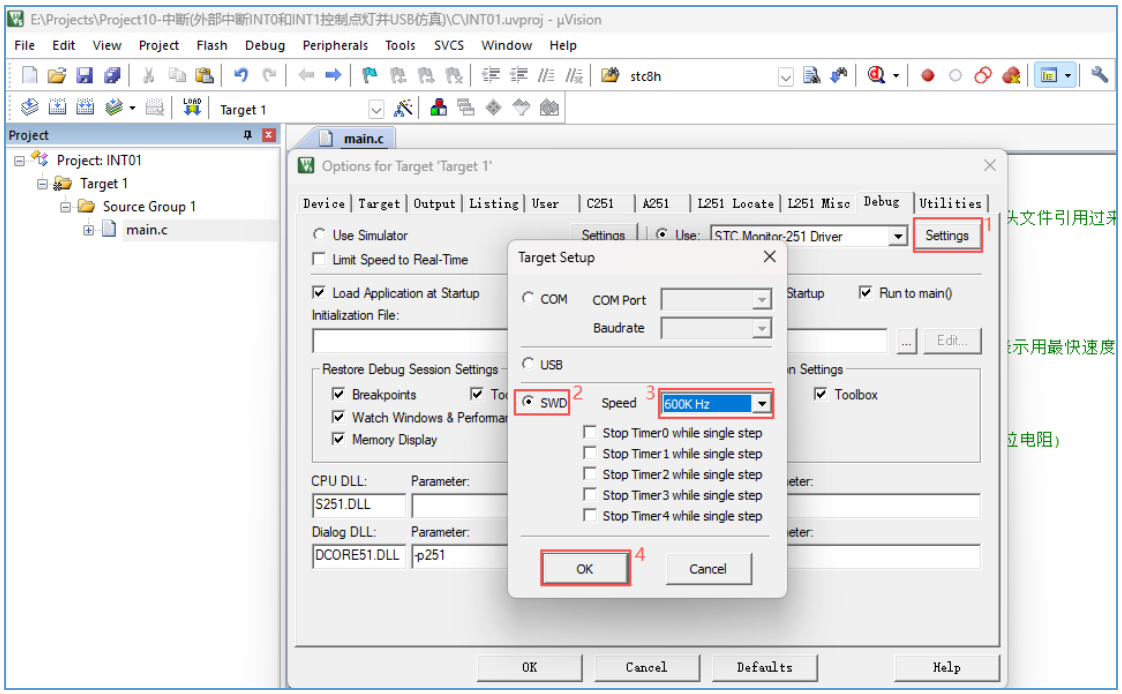
切换到 Debug 选项卡，创建项目时，Keil 默认是软件模拟仿真。我们可以借助 STCAI 的工具进行硬件仿真，如下图：



STC32G144K246 是 32 位 MCU，可以 USB 直接通信仿真、COM 串口通信仿真、SWD 硬件仿真。

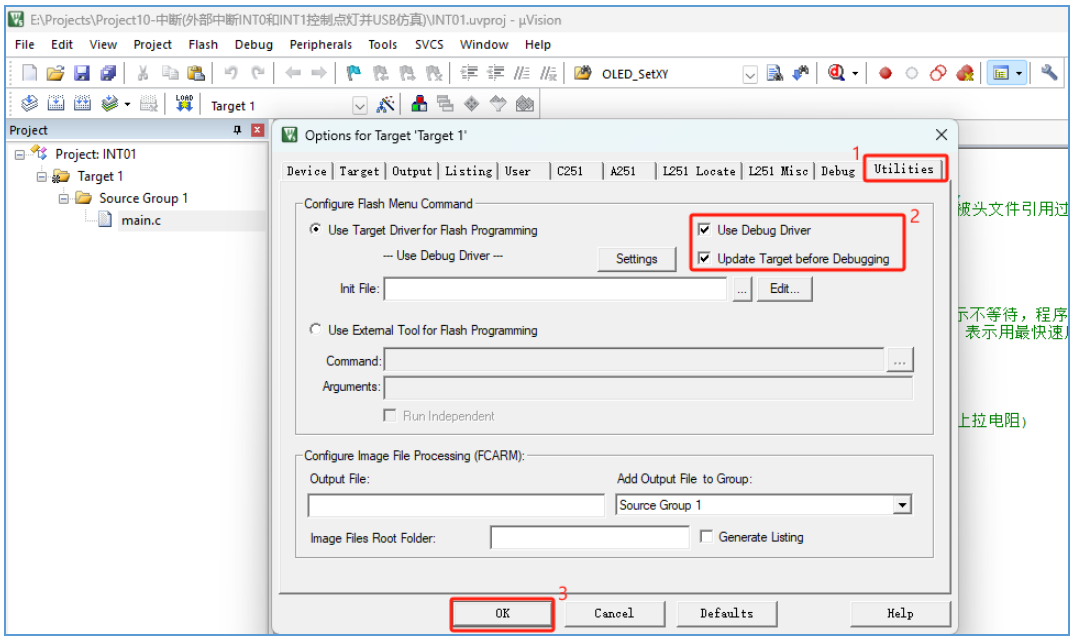
本文介绍 STC32G144K246 实验箱通过 USB Link1D 工具进行 **SWD 硬件通信仿真**，如下图设置，点选“SWD”：

- Speed：SWD 通信速度建议选择 600KHz
- Stop TimerX while single step：如果仿真代码中使能某个定时器中断可勾选相应的选项，否则忽略



2.22.2.2 Utilities 选项卡--勾选“Use Debug Driver”

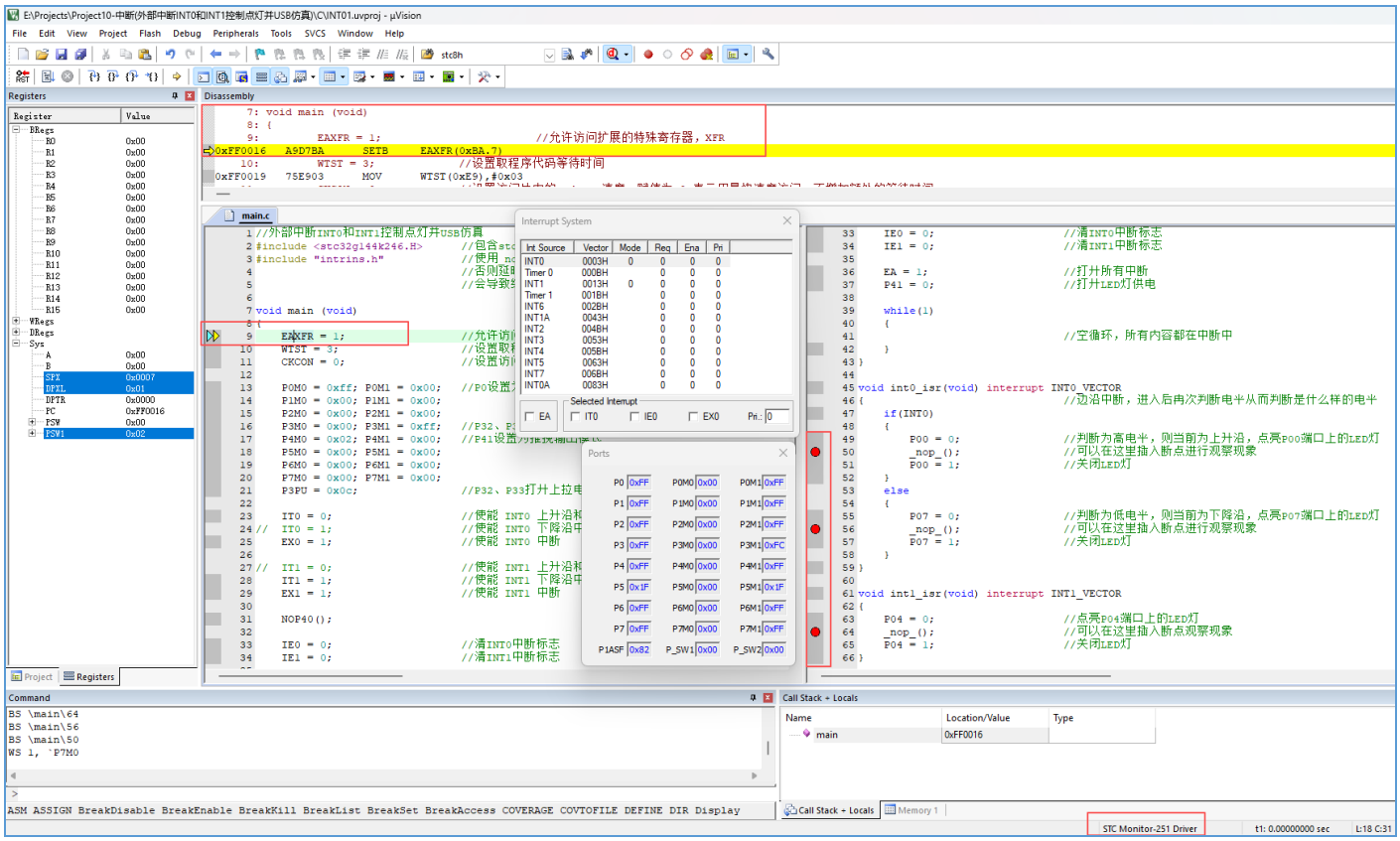
切换到“Utilities”选项卡，勾选“Use Debug Driver”，点击“OK”，完成设置。



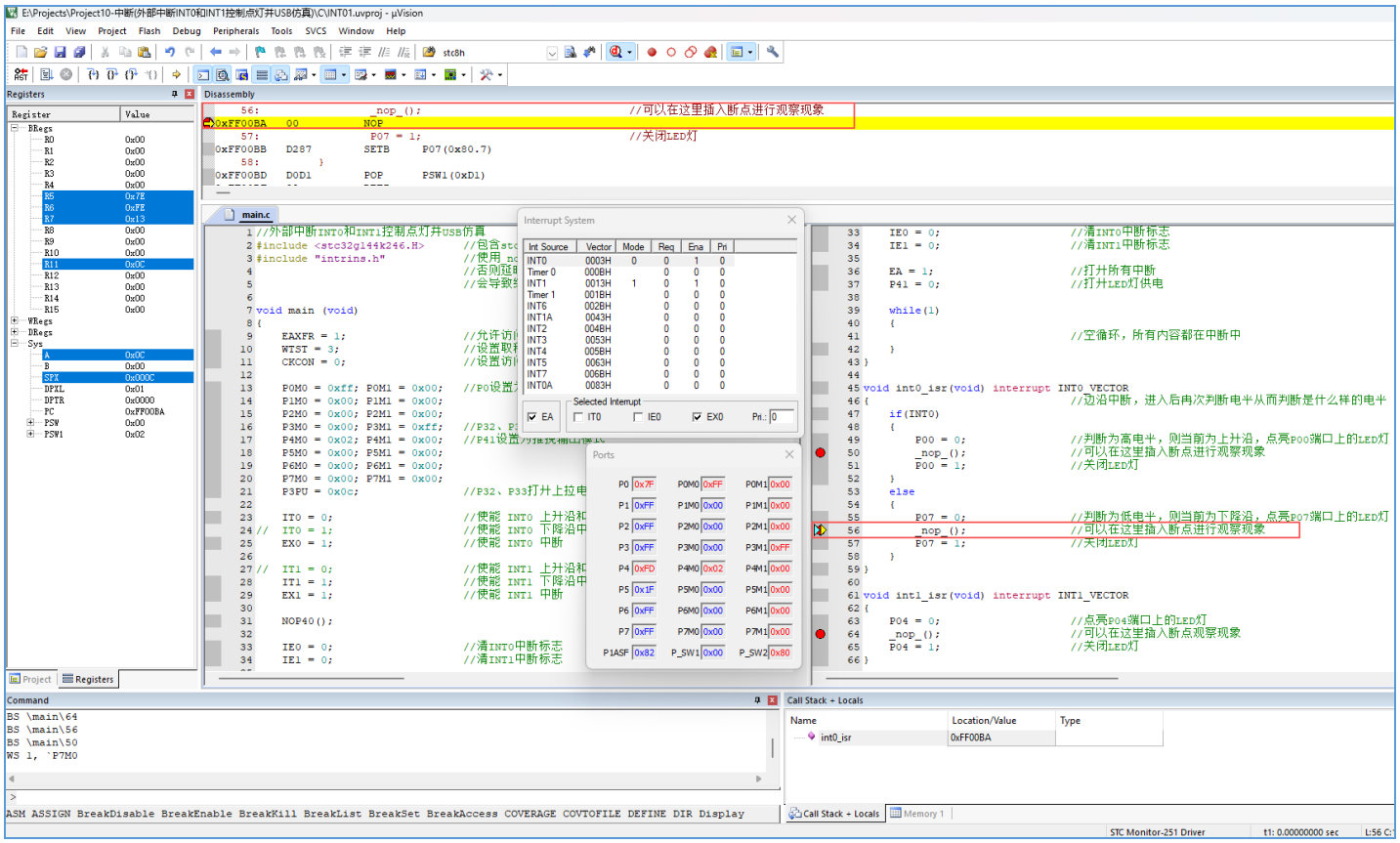
2.22.3 进入 Debug 界面，观察实验现象

步骤一：利用仿真，可以设置运行到断点处，捕捉当前程序运行位置，INT0/P3.2 是下降沿还是上升沿

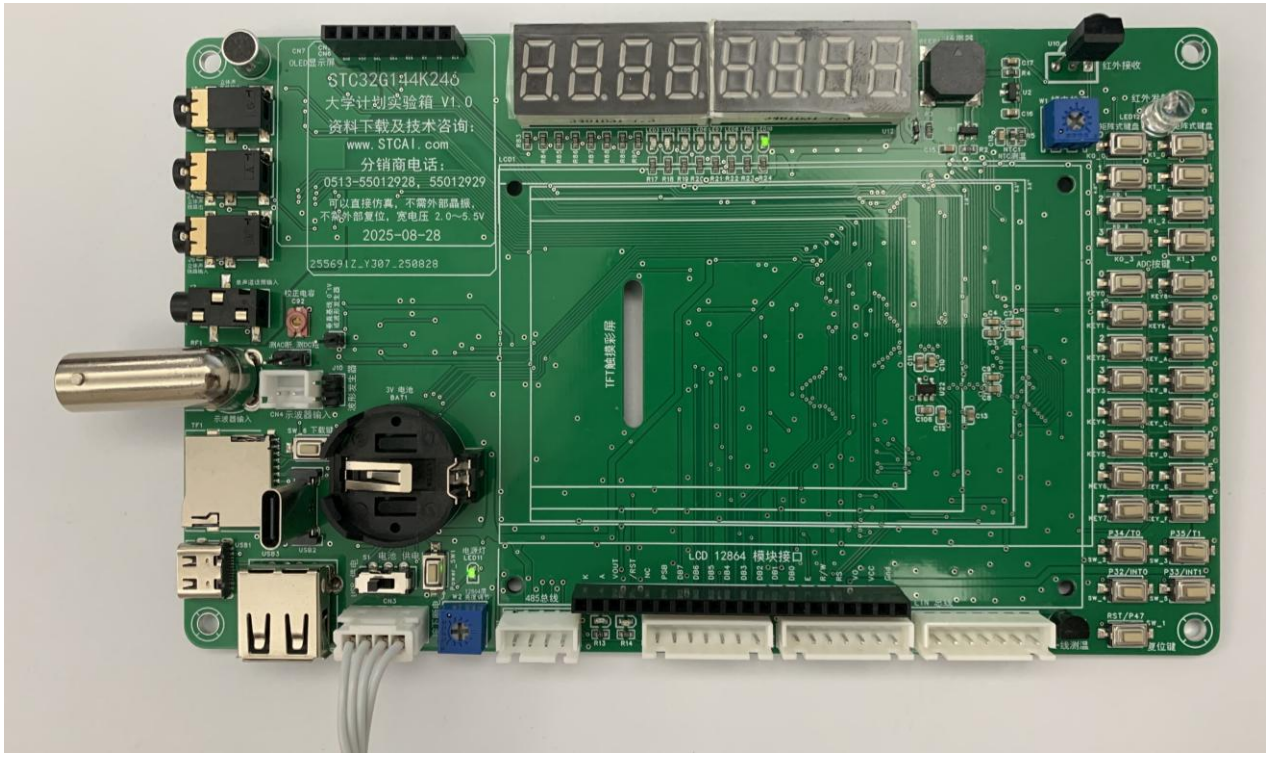
为方便观察实验现象，需在中断服务程序中设置如下 3 个断点。当然，你也可以打开全部外设窗口观察各外设数据在 Debug 过程中的变化：



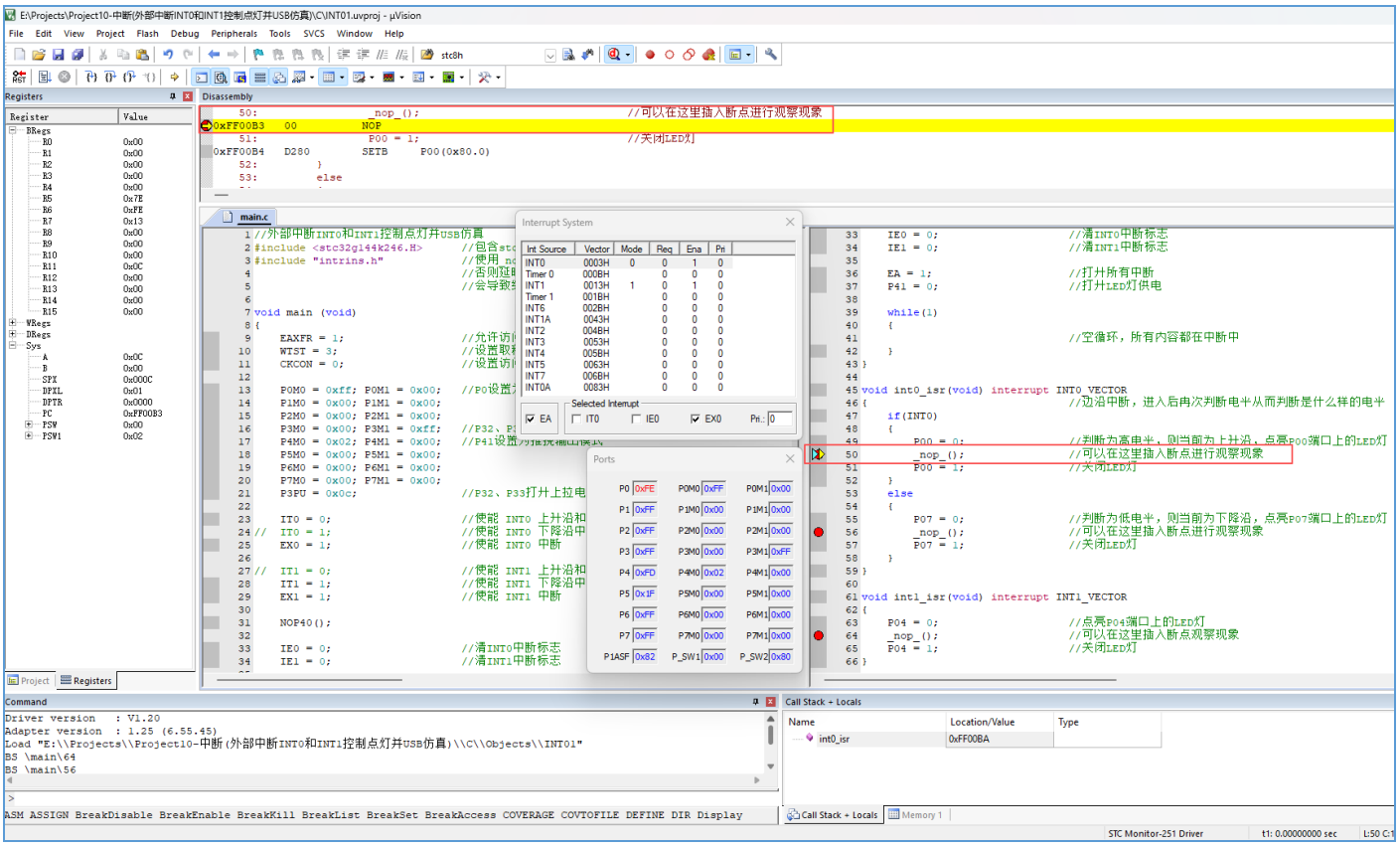
先点击 Run（运行）按钮，再按住 P3.2 按键，保持不松开（为了验证下一步的实验），程序会停在如下图所示断点处。同时，相较于上图，我们观察到：Ports / Interrupt System / Call Stack / Registers / Disassembly 这些窗口显示的数据都有可见的变化。



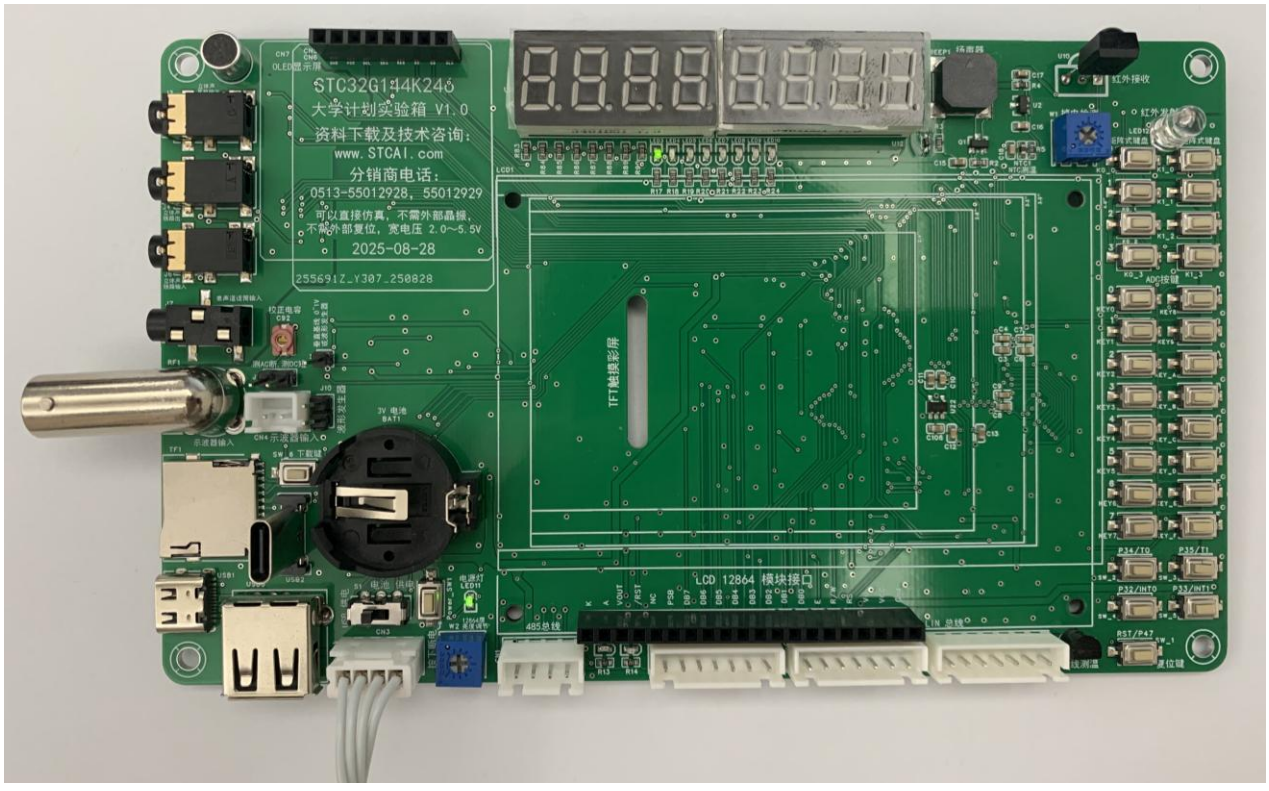
进 INT0 中断后，读到 INT0/P3.2 口是 低，下降沿，点亮 LED0.7，STC32G144K246 实验箱如下图：



继续点击 Run（运行）按钮运行，退出中断服务程序。再松开 P3.2 按键，程序会停在如下图所示断点处。同时，相较于上图，我们观察到：Ports / Registers / Disassembly 这些窗口显示的数据都有可见的变化。



进 INT0 中断后，读到 INT0/P3.2 口是 高，上升沿，点亮 LED0.0，STC32G144K246 实验箱如下图：

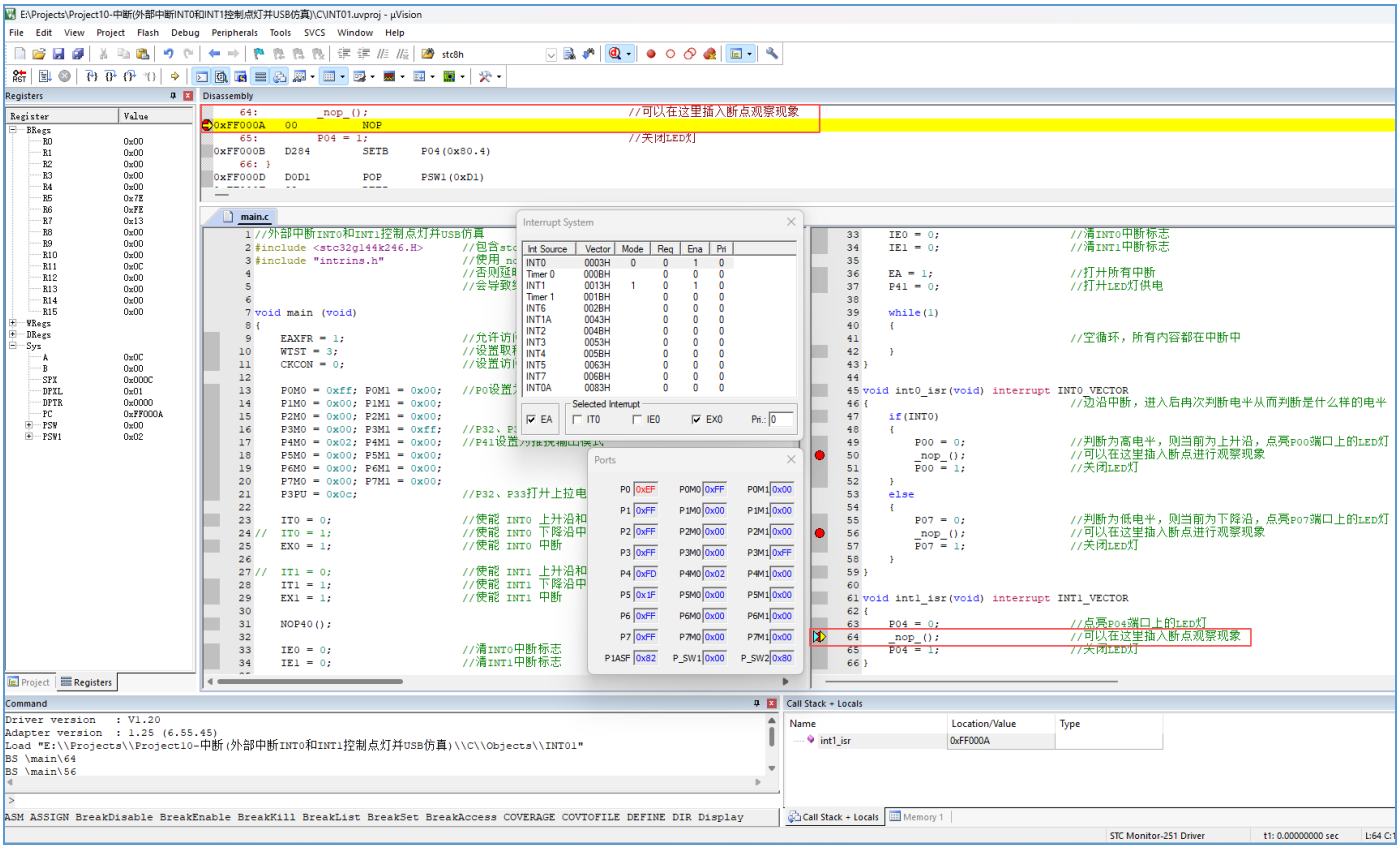


继续点击 Run（运行）按钮运行，退出中断服务程序。

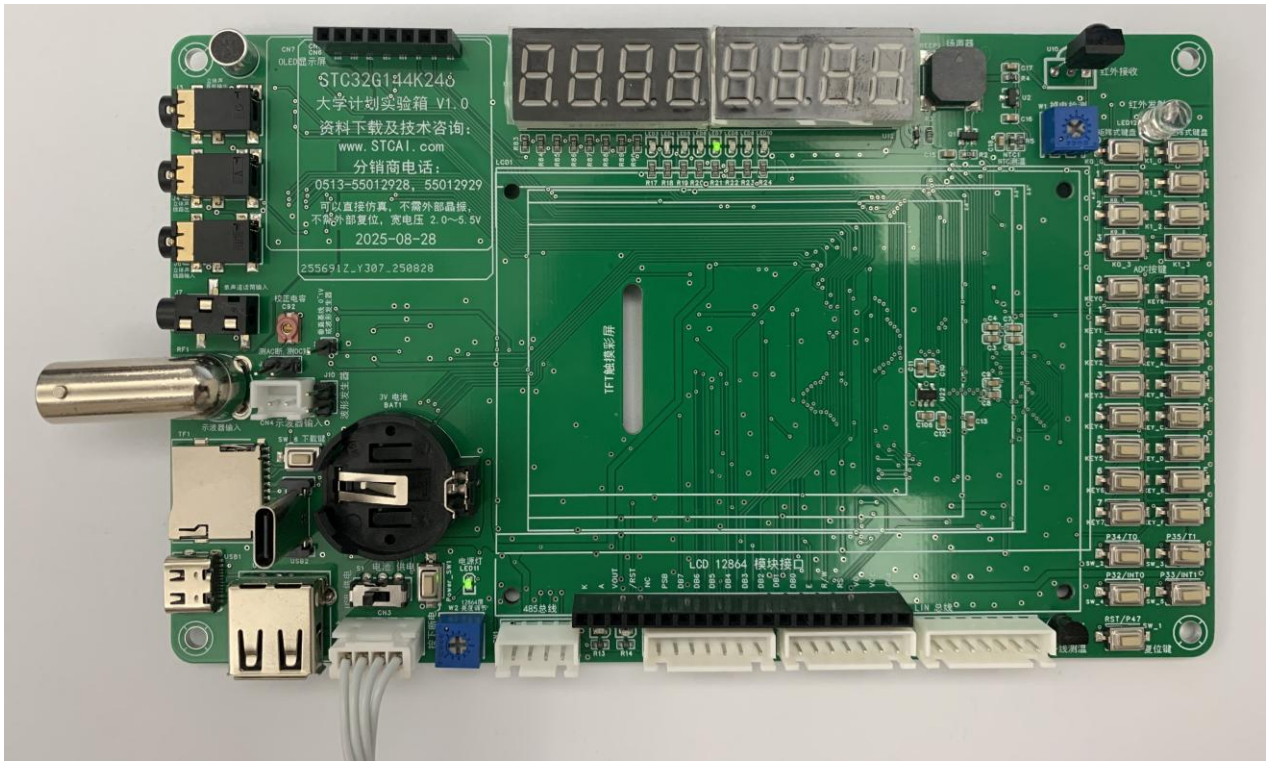
步骤二：利用仿真，可以设置运行到 INT1/P3.3 外部中断的断点处

如果程序没有在运行，需先点击 Run（运行）按钮，再按一下 P3.3 按键，程序会停在如下图所示断点处。

同时，相较于上图，我们观察到：Ports / Call Stack / Registers / Disassembly 窗口的数据都有可见的变化。



进 INT1 中断后，点亮 LED0.4，STC32G144K246 实验箱如下图：



继续点击 Run（运行）按钮运行，退出中断服务程序。

完成本实验。

3 STC32G144K246 系列选型简介、特性、价格、管脚图

3.1 STC32G144K246-LQFP100 总体介绍，USB 下载，烧录 /仿真 线路图

3.1.1 特性及价格

➢ 选型价格（不需要外部晶振、不需要外部复位，两组 12 位 ADC，15+15 通道）

单片机型号	工作电压（V）	Flash 程序存储器 10 万次 字节	edata 内部扩展 DATA RAM 可做堆栈或变量 字节	xdata 内部大容量扩展 SRAM 可做变量 字节	EEPROM 10 万次 字节	I/O 口最多数量	传统 I/O 中断（INT0/INT1/INT2/INT3/INT4）并可掉电唤醒	所有 I/O 口均支持高电平 低电平 上升沿 下降沿中断，可掉电唤醒	DMA 8080/6800 接口 TFT 彩屏模块驱动(8 位和 16 位)	RTC 实时时钟 年/月/日/时/分/秒/星期 对内部时钟追频 掉电唤醒	全速 USB，支持 USB-CDC, USB-HID	DMA USART - LIN / SPI / IrDA 并可 掉电唤醒	汽车电子通讯 LIN 总线	DMA I^2S 音频总线 掉电唤醒（全双工录音 + 放音）	DMA SPI 并可掉电唤醒	DMA QSPI	DMA I^2C 并可掉电唤醒	DMA 汽车电子通讯 CAN-FD 总线 并可掉电唤醒	TFPU (240MHz)硬件浮点 / 硬件三角 / 反三角函数运算器	DSP32，硬件 32 位/64 位 DSP 指令	定时器/计数器（T0～T11, T17/T18 外部管脚 也可掉电唤醒）	240MHz，16 位高级 PWM 定时器 互补对称死区控制	掉电唤醒专用定时器	DMA 支持两个独立的 12 位 ADC，每个 16 通道，共 32 通道	DMA 支持两个独立的 12 位高速 DAC	比较器（6P6N 模式，6 个正极输入，6 个负极输入）	运算放大器（OPA）	内部低压检测中断 并可掉电唤醒	看门狗 复位定时器	内部高可靠复位（可选复位门槛电压）	内部高精准时钟（74MHz 以下可调），满足串口通信，追频	可对外输出时钟及复位	程序加密后传输（防拦截），可设置下次更新程序需口令	支持 RS485 下载	本身就可在线仿真，支持硬件 USB 直接下载	价格及封装				供货信息
																																							LQFP100	LQFP64	LQFP48	LQFP44	
STC32G144K246	1.9-5.5	246K	16K	128K	IAP	91	有	有	有	有	有	8	2	2	3+8	有	2	2	有	有	14	24	有	2 组	2 组	4 组	4 组	有	有	4 级	有	是	是	是	是					

可实现 8+2 组 LIN

➢ **内核**

- ✓ 100MHz ~ 120MHz CPU/DSP，240MHz TFPU/PWM
- ✓ 超高速 32 位 8051 内核（1T），比传统 8051 同频约快 70 倍以上
- ✓ 4 级中断优先级
- ✓ 支持 USB、SWD 在线仿真

➢ **工作电压**

- ✓ 1.9V ～ 5.5V（当工作温度低于-40℃时，工作电压不得低于 3.0V，低温时工作电压要提高）



扫码去微信小商城

➢ **工作温度**

- ✓ -20℃ ~ +65℃（内部高速 IRC 温漂-0.76% ~ +0.98%）
- ✓ -40℃ ~ +85℃（内部高速 IRC 温漂±1.3%）
- ✓ -40℃ ~ +125℃（内部高速 IRC 温漂±3%。当温度高于 85 度时，请使用外部 35MHz 或以下的晶振，再内部 PLL 到高频时钟如 100MHz/200MHz。）

➢ **Flash 存储器**

- ✓ 最大 246K 字节 FLASH 程序存储器（ROM），用于存储用户代码
- ✓ 支持用户可配置部分 FLASH 作为 DATA FLASH/EEPROM 使用，512 字节单页擦除，**典型单个扇区擦除次数可达 10 万次，用户保守可当 2 万次使用**
- ✓ 支持硬件 USB 直接下载和普通串口下载
- ✓ 支持硬件 SWD 实时仿真，P3.0/P3.1（需 USB-Link1D 工具）
- ✓ **Flash 数据保持时间：100 年以上@25℃，20 年以上@105℃，15 年以上@125℃。由于汽车及单片机的平均温度不可能长时间大于 105℃，从严考虑，Flash 数据保持时间可达 20 年以上，不保证 100 年。**

- **SRAM，共 144K+4K 字节**
  - ✓ 16K 字节内部 SRAM（edata）
  - ✓ 128K 字节内部扩展 RAM（内部 xdata）
  - ✓ **4K 字节高端扩展 RAM（可映射到 80H:0000H～80H:0FFFH 区执行用户程序）**

- **时钟控制**
  - ✓ 内部高精度、高稳定的高速 IRC（ISP 编程时可进行上下调整）
    - ⌖ 误差±0.3%（常温下 25℃）
    - ⌖ -1.35% ～ +1.30%温漂（全温度范围，-40℃ ～ 85℃）
    - ⌖ -0.76% ～ +0.98%温漂（温度范围，-20℃ ～ 65℃）
  - ✓ 内部 32KHz 低速 IRC（为了低功耗，省去了温度补偿和电压补偿电路，误差较大）
  - ✓ 支持外部晶振（4MHz ～ 40MHz）和外部时钟，内部设计有专门的外部时钟抗干扰电路，外部时钟可内部 PLL 到高频给 MCU 使用
  - ✓ **内部两组独立的 500MHz 的 PLL，可输出 240MHz 的时钟提供给 TFPU、PWM、I2S、SPI 等高速外设**

  用户可自由选择上面的 4 种时钟源

  （芯片上电工作过程：上电复位/复位脚复位/看门狗复位/低压检测复位时，芯片默认从 ISP 系统程序开始执行代码，此时固定使用内部 24MHz 的高速 IRC 时钟，当需要下载用户程序且下载完成后复位到用户程序区或者不需要下载直接复位到用户程序区时，默认会使用上次用户下载时所调节的高速 IRC 时钟，如果用户程序需要使用外部高速晶振、外部 32.768KHz 晶振或者内部 30KHz 低速 IRC，则需要用户软件先启动相应的时钟，然后通过设置 CLKSEL 寄存器进行切换）

- **复位**
  - ✓ 硬件复位
    - ⌖ 上电复位，复位电压值为 1.7V ～ 1.9V。（在芯片未使能低压复位功能时有效）
    - ⌖ 复位脚复位，出厂时 P5.4 默认为 I/O 口，ISP 下载时可将 P5.4 管脚设置为复位脚（注意：当设置 P5.4 管脚为复位脚时，复位电平为低电平）
    - ⌖ 看门狗溢出复位
    - ⌖ 低压检测复位，提供 4 级低压检测电压：2.0V、2.4V、2.7V、3.0V。
  - ✓ 软件复位
    - ⌖ 软件方式写复位触发寄存器

- **中断**
  - ✓ 中断源：INT0、INT1、INT2、INT3、INT4、定时器 0、定时器 1、定时器 2、定时器 3、定时器 4、定时器 5、定时器 6、定时器 7、定时器 8、定时器 9、定时器 10、定时器 11、定时器 17、定时器 18、USART1、USART2、USART3、USART4、USART5、USART6、USART7、USART8、ADCA 模数转换、ADCB 模数转换、DACA 数模转换、DACB 数模转换、LVD 低压检测、SPI1、SPI2、SPI3、I^2C1、I^2C2、比较器 1、比较器 2、比较器 3、比较器 4、PWMA、PWMB、PWMC、PWMD、PWME、PWMF、USB、TFT 彩屏接口中断、RTC 实时时钟、所有的 I/O 中断（8 组）、所有的 USART 串口的 DMA 接收和发送中断（8 组）、I2C 的 DMA 接收和发送中断（两组）、I2S 的 DMA 接收和发送中断（两组）、SPI 的 DMA 中断（3 组）、QSPI 的 DMA 中断、ADC 的 DMA 中断（两组）、DAC 的 DMA 中断（两组）、LCD 驱动的 DMA 中断、CAN-FD 的 DMA 中断（两组）、PWM 的 DMA 中断（PWMA/PWMC/PWME）、存储器到存储器的 DMA 中断以及**两组外设到外设的 DMA 中断**。
  - ✓ 提供 4 级中断优先级

- **数字外设**
  - ✓ 14 个 16 位定时器：定时器 0、定时器 1、定时器 2、定时器 3、定时器 4、定时器 5、定时器 6、定时器 7、定时器 8、定时器 9、定时器 10、定时器 11、定时器 17、定时器 18，其中定时器 0 的模式 3 具有 NMI（不可屏蔽中断）功能，定时器 0 和定时器 1 的模式 0 为 16 位自动重载模式。定时器 11：可选时钟源。

定时器 17 和定时器 18 可同步触发两组 ADC 或者同步触发两组 DAC。

- ✓ 8 个高速同步/异步串口：串口 1（USART1）、串口 2（USART2）、串口 3（USART3）、串口 4（USART4）、串口 5（USART5）、串口 6（USART6）、串口 7（USART7）、串口 8（USART8），波特率时钟源最快可为 FOSC/4。支持同步串口模式、异步串口模式、SPI 模式、LIN 模式、红外模式（IrDA）、智能卡模式（ISO7816）
- ✓ 24 通道高级 PWM，3+3 组高级 PWM，每组均可实现 4 通道 PWM。PWMA/PWMC/PWME 均可实现 4 对互补对称带死区控制的 PWM，PWMB/PWMD/PWMF 均可实现 4 通道 PWM 输出，并支持外部异常检测功能。PWM 时钟源的输入**速度可支持到 240MHz**
- ✓ SPI：3+8 组硬件 SPI（3 组独立 SPI，8 组 USART 的 SPI 模式）支持主机模式和从机模式以及主机/从机自动切换（注：11 组 SPI 均支持 DMA，11 组 SPI 的 MISO 和 MOSI 脚均可软件设置进行功能交换），SPI 的时钟源支持 **240MHz**
- ✓ QSPI：支持单线、双线和四线模式
- ✓ I^2C：两组独立的硬件 I^2C，均支持主机模式和从机模式
- ✓ ICE：支持硬件仿真，支持串口和 USB 口直接仿真，还支持 SWD 硬件仿真
- ✓ RTC：支持年、月、日、星期、时、分、秒、次秒（1/128 秒），并支持时钟中断和一组闹钟。芯片复位时 RTC 的时间计数值不复位。
- ✓ USB：USB2.0/USB1.1 兼容全速 USB，**16 个双向端点**，支持 4 种端点传输模式（控制传输、中断传输、批量传输和同步传输），端点拥有 64/128/256/512/1024 字节的 FIFO 缓冲区，不同的端点的 FIFO 缓冲区大小不等。
- ✓ CAN：两个独立的 CAN-FD 2.0 控制单元。CAN-FD 的 STB 脚可软件设置是否使能，若不使能 STB 脚可当作 GPIO 使用。
- ✓ LIN：2+8 组硬件 LIN（两组组独立 LIN，8 组 USART 的 LIN 模式），一个独立的 LIN 控制单元（支持 1.3 和 2.1 版本）
- ✓ I2S：两路独立的 I2S 音频总线，**可支持全双工方式的录音和放音**
- ✓ DSP32：硬件 32 位/64 位 DSP 运算指令，**速度可达 74MHz**
- ✓ TFPU：单精度浮点运算器（支持浮点加、减、乘、除以及正弦、余弦、正切和反正切等运算），**速度可达 240MHz**
- ✓ I/O 口中断：所有的 I/O 均支持中断，每组 I/O 中断有独立的中断入口地址，所有的 I/O 中断可支持 4 种中断模式：高电平中断、低电平中断、上升沿中断、下降沿中断。I/O 口中断可以进行掉电唤醒，且有 4 级中断优先级。
- ✓ LCD/TFT 彩屏驱动接口：支持 8080 和 6800 两种接口以及 8 位和 16 位数据宽度。对 16 位数据支持大小端数据格式选择。
- ✓ DMA：支持 SPI（3 组）、QSPI、I2C（两组）、I2S（两组）、串口（8 组）、PWM（3 组，PWMA/PWMC/PWME）、ADC（两组，ADCA/ADCB）、DAC（两组，DACA/DACB）、CAN-DA（两组）、LCM 的 DMA，以及外设到外设直接的 DMA
- ✓ 硬件数字 ID：支持 32+32 字节

➢ 模拟外设

- ✓ ADC：两组独立的高速 ADC，支持 12 位高精度 15 通道（通道 0～通道 14）的模数转换，ADC 的通道 15 用于测试内部参考电压（芯片在出厂时，内部参考电压调整为 1.19V，误差±1%）
- ✓ DAC：两组独立的高速 DAC，支持 12 位转换精度。两组 DAC 的模拟输出均从芯片内部直接连接到比较器和运算放大器。
- ✓ 比较器：4 组独立的 6P6N 模式比较器（CMP）
- ✓ 运算放大器：4 组独立的运算放大器（OPA）

➢ GPIO

- ✓ 最多可达 91 个 GPIO：P0.0 ~ P0.7、P1.0 ~ P1.7（P1.2 和 P5.4 为同一个 PIN 脚、P1.3 和 P1.7 为同一个 PIN

脚）、P2.0 ~ P2.7、P3.0 ~ P3.7、P4.0 ~ P4.7、P5.0 ~ P5.4、P6.0 ~ P6.7、P7.0 ~ P7.7、P8.0 ~ P8.7、P9.0 ~ P9.7、PA.0 ~ PA.7、PB.0 ~ PB.7

- ✓ 所有的 GPIO 均支持如下模式：~~准双向口模式~~、强推挽输出模式、开漏模式、高阻输入模式
- ✓ 所有 I/O 口上电后的状态均为高阻输入状态，用户在使用 I/O 口时必须先设置 I/O 口模式
- ✓ 另外每个 I/O 均可独立使能内部上拉电阻和下拉电阻，常温下：

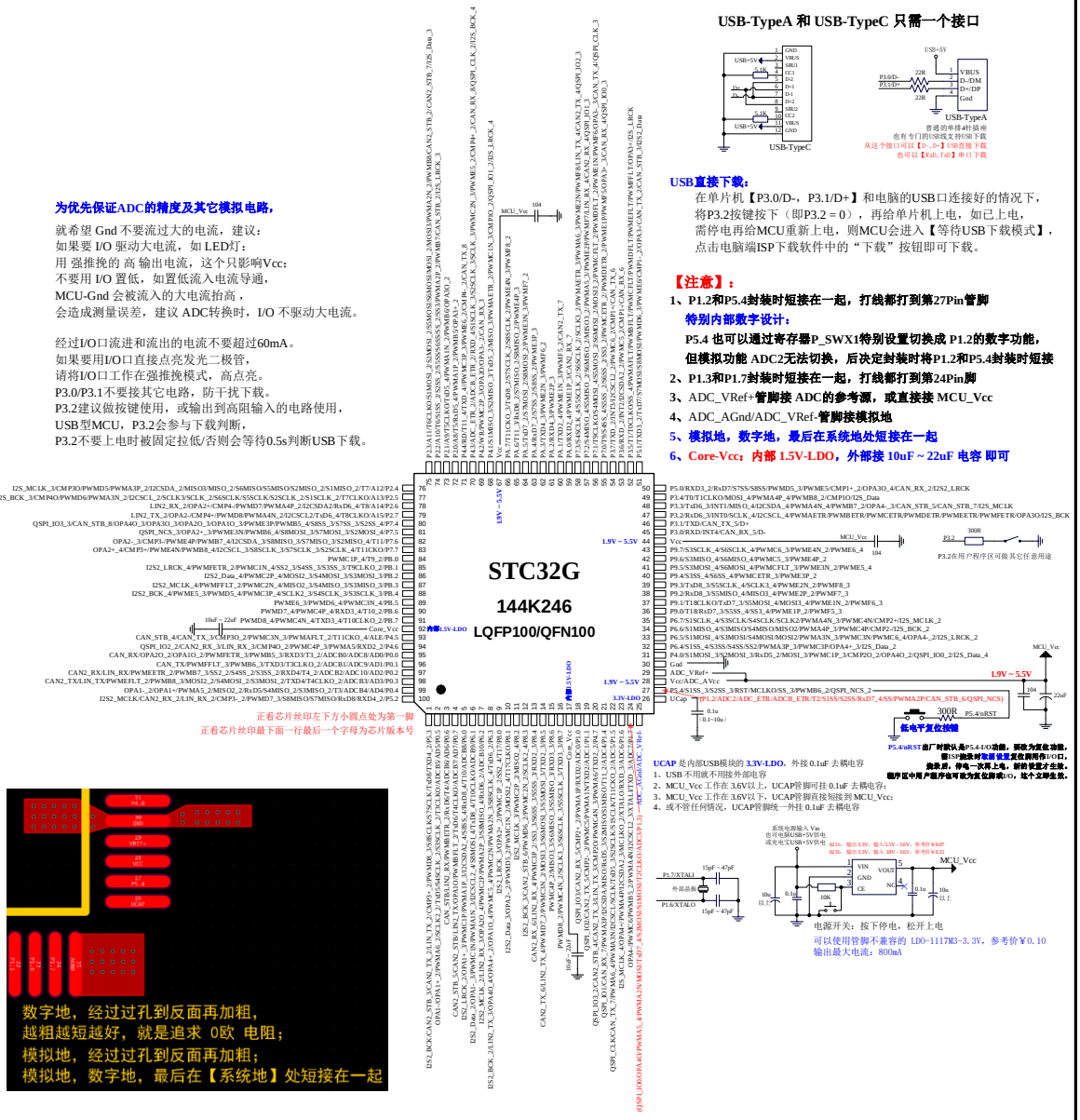
  上拉电阻阻值范围：52.7K@5V，80.9K@3.3V

  下拉电阻阻值范围：46.2K@5V，33.9K@3.3V

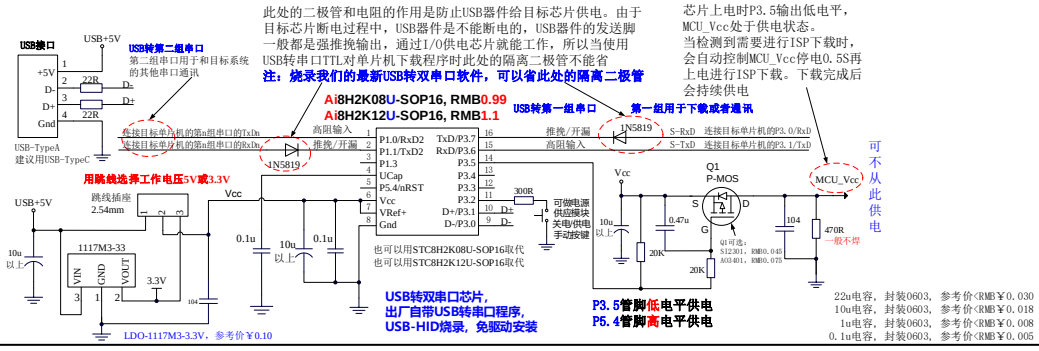
- ➢ **封装**
  - ✓ LQFP100、LQFP64、LQFP48、LQFP44

3.1.2 管脚图，最小系统（LQFP100/QFN100）

自带硬件USB，支持硬件USB 直接 下载 和 仿真



还可以通过【USB转双串口 芯片或工具】 进行 全自动 下载，支持 仿真+串口通讯，5V/3.3V跳线选择



【应用场景一：从本工具给目标系统 自动 停电/上电，供电】
点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0.5秒/再自动供电，
数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0.5秒/再自动供电给目标系统工作。

【应用场景二：不从本工具给目标系统供电】
1、点击 电脑端 ISP 软件的【下载/编程】按钮
2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，
下载编程进行中，数秒后提示下载编程成功，目标MCU会自动复位到用户程序区自动跑用户程序。
部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P5.4/nRST变成复位脚

1、CPU，DSP 可工作在 100MHz 附近
2、TFPU，PWM 可工作在 250MHz 及以下
3、144K SRAM，246K Flash

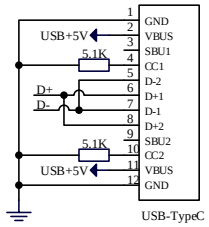
现在带硬件 USB 的 MCU 支持用硬件 USB 下载，因为用的是 USB-HID 通信协议，不需要安装任何驱动。只要 USB 鼠标、USB 键盘能工作，USB-HID 驱动就是好的，不要安装 USB-HID 驱动，免驱。

在 D-/P3.0，D+/P3.1 与 PC-USB 端口连接好的状况下，USB-ISP 下载程序有如下三种模式：

【USB 下载方法一，P3.2 按键，再结合停电上电下载】

1、按下板子上的 P3.2/INT0 按键，就是 P3.2 接地
2、给目标芯片重新上电，不管之前是否已通电。
===电子开关是按下停电后再松开就是上电
等待电脑端 ISP 下载软件中自动识别出“(HID1) USB Writer”后，就与 P3.2 状态无关了，这时可以松开 P3.2 按键（P3.2 在用户程序区可做其它任意用途）
===传统的机械自锁紧开关是按上来停电，按下去是上电
3、点击电脑端下载软件中的【下载/编程】按钮（注意：USB 下载与串口下载的操作顺序不同）
下载进行中，几秒钟后，提示下载成功！

比较一下流行的 USB-TypeC接口：



【USB 下载方法二，复位管脚低电平复位下载】

USB 连接好并已上电的情况下，外部按键复位也可进入 USB 下载模式，注意：
P5.4-RST 出厂时默认是 P5.4-I/O 功能，要改为复位功能，需 ISP 烧录时取消设置复位脚用作 I/O 口，停电一次再上电才生效，程序区中用户程序也可改为复位脚或 I/O，这个立即生效。

1、按下 P5.4-RST 外接的低电平复位按键复位 MCU，
松开复位键，MCU 从系统程序区启动，判断是否要下载用户程序，
等待电脑端 ISP 下载软件中自动识别出“(HID1) USB Writer”后
2、点击电脑端下载软件中的【下载/编程】按钮
下载进行中，几秒钟后，提示下载成功！

备注：如果目标 MCU 的用户程序区已有 USB 通讯程序（含 CDC、HID 等），则通过复位脚复位进入 USB 下载模式不被支持，只能通过停电上电冷启动下载或者从用户程序区软复位到系统区下载

【USB 下载方法三，从用户程序区软复位到系统区下载】

USB 连接好并已上电的情况下，从用户程序区软复位到系统区也可进入 USB 下载模式

1、在用户程序区运行软复位到系统区的程序，就是 IAP_CONTR 寄存器送 60H
等待电脑端 ISP 下载软件中自动识别出“(HID1) USB Writer”后
2、点击电脑端下载软件中的【下载/编程】按钮
下载进行中，几秒钟后，提示下载成功！

USB 下载 注意事项：

拔插 USB 插头不能代替上面线路图中的电源开关。正常操作步骤如下：

USB 的【Gnd，D+，D-】接好的情况下，按下 P3.2 按键接地，再通过正常的电源开关给 MCU 供电或重新供电，让 MCU 冷启动进入系统程序区，判断是否需要等待电脑端 USB 下载程序。

拔插 USB 插头代替电源开关不能每次都能成功进行 USB 下载的原因：

拔插 USB 插头，如【Gnd，USB+5V】已接触好，已供电，而【D+，D-】有一个甚至两个信号线还没有接触好，MCU 已上电，开始跑系统区程序时，发现 USB 还没接触好，则会从系统区软复位到用户程序区跑用户程序，不再进入等待 USB 下载模式，本次就无法顺利进行 USB 下载。

很多人经过多次插拔 USB，才能碰到 1 次【D+，D-】接触好的情况下，【Gnd，USB+5V】才开始接触好，才开始供电，才能成功进入 USB 下载。插 USB 插头代替供电，不能保证【Gnd，D+，D-，USB+5V】的接触顺序，所以，必须使用正常的电源开关，才能确保每次下载都能成功。

3.1.3 多功能管脚切换

USART1

RxD	TxD
P3.0	P3.1
P3.6	P3.7
P1.6	P1.7
P4.3	P4.4

USART1_SPI（MOSI和MISO可切换）

U1SS	U1MOSI	U1MISO	U1SCLK
P1.2	P1.3	P1.4	P1.5
P2.2	P2.3	P2.4	P2.5
P5.4	P4.0	P4.1	P4.3
P6.4	P6.5	P6.6	P6.7

SPI（MOSI和MISO可切换）

SS	MOSI	MISO	SCLK
P1.2	P1.3	P1.4	P1.5
P2.2	P2.3	P2.4	P2.5
P5.4	P4.0	P4.1	P4.3
P3.5	P3.4	P3.3	P3.2

QSPI

NCS	IO0	IO1	SCLK	IO2	IO3
P1.2	P1.3	P1.4	P1.5	P1.1	P1.0
P5.4	P4.0	P4.1	P4.3	P4.6	P4.7
P7.5	P7.0	P7.2	P7.1	P7.3	P7.4
-	-	-	-	-	-

USART2

RxD2	TxD2
P1.0	P1.1
P4.6	P4.7
P8.4	P8.5
PA.0	PA.1

USART2_SPI（MOSI和MISO可切换）

U2SS	U2MOSI	U2MISO	U2SCLK
P1.2	P1.3	P1.4	P1.5
P2.2	P2.3	P2.4	P2.5
P5.4	P4.0	P4.1	P4.3
P7.4	P7.5	P7.6	P7.7

SPI2（MOSI和MISO可切换）

SS2	MOSI2	MISO2	SCLK2
P6.4	P6.5	P6.6	P6.7
P0.2	P0.3	P0.4	P0.5
PB.1	PB.2	PB.3	PB.4
P8.0	P8.1	P8.2	P8.3

I2C（第1个I2C模块）

SCL	SDA
P1.5	P1.4
P2.5	P2.4
P7.7	P7.6
P3.2	P3.3

I2C2（第2个I2C模块）

SCL2	SDA2
P2.7	P2.6
P3.7	P3.6
P1.7	P1.6
P6.1	P6.0

USART3

RxD3	TxD3
P0.0	P0.1
P5.0	P5.1
P8.6	P8.7
PB.6	PB.7

USART3_SPI（MOSI和MISO可切换）

U3SS	U3MOSI	U3MISO	U3SCLK
P6.4	P6.5	P6.6	P6.7
P0.2	P0.3	P0.4	P0.5
PB.1	PB.2	PB.3	PB.4
P9.4	P9.5	P9.6	P9.7

SPI3（MOSI和MISO可切换）

SS3	MOSI3	MISO3	SCLK3
P2.2	P2.3	P2.4	P2.5
P7.0	P7.1	P7.2	P7.3
P8.4	P8.5	P8.6	P8.7
P9.0	P9.1	P9.2	P9.3

I2S（第1个I2S模块）

MCLK	BCK	LRCK	Data
P3.3	P3.2	P3.5	P3.4
P6.7	P6.6	P6.5	P6.4
P2.4	P2.5	P2.2	P2.3
P1.6	P4.3	P4.1	P4.0

I2S2（第2个I2S模块）

MCLK2	BCK2	LRCK2	Data2
P5.2	P5.3	P5.0	P5.1
P6.2	P6.3	P6.0	P6.1
P8.2	P8.3	P8.0	P8.1
PB.3	PB.4	PB.1	PB.2

USART4

RxD4	TxD4
P0.2	P0.3
P5.2	P5.3
PA.2	PA.3
PC.0	PC.1

USART4_SPI（MOSI和MISO可切换）

U4SS	U4MOSI	U4MISO	U4SCLK
P6.4	P6.5	P6.6	P6.7
P0.2	P0.3	P0.4	P0.5
PB.1	PB.2	PB.3	PB.4
P7.0	P7.1	P7.2	P7.3

PWMA（可互补对称带死区输出，每路均可独立选择）

PWMA1P	PWMA1N	PWMA2P	PWMA2N	PWMA3P	PWMA3N	PWMA4P	PWMA4N	PWMA5	PWMA6	PWMAETR	PWMAFLT
P1.0	P1.1	P1.2	P1.3	P1.4	P1.5	P1.6	P1.7	P4.6	P4.7	P3.2	P3.5
P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7	P0.4	P0.5	P4.1	CMPAO
P6.0	P6.1	P6.2	P6.3	P6.4	P6.5	P6.6	P6.7	P7.2	P7.3	P7.3	P4.5
-	-	-	-	-	-	P3.4	P3.3	P1.3	P1.5	-	-

USART5

RxD5	TxD5
P0.4	P0.5
P4.0	P4.1
P1.4	P1.5
P2.0	P2.1

USART5_SPI（MOSI和MISO可切换）

U5SS	U5MOSI	U5MISO	U5SCLK
P2.2	P2.3	P2.4	P2.5
P7.0	P7.1	P7.2	P7.3
P8.4	P8.5	P8.6	P8.7
P9.0	P9.1	P9.2	P9.3

PWMC（可互补对称带死区输出，每路均可独立选择）

PWMC1P	PWMC1N	PWMC2P	PWMC2N	PWMC3P	PWMC3N	PWMC4P	PWMC4N	PWMC5	PWMC6	PWMCETR	PWMCFLT
P6.0	P6.1	P6.2	P6.3	P6.4	P6.5	P6.6	P6.7	P1.1	P1.7	P3.2	P3.5
P8.0	P8.1	P8.2	P8.3	P8.4	P8.5	P8.6	P8.7	P3.6	P3.7	P7.0	CMPCO
P4.0	P4.1	P4.2	P4.3	P4.4	P4.5	P4.6	P4.7	P9.6	P9.7	P9.4	P7.1
PB.0	PB.1	PB.2	PB.3	PB.4	PB.5	PB.6	PB.7	P6.3	P6.5	-	P9.5

USART6

RxD6	TxD6
P0.6	P0.7
P6.2	P6.3
P3.2	P3.3
P2.6	P2.7

USART6_SPI（MOSI和MISO可切换）

U6SS	U6MOSI	U6MISO	U6SCLK
P2.2	P2.3	P2.4	P2.5
P7.0	P7.1	P7.2	P7.3
P8.4	P8.5	P8.6	P8.7
P9.4	P9.5	P9.6	P9.7

PWME（可互补对称带死区输出，每路均可独立选择）

PWME1P	PWME1N	PWME2P	PWME2N	PWME3P	PWME3N	PWME4P	PWME4N	PWME5	PWME6	PWMEETR	PWMEFLT
P7.0	P7.1	P7.2	P7.3	P7.4	P7.5	P7.6	P7.7	P5.0	P5.1	P3.2	P3.5
P9.0	P9.1	P9.2	P9.3	P9.4	P9.5	P9.6	P9.7	P4.3	P4.4	PC.2	CMPEO
PA.0	PA.1	PA.2	PA.3	PA.4	PA.5	PA.6	PA.7	PB.4	PB.5	P0.2	PC.3
-	-	-	-	-	-	-	-	P9.5	P9.7	-	P0.3

USART7

RxD7	TxD7
P5.0	P5.1
PA.4	PA.5
P9.0	P9.1
P1.2	P1.3

USART7_SPI（MOSI和MISO可切换）

U7SS	U7MOSI	U7MISO	U7SCLK
P5.0	P5.1	P5.2	P5.3
PA.4	PA.5	PA.6	PA.7
P7.4	P7.5	P7.6	P7.7
PC.0	PC.1	PC.2	PC.3

PWMB（单端输出，每路均可独立选择）

PWMB5	PWMB6	PWMB7	PWMB8	PWMBETR	PWMBFLT
P2.0	P2.1	P2.2	P2.3	P3.2	P3.5
P1.7	P5.4	P3.3	P3.4	P0.6	CMPBO
P0.0	P0.1	P0.2	P0.3	-	P0.7
P7.4	P7.5	P7.6	P7.7	-	-

CMP（第1个比较器）

CMP1+	CMP1-	CMP1O
P3.7	P3.6	P3.4
P5.0	P5.1	P4.1
OP1O	OP2O	-
ADCIN	1.19V	-
DAC1O	DAC1O	-
DAC2O	DAC2O	-

CMP2（第2个比较器）

CMP2+	CMP2-	CMP2O
P6.7	P6.6	P4.7
P1.0	P1.1	P4.0
OP2O	OP3O	-
ADCIN	1.19V	-
DAC1O	DAC1O	-
DAC2O	DAC2O	-

USART8

RxD8	TxD8
P5.2	P5.3
PA.6	PA.7
P9.2	P9.3
P6.0	P6.1

USART8_SPI（MOSI和MISO可切换）

U8SS	U8MOSI	U8MISO	U8SCLK
P5.0	P5.1	P5.2	P5.3
PA.4	PA.5	PA.6	PA.7
P7.4	P7.5	P7.6	P7.7
PA.0	PA.1	P2.2	P2.3

PWMD（单端输出，每路均可独立选择）

PWMD5	PWMD6	PWMD7	PWMD8	PWMDETR	PWMDFLT
P2.4	P2.5	P2.6	P2.7	P3.2	P3.5
P8.1	P8.3	P8.5	P8.7	P7.0	CMPDO
P5.0	P5.1	P5.2	P5.3	PC.0	P7.1
PB.4	PB.5	PB.6	PB.7	-	PC.1

CMP3（第3个比较器）

CMP3+	CMP3-	CMP3O
P7.7	P7.6	P2.4
P5.3	P5.2	P4.5
OP3O	OP4O	-
ADCBIN	1.19V	-
DAC1O	DAC1O	-
DAC2O	DAC2O	-

CMP4（第4个比较器）

CMP4+	CMP4-	CMP4O
P2.7	P2.6	P2.5
P4.3	P4.4	P4.6
OP4O	OP1O	-
ADCBIN	1.19V	-
DAC1O	DAC1O	-
DAC2O	DAC2O	-

CAN_FD

CANRx	CANTx	CANSTB
P0.0	P0.1	P0.6
P5.0	P5.1	P2.2
P4.2	P4.5	P5.1
P7.0	P7.1	P4.5
P3.0	P3.1	P3.3
P3.6	P3.7	P1.2
P1.4	P1.5	P3.3
P4.3	P4.4	P7.4

CAN_FD2

CAN2Rx	CAN2Tx	CAN2STB
P0.2	P0.3	P0.7
P5.2	P5.3	P2.3
P4.6	P4.7	P5.3
P7.2	P7.3	P4.7
P1.0	P1.1	P0.7
P8.4	P8.5	P8.3
PA.0	PA.1	P2.3
-	-	-

PWMF（单端输出，每路均可独立选择）

PWMF5	PWMF6	PWMF7	PWMF8	PWMFETR	PWMFFLT
P7.0	P7.1	P7.2	P7.3	P3.2	P3.5
PA.1	PA.3	PA.5	PA.7	PB.1	CMPFO
P9.0	P9.1	P9.2	P9.3	P0.0	PB.3
-	-	-	-	-	P0.1

比较器的正极输入端CMPn+和负极输入端CMPn-均可独立选择

OPA1（输入可独立选择）

OP1+	OP1-	OP1O
P0.4	P0.5	P0.7
P0.5	P0.4	P0.0
P6.0	P6.1	P7.4
DAC1O	DAC1O	P6.3

OPA2（输入可独立选择）

OP2+	OP2-	OP2O
P2.6	P2.7	P4.2
P8.0	P8.1	P0.0
P7.5	P7.6	P7.4
P7.7	DAC2O	P6.2

OPA3（输入可独立选择）

OP3+	OP3-	OP3O
P3.5	P5.1	P3.2
P2.0	P4.2	P2.1
P7.0	P7.1	P7.4
DAC2O	DAC2O	P5.0

OPA4（输入可独立选择）

OP4+	OP4-	OP4O
P1.6	P1.7	P1.3
P6.3	P6.5	P4.0
P6.4	P3.3	P7.4
DAC2O	DAC1O	P6.3

LIN

LINRx	LINTx
P0.2	P0.3
P5.2	P5.3
P4.6	P4.7
P7.2	P7.3

LIN2

LIN2Rx	LIN2Tx
P0.6	P0.7
P2.6	P2.7
P6.2	P6.3
P8.4	P8.5

DAC不能直接输出到I/O，必须经过OPA放大后才能输出到I/O

DAC1

输出
DAC1O

DAC2

输出
DAC2O

ADC（第1个ADC模块）

ADC0	ADC1	ADC2	ADC3	ADC4	ADC5	ADC6	ADC7
P1.0	P1.1	P1.2	P1.3	P1.4	P1.5	P1.6	P1.7
ADC8	ADC9	ADC10	ADC11	ADC12	ADC13	ADC14	ADC15
P0.0	P0.1	P0.2	OP1O	OP2O	OP3O	OP4O	1.19V

ADCB（第2个ADC模块）

ADCB0	ADCB1	ADCB2	ADCB3	ADCB4	ADCB5	ADCB6	ADCB7
P0.0	P0.1	P0.2	P0.3	P0.4	P0.5	P0.6	P0.7
ADCB8	ADCB9	ADCB10	ADCB11	ADCB12	ADCB13	ADCB14	ADCB15
P6.0	P6.1	P6.2	OP1O	OP2O	OP3O	OP4O	1.19V

TFT_i8080/M6800控制脚

RS	WR	RD
P4.5	P4.2	P4.4
P4.5	P3.6	P3.7
P4.0	P4.2	P4.4
P4.0	P3.6	P3.7

TFT_i8080/M6800 8位数据

D0	D1	D2	D3	D4	D5	D6	D7
P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
P6.0	P6.1	P6.2	P6.3	P6.4	P6.5	P6.6	P6.7
P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
P6.0	P6.1	P6.2	P6.3	P6.4	P6.5	P6.6	P6.7

TFT_i8080/M6800控制脚

RS	WR	RD
P4.5	P4.2	P4.4
P4.5	P3.6	P3.7
P4.0	P4.2	P4.4
P4.0	P3.6	P3.7

TFT_i8080/M6800 16位数据

D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15
P0.0	P0.1	P0.2	P0.3	P0.4	P0.5	P0.6	P0.7	P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7	P6.0	P6.1	P6.2	P6.3	P6.4	P6.5	P6.6	P6.7
P4.1	P4.3	P4.6	P4.7	P0.4	P0.5	P0.6	P0.7	P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
P7.0	P7.1	P7.2	P7.3	P7.4	P7.5	P7.6	P7.7	P6.0	P6.1	P6.2	P6.3	P6.4	P6.5	P6.6	P6.7

说明：

1、ADC为第一个ADC模块，ADCB为第二个ADC模块，每个ADC模块都有16个通道,其中11个通道的输入可以选择外部I/O，另外5个通道的输入可以选择内部的模拟信号。

2、CMP1/CMP2/CMP3/CMP4为4个比较器模块，每个比较器都是6P6N模式，都支持6个正极输入，6个负极输入。

3、OPA1/OPA2/OPA3/OPA4为4个运算放大器，运算放大器的输入信号可以从外部I/O输入，也可选择内部的模拟信号。OP1O/OP2O/OP3O/OP4O分别为4个运放的输出信号。

4、STC32G144K246芯片内部集成了两个数模转换模块（DAC1/DAC2），DAC1O/DAC2O分别为2个DAC的输出。

由于DAC的输出信号驱动能力弱，不能直接输出到外部管脚，芯片内部将DAC的输出直接连接到OPA（运算放大器）的输入端，驱动能力经过放大后再输出到外部管脚。

5、1.19V信号为内部BandGap电压放大后的电压，是辅助固定参考信号源，不能直接作为ADC或者DAC的参考电压。

6、ADC、PWM、比较器、运算放大器的输入和输出端均可独立选择，其他的外设USART、SPI、QSPI、I2C、I2S、CAN-FD、LIN、TFT，均为整组（整行）选择。

3.1.4 管脚图，最小系统（LQFP64/QFN64）

自带硬件USB，支持硬件USB 直接 下载 和 仿真

为优先保证ADC的精度及其它模拟 电路，

就希望 Gnd 不要流过大的电流，建议：
如果要 I/O 驱动大电流，如 LED灯：
用 强推挽的 高 输出电流，这个只影响Vcc：
不要用 I/O 置低，如置低流入电流导通，
MCU-Gnd 会被流入的大电流抬高 ，
会造成测量误差，建议 ADC转换时，I/O 不驱动大电流。

经过I/O口流进和流出的电流不要超过60mA。
如果要用I/O口直接点亮发光二极管，
请将I/O口工作在强推挽模式，高点亮。
P3.0/P3.1不要接其它电路，防干扰下载。
P3.2建议做按键使用，或输出到高阻输入的电路使用，
USB型MCU，P3.2会参与下载判断，
P3.2不要上电时被固定拉低/否则会等待0.5s判断USB下载。



【重要说明】：

由于封装打线失误，
批号为2534A670239.YZ 和 2544A670239.YZ的
STC32G144K246-LQFP64：

- **Pin8不是Core_Vcc，而是P6.3；**
- **Pin41不是Vcc，而是P4.1；**

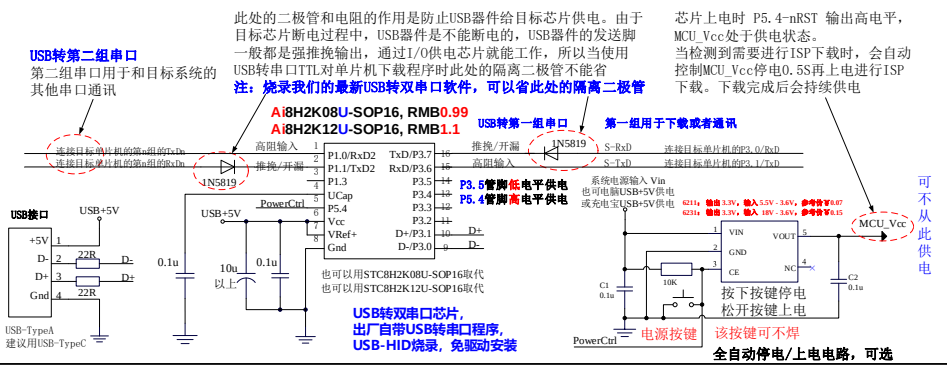
2025年12月2日之后封装的芯片会将
Pin8封装成Core_Vcc，Pin41封装成Vcc。

【注意】：

1、P1.2和P5.4封装时短接在一起，打线都打到第18Pin管脚
特别内部数字设计：
P5.4 也可以通过寄存器P_SWX1特别设置切换成 P1.2的数字功能，
但模拟功能 ADC2无法切换，后决定封装时将P1.2和P5.4封装时短接
2、P1.3和P1.7封装时短接在一起，打线都打到第15Pin脚
3、ADC_VRef+管脚接 ADC的参考源，或直接接 MCU_Vcc
4、ADC_AGnd/ADC_VRef-管脚接模拟地
5、模拟地，数字地，最后在系统地处短接在一起
6、Core-Vcc：内部 1.5V-LDO，外部接 10uF ~ 22uF 电容 即可

22u电容，封装0603，参考价<RMB￥0.030
10u电容，封装0603，参考价<RMB￥0.018
1u电容，封装0603，参考价<RMB￥0.008
0.1u电容，封装0603，参考价<RMB￥0.005

还可以通过【 USB转双串口 芯片或工具】 进行 全自动 停电/上电 下载，支持仿真+串口通讯，3.3V



【应用场景一：从本工具给目标系统 自动 停电/上电，供电】

点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0.5秒/再自动供电，
数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0.5秒/再自动供电给目标系统工作。

【应用场景二：不从本工具给目标系统供电】

1、点击 电脑端 ISP 软件的【下载/编程】按钮
2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，
下载编程进行中，数秒后提示下载编程成功，目标MCU会自动复位到用户程序区自动跑用户程序。
部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P5.4/RST变成复位脚

1、CPU，DSP 可工作在 100MHz 附近
2、TFPU，PWM 可工作在 250MHz 及以下
3、144K SRAM，246K Flash

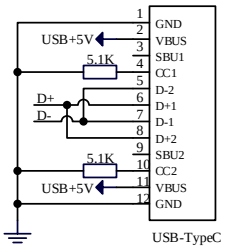
现在带硬件 USB 的 MCU 支持用硬件 USB 下载，因为用的是 USB-HID 通信协议，不需要安装任何驱动。只要 USB 鼠标、USB 键盘能工作，USB-HID 驱动就是好的，不要安装 USB-HID 驱动，免驱。

在 D-/P3.0，D+/P3.1 与 PC-USB 端口连接好的状况下，USB-ISP 下载程序有如下三种模式：

【USB 下载方法一，P3.2 按键，再结合停电上电下载】

1、按下板子上的 P3.2/INT0 按键，就是 P3.2 接地
2、给目标芯片重新上电，不管之前是否已通电。
   ===电子开关是按下停电后再松开就是上电
   等待电脑端 ISP 下载软件中自动识别出“(HID1) USB Writer”后，就与 P3.2 状态无关了，这时可以松开 P3.2 按键（P3.2 在用户程序区可做其它任意用途）
   ===传统的机械自锁紧开关是按上来停电，按下去是上电
3、点击电脑端下载软件中的【下载/编程】按钮（注意：USB 下载与串口下载的操作顺序不同）
   下载进行中，几秒钟后，提示下载成功！

比较一下流行的 USB-TypeC接口：



【USB 下载方法二，复位管脚低电平复位下载】

USB 连接好并已上电的情况下，外部按键复位也可进入 USB 下载模式，注意：
P5.4-RST 出厂时默认是 P5.4-I/O 功能，要改为复位功能，需 ISP 烧录时取消设置复位脚用作 I/O 口，停电一次再上电才生效，程序区中用户程序也可改为复位脚或 I/O，这个立即生效。

1、按下 P5.4-RST 外接的低电平复位按键复位 MCU，
   松开复位键，MCU 从系统程序区启动，判断是否要下载用户程序，
   等待电脑端 ISP 下载软件中自动识别出“(HID1) USB Writer”后
2、点击电脑端下载软件中的【下载/编程】按钮
   下载进行中，几秒钟后，提示下载成功！

备注：如果目标 MCU 的用户程序区已有 USB 通讯程序（含 CDC、HID 等），则通过复位脚复位进入 USB 下载模式不被支持，只能通过停电上电冷启动下载或者从用户程序区软复位到系统区下载

【USB 下载方法三，从用户程序区软复位到系统区下载】

USB 连接好并已上电的情况下，从用户程序区软复位到系统区也可进入 USB 下载模式

1、在用户程序区运行软复位到系统区的程序，就是 IAP_CONTR 寄存器送 60H
   等待电脑端 ISP 下载软件中自动识别出“(HID1) USB Writer”后
2、点击电脑端下载软件中的【下载/编程】按钮
   下载进行中，几秒钟后，提示下载成功！

USB 下载 注意事项：

拔插 USB 插头不能代替上面线路图中的电源开关。正常操作步骤如下：

USB 的【Gnd，D+，D-】接好的情况下，按下 P3.2 按键接地，再通过正常的电源开关给 MCU 供电或重新供电，让 MCU 冷启动进入系统程序区，判断是否需要等待电脑端 USB 下载程序。

拔插 USB 插头代替电源开关不能每次都能成功进行 USB 下载的原因：

拔插 USB 插头，如【Gnd，USB+5V】已接触好，已供电，而【D+，D-】有一个甚至两个信号线还没有接触好，MCU 已上电，开始跑系统区程序时，发现 USB 还没接触好，则会从系统区软复位到用户程序区跑用户程序，不再进入等待 USB 下载模式，本次就无法顺利进行 USB 下载。

很多人经过多次插拔 USB，才能碰到 1 次【D+，D-】接触好的情况下，【Gnd，USB+5V】才开始接触好，才开始供电，才能成功进入 USB 下载。插 USB 插头代替供电，不能保证【Gnd，D+，D-，USB+5V】的接触顺序，所以，必须使用正常的电源开关，才能确保每次下载都能成功。

3.1.5 管脚图，最小系统（LQFP48）

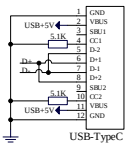
自带硬件USB，支持硬件USB 直接 下载 和 仿真

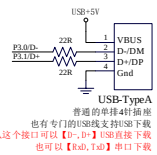
为优先保证ADC的精度及其它模拟电路，

就希望 Gnd 不要流过大的电流，建议：
如果要 I/O 驱动大电流，如 LED灯：
用 强推挽的 高 输出电流，这个只影响Vcc；
不要用 I/O 置低，如置低流入电流导通，
MCU-Gnd 会被流入的大电流抬高 ，
会造成测量误差，建议 ADC转换时，I/O 不驱动大电流。

经过I/O口流进和流出的电流不要超过60mA。
如果要用I/O口直接点亮发光二极管，
请将I/O口工作在强推挽模式，高点亮。
P3.0/P3.1不要接其它电路，防干扰下载。
P3.2建议做按键使用，或输出到高阻输入的电路使用，
USB型MCU，P3.2会参与下载判断，
P3.2不要上电时被固定拉低/否则会等待0.5s判断USB下载。

USB-TypeA 和 USB-TypeC 只需一个接口





USB直接下载：
在单片机【P3.0/D-，P3.1/D+】和电脑的USB口连接好的情况下，
将P3.2按键按下（即P3.2 = 0），再给单片机上电，如已上电，
需停电再给MCU重新上电，则MCU会进入【等待USB下载模式】，
点击电脑端ISP下载软件中的“下载”按钮即可下载。



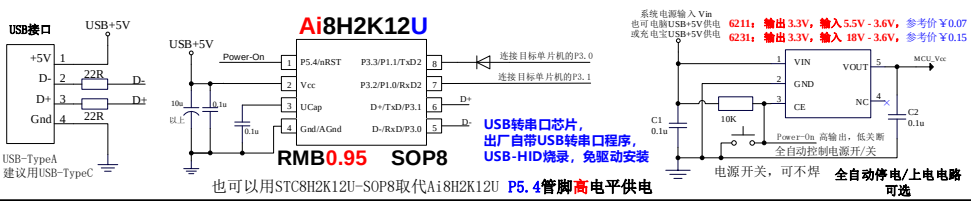
UCAP 是内部USB模块的 **3.3V-LDO**，外接 0.1uF 去耦电容
1、USB 不用就不用接外部电容
2、MCU_Vcc 工作在 3.6V以上，UCAP管脚可挂 0.1uF 去耦电容；
3、MCU_Vcc 工作在 3.6V以下，UCAP管脚直接短接到 MCU_Vcc；
4、或不管任何情况，UCAP管脚统一外挂 0.1uF 去耦电容

P5.4/nRST出厂时默认是P5.4-I/O功能，要改为复位功能，
需ISP烧录时取消设置复位脚用作I/O口，
烧录后，停电一次再上电，新的设置才生效。
程序区中用户程序也可改为复位脚或I/O，这个立即生效。

22u电容，封装0603，参考价<RMB￥0.030
10u电容，封装0603，参考价<RMB￥0.018
1u电容，封装0603，参考价<RMB￥0.008
0.1u电容，封装0603，参考价<RMB￥0.005

【注意】：
1、P1.2和P5.4封装时短接在一起，打线都打到第14Pin管脚
特别内部数字设计：
P5.4 也可以通过寄存器P_SWX1特别设置切换成 P1.2的数字功能，
但模拟功能 ADC2无法切换，后决定封装时将P1.2和P5.4封装时短接
2、P1. 3和P1. 7封装时短接在一起，打线都打到第11Pin脚
3、ADC_VRef+管脚接 ADC的参考源，或直接接 MCU_Vcc
4、ADC_AGnd/ADC_VRef-管脚接模拟地
5、模拟地，数字地，最后在系统地处短接在一起
6、Core-Vcc：内部 1.5V-LDO，外部接 10uF ~ 22uF 电容 即可

还可以通过【 USB转串口 芯片或工具】，进行 ISP烧录/仿真/通信，目标系统自己手动停电/上电



也可以用STC8H2K12U-SOP8取代Ai8H2K12U P5.4管脚高电平供电

【应用场景一：从本工具给目标系统 自动 停电/上电，供电】
点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0.5秒/再自动供电，
数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0.5秒/再自动供电给目标系统工作。

【应用场景二：不从本工具给目标系统供电】
1、点击 电脑端 ISP 软件的【下载/编程】按钮
2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，
下载编程进行中，数秒后提示下载编程成功，目标MCU会自动复位到用户程序区自动跑用户程序。
部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P5.4/nRST变成复位脚

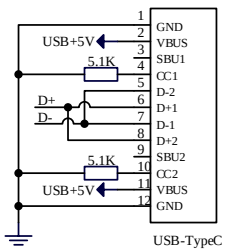
1、CPU，DSP 可工作在 100MHz 附近
2、TFPU，PWM 可工作在 250MHz 及以下
3、144K SRAM，246K Flash

尽量选 LQFP100、LQFP64，尽量不选 LQFP48，减轻库存及生产压力

现在带硬件 USB 的 MCU 支持用硬件 USB 下载，因为用的是 USB-HID 通信协议，不需要安装任何驱动。只要 USB 鼠标、USB 键盘能工作，USB-HID 驱动就是好的，不要安装 USB-HID 驱动，免驱。
在 D-/P3.0，D+/P3.1 与 PC-USB 端口连接好的状况下，USB-ISP 下载程序有如下三种模式：

【USB 下载方法一，P3.2 按键，再结合停电上电下载】
1、按下板子上的 P3.2/INT0 按键，就是 P3.2 接地
2、给目标芯片重新上电，不管之前是否已通电。
===电子开关是按下停电后再松开就是上电
等待电脑端 ISP 下载软件中自动识别出“(HID1) USB Writer”后，就与 P3.2 状态无关了，这时可以松开 P3.2 按键（P3.2 在用户程序区可做其它任意用途）
===传统的机械自锁紧开关是按上来停电，按下去是上电
3、点击电脑端下载软件中的【下载/编程】按钮（注意：USB 下载与串口下载的操作顺序不同）
下载进行中，几秒钟后，提示下载成功！

比较一下流行的 USB-TypeC接口：



【USB 下载方法二，复位管脚低电平复位下载】
USB 连接好并已上电的情况下，外部按键复位也可进入 USB 下载模式，注意：
P5.4-RST 出厂时默认是 P5.4-I/O 功能，要改为复位功能，需 ISP 烧录时取消设置复位脚用作 I/O 口，停电一次再上电才生效，程序区中用户程序也可改为复位脚或 I/O，这个立即生效。
1、按下 P5.4-RST 外接的低电平复位按键复位 MCU，
松开复位键，MCU 从系统程序区启动，判断是否要下载用户程序，
等待电脑端 ISP 下载软件中自动识别出“(HID1) USB Writer”后
2、点击电脑端下载软件中的【下载/编程】按钮
下载进行中，几秒钟后，提示下载成功！
备注：如果目标 MCU 的用户程序区已有 USB 通讯程序（含 CDC、HID 等），则通过复位脚复位进入 USB 下载模式不被支持，只能通过停电上电冷启动下载或者从用户程序区软复位到系统区下载

【USB 下载方法三，从用户程序区软复位到系统区下载】
USB 连接好并已上电的情况下，从用户程序区软复位到系统区也可进入 USB 下载模式
1、在用户程序区运行软复位到系统区的程序，就是 IAP_CONTR 寄存器送 60H
等待电脑端 ISP 下载软件中自动识别出“(HID1) USB Writer”后
2、点击电脑端下载软件中的【下载/编程】按钮，下载进行中，几秒钟后，提示下载成功！

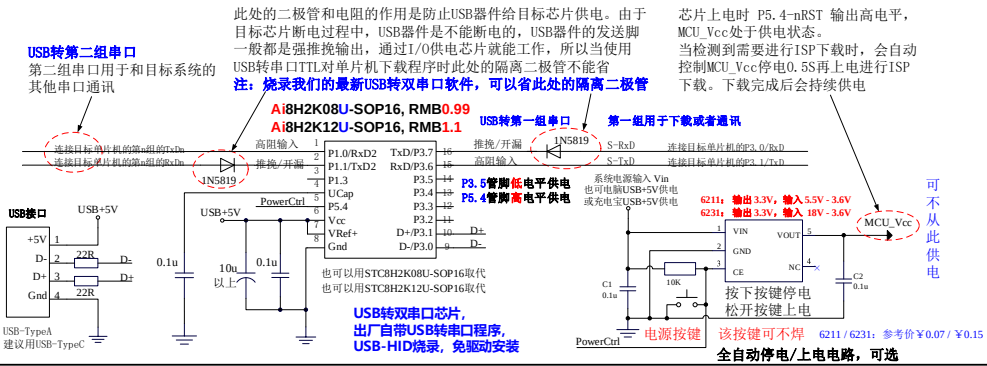
USB 下载 注意事项：
拔插 USB 插头不能代替上面线路图中的电源开关。正常操作步骤如下：
USB 的【Gnd，D+，D-】接好的情况下，按下 P3.2 按键接地，再通过正常的电源开关给 MCU 供电或重新供电，让 MCU 冷启动进入系统程序区，判断是否需要等待电脑端 USB 下载程序。
拔插 USB 插头代替电源开关不能每次都能成功进行 USB 下载的原因：
拔插 USB 插头，如【Gnd，USB+5V】已接触好，已供电，而【D+，D-】有一个甚至两个信号线还没有接触好，MCU 已上电，开始跑系统区程序时，发现 USB 还没接触好，则会从系统区软复位到用户程序区跑用户程序，不再进入等待 USB 下载模式，本次就无法顺利进行 USB 下载。
很多人经过多次插拔 USB，才能碰到 1 次【D+，D-】接触好的情况下，【Gnd，USB+5V】才开始接触好，才开始供电，才能成功进入 USB 下载。插 USB 插头代替供电，不能保证【Gnd，D+，D-，USB+5V】的接触顺序，所以，必须使用正常的电源开关，才能确保每次下载都能成功。

3.1.6 管脚图，最小系统（LQFP44）

自带硬件USB，支持硬件USB 直接 下载 和 仿真



还可以通过【 USB转双串口 芯片或工具】 进行 全自动 停电/上电 下载，支持仿真+串口通讯，3.3V



【应用场景一：从本工具给目标系统 自动 停电/上电，供电】
点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0. 5秒/再自动供电，
数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0. 5秒/再自动供电给目标系统工作。

【应用场景二：不从本工具给目标系统供电】
1、点击 电脑端 ISP 软件的【下载/编程】按钮
2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，
下载编程进行中，数秒后提示下载编程成功，目标MCU会自动复位到用户程序区自动跑用户程序。
部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P5. 4/RST变成复位脚

1、CPU，DSP 可工作在 100MHz 附近
2、TFPU，PWM 可工作在 250MHz 及以下
3、144K SRAM，246K Flash

尽量选 LQFP100、LQFP64，尽量不选 LQFP44，减轻库存及生产压力

备战 2025 年 全国大学生智能汽车竞赛, 全国大学生电子设计竞赛
大家先看下管脚图合不合理，10 月底流片，12 月回来，新年送样
帮设计强大的 STC32G144K246-LQFP100 实验箱

管脚兼容 8H8K64U, 32G12K128, 32G8K64, 32F12K54 系列

3.1.7 不同封装形式管脚对应功能说明

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
1	1	1	1	P5.3	I/O	标准 I/O 口
				TXD4_2	O	串口 4 的发送脚
				TxD8	O	串口 8 的发送脚
				S7SCLK	I/O	USART7－SPI 的时钟脚
				S8SCLK	I/O	USART8－SPI 的时钟脚
				PWMD8_3	I/O	PWMD 的捕获输入和脉冲输出
				CMP3+_2	I	比较器 3 正极输入
				LIN_TX_2	O	LIN 总线发送脚
				CAN2_TX_2	O	CAN2 总线发送脚
				CAN2_STB_3	O	CAN2 低功耗控制脚
				I2S2_BCK	I/O	I2S2 的位时钟线
2	2	2	1	P0.5	I/O	标准 I/O 口
				AD5	I/O	地址/数据总线
				ADCB5	I	ADCB 模拟输入通道 5
				T3CLKO	O	定时器 3 时钟分频输出
				S3SCLK_2	I/O	USART3－SPI 的时钟脚
				S4SCLK_2	I/O	USART4－SPI 的时钟脚
				TxD5	O	串口 5 的发送脚
				SCLK2_2	I/O	SPI2 的时钟脚
				PWMA6_2	O	PWMA 的硬件移相输出
				OPA1+_2	I	PGA1 的正极输入
				OPA1-	I	PGA1 的负极输入
3	3	3	2	P0.6	I/O	标准 I/O 口
				AD6	I/O	地址/数据总线
				ADCB6	I	ADCB 模拟输入通道 6
				T4	I	定时器 4 外部时钟输入
				RxD6	I	串口 6 的接收脚
				PWMBETR_2	I	PWMB 外部触发输入脚
				LIN2_RX	I	LIN2 总线接收脚
				CAN_STB	O	CAN 低功耗控制脚
4	4	4	3	P0.7	I/O	标准 I/O 口
				AD7	I/O	地址/数据总线
				ADCB7	I	ADCB 模拟输入通道 7
				T4CLKO	O	定时器 4 时钟分频输出
				TxD6	O	串口 6 的发送脚
				PWMBFLT_2	I	PWMB 外部异常检测脚
				OPA1O	O	PGA1 的输出
				LIN2_TX	O	LIN2 总线发送脚
				CAN2_STB	O	CAN2 低功耗控制脚
				CAN2_STB_5	O	CAN2 低功耗控制脚

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
5	5			P6.0	I/O	标准 I/O 口
				ADCB8	I	ADCB 模拟输入通道 8
				T10	I	定时器 10 外部时钟输入
				RxD8_4	I	串口 8 的接收脚
				S8SS_4	I/O	USART8－SPI 的从机选择脚（主机为输出）
				I2CSDA2_4	I/O	I2C2 的数据线
				PWMA1P_3	I/O	PWMA 的捕获输入和脉冲输出正极
				PWMC1P	I/O	PWMC 的捕获输入和脉冲输出正极
				OPA1+_3	I	PGA1 的正极输入
				I2S2_LRCK_2	I/O	I2S2 的左右声道时钟线
6	6			P6.1	I/O	标准 I/O 口
				ADCB9	I	ADCB 模拟输入通道 9
				T10CLKO	O	定时器 10 时钟分频输出
				TxD8_4	O	串口 8 的发送脚
				S8MOSI_4	I/O	USART8－SPI 主出从入
				I2CSCL2_4	I/O	I2C2 的时钟线
				PWMA1N_3	O	PWMA 的脉冲输出负极
				PWMC1N	O	PWMC 的脉冲输出负极
				OPA1-_3	I	PGA1 的负极输入
				I2S2_Data_2	I/O	I2S2 的数据线
7	7			P6.2	I/O	标准 I/O 口
				ADCB10	I	ADCB 模拟输入通道 10
				RxD6_2	I	串口 6 的接收脚
				S8MISO_4	I/O	USART8－SPI 的主入从出
				PWMA2P_3	I/O	PWMA 的捕获输入和脉冲输出正极
				PWMC2P	I/O	PWMC 的捕获输入和脉冲输出正极
				OPA2O_4	O	PGA2 的输出
				LIN2_RX_3	I	LIN2 总线接收脚
				I2S2_MCLK_2	I/O	I2S2 的主时钟线
8				P6.3	I/O	标准 I/O 口
				TxD6_2	O	串口 6 的发送脚
				S8SCLK_4	I/O	USART8－SPI 的时钟脚
				PWMA2N_3	O	PWMA 的脉冲输出负极
				PWMC2N	O	PWMC 的脉冲输出负极
				PWMC5_4	O	PWMC 的硬件移相输出
				OPA1O_4	O	PGA1 的输出
				OPA4+_2	I	PGA4 的正极输入
				OPA4O_4	O	PGA4 的输出
				LIN2_TX_3	O	LIN2 总线发送脚
				I2S2_BCK_2	I/O	I2S2 的位时钟线

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
9				P8.0	I/O	标准 I/O 口
				T17	I	定时器 17 外部时钟输入
				SS2_4	I/O	SPI2 的从机选择脚（主机为输出）
				PWMC1P_2	I/O	PWMC 的捕获输入和脉冲输出正极
				OPA2+_2	I	PGA2 的正极输入
				I2S2_LRCK_3	I/O	I2S2 的左右声道时钟线
10				P8.1	I/O	标准 I/O 口
				T17CLKO	O	定时器 17 时钟分频输出
				MOSI2_4	I/O	SPI2 主机输出从机输入
				PWMC1N_2	O	PWMC 的脉冲输出负极
				PWMD5_2	I/O	PWMD 的捕获输入和脉冲输出
				OPA2-_2	I	PGA2 的负极输入
				I2S2_Data_3	I/O	I2S2 的数据线
11				P8.2	I/O	标准 I/O 口
				MISO2_4	I/O	SPI2 主机输入从机输出
				PWMC2P_2	I/O	PWMC 的捕获输入和脉冲输出正极
				I2S2_MCLK_3	I/O	I2S2 的主时钟线
12				P8.3	I/O	标准 I/O 口
				SCLK2_4	I/O	SPI2 的时钟脚
				PWMC2N_2	O	PWMC 的脉冲输出负极
				PWMD6_2	I/O	PWMD 的捕获输入和脉冲输出
				CAN2_STB_6	O	CAN2 低功耗控制脚
				I2S2_BCK_3	I/O	I2S2 的位时钟线
13				P8.4	I/O	标准 I/O 口
				RXD2_3	I	串口 2 的接收脚
				S5SS_3	I/O	USART5－SPI 的从机选择脚（主机为输出）
				S6SS_3	I/O	USART6－SPI 的从机选择脚（主机为输出）
				SS3_3	I/O	SPI3 的从机选择脚（主机为输出）
				PWMC3P_2	I/O	PWMC 的捕获输入和脉冲输出正极
				LIN2_RX_4	I	LIN2 总线接收脚
				CAN2_RX_6	I	CAN2 总线接收脚
14				P8.5	I/O	标准 I/O 口
				TXD2_3	O	串口 2 的发送脚
				S5MOSI_3	I/O	USART5－SPI 主出从入
				S6MOSI_3	I/O	USART6－SPI 主出从入
				MOSI3_3	I/O	SPI3 主机输出从机输入
				PWMC3N_2	O	PWMC 的脉冲输出负极
				PWMD7_2	I/O	PWMD 的捕获输入和脉冲输出
				LIN2_TX_4	O	LIN2 总线发送脚
				CAN2_TX_6	O	CAN2 总线发送脚

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
15				P8.6	I/O	标准 I/O 口
				RXD3_3	I	串口 3 的接收脚
				S5MISO_3	I/O	USART5－SPI 主入从出
				S6MISO_3	I/O	USART6－SPI 主入从出
				MISO3_3	I/O	SPI3 主机输入从机输出
				PWMC4P_2	I/O	PWMC 的捕获输入和脉冲输出正极
16				P8.7	I/O	标准 I/O 口
				TXD3_3	O	串口 3 的发送脚
				S5SCLK_3	I/O	USART5－SPI 的时钟脚
				S6SCLK_3	I/O	USART6－SPI 的时钟脚
				SCLK3_3	I/O	SPI3 的时钟脚
				PWMC4N_2	O	PWMC 的脉冲输出负极
				PWMD8_2	I/O	PWMD 的捕获输入和脉冲输出
17	8	5	4	Core_Vcc	VCC	内核电源脚
18	9			P1.0	I/O	标准 I/O 口
				ADC0	I	ADC 模拟输入通道 0
				RXD2	I	串口 2 的接收脚
				PWMA1P	I/O	PWMA 的捕获输入和脉冲输出正极
				CMP2+_2	I	比较器 2 正极输入
				CAN2_RX_5	I	CAN2 总线接收脚
				QSPI_IO3	I/O	QSPI 数据脚
19	10	6	5	P1.1	I/O	标准 I/O 口
				ADC1	I	ADC 模拟输入通道 1
				TXD2	O	串口 2 的发送脚
				PWMA1N	O	PWMA 的脉冲输出负极
				PWMC5	O	PWMC 的硬件移相输出
				CMP2-_2	I	比较器 2 负极输入
				CAN2_TX_5	O	CAN2 总线发送脚
				QSPI_IO2	I/O	QSPI 数据脚
20	11	7	6	P4.7	I/O	标准 I/O 口
				TXD2_2	O	串口 2 的发送脚
				PWMA6	O	PWMA 的硬件移相输出
				PWMC4N_3	O	PWMC 的脉冲输出负极
				CMP2O	O	比较器 2 输出
				LIN_TX_3	O	LIN 总线发送脚
				CAN2_TX_3	O	CAN2 总线发送脚
				CAN2_STB_4	O	CAN2 低功耗控制脚
				QSPI_IO3_2	I/O	QSPI 数据脚

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
21	12	8	7	P1.4	I/O	标准 I/O 口
				ADC4	I	ADC 模拟输入通道 4
				T11_2	I	定时器 11 外部时钟输入
				S1MISO	I/O	USART1－SPI 主入从出
				S2MISO	I/O	USART2－SPI 主入从出
				RxD5_3	I	串口 5 的接收脚
				MISO	I/O	SPI 主机输入从机输出
				I2CSDA	I/O	I2C 的数据线
				PWMA3P	I/O	PWMA 的捕获输入和脉冲输出正极
				CAN_RX_7	I	CAN 总线接收脚
				QSPI_IO1	I/O	QSPI 数据脚
22	13	9	8	P1.5	I/O	标准 I/O 口
				ADC5	I	ADC 模拟输入通道 5
				T11CKO_2	O	定时器 11 时钟分频输出
				S1SCLK	I/O	USART1－SPI 的时钟脚
				S2SCLK	I/O	USART2－SPI 的时钟脚
				TxD5_3	O	串口 5 的发送脚
				SCLK	I/O	SPI 的时钟脚
				I2CSCL	I/O	I2C 的时钟线
				PWMA3N	O	PWMA 的脉冲输出负极
				PWMA6_4	O	PWMA 的硬件移相输出
				CAN_TX_7	O	CAN 总线发送脚
				QSPI_CLK	O	QSPI 时钟脚
23	14	10	9	P1.6	I/O	标准 I/O 口
				ADC6	I	ADC 模拟输入通道 6
				RXD_3	I	串口 1 的接收脚
				XTALO	O	外部晶振的输出脚
				MCLKO_2	O	主时钟分频输出
				I2CSDA2_3	I/O	I2C2 的数据线
				PWMA4P	I/O	PWMA 的捕获输入和脉冲输出正极
				OPA4+	I	PGA4 的正极输入
				I2S_MCLK_4	I/O	I2S 的主时钟线
24	15	11	10	P1.7	I/O	标准 I/O 口
				ADC7	I	ADC 模拟输入通道 7
				TXD_3	O	串口 1 的发送脚
				XTALI	I	外部晶振/外部时钟的输入脚
				I2CSCL2_3	I/O	I2C2 的时钟线
				PWMA4N	O	PWMA 的脉冲输出负极
				PWMB5_2	I/O	PWMB 的捕获输入和脉冲输出
				PWMC6	O	PWMC 的硬件移相输出
				OPA4-	I	PGA4 的负极输入
				P1.3	I/O	标准 I/O 口

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
				ADC3	I	ADC 模拟输入通道 3
				T2CLKO	O	定时器 2 时钟分频输出
				S1MOSI	I/O	USART1－SPI 主出从入
				S2MOSI	I/O	USART2－SPI 主出从入
				TxD7_4	O	串口 7 的发送脚
				MOSI	I/O	SPI 主机输出从机输入
				PWMA2N	O	PWMA 的脉冲输出负极
				PWMA5_4	O	PWMA 的硬件移相输出
				OPA4O	O	PGA4 的输出
				QSPI_IO0	I/O	QSPI 数据脚
25	16	12	11	ADC_AGnd	GND	ADC 地线
				ADC_VRef-	I	ADC 的参考电压地线
26	17	13	12	UCap	I	USB 内核电源稳压脚
27	18	14	13	P5.4	I/O	标准 I/O 口
				S1SS_3	I/O	USART1－SPI 的从机选择脚（主机为输出）
				S2SS_3	I/O	USART2－SPI 的从机选择脚（主机为输出）
				RST	I	复位引脚
				MCLKO	O	主时钟分频输出
				SS_3	I/O	SPI 的从机选择脚（主机为输出）
				PWMB6_2	I/O	PWMB 的捕获输入和脉冲输出
				QSPI_NCS_2	O	QSPI 片选脚
				P1.2	I/O	标准 I/O 口
				ADC2	I	ADC 模拟输入通道 2
				ADC_ETR	I	ADC 外部触发脚
				ADCB_ETR	I	ADCB 外部触发脚
				T2	I	定时器 2 外部时钟输入
				S1SS	I/O	USART1－SPI 的从机选择脚（主机为输出）
				S2SS	I/O	USART2－SPI 的从机选择脚（主机为输出）
				RxD7_4	I	串口 7 的接收脚
				SS	I/O	SPI 的从机选择脚（主机为输出）
				PWMA2P	I/O	PWMA 的捕获输入和脉冲输出正极
				CAN_STB_6	O	CAN 低功耗控制脚
				QSPI_NCS	O	QSPI 片选脚
28	19	15	14	Vcc	VCC	电源脚
				ADC_AVcc	VCC	ADC 电源脚
29	20	16	15	ADC_VRef+	I	ADC 的参考电压脚
30	21	17	16	Gnd	GND	地线

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
31	22	18	17	P4.0	I/O	标准 I/O 口
				S1MOSI_3	I/O	USART1－SPI 主出从入
				S2MOSI_3	I/O	USART2－SPI 主出从入
				RxD5_2	I	串口 5 的接收脚
				MOSI_3	I/O	SPI 主机输出从机输入
				PWMC1P_3	I/O	PWMC 的捕获输入和脉冲输出正极
				CMP2O_2	O	比较器 2 输出
				OPA4O_2	O	PGA4 的输出
				QSPI_IO0_2	I/O	QSPI 数据脚
				I2S_Data_4	I/O	I2S 的数据线
32	23			P6.4	I/O	标准 I/O 口
				S1SS_4	I/O	USART1－SPI 的从机选择脚（主机为输出）
				S3SS	I/O	USART3－SPI 的从机选择脚（主机为输出）
				S4SS	I/O	USART4－SPI 的从机选择脚（主机为输出）
				SS2	I/O	SPI2 的从机选择脚（主机为输出）
				PWMA3P_3	I/O	PWMA 的捕获输入和脉冲输出正极
				PWMC3P	I/O	PWMC 的捕获输入和脉冲输出正极
				OPA4+_3	I	PGA4 的正极输入
				I2S_Data_2	I/O	I2S 的数据线
33	24			P6.5	I/O	标准 I/O 口
				S1MOSI_4	I/O	USART1－SPI 主出从入
				S3MOSI	I/O	USART3－SPI 主出从入
				S4MOSI	I/O	USART4－SPI 主出从入
				MOSI2	I/O	SPI2 主机输出从机输入
				PWMA3N_3	O	PWMA 的脉冲输出负极
				PWMC3N	O	PWMC 的脉冲输出负极
				PWMC6_4	O	PWMC 的硬件移相输出
				OPA4-_2	I	PGA4 的负极输入
				I2S_LRCK_2	I/O	I2S 的左右声道时钟线
34	25			P6.6	I/O	标准 I/O 口
				S1MISO_4	I/O	USART1－SPI 主入从出
				S3MISO	I/O	USART3－SPI 主入从出
				S4MISO	I/O	USART4－SPI 主入从出
				MISO2	I/O	SPI2 主机输入从机输出
				PWMA4P_3	I/O	PWMA 的捕获输入和脉冲输出正极
				PWMC4P	I/O	PWMC 的捕获输入和脉冲输出正极
				CMP2-	I	比较器 2 负极输入
				I2S_BCK_2	I/O	I2S 的位时钟线

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
35	26			P6.7	I/O	标准 I/O 口
				S1SCLK_4	I/O	USART1－SPI 的时钟脚
				S3SCLK	I/O	USART3－SPI 的时钟脚
				S4SCLK	I/O	USART4－SPI 的时钟脚
				SCLK2	I/O	SPI2 的时钟脚
				PWMA4N_3	O	PWMA 的脉冲输出负极
				PWMC4N	O	PWMC 的脉冲输出负极
				CMP2+	I	比较器 2 正极输入
				I2S_MCLK_2	I/O	I2S 的主时钟线
36				P9.0	I/O	标准 I/O 口
				T18	I	定时器 18 外部时钟输入
				RxD7_3	I	串口 7 的接收脚
				S5SS_4	I/O	USART5－SPI 的从机选择脚（主机为输出）
				SS3_4	I/O	SPI3 的从机选择脚（主机为输出）
				PWME1P_2	I/O	PWME 的捕获输入和脉冲输出正极
				PWMF5_3	I/O	PWMF 的捕获输入和脉冲输出
37				P9.1	I/O	标准 I/O 口
				T18CLKO	O	定时器 18 时钟分频输出
				TxD7_3	O	串口 7 的发送脚
				S5MOSI_4	I/O	USART5－SPI 主出从入
				MOSI3_4	I/O	SPI3 主机输出从机输入
				PWME1N_2	O	PWME 的脉冲输出负极
				PWMF6_3	I/O	PWMF 的捕获输入和脉冲输出
38				P9.2	I/O	标准 I/O 口
				RxD8_3	I	串口 8 的接收脚
				S5MISO_4	I/O	USART5－SPI 主入从出
				MISO3_4	I/O	SPI3 主机输入从机输出
				PWME2P_2	I/O	PWME 的捕获输入和脉冲输出正极
				PWMF7_3	I/O	PWMF 的捕获输入和脉冲输出
39				P9.3	I/O	标准 I/O 口
				TxD8_3	O	串口 8 的发送脚
				S5SCLK_4	I/O	USART5－SPI 的时钟脚
				SCLK3_4	I/O	SPI3 的时钟脚
				PWME2N_2	O	PWME 的脉冲输出负极
				PWMF8_3	I/O	PWMF 的捕获输入和脉冲输出
40				P9.4	I/O	标准 I/O 口
				S3SS_4	I/O	USART3－SPI 的从机选择脚（主机为输出）
				S6SS_4	I/O	USART6－SPI 的从机选择脚（主机为输出）
				PWMCETR_3	I	PWMC 外部触发输入脚
				PWME3P_2	I/O	PWME 的捕获输入和脉冲输出正极

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
41				P9.5	I/O	标准 I/O 口
				S3MOSI_4	I/O	USART3－SPI 主出从入
				S6MOSI_4	I/O	USART6－SPI 主出从入
				PWMCFLT_3	I	PWMC 外部异常检测脚
				PWME3N_2	O	PWME 的脉冲输出负极
				PWME5_4	O	PWME 的硬件移相输出
42				P9.6	I/O	标准 I/O 口
				S3MISO_4	I/O	USART3－SPI 主入从出
				S6MISO_4	I/O	USART6－SPI 主入从出
				PWMC5_3	O	PWMC 的硬件移相输出
				PWME4P_2	I/O	PWME 的捕获输入和脉冲输出正极
43				P9.7	I/O	标准 I/O 口
				S3SCLK_4	I/O	USART3－SPI 的时钟脚
				S6SCLK_4	I/O	USART6－SPI 的时钟脚
				PWMC6_3	O	PWMC 的硬件移相输出
				PWME4N_2	O	PWME 的脉冲输出负极
				PWME6_4	O	PWME 的硬件移相输出
44				Vcc	VCC	电源脚
45	27	19	18	P3.0	I/O	标准 I/O 口
				RXD	I	串口 1 的接收脚
				INT4	I	外部中断 4
				CAN_RX_5	I	CAN 总线接收脚
				D-	I/O	USB 数据口
46	28	20	19	P3.1	I/O	标准 I/O 口
				TXD	O	串口 1 的发送脚
				CAN_TX_5	O	CAN 总线发送脚
				D+	I/O	USB 数据口
47	29	21	20	P3.2	I/O	标准 I/O 口
				RxD6_3	I	串口 6 的接收脚
				INT0	I	外部中断 0
				SCLK_4	I/O	SPI 的时钟脚
				I2CSCL_4	I/O	I2C 的时钟线
				PWMAETR	I	PWMA 外部触发输入脚
				PWMBETR	I	PWMB 外部触发输入脚
				PWMCETR	I	PWMC 外部触发输入脚
				PWMDETR	I	PWMD 外部触发输入脚
				PWMEETR	I	PWME 外部触发输入脚
				PWMFETR	I	PWMF 外部触发输入脚
				OPA3O	O	PGA3 的输出
				I2S_BCK	I/O	I2S 的位时钟线

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
48	30	22	21	P3.3	I/O	标准 I/O 口
				TxD6_3	O	串口 6 的发送脚
				INT1	I	外部中断 1
				MISO_4	I/O	SPI 主机输入从机输出
				I2CSDA_4	I/O	I2C 的数据线
				PWMA4N_4	O	PWMA 的脉冲输出负极
				PWMB7_2	I/O	PWMB 的捕获输入和脉冲输出
				OPA4-_3	I	PGA4 的负极输入
				CAN_STB_5	O	CAN 低功耗控制脚
				CAN_STB_7	O	CAN 低功耗控制脚
				I2S_MCLK	I/O	I2S 的主时钟线
49	31	23	22	P3.4	I/O	标准 I/O 口
				T0	I	定时器 0 外部时钟输入
				T1CLKO	O	定时器 1 时钟分频输出
				MOSI_4	I/O	SPI 主机输出从机输入
				PWMA4P_4	I/O	PWMA 的捕获输入和脉冲输出正极
				PWMB8_2	I/O	PWMB 的捕获输入和脉冲输出
				CMP1O	O	比较器输出
				I2S_Data	I/O	I2S 的数据线
50	32	24	22	P5.0	I/O	标准 I/O 口
				RXD3_2	I	串口 3 的接收脚
				RxD7	I	串口 7 的接收脚
				S7SS	I/O	USART7－SPI 的从机选择脚（主机为输出）
				S8SS	I/O	USART8－SPI 的从机选择脚（主机为输出）
				PWMD5_3	I/O	PWMD 的捕获输入和脉冲输出
				PWME5	O	PWME 的硬件移相输出
				CMP1+_2	I	比较器正极输入
				OPA3O_4	O	PGA3 的输出
				CAN_RX_2	I	CAN 总线接收脚
				I2S2_LRCK	I/O	I2S2 的左右声道时钟线
51	33	25	23	P5.1	I/O	标准 I/O 口
				TXD3_2	O	串口 3 的发送脚
				TxD7	O	串口 7 的发送脚
				S7MOSI	I/O	USART7－SPI 主出从入
				S8MOSI	I/O	USART8－SPI 主出从入
				PWMD6_3	I/O	PWMD 的捕获输入和脉冲输出
				PWME6	O	PWME 的硬件移相输出
				CMP1-_2	I	比较器负极输入
				OPA3-	I	PGA3 的负极输入
				CAN_TX_2	O	CAN 总线发送脚
				CAN_STB_3	O	CAN 低功耗控制脚
				I2S2_Data	I/O	I2S2 的数据线

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
52	34	26	23	P3.5	I/O	标准 I/O 口
				T1	I	定时器 1 外部时钟输入
				T0CLKO	O	定时器 0 时钟分频输出
				SS_4	I/O	SPI 的从机选择脚（主机为输出）
				PWMAFLT	I	PWMA 外部异常检测脚
				PWMBFLT	I	PWMB 外部异常检测脚
				PWMCFLT	I	PWMC 外部异常检测脚
				PWMDFLT	I	PWMD 外部异常检测脚
				PWMEFLT	I	PWME 外部异常检测脚
				PWMFFLT	I	PWMF 外部异常检测脚
				OPA3+	I	PGA3 的正极输入
				I2S_LRCK	I/O	I2S 的左右声道时钟线
53	35	27	24	P3.6	I/O	标准 I/O 口
				RXD_2	I	串口 1 的接收脚
				INT2	I	外部中断 2
				I2CSDA2_2	I/O	I2C2 的数据线
				PWMC5_2	O	PWMC 的硬件移相输出
				CMP1-	I	比较器负极输入
				CAN_RX_6	I	CAN 总线接收脚
54	36	28	25	P3.7	I/O	标准 I/O 口
				TXD_2	O	串口 1 的发送脚
				INT3	I	外部中断 3
				I2CSCL2_2	I/O	I2C2 的时钟线
				PWMC6_2	O	PWMC 的硬件移相输出
				CMP1+	I	比较器正极输入
				CAN_TX_6	O	CAN 总线发送脚
55	37			P7.0	I/O	标准 I/O 口
				T9	I	定时器 9 外部时钟输入
				S4SS_4	I/O	USART4－SPI 的从机选择脚（主机为输出）
				S5SS_2	I/O	USART5－SPI 的从机选择脚（主机为输出）
				S6SS_2	I/O	USART6－SPI 的从机选择脚（主机为输出）
				SS3_2	I/O	SPI3 的从机选择脚（主机为输出）
				PWMCETR_2	I	PWMC 外部触发输入脚
				PWMDETR_2	I	PWMD 外部触发输入脚
				PWME1P	I/O	PWME 的捕获输入和脉冲输出正极
				PWMF5	I/O	PWMF 的捕获输入和脉冲输出
				OPA3+_3	I	PGA3 的正极输入
				CAN_RX_4	I	CAN 总线接收脚
				QSPI_IO0_3	I/O	QSPI 数据脚

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
56	38			P7.1	I/O	标准 I/O 口
				T9CLKO	O	定时器 9 时钟分频输出
				S4MOSI_4	I/O	USART4－SPI 主出从入
				S5MOSI_2	I/O	USART5－SPI 主出从入
				S6MOSI_2	I/O	USART6－SPI 主出从入
				MOSI3_2	I/O	SPI3 主机输出从机输入
				PWMCFLT_2	I	PWMC 外部异常检测脚
				PWMDFLT_2	I	PWMD 外部异常检测脚
				PWME1N	O	PWME 的脉冲输出负极
				PWMF6	I/O	PWMF 的捕获输入和脉冲输出
				OPA3-_3	I	PGA3 的负极输入
				CAN_TX_4	O	CAN 总线发送脚
				QSPI_CLK_3	O	QSPI 时钟脚
57	39			P7.2	I/O	标准 I/O 口
				S4MISO_4	I/O	USART4－SPI 主入从出
				S5MISO_2	I/O	USART5－SPI 主入从出
				S6MISO_2	I/O	USART6－SPI 主入从出
				MISO3_2	I/O	SPI3 主机输入从机输出
				PWMA5_3	O	PWMA 的硬件移相输出
				PWME2P	I/O	PWME 的捕获输入和脉冲输出正极
				PWMF7	I/O	PWMF 的捕获输入和脉冲输出
				LIN_RX_4	I	LIN 总线接收脚
				CAN2_RX_4	I	CAN2 总线接收脚
				QSPI_IO1_3	I/O	QSPI 数据脚
58	40			P7.3	I/O	标准 I/O 口
				S4SCLK_4	I/O	USART4－SPI 的时钟脚
				S5SCLK_2	I/O	USART5－SPI 的时钟脚
				S6SCLK_2	I/O	USART6－SPI 的时钟脚
				SCLK3_2	I/O	SPI3 的时钟脚
				PWMAETR_3	I	PWMA 外部触发输入脚
				PWMA6_3	O	PWMA 的硬件移相输出
				PWME2N	O	PWME 的脉冲输出负极
				PWMF8	I/O	PWMF 的捕获输入和脉冲输出
				LIN_TX_4	O	LIN 总线发送脚
				CAN2_TX_4	O	CAN2 总线发送脚
				QSPI_IO2_3	I/O	QSPI 数据脚
59				PA.0	I/O	标准 I/O 口
				RXD2_4	I	串口 2 的接收脚
				PWME1P_3	I/O	PWME 的捕获输入和脉冲输出正极
				CAN2_RX_7	I	CAN2 总线接收脚

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
60				PA.1	I/O	标准 I/O 口
				TXD2_4	O	串口 2 的发送脚
				PWME1N_3	O	PWME 的脉冲输出负极
				PWMF5_2	I/O	PWMF 的捕获输入和脉冲输出
				CAN2_TX_7	O	CAN2 总线发送脚
61				PA.2	I/O	标准 I/O 口
				RXD4_3	I	串口 4 的接收脚
				PWME2P_3	I/O	PWME 的捕获输入和脉冲输出正极
62				PA.3	I/O	标准 I/O 口
				TXD4_3	O	串口 4 的发送脚
				PWME2N_3	O	PWME 的脉冲输出负极
				PWMF6_2	I/O	PWMF 的捕获输入和脉冲输出
63				PA.4	I/O	标准 I/O 口
				RxD7_2	I	串口 7 的接收脚
				S7SS_2	I/O	USART7－SPI 的从机选择脚（主机为输出）
				S8SS_2	I/O	USART8－SPI 的从机选择脚（主机为输出）
				PWME3P_3	I/O	PWME 的捕获输入和脉冲输出正极
64				PA.5	I/O	标准 I/O 口
				TxD7_2	O	串口 7 的发送脚
				S7MOSI_2	I/O	USART7－SPI 主出从入
				S8MOSI_2	I/O	USART8－SPI 主出从入
				PWME3N_3	O	PWME 的脉冲输出负极
				PWMF7_2	I/O	PWMF 的捕获输入和脉冲输出
65				PA.6	I/O	标准 I/O 口
				T11_3	I	定时器 11 外部时钟输入
				RxD8_2	I	串口 8 的接收脚
				S7MISO_2	I/O	USART7－SPI 主入从出
				S8MISO_2	I/O	USART8－SPI 主入从出
				PWME4P_3	I/O	PWME 的捕获输入和脉冲输出正极
66				PA.7	I/O	标准 I/O 口
				T11CKO_3	O	定时器 11 时钟分频输出
				TxD8_2	O	串口 8 的发送脚
				S7SCLK_2	I/O	USART7－SPI 的时钟脚
				S8SCLK_2	I/O	USART8－SPI 的时钟脚
				PWME4N_3	O	PWME 的脉冲输出负极
				PWMF8_2	I/O	PWMF 的捕获输入和脉冲输出
67	41	29	26	Vcc	VCC	电源脚
68				P4.1	I/O	标准 I/O 口
				S1MISO_3	I/O	USART1－SPI 主入从出
				S2MISO_3	I/O	USART2－SPI 主入从出
				TxD5_2	O	串口 5 的发送脚
				MISO_3	I/O	SPI 主机输入从机输出

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
				PWMAETR_2	I	PWMA 外部触发输入脚
				PWMC1N_3	O	PWMC 的脉冲输出负极
				CMP1O_2	O	比较器输出
				QSPI_IO1_2	I/O	QSPI 数据脚
				I2S_LRCK_4	I/O	I2S 的左右声道时钟线
69	42	30	27	P4.2	I/O	标准 I/O 口
				WR	O	外部总线的写信号线
				PWMC2P_3	I/O	PWMC 的捕获输入和脉冲输出正极
				OPA2O	O	PGA2 的输出
				OPA3-_2	I	PGA3 的负极输入
				CAN_RX_3	I	CAN 总线接收脚
70	43	31	28	P4.3	I/O	标准 I/O 口
				ADC_ETR_2	I	ADC 外部触发脚
				ADCB_ETR_2	I	ADCB 外部触发脚
				RXD_4	I	串口 1 的接收脚
				S1SCLK_3	I/O	USART1－SPI 的时钟脚
				S2SCLK_3	I/O	USART2－SPI 的时钟脚
				SCLK_3	I/O	SPI 的的时钟脚
				PWMC2N_3	O	PWMC 的脉冲输出负极
				PWME5_2	O	PWME 的硬件移相输出
				CMP4+_2	I	比较器 4 正极输入
				CAN_RX_8	I	CAN 总线接收脚
				QSPI_CLK_2	O	QSPI 时钟脚
				I2S_BCK_4	I/O	I2S 的位时钟线
71	44	32	29	P4.4	I/O	标准 I/O 口
				RD	O	外部总线的读信号线
				T11_4	I	定时器 11 外部时钟输入
				TXD_4	O	串口 1 的发送脚
				PWMC3P_3	I/O	PWMC 的捕获输入和脉冲输出正极
				PWME6_2	O	PWME 的硬件移相输出
				CMP4-_2	I	比较器 4 负极输入
				CAN_TX_8	O	CAN 总线发送脚
72	45	33	30	P2.0	I/O	标准 I/O 口
				A8	I/O	地址/数据总线
				T5	I	定时器 5 外部时钟输入
				RxD5_4	I	串口 5 的接收脚
				PWMA1P_2	I/O	PWMA 的捕获输入和脉冲输出正极
				PWMB5	I/O	PWMB 的捕获输入和脉冲输出
				OPA3+_2	I	PGA3 的正极输入

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
73	46	34	31	P2.1	I/O	标准 I/O 口
				A9	I/O	地址/数据总线
				T5CLKO	O	定时器 5 时钟分频输出
				TxD5_4	O	串口 5 的发送脚
				PWMA1N_2	O	PWMA 的脉冲输出负极
				PWMB6	I/O	PWMB 的捕获输入和脉冲输出
				OPA3O_2	O	PGA3 的输出
74	47	35	32	P2.2	I/O	标准 I/O 口
				A10	I/O	地址/数据总线
				T6	I	定时器 6 外部时钟输入
				S1SS_2	I/O	USART1－SPI 的从机选择脚（主机为输出）
				S2SS_2	I/O	USART2－SPI 的从机选择脚（主机为输出）
				S5SS	I/O	USART5－SPI 的从机选择脚（主机为输出）
				S6SS	I/O	USART6－SPI 的从机选择脚（主机为输出）
				SS_2	I/O	SPI 的从机选择脚（主机为输出）
				SS3	I/O	SPI3 的从机选择脚（主机为输出）
				PWMA2P_2	I/O	PWMA 的捕获输入和脉冲输出正极
				PWMB7	I/O	PWMB 的捕获输入和脉冲输出
				CAN_STB_2	O	CAN 低功耗控制脚
				I2S_LRCK_3	I/O	I2S 的左右声道时钟线
75	48	36	33	P2.3	I/O	标准 I/O 口
				A11	I/O	地址/数据总线
				T6CLKO	O	定时器 6 时钟分频输出
				S1MOSI_2	I/O	USART1－SPI 主出从入
				S2MOSI_2	I/O	USART2－SPI 主出从入
				S5MOSI	I/O	USART5－SPI 主出从入
				S6MOSI	I/O	USART6－SPI 主出从入
				MOSI_2	I/O	SPI 主机输出从机输入
				MOSI3	I/O	SPI3 主机输出从机输入
				PWMA2N_2	O	PWMA 的脉冲输出负极
				PWMB8	I/O	PWMB 的捕获输入和脉冲输出
				CAN2_STB_2	O	CAN2 低功耗控制脚
				CAN2_STB_7	O	CAN2 低功耗控制脚
				I2S_Data_3	I/O	I2S 的数据线
76	49	37	34	P2.4	I/O	标准 I/O 口
				A12	I/O	地址/数据总线
				T7	I	定时器 7 外部时钟输入
				S1MISO_2	I/O	USART1－SPI 主入从出
				S2MISO_2	I/O	USART2－SPI 主入从出
				S5MISO	I/O	USART5－SPI 主入从出
				S6MISO	I/O	USART6－SPI 主入从出
				MISO_2	I/O	SPI 主机输入从机输出

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
				MISO3	I/O	SPI3 主机输入从机输出
				I2CSDA_2	I/O	I2C 的数据线
				PWMA3P_2	I/O	PWMA 的捕获输入和脉冲输出正极
				PWMD5	I/O	PWMD 的捕获输入和脉冲输出
				CMP3O	O	比较器 3 输出
				I2S_MCLK_3	I/O	I2S 的主时钟线
77	50	38	35	P2.5	I/O	标准 I/O 口
				A13	I/O	地址/数据总线
				T7CLKO	O	定时器 7 时钟分频输出
				S1SCLK_2	I/O	USART1－SPI 的时钟脚
				S2SCLK_2	I/O	USART2－SPI 的时钟脚
				S5SCLK	I/O	USART5－SPI 的时钟脚
				S6SCLK	I/O	USART6－SPI 的时钟脚
				SCLK_2	I/O	SPI 的时钟脚
				SCLK3	I/O	SPI3 的时钟脚
				I2CSCL_2	I/O	I2C 的时钟线
				PWMA3N_2	O	PWMA 的脉冲输出负极
				PWMD6	I/O	PWMD 的捕获输入和脉冲输出
				CMP4O	O	比较器 4 输出
				I2S_BCK_3	I/O	I2S 的位时钟线
78	51	39	36	P2.6	I/O	标准 I/O 口
				A14	I/O	地址/数据总线
				T8	I	定时器 8 外部时钟输入
				RxD6_4	I	串口 6 的接收脚
				I2CSDA2	I/O	I2C2 的数据线
				PWMA4P_2	I/O	PWMA 的捕获输入和脉冲输出正极
				PWMD7	I/O	PWMD 的捕获输入和脉冲输出
				CMP4-	I	比较器 4 负极输入
				OPA2+	I	PGA2 的正极输入
				LIN2_RX_2	I	LIN2 总线接收脚
79	52	40	37	P2.7	I/O	标准 I/O 口
				A15	I/O	地址/数据总线
				T8CLKO	O	定时器 8 时钟分频输出
				TxD6_4	O	串口 6 的发送脚
				I2CSCL2	I/O	I2C2 的时钟线
				PWMA4N_2	O	PWMA 的脉冲输出负极
				PWMD8	I/O	PWMD 的捕获输入和脉冲输出
				CMP4+	I	比较器 4 正极输入
				OPA2-	I	PGA2 的负极输入
				LIN2_TX_2	O	LIN2 总线发送脚

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
80	53			P7.4	I/O	标准 I/O 口
				S2SS_4	I/O	USART2－SPI 的从机选择脚（主机为输出）
				S7SS_3	I/O	USART7－SPI 的从机选择脚（主机为输出）
				S8SS_3	I/O	USART8－SPI 的从机选择脚（主机为输出）
				PWMB5_4	I/O	PWMB 的捕获输入和脉冲输出
				PWME3P	I/O	PWME 的捕获输入和脉冲输出正极
				OPA1O_3	O	PGA1 的输出
				OPA2O_3	O	PGA2 的输出
				OPA3O_3	O	PGA3 的输出
				OPA4O_3	O	PGA4 的输出
				CAN_STB_8	O	CAN 低功耗控制脚
				QSPI_IO3_3	I/O	QSPI 数据脚
81	54			P7.5	I/O	标准 I/O 口
				S2MOSI_4	I/O	USART2－SPI 主出从入
				S7MOSI_3	I/O	USART7－SPI 主出从入
				S8MOSI_3	I/O	USART8－SPI 主出从入
				PWMB6_4	I/O	PWMB 的捕获输入和脉冲输出
				PWME3N	O	PWME 的脉冲输出负极
				OPA2+_3	I	PGA2 的正极输入
				QSPI_NCS_3	O	QSPI 片选脚
82	55			P7.6	I/O	标准 I/O 口
				T11	I	定时器 11 外部时钟输入
				S2MISO_4	I/O	USART2－SPI 主入从出
				S7MISO_3	I/O	USART7－SPI 主入从出
				S8MISO_3	I/O	USART8－SPI 主入从出
				I2CSDA_3	I/O	I2C 的数据线
				PWMB7_4	I/O	PWMB 的捕获输入和脉冲输出
				PWME4P	I/O	PWME 的捕获输入和脉冲输出正极
				CMP3-	I	比较器 3 负极输入
				OPA2-_3	I	PGA2 的负极输入
83	56			P7.7	I/O	标准 I/O 口
				T11CKO	O	定时器 11 时钟分频输出
				S2SCLK_4	I/O	USART2－SPI 的时钟脚
				S7SCLK_3	I/O	USART7－SPI 的时钟脚
				S8SCLK_3	I/O	USART8－SPI 的时钟脚
				I2CSCL_3	I/O	I2C 的时钟线
				PWMB8_4	I/O	PWMB 的捕获输入和脉冲输出
				PWME4N	O	PWME 的脉冲输出负极
				CMP3+	I	比较器 3 正极输入
				OPA2+_4	I	PGA2 的正极输入

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
84				PB.0	I/O	标准 I/O 口
				T9_2	I	定时器 9 外部时钟输入
				PWMC1P_4	I/O	PWMC 的捕获输入和脉冲输出正极
85				PB.1	I/O	标准 I/O 口
				T9CLKO_2	O	定时器 9 时钟分频输出
				S3SS_3	I/O	USART3－SPI 的从机选择脚（主机为输出）
				S4SS_3	I/O	USART4－SPI 的从机选择脚（主机为输出）
				SS2_3	I/O	SPI2 的从机选择脚（主机为输出）
				PWMC1N_4	O	PWMC 的脉冲输出负极
				PWMFETR_2	I	PWMF 外部触发输入脚
				I2S2_LRCK_4	I/O	I2S2 的左右声道时钟线
86				PB.2	I/O	标准 I/O 口
				S3MOSI_3	I/O	USART3－SPI 主出从入
				S4MOSI_3	I/O	USART4－SPI 主出从入
				MOSI2_3	I/O	SPI2 主机输出从机输入
				PWMC2P_4	I/O	PWMC 的捕获输入和脉冲输出正极
				I2S2_Data_4	I/O	I2S2 的数据线
87				PB.3	I/O	标准 I/O 口
				S3MISO_3	I/O	USART3－SPI 主入从出
				S4MISO_3	I/O	USART4－SPI 主入从出
				MISO2_3	I/O	SPI2 主机输入从机输出
				PWMC2N_4	O	PWMC 的脉冲输出负极
				PWMFFLT_2	I	PWMF 外部异常检测脚
				I2S2_MCLK_4	I/O	I2S2 的主时钟线
88				PB.4	I/O	标准 I/O 口
				S3SCLK_3	I/O	USART3－SPI 的时钟脚
				S4SCLK_3	I/O	USART4－SPI 的时钟脚
				SCLK2_3	I/O	SPI2 的时钟脚
				PWMC3P_4	I/O	PWMC 的捕获输入和脉冲输出正极
				PWMD5_4	I/O	PWMD 的捕获输入和脉冲输出
				PWME5_3	O	PWME 的硬件移相输出
				I2S2_BCK_4	I/O	I2S2 的位时钟线
89				PB.5	I/O	标准 I/O 口
				PWMC3N_4	O	PWMC 的脉冲输出负极
				PWMD6_4	I/O	PWMD 的捕获输入和脉冲输出
				PWME6_3	O	PWME 的硬件移相输出
90				PB.6	I/O	标准 I/O 口
				T10_2	I	定时器 10 外部时钟输入
				RXD3_4	I	串口 3 的接收脚
				PWMC4P_4	I/O	PWMC 的捕获输入和脉冲输出正极
				PWMD7_4	I/O	PWMD 的捕获输入和脉冲输出

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
91				PB.7	I/O	标准 I/O 口
				T10CLKO_2	O	定时器 10 时钟分频输出
				TXD3_4	O	串口 3 的发送脚
				PWMC4N_4	O	PWMC 的脉冲输出负极
				PWMD8_4	I/O	PWMD 的捕获输入和脉冲输出
92				Core_Vcc	VCC	内核电源脚
93	57	41	38	P4.5	I/O	标准 I/O 口
				ALE	O	地址锁存信号
				T11CKO_4	O	定时器 11 时钟分频输出
				PWMAFLT_2	I	PWMA 外部异常检测脚
				PWMC3N_3	O	PWMC 的脉冲输出负极
				CMP3O_2	O	比较器 3 输出
				CAN_TX_3	O	CAN 总线发送脚
				CAN_STB_4	O	CAN 低功耗控制脚
94	58	42	39	P4.6	I/O	标准 I/O 口
				RXD2_2	I	串口 2 的接收脚
				PWMA5	O	PWMA 的硬件移相输出
				PWMC4P_3	I/O	PWMC 的捕获输入和脉冲输出正极
				CMP4O_2	O	比较器 4 输出
				LIN_RX_3	I	LIN 总线接收脚
				CAN2_RX_3	I	CAN2 总线接收脚
				QSPI_IO2_2	I/O	QSPI 数据脚
95	59	43	40	P0.0	I/O	标准 I/O 口
				AD0	I/O	地址/数据总线
				ADC8	I	ADC 模拟输入通道 8
				ADCB0	I	ADCB 模拟输入通道 0
				T3_2	I	定时器 3 外部时钟输入
				RXD3	I	串口 3 的接收脚
				PWMB5_3	I/O	PWMB 的捕获输入和脉冲输出
				PWMFETR_3	I	PWMF 外部触发输入脚
				OPA1O_2	O	PGA1 的输出
				OPA2O_2	O	PGA2 的输出
				CAN_RX	I	CAN 总线接收脚
96	60	44	41	P0.1	I/O	标准 I/O 口
				AD1	I/O	地址/数据总线
				ADC9	I	ADC 模拟输入通道 9
				ADCB1	I	ADCB 模拟输入通道 1
				T3CLKO_2	O	定时器 3 时钟分频输出
				TXD3	O	串口 3 的发送脚
				PWMB6_3	I/O	PWMB 的捕获输入和脉冲输出
				PWMFFLT_3	I	PWMF 外部异常检测脚
				CAN_TX	O	CAN 总线发送脚

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
97	61	45	42	P0.2	I/O	标准 I/O 口
				AD2	I/O	地址/数据总线
				ADC10	I	ADC 模拟输入通道 10
				ADCB2	I	ADCB 模拟输入通道 2
				T4_2	I	定时器 4 外部时钟输入
				RXD4	I	串口 4 的接收脚
				S3SS_2	I/O	USART3－SPI 的从机选择脚（主机为输出）
				S4SS_2	I/O	USART4－SPI 的从机选择脚（主机为输出）
				SS2_2	I/O	SPI2 的从机选择脚（主机为输出）
				PWMB7_3	I/O	PWMB 的捕获输入和脉冲输出
				PWMEETR_2	I	PWME 外部触发输入脚
				LIN_RX	I	LIN 总线接收脚
				CAN2_RX	I	CAN2 总线接收脚
98	62	46	43	P0.3	I/O	标准 I/O 口
				AD3	I/O	地址/数据总线
				ADCB3	I	ADCB 模拟输入通道 3
				T4CLKO_2	O	定时器 4 时钟分频输出
				TXD4	O	串口 4 的发送脚
				S3MOSI_2	I/O	USART3－SPI 主出从入
				S4MOSI_2	I/O	USART4－SPI 主出从入
				MOSI2_2	I/O	SPI2 主机输出从机输入
				PWMB8_3	I/O	PWMB 的捕获输入和脉冲输出
				PWMEFLT_2	I	PWME 外部异常检测脚
				LIN_TX	O	LIN 总线发送脚
				CAN2_TX	O	CAN2 总线发送脚
99	63	47	44	P0.4	I/O	标准 I/O 口
				AD4	I/O	地址/数据总线
				ADCB4	I	ADCB 模拟输入通道 4
				T3	I	定时器 3 外部时钟输入
				S3MISO_2	I/O	USART3－SPI 主入从出
				S4MISO_2	I/O	USART4－SPI 主入从出
				RxD5	I	串口 5 的接收脚
				MISO2_2	I/O	SPI2 主机输入从机输出
				PWMA5_2	O	PWMA 的硬件移相输出
				OPA1+	I	PGA1 的正极输入
				OPA1-_2	I	PGA1 的负极输入

编号				名称	类型	说明
LQFP100	LQFP64	LQFP48	LQFP44			
100	64	48	44	P5.2	I/O	标准 I/O 口
				RXD4_2	I	串口 4 的接收脚
				RxD8	I	串口 8 的接收脚
				S7MISO	I/O	USART7－SPI 主入从出
				S8MISO	I/O	USART8－SPI 主入从出
				PWMD7_3	I/O	PWMD 的捕获输入和脉冲输出
				CMP3-_2	I	比较器 3 负极输入
				LIN_RX_2	I	LIN 总线接收脚
				CAN2_RX_2	I	CAN2 总线接收脚
				I2S2_MCLK	I/O	I2S2 的主时钟线

3.1.8 USB-Link1D 对 STC32G144K246 自动停电 / 上电烧录，串口仿真+串口通讯



USB Link1D工具：支持全自动停电-上电在线下载 / 脱机下载 / 仿真

【应用场景一：从本工具给目标系统 自动 停电/上电，供电】

点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0.5秒/再自动供电，
数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0.5秒/再自动供电给目标系统工作。

【应用场景二：不从本工具给目标系统供电】

1、点击 电脑端 ISP 软件的【下载/编程】按钮
2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，
下载编程进行中，数秒后提示下载编程成功，目标MCU会自动复位到用户程序区自动跑用户程序。
部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P5.4/RST变成复位脚

为优先保证ADC的精度及其它模拟电路，

就希望 Gnd 不要流过大的电流，建议：
如果要 I/O 驱动大电流，如 LED灯：
用 强推挽的 高 输出电流，这个只影响Vcc；
不要用 I/O 置低，如置低流入电流导通，
MCU-Gnd 会被流入的大电流抬高，
会造成测量误差，建议 ADC转换时，I/O 不驱动大电流。

经过I/O口流进和流出的电流不要超过60mA。
如果要用I/O口直接点亮发光二极管，
请将I/O口工作在强推挽模式，高点亮。
P3.0/P3.1不要接其它电路，防干扰下载。
P3.2建议做按键使用，或输出到高阻输入的电路使用，
USB型MCU，P3.2会参与下载判断，
P3.2不要上电时被固定拉低/否则会等待0.5s判断USB下载。

【注意】：

1、P1.2和P5.4封装时短接在一起，打线都打到第27Pin管脚

特别内部数字设计：

P5.4 也可以通过寄存器P_SWX1特别设置切换成 P1.2的数字功能，
但模拟功能 ADC2无法切换，后决定封装时将P1.2和P5.4封装时短接

2、P1.3和P1.7封装时短接在一起，打线都打到第24Pin脚

3、ADC_VRef+管脚接 ADC的参考源，或直接接 MCU_Vcc

4、ADC_AGnd/ADC_VRef-管脚接模拟地

5、模拟地，数字地，最后在系统地处短接在一起

6、Core-Vcc：内部 1.5V-LDO，外部接 10uF ~ 22uF 电容 即可

USB直接下载：

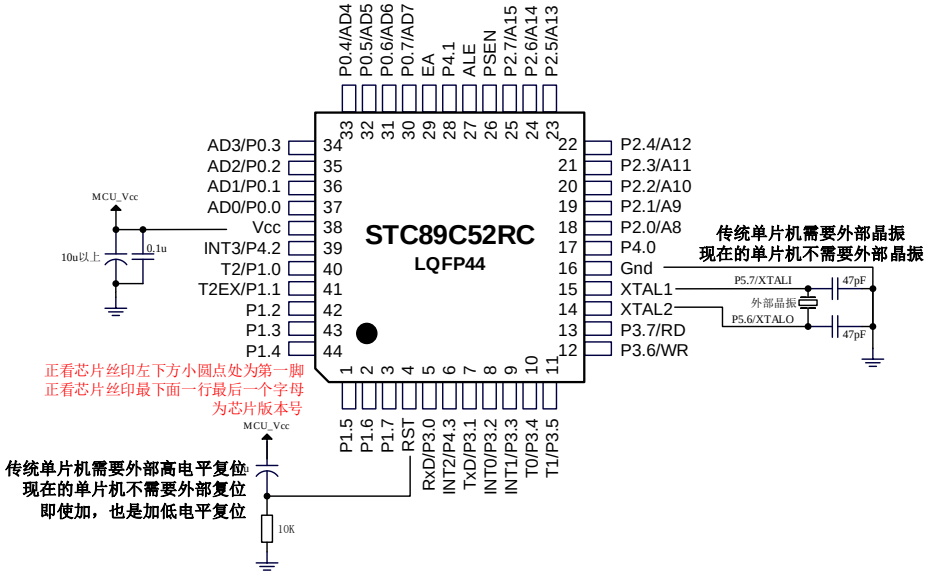
在单片机【P3.0/D-，P3.1/D+】和电脑的USB口连接好的情况下，
将P3.2按键按下（即P3.2 = 0），再给单片机上电，如已上电，
需停电再给MCU重新上电，则MCU会进入【等待USB下载模式】，
点击电脑端ISP下载软件中的“下载”按钮即可下载。

ISP 下载步骤：

1、按照如图所示的连接方式将 USB-Link1D 和目标芯片连接
2、点击 ISP 下载软件中的“下载/编程”按钮
3、开始 ISP 下载
（注意：若是使用 USB-Link1D 给目标系统供电，目标系统的总电流不能大于 200mA，否则会导致下载失败。）

比较下传统的 89C52RC 系列相应下载线路图：

传统单片机需要外部晶振，现在的单片机不需要外部晶振
传统单片机需要外部高电平复位，现在的不需要，并且已改为低电平复位





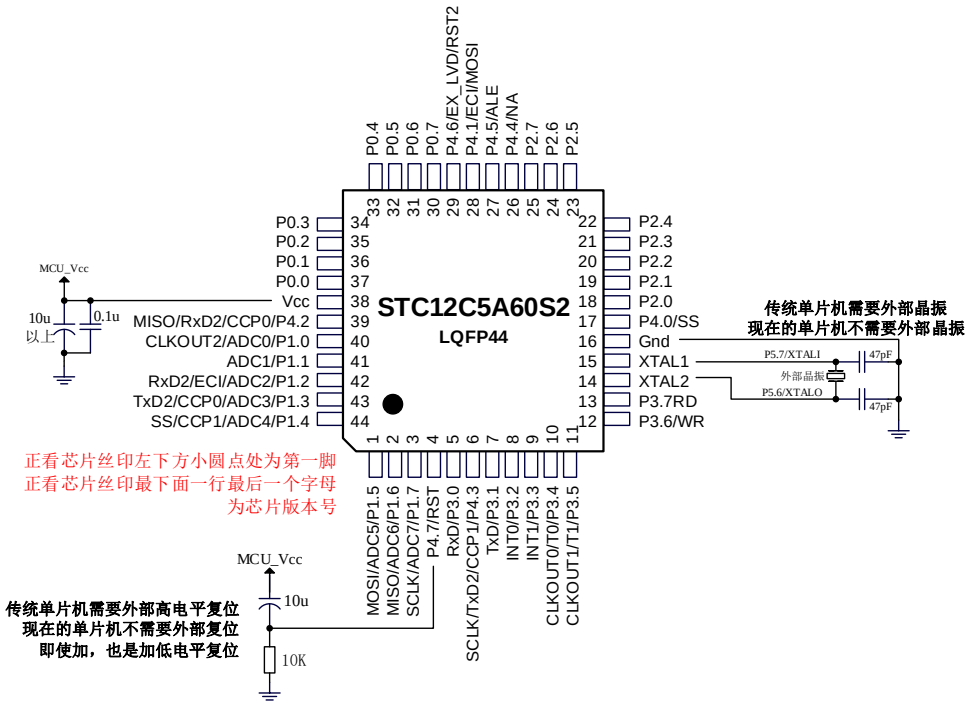
USB Link1D工具：支持全自动停电-上电在线下载 / 脱机下载 / 仿真

【应用场景一：从本工具给目标系统 自动 停电/上电，供电】
点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0.5秒/再自动供电，
数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0.5秒/再自动供电给目标系统工作。

【应用场景二：不从本工具给目标系统供电】
1、点击 电脑端 ISP 软件的【下载/编程】按钮
2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，
下载编程进行中，数秒后提示下载编程成功，目标MCU会自动复位到用户程序区自动跑用户程序。
部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P4.7/nRST变成复位脚

比较下传统的 12C5A60S2 相应下载线路图：

传统单片机需要外部晶振，现在的单片机不需要外部晶振
传统单片机需要外部高电平复位，现在的不需要，并且已改为低电平复位





USB Link1D工具：支持全自动停电-上电在线下载 / 脱机下载 / 仿真

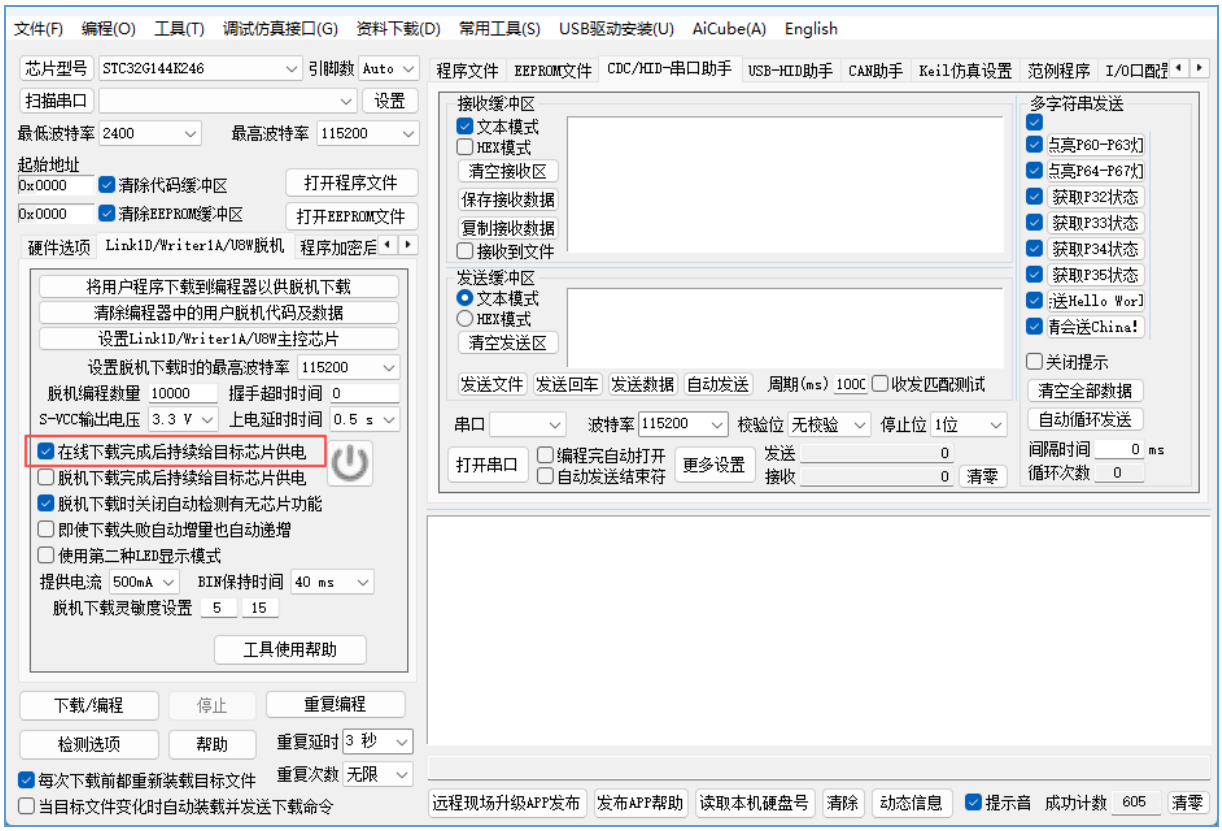
【应用场景一：从本工具给目标系统 自动 停电/上电，供电】

点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0.5秒/再自动供电，
数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0.5秒/再自动供电给目标系统工作。

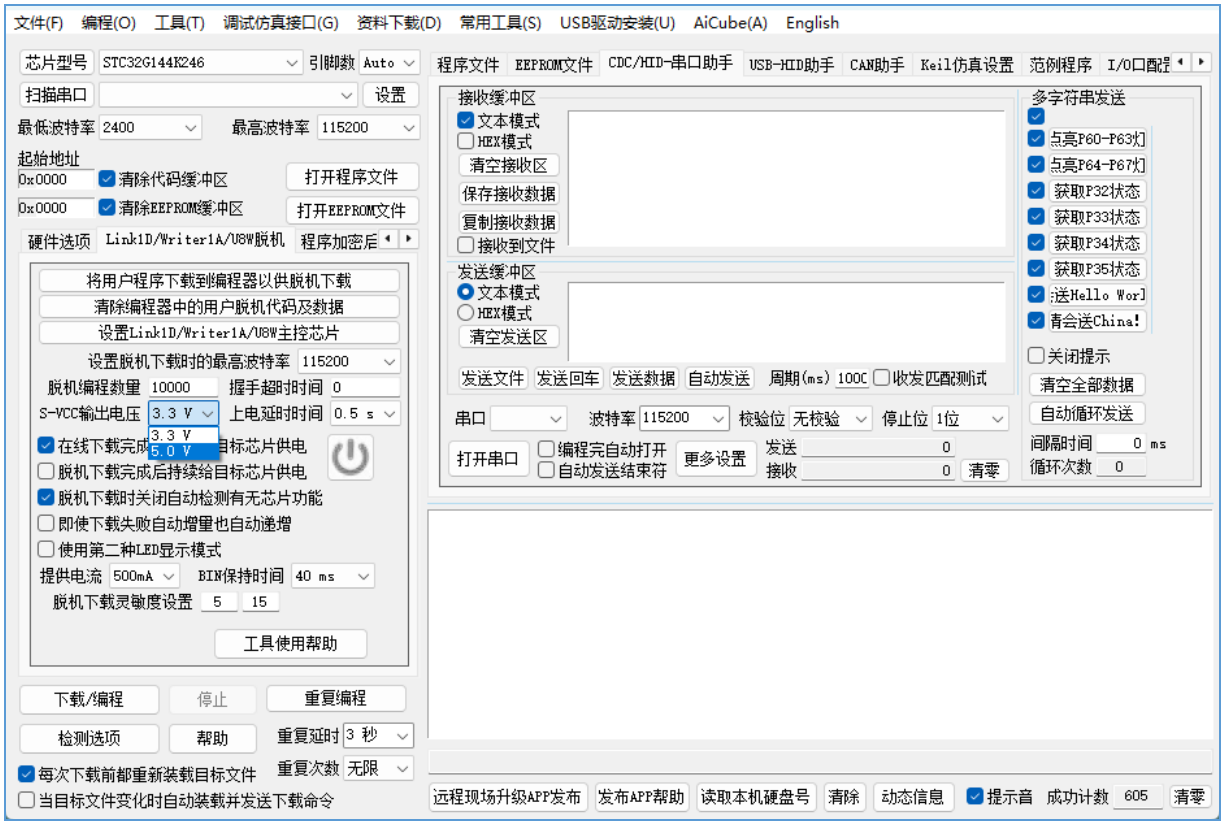
【应用场景二：不从本工具给目标系统供电】

1、点击 电脑端 ISP 软件的【下载/编程】按钮
2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，
下载编程进行中，数秒后提示下载编程成功，目标MCU会自动复位到用户程序区自动跑用户程序。
部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P4.7/nRST变成复位脚

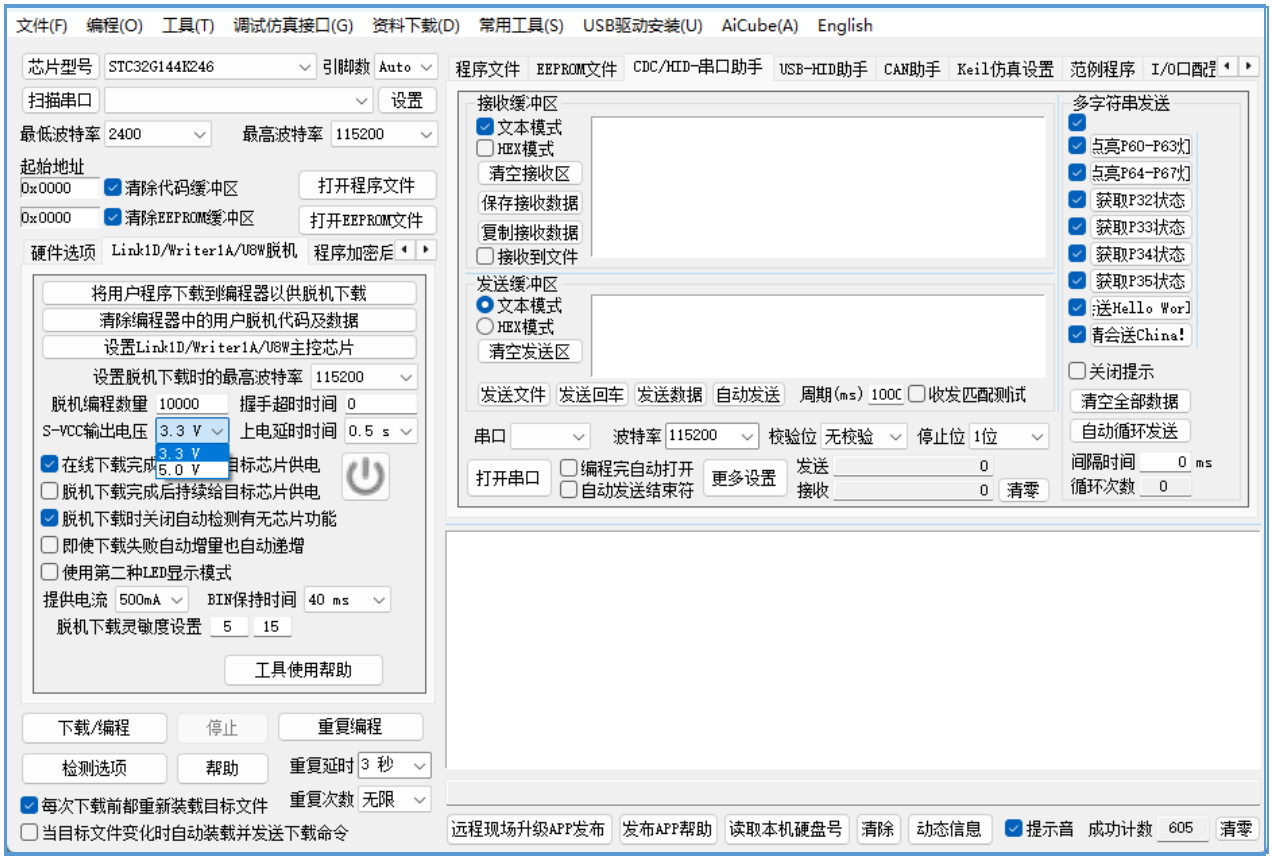
如何设置 USB-Link1D 下载完后持续给目标芯片供电



如何设置 USB-Link1D 输出 5V

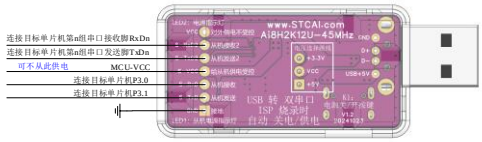


如何设置 USB-Link1D 输出 3.3V



3.1.9 【USB 转双串口@Ai8H2K12U-SOP16】工具自动烧录，串口仿真+串口通讯

【USB转双串口 芯片或工具】，支持 全自动 停电 / 上电，在线下载 / 脱机下载，仿真



USB转双串口工具
@Ai8H2K12U-SOP16

- **包邮RMB￥4.10**
- **新客户【免费+包邮送】**

【应用场景一：从本工具给目标系统 自动 停电/上电，供电】
点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0.5秒/再自动供电，
数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0.5秒/再自动供电给目标系统工作。

【应用场景二：不从本工具给目标系统供电】
1、点击 电脑端 ISP 软件的【下载/编程】按钮
2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，
电脑端软件下载编程进行中，数秒后下载成功，目标MCU会自动复位到用户程序区自动跑用户程序。
部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P5.4/nRST变成复位脚

为优先保证ADC的精度及其它模拟电路，

就希望 Gnd 不要流过大的电流，建议：
如果要 I/O 驱动大电流，如 LED灯；
用 强推挽的 高 输出电流，这个只影响Vcc；
不要用 I/O 置低，如置低流入电流导通，
MCU-Gnd 会被流入的大电流抬高，
会造成测量误差，建议 ADC转换时，I/O 不驱动大电流。

经过I/O口流进和流出的电流不要超过60mA。
如果要用I/O口直接点亮发光二极管，
请将I/O口工作在强推挽模式，高点亮。
P3.0/P3.1不要接其它电路，防干扰下载。
P3.2建议做按键使用，或输出到高阻输入的电路使用，
USB型MCU，P3.2会参与下载判断，
P3.2不要上电时被固定拉低/否则会等待0.5s判断USB下载。



【注意】：
1、P1.2和P5.4封装时短接在一起，打线都打到第27Pin管脚
特别内部数字设计：
P5.4 也可以通过寄存器P_SWX1特别设置切换成 P1.2的数字功能，
但模拟功能 ADC2无法切换，后决定封装时将P1.2和P5.4封装时短接
2、P1.3和P1.7封装时短接在一起，打线都打到第24Pin脚
3、ADC_VRef+管脚接 ADC的参考源，或直接接 MCU_Vcc
4、ADC_AGnd/ADC_VRef-管脚接模拟地
5、模拟地，数字地，最后在系统地处短接在一起
6、Core-Vcc：内部 1.5V-LDO，外部接 10uF ~ 22uF 电容 即可

UCAP 是内部USB模块的 3.3V-LDO，外接 0.1uF 去耦电容
1、USB 不用就不用接外部电容
2、MCU_Vcc 工作在 3.6V以上，UCAP管脚可挂 0.1uF 去耦电容；
3、MCU_Vcc 工作在 3.6V以下，UCAP管脚直接短接到 MCU_Vcc；
4、或不管任何情况，UCAP管脚统一外挂 0.1uF 去耦电容

P5.4/nRST出厂时默认是P5.4/I/O功能，要改为复位功能，
需ISP软件对**软件设置**复位脚用作I/O口，
此状态下，停电一次再上电，新的设置才生效，
程序区中用户程序也可改为复位脚或I/O，这个立即生效。

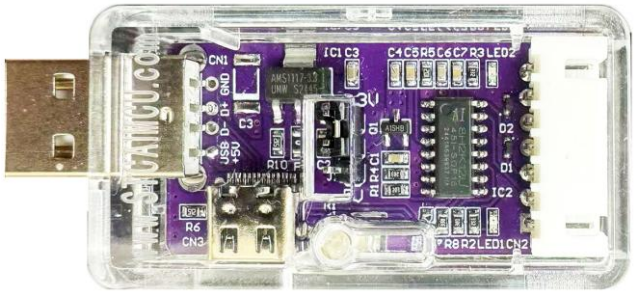
USB直接下载：
在单片机【P3.0/D-，P3.1/D+】和电脑的USB口连接好的情况下，
将P3.2按键按下（即P3.2 = 0），再给单片机上电，如已上电，
需停电再给MCU重新上电，则MCU会进入【等待USB下载模式】，
点击电脑端ISP下载软件中的“下载”按钮即可下载。

建议直接使用如下工具，可以【免费+包邮送】

USB 转双串口工具@Ai8H2K12U-SOP16，包邮 RMB￥4.10

新客户【免费+包邮送】

正面：



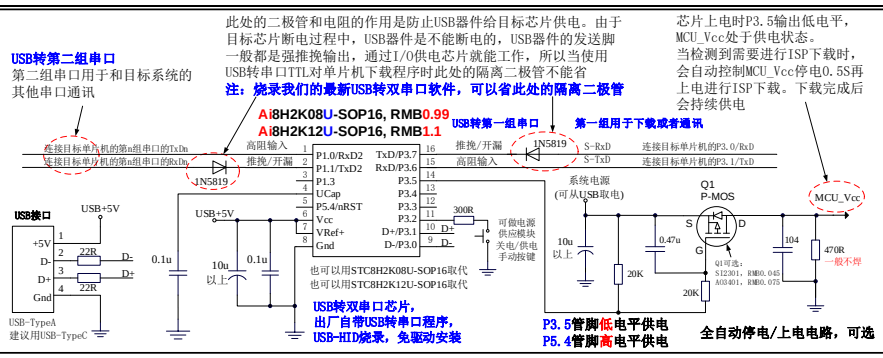


扫码去微信
小商城

背面：



3.1.10 USB 转双串口芯片全自动停电/上电烧录，串口仿真+串口通讯，5V



【应用场景一：从本工具给目标系统 自动 停电/上电，供电】

点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0.5秒/再自动供电，数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0.5秒/再自动供电给目标系统工作。

【应用场景二：不从本工具给目标系统供电】

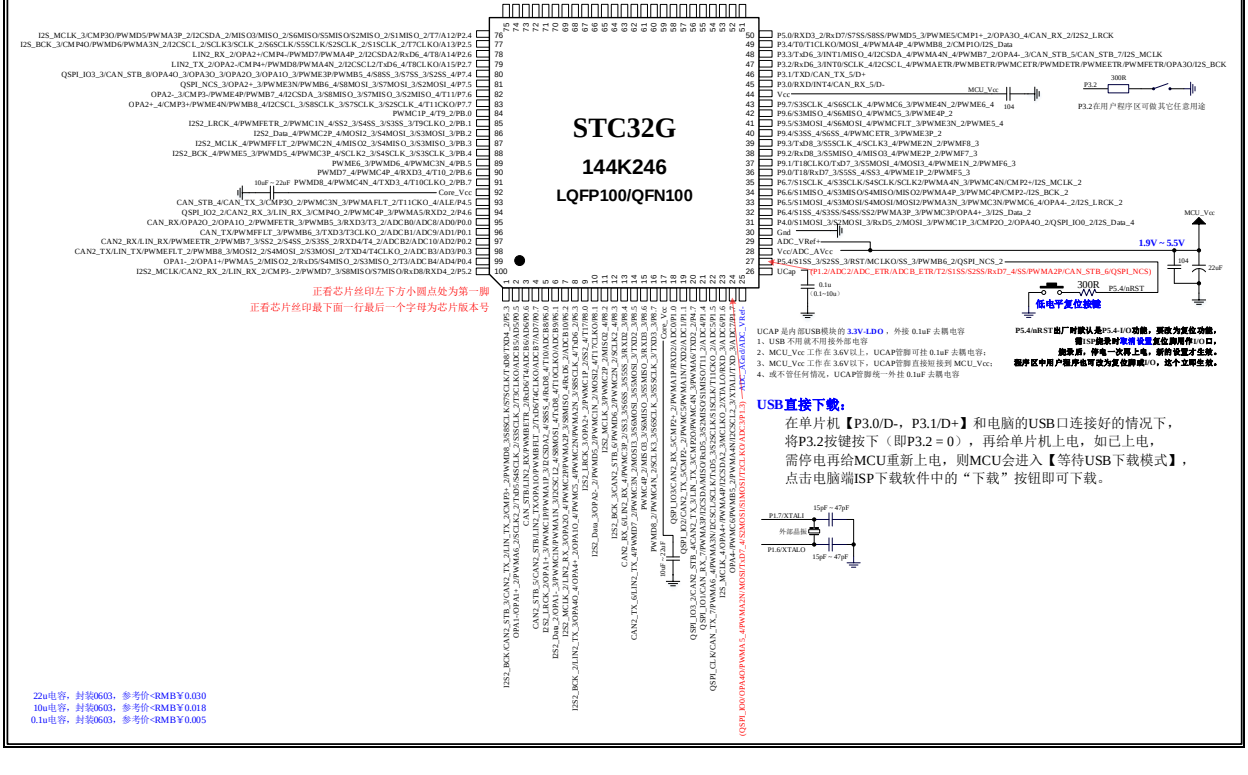
1、点击 电脑端 ISP 软件的【下载/编程】按钮

2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，下载编程进行中，数秒后提示下载编程成功，目标MCU会自动复位到用户程序区自动跑用户程序。
部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P5.4/nRST变成复位脚

为优先保证ADC的精度及其它模拟电路，

就希望 Gnd 不要流过大的电流，建议：
如果要 I/O 驱动大电流，如 LED灯：
用 强推挽的 高 输出电流，这个只影响Vcc：
不要用 I/O 置低，如置低流入电流导通，
MCU-Gnd 会被流入的大电流抬高，
会造成测量误差，建议 ADC转换时，I/O 不驱动大电流。

经过I/O口流进和流出的电流不要超过60mA。
如果要用I/O口直接点亮发光二极管，
请将I/O口工作在强推挽模式，高点亮。
P3.0/P3.1不要接其它电路，防干扰下载。
P3.2建议做按键使用，或输出到高阻输入的电路使用，
USB型MCU，P3.2会参与下载判断，
P3.2不要上电时被固定拉低/否则会等待0.5s判断USB下载。



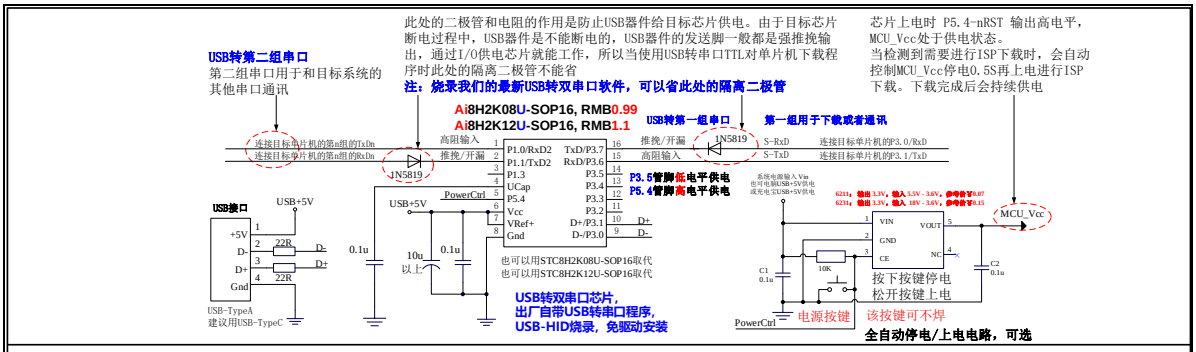
【注意】：

1、P1.2和P5.4封装时短接在一起，打线都打到第27Pin管脚
特别内部数字设计：
P5.4 也可以通过寄存器P_SWX1特别设置切换成 P1.2的数字功能，但模拟功能 ADC2无法切换，后决定封装时将P1.2和P5.4封装时短接
2、P1.3和P1.7封装时短接在一起，打线都打到第24Pin脚
3、ADC_VRef+管脚接 ADC的参考源，或直接接 MCU_Vcc
4、ADC_AGnd/ADC_VRef-管脚接模拟地
5、模拟地，数字地，最后在系统地处短接在一起
6、Core-Vcc：内部 1.5V-LDO，外部接 10uF ~ 22uF 电容 即可

USB直接下载：

在单片机【P3.0/D-，P3.1/D+】和电脑的USB口连接好的情况下，将P3.2按键按下（即P3.2 = 0），再给单片机上电，如已上电，需停电再给MCU重新上电，则MCU会进入【等待USB下载模式】，点击电脑端ISP下载软件中的“下载”按钮即可下载。

3.1.11 USB 转双串口芯片全自动烧录，串口仿真+串口通讯，3.3V 原理图



【应用场景一：从本工具给目标系统 自动 停电/上电，供电】

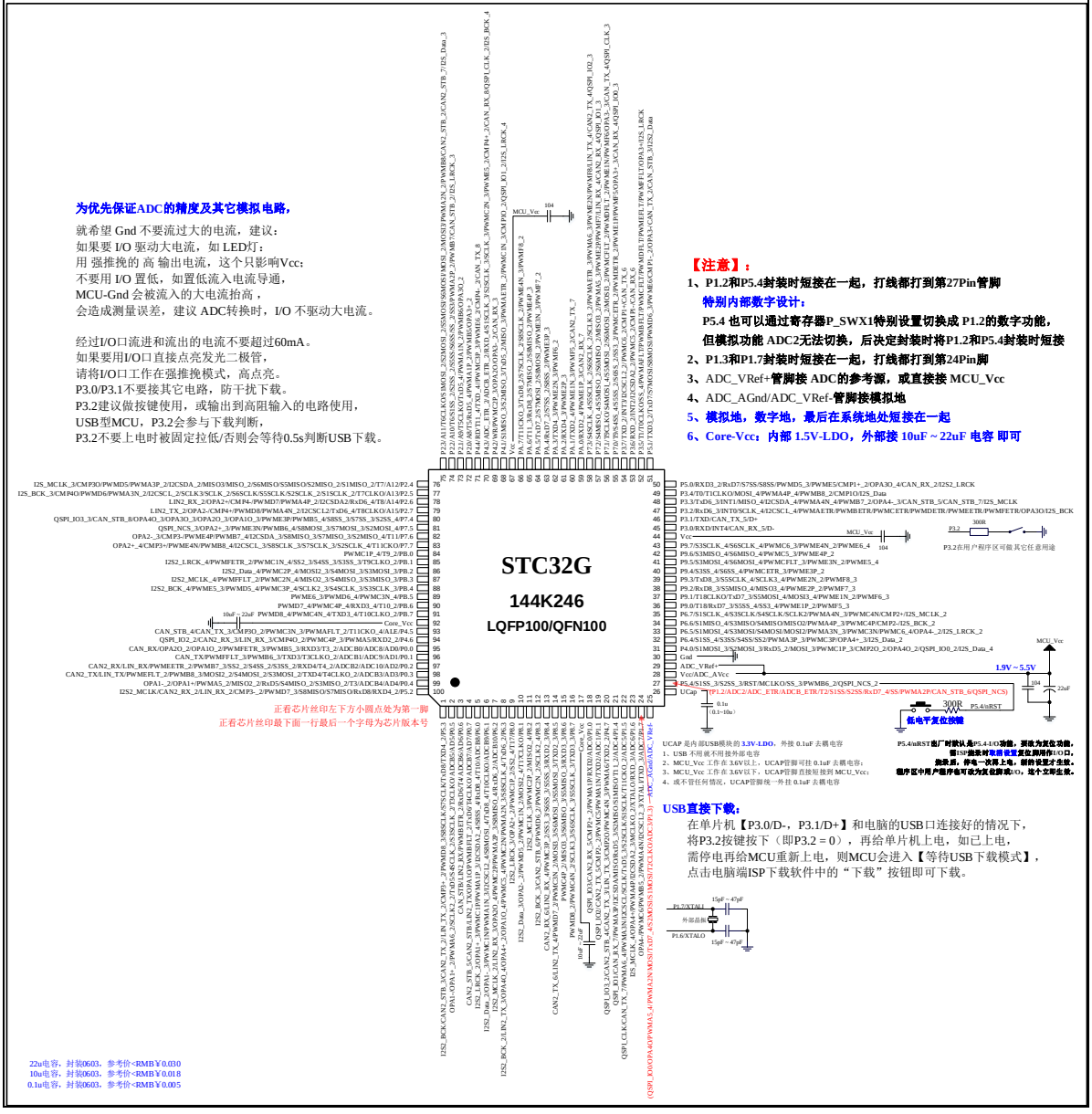
点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0.5秒/再自动供电，数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0.5秒/再自动供电给目标系统工作。

【应用场景二：不从本工具给目标系统供电】

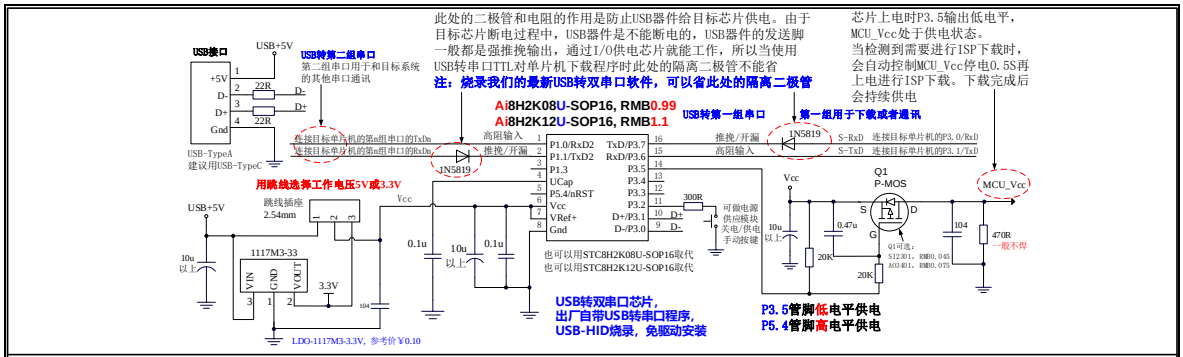
1、点击 电脑端 ISP 软件的【下载/编程】按钮

2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，下载编程进行中，数秒后提示下载编程成功，目标MCU会自动复位到用户程序区自动跑用户程序。

部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P5.4/nRST变成复位脚



3.1.12 USB 转双串口芯片进行自动烧录/仿真+串口通讯，5V/3.3V 跳线选择

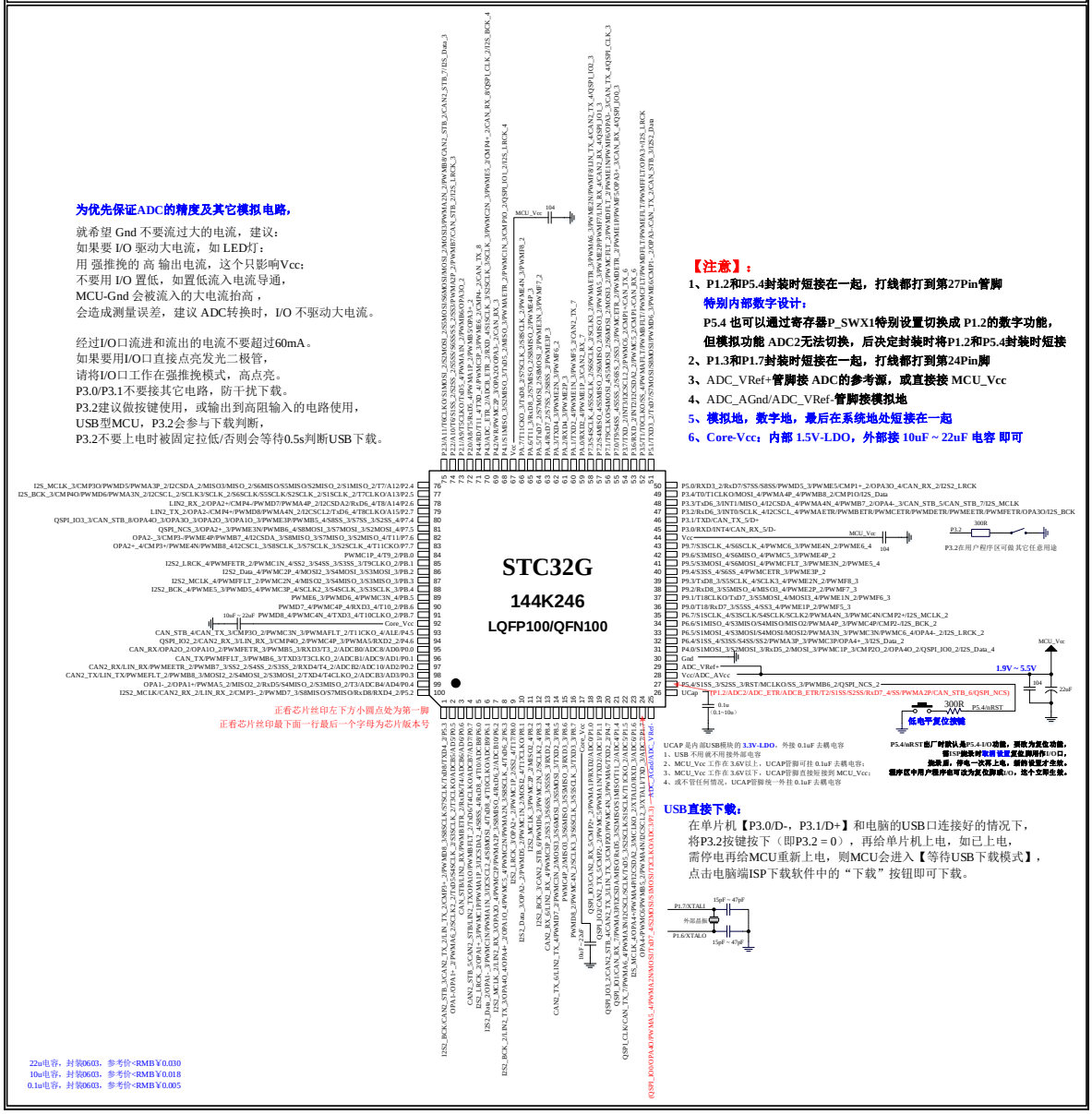


【应用场景一：从本工具给目标系统 自动 停电/上电，供电】

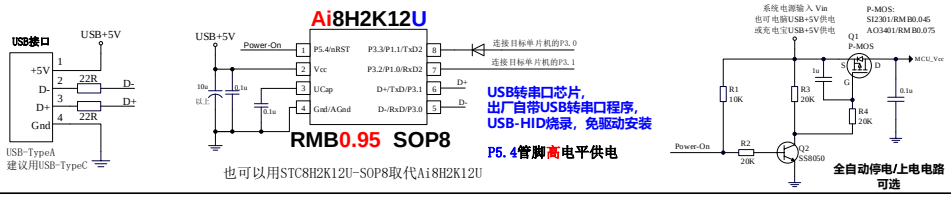
点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0.5秒/再自动供电，
数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0.5秒/再自动供电给目标系统工作。

【应用场景二：不从本工具给目标系统供电】

1、点击 电脑端 ISP 软件的【下载/编程】按钮
2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，
下载编程进行中，数秒后提示下载编程成功，目标MCU会自动复位到用户程序区自动跑用户程序。
部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P5.4/nRST变成复位脚



3.1.13 通用 USB 转串口芯片全自动停电/上电烧录，串口仿真，5V 原理图

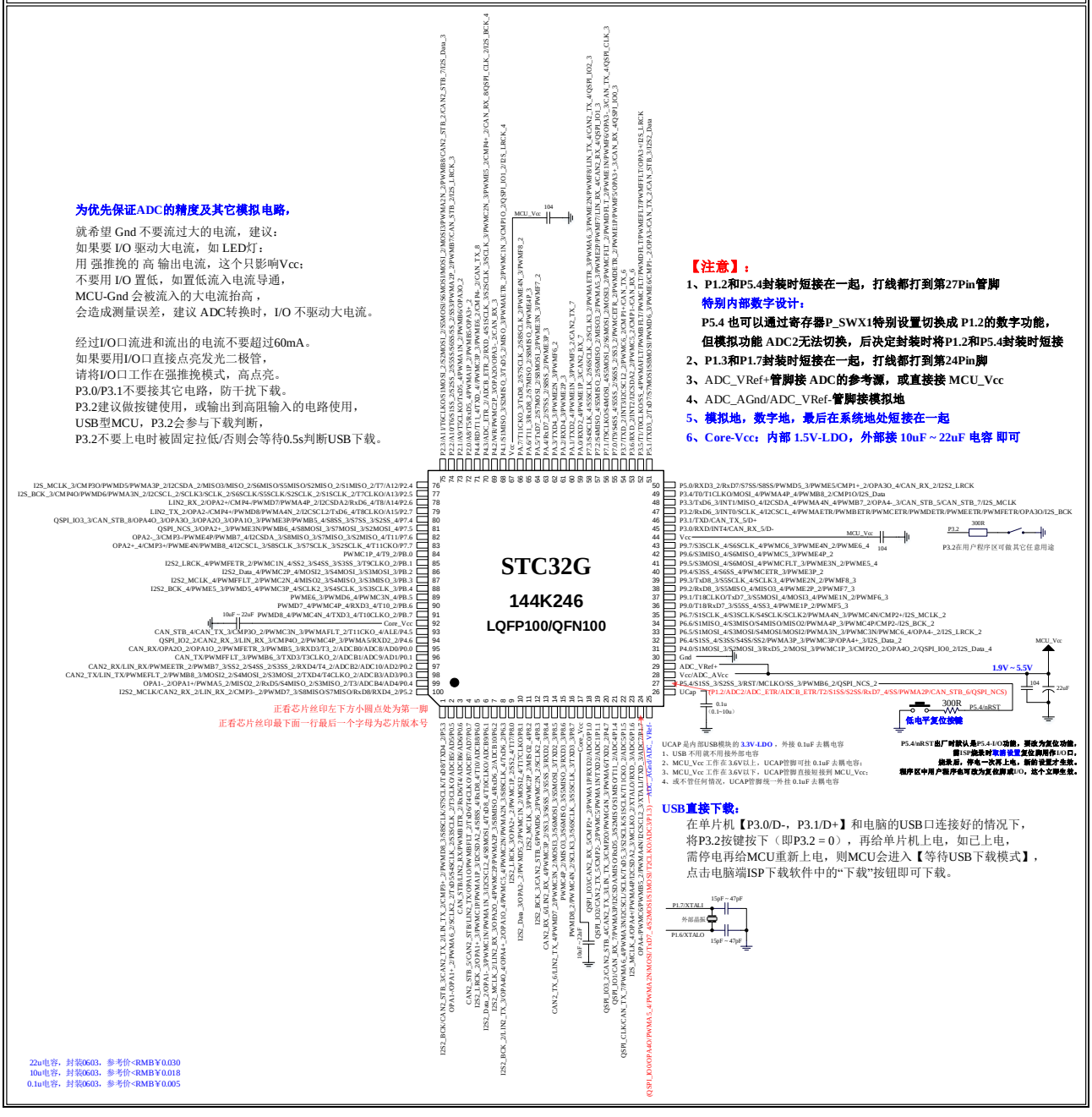


【应用场景一：从本工具给目标系统 自动 停电/上电，供电】

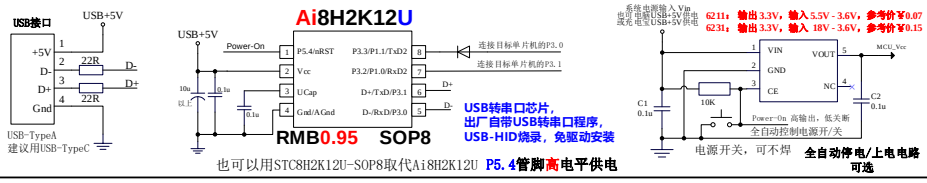
点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0.5秒/再自动供电，
数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0.5秒/再自动供电给目标系统工作。

【应用场景二：不从本工具给目标系统供电】

1、点击 电脑端 ISP 软件的【下载/编程】按钮
2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，
下载编程进行中，数秒后提示下载编程成功，目标MCU会自动复位到用户程序区自动跑用户程序。
部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P5.4/nRST变成复位脚



3.1.14 通用 USB 转串口芯片全自动停电/上电烧录，串口仿真，3.3V 原理图

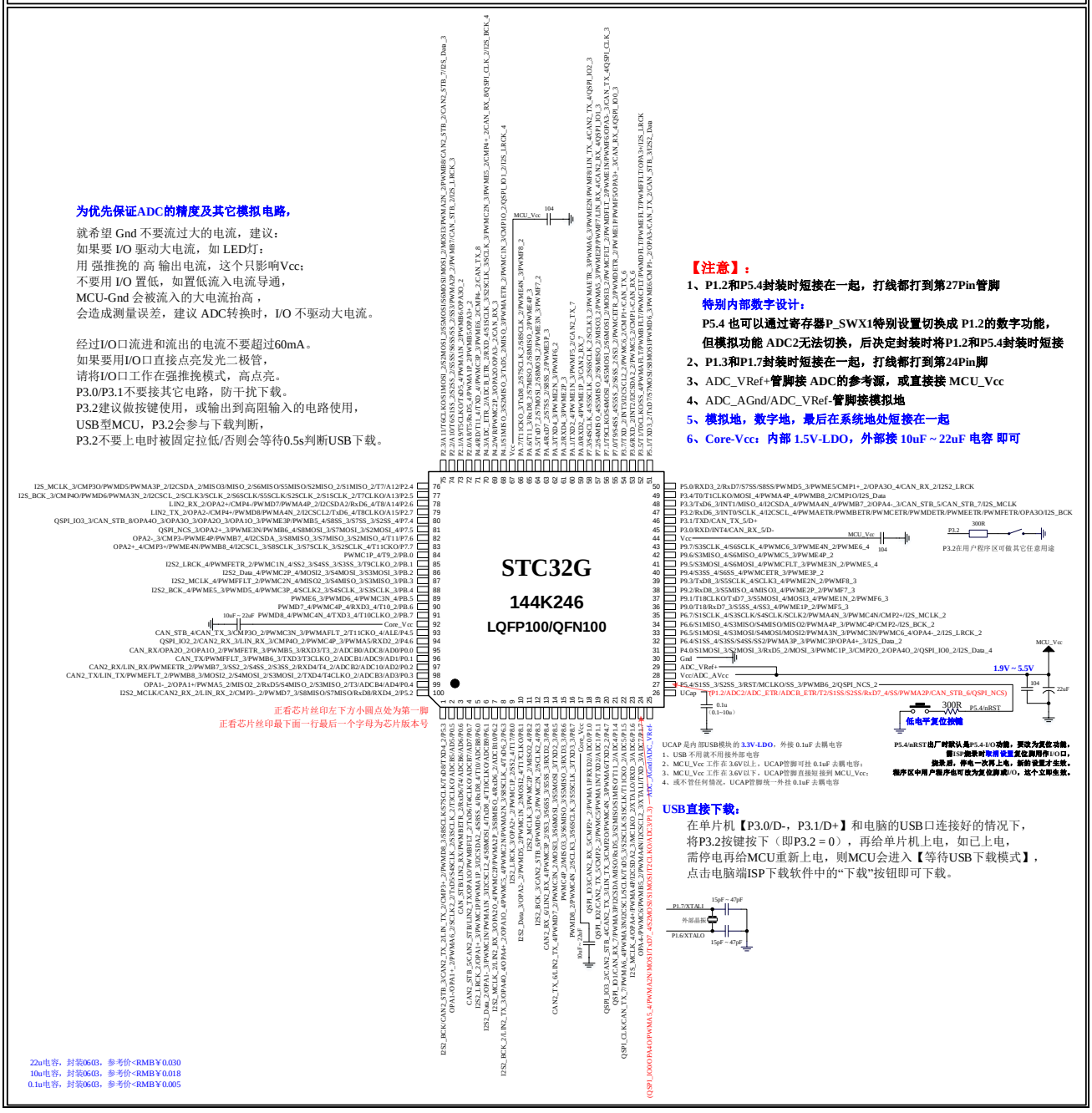


【应用场景一：从本工具给目标系统 自动 停电/上电，供电】

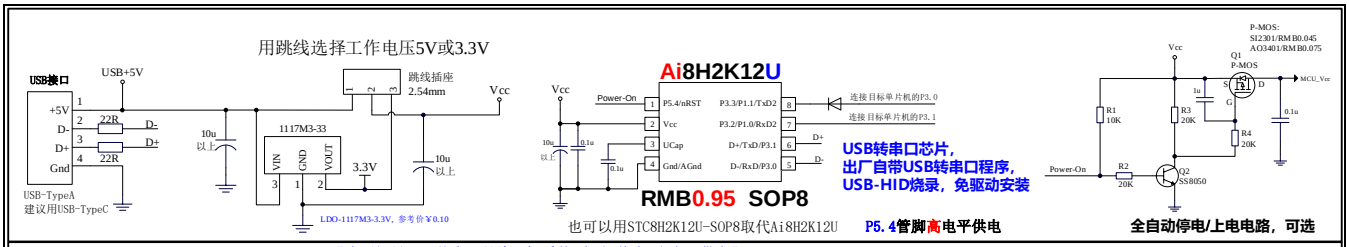
点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0.5秒/再自动供电，
数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0.5秒/再自动供电给目标系统工作。

【应用场景二：不从本工具给目标系统供电】

1、点击 电脑端 ISP 软件的【下载/编程】按钮
2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，
下载编程进行中，数秒后提示下载编程成功，目标MCU会自动复位到用户程序区自动跑用户程序。
部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P5.4/nRST变成复位脚



3.1.15 USB 转串口芯片全自动停电/上电烧录/仿真/通信，5V/3.3V 跳线选择

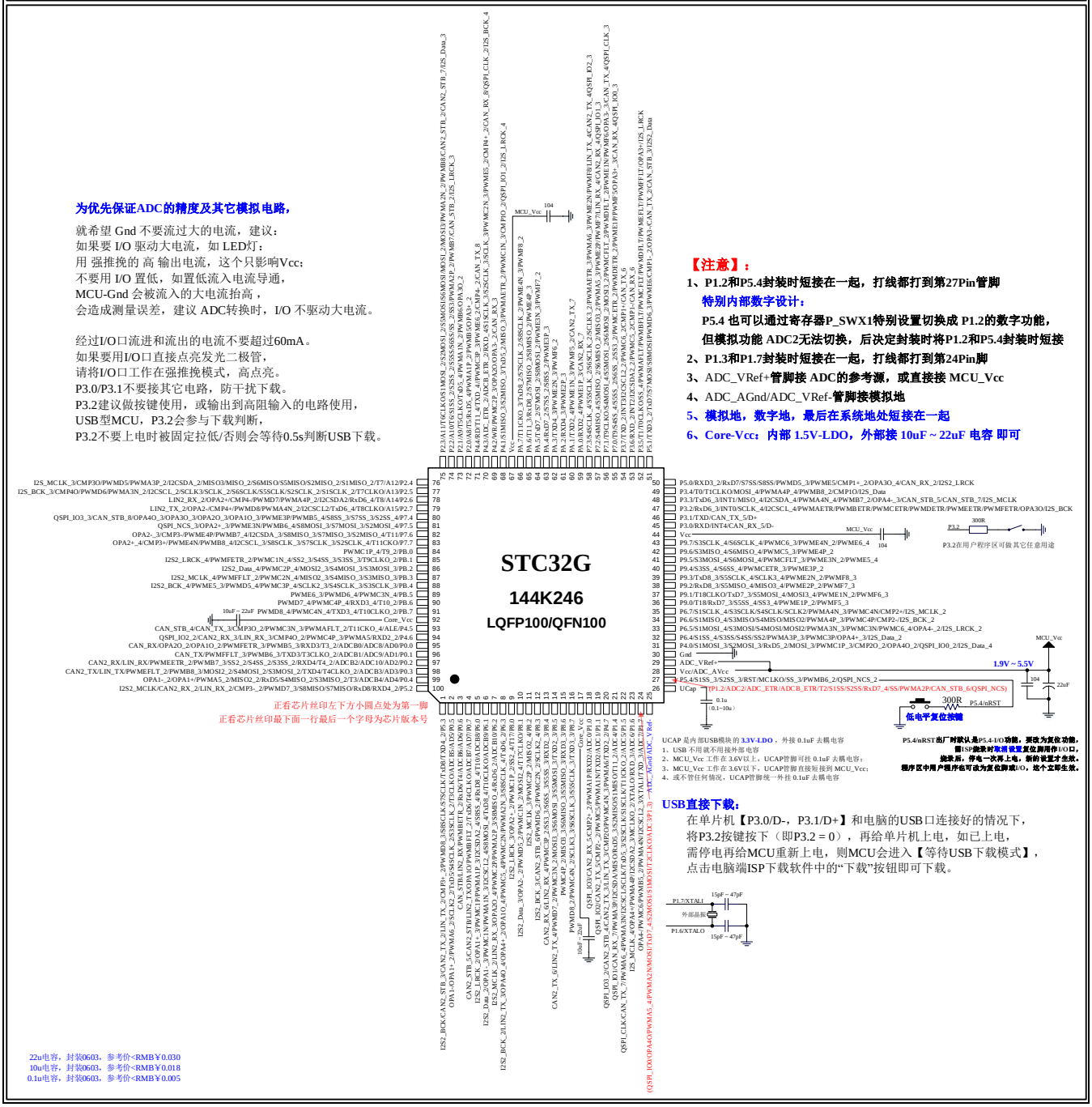


【应用场景一：从本工具给目标系统 自动 停电/上电，供电】

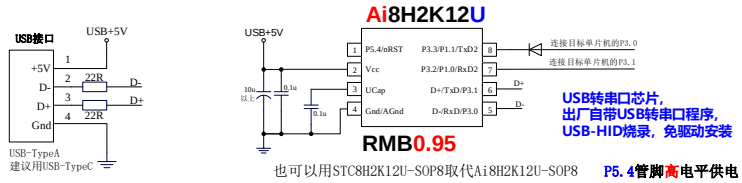
点击 电脑端 ISP 软件的【下载/编程】按钮，工具会 自动 给目标系统停电0.5秒/再自动供电，
数秒后提示下载编程成功，工具默认会再 自动 给目标系统停电0.5秒/再自动供电给目标系统工作。

【应用场景二：不从本工具给目标系统供电】

1、点击 电脑端 ISP 软件的【下载/编程】按钮
2、给目标系统上电，或者重新上电(如果在点击【下载/编程】按钮前已上电，则需要停电重新上电)，
下载编程进行中，数秒后提示下载编程成功，目标MCU会自动复位到用户程序区自动跑用户程序。
部分不常用设置，需要停电上电一次才生效，如改变 EEPROM大小，P5.4/nRST变成复位脚



3.1.16 USB 转串口芯片进行烧录/串口仿真，手动停电/上电，5V/3.3V 原理图



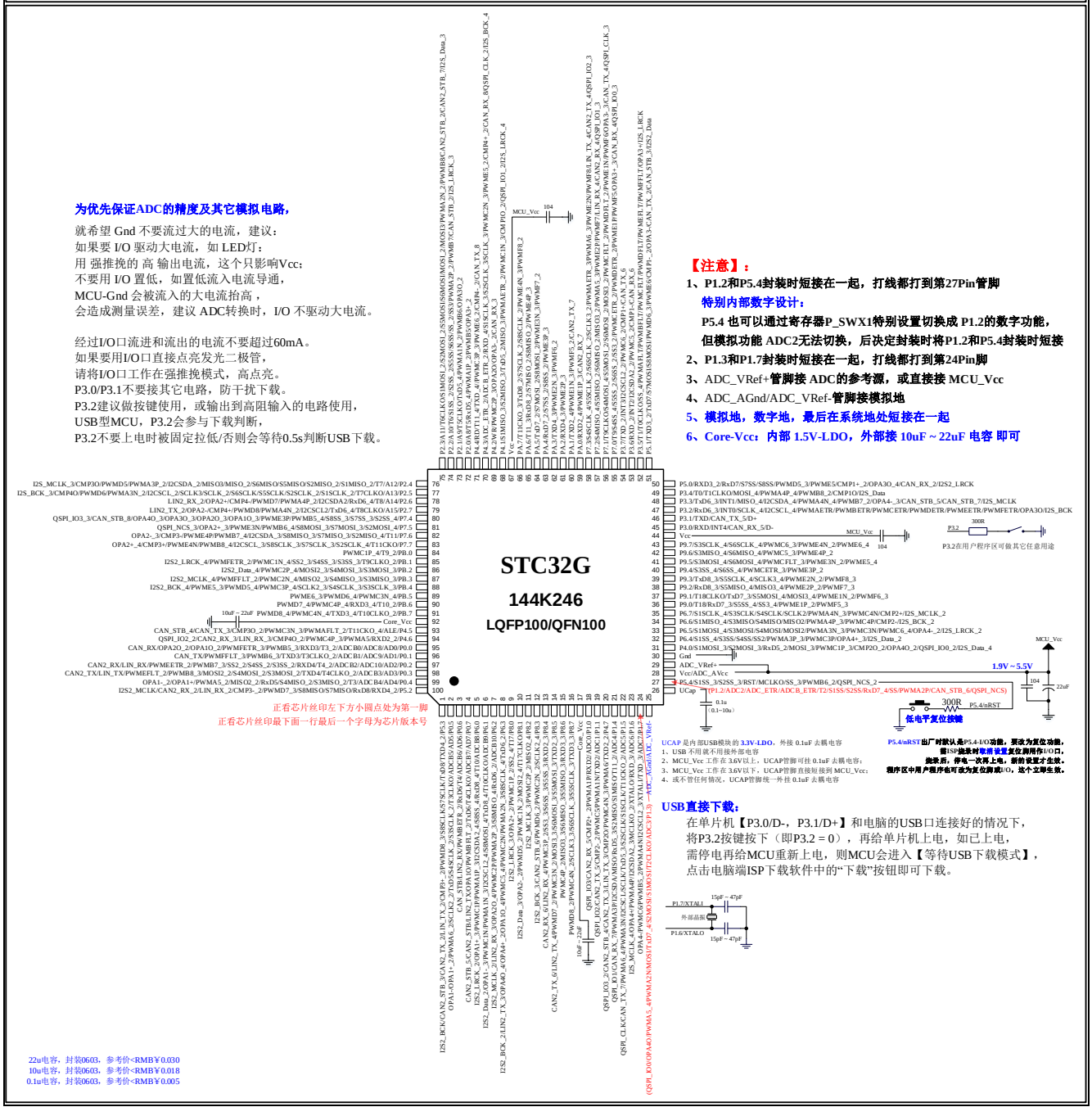
【ISP下载/编程/烧录，操作步骤】

1、点击电脑端 ISP 软件的【下载/编程】按钮

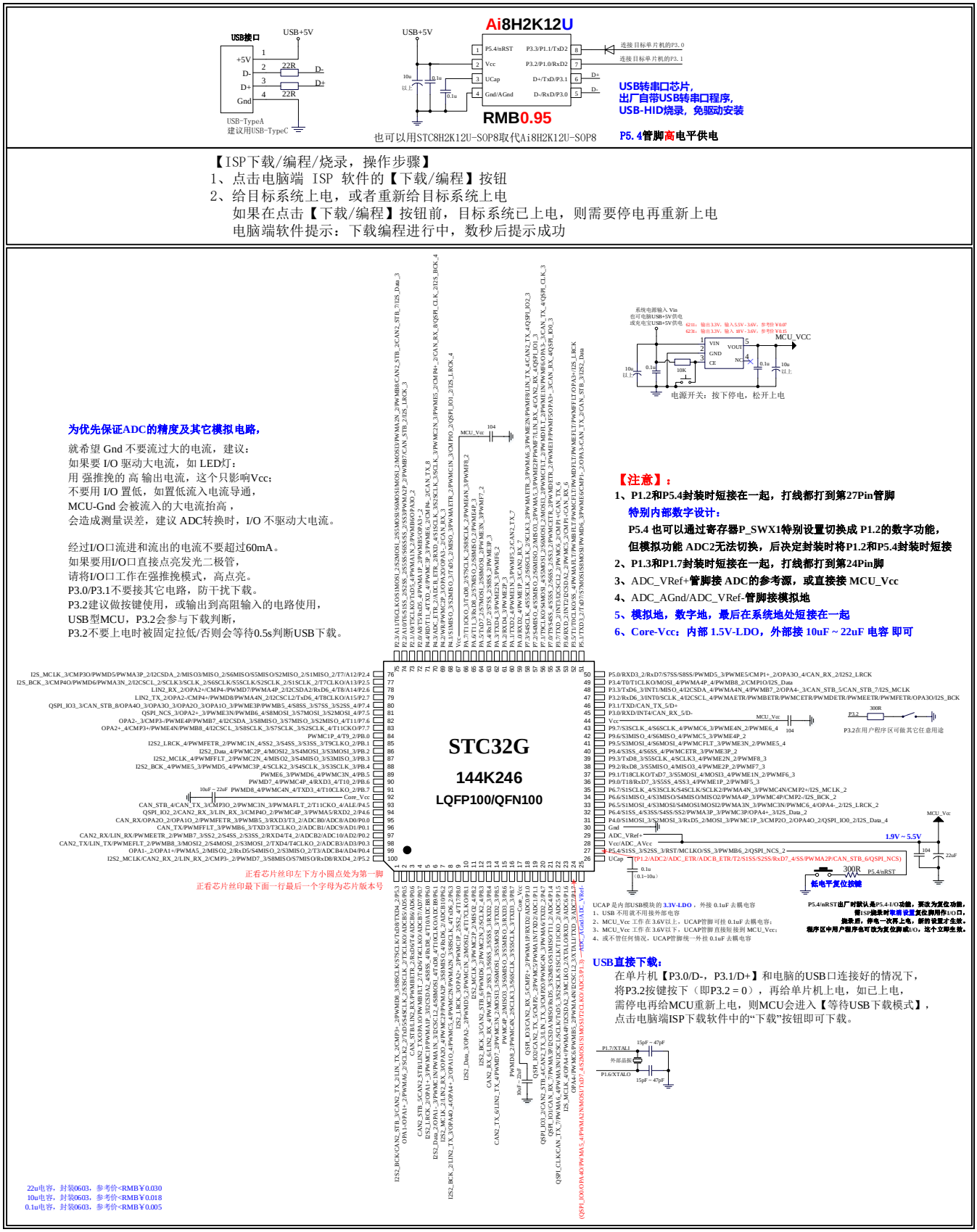
2、给目标系统上电，或者重新给目标系统上电

如果在点击【下载/编程】按钮前，目标系统已上电，则需要停电再重新上电

电脑端软件提示：下载编程进行中，数秒后提示成功



3.1.17 USB 转串口芯片进行烧录，串口仿真，手动停电/上电，3.3V 原理图

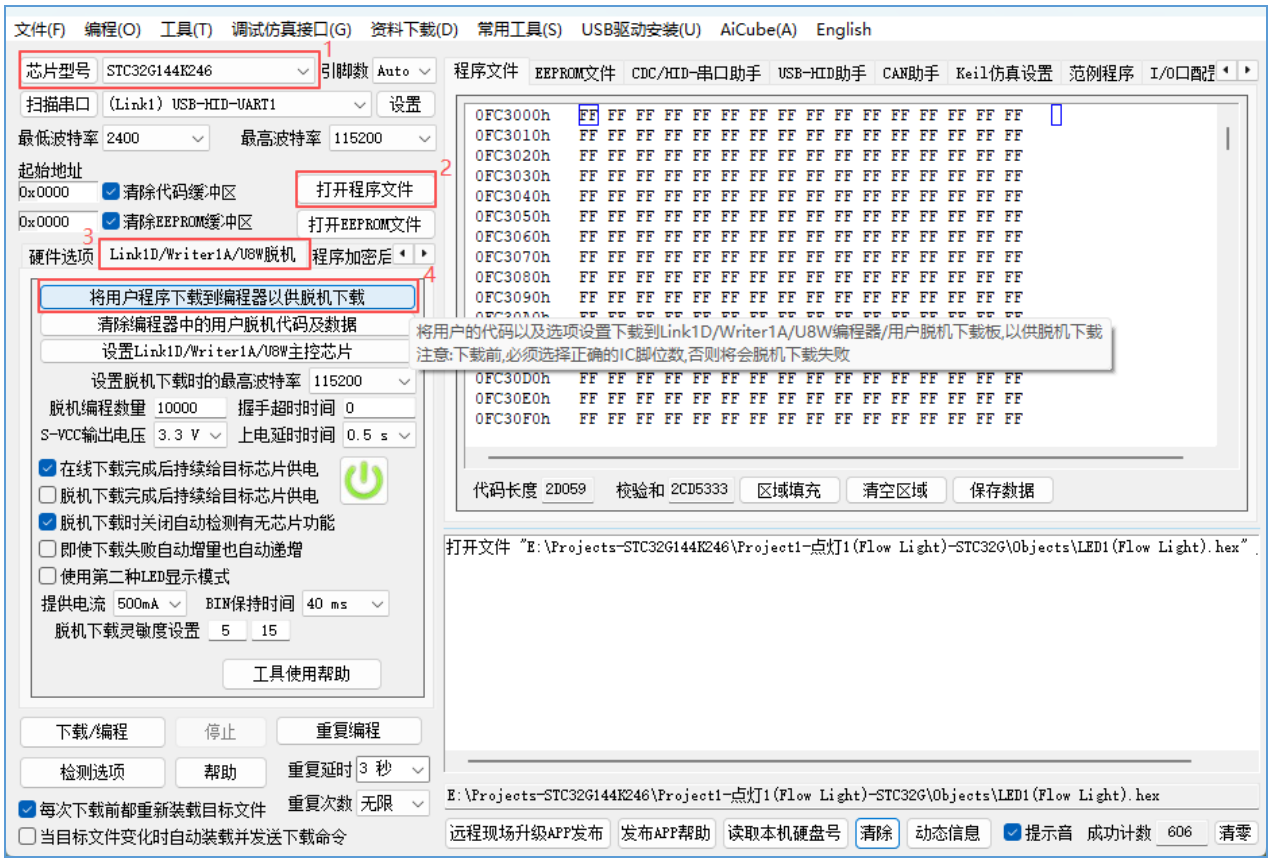


3.1.18 USB-Link1D 支持 脱机下载 说明

脱机下载是指脱离电脑主机进行下载的一种方法，一般是使用下载控制芯片（又称脱机下载母片）进行控制。USB-Link1D 工具除了支持在线 ISP 下载，还支持脱机下载。USB-Link1D 工具使用外部的 22.1184MHz 晶振，可保证对目标芯片进行在线或脱机下载时，校准频率的精度。用户可将代码下载到 USB-Link1D 工具中，就可实现脱机下载。USB-Link1D 主控芯片如内部存储空间不够时，会将部分被下载的用户程序用加密的方式加密放部分到片外串行 Flash。

先将 USB-Link1D 工具使用 USB 线连接到电脑，然后按照下面的步骤进行脱机下载：

1、选择目标芯片的型号
2、打开需要下载的文件
3、设置硬件选项
4、进入“USB-Link1D 脱机”页面，点击“将用户程序下载到编程器以供脱机下载“按钮，即可将用户代码下载到 USB-Linik1D 工具中。下载完成后便使用 USB-Link1D 工具对目标芯片进行脱机下载了



3.1.19 USB-Link1D 支持 脱机下载，如何免烧录环节

大批量生产，如何省去专门的烧录人员，如何无烧录环节

大批量生产，你在将由 STC 的 MCU 作为主控芯片的控制板组装到设备里面之前，在你将 STC MCU 贴片到你的控制板完成之后，你必须测试你的控制板的好坏。不要说 100%，直通无问题，那是抬杠，不是搞生产，只要生产，就会虚焊，短路，部分元件贴错，部分元件采购错。

所以在贴片回来后，组装到外壳里面之前，你必须要测试，你的含有 STC MCU 控制板的好坏，好的去组装，坏的去维修抢救。

控制板的测试/不是烧录！

控制板的测试环节必须有，但烧录环节可以省！

大批量生产，必须要有方便控制板测试的测试架/下面接上我们的脱机烧录工具：

USB-Link1D / U8W-Mini / U8W，还要接上其他控制部分！

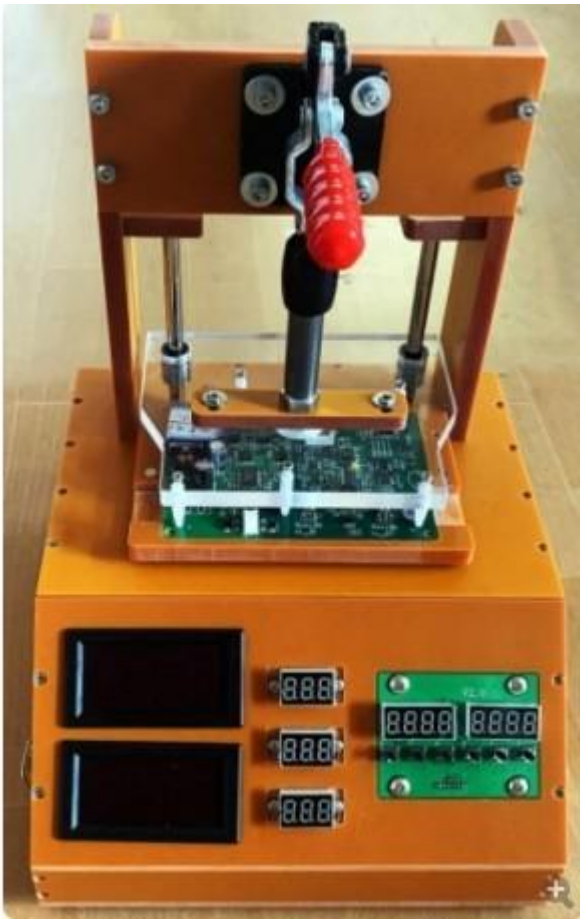
1、通过 USER-VCC、P3.0、P3.1、GND 连接，要工人每次都开电源

2、通过 S-VCC、P3.0、P3.1、GND 连接，不要你开电源，STC 的脱机工具给你自动供电

外面帮你做一个测试架的成本 500 元以下，就是有机玻璃，夹具，顶针。

1 个测试你控制板是否正常的工人管理 2 - 3 个 测试架



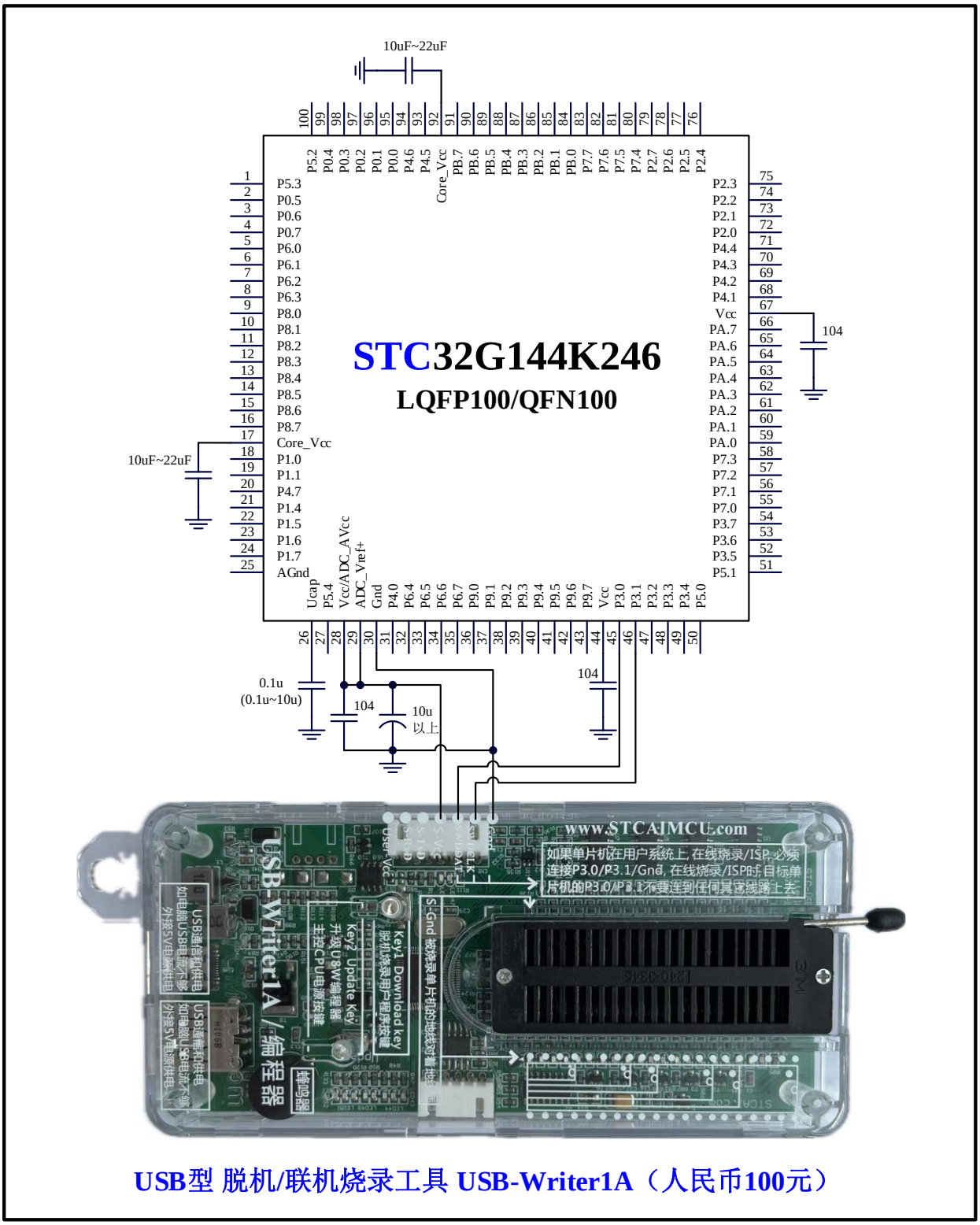


操作流程：

1、将你的 MCU 控制板 卡到测试架 1 上

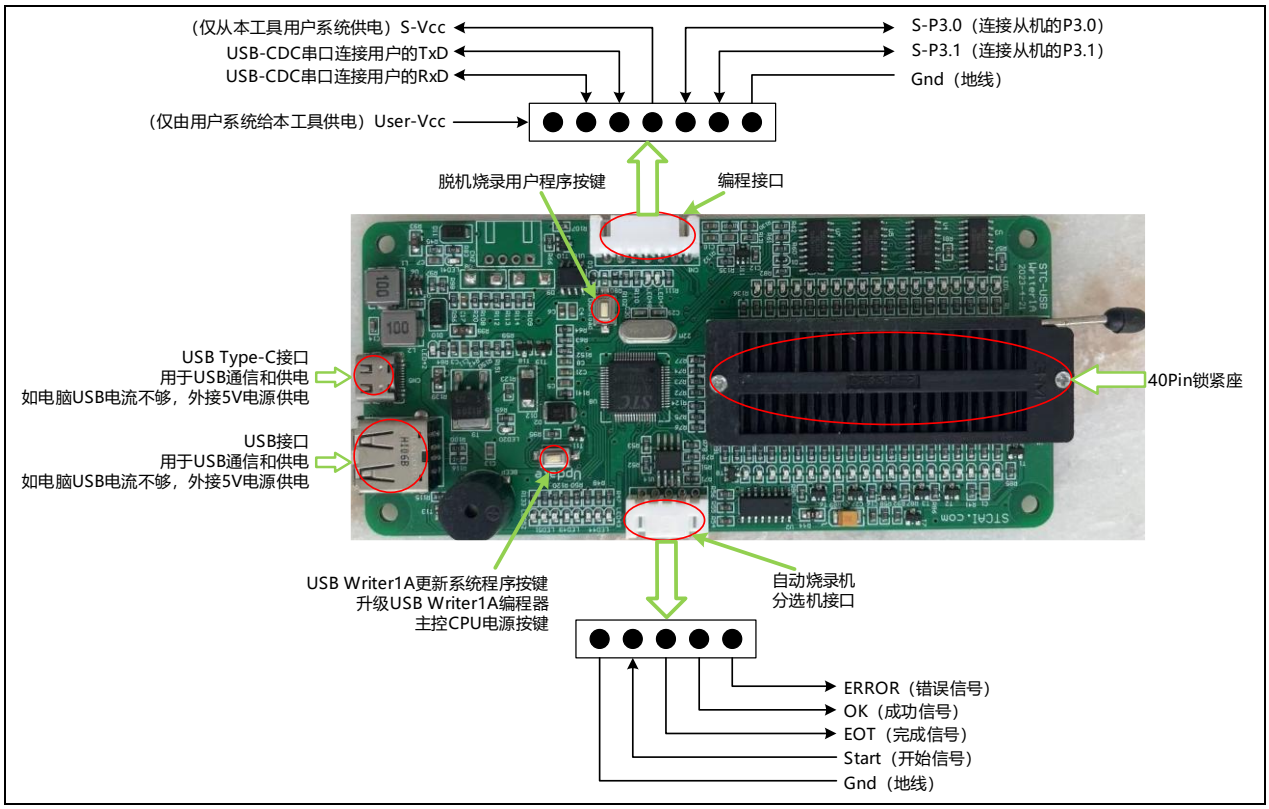
2、将你的 MCU 控制板 卡到测试架 2 上，测试架 1 上的程序已烧录完成/感觉不到烧录时间
3、测试 测试架 1 上的 MCU 主控板功能是否正常，正常放到正常区，不正常，放到不正常区
4、给测试架 1 卡上新的未测试的无程序的控制板
5、测试 测试架 2 上的未测试控制板/程序不知何时早就不知不觉的烧好了，换新的未测试未烧录的控制板
6、循环步骤 3 到步骤 5
========不需要安排烧录人员

3.1.20 USB-Writer1A 编程器/烧录器 • 支持 • 插在 • 锁紧座上 • 烧录



USB型 脱机/联机烧录工具 USB-Writer1A（人民币100元）

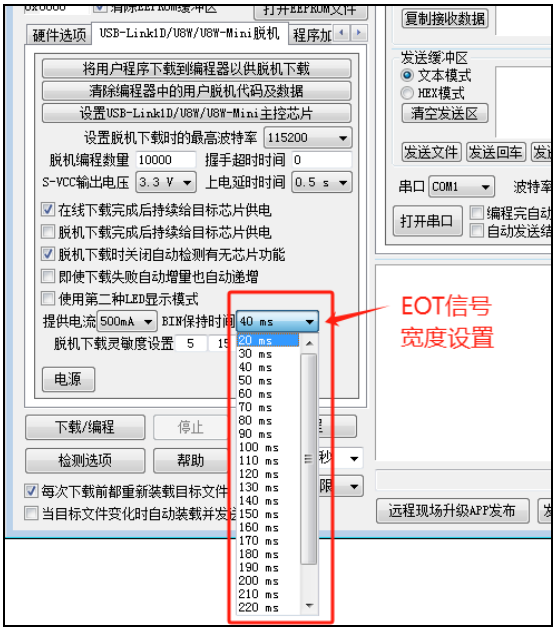
3.1.21 USB-Writer1A 支持 自动烧录机，通信协议和接口



自动烧录接口（分选机自动控制接口）协议：

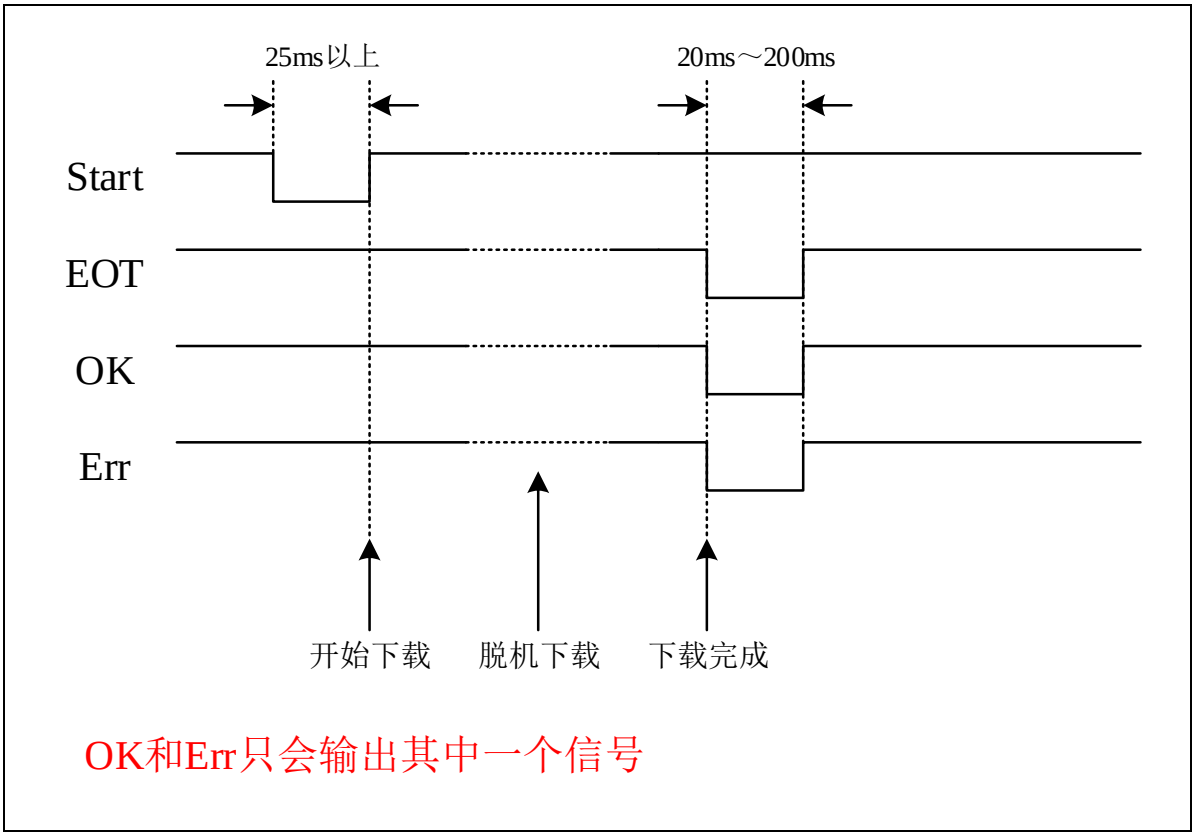
Start： 开始信号输入脚。从外部输入低电平信号触发开始脱机烧录，低电平必须维持 25 毫秒以上

EOT： 烧录完成信号输出脚。脱机烧录完成后，工具输出 20ms～250ms 的低电平 EOT 信号。电平宽度如下图所示的地方进行设置

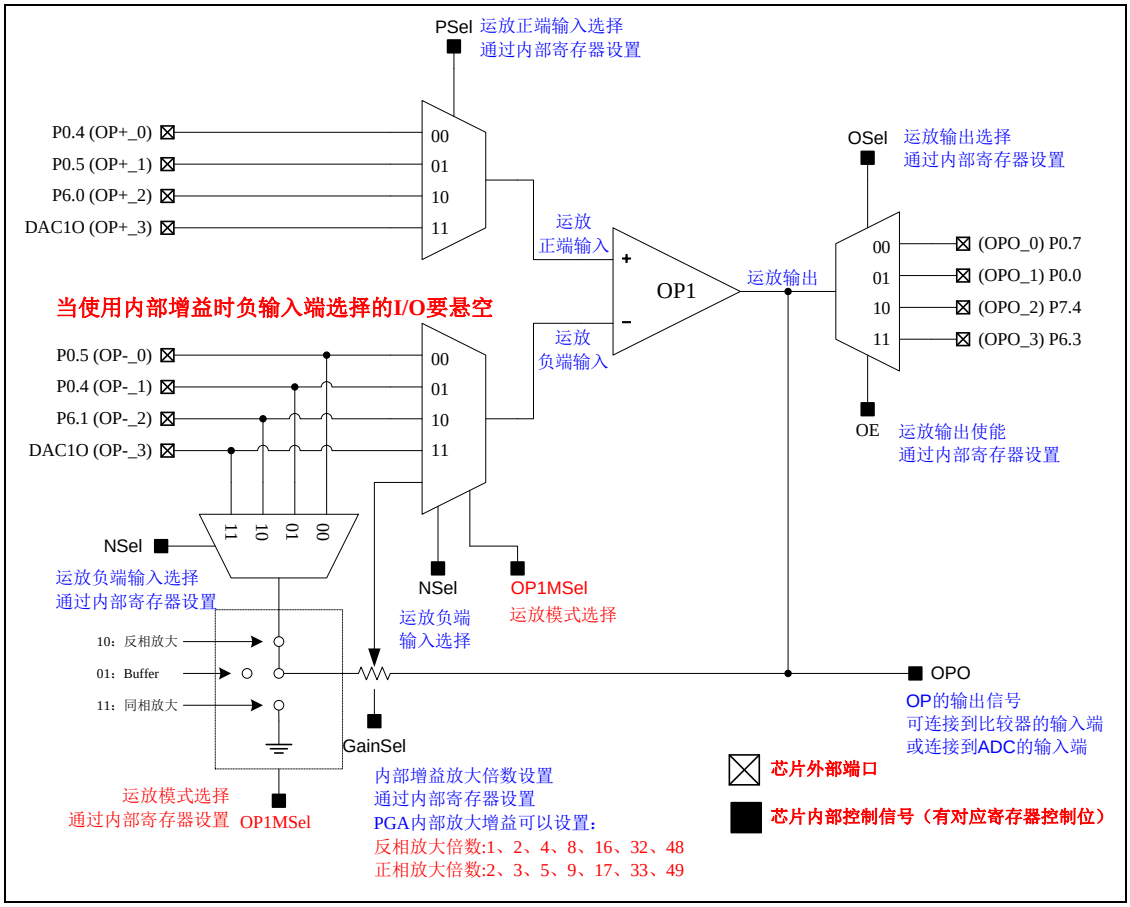


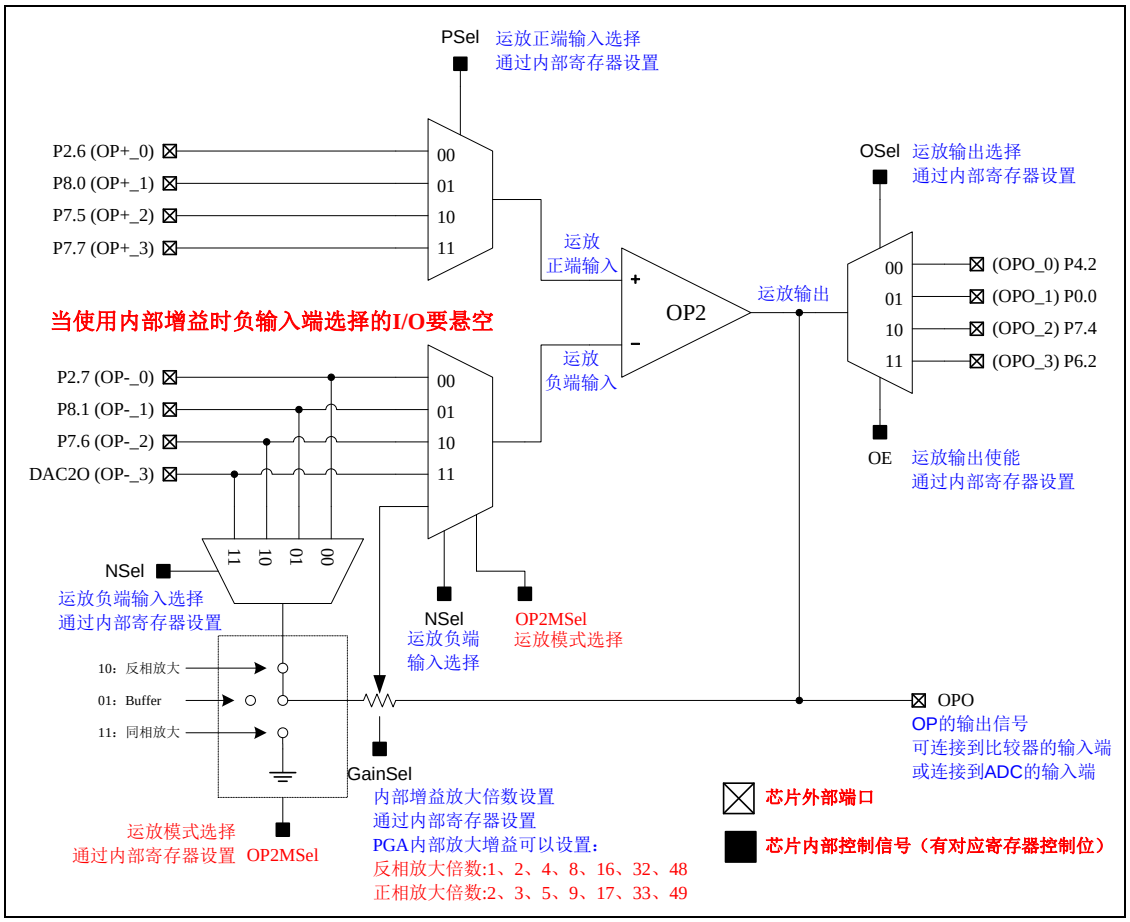
OK： 良品信号输出脚。下载成功后工具从 OK 脚输出低电平信号，信号与 EOT 信号同步。

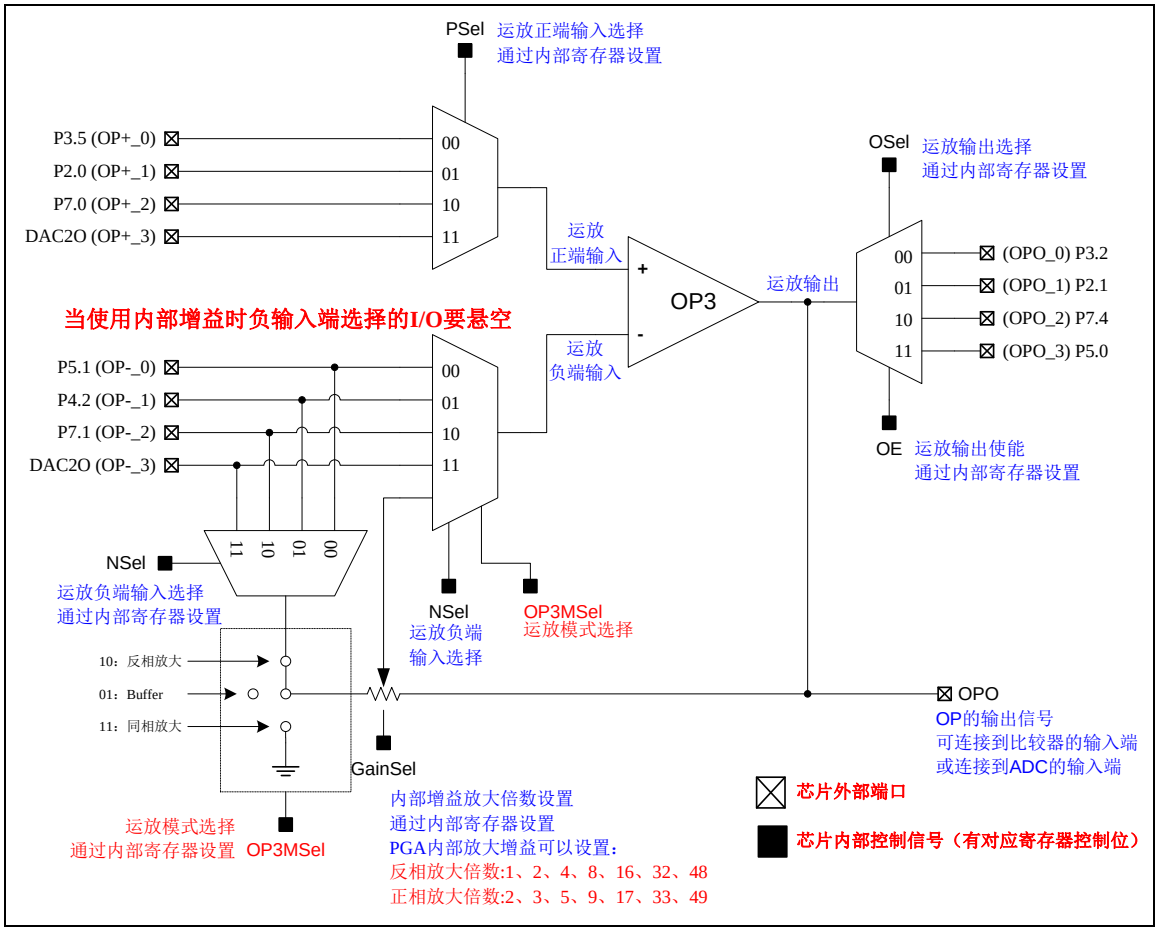
Err： 不良品信号输出脚。若下载失败，工具从 ERR 脚输出输出低电平信号，信号与 EOT 完成信号同步。

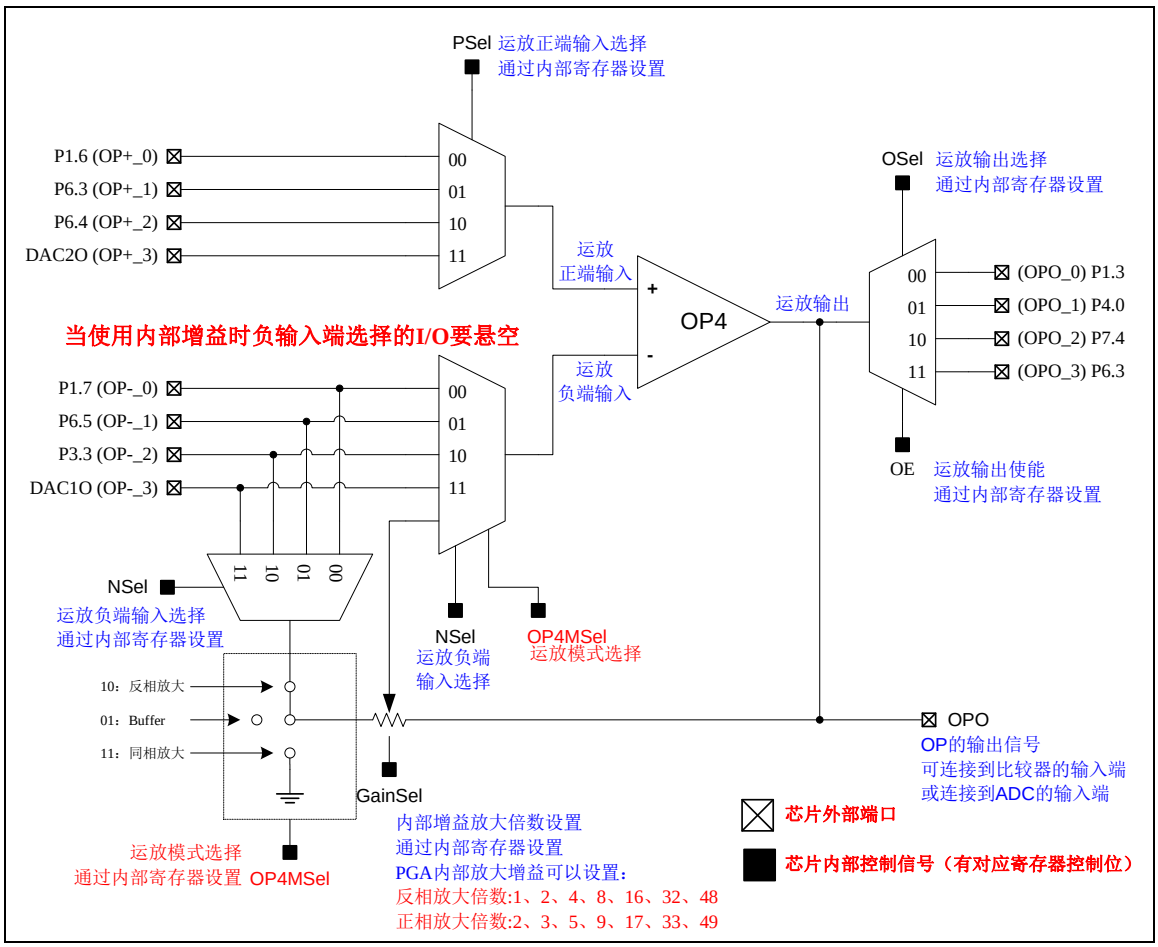


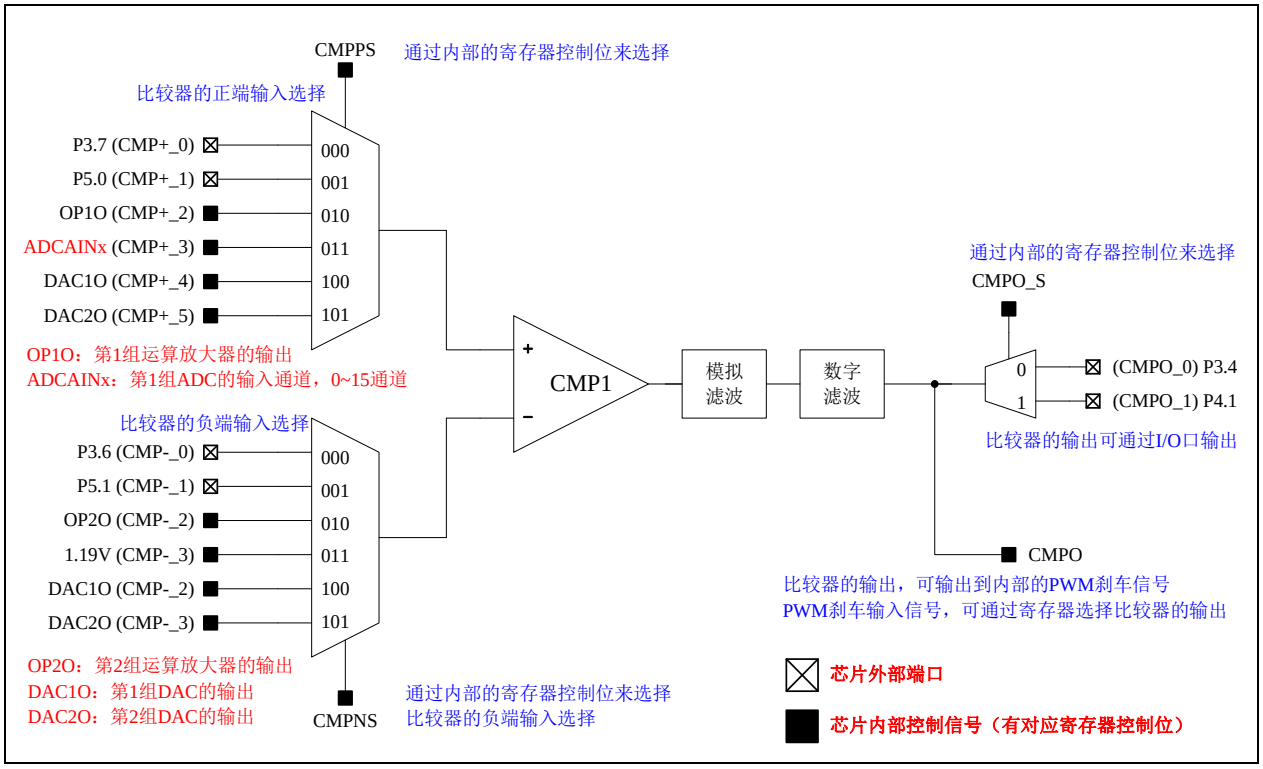
3.2 两组独立 DAC、4 组运放、4 组比较器结构及应用，整理中

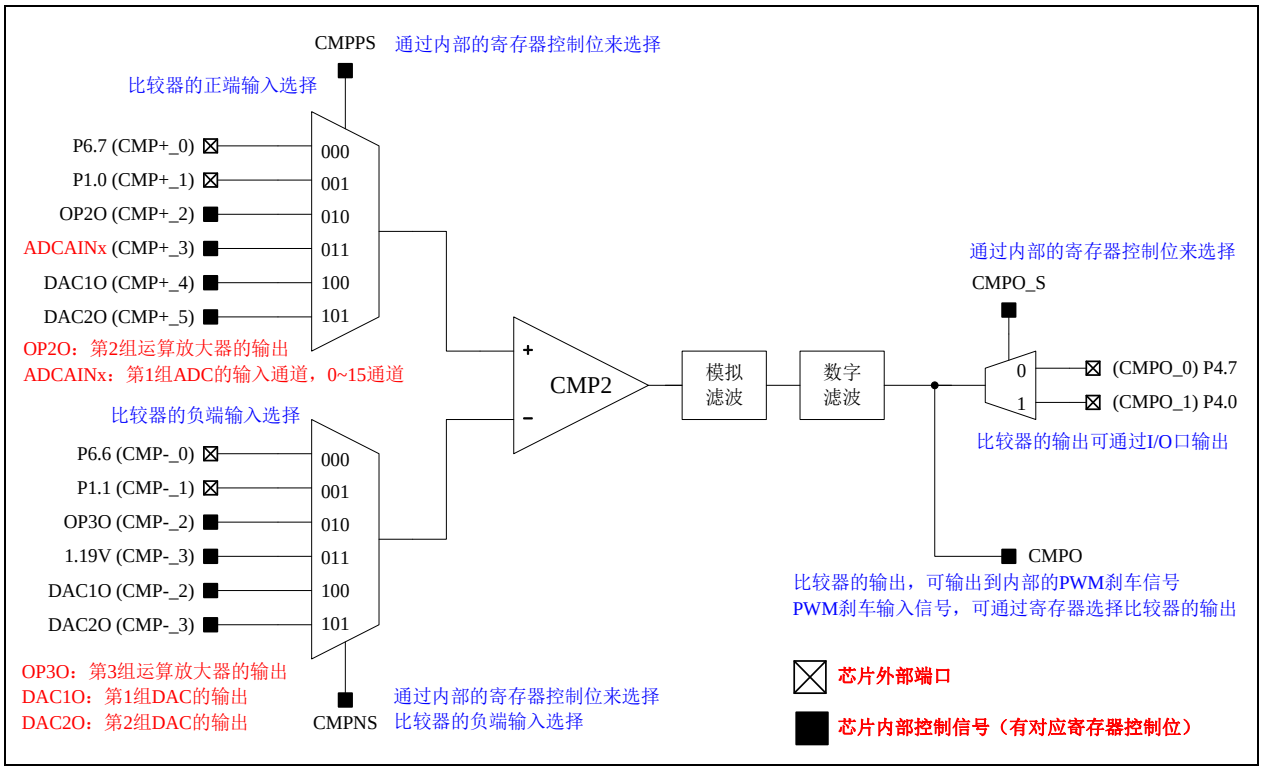


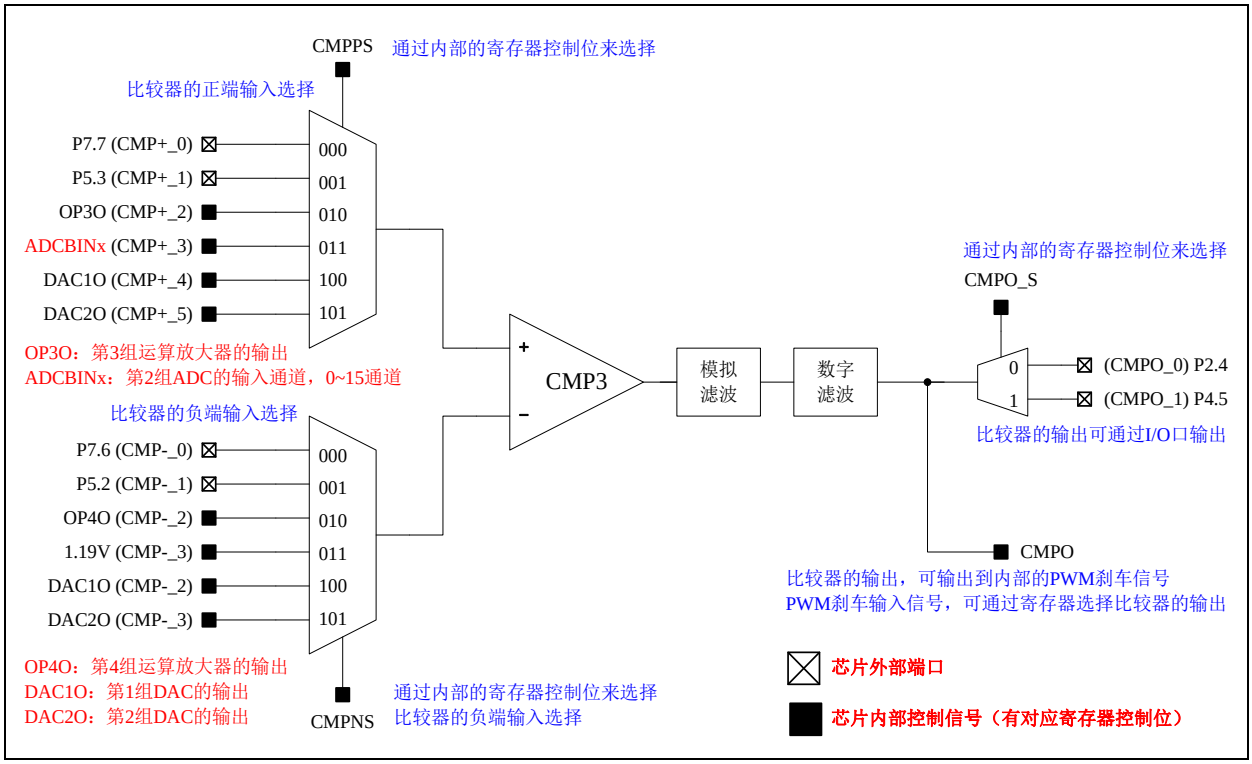


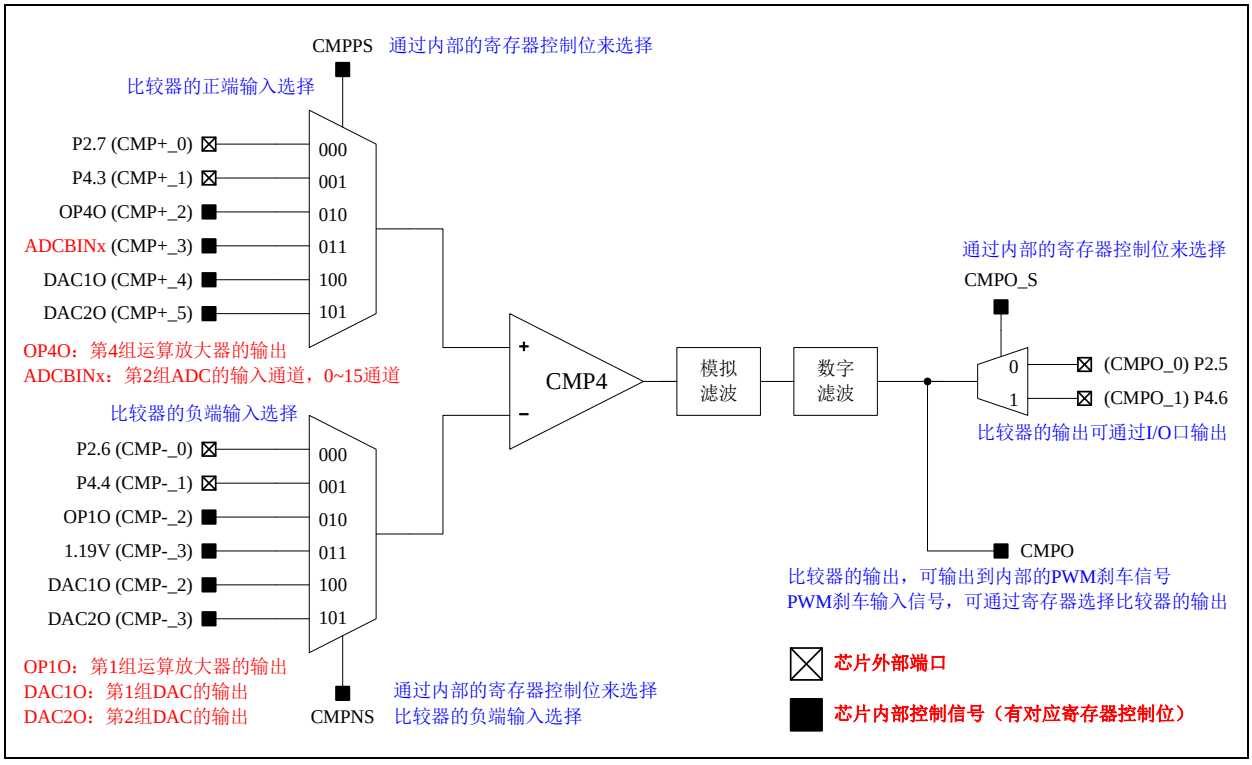

PSel 运放正端输入选择
通过内部寄存器设置
P3.5 (OP+_0)
P2.0 (OP+_1)
P7.0 (OP+_2)
DAC2O (OP+_3)
00
01
10
11
运放
正端输入
+
OP3
-
运放输出
OSel 运放输出选择
通过内部寄存器设置
(OPO_0) P3.2
(OPO_1) P2.1
(OPO_2) P7.4
(OPO_3) P5.0
OE 运放输出使能
通过内部寄存器设置
当使用内部增益时负输入端选择的I/O要悬空
P5.1 (OP-_0)
P4.2 (OP-_1)
P7.1 (OP-_2)
DAC2O (OP-_3)
运放
负端输入
NSel
运放负端输入选择
通过内部寄存器设置
NSel
运放负端
输入选择
OP3MSel
运放模式选择
10：反相放大
01：Buffer
11：同相放大
GainSel
OPO
OP的输出信号
可连接到比较器的输入端
或连接到ADC的输入端
运放模式选择
通过内部寄存器设置 OP3MSel
内部增益放大倍数设置
通过内部寄存器设置
PGA内部放大增益可以设置：
反相放大倍数:1、2、4、8、16、32、48
正相放大倍数:2、3、5、9、17、33、49
芯片外部端口
芯片内部控制信号（有对应寄存器控制位）

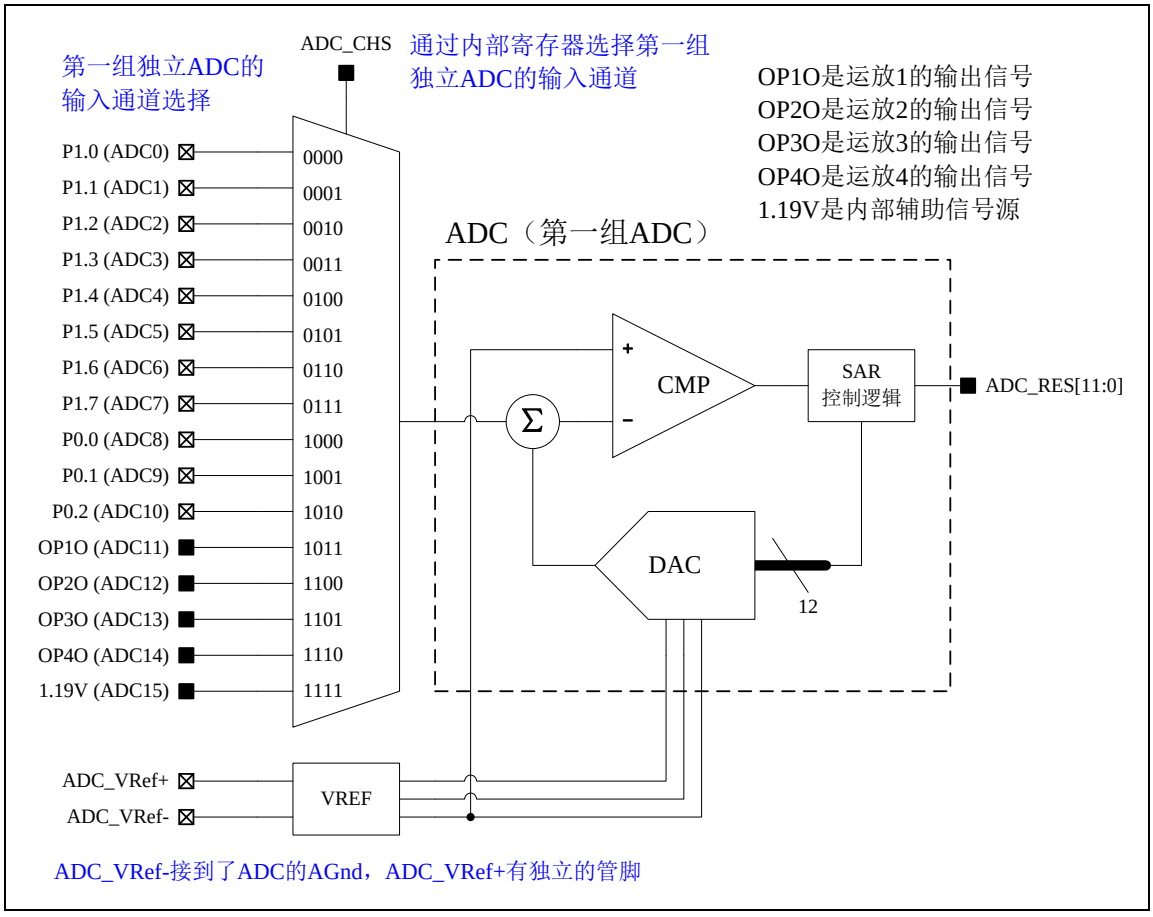

PSel 运放正端输入选择
通过内部寄存器设置
P1.6 (OP+_0)
P6.3 (OP+_1)
P6.4 (OP+_2)
DAC2O (OP+_3)
00
01
10
11
运放
正端输入
+
OP4
-
运放输出
OSel 运放输出选择
通过内部寄存器设置
(OPO_0) P1.3
(OPO_1) P4.0
(OPO_2) P7.4
(OPO_3) P6.3
OE 运放输出使能
通过内部寄存器设置
当使用内部增益时负输入端选择的I/O要悬空
P1.7 (OP-_0)
P6.5 (OP-_1)
P3.3 (OP-_2)
DAC1O (OP-_3)
运放
负端输入
NSel
运放负端输入选择
通过内部寄存器设置
NSel
运放负端
输入选择
OP4MSel
运放模式选择
10：反相放大
01：Buffer
11：同相放大
GainSel
OPO
OP的输出信号
可连接到比较器的输入端
或连接到ADC的输入端
运放模式选择
通过内部寄存器设置 OP4MSel
内部增益放大倍数设置
通过内部寄存器设置
PGA内部放大增益可以设置：
反相放大倍数:1、2、4、8、16、32、48
正相放大倍数:2、3、5、9、17、33、49
芯片外部端口
芯片内部控制信号（有对应寄存器控制位）

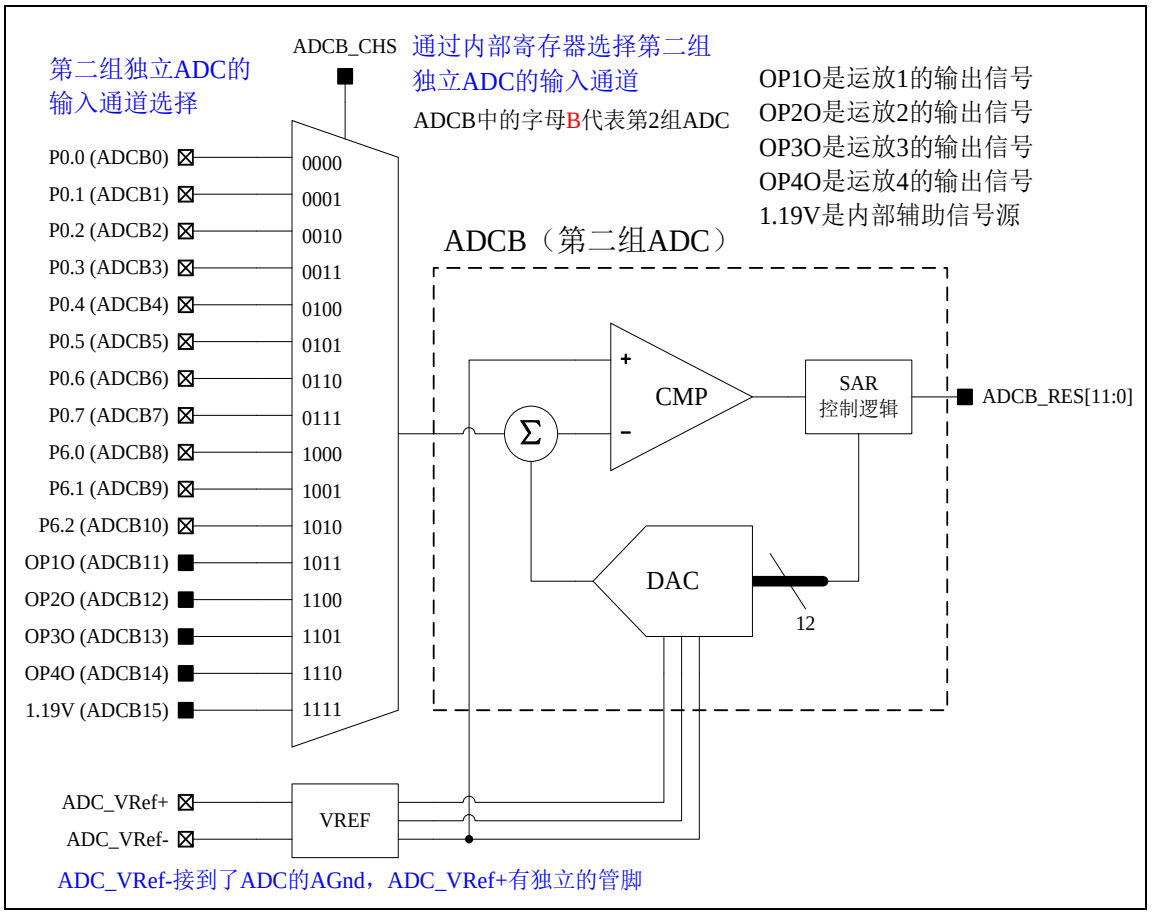

CMPPS
通过内部的寄存器控制位来选择
比较器的正端输入选择
P3.7 (CMP+_0)
P5.0 (CMP+_1)
OP1O (CMP+_2)
ADCAINx (CMP+_3)
DAC1O (CMP+_4)
DAC2O (CMP+_5)
000
001
010
011
100
101
OP1O：第1组运算放大器的输出
ADCAINx：第1组ADC的输入通道，0~15通道
比较器的负端输入选择
P3.6 (CMP-_0)
P5.1 (CMP-_1)
OP2O (CMP-_2)
1.19V (CMP-_3)
DAC1O (CMP-_2)
DAC2O (CMP-_3)
OP2O：第2组运算放大器的输出
DAC1O：第1组DAC的输出
DAC2O：第2组DAC的输出
CMPNS
通过内部的寄存器控制位来选择
比较器的负端输入选择
CMP1
模拟滤波
数字滤波
通过内部的寄存器控制位来选择
CMPO_S
0
1
(CMPO_0) P3.4
(CMPO_1) P4.1
比较器的输出可通过I/O口输出
CMPO
比较器的输出，可输出到内部的PWM刹车信号
PWM刹车输入信号，可通过寄存器选择比较器的输出
芯片外部端口
芯片内部控制信号（有对应寄存器控制位）


CMPPS
通过内部的寄存器控制位来选择
比较器的正端输入选择
P6.7 (CMP+_0)
P1.0 (CMP+_1)
OP2O (CMP+_2)
ADCAINx (CMP+_3)
DAC1O (CMP+_4)
DAC2O (CMP+_5)
000
001
010
011
100
101
OP2O：第2组运算放大器的输出
ADCAINx：第1组ADC的输入通道，0~15通道
比较器的负端输入选择
P6.6 (CMP-_0)
P1.1 (CMP-_1)
OP3O (CMP-_2)
1.19V (CMP-_3)
DAC1O (CMP-_2)
DAC2O (CMP-_3)
OP3O：第3组运算放大器的输出
DAC1O：第1组DAC的输出
DAC2O：第2组DAC的输出
CMPNS
通过内部的寄存器控制位来选择
比较器的负端输入选择
CMP2
模拟滤波
数字滤波
通过内部的寄存器控制位来选择
CMPO_S
0
1
(CMPO_0) P4.7
(CMPO_1) P4.0
比较器的输出可通过I/O口输出
CMPO
比较器的输出，可输出到内部的PWM刹车信号
PWM刹车输入信号，可通过寄存器选择比较器的输出
芯片外部端口
芯片内部控制信号（有对应寄存器控制位）


CMPPS 通过内部的寄存器控制位来选择
比较器的正端输入选择
P7.7 (CMP+_0)
P5.3 (CMP+_1)
OP3O (CMP+_2)
ADCBINx (CMP+_3)
DAC1O (CMP+_4)
DAC2O (CMP+_5)
000
001
010
011
100
101
OP3O：第3组运算放大器的输出
ADCBINx：第2组ADC的输入通道，0~15通道
比较器的负端输入选择
P7.6 (CMP-_0)
P5.2 (CMP-_1)
OP4O (CMP-_2)
1.19V (CMP-_3)
DAC1O (CMP-_2)
DAC2O (CMP-_3)
OP4O：第4组运算放大器的输出
DAC1O：第1组DAC的输出
DAC2O：第2组DAC的输出
CMPNS
通过内部的寄存器控制位来选择
比较器的负端输入选择
CMP3
模拟滤波
数字滤波
通过内部的寄存器控制位来选择
CMPO_S
0
1
(CMPO_0) P2.4
(CMPO_1) P4.5
比较器的输出可通过I/O口输出
CMPO
比较器的输出，可输出到内部的PWM刹车信号
PWM刹车输入信号，可通过寄存器选择比较器的输出
芯片外部端口
芯片内部控制信号（有对应寄存器控制位）


CMPPS 通过内部的寄存器控制位来选择
比较器的正端输入选择
P2.7 (CMP+_0)
P4.3 (CMP+_1)
OP4O (CMP+_2)
ADCBINx (CMP+_3)
DAC1O (CMP+_4)
DAC2O (CMP+_5)
000
001
010
011
100
101
OP4O：第4组运算放大器的输出
ADCBINx：第2组ADC的输入通道，0~15通道
比较器的负端输入选择
P2.6 (CMP-_0)
P4.4 (CMP-_1)
OP1O (CMP-_2)
1.19V (CMP-_3)
DAC1O (CMP-_2)
DAC2O (CMP-_3)
OP1O：第1组运算放大器的输出
DAC1O：第1组DAC的输出
DAC2O：第2组DAC的输出
CMPNS
通过内部的寄存器控制位来选择
比较器的负端输入选择
CMP4
模拟滤波
数字滤波
通过内部的寄存器控制位来选择
CMPO_S
0
1
(CMPO_0) P2.5
(CMPO_1) P4.6
比较器的输出可通过I/O口输出
CMPO
比较器的输出，可输出到内部的PWM刹车信号
PWM刹车输入信号，可通过寄存器选择比较器的输出
芯片外部端口
芯片内部控制信号（有对应寄存器控制位）


第一组独立ADC的
输入通道选择
ADC_CHS
通过内部寄存器选择第一组
独立ADC的输入通道
OP1O是运放1的输出信号
OP2O是运放2的输出信号
OP3O是运放3的输出信号
OP4O是运放4的输出信号
1.19V是内部辅助信号源
ADC（第一组ADC）
P1.0 (ADC0)
P1.1 (ADC1)
P1.2 (ADC2)
P1.3 (ADC3)
P1.4 (ADC4)
P1.5 (ADC5)
P1.6 (ADC6)
P1.7 (ADC7)
P0.0 (ADC8)
P0.1 (ADC9)
P0.2 (ADC10)
OP1O (ADC11)
OP2O (ADC12)
OP3O (ADC13)
OP4O (ADC14)
1.19V (ADC15)
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111
CMP
SAR
控制逻辑
ADC_RES[11:0]
DAC
12
ADC_VRef+
ADC_VRef-
VREF
ADC_VRef-接到了ADC的AGnd，ADC_VRef+有独立的管脚

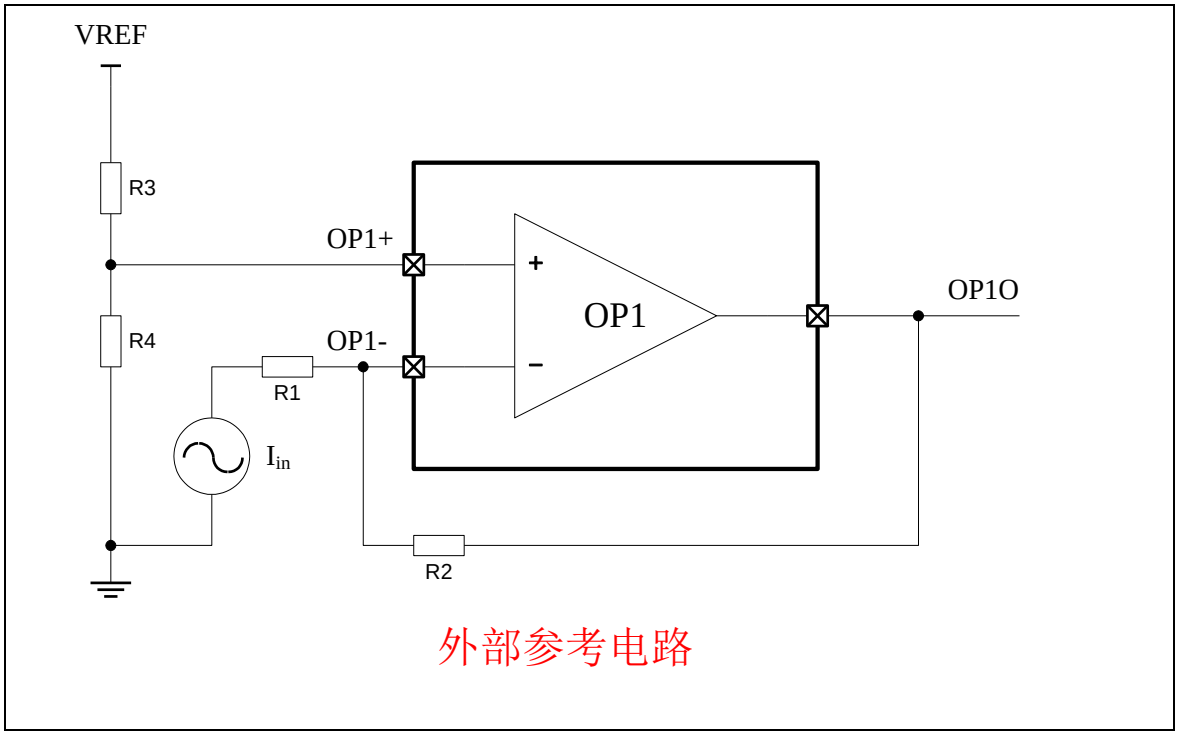

第二组独立ADC的
输入通道选择
ADCB_CHS
通过内部寄存器选择第二组
独立ADC的输入通道
ADCB中的字母B代表第2组ADC
OP1O是运放1的输出信号
OP2O是运放2的输出信号
OP3O是运放3的输出信号
OP4O是运放4的输出信号
1.19V是内部辅助信号源
ADCB（第二组ADC）
P0.0 (ADCB0)
P0.1 (ADCB1)
P0.2 (ADCB2)
P0.3 (ADCB3)
P0.4 (ADCB4)
P0.5 (ADCB5)
P0.6 (ADCB6)
P0.7 (ADCB7)
P6.0 (ADCB8)
P6.1 (ADCB9)
P6.2 (ADCB10)
OP1O (ADCB11)
OP2O (ADCB12)
OP3O (ADCB13)
OP4O (ADCB14)
1.19V (ADCB15)
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111
CMP
SAR
控制逻辑
ADCB_RES[11:0]
DAC
12
ADC_VRef+
ADC_VRef-
VREF
ADC_VRef-接到了ADC的AGnd，ADC_VRef+有独立的管脚

3.2.1 反相放大

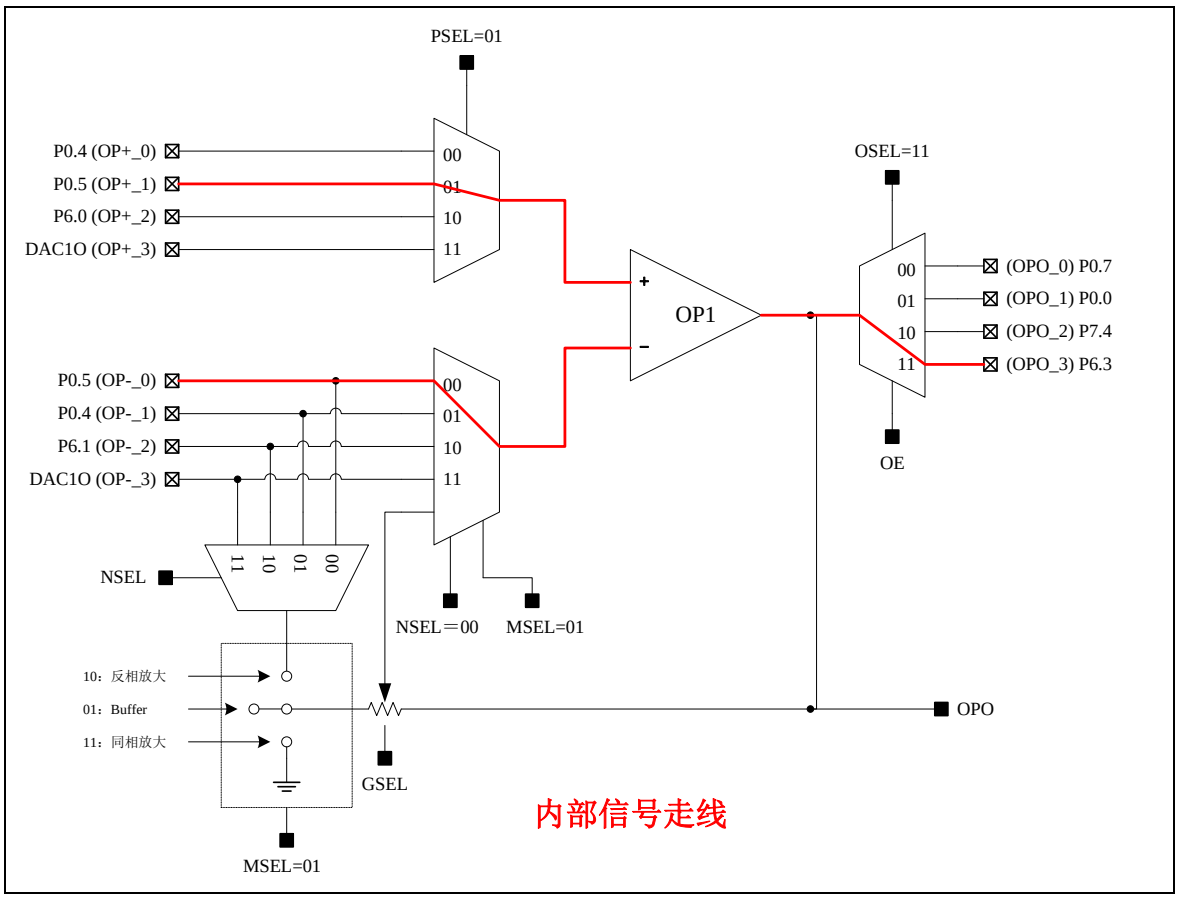
一般作电流转电压时的应用，可以用此电路 (如烟感 ….)

我们的芯片可以做以下设置

1. OP 对外部信号进行反相放大，增益电路外接，OP 工作在缓冲模式

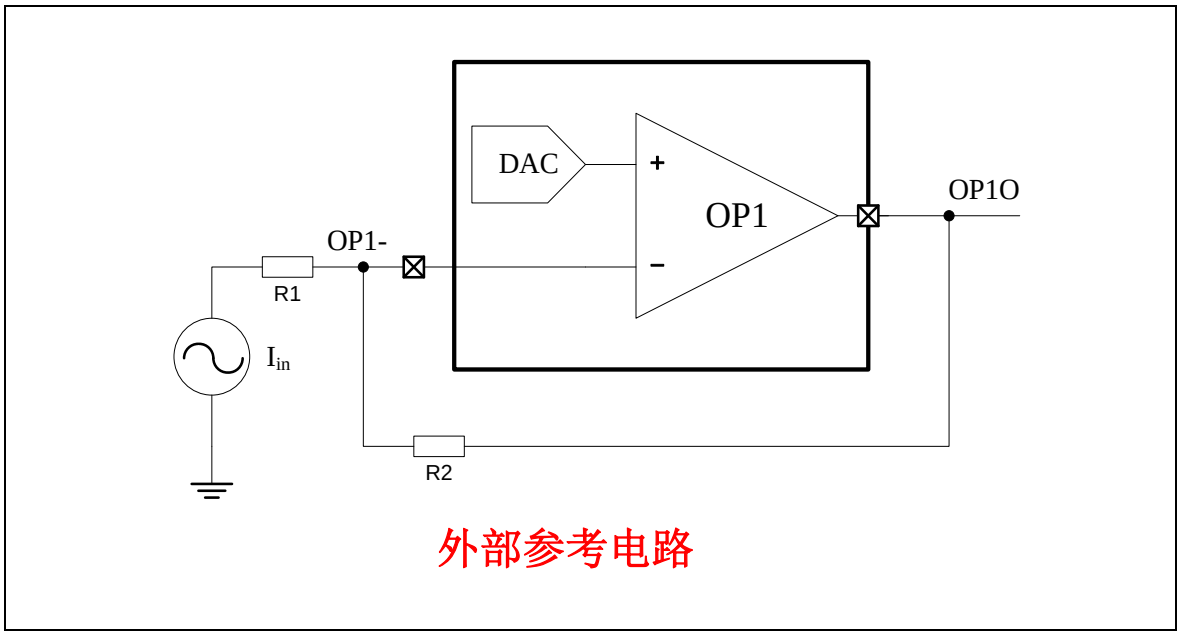


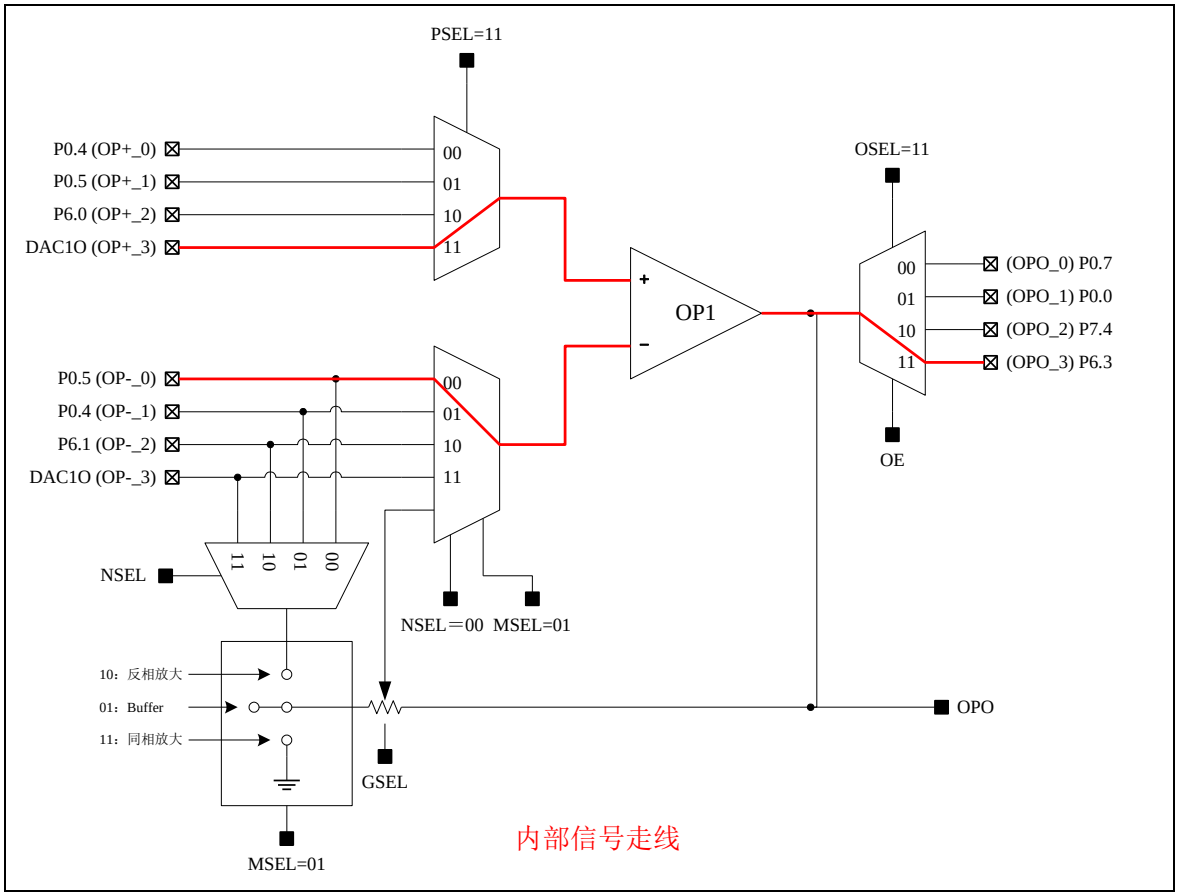
外部参考电路



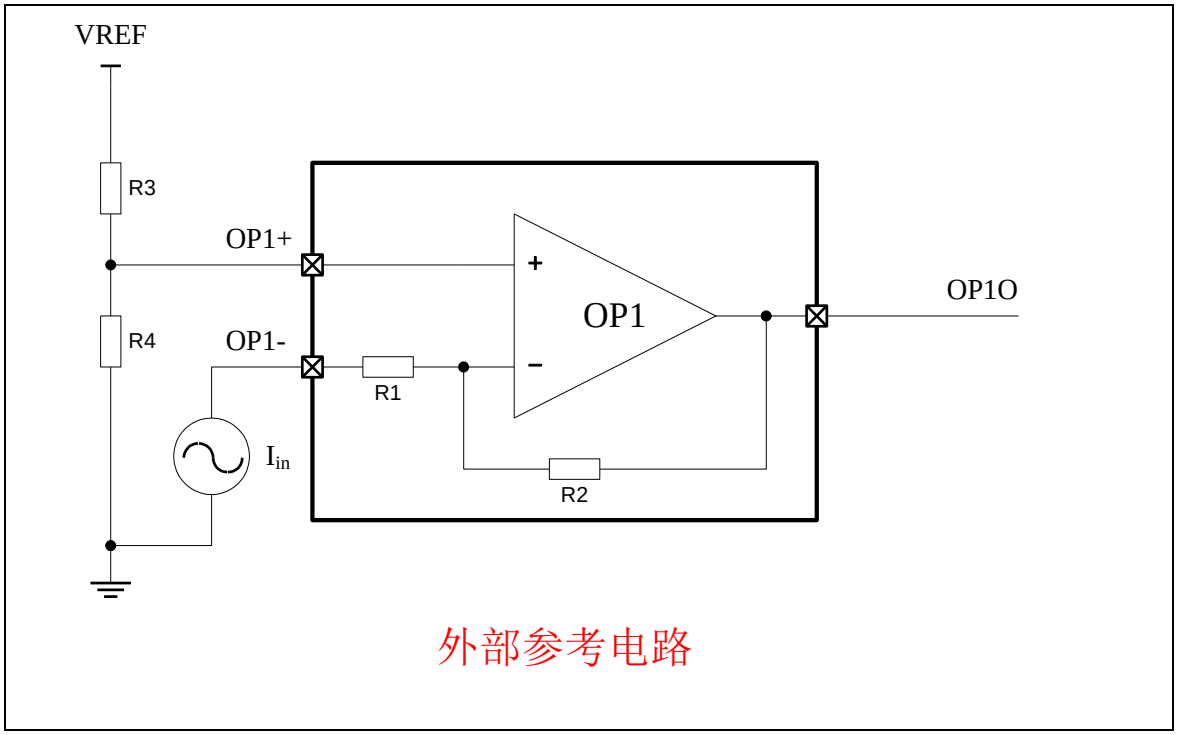
内部信号走线

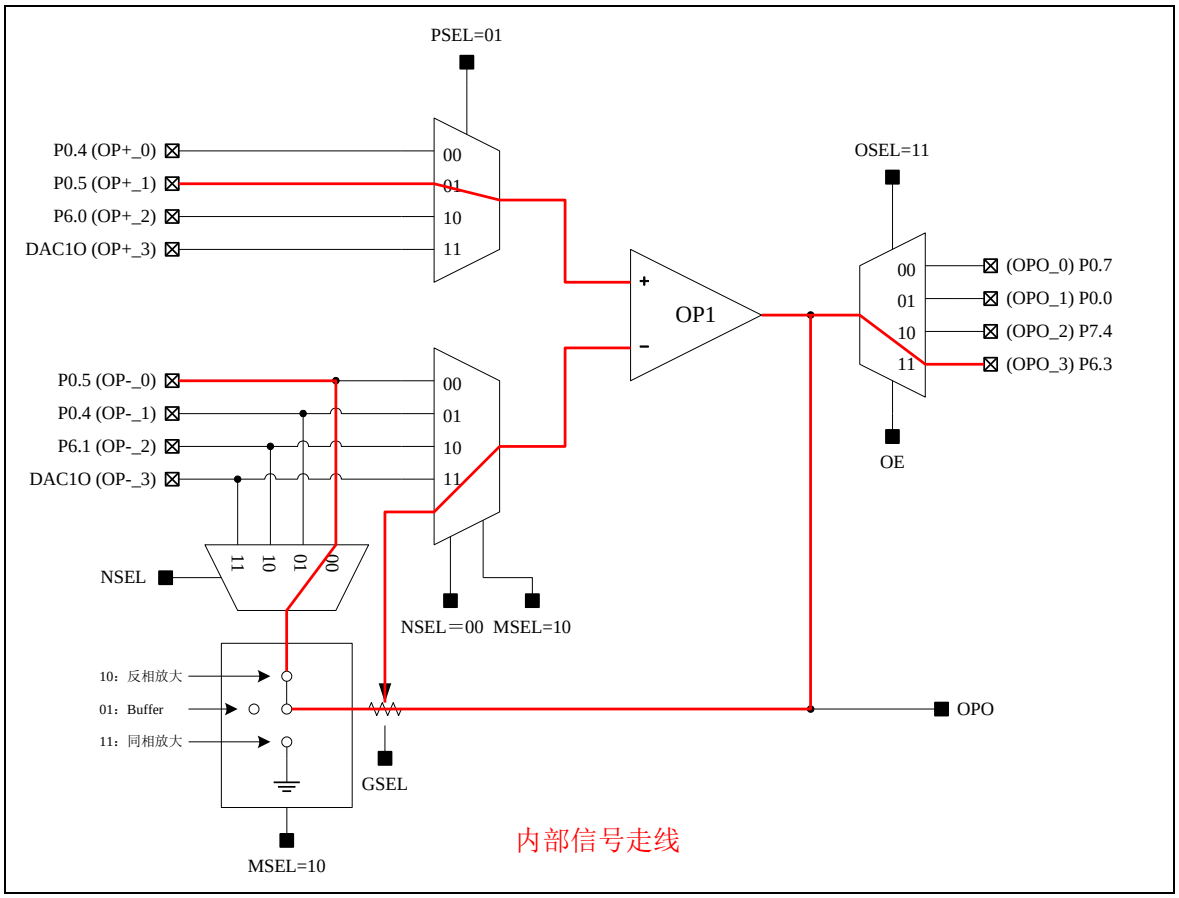
2. OP 对内部 DAC 输出信号进行反相放大，增益电路外接，OP 工作在缓冲模式



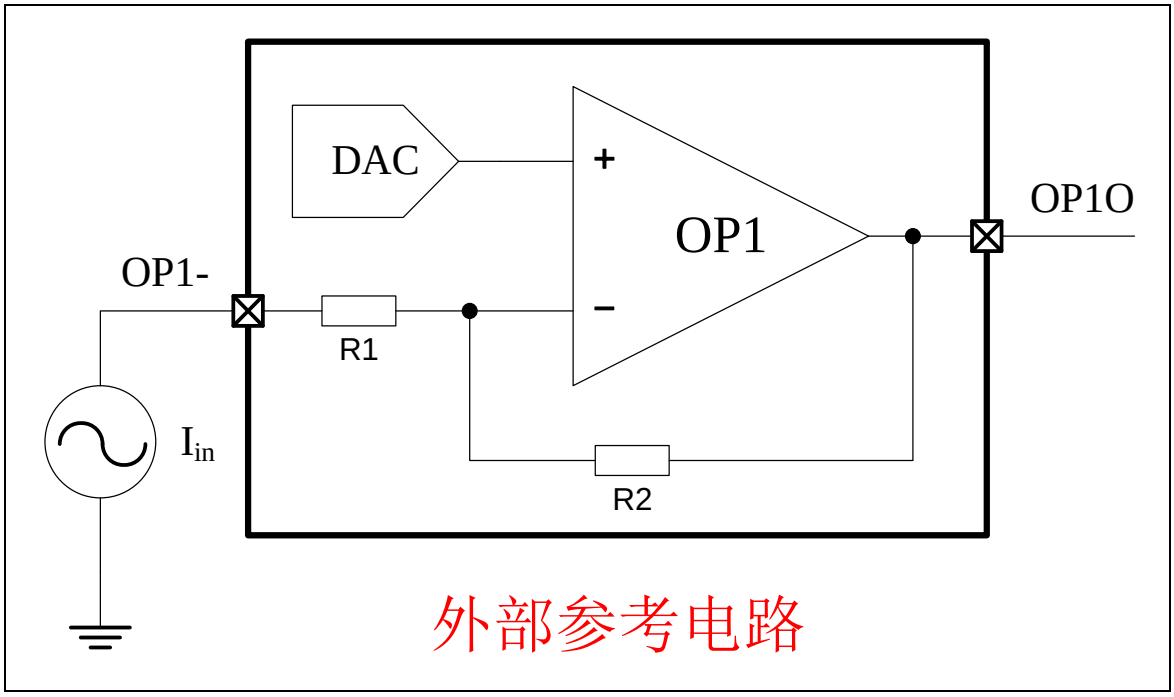


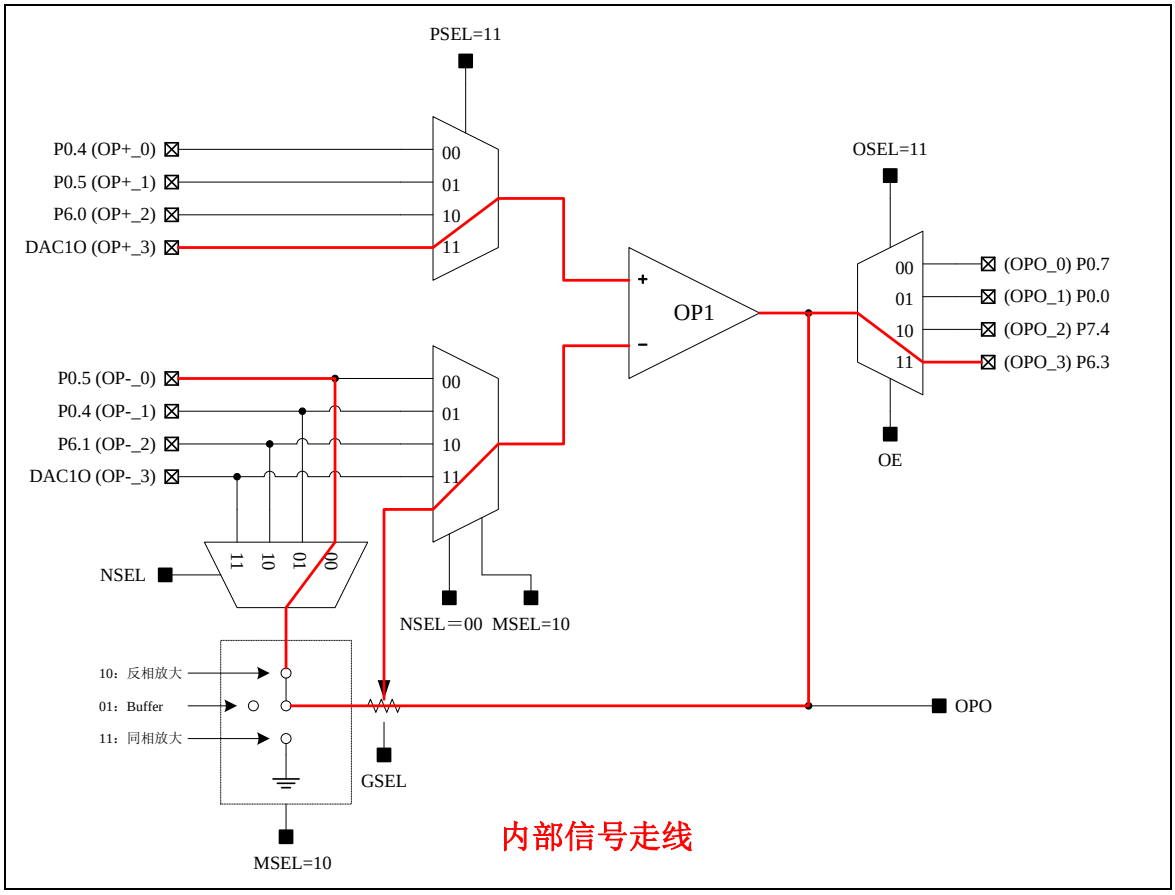
3. OP 对外部信号进行反相放大，使用内部增益电路，OP 工作在反相放大模式（不建议使用，用内部 DAC 更好)





4. OP 对内部 DAC 输出信号进行反相放大，使用内部增益电路，OP 工作在反相放大模式

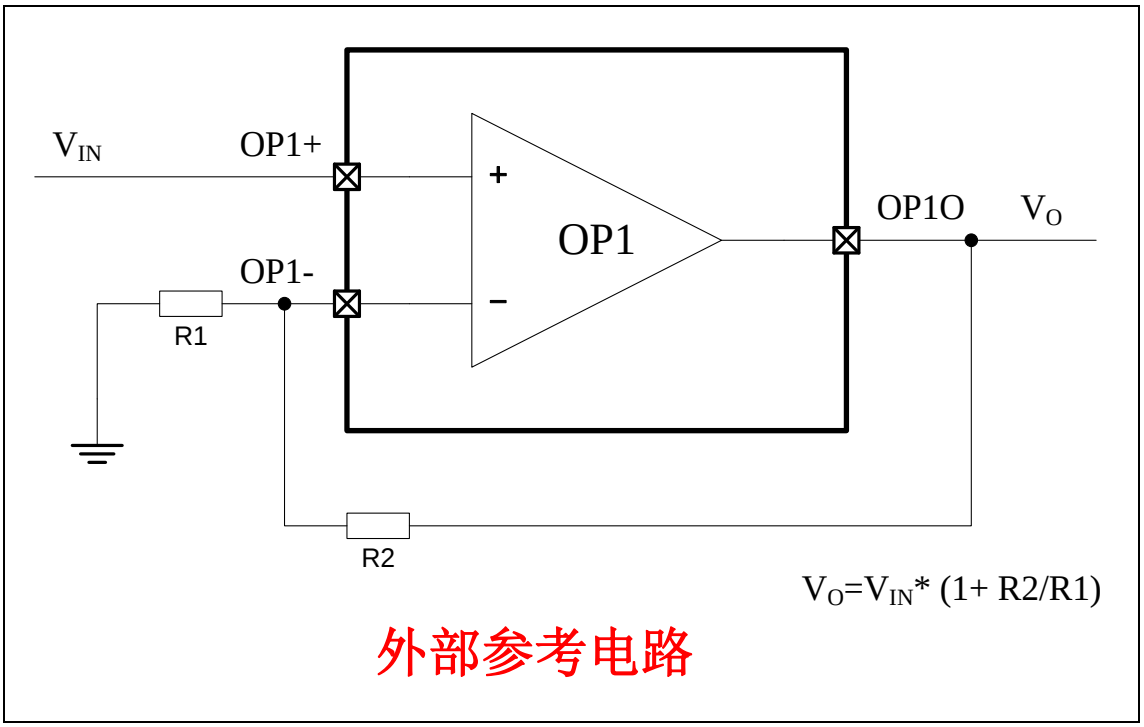


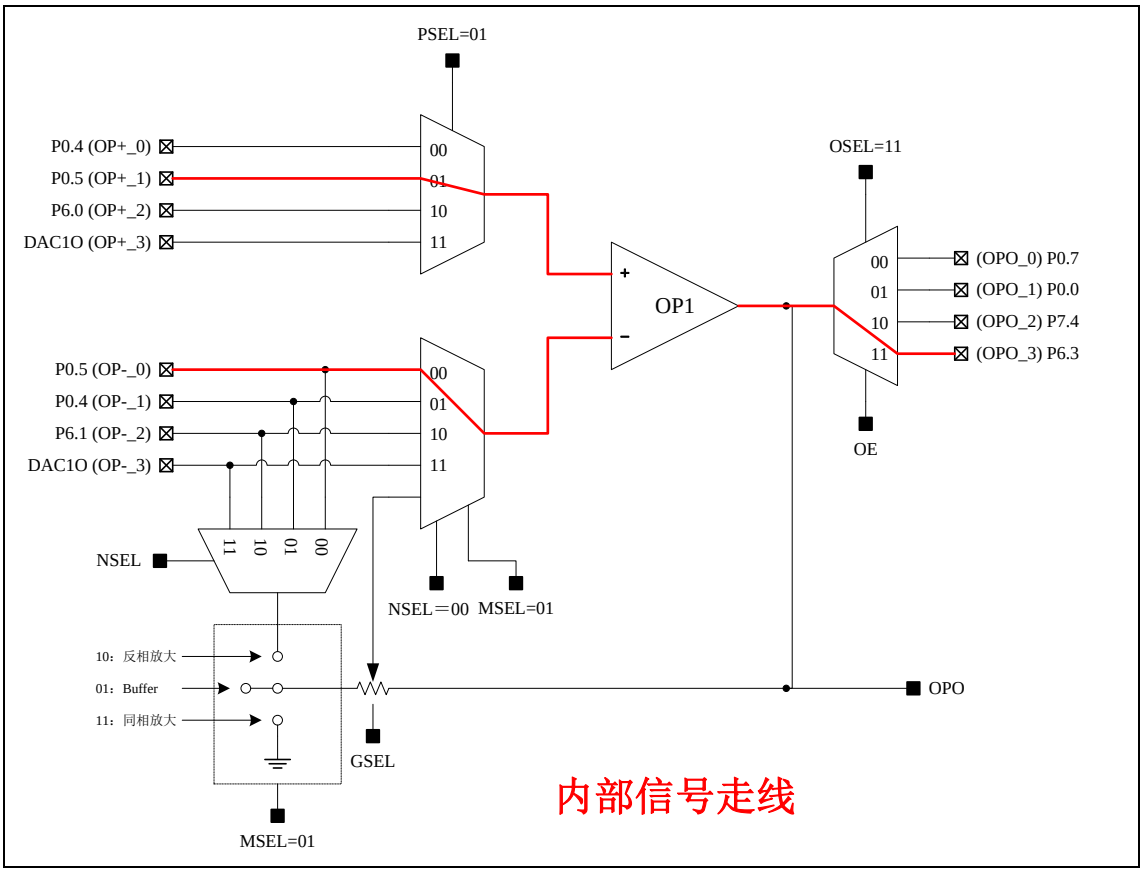


3.2.2 同相/正相放大

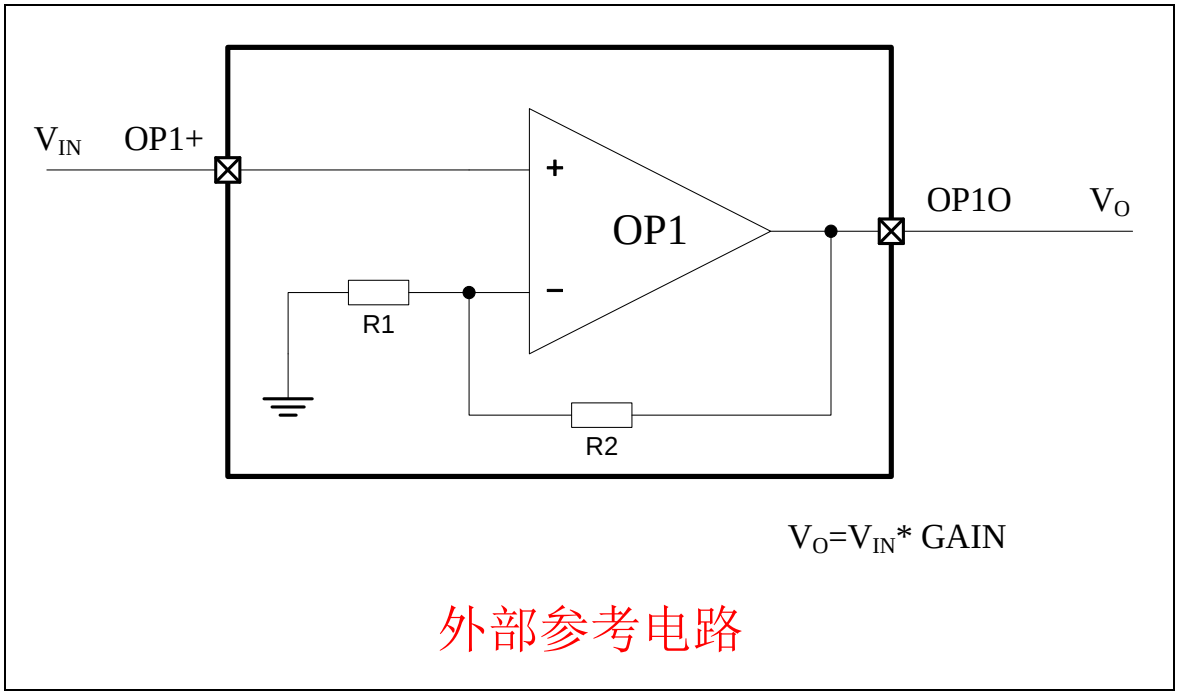
一般作小电压放大时的应用，可以用此电路如(压力表头 ，耳温枪 ，电流侦测 ，过电流保护)
我们的芯片可以做以下设置

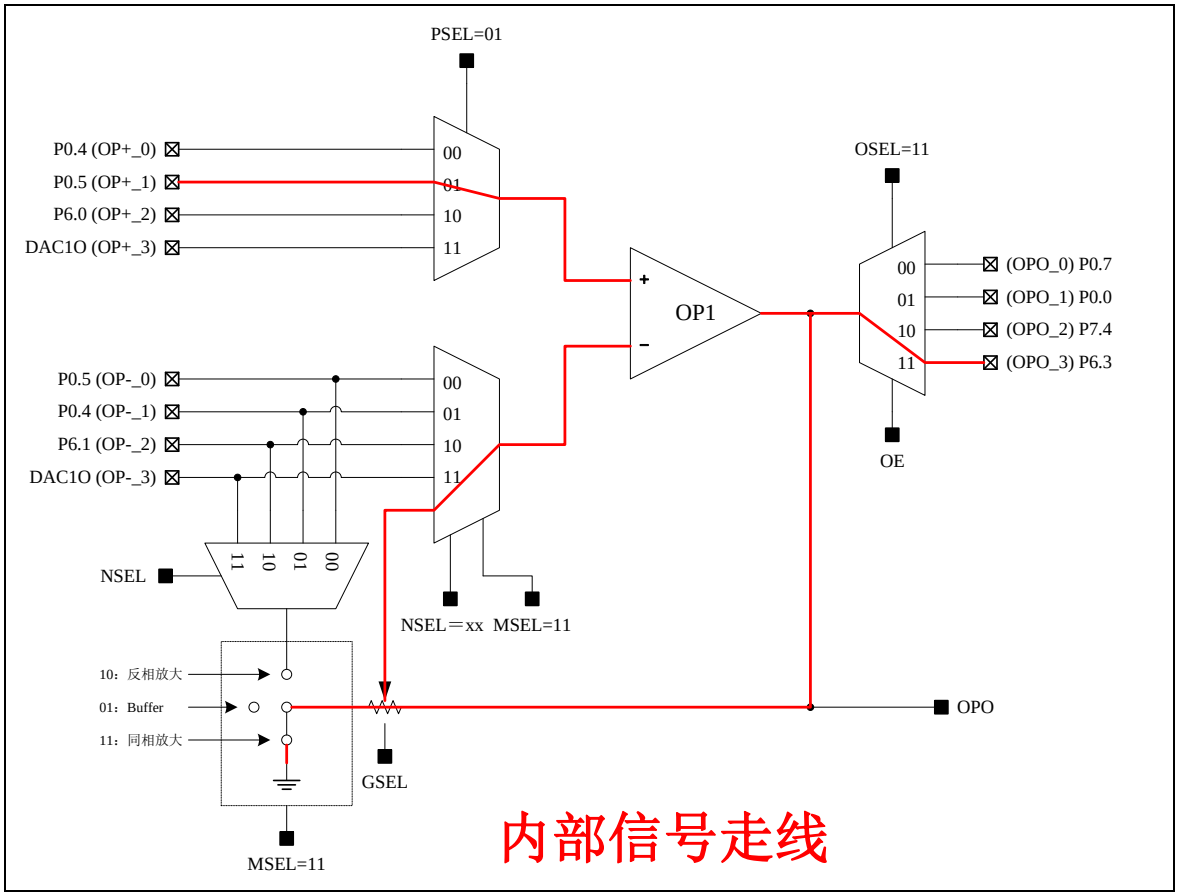
1. OP 对外部信号进行正相放大，增益电路的电阻外接，OP 工作在缓冲模式



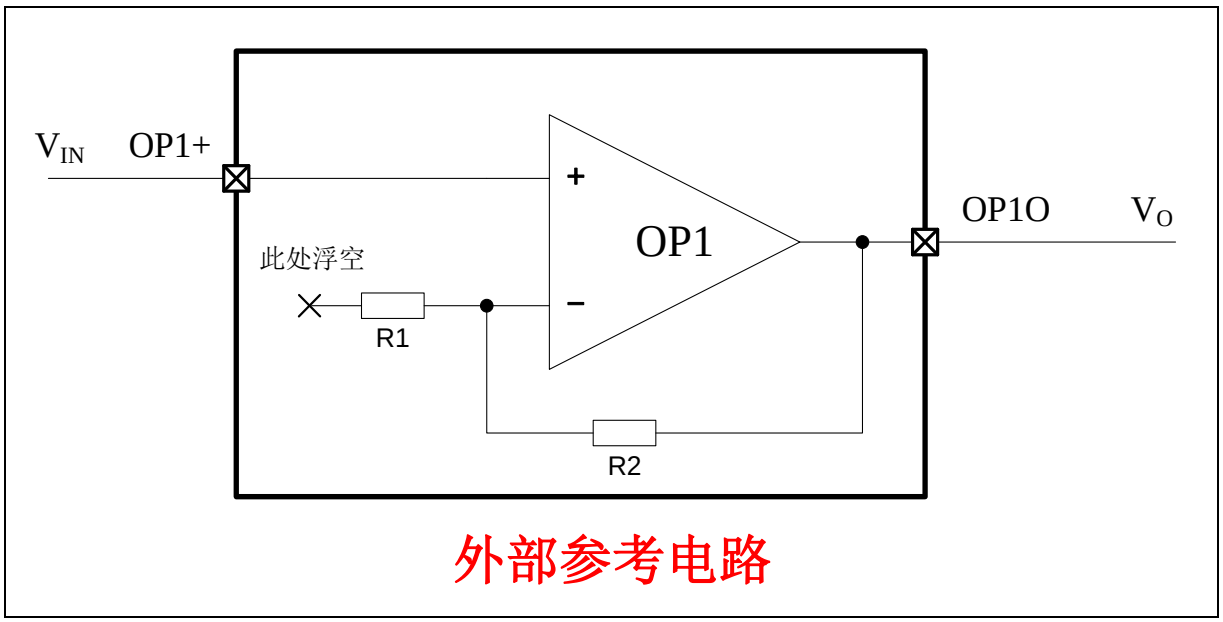


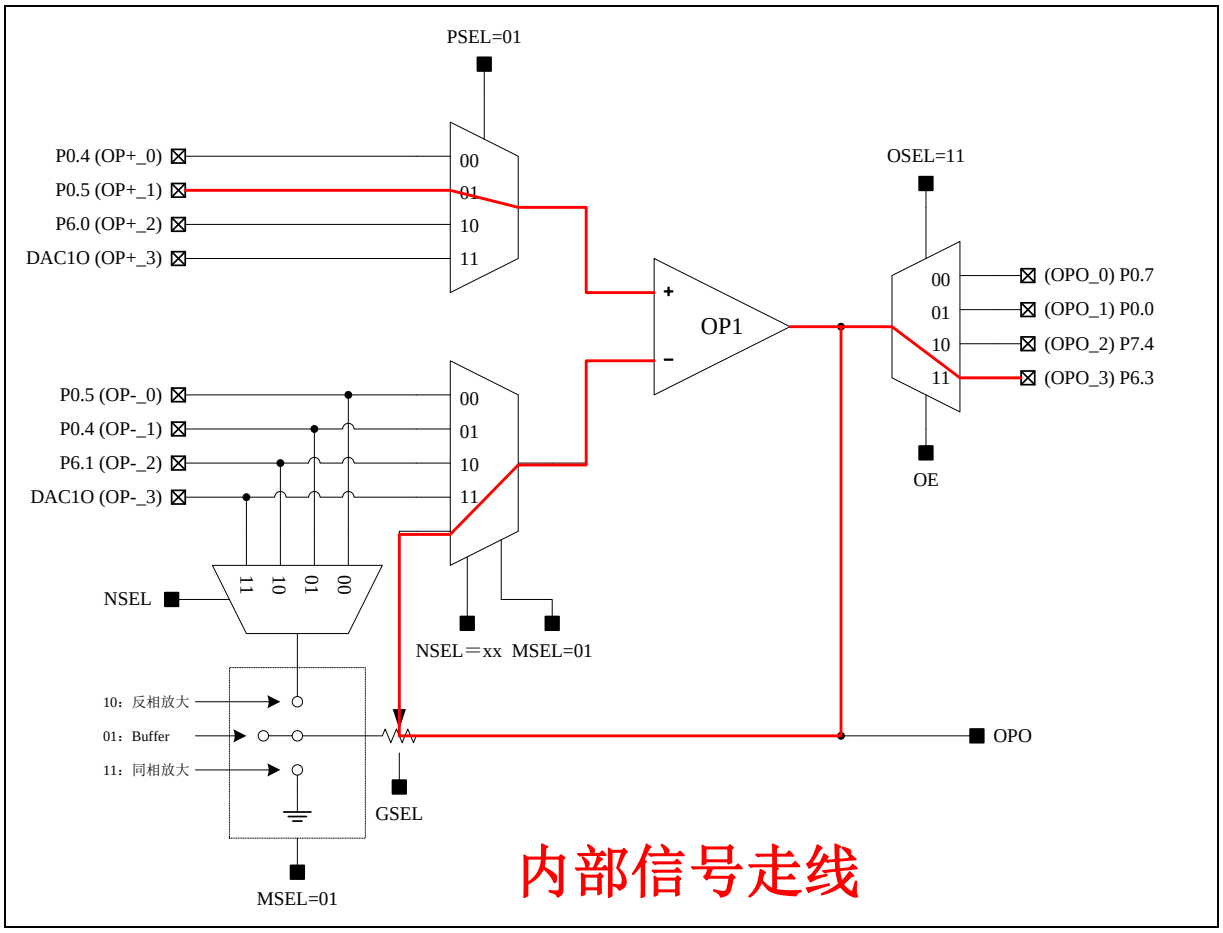
2. OP 对外部信号进行正相放大，使用内部增益电路，OP 工作在正相放大模式



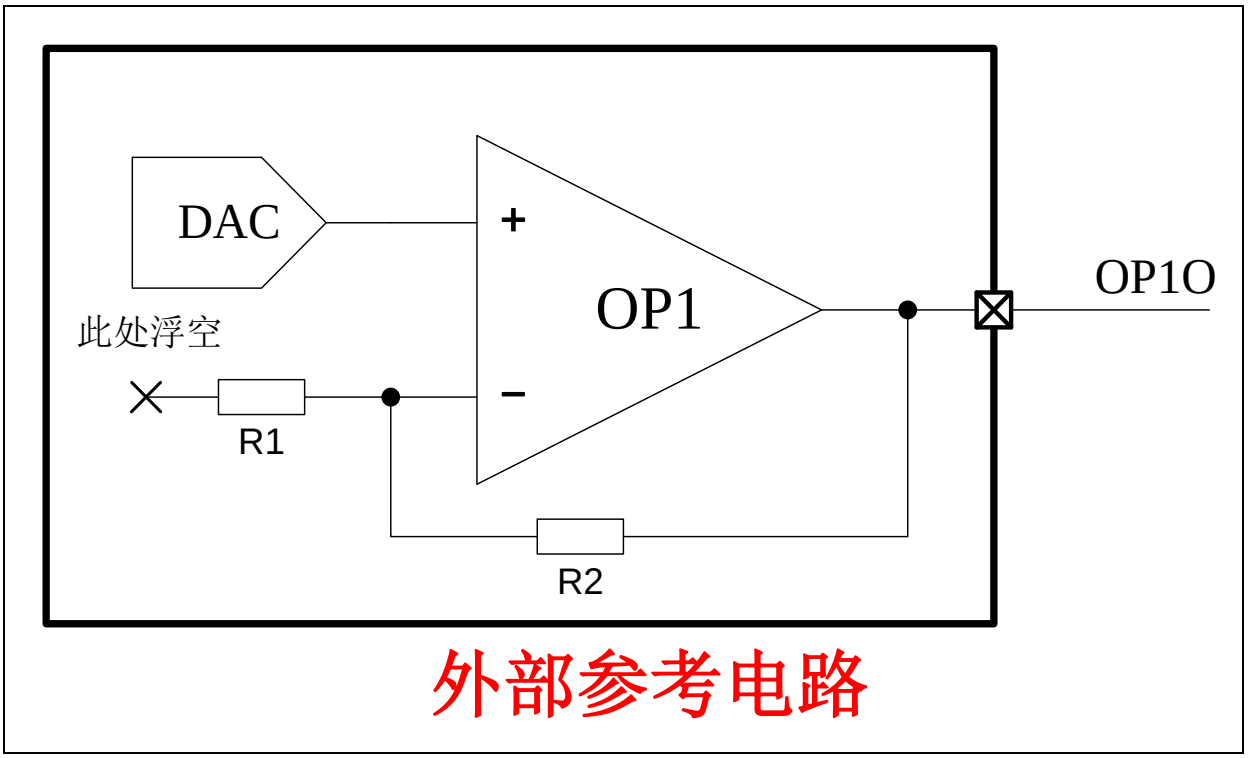


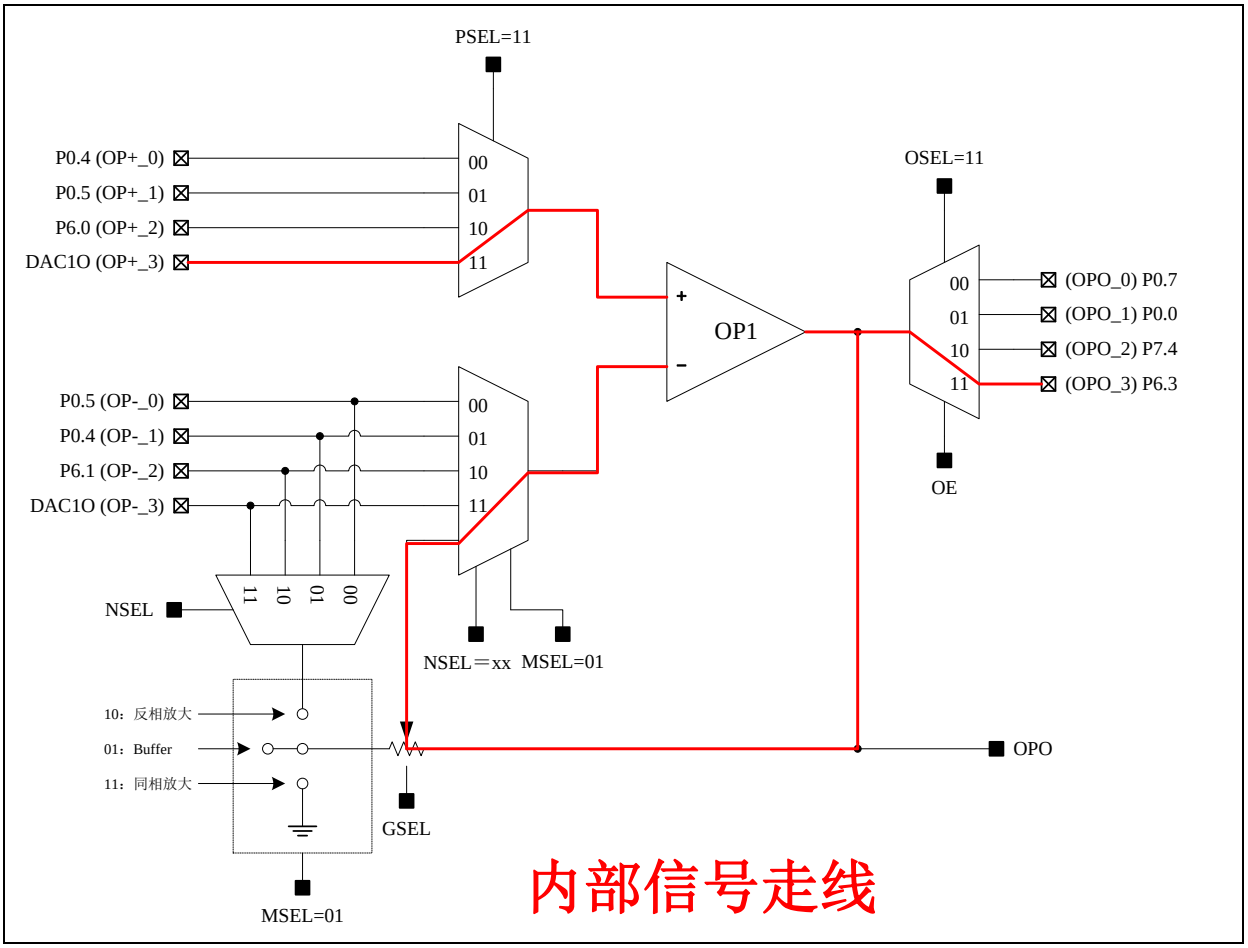
3.2.3 Input BUFFER Mode（输入缓冲模式）





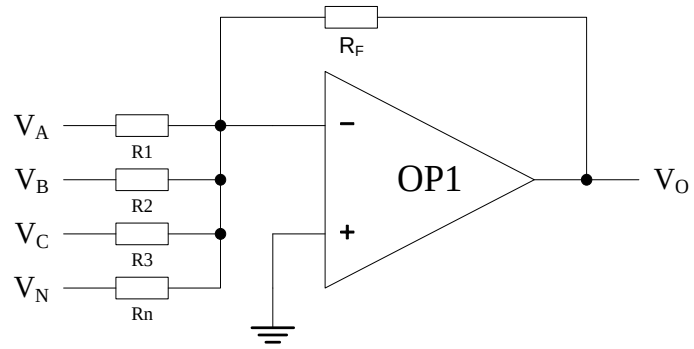
3.2.4 DAC BUFFER Mode（DAC 缓冲模式）





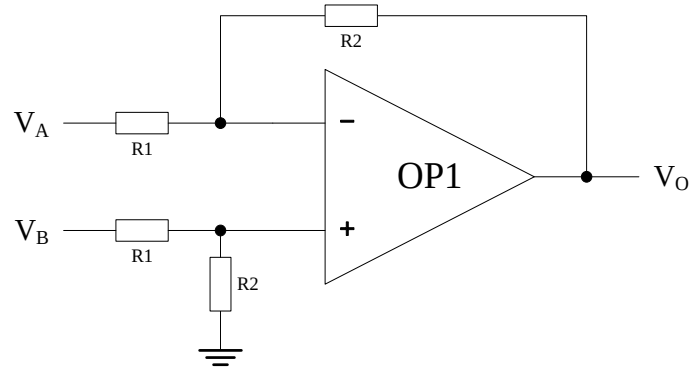
3.2.5 常见的运放应用电路，不是本器件的内部特定电路

Other mode:(建议使用 GP MODE 来外接电路)



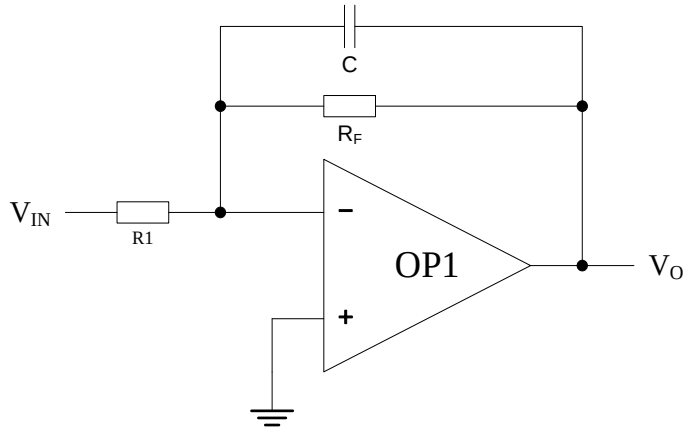
$$Vo = (-R_F)\left(\frac{V_A}{R_1} + \frac{V_B}{R_2} + \frac{V_C}{R_3} + \frac{V_N}{R_n}\right)$$

加 法 器



$$V_O = (V_B - V_A)\left(\frac{R_2}{R_1}\right)$$

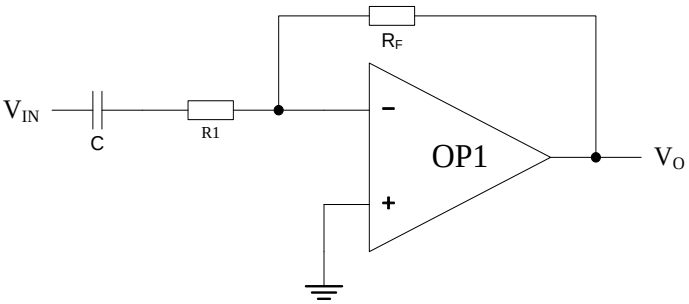
减 法 器



$$T = RC = R_FC$$

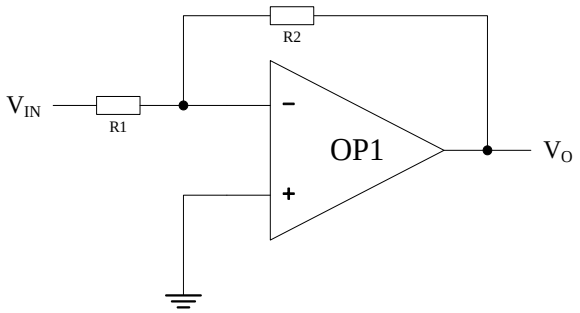
$$V_O = V_{IN}\left(-\frac{R_F}{R_1} \times \frac{1}{1 + j\omega R_FC}\right)$$

低通滤波器/积分器



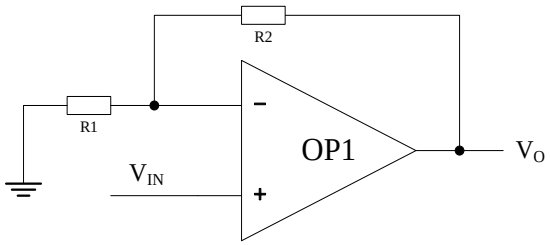
$$V_O = V_{IN}\left(-\frac{R_F}{R_1} \times \frac{1}{1 + \frac{1}{j\omega R_1C}}\right)$$

高通滤波器/微分器



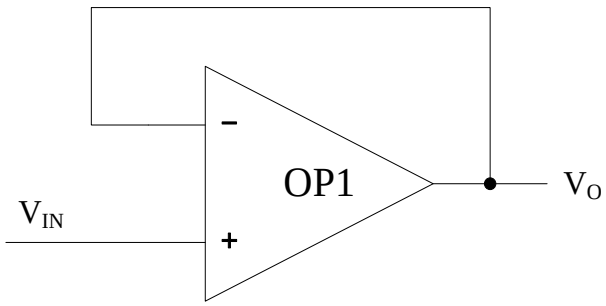
$$V_O = -\frac{R_2}{R_1} V_{IN}$$

反相放大器



$$V_O = \left(1 + \frac{R_2}{R_1}\right) V_{IN}$$

同相放大器



$$V_O = V_{IN}$$

电压跟随器

4 STC32G96K246-64MHz-LQFP100/64/48 系列提前预告

4.1.1 特性及价格

➢ 选型价格（不需要外部晶振、不需要外部复位，两组 12 位 ADC，15+15 通道，对外 28 个通道）

单片机型号	工作电压（V）	Flash 程序存储器 10 万次 字节	edata 内部扩展 DATA RAM 可做堆栈或变量 字节	xdata 内部大容量扩展 SRAM 可做变量 字节	EEPROM 10 万次 字节	I/O 口最多数量	传统 I/O 中断（INT0/INT1/INT2/INT3/INT4）并可掉电唤醒	所有 I/O 口均支持高电平 低电平 上升沿 下降沿中断，可掉电唤醒	DMA 8080/6800 接口 TFT 彩屏模块驱动(8 位和 16 位)	RTC 实时时钟 年/月/日/时/分/秒/星期 对内部时钟追频 掉电唤醒	USB-FS OTG，支持 USB-CDC, USB-HID	DMA USART-LIN / SPI / IrDA 并可掉电唤醒	汽车电子通讯 LIN 总线	单总线（支持 WS2812 / DS18B20 / DHT11 / NEC/R / UART）	DMA I2S 音频总线 掉电唤醒（全双工录音 + 放音）	DMA SPI 并可掉电唤醒	DMA QSPI	DMA I²C 并可掉电唤醒	DMA 汽车电子通讯 CAN-FD 总线 并可掉电唤醒	TFPU (144MHz)硬件浮点 / 硬件三角 / 反三角函数运算器	DSP32 （64MHz），硬件 32 位/64 位 DSP 指令	定时器计数器（T0～T4,T11, T17/T18 外部管脚 也可掉电唤醒）	144MHz，16 位高级 PWM 定时器 互补对称死区控制	掉电唤醒 专用定时器	DMA 支持两个独立的 12 位 ADC，每个 16 通道，共 32 通道	DMA 支持两个独立的 12 位高速 DAC	比较器（6P6N 模式，6 个正极输入，6 个负极输入）	运算放大器（OPA）	内部低压检测中断 并可掉电唤醒	看门狗 复位定时器	内部高可靠复位（可选复位门槛电压）	内部高精准时钟（72MHz 以下可调），满足串口通信，追频	可对外输出时钟及复位	程序加密后传输（防拦截），可设置下次更新程序需口令	支持 RS485 下载	本身就可在线仿真，支持硬件 USB 直接下载和硬件 USB 仿真	价格及封装			供货信息
																																							LQFP100	LQFP64	LQFP48	
STC32G96K246	1.9-5.5	246K	32K	64K	IAP	90	有	有	有	有	有	4	1	4	2	1+4	2	1	3	有	有	8	8	有	2 组	2 组	3 组	4 组	有	有	4 级	有	是	是	是	是				

可实现 4+1 组 LIN

➢ **内核**

- ✓ 64MHz CPU/DSP，144MHz TFPU/PWM
- ✓ 超高速 32 位 8051 内核（1T），比传统 8051 同频快 70 倍以上
- ✓ 4 级中断优先级
- ✓ 支持在线仿真

➢ **工作电压**

- ✓ 1.9V ～ 5.5V（当工作温度低于-40℃时，工作电压不得低于 3.0V，低温时工作电压要提高）

➢ **工作温度**

- ✓ -20℃～65℃（内部高速 IRC 温漂-0.76%~+0.98%）
- ✓ -40℃～85℃（内部高速 IRC 温漂±1.3%）
- ✓ -40℃～125℃（内部高速 IRC 温漂±3%，当温度高于 85℃时请使用外部 35MHz 及以下的晶振）

➢ **Flash 存储器**

- ✓ 最大 246K 字节 FLASH 程序存储器（ROM），用于存储用户代码
- ✓ 支持用户可配置部分 FLASH 作为 DATA FLASH/EEPROM 使用，512 字节单页擦除，**典型单个扇区擦除次数可达 10 万次，用户保守可当 2 万次使用**
- ✓ 支持硬件 USB 直接下载和普通串口下载
- ✓ 支持硬件 SWD 实时仿真，P3.0/P3.1（需 USB-Link1D 工具）
- ✓ **Flash 数据保持时间：100 年@25℃，20 年@105℃，10 年@125℃。由于汽车及单片机的平均温度不可能长时间大于 105℃，从严考虑，Flash 数据保持时间可达 20 年以上，不保证 100 年。**

➢ **SRAM，共 96K 字节**

- ✓ 32K 字节内部 SRAM（edata）
- ✓ 64K 字节内部扩展 RAM（内部 xdata）

车规启航
扫码去微信小商城

- **时钟控制**
  - ✓ 内部高精度、高稳定的高速 IRC（ISP 编程时可进行上下调整）
    - ⌖ 误差±0.3%（常温下 25℃）
    - ⌖ -1.35% ～ +1.30%温漂（全温度范围，-40℃ ～ 85℃）
    - ⌖ -0.76% ～ +0.98%温漂（温度范围，-20℃ ～ 65℃）
  - ✓ 内部 32KHz 低速 IRC（为了低功耗，省去了温度补偿和电压补偿电路，误差较大）
  - ✓ 外部晶振（4MHz ～ 35MHz）和外部时钟，有专门的外部时钟干扰内部电路，可软件启动
  - ✓ **内部两组独立的 96MHz/144MHz 的 PLL，可输出的时钟提供给 TFPU、PWM、I2S、SPI 等高速外设**

用户可自由选择上面的 4 种时钟源

（芯片上电工作过程：上电复位/复位脚复位/看门狗复位/低压检测复位时，芯片默认从 ISP 系统程序开始执行代码，此时固定使用内部 24MHz 的高速 IRC 时钟，当需要下载用户程序且下载完成后复位到用户程序区或者不需要下载直接复位到用户程序区时，默认会使用上次用户下载时所调节的高速 IRC 时钟，如果用户程序需要使用外部高速晶振、外部 32.768KHz 晶振或者内部 30KHz 低速 IRC，则需要用户软件先启动相应的时钟，然后通过设置 CLKSEL 寄存器进行切换）

- **复位**
  - ✓ 硬件复位
    - ⌖ 上电复位，复位电压值为 1.7V ～ 1.9V。（在芯片未使能低压复位功能时有效）
    - ⌖ 复位脚复位，出厂时 P5.4 默认为 I/O 口，ISP 下载时可将 P5.4 管脚设置为复位脚（注意：当设置 P5.4 管脚为复位脚时，复位电平为低电平）
    - ⌖ 看门狗溢出复位
    - ⌖ 低压检测复位，提供 4 级低压检测电压：2.0V、2.4V、2.7V、3.0V。
  - ✓ 软件复位
    - ⌖ 软件方式写复位触发寄存器

- **中断**
  - ✓ 中断源：INT0、INT1、INT2、INT3、INT4、定时器 0、定时器 1、定时器 2、定时器 3、定时器 4、定时器 11、定时器 17、定时器 18、USART1、USART2、USART3、USART4、ADC 模数转换、ADC2 模数转换、DAC1 数模转换、DAC2 数模转换、LVD 低压检测、SPI、QSPI1、QSPI2、I^2C、OW1、OW2、OW3、OW4、比较器 1、比较器 2、比较器 3、PWMA、PWMB、USB、TFT 彩屏接口中断、RTC 实时时钟、所有的 I/O 中断（8 组）、所有的 USART 串口的 DMA 接收和发送中断（4 组）、I2C 的 DMA 接收和发送中断、I2S 的 DMA 接收和发送中断（2 组）、SPI 的 DMA 中断、QSPI 的 DMA 中断（2 组）、OW 的 DMA 中断（4 组）、ADC 的 DMA 中断（2 组）、DAC 的 DMA 中断（2 组）、LCD 驱动的 DMA 中断、CAN-FD 的 DMA 中断（3 组）、PWM 的 DMA 中断（PWMA）、存储器到存储器的 DMA 中断以及**两组外设到外设的 DMA 中断**。
  - ✓ 提供 4 级中断优先级

- **数字外设**
  - ✓ 8 个 16 位定时器：定时器 0、定时器 1、定时器 2、定时器 3、定时器 4、定时器 11、定时器 17、定时器 18，其中定时器 0 的模式 3 具有 NMI（不可屏蔽中断）功能，定时器 0 和定时器 1 的模式 0 为 16 位自动重载模式。定时器 11：可选时钟源。**定时器 17 和定时器 18 可同步触发两组 ADC 或者同步触发两组 DAC**。
  - ✓ 4 个高速同步/异步串口：串口 1（USART1）、串口 2（USART2）、串口 3（USART3）、串口 4（USART4），波特率时钟源最快可为 FOSC/4。支持同步串口模式、异步串口模式、SPI 模式、LIN 模式、红外模式（IrDA）、智能卡模式（ISO7816）
  - ✓ 8 通道高级 PWM，1+1 组高级 PWM，每组均可实现 4 通道 PWM。PWMA 可实现 4 对互补对称带死区控制的 PWM，PWMB 可实现 4 通道 PWM 输出，并支持外部异常检测功能。
  - ✓ **2 个 QSPI：**支持单线、双线和四线模式。QSPI1 支持同时读 2 个 QSPI，一次性读 8 位；支持运行 QSPI-FLASH

中的外部程序。QSPI2 是传统 QSPI，不支持运行 QSPI-FLASH 中的外部程序。

- ✓ SPI：1+4 组硬件 SPI（1 组独立 SPI，4 组 USART 的 SPI 模式）支持主机模式和从机模式以及主机/从机自动切换（注：5 组 SPI 均支持 DMA，5 组 SPI 的 MISO 和 MOSI 脚均可软件设置进行功能交换）。
- ✓ I^2C：独立的硬件 I^2C，支持主机模式和从机模式
- ✓ ICE：支持硬件仿真，支持串口和 USB 口直接仿真，还支持 SWD 硬件仿真
- ✓ RTC：支持年、月、日、星期、时、分、秒、次秒（1/128 秒），并支持时钟中断和一组闹钟。芯片复位时 RTC 的时间计数值不复位。
- ✓ USB-FS OTG：USB2.0/USB1.1 兼容全速 USB，16 个双向端点，支持 4 种端点传输模式（控制传输、中断传输、批量传输和同步传输），端点拥有 64/128/256/512/1024 字节的 FIFO 缓冲区，不同的端点的 FIFO 缓冲区大小不等。由于已支持 OTG，可当 Host 使用。
- ✓ CAN：3 个独立的 CAN-FD 2.0 控制单元。CAN-FD 的 STB 脚可软件设置是否使能，若不使能 STB 脚可当作 GPIO 使用。
- ✓ LIN：1+4 组硬件 LIN（1 组组独立 LIN，4 组 USART 的 LIN 模式），一个独立的 LIN 控制单元（支持 1.3 和 2.1 版本）
- ✓ I2S：两路独立的 I2S 音频总线，**可支持全双工方式的录音和放音**
- ✓ DSP32：硬件 32 位/64 位 DSP 运算指令
- ✓ TFPU：单精度浮点运算器（支持浮点加、减、乘、除以及正弦、余弦、正切和反正切等运算）
- ✓ I/O 口中断：所有的 I/O 均支持中断，每组 I/O 中断有独立的中断入口地址，所有的 I/O 中断可支持 5 种中断模式：高电平中断、低电平中断、上升沿中断、下降沿中断以及上升沿+下降沿中断。I/O 口中断可以进行掉电唤醒，且有 4 级中断优先级。
- ✓ LCD/TFT 彩屏驱动接口：支持 8080 和 6800 两种接口以及 8 位和 16 位数据宽度。对 16 位数据支持大小端数据格式选择。本模块首次尝试支持 DCMI 接口，不是专门的 DCMI 接口。
- ✓ **专门的 DCMI 数字摄像头接口：**支持 8 位、10 位、12 位和 14 位数据接口，支持外部行同步和帧同步
- ✓ DMA：支持 SPI（1 组）、QSPI（两组）、I2C（1 组）、I2S（两组）、串口（4 组）、OW（4 组）、PWM（1 组，PWMA）、ADC（两组）、DAC（两组）、CAN-DA（3 组）、LCM 的 DMA，以及外设到外设直接的 DMA
- ✓ 硬件数字 ID：支持 32+32 字节

➢ 模拟外设

- ✓ ADC：**2 组**独立的高速 ADC，支持 12 位高精度 15 通道（通道 0～通道 14）的模数转换，ADC 的通道 15 用于测试内部参考电压（芯片在出厂时，内部参考电压调整为 1.19V，误差±1%）
- ✓ DAC：两组独立的高速 DAC，支持 12 位转换精度。两组 DAC 的模拟输出均从芯片内部直接连接到比较器和运算放大器。
- ✓ 比较器：3 组独立的 6P6N 模式比较器（CMP）
- ✓ 运算放大器：4 组独立的运算放大器（OPA）

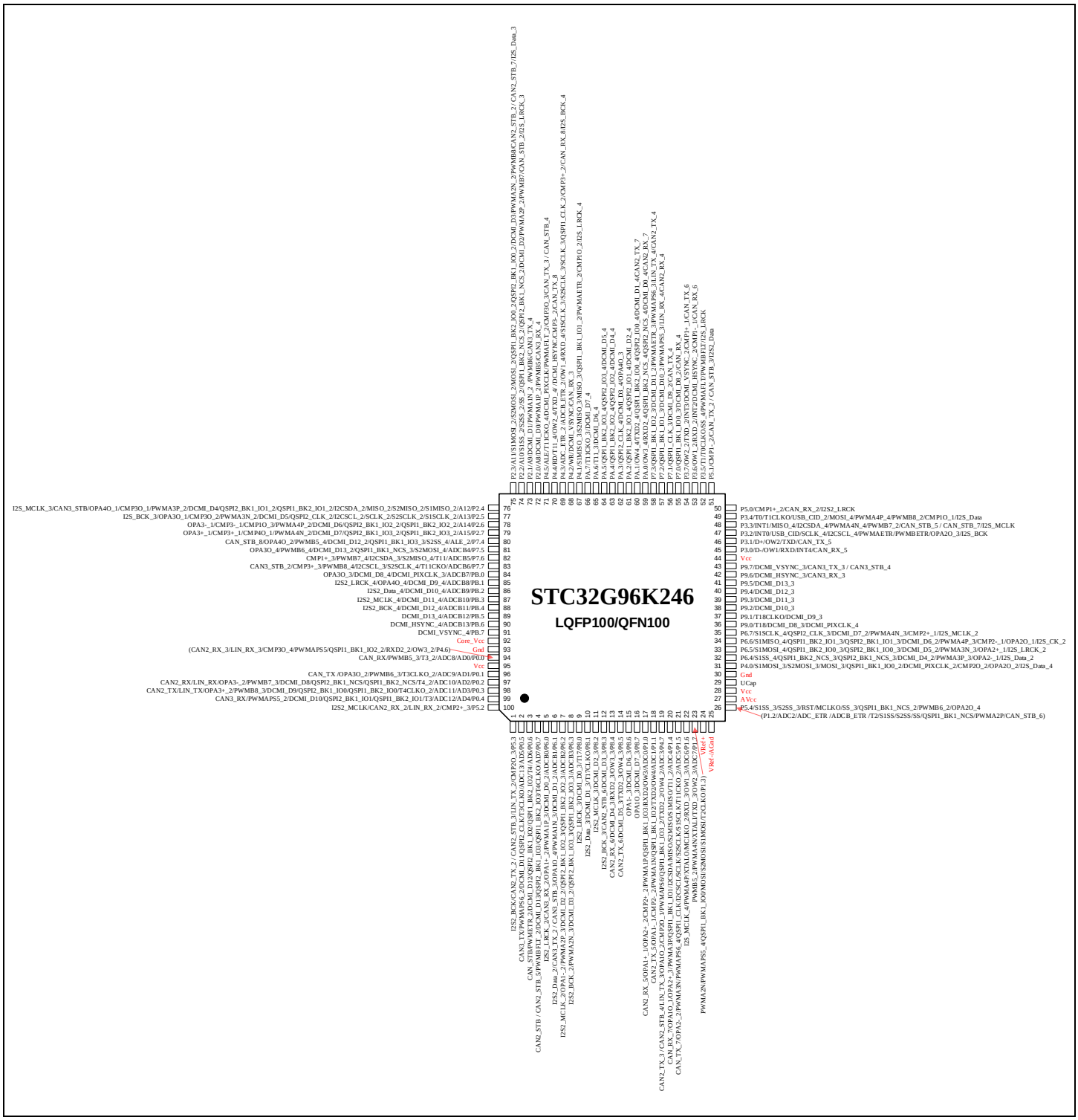
➢ GPIO

- ✓ 最多可达 90 个 GPIO：P0.0 ~ P0.7、P1.0 ~ P1.7（P1.2 和 P5.4 为同一个 PIN 脚、P1.3 和 P1.7 为同一个 PIN 脚）、P2.0 ~ P2.7、P3.0 ~ P3.7、P4.0 ~ P4.7（P4.6 和 P0.0 为同一个 PIN 脚）、P5.0 ~ P5.4、P6.0 ~ P6.7、P7.0 ~ P7.7、P8.0 ~ P8.7、P9.0 ~ P9.7、PA.0 ~ PA.7、PB.0 ~ PB.7
- ✓ 所有的 GPIO 均支持如下模式：**无弱上拉电阻的准双向口模式**、强推挽输出模式、开漏模式、高阻输入模式
- ✓ 所有 I/O 口上电后的状态均为高阻输入状态，用户在使用 I/O 口时必须先设置 I/O 口模式
- ✓ 另外每个 I/O 均可独立使能内部 10K 上拉电阻和 10K 下拉电阻

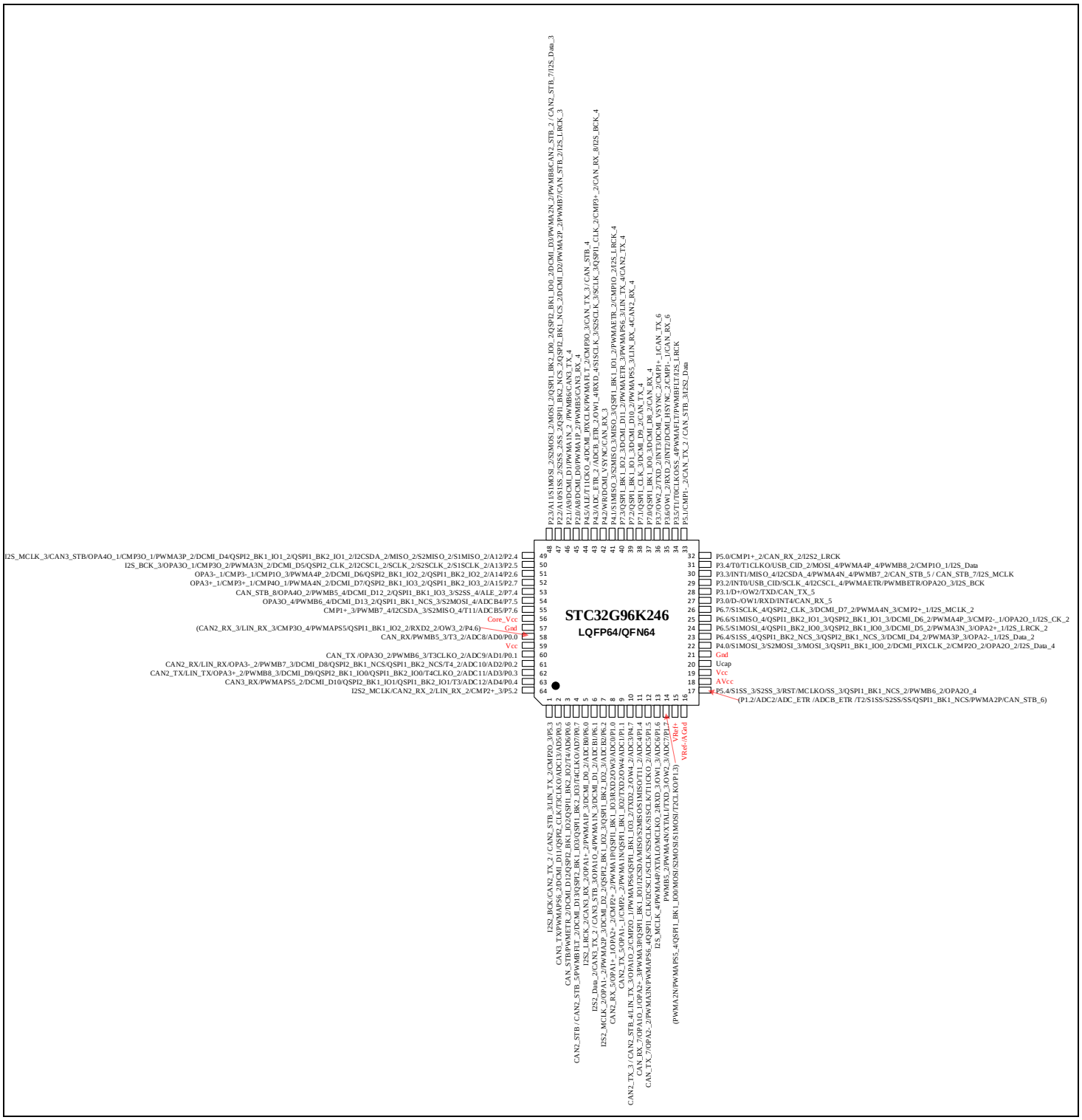
➢ **封装**

- ✓ LQFP100、LQFP64、LQFP48

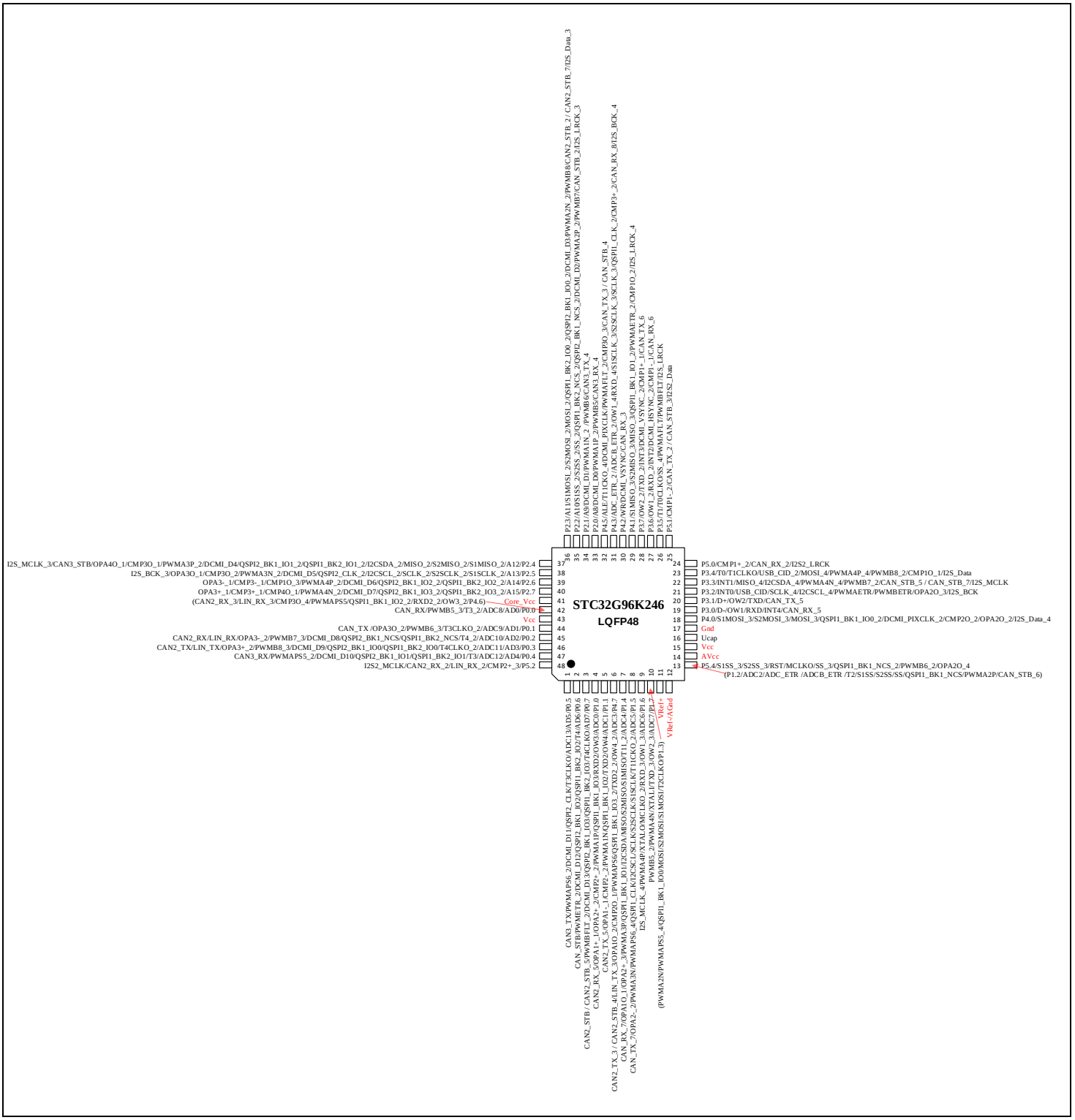
4.1.2 管脚图（LQFP100）



4.1.3 管脚图（LQFP64）



4.1.4 管脚图（LQFP48）



4.1.5 多功能管脚切换

USART1_LIN

RxD	TxD
P3.0	P3.1
P3.6	P3.7
P1.6	P1.7
P4.3	P4.4

USART1_SPI （MOSI和MISO可切换）

U1SS	U1MOSI	U1MISO	U1SCLK
P1.2	P1.3	P1.4	P1.5
P2.2	P2.3	P2.4	P2.5
P5.4	P4.0	P4.1	P4.3
P6.4	P6.5	P6.6	P6.7

PWMA（可互补对称带死区输出，每路均可独立选择）

PWMA1P	PWMA1N	PWMA2P	PWMA2N	PWMA3P	PWMA3N	PWMA4P	PWMA4N	PWMAPS5	PWMAPS6	PWMAETR	PWMAFLT
P1.0	P1.1	P1.2	P1.3	P1.4	P1.5	P1.6	P1.7	P4.6	P4.7	P3.2	P3.5
P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7	P0.4	P0.5	P4.1	P4.5
P6.0	P6.1	P6.2	P6.3	P6.4	P6.5	P6.6	P6.7	P7.2	P7.3	P7.3	-
-	-	-	-	-	-	P3.4	P3.3	P1.3	P1.5	-	-

PWMB（单端输出，每路均可独立选择）

PWMB5P	PWMB6P	PWMB7P	PWMB8P	PWMBETR	P
P2.0	P2.1	P2.2	P2.3	P3.2	
P1.7	P5.4	P3.3	P3.4	P0.6	
P0.0	P0.1	P0.2	P0.3	-	
P7.4	P7.5	P7.6	P7.7	-	

USART2_LIN

RxD2	TxD2
P1.0	P1.1
P4.6	P4.7
P8.4	P8.5
PA.0	PA.1

USART2_SPI （MOSI和MISO可切换）

U2SS	U2MOSI	U2MISO	U2SCLK
P1.2	P1.3	P1.4	P1.5
P2.2	P2.3	P2.4	P2.5
P5.4	P4.0	P4.1	P4.3
P7.4	P7.5	P7.6	P7.7

SPI （MOSI和MISO可切换）

SS	MOSI	MISO	SCLK
P1.2	P1.3	P1.4	P1.5
P2.2	P2.3	P2.4	P2.5
P5.4	P4.0	P4.1	P4.3
P3.5	P3.4	P3.3	P3.2

I2S（第1个I2S模块）

MCLK	BCK	LRCK	Data
P3.3	P3.2	P3.5	P3.4
P6.7	P6.6	P6.5	P6.4
P2.4	P2.5	P2.2	P2.3
P1.6	P4.3	P4.1	P4.0

I2S2（第2个I2S模块）

MCLK2	BCK2	LRCK2	Data2
P5.2	P5.3	P5.0	P5.1
P6.2	P6.3	P6.0	P6.1
P8.2	P8.3	P8.0	P8.1
PB.3	PB.4	PB.1	PB.2

QSPI1_BK1

NCS	IO0	IO1	SCLK	
P1.2	P1.3	P1.4	P1.5	
P5.4	P4.0	P4.1	P4.3	
P7.5	P7.0	P7.2	P7.1	
-	-	-	-	

单总线（OneWire）

OW1	OW2	OW3	OW4
P3.0	P3.1	P1.0	P1.1
P3.6	P3.7	P4.6	P4.7
P1.6	P1.7	P8.4	P8.5
P4.3	P4.4	PA.0	PA.1

OPA1（输入可独立选择）

OP1+	OP1-	OP1O
P1.0	P1.1	P1.4
P6.0	P6.2	P4.7
-	P8.6	P8.7
-	-	P6.1

OPA2（输入可独立选择）

OP2+	OP2-	OP2O
P6.5	P6.4	P6.6
P1.0	P1.5	P4.0
P1.4	-	P3.2
-	-	P5.4

OPA3（输入可独立选择）

OP3+	OP3-	OP3O
P2.7	P2.6	P2.5
P0.3	P0.2	P0.1
-	-	PB.0
-	-	P7.5

OPA4（输入可独立选择）

OP4+	OP4-	OP4O
-	-	P2.4
-	-	P7.4
-	-	PA.3
-	-	PB.1

QSPI1_BK2

NCS	IO0	IO1	SCLK	
P0.2	P0.3	P0.4	-	
P2.2	P2.3	P2.4	-	
P6.4	P6.5	P6.6	-	
PA.0	PA.1	PA.2	-	

I2C

SCL	SDA
P1.5	P1.4
P2.5	P2.4
P7.7	P7.6
P3.2	P3.3

LIN

LINRx	LINTx
P0.2	P0.3
P5.2	P5.3
P4.6	P4.7
P7.2	P7.3

CMP（第1个比较器）

CMP1+	CMP1-	CMP1O
P3.7	P3.6	P3.4
P5.0	P5.1	P4.1
P7.6	-	P2.6
-	-	-
-	-	-
-	-	-

CMP2（第2个比较器）

CMP2+	CMP2-	CMP2O
P6.7	P6.6	P4.7
P1.0	P1.1	P4.0
P5.2	-	P5.3
-	-	-
-	-	-
-	-	-

CMP3（第3个比较器）

CMP3+	CMP3-	CMP3O
P2.7	P2.6	P2.4
P4.3	P4.4	P2.5
P7.7	-	P4.5
-	-	P4.6
-	-	-
-	-	-

QSPI2

NCS	IO0	IO1	SCLK	
P0.2	P0.3	P0.4	P0.5	
P2.2	P2.3	P2.4	P2.5	
P6.4	P6.5	P6.6	P6.7	
PA.0	PA.1	PA.2	PA.3	

CAN_FD

CANRx	CANTx	CANSTB
P0.0	P0.1	P0.6
P5.0	P5.1	P2.2
P4.2	P4.5	P5.1
P7.0	P7.1	P4.5
P3.0	P3.1	P3.3
P3.6	P3.7	P1.2
P1.4	P1.5	P3.3
P4.3	P4.4	P7.4

CAN_FD2

CAN2Rx	CAN2Tx	CAN2STB
P0.2	P0.3	P0.7
P5.2	P5.3	P2.3
P4.6	P4.7	P5.3
P7.2	P7.3	P4.7
P1.0	P1.1	P0.7
P8.4	P8.5	P8.3
PA.0	PA.1	P2.3
-	-	-

CAN_FD3

CAN3Rx	CAN3Tx	CAN3STB
P0.4	P0.5	P2.4
P6.0	P6.1	P7.7
P9.6	P9.7	P6.1
P2.0	P2.1	P9.7
-	-	-
-	-	-
-	-	-
-	-	-

ADC（第一组ADC，12位*16通道）

ADC0	ADC1	ADC2	ADC3	ADC4	ADC5	ADC6	ADC7
P1.0	P1.1	P1.2	P4.7	P1.4	P1.5	P1.6	P1.7
ADC8	ADC9	ADC10	ADC11	ADC12	ADC13	ADC14	ADC15
P0.0	P0.1	P0.2	P0.3	P0.4	P0.5	OP1O	1.19V

ADCB（第二组ADC，12位*16通道）

ADC0	ADC1	ADC2	ADC3	ADC4	ADC5	ADC6	ADC7
P6.0	P6.1	P6.2	P6.3	P7.5	P7.6	P7.7	PB.0
ADC8	ADC9	ADC10	ADC11	ADC12	ADC13	ADC14	ADC15
PB.1	PB.2	PB.3	PB.4	PB.5	PB.6	OP2O	1.19V

TFT_i8080/M6800控制脚

RS	WR	RD
P4.5	P4.2	P4.4
P4.5	P3.6	P3.7
P4.0	P4.2	P4.4
P4.0	P3.6	P3.7

TFT_i8080/M6800 8位数据

D0	D1	D2	D3	D4	D5	D6	D7
P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
P6.0	P6.1	P6.2	P6.3	P6.4	P6.5	P6.6	P6.7
P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
P6.0	P6.1	P6.2	P6.3	P6.4	P6.5	P6.6	P6.7

TFT_i8080/M6800控制脚

RS	WR	RD
P4.5	P4.2	P4.4
P4.5	P3.6	P3.7
P4.0	P4.2	P4.4
P4.0	P3.6	P3.7

TFT_i8080/M6800 16位数据

D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15
P0.0	P0.1	P0.2	P0.3	P0.4	P0.5	P0.6	P0.7	P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7	P6.0	P6.1	P6.2	P6.3	P6.4	P6.5	P6.6	P6.7
P4.1	P4.3	P4.6	P4.7	P0.4	P0.5	P0.6	P0.7	P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
P7.0	P7.1	P7.2	P7.3	P7.4	P7.5	P7.6	P7.7	P6.0	P6.1	P6.2	P6.3	P6.4	P6.5	P6.6	P6.7

定时器

T0/CLKO	T1/CLKO	T2/CLKO	T3/CLKO	T4/CLKO	T11/CLKO	T
P3.4/P3.5	P3.5/P3.4	P1.2/P1.3	P0.4/P0.5	P0.6/P0.7	P7.6/P7.7	
-	-	-	P0.0/P0.1	P0.2/P0.3	P1.4/P1.5	
-	-	-	-	-	PA.6/PA.7	
-	-	-	-	-	P4.4/P4.5	

DCMI接口

PIXCLK	HSYNC	VSYNC	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13
P4.5	P4.4	P4.2	P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7	P0.2	P0.3	P0.4	P0.5	P0.6	P0.7
P4.0	P3.6	P3.7	P6.0	P6.1	P6.2	P6.3	P6.4	P6.5	P6.6	P6.7	P7.0	P7.1	P7.2	P7.3	P7.4	P7.5
PB.0	P9.6	P9.7	P8.0	P8.1	P8.2	P8.3	P8.4	P8.5	P8.6	P8.7	P9.0	P9.1	P9.2	P9.3	P9.4	P9.5
P9.0	PB.6	PB.7	PA.0	PA.1	PA.2	PA.3	PA.4	PA.5	PA.6	PA.7	PB.0	PB.1	PB.2	PB.3	PB.4	PB.5

5 STC32G18K64-64MHz-LQFP48/32，TSSOP20 系列提前预告

5.1.1 特性及价格

➢ 选型价格（不需要外部晶振、不需要外部复位，两组 12 位 ADC，15+15 通道，对外 28 个通道）

单片机型号	工作电压（V）	Flash 程序存储器 10 万次 字节	edata 内部扩展 DATA RAM 可做堆栈或变量 字节	xdata 内部大容量扩展 SRAM 可做变量 字节	EEPROM 10 万次 字节	I/O 口最多数量	传统 I/O 中断（INT0/INT1/INT2/INT3/INT4） 并可掉电唤醒	所有 I/O 口均支持高电平 低电平 上升沿 下降沿中断，可掉电唤醒	DMA 8080/6800 接口 TFT 彩屏模块驱动(8 位和 16 位)	RTC 实时时钟 年/月/日/时/分/秒/星期 对内部时钟追频 掉电唤醒	USB-FS OTG，支持 USB-CDC，USB-HID	DMA USART-LIN / SPI / IrDA 并可掉电唤醒	汽车电子通讯 LIN 总线	单总线（支持 WS2812 / DS18B20 / DHT11 / NEGIR / UART）	DMA I2S 音频总线 掉电唤醒（全双工录音 + 放音）	DMA SPI 并可掉电唤醒	DMA QSPI	DMA I²C 并可掉电唤醒	DMA 汽车电子通讯 CAN-FD 总线 并可掉电唤醒	TFPU (144MHz)硬件浮点 / 硬件三角 / 反三角函数运算器	DSP32 （64MHz~72MHz），硬件 32 位/64 位 DSP 指令	定时器/计数器（T0～T4,T11, T17/T18 外部管脚 也可掉电唤醒）	144MHz，16 位高级 PWM 定时器 互补对称死区控制	掉电唤醒 专用定时器	DMA 支持两个独立的 12 位 ADC，每个 16 通道，共 32 通道	DMA 支持两个独立的 12 位高速 DAC	比较器（6P6N 模式，6 个正极输入，6 个负极输入）	运算放大器（OPA）	内部低压检测中断 并可掉电唤醒	看门狗 复位定时器	内部高可靠复位（可选复位门槛电压）	内部高精准时钟（64MHz 以下可调），满足串口通信 追频	可对外输出时钟及复位	程序加密后传输（防拦截），可设置下次更新程序需口令	支持 RS485 下载	本身就可在线仿真，支持硬件 USB 直接下载和硬件 USB 仿真	价格及封装			供货信息
																																						LQFP48	LQFP32	TSSOP20	
STC32G18K64	1.9-5.5	64K	2K	16K	IAP	42	有	有	有	有	有	4	1	2	2	1+4	2	1	1	有	有	8	8	有	2 组	2 组	3 组	4 组	有	有	4 级	有	是	是	是	是				

可实现 4+1 组 LIN

➢ **内核**

- ✓ 64MHz CPU/DSP，144MHz TFPU/PWM
- ✓ 超高速 32 位 8051 内核（1T），比传统 8051 同频快 70 倍以上
- ✓ 4 级中断优先级
- ✓ 支持在线仿真

➢ **工作电压**

- ✓ 1.9V ～ 5.5V（当工作温度低于-40℃时，工作电压不得低于 3.0V，低温时工作电压要提高）



扫码去微信小商城

➢ **工作温度**

- ✓ -20℃～65℃（内部高速 IRC 温漂-0.76%~+0.98%）
- ✓ -40℃～85℃（内部高速 IRC 温漂±1.3%）
- ✓ -40℃～125℃（内部高速 IRC 温漂±3%，当温度高于 85℃时请使用外部 35MHz 及以下的晶振）

➢ **Flash 存储器**

- ✓ 最大 64K 字节 FLASH 程序存储器（ROM），用于存储用户代码
- ✓ 支持用户可配置部分 FLASH 作为 DATA FLASH/EEPROM 使用，512 字节单页擦除，**典型单个扇区擦除次数可达 10 万次，用户保守可当 2 万次使用**
- ✓ 支持硬件 USB 直接下载和普通串口下载
- ✓ 支持硬件 SWD 实时仿真，P3.0/P3.1（需 USB-Link1D 工具）
- ✓ **Flash 数据保持时间：100 年@25℃，20 年@105℃，10 年@125℃。由于汽车及单片机的平均温度不可能长时间大于 105℃，从严考虑，Flash 数据保持时间可达 20 年以上，不保证 100 年。**

➢ **SRAM，共 18K 字节**

- ✓ 2K 字节内部 SRAM（edata）
- ✓ 16K 字节内部扩展 RAM（内部 xdata）

- **时钟控制**
  - ✓ 内部高精度、高稳定的高速 IRC（ISP 编程时可进行上下调整）
    - ⌖ 误差±0.3%（常温下 25℃）
    - ⌖ -1.35% ～ +1.30%温漂（全温度范围，-40℃ ～ 85℃）
    - ⌖ -0.76% ～ +0.98%温漂（温度范围，-20℃ ～ 65℃）
  - ✓ 内部 32KHz 低速 IRC（为了低功耗，省去了温度补偿和电压补偿电路，误差较大）
  - ✓ 外部晶振（4MHz ～ 35MHz）和外部时钟，有专门的外部时钟干扰内部电路，可软件启动
  - ✓ **内部两组独立的 96MHz/144MHz 的 PLL，可输出的时钟提供给 TFPU、PWM、I2S、SPI 等高速外设**

用户可自由选择上面的 4 种时钟源

（芯片上电工作过程：上电复位/复位脚复位/看门狗复位/低压检测复位时，芯片默认从 ISP 系统程序开始执行代码，此时固定使用内部 24MHz 的高速 IRC 时钟，当需要下载用户程序且下载完成后复位到用户程序区或者不需要下载直接复位到用户程序区时，默认会使用上次用户下载时所调节的高速 IRC 时钟，如果用户程序需要使用外部高速晶振、外部 32.768KHz 晶振或者内部 30KHz 低速 IRC，则需要用户软件先启动相应的时钟，然后通过设置 CLKSEL 寄存器进行切换）

- **复位**
  - ✓ 硬件复位
    - ⌖ 上电复位，复位电压值为 1.7V ～ 1.9V。（在芯片未使能低压复位功能时有效）
    - ⌖ 复位脚复位，出厂时 P5.4 默认为 I/O 口，ISP 下载时可将 P5.4 管脚设置为复位脚（注意：当设置 P5.4 管脚为复位脚时，复位电平为低电平）
    - ⌖ 看门狗溢出复位
    - ⌖ 低压检测复位，提供 4 级低压检测电压：2.0V、2.4V、2.7V、3.0V。
  - ✓ 软件复位
    - ⌖ 软件方式写复位触发寄存器

- **中断**
  - ✓ 中断源：INT0、INT1、INT2、INT3、INT4、定时器 0、定时器 1、定时器 2、定时器 3、定时器 4、定时器 11、定时器 17、定时器 18、USART1、USART2、USART3、USART4、ADC 模数转换、ADC2 模数转换、DAC1 数模转换、DAC2 数模转换、LVD 低压检测、SPI、QSPI1、QSPI2、I^2C、OW1、OW2、比较器 1、比较器 2、比较器 3、PWMA、PWMB、USB、TFT 彩屏接口中断、RTC 实时时钟、所有的 I/O 中断（8 组）、所有的 USART 串口的 DMA 接收和发送中断（4 组）、I2C 的 DMA 接收和发送中断、I2S 的 DMA 接收和发送中断（2 组）、SPI 的 DMA 中断、QSPI 的 DMA 中断（2 组）、OW 的 DMA 中断（2 组）、ADC 的 DMA 中断（2 组）、DAC 的 DMA 中断（2 组）、LCD 驱动的 DMA 中断、CAN-FD 的 DMA 中断（1 组）、PWM 的 DMA 中断（PWMA）、存储器到存储器的 DMA 中断以及**两组外设到外设的 DMA 中断**。
  - ✓ 提供 4 级中断优先级

- **数字外设**
  - ✓ 8 个 16 位定时器：定时器 0、定时器 1、定时器 2、定时器 3、定时器 4、定时器 11、定时器 17、定时器 18，其中定时器 0 的模式 3 具有 NMI（不可屏蔽中断）功能，定时器 0 和定时器 1 的模式 0 为 16 位自动重载模式。定时器 11：可选时钟源。**定时器 17 和定时器 18 可同步触发两组 ADC 或者同步触发两组 DAC**。
  - ✓ 4 个高速同步/异步串口：串口 1（USART1）、串口 2（USART2）、串口 3（USART3）、串口 4（USART4），波特率时钟源最快可为 FOSC/4。支持同步串口模式、异步串口模式、SPI 模式、LIN 模式、红外模式（IrDA）、智能卡模式（ISO7816）
  - ✓ 8 通道高级 PWM，1+1 组高级 PWM，每组均可实现 4 通道 PWM。PWMA 可实现 4 对互补对称带死区控制的 PWM，PWMB 可实现 4 通道 PWM 输出，并支持外部异常检测功能。
  - ✓ **2 个 QSPI：**支持单线、双线和四线模式。QSPI1 支持同时读 2 个 QSPI，一次性读 8 位；支持运行 QSPI-FLASH

中的外部程序。QSPI2 是传统 QSPI，不支持运行 QSPI-FLASH 中的外部程序。

- ✓ SPI：1+4 组硬件 SPI（1 组独立 SPI，4 组 USART 的 SPI 模式）支持主机模式和从机模式以及主机/从机自动切换（注：5 组 SPI 均支持 DMA，5 组 SPI 的 MISO 和 MOSI 脚均可软件设置进行功能交换）。
- ✓ I^2C：独立的硬件 I^2C，支持主机模式和从机模式
- ✓ ICE：支持硬件仿真，支持串口和 USB 口直接仿真，还支持 SWD 硬件仿真
- ✓ RTC：支持年、月、日、星期、时、分、秒、次秒（1/128 秒），并支持时钟中断和一组闹钟。芯片复位时 RTC 的时间计数值不复位。
- ✓ USB-FS OTG：USB2.0/USB1.1 兼容全速 USB，**16 个双向端点**，支持 4 种端点传输模式（控制传输、中断传输、批量传输和同步传输），端点拥有 64/128/256/512/1024 字节的 FIFO 缓冲区，不同的端点的 FIFO 缓冲区大小不等。由于已支持 OTG，可当 Host 使用。
- ✓ CAN：1 个独立的 CAN-FD 2.0 控制单元。CAN-FD 的 STB 脚可软件设置是否使能，若不使能 STB 脚可当作 GPIO 使用。
- ✓ LIN：1+4 组硬件 LIN（1 组组独立 LIN，4 组 USART 的 LIN 模式），一个独立的 LIN 控制单元（支持 1.3 和 2.1 版本）
- ✓ I2S：两路独立的 I2S 音频总线，**可支持全双工方式的录音和放音**
- ✓ DSP32：硬件 32 位/64 位 DSP 运算指令
- ✓ TFPU：单精度浮点运算器（支持浮点加、减、乘、除以及正弦、余弦、正切和反正切等运算）
- ✓ I/O 口中断：所有的 I/O 均支持中断，每组 I/O 中断有独立的中断入口地址，所有的 I/O 中断可支持 5 种中断模式：高电平中断、低电平中断、上升沿中断、下降沿中断以及上升沿+下降沿中断。I/O 口中断可以进行掉电唤醒，且有 4 级中断优先级。
- ✓ LCD/TFT 彩屏驱动接口：支持 8080 和 6800 两种接口以及 8 位和 16 位数据宽度。对 16 位数据支持大小端数据格式选择。
- ✓ **专门的 DCMI 数字摄像头接口：**支持 8 位、10 位、12 位和 14 位数据接口，支持外部行同步和帧同步
- ✓ DMA：支持 SPI（1 组）、QSPI（两组）、I2C（1 组）、I2S（两组）、串口（4 组）、OW（2 组）、PWM（1 组，PWMA）、ADC（两组）、DAC（两组）、CAN-DA（1 组）、LCM 的 DMA，以及外设到外设直接的 DMA
- ✓ 硬件数字 ID：支持 32+32 字节

➢ 模拟外设

- ✓ ADC：**2 组**独立的高速 ADC，支持 12 位高精度 15 通道（通道 0～通道 14）的模数转换，ADC 的通道 15 用于测试内部参考电压（芯片在出厂时，内部参考电压调整为 1.19V，误差±1%）
- ✓ DAC：两组独立的高速 DAC，支持 12 位转换精度。两组 DAC 的模拟输出均从芯片内部直接连接到比较器和运算放大器。
- ✓ 比较器：3 组独立的 6P6N 模式比较器（CMP）
- ✓ 运算放大器：4 组独立的运算放大器（OPA）

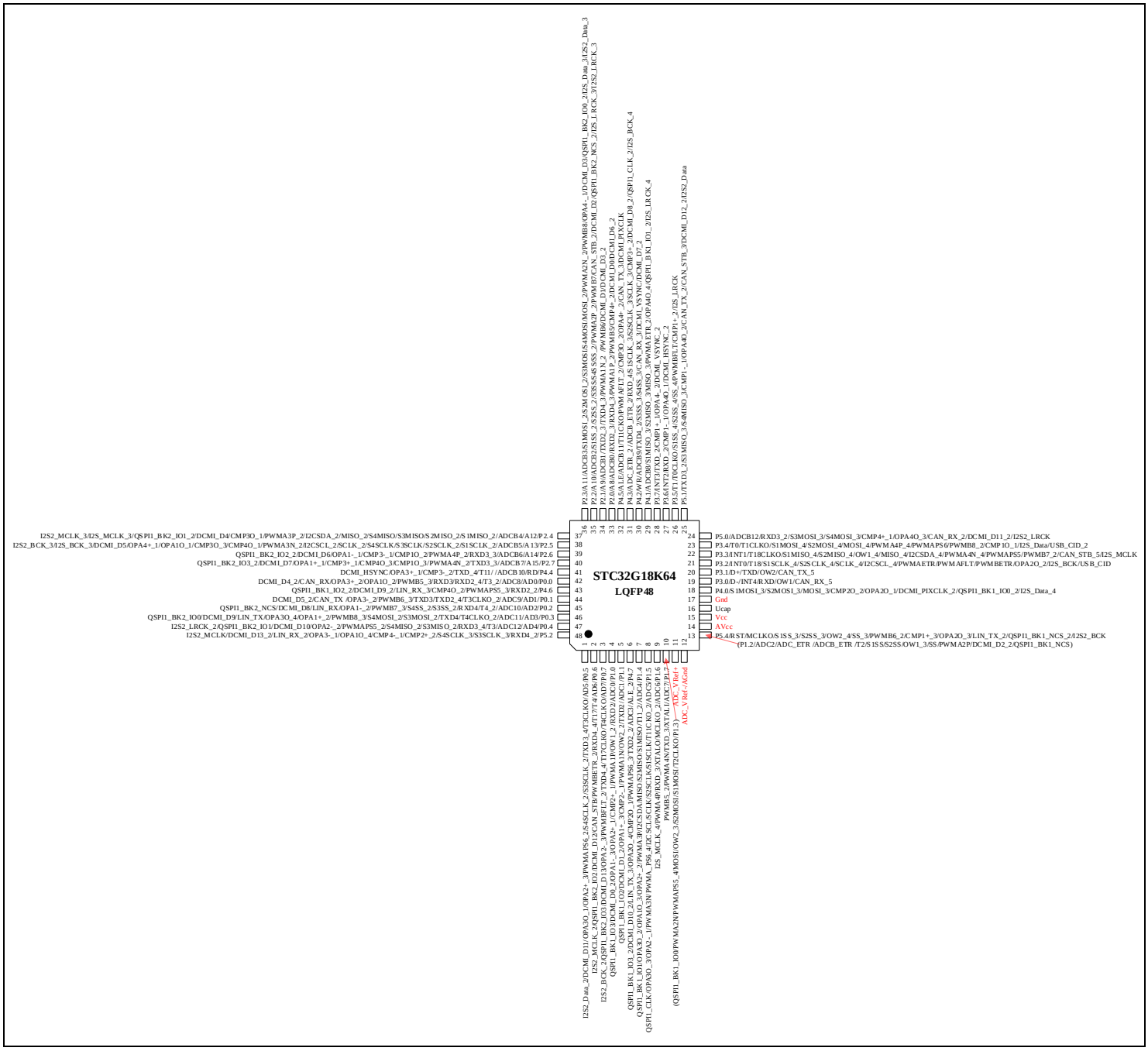
➢ GPIO

- ✓ 最多可达 42 个 GPIO：P0.0 ~ P0.7、P1.0 ~ P1.7（P1.2 和 P5.4 为同一个 PIN 脚、P1.3 和 P1.7 为同一个 PIN 脚）、P2.0 ~ P2.7、P3.0 ~ P3.7、P4.0 ~ P4.7、P5.0 ~ P5.4（无 P5.3）
- ✓ 所有的 GPIO 均支持如下模式：**无弱上拉电阻的准双向口模式**、强推挽输出模式、开漏模式、高阻输入模式
- ✓ 所有 I/O 口上电后的状态均为高阻输入状态，用户在使用 I/O 口时必须先设置 I/O 口模式
- ✓ 另外每个 I/O 均可独立使能内部 10K 上拉电阻和 10K 下拉电阻

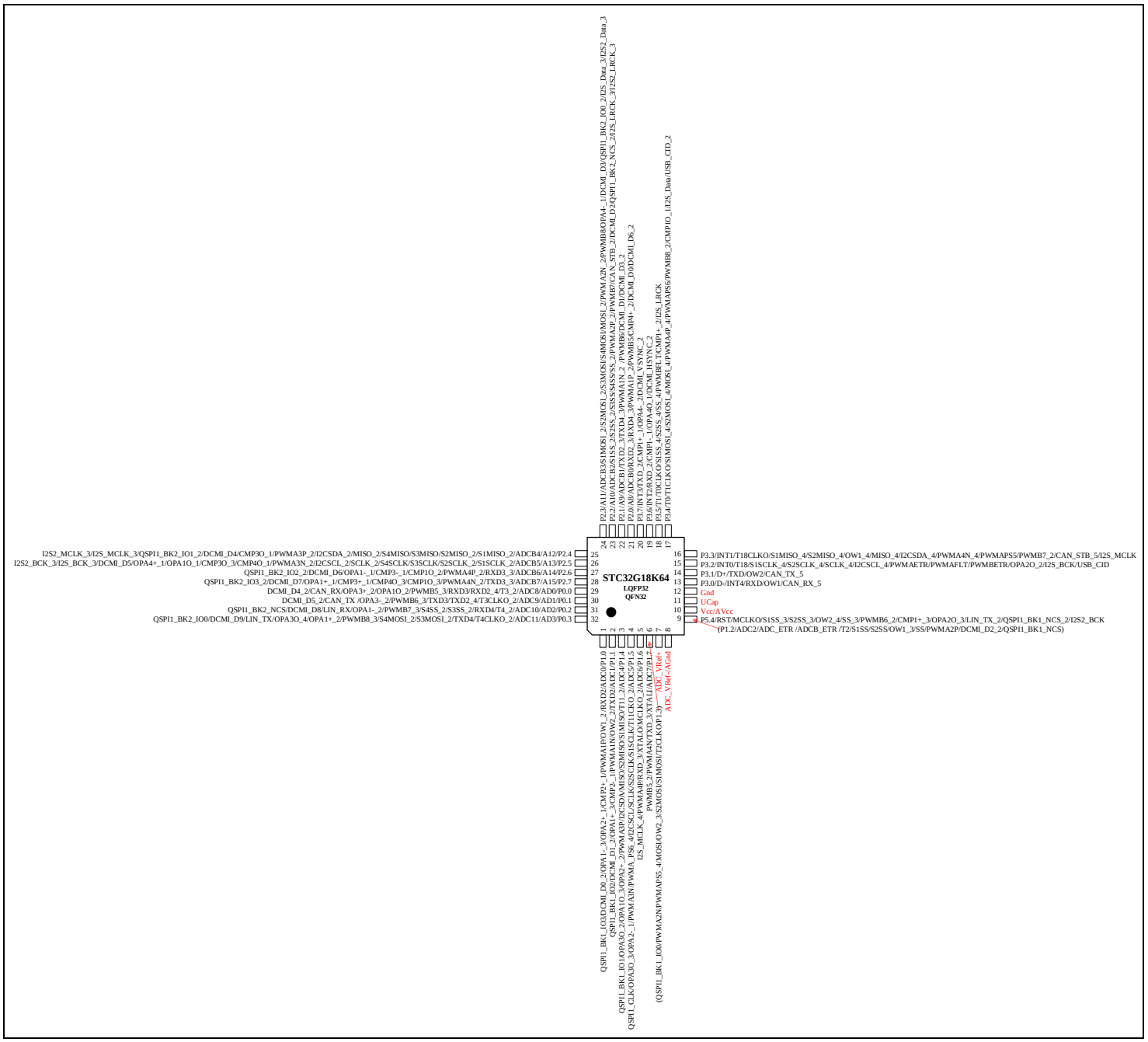
➢ **封装**

- ✓ LQFP48、LQFP32、TSSOP20

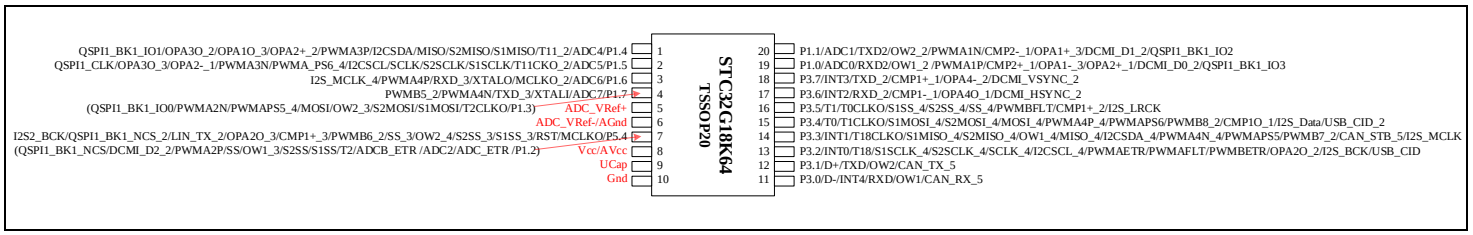
5.1.2 管脚图（LQFP48）



5.1.3 管脚图（LQFP32）



5.1.4 管脚图（TSSOP20）



5.1.5 多功能管脚切换

USART1-LIN

RxD	TxD
P3.0	P3.1
P3.6	P3.7
P1.6	P1.7
P4.3	P4.4

USART1_SPI （MOSI和MISO可切换）

U1SS	U1MOSI	U1MISO	U1SCLK
P1.2	P1.3	P1.4	P1.5
P2.2	P2.3	P2.4	P2.5
P5.4	P4.0	P4.1	P4.3
P3.5	P3.4	P3.3	P3.2

PWMA（可互补对称带死区输出，每路均可独立选择）

PWMA1P	PWMA1N	PWMA2P	PWMA2N	PWMA3P	PWMA3N	PWMA4P	PWMA4N	PWMAPS5	PWMAPS6	PWMAETR	PWMAFLT
P1.0	P1.1	P1.2	P1.3	P1.4	P1.5	P1.6	P1.7	P3.3	P3.4	P3.2	P3.2
P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7	P0.4	P0.5	P4.1	P4.5
-	-	-	-	-	-	-	-	P4.6	P4.7	-	-
-	-	-	-	-	-	P3.4	P3.3	P1.3	P1.5	-	-

PWMB（单端输出，每路均可独立选择）

PWMB5P	PWMB6P	PWMB7P	PWMB8P	PWMBETR	PWMBFLT
P2.0	P2.1	P2.2	P2.3	P3.2	P3.5
P1.7	P5.4	P3.3	P3.4	P0.6	P0.7
P0.0	P0.1	P0.2	P0.3	-	-
-	-	-	-	-	-

USART2-LIN

RxD2	TxD2
P1.0	P1.1
P4.6	P4.7
P2.0	P2.1
P0.0	P0.1

USART2_SPI （MOSI和MISO可切换）

U2SS	U2MOSI	U2MISO	U2SCLK
P1.2	P1.3	P1.4	P1.5
P2.2	P2.3	P2.4	P2.5
P5.4	P4.0	P4.1	P4.3
P3.5	P3.4	P3.3	P3.2

SPI （MOSI和MISO可切换）

SS	MOSI	MISO	SCLK
P1.2	P1.3	P1.4	P1.5
P2.2	P2.3	P2.4	P2.5
P5.4	P4.0	P4.1	P4.3
P3.5	P3.4	P3.3	P3.2

I2S（第1个I2S模块）

MCLK	BCK	LRCK	Data
P3.3	P3.2	P3.5	P3.4
-	-	-	-
P2.4	P2.5	P2.2	P2.3
P1.6	P4.3	P4.1	P4.0

I2S2（第2个I2S模块）

MCLK2	BCK2	LRCK2	Data2
P5.2	P5.4	P5.0	P5.1
P0.6	P0.7	P0.4	P0.5
P2.4	P2.5	P2.2	P2.3
-	-	-	-

QSPI1_BK1

NCS	IO0	IO1	SCLK	IO2	IO3
P1.2	P1.3	P1.4	P1.5	P1.1	P1.0
P5.4	P4.0	P4.1	P4.3	P4.6	P4.7
-	-	-	-	-	-
-	-	-	-	-	-

USART3-LIN

RxD3	TxD3
P0.0	P0.1
P5.0	P5.1
P2.6	P2.7
P0.4	P0.5

USART3_SPI （MOSI和MISO可切换）

U3SS	U3MOSI	U3MISO	U3SCLK
P2.2	P2.3	P2.4	P2.5
P0.2	P0.3	P0.4	P0.5
P4.2	P5.0	P5.1	P5.2
-	-	-	-

OPA1（输入可独立选择）

OP1+	OP1-	OP1O
P2.7	P2.6	P2.5
P0.3	P0.2	P0.0
P1.1	P1.0	P1.4
DAC1O	-	P5.2

OPA2（输入可独立选择）

OP2+	OP2-	OP2O
P1.0	P1.5	P4.0
P1.4	P0.4	P3.2
P0.5	P0.7	P5.4
DAC2O	-	P4.7

OPA3（输入可独立选择）

OP3+	OP3-	OP3O
P4.4	P5.2	P0.5
P0.0	P0.1	P1.4
OP1O	OP2O	P1.5
DAC2O	-	P0.3

OPA4（输入可独立选择）

OP4+	OP4-	OP4O
P2.5	P2.3	P3.6
P4.5	P3.7	P5.1
OP2O	OP1O	P5.0
DAC1O	-	P4.1

QSPI1_BK2

NCS	IO0	IO1	SCLK	IO2	IO3
P0.2	P0.3	P0.4	-	P0.6	P0.7
P2.2	P2.3	P2.4	-	P2.6	P2.7
-	-	-	-	-	-
-	-	-	-	-	-

USART4-LIN

RxD4	TxD4
P0.2	P0.3
P5.2	P4.2
P2.0	P2.1
P0.6	P0.7

USART4_SPI （MOSI和MISO可切换）

U4SS	U4MOSI	U4MISO	U4SCLK
P2.2	P2.3	P2.4	P2.5
P0.2	P0.3	P0.4	P0.5
P4.2	P5.0	P5.1	P5.2
-	-	-	-

CMP（第1个比较器）

CMP1+	CMP1-	CMP1O
P3.7	P3.6	P3.4
P3.5	P5.1	P2.6
P5.4	OP2O	P2.7
OP1O	1.19V	-
ADCIN	DAC1O	-
DAC2O	DAC2O	-

CMP2（第2个比较器）

CMP2+	CMP2-	CMP2O
P1.0	P1.1	P4.7
P5.2	-	P4.0
-	OP3O	-
OP2O	1.19V	-
ADCIN	DAC1O	-
DAC1O	DAC2O	-

CMP3（第3个比较器）

CMP3+	CMP3-	CMP3O
P2.7	P2.6	P2.4
P4.3	P4.4	P4.5
-	OP4O	P2.5
OP3O	1.19V	-
ADCBIN	DAC1O	-
DAC2O	DAC2O	-

CAN-FD

CANRx	CANTx	CANSTB
P0.0	P0.1	P0.6
P5.0	P5.1	P2.2
P4.2	P4.5	P5.1
-	-	-
P3.0	P3.1	P3.3
-	-	-
-	-	-
-	-	-

LIN

LINRx	LINTx
P0.2	P0.3
P5.2	P5.4
P4.6	P4.7
-	-

I2C

SCL	SDA
P1.5	P1.4
P2.5	P2.4
-	-
P3.2	P3.3

单总线（OneWire）

OW1	OW2
P3.0	P3.1
P1.0	P1.1
P1.2	P1.3
P3.3	P5.4

ADC（第一组ADC，12位*16通道）

ADC0	ADC1	ADC2	ADC3	ADC4	ADC5	ADC6	ADC7
P1.0	P1.1	P1.2	P4.7	P1.4	P1.5	P1.6	P1.7
ADC8	ADC9	ADC10	ADC11	ADC12	ADC13	ADC14	ADC15
P0.0	P0.1	P0.2	P0.3	P0.4	OP3O	OP1O	1.19V

DAC1

输出
DAC1O

DAC2

输出
DAC2O

ADCB（第二组ADC，12位*16通道）

ADC0	ADC1	ADC2	ADC3	ADC4	ADC5	ADC6	ADC7
P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
ADC8	ADC9	ADC10	ADC11	ADC12	ADC13	ADC14	ADC15
P4.1	P4.2	P4.4	P4.5	P5.0	OP4O	OP2O	1.19V

TFT-i8080/M6800控制脚

RS	WR	RD
P4.5	P4.2	P4.4
P4.5	P3.6	P3.7
P4.0	P4.2	P4.4
P4.0	P3.6	P3.7

TFT-i8080/M6800 8位数据

D0	D1	D2	D3	D4	D5	D6	D7
P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
-	-	-	-	-	-	-	-
P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
-	-	-	-	-	-	-	-

TFT-i8080/M6800控制脚

RS	WR	RD
P4.5	P4.2	P4.4
P4.5	P3.6	P3.7
P4.0	P4.2	P4.4
P4.0	P3.6	P3.7

TFT-i8080/M6800 16位数据

D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15
P0.0	P0.1	P0.2	P0.3	P0.4	P0.5	P0.6	P0.7	P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
P4.1	P4.3	P4.6	P4.7	P0.4	P0.5	P0.6	P0.7	P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

定时器

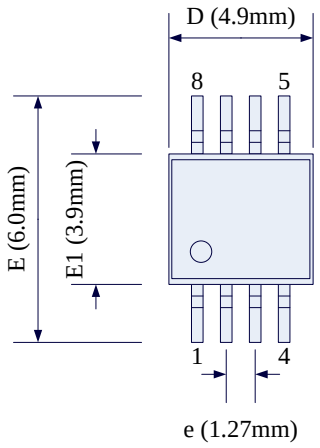
T0/CLKO	T1/CLKO	T2/CLKO	T3/CLKO	T4/CLKO	T11/CLKO	T17/CLKO	T18/CLKO
P3.4/P3.5	P3.5/P3.4	P1.2/P1.3	P0.4/P0.5	P0.6/P0.7	P4.4/P4.5	P0.6/P0.7	P3.2/P3.3
-	-	-	P0.0/P0.1	P0.2/P0.3	P1.4/P1.5	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-

DCMI接口

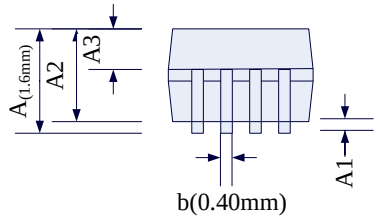
PIXCLK	HSYNC	VSYNC	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13
P4.5	P4.4	P4.2	P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7	P0.2	P0.3	P0.4	P0.5	P0.6	P0.7
P4.0	P3.6	P3.7	-	-	-	P2.1	P0.0	P0.1	P2.0	P4.2	P4.3	P4.6	P4.7	P5.0	P5.1	P5.2
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

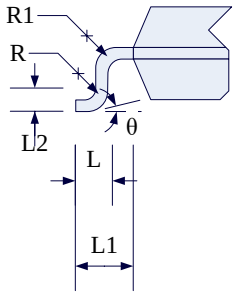
6 封装尺寸图、命名规则、USB-TypeC 封装尺寸图及原理图

6.1 SOP8 封装尺寸图

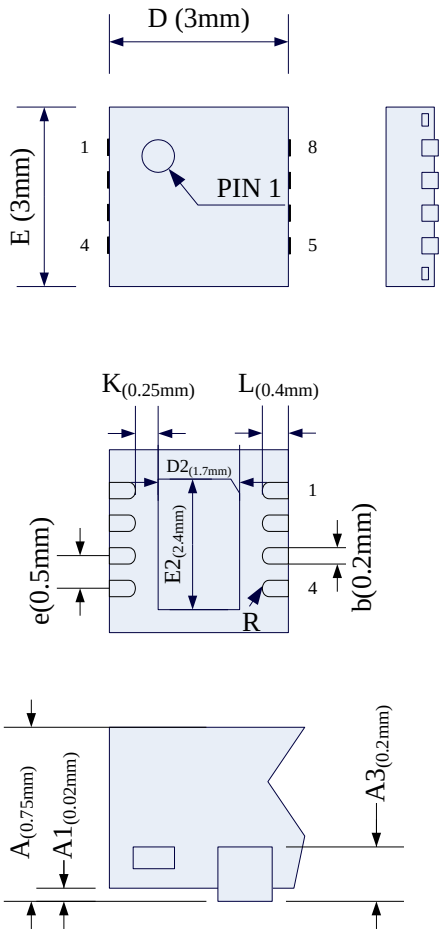


一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	1.35	1.60	1.75
A1	0.10	0.15	0.25
A2	1.25	1.40	1.50
A3	0.55	0.65	0.75
b	0.35	0.40	0.45
D	4.80	4.90	5.00
E	5.80	6.00	6.20
E1	3.80	3.90	4.00
e	1.27BSC		
L	0.45	0.60	0.80
L1	1.04REF		
L2	0.25BSC		
R1	0.07	-	-
R2	0.07	-	-





6.2 DFN8 封装尺寸图（3mm*3mm）

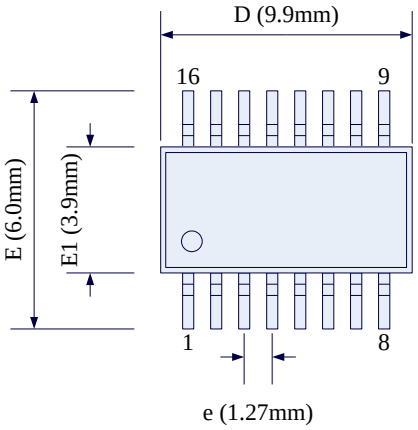


一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	0.70	0.75	0.80
A1	0	0.02	0.05
A2	0.50	0.55	0.60
A3	-	0.203REF	-
b	0.175	0.20	0.225
D	2.89	3.00	3.11
E	2.89	3.00	3.11
D2	1.65	1.70	1.75
E2	2.35	2.40	2.45
e	0.45	0.50	0.55
L	0.381	0.40	0.419
K	0.25REF		
R	0.125	-	-

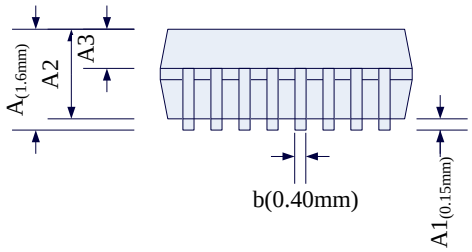
现有 DFN8 封装芯片的背面金属片（衬底），在芯片内部并未接地，在用户的 PCB 板上可以接地，也可以不接地，不会对芯片性能造成影响

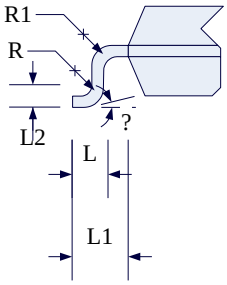
特别说明：造 PCB 的封装库时，焊盘可以向外延伸 0.1～0.2mm，但绝对不能向内延伸，必须确保 K 不能小于 0.25mm，否则会造成管脚短路

6.3 SOP16 封装尺寸图

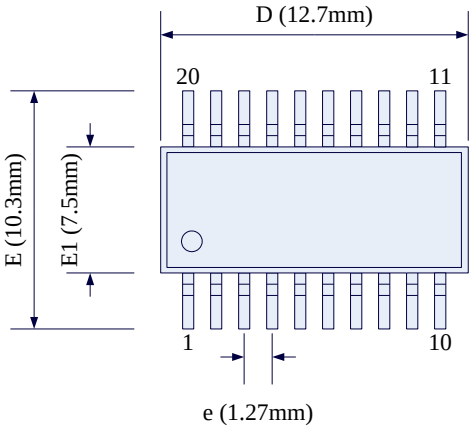


一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	1.35	1.60	1.75
A1	0.10	0.15	0.25
A2	1.25	1.45	1.65
A3	0.55	0.65	0.75
b	0.35	0.40	0.45
D	9.80	9.90	10.00
E	5.80	6.00	6.20
E1	3.80	3.90	4.00
e		1.27BSC	
L	0.45	0.60	0.80
L1		1.04REF	
L2		0.25BSC	
R1	0.07	-	-
R2	0.07	-	-

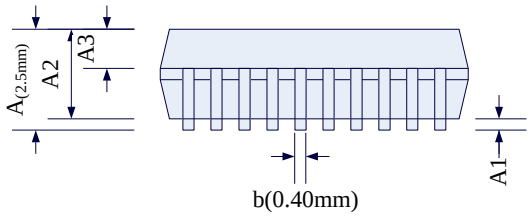


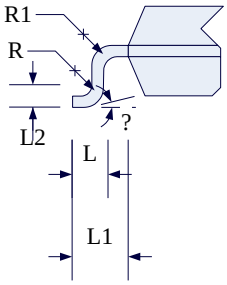


6.4 SOP20 封装尺寸图

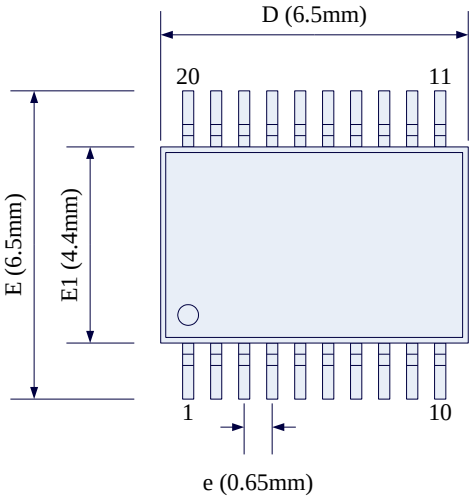


一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	2.465	2.515	2.565
A1	0.10	0.15	0.20
A2	2.10	2.30	2.50
A3	0.55	0.65	0.75
b	0.356	0.406	0.456
D	12.50	12.70	12.90
E	10.20	10.30	10.40
E1	7.45	7.50	7.55
e		1.27BSC	
L	0.80	0.86	0.90
L1		1.40REF	
L2		0.27BSC	
R1	-	0.30	-
R2	-	0.20	-

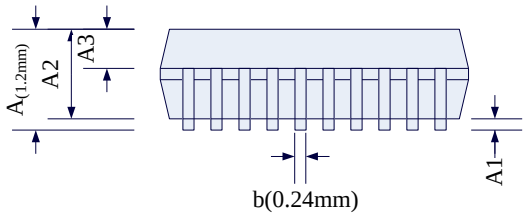


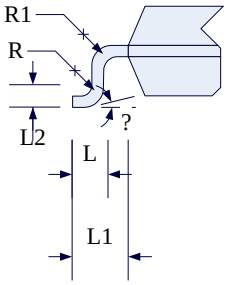


6.5 TSSOP20 封装尺寸图

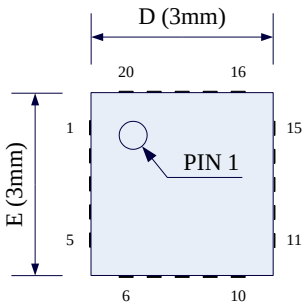


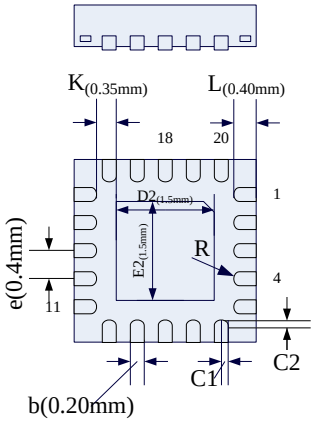
一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	-	-	1.20
A1	0.05	-	0.15
A2	0.90	1.00	1.05
A3	0.34	0.44	0.54
b	0.20	0.24	0.28
D	6.40	6.50	6.60
E	6.20	6.50	6.60
E1	4.30	4.40	4.50
e		0.65BSC	
L	0.45	0.60	0.75
L1		1.00REF	
L2		0.25BSC	
R1	0.09	-	-
R2	0.09	-	-



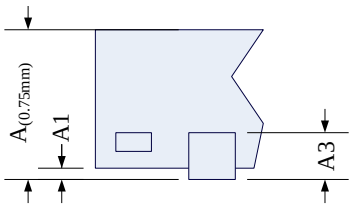


6.6 QFN20 封装尺寸图（3mm*3mm）





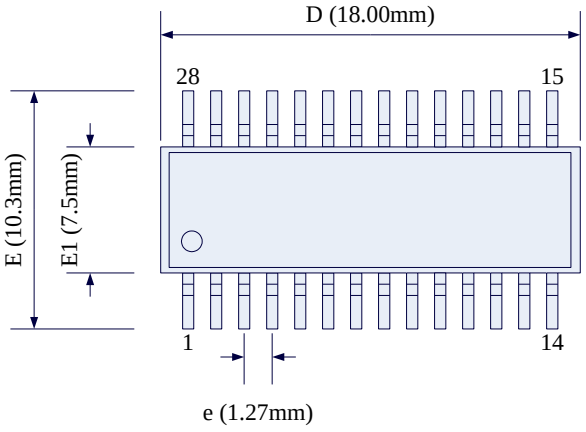
一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	0.70	0.75	0.80
A1	0	0.02	0.05
A2	0.50	0.55	0.60
A3	-	0.20REF	-
b	0.15	0.20	0.25
D	2.90	3.00	3.10
E	2.90	3.00	3.10
D2	1.40	1.50	1.60
E2	1.40	1.50	1.60
e	0.30	0.40	0.50
L	0.35	0.40	0.45
K	0.35REF		
R	0.085	-	-
C1	-	0.07	-
C2	-	0.07	-



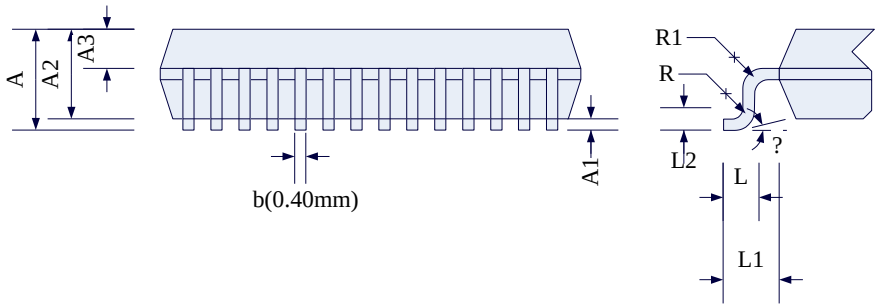
现有 QFN20 封装芯片的背面金属片（衬底），在芯片内部并未接地，在用户的 PCB 板上可以接地，也可以不接地，不会对芯片性能造成影响

特别说明：造 PCB 的封装库时，焊盘可以向外延伸 0.1～0.2mm，但绝对不能向内延伸，必须确保 K 不能小于 0.35mm，否则会造成管脚短路

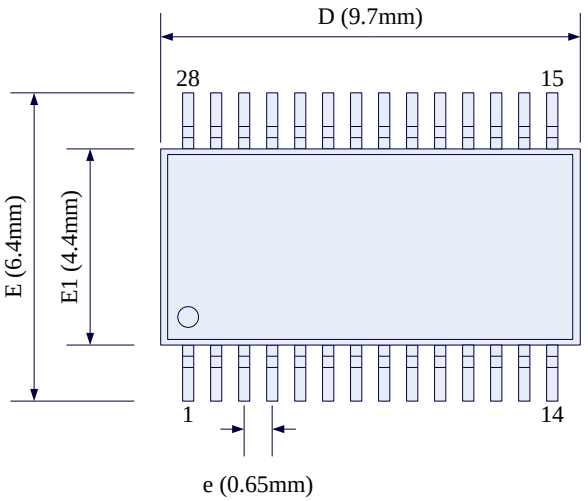
6.7 SOP28 封装尺寸图



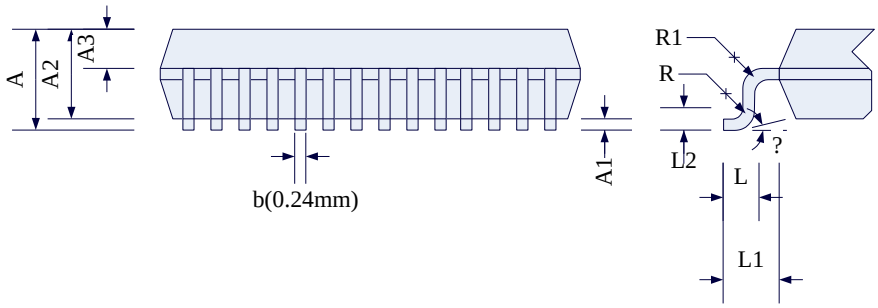
一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	-	-	2.65
A1	0.03	0.08	0.12
A2	2.15	2.20	2.25
A3	0.97	1.02	1.07
b	0.39	-	0.47
D	17.90	18.00	18.10
E	10.10	10.30	10.50
E1	7.40	7.50	7.60
e	1.27BSC		
L	0.70	-	1.00
L1	1.40REF		
L2	0.27BSC		
R1	-	0.30	-
R2	-	0.20	-



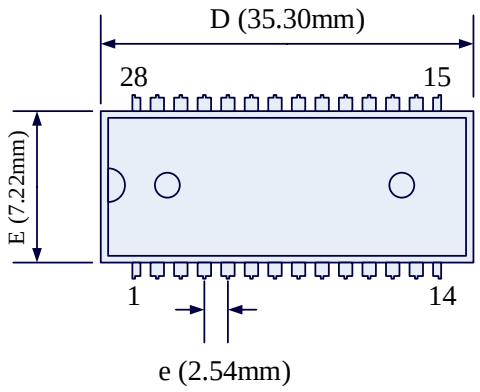
6.8 TSSOP28 封装尺寸图

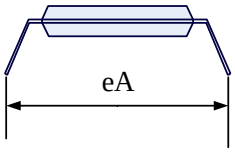


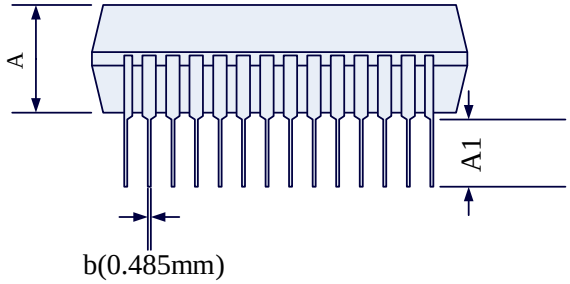
一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	-	-	1.20
A1	0.03	0.08	0.12
A2	0.80	-	1.00
A3	0.39	0.44	0.49
b	0.20	-	0.29
D	9.60	9.70	9.80
E	6.20	6.40	6.60
E1	4.30	4.40	4.50
e		0.65BSC	
L	0.45	0.60	0.75
L1		1.00REF	
L2		0.25BSC	
R1	0.09	-	-
R2	0.09	-	-



6.9 SKDIP28 封装尺寸图

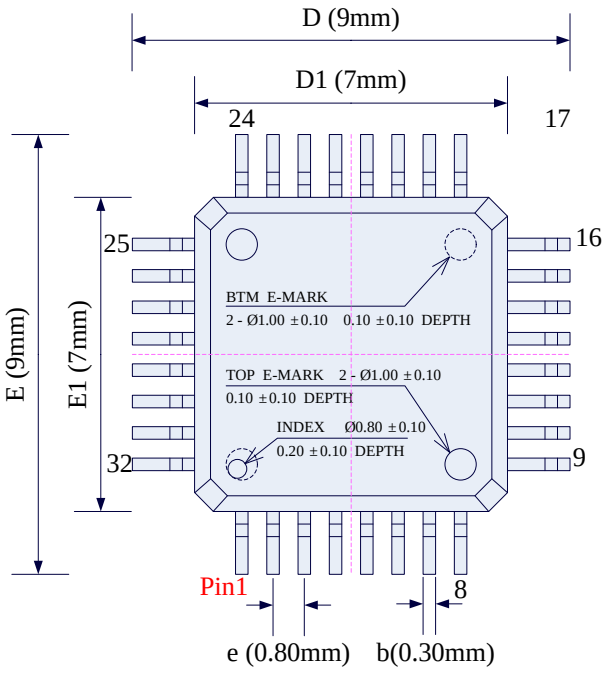




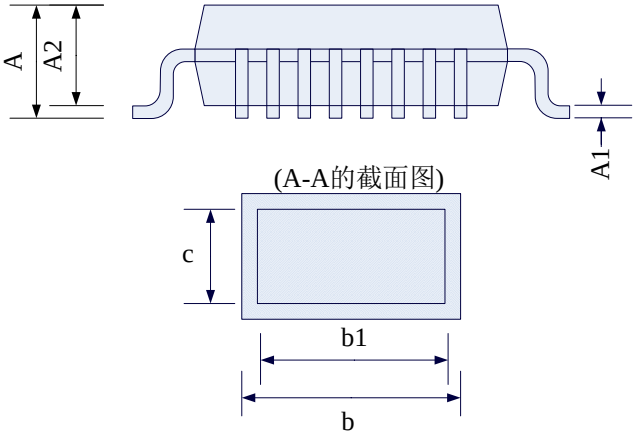


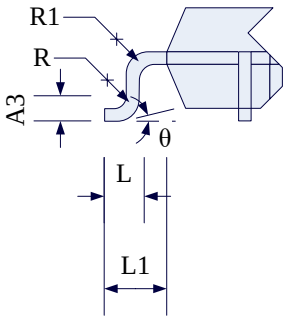
一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	2.90	3.00	3.10
A1	3.20	-	-
b	0.44	-	0.53
D	35.20	35.30	35.40
E	7.12	7.22	7.32
e	2.54BSC		
eA	7.87	-	10.60

6.10 LQFP32 封装尺寸图（9mm*9mm）-- 塑封体正面 3 个顶杆印

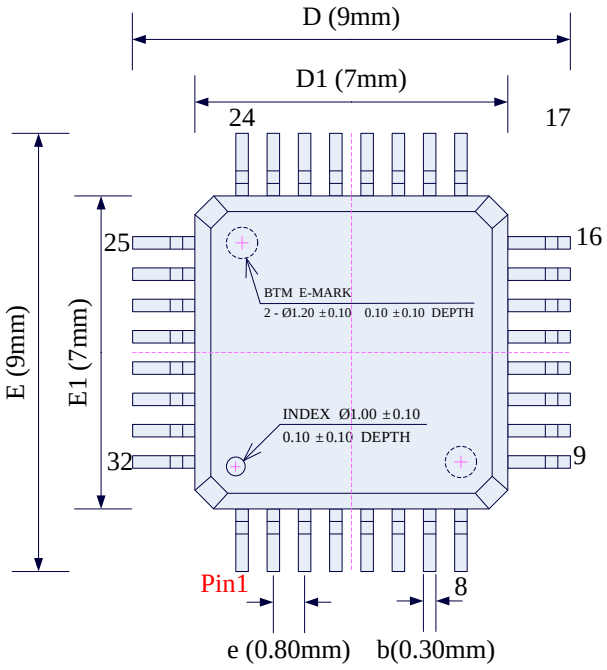


一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	1.45	1.55	1.65
A1	0.05	-	0.15
A2	1.35	1.40	1.45
A3	-	0.254	-
b1	0.30	0.35	0.40
b	0.31	0.37	0.43
c	-	0.127	-
D	8.80	9.00	9.20
D1	6.90	7.00	7.10
E	8.80	9.00	9.20
E1	6.90	7.00	7.10
e	0.70	0.80	0.90
L	0.43	-	0.71
L1	1.00REF		
R	0.1	-	0.25
R1	0.1	-	-
θ	0°	-	10°

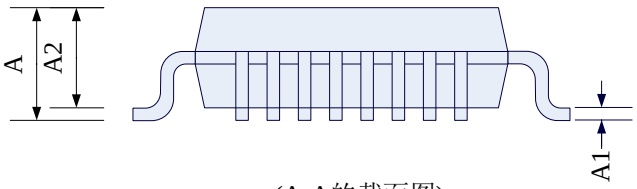


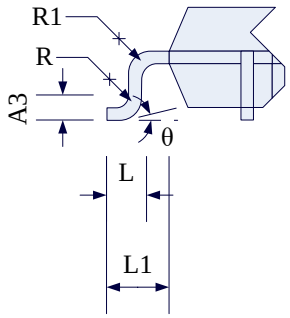


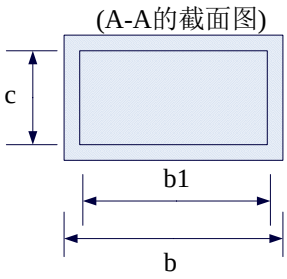
6.11 LQFP32 封装尺寸图（9mm*9mm）-- 塑封体正面 1 个顶杆印



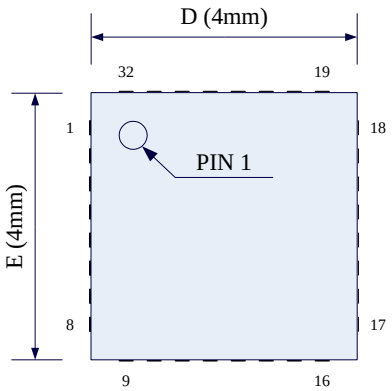
一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	1.45	1.55	1.65
A1	0.05	-	0.15
A2	1.35	1.40	1.45
A3	-	0.254	-
b1	0.30	0.35	0.40
b	0.31	0.37	0.43
c	-	0.127	-
D	8.80	9.00	9.20
D1	6.90	7.00	7.10
E	8.80	9.00	9.20
E1	6.90	7.00	7.10
e	0.70	0.80	0.90
L	0.43	-	0.71
L1		1.00REF	
R	0.1	-	0.25
R1	0.1	-	-
θ	0°	-	10°

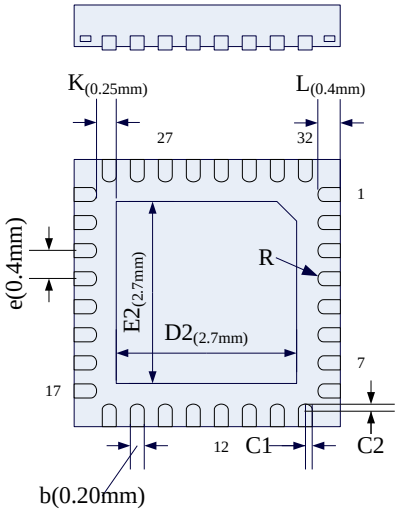


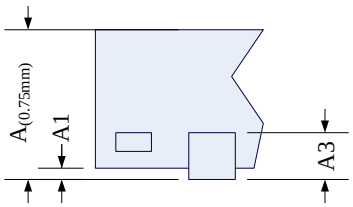




6.12 QFN32 封装尺寸图（4mm*4mm）





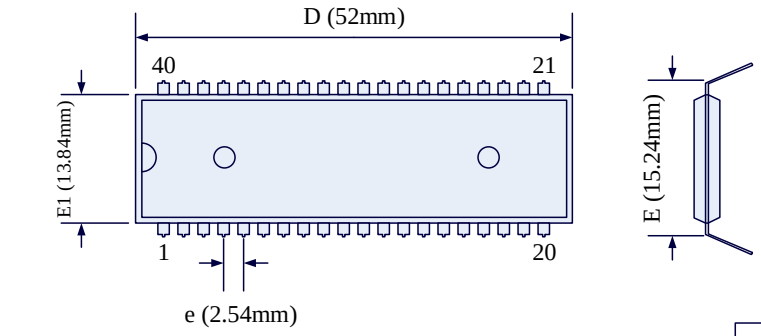


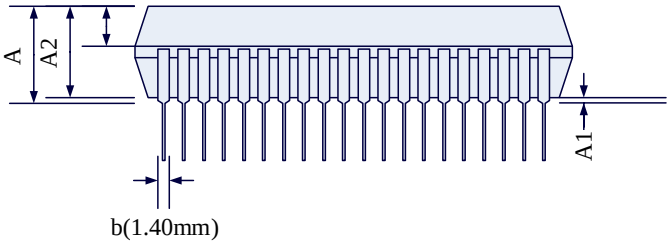
一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	0.70	0.75	0.80
A1	0	0.02	0.05
A2	0.50	0.55	0.60
A3	-	0.20REF	-
b	0.15	0.20	0.25
D	3.90	4.00	4.10
E	3.90	4.00	4.10
D2	2.60	2.70	2.80
E2	2.60	2.70	2.80
e	0.30	0.40	0.50
L	0.35	0.40	0.45
K	0.25REF		
R	0.09	-	-
C1	-	0.16	-
C2	-	0.16	-

现有 QFN32 封装芯片的背面金属片（衬底），在芯片内部并未接地，在用户的 PCB 板上可以接地，也可以不接地，不会对芯片性能造成影响

特别说明：造 PCB 的封装库时，焊盘可以向外延伸 0.1～0.2mm，但绝对不能向内延伸，必须确保 K 不能小于 0.25mm，否则会造成管脚短路

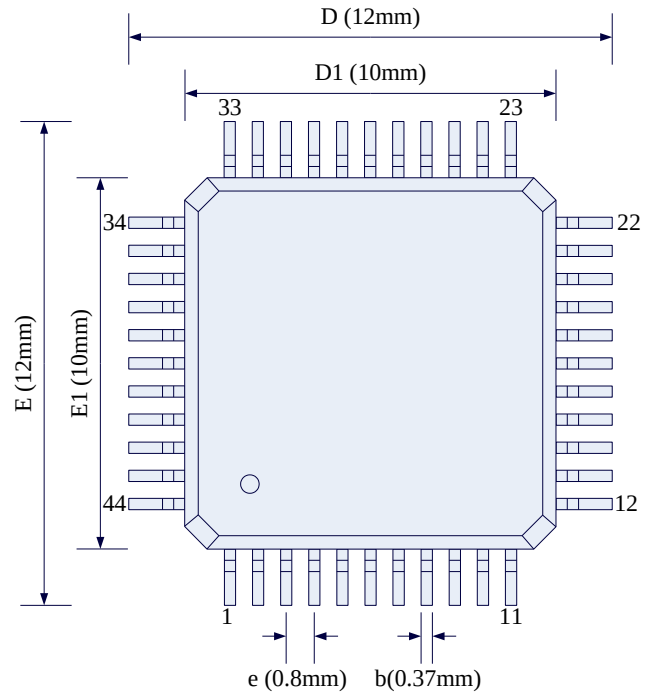
6.13 PDIP40 封装尺寸图



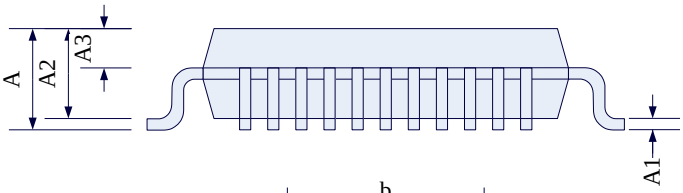


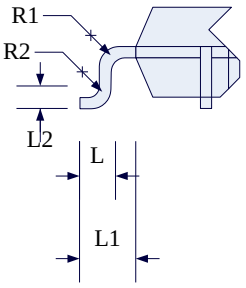
一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	-	-	4.83
A1	0.38	-	0.51
A2	3.81	3.94	4.06
b	1.14	-	1.70
D	51.5	52.00	52.5
E	-	15.24	-
E1	13.72	13.84	13.97
L	3.05	3.30	3.56

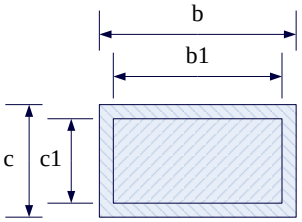
6.14 LQFP44/QFP44 封装尺寸图（12mm*12mm）



一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	-	-	1.60
A1	0.05	-	0.15
A2	1.35	1.40	1.45
A3	0.59	0.64	0.69
b	0.33	0.37	0.42
b1	0.32	0.35	0.38
c	0.13	-	0.18
c1	0.12	0.13	0.14
D	11.95	12.00	12.05
D1	9.90	10.00	10.10
E	11.95	12.00	12.05
E1	9.90	10.00	10.10
e	0.70	0.80	0.90
L	0.45	0.60	0.75
L1	1.00REF		
L2	0.25BSC		
R1	0.08	-	-
R2	0.08	-	0.20
S	0.20	-	-

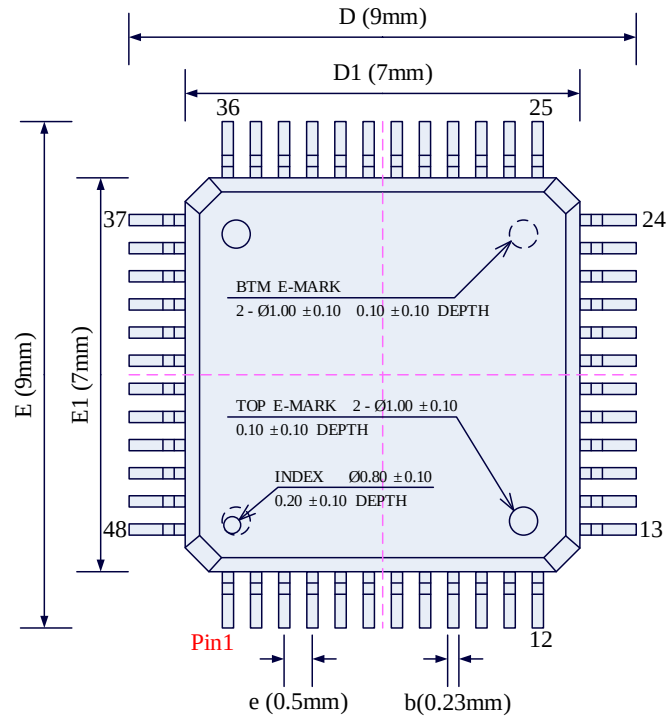




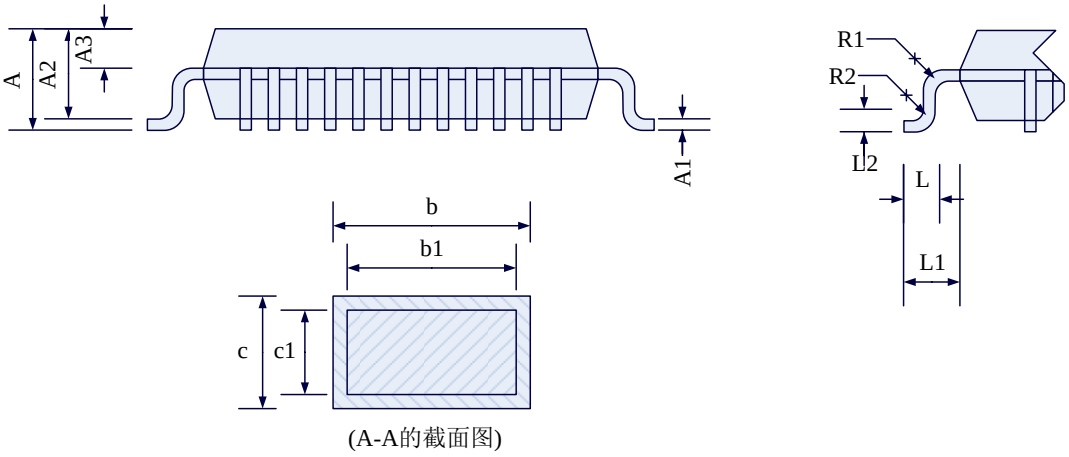


(A-A的截面图)

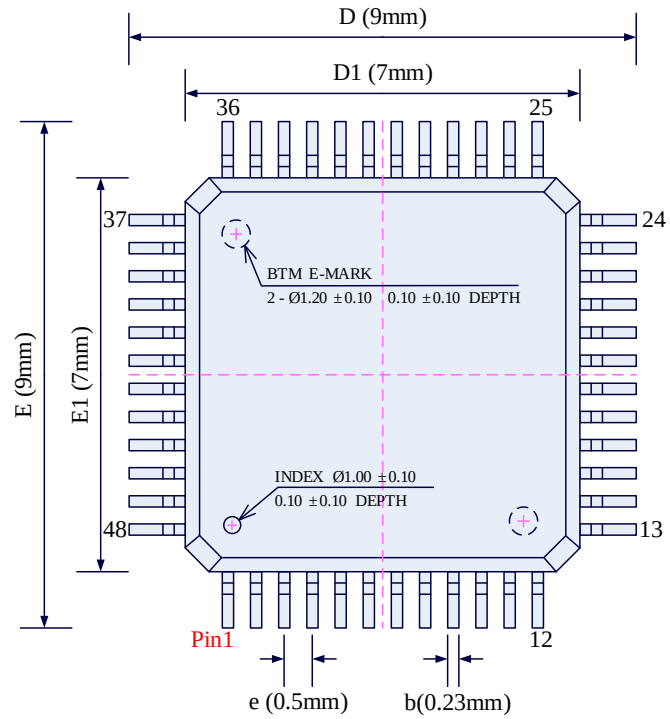
6.15 LQFP48 封装尺寸图（9mm*9mm）-- 塑封体正面 3 个顶杆印



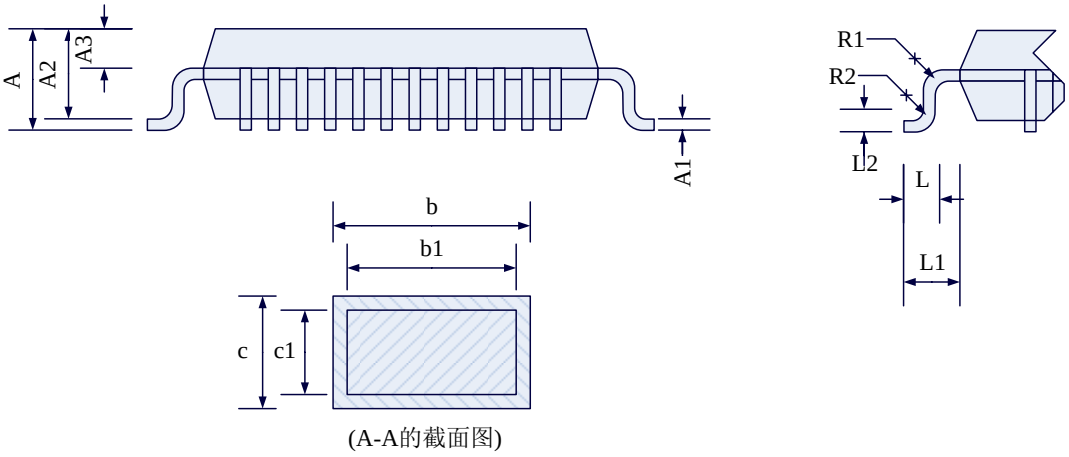
一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	-	-	1.60
A1	0.05	-	0.15
A2	1.35	1.40	1.45
A3	0.59	0.64	0.69
b	0.18	0.23	0.27
b1	0.17	0.20	0.23
c	0.13	-	0.18
c1	0.12	0.127	0.134
D	8.80	9.00	9.20
D1	6.90	7.00	7.10
E	8.80	9.00	9.20
E1	6.90	7.00	7.10
e	0.45	0.50	0.55
L	0.45	0.60	0.75
L1	1.00REF		
L2	0.25BSC		
R1	0.08	-	-
R2	0.08	-	0.20
S	0.20	-	-



6.16 LQFP48 封装尺寸图（9mm*9mm）-- 塑封体正面 1 个顶杆印

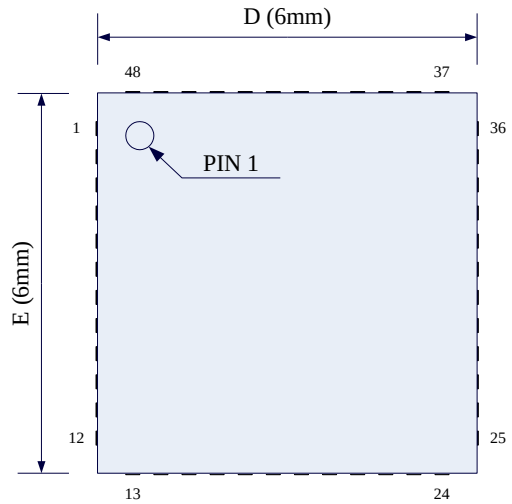


一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	-	-	1.60
A1	0.05	-	0.15
A2	1.35	1.40	1.45
A3	0.59	0.64	0.69
b	0.18	0.23	0.27
b1	0.17	0.20	0.23
c	0.13	-	0.18
c1	0.12	0.127	0.134
D	8.80	9.00	9.20
D1	6.90	7.00	7.10
E	8.80	9.00	9.20
E1	6.90	7.00	7.10
e	0.45	0.50	0.55
L	0.45	0.60	0.75
L1	1.00REF		
L2	0.25BSC		
R1	0.08	-	-
R2	0.08	-	0.20
S	0.20	-	-

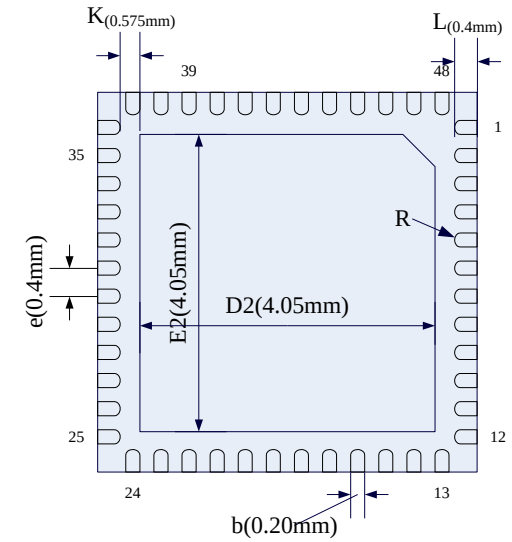


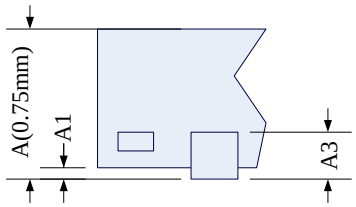
(A-A的截面图)

6.17 QFN48 封装尺寸图（6mm*6mm）



一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	0.70	0.75	0.80
A1	0.00	0.02	0.05
A3		0.20REF	
b	0.15	0.20	0.25
D	5.90	6.00	6.10
E	5.90	6.00	6.10
D2	3.95	4.05	4.15
E2	3.95	4.05	4.15
e	0.35	0.40	0.45
L	0.35	0.40	0.45
K		0.575	
R	0.09	-	-

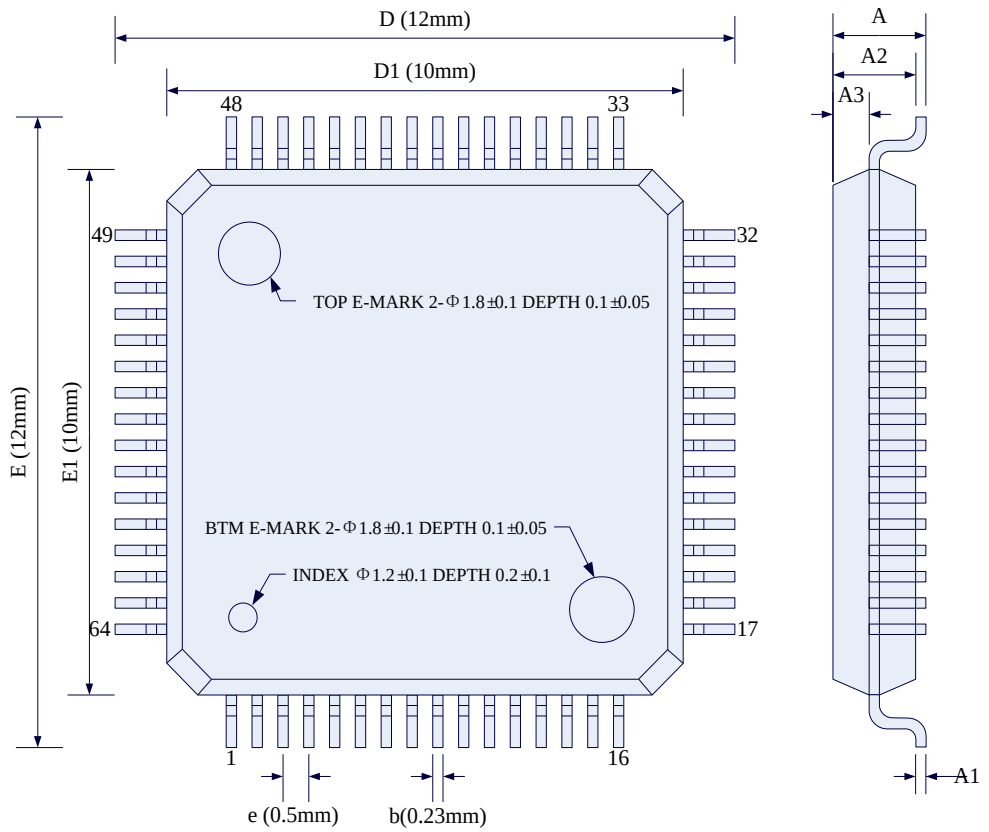




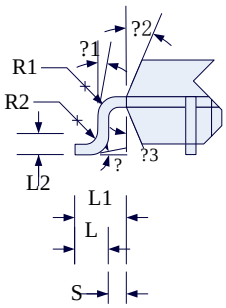
现 Ai8051U 的 QFN48 封装芯片的背面金属片（衬底），接了内部 ADC 的模拟地线，在用户的 PCB 板上可以接模拟地，也可以不接地，不会对芯片性能造成影响。

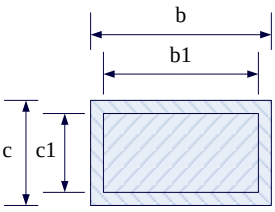
特别说明：造 PCB 的封装库时，焊盘可以向外延伸 0.1～0.2mm，但绝对不能向内延伸，必须确保 K 不能小于 0.2mm，否则会造成管脚短路

6.18 LQFP64 封装尺寸图（12mm*12mm）



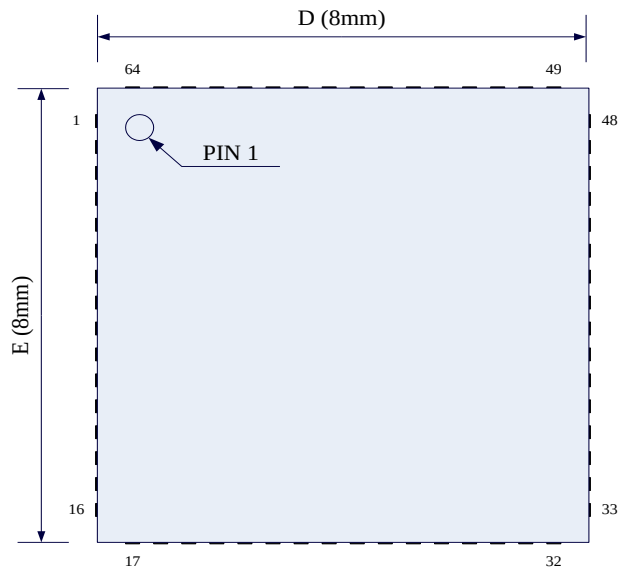
一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	-	-	1.60
A1	0.05	-	0.15
A2	1.35	1.40	1.45
A3	0.59	0.64	0.69
b	0.18	0.23	0.27
b1	0.17	0.20	0.23
c	0.13	-	0.18
c1	0.12	0.127	0.134
D	11.80	12.00	12.20
D1	9.90	10.00	10.10
E	11.80	12.00	12.20
E1	9.90	10.00	10.10
e	0.50BSC		
L	0.45	0.60	0.75
L1	1.00REF		
L2	0.25BSC		
R1	0.08	-	-
R2	0.08	-	0.20
S	0.20	-	-
?	0°	3.5°	7°
?1	0°	-	-
?2	11°	12°	13°
?3	11°	12°	13°



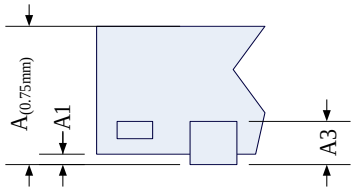


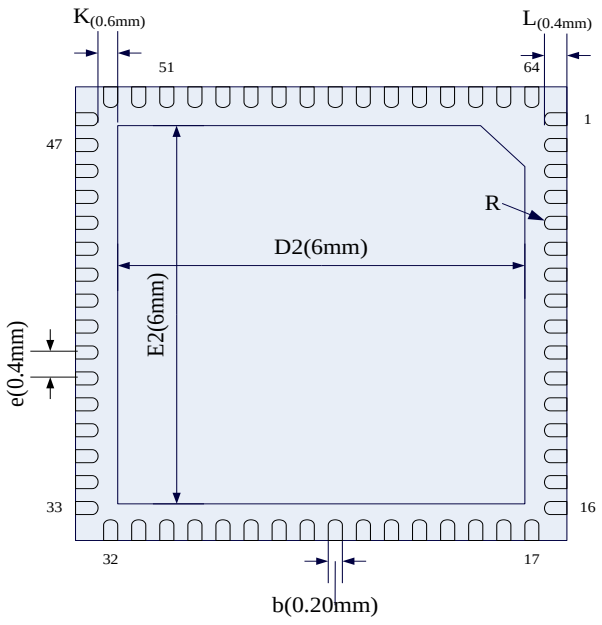
(A-A的截面图)

6.19 QFN64 封装尺寸图（8mm*8mm）



一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	0.70	0.75	0.80
A1	0.00	0.02	0.05
A2	0.50	0.55	0.60
A3		0.20REF	
b	0.15	0.20	0.25
D	7.90	8.00	8.10
E	7.90	8.00	8.10
D2	5.90	6.00	6.10
E2	5.90	6.00	6.10
e	0.30	0.40	0.50
L	0.30	0.40	0.50
K		0.60	
R	0.09	-	-

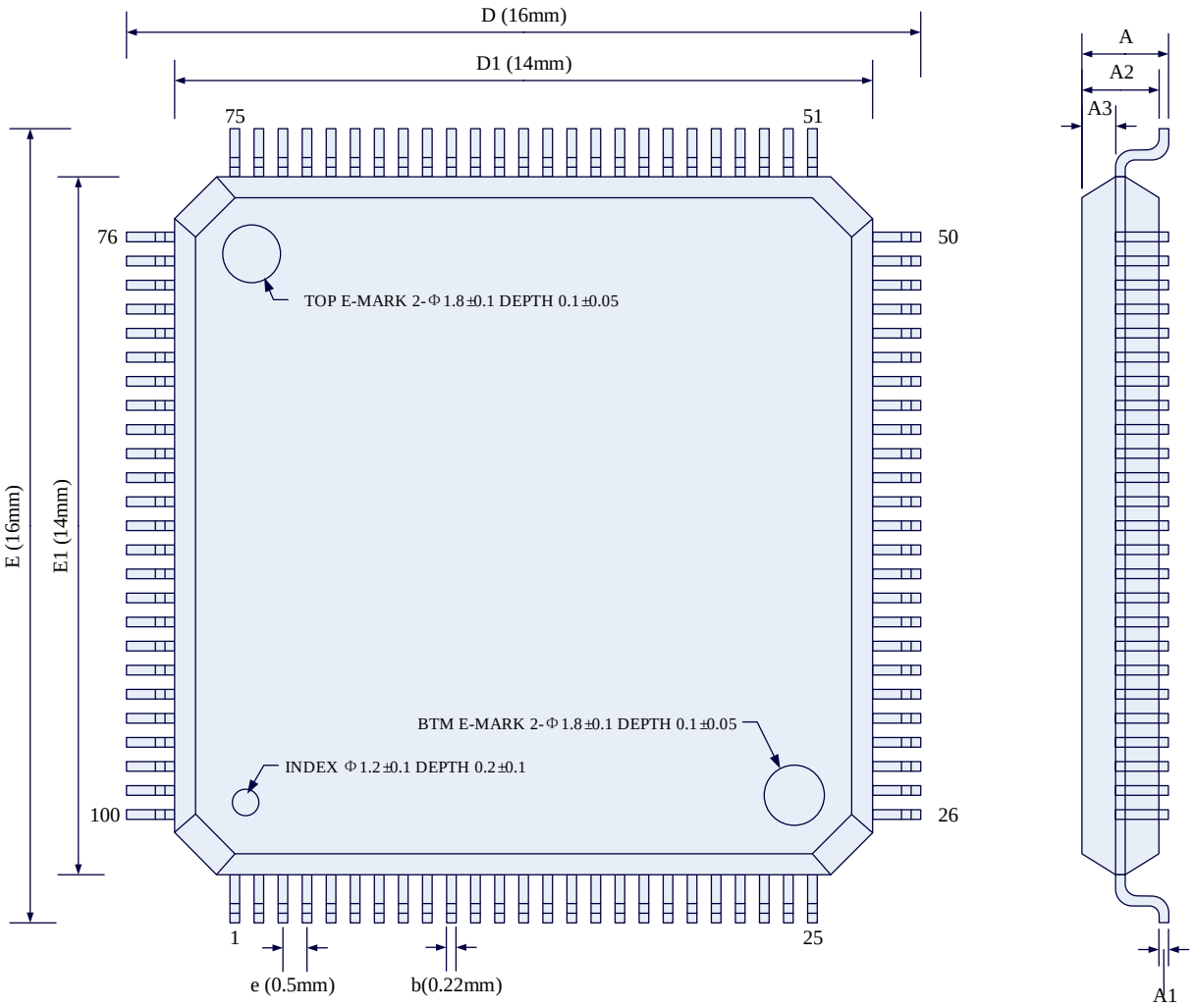


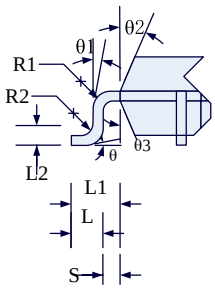


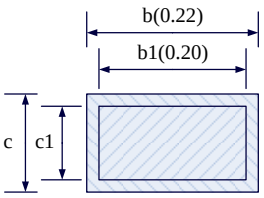
现有 QFN64 封装芯片的背面金属片（衬底），在芯片内部并未接地，在用户的 PCB 板上可以接地，也可以不接地，不会对芯片性能造成影响

特别说明：造 PCB 的封装库时，焊盘可以向外延伸 0.1～0.2mm，但绝对不能向内延伸，必须确保 K 不能小于 0.4mm，否则会造成管脚短路

6.20 LQFP100 封装尺寸图（16mm*16mm）



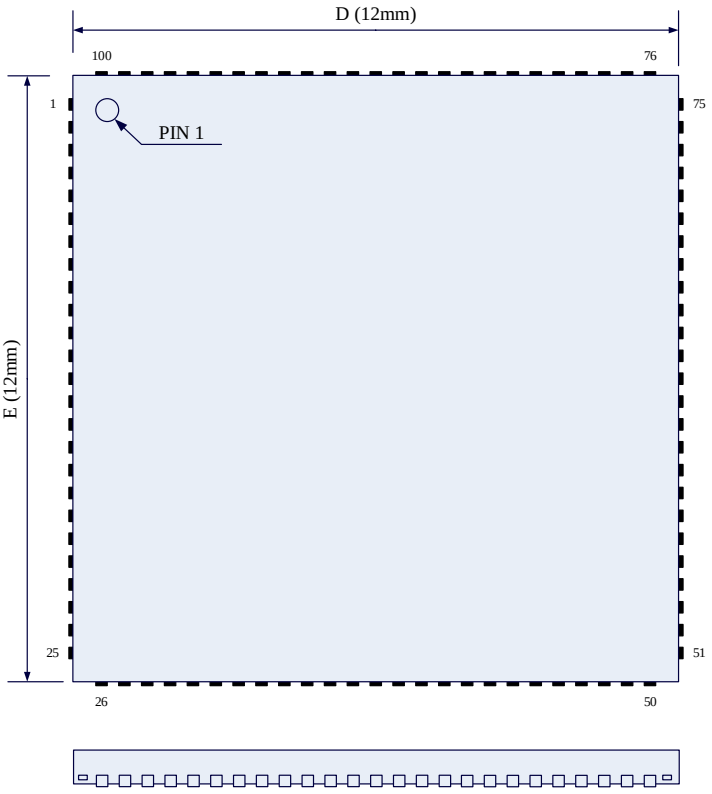




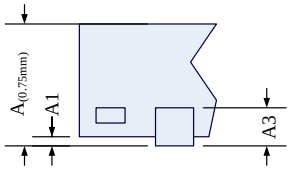
(A-A的截面图)

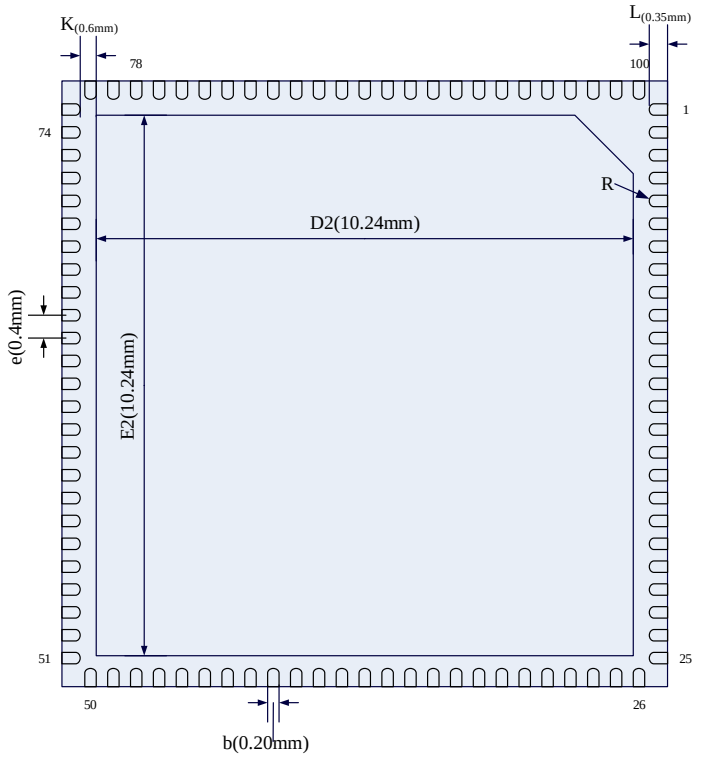
一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	-	-	1.60
A1	0.05	-	0.15
A2	1.35	1.40	1.45
A3	0.59	0.64	0.69
b	0.18	0.22	0.26
b1	0.17	0.20	0.23
c	0.13	-	0.17
c1	0.12	0.13	0.14
D	15.80	16.00	16.20
D1	13.90	14.00	14.10
E	15.80	16.00	16.20
E1	13.90	14.00	14.10
e	0.50BSC		
L	0.45	0.60	0.75
L1	1.00REF		
L2	0.25BSC		
R1	0.08	-	-
R2	0.08	-	0.20
S	0.20	-	-
θ	0°	3.5°	7°
θ1	0°	-	-
θ2	11°	12°	13°
θ3	11°	12°	13°

6.21 QFN100 封装尺寸图（12mm*12mm）

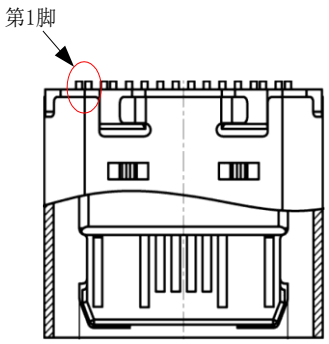


一般尺寸			
测量单位：毫米/mm			
SYMBOL	MIN	TYP	MAX
A	0.70	0.75	0.80
A1	0.00	0.02	0.05
A2	0.50	0.55	0.60
A3		0.20REF	
b	0.15	0.20	0.25
D	11.90	12.00	12.10
E	11.90	12.00	12.10
D2	10.14	10.24	10.34
E2	10.14	10.24	10.34
e	0.30	0.40	0.50
L	0.30	0.35	0.40
K		0.60	
R	0.09	-	-

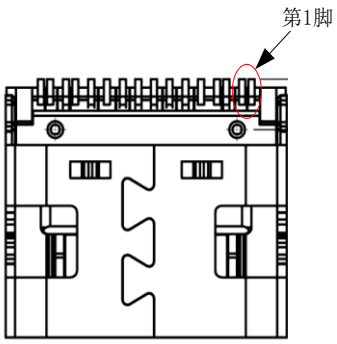




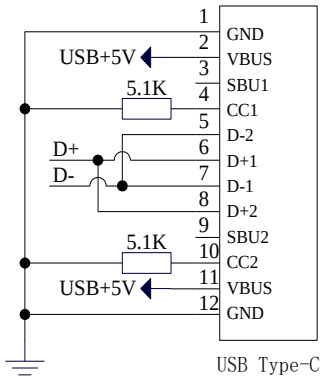
6.22 USB-TypeC 封装尺寸图，原理图



USB Type-C
实物图(正面)



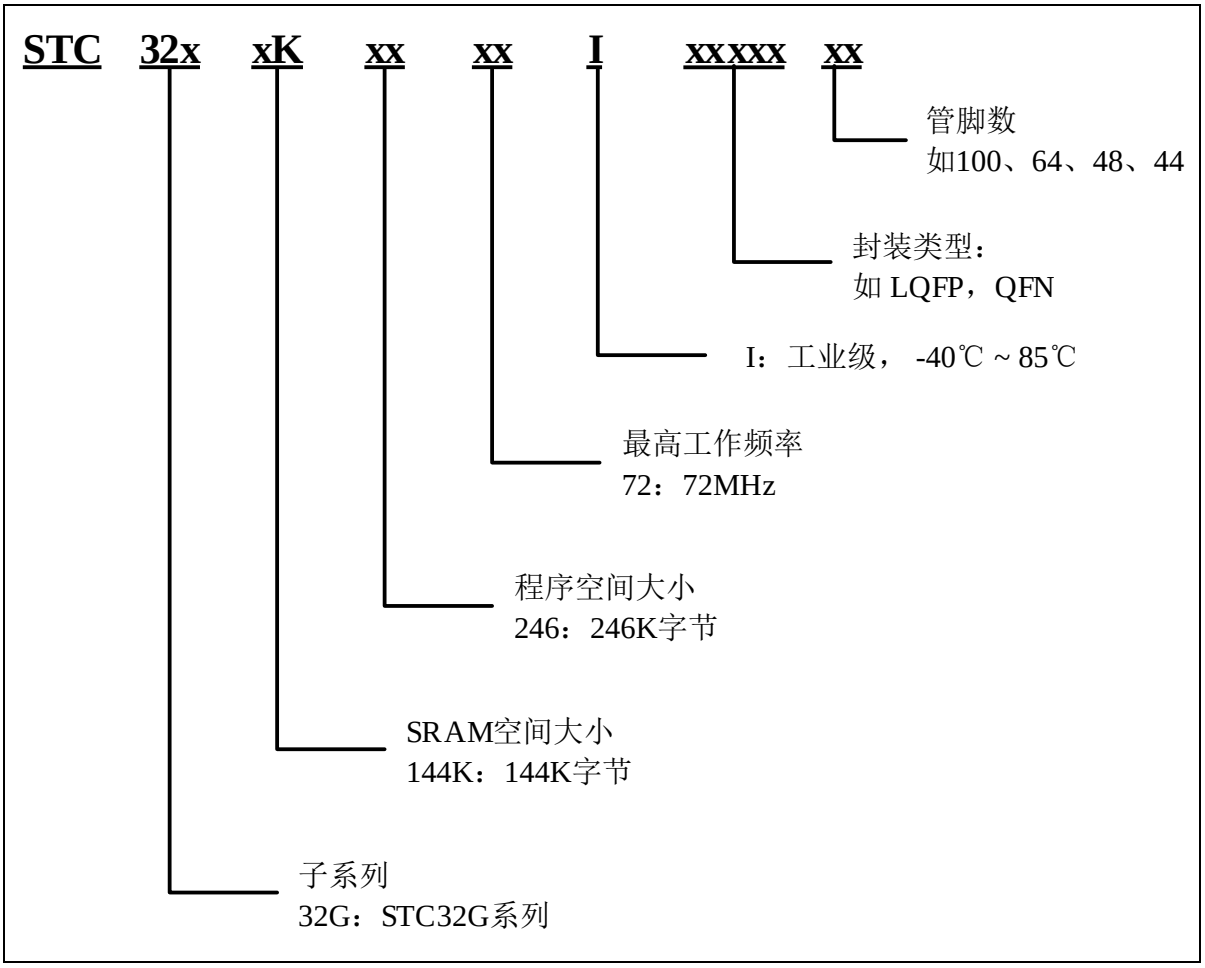
USB Type-C
实物图(反面)



USB Type-C
电路原理图

	A1	B12	A4	B9	A8	A5	B7	A6	A7	B6	B8	B5	B4	A9	B1	A12
	GND	GND	VBUS	VBUS	SBU1	CC1	D-2	D+1	D-1	D+2	SBU2	CC2	VBUS	VBUS	GND	GND
Type-C 16PIN	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Type-C 12PIN	1		2		3	4	5	6	7	8	9	10	11		12	

6.23 单片机命名规则


STC 32x xK xx xx I xxxxx xx
管脚数
如100、64、48、44
封装类型：
如 LQFP，QFN
I：工业级， -40℃ ~ 85℃
最高工作频率
72：72MHz
程序空间大小
246：246K字节
SRAM空间大小
144K：144K字节
子系列
32G：STC32G系列

7 i-Cache 应用，如何让高速 CPU 和慢速 Flash 协同

STC32G144K246 系列单片机采用了先进的设计制程，使系统运行的工作频率可在 64MHz ~ 120MHz 之间，内部 Flash 程序存储器的最高工作频率大致在 33MHz ~ 40MHz 左右，CPU 的执行速度相对较快，Flash 程序存储器的速度相对较慢，这就导致了所谓的“存储墙”限制运行速度的问题。

为了解决这个问题，STC32G144K246 系列单片机内部设计了 i-Cache（程序指令高速缓存）功能。i-Cache 功能是将慢速 Flash 程序存储器中的程序代码加载到 CPU 内部的 i-Cache 高速缓存中，CPU 执行指令时直接从 i-Cache 中进行取指，当出现 CPU 需要运行的指令不在 i-Cache 中时，就会触发 i-Cache 模块重新从程序 Flash 的目标地址中预装载 CPU 所需要的指令，指令缓存结束后，CPU 继续从 i-Cache 中取指运行程序。由于 i-Cache 高速缓存的出现，CPU 的速度再也不受低速 Flash 程序存储器的限制。

STC32G144K246 系列单片机还设计了预取指模块，来提高 CPU / i-Cache 的运行效率。

解决了 CPU 运行速度问题后，高速 i-Cache 与慢速 Flash 程序存储器之间的速度匹配问题是通过 Flash 读取等待控制寄存器 WTST 来解决的。WTST 寄存器的功能是设置从 Flash 中读取指令所需要等待的系统时钟数。由于 Flash 的最高工作频率为 33MHz ~ 40MHz 左右，所以当系统工作频率低于 33MHz 时，WTST 可设置为 0；当系统工作频率为 33MHz ~ 64MHz 之间时，WTST 可设置为 1 ~ 2；当系统工作频率为 64MHz ~ 90MHz 之间时，WTST 可设置为 2 ~ 3；当系统工作频率为 90MHz ~ 120MHz 之间时，WTST 可设置为 3 ~ 4。

【特别注意】：当使能了 i-Cache 功能后，WTST 必须设置为大于等于 1 的值，不能设置为 0，即使系统工作频率低于 33MHz。由于 i-Cache 功能在芯片出厂时就已经默认使能，无法通过软件或硬件设置关闭，所以 WTST 上电后的初始值为 1，且用户也不能将 WTST 设置为 0（软件写 0 无效，硬件加了防错保护）。

7.1 程序读取等待控制寄存器（WTST）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
WTST	E9H	WTST[7:0]							

WTST[7:0]：高速 i-Cache 读取慢速 Flash 程序存储器的等待时时钟设置

WTST[7:0]	等待时钟数	设置建议值
0	0 个时钟	系统时钟 <= 32MHz
1	1 个时钟	32MHz < 系统时钟 <= 64MHz
2	2 个时钟	64MHz < 系统时钟 <= 90MHz
3	3 个时钟	90MHz < 系统时钟 <= 120MHz
4	4 个时钟	系统时钟 > 120MHz
…	…	
7	7 个时钟	
8	保留	
…	保留	
255	保留	

如 CPU 需要执行的指令已被缓存在 i-Cache 中，则此指令的执行时钟数等于指令表中所列的时钟数；当所需的指令不在 i-Cache 中时，则 CPU 需要等待 i-Cache 缓存从程序存取器中预取后才能继续运行。

7.2 速度配置及程序初始化标准参考例程

```
#include "STC32G144K246.H"
#include "INTRINS.H"

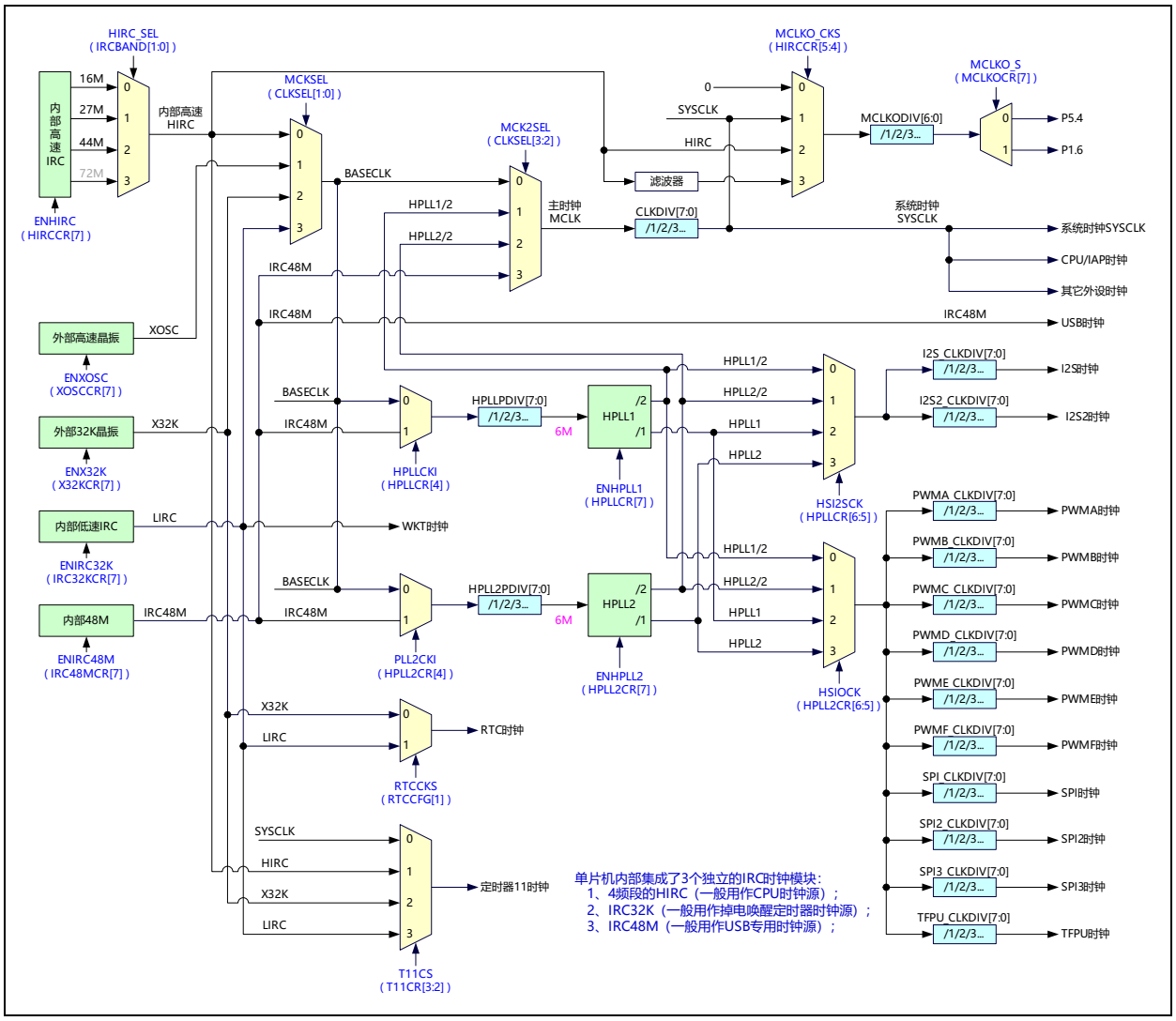
void main()
{
    P_SW2 = 0x80;                   //允许访问扩展的特殊功能寄存器 XFR
//      WTST = 0x00;                //设置读取程序代码等待时间（SYSCLK <= 32MHz）
    WTST = 0x01;                    //设置读取程序代码等待时间（32MHz < SYSCLK <= 64MHz）
//      WTST = 0x02;                //设置读取程序代码等待时间（64MHz < SYSCLK <= 90MHz）
//      WTST = 0x03;                //设置读取程序代码等待时间（90MHz < SYSCLK <= 120MHz）
//      WTST = 0x04;                //设置读取程序代码等待时间（SYSCLK > 120MHz）
    CKCON = 0x00;                   //设置访问片内的 xdata 速度
                                //设置为 0 表示用最快速度访问，不增加额外的等待时间
    P0M1 = 0xff; P0M0 = 0xff; P0PU = 0xff;    // I/O 口初始化成开漏模式并打开内部上拉
    P1M1 = 0xff; P1M0 = 0xff; P1PU = 0xff;
    P2M1 = 0xff; P2M0 = 0xff; P2PU = 0xff;
    P3M1 = 0xff; P3M0 = 0xff; P3PU = 0xff;
    P4M1 = 0xff; P4M0 = 0xff; P4PU = 0xff;
    P5M1 = 0xff; P5M0 = 0xff; P5PU = 0xff;
    P6M1 = 0xff; P6M0 = 0xff; P6PU = 0xff;
    P7M1 = 0xff; P7M0 = 0xff; P7PU = 0xff;
    P8M1 = 0xff; P8M0 = 0xff; P8PU = 0xff;
    P9M1 = 0xff; P9M0 = 0xff; P9PU = 0xff;
    PAM1 = 0xff; PAM0 = 0xff; PAPU = 0xff;
    PBM1 = 0xff; PBM0 = 0xff; PBPU = 0xff;
    PCM1 = 0xff; PCM0 = 0xff; PCPU = 0xff;

    while (1);
}
```

8 时钟管理，芯片上电工作过程，外部晶振抗干扰

8.1 系统时钟控制

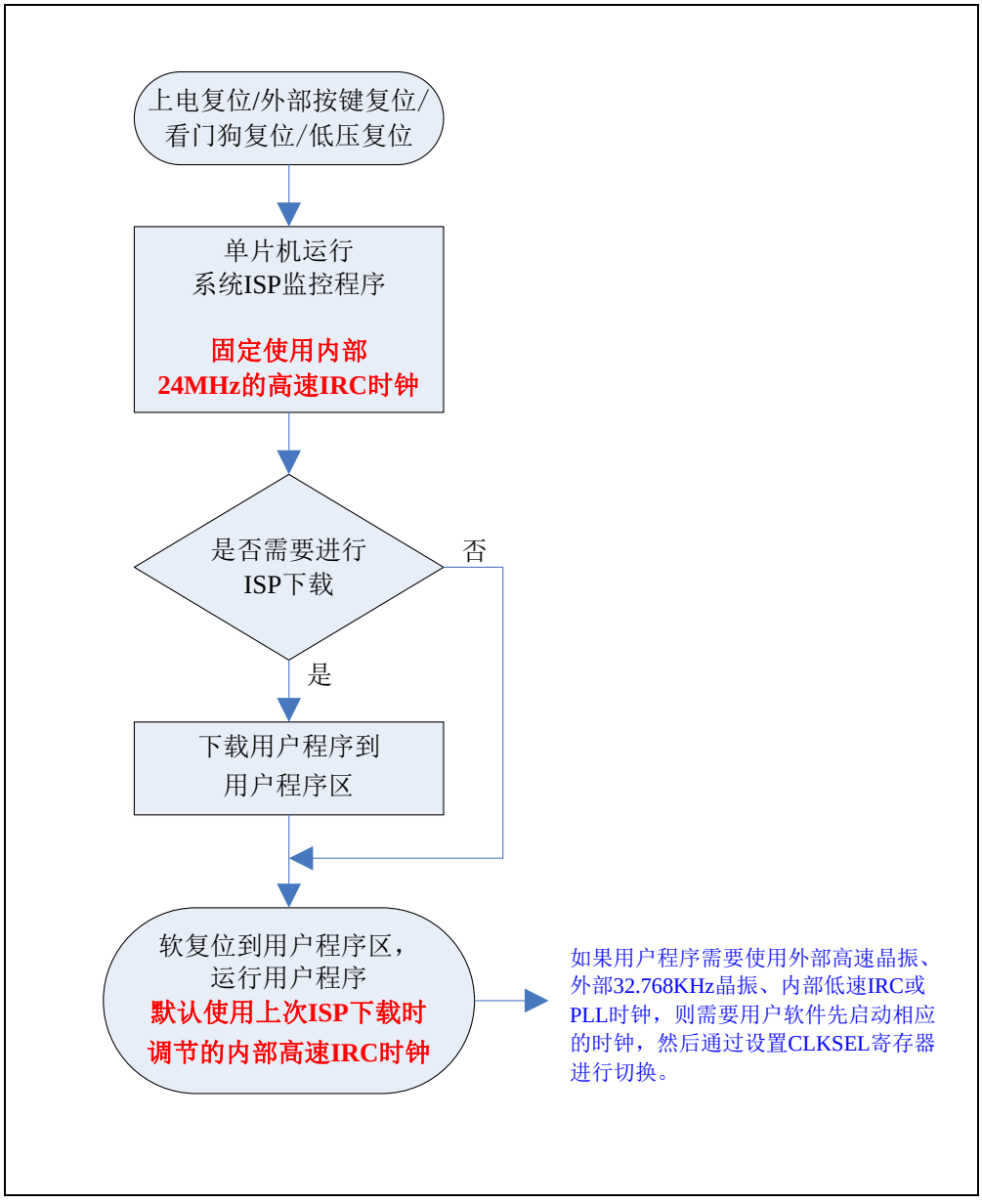
系统时钟控制器为单片机的 CPU 和所有外设系统提供时钟源，系统时钟有 6 个时钟源可供选择：内部高精度高速 IRC、内部低速 IRC（误差较大）、外部高速晶振、外部 32768 晶振、内部高速 HPLL1 输出时钟、内部高速 HPLL2 输出时钟。用户可通过程序分别使能和关闭各个时钟源，以及内部提供时钟分频以达到降低功耗的目的。



8.2 芯片上电工作过程

上电复位/复位脚复位/看门狗复位/低压检测复位时，芯片默认从 ISP 系统程序开始执行代码，此时固定使用内部 24MHz 的高速 IRC 时钟，当需要下载用户程序且下载完成后复位到用户程序区或者不需要下载直接复位到用户程序区时，默认会使用上次用户下载时所调节的高速 IRC 时钟，如果用户程序需要使用外部高速晶振、外部 32.768KHz 晶振、内部低速 IRC 或者高速 PLL 时钟，则需要用户软件先启动相应的时钟，然后通过设置 CLKSEL 寄存器进行切换。

启动流程如下：



8.3 时钟配置寄存器，外设，高频 PLL 时钟及分频，外部晶振抗干扰

相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
IRCBAND	IRC 频段选择检测	9DH	USBCKS	USBCKS2	-	-	-	-	SEL[1:0]		10xx,xxnn
IRTRIM	IRC 频率调整寄存器	9FH	IRTRIM[7:0]								nnnn,nnnn

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CLKSEL	时钟选择寄存器	7EFE00H	-	-	-	-	MCK2SEL[1:0]		MCKSEL[1:0]		xxxx,0000
CLKDIV	时钟分频寄存器	7EFE01H	CLKDIV[7:0]								nnnn,nnn
HIRCCR	内部高速振荡器控制寄存器	7EFE02H	ENHIRC	-	MCLKO_CKS[1:0]		-	-	-	HIRCST	1x00,xxx0
XOSCCR	外部晶振控制寄存器	7EFE03H	ENXOSC	XITYPE	GAIN	-	XCFILTER[1:0]		-	XOSCST	000x,00x0
IRC32KCR	内部低速振荡器控制寄存器	7EFE04H	ENIRC32K	-	-	-	-	-	-	IRC32KST	0xxx,xxx0
MCLKOCR	主时钟输出控制寄存器	7EFE05H	MCLKO_S	MCLKODIV[6:0]							0000,0000
IRCDB	内部高速振荡器去抖控制	7EFE06H	IRCDB[7:0]								1000,0000
IRC48MCR	内部 48M 振荡器控制寄存器	7EFE07H	ENIRC48M	-	-	-	-	-	-	IRC48MST	1xxx,xxx0
X32KCR	外部 32K 晶振控制寄存器	7EFE08H	ENX32K	GAIN32K	-	-	-	-	-	X32KST	00xx,xxx0
IRC48MATRIM	内部 48M 振荡器校准值	7EFE09H	-	-	-	TRIM[12:8]					xxxn,nnnn
IRC48MBTRIM	内部 48M 振荡器校准值	7EFE0AH	TRIM[7:0]								nnnn,nnnn
HPLLCR	高速 PLL 控制寄存器	7EFE0CH	ENHPLL	HSI2SCK[1:0]		HPLLCKS	HPLLDIV[3:0]				0000,0000
HPLLPDIV	高速 PLL 预分频寄存器	7EFE0DH	HPLL_PREDIV[7:0]								0000,0000
HPLL2CR	高速 PLL2 控制寄存器	7EFE0EH	ENHPLL2	HSIOCK[1:0]		HPLL2CKS	HPLL2DIV[3:0]				0000,0000
HPLL2PDIV	高速 PLL2 预分频寄存器	7EFE0FH	HPLL2_PREDIV[7:0]								0000,0000
SPI_CLKDIV	SPI 时钟预分配寄存器	7EFE90H	SPI_CLKDIV[7:0]								0000,0000
SPI2_CLKDIV	SPI2 时钟预分配寄存器	7EF808H	SPI2_CLKDIV[7:0]								0000,0000
SPI3_CLKDIV	SPI3 时钟预分配寄存器	7EF828H	SPI3_CLKDIV[7:0]								0000,0000
PWMA_CLKDIV	PWMA 时钟预分配寄存器	7EFE91H	PWMA_CLKDIV[7:0]								0000,0000
PWMB_CLKDIV	PWMB 时钟预分配寄存器	7EFE92H	PWMB_CLKDIV[7:0]								0000,0000
PWMC_CLKDIV	PWMC 时钟预分配寄存器	7EFE96H	PWMC_CLKDIV[7:0]								0000,0000
PWMD_CLKDIV	PWMD 时钟预分配寄存器	7EFE97H	PWMD_CLKDIV[7:0]								0000,0000
PWME_CLKDIV	PWME 时钟预分配寄存器	7EF68BH	PWME_CLKDIV[7:0]								0000,0000
PWMF_CLKDIV	PWMF 时钟预分配寄存器	7EF68FH	PWMF_CLKDIV[7:0]								0000,0000
TFPU_CLKDIV	TFPU 时钟预分配寄存器	7EFE93H	TFPU_CLKDIV[7:0]								0000,0000
I2S_CLKDIV	I2S 时钟预分配寄存器	7EFE94H	I2S_CLKDIV[7:0]								0000,0000
I2S2_CLKDIV	I2S2 时钟预分配寄存器	7EFE95H	I2S2_CLKDIV[7:0]								0000,0000

8.3.1 系统时钟选择寄存器（CLKSEL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CLKSEL	7EFE00H	-	-	-	-	MCK2SEL[1:0]		MCKSEL[1:0]	

MCK2SEL[1:0]：主时钟源选择

MCK2SEL[1:0]	主时钟源
00	MCKSEL 选择的时钟源
01	内部 HPLL1 输出/2
10	内部 HPLL2 输出/2
11	内部 48MHz 高速 IRC

MCKSEL[1:0]：主时钟源选择

MCKSEL[1:0]	主时钟源
00	内部高精度 IRC
01	外部高速晶振
10	外部 32KHz 晶振
11	内部 32KHz 低速 IRC

8.3.2 时钟分频寄存器（CLKDIV）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CLKDIV	7EFE01H								

CLKDIV：主时钟分频系数。系统时钟 SYSCLK 是对主时钟 MCLK 进行分频后的时钟信号。

CLKDIV	系统时钟频率
0	MCLK/1
1	MCLK/1
2	MCLK/2
3	MCLK/3
…	…
255	MCLK/255

8.3.3 内部高速高精度 IRC 控制寄存器（HIRCCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HIRCCR	7EFE02H	ENHIRC	-	MCLKO_CKS[1:0]		-	-	-	HIRCST

ENHIRC：内部高速高精度 IRC 使能位

0：关闭内部高精度 IRC

1：使能内部高精度 IRC

HIRCST：内部高速高精度 IRC 频率稳定标志位。（只读位）

当内部的 IRC 从停振状态开始使能后，必须经过一段时间，振荡器的频率才会稳定，当振荡器频率稳定后，时钟控制器会自动将 HIRCST 标志位置 1。所以当用户程序需要将时钟切换到使用内部 IRC 时，首先必须设置 ENHIRC=1 使能振荡器，然后一直查询振荡器稳定标志位 HIRCST，直到标志位变为 1 时，才可进行时钟源切换。

MCLKO_CKS[1:0]：时钟输出的时钟源选择（此位只写）

00：关闭主时钟输出

01：输出系统时钟

10：输出内部高速 HIRC 时钟

11：输出滤波后的内部高速 HIRC 时钟

8.3.4 外部振荡器控制寄存器（XOSCCR），外部晶振抗干扰

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
XOSCCR	7EFE03H	ENXOSC	XITYPE	GAIN		XCFILTER[1:0]			XOSCST

ENXOSC：外部晶体振荡器使能位

0：关闭外部晶体振荡器

1：使能外部晶体振荡器

XITYPE：外部时钟源类型

0：外部时钟源是外部时钟信号（或有源晶振）。信号源只需连接单片机的 XTALI（P1.7），此时 P1.6 口为高阻输入模式，P1.6 可当作输入口使用。

1：外部时钟源是晶体振荡器。信号源连接单片机的 XTALI（P1.7）和 XTALO（P1.6）

XCFILTER[1:0]：外部晶体振荡器抗干扰控制寄存器

00：外部晶体振荡器频率在 48M～24M 之间时可选择此项

01：外部晶体振荡器频率在 24M～12M 之间时可选择此项

1x：外部晶体振荡器频率在 12M 及以下时可选择此项

GAIN：外部晶体振荡器振荡增益控制位

0：关闭振荡增益（低增益）

1：使能振荡增益（高增益）

XOSCST：外部晶体振荡器频率稳定标志位。（只读位）

当外部晶体振荡器从停振状态开始使能后，必须经过一段时间，振荡器的频率才会稳定，当振荡器频率稳定后，时钟控制器会自动将 XOSCST 标志位置 1。所以当用户程序需要将时钟切换到使用外部晶体振荡器时，首先必须设置 ENXOSC=1 使能振荡器，然后一直查询振荡器稳定标志位 XOSCST，直到标志位变为 1 时，才可进行时钟源切换。

8.3.5 内部低速 IRC 控制寄存器（IRC32KCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IRC32KCR	7EFE04H	ENIRC32K	-	-	-	-	-	-	IRC32KST

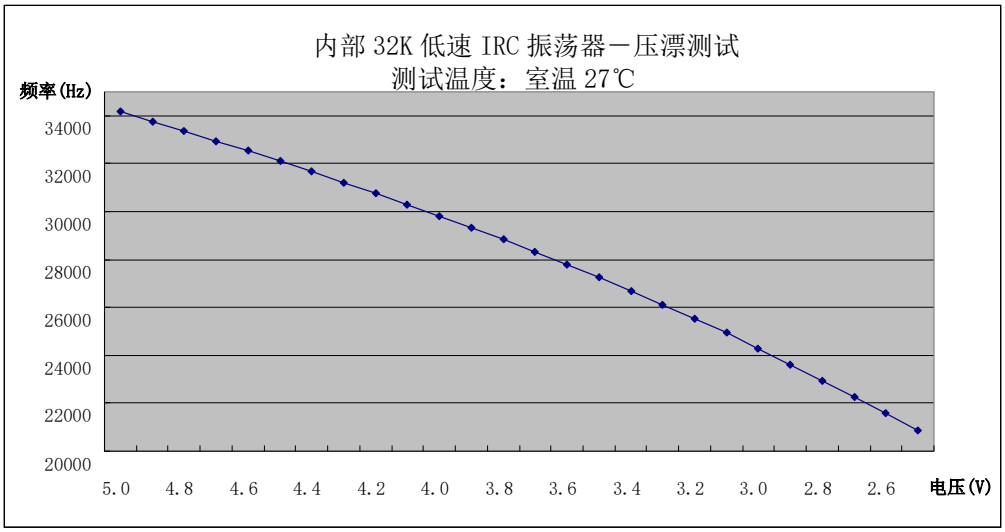
ENIRC32K：内部低速 IRC 使能位

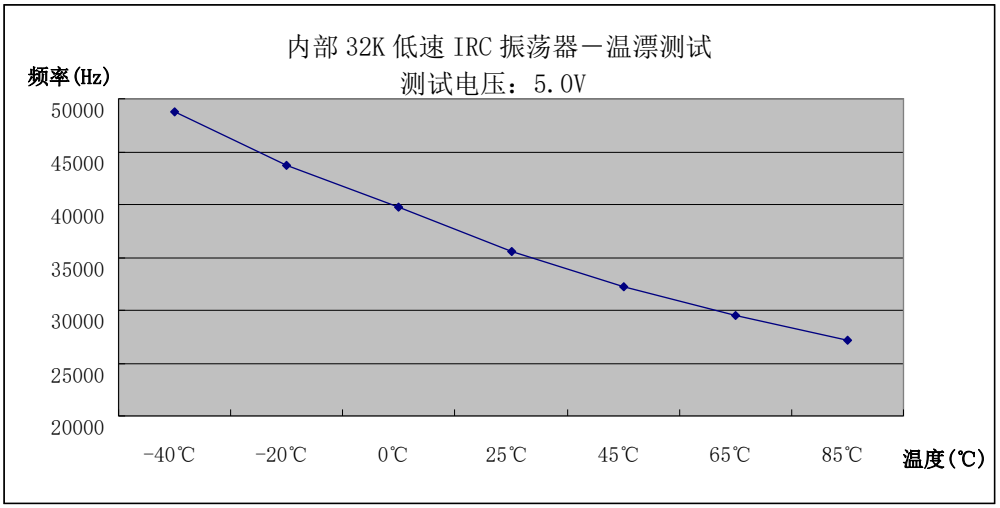
0：关闭内部低速 IRC

1：使能内部低速 IRC

IRC32KST：内部低速 IRC 频率稳定标志位。（只读位）

当内部低速 IRC 从停振状态开始使能后，必须经过一段时间，振荡器的频率才会稳定，当振荡器频率稳定后，时钟控制器会自动将 IRC32KST 标志位置 1。所以当用户程序需要将时钟切换到使用内部低速 IRC 时，首先必须设置 ENIRC32K=1 使能振荡器，然后一直查询振荡器稳定标志位 IRC32KST，直到标志位变为 1 时，才可进行时钟源切换。





8.3.6 外部 32K 振荡器控制寄存器（X32KCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
X32KCR	7EFE08H	ENX32K	GAIN32K	-	-	-	-	-	X32KST

ENX32K：外部 32K 晶体振荡器使能位

0：关闭外部 32K 晶体振荡器

1：使能外部 32K 晶体振荡器

GAIN32K：外部 32K 晶体振荡器振荡增益控制位

0：关闭 32K 振荡增益（低增益）

1：使能 32K 振荡增益（高增益）

X32KST：外部 32K 晶体振荡器频率稳定标志位。（只读位）

8.3.7 内部 48MHz 高速 IRC 控制寄存器（IRC48MCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IRC48MCR	7EFE07H	ENIRC48M	-	-	-	-	-	-	IRC48MST

ENIRC48M：内部 48M 高速 IRC 使能位

0：关闭内部 48M 高速 IRC

1：使能内部 48M 高速 IRC

IRC48MST：内部 48M 高速 IRC 频率稳定标志位。（只读位）

当内部 48M 高速 IRC 从停振状态开始使能后，必须经过一段时间，振荡器的频率才会稳定，当振荡器频率稳定后，时钟控制器会自动将 IRC48MST 标志位置 1。所以当用户程序需要将时钟切换到使用内部 48M 高速 IRC 时，首先必须设置 ENIRC48M=1 使能振荡器，然后一直查询振荡器稳定标志位 IRC48MST，直到标志位变为 1 时，才可进行时钟源切换。

8.3.8 高速 PLL 控制寄存器（HPLLCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HPLLCR	7EFE0CH	ENHPLL	HSI2SCK[1:0]		HPLLCKS	HPLLDIV[3:0]			

ENHPLL：高速 PLL 使能位

0：关闭 HPLL

1：使能 HPLL

HSI2SCK[1:0]：两组高速 I2S 时钟选择

HSI2SCK[1:0]	高速 I2S 时钟
0	HPLL1 输出时钟/2
1	HPLL2 输出时钟/2
2	HPLL1 输出时钟
3	HP1LL2 输出时钟

HPLLCKS：高速 PLL 输入时钟源选择

0：内部基本时钟（BASECLK）

1：内部 48M 的 IRC 时钟

HPLLDIV[3:0]：高速 PLL 锁频的除频数（倍频数）

HPLLDIV[3:0]	倍频数	HPLL 输出频率@6MHz
0	52	312MHz
1	54	324MHz
2	56	336MHz
3	58	348MHz
4	60	360MHz
5	62	372MHz
6	64	384MHz
7	66	396MHz
8	68	408MHz
9	70	420MHz
10	72	432MHz
11	74	444MHz
12	76	456MHz
12	78	468MHz
14	80	480MHz
15	82	492MHz

8.3.9 高速 PLL 预分频寄存器（HPLLPDIV）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HPLLPDIV	7EFE0DH	HPLLPDIV[7:0]							

HPLLPDIV[7:0]：高速 PLL 输入频率的预分频数。

必须保证 HPLL 的锁频基频为 6MHz

比如 HPLL 的输入频率为 48MHz，则需要将 HPLLPDIV 设置为 8。

8.3.10 高速 PLL2 控制寄存器（HPLL2CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HPLL2CR	7EFE0EH	ENHPLL2	HSIOCK[1:0]		HPLL2CKS	HPLL2DIV[3:0]			

ENHPLL2：高速 PLL2 使能位

0：关闭 HPLL2

1：使能 HPLL2

HSIOCK[1:0]：两组高速外设时钟选择

HSIOCK[1:0]	高速外设时钟
0	HPLL1 输出时钟/2
1	HPLL2 输出时钟/2
2	HPLL1 输出时钟
3	HPLL2 输出时钟

HPLL2CKS：高速 PLL2 输入时钟源选择

0：内部基本时钟（BASECLK）

1：内部 48M 的 IRC 时钟

HPLL2DIV[3:0]：高速 PLL2 锁频的除频数（倍频数）

HPLL2DIV[3:0]	倍频数	HPLL2 输出频率@6MHz
0	52	312MHz
1	54	324MHz
2	56	336MHz
3	58	348MHz
4	60	360MHz
5	62	372MHz
6	64	384MHz
7	66	396MHz
8	68	408MHz
9	70	420MHz
10	72	432MHz
11	74	444MHz
12	76	456MHz
12	78	468MHz
14	80	480MHz
15	82	492MHz

8.3.11 高速 PLL2 预分频寄存器（HPLL2PDIV）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HPLL2PDIV	7EFE0FH	HPLL2PDIV[7:0]							

HPLL2PDIV[7:0]：高速 PLL2 输入频率的预分频数。

必须保证 HPLL2 的锁频基频为 6MHz

比如 HPLL2 的输入频率为 48MHz，则需要将 HPLL2PDIV 设置为 8。

8.3.12 主时钟输出控制寄存器（MCLKOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
MCLKOCR	7EFE05H	MCLKO_S	MCLKODIV[6:0]						

MCLKODIV[6:0]：主时钟输出分频系数

MCLKODIV[6:0]	系统时钟分频输出频率
0000000	不输出时钟
0000001	SYSclk/1
0000010	SYSclk /2
0000011	SYSclk /3
...	...
1111110	SYSclk /126
1111111	SYSclk /127

MCLKO_S：系统时钟输出管脚选择

0：系统时钟分频输出到 P5.4 口

1：系统时钟分频输出到 P1.6 口

设置主时钟输出前，需要对 HIRCCR 中的 MCLKO_CKS 进行配置，选择时钟源

8.3.13 内部高速 IRC 时钟恢复振荡到稳定需要等待的时钟数控制寄存器（IRCDB）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IRCDB	FE06H								

IRCDB[7:0]：内部高速 IRC 时钟从主时钟停振/省电模式（掉电模式）恢复振荡到稳定需要等待的时钟数控制寄存器。**上电复位后初始值为 80H。**

IRCDB	系统时钟数
0	255 个时钟
1	0 个时钟
2	1 个时钟
3	2 个时钟
…	…
128	**127 个时钟（上电初始值）**
…	…
255	254 个时钟

【寄存器说明】：

1、芯片使用内部高速 IRC 时钟作为系统时钟时，当芯片从主时钟停振/省电模式（掉电模式）被唤醒后，系统会等待 IRCDB 寄存器设置的等待时钟数，再将时钟提供给 CPU 和系统，CPU 获得时钟后开始继续运行用户程序。建议在主时钟停振语句的下一句后面增加 7~9 个 NOP。

2、需要设置等待多少个时钟再给 CPU 供应时钟，让 CPU 开始跑程序，根据用户实际使用的工作频率来确定，目前测试发现运行在 33.1776MHz 附近或以下时，必须将 IRCDB 设置为 0x10 时，这样就很稳定。

3、如果用户的工作频率高于 33.1776MHz 时，建议用户程序在进入主时钟停振/省电模式前将内部高速 IRC 频率调低，再进入主时钟停振/省电模式。等芯片再次被唤醒后再将内部高速 IRC 调整到用户需要的工作频率。内部高速 IRC 时钟，出厂时有多种校准值，用户程序可以任意设置芯片出厂时预置的几种校准时钟频率。实际测试发现芯片运行在 40MHz 以下时，将 IRCDB 设置为 0x10 也是稳定的。

4、如果程序需要进入省电模式，程序初始化时必须加上下面这条语句：

```
IRCDB = 0x10;
//内部高速 IRC 时钟，停振后被唤醒后起振，等待多少个时钟数后给 MCU 供应时钟
```

8.3.14 SPIn 时钟分频寄存器（SPIn_CLKDIV）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI_CLKDIV	7EFE90H	SPI_CLKDIV[7:0]							
SPI2_CLKDIV	7EF808H	SPI2_CLKDIV[7:0]							
SPI3_CLKDIV	7EF828H	SPI3_CLKDIV[7:0]							

SPIn_CLKDIV：高速 SPI 时钟分频系数。(n=1,2,3)

SPIn_CLKDIV	SPIn 工作时钟频率
0	HSIOCLK /1
1	HSIOCLK /1
2	HSIOCLK /2
…	…
255	HSIOCLK /255

8.3.15 PWMn 时钟分频寄存器（PWMn_CLKDIV）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CLKDIV	7EFE91H	PWMA_CLKDIV[7:0]							
PWMB_CLKDIV	7EFE92H	PWMB_CLKDIV[7:0]							
PWMC_CLKDIV	7EFE96H	PWMC_CLKDIV[7:0]							
PWMD_CLKDIV	7EFE97H	PWMD_CLKDIV[7:0]							
PWME_CLKDIV	7EF68BH	PWME_CLKDIV[7:0]							
PWMF_CLKDIV	7EF68FH	PWMF_CLKDIV[7:0]							

PWMn_CLKDIV：高速 PWMn 时钟分频系数。(n=A,B,C,D,E,F)

PWMn_CLKDIV	PWMn 工作时钟频率
0	HSIOCLK /1
1	HSIOCLK /1
2	HSIOCLK /2
…	…
255	HSIOCLK /255

8.3.16 TFPU 时钟分频寄存器（TFPU_CLKDIV）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TFPU_CLKDIV	7EFE93H	TFPU_CLKDIV[7:0]							

TFPU_CLKDIV：TFPU 时钟分频系数。

TFPU_CLKDIV	TFPU 工作时钟频率
0	HSIOCLK /1
1	HSIOCLK /1
2	HSIOCLK /2
…	…
255	HSIOCLK /255

8.3.17 I2Sn 时钟分频寄存器（I2Sn_CLKDIV）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2S_CLKDIV	7EFE94H	I2S_CLKDIV[7:0]							
I2S2_CLKDIV	7EFE94H	I2S2_CLKDIV[7:0]							

I2Sn_CLKDIV：I2Sn 时钟分频系数。(n=1,2)

I2Sn_CLKDIV	I2Sn 工作时钟频率
0	HSI2SCLK /1
1	HSI2SCLK /1
2	HSI2SCLK /2
…	…
255	HSI2SCLK /255

使用 I2S 一定要启动 PLL，选择 HPLL1/2 或 HPLL2/2。

I2S 时钟 = HPLL1 / 2 / I2S_CLKDIV。

推荐主频 2 个：

1、主频 24.576MHz，PLL 倍频系数为 40，则 HPLL 时钟 = 24.576MHz/4*2*40 = 491.52MHz

2、主频 24MHz，PLL 倍频系数为 41，则 HPLL 时钟 = 24/4 *2*41 = 492MHz，跟 491.52MHz 误差不到 0.1%。

推荐 I2S_CLKDIV=2，这样可以支持采样率为 8KHz ~ 96KHz，后面有推荐的采样率。

此时，I2S 时钟 = HPLL/2/2 = HPLL/4 = 491.52MHz/4 = 122.88MHz。

8.4 STC32G144K246 系列内部高速 IRC 时钟频率调整

STC32G144K246 系列单片机内部均集成有一颗高精度内部 IRC 振荡器。在用户使用 ISP 下载软件进行下载时，ISP 下载软件会根据用户所选择/设置的频率自动进行调整，一般频率值的精度可调整到±0.3%以下，调整后的频率在全温度范围内（－40℃~85℃）的温漂可达-1.35%~1.30%。如需工作在-40℃~125℃，则内部高精度 IRC 时钟的温漂是±3%，这时如串口通讯 1 个字节分两次发送，温漂可以认为是±1.5%。当然也可以使用外部高频晶振，或者使用外部 32768Hz-RTC 晶振自动校准内部高速 IRC 时钟。

STC32G144K246 系列内部 IRC 有 4 个频段，频段的中心频率分别为 10MHz、27MHz、44MHz 和 72MHz，每个频段的调节范围约为±27%（注意：不同的芯片以及不同的生成批次可能会有约 5%左右的制造误差）。

注意：对于一般用户，内部 IRC 频率的调整可以不用关心，因为频率调整工作在进行 ISP 下载时已经自动完成了。所以若用户不需要自行调整频率，那么下面相关的 4 个寄存器也不能随意修改，否则可能会导致工作频率变化。

8.4.1 IRC 频段选择寄存器（IRCBAND）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IRCBAND	9DH	USBCKS	USBCKS2	-	-	-	-	SEL[1:0]	

SEL[1:0]：频段选择

00：选择 10MHz 频段

01：选择 27MHz 频段

10：选择 44MHz 频段

11：选择 72MHz 频段

8.4.2 内部 IRC 频率调整寄存器（IRTRIM）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IRTRIM	9FH	IRTRIM[7:0]							

IRTRIM[7:0]：内部高精度 IRC 频率调整寄存器

IRTRIM 可对 IRC 频率进行 256 个等级的调整，每个等级所调整的频率值在整体上呈线性分布，局部会有波动。宏观上，每一级所调整的频率约为 0.24%，即 IRTRIM 为（n+1）时的频率比 IRTRIM 为（n）时的频率约快 0.24%。但由于 IRC 频率调整并非每一级都是 0.24%（每一级所调整频率的最大值约为 0.55%，最小值约为 0.02%，整体平均值约为 0.24%），所以会造成局部波动。

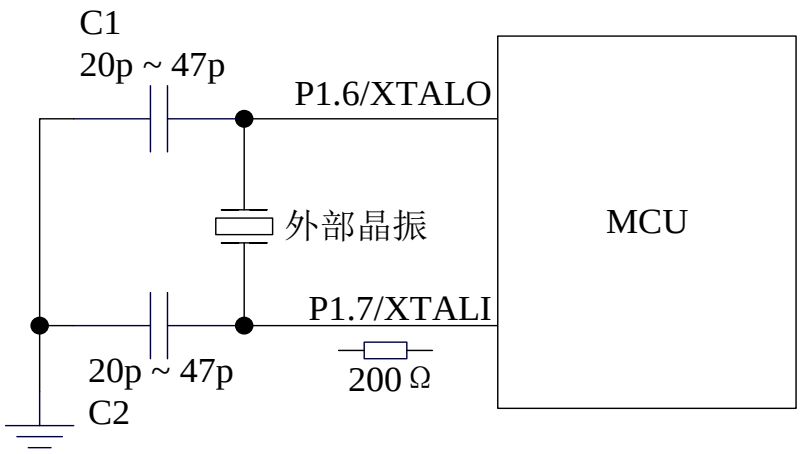
8.4.3 内部 48M IRC 频率调整寄存器（IRC48MnTRIM）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IRC48MATRIM	7FFE09H				IRC48MTRIM[12:8]				
IRC48MBTRIM	7FFE0AH	IRC48MTRIM[7:0]							

IRC48MTRIM[12:0]：内部 48M 的高速 IRC 频率调整寄存器。IRC48M 一般用作 USB 时钟，USB 与电脑通讯时会实时对 IRC48M 自动校准，用户软件一般不需要设置这两个寄存器。

8.5 外部晶振及外部时钟电路

8.5.1 外部晶振输入电路



注意：

1、一般不需外部晶振，我司内部高速HIRC时钟完全满足串口通讯需求。
如接外部晶振，C1、C2两个电容一定不能省，否则晶振可能不起振

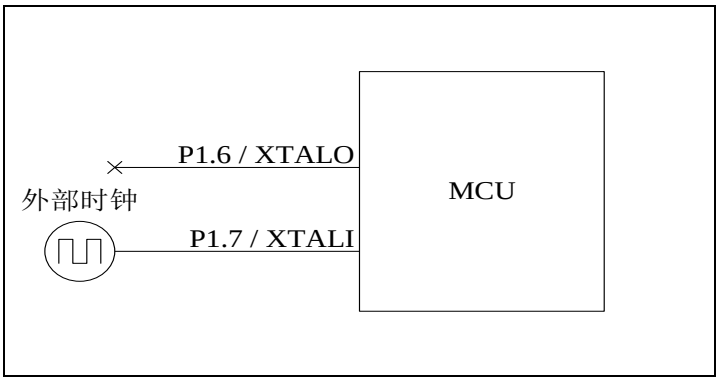
2、RTC时钟必须使用外部32768晶振，外挂电容要<=20pF，例如20p/15p/10p

3、若外部连接高速晶振（例如：24MHz），外挂电容47pF为宜

4、外部晶振会增加额外的系统功耗，32768晶振增加1.5uA的功耗，
高速晶振（例如：24MHz）会增加2mA以上的功耗

5、XTALI / XTAL1 串个 200欧的电阻，有利于外部晶振起振
XTALO / XTAL2 串个 1K的电阻，减少外部晶振振荡后对外的电磁辐射

8.5.2 外部时钟输入电路（P1.6 为高阻输入模式，可当输入口使用）



注：当使用内部时钟时，P1.6/P1.7 都可以当普通 I/O 使用。当 P1.7 口外接有源时钟或外接其他时钟源时，P1.6 口为高阻输入模式，可当输入口使用，此时 P1.6 的端口模式不可改变。有 RTC 功能的 MCU，外部 32768 时钟可从 P1.7 口输入。

9 校准内部 IRC 时钟，利用外部 32768Hz 晶振，降低电磁辐射

== 自动频率校准，校准内部高速 IRC 时钟，自动追频（CRE）

产品线	自动追频
STC32G144K246 系列	●

STC32G144K246 单片机系列内建一个频率自动校准模块（CRE），CRE 模块是使用外部的 32.768KHz 晶振对内部高速 IRC（HIRC）的 IRTRIM 寄存器进行自动调整，以达到自动频率校准的功能。

CPU 使用内部高速 IRC 的时钟，时钟对外电磁辐射低；如 CPU 使用外部高速晶振，外部高速晶振振荡后对外电磁辐射大；如对内部高速 IRC 时钟的温漂要求比较严苛，可以用外部 32768Hz-RTC 晶振来自动校准内部高速 IRC 时钟。

需要使用自动校准时，只需要根据给定的公式设置好目标频率的计数值和误差范围，然启动 CRE 模块，硬件便会进行自动频率校准，当 HIRC 的频率达到用户所设置误差范围内时，校准完成标志会被置位。

9.1 相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CRECR	CRE 控制寄存器	7EFDA8H	ENCRE	MONO	UPT[1:0]		CREHF	CREINC	CREDEC	CRERDY	0000,0000
CRECNTH	CRE 校准目标寄存器	7EFDA9H	CNT[15:8]								0000,0000
CRECNTL	CRE 校准目标寄存器	7EFDAAH	CNT[7:0]								0000,0000
CRERES	CRE 分辨率控制寄存器	7EFDABH	RES[7:0]								0000,0000

9.1.1 CRE 控制寄存器（CRECR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CRECR	7EFDA8H	ENCRE	MONO	UPT[1:0]		CREHF	CREINC	CREDEC	CRERDY

ENCRE：CRE模块控制位

0：关闭 CRE 模块。

1：使能 CRE 模块。

MONO：自动校准步幅控制

0：单步模式。每个校准周期，硬件自动将 IRTRIM 递增或递减 1。

1：双步模式。每个校准周期，硬件自动将 IRTRIM 递增或递减 2。

单步模式比双步模式校准后的 IRC 精度更高，但自动校准的时间比双步模式长。

UPT[1:0]：CRE 校准周期选择

UPT[1:0]	校准周期
00	1ms
01	4ms
10	32ms
11	64ms

CREHF：高频模式选择

0：低频模式（目标频率小于或等于 50MHz）。

1：高频模式（目标频率大于 50MHz）。

CREINC：CRE校准正处于上调状态。只读位。

CREDEC：CRE校准正处于下调状态。只读位。

CRERDY：CRE校准完成状态。只读位。

0：CRE 校准功能未启动或者未校准完成。

1：CRE 校准已完成。

9.1.2 CRE 校准计数值寄存器（CRECNT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CRECNTH	7EFDA9H	CRECNT[15:8]							
CRECNTL	7EFDAAH	CRECNT[7:0]							

CRECNT[15:0]：16位校准计数值。

目标校准值计算公式：

低频模式（CREHF=0）：CRECNT = (16 * 目标频率(Hz)) / 32768

高频模式（CREHF=1）：CRECNT = (8 * 目标频率(Hz)) / 32768

（详细设置见范例程序）

9.1.3 CRE 校准误差值寄存器（CRERES）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CRERES	7EFDABH	CRERES[7:0]							

CRERES[7:0]：8位校准误差值（解析度控制）。

由于内部高速IRC的解析度远低于外部的32.768K晶振，最终的校准值无法与CRECNT所设置的目标值完全一致，所以必须通过CRERES寄存器设定一个误差范围。

校准误差计算公式：

CRERES = 误差范围(%) * 目标校准值

（误差范围一般控制在 1%~0.3%即可，不建议超出此范围）

（详细设置见范例程序）

9.2 内部高速 IRC 的 4 个频段的中心频率，供参考

单位（MHz）		样品 1	样品 2	样品 3	样品 4	平均值
6M 频段	最小频率	4.39	4.42	4.43	4.61	4.46
	中间频率	6.22	6.26	6.20	6.53	6.30
	最大频率	8.03	8.05	8.08	8.36	8.13
10M 频段	最小频率	7.20	7.29	7.34	7.74	7.39
	中间频率	10.11	10.26	10.33	10.91	10.40
	最大频率	12.99	13.11	13.26	13.90	13.32
27M 频段	最小频率	20.06	20.11	20.46	21.11	20.43
	中间频率	27.82	27.67	28.24	29.14	28.22
	最大频率	35.14	34.97	35.75	36.83	35.67
44M 频段	最小频率	27.79	28.20	28.72	29.70	28.60
	中间频率	41.45	41.74	42.55	44.04	42.45
	最大频率	54.04	54.41	55.50	57.47	55.35

10 复位、看门狗、掉电唤醒专用定时器与电源管理

10.1 系统复位

STC32G144K246 系列单片机的复位分为硬件复位和软件复位两种。

硬件复位时，所有的寄存器的值会复位到初始值，系统会重新读取所有的硬件选项。同时根据硬件选项所设置的上电等待时间进行上电等待。硬件复位主要包括：

- 上电复位
- 低压复位
- 复位脚复位（低电平复位）
- 看门狗复位

软件复位时，除与时钟相关的寄存器保持不变外，其余的所有寄存器的值会复位到初始值，软件复位不会重新读取所有的硬件选项。软件复位主要包括：

- 写 IAP_CONTR 的 SWRST 所触发的复位

相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
WDT_CONTR	看门狗控制寄存器	C1H	WDT_FLAG	-	EN_WDT	CLR_WDT	IDL_WDT	WDT_PS[2:0]			0x00,0000
IAP_CONTR	IAP 控制寄存器	C7H	IAPEN	SWBS	SWRST	CMD_FAIL	SWBS2	-			0000,x000
RSTCFG	复位配置寄存器	FFH	-	ENLVR	-	P47RST	-	-	LVDS[1:0]		0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
RSTFLAG	复位标志寄存器	7EFE99H	-	-	-	LVDRSTF	WDTRSTF	SWRSTF	ROMOVF	EXRSTF	xxx1,0100
RSTCR0	复位控制寄存器 0	7EFE9AH	-	-	-	-	RSTTM34	TSTTM2	-	RSTTM01	xxxx,00x0
RSTCR1	复位控制寄存器 1	7EFE9BH	-	-	-	-	RSTUR4	RSTUR3	RSTUR2	RSTUR1	xxxx,0000
RSTCR2	复位控制寄存器 2	7EFE9CH	RSTCAN2	RSTCAN	RSTLIN	RSTRTC	RSTPWMB	RSTPWMA	RSTI2C	RSTSPI	0000,0000
RSTCR3	复位控制寄存器 3	7EFE9DH	RSTFPU	RSTDMA	RSTLCM	RSTLCD	RSTLED	RSTTKS	RSTCMP	RSTADC	0000,0000
RSTCR4	复位控制寄存器 4	7EFE9EH	-	-	-	-	-	-	-	RSTMDU	xxxx,xxx0
RSTCR5	复位控制寄存器 5	7EFE9FH	-	-	-	-	-	-	-	-	xxxx,xxxx

10.1.1 看门狗控制寄存器（WDT_CONTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
WDT_CONTR	C1H	WDT_FLAG	-	EN_WDT	CLR_WDT	IDL_WDT	WDT_PS[2:0]		

WDT_FLAG：看门狗溢出标志

看门狗发生溢出时，硬件自动将此位置 1，需要软件清零。

EN_WDT： 看门狗使能位

0： 对单片机无影响

1： 启动看门狗定时器

CLR_WDT：看门狗定时器清零

0： 对单片机无影响

1： 清零看门狗定时器，硬件自动将此位复位

IDL_WDT： IDLE 模式时的看门狗控制位

0： IDLE 模式时看门狗停止计数

1： IDLE 模式时看门狗继续计数

WDT_PS[2:0]：看门狗定时器时钟分频系数

WDT_PS[2:0]	分频系数	12M 主频时的溢出时间	20M 主频时的溢出时间
000	2	≈ 65.5 毫秒	≈ 39.3 毫秒
001	4	≈ 131 毫秒	≈ 78.6 毫秒
010	8	≈ 262 毫秒	≈ 157 毫秒
011	16	≈ 524 毫秒	≈ 315 毫秒
100	32	≈ 1.05 秒	≈ 629 毫秒
101	64	≈ 2.10 秒	≈ 1.26 秒
110	128	≈ 4.20 秒	≈ 2.52 秒
111	256	≈ 8.39 秒	≈ 5.03 秒

看门狗溢出时间计算公式如下：

$$\text{看门狗溢出时间} = \frac{12 \times 32768 \times 2^{(\text{WDT_PS}+1)}}{\text{SYSclk}}$$

10.1.2 IAP 控制寄存器（IAP_CONTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IAP_CONTR	C7H	IAPEN	SWBS	SWRST	CMD_FAIL	SWBS2	-	-	-

SWBS/SWBS2：软件复位启动选择

SWBS	SWBS2	复位	说明
0	0	软件复位后从用户程序区开始执行代码	用户数据区的数据保持不变
0	1	软件复位后从用户系统区开始执行代码	用户数据区的数据保持不变
1	x	软件复位后从系统 ISP 区开始执行代码	用户数据区的数据会被初始化

SWRST：软件复位触发位

0：对单片机无影响

1：触发软件复位

范例程序：

源程序区 （从此程序区）	目标程序区 （复位到此程序区）	代码
用户程序区	系统 ISP 区	IAP_CONTR = 0x60;
用户系统区		
用户程序区	用户系统区	IAP_CONTR = 0x28;
系统 ISP 区		
用户系统区	用户程序区	IAP_CONTR = 0x20;
系统 ISP 区		

10.1.3 复位配置寄存器（RSTCFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
RSTCFG	FFH	-	ENLVR	-	P47RST	-	-	LVDS[1:0]	

ENLVR：低压复位控制位

0：禁止低压复位。当系统检测到低压事件时，会产生低压中断

1：使能低压复位。当系统检测到低压事件时，自动复位

P47RST：RST 管脚功能选择

0：RST 管脚用作普通 I/O 口（P5.4）

1：RST 管脚用作复位脚（低电平复位）

LVDS[1:0]：低压检测门槛电压设置

LVDS[1:0]	低压检测门槛电压
00	2.0V
01	2.4V
10	2.7V
11	3.0V

10.1.4 复位标志寄存器（RSTFLAG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
RSTFLAG	7EFE99H	-			LVDRSTF	WDTRSTF	SWRSTF	ROMOVF	EXRSTF

LVDRSTF：LVD 低压复位标志

读 0：无意义

读 1：当前的复位是由 LVD 低压复位所触发（上电复位默认值为 1）

写 0：无效果

写 1：清除 LVDRST 标志位

WDTRSTF：看门狗复位标志

读 0：无意义（上电复位默认值为 0）

读 1：当前的复位是由看门狗溢出所触发

写 0：无效果

写 1：清除 WDTRST 标志位

SWRSTF：软复位标志

读 0：无意义

读 1：当前的复位是由软件写 SWRST（IAP_CONTR.5）所触发（用户程序复位默认值为 1）

写 0：无效果

写 1：清除 SWRST 标志位

ROMOVF：代码区溢出标志

读 0：无意义（上电复位默认值为 0）

读 1：当前的复位是由于 CPU 执行代码到非程序区导致的代码区溢出所触发

写 0：无效果

写 1：清除 ROMOV 标志位

EXRSTF：外部复位标志

读 0：无意义（上电复位默认值为 0）

读 1：当前的复位是外部复位脚（P5.4/RST）被拉低所触发

写 0：无效果

写 1：清除 EXRST 标志位

关于用户程序软复位到系统区进行 USB-ISP 下载的说明：

上电时需要 P3.2 同时接地才可进入 USB-ISP 下载模式，为方便用户自主控制 ISP 下载，特别增加：当用户程序软复位到系统区时，可不用 P3.2 接地就可进行 USB-ISP 下载。此功能的判断方式为进入 ISP 后判断 SWRSTF（RSTFLAG.2）是否为 1，若为 1 表示是用户软复位到系统区，则不用 P3.2 接地，否则需要 P3.2 接地。若用户需要按键复位或者看门狗复位或者低压复位后需要进行 USB-ISP 下载，可保持 SWRSTF（RSTFLAG.2）为 1，否则请在用户代码初始化时将 SWRSTF（RSTFLAG.2）写 1，以清零 SWRSTF（RSTFLAG.2）。

详细各种情况说明见下表

基本情况介绍：上电冷启动后，硬件自动将 SWRSTF（RSTFLAG.2）标志位清 0，并进入到系统程序区，此时无论是否进行 ISP 下载，从系统程序区复位到用户程序区，硬件都会自动将 SWRSTF 标志位置 1。用户程序中可对 SWRSTF 标志位写 1 清 0，也可不对 SWRSTF 标志位写 1 清 0，即维持 SWRSTF 标志位的值为 1。

SWRSTF 标志位（RSTFLAG.2）	**复位情况及 USB 下载情况**	**USB 下载是否需要 P3.2 接地**
重新上电复位， 复位脚按键复位， 看门狗复位， LVD 低压复位 都会复位到系统程序区，最后都会从系统程序区软复位到用户程序区，并将 SWRSTF 标志位置 1 如果用户程序中对 SWRSTF 写 1 则会清 0	从用户程序区软复位到系统区时，硬件自动将 SWRSTF 软件标志位重新置 1，此时系统程序判断到此位为 1，此时 USB 下载不需要 P3.2 接地	不需要
	复位脚按键复位，看门狗复位，低压复位， 复位到系统区复位到系统区，系统程序判断到此位为 0，此时 USB 下载需要 P3.2 接地	需要
	重新上电复位到系统区，硬件自动将 SWRSTF 复位为 0，系统程序判断到此位为 0，此时 USB 下载需要 P3.2 接地	需要
重新上电复位， 复位脚按键复位， 看门狗复位， LVD 低压复位 都会复位到系统程序区，最后都会从系统程序区软复位到用户程序区，并将 SWRSTF 标志位置 1 如果用户程序不对 SWRSTF 写 1 清 0，SWRSTF 保持为 1	从用户程序区软复位到系统区时，硬件自动将 SWRSTF 软件标志位重新置 1，此时系统程序判断到此位为 1，此时 USB 下载不需要 P3.2 接地	不需要
	复位脚按键复位，看门狗复位，低压复位， 复位到系统区复位到系统区，系统程序判断到此位为 1，此时 USB 下载不需要 P3.2 接地	不需要
	重新上电复位到系统区，硬件自动将 SWRSTF 复位为 0，系统程序判断到此位为 0，此时 USB 下载需要 P3.2 接地	需要

总结：

1、用户程序软件复位到系统区，无论用户软件是否对 SWRSTF 写 1 清 0，USB 下载均不需要 P3.2 接地

2、对芯片重新上电复位到系统区，无论用户软件是否对 SWRSTF 写 1 清 0，USB 下载均需要 P3.2 接地

3、按键复位/看门狗复位/低压复位到系统区，如果用户软件对 SWRSTF 写 1 清 0，则 USB 下载需要 P3.2 接地；如果用户不对 SWRSTF 写 1 清 0，保持 SWRSTF 为 1，则 USB 下载不需要 P3.2 接地

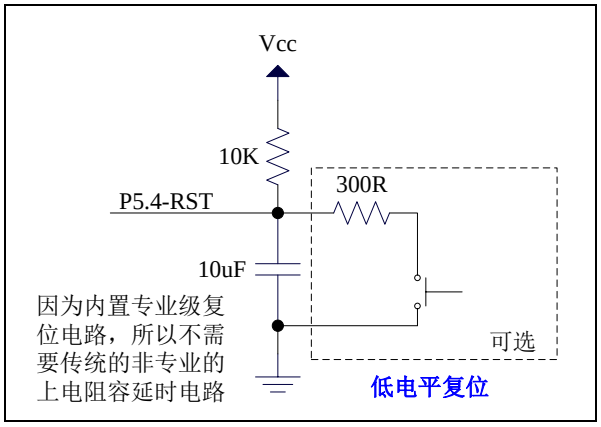
10.1.5 复位控制寄存器（RSTCRx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
RSTCR0	7EFE9AH	-	-	-	-	RSTTM34	RSTTM2	-	RSTTM01
RSTCR1	7EFE9BH	-	-	-	-	RSTUR4	RSTUR3	RSTUR2	RSTUR1
RSTCR2	7EFE9CH	RSTCAN2	RSTCAN	RSTLIN	RSTRTC	RSTPWMB	RSTPWMA	RSTI2C	RSTSPI
RSTCR3	7EFE9DH	RSTFPU	RSTDMA	RSTLCM	RSTLCD	RSTLED	RSTTKS	RSTCMP	RSTADC
RSTCR4	7EFE9EH	-	-	-	-	-	-	-	RSTMDU

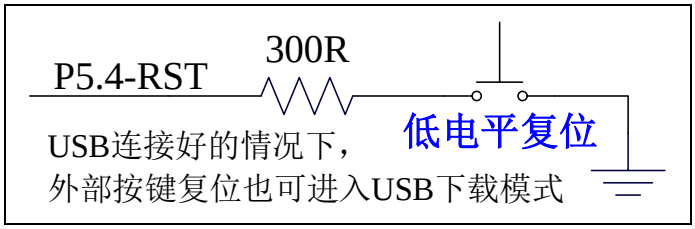
RSTTM01：TIMER0/1 复位控制位
RSTTM2：TIMER2 复位控制位
RSTTM34：TIMER3/4 复位控制位
RSTUARTn：UART1/2/3/4 复位控制位
RSTSPI：SPI 复位控制位
RSTI2C：I2C（SSB）复位控制位
RSTPWMA：PWMA 复位控制位
RSTPWMB：PWMB 复位控制位
RSTRTC：RTC 复位控制位
RSTLIN：LIN 复位控制位
RSTCAN：CAN 复位控制位
RSTCAN2：CAN2 复位控制位
RSTADC：ADC 复位控制位
RSTCMP：CMP（比较器）复位控制位
RSTTKS：TKS（TouchKey）复位控制位
RSTLED：LED 驱动复位控制位
RSTLCD：LCD 驱动复位控制位
RSTLCM：TFT 彩屏驱动复位控制位
RSTDMA：DMA 复位控制位
RSTFPU：FPU 复位控制位
RSTMDU：MDU32 复位控制位

写 1：复位相应的外设模块模块，需软件清零

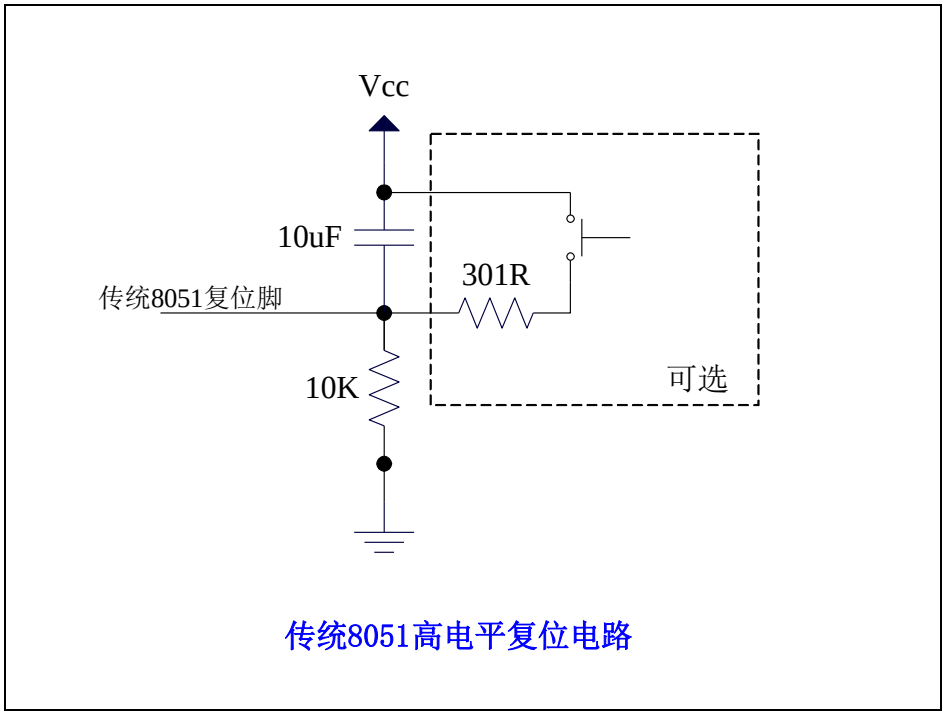
10.1.6 内置专业级复位电路，不需传统的阻容式上电延时电路



10.1.7 外部低电平按键复位电路



10.1.8 传统 8051 高电平上电复位参考电路



上图为传统 8051 的高电平复位电路，STC32G144K246 的复位为低电平复位，与传统复位电路不同

10.2 主时钟停振/省电模式，系统电源管理

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PCON	电源控制寄存器	87H	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL	0011,0000

10.2.1 电源控制寄存器（PCON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PCON	87H	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL

LVDF：低压检测标志位。当系统检测到低压事件时，硬件自动将此位置 1，此位需要用户软件清零。
LVD 低压检测有低压检测门槛电压可选： 2.0V/2.4V/2.7V/3.0V，误差±1%。
LVD 低压检测在 ISP 烧录时默认设置是低压即复位。
当允许 LVD 低压检测复位，并且电压低到指定的 LVD 复位门槛电压后，会产生 LVD 复位。
如设置不允许 LVD 低压检测复位，用户程序可中允许产生 低压检测中断。
为了省电，在省电模式/时钟停止振荡模式，LVD 不工作，不会产生复位。

POF： 上电复位标志位。上电复位硬件自动置 1。需要软件清 0，硬件不会自动清 0。
当 MCU-Vcc 电压 低于 1.65V±10% 以下时，STC8/STC32/AI8051U 系列 MCU 都会掉电复位。
当 MCU-Vcc 电压从低于 1.65V±10% 以下往上爬升，电压爬升到比 1.65V 高后，
硬件自动会将 POF 设置为 1，产生 上电复位。
为了省电，在省电模式/时钟停止振荡模式，
电压掉到 1.2V 以下时，才会产生 掉电复位 / 永不关断的复位。
电压在爬升到比 1.65V 高后，硬件自动会将 POF 设置为 1，产生 上电复位。

PD： 主时钟停振/省电模式控制位
0： 无影响
1： 单片机进入主时钟停振/省电模式，CPU 以及全部外设均停止工作。唤醒后硬件自动清零。

注： 省电模式/主时钟停振模式，俗称掉电模式：
仅主时钟停止振荡，依赖主时钟工作的 CPU/及大部分外设功能模块，
无时钟无法继续工作，耗电的这部分功能模块工作被暂停后，MCU 几乎没有功耗。
不依赖主时钟工作的低功耗外设可继续工作，如：
RTC 如选择的是外部 32768RTC 晶振，RTC 会继续正常工作，
RTC 工作，仅需要 1.5uA 的电流。
段码 LCD，如需要继续工作，功耗也极低，不含屏的功耗，也仅需 1.5uA。

IDL： IDLE（空闲）模式控制位
0：无影响
1：单片机进入 IDLE 模式，只有 CPU 停止工作，其他外设依然在运行。唤醒后硬件自动清零

【特别注意】

STC32G144K246系列单片机进入主时钟停振/省电模式前，必须根据工作频率正确设置IAP_TPS寄存器（若工作频率为12MHz，则需要将IAP_TPS设置为12；若工作频率为24MHz，则需要将IAP_TPS设置为24，其他频率以此类推）。

单片机从主时钟停振/省电模式唤醒后，硬件会自动对主时钟恢复起振，当主时钟开始起振后，硬件会自动等待一定的主时钟震荡周期数，这个等待的主时钟震荡周期数由IRCDB寄存器进行设置，直到主时钟最后稳定。主时钟稳定后，在开始运行用户程序前，还需要对内部的FLASH程序存储器进行唤醒，FLASH唤醒的等待时间基准是1us，要产生1us的时间基准，软件必须根据主时钟的工作频率正确设置IAP_TPS寄存器。

强烈建议用户在进行系统初始化时对IAP_TPS进行初始化。

10.2.2 内部高速 IRC 时钟恢复振荡到稳定需要等待的时钟数控制寄存器（IRCDB）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IRCDB	FE06H								

IRCDB[7:0]：内部高速 IRC 时钟从主时钟停振/省电模式（掉电模式）恢复振荡到稳定需要等待的时钟数控制寄存器。**上电复位后初始值为 80H。**

IRCDB	系统时钟数
0	255 个时钟
1	0 个时钟
2	1 个时钟
3	2 个时钟
…	…
128	**127 个时钟（上电初始值）**
…	…
255	254 个时钟

【寄存器说明】：

1、芯片使用内部高速 IRC 时钟作为系统时钟时，当芯片从主时钟停振/省电模式（掉电模式）被唤醒后，系统会等待 IRCDB 寄存器设置的等待时钟数，再将时钟提供给 CPU 和系统，CPU 获得时钟后开始继续运行用户程序。建议在主时钟停振语句的下一句后面增加 7~9 个 NOP。

2、需要设置等待多少个时钟再给 CPU 供应时钟，让 CPU 开始跑程序，根据用户实际使用的工作频率来确定，目前测试发现运行在 33.1776MHz 附近或以下时，必须将 IRCDB 设置为 0x10 时，这样就很稳定。

3、如果用户的工作频率高于 33.1776MHz 时，建议用户程序在进入主时钟停振/省电模式前将内部高速 IRC 频率调低，再进入主时钟停振/省电模式。等芯片再次被唤醒后再将内部高速 IRC 调整到用户需要的工作频率。内部高速 IRC 时钟，出厂时有多种校准值，用户程序可以任意设置芯片出厂时预置的几种校准时钟频率。实际测试发现芯片运行在 40MHz 以下时，将 IRCDB 设置为 0x10 也是稳定的。

4、如果程序需要进入省电模式，程序初始化时必须加上下面这条语句：

```
IRCDB = 0x10;
//内部高速 IRC 时钟，停振后被唤醒后起振，等待多少个时钟数后给 MCU 供应时钟
```

10.3 可以唤醒省电模式/主时钟停振模式的中断资源

唤醒源	相应端口	边沿类型	唤醒 STOP	唤醒 IDLE
RTC 中断			✓	✓
掉电唤醒定时器唤醒，不是中断			✓	✗
定时器 11 中断	T11（P7.6/P1.4/PA.6/P4.4）		✓	✓
外部中断	INT0（P3.2） INT1（P3.3）	下降沿 上升沿+下降沿	✓	✓
	INT2（P3.6） INT3（P3.7） INT4（P3.0）	下降沿 上升沿 上升沿+下降沿	✓	✓
所有的普通 I/O 口都支持的外部中断	所有的 I/O 口	上升沿，下降沿 注意：不支持高电平和低电平唤醒	✓	✓
串口接收脚	RXD（P3.0/P3.6/P1.6/P4.3） RXD2（P1.0/P4.6/P8.4/PA.） RXD3（P0.0/P5.0/P8.6/PB.6） RXD4（P0.2/P5.2/PA.2/PC.0） RXD5（P0.4/P4.0/P1.4/P2.0） RXD6（P0.6/P6.2/P3.2/P2.6） RXD7（P5.0/PA.4/P9.0/P1.2） RXD8（P5.2/PA.6/P9.2/P6.0）	下降沿 注意：串口接收脚的下降沿唤醒 CPU，如果需要唤醒后能接收到完整的串口数据，需将串口波特率设置到 9600 或以下。	✓	✗
I2C 的数据输入脚	SDA（P1.4/P2.4/P7.6/P3.3） SDA2（P2.6/P3.6/P1.6/P6.0）	下降沿	✓	✗
I2S 的数据输入脚	I2SData（P3.4/P6.4/P2.3/P4.0） I2S2Data（P5.1/P6.1/P8.1/PB.2）	下降沿	✓	✗
SPI 的 SS 脚	SS（P1.2/P2.2/P5.4/P3.5） SS2（P6.4/P0.2/PB.1/P8.0） SS3（P2.2/P7.0/P8.4/P9.0）	下降沿	✓	✗
定时器的外部计数输入管脚	T0（P3.4） T1（P3.5） T2（P1.0） T3（P0.4/P0.0） T4（P0.6/P0.2） T5（P2.0） T6（P2.2） T7（P2.4） T8（P2.6） T9（P7.0/PB.0） T10（P6.0/PB.6） T17（P8.0）	下降沿	✓	✗

唤醒源	相应端口	边沿类型	唤醒 STOP	唤醒 IDLE
	T18（P9.0）			
比较器中断			✓	✓
低压检测中断			✓	✓

10.4 省电模式唤醒必须设置的寄存器，注意事项

内部高速 IRC 时钟恢复振荡到稳定需要等待的时钟数控制寄存器（IRCDB）：

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IRCDB	FE06H								

IRCDB[7:0]：内部高速 IRC 时钟从主时钟停振/省电模式（掉电模式）恢复振荡到稳定需要等待的时钟数控制寄存器。**上电复位后初始值为 80H。**

IRCDB	系统时钟数
0	255 个时钟
1	0 个时钟
2	1 个时钟
3	2 个时钟
…	…
128	**127 个时钟（上电初始值）**
…	…
255	254 个时钟

【寄存器说明】：

1、芯片使用内部高速 IRC 时钟作为系统时钟时，当芯片从主时钟停振/省电模式（掉电模式）被唤醒后，系统会等待 IRCDB 寄存器设置的等待时钟数，再将时钟提供给 CPU 和系统，CPU 获得时钟后开始继续运行用户程序。建议在主时钟停振语句的下一句后面增加 7~9 个 NOP。

2、需要设置等待多少个时钟再给 CPU 供应时钟，让 CPU 开始跑程序，根据用户实际使用的工作频率来确定，目前测试发现运行在 33.1776MHz 附近或以下时，必须将 IRCDB 设置为 0x10 时，这样就很稳定。

3、如果用户的工作频率高于 33.1776MHz 时，建议用户程序在进入主时钟停振/省电模式前将内部高速 IRC 频率调低，再进入主时钟停振/省电模式。等芯片再次被唤醒后再将内部高速 IRC 调整到用户需要的工作频率。内部高速 IRC 时钟，出厂时有多种校准值，用户程序可以任意设置芯片出厂时预置的几种校准时钟频率。实际测试发现芯片运行在 40MHz 以下时，将 IRCDB 设置为 0x10 也是稳定的。

4、如果程序需要进入省电模式，程序初始化时必须加上下面这条语句：

```
IRCDB = 0x10;
//内部高速 IRC 时钟，停振后被唤醒后起振，等待多少个时钟数后给 MCU 供应时钟
```

10.5 主时钟停振/省电模式，I/O 口如何设置才省电

===主时钟停振/省电模式，STC32G144K246 系列如何省电

主时钟停振/省电模式，I/O 口如何设置才省电，进入主时钟停振/省电模式前：

1，不用的 I/O 口，就是浮空的 I/O，关闭数字输入

2，用作模拟输入的口，一般是配置成高阻输入，也必须关闭数字输入

＝＝＝指用作 ADCx 外部模拟输入的 I/O

＝＝＝指用作比较器外部模拟输入的 I/O

3，用作高阻输入的 I/O，也必须关闭数字输入

如你 I/O 外部的输入电平 Vx 在【不是逻辑高的电压，也不是逻辑低的电压】

这时内部数字输入电路就会有翻转，就会有几十 uA 的功耗

关闭数字输入，就不会有功耗

4，I/O 外部是高电平的，你如要工作在输出，你就置高

I/O 外部是低电平的，你如要工作在输出，你就置低

否则两边的电平电位不同，就会水往低处走，有电流流进或流出

如你 I/O 外部的输入电平 Vx 在 MCU_Gnd < Vx < MCU_VCC，这时工作在输出，也会有电流流动所以进省电模式前，必须改设置为高阻输入，并关闭数字输入

5，如有启动 RTC/实时时钟功能，在省电时工作的 MCU，【P1.7/XTALI, P1.6/XTALO】

【P1.7/XTALI，P1.6/XTALO】- 接外部 32768-RTC 晶振，

这 2 个口上电默认是高阻输入，可用户程序配置为高阻输入

省电模式时必须保持高阻输入，并必须关闭数字输入

总之，省电模式时，I/O 尽量高阻输入并关闭数字输入。

10.6 掉电唤醒定时器

内部掉电唤醒定时器是一个 15 位的计数器（由{WKTCH[6:0],WKTCL[7:0]}组成 15 位）。用于唤醒处于主时钟停振/省电模式的 MCU。

10.6.1 掉电唤醒定时器计数寄存器（WKTCL，WKTCH）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
WKTCL	AAH								
WKTCH	ABH	WKTEN							

WKTEN：掉电唤醒定时器的使能控制位

0：停用掉电唤醒定时器

1：启用掉电唤醒定时器

注意：读写分别位不同的寄存器（但地址相同），写必须先写 WKTCL，再同时写 WKTCH 和 WKTEN 位，不能用读-修改-写的方式。

如果 STC32G144K246 系列单片机内置掉电唤醒专用定时器被允许（通过软件将 WKTCH 寄存器中的 WKTEN 位置 1），当 MCU 进入主时钟停振/省电模式后，掉电唤醒专用定时器开始计数，当计数值与用户所设置的值相等时，掉电唤醒专用定时器将 MCU 唤醒。MCU 唤醒后，程序从上次设置单片机进入主时钟停振/省电模式语句的下一条语句开始往下执行，注意不会进入中断。掉电唤醒之后，可以通过读 WKTCH 和 WKTCL 中的内容获取单片机在主时钟停振/省电模式中的睡眠时间。

这里请注意：用户在寄存器{WKTCH[6:0],WKTCL[7:0]}中写入的值必须比实际计数值少 1。如用户需计数 10 次，则将 9 写入寄存器{WKTCH[6:0],WKTCL[7:0]}中。同样，如果用户需计数 32767 次，则应对{WKTCH[6:0],WKTCL[7:0]}写入 7FFEH（即 32766）。（**计数值 0 和计数值 32767 为内部保留值，用户不能使用**）

内部掉电唤醒定时器固定使用内部低速 IRC 时钟，时钟频率大约为 32KHz。其中掉电唤醒定时器计数一次的时间就是由该时钟决定的。出于功耗考虑，无抗温漂压漂电路，误差较大。用户可以通过读 RAM 区 F8H 和 F9H 的内容（F8H 存放频率的高字节，F9H 存放低字节）来获取内部掉电唤醒专用定时器出厂时所记录的时钟频率。注意该频率为出厂温度（25℃）下的频率，当温度不同时，存在温漂。

掉电唤醒专用定时器计数时间的计算公式如下所示：（F_{wt} 为我们从 RAM 区 F8H 和 F9H 获取到的内部掉电唤醒专用定时器的时钟频率）

$$\text{掉电唤醒定时器定时时间} = \frac{10^6 \times 16 \times \text{计数次数}}{F_{wt}} \text{(微秒)}$$

假设 F_{wt}=32KHz，则有：

{WKTCH[6:0],WKTCL[7:0]}	掉电唤醒专用定时器计数时间
~~0（内部保留）~~	
1	$10^6 \div 32K \times 16 \times (1+1) \approx 1$ 毫秒
9	$10^6 \div 32K \times 16 \times (1+9) \approx 5$ 毫秒
99	$10^6 \div 32K \times 16 \times (1+99) \approx 50$ 毫秒
999	$10^6 \div 32K \times 16 \times (1+999) \approx 0.5$ 秒
4095	$10^6 \div 32K \times 16 \times (1+4095) \approx 2$ 秒
32766	$10^6 \div 32K \times 16 \times (1+32766) \approx 16$ 秒
~~32767（内部保留）~~	

10.7 省电模式，I/O 口如何设置才省电

省电模式，I/O 口如何设置才省电，进入主时钟停振/省电模式前：

===主时钟停振/省电模式，STC8/STC32 系列如何省电

1、不用的 I/O 口，就是浮空的 I/O，设置为高阻输入，并关闭数字输入，也不怕短路了

2、用作模拟输入的口，一般是配置成高阻输入，也必须关闭数字输入

===指用作 ADCx 外部模拟输入的 I/O

===指用作 比较器外部模拟输入的 I/O

===省电模式时，他外部来个 1.5V 附近变化的电压，

数字部分不关闭数字输入可能就会产生额外的功耗

3、用作高阻输入的 I/O，也必须关闭数字输入

如你 I/O 外部的输入电平 Vx 在 【不是逻辑高的电压，也不是逻辑低的电压】

这时内部数字输入电路就会有翻转，就会有几十 uA 的功耗

关闭数字输入，就不会有功耗

4、I/O 外部是高电平的，你如要工作在输出，你就置高

I/O 外部是低电平的，你如要工作在输出，你就置低

否则两边的电平电位不同，就会水往低处走，有电流流进或流出

如你 I/O 外部的输入电平 Vx 在 MCU_Gnd < Vx < MCU_VCC

这时工作在输出，也会有电流流动

所以进省电模式前，必须改设置为高阻输入，并关闭数字输入

5、如有启动 RTC/实时时钟功能，在省电时工作的 MCU，【P1.7/XTALI, P1.6/XTALO】

【P1.7/XTALI，P1.6/XTALO】- 接外部 32768-RTC 晶振，

这 2 个口上电默认是高阻输入，可用户程序配置为高阻输入

这个场景有外部 32768 晶振在振荡，省电模式时必须保持高阻输入，

===并必须关闭数字输入，否则浮空的口，外部不停的在变化，就会产生额外的功耗

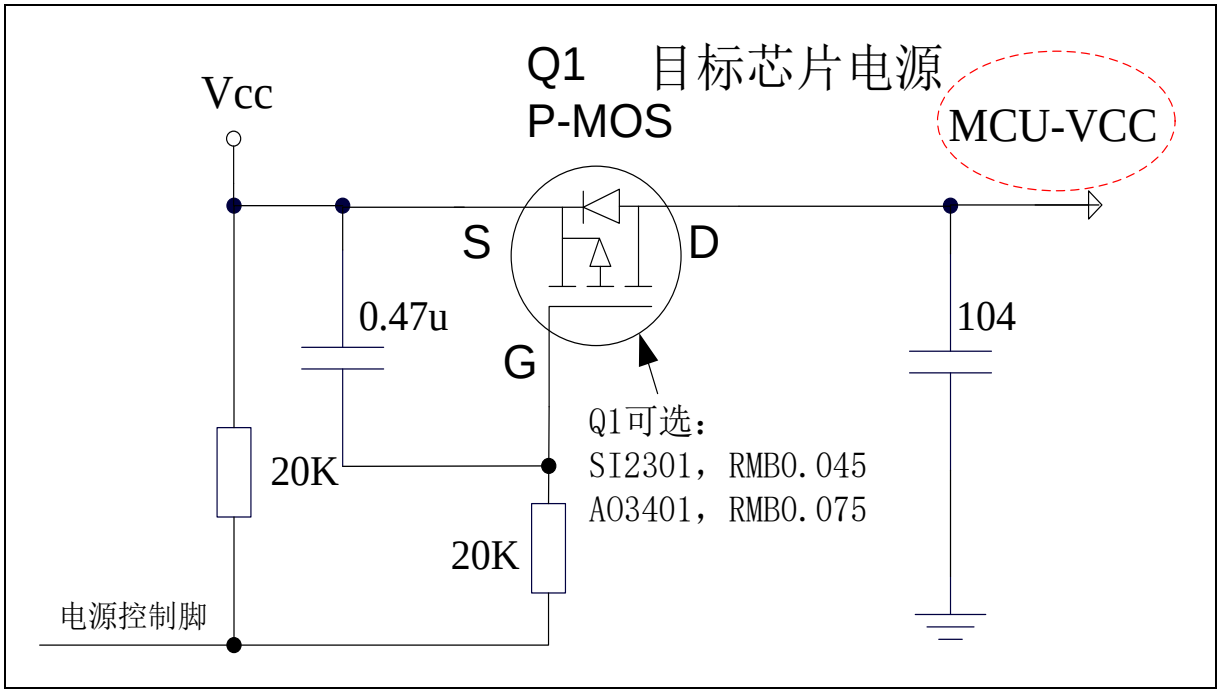
6、MCU 如有 ADC_VRef+，ADC_VRef+不能浮空，否则也会产生额外的电流

总之，省电模式时，I/O 尽量 高阻输入并关闭数字输入

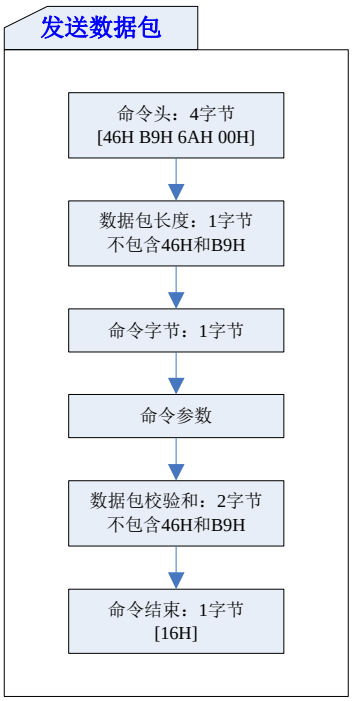
11 第三方对 STC32G144K246 下载程序范例及如何选择内部不同时钟频率

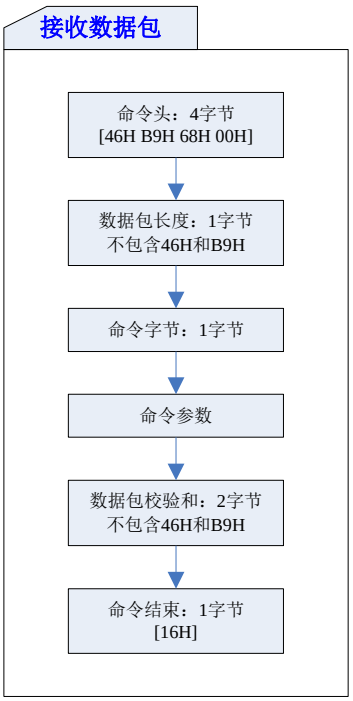
11.1 电源控制参考电路

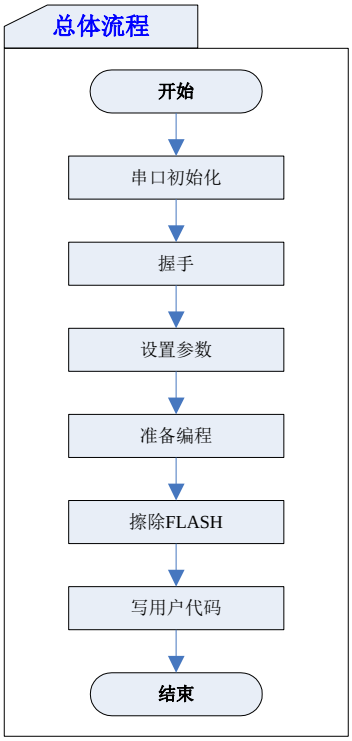
ISP 下载需要对目标进行硬件复位才能进入 ISP 下载模式。当使用第三方 MCU 对芯片进行 ISP 下载时，建议使用下面的电源控制电路来实现。

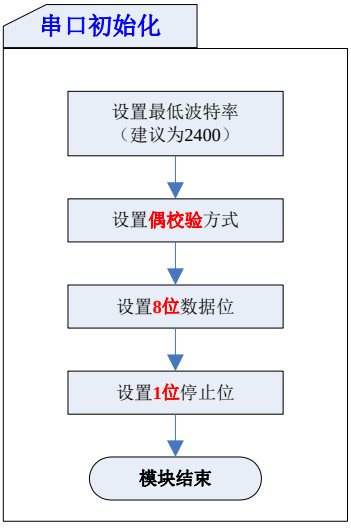


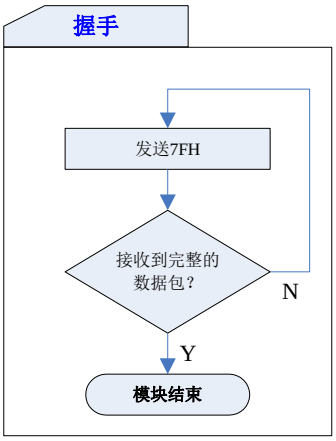
11.2 通信协议流程图

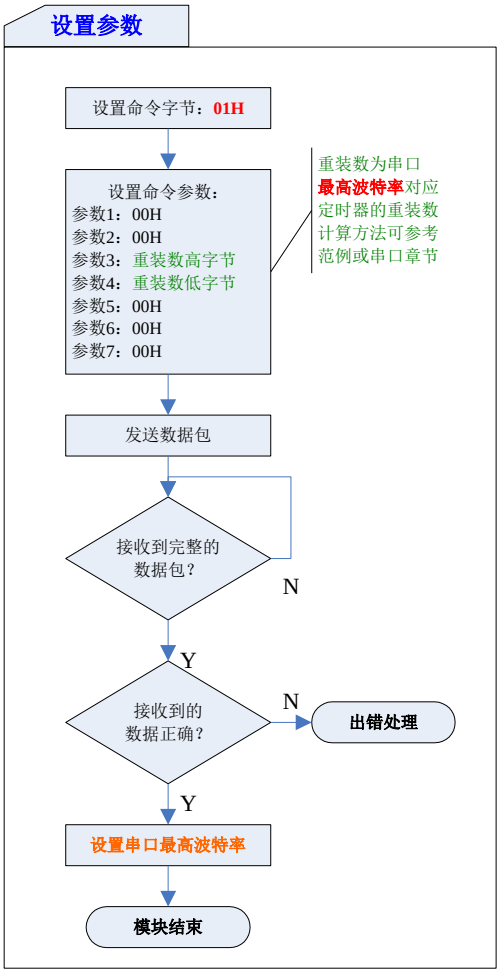

发送数据包
命令头：4字节
[46H B9H 6AH 00H]
数据包长度：1字节
不包含46H和B9H
命令字节：1字节
命令参数
数据包校验和：2字节
不包含46H和B9H
命令结束：1字节
[16H]

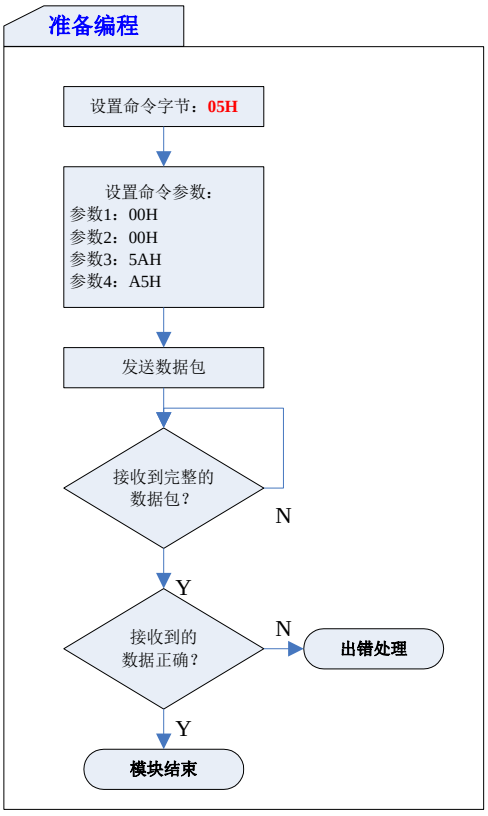

接收数据包
命令头：4字节
[46H B9H 68H 00H]
数据包长度：1字节
不包含46H和B9H
命令字节：1字节
命令参数
数据包校验和：2字节
不包含46H和B9H
命令结束：1字节
[16H]

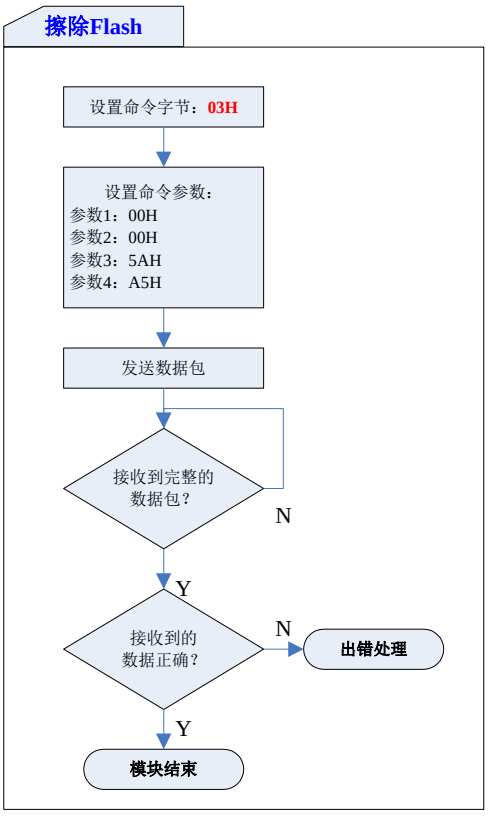

总体流程
开始
串口初始化
握手
设置参数
准备编程
擦除FLASH
写用户代码
结束

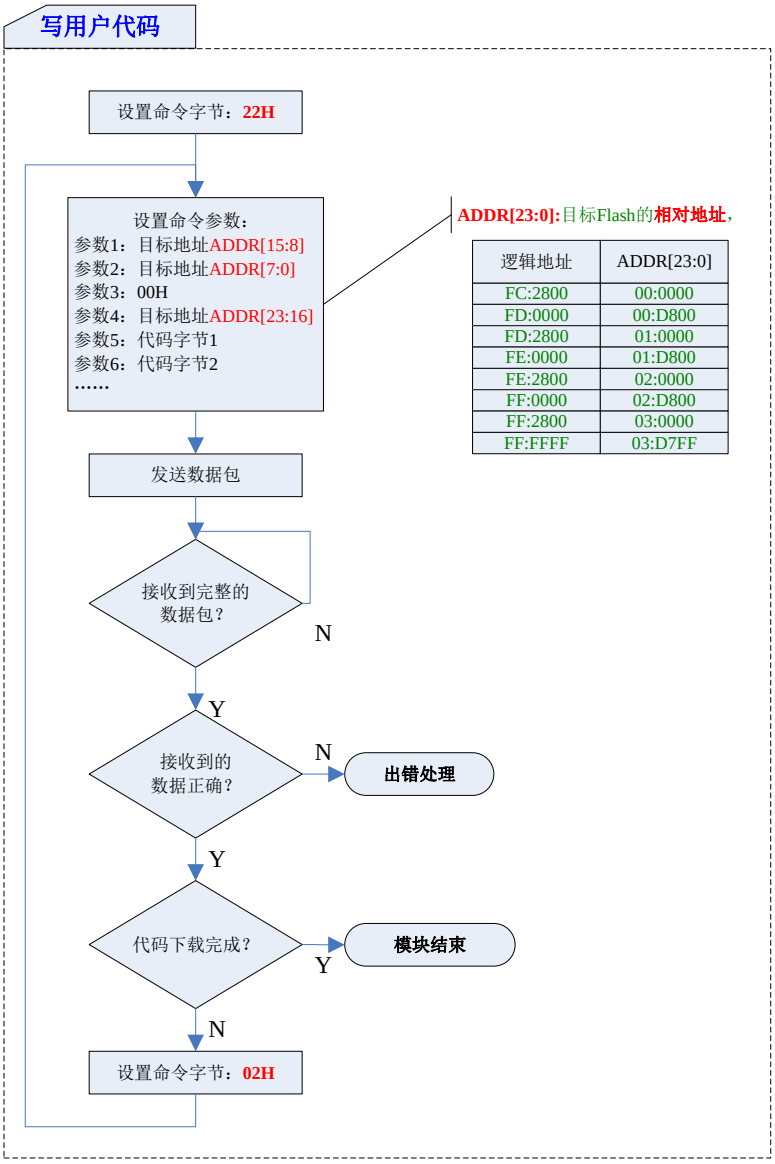

串口初始化
设置最低波特率
（建议为2400）
设置偶校验方式
设置8位数据位
设置1位停止位
模块结束


握手
发送7FH
接收到完整的
数据包？
N
Y
模块结束


设置参数
设置命令字节：01H
设置命令参数：
参数1：00H
参数2：00H
参数3：重装数高字节
参数4：重装数低字节
参数5：00H
参数6：00H
参数7：00H
重装数为串口
最高波特率对应
定时器的重装数
计算方法可参考
范例或串口章节
发送数据包
接收到完整的
数据包？
N
Y
接收到的
数据正确？
N
出错处理
Y
设置串口最高波特率
模块结束


准备编程
设置命令字节：05H
设置命令参数：
参数1：00H
参数2：00H
参数3：5AH
参数4：A5H
发送数据包
接收到完整的
数据包？
N
Y
接收到的
数据正确？
N
出错处理
Y
模块结束


擦除Flash
设置命令字节：03H
设置命令参数：
参数1：00H
参数2：00H
参数3：5AH
参数4：A5H
发送数据包
接收到完整的
数据包？
N
Y
接收到的
数据正确？
N
出错处理
Y
模块结束


写用户代码
设置命令字节：22H
设置命令参数：
参数1：目标地址ADDR[15:8]
参数2：目标地址ADDR[7:0]
参数3：00H
参数4：目标地址ADDR[23:16]
参数5：代码字节1
参数6：代码字节2
……
ADDR[23:0]:目标Flash的相对地址，
逻辑地址 ADDR[23:0]
FC:2800 00:0000
FD:0000 00:D800
FD:2800 01:0000
FE:0000 01:D800
FE:2800 02:0000
FF:0000 02:D800
FF:2800 03:0000
FF:FFFF 03:D7FF
发送数据包
接收到完整的数据包？
N
Y
接收到的数据正确？
N
出错处理
Y
代码下载完成？
Y
模块结束
N
设置命令字节：02H

11.3 参考代码（C 语言）

C 语言代码

```
//注意:使用本代码对 AI8H 系列的单片机进行下载时,必须要执行了 Download 代码之后,
//才能给目标芯片上电,否则目标芯片将无法正确下载

#include "reg51.h"

typedef    bit             BOOL;
typedef    unsigned char   BYTE;
typedef    unsigned short  WORD;

//宏、常量定义
#define    FALSE           0
#define    TRUE            1
#define    LOBYTE(w)       ((BYTE)(WORD)(w))
#define    HIBYTE(w)       ((BYTE)((WORD)(w) >> 8))

#define    MINBAUD         2400L
#define    MAXBAUD         115200L

#define    FOSC            11059200L           //主控芯片工作频率
#define    BR(n)           (65536 - (FOSC/(n)+2) /4)   //主控芯片串口波特率计算公式
                                               //加 2 操作是为了让 Keil 编译器
                                               //自动实现四舍五入运算

#define    T1MS            (65536 - FOSC/1000)         //主控芯片 1ms 定时初值

#define    FUSER           24000000L           //STC32G144K246 系列目标芯片工作频率
#define    RL(n)           (65536 - (FUSER/(n)+2) /4)  //STC32G144K246 系列目标芯片串口波特率计算公式
                                               //加 2 操作是为了让 Keil 编译器
                                               //自动实现四舍五入运算

sfr        AUXR = 0x8e;
sfr        P3M1 = 0xB1;
sfr        P3M0 = 0xB2;

//变量定义
BOOL f1ms;                          //1ms 标志位
BOOL UartBusy;                      //串口发送忙标志位
BOOL UartReceived;                  //串口数据接收完成标志位
BYTE UartRecvStep;                  //串口数据接收控制
BYTE TimeOut;                       //串口通讯超时计数器
BYTE xdata TxBuffer[256];           //串口数据发送缓冲区
BYTE xdata RxBuffer[256];           //串口数据接收缓冲区
char code DEMO[256];                //演示代码数据

//函数声明
void Initial(void);
void DelayXms(WORD x);
BYTE UartSend(BYTE dat);
void CommInit(void);
void CommSend(BYTE size);
BOOL Download(BYTE *pdat, long size);

//主函数入口
void main(void)
```

```
{
    P3M0 = 0x00;
    P3M1 = 0x00;

    Initial();
    if (Download(DEMO, 256))
    {
        //下载成功
        P3 = 0xff;
        DelayXms(500);
        P3 = 0x00;
        DelayXms(500);
        P3 = 0xff;
        DelayXms(500);
        P3 = 0x00;
        DelayXms(500);
        P3 = 0xff;
        DelayXms(500);
        P3 = 0x00;
        DelayXms(500);
        P3 = 0xff;
    }
    else
    {
        //下载失败
        P3 = 0xff;
        DelayXms(500);
        P3 = 0xf3;
        DelayXms(500);
        P3 = 0xff;
        DelayXms(500);
        P3 = 0xf3;
        DelayXms(500);
        P3 = 0xff;
        DelayXms(500);
        P3 = 0xf3;
        DelayXms(500);
        P3 = 0xff;
    }

    while (1);
}

//1ms 定时器中断服务程序
void tm0(void) interrupt 1
{
    static BYTE Counter100;

    f1ms = TRUE;
    if (Counter100-- == 0)
    {
        Counter100 = 100;
        if (TimeOut) TimeOut--;
    }
}

//串口中断服务程序
void uart(void) interrupt 4
{
```

```
    static WORD RecvSum;
    static BYTE RecvIndex;
    static BYTE RecvCount;
    BYTE dat;

    if (TI)
    {
        TI = 0;
        UartBusy = FALSE;
    }

    if (RI)
    {
        RI = 0;
        dat  = SBUF;
        switch (UartRecvStep)
        {
        case 1:
            if (dat != 0xb9) goto L_CheckFirst;
            UartRecvStep++;
            break;
        case 2:
            if (dat != 0x68) goto L_CheckFirst;
            UartRecvStep++;
            break;
        case 3:
            if (dat != 0x00) goto L_CheckFirst;
            UartRecvStep++;
            break;
        case 4:
            RecvSum = 0x68 + dat;
            RecvCount = dat - 6;
            RecvIndex = 0;
            UartRecvStep++;
            break;
        case 5:
            RecvSum += dat;
            RxBuffer[RecvIndex++] = dat;
            if (RecvIndex == RecvCount)         UartRecvStep++;
            break;
        case 6:
            if (dat != HIBYTE(RecvSum))         goto L_CheckFirst;
            UartRecvStep++;
            break;
        case 7:
            if (dat != LOBYTE(RecvSum))         goto L_CheckFirst;
            UartRecvStep++;
            break;
        case 8:
            if (dat != 0x16) goto L_CheckFirst;
            UartReceived = TRUE;
            UartRecvStep++;
            break;
L_CheckFirst:
        case 0:
        default:
            CommInit();
            UartRecvStep = (dat == 0x46         ? 1 : 0);
            break;
```

```
            }
        }
}

//系统初始化
void Initial(void)
{
    UartBusy = FALSE;

    SCON = 0xd0;                    //串口数据模式必须为 8 位数据+1 位偶检验
    AUXR = 0xc0;
    TMOD = 0x00;
    TH0 = HIBYTE(T1MS);
    TL0 = LOBYTE(T1MS);
    TR0 = 1;
    TH1 = HIBYTE(BR(MINBAUD));
    TL1 = LOBYTE(BR(MINBAUD));
    TR1 = 1;
    ET0 = 1;
    ES = 1;
    EA = 1;
}

//Xms 延时程序
void DelayXms(WORD x)
{
    do
    {
        f1ms = FALSE;
        while (!f1ms);
    } while (x--);
}

//串口数据发送程序
BYTE UartSend(BYTE dat)
{
    while (UartBusy);

    UartBusy = TRUE;
    ACC = dat;
    TB8 = P;
    SBUF = ACC;

    return dat;
}

//串口通讯初始化
void CommInit(void)
{
    UartRecvStep = 0;
    TimeOut = 20;
    UartReceived = FALSE;
}

//发送串口通讯数据包
void CommSend(BYTE size)
{
    WORD sum;
    BYTE i;
```

```
        UartSend(0x46);
        UartSend(0xb9);
        UartSend(0x6a);
        UartSend(0x00);
        sum = size + 6 + 0x6a;
        UartSend(size + 6);
        for (i=0; i<size; i++)
        {
            sum  += UartSend(TxBuffer[i]);
        }
        UartSend(HIBYTE(sum));
        UartSend(LOBYTE(sum));
        UartSend(0x16);
        while (UartBusy);

        CommInit();
}

//对 STC32G144K246 系列的芯片进行 ISP 下载程序
BOOL Download(BYTE *pdat, long size)
{
        BYTE offset;
        BYTE cnt;
        DWORD addr;                     //超 64K 代码空间，地址需定义为 4 字节

        //握手
        CommInit();
        while (1)
        {
            if (UartRecvStep == 0)
            {
                UartSend(0x7f);
                DelayXms(10);
            }
            if (UartReceived)
            {
                if (RxBuffer[0] == 0x50) break;
                return FALSE;
            }
        }

        //设置参数(设置从芯片使用最高的波特率)
        TxBuffer[0] = 0x01;
        TxBuffer[1] = 0x00;
        TxBuffer[2] = 0x00;
        TxBuffer[3] = HIBYTE(RL(MAXBAUD));
        TxBuffer[4] = LOBYTE(RL(MAXBAUD));
        TxBuffer[5] = 0x00;
        TxBuffer[6] = 0x00;
        TxBuffer[7] = 0x97;
        CommSend(8);
        while (1)
        {
            if (TimeOut == 0) return FALSE;
            if (UartReceived)
            {
                if (RxBuffer[0] == 0x01) break;
                return FALSE;
```

```
        }
    }

    //准备
    TH1 = HIBYTE(BR(MAXBAUD));
    TL1 = LOBYTE(BR(MAXBAUD));
    DelayXms(10);
    TxBuffer[0] = 0x05;
    TxBuffer[1] = 0x00;
    TxBuffer[2] = 0x00;
    TxBuffer[3] = 0x5a;
    TxBuffer[4] = 0xa5;
    CommSend(5);
    while (1)
    {
        if (TimeOut == 0) return FALSE;
        if (UartReceived)
        {
            if (RxBuffer[0] == 0x05) break;
            return FALSE;
        }
    }

    //擦除
    DelayXms(10);
    TxBuffer[0] = 0x03;
    TxBuffer[1] = 0x00;
    TxBuffer[2] = 0x00;
    TxBuffer[3] = 0x5a;
    TxBuffer[4] = 0xa5;
    CommSend(5);
    TimeOut = 100;
    while (1)
    {
        if (TimeOut == 0) return FALSE;
        if (UartReceived)
        {
            if (RxBuffer[0] == 0x03) break;
            return FALSE;
        }
    }

    //写用户代码
    DelayXms(10);
    addr = 0;
    TxBuffer[0] = 0x22;
    TxBuffer[3] = 0x5a;
    TxBuffer[4] = 0xa5;
    offset = 5;
    while (addr < size)
    {
        if (addr < 0x10000)                 //程序代码的目标地址信息需从原 HEX 文件中获取
        {
            TxBuffer[0] |= 0x10;            //目标编程地址为 FE:0000~FE:FFFF
        }
        else
        {
            TxBuffer[0] &= ~0x10;           //目标编程地址为 FF:0000~FF:FFFF
        }
```

```
            TxBuffer[1] = HIBYTE(addr);
            TxBuffer[2] = LOBYTE(addr);
            cnt    = 0;
            while (addr < size)
            {
                TxBuffer[cnt+offset] = pdat[addr];
                addr++;
                cnt++;
                if (cnt >= 128) break;
            }
            CommSend(cnt + offset);
            while (1)
            {
                if (TimeOut == 0) return FALSE;
                if (UartReceived)
                {
                    if ((RxBuffer[0] == 0x02) && (RxBuffer[1] == 'T')) break;
                    return FALSE;
                }
            }
            TxBuffer[0] = 0x02;
        }

        //下载完成
        return TRUE;
}

char code DEMO[256] =
{
        0x80,0x00,0x75,0xB2,0xFF,0x75,0xB1,0x00,0x05,0xB0,0x11,0x0E,0x80,0xFA,0xD8,0xFE,
        0xD9,0xFC,0x22,
};
```

11.4 如何在用户程序中设置程序运行时的工作频率

11.4.1 在用户程序区用户如何动态选择不同的系统时钟频率

```
//测试工作频率为 11.0592MHz

#include “STC32G144K246.H”                                    //头文件见下载软件
#include "intrins.h"                                          //使用_nop_()函数所必须要包含的头文件,
//否则延时函数中调用的_nop_()函数没有被头文件引用过来，会导致编译器找不到这个而函数而报错。

#define   T22M_ADDR       CHIPID11          //22.1184MHz
#define   T24M_ADDR       CHIPID12          //24MHz
#define   T27M_ADDR       CHIPID13          //27MHz
#define   T30M_ADDR       CHIPID14          //30MHz
#define   T33M_ADDR       CHIPID15          //33.1776MHz
#define   T35M_ADDR       CHIPID16          //35MHz
#define   T36M_ADDR       CHIPID17          //36.864MHz
#define   T40M_ADDR       CHIPID18          //40MHz
#define   T44M_ADDR       CHIPID19          //44.2368MHz
#define   T48M_ADDR       CHIPID20          //48MHz
#define   VRT6M_ADDR      CHIPID21          //VRTRIM_6M
#define   VRT10M_ADDR     CHIPID22          //VRTRIM_10M
#define   VRT27M_ADDR     CHIPID23          //VRTRIM_27M
#define   VRT44M_ADDR     CHIPID24          //VRTRIM_44M

void main()
{
    EAXFR = 1;                                //允许访问扩展的特殊寄存器，XFR
                                              //(32 位模式请使用这句，注释下一句)
//  P_SW2 |= 0x80;                            //允许访问扩展的特殊寄存器，XFR
                                              //(8 位模式请使用这句，注释上一句)
    WTST = 0;                                 //设置取程序代码等待时间，
                                              //赋值为 0 表示不等待，程序以最快速度运行
    CKCON = 0;                                //设置访问片内的 xdata 速度，
                                              //赋值为 0 表示用最快速度访问，不增加额外的等待时间

    P0M0 = 0x00;
    P0M1 = 0x00;
    P1M0 = 0x00;
    P1M1 = 0x00;
    P2M0 = 0x00;
    P2M1 = 0x00;
    P3M0 = 0x00;
    P3M1 = 0x00;
    P4M0 = 0x00;
    P4M1 = 0x00;
    P5M0 = 0x00;
    P5M1 = 0x00;

//  //选择 22.1184MHz
//  CLKDIV = 0x04;
//  IRTRIM = T22M_ADDR;
//  VRTRIM = VRT27M_ADDR;
//  IRCBAND &= ~0x03;
//  IRCBAND |= 0x02;
//  CLKDIV = 0x00;
```

```
    //选择 24MHz
    CLKDIV = 0x04;
    IRTRIM = T24M_ADDR;
    VRTRIM = VRT27M_ADDR;
    IRCBAND &= ~0x03;
    IRCBAND |= 0x02;
    CLKDIV = 0x00;

//  //选择 27MHz
//  CLKDIV = 0x04;
//  IRTRIM = T27M_ADDR;
//  VRTRIM = VRT27M_ADDR;
//  IRCBAND &= ~0x03;
//  IRCBAND |= 0x02;
//  CLKDIV = 0x00;

//  //选择 30MHz
//  CLKDIV = 0x04;
//  IRTRIM = T30M_ADDR;
//  VRTRIM = VRT27M_ADDR;
//  IRCBAND &= ~0x03;
//  IRCBAND |= 0x02;
//  CLKDIV = 0x00;

//  //选择 33.1776MHz
//  CLKDIV = 0x04;
//  IRTRIM = T33M_ADDR;
//  VRTRIM = VRT27M_ADDR;
//  IRCBAND &= ~0x03;
//  IRCBAND |= 0x02;
//  CLKDIV = 0x00;

//  //选择 35MHz
//  CLKDIV = 0x04;
//  IRTRIM = T35M_ADDR;
//  VRTRIM = VRT44M_ADDR;
//  IRCBAND |= 0x03;
//  CLKDIV = 0x00;

//  //选择 44.2368MHz
//  CLKDIV = 0x04;
//  IRTRIM = T44M_ADDR;
//  VRTRIM = VRT44M_ADDR;
//  IRCBAND |= 0x03;
//  CLKDIV = 0x00;

//  //选择 48MHz
//  CLKDIV = 0x04;
//  IRTRIM = T48M_ADDR;
//  VRTRIM = VRT44M_ADDR;
//  IRCBAND |= 0x03;
//  CLKDIV = 0x00;

    while (1);
}
```

11.5 如何在用户程序中设置复位脚、低压检测门槛电压等参数

C 语言代码

```
//测试工作频率为 11.0592MHz

#include “STC32G144K246.H”                    //头文件见下载软件

void main()
{
    EAXFR = 1;                                //允许访问扩展的特殊寄存器，XFR
                                              //(32 位模式请使用这句，注释下一句)
//  P_SW2 |= 0x80;                            //允许访问扩展的特殊寄存器，XFR
                                              //(8 位模式请使用这句，注释上一句)
    WTST = 0;                                 //设置取程序代码等待时间，
                                              //赋值为 0 表示不等待，程序以最快速度运行
    CKCON = 0;                                //设置访问片内的 xdata 速度，
                                              //赋值为 0 表示用最快速度访问，不增加额外的等待时间
    P0M0 = 0x00;
    P0M1 = 0x00;
    P1M0 = 0x00;
    P1M1 = 0x00;
    P2M0 = 0x00;
    P2M1 = 0x00;
    P3M0 = 0x00;
    P3M1 = 0x00;
    P4M0 = 0x00;
    P4M1 = 0x00;
    P5M0 = 0x00;
    P5M1 = 0x00;

    RSTCFG |= 0x10;                           //设置 P5.4 为复位脚
//  RSTCFG &= ~0x10;                          //设置 P5.4 为普通 I/O 口

    RSTCFG |= 0x40;                           //使能低压复位功能
//  RSTCFG &= ~0x40;                          //禁止低压复位

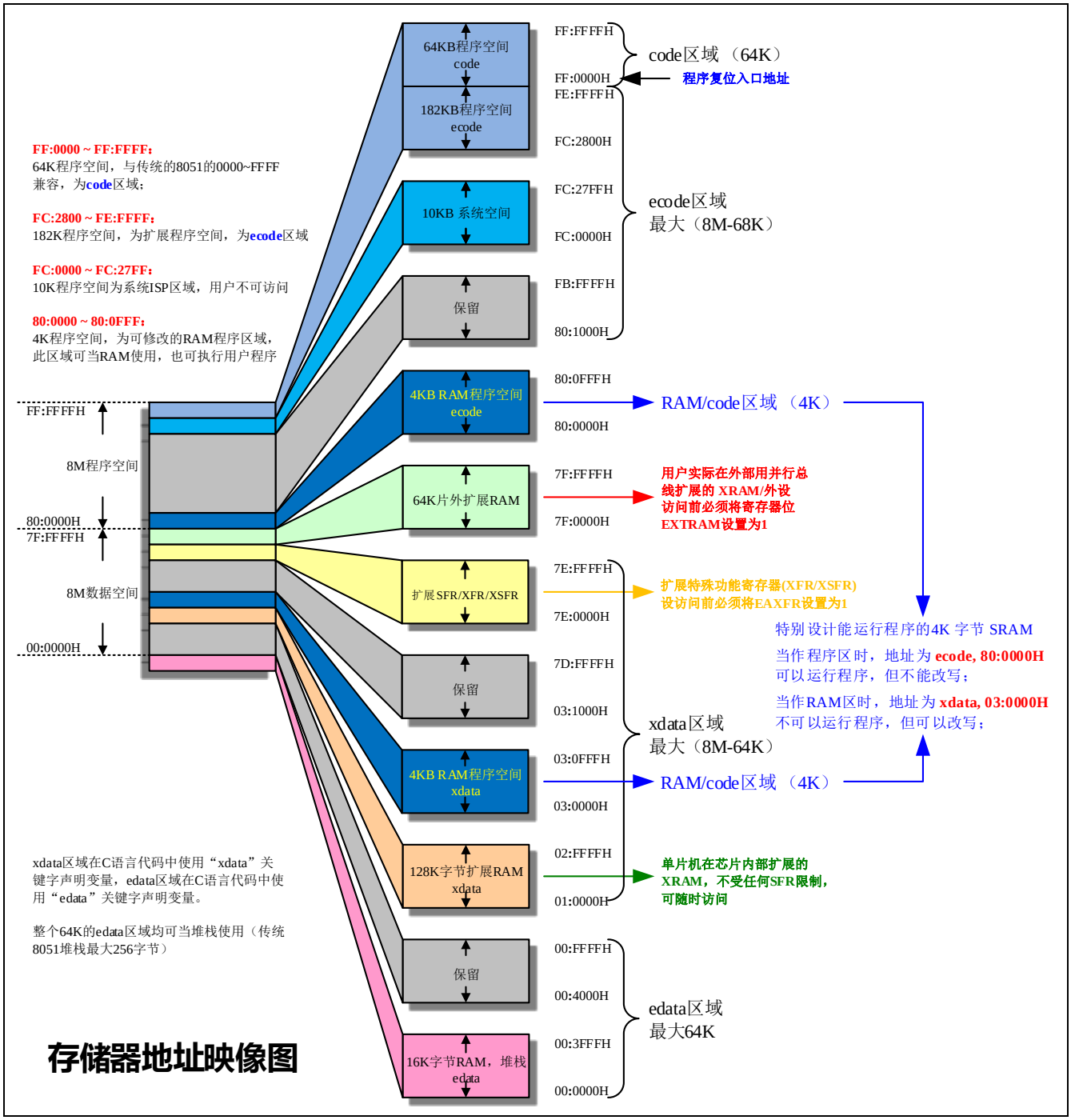
    RSTCFG = (RSTCFG & ~0x03) | 0;            //选择低压复位/检测门槛电压为 2.0V
//  RSTCFG = (RSTCFG & ~0x03) | 1;            //选择低压复位/检测门槛电压为 2.4V
//  RSTCFG = (RSTCFG & ~0x03) | 2;            //选择低压复位/检测门槛电压为 2.7V
//  RSTCFG = (RSTCFG & ~0x03) | 3;            //选择低压复位/检测门槛电压为 3.0V

    while (1);
}
```

12 i-Cache，存储器（32/16/8 位访问），全球唯一 ID 号 CHIPID 储重要参数

STC32G144K246 系列单片机的程序存储器和数据存储器是统一编址的。STC32G144K246 系列单片机提供 24 位寻址空间，最多能够访问 16M 的存储器（8M 数据存储器+8M 程序存储器）。由于没有提供访问外部程序存储器的总线，所以单片机的所有程序存储器都是片上 Flash 存储器，不能访问外部程序存储器。

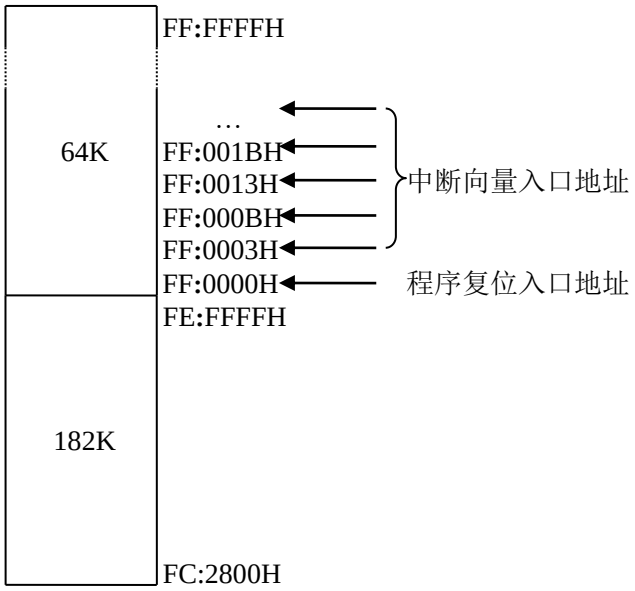
STC32G144K246 系列单片机内部集成了大容量的数据存储器。STC32G144K246 系列单片机内部的数据存储器在物理和逻辑上都分为两个地址空间:内部 RAM（edata）和内部扩展 RAM（xdata）。



12.1 i-Cache 及程序存储器

程序存储器用于存放用户程序、固定不变的数据以及表格等信息。

单片机系列	Flash 程序存储器（ROM）	地址范围
STC32G144K246 系列	246K 字节	FC:2800H~FF:FFFFH



（STC32G144K246 系列与传统 8051 中断入口地址对比）

	STC32G144K246 系列	传统 8051
复位入口地址	FF:0000H	0000H
INT0 中断入口地址	FF:0003H	0003H
TIMER0 中断入口地址	FF:000BH	000BH
INT1 中断入口地址	FF:0013H	0013H
TIMER1 中断入口地址	FF:001BH	001BH
UART 中断入口地址	FF:0023H	0023H

单片机复位后，程序计数器(PC)的内容为 FF:0000H，从 FF:0000H 单元开始执行程序。另外中断服务程序的入口地址(又称中断向量)也位于程序存储器单元。在程序存储器中，每个中断都有一个固定的入口地址，当中断发生并得到响应后，单片机就会自动跳转到相应的中断入口地址去执行程序。外部中断 0（INT0）的中断服务程序的入口地址是 FF:0003H，定时器/计数器 0（TIMER0）中断服务程序的入口地址是 FF:000BH，外部中断 1（INT1）的中断服务程序的入口地址是 FF:0013H，定时器/计数器 1（TIMER1）的中断服务程序的入口地址是 FF:001BH 等。更多的中断服务程序的入口地址(中断向量)请参考中断介绍章节。

由于相邻中断入口地址的间隔区间仅仅有 8 个字节，一般情况下无法保存完整的中断服务程序，因此在中断响应的地址区域存放一条无条件转移指令，指向真正存放中断服务程序的空间去执行。

STC32G144K246 系列单片机中都包含有 Flash 数据存储器（EEPROM）。以字节为单位进行读/写数据，以 512 字节为页单位进行擦除，可在线反复编程擦写 10 万次以上，提高了使用的灵活性和方便性。

12.1.1 程序读取等待控制寄存器（WTST）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
WTST	E9H	WTST[7:0]							

WTST[7:0]：CPU 读取程序存储器的等待时间控制

WTST[7:0]	等待时钟数	设置建议值
0	0 个时钟	系统时钟 <= 32MHz
1	1 个时钟	32MHz < 系统时钟 <= 64MHz
2	2 个时钟	64MHz < 系统时钟 <= 90MHz
3	3 个时钟	90MHz < 系统时钟 <= 120MHz
4	4 个时钟	系统时钟 > 120MHz
…	…	
7	7 个时钟	
8	保留	
…	保留	
255	保留	

如 CPU 需要执行的指令已被缓存在 i-Cache 中，则此指令的执行时钟数等于指令表中所列的时钟数；当所需的指令不在 i-Cache 中时，则 CPU 需要等待 i-Cache 缓存从程序存取器中预取后才能继续运行。

STC32G144K246 系列单片机采用了先进的设计制程，使系统运行的工作频率可在 64MHz ~ 120MHz 之间，内部 Flash 程序存储器的最高工作频率大致在 33MHz ~ 40MHz 左右，CPU 的执行速度相对较快，Flash 程序存储器的速度相对较慢，这就导致了所谓的“存储墙”限制运行速度的问题。

为了解决这个问题，STC32G144K246 系列单片机内部设计了 i-Cache（程序指令高速缓存）功能。i-Cache 功能是将慢速 Flash 程序存储器中的程序代码加载到 CPU 内部的 i-Cache 高速缓存中，CPU 执行指令时直接从 i-Cache 中进行取指，当出现 CPU 需要运行的指令不在 i-Cache 中时，就会触发 i-Cache 模块重新从程序 Flash 的目标地址中预装载 CPU 所需要的指令，指令缓存结束后，CPU 继续从 i-Cache 中取指运行程序。由于 i-Cache 高速缓存的出现，CPU 的速度再也不受低速 Flash 程序存储器的限制。

STC32G144K246 系列单片机还设计了预取指模块，来提高 CPU / i-Cache 的运行效率。

解决了 CPU 运行速度问题后，高速 i-Cache 与慢速 Flash 程序存储器之间的速度匹配问题是通过 Flash 读取等待控制寄存器 WTST 来解决的。WTST 寄存器的功能是设置从 Flash 中读取指令所需要等待的系统时钟数。由于 Flash 的最高工作频率为 33MHz ~ 40MHz 左右，所以当系统工作频率低于 33MHz 时，WTST 可设置为 0；当系统工作频率为 33MHz ~ 64MHz 之间时，WTST 可设置为 1 ~ 2；当系统工作频率为 64MHz ~ 90MHz 之间时，WTST 可设置为 2 ~ 3；当系统工作频率为 90MHz ~ 120MHz 之间时，WTST 可设置为 3 ~ 4。

【特别注意】：当使能了 i-Cache 功能后，WTST 必须设置为大于等于 1 的值，不能设置为 0，即使系统工作频率低于 33MHz。由于 i-Cache 功能在芯片出厂时就已经默认使能，无法通过软件或硬件设置关闭，所以 WTST 上电后的初始值为 1，且用户也不能将 WTST 设置为 0（软件写 0 无效，硬件加了防错保护）。

12.2 数据存储器（32 位访问，16 位访问，8 位访问）

STC32G144K246 系列单片机内部集成的 RAM 可用于存放程序执行的中间结果和过程数据。

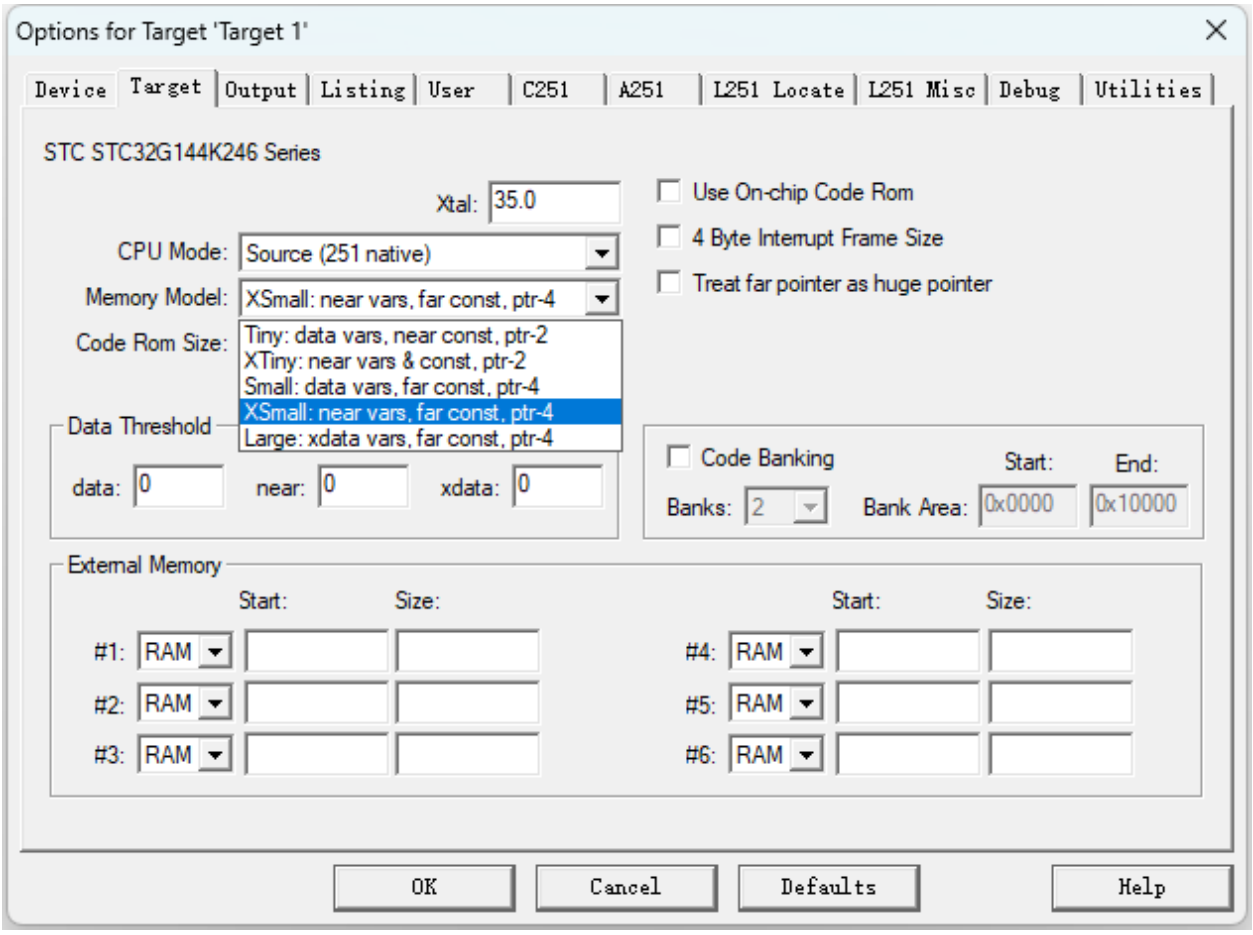
单片机系列	内部 RAM （edata）	内部扩展 RAM （xdata）
STC32G144K246 系列	16K 字节	128K 字节

EDATA：单时钟存取，访问速度快，可做堆栈使用，成本高；

XDATA：存取需要 2~3 个时钟，访问速度慢，寻址空间可达 8M，成本低。

12.2.1 Keil 选项 Memory Model 设置

对于 STC32G144K246 系列的项目的存储器模式，在 Keil 环境下有如下图所示的 5 种模式：



各种模式对比如下表：

Memory Model	默认变量类型（数据存储器）	默认常量类型（程序存储器）	默认指针变量	指针访问范围
Tiny 模式	data	near	2 字节	00:0000 ~ 00:FFFF
XTiny 模式	edata	near	2 字节	00:0000 ~ 00:FFFF
Small 模式	data	far	4 字节	00:0000 ~ FF:FFFF
XSmall 模式	**edata**	**far**	**4 字节**	**00:0000 ~ FF:FFFF**
Large 模式	xdata	far	4 字节	00:0000 ~ FF:FFFF

由于 STC32G144K246 的程序逻辑地址为 FC:2800H~FF:FFFFH，需要使用 24 位地址线才能正确访问，默认的常量类型（程序存储器类型）必须使用“far”类型，默认指针变量必须为 4 字节。

不建议使用“Small”“Tiny”和“XTiny”模式，推荐使用“XSmall”模式，这种模式默认将变量定义在内部 RAM(edata)，单时钟存取，访问速度快，且 STC32G144K246 系列芯片有 16K 的 edata 可以使用；使用“Small”模式时，默认将变量定义在内部 RAM(data)，data 默认只有 128 字节，当用户对 RAM 需求超过 128 字节时，Keil 编译器会报错，data 区数量有限，容易报错，所以不建议使用；不推荐使用“Large”模式，虽然该模式也能正确访问 STC32G144K246 的全部 16M 寻址空间，但“Large”模式默认将变量定义在内部扩展 RAM(xdata)里面，存取需要 2~3 个时钟，访问速度慢

STC32G144K246 系列单片机的 C 语言变量声明建议：

3、 当用户变量需求量较小时，建议不要使用“edata、xdata”等关键字声明变量，而使用如下方式直接声明变量：

```
char    bCounter = 1;       //声明字节变量
int     wCounter = 100;     //声明双字节变量
long    dwCounter = 0x1234; //声明 4 字节变量
```

然后在项目选项中将“Memory Model”设置为“XSmall”，让编译器自动将声明的变量分配到 edata 区域

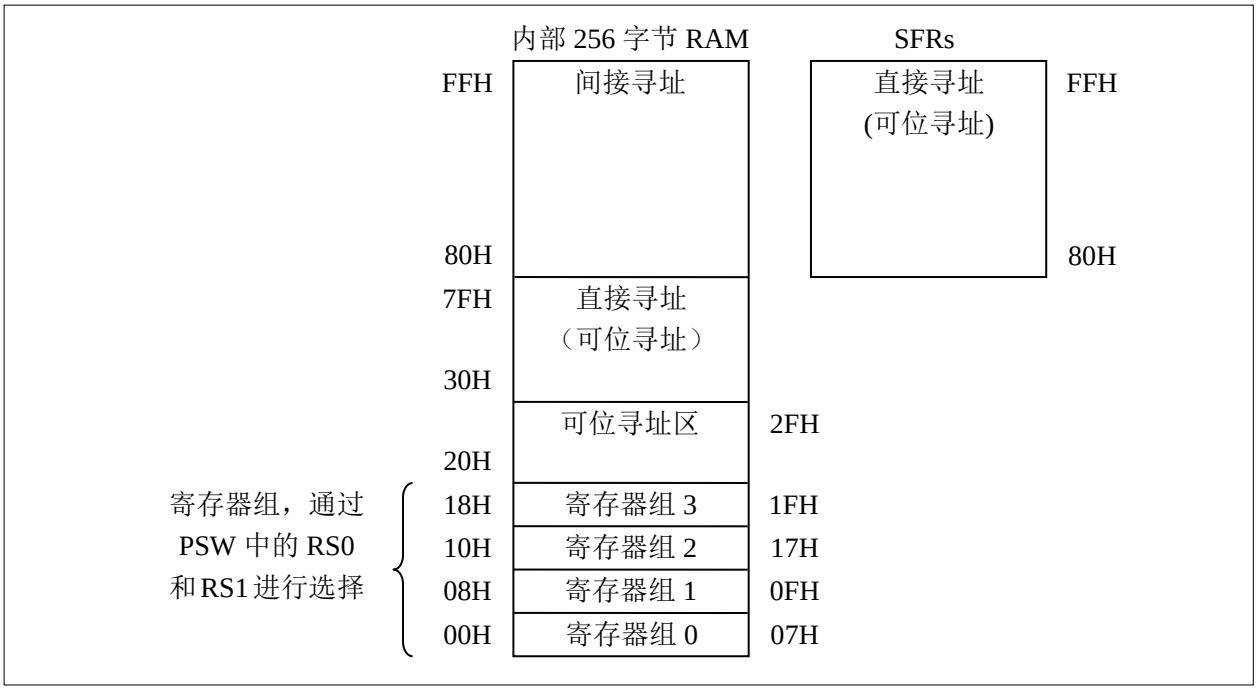
4、 当用户变量需求量接近或超过（单片机 edata 容量-1K）的大小时，建议将超出部分使用“xdata”关键字强制分配到 xdata 区域，如下所示：

```
int  xdata  pBuffer = 5;        //使用 xdata 关键字强制分配到 xdata 区域
```

12.2.2 内部 edata-RAM（C 语言声明关键字为 edata）

STC32G144K246 系列内部可 edata-RAM 共 16K 字节，16K 字节低端的 256 字节与 8051 的 256 字节 DATA 完全兼容，可分为 2 个部分：低 128 字节 RAM 和高 128 字节 RAM。低 128 字节的数据存储器与传统 8051 兼容，既可直接寻址也可间接寻址。高 128 字节 RAM（在 8052 中扩展了高 128 字节 RAM）只能间接寻址。特殊功能寄存器分布在 80H~FFH 区域，只可直接寻址。

低端 256 字节 RAM 的结构如下图所示：



STC32G144K246 的堆栈放在 EDATA，设计上理论深度可达 64K，实际放了 16K Bytes；STC32G144K246 的普通扩展 XDATA，设计上理论深度可达（8M-64K），实际放了 128K Bytes。所以 STC32G144K246 的 SRAM 总共是 144K（16K edata + 128K xdata）。

在 C 语言代码中将变量声明在 EDATA 区域，即可实现单时钟进行 32 位/16 位/8 位的读写操作

```
char    edata   bCounter;      //在 EDATA 区域声明字节变量（单时钟进行 8 位读写操作）
int     edata   wCounter;      //在 EDATA 区域声明双字节变量（单时钟进行 16 位读写操作）
long    edata   dwCounter;     //在 EDATA 区域声明 4 字节变量（单时钟进行 32 位读写操作）
```

在 C 语言代码中将变量声明在 XDATA 区域，即可实现 8 位/16 位的读写操作

```
char    xdata   bCounter;      //在 XDATA 区域声明字节变量（3/2 个时钟进行 8 位读/写操作）
int     xdata   wCounter;      //在 XDATA 区域声明双字节变量（3/2 个时钟进行 16 位读/写操作）
```

12.2.3 程序状态寄存器（PSW）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PSW	D0H	CY	AC	F0	**RS1**	**RS0**	OV	-	P

CY：进位标志

AC：辅助进位标志

F0：通用标志

OV：溢出标志

P：ACC 的偶校验标志

RS1，RS0：工作寄存器选择位

RS1	RS0	工作寄存器组（R0~R7）
0	0	第 0 组（00H~07H）
0	1	第 1 组（08H~0FH）
1	0	第 2 组（10H~17H）
1	1	第 3 组（18H~1FH）

可位寻址区的地址从 20H ~ 2FH 共 16 个字节单元。20H~2FH 单元既可像普通 RAM 单元一样按字节存取，也可以对单元中的任何一位单独存取，共 128 位，所对应的逻辑位地址范围是 00H~7FH。位地址范围是 00H~7FH，内部 RAM 低 128 字节的地址也是 00H~7FH，从外表看，二者地址是一样的，实际上二者具有本质的区别；位地址指向的是一个位，而字节地址指向的是一个字节单元，在程序中使用不同的指令区分。

12.2.4 程序状态寄存器 1（PSW1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PSW1	D1H	CY	AC	N	RS1	RS0	OV	Z	-

CY：进位标志（与 PSW 中的 CY 相同）

AC：辅助进位标志（与 PSW 中的 AC 相同）

N：计算结果负标志位，运算结果的最高位为 1 时 N 为 1，否则为 0

RS1，RS0：工作寄存器选择位（与 PSW 中的 RS0、RS1 相同）

OV：溢出标志（与 PSW 中的 OV 相同）

Z：计算结果零标志位，运算结果为 0 时 Z 为 1，否则为 0

12.2.5 内部 xdata-RAM（C 语言声明关键字为 xdata）

STC32G144K246 系列单片机片内部扩展了 XRAM，不是用户实际在片外扩展的 XRAM）。

访问内部扩展的 XRAM 的方法和传统 8051 单片机访问外部扩展 RAM 的方法相同，但是不影响 P0 口、P2 口、以及 RD、WR 和 ALE 等端口上的信号。

在汇编语言中，内部扩展 RAM 通过 MOVX 指令访问，

```
MOVX    A , @DPTR
MOVX    @DPTR , A
MOVX    A , @Ri
MOVX    @Ri , A
```

在 C 语言中，可使用 xdata 关键字声明存储类型即可。**（强烈建议不要使用 pdata 关键字声明变量）**

12.2.6 辅助寄存器（AUXR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
AUXR	8EH	T0x12	T1x12	UART_M0x6	T2R	T2_C/T	T2x12	**EXTRAM**	S1BRT

EXTRAM：片外用户实际扩展的 XRAM 访问控制

0：可以访问内部扩展的 XRAM，不能访问用户实际在外部用并行总线扩展的 XRAM 或外设，这部分用户可以实际在外部并行扩展的 XRAM 或外设地址范围在：特殊的地址 7F:0000H~7F:FFFFH

1：禁止访问内部扩展的 XRAM，可以访问用户实际在外部用并行总线扩展的 XRAM 或外设，这部分用户可以实际在外部并行扩展的 XRAM 或外设地址范围在：特殊的地址 7F:0000H~7F:FFFFH。当 EXTRAM 设置为 1 后，必须目标地址在 7F:0000H - 7F:FFFFH 范围之内，P0/P2/ALE/WR/RD 才会送出控制信号。

12.2.7 片外用户自己扩展的外部 XRAM/xdata

STC32G144K246 系列单片机具有片外扩展 64KB 外部数据存储器 XRAM 或外设的能力（用户实际在外部用并行总线扩展的 XRAM 或外设），映射到内部的逻辑地址范围为 7F:0000H~7F:FFFFH。访问片外的外部数据存储器或外设期间，P0/P2/WR/RD/ALE 信号才有效。STC32G144K246 系列单片机控制片外扩展的外部 64K 字节数据存储器或外设的总线速度的特殊功能寄存器 BUS_SPEED 说明见下一小节。

12.2.8 片外用户扩展 RAM 的总线速度控制寄存器（BUS_SPEED）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
BUS_SPEED	A1H	RW_S[1:0]					SPEED[2:0]		

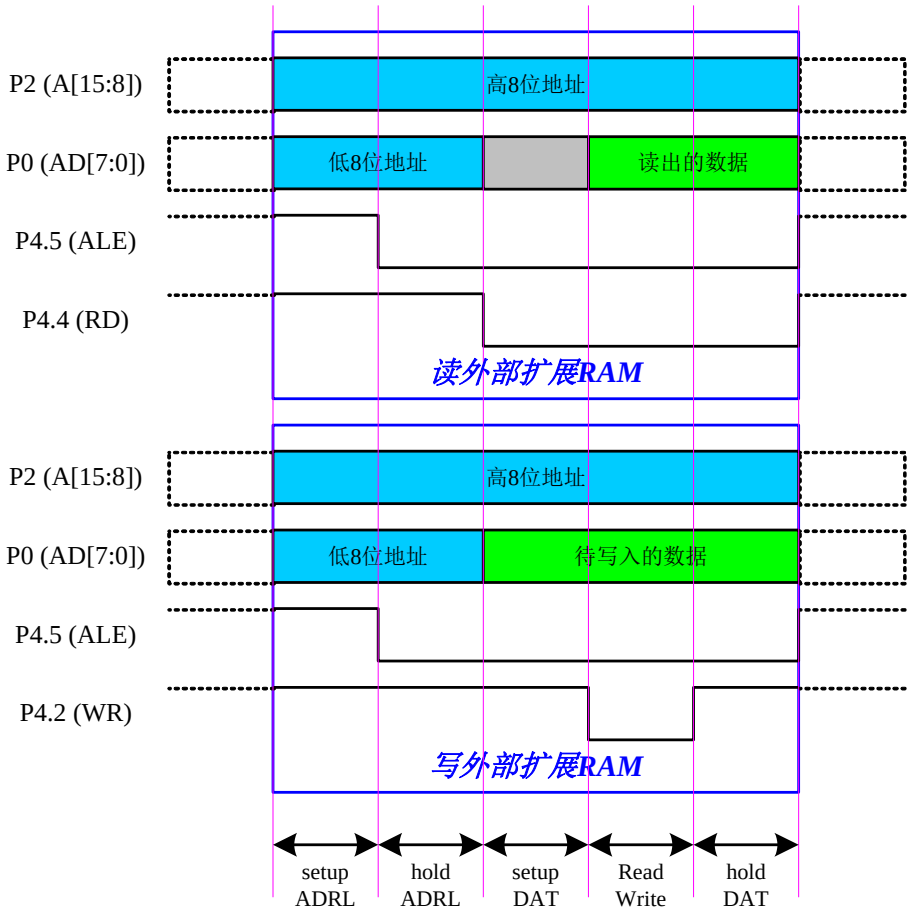
RW_S[1:0]：RD/WR 控制线选择位

00：P4.4 为 RD，P4.2 为 WR

x1：保留

SPEED[2:0]：总线读写速度控制（读写数据时控制信号和数据信号的准备时间和保持时间）

读写外部扩展 RAM 时序如下图所示：



12.2.9 片内 MCU 扩展 RAM 数据总线时钟控制寄存器（CKCON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CKCON	EAH	RAMEXE	-	-	T1M	T0M	CKCON[2:0]		

RAMEXE：80:0000H ~ 80:0FFFH 区域 4K 可执行 RAM 控制位

0：此区域为数据区域，可读、可写、不可运行代码

1：此区域为程序区域，可读、可运行代码、不可写

T1M：保留

T0M：保留

CKCON[2:0]：外部数据总线时钟控制寄存器（上电复位值为 7，强烈建议上电初始化为 0）

CKCON[2:0]	等待时钟数
0	0 个时钟
1	1 个时钟
…	…
7	7 个时钟

12.2.10 片外扩展 RAM 地址总线扩展寄存器（MXAX）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
MXAX	EBH								

使用汇编指令 MOVX A,@Ri 和 MOVX @Ri,A 指令访问 XRAM 时，XRAM[23:16]地址由 MXAX 寄存器设置，XRAM[15:8]地址由 P2 口（P2 寄存器）设置，XRAM[7:0]地址由 Ri（R0/R1）设置。

只有在使用汇编指令 MOVX A,@Ri 和 MOVX @Ri,A 指令访问 XRAM，或者在 C 语言程序中使用 pdata 变量访问 XRAM 时，才需要使用 MXAX 寄存器（注：强烈不建议使用 pdata 定义变量）

12.2.11 STC32G144K246 系列单片机中可位寻址的数据存储器

STC32G144K246 系列单片机内部可位寻址的数据存储器包括两部分：第一部分的地址范围为 DATA 区域的 20H~7FH，第二部分的地址范围是特殊功能寄存器 SFR：80H~FFH，下面对这两部分区域分别进行说明。

DATA 区域

DATA 区域的 00H~1FH 为寄存器 R0~R7 的映射区，不可位寻址，剩下 20H~7FH，共 96 个字节，每个字节均可位寻址。

汇编代码的定义方法：

```
BVAR1       BIT     20H.0           ;定义位变量 BVAR1
BVAR2       BIT     50H.1           ;定义位变量 BVAR2
BVAR3       BIT     70H.2           ;定义位变量 BVAR3
```

汇编代码的使用方法：

```
SETB        BVAR1                   ;BVAR1 = 1
CLR         BVAR2                   ;BVAR2 = 0
CPL         BVAR3                   ;BVAR3 = ~BVAR3
```

C 语言代码的定义方法：

```
char  ebdata      flag;             //在可位寻址区域定义一个字节变量
sbit  bVar1   =   flag^0;           //使用 sbit 在 flag 中声明位变量 bVar1
sbit  bVar2   =   flag^7;           //使用 sbit 在 flag 中声明位变量 bVar2
bit   ebdata      bVar3;            //使用 bit ebdata 直接声明位变量 bVar3
```

C 语言代码的使用方法：

```
bVar1 = 1;                          //位变量置 1
bVar2 = 0;                          //位变量清 0
bVar3 = ~bVar3                      //位变量取反
```

特殊功能寄存器（SFR）区域

全部 SFR 区域的 80H~FFH，共 128 个字节，每个字节均可位寻址。

汇编代码的定义方法：

```
BVAR1       BIT     80H.0           ;定义位变量 BVAR1
BVAR2       BIT     87H.1           ;定义位变量 BVAR2
BVAR3       BIT     0FFH.2          ;定义位变量 BVAR3
```

汇编代码的使用方法：

```
SETB        BVAR1                   ;BVAR1 = 1
CLR         BVAR2                   ;BVAR2 = 0
CPL         BVAR3                   ;BVAR3 = ~BVAR3
```

C 语言代码的定义方法：

```
sfr   P0      =   0x80;             //定义 SFR
sbit  P00     =   P0^0;             //使用 sbit 在 SFR 中声明 SFR 位
sfr   PCON    =   0x87;             //定义 SFR
sbit  PD      =   PCON^1;           //使用 sbit 在 SFR 中声明 SFR 位
sfr   ACC     =   0xE0;             //定义 SFR
sbit  ACC7    =   ACC^7;            //使用 sbit 在 SFR 中声明 SFR 位
```

C 语言代码的使用方法：

```
PD = 0;                             //位变量置 1
P00 = 1;                            //位变量清 0
```

```
ACC7 = ~ACC7                                    //位变量取反
```

注意：位变量不支持数组和指针类型的定义

12.2.12 扩展 SFR 使能寄存器 EAXFR 的使用说明

STC32G144K246 系列单片机存储器地址的编址如下：

区域	地址范围	功能	说明
数据区	00:0000H — 00:FFFFH	数据区（edata）	堆栈区
	01:0000H — 02:FFFFH	内部扩展数据区（xdata）	
	03:0000H — 7D:FFFFH	数据保留区	
	7E:0000H — 7E:FFFFH	扩展 SFR 区（XFR）	需使能 EAXFR（P_SW2.7）才能访问
	7F:0000H — 7F:FFFFH	外部数据区（xdata）	需使能 EXTRAM（AUXR.1）才能访问
代码区	80:0000H — 80:0FFFH	RAM 代码区	需使能 RAMEXE（CKCON.7）才能运行代码
	80:1000H — FB:FFFFH	代码保留区	
	FC:0000H — FC:27FFH	系统代码区	
	FC:2800H — FE:FFFFH	扩展代码区（ecode）	
	FF:0000H — FF:FFFFH	代码区（code）	

如需访问 XFR 区域的扩展 SFR，需要先将 EAXFR（P_SW2.7）置 1，由于 STC32G144G246 的存储器地址的编址方式是线性编址的，XFR 区域的地址是独立的一块地址区域，所以可以在**上电系统初始化时将 EAXFR 寄存器写 1（例如：EAXFR = 1;），后续一直保持为 1 不用再修改**，即可正常访问 XFR 区域。

注意：由于 EAXFR 控制位和 S2_S、S3_S 等位寄存器位共用 P_SW2，在设置 P_SW2 中的 S2_S、S3_S 时，建议直接使用位操作语句（例如 S2_S = 0; S3_S = 1;），避免直接操作 P_SW2 寄存器。如代码中确实需要直接对 P_SW2 进行修改，也一定要使用与或指令（例如 P_SW2 &= ~0x01; P_SW2 |= 0x02;）。

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW2	BAH	EAXFR	-	I2C_S[1:0]		-	S4_S	S3_S	S2_S

EAXFR：扩展 RAM 区特殊功能寄存器（XFR）访问控制寄存器

0：禁止访问 XFR

1：使能访问 XFR。

当需要访问 XFR 时，必须先将 EAXFR 置 1，才能对 XFR 进行正常的读写。建议上电初始化时直接设置为 1，后续不要再修改

12.3 RAM 中运行程序@120MHz

12.3.1 使用说明

51，RAM 中运行程序@120MHz，STC32G144K246（从 QSPI 直接运行程序开发中）
本例程使用内部 24MHz 高速时钟，经过 HPLL1 倍频到 120MHz 后，给系统时钟使用。

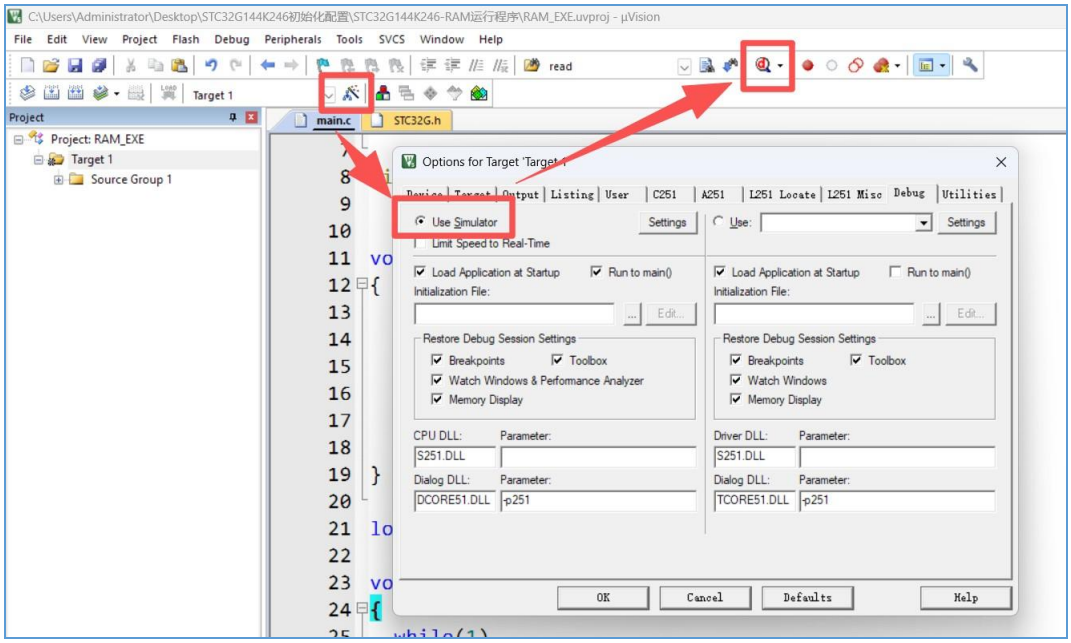
另有实际测试：同样的一段计算函数（1000 次整数乘法和加法），计算时关闭总中断，主程序 120MHz，消耗了 200us，而在 RAM 内运行消耗了 124us（此时 WTST 设置失效，零等待执行），速度提高了 38%

程序拥有两种方式来载入 RAM 程序，一种是同一个工程中，通过对函数取地址来获取程序内容。这种方法在调用其他函数的时候，本质还是会跳转回到 FLASH 部分运行。
第二种是使用独立的工程来编译 RAM 中需要运行的程序，这部分通过设置程序整体偏移地址为 0x800000 实现，同时通过 hex2bin.exe 实现自动转为 bin 文件，这个编译后的 bin 文件通过“打开 EEPROM 文件”载入即可使用。

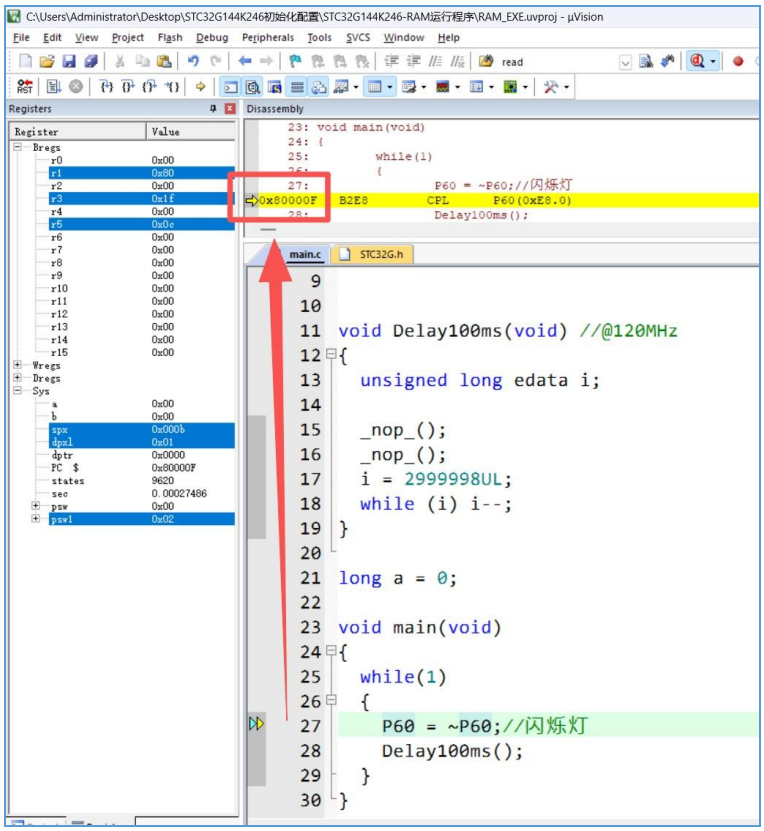
使用同一工程的方式，效果是 P32 和 P35 闪烁，长按 P32 按键后，P32、P35 停止闪烁。这部分程序是主程序，非 RAM 内运行。如果按下 P33 按键，则跳转到 RAM 内运行程序，效果是 P6 上持续自减数值，也就是二进制不断刷新显示。

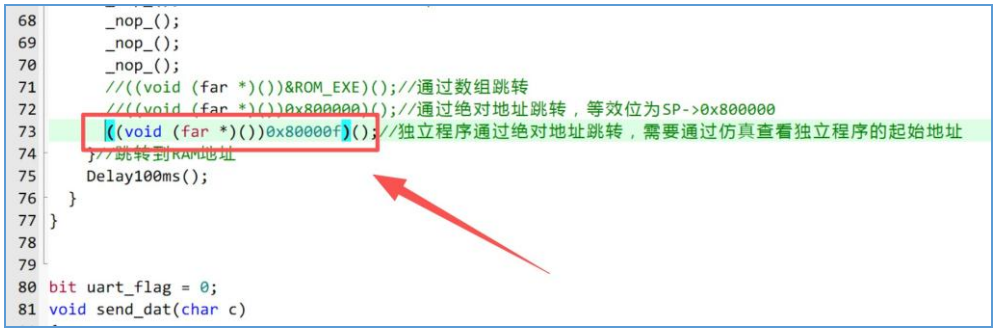
```
58  for (i = 0; i < 4096; i++)RAM_EXE[i] = ((char far *)&exeram_code)[i]; //同一程序下的RAM程序运行
59  //for (i = 0; i < 4096; i++)RAM_EXE[i] = ((char far *)&CODE_RAM_FLASH)[i]; //独立程序下的RAM程序运行
60  while(1)
61  {
62    //用户程序
63    for (i = 0; i < 20; i++)uart_send(sprintf(uart_buff, "%02bx ", RAM_EXE[i]));//串口打印RAM内容的前20位
64    uart_send(sprintf(uart_buff, "\n"));
65    if(P33==0){
66      CKCON |= 0x80;          //切换到ROM运行方式
67      _nop_();                //add 2 nops here at least
68      _nop_();
69      _nop_();
70      _nop_();
71      ////(void (far *)())&ROM_EXE)();//通过数组跳转
72      ((void (far *)())0x800000)();//通过绝对地址跳转，等效位为SP->0x800000
73      ////((void (far *)())0x80000f)();//独立程序通过绝对地址跳转，需要通过仿真查看独立程序的起始地址
74    }//跳转到RAM地址
75    Delay100ms();
76  }
```

需要注意的是，使用不同工程的情况下，需要手动设定 ECALL 跳转地址。具体操作为打开独立的 RAM_EXE 工程：选择模拟仿真后点击仿真按钮



然后查看 main 函数内第一行的语句所在地址：这里是 0x80:000f，将这个地址回填到主程序内的绝对地址跳转调用上即可





【注意事项】在 CKCON 的最高位 bit7 中 RAMEXE 中

9.2.9 片内 MCU 扩展 RAM 数据总线时钟控制寄存器（CKCON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CKCON	EAH	RAMEXE	-	-	T1M	T0M	CKCON[2:0]		

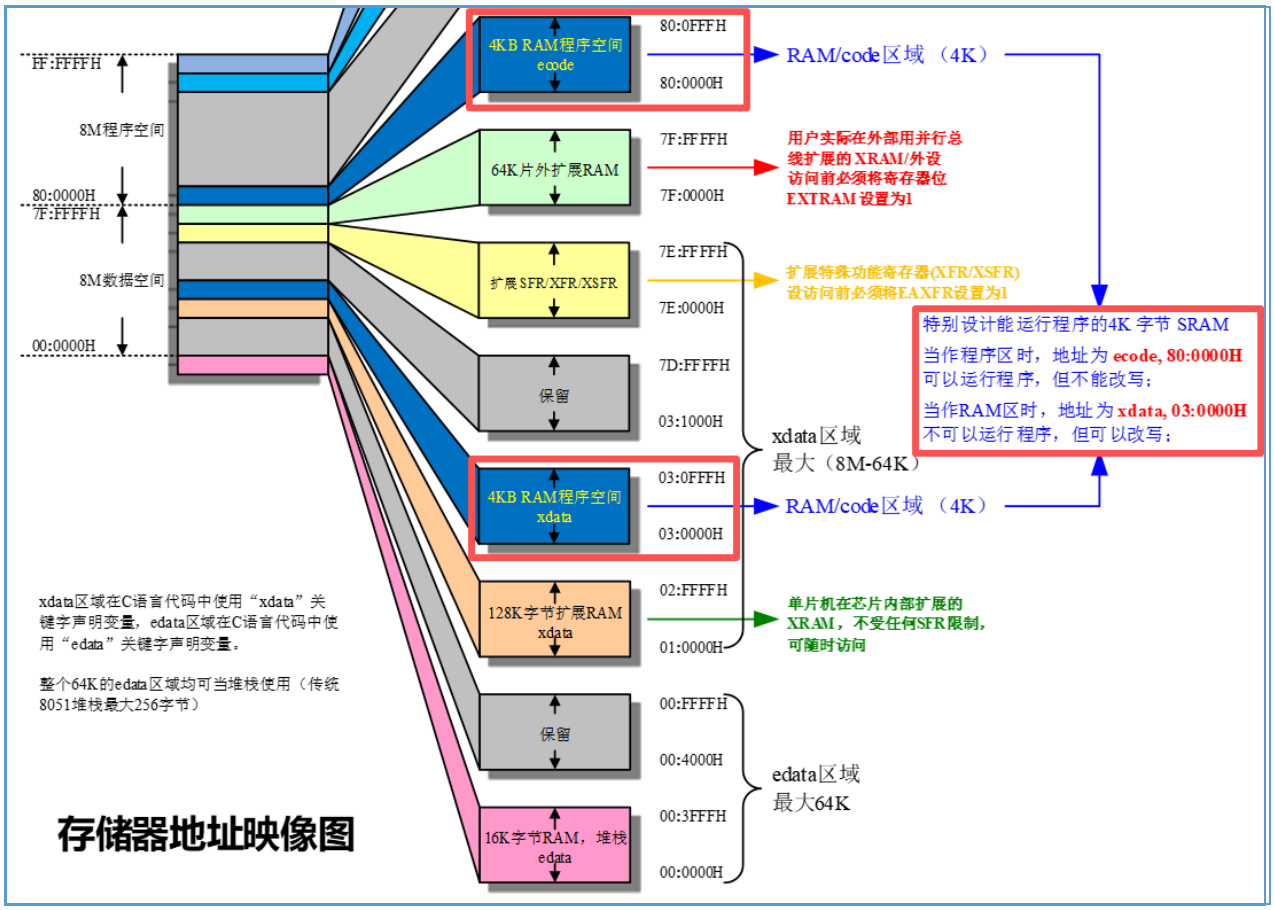
RAMEXE：80:0000H ~ 80:0FFFH 区域 4K 可执行 RAM 控制位

0：此区域为数据区域，可读、可写、不可运行代码

1：此区域为程序区域，可读、可运行代码、不可写

如果 RAMEXE 设置为 0 时，可以通过 03:0000H~03:0FFFH 的 far 地址读写，此时可以认为是拓展的 xdata 地址，这个时候不可以读 80:0000H~80:0FFFH，否则会造成错误，进而导致卡死（因为 80:0000H~80:0FFFH 属于 ecode，本来也不让写）。

如果 RAMEXE 设置为 1 时，可以通过 80:0000H~80:0FFFH 地址来读内容，这个时候写 03:0000H~03:0FFFH，写入内容不会生效，读取内容也和实际内容不符（作为保护程序的一个措施，如果需要更改程序，需要将 RAMEXE 重新设置为 0）



12.3.2 简单科普

far 类型是用来定义变量/常量的，不同于 xdata 这种自带偏移的地址（char xdata a _at_ 0;的实际地址是 01:0000H），far 类型类似一种绝对地址，是从 00:0000H 开始的，覆盖 edata 和 xdata 部分。

以下为 keil 帮助文档的解释：

far：远存类型可用于变量和常量。此内存使用 24 位地址访问，可以是片上或外部的。

对于变量，远存内存限制为 16M。对象限制为 64K，并且不能跨越 64K 边界(链接器确保这一点)。声明为 far 的变量位于 HDATA 内存类别中。

对于常量(ROM 变量)，远区内存被限制为 16M。对象被限制为 64K，且不能跨越 64K 的边界(链接器会确保这一点)。声明为远区的常量变量位于 HCONST 组中。

12.3.3 范例程序

```
#include "STC32G144K246.H"
#include "stdio.h"
#include "stdarg.h"
#include "string.h"

//本例程使用 CHIPID 内预置参数，设置 HIRC 为 24MHz
//使用 HPLL1，提供 60Mhz，80Mhz，120Mhz 的设置例程
#define Fosc_60Mhz 0                    //系统时钟为 60Mhz
```

```
#define Fosc_80Mhz 1                    //系统时钟为 80Mhz
#define Fosc_120Mhz 2                   //系统时钟为 120Mhz

#define Main_Fosc Fosc_120Mhz //设置系统时钟为 120Mhz

void CLK_Init(void);                    //设置系统时钟，由 Main_Fosc 定义设置
void Timer0_Init(void);                 //定时器 0 初始化函数
void Io_Init(void);                     //I/O 口初始化函数，设置 P32 为开漏+打开内部上拉电阻模式
void Uart1_Init(void);                  //串口初始化函数，115200bps
void uart_send(int num);
bit P32_OUT = 1;                        //用于确定输出电平
char uart_buff[64] = {0};

void Delay100ms(void)                   //@120MHz
{
    unsigned long edata i;

    _nop_();
    _nop_();
    i = 2999998UL;
    while (i) i--;
}

char far ROM_EXE[4096] _at_ 0x800000;                                   //指定 ROM_EXE 地址
char ecode CODE_RAM_FLASH[4096] _at_ 0xfc2800;    //指定 CODE_RAM_FLASH 地址
char volatile far RAM_EXE[4096] _at_ 0x030000;                  //指定 RAM_EXE 地址

//用于运行 RAM 区域的函数
void exeram_code()
{
    while (1)
    {
        P6--;
        Delay100ms();
    }
}

void main(void)
{
    int i;
    EAXFR = 1;                          //使能访问扩展 RAM 区特殊功能寄存器（XFR）
    CKCON &= ~0x07;     //清空[2:0]，设置外部数据总线等待时钟为 0（最快），默认为 7
    CLK_Init();                         //设置 HPLL 时钟为指定频率
    Timer0_Init();                      //初始化定时器 0，50 毫秒@120MHz
    Uart1_Init();                       //串口初始化函数，115200bps
    Io_Init();                                  //初始化 I/O 口，设置 P32 等效为原准双向口模式（开漏模式
```

```
+打开内部上拉电阻）
    EA = 1;                                     //打开总中断
    CKCON &= ~0x80;
    for (i = 0; i < 4096; i++)RAM_EXE[i] = ((char far *)&exeram_code)[i];   //同一程序下的 RAM 程序运行
//      for (i = 0; i < 4096; i++)RAM_EXE[i] = ((char far *)&CODE_RAM_FLASH)[i];
                                                                            //独立程序下的 RAM 程序运行
    while(1)
    {
        //用户程序
        for (i = 0; i < 20; i++)uart_send(sprintf(uart_buff, "%02bx ", RAM_EXE[i]));
                                    //串口打印 RAM 内容的前 20 位
        uart_send(sprintf(uart_buff, "\n"));
        if(P33==0)
        {
            CKCON |= 0x80;          //切换到 ROM 运行方式
            _nop_();                            //add 2 nops here at least
            _nop_();
            _nop_();
            _nop_();
//                          ((void (far *)())&ROM_EXE)();          //通过数组跳转
            ((void (far *)())0x800000)();                   //通过绝对地址跳转，等效位为 SP->0x800000
//                          ((void (far *)())0x80000f)();
                                //独立程序通过绝对地址跳转，需要通过仿真查看独立程序的起始地址
        }//跳转到 RAM 地址
        Delay100ms();
    }
}

bit uart_flag = 0;
void send_dat(char c)
{
    uart_flag = 1;
    SBUF = c;
    while(uart_flag);
}

int data dat_len = 0;
void uart_send(int num)
{
    for(dat_len = 0; dat_len<num; dat_len++)
    {
        send_dat(uart_buff[dat_len]);
    }
}

void Uart1_Isr(void) interrupt 4
```

```
{
    if (TI)                             //检测串口 1 发送中断
    {
        TI = 0;                         //清除串口 1 发送中断请求位
        uart_flag = 0;
    }
    if (RI)                             //检测串口 1 接收中断
    {
        RI = 0;                         //清除串口 1 接收中断请求位
    }
}

void Uart1_Init(void)                   //115200bps@120MHz
{
    SCON = 0x50;                        //8 位数据,可变波特率
    AUXR |= 0x40;                       //定时器时钟 1T 模式
    AUXR &= 0xFE;                       //串口 1 选择定时器 1 为波特率发生器
    TMOD &= 0x0F;                       //设置定时器模式
    TL1 = 0xFC;                         //设置定时初始值
    TH1 = 0xFE;                         //设置定时初始值
    ET1 = 0;                            //禁止定时器中断
    TR1 = 1;                            //定时器 1 开始计时
    ES = 1;                             //使能串口 1 中断
}

char data off_t0_cnt = 0;
void Timer0_Isr(void) interrupt 1
{
    if(P32_OUT == 1&&P32 == 0&&off_t0_cnt<100) off_t0_cnt++;
                                        //判断外部 P32 按键按下一定时间时，关闭定时器 0
    if(off_t0_cnt>5){TR0 = 0;}          //注：仅在 P32 输出为 1 的时候，外部的按键按下才能被读到
    P32_OUT = ~P32_OUT;                 //每隔 10ms 亮/灭切换一次
    P32 = P32_OUT;                      //将输出电平给 P32 管脚
}

void Timer0_Init(void)                  //50 毫秒@120MHz
{
    TM0PS = 0x5B;                       //设置定时器时钟预分频 (注：并非所有系列都有此寄存器,详情请查看
数据手册)
    AUXR |= 0x80;                       //定时器时钟 1T 模式
    TMOD &= 0xF0;                       //设置定时器模式
    TL0 = 0x3F;                         //设置定时初始值
    TH0 = 0x01;                         //设置定时初始值
    TF0 = 0;                            //清除 TF0 标志
    TR0 = 1;                            //定时器 0 开始计时
    ET0 = 1;                            //使能定时器 0 中断
```

```
    T0CLKO = 1;                 //使能 P35 输出定时器溢出时钟
}
void Io_Init(void)
{
    P3M0 = 0x26; P3M1 = 0xdd;
    P3PU = 0x0d;
    P3SR = 0xdc;
    P3DR = 0xdc;
    P6M0 = 0xff; P6M1 = 0x00;           //推挽输出，P6，PWM
    P6SR = 0x00;                                //转换速度和驱动电流最大，P6
    P6DR = 0x00;
}

void Delay10ms(void)        //@120MHz
{
    unsigned long edata i;

    _nop_();
    _nop_();
    i = 299998UL;
    while (i) i--;
}

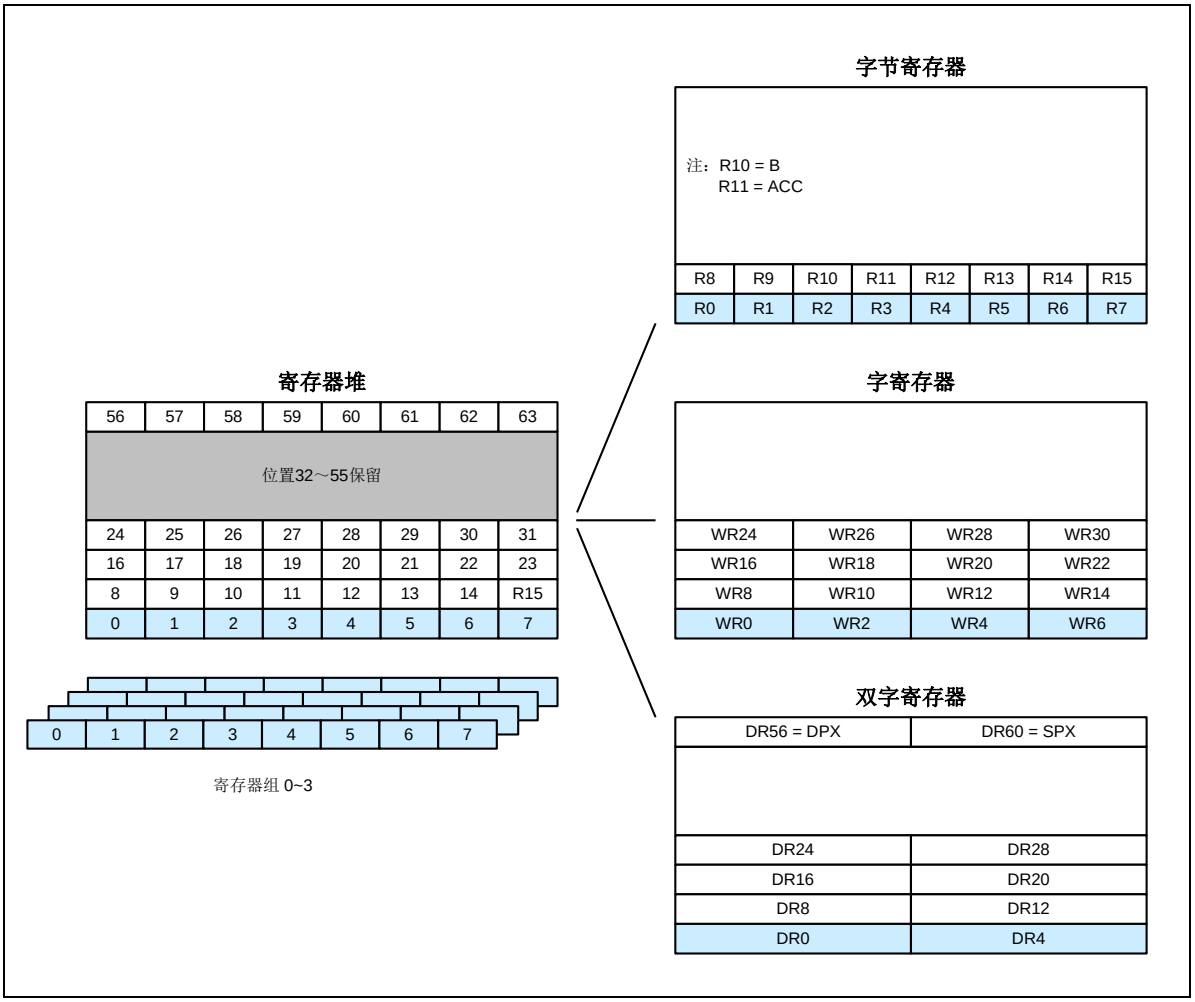
void CLK_Init(void)
{
#if Main_Fosc == Fosc_120Mhz
    WTST = 4;CLKDIV = 2;        //设置系统时钟=480MHz/2/2=120MHz,（因为 CLKSEL 选择时，已经将 HPLL/2
了）
#elif Main_Fosc == Fosc_80Mhz
    WTST = 3;CLKDIV = 3;        //设置系统时钟=480MHz/2/3=80MHz
#elif Main_Fosc == Fosc_60Mhz
    WTST = 2;CLKDIV = 4;        //设置系统时钟=480MHz/2/4=60MHz
#endif
//以下为超过 60MHz 时，系统时钟使用 HPLL 方式提供
    VRTRIM = CHIPID22;          //载入 27MHz 频段的 VRTRIM 值
    IRTRIM = CHIPID12;          //指定当前 HIRC 为 24MHz，此时会覆盖掉 ISP 设置的时钟频率
    IRCBAND &= ~0x03;           //清空频段选择
    IRCBAND |= 0x01;            //选择 27Mhz 频段
    HPLLCR &= ~0x10;            //选择 HPLL 输入时钟源为 HIRC
    HPLLPDIV = 4;                       //24MHz/4=6MHz,需要保证输入 HPLL 的时钟在 6MHz 附
近
    HPLLCR |= 0x0e;                     //HPLL=6MHz*80=480MHz
    HPLLCR |= 0x80;                     //使能 HPLL
    Delay10ms();
    CLKSEL &= ~0x03;            //BASE_CLK 选择为 HIRC，用以提供给 HPLL
    CLKSEL &= ~0x0c;            //清空主时钟源选择
```

```
    CLKSEL |= 1<<2;                         //设置主时钟源为内部 HPLL1 输出/2
}
```

12.4 寄存器堆

12.4.1 寄存器堆结构图

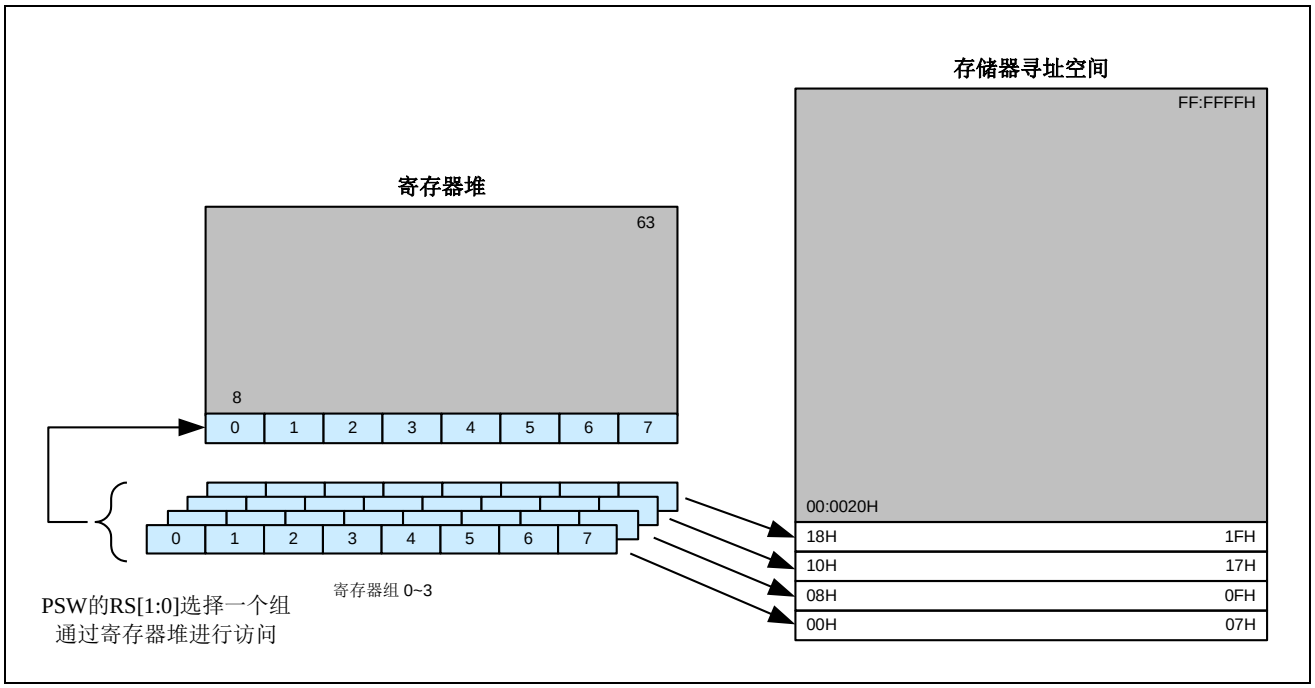
STC32G144K246 系列单片机的寄存器堆由 40 个位置组成：0～31 以及 56～63，如下图所示。这些位置可作为字节、字和双字访问。



位置 0 到 7 的寄存器堆实际上由四个可切换组组成，每个组有 8 个寄存器，如下图所示。这四个组使用片上内存的前 32 个字节，并且始终可在存储地址空间的 00：0000H ～ 00:001FH 的位置访问。通过 PSW 寄存器的 RS1 和 RS0 位选择四个组中的一个进行访问。

位置 8 到 31 和 56 到 63 的寄存器组始终可以访问。这些位置在 CPU 中作为寄存器使用。这些位置在寄存器堆是专用的，所以它们不作为片上内存的一部分。

位置 32 到 55 的寄存器堆保留，不能访问。



寄存器组选择

组	地址范围	PSW 选择位	
		RS1	RS0
工作组 0	00H – 07H	0	0
工作组 1	08H – 0FH	0	1
工作组 2	10H – 17H	1	0
工作组 3	18H – 1FH	1	1

12.4.2 字节、字以及双字寄存器

根据在寄存器堆中的位置，寄存器可作为字节、字以及、或者双字寻址。寄存器以其最低编号的字节位置命名。例如：

R4 是由位置 4 组成的字节寄存器。

WR4 是由第 4 和第 5 寄存器组成的字寄存器。

DR4 是由第 4 到第 7 寄存器组成的双字寄存器。

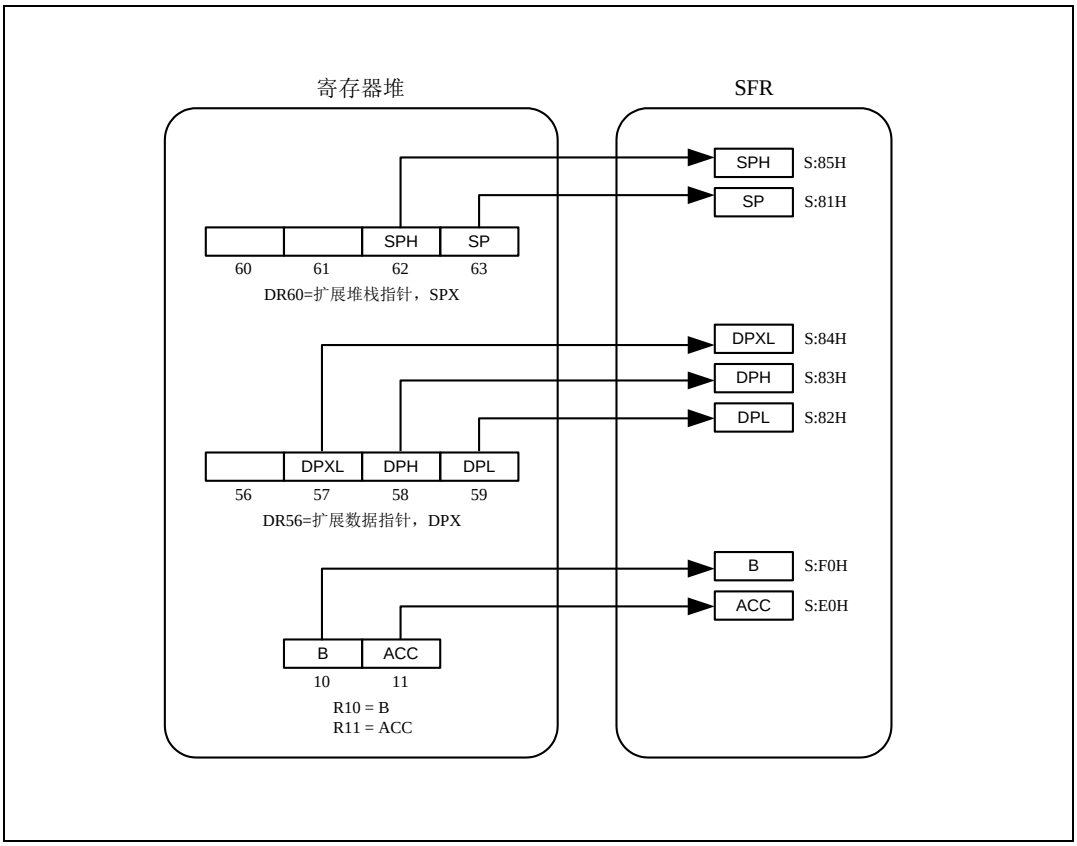
位置 R0～R15 可作为字节、字或者双字寻址。位置 16～31 只能作为字或者双字寻址。位置 56 到 63 只能作为双字寻址。

12.4.3 专用寄存器

寄存器堆有四个专用寄存器：

- R10 是寄存器 B
- R11 是累加器 ACC
- DR56 是扩展数据指针，DPX
- DR60 是扩展堆栈指针，SPX

这些寄存器位于寄存器堆中；然而 R10、R11 以及 DR56 和 DR60 的某些字节也可作为 SFR 访问。寄存器堆中的 DPX 和 SPX 字节只能通过双字寄存器寻址来访问。寄存器堆中的专用寄存器及其对应的 SFR 如下图所示



寄存器堆中的专用寄存器及其对应的 SFR

寄存器堆						SFR	
名称			助记符	寄存器	位置	助记符	地址
堆栈指针（SPX）	—		—	DR60	60	—	—
	—		—		61	—	—
	堆栈指针，高		SPH		62	SPH	S:85H
	堆栈指针，低		SP		63	SP	S:81H
数据指针（DPX）	—		—	DR56	56	—	—
	数据指针，扩展低		DPXL		57	DPXL	S:84H
	DPTR	数据指针，高	DPH		58	DPH	S:83H
		数据指针，低	DPL		59	DPL	S:82H
累加器（寄存器 A）			A	R11	11	ACC	S:E0H
寄存器 B			B	R10	10	B	S:F0H

12.4.4 扩展数据指针，DPX

双字寄存器 DR56 是扩展数据指针 DPX。DPX 的低三个字节（DPL、DPH 和 DPXL）可作为 SFR 访问。DPL 和 DPH 组成 16 位数据指针 DPTR。DPXL 的字节在位置 57 处。在用 MOVX 指令将数据搬入和搬出外部存储器时由 DPXL 寻址指定的区域。DPXL 的复位值为 01H。

12.4.5 扩展堆栈指针，SPX

双字寄存器 DR60 是堆栈指针 SPX。位置 63 的字节是 8 位堆栈指针 SP。位置 62 处的字节是堆栈指针高位 SPH。这两个字节允许堆栈扩展到内存 00 区域的顶部。SP 和 SPH 可以作为 SFR 访问。

PUSH 和 POP 两条指令直接寻址堆栈指针。子程序调用（ACALL、ECALL、LCALL）和返回（ERET、RET、RETI）也使用堆栈指针。请勿将 DR60 用作通用寄存器。

12.5 只读特殊功能寄存器中存储的唯一 ID 号和重要参数（CHIPID 和 CHIPIDX）

产品线	CHIPID
STC32G144K246 系列	●

STC32G144K246 系列单片机内部的只读特殊功能寄存器 CHIPID 中保存有与芯片相关的一些特殊参数，包括：全球唯一 ID 号、32K 掉电唤醒定时器的频率、内部 1.19V 参考信号源值（BGV）以及 IRC 参数。在用户程序中只能读取 CHIPID 中的内容，不可修改。使用 CHIPID 中的数据对用户程序进行加密是官方推荐的最优方案。

12.5.1 CHIPID 相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CHIPID0	硬件数字 ID00	7EFDE0H	全球唯一 ID 号（第 0 字节）								nnnn,nnnn
CHIPID1	硬件数字 ID01	7EFDE1H	全球唯一 ID 号（第 1 字节）								nnnn,nnnn
CHIPID2	硬件数字 ID02	7EFDE2H	全球唯一 ID 号（第 2 字节）								nnnn,nnnn
CHIPID3	硬件数字 ID03	7EFDE3H	全球唯一 ID 号（第 3 字节）								nnnn,nnnn
CHIPID4	硬件数字 ID04	7EFDE4H	全球唯一 ID 号（第 4 字节）								nnnn,nnnn
CHIPID5	硬件数字 ID05	7EFDE5H	全球唯一 ID 号（第 5 字节）								nnnn,nnnn
CHIPID6	硬件数字 ID06	7EFDE6H	全球唯一 ID 号（第 6 字节）								nnnn,nnnn
CHIPID7	硬件数字 ID07	7EFDE7H	内部 1.19V 参考信号源-BGV（高字节）								nnnn,nnnn
CHIPID8	硬件数字 ID08	7EFDE8H	内部 1.19V 参考信号源-BGV（低字节）								nnnn,nnnn
CHIPID9	硬件数字 ID09	7EFDE9H	32K 掉电唤醒定时器的频率（高字节）								nnnn,nnnn
CHIPID10	硬件数字 ID10	7EFDEAH	32K 掉电唤醒定时器的频率（低字节）								nnnn,nnnn
CHIPID11	硬件数字 ID11	7EFDEBH	22.1184MHz 的 IRC 参数（27M 频段）								nnnn,nnnn
CHIPID12	硬件数字 ID12	7EFDECH	24MHz 的 IRC 参数（27M 频段）								nnnn,nnnn
CHIPID13	硬件数字 ID13	7EFDEDH	40MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID14	硬件数字 ID14	7EFDEEH	45.1584MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID15	硬件数字 ID15	7EFDEFH	48MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID16	硬件数字 ID16	7EFDF0H	49.152MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID17	硬件数字 ID17	7EFDF1H	50.8032MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID18	硬件数字 ID18	7EFDF2H	55.296MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID19	硬件数字 ID19	7EFDF3H	60MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID20	硬件数字 ID20	7EFDF4H	64MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID21	硬件数字 ID21	7EFDF5H	10M 频段的 VRTRIM 参数								nnnn,nnnn
CHIPID22	硬件数字 ID22	7EFDF6H	27M 频段的 VRTRIM 参数								nnnn,nnnn
CHIPID23	硬件数字 ID23	7EFDF7H	44M 频段的 VRTRIM 参数								nnnn,nnnn
CHIPID24	硬件数字 ID24	7EFDF8H	72M 频段的 VRTRIM 参数								nnnn,nnnn
CHIPID25	硬件数字 ID25	7EFDF9H	00H								nnnn,nnnn
CHIPID26	硬件数字 ID26	7EFDFAH	用户程序空间结束地址（高字节）								nnnn,nnnn

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CHIPID27	硬件数字 ID27	7EFDFBH	芯片测试时间（年）								nnnn,nnnn
CHIPID28	硬件数字 ID28	7EFDFCH	芯片测试时间（月）								nnnn,nnnn
CHIPID29	硬件数字 ID29	7EFDFDH	芯片测试时间（日）								nnnn,nnnn
CHIPID30	硬件数字 ID30	7EFDFEH	芯片封装形式编号								nnnn,nnnn
CHIPID31	硬件数字 ID31	7EFDFFH	5AH								nnnn,nnnn

12.5.2 CHIPID 之全球唯一 ID 号解读

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CHIPID0	硬件数字 ID00	7EFDE0H	全球唯一 ID 号（第 0 字节）								nnnn,nnnn
CHIPID1	硬件数字 ID01	7EFDE1H	全球唯一 ID 号（第 1 字节）								nnnn,nnnn
CHIPID2	硬件数字 ID02	7EFDE2H	全球唯一 ID 号（第 2 字节）								nnnn,nnnn
CHIPID3	硬件数字 ID03	7EFDE3H	全球唯一 ID 号（第 3 字节）								nnnn,nnnn
CHIPID4	硬件数字 ID04	7EFDE4H	全球唯一 ID 号（第 4 字节）								nnnn,nnnn
CHIPID5	硬件数字 ID05	7EFDE5H	全球唯一 ID 号（第 5 字节）								nnnn,nnnn
CHIPID6	硬件数字 ID06	7EFDE6H	全球唯一 ID 号（第 6 字节）								nnnn,nnnn

[CHIPID0, CHIPID1]：16 位 MCU ID，用于区别不同的单片机型号（高位在前）。

[CHIPID2, CHIPID3]：16 位测试机台编号（高位在前）。

[CHIPID4, CHIPID5, CHIPID6]：24 位测试流水编号（高位在前）。

12.5.3 CHIPID 之内部参考信号源解读

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CHIPID7	硬件数字 ID07	7EFDE7H	内部 1.19V 参考信号源-BGV（高字节）								nnnn,nnnn
CHIPID8	硬件数字 ID08	7EFDE8H	内部 1.19V 参考信号源-BGV（低字节）								nnnn,nnnn

[CHIPID7, CHIPID8]：16 位内部参考信号源电压值（高位在前）。

标准值为 1190（04A6H），单位为 mV，即 1.19V。但实际的芯片由于存在制造误差。内部参考信号源的电压值并不会受工作电压 VCC 的影响，所以内部参考信号源可以和 ADC 结合用于校准 ADC，也可和比较器结合用于侦测工作电压。

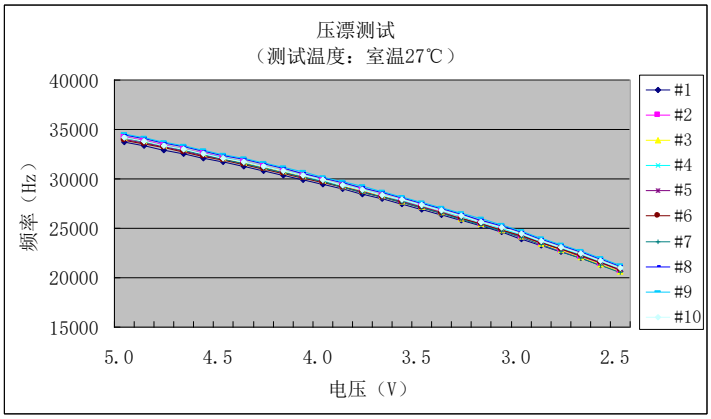
12.5.4 CHIPID 之内部 32K 的 IRC 振荡频率解读

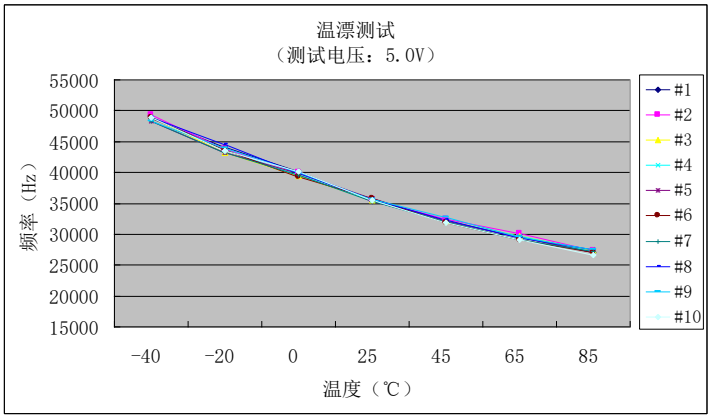
符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CHIPID9	硬件数字 ID09	7EFDE9H	32K 掉电唤醒定时器的频率（高字节）								nnnn,nnnn
CHIPID10	硬件数字 ID10	7EFDEAH	32K 掉电唤醒定时器的频率（低字节）								nnnn,nnnn

[CHIPID9, CHIPID10]：16 位 32K IRC 振荡器频率值（高位在前）。

标准值为 32768（8000H），单位为 Hz，即 32.768KHz。但实际的芯片由于存在制造误差，而且温漂和压漂均比较大。

内部 32K 振荡器的压漂测试线性图和温漂线性图如下：





12.5.5 CHIPID 之高精度 IRC 参数解读

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CHIPID11	硬件数字 ID11	7EFDEBH	22.1184MHz 的 IRC 参数（27M 频段）								nnnn,nnnn
CHIPID12	硬件数字 ID12	7EFDECH	24MHz 的 IRC 参数（27M 频段）								nnnn,nnnn
CHIPID13	硬件数字 ID13	7EFDEDH	40MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID14	硬件数字 ID14	7EFDEEH	45.1584MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID15	硬件数字 ID15	7EFDEFH	48MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID16	硬件数字 ID16	7EFDF0H	49.152MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID17	硬件数字 ID17	7EFDF1H	50.8032MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID18	硬件数字 ID18	7EFDF2H	55.296MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID19	硬件数字 ID19	7EFDF3H	60MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID20	硬件数字 ID20	7EFDF4H	64MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID21	硬件数字 ID21	7EFDF5H	10M 频段的 VRTRIM 参数								nnnn,nnnn
CHIPID22	硬件数字 ID22	7EFDF6H	27M 频段的 VRTRIM 参数								nnnn,nnnn
CHIPID23	硬件数字 ID23	7EFDF7H	44M 频段的 VRTRIM 参数								nnnn,nnnn
CHIPID24	硬件数字 ID24	7EFDF8H	72M 频段的 VRTRIM 参数								nnnn,nnnn

支持 CHIPID 功能的 STC32G144K246 系列单片机，内部集成的高精度 IRC 分 4 个频段，每个频段对应的参考电压值在出厂时已进行了校准，当选择不同的频段时，只需要将相应频段的电压校准值填入 VRTRIM 寄存器即可。4 个频段的中心频率分别为 10MHz、27MHz、44MHz 和 72MHz，由于制造误差，中心频率一般可能有±5%的偏差，为了得到精确的用户频率，可使用 IRTRIM 对频率进行微调校准。使用 STC 官方提供的下载软件下载用户程序时，系统会根据用户所设定频率自动设置 VRTRIM 和 IRTRIM 寄存器。同时，在 CHIPID 也内部预置了 10 个常用频率的 IRTRIM 值以及 4 个频段的参考电压校准值，让用户可以在程序运行过程中动态的修改工作频率。

[CHIPID11 : CHIPID20]：10 个常用频率的 IRTRIM 值。括号里面的注解即为对应的频段

[CHIPID21 : CHIPID24]：4 个频段的参考电压值校准值。

用户动态修改频率时，只需要将[CHIPID11 : CHIPID20]中的某个频率校准值读出并写入 IRTRIM 寄存器，同时根据该频率所对应的频段将[CHIPID21 ： CHIPID24]中的某个电压校准值读出并写入 VRTRIM 寄存器即可。详细操作请参考后续章节的范例程序。

12.5.6 CHIPID 之测试时间参数解读

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CHIPID27	硬件数字 ID27	7EFDFBH	芯片测试时间（年）								nnnn,nnnn
CHIPID28	硬件数字 ID28	7EFDFCH	芯片测试时间（月）								nnnn,nnnn
CHIPID29	硬件数字 ID29	7EFDFDH	芯片测试时间（日）								nnnn,nnnn

测试时间的年、月、日参数均为 BCD 码。（例如：CHIPID27＝0x21，CHIPID28=0x11，CHIPID29=0x18，则目标芯片的生产测试日期为 2021 年 11 月 18 日）

12.5.7 CHIPID 之芯片封装形式编号解读

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CHIPID30	硬件数字 ID30	7EFDFEH	芯片封装形式编号								nnnn,nnnn

封装编号	封装形式		封装编号	封装形式
0x00	DIP8		0x50	SOP32
0x01	SOP8		0x51	LQFP32
0x02	DFN8		0x52	QFN32
0x10	DIP16		0x53	PLCC32
0x11	SOP16		0x54	QFN32S
0x20	DIP18		0x60	PDIP40
0x21	SOP18		0x70	LQFP44
0x30	DIP20		0x71	PLCC44
0x31	SOP20		0x72	PQFP44
0x32	TSSOP20		0x80	LQFP48
0x33	LSSOP20		0x81	QFN48
0x34	QFN20		0x90	LQFP64
0x40	SKDIP28		0x91	LQFP64S
0x41	SOP28		0x92	LQFP64L
0x42	TSSOP28		0x93	LQFP64M
0x43	QFN28		0x94	QFN64

13 特殊功能寄存器（SFR、XFR），EAXFR 允许访问 XFR

13.1 EAXFR 允许访问扩展的 XFR，取指令不减速，访问片内部 XRAM 不减速

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW2	BAH	EAXFR	-	I2C_S[1:0]		-	S4_S	S3_S	S2_S

EAXFR：扩展 RAM 区特殊功能寄存器（XFR）访问控制寄存器

0：禁止访问 XFR

1：使能访问 XFR。

当需要访问 XFR 时，必须先将 EAXFR 置 1，才能对 XFR 进行正常的读写。建议上电初始化时直接设置为 1，后续不要再修改

STC32G144K246 系列单片机存储器地址的编址如下：

区域	地址范围	功能	说明
数据区	00:0000H — 00:FFFFH	数据区（edata）	堆栈区
	01:0000H — 02:FFFFH	内部扩展数据区（xdata）	
	03:0000H — 7D:FFFFH	数据保留区	
	7E:0000H — 7E:FFFFH	扩展 SFR 区（XFR）	需使能 EAXFR（P_SW2.7）才能访问
	7F:0000H — 7F:FFFFH	外部数据区（xdata）	需使能 EXTRAM（AUXR.1）才能访问
代码区	80:0000H — 80:0FFFH	RAM 代码区	需使能 RAMEXE（CKCON.7）才能运行代码
	80:1000H — FB:FFFFH	代码保留区	
	FC:0000H — FC:27FFH	系统代码区	
	FC:2800H — FE:FFFFH	扩展代码区（ecode）	
	FF:0000H — FF:FFFFH	代码区（code）	

如需访问 XFR 区域的扩展 SFR，需要先将 EAXFR（P_SW2.7）置 1，由于 STC32G144K246 的存储器地址的编址方式是线性编址的，XFR 区域的地址是独立的一块地址区域，所以可以在**上电系统初始化时将 EAXFR 寄存器写 1（例如：EAXFR = 1;），后续一直保持为 1 不用再修改**，即可正常访问 XFR 区域。

注意：由于 EAXFR 控制位和 S2_S、S3_S 等位寄存器位共用 P_SW2，在设置 P_SW2 中的 S2_S、S3_S 时，建议直接使用位操作语句（例如 S2_S = 0; S3_S = 1;），避免直接操作 P_SW2 寄存器。如代码中确实需要直接对 P_SW2 进行修改，也一定要使用与或指令（例如 P_SW2 &= ~0x01; P_SW2 |= 0x02;）。

13.2 STC32G144K246 系列

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
F8H	P7	LINICR	LINAR	LINDR	USBADR	S4CON	S4BUF	RSTCFG
F0H	B	CANICR	IP4	IP4H	USBCON	IAP_TPS	IAP_ADDRE	ICHECR
E8H	P6	WTST	CKCON	MXAX	USBDAT	DMAIR	IP3H	AUXINTIF
E0H	ACC	P7M1	P7M0	DPS	~~DPL1~~	~~DPH1~~	~~CMPCR1~~	~~CMPCR2~~
D8H	DPUOP	IAP_DATA1	IAP_DATA2	IAP_DATA3	USBCLK	T4T3M	ADCCFG	IP3
D0H	PSW	PSW1	TH4	TL4	TH3	TL3	TH2	TL2
C8H	P5	P5M1	P5M0	P6M1	P6M0	SPSTAT	SPCTL	SPDAT
C0H	P4	WDT_CONTR	IAP_DATA	IAP_ADDRH	IAP_ADDRL	IAP_CMD	IAP_TRIG	IAP_CONTR
B8H	IP	SADEN	P_SW2	P_SW3	ADC_CONTR	ADC_RES	ADC_RESL	P_SW4
B0H	P3	P3M1	P3M0	P4M1	P4M0	IP2	IP2H	IPH
A8H	IE	SADDR	WKTCL	WKTCH	S3CON	S3BUF	TA	IE2
A0H	P2	BUS_SPEED	P_SW1	reserved	reserved	reserved	VRTRIM	reserved
98H	SCON	SBUF	S2CON	S2BUF	reserved	IRCBAND	LIRTRIM	IRTRIM
90H	P1	P1M1	P1M0	P0M1	P0M0	P2M1	P2M0	AUXR2
88H	TCON	TMOD	TL0	TL1	TH0	TH1	AUXR	INTCLKO
80H	P0	SP	DPL	DPH	DPXL	SPH	DPUST	PCON

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
7EFEF8	PWMB_CCR6L	PWMB_CCR7H	PWMB_CCR7L	PWMB_CCR8H	PWMB_CCR8L	PWMB_BKR	PWMB_DTR	PWMB_OISR
7EFEF0	PWMB_PSCRH	PWMB_PSCRL	PWMB_ARRH	PWMB_ARRL	PWMB_RCR	PWMB_CCR5H	PWMB_CCR5L	PWMB_CCR6H
7EFEE8	PWMB_CCMR1	PWMB_CCMR2	PWMB_CCMR3	PWMB_CCMR4	PWMB_CCER1	PWMB_CCER2	PWMB_CNTRH	PWMB_CNTRL
7EFEE0	PWMB_CR1	PWMB_CR2	PWMB_SMCR	PWMB_ETR	PWMB_IER	PWMB_SR1	PWMB_SR2	PWMB_EGR
7EFED8	PWMA_CCR2L	PWMA_CCR3H	PWMA_CCR3L	PWMA_CCR4H	PWMA_CCR4L	PWMA_BKR	PWMA_DTR	PWMA_OISR
7EFED0	PWMA_PSCRH	PWMA_PSCRL	PWMA_ARRH	PWMA_ARRL	PWMA_RCR	PWMA_CCR1H	PWMA_CCR1L	PWMA_CCR2H
7EFEC8	PWMA_CCMR1	PWMA_CCMR2	PWMA_CCMR3	PWMA_CCMR4	PWMA_CCER1	PWMA_CCER2	PWMA_CNTRH	PWMA_CNTRL
7EFEC0	PWMA_CR1	PWMA_CR2	PWMA_SMCR	PWMA_ETR	PWMA_IER	PWMA_SR1	PWMA_SR2	PWMA_EGR
7EFEB8	PWMA_PS2	PWMA_RCRH	PWMB_RCRH					
7EFEB0	PWMA_ETRPS	PWMA_ENO	PWMA_PS	PWMA_IOAUX	PWMB_ETRPS	PWMB_ENO	PWMB_PS	PWMB_IOAUX
7EFEA8	ADCTIM	INTXIT			TIMERPS	ADCEXCFG		ADCEXCFG2
7EFEA0	TM0PS	TM1PS	TM2PS	TM3PS	TM4PS			
7EFE98		RSTFLAG	RSTCR0	RSTCR1	RSTCR2	RSTCR3	RSTCR4	RSTCR5
7EFE90	SPI_CLKDIV	PWMA_CLKDIV	PWMB_CLKDIV	TFPU_CLKDIV	I2S_CLKDIV	I2S2_CLKDIV	PWMC_CLKDIV	PWMD_CLKDIV
7EFE88	I2CMSAUX	I2CPSCR						
7EFE80	I2CCFG	I2CMSCR	I2CMSST	I2CSLCR	I2CSLST	I2CSLADR	I2CTxD	I2CRxD
7EFE78	T11WCR	T11WPS	T11WH	T11WL				
7EFE70	RTCYEAR	RTCMONTH	RTCDAY	RTCHOUR	RTCMIN	RTCSEC	RTCSSEC	
7EFE68	INIYEAR	INIMONTH	INIDAY	INIHOUR	INIMIN	INISEC	INISSEC	RTCWEEK
7EFE60	RTCCR	RTCCFG	RTCIEN	RTCIF	ALAHOUR	ALAMIN	ALASEC	ALASSEC
7EFE50	LCMIFCFG	LCMIFCFG2	LCMIFCR	LCMIFSTA	LCMIFDATL	LCMDATH	LCMIFPSCR	
7EFE48	P0BP	P1BP	P2BP	P3BP	P4BP	P5BP	P6BP	P7BP
7EFE40	P0PD	P1PD	P2PD	P3PD	P4PD	P5PD	P6PD	P7PD
7EFE30	P0IE	P1IE	P2IE	P3IE	P4IE	P5IE	P6IE	P7IE
7EFE28	P0DR	P1DR	P2DR	P3DR	P4DR	P5DR	P6DR	P7DR
7EFE20	P0SR	P1SR	P2SR	P3SR	P4SR	P5SR	P6SR	P7SR
7EFE18	P0NCS	P1NCS	P2NCS	P3NCS	P4NCS	P5NCS	P6NCS	P7NCS
7EFE10	P0PU	P1PU	P2PU	P3PU	P4PU	P5PU	P6PU	P7PU
7EFE08	X32KCR	IRC48MATRIM	IRC48MBTRIM		HPLLCR	HPLLPDIV	HPLL2CR	HPLL2PDIV
7EFE00	CLKSEL	CLKDIV	HIRCCR	XOSCCR	IRC32KCR	MCLKOCR	IRCDB	IRC48MCR

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
7EFDF8	CHIPID[24]	CHIPID[25]	CHIPID[26]	CHIPID[27]	CHIPID[28]	CHIPID[29]	CHIPID[30]	CHIPID[31]
7EFDF0	CHIPID[16]	CHIPID[17]	CHIPID[18]	CHIPID[19]	CHIPID[20]	CHIPID[21]	CHIPID[22]	CHIPID[23]
7EFDE8	CHIPID[8]	CHIPID[9]	CHIPID[10]	CHIPID[11]	CHIPID[12]	CHIPID[13]	CHIPID[14]	CHIPID[15]
7EFDE0	CHIPID[0]	CHIPID[1]	CHIPID[2]	CHIPID[3]	CHIPID[4]	CHIPID[5]	CHIPID[6]	CHIPID[7]
7EFDD8	USART4CR1	USART4CR2	USART4CR3	USART4CR4	USART4CR5	USART4GTR	USART4BRH	USART4BRL
7EFDD0	USART3CR1	USART3CR2	USART3CR3	USART3CR4	USART3CR5	USART3GTR	USART3BRH	USART3BRL
7EFDC8	USART2CR1	USART2CR2	USART2CR3	USART2CR4	USART2CR5	USART2GTR	USART2BRH	USART2BRL
7EFDC0	USARTCR1	USARTCR2	USARTCR3	USARTCR4	USARTCR5	USARTGTR	USARTBRH	USARTBRL
7EFDB8	S3CFG	S3ADDR	S3ADEN		S4CFG	S4ADDR	S4ADEN	
7EFDB0					S2CFG	S2ADDR	S2ADEN	
7EFDA8	CRECR	CRECNTH	CRECNTL	CRERES	CRECFG			
7EFDA0	-	I2SMCKDIV	I2SINSEL		-	I2S2MCKDIV	I2S2INSEL	
7EFD98	I2SCR	I2SSR	I2SDRH	I2SDRL	I2SPRH	I2SPRL	I2SCFGH	I2SCFGL
7EFD90	I2S2CR	I2S2SR	I2S2DRH	I2S2DRL	I2S2PRH	I2S2PRL	I2S2CFGH	I2S2CFGL
7EFD88	UR1TOTE	UR2TOTE	UR3TOTE	UR4TOTE	SPITOTE	I2CTOTE		
7EFD80	SPITOCR	SPITOSR	SPITOTH	SPITOTL	I2CTOCR	I2CTOSR	I2CTOTH	I2CTOTL
7EFD78	UR3TOCR	UR3TOSR	UR3TOTH	UR3TOTL	UR4TOCR	UR4TOSR	UR4TOTH	UR4TOTL
7EFD70	UR1TOCR	UR1TOSR	UR1TOTH	UR1TOTL	UR2TOCR	UR2TOSR	UR2TOTH	UR2TOTL
7EFD68		P_SWX1	P_SWX2	P_SWX3	P_SWX4	P_SWX5	P_SWX6	P_SWX7
7EFD60	PINIPL	PINIPH	PIN2IPL	PIN2IPH				
7EFD48	P8WKUE	P9WKUE	PAWKUE	PBWKUE				
7EFD40	P0WKUE	P1WKUE	P2WKUE	P3WKUE	P4WKUE	P5WKUE	P6WKUE	P7WKUE
7EFD38	P8IM1	P9IM1	PAIM1	PBIM1				
7EFD30	P0IM1	P1IM1	P2IM1	P3IM1	P4IM1	P5IM1	P6IM1	P7IM1
7EFD28	P8IM0	P9IM0	PAIM0	PBIM0				
7EFD20	P0IM0	P1IM0	P2IM0	P3IM0	P4IM0	P5IM0	P6IM0	P7IM0
7EFD18	P8INTF	P9INTF	PAINTF	PBINTF				
7EFD10	P0INTF	P1INTF	P2INTF	P3INTF	P4INTF	P5INTF	P6INTF	P7INTF
7EFD08	P8INTE	P9INTE	PAINTE	PBINTE				
7EFD00	P0INTE	P1INTE	P2INTE	P3INTE	P4INTE	P5INTE	P6INTE	P7INTE

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
7EFBF8	HSSPI_CFG	HSSPI_CFG2	HSSPI_STA	HSSPI_PSCR				
7EFBF0	HSPWMA_CFG	HSPWMA_ADR	HSPWMA_DAT		HSPWMB_CFG	HSPWMB_ADR	HSPWMB_DAT	
7EFBE8	CHIPIDX[24]	CHIPIDX[25]	CHIPIDX[26]	CHIPIDX[27]	CHIPIDX[28]	CHIPIDX[29]	CHIPIDX[30]	CHIPIDX[31]
7EFBE0	CHIPIDX[16]	CHIPIDX[17]	CHIPIDX[18]	CHIPIDX[19]	CHIPIDX[20]	CHIPIDX[21]	CHIPIDX[22]	CHIPIDX[23]
7EFBD8	CHIPIDX[8]	CHIPIDX[9]	CHIPIDX[10]	CHIPIDX[11]	CHIPIDX[12]	CHIPIDX[13]	CHIPIDX[14]	CHIPIDX[15]
7EFBD0	CHIPIDX[0]	CHIPIDX[1]	CHIPIDX[2]	CHIPIDX[3]	CHIPIDX[4]	CHIPIDX[5]	CHIPIDX[6]	CHIPIDX[7]
7EFBC8	TM17PS	TM18PS	T18T17CFG	T18T17M	TH18	TL18	TH17	TL17
7EFBC0	TM9PS	TM10PS	T10T9CFG	T10T9M	TH10	TL10	TH9	TL9
7EFBB8	TM7PS	TM8PS	T8T7CFG	T8T7M	TH8	TL8	TH7	TL7
7EFBB0	TM5PS	TM6PS	T6T5CFG	T6T5M	TH6	TL6	TH5	TL5
7EFB78	CMP3_6P6N_CR1	CMP3_6P6N_CR2	CMP3_6P6N_CR3	CMP3_6P6N_CR4	CMP4_6P6N_CR1	CMP4_6P6N_CR2	CMP4_6P6N_CR3	CMP4_6P6N_CR4
7EFB70	CMP1_6P6N_CR1	CMP1_6P6N_CR2	CMP1_6P6N_CR3	CMP1_6P6N_CR4	CMP2_6P6N_CR1	CMP2_6P6N_CR2	CMP2_6P6N_CR3	CMP2_6P6N_CR4
7EFB38	ADC2_CONTR	ADC2CFG	ADC2_RES	ADC2_RESL	ADC2TIM	ADC2EXCFG	ADC2INTR	ADC2EXCFG2
7EFB30	CRC_CR	CRC_DIN	CRC_DOH CRC_DO0	CRC_DOL CRC_DO1	CRC_DO2	CRC_DO3		

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
7EFAF8	DMA_ARBCFG	DMA_ARBSTA						
7EFAF0	DMA_P2P1_DEST	DMA_P2P1_SRC	DMA_P2P2_DEST	DMA_P2P2_SRC				
7EFAE8	DMA_ADC2_RXAH	DMA_ADC2_RXAL	DMA_ADC2_CHSW0	DMA_ADC2_CHSW1			DMA_ADC2_ITVH	DMA_ADC2_ITVL
7EFAE0	DMA_ADC2_CFG	DMA_ADC2_CR	DMA_ADC2_STA	DMA_ADC2_CFG2	DMA_ADC2_AMTH	DMA_ADC2_AMT	DMA_ADC2_DONEH	DMA_ADC2_DONE
7EFAD8	DMA_QSPI_RXAL			DMA_QSPI_AMTH	DMA_QSPI_DONEH		DMA_QSPI_ITVH	DMA_QSPI_ITVL
7EFAD0	DMA_QSPI_CFG	DMA_QSPI_CR	DMA_QSPI_STA	DMA_QSPI_AMT	DMA_QSPI_DONE	DMA_QSPI_TXAH	DMA_QSPI_TXAL	DMA_QSPI_RXAH
7EFAC8	DMA_UR1_ITVH	DMA_UR1_ITVL	DMA_UR2_ITVH	DMA_UR2_ITVL	DMA_UR3_ITVH	DMA_UR3_ITVL	DMA_UR4_ITVH	DMA_UR4_ITVL
7EFAC0	DMA_I2ST_AMTH	DMA_I2ST_DONEH	DMA_I2SR_AMTH	DMA_I2SR_DONEH	DMA_I2C_ITVH	DMA_I2C_ITVL	DMA_I2S_ITVH	DMA_I2S_ITVL
7EFAB8	DMA_I2SR_CFG	DMA_I2SR_CR	DMA_I2SR_STA	DMA_I2SR_AMT	DMA_I2SR_DONE	DMA_I2SR_RXAH	DMA_I2SR_RXAL	
7EFAB0	DMA_I2ST_CFG	DMA_I2ST_CR	DMA_I2ST_STA	DMA_I2ST_AMT	DMA_I2ST_DONE	DMA_I2ST_TXAH	DMA_I2ST_TXAL	
7EFAA8	DMA_I2CT_AMTH	DMA_I2CT_DONEH	DMA_I2CR_AMTH	DMA_I2CR_DONEH		DMA_I2C_CR	DMA_I2C_ST1	DMA_I2C_ST2
7EFAA0	DMA_I2CR_CFG	DMA_I2CR_CR	DMA_I2CR_STA	DMA_I2CR_AMT	DMA_I2CR_DONE	DMA_I2CR_RXAH	DMA_I2CR_RXAL	
7EFA98	DMA_I2CT_CFG	DMA_I2CT_CR	DMA_I2CT_STA	DMA_I2CT_AMT	DMA_I2CT_DONE	DMA_I2CT_TXAH	DMA_I2CT_TXAL	
7EFA90	DMA_UR3T_AMTH	DMA_UR3T_DONEH	DMA_UR3R_AMTH	DMA_UR3R_DONEH	DMA_UR4T_AMTH	DMA_UR4T_DONEH	DMA_UR4R_AMTH	DMA_UR4R_DONEH
7EFA88	DMA_UR1T_AMTH	DMA_UR1T_DONEH	DMA_UR1R_AMTH	DMA_UR1R_DONEH	DMA_UR2T_AMTH	DMA_UR2T_DONEH	DMA_UR2R_AMTH	DMA_UR2R_DONEH
7EFA80	DMA_M2M_AMTH	DMA_M2M_DONEH	DMA_ADC_AMTH	DMA_ADC_DONEH	DMA_SPI_AMTH	DMA_SPI_DONEH	DMA_LCM_AMTH	DMA_LCM_DONEH
7EFA78	DMA_LCM_RXAL						DMA_LCM_ITVH	DMA_LCM_ITVL
7EFA70	DMA_LCM_CFG	DMA_LCM_CR	DMA_LCM_STA	DMA_LCM_AMT	DMA_LCM_DONE	DMA_LCM_TXAH	DMA_LCM_TXAL	DMA_LCM_RXAH
7EFA68	DMA_UR4R_CFG	DMA_UR4R_CR	DMA_UR4R_STA	DMA_UR4R_AMT	DMA_UR4R_DONE	DMA_UR4R_RXAH	DMA_UR4R_RXAL	
7EFA60	DMA_UR4T_CFG	DMA_UR4T_CR	DMA_UR4T_STA	DMA_UR4T_AMT	DMA_UR4T_DONE	DMA_UR4T_TXAH	DMA_UR4T_TXAL	
7EFA58	DMA_UR3R_CFG	DMA_UR3R_CR	DMA_UR3R_STA	DMA_UR3R_AMT	DMA_UR3R_DONE	DMA_UR3R_RXAH	DMA_UR3R_RXAL	
7EFA50	DMA_UR3T_CFG	DMA_UR3T_CR	DMA_UR3T_STA	DMA_UR3T_AMT	DMA_UR3T_DONE	DMA_UR3T_TXAH	DMA_UR3T_TXAL	
7EFA48	DMA_UR2R_CFG	DMA_UR2R_CR	DMA_UR2R_STA	DMA_UR2R_AMT	DMA_UR2R_DONE	DMA_UR2R_RXAH	DMA_UR2R_RXAL	
7EFA40	DMA_UR2T_CFG	DMA_UR2T_CR	DMA_UR2T_STA	DMA_UR2T_AMT	DMA_UR2T_DONE	DMA_UR2T_TXAH	DMA_UR2T_TXAL	
7EFA38	DMA_UR1R_CFG	DMA_UR1R_CR	DMA_UR1R_STA	DMA_UR1R_AMT	DMA_UR1R_DONE	DMA_UR1R_RXAH	DMA_UR1R_RXAL	
7EFA30	DMA_UR1T_CFG	DMA_UR1T_CR	DMA_UR1T_STA	DMA_UR1T_AMT	DMA_UR1T_DONE	DMA_UR1T_TXAH	DMA_UR1T_TXAL	
7EFA28	DMA_SPI_RXAL	DMA_SPI_CFG2					DMA_SPI_ITVH	DMA_SPI_ITVL
7EFA20	DMA_SPI_CFG	DMA_SPI_CR	DMA_SPI_STA	DMA_SPI_AMT	DMA_SPI_DONE	DMA_SPI_TXAH	DMA_SPI_TXAL	DMA_SPI_RXAH
7EFA18	DMA_ADC_RXAL	DMA_ADC_CFG2	DMA_ADC_CHSW0	DMA_ADC_CHSW1			DMA_ADC_ITVH	DMA_ADC_ITVL
7EFA10	DMA_ADC_CFG	DMA_ADC_CR	DMA_ADC_STA	DMA_ADC_AMT	DMA_ADC_DONE			DMA_ADC_RXAH
7EFA08	DMA_M2M_RXAL							
7EFA00	DMA_M2M_CFG	DMA_M2M_CR	DMA_M2M_STA	DMA_M2M_AMT	DMA_M2M_DONE	DMA_M2M_TXAH	DMA_M2M_TXAL	DMA_M2M_RXAH

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
7EF9F8	P8BP	P9BP	PABP	PBBP				
7EF9F0	P8PD	P9PD	PAPD	PBPD				
7EF9E0	P8IE	P9IE	PAIE	PBIE				
7EF9D8	P8DR	P9DR	PADR	PBDR				
7EF9D0	P8SR	P9SR	PASR	PBSR				
7EF9C8	P8NCS	P9NCS	PANCS	PBNCS				
7EF9C0	P8PU	P9PU	PAPU	PBPU				
7EF9B8	DMA_PWMCR_RXAH	DMA_PWMCR_RXAL						
7EF9B0	DMA_PWMCR_CFG	DMA_PWMCR_CR	DMA_PWMCR_STA		DMA_PWMCR_AMTH	DMA_PWMCR_AMT	DMA_PWMCR_DONEH	DMA_PWMCR_DONE
7EF9A8	DMA_PWMCT_TXAH	DMA_PWMCT_TXAL					DMA_PWMC_ITVH	DMA_PWMC_ITVL
7EF9A0	DMA_PWMCT_CFG	DMA_PWMCT_CR	DMA_PWMCT_STA		DMA_PWMCT_AMTH	DMA_PWMCT_AMT	DMA_PWMCT_DONEH	DMA_PWMCT_DONE
7EF998	DMA_PWMAR_RXAH	DMA_PWMAR_RXAL						
7EF990	DMA_PWMAR_CFG	DMA_PWMAR_CR	DMA_PWMAR_STA		DMA_PWMAR_AMTH	DMA_PWMAR_AMT	DMA_PWMAR_DONEH	DMA_PWMAR_DONE
7EF988	DMA_PWMAT_TXAH	DMA_PWMAT_TXAL					DMA_PWMA_ITVH	DMA_PWMA_ITVL
7EF980	DMA_PWMAT_CFG	DMA_PWMAT_CR	DMA_PWMAT_STA		DMA_PWMAT_AMTH	DMA_PWMAT_AMT	DMA_PWMAT_DONEH	DMA_PWMAT_DONE
7EF970	HSPWMC_CFG	HSPWMC_ADR	HSPWMC_DAT		HSPWMD_CFG	HSPWMD_ADR	HSPWMD_DAT	
7EF968	PWMC_DER	PWMC_DBA	PWMC_DBL	PWMC_DMACR				
7EF960	PWMC_CCR5H	PWMC_CCR5L	PWMC_CCR5X	PWMC_CCR6H	PWMC_CCR6L			
7EF958	PWMC_CCMR1X	PWMC_CCMR2X	PWMC_CCMR3X	PWMC_CCMR4X	PWMC_CCMR5	PWMC_CCMR5X	PWMC_CCMR6	PWMC_CCMR6X
7EF950	PWMC_ENO2	PWMC_IOAUX2	PWMC_CR3	PWMC_SR3	PWMC_CCER3			
7EF948	PWMA_DER	PWMA_DBA	PWMA_DBL	PWMA_DMACR				
7EF940	PWMA_CCR5H	PWMA_CCR5L	PWMA_CCR5X	PWMA_CCR6H	PWMA_CCR6L			
7EF938	PWMA_CCMR1X	PWMA_CCMR2X	PWMA_CCMR3X	PWMA_CCMR4X	PWMA_CCMR5	PWMA_CCMR5X	PWMA_CCMR6	PWMA_CCMR6X
7EF930	PWMA_ENO2	PWMA_IOAUX2	PWMA_CR3	PWMA_SR3	PWMA_CCER3			
7EF928	QSPI_PSMAR1				QSPI_PIR1	QSPI_PIR2		
7EF920	QSPI_DR				QSPI_PSMKR1			
7EF918	QSPI_AR1	QSPI_AR2	QSPI_AR3	QSPI_AR4	QSPI_ABR1	QSPI_ABR2	QSPI_ABR3	QSPI_ABR4
7EF910	QSPI_DLR1	QSPI_DLR2			QSPI_CCR1	QSPI_CCR2	QSPI_CCR3	QSPI_CCR4
7EF908	QSPI_FCR							
7EF900	QSPI_CR1	QSPI_CR2	QSPI_CR3	QSPI_CR4	QSPI_DCR1	QSPI_DCR2	QSPI_SR1	QSPI_SR2

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
7EF8F8	PWMD_CCR6L	PWMD_CCR7H	PWMD_CCR7L	PWMD_CCR8H	PWMD_CCR8L	PWMD_BKR	PWMD_DTR	PWMD_OISR
7EF8F0	PWMD_PSCRH	PWMD_PSCRL	PWMD_ARRH	PWMD_ARRL	PWMD_RCR	PWMD_CCR5H	PWMD_CCR5L	PWMD_CCR6H
7EF8E8	PWMD_CCMR1	PWMD_CCMR2	PWMD_CCMR3	PWMD_CCMR4	PWMD_CCER1	PWMD_CCER2	PWMD_CNTRH	PWMD_CNTRL
7EF8E0	PWMD_CR1	PWMD_CR2	PWMD_SMCR	PWMD_ETR	PWMD_IER	PWMD_SR1	PWMD_SR2	PWMD_EGR
7EF8D8	PWMC_CCR2L	PWMC_CCR3H	PWMC_CCR3L	PWMC_CCR4H	PWMC_CCR4L	PWMC_BKR	PWMC_DTR	PWMC_OISR
7EF8D0	PWMC_PSCRH	PWMC_PSCRL	PWMC_ARRH	PWMC_ARRL	PWMC_RCR	PWMC_CCR1H	PWMC_CCR1L	PWMC_CCR2H
7EF8C8	PWMC_CCMR1	PWMC_CCMR2	PWMC_CCMR3	PWMC_CCMR4	PWMC_CCER1	PWMC_CCER2	PWMC_CNTRH	PWMC_CNTRL
7EF8C0	PWMC_CR1	PWMC_CR2	PWMC_SMCR	PWMC_ETR	PWMC_IER	PWMC_SR1	PWMC_SR2	PWMC_EGR
7EF8B8	PWMC_PS2	PWMC_RCRH	PWMD_RCRH					
7EF8B0	PWMC_ETRPS	PWMC_ENO	PWMC_PS	PWMC_IOAUX	PWMD_ETRPS	PWMD_ENO	PWMD_PS	PWMD_IOAUX
7EF8A8	DMA_I2S2R_RXAH	DMA_I2S2R_RXAL						
7EF8A0	DMA_I2S2R_CFG	DMA_I2S2R_CR	DMA_I2S2R_STA		DMA_I2S2R_AMTH	DMA_I2S2R_AMT	DMA_I2S2R_DONEH	DMA_I2S2R_DONE
7EF898	DMA_I2S2T_TXAH	DMA_I2S2T_TXAL					DMA_I2S2_ITVH	DMA_I2S2_ITVL
7EF890	DMA_I2S2T_CFG	DMA_I2S2T_CR	DMA_I2S2T_STA		DMA_I2S2T_AMTH	DMA_I2S2T_AMT	DMA_I2S2T_DONEH	DMA_I2S2T_DONE
7EF888	DMA_I2C2R_RXAH	DMA_I2C2R_RXAL						
7EF880	DMA_I2C2R_CFG	DMA_I2C2R_CR	DMA_I2C2R_STA		DMA_I2C2R_AMTH	DMA_I2C2R_AMT	DMA_I2C2R_DONEH	DMA_I2C2R_DONE
7EF878	DMA_I2C2T_TXAH	DMA_I2C2T_TXAL		DMA_I2C2_CR	DMA_I2C2_ST1	DMA_I2C2_ST2	DMA_I2C2_ITVH	DMA_I2C2_ITVL
7EF870	DMA_I2C2T_CFG	DMA_I2C2T_CR	DMA_I2C2T_STA		DMA_I2C2T_AMTH	DMA_I2C2T_AMT	DMA_I2C2T_DONEH	DMA_I2C2T_DONE
7EF868	I2C2MSAUX	I2C2PSCR		I2C2TOTE	I2C2TOCR	I2C2TOSR	I2C2TOTH	I2C2TOTL
7EF860	I2C2CFG	I2C2MSCR	I2C2MSST	I2C2SLCR	I2C2SLST	I2C2SLADR	I2C2TxD	I2C2RxD
7EF838	DMA_SPI3_TXAH	DMA_SPI3_TXAL	DMA_SPI3_RXAH	DMA_SPI3_RXAL			DMA_SPI3_ITVH	DMA_SPI3_ITVL
7EF830	DMA_SPI3_CFG	DMA_SPI3_CR	DMA_SPI3_STA	DMA_SPI3_CFG2	DMA_SPI3_AMTH	DMA_SPI3_AMT	DMA_SPI3_DONEH	DMA_SPI3_DONE
7EF828	SPI3_CLKDIV	HSSPI3_CFG	HSSPI3_CFG2	HSSPI3_STA	HSSPI3_PSCR	SPI3_INTR		
7EF820	SPI3CTL	SPI3STAT	SPI3DAT	SPI3TOTE	SPI3TOCR	SPI3TOSR	SPI3TOTH	SPI3TOTL
7EF818	DMA_SPI2_TXAH	DMA_SPI2_TXAL	DMA_SPI2_RXAH	DMA_SPI2_RXAL			DMA_SPI2_ITVH	DMA_SPI2_ITVL
7EF810	DMA_SPI2_CFG	DMA_SPI2_CR	DMA_SPI2_STA	DMA_SPI2_CFG2	DMA_SPI2_AMTH	DMA_SPI2_AMT	DMA_SPI2_DONEH	DMA_SPI2_DONE
7EF808	SPI2_CLKDIV	HSSPI2_CFG	HSSPI2_CFG2	HSSPI2_STA	HSSPI2_PSCR	SPI2_INTR		
7EF800	SPI2CTL	SPI2STAT	SPI2DAT	SPI2TOTE	SPI2TOCR	SPI2TOSR	SPI2TOTH	SPI2TOTL

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
7EF7F8	DMA_UR8R_RXAH	DMA_UR8R_RXAL				S8CFG	S8ADDR	S8ADEN
7EF7F0	DMA_UR8R_CFG	DMA_UR8R_CR	DMA_UR8R_STA		DMA_UR8R_AMTH	DMA_UR8R_AMT	DMA_UR8R_DONEH	DMA_UR8R_DONE
7EF7E8	DMA_UR8T_TXAH	DMA_UR8T_TXAL					DMA_UR8_ITVH	DMA_UR8_ITVL
7EF7E0	DMA_UR8T_CFG	DMA_UR8T_CR	DMA_UR8T_STA		DMA_UR8T_AMTH	DMA_UR8T_AMT	DMA_UR8T_DONEH	DMA_UR8T_DONE
7EF7D8	USART8CR1	USART8CR2	USART8CR3	USART8CR4	USART8CR8	USART8GTR	USART8BRH	USART8BRL
7EF7D0	S8CON	S8BUF	S8INTR	UR8TOTE	UR8TOCR	UR8TOSR	UR8TOTH	UR8TOTL
7EF7C8	DMA_UR7R_RXAH	DMA_UR7R_RXAL				S7CFG	S7ADDR	S7ADEN
7EF7C0	DMA_UR7R_CFG	DMA_UR7R_CR	DMA_UR7R_STA		DMA_UR7R_AMTH	DMA_UR7R_AMT	DMA_UR7R_DONEH	DMA_UR7R_DONE
7EF7B8	DMA_UR7T_TXAH	DMA_UR7T_TXAL					DMA_UR7_ITVH	DMA_UR7_ITVL
7EF7B0	DMA_UR7T_CFG	DMA_UR7T_CR	DMA_UR7T_STA		DMA_UR7T_AMTH	DMA_UR7T_AMT	DMA_UR7T_DONEH	DMA_UR7T_DONE
7EF7A8	USART7CR1	USART7CR2	USART7CR3	USART7CR4	USART7CR7	USART7GTR	USART7BRH	USART7BRL
7EF7A0	S7CON	S7BUF	S7INTR	UR7TOTE	UR7TOCR	UR7TOSR	UR7TOTH	UR7TOTL
7EF798	DMA_UR6R_RXAH	DMA_UR6R_RXAL				S6CFG	S6ADDR	S6ADEN
7EF790	DMA_UR6R_CFG	DMA_UR6R_CR	DMA_UR6R_STA		DMA_UR6R_AMTH	DMA_UR6R_AMT	DMA_UR6R_DONEH	DMA_UR6R_DONE
7EF788	DMA_UR6T_TXAH	DMA_UR6T_TXAL					DMA_UR6_ITVH	DMA_UR6_ITVL
7EF780	DMA_UR6T_CFG	DMA_UR6T_CR	DMA_UR6T_STA		DMA_UR6T_AMTH	DMA_UR6T_AMT	DMA_UR6T_DONEH	DMA_UR6T_DONE
7EF778	USART6CR1	USART6CR2	USART6CR3	USART6CR4	USART6CR6	USART6GTR	USART6BRH	USART6BRL
7EF770	S6CON	S6BUF	S6INTR	UR6TOTE	UR6TOCR	UR6TOSR	UR6TOTH	UR6TOTL
7EF768	DMA_UR5R_RXAH	DMA_UR5R_RXAL				S5CFG	S5ADDR	S5ADEN
7EF760	DMA_UR5R_CFG	DMA_UR5R_CR	DMA_UR5R_STA		DMA_UR5R_AMTH	DMA_UR5R_AMT	DMA_UR5R_DONEH	DMA_UR5R_DONE
7EF758	DMA_UR5T_TXAH	DMA_UR5T_TXAL					DMA_UR5_ITVH	DMA_UR5_ITVL
7EF750	DMA_UR5T_CFG	DMA_UR5T_CR	DMA_UR5T_STA		DMA_UR5T_AMTH	DMA_UR5T_AMT	DMA_UR5T_DONEH	DMA_UR5T_DONE
7EF748	USART5CR1	USART5CR2	USART5CR3	USART5CR4	USART5CR5	USART5GTR	USART5BRH	USART5BRL
7EF740	S5CON	S5BUF	S5INTR	UR5TOTE	UR5TOCR	UR5TOSR	UR5TOTH	UR5TOTL
7EF728	P8CLRB	P9CLRB	PACLRB	PBCLRB				
7EF720	P8SETB	P9SETB	PASETB	PBSETB				
7EF718	P8M1	P9M1	PAM1	PBM1				
7EF710	P8M0	P9M0	PAM0	PBM0				
7EF708	P8PIN	P9PIN	PAPIN	PBPIN				
7EF700	P8OUT	P9OUT	PAOUT	PBOUT				

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
7EF6F8	PWMF_CCR6L	PWMF_CCR7H	PWMF_CCR7L	PWMF_CCR8H	PWMF_CCR8L	PWMF_BKR	PWMF_DTR	PWMF_OISR
7EF6F0	PWMF_PSCRH	PWMF_PSCRL	PWMF_ARRH	PWMF_ARRL	PWMF_RCR	PWMF_CCR5H	PWMF_CCR5L	PWMF_CCR6H
7EF6E8	PWMF_CCMR1	PWMF_CCMR2	PWMF_CCMR3	PWMF_CCMR4	PWMF_CCER1	PWMF_CCER2	PWMF_CNTRH	PWMF_CNTRL
7EF6E0	PWMF_CR1	PWMF_CR2	PWMF_SMCR	PWMF_ETR	PWMF_IER	PWMF_SR1	PWMF_SR2	PWMF_EGR
7EF6D8	PWME_CCR2L	PWME_CCR3H	PWME_CCR3L	PWME_CCR4H	PWME_CCR4L	PWME_BKR	PWME_DTR	PWME_OISR
7EF6D0	PWME_PSCRH	PWME_PSCRL	PWME_ARRH	PWME_ARRL	PWME_RCR	PWME_CCR1H	PWME_CCR1L	PWME_CCR2H
7EF6C8	PWME_CCMR1	PWME_CCMR2	PWME_CCMR3	PWME_CCMR4	PWME_CCER1	PWME_CCER2	PWME_CNTRH	PWME_CNTRL
7EF6C0	PWME_CR1	PWME_CR2	PWME_SMCR	PWME_ETR	PWME_IER	PWME_SR1	PWME_SR2	PWME_EGR
7EF6B8	PWME_PS2	PWME_RCRH	PWMF_RCRH					
7EF6B0	PWME_ETRPS	PWME_ENO	PWME_PS	PWME_IOAUX	PWMF_ETRPS	PWMF_ENO	PWMF_PS	PWMF_IOAUX
7EF6A8	PWME_DER	PWME_DBA	PWME_DBL	PWME_DMACR				
7EF6A0	PWME_CCR5H	PWME_CCR5L	PWME_CCR5X	PWME_CCR6H	PWME_CCR6L			
7EF698	PWME_CCMR1X	PWME_CCMR2X	PWME_CCMR3X	PWME_CCMR4X	PWME_CCMR5	PWME_CCMR5X	PWME_CCMR6	PWME_CCMR6X
7EF690	PWME_ENO2	PWME_IOAUX2	PWME_CR3	PWME_SR3	PWME_CCER3			
7EF688	HSPWME_CFG	HSPWME_ADR	HSPWME_DAT	PWME_CLKDIV	HSPWMF_CFG	HSPWMF_ADR	HSPWMF_DAT	PWMF_CLKDIV
7EF680	DMA_PWMER_RXAH	DMA_PWMER_RXAL						
7EF678	DMA_PWMER_CFG	DMA_PWMER_CR	DMA_PWMER_STA		DMA_PWMER_AMTH	DMA_PWMER_AMT	DMA_PWMER_DONEH	DMA_PWMER_DONE
7EF670	DMA_PWMET_TXAH	DMA_PWMET_TXAL					DMA_PWME_ITVH	DMA_PWME_ITVL
7EF668	DMA_PWMET_CFG	DMA_PWMET_CR	DMA_PWMET_STA		DMA_PWMET_AMTH	DMA_PWMET_AMT	DMA_PWMET_DONEH	DMA_PWMET_DONE
7EF640	RSTCR6	RSTCR7	RSTCR8	RSTCR9	RSTCR10	RSTCR11	RSTCR12	RSTCR13
7EF638	DMA_DAC2_TXAH	DMA_DAC2_TXAL		DMA_DAC2_CR1			DMA_DAC2_ITVH	DMA_DAC2_ITVL
7EF630	DMA_DAC2_CFG	DMA_DAC2_CR	DMA_DAC2_STA		DMA_DAC2_AMTH	DMA_DAC2_AMT	DMA_DAC2_DONEH	DMA_DAC2_DONE
7EF628	DAC2_CR	DAC2_STA	DAC2_DATH	DAC2_DATL	DAC2_DIVH	DAC2_DIVL		
7EF620	DMA_DAC1_TXAH	DMA_DAC1_TXAL		DMA_DAC1_CR1			DMA_DAC1_ITVH	DMA_DAC1_ITVL
7EF618	DMA_DAC1_CFG	DMA_DAC1_CR	DMA_DAC1_STA		DMA_DAC1_AMTH	DMA_DAC1_AMT	DMA_DAC1_DONEH	DMA_DAC1_DONE
7EF610	DAC1_CR	DAC1_STA	DAC1_DATH	DAC1_DATL	DAC1_DIVH	DAC1_DIVL		
7EF608	PGA3_CR1	PGA3_CR2			PGA4_CR1	PGA4_CR2		
7EF600	PGA1_CR1	PGA1_CR2			PGA2_CR1	PGA2_CR2		

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
7EF5F8	IAP_XDATA8	IAP_XDATA9	IAP_XDATAA	IAP_XDATAB	IAP_XDATAC	IAP_XDATAD	IAP_XDATAE	IAP_XDATAF
7EF5F0	IAP_XDATA0	IAP_XDATA1	IAP_XDATA2	IAP_XDATA3	IAP_XDATA4	IAP_XDATA5	IAP_XDATA6	IAP_XDATA7
7EF558	DMA_CAN2R_RXAH	DMA_CAN2R_RXAL		DMA_CAN2R_FRM	DMA_CAN2R_FCNTH	DMA_CAN2R_FCNTL		
7EF550	DMA_CAN2R_CFG	DMA_CAN2R_CR	DMA_CAN2R_STA	CAN2_AUX_CR	DMA_CAN2R_AMTH	DMA_CAN2R_AMT	DMA_CAN2R_DONEH	DMA_CAN2R_DONE
7EF548	DMA_CAN2T_TXAH	DMA_CAN2T_TXAL		DMA_CAN2T_FRM	DMA_CAN2T_FCNTH	DMA_CAN2T_FCNTL	DMA_CAN2_ITVH	DMA_CAN2_ITVL
7EF540	DMA_CAN2T_CFG	DMA_CAN2T_CR	DMA_CAN2T_STA		DMA_CAN2T_AMTH	DMA_CAN2T_AMT	DMA_CAN2T_DONEH	DMA_CAN2T_DONE
7EF538	DMA_CAN1R_RXAH	DMA_CAN1R_RXAL		DMA_CAN1R_FRM	DMA_CAN1R_FCNTH	DMA_CAN1R_FCNTL		
7EF530	DMA_CAN1R_CFG	DMA_CAN1R_CR	DMA_CAN1R_STA	CAN1_AUX_CR	DMA_CAN1R_AMTH	DMA_CAN1R_AMT	DMA_CAN1R_DONEH	DMA_CAN1R_DONE
7EF528	DMA_CAN1T_TXAH	DMA_CAN1T_TXAL		DMA_CAN1T_FRM	DMA_CAN1T_FCNTH	DMA_CAN1T_FCNTL	DMA_CAN1_ITVH	DMA_CAN1_ITVL
7EF520	DMA_CAN1T_CFG	DMA_CAN1T_CR	DMA_CAN1T_STA		DMA_CAN1T_AMTH	DMA_CAN1T_AMT	DMA_CAN1T_DONEH	DMA_CAN1T_DONE
7EF518	P8OD	P9OD	PAOD	PBOD				
7EF510	P0OD	P1OD	P2OD	P3OD	P4OD	P5OD	P6OD	P7OD
7EF508	P8OS	P9OS	PAOS	PBOS				
7EF500	P0OS	P1OS	P2OS	P3OS	P4OS	P5OS	P6OS	P7OS

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
7EF4F8	IO_WUP_PRD	CPU_WUP_PRD	CPU_STOP_CNTH	CPU_STOP_CNTL				
7EF4F0	CPU_CRR_CFG	CPU_CRR_KEY0	CPU_CRR_KEY1	CPU_CRR_KEY2	CRR_READ_SELH	CRR_READ_SELL	CRR_READ_DATA	
7EF4E8	P8IM2	P9IM2	PAIM2	PBIM2				
7EF4E0	P0IM2	P1IM2	P2IM2	P3IM2	P4IM2	P5IM2	P6IM2	P7IM2
7EF4C8	CAN_TT_WTRIG0	CAN_TT_WTRIG1						
7EF4C0	CAN_REF_MSG0	CAN_REF_MSG1	CAN_REF_MSG2	CAN_REF_MSG3	CAN_TRIG_CFG0	CAN_TRIG_CFG1	CAN_TT_CFG0	CAN_TT_CFG1
7EF4B8	CAN_ACF0	CAN_ACF1	CAN_ACF2	CAN_ACF3	CAN_VER0	CAN_VER1	CAN_TBSLOT	CAN_TTCFG
7EF4B0	CAN_EALCAP	CAN_TDC	CAN_RECNT	CAN_TECNT	CAN_ACFCTRL	CAN_TIMECFG	CAN_ACF_EN0	CAN_ACF_EN1
7EF4A8	CAN_S_SEG1	CAN_S_SEG2	CAN_S_SJW	CAN_S_PRESC	CAN_F_SEG1	CAN_F_SEG2	CAN_F_SJW	CAN_F_PRESC
7EF4A0	CAN_CFG_STAT	CAN_TCMD	CAN_TCTRL	CAN_RCTRL	CAN_RTIE	CAN_RTIF	CAN_ERRINT	CAN_LIMIT
7EF498	CAN_TTS00	CAN_TTS01	CAN_TTS02	CAN_TTS03	CAN_TTS04	CAN_TTS05	CAN_TTS06	CAN_TTS07
7EF490	CAN_TBUF64	CAN_TBUF65	CAN_TBUF66	CAN_TBUF67	CAN_TBUF68	CAN_TBUF69	CAN_TBUF70	CAN_TBUF71
7EF488	CAN_TBUF56	CAN_TBUF57	CAN_TBUF58	CAN_TBUF59	CAN_TBUF60	CAN_TBUF61	CAN_TBUF62	CAN_TBUF63
7EF480	CAN_TBUF48	CAN_TBUF49	CAN_TBUF50	CAN_TBUF51	CAN_TBUF52	CAN_TBUF53	CAN_TBUF54	CAN_TBUF55
7EF478	CAN_TBUF40	CAN_TBUF41	CAN_TBUF42	CAN_TBUF43	CAN_TBUF44	CAN_TBUF45	CAN_TBUF46	CAN_TBUF47
7EF470	CAN_TBUF32	CAN_TBUF33	CAN_TBUF34	CAN_TBUF35	CAN_TBUF36	CAN_TBUF37	CAN_TBUF38	CAN_TBUF39
7EF468	CAN_TBUF24	CAN_TBUF25	CAN_TBUF26	CAN_TBUF27	CAN_TBUF28	CAN_TBUF29	CAN_TBUF30	CAN_TBUF31
7EF460	CAN_TBUF16	CAN_TBUF17	CAN_TBUF18	CAN_TBUF19	CAN_TBUF20	CAN_TBUF21	CAN_TBUF22	CAN_TBUF23
7EF458	CAN_TBUF08	CAN_TBUF09	CAN_TBUF10	CAN_TBUF11	CAN_TBUF12	CAN_TBUF13	CAN_TBUF14	CAN_TBUF15
7EF450	CAN_TBUF00	CAN_TBUF01	CAN_TBUF02	CAN_TBUF03	CAN_TBUF04	CAN_TBUF05	CAN_TBUF06	CAN_TBUF07
7EF448	CAN_RBUF72	CAN_RBUF73	CAN_RBUF74	CAN_RBUF75	CAN_RBUF76	CAN_RBUF77	CAN_RBUF78	CAN_RBUF79
7EF440	CAN_RBUF64	CAN_RBUF65	CAN_RBUF66	CAN_RBUF67	CAN_RBUF68	CAN_RBUF69	CAN_RBUF70	CAN_RBUF71
7EF438	CAN_RBUF56	CAN_RBUF57	CAN_RBUF58	CAN_RBUF59	CAN_RBUF60	CAN_RBUF61	CAN_RBUF62	CAN_RBUF63
7EF430	CAN_RBUF48	CAN_RBUF49	CAN_RBUF50	CAN_RBUF51	CAN_RBUF52	CAN_RBUF53	CAN_RBUF54	CAN_RBUF55
7EF428	CAN_RBUF40	CAN_RBUF41	CAN_RBUF42	CAN_RBUF43	CAN_RBUF44	CAN_RBUF45	CAN_RBUF46	CAN_RBUF47
7EF420	CAN_RBUF32	CAN_RBUF33	CAN_RBUF34	CAN_RBUF35	CAN_RBUF36	CAN_RBUF37	CAN_RBUF38	CAN_RBUF39
7EF418	CAN_RBUF24	CAN_RBUF25	CAN_RBUF26	CAN_RBUF27	CAN_RBUF28	CAN_RBUF29	CAN_RBUF30	CAN_RBUF31
7EF410	CAN_RBUF16	CAN_RBUF17	CAN_RBUF18	CAN_RBUF19	CAN_RBUF20	CAN_RBUF21	CAN_RBUF22	CAN_RBUF23
7EF408	CAN_RBUF08	CAN_RBUF09	CAN_RBUF10	CAN_RBUF11	CAN_RBUF12	CAN_RBUF13	CAN_RBUF14	CAN_RBUF15
7EF400	CAN_RBUF00	CAN_RBUF01	CAN_RBUF02	CAN_RBUF03	CAN_RBUF04	CAN_RBUF05	CAN_RBUF06	CAN_RBUF07

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
7EF3C8	CAN2_TT_WTRIG0	CAN2_TT_WTRIG1						
7EF3C0	CAN2_REF_MSG0	CAN2_REF_MSG1	CAN2_REF_MSG2	CAN2_REF_MSG3	CAN2_TRIG_CFG0	CAN2_TRIG_CFG1	CAN2_TT_CFG0	CAN2_TT_CFG1
7EF3B8	CAN2_ACF0	CAN2_ACF1	CAN2_ACF2	CAN2_ACF3	CAN2_VER0	CAN2_VER1	CAN2_TBSLOT	CAN2_TTCFG
7EF3B0	CAN2_EALCAP	CAN2_TDC	CAN2_RECNT	CAN2_TECNT	CAN2_ACFCTRL	CAN2_TIMECFG	CAN2_ACF_EN0	CAN2_ACF_EN1
7EF3A8	CAN2_S_SEG1	CAN2_S_SEG2	CAN2_S_SJW	CAN2_S_PRESC	CAN2_F_SEG1	CAN2_F_SEG2	CAN2_F_SJW	CAN2_F_PRESC
7EF3A0	CAN2_CFG_STAT	CAN2_TCMD	CAN2_TCTRL	CAN2_RCTRL	CAN2_RTIE	CAN2_RTIF	CAN2_ERRINT	CAN2_LIMIT
7EF398	CAN2_TTS00	CAN2_TTS01	CAN2_TTS02	CAN2_TTS03	CAN2_TTS04	CAN2_TTS05	CAN2_TTS06	CAN2_TTS07
7EF390	CAN2_TBUF64	CAN2_TBUF65	CAN2_TBUF66	CAN2_TBUF67	CAN2_TBUF68	CAN2_TBUF69	CAN2_TBUF70	CAN2_TBUF71
7EF388	CAN2_TBUF56	CAN2_TBUF57	CAN2_TBUF58	CAN2_TBUF59	CAN2_TBUF60	CAN2_TBUF61	CAN2_TBUF62	CAN2_TBUF63
7EF380	CAN2_TBUF48	CAN2_TBUF49	CAN2_TBUF50	CAN2_TBUF51	CAN2_TBUF52	CAN2_TBUF53	CAN2_TBUF54	CAN2_TBUF55
7EF378	CAN2_TBUF40	CAN2_TBUF41	CAN2_TBUF42	CAN2_TBUF43	CAN2_TBUF44	CAN2_TBUF45	CAN2_TBUF46	CAN2_TBUF47
7EF370	CAN2_TBUF32	CAN2_TBUF33	CAN2_TBUF34	CAN2_TBUF35	CAN2_TBUF36	CAN2_TBUF37	CAN2_TBUF38	CAN2_TBUF39
7EF368	CAN2_TBUF24	CAN2_TBUF25	CAN2_TBUF26	CAN2_TBUF27	CAN2_TBUF28	CAN2_TBUF29	CAN2_TBUF30	CAN2_TBUF31
7EF360	CAN2_TBUF16	CAN2_TBUF17	CAN2_TBUF18	CAN2_TBUF19	CAN2_TBUF20	CAN2_TBUF21	CAN2_TBUF22	CAN2_TBUF23
7EF358	CAN2_TBUF08	CAN2_TBUF09	CAN2_TBUF10	CAN2_TBUF11	CAN2_TBUF12	CAN2_TBUF13	CAN2_TBUF14	CAN2_TBUF15
7EF350	CAN2_TBUF00	CAN2_TBUF01	CAN2_TBUF02	CAN2_TBUF03	CAN2_TBUF04	CAN2_TBUF05	CAN2_TBUF06	CAN2_TBUF07
7EF348	CAN2_RBUF72	CAN2_RBUF73	CAN2_RBUF74	CAN2_RBUF75	CAN2_RBUF76	CAN2_RBUF77	CAN2_RBUF78	CAN2_RBUF79
7EF340	CAN2_RBUF64	CAN2_RBUF65	CAN2_RBUF66	CAN2_RBUF67	CAN2_RBUF68	CAN2_RBUF69	CAN2_RBUF70	CAN2_RBUF71
7EF338	CAN2_RBUF56	CAN2_RBUF57	CAN2_RBUF58	CAN2_RBUF59	CAN2_RBUF60	CAN2_RBUF61	CAN2_RBUF62	CAN2_RBUF63
7EF330	CAN2_RBUF48	CAN2_RBUF49	CAN2_RBUF50	CAN2_RBUF51	CAN2_RBUF52	CAN2_RBUF53	CAN2_RBUF54	CAN2_RBUF55
7EF328	CAN2_RBUF40	CAN2_RBUF41	CAN2_RBUF42	CAN2_RBUF43	CAN2_RBUF44	CAN2_RBUF45	CAN2_RBUF46	CAN2_RBUF47
7EF320	CAN2_RBUF32	CAN2_RBUF33	CAN2_RBUF34	CAN2_RBUF35	CAN2_RBUF36	CAN2_RBUF37	CAN2_RBUF38	CAN2_RBUF39
7EF318	CAN2_RBUF24	CAN2_RBUF25	CAN2_RBUF26	CAN2_RBUF27	CAN2_RBUF28	CAN2_RBUF29	CAN2_RBUF30	CAN2_RBUF31
7EF310	CAN2_RBUF16	CAN2_RBUF17	CAN2_RBUF18	CAN2_RBUF19	CAN2_RBUF20	CAN2_RBUF21	CAN2_RBUF22	CAN2_RBUF23
7EF308	CAN2_RBUF08	CAN2_RBUF09	CAN2_RBUF10	CAN2_RBUF11	CAN2_RBUF12	CAN2_RBUF13	CAN2_RBUF14	CAN2_RBUF15
7EF300	CAN2_RBUF00	CAN2_RBUF01	CAN2_RBUF02	CAN2_RBUF03	CAN2_RBUF04	CAN2_RBUF05	CAN2_RBUF06	CAN2_RBUF07

13.3 特殊功能寄存器列表及复位值（SFR：0x80-0xFF）

注：特殊功能寄存器初始值意义

0： 初始值为 0；

1： 初始值为 1；

n： 初始值与 ISP 下载时的硬件选项有关；

x： 不存在这个位，初始值不确定

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P0	P0 端口	80H	P07	P06	P05	P04	P03	P02	P01	P00	1111,1111
SP	堆栈指针	81H	SP[7:0]								0000,0111
DPL	数据指针（低字节）	82H	DPTR[7:0]								0000,0000
DPH	数据指针（高字节）	83H	DPTR[15:8]								0000,0000
DPXL	数据指针（最高字节）	84H	DPTR[23:16]								0000,0001
SPH	堆栈指针高字节	85H	SP[15:8]								0000,0000
DPUST	DSP32 状态寄存器	86H	Z	DBZ	SC[5:0]						0000,0000
PCON	电源控制寄存器	87H	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL	0011,0000
TCON	定时器控制寄存器	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	0000,0000
TMOD	定时器模式寄存器	89H	T1_GATE	T1_C/T	T1_M1	T1_M0	T0_GATE	T0_C/T	T0_M1	T0_M0	0000,0000
TL0	定时器 0 低 8 位寄存器	8AH	TMR0[7:0]								0000,0000
TL1	定时器 1 低 8 位寄存器	8BH	TMR1[7:0]								0000,0000
TH0	定时器 0 高 8 位寄存器	8CH	TMR0[15:8]								0000,0000
TH1	定时器 1 高 8 位寄存器	8DH	TMR1[15:8]								0000,0000
AUXR	辅助寄存器 1	8EH	T0x12	T1x12	UART_M0x6	T2R	T2_C/T	T2x12	EXTRAM	S1BRT	0000,0001
INTCLKO	中断与时钟输出控制寄存器	8FH	-	EX4	EX3	EX2	-	T2CLKO	T1CLKO	T0CLKO	x000,x000
P1	P1 端口	90H	P17	P16	P15	P14	P13	P12	P11	P10	1111,1111
P1M1	P1 口配置寄存器 1	91H	P17M1	P16M1	P15M1	P14M1	P13M1	P12M1	P11M1	P10M1	1111,1111
P1M0	P1 口配置寄存器 0	92H	P17M0	P16M0	P15M0	P14M0	P13M0	P12M0	P11M0	P10M0	0000,0000
P0M1	P0 口配置寄存器 1	93H	P07M1	P06M1	P05M1	P04M1	P03M1	P02M1	P01M1	P00M1	1111,1111
P0M0	P0 口配置寄存器 0	94H	P07M0	P06M0	P05M0	P04M0	P03M0	P02M0	P01M0	P00M0	0000,0000
P2M1	P2 口配置寄存器 1	95H	P27M1	P26M1	P25M1	P24M1	P23M1	P22M1	P21M1	P20M1	1111,1111
P2M0	P2 口配置寄存器 0	96H	P27M0	P26M0	P25M0	P24M0	P23M0	P22M0	P21M0	P20M0	0000,0000
AUXR2	辅助寄存器 2	97H	RAMTINY	CPUMODE	LINSEL	LIN2EN	CANEDIN	CAN2EN	CANEN	LINEN	0000,0000
SCON	串口 1 控制寄存器	98H	SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI	0000,0000
SBUF	串口 1 数据寄存器	99H	SBUF[7:0]								0000,0000
S2CON	串口 2 控制寄存器	9AH	S2SM0/FE	S2SM1	S2SM2	S2REN	S2TB8	S2RB8	S2TI	S2RI	0000,0000
S2BUF	串口 2 数据寄存器	9BH	S2BUF[7:0]								0000,0000
IRCBAND	IRC 频段选择检测	9DH	USBCKS	USBCKS2	-	-	-	-	SEL[1:0]		10xx,xxnn
LIRTRIM	IRC 频率微调寄存器	9EH	-	-	-	-	-	-	LIRTRIM[1:0]		xxxx,xxnn
IRTRIM	IRC 频率调整寄存器	9FH	IRTRIM[7:0]								nnnn,nnnn
P2	P2 端口	A0H	P27	P26	P25	P24	P23	P22	P21	P20	1111,1111
BUS_SPEED	总线速度控制寄存器	A1H	RW_S[1:0]					SPEED[2:0]			00xx,x000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P_SW1	外设端口切换寄存器 1	A2H	S1_S[1:0]		CAN_S[1:0]		SPI_S[1:0]		LIN_S[1:0]		0000,0000
IE	中断允许寄存器	A8H	EA	ELVD	EADC	ES	ET1	EX1	ET0	EX0	0000,0000
SADDR	串口 1 从机地址寄存器	A9H	SADDR[7:0]								0000,0000
WKTCL	掉电唤醒定时器低字节	AAH	WKTC[7:0]								1111,1111
WKTCH	掉电唤醒定时器高字节	ABH	WKTEN	WKTC[14:8]							0111,1111
S3CON	串口 3 控制寄存器	ACH	S3SM0/FE	S3SM1	S3SM2	S3REN	S3TB8	S3RB8	S3TI	S3RI	0000,0000
S3BUF	串口 3 数据寄存器	ADH	S3BUF[7:0]								0000,0000
TA	DPTR 时序控制寄存器	AEH	TA[7:0]								0000,0000
IE2	中断允许寄存器 2	AFH	EUSB	ET4	ET3	ES4	ES3	ET2	ESPI	ES2	0000,0000
P3	P3 端口	B0H	P37	P36	P35	P34	P33	P32	P31	P30	1111,1111
P3M1	P3 口配置寄存器 1	B1H	P37M1	P36M1	P35M1	P34M1	P33M1	P32M1	P31M1	P30M1	1111,1100
P3M0	P3 口配置寄存器 0	B2H	P37M0	P36M0	P35M0	P34M0	P33M0	P32M0	P31M0	P30M0	0000,0000
P4M1	P4 口配置寄存器 1	B3H	P47M1	P46M1	P45M1	P44M1	P43M1	P42M1	P41M1	P40M1	1111,1111
P4M0	P4 口配置寄存器 0	B4H	P47M0	P46M0	P45M0	P44M0	P43M0	P42M0	P41M0	P40M0	0000,0000
IP2	中断优先级控制寄存器 2	B5H	PUSB	PI2C	PCMP	PX4	PPWMB	PPWMA	PSPI	PS2	0000,0000
IP2H	高中断优先级控制寄存器 2	B6H	PUSBH	PI2CH	PCMPH	PX4H	PPWMBH	PPWMAH	PSPIH	PS2H	0000,0000
IPH	高中断优先级控制寄存器	B7H	-	PLVDH	PADCH	PSH	PT1H	PX1H	PT0H	PX0H	x000,0000
IP	中断优先级控制寄存器	B8H	-	PLVD	PADC	PS	PT1	PX1	PT0	PX0	x000,0000
SADEN	串口 1 从机地址屏蔽寄存器	B9H	SADEN[7:0]								0000,0000
P_SW2	外设端口切换寄存器 2	BAH	EAXFR	-	I2C_S[1:0]		-	S4_S[0]	S3_S[0]	S2_S[0]	0x00,x000
P_SW3	外设端口切换寄存器 2	BBH	I2S_S[1:0]		S2SPI_S[1:0]		S1SPI_S[1:0]		CAN2_S[1:0]		0000,0000
ADC_CONTR	ADC 控制寄存器	BCH	ADC_POWER	ADC_START	ADC_FLAG	ADC_EPWMT	ADC_CHS[3:0]				0000,0000
ADC_RES	ADC 转换结果高位寄存器	BDH	ADC_RES[15:8]								0000,0000
ADC_RESL	ADC 转换结果低位寄存器	BEH	ADC_RESL[7:0]								0000,0000
P_SW4	外设端口切换寄存器 4	BFH	-	-	-	-	-	-	QSPI_S[1:0]		xxxx,xx00
P4	P4 端口	C0H	P47	P46	P45	P44	P43	P42	P41	P40	1111,1111
WDT_CONTR	看门狗控制寄存器	C1H	WDT_FLAG	-	EN_WDT	CLR_WDT	IDL_WDT	WDT_PS[2:0]			0x00,0000
IAP_DATA	IAP 数据寄存器	C2H	IAP_DATA[7:0]								0000,0000
IAP_ADDRH	IAP 高地址寄存器	C3H	IAP_ADDR[15:8]								0000,0000
IAP_ADDRL	IAP 低地址寄存器	C4H	IAP_ADDR[7:0]								0000,0000
IAP_CMD	IAP 命令寄存器	C5H	-	-	-	-	CMD[3:0]				xxxx,0000
IAP_TRIG	IAP 触发寄存器	C6H	IAP_TRIG[7:0]								0000,0000
IAP_CONTR	IAP 控制寄存器	C7H	IAPEN	SWBS	SWRST	CMD_FAIL	SWBS2	-	-	-	0000,0xxx
P5	P5 端口	C8H	P57	P56	P55	P54	P53	P52	P51	P50	1111,1111
P5M1	P5 口配置寄存器 1	C9H	P57M1	P56M1	P55M1	P54M1	P53M1	P52M1	P51M1	P50M1	1111,1111
P5M0	P5 口配置寄存器 0	CAH	P57M0	P56M0	P55M0	P54M0	P53M0	P52M0	P51M0	P50M0	0000,0000
P6M1	P6 口配置寄存器 1	CBH	P67M1	P66M1	P65M1	P64M1	P63M1	P62M1	P61M1	P60M1	1111,1111
P6M0	P6 口配置寄存器 0	CCH	P67M0	P66M0	P65M0	P64M0	P63M0	P62M0	P61M0	P60M0	0000,0000
SPSTAT	SPI 状态寄存器	CDH	SPIF	WCOL	-	-	-	-	-	-	00xx,xxxx
SPCTL	SPI 控制寄存器	CEH	SSIG	SPEN	DORD	MSTR	CPOL	CPHA	SPR[1:0]		0000,0100
SPDAT	SPI 数据寄存器	CFH	SPDAT[7:0]								0000,0000
PSW	程序状态字寄存器	D0H	CY	AC	F0	RS1	RS0	OV	F1	P	0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PSW1	程序状态字 1 寄存器	D1H	CY	AC	N	RS1	RS0	OV	Z		0000,000x
DPUOP	DSP32 指令寄存器	D8H	OPCODE[7:0]								0000,0000
T4H	定时器 4 高字节	D2H	TMR4[15:8]								0000,0000
T4L	定时器 4 低字节	D3H	TMR4[7:0]								0000,0000
T3H	定时器 3 高字节	D4H	TMR3[15:8]								0000,0000
T3L	定时器 3 低字节	D5H	TMR3[7:0]								0000,0000
T2H	定时器 2 高字节	D6H	TMR2[15:8]								0000,0000
T2L	定时器 2 低字节	D7H	TMR2[7:0]								0000,0000
USBCLK	USB 时钟控制寄存器	DCH	ENCKM	PCKI[1:0]		CRE	TST_USB	TST_PHY	PHYTST[1:0]		0010,0000
T4T3M	定时器 4/3 控制寄存器	DDH	T4R	T4_C/T	T4x12	T4CLKO	T3R	T3_C/T	T3x12	T3CLKO	0000,0000
ADCCFG	ADC 配置寄存器	DEH	-	-	RESFMT	RESDIFF	SPEED[3:0]				xx00,0000
IP3	中断优先级控制寄存器 3	DFH	PPWMD	PPWMC	PI2C2	PI2S2	PI2S	PRTC	PS4	PS3	xxxx,0000
ACC	累加器	E0H	ACC[7:0]								0000,0000
P7M1	P7 口配置寄存器 1	E1H	P77M1	P76M1	P75M1	P74M1	P73M1	P72M1	P71M1	P70M1	1111,1111
P7M0	P7 口配置寄存器 0	E2H	P77M0	P76M0	P75M0	P74M0	P73M0	P72M0	P71M0	P70M0	0000,0000
DPS	DPTR 指针选择器	E3H	ID1	ID0	TSL	AU1	AU0	-	-	SEL	0000,0xx0
P6	P6 端口	E8H	P67	P66	P65	P64	P63	P62	P61	P60	1111,1111
WTST	程序读取等待控制寄存器	E9H	WTST[7:0]								0000,0111
CKCON	XRAM 控制寄存器	EAH	RAMEXE	-	-	T1M	T0M	CKCON[2:0]			0xx0,0111
MXAX	MOVX 扩展地址寄存器	EBH	MXAX[7:0]								0000,0001
USBDAT	USB 数据寄存器	ECH	USBDAT[7:0]								0000,0000
DMAIR	FMU DMA 指令寄存器	EDH	DMAIR[7:0]								0000,0000
IP3H	高中断优先级控制寄存器 3	EEH	PPWMDH	PPWMCH	PI2C2H	PI2S2H	PI2SH	PRTCH	PS4H	PS3H	xxxx,0000
AUXINTIF	扩展外部中断标志寄存器	EFH	-	INT4IF	INT3IF	INT2IF	-	T4IF	T3IF	T2IF	x000,x000
B	B 寄存器	F0H	B[7:0]								0000,0000
IP4	中断优先级控制寄存器 4	F2H	-	-	-	-	-	-	PPWMF	PPWME	xxxx,xx00
IP4H	高中断优先级控制寄存器 4	F3H	-	-	-	-	-	-	PPWMFH	PPWMEH	xxxx,xx00
USBCON	USB 控制寄存器	F4H	ENUSB	ENUSBRST	PS2M	PUEN	PDEN	DFREC	DP	DM	0000,0000
IAP_TPS	IAP 等待时间控制寄存器	F5H	IAP_TPS[7:0]								0000,0111
IAP_ADDRE	IAP 扩展高地址寄存器	F6H	IAP_ADDR[23:16]								1111,1111
ICHECR	CACHE 控制寄存器	F7H	CON	HIT	CLR					EN	000x,xxx0
P7	P7 端口	F8H	P77	P76	P75	P74	P73	P72	P71	P70	1111,1111
LINICR	LINBUS 中断控制寄存器	F9H	PLIN2H	LIN2IF	LIN2IE	PLIN2L	PLINH	LINIF	LINIE	PLINL	0000,0000
LINAR	LINBUS 地址寄存器	FAH	LINAR[7:0]								0000,0000
LINDR	LINBUS 数据寄存器	FBH	LINDR[7:0]								0000,0000
USBADR	USB 地址寄存器	FCH	BUSY	AUTORD	UADR[5:0]						0000,0000
S4CON	串口 4 控制寄存器	FDH	S4SM0/FE	S4SM1	S4SM2	S4REN	S4TB8	S4RB8	S4TI	S4RI	0000,0000
S4BUF	串口 4 数据寄存器	FEH	S4BUF[7:0]								0000,0000
RSTCFG	复位配置寄存器	FFH	-	ENLVR	-	P54RST	-	-	LVDS[1:0]		x0x0,xx00

13.4 扩展特殊功能寄存器列表（XFR：0x7EFE00-0x7EFEFF）

下列特殊功能寄存器为扩展 SFR（XFR），逻辑地址位于 XDATA 区域，访问前需要将 P_SW2 寄存器的最高位（EAXFR）置 1，然后使用 MOV @DRk, Rm 和 MOV Rm, @DRk 指令进行访问，例如：

```
MOV     A,#00H
MOV     WR6,#WORD0 CLKSEL       ; CLKSEL 可换为需要访问的寄存器
MOV     WR4,#WORD2 CLKSEL
MOV     @DR4,R11
```

和

```
MOV     WR6,#WORD0 CLKSEL       ; CLKSEL 可换为需要访问的寄存器
MOV     WR4,#WORD2 CLKSEL
MOV     R11,@DR4
```

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CLKSEL	时钟选择寄存器	7EFE00H	-	-	-	-	MCK2SEL[1:0]		MCKSEL[1:0]		xxxx,0000
CLKDIV	时钟分频寄存器	7EFE01H	CLKDIV[7:0]								nnnn,nnn
HIRCCR	内部高速振荡器控制寄存器	7EFE02H	ENHIRC	-	MCLKO_CKS[1:0]		-	-	-	HIRCST	1x00,xxx0
XOSCCR	外部晶振控制寄存器	7EFE03H	ENXOSC	XITYPE	GAIN	-	XCFILTER[1:0]		-	XOSCST	000x,00x0
IRC32KCR	内部低速振荡器控制寄存器	7EFE04H	ENIRC32K	-	-	-	-	-	-	IRC32KST	0xxx,xxx0
MCLKOCR	主时钟输出控制寄存器	7EFE05H	MCLKO_S	MCLKODIV[6:0]							0000,0000
IRCDB	内部高速振荡器去抖控制	7EFE06H	IRCDB[7:0]								1000,0000
IRC48MCR	内部 48M 振荡器控制寄存器	7EFE07H	ENIRC48M	-	-	-	-	-	-	IRC48MST	1xxx,xxx0
X32KCR	外部 32K 晶振控制寄存器	7EFE08H	ENX32K	GAIN32K	-	-	-	-	-	X32KST	00xx,xxx0
IRC48MATRIM	内部 48M 振荡器校准值	7EFE09H	-	-	-	TRIM[12:8]					xxxn,nnnn
IRC48MBTRIM	内部 48M 振荡器校准值	7EFE0AH	TRIM[7:0]								nnnn,nnnn
HPLLCR	高速 PLL 控制寄存器	7EFE0CH	ENHPLL	HSI2SCK[1:0]		HPLLCKS	HPLLDIV[3:0]				0000,0000
HPLLPDIV	高速 PLL 预分频寄存器	7EFE0DH	HPLL_PREDIV[7:0]								0000,0000
HPLL2CR	高速 PLL2 控制寄存器	7EFE0EH	ENHPLL2	HSIOCK[1:0]		HPLL2CKS	HPLL2DIV[3:0]				0000,0000
HPLL2PDIV	高速 PLL2 预分频寄存器	7EFE0FH	HPLL2_PREDIV[7:0]								0000,0000
P0PU	P0 口上拉电阻控制寄存器	7EFE10H	P0PU[7:0]								0000,0000
P1PU	P1 口上拉电阻控制寄存器	7EFE11H	P1PU[7:0]								0000,0000
P2PU	P2 口上拉电阻控制寄存器	7EFE12H	P2PU[7:0]								0000,0000
P3PU	P3 口上拉电阻控制寄存器	7EFE13H	P3PU[7:0]								0000,0000
P4PU	P4 口上拉电阻控制寄存器	7EFE14H	P4PU[7:0]								0000,0000
P5PU	P5 口上拉电阻控制寄存器	7EFE15H	P5PU[7:0]								0000,0000
P6PU	P6 口上拉电阻控制寄存器	7EFE16H	P6PU[7:0]								0000,0000
P7PU	P7 口上拉电阻控制寄存器	7EFE17H	P7PU[7:0]								0000,0000
P0NCS	P0 口施密特触发控制寄存器	7EFE18H	P0NCS[7:0]								0000,0000
P1NCS	P1 口施密特触发控制寄存器	7EFE19H	P1NCS[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P2NCS	P2 口施密特触发控制寄存器	7EFE1AH	P2NCS[7:0]								0000,0000
P3NCS	P3 口施密特触发控制寄存器	7EFE1BH	P3NCS[7:0]								0000,0000
P4NCS	P4 口施密特触发控制寄存器	7EFE1CH	P4NCS[7:0]								0000,0000
P5NCS	P5 口施密特触发控制寄存器	7EFE1DH	P5NCS[7:0]								0000,0000
P6NCS	P6 口施密特触发控制寄存器	7EFE1EH	P6NCS[7:0]								0000,0000
P7NCS	P7 口施密特触发控制寄存器	7EFE1FH	P7NCS[7:0]								0000,0000
P0SR	P0 口电平转换速率寄存器	7EFE20H	P0SR[7:0]								1111,1111
P1SR	P1 口电平转换速率寄存器	7EFE21H	P1SR[7:0]								1111,1111
P2SR	P2 口电平转换速率寄存器	7EFE22H	P2SR[7:0]								1111,1111
P3SR	P3 口电平转换速率寄存器	7EFE23H	P3SR[7:0]								1111,1111
P4SR	P4 口电平转换速率寄存器	7EFE24H	P4SR[7:0]								1111,1111
P5SR	P5 口电平转换速率寄存器	7EFE25H	P5SR[7:0]								1111,1111
P6SR	P6 口电平转换速率寄存器	7EFE26H	P6SR[7:0]								1111,1111
P7SR	P7 口电平转换速率寄存器	7EFE27H	P7SR[7:0]								1111,1111
P0DR	P0 口驱动电流控制寄存器	7EFE28H	P0DR[7:0]								1111,1111
P1DR	P1 口驱动电流控制寄存器	7EFE29H	P1DR[7:0]								1111,1111
P2DR	P2 口驱动电流控制寄存器	7EFE2AH	P2DR[7:0]								1111,1111
P3DR	P3 口驱动电流控制寄存器	7EFE2BH	P3DR[7:0]								1111,1111
P4DR	P4 口驱动电流控制寄存器	7EFE2CH	P4DR[7:0]								1111,1111
P5DR	P5 口驱动电流控制寄存器	7EFE2DH	P5DR[7:0]								1111,1111
P6DR	P6 口驱动电流控制寄存器	7EFE2EH	P6DR[7:0]								1111,1111
P7DR	P7 口驱动电流控制寄存器	7EFE2FH	P7DR[7:0]								1111,1111
P0IE	P0 口输入使能控制寄存器	7EFE30H	P0IE[7:0]								1111,1111
P1IE	P1 口输入使能控制寄存器	7EFE31H	P1IE[7:0]								1111,1111
P2IE	P2 口输入使能控制寄存器	7EFE32H	P2IE[7:0]								1111,1111
P3IE	P3 口输入使能控制寄存器	7EFE33H	P3IE[7:0]								1111,1111
P4IE	P4 口输入使能控制寄存器	7EFE34H	P4IE[7:0]								1111,1111
P5IE	P5 口输入使能控制寄存器	7EFE35H	P5IE[7:0]								1111,1111
P6IE	P6 口输入使能控制寄存器	7EFE36H	P6IE[7:0]								1111,1111
P7IE	P7 口输入使能控制寄存器	7EFE37H	P7IE[7:0]								1111,1111
P0PD	P0 口下拉电阻控制寄存器	7EFE40H	P0PD[7:0]								0000,0000
P1PD	P1 口下拉电阻控制寄存器	7EFE41H	P1PD[7:0]								0000,0000
P2PD	P2 口下拉电阻控制寄存器	7EFE42H	P2PD[7:0]								0000,0000
P3PD	P3 口下拉电阻控制寄存器	7EFE43H	P3PD[7:0]								0000,0000
P4PD	P4 口下拉电阻控制寄存器	7EFE44H	P4PD[7:0]								0000,0000
P5PD	P5 口下拉电阻控制寄存器	7EFE45H	P5PD[7:0]								0000,0000
P6PD	P6 口下拉电阻控制寄存器	7EFE46H	P6PD[7:0]								0000,0000
P7PD	P7 口下拉电阻控制寄存器	7EFE47H	P7PD[7:0]								0000,0000
P0BP	P0 口模式用户控制寄存器	7EFE48H	P0BP[7:0]								1111,1111
P1BP	P1 口模式用户控制寄存器	7EFE49H	P1BP[7:0]								1111,1111
P2BP	P2 口模式用户控制寄存器	7EFE4AH	P2BP[7:0]								1111,1111
P3BP	P3 口模式用户控制寄存器	7EFE4BH	P3BP[7:0]								1111,1111

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P4BP	P4 口模式用户控制寄存器	7EFE4CH	P4BP[7:0]								1111,1111
P5BP	P5 口模式用户控制寄存器	7EFE4DH	P5BP[7:0]								1111,1111
P6BP	P6 口模式用户控制寄存器	7EFE4EH	P6BP[7:0]								1111,1111
P7BP	P7 口模式用户控制寄存器	7EFE4FH	P7BP[7:0]								1111,1111
LCMIFCFG	LCM 接口配置寄存器	7EFE50H	LCMIFIE	-	LCMIFIP[1:0]		LCMIFDPS[1:0]		D16_D8	M68_I80	0x00,0000
LCMIFCFG2	LCM 接口配置寄存器 2	7EFE51H	-	LCMIFCPS[1:0]		SETUPT[2:0]			HOLDT[1:0]		x000,0000
LCMIFCR	LCM 接口控制寄存器	7EFE52H	ENLCMIF	-	-	-	-	CMD[2:0]			0xxx,x000
LCMIFSTA	LCM 接口状态寄存器	7EFE53H	-	-	-	-	-	-	-	LCMIFIF	xxxx,xxx0
LCMIFDATL	LCM 接口低字节数据	7EFE54H	LCMIFDAT[7:0]								0000,0000
LCMIFDATH	LCM 接口高字节数据	7EFE55H	LCMIFDAT[15:8]								0000,0000
LCMIFPSCR	LCM 接口时钟预分配寄存器	7EFE56H	LCMIFPSCR[7:0]								0000,0000
RTCCR	RTC 控制寄存器	7EFE60H	-	-	-	-	-	-	-	RUNRTC	xxxx,xxx0
RTCCFG	RTC 配置寄存器	7EFE61H	-	-	-	-	-	-	RTCCKS	SETRTC	xxxx,xx00
RTCIEN	RTC 中断使能寄存器	7EFE62H	EALAI	EDAYI	EHOURI	EMINI	ESECI	ESEC2I	ESEC8I	ESEC32I	0000,0000
RTCIF	RTC 中断请求寄存器	7EFE63H	ALAIF	DAYIF	HOURIF	MINIF	SECIF	SEC2IF	SEC8IF	SEC32IF	0000,0000
ALAHOUR	RTC 闹钟的小时值	7EFE64H	-	-	-	ALAHOUR[4:0]					xxx0,0000
ALAMIN	RTC 闹钟的分钟值	7EFE65H	-	-	ALAMIN[5:0]						xx00,0000
ALASEC	RTC 闹钟的秒值	7EFE66H	-	-	ALASEC[5:0]						xx00,0000
ALASSEC	RTC 闹钟的 1/128 秒值	7EFE67H	-	ALASSEC[6:0]							x000,0000
INIYEAR	RTC 年初始化	7EFE68H	-	INIYEAR[6:0]							x000,0000
INIMONTH	RTC 月初始化	7EFE69H	-	-	-	-	INIMONTH[3:0]				xxxx,0000
INIDAY	RTC 日初始化	7EFE6AH	-	-	-	INIDAY[4:0]					xxx0,0000
INIHOUR	RTC 小时初始化	7EFE6BH	-	-	-	INIHOUR[4:0]					xxx0,0000
INIMIN	RTC 分钟初始化	7EFE6CH	-	-	INIMIN[5:0]						xx00,0000
INISEC	RTC 秒初始化	7EFE6DH	-	-	INISEC[5:0]						xx00,0000
INISSEC	RTC1/128 秒初始化	7EFE6EH	-	INISSEC[6:0]							x000,0000
RTCWEEK	RTC 的星期值	7EFE6FH	-	-	-	-	-	RTCWEEK[2:0]			xxxx,xnnn
RTCYEAR	RTC 的年计数值	7EFE70H	-	RTCYEAR[6:0]							x000,0000
RTCMONTH	RTC 的月计数值	7EFE71H	-	-	-	-	RTCMONTH[4:0]				xxxx,0000
RTCDAY	RTC 的日计数值	7EFE72H	-	-	-	RTCDAY[5:0]					xxx0,0000
RTCHOUR	RTC 的小时计数值	7EFE73H	-	-	-	RTCHOUR[5:0]					xxx0,0000
RTCMIN	RTC 的分钟计数值	7EFE74H	-	-	RTCMIN[6:0]						xx00,0000
RTCSEC	RTC 的秒计数值	7EFE75H	-	-	RTCSEC[6:0]						xx00,0000
RTCSSEC	RTC 的 1/128 秒计数值	7EFE76H	-	RTCSSEC[7:0]							x000,0000
T11CR	定时器 T11 控制寄存器	7EFE78H	T11R	T11_C/T	T11CLKO	T11x12	T11CS[1:0]		ET11I	T11IF	0000,0000
T11PS	定时器 T11 时钟预分频寄存器	7EFE79H	T11PS[7:0]								0000,0000
T11H	定时器 T11 高字节	7EFE7AH	TMR11[15:8]								0000,0000
T11L	定时器 T11 低字节	7EFE7BH	TMR11[7:0]								0000,0000
I2CCFG	I²C 配置寄存器	7EFE80H	ENI2C	I2CMSSL	I2CMSSPEED[5:0]						0000,0000
I2CMSCR	I²C 主机控制寄存器	7EFE81H	EI2CMSI	-	-	-	I2CMSCMD[3:0]				0xxx,0000
I2CMSST	I²C 主机状态寄存器	7EFE82H	I2CMSBUSY	I2CMSIF	-	-	-	-	I2CMSACKI	I2CMSACKO	00xx,xx10
I2CSLCR	I²C 从机控制寄存器	7EFE83H	-	EI2CSTAI	EI2CRXI	EI2CTXI	EI2CSTOI	-	-	I2CSLRST	x000,0xx0

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
I2CSLST	I²C 从机状态寄存器	7EFE84H	I2CSLBUSY	I2CSTAIF	I2CRXIF	I2CTXIF	I2CSTOIF	I2CTXING	I2CSLACKI	I2CSLACKO	0000,0000
I2CSLADR	I²C 从机地址寄存器	7EFE85H	I2CSLADR[6:0]							I2CMA	0000,0000
I2CTXD	I²C 数据发送寄存器	7EFE86H	I2CTXD[7:0]								0000,0000
I2CRXD	I²C 数据接收寄存器	7EFE87H	I2CRXD[7:0]								0000,0000
I2CMSAUX	I²C 主机辅助控制寄存器	7EFE88H	-	-	-	-	-	-	-	I2CWDTA	xxxx,xxx0
I2CPSCR	I²C 时钟预分配寄存器	7EFE89H	I2CMSSPEED[13:6]								0000,0000
SPI_CLKDIV	SPI 时钟预分配寄存器	7EFE90H	SPI_CLKDIV[7:0]								0000,0000
PWMA_CLKDIV	PWMA 时钟预分配寄存器	7EFE91H	PWMA_CLKDIV[7:0]								0000,0000
PWMB_CLKDIV	PWMB 时钟预分配寄存器	7EFE92H	PWMB_CLKDIV[7:0]								0000,0000
TFPU_CLKDIV	TFPU 时钟预分配寄存器	7EFE93H	TFPU_CLKDIV[7:0]								0000,0000
I2S_CLKDIV	I2S 时钟预分配寄存器	7EFE94H	I2S_CLKDIV[7:0]								0000,0000
I2S2_CLKDIV	I2S2 时钟预分配寄存器	7EFE95H	I2S2_CLKDIV[7:0]								0000,0000
PWMC_CLKDIV	PWMC 时钟预分配寄存器	7EFE96H	PWMC_CLKDIV[7:0]								0000,0000
PWMD_CLKDIV	PWMD 时钟预分配寄存器	7EFE97H	PWMD_CLKDIV[7:0]								0000,0000
SPFUNC	辅助控制寄存器	7EFE98H	-	-	-	-	-	-	-	BKSWR	xxxx,xxx0
RSTFLAG	复位标志寄存器	7EFE99H	-	-	-	LVDRSTF	WDTRSTF	SWRSTF	ROMOVF	EXRSTF	xxx1,0100
RSTCR0	复位控制寄存器 0	7EFE9AH	-	-	-	-	RSTTM34	TSTTM2	-	RSTTM01	xxxx,00x0
RSTCR1	复位控制寄存器 1	7EFE9BH	-	-	-	-	RSTUR4	RSTUR3	RSTUR2	RSTUR1	xxxx,0000
RSTCR2	复位控制寄存器 2	7EFE9CH	RSTCAN2	RSTCAN	RSTLIN	RSTRTC	RSTPWMB	RSTPWMA	RSTI2C	RSTSPI	0000,0000
RSTCR3	复位控制寄存器 3	7EFE9DH	RSTFPU	RSTDMA	RSTLCM	RSTLCD	RSTLED	RSTTKS	RSTCMP	RSTADC	0000,0000
RSTCR4	复位控制寄存器 4	7EFE9EH	-	-	-	-	-	-	-	RSTMDU	xxxx,xxx0
RSTCR5	复位控制寄存器 5	7EFE9FH	-	-	-	-	-	-	-	-	xxxx,xxxx
TM0PS	定时器 0 时钟预分频寄存器	7EFEA0H	TM0PS[7:0]								0000,0000
TM1PS	定时器 1 时钟预分频寄存器	7EFEA1H	TM1PS[7:0]								0000,0000
TM2PS	定时器 2 时钟预分频寄存器	7EFEA2H	TM2PS[7:0]								0000,0000
TM3PS	定时器 3 时钟预分频寄存器	7EFEA3H	TM3PS[7:0]								0000,0000
TM4PS	定时器 4 时钟预分频寄存器	7EFEA4H	TM4PS[7:0]								0000,0000
ADCTIM	ADC 时序控制寄存器	7EFEA8H	CSSETUP	CSHOLD[1:0]		SMPDUTY[4:0]					0010,1010
INTXIT	INT2/3/4 中断模式选择寄存器	7EFEA9H	-	-	INT4IT[1:0]		INT3IT[1:0]		INT2IT[1:0]		xx00,0000
TIMERPS	定时器端口选择寄存器	7EFEACH	-	T17T18SEL	T11SEL[1:0]		T9T10SEL	T7T8SEL	T5T6SEL	T3T4SEL	x000,0000
ADCEXCFG	ADC 扩展配置寄存器	7EFEADH	ADCETR_PS[1:0]		ADCETRS [1:0]		-	CVTIMESEL[2:0]			0000,x000
ADCEXCFG2	ADC 扩展配置寄存器 2	7EFEAFH	TRIGON	-	PWMSEL[1:0]		ADC_TRIGSEL[3:0]				0x01,0000
PWMA_ETRPS	PWMA 的 ETR 选择寄存器	7EFEB0H	-	-	CMPAPS[1:0]		BRKAPS[1:0]		ETRAPS[1:0]		xx00,0000
PWMA_ENO	PWMA 输出使能控制	7EFEB1H	ENOA4N	ENOA4P	ENOA3N	ENOA3P	ENOA2N	ENOA2P	ENOA1N	ENOA1P	0000,0000
PWMA_PS	PWMA 输出脚选择寄存器	7EFEB2H	AC4PS[1:0]		AC3PS[1:0]		AC2PS[1:0]		AC1PS[1:0]		0000,0000
PWMA_IOAUX	PWMA 辅助寄存器	7EFEB3H	AUXA4N	AUXA4P	AUXA3N	AUXA3P	AUXA2N	AUXA2P	AUXA1N	AUXA1P	0000,0000
PWMB_ETRPS	PWMB 的 ETR 选择寄存器	7EFEB4H	-	-	CMPBPS[1:0]		BRKBPS[1:0]		ETRBPS[1:0]		xx00,0000
PWMB_ENO	PWMB 输出使能控制	7EFEB5H	-	ENOB8P	-	ENOB7P	-	ENOB6P	-	ENOB5P	x0x0,x0x0
PWMB_PS	PWMB 输出脚选择寄存器	7EFEB6H	BC8PS[1:0]		BC7PS[1:0]		BC6PS[1:0]		BC5PS[1:0]		0000,0000
PWMB_IOAUX	PWMB 辅助寄存器	7EFEB7H	-	AUXB8P	-	AUXB7P	-	AUXB6P	-	AUXB5P	x0x0,x0x0
PWMA_PS2	PWMA 输出脚选择寄存器 2	7EFEB8H	-	-	-	-	AC6PS[1:0]		AC5PS[1:0]		xxxx,0000
PWMA_RCRH	PWMA 重复计数器寄存器	7EFEB9H	REPA[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PWMB_RCRH	PWMB 重复计数器寄存器	7EFEBAH	REPB[15:8]								0000,0000
PWMA_CR1	PWMA 控制寄存器 1	7EFEC0H	ARPEA	CMSA[1:0]		DIRA	OPMA	URSA	UDISA	CENA	0000,0000
PWMA_CR2	PWMA 控制寄存器 2	7EFEC1H	-	MMSA[2:0]			-	COMSA	-	CCPCA	x000,x0x0
PWMA_SMCR	PWMA 从模式控制寄存器	7EFEC2H	MSMA	TSA[2:0]			-	SMSA[2:0]			0000,x000
PWMA_ETR	PWMA 外部触发寄存器	7EFEC3H	ETPA	ECEA	ETPSA[1:0]		ETFA[3:0]				0000,0000
PWMA_IER	PWMA 中断使能寄存器	7EFEC4H	BIEA	TIEA	COMIEA	CCA4IE	CCA3IE	CCA2IE	CCA1IE	UIEA	0000,0000
PWMA_SR1	PWMA 状态寄存器 1	7EFEC5H	BIFA	TIFA	COMIFA	CCA4IF	CCA3IF	CCA2IF	CCA1IF	UIFA	0000,0000
PWMA_SR2	PWMA 状态寄存器 2	7EFEC6H	-	-	-	CCA4OF	CCA3OF	CCA2OF	CCA1OF	-	xxx0,000x
PWMA_EGR	PWMA 事件发生寄存器	7EFEC7H	BGA	TGA	COMGA	CCA4G	CCA3G	CCA2G	CCA1G	UGA	0000,0000
PWMA_CCMR1	PWMA 捕获模式寄存器 1	7EFEC8H	OCA1CE	OCA1M[2:0]			OCA1PE	OCA1FE	CCA1S[1:0]		0000,0000
	PWMA 比较模式寄存器 1		ICA1F[3:0]				ICA1PSC[1:0]		CCA1S[1:0]		0000,0000
PWMA_CCMR2	PWMA 捕获模式寄存器 2	7EFEC9H	OCA2CE	OCA2M[2:0]			OCA2PE	OCA2FE	CCA2S[1:0]		0000,0000
	PWMA 比较模式寄存器 2		ICA2F[3:0]				ICA2PSC[1:0]		CCA2S[1:0]		0000,0000
PWMA_CCMR3	PWMA 捕获模式寄存器 3	7EFECAH	OCA3CE	OCA3M[2:0]			OCA3PE	OCA3FE	CCA3S[1:0]		0000,0000
	PWMA 比较模式寄存器 3		ICA3F[3:0]				ICA3PSC[1:0]		CCA3S[1:0]		0000,0000
PWMA_CCMR4	PWMA 捕获模式寄存器 4	7EFECBH	OCA4CE	OCA4M[2:0]			OCA4PE	OCA4FE	CCA4S[1:0]		0000,0000
	PWMA 比较模式寄存器 4		ICA4F[3:0]				ICA4PSC[1:0]		CCA4S[1:0]		0000,0000
PWMA_CCER1	PWMA 捕获比较使能寄存器 1	7EFECCH	CCA2NP	CCA2NE	CCA2P	CCA2E	CCA1NP	CCA1NE	CCA1P	CCA1E	0000,0000
PWMA_CCER2	PWMA 捕获比较使能寄存器 2	7EFECDH	CCA4NP	CCA4NE	CCA4P	CCA4E	CCA3NP	CCA3NE	CCA3P	CCA3E	0000,0000
PWMA_CNTRH	PWMA 计数器高字节	7EFECEH	CNTA[15:8]								0000,0000
PWMA_CNTRL	PWMA 计数器低字节	7EFECFH	CNTA[7:0]								0000,0000
PWMA_PSCRH	PWMA 预分频高字节	7EFED0H	PSC[15:8]								0000,0000
PWMA_PSCRL	PWMA 预分频低字节	7EFED1H	PSCA[7:0]								0000,0000
PWMA_ARRH	PWMA 自动重装寄存器高字节	7EFED2H	ARRA[15:8]								0000,0000
PWMA_ARRL	PWMA 自动重装寄存器低字节	7EFED3H	ARRA[7:0]								0000,0000
PWMA_RCR	PWMA 重复计数器寄存器	7EFED4H	REPA[7:0]								0000,0000
PWMA_CCR1H	PWMA 比较捕获寄存器 1 高位	7EFED5H	CCRA1[15:8]								0000,0000
PWMA_CCR1L	PWMA 比较捕获寄存器 1 低位	7EFED6H	CCRA1[7:0]								0000,0000
PWMA_CCR2H	PWMA 比较捕获寄存器 2 高位	7EFED7H	CCRA2[15:8]								0000,0000
PWMA_CCR2L	PWMA 比较捕获寄存器 2 低位	7EFED8H	CCRA2[7:0]								0000,0000
PWMA_CCR3H	PWMA 比较捕获寄存器 3 高位	7EFED9H	CCRA3[15:8]								0000,0000
PWMA_CCR3L	PWMA 比较捕获寄存器 3 低位	7EFEDAH	CCRA3[7:0]								0000,0000
PWMA_CCR4H	PWMA 比较捕获寄存器 4 高位	7EFEDBH	CCRA4[15:8]								0000,0000
PWMA_CCR4L	PWMA 比较捕获寄存器 4 低位	7EFEDCH	CCRA4[7:0]								0000,0000
PWMA_BKR	PWMA 刹车寄存器	7EFEDDH	MOEA	AOEA	BKPA	BKEA	OSSRA	OSSIA	LOCKA[1:0]		0000,0000
PWMA_DTR	PWMA 死区控制寄存器	7EFEDEH	DTGA[7:0]								0000,0000
PWMA_OISR	PWMA 输出空闲状态寄存器	7EFEDFH	OISA4N	OISA4	OISA3N	OISA3	OISA2N	OISA2	OISA1N	OISA1	0000,0000
PWMB_CR1	PWMB 控制寄存器 1	7EFEE0H	ARPEB	CMSB[1:0]		DIRB	OPMB	URSB	UDISB	CENB	0000,0000
PWMB_CR2	PWMB 控制寄存器 2	7EFEE1H	-	MMSB[2:0]			-	COMSB	-	CCPCB	x000,x0x0
PWMB_SMCR	PWMB 从模式控制寄存器	7EFEE2H	MSMB	TSB[2:0]			-	SMSB[2:0]			0000,x000
PWMB_ETR	PWMB 外部触发寄存器	7EFEE3H	ETPB	ECEB	ETPSB[1:0]		ETBF[3:0]				0000,0000
PWMB_IER	PWMB 中断使能寄存器	7EFEE4H	BIEB	TIEB	COMIEB	CCB8IE	CCB7IE	CCB6IE	CCB5IE	UIEB	0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PWMB_SR1	PWMB 状态寄存器 1	7EFEE5H	BIFB	TIFB	COMIFB	CCB8IF	CCB7IF	CCB6IF	CCB5IF	UIFB	0000,0000
PWMB_SR2	PWMB 状态寄存器 2	7EFEE6H	-	-	-	CCB8OF	CCB7OF	CCB6OF	CCB5OF	-	xxx0,000x
PWMB_EGR	PWMB 事件发生寄存器	7EFEE7H	BGB	TGB	COMGB	CCB8G	CCB7G	CCB6G	CCB5G	UGB	0000,0000
PWMB_CCMR1	PWMB 捕获模式寄存器 1	7EFEE8H	OCB5CE	OCB5M[2:0]			OCB5PE	OCB5FE	CCB5S[1:0]		0000,0000
	PWMB 比较模式寄存器 1		ICB5F[3:0]				ICB5PSC[1:0]		CCB5S[1:0]		0000,0000
PWMB_CCMR2	PWMB 捕获模式寄存器 2	7EFEE9H	OCB6CE	OCB6M[2:0]			OCB6PE	OCB6FE	CCB6S[1:0]		0000,0000
	PWMB 比较模式寄存器 2		ICB6F[3:0]				ICB6PSC[1:0]		CCB6S[1:0]		0000,0000
PWMB_CCMR3	PWMB 捕获模式寄存器 3	7EFEEAH	OCB7CE	OCB7M[2:0]			OCB7PE	OCB7FE	CCB7S[1:0]		0000,0000
	PWMB 比较模式寄存器 3		ICB7F[3:0]				ICB7PSC[1:0]		CCB7S[1:0]		0000,0000
PWMB_CCMR4	PWMB 捕获模式寄存器 4	7EFEEBH	OCB8CE	OCB8M[2:0]			OCB8PE	OCB8FE	CCB8S[1:0]		0000,0000
	PWMB 比较模式寄存器 4		ICB8F[3:0]				ICB8PSC[1:0]		CCB8S[1:0]		0000,0000
PWMB_CCER1	PWMB 捕获比较使能寄存器 1	7EFEECH	-	-	CCB6P	CCB6E	-	-	CCB5P	CCB5E	xx00,xx00
PWMB_CCER2	PWMB 捕获比较使能寄存器 2	7EFEEDH	-	-	CCB8P	CCB8E	-	-	CCB7P	CCB7E	xx00,xx00
PWMB_CNTRH	PWMB 计数器高字节	7EFEEEH	CNTB[15:8]								0000,0000
PWMB_CNTRL	PWMB 计数器低字节	7EFEEFH	CNTB[7:0]								0000,0000
PWMB_PSCRH	PWMB 预分频高字节	7EFEF0H	PSCB[15:8]								0000,0000
PWMB_PSCRL	PWMB 预分频低字节	7EFEF1H	PSCB[7:0]								0000,0000
PWMB_ARRH	PWMB 自动重装寄存器高字节	7EFEF2H	ARRB[15:8]								0000,0000
PWMB_ARRL	PWMB 自动重装寄存器低字节	7EFEF3H	ARRB[7:0]								0000,0000
PWMB_RCR	PWMB 重复计数器寄存器	7EFEF4H	REPB[7:0]								0000,0000
PWMB_CCR5H	PWMB 比较捕获寄存器 5 高位	7EFEF5H	CCRB5[15:8]								0000,0000
PWMB_CCR5L	PWMB 比较捕获寄存器 5 低位	7EFEF6H	CCRB5[7:0]								0000,0000
PWMB_CCR6H	PWMB 比较捕获寄存器 6 高位	7EFEF7H	CCRB6[15:8]								0000,0000
PWMB_CCR6L	PWMB 比较捕获寄存器 6 低位	7EFEF8H	CCRB6[7:0]								0000,0000
PWMB_CCR7H	PWMB 比较捕获寄存器 7 高位	7EFEF9H	CCRB7[15:8]								0000,0000
PWMB_CCR7L	PWMB 比较捕获寄存器 7 低位	7EFEFAH	CCRB7[7:0]								0000,0000
PWMB_CCR8H	PWMB 比较捕获寄存器 8 高位	7EFEFBH	CCRB8[15:8]								0000,0000
PWMB_CCR8L	PWMB 比较捕获寄存器 8 低位	7EFEFCH	CCRB8[7:0]								0000,0000
PWMB_BKR	PWMB 刹车寄存器	7EFEFDH	MOEB	AOEB	BKPB	BKEB	OSSRB	OSSIB	LOCKB[1:0]-		0000,0000
PWMB_DTR	PWMB 死区控制寄存器	7EFEFEH	DTGB[7:0]								0000,0000
PWMB_OISR	PWMB 输出空闲状态寄存器	7EFEFFH	-	OISB8	-	OISB7	-	OISB6	-	OISB5	x0x0,x0x0

13.5 扩展特殊功能寄存器列表（XFR：0x7EFD00-0x7EFDFF）

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P0INTE	P0 口中断使能寄存器	7EFD00H	P07INTE	P06INTE	P05INTE	P04INTE	P03INTE	P02INTE	P01INTE	P00INTE	0000,0000
P1INTE	P1 口中断使能寄存器	7EFD01H	P17INTE	P16INTE	P15INTE	P14INTE	P13INTE	P12INTE	P11INTE	P10INTE	0000,0000
P2INTE	P2 口中断使能寄存器	7EFD02H	P27INTE	P26INTE	P25INTE	P24INTE	P23INTE	P22INTE	P21INTE	P20INTE	0000,0000
P3INTE	P3 口中断使能寄存器	7EFD03H	P37INTE	P36INTE	P35INTE	P34INTE	P33INTE	P32INTE	P31INTE	P30INTE	0000,0000
P4INTE	P4 口中断使能寄存器	7EFD04H	P47INTE	P46INTE	P45INTE	P44INTE	P43INTE	P42INTE	P41INTE	P40INTE	0000,0000
P5INTE	P5 口中断使能寄存器	7EFD05H	P57INTE	P56INTE	P55INTE	P54INTE	P53INTE	P52INTE	P51INTE	P50INTE	0000,0000
P6INTE	P6 口中断使能寄存器	7EFD06H	P67INTE	P66INTE	P65INTE	P64INTE	P63INTE	P62INTE	P61INTE	P60INTE	0000,0000
P7INTE	P7 口中断使能寄存器	7EFD07H	P77INTE	P76INTE	P75INTE	P74INTE	P73INTE	P72INTE	P71INTE	P70INTE	0000,0000
P8INTE	P8 口中断使能寄存器	7EFD08H	P87INTE	P86INTE	P85INTE	P84INTE	P83INTE	P82INTE	P81INTE	P80INTE	0000,0000
P9INTE	P9 口中断使能寄存器	7EFD09H	P97INTE	P96INTE	P95INTE	P94INTE	P93INTE	P92INTE	P91INTE	P90INTE	0000,0000
PAINTE	PA 口中断使能寄存器	7EFD0AH	PA7INTE	PA6INTE	PA5INTE	PA4INTE	PA3INTE	PA2INTE	PA1INTE	PA0INTE	0000,0000
PBINTE	PB 口中断使能寄存器	7EFD0BH	PB7INTE	PB6INTE	PB5INTE	PB4INTE	PB3INTE	PB2INTE	PB1INTE	PB0INTE	0000,0000
P0INTF	P0 口中断标志寄存器	7EFD10H	P07INTF	P06INTF	P05INTF	P04INTF	P03INTF	P02INTF	P01INTF	P00INTF	0000,0000
P1INTF	P1 口中断标志寄存器	7EFD11H	P17INTF	P16INTF	P15INTF	P14INTF	P13INTF	P12INTF	P11INTF	P10INTF	0000,0000
P2INTF	P2 口中断标志寄存器	7EFD12H	P27INTF	P26INTF	P25INTF	P24INTF	P23INTF	P22INTF	P21INTF	P20INTF	0000,0000
P3INTF	P3 口中断标志寄存器	7EFD13H	P37INTF	P36INTF	P35INTF	P34INTF	P33INTF	P32INTF	P31INTF	P30INTF	0000,0000
P4INTF	P4 口中断标志寄存器	7EFD14H	P47INTF	P46INTF	P45INTF	P44INTF	P43INTF	P42INTF	P41INTF	P40INTF	0000,0000
P5INTF	P5 口中断标志寄存器	7EFD15H	P57INTF	P56INTF	P55INTF	P54INTF	P53INTF	P52INTF	P51INTF	P50INTF	0000,0000
P6INTF	P6 口中断标志寄存器	7EFD16H	P67INTF	P66INTF	P65INTF	P64INTF	P63INTF	P62INTF	P61INTF	P60INTF	0000,0000
P7INTF	P7 口中断标志寄存器	7EFD17H	P77INTF	P76INTF	P75INTF	P74INTF	P73INTF	P72INTF	P71INTF	P70INTF	0000,0000
P8INTF	P8 口中断标志寄存器	7EFD18H	P87INTF	P86INTF	P85INTF	P84INTF	P83INTF	P82INTF	P81INTF	P80INTF	0000,0000
P9INTF	P9 口中断标志寄存器	7EFD19H	P97INTF	P96INTF	P95INTF	P94INTF	P93INTF	P92INTF	P91INTF	P90INTF	0000,0000
PAINTF	PA 口中断标志寄存器	7EFD1AH	PA7INTF	PA6INTF	PA5INTF	PA4INTF	PA3INTF	PA2INTF	PA1INTF	PA0INTF	0000,0000
PBINTF	PB 口中断标志寄存器	7EFD1BH	PB7INTF	PB6INTF	PB5INTF	PB4INTF	PB3INTF	PB2INTF	PB1INTF	PB0INTF	0000,0000
P0IM0	P0 口中断模式寄存器 0	7EFD20H	P07IM0	P06IM0	P05IM0	P04IM0	P03IM0	P02IM0	P01IM0	P00IM0	0000,0000
P1IM0	P1 口中断模式寄存器 0	7EFD21H	P17IM0	P16IM0	P15IM0	P14IM0	P13IM0	P12IM0	P11IM0	P10IM0	0000,0000
P2IM0	P2 口中断模式寄存器 0	7EFD22H	P27IM0	P26IM0	P25IM0	P24IM0	P23IM0	P22IM0	P21IM0	P20IM0	0000,0000
P3IM0	P3 口中断模式寄存器 0	7EFD23H	P37IM0	P36IM0	P35IM0	P34IM0	P33IM0	P32IM0	P31IM0	P30IM0	0000,0000
P4IM0	P4 口中断模式寄存器 0	7EFD24H	P47IM0	P46IM0	P45IM0	P44IM0	P43IM0	P42IM0	P41IM0	P40IM0	0000,0000
P5IM0	P5 口中断模式寄存器 0	7EFD25H	P57IM0	P56IM0	P55IM0	P54IM0	P53IM0	P52IM0	P51IM0	P50IM0	0000,0000
P6IM0	P6 口中断模式寄存器 0	7EFD26H	P67IM0	P66IM0	P65IM0	P64IM0	P63IM0	P62IM0	P61IM0	P60IM0	0000,0000
P7IM0	P7 口中断模式寄存器 0	7EFD27H	P77IM0	P76IM0	P75IM0	P74IM0	P73IM0	P72IM0	P71IM0	P70IM0	0000,0000
P8IM0	P8 口中断模式寄存器 0	7EFD28H	P87IM0	P86IM0	P85IM0	P84IM0	P83IM0	P82IM0	P81IM0	P80IM0	0000,0000
P9IM0	P9 口中断模式寄存器 0	7EFD29H	P97IM0	P96IM0	P95IM0	P94IM0	P93IM0	P92IM0	P91IM0	P90IM0	0000,0000
PAIM0	PA 口中断模式寄存器 0	7EFD2AH	PA7IM0	PA6IM0	PA5IM0	PA4IM0	PA3IM0	PA2IM0	PA1IM0	PA0IM0	0000,0000
PBIM0	PB 口中断模式寄存器 0	7EFD2BH	PB7IM0	PB6IM0	PB5IM0	PB4IM0	PB3IM0	PB2IM0	PB1IM0	PB0IM0	0000,0000
P0IM1	P0 口中断模式寄存器 1	7EFD30H	P07IM1	P06IM1	P05IM1	P04IM1	P03IM1	P02IM1	P01IM1	P00IM1	0000,0000
P1IM1	P1 口中断模式寄存器 1	7EFD31H	P17IM1	P16IM1	P15IM1	P14IM1	P13IM1	P12IM1	P11IM1	P10IM1	0000,0000
P2IM1	P2 口中断模式寄存器 1	7EFD32H	P27IM1	P26IM1	P25IM1	P24IM1	P23IM1	P22IM1	P21IM1	P20IM1	0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P3IM1	P3 口中断模式寄存器 1	7EFD33H	P37IM1	P36IM1	P35IM1	P34IM1	P33IM1	P32IM1	P31IM1	P30IM1	0000,0000
P4IM1	P4 口中断模式寄存器 1	7EFD34H	P47IM1	P46IM1	P45IM1	P44IM1	P43IM1	P42IM1	P41IM1	P40IM1	0000,0000
P5IM1	P5 口中断模式寄存器 1	7EFD35H	P57IM1	P56IM1	P55IM1	P54IM1	P53IM1	P52IM1	P51IM1	P50IM1	0000,0000
P6IM1	P6 口中断模式寄存器 1	7EFD36H	P67IM1	P66IM1	P65IM1	P64IM1	P63IM1	P62IM1	P61IM1	P60IM1	0000,0000
P7IM1	P7 口中断模式寄存器 1	7EFD37H	P77IM1	P76IM1	P75IM1	P74IM1	P73IM1	P72IM1	P71IM1	P70IM1	0000,0000
P8IM1	P8 口中断模式寄存器 1	7EFD38H	P87IM1	P86IM1	P85IM1	P84IM1	P83IM1	P82IM1	P81IM1	P80IM1	0000,0000
P9IM1	P9 口中断模式寄存器 1	7EFD39H	P97IM1	P96IM1	P95IM1	P94IM1	P93IM1	P92IM1	P91IM1	P90IM1	0000,0000
PAIM1	PA 口中断模式寄存器 1	7EFD3AH	PA7IM1	PA6IM1	PA5IM1	PA4IM1	PA3IM1	PA2IM1	PA1IM1	PA0IM1	0000,0000
PBIM1	PB 口中断模式寄存器 1	7EFD3BH	PB7IM1	PB6IM1	PB5IM1	PB4IM1	PB3IM1	PB2IM1	PB1IM1	PB0IM1	0000,0000
P0WKUE	P0 口中断唤醒使能	7EFD40H	P07WKUE	P06WKUE	P05WKUE	P04WKUE	P03WKUE	P02WKUE	P01WKUE	P00WKUE	0000,0000
P1WKUE	P1 口中断唤醒使能	7EFD41H	P17WKUE	P16WKUE	P15WKUE	P14WKUE	P13WKUE	P12WKUE	P11WKUE	P10WKUE	0000,0000
P2WKUE	P2 口中断唤醒使能	7EFD42H	P27WKUE	P26WKUE	P25WKUE	P24WKUE	P23WKUE	P22WKUE	P21WKUE	P20WKUE	0000,0000
P3WKUE	P3 口中断唤醒使能	7EFD43H	P37WKUE	P36WKUE	P35WKUE	P34WKUE	P33WKUE	P32WKUE	P31WKUE	P30WKUE	0000,0000
P4WKUE	P4 口中断唤醒使能	7EFD44H	P47WKUE	P46WKUE	P45WKUE	P44WKUE	P43WKUE	P42WKUE	P41WKUE	P40WKUE	0000,0000
P5WKUE	P5 口中断唤醒使能	7EFD45H	P57WKUE	P56WKUE	P55WKUE	P54WKUE	P53WKUE	P52WKUE	P51WKUE	P50WKUE	0000,0000
P6WKUE	P6 口中断唤醒使能	7EFD46H	P67WKUE	P66WKUE	P65WKUE	P64WKUE	P63WKUE	P62WKUE	P61WKUE	P60WKUE	0000,0000
P7WKUE	P7 口中断唤醒使能	7EFD47H	P77WKUE	P76WKUE	P75WKUE	P74WKUE	P73WKUE	P72WKUE	P71WKUE	P70WKUE	0000,0000
P8WKUE	P8 口中断唤醒使能	7EFD48H	P87WKUE	P86WKUE	P85WKUE	P84WKUE	P83WKUE	P82WKUE	P81WKUE	P80WKUE	0000,0000
P9WKUE	P9 口中断唤醒使能	7EFD49H	P97WKUE	P96WKUE	P95WKUE	P94WKUE	P93WKUE	P92WKUE	P91WKUE	P90WKUE	0000,0000
PAWKUE	PA 口中断唤醒使能	7EFD4AH	PA7WKUE	PA6WKUE	PA5WKUE	PA4WKUE	PA3WKUE	PA2WKUE	PA1WKUE	PA0WKUE	0000,0000
PBWKUE	PB 口中断唤醒使能	7EFD4BH	PB7WKUE	PB6WKUE	PB5WKUE	PB4WKUE	PB3WKUE	PB2WKUE	PB1WKUE	PB0WKUE	0000,0000
PINIPL	I/O 口中断优先级低寄存器	7EFD60H	P7IP	P6IP	P5IP	P4IP	P3IP	P2IP	P1IP	P0IP	0000,0000
PINIPH	I/O 口中断优先级高寄存器	7EFD61H	P7IPH	P6IPH	P5IPH	P4IPH	P3IPH	P2IPH	P1IPH	P0IPH	0000,0000
PIN2IPL	I/O 口中断优先级低寄存器 2	7EFD62H	-	-	-	-	PBIP	PAIP	P9IP	P8IP	xxxx,0000
PIN2IPH	I/O 口中断优先级高寄存器 2	7EFD63H	-	-	-	-	PBIPH	PAIPH	P9IPH	P8IPH	xxxx,0000
FSHWUPPRD	FLASH 唤醒等待时间寄存器	7EFD68H	FSHWUPPRD[7:0]								0011,1100
P_SWX1	功能脚切换扩展寄存器 1	7EFD69H	S4_S[1]	S3_S[1]	S2_S[1]	-	CAN2_S[2]	CAN_S[2]	-	SWP54P12	000x,00x0
P_SWX2	功能脚切换扩展寄存器 2	7EFD6AH	-	-	LIN2_S[1:0]		S6_S[1:0]		S5_S[1:0]		xx00,0000
P_SWX3	功能脚切换扩展寄存器 3	7EFD6BH	S6SPI_S[1:0]		S5SPI_S[1:0]		S4SPI_S[1:0]		S3SPI_S[1:0]		0000,0000
P_SWX4	功能脚切换扩展寄存器 4	7EFD6CH	I2C2_S[1:]		-	-	SPI3_S[1:0]		SPI2_S[1:0]		00xx,0000
P_SWX5	功能脚切换扩展寄存器 5	7EFD6DH	S8_S[1:0]		S7_S[1:0]		I2S2_S[1:0]		-	-	0000,00xx
P_SWX6	功能脚切换扩展寄存器 6	7EFD6EH	-	-	-	-	S8SPI_S[1:0]		S7SPI_S[1:0]		xxxx,0000
UR1TOCR	串口 1 接收超时控制寄存器	7EFD70H	UR1ENTO	UR1ENTOI	UR1SCALE	-	-	-	-	-	000x,xxxx
UR1TOSR	串口 1 接收超时状态寄存器	7EFD71H	-	-	-	-	-	-	-	UR1TOIF	xxxx,xxx0
UR1TOTH	串口 1 接收超时长度寄存器	7EFD72H	UR1TM[15:8]								0000,0000
UR1TOTL	串口 1 接收超时长度寄存器	7EFD73H	UR1TM[7:0]								0000,0000
UR2TOCR	串口 2 接收超时控制寄存器	7EFD74H	UR2ENTO	UR2ENTOI	UR2SCALE	-	-	-	-	-	000x,xxxx
UR2TOSR	串口 2 接收超时状态寄存器	7EFD75H	-	-	-	-	-	-	-	UR2TOIF	xxxx,xxx0
UR2TOTH	串口 2 接收超时长度寄存器	7EFD76H	UR2TM[15:8]								0000,0000
UR2TOTL	串口 2 接收超时长度寄存器	7EFD77H	UR2TM[7:0]								0000,0000
UR3TOCR	串口 3 接收超时控制寄存器	7EFD78H	UR3ENTO	UR3ENTOI	UR3SCALE	-	-	-	-	-	000x,xxxx
UR3TOSR	串口 3 接收超时状态寄存器	7EFD79H	-	-	-	-	-	-	-	UR3TOIF	xxxx,xxx0

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
UR3TOTH	串口 3 接收超时长度寄存器	7EFD7AH	UR3TM[15:8]								0000,0000
UR3TOTL	串口 3 接收超时长度寄存器	7EFD7BH	UR3TM[7:0]								0000,0000
UR4TOCR	串口 4 接收超时控制寄存器	7EFD7CH	UR4ENTO	UR4ENTOI	UR4SCALE	-	-	-	-	-	000x,xxxx
UR4TOSR	串口 4 接收超时状态寄存器	7EFD7DH	-	-	-	-	-	-	-	UR4TOIF	xxxx,xxx0
UR4TOTH	串口 4 接收超时长度寄存器	7EFD7EH	UR4TM[15:8]								0000,0000
UR4TOTL	串口 4 接收超时长度寄存器	7EFD7FH	UR4TM[7:0]								0000,0000
SPITOCR	SPI 从机超时控制寄存器	7EFD80H	SPIENTO	SPIENTOI	SPISCALE	-	-	-	-	-	000x,xxxx
SPITOSR	SPI 从机超时状态寄存器	7EFD81H	SPICTOIF	-	-	-	-	-	-	SPITOIF	0xxx,xxx0
SPITOTH	SPI 从机超时长度寄存器	7EFD82H	SPITM[15:8]								0000,0000
SPITOTL	SPI 从机超时长度寄存器	7EFD83H	SPITM[7:0]								0000,0000
I2CTOCR	I2C 从机超时控制寄存器	7EFD84H	I2CENTO	I2CENTOI	I2CSCALE	-	-	-	-	-	000x,xxxx
I2CTOSR	I2C 从机超时状态寄存器	7EFD85H	I2CCTOF	-	-	-	-	-	-	I2CTOIF	xxxx,xxx0
I2CTOTH	I2C 从机超时长度寄存器	7EFD86H	I2CTM[15:8]								0000,0000
I2CTOTL	I2C 从机超时长度寄存器	7EFD87H	I2CTM[7:0]								0000,0000
UR1TOTE	串口 1 接收超时长度寄存器	7EFD88H	UR1TM[23:16]								0000,0000
UR2TOTE	串口 2 接收超时长度寄存器	7EFD89H	UR2TM[23:16]								0000,0000
UR3TOTE	串口 3 接收超时长度寄存器	7EFD8AH	UR3TM[23:16]								0000,0000
UR4TOTE	串口 4 接收超时长度寄存器	7EFD8BH	UR4TM[23:16]								0000,0000
SPITOTE	SPI 从机超时长度寄存器	7EFD8CH	SPITM[23:16]								0000,0000
I2CTOTE	I2C 从机超时长度寄存器	7EFD8DH	I2CTM[23:16]								0000,0000
I2S2CR	I2S2 控制寄存器	7EFD90H	I2S2TXEIE	I2S2RXNEIE	I2S2ERRIE	I2S2FRF	-	-	I2S2TXDMAEN	I2S2RXDMAEN	0000,xx00
I2S2SR	I2S2 状态寄存器	7EFD91H	-	I2S2FRE	I2S2BUY	I2S2OVR	I2S2UDR	I2S2CHSID	I2S2TXE	I2S2RXNE	x000,0000
I2S2DRH	I2S2 数据寄存器高字节	7EFD92H	I2S2DR[15:8]								0000,0000
I2S2DRL	I2S2 数据寄存器低字节	7EFD93H	I2S2DR[7:0]								0000,0000
I2S2PRH	I2S2 分频寄存器高字节	7EFD94H	-	-	-	-	-	-	I2S2MCKOE	I2S2ODD	xxxx,xx00
I2S2PRL	I2S2 分频寄存器低字节	7EFD95H	I2S2DIV[7:0]								0000,0000
I2S2CFGH	I2S2 配置寄存器高字节	7EFD96H	-	-	-	-	-	I2S2E	I2S2CFG[1:0]		xxxx,x000
I2S2CFGL	I2S2 配置寄存器低字节	7EFD97H	I2S2PCMSYNC	-	I2S2STD[1:0]		I2S2CKPOL	I2S2DATLEN[1:0]		I2S2CHLEN	0x00,0000
I2SCR	I2S 控制寄存器	7EFD98H	I2STXEIE	I2SRXNEIE	I2SERRIE	I2SFRF	-	-	I2STXDMAEN	I2SRXDMAEN	0000,xx00
I2SSR	I2S 状态寄存器	7EFD99H	-	I2SFRE	I2SBUY	I2SOVR	I2SUDR	I2SCHSID	I2STXE	I2SRXNE	x000,0000
I2SDRH	I2S 数据寄存器高字节	7EFD9AH	I2SDR[15:8]								0000,0000
I2SDRL	I2S 数据寄存器低字节	7EFD9BH	I2SDR[7:0]								0000,0000
I2SPRH	I2S 分频寄存器高字节	7EFD9CH	-	-	-	-	-	-	I2SMCKOE	I2SODD	xxxx,xx00
I2SPRL	I2S 分频寄存器低字节	7EFD9DH	DIV[7:0]								0000,0000
I2SCFGH	I2S 配置寄存器高字节	7EFD9EH	-	-	-	-	-	I2SE	I2SCFG[1:0]		xxxx,x000
I2SCFGL	I2S 配置寄存器低字节	7EFD9FH	I2SPCMSYNC	-	I2SSTD[1:0]		I2SCKPOL	I2SDATLEN[1:0]		I2SCHLEN	0x00,0000
I2SMCKDIV	I2S 时钟分频寄存器	7EFDA1H	I2SMCKDIV[7:0]								0000,0000
I2SINSEL	I2S 输入信号选择寄存器	7EFDA2H	-	-	-	-	-	—	I2SINSEL[1:0]		xxxx,xx00
I2S2MCKDIV	I2S2 时钟分频寄存器	7EFDA5H	I2S2MCKDIV[7:0]								0000,0000
I2S2INSEL	I2S2 输入信号选择寄存器	7EFDA6H	-	-	-	-	-	—	I2S2INSEL[1:0]		xxxx,xx00
CRECR	CRE 控制寄存器	7EFDA8H	ENCRE	MONO	UPT[1:0]		CREHF	CREINC	CREDEC	CRERDY	0000,0000
CRECNTH	CRE 校准目标寄存器	7EFDA9H	CNT[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CRECNTL	CRE 校准目标寄存器	7EFDAAH	CNT[7:0]								0000,0000
CRERES	CRE 分辨率控制寄存器	7EFDABH	RES[7:0]								0000,0000
S2CFG	串口 2 配置寄存器	7EFDB4H	S2MOD	S2MOD0	S2M0x6	-	-	-	-	-	000x,xxx1
S2ADDR	串口 2 从机地址寄存器	7EFDB5H	S2ADDR[7:0]								0000,0000
S2ADEN	串口 2 从机地址屏蔽寄存器	7EFDB6H	S2ADEN[7:0]								0000,0000
S3CFG	串口 3 配置寄存器	7EFDB8H	S3MOD	S3MOD0	S3M0x6	-	-	-	-	S3BRT	000x,xxx0
S3ADDR	串口 3 从机地址寄存器	7EFDB9H	S3ADDR[7:0]								0000,0000
S3ADEN	串口 3 从机地址屏蔽寄存器	7EFDBAH	S3ADEN[7:0]								0000,0000
S4CFG	串口 4 配置寄存器	7EFDBCH	S4MOD	S4MOD0	S4M0x6	-	-	-	-	S4BRT	000x,xxx0
S4ADDR	串口 4 从机地址寄存器	7EFDBDH	S4ADDR[7:0]								0000,0000
S4ADEN	串口 4 从机地址屏蔽寄存器	7EFDBEH	S4ADEN[7:0]								0000,0000
USARTCR1	串口 1 控制寄存器 1	7EFDC0H	S1LINEN	S1DORD	S1CLKEN	S1SPMOD	S1SPIEN	S1SPSLV	S1CPOL	S1CPHA	0000,0000
USARTCR2	串口 1 控制寄存器 2	7EFDC1H	S1IREN	S1IRLP	S1SCEN	S1NACK	S1HDSEL	S1PCEN	S1PS	S1PE	0000,0000
USARTCR3	串口 1 控制寄存器 3	7EFDC2H	S1IrDA_LPBAUD[7:0]								0000,0111
USARTCR4	串口 1 控制寄存器 4	7EFDC3H	-	-	-	-	S1SCCKS[1:0]		S1SPICKS[1:0]		xxxx,0000
USARTCR5	串口 1 控制寄存器 5	7EFDC4H	S1BRKDET	S1HDRER	S1SLVEN	S1ASYNC	S1TXCF	S1SENDBK	S1HDRDET	S1SYNC	0000,0000
USARTGTR	串口 1 保护时间寄存器	7EFDC5H	USARTGTR[7:0]								0000,0000
USARTBRH	串口 1 波特率寄存器	7EFDC6H	USARTBR[15:8]								0000,0000
USARTBRL	串口 1 波特率寄存器	7EFDC7H	USARTBR[7:0]								0000,0000
USART2CR1	串口 2 控制寄存器 1	7EFDC8H	S2LINEN	S2DORD	S2CLKEN	S2SPMOD	S2SPIEN	S2SPSLV	S2CPOL	S2CPHA	0000,0000
USART2CR2	串口 2 控制寄存器 2	7EFDC9H	S2IREN	S2IRLP	S2SCEN	S2NACK	S2HDSEL	S2PCEN	S2PS	S2PE	0000,0000
USART2CR3	串口 2 控制寄存器 3	7EFDCAH	S2IrDA_LPBAUD[7:0]								0000,0000
USART2CR4	串口 2 控制寄存器 4	7EFDCBH	-	-	-	-	S2SCCKS[1:0]		S2SPICKS[1:0]		xxxx,0000
USART2CR5	串口 2 控制寄存器 5	7EFDCCH	S2BRKDET	S2HDRER	S2SLVEN	S2ASYNC	S2TXCF	S2SENDBK	S2HDRDET	S2SYNCAN	0000,0000
USART2GTR	串口 2 保护时间寄存器	7EFDCDH	USART2GTR[7:0]								0000,0000
USART2BRH	串口 2 波特率寄存器	7EFDCEH	USART2BR[15:8]								0000,0000
USART2BRL	串口 2 波特率寄存器	7EFDCFH	USART2BR[7:0]								0000,0000
USART3CR1	串口 3 控制寄存器 1	7EFDD0H	S3LINEN	S3DORD	S3CLKEN	S3SPMOD	S3SPIEN	S3SPSLV	S3CPOL	S3CPHA	0000,0000
USART3CR2	串口 3 控制寄存器 2	7EFDD1H	S3IREN	S3IRLP	S3SCEN	S3NACK	S3HDSEL	S3PCEN	S3PS	S3PE	0000,0000
USART3CR3	串口 3 控制寄存器 3	7EFDD2H	S3IrDA_LPBAUD[7:0]								0000,0111
USART3CR4	串口 3 控制寄存器 4	7EFDD3H	-	-	-	-	S3SCCKS[1:0]		S3SPICKS[1:0]		xxxx,0000
USART3CR5	串口 3 控制寄存器 5	7EFDD4H	S3BRKDET	S3HDRER	S3SLVEN	S3ASYNC	S3TXCF	S3SENDBK	S3HDRDET	S3SYNC	0000,0000
USART3GTR	串口 3 保护时间寄存器	7EFDD5H	USART3GTR[7:0]								0000,0000
USART3BRH	串口 3 波特率寄存器	7EFDD6H	USART3BR[15:8]								0000,0000
USART3BRL	串口 3 波特率寄存器	7EFDD7H	USART3BR[7:0]								0000,0000
USART4CR1	串口 4 控制寄存器 1	7EFDD8H	S4LINEN	S4DORD	S4CLKEN	S4SPMOD	S4SPIEN	S4SPSLV	S4CPOL	S4CPHA	0000,0000
USART4CR2	串口 4 控制寄存器 2	7EFDD9H	S4IREN	S4IRLP	S4SCEN	S4NACK	S4HDSEL	S4PCEN	S4PS	S4PE	0000,0000
USART4CR3	串口 4 控制寄存器 3	7EFDDAH	S4IrDA_LPBAUD[7:0]								0000,0000
USART4CR4	串口 4 控制寄存器 4	7EFDDBH	-	-	-	-	S4SCCKS[1:0]		S4SPICKS[1:0]		xxxx,0000
USART4CR5	串口 4 控制寄存器 5	7EFDDCH	S4BRKDET	S4HDRER	S4SLVEN	S4ASYNC	S4TXCF	S4SENDBK	S4HDRDET	S4SYNCAN	0000,0000
USART4GTR	串口 4 保护时间寄存器	7EFDDDH	USART4GTR[7:0]								0000,0000
USART4BRH	串口 4 波特率寄存器	7EFDDEH	USART4BR[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
USART4BRL	串口 4 波特率寄存器	7EFDDFH	USART4BR[7:0]								0000,0000
CHIPID0	硬件数字 ID00	7EFDE0H	全球唯一 ID 号（第 0 字节）								nnnn,nnnn
CHIPID1	硬件数字 ID01	7EFDE1H	全球唯一 ID 号（第 1 字节）								nnnn,nnnn
CHIPID2	硬件数字 ID02	7EFDE2H	全球唯一 ID 号（第 2 字节）								nnnn,nnnn
CHIPID3	硬件数字 ID03	7EFDE3H	全球唯一 ID 号（第 3 字节）								nnnn,nnnn
CHIPID4	硬件数字 ID04	7EFDE4H	全球唯一 ID 号（第 4 字节）								nnnn,nnnn
CHIPID5	硬件数字 ID05	7EFDE5H	全球唯一 ID 号（第 5 字节）								nnnn,nnnn
CHIPID6	硬件数字 ID06	7EFDE6H	全球唯一 ID 号（第 6 字节）								nnnn,nnnn
CHIPID7	硬件数字 ID07	7EFDE7H	内部 1.19V 参考信号源-BGV（高字节）								nnnn,nnnn
CHIPID8	硬件数字 ID08	7EFDE8H	内部 1.19V 参考信号源-BGV（低字节）								nnnn,nnnn
CHIPID9	硬件数字 ID09	7EFDE9H	32K 掉电唤醒定时器的频率（高字节）								nnnn,nnnn
CHIPID10	硬件数字 ID10	7EFDEAH	32K 掉电唤醒定时器的频率（低字节）								nnnn,nnnn
CHIPID11	硬件数字 ID11	7EFDEBH	22.1184MHz 的 IRC 参数（27M 频段）								nnnn,nnnn
CHIPID12	硬件数字 ID12	7EFDECH	24MHz 的 IRC 参数（27M 频段）								nnnn,nnnn
CHIPID13	硬件数字 ID13	7EFDEDH	40MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID14	硬件数字 ID14	7EFDEEH	45.1584MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID15	硬件数字 ID15	7EFDEFH	48MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID16	硬件数字 ID16	7EFDF0H	49.152MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID17	硬件数字 ID17	7EFDF1H	50.8032MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID18	硬件数字 ID18	7EFDF2H	55.296MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID19	硬件数字 ID19	7EFDF3H	60MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID20	硬件数字 ID20	7EFDF4H	64MHz 的 IRC 参数（44M 频段）								nnnn,nnnn
CHIPID21	硬件数字 ID21	7EFDF5H	10M 频段的 VRTRIM 参数								nnnn,nnnn
CHIPID22	硬件数字 ID22	7EFDF6H	27M 频段的 VRTRIM 参数								nnnn,nnnn
CHIPID23	硬件数字 ID23	7EFDF7H	44M 频段的 VRTRIM 参数								nnnn,nnnn
CHIPID24	硬件数字 ID24	7EFDF8H	72M 频段的 VRTRIM 参数								nnnn,nnnn
CHIPID25	硬件数字 ID25	7EFDF9H	00H								nnnn,nnnn
CHIPID26	硬件数字 ID26	7EFDFAH	用户程序空间结束地址（高字节）								nnnn,nnnn
CHIPID27	硬件数字 ID27	7EFDFBH	芯片测试时间（年）								nnnn,nnnn
CHIPID28	硬件数字 ID28	7EFDFCH	芯片测试时间（月）								nnnn,nnnn
CHIPID29	硬件数字 ID29	7EFDFDH	芯片测试时间（日）								nnnn,nnnn
CHIPID30	硬件数字 ID30	7EFDFEH	芯片封装形式编号								nnnn,nnnn
CHIPID31	硬件数字 ID31	7EFDFFH	5AH								nnnn,nnnn

13.6 扩展特殊功能寄存器列表（XFR：0x7EFB00-0x7EFBFF）

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CRC_CR	CRC 控制寄存器	7EFB30H	RST	-	-	MODE	-	INVOUT	REVIN	REVOUT	0xx0,x011
CRC_DIN	CRC 数据输入寄存器	7EFB31H	DIN[7:0]								0000,0000
CRC_DOH	16 位 CRC 结果高字节	7EFB32H	DOUT_16[15:8]								1111,1111
CRC_DOL	16 位 CRC 结果低字节	7EFB33H	DOUT_16[7:0]								1111,1111
CRC_DO0	32 位 CRC 结果最高字节	7EFB32H	DOUT_32[31:24]								1111,1111
CRC_DO1	32 位 CRC 结果高字节	7EFB33H	DOUT_32[23:16]								1111,1111
CRC_DO2	32 位 CRC 结果低字节	7EFB34H	DOUT_32[15:8]								1111,1111
CRC_DO3	32 位 CRC 结果最字节	7EFB35H	DOUT_32[7:0]								1111,1111
ADC2_CONTR	ADC2 控制寄存器	7EFB38H	ADC2_POWER	ADC2_START	ADC2_FLAG	ADC2_EPWMT	ADC2_CHS[3:0]				0000,0000
ADC2CFG	ADC2 配置寄存器	7EFB39H	-	-	ADC2_RESFMT	-	ADC2_SPEED[3:0]				xx0x,0000
ADC2_RES	ADC2 转换结果高位寄存器	7EFB3AH	ADC2_RES[15:8]								0000,0000
ADC2_RESL	ADC2 转换结果低位寄存器	7EFB3BH	ADC2_RESL[7:0]ADC2								0000,0000
ADC2TIM	ADC2 时序控制寄存器	7EFB3CH	ADC2_CSSETUP	ADC2_CSHOLD[1:0]		ADC2_SMPDUTY[4:0]					0010,1010
ADC2EXCFG	ADC2 扩展配置寄存器	7EFB3DH	ADC2ETR_PS[1:0]		ADC2ETRs[1:0]		-	ADC2_CVTIMESEL[2:0]			0000,x000
ADC2INTR	ADC2 中断控制寄存器	7EFB3EH	-	-	-	-	-	PADC2[1:0]		EADC2	xxxx,x000
ADC2EXCFG2	ADC2 扩展配置寄存器 2	7EFB3FH	-	-	PWMSEL[1:0]		ADC_TRIGSEL[3:0]				xx00,0000
CMP1_6P6N_CR1	比较器 1 控制寄存器 1	7EFB70H	CMP1EN	CMP1IF	CMP1PIE	CMP1NIE	CMP1IP[1:0]		CMP1OE	CMP1RES	0000,0000
CMP1_6P6N_CR2	比较器 1 控制寄存器 2	7EFB71H	INVCMP1O	CMP1DISFLT	CMP1LCDTY[5:0]						0000,0000
CMP1_6P6N_CR3	比较器 1 控制寄存器 3	7EFB72H	CMP1CHYS[1:0]		CMP1NS[2:0]		CMP1PS[2:0]				0000,0000
CMP1_6P6N_CR4	比较器 1 控制寄存器 4	7EFB73H	-	-	-	-	-	-	CMP1O_S[1:0]		xxxx,xx00
CMP2_6P6N_CR1	比较器 2 控制寄存器 1	7EFB74H	CMP2EN	CMP2IF	CMP2PIE	CMP2NIE	CMP2IP[1:0]		CMP2OE	CMP2RES	0000,0000
CMP2_6P6N_CR2	比较器 2 控制寄存器 2	7EFB75H	INVCMP2O	CMP2DISFLT	CMP2LCDTY[5:0]						0000,0000
CMP2_6P6N_CR3	比较器 2 控制寄存器 3	7EFB76H	CMP2CHYS[1:0]		CMP2NS[2:0]		CMP2PS[2:0]				0000,0000
CMP2_6P6N_CR4	比较器 2 控制寄存器 4	7EFB77H	-	-	-	-	-	-	CMP2O_S[1:0]		xxxx,xx00
CMP3_6P6N_CR1	比较器 3 控制寄存器 1	7EFB78H	CMP3EN	CMP3IF	CMP3PIE	CMP3NIE	CMP3IP[1:0]		CMP3OE	CMP3RES	0000,0000
CMP3_6P6N_CR2	比较器 3 控制寄存器 2	7EFB79H	INVCMP3O	CMP3DISFLT	CMP3LCDTY[5:0]						0000,0000
CMP3_6P6N_CR3	比较器 3 控制寄存器 3	7EFB7AH	CMP3CHYS[1:0]		CMP3NS[2:0]		CMP3PS[2:0]				0000,0000
CMP3_6P6N_CR4	比较器 3 控制寄存器 4	7EFB7BH	-	-	-	-	-	-	CMP3O_S[1:0]		xxxx,xx00
CMP4_6P6N_CR1	比较器 4 控制寄存器 1	7EFB7CH	CMP4EN	CMP4IF	CMP4PIE	CMP4NIE	CMP4IP[1:0]		CMP4OE	CMP4RES	0000,0000
CMP4_6P6N_CR2	比较器 4 控制寄存器 2	7EFB7DH	INVCMP4O	CMP4DISFLT	CMP4LCDTY[5:0]						0000,0000
CMP4_6P6N_CR3	比较器 4 控制寄存器 3	7EFB7EH	CMP4CHYS[1:0]		CMP4NS[2:0]		CMP4PS[2:0]				0000,0000
CMP4_6P6N_CR4	比较器 4 控制寄存器 4	7EFB7FH	-	-	-	-	-	-	CMP4O_S[1:0]		xxxx,xx00
TM5PS	定时器 5 预分频寄存器	7EFBB0H	TM5PS[7:0]								0000,0000
TM6PS	定时器 6 预分频寄存器	7EFBB1H	TM6PS[7:0]								0000,0000
T6T5CFG	定时器 6/5 配置寄存器	7EFBB2H	T6IF	PT6[1:0]		ET6	T5IF	PT5[1:0]		ET5	0000,0000
T6T5M	定时器 6/5 控制寄存器	7EFBB3H	T6R	T6_C/T	T6x12	T6CLKO	T5R	T5_C/T	T5x12	T5CLKO	0000,0000
T6H	定时器 6 高字节	7EFBB4H	TMR6[15:8]								0000,0000
T6L	定时器 6 低字节	7EFBB5H	TMR6[7:0]								0000,0000
T5H	定时器 5 高字节	7EFBB6H	TMR5[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
T5L	定时器 5 低字节	7EFBB7H	TMR5[7:0]								0000,0000
TM7PS	定时器 7 预分频寄存器	7EFBB8H	TM7PS[7:0]								0000,0000
TM8PS	定时器 8 预分频寄存器	7EFBB9H	TM8PS[7:0]								0000,0000
T8T7CFG	定时器 8/7 配置寄存器	7EFBBAH	T8IF	PT8[1:0]		ET8	T7IF	PT7[1:0]		ET7	0000,0000
T8T7M	定时器 8/7 控制寄存器	7EFBBBH	T8R	T8_C/T	T8x12	T8CLKO	T7R	T7_C/T	T7x12	T7CLKO	0000,0000
T8H	定时器 8 高字节	7EFBBCH	TMR8[15:8]								0000,0000
T8L	定时器 8 低字节	7EFBBDH	TMR8[7:0]								0000,0000
T7H	定时器 7 高字节	7EFBBEH	TMR7[15:8]								0000,0000
T7L	定时器 7 低字节	7EFBBFH	TMR7[7:0]								0000,0000
TM9PS	定时器 9 预分频寄存器	7EFBC0H	TM9PS[7:0]								0000,0000
TM10PS	定时器 10 预分频寄存器	7EFBC1H	TM10PS[7:0]								0000,0000
T10T9CFG	定时器 10/9 配置寄存器	7EFBC2H	T10IF	PT10[1:0]		ET10	T9IF	PT9[1:0]		ET9	0000,0000
T10T9M	定时器 10/9 控制寄存器	7EFBC3H	T10R	T10_C/T	T10x12	T10CLKO	T9R	T9_C/T	T9x12	T9CLKO	0000,0000
T10H	定时器 10 高字节	7EFBC4H	TMR10[15:8]								0000,0000
T10L	定时器 10 低字节	7EFBC5H	TMR10[7:0]								0000,0000
T9H	定时器 9 高字节	7EFBC6H	TMR9[15:8]								0000,0000
T9L	定时器 9 低字节	7EFBC7H	TMR9[7:0]								0000,0000
TM17PS	定时器 17 预分频寄存器	7EFBC8H	TM17PS[7:0]								0000,0000
TM18PS	定时器 18 预分频寄存器	7EFBC9H	TM18PS[7:0]								0000,0000
T18T17CFG	定时器 181/7 配置寄存器	7EFBCAH	T18IF	PT18[1:0]		ET18	T17IF	PT17[1:0]		ET17	0000,0000
T18T17M	定时器 18/17 控制寄存器	7EFBCBH	T18R	T18_C/T	T18x12	T18CLKO	T17R	T17_C/T	T17x12	T17CLKO	0000,0000
T18H	定时器 18 高字节	7EFBCCH	TMR18[15:8]								0000,0000
T18L	定时器 18 低字节	7EFBCDH	TMR18[7:0]								0000,0000
T17H	定时器 17 高字节	7EFBCEH	TMR17[15:8]								0000,0000
T17L	定时器 17 低字节	7EFBCFH	TMR17[7:0]								0000,0000
CHIPIDX0	扩展硬件数字 ID00	7EFBD0H	CHIPIDX0[7:0]								nnnn,nnnn
CHIPIDX1	扩展硬件数字 ID01	7EFBD1H	CHIPIDX1[7:0]								nnnn,nnnn
CHIPIDX2	扩展硬件数字 ID02	7EFBD2H	CHIPIDX2[7:0]								nnnn,nnnn
CHIPIDX3	扩展硬件数字 ID03	7EFBD3H	CHIPIDX3[7:0]								nnnn,nnnn
CHIPIDX4	扩展硬件数字 ID04	7EFBD4H	CHIPIDX4[7:0]								nnnn,nnnn
CHIPIDX5	扩展硬件数字 ID05	7EFBD5H	CHIPIDX5[7:0]								nnnn,nnnn
CHIPIDX6	扩展硬件数字 ID06	7EFBD6H	CHIPIDX6[7:0]								nnnn,nnnn
CHIPIDX7	扩展硬件数字 ID07	7EFBD7H	CHIPIDX7[7:0]								nnnn,nnnn
CHIPIDX8	扩展硬件数字 ID08	7EFBD8H	CHIPIDX8[7:0]								nnnn,nnnn
CHIPIDX9	扩展硬件数字 ID09	7EFBD9H	CHIPIDX9[7:0]								nnnn,nnnn
CHIPIDX10	扩展硬件数字 ID10	7EFBDAH	CHIPIDX10[7:0]								nnnn,nnnn
CHIPIDX11	扩展硬件数字 ID11	7EFBDBH	CHIPIDX11[7:0]								nnnn,nnnn
CHIPIDX12	扩展硬件数字 ID12	7EFBDCH	CHIPIDX12[7:0]								nnnn,nnnn
CHIPIDX13	扩展硬件数字 ID13	7EFBDDH	CHIPIDX13[7:0]								nnnn,nnnn
CHIPIDX14	扩展硬件数字 ID14	7EFBDEH	CHIPIDX14[7:0]								nnnn,nnnn
CHIPIDX15	扩展硬件数字 ID15	7EFBDFH	CHIPIDX15[7:0]								nnnn,nnnn
CHIPIDX16	扩展硬件数字 ID16	7EFBE0H	CHIPIDX16[7:0]								nnnn,nnnn

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CHIPIDX17	扩展硬件数字 ID17	7EFBE1H	CHIPIDX17[7:0]								nnnn,nnnn
CHIPIDX18	扩展硬件数字 ID18	7EFBE2H	CHIPIDX18[7:0]								nnnn,nnnn
CHIPIDX19	扩展硬件数字 ID19	7EFBE3H	CHIPIDX19[7:0]								nnnn,nnnn
CHIPIDX20	扩展硬件数字 ID20	7EFBE4H	CHIPIDX20[7:0]								nnnn,nnnn
CHIPIDX21	扩展硬件数字 ID21	7EFBE5H	CHIPIDX21[7:0]								nnnn,nnnn
CHIPIDX22	扩展硬件数字 ID22	7EFBE6H	CHIPIDX22[7:0]								nnnn,nnnn
CHIPIDX23	扩展硬件数字 ID23	7EFBE7H	CHIPIDX23[7:0]								nnnn,nnnn
CHIPIDX24	扩展硬件数字 ID24	7EFBE8H	CHIPIDX24[7:0]								nnnn,nnnn
CHIPIDX25	扩展硬件数字 ID25	7EFBE9H	CHIPIDX25[7:0]								nnnn,nnnn
CHIPIDX26	扩展硬件数字 ID26	7EFBEAH	CHIPIDX26[7:0]								nnnn,nnnn
CHIPIDX27	扩展硬件数字 ID27	7EFBEBH	CHIPIDX27[7:0]								nnnn,nnnn
CHIPIDX28	扩展硬件数字 ID28	7EFBECH	CHIPIDX28[7:0]								nnnn,nnnn
CHIPIDX29	扩展硬件数字 ID29	7EFBEDH	CHIPIDX29[7:0]								nnnn,nnnn
CHIPIDX30	扩展硬件数字 ID30	7EFBEEH	CHIPIDX30[7:0]								nnnn,nnnn
CHIPIDX31	扩展硬件数字 ID31	7EFBEFH	CHIPIDX31[7:0]								nnnn,nnnn
HSPWMA_CFG	高速 PWMA 配置寄存器	7EFBF0H	-	-	-	PWMA_EXTN	-	PWMA_INTEN	PWMA_ASYNCEN	1	xxx0,x001
HSPWMA_ADR	高速 PWMA 地址寄存器	7EFBF1H	PWMA_RW	PWMA_ADDR[6:0]							0000,0000
HSPWMA_DAT	高速 PWMA 数据寄存器	7EFBF2H	PWMA_DATA[7:0]								0000,0000
HSPWMB_CFG	高速 PWMB 配置寄存器	7EFBF4H	-	-	-	-	-	PWMB_INTEN	PWMB_ASYNCEN	1	xxxx,x001
HSPWMB_ADR	高速 PWMB 地址寄存器	7EFBF5H	PWMB_RW	PWMB_ADDR[6:0]							0000,0000
HSPWMB_DAT	高速 PWMB 数据寄存器	7EFBF6H	PWMB_DATA[7:0]								0000,0000
HSSPI_CFG	高速 SPI 配置寄存器	7EFBF8H	HSSPI_SS_HLD[3:0]				HSSPI_SS_SETUP[3:0]				0011,0011
HSSPI_CFG2	高速 SPI 配置寄存器 2	7EFBF9H	MOSI_PL	SPI_IOSW	HSSPI_EN	HSSPI_FIFOEN	HSSPI_SS_DACT[3:0]				0000,0011
HSSPI_STA	高速 SPI 状态寄存器	7EFBFAH	-	-	-	-	TXFULL	TXEMPT	RXFULL	RXEMPT	xxxx,0000
HSSPI_PSCR	高速 SPI 时钟分频寄存器	7EFBFBH	HSSPI_PSCR [7:0]								0000,0000

13.7 扩展特殊功能寄存器列表（XFR：0x7EFA00-0x7EFAFF）

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_M2M_CFG	M2M_DMA 配置寄存器	7EFA00H	M2MIE	-	TXACO	RXACO	M2MIP[1:0]		M2MPTY[1:0]		0x00,0000
DMA_M2M_CR	M2M_DMA 控制寄存器	7EFA01H	ENM2M	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_M2M_STA	M2M_DMA 状态寄存器	7EFA02H	-	-	-	-	-	-	-	M2MIF	xxxx,xxx0
DMA_M2M_AMT	M2M_DMA 传输总字节数	7EFA03H	M2M_AMT[7:0]								0000,0000
DMA_M2M_DONE	M2M_DMA 传输完成字节数	7EFA04H	M2M_DONE[7:0]								0000,0000
DMA_M2M_TXAH	M2M_DMA 发送高地址	7EFA05H	M2M_TXA[15:8]								0000,0000
DMA_M2M_TXAL	M2M_DMA 发送低地址	7EFA06H	M2M_TXA[7:0]								0000,0000
DMA_M2M_RXAH	M2M_DMA 接收高地址	7EFA07H	M2M_RXA[15:8]								0000,0000
DMA_M2M_RXAL	M2M_DMA 接收低地址	7EFA08H	M2M_RXA[7:0]								0000,0000
DMA_ADC_CFG	ADC_DMA 配置寄存器	7EFA10H	ADCIE	-	-	-	ADCMIP[1:0]		ADCPTY[1:0]		0xxx,0000
DMA_ADC_CR	ADC_DMA 控制寄存器	7EFA11H	ENADC	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_ADC_STA	ADC_DMA 状态寄存器	7EFA12H	-	-	-	-	-	-	-	ADCIF	xxxx,xxx0
DMA_ADC_AMT	ADC_DMA 传输总字节数	7EFA13H	ADC_AMT[7:0]								0000,0000
DMA_ADC_DONE	ADC_DMA 传输完成字节数	7EFA14H	ADC_DONE[7:0]								0000,0000
DMA_ADC_RXAH	ADC_DMA 接收高地址	7EFA17H	ADC_RXA[15:8]								0000,0000
DMA_ADC_RXAL	ADC_DMA 接收低地址	7EFA18H	ADC_RXA[7:0]								0000,0000
DMA_ADC_CFG2	ADC_DMA 配置寄存器 2	7EFA19H	-	-	-	-	CVTIMESEL[3:0]				xxxx,0000
DMA_ADC_CHSW0	ADC_DMA 通道使能	7EFA1AH	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0	0000,0001
DMA_ADC_CHSW1	ADC_DMA 通道使能	7EFA1BH	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	1000,0000
DMA_ADC_ITVH	ADC_DMA 时间间隔寄存器	7EFA1EH	ADC_ITV[15:8]								0000,0000
DMA_ADC_ITVL	ADC_DMA 时间间隔寄存器	7EFA1FH	ADC_ITV[7:0]								0000,0000
DMA_SPI_CFG	SPI_DMA 配置寄存器	7EFA20H	SPIIE	ACT_TX	ACT_RX	-	SPIIP[1:0]		SPIPTY[1:0]		000x,0000
DMA_SPI_CR	SPI_DMA 控制寄存器	7EFA21H	ENSPI	TRIG_M	TRIG_S	-	-	-	-	CLRFIFO	000x,xxx0
DMA_SPI_STA	SPI_DMA 状态寄存器	7EFA22H	-	-	-	-	-	TXOVW	RXLOSS	SPIIF	xxxx,x000
DMA_SPI_AMT	SPI_DMA 传输总字节数	7EFA23H	SPI_AMT[7:0]								0000,0000
DMA_SPI_DONE	SPI_DMA 传输完成字节数	7EFA24H	SPI_DONE[7:0]								0000,0000
DMA_SPI_TXAH	SPI_DMA 发送高地址	7EFA25H	SPI_TXA[15:8]								0000,0000
DMA_SPI_TXAL	SPI_DMA 发送低地址	7EFA26H	SPI_TXA[7:0]								0000,0000
DMA_SPI_RXAH	SPI_DMA 接收高地址	7EFA27H	SPI_RXA[15:8]								0000,0000
DMA_SPI_RXAL	SPI_DMA 接收低地址	7EFA28H	SPI_RXA[7:0]								0000,0000
DMA_SPI_CFG2	SPI_DMA 配置寄存器 2	7EFA29H	-	-	-	-	-	WRPSS	SSS[1:0]		xxxx,x000
DMA_SPI_ITVH	SPI_DMA 时间间隔寄存器	7EFA2EH	SPI_ITV[15:8]								0000,0000
DMA_SPI_ITVL	SPI_DMA 时间间隔寄存器	7EFA2FH	SPI_ITV[7:0]								0000,0000
DMA_UR1T_CFG	UR1T_DMA 配置寄存器	7EFA30H	UR1TIE	-	-	-	UR1TIP[1:0]		UR1TPTY[1:0]		0xxx,0000
DMA_UR1T_CR	UR1T_DMA 控制寄存器	7EFA31H	ENUR1T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR1T_STA	UR1T_DMA 状态寄存器	7EFA32H	-	-	-	-	-	TXOVW	-	UR1TIF	xxxx,x0x0
DMA_UR1T_AMT	UR1T_DMA 传输总字节数	7EFA33H	UR1T_AMT[7:0]								0000,0000
DMA_UR1T_DONE	UR1T_DMA 传输完成字节数	7EFA34H	UR1T_DONE[7:0]								0000,0000
DMA_UR1T_TXAH	UR1T_DMA 发送高地址	7EFA35H	UR1T_TXA[15:8]								0000,0000
DMA_UR1T_TXAL	UR1T_DMA 发送低地址	7EFA36H	UR1T_TXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR1R_CFG	UR1R_DMA 配置寄存器	7EFA38H	UR1RIE	-	-	-	UR1RIP[1:0]		UR1RPTY[1:0]		0xxx,0000
DMA_UR1R_CR	UR1R_DMA 控制寄存器	7EFA39H	ENUR1R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR1R_STA	UR1R_DMA 状态寄存器	7EFA3AH	-	-	-	-	-	-	RXLOSS	UR1RIF	xxxx,xx00
DMA_UR1R_AMT	UR1R_DMA 传输总字节数	7EFA3BH	UR1R_AMT[7:0]								0000,0000
DMA_UR1R_DONE	UR1R_DMA 传输完成字节数	7EFA3CH	UR1R_DONE[7:0]								0000,0000
DMA_UR1R_RXAH	UR1R_DMA 接收高地址	7EFA3DH	UR1R_RXA[15:8]								0000,0000
DMA_UR1R_RXAL	UR1R_DMA 接收低地址	7EFA3EH	UR1R_RXA[7:0]								0000,0000
DMA_UR2T_CFG	UR2T_DMA 配置寄存器	7EFA40H	UR2TIE	-	-	-	UR2TIP[1:0]		UR2TPTY[1:0]		0xxx,0000
DMA_UR2T_CR	UR2T_DMA 控制寄存器	7EFA41H	ENUR2T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR2T_STA	UR2T_DMA 状态寄存器	7EFA42H	-	-	-	-	-	TXOVW	-	UR2TIF	xxxx,x0x0
DMA_UR2T_AMT	UR2T_DMA 传输总字节数	7EFA43H	UR2T_AMT[7:0]								0000,0000
DMA_UR2T_DONE	UR2T_DMA 传输完成字节数	7EFA44H	UR2T_DONE[7:0]								0000,0000
DMA_UR2T_TXAH	UR2T_DMA 发送高地址	7EFA45H	UR2T_TXA[15:8]								0000,0000
DMA_UR2T_TXAL	UR2T_DMA 发送低地址	7EFA46H	UR2T_TXA[7:0]								0000,0000
DMA_UR2R_CFG	UR2R_DMA 配置寄存器	7EFA48H	UR2RIE	-	-	-	UR2RIP[1:0]		UR2RPTY[1:0]		0xxx,0000
DMA_UR2R_CR	UR2R_DMA 控制寄存器	7EFA49H	ENUR2R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR2R_STA	UR2R_DMA 状态寄存器	7EFA4AH	-	-	-	-	-	-	RXLOSS	UR2RIF	xxxx,xx00
DMA_UR2R_AMT	UR2R_DMA 传输总字节数	7EFA4BH	UR2R_AMT[7:0]								0000,0000
DMA_UR2R_DONE	UR2R_DMA 传输完成字节数	7EFA4CH	UR2R_DONE[7:0]								0000,0000
DMA_UR2R_RXAH	UR2R_DMA 接收高地址	7EFA4DH	UR2R_RXA[15:8]								0000,0000
DMA_UR2R_RXAL	UR2R_DMA 接收低地址	7EFA4EH	UR2R_RXA[7:0]								0000,0000
DMA_UR3T_CFG	UR3T_DMA 配置寄存器	7EFA50H	UR3TIE	-	-	-	UR3TIP[1:0]		UR3TPTY[1:0]		0xxx,0000
DMA_UR3T_CR	UR3T_DMA 控制寄存器	7EFA51H	ENUR3T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR3T_STA	UR3T_DMA 状态寄存器	7EFA52H	-	-	-	-	-	TXOVW	-	UR3TIF	xxxx,x0x0
DMA_UR3T_AMT	UR3T_DMA 传输总字节数	7EFA53H	UR3T_AMT[7:0]								0000,0000
DMA_UR3T_DONE	UR3T_DMA 传输完成字节数	7EFA54H	UR3T_DONE[7:0]								0000,0000
DMA_UR3T_TXAH	UR3T_DMA 发送高地址	7EFA55H	UR3T_TXA[15:8]								0000,0000
DMA_UR3T_TXAL	UR3T_DMA 发送低地址	7EFA56H	UR3T_TXA[7:0]								0000,0000
DMA_UR3R_CFG	UR3R_DMA 配置寄存器	7EFA58H	UR3RIE	-	-	-	UR3RIP[1:0]		UR3RPTY[1:0]		0xxx,0000
DMA_UR3R_CR	UR3R_DMA 控制寄存器	7EFA59H	ENUR3R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR3R_STA	UR3R_DMA 状态寄存器	7EFA5AH	-	-	-	-	-	-	RXLOSS	UR3RIF	xxxx,xx00
DMA_UR3R_AMT	UR3R_DMA 传输总字节数	7EFA5BH	UR3R_AMT[7:0]								0000,0000
DMA_UR3R_DONE	UR3R_DMA 传输完成字节数	7EFA5CH	UR3R_DONE[7:0]								0000,0000
DMA_UR3R_RXAH	UR3R_DMA 接收高地址	7EFA5DH	UR3R_RXA[15:8]								0000,0000
DMA_UR3R_RXAL	UR3R_DMA 接收低地址	7EFA5EH	UR3R_RXA[7:0]								0000,0000
DMA_UR4T_CFG	UR4T_DMA 配置寄存器	7EFA60H	UR4TIE	-	-	-	UR4TIP[1:0]		UR4TPTY[1:0]		0xxx,0000
DMA_UR4T_CR	UR4T_DMA 控制寄存器	7EFA61H	ENUR4T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR4T_STA	UR4T_DMA 状态寄存器	7EFA62H	-	-	-	-	-	TXOVW	-	UR4TIF	xxxx,x0x0
DMA_UR4T_AMT	UR4T_DMA 传输总字节数	7EFA63H	UR4T_AMT[7:0]								0000,0000
DMA_UR4T_DONE	UR4T_DMA 传输完成字节数	7EFA64H	UR4T_DONE[7:0]								0000,0000
DMA_UR4T_TXAH	UR4T_DMA 发送高地址	7EFA65H	UR4T_TXA[15:8]								0000,0000
DMA_UR4T_TXAL	UR4T_DMA 发送低地址	7EFA66H	UR4T_TXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR4R_CFG	UR4R_DMA 配置寄存器	7EFA68H	UR4RIE	-	-	-	UR4RIP[1:0]		UR4RPTY[1:0]		0xxx,0000
DMA_UR4R_CR	UR4R_DMA 控制寄存器	7EFA69H	ENUR4R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR4R_STA	UR4R_DMA 状态寄存器	7EFA6AH	-	-	-	-	-	-	RXLOSS	UR4RIF	xxxx,xx00
DMA_UR4R_AMT	UR4R_DMA 传输总字节数	7EFA6BH	UR4R_AMT[7:0]								0000,0000
DMA_UR4R_DONE	UR4R_DMA 传输完成字节数	7EFA6CH	UR4R_DONE[7:0]								0000,0000
DMA_UR4R_RXAH	UR4R_DMA 接收高地址	7EFA6DH	UR4R_RXA[15:8]								0000,0000
DMA_UR4R_RXAL	UR4R_DMA 接收低地址	7EFA6EH	UR4R_RXA[7:0]								0000,0000
DMA_LCM_CFG	LCM_DMA 配置寄存器	7EFA70H	LCMIE	-	-	-	LCMIP[1:0]		LCMPTY[1:0]		0xxx,0000
DMA_LCM_CR	LCM_DMA 控制寄存器	7EFA71H	ENLCM	TRIGWC	TRIGWD	TRIGRC	TRIGRD	-	-	-	0000,0xxx
DMA_LCM_STA	LCM_DMA 状态寄存器	7EFA72H	-	-	-	-	-	-	TXOVW	LCMIF	xxxx,xx00
DMA_LCM_AMT	LCM_DMA 传输总字节数	7EFA73H	LCM_AMT[7:0]								0000,0000
DMA_LCM_DONE	LCM_DMA 传输完成字节数	7EFA74H	LCM_DONE[7:0]								0000,0000
DMA_LCM_TXAH	LCM_DMA 发送高地址	7EFA75H	LCM_TXA[15:8]								0000,0000
DMA_LCM_TXAL	LCM_DMA 发送低地址	7EFA76H	LCM_TXA[7:0]								0000,0000
DMA_LCM_RXAH	LCM_DMA 接收高地址	7EFA77H	LCM_RXA[15:8]								0000,0000
DMA_LCM_RXAL	LCM_DMA 接收低地址	7EFA78H	LCM_RXA[7:0]								0000,0000
DMA_LCM_ITVH	LCM_DMA 时间间隔寄存器	7EFA7EH	LCM_ITV[15:8]								0000,0000
DMA_LCM_ITVL	LCM_DMA 时间间隔寄存器	7EFA7FH	LCM_ITV[7:0]								0000,0000
DMA_M2M_AMTH	M2M_DMA 传输总字节数	7EFA80H	M2M_AMT[15:8]								0000,0000
DMA_M2M_DONEH	M2M_DMA 传输完成字节数	7EFA81H	M2M_DONE[15:8]								0000,0000
DMA_ADC_AMTH	ADC_DMA 传输总字节数	7EFA82H	ADC_AMT[15:8]								0000,0000
DMA_ADC_DONEH	ADC_DMA 传输完成字节数	7EFA83H	ADC_DONE[15:8]								0000,0000
DMA_SPI_AMTH	SPI_DMA 传输总字节数	7EFA84H	SPI_AMT[15:8]								0000,0000
DMA_SPI_DONEH	SPI_DMA 传输完成字节数	7EFA85H	SPI_DONE[15:8]								0000,0000
DMA_LCM_AMTH	LCM_DMA 传输总字节数	7EFA86H	LCM_AMT[15:8]								0000,0000
DMA_LCM_DONEH	LCM_DMA 传输完成字节数	7EFA87H	LCM_DONE[15:8]								0000,0000
DMA_UR1T_AMTH	UR1T_DMA 传输总字节数	7EFA88H	UR1T_AMT[15:8]								0000,0000
DMA_UR1T_DONEH	UR1T_DMA 传输完成字节数	7EFA89H	UR1T_DONE[15:8]								0000,0000
DMA_UR1R_AMTH	UR1R_DMA 传输总字节数	7EFA8AH	UR1R_AMT[15:8]								0000,0000
DMA_UR1R_DONEH	UR1R_DMA 传输完成字节数	7EFA8BH	UR1R_DONE[15:8]								0000,0000
DMA_UR2T_AMTH	UR2T_DMA 传输总字节数	7EFA8CH	UR2T_AMT[15:8]								0000,0000
DMA_UR2T_DONEH	UR2T_DMA 传输完成字节数	7EFA8DH	UR2T_DONE[15:8]								0000,0000
DMA_UR2R_AMTH	UR2R_DMA 传输总字节数	7EFA8EH	UR2R_AMT[15:8]								0000,0000
DMA_UR2R_DONEH	UR2R_DMA 传输完成字节数	7EFA8FH	UR2R_DONE[15:8]								0000,0000
DMA_UR3T_AMTH	UR3T_DMA 传输总字节数	7EFA90H	UR3T_AMT[15:8]								0000,0000
DMA_UR3T_DONEH	UR3T_DMA 传输完成字节数	7EFA91H	UR3T_DONE[15:8]								0000,0000
DMA_UR3R_AMTH	UR3R_DMA 传输总字节数	7EFA92H	UR3R_AMT[15:8]								0000,0000
DMA_UR3R_DONEH	UR3R_DMA 传输完成字节数	7EFA93H	UR3R_DONE[15:8]								0000,0000
DMA_UR4T_AMTH	UR4T_DMA 传输总字节数	7EFA94H	UR4T_AMT[15:8]								0000,0000
DMA_UR4T_DONEH	UR4T_DMA 传输完成字节数	7EFA95H	UR4T_DONE[15:8]								0000,0000
DMA_UR4R_AMTH	UR4R_DMA 传输总字节数	7EFA96H	UR4R_AMT[15:8]								0000,0000
DMA_UR4R_DONEH	UR4R_DMA 传输完成字节数	7EFA97H	UR4R_DONE[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_I2CT_CFG	I2CT_DMA 配置寄存器	7EFA98H	I2CTIE	-	-	-	I2CTIP[1:0]		I2CTPTY[1:0]		0xxx,0000
DMA_I2CT_CR	I2CT_DMA 控制寄存器	7EFA99H	ENI2CT	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_I2CT_STA	I2CT_DMA 状态寄存器	7EFA9AH	-	-	-	-	-	TXOVW	-	I2CTIF	xxxx,x0x0
DMA_I2CT_AMT	I2CT_DMA 传输总字节数	7EFA9BH	I2CT_AMT[7:0]								0000,0000
DMA_I2CT_DONE	I2CT_DMA 传输完成字节数	7EFA9CH	I2CT_DONE[7:0]								0000,0000
DMA_I2CT_TXAH	I2CT_DMA 发送高地址	7EFA9DH	I2CT_TXA[15:8]								0000,0000
DMA_I2CT_TXAL	I2CT_DMA 发送低地址	7EFA9EH	I2CT_TXA[7:0]								0000,0000
DMA_I2CR_CFG	I2CR_DMA 配置寄存器	7EFAA0H	I2CRIE	-	-	-	I2CRIP[1:0]		I2CRPTY[1:0]		0xxx,0000
DMA_I2CR_CR	I2CR_DMA 控制寄存器	7EFAA1H	ENI2CR	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_I2CR_STA	I2CR_DMA 状态寄存器	7EFAA2H	-	-	-	-	-	-	RXLOSS	I2CRIF	xxxx,xx00
DMA_I2CR_AMT	I2CR_DMA 传输总字节数	7EFAA3H	I2CR_AMT[7:0]								0000,0000
DMA_I2CR_DONE	I2CR_DMA 传输完成字节数	7EFAA4H	I2CR_DONE[7:0]								0000,0000
DMA_I2CR_RXAH	I2CR_DMA 接收高地址	7EFAA5H	I2CR_RXA[15:8]								0000,0000
DMA_I2CR_RXAL	I2CR_DMA 接收低地址	7EFAA6H	I2CR_RXA[7:0]								0000,0000
DMA_I2CT_AMTH	I2CT_DMA 传输总字节数	7EFAA8H	I2CT_AMT[15:8]								0000,0000
DMA_I2CT_DONEH	I2CT_DMA 传输完成字节数	7EFAA9H	I2CT_DONE[15:8]								0000,0000
DMA_I2CR_AMTH	I2CR_DMA 传输总字节数	7EFAAAH	I2CR_AMT[15:8]								0000,0000
DMA_I2CR_DONEH	I2CR_DMA 传输完成字节数	7EFAABH	I2CR_DONE[15:8]								0000,0000
DMA_I2C_CR	I2C_DMA 控制寄存器	7EFAADH	RDSEL	-	-	-	-	ACKERR	INTEN	BMMEN	0xxx,x000
DMA_I2C_ST1	I2C_DMA 状态寄存器	7EFAAEH	COUNT[7:0]								0000,0000
DMA_I2C_ST2	I2C_DMA 状态寄存器	7EFAAFH	COUNT[15:8]								0000,0000
DMA_I2ST_CFG	I2ST_DMA 配置寄存器	7EFAB0H	I2STIE	-	-	-	I2STIP[1:0]		I2STPTY[1:0]		0xxx,0000
DMA_I2ST_CR	I2ST_DMA 控制寄存器	7EFAB1H	ENI2ST	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_I2ST_STA	I2ST_DMA 状态寄存器	7EFAB2H	-	-	-	-	-	TXOVW	-	I2STIF	xxxx,x0x0
DMA_I2ST_AMT	I2ST_DMA 传输总字节数	7EFAB3H	I2ST_AMT[7:0]								0000,0000
DMA_I2ST_DONE	I2ST_DMA 传输完成字节数	7EFAB4H	I2ST_DONE[7:0]								0000,0000
DMA_I2ST_TXAH	I2ST_DMA 发送高地址	7EFAB5H	I2ST_TXA[15:8]								0000,0000
DMA_I2ST_TXAL	I2ST_DMA 发送低地址	7EFAB6H	I2ST_TXA[7:0]								0000,0000
DMA_I2SR_CFG	I2SR_DMA 配置寄存器	7EFAB8H	I2SRIE	-	-	-	I2SRIP[1:0]		I2SRPTY[1:0]		0xxx,0000
DMA_I2SR_CR	I2SR_DMA 控制寄存器	7EFAB9H	ENI2SR	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_I2SR_STA	I2SR_DMA 状态寄存器	7EFABAH	-	-	-	-	-	-	RXLOSS	I2SRIF	xxxx,xx00
DMA_I2SR_AMT	I2SR_DMA 传输总字节数	7EFABBH	I2SR_AMT[7:0]								0000,0000
DMA_I2SR_DONE	I2SR_DMA 传输完成字节数	7EFABCH	I2SR_DONE[7:0]								0000,0000
DMA_I2SR_RXAH	I2SR_DMA 接收高地址	7EFABDH	I2SR_RXA[15:8]								0000,0000
DMA_I2SR_RXAL	I2SR_DMA 接收低地址	7EFABEH	I2SR_RXA[7:0]								0000,0000
DMA_I2ST_AMTH	I2ST_DMA 传输总字节数	7EFAC0H	I2ST_AMT[15:8]								0000,0000
DMA_I2ST_DONEH	I2ST_DMA 传输完成字节数	7EFAC1H	I2ST_DONE[15:8]								0000,0000
DMA_I2SR_AMTH	I2SR_DMA 传输总字节数	7EFAC2H	I2SR_AMT[15:8]								0000,0000
DMA_I2SR_DONEH	I2SR_DMA 传输完成字节数	7EFAC3H	I2SR_DONE[15:8]								0000,0000
DMA_I2C_ITVH	I2C_DMA 时间间隔寄存器	7EFAC4H	I2C_ITV[15:8]								0000,0000
DMA_I2C_ITVL	I2C_DMA 时间间隔寄存器	7EFAC5H	I2C_ITV[7:0]								0000,0000
DMA_I2S_ITVH	I2S_DMA 时间间隔寄存器	7EFAC6H	I2S_ITV[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_I2S_ITVL	I2S_DMA 时间间隔寄存器	7EFAC7H	I2S_ITV[7:0]								0000,0000
DMA_UR1_ITVH	UR1_DMA 时间间隔寄存器	7EFAC8H	UR1_ITV[15:8]								0000,0000
DMA_UR1_ITVL	UR1_DMA 时间间隔寄存器	7EFAC9H	UR1_ITV[7:0]								0000,0000
DMA_UR2_ITVH	UR2_DMA 时间间隔寄存器	7EFACAH	UR2_ITV[15:8]								0000,0000
DMA_UR2_ITVL	UR2_DMA 时间间隔寄存器	7EFACBH	UR2_ITV[7:0]								0000,0000
DMA_UR3_ITVH	UR3_DMA 时间间隔寄存器	7EFACCH	UR3_ITV[15:8]								0000,0000
DMA_UR3_ITVL	UR3_DMA 时间间隔寄存器	7EFACDH	UR3_ITV[7:0]								0000,0000
DMA_UR4_ITVH	UR4_DMA 时间间隔寄存器	7EFACEH	UR4_ITV[15:8]								0000,0000
DMA_UR4_ITVL	UR4_DMA 时间间隔寄存器	7EFACFH	UR4_ITV[7:0]								0000,0000
DMA_QSPI_CFG	QSPI_DMA 配置寄存器	7EFAD0H	QSPIIE	ACT_WR	ACT_RD	-	QSPIIP[1:0]		QSPIPTY[1:0]		000x,0000
DMA_QSPI_CR	QSPI_DMA 控制寄存器	7EFAD1H	ENQSPI	TRIG_W	TRIG_R	-	-	-	CLRWFIFO	CLRRFIFO	000x,xx00
DMA_QSPI_STA	QSPI_DMA 状态寄存器	7EFAD2H	-	-	-	-	-	-	-	QSPIIF	xxxx,xxx0
DMA_QSPI_AMT	QSPI_DMA 传输总字节数	7EFAD3H	QSPI_AMT[7:0]								0000,0000
DMA_QSPI_DONE	QSPI_DMA 传输完成字节数	7EFAD4H	QSPI_DONE[7:0]								0000,0000
DMA_QSPI_TXAH	QSPI_DMA 发送高地址	7EFAD5H	QSPI_TXA[15:8]								0000,0000
DMA_QSPI_TXAL	QSPI_DMA 发送低地址	7EFAD6H	QSPI_TXA[7:0]								0000,0000
DMA_QSPI_RXAH	QSPI_DMA 接收高地址	7EFAD7H	QSPI_RXA[15:8]								0000,0000
DMA_QSPI_RXAL	QSPI_DMA 接收低地址	7EFAD8H	QSPI_RXA[7:0]								0000,0000
DMA_QSPI_AMTH	QSPI_DMA 传输总字节数	7EFADBH	QSPI_AMT[15:8]								0000,0000
DMA_QSPI_DONEH	QSPI_DMA 传输完成字节数	7EFADCH	QSPI_DONE[15:8]								0000,0000
DMA_QSPI_ITVH	QSPI_DMA 时间间隔寄存器	7EFADEH	QSPI_ITV[15:8]								0000,0000
DMA_QSPI_ITVL	QSPI_DMA 时间间隔寄存器	7EFADFH	QSPI_ITV[7:0]								0000,0000
DMA_ADC2_CFG	ADC2_DMA 配置寄存器	7EFAE0H	ADC2IE	-	-	-	ADC2MIP[1:0]		ADC2PTY[1:0]		0xxx,0000
DMA_ADC2_CR	ADC2_DMA 控制寄存器	7EFAE1H	ENADC2	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_ADC2_STA	ADC2_DMA 状态寄存器	7EFAE2H	-	-	-	-	-	-	-	ADC2IF	xxxx,xxx0
DMA_ADC2_CFG2	ADC2_DMA 配置寄存器 2	7EFAE3H	-	-	-	-	CVTIMESEL[3:0]				xxxx,0000
DMA_ADC2_AMTH	ADC2_DMA 传输总字节数	7EFAE4H	ADC2_AMT[15:8]								0000,0000
DMA_ADC2_AMT	ADC2_DMA 传输总字节数	7EFAE5H	ADC2_AMT[7:0]								0000,0000
DMA_ADC2_DONEH	ADC2_DMA 传输完成字节数	7EFAE6H	ADC2_DONE[15:8]								0000,0000
DMA_ADC2_DONE	ADC2_DMA 传输完成字节数	7EFAE7H	ADC2_DONE[7:0]								0000,0000
DMA_ADC2_RXAH	ADC2_DMA 接收高地址	7EFAE8H	ADC2_RXA[15:8]								0000,0000
DMA_ADC2_RXAL	ADC2_DMA 接收低地址	7EFAE9H	ADC2_RXA[7:0]								0000,0000
DMA_ADC2_CHSW0	ADC2_DMA 通道使能	7EFAEAH	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0	0000,0001
DMA_ADC2_CHSW1	ADC2_DMA 通道使能	7EFAEBH	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	1000,0000
DMA_ADC2_ITVH	ADC2_DMA 时间间隔寄存器	7EFAEEH	ADC2_ITV[15:8]								0000,0000
DMA_ADC2_ITVL	ADC2_DMA 时间间隔寄存器	7EFAEFH	ADC2_ITV[7:0]								0000,0000
DMA_P2P1_DEST	P2P1_DMA 目标外设寄存器	7EFAF0H	P2P1_DEST[7:0]								0000,0000
DMA_P2P1_SRC	P2P1_DMA 源外设寄存器	7EFAF1H	P2P1_SRC[7:0]								0000,0000
DMA_P2P2_DEST	P2P2_DMA 目标外设寄存器	7EFAF2H	P2P2_DEST[7:0]								0000,0000
DMA_P2P2_SRC	P2P2_DMA 源外设寄存器	7EFAF3H	P2P2_SRC[7:0]								0000,0000
DMA_ARB_CFG	DMA 仲裁配置寄存器	7EFAF8H	WTRREN	-	-	-	STASEL[3:0]-				0xxx,0000
DMA_ARB_STA	DMA 仲裁状态寄存器	7EFAF9H	ARB_STA[7:0]								0000,0000

13.8 扩展特殊功能寄存器列表（XFR：0x7EF900-0x7EF9FF）

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
QSPI_CR1	QSPI 控制寄存器 1	7EF900H	-	-	-	-	-	-	ABORT	EN	xxxx,xx00
QSPI_CR2	QSPI 控制寄存器 2	7EF901H	-	-	-	FTHRES[4:0]					xxx0,0000
QSPI_CR3	QSPI 控制寄存器 3	7EF902H	PMM	APMS	-	-	SMIE	FTIE	TCIE	TEIE	00xx,0000
QSPI_CR4	QSPI 控制寄存器 4	7EF903H	PSCR[7:0]								0000,0000
QSPI_DCR1	QSPI 器件配置寄存器 1	7EF904H	-	CSHT[2:0]			QSPI_IP[1:0]		-	CKMODE	x000,00x0
QSPI_DCR2	QSPI 器件配置寄存器 2	7EF905H	-	-	-	FSIZE[4:0]					0000,0000
QSPI_SR1	QSPI 状态寄存器 1	7EF906H	-	-	BUSY	-	SMF	FTF	TCF	TEF	xx0x,0100
QSPI_SR2	QSPI 状态寄存器 2	7EF907H	-	-	FLEVEL[5:0]						xx00,0000
QSPI_FCR	QSPI 标志清零寄存器	7EF908H	-	-	-	-	CSMF	-	CTCF	CTEF	xxxx,0x00
QSPI_DLR1	QSPI 数据长度寄存器 1	7EF910H	DL[7:0]								0000,0000
QSPI_DLR2	QSPI 数据长度寄存器 2	7EF911H	DL[15:8]								0000,0000
QSPI_CCR1	QSPI 通信配置寄存器 1	7EF914H	INSTRUCTION[7:0]								0000,0000
QSPI_CCR2	QSPI 通信配置寄存器 2	7EF915H	ABMODE[1:0]		ADSIZE[1:0]		ADMODE[1:0]		IMODE[1:0]		0000,0000
QSPI_CCR3	QSPI 通信配置寄存器 3	7EF916H	-	DCYC[4:0]					ABSIZE[1:0]		x000,0000
QSPI_CCR4	QSPI 通信配置寄存器 4	7EF917H	-	-	-	SIOO	FMODE[1:0]		DMODE[1:0]		xxx0,0000
QSPI_AR1	QSPI 地址寄存器 1	7EF918H	ADR[7:0]								0000,0000
QSPI_AR2	QSPI 地址寄存器 2	7EF919H	ADR[15:8]								0000,0000
QSPI_AR3	QSPI 地址寄存器 3	7EF91AH	ADR[23:16]								0000,0000
QSPI_AR4	QSPI 地址寄存器 4	7EF91BH	ADR[31:24]								0000,0000
QSPI_ABR1	QSPI 交替字节寄存器 1	7EF91CH	ALTERNATED[7:0]								0000,0000
QSPI_ABR2	QSPI 交替字节寄存器 2	7EF91DH	ALTERNATED[15:8]								0000,0000
QSPI_ABR3	QSPI 交替字节寄存器 3	7EF91EH	ALTERNATED[23:16]								0000,0000
QSPI_ABR4	QSPI 交替字节寄存器 4	7EF91FH	ALTERNATED[31:24]								0000,0000
QSPI_DR	QSPI 数据寄存器	7EF920H	DATA[7:0]								0000,0000
QSPI_PSMKR1	QSPI 状态屏蔽寄存器	7EF924H	MASK[7:0]								0000,0000
QSPI_PSMAR1	QSPI 状态匹配寄存器	7EF928H	MATCH[7:0]								0000,0000
QSPI_PIR1	QSPI 轮询间隔寄存器 1	7EF92CH	INTERVAL[7:0]								0000,0000
QSPI_PIR2	QSPI 轮询间隔寄存器 2	7EF92DH	INTERVAL[15:8]								0000,0000
PWMA_ENO2	PWMA 输出使能寄存器 2	7EF930H	-	-	-	-	-	ENOA6	-	ENOA5	xxxx,x0x0
PWMA_IOAUX2	PWMA 输出附加寄存器 2	7EF931H	-	-	-	-	-	AUXA6	-	AUXA6	xxxx,x0x0
PWMA_CR3	PWMA 控制寄存器 3	7EF932H	MMSA2[3:0]				-	OISA6	-	OISA5	0000,x0x0
PWMA_SR3	PWMA 状态寄存器 3	7EF933H	-	-	-	-	-	-	CCA6IF	CCA5IF	xxxx,xx00
PWMA_CCER3	PWMA 捕获比较使能寄存器 3	7EF934H	-	-	CCA6P	CCA6E	-	-	CCA5P	CCA5E	xx00,xx00
PWMA_CCMR1X	PWMA 捕获比较模式寄存器 1x	7EF938H	-	-	-	-	-	-	-	OCA1M[3]	xxx,xxx0
PWMA_CCMR2X	PWMA 捕获比较模式寄存器 2x	7EF939H	-	-	-	-	-	-	-	OCA2M[3]	xxx,xxx0
PWMA_CCMR3X	PWMA 捕获比较模式寄存器 3x	7EF93AH	-	-	-	-	-	-	-	OCA3M[3]	xxx,xxx0
PWMA_CCMR4X	PWMA 捕获比较模式寄存器 4x	7EF93BH	-	-	-	-	-	-	-	OCA4M[3]	0000,0000
PWMA_CCMR5	PWMA 捕获比较模式寄存器 5	7EF93CH	OCA5CE	OCA5M[2:0]			OCA5PE	-	-	-	0000,0xxx
PWMA_CCMR5X	PWMA 捕获比较模式寄存器 5x	7EF93DH	-	-	-	-	-	-	-	OCA5M[3]	xxxx,xxx0
PWMA_CCMR6	PWMA 捕获比较模式寄存器 6	7EF93EH	OCA6CE	OCA6M[2:0]			OCA6PE	-	-	-	000x,xxxx

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PWMA_CCMR6X	PWMA 捕获比较模式寄存器 6x	7EF93FH	-	-	-	-	-	-	-	OCA6M[3]	xxx,xxx0
PWMA_CCR5H	PWMA 捕获比较寄存器 5	7EF940H	CCRA5[15:8]								0000,0000
PWMA_CCR5L	PWMA 捕获比较寄存器 5	7EF941H	CCRA5[7:0]								0000,0000
PWMA_CCR5X	PWMA 捕获比较寄存器 5	7EF942H	GCA5C3	GCA5C2	GCA5C1	-	-	-	-	-	000x,xxx
PWMA_CCR6H	PWMA 捕获比较寄存器 6	7EF943H	CCRA6[15:8]								0000,0000
PWMA_CCR6L	PWMA 捕获比较寄存器 6	7EF944H	CCRA6[7:0]								0000,0000
PWMA_DER	PWMA_DMA 事件使能寄存器	7EF948H	-	TDEA	COMDEA	CCA4DE	CCA3DE	CCA2DE	CCA1DE	UDEA	x000,0000
PWMA_DBA	PWMA_DMA 基址寄存器	7EF949H	-	-	-	DBAA[4:0]					xxx0,0000
PWMA_DBL	PWMA_DMA 传输长度寄存器	7EF94AH	-	-	-	DBLA[4:0]					xxx0,0000
PWMA_DMACR	PWMA_DMA 控制寄存器	7EF94BH	-	-	-	DSKIPA	DDIRA	DMAENA	SIZEA[1:0]		xxx0,0000
PWMC_ENO2	PWMC 输出使能寄存器 2	7EF950H	-	-	-	-	-	ENOC6	-	ENOC5	xxxx,x0x0
PWMC_IOAUX2	PWMC 输出附加寄存器 2	7EF951H	-	-	-	-	-	AUXC6	-	AUXC6	xxxx,x0x0
PWMC_CR3	PWMC 控制寄存器 3	7EF952H	MMSC2[3:0]				-	OISC6	-	OISC5	0000,x0x0
PWMC_SR3	PWMC 状态寄存器 3	7EF953H	-	-	-	-	-	-	CCC6IF	CCC5IF	xxxx,xx00
PWMC_CCER3	PWMC 捕获比较使能寄存器 3	7EF954H	-	-	CCC6P	CCC6E	-	-	CCC5P	CCC5E	xx00,xx00
PWMC_CCMR1X	PWMC 捕获比较模式寄存器 1x	7EF958H	-	-	-	-	-	-	-	OCC1M[3]	xxx,xxx0
PWMC_CCMR2X	PWMC 捕获比较模式寄存器 2x	7EF959H	-	-	-	-	-	-	-	OCC2M[3]	xxx,xxx0
PWMC_CCMR3X	PWMC 捕获比较模式寄存器 3x	7EF95AH	-	-	-	-	-	-	-	OCC3M[3]	xxx,xxx0
PWMC_CCMR4X	PWMC 捕获比较模式寄存器 4x	7EF95BH	-	-	-	-	-	-	-	OCC4M[3]	0000,0000
PWMC_CCMR5	PWMC 捕获比较模式寄存器 5	7EF95CH	OCC5CE	OCC5M[2:0]			OCC5PE	-	-	-	0000,0xxx
PWMC_CCMR5X	PWMC 捕获比较模式寄存器 5x	7EF95DH	-	-	-	-	-	-	-	OCC5M[3]	xxxx,xxx0
PWMC_CCMR6	PWMC 捕获比较模式寄存器 6	7EF95EH	OCC6CE	OCC6M[2:0]			OCC6PE	-	-	-	000x,xxxx
PWMC_CCMR6X	PWMC 捕获比较模式寄存器 6x	7EF95FH	-	-	-	-	-	-	-	OCC6M[3]	xxx,xxx0
PWMC_CCR5H	PWMC 捕获比较寄存器 5	7EF960H	CCRC5[15:8]								0000,0000
PWMC_CCR5L	PWMC 捕获比较寄存器 5	7EF961H	CCRC5[7:0]								0000,0000
PWMC_CCR5X	PWMC 捕获比较寄存器 5	7EF962H	GCC5C3	GCC5C2	GCC5C1	-	-	-	-	-	000x,xxx
PWMC_CCR6H	PWMC 捕获比较寄存器 6	7EF963H	CCRC6[15:8]								0000,0000
PWMC_CCR6L	PWMC 捕获比较寄存器 6	7EF964H	CCRC6[7:0]								0000,0000
PWMC_DER	PWMC_DMA 事件使能寄存器	7EF968H	-	TDEC	COMDEC	CCC4DE	CCC3DE	CCC2DE	CCC1DE	UDEC	x000,0000
PWMC_DBA	PWMC_DMA 基址寄存器	7EF969H	-	-	-	DBAC[4:0]					xxx0,0000
PWMC_DBL	PWMC_DMA 传输长度寄存器	7EF96AH	-	-	-	DBLC[4:0]					xxx0,0000
PWMC_DMACR	PWMC_DMA 控制寄存器	7EF96BH	-	-	-	DSKIPC	DDIRC	DMAENC	SIZEC[1:0]		xxx0,0000
HSPWMC_CFG	高速 PWMC 配置寄存器	7EF970H	-	-	-	PWMC_EXTN	-	PWMC_INTEN	PWMC_ASYNCEN	1	xxx0,x001
HSPWMC_ADR	高速 PWMC 地址寄存器	7EF971H	PWMC_RW	PWMC_ADDR[6:0]							0000,0000
HSPWMC_DAT	高速 PWMC 数据寄存器	7EF972H	PWMC_DATA[7:0]								0000,0000
HSPWMD_CFG	高速 PWMD 配置寄存器	7EF974H	-	-	-	-	-	PWMD_INTEN	PWMD_ASYNCEN	1	xxxx,x001
HSPWMD_ADR	高速 PWMD 地址寄存器	7EF975H	PWMD_RW	PWMD_ADDR[6:0]							0000,0000
HSPWMD_DAT	高速 PWMD 数据寄存器	7EF976H	PWMD_DATA[7:0]								0000,0000
DMA_PWMAT_CFG	PWMAT_DMA 配置寄存器	7EF980H	PWMATIE	-	-	-	PWMATIP[1:0]		PWMATPTY[1:0]		0xxx,0000
DMA_PWMAT_CR	PWMAT_DMA 控制寄存器	7EF981H	ENPWMAT	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_PWMAT_STA	PWMAT_DMA 状态寄存器	7EF982H	RDOV	-	-	-	-	TXOVW	-	PWMATIF	0xxx,x0x0
DMA_PWMAT_AMTH	PWMAT_DMA 传输总字节数	7EF984H	PWMAT_AMT[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_PWMAT_AMT	PWMAT_DMA 传输总字节数	7EF985H	PWMAT_AMT[7:0]								0000,0000
DMA_PWMAT_DONEH	PWMAT_DMA 传输完成字节数	7EF986H	PWMAT_DONE[15:8]								0000,0000
DMA_PWMAT_DONE	PWMAT_DMA 传输完成字节数	7EF987H	PWMAT_DONE[7:0]								0000,0000
DMA_PWMAT_TXAH	PWMAT_DMA 发送高地址	7EF988H	PWMAT_TXA[15:8]								0000,0000
DMA_PWMAT_TXAL	PWMAT_DMA 发送低地址	7EF989H	PWMAT_TXA[7:0]								0000,0000
DMA_PWMA_ITVH	PWMA_DMA 时间间隔寄存器	7EF98EH	PWMA_ITV[15:8]								0000,0000
DMA_PWMA_ITVL	PWMA_DMA 时间间隔寄存器	7EF98FH	PWMA_ITV[7:0]								0000,0000
DMA_PWMAR_CFG	PWMAR_DMA 配置寄存器	7EF990H	PWMARIE	-	-	-	PWMARIP[1:0]		PWMARPTY[1:0]		0xxx,0000
DMA_PWMAR_CR	PWMAR_DMA 控制寄存器	7EF991H	ENPWMAR	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_PWMAR_STA	PWMAR_DMA 状态寄存器	7EF992H	WROV	-	-	-	-	-	RXLOSS	PWMARIF	0xxx,xx00
DMA_PWMAR_AMTH	PWMAR_DMA 传输总字节数	7EF994H	PWMAR_AMT[15:8]								0000,0000
DMA_PWMAR_AMT	PWMAR_DMA 传输总字节数	7EF995H	PWMAR_AMT[7:0]								0000,0000
DMA_PWMAR_DONEH	PWMAR_DMA 传输完成字节数	7EF996H	PWMAR_DONE[15:8]								0000,0000
DMA_PWMAR_DONE	PWMAR_DMA 传输完成字节数	7EF997H	PWMAR_DONE[7:0]								0000,0000
DMA_PWMAR_RXAH	PWMAR_DMA 接收高地址	7EF998H	PWMAR_RXA[15:8]								0000,0000
DMA_PWMAR_RXAL	PWMAR_DMA 接收低地址	7EF999H	PWMAR_RXA[7:0]								0000,0000
DMA_PWMCT_CFG	PWMCT_DMA 配置寄存器	7EF9A0H	PWMCTIE	-	-	-	PWMCTIP[1:0]		PWMCTPTY[1:0]		0xxx,0000
DMA_PWMCT_CR	PWMCT_DMA 控制寄存器	7EF9A1H	ENPWMCT	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_PWMCT_STA	PWMCT_DMA 状态寄存器	7EF9A2H	RDOV	-	-	-	-	TXOVW	-	PWMCTIF	0xxx,x0x0
DMA_PWMCT_AMTH	PWMCT_DMA 传输总字节数	7EF9A4H	PWMCT_AMT[15:8]								0000,0000
DMA_PWMCT_AMT	PWMCT_DMA 传输总字节数	7EF9A5H	PWMCT_AMT[7:0]								0000,0000
DMA_PWMCT_DONEH	PWMCT_DMA 传输完成字节数	7EF9A6H	PWMCT_DONE[15:8]								0000,0000
DMA_PWMCT_DONE	PWMCT_DMA 传输完成字节数	7EF9A7H	PWMCT_DONE[7:0]								0000,0000
DMA_PWMCT_TXAH	PWMCT_DMA 发送高地址	7EF9A8H	PWMCT_TXA[15:8]								0000,0000
DMA_PWMCT_TXAL	PWMCT_DMA 发送低地址	7EF9A9H	PWMCT_TXA[7:0]								0000,0000
DMA_PWMC_ITVH	PWMC_DMA 时间间隔寄存器	7EF9AEH	PWMC_ITV[15:8]								0000,0000
DMA_PWMC_ITVL	PWMC_DMA 时间间隔寄存器	7EF9AFH	PWMC_ITV[7:0]								0000,0000
DMA_PWMCR_CFG	PWMCR_DMA 配置寄存器	7EF9B0H	PWMCRIE	-	-	-	PWMCRIP[1:0]		PWMCRPTY[1:0]		0xxx,0000
DMA_PWMCR_CR	PWMCR_DMA 控制寄存器	7EF9B1H	ENPWMCR	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_PWMCR_STA	PWMCR_DMA 状态寄存器	7EF9B2H	WROV	-	-	-	-	-	RXLOSS	PWMCRIF	0xxx,xx00
DMA_PWMCR_AMTH	PWMCR_DMA 传输总字节数	7EF9B4H	PWMCR_AMT[15:8]								0000,0000
DMA_PWMCR_AMT	PWMCR_DMA 传输总字节数	7EF9B5H	PWMCR_AMT[7:0]								0000,0000
DMA_PWMCR_DONEH	PWMCR_DMA 传输完成字节数	7EF9B6H	PWMCR_DONE[15:8]								0000,0000
DMA_PWMCR_DONE	PWMCR_DMA 传输完成字节数	7EF9B7H	PWMCR_DONE[7:0]								0000,0000
DMA_PWMCR_RXAH	PWMCR_DMA 接收高地址	7EF9B8H	PWMCR_RXA[15:8]								0000,0000
DMA_PWMCR_RXAL	PWMCR_DMA 接收低地址	7EF9B9H	PWMCR_RXA[7:0]								0000,0000
P8PU	P8 口上拉电阻控制寄存器	7EF9C0H	P8PU[7:0]								0000,0000
P9PU	P9 口上拉电阻控制寄存器	7EF9C1H	P9PU[7:0]								0000,0000
PAPU	PA 口上拉电阻控制寄存器	7EF9C2H	PAPU[7:0]								0000,0000
PBPU	PB 口上拉电阻控制寄存器	7EF9C3H	PBPU[7:0]								0000,0000
P8NCS	P8 口施密特触发控制寄存器	7EF9C8H	P8NCS[7:0]								0000,0000
P9NCS	P9 口施密特触发控制寄存器	7EF9C9H	P9NCS[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PANCS	PA 口施密特触发控制寄存器	7EF9CAH	PANCS[7:0]								0000,0000
PBNCS	PB 口施密特触发控制寄存器	7EF9CBH	PBNCS[7:0]								0000,0000
P8SR	P8 口电平转换速率寄存器	7EF9D0H	P8SR[7:0]								1111,1111
P9SR	P9 口电平转换速率寄存器	7EF9D1H	P9SR[7:0]								1111,1111
PASR	PA 口电平转换速率寄存器	7EF9D2H	PASR[7:0]								1111,1111
PBSR	PB 口电平转换速率寄存器	7EF9D3H	PBSR[7:0]								1111,1111
P8DR	P8 口驱动电流控制寄存器	7EF9D8H	P8DR[7:0]								1111,1111
P9DR	P9 口驱动电流控制寄存器	7EF9D9H	P9DR[7:0]								1111,1111
PADR	PA 口驱动电流控制寄存器	7EF9DAH	PADR[7:0]								1111,1111
PBDR	PB 口驱动电流控制寄存器	7EF9DBH	PBDR[7:0]								1111,1111
P8IE	P8 口输入使能控制寄存器	7EF9E0H	P8IE[7:0]								1111,1111
P9IE	P9 口输入使能控制寄存器	7EF9E1H	P9IE[7:0]								1111,1111
PAIE	PA 口输入使能控制寄存器	7EF9E2H	PAIE[7:0]								1111,1111
PBIE	PB 口输入使能控制寄存器	7EF9E3H	PBIE[7:0]								1111,1111
P8PD	P8 口下拉电阻控制寄存器	7EF9F0H	P8PD[7:0]								0000,0000
P9PD	P9 口下拉电阻控制寄存器	7EF9F1H	P9PD[7:0]								0000,0000
PAPD	PA 口下拉电阻控制寄存器	7EF9F2H	PAPD[7:0]								0000,0000
PBPD	PB 口下拉电阻控制寄存器	7EF9F3H	PBPD[7:0]								0000,0000
P8BP	P8 口模式用户控制寄存器	7EF9F8H	P8BP[7:0]								1111,1111
P9BP	P9 口模式用户控制寄存器	7EF9F9H	P9BP[7:0]								1111,1111
PABP	PA 口模式用户控制寄存器	7EF9FAH	PABP[7:0]								1111,1111
PBBP	PB 口模式用户控制寄存器	7EF9FBH	PBBP[7:0]								1111,1111

13.9 扩展特殊功能寄存器列表（XFR：0x7EF800-0x7EF8FF）

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
SPI2CTL	SPI2 控制寄存器	7EF800H	SS2IG	SP2EN	SP2DORD	SP2MSTR	SP2CPOL	SP2CPHA	SP2R[1:0]		0000,1100
SPI2STAT	SPI2 状态寄存器	7EF801H	SP2IF	SP2WCOL	-	-	-	-	-	-	00xx,xxxx
SPI2DAT	SPI2 数据寄存器	7EF802H	SP2DAT[7:0]								0000,0000
SPI2TOTE	SPI2 从机超时长度寄存器	7EF803H	SPI2TM[23:16]								0000,0000
SPI2TOCR	SPI2 从机超时控制寄存器	7EF804H	SPI2ENTO	SPI2ENTOI	SPI2SCALE	-	-	-	-	-	000x,xxxx
SPI2TOSR	SPI2 从机超时状态寄存器	7EF805H	SPI2CTOIF	-	-	-	-	-	-	SPI2TOIF	0xxx,xxx0
SPI2TOTH	SPI2 从机超时长度寄存器	7EF806H	SPI2TM[15:8]								0000,0000
SPI2TOTL	SPI2 从机超时长度寄存器	7EF807H	SPI2TM[7:0]								0000,0000
SPI2_CLKDIV	SPI2 时钟预分配寄存器	7EF808H	SPI2_CLKDIV[7:0]								0000,0000
HSSPI2_CFG	高速 SPI2 配置寄存器	7EF809H	SS2_HLD[3:0]				SS2_SETUP[3:0]				0011,0011
HSSPI2_CFG2	高速 SPI2 配置寄存器 2	7EF80AH	MOSI2_PL	SPI2_IOSW	HSSPI2_EN	HSSPI2_FIFOEN	SS2_DACT[3:0]				0000,0011
HSSPI2_STA	高速 SPI2 状态寄存器	7EF80BH	-	-	-	-	TXFULL	TXEMPT	RXFULL	RXEMPT	xxxx,0000
HSSPI2_PSCR	高速 SPI2 时钟分频寄存器	7EF80CH	HSSPI2_PSCR [7:0]								0000,0000
SPI2_INTR	SPI2 中断控制寄存器	7EF80FH	-	-	PSPI2[1:0]	-	-	-	ESPI2	xx00,xxx0	
DMA_SPI2_CFG	SPI2_DMA 配置寄存器	7EF810H	SPI2IE	ACT_TX	ACT_RX	-	SPI2IP[1:0]		SPI2PTY[1:0]		000x,0000
DMA_SPI2_CR	SPI2_DMA 控制寄存器	7EF811H	ENSPI2	TRIG_M	TRIG_S	-	-	-	-	CLRFIFO	000x,xxx0
DMA_SPI2_STA	SPI2_DMA 状态寄存器	7EF812H	-	-	-	-	-	TXOVW	RXLOSS	SPI2IF	xxxx,x000
DMA_SPI2_CFG2	SPI2_DMA 配置寄存器 2	7EF813H	-	-	-	-	-	WRPSS	SSS[1:0]		xxxx,x000
DMA_SPI2_AMTH	SPI2_DMA 传输总字节数	7EF814H	SPI2_AMT[15:8]								0000,0000
DMA_SPI2_AMT	SPI2_DMA 传输总字节数	7EF815H	SPI2_AMT[7:0]								0000,0000
DMA_SPI2_DONEH	SPI2_DMA 传输完成字节数	7EF816H	SPI2_DONE[15:8]								0000,0000
DMA_SPI2_DONE	SPI2_DMA 传输完成字节数	7EF817H	SPI2_DONE[7:0]								0000,0000
DMA_SPI2_TXAH	SPI2_DMA 发送高地址	7EF818H	SPI2_TXA[15:8]								0000,0000
DMA_SPI2_TXAL	SPI2_DMA 发送低地址	7EF819H	SPI2_TXA[7:0]								0000,0000
DMA_SPI2_RXAH	SPI2_DMA 接收高地址	7EF81AH	SPI2_RXA[15:8]								0000,0000
DMA_SPI2_RXAL	SPI2_DMA 接收低地址	7EF81BH	SPI2_RXA[7:0]								0000,0000
DMA_SPI2_ITVH	SPI2_DMA 时间间隔寄存器	7EF81EH	SPI2_ITV[15:8]								0000,0000
DMA_SPI2_ITVL	SPI2_DMA 时间间隔寄存器	7EF81FH	SPI2_ITV[7:0]								0000,0000
SPI3CTL	SPI3 控制寄存器	7EF820H	SS3IG	SP3EN	SP3DORD	SP3MSTR	SP3CPOL	SP3CPHA	SP3R[1:0]		0000,1100
SPI3STAT	SPI3 状态寄存器	7EF821H	SP3IF	SP3WCOL	-	-	-	-	-	-	00xx,xxxx
SPI3DAT	SPI3 数据寄存器	7EF822H	SP3DAT[7:0]								0000,0000
SPI3TOTE	SPI3 从机超时长度寄存器	7EF823H	SPI3TM[23:16]								0000,0000
SPI3TOCR	SPI3 从机超时控制寄存器	7EF824H	SPI3ENTO	SPI3ENTOI	SPI3SCALE	-	-	-	-	-	000x,xxxx
SPI3TOSR	SPI3 从机超时状态寄存器	7EF825H	SPI3CTOIF	-	-	-	-	-	-	SPI3TOIF	0xxx,xxx0
SPI3TOTH	SPI3 从机超时长度寄存器	7EF826H	SPI3TM[15:8]								0000,0000
SPI3TOTL	SPI3 从机超时长度寄存器	7EF827H	SPI3TM[7:0]								0000,0000
SPI3_CLKDIV	SPI3 时钟预分配寄存器	7EF828H	SPI3_CLKDIV[7:0]								0000,0000
HSSPI3_CFG	高速 SPI3 配置寄存器	7EF829H	SS3_HLD[3:0]				SS3_SETUP[3:0]				0011,0011
HSSPI3_CFG2	高速 SPI3 配置寄存器 2	7EF82AH	MOSI3_PL	SPI3_IOSW	HSSPI3_EN	HSSPI3_FIFOEN	SS3_DACT[3:0]				0000,0011
HSSPI3_STA	高速 SPI3 状态寄存器	7EF82BH	-	-	-	-	TXFULL	TXEMPT	RXFULL	RXEMPT	xxxx,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
HSSPI3_PSCR	高速 SPI3 时钟分频寄存器	7EF82CH	HSSPI3_PSCR [7:0]								0000,0000
SPI3_INTR	SPI3 中断控制寄存器	7EF82FH	-	-	PSPI3[1:0]		-	-	ESPI3	xx00,xxx0	
DMA_SPI3_CFG	SPI3_DMA 配置寄存器	7EF830H	SPI3IE	ACT_TX	ACT_RX	-	SPI3IP[1:0]		SPI3PTY[1:0]		000x,0000
DMA_SPI3_CR	SPI3_DMA 控制寄存器	7EF831H	ENSPI3	TRIG_M	TRIG_S	-	-	-	-	CLRFIFO	000x,xxx0
DMA_SPI3_STA	SPI3_DMA 状态寄存器	7EF832H	-	-	-	-	-	TXOVW	RXLOSS	SPI3IF	xxxx,x000
DMA_SPI3_CFG2	SPI3_DMA 配置寄存器 2	7EF833H	-	-	-	-	-	WRPSS	SSS[1:0]		xxxx,x000
DMA_SPI3_AMTH	SPI3_DMA 传输总字节数	7EF834H	SPI3_AMT[15:8]								0000,0000
DMA_SPI3_AMT	SPI3_DMA 传输总字节数	7EF835H	SPI3_AMT[7:0]								0000,0000
DMA_SPI3_DONEH	SPI3_DMA 传输完成字节数	7EF836H	SPI3_DONE[15:8]								0000,0000
DMA_SPI3_DONE	SPI3_DMA 传输完成字节数	7EF837H	SPI3_DONE[7:0]								0000,0000
DMA_SPI3_TXAH	SPI3_DMA 发送高地址	7EF838H	SPI3_TXA[15:8]								0000,0000
DMA_SPI3_TXAL	SPI3_DMA 发送低地址	7EF839H	SPI3_TXA[7:0]								0000,0000
DMA_SPI3_RXAH	SPI3_DMA 接收高地址	7EF83AH	SPI3_RXA[15:8]								0000,0000
DMA_SPI3_RXAL	SPI3_DMA 接收低地址	7EF83BH	SPI3_RXA[7:0]								0000,0000
DMA_SPI3_ITVH	SPI3_DMA 间隔寄存器	7EF83EH	SPI3_ITV[15:8]								0000,0000
DMA_SPI3_ITVL	SPI3_DMA 间隔寄存器	7EF83FH	SPI3_ITV[7:0]								0000,0000
I2C2CFG	I²C 配置寄存器	7EF860H	ENI2C2	I2C2MSSL	I2C2MSSPEED[5:0]						0000,0000
I2C2MSCR	I²C 主机控制寄存器	7EF861H	EI2C2MSI	-	-	-	I2C2MSCMD[3:0]				0xxx,0000
I2C2MSST	I²C 主机状态寄存器	7EF862H	I2C2MSBUSY	I2C2MSIF	-	-	-	-	I2C2MSACKI	I2C2MSACKO	00xx,xx10
I2C2SLCR	I²C 从机控制寄存器	7EF863H	-	EI2C2STAI	EI2C2RXI	EI2C2TXI	EI2C2STOI	-	-	I2C2SLRST	x000,0xx0
I2C2SLST	I²C 从机状态寄存器	7EF864H	I2C2SLBUSY	I2C2STAIF	I2C2RXIF	I2C2TXIF	I2C2STOIF	I2C2TXING	I2C2SLACKI	I2C2SLACKO	0000,0000
I2C2SLADR	I²C 从机地址寄存器	7EF865H	I2C2SLADR[6:0]							I2C2MA	0000,0000
I2C2TXD	I²C 数据发送寄存器	7EF866H	I2C2TXD[7:0]								0000,0000
I2C2RXD	I²C 数据接收寄存器	7EF867H	I2C2RXD[7:0]								0000,0000
I2C2MSAUX	I²C 主机辅助控制寄存器	7EF868H	-	-	-	-	-	-	-	I2C2WDTA	xxxx,xxx0
I2C2PSCR	I²C 时钟预分配寄存器	7EF869H	I2C2MSSPEED[13:6]								0000,0000
I2C2TOTE	I²C2 从机超时长度寄存器	7EF86BH	I2C2TM[23:16]								0000,0000
I2C2TOCR	I²C2 从机超时控制寄存器	7EF86CH	I2C2ENTO	I2C2ENTOI	I2C2SCALE	-	-	-	-	-	000x,xxxx
I2C2TOSR	I²C2 从机超时状态寄存器	7EF86DH	-	-	-	-	-	-	-	I2C2TOIF	xxxx,xxx0
I2C2TOTH	I²C2 从机超时长度寄存器	7EF86EH	I2C2TM[15:8]								0000,0000
I2C2TOTL	I²C2 从机超时长度寄存器	7EF86FH	I2C2TM[7:0]								0000,0000
DMA_I2C2T_CFG	I2C2T_DMA 配置寄存器	7EF870H	I2C2TIE	-	-	-	I2C2TIP[1:0]		I2C2TPTY[1:0]		0xxx,0000
DMA_I2C2T_CR	I2C2T_DMA 控制寄存器	7EF871H	ENI2C2T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_I2C2T_STA	I2C2T_DMA 状态寄存器	7EF872H	-	-	-	-	-	TXOVW	-	I2C2TIF	xxxx,x0x0
DMA_I2C2T_AMTH	I2C2T_DMA 传输总字节数	7EF874H	I2C2T_AMT[15:8]								0000,0000
DMA_I2C2T_AMT	I2C2T_DMA 传输总字节数	7EF875H	I2C2T_AMT[7:0]								0000,0000
DMA_I2C2T_DONEH	I2C2T_DMA 传输完成字节数	7EF876H	I2C2T_DONE[15:8]								0000,0000
DMA_I2C2T_DONE	I2C2T_DMA 传输完成字节数	7EF877H	I2C2T_DONE[7:0]								0000,0000
DMA_I2C2T_TXAH	I2C2T_DMA 发送高地址	7EF878H	I2C2T_TXA[15:8]								0000,0000
DMA_I2C2T_TXAL	I2C2T_DMA 发送低地址	7EF879H	I2C2T_TXA[7:0]								0000,0000
DMA_I2C2_CR	I2C2_DMA 控制寄存器	7EF87BH	RDSEL	-	-	-	-	ACKERR	INTEN	BMMEN	0xxx,x000
DMA_I2C2_ST1	I2C2_DMA 状态寄存器	7EF87CH	COUNT[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_I2C2_ST2	I2C2_DMA 状态寄存器	7EF87DH	COUNT[15:8]								0000,0000
DMA_I2C2_ITVH	I2C2_DMA 时间间隔寄存器	7EF87EH	I2C2_ITV[15:8]								0000,0000
DMA_I2C2_ITVL	I2C2_DMA 时间间隔寄存器	7EF87FH	I2C2_ITV[7:0]								0000,0000
DMA_I2C2R_CFG	I2C2R_DMA 配置寄存器	7EF880H	I2C2RIE	-	-	-	I2C2RIP[1:0]		I2C2RPTY[1:0]		0xxx,0000
DMA_I2C2R_CR	I2C2R_DMA 控制寄存器	7EF881H	ENI2C2R	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_I2C2R_STA	I2C2R_DMA 状态寄存器	7EF882H	-	-	-	-	-	-	RXLOSS	I2C2RIF	xxxx,xx00
DMA_I2C2R_AMTH	I2C2R_DMA 传输总字节数	7EF884H	I2C2R_AMT[15:8]								0000,0000
DMA_I2C2R_AMT	I2C2R_DMA 传输总字节数	7EF885H	I2C2R_AMT[7:0]								0000,0000
DMA_I2C2R_DONEH	I2C2R_DMA 传输完成字节数	7EF886H	I2C2R_DONE[15:8]								0000,0000
DMA_I2C2R_DONE	I2C2R_DMA 传输完成字节数	7EF887H	I2C2R_DONE[7:0]								0000,0000
DMA_I2C2R_RXAH	I2C2R_DMA 接收高地址	7EF888H	I2C2R_RXA[15:8]								0000,0000
DMA_I2C2R_RXAL	I2C2R_DMA 接收低地址	7EF889H	I2C2R_RXA[7:0]								0000,0000
DMA_I2S2T_CFG	I2S2T_DMA 配置寄存器	7EF890H	I2S2TIE	-	-	-	I2S2TIP[1:0]		I2S2TPTY[1:0]		0xxx,0000
DMA_I2S2T_CR	I2S2T_DMA 控制寄存器	7EF891H	ENI2S2T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_I2S2T_STA	I2S2T_DMA 状态寄存器	7EF892H	-	-	-	-	-	TXOVW	-	I2S2TIF	xxxx,x0x0
DMA_I2S2T_AMTH	I2S2T_DMA 传输总字节数	7EF894H	I2S2T_AMT[15:8]								0000,0000
DMA_I2S2T_AMT	I2S2T_DMA 传输总字节数	7EF895H	I2S2T_AMT[7:0]								0000,0000
DMA_I2S2T_DONEH	I2S2T_DMA 传输完成字节数	7EF896H	I2S2T_DONE[15:8]								0000,0000
DMA_I2S2T_DONE	I2S2T_DMA 传输完成字节数	7EF897H	I2S2T_DONE[7:0]								0000,0000
DMA_I2S2T_TXAH	I2S2T_DMA 发送高地址	7EF898H	I2S2T_TXA[15:8]								0000,0000
DMA_I2S2T_TXAL	I2S2T_DMA 发送低地址	7EF899H	I2S2T_TXA[7:0]								0000,0000
DMA_I2S2_ITVH	I2S2T_DMA 发送高地址	7EF89EH	I2S2T_TXA[15:8]								0000,0000
DMA_I2S2_ITVL	I2S2T_DMA 发送低地址	7EF89FH	I2S2T_TXA[7:0]								0000,0000
DMA_I2S2R_CFG	I2S2R_DMA 配置寄存器	7EF8A0H	I2S2RIE	-	-	-	I2S2RIP[1:0]		I2S2RPTY[1:0]		0xxx,0000
DMA_I2S2R_CR	I2S2R_DMA 控制寄存器	7EF8A1H	ENI2S2R	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_I2S2R_STA	I2S2R_DMA 状态寄存器	7EF8A2H	-	-	-	-	-	-	RXLOSS	I2S2RIF	xxxx,xx00
DMA_I2S2R_AMTH	I2S2R_DMA 传输总字节数	7EF8A4H	I2S2R_AMT[15:8]								0000,0000
DMA_I2S2R_AMT	I2S2R_DMA 传输总字节数	7EF8A5H	I2S2R_AMT[7:0]								0000,0000
DMA_I2S2R_DONEH	I2S2R_DMA 传输完成字节数	7EF8A6H	I2S2R_DONE[15:8]								0000,0000
DMA_I2S2R_DONE	I2S2R_DMA 传输完成字节数	7EF8A7H	I2S2R_DONE[7:0]								0000,0000
DMA_I2S2R_RXAH	I2S2R_DMA 接收高地址	7EF8A8H	I2S2R_RXA[15:8]								0000,0000
DMA_I2S2R_RXAL	I2S2R_DMA 接收低地址	7EF8A9H	I2S2R_RXA[7:0]								0000,0000
PWMC_ETRPS	PWMC 的 ETR 选择寄存器	7EF8B0H	-	-	CMPCPS[1:0]		BRKCPS[1:0]		ETRCPS[1:0]		xx00,0000
PWMC_ENO	PWMC 输出使能控制	7EF8B1H	ENOC4N	ENOC4P	ENOC3N	ENOC3P	ENOC2N	ENOC2P	ENOC1N	ENOC1P	0000,0000
PWMC_PS	PWMC 输出脚选择寄存器	7EF8B2H	CC4PS[1:0]		CC3PS[1:0]		CC2PS[1:0]		CC1PS[1:0]		0000,0000
PWMC_IOAUX	PWMC 辅助寄存器	7EF8B3H	AUXC4N	AUXC4P	AUXC3N	AUXC3P	AUXC2N	AUXC2P	AUXC1N	AUXC1P	0000,0000
PWMD_ETRPS	PWMD 的 ETR 选择寄存器	7EF8B4H			CMPDPS[1:0]		BRKDPS[1:0]		ETRDPS[1:0]		xx00,0000
PWMD_ENO	PWMD 输出使能控制	7EF8B5H	-	ENOD8P	-	ENOD7P	-	ENOD6P	-	ENOD5P	x0x0,x0x0
PWMD_PS	PWMD 输出脚选择寄存器	7EF8B6H	DC8PS[1:0]		DC7PS[1:0]		DC6PS[1:0]		DC5PS[1:0]		0000,0000
PWMD_IOAUX	PWMD 辅助寄存器	7EF8B7H	-	AUXD8P	-	AUXD7P	-	AUXD6P	-	AUXD5P	x0x0,x0x0
PWMC_PS2	PWMC 输出脚选择寄存器 2	7EF8B8H	-	-	-	-	CC6PS[1:0]		CC5PS[1:0]		xxxx,0000
PWMC_RCRH	PWMC 重复计数器寄存器	7EF8B9H	REPC[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PWMD_RCRH	PWMD 重复计数器寄存器	7EF8BAH	REPD[15:8]								0000,0000
PWMC_CR1	PWMC 控制寄存器 1	7EF8C0H	ARPEC	CMSC[1:0]		DIRC	OPMC	URSC	UDISC	CENC	0000,0000
PWMC_CR2	PWMC 控制寄存器 2	7EF8C1H	-	MMSC[2:0]			-	COMSC	-	CCPCC	x000,x0x0
PWMC_SMCR	PWMC 从模式控制寄存器	7EF8C2H	MSMC	TSC[2:0]			-	SMSC[2:0]			0000,x000
PWMC_ETR	PWMC 外部触发寄存器	7EF8C3H	ETPC	ECEC	ETPSC[1:0]		ETFC[3:0]				0000,0000
PWMC_IER	PWMC 中断使能寄存器	7EF8C4H	BIEC	TIEC	COMIEC	CCC4IE	CCC3IE	CCC2IE	CCC1IE	UIEC	0000,0000
PWMC_SR1	PWMC 状态寄存器 1	7EF8C5H	BIFC	TIFC	COMIFC	CCC4IF	CCC3IF	CCC2IF	CCC1IF	UIFC	0000,0000
PWMC_SR2	PWMC 状态寄存器 2	7EF8C6H	-	-	-	CCC4OF	CCC3OF	CCC2OF	CCC1OF	-	xxx0,000x
PWMC_EGR	PWMC 事件发生寄存器	7EF8C7H	BGC	TGC	COMGC	CCC4G	CCC3G	CCC2G	CCC1G	UGC	0000,0000
PWMC_CCMR1	PWMC 捕获模式寄存器 1	7EF8C8H	OCC1CE	OCC1M[2:0]			OCC1PE	OCC1FE	CCC1S[1:0]		0000,0000
	PWMC 比较模式寄存器 1		ICC1F[3:0]				ICC1PSC[1:0]		CCC1S[1:0]		0000,0000
PWMC_CCMR2	PWMC 捕获模式寄存器 2	7EF8C9H	OCC2CE	OCC2M[2:0]			OCC2PE	OCC2FE	CCC2S[1:0]		0000,0000
	PWMC 比较模式寄存器 2		ICC2F[3:0]				ICC2PSC[1:0]		CCC2S[1:0]		0000,0000
PWMC_CCMR3	PWMC 捕获模式寄存器 3	7EF8CAH	OCC3CE	OCC3M[2:0]			OCC3PE	OCC3FE	CCC3S[1:0]		0000,0000
	PWMC 比较模式寄存器 3		ICC3F[3:0]				ICC3PSC[1:0]		CCC3S[1:0]		0000,0000
PWMC_CCMR4	PWMC 捕获模式寄存器 4	7EF8CBH	OCC4CE	OCC4M[2:0]			OCC4PE	OCC4FE	CCC4S[1:0]		0000,0000
	PWMC 比较模式寄存器 4		ICC4F[3:0]				ICC4PSC[1:0]		CCC4S[1:0]		0000,0000
PWMC_CCER1	PWMC 捕获比较使能寄存器 1	7EF8CCH	CCC2NP	CCC2NE	CCC2P	CCC2E	CCC1NP	CCC1NE	CCC1P	CCC1E	0000,0000
PWMC_CCER2	PWMC 捕获比较使能寄存器 2	7EF8CDH	CCC4NP	CCC4NE	CCC4P	CCC4E	CCC3NP	CCC3NE	CCC3P	CCC3E	0000,0000
PWMC_CNTRH	PWMC 计数器高字节	7EF8CEH	CNTC[15:8]								0000,0000
PWMC_CNTRL	PWMC 计数器低字节	7EF8CFH	CNTC[7:0]								0000,0000
PWMC_PSCRH	PWMC 预分频高字节	7EF8D0H	PSCC[15:8]								0000,0000
PWMC_PSCRL	PWMC 预分频低字节	7EF8D1H	PSCC[7:0]								0000,0000
PWMC_ARRH	PWMC 自动重装寄存器高字节	7EF8D2H	ARRC[15:8]								0000,0000
PWMC_ARRL	PWMC 自动重装寄存器低字节	7EF8D3H	ARRC[7:0]								0000,0000
PWMC_RCR	PWMC 重复计数器寄存器	7EF8D4H	REPC[7:0]								0000,0000
PWMC_CCR1H	PWMC 比较捕获寄存器 1 高位	7EF8D5H	CCRC1[15:8]								0000,0000
PWMC_CCR1L	PWMC 比较捕获寄存器 1 低位	7EF8D6H	CCRC1[7:0]								0000,0000
PWMC_CCR2H	PWMC 比较捕获寄存器 2 高位	7EF8D7H	CCR2[15:8]								0000,0000
PWMC_CCR2L	PWMC 比较捕获寄存器 2 低位	7EF8D8H	CCR2[7:0]								0000,0000
PWMC_CCR3H	PWMC 比较捕获寄存器 3 高位	7EF8D9H	CCRC3[15:8]								0000,0000
PWMC_CCR3L	PWMC 比较捕获寄存器 3 低位	7EF8DAH	CCRC3[7:0]								0000,0000
PWMC_CCR4H	PWMC 比较捕获寄存器 4 高位	7EF8DBH	CCRC4[15:8]								0000,0000
PWMC_CCR4L	PWMC 比较捕获寄存器 4 低位	7EF8DCH	CCRC4[7:0]								0000,0000
PWMC_BKR	PWMC 刹车寄存器	7EF8DDH	MOEC	AOEC	BKPC	BKEC	OSSRC	OSSIC	LOCKC[1:0]		0000,0000
PWMC_DTR	PWMC 死区控制寄存器	7EF8DEH	DTGC[7:0]								0000,0000
PWMC_OISR	PWMC 输出空闲状态寄存器	7EF8DFH	OISC4N	OISC4	OISC3N	OISC3	OISC2N	OISC2	OISC1N	OISC1	0000,0000
PWMD_CR1	PWMD 控制寄存器 1	7EF8E0H	ARPED	CMSD[1:0]		DIRD	OPMD	URSD	UDISD	CEND	0000,0000
PWMD_CR2	PWMD 控制寄存器 2	7EF8E1H	-	MMSD[2:0]			-	COMSD	-	CCPCD	x000,x0x0
PWMD_SMCR	PWMD 从模式控制寄存器	7EF8E2H	MSMD	TSD[2:0]			-	SMSD[2:0]			0000,x000
PWMD_ETR	PWMD 外部触发寄存器	7EF8E3H	ETPD	ECED	ETPSD[1:0]		ETFD[3:0]				0000,0000
PWMD_IER	PWMD 中断使能寄存器	7EF8E4H	BIED	TIED	COMIED	CCD8IE	CCD7IE	CCD6IE	CCD5IE	UIED	0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PWMD_SR1	PWMD 状态寄存器 1	7EF8E5H	BIFD	TIFD	COMIFD	CCD8IF	CCD7IF	CCD6IF	CCD5IF	UIFD	0000,0000
PWMD_SR2	PWMD 状态寄存器 2	7EF8E6H	-	-	-	CCD8OF	CCD7OF	CCD6OF	CCD5OF	-	xxx0,000x
PWMD_EGR	PWMD 事件发生寄存器	7EF8E7H	BGD	TGD	COMGD	CCD8G	CCD7G	CCD6G	CCD5G	UGD	0000,0000
PWMD_CCMR1	PWMD 捕获模式寄存器 1	7EF8E8H	OCD5CE	OCD5M[2:0]			OCD5PE	OCD5FE	CCD5S[1:0]		0000,0000
	PWMD 比较模式寄存器 1		ICD5F[3:0]				ICD5PSC[1:0]		CCD5S[1:0]		0000,0000
PWMD_CCMR2	PWMD 捕获模式寄存器 2	7EF8E9H	OCD6CE	OCD6M[2:0]			OCD6PE	OCD6FE	CCD6S[1:0]		0000,0000
	PWMD 比较模式寄存器 2		ICD6F[3:0]				ICD6PSC[1:0]		CCD6S[1:0]		0000,0000
PWMD_CCMR3	PWMD 捕获模式寄存器 3	7EF8EAH	OCD7CE	OCD7M[2:0]			OC7DPE	OCD7FE	CCD7S[1:0]		0000,0000
	PWMD 比较模式寄存器 3		ICD7F[3:0]				ICD7PSC[1:0]		CCD7S[1:0]		0000,0000
PWMD_CCMR4	PWMD 捕获模式寄存器 4	7EF8EBH	OCD8CE	OCD8M[2:0]			OCD8PE	OCD8FE	CCD8S[1:0]		0000,0000
	PWMD 比较模式寄存器 4		ICD8F[3:0]				ICD8PSC[1:0]		CCD8S[1:0]		0000,0000
PWMD_CCER1	PWMD 捕获比较使能寄存器 1	7EF8ECH	-	-	CCD6P	CCD6E	-	-	CCD5P	CCD5E	xx00,xx00
PWMD_CCER2	PWMD 捕获比较使能寄存器 2	7EF8EDH	-	-	CCD8P	CCD8E	-	-	CCD7P	CCD7E	xx00,xx00
PWMD_CNTRH	PWMD 计数器高字节	7EF8EEH	CNTD[15:8]								0000,0000
PWMD_CNTRL	PWMD 计数器低字节	7EF8EFH	CNTD[7:0]								0000,0000
PWMD_PSCRH	PWMD 预分频高字节	7EF8F0H	PSCD[15:8]								0000,0000
PWMD_PSCRL	PWMD 预分频低字节	7EF8F1H	PSCD[7:0]								0000,0000
PWMD_ARRH	PWMD 自动重装寄存器高字节	7EF8F2H	ARRD[15:8]								0000,0000
PWMD_ARRL	PWMD 自动重装寄存器低字节	7EF8F3H	ARRD[7:0]								0000,0000
PWMD_RCR	PWMD 重复计数器寄存器	7EF8F4H	REPD[7:0]								0000,0000
PWMD_CCR5H	PWMD 比较捕获寄存器 5 高位	7EF8F5H	CCRD5[15:8]								0000,0000
PWMD_CCR5L	PWMD 比较捕获寄存器 5 低位	7EF8F6H	CCRD5[7:0]								0000,0000
PWMD_CCR6H	PWMD 比较捕获寄存器 6 高位	7EF8F7H	CCRD6[15:8]								0000,0000
PWMD_CCR6L	PWMD 比较捕获寄存器 6 低位	7EF8F8H	CCRD6[7:0]								0000,0000
PWMD_CCR7H	PWMD 比较捕获寄存器 7 高位	7EF8F9H	CCRD7[15:8]								0000,0000
PWMD_CCR7L	PWMD 比较捕获寄存器 7 低位	7EF8FAH	CCRD7[7:0]								0000,0000
PWMD_CCR8H	PWMD 比较捕获寄存器 8 高位	7EF8FBH	CCRD8[15:8]								0000,0000
PWMD_CCR8L	PWMD 比较捕获寄存器 8 低位	7EF8FCH	CCRD8[7:0]								0000,0000
PWMD_BKR	PWMD 刹车寄存器	7EF8FDH	MOED	AOEDD	BKP	BKED	OSSRD	OSSID	LOCKD[1:0]		0000,0000
PWMD_DTR	PWMD 死区控制寄存器	7EF8FEH	DTGD[7:0]								0000,0000
PWMD_OISR	PWMD 输出空闲状态寄存器	7EF8FFH	-	OISD8	-	OISD7	-	OISD6	-	OISD5	x0x0,x0x0

13.10 扩展特殊功能寄存器列表（XFR：0x7EF700-0x7EF7FF）

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P8OUT	P8 端口输出寄存器	7EF700H	P87OUT	P86OUT	P85OUT	P84OUT	P83OUT	P82OUT	P81OUT	P80OUT	1111,1111
P9OUT	P9 端口输出寄存器	7EF701H	P97OUT	P96OUT	P95OUT	P94OUT	P93OUT	P92OUT	P91OUT	P90OUT	1111,1111
PAOUT	PA 端口输出寄存器	7EF702H	PA7OUT	PA6OUT	PA5OUT	PA4OUT	PA3OUT	PA2OUT	PA1OUT	PA0OUT	1111,1111
PBOUT	PB 端口输出寄存器	7EF703H	PB7OUT	PB6OUT	PB5OUT	PB4OUT	PB3OUT	PB2OUT	PB1OUT	PB0OUT	1111,1111
P8IN	P8 端口输入寄存器	7EF708H	P87IN	P86IN	P85IN	P84IN	P83IN	P82IN	P81IN	P80IN	xxxx,xxxx
P9IN	P9 端口输入寄存器	7EF709H	P97IN	P96IN	P95IN	P94IN	P93IN	P92IN	P91IN	P90IN	xxxx,xxxx
PAIN	PA 端口输入寄存器	7EF70AH	PA7IN	PA6IN	PA5IN	PA4IN	PA3IN	PA2IN	PA1IN	PA0IN	xxxx,xxxx
PBIN	PB 端口输入寄存器	7EF70BH	PB7IN	PB6IN	PB5IN	PB4IN	PB3IN	PB2IN	PB1IN	PB0IN	xxxx,xxxx
P8M0	P8 端口配置寄存器 0	7EF710H	P87M0	P86M0	P85M0	P84M0	P83M0	P82M0	P81M0	P80M0	0000,0000
P9M0	P9 端口配置寄存器 0	7EF711H	P97M0	P96M0	P95M0	P94M0	P93M0	P92M0	P91M0	P90M0	0000,0000
PAM0	PA 端口配置寄存器 0	7EF712H	PA7M0	PA6M0	PA5M0	PA4M0	PA3M0	PA2M0	PA1M0	PA0M0	0000,0000
PBM0	PB 端口配置寄存器 0	7EF713H	PB7M0	PB6M0	PB5M0	PB4M0	PB3M0	PB2M0	PB1M0	PB0M0	0000,0000
P8M1	P8 端口配置寄存器 1	7EF718H	P87M1	P86M1	P85M1	P84M1	P83M1	P82M1	P81M1	P80M1	1111,1111
P9M1	P9 端口配置寄存器 1	7EF719H	P97M1	P96M1	P95M1	P94M1	P93M1	P92M1	P91M1	P90M1	1111,1111
PAM1	PA 端口配置寄存器 1	7EF71AH	PA7M1	PA6M1	PA5M1	PA4M1	PA3M1	PA2M1	PA1M1	PA0M1	1111,1111
PBM1	PB 端口配置寄存器 1	7EF71BH	PB7M1	PB6M1	PB5M1	PB4M1	PB3M1	PB2M1	PB1M1	PB0M1	1111,1111
P8SETB	P8 端口置 1 位控制器	7EF720H	P87SETB	P86SETB	P85SETB	P84SETB	P83SETB	P82SETB	P81SETB	P80SETB	0000,0000
P9SETB	P9 端口置 1 位控制器	7EF721H	P97SETB	P96SETB	P95SETB	P94SETB	P93SETB	P92SETB	P91SETB	P90SETB	0000,0000
PASETB	PA 端口置 1 位控制器	7EF722H	PA7SETB	PA6SETB	PA5SETB	PA4SETB	PA3SETB	PA2SETB	PA1SETB	PA0SETB	0000,0000
PBSETB	PB 端口置 1 位控制器	7EF723H	PB7SETB	PB6SETB	PB5SETB	PB4SETB	PB3SETB	PB2SETB	PB1SETB	PB0SETB	0000,0000
P8CLRB	P8 端口清 0 位控制器	7EF728H	P87CLRB	P86CLRB	P85CLRB	P84CLRB	P83CLRB	P82CLRB	P81CLRB	P80CLRB	0000,0000
P9CLRB	P9 端口清 0 位控制器	7EF729H	P97CLRB	P96CLRB	P95CLRB	P94CLRB	P93CLRB	P92CLRB	P91CLRB	P90CLRB	0000,0000
PACLRB	PA 端口清 0 位控制器	7EF72AH	PA7CLRB	PA6CLRB	PA5CLRB	PA4CLRB	PA3CLRB	PA2CLRB	PA1CLRB	PA0CLRB	0000,0000
PBCLRB	PB 端口清 0 位控制器	7EF72BH	PB7CLRB	PB6CLRB	PB5CLRB	PB4CLRB	PB3CLRB	PB2CLRB	PB1CLRB	PB0CLRB	0000,0000
S5CON	串口 5 控制寄存器	7EF740H	S5SM0/FE	S5SM1	S5SM2	S5REN	S5TB8	S5RB8	S5TI	S5RI	0000,0000
S5BUF	串口 5 数据寄存器	7EF741H	S5BUF[7:0]								0000,0000
S5INTR	串口 5 中断寄存器	7EF742H	ES5	-	PS5[1:0]		-	-	-	-	0x00,xxxx
UR5TOTE	串口 5 接收超时长度寄存器	7EF743H	UR5TM[23:16]								0000,0000
UR5TOCR	串口 5 接收超时控制寄存器	7EF744H	UR5ENTO	UR5ENTOI	UR5SCALE	-	-	-	-	-	000x,xxxx
UR5TOSR	串口 5 接收超时状态寄存器	7EF745H	-	-	-	-	-	-	-	UR5TOIF	xxxx,xxx0
UR5TOTH	串口 5 接收超时长度寄存器	7EF746H	UR5TM[15:8]								0000,0000
UR5TOTL	串口 5 接收超时长度寄存器	7EF747H	UR5TM[7:0]								0000,0000
USART5CR1	串口 5 控制寄存器 1	7EF748H	S5LINEN	S5DORD	S5CLKEN	S5SPMOD	S5SPIEN	S5SPSLV	S5CPOL	S5CPHA	0000,0000
USART5CR2	串口 5 控制寄存器 2	7EF749H	S5IREN	S5IRLP	S5SCEN	S5NACK	S5HDSEL	S5PCEN	S5PS	S5PE	0000,0000
USART5CR3	串口 5 控制寄存器 3	7EF74AH	S5IrDA_LPBAUD[7:0]								0000,0000
USART5CR4	串口 5 控制寄存器 4	7EF74BH	-	-	-	-	S5SCCKS[1:0]		S5SPICKS[1:0]		xxxx,0000
USART5CR5	串口 5 控制寄存器 5	7EF74CH	S5BRKDET	S5HDRER	S5SLVEN	S5ASYNC	S5TXCF	S5SENDBK	S5HDRDET	S5SYNCAN	0000,0000
USART5GTR	串口 5 保护时间寄存器	7EF74DH	USART5GTR[7:0]								0000,0000
USART5BR1	串口 5 波特率寄存器	7EF74EH	USART5BR[15:8]								0000,0000
USART5BR2	串口 5 波特率寄存器	7EF74FH	USART5BR[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR5T_CFG	UR5T_DMA 配置寄存器	7EF750H	UR5TIE	-	-	-	UR5TIP[1:0]		UR5TPTY[1:0]		0xxx,0000
DMA_UR5T_CR	UR5T_DMA 控制寄存器	7EF751H	ENUR5T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR5T_STA	UR5T_DMA 状态寄存器	7EF752H	-	-	-	-	-	TXOVW	-	UR5TIF	xxxx,x0x0
DMA_UR5T_AMTH	UR5T_DMA 传输总字节数	7EF754H	UR5T_AMT[15:8]								0000,0000
DMA_UR5T_AMT	UR5T_DMA 传输总字节数	7EF755H	UR5T_AMT[7:0]								0000,0000
DMA_UR5T_DONEH	UR5T_DMA 传输完成字节数	7EF756H	UR5T_DONE[15:8]								0000,0000
DMA_UR5T_DONE	UR5T_DMA 传输完成字节数	7EF757H	UR5T_DONE[7:0]								0000,0000
DMA_UR5T_TXAH	UR5T_DMA 发送高地址	7EF758H	UR5T_TXA[15:8]								0000,0000
DMA_UR5T_TXAL	UR5T_DMA 发送低地址	7EF759H	UR5T_TXA[7:0]								0000,0000
DMA_UR5_ITVH	UR5_DMA 时间间隔寄存器	7EF75EH	UR5_ITV[15:8]								0000,0000
DMA_UR5_ITVL	UR5_DMA 时间间隔寄存器	7EF75FH	UR5_ITV[7:0]								0000,0000
DMA_UR5R_CFG	UR5R_DMA 配置寄存器	7EF760H	UR5RIE	-	-	-	UR5RIP[1:0]		UR5RPTY[1:0]		0xxx,0000
DMA_UR5R_CR	UR5R_DMA 控制寄存器	7EF761H	ENUR5R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR5R_STA	UR5R_DMA 状态寄存器	7EF762H	-	-	-	-	-	-	RXLOSS	UR5RIF	xxxx,xx00
DMA_UR5R_AMTH	UR5R_DMA 传输总字节数	7EF764H	UR5R_AMT[15:8]								0000,0000
DMA_UR5R_AMT	UR5R_DMA 传输总字节数	7EF765H	UR5R_AMT[7:0]								0000,0000
DMA_UR5R_DONEH	UR5R_DMA 传输完成字节数	7EF766H	UR5R_DONE[15:8]								0000,0000
DMA_UR5R_DONE	UR5R_DMA 传输完成字节数	7EF767H	UR5R_DONE[7:0]								0000,0000
DMA_UR5R_RXAH	UR5R_DMA 接收高地址	7EF768H	UR5R_RXA[15:8]								0000,0000
DMA_UR5R_RXAL	UR5R_DMA 接收低地址	7EF769H	UR5R_RXA[7:0]								0000,0000
S5CFG	串口 5 配置寄存器	7EF76DH	S5MOD	S5MOD0	S5M0x6	-	-	-	-	S5BRT	000x,xxx0
S5ADDR	串口 5 从机地址寄存器	7EF76EH	S5ADDR[7:0]								0000,0000
S5ADEN	串口 5 从机地址屏蔽寄存器	7EF76FH	S5ADEN[7:0]								0000,0000
S6CON	串口 6 控制寄存器	7EF770H	S6SM0/FE	S6SM1	S6SM2	S6REN	S6TB8	S6RB8	S6TI	S6RI	0000,0000
S6BUF	串口 6 数据寄存器	7EF771H	S6BUF[7:0]								0000,0000
S6INTR	串口 6 中断寄存器	7EF772H	ES6	-	PS6[1:0]		-	-	-	-	0x00,xxxx
UR6TOTE	串口 6 接收超时长度寄存器	7EF773H	UR6TM[23:16]								0000,0000
UR6TOCR	串口 6 接收超时控制寄存器	7EF774H	UR6ENTO	UR6ENTOI	UR6SCALE	-	-	-	-	-	000x,xxxx
UR6TOSR	串口 6 接收超时状态寄存器	7EF775H	-	-	-	-	-	-	-	UR6TOIF	xxxx,xxx0
UR6TOTH	串口 6 接收超时长度寄存器	7EF776H	UR6TM[15:8]								0000,0000
UR6TOTL	串口 6 接收超时长度寄存器	7EF777H	UR6TM[7:0]								0000,0000
USART6CR1	串口 6 控制寄存器 1	7EF778H	S6LINEN	S6DORD	S6CLKEN	S6SPMOD	S6SPIEN	S6SPSLV	S6CPOL	S6CPHA	0000,0000
USART6CR2	串口 6 控制寄存器 2	7EF779H	S6IREN	S6IRLP	S6SCEN	S6NACK	S6HDSEL	S6PCEN	S6PS	S6PE	0000,0000
USART6CR3	串口 6 控制寄存器 3	7EF77AH	S6IrDA_LPBAUD[7:0]								0000,0000
USART6CR4	串口 6 控制寄存器 4	7EF77BH	-	-	-	-	S6SCCKS[1:0]		S6SPICKS[1:0]		xxxx,0000
USART6CR6	串口 6 控制寄存器 6	7EF77CH	S6BRKDET	S6HDRER	S6SLVEN	S6ASYNC	S6TXCF	S6SENDBK	S6HDRDET	S6SYNCAN	0000,0000
USART6GTR	串口 6 保护时间寄存器	7EF77DH	USART6GTR[7:0]								0000,0000
USART6BR1	串口 6 波特率寄存器	7EF77EH	USART6BR[15:8]								0000,0000
USART6BR2	串口 6 波特率寄存器	7EF77FH	USART6BR[7:0]								0000,0000
DMA_UR6T_CFG	UR6T_DMA 配置寄存器	7EF780H	UR6TIE	-	-	-	UR6TIP[1:0]		UR6TPTY[1:0]		0xxx,0000
DMA_UR6T_CR	UR6T_DMA 控制寄存器	7EF781H	ENUR6T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR6T_STA	UR6T_DMA 状态寄存器	7EF782H	-	-	-	-	-	TXOVW	-	UR6TIF	xxxx,x0x0

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR6T_AMTH	UR6T_DMA 传输总字节数	7EF784H	UR6T_AMT[15:8]								0000,0000
DMA_UR6T_AMT	UR6T_DMA 传输总字节数	7EF785H	UR6T_AMT[7:0]								0000,0000
DMA_UR6T_DONEH	UR6T_DMA 传输完成字节数	7EF786H	UR6T_DONE[15:8]								0000,0000
DMA_UR6T_DONE	UR6T_DMA 传输完成字节数	7EF787H	UR6T_DONE[7:0]								0000,0000
DMA_UR6T_TXAH	UR6T_DMA 发送高地址	7EF788H	UR6T_TXA[15:8]								0000,0000
DMA_UR6T_TXAL	UR6T_DMA 发送低地址	7EF789H	UR6T_TXA[7:0]								0000,0000
DMA_UR6_ITVH	UR6_DMA 时间间隔寄存器	7EF78EH	UR6_ITV[15:8]								0000,0000
DMA_UR6_ITVL	UR6_DMA 时间间隔寄存器	7EF78FH	UR6_ITV[7:0]								0000,0000
DMA_UR6R_CFG	UR6R_DMA 配置寄存器	7EF790H	UR6RIE	-	-	-	UR6RIP[1:0]		UR6RPTY[1:0]		0xxx,0000
DMA_UR6R_CR	UR6R_DMA 控制寄存器	7EF791H	ENUR6R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR6R_STA	UR6R_DMA 状态寄存器	7EF792H	-	-	-	-	-	-	RXLOSS	UR6RIF	xxxx,xx00
DMA_UR6R_AMTH	UR6R_DMA 传输总字节数	7EF794H	UR6R_AMT[15:8]								0000,0000
DMA_UR6R_AMT	UR6R_DMA 传输总字节数	7EF795H	UR6R_AMT[7:0]								0000,0000
DMA_UR6R_DONEH	UR6R_DMA 传输完成字节数	7EF796H	UR6R_DONE[15:8]								0000,0000
DMA_UR6R_DONE	UR6R_DMA 传输完成字节数	7EF797H	UR6R_DONE[7:0]								0000,0000
DMA_UR6R_RXAH	UR6R_DMA 接收高地址	7EF798H	UR6R_RXA[15:8]								0000,0000
DMA_UR6R_RXAL	UR6R_DMA 接收低地址	7EF799H	UR6R_RXA[7:0]								0000,0000
S6CFG	串口 6 配置寄存器	7EF79DH	S6MOD	S6MOD0	S6M0x6	-	-	-	-	S6BRT	000x,xxx0
S6ADDR	串口 6 从机地址寄存器	7EF79EH	S6ADDR[7:0]								0000,0000
S6ADEN	串口 6 从机地址屏蔽寄存器	7EF79FH	S6ADEN[7:0]								0000,0000
S7CON	串口 7 控制寄存器	7EF7A0H	S7SM0/FE	S7SM1	S7SM2	S7REN	S7TB8	S7RB8	S7TI	S7RI	0000,0000
S7BUF	串口 7 数据寄存器	7EF7A1H	S7BUF[7:0]								0000,0000
S7INTR	串口 7 中断寄存器	7EF7A2H	ES7	-	PS7[1:0]		-	-	-	-	0x00,xxxx
UR7TOTE	串口 7 接收超时长度寄存器	7EF7A3H	UR7TM[23:16]								0000,0000
UR7TOCR	串口 7 接收超时控制寄存器	7EF7A4H	UR7ENTO	UR7ENTOI	UR7SCALE	-	-	-	-	-	000x,xxxx
UR7TOSR	串口 7 接收超时状态寄存器	7EF7A5H	-	-	-	-	-	-	-	UR7TOIF	xxxx,xxx0
UR7TOTH	串口 7 接收超时长度寄存器	7EF7A6H	UR7TM[15:8]								0000,0000
UR7TOTL	串口 7 接收超时长度寄存器	7EF7A7H	UR7TM[7:0]								0000,0000
USART7CR1	串口 7 控制寄存器 1	7EF7A8H	S7LINEN	S7DORD	S7CLKEN	S7SPMOD	S7SPIEN	S7SPSLV	S7CPOL	S7CPHA	0000,0000
USART7CR2	串口 7 控制寄存器 2	7EF7A9H	S7IREN	S7IRLP	S7SCEN	S7NACK	S7HDSEL	S7PCEN	S7PS	S7PE	0000,0000
USART7CR3	串口 7 控制寄存器 3	7EF7AAH	S7IrDA_LPBAUD[7:0]								0000,0000
USART7CR4	串口 7 控制寄存器 4	7EF7ABH	-	-	-	-	S7SCCKS[1:0]		S7SPICKS[1:0]		xxxx,0000
USART7CR6	串口 7 控制寄存器 6	7EF7ACH	S7BRKDET	S7HDRER	S7SLVEN	S7ASYNC	S7TXCF	S7SENDBK	S7HDRDET	S7SYNCAN	0000,0000
USART7GTR	串口 7 保护时间寄存器	7EF7ADH	USART7GTR[7:0]								0000,0000
USART7BR1	串口 7 波特率寄存器	7EF7AEH	USART7BR[15:8]								0000,0000
USART7BR2	串口 7 波特率寄存器	7EF7AFH	USART7BR[7:0]								0000,0000
DMA_UR7T_CFG	UR7T_DMA 配置寄存器	7EF7B0H	UR7TIE	-	-	-	UR7TIP[1:0]		UR7TPTY[1:0]		0xxx,0000
DMA_UR7T_CR	UR7T_DMA 控制寄存器	7EF7B1H	ENUR7T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR7T_STA	UR7T_DMA 状态寄存器	7EF7B2H	-	-	-	-	-	TXOVW	-	UR7TIF	xxxx,x0x0
DMA_UR7T_AMTH	UR7T_DMA 传输总字节数	7EF7B4H	UR7T_AMT[15:8]								0000,0000
DMA_UR7T_AMT	UR7T_DMA 传输总字节数	7EF7B5H	UR7T_AMT[7:0]								0000,0000
DMA_UR7T_DONEH	UR7T_DMA 传输完成字节数	7EF7B6H	UR7T_DONE[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR7T_DONE	UR7T_DMA 传输完成字节数	7EF7B7H	UR7T_DONE[7:0]								0000,0000
DMA_UR7T_TXAH	UR7T_DMA 发送高地址	7EF7B8H	UR7T_TXA[15:8]								0000,0000
DMA_UR7T_TXAL	UR7T_DMA 发送低地址	7EF7B9H	UR7T_TXA[7:0]								0000,0000
DMA_UR7_ITVH	UR7_DMA 时间间隔寄存器	7EF7BEH	UR7_ITV[15:8]								0000,0000
DMA_UR7_ITVL	UR7_DMA 时间间隔寄存器	7EF7BFH	UR7_ITV[7:0]								0000,0000
DMA_UR7R_CFG	UR7R_DMA 配置寄存器	7EF7C0H	UR7RIE	-	-	-	UR7RIP[1:0]		UR7RPTY[1:0]		0xxx,0000
DMA_UR7R_CR	UR7R_DMA 控制寄存器	7EF7C1H	ENUR7R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR7R_STA	UR7R_DMA 状态寄存器	7EF7C2H	-	-	-	-	-	-	RXLOSS	UR7RIF	xxxx,xx00
DMA_UR7R_AMTH	UR7R_DMA 传输总字节数	7EF7C4H	UR7R_AMT[15:8]								0000,0000
DMA_UR7R_AMT	UR7R_DMA 传输总字节数	7EF7C5H	UR7R_AMT[7:0]								0000,0000
DMA_UR7R_DONEH	UR7R_DMA 传输完成字节数	7EF7C6H	UR7R_DONE[15:8]								0000,0000
DMA_UR7R_DONE	UR7R_DMA 传输完成字节数	7EF7C7H	UR7R_DONE[7:0]								0000,0000
DMA_UR7R_RXAH	UR7R_DMA 接收高地址	7EF7C8H	UR7R_RXA[15:8]								0000,0000
DMA_UR7R_RXAL	UR7R_DMA 接收低地址	7EF7C9H	UR7R_RXA[7:0]								0000,0000
S7CFG	串口 7 配置寄存器	7EF7CDH	S7MOD	S7MOD0	S7M0x6	-	-	-	-	S7BRT	000x,xxx0
S7ADDR	串口 7 从机地址寄存器	7EF7CEH	S7ADDR[7:0]								0000,0000
S7ADEN	串口 7 从机地址屏蔽寄存器	7EF7CFH	S7ADEN[7:0]								0000,0000
S8CON	串口 8 控制寄存器	7EF7D0H	S8SM0/FE	S8SM1	S8SM2	S8REN	S8TB8	S8RB8	S8TI	S8RI	0000,0000
S8BUF	串口 8 数据寄存器	7EF7D1H	S8BUF[7:0]								0000,0000
S8INTR	串口 8 中断寄存器	7EF7D2H	ES8	-	PS8[1:0]		-	-	-	-	0x00,xxxx
UR8TOTE	串口 8 接收超时长度寄存器	7EF7D3H	UR8TM[23:16]								0000,0000
UR8TOCR	串口 8 接收超时控制寄存器	7EF7D4H	UR8ENTO	UR8ENTOI	UR8SCALE	-	-	-	-	-	000x,xxxx
UR8TOSR	串口 8 接收超时状态寄存器	7EF7D5H	-	-	-	-	-	-	-	UR8TOIF	xxxx,xxx0
UR8TOTH	串口 8 接收超时长度寄存器	7EF7D6H	UR8TM[15:8]								0000,0000
UR8TOTL	串口 8 接收超时长度寄存器	7EF7D7H	UR8TM[7:0]								0000,0000
USART8CR1	串口 8 控制寄存器 1	7EF7D8H	S8LINEN	S8DORD	S8CLKEN	S8SPMOD	S8SPIEN	S8SPSLV	S8CPOL	S8CPHA	0000,0000
USART8CR2	串口 8 控制寄存器 2	7EF7D9H	S8IREN	S8IRLP	S8SCEN	S8NACK	S8HDSEL	S8PCEN	S8PS	S8PE	0000,0000
USART8CR3	串口 8 控制寄存器 3	7EF7DAH	S8IrDA_LPBAUD[7:0]								0000,0000
USART8CR4	串口 8 控制寄存器 4	7EF7DBH	-	-	-	-	S8SCCKS[1:0]		S8SPICKS[1:0]		xxxx,0000
USART8CR6	串口 8 控制寄存器 6	7EF7DCH	S8BRKDET	S8HDRER	S8SLVEN	S8ASYNC	S8TXCF	S8SENDBK	S8HDRDET	S8SYNCAN	0000,0000
USART8GTR	串口 8 保护时间寄存器	7EF7DDH	USART8GTR[7:0]								0000,0000
USART8BR1	串口 8 波特率寄存器	7EF7DEH	USART8BR[15:8]								0000,0000
USART8BR2	串口 8 波特率寄存器	7EF7DFH	USART8BR[7:0]								0000,0000
DMA_UR8T_CFG	UR8T_DMA 配置寄存器	7EF7E0H	UR8TIE	-	-	-	UR8TIP[1:0]		UR8TPTY[1:0]		0xxx,0000
DMA_UR8T_CR	UR8T_DMA 控制寄存器	7EF7E1H	ENUR8T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR8T_STA	UR8T_DMA 状态寄存器	7EF7E2H	-	-	-	-	-	TXOVW	-	UR8TIF	xxxx,x0x0
DMA_UR8T_AMTH	UR8T_DMA 传输总字节数	7EF7E4H	UR8T_AMT[15:8]								0000,0000
DMA_UR8T_AMT	UR8T_DMA 传输总字节数	7EF7E5H	UR8T_AMT[7:0]								0000,0000
DMA_UR8T_DONEH	UR8T_DMA 传输完成字节数	7EF7E6H	UR8T_DONE[15:8]								0000,0000
DMA_UR8T_DONE	UR8T_DMA 传输完成字节数	7EF7E7H	UR8T_DONE[7:0]								0000,0000
DMA_UR8T_TXAH	UR8T_DMA 发送高地址	7EF7E8H	UR8T_TXA[15:8]								0000,0000
DMA_UR8T_TXAL	UR8T_DMA 发送低地址	7EF7E9H	UR8T_TXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR8_ITVH	UR8_DMA 时间间隔寄存器	7EF7EEH	UR8_ITV[15:8]								0000,0000
DMA_UR8_ITVL	UR8_DMA 时间间隔寄存器	7EF7EFH	UR8_ITV[7:0]								0000,0000
DMA_UR8R_CFG	UR8R_DMA 配置寄存器	7EF7F0H	UR8RIE	-	-	-	UR8RIP[1:0]		UR8RPTY[1:0]		0xxx,0000
DMA_UR8R_CR	UR8R_DMA 控制寄存器	7EF7F1H	ENUR8R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR8R_STA	UR8R_DMA 状态寄存器	7EF7F2H	-	-	-	-	-	-	RXLOSS	UR8RIF	xxxx,xx00
DMA_UR8R_AMTH	UR8R_DMA 传输总字节数	7EF7F4H	UR8R_AMT[15:8]								0000,0000
DMA_UR8R_AMT	UR8R_DMA 传输总字节数	7EF7F5H	UR8R_AMT[7:0]								0000,0000
DMA_UR8R_DONEH	UR8R_DMA 传输完成字节数	7EF7F6H	UR8R_DONE[15:8]								0000,0000
DMA_UR8R_DONE	UR8R_DMA 传输完成字节数	7EF7F7H	UR8R_DONE[7:0]								0000,0000
DMA_UR8R_RXAH	UR8R_DMA 接收高地址	7EF7F8H	UR8R_RXA[15:8]								0000,0000
DMA_UR8R_RXAL	UR8R_DMA 接收低地址	7EF7F9H	UR8R_RXA[7:0]								0000,0000
S8CFG	串口 8 配置寄存器	7EF7FDH	S8MOD	S8MOD0	S8M0x6	-	-	-	-	S8BRT	000x,xxx0
S8ADDR	串口 8 从机地址寄存器	7EF7FEH	S8ADDR[7:0]								0000,0000
S8ADEN	串口 8 从机地址屏蔽寄存器	7EF7FFH	S8ADEN[7:0]								0000,0000

13.11 扩展特殊功能寄存器列表（XFR：0x7EF600-0x7EF6FF）

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PGA1_CR1	运放 1 控制寄存器 1	7EF600H	PGA1_MSEL[1:0]		PGA1_OSEL[1:0]		PGA1_NSEL[1:0]		PGA1_PSEL[1:0]		0000,0000
PGA1_CR2	运放 1 控制寄存器 2	7EF601H	-	-	PGA1_GSEL[2:0]			PGA1_OE	PGA1_IBS	PGA1_PD	xx00,0001
PGA2_CR1	运放 2 控制寄存器 1	7EF604H	PGA2_MSEL[1:0]		PGA2_OSEL[1:0]		PGA2_NSEL[1:0]		PGA2_PSEL[1:0]		0000,0000
PGA2_CR2	运放 2 控制寄存器 2	7EF605H	-	-	PGA2_GSEL[2:0]			PGA2_OE	PGA2_IBS	PGA2_PD	xx00,0001
PGA3_CR1	运放 3 控制寄存器 1	7EF608H	PGA3_MSEL[1:0]		PGA3_OSEL[1:0]		PGA3_NSEL[1:0]		PGA3_PSEL[1:0]		0000,0000
PGA3_CR2	运放 3 控制寄存器 2	7EF609H	-	-	PGA3_GSEL[2:0]			PGA3_OE	PGA3_IBS	PGA3_PD	xx00,0001
PGA4_CR1	运放 4 控制寄存器 1	7EF60CH	PGA4_MSEL[1:0]		PGA4_OSEL[1:0]		PGA4_NSEL[1:0]		PGA4_PSEL[1:0]		0000,0000
PGA4_CR2	运放 4 控制寄存器 2	7EF60DH	-	-	PGA4_GSEL[2:0]			PGA4_OE	PGA4_IBS	PGA4_PD	xx00,0001
DAC1_CR	DAC1 控制寄存器	7EF610H	DMAEN	DAC1TG	DAC1TS[1:0]		DAC1IP[1:0]		EDAC1I	DAC1EN	0000,0000
DAC1_STA	DAC1 状态寄存器	7EF611H	BUSY	-	-	-	-	-	-	DAC1IF	0xxx,xxx0
DAC1_DATH	DAC1 数据寄存器	7EF612H	-	-	-	-	DAC1DAT[11:8]				xxxx,0000
DAC1_DATL	DAC1 数据寄存器	7EF613H	DAC1_DAT[7:0]								0000,0000
DAC1_DIVH	DAC1 时钟分频寄存器	7EF614H	-	-	-	-	-	-	DAC1_DIV[9:8]		xxxx,xx00
DAC1_DIVL	DAC1 时钟分频寄存器	7EF615H	DAC1_DIV[7:0]								0000,0000
DMA_DAC1_CFG	DAC1_DMA 配置寄存器	7EF618H	DAC1IE	-	-	-	DAC1IP[1:0]		DAC1PTY[1:0]		0xxx,0000
DMA_DAC1_CR	DAC1_DMA 控制寄存器	7EF619H	ENDAC1	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_DAC1_STA	DAC1_DMA 状态寄存器	7EF61AH	-	-	-	-	-	TXOVW	-	DAC1IF	xxxx,x0x0
DMA_DAC1_AMTH	DAC1_DMA 传输总字节数	7EF61CH	DAC1_AMT[15:8]								0000,0000
DMA_DAC1_AMT	DAC1_DMA 传输总字节数	7EF61DH	DAC1_AMT[7:0]								0000,0000
DMA_DAC1_DONEH	DAC1_DMA 传输完成字节数	7EF61EH	DAC1_DONE[15:8]								0000,0000
DMA_DAC1_DONE	DAC1_DMA 传输完成字节数	7EF61FH	DAC1_DONE[7:0]								0000,0000
DMA_DAC1_TXAH	DAC1_DMA 发送高地址	7EF620H	DAC1_TXA[15:8]								0000,0000
DMA_DAC1_TXAL	DAC1_DMA 发送低地址	7EF621H	DAC1_TXA[7:0]								0000,0000
DMA_DAC1_ITVH	DAC1_DMA 时间间隔寄存器	7EF626H	DAC1_ITV[15:8]								0000,0000
DMA_DAC1_ITVL	DAC1_DMA 时间间隔寄存器	7EF627H	DAC1_ITV[7:0]								0000,0000
DAC2_CR	DAC2 控制寄存器	7EF628H	DMAEN	DAC2TG	DAC2TS[1:0]		DAC2IP[1:0]		EDAC2I	DAC2EN	0000,0000
DAC2_STA	DAC2 状态寄存器	7EF629H	BUSY	-	-	-	-	-	-	DAC2IF	0xxx,xxx0
DAC2_DATH	DAC2 数据寄存器	7EF62AH	-	-	-	-	DAC2DAT[11:8]				xxxx,0000
DAC2_DATL	DAC2 数据寄存器	7EF62BH	DAC2_DAT[7:0]								0000,0000
DAC2_DIVH	DAC2 时钟分频寄存器	7EF62CH	-	-	-	-	-	-	DAC2_DIV[9:8]		xxxx,xx00
DAC2_DIVL	DAC2 时钟分频寄存器	7EF62DH	DAC2_DIV[7:0]								0000,0000
DMA_DAC2_CFG	DAC2_DMA 配置寄存器	7EF630H	DAC2IE	-	-	-	DAC2IP[1:0]		DAC2PTY[1:0]		0xxx,0000
DMA_DAC2_CR	DAC2_DMA 控制寄存器	7EF631H	ENDAC2	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_DAC2_STA	DAC2_DMA 状态寄存器	7EF632H	-	-	-	-	-	TXOVW	-	DAC2IF	xxxx,x0x0
DMA_DAC2_AMTH	DAC2_DMA 传输总字节数	7EF634H	DAC2_AMT[15:8]								0000,0000
DMA_DAC2_AMT	DAC2_DMA 传输总字节数	7EF635H	DAC2_AMT[7:0]								0000,0000
DMA_DAC2_DONEH	DAC2_DMA 传输完成字节数	7EF636H	DAC2_DONE[15:8]								0000,0000
DMA_DAC2_DONE	DAC2_DMA 传输完成字节数	7EF637H	DAC2_DONE[7:0]								0000,0000
DMA_DAC2_TXAH	DAC2_DMA 发送高地址	7EF638H	DAC2_TXA[15:8]								0000,0000
DMA_DAC2_TXAL	DAC2_DMA 发送低地址	7EF639H	DAC2_TXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_DAC2_ITVH	DAC2_DMA 时间间隔寄存器	7EF63EH	DAC2_ITV[15:8]								0000,0000
DMA_DAC2_ITVL	DAC2_DMA 时间间隔寄存器	7EF63FH	DAC2_ITV[7:0]								0000,0000
DMA_PWMET_CFG	PWMET_DMA 配置寄存器	7EF668H	PWMETIE	-	-	-	PWMETIP[1:0]		PWMETPTY[1:0]		0xxx,0000
DMA_PWMET_CR	PWMET_DMA 控制寄存器	7EF669H	ENPWMET	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_PWMET_STA	PWMET_DMA 状态寄存器	7EF66AH	RDOV	-	-	-	-	TXOVW	-	PWMETIF	0xxx,x0x0
DMA_PWMET_AMTH	PWMET_DMA 传输总字节数	7EF66CH	PWMET_AMT[15:8]								0000,0000
DMA_PWMET_AMT	PWMET_DMA 传输总字节数	7EF66DH	PWMET_AMT[7:0]								0000,0000
DMA_PWMET_DONEH	PWMET_DMA 传输完成字节数	7EF66EH	PWMET_DONE[15:8]								0000,0000
DMA_PWMET_DONE	PWMET_DMA 传输完成字节数	7EF66FH	PWMET_DONE[7:0]								0000,0000
DMA_PWMET_TXAH	PWMET_DMA 发送高地址	7EF670H	PWMET_TXA[15:8]								0000,0000
DMA_PWMET_TXAL	PWMET_DMA 发送低地址	7EF671H	PWMET_TXA[7:0]								0000,0000
DMA_PWME_ITVH	PWME_DMA 时间间隔寄存器	7EF676H	PWME_ITV[15:8]								0000,0000
DMA_PWME_ITVL	PWME_DMA 时间间隔寄存器	7EF677H	PWME_ITV[7:0]								0000,0000
DMA_PWMER_CFG	PWMER_DMA 配置寄存器	7EF678H	PWMERIE	-	-	-	PWMERIP[1:0]		PWMERPTY[1:0]		0xxx,0000
DMA_PWMER_CR	PWMER_DMA 控制寄存器	7EF679H	ENPWMER	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_PWMER_STA	PWMER_DMA 状态寄存器	7EF67AH	WROV	-	-	-	-	-	RXLOSS	PWMERIF	0xxx,xx00
DMA_PWMER_AMTH	PWMER_DMA 传输总字节数	7EF67CH	PWMER_AMT[15:8]								0000,0000
DMA_PWMER_AMT	PWMER_DMA 传输总字节数	7EF67DH	PWMER_AMT[7:0]								0000,0000
DMA_PWMER_DONEH	PWMER_DMA 传输完成字节数	7EF67EH	PWMER_DONE[15:8]								0000,0000
DMA_PWMER_DONE	PWMER_DMA 传输完成字节数	7EF67FH	PWMER_DONE[7:0]								0000,0000
DMA_PWMER_RXAH	PWMER_DMA 接收高地址	7EF680H	PWMER_RXA[15:8]								0000,0000
DMA_PWMER_RXAL	PWMER_DMA 接收低地址	7EF681H	PWMER_RXA[7:0]								0000,0000
HSPWME_CFG	高速 PWME 配置寄存器	7EF688H	-	-	-	-	-	PWME_INTEN	PWME_ASYNCEN	1	xxxx,x001
HSPWME_ADR	高速 PWME 地址寄存器	7EF689H	PWME_RW	PWME_ADDR[6:0]							0000,0000
HSPWME_DAT	高速 PWME 数据寄存器	7EF68AH	PWME_DATA[7:0]								0000,0000
PWME_CLKDIV	PWME 时钟预分配寄存器	7EF68BH	PWME_CLKDIV[7:0]								0000,0000
HSPWMF_CFG	高速 PWMF 配置寄存器	7EF68CH	-	-	-	-	-	PWMF_INTEN	PWMF_ASYNCEN	1	xxxx,x001
HSPWMF_ADR	高速 PWMF 地址寄存器	7EF68DH	PWMF_RW	PWMF_ADDR[6:0]							0000,0000
HSPWMF_DAT	高速 PWMF 数据寄存器	7EF68EH	PWMF_DATA[7:0]								0000,0000
PWMF_CLKDIV	PWMF 时钟预分配寄存器	7EF68FH	PWMF_CLKDIV[7:0]								0000,0000
PWME_ENO2	PWME 输出使能寄存器 2	7EF690H	-	-	-	-	-	ENOE6	-	ENOE5	xxxx,x0x0
PWME_IOAUX2	PWME 输出附加寄存器 2	7EF691H	-	-	-	-	-	AUXE6	-	AUXE6	xxxx,x0x0
PWME_CR3	PWME 控制寄存器 3	7EF692H	MMSE2[3:0]				-	OISE6	-	OISE5	0000,x0x0
PWME_SR3	PWME 状态寄存器 3	7EF693H	-	-	-	-	-	-	CCE6IF	CCE5IF	xxxx,xx00
PWME_CCER3	PWME 捕获比较使能寄存器 3	7EF694H	-	-	CCE6P	CCE6E	-	-	CCE5P	CCE5E	xx00,xx00
PWME_CCMR1X	PWME 捕获比较模式寄存器 1x	7EF698H	-	-	-	-	-	-	-	OCE1M[3]	xxx,xxx0
PWME_CCMR2X	PWME 捕获比较模式寄存器 2x	7EF699H	-	-	-	-	-	-	-	OCE2M[3]	xxx,xxx0
PWME_CCMR3X	PWME 捕获比较模式寄存器 3x	7EF69AH	-	-	-	-	-	-	-	OCE3M[3]	xxx,xxx0
PWME_CCMR4X	PWME 捕获比较模式寄存器 4x	7EF69BH	-	-	-	-	-	-	-	OCE4M[3]	0000,0000
PWME_CCMR5	PWME 捕获比较模式寄存器 5	7EF69CH	OCE5CE	OCE5M[2:0]			OCE5PE	-	-	-	0000,0xxx
PWME_CCMR5X	PWME 捕获比较模式寄存器 5x	7EF69DH	-	-	-	-	-	-	-	OCE5M[3]	xxxx,xxx0
PWME_CCMR6	PWME 捕获比较模式寄存器 6	7EF69EH	OCE6CE	OCE6M[2:0]			OCE6PE	-	-	-	000x,xxxx

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PWME_CCMR6X	PWME 捕获比较模式寄存器 6x	7EF69FH	-	-	-	-	-	-	-	OCE6M[3]	xxx,xxx0
PWME_CCR5H	PWME 捕获比较寄存器 5	7EF6A0H	CCRE5[15:8]								0000,0000
PWME_CCR5L	PWME 捕获比较寄存器 5	7EF6A1H	CCRE5[7:0]								0000,0000
PWME_CCR5X	PWME 捕获比较寄存器 5	7EF6A2H	GCE5C3	GCE5C2	GCE5C1	-	-	-	-	-	000x,xxx
PWME_CCR6H	PWME 捕获比较寄存器 6	7EF6A3H	CCRE6[15:8]								0000,0000
PWME_CCR6L	PWME 捕获比较寄存器 6	7EF6A4H	CCRE6[7:0]								0000,0000
PWME_DER	PWME_DMA 事件使能寄存器	7EF6A8H	-	TDEE	COMDEE	CCE4DE	CCE3DE	CCE2DE	CCE1DE	UDEE	x000,0000
PWME_DBA	PWME_DMA 基址寄存器	7EF6A9H	-	-	-	DBAE[4:0]					xxx0,0000
PWME_DBL	PWME_DMA 传输长度寄存器	7EF6AAH	-	-	-	DBLE[4:0]					xxx0,0000
PWME_DMACR	PWME_DMA 控制寄存器	7EF6ABH	-	-	-	DSKIPE	DDIRE	DMAENE	SIZEE[1:0]		xxx0,0000
PWME_ETRPS	PWME 的 ETR 选择寄存器	7EF6B0H	-	-	CMPEPS[1:0]		BRKEPS[1:0]		ETREPS[1:0]		xx00,0000
PWME_ENO	PWME 输出使能控制	7EF6B1H	ENOE4N	ENOE4P	ENOE3N	ENOE3P	ENOE2N	ENOE2P	ENOE1N	ENOE1P	0000,0000
PWME_PS	PWME 输出脚选择寄存器	7EF6B2H	EC4PS[1:0]		EC3PS[1:0]		EC2PS[1:0]		EC1PS[1:0]		0000,0000
PWME_IOAUX	PWME 辅助寄存器	7EF6B3H	AUXE4N	AUXE4P	AUXE3N	AUXE3P	AUXE2N	AUXE2P	AUXE1N	AUXE1P	0000,0000
PWMF_ETRPS	PWMF 的 ETR 选择寄存器	7EF6B4H			CMPFPS[1:0]		BRKFPS[1:0]		ETRFPS[1:0]		xx00,0000
PWMF_ENO	PWMF 输出使能控制	7EF6B5H	-	ENOF8P	-	ENOF7P	-	ENOF6P	-	ENOF5P	x0x0,x0x0
PWMF_PS	PWMF 输出脚选择寄存器	7EF6B6H	FC8PS[1:0]		FC7PS[1:0]		FC6PS[1:0]		FC5PS[1:0]		0000,0000
PWMF_IOAUX	PWMF 辅助寄存器	7EF6B7H	-	AUXF8P	-	AUXF7P	-	AUXF6P	-	AUXF5P	x0x0,x0x0
PWME_PS2	PWME 输出脚选择寄存器 2	7EF6B8H	-	-	-	-	EC6PS[1:0]		EC5PS[1:0]		xxxx,0000
PWME_RCRH	PWME 重复计数器寄存器	7EF6B9H	REPE[15:8]								0000,0000
PWMF_RCRH	PWMF 重复计数器寄存器	7EF6BAH	REPF[15:8]								0000,0000
PWME_CR1	PWME 控制寄存器 1	7EF6C0H	ARPEE	CMSE[1:0]		DIRE	OPME	URSE	UDISE	CENE	0000,0000
PWME_CR2	PWME 控制寄存器 2	7EF6C1H	-	MMSE[2:0]			-	COMSE	-	CCPCE	x000,x0x0
PWME_SMCR	PWME 从模式控制寄存器	7EF6C2H	MSME	TSE[2:0]			-	SMSE[2:0]			0000,x000
PWME_ETR	PWME 外部触发寄存器	7EF6C3H	ETPE	ECEE	ETPSE[1:0]		ETFE[3:0]				0000,0000
PWME_IER	PWME 中断使能寄存器	7EF6C4H	BIEE	TIEE	COMIEE	CCE4IE	CCE3IE	CCE2IE	CCE1IE	UIEE	0000,0000
PWME_SR1	PWME 状态寄存器 1	7EF6C5H	BIFE	TIFE	COMIFE	CCE4IF	CCE3IF	CEC2IF	CCE1IF	UIFE	0000,0000
PWME_SR2	PWME 状态寄存器 2	7EF6C6H	-	-	-	CCE4OF	CCE3OF	CCE2OF	CCE1OF	-	xxx0,000x
PWME_EGR	PWME 事件发生寄存器	7EF6C7H	BGE	TGE	COMGE	CCE4G	CCE3G	CCE2G	CCE1G	UGE	0000,0000
PWME_CCMR1	PWME 捕获模式寄存器 1	7EF6C8H	OCE1CE	OCE1M[2:0]			OCE1PE	OCE1FE	CCE1S[1:0]		0000,0000
	PWME 比较模式寄存器 1		ICE1F[3:0]				ICE1PSC[1:0]		CCE1S[1:0]		0000,0000
PWME_CCMR2	PWME 捕获模式寄存器 2	7EF6C9H	OCE2CE	OCE2M[2:0]			OCE2PE	OCE2FE	CCE2S[1:0]		0000,0000
	PWME 比较模式寄存器 2		ICE2F[3:0]				ICE2PSC[1:0]		CCE2S[1:0]		0000,0000
PWME_CCMR3	PWME 捕获模式寄存器 3	7EF6CAH	OCE3CE	OCE3M[2:0]			OCE3PE	OCE3FE	CCE3S[1:0]		0000,0000
	PWME 比较模式寄存器 3		ICE3F[3:0]				ICE3PSC[1:0]		CCE3S[1:0]		0000,0000
PWME_CCMR4	PWME 捕获模式寄存器 4	7EF6CBH	OCE4CE	OCE4M[2:0]			OCE4PE	OCE4FE	CCE4S[1:0]		0000,0000
	PWME 比较模式寄存器 4		ICE4F[3:0]				ICE4PSC[1:0]		CCE4S[1:0]		0000,0000
PWME_CCER1	PWME 捕获比较使能寄存器 1	7EF6CCH	CCE2NP	CCE2NE	CCE2P	CCE2E	CCE1NP	CCE1NE	CCE1P	CCE1E	0000,0000
PWME_CCER2	PWME 捕获比较使能寄存器 2	7EF6CDH	CCE4NP	CCE4NE	CCE4P	CCE4E	CCE3NP	CCE3NE	CCE3P	CCE3E	0000,0000
PWME_CNTRH	PWME 计数器高字节	7EF6CEH	CNTE[15:8]								0000,0000
PWME_CNTRL	PWME 计数器低字节	7EF6CFH	CNTE[7:0]								0000,0000
PWME_PSCRH	PWME 预分频高字节	7EF6D0H	PSCE[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PWME_PSCRL	PWME 预分频低字节	7EF6D1H	PSCE[7:0]								0000,0000
PWME_ARRH	PWME 自动重装寄存器高字节	7EF6D2H	ARRE[15:8]								0000,0000
PWME_ARRL	PWME 自动重装寄存器低字节	7EF6D3H	ARRE[7:0]								0000,0000
PWME_RCR	PWME 重复计数器寄存器	7EF6D4H	REPE[7:0]								0000,0000
PWME_CCR1H	PWME 比较捕获寄存器 1 高位	7EF6D5H	CCRE1[15:8]								0000,0000
PWME_CCR1L	PWME 比较捕获寄存器 1 低位	7EF6D6H	CCRE1[7:0]								0000,0000
PWME_CCR2H	PWME 比较捕获寄存器 2 高位	7EF6D7H	CCRE2[15:8]								0000,0000
PWME_CCR2L	PWME 比较捕获寄存器 2 低位	7EF6D8H	CCRE2[7:0]								0000,0000
PWME_CCR3H	PWME 比较捕获寄存器 3 高位	7EF6D9H	CCRE3[15:8]								0000,0000
PWME_CCR3L	PWME 比较捕获寄存器 3 低位	7EF6DAH	CCRE3[7:0]								0000,0000
PWME_CCR4H	PWME 比较捕获寄存器 4 高位	7EF6DBH	CCRE4[15:8]								0000,0000
PWME_CCR4L	PWME 比较捕获寄存器 4 低位	7EF6DCH	CCRE4[7:0]								0000,0000
PWME_BKR	PWME 刹车寄存器	7EF6DDH	MOEE	AOEE	BKPE	BKEE	OSSRE	OSSIE	LOCKE[1:0]		0000,0000
PWME_DTR	PWME 死区控制寄存器	7EF6DEH	DTGE[7:0]								0000,0000
PWME_OISR	PWME 输出空闲状态寄存器	7EF6DFH	OISE4N	OISE4	OISE3N	OISE3	OISE2N	OISE2	OISE1N	OISE1	0000,0000
PWMF_CR1	PWMF 控制寄存器 1	7EF6E0H	ARPEF	CMSF[1:0]		DIRF	OPMF	URSF	UDISF	CENF	0000,0000
PWMF_CR2	PWMF 控制寄存器 2	7EF6E1H	-	MMSF[2:0]			-	COMSF	-	CCPCF	x000,x0x0
PWMF_SMCR	PWMF 从模式控制寄存器	7EF6E2H	MSMF	TSF[2:0]			-	SMSF[2:0]			0000,x000
PWMF_ETR	PWMF 外部触发寄存器	7EF6E3H	ETPF	ECEF	ETPSF[1:0]		ETFF[3:0]				0000,0000
PWMF_IER	PWMF 中断使能寄存器	7EF6E4H	BIEF	TIEF	COMIEF	CCF8IE	CCF7IE	CCF6IE	CCF5IE	UIEF	0000,0000
PWMF_SR1	PWMF 状态寄存器 1	7EF6E5H	BIFF	TIFF	COMIFF	CCF8IF	CCF7IF	CCF6IF	CCF5IF	UIFF	0000,0000
PWMF_SR2	PWMF 状态寄存器 2	7EF6E6H	-	-	-	CCF8OF	CCF7OF	CCF6OF	CCF5OF	-	xxx0,000x
PWMF_EGR	PWMF 事件发生寄存器	7EF6E7H	BGF	TGF	COMGF	CCF8G	CCF7G	CCF6G	CCF5G	UGF	0000,0000
PWMF_CCMR1	PWMF 捕获模式寄存器 1	7EF6E8H	OCF5CE	OCF5M[2:0]			OCF5PE	OCF5FE	CCF5S[1:0]		0000,0000
	PWMF 比较模式寄存器 1		ICF5F[3:0]				ICF5PSC[1:0]		CCF5S[1:0]		0000,0000
PWMF_CCMR2	PWMF 捕获模式寄存器 2	7EF6E9H	OCF6CE	OCF6M[2:0]			OCF6PE	OCF6FE	CCF6S[1:0]		0000,0000
	PWMF 比较模式寄存器 2		ICF6F[3:0]				ICF6PSC[1:0]		CCF6S[1:0]		0000,0000
PWMF_CCMR3	PWMF 捕获模式寄存器 3	7EF6EAH	OCF7CE	OCF7M[2:0]			OCF7PE	OCF7FE	CCF7S[1:0]		0000,0000
	PWMF 比较模式寄存器 3		ICF7F[3:0]				ICF7PSC[1:0]		CCF7S[1:0]		0000,0000
PWMF_CCMR4	PWMF 捕获模式寄存器 4	7EF6EBH	OCF8CE	OCF8M[2:0]			OCF8PE	OCF8FE	CCF8S[1:0]		0000,0000
	PWMF 比较模式寄存器 4		ICF8F[3:0]				ICF8PSC[1:0]		CCF8S[1:0]		0000,0000
PWMF_CCER1	PWMF 捕获比较使能寄存器 1	7EF6ECH	-	-	CCF6P	CCF6E	-	-	CCF5P	CCF5E	xx00,xx00
PWMF_CCER2	PWMF 捕获比较使能寄存器 2	7EF6EDH	-	-	CCF8P	CCF8E	-	-	CCF7P	CCF7E	xx00,xx00
PWMF_CNTRH	PWMF 计数器高字节	7EF6EEH	CNTF[15:8]								0000,0000
PWMF_CNTRL	PWMF 计数器低字节	7EF6EFH	CNTF[7:0]								0000,0000
PWMF_PSCRH	PWMF 预分频高字节	7EF6F0H	PSCF[15:8]								0000,0000
PWMF_PSCRL	PWMF 预分频低字节	7EF6F1H	PSCF[7:0]								0000,0000
PWMF_ARRH	PWMF 自动重装寄存器高字节	7EF6F2H	ARRF[15:8]								0000,0000
PWMF_ARRL	PWMF 自动重装寄存器低字节	7EF6F3H	ARRF[7:0]								0000,0000
PWMF_RCR	PWMF 重复计数器寄存器	7EF6F4H	REPF[7:0]								0000,0000
PWMF_CCR5H	PWMF 比较捕获寄存器 5 高位	7EF6F5H	CCRF5[15:8]								0000,0000
PWMF_CCR5L	PWMF 比较捕获寄存器 5 低位	7EF6F6H	CCRF5[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PWMF_CCR6H	PWMF 比较捕获寄存器 6 高位	7EF6F7H	CCRF6[15:8]								0000,0000
PWMF_CCR6L	PWMF 比较捕获寄存器 6 低位	7EF6F8H	CCRF6[7:0]								0000,0000
PWMF_CCR7H	PWMF 比较捕获寄存器 7 高位	7EF6F9H	CCRF7[15:8]								0000,0000
PWMF_CCR7L	PWMF 比较捕获寄存器 7 低位	7EF6FAH	CCRF7[7:0]								0000,0000
PWMF_CCR8H	PWMF 比较捕获寄存器 8 高位	7EF6FBH	CCRF8[15:8]								0000,0000
PWMF_CCR8L	PWMF 比较捕获寄存器 8 低位	7EF6FCH	CCRF8[7:0]								0000,0000
PWMF_BKR	PWMF 刹车寄存器	7EF6FDH	MOEF	AOEF	BKPF	BKEF	OSSRF	OSSIF	LOCKF[1:0]		0000,0000
PWMF_DTR	PWMF 死区控制寄存器	7EF6FEH	DTGF[7:0]								0000,0000
PWMF_OISR	PWMF 输出空闲状态寄存器	7EF6FFH	-	OISF8	-	OISF7	-	OISF6	-	OISF5	x0x0,x0x0

13.12 扩展特殊功能寄存器列表（XFR：0x7EF500-0x7EF5FF）

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P0OS	P0 端口开源控制寄存器	7EF500H	P07OS	P06OS	P05OS	P04OS	P03OS	P02OS	P01OS	P00OS	0000,0000
P1OS	P1 端口开源控制寄存器	7EF501H	P17OS	P16OS	P15OS	P14OS	P13OS	P12OS	P11OS	P10OS	0000,0000
P2OS	P2 端口开源控制寄存器	7EF502H	P27OS	P26OS	P25OS	P24OS	P23OS	P22OS	P21OS	P20OS	0000,0000
P3OS	P3 端口开源控制寄存器	7EF503H	P37OS	P36OS	P35OS	P34OS	P33OS	P32OS	P31OS	P30OS	0000,0000
P4OS	P4 端口开源控制寄存器	7EF504H	P47OS	P46OS	P45OS	P44OS	P43OS	P42OS	P41OS	P40OS	0000,0000
P5OS	P5 端口开源控制寄存器	7EF505H	P57OS	P56OS	P55OS	P54OS	P53OS	P52OS	P51OS	P50OS	0000,0000
P6OS	P6 端口开源控制寄存器	7EF506H	P67OS	P66OS	P65OS	P64OS	P63OS	P62OS	P61OS	P60OS	0000,0000
P7OS	P7 端口开源控制寄存器	7EF507H	P77OS	P76OS	P75OS	P74OS	P73OS	P72OS	P71OS	P70OS	0000,0000
P8OS	P8 端口开源控制寄存器	7EF508H	P87OS	P86OS	P85OS	P84OS	P83OS	P82OS	P81OS	P80OS	0000,0000
P9OS	P9 端口开源控制寄存器	7EF509H	P97OS	P96OS	P95OS	P94OS	P93OS	P92OS	P91OS	P90OS	0000,0000
PAOS	PA 端口开源控制寄存器	7EF50AH	PA7OS	PA6OS	PA5OS	PA4OS	PA3OS	PA2OS	PA1OS	PA0OS	0000,0000
PBOS	PB 端口开源控制寄存器	7EF50BH	PB7OS	PB6OS	PB5OS	PB4OS	PB3OS	PB2OS	PB1OS	PB0OS	0000,0000
P0OD	P0 端口开漏控制寄存器	7EF510H	P07OD	P06OD	P05OD	P04OD	P03OD	P02OD	P01OD	P00OD	0000,0000
P1OD	P1 端口开漏控制寄存器	7EF511H	P17OD	P16OD	P15OD	P14OD	P13OD	P12OD	P11OD	P10OD	0000,0000
P2OD	P2 端口开漏控制寄存器	7EF512H	P27OD	P26OD	P25OD	P24OD	P23OD	P22OD	P21OD	P20OD	0000,0000
P3OD	P3 端口开漏控制寄存器	7EF513H	P37OD	P36OD	P35OD	P34OD	P33OD	P32OD	P31OD	P30OD	0000,0000
P4OD	P4 端口开漏控制寄存器	7EF514H	P47OD	P46OD	P45OD	P44OD	P43OD	P42OD	P41OD	P40OD	0000,0000
P5OD	P5 端口开漏控制寄存器	7EF515H	P57OD	P56OD	P55OD	P54OD	P53OD	P52OD	P51OD	P50OD	0000,0000
P6OD	P6 端口开漏控制寄存器	7EF516H	P67OD	P66OD	P65OD	P64OD	P63OD	P62OD	P61OD	P60OD	0000,0000
P7OD	P7 端口开漏控制寄存器	7EF517H	P77OD	P76OD	P75OD	P74OD	P73OD	P72OD	P71OD	P70OD	0000,0000
P8OD	P8 端口开漏控制寄存器	7EF518H	P87OD	P86OD	P85OD	P84OD	P83OD	P82OD	P81OD	P80OD	0000,0000
P9OD	P9 端口开漏控制寄存器	7EF519H	P97OD	P96OD	P95OD	P94OD	P93OD	P92OD	P91OD	P90OD	0000,0000
PAOD	PA 端口开漏控制寄存器	7EF51AH	PA7OD	PA6OD	PA5OD	PA4OD	PA3OD	PA2OD	PA1OD	PA0OD	0000,0000
PBOD	PB 端口开漏控制寄存器	7EF51BH	PB7OD	PB6OD	PB5OD	PB4OD	PB3OD	PB2OD	PB1OD	PB0OD	0000,0000
DMA_CAN1T_CFG	CAN1T_DMA 配置寄存器	7EF520H	CAN1TIE	-	TADRALIGN	TX_BW	CAN1TIP[1:0]		CAN1TPTY[1:0]		0xxx,0000
DMA_CAN1T_CR	CAN1T_DMA 控制寄存器	7EF521H	ENCAN1T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_CAN1T_STA	CAN1T_DMA 状态寄存器	7EF522H	-	-	-	-	-	TXOVW	-	CAN1TIF	xxxx,x0x0
DMA_CAN1T_AMTH	CAN1T_DMA 传输总字节数	7EF524H	CAN1T_AMT[15:8]								0000,0000
DMA_CAN1T_AMT	CAN1T_DMA 传输总字节数	7EF525H	CAN1T_AMT[7:0]								0000,0000
DMA_CAN1T_DONEH	CAN1T_DMA 传输完成字节数	7EF526H	CAN1T_DONE[15:8]								0000,0000
DMA_CAN1T_DONE	CAN1T_DMA 传输完成字节数	7EF527H	CAN1T_DONE[7:0]								0000,0000
DMA_CAN1T_TXAH	CAN1T_DMA 发送高地址	7EF528H	CAN1T_TXA[15:8]								0000,0000
DMA_CAN1T_TXAL	CAN1T_DMA 发送低地址	7EF529H	CAN1T_TXA[7:0]								0000,0000
DMA_CAN1T_FRM	CAN1T_DMA 发送帧寄存器	7EF52BH	CAN1T_FRM[7:0]								0000,0000
DMA_CAN1T_FCNTH	CAN1T_DMA 发送帧数寄存器	7EF52CH	CAN1T_FCNT[15:8]								0000,0000
DMA_CAN1T_FCNTL	CAN1T_DMA 发送帧数寄存器	7EF52DH	CAN1T_FCNT[7:0]								0000,0000
DMA_CAN1_ITVH	CAN1_DMA 时间间隔寄存器	7EF52EH	CAN1_ITV[15:8]								0000,0000
DMA_CAN1_ITVL	CAN1_DMA 时间间隔寄存器	7EF52FH	CAN1_ITV[7:0]								0000,0000
DMA_CAN1R_CFG	CAN1R_DMA 配置寄存器	7EF530H	CAN1RIE	RAMTALIGN	RADRALIGN	RX_BW	CAN1RIP[1:0]		CAN1RPTY[1:0]		0xxx,0000
DMA_CAN1R_CR	CAN1R_DMA 控制寄存器	7EF531H	ENCAN1R	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_CAN1R_STA	CAN1R_DMA 状态寄存器	7EF532H	-	-	-	-	-	-	RXLOSS	CAN1RIF	xxxx,xx00
CAN1_AUX_CR	CAN1 辅助控制寄存器	7EF533H	-	STBYEN	TRANMANU	DMATXEN	-	-	-	DMARXEN	x000,xxx0
DMA_CAN1R_AMTH	CAN1R_DMA 传输总字节数	7EF534H	CAN1R_AMT[15:8]								0000,0000
DMA_CAN1R_AMT	CAN1R_DMA 传输总字节数	7EF535H	CAN1R_AMT[7:0]								0000,0000
DMA_CAN1R_DONEH	CAN1R_DMA 传输完成字节数	7EF536H	CAN1R_DONE[15:8]								0000,0000
DMA_CAN1R_DONE	CAN1R_DMA 传输完成字节数	7EF537H	CAN1R_DONE[7:0]								0000,0000
DMA_CAN1R_RXAH	CAN1R_DMA 接收高地址	7EF538H	CAN1R_RXA[15:8]								0000,0000
DMA_CAN1R_RXAL	CAN1R_DMA 接收低地址	7EF539H	CAN1R_RXA[7:0]								0000,0000
DMA_CAN1R_FRM	CAN1T_DMA 接收帧寄存器	7EF53BH	CAN1R_FRM[7:0]								0000,0000
DMA_CAN1R_FCNTH	CAN1T_DMA 接收帧数寄存器	7EF53CH	CAN1R_FCNT[15:8]								0000,0000
DMA_CAN1R_FCNTL	CAN1T_DMA 接收帧数寄存器	7EF53DH	CAN1R_FCNT[7:0]								0000,0000
DMA_CAN2T_CFG	CAN2T_DMA 配置寄存器	7EF540H	CAN2TIE	-	TADRALIGN	TX_BW	CAN2TIP[1:0]		CAN2TPTY[1:0]		0xxx,0000
DMA_CAN2T_CR	CAN2T_DMA 控制寄存器	7EF541H	ENCAN2T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_CAN2T_STA	CAN2T_DMA 状态寄存器	7EF542H	-	-	-	-	-	TXOVW	-	CAN2TIF	xxxx,x0x0
DMA_CAN2T_AMTH	CAN2T_DMA 传输总字节数	7EF544H	CAN2T_AMT[15:8]								0000,0000
DMA_CAN2T_AMT	CAN2T_DMA 传输总字节数	7EF545H	CAN2T_AMT[7:0]								0000,0000
DMA_CAN2T_DONEH	CAN2T_DMA 传输完成字节数	7EF546H	CAN2T_DONE[15:8]								0000,0000
DMA_CAN2T_DONE	CAN2T_DMA 传输完成字节数	7EF547H	CAN2T_DONE[7:0]								0000,0000
DMA_CAN2T_TXAH	CAN2T_DMA 发送高地址	7EF548H	CAN2T_TXA[15:8]								0000,0000
DMA_CAN2T_TXAL	CAN2T_DMA 发送低地址	7EF549H	CAN2T_TXA[7:0]								0000,0000
DMA_CAN2T_FRM	CAN2T_DMA 发送帧寄存器	7EF54BH	CAN2T_FRM[7:0]								0000,0000
DMA_CAN2T_FCNTH	CAN2T_DMA 发送帧数寄存器	7EF54CH	CAN2T_FCNT[15:8]								0000,0000
DMA_CAN2T_FCNTL	CAN2T_DMA 发送帧数寄存器	7EF54DH	CAN2T_FCNT[7:0]								0000,0000
DMA_CAN2_ITVH	CAN2_DMA 时间间隔寄存器	7EF54EH	CAN2_ITV[15:8]								0000,0000
DMA_CAN2_ITVL	CAN2_DMA 时间间隔寄存器	7EF54FH	CAN2_ITV[7:0]								0000,0000
DMA_CAN2R_CFG	CAN2R_DMA 配置寄存器	7EF550H	CAN2RIE	RAMTALIGN	RADRALIGN	RX_BW	CAN2RIP[1:0]		CAN2RPTY[1:0]		0xxx,0000
DMA_CAN2R_CR	CAN2R_DMA 控制寄存器	7EF551H	ENCAN2R	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_CAN2R_STA	CAN2R_DMA 状态寄存器	7EF552H	-	-	-	-	-	-	RXLOSS	CAN2RIF	xxxx,xx00
CAN2_AUX_CR	CAN2 辅助控制寄存器	7EF553H	-	STBYEN	TRANMANU	DMATXEN	-	-	-	DMARXEN	x000,xxx0
DMA_CAN2R_AMTH	CAN2R_DMA 传输总字节数	7EF554H	CAN2R_AMT[15:8]								0000,0000
DMA_CAN2R_AMT	CAN2R_DMA 传输总字节数	7EF555H	CAN2R_AMT[7:0]								0000,0000
DMA_CAN2R_DONEH	CAN2R_DMA 传输完成字节数	7EF556H	CAN2R_DONE[15:8]								0000,0000
DMA_CAN2R_DONE	CAN2R_DMA 传输完成字节数	7EF557H	CAN2R_DONE[7:0]								0000,0000
DMA_CAN2R_RXAH	CAN2R_DMA 接收高地址	7EF558H	CAN2R_RXA[15:8]								0000,0000
DMA_CAN2R_RXAL	CAN2R_DMA 接收低地址	7EF559H	CAN2R_RXA[7:0]								0000,0000
DMA_CAN2R_FRM	CAN2T_DMA 接收帧寄存器	7EF55BH	CAN2R_FRM[7:0]								0000,0000
DMA_CAN2R_FCNTH	CAN2T_DMA 接收帧数寄存器	7EF55CH	CAN2R_FCNT[15:8]								0000,0000
DMA_CAN2R_FCNTL	CAN2T_DMA 接收帧数寄存器	7EF55DH	CAN2R_FCNT[7:0]								0000,0000
IAP_XDATA0	IAP 扩展数据寄存器 0	7EF5F0H	IAP_XDATA0[7:0]								0000,0000
IAP_XDATA1	IAP 扩展数据寄存器 1	7EF5F1H	IAP_XDATA1[7:0]								0000,0000
IAP_XDATA2	IAP 扩展数据寄存器 2	7EF5F2H	IAP_XDATA2[7:0]								0000,0000
IAP_XDATA3	IAP 扩展数据寄存器 3	7EF5F3H	IAP_XDATA3[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
IAP_XDATA4	IAP 扩展数据寄存器 4	7EF5F4H	IAP_XDATA4[7:0]								0000,0000
IAP_XDATA5	IAP 扩展数据寄存器 5	7EF5F5H	IAP_XDATA5[7:0]								0000,0000
IAP_XDATA6	IAP 扩展数据寄存器 6	7EF5F6H	IAP_XDATA6[7:0]								0000,0000
IAP_XDATA7	IAP 扩展数据寄存器 7	7EF5F7H	IAP_XDATA7[7:0]								0000,0000
IAP_XDATA8	IAP 扩展数据寄存器 8	7EF5F8H	IAP_XDATA8[7:0]								0000,0000
IAP_XDATA9	IAP 扩展数据寄存器 9	7EF5F9H	IAP_XDATA9[7:0]								0000,0000
IAP_XDATAA	IAP 扩展数据寄存器 A	7EF5FAH	IAP_XDATAA[7:0]								0000,0000
IAP_XDATAB	IAP 扩展数据寄存器 B	7EF5FBH	IAP_XDATAB[7:0]								0000,0000
IAP_XDATAC	IAP 扩展数据寄存器 C	7EF5FCH	IAP_XDATAC[7:0]								0000,0000
IAP_XDATAD	IAP 扩展数据寄存器 D	7EF5FDH	IAP_XDATAD[7:0]								0000,0000
IAP_XDATAE	IAP 扩展数据寄存器 E	7EF5FEH	IAP_XDATAE[7:0]								0000,0000
IAP_XDATAF	IAP 扩展数据寄存器 F	7EF5FFH	IAP_XDATAF[7:0]								0000,0000

13.13 扩展特殊功能寄存器列表（XFR：0x7EF400-0x7EF4FF）

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CAN_RBUF00	CAN 接收缓冲区寄存器	7EF400H	CAN_RBUF00[7:0]								0000,0000
CAN_RBUF01	CAN 接收缓冲区寄存器	7EF401H	CAN_RBUF01[7:0]								0000,0000
CAN_RBUF02	CAN 接收缓冲区寄存器	7EF402H	CAN_RBUF02[7:0]								0000,0000
CAN_RBUF03	CAN 接收缓冲区寄存器	7EF403H	CAN_RBUF03[7:0]								0000,0000
CAN_RBUF04	CAN 接收缓冲区寄存器	7EF404H	CAN_RBUF04[7:0]								0000,0000
CAN_RBUF05	CAN 接收缓冲区寄存器	7EF405H	CAN_RBUF05[7:0]								0000,0000
CAN_RBUF06	CAN 接收缓冲区寄存器	7EF406H	CAN_RBUF06[7:0]								0000,0000
CAN_RBUF07	CAN 接收缓冲区寄存器	7EF407H	CAN_RBUF07[7:0]								0000,0000
CAN_RBUF08	CAN 接收缓冲区寄存器	7EF408H	CAN_RBUF08[7:0]								0000,0000
CAN_RBUF09	CAN 接收缓冲区寄存器	7EF409H	CAN_RBUF09[7:0]								0000,0000
CAN_RBUF10	CAN 接收缓冲区寄存器	7EF40AH	CAN_RBUF10[7:0]								0000,0000
CAN_RBUF11	CAN 接收缓冲区寄存器	7EF40BH	CAN_RBUF11[7:0]								0000,0000
CAN_RBUF12	CAN 接收缓冲区寄存器	7EF40CH	CAN_RBUF12[7:0]								0000,0000
CAN_RBUF13	CAN 接收缓冲区寄存器	7EF40DH	CAN_RBUF13[7:0]								0000,0000
CAN_RBUF14	CAN 接收缓冲区寄存器	7EF40EH	CAN_RBUF14[7:0]								0000,0000
CAN_RBUF15	CAN 接收缓冲区寄存器	7EF40FH	CAN_RBUF15[7:0]								0000,0000
CAN_RBUF16	CAN 接收缓冲区寄存器	7EF410H	CAN_RBUF16[7:0]								0000,0000
CAN_RBUF17	CAN 接收缓冲区寄存器	7EF411H	CAN_RBUF17[7:0]								0000,0000
CAN_RBUF18	CAN 接收缓冲区寄存器	7EF412H	CAN_RBUF18[7:0]								0000,0000
CAN_RBUF19	CAN 接收缓冲区寄存器	7EF413H	CAN_RBUF19[7:0]								0000,0000
CAN_RBUF20	CAN 接收缓冲区寄存器	7EF414H	CAN_RBUF20[7:0]								0000,0000
CAN_RBUF21	CAN 接收缓冲区寄存器	7EF415H	CAN_RBUF21[7:0]								0000,0000
CAN_RBUF22	CAN 接收缓冲区寄存器	7EF416H	CAN_RBUF22[7:0]								0000,0000
CAN_RBUF23	CAN 接收缓冲区寄存器	7EF417H	CAN_RBUF23[7:0]								0000,0000
CAN_RBUF24	CAN 接收缓冲区寄存器	7EF418H	CAN_RBUF24[7:0]								0000,0000
CAN_RBUF25	CAN 接收缓冲区寄存器	7EF419H	CAN_RBUF25[7:0]								0000,0000
CAN_RBUF26	CAN 接收缓冲区寄存器	7EF41AH	CAN_RBUF26[7:0]								0000,0000
CAN_RBUF27	CAN 接收缓冲区寄存器	7EF41BH	CAN_RBUF27[7:0]								0000,0000
CAN_RBUF28	CAN 接收缓冲区寄存器	7EF41CH	CAN_RBUF28[7:0]								0000,0000
CAN_RBUF29	CAN 接收缓冲区寄存器	7EF41DH	CAN_RBUF29[7:0]								0000,0000
CAN_RBUF30	CAN 接收缓冲区寄存器	7EF41EH	CAN_RBUF30[7:0]								0000,0000
CAN_RBUF31	CAN 接收缓冲区寄存器	7EF41FH	CAN_RBUF31[7:0]								0000,0000
CAN_RBUF32	CAN 接收缓冲区寄存器	7EF420H	CAN_RBUF32[7:0]								0000,0000
CAN_RBUF33	CAN 接收缓冲区寄存器	7EF421H	CAN_RBUF33[7:0]								0000,0000
CAN_RBUF34	CAN 接收缓冲区寄存器	7EF422H	CAN_RBUF34[7:0]								0000,0000
CAN_RBUF35	CAN 接收缓冲区寄存器	7EF423H	CAN_RBUF35[7:0]								0000,0000
CAN_RBUF36	CAN 接收缓冲区寄存器	7EF424H	CAN_RBUF36[7:0]								0000,0000
CAN_RBUF37	CAN 接收缓冲区寄存器	7EF425H	CAN_RBUF37[7:0]								0000,0000
CAN_RBUF38	CAN 接收缓冲区寄存器	7EF426H	CAN_RBUF38[7:0]								0000,0000
CAN_RBUF39	CAN 接收缓冲区寄存器	7EF427H	CAN_RBUF39[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CAN_RBUF40	CAN 接收缓冲区寄存器	7EF428H	CAN_RBUF40[7:0]								0000,0000
CAN_RBUF41	CAN 接收缓冲区寄存器	7EF429H	CAN_RBUF41[7:0]								0000,0000
CAN_RBUF42	CAN 接收缓冲区寄存器	7EF42AH	CAN_RBUF42[7:0]								0000,0000
CAN_RBUF43	CAN 接收缓冲区寄存器	7EF42BH	CAN_RBUF43[7:0]								0000,0000
CAN_RBUF44	CAN 接收缓冲区寄存器	7EF42CH	CAN_RBUF44[7:0]								0000,0000
CAN_RBUF45	CAN 接收缓冲区寄存器	7EF42DH	CAN_RBUF45[7:0]								0000,0000
CAN_RBUF46	CAN 接收缓冲区寄存器	7EF42EH	CAN_RBUF46[7:0]								0000,0000
CAN_RBUF47	CAN 接收缓冲区寄存器	7EF42FH	CAN_RBUF47[7:0]								0000,0000
CAN_RBUF48	CAN 接收缓冲区寄存器	7EF430H	CAN_RBUF48[7:0]								0000,0000
CAN_RBUF49	CAN 接收缓冲区寄存器	7EF431H	CAN_RBUF49[7:0]								0000,0000
CAN_RBUF50	CAN 接收缓冲区寄存器	7EF432H	CAN_RBUF50[7:0]								0000,0000
CAN_RBUF51	CAN 接收缓冲区寄存器	7EF433H	CAN_RBUF51[7:0]								0000,0000
CAN_RBUF52	CAN 接收缓冲区寄存器	7EF434H	CAN_RBUF52[7:0]								0000,0000
CAN_RBUF53	CAN 接收缓冲区寄存器	7EF435H	CAN_RBUF53[7:0]								0000,0000
CAN_RBUF54	CAN 接收缓冲区寄存器	7EF436H	CAN_RBUF54[7:0]								0000,0000
CAN_RBUF55	CAN 接收缓冲区寄存器	7EF437H	CAN_RBUF55[7:0]								0000,0000
CAN_RBUF56	CAN 接收缓冲区寄存器	7EF438H	CAN_RBUF56[7:0]								0000,0000
CAN_RBUF57	CAN 接收缓冲区寄存器	7EF439H	CAN_RBUF57[7:0]								0000,0000
CAN_RBUF58	CAN 接收缓冲区寄存器	7EF43AH	CAN_RBUF58[7:0]								0000,0000
CAN_RBUF59	CAN 接收缓冲区寄存器	7EF43BH	CAN_RBUF59[7:0]								0000,0000
CAN_RBUF60	CAN 接收缓冲区寄存器	7EF43CH	CAN_RBUF60[7:0]								0000,0000
CAN_RBUF61	CAN 接收缓冲区寄存器	7EF43DH	CAN_RBUF61[7:0]								0000,0000
CAN_RBUF62	CAN 接收缓冲区寄存器	7EF43EH	CAN_RBUF62[7:0]								0000,0000
CAN_RBUF63	CAN 接收缓冲区寄存器	7EF43FH	CAN_RBUF63[7:0]								0000,0000
CAN_RBUF64	CAN 接收缓冲区寄存器	7EF440H	CAN_RBUF64[7:0]								0000,0000
CAN_RBUF65	CAN 接收缓冲区寄存器	7EF441H	CAN_RBUF65[7:0]								0000,0000
CAN_RBUF66	CAN 接收缓冲区寄存器	7EF442H	CAN_RBUF66[7:0]								0000,0000
CAN_RBUF67	CAN 接收缓冲区寄存器	7EF443H	CAN_RBUF67[7:0]								0000,0000
CAN_RBUF68	CAN 接收缓冲区寄存器	7EF444H	CAN_RBUF68[7:0]								0000,0000
CAN_RBUF69	CAN 接收缓冲区寄存器	7EF445H	CAN_RBUF69[7:0]								0000,0000
CAN_RBUF70	CAN 接收缓冲区寄存器	7EF446H	CAN_RBUF70[7:0]								0000,0000
CAN_RBUF71	CAN 接收缓冲区寄存器	7EF447H	CAN_RBUF71[7:0]								0000,0000
CAN_RBUF72	CAN 接收缓冲区寄存器	7EF448H	CAN_RBUF72[7:0]								0000,0000
CAN_RBUF73	CAN 接收缓冲区寄存器	7EF449H	CAN_RBUF73[7:0]								0000,0000
CAN_RBUF74	CAN 接收缓冲区寄存器	7EF44AH	CAN_RBUF74[7:0]								0000,0000
CAN_RBUF75	CAN 接收缓冲区寄存器	7EF44BH	CAN_RBUF75[7:0]								0000,0000
CAN_RBUF76	CAN 接收缓冲区寄存器	7EF44CH	CAN_RBUF76[7:0]								0000,0000
CAN_RBUF77	CAN 接收缓冲区寄存器	7EF44DH	CAN_RBUF77[7:0]								0000,0000
CAN_RBUF78	CAN 接收缓冲区寄存器	7EF44EH	CAN_RBUF78[7:0]								0000,0000
CAN_RBUF79	CAN 接收缓冲区寄存器	7EF44FH	CAN_RBUF79[7:0]								0000,0000
CAN_TBUF00	CAN 发送缓冲区寄存器	7EF450H	CAN_TBUF00[7:0]								0000,0000
CAN_TBUF01	CAN 发送缓冲区寄存器	7EF451H	CAN_TBUF01[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CAN_TBUF02	CAN 发送缓冲区寄存器	7EF452H	CAN_TBUF02[7:0]								0000,0000
CAN_TBUF03	CAN 发送缓冲区寄存器	7EF453H	CAN_TBUF03[7:0]								0000,0000
CAN_TBUF04	CAN 发送缓冲区寄存器	7EF454H	CAN_TBUF04[7:0]								0000,0000
CAN_TBUF05	CAN 发送缓冲区寄存器	7EF455H	CAN_TBUF05[7:0]								0000,0000
CAN_TBUF06	CAN 发送缓冲区寄存器	7EF456H	CAN_TBUF06[7:0]								0000,0000
CAN_TBUF07	CAN 发送缓冲区寄存器	7EF457H	CAN_TBUF07[7:0]								0000,0000
CAN_TBUF08	CAN 发送缓冲区寄存器	7EF458H	CAN_TBUF08[7:0]								0000,0000
CAN_TBUF09	CAN 发送缓冲区寄存器	7EF459H	CAN_TBUF09[7:0]								0000,0000
CAN_TBUF10	CAN 发送缓冲区寄存器	7EF45AH	CAN_TBUF10[7:0]								0000,0000
CAN_TBUF11	CAN 发送缓冲区寄存器	7EF45BH	CAN_TBUF11[7:0]								0000,0000
CAN_TBUF12	CAN 发送缓冲区寄存器	7EF45CH	CAN_TBUF12[7:0]								0000,0000
CAN_TBUF13	CAN 发送缓冲区寄存器	7EF45DH	CAN_TBUF13[7:0]								0000,0000
CAN_TBUF14	CAN 发送缓冲区寄存器	7EF45EH	CAN_TBUF14[7:0]								0000,0000
CAN_TBUF15	CAN 发送缓冲区寄存器	7EF45FH	CAN_TBUF15[7:0]								0000,0000
CAN_TBUF16	CAN 发送缓冲区寄存器	7EF460H	CAN_TBUF16[7:0]								0000,0000
CAN_TBUF17	CAN 发送缓冲区寄存器	7EF461H	CAN_TBUF17[7:0]								0000,0000
CAN_TBUF18	CAN 发送缓冲区寄存器	7EF462H	CAN_TBUF18[7:0]								0000,0000
CAN_TBUF19	CAN 发送缓冲区寄存器	7EF463H	CAN_TBUF19[7:0]								0000,0000
CAN_TBUF20	CAN 发送缓冲区寄存器	7EF464H	CAN_TBUF20[7:0]								0000,0000
CAN_TBUF21	CAN 发送缓冲区寄存器	7EF465H	CAN_TBUF21[7:0]								0000,0000
CAN_TBUF22	CAN 发送缓冲区寄存器	7EF466H	CAN_TBUF22[7:0]								0000,0000
CAN_TBUF23	CAN 发送缓冲区寄存器	7EF467H	CAN_TBUF23[7:0]								0000,0000
CAN_TBUF24	CAN 发送缓冲区寄存器	7EF468H	CAN_TBUF24[7:0]								0000,0000
CAN_TBUF25	CAN 发送缓冲区寄存器	7EF469H	CAN_TBUF25[7:0]								0000,0000
CAN_TBUF26	CAN 发送缓冲区寄存器	7EF46AH	CAN_TBUF26[7:0]								0000,0000
CAN_TBUF27	CAN 发送缓冲区寄存器	7EF46BH	CAN_TBUF27[7:0]								0000,0000
CAN_TBUF28	CAN 发送缓冲区寄存器	7EF46CH	CAN_TBUF28[7:0]								0000,0000
CAN_TBUF29	CAN 发送缓冲区寄存器	7EF46DH	CAN_TBUF29[7:0]								0000,0000
CAN_TBUF30	CAN 发送缓冲区寄存器	7EF46EH	CAN_TBUF30[7:0]								0000,0000
CAN_TBUF31	CAN 发送缓冲区寄存器	7EF46FH	CAN_TBUF31[7:0]								0000,0000
CAN_TBUF32	CAN 发送缓冲区寄存器	7EF470H	CAN_TBUF32[7:0]								0000,0000
CAN_TBUF33	CAN 发送缓冲区寄存器	7EF471H	CAN_TBUF33[7:0]								0000,0000
CAN_TBUF34	CAN 发送缓冲区寄存器	7EF472H	CAN_TBUF34[7:0]								0000,0000
CAN_TBUF35	CAN 发送缓冲区寄存器	7EF473H	CAN_TBUF35[7:0]								0000,0000
CAN_TBUF36	CAN 发送缓冲区寄存器	7EF474H	CAN_TBUF36[7:0]								0000,0000
CAN_TBUF37	CAN 发送缓冲区寄存器	7EF475H	CAN_TBUF37[7:0]								0000,0000
CAN_TBUF38	CAN 发送缓冲区寄存器	7EF476H	CAN_TBUF38[7:0]								0000,0000
CAN_TBUF39	CAN 发送缓冲区寄存器	7EF477H	CAN_TBUF39[7:0]								0000,0000
CAN_TBUF40	CAN 发送缓冲区寄存器	7EF478H	CAN_TBUF40[7:0]								0000,0000
CAN_TBUF41	CAN 发送缓冲区寄存器	7EF479H	CAN_TBUF41[7:0]								0000,0000
CAN_TBUF42	CAN 发送缓冲区寄存器	7EF47AH	CAN_TBUF42[7:0]								0000,0000
CAN_TBUF43	CAN 发送缓冲区寄存器	7EF47BH	CAN_TBUF43[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CAN_TBUF44	CAN 发送缓冲区寄存器	7EF47CH	CAN_TBUF44[7:0]								0000,0000
CAN_TBUF45	CAN 发送缓冲区寄存器	7EF47DH	CAN_TBUF45[7:0]								0000,0000
CAN_TBUF46	CAN 发送缓冲区寄存器	7EF47EH	CAN_TBUF46[7:0]								0000,0000
CAN_TBUF47	CAN 发送缓冲区寄存器	7EF47FH	CAN_TBUF47[7:0]								0000,0000
CAN_TBUF48	CAN 发送缓冲区寄存器	7EF480H	CAN_TBUF48[7:0]								0000,0000
CAN_TBUF49	CAN 发送缓冲区寄存器	7EF481H	CAN_TBUF49[7:0]								0000,0000
CAN_TBUF50	CAN 发送缓冲区寄存器	7EF482H	CAN_TBUF50[7:0]								0000,0000
CAN_TBUF51	CAN 发送缓冲区寄存器	7EF483H	CAN_TBUF51[7:0]								0000,0000
CAN_TBUF52	CAN 发送缓冲区寄存器	7EF484H	CAN_TBUF52[7:0]								0000,0000
CAN_TBUF53	CAN 发送缓冲区寄存器	7EF485H	CAN_TBUF53[7:0]								0000,0000
CAN_TBUF54	CAN 发送缓冲区寄存器	7EF486H	CAN_TBUF54[7:0]								0000,0000
CAN_TBUF55	CAN 发送缓冲区寄存器	7EF487H	CAN_TBUF55[7:0]								0000,0000
CAN_TBUF56	CAN 发送缓冲区寄存器	7EF488H	CAN_TBUF56[7:0]								0000,0000
CAN_TBUF57	CAN 发送缓冲区寄存器	7EF489H	CAN_TBUF57[7:0]								0000,0000
CAN_TBUF58	CAN 发送缓冲区寄存器	7EF48AH	CAN_TBUF58[7:0]								0000,0000
CAN_TBUF59	CAN 发送缓冲区寄存器	7EF48BH	CAN_TBUF59[7:0]								0000,0000
CAN_TBUF60	CAN 发送缓冲区寄存器	7EF48CH	CAN_TBUF60[7:0]								0000,0000
CAN_TBUF61	CAN 发送缓冲区寄存器	7EF48DH	CAN_TBUF61[7:0]								0000,0000
CAN_TBUF62	CAN 发送缓冲区寄存器	7EF48EH	CAN_TBUF62[7:0]								0000,0000
CAN_TBUF63	CAN 发送缓冲区寄存器	7EF48FH	CAN_TBUF63[7:0]								0000,0000
CAN_TBUF64	CAN 发送缓冲区寄存器	7EF490H	CAN_TBUF64[7:0]								0000,0000
CAN_TBUF65	CAN 发送缓冲区寄存器	7EF491H	CAN_TBUF65[7:0]								0000,0000
CAN_TBUF66	CAN 发送缓冲区寄存器	7EF492H	CAN_TBUF66[7:0]								0000,0000
CAN_TBUF67	CAN 发送缓冲区寄存器	7EF493H	CAN_TBUF67[7:0]								0000,0000
CAN_TBUF68	CAN 发送缓冲区寄存器	7EF494H	CAN_TBUF68[7:0]								0000,0000
CAN_TBUF69	CAN 发送缓冲区寄存器	7EF495H	CAN_TBUF69[7:0]								0000,0000
CAN_TBUF70	CAN 发送缓冲区寄存器	7EF496H	CAN_TBUF70[7:0]								0000,0000
CAN_TBUF71	CAN 发送缓冲区寄存器	7EF497H	CAN_TBUF71[7:0]								0000,0000
CAN_TTS00	CAN 传输时间戳	7EF498H	CAN_TTS00[7:0]								0000,0000
CAN_TTS01	CAN 传输时间戳	7EF499H	CAN_TTS01[7:0]								0000,0000
CAN_TTS02	CAN 传输时间戳	7EF49AH	CAN_TTS02[7:0]								0000,0000
CAN_TTS03	CAN 传输时间戳	7EF49BH	CAN_TTS03[7:0]								0000,0000
CAN_TTS04	CAN 传输时间戳	7EF49CH	CAN_TTS04[7:0]								0000,0000
CAN_TTS05	CAN 传输时间戳	7EF49DH	CAN_TTS05[7:0]								0000,0000
CAN_TTS06	CAN 传输时间戳	7EF49EH	CAN_TTS06[7:0]								0000,0000
CAN_TTS07	CAN 传输时间戳	7EF49FH	CAN_TTS07[7:0]								0000,0000
CAN_CFG_STAT	CAN 配置状态寄存器	7EF4A0H	RESET	LBME	LBMI	TPSS	TSSS	RACTIVE	TACTIVE	BUSOFF	1000,0000
CAN_TCMD	CAN 命令寄存器	7EF4A1H	TBSEL	LOM	-	TPE	TPA	TSONE	TSALL	TSA	0000,0000
CAN_TCTRL	CAN 发送控制寄存器	7EF4A2H	FD_ISO	TSNEXT	TSMODE	TTTBM	-	-	TSSTAT[1:0]		1001,0000
CAN_RCTRL	CAN 接收控制寄存器	7EF4A3H	SACK	ROM	ROV	RREL	RBALL	-	RSTAT[1:0]		0000,0000
CAN_RTIE	CAN 中断使能寄存器	7EF4A4H	RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	TSFF	1111,1110
CAN_RTIF	CAN 中断状态寄存器	7EF4A5H	RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF	0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CAN_ERRINT	CAN 错误中断寄存器	7EF4A6H	EWARN	EPASS	EPIE	EPIF	ALIE	ALIF	BEIE	BEIF	0000,0000
CAN_LIMIT	CAN 警告限制寄存器	7EF4A7H	AFWL[3:0]				EWL[3:0]				0001,1011
CAN_S_SEG1	CAN 慢速位段 1 时间设定	7EF4A8H	S_SEG1[7:0]								0000,0000
CAN_S_SEG2	CAN 慢速位段 2 时间设定	7EF4A9H	-	S_SEG2[6:0]							0000,0000
CAN_S_SJW	CAN 慢速同步补偿时间设定	7EF4AAH	-	S_SJW[6:0]							0000,0000
CAN_S_PRESC	CAN 慢速预分频寄存器	7EF4ABH	S_PRESC[7:0]								0000,0000
CAN_F_SEG1	CAN 快速位段 1 时间设定	7EF4ACH	FS_SEG1[7:0]								0000,0000
CAN_F_SEG2	CAN 快速位段 2 时间设定	7EF4ADH	-	F_SEG2[6:0]							0000,0000
CAN_F_SJW	CAN 快速同步补偿时间设定	7EF4AEH	-	F_SJW[6:0]							0000,0000
CAN_F_PRESC	CAN 快速预分频寄存器	7EF4AFH	F_PRESC[7:0]								0000,0000
CAN_EALCAP	CAN 错误和仲裁丢失捕获	7EF4B0H	KOER[2:0]			ALC[4:0]					0000,0000
CAN_TDC	CAN 发送器延迟补偿	7EF4B1H	TDCEN	SSPOFF[6:0]							0000,0000
CAN_RECNT	CAN 接收错误计数器	7EF4B2H	RECNT[7:0]								0000,0000
CAN_TECNT	CAN 发送错误计数器	7EF4B3H	TECNT[7:0]								0000,0000
CAN_ACFCTRL	CAN 筛选器组控制寄存器	7EF4B4H	-	-	SELMASK	-	ACFADR[3:0]				0000,0000
CAN_TIMECFG	CAN 时间戳控制寄存器	7EF4B5H	-	-	-	-	-	-	TIMEPOS	TIMEEN	0000,0000
CAN_ACF_EN0	CAN 筛选器组使能寄存器	7EF4B6H	AE_7	AE_6	AE_5	AE_4	AE_3	AE_2	AE_1	AE_0	0000,0001
CAN_ACF_EN1	CAN 筛选器组使能寄存器	7EF4B7H	AE_15	AE_14	AE_13	AE_12	AE_11	AE_10	AE_9	AE_8	0000,0000
CAN_ACF0	CAN 筛选器 code 和 mask	7EF4B8H	ACODE[7:0]/AMASK[7:0]								0000,0000
CAN_ACF1	CAN 筛选器 code 和 mask	7EF4B9H	ACODE[15:8]/AMASK[15:8]								0000,0000
CAN_ACF2	CAN 筛选器 code 和 mask	7EF4BAH	ACODE[23:16]/AMASK[23:16]								0000,0000
CAN_ACF3	CAN 筛选器 code 和 mask	7EF4BBH	-	AIDEE	AIDE	ACODE[28:24]/AMASK[28:24]					0000,0000
CAN_VER0	CAN 版本信息	7EF4BCH	VER0[7:0]								0000,0000
CAN_VER1	CAN 版本信息	7EF4BDH	VER1[7:0]								0000,0000
CAN_TBSLOT	CAN TB SLOT 指针寄存器	7EF4BEH	TBE	TBF	-	-	-	TBPTR[2:0]			0000,0000
CAN_TTCFG	CAN 时间触发配置寄存器	7EF4BFH	WTIE	WTIF	TEIF	TTIE	TTIF	T_PRESC[1:0]		TTEN	0000,0000
CAN_REF_MSG0	CAN 参考消息	7EF4C0H	REF_ID[7:0]								0000,0000
CAN_REF_MSG1	CAN 参考消息	7EF4C1H	REF_ID[15:8]								0000,0000
CAN_REF_MSG2	CAN 参考消息	7EF4C2H	REF_ID[23:16]								0000,0000
CAN_REF_MSG3	CAN 参考消息	7EF4C3H	REF_IDE	-	-	REF_ID[28:24]					0000,0000
CAN_TRIG_CFG0	CAN 触发配置寄存器	7EF4C4H	-	-	TTPTR[5:0]						0000,0000
CAN_TRIG_CFG1	CAN 触发配置寄存器	7EF4C5H	TEW[3:0]				-	TTYPE[2:0]			0000,0000
CAN_TT_TRIG0	CAN 触发时间寄存器	7EF4C6H	TT_TRIG[7:0]								0000,0000
CAN_TT_TRIG1	CAN 触发时间寄存器	7EF4C7H	TT_TRIG[15:8]								0000,0000
CAN_TT_WTRIG0	CAN 触发看门时间寄存器	7EF4C8H	TT_WTRIG[7:0]								0000,0000
CAN_TT_WTRIG1	CAN 触发看门时间寄存器	7EF4C9H	TT_WTRIG[15:8]								0000,0000
P0IM2	P0 口中断模式寄存器 2	7EF4E0H	P07IM2	P06IM2	P05IM2	P04IM2	P03IM2	P02IM2	P01IM2	P00IM2	0000,0000
P1IM2	P1 口中断模式寄存器 2	7EF4E1H	P17IM2	P16IM2	P15IM2	P14IM2	P13IM2	P12IM2	P11IM2	P10IM2	0000,0000
P2IM2	P2 口中断模式寄存器 2	7EF4E2H	P27IM2	P26IM2	P25IM2	P24IM2	P23IM2	P22IM2	P21IM2	P20IM2	0000,0000
P3IM2	P3 口中断模式寄存器 2	7EF4E3H	P37IM2	P36IM2	P35IM2	P34IM2	P33IM2	P32IM2	P31IM2	P30IM2	0000,0000
P4IM2	P4 口中断模式寄存器 2	7EF4E4H	P47IM2	P46IM2	P45IM2	P44IM2	P43IM2	P42IM2	P41IM2	P40IM2	0000,0000
P5IM2	P5 口中断模式寄存器 2	7EF4E5H	P57IM2	P56IM2	P55IM2	P54IM2	P53IM2	P52IM2	P51IM2	P50IM2	0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P6IM2	P6 口中断模式寄存器 2	7EF4E6H	P67IM2	P66IM2	P65IM2	P64IM2	P63IM2	P62IM2	P61IM2	P60IM2	0000,0000
P7IM2	P7 口中断模式寄存器 2	7EF4E7H	P77IM2	P76IM2	P75IM2	P74IM2	P73IM2	P72IM2	P71IM2	P70IM2	0000,0000
P8IM2	P8 口中断模式寄存器 2	7EF4E8H	P87IM2	P86IM2	P85IM2	P84IM2	P83IM2	P82IM2	P81IM2	P80IM2	0000,0000
P9IM2	P9 口中断模式寄存器 2	7EF4E9H	P97IM2	P96IM2	P95IM2	P94IM2	P93IM2	P92IM2	P91IM2	P90IM2	0000,0000
PAIM2	PA 口中断模式寄存器 2	7EF4EAH	PA7IM2	PA6IM2	PA5IM2	PA4IM2	PA3IM2	PA2IM2	PA1IM2	PA0IM2	0000,0000
PBIM2	PB 口中断模式寄存器 2	7EF4EBH	PB7IM2	PB6IM2	PB5IM2	PB4IM2	PB3IM2	PB2IM2	PB1IM2	PB0IM2	0000,0000

13.14 扩展特殊功能寄存器列表（XFR：0x7EF300-0x7EF3FF）

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CAN2_RBUF00	CAN2 接收缓冲区寄存器	7EF300H	CAN2_RBUF00[7:0]								0000,0000
CAN2_RBUF01	CAN2 接收缓冲区寄存器	7EF301H	CAN2_RBUF01[7:0]								0000,0000
CAN2_RBUF02	CAN2 接收缓冲区寄存器	7EF302H	CAN2_RBUF02[7:0]								0000,0000
CAN2_RBUF03	CAN2 接收缓冲区寄存器	7EF303H	CAN2_RBUF03[7:0]								0000,0000
CAN2_RBUF04	CAN2 接收缓冲区寄存器	7EF304H	CAN2_RBUF04[7:0]								0000,0000
CAN2_RBUF05	CAN2 接收缓冲区寄存器	7EF305H	CAN2_RBUF05[7:0]								0000,0000
CAN2_RBUF06	CAN2 接收缓冲区寄存器	7EF306H	CAN2_RBUF06[7:0]								0000,0000
CAN2_RBUF07	CAN2 接收缓冲区寄存器	7EF307H	CAN2_RBUF07[7:0]								0000,0000
CAN2_RBUF08	CAN2 接收缓冲区寄存器	7EF308H	CAN2_RBUF08[7:0]								0000,0000
CAN2_RBUF09	CAN2 接收缓冲区寄存器	7EF309H	CAN2_RBUF09[7:0]								0000,0000
CAN2_RBUF10	CAN2 接收缓冲区寄存器	7EF30AH	CAN2_RBUF10[7:0]								0000,0000
CAN2_RBUF11	CAN2 接收缓冲区寄存器	7EF30BH	CAN2_RBUF11[7:0]								0000,0000
CAN2_RBUF12	CAN2 接收缓冲区寄存器	7EF30CH	CAN2_RBUF12[7:0]								0000,0000
CAN2_RBUF13	CAN2 接收缓冲区寄存器	7EF30DH	CAN2_RBUF13[7:0]								0000,0000
CAN2_RBUF14	CAN2 接收缓冲区寄存器	7EF30EH	CAN2_RBUF14[7:0]								0000,0000
CAN2_RBUF15	CAN2 接收缓冲区寄存器	7EF30FH	CAN2_RBUF15[7:0]								0000,0000
CAN2_RBUF16	CAN2 接收缓冲区寄存器	7EF310H	CAN2_RBUF16[7:0]								0000,0000
CAN2_RBUF17	CAN2 接收缓冲区寄存器	7EF311H	CAN2_RBUF17[7:0]								0000,0000
CAN2_RBUF18	CAN2 接收缓冲区寄存器	7EF312H	CAN2_RBUF18[7:0]								0000,0000
CAN2_RBUF19	CAN2 接收缓冲区寄存器	7EF313H	CAN2_RBUF19[7:0]								0000,0000
CAN2_RBUF20	CAN2 接收缓冲区寄存器	7EF314H	CAN2_RBUF20[7:0]								0000,0000
CAN2_RBUF21	CAN2 接收缓冲区寄存器	7EF315H	CAN2_RBUF21[7:0]								0000,0000
CAN2_RBUF22	CAN2 接收缓冲区寄存器	7EF316H	CAN2_RBUF22[7:0]								0000,0000
CAN2_RBUF23	CAN2 接收缓冲区寄存器	7EF317H	CAN2_RBUF23[7:0]								0000,0000
CAN2_RBUF24	CAN2 接收缓冲区寄存器	7EF318H	CAN2_RBUF24[7:0]								0000,0000
CAN2_RBUF25	CAN2 接收缓冲区寄存器	7EF319H	CAN2_RBUF25[7:0]								0000,0000
CAN2_RBUF26	CAN2 接收缓冲区寄存器	7EF31AH	CAN2_RBUF26[7:0]								0000,0000
CAN2_RBUF27	CAN2 接收缓冲区寄存器	7EF31BH	CAN2_RBUF27[7:0]								0000,0000
CAN2_RBUF28	CAN2 接收缓冲区寄存器	7EF31CH	CAN2_RBUF28[7:0]								0000,0000
CAN2_RBUF29	CAN2 接收缓冲区寄存器	7EF31DH	CAN2_RBUF29[7:0]								0000,0000
CAN2_RBUF30	CAN2 接收缓冲区寄存器	7EF31EH	CAN2_RBUF30[7:0]								0000,0000
CAN2_RBUF31	CAN2 接收缓冲区寄存器	7EF31FH	CAN2_RBUF31[7:0]								0000,0000
CAN2_RBUF32	CAN2 接收缓冲区寄存器	7EF320H	CAN2_RBUF32[7:0]								0000,0000
CAN2_RBUF33	CAN2 接收缓冲区寄存器	7EF321H	CAN2_RBUF33[7:0]								0000,0000
CAN2_RBUF34	CAN2 接收缓冲区寄存器	7EF322H	CAN2_RBUF34[7:0]								0000,0000
CAN2_RBUF35	CAN2 接收缓冲区寄存器	7EF323H	CAN2_RBUF35[7:0]								0000,0000
CAN2_RBUF36	CAN2 接收缓冲区寄存器	7EF324H	CAN2_RBUF36[7:0]								0000,0000
CAN2_RBUF37	CAN2 接收缓冲区寄存器	7EF325H	CAN2_RBUF37[7:0]								0000,0000
CAN2_RBUF38	CAN2 接收缓冲区寄存器	7EF326H	CAN2_RBUF38[7:0]								0000,0000
CAN2_RBUF39	CAN2 接收缓冲区寄存器	7EF327H	CAN2_RBUF39[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CAN2_RBUF40	CAN2 接收缓冲区寄存器	7EF328H	CAN2_RBUF40[7:0]								0000,0000
CAN2_RBUF41	CAN2 接收缓冲区寄存器	7EF329H	CAN2_RBUF41[7:0]								0000,0000
CAN2_RBUF42	CAN2 接收缓冲区寄存器	7EF32AH	CAN2_RBUF42[7:0]								0000,0000
CAN2_RBUF43	CAN2 接收缓冲区寄存器	7EF32BH	CAN2_RBUF43[7:0]								0000,0000
CAN2_RBUF44	CAN2 接收缓冲区寄存器	7EF32CH	CAN2_RBUF44[7:0]								0000,0000
CAN2_RBUF45	CAN2 接收缓冲区寄存器	7EF32DH	CAN2_RBUF45[7:0]								0000,0000
CAN2_RBUF46	CAN2 接收缓冲区寄存器	7EF32EH	CAN2_RBUF46[7:0]								0000,0000
CAN2_RBUF47	CAN2 接收缓冲区寄存器	7EF32FH	CAN2_RBUF47[7:0]								0000,0000
CAN2_RBUF48	CAN2 接收缓冲区寄存器	7EF330H	CAN2_RBUF48[7:0]								0000,0000
CAN2_RBUF49	CAN2 接收缓冲区寄存器	7EF331H	CAN2_RBUF49[7:0]								0000,0000
CAN2_RBUF50	CAN2 接收缓冲区寄存器	7EF332H	CAN2_RBUF50[7:0]								0000,0000
CAN2_RBUF51	CAN2 接收缓冲区寄存器	7EF333H	CAN2_RBUF51[7:0]								0000,0000
CAN2_RBUF52	CAN2 接收缓冲区寄存器	7EF334H	CAN2_RBUF52[7:0]								0000,0000
CAN2_RBUF53	CAN2 接收缓冲区寄存器	7EF335H	CAN2_RBUF53[7:0]								0000,0000
CAN2_RBUF54	CAN2 接收缓冲区寄存器	7EF336H	CAN2_RBUF54[7:0]								0000,0000
CAN2_RBUF55	CAN2 接收缓冲区寄存器	7EF337H	CAN2_RBUF55[7:0]								0000,0000
CAN2_RBUF56	CAN2 接收缓冲区寄存器	7EF338H	CAN2_RBUF56[7:0]								0000,0000
CAN2_RBUF57	CAN2 接收缓冲区寄存器	7EF339H	CAN2_RBUF57[7:0]								0000,0000
CAN2_RBUF58	CAN2 接收缓冲区寄存器	7EF33AH	CAN2_RBUF58[7:0]								0000,0000
CAN2_RBUF59	CAN2 接收缓冲区寄存器	7EF33BH	CAN2_RBUF59[7:0]								0000,0000
CAN2_RBUF60	CAN2 接收缓冲区寄存器	7EF33CH	CAN2_RBUF60[7:0]								0000,0000
CAN2_RBUF61	CAN2 接收缓冲区寄存器	7EF33DH	CAN2_RBUF61[7:0]								0000,0000
CAN2_RBUF62	CAN2 接收缓冲区寄存器	7EF33EH	CAN2_RBUF62[7:0]								0000,0000
CAN2_RBUF63	CAN2 接收缓冲区寄存器	7EF33FH	CAN2_RBUF63[7:0]								0000,0000
CAN2_RBUF64	CAN2 接收缓冲区寄存器	7EF340H	CAN2_RBUF64[7:0]								0000,0000
CAN2_RBUF65	CAN2 接收缓冲区寄存器	7EF341H	CAN2_RBUF65[7:0]								0000,0000
CAN2_RBUF66	CAN2 接收缓冲区寄存器	7EF342H	CAN2_RBUF66[7:0]								0000,0000
CAN2_RBUF67	CAN2 接收缓冲区寄存器	7EF343H	CAN2_RBUF67[7:0]								0000,0000
CAN2_RBUF68	CAN2 接收缓冲区寄存器	7EF344H	CAN2_RBUF68[7:0]								0000,0000
CAN2_RBUF69	CAN2 接收缓冲区寄存器	7EF345H	CAN2_RBUF69[7:0]								0000,0000
CAN2_RBUF70	CAN2 接收缓冲区寄存器	7EF346H	CAN2_RBUF70[7:0]								0000,0000
CAN2_RBUF71	CAN2 接收缓冲区寄存器	7EF347H	CAN2_RBUF71[7:0]								0000,0000
CAN2_RBUF72	CAN2 接收缓冲区寄存器	7EF348H	CAN2_RBUF72[7:0]								0000,0000
CAN2_RBUF73	CAN2 接收缓冲区寄存器	7EF349H	CAN2_RBUF73[7:0]								0000,0000
CAN2_RBUF74	CAN2 接收缓冲区寄存器	7EF34AH	CAN2_RBUF74[7:0]								0000,0000
CAN2_RBUF75	CAN2 接收缓冲区寄存器	7EF34BH	CAN2_RBUF75[7:0]								0000,0000
CAN2_RBUF76	CAN2 接收缓冲区寄存器	7EF34CH	CAN2_RBUF76[7:0]								0000,0000
CAN2_RBUF77	CAN2 接收缓冲区寄存器	7EF34DH	CAN2_RBUF77[7:0]								0000,0000
CAN2_RBUF78	CAN2 接收缓冲区寄存器	7EF34EH	CAN2_RBUF78[7:0]								0000,0000
CAN2_RBUF79	CAN2 接收缓冲区寄存器	7EF34FH	CAN2_RBUF79[7:0]								0000,0000
CAN2_TBUF00	CAN2 发送缓冲区寄存器	7EF350H	CAN2_TBUF00[7:0]								0000,0000
CAN2_TBUF01	CAN2 发送缓冲区寄存器	7EF351H	CAN2_TBUF01[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CAN2_TBUF02	CAN2 发送缓冲区寄存器	7EF352H	CAN2_TBUF02[7:0]								0000,0000
CAN2_TBUF03	CAN2 发送缓冲区寄存器	7EF353H	CAN2_TBUF03[7:0]								0000,0000
CAN2_TBUF04	CAN2 发送缓冲区寄存器	7EF354H	CAN2_TBUF04[7:0]								0000,0000
CAN2_TBUF05	CAN2 发送缓冲区寄存器	7EF355H	CAN2_TBUF05[7:0]								0000,0000
CAN2_TBUF06	CAN2 发送缓冲区寄存器	7EF356H	CAN2_TBUF06[7:0]								0000,0000
CAN2_TBUF07	CAN2 发送缓冲区寄存器	7EF357H	CAN2_TBUF07[7:0]								0000,0000
CAN2_TBUF08	CAN2 发送缓冲区寄存器	7EF358H	CAN2_TBUF08[7:0]								0000,0000
CAN2_TBUF09	CAN2 发送缓冲区寄存器	7EF359H	CAN2_TBUF09[7:0]								0000,0000
CAN2_TBUF10	CAN2 发送缓冲区寄存器	7EF35AH	CAN2_TBUF10[7:0]								0000,0000
CAN2_TBUF11	CAN2 发送缓冲区寄存器	7EF35BH	CAN2_TBUF11[7:0]								0000,0000
CAN2_TBUF12	CAN2 发送缓冲区寄存器	7EF35CH	CAN2_TBUF12[7:0]								0000,0000
CAN2_TBUF13	CAN2 发送缓冲区寄存器	7EF35DH	CAN2_TBUF13[7:0]								0000,0000
CAN2_TBUF14	CAN2 发送缓冲区寄存器	7EF35EH	CAN2_TBUF14[7:0]								0000,0000
CAN2_TBUF15	CAN2 发送缓冲区寄存器	7EF35FH	CAN2_TBUF15[7:0]								0000,0000
CAN2_TBUF16	CAN2 发送缓冲区寄存器	7EF360H	CAN2_TBUF16[7:0]								0000,0000
CAN2_TBUF17	CAN2 发送缓冲区寄存器	7EF361H	CAN2_TBUF17[7:0]								0000,0000
CAN2_TBUF18	CAN2 发送缓冲区寄存器	7EF362H	CAN2_TBUF18[7:0]								0000,0000
CAN2_TBUF19	CAN2 发送缓冲区寄存器	7EF363H	CAN2_TBUF19[7:0]								0000,0000
CAN2_TBUF20	CAN2 发送缓冲区寄存器	7EF364H	CAN2_TBUF20[7:0]								0000,0000
CAN2_TBUF21	CAN2 发送缓冲区寄存器	7EF365H	CAN2_TBUF21[7:0]								0000,0000
CAN2_TBUF22	CAN2 发送缓冲区寄存器	7EF366H	CAN2_TBUF22[7:0]								0000,0000
CAN2_TBUF23	CAN2 发送缓冲区寄存器	7EF367H	CAN2_TBUF23[7:0]								0000,0000
CAN2_TBUF24	CAN2 发送缓冲区寄存器	7EF368H	CAN2_TBUF24[7:0]								0000,0000
CAN2_TBUF25	CAN2 发送缓冲区寄存器	7EF369H	CAN2_TBUF25[7:0]								0000,0000
CAN2_TBUF26	CAN2 发送缓冲区寄存器	7EF36AH	CAN2_TBUF26[7:0]								0000,0000
CAN2_TBUF27	CAN2 发送缓冲区寄存器	7EF36BH	CAN2_TBUF27[7:0]								0000,0000
CAN2_TBUF28	CAN2 发送缓冲区寄存器	7EF36CH	CAN2_TBUF28[7:0]								0000,0000
CAN2_TBUF29	CAN2 发送缓冲区寄存器	7EF36DH	CAN2_TBUF29[7:0]								0000,0000
CAN2_TBUF30	CAN2 发送缓冲区寄存器	7EF36EH	CAN2_TBUF30[7:0]								0000,0000
CAN2_TBUF31	CAN2 发送缓冲区寄存器	7EF36FH	CAN2_TBUF31[7:0]								0000,0000
CAN2_TBUF32	CAN2 发送缓冲区寄存器	7EF370H	CAN2_TBUF32[7:0]								0000,0000
CAN2_TBUF33	CAN2 发送缓冲区寄存器	7EF371H	CAN2_TBUF33[7:0]								0000,0000
CAN2_TBUF34	CAN2 发送缓冲区寄存器	7EF372H	CAN2_TBUF34[7:0]								0000,0000
CAN2_TBUF35	CAN2 发送缓冲区寄存器	7EF373H	CAN2_TBUF35[7:0]								0000,0000
CAN2_TBUF36	CAN2 发送缓冲区寄存器	7EF374H	CAN2_TBUF36[7:0]								0000,0000
CAN2_TBUF37	CAN2 发送缓冲区寄存器	7EF375H	CAN2_TBUF37[7:0]								0000,0000
CAN2_TBUF38	CAN2 发送缓冲区寄存器	7EF376H	CAN2_TBUF38[7:0]								0000,0000
CAN2_TBUF39	CAN2 发送缓冲区寄存器	7EF377H	CAN2_TBUF39[7:0]								0000,0000
CAN2_TBUF40	CAN2 发送缓冲区寄存器	7EF378H	CAN2_TBUF40[7:0]								0000,0000
CAN2_TBUF41	CAN2 发送缓冲区寄存器	7EF379H	CAN2_TBUF41[7:0]								0000,0000
CAN2_TBUF42	CAN2 发送缓冲区寄存器	7EF37AH	CAN2_TBUF42[7:0]								0000,0000
CAN2_TBUF43	CAN2 发送缓冲区寄存器	7EF37BH	CAN2_TBUF43[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CAN2_TBUF44	CAN2 发送缓冲区寄存器	7EF37CH	CAN2_TBUF44[7:0]								0000,0000
CAN2_TBUF45	CAN2 发送缓冲区寄存器	7EF37DH	CAN2_TBUF45[7:0]								0000,0000
CAN2_TBUF46	CAN2 发送缓冲区寄存器	7EF37EH	CAN2_TBUF46[7:0]								0000,0000
CAN2_TBUF47	CAN2 发送缓冲区寄存器	7EF37FH	CAN2_TBUF47[7:0]								0000,0000
CAN2_TBUF48	CAN2 发送缓冲区寄存器	7EF380H	CAN2_TBUF48[7:0]								0000,0000
CAN2_TBUF49	CAN2 发送缓冲区寄存器	7EF381H	CAN2_TBUF49[7:0]								0000,0000
CAN2_TBUF50	CAN2 发送缓冲区寄存器	7EF382H	CAN2_TBUF50[7:0]								0000,0000
CAN2_TBUF51	CAN2 发送缓冲区寄存器	7EF383H	CAN2_TBUF51[7:0]								0000,0000
CAN2_TBUF52	CAN2 发送缓冲区寄存器	7EF384H	CAN2_TBUF52[7:0]								0000,0000
CAN2_TBUF53	CAN2 发送缓冲区寄存器	7EF385H	CAN2_TBUF53[7:0]								0000,0000
CAN2_TBUF54	CAN2 发送缓冲区寄存器	7EF386H	CAN2_TBUF54[7:0]								0000,0000
CAN2_TBUF55	CAN2 发送缓冲区寄存器	7EF387H	CAN2_TBUF55[7:0]								0000,0000
CAN2_TBUF56	CAN2 发送缓冲区寄存器	7EF388H	CAN2_TBUF56[7:0]								0000,0000
CAN2_TBUF57	CAN2 发送缓冲区寄存器	7EF389H	CAN2_TBUF57[7:0]								0000,0000
CAN2_TBUF58	CAN2 发送缓冲区寄存器	7EF38AH	CAN2_TBUF58[7:0]								0000,0000
CAN2_TBUF59	CAN2 发送缓冲区寄存器	7EF38BH	CAN2_TBUF59[7:0]								0000,0000
CAN2_TBUF60	CAN2 发送缓冲区寄存器	7EF38CH	CAN2_TBUF60[7:0]								0000,0000
CAN2_TBUF61	CAN2 发送缓冲区寄存器	7EF38DH	CAN2_TBUF61[7:0]								0000,0000
CAN2_TBUF62	CAN2 发送缓冲区寄存器	7EF38EH	CAN2_TBUF62[7:0]								0000,0000
CAN2_TBUF63	CAN2 发送缓冲区寄存器	7EF38FH	CAN2_TBUF63[7:0]								0000,0000
CAN2_TBUF64	CAN2 发送缓冲区寄存器	7EF390H	CAN2_TBUF64[7:0]								0000,0000
CAN2_TBUF65	CAN2 发送缓冲区寄存器	7EF391H	CAN2_TBUF65[7:0]								0000,0000
CAN2_TBUF66	CAN2 发送缓冲区寄存器	7EF392H	CAN2_TBUF66[7:0]								0000,0000
CAN2_TBUF67	CAN2 发送缓冲区寄存器	7EF393H	CAN2_TBUF67[7:0]								0000,0000
CAN2_TBUF68	CAN2 发送缓冲区寄存器	7EF394H	CAN2_TBUF68[7:0]								0000,0000
CAN2_TBUF69	CAN2 发送缓冲区寄存器	7EF395H	CAN2_TBUF69[7:0]								0000,0000
CAN2_TBUF70	CAN2 发送缓冲区寄存器	7EF396H	CAN2_TBUF70[7:0]								0000,0000
CAN2_TBUF71	CAN2 发送缓冲区寄存器	7EF397H	CAN2_TBUF71[7:0]								0000,0000
CAN2_TTS00	CAN2 传输时间戳	7EF398H	CAN2_TTS00[7:0]								0000,0000
CAN2_TTS01	CAN2 传输时间戳	7EF399H	CAN2_TTS01[7:0]								0000,0000
CAN2_TTS02	CAN2 传输时间戳	7EF39AH	CAN2_TTS02[7:0]								0000,0000
CAN2_TTS03	CAN2 传输时间戳	7EF39BH	CAN2_TTS03[7:0]								0000,0000
CAN2_TTS04	CAN2 传输时间戳	7EF39CH	CAN2_TTS04[7:0]								0000,0000
CAN2_TTS05	CAN2 传输时间戳	7EF39DH	CAN2_TTS05[7:0]								0000,0000
CAN2_TTS06	CAN2 传输时间戳	7EF39EH	CAN2_TTS06[7:0]								0000,0000
CAN2_TTS07	CAN2 传输时间戳	7EF39FH	CAN2_TTS07[7:0]								0000,0000
CAN2_CFG_STAT	CAN2 配置状态寄存器	7EF3A0H	RESET	LBME	LBMI	TPSS	TSSS	RACTIVE	TACTIVE	BUSOFF	1000,0000
CAN2_TCMD	CAN2 命令寄存器	7EF3A1H	TBSEL	LOM	-	TPE	TPA	TSONE	TSALL	TSA	0000,0000
CAN2_TCTRL	CAN2 发送控制寄存器	7EF3A2H	FD_ISO	TSNEXT	TSMODE	TTTBM	-	-	TSSTAT[1:0]		1001,0000
CAN2_RCTRL	CAN2 接收控制寄存器	7EF3A3H	SACK	ROM	ROV	RREL	RBALL	-	RSTAT[1:0]		0000,0000
CAN2_RTIE	CAN2 中断使能寄存器	7EF3A4H	RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	TSFF	1111,1110
CAN2_RTIF	CAN2 中断状态寄存器	7EF3A5H	RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF	0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CAN2_ERRINT	CAN2 错误中断寄存器	7EF3A6H	EWARN	EPASS	EPIE	EPIF	ALIE	ALIF	BEIE	BEIF	0000,0000
CAN2_LIMIT	CAN2 警告限制寄存器	7EF3A7H	AFWL[3:0]				EWL[3:0]				0000,0000
CAN2_S_SEG1	CAN2 慢速位段 1 时间设定	7EF3A8H	S_SEG1[7:0]								0000,0000
CAN2_S_SEG2	CAN2 慢速位段 2 时间设定	7EF3A9H	-	S_SEG2[6:0]							0000,0000
CAN2_S_SJW	CAN2 慢速同步补偿时间设定	7EF3AAH	-	S_SJW[6:0]							0000,0000
CAN2_S_PRESC	CAN2 慢速预分频寄存器	7EF3ABH	S_PRESC[7:0]								0000,0000
CAN2_F_SEG1	CAN2 快速位段 1 时间设定	7EF3ACH	FS_SEG1[7:0]								0000,0000
CAN2_F_SEG2	CAN2 快速位段 2 时间设定	7EF3ADH	-	F_SEG2[6:0]							0000,0000
CAN2_F_SJW	CAN2 快速同步补偿时间设定	7EF3AEH	-	F_SJW[6:0]							0000,0000
CAN2_F_PRESC	CAN2 快速预分频寄存器	7EF3AFH	F_PRESC[7:0]								0000,0000
CAN2_EALCAP	CAN2 错误和仲裁丢失捕获	7EF3B0H	KOER[2:0]			ALC[4:0]					0000,0000
CAN2_TDC	CAN2 发送器延迟补偿	7EF3B1H	TDCEN	SSPOFF[6:0]							0000,0000
CAN2_RECNT	CAN2 接收错误计数器	7EF3B2H	RECNT[7:0]								0000,0000
CAN2_TECNT	CAN2 发送错误计数器	7EF3B3H	TECNT[7:0]								0000,0000
CAN2_ACFCTRL	CAN2 筛选器组控制寄存器	7EF3B4H	-	-	SELMASK	-	ACFADR[3:0]				0000,0000
CAN2_TIMECFG	CAN2 时间戳控制寄存器	7EF3B5H	-	-	-	-	-	-	TIMEPOS	TIMEEN	0000,0000
CAN2_ACF_EN0	CAN2 筛选器组使能寄存器	7EF3B6H	AE_7	AE_6	AE_5	AE_4	AE_3	AE_2	AE_1	AE_0	0000,0001
CAN2_ACF_EN1	CAN2 筛选器组使能寄存器	7EF3B7H	AE_15	AE_14	AE_13	AE_12	AE_11	AE_10	AE_9	AE_8	0000,0000
CAN2_ACF0	CAN2 筛选器 code 和 mask	7EF3B8H	ACODE[7:0]/AMASK[7:0]								0000,0000
CAN2_ACF1	CAN2 筛选器 code 和 mask	7EF3B9H	ACODE[15:8]/AMASK[15:8]								0000,0000
CAN2_ACF2	CAN2 筛选器 code 和 mask	7EF3BAH	ACODE[23:16]/AMASK[23:16]								0000,0000
CAN2_ACF3	CAN2 筛选器 code 和 mask	7EF3BBH	-	AIDEE	AIDE	ACODE[28:24]/AMASK[28:24]					0000,0000
CAN2_VER0	CAN2 版本信息	7EF3BCH	VER0[7:0]								0000,0000
CAN2_VER1	CAN2 版本信息	7EF3BDH	VER1[7:0]								0000,0000
CAN2_TBSLOT	CAN2 TB SLOT 指针寄存器	7EF3BEH	TBE	TBF	-	-	-	TBPTR[2:0]			0000,0000
CAN2_TTCFG	CAN2 时间触发配置寄存器	7EF3BFH	WTIE	WTIF	TEIF	TTIE	TTIF	T_PRESC[1:0]		TTEN	0000,0000
CAN2_REF_MSG0	CAN2 参考消息	7EF3C0H	REF_ID[7:0]								0000,0000
CAN2_REF_MSG1	CAN2 参考消息	7EF3C1H	REF_ID[15:8]								0000,0000
CAN2_REF_MSG2	CAN2 参考消息	7EF3C2H	REF_ID[23:16]								0000,0000
CAN2_REF_MSG3	CAN2 参考消息	7EF3C3H	REF_IDE	-	-	REF_ID[28:24]					0000,0000
CAN2_TRIG_CFG0	CAN2 触发配置寄存器	7EF3C4H	-	-	TTPTR[5:0]						0000,0000
CAN2_TRIG_CFG1	CAN2 触发配置寄存器	7EF3C5H	TEW[3:0]				-	TTYPE[2:0]			0000,0000
CAN2_TT_TRIG0	CAN2 触发时间寄存器	7EF3C6H	TT_TRIG[7:0]								0000,0000
CAN2_TT_TRIG1	CAN2 触发时间寄存器	7EF3C7H	TT_TRIG[15:8]								0000,0000
CAN2_TT_WTRIG0	CAN2 触发看门时间寄存器	7EF3C8H	TT_WTRIG[7:0]								0000,0000
CAN2_TT_WTRIG1	CAN2 触发看门时间寄存器	7EF3C9H	TT_WTRIG[15:8]								0000,0000

注：特殊功能寄存器初始值意义

0： 初始值为 0；

1： 初始值为 1；

n： 初始值与 ISP 下载时的硬件选项有关；

x： 不存在这个位，初始值不确定

14 I/O 口

产品线	最多 I/O 口数量
STC32G144K246 系列	**91**

STC32G144K246 系列单片机所有的 I/O 口均有 **4 种工作模式**：

【无弱上拉 准双向口】工作模式、推挽输出/强上拉、高阻输入(电流既不能流入也不能流出)、开漏模式。可使用软件对 I/O 口的工作模式进行各种工作模式的设置。

关于 I/O 的注意事项：

1、 STC32G144K246 系列单片机 I/O 口的【无弱上拉 准双向口】工作模式，和【标准 准双向口】的区别是，需要打开【芯片内部 / 管脚外部】的内部上拉电阻，或芯片外部接上拉电阻，这时就可以当做【标准 准双向口】使用，但 I/O 管脚外部如输入低电平，无法关闭此上拉电阻，【标准 准双向口】是可以关闭【标准 准双向口】内部的弱上拉。

2、 所有 I/O 口上电后的状态均为高阻输入状态，用户在使用 I/O 口前必须先设置 I/O 口模式

3、 芯片上电时如果不需要使用 USB 进行 ISP 下载，P3.0/P3.1/P3.2 这 3 个 I/O 口不能同时为低电平，否则会进入 USB 下载模式而无法运行用户代码

4、 芯片上电时，若 P3.0 和 P3.1 同时为低电平，系统程序会短时间将上电初始状态是高阻输入的 P3.2 口对应的内部上拉电阻打开，并延时 1ms，来准确读取上电时 P3.2 口的外部状态，如果读到 P3.2 外部状态是低电平，则进入等待 USB 下载模式；如果读到 P3.2 外部状态是高电平，则不进入 USB 下载模式，并立即将 P3.2 口的内部上拉电阻关闭

5、 当使用 P5.4 当作复位脚时，这个端口内部的 20K 上拉电阻会一直打开；但 P5.4 做普通 I/O 口时，基于这个 I/O 口与复位脚共享管脚的特殊考量，端口内部的 20K 上拉电阻依然会打开大约 6.5 毫秒时间，再自动关闭（当用户的电路设计需要使用 P5.4 口驱动外部电路时，请务必考虑上电瞬间会有 6.5 毫秒时间的高电平的问题）

6、 内部上拉电阻和外部上拉电阻的部分区别：

外部上拉电阻随系统电源从 0V 上升到 MCU_Vcc，到 3.3V 或 5V，这个时间是足够的长，系统上电复位后到初始化运行用户程序时，外部上拉电阻已将外部拉高到高电平。

内部上拉电阻是运行用户程序时才使能打开的，由于外部电路的分布电容效应，要等待 0.7RC 时间（R:电阻，C:电容）才能将外部拉高到 0.7MCU_Vcc 高电平。建议将需要打开的内部上拉电阻使能后，直接多等待一会，如等待 1ms 后程序才认为外部已被拉高。

7、 特别注意，有硬件 USB 的 MCU：

P3.2 口是高电平（是高阻输入 + 内部【50K@5V ~ 82K@3.3V】上拉电阻被打开）@系统程序区时芯片上电复位后，首先跑的是系统程序，而不是用户程序。

为了判断是否要等待硬件 USB 下载，后来软件为了支持硬件 USB 下载做了优化，
无论是否需要下载还是进行串口下载或者 USB 下载，
P3.2 口都是高电平（是 高阻输入 + 内部【50K@5V ~ 82K@3.3V】上拉电阻被打开），
在系统区 P3.2 的【50K@5V ~ 82K@3.3V】上拉电阻被打开了。

等从系统区复位到用户区后，P3.2 的内部上拉电阻才被复位关闭。

14.1 I/O 口相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P0	P0 端口	80H	P07	P06	P05	P04	P03	P02	P01	P00	1111,1111
P1	P1 端口	90H	P17	P16	P15	P14	P13	P12	P11	P10	1111,1111
P2	P2 端口	A0H	P27	P26	P25	P24	P23	P22	P21	P20	1111,1111
P3	P3 端口	B0H	P37	P36	P35	P34	P33	P32	P31	P30	1111,1111
P4	P4 端口	C0H	P47	P46	P45	P44	P43	P42	P41	P40	1111,1111
P5	P5 端口	C8H	P57	P56	P55	P54	P53	P52	P51	P50	1111,1111
P6	P6 端口	E8H	P67	P66	P65	P64	P63	P62	P61	P60	1111,1111
P7	P7 端口	F8H	P77	P76	P75	P74	P73	P72	P71	P70	1111,1111
P8OUT	P8 端口输出寄存器	7EF700H	P87OUT	P86OUT	P85OUT	P84OUT	P83OUT	P82OUT	P81OUT	P80OUT	1111,1111
P9OUT	P9 端口输出寄存器	7EF701H	P97OUT	P96OUT	P95OUT	P94OUT	P93OUT	P92OUT	P91OUT	P90OUT	1111,1111
PAOUT	PA 端口输出寄存器	7EF702H	PA7OUT	PA6OUT	PA5OUT	PA4OUT	PA3OUT	PA2OUT	PA1OUT	PA0OUT	1111,1111
PBOUT	PB 端口输出寄存器	7EF703H	PB7OUT	PB6OUT	PB5OUT	PB4OUT	PB3OUT	PB2OUT	PB1OUT	PB0OUT	1111,1111
P8IN	P8 端口输入寄存器	7EF708H	P87IN	P86IN	P85IN	P84IN	P83IN	P82IN	P81IN	P80IN	xxxx,xxxx
P9IN	P9 端口输入寄存器	7EF709H	P97IN	P96IN	P95IN	P94IN	P93IN	P92IN	P91IN	P90IN	xxxx,xxxx
PAIN	PA 端口输入寄存器	7EF70AH	PA7IN	PA6IN	PA5IN	PA4IN	PA3IN	PA2IN	PA1IN	PA0IN	xxxx,xxxx
PBIN	PB 端口输入寄存器	7EF70BH	PB7IN	PB6IN	PB5IN	PB4IN	PB3IN	PB2IN	PB1IN	PB0IN	xxxx,xxxx
P0M1	P0 口配置寄存器 1	93H	P0M1[7:0]								1111,1111
P0M0	P0 口配置寄存器 0	94H	P0M0[7:0]								0000,0000
P1M1	P1 口配置寄存器 1	91H	P1M1[7:0]								1111,1111
P1M0	P1 口配置寄存器 0	92H	P1M0[7:0]								0000,0000
P2M1	P2 口配置寄存器 1	95H	P2M1[7:0]								1111,1111
P2M0	P2 口配置寄存器 0	96H	P2M0[7:0]								0000,0000
P3M1	P3 口配置寄存器 1	B1H	P3M1[7:0]								1111,1100
P3M0	P3 口配置寄存器 0	B2H	P3M0[7:0]								0000,0000
P4M1	P4 口配置寄存器 1	B3H	P4M1[7:0]								1111,1111
P4M0	P4 口配置寄存器 0	B4H	P4M0[7:0]								0000,0000
P5M1	P5 口配置寄存器 1	C9H	P5M1[7:0]								1111,1111
P5M0	P5 口配置寄存器 0	CAH	P5M0[7:0]								xx00,0000
P6M1	P6 口配置寄存器 1	CBH	P6M1[7:0]								1111,1111
P6M0	P6 口配置寄存器 0	CCH	P6M0[7:0]								0000,0000
P7M1	P7 口配置寄存器 1	E1H	P7M1[7:0]								1111,1111
P7M0	P7 口配置寄存器 0	E2H	P7M0[7:0]								0000,0000
P8M1	P8 口配置寄存器 1	7EF718H	P8M1[7:0]								1111,1111
P8M0	P8 口配置寄存器 0	7EF710H	P8M0[7:0]								0000,0000
P9M1	P9 口配置寄存器 1	7EF719H	P9M1[7:0]								1111,1111
P9M0	P9 口配置寄存器 0	7EF711H	P9M0[7:0]								0000,0000
PAM1	PA 口配置寄存器 1	7EF71AH	PAM1[7:0]								1111,1111
PAM0	PA 口配置寄存器 0	7EF712H	PAM0[7:0]								0000,0000
PBM1	PB 口配置寄存器 1	7EF71BH	PBM1[7:0]								1111,1111
PBM0	PB 口配置寄存器 0	7EF713H	PBM0[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P8SETB	P8 端口置 1 位控制器	7EF720H	P87SETB	P86SETB	P85SETB	P84SETB	P83SETB	P82SETB	P81SETB	P80SETB	0000,0000
P9SETB	P9 端口置 1 位控制器	7EF721H	P97SETB	P96SETB	P95SETB	P94SETB	P93SETB	P92SETB	P91SETB	P90SETB	0000,0000
PASETB	PA 端口置 1 位控制器	7EF722H	PA7SETB	PA6SETB	PA5SETB	PA4SETB	PA3SETB	PA2SETB	PA1SETB	PA0SETB	0000,0000
PBSETB	PB 端口置 1 位控制器	7EF723H	PB7SETB	PB6SETB	PB5SETB	PB4SETB	PB3SETB	PB2SETB	PB1SETB	PB0SETB	0000,0000
P8CLRB	P8 端口清 0 位控制器	7EF728H	P87CLRB	P86CLRB	P85CLRB	P84CLRB	P83CLRB	P82CLRB	P81CLRB	P80CLRB	0000,0000
P9CLRB	P9 端口清 0 位控制器	7EF729H	P97CLRB	P96CLRB	P95CLRB	P94CLRB	P93CLRB	P92CLRB	P91CLRB	P90CLRB	0000,0000
PACLRB	PA 端口清 0 位控制器	7EF72AH	PA7CLRB	PA6CLRB	PA5CLRB	PA4CLRB	PA3CLRB	PA2CLRB	PA1CLRB	PA0CLRB	0000,0000
PBCLRB	PB 端口清 0 位控制器	7EF72BH	PB7CLRB	PB6CLRB	PB5CLRB	PB4CLRB	PB3CLRB	PB2CLRB	PB1CLRB	PB0CLRB	0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P0PU	P0 口上拉电阻控制寄存器	7EFE10H	P0PU[7:0]								0000,0000
P1PU	P1 口上拉电阻控制寄存器	7EFE11H	P1PU[7:0]								0000,0000
P2PU	P2 口上拉电阻控制寄存器	7EFE12H	P2PU[7:0]								0000,0000
P3PU	P3 口上拉电阻控制寄存器	7EFE13H	P3PU[7:0]								0000,0000
P4PU	P4 口上拉电阻控制寄存器	7EFE14H	P4PU[7:0]								0000,0000
P5PU	P5 口上拉电阻控制寄存器	7EFE15H	P5PU[7:0]								0000,0000
P6PU	P6 口上拉电阻控制寄存器	7EFE16H	P6PU[7:0]								0000,0000
P7PU	P7 口上拉电阻控制寄存器	7EFE17H	P7PU[7:0]								0000,0000
P8PU	P8 口上拉电阻控制寄存器	7EF9C0H	P8PU[7:0]								0000,0000
P9PU	P9 口上拉电阻控制寄存器	7EF9C1H	P9PU[7:0]								0000,0000
PAPU	PA 口上拉电阻控制寄存器	7EF9C2H	PAPU[7:0]								0000,0000
PBPU	PB 口上拉电阻控制寄存器	7EF9C3H	PBPU[7:0]								0000,0000
P0NCS	P0 口施密特触发控制寄存器	7EFE18H	P0NCS[7:0]								0000,0000
P1NCS	P1 口施密特触发控制寄存器	7EFE19H	P1NCS[7:0]								0000,0000
P2NCS	P2 口施密特触发控制寄存器	7EFE1AH	P2NCS[7:0]								0000,0000
P3NCS	P3 口施密特触发控制寄存器	7EFE1BH	P3NCS[7:0]								0000,0000
P4NCS	P4 口施密特触发控制寄存器	7EFE1CH	P4NCS[7:0]								0000,0000
P5NCS	P5 口施密特触发控制寄存器	7EFE1DH	P5NCS[7:0]								0000,0000
P6NCS	P6 口施密特触发控制寄存器	7EFE1EH	P6NCS[7:0]								0000,0000
P7NCS	P7 口施密特触发控制寄存器	7EFE1FH	P7NCS[7:0]								0000,0000
P8NCS	P8 口施密特触发控制寄存器	7EF9C8H	P8NCS[7:0]								0000,0000
P9NCS	P9 口施密特触发控制寄存器	7EF9C9H	P9NCS[7:0]								0000,0000
PANCS	PA 口施密特触发控制寄存器	7EF9CAH	PANCS[7:0]								0000,0000
PBNCS	PB 口施密特触发控制寄存器	7EF9CBH	PBNCS[7:0]								0000,0000
P0SR	P0 口电平转换速率寄存器	7EFE20H	P0SR[7:0]								1111,1111
P1SR	P1 口电平转换速率寄存器	7EFE21H	P1SR[7:0]								1111,1111
P2SR	P2 口电平转换速率寄存器	7EFE22H	P2SR[7:0]								1111,1111
P3SR	P3 口电平转换速率寄存器	7EFE23H	P3SR[7:0]								1111,1111
P4SR	P4 口电平转换速率寄存器	7EFE24H	P4SR[7:0]								1111,1111
P5SR	P5 口电平转换速率寄存器	7EFE25H	P5SR[7:0]								1111,1111

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P6SR	P6 口电平转换速率寄存器	7EFE26H	P6SR[7:0]								1111,1111
P7SR	P7 口电平转换速率寄存器	7EFE27H	P7SR[7:0]								1111,1111
P8SR	P8 口电平转换速率寄存器	7EF9D0H	P8SR[7:0]								1111,1111
P9SR	P9 口电平转换速率寄存器	7EF9D1H	P9SR[7:0]								1111,1111
PASR	PA 口电平转换速率寄存器	7EF9D2H	PASR[7:0]								1111,1111
PBSR	PB 口电平转换速率寄存器	7EF9D3H	PBSR[7:0]								1111,1111
P0DR	P0 口驱动电流控制寄存器	7EFE28H	P0DR[7:0]								1111,1111
P1DR	P1 口驱动电流控制寄存器	7EFE29H	P1DR[7:0]								1111,1111
P2DR	P2 口驱动电流控制寄存器	7EFE2AH	P2DR[7:0]								1111,1111
P3DR	P3 口驱动电流控制寄存器	7EFE2BH	P3DR[7:0]								1111,1111
P4DR	P4 口驱动电流控制寄存器	7EFE2CH	P4DR[7:0]								1111,1111
P5DR	P5 口驱动电流控制寄存器	7EFE2DH	P5DR[7:0]								1111,1111
P6DR	P6 口驱动电流控制寄存器	7EFE2EH	P6DR[7:0]								1111,1111
P7DR	P7 口驱动电流控制寄存器	7EFE2FH	P7DR[7:0]								1111,1111
P8DR	P8 口驱动电流控制寄存器	7EF9D8H	P8DR[7:0]								1111,1111
P9DR	P9 口驱动电流控制寄存器	7EF9D9H	P9DR[7:0]								1111,1111
PADR	PA 口驱动电流控制寄存器	7EF9DAH	PADR[7:0]								1111,1111
PBDR	PB 口驱动电流控制寄存器	7EF9DBH	PBDR[7:0]								1111,1111
P0IE	P0 口输入使能控制寄存器	7EFE30H	P0IE[7:0]								1111,1111
P1IE	P1 口输入使能控制寄存器	7EFE31H	P1IE[7:0]								1111,1111
P2IE	P2 口输入使能控制寄存器	7EFE32H	P2IE[7:0]								1111,1111
P3IE	P3 口输入使能控制寄存器	7EFE33H	P3IE[7:0]								1111,1111
P4IE	P4 口输入使能控制寄存器	7EFE34H	P4IE[7:0]								1111,1111
P5IE	P5 口输入使能控制寄存器	7EFE35H	P5IE[7:0]								1111,1111
P6IE	P6 口输入使能控制寄存器	7EFE36H	P6IE[7:0]								1111,1111
P7IE	P7 口输入使能控制寄存器	7EFE37H	P7IE[7:0]								1111,1111
P8IE	P8 口输入使能控制寄存器	7EF9E0H	P8IE[7:0]								1111,1111
P9IE	P9 口输入使能控制寄存器	7EF9E1H	P9IE[7:0]								1111,1111
PAIE	PA 口输入使能控制寄存器	7EF9E2H	PAIE[7:0]								1111,1111
PBIE	PB 口输入使能控制寄存器	7EF9E3H	PBIE[7:0]								1111,1111
P0PD	P0 口下拉电阻控制寄存器	7EFE40H	P0PD[7:0]								0000,0000
P1PD	P1 口下拉电阻控制寄存器	7EFE41H	P1PD[7:0]								0000,0000
P2PD	P2 口下拉电阻控制寄存器	7EFE42H	P2PD[7:0]								0000,0000
P3PD	P3 口下拉电阻控制寄存器	7EFE43H	P3PD[7:0]								0000,0000
P4PD	P4 口下拉电阻控制寄存器	7EFE44H	P4PD[7:0]								0000,0000
P5PD	P5 口下拉电阻控制寄存器	7EFE45H	P5PD[7:0]								0000,0000
P6PD	P6 口下拉电阻控制寄存器	7EFE46H	P6PD[7:0]								0000,0000
P7PD	P7 口下拉电阻控制寄存器	7EFE47H	P7PD[7:0]								0000,0000
P8PD	P8 口下拉电阻控制寄存器	7EF9F0H	P8PD[7:0]								0000,0000
P9PD	P9 口下拉电阻控制寄存器	7EF9F1H	P9PD[7:0]								0000,0000
PAPD	PA 口下拉电阻控制寄存器	7EF9F2H	PAPD[7:0]								0000,0000
PBPD	PB 口下拉电阻控制寄存器	7EF9F3H	PBPD[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P0BP	P0 口模式用户控制寄存器	7EFE48H	P0BP[7:0]								1111,1111
P1BP	P1 口模式用户控制寄存器	7EFE49H	P1BP[7:0]								1111,1111
P2BP	P2 口模式用户控制寄存器	7EFE4AH	P2BP[7:0]								1111,1111
P3BP	P3 口模式用户控制寄存器	7EFE4BH	P3BP[7:0]								1111,1111
P4BP	P4 口模式用户控制寄存器	7EFE4CH	P4BP[7:0]								1111,1111
P5BP	P5 口模式用户控制寄存器	7EFE4DH	P5BP[7:0]								1111,1111
P6BP	P6 口模式用户控制寄存器	7EFE4EH	P6BP[7:0]								1111,1111
P7BP	P7 口模式用户控制寄存器	7EFE4FH	P7BP[7:0]								1111,1111
P8BP	P8 口模式用户控制寄存器	7EF9F8H	P8BP[7:0]								1111,1111
P9BP	P9 口模式用户控制寄存器	7EF9F9H	P9BP[7:0]								1111,1111
PABP	PA 口模式用户控制寄存器	7EF9FAH	PABP[7:0]								1111,1111
PBBP	PB 口模式用户控制寄存器	7EF9FBH	PBBP[7:0]								1111,1111

14.1.1 端口数据寄存器（Pn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0	80H	P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0
P1	90H	P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0
P2	A0H	P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0
P3	B0H	P3.7	P3.6	P3.5	P3.4	P3.3	P3.2	P3.1	P3.0
P4	C0H	P4.7	P4.6	P4.5	P4.4	P4.3	P4.2	P4.1	P4.0
P5	C8H	P5.7	P5.6	P5.5	P5.4	P5.3	P5.2	P5.1	P5.0
P6	E8H	P6.7	P6.6	P6.5	P6.4	P6.3	P6.2	P6.1	P6.0
P7	F8H	P7.7	P7.6	P7.5	P7.4	P7.3	P7.2	P7.1	P7.0

读写端口状态

写 0：输出低电平到端口缓冲区

写 1：输出高电平到端口缓冲区

读：直接读端口管脚上的电平

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P8OUT	7EF700H	P87OUT	P86OUT	P85OUT	P84OUT	P83OUT	P82OUT	P81OUT	P80OUT
P9OUT	7EF701H	P97OUT	P96OUT	P95OUT	P94OUT	P93OUT	P92OUT	P91OUT	P90OUT
PAOUT	7EF702H	PA7OUT	PA6OUT	PA5OUT	PA4OUT	PA3OUT	PA2OUT	PA1OUT	PA0OUT
PBOUT	7EF703H	PB7OUT	PB6OUT	PB5OUT	PB4OUT	PB3OUT	PB2OUT	PB1OUT	PB0OUT
P8IN	7EF708H	P87IN	P86IN	P85IN	P84IN	P83IN	P82IN	P81IN	P80IN
P9IN	7EF709H	P97IN	P96IN	P95IN	P94IN	P93IN	P92IN	P91IN	P90IN
PAIN	7EF70AH	PA7IN	PA6IN	PA5IN	PA4IN	PA3IN	PA2IN	PA1IN	PA0IN
PBIN	7EF70BH	PB7IN	PB6IN	PB5IN	PB4IN	PB3IN	PB2IN	PB1IN	PB0IN

PnOUT：写端口状态

写 0：输出低电平到端口缓冲区

写 1：输出高电平到端口缓冲区

PnIN：直接读端口管脚上的电平

【说明】：STC32G144K246的P8/P9/PA/PB在XFR区不可位寻址，后续芯片改版，会将P8/P9/PA/PB改到SFR区，P8口在E4H、P9口在E5H、PA口在E6H、PB口在E7H，便可对P8/P9/PA/PB进行位寻址。

14.1.2 端口模式配置寄存器（PnM0，PnM1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0M0	94H	P07M0	P06M0	P05M0	P04M0	P03M0	P02M0	P01M0	P00M0
P0M1	93H	P07M1	P06M1	P05M1	P04M1	P03M1	P02M1	P01M1	P00M1
P1M0	92H	P17M0	P16M0	P15M0	P14M0	P13M0	P12M0	P11M0	P10M0
P1M1	91H	P17M1	P16M1	P15M1	P14M1	P13M1	P12M1	P11M1	P10M1
P2M0	96H	P27M0	P26M0	P25M0	P24M0	P23M0	P22M0	P21M0	P20M0
P2M1	95H	P27M1	P26M1	P25M1	P24M1	P23M1	P22M1	P21M1	P20M1
P3M0	B2H	P37M0	P36M0	P35M0	P34M0	P33M0	P32M0	P31M0	P30M0
P3M1	B1H	P37M1	P36M1	P35M1	P34M1	P33M1	P32M1	P31M1	P30M1
P4M0	B4H	P47M0	P46M0	P45M0	P44M0	P43M0	P42M0	P41M0	P40M0
P4M1	B3H	P47M1	P46M1	P45M1	P44M1	P43M1	P42M1	P41M1	P40M1
P5M0	CAH	P57M0	P56M0	P55M0	P54M0	P53M0	P52M0	P51M0	P50M0
P5M1	C9H	P57M1	P56M1	P55M1	P54M1	P53M1	P52M1	P51M1	P50M1
P6M0	CCH	P67M0	P66M0	P65M0	P64M0	P63M0	P62M0	P61M0	P60M0
P6M1	CBH	P67M1	P66M1	P65M1	P64M1	P63M1	P62M1	P61M1	P60M1
P7M0	E2H	P77M0	P76M0	P75M0	P74M0	P73M0	P72M0	P71M0	P70M0
P7M1	E1H	P77M1	P76M1	P75M1	P74M1	P73M1	P72M1	P71M1	P70M1
P8M0	7EF710H	P87M0	P86M0	P85M0	P84M0	P83M0	P82M0	P81M0	P80M0
P8M1	7EF718H	P87M1	P86M1	P85M1	P84M1	P83M1	P82M1	P81M1	P80M1
P9M0	7EF711H	P97M0	P96M0	P95M0	P94M0	P93M0	P92M0	P91M0	P90M0
P9M1	7EF719H	P97M1	P96M1	P95M1	P94M1	P93M1	P92M1	P91M1	P90M1
PAM0	7EF712H	PA7M0	PA6M0	PA5M0	PA4M0	PA3M0	PA2M0	PA1M0	PA0M0
PAM1	7EF71AH	PA7M1	PA6M1	PA5M1	PA4M1	PA3M1	PA2M1	PA1M1	PA0M1
PBM0	7EF713H	PB7M0	PB6M0	PB5M0	PB4M0	PB3M0	PB2M0	PB1M0	PB0M0
PBM1	7EF71BH	PB7M1	PB6M1	PB5M1	PB4M1	PB3M1	PB2M1	PB1M1	PB0M1

配置端口的模式

PnM1.x	PnM0.x	Pn.x 口工作模式
0	0	【无弱上拉 准双向口】： 输出低时，灌电流可达20mA； 输出由0变1时，有2个时钟的强推挽输出，提高I/O速度，2个时钟后，强推挽自动关闭，需提前打开内部上拉电阻或外接上拉电阻。
0	1	推挽输出
1	0	高阻输入
1	1	开漏模式

14.1.3 端口上拉电阻控制寄存器（PnPU）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0PU	7EFE10H	P07PU	P06PU	P05PU	P04PU	P03PU	P02PU	P01PU	P00PU
P1PU	7EFE11H	P17PU	P16PU	P15PU	P14PU	P13PU	P12PU	P11PU	P10PU
P2PU	7EFE12H	P27PU	P26PU	P25PU	P24PU	P23PU	P22PU	P21PU	P20PU
P3PU	7EFE13H	P37PU	P36PU	P35PU	P34PU	P33PU	P32PU	P31PU	P30PU
P4PU	7EFE14H	P47PU	P46PU	P45PU	P44PU	P43PU	P42PU	P41PU	P40PU
P5PU	7EFE15H	P57PU	P56PU	P55PU	P54PU	P53PU	P52PU	P51PU	P50PU
P6PU	7EFE16H	P67PU	P66PU	P65PU	P64PU	P63PU	P62PU	P61PU	P60PU
P7PU	7EFE17H	P77PU	P76PU	P75PU	P74PU	P73PU	P72PU	P71PU	P70PU
P8PU	7EF9C0H	P87PU	P86PU	P85PU	P84PU	P83PU	P82PU	P81PU	P80PU
P9PU	7EF9C1H	P97PU	P96PU	P95PU	P94PU	P93PU	P92PU	P91PU	P90PU
PAPU	7EF9C2H	PA7PU	PA6PU	PA5PU	PA4PU	PA3PU	PA2PU	PA1PU	PA0PU
PBPU	7EF9C3H	PB7PU	PB6PU	PB5PU	PB4PU	PB3PU	PB2PU	PB1PU	PB0PU

端口内部上拉电阻控制位（常温下，上拉电阻阻值范围：52.7K@5V，80.9K@3.3V）

0：禁止端口内部的上拉电阻

1：使能端口内部的上拉电阻

14.1.4 端口施密特触发控制寄存器（PnNCS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0NCS	7EFE18H	P07NCS	P06NCS	P05NCS	P04NCS	P03NCS	P02NCS	P01NCS	P00NCS
P1NCS	7EFE19H	P17NCS	P16NCS	P15NCS	P14NCS	P13NCS	P12NCS	P11NCS	P10NCS
P2NCS	7EFE1AH	P27NCS	P26NCS	P25NCS	P24NCS	P23NCS	P22NCS	P21NCS	P20NCS
P3NCS	7EFE1BH	P37NCS	P36NCS	P35NCS	P34NCS	P33NCS	P32NCS	P31NCS	P30NCS
P4NCS	7EFE1CH	P47NCS	P46NCS	P45NCS	P44NCS	P43NCS	P42NCS	P41NCS	P40NCS
P5NCS	7EFE1DH	P57NCS	P56NCS	P55NCS	P54NCS	P53NCS	P52NCS	P51NCS	P50NCS
P6NCS	7EFE1EH	P67NCS	P66NCS	P65NCS	P64NCS	P63NCS	P62NCS	P61NCS	P60NCS
P7NCS	7EFE1FH	P77NCS	P76NCS	P75NCS	P74NCS	P73NCS	P72NCS	P71NCS	P70NCS
P8NCS	7EF9C8H	P87NCS	P86NCS	P85NCS	P84NCS	P83NCS	P82NCS	P81NCS	P80NCS
P9NCS	7EF9C9H	P97NCS	P96NCS	P95NCS	P94NCS	P93NCS	P92NCS	P91NCS	P90NCS
PANCS	7EF9CAH	PA7NCS	PA6NCS	PA5NCS	PA4NCS	PA3NCS	PA2NCS	PA1NCS	PB0NCS
PBNCS	7EF9CBH	PB7NCS	PB6NCS	PB5NCS	PB4NCS	PB3NCS	PB2NCS	PB1NCS	PB0NCS

端口施密特触发控制位

0：使能端口的施密特触发功能。（上电复位后默认使能施密特触发）

1：禁止端口的施密特触发功能。

14.1.5 端口电平转换速度控制寄存器（PnSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0SR	7EFE20H	P07SR	P06SR	P05SR	P04SR	P03SR	P02SR	P01SR	P00SR
P1SR	7EFE21H	P17SR	P16SR	P15SR	P14SR	P13SR	P12SR	P11SR	P10SR
P2SR	7EFE22H	P27SR	P26SR	P25SR	P24SR	P23SR	P22SR	P21SR	P20SR
P3SR	7EFE23H	P37SR	P36SR	P35SR	P34SR	P33SR	P32SR	P31SR	P30SR
P4SR	7EFE24H	P47SR	P46SR	P45SR	P44SR	P43SR	P42SR	P41SR	P40SR
P5SR	7EFE25H	P57SR	P56SR	P55SR	P54SR	P53SR	P52SR	P51SR	P50SR
P6SR	7EFE26H	P67SR	P66SR	P65SR	P64SR	P63SR	P62SR	P61SR	P60SR
P7SR	7EFE27H	P77SR	P76SR	P75SR	P74SR	P73SR	P72SR	P71SR	P70SR
P8SR	7EF9D0H	P87SR	P86SR	P85SR	P84SR	P83SR	P82SR	P81SR	P80SR
P9SR	7EF9D1H	P97SR	P96SR	P95SR	P94SR	P93SR	P92SR	P91SR	P90SR
PASR	7EF9D2H	PA7SR	PA6SR	PA5SR	PA4SR	PA3SR	PA2SR	PA1SR	PA0SR
PBSR	7EF9D3H	PB7SR	PB6SR	PB5SR	PB4SR	PB3SR	PB2SR	PB1SR	PB0SR

控制端口电平转换的速度

0：电平转换速度快，相应的上下冲会比较大

1：电平转换速度慢，相应的上下冲比较小

14.1.6 端口驱动电流控制寄存器（PnDR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0DR	7EFE28H	P07DR	P06DR	P05DR	P04DR	P03DR	P02DR	P01DR	P00DR
P1DR	7EFE29H	P17DR	P16DR	P15DR	P14DR	P13DR	P12DR	P11DR	P10DR
P2DR	7EFE2AH	P27DR	P26DR	P25DR	P24DR	P23DR	P22DR	P21DR	P20DR
P3DR	7EFE2BH	P37DR	P36DR	P35DR	P34DR	P33DR	P32DR	P31DR	P30DR
P4DR	7EFE2CH	P47DR	P46DR	P45DR	P44DR	P43DR	P42DR	P41DR	P40DR
P5DR	7EFE2DH	P57DR	P56DR	P55DR	P54DR	P53DR	P52DR	P51DR	P50DR
P6DR	7EFE2EH	P67DR	P66DR	P65DR	P64DR	P63DR	P62DR	P61DR	P60DR
P7DR	7EFE2FH	P77DR	P76DR	P75DR	P74DR	P73DR	P72DR	P71DR	P70DR
P8DR	7EF9D8H	P87DR	P86DR	P85DR	P84DR	P83DR	P82DR	P81DR	P80DR
P9DR	7EF9D9H	P97DR	P96DR	P95DR	P94DR	P93DR	P92DR	P91DR	P90DR
PADR	7EF9DAH	PA7DR	PA6DR	PA5DR	PA4DR	PA3DR	PA2DR	PA1DR	PA0DR
PBDR	7EF9DBH	PB7DR	PB6DR	PB5DR	PB4DR	PB3DR	PB2DR	PB1DR	PB0DR

控制端口的驱动能力

0：增强驱动能力

1：一般驱动能力

14.1.7 端口数字信号输入使能控制寄存器（PnIE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0IE	7EFE30H	P07IE	P06IE	P05IE	P04IE	P03IE	P02IE	P01IE	P00IE
P1IE	7EFE31H	P17IE	P16IE	P15IE	P14IE	P13IE	P12IE	P11IE	P10IE
P2IE	7EFE32H	P27IE	P26IE	P25IE	P24IE	P23IE	P22IE	P21IE	P20IE
P3IE	7EFE33H	P37IE	P36IE	P35IE	P34IE	P33IE	P32IE	P31IE	P30IE
P4IE	7EFE34H	P47IE	P46IE	P45IE	P44IE	P43IE	P42IE	P41IE	P40IE
P5IE	7EFE35H	P57IE	P56IE	P55IE	P54IE	P53IE	P52IE	P51IE	P50IE
P6IE	7EFE36H	P67IE	P66IE	P65IE	P64IE	P63IE	P62IE	P61IE	P60IE
P7IE	7EFE37H	P77IE	P76IE	P75IE	P74IE	P73IE	P72IE	P71IE	P70IE
P8IE	7EF9E0H	P87IE	P86IE	P85IE	P84IE	P83IE	P82IE	P81IE	P80IE
P9IE	7EF9E1H	P97IE	P96IE	P95IE	P94IE	P93IE	P92IE	P91IE	P90IE
PAIE	7EF9E2H	PA7IE	PA6IE	PA5IE	PA4IE	PA3IE	PA2IE	PA1IE	PA0IE
PBIE	7EF9E3H	PB7IE	PB6IE	PB5IE	PB4IE	PB3IE	PB2IE	PB1IE	PB0IE

数字信号输入使能控制

0：禁止数字信号输入。若 I/O 被当作比较器输入口、ADC 输入口或者触摸按键输入口等模拟口时，进入主时钟停振/省电模式前，必须设置为 0，否则会有额外的耗电。

1：使能数字信号输入。若 I/O 被当作数字口时，必须设置为 1，否 MCU 无法读取外部端口的电平。

14.1.8 端口下拉电阻控制寄存器（PnPD）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0PD	7EFE40H	P07PD	P06PD	P05PD	P04PD	P03PD	P02PD	P01PD	P00PD
P1PD	7EFE41H	P17PD	P16PD	P15PD	P14PD	P13PD	P12PD	P11PD	P10PD
P2PD	7EFE42H	P27PD	P26PD	P25PD	P24PD	P23PD	P22PD	P21PD	P20PD
P3PD	7EFE43H	P37PD	P36PD	P35PD	P34PD	P33PD	P32PD	P31PD	P30PD
P4PD	7EFE44H	P47PD	P46PD	P45PD	P44PD	P43PD	P42PD	P41PD	P40PD
P5PD	7EFE45H	P57PD	P56PD	P55PD	P54PD	P53PD	P52PD	P51PD	P50PD
P6PD	7EFE46H	P67PD	P66PD	P65PD	P64PD	P63PD	P62PD	P61PD	P60PD
P7PD	7EFE47H	P77PD	P76PD	P75PD	P74PD	P73PD	P72PD	P71PD	P70PD
P8PD	7EF9F0H	P87PD	P86PD	P85PD	P84PD	P83PD	P82PD	P81PD	P80PD
P9PD	7EF9F1H	P97PD	P96PD	P95PD	P94PD	P93PD	P92PD	P91PD	P90PD
PAPD	7EF9F2H	PA7PD	PA6PD	PA5PD	PA4PD	PA3PD	PA2PD	PA1PD	PA0PD
PBPD	7EF9F3H	PB7PD	PB6PD	PB5PD	PB4PD	PB3PD	PB2PD	PB1PD	PB0PD

端口内部下拉电阻控制位（常温下，下拉电阻阻值范围：46.2K@5V，33.9K@3.3V）

0：禁止端口内部的下拉电阻

1：使能端口内部的下拉电阻

14.1.9 端口模式用户控制寄存器（PnBP）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0BP	7EFE48H	P0BP[7:0]							
P1BP	7EFE49H	P1BP[7:0]							
P2BP	7EFE4AH	P2BP[7:0]							
P3BP	7EFE4BH	P3BP[7:0]							
P4BP	7EFE4CH	P4BP[7:0]							
P5BP	7EFE4DH	P5BP[7:0]							
P6BP	7EFE4EH	P6BP[7:0]							
P7BP	7EFE4FH	P7BP[7:0]							
P8BP	7EF9F8H	P8BP[7:0]							
P9BP	7EF9F9H	P9BP[7:0]							
PABP	7EF9FAH	PABP[7:0]							
PBBP	7EF9FBH	PBBP[7:0]							

端口模式用户控制位

0：外设模块对所用 I/O 的模式进行自动配置，忽略用户使用 PxM0/PxM1 对相应 I/O 进行的配置。

1：外设模块对所用 I/O 的模式不自动配置，需要用户使用 PxM0/PxM1 寄存器对 I/O 进行配置。

14.1.10 P5.4 和 P1.2 脚功能切换寄存器（P_SWX1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX1	7EFD69H	S4_S[1]	S3_S[1]	S2_S[1]	-	CAN2_S[2]	CAN_S[2]	-	**SWP54P12**

SWP54P12：P5.4 和 P1.2 管脚数字功能切换

0：P5.4 外部端口的数组功能为 P5.4 口相关的数字功能
（S1SS_3/S2SS_3/RST/MCLKO/SS_3/PWMB6_2/QSPI_NCS_2）

1：P5.4 外部端口的数组功能为 P1.2 口相关的数字功能
（ADC_ETR/ADCB_ETR/T2/S1SS/S2SS/RxD7_4/SS/PWMA2P/CAN_STB_6/QSPI_NCS）

注 1： SWP54P12 只能切换 P1.2 口的数字功能到 P5.4 口，P1.2 上的 ADC2 无法进行切换

注 2： 由于芯片封装时已在内部将 P5.4 口和 P1.2 口共同打线到外部 P5.4 的端口上，所以 P5.4 口和 P1.2 口上的功能均可以之间使用，不需要使用 SWP54P12 进行切换

14.2 配置 I/O 口

每个 I/O 的配置都需要使用两个寄存器进行设置。

以 P0 口为例，配置 P0 口需要使用 P0M0 和 P0M1 两个寄存器进行配置，如下图所示：

即 P0M0 的第 0 位和 P0M1 的第 0 位组合起来配置 P0.0 口的模式
即 P0M0 的第 1 位和 P0M1 的第 1 位组合起来配置 P0.1 口的模式
其他所有 I/O 的配置都与此类似。

PnM0 与 PnM1 的组合方式如下表所示

PnM1	PnM0	I/O 口工作模式
0	0	【无弱上拉 准双向口】/无弱上拉： 输出低时，灌电流可达20mA； 输出由0变1时，有2个时钟的强推挽输出，提高I/O速度，2个时钟后，强推挽自动关闭，需提前打开内部上拉电阻或外接上拉电阻。
0	1	推挽输出（强上拉输出，可达20mA，要加限流电阻）
1	0	高阻输入（电流既不能流入也不能流出）
1	1	开漏模式（Open-Drain），内部上拉电阻断开 开漏模式既可读外部状态也可对外输出（高电平或低电平）。如要正确读外部状态或需要对外输出高电平，需外加上拉电阻，否则读不到外部状态，也对外输不出高电平。 **===【开漏工作模式】，对外设置输出为 1，等同于 【高阻输入】** **===【开漏工作模式】，【打开内部上拉电阻 | 或外部加上拉电阻】，简单等同于 【准双向口】**

注：n = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B

注意：

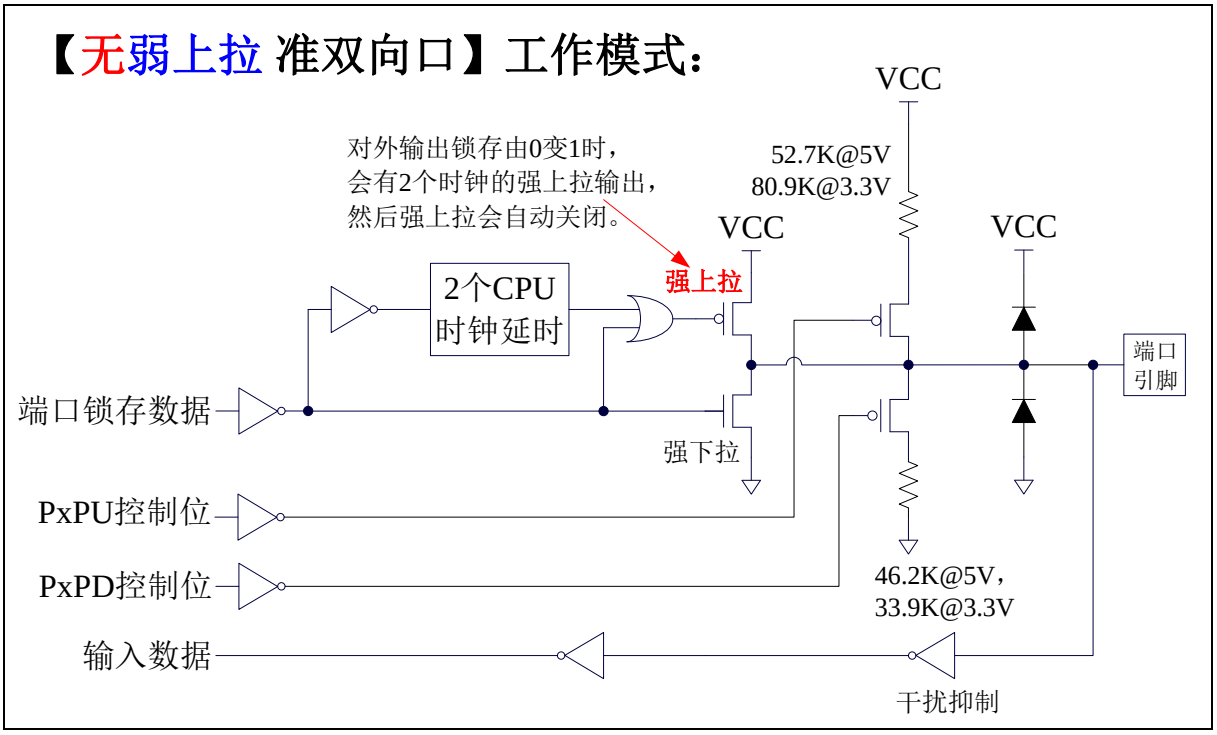
虽然每个 I/O 口在弱上拉（准双向口）/强推挽输出/开漏模式时都能承受 20mA 的灌电流（还是要加限流电阻，如 1K、560Ω、472Ω 等），在强推挽输出时能输出 20mA 的拉电流（也要加限流电阻），但整个芯片的工作电流推荐不要超过 90mA，即从 VCC 流入的电流建议不要超过 90mA，从 GND 流出电流建议不要超过 90mA，整体流入/流出电流建议都不要超过 90mA。

14.3 I/O 的结构图

14.3.1 【无弱上拉 准双向口】+上拉电阻 = 准双向口

和 STC 传统的准双向口不同，STC32G144K246 系列的【无弱上拉 准双向口】，传统的弱上拉和极弱上拉都没有，当准双向口使用时，需要打开芯片内部为其增加的上拉电阻，或在芯片外部加 5K～50K 上拉电阻。

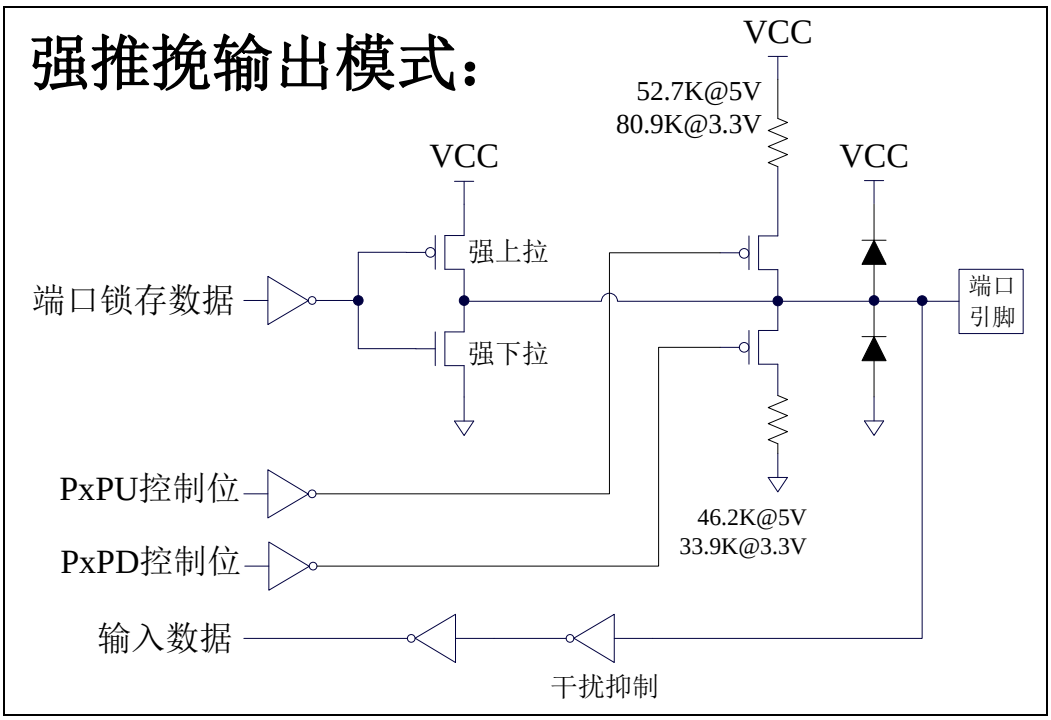
【无弱上拉 准双向口】模式，当对外输出锁存器从 0 到 1 变化时，有 2 个时钟的强推挽输出，后续这个强推挽输出自动关闭，2 个时钟的强上拉是为了加快 I/O 从 0 上升到 1 的速度。这是和【开漏模式】的根本区别，这个模式就是在【开漏模式】的基础上，增加了 2 个时钟的强推挽输出。



14.3.2 强推挽输出/强上拉

强推挽输出配置的下拉结构与开漏模式以及准双向口的下拉结构相同，但当锁存器为 1 时提供持续的强上拉。推挽模式一般用于需要更大驱动电流的情况。

强推挽引脚配置如下图所示：

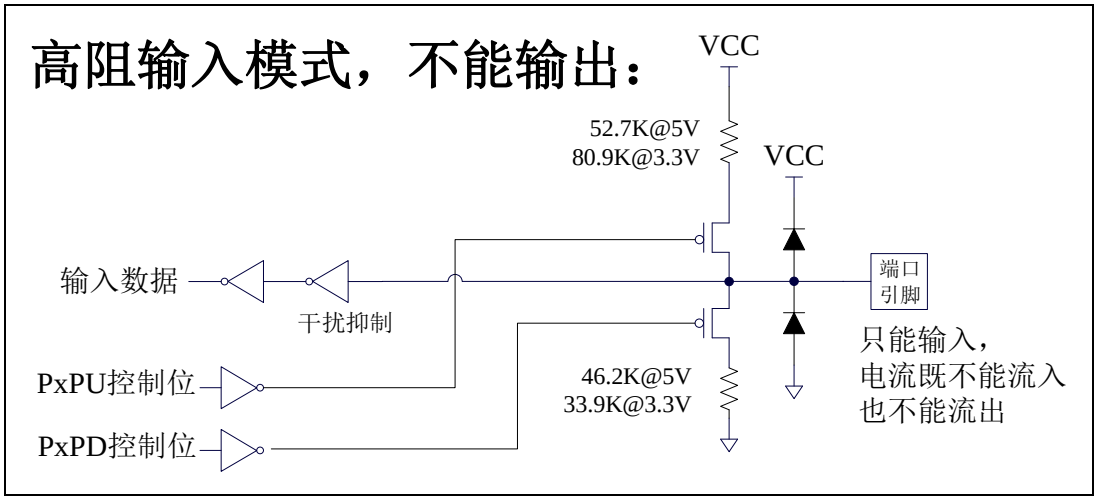


14.3.3 高阻输入(电流既不能流入也不能流出)

电流既不能流入也不能流出

输入口带有一个施密特触发输入以及一个干扰抑制电路

高阻输入引脚配置如下图所示：



14.3.4 开漏模式

===【开漏工作模式】，对外设置输出为 1，等同于 【高阻输入】

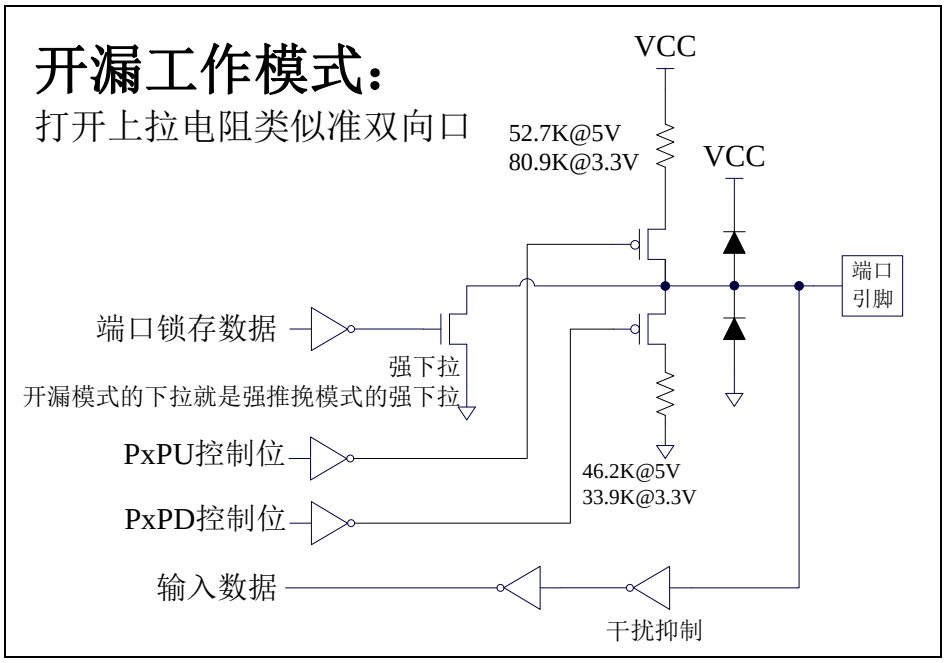
===【开漏工作模式】，【打开内部上拉电阻 | 或外部加上拉电阻】，简单等同于 【准双向口】

开漏模式既可读外部状态也可对外输出（高电平或低电平）。如要正确读外部状态或需要对外输出高电平，需外加上拉电阻。

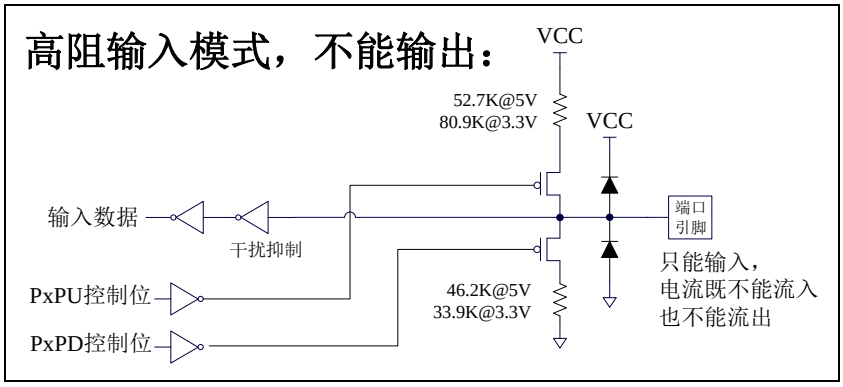
当端口锁存器为 0 时，开漏模式关闭所有上拉晶体管。当作为一个逻辑输出高电平时，这种配置方式必须有外部上拉，一般通过电阻外接到 VCC。如果外部有上拉电阻，开漏的 I/O 口还可读外部状态，即此时被配置为开漏模式的 I/O 口还可作为输入 I/O 口。这种方式的下拉与准双向口相同。

开漏端口带有一个施密特触发输入以及一个干扰抑制电路。

输出端口配置如下图所示：



14.3.5 STC32G144K246 内部上拉电阻/下拉电阻，小结



1、STC32G144K246 的每个 I/O 均可独立使能内部上拉电阻和下拉电阻，常温下：
上拉电阻阻值范围：52.7K@5V，80.9K@3.3V
下拉电阻阻值范围：46.2K@5V，33.9K@3.3V

2、端口上拉电阻控制寄存器（PnPU）
0：禁止端口内部的上拉电阻
1：使能端口内部的上拉电阻

3、端口下拉电阻控制寄存器（PnPD）
0：禁止端口内部的下拉电阻
1：使能端口内部的下拉电阻

4、内部上拉电阻和外部上拉电阻的部分区别：
- 内部上拉电阻是运行用户程序时才使能打开的，由于外部电路的分布电容效应，要等待 0.7RC 时间（R：电阻，C：电容）才能将外部拉高到 0.7MCU_Vcc 高电平。建议将需要打开的内部上拉电阻使能后，直接多等待一会，如等待 1ms 后程序才认为外部已被拉高。
- 外部上拉电阻随系统电源从 0V 上升到 MCU_Vcc，到 3.3V 或 5V，这个时间是足够的长，系统上电复位后到初始化运行用户程序时，外部上拉电阻已将外部拉高到高电平。

5、STC32G144K246 的 I/O 口 4 种工作模式：
- 【无弱上拉 准双向口】模式：打开芯片内部上拉电阻，或在芯片外部加 5K ~ 50K 上拉电阻，即可当准双向口使用。
- 强推挽输出/强上拉：大电流驱动，只能当作输出模式。
- 高阻输入：电流既不能流入也不能流出，只能当作输入模式。一般需要打开内部上拉电阻，或外部加上拉电阻。
- 开漏模式：打开芯片内部上拉电阻，简单等同于【准双向口】。一般一条总线上，需要 1 个 MCU 打开内部上拉电阻，或外部加 1 个上拉电阻到电压低的芯片的 VCC。

6、单片机 I/O 口 每种工作模式的应用场景：
- 准双向口：兼容传统 8051 的输入/输出模式，读外部时要确保已对外输出高电平。
- 强推挽输出：建议用于输出模式
- 高阻输入：建议用于输入模式/要配合上拉电阻，内部有。
- 开漏模式：主要是接比自己工作电压低的外设/要加 10K 上拉电阻到低压端的 VCC。

14.3.6 如何设置 I/O 口对外输出速度

当用户需要 I/O 口对外输出较快的频率时，可通过加大 I/O 口驱动电流以及增加 I/O 口电平转换速度以达到提高 I/O 口对外输出速度。

设置 PxSR 寄存器，可用于控制 I/O 口电平转换速度，设置为 0 时相应的 I/O 口为快速翻转，设置为 1 时为慢速翻转。

设置 PxDR 寄存器，可用于控制 I/O 口驱动电流大小，设置为 1 时 I/O 输出为一般驱动电流，设置为 0 时为强驱动电流。

14.3.7 如何设置 I/O 口电流驱动能力

若需要改变 I/O 口的电流驱动能力，可通过设置 PxDR 寄存器来实现。

设置 PxDR 寄存器，可用于控制 I/O 口驱动电流大小，设置为 1 时 I/O 输出为一般驱动电流，设置为 0 时为强驱动电流。

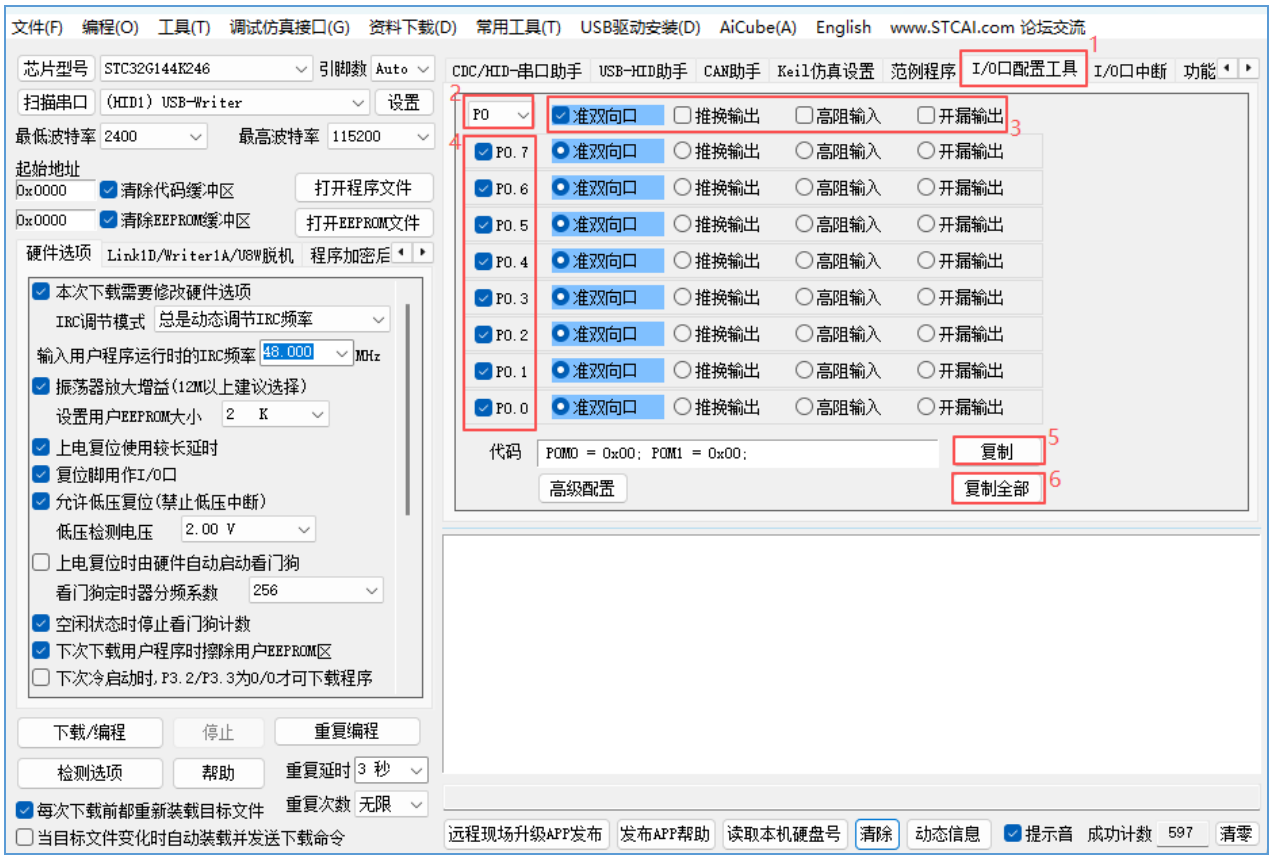
14.3.8 如何降低 I/O 口对外辐射

由于设置 PxSR 寄存器，可用于控制 I/O 口电平转换速度，设置 PxDR 寄存器，可用于控制 I/O 口驱动电流大小。

当需要降低 I/O 口对外的辐射时，需要将 PxSR 寄存器设置为 1 以降低 I/O 口电平转换速度，同时需要将 PxDR 寄存器设为 1 以降低 I/O 驱动电流，最终达到降低 I/O 口对外辐射。

14.4 AiCube-ISP | I/O 口配置工具

14.4.1 普通配置模式

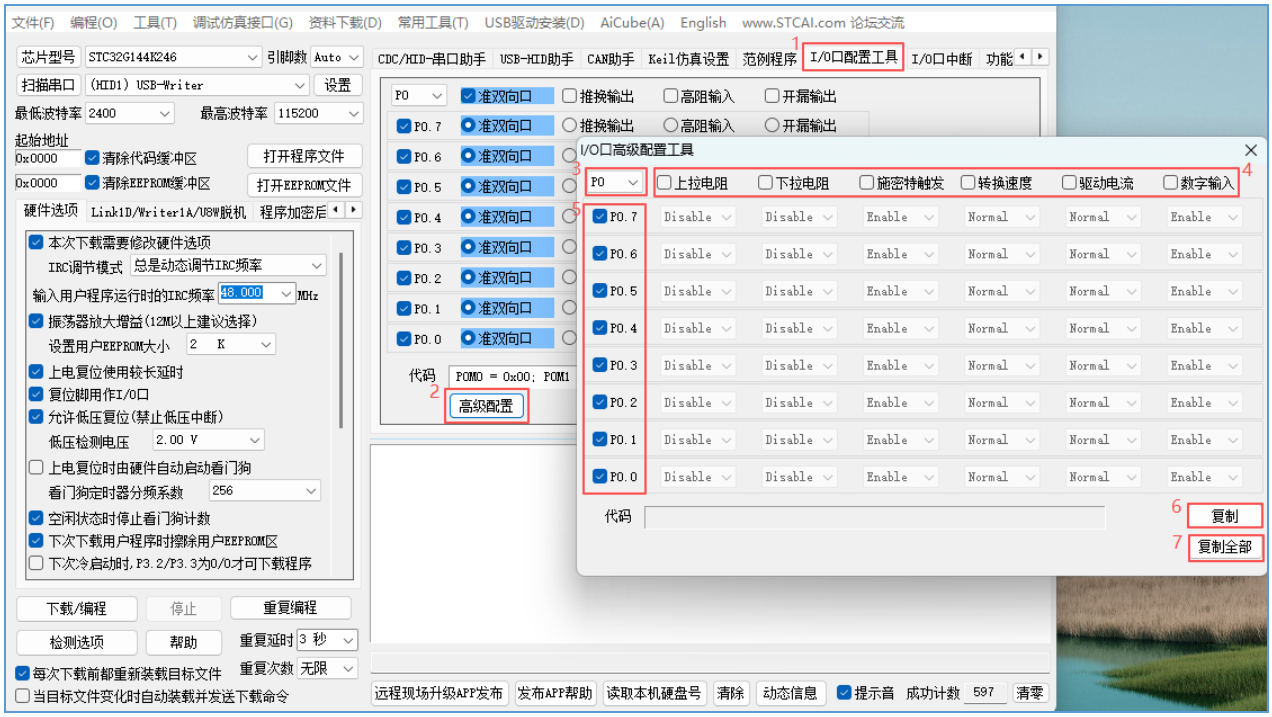


普通模式可以配置所有 I/O 的 4 中模式：

准双向口模式
推挽输出模式
高阻输入模式
开漏模式

①：在下载软件中选择“I/O 口配置工具”功能页，进入 I/O 口配置界面
②：选择需要配置的 I/O 口组，支持 P0 ~ P7
③：整组 I/O 进行设置（将整组 P0/P1/…/P7 设置为）准双向口模式、推挽输出模式等
④：选择使能配置每组 I/O 中的端口
⑤：复制当前组 I/O 的配置代码
⑥：复制 P0 ~ P7 口的全部配置代码

14.4.2 高级配置模式



高级模式可以配置 I/O：

使能/关闭上拉电阻
使能/关闭下拉电阻
使能/关闭施密特触发功能
选择 I/O 转换速度
选择 I/O 口驱动电流
使能/关闭 I/O 的数组输入功能

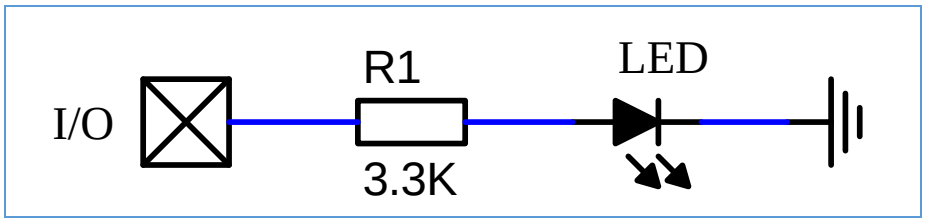
①：在下载软件中选择“I/O 口配置工具”功能页，进入 I/O 口配置界面
②：点击界面中的“高级配置”按钮进入 I/O 口高级配置界面
③：选择需要配置的 I/O 口组，支持 P0 ~ P7
④：选择需要的配置模式
⑤：选择使能配置每组 I/O 中的端口
⑥：复制当前组 I/O 的配置代码
⑦：复制 P0 ~ P7 口的全部配置代码

14.5 典型发光二极管控制电路

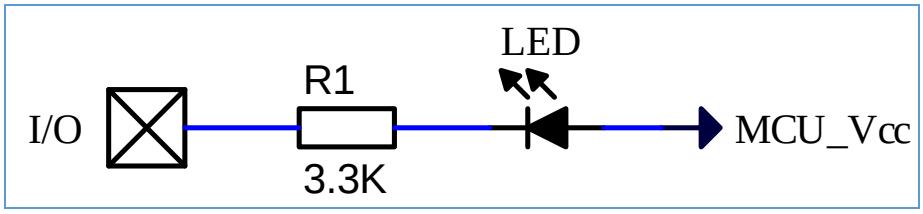
LED 点灯，NPN/PNP 三极管，N-MOS 管/P-MOS 管，光耦，继电器

I/O 口，控制 LED，尽量用强推挽**高电平导通**

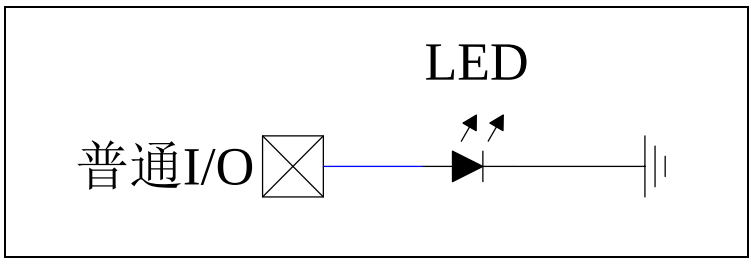
===原因，现在 ADC 的模拟地和 MCU 的数字地封装时打在一起，
低电平点灯的电流如向 MCU 内部流，会干扰 ADC 的模拟地线。
STC 大部分的 MCU 的地线要当模拟地来看待，
短而粗的 MCU 地线，独立走线最后接到系统地。
被测量信号的模拟地：
===1，直接接到 MCU-GND 的管脚上
===2，独立走线最后接到系统地



使用**强推挽**模式对外输出高，点亮 LED 灯，
电流向外流，不影响内部 ADC 的地线
===1，串口下载，P3.0/P3.1 处不能如此高电平点灯，否则无法通信
===2，USB 通信的【D-/P3.0, D+/P3.1】，不能外接点灯电路

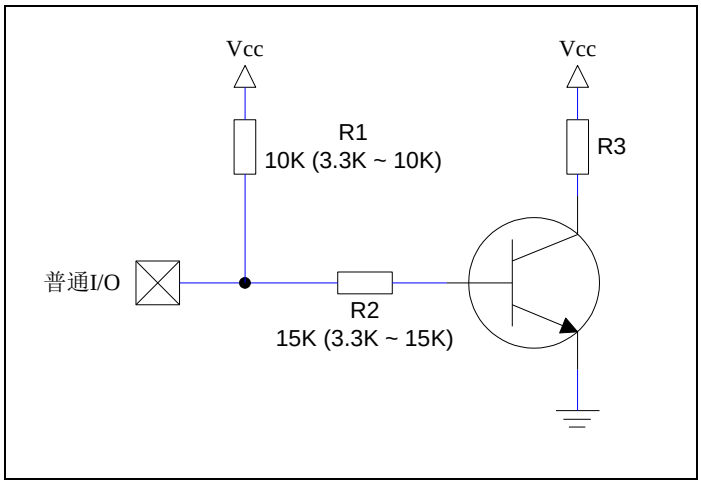


无 ADC，无 USB 的 MCU
P3.0/P3.1 建议使用上图点灯，用低电平导通电路点亮 LED 灯，
ISP 下载程序时可以看到通信时灯会闪烁，方便观察。

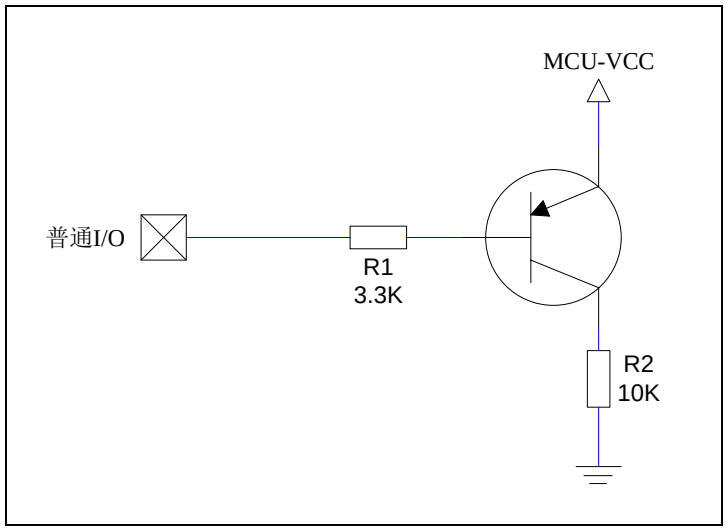


有网友说不管亮度，只要省成本，I/O 口，不接限流电阻，直接驱动 LED 灯。
要省外部这个 1K ~ 5K 的限流电阻，那就用内部粗糙的 4K 上拉电阻代替外部的限流电阻。

14.6 一种典型三极管控制电路



上图中，如果使用弱上拉控制，建议加上拉电阻 R1(3.3K~10K)，如果不加上拉电阻 R1(3.3K~10K)，建议 R2 的值在 15K 以上，或用强推挽输出。

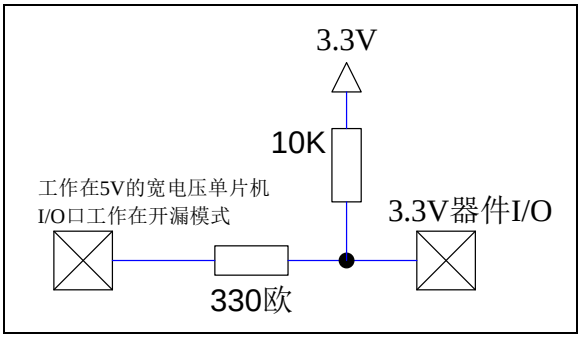


上图中，不接限流电阻 R1，I/O 口直接控制 PNP 三级管，网友说只要省成本
要省这个 3.3K 限流电阻，那就用内部粗糙的 47K 下拉电阻代替
STC32G144K246 的 I/O 口上电是：【高阻输入+ 4K 上拉电阻关闭+ 47K 下拉电阻关闭】
可 I/O 口直接控制 PNP 三级管，保持高阻输入：
===打开内部 47K 下拉电阻，PNP 三级管导通
===关闭内部 47K 下拉电阻，PNP 三级管关闭

14.7 混合电压供电系统 3V/5V 器件 I/O 口互连

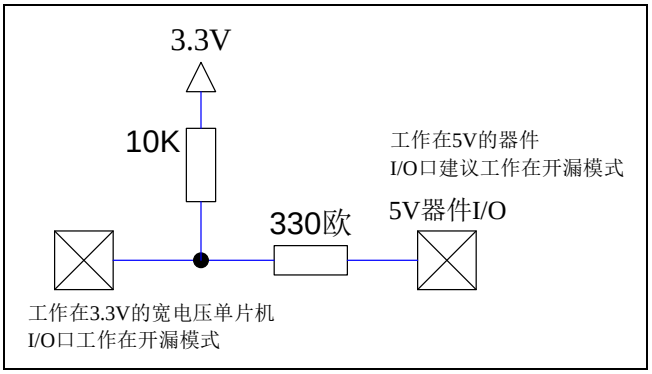
宽电压单片机工作在 5V 时：

如需要直接连接 3.3V 器件时，为防止 3.3V 器件承受不了 5V，可将相应的单片机 I/O 口先串一个 330Ω 的限流电阻到 3.3V 器件 I/O 口，程序初始化时将单片机的 I/O 口设置成开漏工作模式，断开内部上拉电阻，相应的 3.3V 器件 I/O 口外部加 10K 上拉电阻到 3.3V 器件的 Vcc，这样高电平是 3.3V，低电平是 0V，输入输出一切正常。宽电压单片机 2.2V 以上肯定是高电平。



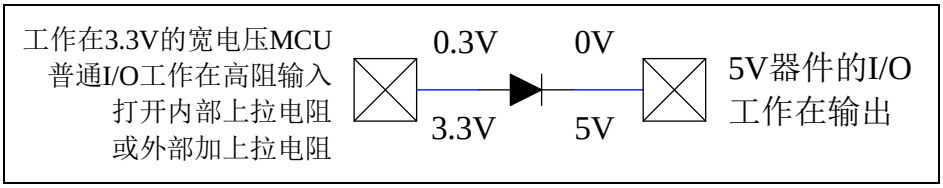
宽电压 MCU 如工作在 3.3V 时，如需要连接到 5V 器件：

1、如果对方 5V 器件可以工作在开漏模式：

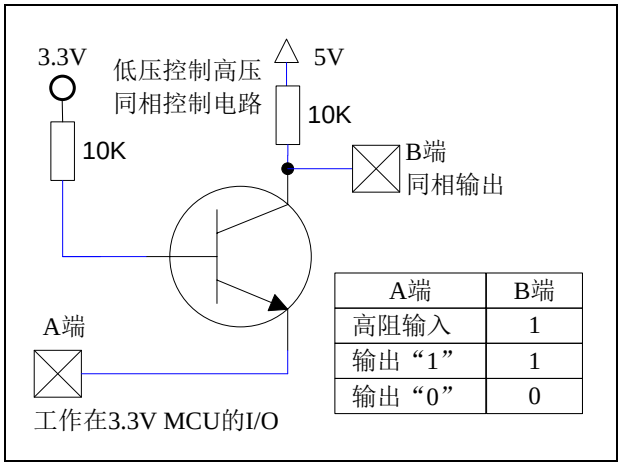


2、如是连接到 5V 器件的高阻输入，则可以直接相连，或串个 300 欧电阻相连。工作在 3.3V 的宽电压单片机的 I/O 可以工作在强推挽输出模式，或者工作在准双向口/弱上拉模式。

3、如果对方是 5V 输出，我方可工作在高阻输入模式，打开内部上拉电阻，外部串接一个肖特基二极管隔离，隔离高压 5V。外部电压高于单片机工作电压时肖特基二极管截止，I/O 口因内部上拉到高电平，所以读 I/O 口状态是高电平；外部电压为低时隔离的肖特基二极管导通，I/O 口被钳位在 0.3V，小于 0.8V 时单片机读 I/O 口状态是低电平。肖特基二极管可以使用 1N5817/1N5819，不要使用硅二极管。2.2V 以上是高电平，0.8V 以下是低电平。

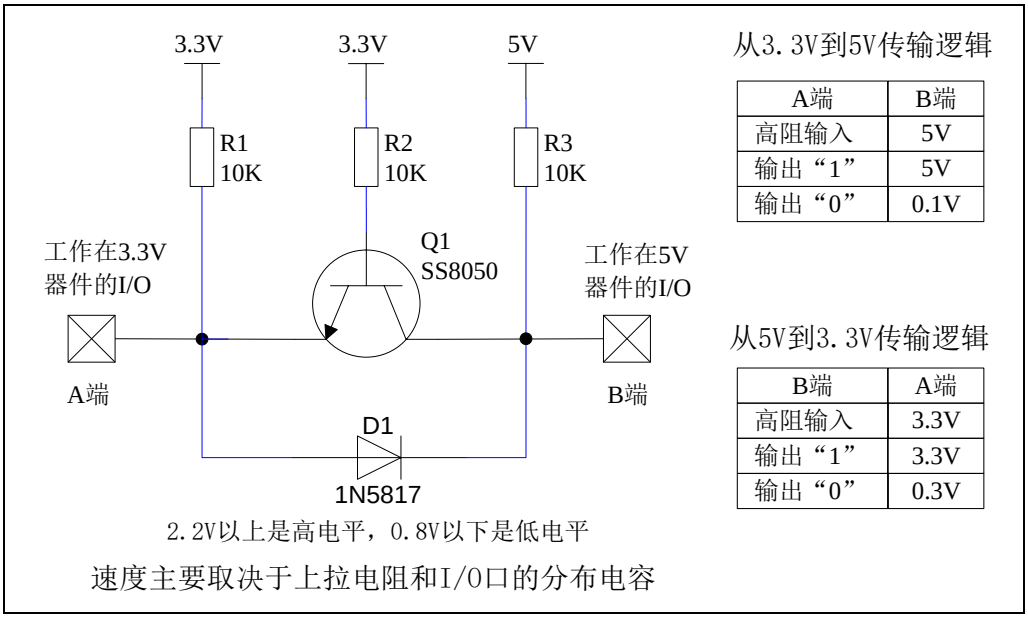


4、宽电压单片机工作在 3.3V 时，如需要控制 5V 器件，还可以用下图同相控制电路：



A端	B端
高阻输入	1
输出“1”	1
输出“0”	0

5、工作在 3.3V 的器件和工作在 5V 的器件打交道，还可以用如下【三极管+二极管】双向电路：



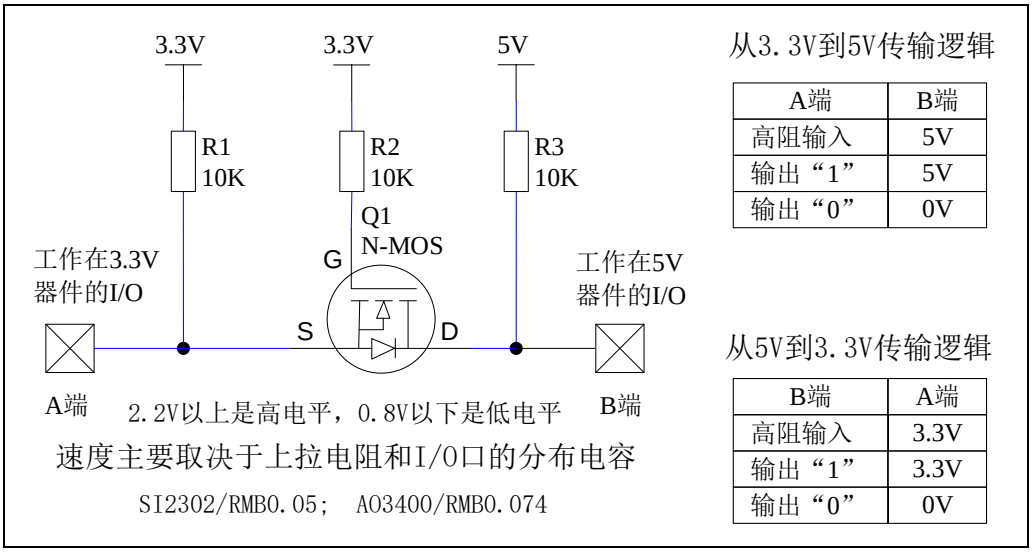
从3.3V到5V传输逻辑

A端	B端
高阻输入	5V
输出“1”	5V
输出“0”	0.1V

从5V到3.3V传输逻辑

B端	A端
高阻输入	3.3V
输出“1”	3.3V
输出“0”	0.3V

6、工作在 3.3V 的器件和工作在 5V 的器件打交道，还可以用如下 MOS 管双向电路：



从3.3V到5V传输逻辑

A端	B端
高阻输入	5V
输出“1”	5V
输出“0”	0V

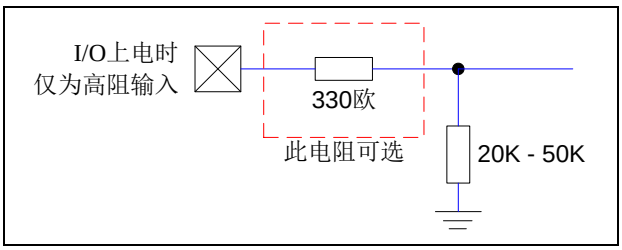
从5V到3.3V传输逻辑

B端	A端
高阻输入	3.3V
输出“1”	3.3V
输出“0”	0V

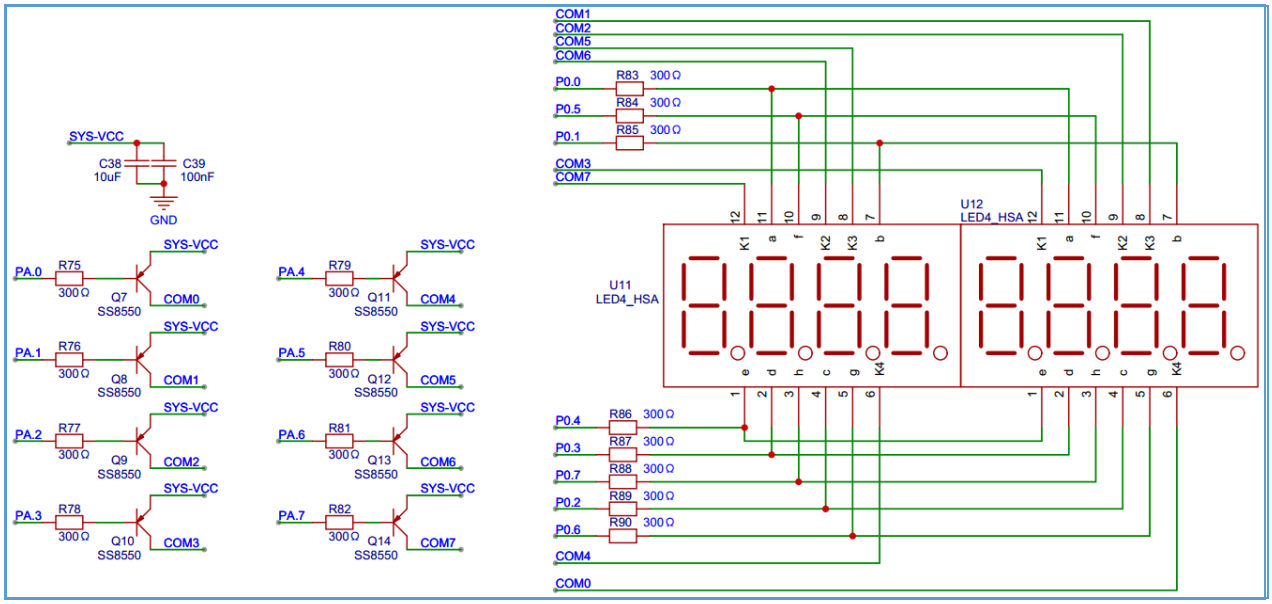
14.8 如何让 I/O 口上电复位时为低电平

普通 8051 单片机上电复位时普通 I/O 口为弱上拉(准双向口)高电平输出，而很多实际应用要求上电时某些 I/O 口为低电平输出，否则所控制的系统(如马达)就会误动作，现 STC32G144K246/8H/32G 系列单片机上电复位后 I/O 口为高阻输入。如外部加下拉电阻【20K - 50K】，则上电后就是低电平。如外部加上拉电阻【20K - 50K】，则上电后就是高电平。由于 I/O 口可以工作在准双向口/弱上拉，又可以工作在强推挽模式，后续可以很容易拉高或拉低，很简单的解决了这个问题。

如下图可在单片机 I/O 口上加一个下拉电阻(20K 左右)，这样上电复位时，除了下载口 P3.0 和 P3.1 为弱上拉(准双向口)外，其他 I/O 口均为高阻输入模式，而外部加下拉电阻，所以该 I/O 口上电复位时外部为低电平。如果要将此 I/O 口驱动为高电平，可将此 I/O 口设置为强推挽输出，而强推挽输出时，I/O 口驱动电流可达 20mA，故肯定可以将该口驱动为高电平输出。

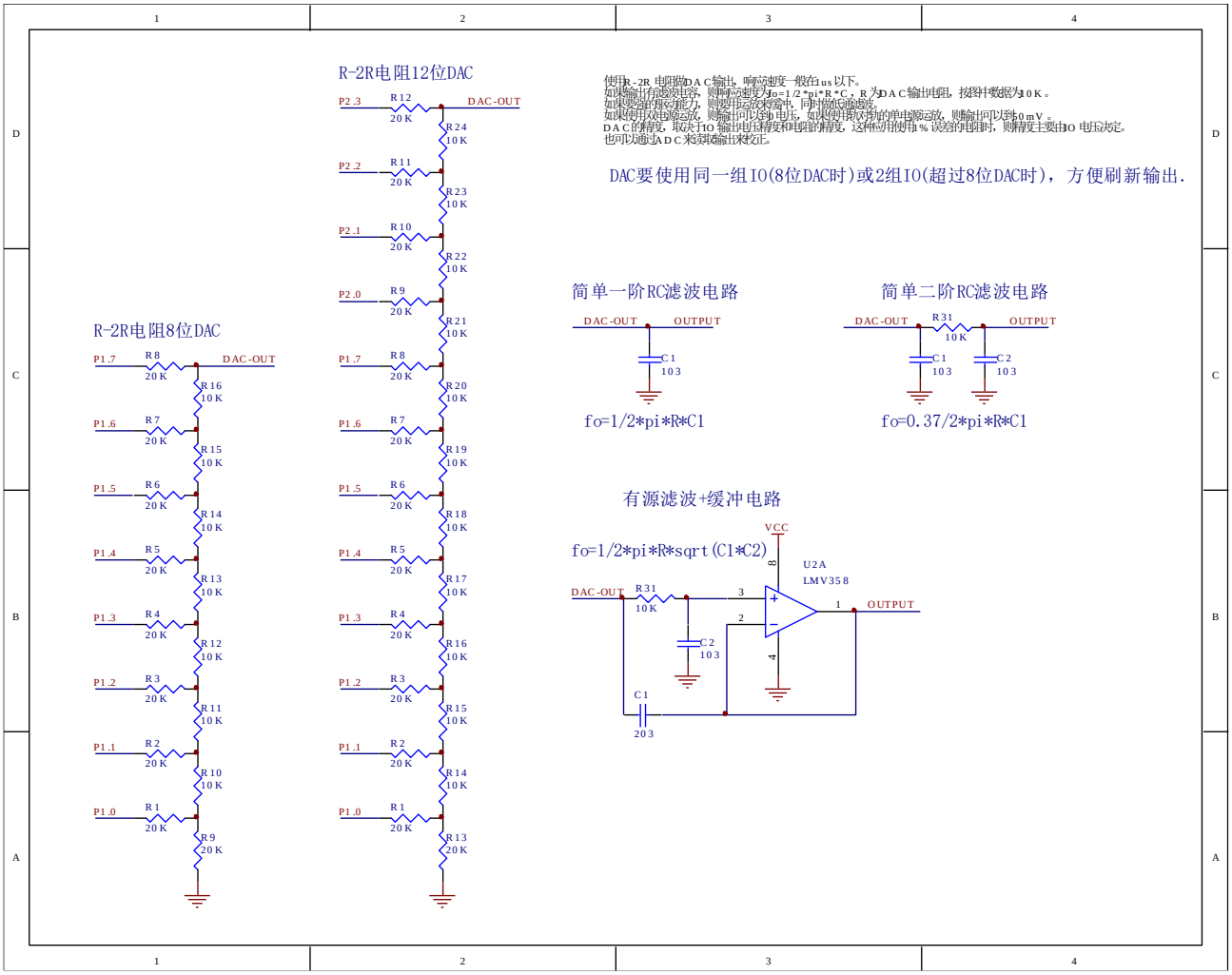


14.9 I/O 口直接驱动 LED 数码管应用线路图

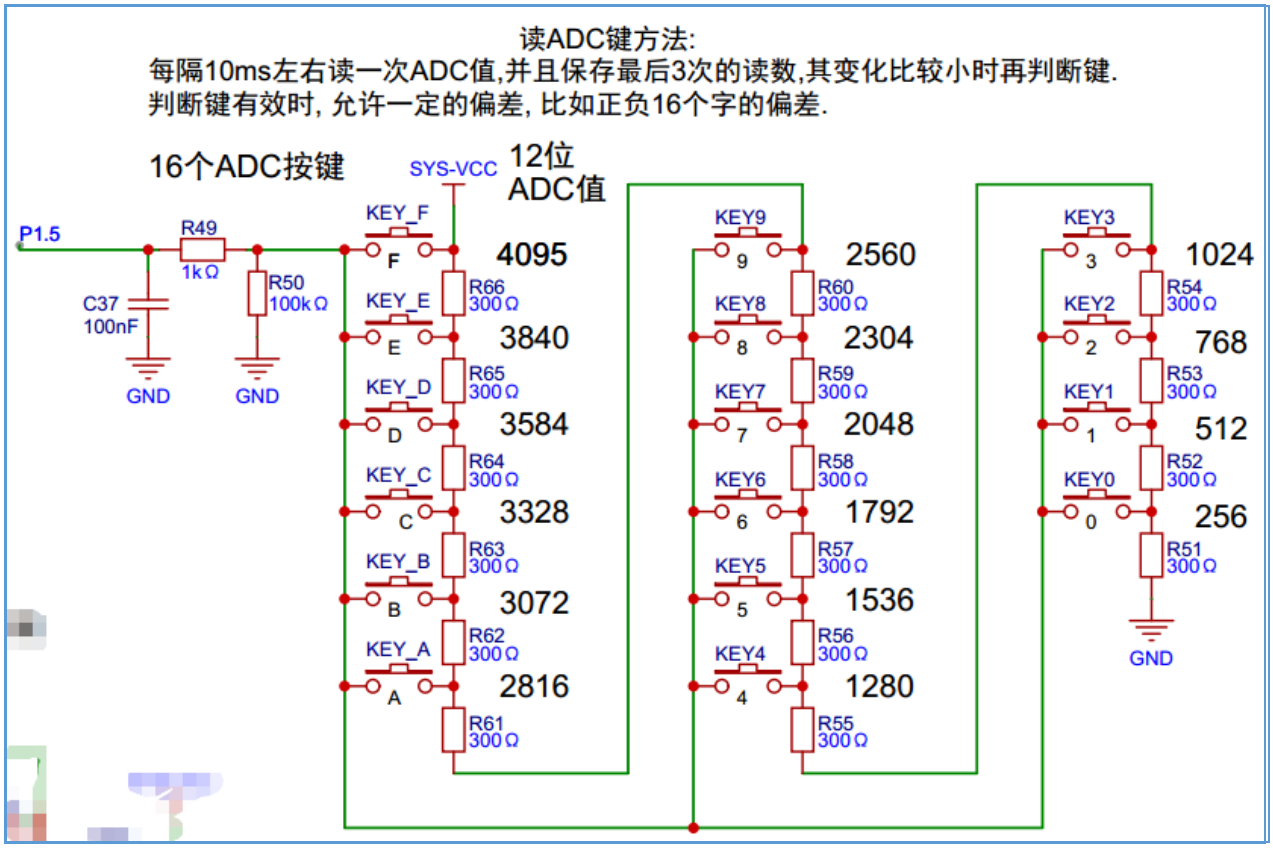


注：上图的 COM0 ～ COM7 公共端可以使用 I/O 口来控制（例如：P7.0~P7.7），不需要加限流电阻。段码限流电阻使用 470 欧 ～ 1K，大些好

14.10 使用 I/O 和 R-2R 电阻分压实现 DAC 的经典线路图

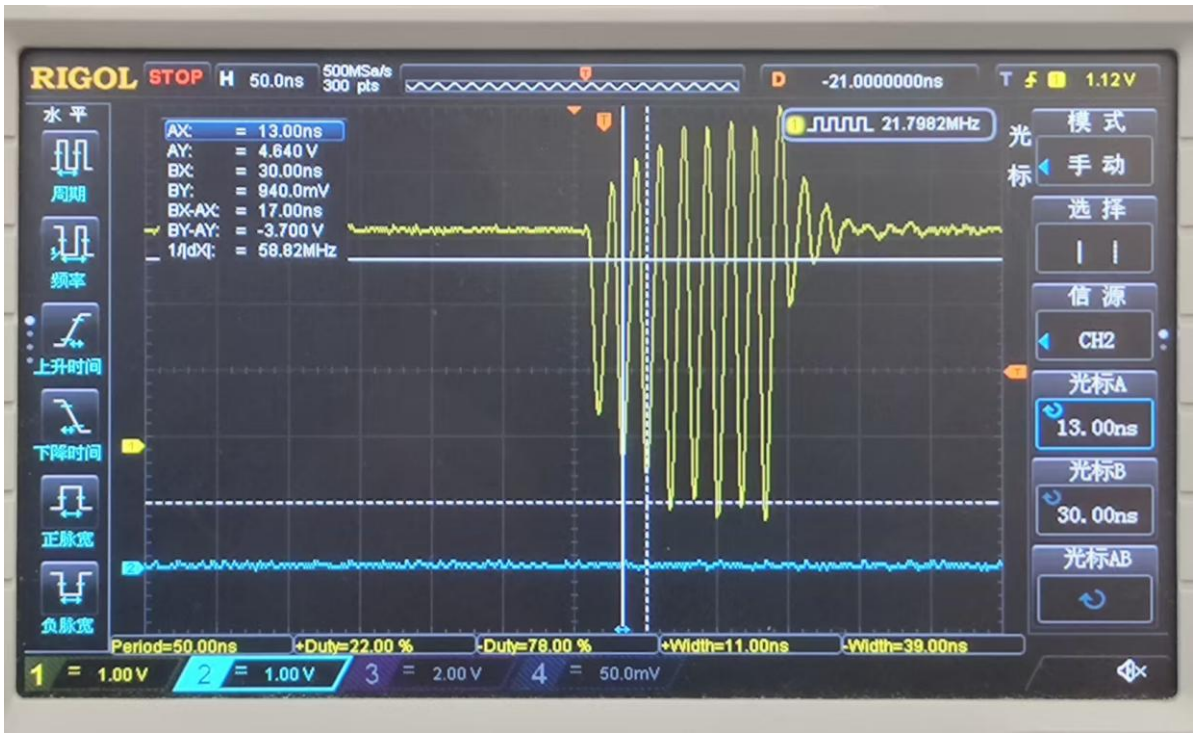


14.11 利用 ADC 做按键扫描节省 I/O 口



14.12 SPI 实测波形图

工作电压： 3.3V
SPI 工作频率： 58.82MHz
SPI 时钟线模式： 强推挽输出模式



工作电压： 3.3V
SPI 工作频率： 30.30MHz
SPI 时钟线模式： 强推挽输出模式



14.13 QSPI 实测波形图

工作电压： 3.3V
QSPI 工作频率： 43.48MHz
QSPI 时钟线模式：强推挽输出模式
QSPI 数据线模式：开漏模式+内部上拉，硬件自动控制输入/输出方向



工作电压： 3.3V
QSPI 工作频率： 31.25MHz
QSPI 时钟线模式：强推挽输出模式
QSPI 数据线模式：开漏模式+内部上拉，硬件自动控制输入/输出方向



工作电压：　　3.3V

QSPI 工作频率：　20.00MHz

QSPI 时钟线模式：强推挽输出模式

QSPI 数据线模式：开漏模式+内部上拉，硬件自动控制输入/输出方向



15 中断系统及外部中断 INT0/1/2/3/4

中断系统是为使 CPU 具有对外界紧急事件的实时处理能力而设置的。

当中央处理机 CPU 正在处理某件事的时候外界发生了紧急事件请求，要求 CPU 暂停当前的工作，转而去处理这个紧急事件，处理完以后，再回到原来被中断的地方，继续原来的工作，这样的过程称为中断。实现这种功能的部件称为中断系统，请示 CPU 中断的请求源称为中断源。微型机的中断系统一般允许多个中断源，当几个中断源同时向 CPU 请求中断，要求为它服务的时候，这就存在 CPU 优先响应哪一个中断源请求的问题。通常根据中断源的轻重缓急排队，优先处理最紧急事件的中断请求源，即规定每一个中断源有一个优先级别。CPU 总是先响应优先级别最高的中断请求。

当 CPU 正在处理一个中断源请求的时候（执行相应的中断服务程序），发生了另外一个优先级比它还高的中断源请求。如果 CPU 能够暂停对原来中断源的服务程序，转而去处理优先级更高的中断请求源，处理完以后，再回到原低级中断服务程序，这样的过程称为中断嵌套。这样的中断系统称为多级中断系统，没有中断嵌套功能的中断系统称为单级中断系统。

用户可以用关总中断允许位（EA/IE.7）或相应中断的允许位屏蔽相应的中断请求，也可以用打开相应的中断允许位来使 CPU 响应相应的中断申请，每一个中断源可以用软件独立地控制为开中断或关中断状态，部分中断的优先级别均可用软件设置。高优先级的中断请求可以打断低优先级的中断，反之，低优先级的中断请求不可以打断高优先级的中断。当两个相同优先级的中断同时产生时，将由查询次序来决定系统先响应哪个中断。

15.1 STC32G144K246 系列中断源

中断源	STC32G144K246系列
外部中断 0 中断（INT0，从 P3.2 管脚输入）	√
定时器 0 中断（Timer0）	√
外部中断 1 中断（INT1，从 P3.3 管脚输入）	√
定时器 1 中断（Timer1）	√
串口 1 中断（UART1）	√
模数转换中断（ADC）	√
低压检测中断（LVD）	√
CCP/PCA/PWM 中断	-
串口 2 中断（UART2）	√
串行外设接口中断（SPI）	√
外部中断 2 中断（INT2，从 P3.6 管脚输入）	√
外部中断 3 中断（INT3，从 P3.7 管脚输入）	√
定时器 2 中断（Timer2）	√
外部中断 4 中断（INT4，从 P3.0 管脚输入）	√
串口 3 中断（UART3）	√
串口 4 中断（UART4）	√
定时器 3 中断（Timer3）	√
定时器 4 中断（Timer4）	√

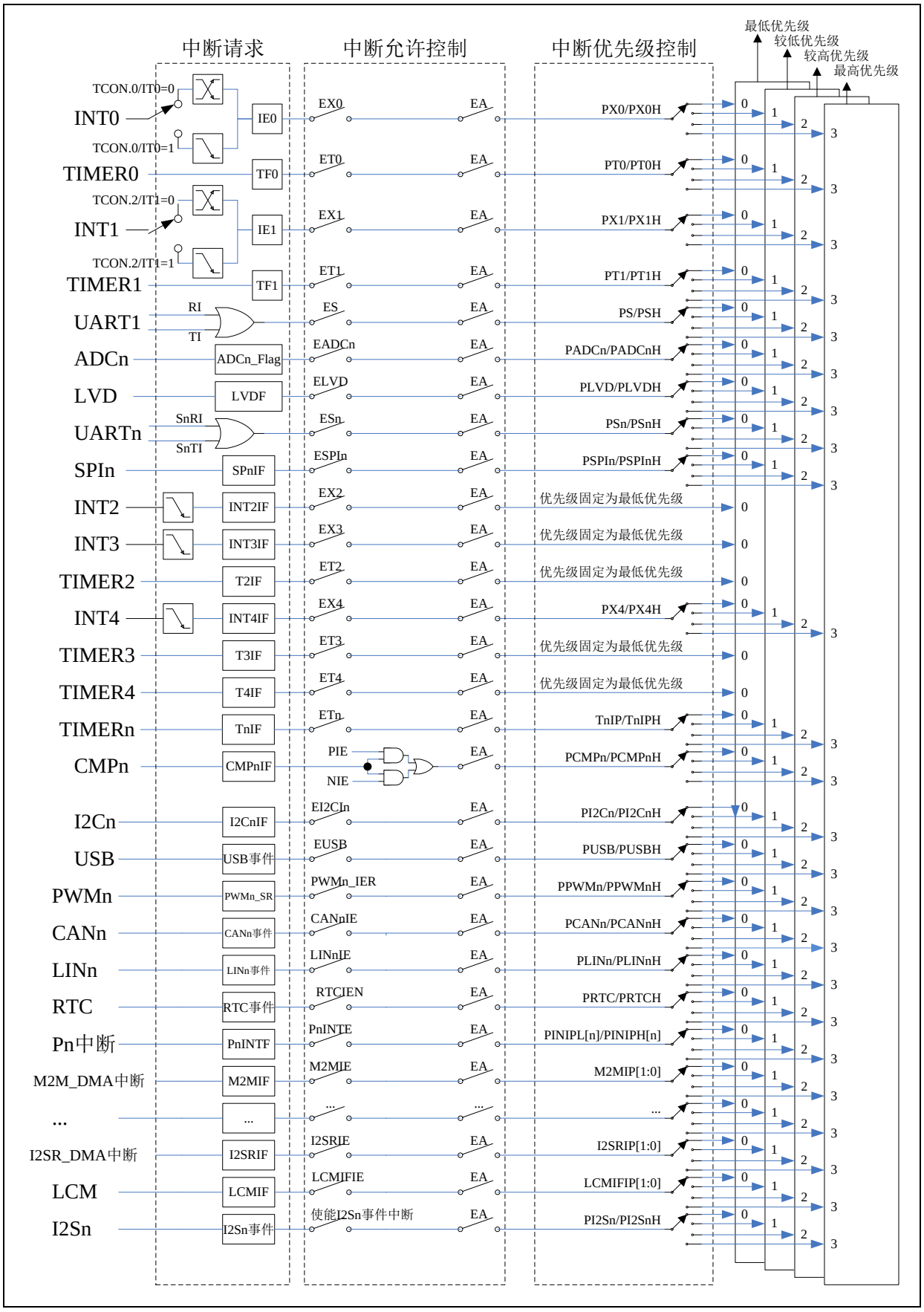
中断源	STC32G144K246系列
比较器中断（CMP）	√
I2C 总线中断	√
USB 中断	√
PWMA	√
PWMB	√
CAN1 中断	√
CAN2 中断	√
LIN1 中断	√
LIN2 中断	√
RTC 中断	√
P0 口中断	√
P1 口中断	√
P2 口中断	√
P3 口中断	√
P4 口中断	√
P5 口中断	√
P6 口中断	√
P7 口中断	√
P8 口中断	√
P9 口中断	√
M2M_DMA 中断	√
ADC_DMA 中断	√
SPI_DMA 中断	√
串口 1 发送 DMA 中断	√
串口 1 接收 DMA 中断	√
串口 2 发送 DMA 中断	√
串口 2 接收 DMA 中断	√
串口 3 发送 DMA 中断	√
串口 3 接收 DMA 中断	√
串口 4 发送 DMA 中断	√
串口 4 接收 DMA 中断	√
TFT 彩屏 DMA 中断	√
TFT 彩屏中断	√
I2C 发送 DMA 中断	√
I2C 接收 DMA 中断	√
I2S 中断	√

中断源	STC32G144K246系列
I2S 发送 DMA 中断	√
I2S 接收 DMA 中断	√
QSPI_DMA 中断	√
QSPI 中断	√
定时器 11 中断（Timer11）	√
I2C2 发送 DMA 中断	√
I2C2 接收 DMA 中断	√
I2S2 发送 DMA 中断	√
I2S2 接收 DMA 中断	√
PWMA 发送 DMA 中断	√
PWMA 接收 DMA 中断	√
PWMC 发送 DMA 中断	√
PWMC 接收 DMA 中断	√
ADC2_DMA 中断	√
DAC1_DMA 中断	√
DAC2_DMA 中断	√
SPI2_DMA 中断	√
SPI3_DMA 中断	√
SPI4_DMA 中断	-
串口 5 发送 DMA 中断	√
串口 5 接收 DMA 中断	√
串口 6 发送 DMA 中断	√
串口 6 接收 DMA 中断	√
串口 7 发送 DMA 中断	√
串口 7 接收 DMA 中断	√
串口 8 发送 DMA 中断	√
串口 8 接收 DMA 中断	√
PA 口中断	√
PB 口中断	√
PC 口中断	-
PD 口中断	-
PE 口中断	-
PF 口中断	-
定时器 5/定时器 6 中断（Timer5/Timer6）	√
定时器 7/定时器 8 中断（Timer5/Timer6）	√
定时器 9/定时器 10 中断（Timer5/Timer6）	√

中断源	STC32G144K246系列
定时器 17/定时器 18 中断（Timer5/Timer6）	√
串口 5 中断（UART5）	√
串口 6 中断（UART6）	√
串口 7 中断（UART7）	√
串口 8 中断（UART8）	√
模数转换 2 中断（ADC2）	√
数模转换 1 中断（DAC1）	√
数模转换 2 中断（DAC2）	√
I2C2 总线中断	√
I2S2 中断	√
串行外设接口 2 中断（SPI2）	√
串行外设接口 3 中断（SPI3）	√
串行外设接口 4 中断（SPI4）	-
比较器 2 中断（CMP2）	√
比较器 3 中断（CMP3）	√
比较器 4 中断（CMP4）	√
CAN1 发送 DMA 中断	√
CAN1 接收 DMA 中断	√
CAN2 发送 DMA 中断	√
CAN2 接收 DMA 中断	√
PWMC	√
PWMD	√
PWME	√
PWMF	√
PWME 发送 DMA 中断	√
PWME 接收 DMA 中断	√

15.2 STC32G144K246 中断及中断优先级结构图

注：高优先级中断可以打断正在执行中的低优先级中断，跟传统 STC89C52 一样



15.3 STC32G144K246 系列中断向量地址及同级中断优先级中断查询次序表

中断源	中断向量	相同优先级内的查询次序	优先级设置	优先级 0（最低）	优先级 1	优先级 2	优先级 3（最高）	中断请求位	中断允许位
INT0/P3.2 中断	FF:0003H	0（最高）	PX0, PX0H	0	1	2	3	IE0	EX0
Timer0 中断	FF:000BH	1	PT0, PT0H	0	1	2	3	TF0	ET0
INT1/P3.3 中断	FF:0013H	2	PX1, PX1H	0	1	2	3	IE1	EX1
Timer1 中断	FF:001BH	3	PT1, PT1H	0	1	2	3	TF1	ET1
UART1 中断	FF:0023H	4	PS, PSH	0	1	2	3	RI || TI	ES
ADC 中断	FF:002BH	5	PADC, PADCH	0	1	2	3	ADC_FLAG	EADC
LVD 中断	FF:0033H	6	PLVD, PLVDH	0	1	2	3	LVDF	ELVD
UART2 中断	FF:0043H	8	PS2, PS2H	0	1	2	3	S2RI || S2TI	ES2
SPI 中断	FF:004BH	9	PSPI, PSPIH	0	1	2	3	SPIF	ESPI
INT2/P3.6 中断	FF:0053H	10		0	0	0	0	INT2IF	EX2
INT3/P3.7 中断	FF:005BH	11		0	0	0	0	INT3IF	EX3
Timer2 中断	FF:0063H	12		0	0	0	0	T2IF	ET2
INT4/P3.0 中断	FF:0083H	16	PX4, PX4H	0	1	2	3	INT4IF	EX4
UART3 中断	FF:008BH	17	PS3, PS3H	0	1	2	3	S3RI || S3TI	ES3
UART4 中断	FF:0093H	18	PS4, PS4H	0	1	2	3	S4RI || S4TI	ES4
Timer3 中断	FF:009BH	19		0	0	0	0	T3IF	ET3
Timer4 中断	FF:00A3H	20		0	0	0	0	T4IF	ET4
CMP 中断	FF:00ABH	21	PCMP, PCMPH	0	1	2	3	CMPIF	PIE|NIE
I2C 中断	FF:00C3H	24	PI2C, PI2CH	0	1	2	3	MSIF	EMSI
								STAIF	ESTAI
								RXIF	ERXI
								TXIF	ETXI
								STOIF	ESTOI
USB 中断	FF:00CBH	25	PUSB, PUSBH	0	1	2	3	USB Events	EUSB
PWMA 中断	FF:00D3H	26	PPWMA, PPWMAH	0	1	2	3	PWMA_SR	PWMA_IER
PWMB 中断	FF:00DBH	27	PPWMB, PPWMBH	0	1	2	3	PWMB_SR	PWMB_IER

中断源	中断向量	相同优先级内的查询次序	优先级设置	优先级 0（最低）	优先级 1	优先级 2	优先级 3（最高）	中断请求位	中断允许位
CAN1 中断	FF:00E3H	28	PCANL, PCANH	0	1	2	3	ALI	ALIM
								EWI	EWIM
								EPI	EPIM
								RI	RIM
								TI	TIM
								BEI	BEIM
								DOI	DOIM
CAN2 中断	FF:00EBH	29	PCAN2L, PCAN2H	0	1	2	3	ALI	ALIM
								EWI	EWIM
								EPI	EPIM
								RI	RIM
								TI	TIM
								BEI	BEIM
								DOI	DOIM
LIN1 中断	FF:00F3H	30	PLINL, PLIN1H	0	1	2	3	ABORT	ABORTE
								ERR	ERRE
								RDY	RDYE
								LID	LIDE
LIN2 中断	FF:00FBH	31	PLIN2L, PLIN2H	0	1	2	3	ABORT	ABORTE
								ERR	ERRE
								RDY	RDYE
								LID	LIDE
RTC 中断	FF:0123H	36	PRTC, PRTCH	0	1	2	3	ALAIF	EALAI
								DAYIF	EDAYI
								HOURIF	EHOURI
								MINIF	EMINI
								SECIF	ESECI
								SEC2IF	ESEC2I
								SEC8IF	ESEC8I
								SEC32IF	ESEC32I

中断源	中断向量	相同优先级内的查询次序	优先级设置	优先级 0（最低）	优先级 1	优先级 2	优先级 3（最高）	中断请求位	中断允许位
P0 中断	FF:012BH	37	PINIPL[0], PINIPH[0]	0	1	2	3	P0INTF	P0INTE
P1 中断	FF:0133H	38	PINIPL[1], PINIPH[1]	0	1	2	3	P1INTF	P1INTE
P2 中断	FF:013BH	39	PINIPL[2], PINIPH[2]	0	1	2	3	P2INTF	P2INTE
P3 中断	FF:0143H	40	PINIPL[3], PINIPH[3]	0	1	2	3	P3INTF	P3INTE
P4 中断	FF:014BH	41	PINIPL[4], PINIPH[4]	0	1	2	3	P4INTF	P4INTE
P5 中断	FF:0153H	42	PINIPL[5], PINIPH[5]	0	1	2	3	P5INTF	P5INTE
P6 中断	FF:015BH	43	PINIPL[6], PINIPH[6]	0	1	2	3	P6INTF	P6INTE
P7 中断	FF:0163H	44	PINIPL[7], PINIPH[7]	0	1	2	3	P7INTF	P7INTE
P8 中断	FF:016BH	45	PIN2IPL[0], PIN2IPH[0]	0	1	2	3	P8INTF	P8INTE
P9 中断	FF:0173H	46	PIN2IPL[1], PIN2IPH[1]	0	1	2	3	P9INTF	P9INTE
DMA_M2M 中断	FF:017BH	47	M2MIP[1:0]	0	1	2	3	M2MIF	M2MIE
DMA_ADC 中断	FF:0183H	48	ADCIP[1:0]	0	1	2	3	ADCIF	ADCIE
DMA_SPI 中断	FF:018BH	49	SPIIP[1:0]	0	1	2	3	SPIIF	SPIIE
DMA_UR1T 中断	FF:0193H	50	UR1TIP[1:0]	0	1	2	3	UR1TIF	UR1TIE
DMA_UR1R 中断	FF:019BH	51	UR1RIP[1:0]	0	1	2	3	UR1RIF	UR1RIE
DMA_UR2T 中断	FF:01A3H	52	UR2TIP[1:0]	0	1	2	3	UR2TIF	UR2TIE
DMA_UR2R 中断	FF:01ABH	53	UR2RIP[1:0]	0	1	2	3	UR2RIF	UR2RIE
DMA_UR3T 中断	FF:01B3H	54	UR3TIP[1:0]	0	1	2	3	UR3TIF	UR3TIE
DMA_UR3R 中断	FF:01BBH	55	UR3RIP[1:0]	0	1	2	3	UR3RIF	UR3RIE
DMA_UR4T 中断	FF:01C3H	56	UR4TIP[1:0]	0	1	2	3	UR4TIF	UR4TIE
DMA_UR4R 中断	FF:01CBH	57	UR4RIP[1:0]	0	1	2	3	UR4RIF	UR4RIE
TFT 彩屏 DMA 中断	FF:01D3H	58	LCMIP[1:0]	0	1	2	3	LCMIF	LCMIE
TFT 彩屏中断	FF:01DBH	59	LCMIFIP[1:0]	0	1	2	3	LCMIFIF	LCMIFIE
DMA_I2CT 中断	FF:01E3H	60	I2CTIP[1:0]	0	1	2	3	I2CTIF	I2CTIE
DMA_I2CR 中断	FF:01EBH	61	I2CRIP[1:0]	0	1	2	3	I2CRIF	I2CRIE
I2S 中断	FF:01F3H	62	PI2S,PI2SH	0	1	2	3	TXE	TXEIE
				0	1	2	3	RXNE	RXNEIE
				0	1	2	3	FRE	ERRIE
								OVR	
								UDR	

中断源	中断向量	相同优先级内的查询次序	优先级设置	优先级 0（最低）	优先级 1	优先级 2	优先级 3（最高）	中断请求位	中断允许位
DMA_I2ST 中断	FF:01FBH	63	I2STIP[1:0]	0	1	2	3	I2STIF	I2STIE
DMA_I2SR 中断	FF:0203H	64	I2SRIP[1:0]	0	1	2	3	I2SRIF	I2SRIE
DMA_QSPI 中断	FF:020BH	65	QSPIIP[1:0]	0	1	2	3	QSPIIF	QSPIIE
QSPI 中断	FF:0213H	66	QSPI_IP[1:0]	0	1	2	3	SMF	SMIE
								FTF	FTIE
								TCF	TCIE
								TEF	TEIE
Timer11	FF:021BH	67		0	0	0	0	T11IF	ET11I
DMA_I2C2T 中断	FF:0223H	68	I2C2TIP[1:0]	0	1	2	3	I2C2TIF	I2C2TIE
DMA_I2C2R 中断	FF:022BH	69	I2C2RIP[1:0]	0	1	2	3	I2C2RIF	I2C2RIE
DMA_I2S2T 中断	FF:0233H	70	I2S2TIP[1:0]	0	1	2	3	I2S2TIF	I2S2TIE
DMA_I2S2R 中断	FF:023BH	71	I2S2RIP[1:0]	0	1	2	3	I2S2RIF	I2S2RIE
DMA_PWMAT 中断	FF:0243H	72	PWMATIP[1:0]	0	1	2	3	PWMATIF	PWMATIE
DMA_PWMAR 中断	FF:024BH	73	PWMARIP[1:0]	0	1	2	3	PWMARIF	PWMARIE
DMA_PWMCT 中断	FF:0253H	74	PWMCTIP[1:0]	0	1	2	3	PWMCTIF	PWMCTIE
DMA_PWMCR 中断	FF:025BH	75	PWMCRIP[1:0]	0	1	2	3	PWMCRIF	PWMCRIE
DMA_ADC2 中断	FF:0263H	76	ADC2IP[1:0]	0	1	2	3	ADC2IF	ADC2IE
DMA_DAC1 中断	FF:026BH	77	DAC1IP[1:0]	0	1	2	3	DAC1IF	DAC1IE
DMA_DAC2 中断	FF:0273H	78	DAC2IP[1:0]	0	1	2	3	DAC2IF	DAC2IE
DMA_SPI2 中断	FF:027BH	79	SPI2IP[1:0]	0	1	2	3	SPI2IF	SPI2IE
DMA_SPI3 中断	FF:0283H	80	SPI3IP[1:0]	0	1	2	3	SPI3IF	SPI3IE
DMA_UR5T 中断	FF:0293H	82	UR5TIP[1:0]	0	1	2	3	UR5TIF	UR5TIE
DMA_UR5R 中断	FF:029BH	83	UR5RIP[1:0]	0	1	2	3	UR5RIF	UR5RIE
DMA_UR6T 中断	FF:02A3H	84	UR6TIP[1:0]	0	1	2	3	UR5TIF	UR5TIE
DMA_UR6R 中断	FF:02ABH	85	UR6RIP[1:0]	0	1	2	3	UR6RIF	UR6RIE
DMA_UR7T 中断	FF:02B3H	86	UR7TIP[1:0]	0	1	2	3	UR7TIF	UR7TIE
DMA_UR7R 中断	FF:02BBH	87	UR7RIP[1:0]	0	1	2	3	UR7RIF	UR7RIE
DMA_UR8T 中断	FF:02C3H	88	UR8TIP[1:0]	0	1	2	3	UR8TIF	UR8TIE
DMA_UR8R 中断	FF:02CBH	89	UR8RIP[1:0]	0	1	2	3	UR8RIF	UR8RIE

中断源	中断向量	相同优先级内的查询次序	优先级设置	优先级 0（最低）	优先级 1	优先级 2	优先级 3（最高）	中断请求位	中断允许位
PA 中断	FF:02D3H	90	PIN2IPL[2], PIN2IPH[2]	0	1	2	3	PAINTF	PAINTE
PB 中断	FF:02DBH	91	PIN2IPL[3], PIN2IPH[3]	0	1	2	3	PBINTF	PBINTE
Timer5 中断	FF:0303H	96	PT5[1:0]	0	0	0	0	T5IF	ET5
Timer6 中断			PT6[1:0]	0	0	0	0	T6IF	ET6
Timer7 中断	FF:030BH	97	PT7[1:0]	0	0	0	0	T7IF	ET7
Timer8 中断			PT8[1:0]	0	0	0	0	T8IF	ET8
Timer9 中断	FF:0313H	98	PT9[1:0]	0	0	0	0	T9IF	ET9
Timer10 中断			PT10[1:0]	0	0	0	0	T10IF	ET10
Timer17 中断	FF:031BH	99	PT17[1:0]	0	0	0	0	T17IF	ET17
Timer18 中断			PT18[1:0]	0	0	0	0	T18IF	ET18
UART5 中断	FF:0333H	102	PS5[1:0]	0	1	2	3	S5RI || S5TI	ES5
UART6 中断	FF:033BH	103	PS6[1:0]	0	1	2	3	S6RI || S6TI	ES6
UART7 中断	FF:0343H	104	PS7[1:0]	0	1	2	3	S7RI || S7TI	ES7
UART8 中断	FF:034BH	105	PS8[1:0]	0	1	2	3	S8RI || S8TI	ES8
ADC2 中断	FF:0353H	106	PADC2[1:0]	0	1	2	3	ADC2_FLAG	EADC2
DAC1 中断	FF:035BH	107	DAC1IP[1:0]	0	1	2	3	DAC1IF	EDAC1I
DAC2 中断	FF:0363H	108	DAC2IP[1:0]	0	1	2	3	DAC2IF	EDAC2I
I2C2 中断	FF:036BH	109	PI2C2, PI2C2H	0	1	2	3	MSIF	EMSI
								STAIF	ESTAI
								RXIF	ERXI
								TXIF	ETXI
								STOIF	ESTOI
I2S2 中断	FF:0373H	110	PI2S2,PI2S2H	0	1	2	3	TXE	TXEIE
				0	1	2	3	RXNE	RXNEIE
				0	1	2	3	FRE	ERRIE
								OVR	
								UDR	
SPI2 中断	FF:037BH	111	PSPI2[1:0]	0	1	2	3	SP2IF	ESPI2
SPI3 中断	FF:0383H	112	PSPI3[1:0]	0	1	2	3	SP3IF	ESPI3
CMP2 中断	FF:0393H	114	CMP2IP[1:0]	0	1	2	3	CMP2IF	CMP2PIE |

中断源	中断向量	相同优先级内的查询次序	优先级设置	优先级 0（最低）	优先级 1	优先级 2	优先级 3（最高）	中断请求位	中断允许位
									CMP2NIE
CMP3 中断	FF:039BH	115	CMP3IP[1:0]	0	1	2	3	CMP2IF	CMP3PIE | CMP3NIE
CMP4 中断	FF:03A3H	116	CMP3IP[1:0]	0	1	2	3	CMP2IF	CMP4PIE | CMP4NIE
DMA_CAN1T 中断	FF:03ABH	117	CAN1TIP[1:0]	0	1	2	3	CAN1TIF	CAN1TIE
DMA_CAN1R 中断	FF:03B3H	118	CAN1RIP[1:0]	0	1	2	3	CAN1RIF	CAN1RIE
DMA_CAN2T 中断	FF:03BBH	119	CAN2TIP[1:0]	0	1	2	3	CAN2TIF	CAN2TIE
DMA_CAN2R 中断	FF:03C3H	120	CAN2RIP[1:0]	0	1	2	3	CAN2RIF	CAN2RIE
PWMC 中断	FF:03CBH	121	PPWMC, PPWMCH	0	1	2	3	PWMC_SR	PWMC_IER
PWMD 中断	FF:03D3H	122	PPWMD, PPWMDH	0	1	2	3	PWMD_SR	PWMD_IER
PWME 中断	FF:03DBH	123	PPWME, PPWMEH	0	1	2	3	PWME_SR	PWME_IER
PWMF 中断	FF:03E3H	124	PPWMF, PPWMFH	0	1	2	3	PWMF_SR	PWMF_IER
DMA_PWMET 中断	FF:03EBH	125	PWMETIP[1:0]	0	1	2	3	PWMETIF	PWMETIE
DMA_PWMER 中断	FF:03F3H	126	PWMERIP[1:0]	0	1	2	3	PWMERIF	PWMERIE

此处的查询次序是指：

同一个中断优先级中，如有多个中断源产生，内部硬件逻辑会按从 0～N 的顺序，优先响应查询次序小的中断源，即优先执行查询次序小的中断源的中断服务程序

在 C 语言中声明中断服务程序

```
void    INT0_Routine(void)      interrupt    0;
void    TM0_Rountine(void)      interrupt    1;
void    INT1_Routine(void)      interrupt    2;
void    TM1_Rountine(void)      interrupt    3;
void    UART1_Routine(void)     interrupt    4;
void    ADC_Routine(void)       interrupt    5;
void    LVD_Routine(void)       interrupt    6;
void    PCA_Routine(void)       interrupt    7;
void    UART2_Routine(void)     interrupt    8;
void    SPI_Routine(void)       interrupt    9;
void    INT2_Routine(void)      interrupt    10;
void    INT3_Routine(void)      interrupt    11;
void    TM2_Routine(void)       interrupt    12;
void    INT4_Routine(void)      interrupt    16;
void    UART3_Routine(void)     interrupt    17;
void    UART4_Routine(void)     interrupt    18;
void    TM3_Routine(void)       interrupt    19;
void    TM4_Routine(void)       interrupt    20;
```

```
void    CMP_Routine(void)     interrupt     21;
void    I2C_Routine(void)     interrupt     24;
void    USB_Routine(void)     interrupt     25;
void    PWMA_Routine(void)    interrupt     26;
void    PWMB_Routine(void)    interrupt     27;
```

中断号超过31的C语言中断服务程序不能直接用interrupt声明，请参考“开发环境的建立与ISP下载”章节中的“关于中断号大于31在Keil中编译出错的处理”小节的处理方法。汇编语言不受影响

15.4 中断相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
IE	中断允许寄存器	A8H	EA	ELVD	EADC	ES	ET1	EX1	ET0	EX0	0000,0000
IE2	中断允许寄存器 2	AFH	EUSB	ET4	ET3	ES4	ES3	ET2	ESPI	ES2	0000,0000
IP	中断优先级控制寄存器	B8H	-	PLVD	PADC	PS	PT1	PX1	PT0	PX0	x000,0000
IPH	高中断优先级控制寄存器	B7H	-	PLVDH	PADCH	PSH	PT1H	PX1H	PT0H	PX0H	x000,0000
IP2	中断优先级控制寄存器 2	B5H	PUSB	PI2C	PCMP	PX4	PPWMB	PPWMA	PSPI	PS2	0000,0000
IP2H	高中断优先级控制寄存器 2	B6H	PUSBH	PI2CH	PCMPH	PX4H	PPWMBH	PPWMAH	PSPIH	PS2H	0000,0000
IP3	中断优先级控制寄存器 3	DFH	PPWMD	PPWMC	PI2C2	PI2S2	PI2S	PRTC	PS4	PS3	0000,0000
IP3H	高中断优先级控制寄存器 3	EEH	PPWMDH	PPWMCH	PI2C2H	PI2S2H	PI2SH	PRTCH	PS4H	PS3H	0000,0000
IP4	中断优先级控制寄存器 4	F2H	-	-	-	-	-	-	PPWMF	PPWME	xxxx,xx00
IP4H	高中断优先级控制寄存器 4	F3H	-	-	-	-	-	-	PPWMFH	PPWMEH	xxxx,xx00
PCON	电源控制寄存器	87H	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL	0011,0000
TCON	定时器控制寄存器	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	0000,0000
INTCLKO	中断与时钟输出控制寄存器	8FH	-	EX4	EX3	EX2	-	T2CLKO	T1CLKO	T0CLKO	x000,x000
SCON	串口 1 控制寄存器	98H	SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI	0000,0000
S2CON	串口 2 控制寄存器	9AH	S2SM0/FE	S2SM1	S2SM2	S2REN	S2TB8	S2RB8	S2TI	S2RI	0000,0000
S3CON	串口 3 控制寄存器	ACH	S3SM0/FE	S3SM1	S3SM2	S3REN	S3TB8	S3RB8	S3TI	S3RI	0000,0000
ADC_CONTR	ADC 控制寄存器	BCH	ADC_POWER	ADC_START	ADC_FLAG	ADC_EPWMT	ADC_CHS[3:0]				0000,0000
SPSTAT	SPI 状态寄存器	CDH	SPIF	WCOL	-	-	-	-	-	-	00xx,xxxx
AUXINTIF	扩展外部中断标志寄存器	EFH	-	INT4IF	INT3IF	INT2IF	-	T4IF	T3IF	T2IF	x000,x000
LINICR	LINBUS 中断控制寄存器	F9H	PLIN2H	LIN2IF	LIN2IE	PLIN2L	PLINH	LINIF	LINIE	PLINL	0000,0000
S4CON	串口 4 控制寄存器	FDH	S4SM0/FE	S4SM1	S4SM2	S4REN	S4TB8	S4RB8	S4TI	S4RI	0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
LCMIFCFG	LCM 接口配置寄存器	7EFE50H	LCMIFIE	-	LCMIFIP[1:0]		LCMIFDPS[1:0]		D16_D8	M68_I80	0x00,0000
LCMIFSTA	LCM 接口状态寄存器	7EFE53H	-	-	-	-	-	-	-	LCMIFIF	xxxx,xxx0
RTCIEN	RTC 中断使能寄存器	7EFE62H	EALAI	EDAYI	EHOURI	EMINI	ESECI	ESEC2I	ESEC8I	ESEC32I	0000,0000
RTCIF	RTC 中断请求寄存器	7EFE63H	ALAIF	DAYIF	HOURIF	MINIF	SECIF	SEC2IF	SEC8IF	SEC32IF	0000,0000
T11CR	定时器 T11 控制寄存器	7EFE78H	T11R	T11_C/T	T11CLKO	T11x12	T11CS[1:0]		ET11I	T11IF	0000,0000
I2CMSCR	I²C 主机控制寄存器	7EFE81H	EI2CMSI	-	-	-	I2CMSCMD[3:0]				0xxx,0000
I2CMSST	I²C 主机状态寄存器	7EFE82H	I2CMSBUSY	I2CMSIF	-	-	-	-	I2CMSACKI	I2CMSACKO	00xx,xx10
I2CSLCR	I²C 从机控制寄存器	7EFE83H	-	EI2CSTAI	EI2CRXI	EI2CTXI	EI2CSTOI	-	-	I2CSLRST	x000,0xx0
I2CSLST	I²C 从机状态寄存器	7EFE84H	I2CSLBUSY	I2CSTAIF	I2CRXIF	I2CTXIF	I2CSTOIF	I2CTXING	I2CSLACKI	I2CSLACKO	0000,0000
INTXIT	INT2/3/4 中断模式选择寄存器	7EFEA9H	-	-	INT4IT[1:0]		INT3IT[1:0]		INT2IT[1:0]		xx00,0000
PWMA_IER	PWMA 中断使能寄存器	7EFEC4H	BIEA	TIEA	COMIEA	CCA4IE	CCA3IE	CCA2IE	CCA1IE	UIEA	0000,0000
PWMA_SR1	PWMA 状态寄存器 1	7EFEC5H	BIFA	TIFA	COMIFA	CCA4IF	CCA3IF	CCA2IF	CCA1IF	UIFA	0000,0000
PWMA_SR2	PWMA 状态寄存器 2	7EFEC6H	-	-	-	CCA4OF	CCA3OF	CCA2OF	CCA1OF	-	xxx0,000x
PWMB_IER	PWMB 中断使能寄存器	7EFEE4H	BIEB	TIEB	COMIEB	CCB8IE	CCB7IE	CCB6IE	CCB5IE	UIEB	0000,0000
PWMB_SR1	PWMB 状态寄存器 1	7EFEE5H	BIFB	TIFB	COMIFB	CCB8IF	CCB7IF	CCB6IF	CCB5IF	UIFB	0000,0000
PWMB_SR2	PWMB 状态寄存器 2	7EFEE6H	-	-	-	CCB8OF	CCB7OF	CCB6OF	CCB5OF	-	xxx0,000x

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P0INTE	P0 口中断使能寄存器	7EFD00H	P07INTE	P06INTE	P05INTE	P04INTE	P03INTE	P02INTE	P01INTE	P00INTE	0000,0000
P1INTE	P1 口中断使能寄存器	7EFD01H	P17INTE	P16INTE	P15INTE	P14INTE	P13INTE	P12INTE	P11INTE	P10INTE	0000,0000
P2INTE	P2 口中断使能寄存器	7EFD02H	P27INTE	P26INTE	P25INTE	P24INTE	P23INTE	P22INTE	P21INTE	P20INTE	0000,0000
P3INTE	P3 口中断使能寄存器	7EFD03H	P37INTE	P36INTE	P35INTE	P34INTE	P33INTE	P32INTE	P31INTE	P30INTE	0000,0000
P4INTE	P4 口中断使能寄存器	7EFD04H	P47INTE	P46INTE	P45INTE	P44INTE	P43INTE	P42INTE	P41INTE	P40INTE	0000,0000
P5INTE	P5 口中断使能寄存器	7EFD05H	P57INTE	P56INTE	P55INTE	P54INTE	P53INTE	P52INTE	P51INTE	P50INTE	0000,0000
P6INTE	P6 口中断使能寄存器	7EFD06H	P67INTE	P66INTE	P65INTE	P64INTE	P63INTE	P62INTE	P61INTE	P60INTE	0000,0000
P7INTE	P7 口中断使能寄存器	7EFD07H	P77INTE	P76INTE	P75INTE	P74INTE	P73INTE	P72INTE	P71INTE	P70INTE	0000,0000
P8INTE	P8 口中断使能寄存器	7EFD08H	P87INTE	P86INTE	P85INTE	P84INTE	P83INTE	P82INTE	P81INTE	P80INTE	0000,0000
P9INTE	P9 口中断使能寄存器	7EFD09H	P97INTE	P96INTE	P95INTE	P94INTE	P93INTE	P92INTE	P91INTE	P90INTE	0000,0000
PAINTE	PA 口中断使能寄存器	7EFD0AH	PA7INTE	PA6INTE	PA5INTE	PA4INTE	PA3INTE	PA2INTE	PA1INTE	PA0INTE	0000,0000
PBINTE	PB 口中断使能寄存器	7EFD0BH	PB7INTE	PB6INTE	PB5INTE	PB4INTE	PB3INTE	PB2INTE	PB1INTE	PB0INTE	0000,0000
P0INTF	P0 口中断标志寄存器	7EFD10H	P07INTF	P06INTF	P05INTF	P04INTF	P03INTF	P02INTF	P01INTF	P00INTF	0000,0000
P1INTF	P1 口中断标志寄存器	7EFD11H	P17INTF	P16INTF	P15INTF	P14INTF	P13INTF	P12INTF	P11INTF	P10INTF	0000,0000
P2INTF	P2 口中断标志寄存器	7EFD12H	P27INTF	P26INTF	P25INTF	P24INTF	P23INTF	P22INTF	P21INTF	P20INTF	0000,0000
P3INTF	P3 口中断标志寄存器	7EFD13H	P37INTF	P36INTF	P35INTF	P34INTF	P33INTF	P32INTF	P31INTF	P30INTF	0000,0000
P4INTF	P4 口中断标志寄存器	7EFD14H	P47INTF	P46INTF	P45INTF	P44INTF	P43INTF	P42INTF	P41INTF	P40INTF	0000,0000
P5INTF	P5 口中断标志寄存器	7EFD15H	P57INTF	P56INTF	P55INTF	P54INTF	P53INTF	P52INTF	P51INTF	P50INTF	0000,0000
P6INTF	P6 口中断标志寄存器	7EFD16H	P67INTF	P66INTF	P65INTF	P64INTF	P63INTF	P62INTF	P61INTF	P60INTF	0000,0000
P7INTF	P7 口中断标志寄存器	7EFD17H	P77INTF	P76INTF	P75INTF	P74INTF	P73INTF	P72INTF	P71INTF	P70INTF	0000,0000
P8INTF	P8 口中断标志寄存器	7EFD18H	P87INTF	P86INTF	P85INTF	P84INTF	P83INTF	P82INTF	P81INTF	P80INTF	0000,0000
P9INTF	P9 口中断标志寄存器	7EFD19H	P97INTF	P96INTF	P95INTF	P94INTF	P93INTF	P92INTF	P91INTF	P90INTF	0000,0000
PAINTF	PA 口中断标志寄存器	7EFD1AH	PA7INTF	PA6INTF	PA5INTF	PA4INTF	PA3INTF	PA2INTF	PA1INTF	PA0INTF	0000,0000
PBINTF	PB 口中断标志寄存器	7EFD1BH	PB7INTF	PB6INTF	PB5INTF	PB4INTF	PB3INTF	PB2INTF	PB1INTF	PB0INTF	0000,0000
PINIPL	I/O 口中断优先级寄存器	7EFD60H	P7IP	P6IP	P5IP	P4IP	P3IP	P2IP	P1IP	P0IP	0000,0000
PINIPH	I/O 口中断优先级寄存器	7EFD61H	P7IPH	P6IPH	P5IPH	P4IPH	P3IPH	P2IPH	P1IPH	P0IPH	0000,0000
PIN2IPL	I/O 口中断优先级寄存器 2	7EFD62H	-	-	-	-	PBIP	PAIP	P9IP	P8IP	xxxx,0000
PIN2IPH	I/O 口中断优先级寄存器 2	7EFD63H	-	-	-	-	PBIPH	PAIPH	P9IPH	P8IPH	xxxx,0000
UR1TOCR	串口 1 接收超时控制寄存器	7EFD70H	UR1ENTO	UR1ENTOI	UR1SCALE	-	-	-	-	-	000x,xxxx
UR1TOSR	串口 1 接收超时状态寄存器	7EFD71H	-	-	-	-	-	-	-	UR1TOIF	xxxx,xxx0
UR2TOCR	串口 2 接收超时控制寄存器	7EFD74H	UR2ENTO	UR2ENTOI	UR2SCALE	-	-	-	-	-	000x,xxxx
UR2TOSR	串口 2 接收超时状态寄存器	7EFD75H	-	-	-	-	-	-	-	UR2TOIF	xxxx,xxx0
UR3TOCR	串口 3 接收超时控制寄存器	7EFD78H	UR3ENTO	UR3ENTOI	UR3SCALE	-	-	-	-	-	000x,xxxx
UR3TOSR	串口 3 接收超时状态寄存器	7EFD79H	-	-	-	-	-	-	-	UR3TOIF	xxxx,xxx0
UR4TOCR	串口 4 接收超时控制寄存器	7EFD7CH	UR4ENTO	UR4ENTOI	UR4SCALE	-	-	-	-	-	000x,xxxx
UR4TOSR	串口 4 接收超时状态寄存器	7EFD7DH	-	-	-	-	-	-	-	UR4TOIF	xxxx,xxx0
SPITOCR	SPI 从机超时控制寄存器	7EFD80H	SPIENTO	SPIENTOI	SPISCALE	-	-	-	-	-	000x,xxxx
SPITOSR	SPI 从机超时状态寄存器	7EFD81H	SPICTOIF	-	-	-	-	-	-	SPITOIF	0xxx,xxx0
I2CTOCR	I2C 从机超时控制寄存器	7EFD84H	I2CENTO	I2CENTOI	I2CSCALE	-	-	-	-	-	000x,xxxx
I2CTOSR	I2C 从机超时状态寄存器	7EFD85H	I2CCTOF	-	-	-	-	-	-	I2CTOIF	xxxx,xxx0
I2S2CR	I2S2 控制寄存器	7EFD90H	I2S2TXEIE	I2S2RXNEIE	I2S2ERRIE	I2S2FRF	-	-	I2S2TXDMAEN	I2S2RXDMAEN	0000,xx00

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
I2S2SR	I2S2 状态寄存器	7EFD91H	-	I2S2FRE	I2S2BUY	I2S2OVR	I2S2UDR	I2S2CHSID	I2S2TXE	I2S2RXNE	x000,0000
I2SCR	I2S 控制寄存器	7EFD98H	I2STXEIE	I2SRXNEIE	I2SERRIE	I2SFRF	-	-	I2STXDMAEN	I2SRXDMAEN	0000,xx00
I2SSR	I2S 状态寄存器	7EFD99H	-	I2SFRE	I2SBUY	I2SOVR	I2SUDR	I2SCHSID	I2STXE	I2SRXNE	x000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
ADC2_CONTR	ADC2 控制寄存器	7EFB38H	ADC2_POWER	ADC2_START	ADC2_FLAG	ADC2_EPWMT	ADC2_CHS[3:0]				0000,0000
ADC2INTR	ADC2 中断控制寄存器	7EFB3EH	-	-	-	-	-	PADC2[1:0]		EADC2	xxxx,x000
CMP1_6P6N_CR1	比较器 1 控制寄存器 1	7EFB70H	CMP1EN	CMP1IF	CMP1PIE	CMP1NIE	CMP1IP[1:0]		CMP1OE	CMP1RES	0000,0000
CMP2_6P6N_CR1	比较器 2 控制寄存器 1	7EFB74H	CMP2EN	CMP2IF	CMP2PIE	CMP2NIE	CMP2IP[1:0]		CMP2OE	CMP2RES	0000,0000
CMP3_6P6N_CR1	比较器 3 控制寄存器 1	7EFB78H	CMP3EN	CMP3IF	CMP3PIE	CMP3NIE	CMP3IP[1:0]		CMP3OE	CMP3RES	0000,0000
CMP4_6P6N_CR1	比较器 4 控制寄存器 1	7EFB7CH	CMP4EN	CMP4IF	CMP4PIE	CMP4NIE	CMP4IP[1:0]		CMP4OE	CMP4RES	0000,0000
T6T5CFG	定时器 6/5 配置寄存器	7EFBB2H	T6IF	PT6[1:0]		ET6	T5IF	PT5[1:0]		ET5	0000,0000
T8T7CFG	定时器 8/7 配置寄存器	7EFBBAH	T8IF	PT8[1:0]		ET8	T7IF	PT7[1:0]		ET7	0000,0000
T10T9CFG	定时器 10/9 配置寄存器	7EFBC2H	T10IF	PT10[1:0]		ET10	T9IF	PT9[1:0]		ET9	0000,0000
T18T17CFG	定时器 181/7 配置寄存器	7EFBCAH	T18IF	PT18[1:0]		ET18	T17IF	PT17[1:0]		ET17	0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_M2M_CFG	M2M_DMA 配置寄存器	7EFA00H	M2MIE	-	TXACO	RXACO	M2MIP[1:0]		M2MPTY[1:0]		0x00,0000
DMA_M2M_STA	M2M_DMA 状态寄存器	7EFA02H	-	-	-	-	-	-	-	M2MIF	xxxx,xxx0
DMA_ADC_CFG	ADC_DMA 配置寄存器	7EFA10H	ADCIE	-	-	-	ADCMIP[1:0]		ADCPTY[1:0]		0xxx,0000
DMA_ADC_STA	ADC_DMA 状态寄存器	7EFA12H	-	-	-	-	-	-	-	ADCIF	xxxx,xxx0
DMA_SPI_CFG	SPI_DMA 配置寄存器	7EFA20H	SPIIE	ACT_TX	ACT_RX	-	SPIIP[1:0]		SPIPTY[1:0]		000x,0000
DMA_SPI_STA	SPI_DMA 状态寄存器	7EFA22H	-	-	-	-	-	TXOVW	RXLOSS	SPIIF	xxxx,x000
DMA_UR1T_CFG	UR1T_DMA 配置寄存器	7EFA30H	UR1TIE	-	-	-	UR1TIP[1:0]		UR1TPTY[1:0]		0xxx,0000
DMA_UR1T_STA	UR1T_DMA 状态寄存器	7EFA32H	-	-	-	-	-	TXOVW	-	UR1TIF	xxxx,x0x0
DMA_UR1R_CFG	UR1R_DMA 配置寄存器	7EFA38H	UR1RIE	-	-	-	UR1RIP[1:0]		UR1RPTY[1:0]		0xxx,0000
DMA_UR1R_STA	UR1R_DMA 状态寄存器	7EFA3AH	-	-	-	-	-	-	RXLOSS	UR1RIF	xxxx,xx00
DMA_UR2T_CFG	UR2T_DMA 配置寄存器	7EFA40H	UR2TIE	-	-	-	UR2TIP[1:0]		UR2TPTY[1:0]		0xxx,0000
DMA_UR2T_STA	UR2T_DMA 状态寄存器	7EFA42H	-	-	-	-	-	TXOVW	-	UR2TIF	xxxx,x0x0
DMA_UR2R_CFG	UR2R_DMA 配置寄存器	7EFA48H	UR2RIE	-	-	-	UR2RIP[1:0]		UR2RPTY[1:0]		0xxx,0000
DMA_UR2R_STA	UR2R_DMA 状态寄存器	7EFA4AH	-	-	-	-	-	-	RXLOSS	UR2RIF	xxxx,xx00
DMA_UR3T_CFG	UR3T_DMA 配置寄存器	7EFA50H	UR3TIE	-	-	-	UR3TIP[1:0]		UR3TPTY[1:0]		0xxx,0000
DMA_UR3T_STA	UR3T_DMA 状态寄存器	7EFA52H	-	-	-	-	-	TXOVW	-	UR3TIF	xxxx,x0x0
DMA_UR3R_CFG	UR3R_DMA 配置寄存器	7EFA58H	UR3RIE	-	-	-	UR3RIP[1:0]		UR3RPTY[1:0]		0xxx,0000
DMA_UR3R_STA	UR3R_DMA 状态寄存器	7EFA5AH	-	-	-	-	-	-	RXLOSS	UR3RIF	xxxx,xx00
DMA_UR4T_CFG	UR4T_DMA 配置寄存器	7EFA60H	UR4TIE	-	-	-	UR4TIP[1:0]		UR4TPTY[1:0]		0xxx,0000
DMA_UR4T_STA	UR4T_DMA 状态寄存器	7EFA62H	-	-	-	-	-	TXOVW	-	UR4TIF	xxxx,x0x0
DMA_UR4R_CFG	UR4R_DMA 配置寄存器	7EFA68H	UR4RIE	-	-	-	UR4RIP[1:0]		UR4RPTY[1:0]		0xxx,0000
DMA_UR4R_STA	UR4R_DMA 状态寄存器	7EFA6AH	-	-	-	-	-	-	RXLOSS	UR4RIF	xxxx,xx00
DMA_LCM_CFG	LCM_DMA 配置寄存器	7EFA70H	LCMIE	-	-	-	LCMIP[1:0]		LCMPTY[1:0]		0xxx,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_LCM_STA	LCM_DMA 状态寄存器	7EFA72H	-	-	-	-	-	-	TXOVW	LCMIF	xxxx,xx00
DMA_I2CT_CFG	I2CT_DMA 配置寄存器	7EFA98H	I2CTIE	-	-	-	I2CTIP[1:0]		I2CTPTY[1:0]		0xxx,0000
DMA_I2CT_STA	I2CT_DMA 状态寄存器	7EFA9AH	-	-	-	-	-	TXOVW	-	I2CTIF	xxxx,x0x0
DMA_I2CR_CFG	I2CR_DMA 配置寄存器	7EFAA0H	I2CRIE	-	-	-	I2CRIP[1:0]		I2CRPTY[1:0]		0xxx,0000
DMA_I2CR_STA	I2CR_DMA 状态寄存器	7EFAA2H	-	-	-	-	-	-	RXLOSS	I2CRIF	xxxx,xx00
DMA_I2ST_CFG	I2ST_DMA 配置寄存器	7EFAB0H	I2STIE	-	-	-	I2STIP[1:0]		I2STPTY[1:0]		0xxx,0000
DMA_I2ST_STA	I2ST_DMA 状态寄存器	7EFAB2H	-	-	-	-	-	TXOVW	-	I2STIF	xxxx,x0x0
DMA_I2SR_CFG	I2SR_DMA 配置寄存器	7EFAB8H	I2SRIE	-	-	-	I2SRIP[1:0]		I2SRPTY[1:0]		0xxx,0000
DMA_I2SR_STA	I2SR_DMA 状态寄存器	7EFABAH	-	-	-	-	-	-	RXLOSS	I2SRIF	xxxx,xx00
DMA_QSPI_CFG	QSPI_DMA 配置寄存器	7EFAD0H	QSPIIE	ACT_WR	ACT_RD	-	QSPIIP[1:0]		QSPIPTY[1:0]		000x,0000
DMA_QSPI_STA	QSPI_DMA 状态寄存器	7EFAD2H	-	-	-	-	-	-	-	QSPIIF	xxxx,xxx0
DMA_ADC2_CFG	ADC2_DMA 配置寄存器	7EFAE0H	ADC2IE	-	-	-	ADC2MIP[1:0]		ADC2PTY[1:0]		0xxx,0000
DMA_ADC2_STA	ADC2_DMA 状态寄存器	7EFAE2H	-	-	-	-	-	-	-	ADC2IF	xxxx,xxx0

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
QSPI_CR3	QSPI 控制寄存器 3	7EF902H	PMM	APMS	-	-	SMIE	FTIE	TCIE	TEIE	00xx,0000
QSPI_DCR1	QSPI 器件配置寄存器 1	7EF904H	-	CSHT[2:0]			QSPI_IP[1:0]		-	CKMODE	x000,00x0
QSPI_SR1	QSPI 状态寄存器 1	7EF906H	-	-	BUSY	-	SMF	FTF	TCF	TEF	xx0x,0100
DMA_PWMAT_CFG	PWMAT_DMA 配置寄存器	7EF980H	PWMATIE	-	-	-	PWMATIP[1:0]		PWMATPTY[1:0]		0xxx,0000
DMA_PWMAT_STA	PWMAT_DMA 状态寄存器	7EF982H	RDOV	-	-	-	-	TXOVW	-	PWMATIF	0xxx,x0x0
DMA_PWMAR_CFG	PWMAR_DMA 配置寄存器	7EF990H	PWMARIE	-	-	-	PWMARIP[1:0]		PWMARPTY[1:0]		0xxx,0000
DMA_PWMAR_STA	PWMAR_DMA 状态寄存器	7EF992H	WROV	-	-	-	-	-	RXLOSS	PWMARIF	0xxx,xx00
DMA_PWMCT_CFG	PWMCT_DMA 配置寄存器	7EF9A0H	PWMCTIE	-	-	-	PWMCTIP[1:0]		PWMCTPTY[1:0]		0xxx,0000
DMA_PWMCT_STA	PWMCT_DMA 状态寄存器	7EF9A2H	RDOV	-	-	-	-	TXOVW	-	PWMCTIF	0xxx,x0x0
DMA_PWMCR_CFG	PWMCR_DMA 配置寄存器	7EF9B0H	PWMCRIE	-	-	-	PWMCRIP[1:0]		PWMCRPTY[1:0]		0xxx,0000
DMA_PWMCR_STA	PWMCR_DMA 状态寄存器	7EF9B2H	WROV	-	-	-	-	-	RXLOSS	PWMCRIF	0xxx,xx00

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
SPI2STAT	SPI2 状态寄存器	7EF801H	SP2IF	SP2WCOL	-	-	-	-	-	-	00xx,xxxx
SPI2TOCR	SPI2 从机超时控制寄存器	7EF804H	SPI2ENTO	SPI2ENTOI	SPI2SCALE	-	-	-	-	-	000x,xxxx
SPI2TOSR	SPI2 从机超时状态寄存器	7EF805H	SPI2CTOIF	-	-	-	-	-	-	SPI2TOIF	0xxx,xxx0
SPI2_INTR	SPI2 中断控制寄存器	7EF80FH	-	-	PSPI2[1:0]		-	-	-	ESPI2	xx00,xxx0
DMA_SPI2_CFG	SPI2_DMA 配置寄存器	7EF810H	SPI2IE	ACT_TX	ACT_RX	-	SPI2IP[1:0]		SPI2PTY[1:0]		000x,0000
DMA_SPI2_STA	SPI2_DMA 状态寄存器	7EF812H	-	-	-	-	-	TXOVW	RXLOSS	SPI2IF	xxxx,x000
SPI3STAT	SPI3 状态寄存器	7EF821H	SP3IF	SP3WCOL	-	-	-	-	-	-	00xx,xxxx
SPI3TOCR	SPI3 从机超时控制寄存器	7EF824H	SPI3ENTO	SPI3ENTOI	SPI3SCALE	-	-	-	-	-	000x,xxxx
SPI3TOSR	SPI3 从机超时状态寄存器	7EF825H	SPI3CTOIF	-	-	-	-	-	-	SPI3TOIF	0xxx,xxx0
SPI3_INTR	SPI3 中断控制寄存器	7EF82FH	-	-	PSPI3[1:0]		-	-	-	ESPI3	xx00,xxx0

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_SPI3_CFG	SPI3_DMA 配置寄存器	7EF830H	SPI3IE	ACT_TX	ACT_RX	-	SPI3IP[1:0]		SPI3PTY[1:0]		000x,0000
DMA_SPI3_STA	SPI3_DMA 状态寄存器	7EF832H	-	-	-	-	-	TXOVW	RXLOSS	SPI3IF	xxxx,x000
I2C2MSCR	I²C 主机控制寄存器	7EF861H	EI2C2MSI	-	-	-	I2C2MSCMD[3:0]				0xxx,0000
I2C2MSST	I²C 主机状态寄存器	7EF862H	I2C2MSBUSY	I2C2MSIF	-	-	-	-	I2C2MSACKI	I2C2MSACKO	00xx,xx10
I2C2SLCR	I²C 从机控制寄存器	7EF863H	-	EI2C2STAI	EI2C2RXI	EI2C2TXI	EI2C2STOI	-	-	I2C2SLRST	x000,0xx0
I2C2SLST	I²C 从机状态寄存器	7EF864H	I2C2SLBUSY	I2C2STAIF	I2C2RXIF	I2C2TXIF	I2C2STOIF	I2C2TXING	I2C2SLACKI	I2C2SLACKO	0000,0000
I2C2TOCR	I²C2 从机超时控制寄存器	7EF86CH	I2C2ENTO	I2C2ENTOI	I2C2SCALE	-	-	-	-	-	000x,xxxx
I2C2TOSR	I²C2 从机超时状态寄存器	7EF86DH	-	-	-	-	-	-	-	I2C2TOIF	xxxx,xxx0
DMA_I2C2T_CFG	I2C2T_DMA 配置寄存器	7EF870H	I2C2TIE	-	-	-	I2C2TIP[1:0]		I2C2TPTY[1:0]		0xxx,0000
DMA_I2C2T_STA	I2C2T_DMA 状态寄存器	7EF872H	-	-	-	-	-	TXOVW	-	I2C2TIF	xxxx,x0x0
DMA_I2C2R_CFG	I2C2R_DMA 配置寄存器	7EF880H	I2C2RIE	-	-	-	I2C2RIP[1:0]		I2C2RPTY[1:0]		0xxx,0000
DMA_I2C2R_STA	I2C2R_DMA 状态寄存器	7EF882H	-	-	-	-	-	-	RXLOSS	I2C2RIF	xxxx,xx00
DMA_I2S2T_CFG	I2S2T_DMA 配置寄存器	7EF890H	I2S2TIE	-	-	-	I2S2TIP[1:0]		I2S2TPTY[1:0]		0xxx,0000
DMA_I2S2T_STA	I2S2T_DMA 状态寄存器	7EF892H	-	-	-	-	-	TXOVW	-	I2S2TIF	xxxx,x0x0
DMA_I2S2R_CFG	I2S2R_DMA 配置寄存器	7EF8A0H	I2S2RIE	-	-	-	I2S2RIP[1:0]		I2S2RPTY[1:0]		0xxx,0000
DMA_I2S2R_STA	I2S2R_DMA 状态寄存器	7EF8A2H	-	-	-	-	-	-	RXLOSS	I2S2RIF	xxxx,xx00
PWMC_IER	PWMC 中断使能寄存器	7EF8C4H	BIEC	TIEC	COMIEC	CCC4IE	CCC3IE	CCC2IE	CCC1IE	UIEC	0000,0000
PWMC_SR1	PWMC 状态寄存器 1	7EF8C5H	BIFC	TIFC	COMIFC	CCC4IF	CCC3IF	CCC2IF	CCC1IF	UIFC	0000,0000
PWMC_SR2	PWMC 状态寄存器 2	7EF8C6H	-	-	-	CCC4OF	CCC3OF	CCC2OF	CCC1OF	-	xxx0,000x
PWMD_IER	PWMD 中断使能寄存器	7EF8E4H	BIED	TIED	COMIED	CCD8IE	CCD7IE	CCD6IE	CCD5IE	UIED	0000,0000
PWMD_SR1	PWMD 状态寄存器 1	7EF8E5H	BIFD	TIFD	COMIFD	CCD8IF	CCD7IF	CCD6IF	CCD5IF	UIFD	0000,0000
PWMD_SR2	PWMD 状态寄存器 2	7EF8E6H	-	-	-	CCD8OF	CCD7OF	CCD6OF	CCD5OF	-	xxx0,000x

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S5CON	串口 5 控制寄存器	7EF740H	S5SM0/FE	S5SM1	S5SM2	S5REN	S5TB8	S5RB8	S5TI	S5RI	0000,0000
S5INTR	串口 5 中断寄存器	7EF742H	ES5	-	PS5[1:0]		-	-	-	-	0x00,xxxx
UR5TOCR	串口 5 接收超时控制寄存器	7EF744H	UR5ENTO	UR5ENTOI	UR5SCALE	-	-	-	-	-	000x,xxxx
UR5TOSR	串口 5 接收超时状态寄存器	7EF745H	-	-	-	-	-	-	-	UR5TOIF	xxxx,xxx0
DMA_UR5T_CFG	UR5T_DMA 配置寄存器	7EF750H	UR5TIE	-	-	-	UR5TIP[1:0]		UR5TPTY[1:0]		0xxx,0000
DMA_UR5T_STA	UR5T_DMA 状态寄存器	7EF752H	-	-	-	-	-	TXOVW	-	UR5TIF	xxxx,x0x0
DMA_UR5R_CFG	UR5R_DMA 配置寄存器	7EF760H	UR5RIE	-	-	-	UR5RIP[1:0]		UR5RPTY[1:0]		0xxx,0000
DMA_UR5R_STA	UR5R_DMA 状态寄存器	7EF762H	-	-	-	-	-	-	RXLOSS	UR5RIF	xxxx,xx00
S6CON	串口 6 控制寄存器	7EF770H	S6SM0/FE	S6SM1	S6SM2	S6REN	S6TB8	S6RB8	S6TI	S6RI	0000,0000
S6INTR	串口 6 中断寄存器	7EF772H	ES6	-	PS6[1:0]		-	-	-	-	0x00,xxxx
UR6TOCR	串口 6 接收超时控制寄存器	7EF774H	UR6ENTO	UR6ENTOI	UR6SCALE	-	-	-	-	-	000x,xxxx
UR6TOSR	串口 6 接收超时状态寄存器	7EF775H	-	-	-	-	-	-	-	UR6TOIF	xxxx,xxx0
DMA_UR6T_CFG	UR6T_DMA 配置寄存器	7EF780H	UR6TIE	-	-	-	UR6TIP[1:0]		UR6TPTY[1:0]		0xxx,0000
DMA_UR6T_STA	UR6T_DMA 状态寄存器	7EF782H	-	-	-	-	-	TXOVW	-	UR6TIF	xxxx,x0x0
DMA_UR6R_CFG	UR6R_DMA 配置寄存器	7EF790H	UR6RIE	-	-	-	UR6RIP[1:0]		UR6RPTY[1:0]		0xxx,0000
DMA_UR6R_STA	UR6R_DMA 状态寄存器	7EF792H	-	-	-	-	-	-	RXLOSS	UR6RIF	xxxx,xx00
S7CON	串口 7 控制寄存器	7EF7A0H	S7SM0/FE	S7SM1	S7SM2	S7REN	S7TB8	S7RB8	S7TI	S7RI	0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S7INTR	串口 7 中断寄存器	7EF7A2H	ES7	-	PS7[1:0]		-	-	-	-	0x00,xxxx
UR7TOCR	串口 7 接收超时控制寄存器	7EF7A4H	UR7ENTO	UR7ENTOI	UR7SCALE	-	-	-	-	-	000x,xxxx
UR7TOSR	串口 7 接收超时状态寄存器	7EF7A5H	-	-	-	-	-	-	-	UR7TOIF	xxxx,xxx0
DMA_UR7T_CFG	UR7T_DMA 配置寄存器	7EF7B0H	UR7TIE	-	-	-	UR7TIP[1:0]		UR7TPTY[1:0]		0xxx,0000
DMA_UR7T_STA	UR7T_DMA 状态寄存器	7EF7B2H	-	-	-	-	-	TXOVW	-	UR7TIF	xxxx,x0x0
DMA_UR7R_CFG	UR7R_DMA 配置寄存器	7EF7C0H	UR7RIE	-	-	-	UR7RIP[1:0]		UR7RPTY[1:0]		0xxx,0000
DMA_UR7R_STA	UR7R_DMA 状态寄存器	7EF7C2H	-	-	-	-	-	-	RXLOSS	UR7RIF	xxxx,xx00
S8CON	串口 8 控制寄存器	7EF7D0H	S8SM0/FE	S8SM1	S8SM2	S8REN	S8TB8	S8RB8	S8TI	S8RI	0000,0000
S8INTR	串口 8 中断寄存器	7EF7D2H	ES8	-	PS8[1:0]		-	-	-	-	0x00,xxxx
UR8TOCR	串口 8 接收超时控制寄存器	7EF7D4H	UR8ENTO	UR8ENTOI	UR8SCALE	-	-	-	-	-	000x,xxxx
UR8TOSR	串口 8 接收超时状态寄存器	7EF7D5H	-	-	-	-	-	-	-	UR8TOIF	xxxx,xxx0
DMA_UR8T_CFG	UR8T_DMA 配置寄存器	7EF7E0H	UR8TIE	-	-	-	UR8TIP[1:0]		UR8TPTY[1:0]		0xxx,0000
DMA_UR8T_STA	UR8T_DMA 状态寄存器	7EF7E2H	-	-	-	-	-	TXOVW	-	UR8TIF	xxxx,x0x0
DMA_UR8R_CFG	UR8R_DMA 配置寄存器	7EF7F0H	UR8RIE	-	-	-	UR8RIP[1:0]		UR8RPTY[1:0]		0xxx,0000
DMA_UR8R_STA	UR8R_DMA 状态寄存器	7EF7F2H	-	-	-	-	-	-	RXLOSS	UR8RIF	xxxx,xx00

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DAC1_CR	DAC1 控制寄存器	7EF610H	DMAEN	DAC1TG	DAC1TS[1:0]		DAC1IP[1:0]		EDAC1I	DAC1EN	0000,0000
DAC1_STA	DAC1 状态寄存器	7EF611H	BUSY	-	-	-	-	-	-	DAC1IF	0xxx,xxx0
DMA_DAC1_CFG	DAC1_DMA 配置寄存器	7EF618H	DAC1IE	-	-	-	DAC1IP[1:0]		DAC1PTY[1:0]		0xxx,0000
DMA_DAC1_STA	DAC1_DMA 状态寄存器	7EF61AH	-	-	-	-	-	TXOVW	-	DAC1IF	xxxx,x0x0
DAC2_CR	DAC2 控制寄存器	7EF628H	DMAEN	DAC2TG	DAC2TS[1:0]		DAC2IP[1:0]		EDAC2I	DAC2EN	0000,0000
DAC2_STA	DAC2 状态寄存器	7EF629H	BUSY	-	-	-	-	-	-	DAC2IF	0xxx,xxx0
DMA_DAC2_CFG	DAC2_DMA 配置寄存器	7EF630H	DAC2IE	-	-	-	DAC2IP[1:0]		DAC2PTY[1:0]		0xxx,0000
DMA_DAC2_STA	DAC2_DMA 状态寄存器	7EF632H	-	-	-	-	-	TXOVW	-	DAC2IF	xxxx,x0x0
DMA_PWMET_CFG	PWMET_DMA 配置寄存器	7EF668H	PWMETIE	-	-	-	PWMETIP[1:0]		PWMETPTY[1:0]		0xxx,0000
DMA_PWMET_STA	PWMET_DMA 状态寄存器	7EF66AH	RDOV	-	-	-	-	TXOVW	-	PWMETIF	0xxx,x0x0
DMA_PWMER_CFG	PWMER_DMA 配置寄存器	7EF678H	PWMERIE	-	-	-	PWMERIP[1:0]		PWMERPTY[1:0]		0xxx,0000
DMA_PWMER_STA	PWMER_DMA 状态寄存器	7EF67AH	WROV	-	-	-	-	-	RXLOSS	PWMERIF	0xxx,xx00
PWME_IER	PWME 中断使能寄存器	7EF6C4H	BIEE	TIEE	COMIEE	CCE4IE	CCE3IE	CCE2IE	CCE1IE	UIEE	0000,0000
PWME_SR1	PWME 状态寄存器 1	7EF6C5H	BIFE	TIFE	COMIFE	CCE4IF	CCE3IF	CEC2IF	CCE1IF	UIFE	0000,0000
PWME_SR2	PWME 状态寄存器 2	7EF6C6H	-	-	-	CCE4OF	CCE3OF	CCE2OF	CCE1OF	-	xxx0,000x
PWMF_IER	PWMF 中断使能寄存器	7EF6E4H	BIEF	TIEF	COMIEF	CCF8IE	CCF7IE	CCF6IE	CCF5IE	UIEF	0000,0000
PWMF_SR1	PWMF 状态寄存器 1	7EF6E5H	BIFF	TIFF	COMIFF	CCF8IF	CCF7IF	CCF6IF	CCF5IF	UIFF	0000,0000
PWMF_SR2	PWMF 状态寄存器 2	7EF6E6H	-	-	-	CCF8OF	CCF7OF	CCF6OF	CCF5OF	-	xxx0,000x

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_CAN1T_CFG	CAN1T_DMA 配置寄存器	7EF520H	CAN1TIE	-	-	-	CAN1TIP[1:0]		CAN1TPTY[1:0]		0xxx,0000
DMA_CAN1T_STA	CAN1T_DMA 状态寄存器	7EF522H	-	-	-	-	-	TXOVW	-	CAN1TIF	xxxx,x0x0
DMA_CAN1R_CFG	CAN1R_DMA 配置寄存器	7EF530H	CAN1RIE	-	-	-	CAN1RIP[1:0]		CAN1RPTY[1:0]		0xxx,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_CAN1R_STA	CAN1R_DMA 状态寄存器	7EF532H	-	-	-	-	-	-	RXLOSS	CAN1RIF	xxxx,xx00
DMA_CAN2T_CFG	CAN2T_DMA 配置寄存器	7EF540H	CAN2TIE	-	-	-	CAN2TIP[1:0]		CAN2TPTY[1:0]		0xxx,0000
DMA_CAN2T_STA	CAN2T_DMA 状态寄存器	7EF542H	-	-	-	-	-	TXOVW	-	CAN2TIF	xxxx,x0x0
DMA_CAN2R_CFG	CAN2R_DMA 配置寄存器	7EF550H	CAN2RIE	-	-	-	CAN2RIP[1:0]		CAN2RPTY[1:0]		0xxx,0000
DMA_CAN2R_STA	CAN2R_DMA 状态寄存器	7EF552H	-	-	-	-	-	-	RXLOSS	CAN2RIF	xxxx,xx00

15.5 总中断控制

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IE	A8H	**EA**	ELVD	EADC	ES	ET1	EX1	ET0	EX0

EA：总中断允许控制位。EA 的作用是使中断允许形成多级控制。

即各中断源首先受 EA 控制;其次还受各中断源自己的中断允许控制位控制。

0：CPU 屏蔽所有的中断申请

1：CPU 开放中断

15.6 外部中断

15.6.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
INT0/P3.2 中断	FF:0003H	0
INT1/P3.3 中断	FF:0013H	2
INT2/P3.6 中断	FF:0053H	10
INT3/P3.7 中断	FF:005BH	11
INT4/P3.0 中断	FF:0083H	16

15.6.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IE	A8H	EA	ELVD	EADC	ES	ET1	**EX1**	ET0	**EX0**

EX1：外部中断 1 中断允许位。

0：禁止 INT1 中断

1：允许 INT1 中断

EX0：外部中断 0 中断允许位。

0：禁止 INT0 中断

1：允许 INT0 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTCLKO	8FH	-	**EX4**	**EX3**	**EX2**	-	T2CLKO	T1CLKO	T0CLKO

EX4：外部中断 4 中断允许位。

0：禁止 INT4 中断

1：允许 INT4 中断

EX3：外部中断 3 中断允许位。

0：禁止 INT3 中断

1：允许 INT3 中断

EX2：外部中断 2 中断允许位。

0：禁止 INT2 中断

1：允许 INT2 中断

15.6.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TCON	88H	TF1	TR1	TF0	TR0	**IE1**	IT1	**IE0**	IT0

IE1：外部中断1中断请求标志。中断服务程序中，硬件自动清零。

IE0：外部中断0中断请求标志。中断服务程序中，硬件自动清零。

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
AUXINTIF	EFH	-	INT4IF	INT3IF	INT2IF	-	T4IF	T3IF	T2IF

INT4IF：外部中断4中断请求标志。中断服务程序中，硬件自动清零。

INT3IF：外部中断3中断请求标志。中断服务程序中，硬件自动清零。

INT2IF：外部中断2中断请求标志。中断服务程序中，硬件自动清零。

15.6.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IP	B8H	PPCA	PLVD	PADC	PS	PT1	**PX1**	PT0	**PX0**
IPH	B7H	PPCAH	PLVDH	PADCH	PSH	PT1H	**PX1H**	PT0H	**PX0H**
IP2	B5H	PUSB	PI2C	PCMP	**PX4**	PPWMB	PPWMA	PSPI	PS2
IP2H	B6H	PUSBH	PI2CH	PCMPH	**PX4H**	PPWMBH	PPWMAH	PSPIH	PS2H

PX0H,PX0：外部中断0中断优先级控制位

00：INT0 中断优先级为 0 级（最低级）

01：INT0 中断优先级为 1 级（较低级）

10：INT0 中断优先级为 2 级（较高级）

11：INT0 中断优先级为 3 级（最高级）

PX1H,PX1：外部中断1中断优先级控制位

00：INT1 中断优先级为 0 级（最低级）

01：INT1 中断优先级为 1 级（较低级）

10：INT1 中断优先级为 2 级（较高级）

11：INT1 中断优先级为 3 级（最高级）

PX4H,PX4：外部中断4中断优先级控制位

00：INT4 中断优先级为 0 级（最低级）

01：INT4 中断优先级为 1 级（较低级）

10：INT4 中断优先级为 2 级（较高级）

11：INT4 中断优先级为 3 级（最高级）

外部中断 INT2、INT3 的中断优先级固定位最低级

15.6.5 外部中断类型

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TCON	88H	TF1	TR1	TF0	TR0	IE1	**IT1**	IE0	**IT0**

IT1：INT1中断模式控制

0：上升沿中断+下降沿中断

1：下降沿中断

IT0：INT0中断模式控制

0：上升沿中断+下降沿中断

1：下降沿中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTXIT	7EFEA9H	-	-	**INT4IT[1:0]**		**INT3IT[1:0]**		**INT2IT[1:0]**	

INT4IT[1:0]：INT4中断模式控制

00：下降沿中断

01：上升沿**唤醒（不支持中断）**

10：上升沿+下降沿**唤醒（不支持中断）**

11：保留

INT3IT[1:0]：INT3中断模式控制

00：下降沿中断

01：上升沿**唤醒（不支持中断）**

10：上升沿+下降沿**唤醒（不支持中断）**

11：保留

INT2IT[1:0]：INT2中断模式控制

00：下降沿中断

01：上升沿**唤醒（不支持中断）**

10：上升沿+下降沿**唤醒（不支持中断）**

11：保留

15.7 普通 I/O 口中断

15.7.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
P0 中断	FF:012BH	37
P1 中断	FF:0133H	38
P2 中断	FF:013BH	39
P3 中断	FF:0143H	40
P4 中断	FF:014BH	41
P5 中断	FF:0153H	42
P6 中断	FF:015BH	43
P7 中断	FF:0163H	44
P8 中断	FF:016BH	45
P9 中断	FF:0173H	46
PA 中断	FF:02CBH	90
PB 中断	FF:02D3H	91

15.7.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0INTE	7EFD00H	P07INTE	P06INTE	P05INTE	P04INTE	P03INTE	P02INTE	P01INTE	P00INTE
P1INTE	7EFD01H	P17INTE	P16INTE	P15INTE	P14INTE	P13INTE	P12INTE	P11INTE	P10INTE
P2INTE	7EFD02H	P27INTE	P26INTE	P25INTE	P24INTE	P23INTE	P22INTE	P21INTE	P20INTE
P3INTE	7EFD03H	P37INTE	P36INTE	P35INTE	P34INTE	P33INTE	P32INTE	P31INTE	P30INTE
P4INTE	7EFD04H	P47INTE	P46INTE	P45INTE	P44INTE	P43INTE	P42INTE	P41INTE	P40INTE
P5INTE	7EFD05H	P57INTE	P56INTE	P55INTE	P54INTE	P53INTE	P52INTE	P51INTE	P50INTE
P6INTE	7EFD06H	P67INTE	P66INTE	P65INTE	P64INTE	P63INTE	P62INTE	P61INTE	P60INTE
P7INTE	7EFD07H	P77INTE	P76INTE	P75INTE	P74INTE	P73INTE	P72INTE	P71INTE	P70INTE
P8INTE	7EFD08H	P87INTE	P86INTE	P85INTE	P84INTE	P83INTE	P82INTE	P81INTE	P80INTE
P9INTE	7EFD09H	P97INTE	P96INTE	P95INTE	P94INTE	P93INTE	P92INTE	P91INTE	P90INTE
PAINTE	7EFD0AH	PA7INTE	PA6INTE	PA5INTE	PA4INTE	PA3INTE	PA2INTE	PA1INTE	PA0INTE
PBINTE	7EFD0BH	PB7INTE	PB6INTE	PB5INTE	PB4INTE	PB3INTE	PB2INTE	PB1INTE	PB0INTE

PnINTE.x：端口中断使能控制位（n=0,1,2,3,4,5,6,7,8,9,A,B；x=0~7）

0：关闭 Pn.x 口中断功能

1：使能 Pn.x 口中断功能

15.7.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0INTF	7EFD10H	P07INTF	P06INTF	P05INTF	P04INTF	P03INTF	P02INTF	P01INTF	P00INTF
P1INTF	7EFD11H	P17INTF	P16INTF	P15INTF	P14INTF	P13INTF	P12INTF	P11INTF	P10INTF
P2INTF	7EFD12H	P27INTF	P26INTF	P25INTF	P24INTF	P23INTF	P22INTF	P21INTF	P20INTF
P3INTF	7EFD13H	P37INTF	P36INTF	P35INTF	P34INTF	P33INTF	P32INTF	P31INTF	P30INTF
P4INTF	7EFD14H	P47INTF	P46INTF	P45INTF	P44INTF	P43INTF	P42INTF	P41INTF	P40INTF
P5INTF	7EFD15H	P57INTF	P56INTF	P55INTF	P54INTF	P53INTF	P52INTF	P51INTF	P50INTF
P6INTF	7EFD16H	P67INTF	P66INTF	P65INTF	P64INTF	P63INTF	P62INTF	P61INTF	P60INTF
P7INTF	7EFD17H	P77INTF	P76INTF	P75INTF	P74INTF	P73INTF	P72INTF	P71INTF	P70INTF
P8INTF	7EFD18H	P87INTF	P86INTF	P85INTF	P84INTF	P83INTF	P82INTF	P81INTF	P80INTF
P9INTF	7EFD19H	P97INTF	P96INTF	P95INTF	P94INTF	P93INTF	P92INTF	P91INTF	P90INTF
PAINTF	7EFD1AH	PA7INTF	PA6INTF	PA5INTF	PA4INTF	PA3INTF	PA2INTF	PA1INTF	PA0INTF
PBINTF	7EFD1BH	PB7INTF	PB6INTF	PB5INTF	PB4INTF	PB3INTF	PB2INTF	PB1INTF	PB0INTF

PnINTF.x：端口中断请求标志位

0：Pn.x 口没有中断请求

1：Pn.x 口有中断请求，若使能中断，则会进入中断服务程序。标志位需软件清 0。

15.7.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PINIPL	7EFD60H	P7IP	P6IP	P5IP	P4IP	P3IP	P2IP	P1IP	P0IP
PINIPH	7EFD61H	P7IPH	P6IPH	P5IPH	P4IPH	P3IPH	P2IPH	P1IPH	P0IPH
PIN2IPL	7EFD62H	-	-	-	-	PBIP	PAIP	P9IP	P8IP
PIN2IPH	7EFD63H	-	-	-	-	PBIPH	PAIPH	P9IPH	P8IPH

PxIPH,PxIP：Px口中断优先级控制位

00：Px 口中断优先级为 0 级（最低级）

01：Px 口中断优先级为 1 级（较低级）

10：Px 口中断优先级为 2 级（较高级）

11：Px 口中断优先级为 3 级（最高级）

15.7.5 中断类型

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0IM0	7EFD20H	P07IM0	P06IM0	P05IM0	P04IM0	P03IM0	P02IM0	P01IM0	P00IM0
P1IM0	7EFD21H	P17IM0	P16IM0	P15IM0	P14IM0	P13IM0	P12IM0	P11IM0	P10IM0
P2IM0	7EFD22H	P27IM0	P26IM0	P25IM0	P24IM0	P23IM0	P22IM0	P21IM0	P20IM0
P3IM0	7EFD23H	P37IM0	P36IM0	P35IM0	P34IM0	P33IM0	P32IM0	P31IM0	P30IM0
P4IM0	7EFD24H	P47IM0	P46IM0	P45IM0	P44IM0	P43IM0	P42IM0	P41IM0	P40IM0
P5IM0	7EFD25H	P57IM0	P56IM0	P55IM0	P54IM0	P53IM0	P52IM0	P51IM0	P50IM0
P6IM0	7EFD26H	P67IM0	P66IM0	P65IM0	P64IM0	P63IM0	P62IM0	P61IM0	P60IM0
P7IM0	7EFD27H	P77IM0	P76IM0	P75IM0	P74IM0	P73IM0	P72IM0	P71IM0	P70IM0

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P8IM0	7EFD28H	P87IM0	P86IM0	P85IM0	P84IM0	P83IM0	P82IM0	P81IM0	P80IM0
P9IM0	7EFD29H	P97IM0	P96IM0	P95IM0	P94IM0	P93IM0	P92IM0	P91IM0	P90IM0
PAIM0	7EFD2AH	PA7IM0	PA6IM0	PA5IM0	PA4IM0	PA3IM0	PA2IM0	PA1IM0	PA0IM0
PBIM0	7EFD2BH	PB7IM0	PB6IM0	PB5IM0	PB4IM0	PB3IM0	PB2IM0	PB1IM0	PB0IM0
P0IM1	7EFD30H	P07IM1	P06IM1	P05IM1	P04IM1	P03IM1	P02IM1	P01IM1	P00IM1
P1IM1	7EFD31H	P17IM1	P16IM1	P15IM1	P14IM1	P13IM1	P12IM1	P11IM1	P10IM1
P2IM1	7EFD32H	P27IM1	P26IM1	P25IM1	P24IM1	P23IM1	P22IM1	P21IM1	P20IM1
P3IM1	7EFD33H	P37IM1	P36IM1	P35IM1	P34IM1	P33IM1	P32IM1	P31IM1	P30IM1
P4IM1	7EFD34H	P47IM1	P46IM1	P45IM1	P44IM1	P43IM1	P42IM1	P41IM1	P40IM1
P5IM1	7EFD35H	P57IM1	P56IM1	P55IM1	P54IM1	P53IM1	P52IM1	P51IM1	P50IM1
P6IM1	7EFD36H	P67IM1	P66IM1	P65IM1	P64IM1	P63IM1	P62IM1	P61IM1	P60IM1
P7IM1	7EFD37H	P77IM1	P76IM1	P75IM1	P74IM1	P73IM1	P72IM1	P71IM1	P70IM1
P8IM1	7EFD38H	P87IM1	P86IM1	P85IM1	P84IM1	P83IM1	P82IM1	P81IM1	P80IM1
P9IM1	7EFD39H	P97IM1	P96IM1	P95IM1	P94IM1	P93IM1	P92IM1	P91IM1	P90IM1
PAIM1	7EFD3AH	PA7IM1	PA6IM1	PA5IM1	PA4IM1	PA3IM1	PA2IM1	PA1IM1	PA0IM1
PBIM1	7EFD3BH	PB7IM1	PB6IM1	PB5IM1	PB4IM1	PB3IM1	PB2IM1	PB1IM1	PB0IM1
P0IM2	7EF4E0H	P07IM2	P06IM2	P05IM2	P04IM2	P03IM2	P02IM2	P01IM2	P00IM2
P1IM2	7EF4E1H	P17IM2	P16IM2	P15IM2	P14IM2	P13IM2	P12IM2	P11IM2	P10IM2
P2IM2	7EF4E2H	P27IM2	P26IM2	P25IM2	P24IM2	P23IM2	P22IM2	P21IM2	P20IM2
P3IM2	7EF4E3H	P37IM2	P36IM2	P35IM2	P34IM2	P33IM2	P32IM2	P31IM2	P30IM2
P4IM2	7EF4E4H	P47IM2	P46IM2	P45IM2	P44IM2	P43IM2	P42IM2	P41IM2	P40IM2
P5IM2	7EF4E5H	P57IM2	P56IM2	P55IM2	P54IM2	P53IM2	P52IM2	P51IM2	P50IM2
P6IM2	7EF4E6H	P67IM2	P66IM2	P65IM2	P64IM2	P63IM2	P62IM2	P61IM2	P60IM2
P7IM2	7EF4E7H	P77IM2	P76IM2	P75IM2	P74IM2	P73IM2	P72IM2	P71IM2	P70IM2
P8IM2	7EF4E8H	P87IM2	P86IM2	P85IM2	P84IM2	P83IM2	P82IM2	P81IM2	P80IM2
P9IM2	7EF4E9H	P97IM2	P96IM2	P95IM2	P94IM2	P93IM2	P92IM2	P91IM2	P90IM2
PAIM2	7EF4EAH	PA7IM2	PA6IM2	PA5IM2	PA4IM2	PA3IM2	PA2IM2	PA1IM2	PA0IM2
PBIM2	7EF4EBH	PB7IM2	PB6IM2	PB5IM2	PB4IM2	PB3IM2	PB2IM2	PB1IM2	PB0IM2

配置端口的模式

PnIM2.x	PnIM1.x	PnIM0.x	Pn.x 口中断模式	掉电唤醒支持
0	0	0	下降沿中断	支持
0	0	1	上升沿中断	支持
0	1	0	低电平中断	不支持
0	1	1	高电平中断	不支持
1	x	x	下降沿+上升沿中断	支持

15.8 定时器中断

15.8.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
Timer0 中断	FF:000BH	1
Timer1 中断	FF:001BH	3
Timer2 中断	FF:0063H	12
Timer3 中断	FF:009BH	19
Timer4 中断	FF:00A3H	20
Timer11 中断	FF:021BH	67
Timer5 中断	FF:0303H	96
Timer6 中断		
Timer7 中断	FF:030BH	97
Timer8 中断		
Timer9 中断	FF:0313H	98
Timer10 中断		
Timer17 中断	FF:031BH	99
Timer18 中断		

15.8.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IE	A8H	EA	ELVD	EADC	ES	**ET1**	EX1	**ET0**	EX0
IE2	AFH	EUSB	**ET4**	**ET3**	ES4	ES3	**ET2**	ESPI	ES2

ET1：定时/计数器 T1 的溢出中断允许位。

0：禁止 T1 中断

1：允许 T1 中断

ET0：定时/计数器 T0 的溢出中断允许位。

0：禁止 T0 中断

1：允许 T0 中断

ET4：定时/计数器 T4 的溢出中断允许位。

0：禁止 T4 中断

1：允许 T4 中断

ET3：定时/计数器 T3 的溢出中断允许位。

0：禁止 T3 中断

1：允许 T3 中断

ET2：定时/计数器 T2 的溢出中断允许位。

0：禁止 T2 中断

1：允许 T2 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T6T5CFG	7EFBB2H	T6IF	PT6[1:0]		**ET6**	T5IF	PT5[1:0]		**ET5**
T8T7CFG	7EFBBAH	T8IF	PT8[1:0]		**ET8**	T7IF	PT7[1:0]		**ET7**
T10T9CFG	7EFBC2H	T10IF	PT10[1:0]		**ET10**	T9IF	PT9[1:0]		**ET9**
T18T17CFG	7EFBCAH	T18IF	PT18[1:0]		**ET18**	T17IF	PT17[1:0]		**ET17**

ET5：定时/计数器 T5 的溢出中断允许位。

0：禁止 T5 中断

1：允许 T5 中断

ET6：定时/计数器 T6 的溢出中断允许位。

0：禁止 T6 中断

1：允许 T6 中断

ET7：定时/计数器 T7 的溢出中断允许位。

0：禁止 T7 中断

1：允许 T7 中断

ET8：定时/计数器 T8 的溢出中断允许位。

0：禁止 T8 中断

1：允许 T8 中断

ET9：定时/计数器 T9 的溢出中断允许位。

0：禁止 T9 中断

1：允许 T9 中断

ET10：定时/计数器 T10 的溢出中断允许位。

0：禁止 T10 中断

1：允许 T10 中断

ET17：定时/计数器 T17 的溢出中断允许位。

0：禁止 T17 中断

1：允许 T17 中断

ET18：定时/计数器 T18 的溢出中断允许位。

0：禁止 T18 中断

1：允许 T18 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T11CR	7EFE78H	T11R	T11_C/T	T11CLKO	T11x12	T11CS[1:0]		**ET11I**	T11IF

ET11I：定时/计数器 T11 的溢出中断允许位。

0：禁止 T11 中断

1：允许 T11 中断

15.8.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TCON	88H	**TF1**	TR1	**TF0**	TR0	IE1	IT1	IE0	IT0

TF1：定时器1溢出中断标志。中断服务程序中，硬件自动清零。

TF0：定时器0溢出中断标志。中断服务程序中，硬件自动清零。

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
AUXINTIF	EFH	-	INT4IF	INT3IF	INT2IF	-	**T4IF**	**T3IF**	**T2IF**

T4IF：定时器4溢出中断标志。中断服务程序中，硬件自动清零。

T3IF：定时器3溢出中断标志。中断服务程序中，硬件自动清零。

T2IF：定时器2溢出中断标志。中断服务程序中，硬件自动清零。

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T6T5CFG	7EFBB2H	**T6IF**	PT6[1:0]		ET6	**T5IF**	PT5[1:0]		ET5
T8T7CFG	7EFBBAH	**T8IF**	PT8[1:0]		ET8	**T7IF**	PT7[1:0]		ET7
T10T9CFG	7EFBC2H	**T10IF**	PT10[1:0]		ET10	**T9IF**	PT9[1:0]		ET9
T18T17CFG	7EFBCAH	**T18IF**	PT18[1:0]		ET18	**T17IF**	PT17[1:0]		ET17

T5IF：定时器5溢出中断标志。**需软件写0清零**。

T6IF：定时器6溢出中断标志。**需软件写0清零**。

T7IF：定时器7溢出中断标志。**需软件写0清零**。

T8IF：定时器8溢出中断标志。**需软件写0清零**。

T9IF：定时器9溢出中断标志。**需软件写0清零**。

T10IF：定时器10溢出中断标志。**需软件写0清零**。

T17IF：定时器17溢出中断标志。**需软件写0清零**。

T18IF：定时器18溢出中断标志。**需软件写0清零**。

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T11CR	7EFE78H	T11R	T11_C/T	T11CLKO	T11x12	T11CS[1:0]		ET11I	**T11IF**

T11IF：T11溢出中断标志。T11被允许计数以后，从初值开始加1计数，当产生溢出时，由硬件置“1”T11IF，向CPU请求中断，一直保持CPU响应该中断时，才由硬件清0（也可由查询软件清0）。

15.8.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IP	B8H	PPCA	PLVD	PADC	PS	**PT1**	PX1	**PT0**	PX0
IPH	B7H	PPCAH	PLVDH	PADCH	PSH	**PT1H**	PX1H	**PT0H**	PX0H

PT0H,PT0：定时器0中断优先级控制位

00：定时器 0 中断优先级为 0 级（最低级）
01：定时器 0 中断优先级为 1 级（较低级）
10：定时器 0 中断优先级为 2 级（较高级）
11：定时器 0 中断优先级为 3 级（最高级）

PT1H,PT1：定时器1中断优先级控制位

00：定时器 1 中断优先级为 0 级（最低级）
01：定时器 1 中断优先级为 1 级（较低级）
10：定时器 1 中断优先级为 2 级（较高级）
11：定时器 1 中断优先级为 3 级（最高级）

定时器 2、定时器 3、定时器 4、定时器 11 的中断优先级固定位最低级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T6T5CFG	7EFBB2H	T6IF	**PT6[1:0]**		ET6	T5IF	**PT5[1:0]**		ET5
T8T7CFG	7EFBBAH	T8IF	**PT8[1:0]**		ET8	T7IF	**PT7[1:0]**		ET7
T10T9CFG	7EFBC2H	T10IF	**PT10[1:0]**		ET10	T9IF	**PT9[1:0]**		ET9
T18T17CFG	7EFBCAH	T18IF	**PT18[1:0]**		ET18	T17IF	**PT17[1:0]**		ET17

PT5[1:0]：定时器5中断优先级控制位

00：定时器 5 中断优先级为 0 级（最低级）
01：定时器 5 中断优先级为 1 级（较低级）
10：定时器 5 中断优先级为 2 级（较高级）
11：定时器 5 中断优先级为 3 级（最高级）

PT6[1:0]：定时器6中断优先级控制位

00：定时器 6 中断优先级为 0 级（最低级）
01：定时器 6 中断优先级为 1 级（较低级）
10：定时器 6 中断优先级为 2 级（较高级）
11：定时器 6 中断优先级为 3 级（最高级）

PT7[1:0]：定时器7中断优先级控制位

00：定时器 7 中断优先级为 0 级（最低级）
01：定时器 7 中断优先级为 1 级（较低级）
10：定时器 7 中断优先级为 2 级（较高级）
11：定时器 7 中断优先级为 3 级（最高级）

PT8[1:0]：定时器8中断优先级控制位

00：定时器 8 中断优先级为 0 级（最低级）
01：定时器 8 中断优先级为 1 级（较低级）
10：定时器 8 中断优先级为 2 级（较高级）
11：定时器 8 中断优先级为 3 级（最高级）

PT9[1:0]：定时器 9 中断优先级控制位

00：定时器 9 中断优先级为 0 级（最低级）

01：定时器 9 中断优先级为 1 级（较低级）
10：定时器 9 中断优先级为 2 级（较高级）
11：定时器 9 中断优先级为 3 级（最高级）

PT10[1:0]：定时器10中断优先级控制位

00：定时器 10 中断优先级为 0 级（最低级）
01：定时器 10 中断优先级为 1 级（较低级）
10：定时器 10 中断优先级为 2 级（较高级）
11：定时器 10 中断优先级为 3 级（最高级）

PT17[1:0]：定时器17中断优先级控制位

00：定时器 17 中断优先级为 0 级（最低级）
01：定时器 17 中断优先级为 1 级（较低级）
10：定时器 17 中断优先级为 2 级（较高级）
11：定时器 17 中断优先级为 3 级（最高级）

PT18[1:0]：定时器18中断优先级控制位

00：定时器 18 中断优先级为 0 级（最低级）
01：定时器 18 中断优先级为 1 级（较低级）
10：定时器 18 中断优先级为 2 级（较高级）
11：定时器 18 中断优先级为 3 级（最高级）

15.9 串口中断

15.9.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
UART1 中断	FF:0023H	4
UART2 中断	FF:0043H	8
UART3 中断	FF:008BH	17
UART4 中断	FF:0093H	18
UART5 中断	FF:0333H	102
UART6 中断	FF:033BH	103
UART7 中断	FF:0343H	104
UART8 中断	FF:034BH	105

15.9.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IE	A8H	EA	ELVD	EADC	**ES**	ET1	EX1	ET0	EX0
IE2	AFH	EUSB	ET4	ET3	**ES4**	**ES3**	ET2	ESPI	**ES2**

ES：串口 1 中断允许位。

0：禁止串口 1 中断

1：允许串口 1 中断

ES4：串口 4 中断允许位。

0：禁止串口 4 中断

1：允许串口 4 中断

ES3：串口 3 中断允许位。

0：禁止串口 3 中断

1：允许串口 3 中断

ES2：串口 2 中断允许位。

0：禁止串口 2 中断

1：允许串口 2 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S5INTR	7EF742H	**ES5**	-	PS5[1:0]		-	-	-	-
S6INTR	7EF772H	**ES6**	-	PS6[1:0]		-	-	-	-
S7INTR	7EF7A2H	**ES7**	-	PS7[1:0]		-	-	-	-
S8INTR	7EF7D2H	**ES8**	-	PS8[1:0]		-	-	-	-

ES5：串口 5 中断使能控制位

0：禁止串口 5 中断

1：允许串口 5 中断

ES6：串口 6 中断使能控制位

0：禁止串口 6 中断

1：允许串口 6 中断

ES7：串口 7 中断使能控制位

0：禁止串口 7 中断

1：允许串口 7 中断

ES8：串口 8 中断使能控制位

0：禁止串口 8 中断

1：允许串口 8 中断

15.9.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SCON	98H	SM0/FE	SM1	SM2	REN	TB8	RB8	**TI**	**RI**
S2CON	9AH	S2SM0/FE	S2SM1	S2SM2	S2REN	S2TB8	S2RB8	**S2TI**	**S2RI**
S3CON	ACH	S3SM0	S3ST3	S3SM2	S3REN	S3TB8	S3RB8	**S3TI**	**S3RI**
S4CON	FDH	S4SM0	S4ST4	S4SM2	S4REN	S4TB8	S4RB8	**S4TI**	**S4RI**

TI：串口1发送完成中断请求标志。需要软件清零。

RI：串口1接收完成中断请求标志。需要软件清零。

S2TI：串口2发送完成中断请求标志。需要软件清零。

S2RI：串口2接收完成中断请求标志。需要软件清零。

S3TI：串口3发送完成中断请求标志。需要软件清零。

S3RI：串口3接收完成中断请求标志。需要软件清零。

S4TI：串口4发送完成中断请求标志。需要软件清零。

S4RI：串口 4 接收完成中断请求标志。需要软件清零。

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S5CON	7EF740H	S5SM0/FE	S5SM1	S5SM2	S5REN	S5TB8	S5RB8	**S5TI**	**S5RI**
S6CON	7EF770H	S6SM0/FE	S6SM1	S6SM2	S6REN	S6TB8	S6RB8	**S6TI**	**S6RI**
S7CON	7EF7A0H	S7SM0/FE	S7SM1	S7SM2	S7REN	S7TB8	S7RB8	**S7TI**	**S7RI**
S8CON	7EF7D0H	S8SM0/FE	S8SM1	S8SM2	S8REN	S8TB8	S8RB8	**S8TI**	**S8RI**

S5TI：串口5发送完成中断请求标志。需要软件清零。

S5RI：串口5接收完成中断请求标志。需要软件清零。

S6TI：串口6发送完成中断请求标志。需要软件清零。

S6RI：串口6接收完成中断请求标志。需要软件清零。

S7TI：串口7发送完成中断请求标志。需要软件清零。

S7RI：串口7接收完成中断请求标志。需要软件清零。

S8TI：串口8发送完成中断请求标志。需要软件清零。

S8RI：串口8接收完成中断请求标志。需要软件清零。

15.9.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IP	B8H	PPCA	PLVD	PADC	**PS**	PT1	PX1	PT0	PX0
IPH	B7H	PPCAH	PLVDH	PADCH	**PSH**	PT1H	PX1H	PT0H	PX0H
IP2	B5H	PUSB	PI2C	PCMP	PX4	PPWMB	PPWMA	PSPI	**PS2**
IP2H	B6H	PUSBH	PI2CH	PCMPH	PX4H	PPWMBH	PPWMAH	PSPIH	**PS2H**
IP3	DFH	PPWMD	PPWMC	PI2C2	PI2S2	PI2S	PRTC	**PS4**	**PS3**
IP3H	EEH	PPWMDH	PPWMCH	PI2C2H	PI2S2H	PI2SH	PRTCH	**PS4H**	**PS3H**

PSH,PS：串口1中断优先级控制位

00：串口 1 中断优先级为 0 级（最低级）
01：串口 1 中断优先级为 1 级（较低级）
10：串口 1 中断优先级为 2 级（较高级）
11：串口 1 中断优先级为 3 级（最高级）

PS2H,PS2：串口2中断优先级控制位

00：串口 2 中断优先级为 0 级（最低级）
01：串口 2 中断优先级为 1 级（较低级）
10：串口 2 中断优先级为 2 级（较高级）
11：串口 2 中断优先级为 3 级（最高级）

PS3H,PS3：串口3中断优先级控制位

00：串口 3 中断优先级为 0 级（最低级）
01：串口 3 中断优先级为 1 级（较低级）
10：串口 3 中断优先级为 2 级（较高级）
11：串口 3 中断优先级为 3 级（最高级）

PS4H,PS4：串口4中断优先级控制位

00：串口 4 中断优先级为 0 级（最低级）
01：串口 4 中断优先级为 1 级（较低级）
10：串口 4 中断优先级为 2 级（较高级）
11：串口 4 中断优先级为 3 级（最高级）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S5INTR	7EF742H	ES5	-	**PS5[1:0]**		-	-	-	-
S6INTR	7EF772H	ES6	-	**PS6[1:0]**		-	-	-	-
S7INTR	7EF7A2H	ES7	-	**PS7[1:0]**		-	-	-	-
S8INTR	7EF7D2H	ES8	-	**PS8[1:0]**		-	-	-	-

PS5[1:0]：串口 5 中断优先级控制位

00：串口 5 中断优先级为 0 级（最低级）
01：串口 5 中断优先级为 1 级（较低级）
10：串口 5 中断优先级为 2 级（较高级）
11：串口 5 中断优先级为 3 级（最高级）

PS6[1:0]：串口 6 中断优先级控制位

00：串口 6 中断优先级为 0 级（最低级）
01：串口 6 中断优先级为 1 级（较低级）
10：串口 6 中断优先级为 2 级（较高级）

11：串口 6 中断优先级为 3 级（最高级）

PS7[1:0]：串口 7 中断优先级控制位

00：串口 7 中断优先级为 0 级（最低级）

01：串口 7 中断优先级为 1 级（较低级）

10：串口 7 中断优先级为 2 级（较高级）

11：串口 7 中断优先级为 3 级（最高级）

PS8[1:0]：串口 8 中断优先级控制位

00：串口 8 中断优先级为 0 级（最低级）

01：串口 8 中断优先级为 1 级（较低级）

10：串口 8 中断优先级为 2 级（较高级）

11：串口 8 中断优先级为 3 级（最高级）

15.9.5 超时中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR1TOCR	7EFD70H	UR1ENTO	UR1ENTOI	UR1SCALE	-	-	-	-	-
UR2TOCR	7EFD74H	UR2ENTO	UR2ENTOI	UR2SCALE	-	-	-	-	-
UR3TOCR	7EFD78H	UR3ENTO	UR3ENTOI	UR3SCALE	-	-	-	-	-
UR4TOCR	7EFD7CH	UR4ENTO	UR4ENTOI	UR4SCALE	-	-	-	-	-
UR5TOCR	7EF744H	UR5ENTO	UR5ENTOI	UR5SCALE	-	-	-	-	-
UR6TOCR	7EF774H	UR6ENTO	UR6ENTOI	UR6SCALE	-	-	-	-	-
UR7TOCR	7EF7A4H	UR7ENTO	UR7ENTOI	UR7SCALE	-	-	-	-	-
UR8TOCR	7EF7D4H	UR8ENTO	UR8ENTOI	UR8SCALE	-	-	-	-	-

UR1ENTOI：串口 1 接收超时中断控制位

0：禁止串口 1 接收超时中断

1：使能串口 1 接收超时中断

UR2ENTOI：串口 2 接收超时中断控制位

0：禁止串口 2 接收超时中断

1：使能串口 2 接收超时中断

UR3ENTOI：串口 3 接收超时中断控制位

0：禁止串口 3 接收超时中断

1：使能串口 3 接收超时中断

UR4ENTOI：串口 4 接收超时中断控制位

0：禁止串口 4 接收超时中断

1：使能串口 4 接收超时中断

UR5ENTOI：串口 5 接收超时中断控制位

0：禁止串口 5 接收超时中断

1：使能串口 5 接收超时中断

UR6ENTOI：串口 6 接收超时中断控制位

0：禁止串口 6 接收超时中断

1：使能串口 6 接收超时中断

UR7ENTOI：串口 7 接收超时中断控制位

0：禁止串口 7 接收超时中断

1：使能串口 7 接收超时中断

UR8ENTOI：串口 8 接收超时中断控制位

0：禁止串口 8 接收超时中断

1：使能串口 8 接收超时中断

15.9.6 超时中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR1TOSR	7EFD71H	UR1CTOIF	-	-	-	-	-	-	UR1TOIF
UR2TOSR	7EFD75H	UR2CTOIF	-	-	-	-	-	-	UR2TOIF
UR3TOSR	7EFD79H	UR3CTOIF	-	-	-	-	-	-	UR3TOIF
UR4TOSR	7EFD7DH	UR4CTOIF	-	-	-	-	-	-	UR4TOIF
UR5TOSR	7EF745H	UR5CTOIF	-	-	-	-	-	-	UR5TOIF
UR6TOSR	7EF775H	UR6CTOIF	-	-	-	-	-	-	UR6TOIF
UR7TOSR	7EF7A5H	UR7CTOIF	-	-	-	-	-	-	UR7TOIF
UR8TOSR	7EF7D5H	UR8CTOIF	-	-	-	-	-	-	UR8TOIF

UR1TOIF：串口 1 超时中断请求标志位。标志位需要软件向 UR1CTOIF 位写“1”清零

UR2TOIF：串口 2 超时中断请求标志位。标志位需要软件向 UR2CTOIF 位写“1”清零

UR3TOIF：串口 3 超时中断请求标志位。标志位需要软件向 UR3CTOIF 位写“1”清零

UR4TOIF：串口 4 超时中断请求标志位。标志位需要软件向 UR4CTOIF 位写“1”清零

UR5TOIF：串口 5 超时中断请求标志位。标志位需要软件向 UR5CTOIF 位写“1”清零

UR6TOIF：串口 6 超时中断请求标志位。标志位需要软件向 UR6CTOIF 位写“1”清零

UR7TOIF：串口 7 超时中断请求标志位。标志位需要软件向 UR7CTOIF 位写“1”清零

UR8TOIF：串口 8 超时中断请求标志位。标志位需要软件向 UR8CTOIF 位写“1”清零

15.10 SPI 中断

15.10.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
SPI 中断	FF:004BH	9
SPI2 中断	FF:037BH	111
SPI3 中断	FF:0383H	112

15.10.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IE2	AFH	EUSB	ET4	ET3	ES4	ES3	ET2	**ESPI**	ES2

ESPI：SPI 中断允许位。

0：禁止 SPI 中断

1：允许 SPI 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI2_INTR	7EF80FH	-	-	PSPI2[1:0]		-	-	-	**ESPI2**
SPI3_INTR	7EF82FH	-	-	PSPI3[1:0]		-	-	-	**ESPI3**

ESPI2：SPI2 中断使能控制位

0：禁止 SPI2 中断

1：允许 SPI2 中断

ESPI3：SPI3 中断使能控制位

0：禁止 SPI3 中断

1：允许 SPI3 中断

15.10.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPSTAT	CDH	**SPIF**	WCOL	-	-	-	-	-	-
SPI2STAT	7EF801H	**SP2IF**	SP2WCOL	-	-	-	-	-	-
SPI3STAT	7EF821H	**SP3IF**	SP3WCOL	-	-	-	-	-	-

SPIF：SPI 数据传输完成中断请求标志。**需要软件写“1”清零**。

SP2IF：SPI2 数据传输完成中断请求标志。**需要软件写“1”清零**。

SP3IF：SPI3 数据传输完成中断请求标志。**需要软件写“1”清零**。

15.10.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IP2	B5H	PUSB	PI2C	PCMP	PX4	PPWMB	PPWMA	**PSPI**	PS2
IP2H	B6H	PUSBH	PI2CH	PCMPH	PX4H	PPWMBH	PPWMAH	**PSPIH**	PS2H

PSPIH,PSPI：SPI中断优先级控制位

00：SPI 中断优先级为 0 级（最低级）

01：SPI 中断优先级为 1 级（较低级）

10：SPI 中断优先级为 2 级（较高级）

11：SPI 中断优先级为 3 级（最高级）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI2_INTR	7EF80FH	-	-	**PSPI2[1:0]**		-	-	-	ESPI2
SPI3_INTR	7EF82FH	-	-	**PSPI3[1:0]**		-	-	-	ESPI3

PSPI2[1:0]：SPI2 中断优先级控制位

00：SPI2 中断优先级为 0 级（最低级）

01：SPI2 中断优先级为 1 级（较低级）

10：SPI2 中断优先级为 2 级（较高级）

11：SPI2 中断优先级为 3 级（最高级）

PSPI3[1:0]：SPI3 中断优先级控制位

00：SPI3 中断优先级为 0 级（最低级）

01：SPI3 中断优先级为 1 级（较低级）

10：SPI3 中断优先级为 2 级（较高级）

11：SPI3 中断优先级为 3 级（最高级）

15.10.5 超时中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPITOCR	7EFD80H	SPIENTO	SPIENTOI	SPISCALE	-	-	-	-	-
SPI2TOCR	7EF804H	SPI2ENTO	SPI2ENTOI	SPI2SCALE	-	-	-	-	-
SPI3TOCR	7EF824H	SPI3ENTO	SPI3ENTOI	SPI3SCALE	-	-	-	-	-

SPIENTOI：SPI 接收超时中断控制位

0：禁止 SPI 接收超时中断

1：使能 SPI 接收超时中断

SPI2ENTOI：SPI2 接收超时中断控制位

0：禁止 SPI2 接收超时中断

1：使能 SPI2 接收超时中断

SPI3ENTOI：SPI3 接收超时中断控制位

0：禁止 SPI3 接收超时中断

1：使能 SPI3 接收超时中断

15.10.6 超时中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPITOSR	7EFD81H	SPICTOIF	-	-	-	-	-	-	SPITOIF
SPI2TOSR	7EF805H	SPI2CTOIF	-	-	-	-	-	-	SPI2TOIF
SPI3TOSR	7EF825H	SPI3CTOIF	-	-	-	-	-	-	SPI3TOIF

SPITOIF：SPI 超时中断请求标志位。标志位需要软件向 SPICTOIF 位写“1”清零

SPI2TOIF：SPI2 超时中断请求标志位。标志位需要软件向 SPI2CTOIF 位写“1”清零

SPI3TOIF：SPI3 超时中断请求标志位。标志位需要软件向 SPI3CTOIF 位写“1”清零

15.11 QSPI 中断

15.11.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
QSPI 中断	FF:0213H	66

15.11.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPICR3	7EF902H	PMM	APMS	-	-	SMIE	FTIE	TCIE	TEIE

SMIE：QSPI状态匹配中断允许位。

0：禁止 QSPI 状态匹配中断

1：允许 QSPI 状态匹配中断

FTIE：QSPI FIFO阈值中断允许位。

0：禁止 QSPI FIFO 阈值中断

1：允许 QSPI FIFO 阈值中断

TCIE：QSPI传输完成中断允许位。

0：禁止 QSPI 传输完成中断

1：允许 QSPI 传输完成中断

TEIE：QSPI传输错误中断允许位。

0：禁止 QSPI 传输错误中断

1：允许 QSPI 传输错误中断

15.11.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_SR1	7EF906H	-	-	BUSY	-	SMF	FTF	TCF	TEF

SMF：QSPI状态匹配中断请求标志。需要软件清零。

FTF：QSPI FIFO阈值中断请求标志。需要软件清零。

TCF：QSPI传输完成中断请求标志。需要软件清零。

TEF：QSPI传输错误中断请求标志。需要软件清零。

15.11.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_DCR1	7EF904H	-	CSHT[2:0]			QSPI_IP[1:0]		-	CKMODE

QSPI_IP[1:0]：QSPI中断优先级控制位

00：QSPI 中断优先级为 0 级（最低级）

01：QSPI 中断优先级为 1 级（较低级）

10：QSPI 中断优先级为 2 级（较高级）

11：QSPI 中断优先级为 3 级（最高级）

15.12 I2C 中断

15.12.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
I2C 中断	FF:00C3H	24
I2C2 中断	FF:036BH	109

15.12.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CMSCR	7EFE81H	**EMSI**	-	-	-	MSCMD[3:0]			
I2CSLCR	7EFE83H	-	**ESTAI**	**ERXI**	**ETXI**	**ESTOI**	-	-	SLRST

EMSI：I²C主机模式中断允许位。

0：禁止 I²C 主机模式中断

1：允许 I²C 主机模式中断

ESTAI：I²C从机接收START事件中断允许位。

0：禁止 I²C 从机接收 START 事件中断

1：允许 I²C 从机接收 START 事件中断

ERXI：I²C从机接收数据完成事件中断允许位。

0：禁止 I²C 从机接收数据完成事件中断

1：允许 I²C 从机接收数据完成事件中断

ETXI：I²C从机发送数据完成事件中断允许位。

0：禁止 I²C 从机发送数据完成事件中断

1：允许 I²C 从机发送数据完成事件中断

ESTOI：I²C从机接收STOP事件中断允许位。

0：禁止 I²C 从机接收 STOP 事件中断

1：允许 I²C 从机接收 STOP 事件中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2MSCR	7EF861H	**EI2C2MSI**	-	-	-	I2C2MSCMD[3:0]			

EI2C2MSI：主机模式中断使能控制位

0：关闭主机模式的中断

1：允许主机模式的中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2SLCR	7EF863H	-	**EI2C2STAI**	**EI2C2RXI**	**EI2C2TXI**	**EI2C2STOI**	-	-	I2C2SLRST

EI2C2STAI：从机模式时接收到 START 信号中断允许位

0：禁止从机模式时接收到 START 信号时发生中断

1：使能从机模式时接收到 START 信号时发生中断

EI2C2RXI：从机模式时接收到 1 字节数据后中断允许位

0：禁止从机模式时接收到数据后发生中断

1：使能从机模式时接收到 1 字节数据后发生中断

EI2C2TXI：从机模式时发送完成 1 字节数据后中断允许位

0：禁止从机模式时发送完成数据后发生中断

1：使能从机模式时发送完成 1 字节数据后发生中断

EI2C2STOI：从机模式时接收到 STOP 信号中断允许位

0：禁止从机模式时接收到 STOP 信号时发生中断

1：使能从机模式时接收到 STOP 信号时发生中断

15.12.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CMSST	7EFE82H	MSBUSY	**MSIF**	-	-	-	-	MSACKI	MSACKO
I2CSLST	7EFE84H	SLBUSY	**STAIF**	**RXIF**	**TXIF**	**STOIF**	TXING	SLACKI	SLACKO

MSIF：I²C主机模式中断请求标志。需要软件清零。

STAIF：I²C从机接收START事件中断请求标志。需要软件清零。

RXIF：I²C从机接收数据完成事件中断请求标志。需要软件清零。

TXIF：I²C从机发送数据完成事件中断请求标志。需要软件清零。

STOIF：I²C从机接收STOP事件中断请求标志。需要软件清零。

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2MSST	7EF862H	I2C2MSBUSY	**I2C2MSIF**	-	-	-	-	I2C2MSACKI	I2C2MSACKO

I2C2MSIF：主机模式的中断请求位（中断标志位）。需要软件清零。

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2SLST	7EF864H	I2C2SLBUSY	**I2C2STAIF**	**I2C2RXIF**	**I2C2TXIF**	**I2C2STOIF**	-	I2C2SLACKI	I2C2SLACKO

I2C2STAIF：I²C从机接收START事件中断请求标志。需要软件清零。

I2C2RXIF：I²C从机接收数据完成事件中断请求标志。需要软件清零。

I2C2TXIF：I²C从机发送数据完成事件中断请求标志。需要软件清零。

I2C2STOIF：I²C从机接收STOP事件中断请求标志。需要软件清零。

15.12.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IP2	B5H	PUSB	**PI2C**	PCMP	PX4	PPWMB	PPWMA	PSPI	PS2
IP2H	B6H	PUSBH	**PI2CH**	PCMPH	PX4H	PPWMBH	PPWMAH	PSPIH	PS2H
IP3	DFH	PPWMD	PPWMC	**PI2C2**	PI2S2	PI2S	PRTC	PS4	PS3
IP3H	EEH	PPWMDH	PPWMCH	**PI2C2H**	PI2S2H	PI2SH	PRTCH	PS4H	PS3H

PI2CH,PI2C：I2C中断优先级控制位

00：I2C 中断优先级为 0 级（最低级）

01：I2C 中断优先级为 1 级（较低级）

10：I2C 中断优先级为 2 级（较高级）

11：I2C 中断优先级为 3 级（最高级）

PI2C2H,PI2C2：I2C2中断优先级控制位

00：I2C2 中断优先级为 0 级（最低级）

01：I2C2 中断优先级为 1 级（较低级）

10：I2C2 中断优先级为 2 级（较高级）

11：I2C2 中断优先级为 3 级（最高级）

15.12.5 超时中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CTOCR	7EFD84H	I2CENTO	I2CENTOI	I2CSCALE	-	-	-	-	-
I2C2TOCR	7EF86CH	I2C2ENTO	I2C2ENTOI	I2C2SCALE	-	-	-	-	-

I2CENTOI：I2C 接收超时中断控制位

0：禁止 I2C 接收超时中断

1：使能 I2C 接收超时中断

I2C2ENTOI：I2C2 接收超时中断控制位

0：禁止 I2C2 接收超时中断

1：使能 I2C2 接收超时中断

15.12.6 超时中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CTOSR	7EFD85H	I2CCTOIF	-	-	-	-	-	-	I2CTOIF
I2C2TOSR	7EF86DH	I2C2CTOIF	-	-	-	-	-	-	I2C2TOIF

I2CTOIF：I2C1 超时中断请求标志位。标志位需要软件向 I2CCTOIF 位写“1”清零

I2C2TOIF：I2C2 超时中断请求标志位。标志位需要软件向 I2C2CTOIF 位写“1”清零

15.13 I2S 中断

15.13.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
I2S 中断	FF:01F3H	62
I2S2 中断	FF:0373H	110

15.13.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2SCR	7EFD98H	I2STXEIE	I2SRXNEIE	I2SERRIE	I2SFRF	-	-	I2STXDMAEN	I2SRXDMAEN
I2S2CR	7EFD90H	I2S2TXEIE	I2S2RXNEIE	I2S2ERRIE	I2S2FRF	-	-	I2S2TXDMAEN	I2S2RXDMAEN

I2STXEIE：I2S发送缓冲区空中断允许位。

0：禁止 I2S 发送缓冲区空中断

1：允许 I2S 发送缓冲区空中断

I2SRXNEIE：I2S接收缓冲区非空中断允许位。

0：禁止 I2S 接收缓冲区非空中断

1：允许 I2S 接收缓冲区非空中断

I2SERRIE：I2S错误中断允许位。

0：禁止 I2S 错误中断

1：允许 I2S 错误中断

I2S2TXEIE：I2S2发送缓冲区空中断允许位。

0：禁止 I2S2 发送缓冲区空中断

1：允许 I2S2 发送缓冲区空中断

I2S2RXNEIE：I2S2接收缓冲区非空中断允许位。

0：禁止 I2S2 接收缓冲区非空中断

1：允许 I2S2 接收缓冲区非空中断

I2S2ERRIE：I2S2错误中断允许位。

0：禁止 I2S2 错误中断

1：允许 I2S2 错误中断

15.13.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2SSR	7EFD99H	-	I2SFRE	I2SBUY	I2SOVR	I2SUDR	I2SCHSID	I2STXE	I2SRXNE
I2S2SR	7EFD91H	-	I2S2FRE	I2S2BUY	I2S2OVR	I2S2UDR	I2S2CHSID	I2S2TXE	I2S2RXNE

I2STXE：I2S发送缓冲区空中断中断请求标志。不需要软件清零。

I2SRXNE：I2S接收缓冲区非空中断请求标志。不需要软件清零。

I2S2TXE：I2S2发送缓冲区空中断中断请求标志。不需要软件清零。

I2S2RXNE：I2S2接收缓冲区非空中断请求标志。不需要软件清零。

15.13.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IP3	DFH	PPWMD	PPWMC	PI2C2	**PI2S2**	**PI2S**	PRTC	PS4	PS3
IP3H	EEH	PPWMDH	PPWMCH	PI2C2H	**PI2S2H**	**PI2SH**	PRTCH	PS4H	PS3H

PI2SH,PI2S：I2S中断优先级控制位

00：I2S 中断优先级为 0 级（最低级）
01：I2S 中断优先级为 1 级（较低级）
10：I2S 中断优先级为 2 级（较高级）
11：I2S 中断优先级为 3 级（最高级）

PI2S2H,PI2S2：I2S2中断优先级控制位

00：I2S2 中断优先级为 0 级（最低级）
01：I2S2 中断优先级为 1 级（较低级）
10：I2S2 中断优先级为 2 级（较高级）
11：I2S2 中断优先级为 3 级（最高级）

15.14 RTC 中断

15.14.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
RTC 中断	FF:0123H	36

15.14.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
RTCIEN	7EFE62H	EALAI	EDAYI	EHOURI	EMINI	ESECI	ESEC2I	ESEC8I	ESEC32I

EALAI：闹钟中断使能位

0：关闭闹钟中断

1：使能闹钟中断

EDAYI：一日（24 小时）中断使能位

0：关闭一日中断

1：使能一日中断

EHOURI：一小时（60 分钟）中断使能位

0：关闭小时中断

1：使能小时中断

EMINI：一分钟（60 秒）中断使能位

0：关闭分钟中断

1：使能分钟中断

ESECI：一秒中断使能位

0：关闭秒中断

1：使能秒中断

ESEC2I：1/2 秒中断使能位

0：关闭 1/2 秒中断

1：使能 1/2 秒中断

ESEC8I：1/8 秒中断使能位

0：关闭 1/8 秒中断

1：使能 1/8 秒中断

ESEC32I：1/32 秒中断使能位

0：关闭 1/32 秒中断

1：使能 1/32 秒中断

15.14.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
RTCIF	7EFE63H	ALAIF	DAYIF	HOURIF	MINIF	SECIF	SEC2IF	SEC8IF	SEC32IF

ALAIF：闹钟中断请求位。需软件清零。

DAYIF：一日（24 小时）中断请求位。需软件清零。

HOURIF：一小时（60 分钟）中断请求位。需软件清零。

MINIF：一分钟（60 秒）中断请求位。需软件清零。

SECIF：一秒中断请求位。需软件清零。

SEC2IF：1/2 秒中断请求位。需软件清零

SEC8IF：1/8 秒中断请求位。需软件清零。

SEC32IF：1/32 秒中断请求位。需软件清零。

15.14.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IP3	DFH	PPWMD	PPWMC	PI2C2	PI2S2	PI2S	**PRTC**	PS4	PS3
IP3H	EEH	PPWMDH	PPWMCH	PI2C2H	PI2S2H	PI2SH	**PRTCH**	PS4H	PS3H

PRTCH,PRTC：RTC中断优先级控制位

00：RTC 中断优先级为 0 级（最低级）

01：RTC 中断优先级为 1 级（较低级）

10：RTC 中断优先级为 2 级（较高级）

11：RTC 中断优先级为 3 级（最高级）

15.15 TFT 彩屏/LCM 中断

15.15.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
TFT 彩屏中断	FF:01DBH	59

15.15.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LCMIFCFG	7EFE50H	**LCMIFIE**	-	LCMIFIP[1:0]		LCMIFDPS[1:0]		D16_D8	M68_I80

LCMIFIE：TFT彩屏接口中断允许位。

0：禁止 TFT 彩屏接口中断

1：允许 TFT 彩屏接口中断

15.15.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LCMIFSTA	7EFE53H	-	-	-	-	-	-	-	**LCMIFIF**

LCMIFIF：TFT彩屏接口中断请求标志。需要软件清零。

15.15.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LCMIFCFG	7EFE50H	LCMIFIE	-	**LCMIFIP[1:0]**		LCMIFDPS[1:0]		D16_D8	M68_I80

LCMIFIP[1:0]：TFT彩屏接口中断优先级控制位

00：TFT 彩屏接口中断优先级为 0 级（最低级）

01：TFT 彩屏接口中断优先级为 1 级（较低级）

10：TFT 彩屏接口中断优先级为 2 级（较高级）

11：TFT 彩屏接口中断优先级为 3 级（最高级）

15.16 高级 PWM 中断

15.16.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
PWMA 中断	FF:00D3H	26
PWMB 中断	FF:00DBH	27
PWMC 中断	FF:03CBH	121
PWMD 中断	FF:03D3H	122
PWME 中断	FF:03DBH	123
PWMF 中断	FF:03E3H	124

15.16.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_IER	7EFEC4H	BIEA	TIEA	COMIEA	CCA4IE	CCA3IE	CCA2IE	CCA1IE	UIEA
PWMB_IER	7EFEE4H	BIEB	TIEB	COMIEB	CCB8IE	CCB7IE	CCB6IE	CCB5IE	UIEB
PWMC_IER	7EF8C4H	BIEC	TIEC	COMIEC	CCC4IE	CCC3IE	CCC2IE	CCC1IE	UIEC
PWMD_IER	7EF8E4H	BIED	TIED	COMIED	CCD8IE	CCD7IE	CCD6IE	CCD5IE	UIED
PWME_IER	7EF6C4H	BIEE	TIEE	COMIEE	CCE4IE	CCE3IE	CCE2IE	CCE1IE	UIEE
PWMF_IER	7EF6E4H	BIEF	TIEF	COMIEF	CCF8IE	CCF7IE	CCF6IE	CCF5IE	UIEF

BIEn：允许刹车中断（n=A/B/C/D/E/F）

0：禁止刹车中断；

1：允许刹车中断。

TIEn：触发中断使能（n=A/B/C/D/E/F）

0：禁止触发中断；

1：使能触发中断。

COMIE：允许 COM 中断（n=A/B/C/D/E/F）

0：禁止 COM 中断；

1：允许 COM 中断。

CCnxIE：允许捕获/比较 n 中断（n=A/B/C/D/E/F，x=1/2/3/4/5/6/7/8）

0：禁止捕获/比较 n 中断；

1：允许捕获/比较 n 中断。

UIEn：允许更新中断（n=A/B/C/D/E/F）

0：禁止更新中断；

1：允许更新中断。

15.16.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_SR1	7EFEC5H	BIFA	TIFA	COMIFA	CCA4IF	CCA3IF	CCA2IF	CCA1IF	UIFA
PWMB_SR1	7EFEE5H	BIFB	TIFB	COMIFB	CCB8IF	CCB7IF	CCB6IF	CCB5IF	UIFB
PWMC_SR1	7EF8C5H	BIFC	TIFC	COMIFC	CCC4IF	CCC3IF	CCC2IF	CCC1IF	UIFA
PWMD_SR1	7EF8E5H	BIFD	TIFD	COMIFD	CCD8IF	CCD7IF	CCD6IF	CCD5IF	UIFB
PWME_SR1	7EF6C5H	BIFE	TIFE	COMIFE	CCE4IF	CCE3IF	CCE2IF	CCE1IF	UIFA
PWMF_SR1	7EF6E5H	BIFF	TIFF	COMIFF	CCF8IF	CCF7IF	CCF6IF	CCF5IF	UIFB

BIFnn：PWMn刹车中断请求标志。需要软件清零。（n=A/B/C/D/E/F）

TIF：PWMn触发中断请求标志。需要软件清零。（n=A/B/C/D/E/F）

COMIFn：PWMn比较中断请求标志。需要软件清零。（n=A/B/C/D/E/F）

CCxIFn：PWMn通道4发生捕获比较中断请求标志。需要软件清零。

（n=A/B/C/D/E/F，x=1/2/3/4/5/6/7/8）

UIFn：PWMn更新中断请求标志。需要软件清零。（n=A/B/C/D/E/F）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_SR2	7EFEC6H	-	-	-	CCA4OF	CCA3OF	CCA2OF	CCA1OF	-
PWMB_SR2	7EFEE6H	-	-	-	CCB8OF	CCB7OF	CCB6OF	CCB5OF	-
PWMC_SR2	7EF8C6H	-	-	-	CCC4OF	CCC3OF	CCC2OF	CCC1OF	-
PWMD_SR2	7EF8E6H	-	-	-	CCD8OF	CCD7OF	CCD6OF	CCD5OF	-
PWME_SR2	7EF6C6H	-	-	-	CCE4OF	CCE3OF	CCE2OF	CCE1OF	-
PWMF_SR2	7EF6E6H	-	-	-	CCF8OF	CCF7OF	CCF6OF	CCF5OF	-

CCxOFn：PWMn通道4发生重复捕获中断请求标志。需要软件清零。

（n=A/B/C/D/E/F，x=1/2/3/4/5/6/7/8）

15.16.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IP2	B5H	PUSB	PI2C	PCMP	PX4	**PPWMB**	**PPWMA**	PSPI	PS2
IP2H	B6H	PUSBH	PI2CH	PCMPH	PX4H	**PPWMBH**	**PPWMAH**	PSPIH	PS2H
IP3	DFH	**PPWMD**	**PPWMC**	PI2C2	PI2S2	PI2S	PRTC	PS4	PS3
IP3H	EEH	**PPWMDH**	**PPWMCH**	PI2C2H	PI2S2H	PI2SH	PRTCH	PS4H	PS3H
IP4	DFH	-	-	-	-	-	-	**PPWMF**	**PPWME**
IP4H	EEH	-	-	-	-	-	-	**PPWMFH**	**PPWMEH**

PPWMnH,PPWMn：高级PWMn中断优先级控制位（n=A/B/C/D/E/F）

00：高级 PWMn 中断优先级为 0 级（最低级）

01：高级 PWMn 中断优先级为 1 级（较低级）

10：高级 PWMn 中断优先级为 2 级（较高级）

11：高级 PWMn 中断优先级为 3 级（最高级）

15.17 USB 中断

15.17.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
USB 中断	FF:00CBH	25

15.17.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IE2	AFH	**EUSB**	ET4	ET3	ES4	ES3	ET2	ESPI	ES2

EUSB：USB 中断允许位。

0：禁止 USB 中断

1：允许 USB 中断。打开 USB 中断总开关后，还需要使能 USB 的 IN/OUT 端点中断才能产生 USB 中断。

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTRIN1E	07H	-	-	EP5INIE	EP4INIE	EP3INIE	EP2INIE	EP1INIE	EP0IE

EP5INIE：端点 5 的 IN 中断控制位

0：禁止端点 5 的 IN 中断

1：允许端点 5 的 IN 中断

EP4INIE：端点 4 的 IN 中断控制位

0：禁止端点 4 的 IN 中断

1：允许端点 4 的 IN 中断

EP3INIE：端点 3 的 IN 中断控制位

0：禁止端点 3 的 IN 中断

1：允许端点 3 的 IN 中断

EP2INIE：端点 2 的 IN 中断控制位

0：禁止端点 2 的 IN 中断

1：允许端点 2 的 IN 中断

EP1INIE：端点 1 的 IN 中断控制位

0：禁止端点 1 的 IN 中断

1：允许端点 1 的 IN 中断

EP0IE：端点 0 的 IN/OUT 中断控制位

0：禁止端点 0 的 IN/OUT 中断

1：允许端点 0 的 IN/OUT 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTROUT1E	09H	-	-	EP5OUTIE	EP4OUTIE	EP3OUTIE	EP2OUTIE	EP1OUTIE	-

EP5OUTIE：端点 5 的 OUT 中断控制位

0：禁止端点 5 的 OUT 中断

1：允许端点 5 的 OUT 中断

EP4OUTIE：端点 4 的 OUT 中断控制位

0：禁止端点 4 的 OUT 中断

1：允许端点 4 的 OUT 中断

EP3OUTIE：端点 3 的 OUT 中断控制位

0：禁止端点 3 的 OUT 中断

1：允许端点 3 的 OUT 中断

EP2OUTIE：端点 2 的 OUT 中断控制位

0：禁止端点 2 的 OUT 中断

1：允许端点 2 的 OUT 中断

EP1OUTIE：端点 1 的 OUT 中断控制位

0：禁止端点 1 的 OUT 中断

1：允许端点 1 的 OUT 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTRUSBE	0BH	-	-	-	-	SOFIE	RSTIE	RSUIE	SUSIE

SOFIE：USB 帧起始信号中断控制位

0：禁止 USB 帧起始信号中断

1：允许 USB 帧起始信号中断

RSTIE：USB 复位信号中断控制位

0：禁止 USB 复位信号中断

1：允许 USB 复位信号中断

RSUIE：USB 恢复信号中断控制位

0：禁止 USB 恢复信号中断

1：允许 USB 恢复信号中断

SUSIE：USB 挂起信号中断控制位

0：禁止 USB 挂起信号中断

1：允许 USB 挂起信号中断

15.17.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTRIN1	02H	-	-	EP5INIF	EP4INIF	EP3INIF	EP2INIF	EP1INIF	EP0IF

EP5INIF：端点 5 的 IN 中断标志位

0：端点 5 的 IN 中断无效

1：端点 5 的 IN 中断有效

EP4INIF：端点 4 的 IN 中断标志位

0：端点 4 的 IN 中断无效

1：端点 4 的 IN 中断有效

EP3INIF：端点 3 的 IN 中断标志位

0：端点 3 的 IN 中断无效

1：端点 3 的 IN 中断有效

EP2INIF：端点 2 的 IN 中断标志位

0：端点 2 的 IN 中断无效

1：端点 2 的 IN 中断有效

EP1INIF：端点 1 的 IN 中断标志位

0：端点 1 的 IN 中断无效

1：端点 1 的 IN 中断有效

EP0IF：端点 0 的 IN/OUT 中断标志位

0：端点 0 的 IN/OUT 中断无效

1：端点 0 的 IN/OUT 中断有效

在软件读取 INTRIN1 寄存器后，硬件将自动清除 INTRIN1 中的所有的中断标志

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTROUT1	04H	-	-	EP5OUTIF	EP4OUTIF	EP3OUTIF	EP2OUTIF	EP1OUTIF	-

EP5OUTIF：端点 5 的 OUT 中断标志位

0：端点 5 的 OUT 中断无效

1：端点 5 的 OUT 中断有效

EP4OUTIF：端点 4 的 OUT 中断标志位

0：端点 4 的 OUT 中断无效

1：端点 4 的 OUT 中断有效

EP3OUTIF：端点 3 的 OUT 中断标志位

0：端点 3 的 OUT 中断无效

1：端点 3 的 OUT 中断有效

EP2OUTIF：端点 2 的 OUT 中断标志位

0：端点 2 的 OUT 中断无效

1：端点 2 的 OUT 中断有效

EP1OUTIF：端点 1 的 OUT 中断标志位

0：端点 1 的 OUT 中断无效

1：端点 1 的 OUT 中断有效

在软件读取 INTROUT1 寄存器后，硬件将自动清除 INTROUT1 中的所有的中断标志

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTRUSB	06H	-	-	-	-	SOFIF	RSTIF	RSUIF	SUSIF

SOFIF：USB 帧起始信号中断标志

0：USB 帧起始信号中断无效

1：USB 帧起始信号中断有效

RSTIF：USB 复位信号中断标志

0：USB 复位信号中断无效

1：USB 复位信号中断有效

RSUIF：USB 恢复信号中断标志

0：USB 恢复信号中断无效

1：USB 恢复信号中断有效

SUSIF：USB 挂起信号中断标志

0：USB 挂起信号中断无效

1：USB 挂起信号中断有效

在软件读取 INTRUSB 寄存器后，硬件将自动清除 INTRUSB 中的所有的中断标志

15.17.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IP2	B5H	**PUSB**	PI2C	PCMP	PX4	PPWMB	PPWMA	PSPI	PS2
IP2H	B6H	**PUSBH**	PI2CH	PCMPH	PX4H	PPWMBH	PPWMAH	PSPIH	PS2H

PUSBH,PUSB：USB中断优先级控制位

00：USB 中断优先级为 0 级（最低级）

01：USB 中断优先级为 1 级（较低级）

10：USB 中断优先级为 2 级（较高级）

11：USB 中断优先级为 3 级（最高级）

15.18 ADC 中断

15.18.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
ADC 中断	FF:002BH	5
ADC2 中断	FF:0353H	106

15.18.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IE	A8H	EA	ELVD	**EADC**	ES	ET1	EX1	ET0	EX0

EADC：A/D 转换中断允许位。

0：禁止 A/D 转换中断

1：允许 A/D 转换中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADC2INTR	7EFB3EH	-	-	-	-	-	PADC2[1:0]		**EADC2**

EADC2：ADC2 中断使能控制位

0：禁止 ADC2 中断

1：允许 ADC2 中断

15.18.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADC_CONTR	BCH	ADC_POWER	ADC_START	ADC_FLAG	ADC_EPWMT	ADC_CHS[3:0]			
ADC2_CONTR	7EFB38H	ADC2_POWER	ADC2_START	ADC2_FLAG	ADC2_EPWMT	ADC2_CHS[3:0]			

ADC_FLAG：ADC 转换完成中断请求标志。需要软件清零。

ADC2_FLAG：ADC2 转换完成中断请求标志。需要软件清零。

15.18.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IP	B8H	PPCA	PLVD	**PADC**	PS	PT1	PX1	PT0	PX0
IPH	B7H	PPCAH	PLVDH	**PADCH**	PSH	PT1H	PX1H	PT0H	PX0H

PADCH,PADC：ADC中断优先级控制位

00：ADC 中断优先级为 0 级（最低级）

01：ADC 中断优先级为 1 级（较低级）

10：ADC 中断优先级为 2 级（较高级）

11：ADC 中断优先级为 3 级（最高级）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADC2INTR	7EFB3EH	-	-	-	-	-	**PADC2[1:0]**		EADC2

PADC2[1:0]：ADC2 中断优先级控制位

00：ADC2 中断优先级为 0 级（最低级）

01：ADC2 中断优先级为 1 级（较低级）

10：ADC2 中断优先级为 2 级（较高级）

11：ADC2 中断优先级为 3 级（最高级）

15.19 DAC 中断

15.19.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
DAC1 中断	FF:035BH	107
DAC2 中断	FF:0363H	108

15.19.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DAC1_CR	7EF610H	DMAEN	DAC1TG	DAC1TS[1:0]		DAC1IP[1:0]		**EDAC1I**	DAC1EN
DAC2_CR	7EF628H	DMAEN	DAC2TG	DAC2TS[1:0]		DAC2IP[1:0]		**EDAC2I**	DAC2EN

EDAC1I：DAC1中断控制位

0：禁止 DAC1 中断

1：使能 DAC1 中断

EDAC2I：DAC2中断控制位

0：禁止 DAC2 中断

1：使能 DAC2 中断

15.19.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DAC1_STA	7EF611H	BUSY	-	-	-	-	-	-	**DAC1IF**
DAC2_STA	7EF629H	BUSY	-	-	-	-	-	-	**DAC2IF**

DAC1IF：DAC1中断请求标志位，需要软件写0清零

DAC2IF：DAC2中断请求标志位，需要软件写0清零

15.19.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DAC1_CR	7EF610H	DMAEN	DAC1TG	DAC1TS[1:0]		**DAC1IP[1:0]**		EDAC1I	DAC1EN
DAC2_CR	7EF628H	DMAEN	DAC2TG	DAC2TS[1:0]		**DAC2IP[1:0]**		EDAC2I	DAC2EN

DAC1IP[1:0]：DAC1中断优先级控制位

00：DAC1 中断优先级为 0 级（最低级）

01：DAC1 中断优先级为 1 级（较低级）

10：DAC1 中断优先级为 2 级（较高级）

11：DAC1 中断优先级为 3 级（最高级）

DAC2IP[1:0]：DAC2中断优先级控制位

00：DAC2 中断优先级为 0 级（最低级）

01：DAC2 中断优先级为 1 级（较低级）

10：DAC2 中断优先级为 2 级（较高级）

11：DAC2 中断优先级为 3 级（最高级）

15.20 CMP 比较器中断

15.20.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
CMP 中断	FF:00ABH	21
CMP2 中断	FF:0393H	114
CMP3 中断	FF:039BH	115
CMP4 中断	FF:03A3H	116

15.20.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP1_6P6N_CR1	7EFB70H	CMP1EN	CMP1IF	**CMP1PIE**	**CMP1NIE**	CMP1IP[1:0]		CMP1OE	CMP1RES
CMP2_6P6N_CR1	7EFB74H	CMP2EN	CMP2IF	CMP2PIE	CMP2NIE	CMP2IP[1:0]		CMP2OE	CMP2RES
CMP3_6P6N_CR1	7EFB78H	CMP3EN	CMP3IF	CMP3PIE	CMP3NIE	CMP3IP[1:0]		CMP3OE	CMP3RES
CMP4_6P6N_CR1	7EFB7CH	CMP4EN	CMP4IF	CMP4PIE	CMP4NIE	CMP4IP[1:0]		CMP4OE	CMP4RES

CMP1PIE：比较器上升沿中断使能位。

0：禁止比较器上升沿中断。

1：使能比较器上升沿中断。使能比较器的比较结果由 0 变成 1 时产生中断请求。

CMP1NIE：比较器下降沿中断使能位。

0：禁止比较器下降沿中断。

1：使能比较器下降沿中断。使能比较器的比较结果由 1 变成 0 时产生中断请求。

CMP2PIE：比较器 2 上升沿中断使能位。

0：禁止比较器 2 上升沿中断。

1：使能比较器 2 上升沿中断。使能比较器 2 的比较结果由 0 变成 1 时产生中断请求。

CMP2NIE：比较器 2 下降沿中断使能位。

0：禁止比较器 2 下降沿中断。

1：使能比较器 2 下降沿中断。使能比较器 2 的比较结果由 1 变成 0 时产生中断请求。

CMP3PIE：比较器 3 上升沿中断使能位。

0：禁止比较器 3 上升沿中断。

1：使能比较器 3 上升沿中断。使能比较器 3 的比较结果由 0 变成 1 时产生中断请求。

CMP3NIE：比较器 3 下降沿中断使能位。

0：禁止比较器 3 下降沿中断。

1：使能比较器 3 下降沿中断。使能比较器 3 的比较结果由 1 变成 0 时产生中断请求。

CMP4PIE：比较器 4 上升沿中断使能位。

0：禁止比较器 4 上升沿中断。

1：使能比较器 4 上升沿中断。使能比较器 4 的比较结果由 0 变成 1 时产生中断请求。

CMP4NIE：比较器 4 下降沿中断使能位。

0：禁止比较器 4 下降沿中断。

1：使能比较器 4 下降沿中断。使能比较器 4 的比较结果由 1 变成 0 时产生中断请求。

15.20.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP1_6P6N_CR1	7EFB70H	CMP1EN	**CMP1IF**	CMP1PIE	CMP1NIE	CMP1IP[1:0]		CMP1OE	CMP1RES
CMP2_6P6N_CR1	7EFB74H	CMP2EN	**CMP2IF**	CMP2PIE	CMP2NIE	CMP2IP[1:0]		CMP2OE	CMP2RES
CMP3_6P6N_CR1	7EFB78H	CMP3EN	**CMP3IF**	CMP3PIE	CMP3NIE	CMP3IP[1:0]		CMP3OE	CMP3RES
CMP4_6P6N_CR1	7EFB7CH	CMP4EN	**CMP4IF**	CMP4PIE	CMP4NIE	CMP4IP[1:0]		CMP4OE	CMP4RES

CMP1IF：比较器 1 中断标志位。当 CMP1PIE 或 CMP1NIE 被使能后，若产生相应的中断信号，硬件自动将 CMP1IF 置 1，并向 CPU 提出中断请求。此标志位必须用户软件向此位写 0 清零。

CMP2IF：比较器 2 中断标志位。当 CMP2PIE 或 CMP2NIE 被使能后，若产生相应的中断信号，硬件自动将 CMP2IF 置 1，并向 CPU 提出中断请求。此标志位必须用户软件向此位写 0 清零。

CMP3IF：比较器 3 中断标志位。当 CMP3PIE 或 CMP3NIE 被使能后，若产生相应的中断信号，硬件自动将 CMP3IF 置 1，并向 CPU 提出中断请求。此标志位必须用户软件向此位写 0 清零。

CMP4IF：比较器 4 中断标志位。当 CMP4PIE 或 CMP4NIE 被使能后，若产生相应的中断信号，硬件自动将 CMP4IF 置 1，并向 CPU 提出中断请求。此标志位必须用户软件向此位写 0 清零。

15.20.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP1_6P6N_CR1	7EFB70H	CMP1EN	CMP1IF	CMP1PIE	CMP1NIE	**CMP1IP[1:0]**		CMP1OE	CMP1RES
CMP2_6P6N_CR1	7EFB74H	CMP2EN	CMP2IF	CMP2PIE	CMP2NIE	**CMP2IP[1:0]**		CMP2OE	CMP2RES
CMP3_6P6N_CR1	7EFB78H	CMP3EN	CMP3IF	CMP3PIE	CMP3NIE	**CMP3IP[1:0]**		CMP3OE	CMP3RES
CMP4_6P6N_CR1	7EFB7CH	CMP4EN	CMP4IF	CMP4PIE	CMP4NIE	**CMP4IP[1:0]**		CMP4OE	CMP4RES

CMP1IP[1:0]：比较器 1 中断优先级设置。

00：比较器 1 中断为最低优先级。

01：比较器 1 中断为较低优先级。

10：比较器 1 中断为较高优先级。

11：比较器 1 中断为最高优先级。

CMP2IP[1:0]：比较器 2 中断优先级设置。

00：比较器 2 中断为最低优先级。

01：比较器 2 中断为较低优先级。

10：比较器 2 中断为较高优先级。

11：比较器 2 中断为最高优先级。

CMP3IP[1:0]：比较器 3 中断优先级设置。

00：比较器 3 中断为最低优先级。

01：比较器 3 中断为较低优先级。

10：比较器 3 中断为较高优先级。

11：比较器 3 中断为最高优先级。

CMP4IP[1:0]：比较器 4 中断优先级设置。

00：比较器 4 中断为最低优先级。

01：比较器 4 中断为较低优先级。

10：比较器 4 中断为较高优先级。

11：比较器 4 中断为最高优先级。

15.21 LVD 低压检测中断

15.21.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
LVD 中断	FF:0033H	6

15.21.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IE	A8H	EA	**ELVD**	EADC	ES	ET1	EX1	ET0	EX0

ELVD：低压检测中断允许位。

0：禁止低压检测中断

1：允许低压检测中断

15.21.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PCON	87H	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL

LVDF：低压检测中断请求标志。需要软件清零。

15.21.4 中断优先级

IP	B8H	PPCA	**PLVD**	PADC	PS	PT1	PX1	PT0	PX0
IPH	B7H	PPCAH	**PLVDH**	PADCH	PSH	PT1H	PX1H	PT0H	PX0H

PLVDH,PLVD：低压检测中断优先级控制位

00：LVD 中断优先级为 0 级（最低级）

01：LVD 中断优先级为 1 级（较低级）

10：LVD 中断优先级为 2 级（较高级）

11：LVD 中断优先级为 3 级（最高级）

15.22 DMA 中断

15.22.1 中断入口地址

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
DMA_M2M 中断	FF:017BH	47
DMA_ADC 中断	FF:0183H	48
DMA_ADC2 中断	FF:0263H	76
DMA_DAC1 中断	FF:026BH	77
DMA_DAC2 中断	FF:0273H	78
DMA_SPI 中断	FF:018BH	49
DMA_SPI2 中断	FF:027BH	79
DMA_SPI3 中断	FF:0283H	80
DMA_UR1T 中断	FF:0193H	50
DMA_UR1R 中断	FF:019BH	51
DMA_UR2T 中断	FF:01A3H	52
DMA_UR2R 中断	FF:01ABH	53
DMA_UR3T 中断	FF:01B3H	54
DMA_UR3R 中断	FF:01BBH	55
DMA_UR4T 中断	FF:01C3H	56
DMA_UR4R 中断	FF:01CBH	57
DMA_UR5T 中断	FF:028BH	82
DMA_UR5R 中断	FF:0293H	83
DMA_UR6T 中断	FF:029BH	84
DMA_UR6R 中断	FF:02A3H	85
DMA_UR7T 中断	FF:02ABH	86
DMA_UR7R 中断	FF:02B3H	87
DMA_UR8T 中断	FF:02BBH	88
DMA_UR8R 中断	FF:02C3H	89
TFT 彩屏 DMA 中断	FF:01D3H	58
DMA_I2CT 中断	FF:01E3H	60
DMA_I2CR 中断	FF:01EBH	61
DMA_I2C2T 中断	FF:0223H	68
DMA_I2C2R 中断	FF:022BH	69
DMA_I2ST 中断	FF:01FBH	63
DMA_I2SR 中断	FF:0203H	64
DMA_I2S2T 中断	FF:0233H	70
DMA_I2S2R 中断	FF:023BH	71
DMA_QSPI 中断	FF:020BH	65
DMA_PWMAT 中断	FF:0243H	72
DMA_PWMAR 中断	FF:024BH	73
DMA_PWMCT 中断	FF:0253H	74
DMA_PWMCR 中断	FF:025BH	75

中断源	中断向量 中断入口地址	Keil C 中断号 相同优先级内的查询次序
DMA_PWMET 中断	FF:03EBH	125
DMA_PWMER 中断	FF:03F3H	126
DMA_CAN1T 中断	FF:03ABH	117
DMA_CAN1R 中断	FF:03B3H	118
DMA_CAN2T 中断	FF:03BBH	119
DMA_CAN2R 中断	FF:03C3H	120

15.22.2 中断使能

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_M2M_CFG	7EFA00H	**M2MIE**	-	TXACO	RXACO	M2MIP[1:0]		M2MPTY[1:0]	

M2MIE：M2M_DMA 中断使能控制位

0：禁止 M2M_DMA 中断

1：允许 M2M_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_ADC_CFG	7EFA10H	**ADCIE**	-			ADCIP[1:0]		ADCPTY[1:0]	
DMA_ADC2_CFG	7EFAE0H	**ADC2IE**	-			ADC2IP[1:0]		ADC2PTY[1:0]	

ADCnIE：ADCn_DMA 中断使能控制位（n=1/2）

0：禁止 ADCn_DMA 中断

1：允许 ADCn_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_SPI_CFG	7EFA20H	**SPIIE**	ACT_TX	ACT_RX	-	SPIIP[1:0]		SPIPTY[1:0]	
DMA_SPI2_CFG	7EF810H	**SPI2IE**	ACT_TX2	ACT_RX2	-	SPI2IP[1:0]		SPI2PTY[1:0]	
DMA_SPI3_CFG	7EF830H	**SPI3IE**	ACT_TX3	ACT_RX3	-	SPI3IP[1:0]		SPI3PTY[1:0]	

SPInIE：SPIn_DMA 中断使能控制位（n=1/2/3）

0：禁止 SPIn_DMA 中断

1：允许 SPIn_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1T_CFG	7EFA30H	**UR1TIE**	-	-	-	UR1TIP[1:0]		UR1TPTY[1:0]	
DMA_UR2T_CFG	7EFA40H	**UR2TIE**	-	-	-	UR2TIP[1:0]		UR2TPTY[1:0]	
DMA_UR3T_CFG	7EFA50H	**UR3TIE**	-	-	-	UR3TIP[1:0]		UR3TPTY[1:0]	
DMA_UR4T_CFG	7EFA60H	**UR4TIE**	-	-	-	UR4TIP[1:0]		UR4TPTY[1:0]	
DMA_UR5T_CFG	7EF750H	**UR5TIE**	-	-	-	UR5TIP[1:0]		UR5TPTY[1:0]	
DMA_UR6T_CFG	7EF780H	**UR6TIE**	-	-	-	UR6TIP[1:0]		UR6TPTY[1:0]	
DMA_UR7T_CFG	7EF7B0H	**UR7TIE**	-	-	-	UR7TIP[1:0]		UR7TPTY[1:0]	
DMA_UR8T_CFG	7EF7E0H	**UR8TIE**	-	-	-	UR8TIP[1:0]		UR8TPTY[1:0]	

URnTIE：URnT_DMA 中断使能控制位（n=1/2/3/4/5/6/7/8）

0：禁止 URnT_DMA 中断

1：允许 URnT_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1R_CFG	7EFA38H	**UR1RIE**	-	-	-	UR1RIP[1:0]		UR1RPTY[1:0]	
DMA_UR2R_CFG	7EFA48H	**UR2RIE**	-	-	-	UR2RIP[1:0]		UR2RPTY[1:0]	
DMA_UR3R_CFG	7EFA58H	**UR3RIE**	-	-	-	UR3RIP[1:0]		UR3RPTY[1:0]	
DMA_UR4R_CFG	7EFA68H	**UR4RIE**	-	-	-	UR4RIP[1:0]		UR4RPTY[1:0]	
DMA_UR5R_CFG	7EF760H	**UR5RIE**	-	-	-	UR5RIP[1:0]		UR5RPTY[1:0]	
DMA_UR6R_CFG	7EF790H	**UR6RIE**	-	-	-	UR6RIP[1:0]		UR6RPTY[1:0]	
DMA_UR7R_CFG	7EF7C0H	**UR7RIE**	-	-	-	UR7RIP[1:0]		UR7RPTY[1:0]	
DMA_UR8R_CFG	7EF7F0H	**UR8RIE**	-	-	-	UR8RIP[1:0]		UR8RPTY[1:0]	

URnRIE：URnR_DMA 中断使能控制位（n=1/2/3/4/5/6/7/8）

0：禁止 URnR_DMA 中断

1：允许 URnR_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_LCM_CFG	7EFA70H	**LCMIE**	-	-	-	LCMIP[1:0]		LCMPTY[1:0]	

LCMIE：TFT 彩屏 DMA 中断使能控制位

0：禁止 TFT 彩屏 DMA 中断

1：允许 TFT 彩屏 DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CT_CFG	7EFA98H	**I2CTIE**	-	-	-	I2CTIP[1:0]		I2CTPTY[1:0]	
DMA_I2C2T_CFG	7EF870H	**I2C2TIE**	-	-	-	I2C2TIP[1:0]		I2C2TPTY[1:0]	

I2CnTIE：I2CnT_DMA 中断使能控制位（n=1/2）

0：禁止 I2CnT_DMA 中断

1：允许 I2CnT_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CR_CFG	7EFAA0H	**I2CRIE**	-	-	-	I2CRIP[1:0]		I2CRPTY[1:0]	
DMA_I2C2R_CFG	7EF880H	**I2C2RIE**	-	-	-	I2C2RIP[1:0]		I2C2RPTY[1:0]	

I2CnRIE：I2CnR_DMA 中断使能控制位（n=1/2）

0：禁止 I2CnR_DMA 中断

1：允许 I2CnR_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2ST_CFG	7EFAB0H	**I2STIE**	-	-	-	I2STIP[1:0]		I2STPTY[1:0]	
DMA_I2S2T_CFG	7EF890H	**I2S2TIE**	-	-	-	I2S2TIP[1:0]		I2S2TPTY[1:0]	

I2SnTIE：I2SnT_DMA 中断使能控制位（n=1/2）

0：禁止 I2SnT_DMA 中断

1：允许 I2SnT_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2SR_CFG	7EFAB8H	**I2SRIE**	-	-	-	I2SRIP[1:0]		I2SRPTY[1:0]	
DMA_I2S2R_CFG	7EF8A0H	**I2S2RIE**	-	-	-	I2S2RIP[1:0]		I2S2RPTY[1:0]	

I2SnRIE：I2SnR_DMA 中断使能控制位（n=1/2）

0：禁止 I2SnR_DMA 中断

1：允许 I2SnR_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_QSPI_CFG	7EFAD0H	**QSPIIE**	ACT_WR	ACT_RD	-	QSPIIP[1:0]		QSPIPTY[1:0]	

QSPIIE：QSPI_DMA 中断使能控制位

0：禁止 QSPI_DMA 中断

1：允许 QSPI_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAT_CFG	7EF980H	**PWMATIE**	-	-	-	PWMATIP[1:0]		PWMATPTY[1:0]	
DMA_PWMCT_CFG	7EF9A0H	**PWMCTIE**	-	-	-	PWMCTIP[1:0]		PWMCTPTY[1:0]	
DMA_PWMET_CFG	7EF668H	**PWMETIE**	-	-	-	PWMETIP[1:0]		PWMETPTY[1:0]	

PWMnTIE：PWMnT_DMA 中断使能控制位（n=A/C/E）

0：禁止 PWMnT_DMA 中断

1：允许 PWMnT_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAR_CFG	7EF990H	**PWMARIE**	-	-	-	PWMARIP[1:0]		PWMARPTY[1:0]	
DMA_PWMCR_CFG	7EF9B0H	**PWMCRIE**	-	-	-	PWMCRIP[1:0]		PWMCRPTY[1:0]	
DMA_PWMER_CFG	7EF678H	**PWMERIE**	-	-	-	PWMERIP[1:0]		PWMERPTY[1:0]	

PWMnRIE：PWMnR_DMA 中断使能控制位（n=A/C/E）

0：禁止 PWMnR_DMA 中断

1：允许 PWMnR_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_DAC1_CFG	7EF618H	**DAC1IE**	-			DAC1IP[1:0]		DAC1PTY[1:0]	
DMA_DAC2_CFG	7EF630H	**DAC2IE**	-			DAC2IP[1:0]		DAC2PTY[1:0]	

DACnIE：DACn_DMA 中断使能控制位（n=1/2）

0：禁止 DACn_DMA 中断

1：允许 DACn_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1T_CFG	7EF520H	**CAN1TIE**	-	-	-	CAN1TIP[1:0]		CAN1TPTY[1:0]	
DMA_CAN2T_CFG	7EF540H	**CAN2TIE**	-	-	-	CAN2TIP[1:0]		CAN2TPTY[1:0]	

CANnTIE：CANnT_DMA 中断使能控制位（n=1/2）

0：禁止 CANnT_DMA 中断

1：允许 CANnT_DMA 中断

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1R_CFG	7EF530H	**CAN1RIE**	-	-	-	CAN1RIP[1:0]		CAN1RPTY[1:0]	
DMA_CAN2R_CFG	7EF550H	**CAN2RIE**	-	-	-	CAN2RIP[1:0]		CAN2RPTY[1:0]	

CANnRIE：CANnR_DMA 中断使能控制位（n=1/2）

0：禁止 CANnR_DMA 中断

1：允许 CANnR_DMA 中断

15.22.3 中断请求

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_M2M_STA	7EFA02H	-	-	-	-	-	-	-	**M2MIF**

M2MIF：M2M_DMA 中断请求标志位，标志位需软件清零

DMA_ADC_STA	7EFA12H	-	-	-	-	-	-	-	**ADCIF**
DMA_ADC2_STA	7EFAE2H	-	-	-	-	-	-	-	**ADC2IF**

ADCnIF：ADCn_DMA 中断请求标志位，标志位需软件清零（n=1/2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1T_STA	7EFA32H	-	-	-	-	-	TXOVW	-	**UR1TIF**
DMA_UR2T_STA	7EFA42H	-	-	-	-	-	TXOVW2	-	**UR2TIF**
DMA_UR3T_STA	7EFA52H	-	-	-	-	-	TXOVW3	-	**UR3TIF**
DMA_UR4T_STA	7EFA62H	-	-	-	-	-	TXOVW4	-	**UR4TIF**
DMA_UR5T_STA	7EF752H	-	-	-	-	-	TXOVW5	-	**UR5TIF**
DMA_UR6T_STA	7EF782H	-	-	-	-	-	TXOVW6	-	**UR6TIF**
DMA_UR7T_STA	7EF7B2H	-	-	-	-	-	TXOVW7	-	**UR7TIF**
DMA_UR8T_STA	7EF7E2H	-	-	-	-	-	TXOVW8	-	**UR8TIF**

URnTIF：URnT_DMA 中断请求标志位，标志位需软件清零（n=1/2/3/4/5/6/7/8）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1R_STA	7EFA3AH	-	-	-	-	-	-	RXLOSS1	**UR1RIF**
DMA_UR2R_STA	7EFA4AH	-	-	-	-	-	-	RXLOSS2	**UR2RIF**
DMA_UR3R_STA	7EFA5AH	-	-	-	-	-	-	RXLOSS3	**UR3RIF**
DMA_UR4R_STA	7EFA6AH	-	-	-	-	-	-	RXLOSS4	**UR4RIF**
DMA_UR5R_STA	7EF762H	-	-	-	-	-	-	RXLOSS5	**UR5RIF**
DMA_UR6R_STA	7EF792H	-	-	-	-	-	-	RXLOSS6	**UR6RIF**
DMA_UR7R_STA	7EF7C2H	-	-	-	-	-	-	RXLOSS7	**UR7RIF**
DMA_UR8R_STA	7EF7F2H	-	-	-	-	-	-	RXLOSS8	**UR8RIF**

URnRIF：URnR_DMA 中断请求标志位，标志位需软件清零（n=1/2/3/4/5/6/7/8）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_LCM_STA	7EFA72H	-	-	-	-	-	-	TXOVW	**LCMIF**

LCMIF：TFT 彩屏 DMA 中断请求标志位，标志位需软件清零

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CT_STA	7EFA9AH	-	-	-	-	-	TXOVW1	-	**I2CTIF**
DMA_I2C2T_STA	7EF872H	-	-	-	-	-	TXOVW2	-	**I2C2TIF**

I2CnTIF：I2CnT_DMA 中断请求标志位，标志位需软件清零（n=1/2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CR_STA	7EFAA2H	-	-	-	-	-	-	RXLOSS	**I2CRIF**
DMA_I2C2R_STA	7EF882H	-	-	-	-	-	-	RXLOSS2	**I2C2RIF**

I2CnRIF：I2CnR_DMA 中断请求标志位，标志位需软件清零（n=1/2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2ST_STA	7EFAB2H	-	-	-	-	-	TXOVW	-	**I2STIF**
DMA_I2S2T_STA	7EF892H	-	-	-	-	-	TXOVW2	-	**I2S2TIF**

I2SnTIF：I2SnT_DMA 中断请求标志位，标志位需软件清零（n=1/2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2SR_STA	7EFABAH	-	-	-	-	-	-	RXLOSS	**I2SRIF**
DMA_I2S2R_STA	7EF8A2H	-	-	-	-	-	-	RXLOSS2	**I2SnRIF**

I2SnRIF：I2SnR_DMA 中断请求标志位，标志位需软件清零（n=1/2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_QSPI_STA	7EFAD2H	-	-	-	-	-	-	-	**QSPIIF**

QSPIIF：QSPI_DMA 中断请求标志位，标志位需软件清零

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAT_STA	7EF982H	RDOVA	-	-	-	-	TXOVWA	-	**PWMATIF**
DMA_PWMCT_STA	7EF9B2H	RDOVC	-	-	-	-	TXOVWC	-	**PWMCTIF**
DMA_PWMET_STA	7EF66AH	RDOVE	-	-	-	-	TXOVWE	-	**PWMETIF**

PWMnTIF：PWMnT_DMA 中断请求标志位，标志位需软件清零（n=A/C/E）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAR_STA	7EF992H	WROVA	-	-	-	-	-	RXLOSSA	**PWMARIF**
DMA_PWMCR_STA	7EF9B2H	WROVC	-	-	-	-	-	RXLOSSC	**PWMCRIF**
DMA_PWMER_STA	7EF67AH	WROVE	-	-	-	-	-	RXLOSSE	**PWMERIF**

PWMnRIF：PWMnR_DMA 中断请求标志位，标志位需软件清零（n=A/C/E）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_DAC1_STA	7EF61AH	-	-	-	-	-	-	-	**DAC1IF**
DMA_DAC2_STA	7EF632H	-	-	-	-	-	-	-	**DAC2IF**

DACnIF：DACn_DMA 中断请求标志位，标志位需软件清零（n=1/2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1T_STA	7EF522H	-	-	-	-	-	TXOVW1	-	**CAN1TIF**
DMA_CAN2T_STA	7EF542H	-	-	-	-	-	TXOVW2	-	**CAN2TIF**

CANnTIF：CANnT_DMA 中断请求标志位，标志位需软件清零（n=1/2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1R_STA	7EF532H	-	-	-	-	-	-	RXLOSS	**CAN1RIF**
DMA_CAN2R_STA	7EF552H	-	-	-	-	-	-	RXLOSS2	**CAN2RIF**

CANnRIF：CANnR_DMA 中断请求标志位，标志位需软件清零（n=1/2）

15.22.4 中断优先级

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_M2M_CFG	7EFA00H	M2MIE	-	TXACO	RXACO	**M2MIP[1:0]**		M2MPTY[1:0]	

M2MIP[1:0]：M2M_DMA 中断优先级控制位

M2MIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_ADC_CFG	7EFA10H	ADCIE	-			**ADCIP[1:0]**		ADCPTY[1:0]	
DMA_ADC2_CFG	7EFAE0H	ADC2IE	-			**ADC2IP[1:0]**		ADC2PTY[1:0]	

ADCnIP[1:0]：ADCn_DMA 中断优先级控制位（n=1/2）

ADCnIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_SPI_CFG	7EFA20H	SPIIE	ACT_TX	ACT_RX	-	**SPIIP[1:0]**		SPIPTY[1:0]	
DMA_SPI2_CFG	7EF810H	SPI2IE	ACT_TX2	ACT_RX2	-	**SPI2IP[1:0]**		SPI2PTY[1:0]	
DMA_SPI3_CFG	7EF830H	SPI3IE	ACT_TX3	ACT_RX3	-	**SPI3IP[1:0]**		SPI3PTY[1:0]	

SPnIP[1:0]：SPIn_DMA 中断优先级控制位（n=1/2/3）

SPInIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1T_CFG	7EFA30H	UR1TIE	-	-	-	**UR1TIP[1:0]**		UR1TPTY[1:0]	
DMA_UR2T_CFG	7EFA40H	UR2TIE	-	-	-	**UR2TIP[1:0]**		UR2TPTY[1:0]	
DMA_UR3T_CFG	7EFA50H	UR3TIE	-	-	-	**UR3TIP[1:0]**		UR3TPTY[1:0]	
DMA_UR4T_CFG	7EFA60H	UR4TIE	-	-	-	**UR4TIP[1:0]**		UR4TPTY[1:0]	
DMA_UR5T_CFG	7EF750H	UR5TIE	-	-	-	**UR5TIP[1:0]**		UR5TPTY[1:0]	
DMA_UR6T_CFG	7EF780H	UR6TIE	-	-	-	**UR6TIP[1:0]**		UR6TPTY[1:0]	
DMA_UR7T_CFG	7EF7B0H	UR7TIE	-	-	-	**UR7TIP[1:0]**		UR7TPTY[1:0]	
DMA_UR8T_CFG	7EF7E0H	UR8TIE	-	-	-	**UR8TIP[1:0]**		UR8TPTY[1:0]	

URnTIP[1:0]：URnT_DMA 中断优先级控制位（n=1/2/3/4/5/6/7/8）

URnTIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1R_CFG	7EFA38H	UR1RIE	-	-	-	**UR1RIP[1:0]**		UR1RPTY[1:0]	
DMA_UR2R_CFG	7EFA48H	UR2RIE	-	-	-	**UR2RIP[1:0]**		UR2RPTY[1:0]	
DMA_UR3R_CFG	7EFA58H	UR3RIE	-	-	-	**UR3RIP[1:0]**		UR3RPTY[1:0]	
DMA_UR4R_CFG	7EFA68H	UR4RIE	-	-	-	**UR4RIP[1:0]**		UR4RPTY[1:0]	
DMA_UR5R_CFG	7EF760H	UR5RIE	-	-	-	**UR5RIP[1:0]**		UR5RPTY[1:0]	
DMA_UR6R_CFG	7EF790H	UR6RIE	-	-	-	**UR6RIP[1:0]**		UR6RPTY[1:0]	
DMA_UR7R_CFG	7EF7C0H	UR7RIE	-	-	-	**UR7RIP[1:0]**		UR7RPTY[1:0]	
DMA_UR8R_CFG	7EF7F0H	UR8RIE	-	-	-	**UR8RIP[1:0]**		UR8RPTY[1:0]	

URnRIP[1:0]：URnR_DMA 中断优先级控制位（n=1/2/3/4/5/6/7/8）

URnRIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_LCM_CFG	7EFA70H	LCMIE	-	-	-	**LCMIP[1:0]**		LCMPTY[1:0]	

LCMIE：TFT 彩屏 DMA 中断使能控制位

0：禁止 TFT 彩屏 DMA 中断

1：允许 TFT 彩屏 DMA 中断

LCMIP[1:0]：TFT 彩屏 DMA 中断优先级控制位

LCMIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CT_CFG	7EFA98H	I2CTIE	-	-	-	**I2CTIP[1:0]**		I2CTPTY[1:0]	
DMA_I2C2T_CFG	7EF870H	I2C2TIE	-	-	-	**I2C2TIP[1:0]**		I2C2TPTY[1:0]	

I2CnTIP[1:0]：I2CnT_DMA 中断优先级控制位（n=1/2）

I2CnTIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2ST_CFG	7EFAB0H	I2STIE	-	-	-	**I2STIP[1:0]**		I2STPTY[1:0]	
DMA_I2S2T_CFG	7EF890H	I2S2TIE	-	-	-	**I2S2TIP[1:0]**		I2S2TPTY[1:0]	

I2SnTIP[1:0]：I2SnT_DMA 中断优先级控制位（n=1/2）

I2SnTIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CR_CFG	7EFAA0H	I2CRIE	-	-	-	**I2CRIP[1:0]**		I2CRPTY[1:0]	
DMA_I2C2R_CFG	7EF880H	I2C2RIE	-	-	-	**I2C2RIP[1:0]**		I2C2RPTY[1:0]	

I2CnRIP[1:0]：I2CnR_DMA 中断优先级控制位（n=1/2）

I2CnRIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2SR_CFG	7EFAB8H	I2SRIE	-	-	-	**I2SRIP[1:0]**		I2SRPTY[1:0]	
DMA_I2S2R_CFG	7EF8A0H	I2S2RIE	-	-	-	**I2S2RIP[1:0]**		I2S2RPTY[1:0]	

I2SnRIP[1:0]：I2SnR_DMA 中断优先级控制位（n=1/2）

I2SnRIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_QSPI_CFG	7EFAD0H	QSPIIE	ACT_WR	ACT_RD	-	**QSPIIP[1:0]**		QSPIPTY[1:0]	

QSPIIP[1:0]：QSPI_DMA 中断优先级控制位

QSPIIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAT_CFG	7EF980H	PWMATIE	-	-	-	**PWMATIP[1:0]**		PWMATPTY[1:0]	
DMA_PWMCT_CFG	7EF9A0H	PWMCTIE	-	-	-	**PWMCTIP[1:0]**		PWMCTPTY[1:0]	
DMA_PWMET_CFG	7EF668H	PWMETIE	-	-	-	**PWMETIP[1:0]**		PWMETPTY[1:0]	

PWMnTIP[1:0]：PWMnT_DMA 中断优先级控制位（n=A/C/E）

PWMnTIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAR_CFG	7EF990H	PWMARIE	-	-	-	**PWMARIP[1:0]**		PWMARPTY[1:0]	
DMA_PWMCR_CFG	7EF9B0H	PWMCRIE	-	-	-	**PWMCRIP[1:0]**		PWMCRPTY[1:0]	
DMA_PWMER_CFG	7EF678H	PWMERIE	-	-	-	**PWMERIP[1:0]**		PWMERPTY[1:0]	

PWMnRIP[1:0]：PWMnR_DMA 中断优先级控制位（n=A/C/E）

PWMnRIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_DAC1_CFG	7EF618H	DAC1IE	-			**DAC1IP[1:0]**		DAC1PTY[1:0]	
DMA_DAC2_CFG	7EF630H	DAC2IE	-			**DAC2IP[1:0]**		DAC2PTY[1:0]	

DACnIP[1:0]：DACn_DMA 中断优先级控制位（n=1/2）

DACnIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1T_CFG	7EF520H	CAN1TIE	-	-	-	**CAN1TIP[1:0]**		CAN1TPTY[1:0]	
DMA_CAN2T_CFG	7EF540H	CAN2TIE	-	-	-	**CAN2TIP[1:0]**		CAN2TPTY[1:0]	

CANnTIP[1:0]：CANnT_DMA 中断优先级控制位（n=1/2）

CANnTIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1R_CFG	7EF530H	CAN1RIE	-	-	-	**CAN1RIP[1:0]**		CAN1RPTY[1:0]	
DMA_CAN2R_CFG	7EF550H	CAN2RIE	-	-	-	**CAN2RIP[1:0]**		CAN2RPTY[1:0]	

CANnRIP[1:0]：CANnR_DMA 中断优先级控制位（n=1/2）

CANnRIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

16 普通 I/O 口均可中断，不是传统外部中断

产品线	I/O 中断	I/O 中断优先级	I/O 中断唤醒功能
STC32G144K246 系列	●	**4 级**	●

STC32G144K246 系列支持所有的 I/O 中断，且支持 5 种中断模式：下降沿中断、上升沿中断、低电平中断、高电平中断、下降沿+上升沿中断。每组 I/O 口都有独立的中断入口地址，且每个 I/O 可独立设置中断模式。

16.1 I/O 口中断相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
P0INTE	P0 口中断使能寄存器	7EFD00H	P07INTE	P06INTE	P05INTE	P04INTE	P03INTE	P02INTE	P01INTE	P00INTE	0000,0000
P1INTE	P1 口中断使能寄存器	7EFD01H	P17INTE	P16INTE	P15INTE	P14INTE	P13INTE	P12INTE	P11INTE	P10INTE	0000,0000
P2INTE	P2 口中断使能寄存器	7EFD02H	P27INTE	P26INTE	P25INTE	P24INTE	P23INTE	P22INTE	P21INTE	P20INTE	0000,0000
P3INTE	P3 口中断使能寄存器	7EFD03H	P37INTE	P36INTE	P35INTE	P34INTE	P33INTE	P32INTE	P31INTE	P30INTE	0000,0000
P4INTE	P4 口中断使能寄存器	7EFD04H	P47INTE	P46INTE	P45INTE	P44INTE	P43INTE	P42INTE	P41INTE	P40INTE	0000,0000
P5INTE	P5 口中断使能寄存器	7EFD05H	P57INTE	P56INTE	P55INTE	P54INTE	P53INTE	P52INTE	P51INTE	P50INTE	xx00,0000
P6INTE	P6 口中断使能寄存器	7EFD06H	P67INTE	P66INTE	P65INTE	P64INTE	P63INTE	P62INTE	P61INTE	P60INTE	0000,0000
P7INTE	P7 口中断使能寄存器	7EFD07H	P77INTE	P76INTE	P75INTE	P74INTE	P73INTE	P72INTE	P71INTE	P70INTE	0000,0000
P8INTE	P8 口中断使能寄存器	7EFD08H	P87INTE	P86INTE	P85INTE	P84INTE	P83INTE	P82INTE	P81INTE	P80INTE	0000,0000
P9INTE	P9 口中断使能寄存器	7EFD09H	P97INTE	P96INTE	P95INTE	P94INTE	P93INTE	P92INTE	P91INTE	P90INTE	0000,0000
PAINTE	PA 口中断使能寄存器	7EFD0AH	PA7INTE	PA6INTE	PA5INTE	PA4INTE	PA3INTE	PA2INTE	PA1INTE	PA0INTE	0000,0000
PBINTE	PB 口中断使能寄存器	7EFD0BH	PB7INTE	PB6INTE	PB5INTE	PB4INTE	PB3INTE	PB2INTE	PB1INTE	PB0INTE	0000,0000
P0INTF	P0 口中断标志寄存器	7EFD10H	P07INTF	P06INTF	P05INTF	P04INTF	P03INTF	P02INTF	P01INTF	P00INTF	0000,0000
P1INTF	P1 口中断标志寄存器	7EFD11H	P17INTF	P16INTF	P15INTF	P14INTF	P13INTF	P12INTF	P11INTF	P10INTF	0000,0000
P2INTF	P2 口中断标志寄存器	7EFD12H	P27INTF	P26INTF	P25INTF	P24INTF	P23INTF	P22INTF	P21INTF	P20INTF	0000,0000
P3INTF	P3 口中断标志寄存器	7EFD13H	P37INTF	P36INTF	P35INTF	P34INTF	P33INTF	P32INTF	P31INTF	P30INTF	0000,0000
P4INTF	P4 口中断标志寄存器	7EFD14H	P47INTF	P46INTF	P45INTF	P44INTF	P43INTF	P42INTF	P41INTF	P40INTF	0000,0000
P5INTF	P5 口中断标志寄存器	7EFD15H	P57INTF	P56INTF	P55INTF	P54INTF	P53INTF	P52INTF	P51INTF	P50INTF	xx00,0000
P6INTF	P6 口中断标志寄存器	7EFD16H	P67INTF	P66INTF	P65INTF	P64INTF	P63INTF	P62INTF	P61INTF	P60INTF	0000,0000
P7INTF	P7 口中断标志寄存器	7EFD17H	P77INTF	P76INTF	P75INTF	P74INTF	P73INTF	P72INTF	P71INTF	P70INTF	0000,0000
P8INTF	P8 口中断标志寄存器	7EFD18H	P87INTF	P86INTF	P85INTF	P84INTF	P83INTF	P82INTF	P81INTF	P80INTF	0000,0000
P9INTF	P9 口中断标志寄存器	7EFD19H	P97INTF	P96INTF	P95INTF	P94INTF	P93INTF	P92INTF	P91INTF	P90INTF	0000,0000
PAINTF	PA 口中断标志寄存器	7EFD1AH	PA7INTF	PA6INTF	PA5INTF	PA4INTF	PA3INTF	PA2INTF	PA1INTF	PA0INTF	0000,0000
PBINTF	PB 口中断标志寄存器	7EFD1BH	PB7INTF	PB6INTF	PB5INTF	PB4INTF	PB3INTF	PB2INTF	PB1INTF	PB0INTF	0000,0000
P0IM0	P0 口中断模式寄存器 0	7EFD20H	P07IM0	P06IM0	P05IM0	P04IM0	P03IM0	P02IM0	P01IM0	P00IM0	0000,0000
P1IM0	P1 口中断模式寄存器 0	7EFD21H	P17IM0	P16IM0	P15IM0	P14IM0	P13IM0	P12IM0	P11IM0	P10IM0	0000,0000
P2IM0	P2 口中断模式寄存器 0	7EFD22H	P27IM0	P26IM0	P25IM0	P24IM0	P23IM0	P22IM0	P21IM0	P20IM0	0000,0000
P3IM0	P3 口中断模式寄存器 0	7EFD23H	P37IM0	P36IM0	P35IM0	P34IM0	P33IM0	P32IM0	P31IM0	P30IM0	0000,0000
P4IM0	P4 口中断模式寄存器 0	7EFD24H	P47IM0	P46IM0	P45IM0	P44IM0	P43IM0	P42IM0	P41IM0	P40IM0	0000,0000

P5IM0	P5 口中断模式寄存器 0	7EFD25H	P57IM0	P56IM0	P55IM0	P54IM0	P53IM0	P52IM0	P51IM0	P50IM0	xx00,0000
P6IM0	P6 口中断模式寄存器 0	7EFD26H	P67IM0	P66IM0	P65IM0	P64IM0	P63IM0	P62IM0	P61IM0	P60IM0	0000,0000
P7IM0	P7 口中断模式寄存器 0	7EFD27H	P77IM0	P76IM0	P75IM0	P74IM0	P73IM0	P72IM0	P71IM0	P70IM0	0000,0000
P8IM0	P8 口中断模式寄存器 0	7EFD28H	P87IM0	P86IM0	P85IM0	P84IM0	P83IM0	P82IM0	P81IM0	P80IM0	0000,0000
P9IM0	P9 口中断模式寄存器 0	7EFD29H	P97IM0	P96IM0	P95IM0	P94IM0	P93IM0	P92IM0	P91IM0	P90IM0	0000,0000
PAIM0	PA 口中断模式寄存器 0	7EFD2AH	PA7IM0	PA6IM0	PA5IM0	PA4IM0	PA3IM0	PA2IM0	PA1IM0	PA0IM0	0000,0000
PBIM0	PB 口中断模式寄存器 0	7EFD2BH	PB7IM0	PB6IM0	PB5IM0	PB4IM0	PB3IM0	PB2IM0	PB1IM0	PB0IM0	0000,0000
P1IM1	P1 口中断模式寄存器 1	7EFD31H	P17IM1	P16IM1	P15IM1	P14IM1	P13IM1	P12IM1	P11IM1	P10IM1	0000,0000
P2IM1	P2 口中断模式寄存器 1	7EFD32H	P27IM1	P26IM1	P25IM1	P24IM1	P23IM1	P22IM1	P21IM1	P20IM1	0000,0000
P3IM1	P3 口中断模式寄存器 1	7EFD33H	P37IM1	P36IM1	P35IM1	P34IM1	P33IM1	P32IM1	P31IM1	P30IM1	0000,0000
P4IM1	P4 口中断模式寄存器 1	7EFD34H	P47IM1	P46IM1	P45IM1	P44IM1	P43IM1	P42IM1	P41IM1	P40IM1	0000,0000
P5IM1	P5 口中断模式寄存器 1	7EFD35H	P57IM1	P56IM1	P55IM1	P54IM1	P53IM1	P52IM1	P51IM1	P50IM1	xx00,0000
P6IM1	P6 口中断模式寄存器 1	7EFD36H	P67IM1	P66IM1	P65IM1	P64IM1	P63IM1	P62IM1	P61IM1	P60IM1	0000,0000
P7IM1	P7 口中断模式寄存器 1	7EFD37H	P77IM1	P76IM1	P75IM1	P74IM1	P73IM1	P72IM1	P71IM1	P70IM1	0000,0000
P8IM1	P8 口中断模式寄存器 1	7EFD38H	P87IM1	P86IM1	P85IM1	P84IM1	P83IM1	P82IM1	P81IM1	P80IM1	0000,0000
P9IM1	P9 口中断模式寄存器 1	7EFD39H	P97IM1	P96IM1	P95IM1	P94IM1	P93IM1	P92IM1	P91IM1	P90IM1	0000,0000
PAIM1	PA 口中断模式寄存器 1	7EFD3AH	PA7IM1	PA6IM1	PA5IM1	PA4IM1	PA3IM1	PA2IM1	PA1IM1	PA0IM1	0000,0000
PBIM1	PB 口中断模式寄存器 1	7EFD3BH	PB7IM1	PB6IM1	PB5IM1	PB4IM1	PB3IM1	PB2IM1	PB1IM1	PB0IM1	0000,0000
P0IM2	P0 口中断模式寄存器 2	7EF4E0H	P07IM2	P06IM2	P05IM2	P04IM2	P03IM2	P02IM2	P01IM2	P00IM2	0000,0000
P1IM2	P1 口中断模式寄存器 2	7EF4E1H	P17IM2	P16IM2	P15IM2	P14IM2	P13IM2	P12IM2	P11IM2	P10IM2	0000,0000
P2IM2	P2 口中断模式寄存器 2	7EF4E2H	P27IM2	P26IM2	P25IM2	P24IM2	P23IM2	P22IM2	P21IM2	P20IM2	0000,0000
P3IM2	P3 口中断模式寄存器 2	7EF4E3H	P37IM2	P36IM2	P35IM2	P34IM2	P33IM2	P32IM2	P31IM2	P30IM2	0000,0000
P4IM2	P4 口中断模式寄存器 2	7EF4E4H	P47IM2	P46IM2	P45IM2	P44IM2	P43IM2	P42IM2	P41IM2	P40IM2	0000,0000
P5IM2	P5 口中断模式寄存器 2	7EF4E5H	P57IM2	P56IM2	P55IM2	P54IM2	P53IM2	P52IM2	P51IM2	P50IM2	0000,0000
P6IM2	P6 口中断模式寄存器 2	7EF4E6H	P67IM2	P66IM2	P65IM2	P64IM2	P63IM2	P62IM2	P61IM2	P60IM2	0000,0000
P7IM2	P7 口中断模式寄存器 2	7EF4E7H	P77IM2	P76IM2	P75IM2	P74IM2	P73IM2	P72IM2	P71IM2	P70IM2	0000,0000
P8IM2	P8 口中断模式寄存器 2	7EF4E8H	P87IM2	P86IM2	P85IM2	P84IM2	P83IM2	P82IM2	P81IM2	P80IM2	0000,0000
P9IM2	P9 口中断模式寄存器 2	7EF4E9H	P97IM2	P96IM2	P95IM2	P94IM2	P93IM2	P92IM2	P91IM2	P90IM2	0000,0000
PAIM2	PA 口中断模式寄存器 2	7EF4EAH	PA7IM2	PA6IM2	PA5IM2	PA4IM2	PA3IM2	PA2IM2	PA1IM2	PA0IM2	0000,0000
PBIM2	PB 口中断模式寄存器 2	7EF4EBH	PB7IM2	PB6IM2	PB5IM2	PB4IM2	PB3IM2	PB2IM2	PB1IM2	PB0IM2	0000,0000
PINIPL	I/O 口中断优先级低寄存器	7EFD40H	P7IP	P6IP	P5IP	P4IP	P3IP	P2IP	P1IP	P0IP	0000,0000
PINIPH	I/O 口中断优先级高寄存器	7EFD41H	P7IPH	P6IPH	P5IPH	P4IPH	P3IPH	P2IPH	P1IPH	P0IPH	0000,0000
PIN2IPL	I/O 口中断优先级低寄存器 2	7EFD62H	-	-	-	-	PBIP	PAIP	P9IP	P8IP	xxxx,0000
PIN2IPH	I/O 口中断优先级高寄存器 2	7EFD63H	-	-	-	-	PBIPH	PAIPH	P9IPH	P8IPH	xxxx,0000
P0WKUE	P0 口中断唤醒使能寄存器	7EFD42H	P07WKUE	P06WKUE	P05WKUE	P04WKUE	P03WKUE	P02WKUE	P01WKUE	P00WKUE	0000,0000
P1WKUE	P1 口中断唤醒使能寄存器	7EFD43H	P17WKUE	P16WKUE	P15WKUE	P14WKUE	P13WKUE	P12WKUE	P11WKUE	P10WKUE	0000,0000
P2WKUE	P2 口中断唤醒使能寄存器	7EFD44H	P27WKUE	P26WKUE	P25WKUE	P24WKUE	P23WKUE	P22WKUE	P21WKUE	P20WKUE	0000,0000
P3WKUE	P3 口中断唤醒使能寄存器	7EFD45H	P37WKUE	P36WKUE	P35WKUE	P34WKUE	P33WKUE	P32WKUE	P31WKUE	P30WKUE	0000,0000
P4WKUE	P4 口中断唤醒使能寄存器	7EFD46H	P47WKUE	P46WKUE	P45WKUE	P44WKUE	P43WKUE	P42WKUE	P41WKUE	P40WKUE	0000,0000
P5WKUE	P5 口中断唤醒使能寄存器	7EFD47H	P57WKUE	P56WKUE	P55WKUE	P54WKUE	P53WKUE	P52WKUE	P51WKUE	P50WKUE	xx00,0000
P6WKUE	P6 口中断唤醒使能寄存器	7EFD60H	P67WKUE	P66WKUE	P65WKUE	P64WKUE	P63WKUE	P62WKUE	P61WKUE	P60WKUE	0000,0000
P7WKUE	P7 口中断唤醒使能寄存器	7EFD61H	P77WKUE	P76WKUE	P75WKUE	P74WKUE	P73WKUE	P72WKUE	P71WKUE	P70WKUE	0000,0000
P8WKUE	P8 口中断唤醒使能寄存器	7EFD48H	P87WKUE	P86WKUE	P85WKUE	P84WKUE	P83WKUE	P82WKUE	P81WKUE	P80WKUE	0000,0000
P9WKUE	P9 口中断唤醒使能寄存器	7EFD49H	P97WKUE	P96WKUE	P95WKUE	P94WKUE	P93WKUE	P92WKUE	P91WKUE	P90WKUE	0000,0000

PAWKUE	PA 口中断唤醒使能寄存器	7EFD4AH	PA7WKUE	PA6WKUE	PA5WKUE	PA4WKUE	PA3WKUE	PA2WKUE	PA1WKUE	PA0WKUE	0000,0000
PBWKUE	PB 口中断唤醒使能寄存器	7EFD4BH	PB7WKUE	PB6WKUE	PB5WKUE	PB4WKUE	PB3WKUE	PB2WKUE	PB1WKUE	PB0WKUE	0000,0000

16.1.1 端口中断使能寄存器（PnINTE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0INTE	7EFD00H	P07INTE	P06INTE	P05INTE	P04INTE	P03INTE	P02INTE	P01INTE	P00INTE
P1INTE	7EFD01H	P17INTE	P16INTE	P15INTE	P14INTE	P13INTE	P12INTE	P11INTE	P10INTE
P2INTE	7EFD02H	P27INTE	P26INTE	P25INTE	P24INTE	P23INTE	P22INTE	P21INTE	P20INTE
P3INTE	7EFD03H	P37INTE	P36INTE	P35INTE	P34INTE	P33INTE	P32INTE	P31INTE	P30INTE
P4INTE	7EFD04H	P47INTE	P46INTE	P45INTE	P44INTE	P43INTE	P42INTE	P41INTE	P40INTE
P5INTE	7EFD05H	P57INTE	P56INTE	P55INTE	P54INTE	P53INTE	P52INTE	P51INTE	P50INTE
P6INTE	7EFD06H	P67INTE	P66INTE	P65INTE	P64INTE	P63INTE	P62INTE	P61INTE	P60INTE
P7INTE	7EFD07H	P77INTE	P76INTE	P75INTE	P74INTE	P73INTE	P72INTE	P71INTE	P70INTE
P8INTE	7EFD08H	P87INTE	P86INTE	P85INTE	P84INTE	P83INTE	P82INTE	P81INTE	P80INTE
P9INTE	7EFD09H	P97INTE	P96INTE	P95INTE	P94INTE	P93INTE	P92INTE	P91INTE	P90INTE
PAINTE	7EFD0AH	PA7INTE	PA6INTE	PA5INTE	PA4INTE	PA3INTE	PA2INTE	PA1INTE	PA0INTE
PBINTE	7EFD0BH	PB7INTE	PB6INTE	PB5INTE	PB4INTE	PB3INTE	PB2INTE	PB1INTE	PB0INTE

PnINTE.x：端口中断使能控制位（n=0,1,2,3,4,5,6,7,8,9,A,B；x=0~7）

0：关闭 Pn.x 口中断功能

1：使能 Pn.x 口中断功能

16.1.2 端口中断标志寄存器（PnINTF）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0INTF	7EFD10H	P07INTF	P06INTF	P05INTF	P04INTF	P03INTF	P02INTF	P01INTF	P00INTF
P1INTF	7EFD11H	P17INTF	P16INTF	P15INTF	P14INTF	P13INTF	P12INTF	P11INTF	P10INTF
P2INTF	7EFD12H	P27INTF	P26INTF	P25INTF	P24INTF	P23INTF	P22INTF	P21INTF	P20INTF
P3INTF	7EFD13H	P37INTF	P36INTF	P35INTF	P34INTF	P33INTF	P32INTF	P31INTF	P30INTF
P4INTF	7EFD14H	P47INTF	P46INTF	P45INTF	P44INTF	P43INTF	P42INTF	P41INTF	P40INTF
P5INTF	7EFD15H	P57INTF	P56INTF	P55INTF	P54INTF	P53INTF	P52INTF	P51INTF	P50INTF
P6INTF	7EFD16H	P67INTF	P66INTF	P65INTF	P64INTF	P63INTF	P62INTF	P61INTF	P60INTF
P7INTF	7EFD17H	P77INTF	P76INTF	P75INTF	P74INTF	P73INTF	P72INTF	P71INTF	P70INTF
P8INTF	7EFD18H	P87INTF	P86INTF	P85INTF	P84INTF	P83INTF	P82INTF	P81INTF	P80INTF
P9INTF	7EFD19H	P97INTF	P96INTF	P95INTF	P94INTF	P93INTF	P92INTF	P91INTF	P90INTF
PAINTF	7EFD1AH	PA7INTF	PA6INTF	PA5INTF	PA4INTF	PA3INTF	PA2INTF	PA1INTF	PA0INTF
PBINTF	7EFD1BH	PB7INTF	PB6INTF	PB5INTF	PB4INTF	PB3INTF	PB2INTF	PB1INTF	PB0INTF

PnINTF.x：端口中断请求标志位

0：Pn.x 口没有中断请求

1：Pn.x 口有中断请求，若使能中断，则会进入中断服务程序。标志位需软件清 0。

16.1.3 端口中断模式配置寄存器（PnIM0，PxIM1，PxIM2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0IM0	7EFD20H	P07IM0	P06IM0	P05IM0	P04IM0	P03IM0	P02IM0	P01IM0	P00IM0
P1IM0	7EFD21H	P17IM0	P16IM0	P15IM0	P14IM0	P13IM0	P12IM0	P11IM0	P10IM0
P2IM0	7EFD22H	P27IM0	P26IM0	P25IM0	P24IM0	P23IM0	P22IM0	P21IM0	P20IM0
P3IM0	7EFD23H	P37IM0	P36IM0	P35IM0	P34IM0	P33IM0	P32IM0	P31IM0	P30IM0
P4IM0	7EFD24H	P47IM0	P46IM0	P45IM0	P44IM0	P43IM0	P42IM0	P41IM0	P40IM0
P5IM0	7EFD25H	P57IM0	P56IM0	P55IM0	P54IM0	P53IM0	P52IM0	P51IM0	P50IM0
P6IM0	7EFD26H	P67IM0	P66IM0	P65IM0	P64IM0	P63IM0	P62IM0	P61IM0	P60IM0
P7IM0	7EFD27H	P77IM0	P76IM0	P75IM0	P74IM0	P73IM0	P72IM0	P71IM0	P70IM0
P8IM0	7EFD28H	P87IM0	P86IM0	P85IM0	P84IM0	P83IM0	P82IM0	P81IM0	P80IM0
P9IM0	7EFD29H	P97IM0	P96IM0	P95IM0	P94IM0	P93IM0	P92IM0	P91IM0	P90IM0
PAIM0	7EFD2AH	PA7IM0	PA6IM0	PA5IM0	PA4IM0	PA3IM0	PA2IM0	PA1IM0	PA0IM0
PBIM0	7EFD2BH	PB7IM0	PB6IM0	PB5IM0	PB4IM0	PB3IM0	PB2IM0	PB1IM0	PB0IM0
P0IM1	7EFD30H	P07IM1	P06IM1	P05IM1	P04IM1	P03IM1	P02IM1	P01IM1	P00IM1
P1IM1	7EFD31H	P17IM1	P16IM1	P15IM1	P14IM1	P13IM1	P12IM1	P11IM1	P10IM1
P2IM1	7EFD32H	P27IM1	P26IM1	P25IM1	P24IM1	P23IM1	P22IM1	P21IM1	P20IM1
P3IM1	7EFD33H	P37IM1	P36IM1	P35IM1	P34IM1	P33IM1	P32IM1	P31IM1	P30IM1
P4IM1	7EFD34H	P47IM1	P46IM1	P45IM1	P44IM1	P43IM1	P42IM1	P41IM1	P40IM1
P5IM1	7EFD35H	P57IM1	P56IM1	P55IM1	P54IM1	P53IM1	P52IM1	P51IM1	P50IM1
P6IM1	7EFD36H	P67IM1	P66IM1	P65IM1	P64IM1	P63IM1	P62IM1	P61IM1	P60IM1
P7IM1	7EFD37H	P77IM1	P76IM1	P75IM1	P74IM1	P73IM1	P72IM1	P71IM1	P70IM1
P8IM1	7EFD38H	P87IM1	P86IM1	P85IM1	P84IM1	P83IM1	P82IM1	P81IM1	P80IM1
P9IM1	7EFD39H	P97IM1	P96IM1	P95IM1	P94IM1	P93IM1	P92IM1	P91IM1	P90IM1
PAIM1	7EFD3AH	PA7IM1	PA6IM1	PA5IM1	PA4IM1	PA3IM1	PA2IM1	PA1IM1	PA0IM1
PBIM1	7EFD3BH	PB7IM1	PB6IM1	PB5IM1	PB4IM1	PB3IM1	PB2IM1	PB1IM1	PB0IM1
P0IM2	7EF4E0H	P07IM2	P06IM2	P05IM2	P04IM2	P03IM2	P02IM2	P01IM2	P00IM2
P1IM2	7EF4E1H	P17IM2	P16IM2	P15IM2	P14IM2	P13IM2	P12IM2	P11IM2	P10IM2
P2IM2	7EF4E2H	P27IM2	P26IM2	P25IM2	P24IM2	P23IM2	P22IM2	P21IM2	P20IM2
P3IM2	7EF4E3H	P37IM2	P36IM2	P35IM2	P34IM2	P33IM2	P32IM2	P31IM2	P30IM2
P4IM2	7EF4E4H	P47IM2	P46IM2	P45IM2	P44IM2	P43IM2	P42IM2	P41IM2	P40IM2
P5IM2	7EF4E5H	P57IM2	P56IM2	P55IM2	P54IM2	P53IM2	P52IM2	P51IM2	P50IM2
P6IM2	7EF4E6H	P67IM2	P66IM2	P65IM2	P64IM2	P63IM2	P62IM2	P61IM2	P60IM2
P7IM2	7EF4E7H	P77IM2	P76IM2	P75IM2	P74IM2	P73IM2	P72IM2	P71IM2	P70IM2
P8IM2	7EF4E8H	P87IM2	P86IM2	P85IM2	P84IM2	P83IM2	P82IM2	P81IM2	P80IM2
P9IM2	7EF4E9H	P97IM2	P96IM2	P95IM2	P94IM2	P93IM2	P92IM2	P91IM2	P90IM2
PAIM2	7EF4EAH	PA7IM2	PA6IM2	PA5IM2	PA4IM2	PA3IM2	PA2IM2	PA1IM2	PA0IM2
PBIM2	7EF4EBH	PB7IM2	PB6IM2	PB5IM2	PB4IM2	PB3IM2	PB2IM2	PB1IM2	PB0IM2

配置端口的模式

PnIM2.x	PnIM1.x	PnIM0.x	Pn.x 口中断模式	掉电唤醒支持
0	0	0	下降沿中断	支持
0	0	1	上升沿中断	支持
0	1	0	低电平中断	不支持
0	1	1	高电平中断	不支持
1	x	x	下降沿+上升沿中断	支持

16.1.4 端口中断优先级控制寄存器（PINnIPL，PINnIPH）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PINIPL	7EFD60H	P7IP	P6IP	P5IP	P4IP	P3IP	P2IP	P1IP	P0IP
PINIPH	7EFD61H	P7IPH	P6IPH	P5IPH	P4IPH	P3IPH	P2IPH	P1IPH	P0IPH
PIN2IPL	7EFD62H	-	-	-	-	PBIP	PAIP	P9IP	P8IP
PIN2IPH	7EFD63H	-	-	-	-	PBIPH	PAIPH	P9IPH	P8IPH

PxIPH,PxIP：Px口中断优先级控制位

00：Px 口中断优先级为 0 级（最低级）

01：Px 口中断优先级为 1 级（较低级）

10：Px 口中断优先级为 2 级（较高级）

11：Px 口中断优先级为 3 级（最高级）

16.1.5 端口中断掉电唤醒使能寄存器（PnWKUE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P0WKUE	7EFD40H	P07WKUE	P06WKUE	P05WKUE	P04WKUE	P03WKUE	P02WKUE	P01WKUE	P00WKUE
P1WKUE	7EFD41H	P17WKUE	P16WKUE	P15WKUE	P14WKUE	P13WKUE	P12WKUE	P11WKUE	P10WKUE
P2WKUE	7EFD42H	P27WKUE	P26WKUE	P25WKUE	P24WKUE	P23WKUE	P22WKUE	P21WKUE	P20WKUE
P3WKUE	7EFD43H	P37WKUE	P36WKUE	P35WKUE	P34WKUE	P33WKUE	P32WKUE	P31WKUE	P30WKUE
P4WKUE	7EFD44H	P47WKUE	P46WKUE	P45WKUE	P44WKUE	P43WKUE	P42WKUE	P41WKUE	P40WKUE
P5WKUE	7EFD45H	P57WKUE	P56WKUE	P55WKUE	P54WKUE	P53WKUE	P52WKUE	P51WKUE	P50WKUE
P6WKUE	7EFD46H	P67WKUE	P66WKUE	P65WKUE	P64WKUE	P63WKUE	P62WKUE	P61WKUE	P60WKUE
P7WKUE	7EFD47H	P77WKUE	P76WKUE	P75WKUE	P74WKUE	P73WKUE	P72WKUE	P71WKUE	P70WKUE
P8WKUE	7EFD48H	P87WKUE	P86WKUE	P85WKUE	P84WKUE	P83WKUE	P82WKUE	P81WKUE	P80WKUE
P9WKUE	7EFD49H	P97WKUE	P96WKUE	P95WKUE	P94WKUE	P93WKUE	P92WKUE	P91WKUE	P90WKUE
PAWKUE	7EFD4AH	PA7WKUE	PA6WKUE	PA5WKUE	PA4WKUE	PA3WKUE	PA2WKUE	PA1WKUE	PA0WKUE
PBWKUE	7EFD4BH	PB7WKUE	PB6WKUE	PB5WKUE	PB4WKUE	PB3WKUE	PB2WKUE	PB1WKUE	PB0WKUE

PnxWKUE：端口中断掉电唤醒使能控制位

0：关闭 Pn.x 口中断掉电唤醒功能

1：使能 Pn.x 口中断掉电唤醒功能**（注意：I/O 口中断模式必须选择下降沿中断模式、上升沿中断模式或者下降沿+上升沿中断模式才能实现掉电唤醒/主时钟停振被唤醒，高电平中断模式或者低电平中断模式无法掉电唤醒/主时钟停振被唤醒）**

17 14 个 24 位定时器/计数器（16 位自动重装载定时器 + 8 位预分频）

产品线	定时器
STC32G144K246 系列	**T0 ~ T11，T17~T18**

STC32G144K246 系列单片机内部设置了 14 个 24 位定时器/计数器（8 位预分频+16 位计数）。所有的定时器都具有计数方式和定时方式两种工作方式。定时器/计数器的核心部件是一个加法计数器，其本质是对脉冲进行计数。只是计数脉冲来源不同：如果计数脉冲来自系统时钟，则为定时方式，此时定时器/计数器每 12 个时钟或者每 1 个时钟得到一个计数脉冲，计数值加 1；如果计数脉冲来自单片机外部引脚，则为计数方式，每来一个脉冲加 1。

当定时器/计数器工作在定时模式时，特殊功能寄存器可设置定时器使用系统时钟/12 还是系统时钟/1（不分频）进行计数。当定时器/计数器工作在计数模式时，对外部脉冲计数不分频。

定时器/计数器 0 有 4 种工作模式：模式 0（16 位自动重装载模式），模式 1（16 位不可重装载模式），模式 2（8 位自动重装模式），模式 3（不可屏蔽中断的 16 位自动重装载模式）。定时器/计数器 1 除模式 3 外，其他工作模式与定时器/计数器 0 相同。T1 在模式 3 时无效，停止计数。定时器 T2 的工作模式固定为 16 位自动重装载模式。T2 可以当定时器使用，也可以当串口的波特率发生器和可编程时钟输出。定时器 3/4/5/6/7/8/9/10/11/17/18 与定时器 T2 一样，它们的工作模式固定为 16 位自动重装载模式。

17.1 定时器 0/1，16 位自动重装载，+ 8 位预分频就是 24 位定时器

17.1.1 定时器相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
TCON	定时器控制寄存器	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	0000,0000
TMOD	定时器模式寄存器	89H	T1_GATE	T1_C/T	T1_M1	T1_M0	T0_GATE	T0_C/T	T0_M1	T0_M0	0000,0000
TL0	定时器 0 低 8 位寄存器	8AH									0000,0000
TL1	定时器 1 低 8 位寄存器	8BH									0000,0000
TH0	定时器 0 高 8 位寄存器	8CH									0000,0000
TH1	定时器 1 高 8 位寄存器	8DH									0000,0000
AUXR	辅助寄存器 1	8EH	T0x12	T1x12	UART_M0x6	T2R	T2_C/T	T2x12	EXTRAM	S1BRT	0000,0001
INTCLKO	中断与时钟输出控制寄存器	8FH	-	EX4	EX3	EX2	-	T2CLKO	T1CLKO	T0CLKO	x000,x000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
TM0PS	定时器 0 时钟预分频寄存器	7EFEA0H									0000,0000
TM1PS	定时器 1 时钟预分频寄存器	7EFEA1H									0000,0000

17.1.2 定时器 0/1 控制寄存器（TCON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TCON	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1：T1溢出中断标志。T1被允许计数以后，从初值开始加1计数。当产生溢出时由硬件将TF1位置“1”，并向CPU请求中断，一直保持到CPU响应中断时，才由硬件清“0”（也可由查询软件清“0”）。

TR1：定时器T1的运行控制位。该位由软件置位和清零。当GATE（TMOD.7）=0，TR1=1时就允许T1开始计数，TR1=0时禁止T1计数。当GATE（TMOD.7）=1，TR1=1且INT1输入高电平时，才允许T1计数。

TF0：T0溢出中断标志。T0被允许计数以后，从初值开始加1计数，当产生溢出时，由硬件置“1”TF0，向CPU请求中断，一直保持CPU响应该中断时，才由硬件清0（也可由查询软件清0）。

TR0：定时器T0的运行控制位。该位由软件置位和清零。当GATE（TMOD.3）=0，TR0=1时就允许T0开始计数，TR0=0时禁止T0计数。当GATE（TMOD.3）=1，TR0=1且INT0输入高电平时，才允许T0计数，TR0=0时禁止T0计数。

IE1：外部中断1请求源（INT1/P3.3）标志。IE1=1，外部中断向CPU请求中断，当CPU响应该中断时由硬件清“0”IE1。

IT1：外部中断源1触发控制位。IT1=0，上升沿或下降沿均可触发外部中断1。IT1=1，外部中断1程控为下降沿触发方式。

IE0：外部中断0请求源（INT0/P3.2）标志。IE0=1外部中断0向CPU请求中断，当CPU响应外部中断时，由硬件清“0”IE0（边沿触发方式）。

IT0：外部中断源0触发控制位。IT0=0，上升沿或下降沿均可触发外部中断0。IT0=1，外部中断0程控为下降沿触发方式。

17.1.3 定时器 0/1 模式寄存器（TMOD）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TMOD	89H	T1_GATE	T1_C/T	T1_M1	T1_M0	T0_GATE	T0_C/T	T0_M1	T0_M0

T1_GATE：控制定时器1，置1时只有在INT1脚为高及TR1控制位置1时才可打开定时器/计数器1。

T0_GATE：控制定时器0，置1时只有在INT0脚为高及TR0控制位置1时才可打开定时器/计数器0。

T1_C/T：控制定时器1用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T1/P3.5外部脉冲进行计数）。

T0_C/T：控制定时器0用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T0/P3.4外部脉冲进行计数）。

T1_M1/T1_M0：定时器定时器/计数器1模式选择

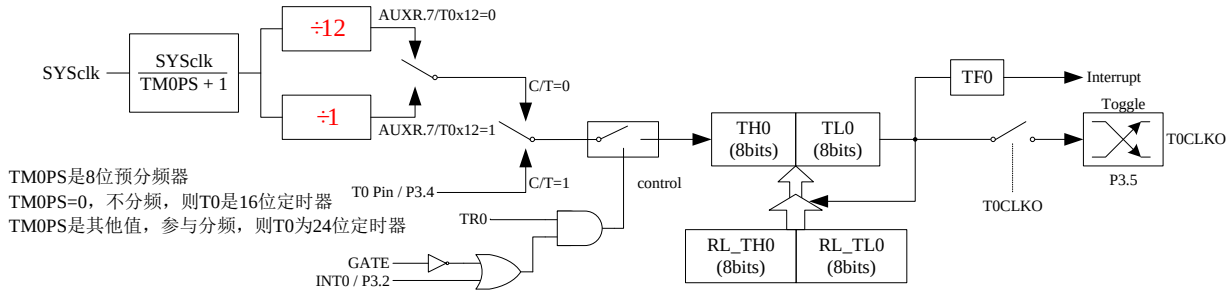
T1_M1	T1_M0	定时器/计数器1工作模式
0	0	16位自动重载模式 当[TH1,TL1]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[TH1,TL1]中。
0	1	16位不自动重载模式 当[TH1,TL1]中的16位计数值溢出时，定时器1将从0开始计数
1	0	8位自动重载模式 当TL1中的8位计数值溢出时，系统会自动将TH1中的重载值装入TL1中。
1	1	T1停止工作

T0_M1/T0_M0：定时器定时器/计数器0模式选择

T0_M1	T0_M0	定时器/计数器0工作模式
0	0	16位自动重载模式 当[TH0,TL0]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[TH0,TL0]中。
0	1	16位不自动重载模式 当[TH0,TL0]中的16位计数值溢出时，定时器0将从0开始计数
1	0	8位自动重载模式 当TL0中的8位计数值溢出时，系统会自动将TH0中的重载值装入TL0中。
1	1	不可屏蔽中断的16位自动重载模式 与模式0相同，不可屏蔽中断，中断优先级最高，高于其他所有中断的优先级，并且不可关闭，可用作操作系统的系统节拍定时器，或者系统监控定时器。

17.1.4 定时器 0 模式 0，16 位自动重装载，启动 8 位预分频就是 24 位定时器

此模式下定时器/计数器 0 作为可自动重装载的 16 位计数器，如下图所示：



T0工作在12T模式（AUXR.7/T0x12=0）时的输出时钟频率 = (SYSclk)/(TM0PS+1)/12/(65536-[RL_TH0, RL_TL0])/2

T0工作在1T模式（AUXR.7/T0x12=1）时的输出时钟频率 = (SYSclk)/(TM0PS+1)/1/(65536-[RL_TH0, RL_TL0])/2

定时器/计数器 0 的模式 0：16 位自动重装载模式，如果启动 8 位预分频 TM0PS 就是 24 位定时器

当 GATE=0（TMOD.3）时，如 TR0=1，则定时器计数。GATE=1 时，允许由外部输入 INT0 控制定时器 0，这样可实现脉宽测量。TR0 为 TCON 寄存器内的控制位，TCON 寄存器各位的具体功能描述见上节 TCON 寄存器的介绍。

当 C/T=0 时，多路开关连接到系统时钟的分频输出，T0 对内部系统时钟计数，T0 工作在定时方式。当 C/T=1 时，多路开关连接到外部脉冲输入 P3.4/T0，即 T0 工作在计数方式。

单片机的定时器 0 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T0 的速率由特殊功能寄存器 AUXR 中的 T0x12 决定，如果 T0x12=0，T0 则工作在 12T 模式；如果 T0x12=1，T0 则工作在 1T 模式

定时器 0 有两个隐藏的寄存器 RL_TH0 和 RL_TL0。RL_TH0 与 TH0 共有同一个地址，RL_TL0 与 TL0 共有同一个地址。当 TR0=0 即定时器/计数器 0 被禁止工作时，对 TL0 写入的内容会同时写入 RL_TL0，对 TH0 写入的内容也会同时写入 RL_TH0。当 TR0=1 即定时器/计数器 0 被允许工作时，对 TL0 写入内容，实际上不是写入当前寄存器 TL0 中，而是写入隐藏的寄存器 RL_TL0 中，对 TH0 写入内容，实际上也不是写入当前寄存器 TH0 中，而是写入隐藏的寄存器 RL_TH0，这样可以巧妙地实现 16 位重装载定时器。当读 TH0 和 TL0 的内容时，所读的内容就是 TH0 和 TL0 的内容，而不是 RL_TH0 和 RL_TL0 的内容。

当定时器 0 工作在模式 0（TMOD[1:0]/[M1,M0]=00B）时，[TH0,TL0]的溢出不仅置位 TF0，而且会自动将[RL_TH0,RL_TL0]的内容重新装入[TH0,TL0]。

当 T0CLKO/INT_CLKO.0=1 时，P3.5/T1 管脚配置为定时器 0 的时钟输出 T0CLKO。输出时钟频率为 T0 溢出率/2。

如果 C/T=0，定时器/计数器 T0 对内部系统时钟计数，则：

T0 工作在 1T 模式（AUXR.7/T0x12=1）时的输出时钟频率 = (SYSclk)/(TM0PS+1)/1/(65536-[RL_TH0, RL_TL0])/2

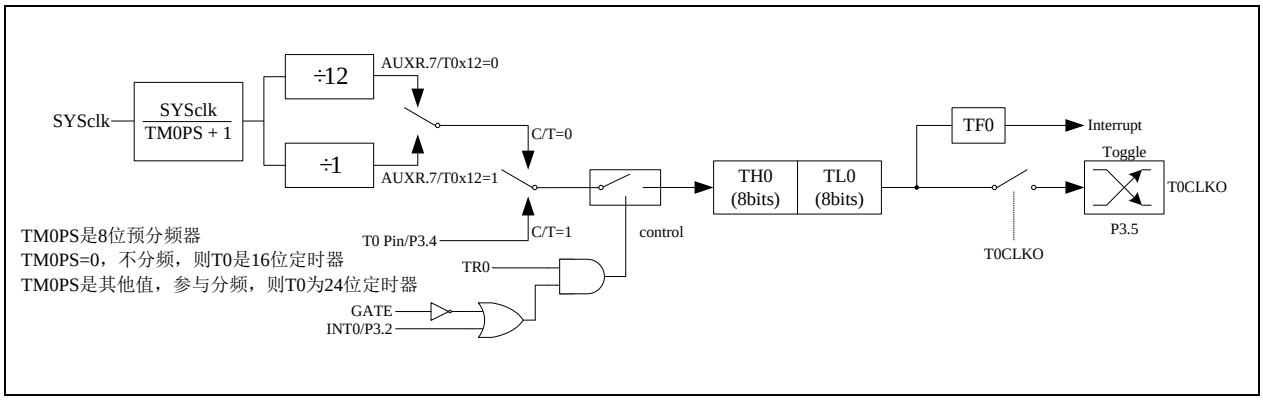
T0 工作在 12T 模式（AUXR.7/T0x12=0）时的输出时钟频率 = (SYSclk)/(TM0PS+1)/12/(65536-[RL_TH0, RL_TL0])/2

如果 C/T=1，定时器/计数器 T0 是对外部脉冲输入(P3.4/T0)计数，则：

输出时钟频率 = (T0_Pin_CLK) / (65536-[RL_TH0, RL_TL0])/2

17.1.5 定时器 0 模式 1（16 位不可重装载模式）

此模式下定时器/计数器 0 工作在 16 位不可重装载模式，如下图所示：



定时器/计数器 0 的模式 1：16 位不可重装载模式

此模式下，定时器/计数器 0 配置为 16 位不可重装载模式，由 TL0 的 8 位和 TH0 的 8 位所构成。TL0 的 8 位溢出向 TH0 进位，TH0 计数溢出置位 TCON 中的溢出标志位 TF0。

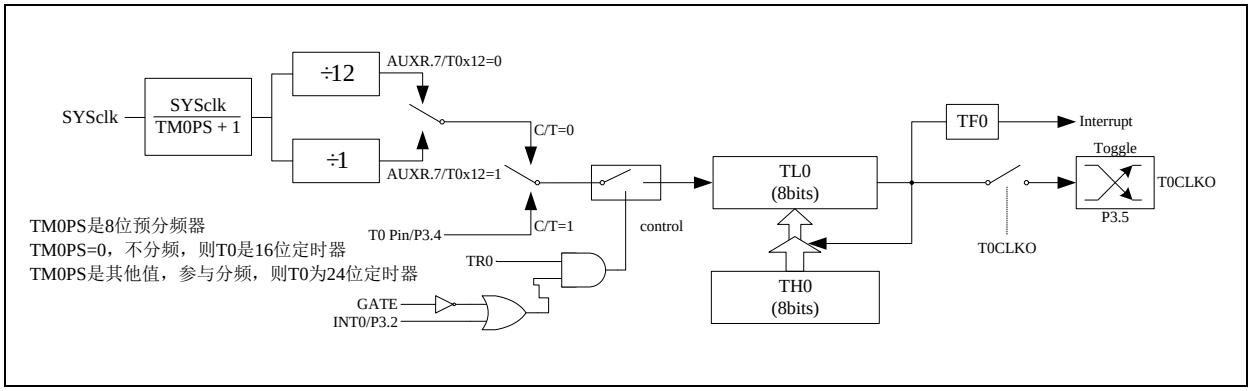
当 GATE=0(TMOD.3)时，如 TR0=1，则定时器计数。GATE=1 时，允许由外部输入 INT0 控制定时器 0，这样可实现脉宽测量。TR0 为 TCON 寄存器内的控制位，TCON 寄存器各位的具体功能描述见上节 TCON 寄存器的介绍。

当 C/T=0 时，多路开关连接到系统时钟的分频输出，T0 对内部系统时钟计数，T0 工作在定时方式。当 C/T=1 时，多路开关连接到外部脉冲输入 P3.4/T0，即 T0 工作在计数方式。

单片机的定时器 0 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T0 的速率由特殊功能寄存器 AUXR 中的 T0x12 决定，如果 T0x12=0，T0 则工作在 12T 模式；如果 T0x12=1，T0 则工作在 1T 模式。

17.1.6 定时器 0 模式 2（8 位自动重装载模式）

此模式下定时器/计数器 0 作为可自动重装载的 8 位计数器，如下图所示：



定时器/计数器 0 的模式 2：8 位自动重装载模式

TL0 的溢出不仅置位 TF0，而且将 TH0 的内容重新装入 TL0，TH0 内容由软件预置，重装时 TH0 内容不变。

当 T0CLKO/INT_CLKO.0=1 时，P3.5/T1 管脚配置为定时器 0 的时钟输出 T0CLKO。输出时钟频率为 T0 溢出率/2。

如果 C/T=0，定时器/计数器 T0 对内部系统时钟计数，则：

T0 工作在 1T 模式（AUXR.7/T0x12=1）时的输出时钟频率 = (SYSclk)/(TM0PS+1)/(256-TH0)/2

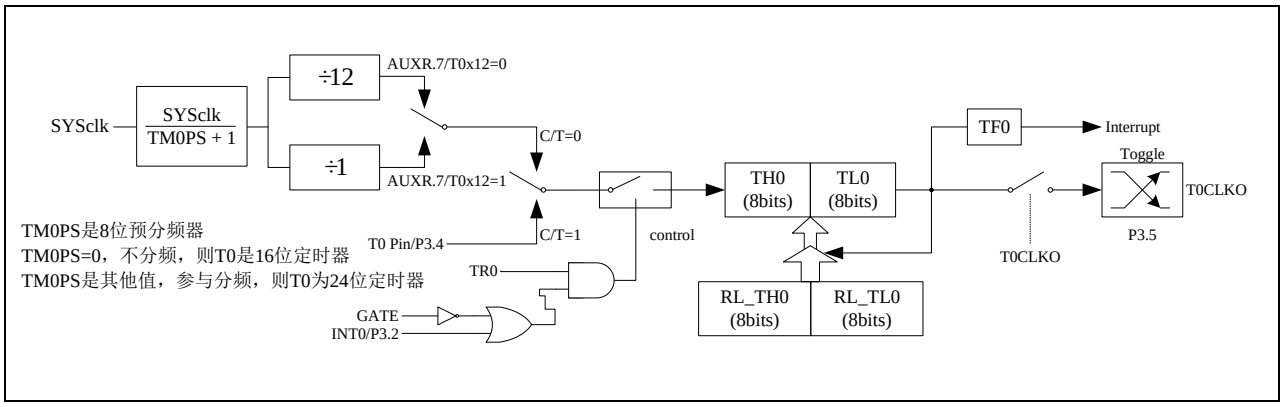
T0 工作在 12T 模式（AUXR.7/T0x12=0）时的输出时钟频率 = (SYSclk)/(TM0PS+1)/12/(256-TH0)/2

如果 C/T=1，定时器/计数器 T0 是对外部脉冲输入(P3.4/T0)计数，则：

输出时钟频率 = (T0_Pin_CLK) / (256-TH0)/2

17.1.7 定时器 0 模式 3（不可屏蔽中断 16 位自动重装载，实时操作系统节拍器）

对定时器/计数器 0，其工作模式模式 3 与工作模式 0 是一样的（下图定时器模式 3 的原理图，与工作模式 0 是一样的）。唯一不同的是：当定时器/计数器 0 工作在模式 3 时，只需允许 ET0/IE.1(定时器/计数器 0 中断允许位)，不需要允许 EA/IE.7(总中断使能位)就能打开定时器/计数器 0 的中断，此模式下的定时器/计数器 0 中断与总中断使能位 EA 无关，一旦工作在模式 3 下的定时器/计数器 0 中断被打开(ET0=1)，那么该中断是不可屏蔽的，该中断的优先级是最高的，即该中断不能被任何中断所打断，而且该中断打开后既不受 EA/IE.7 控制也不再受 ET0 控制，当 EA=0 或 ET0=0 时都不能屏蔽此中断。故将此模式称为不可屏蔽中断的 16 位自动重装载模式。

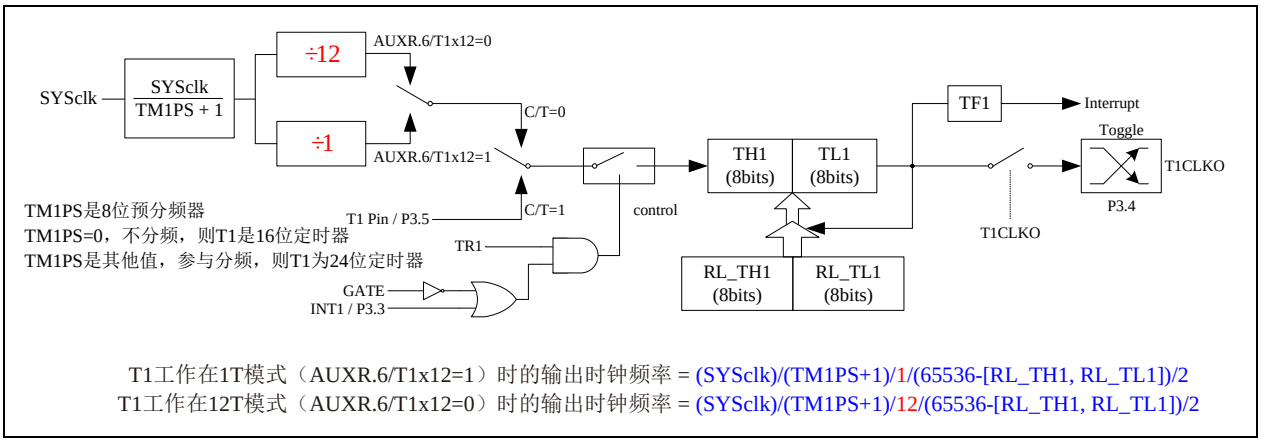


定时器/计数器 0 的模式 3：不可屏蔽中断的 16 位自动重装载模式

注意：当定时器/计数器 0 工作在模式 3(不可屏蔽中断的 16 位自动重装载模式)时，不需要允许 EA/IE.7(总中断使能位)，只需允许 ET0/IE.1(定时器/计数器 0 中断允许位)就能打开定时器/计数器 0 的中断，此模式下的定时器/计数器 0 中断与总中断使能位 EA 无关。一旦此模式下的定时器/计数器 0 中断被打开后，该定时器/计数器 0 中断优先级就是最高的，它不能被其它任何中断所打断(不管是比定时器/计数器 0 中断优先级低的中断还是比其优先级高的中断，都不能打断此时的定时器/计数器 0 中断)，而且该中断打开后既不受 EA/IE.7 控制也不再受 ET0 控制了，清零 EA 或 ET0 都不能关闭此中断。

17.1.8 定时器 1 模式 0，16 位自动重装载，启动 8 位预分频就是 24 位定时器

此模式下定时器/计数器 1 作为可自动重装载的 16 位计数器，如下图所示：



定时器/计数器 1 的模式 0：16 位自动重装载模式，如果启动 8 位预分频 TM1PS 就是 24 位定时器

当 GATE=0（TMOD.7）时，如 TR1=1，则定时器计数。GATE=1 时，允许由外部输入 INT1 控制定时器 1，这样可实现脉宽测量。TR1 为 TCON 寄存器内的控制位，TCON 寄存器各位的具体功能描述见上节 TCON 寄存器的介绍。

当 C/T=0 时，多路开关连接到系统时钟的分频输出，T1 对内部系统时钟计数，T1 工作在定时方式。当 C/T=1 时，多路开关连接到外部脉冲输入 P3.5/T1，即 T1 工作在计数方式。

单片机的定时器 1 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T1 的速率由特殊功能寄存器 AUXR 中的 T1x12 决定，如果 T1x12=0，T1 则工作在 12T 模式；如果 T1x12=1，T1 则工作在 1T 模式

定时器 1 有两个隐藏的寄存器 RL_TH1 和 RL_TL1。RL_TH1 与 TH1 共有同一个地址，RL_TL1 与 TL1 共有同一个地址。当 TR1=0 即定时器/计数器 1 被禁止工作时，对 TL1 写入的内容会同时写入 RL_TL1，对 TH1 写入的内容也会同时写入 RL_TH1。当 TR1=1 即定时器/计数器 1 被允许工作时，对 TL1 写入内容，实际上不是写入当前寄存器 TL1 中，而是写入隐藏的寄存器 RL_TL1 中，对 TH1 写入内容，实际上也不是写入当前寄存器 TH1 中，而是写入隐藏的寄存器 RL_TH1，这样可以巧妙地实现 16 位重装载定时器。当读 TH1 和 TL1 的内容时，所读的内容就是 TH1 和 TL1 的内容，而不是 RL_TH1 和 RL_TL1 的内容。

当定时器 1 工作在模式 1（TMOD[5:4]/[M1,M0]=00B）时，[TH1,TL1]的溢出不仅置位 TF1，而且会自动将[RL_TH1,RL_TL1]的内容重新装入[TH1,TL1]。

当 T1CLKO/INT_CLKO.1=1 时，P3.4/T0 管脚配置为定时器 1 的时钟输出 T1CLKO。输出时钟频率为 T1 溢出率/2。

如果 C/T=0，定时器/计数器 T1 对内部系统时钟计数，则：

T1 工作在 1T 模式（AUXR.6/T1x12=1）时的输出时钟频率 = (SYSclk)/(TM1PS+1)/1/(65536-[RL_TH1, RL_TL1])/2

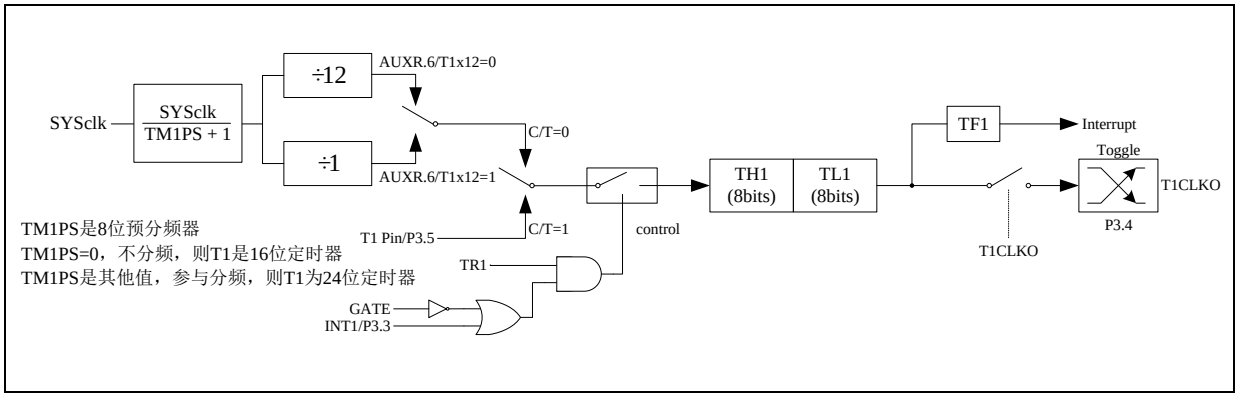
T1 工作在 12T 模式（AUXR.6/T1x12=0）时的输出时钟频率 = (SYSclk)/(TM1PS+1)/12/(65536-[RL_TH1, RL_TL1])/2

如果 C/T=1，定时器/计数器 T1 是对外部脉冲输入(P3.5/T1)计数，则：

输出时钟频率 = (T1_Pin_CLK) / (65536-[RL_TH1, RL_TL1])/2

17.1.9 定时器 1 模式 1（16 位不可重装载模式）

此模式下定时器/计数器 1 工作在 16 位不可重装载模式，如下图所示：



定时器/计数器 1 的模式 1：16 位不可重装载模式

此模式下，定时器/计数器 1 配置为 16 位不可重装载模式，由 TL1 的 8 位和 TH1 的 8 位所构成。TL1 的 8 位溢出向 TH1 进位，TH1 计数溢出置位 TCON 中的溢出标志位 TF1。

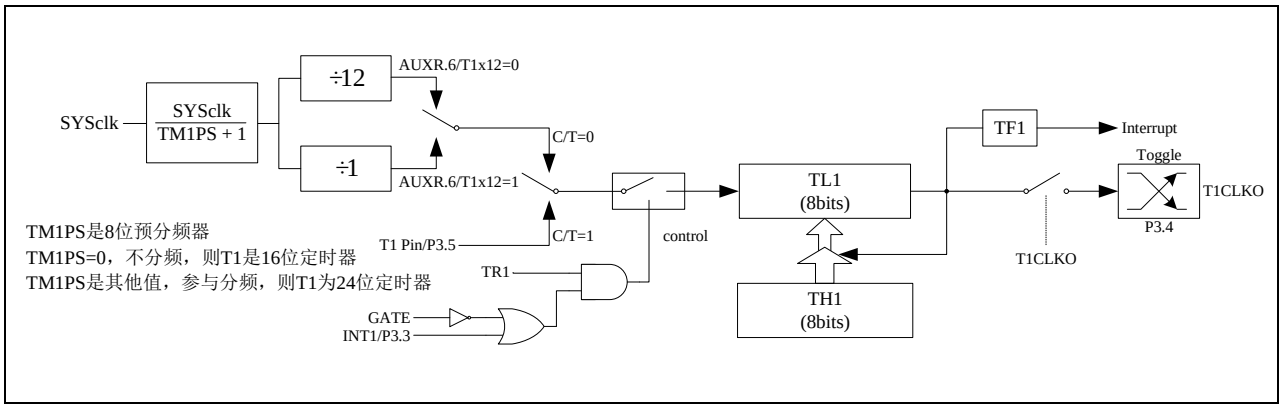
当 GATE=0(TMOD.7)时，如 TR1=1，则定时器计数。GATE=1 时，允许由外部输入 INT1 控制定时器 1，这样可实现脉宽测量。TR1 为 TCON 寄存器内的控制位，TCON 寄存器各位的具体功能描述见上节 TCON 寄存器的介绍。

当 C/T=0 时，多路开关连接到系统时钟的分频输出，T1 对内部系统时钟计数，T1 工作在定时方式。当 C/T=1 时，多路开关连接到外部脉冲输入 P3.5/T1，即 T1 工作在计数方式。

单片机的定时器 1 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T1 的速率由特殊功能寄存器 AUXR 中的 T1x12 决定，如果 T1x12=0，T1 则工作在 12T 模式；如果 T1x12=1，T1 则工作在 1T 模式。

17.1.10 定时器 1 模式 2（8 位自动重装载模式）

此模式下定时器/计数器 1 作为可自动重装载的 8 位计数器，如下图所示：



定时器/计数器 1 的模式 2：8 位自动重装载模式

TL1 的溢出不仅置位 TF1，而且将 TH1 的内容重新装入 TL1，TH1 内容由软件预置，重装时 TH1 内容不变。

当 T1CLKO/INT_CLKO.1=1 时，P3.4/T0 管脚配置为定时器 1 的时钟输出 T1CLKO。输出时钟频率为 T1 溢出率/2。

如果 C/T=0，定时器/计数器 T1 对内部系统时钟计数，则：

T1 工作在 1T 模式（AUXR.6/T1x12=1）时的输出时钟频率 = (SYSclk)/(TM1PS+1)/(256-TH1)/2

T1 工作在 12T 模式（AUXR.6/T1x12=0）时的输出时钟频率 = (SYSclk)/(TM1PS+1)/12/(256-TH1)/2

如果 C/T=1，定时器/计数器 T1 是对外部脉冲输入(P3.5/T1)计数，则：

输出时钟频率 = (T1_Pin_CLK) / (256-TH1)/2

17.1.11 定时器 0 计数寄存器（TL0，TH0）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TL0	8AH								
TH0	8CH								

当定时器/计数器0工作在16位模式（模式0、模式1、模式3）时，TL0和TH0组合成为一个16位寄存器，TL0为低字节，TH0为高字节。若为8位模式（模式2）时，TL0和TH0为两个独立的8位寄存器。

17.1.12 定时器 1 计数寄存器（TL1，TH1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TL1	8BH								
TH1	8DH								

当定时器/计数器1工作在16位模式（模式0、模式1）时，TL1和TH1组合成为一个16位寄存器，TL1为低字节，TH1为高字节。若为8位模式（模式2）时，TL1和TH1为两个独立的8位寄存器。

17.1.13 辅助寄存器 1（AUXR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
AUXR	8EH	T0x12	T1x12	UART_M0x6	T2R	T2_C/T	T2x12	EXTRAM	S1BRT

T0x12：定时器0速度控制位

0：12T 模式，即 CPU 时钟 12 分频（FOSC/12）

1：1T 模式，即 CPU 时钟不分频分频（FOSC/1）

T1x12：定时器1速度控制位

0：12T 模式，即 CPU 时钟 12 分频（FOSC/12）

1：1T 模式，即 CPU 时钟不分频分频（FOSC/1）

17.1.14 中断与时钟输出控制寄存器（INTCLKO）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTCLKO	8FH	-	EX4	EX3	EX2	-	T2CLKO	T1CLKO	T0CLKO

T0CLKO：定时器0时钟输出控制

0：关闭时钟输出

1：使能 P3.5 口的是定时器 0 时钟输出功能

当定时器 0 计数发生溢出时，P3.5 口的电平自动发生翻转。

T1CLKO：定时器1时钟输出控制

0：关闭时钟输出

1：使能 P3.4 口的是定时器 1 时钟输出功能

当定时器 1 计数发生溢出时，P3.4 口的电平自动发生翻转。

17.1.15 定时器 0 的 8 位预分频寄存器（TM0PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TM0PS	7EFEA0H								

定时器0的时钟 ＝ 系统时钟SYSclk ÷ (TM0PS + 1)

17.1.16 定时器 1 的 8 位预分频寄存器（TM1PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TM1PS	7EFEA1H								

定时器1的时钟 ＝ 系统时钟SYSclk ÷ (TM1PS + 1)

17.1.17 定时器 0 计算公式

定时器模式	定时器速度	周期计算公式
模式0/3 （16位自动重载）	1T	定时器周期 = $\frac{65536 - [TH0, TL0]}{SYSclk/(TM0PS+1)}$ （自动重载）
	12T	定时器周期 = $\frac{65536 - [TH0, TL0]}{SYSclk/(TM0PS+1)} \times 12$ （自动重载）
模式1 （16位不自动重载）	1T	定时器周期 = $\frac{65536 - [TH0, TL0]}{SYSclk/(TM0PS+1)}$ （需软件装载）
	12T	定时器周期 = $\frac{65536 - [TH0, TL0]}{SYSclk/(TM0PS+1)} \times 12$ （需软件装载）
模式2 （8位自动重载）	1T	定时器周期 = $\frac{256 - TH0}{SYSclk/(TM0PS+1)}$ （自动重载）
	12T	定时器周期 = $\frac{256 - TH0}{SYSclk/(TM0PS+1)} \times 12$ （自动重载）

17.1.18 定时器 1 计算公式

定时器模式	定时器速度	周期计算公式
模式0 （16位自动重载）	1T	定时器周期 = $\frac{65536 - [TH1, TL1]}{SYSclk/(TM1PS+1)}$ （自动重载）
	12T	定时器周期 = $\frac{65536 - [TH1, TL1]}{SYSclk/(TM1PS+1)} \times 12$ （自动重载）
模式1 （16位不自动重载）	1T	定时器周期 = $\frac{65536 - [TH1, TL1]}{SYSclk/(TM1PS+1)}$ （需软件装载）
	12T	定时器周期 = $\frac{65536 - [TH1, TL1]}{SYSclk/(TM1PS+1)} \times 12$ （需软件装载）
模式2 （8位自动重载）	1T	定时器周期 = $\frac{256 - TH1}{SYSclk/(TM1PS+1)}$ （自动重载）
	12T	定时器周期 = $\frac{256 - TH1}{SYSclk/(TM1PS+1)} \times 12$ （自动重载）

17.2 定时器 2，16 位自动重装载，+ 8 位预分频就是 24 位定时器

17.2.1 定时器相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
AUXR	辅助寄存器 1	8EH	T0x12	T1x12	UART_M0x6	T2R	T2_C/T	T2x12	EXTRAM	S1BRT	0000,0001
INTCLKO	中断与时钟输出控制寄存器	8FH	-	EX4	EX3	EX2	-	T2CLKO	T1CLKO	T0CLKO	x000,x000
T2H	定时器 2 高字节	D6H									0000,0000
T2L	定时器 2 低字节	D7H									0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
TM2PS	定时器 2 时钟预分频寄存器	7EFEA2H									0000,0000

17.2.2 辅助寄存器 1（AUXR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
AUXR	8EH	T0x12	T1x12	UART_M0x6	T2R	T2_C/T	T2x12	EXTRAM	S1BRT

T2R：定时器2的运行控制位

0：定时器 2 停止计数

1：定时器 2 开始计数

T2_C/T：控制定时器2用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T2/P1.0外部脉冲进行计数）。

T2x12：定时器2速度控制位

0：12T 模式，即 CPU 时钟 12 分频（FOSC/12）

1：1T 模式，即 CPU 时钟不分频分频（FOSC/1）

17.2.3 中断与时钟输出控制寄存器（INTCLKO）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTCLKO	8FH	-	EX4	EX3	EX2	-	T2CLKO	T1CLKO	T0CLKO

T2CLKO：定时器2时钟输出控制

0：关闭时钟输出

1：使能 P1.3 口的是定时器 2 时钟输出功能

当定时器 2 计数发生溢出时，P1.3 口的电平自动发生翻转。

17.2.4 定时器 2 计数寄存器（T2L，T2H）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T2L	D7H								
T2H	D6H								

定时器/计数器2的工作模式固定为16位重载模式，T2L和T2H组合成为一个16位寄存器，T2L为低字节，T2H为高字节。当[T2H,T2L]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[T2H,T2L]中。

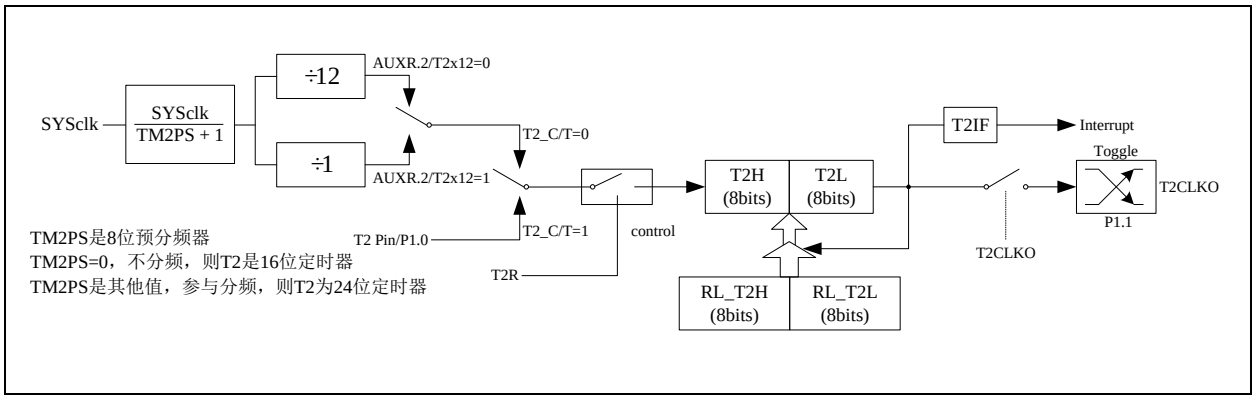
17.2.5 定时器 2 的 8 位预分频寄存器（TM2PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TM2PS	7EFEA2H								

定时器2的时钟 ＝ 系统时钟SYSclk ÷ (TM2PS + 1)

17.2.6 定时器 2 工作模式

定时器/计数器2的原理框图如下：



定时器/计数器 2 的工作模式：16 位自动重装载模式

T2R/AUXR.4 为 AUXR 寄存器内的控制位，AUXR 寄存器各位的具体功能描述见上节 AUXR 寄存器的介绍。

当 T2_C/T=0 时，多路开关连接到系统时钟输出，T2 对内部系统时钟计数，T2 工作在定时方式。当 T2_C/T=1 时，多路开关连接到外部脉冲输 T2，即 T2 工作在计数方式。

单片机的定时器 2 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T2 的速率由特殊功能寄存器 AUXR 中的 T2x12 决定，如果 T2x12=0，T2 则工作在 12T 模式；如果 T2x12=1，T2 则工作在 1T 模式

定时器 2 有两个隐藏的寄存器 RL_T2H 和 RL_T2L。RL_T2H 与 T2H 共有同一个地址，RL_T2L 与 T2L 共有同一个地址。当 T2R=0 即定时器/计数器 2 被禁止工作时，对 T2L 写入的内容会同时写入 RL_T2L，对 T2H 写入的内容也会同时写入 RL_T2H。当 T2R=1 即定时器/计数器 2 被允许工作时，对 T2L 写入内容，实际上不是写入当前寄存器 T2L 中，而是写入隐藏的寄存器 RL_T2L 中，对 T2H 写入内容，实际上也不是写入当前寄存器 T2H 中，而是写入隐藏的寄存器 RL_T2H，这样可以巧妙地实现 16 位重装载定时器。当读 T2H 和 T2L 的内容时，所读的内容就是 T2H 和 T2L 的内容，而不是 RL_T2H 和 RL_T2L 的内容。

[T2H,T2L]的溢出不仅置位中断请求标志位（T2IF），使 CPU 转去执行定时器 2 的中断程序，而且会自动将[RL_T2H,RL_T2L]的内容重新装入[T2H,T2L]。

17.2.7 定时器 2 计算公式

定时器速度	周期计算公式
1T	定时器周期 = $\frac{65536 - [T2H, T2L]}{SYSclk/(TM2PS+1)}$ （自动重载）
12T	定时器周期 = $\frac{65536 - [T2H, T2L]}{SYSclk/(TM2PS+1)} \times 12$ （自动重载）

17.3 定时器 3/4，24 位定时器，8 位预分频+16 位自动重装载定时器

17.3.1 定时器功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TIMERPS	7EFEACH	-	T17T18SEL	T11SEL[1:0]		T9T10SEL	T7T8SEL	T5T6SEL	**T3T4SEL**

T3T4SEL：T3/T4 功能脚切换

T3T4SEL	T3	T3CLKO	T4	T4CLKO
0	P0.4	P0.5	P0.6	P0.7
1	P0.0	P0.1	P0.2	P0.3

17.3.2 定时器相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
T4T3M	定时器 4/3 控制寄存器	DDH	T4R	T4_C/T	T4x12	T4CLKO	T3R	T3_C/T	T3x12	T3CLKO	0000,0000
T4H	定时器 4 高字节	D2H									0000,0000
T4L	定时器 4 低字节	D3H									0000,0000
T3H	定时器 3 高字节	D4H									0000,0000
T3L	定时器 3 低字节	D5H									0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
TM3PS	定时器 3 时钟预分频寄存器	7EFEA3H									0000,0000
TM4PS	定时器 4 时钟预分频寄存器	7EFEA4H									0000,0000

17.3.3 定时器 4/3 控制寄存器（T4T3M）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T4T3M	DDH	T4R	T4_C/T	T4x12	T4CLKO	T3R	T3_C/T	T3x12	T3CLKO

T4R：定时器4的运行控制位

0：定时器 4 停止计数

1：定时器 4 开始计数

T4_C/T：控制定时器4用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T4/P0.6外部脉冲进行计数）。

T4x12：定时器4速度控制位

0：12T 模式，即 CPU 时钟 12 分频（FOSC/12）

1：1T 模式，即 CPU 时钟不分频分频（FOSC/1）

T4CLKO：定时器4时钟输出控制

0：关闭时钟输出

1：使能 P0.7 口的是定时器 4 时钟输出功能

当定时器 4 计数发生溢出时，P0.7 口的电平自动发生翻转。

T3R：定时器3的运行控制位

0：定时器 3 停止计数

1：定时器 3 开始计数

T3_C/T：控制定时器3用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T3/P0.4外部脉冲进行计数）。

T3x12：定时器3速度控制位

0：12T 模式，即 CPU 时钟 12 分频（FOSC/12）

1：1T 模式，即 CPU 时钟不分频分频（FOSC/1）

T3CLKO：定时器3时钟输出控制

0：关闭时钟输出

1：使能 P0.5 口的是定时器 3 时钟输出功能

当定时器 3 计数发生溢出时，P0.5 口的电平自动发生翻转。

17.3.4 定时器 3 计数寄存器（T3L，T3H）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T3L	D5H								
T3H	D4H								

定时器/计数器3的工作模式固定为16位重载模式，T3L和T3H组合成为一个16位寄存器，T3L为低字节，T3H为高字节。当[T3H,T3L]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[T3H,T3L]中。

17.3.5 定时器 4 计数寄存器（T4L，T4H）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T4L	D3H								
T4H	D2H								

定时器/计数器4的工作模式固定为16位重载模式，T4L和T4H组合成为一个16位寄存器，T4L为低字节，T4H为高字节。当[T4H,T4L]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[T4H,T4L]中。

17.3.6 定时器 3 的 8 位预分频寄存器（TM3PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TM3PS	7EFEA3H								

定时器3的时钟 ＝ 系统时钟SYSclk ÷ (TM3PS + 1)

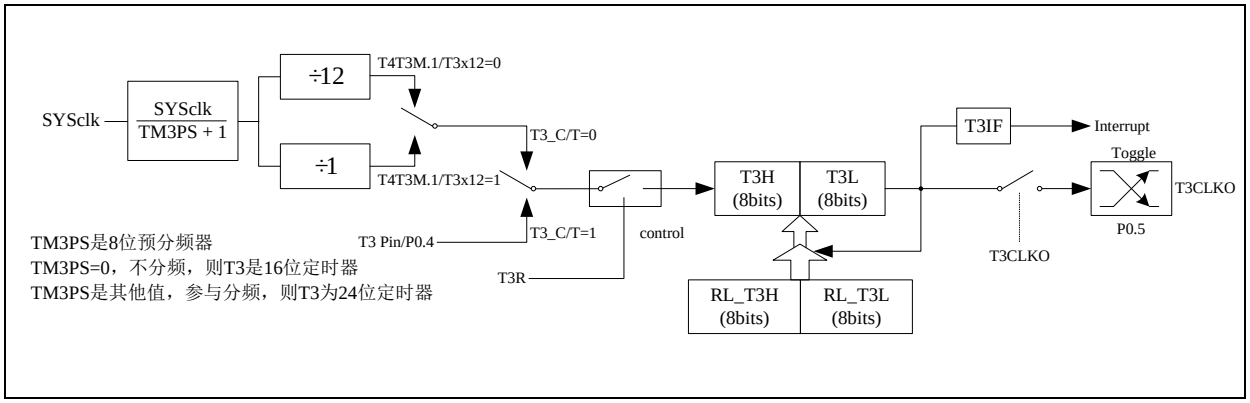
17.3.7 定时器 4 的 8 位预分频寄存器（TM4PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TM4PS	7EFEA4H								

定时器4的时钟 ＝ 系统时钟SYSclk ÷ (TM4PS + 1)

17.3.8 定时器 3 工作模式

定时器/计数器3的原理框图如下：



定时器/计数器 3 的工作模式：16 位自动重装载模式

T3R/T4T3M.3 为 T4T3M 寄存器内的控制位，T4T3M 寄存器各位的具体功能描述见上节 T4T3M 寄存器的介绍。

当 T3_C/T=0 时，多路开关连接到系统时钟输出，T3 对内部系统时钟计数，T3 工作在定时方式。当 T3_C/T=1 时，多路开关连接到外部脉冲输 T3，即 T3 工作在计数方式。

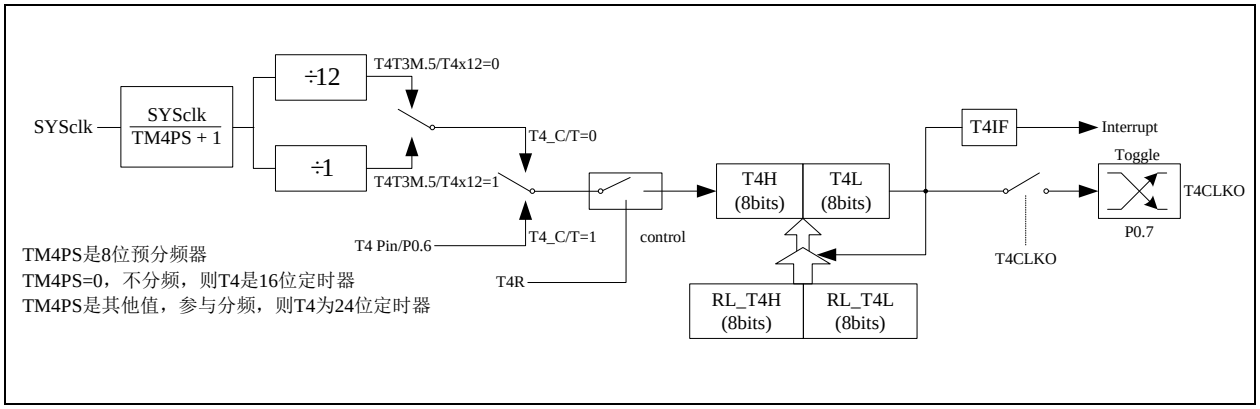
单片机的定时器 3 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T3 的速率由特殊功能寄存器 T4T3M 中的 T3x12 决定，如果 T3x12=0，T3 则工作在 12T 模式；如果 T3x12=1，T3 则工作在 1T 模式

定时器 3 有两个隐藏的寄存器 RL_T3H 和 RL_T3L。RL_T3H 与 T3H 共有同一个地址，RL_T3L 与 T3L 共有同一个地址。当 T3R=0 即定时器/计数器 3 被禁止工作时，对 T3L 写入的内容会同时写入 RL_T3L，对 T3H 写入的内容也会同时写入 RL_T3H。当 T3R=1 即定时器/计数器 3 被允许工作时，对 T3L 写入内容，实际上不是写入当前寄存器 T3L 中，而是写入隐藏的寄存器 RL_T3L 中，对 T3H 写入内容，实际上也不是写入当前寄存器 T3H 中，而是写入隐藏的寄存器 RL_T3H，这样可以巧妙地实现 16 位重装载定时器。当读 T3H 和 T3L 的内容时，所读的内容就是 T3H 和 T3L 的内容，而不是 RL_T3H 和 RL_T3L 的内容。

[T3H,T3L]的溢出不仅置位中断请求标志位（T3IF），使 CPU 转去执行定时器 3 的中断程序，而且会自动将[RL_T3H,RL_T3L]的内容重新装入[T3H,T3L]。

17.3.9 定时器 4 工作模式

定时器/计数器 4 的原理框图如下：



定时器/计数器 4 的工作模式：16 位自动重装载模式

T4R/T4T3M.7 为 T4T3M 寄存器内的控制位，T4T3M 寄存器各位的具体功能描述见上节 T4T3M 寄存器的介绍。

当 T4_C/T=0 时，多路开关连接到系统时钟输出，T4 对内部系统时钟计数，T4 工作在定时方式。当 T4_C/T=1 时，多路开关连接到外部脉冲输 T4，即 T4 工作在计数方式。

单片机的定时器 4 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T4 的速率由特殊功能寄存器 T4T3M 中的 T4x12 决定，如果 T4x12=0，T4 则工作在 12T 模式；如果 T4x12=1，T4 则工作在 1T 模式

定时器 4 有两个隐藏的寄存器 RL_T4H 和 RL_T4L。RL_T4H 与 T4H 共有同一个地址，RL_T4L 与 T4L 共有同一个地址。当 T4R=0 即定时器/计数器 4 被禁止工作时，对 T4L 写入的内容会同时写入 RL_T4L，对 T4H 写入的内容也会同时写入 RL_T4H。当 T4R=1 即定时器/计数器 4 被允许工作时，对 T4L 写入内容，实际上不是写入当前寄存器 T4L 中，而是写入隐藏的寄存器 RL_T4L 中，对 T4H 写入内容，实际上也不是写入当前寄存器 T4H 中，而是写入隐藏的寄存器 RL_T4H，这样可以巧妙地实现 16 位重装载定时器。当读 T4H 和 T4L 的内容时，所读的内容就是 T4H 和 T4L 的内容，而不是 RL_T4H 和 RL_T4L 的内容。

[T4H,T4L]的溢出不仅置位中断请求标志位（T4IF），使 CPU 转去执行定时器 4 的中断程序，而且会自动将[RL_T4H,RL_T4L]的内容重新装入[T4H,T4L]。

17.3.10 定时器 3 计算公式

定时器速度	周期计算公式
1T	定时器周期 = $\frac{65536 - [T3H, T3L]}{SYSclk/(TM3PS+1)}$ （自动重载）
12T	定时器周期 = $\frac{65536 - [T3H, T3L]}{SYSclk/(TM3PS+1)} \times 12$ （自动重载）

17.3.11 定时器 4 计算公式

定时器速度	周期计算公式
1T	定时器周期 = $\frac{65536 - [T4H, T4L]}{SYSclk/(TM4PS+1)}$ （自动重载）
12T	定时器周期 = $\frac{65536 - [T4H, T4L]}{SYSclk/(TM4PS+1)} \times 12$ （自动重载）

17.4 定时器 5/6，24 位定时器，8 位预分频+16 位自动重装载定时器

17.4.1 定时器功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TIMERPS	7EFEACH	-	T17T18SEL	T11SEL[1:0]		T9T10SEL	T7T8SEL	**T5T6SEL**	T3T4SEL

T5T6SEL：T5/T6 功能脚切换

T5T6SEL	T5	T5CLKO	T6	T6CLKO
0	P2.0	P2.1	P2.2	P2.3
1	-	-	-	-

17.4.2 定时器相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
TM5PS	定时器 5 预分频寄存器	7EFBB0H	TM5PS[7:0]								0000,0000
TM6PS	定时器 6 预分频寄存器	7EFBB1H	TM6PS[7:0]								0000,0000
T6T5CFG	定时器 6/5 配置寄存器	7EFBB2H	T6IF	PT6[1:0]		ET6	T5IF	PT5[1:0]		ET5	0000,0000
T6T5M	定时器 6/5 控制寄存器	7EFBB3H	T6R	T6_C/T	T6x12	T6CLKO	T5R	T5_C/T	T5x12	T5CLKO	0000,0000
T6H	定时器 6 高字节	7EFBB4H	TMR6[15:8]								0000,0000
T6L	定时器 6 低字节	7EFBB5H	TMR6[7:0]								0000,0000
T5H	定时器 5 高字节	7EFBB6H	TMR5[15:8]								0000,0000
T5L	定时器 5 低字节	7EFBB7H	TMR5[7:0]								0000,0000

17.4.3 定时器 6/5 控制寄存器（T6T5M）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T6T5M	7EFBB3H	T6R	T6_C/T	T6x12	T6CLKO	T5R	T5_C/T	T5x12	T5CLKO

T6R：定时器6的运行控制位

0：定时器 6 停止计数

1：定时器 6 开始计数

T6_C/T：控制定时器6用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T6/P2.2外部脉冲进行计数）。

T6x12：定时器6速度控制位

0：12T 模式，即 CPU 时钟 12 分频（FOSC/12）

1：1T 模式，即 CPU 时钟不分频分频（FOSC/1）

T6CLKO：定时器6时钟输出控制

0：关闭时钟输出

1：使能 T6CLKO/P2.3 口的是定时器 6 时钟输出功能

当定时器 6 计数发生溢出时，T6CLKO/P2.3 口的电平自动发生翻转。

T5R：定时器5的运行控制位

0：定时器 5 停止计数

1：定时器 5 开始计数

T5_C/T：控制定时器5用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T5/P2.0外部脉冲进行计数）。

T5x12：定时器5速度控制位

0：12T 模式，即 CPU 时钟 12 分频（FOSC/12）

1：1T 模式，即 CPU 时钟不分频分频（FOSC/1）

T5CLKO：定时器5时钟输出控制

0：关闭时钟输出

1：使能 T5CLKO/P2.1 口的是定时器 5 时钟输出功能

当定时器 5 计数发生溢出时，T5CLKO/P2.1 口的电平自动发生翻转。

17.4.4 定时器 6/5 配置寄存器（T6T5CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T6T5CFG	7EFBB2H	T6IF	PT6[1:0]		ET6	T5IF	PT5[1:0]		ET5

ET5：定时/计数器 T5 的溢出中断允许位。

0：禁止 T5 中断

1：允许 T5 中断

T5IF：定时器5溢出中断标志。**需软件写0清零**。

PT5[1:0]：定时器5中断优先级控制位

00：定时器 5 中断优先级为 0 级（最低级）

01：定时器 5 中断优先级为 1 级（较低级）

10：定时器 5 中断优先级为 2 级（较高级）

11：定时器 5 中断优先级为 3 级（最高级）

ET6：定时/计数器 T6 的溢出中断允许位。

0：禁止 T6 中断

1：允许 T6 中断

T6IF：定时器6溢出中断标志。**需软件写0清零**。

PT6[1:0]：定时器6中断优先级控制位

00：定时器 6 中断优先级为 0 级（最低级）

01：定时器 6 中断优先级为 1 级（较低级）

10：定时器 6 中断优先级为 2 级（较高级）

11：定时器 6 中断优先级为 3 级（最高级）

17.4.5 定时器 5 计数寄存器（T5L，T5H）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T5L	7EFBB7H								
T5H	7EFBB6H								

定时器/计数器5的工作模式固定为16位重载模式，T5L和T5H组合成为一个16位寄存器，T5L为低字节，T5H为高字节。当[T5H,T5L]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[T5H,T5L]中。

17.4.6 定时器 6 计数寄存器（T6L，T6H）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T6L	7EFBB5H								
T6H	7EFBB4H								

定时器/计数器6的工作模式固定为16位重载模式，T6L和T6H组合成为一个16位寄存器，T6L为低字节，T6H为高字节。当[T6H,T6L]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[T6H,T6L]中。

17.4.7 定时器 5 的 8 位预分频寄存器（TM5PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TM5PS	7EFBB0H								

定时器5的时钟 ＝ 系统时钟SYSclk ÷ (TM5PS + 1)

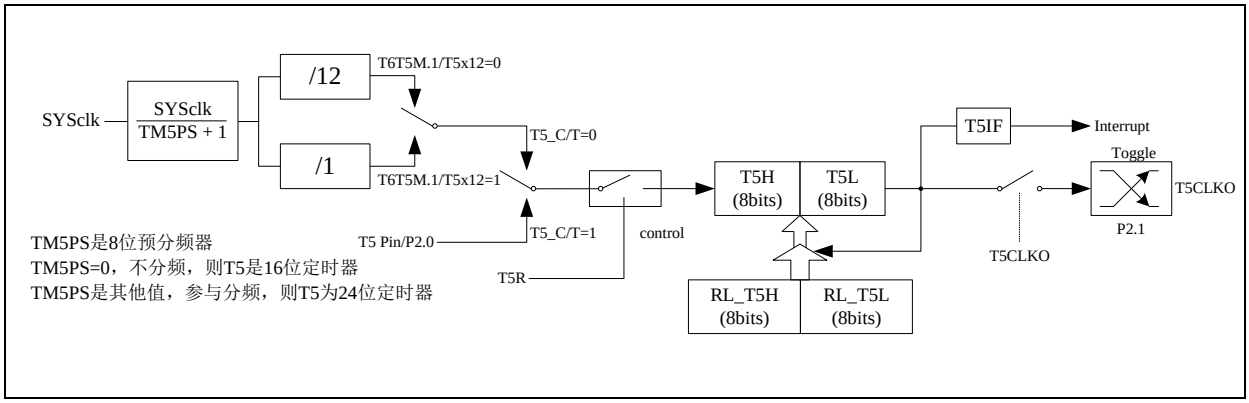
17.4.8 定时器 6 的 8 位预分频寄存器（TM6PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TM6PS	7EFBB1H								

定时器6的时钟 ＝ 系统时钟SYSclk ÷ (TM6PS + 1)

17.4.9 定时器 5 工作模式

定时器/计数器5的原理框图如下：



定时器/计数器 5 的工作模式：16 位自动重装载模式

T5R/T6T5M.3 为 T6T5M 寄存器内的控制位，T6T5M 寄存器各位的具体功能描述见上节 T6T5M 寄存器的介绍。

当 T5_C/T=0 时，多路开关连接到系统时钟输出，T5 对内部系统时钟计数，T5 工作在定时方式。当 T5_C/T=1 时，多路开关连接到外部脉冲输 T5，即 T5 工作在计数方式。

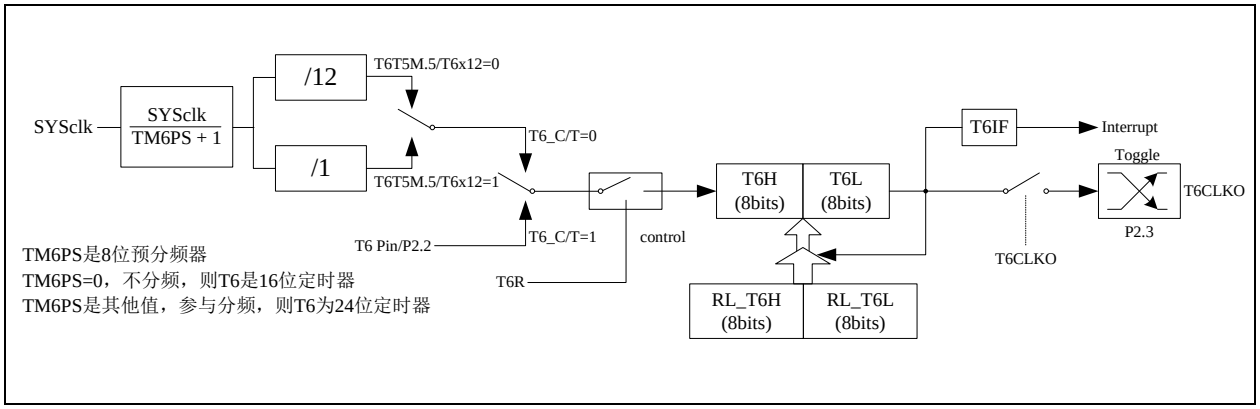
单片机的定时器 5 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T5 的速率由特殊功能寄存器 T6T5M 中的 T5x12 决定，如果 T5x12=0，T5 则工作在 12T 模式；如果 T5x12=1，T5 则工作在 1T 模式

定时器 5 有两个隐藏的寄存器 RL_T5H 和 RL_T5L。RL_T5H 与 T5H 共有同一个地址，RL_T5L 与 T5L 共有同一个地址。当 T5R=0 即定时器/计数器 3 被禁止工作时，对 T5L 写入的内容会同时写入 RL_T5L，对 T5H 写入的内容也会同时写入 RL_T5H。当 T5R=1 即定时器/计数器 3 被允许工作时，对 T5L 写入内容，实际上不是写入当前寄存器 T5L 中，而是写入隐藏的寄存器 RL_T5L 中，对 T5H 写入内容，实际上也不是写入当前寄存器 T5H 中，而是写入隐藏的寄存器 RL_T5H，这样可以巧妙地实现 16 位重装载定时器。当读 T5H 和 T5L 的内容时，所读的内容就是 T5H 和 T5L 的内容，而不是 RL_T5H 和 RL_T5L 的内容。

[T5H,T5L]的溢出不仅置位中断请求标志位（T5IF），使 CPU 转去执行定时器 5 的中断程序，而且会自动将[RL_T5H,RL_T5L]的内容重新装入[T5H,T5L]。

17.4.10 定时器 6 工作模式

定时器/计数器 6 的原理框图如下：



定时器/计数器 6 的工作模式：16 位自动重装载模式

T6R/T6T5M.7 为 T6T5M 寄存器内的控制位，T6T5M 寄存器各位的具体功能描述见上节 T6T5M 寄存器的介绍。

当 T6_C/T=0 时，多路开关连接到系统时钟输出，T6 对内部系统时钟计数，T6 工作在定时方式。当 T6_C/T=1 时，多路开关连接到外部脉冲输 T6，即 T6 工作在计数方式。

单片机的定时器 6 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T6 的速率由特殊功能寄存器 T6T5M 中的 T6x12 决定，如果 T6x12=0，T6 则工作在 12T 模式；如果 T6x12=1，T6 则工作在 1T 模式

定时器 6 有两个隐藏的寄存器 RL_T6H 和 RL_T6L。RL_T6H 与 T6H 共有同一个地址，RL_T6L 与 T6L 共有同一个地址。当 T6R=0 即定时器/计数器 4 被禁止工作时，对 T6L 写入的内容会同时写入 RL_T6L，对 T6H 写入的内容也会同时写入 RL_T6H。当 T6R=1 即定时器/计数器 4 被允许工作时，对 T6L 写入内容，实际上不是写入当前寄存器 T6L 中，而是写入隐藏的寄存器 RL_T6L 中，对 T6H 写入内容，实际上也不是写入当前寄存器 T6H 中，而是写入隐藏的寄存器 RL_T6H，这样可以巧妙地实现 16 位重装载定时器。当读 T6H 和 T6L 的内容时，所读的内容就是 T6H 和 T6L 的内容，而不是 RL_T6H 和 RL_T6L 的内容。

[T6H,T6L]的溢出不仅置位中断请求标志位（T6IF），使 CPU 转去执行定时器 6 的中断程序，而且会自动将[RL_T6H,RL_T6L]的内容重新装入[T6H,T6L]。

17.4.11 定时器 5 计算公式

定时器速度	周期计算公式
1T	定时器周期 = $\frac{65536 - [T5H, T5L]}{SYSclk/(TM5PS+1)}$ （自动重载）
12T	定时器周期 = $\frac{65536 - [T5H, T5L]}{SYSclk/(TM5PS+1)} \times 12$ （自动重载）

17.4.12 定时器 6 计算公式

定时器速度	周期计算公式
1T	定时器周期 = $\frac{65536 - [T6H, T6L]}{SYSclk/(TM6PS+1)}$ （自动重载）
12T	定时器周期 = $\frac{65536 - [T6H, T6L]}{SYSclk/(TM6PS+1)} \times 12$ （自动重载）

17.5 定时器 7/8，24 位定时器，8 位预分频+16 位自动重装载定时器

17.5.1 定时器功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TIMERPS	7EFEACH	-	T17T18SEL	T11SEL[1:0]		T9T10SEL	**T7T8SEL**	T5T6SEL	T3T4SEL

T7T8SEL：T7/T8 功能脚切换

T7T8SEL	T7	T7CLKO	T8	T8CLKO
0	P2.4	P2.5	P2.6	P2.7
1	-	-	-	-

17.5.2 定时器相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
TM7PS	定时器 7 预分频寄存器	7EFBB8H	TM7PS[7:0]								0000,0000
TM8PS	定时器 8 预分频寄存器	7EFBB9H	TM8PS[7:0]								0000,0000
T8T7CFG	定时器 8/7 配置寄存器	7EFBBAH	T8IF	PT8[1:0]		ET8	T7IF	PT7[1:0]		ET7	0000,0000
T8T7M	定时器 8/7 控制寄存器	7EFBBBH	T8R	T8_C/T	T8x12	T8CLKO	T7R	T7_C/T	T7x12	T7CLKO	0000,0000
T8H	定时器 8 高字节	7EFBBCH	TMR8[15:8]								0000,0000
T8L	定时器 8 低字节	7EFBBDH	TMR8[7:0]								0000,0000
T7H	定时器 7 高字节	7EFBBEH	TMR7[15:8]								0000,0000
T7L	定时器 7 低字节	7EFBBFH	TMR7[7:0]								0000,0000

17.5.3 定时器 8/7 控制寄存器（T8T7M）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T8T7M	7EFBBBH	T8R	T8_C/T	T8x12	T8CLKO	T7R	T7_C/T	T7x12	T7CLKO

T8R：定时器8的运行控制位

0：定时器 8 停止计数

1：定时器 8 开始计数

T8_C/T：控制定时器8用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T8/P2.6外部脉冲进行计数）。

T8x12：定时器8速度控制位

0：12T 模式，即 CPU 时钟 12 分频（FOSC/12）

1：1T 模式，即 CPU 时钟不分频分频（FOSC/1）

T8CLKO：定时器8时钟输出控制

0：关闭时钟输出

1：使能 T8CLKO/P2.7 口的是定时器 8 时钟输出功能

当定时器 8 计数发生溢出时，T8CLKO/P2.7 口的电平自动发生翻转。

T7R：定时器7的运行控制位

0：定时器 7 停止计数

1：定时器 7 开始计数

T7_C/T：控制定时器7用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T7/P2.4外部脉冲进行计数）。

T7x12：定时器7速度控制位

0：12T 模式，即 CPU 时钟 12 分频（FOSC/12）

1：1T 模式，即 CPU 时钟不分频分频（FOSC/1）

T7CLKO：定时器7时钟输出控制

0：关闭时钟输出

1：使能 T7CLKO/P2.5 口的是定时器 7 时钟输出功能

当定时器 7 计数发生溢出时，T7CLKO/P2.5 口的电平自动发生翻转。

17.5.4 定时器 8/7 配置寄存器（T8T7CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T8T7CFG	7EFBBAH	T8IF	PT8[1:0]		ET8	T7IF	PT7[1:0]		ET7

ET7：定时/计数器 T7 的溢出中断允许位。

0：禁止 T7 中断

1：允许 T7 中断

T7IF：定时器7溢出中断标志。**需软件写0清零**。

PT7[1:0]：定时器7中断优先级控制位

00：定时器 7 中断优先级为 0 级（最低级）

01：定时器 7 中断优先级为 1 级（较低级）

10：定时器 7 中断优先级为 2 级（较高级）

11：定时器 7 中断优先级为 3 级（最高级）

ET8：定时/计数器 T8 的溢出中断允许位。

0：禁止 T8 中断

1：允许 T8 中断

T8IF：定时器8溢出中断标志。**需软件写0清零**。

PT8[1:0]：定时器8中断优先级控制位

00：定时器 8 中断优先级为 0 级（最低级）

01：定时器 8 中断优先级为 1 级（较低级）

10：定时器 8 中断优先级为 2 级（较高级）

11：定时器 8 中断优先级为 3 级（最高级）

17.5.5 定时器 7 计数寄存器（T7L，T7H）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T7L	7EFBBFH								
T7H	7EFBBEH								

定时器/计数器7的工作模式固定为16位重载模式，T7L和T7H组合成为一个16位寄存器，T7L为低字节，T7H为高字节。当[T7H,T7L]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[T7H,T7L]中。

17.5.6 定时器 8 计数寄存器（T8L，T8H）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T8L	7EFBBDH								
T8H	7EFBBCH								

定时器/计数器8的工作模式固定为16位重载模式，T8L和T8H组合成为一个16位寄存器，T8L为低字节，T8H为高字节。当[T8H,T8L]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[T8H,T8L]中。

17.5.7 定时器 7 的 8 位预分频寄存器（TM7PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TM7PS	7EFBB8H								

定时器7的时钟 ＝ 系统时钟SYSclk ÷ (TM7PS + 1)

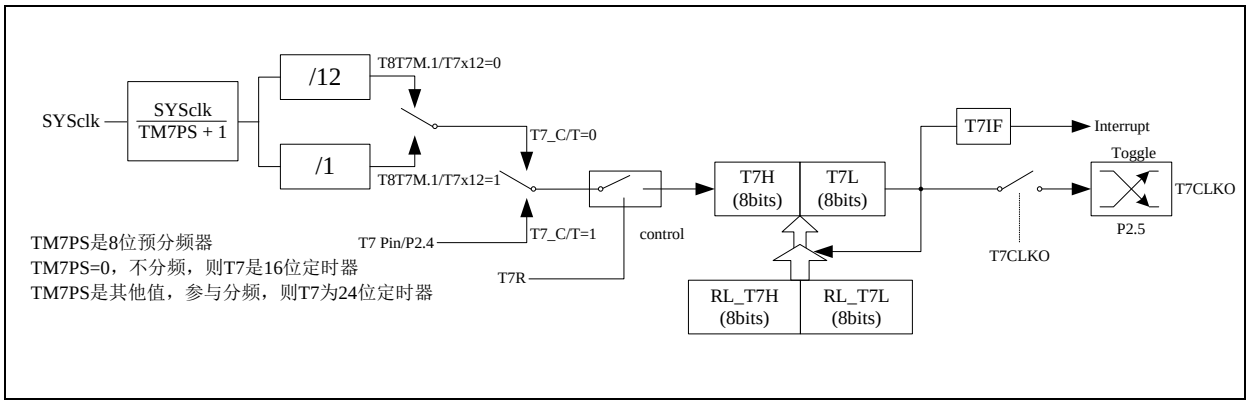
17.5.8 定时器 8 的 8 位预分频寄存器（TM8PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TM8PS	7EFBB9H								

定时器8的时钟 ＝ 系统时钟SYSclk ÷ (TM8PS + 1)

17.5.9 定时器 7 工作模式

定时器/计数器7的原理框图如下：



定时器/计数器 7 的工作模式：16 位自动重装载模式

T7R/T8T7M.3 为 T8T7M 寄存器内的控制位，T8T7M 寄存器各位的具体功能描述见上节 T8T7M 寄存器的介绍。

当 T7_C/T=0 时，多路开关连接到系统时钟输出，T7 对内部系统时钟计数，T7 工作在定时方式。当 T7_C/T=1 时，多路开关连接到外部脉冲输 T7，即 T7 工作在计数方式。

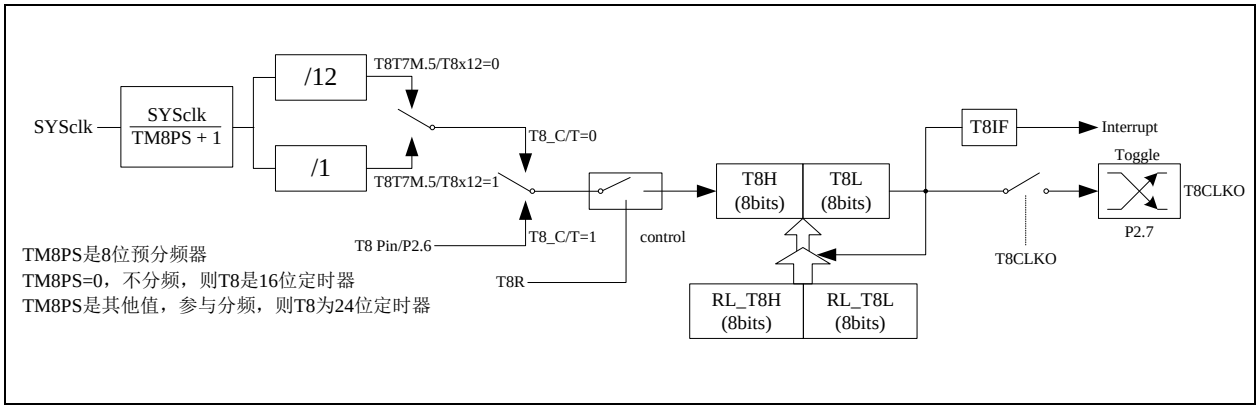
单片机的定时器 7 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T7 的速率由特殊功能寄存器 T8T7M 中的 T7x12 决定，如果 T7x12=0，T7 则工作在 12T 模式；如果 T7x12=1，T7 则工作在 1T 模式

定时器 7 有两个隐藏的寄存器 RL_T7H 和 RL_T7L。RL_T7H 与 T7H 共有同一个地址，RL_T7L 与 T7L 共有同一个地址。当 T7R=0 即定时器/计数器 3 被禁止工作时，对 T7L 写入的内容会同时写入 RL_T7L，对 T7H 写入的内容也会同时写入 RL_T7H。当 T7R=1 即定时器/计数器 3 被允许工作时，对 T7L 写入内容，实际上不是写入当前寄存器 T7L 中，而是写入隐藏的寄存器 RL_T7L 中，对 T7H 写入内容，实际上也不是写入当前寄存器 T7H 中，而是写入隐藏的寄存器 RL_T7H，这样可以巧妙地实现 16 位重装载定时器。当读 T7H 和 T7L 的内容时，所读的内容就是 T7H 和 T7L 的内容，而不是 RL_T7H 和 RL_T7L 的内容。

[T7H,T7L]的溢出不仅置位中断请求标志位（T7IF），使 CPU 转去执行定时器 7 的中断程序，而且会自动将[RL_T7H,RL_T7L]的内容重新装入[T7H,T7L]。

17.5.10 定时器 8 工作模式

定时器/计数器 8 的原理框图如下：



定时器/计数器 8 的工作模式：16 位自动重装载模式

T8R/T8T7M.7 为 T8T7M 寄存器内的控制位，T8T7M 寄存器各位的具体功能描述见上节 T8T7M 寄存器的介绍。

当 T8_C/T=0 时，多路开关连接到系统时钟输出，T8 对内部系统时钟计数，T8 工作在定时方式。当 T8_C/T=1 时，多路开关连接到外部脉冲输 T8，即 T8 工作在计数方式。

单片机的定时器 8 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T8 的速率由特殊功能寄存器 T8T7M 中的 T8x12 决定，如果 T8x12=0，T8 则工作在 12T 模式；如果 T8x12=1，T8 则工作在 1T 模式

定时器 8 有两个隐藏的寄存器 RL_T8H 和 RL_T8L。RL_T8H 与 T8H 共有同一个地址，RL_T8L 与 T8L 共有同一个地址。当 T8R=0 即定时器/计数器 4 被禁止工作时，对 T8L 写入的内容会同时写入 RL_T8L，对 T8H 写入的内容也会同时写入 RL_T8H。当 T8R=1 即定时器/计数器 4 被允许工作时，对 T8L 写入内容，实际上不是写入当前寄存器 T8L 中，而是写入隐藏的寄存器 RL_T8L 中，对 T8H 写入内容，实际上也不是写入当前寄存器 T8H 中，而是写入隐藏的寄存器 RL_T8H，这样可以巧妙地实现 16 位重装载定时器。当读 T8H 和 T8L 的内容时，所读的内容就是 T8H 和 T8L 的内容，而不是 RL_T8H 和 RL_T8L 的内容。

[T8H,T8L]的溢出不仅置位中断请求标志位（T8IF），使 CPU 转去执行定时器 8 的中断程序，而且会自动将[RL_T8H,RL_T8L]的内容重新装入[T8H,T8L]。

17.5.11 定时器 7 计算公式

定时器速度	周期计算公式
1T	定时器周期 = $\frac{65536 - [T7H, T7L]}{SYSclk/(TM7PS+1)}$ （自动重载）
12T	定时器周期 = $\frac{65536 - [T7H, T7L]}{SYSclk/(TM7PS+1)} \times 12$ （自动重载）

17.5.12 定时器 8 计算公式

定时器速度	周期计算公式
1T	定时器周期 = $\frac{65536 - [T8H, T8L]}{SYSclk/(TM8PS+1)}$ （自动重载）
12T	定时器周期 = $\frac{65536 - [T8H, T8L]}{SYSclk/(TM8PS+1)} \times 12$ （自动重载）

17.6 定时器 9/10，24 位定时器，8 位预分频+16 位自动重装载定时器

17.6.1 定时器功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TIMERPS	7EFEACH	-	T17T18SEL	T11SEL[1:0]		**T9T10SEL**	T7T8SEL	T5T6SEL	T3T4SEL

T9T10SEL：T9/T10 功能脚切换

T9T10SEL	T9	T9CLKO	T10	T10CLKO
0	P7.0	P7.1	P6.0	P6.1
1	PB.0	PB.1	PB.6	PB.7

17.6.2 定时器相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
TM9PS	定时器 9 预分频寄存器	7EFBC0H	TM9PS[7:0]								0000,0000
TM10PS	定时器 10 预分频寄存器	7EFBC1H	TM10PS[7:0]								0000,0000
T10T9CFG	定时器 10/9 配置寄存器	7EFBC2H	T10IF	PT10[1:0]		ET10	T9IF	PT9[1:0]		ET9	0000,0000
T10T9M	定时器 10/9 控制寄存器	7EFBC3H	T10R	T10_C/T	T10x12	T10CLKO	T9R	T9_C/T	T9x12	T9CLKO	0000,0000
T10H	定时器 10 高字节	7EFBC4H	TMR10[15:8]								0000,0000
T10L	定时器 10 低字节	7EFBC5H	TMR10[7:0]								0000,0000
T9H	定时器 9 高字节	7EFBC6H	TMR9[15:8]								0000,0000
T9L	定时器 9 低字节	7EFBC7H	TMR9[7:0]								0000,0000

17.6.3 定时器 10/9 控制寄存器（T10T9M）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T10T9M	7EFBC3H	T10R	T10_C/T	T10x12	T10CLKO	T9R	T9_C/T	T9x12	T9CLKO

T10R：定时器10的运行控制位

0：定时器 10 停止计数

1：定时器 10 开始计数

T10_C/T：控制定时器10用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T10/P6.0外部脉冲进行计数）。

T10x12：定时器10速度控制位

0：12T 模式，即 CPU 时钟 12 分频（FOSC/12）

1：1T 模式，即 CPU 时钟不分频分频（FOSC/1）

T10CLKO：定时器10时钟输出控制

0：关闭时钟输出

1：使能 T10CLKO/P6.1 口的是定时器 10 时钟输出功能

当定时器 10 计数发生溢出时，T10CLKO/P6.1 口的电平自动发生翻转。

T9R：定时器9的运行控制位

0：定时器 9 停止计数

1：定时器 9 开始计数

T9_C/T：控制定时器9用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T9/P7.0外部脉冲进行计数）。

T9x12：定时器9速度控制位

0：12T 模式，即 CPU 时钟 12 分频（FOSC/12）

1：1T 模式，即 CPU 时钟不分频分频（FOSC/1）

T9CLKO：定时器9时钟输出控制

0：关闭时钟输出

1：使能 T9CLKO/P7.1 口的是定时器 9 时钟输出功能

当定时器 9 计数发生溢出时，T9CLKO/P7.1 口的电平自动发生翻转。

17.6.4 定时器 10/9 配置寄存器（T10T9CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T10T9CFG	7EFBC2H	T10IF	PT10[1:0]		ET10	T9IF	PT9[1:0]		ET9

ET9：定时/计数器 T9 的溢出中断允许位。

0：禁止 T9 中断

1：允许 T9 中断

T9IF：定时器9溢出中断标志。**需软件写0清零**。

PT9[1:0]：定时器9中断优先级控制位

00：定时器 9 中断优先级为 0 级（最低级）

01：定时器 9 中断优先级为 1 级（较低级）

10：定时器 9 中断优先级为 2 级（较高级）

11：定时器 9 中断优先级为 3 级（最高级）

ET10：定时/计数器 T10 的溢出中断允许位。

0：禁止 T10 中断

1：允许 T10 中断

T10IF：定时器10溢出中断标志。**需软件写0清零**。

PT10[1:0]：定时器10中断优先级控制位

00：定时器 10 中断优先级为 0 级（最低级）

01：定时器 10 中断优先级为 1 级（较低级）

10：定时器 10 中断优先级为 2 级（较高级）

11：定时器 10 中断优先级为 3 级（最高级）

17.6.5 定时器 9 计数寄存器（T9L，T9H）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T9L	7EFBC7H								
T9H	7EFBC6H								

定时器/计数器9的工作模式固定为16位重载模式，T9L和T9H组合成为一个16位寄存器，T9L为低字节，T9H为高字节。当[T9H,T9L]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[T9H,T9L]中。

17.6.6 定时器 10 计数寄存器（T10L，T10H）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T10L	7EFBC5H								
T10H	7EFBC4H								

定时器/计数器10的工作模式固定为16位重载模式，T10L和T10H组合成为一个16位寄存器，T10L为低字节，T10H为高字节。当[T10H,T10L]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[T10H,T10L]中。

17.6.7 定时器 9 的 8 位预分频寄存器（TM9PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TM9PS	7EFBC0H								

定时器9的时钟 ＝ 系统时钟SYSclk ÷ (TM9PS + 1)

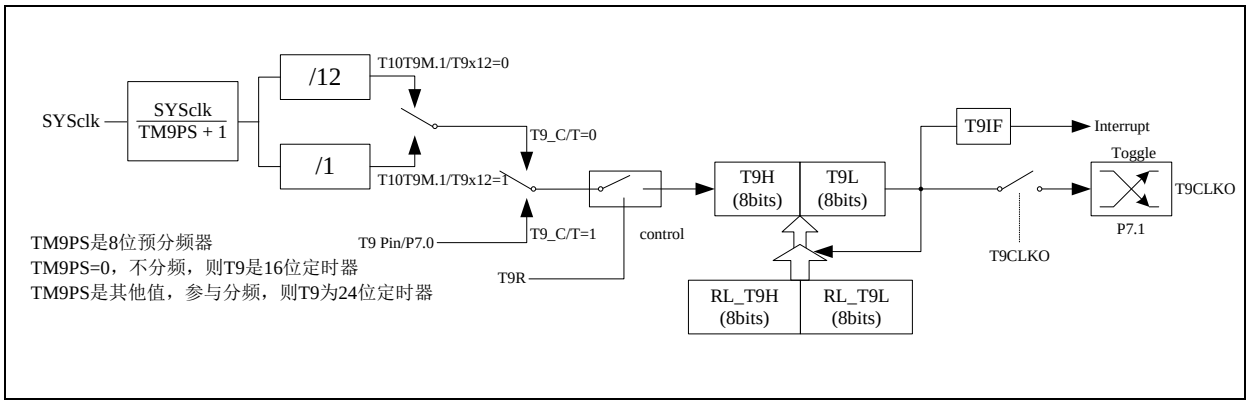
17.6.8 定时器 10 的 8 位预分频寄存器（TM10PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TM10PS	7EFBC1H								

定时器10的时钟 ＝ 系统时钟SYSclk ÷ (TM10PS + 1)

17.6.9 定时器 9 工作模式

定时器/计数器9的原理框图如下：



定时器/计数器 9 的工作模式：16 位自动重装载模式

T9R/T10T9M.3 为 T10T9M 寄存器内的控制位，T10T9M 寄存器各位的具体功能描述见上节 T10T9M 寄存器的介绍。

当 T9_C/T=0 时，多路开关连接到系统时钟输出，T9 对内部系统时钟计数，T9 工作在定时方式。当 T9_C/T=1 时，多路开关连接到外部脉冲输 T9，即 T9 工作在计数方式。

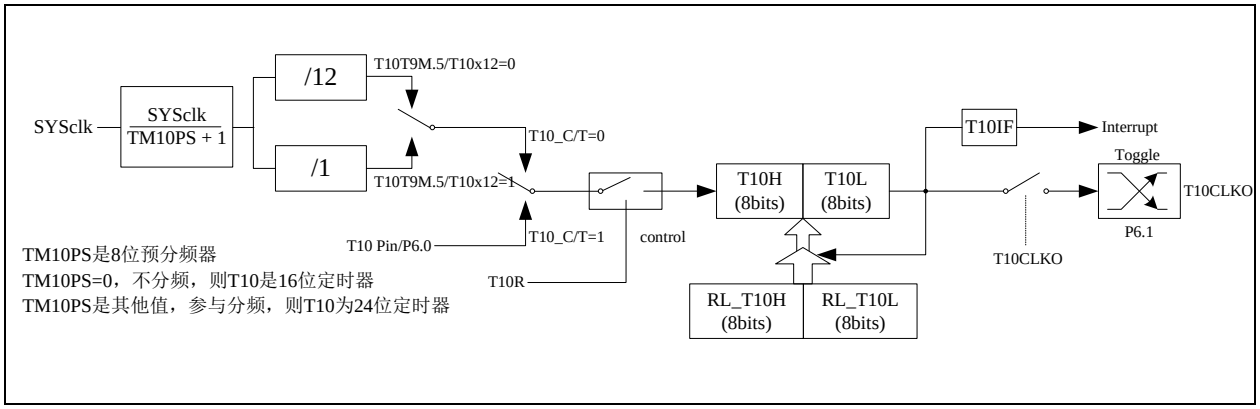
单片机的定时器 9 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T9 的速率由特殊功能寄存器 T10T9M 中的 T9x12 决定，如果 T9x12=0，T9 则工作在 12T 模式；如果 T9x12=1，T9 则工作在 1T 模式

定时器 9 有两个隐藏的寄存器 RL_T9H 和 RL_T9L。RL_T9H 与 T9H 共有同一个地址，RL_T9L 与 T9L 共有同一个地址。当 T9R=0 即定时器/计数器 3 被禁止工作时，对 T9L 写入的内容会同时写入 RL_T9L，对 T9H 写入的内容也会同时写入 RL_T9H。当 T9R=1 即定时器/计数器 3 被允许工作时，对 T9L 写入内容，实际上不是写入当前寄存器 T9L 中，而是写入隐藏的寄存器 RL_T9L 中，对 T9H 写入内容，实际上也不是写入当前寄存器 T9H 中，而是写入隐藏的寄存器 RL_T9H，这样可以巧妙地实现 16 位重装载定时器。当读 T9H 和 T9L 的内容时，所读的内容就是 T9H 和 T9L 的内容，而不是 RL_T9H 和 RL_T9L 的内容。

[T9H,T9L]的溢出不仅置位中断请求标志位（T9IF），使 CPU 转去执行定时器 9 的中断程序，而且会自动将[RL_T9H,RL_T9L]的内容重新装入[T9H,T9L]。

17.6.10 定时器 10 工作模式

定时器/计数器 10 的原理框图如下：



定时器/计数器 10 的工作模式：16 位自动重装载模式

T10R/T10T9M.7 为 T10T9M 寄存器内的控制位，T10T9M 寄存器各位的具体功能描述见上节 T10T9M 寄存器的介绍。

当 T10_C/T=0 时，多路开关连接到系统时钟输出，T10 对内部系统时钟计数，T10 工作在定时方式。当 T10_C/T=1 时，多路开关连接到外部脉冲输 T10，即 T10 工作在计数方式。

单片机的定时器 10 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T10 的速率由特殊功能寄存器 T10T9M 中的 T10x12 决定，如果 T10x12=0，T10 则工作在 12T 模式；如果 T10x12=1，T10 则工作在 1T 模式

定时器 10 有两个隐藏的寄存器 RL_T10H 和 RL_T10L。RL_T10H 与 T10H 共有同一个地址，RL_T10L 与 T10L 共有同一个地址。当 T10R=0 即定时器/计数器 4 被禁止工作时，对 T10L 写入的内容会同时写入 RL_T10L，对 T10H 写入的内容也会同时写入 RL_T10H。当 T10R=1 即定时器/计数器 4 被允许工作时，对 T10L 写入内容，实际上不是写入当前寄存器 T10L 中，而是写入隐藏的寄存器 RL_T10L 中，对 T10H 写入内容，实际上也不是写入当前寄存器 T10H 中，而是写入隐藏的寄存器 RL_T10H，这样可以巧妙地实现 16 位重装载定时器。当读 T10H 和 T10L 的内容时，所读的内容就是 T10H 和 T10L 的内容，而不是 RL_T10H 和 RL_T10L 的内容。

[T10H,T10L]的溢出不仅置位中断请求标志位（T10IF），使 CPU 转去执行定时器 10 的中断程序，而且会自动将[RL_T10H,RL_T10L]的内容重新装入[T10H,T10L]。

17.6.11 定时器 9 计算公式

定时器速度	周期计算公式
1T	定时器周期 = $\frac{65536 - [T9H, T9L]}{SYSclk/(TM9PS+1)}$ （自动重载）
12T	定时器周期 = $\frac{65536 - [T9H, T9L]}{SYSclk/(TM9PS+1)} \times 12$ （自动重载）

17.6.12 定时器 10 计算公式

定时器速度	周期计算公式
1T	定时器周期 = $\frac{65536 - [T10H, T10L]}{SYSclk/(TM10PS+1)}$ （自动重载）
12T	定时器周期 = $\frac{65536 - [T10H, T10L]}{SYSclk/(TM10PS+1)} \times 12$ （自动重载）

17.7 定时器 T11（24 位定时器，8 位预分频+16 位自动重装载定时器）

17.7.1 定时器功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TIMERPS	7EFEACH	-	T17T18SEL	**T11SEL[1:0]**		T9T10SEL	T7T8SEL	T5T6SEL	T3T4SEL

T11SEL：T11 功能脚切换

T11SEL[1:0]	T11	T11CLKO
00	P7.6	P7.7
01	P1.4	P1.5
10	PA.6	PA.7
11	P4.4	P4.5

17.7.2 定时器相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
T11CR	定时器 T11 控制寄存器	7EFE78H	T11R	T11_C/T	T11CLKO	T11x12	T11CS[1:0]		ET11I	T11IF	0000,0000
T11PS	定时器 T11 时钟预分频寄存器	7EFE79H									0000,0000
T11H	定时器 T11 高字节	7EFE7AH									0000,0000
T11L	定时器 T11 低字节	7EFE7BH									0000,0000

17.7.3 定时器 T11 控制寄存器（T11CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T11CR	7EFE78H	T11R	T11_C/T	T11CLKO	T11x12	T11CS[1:0]		ET11I	T11IF

T11R：定时器T11的运行控制位

0：定时器 T11 停止计数

1：定时器 T11 开始计数

T11_C/T：控制定时器T11用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T11(P7.6/P1.4/PA.6/P4.4)外部脉冲进行计数）。

T11CLKO：定时器T11时钟输出控制

0：关闭时钟输出

1：使能（P7.7/P1.5/PA.7/P4.5）口的是定时器 T11 时钟输出功能

当定时器 T11 计数发生溢出时，（P7.7/P1.5/PA.7/P4.5）口的电平自动发生翻转。

T11x12：定时器T11速度控制位

0：12T 模式，即 T11 时钟源 12 分频（T11CLK/12）

1：1T 模式，即 T11 时钟源不分频分频（T11CLK /1）

T11CS[1:0]：定时器T11时钟源（T11CLK）选择

T11CS[1:0]	定时器T11时钟源（T11CLK）
00	系统时钟SYSclk
01	内部高速IRC
10	外部32K晶振
11	内部低速IRC

特别说明：若定时器 T11 选择外部 32K 或者内部低速 IRC 作为时钟源，当 MCU 进入主时钟停振/省电模式时，定时器 T11 会继续工作，发送 T11 定时中断时可唤醒主时钟停振/省电模式

ET11I：定时/计数器 T11 的溢出中断允许位。

0：禁止 T11 中断

1：允许 T11 中断

T11IF：T11溢出中断标志。T11被允许计数以后，从初值开始加1计数，当产生溢出时，由硬件置“1”T11IF，向CPU请求中断，一直保持CPU响应该中断时，才由硬件清0（也可由查询软件清0）。

17.7.4 定时器 T11 的 8 位预分频寄存器（T11PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T11PS	7EFE79H								

定时器T11的时钟 ＝ 系统时钟SYSclk ÷ (T11PS + 1)

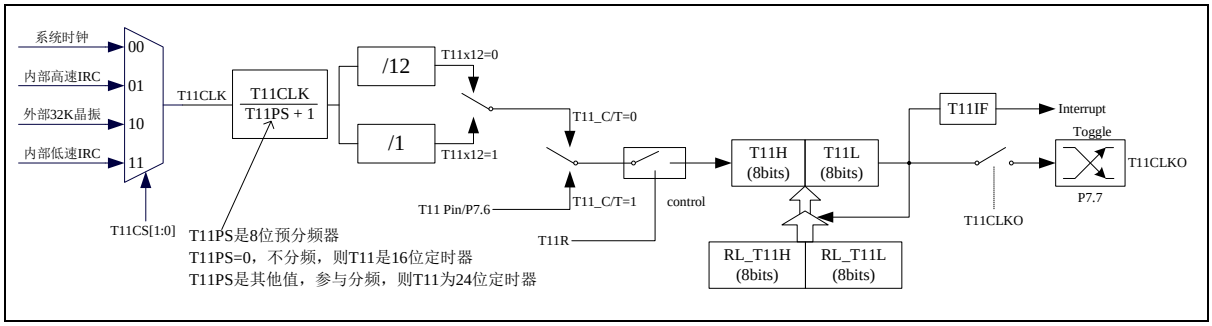
17.7.5 定时器 T11 计数寄存器（T11L，T11H）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T11L	7EFE7BH								
T11H	7EFE7AH								

定时器/计数器T11的工作模式固定为16位重载模式，T11L和T11H组合成为一个16位寄存器，T11L为低字节，T11H为高字节。当[T11H,T11L]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[T11H,T11L]中。

17.7.6 定时器 T11 工作模式

定时器/计数器 T11 的原理框图如下：



定时器/计数器 T11 的工作模式：16 位自动重装载模式

当 T11_C/T=0 时，多路开关连接到内部时钟 T11CLK，T11 对内部 T11CLK 时钟计数，T11 工作在定时方式。当 T11_C/T=1 时，多路开关连接到外部脉冲输 T11（P2.0），即 T11 工作在计数方式。

单片机的定时器 T11 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T11 的速率由特殊功能寄存器 T11x12 决定，如果 T11x12=0，T11 则工作在 12T 模式；如果 T11x12=1，T11 则工作在 1T 模式

定时器 T11 有两个隐藏的寄存器 RL_T11H 和 RL_T11L。RL_T11H 与 T11H 共有同一个地址，RL_T11L 与 T11L 共有同一个地址。当 T11R=0 即定时器/计数器 T11 被禁止工作时，对 T11L 写入的内容会同时写入 RL_T11L，对 T11H 写入的内容也会同时写入 RL_T11H。当 T11R=1 即定时器/计数器 T11 被允许工作时，对 T11L 写入内容，实际上不是写入当前寄存器 T11L 中，而是写入隐藏的寄存器 RL_T11L 中，对 T11H 写入内容，实际上也不是写入当前寄存器 T11H 中，而是写入隐藏的寄存器 RL_T11H，这样可以巧妙地实现 16 位重装载定时器。当读 T11H 和 T11L 的内容时，所读的内容就是 T11H 和 T11L 的内容，而不是 RL_T11H 和 RL_T11L 的内容。

[T11H,T11L]的溢出不仅置位中断请求标志位（T11IF），使 CPU 转去执行定时器 T11 的中断程序，而且会自动将[RL_T11H,RL_T11L]的内容重新装入[T11H,T11L]。

特别的，定时器 T11 的时钟源可通过 T11CS 寄存器进行选择，可选择：系统时钟、内部高速 IRC 时钟、外部 32K 晶振以及内部低速 IRC 时钟

17.7.7 定时器 T11 计算公式

定时器速度	周期计算公式
1T	定时器周期 = $\frac{65536 - [T11H, T11L]}{T11CLK/(T11PS+1)}$ （自动重载）
12T	定时器周期 = $\frac{65536 - [T11H, T11L]}{T11CLK/(T11PS+1)} \times 12$ （自动重载）

17.8 定时器 17/18，24 位定时器，8 位预分频+16 位自动重装载定时器

17.8.1 定时器功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TIMERPS	7EFEACH	-	**T17T18SEL**	T11SEL[1:0]		T9T10SEL	T7T8SEL	T5T6SEL	T3T4SEL

T17T18SEL：T17/T18 功能脚切换

T17T18SEL	T17	T17CLKO	T18	T18CLKO
0	P8.0	P8.1	P9.0	P9.1
1	-	-	-	-

17.8.2 定时器相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
TM17PS	定时器 17 预分频寄存器	7EFBC8H	TM17PS[7:0]								0000,0000
TM18PS	定时器 18 预分频寄存器	7EFBC9H	TM18PS[7:0]								0000,0000
T18T17CFG	定时器 181/7 配置寄存器	7EFBCAH	T18IF	PT18[1:0]		ET18	T17IF	PT17[1:0]		ET17	0000,0000
T18T17M	定时器 18/17 控制寄存器	7EFBCBH	T18R	T18_C/T	T18x12	T18CLKO	T17R	T17_C/T	T17x12	T17CLKO	0000,0000
T18H	定时器 18 高字节	7EFBCCH	TMR18[15:8]								0000,0000
T18L	定时器 18 低字节	7EFBCDH	TMR18[7:0]								0000,0000
T17H	定时器 17 高字节	7EFBCEH	TMR17[15:8]								0000,0000
T17L	定时器 17 低字节	7EFBCFH	TMR17[7:0]								0000,0000

17.8.3 定时器 18/17 控制寄存器（T18T17M）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T18T17M	7EFBCBH	T18R	T18_C/T	T18x12	T18CLKO	T17R	T17_C/T	T17x12	T17CLKO

T18R：定时器18的运行控制位

0：定时器 18 停止计数

1：定时器 18 开始计数

T18_C/T：控制定时器18用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T18/P9.0外部脉冲进行计数）。

T18x12：定时器18速度控制位

0：12T 模式，即 CPU 时钟 12 分频（FOSC/12）

1：1T 模式，即 CPU 时钟不分频分频（FOSC/1）

T18CLKO：定时器18时钟输出控制

0：关闭时钟输出

1：使能 T18CLKO/P9.1 口的是定时器 18 时钟输出功能

当定时器 18 计数发生溢出时，T18CLKO/P9.1 口的电平自动发生翻转。

T17R：定时器17的运行控制位

0：定时器 17 停止计数

1：定时器 17 开始计数

T17_C/T：控制定时器17用作定时器或计数器，清0则用作定时器（对内部系统时钟进行计数），置1用作计数器（对引脚T17/P8.0外部脉冲进行计数）。

T17x12：定时器17速度控制位

0：12T 模式，即 CPU 时钟 12 分频（FOSC/12）

1：1T 模式，即 CPU 时钟不分频分频（FOSC/1）

T17CLKO：定时器17时钟输出控制

0：关闭时钟输出

1：使能 T17CLKO/P8.1 口的是定时器 17 时钟输出功能

当定时器 17 计数发生溢出时，T17CLKO/P8.1 口的电平自动发生翻转。

17.8.4 定时器 18/17 配置寄存器（T18T17CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T18T17CFG	7EFBCAH	T18IF	PT18[1:0]		ET18	T17IF	PT17[1:0]		ET17

ET17：定时/计数器 T17 的溢出中断允许位。

0：禁止 T17 中断

1：允许 T17 中断

T17IF：定时器17溢出中断标志。**需软件写0清零**。

PT17[1:0]：定时器17中断优先级控制位

00：定时器 17 中断优先级为 0 级（最低级）

01：定时器 17 中断优先级为 1 级（较低级）

10：定时器 17 中断优先级为 2 级（较高级）

11：定时器 17 中断优先级为 3 级（最高级）

ET18：定时/计数器 T18 的溢出中断允许位。

0：禁止 T18 中断

1：允许 T18 中断

T18IF：定时器18溢出中断标志。**需软件写0清零**。

PT18[1:0]：定时器18中断优先级控制位

00：定时器 18 中断优先级为 0 级（最低级）

01：定时器 18 中断优先级为 1 级（较低级）

10：定时器 18 中断优先级为 2 级（较高级）

11：定时器 18 中断优先级为 3 级（最高级）

17.8.5 定时器 17 计数寄存器（T17L，T17H）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T17L	7EFBCFH								
T17H	7EFBCEH								

定时器/计数器17的工作模式固定为16位重载模式，T17L和T17H组合成为一个16位寄存器，T17L为低字节，T17H为高字节。当[T17H,T17L]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[T17H,T17L]中。

17.8.6 定时器 18 计数寄存器（T18L，T18H）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
T18L	7EFBBDH								
T18H	7EFBBCH								

定时器/计数器18的工作模式固定为16位重载模式，T18L和T18H组合成为一个16位寄存器，T18L为低字节，T18H为高字节。当[T18H,T18L]中的16位计数值溢出时，系统会自动将内部16位重载寄存器中的重载值装入[T18H,T18L]中。

17.8.7 定时器 17 的 8 位预分频寄存器（TM7PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TM17PS	7EFBC8H								

定时器17的时钟 ＝ 系统时钟SYSclk ÷ (TM17PS + 1)

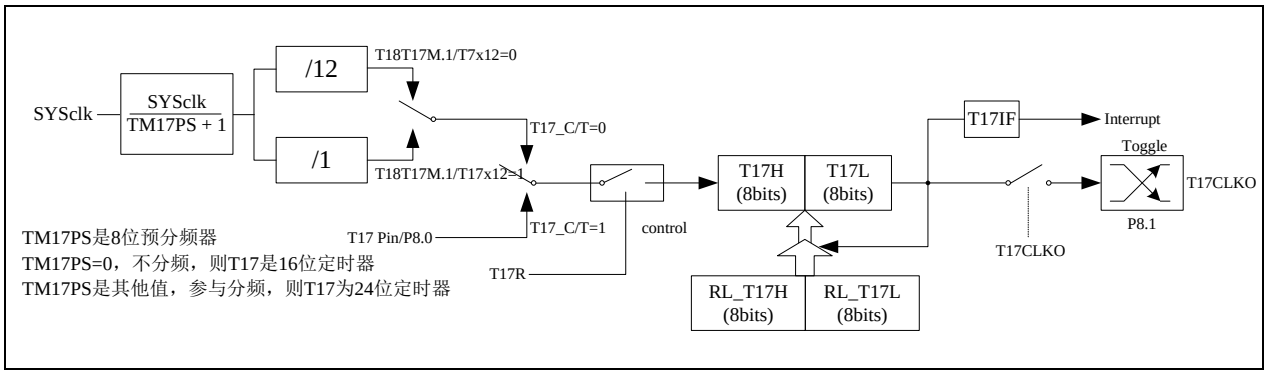
17.8.8 定时器 18 的 8 位预分频寄存器（TM8PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
TM18PS	7EFBC9H								

定时器18的时钟 ＝ 系统时钟SYSclk ÷ (TM18PS + 1)

17.8.9 定时器 17 工作模式

定时器/计数器17的原理框图如下：



定时器/计数器 17 的工作模式：16 位自动重装载模式

T17R/T18T17M.3 为 T18T17M 寄存器内的控制位，T18T17M 寄存器各位的具体功能描述见上节 T18T17M 寄存器的介绍。

当 T17_C/T=0 时，多路开关连接到系统时钟输出，T17 对内部系统时钟计数，T17 工作在定时方式。当 T17_C/T=1 时，多路开关连接到外部脉冲输 T17，即 T17 工作在计数方式。

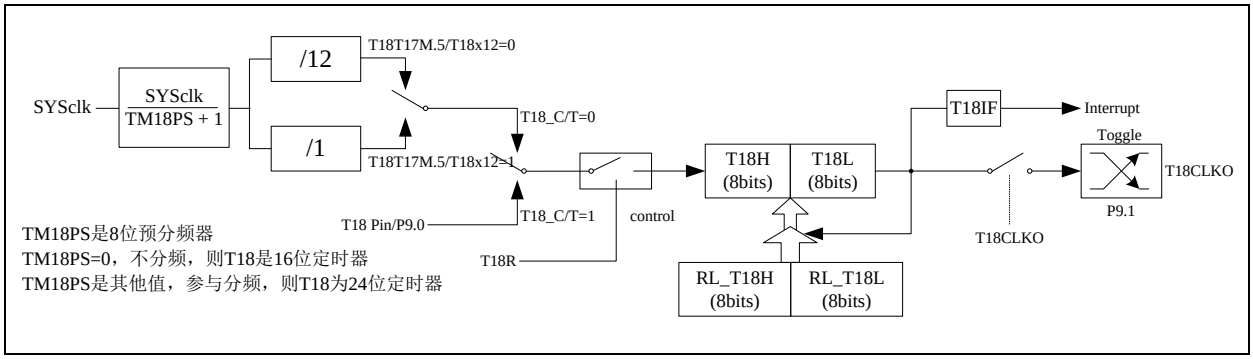
单片机的定时器 17 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T17 的速率由特殊功能寄存器 T18T17M 中的 T17x12 决定，如果 T17x12=0，T17 则工作在 12T 模式；如果 T17x12=1，T17 则工作在 1T 模式

定时器 17 有两个隐藏的寄存器 RL_T17H 和 RL_T17L。RL_T17H 与 T17H 共有同一个地址，RL_T17L 与 T17L 共有同一个地址。当 T17R=0 即定时器/计数器 3 被禁止工作时，对 T17L 写入的内容会同时写入 RL_T17L，对 T17H 写入的内容也会同时写入 RL_T17H。当 T17R=1 即定时器/计数器 3 被允许工作时，对 T17L 写入内容，实际上不是写入当前寄存器 T17L 中，而是写入隐藏的寄存器 RL_T17L 中，对 T17H 写入内容，实际上也不是写入当前寄存器 T17H 中，而是写入隐藏的寄存器 RL_T17H，这样可以巧妙地实现 16 位重装载定时器。当读 T17H 和 T17L 的内容时，所读的内容就是 T17H 和 T17L 的内容，而不是 RL_T17H 和 RL_T17L 的内容。

[T17H,T17L]的溢出不仅置位中断请求标志位（T17IF），使 CPU 转去执行定时器 17 的中断程序，而且会自动将[RL_T17H,RL_T17L]的内容重新装入[T17H,T17L]。

17.8.10 定时器 18 工作模式

定时器/计数器 18 的原理框图如下：



定时器/计数器 18 的工作模式：16 位自动重装载模式

T18R/T18T17M.7 为 T18T17M 寄存器内的控制位，T18T17M 寄存器各位的具体功能描述见上节 T18T17M 寄存器的介绍。

当 T18_C/T=0 时，多路开关连接到系统时钟输出，T18 对内部系统时钟计数，T18 工作在定时方式。当 T18_C/T=1 时，多路开关连接到外部脉冲输 T18，即 T18 工作在计数方式。

单片机的定时器 18 有两种计数速率：一种是 12T 模式，每 12 个时钟加 1，与传统 8051 单片机相同；另外一种是 1T 模式，每个时钟加 1，速度是传统 8051 单片机的 12 倍。T18 的速率由特殊功能寄存器 T18T17M 中的 T18x12 决定，如果 T18x12=0，T18 则工作在 12T 模式；如果 T18x12=1，T18 则工作在 1T 模式

定时器 18 有两个隐藏的寄存器 RL_T18H 和 RL_T18L。RL_T18H 与 T18H 共有同一个地址，RL_T18L 与 T18L 共有同一个地址。当 T18R=0 即定时器/计数器 4 被禁止工作时，对 T18L 写入的内容会同时写入 RL_T18L，对 T18H 写入的内容也会同时写入 RL_T18H。当 T18R=1 即定时器/计数器 4 被允许工作时，对 T18L 写入内容，实际上不是写入当前寄存器 T18L 中，而是写入隐藏的寄存器 RL_T18L 中，对 T18H 写入内容，实际上也不是写入当前寄存器 T18H 中，而是写入隐藏的寄存器 RL_T18H，这样可以巧妙地实现 16 位重装载定时器。当读 T18H 和 T18L 的内容时，所读的内容就是 T18H 和 T18L 的内容，而不是 RL_T18H 和 RL_T18L 的内容。

[T18H,T18L]的溢出不仅置位中断请求标志位（T18IF），使 CPU 转去执行定时器 18 的中断程序，而且会自动将[RL_T18H,RL_T18L]的内容重新装入[T18H,T18L]。

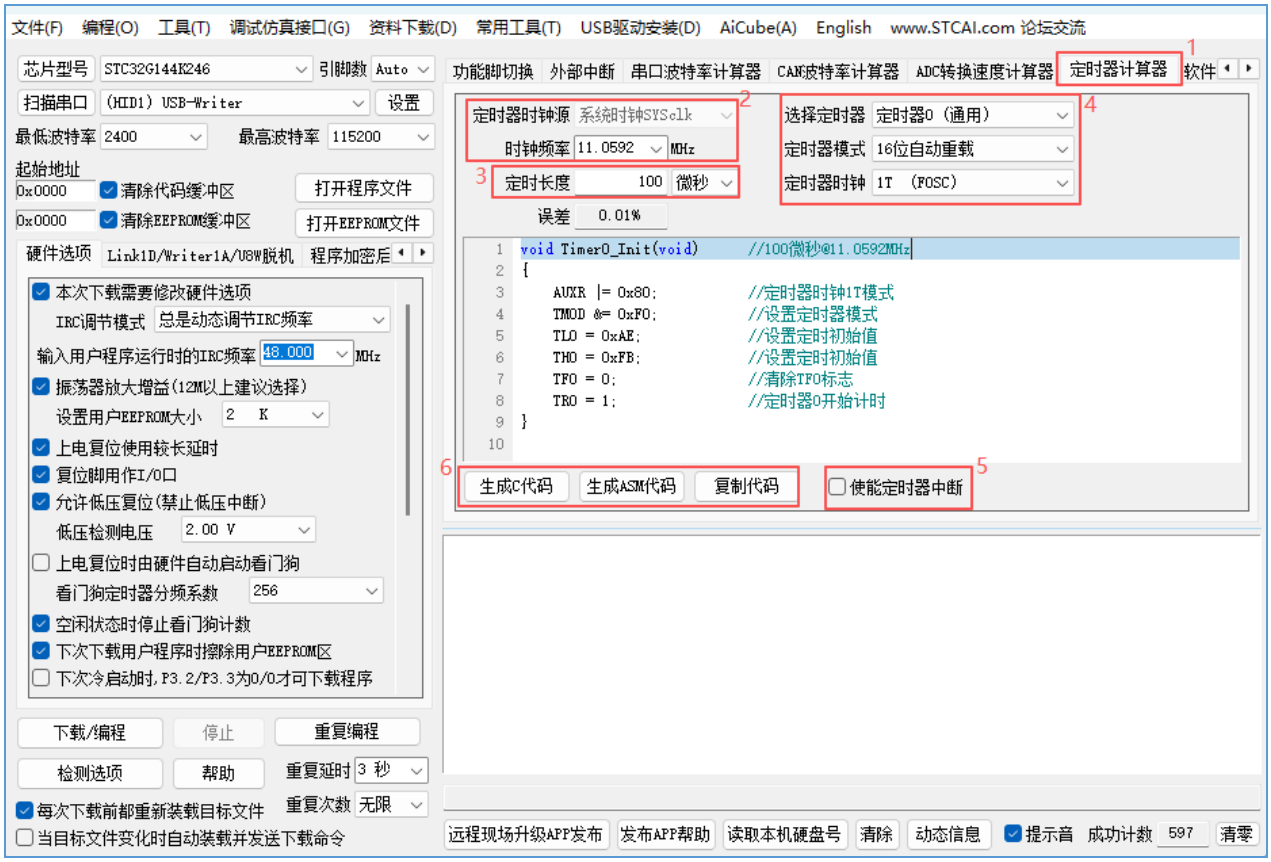
17.8.11 定时器 17 计算公式

定时器速度	周期计算公式
1T	定时器周期 = $\frac{65536 - [T17H, T17L]}{SYSclk/(TM17PS+1)}$ （自动重载）
12T	定时器周期 = $\frac{65536 - [T17H, T17L]}{SYSclk/(TM17PS+1)} \times 12$ （自动重载）

17.8.12 定时器 18 计算公式

定时器速度	周期计算公式
1T	定时器周期 = $\frac{65536 - [T18H, T18L]}{SYSclk/(TM18PS+1)}$ （自动重载）
12T	定时器周期 = $\frac{65536 - [T18H, T18L]}{SYSclk/(TM18PS+1)} \times 12$ （自动重载）

17.9 AiCube-ISP | 定时器计算器工具



①：在下载软件中选择“定时器计算器”功能页，进定时器代码生成界面

②：设置系统工作频率（单位：MHz）

③：设置定时时间长度（单位：毫秒/微秒）

④：选择目标定时器，并设置定时器工作模式

⑤：选择是否需要使能定时器中断

⑥：手动生成 C 代码或者 ASM 代码，复制范例

18 RTC 实时时钟，年/月/日/星期/时/分/秒

产品线	RTC	支持星期	芯片复位时 RTC 寄存器不复位[1]
STC32G144K246 系列	●	●	●

[1]：包括硬件复位和软件复位

STC32G144K246 系列部分单片机内部集成一个实时时钟控制电路，主要有如下特性：

- 低功耗：RTC 模块工作电流低至 **2uA@VCC=3.3V、3uA@VCC=5.0V（典型值）**
- 长时间跨度：支持 2000 年~2099 年，并自动判断闰年
- 根据输入的年/月/日自动计算星期值
- 闹钟：支持一组闹钟设置
- 支持多个中断：
  - 一组闹钟中断（每天中断一次，中断的时间点为闹钟寄存器所设置的任意时/分/秒）
  - 日中断（每天中断一次，中断的时间点为每天的 0 时 0 分 0 秒）
  - 小时中断（每小时中断一次，中断的时间点为分/秒均为 0，即整点时）
  - 分钟中断（每分钟中断一次，中断的时间点为秒为 0，即分钟寄存器发生变化时）
  - 秒中断（每秒中断一次，中断的时间点为秒寄存器发生变化时）
  - 1/2 秒中断（每 1/2 秒中断一次）
  - 1/8 秒中断（每 1/8 秒中断一次）
  - 1/32 秒中断（每 1/32 秒中断一次）
- 支持掉电唤醒

18.1 RTC 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
RTCCR	RTC 控制寄存器	7EFE60H	-	-	-	-	-	-	-	RUNRTC	xxxx,xxx0
RTCCFG	RTC 配置寄存器	7EFE61H	-	-	-	-	-	-	RTCCKS	SETRTC	xxxx,xx00
RTCIEN	RTC 中断使能寄存器	7EFE62H	EALAI	EDAYI	EHOURI	EMINI	ESECI	ESEC2I	ESEC8I	ESEC32I	0000,0000
RTCIF	RTC 中断请求寄存器	7EFE63H	ALAIF	DAYIF	HOURIF	MINIF	SECIF	SEC2IF	SEC8IF	SEC32IF	0000,0000
ALAHOUR	RTC 闹钟的小时值	7EFE64H	-	-	-	ALAHOUR[4:0]					xxx0,0000
ALAMIN	RTC 闹钟的分钟值	7EFE65H	-	-	ALAMIN[5:0]						xx00,0000
ALASEC	RTC 闹钟的秒值	7EFE66H	-	-	ALASEC[5:0]						xx00,0000
ALASSEC	RTC 闹钟的 1/128 秒值	7EFE67H	-	ALASSEC[6:0]							x000,0000
INIYEAR	RTC 年初始化	7EFE68H	-	INIYEAR[6:0]							x000,0000
INIMONTH	RTC 月初始化	7EFE69H	-	-	-	-	INIMONTH[3:0]				xxxx,0000
INIDAY	RTC 日初始化	7EFE6AH	-	-	-	INIDAY[4:0]					xxx0,0000
INIHOUR	RTC 小时初始化	7EFE6BH	-	-	-	INIHOUR[4:0]					xxx0,0000
INIMIN	RTC 分钟初始化	7EFE6CH	-	-	INIMIN[5:0]						xx00,0000
INISEC	RTC 秒初始化	7EFE6DH	-	-	INISEC[5:0]						xx00,0000
INISSEC	RTC1/128 秒初始化	7EFE6EH	-	INISSEC[6:0]							x000,0000
RTCWEEK	RTC 的星期值	7EFE6FH	-	-	-	-	-	RTCWEEK[2:0]			xxxx,xnnn
RTCYEAR	RTC 的年计数值	7EFE70H	-	RTCYEAR[6:0]							x000,0000
RTCMONTH	RTC 的月计数值	7EFE71H	-	-	-	-	RTCMONTH[3:0]				xxxx,0000
RTCDAY	RTC 的日计数值	7EFE72H	-	-	-	RTCDAY[4:0]					xxx0,0000
RTCHOUR	RTC 的小时计数值	7EFE73H	-	-	-	RTCHOUR[4:0]					xxx0,0000
RTCMIN	RTC 的分钟计数值	7EFE74H	-	-	RTCMIN[5:0]						xx00,0000
RTCSEC	RTC 的秒计数值	7EFE75H	-	-	RTCSEC[5:0]						xx00,0000
RTCSSEC	RTC 的 1/128 秒计数值	7EFE76H	-	RTCSSEC[6:0]							x000,0000

18.1.1 RTC 控制寄存器（RTCCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
RTCCR	7EFE60H	-	-	-	-	-	-	-	RUNRTC

RUNRTC：RTC 模块控制位

0：关闭 RTC，RTC 停止计数

1：使能 RTC，并开始 RTC 计数

18.1.2 RTC 配置寄存器（RTCCFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
RTCCFG	7EFE61H	-	-	-	-	-	-	RTCCKS	SETRTC

RTCCKS：RTC 时钟源选择

0：选择外部 32.768KHz 时钟源（需先软件启动外部 32K 晶振）

1：选择内部 32K 时钟源（需先软件启动内部 32K 振荡器）

SETRTC：设置 RTC 初始值

写 0：无意义

写 1：触发 RTC 寄存器初始化。当 SETRTC 设置为 1 时，硬件会自动将寄存器 INIYEAR、INIMONTH、INIDAY、INIWEEK、INIHOUR、INIMIN、INISEC、INISSEC 中的值复制到寄存器 RTCYEAR、RTCMONTH、RTCDAY、RTCWEEK、RTCHOUR、RTCMIN、RTCSEC、RTCSSEC 中。初始完成后，硬件会自动将 SETRTC 位清 0。

读 0：设置 RTC 相关时间寄存器已完成

读 1：硬件正在进行设置 RTC，还未完成

注：等待初始化完成，需要在"RTC 使能"之后判断。设置 RTC 时间需要 32768Hz 的 1 个周期时间，大约 30.5us。由于同步，所以实际等待时间是 0~30.5us，如果不等待设置完成就睡眠，则 RTC 会由于设置没完成，停止计数，唤醒后才继续完成设置并继续计数，如果此时设置的是使用 RTC 中断进行唤醒，则会出现无法唤醒 MCU 的情况。

18.1.3 RTC 中断使能寄存器（RTCIEN）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
RTCIEN	7EFE62H	EALAI	EDAYI	EHOURI	EMINI	ESECI	ESEC2I	ESEC8I	ESEC32I

EALAI：闹钟中断使能位

0：关闭闹钟中断

1：使能闹钟中断

EDAYI：一日（24 小时）中断使能位

0：关闭一日中断

1：使能一日中断

EHOURI：一小时（60 分钟）中断使能位

0：关闭小时中断

1：使能小时中断

EMINI：一分钟（60 秒）中断使能位

0：关闭分钟中断

1：使能分钟中断

ESECI：一秒中断使能位

0：关闭秒中断

1：使能秒中断

ESEC2I：1/2 秒中断使能位

0：关闭 1/2 秒中断

1：使能 1/2 秒中断

ESEC8I：1/8 秒中断使能位

0：关闭 1/8 秒中断

1：使能 1/8 秒中断

ESEC32I：1/32 秒中断使能位

0：关闭 1/32 秒中断

1：使能 1/32 秒中断

18.1.4 RTC 中断请求寄存器（RTCIF）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
RTCIF	7EFE63H	ALAIF	DAYIF	HOURIF	MINIF	SECIF	SEC2IF	SEC8IF	SEC32IF

ALAIF：闹钟中断请求位。需软件写 0 清 0。
DAYIF：一日（24 小时）中断请求位。需软件写 0 清 0。
HOURIF：一小时（60 分钟）中断请求位。需软件写 0 清 0。
MINIF：一分钟（60 秒）中断请求位。需软件写 0 清 0。
SECIF：一秒中断请求位。需软件写 0 清 0。
SEC2IF：1/2 秒中断请求位。需软件写 0 清 0。
SEC8IF：1/8 秒中断请求位。需软件写 0 清 0。
SEC32IF：1/32 秒中断请求位。需软件写 0 清 0。

18.1.5 RTC 闹钟设置寄存器

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ALAHOUR	7EFE64H	-	-	-					
ALAMIN	7EFE65H	-	-						
ALASEC	7EFE66H	-	-						
ALASSEC	7EFE67H	-							

ALAHOUR：设置每天闹钟的小时值。

注意：设置的值不是 BCD 码，而是 HEX 码，比如需要设置小时值 20 到 ALAHOUR，则需使用如下代码进行设置

```
MOV     WR6,#WORD0 ALAHOUR
MOV     WR4,#WORD2 ALAHOUR
MOV     A,#14H
MOV     @DR4,R11
```

ALAMIN：设置每天闹钟的分钟值。数字编码与 ALAHOUR 相同。
ALASEC：设置每天闹钟的秒值。数字编码与 ALAHOUR 相同。
ALASSEC：设置每天闹钟的 1/128 秒值。数字编码与 ALAHOUR 相同。

18.1.6 RTC 实时时钟初始值设置寄存器

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INIYEAR	7EFE68H	-							
INIMONTH	7EFE69H								
INIDAY	7EFE6AH								
INIHOUR	7EFE6BH	-	-	-					
INIMIN	7EFE6CH	-	-						
INISEC	7EFE6DH	-	-						
INISSEC	7EFE6EH	-							

INIYEAR：设置当前实时时间的年值。有效值范围 00~99。对应 2000 年~2099 年

注意：设置的值不是 BCD 码，而是 HEX 码，比如需要设置 20 到 INIYEAR，则需使用如下代码进行设置

```
MOV     WR6,#WORD0 INIYEAR
MOV     WR4,#WORD2 INIYEAR
MOV     A,#14H
MOV     @DR4,R11
```

INIMONTH：设置当前实时时间的月值。有效值范围 1~12。数字编码与 INIYEAR 相同。

INIDAY：设置当前实时时间的日值。有效值范围 1~31。数字编码与 INIYEAR 相同。

INIHOUR：设置当前实时时间的小时值。有效值范围 00~23。数字编码与 INIYEAR 相同。

INIMIN：设置当前实时时间的分钟值。有效值范围 00~59。数字编码与 INIYEAR 相同。

INISEC：设置当前实时时间的秒值。有效值范围 00~59。数字编码与 INIYEAR 相同。

INISSEC：设置当前实时时间的 1/128 秒值。有效值范围 00~127。数字编码与 INIYEAR 相同。

当用户设置完成上面的初始值寄存器后，用户还需要向 SETRTC 位（RTCCFG.0）写 1 来触发硬件将初始值装载到 RTC 实时计数器中

硬件会对初始化数据的有效性进行检查。

18.1.7 RTC 实时时钟计数寄存器

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
RTCWEEK	7EFE6FH	-	-	-	-	-	RTCWEEK[2:0]		
RTCYEAR	7EFE70H	-	RTCYEAR[6:0]						
RTCMONTH	7EFE71H	-	-	-	-	RTCMONTH[3:0]			
RTCDAY	7EFE72H	-	-	-	RTCDAY[4:0]				
RTCHOUR	7EFE73H	-	-	-	RTCHOUR[4:0]				
RTCMIN	7EFE74H	-	-	RTCMIN[5:0]					
RTCSEC	7EFE75H	-	-	RTCSEC[5:0]					
RTCSSEC	7EFE76H	-	RTCSSEC[6:0]						

RTCWEEK：当前实时时间的星期值。**硬件会根据当前的年月日日期值自动计算星期。**

RTCYEAR：当前实时时间的年值。

RTCMONTH：当前实时时间的月值。数字编码与 MONTH 相同。

RTCDAY：当前实时时间的日值。数字编码与 DAY 相同。

RTCHOUR：当前实时时间的小时值。数字编码与 HOUR 相同。

RTCMIN：当前实时时间的分钟值。数字编码与 MIN 相同。

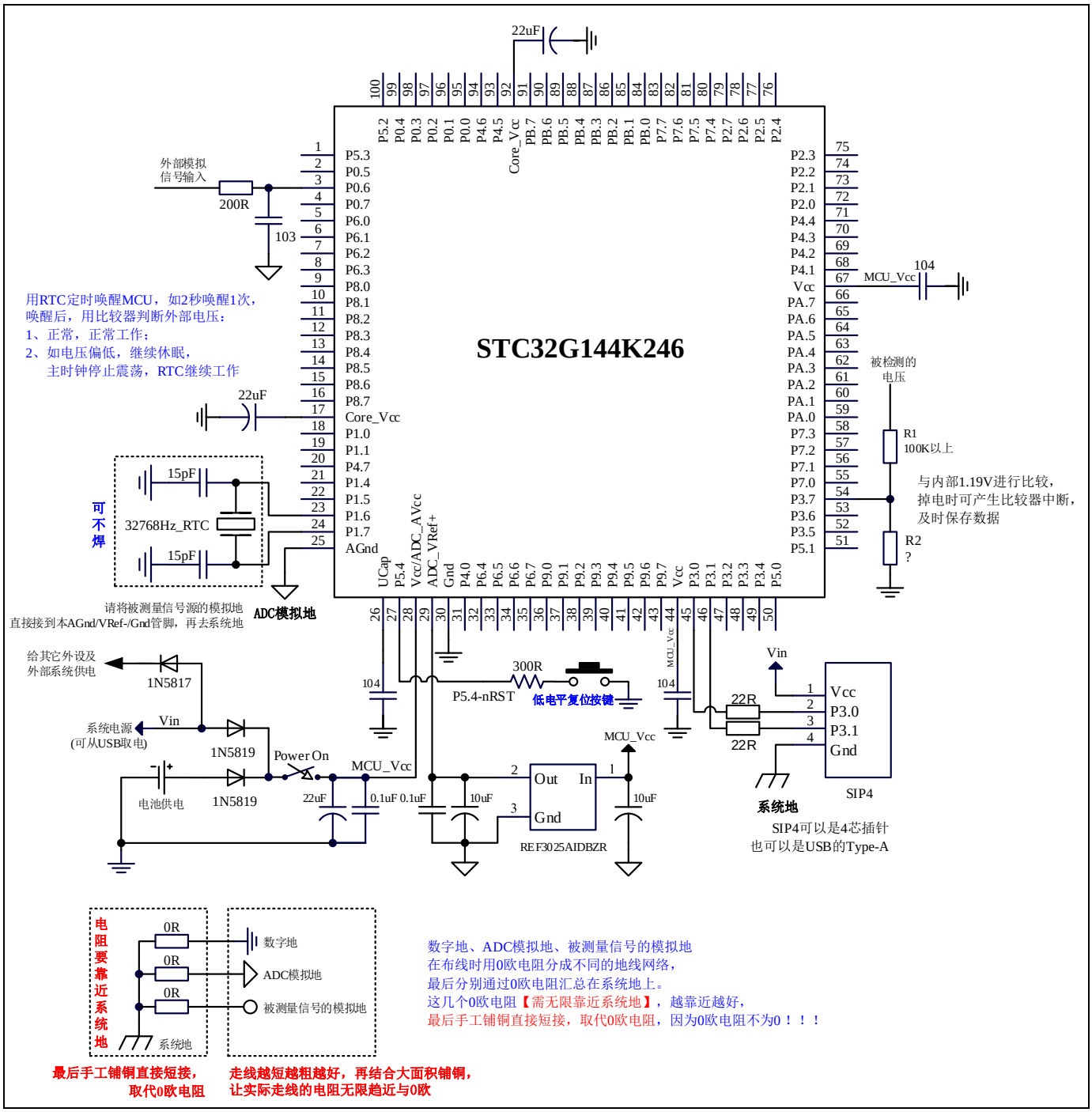
RTCSEC：当前实时时间的秒值。数字编码与 SEC 相同。

RTCSSEC：当前实时时间的 1/128 秒值。数字编码与 SSEC 相同。

注意：

- RTCYEAR、RTCMONTH、RTCDAY、RTCHOUR、RTCMIN、RTCSEC 和 RTCSSEC 均为只读寄存器，若需要对这些寄存器执行写操作，必须通过寄存器 INIYEAR、INIMONTlH、INIDAT、INIHOU、INIMIN、INISEC、INISSEC 和 SETRTC 来实现。
- RTCWEEK 也是只读寄存器，**硬件会根据当前的年月日日期值自动计算星期，无需用户设置。**

18.2 RTC 实战线路图



19 8 组 USART-LIN/SPI，同步/异步串口通信，硬件奇偶校验

产品线	同步/异步串口数量
STC32G144K246 系列	**8 组**

STC32G144K246 系列单片机具有 8 组 USART-LIN/SPI，全双工同步/异步串行通信接口（USART1～USART8），支持 LIN、SPI，可省电模式唤醒。每个串口由 2 个数据缓冲器、一个移位寄存器、一个串行控制寄存器和一个波特率发生器等组成。每个串口的数据缓冲器由 2 个互相独立的接收、发送缓冲器构成，可以同时发送和接收数据。

STC32G144K246 系列单片机的 8 组串口均有 4 种工作方式，其中两种方式的波特率是可变的，另两种是固定的，以供不同应用场合选用。用户可用软件设置不同的波特率和选择不同的工作方式。主机可通过查询或中断方式对接收/发送进行程序处理，使用十分灵活。

串口通讯口均可以通过功能管脚的切换功能切换到多组端口，从而可以将一个通讯口分时复用为多个通讯口。

19.1 利用串口 DMA 的发送时间间隔寄存器可实现 1.5bit/2bit 停止位

STC32G144K246 系列单片机的 USART 串口硬件只支持 1 位停止位，若实际应用中不需要校验位，则可将第 9 位设置 1，并将串口设置为 9 位数据，则可将实现两位停止位。

如果实际应用中需要使用校验位，则可通过设置串口 DMA 的发送时间间隔寄存器来实现 1.5 位或者 2 位停止位。具体实现方法：

- 将寄存器 DMA_UR1_ITVH 和 DMA_UR1_ITVL 中的值设为 0.5 位串口数据位时间，则可实现 1.5 位停止位；
- 将寄存器 DMA_UR1_ITVH 和 DMA_UR1_ITVL 中的值设为 1 位串口数据位时间，则可实现 2 位停止位。

【特别注意】：使用此方法必须使用 DMA 方式发送串口数据才有效。

19.2 串口 1（同步/异步串口 USART）

19.2.1 串口 1 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW1	A2H	**S1_S[1:0]**		-	-	SPI_S[1:0]		-	-

S1_S[1:0]：串口 1 功能脚选择位

S1_S[1:0]	RxD	TxD
00	P3.0	P3.1
01	P3.6	P3.7
10	P1.6	P1.7
11	P4.3	P4.4

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW3	BBH	I2S_S		S2SPI_S[1:0]		**S1SPI_S[1:0]**		-	-

S1SPI_S[1:0]：USART1 的 SPI 功能脚选择位

S1SPI_S[1:0]	S1SS	S1MOSI	S1MISO	S1SCLK
00	P1.2	P1.3	P1.4	P1.5
01	P2.2	P2.3	P2.4	P2.5
10	P5.4	P4.0	P4.1	P4.3
11	P6.4	P6.5	P6.6	P6.7

19.2.2 串口 1 相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
SCON	串口 1 控制寄存器	98H	SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI	0000,0000
SBUF	串口 1 数据寄存器	99H									0000,0000
PCON	电源控制寄存器	87H	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL	0011,0000
AUXR	辅助寄存器 1	8EH	T0x12	T1x12	UART_M0x6	T2R	T2_C/T	T2x12	EXTRAM	S1BRT	0000,0001
SADDR	串口 1 从机地址寄存器	A9H									0000,0000
SADEN	串口 1 从机地址屏蔽寄存器	B9H									0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
USARTCR1	串口 1 控制寄存器 1	7EFDC0H	S1LINEN	S1DORD	S1CLKEN	S1SPMOD	S1SPIEN	S1SPSLV	S1CPOL	S1CPHA	0000,0000
USARTCR2	串口 1 控制寄存器 2	7EFDC1H	S1IREN	S1IRLP	S1SCEN	S1NACK	S1HDSEL	S1PCEN	S1PS	S1PE	0000,0000
USARTCR3	串口 1 控制寄存器 3	7EFDC2H	S1IrDA_LPBAUD[7:0]								0000,0111
USARTCR4	串口 1 控制寄存器 4	7EFDC3H	-	-	-	-	S1SCCKS[1:0]		S1SPICKS[1:0]		xxxx,0000
USARTCR5	串口 1 控制寄存器 5	7EFDC4H	S1BRKDET	S1HDRER	S1SLVEN	S1ASYNC	S1TXCF	S1SENDBK	S1HDRDET	S1SYNC	0000,0000
USARTGTR	串口 1 保护时间寄存器	7EFDC5H	USARTGTR[7:0]								0000,0000
USARTBRH	串口 1 波特率寄存器	7EFDC6H	USARTBR[15:8]								0000,0000
USARTBRL	串口 1 波特率寄存器	7EFDC7H	USARTBR[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
UR1TOCR	串口 1 接收超时控制寄存器	7EFD70H	UR1ENTO	UR1ENTOI	UR1SCALE	-	-	-	-	-	000x,xxxx
UR1TOSR	串口 1 接收超时状态寄存器	7EFD71H	-	-	-	-	-	-	-	UR1TOIF	xxxx,xxx0
UR1TOTH	串口 1 接收超时长度寄存器	7EFD72H	UR1TM[15:8]								0000,0000
UR1TOTL	串口 1 接收超时长度寄存器	7EFD73H	UR1TM[7:0]								0000,0000
UR1TOTE	串口 1 接收超时长度寄存器	7EFD88H	UR1TM[23:16]								0000,0000

19.2.3 串口 1 控制寄存器（SCON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SCON	98H	SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI

SM0/FE：当PCON寄存器中的SMOD0位为1时，该位为帧错误检测标志位。当串口在接收过程中检测到一个无效停止位时，通过串口接收器将该位置1，必须由软件清零。当PCON寄存器中的SMOD0位为0时，该位和SM1一起指定串口1的通信工作模式，如下表所示：

SM0	SM1	串口1工作模式	功能说明
0	0	模式0	同步移位串行方式
0	1	模式1	可变波特率8位数据方式
1	0	模式2	固定波特率9位数据方式
1	1	模式3	可变波特率9位数据方式

当使能 FE 功能后，串口的工作模式只能使用 SM1 进行设置，且只能设置为模式 0 或者模式 1。FE 当作帧错误标志位时，此标志位必须软件写 0 清零。

SM2：允许模式 2 或模式 3 多机通信控制位。当串口 1 使用模式 2 或模式 3 时，如果 SM2 位为 1 且 REN 位为 1，则接收机处于地址帧筛选状态。此时可以利用接收到的第 9 位（即 RB8）来筛选地址帧，若 RB8=1，说明该帧是地址帧，地址信息可以进入 SBUF，并使 RI 为 1，进而在中断服务程序中再进行地址号比较；若 RB8=0，说明该帧不是地址帧，应丢掉且保持 RI=0。在模式 2 或模式 3 中，如果 SM2 位为 0 且 REN 位为 1，接收收机处于地址帧筛选被禁止状态，不论收到的 RB8 为 0 或 1，均可使接收到的信息进入 SBUF，并使 RI=1，此时 RB8 通常为校验位。模式 1 和模式 0 为非多机通信方式，在这两种方式时，SM2 应设置为 0。

REN：允许/禁止串口接收控制位

0：禁止串口接收数据

1：允许串口接收数据

TB8：当串口 1 使用模式 2 或模式 3 时，TB8 为要发送的第 9 位数据，按需要由软件置位或清 0。在模式 0 和模式 1 中，该位不用。

RB8：当串口 1 使用模式 2 或模式 3 时，RB8 为接收到的第 9 位数据，一般用作校验位或者地址帧/数据帧标志位。在模式 0 和模式 1 中，该位不用。

TI：串口 1 发送中断请求标志位。在模式 0 中，当串口发送数据第 8 位结束时，由硬件自动将 TI 置 1，向主机请求中断，响应中断后 TI 必须用软件清零。在其他模式中，则在停止位开始发送时由硬件自动将 TI 置 1，向 CPU 发请求中断，响应中断后 TI 必须用软件清零。

RI：串口 1 接收中断请求标志位。在模式 0 中，当串口接收第 8 位数据结束时，由硬件自动将 RI 置 1，向主机请求中断，响应中断后 RI 必须用软件清零。在其他模式中，串行接收到停止位的中间时刻由硬件自动将 RI 置 1，向 CPU 发中断申请，响应中断后 RI 必须由软件清零。

19.2.4 串口 1 数据寄存器（SBUF）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SBUF	99H								

SBUF：串口 1 数据接收/发送缓冲区。SBUF 实际是 2 个缓冲器，读缓冲器和写缓冲器，两个操作分别对应两个不同的寄存器，1 个是只写寄存器（写缓冲器），1 个是只读寄存器（读缓冲器）。对 SBUF 进行读操作，实际是读取串口接收缓冲区，对 SBUF 进行写操作则是触发串口开始发送数据。

19.2.5 电源管理寄存器（PCON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PCON	87H	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL

SMOD：串口 1 波特率控制位

0：串口 1 的各个模式的波特率都不加倍

1：串口 1 模式 1（定时器 1 的模式 2 作为波特率发生器时有效）、模式 2、模式 3（定时器 1 的模式 2 作为波特率发生器时有效）的波特率加倍

SMOD0：帧错误检测控制位

0：无帧错检测功能

1：使能帧错误检测功能。此时 SCON 的 SM0/FE 为 FE 功能，即为帧错误检测标志位。

19.2.6 辅助寄存器 1（AUXR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
AUXR	8EH	T0x12	T1x12	UART_M0x6	T2R	T2_C/T	T2x12	EXTRAM	S1BRT

UART_M0x6：串口 1 模式 0 的通讯速度控制

0：串口 1 模式 0 的波特率不加倍，固定为 Fosc/12

1：串口 1 模式 0 的波特率 6 倍速，即固定为 Fosc/12*6 ＝ Fosc/2

S1BRT：串口 1 波特率发生器选择位

0：选择定时器 1 作为波特率发生器

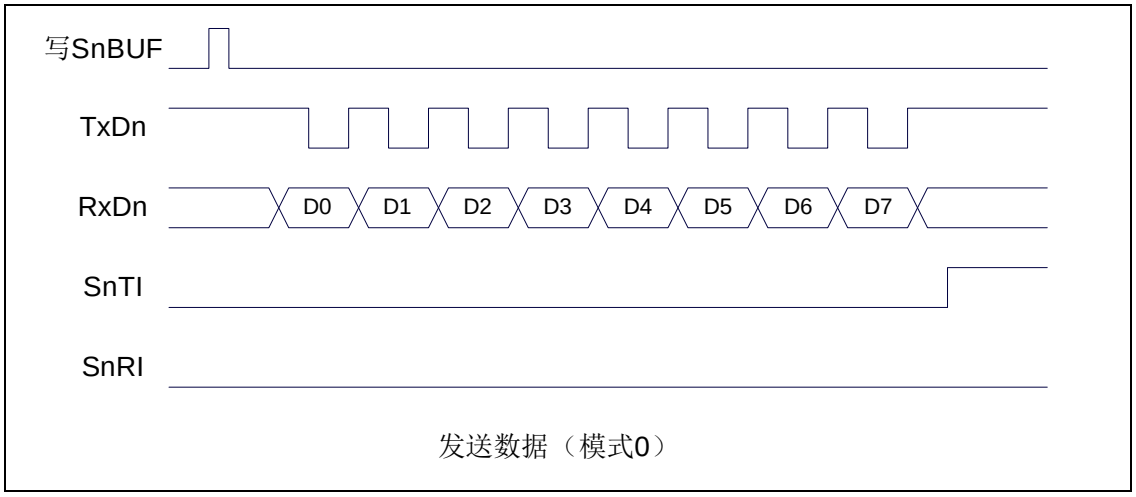
1：选择定时器 2 作为波特率发生器**（默认值）**

19.2.7 串口 1 模式 0，模式 0 波特率计算公式

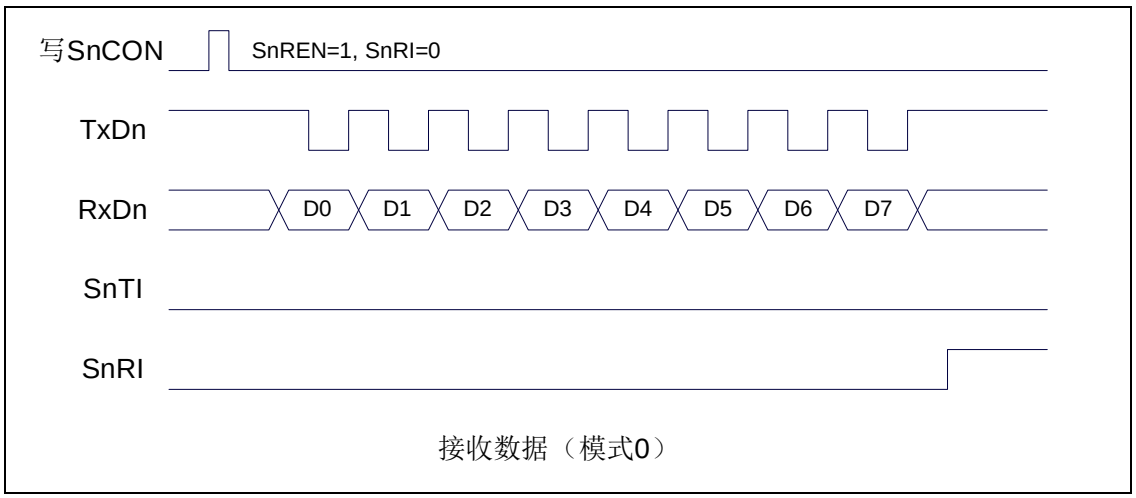
当串口 1 选择工作模式为模式 0 时，串行通信接口工作在同步移位寄存器模式，当串口模式 0 的通信速度设置位 UART_M0x6 为 0 时，其波特率固定为系统时钟时钟的 12 分频（SYSclk/12）；当设置 UART_M0x6 为 1 时，其波特率固定为系统时钟频率的 2 分频（SYSclk/2）。RxD 为串行通讯的数据口，TxD 为同步移位脉冲输出脚，发送、接收的是 8 位数据，低位在先。

模式 0 的发送过程：当主机执行将数据写入发送缓冲器 SBUF 指令时启动发送，串口即将 8 位数据以 SYSclk/12 或 SYSclk/2（由 UART_M0x6 确定是 12 分频还是 2 分频）的波特率从 RxD 管脚输出(从低位到高位)，发送完中断标志 TI 置 1，TxD 管脚输出同步移位脉冲信号。当写信号有效后，相隔一个时钟，发送控制端 SEND 有效(高电平)，允许 RxD 发送数据，同时允许 TxD 输出同步移位脉冲。一帧(8 位)数据发送完毕时，各控制端均恢复原状态，只有 TI 保持高电平，呈中断申请状态。在再次发送数据前，必须用软件将 TI 清 0。

模式 0 的接收过程：首先将接收中断请求标志 RI 清零并置位允许接收控制位 REN 时启动模式 0 接收过程。启动接收过程后，RxD 为串行数据输入端，TxD 为同步脉冲输出端。串行接收的波特率为 SYSclk/12 或 SYSclk/2（由 UART_M0x6 确定是 12 分频还是 2 分频）。当接收完成一帧数据（8 位）后，控制信号复位，中断标志 RI 被置 1，呈中断申请状态。当再次接收时，必须通过软件将 RI 清 0



发送数据（模式0）



接收数据（模式0）

工作于模式 0 时，必须清 0 多机通信控制位 SM2，使之不影响 TB8 位和 RB8 位。由于波特率固定为 SYSclk/12 或 SYSclk/2，无需定时器提供，直接由单片机的时钟作为同步移位脉冲。

串口 1 模式 0 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

UART_M0x6	波特率计算公式
0	$\text{波特率} = \frac{\text{SYSclk}}{12}$
1	$\text{波特率} = \frac{\text{SYSclk}}{2}$

19.2.8 串口 1 模式 1，模式 1 波特率计算公式

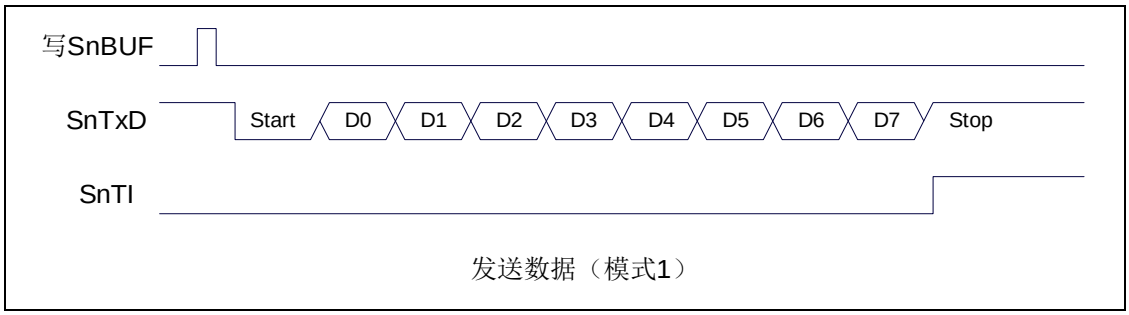
当软件设置 SCON 的 SM0、SM1 为“01”时，串口 1 则以模式 1 进行工作。此模式为 8 位 UART 格式，一帧信息为 10 位：1 位起始位，8 位数据位（低位在先）和 1 位停止位。波特率可变，即可根据需要进行设置波特率。TxD 为数据发送口，RxD 为数据接收口，串口全双工接受/发送。

模式 1 的发送过程：串行通信模式发送时，数据由串行发送端 TxD 输出。当主机执行一条写 SBUF 的指令就启动串行通信的发送，写“SBUF”信号还把“1”装入发送移位寄存器的第 9 位，并通知 TX 控制单元开始发送。移位寄存器将数据不断右移送 TxD 端口发送，在数据的左边不断移入“0”作补充。当数据的最高位移到移位寄存器的输出位置，紧跟其后的是第 9 位“1”，在它的左边各位全为“0”，这个状态条件，使 TX 控制单元作最后一次移位输出，然后使允许发送信号“SEND”失效，完成一帧信息的发送，并置位中断请求位 TI，即 TI=1，向主机请求中断处理。

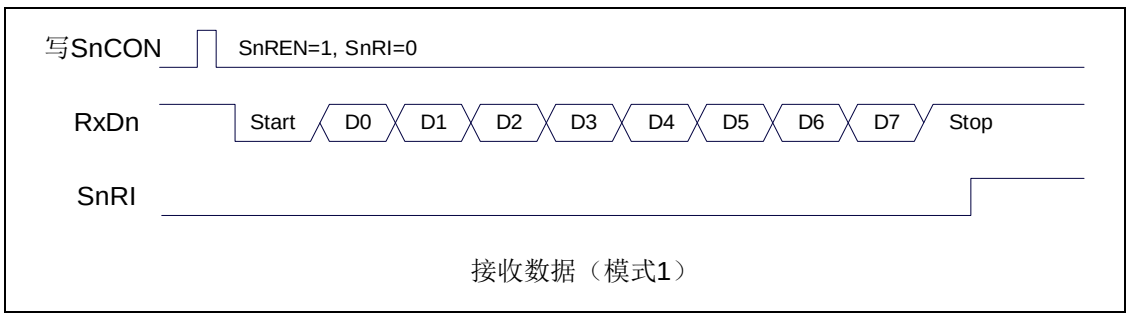
模式 1 的接收过程：当软件置位接收允许标志位 REN，即 REN=1 时，接收器便对 RxD 端口的信号进行检测，当检测到 RxD 端口发送从“1”→“0”的下降沿跳变时就启动接收器准备接收数据，并立即复位波特率发生器的接收计数器，将 1FFH 装入移位寄存器。接收的数据从接收移位寄存器的右边移入，已装入的 1FFH 向左边移出，当起始位"0"移到移位寄存器的最左边时，使 RX 控制器作最后一次移位，完成一帧的接收。若同时满足以下两个条件：

- RI=0；
- SM2=0 或接收到的停止位为 1。

则接收到的数据有效，实现装载入 SBUF，停止位进入 RB8，RI 标志位被置 1，向主机请求中断，若上述两条件不能同时满足，则接收到的数据作废并丢失，无论条件满足与否，接收器重又检测 RxD 端口上的"1"→"0"的跳变，继续下一帧的接收。接收有效，在响应中断后，RI 标志位必须由软件清 0。通常情况下，串行通信工作于模式 1 时，SM2 设置为"0"。



发送数据（模式1）



接收数据（模式1）

串口 1 的波特率是可变的，其波特率可由定时器 1 或者定时器 2 产生。当定时器采用 1T 模式时（12 倍速），相应的波特率的速度也会相应提高 12 倍。

串口 1 模式 1 的波特率计算公式如下表所示：（SYSclk 为系统工作频率）

选择定时器	定时器速度	重装值计算公式	波特率
定时器2	1T	定时器2重载值 = $65536-\frac{SYSclk}{4\times波特率}$	波特率 = $\frac{SYSclk}{4\times(65536-定时器重装数)}$
	12T	定时器2重载值 = $65536-\frac{SYSclk}{12\times4\times波特率}$	波特率 = $\frac{SYSclk}{12\times4\times(65536-定时器重装数)}$
定时器1模式0	1T	定时器1重载值 = $65536-\frac{SYSclk}{4\times波特率}$	波特率 = $\frac{SYSclk}{4\times(65536-定时器重装数)}$
	12T	定时器1重载值 = $65536-\frac{SYSclk}{12\times4\times波特率}$	波特率 = $\frac{SYSclk}{12\times4\times(65536-定时器重装数)}$
定时器1模式2	1T	定时器1重载值 = $256-\frac{2^{SMOD}\times SYSclk}{32\times波特率}$	波特率 = $\frac{2^{SMOD}\times SYSclk}{32\times(256-定时器重装数)}$
	12T	定时器1重载值 = $256-\frac{2^{SMOD}\times SYSclk}{12\times32\times波特率}$	波特率 = $\frac{2^{SMOD}\times SYSclk}{12\times32\times(256-定时器重装数)}$

下面为常用频率与常用波特率所对应定时器的重载值

频率(MHz)	波特率	定时器 2		定时器 1 模式 0		定时器 1 模式 2			
						SMOD=1		SMOD=0	
		1T 模式	12T 模式	1T 模式	12T 模式	1T 模式	12T 模式	1T 模式	12T 模式
11.0592	115200	FFE8H	FFFEH	FFE8H	FFFEH	FAH	-	FDH	-
	57600	FFD0H	FFFCH	FFD0H	FFFCH	F4H	FFH	FAH	-
	38400	FFB8H	FFFAH	FFB8H	FFFAH	EEH	-	F7H	-
	19200	FF70H	FFF4H	FF70H	FFF4H	DCH	FDH	EEH	-
	9600	FEE0H	FFE8H	FEE0H	FFE8H	B8H	FAH	DCH	FDH
18.432	115200	FFD8H	-	FFD8H	-	F6H	-	FBH	-
	57600	FFB0H	-	FFB0H	-	ECH	-	F6H	-
	38400	FF88H	FFF6H	FF88H	FFF6H	E2H	-	F1H	-
	19200	FF10H	FFECH	FF10H	FFECH	C4H	FBH	E2H	-
	9600	FE20H	FFD8H	FE20H	FFD8H	88H	F6H	C4H	FBH
22.1184	115200	FFD0H	FFFCH	FFD0H	FFFCH	F4H	FFH	FAH	-
	57600	FFA0H	FFF8H	FFA0H	FFF8H	E8H	FEH	F4H	FFH
	38400	FF70H	FFF4H	FF70H	FFF4H	DCH	FDH	EEH	-
	19200	FEE0H	FFE8H	FEE0H	FFE8H	B8H	FAH	DCH	FDH
	9600	FDC0H	FFD0H	FDC0H	FFD0H	70H	F4H	B8H	FAH

19.2.9 串口 1 模式 2，模式 2 波特率计算公式

当 SM0、SM1 两位为 10 时，串口 1 工作在模式 2。串口 1 工作模式 2 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 SCON 中的 TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 TB8（TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 SCON 的 RB8。TxD 为发送端口，RxD 为接收端口，以全双工模式进行接收/发送。

模式 2 的波特率固定为系统时钟的 64 分频或 32 分频（取决于 PCON 中 SMOD 的值）

串口 1 模式 2 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

SMOD	波特率计算公式
0	$\text{波特率} = \frac{\text{SYSclk}}{64}$
1	$\text{波特率} = \frac{\text{SYSclk}}{32}$

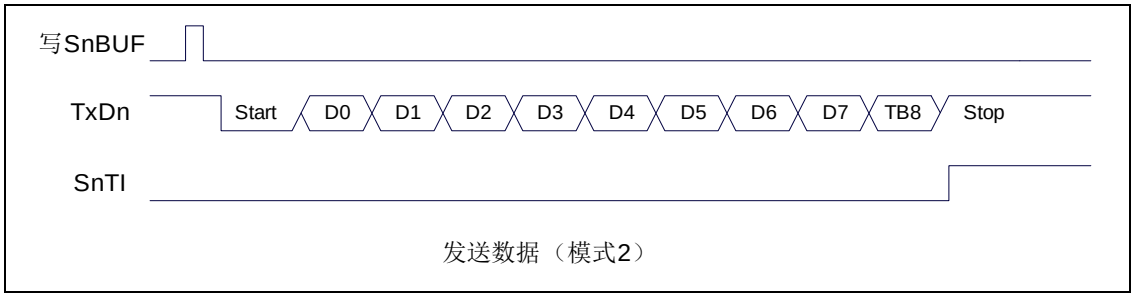
模式 2 和模式 1 相比，除波特率发生源略有不同，发送时由 TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收/发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

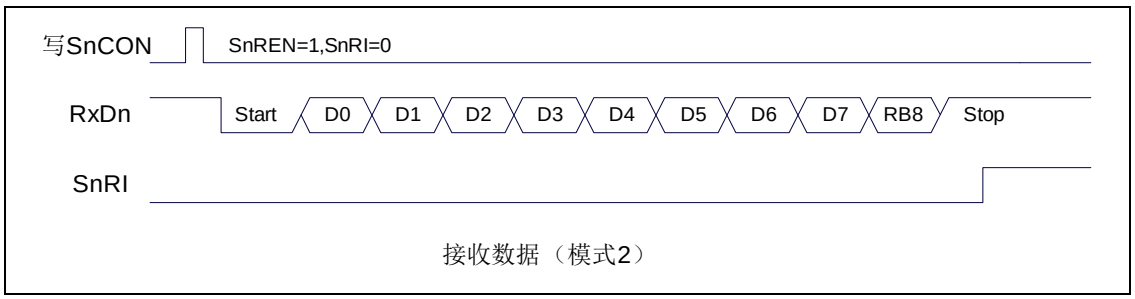
- RI=0
- SM2=0 或者 SM2=1 且接收到的第 9 数据位 RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 SBUF 和 RB8 中，RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 RI。无论上述条件满足与否，接收器又重新开始检测 RxD 输入端口的跳变信息，接收下一帧的输入信息。在模式 2 中，接收到的停止位与 SBUF、RB8 和 RI 无关。

通过软件对 SCON 中的 SM2、TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式2）



接收数据（模式2）

19.2.10 串口 1 模式 3，模式 3 波特率计算公式

当 SM0、SM1 两位为 11 时，串口 1 工作在模式 3。串行通信模式 3 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 SCON 中的 TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 TB8（TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 SCON 的 RB8。TxD 为发送端口，RxD 为接收端口，以全双工模式进行接收/发送。

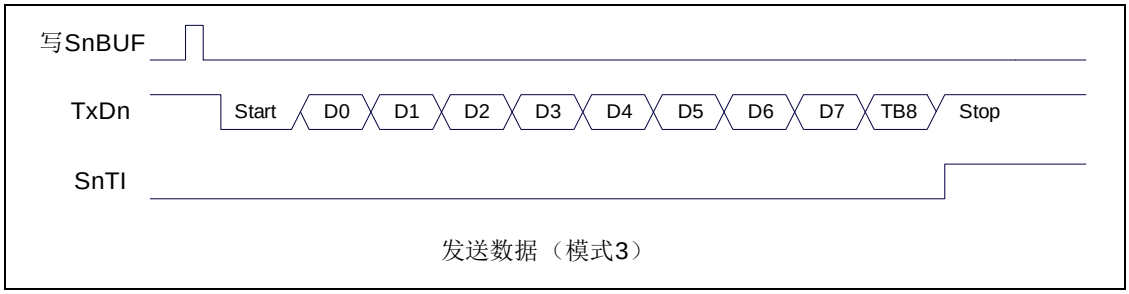
模式 3 和模式 1 相比，除发送时由 TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收‘发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

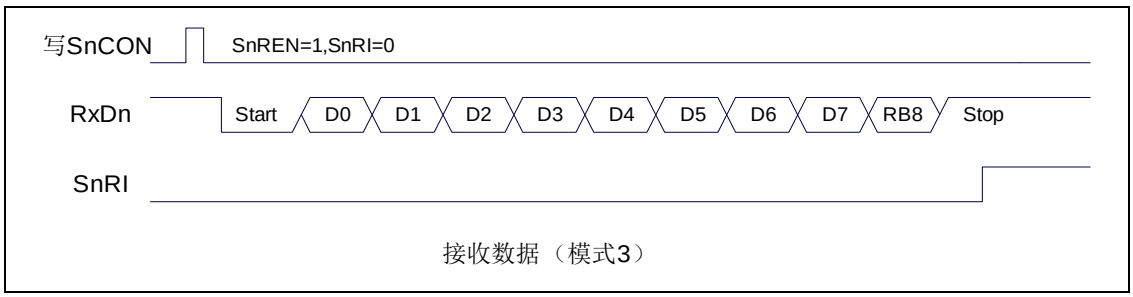
- RI=0
- SM2=0 或者 SM2=1 且接收到的第 9 数据位 RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 SBUF 和 RB8 中，RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 RI。无论上述条件满足与否，接收器又重新开始检测 RxD 输入端口的跳变信息，接收下一帧的输入信息。在模式 3 中，接收到的停止位与 SBUF、RB8 和 RI 无关。

通过软件对 SCON 中的 SM2、TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式3）



接收数据（模式3）

串口 1 模式 3 的波特率计算公式与模式 1 是完全相同的。请参考模式 1 的波特率计算公式。

19.2.11 自动地址识别，从机地址控制寄存器（SADDR，SADEN）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SADDR	A9H								
SADEN	B9H								

SADDR：从机地址寄存器

SADEN：从机地址屏蔽位寄存器

自动地址识别功能典型应用在多机通讯领域，其主要原理是从机系统通过硬件比较功能来识别来自于主机串口数据流中的地址信息，通过寄存器 SADDR 和 SADEN 设置的本机的从机地址，硬件自动对从机地址进行过滤，当来自于主机的从机地址信息与本机所设置的从机地址相匹配时，硬件产生串口中断；否则硬件自动丢弃串口数据，而不产生中断。当众多处于空闲模式的从机链接在一起时，只有从机地址相匹配的从机才会从空闲模式唤醒，从而可以大大降低从机 MCU 的功耗，即使从机处于正常工作状态也可避免不停地进入串口中断而降低系统执行效率。

要使用串口的自动地址识别功能，首先需要将参与通讯的 MCU 的串口通讯模式设置为模式 2 或者模式 3（通常都选择波特率可变的模式 3，因为模式 2 的波特率是固定的，不便于调节），并开启从机的 SCON 的 SM2 位。对于串口模式 2 或者模式 3 的 9 位数据位中，第 9 位数据（存放在 RB8 中）为地址/数据的标志位，当第 9 位数据为 1 时，表示前面的 8 位数据（存放在 SBUF 中）为地址信息。当 SM2 被设置为 1 时，从机 MCU 会自动过滤掉非地址数据（第 9 位为 0 的数据），而对 SBUF 中的地址数据（第 9 位为 1 的数据）自动与 SADDR 和 SADEN 所设置的本机地址进行比较，若地址相匹配，则会将 RI 置“1”，并产生中断，否则不予处理本次接收的串口数据。

从机地址的设置是通过 SADDR 和 SADEN 两个寄存器进行设置的。SADDR 为从机地址寄存器，里面存放本机的从机地址。SADEN 为从机地址屏蔽位寄存器，用于设置地址信息中的忽略位，设置方法如下：

例如

SADDR ＝ 11001010

SADEN ＝ 10000001

则匹配地址为 1xxxxxx0

即，只要主机送出的地址数据中的 bit0 为 0 且 bit7 为 1 就可以和本机地址相匹配

再例如

SADDR ＝ 11001010

SADEN ＝ 00001111

则匹配地址为 xxxx1010

即，只要主机送出的地址数据中的低 4 位为 1010 就可以和本机地址相匹配，而高 4 为被忽略，可以为任意值。

主机可以使用广播地址（FFH）同时选中所有的从机来进行通讯。

19.2.12 串口 1 同步模式控制寄存器 1（USARTCR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USARTCR1	7EFDC0H	S1LINEN	S1DORD	S1CLKEN	S1SPMOD	S1SPIEN	S1SPSLV	S1CPOL	S1CPHA

S1LINEN：LIN 模式使能位

0：禁止 LIN 模式

1：使能 LIN 模式

S1DORD：SPI 模式数据位发送/接收的顺序

0：先发送/接收数据的高位（MSB）

1：先发送/接收数据的低位（LSB）

S1CLKEN：SmartCard 模式时钟输出控制位

0：禁止时钟输出

1：使能时钟输出

S1SPMOD：SPI 模式使能位

0：禁止 SPI 模式

1：使能 SPI 模式

S1SPIEN：SPI 使能位

0：禁止 SPI 功能

1：使能 SPI 功能

当 USART1 需要工作在 SPI 模式时，必须先将 SPI 相关的模式寄存器、速度寄存器、时钟极性寄存器设置完成后，最后再设置 SPIEN 寄存器。

S1SPSLV：SPI 从机模式使能位

0：SPI 为主机模式

1：SPI 为从机模式

S1CPOL：SPI 时钟极性控制

0：SCLK 空闲时为低电平，SCLK 的前时钟沿为上升沿，后时钟沿为下降沿

1：SCLK 空闲时为高电平，SCLK 的前时钟沿为下降沿，后时钟沿为上升沿

S1CPHA：SPI 时钟相位控制

0：数据 SS 管脚为低电平驱动第一位数据并在 SCLK 的后时钟沿改变数据，前时钟沿采样数据

1：数据在 SCLK 的前时钟沿驱动，后时钟沿采样

19.2.13 串口 1 同步模式控制寄存器 2，硬件奇偶校验位设置（USARTCR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USARTCR2	7EFDC1H	S1IREN	S1IRLP	S1SCEN	S1NACK	S1HDSEL	S1PCEN	S1PS	S1PE

S1IREN：IrDA 模式使能位

0：禁止 IrDA 模式

1：使能 IrDA 模式

I S1RLP：IrDA 低电模式控制位

0：IrDA 为普通模式

1：IrDA 为低电模式

S1SCEN：SmartCard 模式使能位

0：禁止 SmartCard 模式

1：使能 SmartCard 模式

S1NACK：SmartCard 模式 NACK 输出使能位

0：禁止输出 NACK 信号

1：使能输出 NACK 信号

S1HDSEL：单线半双工模式使能位

0：禁止单线半双工模式

1：使能单线半双工模式

S1PCEN：硬件自动产生硬件奇偶校验位控制使能

0：禁止自动产生硬件奇偶校验位（串口的校验位为 TB8 设置的值）

1：使能自动产生硬件奇偶校验位

S1PS：硬件奇偶校验位模式选择

0：硬件根据 SBUF 的值自动产生偶校验位

1：硬件根据 SBUF 的值自动产生奇校验位

S1PE：校验位错误标志（必须软件清零）

0：无检验错误

1：有校验错误（串口接收 DMA 过程中如果发生接收数据校验位错误，DMA 不会停止，但校验位错误标志会一直保持直到 DMA 完成）

19.2.14 串口 1 同步模式控制寄存器 3（USARTCR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USARTCR3	7EFDC2H	S1IrDA_LPBAUD[7:0]							

S1IrDA_LPBAUD：IrDA 低电模式波特率控制寄存器

IrDA_LPBAUD[7:0]	IrDA 低电模式波特率
0	SYSclk/16/256
1	SYSclk/16/1
2	SYSclk/16/2
...	...
255	SYSclk/16/255

19.2.15 串口 1 同步模式控制寄存器 4（USARTCR4）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USARTCR4	7EFDC3H	-	-	-	-	S1SCCKS[1:0]		S1SPICKS[1:0]	

S1SCCKS[1:0]：SmartCard 模式时钟选择

SCCKS[1:0]	SmartCard 模式时钟
00	SYSclk/64
01	SYSclk/32
10	SYSclk/16
11	SYSclk/8

S1SPICKS[1:0]：SPI 模式时钟选择

SPICKS[1:0]	SPI 模式时钟
00	SYSclk/4
01	SYSclk/8
10	SYSclk/16
11	SYSclk/2

19.2.16 串口 1 同步模式控制寄存器 5（USARTCR5）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USARTCR5	7EFDC4H	S1BRKDET	S1HDRER	S1SLVEN	S1ASYNC	S1TXCF	S1SENDBK	S1HDRDET	S1SYNC

S1BRKDET：LIN 从模式间隔场（BREAK）检测标志

0：未检测到 LIN 间隔场

1：检测到 LIN 间隔场，需要软件清零

S1HDRER：LIN 从模式报文头（HEADER）错误

0：未检测到 LIN 报文头错误

1：检测到 LIN 报文头错误，需要软件清零

S1SLVEN：LIN 从机模式使能位

0：LIN 为主机模式

1：LIN 为从机模式

S1ASYNC：LIN 自动同步功能使能位

0：禁止自动同步

1：使能自动同步

【注意】：从机使能自动同步功能后，如果从机跟主机波特率存在小范围误差，从机收到主机同步信号后，会自动将从机波特率设置成与主机相同的波特率。

例如，主机波特率 9600：

从机使能自动同步时，如果从机波特率在 9100～10000 附近，收到主机同步信号后，从机会自动将本身波特率设置为 9600。误差自动同步范围 -5.2%～+4.16%左右。

S1TXCF：传输冲突标志位。单线半双工模式、LIN 模式和 SmartCard 模式有效

0：未检测到传输冲突（发送的数据与接收的数据相同）

1：检测到传输冲突（发送的数据与接收的数据不同）

S1SENDBK：发送间隔场。软件写 1 触发发送间隔场，发送完成后硬件自动清 0

S1HDRDET：LIN 从模式报文头（HEADER）检测标志

0：未检测到 LIN 报文头

1：检测到 LIN 报文头，需要软件清零

S1SYNC：LIN 从模式同步场检测标志。

正确分析到同步场后，硬件将 SYNC 标志为置 1。在接收标志符场时硬件会自动清零 SYNC

19.2.17 串口 1 同步模式保护时间寄存器（USARTGTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USARTGTR	7EFDC5H								

USARTGTR 寄存器仅在 SmartCard 模式有效。该寄存器是根据波特率时钟数给出保护时间值。TI 标志在 SmartCard 发送的数据位等于 S1USARTGTR 寄存器所设置的保护时间值时被硬件置 1。

注：USARTGTR 寄存器的值应大于 11

19.2.18 串口 1 同步模式波特率寄存器（USARTBR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USARTBRH	7EFDC6H	USARTBR[15:8]							
USARTBRL	7EFDC7H	USARTBR[7:0]							

LIN 模式、IrDA 模式和 SmartCard 模式时的波特率计算公式

$$\text{同步波特率} = \frac{\text{SYSclk}}{16 * \text{USARTBR[15:0]}}$$

19.2.19 串口 1 接收超时控制寄存器（UR1TOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR1TOCR	7EFD70H	UR1ENTO	UR1ENTOI	UR1SCALE	-	-	-	-	-

UR1ENTO：串口 1 接收超时功能控制位

0：禁止串口 1 接收超时功能

1：使能串口 1 接收超时功能

UR1ENTOI：串口 1 接收超时中断控制位

0：禁止串口 1 接收超时中断

1：使能串口 1 接收超时中断

UR1SCALE：串口 1 超时计数时钟源选择

0：串口数据位率（波特率）

1：系统时钟

19.2.20 串口 1 超时状态寄存器（UR1TOSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR1TOSR	7EFD71H	UR1CTOIF	-	-	-	-	-	-	UR1TOIF

UR1CTOIF：写“1”清除串口 1 超时中断标志位 TOIF。（只写）

UR1TOIF：串口 1 超时中断请求标志位。（只读）

当发生串口 1 超时时，TOIF 会被硬件置“1”。如果 ENTOI 为 1 时会产生串口中断，中断入口地址为原串口 1 的中断入口地址。标志位需要软件向 CTOIF 位写“1”清零

19.2.21 串口 1 超时长度控制寄存器（UR1TOTE/H/L）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR1TOTE	7EFD88H	UR1TM[23:16]							
UR1TOTH	7EFD72H	UR1TM[15:8]							
UR1TOTL	7EFD73H	UR1TM[7:0]							

TM[23:0]：串口 1 超时时间控制位。

当串口 1 处于接收空闲状态时，内部超时计数器根据 SCALE 寄存器所选择的时钟源进行计数，当计数时间达到 TM 所设置的超时时间时，便会产生超时中断。当串口 1 接收数据完成时，复位内部超时计数器，重新进行超时计数。

注：

1、如果需要使能接收超时中断功能，则 TM 不可设置为 0，且 UR1TOTL、UR1TOTH、UR1TOTE 寄存器的设置必须先设置 UR1TOTL 和 UR1TOTH，最后设置 UR1TOTE

2、必须使能串口接收才有接收超时功能

3、接收超时功能必须在正确接收到一字节数据后才能触发

4、正在接收过程中，无接收超时功能

19.3 串口 2（同步/异步串口 USART2）

19.3.1 串口 2 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW2	BAH	EAXFR	-	I2C_S[1:0]		-	S4_S[0]	S3_S[0]	**S2_S[0]**
P_SWX1	7EFD69H	S4_S[1]	S3_S[1]	**S2_S[1]**	-	CAN2_S[2]	CAN_S[2]	-	SWP54P12

S2_S[1:0]：串口 2 功能脚选择位

S2_S[1:0]	RxD2	TxD2
00	P1.0	P1.1
01	P4.6	P4.7
10	P8.4	P8.5
11	PA.0	PA.1

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW3	BBH	I2S_S		**S2SPI_S[1:0]**		S1SPI_S[1:0]		-	-

S2SPI_S[1:0]：USART2 的 SPI 功能脚选择位

S2SPI_S[1:0]	S2SS	S2MOSI	S2MISO	S2SCLK
00	P1.2	P1.3	P1.4	P1.5
01	P2.2	P2.3	P2.4	P2.5
10	P5.4	P4.0	P4.1	P4.3
11	P7.4	P7.5	P7.6	P7.7

19.3.2 串口 2 相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S2CON	串口 2 控制寄存器	9AH	S2SM0/FE	S2SM1	S2SM2	S2REN	S2TB8	S2RB8	S2TI	S2RI	0000,0000
S2BUF	串口 2 数据寄存器	9BH									0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S2CFG	串口 2 配置寄存器	7EFDB4H	-	S2MOD0	S2M0x6	-	-	-	-	1	000x,xxx1
S2ADDR.	串口 2 从机地址寄存器	7EFDB5H									0000,0000
S2ADEN	串口 2 从机地址屏蔽寄存器	7EFDB6H									0000,0000
USART2CR1	串口 2 控制寄存器 1	7EFDC8H	S2LINEN	S2DORD	S2CLKEN	S2SPMOD	S2SPIEN	S2SPSLV	S2CPOL	S2CPHA	0000,0000
USART2CR2	串口 2 控制寄存器 2	7EFDC9H	S2IREN	S2IRLP	S2SCEN	S2NACK	S2HDSEL	S2PCEN	S2PS	S2PE	0000,0000
USART2CR3	串口 2 控制寄存器 3	7EFDCAH	S2IrDA_LPBAUD[7:0]								0000,0000
USART2CR4	串口 2 控制寄存器 4	7EFDCBH	-	-	-	-	S2SCCKS[1:0]		S2SPICKS[1:0]		xxxx,0000
USART2CR5	串口 2 控制寄存器 5	7EFDCCH	S2BRKDET	S2HDRER	S2SLVEN	S2ASYNC	S2TXCF	S2SENDBK	S2HDRDET	S2SYNCAN	0000,0000
USART2GTR	串口 2 保护时间寄存器	7EFDCDH	USART2GTR[7:0]								0000,0000
USART2BRH	串口 2 波特率寄存器	7EFDCEH	USART2BR[15:8]								0000,0000
USART2BRL	串口 2 波特率寄存器	7EFDCFH	USART2BR[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
UR2TOCR	串口 2 接收超时控制寄存器	7EFD74H	UR2ENTO	UR2ENTOI	UR2SCALE	-	-	-	-	-	000x,xxxx
UR2TOSR	串口 2 接收超时状态寄存器	7EFD75H	-	-	-	-	-	-	-	UR2TOIF	xxxx,xxx0
UR2TOTH	串口 2 接收超时长度寄存器	7EFD76H	UR2TM[15:8]								0000,0000
UR2TOTL	串口 2 接收超时长度寄存器	7EFD77H	UR2TM[7:0]								0000,0000
UR2TOTE	串口 2 接收超时长度寄存器	7EFD89H	UR2TM[23:16]								0000,0000

19.3.3 串口 2 控制寄存器（S2CON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S2CON	9AH	S2SM0/FE	S2SM1	S2SM2	S2REN	S2TB8	S2RB8	S2TI	S2RI

S2SM0/FE：当S2CFG寄存器中的S2MOD0位为1时，该位为帧错误检测标志位。当串口2在接收过程中检测到一个无效停止位时，通过串口2接收器将该位置1，必须由软件清零。当S2CFG寄存器中的S2MOD0位为0时，该位和S2SM1一起指定串口2的通信工作模式，如下表所示：

S2SM0	S2SM1	串口2工作模式	功能说明
0	0	模式0	同步移位串行方式
0	1	模式1	可变波特率8位数据方式
1	0	模式2	固定波特率9位数据方式
1	1	模式3	可变波特率9位数据方式

当使能 FE 功能后，串口的工作模式只能使用 SM1 进行设置，且只能设置为模式 0 或者模式 1。FE 当作帧错误标志位时，此标志位必须软件写 0 清零。

S2SM2：允许模式 2 或模式 3 多机通信控制位。当串口 2 使用模式 2 或模式 3 时，如果 S2SM2 位为 1 且 S2REN 位为 1，则接收机处于地址帧筛选状态。此时可以利用接收到的第 9 位（即 S2RB8）来筛选地址帧，若 S2RB8=1，说明该帧是地址帧，地址信息可以进入 S2BUF，并使 S2RI 为 1，进而在中断服务程序中再进行地址号比较；若 S2RB8=0，说明该帧不是地址帧，应丢掉且保持 S2RI=0。在模式 2 或模式 3 中，如果 S2SM2 位为 0 且 S2REN 位为 1，接收收机处于地址帧筛选被禁止状态，不论收到的 S2RB8 为 0 或 1，均可使接收到的信息进入 S2BUF，并使 S2RI=1，此时 S2RB8 通常为校验位。模式 1 和模式 0 为非多机通信方式，在这两种方式时，S2SM2 应设置为 0。

S2REN：允许/禁止串口接收控制位

0：禁止串口接收数据

1：允许串口接收数据

S2TB8：当串口 2 使用模式 2 或模式 3 时，S2TB8 为要发送的第 9 位数据，按需要由软件置位或清 0。在模式 0 和模式 1 中，该位不用。

S2RB8：当串口 2 使用模式 2 或模式 3 时，S2RB8 为接收到的第 9 位数据，一般用作校验位或者地址帧/数据帧标志位。在模式 0 和模式 1 中，该位不用。

S2TI：串口 2 发送中断请求标志位。在模式 0 中，当串口发送数据第 8 位结束时，由硬件自动将 S2TI 置 1，向主机请求中断，响应中断后 S2TI 必须用软件清零。在其他模式中，则在停止位开始发送时由硬件自动将 S2TI 置 1，向 CPU 发请求中断，响应中断后 S2TI 必须用软件清零。

S2RI：串口 2 接收中断请求标志位。在模式 0 中，当串口接收第 8 位数据结束时，由硬件自动将 S2RI 置 1，向主机请求中断，响应中断后 S2RI 必须用软件清零。在其他模式中，串行接收到停止位的中间时刻由硬件自动将 S2RI 置 1，向 CPU 发中断申请，响应中断后 S2RI 必须由软件清零。

19.3.4 串口 2 数据寄存器（S2BUF）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S2BUF	9BH								

S2BUF：串口 1 数据接收/发送缓冲区。S2BUF 实际是 2 个缓冲器，读缓冲器和写缓冲器，两个操作分别对应两个不同的寄存器，1 个是只写寄存器（写缓冲器），1 个是只读寄存器（读缓冲器）。对 S2BUF 进行读操作，实际是读取串口接收缓冲区，对 S2BUF 进行写操作则是触发串口开始发送数据。

19.3.5 串口 2 配置寄存器（S2CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S2CFG	7EFDB4H	S2MOD	S2MOD0	S2M0x6					1

S2MOD：串口 2 波特率控制位

0：串口 2 之模式 0/1/2/3 的波特率都不加倍

1：串口 2 之模式 2 的波特率加倍

S2MOD0：帧错误检测控制位

0：无帧错检测功能

1：使能帧错误检测功能。此时 S2CON 的 S2SM0/FE 为 FE 功能，即为帧错误检测标志位。

S2M0x6：串口 2 模式 0 的通讯速度控制

0：串口 2 模式 0 的波特率不加倍，固定为 Fosc/12

1：串口 2 模式 0 的波特率 6 倍速，即固定为 Fosc/12*6 ＝ Fosc/2

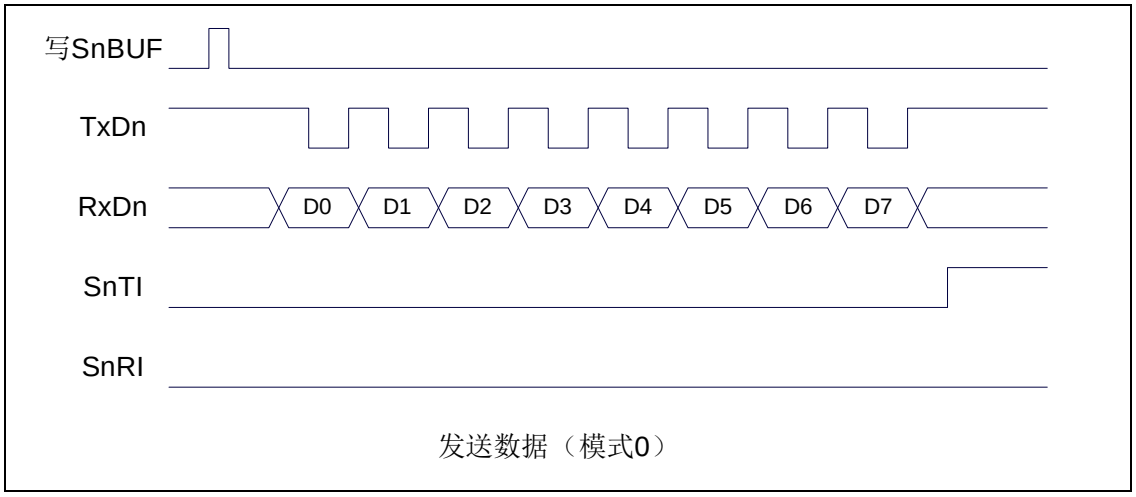
1：此位固定为 1。

19.3.6 串口 2 模式 0，模式 0 波特率计算公式

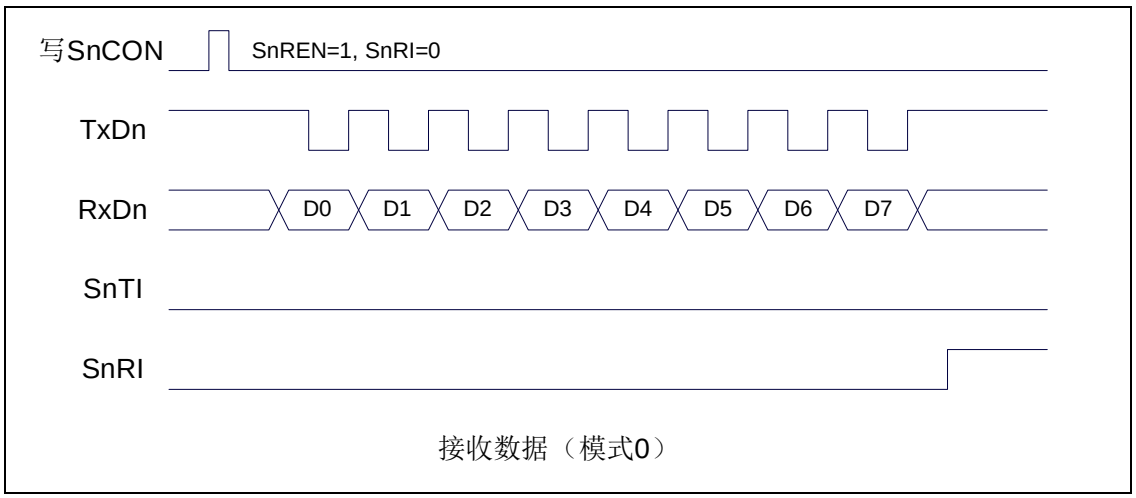
当串口 2 选择工作模式为模式 0 时，串行通信接口工作在同步移位寄存器模式，当串口模式 0 的通信速度设置位 S2M0x6 为 0 时，其波特率固定为系统时钟时钟的 12 分频（SYSclk/12）；当设置 S2M0x6 为 1 时，其波特率固定为系统时钟频率的 2 分频（SYSclk/2）。RxD2 为串行通讯的数据口，TxD2 为同步移位脉冲输出脚，发送、接收的是 8 位数据，低位在先。

模式 0 的发送过程：当主机执行将数据写入发送缓冲器 S2BUF 指令时启动发送，串口即将 8 位数据以 SYSclk/12 或 SYSclk/2（由 S2M0x6 确定是 12 分频还是 2 分频）的波特率从 RxD2 管脚输出(从低位到高位)，发送完中断标志 S2TI 置 1，TxD2 管脚输出同步移位脉冲信号。当写信号有效后，相隔一个时钟，发送控制端 SEND 有效(高电平)，允许 RxD2 发送数据，同时允许 TxD2 输出同步移位脉冲。一帧(8 位)数据发送完毕时，各控制端均恢复原状态，只有 S2TI 保持高电平，呈中断申请状态。在再次发送数据前，必须用软件将 S2TI 清 0。

模式 0 的接收过程：首先将接收中断请求标志 S2RI 清零并置位允许接收控制位 S2REN 时启动模式 0 接收过程。启动接收过程后，RxD2 为串行数据输入端，TxD2 为同步脉冲输出端。串行接收的波特率为 SYSclk/12 或 SYSclk/2（由 S2M0x6 确定是 12 分频还是 2 分频）。当接收完成一帧数据（8 位）后，控制信号复位，中断标志 S2RI 被置 1，呈中断申请状态。当再次接收时，必须通过软件将 S2RI 清 0



发送数据（模式0）



接收数据（模式0）

工作于模式 0 时，必须清 0 多机通信控制位 S2SM2，使之不影响 S2TB8 位和 S2RB8 位。由于波特率固定为 SYSclk/12 或 SYSclk/2，无需定时器提供，直接由单片机的时钟作为同步移位脉冲。

串口 2 模式 0 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

S2M0x6	波特率计算公式
0	$\text{波特率} = \frac{\text{SYSclk}}{12}$
1	$\text{波特率} = \frac{\text{SYSclk}}{2}$

19.3.7 串口 2 模式 1，模式 1 波特率计算公式

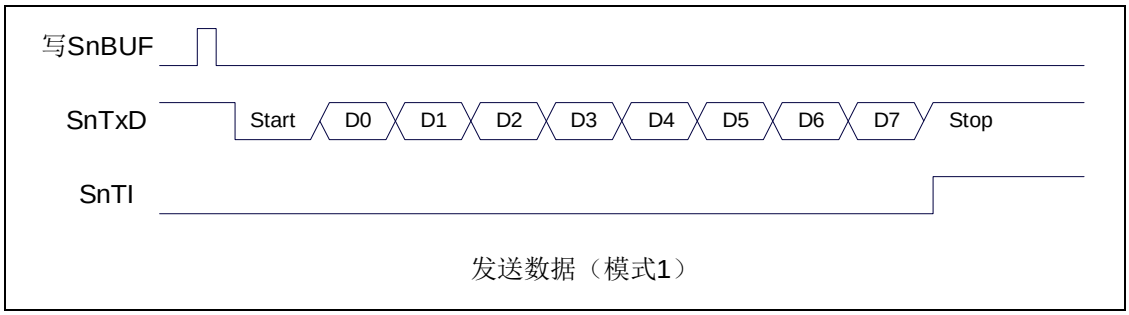
当软件设置 S2CON 的 S2SM0、S2SM1 为“01”时，串口 2 则以模式 1 进行工作。此模式为 8 位 UART 格式，一帧信息为 10 位：1 位起始位，8 位数据位（低位在先）和 1 位停止位。波特率可变，即可根据需要进行设置波特率。TxD2 为数据发送口，RxD2 为数据接收口，串口全双工接受/发送。

模式 1 的发送过程：串行通信模式发送时，数据由串行发送端 TxD2 输出。当主机执行一条写 S2BUF 的指令就启动串行通信的发送，写“S2BUF”信号还把“1”装入发送移位寄存器的第 9 位，并通知 TX 控制单元开始发送。移位寄存器将数据不断右移送 TxD 端口发送，在数据的左边不断移入“0”作补充。当数据的最高位移到移位寄存器的输出位置，紧跟其后的是第 9 位“1”，在它的左边各位全为“0”，这个状态条件，使 TX 控制单元作最后一次移位输出，然后使允许发送信号“SEND”失效，完成一帧信息的发送，并置位中断请求位 S2TI，即 S2TI=1，向主机请求中断处理。

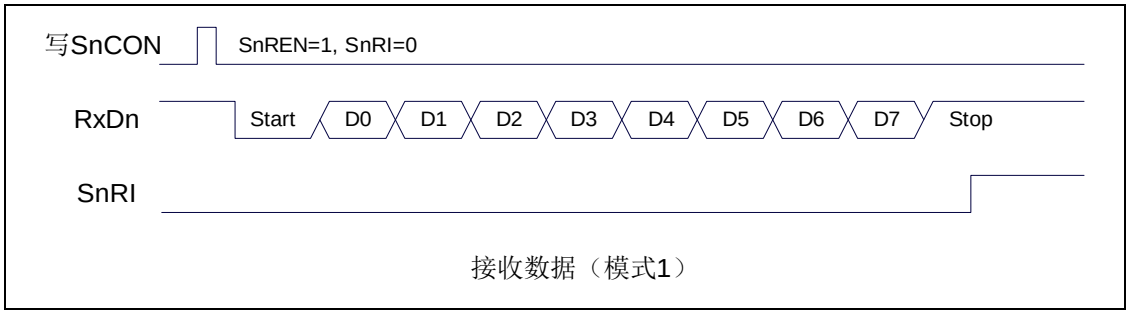
模式 1 的接收过程：当软件置位接收允许标志位 S2REN，即 S2REN=1 时，接收器便对 RxD2 端口的信号进行检测，当检测到 RxD2 端口发送从“1”→“0”的下降沿跳变时就启动接收器准备接收数据，并立即复位波特率发生器的接收计数器，将 1FFH 装入移位寄存器。接收的数据从接收移位寄存器的右边移入，已装入的 1FFH 向左边移出，当起始位"0"移到移位寄存器的最左边时，使 RX 控制器作最后一次移位，完成一帧的接收。若同时满足以下两个条件：

- S2RI=0；
- S2SM2=0 或接收到的停止位为 1。

则接收到的数据有效，实现装载入 S2BUF，停止位进入 S2RB8，S2RI 标志位被置 1，向主机请求中断，若上述两条件不能同时满足，则接收到的数据作废并丢失，无论条件满足与否，接收器重又检测 RxD2 端口上的"1"→"0"的跳变，继续下一帧的接收。接收有效，在响应中断后，S2RI 标志位必须由软件清 0。通常情况下，串行通信工作于模式 1 时，S2SM2 设置为"0"。



发送数据（模式1）



接收数据（模式1）

串口 2 的波特率是可变的，其波特率固定由定时器 2 产生。当定时器采用 1T 模式时（12 倍速），相应的波特率的速度也会相应提高 12 倍。

串口 2 模式 1 的波特率计算公式如下表所示：（SYSclk 为系统工作频率）

选择定时器	定时器速度	重装值计算公式	波特率
定时器 2	1T	定时器重载值 = 65536 - $\frac{SYSclk}{4\times波特率}$	波特率 = $\frac{SYSclk}{4\times(65536 - 定时器重装数)}$
	12T	定时器重载值 = 65536 - $\frac{SYSclk}{12\times4\times波特率}$	波特率 = $\frac{SYSclk}{12\times4\times(65536 - 定时器重装数)}$

下面为常用频率与常用波特率所对应定时器的重载值

频率 (MHz)	波特率	定时器 2	
		1T 模式	12T 模式
11.0592	115200	FFE8H	FFFEH
	57600	FFD0H	FFFCH
	38400	FFB8H	FFFAH
	19200	FF70H	FFF4H
	9600	FEE0H	FFE8H
18.432	115200	FFD8H	-
	57600	FFB0H	-
	38400	FF88H	FFF6H
	19200	FF10H	FFECH
	9600	FE20H	FFD8H
22.1184	115200	FFD0H	FFFCH
	57600	FFA0H	FFF8H
	38400	FF70H	FFF4H
	19200	FEE0H	FFE8H
	9600	FDC0H	FFD0H

19.3.8 串口 2 模式 2，模式 2 波特率计算公式

当 S2SM0、S2SM1 两位为 10 时，串口 2 工作在模式 2。串口 2 工作模式 2 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S2CON 中的 S2TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S2TB8（S2TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S2CON 的 S2RB8。TxD2 为发送端口，RxD2 为接收端口，以全双工模式进行接收/发送。

模式 2 的波特率固定为系统时钟的 64 分频或 32 分频（取决于 S2CFG 中 S2MOD 的值）

串口 2 模式 2 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

SMOD	波特率计算公式
0	$波特率 = \frac{SYSclk}{64}$
1	$波特率 = \frac{SYSclk}{32}$

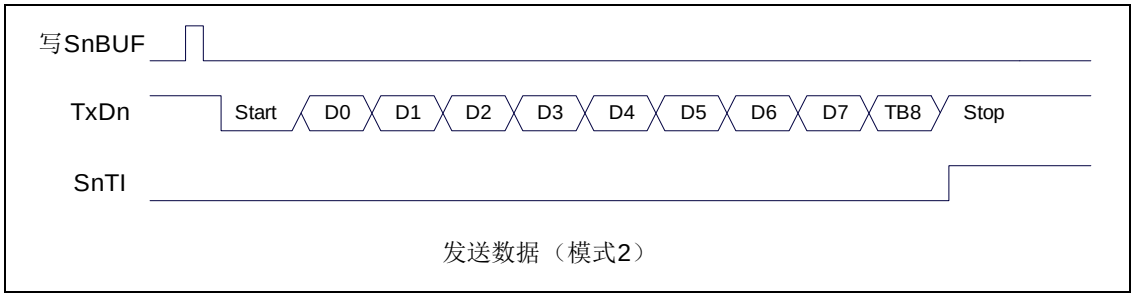
模式 2 和模式 1 相比，除波特率发生源略有不同，发送时由 S2TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收/发送操作过程及时序也基本相同。

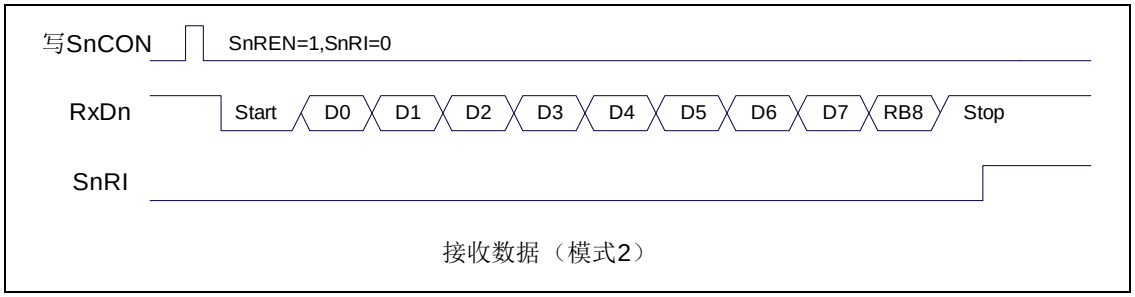
当接收器接收完一帧信息后必须同时满足下列条件：

- S2RI=0
- S2SM2=0 或者 S2SM2=1 且接收到的第 9 数据位 S2RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S2BUF 和 S2RB8 中，S2RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S2RI。无论上述条件满足与否，接收器又重新开始检测 RxD2 输入端口的跳变信息，接收下一帧的输入信息。在模式 2 中，接收到的停止位与 S2BUF、S2RB8 和 S2RI 无关。

通过软件对 S2CON 中的 S2SM2、S2TB8 的设置以及通信协议的约定，为多机通信提供了方便。





19.3.9 串口 2 模式 3，模式 3 波特率计算公式

当 S2SM0、S2SM1 两位为 11 时，串口 2 工作在模式 3。串行通信模式 3 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S2CON 中的 S2TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S2TB8（S2TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S2CON 的 S2RB8。TxD2 为发送端口，RxD2 为接收端口，以全双工模式进行接收/发送。

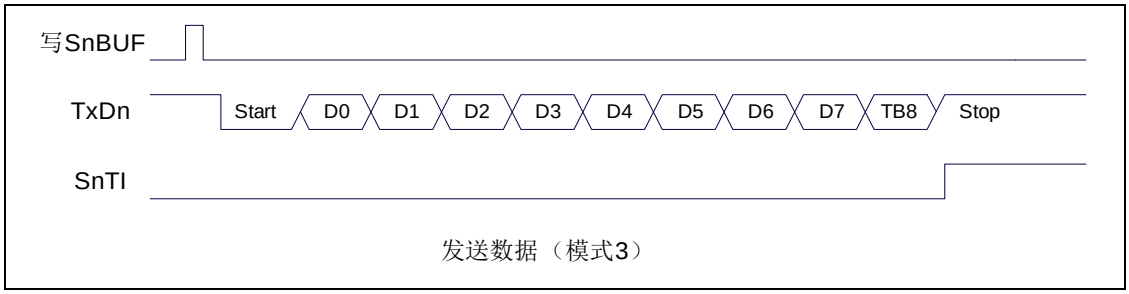
模式 3 和模式 1 相比，除发送时由 S2TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收‘发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

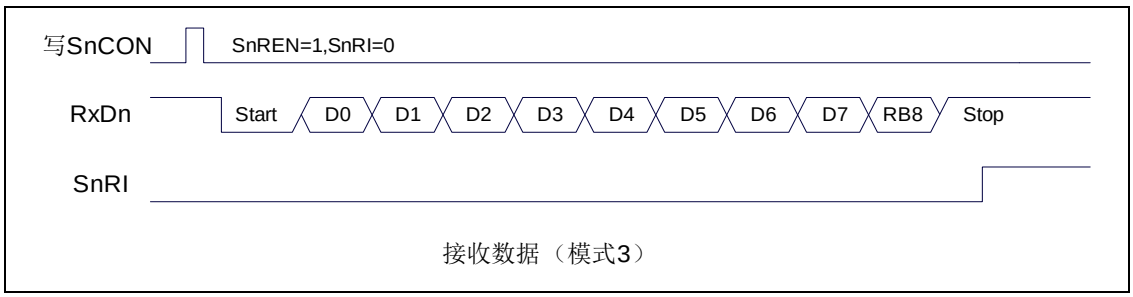
- S2RI=0
- S2SM2=0 或者 S2SM2=1 且接收到的第 9 数据位 S2RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S2BUF 和 S2RB8 中，S2RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S2RI。无论上述条件满足与否，接收器又重新开始检测 RxD2 输入端口的跳变信息，接收下一帧的输入信息。在模式 3 中，接收到的停止位与 S2BUF、S2RB8 和 S2RI 无关。

通过软件对 S2CON 中的 S2SM2、S2TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式3）



接收数据（模式3）

串口 2 模式 3 的波特率计算公式与模式 1 是完全相同的。请参考模式 1 的波特率计算公式。

19.3.10 串口 2 自动地址识别，从机地址控制寄存器（S2ADDR，S2ADEN）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S2ADDR	7EFDB5H								
S2ADEN	7EFDB6H								

S2ADDR：从机地址寄存器

S2ADEN：从机地址屏蔽位寄存器

自动地址识别功能典型应用在多机通讯领域，其主要原理是从机系统通过硬件比较功能来识别来自于主机串口 2 数据流中的地址信息，通过寄存器 S2ADDR 和 S2ADEN 设置的本机的从机地址，硬件自动对从机地址进行过滤，当来自于主机的从机地址信息与本机所设置的从机地址相匹配时，硬件产生串口 2 中断；否则硬件自动丢弃串口 2 数据，而不产生中断。当众多处于空闲模式的从机链接在一起时，只有从机地址相匹配的从机才会从空闲模式唤醒，从而可以大大降低从机 MCU 的功耗，即使从机处于正常工作状态也可避免不停地进入串口 2 中断而降低系统执行效率。

要使用串口 2 的自动地址识别功能，首先需要将参与通讯的 MCU 的串口 2 通讯模式设置为模式 1，并开启从机的 S2CON 的 S2SM2 位。对于串口 2 模式 1 的 9 位数据位中，第 9 位数据（存放在 S2RB8 中）为地址/数据的标志位，当第 9 位数据为 1 时，表示前面的 8 位数据（存放在 S2BUF 中）为地址信息。当 S2SM2 被设置为 1 时，从机 MCU 会自动过滤掉非地址数据（第 9 位为 0 的数据），而对 S2BUF 中的地址数据（第 9 位为 1 的数据）自动与 S2ADDR 和 S2ADEN 所设置的本机地址进行比较，若地址相匹配，则会将 S2RI 置“1”，并产生中断，否则不予处理本次接收的串口 2 数据。

从机地址的设置是通过 S2ADDR 和 S2ADEN 两个寄存器进行设置的。S2ADDR 为从机地址寄存器，里面存放本机的从机地址。S2ADEN 为从机地址屏蔽位寄存器，用于设置地址信息中的忽略位，设置方法如下：

例如

S2ADDR ＝ 11001010

S2ADEN ＝ 10000001

则匹配地址为 1xxxxxx0

即，只要主机送出的地址数据中的 bit0 为 0 且 bit7 为 1 就可以和本机地址相匹配

再例如

S2ADDR ＝ 11001010

S2ADEN ＝ 00001111

则匹配地址为 xxxx1010

即，只要主机送出的地址数据中的低 4 位为 1010 就可以和本机地址相匹配，而高 4 为被忽略，可以为任意值。

主机可以使用广播地址（FFH）同时选中所有的从机来进行通讯。

19.3.11 串口 2 同步模式控制寄存器 1（USART2CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART2CR1	7EFDC8H	S2LINEN	S2DORD	S2CLKEN	S2SPMOD	S2SPIEN	S2SPSLV	S2CPOL	S2CPHA

S2LINEN：LIN 模式使能位

0：禁止 LIN 模式

1：使能 LIN 模式

S2DORD：SPI 模式数据位发送/接收的顺序

0：先发送/接收数据的高位（MSB）

1：先发送/接收数据的低位（LSB）

S2CLKEN：SmartCard 模式时钟输出控制位

0：禁止时钟输出

1：使能时钟输出

S2SPMOD：SPI 模式使能位

0：禁止 SPI 模式

1：使能 SPI 模式

S2SPIEN：SPI 使能位

0：禁止 SPI 功能

1：使能 SPI 功能

当 USART2 需要工作在 SPI 模式时，必须先将 SPI 相关的模式寄存器、速度寄存器、时钟极性寄存器设置完成后，最后再设置 SPIEN 寄存器。

S2SPSLV：SPI 从机模式使能位

0：SPI 为主机模式

1：SPI 为从机模式

S2CPOL：SPI 时钟极性控制

0：SCLK 空闲时为低电平，SCLK 的前时钟沿为上升沿，后时钟沿为下降沿

1：SCLK 空闲时为高电平，SCLK 的前时钟沿为下降沿，后时钟沿为上升沿

S2CPHA：SPI 时钟相位控制

0：数据 SS 管脚为低电平驱动第一位数据并在 SCLK 的后时钟沿改变数据，前时钟沿采样数据

1：数据在 SCLK 的前时钟沿驱动，后时钟沿采样

19.3.12 串口 2 同步模式控制寄存器 2（USART2CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART2CR2	7EFDC9H	S2IREN	S2IRLP	S2SCEN	S2NACK	S2HDSEL	S2PCEN	S2PS	S2PE

S2IREN：IrDA 模式使能位

0：禁止 IrDA 模式

1：使能 IrDA 模式

S2IRLP：IrDA 低电模式控制位

0：IrDA 为普通模式

1：IrDA 为低电模式

S2SCEN：SmartCard 模式使能位

0：禁止 SmartCard 模式

1：使能 SmartCard 模式

S2NACK：SmartCard 模式 NACK 输出使能位

0：禁止输出 NACK 信号

1：使能输出 NACK 信号

S2HDSEL：单线半双工模式使能位

0：禁止单线半双工模式

1：使能单线半双工模式

S2PCEN：硬件自动产生校验位控制使能

0：禁止硬件自动产生校验位（串口的校验位为 S2TB8 设置的值）

1：使能硬件自动产生校验位

S2PS：硬件校验位模式选择

0：硬件根据 S2BUF 的值自动产生偶校验位

1：硬件根据 S2BUF 的值自动产生奇校验位

S2PE：校验位错误标志（必须软件清零）

0：无检验错误

1：有校验错误（串口接收 DMA 过程中如果发生接收数据校验位错误，DMA 不会停止，但校验位错误标志会一直保持直到 DMA 完成）

19.3.13 串口 2 同步模式控制寄存器 3（USART2CR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART2CR3	7EFDCAH	S2IrDA_LPBAUD[7:0]							

S2IrDA_LPBAUD：IrDA 低电模式波特率控制寄存器

IrDA_LPBAUD[7:0]	IrDA 低电模式波特率
0	SYSclk/16/256
1	SYSclk/16/1
2	SYSclk/16/2
...	...
255	SYSclk/16/255

19.3.14 串口 2 同步模式控制寄存器 4（USART2CR4）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART2CR4	7EFDCBH	-	-	-	-	S2SCCKS[1:0]		S2SPICKS[1:0]	

S2SCCKS[1:0]：SmartCard 模式时钟选择

SCCKS[1:0]	SmartCard 模式时钟
00	SYSclk/64
01	SYSclk/32
10	SYSclk/16
11	SYSclk/8

S2SPICKS[1:0]：SPI 模式时钟选择

SPICKS[1:0]	SPI 模式时钟
00	SYSclk/4
01	SYSclk/8
10	SYSclk/16
11	SYSclk/2

19.3.15 串口 2 同步模式控制寄存器 5（USART2CR5）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART2CR5	7EFDCCH	S2BRKDET	S2HDRER	S2SLVEN	S2ASYNC	S2TXCF	S2SENDBK	S2HDRDET	S2SYNC

S2BRKDET：LIN 从模式间隔场（BREAK）检测标志

0：未检测到 LIN 间隔场

1：检测到 LIN 间隔场，需要软件清零

S2HDRER：LIN 从模式报文头（HEADER）错误

0：未检测到 LIN 报文头错误

1：检测到 LIN 报文头错误，需要软件清零

S2SLVEN：LIN 从机模式使能位

0：LIN 为主机模式

1：LIN 为从机模式

S2ASYNC：LIN 自动同步功能使能位

0：禁止自动同步

1：使能自动同步

【注意】：从机使能自动同步功能后，如果从机跟主机波特率存在小范围误差，从机收到主机同步信号后，会自动将从机波特率设置成与主机相同的波特率。

例如，主机波特率 9600：

从机使能自动同步时，如果从机波特率在 9100～10000 附近，收到主机同步信号后，从机会自动将本身波特率设置为 9600。误差自动同步范围 -5.2%～+4.16%左右。

S2TXCF：传输冲突标志位。单线半双工模式、LIN 模式和 SmartCard 模式有效

0：未检测到传输冲突（发送的数据与接收的数据相同）

1：检测到传输冲突（发送的数据与接收的数据不同）

S2SENDBK：发送间隔场。软件写 1 触发发送间隔场，发送完成后硬件自动清 0

S2HDRDET：LIN 从模式报文头（HEADER）检测标志

0：未检测到 LIN 报文头

1：检测到 LIN 报文头，需要软件清零

S2SYNC：LIN 从模式同步场检测标志。

正确分析到同步场后，硬件将 SYNC 标志为置 1。在接收标志符场时硬件会自动清零 SYNC

19.3.16 串口 2 同步模式保护时间寄存器（USART2GTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART2GTR	7EFDCDH								

USART2GTR 寄存器仅在 SmartCard 模式有效。该寄存器是根据波特率时钟数给出保护时间值。S2TI 标志在 SmartCard 发送的数据位等于 USART2GTR 寄存器所设置的保护时间值时被硬件置 1。注：USART2GTR 寄存器的值应大于 11

19.3.17 串口 2 同步模式波特率寄存器（USART2BR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART2BRH	7EFDCEH	USART2BR[15:8]							
USART2BRL	7EFDCFH	USART2BR[7:0]							

LIN 模式、IrDA 模式和 SmartCard 模式时的波特率计算公式

$$\text{同步波特率} = \frac{\text{SYSclk}}{16 * \text{USART2BR[15:0]}}$$

19.3.18 串口 2 接收超时控制寄存器（UR2TOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR2TOCR	7EFD74H	UR2ENTO	UR2ENTOI	UR2SCALE	-	-	-	-	-

UR2ENTO：串口 2 接收超时功能控制位

0：禁止串口 2 接收超时功能

1：使能串口 2 接收超时功能

UR2ENTOI：串口 2 接收超时中断控制位

0：禁止串口 2 接收超时中断

1：使能串口 2 接收超时中断

UR2SCALE：串口 2 超时计数时钟源选择

0：串口数据位率（波特率）

1：系统时钟

19.3.19 串口 2 超时状态寄存器（UR2TOSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR2TOSR	7EFD75H	UR2CTOIF	-	-	-	-	-	-	UR2TOIF

UR2CTOIF：写“1”清除串口超时中断标志位 TOIF。（只写）

UR2TOIF：串口 2 超时中断请求标志位。（只读）

当发生串口 2 超时时，TOIF 会被硬件置“1”。如果 ENTOI 为 1 时会产生串口中断，中断入口地址为原串口 2 的中断入口地址。标志位需要软件向 CTOIF 位写“1”清零

19.3.20 串口 2 超时长度控制寄存器（UR2TOTE/H/L）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR2TOTE	7EFD89H	UR2TM[23:16]							
UR2TOTH	7EFD76H	UR2TM[15:8]							
UR2TOTL	7EFD77H	UR2TM[7:0]							

UR2TM[23:0]：串口 2 超时时间控制位。

当串口 2 处于接收空闲状态时，内部超时计数器根据 SCALE 寄存器所选择的时钟源进行计数，当计数时间达到 UR2TM 所设置的超时时间时，便会产生超时中断。当串口 2 接收数据完成时，复位内部超时计数器，重新进行超时计数。

注：

1、如果需要使能接收超时中断功能，则 TM 不可设置为 0，且 UR2TOTL、UR2TOTH、UR2TOTE 寄存器的设置必须先设置 UR2TOTL 和 UR2TOTH，最后设置 UR2TOTE

2、必须使能串口 2 接收才有接收超时功能

3、接收超时功能必须在正确接收到一字节数据后才能触发

4、正在接收过程中，无接收超时功能

19.4 串口 3（同步/异步串口 USART3）

19.4.1 串口 3 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW2	BAH	EAXFR	-	I2C_S[1:0]		-	S4_S[0]	**S3_S[0]**	S2_S[0]
P_SWX1	7EFD69H	S4_S[1]	**S3_S[1]**	S2_S[1]	-	CAN2_S[2]	CAN_S[2]	-	SWP54P12

S3_S[1:0]：串口 3 功能脚选择位

S3_S[1:0]	RxD3	TxD3
00	P0.0	P0.1
01	P5.0	P5.1
10	P8.6	P8.7
11	PB.6	PB.7

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX3	7EFD6BH	S6SPI_S[1:0]		S5SPI_S[1:0]		S4SPI_S[1:0]		**S3SPI_S[1:0]**	

S3SPI_S[1:0]：USART3 的 SPI 功能脚选择位

S3SPI_S[1:0]	S3SS	S3MOSI	S3MISO	S3SCLK
00	P6.4	P6.5	P6.6	P6.7
01	P0.2	P0.3	P0.4	P0.5
10	PB.1	PB.2	PB.3	PB.4
11	P9.4	P9.5	P9.6	P9.7

19.4.2 串口 3 相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S3CON	串口 3 控制寄存器	ACH	S3SM0/FE	S3SM1	S3SM2	S3REN	S3TB8	S3RB8	S3TI	S3RI	0000,0000
S3BUF	串口 3 数据寄存器	ADH	S3BUF[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S3CFG	串口 3 配置寄存器	7EFDB8H	S3MOD	S3MOD0	S3M0x6	-	-	-	-	S3BRT	000x,xxx0
S3ADDR	串口 3 从机地址寄存器	7EFDB9H	S3ADDR[7:0]								0000,0000
S3ADEN	串口 3 从机地址屏蔽寄存器	7EFDBAH	S3ADEN[7:0]								0000,0000
USART3CR1	串口 3 控制寄存器 1	7EFDD0H	S3LINEN	S3DORD	S3CLKEN	S3SPMOD	S3SPIEN	S3SPSLV	S3CPOL	S3CPHA	0000,0000
USART3CR2	串口 3 控制寄存器 2	7EFDD1H	S3IREN	S3IRLP	S3SCEN	S3NACK	S3HDSEL	S3PCEN	S3PS	S3PE	0000,0000
USART3CR3	串口 3 控制寄存器 3	7EFDD2H	S3IrDA_LPBAUD[7:0]								0000,0111
USART3CR4	串口 3 控制寄存器 4	7EFDD3H	-	-	-	-	S3SCCKS[1:0]		S3SPICKS[1:0]		xxxx,0000
USART3CR5	串口 3 控制寄存器 5	7EFDD4H	S3BRKDET	S3HDRER	S3SLVEN	S3ASYNC	S3TXCF	S3SENDBK	S3HDRDET	S3SYNC	0000,0000
USART3GTR	串口 3 保护时间寄存器	7EFDD5H	USART3GTR[7:0]								0000,0000
USART3BRH	串口 3 波特率寄存器	7EFDD6H	USART3BR[15:8]								0000,0000
USART3BRL	串口 3 波特率寄存器	7EFDD7H	USART3BR[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
UR3TOCR	串口 3 接收超时控制寄存器	7EFD78H	UR3ENTO	UR3ENTOI	UR3SCALE	-	-	-	-	-	000x,xxxx
UR3TOSR	串口 3 接收超时状态寄存器	7EFD79H	-	-	-	-	-	-	-	UR3TOIF	xxxx,xxx0
UR3TOTH	串口 3 接收超时长度寄存器	7EFD7AH	UR3TM[15:8]								0000,0000
UR3TOTL	串口 3 接收超时长度寄存器	7EFD7BH	UR3TM[7:0]								0000,0000
UR3TOTE	串口 3 接收超时长度寄存器	7EFD8AH	UR3TM[23:16]								0000,0000

19.4.3 串口 3 控制寄存器（S3CON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S3CON	ACH	S3SM0/FE	S3SM1	S3SM2	S3REN	S3TB8	S3RB8	S3TI	S3RI

S3SM0/FE：当S3CFG寄存器中的S3MOD0位为1时，该位为帧错误检测标志位。当串口3在接收过程中检测到一个无效停止位时，通过串口3接收器将该位置1，必须由软件清零。当S3CFG寄存器中的S3MOD0位为0时，该位和S3SM1一起指定串口3的通信工作模式，如下表所示：

S3SM0	S3SM1	串口3工作模式	功能说明
0	0	模式0	同步移位串行方式
0	1	模式1	可变波特率8位数据方式
1	0	模式2	固定波特率9位数据方式
1	1	模式3	可变波特率9位数据方式

当使能 FE 功能后，串口的工作模式只能使用 SM1 进行设置，且只能设置为模式 0 或者模式 1。FE 当作帧错误标志位时，此标志位必须软件写 0 清零。

S3SM2：允许模式 2 或模式 3 多机通信控制位。当串口 3 使用模式 2 或模式 3 时，如果 S3SM2 位为 1 且 S3REN 位为 1，则接收机处于地址帧筛选状态。此时可以利用接收到的第 9 位（即 S3RB8）来筛选地址帧，若 S3RB8=1，说明该帧是地址帧，地址信息可以进入 S3BUF，并使 S3RI 为 1，进而在中断服务程序中再进行地址号比较；若 S3RB8=0，说明该帧不是地址帧，应丢掉且保持 S3RI=0。在模式 2 或模式 3 中，如果 S3SM2 位为 0 且 S3REN 位为 1，接收收机处于地址帧筛选被禁止状态，不论收到的 S3RB8 为 0 或 1，均可使接收到的信息进入 S3BUF，并使 S3RI=1，此时 S3RB8 通常为校验位。模式 1 和模式 0 为非多机通信方式，在这两种方式时，S3SM2 应设置为 0。

S3REN：允许/禁止串口接收控制位

0：禁止串口接收数据

1：允许串口接收数据

S3TB8：当串口 3 使用模式 2 或模式 3 时，S3TB8 为要发送的第 9 位数据，按需要由软件置位或清 0。在模式 0 和模式 1 中，该位不用。

S3RB8：当串口 3 使用模式 2 或模式 3 时，S3RB8 为接收到的第 9 位数据，一般用作校验位或者地址帧/数据帧标志位。在模式 0 和模式 1 中，该位不用。

S3TI：串口 3 发送中断请求标志位。在模式 0 中，当串口发送数据第 8 位结束时，由硬件自动将 S3TI 置 1，向主机请求中断，响应中断后 S3TI 必须用软件清零。在其他模式中，则在停止位开始发送时由硬件自动将 S3TI 置 1，向 CPU 发请求中断，响应中断后 S3TI 必须用软件清零。

S3RI：串口 3 接收中断请求标志位。在模式 0 中，当串口接收第 8 位数据结束时，由硬件自动将 S3RI 置 1，向主机请求中断，响应中断后 S3RI 必须用软件清零。在其他模式中，串行接收到停止位的中间时刻由硬件自动将 S3RI 置 1，向 CPU 发中断申请，响应中断后 S3RI 必须由软件清零。

19.4.4 串口 3 数据寄存器（S3BUF）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S3BUF	ADH								

S3BUF：串口 1 数据接收/发送缓冲区。S3BUF 实际是 2 个缓冲器，读缓冲器和写缓冲器，两个操作分别对应两个不同的寄存器，1 个是只写寄存器（写缓冲器），1 个是只读寄存器（读缓冲器）。对 S3BUF 进行读操作，实际是读取串口接收缓冲区，对 S3BUF 进行写操作则是触发串口开始发送数据。

19.4.5 串口 3 配置寄存器（S3CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S3CFG	7EFDB8H	S3MOD	S3MOD0	S3M0x6					S3BRT

S3MOD：串口 3 波特率控制位

0：串口 3 之模式 0/1/2/3 的波特率都不加倍

1：串口 3 之模式 2 的波特率加倍

S3MOD0：帧错误检测控制位

0：无帧错检测功能

1：使能帧错误检测功能。此时 S3CON 的 S3SM0/FE 为 FE 功能，即为帧错误检测标志位。

S3M0x6：串口 3 模式 0 的通讯速度控制

0：串口 3 模式 0 的波特率不加倍，固定为 Fosc/12

1：串口 3 模式 0 的波特率 6 倍速，即固定为 Fosc/12*6 ＝ Fosc/2

S3BRT：串口 3 波特率发生器选择位

0：选择定时器 2 作为波特率发生器**（默认值）**

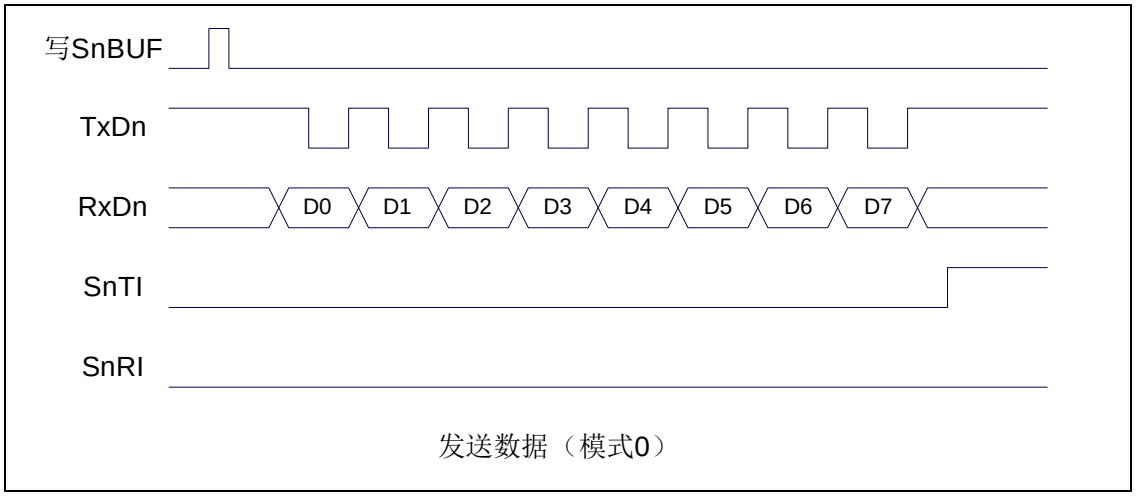
1：选择定时器 3 作为波特率发生器

19.4.6 串口 3 模式 0，模式 0 波特率计算公式

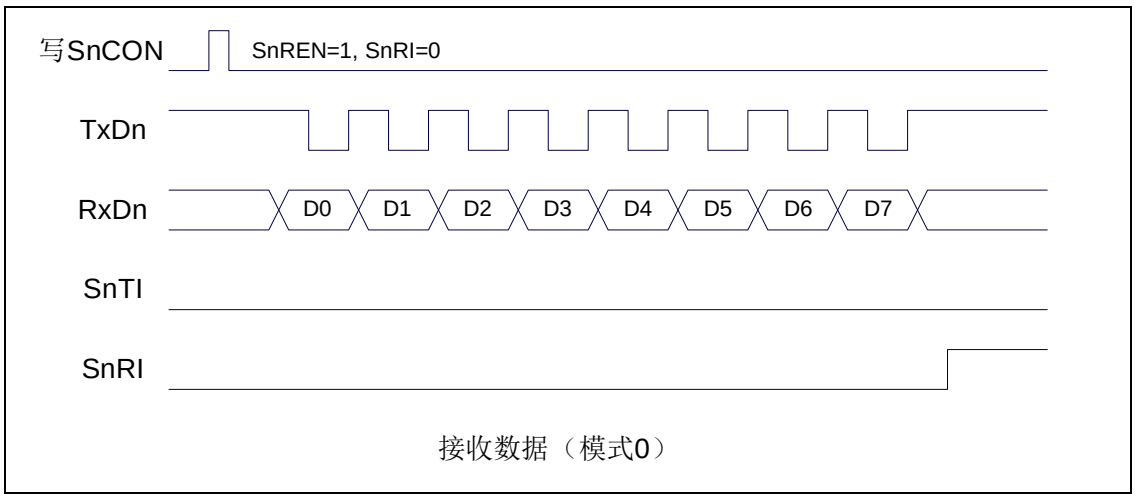
当串口 3 选择工作模式为模式 0 时，串行通信接口工作在同步移位寄存器模式，当串口 3 模式 0 的通信速度设置位 S3M0x6 为 0 时，其波特率固定为系统时钟时钟的 12 分频（SYSclk/12）；当设置 S3M0x6 为 1 时，其波特率固定为系统时钟频率的 2 分频（SYSclk/2）。RxD3 为串行通讯的数据口，TxD3 为同步移位脉冲输出脚，发送、接收的是 8 位数据，低位在先。

模式 0 的发送过程：当主机执行将数据写入发送缓冲器 S3BUF 指令时启动发送，串口 3 即将 8 位数据以 SYSclk/12 或 SYSclk/2（由 S3M0x6 确定是 12 分频还是 2 分频）的波特率从 RxD3 管脚输出(从低位到高位)，发送完中断标志 S3TI 置 1，TxD3 管脚输出同步移位脉冲信号。当写信号有效后，相隔一个时钟，发送控制端 SEND 有效(高电平)，允许 RxD3 发送数据，同时允许 TxD3 输出同步移位脉冲。一帧(8 位)数据发送完毕时，各控制端均恢复原状态，只有 S3TI 保持高电平，呈中断申请状态。在再次发送数据前，必须用软件将 S3TI 清 0。

模式 0 的接收过程：首先将接收中断请求标志 S3RI 清零并置位允许接收控制位 S3REN 时启动模式 0 接收过程。启动接收过程后，RxD3 为串行数据输入端，TxD3 为同步脉冲输出端。串行接收的波特率为 SYSclk/12 或 SYSclk/2（由 S3M0x6 确定是 12 分频还是 2 分频）。当接收完成一帧数据（8 位）后，控制信号复位，中断标志 S3RI 被置 1，呈中断申请状态。当再次接收时，必须通过软件将 S3RI 清 0



发送数据（模式0）



接收数据（模式0）

工作于模式 0 时，必须清 0 多机通信控制位 S3SM2，使之不影响 S3TB8 位和 S3RB8 位。由于波特率固定为 SYSclk/12 或 SYSclk/2，无需定时器提供，直接由单片机的时钟作为同步移位脉冲。

串口 3 模式 0 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

S3M0x6	波特率计算公式
0	波特率 $= \frac{\text{SYSclk}}{12}$
1	波特率 $= \frac{\text{SYSclk}}{2}$

19.4.7 串口 3 模式 1，模式 1 波特率计算公式

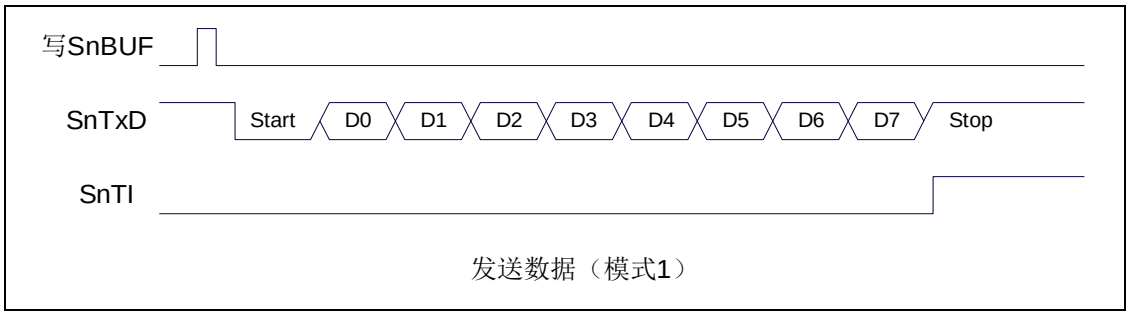
当软件设置 S3CON 的 S3SM0、S3SM1 为“01”时，串口 3 则以模式 1 进行工作。此模式为 8 位 UART 格式，一帧信息为 10 位：1 位起始位，8 位数据位（低位在先）和 1 位停止位。波特率可变，即可根据需要进行设置波特率。TxD3 为数据发送口，RxD3 为数据接收口，串口 3 全双工接受/发送。

模式 1 的发送过程：串行通信模式发送时，数据由串行发送端 TxD3 输出。当主机执行一条写 S3BUF 的指令就启动串行通信的发送，写“S3BUF”信号还把“1”装入发送移位寄存器的第 9 位，并通知 TX 控制单元开始发送。移位寄存器将数据不断右移送 TxD3 端口发送，在数据的左边不断移入“0”作补充。当数据的最高位移到移位寄存器的输出位置，紧跟其后的是第 9 位“1”，在它的左边各位全为“0”，这个状态条件，使 TX 控制单元作最后一次移位输出，然后使允许发送信号“SEND”失效，完成一帧信息的发送，并置位中断请求位 S3TI，即 S3TI=1，向主机请求中断处理。

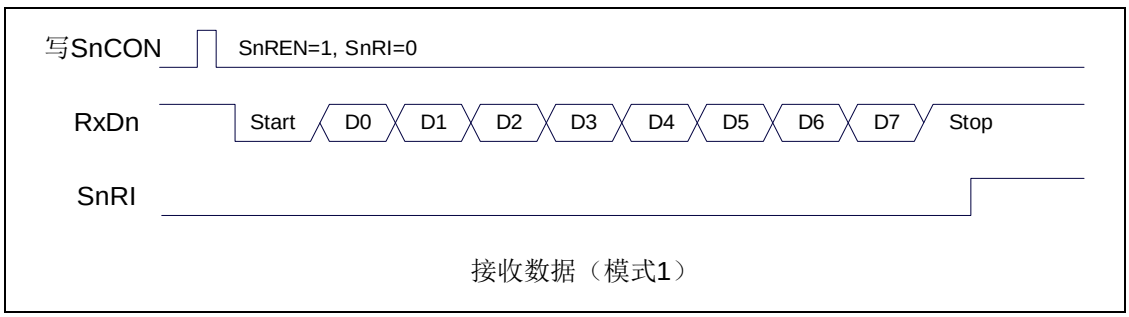
模式 1 的接收过程：当软件置位接收允许标志位 S3REN，即 S3REN=1 时，接收器便对 RxD3 端口的信号进行检测，当检测到 RxD3 端口发送从“1”→“0”的下降沿跳变时就启动接收器准备接收数据，并立即复位波特率发生器的接收计数器，将 1FFH 装入移位寄存器。接收的数据从接收移位寄存器的右边移入，已装入的 1FFH 向左边移出，当起始位"0"移到移位寄存器的最左边时，使 RX 控制器作最后一次移位，完成一帧的接收。若同时满足以下两个条件：

- S3RI=0；
- S3SM2=0 或接收到的停止位为 1。

则接收到的数据有效，实现装载入 S3BUF，停止位进入 S3RB8，S3RI 标志位被置 1，向主机请求中断，若上述两条件不能同时满足，则接收到的数据作废并丢失，无论条件满足与否，接收器重又检测 RxD3 端口上的"1"→"0"的跳变，继续下一帧的接收。接收有效，在响应中断后，S3RI 标志位必须由软件清 0。通常情况下，串行通信工作于模式 1 时，S3SM2 设置为"0"。



发送数据（模式1）



接收数据（模式1）

串口 3 的波特率是可变的，其波特率固定由定时器 2 或定时器 3 产生。当定时器采用 1T 模式时（12 倍速），相应的波特率的速度也会相应提高 12 倍。

串口 3 模式 1 的波特率计算公式如下表所示：（SYSclk 为系统工作频率）

选择定时器	定时器速度	重装值计算公式	波特率
定时器2 或 定时器3	1T	定时器重载值 = 65536 - $\frac{SYSclk}{4\times 波特率}$	波特率 = $\frac{SYSclk}{4\times(65536 - 定时器重装数)}$
	12T	定时器重载值 = 65536 - $\frac{SYSclk}{12\times 4\times 波特率}$	波特率 = $\frac{SYSclk}{12\times 4\times(65536 - 定时器重装数)}$

下面为常用频率与常用波特率所对应定时器的重载值

频率 (MHz)	波特率	定时器	
		1T 模式	12T 模式
11.0592	115200	FFE8H	FFFEH
	57600	FFD0H	FFFCH
	38400	FFB8H	FFFAH
	19200	FF70H	FFF4H
	9600	FEE0H	FFE8H
18.432	115200	FFD8H	-
	57600	FFB0H	-
	38400	FF88H	FFF6H
	19200	FF10H	FFECH
	9600	FE20H	FFD8H
22.1184	115200	FFD0H	FFFCH
	57600	FFA0H	FFF8H
	38400	FF70H	FFF4H
	19200	FEE0H	FFE8H
	9600	FDC0H	FFD0H

19.4.8 串口 3 模式 2，模式 2 波特率计算公式

当 S3SM0、S3SM1 两位为 10 时，串口 3 工作在模式 2。串口 3 工作模式 2 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S3CON 中的 S3TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S3TB8（S3TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S3CON 的 S3RB8。TxD3 为发送端口，RxD3 为接收端口，以全双工模式进行接收/发送。

模式 2 的波特率固定为系统时钟的 64 分频或 32 分频（取决于 S3CFG 中 S3MOD 的值）

串口 3 模式 2 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

SMOD	波特率计算公式
0	$\text{波特率} = \frac{\text{SYSclk}}{64}$
1	$\text{波特率} = \frac{\text{SYSclk}}{32}$

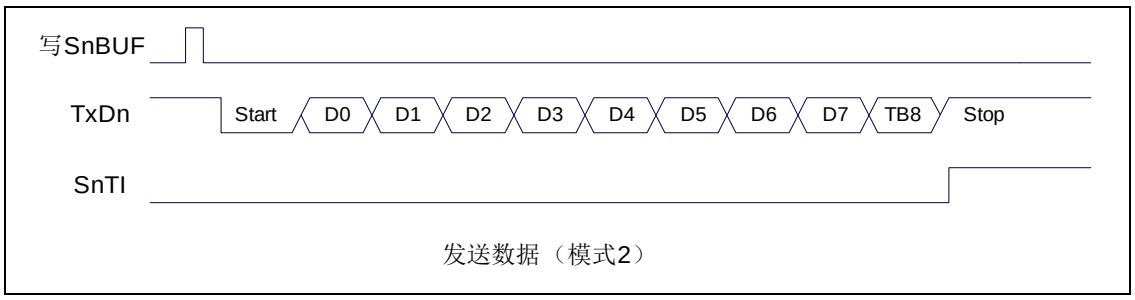
模式 2 和模式 1 相比，除波特率发生源略有不同，发送时由 S3TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收/发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

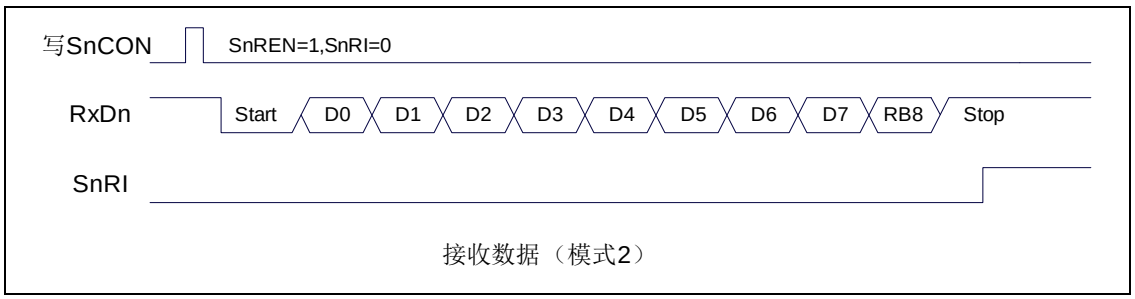
- S3RI=0
- S3SM2=0 或者 S3SM2=1 且接收到的第 9 数据位 S3RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S3BUF 和 S3RB8 中，S3RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S3RI。无论上述条件满足与否，接收器又重新开始检测 RxD3 输入端口的跳变信息，接收下一帧的输入信息。在模式 2 中，接收到的停止位与 S3BUF、S3RB8 和 S3RI 无关。

通过软件对 S3CON 中的 S3SM2、S3TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式2）



接收数据（模式2）

19.4.9 串口 3 模式 3，模式 3 波特率计算公式

当 S3SM0、S3SM1 两位为 11 时，串口 3 工作在模式 3。串行通信模式 3 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S3CON 中的 S3TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S3TB8（S3TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S3CON 的 S3RB8。TxD3 为发送端口，RxD3 为接收端口，以全双工模式进行接收/发送。

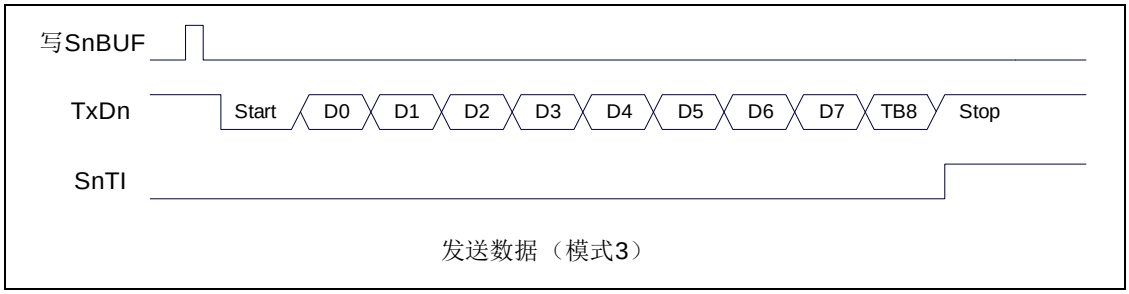
模式 3 和模式 1 相比，除发送时由 S3TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收‘发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

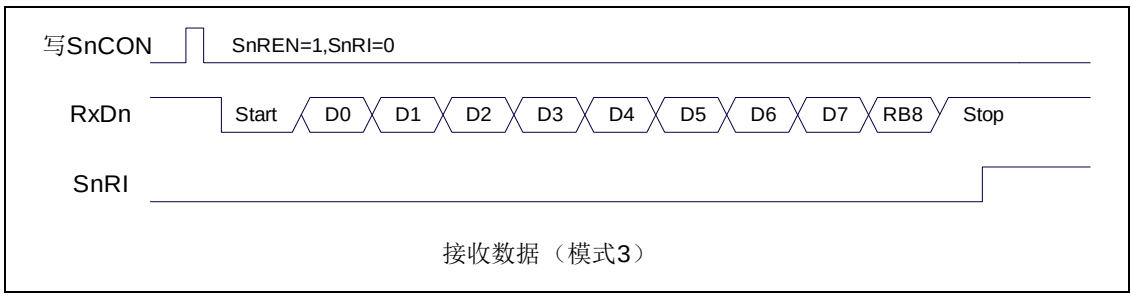
- S3RI=0
- S3SM2=0 或者 S3SM2=1 且接收到的第 9 数据位 S3RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S3BUF 和 S3RB8 中，S3RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S3RI。无论上述条件满足与否，接收器又重新开始检测 RxD3 输入端口的跳变信息，接收下一帧的输入信息。在模式 3 中，接收到的停止位与 S3BUF、S3RB8 和 S3RI 无关。

通过软件对 S3CON 中的 S3SM2、S3TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式3）



接收数据（模式3）

串口 3 模式 3 的波特率计算公式与模式 1 是完全相同的。请参考模式 1 的波特率计算公式。

19.4.10 串口 3 自动地址识别，从机地址控制寄存器（S3ADDR，S3ADEN）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S3ADDR	7EFDB9H								
S3ADEN	7EFDBAH								

S3ADDR：从机地址寄存器

S3ADEN：从机地址屏蔽位寄存器

自动地址识别功能典型应用在多机通讯领域，其主要原理是从机系统通过硬件比较功能来识别来自于主机串口 3 数据流中的地址信息，通过寄存器 S3ADDR 和 S3ADEN 设置的本机的从机地址，硬件自动对从机地址进行过滤，当来自于主机的从机地址信息与本机所设置的从机地址相匹配时，硬件产生串口 3 中断；否则硬件自动丢弃串口 3 数据，而不产生中断。当众多处于空闲模式的从机链接在一起时，只有从机地址相匹配的从机才会从空闲模式唤醒，从而可以大大降低从机 MCU 的功耗，即使从机处于正常工作状态也可避免不停地进入串口 3 中断而降低系统执行效率。

要使用串口 3 的自动地址识别功能，首先需要将参与通讯的 MCU 的串口 3 通讯模式设置为模式 1，并开启从机的 S3CON 的 S3SM2 位。对于串口 3 模式 1 的 9 位数据位中，第 9 位数据（存放在 S3RB8 中）为地址/数据的标志位，当第 9 位数据为 1 时，表示前面的 8 位数据（存放在 S3BUF 中）为地址信息。当 S3SM2 被设置为 1 时，从机 MCU 会自动过滤掉非地址数据（第 9 位为 0 的数据），而对 S3BUF 中的地址数据（第 9 位为 1 的数据）自动与 S3ADDR 和 S3ADEN 所设置的本机地址进行比较，若地址相匹配，则会将 S3RI 置“1”，并产生中断，否则不予处理本次接收的串口 3 数据。

从机地址的设置是通过 S3ADDR 和 S3ADEN 两个寄存器进行设置的。S3ADDR 为从机地址寄存器，里面存放本机的从机地址。S3ADEN 为从机地址屏蔽位寄存器，用于设置地址信息中的忽略位，设置方法如下：

例如

S3ADDR ＝ 11001010

S3ADEN ＝ 10000001

则匹配地址为 1xxxxxx0

即，只要主机送出的地址数据中的 bit0 为 0 且 bit7 为 1 就可以和本机地址相匹配

再例如

S3ADDR ＝ 11001010

S3ADEN ＝ 00001111

则匹配地址为 xxxx1010

即，只要主机送出的地址数据中的低 4 位为 1010 就可以和本机地址相匹配，而高 4 为被忽略，可以为任意值。

主机可以使用广播地址（FFH）同时选中所有的从机来进行通讯。

19.4.11 串口 3 同步模式控制寄存器 1（USART3CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART3CR1	7EFDD0H	S3LINEN	S3DORD	S3CLKEN	S3SPMOD	S3SPIEN	S3SPSLV	S3CPOL	S3CPHA

S3LINEN：LIN 模式使能位

0：禁止 LIN 模式

1：使能 LIN 模式

S3DORD：SPI 模式数据位发送/接收的顺序

0：先发送/接收数据的高位（MSB）

1：先发送/接收数据的低位（LSB）

S3CLKEN：SmartCard 模式时钟输出控制位

0：禁止时钟输出

1：使能时钟输出

S3SPMOD：SPI 模式使能位

0：禁止 SPI 模式

1：使能 SPI 模式

S3SPIEN：SPI 使能位

0：禁止 SPI 功能

1：使能 SPI 功能

当 USART3 需要工作在 SPI 模式时，必须先将 SPI 相关的模式寄存器、速度寄存器、时钟极性寄存器设置完成后，最后再设置 SPIEN 寄存器。

S3SPSLV：SPI 从机模式使能位

0：SPI 为主机模式

1：SPI 为从机模式

S3CPOL：SPI 时钟极性控制

0：SCLK 空闲时为低电平，SCLK 的前时钟沿为上升沿，后时钟沿为下降沿

1：SCLK 空闲时为高电平，SCLK 的前时钟沿为下降沿，后时钟沿为上升沿

S3CPHA：SPI 时钟相位控制

0：数据 SS 管脚为低电平驱动第一位数据并在 SCLK 的后时钟沿改变数据，前时钟沿采样数据

1：数据在 SCLK 的前时钟沿驱动，后时钟沿采样

19.4.12 串口 3 同步模式控制寄存器 2（USART3CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART3CR2	7EFDD1H	S3IREN	S3IRLP	S3SCEN	S3NACK	S3HDSEL	S3PCEN	S3PS	S3PE

S3IREN：IrDA 模式使能位

0：禁止 IrDA 模式

1：使能 IrDA 模式

S3IRLP：IrDA 低电模式控制位

0：IrDA 为普通模式

1：IrDA 为低电模式

S3SCEN：SmartCard 模式使能位

0：禁止 SmartCard 模式

1：使能 SmartCard 模式

S3NACK：SmartCard 模式 NACK 输出使能位

0：禁止输出 NACK 信号

1：使能输出 NACK 信号

S3HDSEL：单线半双工模式使能位

0：禁止单线半双工模式

1：使能单线半双工模式

S3PCEN：硬件自动产生校验位控制使能

0：禁止硬件自动产生校验位（串口的校验位为 S3TB8 设置的值）

1：使能硬件自动产生校验位

S3PS：硬件校验位模式选择

0：硬件根据 S3BUF 的值自动产生偶校验位

1：硬件根据 S3BUF 的值自动产生奇校验位

S3PE：校验位错误标志（必须软件清零）

0：无检验错误

1：有校验错误（串口接收 DMA 过程中如果发生接收数据校验位错误，DMA 不会停止，但校验位错误标志会一直保持直到 DMA 完成）

19.4.13 串口 3 同步模式控制寄存器 3（USART3CR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART3CR3	7EFDD2H	S3IrDA_LPBAUD[7:0]							

S3IrDA_LPBAUD：IrDA 低电模式波特率控制寄存器

IrDA_LPBAUD[7:0]	IrDA 低电模式波特率
0	SYSclk/16/256
1	SYSclk/16/1
2	SYSclk/16/2
...	...
255	SYSclk/16/255

19.4.14 串口 3 同步模式控制寄存器 4（USART3CR4）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART3CR4	7EFDD3H	-	-	-	-	S3SCCKS[1:0]		S3SPICKS[1:0]	

S3SCCKS[1:0]：SmartCard 模式时钟选择

SCCKS[1:0]	SmartCard 模式时钟
00	SYSclk/64
01	SYSclk/32
10	SYSclk/16
11	SYSclk/8

S3SPICKS[1:0]：SPI 模式时钟选择

SPICKS[1:0]	SPI 模式时钟
00	SYSclk/4
01	SYSclk/8
10	SYSclk/16
11	SYSclk/2

19.4.15 串口 3 同步模式控制寄存器 5（USART3CR5）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART3CR5	7EFDD4H	S3BRKDET	S3HDRER	S3SLVEN	S3ASYNC	S3TXCF	S3SENDBK	S3HDRDET	S3SYNC

S3BRKDET：LIN 从模式间隔场（BREAK）检测标志

0：未检测到 LIN 间隔场

1：检测到 LIN 间隔场，需要软件清零

S3HDRER：LIN 从模式报文头（HEADER）错误

0：未检测到 LIN 报文头错误

1：检测到 LIN 报文头错误，需要软件清零

S3SLVEN：LIN 从机模式使能位

0：LIN 为主机模式

1：LIN 为从机模式

S3ASYNC：LIN 自动同步功能使能位

0：禁止自动同步

1：使能自动同步

【注意】：从机使能自动同步功能后，如果从机跟主机波特率存在小范围误差，从机收到主机同步信号后，会自动将从机波特率设置成与主机相同的波特率。

例如，主机波特率 9600：

从机使能自动同步时，如果从机波特率在 9100～10000 附近，收到主机同步信号后，从机会自动将本身波特率设置为 9600。误差自动同步范围 -5.2%～+4.16%左右。

S3TXCF：传输冲突标志位。单线半双工模式、LIN 模式和 SmartCard 模式有效

0：未检测到传输冲突（发送的数据与接收的数据相同）

1：检测到传输冲突（发送的数据与接收的数据不同）

S3SENDBK：发送间隔场。软件写 1 触发发送间隔场，发送完成后硬件自动清 0

S3HDRDET：LIN 从模式报文头（HEADER）检测标志

0：未检测到 LIN 报文头

1：检测到 LIN 报文头，需要软件清零

S3SYNC：LIN 从模式同步场检测标志。

正确分析到同步场后，硬件将 SYNC 标志为置 1。在接收标志符场时硬件会自动清零 SYNC

19.4.16 串口 3 同步模式保护时间寄存器（USART3GTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART3GTR	7EFDD5H								

USART3GTR 寄存器仅在 SmartCard 模式有效。该寄存器是根据波特率时钟数给出保护时间值。S3TI 标志在 SmartCard 发送的数据位等于 USART3GTR 寄存器所设置的保护时间值时被硬件置 1。注：USART3GTR 寄存器的值应大于 11

19.4.17 串口 3 同步模式波特率寄存器（USART3BR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART3BRH	7EFDD6H	USART3BR[15:8]							
USART3BRL	7EFDD7H	USART3BR[7:0]							

LIN 模式、IrDA 模式和 SmartCard 模式时的波特率计算公式

$$\text{同步波特率} = \frac{\text{SYSclk}}{16 * \text{USART3BR[15:0]}}$$

19.4.18 串口 3 接收超时控制寄存器（UR3TOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR3TOCR	7EFD78H	UR3ENTO	UR3ENTOI	UR3SCALE	-	-	-	-	-

UR3ENTO：串口 3 接收超时功能控制位

0：禁止串口 3 接收超时功能

1：使能串口 3 接收超时功能

UR3ENTOI：串口 3 接收超时中断控制位

0：禁止串口 3 接收超时中断

1：使能串口 3 接收超时中断

UR3SCALE：串口 3 超时计数时钟源选择

0：串口数据位率（波特率）

1：系统时钟

19.4.19 串口 3 超时状态寄存器（UR3TOSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR3TOSR	7EFD79H	UR3CTOIF	-	-	-	-	-	-	UR3TOIF

UR3CTOIF：写“1”清除串口超时中断标志位 TOIF。（只写）

UR3TOIF：串口 3 超时中断请求标志位。（只读）

当发生串口 3 超时时，TOIF 会被硬件置“1”。如果 ENTOI 为 1 时会产生串口中断，中断入口地址为原串口 3 的中断入口地址。标志位需要软件向 CTOIF 位写“1”清零

19.4.20 串口 3 超时长度控制寄存器（UR3TOTE/H/L）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR3TOTE	7EFD8AH	UR3TM[23:16]							
UR3TOTH	7EFD7AH	UR3TM[15:8]							
UR3TOTL	7EFD7BH	UR3TM[7:0]							

UR3TM[23:0]：串口 3 超时时间控制位。

当串口 3 处于接收空闲状态时，内部超时计数器根据 SCALE 寄存器所选择的时钟源进行计数，当计数时间达到 UR3TM 所设置的超时时间时，便会产生超时中断。当串口 3 接收数据完成时，复位内部超时计数器，重新进行超时计数。

注：

1、如果需要使能接收超时中断功能，则 TM 不可设置为 0，且 UR3TOTL、UR3TOTH、UR3TOTE 寄存器的设置必须先设置 UR3TOTL 和 UR3TOTH，最后设置 UR3TOTE

2、必须使能串口 3 接收才有接收超时功能

3、接收超时功能必须在正确接收到一字节数据后才能触发

4、正在接收过程中，无接收超时功能

19.5 串口 4（同步/异步串口 USART4）

19.5.1 串口 4 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW2	BAH	EAXFR	–	I2C_S[1:0]		-	**S4_S[0]**	S3_S[0]	S2_S[0]
P_SWX1	7EFD69H	**S4_S[1]**	S3_S[1]	S2_S[1]	-	CAN2_S[2]	CAN_S[2]	-	SWP54P12

S4_S[1:0]：串口 4 功能脚选择位

S4_S[1:0]	RxD4	TxD4
00	P0.2	P0.3
01	P5.2	P5.3
10	PA.2	PA.3
11	PC.0	PC.1

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX3	7EFD6BH	S6SPI_S[1:0]		S5SPI_S[1:0]		**S4SPI_S[1:0]**		S3SPI_S[1:0]	

S4SPI_S[1:0]：USART4 的 SPI 功能脚选择位

S4SPI_S[1:0]	S4SS	S4MOSI	S4MISO	S4SCLK
00	P6.4	P6.5	P6.6	P6.7
01	P0.2	P0.3	P0.4	P0.5
10	PB.1	PB.2	PB.3	PB.4
11	P7.0	P7.1	P7.2	P7.3

19.5.2 串口 4 相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S4CON	串口 4 控制寄存器	FDH	S4SM0/FE	S4SM1	S4SM2	S4REN	S4TB8	S4RB8	S4TI	S4RI	0000,0000
S4BUF	串口 4 数据寄存器	FEH	S4BUF[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S4CFG	串口 4 配置寄存器	7EFDBCH	S4MOD	S4MOD0	S4M0x6	-	-	-	-	S4BRT	000x,xxx0
S4ADDR	串口 4 从机地址寄存器	7EFDBDH	S4ADDR[7:0]								0000,0000
S4ADEN	串口 4 从机地址屏蔽寄存器	7EFDBEH	S4ADEN[7:0]								0000,0000
USART4CR1	串口 4 控制寄存器 1	7EFDD8H	S4LINEN	S4DORD	S4CLKEN	S4SPMOD	S4SPIEN	S4SPSLV	S4CPOL	S4CPHA	0000,0000
USART4CR2	串口 4 控制寄存器 2	7EFDD9H	S4IREN	S4IRLP	S4SCEN	S4NACK	S4HDSEL	S4PCEN	S4PS	S4PE	0000,0000
USART4CR3	串口 4 控制寄存器 3	7EFDDAH	S4IrDA_LPBAUD[7:0]								0000,0111
USART4CR4	串口 4 控制寄存器 4	7EFDDBH	-	-	-	-	S4SCCKS[1:0]		S4SPICKS[1:0]		xxxx,0000
USART4CR5	串口 4 控制寄存器 5	7EFDDCH	S4BRKDET	S4HDRER	S4SLVEN	S4ASYNC	S4TXCF	S4SENDBK	S4HDRDET	S4SYNC	0000,0000
USART4GTR	串口 4 保护时间寄存器	7EFDDDH	USART4GTR[7:0]								0000,0000
USART4BRH	串口 4 波特率寄存器	7EFDDEH	USART4BR[15:8]								0000,0000
USART4BRL	串口 4 波特率寄存器	7EFDDFH	USART4BR[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
UR4TOCR	串口 4 接收超时控制寄存器	7EFD7CH	UR4ENTO	UR4ENTOI	UR4SCALE	-	-	-	-	-	000x,xxxx
UR4TOSR	串口 4 接收超时状态寄存器	7EFD7DH	-	-	-	-	-	-	-	UR4TOIF	xxxx,xxx0
UR4TOTH	串口 4 接收超时长度寄存器	7EFD7EH	UR4TM[15:8]								0000,0000
UR4TOTL	串口 4 接收超时长度寄存器	7EFD7FH	UR4TM[7:0]								0000,0000
UR4TOTE	串口 4 接收超时长度寄存器	7EFD8BH	UR4TM[23:16]								0000,0000

19.5.3 串口 4 控制寄存器（S4CON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S4CON	FDH	S4SM0/FE	S4SM1	S4SM2	S4REN	S4TB8	S4RB8	S4TI	S4RI

S4SM0/FE：当S4CFG寄存器中的S4MOD0位为1时，该位为帧错误检测标志位。当串口4在接收过程中检测到一个无效停止位时，通过串口4接收器将该位置1，必须由软件清零。当S4CFG寄存器中的S4MOD0位为0时，该位和S4SM1一起指定串口4的通信工作模式，如下表所示：

S4SM0	S4SM1	串口4工作模式	功能说明
0	0	模式0	同步移位串行方式
0	1	模式1	可变波特率8位数据方式
1	0	模式2	固定波特率9位数据方式
1	1	模式3	可变波特率9位数据方式

当使能 FE 功能后，串口的工作模式只能使用 SM1 进行设置，且只能设置为模式 0 或者模式 1。FE 当作帧错误标志位时，此标志位必须软件写 0 清零。

S4SM2：允许模式 2 或模式 3 多机通信控制位。当串口 4 使用模式 2 或模式 3 时，如果 S4SM2 位为 1 且 S4REN 位为 1，则接收机处于地址帧筛选状态。此时可以利用接收到的第 9 位（即 S4RB8）来筛选地址帧，若 S4RB8=1，说明该帧是地址帧，地址信息可以进入 S4BUF，并使 S4RI 为 1，进而在中断服务程序中再进行地址号比较；若 S4RB8=0，说明该帧不是地址帧，应丢掉且保持 S4RI=0。在模式 2 或模式 3 中，如果 S4SM2 位为 0 且 S4REN 位为 1，接收收机处于地址帧筛选被禁止状态，不论收到的 S4RB8 为 0 或 1，均可使接收到的信息进入 S4BUF，并使 S4RI=1，此时 S4RB8 通常为校验位。模式 1 和模式 0 为非多机通信方式，在这两种方式时，S4SM2 应设置为 0。

S4REN：允许/禁止串口接收控制位

0：禁止串口接收数据

1：允许串口接收数据

S4TB8：当串口 4 使用模式 2 或模式 3 时，S4TB8 为要发送的第 9 位数据，按需要由软件置位或清 0。在模式 0 和模式 1 中，该位不用。

S4RB8：当串口 4 使用模式 2 或模式 3 时，S4RB8 为接收到的第 9 位数据，一般用作校验位或者地址帧/数据帧标志位。在模式 0 和模式 1 中，该位不用。

S4TI：串口 4 发送中断请求标志位。在模式 0 中，当串口发送数据第 8 位结束时，由硬件自动将 S4TI 置 1，向主机请求中断，响应中断后 S4TI 必须用软件清零。在其他模式中，则在停止位开始发送时由硬件自动将 S4TI 置 1，向 CPU 发请求中断，响应中断后 S4TI 必须用软件清零。

S4RI：串口 4 接收中断请求标志位。在模式 0 中，当串口接收第 8 位数据结束时，由硬件自动将 S4RI 置 1，向主机请求中断，响应中断后 S4RI 必须用软件清零。在其他模式中，串行接收到停止位的中间时刻由硬件自动将 S4RI 置 1，向 CPU 发中断申请，响应中断后 S4RI 必须由软件清零。

19.5.4 串口 4 数据寄存器（S4BUF）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S4BUF	FEH								

S4BUF：串口 1 数据接收/发送缓冲区。S4BUF 实际是 2 个缓冲器，读缓冲器和写缓冲器，两个操作分别对应两个不同的寄存器，1 个是只写寄存器（写缓冲器），1 个是只读寄存器（读缓冲器）。对 S4BUF 进行读操作，实际是读取串口接收缓冲区，对 S4BUF 进行写操作则是触发串口开始发送数据。

19.5.5 串口 4 配置寄存器（S4CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S4CFG	7EFDBCH	S4MOD	S4MOD0	S4M0x6					S4BRT

S4MOD：串口 4 波特率控制位

0：串口 4 之模式 0/1/2/3 的波特率都不加倍

1：串口 4 之模式 2 的波特率加倍

S4MOD0：帧错误检测控制位

0：无帧错检测功能

1：使能帧错误检测功能。此时 S4CON 的 S4SM0/FE 为 FE 功能，即为帧错误检测标志位。

S4M0x6：串口 4 模式 0 的通讯速度控制

0：串口 4 模式 0 的波特率不加倍，固定为 Fosc/12

1：串口 4 模式 0 的波特率 6 倍速，即固定为 Fosc/12*6 ＝ Fosc/2

S4BRT：串口 4 波特率发生器选择位

0：选择定时器 2 作为波特率发生器**（默认值）**

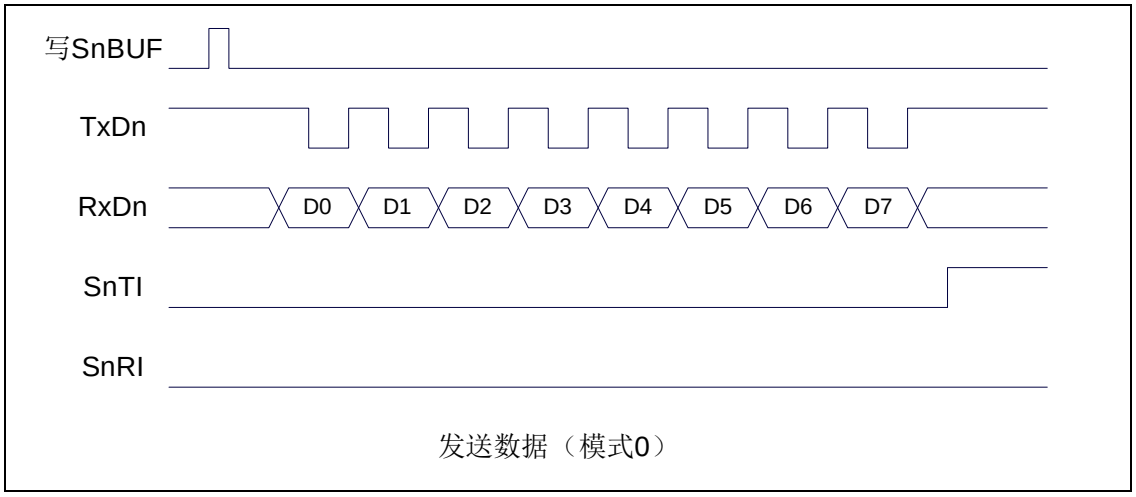
1：选择定时器 4 作为波特率发生器

19.5.6 串口 4 模式 0，模式 0 波特率计算公式

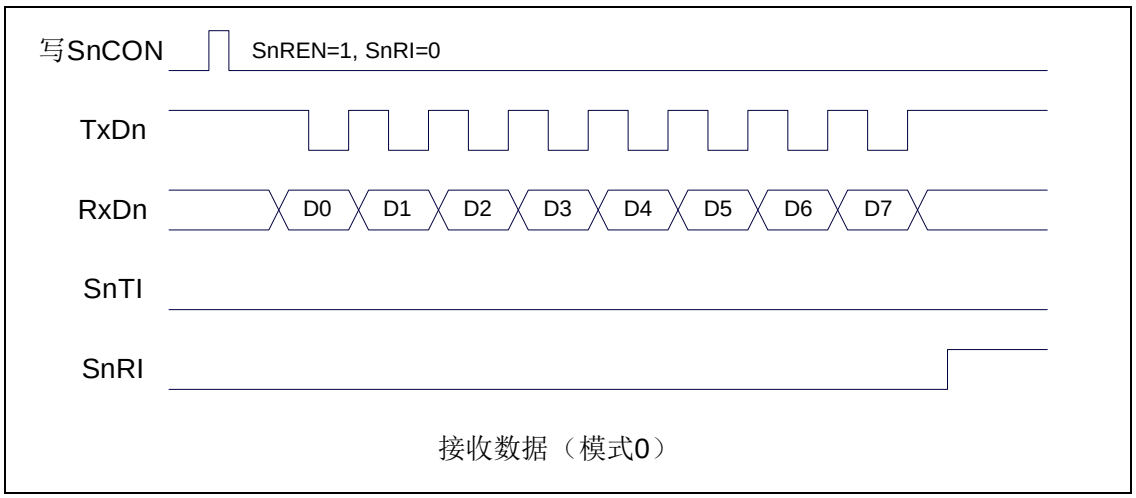
当串口 4 选择工作模式为模式 0 时，串行通信接口工作在同步移位寄存器模式，当串口 4 模式 0 的通信速度设置位 S4M0x6 为 0 时，其波特率固定为系统时钟时钟的 12 分频（SYSclk/12）；当设置 S4M0x6 为 1 时，其波特率固定为系统时钟频率的 2 分频（SYSclk/2）。RxD4 为串行通讯的数据口，TxD4 为同步移位脉冲输出脚，发送、接收的是 8 位数据，低位在先。

模式 0 的发送过程：当主机执行将数据写入发送缓冲器 S4BUF 指令时启动发送，串口 4 即将 8 位数据以 SYSclk/12 或 SYSclk/2（由 S4M0x6 确定是 12 分频还是 2 分频）的波特率从 RxD4 管脚输出(从低位到高位)，发送完中断标志 S4TI 置 1，TxD4 管脚输出同步移位脉冲信号。当写信号有效后，相隔一个时钟，发送控制端 SEND 有效(高电平)，允许 RxD4 发送数据，同时允许 TxD4 输出同步移位脉冲。一帧(8 位)数据发送完毕时，各控制端均恢复原状态，只有 S4TI 保持高电平，呈中断申请状态。在再次发送数据前，必须用软件将 S4TI 清 0。

模式 0 的接收过程：首先将接收中断请求标志 S4RI 清零并置位允许接收控制位 S4REN 时启动模式 0 接收过程。启动接收过程后，RxD4 为串行数据输入端，TxD4 为同步脉冲输出端。串行接收的波特率为 SYSclk/12 或 SYSclk/2（由 S4M0x6 确定是 12 分频还是 2 分频）。当接收完成一帧数据（8 位）后，控制信号复位，中断标志 S4RI 被置 1，呈中断申请状态。当再次接收时，必须通过软件将 S4RI 清 0



发送数据（模式0）



接收数据（模式0）

工作于模式 0 时，必须清 0 多机通信控制位 S4SM2，使之不影响 S4TB8 位和 S4RB8 位。由于波特率固定为 SYSclk/12 或 SYSclk/2，无需定时器提供，直接由单片机的时钟作为同步移位脉冲。

串口 4 模式 0 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

S4M0x6	波特率计算公式
0	$\text{波特率} = \frac{\text{SYSclk}}{12}$
1	$\text{波特率} = \frac{\text{SYSclk}}{2}$

19.5.7 串口 4 模式 1，模式 1 波特率计算公式

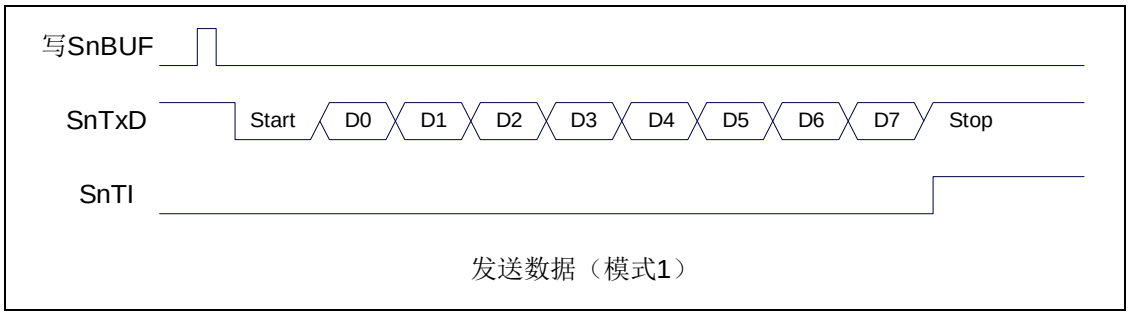
当软件设置 S4CON 的 S4SM0、S4SM1 为“01”时，串口 4 则以模式 1 进行工作。此模式为 8 位 UART 格式，一帧信息为 10 位：1 位起始位，8 位数据位（低位在先）和 1 位停止位。波特率可变，即可根据需要进行设置波特率。TxD4 为数据发送口，RxD4 为数据接收口，串口 4 全双工接受/发送。

模式 1 的发送过程：串行通信模式发送时，数据由串行发送端 TxD4 输出。当主机执行一条写 S4BUF 的指令就启动串行通信的发送，写“S4BUF”信号还把“1”装入发送移位寄存器的第 9 位，并通知 TX 控制单元开始发送。移位寄存器将数据不断右移送 TxD4 端口发送，在数据的左边不断移入“0”作补充。当数据的最高位移到移位寄存器的输出位置，紧跟其后的是第 9 位“1”，在它的左边各位全为“0”，这个状态条件，使 TX 控制单元作最后一次移位输出，然后使允许发送信号“SEND”失效，完成一帧信息的发送，并置位中断请求位 S4TI，即 S4TI=1，向主机请求中断处理。

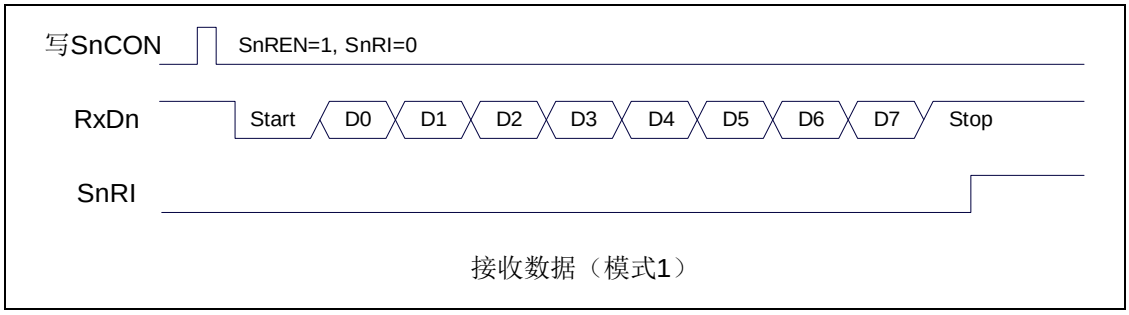
模式 1 的接收过程：当软件置位接收允许标志位 S4REN，即 S4REN=1 时，接收器便对 RxD4 端口的信号进行检测，当检测到 RxD4 端口发送从“1”→“0”的下降沿跳变时就启动接收器准备接收数据，并立即复位波特率发生器的接收计数器，将 1FFH 装入移位寄存器。接收的数据从接收移位寄存器的右边移入，已装入的 1FFH 向左边移出，当起始位"0"移到移位寄存器的最左边时，使 RX 控制器作最后一次移位，完成一帧的接收。若同时满足以下两个条件：

- S4RI=0；
- S4SM2=0 或接收到的停止位为 1。

则接收到的数据有效，实现装载入 S4BUF，停止位进入 S4RB8，S4RI 标志位被置 1，向主机请求中断，若上述两条件不能同时满足，则接收到的数据作废并丢失，无论条件满足与否，接收器重又检测 RxD4 端口上的"1"→"0"的跳变，继续下一帧的接收。接收有效，在响应中断后，S4RI 标志位必须由软件清 0。通常情况下，串行通信工作于模式 1 时，S4SM2 设置为"0"。



发送数据（模式1）



接收数据（模式1）

串口 4 的波特率是可变的，其波特率固定由定时器 2 或定时器 4 产生。当定时器采用 1T 模式时（12 倍速），相应的波特率的速度也会相应提高 12 倍。

串口 4 模式 1 的波特率计算公式如下表所示：（SYSclk 为系统工作频率）

选择定时器	定时器速度	重装值计算公式	波特率
定时器2 或 定时器4	1T	定时器重载值 = 65536 - $\frac{SYSclk}{4\times波特率}$	波特率 = $\frac{SYSclk}{4\times(65536 - 定时器重装数)}$
	12T	定时器重载值 = 65536 - $\frac{SYSclk}{12\times4\times波特率}$	波特率 = $\frac{SYSclk}{12\times4\times(65536 - 定时器重装数)}$

下面为常用频率与常用波特率所对应定时器的重载值

频率 (MHz)	波特率	定时器	
		1T 模式	12T 模式
11.0592	115200	FFE8H	FFFEH
	57600	FFD0H	FFFCH
	38400	FFB8H	FFFAH
	19200	FF70H	FFF4H
	9600	FEE0H	FFE8H
18.432	115200	FFD8H	-
	57600	FFB0H	-
	38400	FF88H	FFF6H
	19200	FF10H	FFECH
	9600	FE20H	FFD8H
22.1184	115200	FFD0H	FFFCH
	57600	FFA0H	FFF8H
	38400	FF70H	FFF4H
	19200	FEE0H	FFE8H
	9600	FDC0H	FFD0H

19.5.8 串口 4 模式 2，模式 2 波特率计算公式

当 S4SM0、S4SM1 两位为 10 时，串口 4 工作在模式 2。串口 4 工作模式 2 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S4CON 中的 S4TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S4TB8（S4TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S4CON 的 S4RB8。TxD4 为发送端口，RxD4 为接收端口，以全双工模式进行接收/发送。

模式 2 的波特率固定为系统时钟的 64 分频或 32 分频（取决于 S4CFG 中 S4MOD 的值）

串口 4 模式 2 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

SMOD	波特率计算公式
0	$波特率 = \frac{SYSclk}{64}$
1	$波特率 = \frac{SYSclk}{32}$

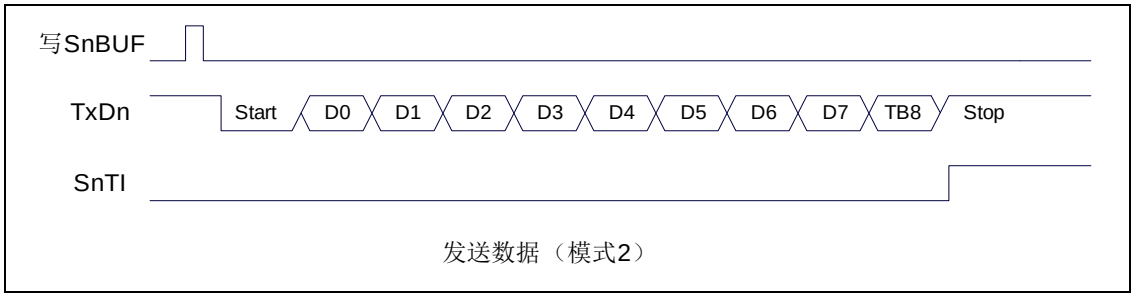
模式 2 和模式 1 相比，除波特率发生源略有不同，发送时由 S4TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收/发送操作过程及时序也基本相同。

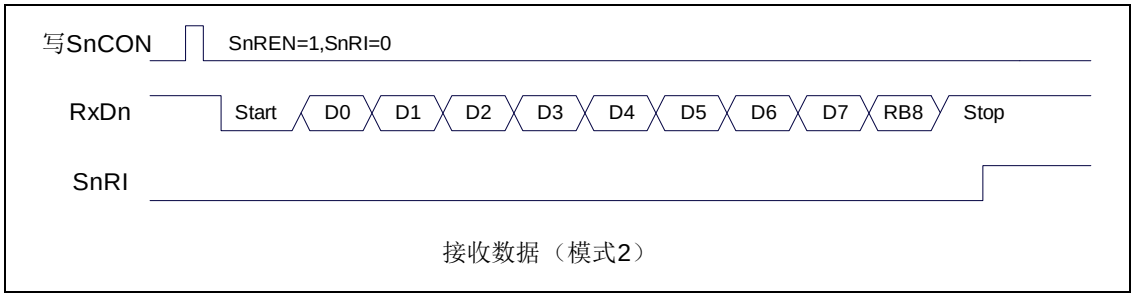
当接收器接收完一帧信息后必须同时满足下列条件：

- S4RI=0
- S4SM2=0 或者 S4SM2=1 且接收到的第 9 数据位 S4RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S4BUF 和 S4RB8 中，S4RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S4RI。无论上述条件满足与否，接收器又重新开始检测 RxD4 输入端口的跳变信息，接收下一帧的输入信息。在模式 2 中，接收到的停止位与 S4BUF、S4RB8 和 S4RI 无关。

通过软件对 S4CON 中的 S4SM2、S4TB8 的设置以及通信协议的约定，为多机通信提供了方便。





19.5.9 串口 4 模式 3，模式 3 波特率计算公式

当 S4SM0、S4SM1 两位为 11 时，串口 4 工作在模式 3。串行通信模式 3 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S4CON 中的 S4TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S4TB8（S4TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S4CON 的 S4RB8。TxD4 为发送端口，RxD4 为接收端口，以全双工模式进行接收/发送。

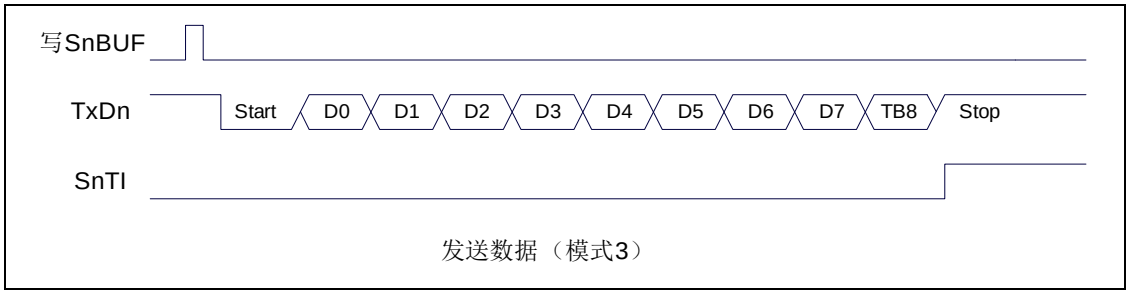
模式 3 和模式 1 相比，除发送时由 S4TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收‘发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

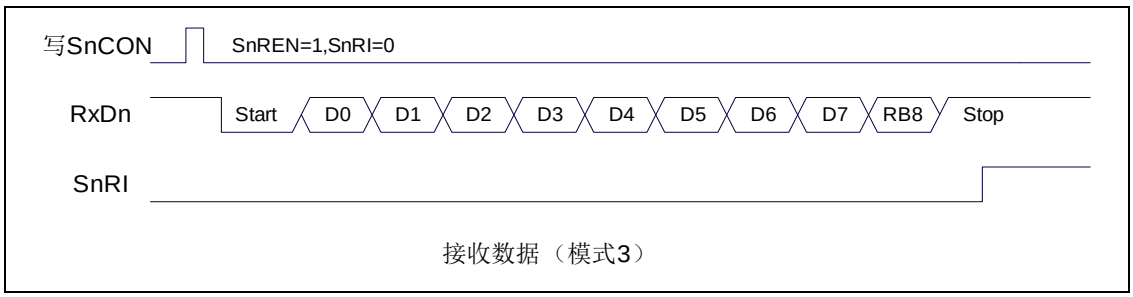
- S4RI=0
- S4SM2=0 或者 S4SM2=1 且接收到的第 9 数据位 S4RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S4BUF 和 S4RB8 中，S4RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S4RI。无论上述条件满足与否，接收器又重新开始检测 RxD4 输入端口的跳变信息，接收下一帧的输入信息。在模式 3 中，接收到的停止位与 S4BUF、S4RB8 和 S4RI 无关。

通过软件对 S4CON 中的 S4SM2、S4TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式3）



接收数据（模式3）

串口 4 模式 3 的波特率计算公式与模式 1 是完全相同的。请参考模式 1 的波特率计算公式。

19.5.10 串口 4 自动地址识别，从机地址控制寄存器（S4ADDR，S4ADEN）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S4ADDR	7EFDBDH								
S4ADEN	7EFDBEH								

S4ADDR：从机地址寄存器

S4ADEN：从机地址屏蔽位寄存器

自动地址识别功能典型应用在多机通讯领域，其主要原理是从机系统通过硬件比较功能来识别来自于主机串口 4 数据流中的地址信息，通过寄存器 S4ADDR 和 S4ADEN 设置的本机的从机地址，硬件自动对从机地址进行过滤，当来自于主机的从机地址信息与本机所设置的从机地址相匹配时，硬件产生串口 4 中断；否则硬件自动丢弃串口 4 数据，而不产生中断。当众多处于空闲模式的从机链接在一起时，只有从机地址相匹配的从机才会从空闲模式唤醒，从而可以大大降低从机 MCU 的功耗，即使从机处于正常工作状态也可避免不停地进入串口 4 中断而降低系统执行效率。

要使用串口 4 的自动地址识别功能，首先需要将参与通讯的 MCU 的串口 4 通讯模式设置为模式 1，并开启从机的 S4CON 的 S4SM2 位。对于串口 4 模式 1 的 9 位数据位中，第 9 位数据（存放在 S4RB8 中）为地址/数据的标志位，当第 9 位数据为 1 时，表示前面的 8 位数据（存放在 S4BUF 中）为地址信息。当 S4SM2 被设置为 1 时，从机 MCU 会自动过滤掉非地址数据（第 9 位为 0 的数据），而对 S4BUF 中的地址数据（第 9 位为 1 的数据）自动与 S4ADDR 和 S4ADEN 所设置的本机地址进行比较，若地址相匹配，则会将 S4RI 置“1”，并产生中断，否则不予处理本次接收的串口 4 数据。

从机地址的设置是通过 S4ADDR 和 S4ADEN 两个寄存器进行设置的。S4ADDR 为从机地址寄存器，里面存放本机的从机地址。S4ADEN 为从机地址屏蔽位寄存器，用于设置地址信息中的忽略位，设置方法如下：

例如

S4ADDR ＝ 11001010

S4ADEN ＝ 10000001

则匹配地址为 1xxxxxx0

即，只要主机送出的地址数据中的 bit0 为 0 且 bit7 为 1 就可以和本机地址相匹配

再例如

S4ADDR ＝ 11001010

S4ADEN ＝ 00001111

则匹配地址为 xxxx1010

即，只要主机送出的地址数据中的低 4 位为 1010 就可以和本机地址相匹配，而高 4 为被忽略，可以为任意值。

主机可以使用广播地址（FFH）同时选中所有的从机来进行通讯。

19.5.11 串口 4 同步模式控制寄存器 1（USART4CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART4CR1	7EFDD8H	S4LINEN	S4DORD	S4CLKEN	S4SPMOD	S4SPIEN	S4SPSLV	S4CPOL	S4CPHA

S4LINEN：LIN 模式使能位

0：禁止 LIN 模式

1：使能 LIN 模式

S4DORD：SPI 模式数据位发送/接收的顺序

0：先发送/接收数据的高位（MSB）

1：先发送/接收数据的低位（LSB）

S4CLKEN：SmartCard 模式时钟输出控制位

0：禁止时钟输出

1：使能时钟输出

S4SPMOD：SPI 模式使能位

0：禁止 SPI 模式

1：使能 SPI 模式

S4SPIEN：SPI 使能位

0：禁止 SPI 功能

1：使能 SPI 功能

当 USART4 需要工作在 SPI 模式时，必须先将 SPI 相关的模式寄存器、速度寄存器、时钟极性寄存器设置完成后，最后再设置 SPIEN 寄存器。

S4SPSLV：SPI 从机模式使能位

0：SPI 为主机模式

1：SPI 为从机模式

S4CPOL：SPI 时钟极性控制

0：SCLK 空闲时为低电平，SCLK 的前时钟沿为上升沿，后时钟沿为下降沿

1：SCLK 空闲时为高电平，SCLK 的前时钟沿为下降沿，后时钟沿为上升沿

S4CPHA：SPI 时钟相位控制

0：数据 SS 管脚为低电平驱动第一位数据并在 SCLK 的后时钟沿改变数据，前时钟沿采样数据

1：数据在 SCLK 的前时钟沿驱动，后时钟沿采样

19.5.12 串口 4 同步模式控制寄存器 2（USART4CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART4CR2	7EFDD9H	S4IREN	S4IRLP	S4SCEN	S4NACK	S4HDSEL	S4PCEN	S4PS	S4PE

S4IREN：IrDA 模式使能位

0：禁止 IrDA 模式

1：使能 IrDA 模式

S4IRLP：IrDA 低电模式控制位

0：IrDA 为普通模式

1：IrDA 为低电模式

S4SCEN：SmartCard 模式使能位

0：禁止 SmartCard 模式

1：使能 SmartCard 模式

S4NACK：SmartCard 模式 NACK 输出使能位

0：禁止输出 NACK 信号

1：使能输出 NACK 信号

S4HDSEL：单线半双工模式使能位

0：禁止单线半双工模式

1：使能单线半双工模式

S4PCEN：硬件自动产生校验位控制使能

0：禁止硬件自动产生校验位（串口的校验位为 S4TB8 设置的值）

1：使能硬件自动产生校验位

S4PS：硬件校验位模式选择

0：硬件根据 S4BUF 的值自动产生偶校验位

1：硬件根据 S4BUF 的值自动产生奇校验位

S4PE：校验位错误标志（必须软件清零）

0：无检验错误

1：有校验错误（串口接收 DMA 过程中如果发生接收数据校验位错误，DMA 不会停止，但校验位错误标志会一直保持直到 DMA 完成）

19.5.13 串口 4 同步模式控制寄存器 3（USART4CR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART4CR3	7EFDDAH	S4IrDA_LPBAUD[7:0]							

S4IrDA_LPBAUD：IrDA 低电模式波特率控制寄存器

IrDA_LPBAUD[7:0]	IrDA 低电模式波特率
0	SYSclk/16/256
1	SYSclk/16/1
2	SYSclk/16/2
...	...
255	SYSclk/16/255

19.5.14 串口 4 同步模式控制寄存器 4（USART4CR4）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART4CR4	7EFDDBH	-	-	-	-	S4SCCKS[1:0]		S4SPICKS[1:0]	

S4SCCKS[1:0]：SmartCard 模式时钟选择

SCCKS[1:0]	SmartCard 模式时钟
00	SYSclk/64
01	SYSclk/32
10	SYSclk/16
11	SYSclk/8

S4SPICKS[1:0]：SPI 模式时钟选择

SPICKS[1:0]	SPI 模式时钟
00	SYSclk/4
01	SYSclk/8
10	SYSclk/16
11	SYSclk/2

19.5.15 串口 4 同步模式控制寄存器 5（USART4CR5）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART4CR5	7EFDDCH	S4BRKDET	S4HDRER	S4SLVEN	S4ASYNC	S4TXCF	S4SENDBK	S4HDRDET	S4SYNC

S4BRKDET：LIN 从模式间隔场（BREAK）检测标志

0：未检测到 LIN 间隔场

1：检测到 LIN 间隔场，需要软件清零

S4HDRER：LIN 从模式报文头（HEADER）错误

0：未检测到 LIN 报文头错误

1：检测到 LIN 报文头错误，需要软件清零

S4SLVEN：LIN 从机模式使能位

0：LIN 为主机模式

1：LIN 为从机模式

S4ASYNC：LIN 自动同步功能使能位

0：禁止自动同步

1：使能自动同步

【注意】：从机使能自动同步功能后，如果从机跟主机波特率存在小范围误差，从机收到主机同步信号后，会自动将从机波特率设置成与主机相同的波特率。

例如，主机波特率 9600：

从机使能自动同步时，如果从机波特率在 9100～10000 附近，收到主机同步信号后，从机会自动将本身波特率设置为 9600。误差自动同步范围 -5.2%～+4.16%左右。

S4TXCF：传输冲突标志位。单线半双工模式、LIN 模式和 SmartCard 模式有效

0：未检测到传输冲突（发送的数据与接收的数据相同）

1：检测到传输冲突（发送的数据与接收的数据不同）

S4SENDBK：发送间隔场。软件写 1 触发发送间隔场，发送完成后硬件自动清 0

S4HDRDET：LIN 从模式报文头（HEADER）检测标志

0：未检测到 LIN 报文头

1：检测到 LIN 报文头，需要软件清零

S4SYNC：LIN 从模式同步场检测标志。

正确分析到同步场后，硬件将 SYNC 标志为置 1。在接收标志符场时硬件会自动清零 SYNC

19.5.16 串口 4 同步模式保护时间寄存器（USART4GTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART4GTR	7EFDDDH								

USART4GTR 寄存器仅在 SmartCard 模式有效。该寄存器是根据波特率时钟数给出保护时间值。S4TI 标志在 SmartCard 发送的数据位等于 USART4GTR 寄存器所设置的保护时间值时被硬件置 1。注：USART4GTR 寄存器的值应大于 11

19.5.17 串口 4 同步模式波特率寄存器（USART4BR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART4BRH	7EFDDEH	USART4BR[15:8]							
USART4BRL	7EFDDFH	USART4BR[7:0]							

LIN 模式、IrDA 模式和 SmartCard 模式时的波特率计算公式

$$\text{同步波特率} = \frac{\text{SYSclk}}{16 * \text{USART4BR[15:0]}}$$

19.5.18 串口 4 接收超时控制寄存器（UR4TOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR4TOCR	7EFD7CH	UR4ENTO	UR4ENTOI	UR4SCALE	-	-	-	-	-

UR4ENTO：串口 4 接收超时功能控制位

0：禁止串口 4 接收超时功能

1：使能串口 4 接收超时功能

UR4ENTOI：串口 4 接收超时中断控制位

0：禁止串口 4 接收超时中断

1：使能串口 4 接收超时中断

UR4SCALE：串口 4 超时计数时钟源选择

0：串口数据位率（波特率）

1：系统时钟

19.5.19 串口 4 超时状态寄存器（UR4TOSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR4TOSR	7EFD7DH	UR4CTOIF	-	-	-	-	-	-	UR4TOIF

UR4CTOIF：写“1”清除串口超时中断标志位 TOIF。（只写）

UR4TOIF：串口 4 超时中断请求标志位。（只读）

当发生串口 4 超时时，TOIF 会被硬件置“1”。如果 ENTOI 为 1 时会产生串口中断，中断入口地址为原串口 4 的中断入口地址。标志位需要软件向 CTOIF 位写“1”清零

19.5.20 串口 4 超时长度控制寄存器（UR4TOTE/H/L）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR4TOTE	7EFD8BH	UR4TM[23:16]							
UR4TOTH	7EFD7EH	UR4TM[15:8]							
UR4TOTL	7EFD7FH	UR4TM[7:0]							

UR4TM[23:0]：串口 4 超时时间控制位。

当串口 4 处于接收空闲状态时，内部超时计数器根据 SCALE 寄存器所选择的时钟源进行计数，当计数时间达到 UR4TM 所设置的超时时间时，便会产生超时中断。当串口 4 接收数据完成时，复位内部超时计数器，重新进行超时计数。

注：

1、如果需要使能接收超时中断功能，则 TM 不可设置为 0，且 UR4TOTL、UR4TOTH、UR4TOTE 寄存器的设置必须先设置 UR4TOTL 和 UR4TOTH，最后设置 UR4TOTE

2、必须使能串口 4 接收才有接收超时功能

3、接收超时功能必须在正确接收到一字节数据后才能触发

4、正在接收过程中，无接收超时功能

19.6 串口 5（同步/异步串口 USART5）

19.6.1 串口 5 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX2	7EFD6AH	-	-	LIN2_S[1:0]		S6_S[1:0]		**S5_S[1:0]**	

S5_S[1:0]：串口 5 功能脚选择位

S5_S[1:0]	RxD5	TxD5
00	P0.4	P0.5
01	P4.0	P4.1
10	P1.4	P1.5
11	P2.0	P2.1

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX3	7EFD6BH	S6SPI_S[1:0]		**S5SPI_S[1:0]**		S4SPI_S[1:0]		S3SPI_S[1:0]	

S5SPI_S[1:0]：USART5 的 SPI 功能脚选择位

S5SPI_S[1:0]	S5SS	S5MOSI	S5MISO	S5SCLK
00	P2.2	P2.3	P2.4	P2.5
01	P7.0	P7.1	P7.2	P7.3
10	P8.4	P8.5	P8.6	P8.7
11	P9.0	P9.1	P9.2	P9.3

19.6.2 串口 5 相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S5CON	串口 5 控制寄存器	7EF740H	S5SM0/FE	S5SM1	S5SM2	S5REN	S5TB8	S5RB8	S5TI	S5RI	0000,0000
S5BUF	串口 5 数据寄存器	7EF741H	S5BUF[7:0]								0000,0000
S5INTR	串口 5 中断寄存器	7EF742H	ES5	-	PS5[1:0]		-	-	-	-	0x00,xxxx

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S5CFG	串口 5 配置寄存器	7EF76DH	S5MOD	S5MOD0	S5M0x6	-	-	-	-	S5BRT	000x,xxx0
S5ADDR	串口 5 从机地址寄存器	7EF76EH	S5ADDR[7:0]								0000,0000
S5ADEN	串口 5 从机地址屏蔽寄存器	7EF76FH	S5ADEN[7:0]								0000,0000
USART5CR1	串口 5 控制寄存器 1	7EF748H	S5LINEN	S5DORD	S5CLKEN	S5SPMOD	S5SPIEN	S5SPSLV	S5CPOL	S5CPHA	0000,0000
USART5CR2	串口 5 控制寄存器 2	7EF749H	S5IREN	S5IRLP	S5SCEN	S5NACK	S5HDSEL	S5PCEN	S5PS	S5PE	0000,0000
USART5CR3	串口 5 控制寄存器 3	7EF74AH	S5IrDA_LPBAUD[7:0]								0000,0111
USART5CR4	串口 5 控制寄存器 4	7EF74BH	-	-	-	-	S5SCCKS[1:0]		S5SPICKS[1:0]		xxxx,0000
USART5CR5	串口 5 控制寄存器 5	7EF74CH	S5BRKDET	S5HDRER	S5SLVEN	S5ASYNC	S5TXCF	S5SENDBK	S5HDRDET	S5SYNC	0000,0000
USART5GTR	串口 5 保护时间寄存器	7EF74DH	USART5GTR[7:0]								0000,0000
USART5BRH	串口 5 波特率寄存器	7EF74EH	USART5BR[15:8]								0000,0000
USART5BRL	串口 5 波特率寄存器	7EF74FH	USART5BR[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
UR5TOCR	串口 5 接收超时控制寄存器	7EF744H	UR5ENTO	UR5ENTOI	UR5SCALE	-	-	-	-	-	000x,xxxx
UR5TOSR	串口 5 接收超时状态寄存器	7EF745H	-	-	-	-	-	-	-	UR5TOIF	xxxx,xxx0
UR5TOTH	串口 5 接收超时长度寄存器	7EF746H	UR5TM[15:8]								0000,0000
UR5TOTL	串口 5 接收超时长度寄存器	7EF747H	UR5TM[7:0]								0000,0000
UR5TOTE	串口 5 接收超时长度寄存器	7EF743H	UR5TM[23:16]								0000,0000

19.6.3 串口 5 控制寄存器（S5CON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S5CON	7EF740H	S5SM0/FE	S5SM1	S5SM2	S5REN	S5TB8	S5RB8	S5TI	S5RI

S5SM0/FE：当S5CFG寄存器中的S5MOD0位为1时，该位为帧错误检测标志位。当串口5在接收过程中检测到一个无效停止位时，通过串口5接收器将该位置1，必须由软件清零。当S5CFG寄存器中的S5MOD0位为0时，该位和S5SM1一起指定串口5的通信工作模式，如下表所示：

S5SM0	S5SM1	串口5工作模式	功能说明
0	0	模式0	同步移位串行方式
0	1	模式1	可变波特率8位数据方式
1	0	模式2	固定波特率9位数据方式
1	1	模式3	可变波特率9位数据方式

当使能 FE 功能后，串口的工作模式只能使用 SM1 进行设置，且只能设置为模式 0 或者模式 1。FE 当作帧错误标志位时，此标志位必须软件写 0 清零。

S5SM2：允许模式 2 或模式 3 多机通信控制位。当串口 5 使用模式 2 或模式 3 时，如果 S5SM2 位为 1 且 S5REN 位为 1，则接收机处于地址帧筛选状态。此时可以利用接收到的第 9 位（即 S5RB8）来筛选地址帧，若 S5RB8=1，说明该帧是地址帧，地址信息可以进入 S5BUF，并使 S5RI 为 1，进而在中断服务程序中再进行地址号比较；若 S5RB8=0，说明该帧不是地址帧，应丢掉且保持 S5RI=0。在模式 2 或模式 3 中，如果 S5SM2 位为 0 且 S5REN 位为 1，接收收机处于地址帧筛选被禁止状态，不论收到的 S5RB8 为 0 或 1，均可使接收到的信息进入 S5BUF，并使 S5RI=1，此时 S5RB8 通常为校验位。模式 1 和模式 0 为非多机通信方式，在这两种方式时，S5SM2 应设置为 0。

S5REN：允许/禁止串口接收控制位

0：禁止串口接收数据

1：允许串口接收数据

S5TB8：当串口 5 使用模式 2 或模式 3 时，S5TB8 为要发送的第 9 位数据，按需要由软件置位或清 0。在模式 0 和模式 1 中，该位不用。

S5RB8：当串口 5 使用模式 2 或模式 3 时，S5RB8 为接收到的第 9 位数据，一般用作校验位或者地址帧/数据帧标志位。在模式 0 和模式 1 中，该位不用。

S5TI：串口 5 发送中断请求标志位。在模式 0 中，当串口发送数据第 8 位结束时，由硬件自动将 S5TI 置 1，向主机请求中断，响应中断后 S5TI 必须用软件清零。在其他模式中，则在停止位开始发送时由硬件自动将 S5TI 置 1，向 CPU 发请求中断，响应中断后 S5TI 必须用软件清零。

S5RI：串口 5 接收中断请求标志位。在模式 0 中，当串口接收第 8 位数据结束时，由硬件自动将 S5RI 置 1，向主机请求中断，响应中断后 S5RI 必须用软件清零。在其他模式中，串行接收到停止位的中间时刻由硬件自动将 S5RI 置 1，向 CPU 发中断申请，响应中断后 S5RI 必须由软件清零。

19.6.4 串口 5 数据寄存器（S5BUF）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S5BUF	7EF741H								

S5BUF：串口 1 数据接收/发送缓冲区。S5BUF 实际是 2 个缓冲器，读缓冲器和写缓冲器，两个操作分别对应两个不同的寄存器，1 个是只写寄存器（写缓冲器），1 个是只读寄存器（读缓冲器）。对 S5BUF 进行读操作，实际是读取串口接收缓冲区，对 S5BUF 进行写操作则是触发串口开始发送数据。

19.6.5 串口 5 中断控制寄存器（S5INTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S5INTR	7EF742H	ES5	-	PS5[1:0]		-	-	-	-

ES5：串口 5 中断使能控制位

0：禁止串口 5 中断

1：允许串口 5 中断

PS5[1:0]：串口 5 中断优先级控制位

00：串口 5 中断优先级为 0 级（最低级）

01：串口 5 中断优先级为 1 级（较低级）

10：串口 5 中断优先级为 2 级（较高级）

11：串口 5 中断优先级为 3 级（最高级）

19.6.6 串口 5 配置寄存器（S5CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S5CFG	7EF76DH	S5MOD	S5MOD0	S5M0x6					S5BRT

S5MOD：串口 5 波特率控制位

0：串口 5 之模式 0/1/2/3 的波特率都不加倍

1：串口 5 之模式 2 的波特率加倍

S5MOD0：帧错误检测控制位

0：无帧错检测功能

1：使能帧错误检测功能。此时 S5CON 的 S5SM0/FE 为 FE 功能，即为帧错误检测标志位。

S5M0x6：串口 5 模式 0 的通讯速度控制

0：串口 5 模式 0 的波特率不加倍，固定为 Fosc/12

1：串口 5 模式 0 的波特率 6 倍速，即固定为 Fosc/12*6 ＝ Fosc/2

S5BRT：串口 5 波特率发生器选择位

0：选择定时器 2 作为波特率发生器（**默认值**）

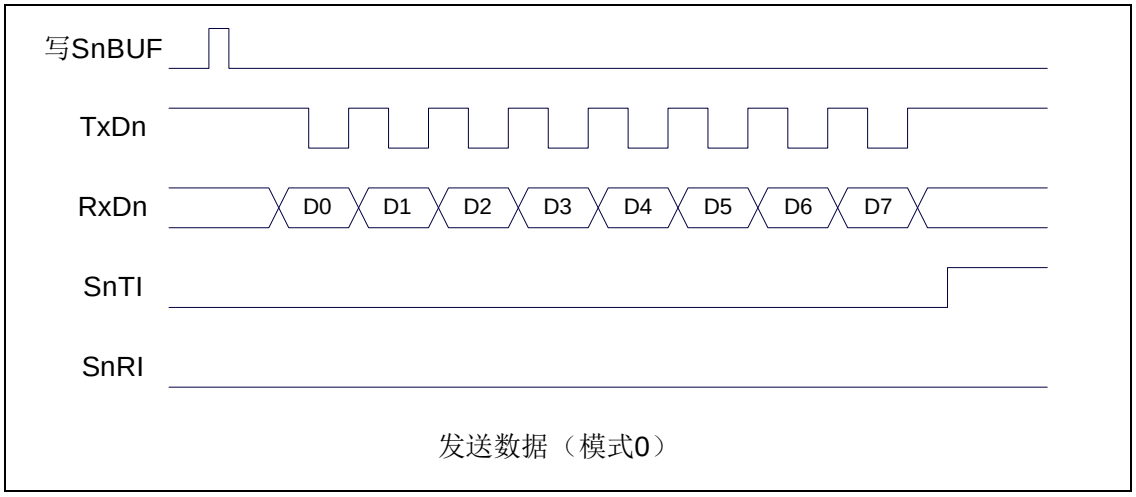
1：选择定时器 5 作为波特率发生器

19.6.7 串口 5 模式 0，模式 0 波特率计算公式

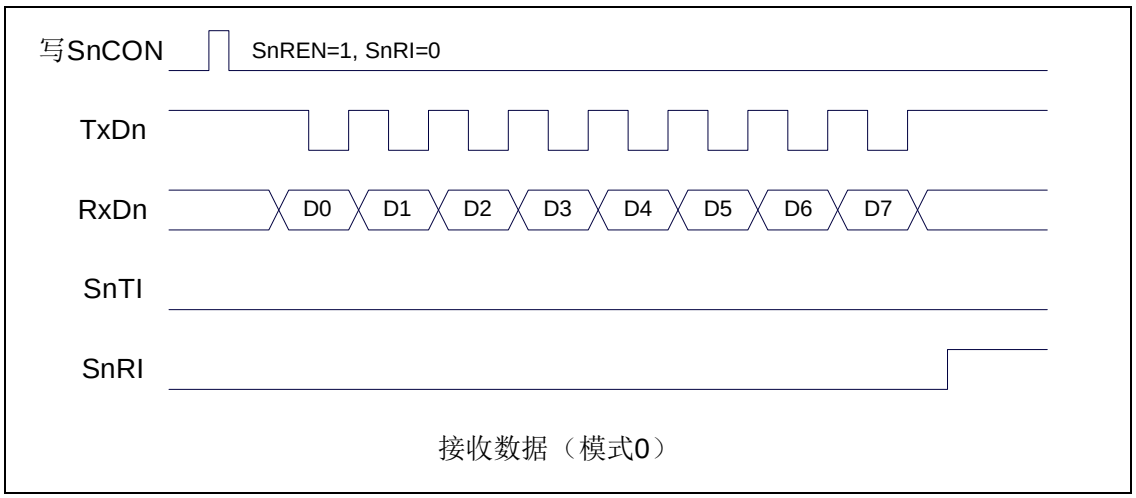
当串口 5 选择工作模式为模式 0 时，串行通信接口工作在同步移位寄存器模式，当串口 5 模式 0 的通信速度设置位 S5M0x6 为 0 时，其波特率固定为系统时钟时钟的 12 分频（SYSclk/12）；当设置 S5M0x6 为 1 时，其波特率固定为系统时钟频率的 2 分频（SYSclk/2）。RxD5 为串行通讯的数据口，TxD5 为同步移位脉冲输出脚，发送、接收的是 8 位数据，低位在先。

模式 0 的发送过程：当主机执行将数据写入发送缓冲器 S5BUF 指令时启动发送，串口 5 即将 8 位数据以 SYSclk/12 或 SYSclk/2（由 S5M0x6 确定是 12 分频还是 2 分频）的波特率从 RxD5 管脚输出(从低位到高位)，发送完中断标志 S5TI 置 1，TxD5 管脚输出同步移位脉冲信号。当写信号有效后，相隔一个时钟，发送控制端 SEND 有效(高电平)，允许 RxD5 发送数据，同时允许 TxD5 输出同步移位脉冲。一帧(8 位)数据发送完毕时，各控制端均恢复原状态，只有 S5TI 保持高电平，呈中断申请状态。在再次发送数据前，必须用软件将 S5TI 清 0。

模式 0 的接收过程：首先将接收中断请求标志 S5RI 清零并置位允许接收控制位 S5REN 时启动模式 0 接收过程。启动接收过程后，RxD5 为串行数据输入端，TxD5 为同步脉冲输出端。串行接收的波特率为 SYSclk/12 或 SYSclk/2（由 S5M0x6 确定是 12 分频还是 2 分频）。当接收完成一帧数据（8 位）后，控制信号复位，中断标志 S5RI 被置 1，呈中断申请状态。当再次接收时，必须通过软件将 S5RI 清 0



发送数据（模式0）



接收数据（模式0）

工作于模式 0 时，必须清 0 多机通信控制位 S5SM2，使之不影响 S5TB8 位和 S5RB8 位。由于波特率固定为 SYSclk/12 或 SYSclk/2，无需定时器提供，直接由单片机的时钟作为同步移位脉冲。

串口 5 模式 0 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

S5M0x6	波特率计算公式
0	$\text{波特率} = \frac{\text{SYSclk}}{12}$
1	$\text{波特率} = \frac{\text{SYSclk}}{2}$

19.6.8 串口 5 模式 1，模式 1 波特率计算公式

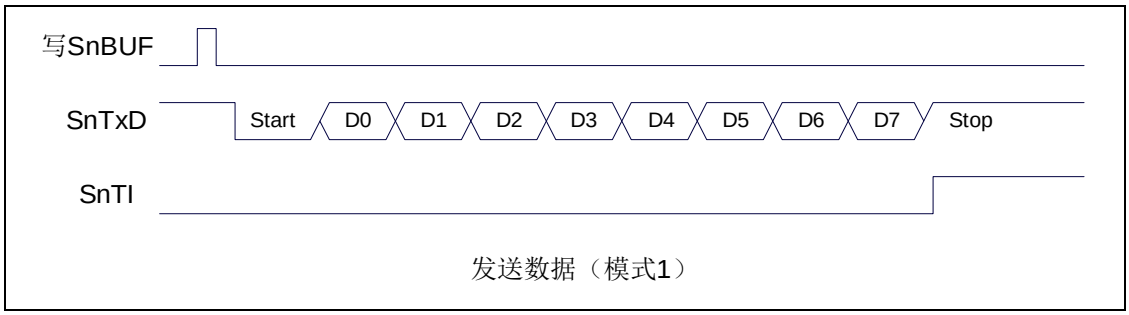
当软件设置 S5CON 的 S5SM0、S5SM1 为“01”时，串口 5 则以模式 1 进行工作。此模式为 8 位 UART 格式，一帧信息为 10 位：1 位起始位，8 位数据位（低位在先）和 1 位停止位。波特率可变，即可根据需要进行设置波特率。TxD5 为数据发送口，RxD5 为数据接收口，串口 5 全双工接受/发送。

模式 1 的发送过程：串行通信模式发送时，数据由串行发送端 TxD5 输出。当主机执行一条写 S5BUF 的指令就启动串行通信的发送，写“S5BUF”信号还把“1”装入发送移位寄存器的第 9 位，并通知 TX 控制单元开始发送。移位寄存器将数据不断右移送 TxD5 端口发送，在数据的左边不断移入“0”作补充。当数据的最高位移到移位寄存器的输出位置，紧跟其后的是第 9 位“1”，在它的左边各位全为“0”，这个状态条件，使 TX 控制单元作最后一次移位输出，然后使允许发送信号“SEND”失效，完成一帧信息的发送，并置位中断请求位 S5TI，即 S5TI=1，向主机请求中断处理。

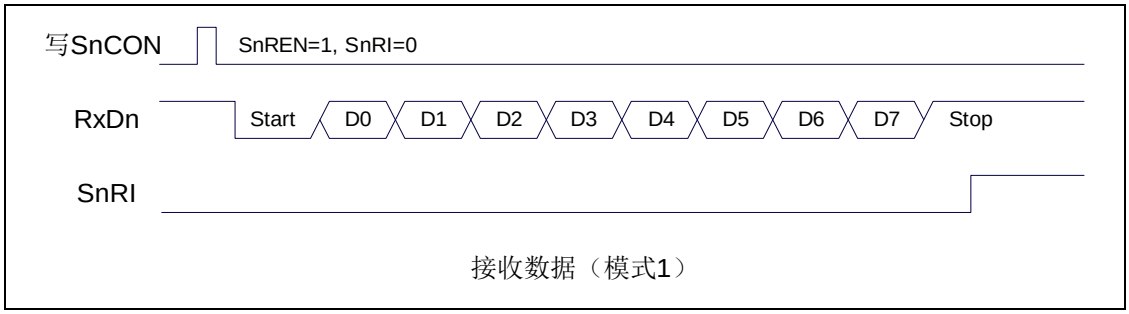
模式 1 的接收过程：当软件置位接收允许标志位 S5REN，即 S5REN=1 时，接收器便对 RxD5 端口的信号进行检测，当检测到 RxD5 端口发送从“1”→“0”的下降沿跳变时就启动接收器准备接收数据，并立即复位波特率发生器的接收计数器，将 1FFH 装入移位寄存器。接收的数据从接收移位寄存器的右边移入，已装入的 1FFH 向左边移出，当起始位"0"移到移位寄存器的最左边时，使 RX 控制器作最后一次移位，完成一帧的接收。若同时满足以下两个条件：

- S5RI=0；
- S5SM2=0 或接收到的停止位为 1。

则接收到的数据有效，实现装载入 S5BUF，停止位进入 S5RB8，S5RI 标志位被置 1，向主机请求中断，若上述两条件不能同时满足，则接收到的数据作废并丢失，无论条件满足与否，接收器重又检测 RxD5 端口上的"1"→"0"的跳变，继续下一帧的接收。接收有效，在响应中断后，S5RI 标志位必须由软件清 0。通常情况下，串行通信工作于模式 1 时，S5SM2 设置为"0"。



发送数据（模式1）



接收数据（模式1）

串口 5 的波特率是可变的，其波特率固定由定时器 2 或定时器 5 产生。当定时器采用 1T 模式时（12 倍速），相应的波特率的速度也会相应提高 12 倍。

串口 5 模式 1 的波特率计算公式如下表所示：（SYSclk 为系统工作频率）

选择定时器	定时器速度	重装值计算公式	波特率
定时器2 或 定时器5	1T	定时器重载值 = 65536 - $\frac{SYSclk}{4\times波特率}$	波特率 = $\frac{SYSclk}{4\times(65536 - 定时器重装数)}$
	12T	定时器重载值 = 65536 - $\frac{SYSclk}{12\times4\times波特率}$	波特率 = $\frac{SYSclk}{12\times4\times(65536 - 定时器重装数)}$

下面为常用频率与常用波特率所对应定时器的重载值

频率 (MHz)	波特率	定时器	
		1T 模式	12T 模式
11.0592	115200	FFE8H	FFFEH
	57600	FFD0H	FFFCH
	38400	FFB8H	FFFAH
	19200	FF70H	FFF4H
	9600	FEE0H	FFE8H
18.432	115200	FFD8H	-
	57600	FFB0H	-
	38400	FF88H	FFF6H
	19200	FF10H	FFECH
	9600	FE20H	FFD8H
22.1184	115200	FFD0H	FFFCH
	57600	FFA0H	FFF8H
	38400	FF70H	FFF4H
	19200	FEE0H	FFE8H
	9600	FDC0H	FFD0H

19.6.9 串口 5 模式 2，模式 2 波特率计算公式

当 S5SM0、S5SM1 两位为 10 时，串口 5 工作在模式 2。串口 5 工作模式 2 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S5CON 中的 S5TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S5TB8（S5TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S5CON 的 S5RB8。TxD5 为发送端口，RxD5 为接收端口，以全双工模式进行接收/发送。

模式 2 的波特率固定为系统时钟的 64 分频或 32 分频（取决于 S5CFG 中 S5MOD 的值）

串口 5 模式 2 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

SMOD	波特率计算公式
0	波特率 $= \frac{SYSclk}{64}$
1	波特率 $= \frac{SYSclk}{32}$

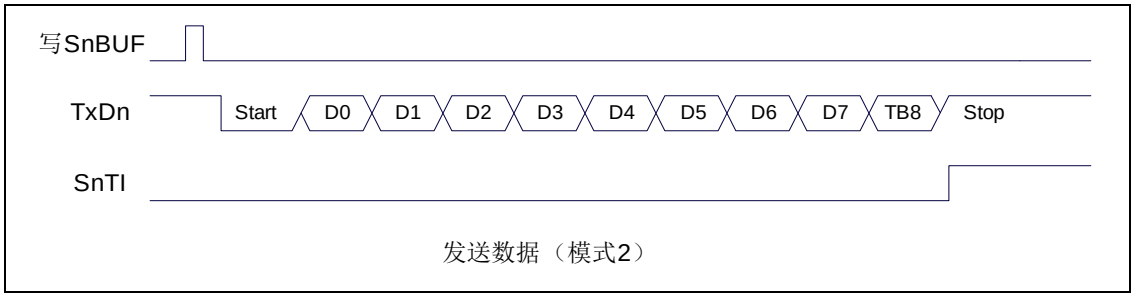
模式 2 和模式 1 相比，除波特率发生源略有不同，发送时由 S5TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收/发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

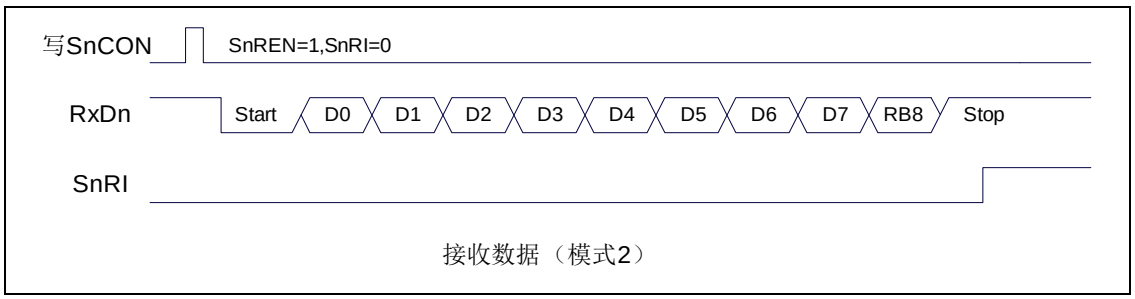
- S5RI=0
- S5SM2=0 或者 S5SM2=1 且接收到的第 9 数据位 S5RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S5BUF 和 S5RB8 中，S5RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S5RI。无论上述条件满足与否，接收器又重新开始检测 RxD5 输入端口的跳变信息，接收下一帧的输入信息。在模式 2 中，接收到的停止位与 S5BUF、S5RB8 和 S5RI 无关。

通过软件对 S5CON 中的 S5SM2、S5TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式2）



接收数据（模式2）

19.6.10 串口 5 模式 3，模式 3 波特率计算公式

当 S5SM0、S5SM1 两位为 11 时，串口 5 工作在模式 3。串行通信模式 3 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S5CON 中的 S5TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S5TB8（S5TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S5CON 的 S5RB8。TxD5 为发送端口，RxD5 为接收端口，以全双工模式进行接收/发送。

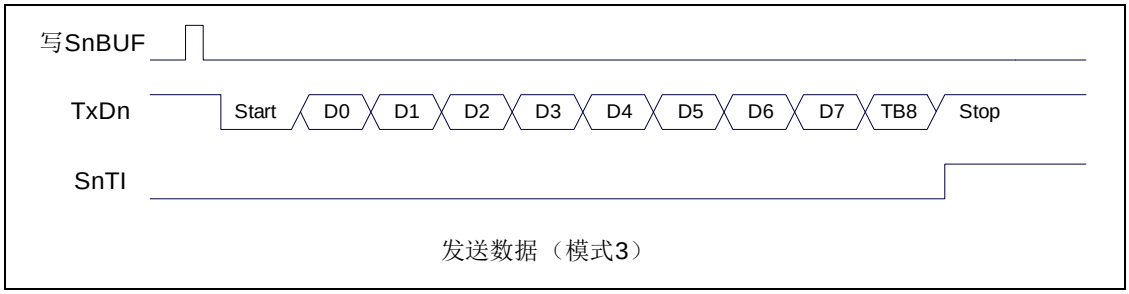
模式 3 和模式 1 相比，除发送时由 S5TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收‘发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

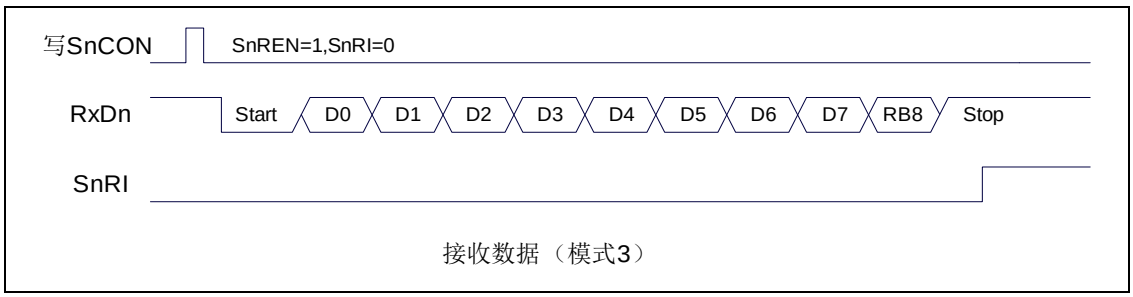
- S5RI=0
- S5SM2=0 或者 S5SM2=1 且接收到的第 9 数据位 S5RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S5BUF 和 S5RB8 中，S5RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S5RI。无论上述条件满足与否，接收器又重新开始检测 RxD5 输入端口的跳变信息，接收下一帧的输入信息。在模式 3 中，接收到的停止位与 S5BUF、S5RB8 和 S5RI 无关。

通过软件对 S5CON 中的 S5SM2、S5TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式3）



接收数据（模式3）

串口 5 模式 3 的波特率计算公式与模式 1 是完全相同的。请参考模式 1 的波特率计算公式。

19.6.11 串口 5 自动地址识别，从机地址控制寄存器（S5ADDR，S5ADEN）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S5ADDR	7EF76EH								
S5ADEN	7EF76FH								

S5ADDR：从机地址寄存器

S5ADEN：从机地址屏蔽位寄存器

自动地址识别功能典型应用在多机通讯领域，其主要原理是从机系统通过硬件比较功能来识别来自于主机串口 5 数据流中的地址信息，通过寄存器 S5ADDR 和 S5ADEN 设置的本机的从机地址，硬件自动对从机地址进行过滤，当来自于主机的从机地址信息与本机所设置的从机地址相匹配时，硬件产生串口 5 中断；否则硬件自动丢弃串口 5 数据，而不产生中断。当众多处于空闲模式的从机链接在一起时，只有从机地址相匹配的从机才会从空闲模式唤醒，从而可以大大降低从机 MCU 的功耗，即使从机处于正常工作状态也可避免不停地进入串口 5 中断而降低系统执行效率。

要使用串口 5 的自动地址识别功能，首先需要将参与通讯的 MCU 的串口 5 通讯模式设置为模式 1，并开启从机的 S5CON 的 S5SM2 位。对于串口 5 模式 1 的 9 位数据位中，第 9 位数据（存放在 S5RB8 中）为地址/数据的标志位，当第 9 位数据为 1 时，表示前面的 8 位数据（存放在 S5BUF 中）为地址信息。当 S5SM2 被设置为 1 时，从机 MCU 会自动过滤掉非地址数据（第 9 位为 0 的数据），而对 S5BUF 中的地址数据（第 9 位为 1 的数据）自动与 S5ADDR 和 S5ADEN 所设置的本机地址进行比较，若地址相匹配，则会将 S5RI 置“1”，并产生中断，否则不予处理本次接收的串口 5 数据。

从机地址的设置是通过 S5ADDR 和 S5ADEN 两个寄存器进行设置的。S5ADDR 为从机地址寄存器，里面存放本机的从机地址。S5ADEN 为从机地址屏蔽位寄存器，用于设置地址信息中的忽略位，设置方法如下：

例如
S5ADDR ＝ 11001010
S5ADEN ＝ 10000001
则匹配地址为 1xxxxxx0
即，只要主机送出的地址数据中的 bit0 为 0 且 bit7 为 1 就可以和本机地址相匹配

再例如
S5ADDR ＝ 11001010
S5ADEN ＝ 00001111
则匹配地址为 xxxx1010
即，只要主机送出的地址数据中的低 4 位为 1010 就可以和本机地址相匹配，而高 4 为被忽略，可以为任意值。

主机可以使用广播地址（FFH）同时选中所有的从机来进行通讯。

19.6.12 串口 5 同步模式控制寄存器 1（USART5CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART5CR1	7EF748H	S5LINEN	S5DORD	S5CLKEN	S5SPMOD	S5SPIEN	S5SPSLV	S5CPOL	S5CPHA

S5LINEN：LIN 模式使能位

0：禁止 LIN 模式

1：使能 LIN 模式

S5DORD：SPI 模式数据位发送/接收的顺序

0：先发送/接收数据的高位（MSB）

1：先发送/接收数据的低位（LSB）

S5CLKEN：SmartCard 模式时钟输出控制位

0：禁止时钟输出

1：使能时钟输出

S5SPMOD：SPI 模式使能位

0：禁止 SPI 模式

1：使能 SPI 模式

S5SPIEN：SPI 使能位

0：禁止 SPI 功能

1：使能 SPI 功能

当 USART5 需要工作在 SPI 模式时，必须先将 SPI 相关的模式寄存器、速度寄存器、时钟极性寄存器设置完成后，最后再设置 SPIEN 寄存器。

S5SPSLV：SPI 从机模式使能位

0：SPI 为主机模式

1：SPI 为从机模式

S5CPOL：SPI 时钟极性控制

0：SCLK 空闲时为低电平，SCLK 的前时钟沿为上升沿，后时钟沿为下降沿

1：SCLK 空闲时为高电平，SCLK 的前时钟沿为下降沿，后时钟沿为上升沿

S5CPHA：SPI 时钟相位控制

0：数据 SS 管脚为低电平驱动第一位数据并在 SCLK 的后时钟沿改变数据，前时钟沿采样数据

1：数据在 SCLK 的前时钟沿驱动，后时钟沿采样

19.6.13 串口 5 同步模式控制寄存器 2（USART5CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART5CR2	7EF749H	S5IREN	S5IRLP	S5SCEN	S5NACK	S5HDSEL	S5PCEN	S5PS	S5PE

S5IREN：IrDA 模式使能位

0：禁止 IrDA 模式

1：使能 IrDA 模式

S5IRLP：IrDA 低电模式控制位

0：IrDA 为普通模式

1：IrDA 为低电模式

S5SCEN：SmartCard 模式使能位

0：禁止 SmartCard 模式

1：使能 SmartCard 模式

S5NACK：SmartCard 模式 NACK 输出使能位

0：禁止输出 NACK 信号

1：使能输出 NACK 信号

S5HDSEL：单线半双工模式使能位

0：禁止单线半双工模式

1：使能单线半双工模式

S5PCEN：硬件自动产生校验位控制使能

0：禁止硬件自动产生校验位（串口的校验位为 S5TB8 设置的值）

1：使能硬件自动产生校验位

S5PS：硬件校验位模式选择

0：硬件根据 S5BUF 的值自动产生偶校验位

1：硬件根据 S5BUF 的值自动产生奇校验位

S5PE：校验位错误标志（必须软件清零）

0：无检验错误

1：有校验错误（串口接收 DMA 过程中如果发生接收数据校验位错误，DMA 不会停止，但校验位错误标志会一直保持直到 DMA 完成）

19.6.14 串口 5 同步模式控制寄存器 3（USART5CR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART5CR3	7EF74AH	S5IrDA_LPBAUD[7:0]							

S5IrDA_LPBAUD：IrDA 低电模式波特率控制寄存器

IrDA_LPBAUD[7:0]	IrDA 低电模式波特率
0	SYSclk/16/256
1	SYSclk/16/1
2	SYSclk/16/2
...	...
255	SYSclk/16/255

19.6.15 串口 5 同步模式控制寄存器 4（USART5CR4）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART5CR4	7EF74BH	-	-	-	-	S5SCCKS[1:0]		S5SPICKS[1:0]	

S5SCCKS[1:0]：SmartCard 模式时钟选择

SCCKS[1:0]	SmartCard 模式时钟
00	SYSclk/64
01	SYSclk/32
10	SYSclk/16
11	SYSclk/8

S5SPICKS[1:0]：SPI 模式时钟选择

SPICKS[1:0]	SPI 模式时钟
00	SYSclk/4
01	SYSclk/8
10	SYSclk/16
11	SYSclk/2

19.6.16 串口 5 同步模式控制寄存器 5（USART5CR5）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART5CR5	7EF74CH	S5BRKDET	S5HDRER	S5SLVEN	S5ASYNC	S5TXCF	S5SENDBK	S5HDRDET	S5SYNC

S5BRKDET：LIN 从模式间隔场（BREAK）检测标志

0：未检测到 LIN 间隔场

1：检测到 LIN 间隔场，需要软件清零

S5HDRER：LIN 从模式报文头（HEADER）错误

0：未检测到 LIN 报文头错误

1：检测到 LIN 报文头错误，需要软件清零

S5SLVEN：LIN 从机模式使能位

0：LIN 为主机模式

1：LIN 为从机模式

S5ASYNC：LIN 自动同步功能使能位

0：禁止自动同步

1：使能自动同步

【注意】：从机使能自动同步功能后，如果从机跟主机波特率存在小范围误差，从机收到主机同步信号后，会自动将从机波特率设置成与主机相同的波特率。

例如，主机波特率 9600：

从机使能自动同步时，如果从机波特率在 9100～10000 附近，收到主机同步信号后，从机会自动将本身波特率设置为 9600。误差自动同步范围 -5.2%～+4.16%左右。

S5TXCF：传输冲突标志位。单线半双工模式、LIN 模式和 SmartCard 模式有效

0：未检测到传输冲突（发送的数据与接收的数据相同）

1：检测到传输冲突（发送的数据与接收的数据不同）

S5SENDBK：发送间隔场。软件写 1 触发发送间隔场，发送完成后硬件自动清 0

S5HDRDET：LIN 从模式报文头（HEADER）检测标志

0：未检测到 LIN 报文头

1：检测到 LIN 报文头，需要软件清零

S5SYNC：LIN 从模式同步场检测标志。

正确分析到同步场后，硬件将 SYNC 标志为置 1。在接收标志符场时硬件会自动清零 SYNC

19.6.17 串口 5 同步模式保护时间寄存器（USART5GTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART5GTR	7EF74DH								

USART5GTR 寄存器仅在 SmartCard 模式有效。该寄存器是根据波特率时钟数给出保护时间值。S4TI 标志在 SmartCard 发送的数据位等于 USART5GTR 寄存器所设置的保护时间值时被硬件置 1。注：USART5GTR 寄存器的值应大于 11

19.6.18 串口 5 同步模式波特率寄存器（USART5BR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART5BRH	7EF74EH	USART5BR[15:8]							
USART5BRL	7EF74FH	USART5BR[7:0]							

LIN 模式、IrDA 模式和 SmartCard 模式时的波特率计算公式

$$\text{同步波特率} = \frac{\text{SYSclk}}{16 * \text{USART5BR[15:0]}}$$

19.6.19 串口 5 接收超时控制寄存器（UR5TOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR5TOCR	7EF744H	UR5ENTO	UR5ENTOI	UR5SCALE	-	-	-	-	-

UR5ENTO：串口 5 接收超时功能控制位

0：禁止串口 5 接收超时功能

1：使能串口 5 接收超时功能

UR5ENTOI：串口 5 接收超时中断控制位

0：禁止串口 5 接收超时中断

1：使能串口 5 接收超时中断

UR5SCALE：串口 5 超时计数时钟源选择

0：串口数据位率（波特率）

1：系统时钟

19.6.20 串口 5 超时状态寄存器（UR5TOSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR5TOSR	7EF745H	UR5CTOIF	-	-	-	-	-	-	UR5TOIF

UR5CTOIF：写“1”清除串口超时中断标志位 TOIF。（只写）

UR5TOIF：串口 5 超时中断请求标志位。（只读）

当发生串口 5 超时时，TOIF 会被硬件置“1”。如果 ENTOI 为 1 时会产生串口中断，中断入口地址为原串口 5 的中断入口地址。标志位需要软件向 CTOIF 位写“1”清零

19.6.21 串口 5 超时长度控制寄存器（UR5TOTE/H/L）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR5TOTE	7EF743H	UR5TM[23:16]							
UR5TOTH	7EF746H	UR5TM[15:8]							
UR5TOTL	7EF747H	UR5TM[7:0]							

UR5TM[23:0]：串口 5 超时时间控制位。

当串口 5 处于接收空闲状态时，内部超时计数器根据 SCALE 寄存器所选择的时钟源进行计数，当计数时间达到 UR5TM 所设置的超时时间时，便会产生超时中断。当串口 5 接收数据完成时，复位内部超时计数器，重新进行超时计数。

注：

1、如果需要使能接收超时中断功能，则 TM 不可设置为 0，且 UR5TOTL、UR5TOTH、UR5TOTE 寄存器的设置必须先设置 UR5TOTL 和 UR5TOTH，最后设置 UR5TOTE

2、必须使能串口 5 接收才有接收超时功能

3、接收超时功能必须在正确接收到一字节数据后才能触发

4、正在接收过程中，无接收超时功能

19.7 串口 6（同步/异步串口 USART6）

19.7.1 串口 6 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX2	7EFD6AH	-	-	LIN2_S[1:0]		**S6_S[1:0]**		S5_S[1:0]	

S6_S[1:0]：串口 6 功能脚选择位

S6_S[1:0]	RxD6	TxD6
00	P0.6	P0.7
01	P6.2	P6.3
10	P3.2	P3.3
11	P2.6	P2.7

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX3	7EFD6BH	**S6SPI_S[1:0]**		S5SPI_S[1:0]		S4SPI_S[1:0]		S3SPI_S[1:0]	

S6SPI_S[1:0]：USART6 的 SPI 功能脚选择位

S6SPI_S[1:0]	S6SS	S6MOSI	S6MISO	S6SCLK
00	P2.2	P2.3	P2.4	P2.5
01	P7.0	P7.1	P7.2	P7.3
10	P8.4	P8.5	P8.6	P8.7
11	P9.4	P9.5	P9.6	P9.7

19.7.2 串口 6 相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S6CON	串口 6 控制寄存器	7EF770H	S6SM0/FE	S6SM1	S6SM2	S6REN	S6TB8	S6RB8	S6TI	S6RI	0000,0000
S6BUF	串口 6 数据寄存器	7EF771H	S6BUF[7:0]								0000,0000
S6INTR	串口 6 中断寄存器	7EF772H	ES6	-	PS6[1:0]		-	-	-	-	0x00,xxxx

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S6CFG	串口 6 配置寄存器	7EF79DH	S6MOD	S6MOD0	S6M0x6	-	-	-	-	S6BRT	000x,xxx0
S6ADDR	串口 6 从机地址寄存器	7EF79EH	S6ADDR[7:0]								0000,0000
S6ADEN	串口 6 从机地址屏蔽寄存器	7EF79FH	S6ADEN[7:0]								0000,0000
USART6CR1	串口 6 控制寄存器 1	7EF778H	S6LINEN	S6DORD	S6CLKEN	S6SPMOD	S6SPIEN	S6SPSLV	S6CPOL	S6CPHA	0000,0000
USART6CR2	串口 6 控制寄存器 2	7EF779H	S6IREN	S6IRLP	S6SCEN	S6NACK	S6HDSEL	S6PCEN	S6PS	S6PE	0000,0000
USART6CR3	串口 6 控制寄存器 3	7EF77AH	S6IrDA_LPBAUD[7:0]								0000,0111
USART6CR4	串口 6 控制寄存器 4	7EF77BH	-	-	-	-	S6SCCKS[1:0]		S6SPICKS[1:0]		xxxx,0000
USART6CR5	串口 6 控制寄存器 5	7EF77CH	S6BRKDET	S6HDRER	S6SLVEN	S6ASYNC	S6TXCF	S6SENDBK	S6HDRDET	S6SYNC	0000,0000
USART6GTR	串口 6 保护时间寄存器	7EF77DH	USART6GTR[7:0]								0000,0000
USART6BRH	串口 6 波特率寄存器	7EF77EH	USART6BR[15:8]								0000,0000
USART6BRL	串口 6 波特率寄存器	7EF77FH	USART6BR[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
UR6TOCR	串口 6 接收超时控制寄存器	7EF774H	UR6ENTO	UR6ENTOI	UR6SCALE	-	-	-	-	-	000x,xxxx
UR6TOSR	串口 6 接收超时状态寄存器	7EF775H	-	-	-	-	-	-	-	UR6TOIF	xxxx,xxx0
UR6TOTH	串口 6 接收超时长度寄存器	7EF776H	UR6TM[15:8]								0000,0000
UR6TOTL	串口 6 接收超时长度寄存器	7EF777H	UR6TM[7:0]								0000,0000
UR6TOTE	串口 6 接收超时长度寄存器	7EF773H	UR6TM[23:16]								0000,0000

19.7.3 串口 6 控制寄存器（S6CON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S6CON	7EF770H	S6SM0/FE	S6SM1	S6SM2	S6REN	S6TB8	S6RB8	S6TI	S6RI

S6SM0/FE：当S6CFG寄存器中的S6MOD0位为1时，该位为帧错误检测标志位。当串口6在接收过程中检测到一个无效停止位时，通过串口6接收器将该位置1，必须由软件清零。当S6CFG寄存器中的S6MOD0位为0时，该位和S6SM1一起指定串口6的通信工作模式，如下表所示：

S6SM0	S6SM1	串口6工作模式	功能说明
0	0	模式0	同步移位串行方式
0	1	模式1	可变波特率8位数据方式
1	0	模式2	固定波特率9位数据方式
1	1	模式3	可变波特率9位数据方式

当使能 FE 功能后，串口的工作模式只能使用 SM1 进行设置，且只能设置为模式 0 或者模式 1。FE 当作帧错误标志位时，此标志位必须软件写 0 清零。

S6SM2：允许模式 2 或模式 3 多机通信控制位。当串口 6 使用模式 2 或模式 3 时，如果 S6SM2 位为 1 且 S6REN 位为 1，则接收机处于地址帧筛选状态。此时可以利用接收到的第 9 位（即 S6RB8）来筛选地址帧，若 S6RB8=1，说明该帧是地址帧，地址信息可以进入 S6BUF，并使 S6RI 为 1，进而在中断服务程序中再进行地址号比较；若 S6RB8=0，说明该帧不是地址帧，应丢掉且保持 S6RI=0。在模式 2 或模式 3 中，如果 S6SM2 位为 0 且 S6REN 位为 1，接收收机处于地址帧筛选被禁止状态，不论收到的 S6RB8 为 0 或 1，均可使接收到的信息进入 S6BUF，并使 S6RI=1，此时 S6RB8 通常为校验位。模式 1 和模式 0 为非多机通信方式，在这两种方式时，S6SM2 应设置为 0。

S6REN：允许/禁止串口接收控制位

0：禁止串口接收数据

1：允许串口接收数据

S6TB8：当串口 6 使用模式 2 或模式 3 时，S6TB8 为要发送的第 9 位数据，按需要由软件置位或清 0。在模式 0 和模式 1 中，该位不用。

S6RB8：当串口 6 使用模式 2 或模式 3 时，S6RB8 为接收到的第 9 位数据，一般用作校验位或者地址帧/数据帧标志位。在模式 0 和模式 1 中，该位不用。

S6TI：串口 6 发送中断请求标志位。在模式 0 中，当串口发送数据第 8 位结束时，由硬件自动将 S6TI 置 1，向主机请求中断，响应中断后 S6TI 必须用软件清零。在其他模式中，则在停止位开始发送时由硬件自动将 S6TI 置 1，向 CPU 发请求中断，响应中断后 S6TI 必须用软件清零。

S6RI：串口 6 接收中断请求标志位。在模式 0 中，当串口接收第 8 位数据结束时，由硬件自动将 S6RI 置 1，向主机请求中断，响应中断后 S6RI 必须用软件清零。在其他模式中，串行接收到停止位的中间时刻由硬件自动将 S6RI 置 1，向 CPU 发中断申请，响应中断后 S6RI 必须由软件清零。

19.7.4 串口 6 数据寄存器（S6BUF）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S6BUF	7EF771H								

S6BUF：串口 1 数据接收/发送缓冲区。S6BUF 实际是 2 个缓冲器，读缓冲器和写缓冲器，两个操作分别对应两个不同的寄存器，1 个是只写寄存器（写缓冲器），1 个是只读寄存器（读缓冲器）。对 S6BUF 进行读操作，实际是读取串口接收缓冲区，对 S6BUF 进行写操作则是触发串口开始发送数据。

19.7.5 串口 6 中断控制寄存器（S6INTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S6INTR	7EF772H	ES6	-	PS6[1:0]		-	-	-	-

ES6：串口 6 中断使能控制位

0：禁止串口 6 中断

1：允许串口 6 中断

PS6[1:0]：串口 6 中断优先级控制位

00：串口 6 中断优先级为 0 级（最低级）

01：串口 6 中断优先级为 1 级（较低级）

10：串口 6 中断优先级为 2 级（较高级）

11：串口 6 中断优先级为 3 级（最高级）

19.7.6 串口 6 配置寄存器（S6CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S6CFG	7EF79DH	S6MOD	S6MOD0	S6M0x6					S6BRT

S6MOD：串口 6 波特率控制位

0：串口 6 之模式 0/1/2/3 的波特率都不加倍

1：串口 6 之模式 2 的波特率加倍

S6MOD0：帧错误检测控制位

0：无帧错检测功能

1：使能帧错误检测功能。此时 S6CON 的 S6SM0/FE 为 FE 功能，即为帧错误检测标志位。

S6M0x6：串口 6 模式 0 的通讯速度控制

0：串口 6 模式 0 的波特率不加倍，固定为 Fosc/12

1：串口 6 模式 0 的波特率 6 倍速，即固定为 Fosc/12*6 ＝ Fosc/2

S6BRT：串口 6 波特率发生器选择位

0：选择定时器 2 作为波特率发生器（**默认值**）

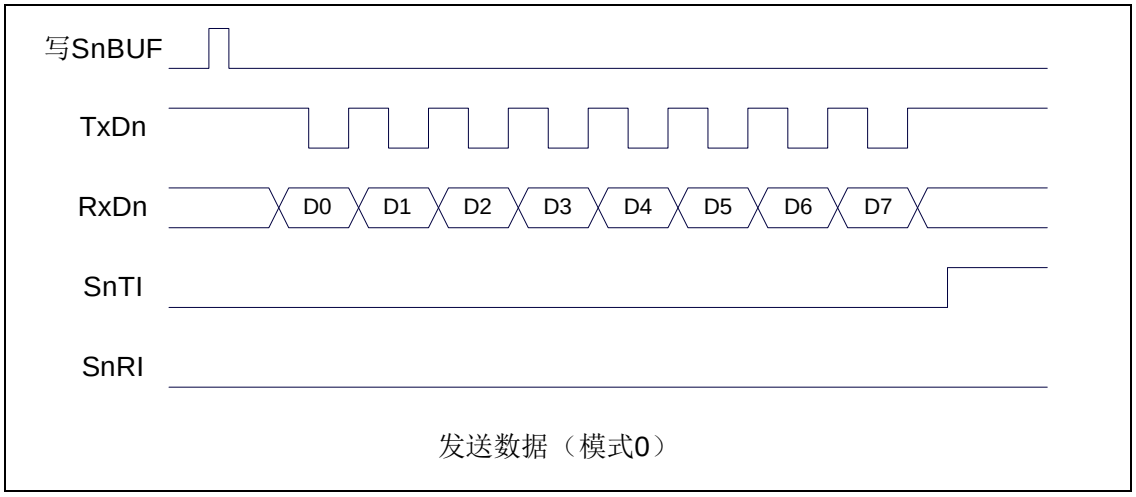
1：选择定时器 6 作为波特率发生器

19.7.7 串口 6 模式 0，模式 0 波特率计算公式

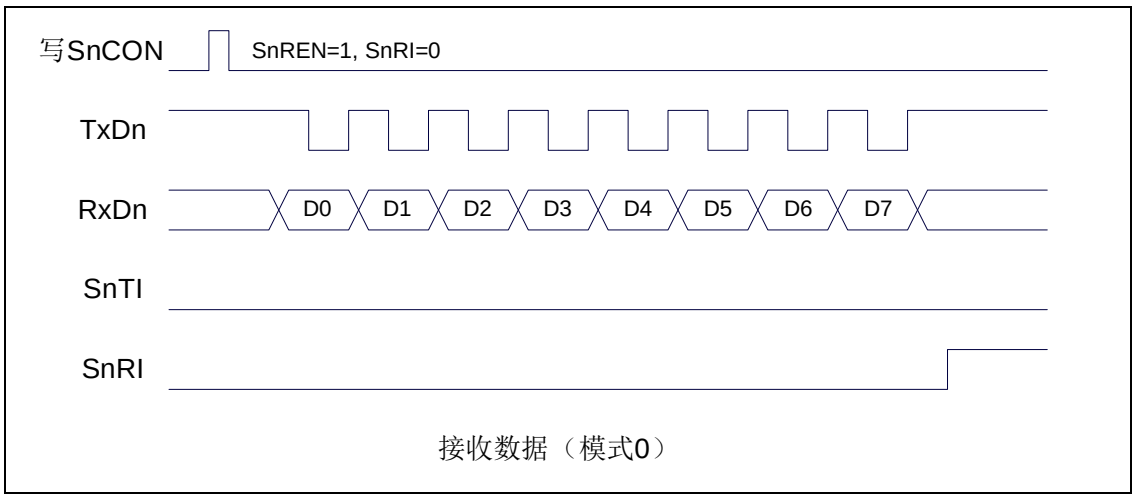
当串口 6 选择工作模式为模式 0 时，串行通信接口工作在同步移位寄存器模式，当串口 6 模式 0 的通信速度设置位 S6M0x6 为 0 时，其波特率固定为系统时钟时钟的 12 分频（SYSclk/12）；当设置 S6M0x6 为 1 时，其波特率固定为系统时钟频率的 2 分频（SYSclk/2）。RxD6 为串行通讯的数据口，TxD6 为同步移位脉冲输出脚，发送、接收的是 8 位数据，低位在先。

模式 0 的发送过程：当主机执行将数据写入发送缓冲器 S6BUF 指令时启动发送，串口 6 即将 8 位数据以 SYSclk/12 或 SYSclk/2（由 S6M0x6 确定是 12 分频还是 2 分频）的波特率从 RxD6 管脚输出(从低位到高位)，发送完中断标志 S6TI 置 1，TxD6 管脚输出同步移位脉冲信号。当写信号有效后，相隔一个时钟，发送控制端 SEND 有效(高电平)，允许 RxD6 发送数据，同时允许 TxD6 输出同步移位脉冲。一帧(8 位)数据发送完毕时，各控制端均恢复原状态，只有 S6TI 保持高电平，呈中断申请状态。在再次发送数据前，必须用软件将 S6TI 清 0。

模式 0 的接收过程：首先将接收中断请求标志 S6RI 清零并置位允许接收控制位 S6REN 时启动模式 0 接收过程。启动接收过程后，RxD6 为串行数据输入端，TxD6 为同步脉冲输出端。串行接收的波特率为 SYSclk/12 或 SYSclk/2（由 S6M0x6 确定是 12 分频还是 2 分频）。当接收完成一帧数据（8 位）后，控制信号复位，中断标志 S6RI 被置 1，呈中断申请状态。当再次接收时，必须通过软件将 S6RI 清 0



发送数据（模式0）



接收数据（模式0）

工作于模式 0 时，必须清 0 多机通信控制位 S6SM2，使之不影响 S6TB8 位和 S6RB8 位。由于波特率固定为 SYSclk/12 或 SYSclk/2，无需定时器提供，直接由单片机的时钟作为同步移位脉冲。

串口 6 模式 0 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

S6M0x6	波特率计算公式
0	$\text{波特率} = \frac{\text{SYSclk}}{12}$
1	$\text{波特率} = \frac{\text{SYSclk}}{2}$

19.7.8 串口 6 模式 1，模式 1 波特率计算公式

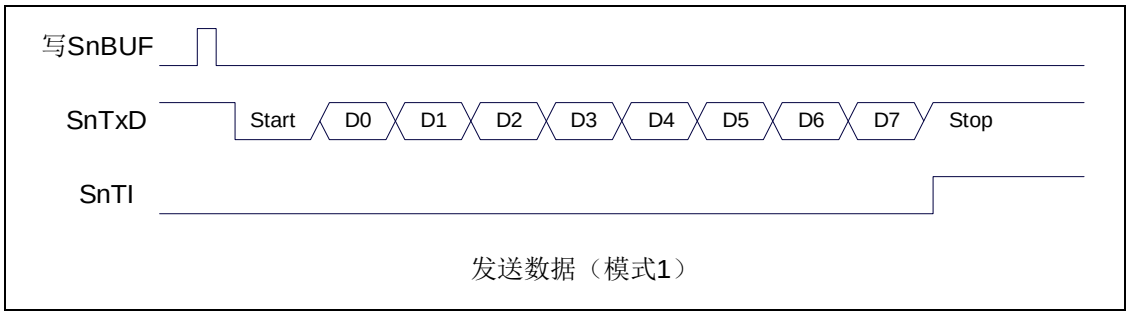
当软件设置 S6CON 的 S6SM0、S6SM1 为“01”时，串口 6 则以模式 1 进行工作。此模式为 8 位 UART 格式，一帧信息为 10 位：1 位起始位，8 位数据位（低位在先）和 1 位停止位。波特率可变，即可根据需要进行设置波特率。TxD6 为数据发送口，RxD6 为数据接收口，串口 6 全双工接受/发送。

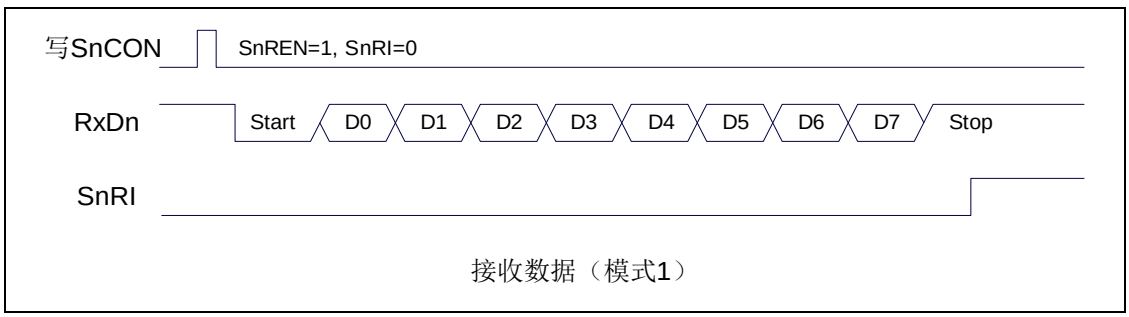
模式 1 的发送过程：串行通信模式发送时，数据由串行发送端 TxD6 输出。当主机执行一条写 S6BUF 的指令就启动串行通信的发送，写“S6BUF”信号还把“1”装入发送移位寄存器的第 9 位，并通知 TX 控制单元开始发送。移位寄存器将数据不断右移送 TxD6 端口发送，在数据的左边不断移入“0”作补充。当数据的最高位移到移位寄存器的输出位置，紧跟其后的是第 9 位“1”，在它的左边各位全为“0”，这个状态条件，使 TX 控制单元作最后一次移位输出，然后使允许发送信号“SEND”失效，完成一帧信息的发送，并置位中断请求位 S6TI，即 S6TI=1，向主机请求中断处理。

模式 1 的接收过程：当软件置位接收允许标志位 S6REN，即 S6REN=1 时，接收器便对 RxD6 端口的信号进行检测，当检测到 RxD6 端口发送从“1”→“0”的下降沿跳变时就启动接收器准备接收数据，并立即复位波特率发生器的接收计数器，将 1FFH 装入移位寄存器。接收的数据从接收移位寄存器的右边移入，已装入的 1FFH 向左边移出，当起始位"0"移到移位寄存器的最左边时，使 RX 控制器作最后一次移位，完成一帧的接收。若同时满足以下两个条件：

- S6RI=0；
- S6SM2=0 或接收到的停止位为 1。

则接收到的数据有效，实现装载入 S6BUF，停止位进入 S6RB8，S6RI 标志位被置 1，向主机请求中断，若上述两条件不能同时满足，则接收到的数据作废并丢失，无论条件满足与否，接收器重又检测 RxD6 端口上的"1"→"0"的跳变，继续下一帧的接收。接收有效，在响应中断后，S6RI 标志位必须由软件清 0。通常情况下，串行通信工作于模式 1 时，S6SM2 设置为"0"。





串口 6 的波特率是可变的，其波特率固定由定时器 2 或定时器 6 产生。当定时器采用 1T 模式时（12 倍速），相应的波特率的速度也会相应提高 12 倍。

串口 6 模式 1 的波特率计算公式如下表所示：（SYSclk 为系统工作频率）

选择定时器	定时器速度	重装值计算公式	波特率
定时器2 或 定时器6	1T	定时器重载值 = 65536 - $\frac{SYSclk}{4\times 波特率}$	波特率 = $\frac{SYSclk}{4\times(65536 - 定时器重装数)}$
	12T	定时器重载值 = 65536 - $\frac{SYSclk}{12\times 4\times 波特率}$	波特率 = $\frac{SYSclk}{12\times 4\times(65536 - 定时器重装数)}$

下面为常用频率与常用波特率所对应定时器的重载值

频率 (MHz)	波特率	定时器	
		1T 模式	12T 模式
11.0592	115200	FFE8H	FFFEH
	57600	FFD0H	FFFCH
	38400	FFB8H	FFFAH
	19200	FF70H	FFF4H
	9600	FEE0H	FFE8H
18.432	115200	FFD8H	-
	57600	FFB0H	-
	38400	FF88H	FFF6H
	19200	FF10H	FFECH
	9600	FE20H	FFD8H
22.1184	115200	FFD0H	FFFCH
	57600	FFA0H	FFF8H
	38400	FF70H	FFF4H
	19200	FEE0H	FFE8H
	9600	FDC0H	FFD0H

19.7.9 串口 6 模式 2，模式 2 波特率计算公式

当 S6SM0、S6SM1 两位为 10 时，串口 6 工作在模式 2。串口 6 工作模式 2 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S6CON 中的 S6TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S6TB8（S6TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S6CON 的 S6RB8。TxD6 为发送端口，RxD6 为接收端口，以全双工模式进行接收/发送。

模式 2 的波特率固定为系统时钟的 64 分频或 32 分频（取决于 S6CFG 中 S6MOD 的值）

串口 6 模式 2 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

SMOD	波特率计算公式
0	波特率 $= \frac{SYSclk}{64}$
1	波特率 $= \frac{SYSclk}{32}$

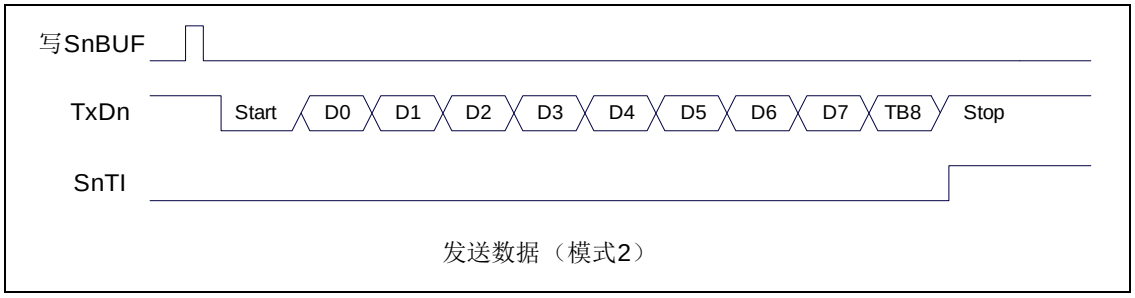
模式 2 和模式 1 相比，除波特率发生源略有不同，发送时由 S6TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收/发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

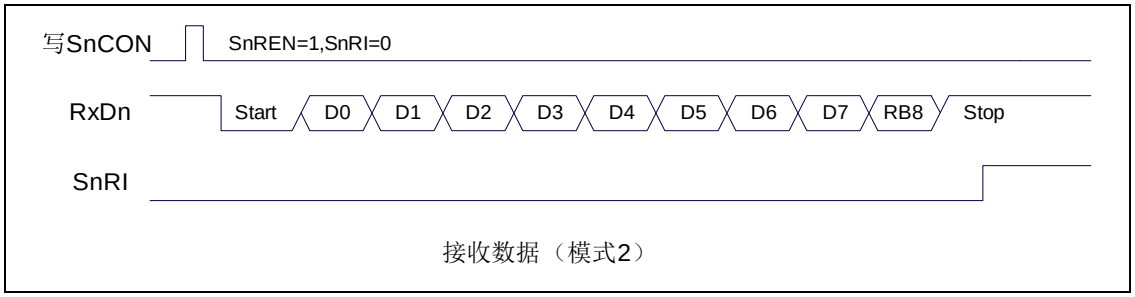
- S6RI=0
- S6SM2=0 或者 S6SM2=1 且接收到的第 9 数据位 S6RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S6BUF 和 S6RB8 中，S6RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S6RI。无论上述条件满足与否，接收器又重新开始检测 RxD6 输入端口的跳变信息，接收下一帧的输入信息。在模式 2 中，接收到的停止位与 S6BUF、S6RB8 和 S6RI 无关。

通过软件对 S6CON 中的 S6SM2、S6TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式2）



接收数据（模式2）

19.7.10 串口 6 模式 3，模式 3 波特率计算公式

当 S6SM0、S6SM1 两位为 11 时，串口 6 工作在模式 3。串行通信模式 3 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S6CON 中的 S6TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S6TB8（S6TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S6CON 的 S6RB8。TxD6 为发送端口，RxD6 为接收端口，以全双工模式进行接收/发送。

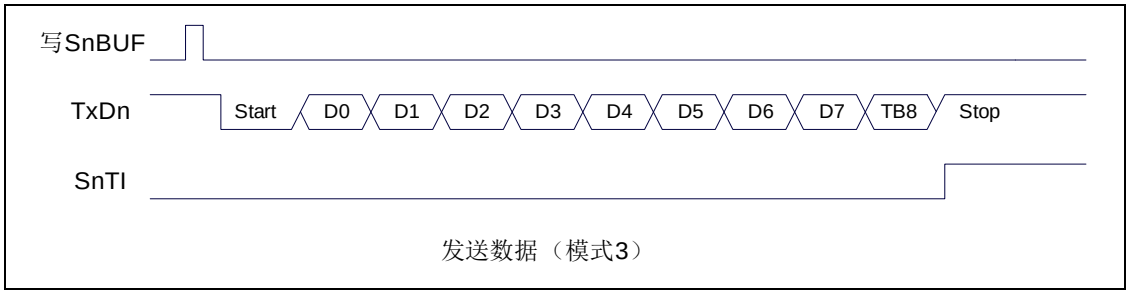
模式 3 和模式 1 相比，除发送时由 S6TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收‘发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

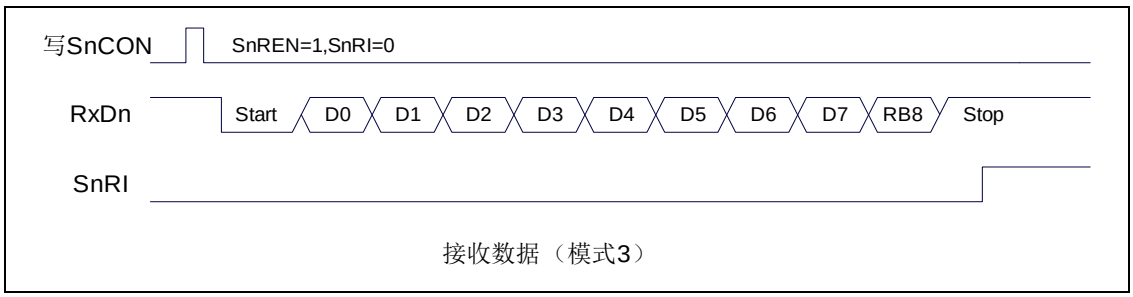
- S6RI=0
- S6SM2=0 或者 S6SM2=1 且接收到的第 9 数据位 S6RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S6BUF 和 S6RB8 中，S6RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S6RI。无论上述条件满足与否，接收器又重新开始检测 RxD6 输入端口的跳变信息，接收下一帧的输入信息。在模式 3 中，接收到的停止位与 S6BUF、S6RB8 和 S6RI 无关。

通过软件对 S6CON 中的 S6SM2、S6TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式3）



接收数据（模式3）

串口 6 模式 3 的波特率计算公式与模式 1 是完全相同的。请参考模式 1 的波特率计算公式。

19.7.11 串口 6 自动地址识别，从机地址控制寄存器（S6ADDR，S6ADEN）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S6ADDR	7EF79EH								
S6ADEN	7EF79FH								

S6ADDR：从机地址寄存器

S6ADEN：从机地址屏蔽位寄存器

自动地址识别功能典型应用在多机通讯领域，其主要原理是从机系统通过硬件比较功能来识别来自于主机串口 6 数据流中的地址信息，通过寄存器 S6ADDR 和 S6ADEN 设置的本机的从机地址，硬件自动对从机地址进行过滤，当来自于主机的从机地址信息与本机所设置的从机地址相匹配时，硬件产生串口 6 中断；否则硬件自动丢弃串口 6 数据，而不产生中断。当众多处于空闲模式的从机链接在一起时，只有从机地址相匹配的从机才会从空闲模式唤醒，从而可以大大降低从机 MCU 的功耗，即使从机处于正常工作状态也可避免不停地进入串口 6 中断而降低系统执行效率。

要使用串口 6 的自动地址识别功能，首先需要将参与通讯的 MCU 的串口 6 通讯模式设置为模式 1，并开启从机的 S6CON 的 S6SM2 位。对于串口 6 模式 1 的 9 位数据位中，第 9 位数据（存放在 S6RB8 中）为地址/数据的标志位，当第 9 位数据为 1 时，表示前面的 8 位数据（存放在 S6BUF 中）为地址信息。当 S6SM2 被设置为 1 时，从机 MCU 会自动过滤掉非地址数据（第 9 位为 0 的数据），而对 S6BUF 中的地址数据（第 9 位为 1 的数据）自动与 S6ADDR 和 S6ADEN 所设置的本机地址进行比较，若地址相匹配，则会将 S6RI 置“1”，并产生中断，否则不予处理本次接收的串口 6 数据。

从机地址的设置是通过 S6ADDR 和 S6ADEN 两个寄存器进行设置的。S6ADDR 为从机地址寄存器，里面存放本机的从机地址。S6ADEN 为从机地址屏蔽位寄存器，用于设置地址信息中的忽略位，设置方法如下：

例如

S6ADDR ＝ 11001010

S6ADEN ＝ 10000001

则匹配地址为 1xxxxxx0

即，只要主机送出的地址数据中的 bit0 为 0 且 bit7 为 1 就可以和本机地址相匹配

再例如

S6ADDR ＝ 11001010

S6ADEN ＝ 00001111

则匹配地址为 xxxx1010

即，只要主机送出的地址数据中的低 4 位为 1010 就可以和本机地址相匹配，而高 4 为被忽略，可以为任意值。

主机可以使用广播地址（FFH）同时选中所有的从机来进行通讯。

19.7.12 串口 6 同步模式控制寄存器 1（USART6CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART6CR1	7EF778H	S6LINEN	S6DORD	S6CLKEN	S6SPMOD	S6SPIEN	S6SPSLV	S6CPOL	S6CPHA

S6LINEN：LIN 模式使能位

0：禁止 LIN 模式

1：使能 LIN 模式

S6DORD：SPI 模式数据位发送/接收的顺序

0：先发送/接收数据的高位（MSB）

1：先发送/接收数据的低位（LSB）

S6CLKEN：SmartCard 模式时钟输出控制位

0：禁止时钟输出

1：使能时钟输出

S6SPMOD：SPI 模式使能位

0：禁止 SPI 模式

1：使能 SPI 模式

S6SPIEN：SPI 使能位

0：禁止 SPI 功能

1：使能 SPI 功能

当 USART6 需要工作在 SPI 模式时，必须先将 SPI 相关的模式寄存器、速度寄存器、时钟极性寄存器设置完成后，最后再设置 SPIEN 寄存器。

S6SPSLV：SPI 从机模式使能位

0：SPI 为主机模式

1：SPI 为从机模式

S6CPOL：SPI 时钟极性控制

0：SCLK 空闲时为低电平，SCLK 的前时钟沿为上升沿，后时钟沿为下降沿

1：SCLK 空闲时为高电平，SCLK 的前时钟沿为下降沿，后时钟沿为上升沿

S6CPHA：SPI 时钟相位控制

0：数据 SS 管脚为低电平驱动第一位数据并在 SCLK 的后时钟沿改变数据，前时钟沿采样数据

1：数据在 SCLK 的前时钟沿驱动，后时钟沿采样

19.7.13 串口 6 同步模式控制寄存器 2（USART6CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART6CR2	7EF779H	S6IREN	S6IRLP	S6SCEN	S6NACK	S6HDSEL	S6PCEN	S6PS	S6PE

S6IREN：IrDA 模式使能位

0：禁止 IrDA 模式

1：使能 IrDA 模式

S6IRLP：IrDA 低电模式控制位

0：IrDA 为普通模式

1：IrDA 为低电模式

S6SCEN：SmartCard 模式使能位

0：禁止 SmartCard 模式

1：使能 SmartCard 模式

S6NACK：SmartCard 模式 NACK 输出使能位

0：禁止输出 NACK 信号

1：使能输出 NACK 信号

S6HDSEL：单线半双工模式使能位

0：禁止单线半双工模式

1：使能单线半双工模式

S6PCEN：硬件自动产生校验位控制使能

0：禁止硬件自动产生校验位（串口的校验位为 S6TB8 设置的值）

1：使能硬件自动产生校验位

S6PS：硬件校验位模式选择

0：硬件根据 S6BUF 的值自动产生偶校验位

1：硬件根据 S6BUF 的值自动产生奇校验位

S6PE：校验位错误标志（必须软件清零）

0：无检验错误

1：有校验错误（串口接收 DMA 过程中如果发生接收数据校验位错误，DMA 不会停止，但校验位错误标志会一直保持直到 DMA 完成）

19.7.14 串口 6 同步模式控制寄存器 3（USART6CR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART6CR3	7EF77AH	S6IrDA_LPBAUD[7:0]							

S6IrDA_LPBAUD：IrDA 低电模式波特率控制寄存器

IrDA_LPBAUD[7:0]	IrDA 低电模式波特率
0	SYSclk/16/256
1	SYSclk/16/1
2	SYSclk/16/2
...	...
255	SYSclk/16/255

19.7.15 串口 6 同步模式控制寄存器 4（USART6CR4）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART6CR4	7EF77BH	-	-	-	-	S6SCCKS[1:0]		S6SPICKS[1:0]	

S6SCCKS[1:0]：SmartCard 模式时钟选择

SCCKS[1:0]	SmartCard 模式时钟
00	SYSclk/64
01	SYSclk/32
10	SYSclk/16
11	SYSclk/8

S6SPICKS[1:0]：SPI 模式时钟选择

SPICKS[1:0]	SPI 模式时钟
00	SYSclk/4
01	SYSclk/8
10	SYSclk/16
11	SYSclk/2

19.7.16 串口 6 同步模式控制寄存器 5（USART6CR5）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART6CR5	7EF77CH	S6BRKDET	S6HDRER	S6SLVEN	S6ASYNC	S6TXCF	S6SENDBK	S6HDRDET	S6SYNC

S6BRKDET：LIN 从模式间隔场（BREAK）检测标志

0：未检测到 LIN 间隔场

1：检测到 LIN 间隔场，需要软件清零

S6HDRER：LIN 从模式报文头（HEADER）错误

0：未检测到 LIN 报文头错误

1：检测到 LIN 报文头错误，需要软件清零

S6SLVEN：LIN 从机模式使能位

0：LIN 为主机模式

1：LIN 为从机模式

S6ASYNC：LIN 自动同步功能使能位

0：禁止自动同步

1：使能自动同步

【注意】：从机使能自动同步功能后，如果从机跟主机波特率存在小范围误差，从机收到主机同步信号后，会自动将从机波特率设置成与主机相同的波特率。

例如，主机波特率 9600：

从机使能自动同步时，如果从机波特率在 9100～10000 附近，收到主机同步信号后，从机会自动将本身波特率设置为 9600。误差自动同步范围 -5.2%～+4.16%左右。

S6TXCF：传输冲突标志位。单线半双工模式、LIN 模式和 SmartCard 模式有效

0：未检测到传输冲突（发送的数据与接收的数据相同）

1：检测到传输冲突（发送的数据与接收的数据不同）

S6SENDBK：发送间隔场。软件写 1 触发发送间隔场，发送完成后硬件自动清 0

S6HDRDET：LIN 从模式报文头（HEADER）检测标志

0：未检测到 LIN 报文头

1：检测到 LIN 报文头，需要软件清零

S6SYNC：LIN 从模式同步场检测标志。

正确分析到同步场后，硬件将 SYNC 标志为置 1。在接收标志符场时硬件会自动清零 SYNC

19.7.17 串口 6 同步模式保护时间寄存器（USART6GTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART6GTR	7EF77DH								

USART6GTR 寄存器仅在 SmartCard 模式有效。该寄存器是根据波特率时钟数给出保护时间值。S4TI 标志在 SmartCard 发送的数据位等于 USART6GTR 寄存器所设置的保护时间值时被硬件置 1。注：USART6GTR 寄存器的值应大于 11

19.7.18 串口 6 同步模式波特率寄存器（USART6BR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART6BRH	7EF77EH	USART6BR[15:8]							
USART6BRL	7EF77FH	USART6BR[7:0]							

LIN 模式、IrDA 模式和 SmartCard 模式时的波特率计算公式

$$\text{同步波特率} = \frac{\text{SYSclk}}{16 * \text{USART6BR[15:0]}}$$

19.7.19 串口 6 接收超时控制寄存器（UR6TOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR6TOCR	7EF774H	UR6ENTO	UR6ENTOI	UR6SCALE	-	-	-	-	-

UR6ENTO：串口 6 接收超时功能控制位

0：禁止串口 6 接收超时功能

1：使能串口 6 接收超时功能

UR6ENTOI：串口 6 接收超时中断控制位

0：禁止串口 6 接收超时中断

1：使能串口 6 接收超时中断

UR6SCALE：串口 6 超时计数时钟源选择

0：串口数据位率（波特率）

1：系统时钟

19.7.20 串口 6 超时状态寄存器（UR6TOSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR6TOSR	7EF775H	UR6CTOIF	-	-	-	-	-	-	UR6TOIF

UR6CTOIF：写“1”清除串口超时中断标志位 TOIF。（只写）

UR6TOIF：串口 6 超时中断请求标志位。（只读）

当发生串口 6 超时时，TOIF 会被硬件置“1”。如果 ENTOI 为 1 时会产生串口中断，中断入口地址为原串口 6 的中断入口地址。标志位需要软件向 CTOIF 位写“1”清零

19.7.21 串口 6 超时长度控制寄存器（UR6TOTE/H/L）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR6TOTE	7EF773H	UR6TM[23:16]							
UR6TOTH	7EF776H	UR6TM[15:8]							
UR6TOTL	7EF777H	UR6TM[7:0]							

UR6TM[23:0]：串口 6 超时时间控制位。

当串口 6 处于接收空闲状态时，内部超时计数器根据 SCALE 寄存器所选择的时钟源进行计数，当计数时间达到 UR6TM 所设置的超时时间时，便会产生超时中断。当串口 6 接收数据完成时，复位内部超时计数器，重新进行超时计数。

注：

1、如果需要使能接收超时中断功能，则 TM 不可设置为 0，且 UR6TOTL、UR6TOTH、UR6TOTE 寄存器的设置必须先设置 UR6TOTL 和 UR6TOTH，最后设置 UR6TOTE

2、必须使能串口 6 接收才有接收超时功能

3、接收超时功能必须在正确接收到一字节数据后才能触发

4、正在接收过程中，无接收超时功能

19.8 串口 7（同步/异步串口 USART7）

19.8.1 串口 7 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX5	7EFD6DH	S8_S[1:0]		**S7_S[1:0]**		I2S2_S[1:0]		-	-

S7_S[1:0]：串口 7 功能脚选择位

S7_S[1:0]	RxD7	TxD7
00	P5.0	P5.1
01	PA.4	PA.5
10	P9.0	P9.1
11	P1.2	P1.3

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX6	7EFD6EH	-	-	-	-	S8SPI_S[1:0]		**S7SPI_S[1:0]**	

S7SPI_S[1:0]：USART7 的 SPI 功能脚选择位

S7SPI_S[1:0]	S7SS	S7MOSI	S7MISO	S7SCLK
00	P5.0	P5.1	P5.2	P5.3
01	PA.4	PA.5	PA.6	PA.7
10	P7.4	P7.5	P7.6	P7.7
11	PC.0	PC.1	PC.2	PC.3

19.8.2 串口 7 相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S7CON	串口 7 控制寄存器	7EF7A0H	S7SM0/FE	S7SM1	S7SM2	S7REN	S7TB8	S7RB8	S7TI	S7RI	0000,0000
S7BUF	串口 7 数据寄存器	7EF7A1H	S7BUF[7:0]								0000,0000
S7INTR	串口 7 中断寄存器	7EF7A2H	ES7	-	PS7[1:0]		-	-	-	-	0x00,xxxx

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S7CFG	串口 7 配置寄存器	7EF7CDH	S7MOD	S7MOD0	S7M0x6	-	-	-	-	S7BRT	000x,xxx0
S7ADDR	串口 7 从机地址寄存器	7EF7CEH	S7ADDR[7:0]								0000,0000
S7ADEN	串口 7 从机地址屏蔽寄存器	7EF7CFH	S7ADEN[7:0]								0000,0000
USART7CR1	串口 7 控制寄存器 1	7EF7A8H	S7LINEN	S7DORD	S7CLKEN	S7SPMOD	S7SPIEN	S7SPSLV	S7CPOL	S7CPHA	0000,0000
USART7CR2	串口 7 控制寄存器 2	7EF7A9H	S7IREN	S7IRLP	S7SCEN	S7NACK	S7HDSEL	S7PCEN	S7PS	S7PE	0000,0000
USART7CR3	串口 7 控制寄存器 3	7EF7AAH	S7IrDA_LPBAUD[7:0]								0000,0111
USART7CR4	串口 7 控制寄存器 4	7EF7ABH	-	-	-	-	S7SCCKS[1:0]		S7SPICKS[1:0]		xxxx,0000
USART7CR5	串口 7 控制寄存器 5	7EF7ACH	S7BRKDET	S7HDRER	S7SLVEN	S7ASYNC	S7TXCF	S7SENDBK	S7HDRDET	S7SYNC	0000,0000
USART7GTR	串口 7 保护时间寄存器	7EF7ADH	USART7GTR[7:0]								0000,0000
USART7BRH	串口 7 波特率寄存器	7EF7AEH	USART7BR[15:8]								0000,0000
USART7BRL	串口 7 波特率寄存器	7EF7AFH	USART7BR[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
UR7TOCR	串口 7 接收超时控制寄存器	7EF7A4H	UR7ENTO	UR7ENTOI	UR7SCALE	-	-	-	-	-	000x,xxxx
UR7TOSR	串口 7 接收超时状态寄存器	7EF7A5H	-	-	-	-	-	-	-	UR7TOIF	xxxx,xxx0
UR7TOTH	串口 7 接收超时长度寄存器	7EF7A6H	UR7TM[15:8]								0000,0000
UR7TOTL	串口 7 接收超时长度寄存器	7EF7A7H	UR7TM[7:0]								0000,0000
UR7TOTE	串口 7 接收超时长度寄存器	7EF7A3H	UR7TM[23:16]								0000,0000

19.8.3 串口 7 控制寄存器（S7CON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S7CON	7EF7A0H	S7SM0/FE	S7SM1	S7SM2	S7REN	S7TB8	S7RB8	S7TI	S7RI

S7SM0/FE：当S7CFG寄存器中的S7MOD0位为1时，该位为帧错误检测标志位。当串口7在接收过程中检测到一个无效停止位时，通过串口7接收器将该位置1，必须由软件清零。当S7CFG寄存器中的S7MOD0位为0时，该位和S7SM1一起指定串口7的通信工作模式，如下表所示：

S7SM0	S7SM1	串口7工作模式	功能说明
0	0	模式0	同步移位串行方式
0	1	模式1	可变波特率8位数据方式
1	0	模式2	固定波特率9位数据方式
1	1	模式3	可变波特率9位数据方式

当使能 FE 功能后，串口的工作模式只能使用 SM1 进行设置，且只能设置为模式 0 或者模式 1。FE 当作帧错误标志位时，此标志位必须软件写 0 清零。

S7SM2：允许模式 2 或模式 3 多机通信控制位。当串口 7 使用模式 2 或模式 3 时，如果 S7SM2 位为 1 且 S7REN 位为 1，则接收机处于地址帧筛选状态。此时可以利用接收到的第 9 位（即 S7RB8）来筛选地址帧，若 S7RB8=1，说明该帧是地址帧，地址信息可以进入 S7BUF，并使 S7RI 为 1，进而在中断服务程序中再进行地址号比较；若 S7RB8=0，说明该帧不是地址帧，应丢掉且保持 S7RI=0。在模式 2 或模式 3 中，如果 S7SM2 位为 0 且 S7REN 位为 1，接收收机处于地址帧筛选被禁止状态，不论收到的 S7RB8 为 0 或 1，均可使接收到的信息进入 S7BUF，并使 S7RI=1，此时 S7RB8 通常为校验位。模式 1 和模式 0 为非多机通信方式，在这两种方式时，S7SM2 应设置为 0。

S7REN：允许/禁止串口接收控制位

0：禁止串口接收数据

1：允许串口接收数据

S7TB8：当串口 7 使用模式 2 或模式 3 时，S7TB8 为要发送的第 9 位数据，按需要由软件置位或清 0。在模式 0 和模式 1 中，该位不用。

S7RB8：当串口 7 使用模式 2 或模式 3 时，S7RB8 为接收到的第 9 位数据，一般用作校验位或者地址帧/数据帧标志位。在模式 0 和模式 1 中，该位不用。

S7TI：串口 7 发送中断请求标志位。在模式 0 中，当串口发送数据第 8 位结束时，由硬件自动将 S7TI 置 1，向主机请求中断，响应中断后 S7TI 必须用软件清零。在其他模式中，则在停止位开始发送时由硬件自动将 S7TI 置 1，向 CPU 发请求中断，响应中断后 S7TI 必须用软件清零。

S7RI：串口 7 接收中断请求标志位。在模式 0 中，当串口接收第 8 位数据结束时，由硬件自动将 S7RI 置 1，向主机请求中断，响应中断后 S7RI 必须用软件清零。在其他模式中，串行接收到停止位的中间时刻由硬件自动将 S7RI 置 1，向 CPU 发中断申请，响应中断后 S7RI 必须由软件清零。

19.8.4 串口 7 数据寄存器（S7BUF）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S7BUF	7EF7A1H								

S7BUF：串口 1 数据接收/发送缓冲区。S7BUF 实际是 2 个缓冲器，读缓冲器和写缓冲器，两个操作分别对应两个不同的寄存器，1 个是只写寄存器（写缓冲器），1 个是只读寄存器（读缓冲器）。对 S7BUF 进行读操作，实际是读取串口接收缓冲区，对 S7BUF 进行写操作则是触发串口开始发送数据。

19.8.5 串口 7 中断控制寄存器（S7INTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S7INTR	7EF7A2H	ES7	-	PS7[1:0]		-	-	-	-

ES7：串口 7 中断使能控制位

0：禁止串口 7 中断

1：允许串口 7 中断

PS7[1:0]：串口 7 中断优先级控制位

00：串口 7 中断优先级为 0 级（最低级）

01：串口 7 中断优先级为 1 级（较低级）

10：串口 7 中断优先级为 2 级（较高级）

11：串口 7 中断优先级为 3 级（最高级）

19.8.6 串口 7 配置寄存器（S7CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S7CFG	7EF7CDH	S7MOD	S7MOD0	S7M0x6					S7BRT

S7MOD：串口 7 波特率控制位

0：串口 7 之模式 0/1/2/3 的波特率都不加倍

1：串口 7 之模式 2 的波特率加倍

S7MOD0：帧错误检测控制位

0：无帧错检测功能

1：使能帧错误检测功能。此时 S7CON 的 S7SM0/FE 为 FE 功能，即为帧错误检测标志位。

S7M0x6：串口 7 模式 0 的通讯速度控制

0：串口 7 模式 0 的波特率不加倍，固定为 Fosc/12

1：串口 7 模式 0 的波特率 6 倍速，即固定为 Fosc/12*6 ＝ Fosc/2

S7BRT：串口 7 波特率发生器选择位

0：选择定时器 2 作为波特率发生器（**默认值**）

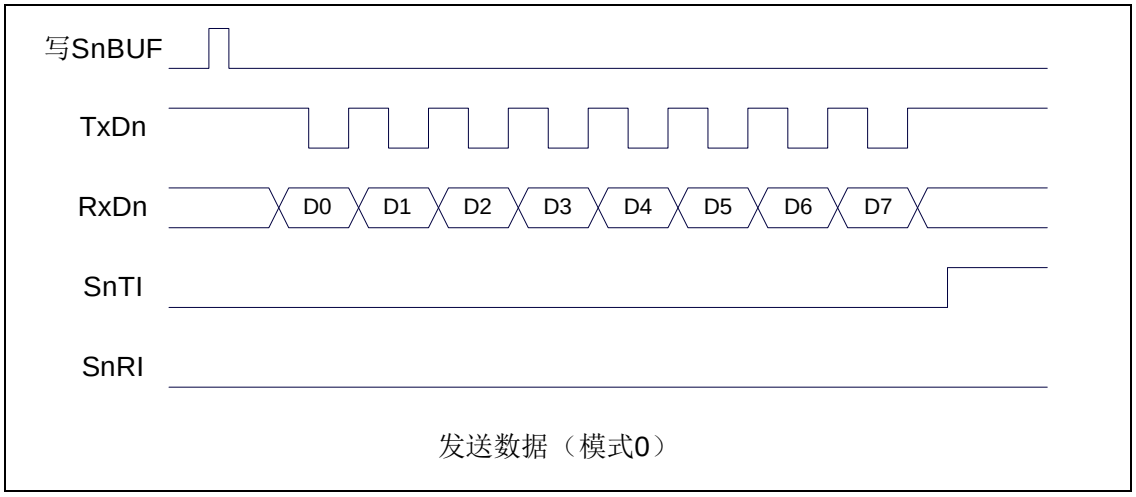
1：选择定时器 7 作为波特率发生器

19.8.7 串口 7 模式 0，模式 0 波特率计算公式

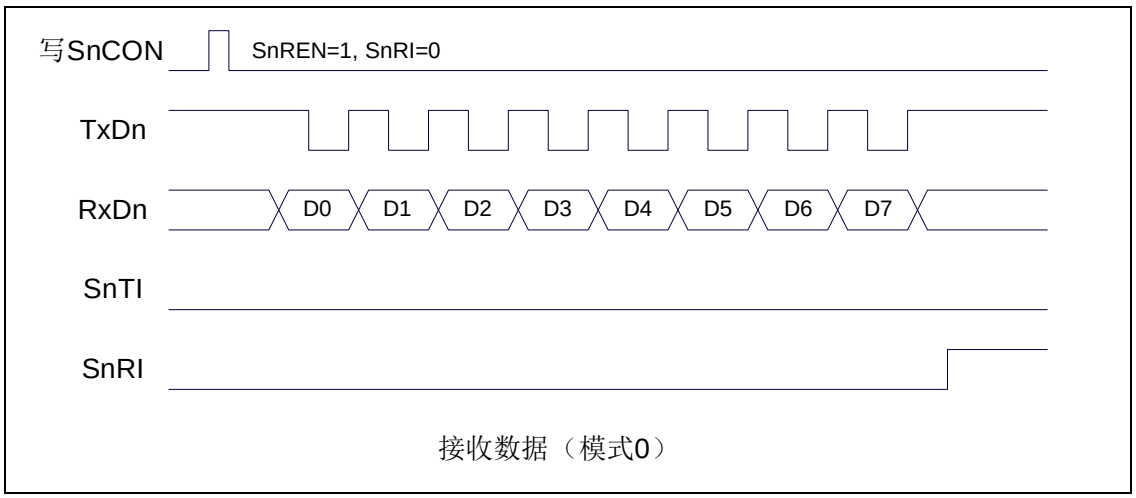
当串口 7 选择工作模式为模式 0 时，串行通信接口工作在同步移位寄存器模式，当串口 7 模式 0 的通信速度设置位 S7M0x6 为 0 时，其波特率固定为系统时钟时钟的 12 分频（SYSclk/12）；当设置 S7M0x6 为 1 时，其波特率固定为系统时钟频率的 2 分频（SYSclk/2）。RxD7 为串行通讯的数据口，TxD6 为同步移位脉冲输出脚，发送、接收的是 8 位数据，低位在先。

模式 0 的发送过程：当主机执行将数据写入发送缓冲器 S7BUF 指令时启动发送，串口 7 即将 8 位数据以 SYSclk/12 或 SYSclk/2（由 S7M0x6 确定是 12 分频还是 2 分频）的波特率从 RxD7 管脚输出(从低位到高位)，发送完中断标志 S7TI 置 1，TxD7 管脚输出同步移位脉冲信号。当写信号有效后，相隔一个时钟，发送控制端 SEND 有效(高电平)，允许 RxD7 发送数据，同时允许 TxD7 输出同步移位脉冲。一帧(8 位)数据发送完毕时，各控制端均恢复原状态，只有 S7TI 保持高电平，呈中断申请状态。在再次发送数据前，必须用软件将 S7TI 清 0。

模式 0 的接收过程：首先将接收中断请求标志 S7RI 清零并置位允许接收控制位 S7REN 时启动模式 0 接收过程。启动接收过程后，RxD7 为串行数据输入端，TxD7 为同步脉冲输出端。串行接收的波特率为 SYSclk/12 或 SYSclk/2（由 S7M0x6 确定是 12 分频还是 2 分频）。当接收完成一帧数据（8 位）后，控制信号复位，中断标志 S7RI 被置 1，呈中断申请状态。当再次接收时，必须通过软件将 S7RI 清 0



发送数据（模式0）



接收数据（模式0）

工作于模式 0 时，必须清 0 多机通信控制位 S7SM2，使之不影响 S7TB8 位和 S7RB8 位。由于波特率固定为 SYSclk/12 或 SYSclk/2，无需定时器提供，直接由单片机的时钟作为同步移位脉冲。

串口 7 模式 0 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

S7M0x6	波特率计算公式
0	$\text{波特率} = \frac{\text{SYSclk}}{12}$
1	$\text{波特率} = \frac{\text{SYSclk}}{2}$

19.8.8 串口 7 模式 1，模式 1 波特率计算公式

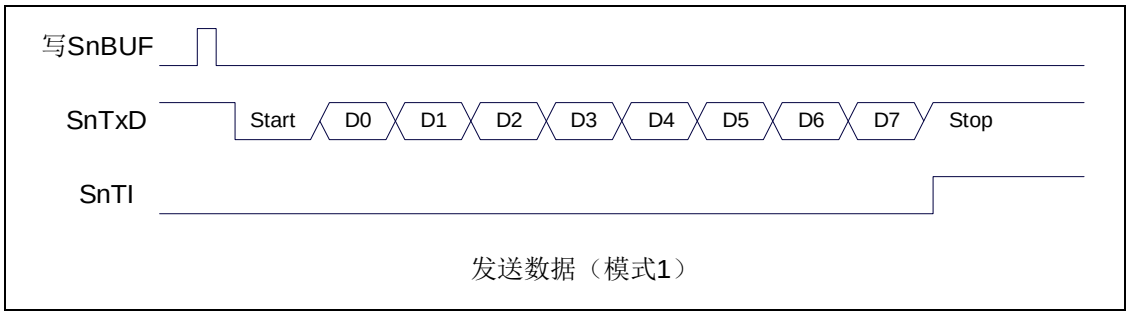
当软件设置 S7CON 的 S7SM0、S7SM1 为“01”时，串口 7 则以模式 1 进行工作。此模式为 8 位 UART 格式，一帧信息为 10 位：1 位起始位，8 位数据位（低位在先）和 1 位停止位。波特率可变，即可根据需要进行设置波特率。TxD7 为数据发送口，RxD7 为数据接收口，串口 7 全双工接受/发送。

模式 1 的发送过程：串行通信模式发送时，数据由串行发送端 TxD7 输出。当主机执行一条写 S7BUF 的指令就启动串行通信的发送，写“S7BUF”信号还把“1”装入发送移位寄存器的第 9 位，并通知 TX 控制单元开始发送。移位寄存器将数据不断右移送 TxD7 端口发送，在数据的左边不断移入“0”作补充。当数据的最高位移到移位寄存器的输出位置，紧跟其后的是第 9 位“1”，在它的左边各位全为“0”，这个状态条件，使 TX 控制单元作最后一次移位输出，然后使允许发送信号“SEND”失效，完成一帧信息的发送，并置位中断请求位 S7TI，即 S7TI=1，向主机请求中断处理。

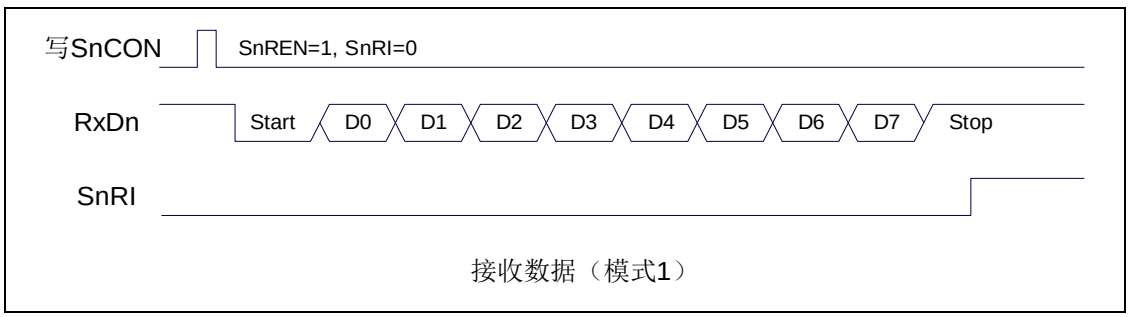
模式 1 的接收过程：当软件置位接收允许标志位 S7REN，即 S7REN=1 时，接收器便对 RxD7 端口的信号进行检测，当检测到 RxD7 端口发送从“1”→“0”的下降沿跳变时就启动接收器准备接收数据，并立即复位波特率发生器的接收计数器，将 1FFH 装入移位寄存器。接收的数据从接收移位寄存器的右边移入，已装入的 1FFH 向左边移出，当起始位"0"移到移位寄存器的最左边时，使 RX 控制器作最后一次移位，完成一帧的接收。若同时满足以下两个条件：

- S7RI=0；
- S7SM2=0 或接收到的停止位为 1。

则接收到的数据有效，实现装载入 S7BUF，停止位进入 S7RB8，S7RI 标志位被置 1，向主机请求中断，若上述两条件不能同时满足，则接收到的数据作废并丢失，无论条件满足与否，接收器重又检测 RxD7 端口上的"1"→"0"的跳变，继续下一帧的接收。接收有效，在响应中断后，S7RI 标志位必须由软件清 0。通常情况下，串行通信工作于模式 1 时，S7SM2 设置为"0"。



发送数据（模式1）



接收数据（模式1）

串口 7 的波特率是可变的，其波特率固定由定时器 2 或定时器 7 产生。当定时器采用 1T 模式时（12 倍速），相应的波特率的速度也会相应提高 12 倍。

串口 7 模式 1 的波特率计算公式如下表所示：（SYSclk 为系统工作频率）

选择定时器	定时器速度	重装值计算公式	波特率
定时器2 或 定时器7	1T	定时器重载值 = 65536 - $\frac{SYSclk}{4\times波特率}$	波特率 = $\frac{SYSclk}{4\times(65536 - 定时器重装数)}$
	12T	定时器重载值 = 65536 - $\frac{SYSclk}{12\times4\times波特率}$	波特率 = $\frac{SYSclk}{12\times4\times(65536 - 定时器重装数)}$

下面为常用频率与常用波特率所对应定时器的重载值

频率 (MHz)	波特率	定时器	
		1T 模式	12T 模式
11.0592	115200	FFE8H	FFFEH
	57600	FFD0H	FFFCH
	38400	FFB8H	FFFAH
	19200	FF70H	FFF4H
	9600	FEE0H	FFE8H
18.432	115200	FFD8H	-
	57600	FFB0H	-
	38400	FF88H	FFF6H
	19200	FF10H	FFECH
	9600	FE20H	FFD8H
22.1184	115200	FFD0H	FFFCH
	57600	FFA0H	FFF8H
	38400	FF70H	FFF4H
	19200	FEE0H	FFE8H
	9600	FDC0H	FFD0H

19.8.9 串口 7 模式 2，模式 2 波特率计算公式

当 S7SM0、S7SM1 两位为 10 时，串口 7 工作在模式 2。串口 7 工作模式 2 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S7CON 中的 S7TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S7TB8（S7TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S7CON 的 S7RB8。TxD7 为发送端口，RxD7 为接收端口，以全双工模式进行接收/发送。

模式 2 的波特率固定为系统时钟的 64 分频或 32 分频（取决于 S7CFG 中 S7MOD 的值）

串口 7 模式 2 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

SMOD	波特率计算公式
0	波特率 $= \frac{SYSclk}{64}$
1	波特率 $= \frac{SYSclk}{32}$

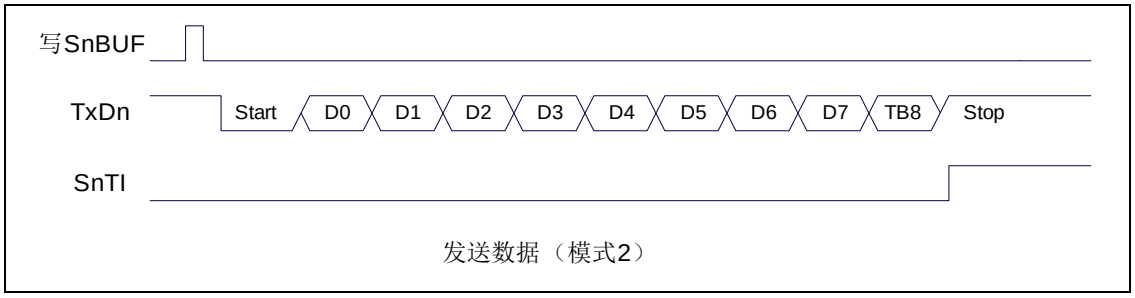
模式 2 和模式 1 相比，除波特率发生源略有不同，发送时由 S7TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收/发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

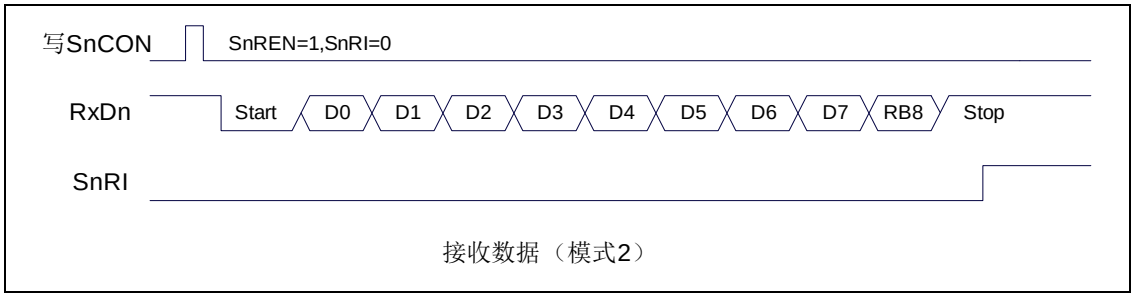
- S7RI=0
- S7SM2=0 或者 S7SM2=1 且接收到的第 9 数据位 S7RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S7BUF 和 S7RB8 中，S7RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S7RI。无论上述条件满足与否，接收器又重新开始检测 RxD7 输入端口的跳变信息，接收下一帧的输入信息。在模式 2 中，接收到的停止位与 S7BUF、S7RB8 和 S7RI 无关。

通过软件对 S7CON 中的 S7SM2、S7TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式2）



接收数据（模式2）

19.8.10 串口 7 模式 3，模式 3 波特率计算公式

当 S7SM0、S7SM1 两位为 11 时，串口 7 工作在模式 3。串行通信模式 3 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S7CON 中的 S7TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S7TB8（S7TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S7CON 的 S7RB8。TxD7 为发送端口，RxD7 为接收端口，以全双工模式进行接收/发送。

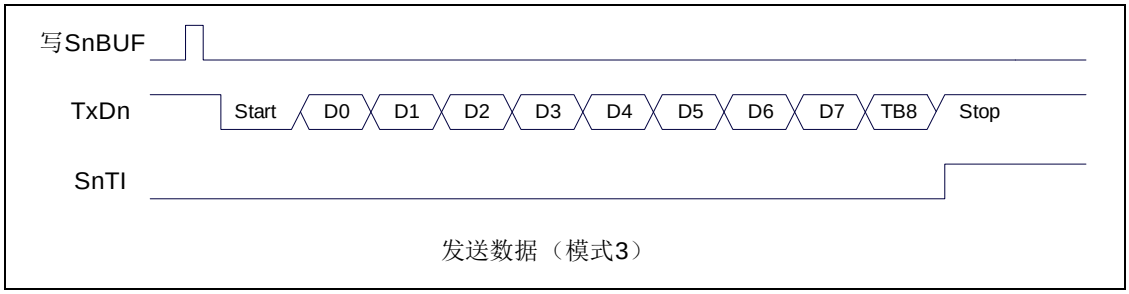
模式 3 和模式 1 相比，除发送时由 S7TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收‘发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

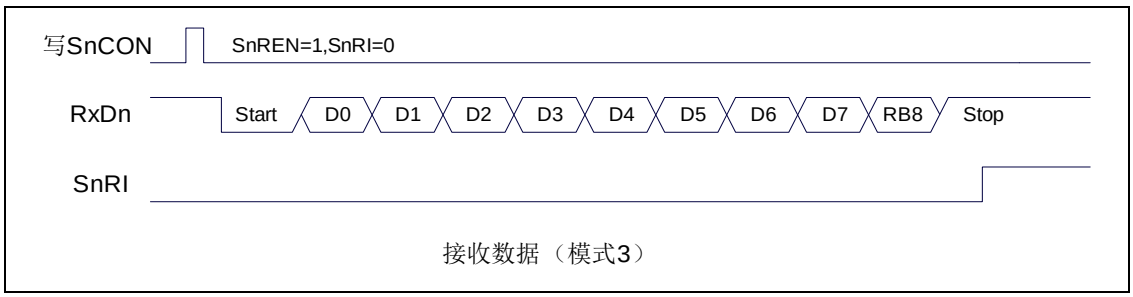
- S7RI=0
- S7SM2=0 或者 S7SM2=1 且接收到的第 9 数据位 S7RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S7BUF 和 S7RB8 中，S7RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S7RI。无论上述条件满足与否，接收器又重新开始检测 RxD7 输入端口的跳变信息，接收下一帧的输入信息。在模式 3 中，接收到的停止位与 S7BUF、S7RB8 和 S7RI 无关。

通过软件对 S7CON 中的 S7SM2、S7TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式3）



接收数据（模式3）

串口 7 模式 3 的波特率计算公式与模式 1 是完全相同的。请参考模式 1 的波特率计算公式。

19.8.11 串口 7 自动地址识别，从机地址控制寄存器（S7ADDR，S7ADEN）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S7ADDR	7EF7CEH								
S7ADEN	7EF7CFH								

S7ADDR：从机地址寄存器

S7ADEN：从机地址屏蔽位寄存器

自动地址识别功能典型应用在多机通讯领域，其主要原理是从机系统通过硬件比较功能来识别来自于主机串口 7 数据流中的地址信息，通过寄存器 S7ADDR 和 S7ADEN 设置的本机的从机地址，硬件自动对从机地址进行过滤，当来自于主机的从机地址信息与本机所设置的从机地址相匹配时，硬件产生串口 7 中断；否则硬件自动丢弃串口 7 数据，而不产生中断。当众多处于空闲模式的从机链接在一起时，只有从机地址相匹配的从机才会从空闲模式唤醒，从而可以大大降低从机 MCU 的功耗，即使从机处于正常工作状态也可避免不停地进入串口 7 中断而降低系统执行效率。

要使用串口 7 的自动地址识别功能，首先需要将参与通讯的 MCU 的串口 7 通讯模式设置为模式 1，并开启从机的 S7CON 的 S7SM2 位。对于串口 7 模式 1 的 9 位数据位中，第 9 位数据（存放在 S7RB8 中）为地址/数据的标志位，当第 9 位数据为 1 时，表示前面的 8 位数据（存放在 S7BUF 中）为地址信息。当 S7SM2 被设置为 1 时，从机 MCU 会自动过滤掉非地址数据（第 9 位为 0 的数据），而对 S7BUF 中的地址数据（第 9 位为 1 的数据）自动与 S7ADDR 和 S7ADEN 所设置的本机地址进行比较，若地址相匹配，则会将 S7RI 置“1”，并产生中断，否则不予处理本次接收的串口 7 数据。

从机地址的设置是通过 S7ADDR 和 S7ADEN 两个寄存器进行设置的。S7ADDR 为从机地址寄存器，里面存放本机的从机地址。S7ADEN 为从机地址屏蔽位寄存器，用于设置地址信息中的忽略位，设置方法如下：

例如

S7ADDR ＝ 11001010

S7ADEN ＝ 10000001

则匹配地址为 1xxxxxx0

即，只要主机送出的地址数据中的 bit0 为 0 且 bit7 为 1 就可以和本机地址相匹配

再例如

S7ADDR ＝ 11001010

S7ADEN ＝ 00001111

则匹配地址为 xxxx1010

即，只要主机送出的地址数据中的低 4 位为 1010 就可以和本机地址相匹配，而高 4 为被忽略，可以为任意值。

主机可以使用广播地址（FFH）同时选中所有的从机来进行通讯。

19.8.12 串口 7 同步模式控制寄存器 1（USART7CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART7CR1	7EF7A8H	S7LINEN	S7DORD	S7CLKEN	S7SPMOD	S7SPIEN	S7SPSLV	S7CPOL	S7CPHA

S7LINEN：LIN 模式使能位

0：禁止 LIN 模式

1：使能 LIN 模式

S7DORD：SPI 模式数据位发送/接收的顺序

0：先发送/接收数据的高位（MSB）

1：先发送/接收数据的低位（LSB）

S7CLKEN：SmartCard 模式时钟输出控制位

0：禁止时钟输出

1：使能时钟输出

S7SPMOD：SPI 模式使能位

0：禁止 SPI 模式

1：使能 SPI 模式

S7SPIEN：SPI 使能位

0：禁止 SPI 功能

1：使能 SPI 功能

当 USART7 需要工作在 SPI 模式时，必须先将 SPI 相关的模式寄存器、速度寄存器、时钟极性寄存器设置完成后，最后再设置 SPIEN 寄存器。

S7SPSLV：SPI 从机模式使能位

0：SPI 为主机模式

1：SPI 为从机模式

S7CPOL：SPI 时钟极性控制

0：SCLK 空闲时为低电平，SCLK 的前时钟沿为上升沿，后时钟沿为下降沿

1：SCLK 空闲时为高电平，SCLK 的前时钟沿为下降沿，后时钟沿为上升沿

S7CPHA：SPI 时钟相位控制

0：数据 SS 管脚为低电平驱动第一位数据并在 SCLK 的后时钟沿改变数据，前时钟沿采样数据

1：数据在 SCLK 的前时钟沿驱动，后时钟沿采样

19.8.13 串口 7 同步模式控制寄存器 2（USART7CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART7CR2	7EF7A9H	S7IREN	S7IRLP	S7SCEN	S7NACK	S7HDSEL	S7PCEN	S7PS	S7PE

S7IREN：IrDA 模式使能位

0：禁止 IrDA 模式

1：使能 IrDA 模式

S7IRLP：IrDA 低电模式控制位

0：IrDA 为普通模式

1：IrDA 为低电模式

S7SCEN：SmartCard 模式使能位

0：禁止 SmartCard 模式

1：使能 SmartCard 模式

S7NACK：SmartCard 模式 NACK 输出使能位

0：禁止输出 NACK 信号

1：使能输出 NACK 信号

S7HDSEL：单线半双工模式使能位

0：禁止单线半双工模式

1：使能单线半双工模式

S7PCEN：硬件自动产生校验位控制使能

0：禁止硬件自动产生校验位（串口的校验位为 S7TB8 设置的值）

1：使能硬件自动产生校验位

S7PS：硬件校验位模式选择

0：硬件根据 S7BUF 的值自动产生偶校验位

1：硬件根据 S7BUF 的值自动产生奇校验位

S7PE：校验位错误标志（必须软件清零）

0：无检验错误

1：有校验错误（串口接收 DMA 过程中如果发生接收数据校验位错误，DMA 不会停止，但校验位错误标志会一直保持直到 DMA 完成）

19.8.14 串口 7 同步模式控制寄存器 3（USART7CR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART7CR3	7EF7AAH	S7IrDA_LPBAUD[7:0]							

S7IrDA_LPBAUD：IrDA 低电模式波特率控制寄存器

IrDA_LPBAUD[7:0]	IrDA 低电模式波特率
0	SYSclk/16/256
1	SYSclk/16/1
2	SYSclk/16/2
...	...
255	SYSclk/16/255

19.8.15 串口 7 同步模式控制寄存器 4（USART7CR4）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART7CR4	7EF7ABH	-	-	-	-	S7SCCKS[1:0]		S7SPICKS[1:0]	

S7SCCKS[1:0]：SmartCard 模式时钟选择

SCCKS[1:0]	SmartCard 模式时钟
00	SYSclk/64
01	SYSclk/32
10	SYSclk/16
11	SYSclk/8

S7SPICKS[1:0]：SPI 模式时钟选择

SPICKS[1:0]	SPI 模式时钟
00	SYSclk/4
01	SYSclk/8
10	SYSclk/16
11	SYSclk/2

19.8.16 串口 7 同步模式控制寄存器 5（USART7CR5）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART7CR5	7EF7ACH	S7BRKDET	S7HDRER	S7SLVEN	S7ASYNC	S7TXCF	S7SENDBK	S7HDRDET	S7SYNC

S7BRKDET：LIN 从模式间隔场（BREAK）检测标志

0：未检测到 LIN 间隔场

1：检测到 LIN 间隔场，需要软件清零

S7HDRER：LIN 从模式报文头（HEADER）错误

0：未检测到 LIN 报文头错误

1：检测到 LIN 报文头错误，需要软件清零

S7SLVEN：LIN 从机模式使能位

0：LIN 为主机模式

1：LIN 为从机模式

S7ASYNC：LIN 自动同步功能使能位

0：禁止自动同步

1：使能自动同步

【注意】：从机使能自动同步功能后，如果从机跟主机波特率存在小范围误差，从机收到主机同步信号后，会自动将从机波特率设置成与主机相同的波特率。

例如，主机波特率 9600：

从机使能自动同步时，如果从机波特率在 9100～10000 附近，收到主机同步信号后，从机会自动将本身波特率设置为 9600。误差自动同步范围 -5.2%～+4.16%左右。

S7TXCF：传输冲突标志位。单线半双工模式、LIN 模式和 SmartCard 模式有效

0：未检测到传输冲突（发送的数据与接收的数据相同）

1：检测到传输冲突（发送的数据与接收的数据不同）

S7SENDBK：发送间隔场。软件写 1 触发发送间隔场，发送完成后硬件自动清 0

S7HDRDET：LIN 从模式报文头（HEADER）检测标志

0：未检测到 LIN 报文头

1：检测到 LIN 报文头，需要软件清零

S7SYNC：LIN 从模式同步场检测标志。

正确分析到同步场后，硬件将 SYNC 标志为置 1。在接收标志符场时硬件会自动清零 SYNC

19.8.17 串口 7 同步模式保护时间寄存器（USART7GTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART7GTR	7EF7ADH								

USART7GTR 寄存器仅在 SmartCard 模式有效。该寄存器是根据波特率时钟数给出保护时间值。S4TI 标志在 SmartCard 发送的数据位等于 USART7GTR 寄存器所设置的保护时间值时被硬件置 1。注：USART7GTR 寄存器的值应大于 11

19.8.18 串口 7 同步模式波特率寄存器（USART7BR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART7BRH	7EF7AEH	USART7BR[15:8]							
USART7BRL	7EF7AFH	USART7BR[7:0]							

LIN 模式、IrDA 模式和 SmartCard 模式时的波特率计算公式

$$\text{同步波特率} = \frac{\text{SYSclk}}{16 * \text{USART7BR[15:0]}}$$

19.8.19 串口 7 接收超时控制寄存器（UR7TOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR7TOCR	7EF7A4H	UR7ENTO	UR7ENTOI	UR7SCALE	-	-	-	-	-

UR7ENTO：串口 7 接收超时功能控制位

0：禁止串口 7 接收超时功能

1：使能串口 7 接收超时功能

UR7ENTOI：串口 7 接收超时中断控制位

0：禁止串口 7 接收超时中断

1：使能串口 7 接收超时中断

UR7SCALE：串口 7 超时计数时钟源选择

0：串口数据位率（波特率）

1：系统时钟

19.8.20 串口 7 超时状态寄存器（UR7TOSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR7TOSR	7EF7A5H	UR7CTOIF	-	-	-	-	-	-	UR7TOIF

UR7CTOIF：写“1”清除串口超时中断标志位 TOIF。（只写）

UR7TOIF：串口 7 超时中断请求标志位。（只读）

当发生串口 7 超时时，TOIF 会被硬件置“1”。如果 ENTOI 为 1 时会产生串口中断，中断入口地址为原串口 7 的中断入口地址。标志位需要软件向 CTOIF 位写“1”清零

19.8.21 串口 7 超时长度控制寄存器（UR7TOTE/H/L）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR7TOTE	7EF7A3H	UR7TM[23:16]							
UR7TOTH	7EF7A6H	UR7TM[15:8]							
UR7TOTL	7EF7A7H	UR7TM[7:0]							

UR7TM[23:0]：串口 7 超时时间控制位。

当串口 7 处于接收空闲状态时，内部超时计数器根据 SCALE 寄存器所选择的时钟源进行计数，当计数时间达到 UR7TM 所设置的超时时间时，便会产生超时中断。当串口 7 接收数据完成时，复位内部超时计数器，重新进行超时计数。

注：

1、如果需要使能接收超时中断功能，则 TM 不可设置为 0，且 UR7TOTL、UR7TOTH、UR7TOTE 寄存器的设置必须先设置 UR7TOTL 和 UR7TOTH，最后设置 UR7TOTE

2、必须使能串口 7 接收才有接收超时功能

3、接收超时功能必须在正确接收到一字节数据后才能触发

4、正在接收过程中，无接收超时功能

19.9 串口 8（同步/异步串口 USART8）

19.9.1 串口 8 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX5	7EFD6DH	**S8_S[1:0]**		S7_S[1:0]		I2S2_S[1:0]		-	-

S8_S[1:0]：串口 8 功能脚选择位

S8_S[1:0]	RxD8	TxD8
00	P5.2	P5.3
01	PA.6	PA.7
10	P9.2	P9.3
11	P6.0	P6.1

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX6	7EFD6EH	-	-	-	-	**S8SPI_S[1:0]**		S7SPI_S[1:0]	

S8SPI_S[1:0]：USART8 的 SPI 功能脚选择位

S8SPI_S[1:0]	S8SS	S8MOSI	S8MISO	S8SCLK
00	P5.0	P5.1	P5.2	P5.3
01	PA.4	PA.5	PA.6	PA.7
10	P7.4	P7.5	P7.6	P7.7
11	P6.0	P6.1	P6.2	P6.3

19.9.2 串口 8 相关寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S8CON	串口 8 控制寄存器	7EF7D0H	S8SM0/FE	S8SM1	S8SM2	S8REN	S8TB8	S8RB8	S8TI	S8RI	0000,0000
S8BUF	串口 8 数据寄存器	7EF7D1H	S8BUF[7:0]								0000,0000
S8INTR	串口 8 中断寄存器	7EF7D2H	ES8	-	PS8[1:0]		-	-	-	-	0x00,xxxx

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
S8CFG	串口 8 配置寄存器	7EF7FDH	S8MOD	S8MOD0	S8M0x6	-	-	-	-	S8BRT	000x,xxx0
S8ADDR	串口 8 从机地址寄存器	7EF7FEH	S8ADDR[7:0]								0000,0000
S8ADEN	串口 8 从机地址屏蔽寄存器	7EF7FFH	S8ADEN[7:0]								0000,0000
USART8CR1	串口 8 控制寄存器 1	7EF7D8H	S8LINEN	S8DORD	S8CLKEN	S8SPMOD	S8SPIEN	S8SPSLV	S8CPOL	S8CPHA	0000,0000
USART8CR2	串口 8 控制寄存器 2	7EF7D9H	S8IREN	S8IRLP	S8SCEN	S8NACK	S8HDSEL	S8PCEN	S8PS	S8PE	0000,0000
USART8CR3	串口 8 控制寄存器 3	7EF7DAH	S8IrDA_LPBAUD[7:0]								0000,0111
USART8CR4	串口 8 控制寄存器 4	7EF7DBH	-	-	-	-	S8SCCKS[1:0]		S8SPICKS[1:0]		xxxx,0000
USART8CR5	串口 8 控制寄存器 5	7EF7DCH	S8BRKDET	S8HDRER	S8SLVEN	S8ASYNC	S8TXCF	S8SENDBK	S8HDRDET	S8SYNC	0000,0000
USART8GTR	串口 8 保护时间寄存器	7EF7DDH	USART8GTR[7:0]								0000,0000
USART8BRH	串口 8 波特率寄存器	7EF7DEH	USART8BR[15:8]								0000,0000
USART8BRL	串口 8 波特率寄存器	7EF7DFH	USART8BR[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
UR8TOCR	串口 8 接收超时控制寄存器	7EF7D4H	UR8ENTO	UR8ENTOI	UR8SCALE	-	-	-	-	-	000x,xxxx
UR8TOSR	串口 8 接收超时状态寄存器	7EF7D5H	-	-	-	-	-	-	-	UR8TOIF	xxxx,xxx0
UR8TOTH	串口 8 接收超时长度寄存器	7EF7D6H	UR8TM[15:8]								0000,0000
UR8TOTL	串口 8 接收超时长度寄存器	7EF7D7H	UR8TM[7:0]								0000,0000
UR8TOTE	串口 8 接收超时长度寄存器	7EF7D3H	UR8TM[23:16]								0000,0000

19.9.3 串口 8 控制寄存器（S8CON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S8CON	7EF7D0H	S8SM0/FE	S8SM1	S8SM2	S8REN	S8TB8	S8RB8	S8TI	S8RI

S8SM0/FE：当S8CFG寄存器中的S8MOD0位为1时，该位为帧错误检测标志位。当串口8在接收过程中检测到一个无效停止位时，通过串口8接收器将该位置1，必须由软件清零。当S8CFG寄存器中的S8MOD0位为0时，该位和S8SM1一起指定串口8的通信工作模式，如下表所示：

S8SM0	S8SM1	串口8工作模式	功能说明
0	0	模式0	同步移位串行方式
0	1	模式1	可变波特率8位数据方式
1	0	模式2	固定波特率9位数据方式
1	1	模式3	可变波特率9位数据方式

当使能 FE 功能后，串口的工作模式只能使用 SM1 进行设置，且只能设置为模式 0 或者模式 1。FE 当作帧错误标志位时，此标志位必须软件写 0 清零。

S8SM2：允许模式 2 或模式 3 多机通信控制位。当串口 8 使用模式 2 或模式 3 时，如果 S8SM2 位为 1 且 S8REN 位为 1，则接收机处于地址帧筛选状态。此时可以利用接收到的第 9 位（即 S8RB8）来筛选地址帧，若 S8RB8=1，说明该帧是地址帧，地址信息可以进入 S8BUF，并使 S8RI 为 1，进而在中断服务程序中再进行地址号比较；若 S8RB8=0，说明该帧不是地址帧，应丢掉且保持 S8RI=0。在模式 2 或模式 3 中，如果 S8SM2 位为 0 且 S8REN 位为 1，接收收机处于地址帧筛选被禁止状态，不论收到的 S8RB8 为 0 或 1，均可使接收到的信息进入 S8BUF，并使 S8RI=1，此时 S8RB8 通常为校验位。模式 1 和模式 0 为非多机通信方式，在这两种方式时，S8SM2 应设置为 0。

S8REN：允许/禁止串口接收控制位

0：禁止串口接收数据

1：允许串口接收数据

S8TB8：当串口 8 使用模式 2 或模式 3 时，S8TB8 为要发送的第 9 位数据，按需要由软件置位或清 0。在模式 0 和模式 1 中，该位不用。

S8RB8：当串口 8 使用模式 2 或模式 3 时，S8RB8 为接收到的第 9 位数据，一般用作校验位或者地址帧/数据帧标志位。在模式 0 和模式 1 中，该位不用。

S8TI：串口 8 发送中断请求标志位。在模式 0 中，当串口发送数据第 8 位结束时，由硬件自动将 S8TI 置 1，向主机请求中断，响应中断后 S8TI 必须用软件清零。在其他模式中，则在停止位开始发送时由硬件自动将 S8TI 置 1，向 CPU 发请求中断，响应中断后 S8TI 必须用软件清零。

S8RI：串口 8 接收中断请求标志位。在模式 0 中，当串口接收第 8 位数据结束时，由硬件自动将 S8RI 置 1，向主机请求中断，响应中断后 S8RI 必须用软件清零。在其他模式中，串行接收到停止位的中间时刻由硬件自动将 S8RI 置 1，向 CPU 发中断申请，响应中断后 S8RI 必须由软件清零。

19.9.4 串口 8 数据寄存器（S8BUF）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S8BUF	7EF7D1H								

S8BUF：串口 1 数据接收/发送缓冲区。S8BUF 实际是 2 个缓冲器，读缓冲器和写缓冲器，两个操作分别对应两个不同的寄存器，1 个是只写寄存器（写缓冲器），1 个是只读寄存器（读缓冲器）。对 S8BUF 进行读操作，实际是读取串口接收缓冲区，对 S8BUF 进行写操作则是触发串口开始发送数据。

19.9.5 串口 8 中断控制寄存器（S8INTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S8INTR	7EF7D2H	ES8	-	PS8[1:0]		-	-	-	-

ES8：串口 8 中断使能控制位

0：禁止串口 8 中断

1：允许串口 8 中断

PS8[1:0]：串口 8 中断优先级控制位

00：串口 8 中断优先级为 0 级（最低级）

01：串口 8 中断优先级为 1 级（较低级）

10：串口 8 中断优先级为 2 级（较高级）

11：串口 8 中断优先级为 3 级（最高级）

19.9.6 串口 8 配置寄存器（S8CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S8CFG	7EF7FDH	S8MOD	S8MOD0	S8M0x6	-	-	-	-	S8BRT

S8MOD：串口 8 波特率控制位

0：串口 8 之模式 0/1/2/3 的波特率都不加倍

1：串口 8 之模式 2 的波特率加倍

S8MOD0：帧错误检测控制位

0：无帧错检测功能

1：使能帧错误检测功能。此时 S8CON 的 S8SM0/FE 为 FE 功能，即为帧错误检测标志位。

S8M0x6：串口 8 模式 0 的通讯速度控制

0：串口 8 模式 0 的波特率不加倍，固定为 Fosc/12

1：串口 8 模式 0 的波特率 6 倍速，即固定为 Fosc/12*6 ＝ Fosc/2

S8BRT：串口 8 波特率发生器选择位

0：选择定时器 2 作为波特率发生器（**默认值**）

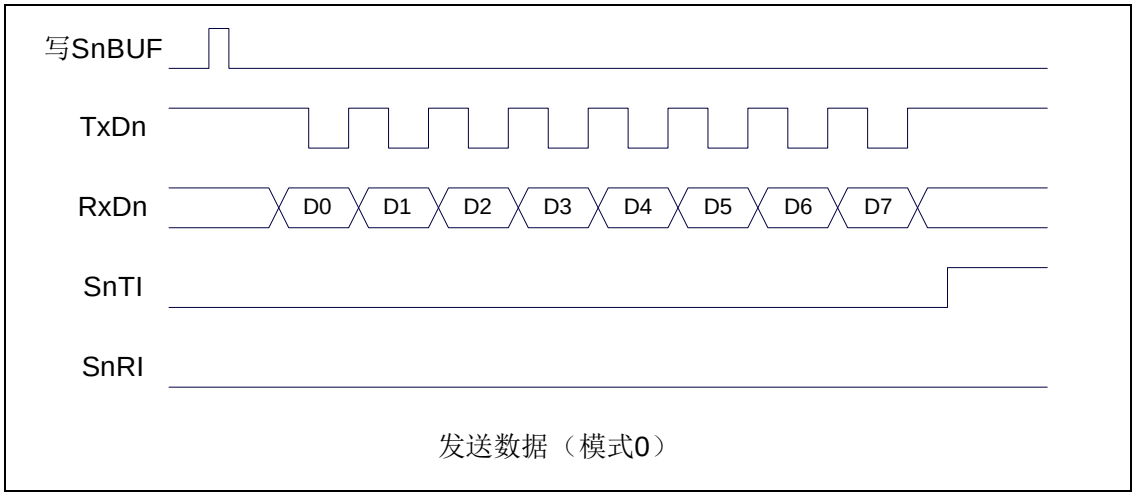
1：选择定时器 8 作为波特率发生器

19.9.7 串口 8 模式 0，模式 0 波特率计算公式

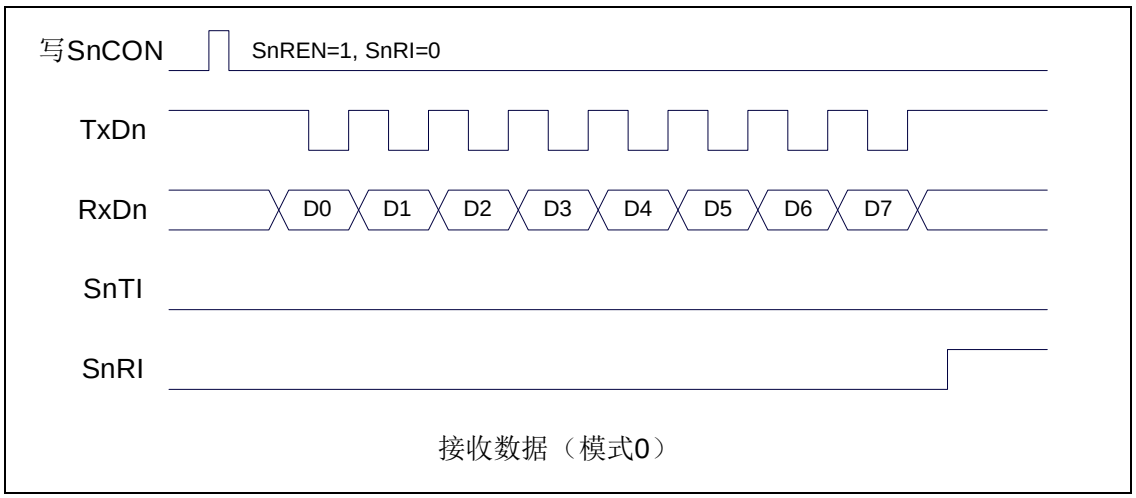
当串口 8 选择工作模式为模式 0 时，串行通信接口工作在同步移位寄存器模式，当串口 8 模式 0 的通信速度设置位 S8M0x6 为 0 时，其波特率固定为系统时钟时钟的 12 分频（SYSclk/12）；当设置 S8M0x6 为 1 时，其波特率固定为系统时钟频率的 2 分频（SYSclk/2）。RxD8 为串行通讯的数据口，TxD8 为同步移位脉冲输出脚，发送、接收的是 8 位数据，低位在先。

模式 0 的发送过程：当主机执行将数据写入发送缓冲器 S8BUF 指令时启动发送，串口 8 即将 8 位数据以 SYSclk/12 或 SYSclk/2（由 S8M0x6 确定是 12 分频还是 2 分频）的波特率从 RxD8 管脚输出(从低位到高位)，发送完中断标志 S8TI 置 1，TxD8 管脚输出同步移位脉冲信号。当写信号有效后，相隔一个时钟，发送控制端 SEND 有效(高电平)，允许 RxD8 发送数据，同时允许 TxD8 输出同步移位脉冲。一帧(8 位)数据发送完毕时，各控制端均恢复原状态，只有 S8TI 保持高电平，呈中断申请状态。在再次发送数据前，必须用软件将 S8TI 清 0。

模式 0 的接收过程：首先将接收中断请求标志 S8RI 清零并置位允许接收控制位 S8REN 时启动模式 0 接收过程。启动接收过程后，RxD8 为串行数据输入端，TxD8 为同步脉冲输出端。串行接收的波特率为 SYSclk/12 或 SYSclk/2（由 S8M0x6 确定是 12 分频还是 2 分频）。当接收完成一帧数据（8 位）后，控制信号复位，中断标志 S8RI 被置 1，呈中断申请状态。当再次接收时，必须通过软件将 S8RI 清 0



发送数据（模式0）



接收数据（模式0）

工作于模式 0 时，必须清 0 多机通信控制位 S8SM2，使之不影响 S8TB8 位和 S8RB8 位。由于波特率固定为 SYSclk/12 或 SYSclk/2，无需定时器提供，直接由单片机的时钟作为同步移位脉冲。

串口 8 模式 0 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

S8M0x6	波特率计算公式
0	$\text{波特率} = \frac{\text{SYSclk}}{12}$
1	$\text{波特率} = \frac{\text{SYSclk}}{2}$

19.9.8 串口 8 模式 1，模式 1 波特率计算公式

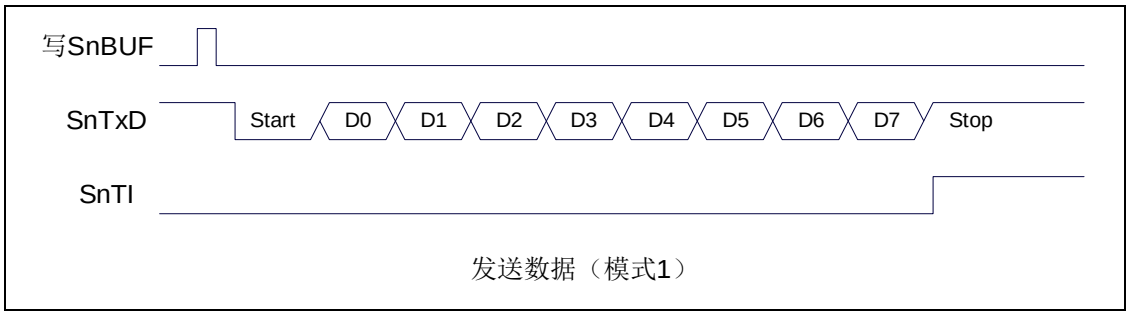
当软件设置 S8CON 的 S8SM0、S8SM1 为“01”时，串口 8 则以模式 1 进行工作。此模式为 8 位 UART 格式，一帧信息为 10 位：1 位起始位，8 位数据位（低位在先）和 1 位停止位。波特率可变，即可根据需要进行设置波特率。TxD8 为数据发送口，RxD8 为数据接收口，串口 8 全双工接受/发送。

模式 1 的发送过程：串行通信模式发送时，数据由串行发送端 TxD8 输出。当主机执行一条写 S8BUF 的指令就启动串行通信的发送，写“S8BUF”信号还把“1”装入发送移位寄存器的第 9 位，并通知 TX 控制单元开始发送。移位寄存器将数据不断右移送 TxD8 端口发送，在数据的左边不断移入“0”作补充。当数据的最高位移到移位寄存器的输出位置，紧跟其后的是第 9 位“1”，在它的左边各位全为“0”，这个状态条件，使 TX 控制单元作最后一次移位输出，然后使允许发送信号“SEND”失效，完成一帧信息的发送，并置位中断请求位 S8TI，即 S8TI=1，向主机请求中断处理。

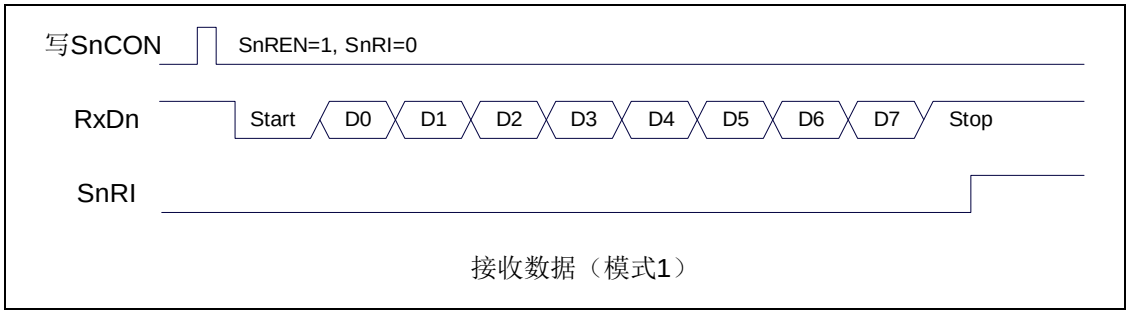
模式 1 的接收过程：当软件置位接收允许标志位 S8REN，即 S8REN=1 时，接收器便对 RxD8 端口的信号进行检测，当检测到 RxD8 端口发送从“1”→“0”的下降沿跳变时就启动接收器准备接收数据，并立即复位波特率发生器的接收计数器，将 1FFH 装入移位寄存器。接收的数据从接收移位寄存器的右边移入，已装入的 1FFH 向左边移出，当起始位"0"移到移位寄存器的最左边时，使 RX 控制器作最后一次移位，完成一帧的接收。若同时满足以下两个条件：

- S8RI=0；
- S8SM2=0 或接收到的停止位为 1。

则接收到的数据有效，实现装载入 S8BUF，停止位进入 S8RB8，S8RI 标志位被置 1，向主机请求中断，若上述两条件不能同时满足，则接收到的数据作废并丢失，无论条件满足与否，接收器重又检测 RxD8 端口上的"1"→"0"的跳变，继续下一帧的接收。接收有效，在响应中断后，S8RI 标志位必须由软件清 0。通常情况下，串行通信工作于模式 1 时，S8SM2 设置为"0"。



发送数据（模式1）



接收数据（模式1）

串口 8 的波特率是可变的，其波特率固定由定时器 2 或定时器 8 产生。当定时器采用 1T 模式时（12 倍速），相应的波特率的速度也会相应提高 12 倍。

串口 8 模式 1 的波特率计算公式如下表所示：（SYSclk 为系统工作频率）

选择定时器	定时器速度	重装值计算公式	波特率
定时器2 或 定时器8	1T	定时器重载值 = 65536 - $\frac{SYSclk}{4\times波特率}$	波特率 = $\frac{SYSclk}{4\times(65536 - 定时器重装数)}$
	12T	定时器重载值 = 65536 - $\frac{SYSclk}{12\times4\times波特率}$	波特率 = $\frac{SYSclk}{12\times4\times(65536 - 定时器重装数)}$

下面为常用频率与常用波特率所对应定时器的重载值

频率 (MHz)	波特率	定时器	
		1T 模式	12T 模式
11.0592	115200	FFE8H	FFFEH
	57600	FFD0H	FFFCH
	38400	FFB8H	FFFAH
	19200	FF70H	FFF4H
	9600	FEE0H	FFE8H
18.432	115200	FFD8H	-
	57600	FFB0H	-
	38400	FF88H	FFF6H
	19200	FF10H	FFECH
	9600	FE20H	FFD8H
22.1184	115200	FFD0H	FFFCH
	57600	FFA0H	FFF8H
	38400	FF70H	FFF4H
	19200	FEE0H	FFE8H
	9600	FDC0H	FFD0H

19.9.9 串口 8 模式 2，模式 2 波特率计算公式

当 S8SM0、S8SM1 两位为 10 时，串口 8 工作在模式 2。串口 8 工作模式 2 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S8CON 中的 S8TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S8TB8（S8TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S8CON 的 S8RB8。TxD8 为发送端口，RxD8 为接收端口，以全双工模式进行接收/发送。

模式 2 的波特率固定为系统时钟的 64 分频或 32 分频（取决于 S8CFG 中 S8MOD 的值）

串口 8 模式 2 的波特率计算公式如下表所示（SYSclk 为系统工作频率）：

SMOD	波特率计算公式
0	$\text{波特率} = \frac{\text{SYSclk}}{64}$
1	$\text{波特率} = \frac{\text{SYSclk}}{32}$

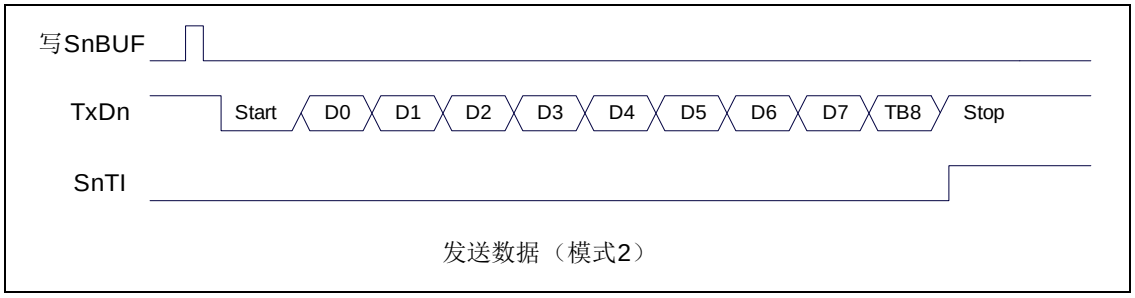
模式 2 和模式 1 相比，除波特率发生源略有不同，发送时由 S8TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收/发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

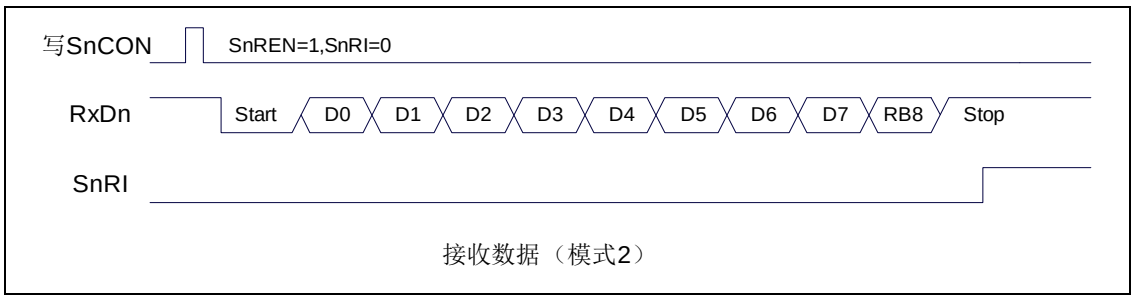
- S8RI=0
- S8SM2=0 或者 S8SM2=1 且接收到的第 9 数据位 S8RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S8BUF 和 S8RB8 中，S8RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S8RI。无论上述条件满足与否，接收器又重新开始检测 RxD8 输入端口的跳变信息，接收下一帧的输入信息。在模式 2 中，接收到的停止位与 S8BUF、S8RB8 和 S8RI 无关。

通过软件对 S8CON 中的 S8SM2、S8TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式2）



接收数据（模式2）

19.9.10 串口 8 模式 3，模式 3 波特率计算公式

当 S8SM0、S8SM1 两位为 11 时，串口 8 工作在模式 3。串行通信模式 3 为 9 位数据异步通信 UART 模式，其一帧的信息由 11 位组成：1 位起始位，8 位数据位（低位在先），1 位可编程位（第 9 位数据）和 1 位停止位。发送时可编程位（第 9 位数据）由 S8CON 中的 S8TB8 提供，可软件设置为 1 或 0，或者可将 PSW 中的奇/偶校验位 P 值装入 S8TB8（S8TB8 既可作为多机通信中的地址数据标志位，又可作为数据的奇偶校验位）。接收时第 9 位数据装入 S8CON 的 S8RB8。TxD8 为发送端口，RxD8 为接收端口，以全双工模式进行接收/发送。

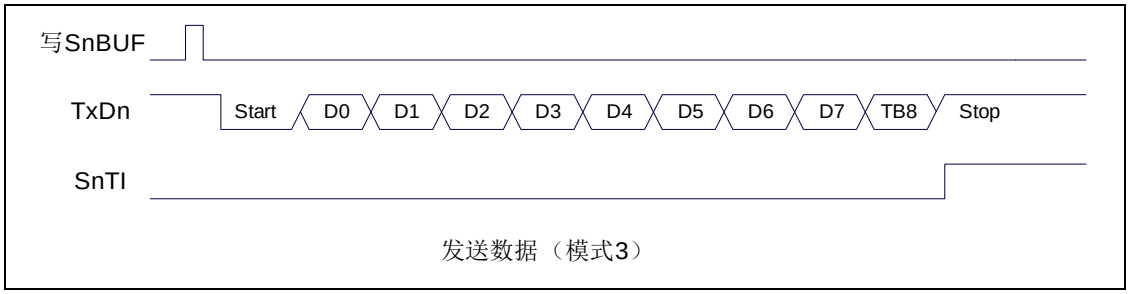
模式 3 和模式 1 相比，除发送时由 S8TB8 提供给移位寄存器第 9 数据位不同外，其余功能结构均基本相同，其接收‘发送操作过程及时序也基本相同。

当接收器接收完一帧信息后必须同时满足下列条件：

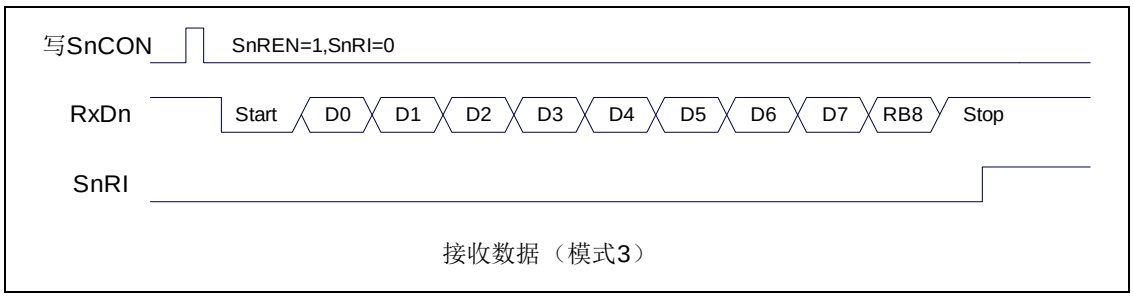
- S8RI=0
- S8SM2=0 或者 S8SM2=1 且接收到的第 9 数据位 S8RB8=1。

当上述两条件同时满足时，才将接收到的移位寄存器的数据装入 S8BUF 和 S8RB8 中，S8RI 标志位被置 1，并向主机请求中断处理。如果上述条件有一个不满足，则刚接收到移位寄存器中的数据无效而丢失，也不置位 S8RI。无论上述条件满足与否，接收器又重新开始检测 RxD8 输入端口的跳变信息，接收下一帧的输入信息。在模式 3 中，接收到的停止位与 S8BUF、S8RB8 和 S8RI 无关。

通过软件对 S8CON 中的 S8SM2、S8TB8 的设置以及通信协议的约定，为多机通信提供了方便。



发送数据（模式3）



接收数据（模式3）

串口 8 模式 3 的波特率计算公式与模式 1 是完全相同的。请参考模式 1 的波特率计算公式。

19.9.11 串口 8 自动地址识别，从机地址控制寄存器（S8ADDR，S8ADEN）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
S8ADDR	7EF7FEH								
S8ADEN	7EF7FFH								

S8ADDR：从机地址寄存器

S8ADEN：从机地址屏蔽位寄存器

自动地址识别功能典型应用在多机通讯领域，其主要原理是从机系统通过硬件比较功能来识别来自于主机串口 8 数据流中的地址信息，通过寄存器 S8ADDR 和 S8ADEN 设置的本机的从机地址，硬件自动对从机地址进行过滤，当来自于主机的从机地址信息与本机所设置的从机地址相匹配时，硬件产生串口 8 中断；否则硬件自动丢弃串口 8 数据，而不产生中断。当众多处于空闲模式的从机链接在一起时，只有从机地址相匹配的从机才会从空闲模式唤醒，从而可以大大降低从机 MCU 的功耗，即使从机处于正常工作状态也可避免不停地进入串口 8 中断而降低系统执行效率。

要使用串口 8 的自动地址识别功能，首先需要将参与通讯的 MCU 的串口 8 通讯模式设置为模式 1，并开启从机的 S8CON 的 S8SM2 位。对于串口 8 模式 1 的 9 位数据位中，第 9 位数据（存放在 S8RB8 中）为地址/数据的标志位，当第 9 位数据为 1 时，表示前面的 8 位数据（存放在 S8BUF 中）为地址信息。当 S8SM2 被设置为 1 时，从机 MCU 会自动过滤掉非地址数据（第 9 位为 0 的数据），而对 S8BUF 中的地址数据（第 9 位为 1 的数据）自动与 S8ADDR 和 S8ADEN 所设置的本机地址进行比较，若地址相匹配，则会将 S8RI 置“1”，并产生中断，否则不予处理本次接收的串口 8 数据。

从机地址的设置是通过 S8ADDR 和 S8ADEN 两个寄存器进行设置的。S8ADDR 为从机地址寄存器，里面存放本机的从机地址。S8ADEN 为从机地址屏蔽位寄存器，用于设置地址信息中的忽略位，设置方法如下：

例如
S8ADDR ＝ 11001010
S8ADEN ＝ 10000001
则匹配地址为 1xxxxxx0
即，只要主机送出的地址数据中的 bit0 为 0 且 bit7 为 1 就可以和本机地址相匹配

再例如
S8ADDR ＝ 11001010
S8ADEN ＝ 00001111
则匹配地址为 xxxx1010
即，只要主机送出的地址数据中的低 4 位为 1010 就可以和本机地址相匹配，而高 4 为被忽略，可以为任意值。

主机可以使用广播地址（FFH）同时选中所有的从机来进行通讯。

19.9.12 串口 8 同步模式控制寄存器 1（USART8CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART8CR1	7EF7D8H	S8LINEN	S8DORD	S8CLKEN	S8SPMOD	S8SPIEN	S8SPSLV	S8CPOL	S8CPHA

S8LINEN：LIN 模式使能位

0：禁止 LIN 模式

1：使能 LIN 模式

S8DORD：SPI 模式数据位发送/接收的顺序

0：先发送/接收数据的高位（MSB）

1：先发送/接收数据的低位（LSB）

S8CLKEN：SmartCard 模式时钟输出控制位

0：禁止时钟输出

1：使能时钟输出

S8SPMOD：SPI 模式使能位

0：禁止 SPI 模式

1：使能 SPI 模式

S8SPIEN：SPI 使能位

0：禁止 SPI 功能

1：使能 SPI 功能

当 USART8 需要工作在 SPI 模式时，必须先将 SPI 相关的模式寄存器、速度寄存器、时钟极性寄存器设置完成后，最后再设置 SPIEN 寄存器。

S8SPSLV：SPI 从机模式使能位

0：SPI 为主机模式

1：SPI 为从机模式

S8CPOL：SPI 时钟极性控制

0：SCLK 空闲时为低电平，SCLK 的前时钟沿为上升沿，后时钟沿为下降沿

1：SCLK 空闲时为高电平，SCLK 的前时钟沿为下降沿，后时钟沿为上升沿

S8CPHA：SPI 时钟相位控制

0：数据 SS 管脚为低电平驱动第一位数据并在 SCLK 的后时钟沿改变数据，前时钟沿采样数据

1：数据在 SCLK 的前时钟沿驱动，后时钟沿采样

19.9.13 串口 8 同步模式控制寄存器 2（USART8CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART8CR2	7EF7D9H	S8IREN	S8IRLP	S8SCEN	S8NACK	S8HDSEL	S8PCEN	S8PS	S8PE

S8IREN：IrDA 模式使能位

0：禁止 IrDA 模式

1：使能 IrDA 模式

S8IRLP：IrDA 低电模式控制位

0：IrDA 为普通模式

1：IrDA 为低电模式

S8SCEN：SmartCard 模式使能位

0：禁止 SmartCard 模式

1：使能 SmartCard 模式

S8NACK：SmartCard 模式 NACK 输出使能位

0：禁止输出 NACK 信号

1：使能输出 NACK 信号

S8HDSEL：单线半双工模式使能位

0：禁止单线半双工模式

1：使能单线半双工模式

S8PCEN：硬件自动产生校验位控制使能

0：禁止硬件自动产生校验位（串口的校验位为 S8TB8 设置的值）

1：使能硬件自动产生校验位

S8PS：硬件校验位模式选择

0：硬件根据 S8BUF 的值自动产生偶校验位

1：硬件根据 S8BUF 的值自动产生奇校验位

S8PE：校验位错误标志（必须软件清零）

0：无检验错误

1：有校验错误（串口接收 DMA 过程中如果发生接收数据校验位错误，DMA 不会停止，但校验位错误标志会一直保持直到 DMA 完成）

19.9.14 串口 8 同步模式控制寄存器 3（USART8CR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART8CR3	7EF7DAH	S8IrDA_LPBAUD[7:0]							

S8IrDA_LPBAUD：IrDA 低电模式波特率控制寄存器

IrDA_LPBAUD[7:0]	IrDA 低电模式波特率
0	SYSclk/16/256
1	SYSclk/16/1
2	SYSclk/16/2
...	...
255	SYSclk/16/255

19.9.15 串口 8 同步模式控制寄存器 4（USART8CR4）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART8CR4	7EF7DBH	-	-	-	-	S8SCCKS[1:0]		S8SPICKS[1:0]	

S8SCCKS[1:0]：SmartCard 模式时钟选择

SCCKS[1:0]	SmartCard 模式时钟
00	SYSclk/64
01	SYSclk/32
10	SYSclk/16
11	SYSclk/8

S8SPICKS[1:0]：SPI 模式时钟选择

SPICKS[1:0]	SPI 模式时钟
00	SYSclk/4
01	SYSclk/8
10	SYSclk/16
11	SYSclk/2

19.9.16 串口 8 同步模式控制寄存器 5（USART8CR5）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART8CR5	7EF7DCH	S8BRKDET	S8HDRER	S8SLVEN	S8ASYNC	S8TXCF	S8SENDBK	S8HDRDET	S8SYNC

S8BRKDET：LIN 从模式间隔场（BREAK）检测标志

0：未检测到 LIN 间隔场

1：检测到 LIN 间隔场，需要软件清零

S8HDRER：LIN 从模式报文头（HEADER）错误

0：未检测到 LIN 报文头错误

1：检测到 LIN 报文头错误，需要软件清零

S8SLVEN：LIN 从机模式使能位

0：LIN 为主机模式

1：LIN 为从机模式

S8ASYNC：LIN 自动同步功能使能位

0：禁止自动同步

1：使能自动同步

【注意】：从机使能自动同步功能后，如果从机跟主机波特率存在小范围误差，从机收到主机同步信号后，会自动将从机波特率设置成与主机相同的波特率。

例如，主机波特率 9600：

从机使能自动同步时，如果从机波特率在 9100～10000 附近，收到主机同步信号后，从机会自动将本身波特率设置为 9600。误差自动同步范围 -5.2%～+4.16%左右。

S8TXCF：传输冲突标志位。单线半双工模式、LIN 模式和 SmartCard 模式有效

0：未检测到传输冲突（发送的数据与接收的数据相同）

1：检测到传输冲突（发送的数据与接收的数据不同）

S8SENDBK：发送间隔场。软件写 1 触发发送间隔场，发送完成后硬件自动清 0

S8HDRDET：LIN 从模式报文头（HEADER）检测标志

0：未检测到 LIN 报文头

1：检测到 LIN 报文头，需要软件清零

S8SYNC：LIN 从模式同步场检测标志。

正确分析到同步场后，硬件将 SYNC 标志为置 1。在接收标志符场时硬件会自动清零 SYNC

19.9.17 串口 8 同步模式保护时间寄存器（USART8GTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART8GTR	7EF7DDH								

USART8GTR 寄存器仅在 SmartCard 模式有效。该寄存器是根据波特率时钟数给出保护时间值。S4TI 标志在 SmartCard 发送的数据位等于 USART8GTR 寄存器所设置的保护时间值时被硬件置 1。注：USART8GTR 寄存器的值应大于 11

19.9.18 串口 8 同步模式波特率寄存器（USART8BR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USART8BRH	7EF7DEH	USART8BR[15:8]							
USART8BRL	7EF7DFH	USART8BR[7:0]							

LIN 模式、IrDA 模式和 SmartCard 模式时的波特率计算公式

$$\text{同步波特率} = \frac{\text{SYSclk}}{16 * \text{USART8BR[15:0]}}$$

19.9.19 串口 8 接收超时控制寄存器（UR8TOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR8TOCR	7EF7D4H	UR8ENTO	UR8ENTOI	UR8SCALE	-	-	-	-	-

UR8ENTO：串口 8 接收超时功能控制位

0：禁止串口 8 接收超时功能

1：使能串口 8 接收超时功能

UR8ENTOI：串口 8 接收超时中断控制位

0：禁止串口 8 接收超时中断

1：使能串口 8 接收超时中断

UR8SCALE：串口 8 超时计数时钟源选择

0：串口数据位率（波特率）

1：系统时钟

19.9.20 串口 8 超时状态寄存器（UR8TOSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR8TOSR	7EF7D5H	UR8CTOIF	-	-	-	-	-	-	UR8TOIF

UR8CTOIF：写“1”清除串口超时中断标志位 TOIF。（只写）

UR8TOIF：串口 8 超时中断请求标志位。（只读）

当发生串口 8 超时时，TOIF 会被硬件置“1”。如果 ENTOI 为 1 时会产生串口中断，中断入口地址为原串口 8 的中断入口地址。标志位需要软件向 CTOIF 位写“1”清零

19.9.21 串口 8 超时长度控制寄存器（UR8TOTE/H/L）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UR8TOTE	7EF7D3H	UR8TM[23:16]							
UR8TOTH	7EF7D6H	UR8TM[15:8]							
UR8TOTL	7EF7D7H	UR8TM[7:0]							

UR8TM[23:0]：串口 8 超时时间控制位。

当串口 8 处于接收空闲状态时，内部超时计数器根据 SCALE 寄存器所选择的时钟源进行计数，当计数时间达到 UR8TM 所设置的超时时间时，便会产生超时中断。当串口 8 接收数据完成时，复位内部超时计数器，重新进行超时计数。

注：

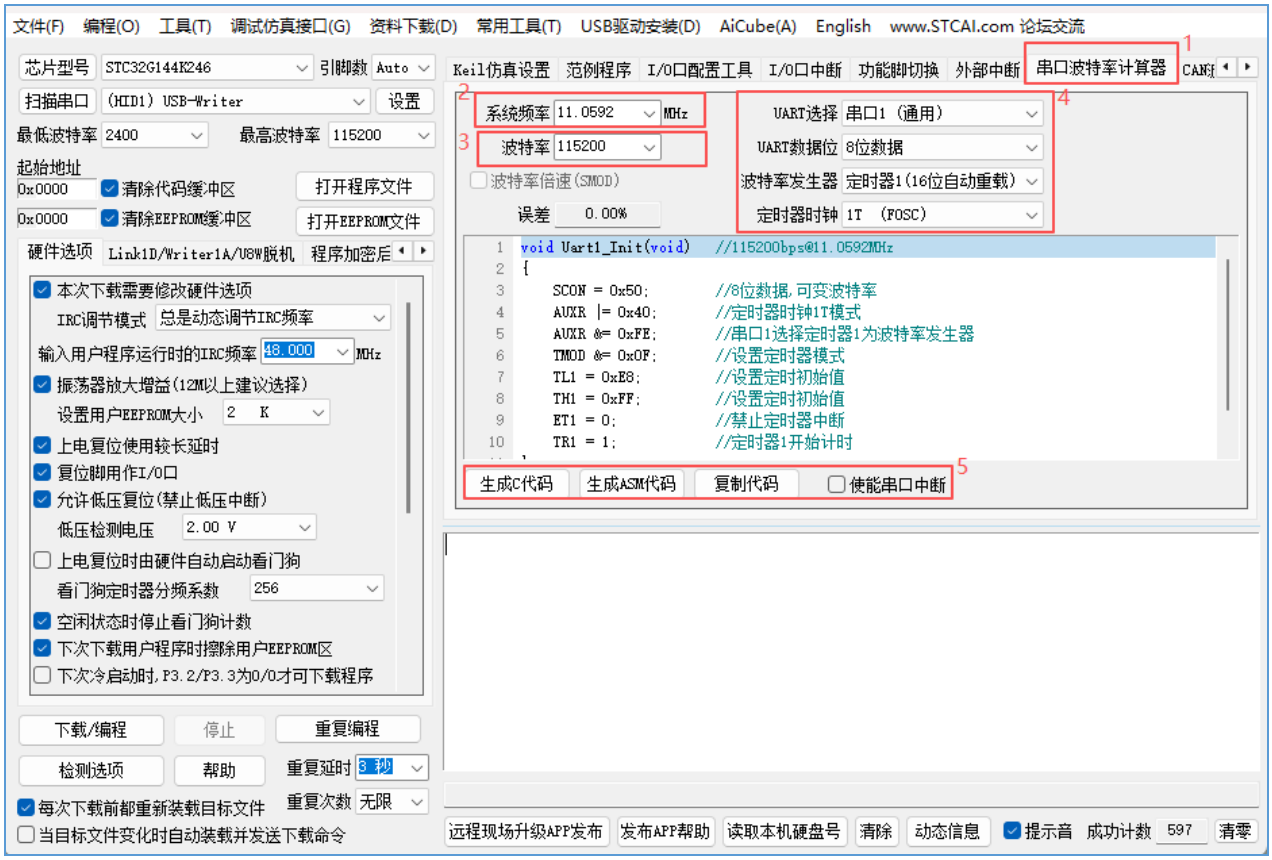
1、如果需要使能接收超时中断功能，则 TM 不可设置为 0，且 UR8TOTL、UR8TOTH、UR8TOTE 寄存器的设置必须先设置 UR8TOTL 和 UR8TOTH，最后设置 UR8TOTE

2、必须使能串口 8 接收才有接收超时功能

3、接收超时功能必须在正确接收到一字节数据后才能触发

4、正在接收过程中，无接收超时功能

19.10 AiCube-ISP | 串口波特率计算器工具



①：在下载软件中选择“串口波特率计算器”功能页，进入串口代码生成界面

②：设置系统工作频率（单位：MHz）

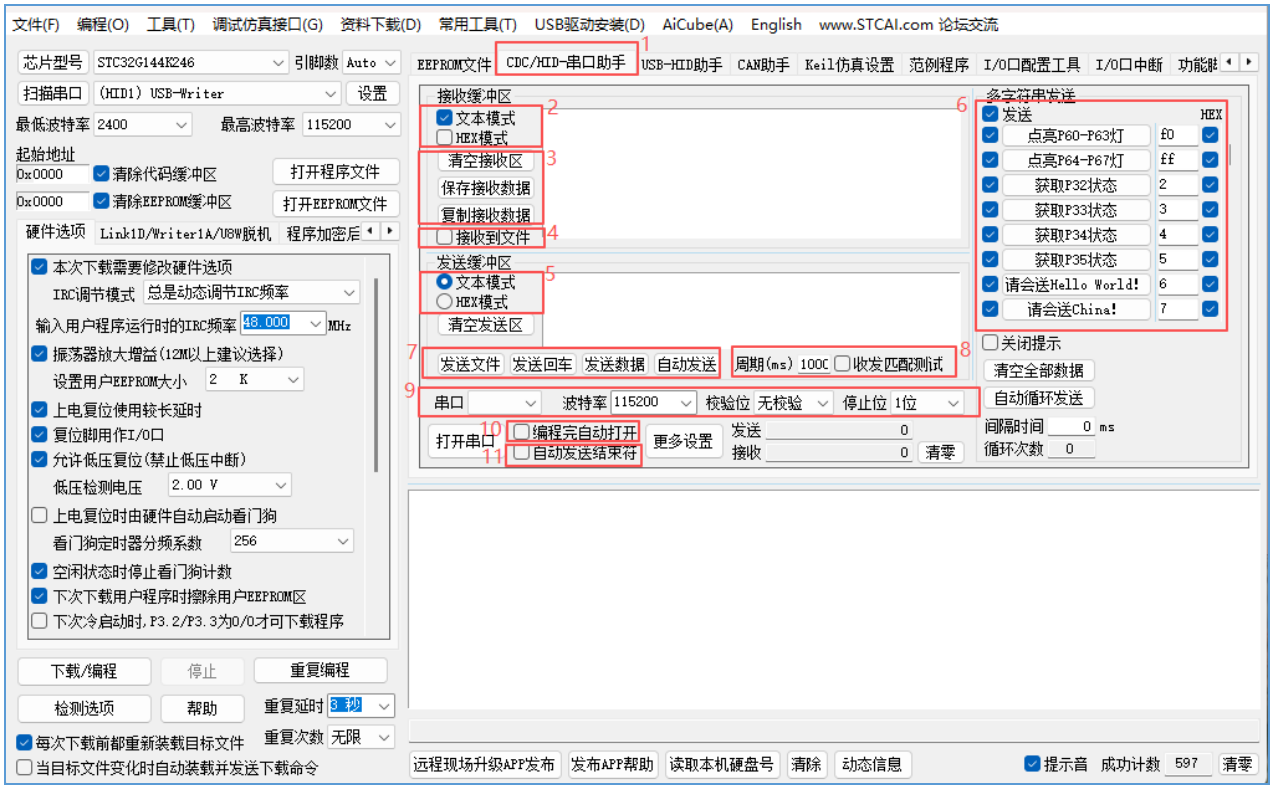
③：设置串口波特率

④：选择目标串口，并设置串口模式、波特率发生器等参数

⑤：手动生成 C 代码或者 ASM 代码，复制范例

19.11 AiCube-ISP | 串口助手/USB-CDC

串口助手主界面



①：在下载软件中选择“USB-CDC/串口助手”功能页，进入串口助手界面

②：选择接收数据的显示格式

③：保存/复制接收的数据

④：设置接收数据自动存储到文件

⑤：选择发送数据的格式

⑥：“多字符串”控制界面

⑦：数据/文件发送按钮

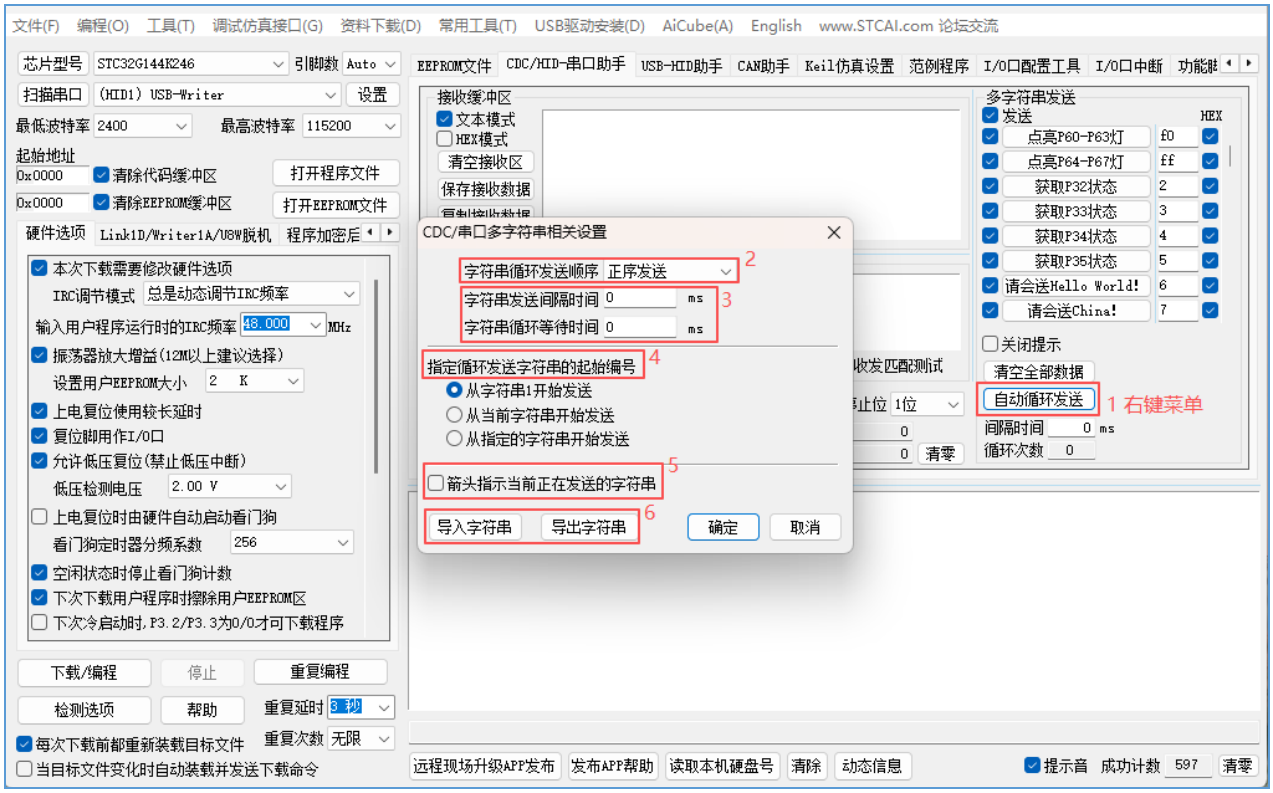
⑧：设置重复下载的周期

⑨：选择串口号，设置串口参数

⑩：设置 ISP 下载完成后是否自动打开串口

⑪：设置发送完成数据后，是否自动发送结束符

多字符串设置界面



①：鼠标右键点击“自动循环发送”按钮，点击右键菜单“设置...”进入多字符串设置界面

②：设置多字符串循环发送顺序

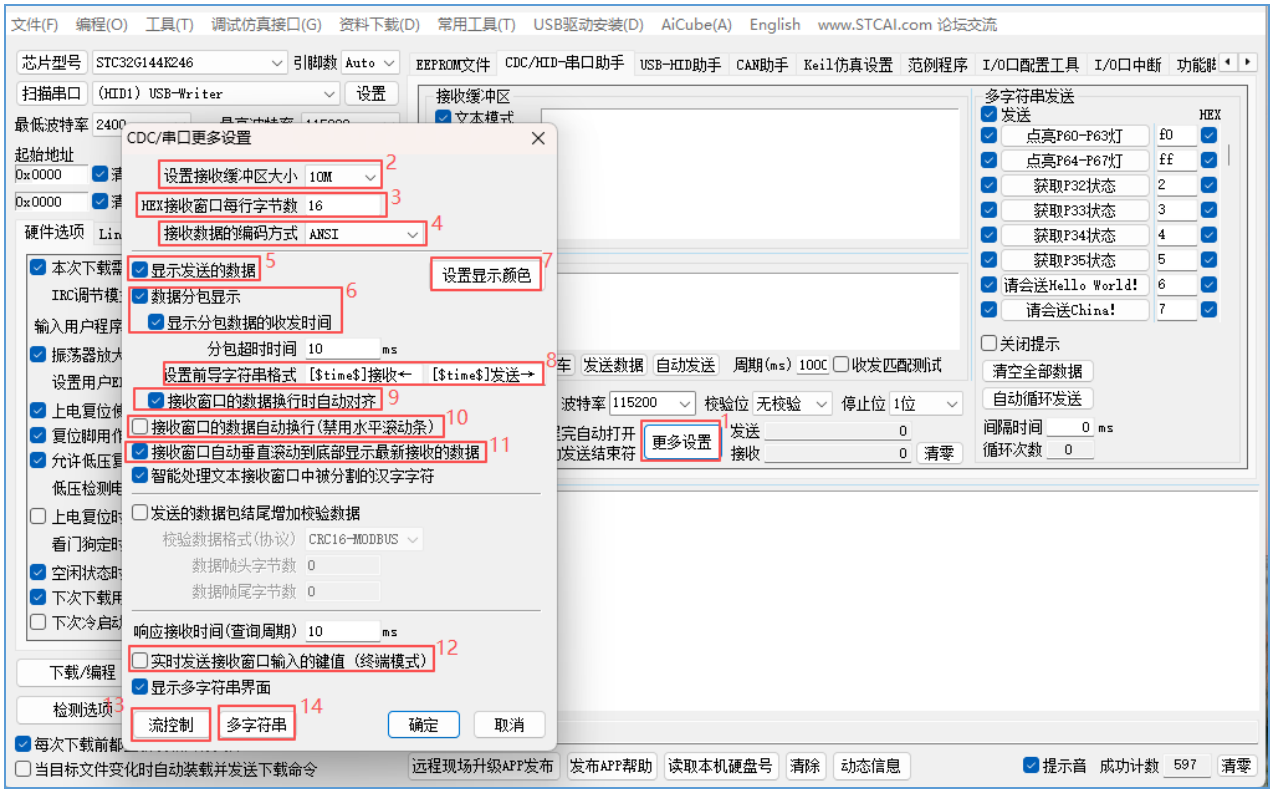
③：设置多字符串循环发送间隔时间

④：设置多字符串循环发送的起始编号

⑤：设置是否需要用箭头指示当前正在发送的字符串

⑥：多字符串导出到文件或从文件导入

串口助手更多设置



①：点击“更多设置”按钮，进入串口助手更多设置界面

②：设置接收缓存大小（缓存越大，软件反应越慢）

③：设置 HEX 接收数据每行显示的数据个数

④：设置接收数据的汉字编码格式

支持如下汉字编码格式：

ANSI：GB2312 汉字编码

UTF8：UNICODE 互联网常用编码

UTF16-LE：小端 UTF16 编码

UTF16=BE：大端 UTF16 编码

⑤：设置是否显示发送的数据

⑥：设置数据是否自动分包显示

⑦：设置界面的显示颜色

⑧：设置接收窗口数据显示的前导字符

⑨：设置接收数据的换行显示模式

⑩：设置接收窗口是否自动换行

⑪：设置发送数据是否自动追加校验数据

支持如下校验格式：

ADD8：字节校验和

ADD8N：字节校验和补码

ADD16-LE：小端双字节校验和

ADD16-BE：大端双字节校验和

XOR8：字节异或

CRC16-MODBUS：MODBUS 协议的 CRC16

CRC16-USB：USB 协议的 CRC16

CRC16-XMODEM：XMODEM 协议的 CRC16

CRC32：32 位 CRC 校验

⑫：设置是否使能终端模式（终端模式：将光标定位到接收窗口，按下键盘按键实时发送相应的键码）
⑬：进入流控制设置界面
⑭：进入多字符串界面设置界面

20 4 组 6P6N 比较器，掉电检测，内部固定比较电压

产品线	比较器（6P+6N）
STC32G144K246 系列	**4 组**

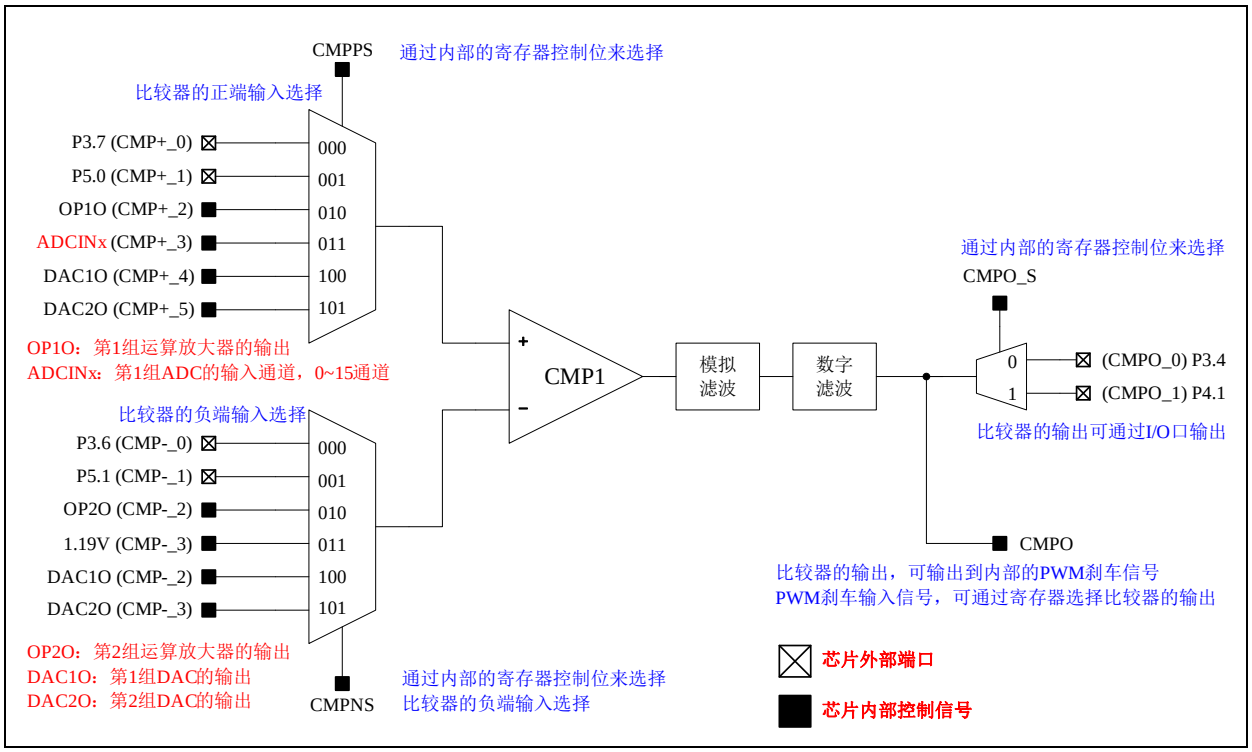
STC32G144K246 系列单片机内部集成了 4 组 6P＋6N 的比较器。通过多路选择器和分时复用，每组比较器都可实现多个比较器的应用。

比较器内部有可程序控制的两级滤波：模拟滤波和数字滤波。模拟滤波可以过滤掉比较输入信号中的毛刺信号，数字滤波可以等待输入信号更加稳定后再进行比较。比较结果可直接通过读取内部寄存器位获得，也可将比较器结果正向或反向输出到外部端口。将比较结果输出到外部端口可用作外部事件的触发信号和反馈信号，可扩大比较的应用范围。

当比较器正极选择 ADC/ADC2 的输入通道时，请务必要打开 ADC_CONTR/ADC2_CONTR 寄存器中的 ADC 电源控制位 ADC_POWER/ADC2_POWER 和 ADC 通道选择位 ADC_CHS/ADC2_CHS。

20.1 比较器 1

20.1.1 比较器 1 内部结构图



20.1.2 比较器 1 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP1_6P6N_CR3	7EFB72H	CMP1CHYS[1:0]		**CMP1NS[2:0]**			**CMP1PS[2:0]**		
CMP1_6P6N_CR4	7EFB73H	-	-	-	-	-	-	**CMP1O_S[1:0]**	

CMP1PS[2:0]：比较器正端输入选择

CMP1NS[2:0]：比较器负端输入选择

CMP1O_S[1:0]：比较器输出端口选择

CMP1PS[2:0]	CMP1+	CMP1NS[2:0]	CMP1-	CMP1O_S[2:0]	CMP1O
000	P3.7	000	P3.6	00	P3.4
001	P5.0	001	P5.1	01	P4.1
010	OP1O	010	OP2O	10	-
011	ADCIN	011	1.19V	11	-
100	DAC1O	100	DAC1O		
101	DAC2O	101	DAC2O		

20.1.3 比较器 1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CMP1_6P6N_CR1	比较器 1 控制寄存器 1	7EFB70H	CMP1EN	CMP1IF	CMP1PIE	CMP1NIE	CMP1IP[1:0]		CMP1OE	CMP1RES	0000,0000
CMP1_6P6N_CR2	比较器 1 控制寄存器 2	7EFB71H	INVCMP1O	CMP1DISFLT	CMP1LCDTY[5:0]						0000,0000
CMP1_6P6N_CR3	比较器 1 控制寄存器 3	7EFB72H	CMP1CHYS[1:0]		CMP1NS[2:0]			CMP1PS[2:0]			0000,0000
CMP1_6P6N_CR4	比较器 1 控制寄存器 4	7EFB73H	-	-	-	-	-	-	CMP1O_S[1:0]		xxxx,xx00

20.1.4 比较器 1 控制寄存器 1（CMP1_6P6N_CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP1_6P6N_CR1	7EFB70H	CMP1EN	CMP1IF	CMP1PIE	CMP1NIE	CMP1IP[1:0]		CMP1OE	CMP1RES

CMP1EN：比较器模块使能位

0：关闭比较功能

1：使能比较功能

CMP1IF：比较器中断标志位。当 PIE 或 NIE 被使能后，若产生相应的中断信号，硬件自动将 CMPIF 置 1，并向 CPU 提出中断请求。此标志位必须用户软件向此位写 0 清零。

CMP1PIE：比较器上升沿中断使能位。

0：禁止比较器上升沿中断。

1：使能比较器上升沿中断。使能比较器的比较结果由 0 变成 1 时产生中断请求。

CMP1NIE：比较器下降沿中断使能位。

0：禁止比较器下降沿中断。

1：使能比较器下降沿中断。使能比较器的比较结果由 1 变成 0 时产生中断请求。

CMP1IP[1:0]：比较器中断优先级设置。

00：比较器中断为最低优先级。

01：比较器中断为较低优先级。

10：比较器中断为较高优先级。

11：比较器中断为最高优先级。

CMP1OE：比较器结果输出控制位

0：禁止比较器结果输出

1：使能比较器结果输出。

CMP1RES：比较器的比较结果。此位为只读。

0：表示 CMP1+的电平低于 CMP1-的电平

1：表示 CMP1+的电平高于 CMP1-的电平

CMPRES 是经过数字滤波后的输出信号，而不是比较器的直接输出结果。

20.1.5 比较器 1 控制寄存器 2（CMP1_6P6N_CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP1_6P6N_CR2	7EFB71H	INVCMP1O	CMP1DISFLT	CMP1LCDTY[5:0]					

INVCMP1O：比较器结果输出控制

0：比较器结果正向输出。若 CMP1RES 为 0，则输出低电平，反之输出高电平。

1：比较器结果反向输出。若 CMP1RES 为 0，则输出高电平，反之输出低电平。

CMP1DISFLT：模拟滤波功能控制

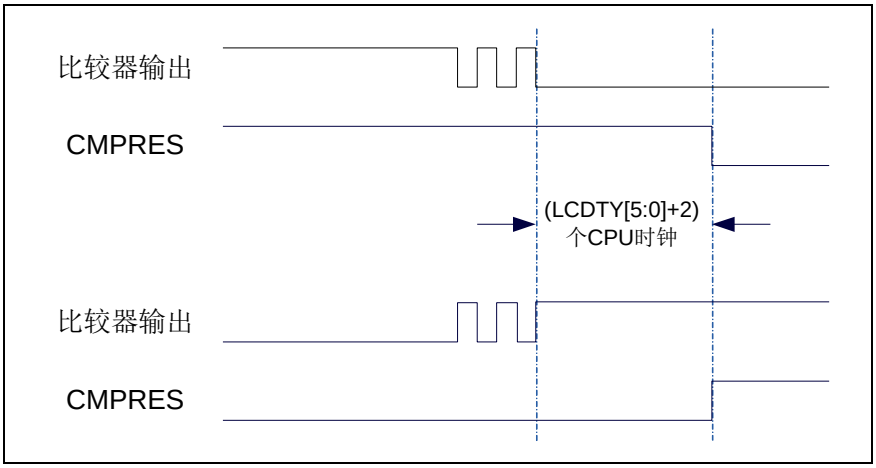
0：使能 0.1us 模拟滤波功能

1：关闭 0.1us 模拟滤波功能，可略微提高比较器的比较速度。

CMP1LCDTY[5:0]：数字滤波功能控制

数字滤波功能即为数字信号去抖动功能。当比较结果发生上升沿或者下降沿变化时，比较器侦测变化后的信号必须维持 LCDTY 所设置的 CPU 时钟数不发生变化，才认为数据变化是有效的；否则将视同信号无变化。

注意：当使能数字滤波功能后，芯片内部实际的等待时钟需额外增加两个状态机切换时间，即若 LCDTY 设置为 0 时，为关闭数字滤波功能；若 LCDTY 设置为非 0 值 n（n=1~63）时，则实际的数字滤波时间为（n+2）个系统时钟



20.1.6 比较器 1 控制寄存器 3（CMP1_6P6N_CR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP1_6P6N_CR3	7EFB72H	CMP1CHYS[1:0]		CMP1NS[2:0]			CMP1PS[2:0]		

CMP1CHYS[1:0]：比较器 DC 迟滞输入选择

CHYS [1:0]	比较器 DC 迟滞输入选择
00	0mV
01	10mV
10	20mV
11	30mV

CMP1NS[2:0]：比较器负端输入选择

CMP1NS[2:0]	CMP1-
000	P3.6
001	P5.1
010	OP2O
011	1.19V
100	DAC1O
101	DAC2O

CMPnPS[2:0]：比较器正端输入选择

CMP1PS[2:0]	CMP1+
000	P3.7
001	P5.0
010	OP1O
011	ADCIN
100	DAC1O
101	DAC2O

20.1.7 比较器 1 控制寄存器 4（CMP1_6P6N_CR4）

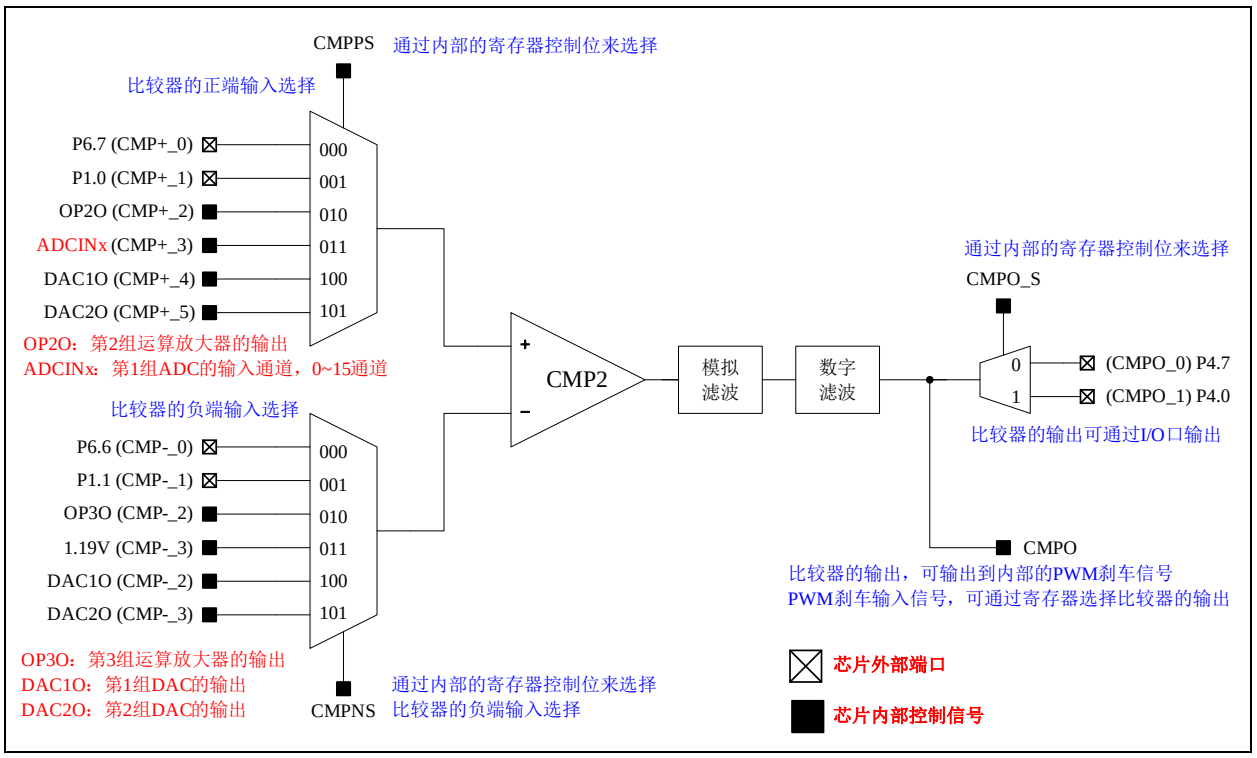
符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP1_6P6N_CR4	7EFB73H	-	-	-	-	-	-	CMP1O_S[1:0]	

CMPnO_S[1:0]：比较器输出端口选择

CMP1O_S[2:0]	CMP1O
00	P3.4
01	P4.1
10	-
11	-

20.2 比较器 2

20.2.1 比较器 2 内部结构图



20.2.2 比较器 2 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP2_6P6N_CR3	7EFB76H	CMP2CHYS[1:0]		**CMP2NS[2:0]**			**CMP2PS[2:0]**		
CMP2_6P6N_CR4	7EFB77H	-	-	-	-	-	-	**CMP2O_S[1:0]**	

CMP2PS[2:0]：比较器正端输入选择

CMP2NS[2:0]：比较器负端输入选择

CMP2O_S[1:0]：比较器输出端口选择

CMP2PS[2:0]	CMP2+	CMP2NS[2:0]	CMP2-	CMP2O_S[2:0]	CMP2O
000	P6.7	000	P6.6	00	P4.7
001	P1.0	001	P1.1	01	P4.0
010	OP2O	010	OP3O	10	-
011	ADCIN	011	1.19V	11	-
100	DAC1O	100	DAC1O		
101	DAC2O	101	DAC2O		

20.2.3 比较器 2 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CMP2_6P6N_CR1	比较器 2 控制寄存器 1	7EFB74H	CMP2EN	CMP2IF	CMP2PIE	CMP2NIE	CMP2IP[1:0]		CMP2OE	CMP2RES	0000,0000
CMP2_6P6N_CR2	比较器 2 控制寄存器 2	7EFB75H	INVCMP2O	CMP2DISFLT	CMP2LCDTY[5:0]						0000,0000
CMP2_6P6N_CR3	比较器 2 控制寄存器 3	7EFB76H	CMP2CHYS[1:0]		CMP2NS[2:0]			CMP2PS[2:0]			0000,0000
CMP2_6P6N_CR4	比较器 2 控制寄存器 4	7EFB77H	-	-	-	-	-	-	CMP2O_S[1:0]		xxxx,xx00

20.2.4 比较器 2 控制寄存器 1（CMP2_6P6N_CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP2_6P6N_CR1	7EFB74H	CMP2EN	CMP2IF	CMP2PIE	CMP2NIE	CMP2IP[1:0]		CMP2OE	CMP2RES

CMP2EN：比较器模块使能位

0：关闭比较功能

1：使能比较功能

CMP2IF：比较器中断标志位。当 PIE 或 NIE 被使能后，若产生相应的中断信号，硬件自动将 CMPIF 置 1，并向 CPU 提出中断请求。此标志位必须用户软件向此位写 0 清零。

CMP2PIE：比较器上升沿中断使能位。

0：禁止比较器上升沿中断。

1：使能比较器上升沿中断。使能比较器的比较结果由 0 变成 1 时产生中断请求。

CMP2NIE：比较器下降沿中断使能位。

0：禁止比较器下降沿中断。

1：使能比较器下降沿中断。使能比较器的比较结果由 1 变成 0 时产生中断请求。

CMP2IP[1:0]：比较器中断优先级设置。

00：比较器中断为最低优先级。

01：比较器中断为较低优先级。

10：比较器中断为较高优先级。

11：比较器中断为最高优先级。

CMP2OE：比较器结果输出控制位

0：禁止比较器结果输出

1：使能比较器结果输出。

CMP2RES：比较器的比较结果。此位为只读。

0：表示 CMP2+的电平低于 CMP2-的电平

1：表示 CMP2+的电平高于 CMP2-的电平

CMPRES 是经过数字滤波后的输出信号，而不是比较器的直接输出结果。

20.2.5 比较器 2 控制寄存器 2（CMP2_6P6N_CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP2_6P6N_CR2	7EFB75H	INVCMP2O	CMP2DISFLT	CMP2LCDTY[5:0]					

INVCMP2O：比较器结果输出控制

0：比较器结果正向输出。若 CMP2RES 为 0，则输出低电平，反之输出高电平。

1：比较器结果反向输出。若 CMP2RES 为 0，则输出高电平，反之输出低电平。

CMP2DISFLT：模拟滤波功能控制

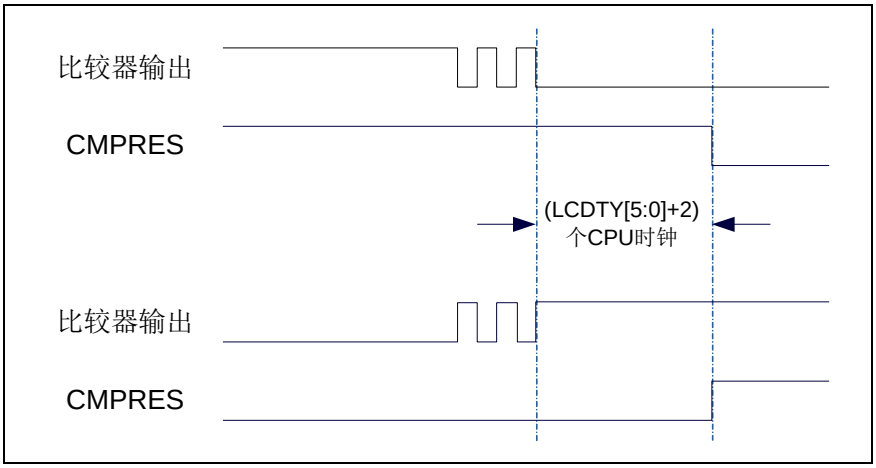
0：使能 0.1us 模拟滤波功能

1：关闭 0.1us 模拟滤波功能，可略微提高比较器的比较速度。

CMP2LCDTY[5:0]：数字滤波功能控制

数字滤波功能即为数字信号去抖动功能。当比较结果发生上升沿或者下降沿变化时，比较器侦测变化后的信号必须维持 LCDTY 所设置的 CPU 时钟数不发生变化，才认为数据变化是有效的；否则将视同信号无变化。

注意：当使能数字滤波功能后，芯片内部实际的等待时钟需额外增加两个状态机切换时间，即若 LCDTY 设置为 0 时，为关闭数字滤波功能；若 LCDTY 设置为非 0 值 n（n=1~63）时，则实际的数字滤波时间为（n+2）个系统时钟



20.2.6 比较器 2 控制寄存器 3（CMP2_6P6N_CR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP2_6P6N_CR3	7EFB76H	CMP2CHYS[1:0]		CMP2NS[2:0]			CMP2PS[2:0]		

CMP2CHYS[1:0]：比较器 DC 迟滞输入选择

CHYS [1:0]	比较器 DC 迟滞输入选择
00	0mV
01	10mV
10	20mV
11	30mV

CMP2NS[2:0]：比较器负端输入选择

CMP2NS[2:0]	CMP2-
000	P6.6
001	P1.1
010	OP3O
011	1.19V
100	DAC1O
101	DAC2O

CMP2PS[2:0]：比较器正端输入选择

CMP2PS[2:0]	CMP2+
000	P6.7
001	P1.0
010	OP2O
011	ADCIN
100	DAC1O
101	DAC2O

20.2.7 比较器 2 控制寄存器 4（CMP2_6P6N_CR4）

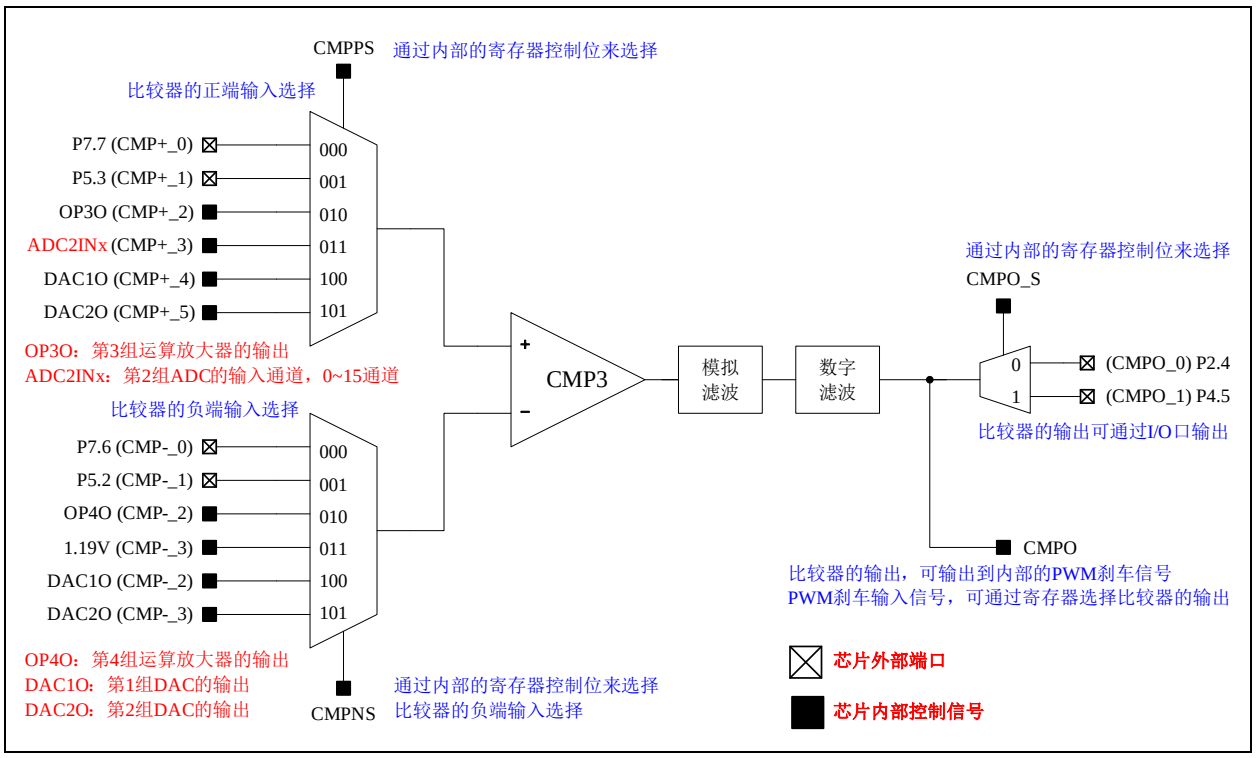
符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP2_6P6N_CR4	7EFB77H	-	-	-	-	-	-	CMP2O_S[1:0]	

CMP2O_S[1:0]：比较器输出端口选择

CMP2O_S[2:0]	CMP2O
00	P4.7
01	P4.0
10	-
11	-

20.3 比较器 3

20.3.1 比较器 3 内部结构图



20.3.2 比较器 3 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP3_6P6N_CR3	7EFB7AH	CMP3CHYS[1:0]		**CMP3NS[2:0]**			**CMP3PS[2:0]**		
CMP3_6P6N_CR4	7EFB7BH	-	-	-	-	-	-	**CMP3O_S[1:0]**	

CMP3PS[2:0]：比较器正端输入选择

CMP3NS[2:0]：比较器负端输入选择

CMP3O_S[1:0]：比较器输出端口选择

CMP3PS[2:0]	CMP3+	CMP3NS[2:0]	CMP3-	CMP3O_S[2:0]	CMP3O
000	P7.7	000	P7.6	00	P2.4
001	P5.3	001	P5.2	01	P4.5
010	OP3O	010	OP4O	10	-
011	ADC2IN	011	1.19V	11	-
100	DAC1O	100	DAC1O		
101	DAC2O	101	DAC2O		

20.3.3 比较器 3 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CMP3_6P6N_CR1	比较器 3 控制寄存器 1	7EFB78H	CMP3EN	CMP3IF	CMP3PIE	CMP3NIE	CMP3IP[1:0]		CMP3OE	CMP3RES	0000,0000
CMP3_6P6N_CR2	比较器 3 控制寄存器 2	7EFB79H	INVCMP3O	CMP3DISFLT	CMP3LCDTY[5:0]						0000,0000
CMP3_6P6N_CR3	比较器 3 控制寄存器 3	7EFB7AH	CMP3CHYS[1:0]		CMP3NS[2:0]			CMP3PS[2:0]			0000,0000
CMP3_6P6N_CR4	比较器 3 控制寄存器 4	7EFB7BH	-	-	-	-	-	-	CMP3O_S[1:0]		xxxx,xx00

20.3.4 比较器 3 控制寄存器 1（CMP3_6P6N_CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP3_6P6N_CR1	7EFB78H	CMP3EN	CMP3IF	CMP3PIE	CMP3NIE	CMP3IP[1:0]		CMP3OE	CMP3RES

CMP3EN：比较器模块使能位

0：关闭比较功能

1：使能比较功能

CMP3IF：比较器中断标志位。当 PIE 或 NIE 被使能后，若产生相应的中断信号，硬件自动将 CMPIF 置 1，并向 CPU 提出中断请求。此标志位必须用户软件向此位写 0 清零。

CMP3PIE：比较器上升沿中断使能位。

0：禁止比较器上升沿中断。

1：使能比较器上升沿中断。使能比较器的比较结果由 0 变成 1 时产生中断请求。

CMP3NIE：比较器下降沿中断使能位。

0：禁止比较器下降沿中断。

1：使能比较器下降沿中断。使能比较器的比较结果由 1 变成 0 时产生中断请求。

CMP3IP[1:0]：比较器中断优先级设置。

00：比较器中断为最低优先级。

01：比较器中断为较低优先级。

10：比较器中断为较高优先级。

11：比较器中断为最高优先级。

CMP3OE：比较器结果输出控制位

0：禁止比较器结果输出

1：使能比较器结果输出。

CMP3RES：比较器的比较结果。此位为只读。

0：表示 CMP3+的电平低于 CMP3-的电平

1：表示 CMP3+的电平高于 CMP3-的电平

CMPRES 是经过数字滤波后的输出信号，而不是比较器的直接输出结果。

20.3.5 比较器 3 控制寄存器 2（CMP3_6P6N_CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP3_6P6N_CR2	7EFB79H	INVCMP3O	CMP3DISFLT	CMP3LCDTY[5:0]					

INVCMP3O：比较器结果输出控制

0：比较器结果正向输出。若 CMP3RES 为 0，则输出低电平，反之输出高电平。

1：比较器结果反向输出。若 CMP3RES 为 0，则输出高电平，反之输出低电平。

CMP3DISFLT：模拟滤波功能控制

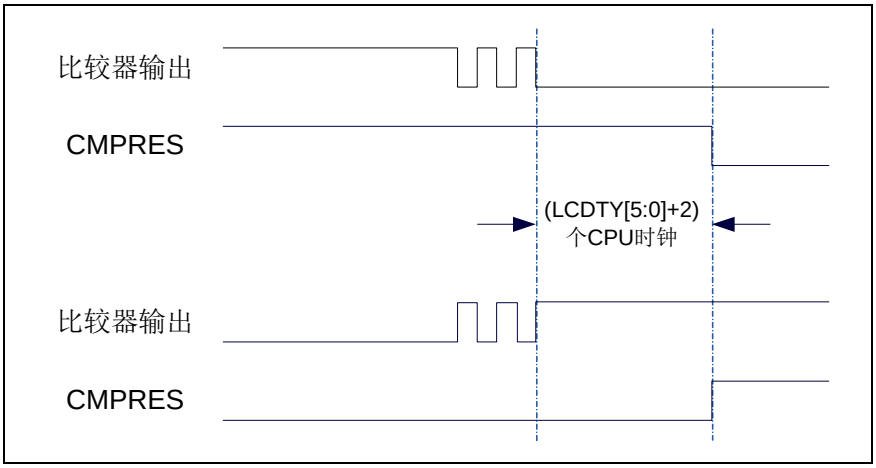
0：使能 0.1us 模拟滤波功能

1：关闭 0.1us 模拟滤波功能，可略微提高比较器的比较速度。

CMP3LCDTY[5:0]：数字滤波功能控制

数字滤波功能即为数字信号去抖动功能。当比较结果发生上升沿或者下降沿变化时，比较器侦测变化后的信号必须维持 LCDTY 所设置的 CPU 时钟数不发生变化，才认为数据变化是有效的；否则将视同信号无变化。

注意：当使能数字滤波功能后，芯片内部实际的等待时钟需额外增加两个状态机切换时间，即若 LCDTY 设置为 0 时，为关闭数字滤波功能；若 LCDTY 设置为非 0 值 n（n=1~63）时，则实际的数字滤波时间为（n+2）个系统时钟



20.3.6 比较器 3 控制寄存器 3（CMP3_6P6N_CR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP3_6P6N_CR3	7EFB7AH	CMP3CHYS[1:0]		CMP3NS[2:0]			CMP3PS[2:0]		

CMP3CHYS[1:0]：比较器 DC 迟滞输入选择

CHYS [1:0]	比较器 DC 迟滞输入选择
00	0mV
01	10mV
10	20mV
11	30mV

CMP3NS[2:0]：比较器负端输入选择

CMP3NS[2:0]	CMP3-
000	P7.6
001	P5.2
010	OP4O
011	1.19V
100	DAC1O
101	DAC2O

CMP3PS[2:0]：比较器正端输入选择

CMP3PS[2:0]	CMP3+
000	P7.7
001	P5.3
010	OP3O
011	ADC2IN
100	DAC1O
101	DAC2O

20.3.7 比较器 3 控制寄存器 4（CMP3_6P6N_CR4）

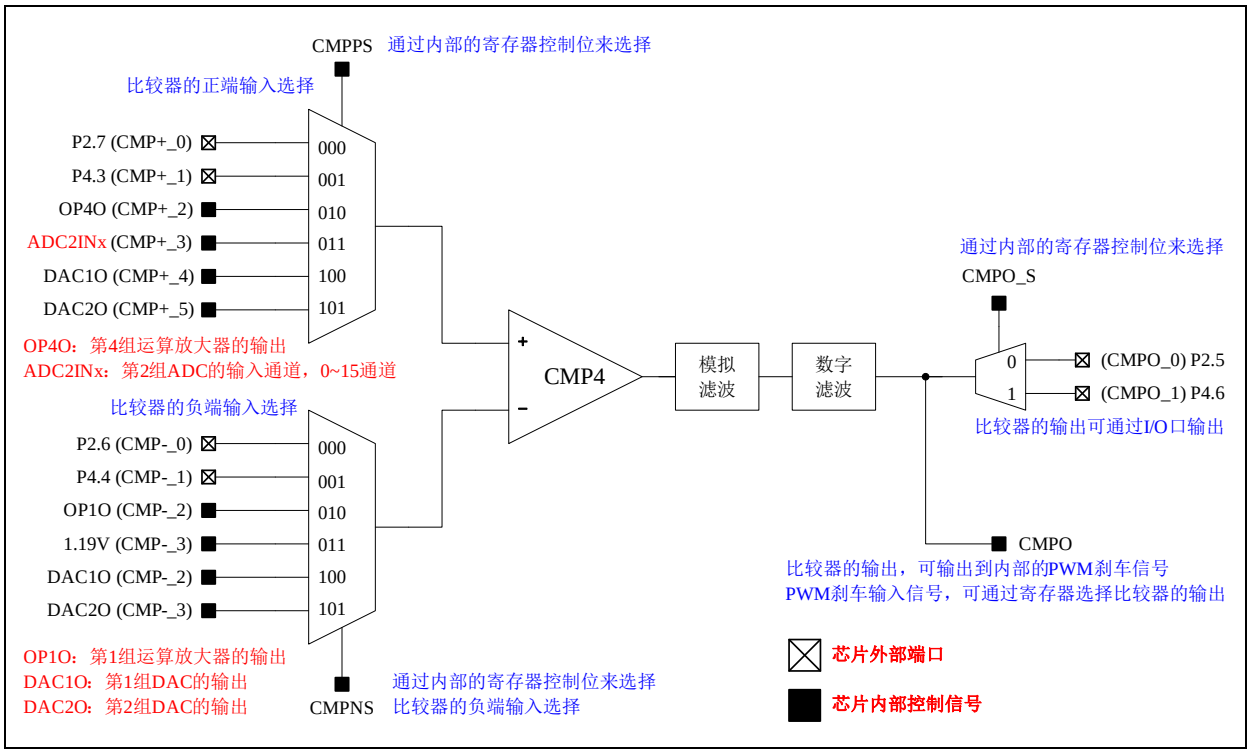
符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP3_6P6N_CR4	7EFB7BH	-	-	-	-	-	-	CMP3O_S[1:0]	

CMP3O_S[1:0]：比较器输出端口选择

CMP3O_S[2:0]	CMP3O
00	P2.4
01	P4.5
10	-
11	-

20.4 比较器 4

20.4.1 比较器 4 内部结构图



20.4.2 比较器 4 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP4_6P6N_CR3	7EFB7EH	CMP4CHYS[1:0]		CMP4NS[2:0]			CMP4PS[2:0]		
CMP4_6P6N_CR4	7EFB7FH	-	-	-	-	-	-	CMP4O_S[1:0]	

CMP4PS[2:0]：比较器正端输入选择

CMP4NS[2:0]：比较器负端输入选择

CMP4O_S[1:0]：比较器输出端口选择

CMP4PS[2:0]	CMP4+	CMP4NS[2:0]	CMP4-	CMP4O_S[2:0]	CMP4O
000	P2.7	000	P2.6	00	P2.5
001	P4.3	001	P4.4	01	P4.6
010	OP4O	010	OP1O	10	-
011	ADC2IN	011	1.19V	11	-
100	DAC1O	100	DAC1O		
101	DAC2O	101	DAC2O		

20.4.3 比较器 4 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CMP4_6P6N_CR1	比较器 4 控制寄存器 1	7EFB7CH	CMP4EN	CMP4IF	CMP4PIE	CMP4NIE	CMP4IP[1:0]		CMP4OE	CMP4RES	0000,0000
CMP4_6P6N_CR2	比较器 4 控制寄存器 2	7EFB7DH	INVCMP4O	CMP4DISFLT	CMP4LCDTY[5:0]						0000,0000
CMP4_6P6N_CR3	比较器 4 控制寄存器 3	7EFB7EH	CMP4CHYS[1:0]		CMP4NS[2:0]			CMP4PS[2:0]			0000,0000
CMP4_6P6N_CR4	比较器 4 控制寄存器 4	7EFB7FH	-	-	-	-	-	-	CMP4O_S[1:0]		xxxx,xx00

20.4.4 比较器 4 控制寄存器 1（CMP4_6P6N_CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP4_6P6N_CR1	7EFB7CH	CMP4EN	CMP4IF	CMP4PIE	CMP4NIE	CMP4IP[1:0]		CMP4OE	CMP4RES

CMP4EN：比较器模块使能位

0：关闭比较功能

1：使能比较功能

CMP4IF：比较器中断标志位。当 PIE 或 NIE 被使能后，若产生相应的中断信号，硬件自动将 CMPIF 置 1，并向 CPU 提出中断请求。此标志位必须用户软件向此位写 0 清零。

CMP4PIE：比较器上升沿中断使能位。

0：禁止比较器上升沿中断。

1：使能比较器上升沿中断。使能比较器的比较结果由 0 变成 1 时产生中断请求。

CMP4NIE：比较器下降沿中断使能位。

0：禁止比较器下降沿中断。

1：使能比较器下降沿中断。使能比较器的比较结果由 1 变成 0 时产生中断请求。

CMP4IP[1:0]：比较器中断优先级设置。

00：比较器中断为最低优先级。

01：比较器中断为较低优先级。

10：比较器中断为较高优先级。

11：比较器中断为最高优先级。

CMP4OE：比较器结果输出控制位

0：禁止比较器结果输出

1：使能比较器结果输出。

CMP4RES：比较器的比较结果。此位为只读。

0：表示 CMP4+的电平低于 CMP4-的电平

1：表示 CMP4+的电平高于 CMP4-的电平

CMPRES 是经过数字滤波后的输出信号，而不是比较器的直接输出结果。

20.4.5 比较器 4 控制寄存器 2（CMP4_6P6N_CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP4_6P6N_CR2	7EFB7DH	INVCMP4O	CMP4DISFLT	CMP4LCDTY[5:0]					

INVCMP4O：比较器结果输出控制

0：比较器结果正向输出。若 CMP4RES 为 0，则输出低电平，反之输出高电平。

1：比较器结果反向输出。若 CMP4RES 为 0，则输出高电平，反之输出低电平。

CMP4DISFLT：模拟滤波功能控制

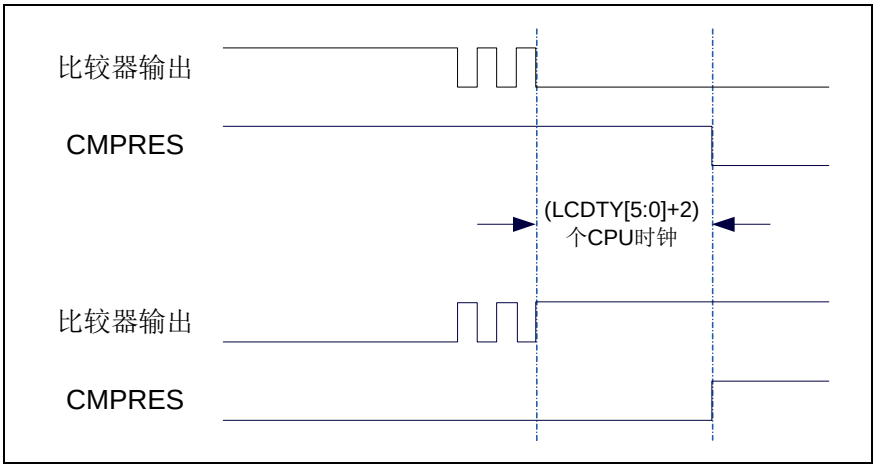
0：使能 0.1us 模拟滤波功能

1：关闭 0.1us 模拟滤波功能，可略微提高比较器的比较速度。

CMP4LCDTY[5:0]：数字滤波功能控制

数字滤波功能即为数字信号去抖动功能。当比较结果发生上升沿或者下降沿变化时，比较器侦测变化后的信号必须维持 LCDTY 所设置的 CPU 时钟数不发生变化，才认为数据变化是有效的；否则将视同信号无变化。

注意：当使能数字滤波功能后，芯片内部实际的等待时钟需额外增加两个状态机切换时间，即若 LCDTY 设置为 0 时，为关闭数字滤波功能；若 LCDTY 设置为非 0 值 n（n=1~63）时，则实际的数字滤波时间为（n+2）个系统时钟



20.4.6 比较器 4 控制寄存器 3（CMP4_6P6N_CR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP4_6P6N_CR3	7EFB7EH	CMP4CHYS[1:0]		CMP4NS[2:0]			CMP4PS[2:0]		

CMP4CHYS[1:0]：比较器 DC 迟滞输入选择

CHYS [1:0]	比较器 DC 迟滞输入选择
00	0mV
01	10mV
10	20mV
11	30mV

CMP4NS[2:0]：比较器负端输入选择

CMP4NS[2:0]	CMP4-
000	P2.6
001	P4.4
010	OP1O
011	1.19V
100	DAC1O
101	DAC2O

CMP4PS[2:0]：比较器正端输入选择

CMP4PS[2:0]	CMP4+
000	P2.7
001	P4.3
010	OP4O
011	ADC2IN
100	DAC1O
101	DAC2O

20.4.7 比较器 4 控制寄存器 4（CMP4_6P6N_CR4）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CMP4_6P6N_CR4	7EFB7FH	-	-	-	-	-	-	CMP4O_S[1:0]	

CMP4O_S[1:0]：比较器输出端口选择

CMP4O_S[2:0]	CMP4O
00	P2.5
01	P4.6
10	-
11	-

21 IAP/EEPROM，1/4/8/16 字节读写

STC32G144K246 系列单片机内部集成了大容量的 EEPROM。利用 ISP/IAP 技术可将内部 Data Flash 当 EEPROM，擦写次数在 10 万次以上。EEPROM 可分为若干个扇区，每个扇区包含 512 字节。

注意：EEPROM 的写操作只能将字节中的 1 写为 0，当需要将字节中的 0 写为 1，则必须执行扇区擦除操作。EEPROM 的读/写操作是以 1 字节为单位进行，而 EEPROM 擦除操作是以 1 扇区（512 字节）为单位进行，在执行擦除操作时，如果目标扇区中有需要保留的数据，则必须预先将这些数据读取到 RAM 中暂存，待擦除完成后再将保存的数据和需要更新的数据一起再写回 EEPROM/DATA-FLASH。

所以在使用 EEPROM 时，建议同一次修改的数据放在同一个扇区，不是同一次修改的数据放在不同的扇区，不一定要用满。数据存储器的擦除操作是按扇区进行的（每扇区 512 字节）。

EEPROM 可用于保存一些需要在应用过程中修改并且掉电不丢失的参数数据。在用户程序中，可以对 EEPROM 进行字节读/字节编程/扇区擦除操作。在工作电压偏低时，建议不要进行 EEPROM 操作，以免发生数据丢失的情况。

STC32G144K246 系列单片机的 EEPROM 读写均支持单字节、4 字节、8 字节和 16 字节模式，单字节读写时，使用 IAP_DATA 数据寄存器，若需要使用 4 字节、8 字节和 16 字节模式进行读写 EEPROM，则需要使用 IAP_XDATA0 ~ IAP_XDATAF 数据寄存器

21.1 EEPROM 操作时间

（测试频率 24MHz，IAP_TPS=24）

	最小值	典型值	最大值
字节读取	-	4 个系统时钟	-
字节编程	-	31us	-
扇区擦除（512 字节）	-	4571us / 4.5ms	-

EEPROM 操作所需时间是硬件自动控制的，用户只需要正确设置 IAP_TPS 寄存器即可。

IAP_TPS＝系统工作频率/1000000（小数部分四舍五入进行取整）

例如：系统工作频率为 12MHz，则 IAP_TPS 设置为 12

系统工作频率为 22.1184MHz，则 IAP_TPS 设置为 22

系统工作频率为 5.5296MHz，则 IAP_TPS 设置为 6

21.2 关于 EEPROM 操作时是否需要关闭中断的问题

- 如果只有主循环的代码中或者只有中断代码中才使用 IAP 方式对 EEPROM 进行操作，则 EEPROM 操作时不需要关闭中断。
- 如果主循环代码和中断代码中都有使用 IAP 方式对 EEPROM 进行操作，则 EEPROM 操作时必须关闭中断。因为如果在主循环代码中设置完成 IAP 相关寄存器后在 IAP 触发前或 IAP 触发一半时，产生了中断，而在中断中进行 EEPROM 操作时，又对 IAP 相关寄存器进行重新设置，中断返回后 IAP 相关寄存器已经发生了改变，回到主循环中继续之前的 IAP 操作必然会出错。

建议程序代码按照如下方式编写：

```
…                                                    ;设置 ISP/IAP 的其他相关寄存器
CLR     EA                                           ;触发 ISP/IAP 前，先关闭总中断
MOV     IAP_TRIG, #5AH          ;先送 5Ah，到 ISP/IAP 触发寄存器
MOV     IAP_TRIG, #0A5H         ;再送 A5h，到 ISP/IAP 触发寄存器，每次都需如此
                                                     ;送完 A5h 后，ISP/IAP 命令立即被触发启动
                                                     ;IAP 命令执行完成前，CPU 不会向下继续执行程序
    SETB    EA                                       ;IAP 完成后再开总中断，防止【IAP 触发命令序列】被任意中断打断
                                                     ;而这个任意中断服务程序中又有【IAP 触发命令序列】
                                                     ;确保操作的【原子性】
```

21.3 EEPROM 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
IAP_DATA	IAP 数据寄存器	C2H	DATA[7:0]								0000,0000
IAP_ADDRE	IAP 扩展地址寄存器	F6H	ADDR[23:16]								1111,1111
IAP_ADDRH	IAP 高地址寄存器	C3H	ADDR[15:8]								0000,0000
IAP_ADDRL	IAP 低地址寄存器	C4H	ADDR[7:0]								0000,0000
IAP_CMD	IAP 命令寄存器	C5H	-	-	-	-	CMD[3:0]				xxxx,0000
IAP_TRIG	IAP 触发寄存器	C6H									0000,0000
IAP_CONTR	IAP 控制寄存器	C7H	IAPEN	SWBS	SWRST	CMD_FAIL	SWBS2	-	-	-	0000,0xxx
IAP_TPS	IAP 等待时间控制寄存器	F5H	IAP_TPS[7:0]								0000,0111

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
IAP_XDATA0	IAP 扩展数据寄存器 0	7EF5F0H	IAP_XDATA0[7:0]								0000,0000
IAP_XDATA1	IAP 扩展数据寄存器 1	7EF5F1H	IAP_XDATA1[7:0]								0000,0000
IAP_XDATA2	IAP 扩展数据寄存器 2	7EF5F2H	IAP_XDATA2[7:0]								0000,0000
IAP_XDATA3	IAP 扩展数据寄存器 3	7EF5F3H	IAP_XDATA3[7:0]								0000,0000
IAP_XDATA4	IAP 扩展数据寄存器 4	7EF5F4H	IAP_XDATA4[7:0]								0000,0000
IAP_XDATA5	IAP 扩展数据寄存器 5	7EF5F5H	IAP_XDATA5[7:0]								0000,0000
IAP_XDATA6	IAP 扩展数据寄存器 6	7EF5F6H	IAP_XDATA6[7:0]								0000,0000
IAP_XDATA7	IAP 扩展数据寄存器 7	7EF5F7H	IAP_XDATA7[7:0]								0000,0000
IAP_XDATA8	IAP 扩展数据寄存器 8	7EF5F8H	IAP_XDATA8[7:0]								0000,0000
IAP_XDATA9	IAP 扩展数据寄存器 9	7EF5F9H	IAP_XDATA9[7:0]								0000,0000
IAP_XDATAA	IAP 扩展数据寄存器 A	7EF5FAH	IAP_XDATAA[7:0]								0000,0000
IAP_XDATAB	IAP 扩展数据寄存器 B	7EF5FBH	IAP_XDATAB[7:0]								0000,0000
IAP_XDATAC	IAP 扩展数据寄存器 C	7EF5FCH	IAP_XDATAC[7:0]								0000,0000
IAP_XDATAD	IAP 扩展数据寄存器 D	7EF5FDH	IAP_XDATAD[7:0]								0000,0000
IAP_XDATAE	IAP 扩展数据寄存器 E	7EF5FEH	IAP_XDATAE[7:0]								0000,0000
IAP_XDATAF	IAP 扩展数据寄存器 F	7EF5FFH	IAP_XDATAF[7:0]								0000,0000

21.3.1 EEPROM 数据寄存器（IAP_DATA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IAP_DATA	C2H	DATA[7:0]							

在进行 EEPROM 的读操作时，命令执行完成后读出的 EEPROM 数据保存在 IAP_DATA 寄存器中。在进行 EEPROM 的写操作时，在执行写命令前，必须将待写入的数据存放在 IAP_DATA 寄存器中，再发送写命令。擦除 EEPROM 命令与 IAP_DATA 寄存器无关。

21.3.2 EEPROM 扩展数据寄存器（IAP_XDATA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IAP_XDATA0	7EF5F0H	IAP_XDATA0[7:0]							
IAP_XDATA1	7EF5F1H	IAP_XDATA1[7:0]							
IAP_XDATA2	7EF5F2H	IAP_XDATA2[7:0]							
IAP_XDATA3	7EF5F3H	IAP_XDATA3[7:0]							
IAP_XDATA4	7EF5F4H	IAP_XDATA4[7:0]							
IAP_XDATA5	7EF5F5H	IAP_XDATA5[7:0]							
IAP_XDATA6	7EF5F6H	IAP_XDATA6[7:0]							
IAP_XDATA7	7EF5F7H	IAP_XDATA7[7:0]							
IAP_XDATA8	7EF5F8H	IAP_XDATA8[7:0]							
IAP_XDATA9	7EF5F9H	IAP_XDATA9[7:0]							
IAP_XDATAA	7EF5FAH	IAP_XDATAA[7:0]							
IAP_XDATAB	7EF5FBH	IAP_XDATAB[7:0]							
IAP_XDATAC	7EF5FCH	IAP_XDATAC[7:0]							
IAP_XDATAD	7EF5FDH	IAP_XDATAD[7:0]							
IAP_XDATAE	7EF5FEH	IAP_XDATAE[7:0]							
IAP_XDATAF	7EF5FFH	IAP_XDATAF[7:0]							

若需要使用 4 字节、8 字节和 16 字节模式进行读写 EEPROM，则需要使用 IAP_XDATA 数据寄存器。

21.3.3 EEPROM 地址寄存器（IAP_ADDR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IAP_ADDRE	F6H	ADDR[23:16]							
IAP_ADDRH	C3H	ADDR[15:8]							
IAP_ADDRL	C4H	ADDR[7:0]							

EEPROM 进行读、写、擦除操作的目标地址寄存器。IAP_ADDRH 保存地址的高字节，IAP_ADDRL 保存地址的低字节

【注意】：使能用户系统区功能后，在用户系统区，IAP 操作 EEPROM 要使用绝对地址。

21.3.4 EEPROM 命令寄存器（IAP_CMD）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IAP_CMD	C5H	-	-	-	-	CMD[3:0]			

CMD[3:0]：发送EEPROM操作命令

CMD[3:0]	EEPROM 操作	说明
0	无操作	
1	单字节读取	
2	单字节编程	
3	扇区擦除	512 字节
4	-	
5	4 字节编程	地址需 4 字节对齐
6	8 字节编程	地址需 8 字节对齐
7	16 字节编程	地址需 16 字节对齐
8	-	
9	-	
A	-	
B	-	
C	-	
D	4 字节读取	地址需 4 字节对齐
E	8 字节读取	地址需 8 字节对齐
F	16 字节读取	地址需 16 字节对齐

21.3.5 EEPROM 触发寄存器（IAP_TRIG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IAP_TRIG	C6H								

设置完成 EEPROM 读、写、擦除的命令寄存器、地址寄存器、数据寄存器以及控制寄存器后，需要向触发寄存器 IAP_TRIG 依次写入 5AH、A5H（顺序不能交换）两个触发命令来触发相应的读、写、擦除操作。操作完成后，EEPROM 地址寄存器 IAP_ADDRE、IAP_ADDRH、IAP_ADDRL 和 EEPROM 命令寄存器 IAP_CMD 的内容不变。如果接下来要对下一个地址的数据进行操作，需手动更新地址寄存器 IAP_ADDRH 和寄存器 IAP_ADDRL 的值。

注意：每次 EEPROM 操作时，都要对 IAP_TRIG 先写入 5AH，再写入 A5H，相应的命令才会生效。写完触发命令后，CPU 会处于 IDLE 等待状态，直到相应的 IAP 操作执行完成后 CPU 才会从 IDLE 状态返回正常状态继续执行 CPU 指令。

如果主循环代码和中断代码中都有使用 IAP 方式对 EEPROM 进行操作，则 EEPROM 操作时必须关闭中断。因为如果在主循环代码中设置完成 IAP 相关寄存器后在 IAP 触发前或 IAP 触发一半时，产生了中断，而在中断中进行 EEPROM 操作时，又对 IAP 相关寄存器进行重新设置，中断返回后 IAP 相关寄存器已经发生了改变，回到主循环中继续之前的 IAP 操作必然会出错。

建议程序代码按照如下方式编写：

```
…                                                   ;设置 ISP/IAP 的其他相关寄存器
CLR     EA                                          ;触发 ISP/IAP 前，先关闭总中断
MOV     IAP_TRIG, #5AH          ;先送 5Ah，到 ISP/IAP 触发寄存器
MOV     IAP_TRIG, #0A5H         ;再送 A5h，到 ISP/IAP 触发寄存器，每次都需如此
                                                    ;送完 A5h 后，ISP/IAP 命令立即被触发启动
                                                    ;IAP 命令执行完成前，CPU 不会向下继续执行程序
SETB    EA                                  ;IAP 完成后再开总中断，防止【IAP 触发命令序列】被任意中断打断
                                                    ;而这个任意中断服务程序中又有【IAP 触发命令序列】
                                                    ;确保操作的【原子性】
```

21.3.6 EEPROM 控制寄存器（IAP_CONTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IAP_CONTR	C7H	IAPEN	SWBS	SWRST	CMD_FAIL	SWBS2	-	-	-

IAPEN：EEPROM操作使能控制位

0：禁止 EEPROM 操作

1：使能 EEPROM 操作

CMD_FAIL：EEPROM操作失败状态位，需要软件清零

0：EEPROM 操作正确

1：EEPROM 操作失败

SWBS/SWBS2：软件复位启动选择

SWBS	SWBS2	复位	说明
0	0	软件复位后从用户程序区开始执行代码	用户数据区的数据保持不变
0	1	软件复位后从用户系统区开始执行代码	用户数据区的数据保持不变
1	x	软件复位后从系统 ISP 区开始执行代码	用户数据区的数据会被初始化

SWRST：软件复位触发位

0：对单片机无影响

1：触发软件复位

范例程序：

源程序区 （从此程序区）	目标程序区 （复位到此程序区）	代码
用户程序区	系统 ISP 区	IAP_CONTR = 0x60;
用户系统区		
用户程序区	用户系统区	IAP_CONTR = 0x28;
系统 ISP 区		
用户系统区	用户程序区	IAP_CONTR = 0x20;
系统 ISP 区		

21.3.7 EEPROM 擦除等待时间控制寄存器（IAP_TPS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
IAP_TPS	F5H	IAP_TPS[7:0]							

需要根据工作频率进行设置

若工作频率为12MHz，则需要将IAP_TPS设置为12；若工作频率为24MHz，则需要将IAP_TPS设置为24，其他频率以此类推。

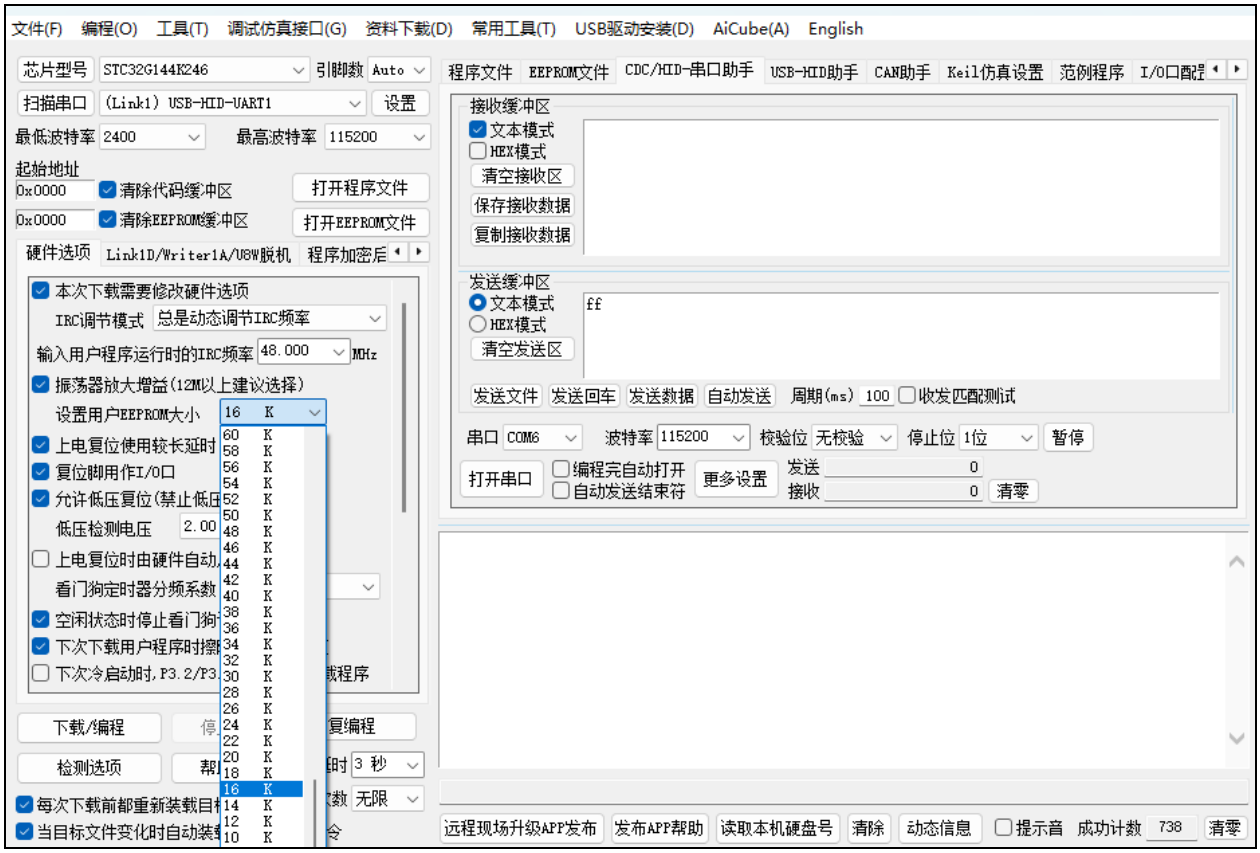
21.4 EEPROM 大小及地址

STC32G144K246 系列单片机内部均有用于保存用户数据的 EEPROM。内部的 EEPROM 有 3 操作方式：读、写和擦除，其中擦除操作是以扇区为单位进行操作，每扇区为 512 字节，即每执行一次擦除命令就会擦除一个扇区，而读数据和写数据可以以 1 字节/4 字节/8 字节/16 字节为单位进行操作，即每执行一次读或者写命令时能读出或者写入 1 字节/4 字节/8 字节/16 字节。

STC32G144K246 系列单片机内部的 EEPROM 的访问方式有两种：IAP 方式和 MOV 方式。IAP 方式可对 EEPROM 执行读、写、擦除操作，但 MOV 只能对 EEPROM 进行读操作，而不能进行写和擦除操作。无论是使用 IAP 方式还是使用 MOV 方式访问 EEPROM，首先都需要设置正确的目标地址。

21.4.1 STC32G144K246 系列 EEPROM 操作

STC32G144K246 的 EEPROM 大小是需要在 ISP 下载时进行设置的，如下图所示，将 EEPROM 设置为 16K



EEPROM 在 246K 的 Flash 存储空间中位于 FC:2800h~FF:FFFFh（注意：无论 EEPROM 设置为多少，STC32G144K246 系列的 EEPROM 在 Flash 存储空间中起始地址始终为 FC:2800h，即 **EEPROM 总是从低地址向高地址进行规划的**）。对于 STC32G144K246 系列，使用 IAP 方式时，地址数据为 EEPROM 的目标地址，地址从 0000 开始，若使用 MOV 指令读取 EEPROM 数据，则地址数据为：**基地址 FC:2800h+EEPROM 的目标地址**。

例如：现在需要对 EEPROM 地址 1234h 的单元进行读、写、擦除时，若使用 IAP 方式进行访问时，设置的目标地址为 1234h，即 IAP_ADDRE 设置为 00h，IAP_ADDRH 设置 12h，IAP_ADDRL 设置 34h，然后设置相应的触发命令即可对 1234h 单元进行正确操作了。但若是使用 MOV 方式读取 EEPROM 的 1234h 单元，则目标地址为**基地址 FC:2800h，**再加上 1234h，即必须将 32 位寄存器 DRx 设置为 FC:3A34h，才能使用 MOV 指令正确读取。

22 两组 12 位模数转换（ADC1/ADC2）

产品线	ADC 分辨率	ADC 通道数
STC32G144K246 系列	**两组*12 位**	**15+15 通道**

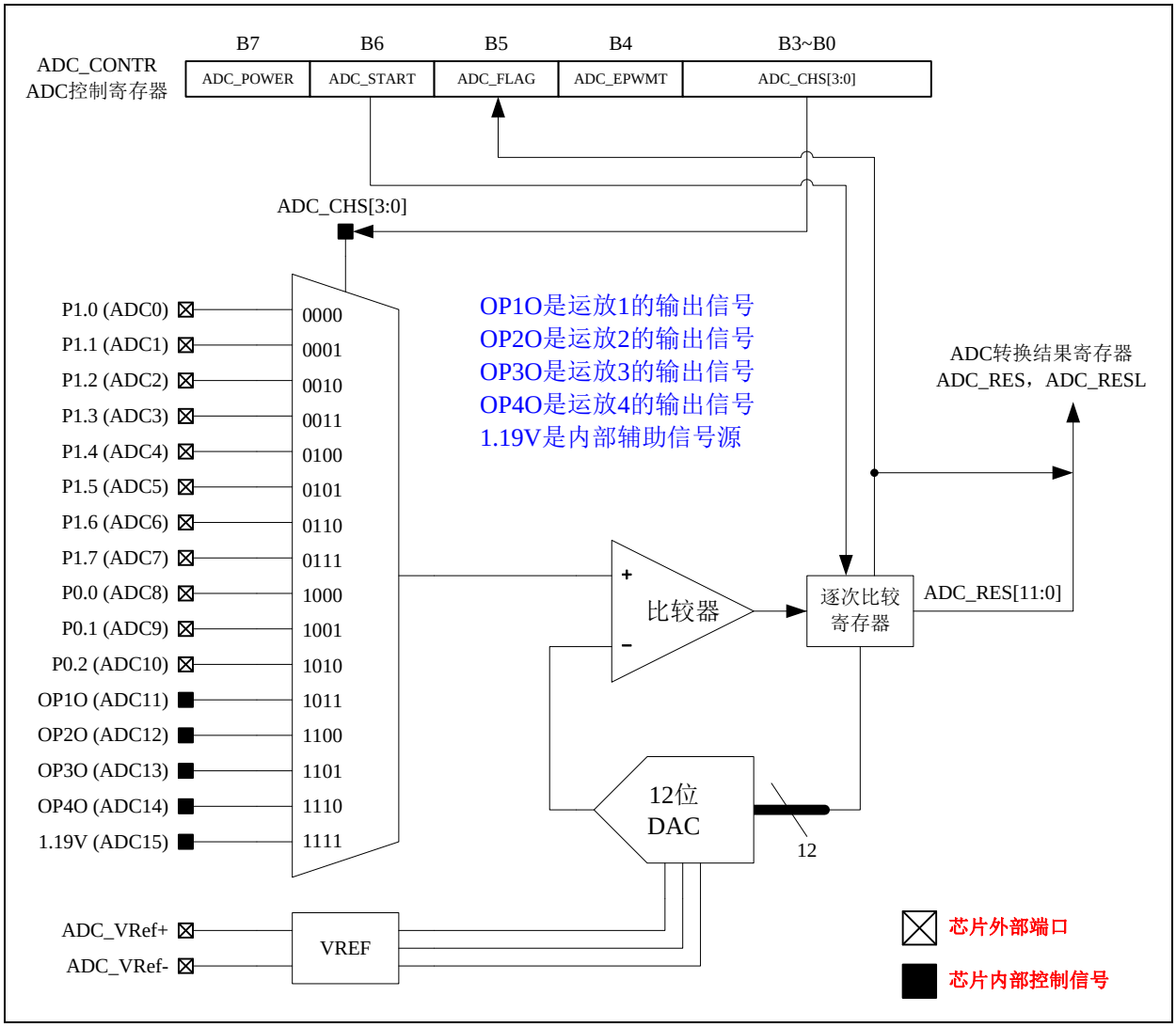
STC32G144K246 系列单片机内部集成了两组 12 位高速 A/D 转换器。ADC 的时钟频率为系统频率 2 分频再经过用户设置的分频系数进行再次分频（ADC 的时钟频率范围为 SYSclk/2/1~SYSclk/2/16）。

ADC 转换结果的数据格式有两种：左对齐和右对齐。可方便用户程序进行读取和引用。

两组 ADC 除了可以使用原始的独立方式外，还可以使用同步方式，使能内部控制信号同时触发两组 ADC，以实现差分 ADC 的效果。

22.1 ADC1

22.1.1 ADC1 内部结构图（ADC 框图）



22.1.2 ADC1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
ADC_CONTR	ADC 控制寄存器	BCH	ADC_POWER	ADC_START	ADC_FLAG	ADC_EPWMT	ADC_CHS[3:0]				0000,0000
ADC_RES	ADC 转换结果高位寄存器	BDH									0000,0000
ADC_RESL	ADC 转换结果低位寄存器	BEH									0000,0000
ADCCFG	ADC 配置寄存器	DEH	-	-	RESFMT	RESDIFF	SPEED[3:0]				xx00,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
ADCTIM	ADC 时序控制寄存器	7EFEA8H	CSSETUP	CSHOLD[1:0]		SMPDUTY[4:0]					0010,1010
ADCEXCFG	ADC 扩展配置寄存器	7EFEADH	ADCETR_PS[1:0]		ADCETRS[1:0]		-	CVTIMESEL[2:0]			0000,x000
ADCEXCFG2	ADC 扩展配置寄存器 2	7EFEAFH	TRIGON	-	PWMSEL[1:0]		ADC_TRIGSEL[3:0]				0x01,0000

22.1.3 ADC1 控制寄存器（ADC_CONTR），PWM 触发 ADC 控制

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADC_CONTR	BCH	ADC_POWER	ADC_START	ADC_FLAG	ADC_EPWMT	ADC_CHS[3:0]			

ADC_POWER：ADC 电源控制位

0：关闭 ADC 电源

1：打开 ADC 电源。

建议进入空闲模式和主时钟停振/省电模式前将 ADC 电源关闭，以降低功耗

特别注意：

1、给 MCU 的内部 ADC 模块电源打开后，需等待约 1ms，等 MCU 内部的 ADC 电源稳定后再让 ADC 工作；

2、适当加长对外部信号的采样时间，就是对 ADC 内部采样保持电容的充电或放电时间，时间够，内部才能和外部电势相等。

ADC_START：ADC 转换启动控制位。写入 1 后开始 ADC 转换，转换完成后硬件自动将此位清零。

0：无影响。即使 ADC 已经开始转换工作，写 0 也不会停止 A/D 转换。

1：开始 ADC 转换，转换完成后硬件自动将此位清零。

ADC_FLAG：ADC 转换结束标志位。当 ADC 完成一次转换后，硬件会自动将此位置 1，并向 CPU 提出中断请求。此标志位必须软件清零。

ADC_EPWMT：使能 PWM 实时触发 ADC 功能。

ADC_CHS[3:0]：ADC 模拟通道选择位

（注意：被选择为 ADC 输入通道的 I/O 口，必须设置 PxM0/PxM1 寄存器将 I/O 口模式设置为高阻输入模式。另外如果 MCU 进入主时钟停振/省电模式后，仍需要使能 ADC 通道，则需要设置 PxIE 寄存器关闭数字输入通道，以防止外部模拟输入信号忽高忽低而产生额外的功耗）

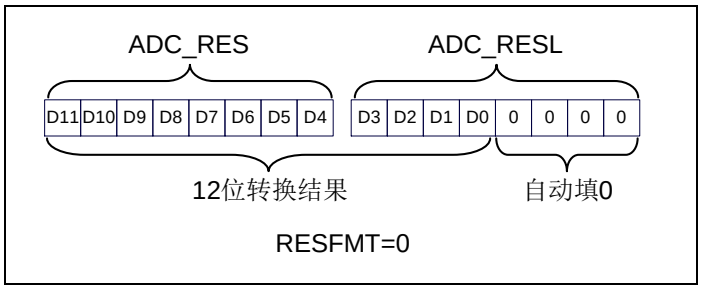
ADC_CHS[3:0]	ADC 通道	ADC_CHS[3:0]	ADC 通道
0000	P1.0	1000	P0.0
0001	P1.1	1001	P0.1
0010	P1.2	1010	P0.2
0011	P1.3	1011	OP1O
0100	P1.4	1100	OP2O
0101	P1.5	1101	OP3O
0110	P1.6	1110	OP4O
0111	P1.7	1111	测试内部 1.19V

22.1.4 ADC1 配置寄存器（ADCCFG）

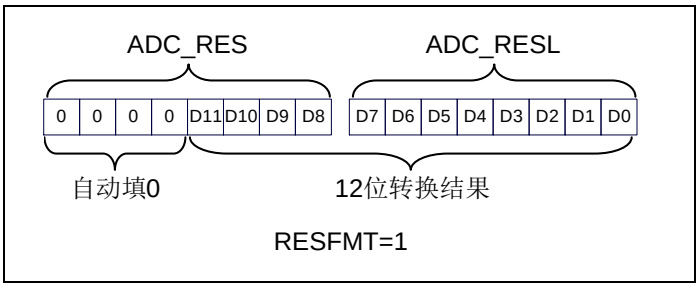
符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADCCFG	DEH	-	-	RESFMT	RESDIFF	SPEED[3:0]			

RESFMT：ADC 转换结果格式控制位

0：转换结果左对齐。ADC_RES 保存结果的高 8 位，ADC_RESL 保存结果的低 4 位。格式如下：



1：转换结果右对齐。ADC_RES 保存结果的高 4 位，ADC_RESL 保存结果的低 8 位。格式如下：



RESDIFF：ADC 转换结果为差分格式

0：非差分格式。ADC_RES[11:0]为原始的 ADC 转换结果

1：差分格式。当 ADC1 和 ADC2 为同步触发模式时，可将 RESDIFF 设置为 1，可以模拟出差分通道的效果，ADC_RES[11:0]为（ADC1 转换结果 - ADC2 的转换结果）

SPEED[3:0]：设置 ADC 时钟｛FADC＝SYSclk/2/(SPEED+1)｝

SPEED[3:0]	ADC 时钟频率
0000	SYSclk/2/1
0001	SYSclk/2/2
0010	SYSclk/2/3
...	. . .
1101	SYSclk/2/14
1110	SYSclk/2/15
1111	SYSclk/2/16

22.1.5 ADC1 转换结果寄存器（ADC_RES，ADC_RESL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADC_RES	BDH								
ADC_RESL	BEH								

当 A/D 转换完成后，转换结果会自动保存到 ADC_RES 和 ADC_RESL 中。保存结果的数据格式请参考 ADC_CFG 寄存器中的 RESFMT 设置。

22.1.6 ADC1 时序控制寄存器（ADCTIM）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADCTIM	7EFEA8H	CSSETUP	CSHOLD[1:0]		SMPDUTY[4:0]				

CSSETUP：ADC 通道选择时间控制 Tsetup

CSSETUP	ADC 时钟数
0	**1（默认值）**
1	2

CSHOLD[1:0]：ADC 通道选择保持时间控制 Thold

CSHOLD[1:0]	ADC 时钟数
00	1
01	**2（默认值）**
10	3
11	4

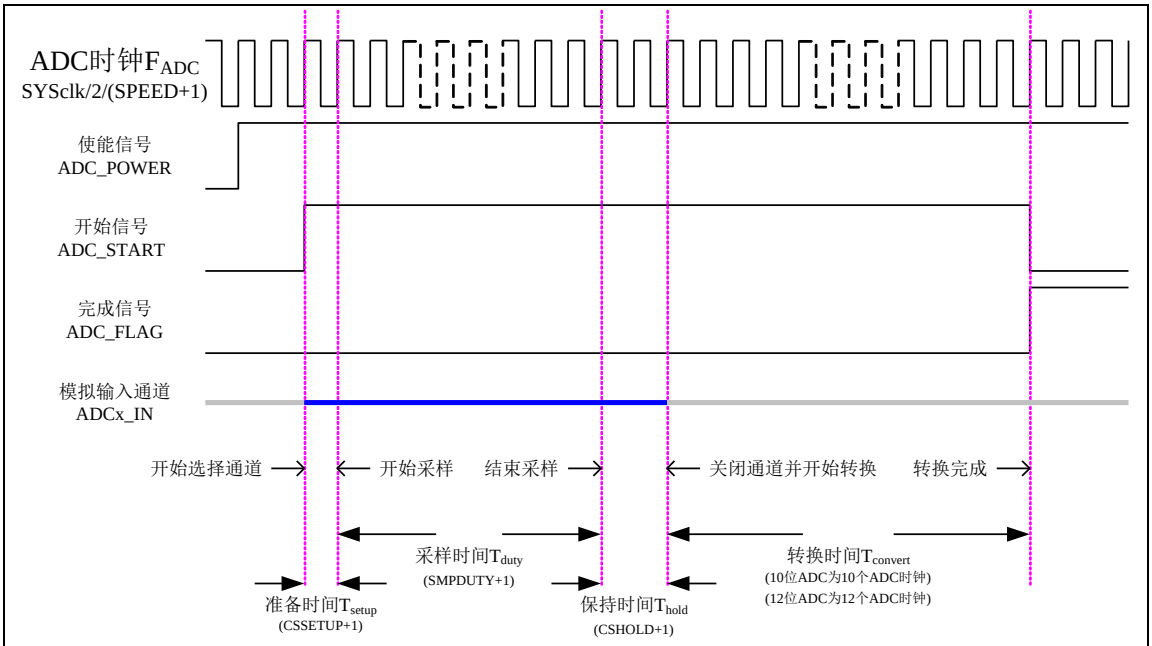
SMPDUTY[4:0]：ADC 模拟信号采样时间控制 Tduty（**注意：SMPDUTY 一定不能设置小于 01010B**）

SMPDUTY[4:0]	ADC 时钟数
00000	1
00001	2
...	...
01010	**11（默认值）**
...	...
11110	31
11111	32

ADC 数模转换时间：$T_{convert}$

12 位 ADC 的转换时间固定为 12 个 ADC 工作时钟

一个完整的 ADC 转换时间为： $T_{setup} + T_{duty} + T_{hold} + T_{convert}$，如下图所示



ADC整体转换时序图

22.1.7 ADC1 扩展配置寄存器（ADCEXCFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADCEXCFG	7EFEADH	ADCETR_PS[1:0]		ADCETRS[1:0]		-	CVTIMESEL[2:0]		

ADCETR_PS[1:0]：ADC 外部触发脚 ADC_ETR 功能脚选择

ADCETR_PS[1:0]	ADC_ETR
00	P1.2
01	P4.3
10	-
11	-

ADCETRS[1:0]：ADC 外部触发脚 ADC_ETR 控制位

ADCETRS[1:0]	ADC_ETR 设置
0x	禁止 ETR 功能
10	使能 ADC_ETR 的上升沿触发 ADC
11	使能 ADC_ETR 的下降沿触发 ADC

注：使用此功能前，必须打开 ADC_CONTR 中的 ADC 电源开关，并设置好相应的 ADC 通道

CVTIMESEL[2:0]：ADC 自动转换次数选择

CVTIMESEL [2:0]	ADC 自动转换次数
0xx	转换 1 次
100	转换 2 次并取平均值
101	转换 4 次并取平均值
110	转换 8 次并取平均值
111	转换 16 次并取平均值

注：

1、当使能 ADC 自动转换多次功能后，ADC 中断标志只会在 ADC 自动转换到设置的次数后，才会被置 1（例如：设置 CVTIMESEL 为 101B，即 ADC 自动转换 4 次并取平均值，则 ADC 中断标志位每完成 4 次 ADC 转换才会被置 1）

2、当 ADC 处于 DMA 模式下时，ADCEXCFG 设置的多次转换次数无效。
ADCEXCFG 设置的重复转换次数只有在非 DMA 模式下才有效。

22.1.8 ADC1 扩展配置寄存器 2（ADCEXCFG2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADCEXCFG2	7EFEAFH	TRIGON	-	PWMSEL[1:0]		ADC_TRIGSEL[3:0]			

TRIGON：两组 ADC 同步触发控制

0：两组 ADC 为各自独立工作模式

1：两组 ADC 为同步模式

ADC_TRIGSEL[3:0]：选择 ADC 触发源（必须使能 ADC_EPWMT，触发源才有效）

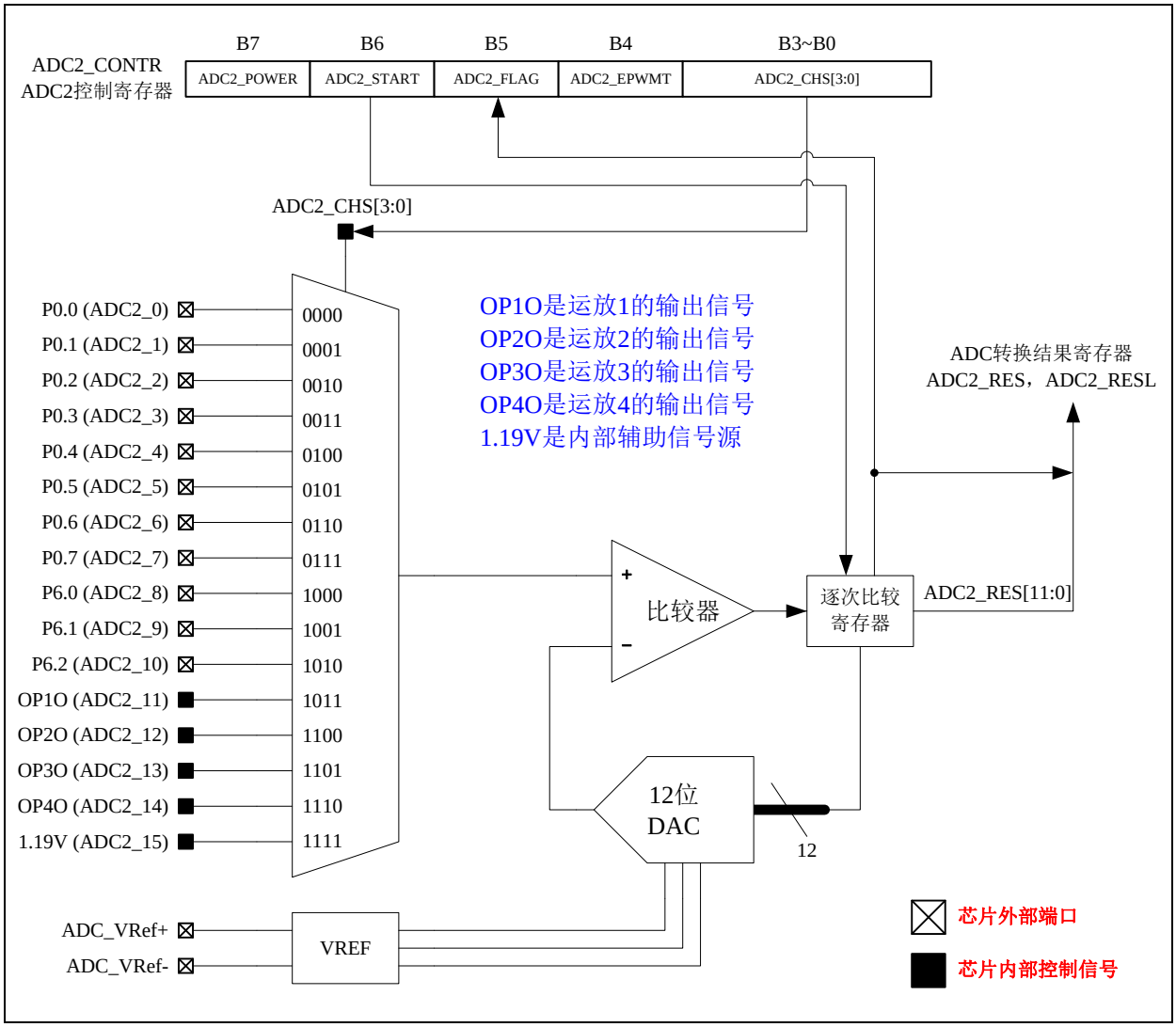
ADC_TRIGSEL[3:0]	ADC 触发源
0000	PWMA_TA
0001	PWMC_TA
0010	PWME_TA
0011	PWMA_TA2
0100	PWMC_TA2
0101	PWME_TA2
0110	PWMB_TA
0111	PWMD_TA
1000	PWMF_TA
1001	1
1010	0
1011	PWMA_TA
1100	TIMER17 OV
1101	TIMER18 OV
1110	TIMER17 OV phase 180
1111	TIMER18 OV phase 180

PWMSEL[1:0]：PWM 触发信号配置

PWM 触发信号	PWMSEL[1:0]			
	00	01	10	11
PWMA_TA	0	PWMA TRGO 上升沿	PWMA TRGO 下降沿	PWMA TRGO 上升沿+下降沿
PWMC_TA	0	PWMC TRGO 上升沿	PWMC TRGO 下降沿	PWMC TRGO 上升沿+下降沿
PWME_TA	0	PWME TRGO 上升沿	PWME TRGO 下降沿	PWME TRGO 上升沿+下降沿
PWMA_TA2	0	PWMA TRGO2 上升沿	PWMA TRGO2 下降沿	PWMA TRGO2 上升沿+下降沿
PWMC_TA2	0	PWMC TRGO2 上升沿	PWMC TRGO2 下降沿	PWMC TRGO2 上升沿+下降沿
PWME_TA2	0	PWME TRGO2 上升沿	PWME TRGO2 下降沿	PWME TRGO2 上升沿+下降沿
PWMB_TA	0	PWMB TRGO 上升沿	PWMB TRGO 下降沿	PWMB TRGO 上升沿+下降沿
PWMD_TA	0	PWMD TRGO 上升沿	PWMD TRGO 下降沿	PWMD TRGO 上升沿+下降沿
PWMF_TA	0	PWMF TRGO 上升沿	PWMF TRGO 下降沿	PWMF TRGO 上升沿+下降沿

22.2 ADC2

22.2.1 ADC2 内部结构图（ADC2 框图）



22.2.2 ADC2 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
ADC2_CONTR	ADC2 控制寄存器	7EFB38H	ADC2_POWER	ADC2_START	ADC2_FLAG	ADC2_EPWMT	ADC2_CHS[3:0]				0000,0000
ADC2CFG	ADC2 配置寄存器	7EFB39H	-	-	ADC2_RESFMT	-	ADC2_SPEED[3:0]				xx0x,0000
ADC2_RES	ADC2 结果高位寄存器	7EFB3AH	ADC2_RES[15:8]								0000,0000
ADC2_RESL	ADC2 结果低位寄存器	7EFB3BH	ADC2_RESL[7:0]ADC2								0000,0000
ADC2TIM	ADC2 时序控制寄存器	7EFB3CH	ADC2_CSSETUP	ADC2_CSHOLD[1:0]		ADC2_SMPDUTY[4:0]					0010,1010
ADC2EXCFG	ADC2 扩展配置寄存器	7EFB3DH	ADC2ETR_PS[1:0]		ADC2ETRS[1:0]		-	ADC2_CVTIMESEL[2:0]			0000,x000
ADC2INTR	ADC2 中断控制寄存器	7EFB3EH	-	-	-	-	-	PADC2[1:0]		EADC2	xxxx,x000
ADC2EXCFG2	ADC2 扩展配置寄存器 2	7EFB3FH	-	-	PWMSEL[1:0]		ADC_TRIGSEL[3:0]				xx00,0000

22.2.3 ADC2 控制寄存器（ADC2_CONTR），PWM 触发 ADC2 控制

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADC2_CONTR	7EFB38H	ADC2_POWER	ADC2_START	ADC2_FLAG	ADC2_EPWMT	ADC2_CHS[3:0]			

ADC2_POWER：ADC2 电源控制位

0：关闭 ADC2 电源

1：打开 ADC2 电源。

建议进入空闲模式和主时钟停振/省电模式前将 ADC2 电源关闭，以降低功耗

特别注意：

1、给 MCU 的内部 ADC2 模块电源打开后，需等待约 1ms，等 MCU 内部的 ADC2 电源稳定后再让 ADC2 工作；

2、适当加长对外部信号的采样时间，就是对 ADC2 内部采样保持电容的充电或放电时间，时间够，内部才能和外部电势相等。

ADC2_START：ADC2 转换启动控制位。写入 1 后开始 ADC2 转换，转换完成后硬件自动将此位清零。

0：无影响。即使 ADC2 已经开始转换工作，写 0 也不会停止 A/D 转换。

1：开始 ADC2 转换，转换完成后硬件自动将此位清零。

ADC2_FLAG：ADC2 转换结束标志位。当 ADC2 完成一次转换后，硬件会自动将此位置 1，并向 CPU 提出中断请求。此标志位必须软件清零。

ADC2_EPWMT：使能 PWM 实时触发 ADC2 功能

ADC2_CHS[3:0]：ADC2 模拟通道选择位

（注意：被选择为 ADC2 输入通道的 I/O 口，必须设置 PxM0/PxM1 寄存器将 I/O 口模式设置为高阻输入模式。另外如果 MCU 进入主时钟停振/省电模式后，仍需要使能 ADC2 通道，则需要设置 PxIE 寄存器关闭数字输入通道，以防止外部模拟输入信号忽高忽低而产生额外的功耗）

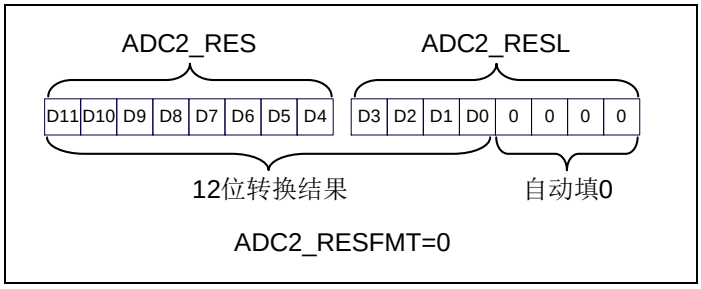
ADC2_CHS[3:0]	ADC2 通道	ADC2_CHS[3:0]	ADC2 通道
0000	P0.0	1000	P6.0
0001	P0.1	1001	P6.1
0010	P0.2	1010	P6.2
0011	P0.3	1011	OP1O
0100	P0.4	1100	OP2O
0101	P0.5	1101	OP3O
0110	P0.6	1110	OP4O
0111	P0.7	1111	测试内部 1.19V

22.2.4 ADC2 配置寄存器（ADC2CFG）

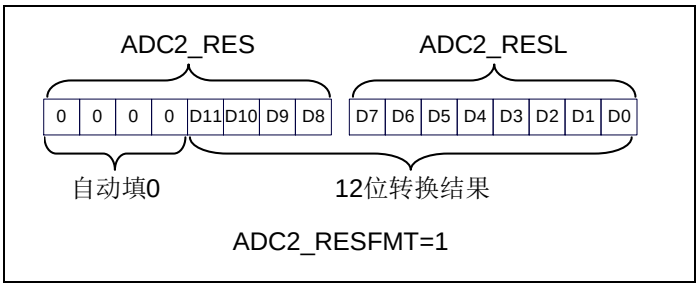
符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADC2CFG	7EFB39H	-	-	ADC2_RESFMT	-	ADC2_SPEED[3:0]			

ADC2_RESFMT：ADC2 转换结果格式控制位

0：转换结果左对齐。ADC2_RES 保存结果的高 8 位，ADC2_RESL 保存结果的低 4 位。格式如下：



1：转换结果右对齐。ADC2_RES 保存结果的高 4 位，ADC2_RESL 保存结果的低 8 位。格式如下：



ADC2_SPEED[3:0]：设置 ADC2 时钟｛FADC2＝SYSclk/2/(ADC2_SPEED+1)｝

ADC2_SPEED[3:0]	ADC2 时钟频率
0000	SYSclk/2/1
0001	SYSclk/2/2
0010	SYSclk/2/3
...	. . .
1101	SYSclk/2/14
1110	SYSclk/2/15
1111	SYSclk/2/16

22.2.5 ADC2 转换结果寄存器（ADC2_RES，ADC2_RESL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADC2_RES	7EFB3AH								
ADC2_RESL	7EFB3BH								

当 A/D 转换完成后，转换结果会自动保存到 ADC2_RES 和 ADC2_RESL 中。保存结果的数据格式请参考 ADC2_CFG 寄存器中的 ADC2_RESFMT 设置。

22.2.6 ADC2 时序控制寄存器（ADC2TIM）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADC2TIM	7EFB3CH	ADC2_CSSETUP	ADC2_CSHOLD[1:0]		ADC2_SMPDUTY[4:0]				

ADC2_CSSETUP：ADC2 通道选择时间控制 Tsetup

ADC2_CSSETUP	ADC2 时钟数
0	**1（默认值）**
1	2

ADC2_CSHOLD[1:0]：ADC2 通道选择保持时间控制 Thold

ADC2_CSHOLD[1:0]	ADC2 时钟数
00	1
01	**2（默认值）**
10	3
11	4

ADC2_SMPDUTY[4:0]：ADC2 模拟信号采样时间控制 Tduty（**注意：SMPDUTY 一定不能设置小于 01010B**）

ADC2_SMPDUTY[4:0]	ADC2 时钟数
00000	1
00001	2
...	...
01010	**11（默认值）**
...	...
11110	31
11111	32

22.2.7 ADC2 扩展配置寄存器（ADC2EXCFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADC2EXCFG	7EFB3DH	ADC2ETR_PS[1:0]		ADC2ETRS[1:0]		-	ADC2_CVTIMESEL[2:0]		

ADC2ETR_PS[1:0]：ADC2 外部触发脚 ADC2_ETR 功能脚选择

ADC2ETR_PS[1:0]	ADC2_ETR
00	P1.2
01	P4.3
10	-
11	-

ADC2ETRS[1:0]：ADC2 外部触发脚 ADC2_ETR 控制位

ADC2ETRS[1:0]	ADC2_ETR 设置
0x	禁止 ETR 功能
10	使能 ADC2_ETR 的上升沿触发 ADC2
11	使能 ADC2_ETR 的下降沿触发 ADC2

注：使用此功能前，必须打开 ADC2_CONTR 中的 ADC2 电源开关，并设置好相应的 ADC2 通道

ADC2_CVTIMESEL[2:0]：ADC2 自动转换次数选择

ADC2_CVTIMESEL [2:0]	ADC2 自动转换次数
0xx	转换 1 次
100	转换 2 次并取平均值
101	转换 4 次并取平均值
110	转换 8 次并取平均值
111	转换 16 次并取平均值

注：

1、当使能 ADC2 自动转换多次功能后，ADC2 中断标志只会在 ADC2 自动转换到设置的次数后，才会被置 1（例如：设置 CVTIMESEL 为 101B，即 ADC2 自动转换 4 次并取平均值，则 ADC2 中断标志位每完成 4 次 ADC2 转换才会被置 1）

2、当 ADC2 处于 DMA 模式下时，ADC2EXCFG 设置的多次转换次数无效。
ADC2EXCFG 设置的重复转换次数只有在非 DMA 模式下才有效。

22.2.8 ADC2 扩展配置寄存器 2（ADC2EXCFG2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADC2EXCFG2	7EFB3FH	TRIGON	-	PWMSEL[1:0]		ADC2_TRIGSEL[3:0]			

TRIGON：两组 ADC 同步触发控制

0：两组 ADC 为各自独立工作模式

1：两组 ADC 为同步模式

ADC2_TRIGSEL[3:0]：选择 ADC2 触发源（必须使能 ADC2_EPWMT，触发源才有效）

ADC2_TRIGSEL[3:0]	ADC2 触发源
0000	PWMA_TA
0001	PWMC_TA
0010	PWME_TA
0011	PWMA_TA2
0100	PWMC_TA2
0101	PWME_TA2
0110	PWMB_TA
0111	PWMD_TA
1000	PWMF_TA
1001	1
1010	0
1011	PWMA_TA
1100	TIMER17 OV
1101	TIMER18 OV
1110	TIMER17 OV phase 180
1111	TIMER18 OV phase 180

PWMSEL[1:0]：PWM 触发信号配置

PWM 触发信号	PWMSEL[1:0]			
	00	01	10	11
PWMA_TA	0	PWMA TRGO 上升沿	PWMA TRGO 下降沿	PWMA TRGO 上升沿+下降沿
PWMC_TA	0	PWMC TRGO 上升沿	PWMC TRGO 下降沿	PWMC TRGO 上升沿+下降沿
PWME_TA	0	PWME TRGO 上升沿	PWME TRGO 下降沿	PWME TRGO 上升沿+下降沿
PWMA_TA2	0	PWMA TRGO2 上升沿	PWMA TRGO2 下降沿	PWMA TRGO2 上升沿+下降沿
PWMC_TA2	0	PWMC TRGO2 上升沿	PWMC TRGO2 下降沿	PWMC TRGO2 上升沿+下降沿
PWME_TA2	0	PWME TRGO2 上升沿	PWME TRGO2 下降沿	PWME TRGO2 上升沿+下降沿
PWMB_TA	0	PWMB TRGO 上升沿	PWMB TRGO 下降沿	PWMB TRGO 上升沿+下降沿
PWMD_TA	0	PWMD TRGO 上升沿	PWMD TRGO 下降沿	PWMD TRGO 上升沿+下降沿
PWMF_TA	0	PWMF TRGO 上升沿	PWMF TRGO 下降沿	PWMF TRGO 上升沿+下降沿

22.2.9 ADC2 中断控制寄存器（ADC2INTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
ADC2INTR	7EFB3EH	-	-	-	-	-	PADC2[1:0]		EADC2

EADC2：ADC2 中断使能控制位

0：禁止 ADC2 中断

1：允许 ADC2 中断

PADC2[1:0]：ADC2 中断优先级控制位

00：ADC2 中断优先级为 0 级（最低级）

01：ADC2 中断优先级为 1 级（较低级）

10：ADC2 中断优先级为 2 级（较高级）

11：ADC2 中断优先级为 3 级（最高级）

22.3 ADC 静态特性

符号	描述	最小值	典型值	最大值	单位
RES	分辨率	-	12	-	Bits
E_T	整体误差	-	0.5	1	LSB
E_O	偏移误差	-	-0.1	1	LSB
E_G	增益误差	-	0	1	LSB
E_D	微分非线性误差	-	0.7	1.5	LSB
E_I	积分非线性误差	-	1	2	LSB
R_{AIN}	通道等效电阻	-	∞	-	ohm
R_{ESD}	采样保持电容前串接的抗静电电阻	-	700	-	ohm
C_{ADC}	内部采样保持电容	-	16.5	-	pF

22.4 ADC 相关计算公式

22.4.1 ADC 速度计算公式

ADC 的转换速度由 ADCCFG 寄存器中的 SPEED 和 ADCTIM 寄存器共同控制。转换速度的计算公式如下所示：

$$12位ADC转换速度 = \frac{MCU工作频率SYSclk}{2\times(SPEED[3:0]+1)\times[(CSSETUP+1)+(CSHOLD+1)+(SMPDUTY+1)+12]}$$

注意：

- 12 位 ADC 的速度不能高于 800KHz
- SMPDUTY 的值不能小于 10，建议设置为 15
- CSSETUP 可使用上电默认值 0
- CHOLD 可使用上电默认值 1（ADCTIM 建议设置为 3FH）

22.4.2 ADC 转换结果计算公式

$$12位ADC转换结果 = 4096 \times \frac{ADC被转换通道的输入电压Vin}{AVcc电压}$$

22.4.3 反推 ADC 输入电压计算公式

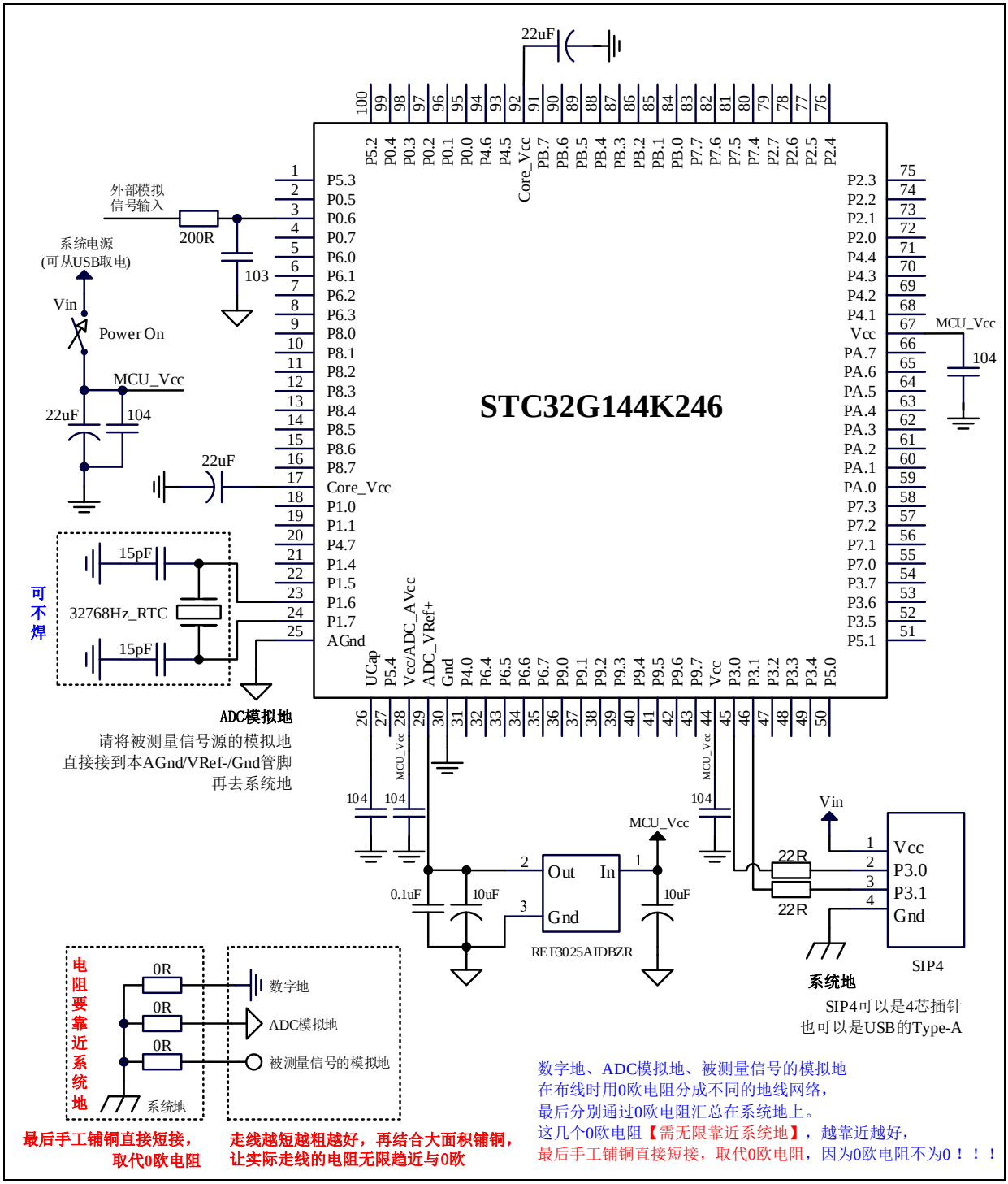
$$ADC被转换通道的输入电压Vin = AVcc电压 \times \frac{12位ADC转换结果}{4096}$$

22.4.4 反推工作电压计算公式

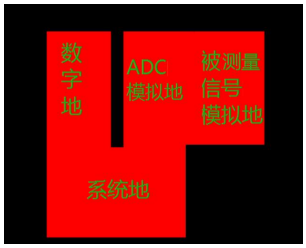
$$MCU工作电压Vcc = 4096 \times \frac{ADC被转换通道的输入电压Vin}{12位ADC转换结果}$$

22.5 ADC 应用参考线路图

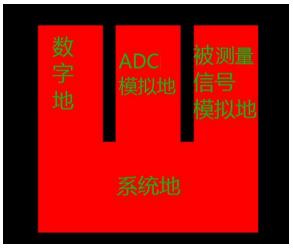
22.5.1 ADC 参考线路图 1 -- 有外部 专业 ADC 转换参考源 REF3025



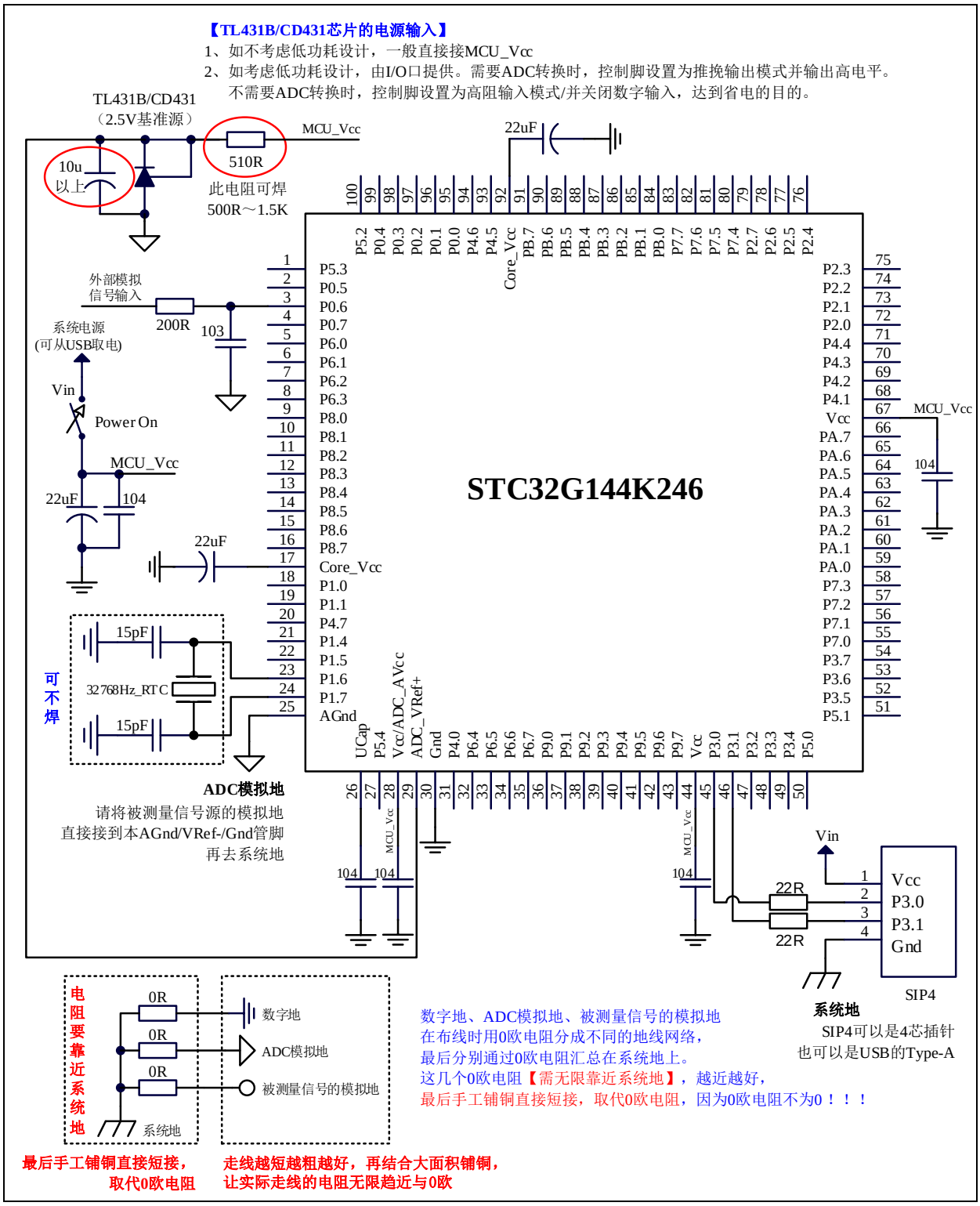
PCB 布线示意图



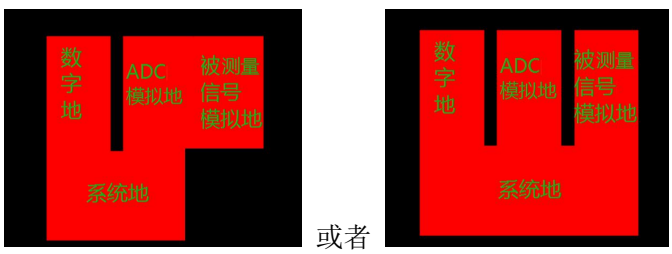
或者



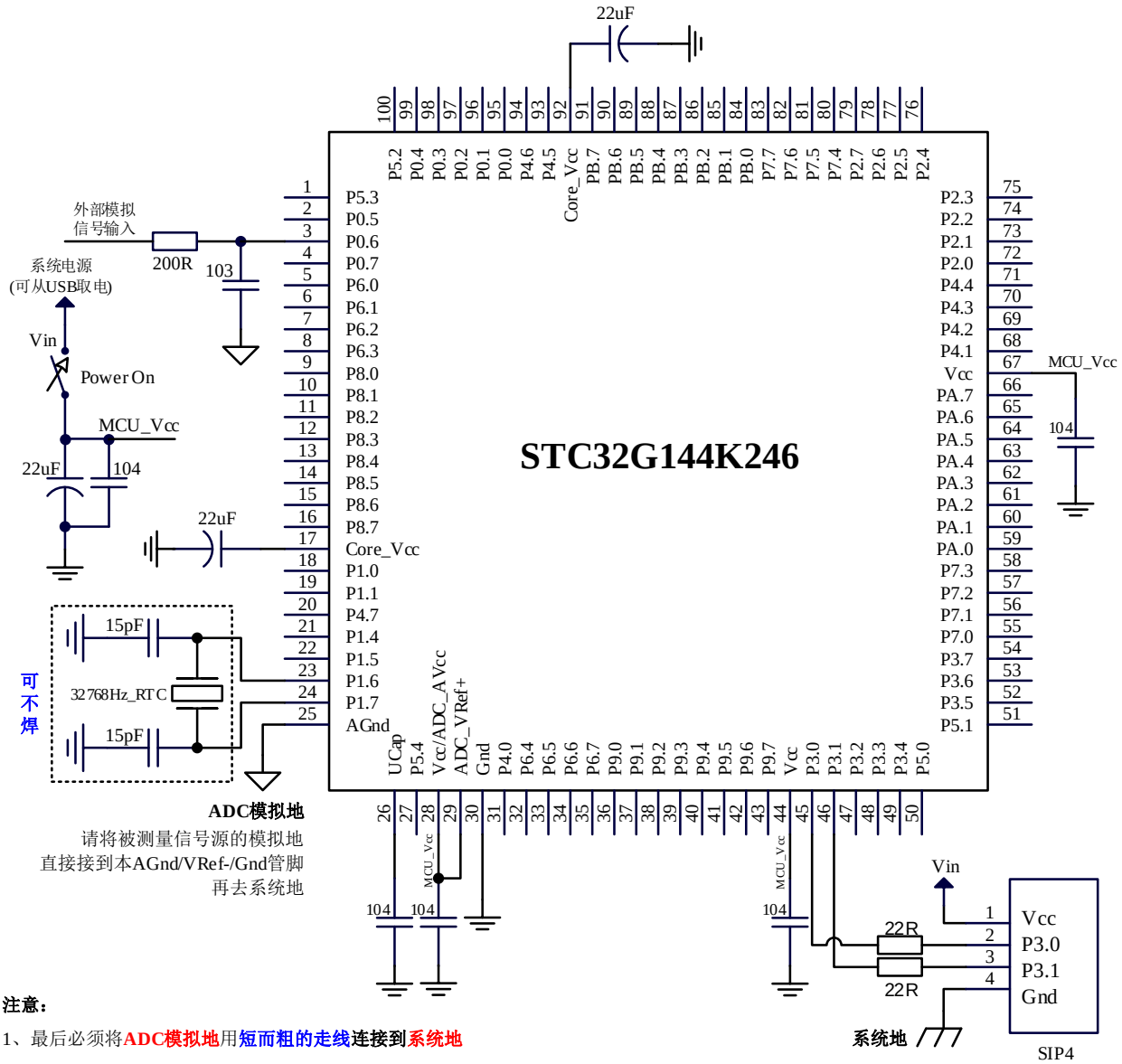
22.5.2 ADC 参考线路图 2 -- 有外部 普通 ADC 转换参考源 TL431B



PCB 布线示意图



22.5.3 ADC 参考线路图 3 -- 无外接 ADC 参考源，直接用内部 1.19V BandGap，反推外部输入电压



注意：

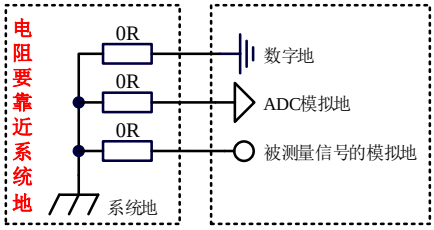
1、最后必须将**ADC模拟地**用**短而粗的走线连接到系统地**

2、假定2次采样转换间隔足够短，则【ADC_VRef+ = ADC_Vcc = MCU_Vcc】不变，

使用内部1.19V辅助信号源：对 ADC15采样转换一次， ADC15内部固定接 BandGap电压 1.19V，
就可以知道 ADC_VRef+，再对其他外部ADC输入通道 ADCx 采样转换一次就知道转换后的 ADCx 的值

【公式】外部通道输入电压**Vx = BGV / resbg * resx**；（Vx单位：mV）

BGV：是出厂前标定的值，从CHIPID中读取；
resbg：是ADC15采样转换内部辅助信号源BandGap电压后获得的转换结果；
resx ： 是对外部ADCx输入通道采样转换后获得的转换结果；
Vx： 计算得到的外部ADCx输入通道的实际电压，采样转换2次，只需计算一次即可。

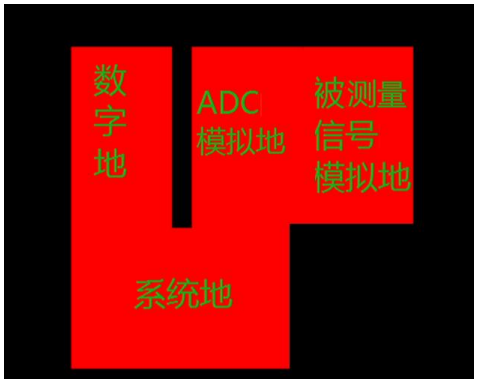


数字地、ADC模拟地、被测量信号的模拟地
在布线时用0欧电阻分成不同的地线网络，
最后分别通过0欧电阻汇总在系统地上。
这几个0欧电阻【需无限靠近系统地】，越近越好，
最后手工铺铜直接短接，取代0欧电阻，因为0欧电阻不为0！！！

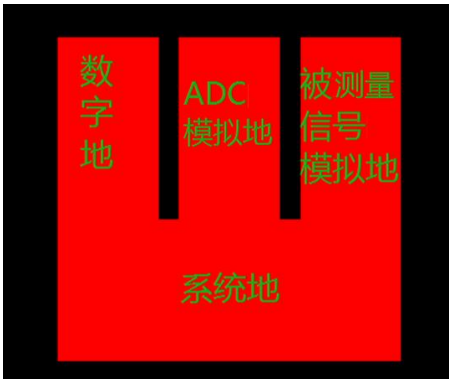
最后手工铺铜直接短接，取代0欧电阻

走线越短越粗越好，再结合大面积铺铜，让实际走线的电阻无限趋近与0欧

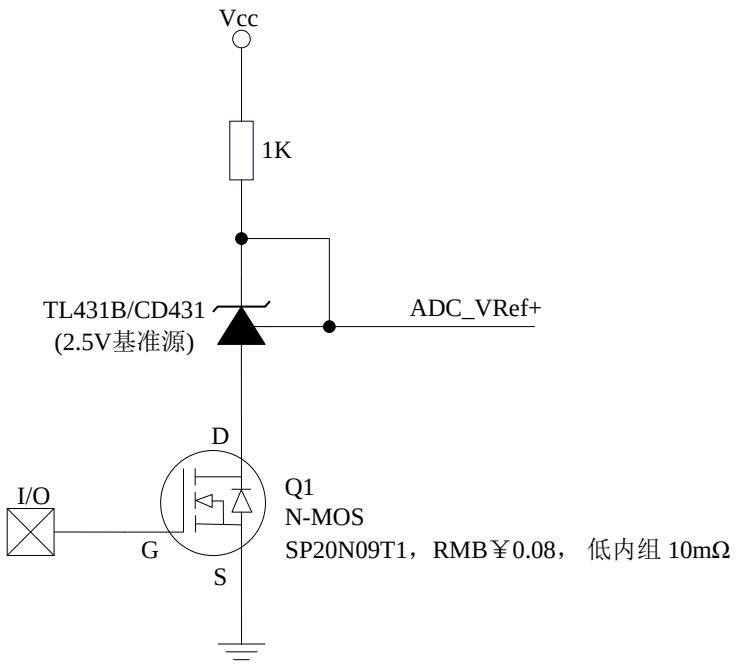
PCB 布线示意图



或者

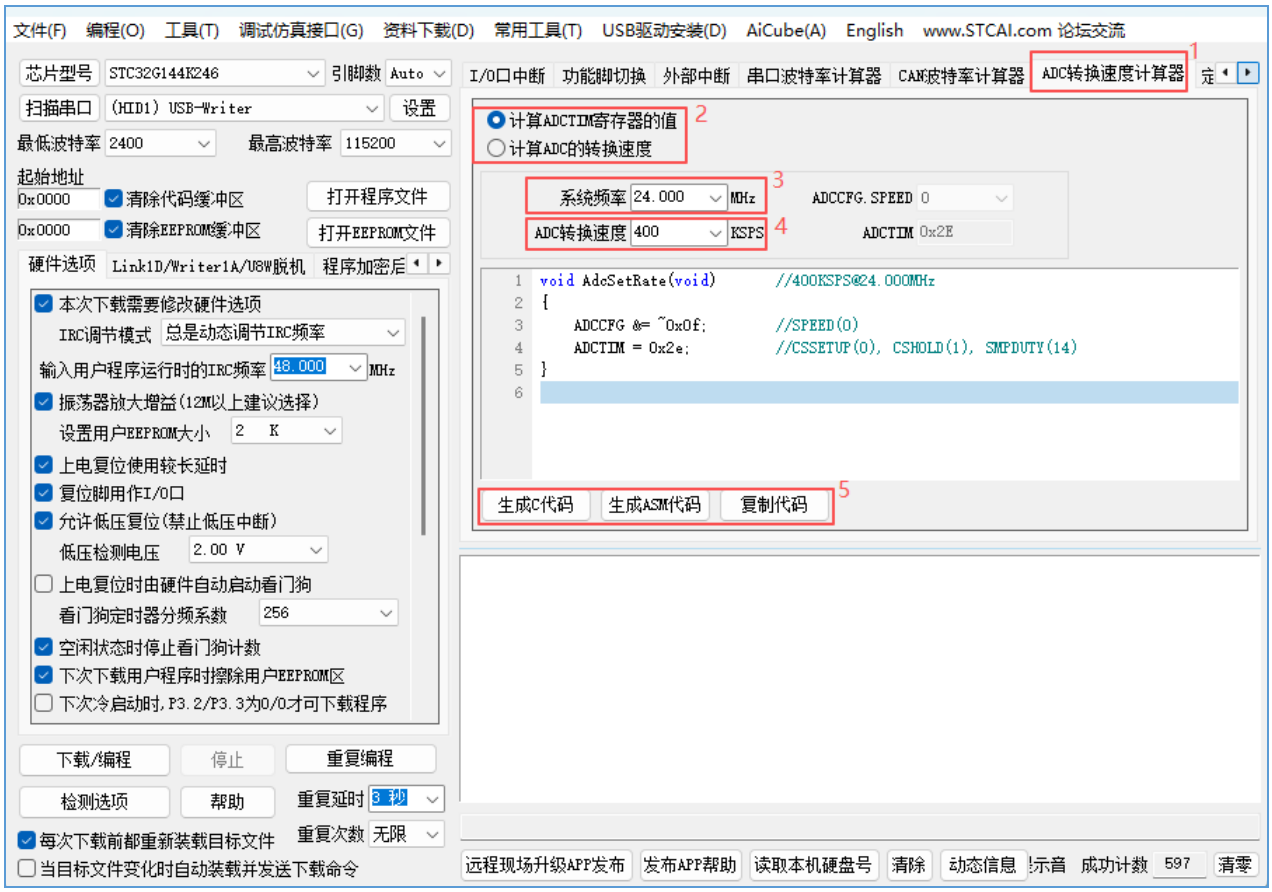


22.5.4 TL431B/CD431 省电模式正规控制电路



工作时， I/O = 1，N-MOS 导通，ADC_VRef+ = +2.5V
需省电时，I/O = 0，N-MOS 关闭，无电流，ADC_VRef+ = Vcc

22.6 AiCube-ISP | ADC 转换速度计算器工具



①：在下载软件中选择“ADC 转换速度计算器”功能页，进入 ADC 代码生成界面

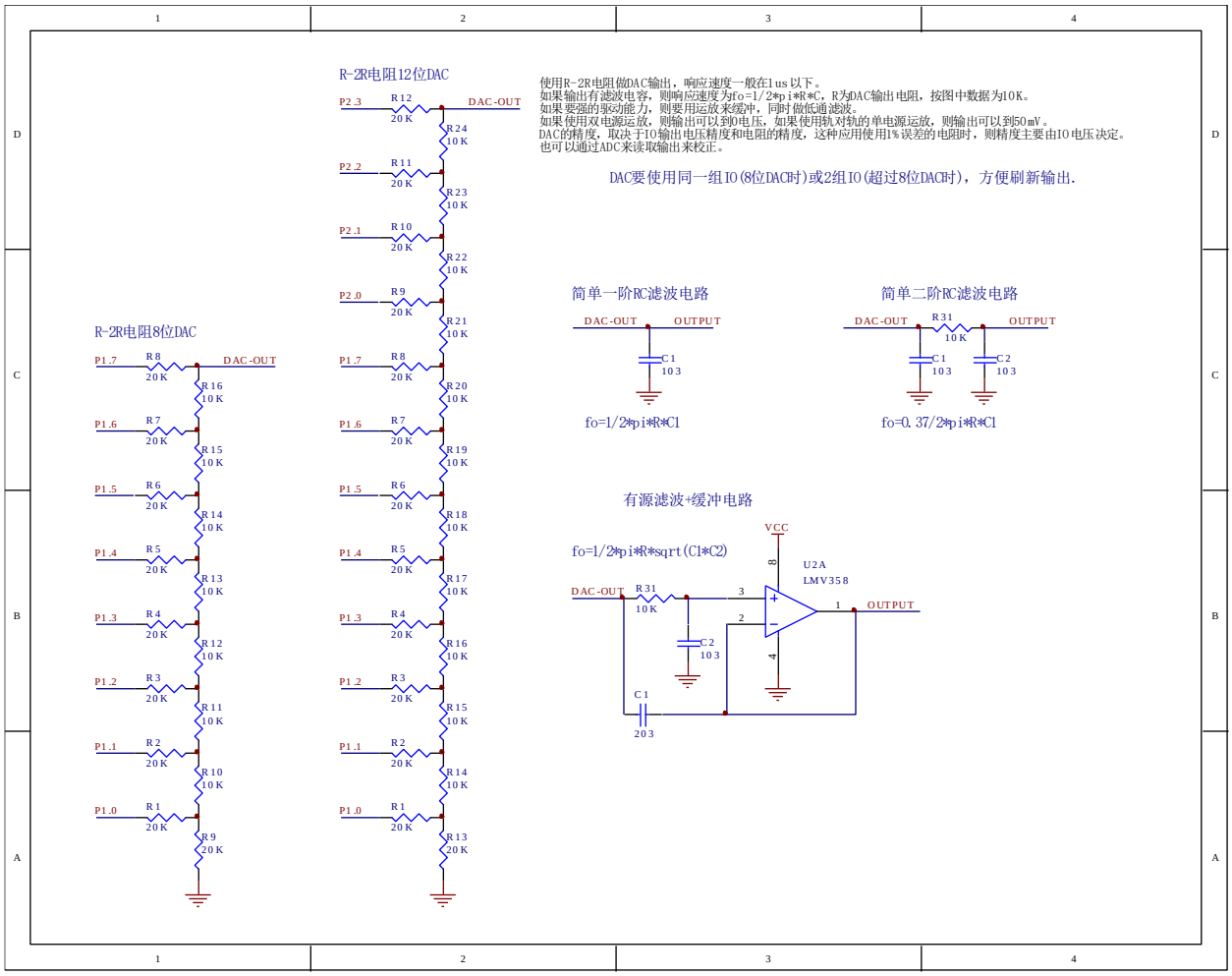
②：选择“根据速度计算配置寄存器功能”还是“根据寄存器配置反推转换速度”

③：设置系统工作频率

④：设置 ADC 转换速度

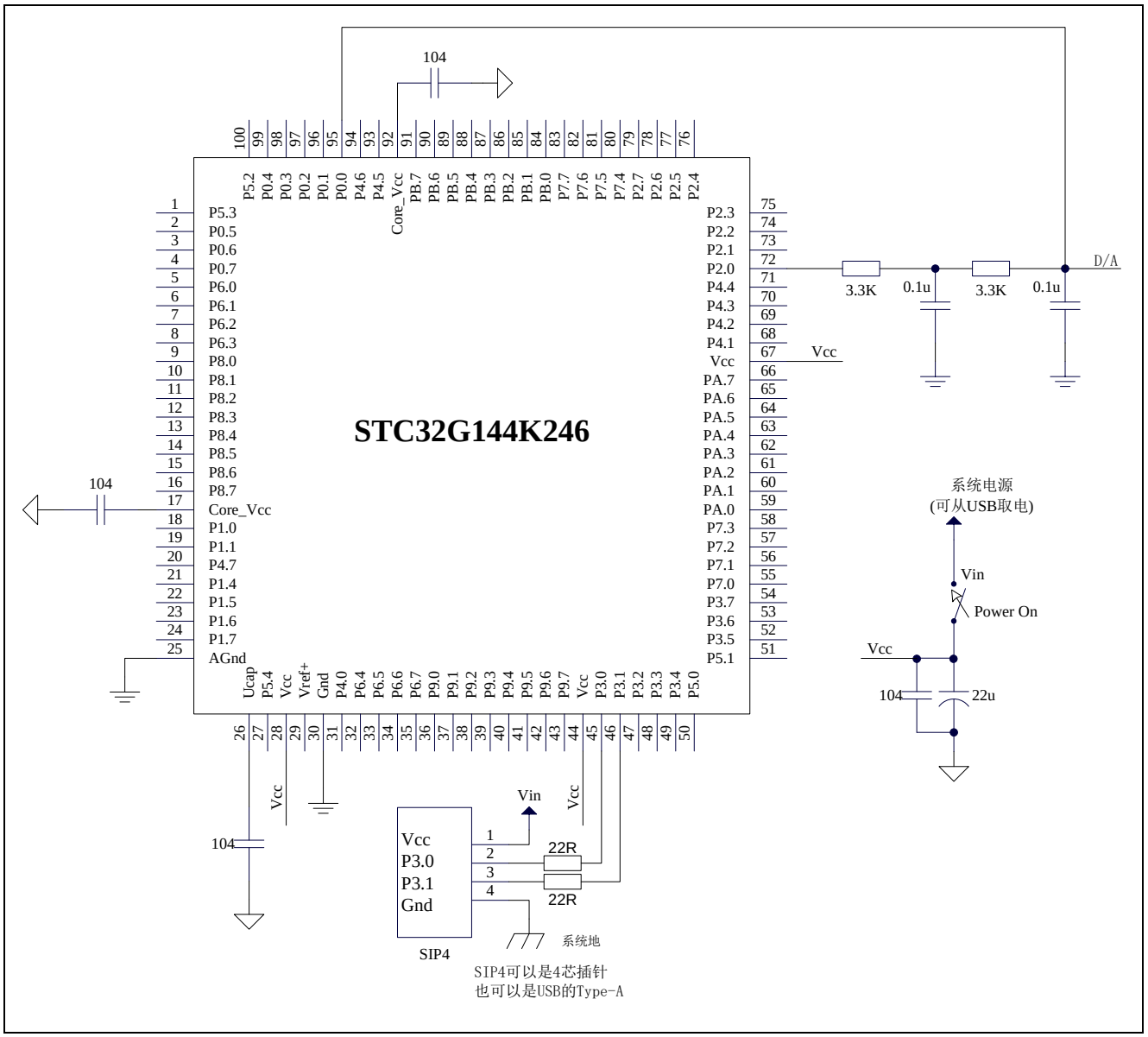
⑤：手动生成 C 代码或者 ASM 代码，复制范例

22.7 使用 I/O 和 R-2R 电阻分压实现 DAC 的经典线路图



22.8 利用 PWM 实现 16 位 DAC 的参考线路图

STC32G144K246 系列单片机的高级 PWM 定时器可输出 16 位的 PWM 波形，再经过两级低通滤波即可产生 16 位的 DAC 信号，通过调节 PWM 波形的高电平占空比即可实现 DAC 信号的改变。应用线路图如下图所示，输出的 DAC 信号可输入到 MCU 的 ADC 进行反馈测量。



22.9 ADC 高级应用 1-使用定时器同时触发两组 ADC 实现两路 ADC 同步转换

示例代码后续补充

22.10 ADC 高级应用 2-两组 ADC 同时采样两路信号实现差分 ADC 转换

示例代码后续补充

22.11 ADC 高级应用 3-使用定时器交替触发两组 ADC 对同一个信号进行采样,实现 ADC 转换速度加倍的目的

示例代码后续补充

23 两组 12 位数模转换（DAC1/DAC2）

产品线	DAC
STC32G144K246 系列	**两组*12 位**

STC32G144K246 系列单片机内部集成了两组 12 位高速 D/A 转换器。DAC 模块本身不自带缓冲器，不具备电流驱动能力，DAC 的输出电平无法直接输出到芯片的外部管脚。本模块较常使用的场合为提供芯片片上比较器或片上运放当作参考电压或偏置电压，也可以接上芯片片上缓冲级后驱动芯片外部电路。

本模块是一个 R-2R 数模转换器（DAC），支持高达 2MHz 转换速度，具有 12 位分辨率。该 DAC 可用于各种低功耗应用。

23.1 主要特性

DAC 有如下主要特性：

- 12 位分辨率
- 保证在所有操作范围内都是单调的
- 最大速度：2MHz 转换速度
- 两组 DAC 的操作模式有独立模式和同步模式
  - 独立模式：两组 DAC 分别使用各自的 XFR 进行独立控制
  - 同步模式：使用定时器 17 和定时器 18 同步触发

23.2 DAC1

23.2.1 DAC1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DAC1_CR	DAC1 控制寄存器	7EF610H	DMAEN	DAC1TG	DAC1TS[1:0]		DAC1IP[1:0]		EDAC1I	DAC1EN	0000,0000
DAC1_STA	DAC1 状态寄存器	7EF611H	BUSY	-	-	-	-	-	-	DAC1IF	0xxx,xxx0
DAC1_DATH	DAC1 数据寄存器	7EF612H	-	-	-	-	DAC1DAT[11:8]				xxxx,0000
DAC1_DATL	DAC1 数据寄存器	7EF613H	DAC1_DAT[7:0]								0000,0000
DAC1_DIVH	DAC1 时钟分频寄存器	7EF614H	-	-	-	-	-	-	DAC1_DIV[9:8]		xxxx,xx00
DAC1_DIVL	DAC1 时钟分频寄存器	7EF615H	DAC1_DIV[7:0]								0000,0000

23.2.2 DAC1 控制寄存器（DAC1_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DAC1_CR	7EF610H	DMAEN	DAC1TG	DAC1TS[1:0]		DAC1IP[1:0]		EDAC1I	DAC1EN

DMAEN：DMA模式控制位

0：禁止 DMA 模式

1：使能 DMA 模式

DAC1TG：触发DAC输出

0：无效

1：触发 DAC 输出，硬件自动清零

DAC1TS[1:0]：DAC1触发源选择

DAC1TS[1:0]	自动触发源	触发说明
00	单次触发	软件使用DAC1TG触发
01	定时器17自动触发	
10	定时器18自动触发	
11	连续触发模式	

DAC1IP[1:0]：DAC1中断优先级控制位

00：DAC1 中断优先级为 0 级（最低级）

01：DAC1 中断优先级为 1 级（较低级）

10：DAC1 中断优先级为 2 级（较高级）

11：DAC1 中断优先级为 3 级（最高级）

EDAC1I：DAC1中断控制位

0：禁止 DAC1 中断

1：使能 DAC1 中断

DAC1EN：DAC1模块使能控制位

0：禁止 DAC1 功能

1：使能 DAC1 功能

23.2.3 DAC1 状态寄存器（DAC1_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DAC1_STA	7EF611H	BUSY	-	-	-	-	-	-	DAC1IF

BUSY：DAC1模块忙标准位（只读）

DAC1IF：DAC1中断请求标志位，需要软件写0清零

23.2.4 DAC1 数据寄存器（DAC1_DATH/DATL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DAC1_DATH	7EF612H	-	-	-	-	DAC1_DAT[11:8]			
DAC1_DATL	7EF613H	DAC1_DAT[7:0]							

DAC1_DAT[11:0]：DAC1 数模转换数据寄存器

23.2.5 DAC1 时钟分频寄存器（DAC1_DIVH/DIVL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DAC1_DIVH	7EF614H	-	-	-	-	-	-	DAC1_DIV[9:8]	
DAC1_DIVL	7EF615H	DAC1_DIV[7:0]							

DAC1_DIV[9:0]：DAC1 时钟分频。

DAC1 模块的工作频率=系统时钟 / 4 / DAC1_DIV[9:0]，当 DAC1_DIV[9:0]的值为 0 时，DAC1 直接使用系统时钟。

必须保证 DAC1 模块的工作频率不得高于 2MHz

23.3 DAC2

23.3.1 DAC2 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DAC2_CR	DAC2 控制寄存器	7EF628H	DMAEN	DAC2TG	DAC2TS[1:0]		DAC2IP[1:0]		EDAC2I	DAC2EN	0000,0000
DAC2_STA	DAC2 状态寄存器	7EF629H	BUSY	-	-	-	-	-	-	DAC2IF	0xxx,xxx0
DAC2_DATH	DAC2 数据寄存器	7EF62AH	-	-	-	-	DAC2DAT[11:8]				xxxx,0000
DAC2_DATL	DAC2 数据寄存器	7EF62BH	DAC2_DAT[7:0]								0000,0000
DAC2_DIVH	DAC2 时钟分频寄存器	7EF62CH	-	-	-	-	-	-	DAC2_DIV[9:8]		xxxx,xx00
DAC2_DIVL	DAC2 时钟分频寄存器	7EF62DH	DAC2_DIV[7:0]								0000,0000

23.3.2 DAC2 控制寄存器（DAC2_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DAC2_CR	7EF628H	DMAEN	DAC2TG	DAC2TS[1:0]		DAC2IP[1:0]		EDAC2I	DAC2EN

DMAEN：DMA模式控制位

0：禁止 DMA 模式

1：使能 DMA 模式

DAC2TG：触发DAC输出

0：无效

1：触发 DAC 输出，硬件自动清零

DAC2TS[1:0]：DAC2触发源选择

DAC2TS[1:0]	自动触发源	触发说明
00	单次触发	软件使用DAC2TG触发
01	定时器17自动触发	
10	定时器18自动触发	
11	连续触发模式	

DAC2IP[1:0]：DAC2中断优先级控制位

00：DAC2 中断优先级为 0 级（最低级）

01：DAC2 中断优先级为 1 级（较低级）

10：DAC2 中断优先级为 2 级（较高级）

11：DAC2 中断优先级为 3 级（最高级）

EDAC2I：DAC2中断控制位

0：禁止 DAC2 中断

1：使能 DAC2 中断

DAC2EN：DAC2模块使能控制位

0：禁止 DAC2 功能

1：使能 DAC2 功能

23.3.3 DAC2 状态寄存器（DAC2_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DAC2_STA	7EF629H	BUSY	-	-	-	-	-	-	DAC2IF

BUSY：DAC2模块忙标准位（只读）

DAC2IF：DAC2中断请求标志位，需要软件写0清零

23.3.4 DAC2 数据寄存器（DAC2_DATH/DATL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DAC2_DATH	7EF62AH	-	-	-	-	DAC2_DAT[11:8]			
DAC2_DATL	7EF62BH	DAC2_DAT[7:0]							

DAC2_DAT[11:0]：DAC2 数模转换数据寄存器

23.3.5 DAC2 时钟分频寄存器（DAC2_DIVH/DIVL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DAC2_DIVH	7EF62CH	-	-	-	-	-	-	DAC2_DIV[9:8]	
DAC2_DIVL	7EF62DH	DAC2_DIV[7:0]							

DAC2_DIV[9:0]：DAC2 时钟分频。

DAC2 模块的工作频率=系统时钟 / 4 / DAC2_DIV[9:0]，当 DAC2_DIV[9:0]的值为 0 时，DAC2 直接使用系统时钟。

必须保证 DAC2 模块的工作频率不得高于 2MHz

23.4 电气特性

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
VDDA50	Analog power supply voltage	-	1.8	5	5.5	V
VDDD15	Internal core power supply voltage	-	1.35	1.5	1.65	V
Temp	Operating ambient temperature	-	-40	25	125	℃
RES	Resolution	-		12	-	Bit
INL	Integral non-linearity error	-	-4	±1.0	4.0	LSB
DNL	Differential non-linearity error	-		±1	2	LSB
Offset	Offset error drift			+/-1.6		uV/℃
Gain error	Gain error			±0.5	±1	LSB
V_{refp}	Positive reference input voltage range		1.2		VDDA50	V
V_{refn}	Negative reference input voltage range			0		V
CLK	Clock frequency	-	0.2	2		MHz
I_{vrefp}	Power of VREFP@ enable mode at 5V. Power of VREFP @ disable mode	-	-	0.6 10		mA nA
I_{VDDA50}	Power of VDDA50 @ enable mode Power of VDDA50 @ disable mode.	-	-	1 5		uA nA
I_{VDDD15}	PowerVDDD15 @ enable mode VDDD15 @ disable modePower	-	-	1 5		uA nA
C_{load}	Output Loading			10		fF
R_{out}	Output resistance			4		KOhm

23.5 范例程序

23.5.1 【DAC+OP】应用：【DAC1+OP1；DAC2+OP3；DAC2+OP4】同时输出 3 个正弦波

STC32G144K246 的【DAC+OP】应用：（点击到“技术交流及选型论坛”相关链接）
【DAC1+OP1；DAC2+OP3；DAC2+OP4】同时输出 3 个正弦波

//请直接下载"obj"里面的"DAC.hex"测试，下载时选择主频 48MHz。
3 个 DAC 同时输出正弦波，输出正弦波最高频率大约 200KHz（DAC+运放的带宽决定），波形漂亮！
DAC1+OP1 从 P0.7 输出正弦波，P0.5(OP1 负输出入端)要悬空。输出接一阶 RC 滤波（3K+220pF）送示波器通道 1。
DAC2+OP3 从 P7.4 输出正弦波，P7.1(OP3 负输出入端)要悬空。输出接一阶 RC 滤波（3K+220pF）送示波器通道 2。
DAC2+OP4 从 P4.0 输出正弦波，P6.5(OP4 负输出入端)要悬空。输出接一阶 RC 滤波（3K+220pF）送示波器通道 3。
输出正弦波可自行决定，我给的数据是 500Hz ~100KHz。

```
//主循环
while(1) //循环 40T, 采样 1.2MHz@48MHz, 3MHz@120MHz
{
//      sum1 += 22;             //1000Hz, N = fx * 65536/3000000 = fx*1024/46875
//      sum2 += 11;             //500Hz

//      sum1 += 218;            //10KHz
//      sum2 += 109;            // 5KHz

    sum1 += 2185;           //100KHz
    sum2 += 1092;           // 50KHz

//主循环
while(1) //循环 40T, 采样 1.2MHz@48MHz, 3MHz@120MHz
{
//      sum1 += 22;             //1000Hz, N = fx * 65536/3000000 = fx*1024/46875
//      sum2 += 11;             //500Hz

//      sum1 += 218;            //10KHz
//      sum2 += 109;            // 5KHz

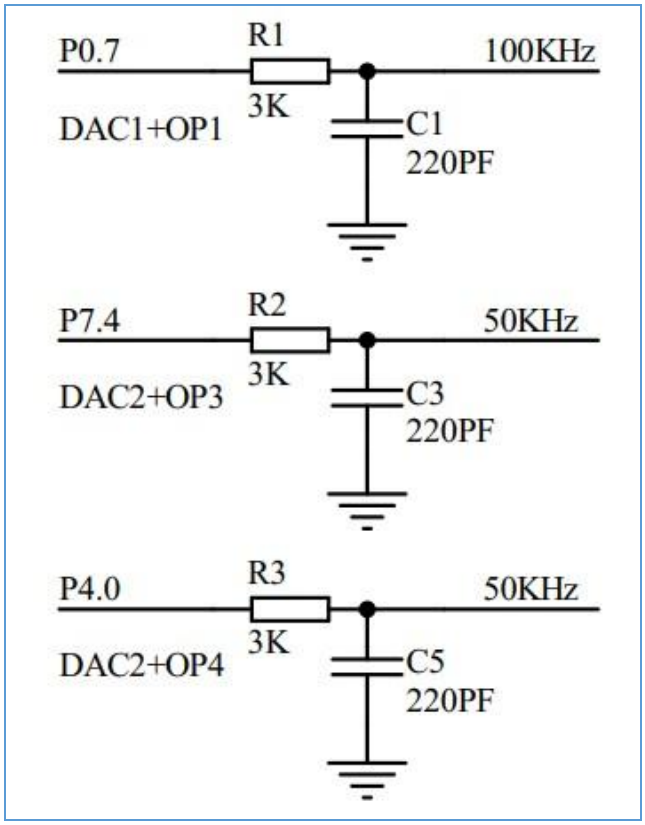
    sum1 += 2185;           //100KHz

    sum2 += 1092;           // 50KHz

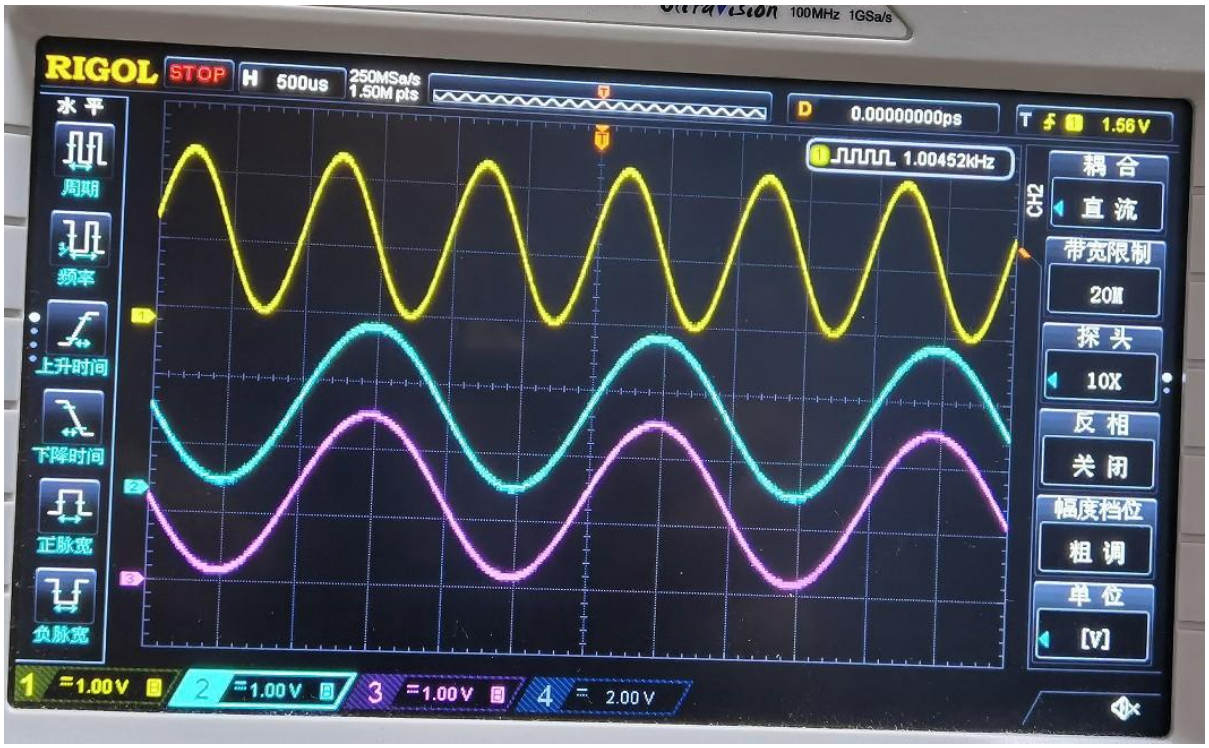
    DAC1_DAT = T_SineTable[(u8)(sum1 >>8)];
```

```
    DAC1_CR = 0x41;                 //刷新 DAC1 输出
    DAC2_DAT = T_SineTable[(u8)(sum2 >>8)];
    DAC2_CR = 0x41;                 //刷新 DAC2 输出
}
```

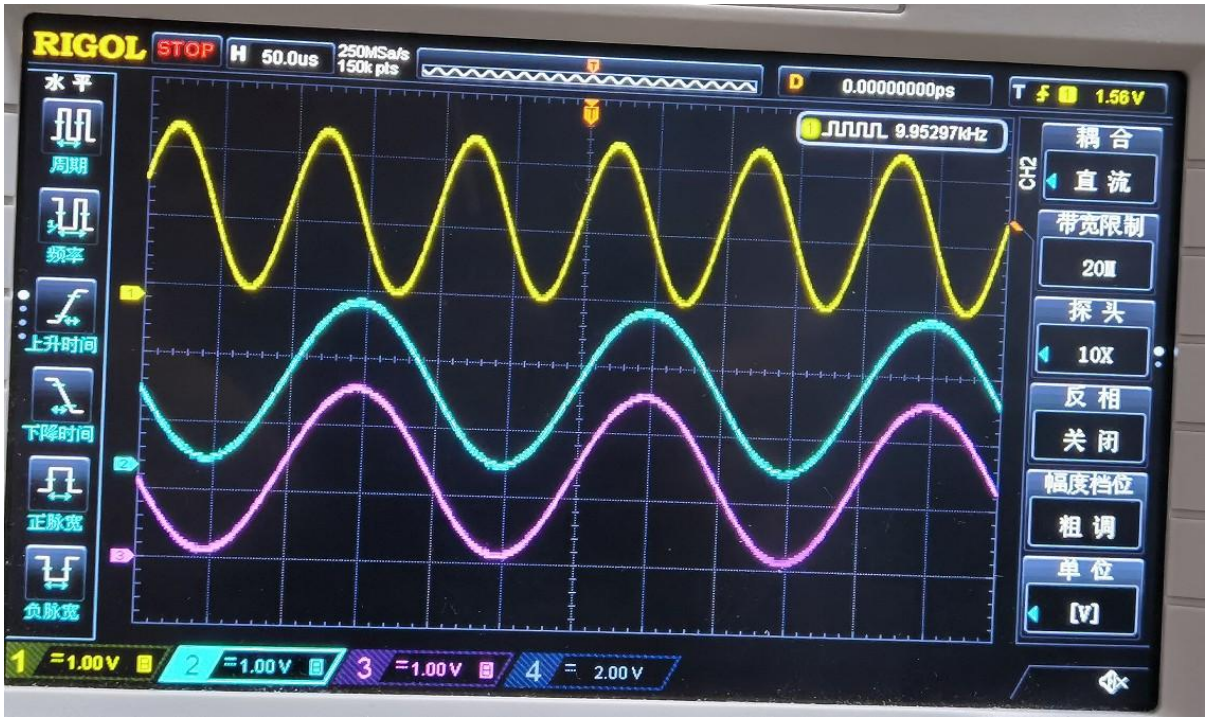
滤波电路：



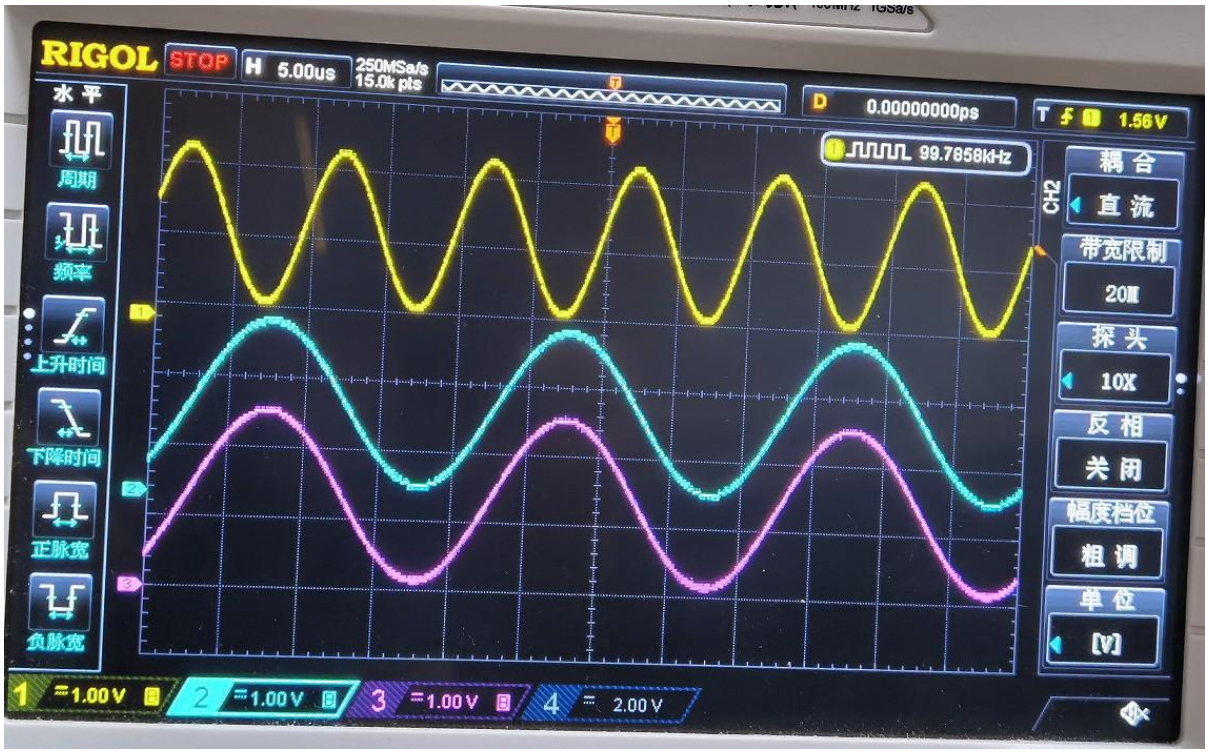
输出 1000Hz 和 500Hz 正弦波照片：



输出 10KHz 和 5KHz 正弦波照片：



输出 100KHz 和 50KHz 正弦波照片：



【源代码】：

/********** 功能说明 **********

请先别修改程序，直接下载"obj"里面的"DAC.hex"测试，下载时选择主频 48MHz.

DAC1+OP1 从 P0.7 输出正弦波，P0.5(OP1 负输出入端)要悬空。输出接一阶 RC 滤波（3K+220pF）送示

波器通道 1。
DAC2+OP3 从 P7.4 输出正弦波，P7.1(OP3 负输出入端)要悬空。输出接一阶 RC 滤波（3K+220pF）送示波器通道 2。
DAC2+OP4 从 P4.0 输出正弦波，P6.5(OP4 负输出入端)要悬空。输出接一阶 RC 滤波（3K+220pF）送示波器通道 3。

输出正弦波可自行决定，我给的数据是 500Hz ~100KHz。
***/

```
#define    HPLL_ENABLE              1                       // 1:启动 PLL，0：关闭 PLL

#if (HPLL_ENABLE == 0)
    //不用 PLL
    #define FOSC              48000000UL                    //定义主时钟
#else
    //使用 PLL
    #define FOSC             120000000UL                    //定义主时钟,
    #define HPLL_CLK         480000000UL              //HPLL 时钟
    #define HPLL_SCALE                    40              //HPLL 时钟倍频系数 26~41，
                                          //串口或 USB 下载时选择 48MHz,
                                          //HPLL 时钟  = 48MHz/8 *2 *40 = 480MHz
#endif

#include  "STC32G144K246.h"
//#include"SineTable-12bit-256dots.h"
const unsigned int T_SIN[256]=
{
    2048,2098,2148,2198,2248,2298,2347,2397,2446,2495,2544,2592,2640,2688,2735,2782,
    2829,2875,2920,2965,3010,3054,3097,3139,3181,3223,3263,3303,3342,3380,3418,3455,
    3490,3525,3560,3593,3625,3656,3687,3716,3744,3772,3798,3823,3847,3870,3892,3913,
    3933,3951,3969,3985,4000,4014,4027,4038,4049,4058,4066,4073,4078,4082,4086,4087,
    4088,4087,4086,4082,4078,4073,4066,4058,4049,4038,4027,4014,4000,3985,3969,3951,
    3933,3913,3892,3870,3847,3823,3798,3772,3744,3716,3687,3656,3625,3593,3560,3525,
    3490,3455,3418,3380,3342,3303,3263,3223,3181,3139,3097,3054,3010,2965,2920,2875,
    2829,2782,2735,2688,2640,2592,2544,2495,2446,2397,2347,2298,2248,2198,2148,2098,
    2048,1998,1948,1898,1848,1798,1749,1699,1650,1601,1552,1504,1456,1408,1361,1314,
    1267,1221,1176,1131,1086,1042,999,957,915,873,833,793,754,716,678,641,
    606,571,536,503,471,440,409,380,352,324,298,273,249,226,204,183,
    163,145,127,111,96,82,69,58,47,38,30,23,18,14,10,9,
    8,9,10,14,18,23,30,38,47,58,69,82,96,111,127,145,
    163,183,204,226,249,273,298,324,352,380,409,440,471,503,536,571,
    606,641,678,716,754,793,833,873,915,957,999,1042,1086,1131,1176,1221,
    1267,1314,1361,1408,1456,1504,1552,1601,1650,1699,1749,1798,1848,1898,1948,1998
};

/*************    本地常量声明      **************/
#define    PGA1_PSEL                3        //[Bit1   Bit0]    正输入端选择     0:P0.4,  1:P0.5,  2:P6.0,
```

```
3:DAC1O
#define  PGA1_NSEL      0       //[Bit3 Bit2]  负输入端选择  0:P0.5, 1:P0.4, 2:P6.1,
3:DAC1O
#define  PGA1_OSEL      0       //[Bit5 Bit4]  输出端选择  0:P0.7, 1:P0.0, 2:P7.4, 3:P6.3
#define  PGA1_MSEL      1       //[Bit7 Bit6]  模式选择  0:non,  1:Buffer, 2:Invert,
3:NonInvert
#define  PGA1_PWD       0       //[Bit0]       掉电控制  0:电源打开，  1：电源关闭
#define  PGA1_IBS       0       //[Bit1]       偏置电流选择  0:偏置电流小  1：偏置电流
大
#define  PGA1_OE        1       //[Bit2]                1:使能放大器外扩接口
#define  PGA1_GSEL      0       //[Bit5~Bit3]  增益选择
                        //反相增益 1->1、2->2、3->4、4->8、5->16、6->32、7->48，
                        //同相增益 0->1、1->2、2->3、3->5、4->9、5->17、6->33、7->49

#define  PGA2_PSEL      3       //[Bit1 Bit0]  正输入端选择  0:P2.6, 1:P6.0, 2:P7.5, 3:P7.7
#define  PGA2_NSEL      0       //[Bit3 Bit2]  负输入端选择  0:P2.7, 1:P6.1, 2:P7.6,
3:DAC2O
#define  PGA2_OSEL      0       //[Bit5 Bit4]  输出端选择  0:P4.2, 1:P0.0, 2:P7.4, 3:P6.2
#define  PGA2_MSEL      1       //[Bit7 Bit6]  模式选择  0:non,  1:Buffer, 2:Invert,
3:NonInvert
#define  PGA2_PWD       0       //[Bit0]       掉电控制  0:电源打开，  1：电源关闭
#define  PGA2_IBS       0       //[Bit1]       偏置电流选择  0:偏置电流小  1：偏置电流
大
#define  PGA2_OE        1       //[Bit2]                1:使能放大器外扩接口
#define  PGA2_GSEL      0       //[Bit5~Bit3]  增益选择
                        //反相增益 1->1、2->2、3->4、4->8、5->16、6->32、7->48，
                        //同相增益 0->1、1->2、2->3、3->5、4->9、5->17、6->33、7->49

#define  PGA3_PSEL      3       //[Bit1 Bit0]  正输入端选择  0:P3.5, 1:P2.0, 2:P7.0,
3:DAC2O
#define  PGA3_NSEL      2       //[Bit3 Bit2]  负输入端选择  0:P5.1, 1:P4.2, 2:P7.1,
3:DAC2O
#define  PGA3_OSEL      2       //[Bit5 Bit4]  输出端选择  0:P3.2, 1:P2.1, 2:P7.4, 3:P5.0
#define  PGA3_MSEL      1       //[Bit7 Bit6]  模式选择  0:non,  1:Buffer, 2:Invert,
3:NonInvert
#define  PGA3_PWD       0       //[Bit0]       掉电控制  0:电源打开，  1：电源关
闭
#define  PGA3_IBS       0       //[Bit1]       偏置电流选择  0:偏置电流小  1：偏置电
流大
#define  PGA3_OE        1       //[Bit2]                1:使能放大器外扩接口
#define  PGA3_GSEL      0       //[Bit5~Bit3]  增益选择
                        //反相增益 1->1、2->2、3->4、4->8、5->16、6->32、7->48，
                        //同相增益 0->1、1->2、2->3、3->5、4->9、5->17、6->33、7->49

#define  PGA4_PSEL      3       //[Bit1 Bit0]  正输入端选择  0:P1.6, 1:P6.3, 2:P6.4,
3:DAC2O
```

```
#define   PGA4_NSEL                 1         //[Bit3  Bit2]   负输入端选择    0:P1.7,  1:P6.5,  2:P3.3,
3:DAC1O
#define   PGA4_OSEL                 1         //[Bit5 Bit4]   输出端选择      0:P1.3, 1:P4.0, 2:P7.4, 3:P6.3
#define   PGA4_MSEL                 1         //[Bit7  Bit6]   模式选择         0:non,   1:Buffer,  2:Invert,
3:NonInvert
#define   PGA4_PWD                  0         //[Bit0]        掉电控制        0:电源打开，   1：电源关闭
#define   PGA4_IBS                  0         //[Bit1]        偏置电流选择   0:偏置电流小   1：偏置电流
大
#define   PGA4_OE                   1         //[Bit2]                        1:使能放大器外扩接口
#define   PGA4_GSEL                 0         //[Bit5~Bit3]   增益选择
                                  //反相增益 1->1、2->2、3->4、4->8、5->16、6->32、7->48，
                                  //同相增益 0->1、1->2、2->3、3->5、4->9、5->17、6->33、7->49

/*************   本地变量声明      ***************/
u16       sum1;
u16 sum2;
u16       amp;                                          //幅度  1~256;
u16       T_SineTable[256];                 //正弦表

/*************   本地函数声明      ***************/
void      delay_us(u16 us)                  //N 微秒延时函数
{
    u8   i;
    do
    {
        i = (u8)(FOSC / 6000000UL) -1;
        while(--i) ;
    }while(--us);
}

//========================================================================
// 函数: void main(void)
// 描述: 主函数.
// 参数: none.
// 返回: none.
// 备注:
//========================================================================
void main(void)
{
    u16  i = 0;

    EAXFR = 1;                    //SFR enable
    CKCON = 0;
    #if (FOSC <= 64000000UL)
    WTST    = 1;                  // 64MHz 以下写 1
    #elif (FOSC <= 90000000UL)
```

```
    WTST   = 2;                     // 64~90MHz 写 2
    #elif (FOSC <= 120000000UL)
    WTST   = 3;                     // 90~120MHz 写 3
    #else
    WTST   = 4;                     // 120MHz 以上写 1
    #endif

    #ifdef HPLL_SCALE
//============ 使用 HPLL 时钟 @48MHz 初始化 USB 后再初始化 PLL =============
HPLLCR = (0<<5) + (HPLL_SCALE-26);
/* (2<<5):HPLL1, (0<<5):HPLL1/2,    HPLL = HPLL_IN*2*(HPLLCR+26), HPLL_IN 必须为 6MHz, 设置 HPLL 的除频系数 HPLLCR = N-26, N=26~41, F_HPLL = PLL 输入时钟*2 * N,  比如 Fosc=48MHz, 8 分频,  N=40 时 F_HPLL = 48M/8*2*40=480M*/
//        HPLLCR &= ~0x10;          //选择 HPLL 输入时钟源为 HIRC, 默认
//        HPLLCR |= 0x10;           //选择 HPLL 输入时钟源为 IRCM
    HPLLPDIV = 8;                   //设置 HPLL 输入时钟预分频为 8@48MHz（HPLL 输入频率必须为 6MHz）
    HPLLCR |= 0x80;                 //使能 HPLL
    CLKDIV = (u8)(HPLL_CLK/2/FOSC);             //系统时钟=HPLL/2/N
    CLKSEL |= 0x04;                 //选择 HPLL/2 作为主时钟源
    #endif

    P5n_push_pull(Pin4);
    HIRCCR = (HIRCCR & ~0x30) | (1 <<4);
/* (HIRCCR:0xFE02)的 IRC20MCR[5:4]为主时钟分频输出的时钟源选择位，0:关闭输出，1:输出系统时钟，2:输出内部高速 HIRC 时钟，3:输出滤波后的内部高速 HIRC 时钟*/
//        MCLKO16_DIV(100);       P1n_push_pull(Pin6);
    MCLKO54_DIV(100);  P5n_push_pull(Pin4);

    delay_us(1000);

//        DAC1_DIVH = 0;
    DAC1_DIV = 2;
/*DAC 时钟大约 200KHz, = MCLK/(DAC1_DIV*4), DAC1_DIV = 0 或 1 均为 1。实测写 1 都可以正常操作 DAC，DAC1+OP1_BUffer 输出 32 点正弦波，采样 2.816MHz@48MHz，正弦波 88KHz，幅度正常，速度响应正常。*/
//        DAC2_DIVH = 0;
        DAC2_DIV = 2;
/*DAC 时钟大约 200KHz, = MCLK/(DAC1_DIV*4), DAC1_DIV = 0 或 1 均为 1。实测写 1 都可以正常操作 DAC，DAC1+OP1_BUffer 输出 32 点正弦波，采样 2.816MHz@48MHz，正弦波 88KHz，幅度正常，速度响应正常。*/

    PGA1_CR1 = (PGA1_MSEL<<6) | (PGA1_OSEL<<4) | (PGA1_NSEL<<2) | PGA1_PSEL;
    PGA1_CR2 = (PGA1_GSEL<<4) | (PGA1_OE<<2)    | (PGA1_IBS<<1)   | PGA1_PWD;
//        PGA2_CR1 = (PGA2_MSEL<<6) | (PGA2_OSEL<<4) | (PGA2_NSEL<<2) | PGA2_PSEL;
//        PGA2_CR2 = (PGA2_GSEL<<4) | (PGA2_OE<<2)    | (PGA2_IBS<<1)   | PGA2_PWD;
```

```
    PGA3_CR1 = (PGA3_MSEL<<6) | (PGA3_OSEL<<4) | (PGA3_NSEL<<2) | PGA3_PSEL;
    PGA3_CR2 = (PGA3_GSEL<<4) | (PGA3_OE<<2)     | (PGA3_IBS<<1)   | PGA3_PWD;
    PGA4_CR1 = (PGA4_MSEL<<6) | (PGA4_OSEL<<4) | (PGA4_NSEL<<2) | PGA4_PSEL;
    PGA4_CR2 = (PGA4_GSEL<<4) | (PGA4_OE<<2)     | (PGA4_IBS<<1)   | PGA4_PWD;
    P0n_HighZ(Pin7+Pin5);
    P7n_HighZ(Pin4+Pin1);
    P4n_HighZ(Pin0);
    P6n_HighZ(Pin5);

    sum1 = 0;
    sum2 = 0;

    amp = 240;                          //幅度 1~256 /256
    for(i=0;     i<128; i++)   T_SineTable[i] = 2048 + (u16)((T_SIN[i] - 2048)*(u32)amp/256);
    for(i=128; i<256; i++)    T_SineTable[i] = 2048 - (u16)((2048 - T_SIN[i])*(u32)amp/256);

    while(1)                                //40T, 采样 1.2MHz@48MHz, 3MHz@120MHz
    {
//                  sum1 += 22;             //1000Hz // N = fx * 65536 / 3000000 = fx*1024/46875
//                  sum2 += 11;             //500Hz

//                  sum1 += 218;            //10KHz
//                  sum2 += 109;            // 5KHz

        sum1 += 2185;       //100KHz
        sum2 += 1092;       // 50KHz

        DAC1_DAT = T_SineTable[(u8)(sum1 >>8)];     DAC1_CR    = 0x41;          //刷新 DAC1 输出
        DAC2_DAT = T_SineTable[(u8)(sum2 >>8)];     DAC2_CR    = 0x41;          //刷新 DAC2 输出
    }
}
```

23.5.2 录音/放音，ADC+DAC 应用：STC32G144K246-实验箱，复读机-A 率-存储在 SRAM

录音/放音，ADC+DAC 应用：STC32G144K246-实验箱（点击到“技术交流及选型论坛”相关链接）

ADC+DAC 做 双工的音频输入、输出，不用专业的 I2S！

类似复读机，缓冲 8 秒的声音，用户可自行修改成

数字录音、数字对讲机、播放器等等，实现低成本语音方案。

功能说明：

使用 **STC32G144K246-LQFP100 实验箱-V1.0** 的 **144KSRAM 做存储 ！**

咪头电路(**ADC2**-10 通道)做**语音输入**、**DAC2+OP4(P4.0)输出声音**，实现实时、单声道的音频输入输出。

不需要专门的音频 CODEC 芯片，低成本，还能保证良好的音质。

A 率的原始 PCM 采样信号位长就是 12bit 的，压缩成 8bit，符合国际电工委的协议标准。

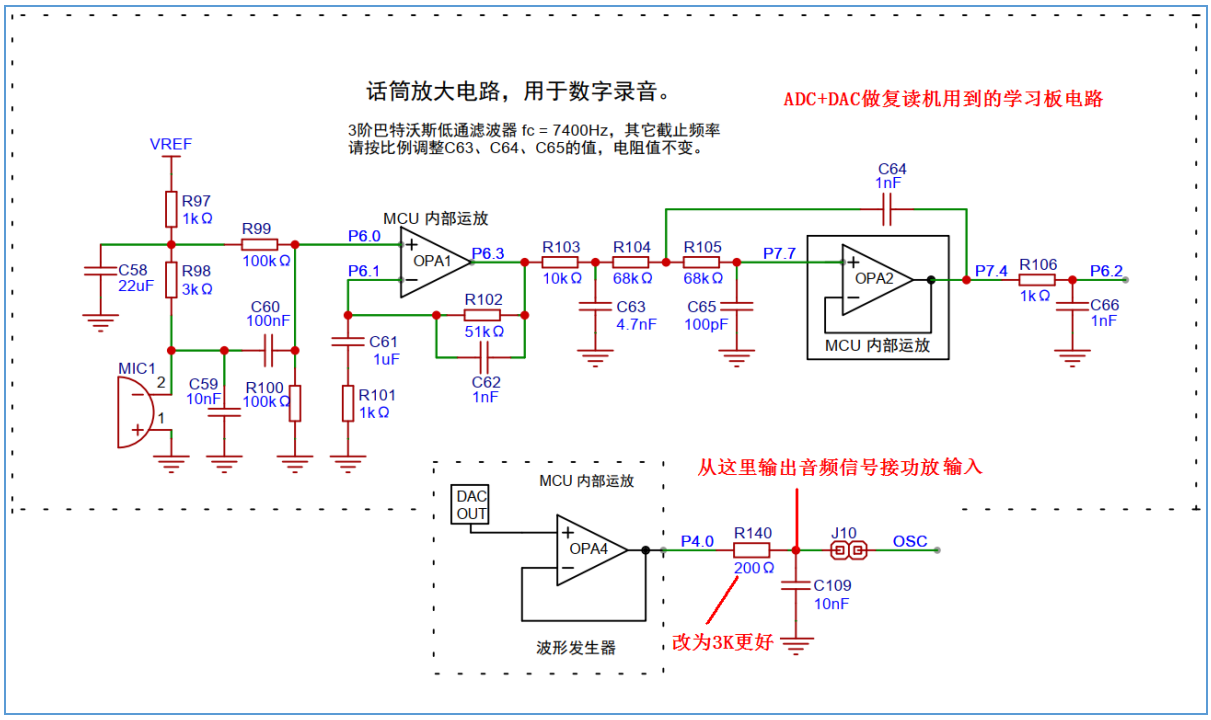
使用 12 位的 DAC 输出，刚好能满足 A 率的动态范围，音质也很好。

咪头、话筒输入放大电路的噪声越低越好。

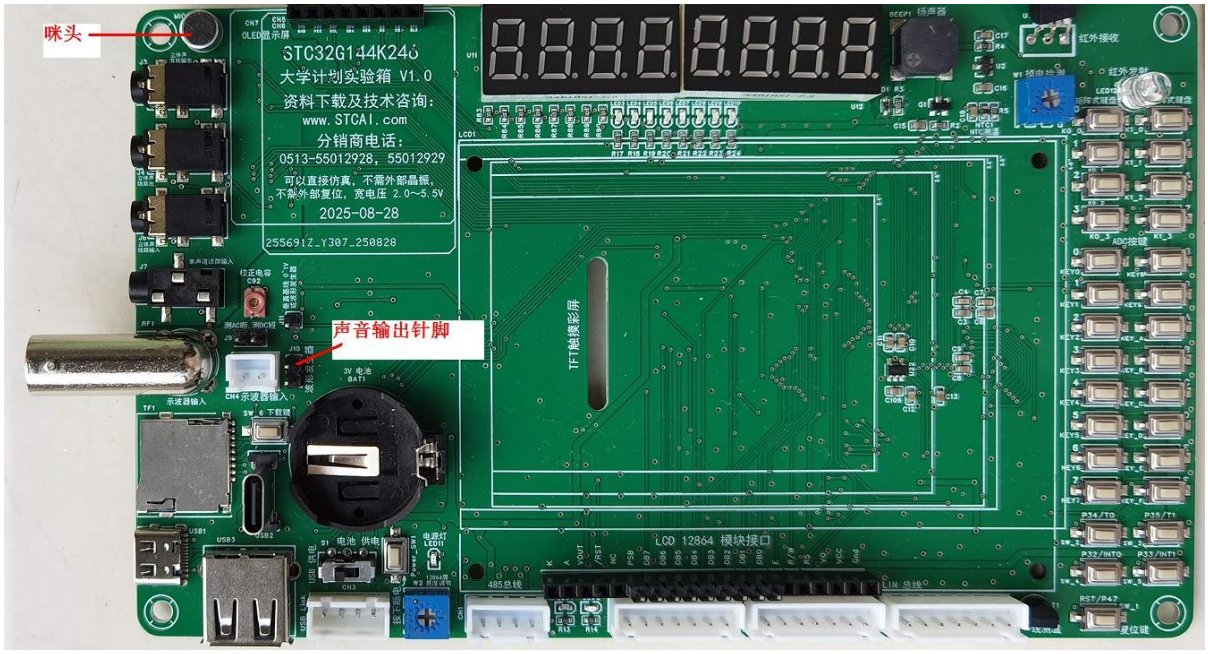
为了应用简单，本例将话筒 ADC 后的数据使用 A 率压缩，然后放在 SRAM 中的循环队列，可以缓冲 8 秒的声音（8K 采样），声音延时 8 秒输出。实测效果非常好。

用户设计电路时，AVREF 一定要接低噪声参考电压源，推荐便宜的 TL431 这个 2.5V 做基准即可，RMB0.08。

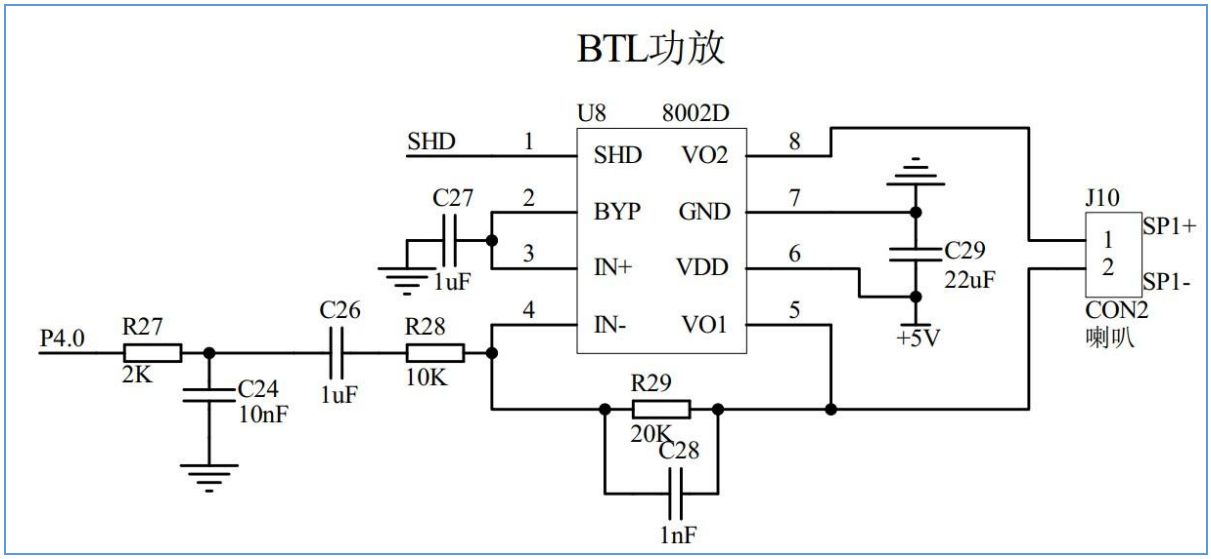
实验箱 用到的电路：



STC32G144K246 实验箱实物图：



推荐 ＜RMB0.1 的，简易但电源低压 BTL 功放电路：



【源代码】：

```
#include  "STC32G144K246.h"
#include  "intrins.h"
//STC32G144K246 LQFP100 学习板  V1.0  验证，使用板载咪头(ADC2-10)做语音输入，从 DAC2+OP4(P4.0)输出声音，低成本。
/*********************************
功能说明：
使用 STC32G144K246 LQFP100  学习板 V1.0 的咪头电路(ADC2-10)z 做语音输入、DAC2+OP4(P4.0)输出声音，实现实时、单声道的音频输入输出，不需要专门的音频 CODEC 芯片。
A 率的原始 PCM 采样信号位长就是 12bit 的，压缩成 8bit，符合国际电工委的协议标准。
使用 12 位的 DAC 输出，刚好能满足 A 率的动态范围，音质也很好。
咪头、话筒输入放大电路的噪声越低越好。
为了应用简单，本例将话筒 ADC 后的数据使用 A 率压缩，然后放在 SRAM 中的循环队列，可以缓冲 8
```

秒的声音（8K 采样），声音延时 8 秒输出。
实测效果非常好。
用户设计电路时，AVREF 一定要接低噪声参考电压源，推荐便宜的 TL431 这个 2.5V 做基准即可。
**********/

```
#define   HPLL_ENABLE             0         // 1:启动 PLL，0：关闭 PLL

#if (HPLL_ENABLE == 0)
    //不用 PLL
    #define FOSC              48000000UL              //定义主时钟
#else
    //使用 PLL
    #define FOSC              60000000UL              //定义主时钟,
    #define HPLL_CLK      480000000UL              //HPLL 时钟
    #define HPLL_SCALE                  40              //HPLL 时钟倍频系数 26~41，串口或
USB 下载时选择 48MHz,     HPLL 时钟  = 48MHz/8 *2 *40 = 480MHz
#endif
//支持的采样率：96000UL,80000UL,60000,48000,40000,32000,24000,20000,16000,15000,12000,10000,8000.
//        #define SampleRate                  96000UL         //定义采样率
//        #define SampleRate                  80000UL         //定义采样率
//        #define SampleRate                  60000           //定义采样率
//        #define SampleRate                  48000           //定义采样率
//        #define SampleRate                  40000           //定义采样率
//        #define SampleRate                  32000           //定义采样率
//        #define SampleRate                  24000           //定义采样率
//        #define SampleRate                  20000           //定义采样率
//        #define SampleRate                  16000           //定义采样率
//        #define SampleRate                  15000           //定义采样率
//        #define SampleRate                  12000           //定义采样率
//        #define SampleRate                  10000           //定义采样率
          #define SampleRate                   8000           //定义采样率

//        #define SampleRate                  88200UL         //定义采样率
//        #define SampleRate                  44100           //定义采样率
//        #define SampleRate                  22050           //定义采样率
//        #define SampleRate                  11025           //定义采样率

/*************   IO 口定义          ***************/
#define   ADC2_CHN   10             //设置 ADC 通道(咪头通道), 取值为 0~7, 对应 P0.0~P0.7,
                                    //8~10 对应 P6.0~P6.2，11-OP1O, 12-OP2O, 13-OP3O, 14-OP4O, 15-1.19V.

/*************   变量定义          ***************/
#define   VOICE_BUFF_LENGTH    64000
u16               wr_voice;
u8                xdata voice_buff[VOICE_BUFF_LENGTH];
u16               dac;
/*************   函数声明          ***************/
```

```
void    delay_ms(u16 ms);
u8              Timer0_Config(u8 t, u32 reload);        //t=0: reload 值是主时钟周期数,  t=1: reload 值是时间(单位 us)
void            ADC2_config(void);
void            DAC_config(void);

void main(void)
{
    u16  i;

    EAXFR = 1;                  //SFR enable
    CKCON = 0;
    #if (FOSC <= 64000000UL)
    WTST    = 1;                // 64MHz 以下写 1
    #elif (FOSC <= 90000000UL)
    WTST    = 2;                // 64~90MHz 写 2
    #elif (FOSC <= 120000000UL)
    WTST    = 3;                // 90~120MHz 写 3
    #else
    WTST    = 4;                // 120MHz 以上写 1
    #endif

#ifdef HPLL_SCALE
//============ 使用 HPLL 时钟 @48MHz 初始化 USB 后再初始化 PLL ==============
        HPLLCR = (0<<5) + (HPLL_SCALE-26);
/*(2<<5):HPLL1, (0<<5):HPLL1/2,   HPLL = HPLL_IN*2*(HPLLCR+26), HPLL_IN 必须为 6MHz, 设置 HPLL 的除频系数 HPLLCR = N-26, N=26~41, F_HPLL = PLL 输入时钟*2 * N,  比如 Fosc=48MHz, 8 分频,  N=40 时  F_HPLL = 48M/8*2*40=480M*/
//      HPLLCR &= ~0x10;        //选择 HPLL 输入时钟源为 HIRC, 默认
//      HPLLCR |= 0x10;         //选择 HPLL 输入时钟源为 IRCM
    HPLLPDIV = 8;               //设置 HPLL 输入时钟预分频为 8@48MHz（HPLL 输入频率必须为 6MHz）
    HPLLCR |= 0x80;             //使能 HPLL
    CLKDIV = (u8)(HPLL_CLK/2/FOSC);             //系统时钟=HPLL/2/N
    CLKSEL |= 0x04;             //选择 HPLL/2 作为主时钟源
    #endif

    HIRCCR = (HIRCCR & ~0x30) | (1 <<4);
/* (HIRCCR:0xFE02)的 IRC20MCR[5:4]为主时钟分频输出的时钟源选择位，0:关闭输出，1:输出系统时钟，2:输出内部高速 HIRC 时钟，3:输出滤波后的内部高速 HIRC 时钟*/
    MCLKO16_DIV(100);  P1n_push_pull(Pin6);
//      MCLKO54_DIV(100);       P5n_push_pull(Pin4);
    P54 = 0;        P5n_push_pull(Pin4);        //关闭蜂鸣器

    DAC_config();
    for(i=0; i<2048; i+=2)   //1 秒时间  DAC Ramp-up/Ramp down  消除 POP 噪声
```

```
    {
        delay_ms(1);
    //  DAC1_DAT = i;    DAC1_CR    = 0x41;        //DAC1 Ramp-up/Ramp down  消除 POP 噪声
        DAC2_DAT = i;    DAC2_CR    = 0x41;        //DAC2 Ramp-up/Ramp down  消除 POP 噪声
    }

    for(wr_voice=0; wr_voice<VOICE_BUFF_LENGTH; wr_voice++)        voice_buff[wr_voice] = 0x55;
    //初始化全 0^0x55
    wr_voice = 0;

    ADC2_config();
    Timer0_Config(0, FOSC/SampleRate);        // (u8 t, u32 reload);
    //t=0: reload 值是主时钟周期数,  t=1: reload 值是时间(单位 us), 返回 0 正确, 返回 1 装载值过大错误.
    EA = 1;

    P2n_push_pull(Pin1+Pin0);

    while(1)
    {
        NOP(10);
    }
}

//========================================================================
// 函数: void    delay_ms(u16 ms)
// 描述: 延时函数。
// 参数: ms,要延时的 ms 数, 1~65535ms. 自动适应主时钟.
// 返回: none.
// 备注:
//========================================================================
void    delay_ms(u16 ms)
{
    u16 i;
    do
    {
        i = FOSC / 6000;
        while(--i);
    }while(--ms);
}
//========================================================================
/********** A 率压缩/解压缩算法 ***************************
根据国际标准，A 率是以 12 位 AD，A=87.6 为基准的，13 折线法逼近。
本程序是 PC 声音输出，是 16 位的 DA，所以计算时，所有数据都*8。

law_data      b7      b6 b5 b4      b3 b2 b1 b0
              minus   Exponent      Mantissa
```

```
**************************************************************/
/*************** A 率压缩 ******************
    将 12bit AD 线性二进制值数据压缩成 8bit A-law 数据并存入缓冲
    Convert 12 Bit Liner Data to 8bit ALAW data
*******************************************************/
//#include"ALAW_encode.h"
const u8 T_Alaw_encode[2049]={
0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0A,0x0B,0x0C,0x0D,0x0E,0x0F,
0x10,0x11,0x12,0x13,0x14,0x15,0x16,0x17,0x18,0x19,0x1A,0x1B,0x1C,0x1D,0x1E,0x1F,
0x20,0x20,0x21,0x21,0x22,0x22,0x23,0x23,0x24,0x24,0x25,0x25,0x26,0x26,0x27,0x27,
0x28,0x28,0x29,0x29,0x2A,0x2A,0x2B,0x2B,0x2C,0x2C,0x2D,0x2D,0x2E,0x2E,0x2F,0x2F,
0x30,0x30,0x30,0x30,0x31,0x31,0x31,0x31,0x32,0x32,0x32,0x32,0x33,0x33,0x33,0x33,
0x34,0x34,0x34,0x34,0x35,0x35,0x35,0x35,0x36,0x36,0x36,0x36,0x37,0x37,0x37,0x37,
0x38,0x38,0x38,0x38,0x39,0x39,0x39,0x39,0x3A,0x3A,0x3A,0x3A,0x3B,0x3B,0x3B,0x3B,
0x3C,0x3C,0x3C,0x3C,0x3D,0x3D,0x3D,0x3D,0x3E,0x3E,0x3E,0x3E,0x3F,0x3F,0x3F,0x3F,
0x40,0x40,0x40,0x40,0x40,0x40,0x40,0x40,0x41,0x41,0x41,0x41,0x41,0x41,0x41,0x41,
0x42,0x42,0x42,0x42,0x42,0x42,0x42,0x42,0x43,0x43,0x43,0x43,0x43,0x43,0x43,0x43,
0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x44,0x45,0x45,0x45,0x45,0x45,0x45,0x45,0x45,
0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x46,0x47,0x47,0x47,0x47,0x47,0x47,0x47,0x47,
0x48,0x48,0x48,0x48,0x48,0x48,0x48,0x48,0x49,0x49,0x49,0x49,0x49,0x49,0x49,0x49,
0x4A,0x4A,0x4A,0x4A,0x4A,0x4A,0x4A,0x4A,0x4B,0x4B,0x4B,0x4B,0x4B,0x4B,0x4B,0x4B,
0x4C,0x4C,0x4C,0x4C,0x4C,0x4C,0x4C,0x4C,0x4D,0x4D,0x4D,0x4D,0x4D,0x4D,0x4D,0x4D,
0x4E,0x4E,0x4E,0x4E,0x4E,0x4E,0x4E,0x4E,0x4F,0x4F,0x4F,0x4F,0x4F,0x4F,0x4F,0x4F,
0x50,0x50,0x50,0x50,0x50,0x50,0x50,0x50,0x50,0x50,0x50,0x50,0x50,0x50,0x50,0x50,
0x51,0x51,0x51,0x51,0x51,0x51,0x51,0x51,0x51,0x51,0x51,0x51,0x51,0x51,0x51,0x51,
0x52,0x52,0x52,0x52,0x52,0x52,0x52,0x52,0x52,0x52,0x52,0x52,0x52,0x52,0x52,0x52,
0x53,0x53,0x53,0x53,0x53,0x53,0x53,0x53,0x53,0x53,0x53,0x53,0x53,0x53,0x53,0x53,
0x54,0x54,0x54,0x54,0x54,0x54,0x54,0x54,0x54,0x54,0x54,0x54,0x54,0x54,0x54,0x54,
0x55,0x55,0x55,0x55,0x55,0x55,0x55,0x55,0x55,0x55,0x55,0x55,0x55,0x55,0x55,0x55,
0x56,0x56,0x56,0x56,0x56,0x56,0x56,0x56,0x56,0x56,0x56,0x56,0x56,0x56,0x56,0x56,
0x57,0x57,0x57,0x57,0x57,0x57,0x57,0x57,0x57,0x57,0x57,0x57,0x57,0x57,0x57,0x57,
0x58,0x58,0x58,0x58,0x58,0x58,0x58,0x58,0x58,0x58,0x58,0x58,0x58,0x58,0x58,0x58,
0x59,0x59,0x59,0x59,0x59,0x59,0x59,0x59,0x59,0x59,0x59,0x59,0x59,0x59,0x59,0x59,
0x5A,0x5A,0x5A,0x5A,0x5A,0x5A,0x5A,0x5A,0x5A,0x5A,0x5A,0x5A,0x5A,0x5A,0x5A,0x5A,
0x5B,0x5B,0x5B,0x5B,0x5B,0x5B,0x5B,0x5B,0x5B,0x5B,0x5B,0x5B,0x5B,0x5B,0x5B,0x5B,
0x5C,0x5C,0x5C,0x5C,0x5C,0x5C,0x5C,0x5C,0x5C,0x5C,0x5C,0x5C,0x5C,0x5C,0x5C,0x5C,
0x5D,0x5D,0x5D,0x5D,0x5D,0x5D,0x5D,0x5D,0x5D,0x5D,0x5D,0x5D,0x5D,0x5D,0x5D,0x5D,
0x5E,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E,0x5E,
0x5F,0x5F,0x5F,0x5F,0x5F,0x5F,0x5F,0x5F,0x5F,0x5F,0x5F,0x5F,0x5F,0x5F,0x5F,0x5F,
0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,
0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,0x60,
0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,
0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,0x61,
0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,
0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,0x62,
0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,
```

0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,0x63,
0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,
0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,0x64,
0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,
0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,0x65,
0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,
0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,0x66,
0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,
0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,0x67,
0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,
0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,0x68,
0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,
0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,0x69,
0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,
0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,0x6A,
0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,
0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,0x6B,
0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,
0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,0x6C,
0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,
0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,0x6D,
0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,
0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,0x6E,
0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,
0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,0x6F,
0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,
0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,
0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,
0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,0x70,
0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,
0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,
0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,
0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,0x71,
0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,
0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,
0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,
0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,0x72,
0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,
0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,
0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,
0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,0x73,
0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,
0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,
0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,
0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,0x74,
0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,

0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,
0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,
0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,0x75,
0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,
0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,
0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,
0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,0x76,
0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,
0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,
0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,
0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,0x77,
0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,
0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,
0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,
0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,0x78,
0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,
0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,
0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,
0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,0x79,
0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,
0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,
0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,
0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,0x7A,
0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,
0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,
0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,
0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,0x7B,
0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,
0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,
0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,
0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,0x7C,
0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,
0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,
0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,
0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,0x7D,
0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,
0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,
0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,
0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,0x7E,
0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,
0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,
0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,
0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F,0x7F};

```
//****************** A 率压缩、解压缩 ********************
//   将 8bit A-law 数据解压缩成 12bit 数据并输出到 DAC
//          Convert 8bit Alaw data to 12 Bit Liner Data
```

```
//*************************************************
const	u16 T_Alaw_decode[128]={
0x0000,0x0001,0x0002,0x0003,0x0004,0x0005,0x0006,0x0007,0x0008,0x0009,0x000A,0x000B,0x000C,0x000D,0x000E,0x000F,
0x0010,0x0011,0x0012,0x0013,0x0014,0x0015,0x0016,0x0017,0x0018,0x0019,0x001A,0x001B,0x001C,0x001D,0x001E,0x001F,
0x0020,0x0022,0x0024,0x0026,0x0028,0x002A,0x002C,0x002E,0x0030,0x0032,0x0034,0x0036,0x0038,0x003A,0x003C,0x003E,
0x0040,0x0044,0x0048,0x004C,0x0050,0x0054,0x0058,0x005C,0x0060,0x0064,0x0068,0x006C,0x0070,0x0074,0x0078,0x007C,
0x0080,0x0088,0x0090,0x0098,0x00A0,0x00A8,0x00B0,0x00B8,0x00C0,0x00C8,0x00D0,0x00D8,0x00E0,0x00E8,0x00F0,0x00F8,
0x0100,0x0110,0x0120,0x0130,0x0140,0x0150,0x0160,0x0170,0x0180,0x0190,0x01A0,0x01B0,0x01C0,0x01D0,0x01E0,0x01F0,
0x0200,0x0220,0x0240,0x0260,0x0280,0x02A0,0x02C0,0x02E0,0x0300,0x0320,0x0340,0x0360,0x0380,0x03A0,0x03C0,0x03E0,
0x0400,0x0440,0x0480,0x04C0,0x0500,0x0540,0x0580,0x05C0,0x0600,0x0640,0x0680,0x06C0,0x0700,0x0740,0x0780,0x07C0};

#define	HighPassEnable	1			// 1: 允许高通滤波，消除直流偏移, 0: 禁止高通滤波

	//		32741, -32741, 0, -32715, 0			//1 阶巴特沃斯高通	8Hz/32000Hz
#define	COEF_A0			32741
#define	COEF_B0			-32715
#if (HighPassEnable ==1)
long	Xin1, Xin0, Yout0, out;
#endif

void Alaw_encode_decode(u16 adc)	//输入 12bit 无符号的 ADC 值		//编解码一起, 先解码，再编码
{
	u8	law_data;
	u8	minus;
//====================== 先解压缩 ==========================
	law_data = voice_buff[wr_voice] ^ 0x55;	//Get Complement 偶数位取反, 从缓冲读数据字节，
						//偶数位取反，循环队列，指向下一个数据
	dac = T_Alaw_decode[law_data & 0x7f];	// 解压缩码
	if(law_data & 0x80)	dac += 2048;		//正数，移到中点以上 dac+2048
	else			dac = 2048-dac;		//负数，移到中点以下 2048-dac
//====================== 再压缩 ==========================
	#if (HighPassEnable ==1)		//允许高通滤波，消除直流偏移
		Xin1 = Xin0;	Xin0 = (long)adc - 2048;	//输入 ADC 值为 12 位, 转为双极性
		out = (COEF_A0 * (Xin0 - Xin1) - COEF_B0 * Yout0)/32768;	//一阶高通
		Yout0 = out;
		if(out >=  2048)	out =  2047;	//输出限幅
		if(out <= -2048)	out = -2047;	//输出限幅
		adc = (u16)(out+2048);
```

```
    #endif
    if(adc & 0x0800)        minus = 0x80,       adc = adc - 2048;   // >= 2048 为正, 0~2047, 以中间值 2048 为 0 点.
    else                                minus = 0x00,       adc = 2048 - adc;   //   < 2048 为负. 1~2048
    voice_buff[wr_voice] = (T_Alaw_encode[adc] + minus) ^ 0x55; //数据写入缓冲，偶数位取反
                                                                //循环队列，指向下一个数据
    if(++wr_voice >= VOICE_BUFF_LENGTH)         wr_voice = 0;       //下一个采样
}

#define   PGA1_PSEL             2       //[Bit1  Bit0]   正输入端选择   0:P0.4, 1:P0.5, 2:P6.0, 3:DAC1O
#define   PGA1_NSEL             2       //[Bit3  Bit2]   负输入端选择   0:P0.5, 1:P0.4, 2:P6.1, 3:DAC1O
#define   PGA1_OSEL             3       //[Bit5 Bit4]   输出端选择     0:P0.7, 1:P0.0, 2:P7.4, 3:P6.3
#define   PGA1_MSEL             0       //[Bit7  Bit6]   模式选择       0:non,  1:Buffer, 2:Invert, 3:NonInvert
#define   PGA1_PWD              0       //[Bit0]        掉电控制       0:电源打开，  1：电源关闭
#define   PGA1_IBS              0       //[Bit1]        偏置电流选择   0:偏置电流小  1：偏置电流大
#define   PGA1_OE               1       //[Bit2]                       1:使能放大器外扩接口
#define   PGA1_GSEL             0       //[Bit5~Bit3]   增益选择
                        //反相增益 0->断开负输入, 1->1、2->2、3->4、4->8、5->16、6->32、7->48，
                        //同相增益 0->1、1->2、2->3、3->5、4->9、5->17、6->33、7->49

#define   PGA2_PSEL             3       //[Bit1 Bit0]   正输入端选择   0:P2.6, 1:P8.0, 2:P7.5, 3:P7.7
#define   PGA2_NSEL             0       //[Bit3  Bit2]   负输入端选择   0:P2.7, 1:P8.1, 2:P7.6, 3:DAC2O
#define   PGA2_OSEL             2       //[Bit5 Bit4]   输出端选择     0:P4.2, 1:P0.0, 2:P7.4, 3:P6.2
#define   PGA2_MSEL             1       //[Bit7  Bit6]   模式选择       0:non,  1:Buffer, 2:Invert, 3:NonInvert
#define   PGA2_PWD              0       //[Bit0]        掉电控制       0:电源打开，  1：电源关闭
#define   PGA2_IBS              0       //[Bit1]        偏置电流选择   0:偏置电流小  1：偏置电流大
#define   PGA2_OE               1       //[Bit2]                       1:使能放大器外扩接口
#define   PGA2_GSEL             0       //[Bit5~Bit3]   增益选择
                        //反相增益 0->断开负输入, 1->1、2->2、3->4、4->8、5->16、6->32、7->48，
                        //同相增益 0->1、1->2、2->3、3->5、4->9、5->17、6->33、7->49

#define   PGA3_PSEL             3       //[Bit1  Bit0]   正输入端选择   0:P3.5, 1:P2.0, 2:P7.0, 3:DAC2O
#define   PGA3_NSEL             0       //[Bit3  Bit2]   负输入端选择   0:P5.1, 1:P4.2, 2:P7.1, 3:DAC2O
#define   PGA3_OSEL             3       //[Bit5 Bit4]   输出端选择     0:P3.2, 1:P2.1, 2:P7.4, 3:P5.0
#define   PGA3_MSEL             1       //[Bit7  Bit6]   模式选择       0:non,  1:Buffer, 2:Invert, 3:NonInvert
```

```
#define   PGA3_PWD          0       //[Bit0]      掉电控制        0:电源打开，  1：电源关闭
#define   PGA3_IBS          0       //[Bit1]      偏置电流选择    0:偏置电流小  1：偏置电流大
#define   PGA3_OE           1       //[Bit2]                      1:使能放大器外扩接口
#define   PGA3_GSEL         1       //[Bit5~Bit3]   增益选择
                          //反相增益 0->断开负输入, 1->1、2->2、3->4、4->8、5->16、6->32、7->48，
                          //同相增益 0->1、1->2、2->3、3->5、4->9、5->17、6->33、7->49

#define   PGA4_PSEL         3       //[Bit1  Bit0]   正输入端选择   0:P1.6, 1:P6.3, 2:P6.4, 3:DAC2O
#define   PGA4_NSEL         0       //[Bit3  Bit2]   负输入端选择   0:P1.7, 1:P6.5, 2:P3.3, 3:DAC1O
#define   PGA4_OSEL         1       //[Bit5 Bit4]   输出端选择     0:P1.3, 1:P4.0, 2:P7.4, 3:P6.3
#define   PGA4_MSEL         1       //[Bit7  Bit6]   模式选择       0:non,  1:Buffer, 2:Invert, 3:NonInvert
#define   PGA4_PWD          0       //[Bit0]      掉电控制        0:电源打开，  1：电源关闭
#define   PGA4_IBS          0       //[Bit1]      偏置电流选择    0:偏置电流小  1：偏置电流大
#define   PGA4_OE           1       //[Bit2]                      1:使能放大器外扩接口
#define   PGA4_GSEL         0       //[Bit5~Bit3]   增益选择
                          //反相增益 0->断开负输入, 1->1、2->2、3->4、4->8、5->16、6->32、7->48，
                          //同相增益 0->1、1->2、2->3、3->5、4->9、5->17、6->33、7->49

void    DAC_config(void)
{
//      DAC1_DIV = 2;
/*DAC 时钟大约 200KHz, = MCLK/(DAC1_DIV*4), DAC1_DIV = 0 或 1 均为 1。实测写 1 都可以正常操作 DAC，DAC1+OP1_BUffer 输出 32 点正弦波，采样 2.816MHz@48MHz，正弦波 88KHz，幅度正常，速度响应正常。*/
    DAC2_DIV = 2;
/*DAC 时钟大约 200KHz, = MCLK/(DAC1_DIV*4), DAC1_DIV = 0 或 1 均为 1。实测写 1 都可以正常操作 DAC，DAC1+OP1_BUffer 输出 32 点正弦波，采样 2.816MHz@48MHz，正弦波 88KHz，幅度正常，速度响应正常。*/

    PGA1_CR1 = (PGA1_MSEL<<6) | (PGA1_OSEL<<4) | (PGA1_NSEL<<2) | PGA1_PSEL;
    PGA1_CR2 = (PGA1_GSEL<<4) | (PGA1_OE<<2)   | (PGA1_IBS<<1)  | PGA1_PWD;
    PGA2_CR1 = (PGA2_MSEL<<6) | (PGA2_OSEL<<4) | (PGA2_NSEL<<2) | PGA2_PSEL;
    PGA2_CR2 = (PGA2_GSEL<<4) | (PGA2_OE<<2)   | (PGA2_IBS<<1)  | PGA2_PWD;
//      PGA3_CR1 = (PGA3_MSEL<<6) | (PGA3_OSEL<<4) | (PGA3_NSEL<<2) | PGA3_PSEL;
//      PGA3_CR2 = (PGA3_GSEL<<4) | (PGA3_OE<<2)   | (PGA3_IBS<<1)  | PGA3_PWD;
    PGA4_CR1 = (PGA4_MSEL<<6) | (PGA4_OSEL<<4) | (PGA4_NSEL<<2) | PGA4_PSEL;
    PGA4_CR2 = (PGA4_GSEL<<4) | (PGA4_OE<<2)   | (PGA4_IBS<<1)  | PGA4_PWD;

    P6n_HighZ(0x0f);
    P7n_HighZ(Pin7+Pin6+Pin4);
    P2n_HighZ(Pin7);
```

```
    P1n_HighZ(Pin7);
    P4n_HighZ(Pin0);

//      DAC1_DAT = 0;   DAC1_CR    = 0x41;      //DAC1 输出 0V @VREF=2.5V
    DAC2_DAT = 0;       DAC2_CR    = 0x41;      //DAC2 输出 0V @VREF=2.5V

}

//========================================================================
// 函数: void      ADC2_config(void)
// 描述: ADC2 初始化函数
// 参数: none.
// 返回: none.
// 备注:
//========================================================================
//ADC2_CONTR ADC2 控制寄存器
#define ADC2_POWER      (1<<7)  /* ADC2 电源，1 开启，0 关闭 */
#define ADC2_START      (1<<6)  /* 启动转换，自动清 0 */
#define ADC2_FLAG       (1<<5)  /* 完成标志，软件清 0 */
#define ADC2_EPWMT      (0<<4)  /* 允许 PWMA 触发 ADC */
#define ADC2_CHS            0   /* ADC2 通道 取值为 0~7, 对应 P0.0~P0.7, 8~10 对应 P6.0~P6.2，
11-OP1O, 12-OP2O, 13-OP3O, 14-OP4O, 15-1.19V */
//ADC2CFG ADC2 配置寄存器
#define ADC2_RES_FMT (1<<5)  /* ADC2 结果格式 0: 左对齐, ADC_RES: D11 D10 D9 D8 D7 D6
D5 D4, ADC_RESL: D3 D2 D1 D0  0  0  0  0 */
                                                    /*          1: 右对齐,
ADC_RES:  0   0   0   0   D11 D10 D9 D8, ADC_RESL: D7 D6 D5 D4 D3 D2 D1 D0 */
#define ADC2_SPEED          7       /* 0~15, ADC 时钟 = SYSclk/2/(n+1) */
//ADC2TIM  ADC2 时序控制寄存器
#define ADC2_CSSETUP    (1<<7)  /* 0~1,  ADC2 通道选择时间      0: 1 个 ADC 时钟, 1: 2 个 ADC
时钟,  默认 0(默认 1 个 ADC 时钟) */
#define ADC2_CSHOLD     (1<<5)  /* 0~3,  ADC2 通道选择保持时间  (n+1)个 ADC 时钟, 默认 1(默认
2 个 ADC 时钟)                                  */
#define ADC2_SMPDUTY    10      /* 10~31, ADC2 模拟信号采样时间  (n+1)个 ADC 时钟,
默认 10(默认 11 个 ADC 时钟)                                    */
                                                    /* ADC2 转换时间: 12 位 ADC
固定为 12 个 ADC 时钟.          */
//ADC2EXCFG  ADC2 扩展配置寄存器
#define ADC2_ETR_PS         (0<<6)  /* 0~3, ADC2 外部触发脚选择,   0:P1.2, 1:P4.3, 其余保留
*/
#define ADC2_ETRS           (0<<4)  /* 0~3, ADC2 外部触发脚控制位, 2:允许 ADC2_ETR 的上
升沿触发 ADC2, 3:下降沿触发 ADC2，其余禁止 ETR 功能 */
#define ADC2_CVTIMESEL      0       /* 0~7, ADC2 自动转换次数选择(非 DMA 模式), 0~3:转换 1
次，4:转换 2 次取平均值，5:转换 4 次取平均值，6:转换 8 次取平均值，7:转换 16 次取平均值 */
//ADC2EXCFG2  ADC2 扩展配置寄存器
#define ADC2_TRIGON         (0<<7)  /* 两组 ADC 同步触发控制, 0: 两组 ADC 各自独立工作，
```

```
1：两组 ADC 位同步模式 */
#define ADC2_PWMSEL             (0<<4)  /* 0~3,  PWM 触发信号配置 */
#define ADC2_TRIGSEL            0       /* 0~15, 选择 ADC2 触发源, 0:PWMA_TA, 1:PWMC_TA, 2:PWME_TA, 3:PWMA_TA2, 4:PWMC_TA2, 5:PWME_TA2, 6:PWMB_TA, 7:PWMD_TA, 8:PWMF_TA, 9:1, 10:0, 11:PWMA_TA, 12:TIMER17 OV, 13:TIMER18 OV, 14:TIMER17 OV phase 180, 15:TIMER18 OV phase 180 */
//ADC2INTR  ADC2 中断控制寄存器
#define PADC2                   (3<<1)  /* 0~3,  ADC2 中断优先级，0 最低，3 最高 */
#define EADC2                   0       /* ADC2 中断使能，0:禁止中断，1:允许中断 */

void    ADC2_config(void)
{
//      EAXFR = 1;      //SFR enable
/*ADC2_CHN: ADC 通道号,  取值为 0~7, 对应 P0.0~P0.7, 8~10 对应 P6.0~P6.2，11-OP1O, 12-OP2O, 13-OP3O, 14-OP4O, 15-1.19V. */
    #if (ADC2_CHN <= 7)
    P0n_HighZ(1<<ADC2_CHN);             //设置要做 ADC 的 IO 做高阻输入
    #elif (ADC2_CHN <= 10)
    P6n_HighZ(1<<(ADC2_CHN-8));         //设置要做 ADC 的 IO 做高阻输入
    #endif
    ADC2_CONTR = ADC2_POWER   | ADC2_CHN;           //ADC on + channel
    ADC2CFG    = ADC2_RES_FMT | ADC2_SPEED;
    ADC2TIM    = ADC2_CSSETUP | ADC2_CSHOLD | ADC2_SMPDUTY;
    ADC2EXCFG  = ADC2_ETR_PS  | ADC2_ETRS    | ADC2_CVTIMESEL;
    ADC2EXCFG2 = ADC2_TRIGON  | ADC2_PWMSEL | ADC2_TRIGSEL;
    ADC2INTR   = PADC2 | EADC2;
}
//========================================================================
// 函数: u8         Timer0_Config(u8 t, u32 reload)
// 描述: timer0 初始化函数.
// 参数:        t: 重装值类型, 0 表示重装的是系统时钟数, 其余值表示重装的是时间(us).
//      reload: 重装值.
// 返回: 0: 初始化正确,  1: 重装值过大, 初始化错误.
//========================================================================
u8      Timer0_Config(u8 t, u32 reload)     //t=0: reload 值是主时钟周期数,  t=1: reload 值是时间(单位 us)
{
    u16  i;

    Timer0_Stop();                                  //停止定时器 0
    if(t != 0)      reload = (u32)(((float)FOSC * (float)reload)/1000000UL);
                                    //t!=0, 重装的是时间(us), 计算所需要的系统时钟数.
    if(reload >= (65536UL * 12 *256))           return 1;  //值过大, 返回错误
    if(reload >= (65536UL *256))    //12T 模式
    {
        Timer0_12T();                               //12T 模式
```

```
        reload /= 12;
    }
    else  Timer0_1T();                          //1T 模式
    i = (u16)(reload >> 16);                    //预分频值，最大 255.
    TM0PS = (u8)i;                              //设置定时器时钟预分频
    i = (u16)(65536UL - reload / (i+1));

    TH0 = (u8)(i /256);                         //Timer0 重装值
    TL0 = (u8)(i %256);                         //TImer0 重装值
    Timer0_AsTimer();                           //定时器模式       Timer0_AsTimer();
        Timer0_AsCounterP34();
    Timer0_16bitAutoReload();                   //定时器模式
    Timer0_InterruptEnable();                   //允许中断
//      PT0 = 1;                                    //高优先级中断
    Timer0_Run();                               //开始运行

//      Timer0_CLKO_Enable_P35();           //允许输出时钟
//      P3n_push_pull(Pin5);                        //P3.5 设置为准双向口
    return 0;
}

/********************** Timer0 1ms 中断函数  ************************/
void Timer0_ISR (void) interrupt TMR0_VECTOR
{
    P20 = 1;                                                    //编解码时间指示

    DAC2_DAT = dac;       DAC2_CR = 0x41; //刷新 DAC2 输出

    Alaw_encode_decode((u16)ADC2_RES * 256 + (u16)ADC2_RESL);     //输入 12bit 无符号的 ADC 值
    ADC2_CONTR = ADC2_POWER | ADC2_START | ADC2_CHN;                    //触发 ADC
    P20 = 0;
}
```

23.5.3 STC32G144K246，DAC-DMA 例程及注意事项

STC32G144K246，DAC-DMA 输出，离散信号输出，足够快成连续信号 ！

需要注意以下问题：（点击到“技术交流及选型论坛”相关链接）

1.DMA_DAC1_AMT 的**单位是 16bit** 的，例如想要 DMA 传输 50 个 16 位的数据(数据最大不能超过 12 位)，则 DMA_DAC1_AMT 需要填入 49(DMA_DAC1_AMT 填 0 时为传输 1 字节，所以填入时需要减 1)

2.DAC-DMA 传输时，DAC1_DIV 是和 **DMA_DAC1_ITV 共同作用**的

3.传入 DAC 的 **DMA 数据，需要转换为小端模式**，这里提供一个简单的函数，可以进行转换：

```
//大小端交换，传入数组地址和所需转换数量（以 16bit 为单位）
void swap_endian_uint16_array(unsigned int *arr, unsigned int count)
{
    unsigned int i;
    unsigned short temp;

    for(i = 0; i < count; i++)
    {
        temp = arr[i];
        arr[i] = ((temp & 0x00FF) << 8) | ((temp & 0xFF00) >> 8);
    }
}
```

4.需要使用 DAC-DMA 模式时，DAC1_CR 中，需要将**转换模式配置为连续转换模式**

DMA 只管更新 DMA_DAT，不会主动触发 DAC TG，如果配置为单次转换或者其他定时器触发，仍然需要定时器触发或者软件触发 DAC TG 才能看到波形。

否则，DAC 将不会更新输出。

21.2.2 DAC1 控制寄存器（DAC1_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DAC1_CR	7EF610H	DMAEN	DAC1TG	DAC1TS[1:0]		DAC1IP[1:0]		EDAC1I	DAC1EN

DMAEN：DMA模式控制位

0：禁止 DMA 模式

1：使能 DMA 模式

DAC1TG：触发DAC输出

0：无效

1：触发 DAC 输出，硬件自动清零

DAC1TS[1:0]：DAC1触发源选择

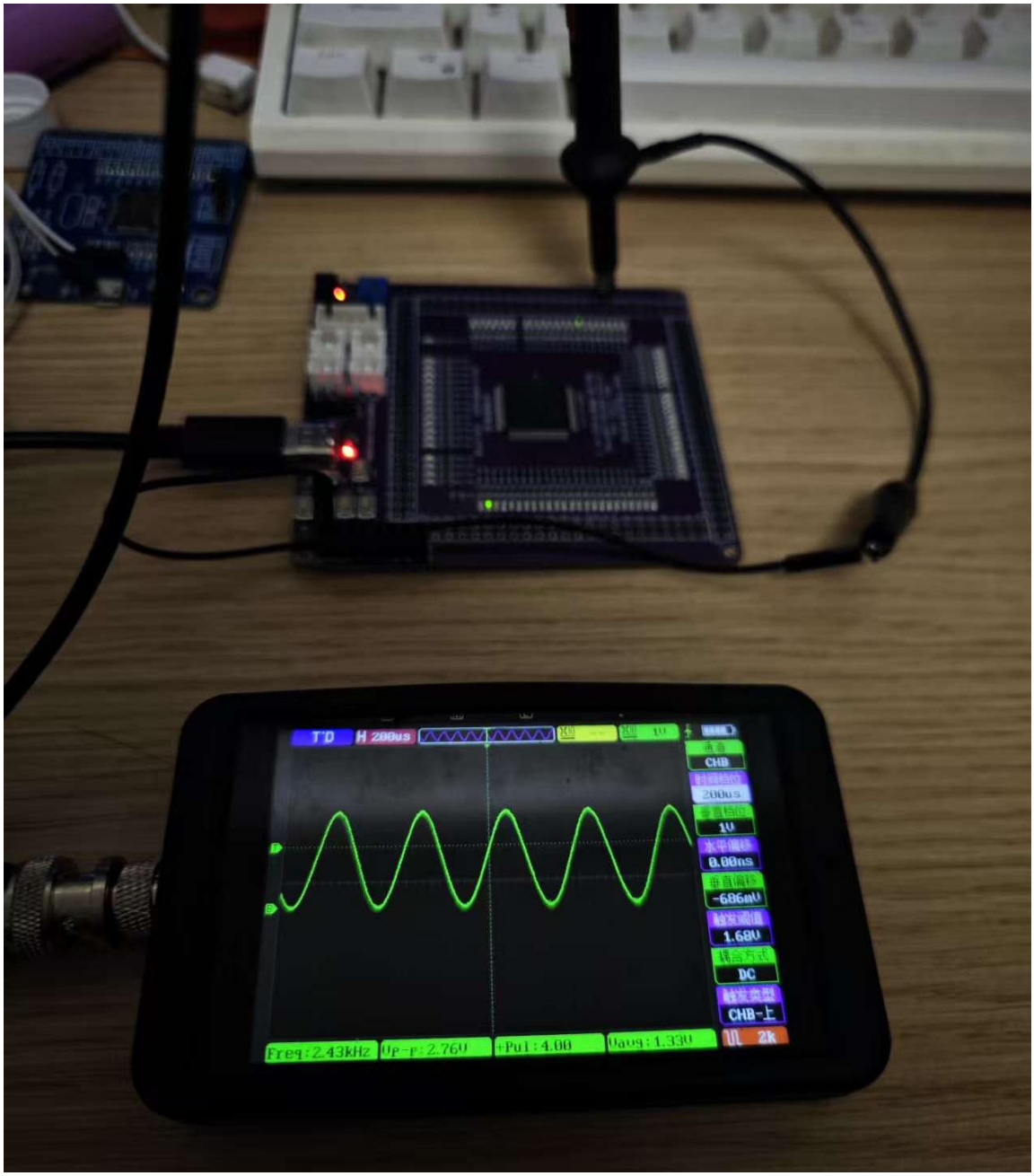
DAC1TS[1:0]	自动触发源	触发说明
00	单次触发	软件使用DAC1TG触发
01	定时器17自动触发	
10	定时器18自动触发	
11	连续触发模式	

5.**DMA_DAC1_DONE 寄存器，单位仍然为 8bit**，即一个字节，例如 DMA_DAC1_AMT 填入 49，则全部转换完成后，将可以在 DMA_DAC1_DONE 读到 100

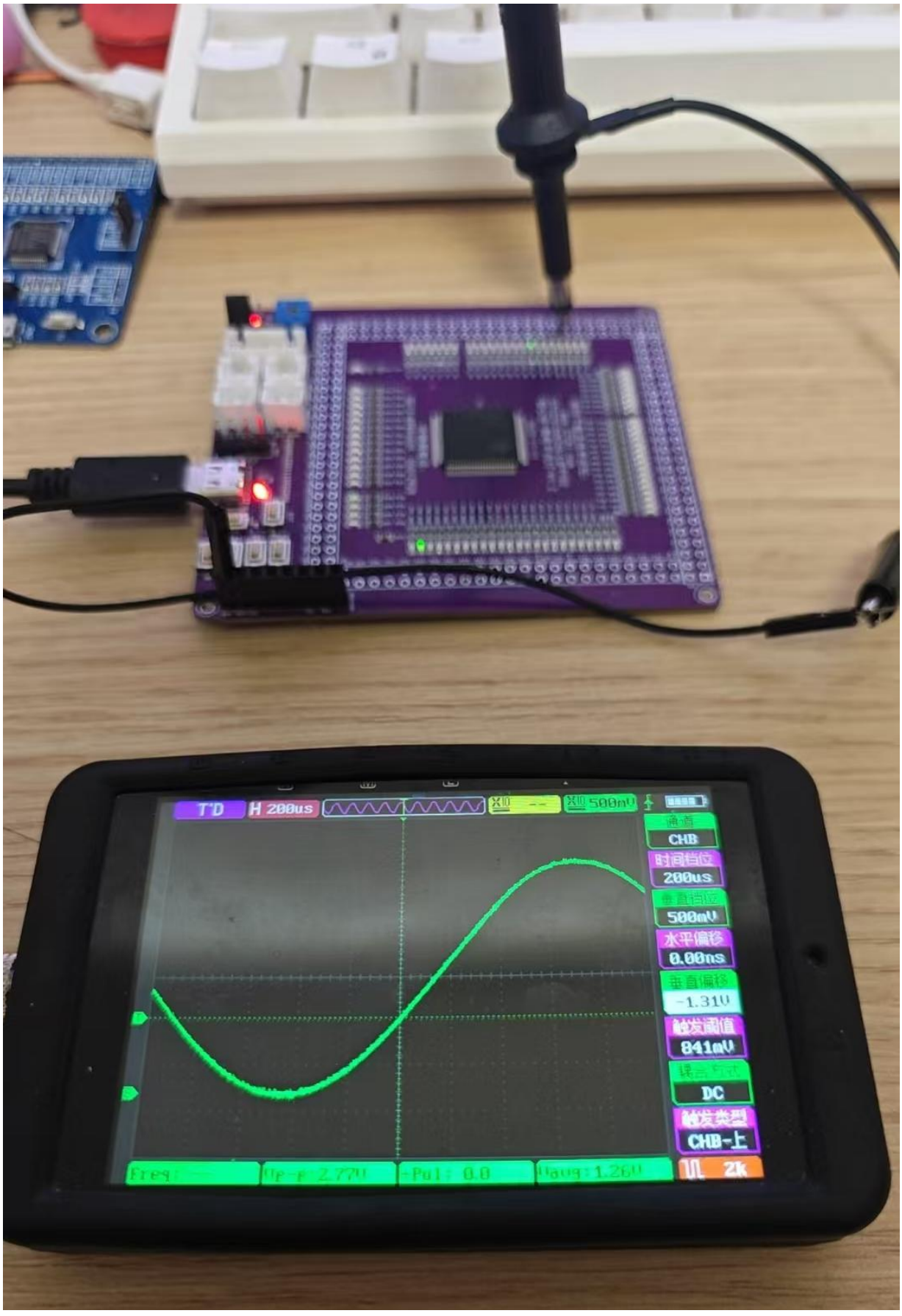
6.**DAC1_DIV 和 DAC1_DAT 为 16 位寄存器**，可以直接填入值，不需要拆分成 8 位填入，寄存器是否支持 16 位，请看头文件定义，是否为 int 类型

```
2159 #define      DAC1_DIVL          (*(unsigned char  volatile far *)0x7ef615)
2160 #define      DAC1_DAT           (*(unsigned int   volatile far *)0x7ef612)
2161 #define      DAC1_DIV           (*(unsigned int   volatile far *)0x7ef614)
2162
```

以下为在 STC32G144K246 核心板上验证通过的正弦波输出测试程序，上电自动在 P63 端口输出一个正弦波：
程序运行在 120Mhz，采用 4096 点满分辨率输出，无需滤波，输出速度可达 2.43KHz，无需 RC 滤波即可看到非常平滑的波形。
想要 DAC 看不到阶梯感，使用较高的分辨率即可，或加滤波/毕竟 DAC 输出是离散信号不是连续信号。



高分辨率下，放大后依然是看不出来阶梯感的



【源代码】：

```
#include "STC32G144K246.H"
#include "math.h"
//本例程使用 CHIPID 内预置参数，设置 HIRC 为 24MHz
//使用 HPLL1，提供 60Mhz，80Mhz，120Mhz 的设置例程
#define Fosc_60Mhz 0 //系统时钟为 60Mhz
#define Fosc_80Mhz 1 //系统时钟为 80Mhz
#define Fosc_120Mhz 2//系统时钟为 120Mhz

#define Main_Fosc Fosc_120Mhz //设置系统时钟为 120Mhz

//正弦波表，最大值 4095,使用计算方式在初始化的时候生成
unsigned int xdata SinTable_Software[4096];

void CLK_Init(void);                    //设置系统时钟，由 Main_Fosc 定义设置
void Io_Init(void);                     //I/O 口初始化函数，设置 P32 为开漏+打开内部上拉电阻模式
void DAC_Init(void);                    //DAC 部分初始化函数，DAC1 由 OPA1 的缓冲模式作为输出

//大小端交换，传入数组地址和所需转换数量（以 16bit 为单位）
void swap_endian_uint16_array(unsigned int *arr, unsigned int count)
{
    unsigned int i;
    unsigned short temp;

    for(i = 0; i < count; i++)
    {
        temp = arr[i];
        arr[i] = ((temp & 0x00FF) << 8) | ((temp & 0xFF00) >> 8);
    }
}

void GenerateSinTable(unsigned int *pArray, unsigned int points)
{
    unsigned int i;
    double angle, sinValue;

    for(i = 0; i < points; i++)
    {
        /* 生成 0 到 2π之间的角度 */
        angle = (2.0 * 3.14159265358979323846 * i) / points;

        /* 计算 sin 值 (-1 到 1) */
        sinValue = sin(angle);
```

```
            /* 映射到 0-4095 范围 */
            /* sinValue 从-1 到 1，先+1 变成 0 到 2，除以 2 变成 0 到 1，再乘 4095 */
            pArray[i] = (unsigned int)((sinValue + 1.0) * 2047.5 + 0.5);

            /* 限制在 0-4095 范围内 */
            if(pArray[i] > 4095)
            {
                pArray[i] = 4095;
            }
        }
}

void Delay10ms(void)        //@120MHz
{
    unsigned long edata i;

    _nop_();
    _nop_();
    i = 299998UL;
    while (i) i--;
}

int cnt = 0;
void main(void)
{
    EAXFR = 1;                                  //使能访问扩展 RAM 区特殊功能寄存器（XFR）
    CKCON &= ~0x07;                             //清空[2:0]，设置外部数据总线等待时钟为 0（最快），默认为 7
    CLK_Init();                                 //设置 HPLL 时钟为指定频率
    GenerateSinTable(SinTable_Software,4096);
    swap_endian_uint16_array(SinTable_Software,4096);
    Io_Init();                                  //初始化 I/O 口，设置 P32 等效为原准双向口模式（开漏模式+打开内部上拉电阻）
    DAC_Init();                                 //初始化 DAC
    EA = 1;                                     //打开总中断

    while(1)
    {
        //用户程序
        if(P32 == 0)
        {
            DAC1_STA = 0x00;                    //清空状态标志位
            DMA_DAC1_STA = 0x00;                //清空中断标志位
            DMA_DAC1_CR = 0xc0;                 //触发启动
            while(~P32);
        }
```

```
    }
}


void Io_Init(void)
{
    P3M0 |= 0x04; P3M1 |= 0x04;     //P32 设置为开漏输出
    P3PU |= 0x04;                                   //打开 P32 的上拉电阻
    //I/O 口设置为开漏输出+打开上拉电阻==原准双向口模式，写 1 可读外部电平
    P3M0 |= 0x20; P3M1 &= ~0x20;  //设置 P35 为推挽输出模式，用于观察定时器 0 的 T0CLKO 输出
    P0M0 = 0x00; P0M1 = 0xff;
}

void CLK_Init(void)
{
    #if Main_Fosc == Fosc_120Mhz
    WTST = 4;CLKDIV = 2;          //设置系统时钟=480MHz/2/2=120MHz,（因为 CLKSEL 选择时，已经将 HPLL/2
了）
    #elif Main_Fosc == Fosc_80Mhz
    WTST = 3;CLKDIV = 3;          //设置系统时钟=480MHz/2/3=80MHz
    #elif Main_Fosc == Fosc_60Mhz
    WTST = 2;CLKDIV = 4;          //设置系统时钟=480MHz/2/4=60MHz
    #endif
    //以下为超过 60MHz 时，系统时钟使用 HPLL 方式提供
    VRTRIM = CHIPID22;            //载入 27MHz 频段的 VRTRIM 值
    IRTRIM = CHIPID12;            //指定当前 HIRC 为 24MHz，此时会覆盖掉 ISP 设置的时钟频率
    IRCBAND &= ~0x03;             //清空频段选择
    IRCBAND |= 0x01;              //选择 27Mhz 频段
    HPLLCR &= ~0x10;              //选择 HPLL 输入时钟源为 HIRC
    HPLLPDIV = 4;                       //24MHz/4=6MHz,需要保证输入 HPLL 的时钟在 6MHz 附
近
    HPLLCR |= 0x0e;                     //HPLL=6MHz*80=480MHz
    HPLLCR |= 0x80;                     //使能 HPLL
    Delay10ms();
    CLKSEL &= ~0x03;              //BASE_CLK 选择为 HIRC，用以提供给 HPLL
    CLKSEL &= ~0x0c;              //清空主时钟源选择
    CLKSEL |= 1<<2;                     //设置主时钟源为内部 HPLL1 输出/2
}

//配置 DAC 为 DMA 输出，并且使用 OPA1 缓冲模式作为输出
void DAC_Init(void)
{
    //OPA 配置输出
    PGA1_CR1 = 0x73;     //缓冲模式，DACIO 正极输入，P63 输出
    PGA1_CR2 = 0x04;     //允许 PGA1 输出
```

```
    //DAC 配置
    DAC1_DIV = 0;                   //设置 DAC 分频系数，防止超过 2Mhz 推荐最大速度
    //DAC1_CR = 0x81;     //使能 DAC 功能，DAC 使用 DMA 模式，单次触发，无法启动 DMA
    DAC1_CR = 0xb3;                 //使能 DAC 功能，DAC 使用 DMA 模式，连续触发，打开 DAC1
中断
    //DAC1_CR = 0x41;     //使能 DAC 功能，连续触发，用于测试无 DMA 情况下是否正常输出，测试结果：正常
    DAC1_STA = 0x00;      //清空状态标志位
    //DMA 配置
    DMA_ARB_CFG = (0<<5);               //切换到 xdata 部分
//      DMA_DAC1_CFG = 0x00;            //关闭 DMA_DAC 中断
    DMA_DAC1_CFG = 0x8f;                //使能 DMA_DAC 中断
    DMA_DAC1_STA = 0x00;                //清空中断标志位
    DMA_DAC1_AMTH = (unsigned char)((4095-1)>>8);
    DMA_DAC1_AMT = (unsigned char)(4095-1);             //单位是 16bit
    DMA_DAC1_TXAH = (unsigned char)(((unsigned int)&SinTable_Software)>>8);
    DMA_DAC1_TXAL = (unsigned char)((unsigned int)&SinTable_Software);
    DMA_DAC1_ITVH = 0x00;               //长延时，DAC 输出速度主要由此提供
    DMA_DAC1_CR = 0xc0;                 //触发启动
}

void dac1_isr(void) interrupt DAC1_VECTOR
{
    DAC1_STA = 0x00;                    //清空状态标志位
}

void DMA_DAC1_Interrupt(void) interrupt DMA_DAC_VECTOR
{
    DMA_DAC1_STA = 0;P35 = ~P35;DMA_DAC1_CR = 0xc0;   //触发启动
}
```

24 4 组可编程增益放大器（PGA/OPA）

产品线	PGA
STC32G144K246 系列	**4 组**

STC32G144K246 系列单片机内部集成了 4 组可编程增益放大器。模块内含一个 CMOS 运算放大器、偏置电流产生电路、模式可配置等众多电路，当配置成 PGA 功能时可调整增益的电阻器件。单片机内集成了 4 组运算放大电路，每组输出入接口已扩展成正负输入端口各有四个，输出端口也扩展成四个，搭配软件配置，每组放大器最多可将此电路适配于四组输出入 I/O 口，实现四组 I/O 口能有运放/PGA 的功能。

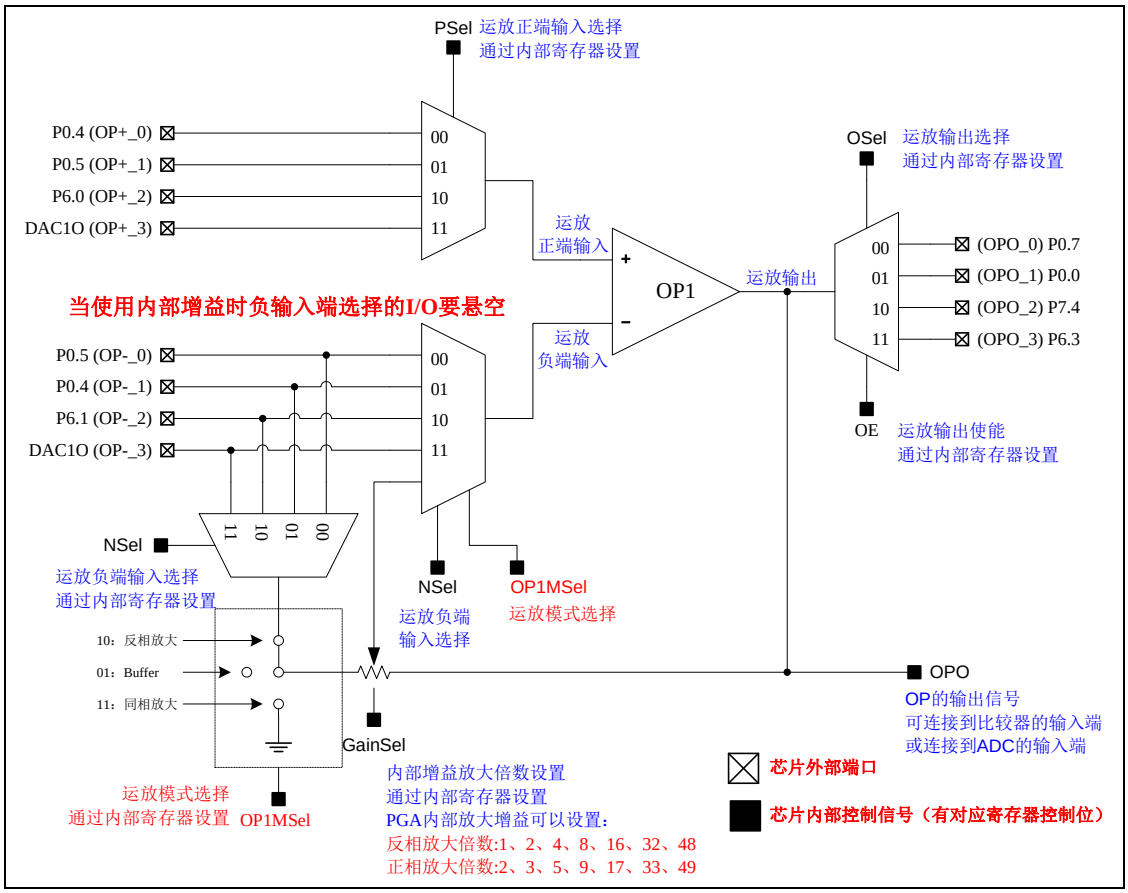
24.1 主要特性

放大器有如下主要特性：

- 工作电压范围：单电源 2.4V~5.5V
- 工作温度范围：-40℃ ~ 125℃
- 工作模式：通用运放、缓冲级、反相放大 PGA 和非反相（同相/正相）放大 PGA
- 输出入电压范围：当作运放使用时，在工作电压范围内，能达到轨对轨(Rail-to-Rail)输入及轨对轨(Rail-to-Rail)输出，即 RRIO
- offset 电压：不具备自校准功能
- 反相 PGA 模式：7 级放大增益可选（1、2、4、8、16、32、48 倍）
- 非反相 PGA 模式：7 级放大增益可选（2、3、5、9、17、33、49 倍）
- 最大负载电流(输出电流)：4.5mA

24.2 PGA1

24.2.1 PGA1 内部结构图



24.2.2 PGA1 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PGA1_CR1	7EF600H	PGA1_MSEL[1:0]		**PGA1_OSEL[1:0]**		**PGA1_NSEL[1:0]**		**PGA1_PSEL[1:0]**	

PGA1_PSEL[1:0]：PGA1正极端口选择寄存器

PGA1_PSEL[1:0]	PGA1正极
00	P0.4
01	P0.5
10	P6.0
11	DAC1O

PGA1_NSEL[1:0]：PGA1负极端口选择寄存器

PGA1_NSEL[1:0]	PGA1负极
00	P0.5
01	P0.4
10	P6.1
11	DAC1O

PGA1_OSEL[1:0]：PGA1输出端口选择寄存器

PGA1_OSEL[1:0]	PGA1输出端口
00	P0.7
01	P0.0
10	P7.4
11	P6.3

24.2.3 PGA1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PGA1_CR1	PGA1 控制寄存器 1	7EF600H	PGA1_MSEL[1:0]		PGA1_OSEL[1:0]		PGA1_NSEL[1:0]		PGA1_PSEL[1:0]		0000,0000
PGA1_CR2	PGA1 控制寄存器 2	7EF601H	-	PGA1_GSEL[2:0]			-	PGA1_OE	PGA1_IBS	PGA1_PD	xx00,0001

24.2.4 PGA1 控制寄存器 1（PGA1_CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PGA1_CR1	7EF600H	PGA1_MSEL[1:0]		PGA1_OSEL[1:0]		PGA1_NSEL[1:0]		PGA1_PSEL[1:0]	

PGA1_MSEL[1:0]：PGA1工作模式选择寄存器

PGA1_MSEL[1:0]	PGA1模式
00	通用运放模式
01	缓冲模式
10	反相放大模式
11	非反相放大模式

PGA1_PSEL[1:0]：PGA1正极端口选择寄存器

PGA1_PSEL[1:0]	PGA1正极
00	P0.4
01	P0.5
10	P6.0
11	DAC1O

PGA1_NSEL[1:0]：PGA1负极端口选择寄存器

PGA1_NSEL[1:0]	PGA1负极
00	P0.5
01	P0.4
10	P6.1
11	DAC1O

PGA1_OSEL[1:0]：PGA1输出端口选择寄存器

PGA1_OSEL[1:0]	PGA1输出端口
00	P0.7
01	P0.0
10	P7.4
11	P6.3

24.2.5 PGA1 控制寄存器 2（PGA1_CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PGA1_CR2	7EF601H	-	PGA1_GSEL[2:0]			-	PGA1_OE	PGA1_IBS	PGA1_PD

PGA1_GSEL[2:0]：PGA1增益控制寄存器

PGA1_GSEL[2:0]	反相放大增益倍数	非反相放大增益倍数
000	N/A	N/A
001	1	2
010	2	3
011	4	5
100	8	9
101	16	17
110	32	33
111	48	49

PGA1_OE：PGA1 输出控制位

0：禁止 PGA1 输出

1：允许 PGA1 输出

PGA1_IBS：PGA1 偏压电流大小选择

0：正常电流大小

1：电流倍增

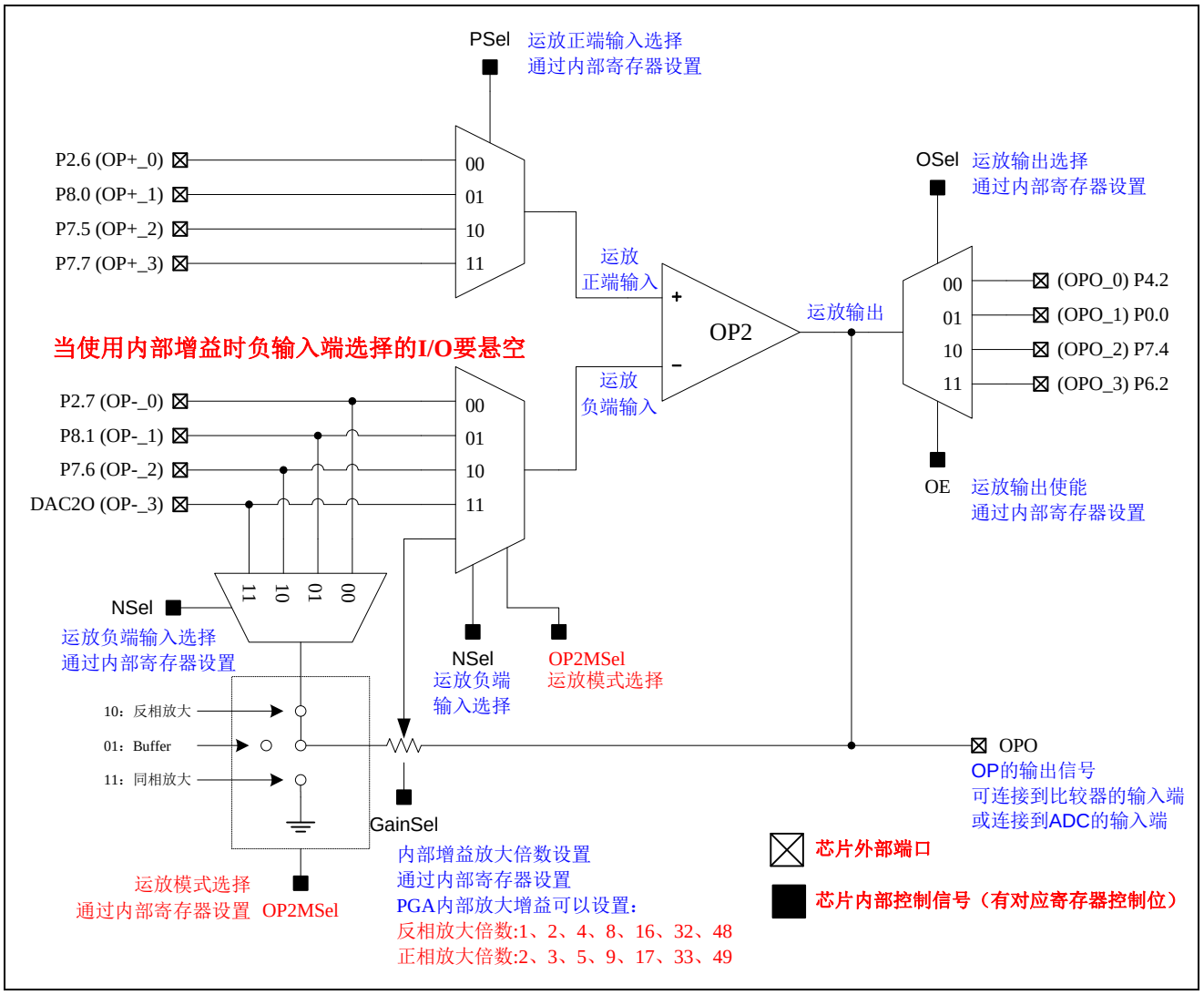
PGA1_PD：PGA1 电源控制

0：禁止掉电（使能 PGA1）

1：使能掉电（关闭 PGA1）

24.3 PGA2

24.3.1 PGA2 内部结构图



24.3.2 PGA2 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PGA2_CR1	7EF604H	PGA2_MSEL[1:0]		**PGA2_OSEL[1:0]**		**PGA2_NSEL[1:0]**		**PGA2_PSEL[1:0]**	

PGA2_PSEL[1:0]：PGA2正极端口选择寄存器

PGA2_PSEL[1:0]	PGA2正极
00	P2.6
01	P6.0
10	P7.5
11	P7.7

PGA2_NSEL[1:0]：PGA2负极端口选择寄存器

PGA2_NSEL[1:0]	PGA2负极
00	P2.7
01	P6.1
10	P7.6
11	DAC2O

PGA2_OSEL[1:0]：PGA2输出端口选择寄存器

PGA2_OSEL[1:0]	PGA2输出端口
00	P4.2
01	P0.0
10	P7.4
11	P6.2

24.3.3 PGA2 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PGA2_CR1	PGA2 控制寄存器 1	7EF604H	PGA2_MSEL[1:0]		PGA2_OSEL[1:0]		PGA2_NSEL[1:0]		PGA2_PSEL[1:0]		0000,0000
PGA2_CR2	PGA2 控制寄存器 2	7EF605H	-	PGA2_GSEL[2:0]			-	PGA2_OE	PGA2_IBS	PGA2_PD	xx00,0001

24.3.4 PGA2 控制寄存器 1（PGA2_CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PGA2_CR1	7EF604H	PGA2_MSEL[1:0]		PGA2_OSEL[1:0]		PGA2_NSEL[1:0]		PGA2_PSEL[1:0]	

PGA2_MSEL[1:0]：PGA2工作模式选择寄存器

PGA2_MSEL[1:0]	PGA2模式
00	通用运放模式
01	缓冲模式
10	反相放大模式
11	非反相放大模式

PGA2_PSEL[1:0]：PGA2正极端口选择寄存器

PGA2_PSEL[1:0]	PGA2正极
00	P2.6
01	P6.0
10	P7.5
11	P7.7

PGA2_NSEL[1:0]：PGA2负极端口选择寄存器

PGA2_NSEL[1:0]	PGA2负极
00	P2.7
01	P6.1
10	P7.6
11	DAC2O

PGA2_OSEL[1:0]：PGA2输出端口选择寄存器

PGA2_OSEL[1:0]	PGA2输出端口
00	P4.2
01	P0.0
10	P7.4
11	P6.2

24.3.5 PGA2 控制寄存器 2（PGA2_CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PGA2_CR2	7EF605H	-	PGA2_GSEL[2:0]			-	PGA2_OE	PGA2_IBS	PGA2_PD

PGA2_GSEL[2:0]：PGA2增益控制寄存器

PGA2_GSEL[2:0]	反相放大增益倍数	非反相放大增益倍数
000	N/A	N/A
001	1	2
010	2	3
011	4	5
100	8	9
101	16	17
110	32	33
111	48	49

PGA2_OE：PGA2 输出控制位

0：禁止 PGA2 输出

1：允许 PGA2 输出

PGA2_IBS：PGA2 偏压电流大小选择

0：正常电流大小

1：电流倍增

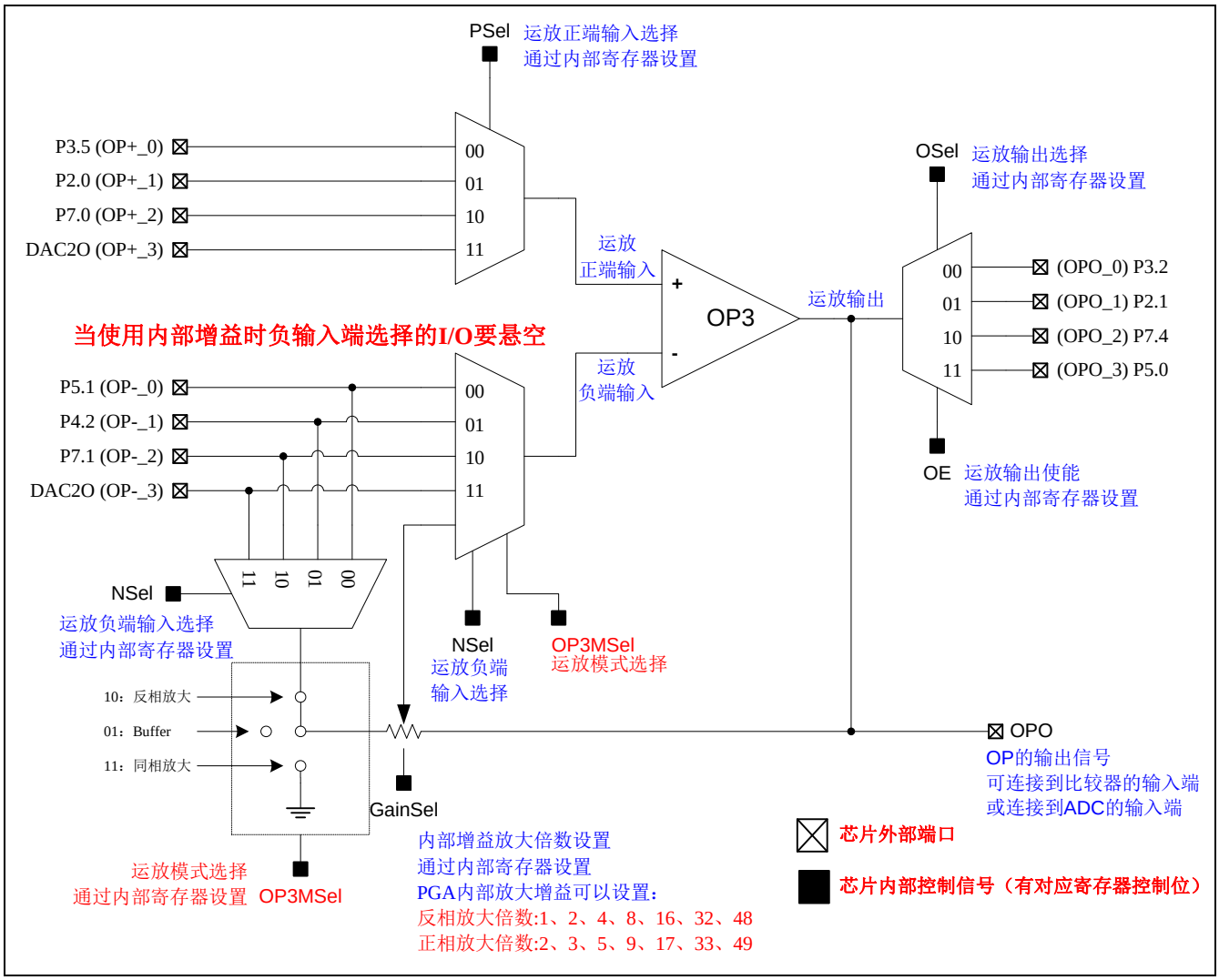
PGA2_PD：PGA2 电源控制

0：禁止掉电（使能 PGA2）

1：使能掉电（关闭 PGA2）

24.4 PGA3

24.4.1 PGA3 内部结构图



24.4.2 PGA3 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PGA3_CR1	7EF608H	PGA3_MSEL[1:0]		**PGA3_OSEL[1:0]**		**PGA3_NSEL[1:0]**		**PGA3_PSEL[1:0]**	

PGA3_PSEL[1:0]：PGA3正极端口选择寄存器

PGA3_PSEL[1:0]	PGA3正极
00	P3.5
01	P2.0
10	P7.0
11	DAC2O

PGA3_NSEL[1:0]：PGA3负极端口选择寄存器

PGA3_NSEL[1:0]	PGA3负极
00	P5.1
01	P4.2
10	P7.1
11	DAC2O

PGA3_OSEL[1:0]：PGA3输出端口选择寄存器

PGA3_OSEL[1:0]	PGA3输出端口
00	P3.2
01	P2.1
10	P7.4
11	P5.0

24.4.3 PGA3 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PGA3_CR1	PGA3 控制寄存器 1	7EF608H	PGA3_MSEL[1:0]		PGA3_OSEL[1:0]		PGA3_NSEL[1:0]		PGA3_PSEL[1:0]		0000,0000
PGA3_CR2	PGA3 控制寄存器 2	7EF609H	-	PGA3_GSEL[2:0]			-	PGA3_OE	PGA3_IBS	PGA3_PD	xx00,0001

24.4.4 PGA3 控制寄存器 1（PGA3_CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PGA3_CR1	7EF608H	PGA3_MSEL[1:0]		PGA3_OSEL[1:0]		PGA3_NSEL[1:0]		PGA3_PSEL[1:0]	

PGA3_MSEL[1:0]：PGA3工作模式选择寄存器

PGA3_MSEL[1:0]	PGA3模式
00	通用运放模式
01	缓冲模式
10	反相放大模式
11	非反相放大模式

PGA3_PSEL[1:0]：PGA3正极端口选择寄存器

PGA3_PSEL[1:0]	PGA3正极
00	P3.5
01	P2.0
10	P7.0
11	DAC2O

PGA3_NSEL[1:0]：PGA3负极端口选择寄存器

PGA3_NSEL[1:0]	PGA3负极
00	P5.1
01	P4.2
10	P7.1
11	DAC2O

PGA3_OSEL[1:0]：PGA3输出端口选择寄存器

PGA3_OSEL[1:0]	PGA3输出端口
00	P3.2
01	P2.1
10	P7.4
11	P5.0

24.4.5 PGA3 控制寄存器 2（PGA3_CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PGA3_CR2	7EF609H	-	PGA3_GSEL[2:0]			-	PGA3_OE	PGA3_IBS	PGA3_PD

PGA3_GSEL[2:0]：PGA3增益控制寄存器

PGA3_GSEL[2:0]	反相放大增益倍数	非反相放大增益倍数
000	N/A	N/A
001	1	2
010	2	3
011	4	5
100	8	9
101	16	17
110	32	33
111	48	49

PGA3_OE：PGA3 输出控制位

0：禁止 PGA3 输出

1：允许 PGA3 输出

PGA3_IBS：PGA3 偏压电流大小选择

0：正常电流大小

1：电流倍增

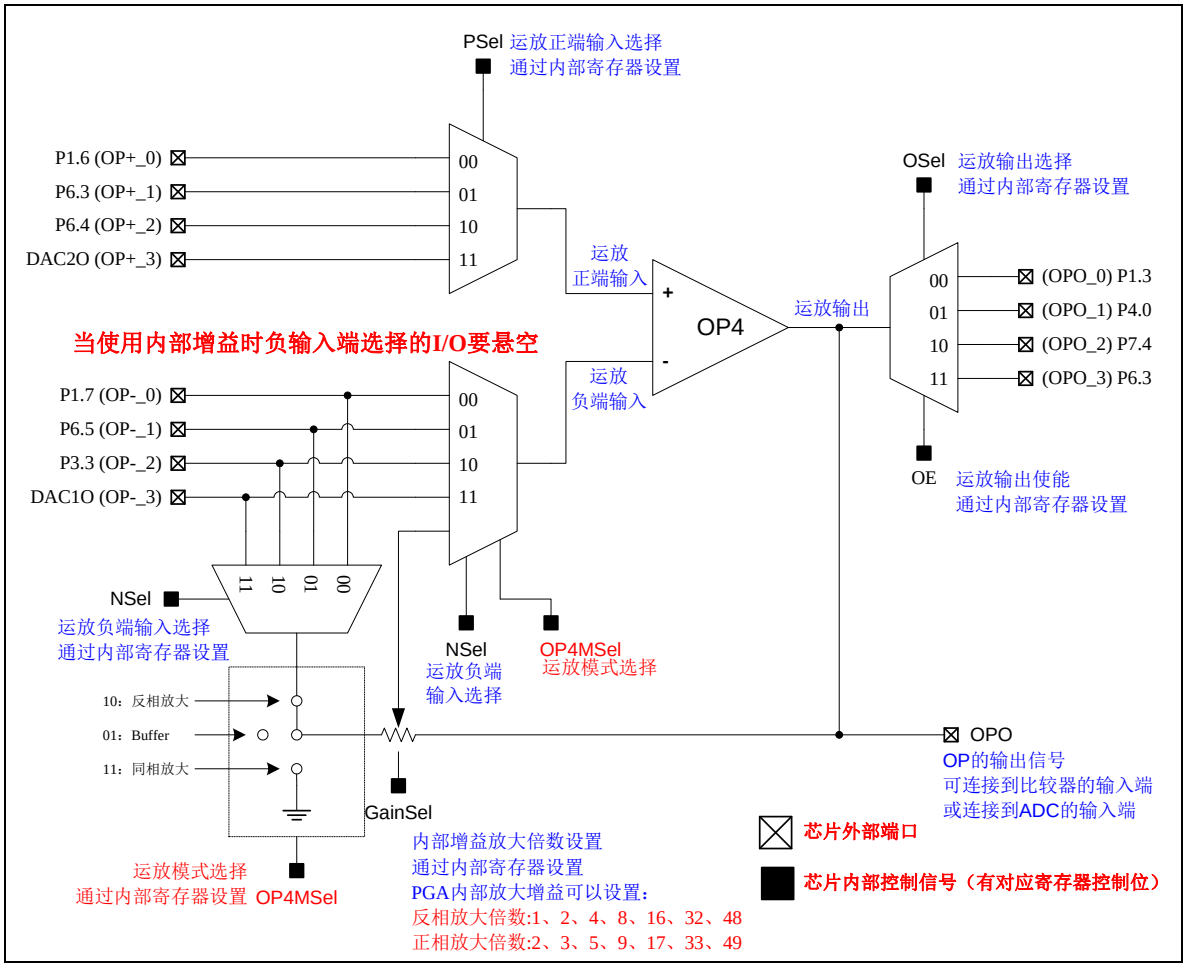
PGA3_PD：PGA3 电源控制

0：禁止掉电（使能 PGA3）

1：使能掉电（关闭 PGA3）

24.5 PGA4

24.5.1 PGA4 内部结构图



24.5.2 PGA4 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PGA4_CR1	7EF60CH	PGA4_MSEL[1:0]		**PGA4_OSEL[1:0]**		**PGA4_NSEL[1:0]**		**PGA4_PSEL[1:0]**	

PGA4_PSEL[1:0]：PGA4正极端口选择寄存器

PGA4_PSEL[1:0]	PGA4正极
00	P1.6
01	P6.3
10	P6.4
11	DAC2O

PGA4_NSEL[1:0]：PGA4负极端口选择寄存器

PGA4_NSEL[1:0]	PGA4负极
00	P1.7
01	P6.5
10	P3.3
11	DAC1O

PGA4_OSEL[1:0]：PGA4输出端口选择寄存器

PGA4_OSEL[1:0]	PGA4输出端口
00	P1.3
01	P4.0
10	P7.4
11	P6.3

24.5.3 PGA4 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PGA4_CR1	PGA4 控制寄存器 1	7EF60CH	PGA4_MSEL[1:0]		PGA4_OSEL[1:0]		PGA4_NSEL[1:0]		PGA4_PSEL[1:0]		0000,0000
PGA4_CR2	PGA4 控制寄存器 2	7EF60DH	-	PGA4_GSEL[2:0]			-	PGA4_OE	PGA4_IBS	PGA4_PD	xx00,0001

24.5.4 PGA4 控制寄存器 1（PGA4_CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PGA4_CR1	7EF60CH	PGA4_MSEL[1:0]		PGA4_OSEL[1:0]		PGA4_NSEL[1:0]		PGA4_PSEL[1:0]	

PGA4_MSEL[1:0]：PGA4工作模式选择寄存器

PGA4_MSEL[1:0]	PGA4模式
00	通用运放模式
01	缓冲模式
10	反相放大模式
11	非反相放大模式

PGA4_PSEL[1:0]：PGA4正极端口选择寄存器

PGA4_PSEL[1:0]	PGA4正极
00	P1.6
01	P6.3
10	P6.4
11	DAC2O

PGA4_NSEL[1:0]：PGA4负极端口选择寄存器

PGA4_NSEL[1:0]	PGA4负极
00	P1.7
01	P6.5
10	P3.3
11	DAC1O

PGA4_OSEL[1:0]：PGA4输出端口选择寄存器

PGA4_OSEL[1:0]	PGA4输出端口
00	P1.3
01	P4.0
10	P7.4
11	P6.3

24.5.5 PGA4 控制寄存器 2（PGA4_CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PGA4_CR2	7EF60DH	-	PGA4_GSEL[2:0]			-	PGA4_OE	PGA4_IBS	PGA4_PD

PGA4_GSEL[2:0]：PGA4增益控制寄存器

PGA4_GSEL[2:0]	反相放大增益倍数	非反相放大增益倍数
000	N/A	N/A
001	1	2
010	2	3
011	4	5
100	8	9
101	16	17
110	32	33
111	48	49

PGA4_OE：PGA4 输出控制位

0：禁止 PGA4 输出

1：允许 PGA4 输出

PGA4_IBS：PGA4 偏压电流大小选择

0：正常电流大小

1：电流倍增

PGA4_PD：PGA4 电源控制

0：禁止掉电（使能 PGA4）

1：使能掉电（关闭 PGA4）

24.6 静态特性（IBS=0）

符号	描述	最小值	典型值	最大值	单位	测试条件
AV_{CC}	工作电压	2.4	5	5.5	V	
$CoreV_{CC}$	内核工作电压	1.35	-	1.65	V	
V_{OS}	输入偏移电压	-10	-	10	mV	V_{CM}= 0.1V ~ (AV_{CC}-0.1V)
V_{CM}	输入共模电压范围	0	-	AV_{CC}	V	
I_{DD_OA}	OPAMP 静态电流	-	127	-	uA	缓冲模式
I_{PD}	睡眠电流	-	0.02	-	uA	
I_{OUT}	输出电流	-4.5	-	4.5	mA	
C_{MRR}	共模抑制比	-	113	-	dB	C_L=50pF
P_{SRR+}	电源抑制比	-	99	-	dB	C_L=50pF
G_{BW}	增益带宽	-	2.5	-	MHz	C_L=50pF
A_{OL}	开环增益	-	113	-	dB	C_L=50pF
Φ_m	相位裕度	-	72	-	deg	C_L=50pF，R_L=2KΩ
T_{SR+}	正转换速率	-	1.5	-	V/us	C_L=50pF，step 1V
T_{settle}	稳定时间	-	0.8	-	us	0.1% final value，Gain=1，+1V step 1V
E_{NI}	ENI 输入噪声电压	-	38	-	uV	AV_{CC}=5V,V_{IN}=2.5V f=0.1~10Hz
E_{NI}	输入噪声密度	-	180	-	nV/√Hz	AV_{CC}=5V,V_{IN}=2.5V f=1KHz
		-	61	-		AV_{CC}=5V,V_{IN}=2.5V f=10KHz
反相 PGA 模式	GSEL[2:0]=0	-	N/A	-		AV_{CC}=5.0V, I_{OUT}=0mA
	GSEL[2:0]=1	0.98	1	1.02		
	GSEL[2:0]=2	1.96	2	2.04		
	GSEL[2:0]=3	3.92	4	4.08		
	GSEL[2:0]=4	7.84	8	8.16		
	GSEL[2:0]=5	15.68	16	16.32		
	GSEL[2:0]=6	31.36	32	32.64		
	GSEL[2:0]=7	47.04	48	48.96		
非反相 PGA 模式	GSEL[2:0]=0	-	N/A	-		AV_{CC}=5.0V, I_{OUT}=1mA
	GSEL[2:0]=1	1.96	2	2.04		
	GSEL[2:0]=2	2.94	3	3.06		
	GSEL[2:0]=3	4.9	5	5.1		
	GSEL[2:0]=4	8.82	9	9.18		
	GSEL[2:0]=5	16.66	17	17.34		
	GSEL[2:0]=6	32.36	33	33.66		
	GSEL[2:0]=7	48.02	49	49.98		

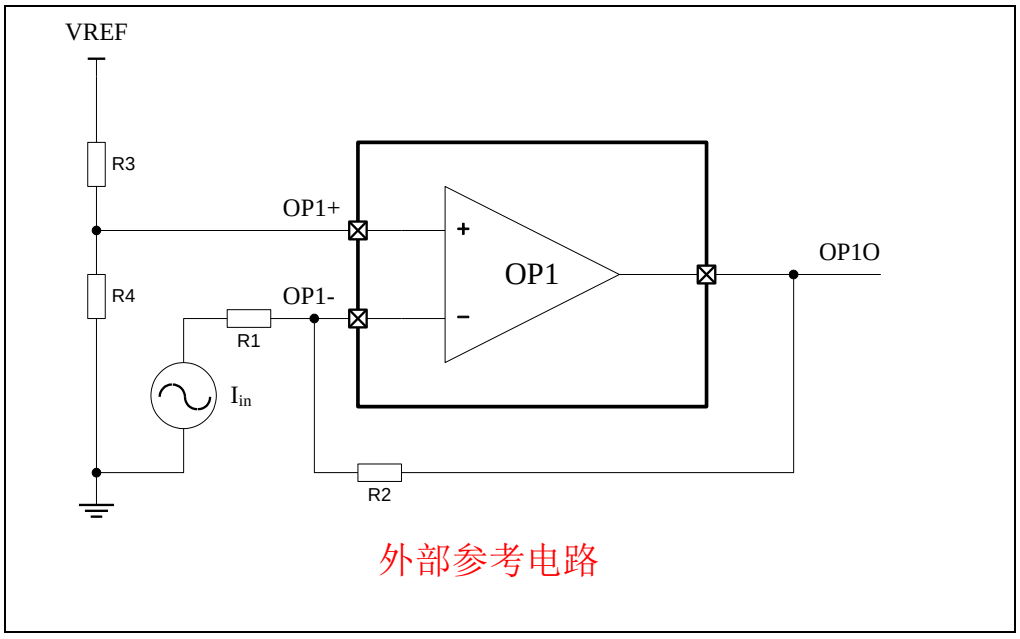
24.7 放大器的典型应用

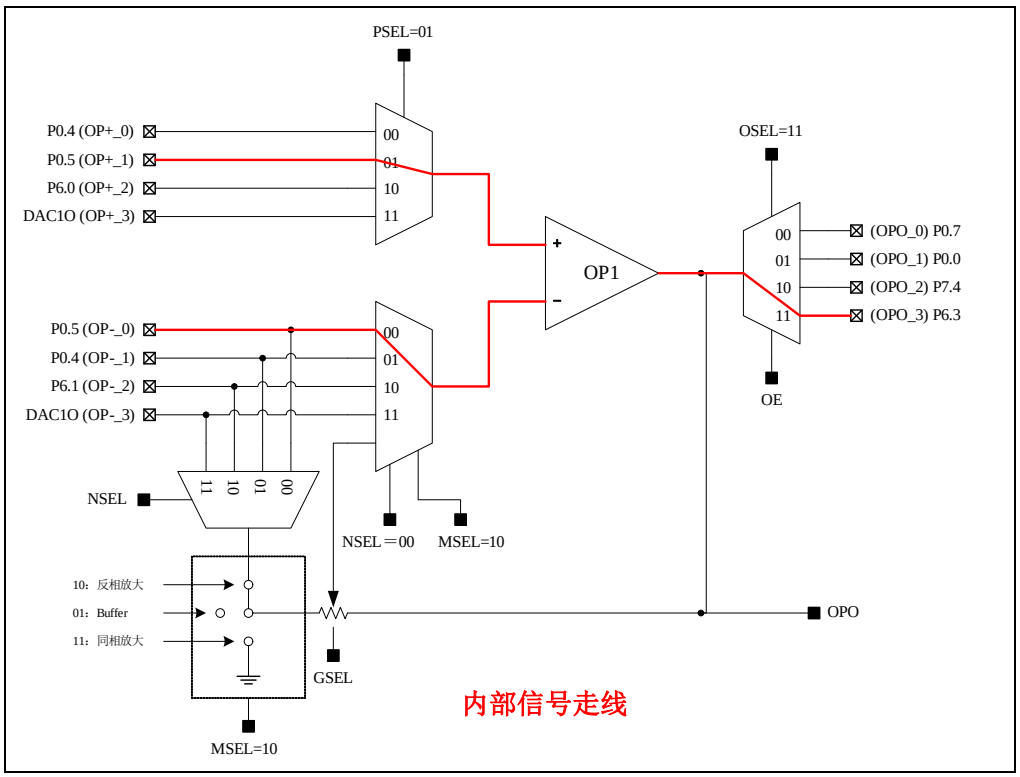
24.7.1 反相放大模式

一般作电流转电压时的应用，可以用此电路 (如烟感 ….)

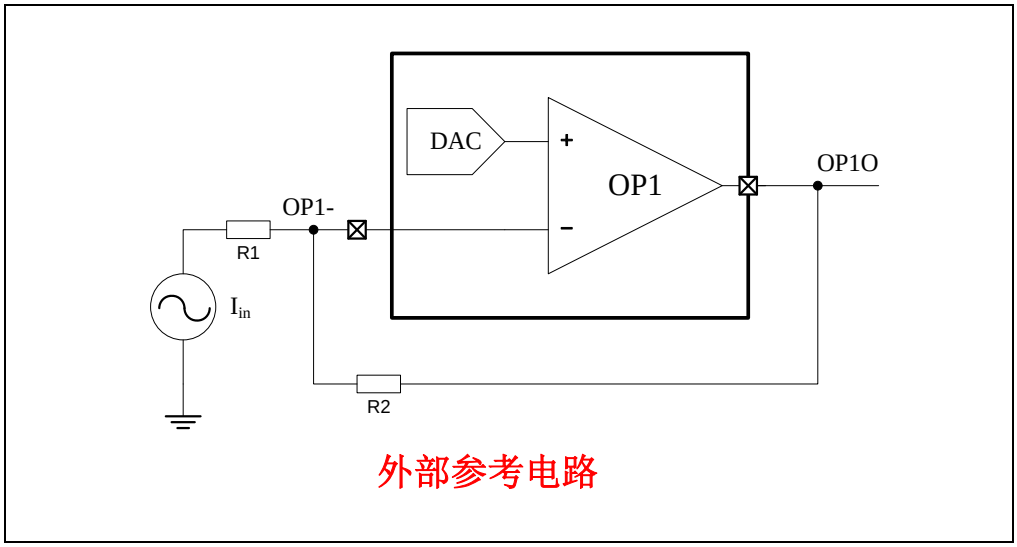
我们的芯片可以做以下设置

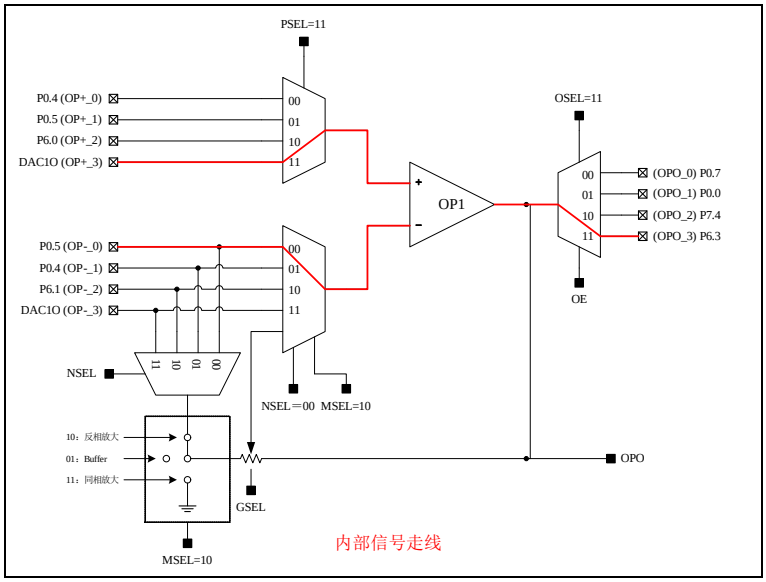
1. OP 对外部信号进行反相放大，增益电路外接，OP 工作在缓冲模式



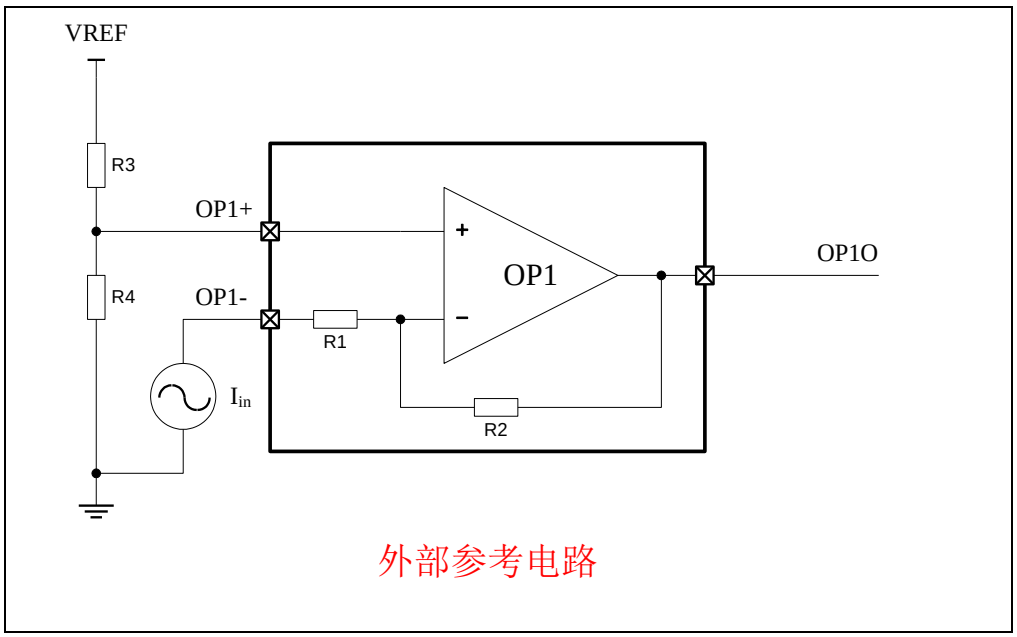


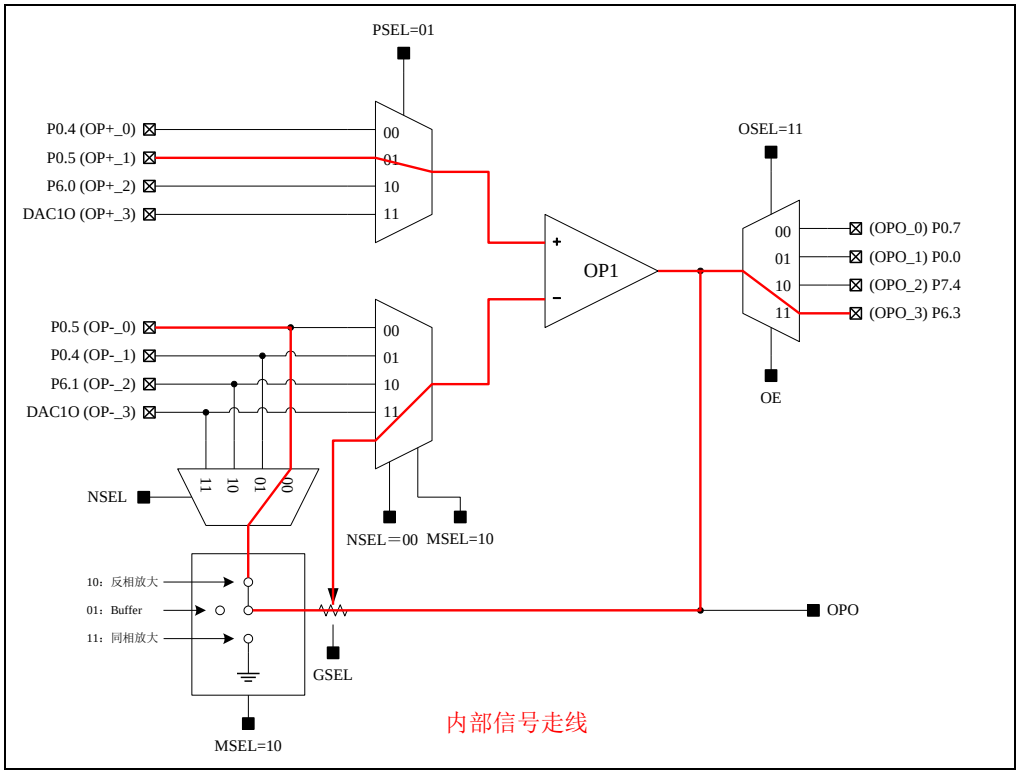
2. OP 对内部 DAC 输出信号进行反相放大，增益电路外接，OP 工作在缓冲模式



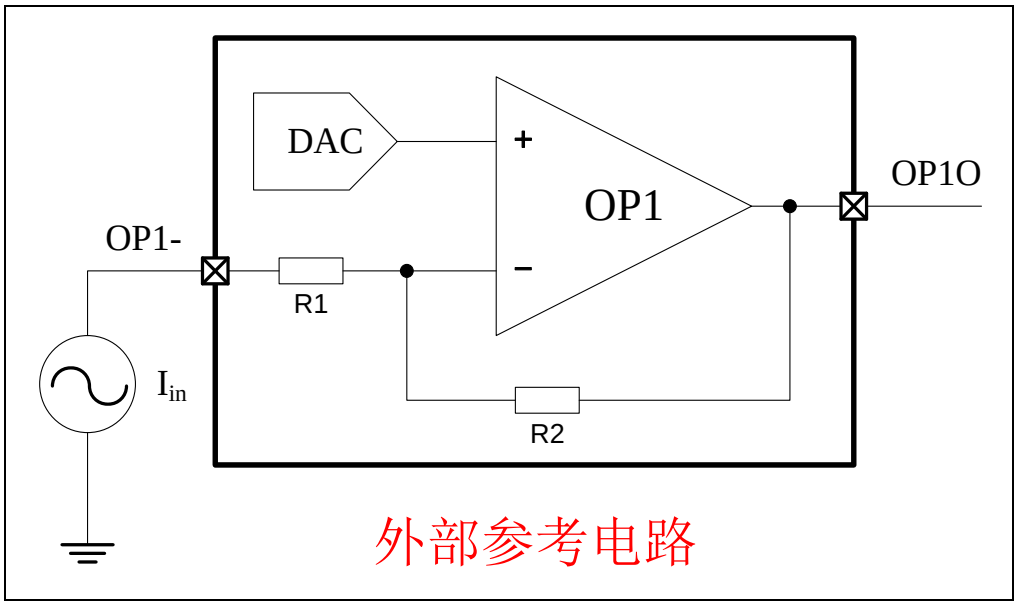


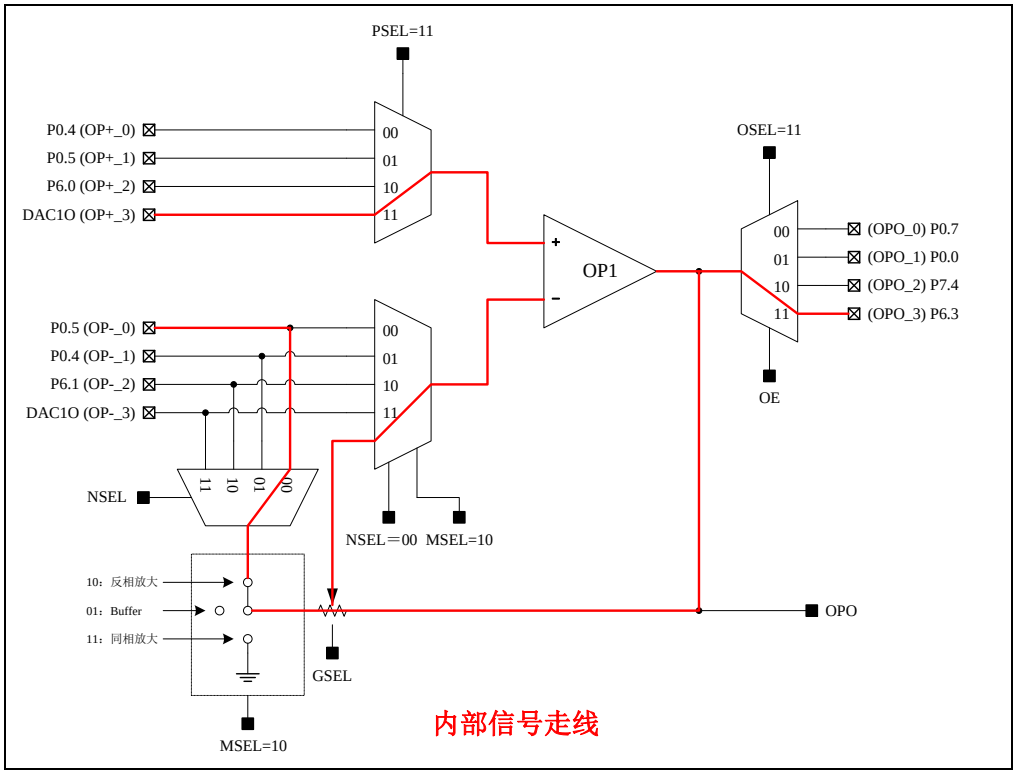
3. OP 对外部信号进行反相放大，使用内部增益电路，OP 工作在反相放大模式（不建议使用，用内部 DAC 更好)





4. OP 对内部 DAC 输出信号进行反相放大，使用内部增益电路，OP 工作在反相放大模式

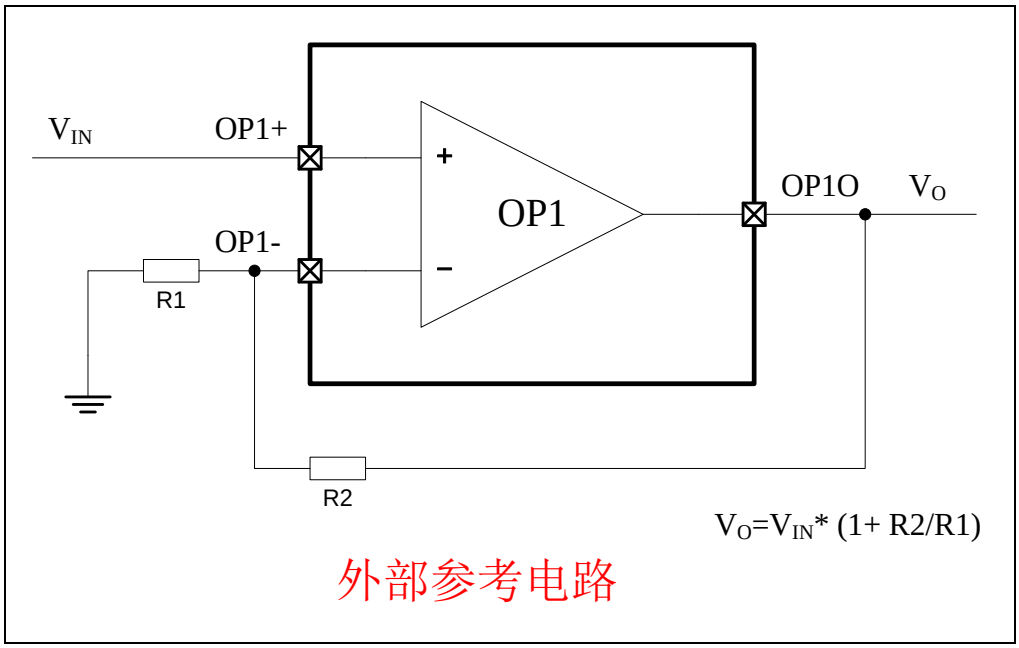


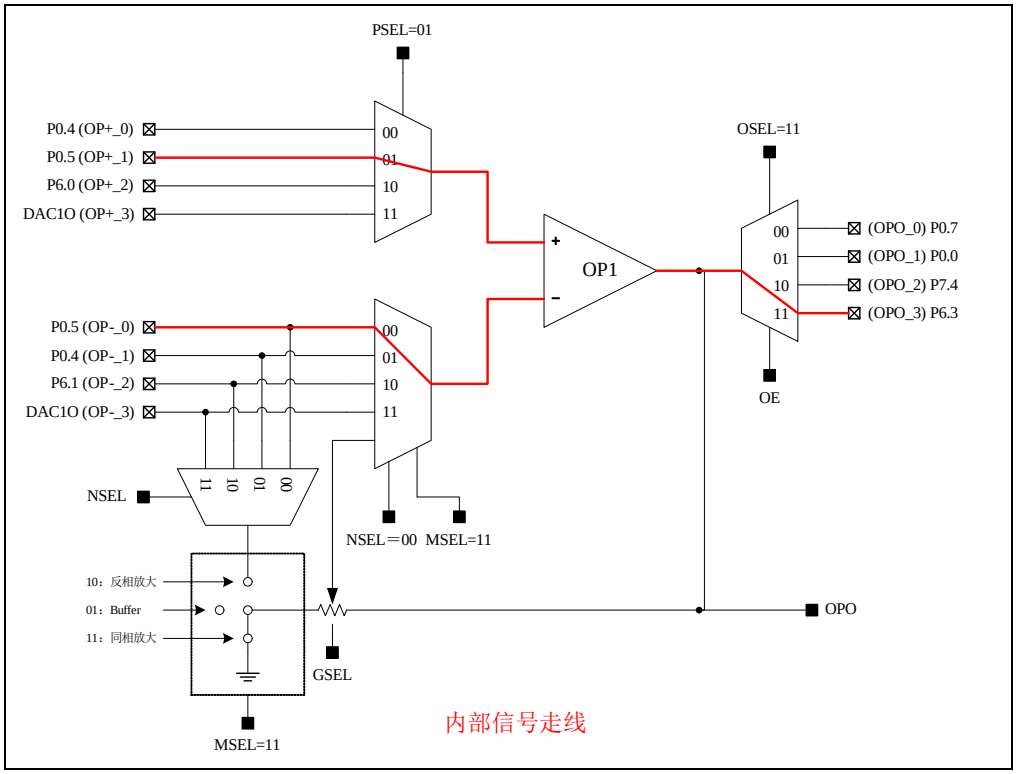


24.7.2 非反相放大模式，同相/正相放大

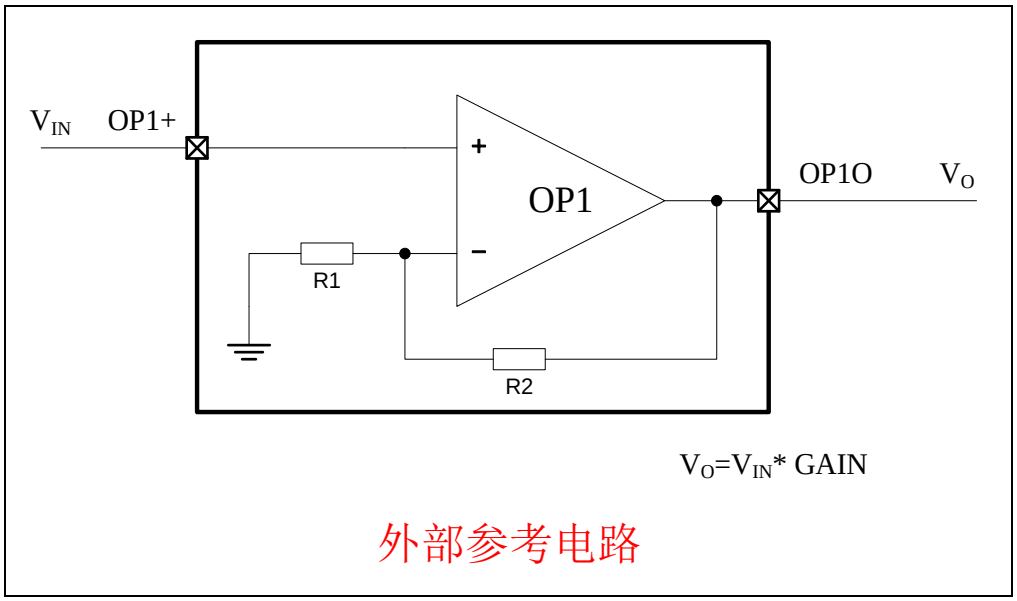
一般作小电压放大时的应用，可以用此电路如(压力表头，耳温枪，电流侦测，过电流保护)
我们的芯片可以做以下设置

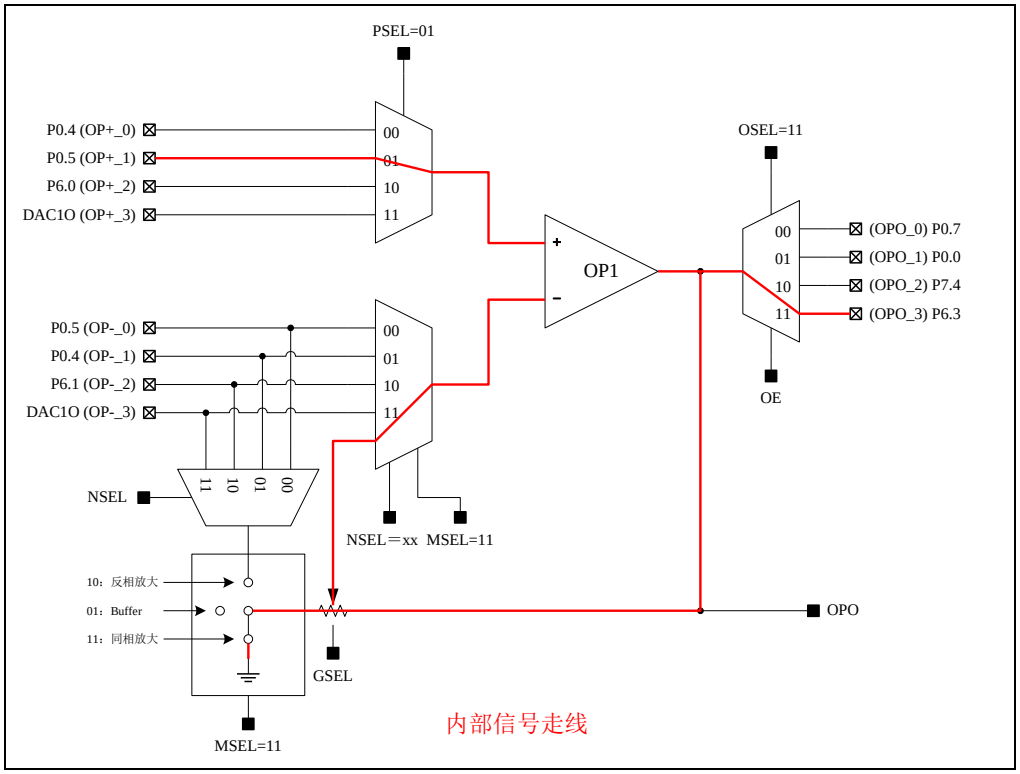
1. OP 对外部信号进行正相放大，增益电路的电阻外接，OP 工作在缓冲模式



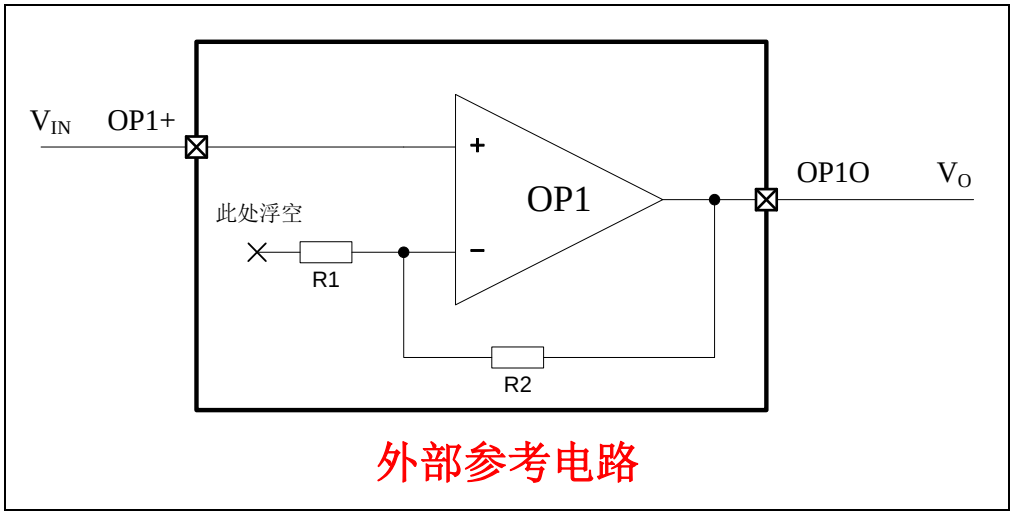


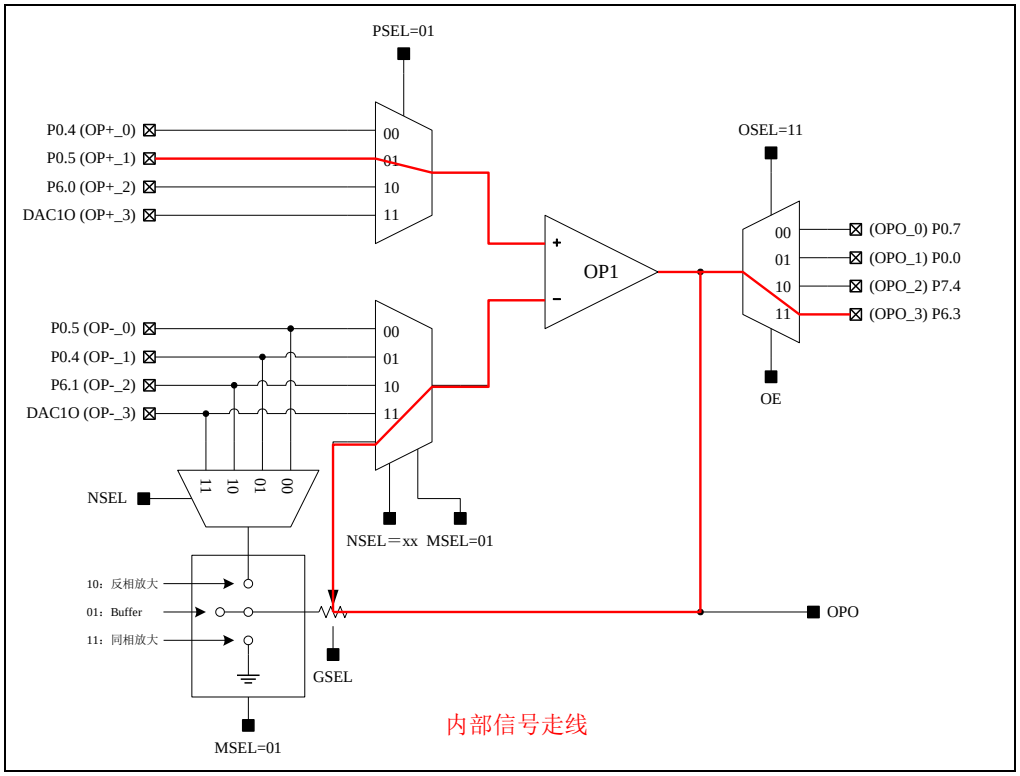
2. OP 对外部信号进行正相放大，使用内部增益电路，OP 工作在正相放大模式



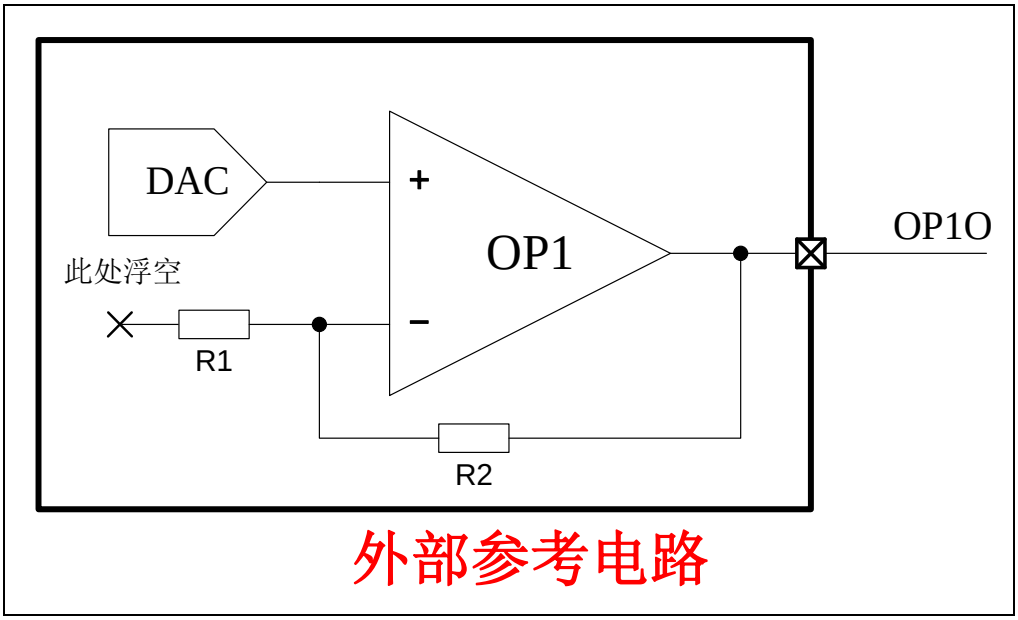


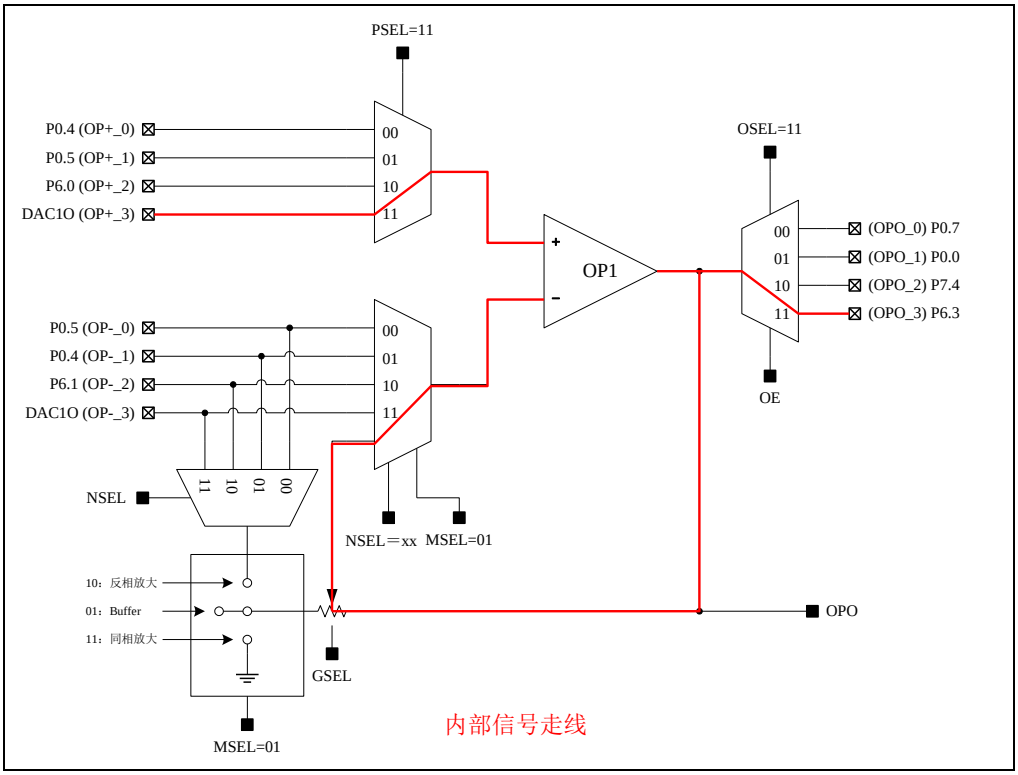
24.7.3 输入缓冲模式


V_{IN}
OP1+
+
OP1
−
OP1O
V_O
此处浮空
R1
R2
外部参考电路


PSEL=01
P0.4 (OP+_0)
P0.5 (OP+_1)
P6.0 (OP+_2)
DAC1O (OP+_3)
00
01
10
11
OSEL=11
OP1
(OPO_0) P0.7
(OPO_1) P0.0
(OPO_2) P7.4
(OPO_3) P6.3
OE
P0.5 (OP-_0)
P0.4 (OP-_1)
P6.1 (OP-_2)
DAC1O (OP-_3)
NSEL
NSEL＝xx MSEL=01
10：反相放大
01：Buffer
11：同相放大
OPO
GSEL
MSEL=01
内部信号走线

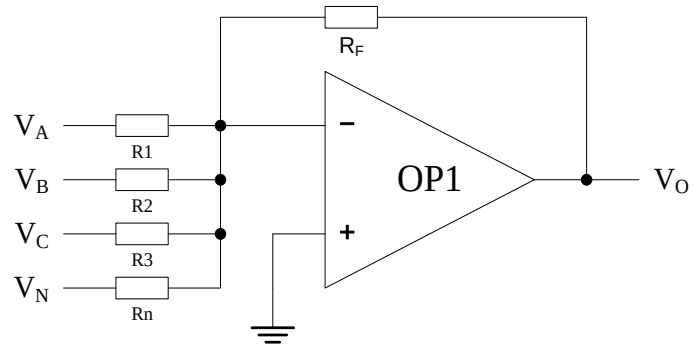
24.7.4 DAC 缓冲模式





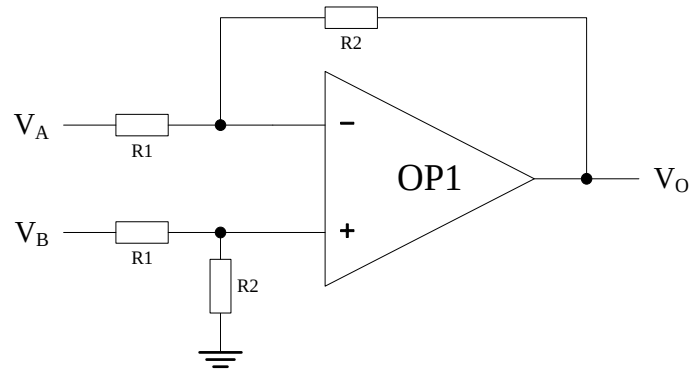
24.7.5 常见的运放应用电路，不是本器件的内部特定电路

Other mode:(建议使用 GP MODE 来外接电路)



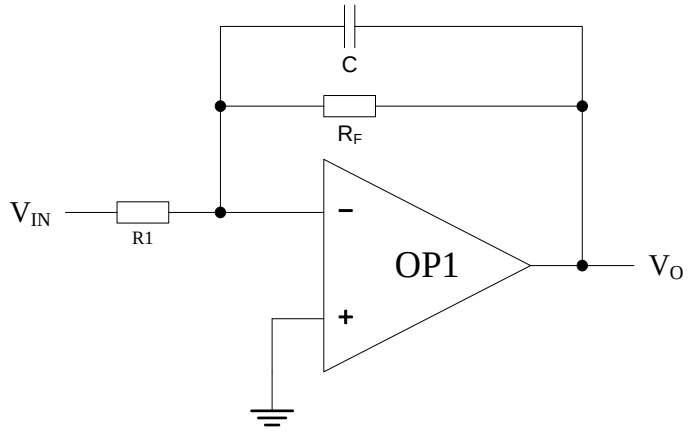
$$Vo = (-R_F)\left(\frac{V_A}{R_1} + \frac{V_B}{R_2} + \frac{V_C}{R_3} + \frac{V_N}{R_n}\right)$$

加 法 器



$$V_O = (V_B - V_A)\left(\frac{R_2}{R_1}\right)$$

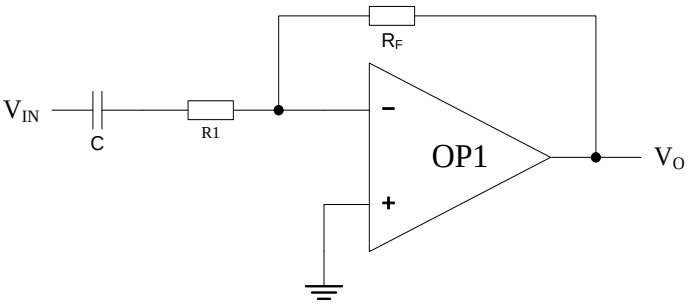
减 法 器



$$T = RC = R_FC$$

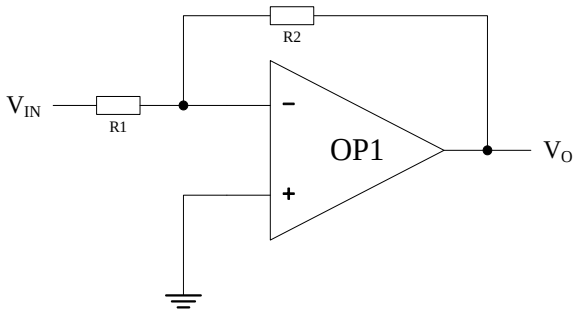
$$V_O = V_{IN}\left(-\frac{R_F}{R_1} \times \frac{1}{1 + j\omega R_FC}\right)$$

低通滤波器/积分器



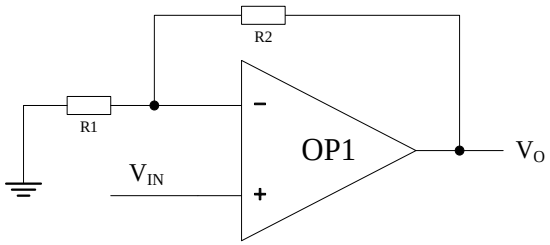
$$V_O = V_{IN}\left(-\frac{R_F}{R_1} \times \frac{1}{1 + \frac{1}{j\omega R_1C}}\right)$$

高通滤波器/微分器



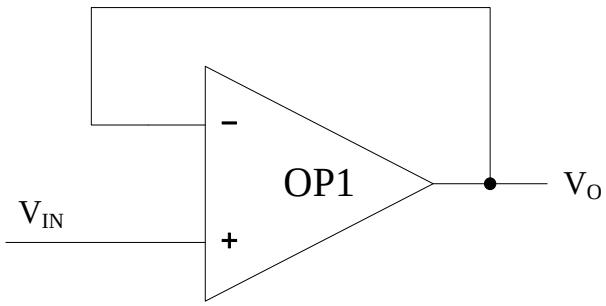
$$V_O = -\frac{R_2}{R_1} V_{IN}$$

反相放大器



$$V_O = \left(1 + \frac{R_2}{R_1}\right) V_{IN}$$

同相放大器

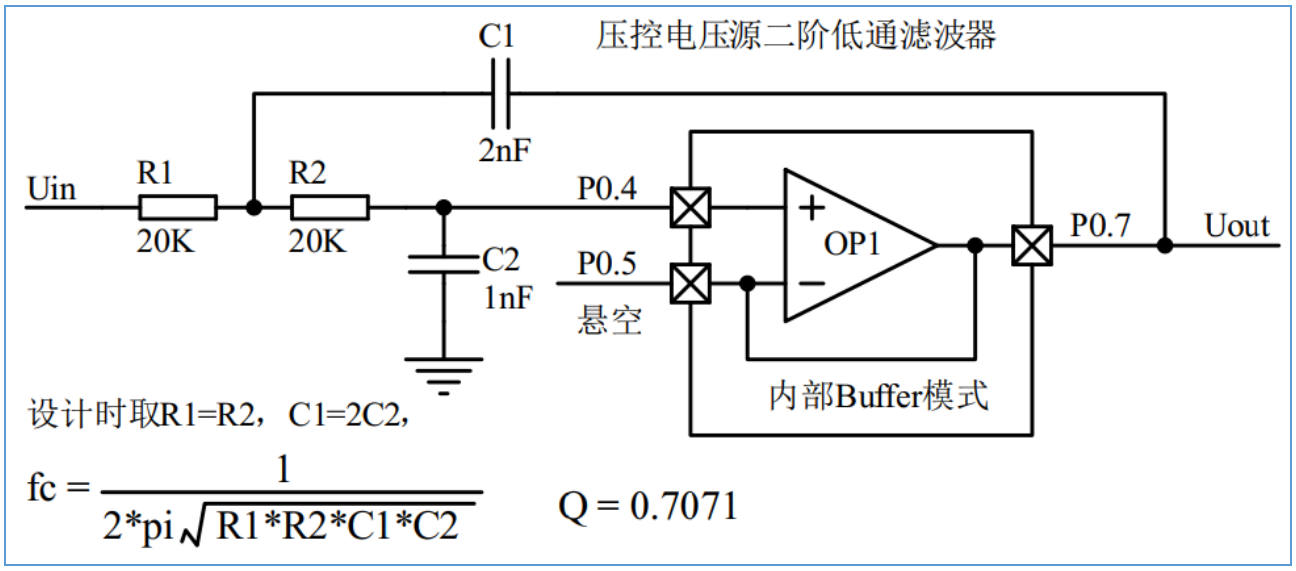


$$V_O = V_{IN}$$

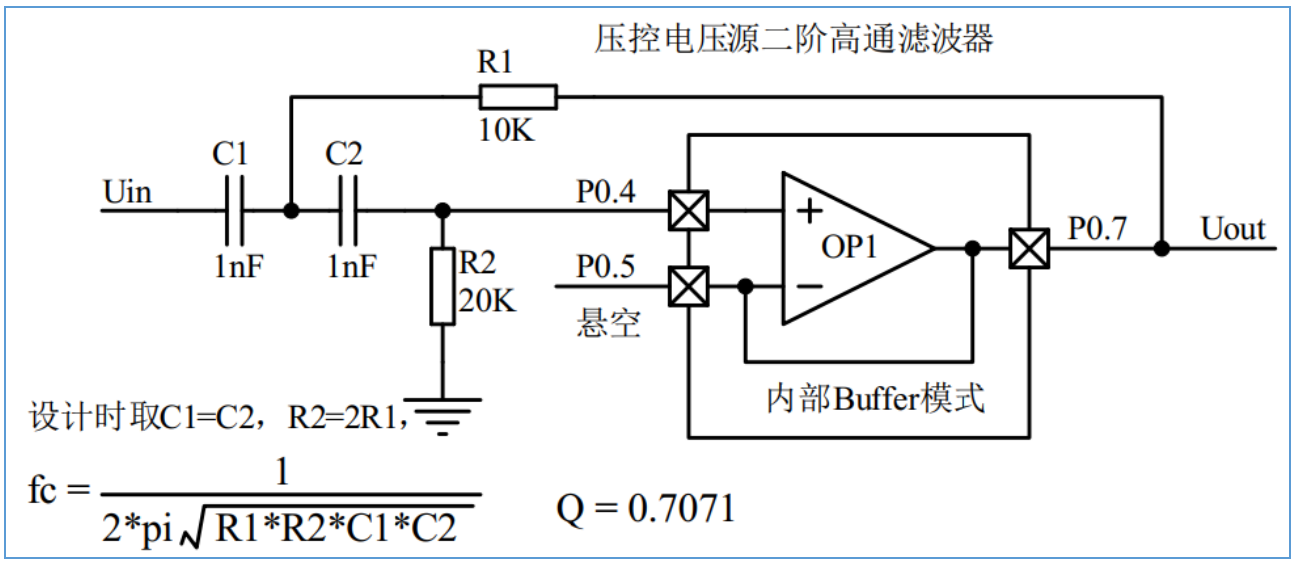
电压跟随器

24.7.6 有源滤波器应用

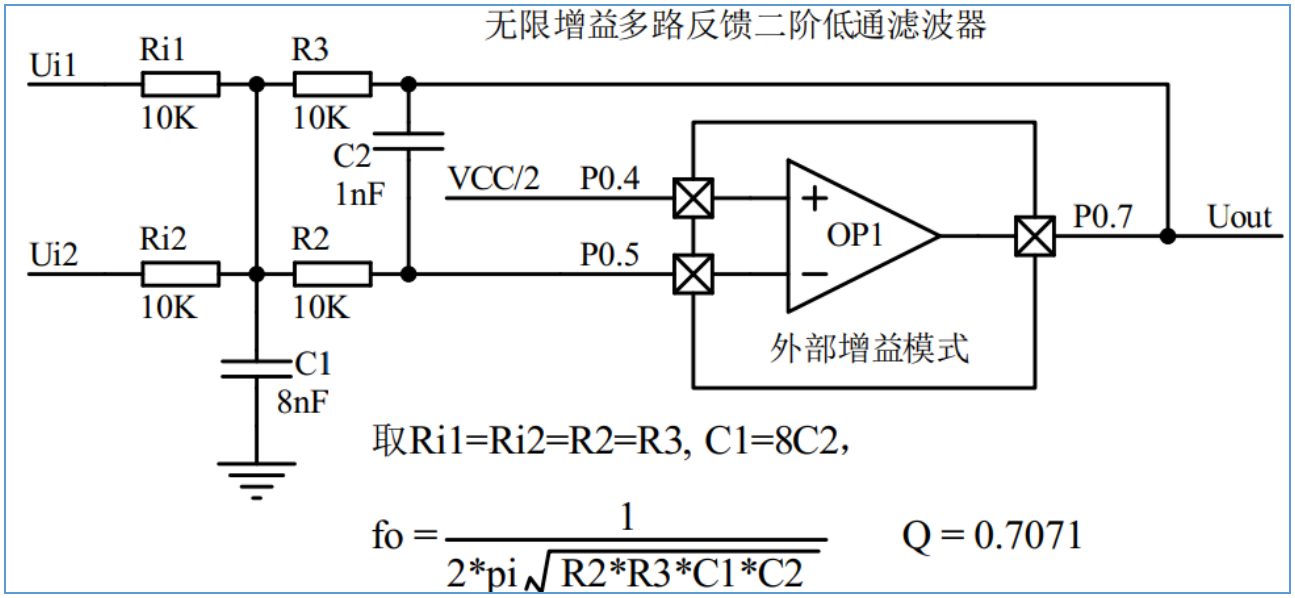
1、



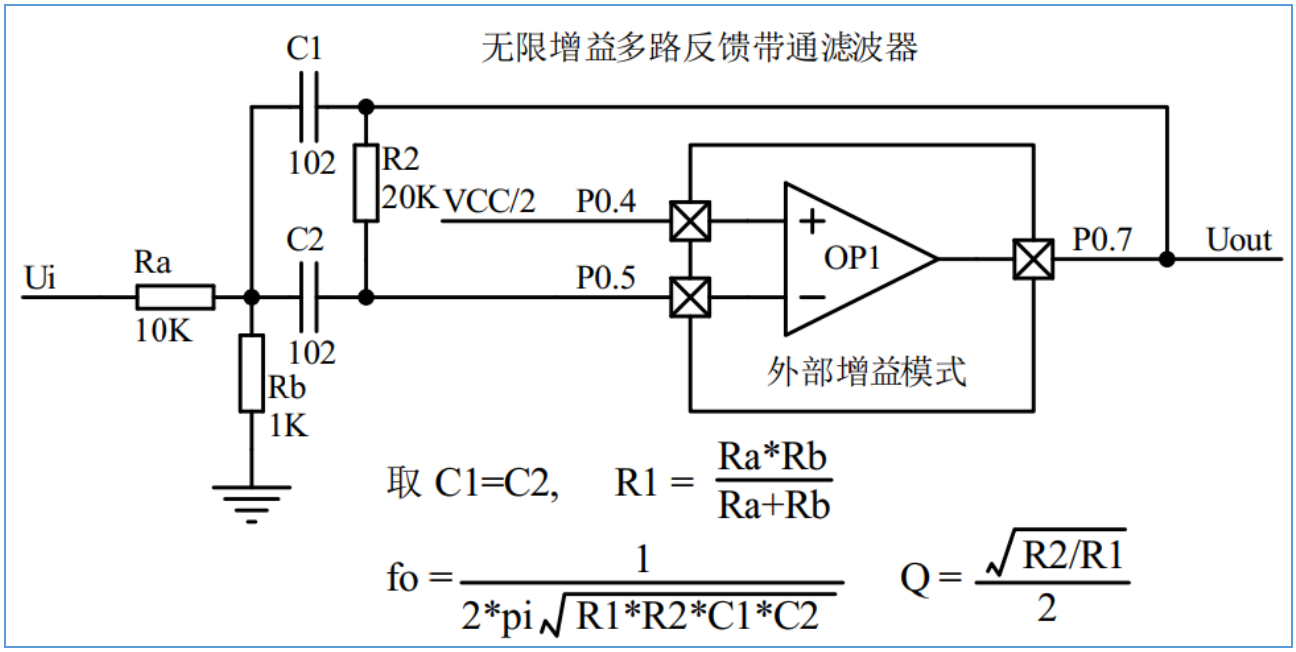
2、



3、


无限增益多路反馈二阶低通滤波器
Ui1
Ri1
10K
R3
10K
C2
1nF
VCC/2
P0.4
Ui2
Ri2
10K
R2
10K
P0.5
OP1
P0.7
Uout
外部增益模式
C1
8nF
取Ri1=Ri2=R2=R3, C1=8C2,
fo = 1 / 2*pi√(R2*R3*C1*C2)
Q = 0.7071

4、

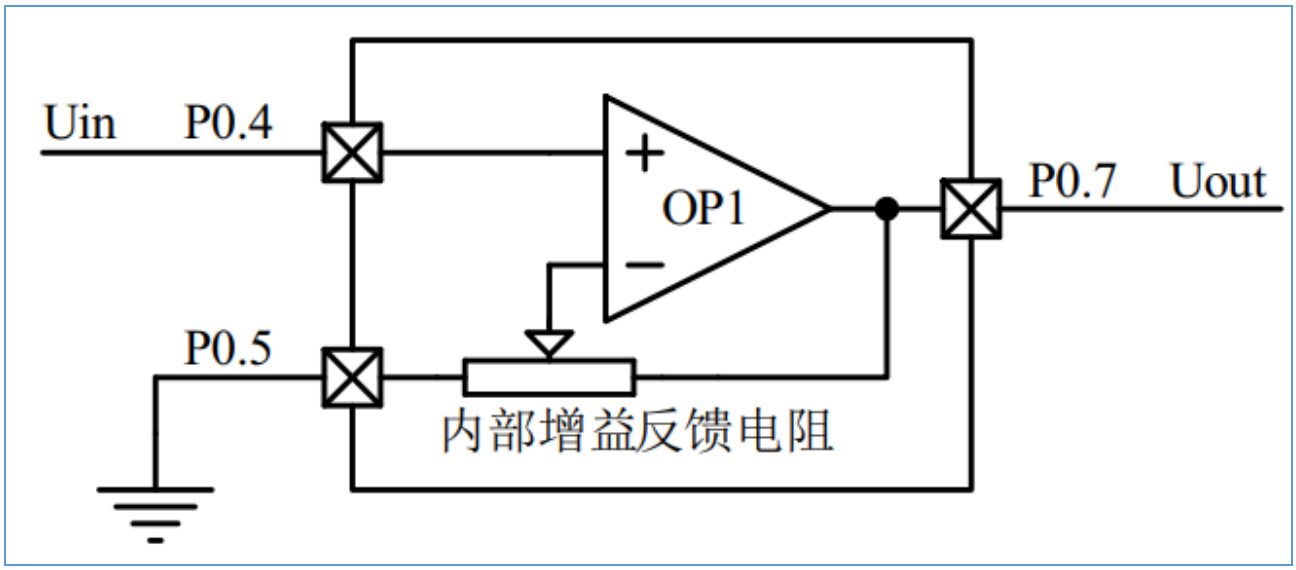

无限增益多路反馈带通滤波器
C1
102
R2
20K
VCC/2
P0.4
C2
Ui
Ra
10K
P0.5
OP1
P0.7
Uout
102
Rb
1K
外部增益模式
取 C1=C2,
R1 = Ra*Rb / (Ra+Rb)
fo = 1 / 2*pi√(R1*R2*C1*C2)
Q = √(R2/R1) / 2

24.8 运放应用笔记及注意事项

1、内部正相放大增益的反馈电阻接到内部的 GND，内部地线有复杂的电流回路，所以地线可能会有较大噪声，要获得更低的噪声，可以：

A、运放模式设置为 mode2：反相放大模式，将负输入端选择的 I/O 再外部接低噪声地。

举例：OP1 正输入端选择 P0.4，负输入端选择 P0.5，输出选择 P0.7，反相放大模式，增益为 8。外部电路将 P0.5 接低噪声地，信号从 P0.4 输入，此时实际仍旧是正相放大，增益为 9。

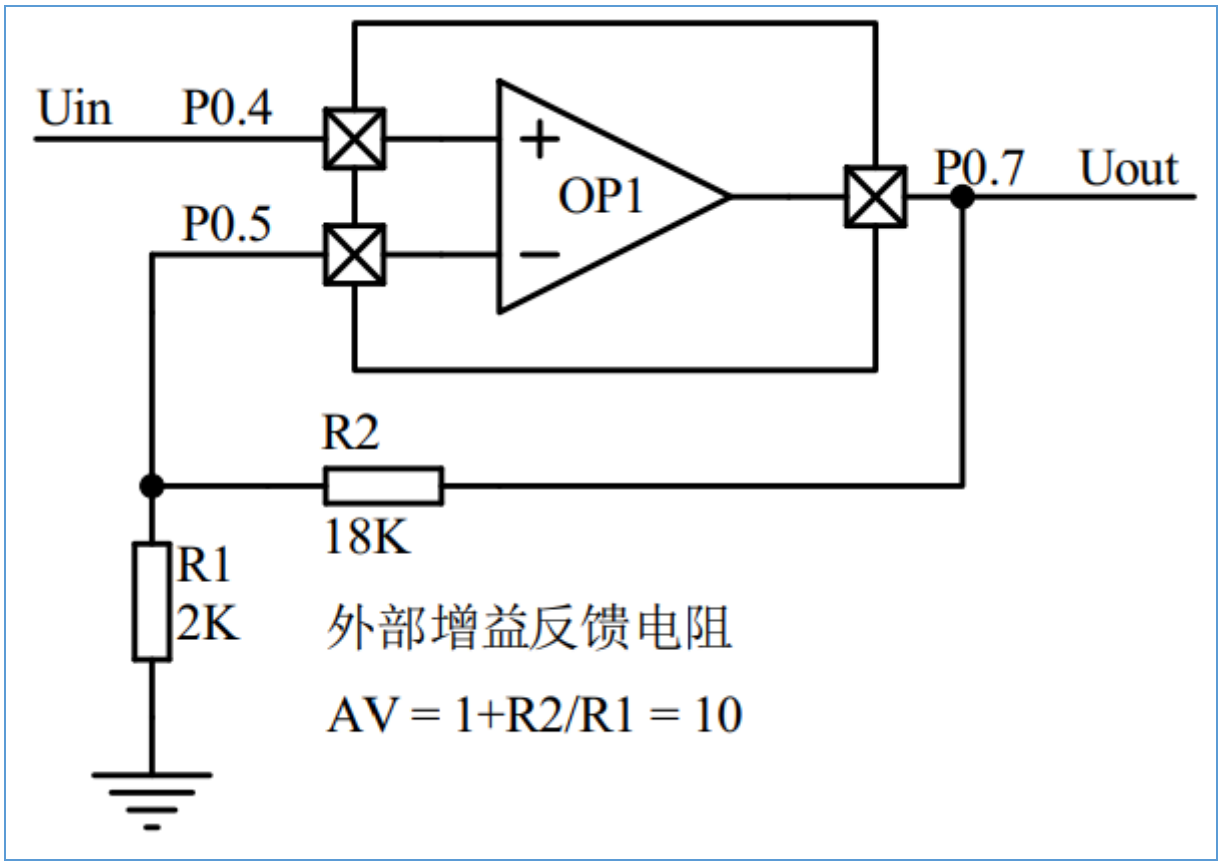



```
#define  PGA1_PSEL      0     //[Bit1 Bit0]   正输入端选择   0:P0.4, 1:P0.5, 2:P6.0, 3:DAC1O
#define  PGA1_NSEL      0     //[Bit3 Bit2]   负输入端选择   0:P0.5, 1:P0.4, 2:P6.1, 3:DAC1O
#define  PGA1_OSEL      0     //[Bit5 Bit4]   输出端选择     0:P0.7, 1:P0.0, 2:P7.4, 3:P6.3
#define  PGA1_MSEL      2     //[Bit7 Bit6]   模式选择       0:non, 1:Buffer, 2:Invert, 3:NonInvert
#define  PGA1_PWD       0     //[Bit0]        掉电控制       0: 电源打开，  1：电源关闭
#define  PGA1_IBS       0     //[Bit1]        偏置电流选择   0:偏置电流小  1：偏置电流大
#define  PGA1_OE        1     //[Bit2]                       1: 使能放大器外扩接口
#define  PGA1_GSEL      4     //[Bit5~Bit3]   增益选择       反相增益 1、2、4、8、16、32、48。
                              //                             同相增益 2、3、5、9、17、33、49。
PGA1_CR1 = (PGA1_MSEL<<6) | (PGA1_OSEL<<4) | (PGA1_NSEL<<2) | PGA1_PSEL;
PGA1_CR2 = (PGA1_GSEL<<4) | (PGA1_OE<<2)    | (PGA1_IBS<<1)   | PGA1_PWD;
```

B、运放模式设置为 mode0，增益由外部电阻确定，根据需要设置合适的增益，反馈电阻接低噪声地。

举例：OP1 正输入端选择 P0.4，负输入端选择 P0.5，输出选择 P0.7，外部增益模式，增益为 10。
上面的模式选择改为 mode0：外部增益控制。

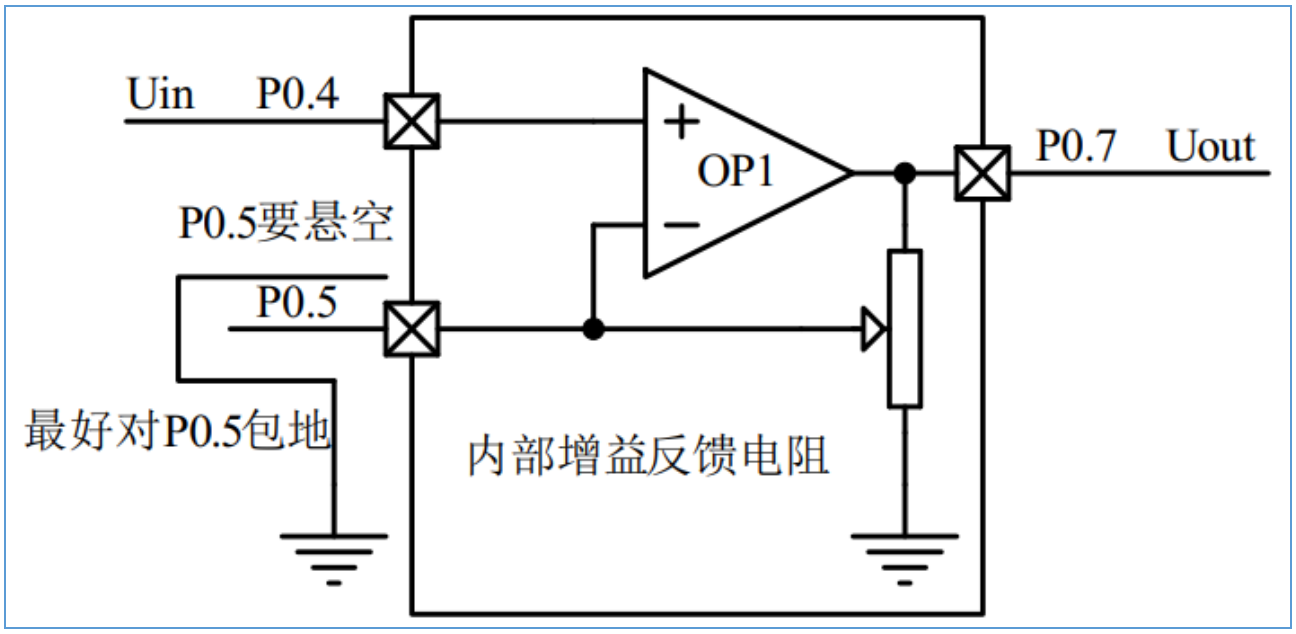
```
#define     PGA1_MSEL     0      //[Bit7 Bit6]   模式选择      0:non,   1:Buffer, 2:Invert, 3:NonInvert
```



2、运放的负输入端 INN 选择某个 I/O 后，INN 就会跟这个 I/O 链接，即使使用内部正相放大增益，这个 I/O 依旧会保持跟 INN 链接，为了避免干扰，这个 I/O 要悬空，PCB 能包地更好。

上面的模式选择改为 mode3：正相放大、内部增益控制。

```
#define     PGA1_MSEL     0      //[Bit7 Bit6]   模式选择      0:non,   1:Buffer, 2:Invert, 3:NonInvert
```


Uin
P0.4
P0.5要悬空
P0.5
最好对P0.5包地
OP1
内部增益反馈电阻
P0.7
Uout

25 3 组同步串行外设接口（SPI）

产品线	SPI
STC32G144K246 系列	**3 组**

STC32G144K246 系列单片机内部集成了 3 组高速串行通信接口——SPI 接口。SPI 是一种全双工的高速同步通信总线。STC32G144K246 系列集成的 SPI 接口提供了两种操作模式：主模式和从模式。

STC32G144K246 系列单片机为 SPI 提供了高速模式（HSSPI）。高速 SPI 是以普通 SPI 为基础，增加了高速模式。当系统运行在较低工作频率时，高速 SPI 可工作在高达 144M 的频率下。从而可以达到降低内核功耗，提升外设性能的目的

注：USART1~USART8 的 SPI 模式可支持两组完整的 SPI，详细情况请参考 USART 章节

25.1 SPI1

25.1.1 SPI 的 MOSI 和 MISO 脚交换控制

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI_CFG2	7EFBF9H	MOSI_PL	**SPI_IOSW**	HSSPI_EN	HSSPI_FIFOEN	HSSPI_SS_DACT[3:0]			

SPI_IOSW：交换 MOSI 和 MISO 脚位（普通 SPI 模式和高速 SPI 模式通用）

0：不交换，维持上电默认脚位。

1：交换 MOSI 和 MISO 的脚位。

25.1.2 SPI 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW1	A2H	S1_S[1:0]		CAN_S[1:0]		**SPI_S[1:0]**		LIN_S[1:0]	

SPI_S[1:0]：SPI 功能脚选择位

SPI_S[1:0]	SS	MOSI	MISO	SCLK
00	P1.2	P1.3	P1.4	P1.5
01	P2.2	P2.3	P2.4	P2.5
10	P5.4	P4.0	P4.1	P4.3
11	P3.5	P3.4	P3.3	P3.2

25.1.3 SPI 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
SPSTAT	SPI 状态寄存器	CDH	SPIF	WCOL	-	-	-	-	-	-	00xx,xxxx
SPCTL	SPI 控制寄存器	CEH	SSIG	SPEN	DORD	MSTR	CPOL	CPHA	SPR[1:0]		0000,0100
SPDAT	SPI 数据寄存器	CFH									0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
SPITOCR	SPI 从机超时控制寄存器	7EFD80H	SPIENTO	SPIENTOI	SPISCALE	-	-	-	-	-	000x,xxxx
SPITOSR	SPI 从机超时状态寄存器	7EFD81H	SPICTOIF	-	-	-	-	-	-	SPITOIF	0xxx,xxx0
SPITOTH	SPI 从机超时长度寄存器	7EFD82H	SPITM[15:8]								0000,0000
SPITOTL	SPI 从机超时长度寄存器	7EFD83H	SPITM[7:0]								0000,0000
SPITOTE	SPI 接收超时长度寄存器	7EFD8CH	SPITM[23:16]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
HSSPI_CFG	高速 SPI 配置寄存器	7EFBF8H	HSSPI_SS_HLD[3:0]				HSSPI_SS_SETUP[3:0]				0011,0011
HSSPI_CFG2	高速 SPI 配置寄存器 2	7EFBF9H	MOSI_PL	SPI_IOSW	HSSPI_EN	HSSPI_FIFOEN	HSSPI_SS_DACT[3:0]				0000,0011
HSSPI_STA	高速 SPI 状态寄存器	7EFBFAH	-	-	-	-	TXFULL	TXEMPT	RXFULL	RXEMPT	xxxx,0000
HSSPI_PSCR	高速 SPI 时钟分配器	7EFBFBH	HSSPI_PSCR [7:0]								0000,0000
SPI_CLKDIV	SPI 时钟预分配寄存器	7EF808H	SPI_CLKDIV[7:0]								0000,0000

25.1.4 SPI 状态寄存器（SPSTAT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPSTAT	CDH	SPIF	WCOL	-	-	-	-	-	-

SPIF：SPI 中断标志位。

当发送/接收完成 1 字节的数据后，硬件自动将此位置 1，并向 CPU 提出中断请求。当 SSIG 位被设置为 0 时，由于 SS 管脚电平的变化而使得设备的主/从模式发生改变时，此标志位也会被硬件自动置 1，以标志设备模式发生变化。

注意：此标志位必须用户通过软件方式向此位写 1 进行清零。

WCOL：SPI 写冲突标志位。

当 SPI 在进行数据传输的过程中写 SPDAT 寄存器时，硬件将此位置 1。

注意：此标志位必须用户通过软件方式向此位写 1 进行清零。

25.1.5 SPI 控制寄存器（SPCTL），SPI 速度控制

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPCTL	CEH	SSIG	SPEN	DORD	MSTR	CPOL	CPHA	SPR[1:0]	

SSIG：SS 引脚功能控制位

0：SS 引脚确定器件是主机还是从机

1：忽略 SS 引脚功能，使用 MSTR 确定器件是主机还是从机

SPEN：SPI 使能控制位

0：关闭 SPI 功能

1：使能 SPI 功能

DORD：SPI 数据位发送/接收的顺序

0：先发送/接收数据的高位（MSB）

1：先发送/接收数据的低位（LSB）

MSTR：器件主/从模式选择位

设置主机模式：

若 SSIG＝0，则 SS 管脚必须为高电平且设置 MSTR 为 1

若 SSIG＝1，则只需要设置 MSTR 为 1（忽略 SS 管脚的电平）

设置从机模式：

若 SSIG＝0，则 SS 管脚必须为低电平（与 MSTR 位无关）

若 SSIG＝1，则只需要设置 MSTR 为 0（忽略 SS 管脚的电平）

CPOL：SPI 时钟极性控制

0：SCLK 空闲时为低电平，SCLK 的前时钟沿为上升沿，后时钟沿为下降沿

1：SCLK 空闲时为高电平，SCLK 的前时钟沿为下降沿，后时钟沿为上升沿

CPHA：SPI 时钟相位控制

0：数据 SS 管脚为低电平驱动第一位数据并在 SCLK 的后时钟沿改变数据，前时钟沿采样数据（必须 SSIG＝0）

1：数据在 SCLK 的前时钟沿驱动，后时钟沿采样

SPR[1:0]：SPI 时钟频率选择

SPR[1:0]	SCLK 频率
00	SPI 输入时钟/4
01	SPI 输入时钟/8
10	SPI 输入时钟/16
11	SPI 输入时钟/2

25.1.6 SPI 数据寄存器（SPDAT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPDAT	CFH								

SPI 发送/接收数据缓冲器。

25.1.7 SPI 从机超时控制寄存器（SPITOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPITOCR	7EFD80H	SPIENTO	SPIENTOI	SPISCALE	-	-	-	-	-

SPIENTO：SPI 从机超时功能控制位

0：禁止 SPI 从机超时功能

1：使能 SPI 从机超时功能（**注：SPI 的主机模式无接收超时功能**）

SPIENTOI：SPI 从机超时中断控制位

0：禁止 SPI 从机超时中断

1：使能 SPI 从机超时中断

SPISCALE：SPI 从机计数时钟源选择

0：1us 时钟（1MHz 时钟）。（注意：如需要使用此时钟源，用户必须先正确设置 IAP_TPS 寄存器。例如：若系统时钟为 12MHz，则需要将 IAP_TPS 设置为 12；若系统时钟为 22.1184MHz，则需要将 IAP_TPS 设置为 22；其他频率以此类推。特别注意，此 1us 的时钟并不是精准时间）

1：系统时钟

25.1.8 SPI 从机超时状态寄存器（SPITOSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPITOSR	7EFD81H	SPICTOIF	-	-	-	-	-	-	SPITOIF

SPICTOIF：写“1”清除 SPI 超时中断标志位 TOIF。（只写）

SPITOIF：SPI 超时中断请求标志位。（只读）

当发生 SPI 超时时，TOIF 会被硬件置“1”。如果 ENTOI 为 1 时会产生 SPI 中断，中断入口地址为原 SPI 的中断入口地址。标志位需要软件向 CTOIF 位写“1”清零

25.1.9 SPI 从机超时长度控制寄存器（SPITOTE/H/L）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPITOTE	7EFD8CH	SPITM[23:16]							
SPITOTH	7EFD82H	SPITM[15:8]							
SPITOTL	7EFD83H	SPITM[7:0]							

SPITM[23:0]：SPI 超时时间控制位。

当 SPI 处于接收空闲状态时，内部超时计数器根据 SCALE 寄存器所选择的时钟源进行计数，当计数时间达到 TM 所设置的超时时间时，便会产生超时中断。当 SPI 接收数据完成时，复位内部超时计数器，重新进行超时计数。

注：

1、如果需要使能接收超时中断功能，则 TM 不可设置为 0，且 SPITOTL、SPITOTH、SPITOTE 寄存器的设置必须先设置 SPITOTL 和 SPITOTH，最后设置 SPITOTE

2、必须设置 SPI 为从机模式才有接收超时功能

3、接收超时功能必须在正确接收到一字节数据后才能触发

4、正在接收过程中，无接收超时功能

25.1.10 高速 SPI 配置寄存器（HSSPI_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI_CFG	7EFBF8H	HSSPI_SS_HLD[3:0]				HSSPI_SS_SETUP[3:0]			

HSSPI_SS_HLD[3:0]：高速模式时 SS 控制信号的 HOLD 时间

HSSPI_SS_SETUP[3:0]：高速模式时 SS 控制信号的 SETUP 时间

25.1.11 高速 SPI 配置寄存器 2（HSSPI_CFG2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI_CFG2	7EFBF9H	MOSI_PL	SPI_IOSW	HSSPI_EN	HSSPI_FIFOEN	HSSPI_SS_DACT[3:0]			

MOSI_PL：配置 IDLE 状态时 MOSI 管脚的电平

0：IDLE 状态时，MOSI 维持低电平。

1：IDLE 状态时，MOSI 维持高电平。

SPI_IOSW：交换 MOSI 和 MISO 脚位（普通 SPI 模式和高速 SPI 模式通用）

0：不交换，维持上电默认脚位。

1：交换 MOSI 和 MISO 的脚位。

HSSPI_EN：高速 SPI 使能位

0：关闭高速模式。

1：使能高速模式。

当 SPI 速度为系统时钟/2（SPCTL.SPR＝3）时，必须设置使能高速模式

SPI+DMA 读取数据时，如果开启了 FIFO，也必须开启高速 SPI 模式

HSSPI_FIFOEN：高速 SPI 的 FIFO 模式使能位

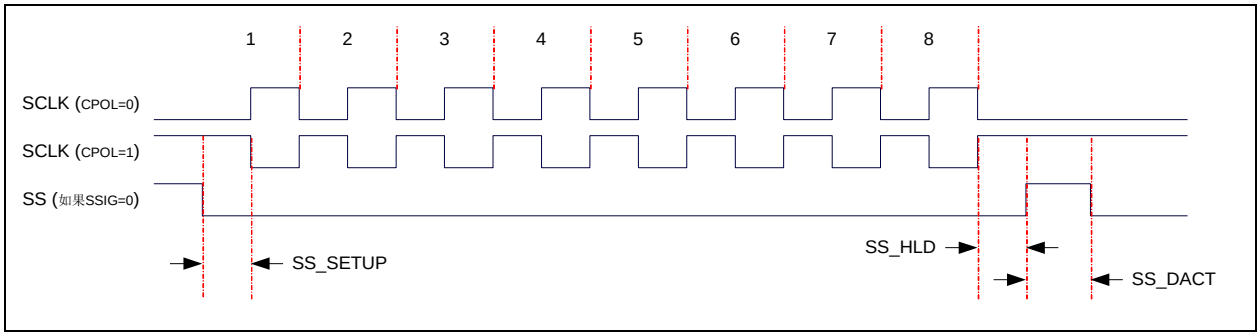
0：关闭 FIFO 模式。

1：使能 FIFO 模式。

SPI 发送和接收的 FIFO 深度均为 4 字节。设置 FIFOEN 可以减少 SPI 数据收发间隔时间，提高 SPI 收发速度。FIFOEN 只有在使用 SPI+DMA 传输时才可以使能，使用普通 SPI 收发时需要关闭 FIFOEN 才能收到 SPIF 标志。

HSSPI_SS_DACT[3:0]：高速模式时 SS 控制信号的 DEACTIVE 时间

注：HSSPI_SS_HLD、HSSPI_SS_SETUP 和 HSSPI_SS_DACT 所设置的值只有在 SPI 主机模式的 DMA 才有意义，只有 SPI 在主机模式时，执行 DMA 数据传输时才需要硬件自动输出 SS 控制信号。当 SPI 速度为系统时钟/2（SPCTL.SPR＝3）时执行 DMA，HSSPI_SS_HLD、HSSPI_SS_SETUP 和 HSSPI_SS_DACT 都必须设置为大于 2 的值。



25.1.12 高速 SPI 状态寄存器（HSSPI_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI_STA	7EFBFAH	-	-	-	-	TXFULL	TXEMPT	RXFULL	RXEMPT

TXFULL：发送数据 FIFO 满标志（只读）
TXEMPT：发送数据 FIFO 空标志（只读）
RXFULL：接收数据 FIFO 满标志（只读）
RXEMPT：接收数据 FIFO 空标志（只读）

25.1.13 高速 SPI 时钟分频器（HSSPI_PSCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI_PSCR	7EFBFBH	HSSPI_PSCR[7:0]							

HSSPI_PSCR[7:0]：高速 SPI 时钟分频器
高速 SPI 工作时钟＝高速 SPI 输入时钟 / (HSSPI_PSCR[7:0] + 1)

25.1.14 SPI 时钟分频寄存器（SPI_CLKDIV）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI_CLKDIV	7EFE90H	SPI_CLKDIV[7:0]							

SPI_CLKDIV：高速 SPI 时钟分频系数。

SPI_CLKDIV	SPI 工作时钟频率
0	HSIOCLK /1
1	HSIOCLK /1
2	HSIOCLK /2
…	…
255	HSIOCLK /255

25.1.15 如何提高 SPI+DMA 的传输效率

SPI 使用 DMA 传输数据时，可通过减少字节之间的间隔时间来提高 SPI+DMA 的传输效率。进行如下的设置可减少 DMA 字节之间的间隔时间：

1、配置 HSSPI_CFG 寄存器中的 HSSPI_SS_HLD 和 HSSPI_SS_SETUP，减少 SS 信号的 HOLD 时间和 SETUP 时间；

2、设置 HSSPI_CFG2 寄存器中的 HSSPI_FIFOEN 位，开启高速 SPI 的 FIFO 模式；

（上面的两点设置对于 SPI+DMA 和高速 SPI+DMA 均适用）

3、对于 STC32G144K246 型号，增加额外增加了 DMA_SPI_ITVH 和 DMA_SPI_ITVL 寄存器来控制 DMA 时字节间隔时间，上电默认值位 3，可软件将这两个寄存器设置位 0，可降低间隔时间。

25.2 SPI2

25.2.1 SPI2 的 MOSI2 和 MISO2 脚交换控制

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI2_CFG2	7EF80AH	MOSI2_PL	**SPI2_IOSW**	HSSPI2_EN	HSSPI2_FIFOEN	SS2_DACT[3:0]			

SPI2_IOSW：交换 MOSI2 和 MISO2 脚位（普通 SPI2 模式和高速 SPI2 模式通用）

0：不交换，维持上电默认脚位。

1：交换 MOSI2 和 MISO2 的脚位。

25.2.2 SPI2 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX4	7EFD6CH	I2C2_S[1:]		-	-	SPI3_S[1:0]		**SPI2_S[1:0]**	

SPI2_S[1:0]：SPI2 功能脚选择位

SPI2_S[1:0]	SS2	MOSI2	MISO2	SCLK2
00	P6.4	P6.5	P6.6	P6.7
01	P0.2	P0.3	P0.4	P0.5
10	PB.1	PB.2	PB.3	PB.4
11	P8.0	P8.1	P8.2	P8.3

25.2.3 SPI2 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
SPI2CTL	SPI2 控制寄存器	7EF800H	SS2IG	SP2EN	SP2DORD	SP2MSTR	SP2CPOL	SP2CPHA	SP2R[1:0]		0000,1100
SPI2STAT	SPI2 状态寄存器	7EF801H	SP2IF	SP2WCOL	-	-	-	-	-	-	00xx,xxxx
SPI2DAT	SPI2 数据寄存器	7EF802H	SP2DAT[7:0]								0000,0000
SPI2TOTE	SPI2 从机超时长度寄存器	7EF803H	SPI2TM[23:16]								0000,0000
SPI2TOCR	SPI2 从机超时控制寄存器	7EF804H	SPI2ENTO	SPI2ENTOI	SPI2SCALE	-	-	-	-	-	000x,xxxx
SPI2TOSR	SPI2 从机超时状态寄存器	7EF805H	-	-	-	-	-	-	-	SPI2TOIF	xxxx,xxx0
SPI2TOTH	SPI2 从机超时长度寄存器	7EF806H	SPI2TM[15:8]								0000,0000
SPI2TOTL	SPI2 从机超时长度寄存器	7EF807H	SPI2TM[7:0]								0000,0000
SPI2_CLKDIV	SPI2 时钟预分配寄存器	7EF808H	SPI2_CLKDIV[7:0]								0000,0000
HSSPI2_CFG	高速 SPI2 配置寄存器	7EF809H	SS2_HLD[3:0]				SS2_SETUP[3:0]				0011,0011
HSSPI2_CFG2	高速 SPI2 配置寄存器 2	7EF80AH	MOSI2_PL	SPI2_IOSW	HSSPI2_EN	HSSPI2_FIFOEN	SS2_DACT[3:0]				0000,0011
HSSPI2_STA	高速 SPI2 状态寄存器	7EF80BH	-	-	-	-	TXFULL	TXEMPT	RXFULL	RXEMPT	xxxx,0000
HSSPI2_PSCR	高速 SPI2 时钟分频寄存器	7EF80CH	HSSPI2_PSCR [7:0]								0000,0000
SPI2_INTR	SPI2 中断控制寄存器	7EF80FH	-	-	PSPI2	-	-	-	ESPI2	xx00,xxx0	

25.2.4 SPI2 状态寄存器（SPI2STAT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI2STAT	7EF801H	SPI2IF	SP2WCOL	-	-	-	-	-	-

SPI2IF：SPI2 中断标志位。

当发送/接收完成 1 字节的数据后，硬件自动将此位置 1，并向 CPU 提出中断请求。当 SS2IG 位被设置为 0 时，由于 SS2 管脚电平的变化而使得设备的主/从模式发生改变时，此标志位也会被硬件自动置 1，以标志设备模式发生变化。

注意：此标志位必须用户通过软件方式向此位写 1 进行清零。

SP2WCOL：SPI2 写冲突标志位。

当 SPI2 在进行数据传输的过程中写 SP2DAT 寄存器时，硬件将此位置 1。

注意：此标志位必须用户通过软件方式向此位写 1 进行清零。

25.2.5 SPI2 控制寄存器（SPI2CTL），SPI2 速度控制

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI2CTL	7EF800H	SS2IG	SP2EN	SP2DORD	SP2MSTR	SP2CPOL	SP2CPHA	SP2R[1:0]	

SS2IG：SS2 引脚功能控制位

0：SS2 引脚确定器件是主机还是从机

1：忽略 SS2 引脚功能，使用 SP2MSTR 确定器件是主机还是从机

SP2EN：SPI2 使能控制位

0：关闭 SPI2 功能

1：使能 SPI2 功能

SP2DORD：SPI2 数据位发送/接收的顺序

0：先发送/接收数据的高位（MSB）

1：先发送/接收数据的低位（LSB）

SP2MSTR：器件主/从模式选择位

设置主机模式：

若 SS2IG＝0，则 SS2 管脚必须为高电平且设置 SP2MSTR 为 1

若 SS2IG＝1，则只需要设置 SP2MSTR 为 1（忽略 SS2 管脚的电平）

设置从机模式：

若 SS2IG＝0，则 SS2 管脚必须为低电平（与 SP2MSTR 位无关）

若 SS2IG＝1，则只需要设置 SP2MSTR 为 0（忽略 SS2 管脚的电平）

SP2CPOL：SPI2 时钟极性控制

0：SCLK2 空闲时为低电平，SCLK2 的前时钟沿为上升沿，后时钟沿为下降沿

1：SCLK2 空闲时为高电平，SCLK2 的前时钟沿为下降沿，后时钟沿为上升沿

SP2CPHA：SPI2 时钟相位控制

0：数据 SS2 管脚为低电平驱动第一位数据并在 SCLK2 的后时钟沿改变数据，前时钟沿采样数据（必须 SS2IG＝0）

1：数据在 SCLK2 的前时钟沿驱动，后时钟沿采样

SP2R[1:0]：SPI2 时钟频率选择

SP2R[1:0]	SCLK2 频率
00	SPI2 输入时钟/4
01	SPI2 输入时钟/8
10	SPI2 输入时钟/16
11	SPI2 输入时钟/2

25.2.6 SPI2 数据寄存器（SP2DAT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SP2DAT	7EF802H								

SPI2 发送/接收数据缓冲器。

25.2.7 SPI2 中断控制寄存器（SPI2_INTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI2_INTR	7EF80FH	-	-	PSPI2[1:0]		-	-	-	ESPI2

ESPI2：SPI2 中断使能控制位

0：禁止 SPI2 中断

1：允许 SPI2 中断

PSPI2[1:0]：SPI2 中断优先级控制位

00：SPI2 中断优先级为 0 级（最低级）

01：SPI2 中断优先级为 1 级（较低级）

10：SPI2 中断优先级为 2 级（较高级）

11：SPI2 中断优先级为 3 级（最高级）

25.2.8 SPI2 从机超时控制寄存器（SPI2TOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI2TOCR	7EF804H	SPI2ENTO	SPI2ENTOI	SPI2SCALE	-	-	-	-	-

SPI2ENTO：SPI2 从机超时功能控制位

0：禁止 SPI2 从机超时功能

1：使能 SPI2 从机超时功能（**注：SPI2 的主机模式无接收超时功能**）

SPI2ENTOI：SPI2 从机超时中断控制位

0：禁止 SPI2 从机超时中断

1：使能 SPI2 从机超时中断

SPI2SCALE：SPI2 从机计数时钟源选择

0：1us 时钟（1MHz 时钟）。（注意：如需要使用此时钟源，用户必须先正确设置 IAP_TPS 寄存器。例如：若系统时钟为 12MHz，则需要将 IAP_TPS 设置为 12；若系统时钟为 22.1184MHz，则需要将 IAP_TPS 设置为 22；其他频率以此类推。特别注意，此 1us 的时钟并不是精准时间）

1：系统时钟

25.2.9 SPI2 从机超时状态寄存器（SPI2TOSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI2TOSR	7EF805H	SPI2CTOIF	-	-	-	-	-	-	SPI2TOIF

SPI2CTOIF：写“1”清除 SPI2 超时中断标志位 SPI2TOIF。（只写）

SPI2TOIF：SPI2 超时中断请求标志位。（只读）

当发生 SPI2 超时时，SPI2TOIF 会被硬件置“1”。如果 SPI2ENTOI 为 1 时会产生 SPI2 中断，中断入口地址为原 SPI2 的中断入口地址。标志位需要软件向 SPI2CTOIF 位写“1”清零

25.2.10 SPI2 从机超时长度控制寄存器（SPI2TOTE/H/L）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI2TOTE	7EF803H	SPI2TM[23:16]							
SPI2TOTH	7EF806H	SPI2TM[15:8]							
SPI2TOTL	7EF807H	SPI2TM[7:0]							

SPI2TM[23:0]：SPI2 超时时间控制位。

当 SPI2 处于接收空闲状态时，内部超时计数器根据 SCALE 寄存器所选择的时钟源进行计数，当计数时间达到 TM 所设置的超时时间时，便会产生超时中断。当 SPI2 接收数据完成时，复位内部超时计数器，重新进行超时计数。

注：

1、如果需要使能接收超时中断功能，则 TM 不可设置为 0，且 SPI2TOTL、SPI2TOTH、SPI2TOTE 寄存器的设置必须先设置 SPI2TOTL 和 SPI2TOTH，最后设置 SPI2TOTE

2、必须设置 SPI2 为从机模式才有接收超时功能

3、接收超时功能必须在正确接收到一字节数据后才能触发

4、正在接收过程中，无接收超时功能

25.2.11 高速 SPI2 配置寄存器（HSSPI2_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI2_CFG	7EF809H	SS2_HLD[3:0]				SS2_SETUP[3:0]			

SS2_HLD[3:0]：高速模式时 SS2 控制信号的 HOLD 时间

SS2_SETUP[3:0]：高速模式时 SS2 控制信号的 SETUP 时间

25.2.12 高速 SPI2 配置寄存器 2（HSSPI2_CFG2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI2_CFG2	7EF80AH	MOSI2_PL	SPI2_IOSW	HSSPI2_EN	HSSPI2_FIFOEN	SS2_DACT[3:0]			

MOSI2_PL：配置 IDLE 状态时 MOSI2 管脚的电平

0：IDLE 状态时，MOSI2 维持低电平。

1：IDLE 状态时，MOSI2 维持高电平。

SPI2_IOSW：交换 MOSI2 和 MISO2 脚位（普通 SPI2 模式和高速 SPI2 模式通用）

0：不交换，维持上电默认脚位。

1：交换 MOSI2 和 MISO2 的脚位。

HSSPI2_EN：高速 SPI2 使能位

0：关闭高速模式。

1：使能高速模式。

当 SPI2 速度为系统时钟/2（SP2CTL.SP2R＝3）时，必须设置使能高速模式

SPI2+DMA 读取数据时，如果开启了 FIFO，也必须开启高速 SPI2 模式

HSSPI2_FIFOEN：高速 SPI2 的 FIFO 模式使能位

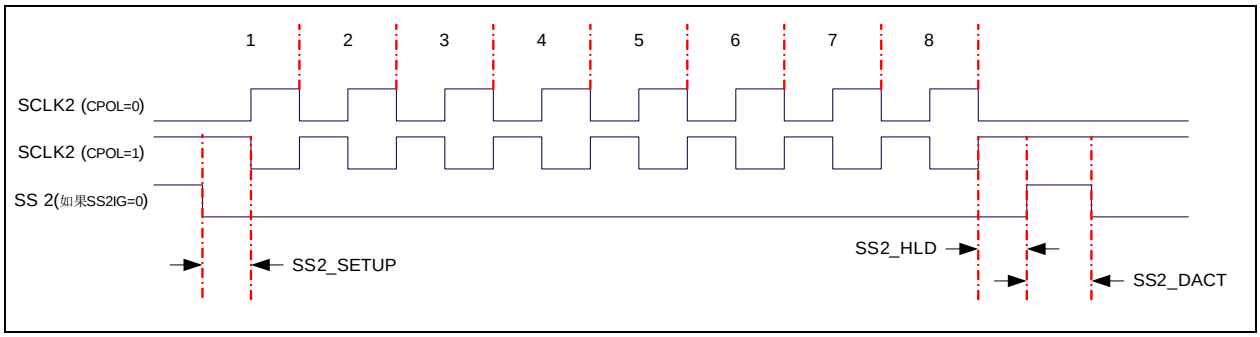
0：关闭 FIFO 模式。

1：使能 FIFO 模式。

SPI2 发送和接收的 FIFO 深度均为 4 字节。设置 HSSPI2_FIFOEN 可以减少 SPI2 数据收发间隔时间，提高 SPI2 收发速度。HSSPI2_FIFOEN 只有在使用 SPI2+DMA 传输时才可以使能，使用普通 SPI2 收发时需要关闭 HSSPI2_FIFOEN 才能收到 SPI2F 标志。

SS2_DACT[3:0]：高速模式时 SS2 控制信号的 DEACTIVE 时间

注：SS2_HLD、SS2_SETUP 和 SS2_DACT 所设置的值只有在 SPI2 主机模式的 DMA 才有意义，只有 SPI2 在主机模式时，执行 DMA 数据传输时才需要硬件自动输出 SS 控制信号。当 SPI2 速度为系统时钟/2（SPCTL.SPR＝3）时执行 DMA，SS2_HLD、SS2_SETUP 和 SS2_DACT 都必须设置为大于 2 的值。



25.2.13 高速 SPI2 状态寄存器（HSSPI2_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI2_STA	7EF80BH	-	-	-	-	TXFULL	TXEMPT	RXFULL	RXEMPT

TXFULL：发送数据 FIFO 满标志（只读）
TXEMPT：发送数据 FIFO 空标志（只读）
RXFULL：接收数据 FIFO 满标志（只读）
RXEMPT：接收数据 FIFO 空标志（只读）

25.2.14 高速 SPI2 时钟分频器（HSSPI2_PSCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI2_PSCR	7EF80CH	HSSPI2_PSCR[7:0]							

HSSPI2_PSCR[7:0]：高速 SPI2 时钟分频器
高速 SPI2 工作时钟＝高速 SPI2 输入时钟 / (HSSPI2_PSCR[7:0] + 1)

25.2.15 SPI2 时钟分频寄存器（SPI2_CLKDIV）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI2_CLKDIV	7EF808H	SPI2_CLKDIV[7:0]							

SPI2_CLKDIV：高速 SPI 时钟分频系数。

SPI2_CLKDIV	SPI2 工作时钟频率
0	HSIOCLK /1
1	HSIOCLK /1
2	HSIOCLK /2
…	…
255	HSIOCLK /255

25.3 SPI3

25.3.1 SPI3 的 MOSI3 和 MISO3 脚交换控制

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI3_CFG2	7EF82AH	MOSI3_PL	**SPI3_IOSW**	HSSPI3_EN	HSSPI3_FIFOEN	SS3_DACT[3:0]			

SPI3_IOSW：交换 MOSI3 和 MISO3 脚位（普通 SPI3 模式和高速 SPI3 模式通用）

0：不交换，维持上电默认脚位。

1：交换 MOSI3 和 MISO3 的脚位。

25.3.2 SPI3 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX4	7EFD6CH	I2C2_S[1:]		-	-	**SPI3_S[1:0]**		SPI2_S[1:0]	

SPI3_S[1:0]：SPI3 功能脚选择位

SPI3_S[1:0]	SS3	MOSI3	MISO3	SCLK3
00	P2.2	P2.3	P2.4	P2.5
01	P7.0	P7.1	P7.2	P7.3
10	P8.4	P8.5	P8.6	P8.7
11	P9.0	P9.1	P9.2	P9.3

25.3.3 SPI3 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
SPI3CTL	SPI3 控制寄存器	7EF820H	SS3IG	SP3EN	SP3DORD	SP3MSTR	SP3CPOL	SP3CPHA	SP3R[1:0]		0000,1100
SPI3STAT	SPI3 状态寄存器	7EF821H	SP3IF	SP3WCOL	-	-	-	-	-	-	00xx,xxxx
SPI3DAT	SPI3 数据寄存器	7EF822H	SP3DAT[7:0]								0000,0000
SPI3TOTE	SPI3 从机超时长度寄存器	7EF823H	SPI3TM[23:16]								0000,0000
SPI3TOCR	SPI3 从机超时控制寄存器	7EF824H	SPI3ENTO	SPI3ENTOI	SPI3SCALE	-	-	-	-	-	000x,xxxx
SPI3TOSR	SPI3 从机超时状态寄存器	7EF825H	-	-	-	-	-	-	-	SPI3TOIF	xxxx,xxx0
SPI3TOTH	SPI3 从机超时长度寄存器	7EF826H	SPI3TM[15:8]								0000,0000
SPI3TOTL	SPI3 从机超时长度寄存器	7EF827H	SPI3TM[7:0]								0000,0000
SPI3_CLKDIV	SPI3 时钟预分配寄存器	7EF828H	SPI3_CLKDIV[7:0]								0000,0000
HSSPI3_CFG	高速 SPI3 配置寄存器	7EF829H	SS3_HLD[3:0]				SS3_SETUP[3:0]				0011,0011
HSSPI3_CFG2	高速 SPI3 配置寄存器 2	7EF82AH	MOSI3_PL	SPI3_IOSW	HSSPI3_EN	HSSPI3_FIFOEN	SS3_DACT[3:0]				0000,0011
HSSPI3_STA	高速 SPI3 状态寄存器	7EF82BH	-	-	-	-	TXFULL	TXEMPT	RXFULL	RXEMPT	xxxx,0000
HSSPI3_PSCR	高速 SPI3 时钟分频寄存器	7EF82CH	HSSPI3_PSCR [7:0]								0000,0000
SPI3_INTR	SPI3 中断控制寄存器	7EF82FH	-	-	PSPI3	-	-	-	ESPI3	xx00,xxx0	

25.3.4 SPI3 状态寄存器（SPSTAT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI3STAT	7EF821H	SP3IF	SP3WCOL	-	-	-	-	-	-

SP3IF：SPI3 中断标志位。

当发送/接收完成 1 字节的数据后，硬件自动将此位置 1，并向 CPU 提出中断请求。当 SS3IG 位被设置为 0 时，由于 SS3 管脚电平的变化而使得设备的主/从模式发生改变时，此标志位也会被硬件自动置 1，以标志设备模式发生变化。

注意：此标志位必须用户通过软件方式向此位写 1 进行清零。

SP3WCOL：SPI3 写冲突标志位。

当 SPI3 在进行数据传输的过程中写 SP3DAT 寄存器时，硬件将此位置 1。

注意：此标志位必须用户通过软件方式向此位写 1 进行清零。

25.3.5 SPI3 控制寄存器（SPCTL），SPI3 速度控制

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SP3CTL	7EF820H	SS3IG	SP3EN	SP3DORD	SP3MSTR	SP3CPOL	SP3CPHA	SP3R[1:0]	

SS3IG：SS3 引脚功能控制位

0：SS3 引脚确定器件是主机还是从机

1：忽略 SS3 引脚功能，使用 SP3MSTR 确定器件是主机还是从机

SP2EN：SPI3 使能控制位

0：关闭 SPI3 功能

1：使能 SPI3 功能

SP3DORD：SPI3 数据位发送/接收的顺序

0：先发送/接收数据的高位（MSB）

1：先发送/接收数据的低位（LSB）

SP3MSTR：器件主/从模式选择位

设置主机模式：

若 SS3IG＝0，则 SS3 管脚必须为高电平且设置 SP3MSTR 为 1

若 SS3IG＝1，则只需要设置 SP3MSTR 为 1（忽略 SS3 管脚的电平）

设置从机模式：

若 SS3IG＝0，则 SS 管脚必须为低电平（与 SP3MSTR 位无关）

若 SS3IG＝1，则只需要设置 SP3MSTR 为 0（忽略 SS3 管脚的电平）

SP3CPOL：SPI3 时钟极性控制

0：SCLK3 空闲时为低电平，SCLK3 的前时钟沿为上升沿，后时钟沿为下降沿

1：SCLK3 空闲时为高电平，SCLK3 的前时钟沿为下降沿，后时钟沿为上升沿

SP3CPHA：SPI3 时钟相位控制

0：数据 SS3 管脚为低电平驱动第一位数据并在 SCLK3 的后时钟沿改变数据，前时钟沿采样数据（必须 SS3IG＝0）

1：数据在 SCLK3 的前时钟沿驱动，后时钟沿采样

SP3R[1:0]：SPI3 时钟频率选择

SP3R[1:0]	SCLK3 频率
00	SPI3 输入时钟/4
01	SPI3 输入时钟/8
10	SPI3 输入时钟/16
11	SPI3 输入时钟/2

25.3.6 SPI3 数据寄存器（SPDAT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SP3DAT	7EF822H								

SPI3 发送/接收数据缓冲器。

25.3.7 SPI3 中断控制寄存器（SPI3_INTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI3_INTR	7EF82FH	-	-	PSPI3[1:0]		-	-	-	ESPI3

ESPI3：SPI3 中断使能控制位

0：禁止 SPI3 中断

1：允许 SPI3 中断

PSPI3[1:0]：SPI3 中断优先级控制位

00：SPI3 中断优先级为 0 级（最低级）

01：SPI3 中断优先级为 1 级（较低级）

10：SPI3 中断优先级为 2 级（较高级）

11：SPI3 中断优先级为 3 级（最高级）

25.3.8 SPI3 从机超时控制寄存器（SPI3TOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI3TOCR	7EF824H	SPI3ENTO	SPI3ENTOI	SPI3SCALE	-	-	-	-	-

SPI3ENTO：SPI3 从机超时功能控制位

0：禁止 SPI3 从机超时功能

1：使能 SPI3 从机超时功能（**注：SPI3 的主机模式无接收超时功能**）

SPI3ENTOI：SPI3 从机超时中断控制位

0：禁止 SPI3 从机超时中断

1：使能 SPI3 从机超时中断

SPI3SCALE：SPI3 从机计数时钟源选择

0：1us 时钟（1MHz 时钟）。（注意：如需要使用此时钟源，用户必须先正确设置 IAP_TPS 寄存器。例如：若系统时钟为 12MHz，则需要将 IAP_TPS 设置为 12；若系统时钟为 22.1184MHz，则需要将 IAP_TPS 设置为 22；其他频率以此类推。特别注意，此 1us 的时钟并不是精准时间）

1：系统时钟

25.3.9 SPI3 从机超时状态寄存器（SPI3TOSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI3TOSR	7EF825H	SPI3CTOIF	-	-	-	-	-	-	SPI3TOIF

SPI3CTOIF：写“1”清除 SPI3 超时中断标志位 SP3TOIF。（只写）

SPI3TOIF：SPI3 超时中断请求标志位。（只读）

当发生 SPI3 超时时，SP3TOIF 会被硬件置“1”。如果 SP3ENTOI 为 1 时会产生 SPI3 中断，中断入口地址为原 SPI3 的中断入口地址。标志位需要软件向 SP3CTOIF 位写“1”清零

25.3.10 SPI3 从机超时长度控制寄存器（SPI3TOTE/H/L）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI3TOTE	7EF823H	SPI3TM[23:16]							
SPI3TOTH	7EF826H	SPI3TM[15:8]							
SPI3TOTL	7EF827H	SPI3TM[7:0]							

SPI3TM[23:0]：SPI3 超时时间控制位。

当 SPI3 处于接收空闲状态时，内部超时计数器根据 SCALE 寄存器所选择的时钟源进行计数，当计数时间达到 SPI3TM 所设置的超时时间时，便会产生超时中断。当 SPI3 接收数据完成时，复位内部超时计数器，重新进行超时计数。

注：

1、如果需要使能接收超时中断功能，则 TM 不可设置为 0，且 SPI3TOTL、SPI3TOTH、SPI3TOTE 寄存器的设置必须先设置 SPI3TOTL 和 SPI3TOTH，最后设置 SPI3TOTE

2、必须设置 SPI3 为从机模式才有接收超时功能

3、接收超时功能必须在正确接收到一字节数据后才能触发

4、正在接收过程中，无接收超时功能

25.3.11 高速 SPI3 配置寄存器（HSSPI3_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI3_CFG	7EF829H	SS3_HLD[3:0]				SS3_SETUP[3:0]			

SS3_HLD[3:0]：高速模式时 SS3 控制信号的 HOLD 时间

SS3_SETUP[3:0]：高速模式时 SS3 控制信号的 SETUP 时间

25.3.12 高速 SPI3 配置寄存器 2（HSSPI3_CFG2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI3_CFG2	7EF82AH	MOSI3_PL	SPI3_IOSW	HSSPI3_EN	HSSPI3_FIFOEN	SS3_DACT[3:0]			

MOSI3_PL：配置 IDLE 状态时 MOSI3 管脚的电平

0：IDLE 状态时，MOSI3 维持低电平。

1：IDLE 状态时，MOSI3 维持高电平。

SPI3_IOSW：交换 MOSI3 和 MISO3 脚位（普通 SPI3 模式和高速 SPI3 模式通用）

0：不交换，维持上电默认脚位。

1：交换 MOSI3 和 MISO3 的脚位。

HSSPI3_EN：高速 SPI3 使能位

0：关闭高速模式。

1：使能高速模式。

当 SPI3 速度为系统时钟/2（SP3CTL.SP3R＝3）时，必须设置使能高速模式

SPI3+DMA 读取数据时，如果开启了 FIFO，也必须开启高速 SPI3 模式

HSSPI3_FIFOEN：高速 SPI3 的 FIFO 模式使能位

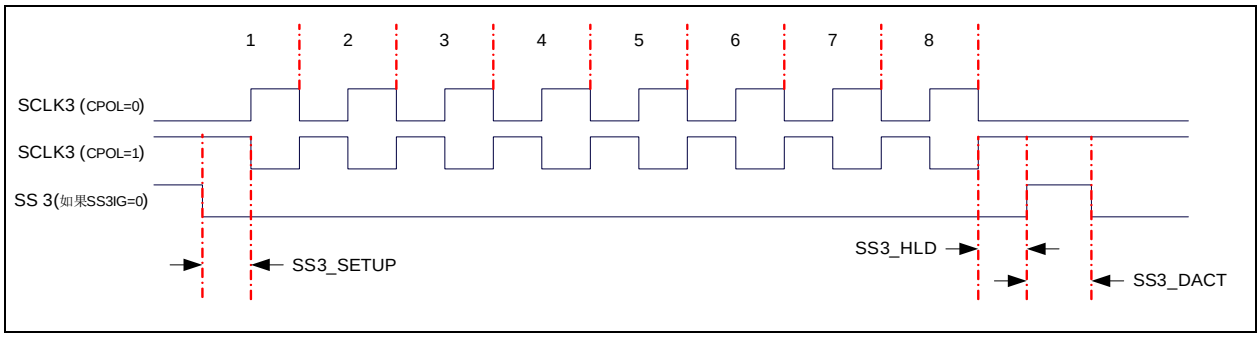
0：关闭 FIFO 模式。

1：使能 FIFO 模式。

SPI3 发送和接收的 FIFO 深度均为 4 字节。设置 HSSPI3_FIFOEN 可以减少 SPI3 数据收发间隔时间，提高 SPI3 收发速度。HSSPI3_FIFOEN 只有在使用 SPI3+DMA 传输时才可以使能，使用普通 SPI3 收发时需要关闭 HSSPI3_FIFOEN 才能收到 SPI3F 标志。

SS3_DACT[3:0]：高速模式时 SS3 控制信号的 DEACTIVE 时间

注：SS3_HLD、SS3_SETUP 和 SS3_DACT 所设置的值只有在 SPI3 主机模式的 DMA 才有意义，只有 SPI3 在主机模式时，执行 DMA 数据传输时才需要硬件自动输出 SS 控制信号。当 SPI3 速度为系统时钟/2（SPCTL.SPR＝3）时执行 DMA，SS3_HLD、SS3_SETUP 和 SS3_DACT 都必须设置为大于 2 的值。



25.3.13 高速 SPI3 状态寄存器（HSSPI3_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI3_STA	7EF82BH	-	-	-	-	TXFULL	TXEMPT	RXFULL	RXEMPT

TXFULL：发送数据 FIFO 满标志（只读）

TXEMPT：发送数据 FIFO 空标志（只读）

RXFULL：接收数据 FIFO 满标志（只读）

RXEMPT：接收数据 FIFO 空标志（只读）

25.3.14 高速 SPI3 时钟分频器（HSSPI3_PSCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSSPI3_PSCR	7EF82CH	HSSPI3_PSCR[7:0]							

HSSPI3_PSCR[7:0]：高速 SPI3 时钟分频器

高速 SPI3 工作时钟＝高速 SPI3 输入时钟 / (HSSPI3_PSCR[7:0] + 1)

25.3.15 SPI3 时钟分频寄存器（SPI3_CLKDIV）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
SPI3_CLKDIV	7EF828H	SPI3_CLKDIV[7:0]							

SPI3_CLKDIV：高速 SPI 时钟分频系数。

SPI3_CLKDIV	SPI3 工作时钟频率
0	HSIOCLK /1
1	HSIOCLK /1
2	HSIOCLK /2
…	…
255	HSIOCLK /255

25.4 SPI 通信方式

SPI 的通信方式通常有 3 种：单主单从（一个主机设备连接一个从机设备）、互为主从（两个设备连接，设备和互为主机和从机）、单主多从（一个主机设备连接多个从机设备）

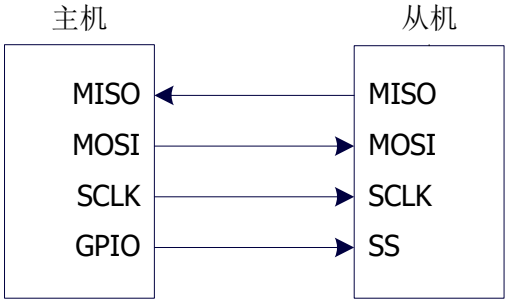
25.4.1 单主单从

两个设备相连，其中一个设备固定作为主机，另外一个固定作为从机。

主机设置：SSIG 设置为 1，MSTR 设置为 1，固定为主机模式。主机可以使用任意端口连接从机的 SS 管脚，拉低从机的 SS 脚即可使能从机

从机设置：SSIG 设置为 0，SS 管脚作为从机的片选信号。

单主单从连接配置图如下所示：



单主单从配置

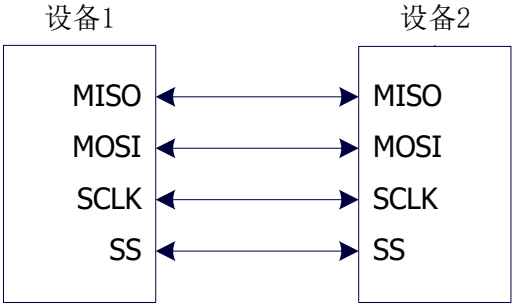
25.4.2 互为主从

两个设备相连，主机和从机不固定。

设置方法 1：两个设备初始化时都设置为 SSIG 设置为 0，MSTR 设置为 1，且将 SS 脚设置为双向口模式输出高电平。此时两个设备都是不忽略 SS 的主机模式。当其中一个设备需要启动传输时，可将自己的 SS 脚设置为输出模式并输出低电平，拉低对方的 SS 脚，这样另一个设备就被强行设置为从机模式了。

设置方法 2：两个设备初始化时都将自己设置成忽略 SS 的从机模式，即将 SSIG 设置为 1，MSTR 设置为 0。当其中一个设备需要启动传输时，先检测 SS 管脚的电平，如果时候高电平，就将自己设置成忽略 SS 的主模式，即可进行数据传输了。

互为主从连接配置图如下所示：



互为主从配置

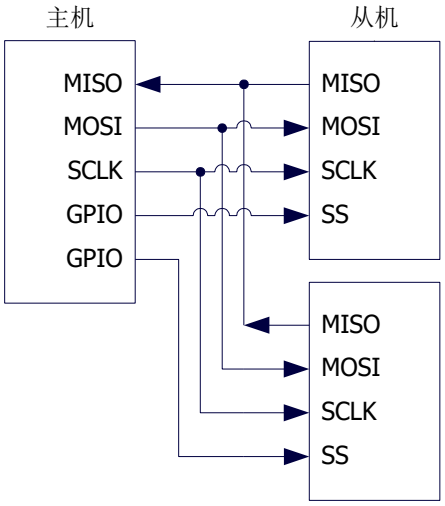
25.4.3 单主多从

多个设备相连，其中一个设备固定作为主机，其他设备固定作为从机。

主机设置：SSIG 设置为 1，MSTR 设置为 1，固定为主机模式。主机可以使用任意端口分别连接各个从机的 SS 管脚，拉低其中一个从机的 SS 脚即可使能相应的从机设备

从机设置：SSIG 设置为 0，SS 管脚作为从机的片选信号。

单主多从连接配置图如下所示：



单主多从配置

25.5 配置 SPI

控制位			通信端口				说明
SPEN	SSIG	MSTR	SS	MISO	MOSI	SCLK	
0	x	x	x	输入	输入	输入	关闭 SPI 功能，SS/MOSI/MISO/SCLK 均为普通 IO
1	0	0	0	输出	输入	输入	**从机模式**，且被选中
1	0	0	1	高阻	输入	输入	**从机模式**，但未被选中
1	0	1→0	0	输出	输入	输入	**从机模式**，不忽略 SS 且 MSTR 为 1 的主机模式，当 SS 管脚被拉低时，MSTR 将被硬件自动清零，工作模式将被被动设置为从机模式
1	0	1	1	输入	高阻	高阻	**主机模式**，空闲状态
					输出	输出	**主机模式**，激活状态
1	1	0	x	输出	输入	输入	**从机模式**
1	1	1	x	输入	输出	输出	**主机模式**

从机模式的注意事项：

当 CPHA＝0 时，SSIG 必须为 0（即不能忽略 SS 脚）。在每次串行字节开始还发送前 SS 脚必须拉低，并且在串行字节发送完后须重新设置为高电平。SS 管脚为低电平时不能对 SPDAT 寄存器执行写操作，否则将导致一个写冲突错误。CPHA=0 且 SSIG=1 时的操作未定义。

当 CPHA＝1 时，SSIG 可以置 1（即可以忽略脚）。如果 SSIG＝0，SS 脚可在连续传输之间保持低有效（即一直固定为低电平）。这种方式适用于固定单主单从的系统。

主机模式的注意事项：

在 SPI 中，传输总是由主机启动的。如果 SPI 使能（SPEN=1）并选择作为主机时，主机对 SPI 数据寄存器 SPDAT 的写操作将启动 SPI 时钟发生器和数据的传输。在数据写入 SPDAT 之后的半个到一个 SPI 位时间后，数据将出现在 MOSI 脚。写入主机 SPDAT 寄存器的数据从 MOSI 脚移出发送到从机的 MOSI 脚。同时从机 SPDAT 寄存器的数据从 MISO 脚移出发送到主机的 MISO 脚。

传输完一个字节后，SPI 时钟发生器停止，传输完成标志（SPIF）置位，如果 SPI 中断使能则会产生一个 SPI 中断。主机和从机 CPU 的两个移位寄存器可以看作是一个 16 位循环移位寄存器。当数据从主机移位传送到从机的同时，数据也以相反的方向移入。这意味着在一个移位周期中，主机和从机的数据相互交换。

通过 SS 改变模式

如果 SPEN=1，SSIG=0 且 MSTR=1，SPI 使能为主机模式，并将 SS 脚可配置为输入模式化或准双向口模式。这种情况下，另外一个主机可将该脚驱动为低电平，从而将该器件选择为 SPI 从机并向其发送数据。为了避免争夺总线，SPI 系统将该从机的 MSTR 清零，MOSI 和 SCLK 强制变为输入模式，而 MISO 则变为输出模式，同时 SPSTAT 的 SPIF 标志位置 1。

用户软件必须一直对 MSTR 位进行检测，如果该位被一个从机选择动作而被动清零，而用户想继续将 SPI 作为主机，则必须重新设置 MSTR 位，否则将一直处于从机模式。

写冲突

SPI 在发送时为单缓冲，在接收时为双缓冲。这样在前一次发送尚未完成之前，不能将新的数据写入移位寄存器。当发送过程中对数据寄存器 SPDAT 进行写操作时，WCOL 位将被置 1 以指示发生数据写冲突错误。在这种情况下，当前发送的数据继续发送，而新写入的数据将丢失。

当对主机或从机进行写冲突检测时，主机发生写冲突的情况是很罕见的，因为主机拥有数据传输的完全控制权。但从机有可能发生写冲突，因为当主机启动传输时，从机无法进行控制。

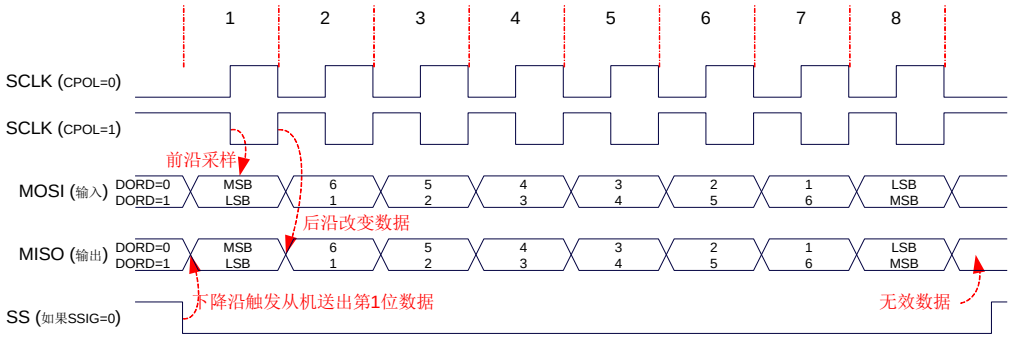
接收数据时，接收到的数据传送到一个并行读数据缓冲区，这样将释放移位寄存器以进行下一个数据的接收。但必须在下个字符完全移入之前从数据寄存器中读出接收到的数据，否则，前一个接收数据将丢失。

WCOL 可通过软件向其写入“1”清零。

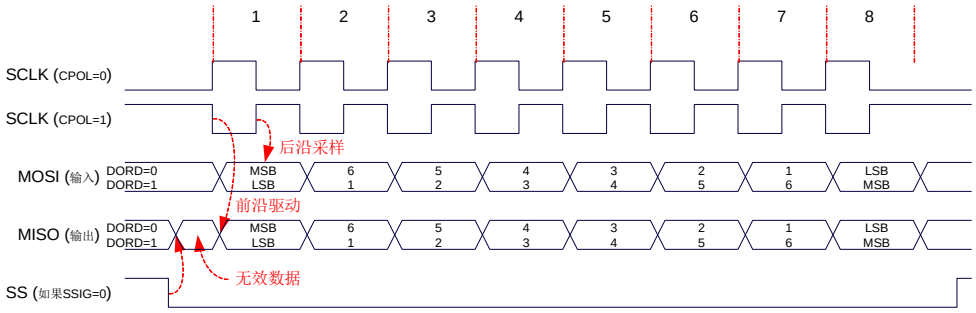
25.6 数据模式

SPI 的时钟相位控制位 CPHA 可以让用户设定数据采样和改变时的时钟沿。时钟极性位 CPOL 可以让用户设定时钟极性。下面图例显示了不同时钟相位、极性设置下 SPI 通讯时序。

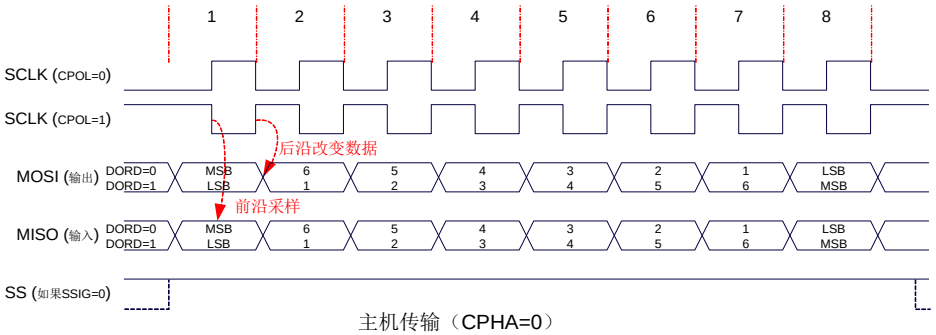
MOSI 发送完成后，MOSI 管脚的空闲状态电平由配置寄存器 2（HSSPIn_CFG2）的 MOSIn_PL 来设定。



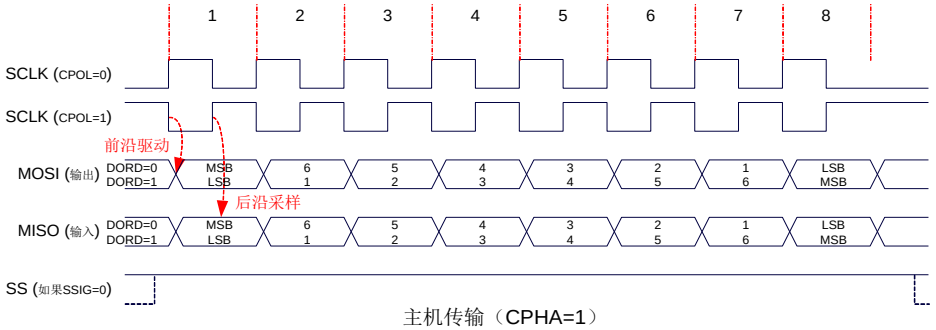
从机传输（CPHA=0）



从机传输（CPHA=1）



主机传输（CPHA=0）



主机传输（CPHA=1）

26 四线 SPI（QSPI）

产品线	QSPI
STC32G144K246 系列	●

STC32G144K246 系列单片机集成了 QSPI 控制器。QSPI 是一种专用的通信接口，连接单线、双线或四线的 SPI Flash 存储介质。该接口可以在以下两种模式下工作：

- 间接模式：使用 QSPI 寄存器执行全部操作
- 状态轮询模式：周期性读取外部 Flash 状态寄存器，而且标志位置 1 时会产生中断（如擦除或烧写完成，会产生中断）

26.1 QSPI 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW4	BFH	-		-		-		QSPI_S[1:0]	

QSPI_S[1:0]：QSPI 功能脚选择位

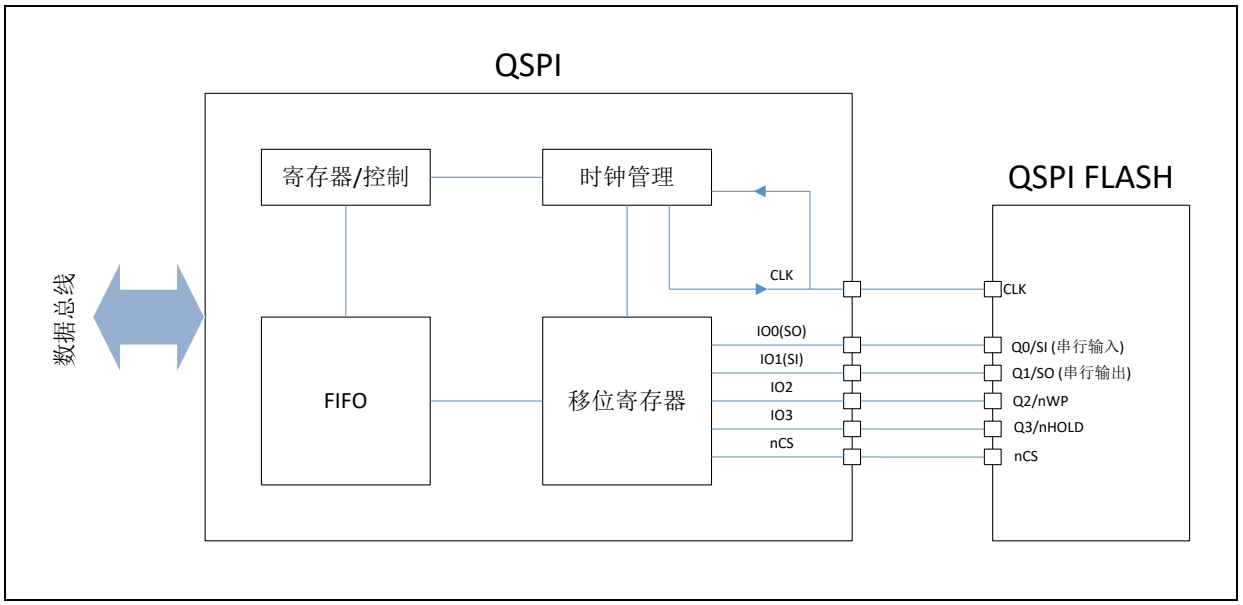
QSPI_S[1:0]	QSPINCS	QSPIIO0	QSPIIO1	QSPIIO2	QSPIIO3	QSPICLK
00	P1.2	P1.3	P1.4	P1.1	P1.0	P1.5
01	P5.4	P4.0	P4.1	P4.6	P4.7	P4.3
10	P7.5	P7.0	P7.2	P7.3	P7.4	P7.1
11	-	-	-	-	-	-

26.2 QSPI 主要特性

- 两种功能模式：间接模式和　　状态轮询模式
- 支持 SDR 模式
- 针对间接模式，完全可编程操作码
- 针对间接模式，完全可编程帧格式
- 集成 32 字节的 FIFO，用于发送和接收
- 支持 8 位数据访问
- 在达到 FIFO 阈值、超时、操作完成以及发生访问错误时产生中断

26.3 QSPI 功能说明

26.3.1 QSPI 框图



26.3.2 QSPI 应用注意事项

1、 当使能 QSPI 功能后，QSPI 相关的 6 个 I/O 口均会由 QSPI 模块进行托管，用户程序无法再对这些 I/O 进行输出控制，即使 QSPI 工作在单线模式（只使用 IO0 和 IO1 进行数据传输），用户程序也无法操作 IO2 和 IO3。

2、 QSPI 的 nCS 脚的有效信号是由硬件自动控制的，不能用普通的 I/O 口来代替 nCS 进行控制。

3、 一组硬件 QSPI 口只能用于控制一个 QSPI 设备，不能用类似普通 SPI 的方式采用多个普通 I/O 口来当作 nCS 片选多个设备。但可以使用分时复用的方法将整组 QSPI 切换到不同的 I/O 管脚，来控制多个 QSPI 设备。

4、 STC32G144K246 的 QSPI 只有主机模式，没有从机模式。

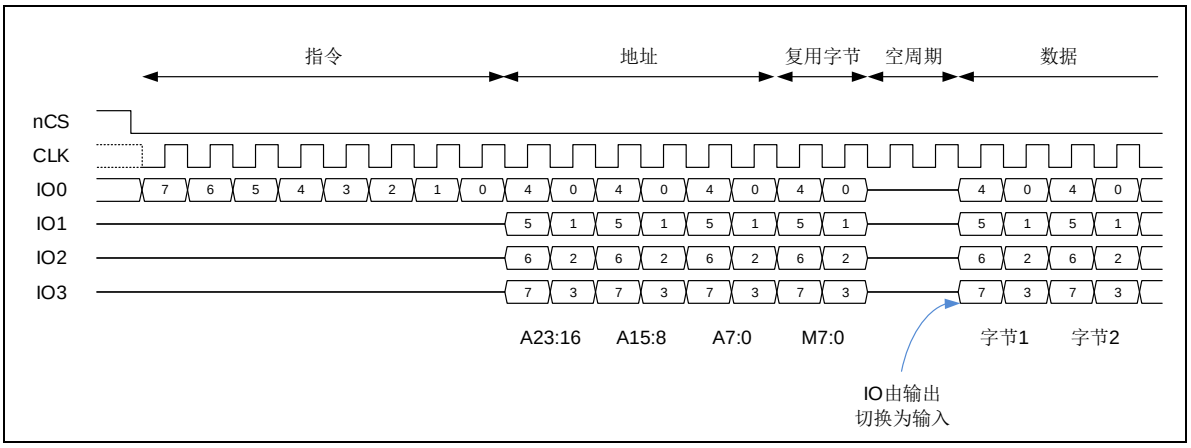
5、 STC32G144K246 的 QSPI 的工作模式可配置为兼容普通 SPI 的工作模式 0（空闲是 CLK 为低电平）和兼容普通 SPI 的工作模式 3（空闲是 CLK 为高电平），使用 QSPI_DCR1 寄存器中的 CKMODE 进行配置。

26.3.3 QSPI 命令序列

QSPI 通过命令与 Flash 通信每条命令包括指令、地址、交替字节、空指令和数据这五个阶段。任一阶段均可跳过，但至少要包含指令、地址、交替字节或数据阶段之一。

nCS 在每条指令开始前下降到低电平，在每条指令完成后再次上升为高电平。

（四线模式下的读命令示例）



指令阶段

这一阶段，将在 QSPI_CCR1 寄存器的 INSTRUCTION 字段中配置的一条 8 位指令发送到 Flash，指定待执行操作的类型。

尽管大多数 Flash 从 IO0/SO 信号（单线 SPI 模式）只能以一次 1 位的方式接收指令，但指令阶段可选择一次发送 2 位（在双线 SPI 模式中通过 IO0 / IO1）或一次发送 4 位（在四线 SPI 模式中通过 IO0 / IO1 / IO2 / IO3）。这可通过 QSPI_CCR2 寄存器中的 IMODE[1:0]字段进行配置。若 IMODE = 00，则跳过指令阶段，命令序列从地址阶段（如果存在）开始。

地址阶段

在地址阶段，将 1～4 字节地址发送到 Flash，指示操作地址。待发送的地址字节数在 QSPI_CCR2 寄存器的 ADSIZE[1:0]字段中进行配置。在间接模式和自动轮询模式下，待发送的地址字节在 QSPI_AR1～QSPI_AR4 寄存器的 ADDRESS[31:0]中指定。

地址阶段可一次发送 1 位（在单线 SPI 模式中通过 SO）、2 位（在双线 SPI 模式中通过 IO0 / IO1）或 4 位（在四线 SPI 模式中通过 IO0 / IO1 / IO2 / IO3）。这可通过 QSPI_CCR2 寄存器中的 ADMODE[1:0]字段进行配置。若 ADMODE = 00，则跳过地址阶段，命令序列直接进入下一阶段（如果存在）。

交替字节阶段

在交替字节阶段，将 1 字节发送到 Flash，一般用于控制操作模式。待发送的交替字节数在 QSPI_CCR3 寄存器的 ABSIZE[1:0]字段中进行配置。待发送的字节在 QSPI_ABR 寄存器中指定。

交替字节阶段可一次发送 1 位（在单线 SPI 模式中通过 S0）、2 位（在双线 SPI 模式中通过 IO0 / IO1）或 4 位（在四线 SPI 模式中通过 IO0 / IO1 / IO2 / IO3）。这可通过 QSPI_CCR2 寄存器中的 ABMODE[1:0]字段进行配置。若 ABMODE = 00,则跳过交替字节阶段，命令序列直接进入

下一阶段（如果存在）。

交替字节阶段存在仅需发送单个半字节而不是一个全字节的情况，比如采用双线模式并且仅使用两个周期发送交替字节时。在这种情况下，固件可采用四线模式（ABMODE = 11）并发送一个字节，方法是 ALTERNATE 的 Bit7 和 Bit3 置“1”（I03 保持高电平）且 Bit6 和 Bit2 置“0”（I02 线保持低电平）。此时，半字节的高 2 位存放在 ALTERNATE 的 Bit4 和 Bit3,低 2 位存放在位 1 和 0 中。例如，如果半字节 2（0010）通过 IO0/IO1 发送，则 ALTERNATE 应设置为 0x8A（1000_1010）。

空指令周期阶段

在空指令周期阶段，给定的 1-31 个周期内不发送或不接收任何数据，目的是当采用更高的时钟频率时，给 Flash 留出准备数据阶段的时间。这一阶段中给定的周期数在 QSPI_CCR3 寄存器的 DCYC[4:0]字段中指定。在 SDR 模式下，持续时间被指定为一定个数的全时钟周期。

若 DCYC 为零，则跳过空指令周期阶段，命令序列直接进入数据阶段（如果存在）。

空指令周期阶段的操作模式由 DMODE 确定。

为确保数据信号从输出模式转变为输入模式有足够的“周转”时间，使用双线和四线模式从 Flash 接收数据时，至少需要指定一个空指令周期。

数据阶段

在数据阶段，可从 Flash 接收或向其发送任意数量的字节。

在间接模式和自动轮询模式下，待发送/接收的字节数在 QSPI_DLR 寄存器中指定。

在间接写入模式下，发送到 Flash 的数据必须写入 QSPI_DR 寄存器。在间接读取模式下，通过读取 QSPI_DR 寄存器获得从 Flash 接收的数据。

数据阶段可一次发送/接收 1 位（在单线 SPI 模式中通过 SO）、2 位（在双线 SPI 模式中通过 IO0 / IO1）或 4 位（在四线 SPI 模式中通过 IO0 / IO1 / IO2 / IO3）。这可通过 QSPI_CCR2 寄存器中的 ABMODE[1:0]字段进行配置。

若 DMODE = 00，则跳过数据阶段，命令序列在拉高 nCS 时立即完成。这一配置仅可用于仅间接写入模式。

26.3.4 QSPI 信号接口协议模式

单线 SPI 模式

传统 SPI 模式允许串行发送/接收单独的 1 位。在此模式下，数据通过 SO（IO0）信号发送到 Flash。从 Flash 接收到的数据通过 SI（IO1）送达。

通过将 QSPI_CCRn 寄存器中的 IMODE/ADMODE/ABMODE/DMODE 字段设置为 01，可对不同的命令阶段分别进行配置，以使用此单个位模式。

在每个已配置为单线模式的阶段中：

- IO0（SO）为输出模式
- IO1（SI）为高阻输入模式）

- IO2 为输出模式并强制置“1”（以禁止“写保护”功能）
- IO3 为输出模式并强制置“1”（以禁止“保持”功能）

若 DMODE = 01，这对于空指令阶段也同样如此。

双线 SPI 模式

在双线模式下，通过 IO0 / IO1 信号同时发送/接收两位。

通过将 QSPI_CCRn 寄存器中的 IMODE/ADMODE/ABMODE/DMODE 字段设置为 10,可对不同的命令阶段分别进行配置，以使用双线 SPI 模式。

在每个已配置为双线模式的阶段中：

- IO0/IO1 在数据阶段进行读取操作时处于高阻态（输入），在其他情况下为输出
- IO2 处于输出模式并强制置“1”
- IO3 处于输出模式并强制置“1”

在空指令阶段，若 DMODE = 01，则 IO0/IO1 始终保持高阻态。

四线 SPI 模式

在四线模式下，通过 IO0/IO1/IO2/IO3 信号同时发送/接收四位。

通过将 QSPI_CCRn 寄存器中的 IMODE/ADMODE/ABMODE/DMODE 字段设置为 11,可 对不同的命令阶段分别进行配置，以使用四线 SPI 模式。

在每个已配置为四线模式的阶段中，IO0/IO1/IO2/IO3 在数据阶段进行读取操作时均处于高 阻态（输入），在其他情况下为输出。

在空指令阶段中，若 DMODE = 11,则 IO0/IO1/IO2/IO3 均为高阻态。

I02 和 I03 仅用于 QSPI 模式如果未配置任何阶段使用四线 SPI 模式，即使 QSPI 激活，对应 IO2 和 I03 的引脚也可用于其他功能。

SDR 模式

默认情况下，QSPI 在单倍数据速率（SDR）模式下工作。

在 SDR 模式下，当 QSPI 驱动 IO0/SO、IO1、IO2、IO3 信号时，这些信号仅在 CLK 的下降沿发生转变。

在 SDR 模式下接收数据时，QSPI 假定 Flash 也通过 CLK 的下降沿发送数据。默认情 况下（SSHIFT = 0 时），将使用 CLK 后续的边沿（上升沿）对信号进行采样。

26.3.5 QSPI 间接模式

在间接模式下，通过写入 QSPI 寄存器来触发命令；并通过读写数据寄存器来传输数据，就如同对待其他通信外设那样。

若 FMODE = 00，则 QSPI 处于间接写入模式，字节在数据阶段中发送到 Flash。通过写入数据寄存器 QSPI_DR 的方式提供数据。

若 FMODE = 01，则 QSPI 处于间接读取模式，在数据阶段中从 Flash 接收字节。通过读取 QSPI_DR 来获取数据。

读取/写入的字节数在数据长度寄存器（QSPI_DLR）中指定。如果 QSPI_DLR = OxFFF_FFFF（全为“1”），则数据长度视为未定义，QSPI 将继续传输数据，直到到达（由 FSIZE 定义的）Flash 的结尾。如果不传输任何字节，DMODE 应设置为 00。

如果 QSPI_DLR = 0xFF，并且 FSIZE = 0x1F （最大值指示一个 4GB 的 Flash），在此特殊情况下，传输将无限继续下去，仅在出现终止请求或 QSPI 被禁止后停止。在读取最后一个存储器地址后（地址为 OxFFFF_FFFF），将从地址=0x0000_0000 开始继续读取。

当发送或接收的字节数达到编程设定值时，如果 TCIE = 1，则 TCF 置 1 并产生中断。在数据数量不确定的情况下，将根据 QSPI_CR 中定义的 Flash 大小，在达到外部 SPI 的限制时，TCF 置 1。

触发命令启动

从本质上讲，在固件给出命令所需的最后一点信息时，命令即会启动。根据 QSPI 的配 置，在间接模式下有三种触发命令启动的方式。在出现以下情形时，命令立即启动：

1、 对 INSTRUCTION[7:0] （QSPI_CCR1）执行写入操作，如果没有地址是必需的（当 ADMODE = 00）并且不需要固件提供数据（当 FMODE = 01 或 DMODE = 00）
2、 对 ADDRESS[31:0] （QSPI_AR）执行写入操作，如果地址是必需的（当 ADMODE = 00）并且不需要固件提供数据（当 FMODE = 01 或 DMODE = 00）
3、 对 DATA[7:0]（QSPI_DR）执行写入操作，如果地址是必需的（当 ADMODE != 00） 并且需要固件提供数据（当 FMODE = 00 并且 DMODE != 00）

写入交替字节寄存器（QSPI_ABR）始终不会触发命令启动。如果需要交替字节，必须预先进行编程。

如果命令启动，BUSY 位（QSPI_LSR 的位 5）将自动置 1。

FIFO 和数据管理

在间接模式中，数据将通过 QSPI 内部的一个 32 字节 FIFO。FLEVEL[5:0] （QSPI_SR2）指示 FIFO 目前保存了多少字节。

在间接写入模式下（FMODE = 00）,固件写入 QSPI_DR 时，将在 FIFO 中加入数据。 字写入将在 FIFO 中增加 4 个字节，半字写入增加 2 个字节，而字节写入仅增加 1 个字节。

如果固件在 FIFO 中加入的数据过多（超过 DL[31:0]指示的值），将在写入操作结束（TCF 置 1）时从 FIFO 中清除超出的字节。

对 QSPI_DR 的字节/半字访问必须仅针对该 32 位寄存器的最低有效字节/半字。

FTHRES[3:0]用于定义 FIFO 的阈值如果达到阈值，FTF（FIFO 阈值标志）置 1 在间接读取模式下，从 FIFO 中读取的有效字节数超过阈值时，FTF 置 1。从 Flash 中读取最后一个字节后，如果 FIFO 中依然有数据，则无论 FTHRES 的设置为何，FTF 也都会置 1。在间接写入模式下，当 FIFO 中的空字节数超过阈值时，FTF 置 1。

如果 FTIE = 1,则 FTF 置 1 时产生中断。如果阈值条件不再为“真”（CPU 传输了足够的数据后），则 FTF 由 HW 清零。

在间接模式下，当 FIFO 已满，QSPI 将暂时停止从 Flash 读取字节以避免上溢。

26.3.6 QSPI 状态标志轮询模式

在自动轮询模式下，QSPI 周期性启动命令以读取一定数量的状态字节（最多 4 个）。 可屏蔽接收的字节以隔离一些状态位，从而在所选的位具有定义的值时可产生中断。

对 Flash 的访问最初与在间接读取模式下相同：如果不需要地址（AMODE = 00），则在写入 QSPI_CCR 时即开始访问。否则，如果需要地址，则在写入 QSPI_AR 时开始第一次访问。BUSY 在此时变为高电平，即使在周期性访问期间也保持不变。

在自动轮询模式下，MASK[7:0]（QSPI_PSMAR）的内容用于屏蔽来自 Flash 的数据。

如果 MASK[n] = 0，则屏蔽结果的位 n，从而不考虑该位。如果 MASK[n] = 1 并且位[n]的内 容与 MATCH[n] （QSPI_PSMAR）相同，说明存在位 n 匹配。

如果轮询匹配模式位（PMM， QSPI_CR3[7]）为 0，将激活“AND”匹配模式。这意味着状态匹配标志（SMF）仅在全部未屏蔽位均存在匹配时置 1。

如果 PMM = 1，则激活“OR”匹配模式。这意味着 SMF 在任意未屏蔽位存在匹配时置 1。

如果 SMIE = 1，则在 SMF 置 1 时调用一个中断。

如果自动轮询模式停止（APMS）位置 1，则操作停止并且 BUSY 位在检测到匹配时清零。否则，BUSY 位保持为“1”，在发生中止或禁止 QSPI（EN = 0）前继续进行周期性访问。

数据寄存器（QSPI_DR）包含最新接收的状态字节（FIFO 停用）。数据寄存器的内容不受匹配逻辑所用屏蔽方法的影响。FTF 状态位在新一次状态读取完成后置 1，并且 FTF 在数据读取后清零。

26.3.7 QSPI Flash 配置

设备配置寄存器（QSPI_DCR）可用于指定外部 SPI Flash 的特性。

FSIZE[4:0]字段使用下面的公式定义外部存储器的大小：

Flash 中的字节数=$2^{[FSIZE+1]}$

FSIZE+1 是对 Flash 寻址所需的地址位数。在间接模式下，Flash 容量最高可达 4GB （使用 32 位进行寻址）。

QSPI 连续执行两条命令时，它在两条命令之间将片选信号（nCS）置为高电平默认仅一个 CLK 周期时长。如果 Flash 需要命令之间的时间更长，可使用片选高电平时间（CSHT）字段指定 nCS

必须保持高电平的最少 CLK 周期数（最大为 8）。

时钟模式（CKMODE）位指示命令之间的 CLK 信号逻辑电平（nCS = 1 时）。

26.3.8 QSPI 延迟数据采样

默认情况下，QSPI 在 Flash 驱动信号后过半个 CLK 周期才对 Flash 驱动的数据采样。

26.3.9 QSPI 配置

QSPI 配置分两个阶段

- QSPI IP 配置
- QSPI Flash 配置

QSPI 在配置完毕并使能后，即可在间接模式和状态轮询模式这两种操作模式之一下工作。

QSPI IP 配置

通过 QSPI_CR 配置 QSPI IP 。用户应配置传入数据的时钟预分频器的分频系数以及采样移位设置。

DMA 请求通过 DMAEN 位置 1 使能。若是用于中断，则相关使能位也可在该阶段置 1。

生成 DMA 请求或生成中断的 FIFO 电平在 FTHRES 位中进行编程。

QSPI Flash 配置

与外部目标 Flash 相关的参数通过 QSPI_DCR 寄存器进行配置。用户应在 FSIZE 位中编程 Flash 的大小、在 CSHT 位中编程片选保持高电平的最短时间以及在 MODE 位中编程功能模式（模式 0 或模式 3）。

26.3.10 QSPI 的用法

使用 FMODE[1:0]（QSPI_CCR4）选择操作模式。

间接模式的操作步骤

FMODE 编程为 00 可选择间接写入模式，将数据发送到 Flash。
FMODE 编程为 01 可选择间接读取模式，读取 Flash 中的数据。

QSPI 用于间接模式时，采用以下方式构建帧：

- 在 QSPI_DLR 中指定待读取或写入的字节数
- 在 QSPI_CCR 中指定帧格式、模式和指令代码
- 在 QSPI_ABR 中指定要在地址阶段后立即发送的可选交替字节
- 在 QSPI_AR 中指定目标地址
- 通过 QSPI_DR 从 FIFO 读取数据/向 FIFO 写入数据

在写入控制寄存器（QSPI_CR）时，用户可指定以下设置：

- 使能位（EN）设置为“1”
- 超时计数器使能位（TCEN）
- FIFO 阈值（FTRHES），以指示 FTF 标志在何时置 1
- 中断使能
- 自动轮询模式参数：匹配模式和停止模式（在 FMODE = 11 时有效）
- 时钟预分频器

在写入通信配置寄存器（QSPI_CCRn）时，用户指定以下参数：

- 通过 INSTRUCTION 位指定指令字节
- 通过 IMODE 位指定指令发送方式（1/2/4 线）
- 通过 ADMODE 位指定地址发送方式（无/1/2/4 线）
- 通过 ADSIZE 位指定地址长度（8/16/24/32 位）
- 通过 ABMODE 位指定交替字节发送方式（无/1/2/4 线）
- 通过 ABSIZE 位指定交替字节数（1/2/3/4）
- 通过 DCYC 位指定空指令的周期数
- 通过 DMODE 位指定数据发送/接收方式（无/1/2/4 线）

如果无需为某个命令更新地址寄存器（QSPI_ARn）与数据寄存器（QSPI_DR），则在写入 QSPI_CCR1 时，该命令序列便立即启动。在 ADMODE 和 DMODE 应为 00 时，或在间接读取模式（FMODE = 01）下仅 ADMODE = 00 时，便属于此情况。在需要地址（ADMODE 不为 00），但无需写入数据寄存器（FMODE = 01 或 DMODE = 00）时，通过写入 QSPI_AR 更新地址后，命令序列便立即启动。

在数据传输（FMODE = 00 并且 DMODE ! = 00）中，通过 QSPI_DR 写入 FIFO 触发通道启动。

状态标志轮询模式

将 FMODE 字段（QSPI_CCR4）设置为 10，使能状态标志轮询模式在此模式下，将发送编程的帧并周期性检素数据。

每帧中读取的最大数据量为 4 字节。如果 QSPI_DLR 请求更多的数据，则忽略多余的部分并仅读取 4 个字节。

在 QSPI_PISR 寄存器中指定周期性。

在检索到状态数据后，可在内部进行处理，以达到以下目的：

- 将状态匹配标志位置 1，如果使能，还将产生中断
- 自动停止周期性检索状态字节

接收到的值可通过存储于 QSPI_PSMKR 中的值进行屏蔽，并与存储在 QSPI_PSMAR 中的值进行或运算或与运算。

若是存在匹配，则状态匹配标志置 1，并且在使能了中断的情况下还将产生中断；如果 AMPS 位置 1，则 QSPI 自动停止。

在任何情况下，最新的检索值都在 QSPI_DR 中可用。

26.3.11 指令仅发送一次

一些 Flash （例如 Winbound）能够提供一种模式，指令在该模式中仅通过第一个命令序列进行发送，后续的命令根据地址直接启动。用户通过使用 SIOO 位可利用此功能的优势。

SIOO 对于所有功能模式（间接模式和状态轮询模式）均有效。如果 SIOO 位置 1，仅第一条命令发送指令，接着对 QSPI_CCRn 执行写入操作。后续命令序列都将跳过指令阶段，直到 QSPI_CCR 被写入为止。

IMODE = 00 （无指令）时，SIOO 不起作用。

26.3.12 QSPI 差错管理

在以下情况下可能产生错误：

- 在间接模式或状态标志轮询模式下，如果在 QSPI_AR 中编程了错误的地址（根据 QSPI_DCR 中 FSIZE[4:0]定义的 Flash 大小）：TEF 将置 1，如果使能，还将产生中断。
- 另外，在间接模式下，如果地址加数据的长度超过 Flash 的大小，TEF 将在访问被触发 时置 1。

26.3.13 QSPI 的繁忙位和中止功能

在 QSPI 启动对 Flash 的操作时，QSPI_SR 中的 BUSY 位自动置 1。

在间接模式下，在 QSPI 完成了请求的命令序列并且 FIFO 为空时，BUSY 位复位。

在自动轮询模式下，仅当最后一次周期性访问完成时（因 APMS = 1 时发生匹配，或因中止），BUSY 位才变为低电平。

任何操作都可通过将 QSPI_CR 中的 ABORT 位置 1 来中止。在完成中止时，BUSY 位和 ABORT 位自动复位，FIFO 淆空。

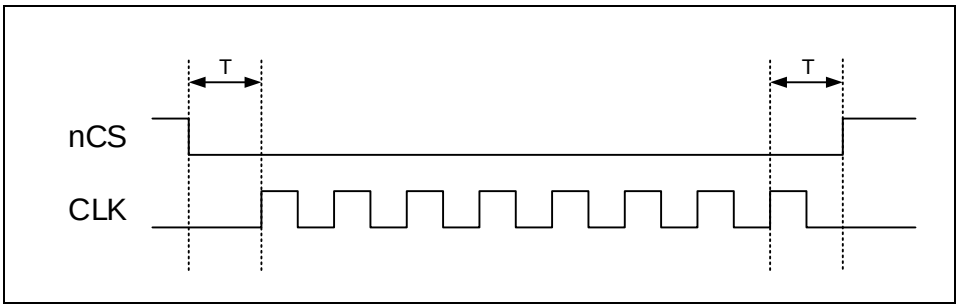
注: 如果中止对状态寄存器的写入操作，有些 Rash 可能发生错误行为。

26.3.14 nCS 行为

默认情况下，nCS 为高电平，取消选择外部 Flash。nCS 在操作开始前下降，在操作完成时立即上升。

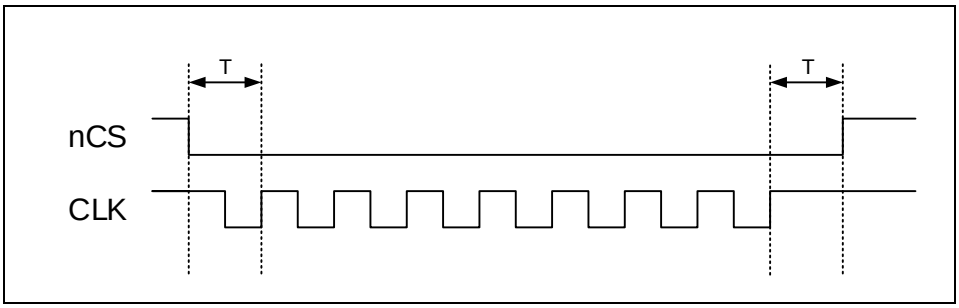
当 CKMODE = 0 （“模式 0”，在未进行任何操作时 CLK 保持低电平）时，nCS 在操作首次升高 CLK 边沿时的一个 CLK 周期前降至低电平，在操作最后一次升高 CLK 边沿时的一个 CLK 周期后升至高电平，如下图所示：

CKMODE = 0 时的 nCS （T = CLK 周期）



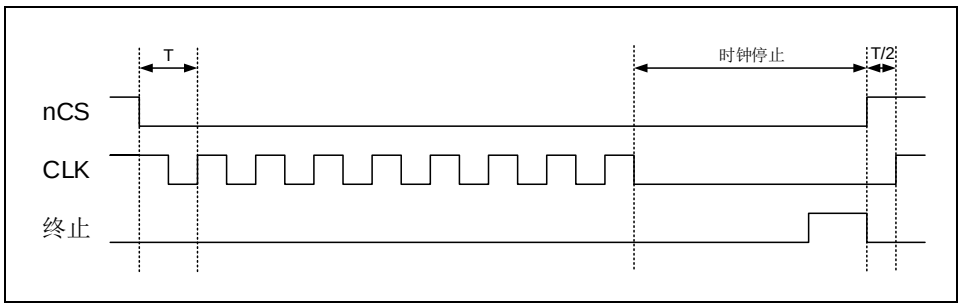
当 CKMODE = 1（“模式 3”，在未进行任何操作时 CLK 升至高电平）且 DDRM = 0（SDR 模式）时，nCS 仍在操作首次升高 CLK 边沿时的一个 CLK 周期前降至低电平，在操作最后一次升高 CLK 边沿时的一个 CLK 周期后升至高电平，如下图所示：

SDR 模式下 CKMODE = 1 时的 nCS （T = CLK 周期）



若 FIFO 在读取操作中保持写满状态或在写入操作中保持为空，则在固件干预 FIFO 前，操作停止并且 CLK 保持低电平。若操作停止时发生中止，则 nCS 在请求中止后即升至高电平，CLK 则在半个周期后升至高电平，如下图所示：

CKMODE=1 且发生中止时的 nCS（T=CLK 周期）



26.4 QSPI 中断

发生如下事件时可生成中断:

- 状态匹配
- FIFO 阈值
- 传输完成

QSPI 中断请求

中断事件	事件标志	使能控制位
状态匹配	SMF	SMIE
FIFO 阈值	FTF	FTIE
传输完成	TCF	TCIE

26.5 QSPI 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
QSPI_CR1	QSPI 控制寄存器 1	7EF900H	-	-	-	-	-	-	ABORT	EN	xxxx,xx00
QSPI_CR2	QSPI 控制寄存器 2	7EF901H	-	-	-	FTHRES[4:0]					xxx0,0000
QSPI_CR3	QSPI 控制寄存器 3	7EF902H	PMM	APMS	-	-	SMIE	FTIE	TCIE	TEIE	00xx,0000
QSPI_CR4	QSPI 控制寄存器 4	7EF903H	PSCR[7:0]								0000,0000
QSPI_DCR1	QSPI 器件配置寄存器 1	7EF904H	-	CSHT[2:0]			QSPI_IP[1:0]		-	CKMODE	x000,00x0
QSPI_DCR2	QSPI 器件配置寄存器 2	7EF905H	-	-	-	FSIZE[4:0]					0000,0000
QSPI_SR1	QSPI 状态寄存器 1	7EF906H	-	-	BUSY	-	SMF	FTF	TCF	TEF	xx0x,0100
QSPI_SR2	QSPI 状态寄存器 2	7EF907H	-	-	FLEVEL[5:0]						xx00,0000
QSPI_FCR	QSPI 标志清零寄存器	7EF908H	-	-	-	-	CSMF	-	CTCF	CTEF	xxxx,0x00
QSPI_DLR1	QSPI 数据长度寄存器 1	7EF910H	DL[7:0]								0000,0000
QSPI_DLR2	QSPI 数据长度寄存器 2	7EF911H	DL[15:8]								0000,0000
QSPI_CCR1	QSPI 通信配置寄存器 1	7EF914H	INSTRUCTION[7:0]								0000,0000
QSPI_CCR2	QSPI 通信配置寄存器 2	7EF915H	ABMODE[1:0]		ADSIZE[1:0]		ADMODE[1:0]		IMODE[1:0]		0000,0000
QSPI_CCR3	QSPI 通信配置寄存器 3	7EF916H	-	DCYC[4:0]					ABSIZE[1:0]		x000,0000
QSPI_CCR4	QSPI 通信配置寄存器 4	7EF917H	-	-	-	SIOO	FMODE[1:0]		DMODE[1:0]		xxx0,0000
QSPI_AR1	QSPI 地址寄存器 1	7EF918H	ADR[7:0]								0000,0000
QSPI_AR2	QSPI 地址寄存器 2	7EF919H	ADR[15:8]								0000,0000
QSPI_AR3	QSPI 地址寄存器 3	7EF91AH	ADR[23:16]								0000,0000
QSPI_AR4	QSPI 地址寄存器 4	7EF91BH	ADR[31:24]								0000,0000
QSPI_ABR1	QSPI 交替字节寄存器 1	7EF91CH	ALTERNATED[7:0]								0000,0000
QSPI_ABR2	QSPI 交替字节寄存器 2	7EF91DH	ALTERNATED[15:8]								0000,0000
QSPI_ABR3	QSPI 交替字节寄存器 3	7EF91EH	ALTERNATED[23:16]								0000,0000
QSPI_ABR4	QSPI 交替字节寄存器 4	7EF91FH	ALTERNATED[31:24]								0000,0000
QSPI_DR	QSPI 数据寄存器	7EF920H	DATA[7:0]								0000,0000
QSPI_PSMKR1	QSPI 状态屏蔽寄存器	7EF924H	MASK[7:0]								0000,0000
QSPI_PSMAR1	QSPI 状态匹配寄存器	7EF928H	MATCH[7:0]								0000,0000
QSPI_PIR1	QSPI 轮询间隔寄存器 1	7EF92CH	INTERVAL[7:0]								0000,0000
QSPI_PIR2	QSPI 轮询间隔寄存器 2	7EF92DH	INTERVAL[15:8]								0000,0000

26.5.1 QSPI 控制寄存器 1（QSPI_CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_CR1	7EF900H	-	-	-	-	-	-	ABORT	EN

ABORT：中止请求。该位中止执行中的命令序列。在中止完成时自动复位。该位可停止当前的传输。在轮询模式或内存映射模式下，该位也用以复位 APM 位或 DM 位。

0：不请求中止

1：请求中止

EN：QSPI 功能使能控制位

0：禁止 QSPI

1：使能 QSPI

26.5.2 QSPI 控制寄存器 2（QSPI_CR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_CR2	7EF901H	-	-	-	FTHRES[4:0]				

FTHRES[4:0]：FIFO 阈值级别（FIFO threshold level）。

定义在间接模式下 FIFO 中将导致 FIFO 阈值标志 FTF 置 1 的字节数阈值。

在间接写入模式下（FMODE = 00）

0：如果 FIFO 中存在 1 个或更多空闲字节可供写入，则 FTF 置 1

1：如果 FIFO 中存在 2 个或更多空闲字节可供写入，则 FTF 置 1

…

31：如果 FIFO 中存在 32 个空闲字节可供写入，则 FTF 置 1

在间接读取模式下（FMODE = 01）

0：如果 FIFO 中存在 1 个或更多有效字节可供读取，则 FTF 置 1

1：如果 FIFO 中存在 2 个或更多有效字节可供读取，则 FTF 置 1

…

31：如果 FIFO 中存在 32 个有效字节可供读取，则 FTF 置 1

26.5.3 QSPI 控制寄存器 3（QSPI_CR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_CR3	7EF902H	PMM	APMS	-	-	SMIE	FTIE	TCIE	TEIE

PMM：轮询匹配模式（Polling match mode）。

该位指示在自动轮询模式期间用来确定是否匹配的方法。仅可在 BUSY = 0 时修改该位。

0：AND 匹配模式。

如果从 Flash 接收的所有未屏蔽位均与匹配寄存器中的对应位相匹配，则 SMF 置 1。

1：OR 匹配模式。

如果从 Flash 接收的任意一个未屏蔽位与匹配寄存器中的对应位相匹配，则 SMF 置 1。

APMS：自动轮询模式停止（Automatic poll mode stop）

该位确定在匹配后自动轮询是否停止。仅可在 BUSY = 0 时修改该位。

0：仅通过中止或禁用 QSPI 停止自动轮询模式。

1：发生匹配时，自动轮询模式停止。

SMIE：状态匹配中断使能（Status match interrupt enabl）

0：禁止状态匹配中断

1：使能状态匹配中断

FTIE：FIFO 阈值中断使能（FIFO threshold interrupt enable）

0：禁止 FIFO 阈值中断

1：使能 FIFO 阈值中断

TCIE：传输完成中断使能（Transfer complete interrupt enable）

0：禁止传输完成中断

1：使能传输完成中断

TEIE：传输错误中断使能（Transfer error interrupt enable）

0：禁止传输错误中断

1：使能传输错误中断

26.5.4 QSPI 控制寄存器 4（QSPI_CR4）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_CR4	7EF903H	PSCR[7:0]							

PSCR[7:0]：时钟预分频器（Clock prescaler）。

对于奇数时钟分频系数，CLK 的占空比并非 50%。时钟信号的高电平持续时间比低电平持续时间多一个周期。仅可在 BUSY = 0 时修改该位。

PSCR	QSPI 时钟频率
0	系统时钟/1
1	系统时钟/2
2	系统时钟/3
3	系统时钟/4
…	…
255	系统时钟/256

26.5.5 QSPI 器件配置寄存器 1（QSPI_DCR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_DCR1	7EF904H	-	CSHT[2:0]			QSPI_IP[1:0]		-	CKMODE

CSHT[2:0]：片选高电平时间（Chip select high time）。仅可在 BUSY = 0 时修改该位。

CSHT[2:0]	nCS 在 Flash 命令之间保持高电平时间
0	1 个周期
1	2 个周期
2	3 个周期
3	4 个周期
…	…
7	8 个周期

QSPI_IP[1:0]：QSPI 中断优先级

CKMODE：SPI 模式 0/模式 3（Mode 0 / mode 3）选择位。该位指示 CLK 在 nCS=1 时的电平。仅可在 BUSY = 0 时修改该位。

0：nCS 为高电平（片选释放）时，CLK 必须保持低电平。这称为模式 0。

1：nCS 为高电平（片选释放）时，CLK 必须保持高电平。这称为模式 3。

26.5.6 QSPI 器件配置寄存器 2（QSPI_DCR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_DCR2	7EF905H	-	-	-	FSIZE[4:0]				

FSIZE[4:0]：配置外部 Flash 存储器的大小。仅可在 BUSY = 0 时修改该位。

Flash 中的字节数 $= 2^{[FSIZE+1]}$

FSIZE+1 是对 Flash 寻址所需的地址位数。在间接模式下，Flash 容量最高可达 4GB（使用 32 位进行寻址）。

26.5.7 QSPI 状态寄存器 1（QSPI_SR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_SR1	7EF906H	-	-	BUSY	-	SMF	FTF	TCF	TEF

BUSY：QSPI 忙标志位（只读位）

QSPI 操作进行时，该位置 1。在对 Flash 的操作完成并且 FIFO 为空时，该位自动清零。

SMF：状态匹配标志（Status match flag）

在自动轮询模式下，若未屏蔽的接收数据与匹配寄存器（QSPI_PSMAR）中的对应位相匹配，则该位置 1。向 CSMF 写入 1 可将该位清零。

FTF：FIFO 阈值标志（FIFO threshold flag）

在间接模式下，若达到 FIFO 阈值，或从 Flash 读取完成后，FIFO 中留有数据时，该位置 1。只要阈值条件不再为“真”，该位就自动清零。

在自动轮询模式下，每次读取状态寄存器时，该位即置 1；读取数据寄存器时，该位清零。

TCF：传输完成标志（Transfer complete flag）

在间接模式下，当传输的数据数量达到编程设定值，或在任何模式下传输中止时，该位置 1。向 CTCF 写入 1 时，该位清零。

TEF：传输错误标志（Transfer error flag）

在间接模式下访问无效地址时，该位置 1。向 CTEF 写入 1 可将该位清零。

26.5.8 QSPI 状态寄存器 2（QSPI_SR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_SR2	7EF907H	-	-	FLEVEL[5:0]					

FLEVEL[5:0]：FIFO 级别（FIFO level）

该字段给出 FIFO 中的有效字节数。FIFO 为空时 FLEVEL = 0；写满时 FLEVEL = 32

在自动状态轮询模式下，FLEVEL 为零。

26.5.9 QSPI 标志清零寄存器（QSPI_FCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_FCR	7EF908H	-	-	-	-	CSMF	-	CTCF	CTEF

CSMF：清除状态匹配标志（Clear status match flag）。

写入 1 可将 QSPI_SR1 寄存器中的 SMF 标志清零

CTCF：清除传输完成标志（Clear transfer complete flag）。

写入 1 可将 QSPI_SR1 寄存器中的 TCF 标志清零

CTEF：清除传输错误标志（Clear Transfer error flag）。

写入 1 可将 QSPI_SR1 寄存器中的 TEF 标志清零

26.5.10 QSPI 数据长度寄存器（QSPI_DLR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_DLR1	7EF910H	DL[7:0]							
QSPI_DLR2	7EF911H	DL[15:8]							

DL[15: 0]：QSPI 传输的数据长度

在间接模式和状态轮询模式下待检索的数据数量（值+1）。对状态轮询模式应使用不大于 3 的值（表示 4 字节）。在间接模式下，所有位置 1 表示未定义长度，QSPI 将继续传输数据直到到达由 FSIZE 定义的存储器末尾。

DL[15: 0]	传输的数据长度
0000H	传输 1 个字节
0001H	传输 2 个字节
0002H	传输 3 个字节
0003H	传输 4 个字节
…	…
FFFEH	传输 65535 个字节
FFFFH	未定义长度 传输所有字节直到到达由 FSIZE 定义的 Flash 的结尾

26.5.11 QSPI 通信配置寄存器 1（QSPI_CCR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_CCR1	7EF914H	INSTRUCTION[7:0]							

INSTRUCTION[7:0]：指令(Instruction)。指定要发送到外部 SPI 设备的指令。该位指示在自动轮询模式期间用来确定是否匹配的方法。仅可在 BUSY = 0 时修改该位。

26.5.12 QSPI 通信配置寄存器 2（QSPI_CCR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_CCR2	7EF915H	ABMODE[1:0]		ADSIZE[1:0]		ADMODE[1:0]		IMODE[1:0]	

ABMODE[1:0]：交替字节模式。仅可在 BUSY = 0 时修改该位。

ABMODE[1:0]	交替字节模式
00	无交替字节
01	单线传输交替字节
10	双线传输交替字节
11	四线传输交替字节

ADSIZE[1:0]：地址长度。仅可在 BUSY = 0 时修改该位。

ABMODE[1:0]	地址长度
00	8 位地址
01	16 位地址
10	24 位地址
11	32 位地址

ADMODE[1:0]：地址模式。仅可在 BUSY = 0 时修改该位。

ADMODE[1:0]	地址模式
00	无地址
01	单线传输地址
10	双线传输地址
11	四线传输地址

IMODE[1:0]：指令模式。仅可在 BUSY = 0 时修改该位。

IMODE[1:0]	指令模式
00	无指令
01	单线传输指令
10	双线传输指令
11	四线传输指令

26.5.13 QSPI 通信配置寄存器 3（QSPI_CCR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_CCR3	7EF916H	-	DCYC[4:0]					ABSIZE[1:0]	

DCYC[4:0]：空指令周期数（Number of dummy cycles）。该字段定义空指令阶段的持续时间。在 SDR 模式下，它指定 CLK 周期数(0-31)。仅可在 BUSY = 0 时修改该位。

ADSIZE[1:0]：交替字节长度。仅可在 BUSY = 0 时修改该位。

ABMODE[1:0]	交替字节长度
00	8 位交替字节
01	-
10	-
11	-

26.5.14 QSPI 通信配置寄存器 4（QSPI_CCR4）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_CCR4	7EF917H	-	-	-	SIOO	FMODE[1:0]		DMODE[1:0]	

SIOO：仅发送指令一次模式。IMODE = 00 时，该位不起作用。仅可在 BUSY = 0 时修改该位。

0：在每个事务中发送指令

1：仅为第一条命令发送指令

FMODE[1:0]：设置 QSPI 操作的功能模式。仅可在 BUSY = 0 时修改该位。

FMODE [1:0]	功能模式
00	间接写入模式
01	间接读取模式
10	自动轮询模式
11	-

DMODE[1:0]：数据模式。该字段还定义空指令阶段的操作模式。仅可在 BUSY = 0 时修改该位。

DMODE[1:0]	数据模式
00	无数据
01	单线传输数据
10	双线传输数据
11	四线传输数据

26.5.15 QSPI 地址寄存器（QSPI_AR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_AR1	7EF918H	ADR[7:0]							
QSPI_AR2	7EF919H	ADR[15:8]							
QSPI_AR3	7EF91AH	ADR[23:16]							
QSPI_AR4	7EF91BH	ADR[31:24]							

ADR[31:0]：设置要发送到外部 Flash 的地址。

必须先写 ADR 的 MSB，写完 LSB 后触发操作。

26.5.16 QSPI 交替字节寄存器（QSPI_ABR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_ABR1	7EF91CH	ALTERNATED[7:0]							
QSPI_ABR2	7EF91DH	ALTERNATED [15:8]							
QSPI_ABR3	7EF91EH	ALTERNATED [23:16]							
QSPI_ABR4	7EF91FH	ALTERNATED [31:24]							

ALTERNATE[31:0]：设置要在地址后立即发送到外部 SPI 设备的可选数据。仅可在 BUSY = 0 时可写入。

26.5.17 QSPI 数据寄存器（QSPI_DR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_DR	7EF920H	DATA[7:0]							

DATA[7:0]：设置要与外部 SPI 设备交换的数据。

在间接写入模式下，写入该寄存器的数据在数据阶段发送到 Flash，在此之前则存储于 FIFO，如果 FIFO 太满，将暂停写入，直到 FIFO 具有足够的空间接受要写入的数据才继续。字节写入将在 FIFO 中增加 1 个字节

在间接读取模式下，读取该寄存器可获得（通过 FIFO）已从 Flash 接收的数据。如果 FIFO 所含字节数比读取操作要求的字节数少并且 BUSY=1，将暂停读取，直到足够的数据出现或传输完成（不分先后）才继续。字节读取将移除 FIFO 中的 1 个字节。

在自动轮询模式下，该寄存器包含最后从 Flash 读取的数据（未进行屏蔽）。

26.5.18 QSPI 状态屏蔽寄存器（QSPI_PSMKR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_PSMKR1	7EF924H	MASK[7:0]							

MASK[7:0]：状态屏蔽。对在轮询模式下接收的状态字节进行屏蔽。仅可在 BUSY = 0 时可写入。

对于位 n：

0：屏蔽在自动轮询模式下所接收数据的位 n，在匹配逻辑中不考虑其值。

1：不屏蔽在自动轮询模式下所接收数据的位 n，在匹配逻辑中考虑其值。

26.5.19 QSPI 状态匹配寄存器（QSPI_PSMAR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_PSMAR1	7EF928H	MATCH[7:0]							

MATCH[7:0]：该值将与屏蔽状态寄存器比较以进行匹配。仅可在 BUSY = 0 时可写入。

26.5.20 QSPI 轮询间隔寄存器（QSPI_PIR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
QSPI_PIR1	7EF92CH	INTERVAL[7:0]							
QSPI_PIR2	7EF92DH	INTERVAL[15:8]							

INTERVAL[15: 0]：轮询间隔（Polling interval）。自动轮询阶段读取操作之间的 CLK 周期数。

仅可在 BUSY = 0 时可写入。

27 两组 I2C 总线

产品线	I2C
STC32G144K246 系列	**两组**

STC32G144K246 系列的单片机内部集成了两组 I2C 串行总线控制器。I2C 是一种高速同步通讯总线，通讯使用 SCL（时钟线）和 SDA（数据线）两线进行同步通讯。对于 SCL 和 SDA 的端口分配，STC32G144K246 系列的单片机提供了切换模式，可将 SCL 和 SDA 切换到不同的 I/O 口上，以方便用户将一组 I2C 总线当作多组进行分时复用。

与标准 I2C 协议相比较，忽略了如下两种机制：

- 发送起始信号（START）后不进行仲裁
- 时钟信号（SCL）停留在低电平时不进行超时检测

STC32G144K246 系列的 I2C 总线提供了两种操作模式：主机模式（SCL 为输出口，发送同步时钟信号）和从机模式（SCL 为输入口，接收同步时钟信号）

27.1 I2C1

27.1.1 I2C 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW2	BAH	EAXFR	-	**I2C_S[1:0]**		-	S4_S	S3_S	S2_S

I2C_S[1:0]：I2C 功能脚选择位

I2C_S[1:0]	SCL	SDA
00	P1.5	P1.4
01	P2.5	P2.4
10	P7.7	P7.6
11	P3.2	P3.3

27.1.2 I2C 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
I2CCFG	I2C 配置寄存器	7EFE80H	ENI2C	I2CMSSL	I2CMSSPEED[5:0]						0000,0000
I2CMSCR	I2C 主机控制寄存器	7EFE81H	EI2CMSI	-	-	-	I2CMSCMD[3:0]				0xxx,0000
I2CMSST	I2C 主机状态寄存器	7EFE82H	I2CMSBUSY	I2CMSIF	-	-	-	-	I2CMSACKI	I2CMSACKO	00xx,xx10
I2CSLCR	I2C 从机控制寄存器	7EFE83H	-	EI2CSTAI	EI2CRXI	EI2CTXI	EI2CSTOI	-	-	I2CSLRST	x000,0xx0
I2CSLST	I2C 从机状态寄存器	7EFE84H	I2CSLBUSY	I2CSTAIF	I2CRXIF	I2CTXIF	I2CSTOIF	I2CTXING	I2CSLACKI	I2CSLACKO	0000,0000
I2CSLADR	I2C 从机地址寄存器	7EFE85H	I2CSLADR[6:0]							I2CMA	0000,0000
I2CTXD	I2C 数据发送寄存器	7EFE86H	I2CTXD[7:0]								0000,0000
I2CRXD	I2C 数据接收寄存器	7EFE87H	I2CRXD[7:0]								0000,0000
I2CMSAUX	I2C 主机辅助控制寄存器	7EFE88H	-	-	-	-	-	-	-	I2CWDTA	xxxx,xxx0
I2CPSCR	I2C 主机时钟分频寄存器	7EFE89H	I2CMSSPEED[13:6]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
I2CTOCR	I2C 从机超时控制寄存器	7EFD84H	I2CENTO	I2CENTOI	I2CSCALE	-	-	-	-	-	000x,xxxx
I2CTOSR	I2C 从机超时状态寄存器	7EFD85H	I2CCTOF	-	-	-	-	-	-	I2CTOIF	0xxx,xxx0
I2CTOTH	I2C 从机超时长度寄存器	7EFD86H	I2CTM[15:8]								0000,0000
I2CTOTL	I2C 从机超时长度寄存器	7EFD87H	I2CTM[7:0]								0000,0000
I2CTOTE	I2C 从机超时长度寄存器	7EFD8DH	I2CTM[23:16]								0000,0000

27.1.3 I2C 配置寄存器（I2CCFG），总线速度控制 1

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CCFG	7EFE80H	ENI2C	I2CMSSL	I2CMSSPEED[5:0]					

ENI2C：I2C 功能使能控制位

0：禁止 I2C 功能

1：允许 I2C 功能

I2CMSSL：I2C 工作模式选择位

0：从机模式

1：主机模式

27.1.4 I2C 主机时钟分频寄存器（I2CPSCR），总线速度控制 2

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CPSCR	7EFE89H	I2CMSSPEED[13:6]							

I2CMSSPEED[13:6]和 I2CCFG 中的 I2CMSSPEED[5:0]合并为 14 位 I2CMSSPEED[13:0]分频器

I2CMSSPEED[13:0]：I2C 总线速度（等待时钟数）控制

I2C 总线速度＝SYSCLK / 2 / (I2CMSSPEED * 2 + 4)

（I2C 最快速度为 SYSCLK/8）

只有当 I2C 模块工作在主机模式时，I2CMSSPEED 参数设置的等待参数才有效。此等待参数主要用于主机模式的以下几个信号：

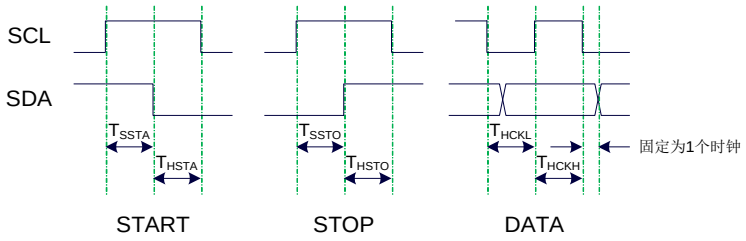
T_{SSTA}：起始信号的建立时间（Setup Time of START）

T_{HSTA}：起始信号的保持时间（Hold Time of START）

T_{SSTO}：停止信号的建立时间（Setup Time of STOP）

T_{HSTO}：停止信号的保持时间（Hold Time of STOP）

T_{HCKL}：时钟信号的低电平保持时间（Hold Time of SCL Low）



例 1：当 I2CMSSPEED＝10 时，T_{SSTA}＝T_{HSTA}＝T_{SSTO}＝T_{HSTO}＝T_{HCKL}＝24/FOSC

例 2：当 24MHz 的工作频率下需要 400K 的 I2C 总线速度时，

I2CMSSPEED＝(24M / 400K / 2 - 4) / 2＝13

27.1.5 I2C 主机控制寄存器（I2CMSCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CMSCR	7EFE81H	EI2CMSI	-	-	-	I2CMSCMD[3:0]			

EI2CMSI：主机模式中断使能控制位

0：关闭主机模式的中断

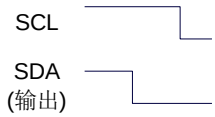
1：允许主机模式的中断

I2CMSCMD[3:0]：主机命令

0000：待机，无动作。

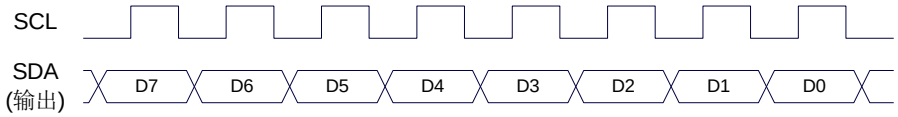
0001：起始命令。

发送 START 信号。如果当前 I2C 控制器处于空闲状态，即 I2CMSBUSY（I2CMSST.7）为 0 时，写此命令会使控制器进入忙状态，硬件自动将 I2CMSBUSY 状态位置 1，并开始发送 START 信号；**若当前 I2C 控制器处于忙状态，写此命令可触发发送 START 信号**。发送 START 信号的波形如下图所示：



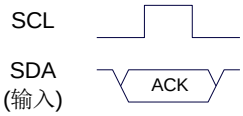
0010：发送数据命令。

写此命令后，I2C 总线控制器会在 SCL 管脚上产生 8 个时钟，并将 I2CTXD 寄存器里面数据按位送到 SDA 管脚上（先发送高位数据）。发送数据的波形如下图所示：



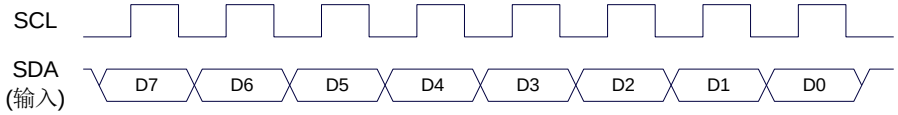
0011：接收 ACK 命令。

写此命令后，I2C 总线控制器会在 SCL 管脚上产生 1 个时钟，并将从 SDA 端口上读取的数据保存到 I2CMSACKI（I2CMSST.1）。接收 ACK 的波形如下图所示：



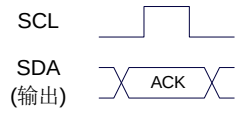
0100：接收数据命令。

写此命令后，I2C 总线控制器会在 SCL 管脚上产生 8 个时钟，并将从 SDA 端口上读取的数据依次左移到 I2CRXD 寄存器（先接收高位数据）。接收数据的波形如下图所示：



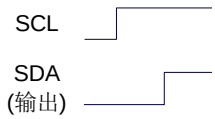
0101：发送 ACK 命令。

写此命令后，I2C 总线控制器会在 SCL 管脚上产生 1 个时钟，并将 I2CMSACKO（I2CMSST.0）中的数据发送到 SDA 端口。发送 ACK 的波形如下图所示：



0110：停止命令。

发送 STOP 信号。写此命令后，I2C 总线控制器开始发送 STOP 信号。信号发送完成后，硬件自动将 I2CMSBUSY 状态位清零。STOP 信号的波形如下图所示：



0111：保留。

1000：保留。

1001：起始命令+发送数据命令+接收 ACK 命令。

此命令为命令 0001、命令 0010、命令 0011 三个命令的组合，下此命令后控制器会依次执行这三个命令。

1010：发送数据命令+接收 ACK 命令。

此命令为命令 0010、命令 0011 两个命令的组合，下此命令后控制器会依次执行这两个命令。

1011：接收数据命令+发送 ACK(0)命令。

此命令为命令 0100、命令 0101 两个命令的组合，下此命令后控制器会依次执行这两个命令。

注意：此命令所返回的应答信号固定为 ACK（0），不受 I2CMSACKO 位的影响。

1100：接收数据命令+发送 NAK(1)命令。

此命令为命令 0100、命令 0101 两个命令的组合，下此命令后控制器会依次执行这两个命令。

注意：此命令所返回的应答信号固定为 NAK（1），不受 I2CMSACKO 位的影响。

27.1.6 I2C 主机辅助控制寄存器（I2CMSAUX）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CMSAUX	7EFE88H	-	-	-	-	-	-	-	I2CWDTA

I2CWDTA：主机模式时 I2C 数据自动发送允许位

0：禁止自动发送

1：使能自动发送

若自动发送功能被使能，当 MCU 执行完成对 I2CTXD 数据寄存器的写操作后，I2C 控制器会自动触发“1010”命令，即自动发送数据并接收 ACK 信号。

27.1.7 I2C 主机状态寄存器（I2CMSST）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CMSST	7EFE82H	I2CMSBUSY	I2CMSIF	-	-	-	-	I2CMSACKI	I2CMSACKO

I2CMSBUSY：主机模式时 I2C 控制器状态位（只读位）

0：控制器处于空闲状态

1：控制器处于忙碌状态

当 I2C 控制器处于主机模式时，在空闲状态下，发送完成 START 信号后，控制器便进入到忙碌状态，忙碌状态会一直维持到成功发送完成 STOP 信号，之后状态会再次恢复到空闲状态。

I2CMSIF：主机模式的中断请求位（中断标志位）。当处于主机模式的 I2C 控制器执行完成寄存器 I2CMSCR 中 MSCMD 命令后产生中断信号，硬件自动将此位 1，向 CPU 发请求中断，响应中断后 MSIF 位必须用软件清零。

I2CMSACKI：主机模式时，发送“0011”命令到 I2CMSCR 的 I2CMSCMD 位后所接收到的 ACK 数据。（只读位）

I2CMSACKO：主机模式时，准备将要发送出去的 ACK 信号。当发送“0101”命令到 I2CMSCR 的 I2CMSCMD 位后，控制器会自动读取此位的数据当作 ACK 发送到 SDA。

27.1.8 I2C 从机控制寄存器（I2CSLCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CSLCR	7EFE83H	-	EI2CSTAI	EI2CRXI	EI2CTXI	EI2CSTOI	-	-	I2CSLRST

EI2CSTAI：从机模式时接收到 START 信号中断允许位

0：禁止从机模式时接收到 START 信号时发生中断

1：使能从机模式时接收到 START 信号时发生中断

EI2CRXI：从机模式时接收到 1 字节数据后中断允许位

0：禁止从机模式时接收到数据后发生中断

1：使能从机模式时接收到 1 字节数据后发生中断

EI2CTXI：从机模式时发送完成 1 字节数据后中断允许位

0：禁止从机模式时发送完成数据后发生中断

1：使能从机模式时发送完成 1 字节数据后发生中断

EI2CSTOI：从机模式时接收到 STOP 信号中断允许位

0：禁止从机模式时接收到 STOP 信号时发生中断

1：使能从机模式时接收到 STOP 信号时发生中断

I2CSLRST：复位从机模式

27.1.9 I2C 从机状态寄存器（I2CSLST）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CSLST	7EFE84H	I2CSLBUSY	I2CSTAIF	I2CRXIF	I2CTXIF	I2CSTOIF	-	I2CSLACKI	I2CSLACKO

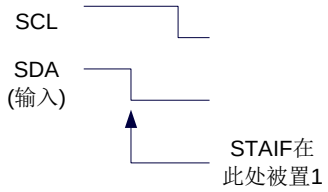
I2CSLBUSY：从机模式时 I2C 控制器状态位（只读位）

0：控制器处于空闲状态

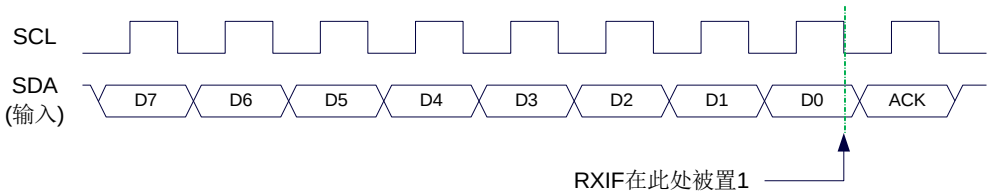
1：控制器处于忙碌状态

当 I2C 控制器处于从机模式时，在空闲状态下，接收到主机发送 START 信号后，控制器会继续检测之后的设备地址数据，若设备地址与当前 I2CSLADR 寄存器中所设置的从机地址像匹配时，控制器便进入到忙碌状态，忙碌状态会一直维持到成功接收到主机发送 STOP 信号，之后状态会再次恢复到空闲状态。

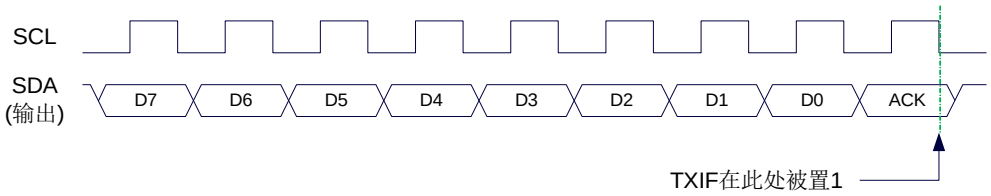
I2CSTAIF：从机模式时接收到 START 信号后的中断请求位。从机模式的 I2C 控制器接收到 START 信号后，硬件会自动将此位置 1，并向 CPU 发请求中断，响应中断后 STAIF 位必须用软件清零。STAIF 被置 1 的时间点如下图所示：



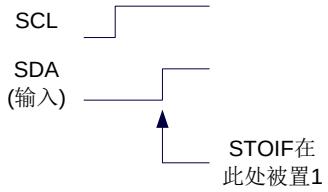
I2CRXIF：从机模式时接收到 1 字节的数据后的中断请求位。从机模式的 I2C 控制器接收到 1 字节的数据后，在第 8 个时钟的下降沿时硬件会自动将此位置 1，并向 CPU 发请求中断，响应中断后 I2CRXIF 位必须用软件清零。I2CRXIF 被置 1 的时间点如下图所示：



I2CTXIF：从机模式时发送完成 1 字节的数据后的中断请求位。从机模式的 I2C 控制器发送完成 1 字节的数据并成功接收到 1 位 ACK 信号后，在第 9 个时钟的下降沿时硬件会自动将此位置 1，并向 CPU 发请求中断，响应中断后 I2CTXIF 位必须用软件清零。TXIF 被置 1 的时间点如下图所示：

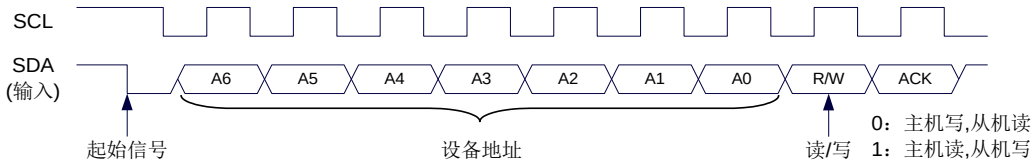


I2CSTOIF：从机模式时接收到 STOP 信号后的中断请求位。从机模式的 I2C 控制器接收到 STOP 信号后，硬件会自动将此位置 1，并向 CPU 发请求中断，响应中断后 I2CSTOIF 位必须用软件清零。I2CSTOIF 被置 1 的时间点如下图所示：



I2CSLACKI：从机模式时，接收到的 ACK 数据。

I2CSLACKO：从机模式时，准备将要发送出去的 ACK 信号。



27.1.10 I2C 从机地址寄存器（I2CSLADR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CSLADR	7EFE85H	I2CSLADR[6:0]							I2CMA

I2CSLADR[6:0]：从机设备地址

当 I2C 控制器处于从机模式时，控制器在接收到 START 信号后，会继续检测接下来主机发送出的设备地址数据以及读/写信号。当主机发送出的设备地址与 SLADR[6:0]中所设置的从机设备地址相匹配时，控制器才会向 CPU 发出中断求，请求 CPU 处理 I2C 事件；否则若设备地址不匹配，I2C 控制器继续继续监控，等待下一个起始信号，对下一个设备地址继续匹配。

I2CMA：从机设备地址匹配控制

0：设备地址必须与 I2CSLADR[6:0]继续匹配

1：忽略 I2CSLADR 中的设置，匹配所有的设备地址

27.1.11 I2C 数据寄存器（I2CTXD，I2CRXD）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CTXD	7EFE86H								
I2CRXD	7EFE87H								

I2CTXD 是 I2C 发送数据寄存器，存放将要发送的 I2C 数据

I2CRXD 是 I2C 接收数据寄存器，存放接收完成的 I2C 数据

27.1.12 I2C 从机超时控制寄存器（I2CTOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CTOCR	7EFD84H	I2CENTO	I2CENTOI	I2CSCALE	-	-	-	-	-

I2CENTO：I2C 从机超时功能控制位

0：禁止 I2C 从机超时功能

1：使能 I2C 从机超时功能（**注：I2C 的主机模式不要使能接收超时功能**）

I2CENTOI：I2C 从机超时中断控制位

0：禁止 I2C 从机超时中断

1：使能 I2C 从机超时中断

I2CSCALE：I2C 超时计数时钟源选择

0：1us 时钟（1MHz 时钟）。（注意：如需要使用此时钟源，用户必须先正确设置 IAP_TPS 寄存器。例如：若系统时钟为 12MHz，则需要将 IAP_TPS 设置为 12；若系统时钟为 22.1184MHz，则需要将 IAP_TPS 设置为 22；其他频率以此类推。特别注意，此 1us 的时钟并不是精准时间）

1：系统时钟

27.1.13 I2C 从机超时状态寄存器（I2CTOSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CTOSR	7EFD85H	I2CCTOIF	-	-	-	-	-	-	I2CTOIF

I2CCTOIF：写“1”清除 I2C 超时中断标志位 I2CTOIF。（只写）

I2CTOIF：I2C 超时中断请求标志位。（只读）

当发生 I2C 超时时，I2CTOIF 会被硬件置“1”。如果 I2CENTOI 为 1 时会产生 I2C 中断，中断入口地址为原 I2C 的中断入口地址。标志位需要软件向 I2CCTOIF 位写“1”清零

27.1.14 I2C 从机超时长度控制寄存器（I2CTOTE/H/L）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2CTOTE	7EFD8DH	I2CTM[23:16]							
I2CTOTH	7EFD86H	I2CTM[15:8]							
I2CTOTL	7EFD87H	I2CTM[7:0]							

I2CTM[23:0]：I2C 超时时间控制位。

当 I2C 处于接收空闲状态时，内部超时计数器根据 SCALE 寄存器所选择的时钟源进行计数，当计数时间达到 I2CTM 所设置的超时时间时，便会产生超时中断。当 I2C 接收数据完成时，复位内部超时计数器，重新进行超时计数。

注：

1、如果需要使能接收超时中断功能，则 I2CTM 不可设置为 0，且 I2CTOTL、I2CTOTH、I2CTOTE

寄存器的设置必须先设置 I2CTOTL 和 I2CTOTH，最后设置 I2CTOTE

2、必须使能 I2C 为从机模式才有接收超时功能

3、接收超时功能必须在正确接收到一字节数据后才能触发

4、正在接收过程中，无接收超时功能

27.2 I2C2

27.2.1 I2C2 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX4	7EFD6CH	**I2C2_S[1:]**		-	-	SPI3_S[1:0]		SPI2_S[1:0]	

I2C2_S[1:0]：I2C2 功能脚选择位 SPI2_S[1:0]：SPI2 功能脚选择位

I2C2_S[1:0]	SCL2	SDA2
00	P2.7	P2.6
01	P3.7	P3.6
10	P1.7	P1.6
11	P6.1	P6.0

27.2.2 I2C2 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
I2C2CFG	I2C2 配置寄存器	7EF860H	ENI2C2	I2C2MSSL	I2C2MSSPEED[5:0]						0000,0000
I2C2MSCR	I2C2 主机控制寄存器	7EF861H	EI2C2MSI	-	-	-	I2C2MSCMD[3:0]				0xxx,0000
I2C2MSST	I2C2 主机状态寄存器	7EF862H	I2C2MSBUSY	I2C2MSIF	-	-	-	-	I2C2MSACKI	I2C2MSACKO	00xx,xx10
I2C2SLCR	I2C2 从机控制寄存器	7EF863H	-	EI2C2STAI	EI2C2RXI	EI2C2TXI	EI2C2STOI	-	-	I2C2SLRST	x000,0xx0
I2C2SLST	I2C2 从机状态寄存器	7EF864H	I2C2SLBUSY	I2C2STAIF	I2C2RXIF	I2C2TXIF	I2C2STOIF	I2C2TXING	I2C2SLACKI	I2C2SLACKO	0000,0000
I2C2SLADR	I2C2 从机地址寄存器	7EF865H	I2C2SLADR[6:0]							I2C2MA	0000,0000
I2C2TXD	I2C2 数据发送寄存器	7EF866H	I2C2TXD[7:0]								0000,0000
I2C2RXD	I2C2 数据接收寄存器	7EF867H	I2C2RXD[7:0]								0000,0000
I2C2MSAUX	I2C2 主机辅助控制寄存器	7EF868H	-	-	-	-	-	-	-	I2C2WDTA	xxxx,xxx0
I2C2PSCR	I2C2 主机时钟分频寄存器	7EF869H	I2C2MSSPEED[13:6]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
I2C2TOCR	I2C2 从机超时控制寄存器	7EF86CH	I2C2ENTO	I2C2ENTOI	I2C2SCALE	-	-	-	-	-	000x,xxxx
I2C2TOSR	I2C2 从机超时状态寄存器	7EF86DH	-	-	-	-	-	-	-	I2C2TOIF	0xxx,xxx0
I2C2TOTH	I2C2 从机超时长度寄存器	7EF86EH	I2C2TM[15:8]								0000,0000
I2C2TOTL	I2C2 从机超时长度寄存器	7EF86FH	I2C2TM[7:0]								0000,0000
I2C2TOTE	I2C2 从机超时长度寄存器	7EF86BH	I2C2TM[23:16]								0000,0000

27.2.3 I2C2 配置寄存器（I2C2CFG），总线速度控制 1

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2CFG	7EF860H	ENI2C2	I2C2MSSL	I2C2MSSPEED[5:0]					

ENI2C2：I2C2 功能使能控制位

0：禁止 I2C2 功能

1：允许 I2C2 功能

I2C2MSSL：I2C2 工作模式选择位

0：从机模式

1：主机模式

27.2.4 I2C2 主机时钟分频寄存器（I2C2PSCR），总线速度控制 2

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2PSCR	7EF869H	I2C2MSSPEED[13:6]							

I2C2MSSPEED[13:6]和 I2C2CFG 中的 I2C2MSSPEED[5:0]合并为 14 位 I2C2MSSPEED[13:0]分频器

I2C2MSSPEED[13:0]：I2C2 总线速度（等待时钟数）控制

I2C2 总线速度＝SYSCLK / 2 / (I2C2MSSPEED * 2 + 4)

（I2C2 最快速度为 SYSCLK/8）

只有当 I2C2 模块工作在主机模式时，I2C2MSSPEED 参数设置的等待参数才有效。此等待参数主要用于主机模式的以下几个信号：

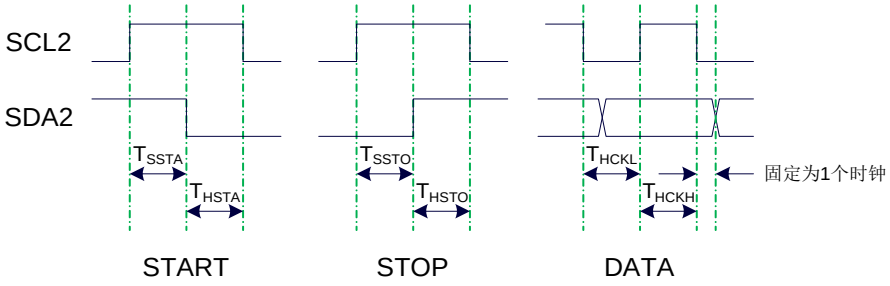
T_{SSTA}：起始信号的建立时间（Setup Time of START）

T_{HSTA}：起始信号的保持时间（Hold Time of START）

T_{SSTO}：停止信号的建立时间（Setup Time of STOP）

T_{HSTO}：停止信号的保持时间（Hold Time of STOP）

T_{HCKL}：时钟信号的低电平保持时间（Hold Time of SCL2 Low）



例 1：当 I2C2MSSPEED＝10 时，T_{SSTA}＝T_{HSTA}＝T_{SSTO}＝T_{HSTO}＝T_{HCKL}＝24/FOSC

例 2：当 24MHz 的工作频率下需要 400K 的 I2C2 总线速度时，

I2C2MSSPEED＝(24M / 400K / 2 - 4) / 2＝13

27.2.5 I2C2 主机控制寄存器（I2C2MSCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2MSCR	7EF861H	EI2C2MSI	-	-	-	I2C2MSCMD[3:0]			

EI2C2MSI：主机模式中断使能控制位

0：关闭主机模式的中断

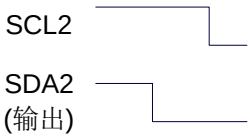
1：允许主机模式的中断

I2C2MSCMD[3:0]：主机命令

0000：待机，无动作。

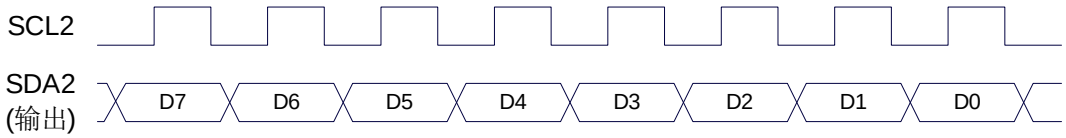
0001：起始命令。

发送 START 信号。如果当前 I2C2 控制器处于空闲状态，即 I2C2MSBUSY（I2C2MSST.7）为 0 时，写此命令会使控制器进入忙状态，硬件自动将 I2C2MSBUSY 状态位置 1，并开始发送 START 信号；**若当前 I2C2 控制器处于忙状态，写此命令可触发发送 START 信号**。发送 START 信号的波形如下图所示：



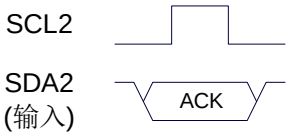
0010：发送数据命令。

写此命令后，I2C2 总线控制器会在 SCL2 管脚上产生 8 个时钟，并将 I2C2TXD 寄存器里面数据按位送到 SDA2 管脚上（先发送高位数据）。发送数据的波形如下图所示：



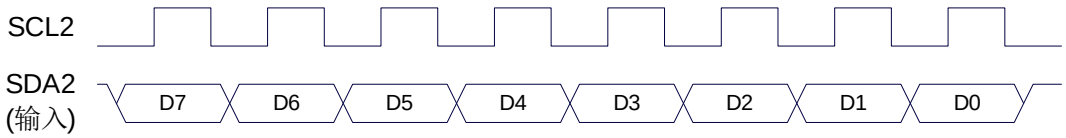
0011：接收 ACK 命令。

写此命令后，I2C2 总线控制器会在 SCL2 管脚上产生 1 个时钟，并将从 SDA2 端口上读取的数据保存到 I2C2MSACKI（I2C2MSST.1）。接收 ACK 的波形如下图所示：



0100：接收数据命令。

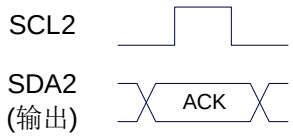
写此命令后，I2C2 总线控制器会在 SCL2 管脚上产生 8 个时钟，并将从 SDA2 端口上读取的数据依次左移到 I2C2RXD 寄存器（先接收高位数据）。接收数据的波形如下图所示：



0101：发送 ACK 命令。

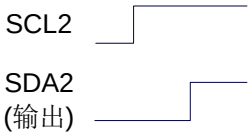
写此命令后，I2C2 总线控制器会在 SCL2 管脚上产生 1 个时钟，并将 I2C2MSACKO

（I2C2MSST.0）中的数据发送到 SDA2 端口。发送 ACK 的波形如下图所示：



0110：停止命令。

发送 STOP 信号。写此命令后，I2C2 总线控制器开始发送 STOP 信号。信号发送完成后，硬件自动将 I2C2MSBUSY 状态位清零。STOP 信号的波形如下图所示：



0111：保留。

1000：保留。

1001：起始命令+发送数据命令+接收 ACK 命令。

此命令为命令 0001、命令 0010、命令 0011 三个命令的组合，下此命令后控制器会依次执行这三个命令。

1010：发送数据命令+接收 ACK 命令。

此命令为命令 0010、命令 0011 两个命令的组合，下此命令后控制器会依次执行这两个命令。

1011：接收数据命令+发送 ACK(0)命令。

此命令为命令 0100、命令 0101 两个命令的组合，下此命令后控制器会依次执行这两个命令。

注意：此命令所返回的应答信号固定为 ACK（0），不受 I2C2MSACKO 位的影响。

1100：接收数据命令+发送 NAK(1)命令。

此命令为命令 0100、命令 0101 两个命令的组合，下此命令后控制器会依次执行这两个命令。

注意：此命令所返回的应答信号固定为 NAK（1），不受 I2C2MSACKO 位的影响。

27.2.6 I2C2 主机辅助控制寄存器（I2C2MSAUX）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2MSAUX	7EF868H	-	-	-	-	-	-	-	I2C2WDTA

I2C2WDTA：主机模式时 I2C2 数据自动发送允许位

0：禁止自动发送

1：使能自动发送

若自动发送功能被使能，当 MCU 执行完成对 I2C2TXD 数据寄存器的写操作后，I2C2 控制器会自动触发“1010”命令，即自动发送数据并接收 ACK 信号。

27.2.7 I2C2 主机状态寄存器（I2C2MSST）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2MSST	7EF862H	I2C2MSBUSY	I2C2MSIF	-	-	-	-	I2C2MSACKI	I2C2MSACKO

I2C2MSBUSY：主机模式时 I2C2 控制器状态位（只读位）

0：控制器处于空闲状态

1：控制器处于忙碌状态

当 I2C2 控制器处于主机模式时，在空闲状态下，发送完成 START 信号后，控制器便进入到忙碌状态，忙碌状态会一直维持到成功发送完成 STOP 信号，之后状态会再次恢复到空闲状态。

I2C2MSIF：主机模式的中断请求位（中断标志位）。当处于主机模式的 I2C2 控制器执行完成寄存器 I2C2MSCR 中 MSCMD 命令后产生中断信号，硬件自动将此位 1，向 CPU 发请求中断，响应中断后 MSIF 位必须用软件清零。

I2C2MSACKI：主机模式时，发送“0011”命令到 I2C2MSCR 的 I2C2MSCMD 位后所接收到的 ACK 数据。（只读位）

I2C2MSACKO：主机模式时，准备将要发送出去的 ACK 信号。当发送“0101”命令到 I2C2MSCR 的 I2C2MSCMD 位后，控制器会自动读取此位的数据当作 ACK 发送到 SDA2。

27.2.8 I2C2 从机控制寄存器（I2C2SLCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2SLCR	7EF863H	-	EI2C2STAI	EI2C2RXI	EI2C2TXI	EI2C2STOI	-	-	I2C2SLRST

EI2C2STAI：从机模式时接收到 START 信号中断允许位

0：禁止从机模式时接收到 START 信号时发生中断

1：使能从机模式时接收到 START 信号时发生中断

EI2C2RXI：从机模式时接收到 1 字节数据后中断允许位

0：禁止从机模式时接收到数据后发生中断

1：使能从机模式时接收到 1 字节数据后发生中断

EI2C2TXI：从机模式时发送完成 1 字节数据后中断允许位

0：禁止从机模式时发送完成数据后发生中断

1：使能从机模式时发送完成 1 字节数据后发生中断

EI2C2STOI：从机模式时接收到 STOP 信号中断允许位

0：禁止从机模式时接收到 STOP 信号时发生中断

1：使能从机模式时接收到 STOP 信号时发生中断

I2C2SLRST：复位从机模式

27.2.9 I2C2 从机状态寄存器（I2C2SLST）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2SLST	7EF864H	I2C2SLBUSY	I2C2STAIF	I2C2RXIF	I2C2TXIF	I2C2STOIF	-	I2C2SLACKI	I2C2SLACKO

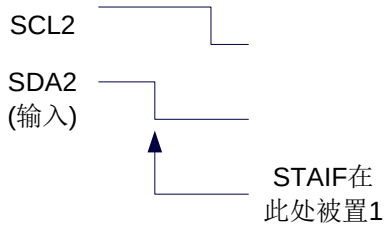
I2C2SLBUSY：从机模式时 I2C2 控制器状态位（只读位）

0：控制器处于空闲状态

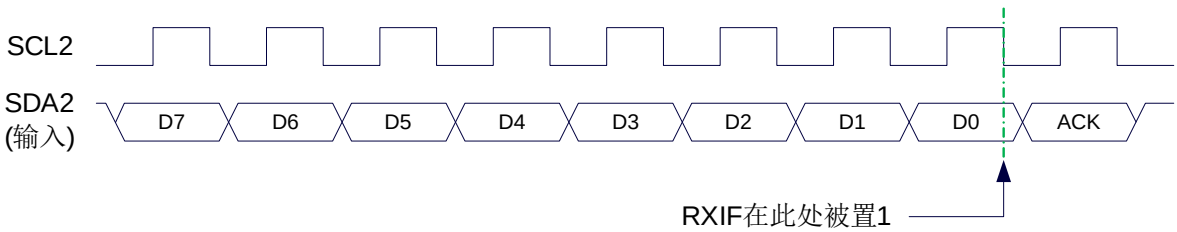
1：控制器处于忙碌状态

当 I2C2 控制器处于从机模式时，在空闲状态下，接收到主机发送 START 信号后，控制器会继续检测之后的设备地址数据，若设备地址与当前 I2C2SLADR 寄存器中所设置的从机地址像匹配时，控制器便进入到忙碌状态，忙碌状态会一直维持到成功接收到主机发送 STOP 信号，之后状态会再次恢复到空闲状态。

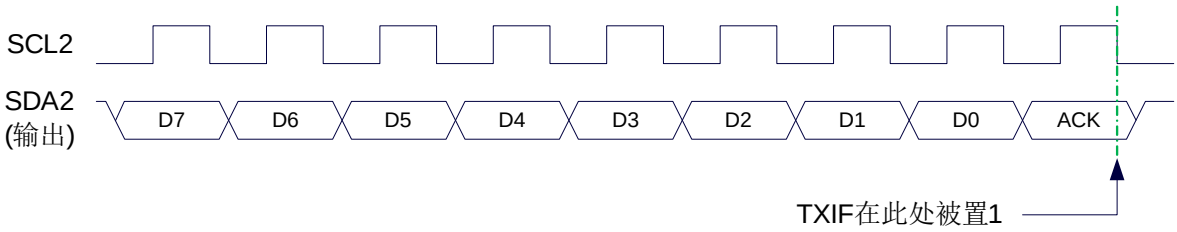
I2C2STAIF：从机模式时接收到 START 信号后的中断请求位。从机模式的 I2C2 控制器接收到 START 信号后，硬件会自动将此位置 1，并向 CPU 发请求中断，响应中断后 STAIF 位必须用软件清零。STAIF 被置 1 的时间点如下图所示：



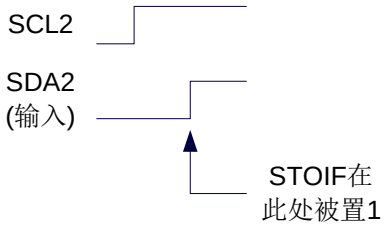
I2C2RXIF：从机模式时接收到 1 字节的数据后的中断请求位。从机模式的 I2C2 控制器接收到 1 字节的数据后，在第 8 个时钟的下降沿时硬件会自动将此位置 1，并向 CPU 发请求中断，响应中断后 I2C2RXIF 位必须用软件清零。I2C2RXIF 被置 1 的时间点如下图所示：



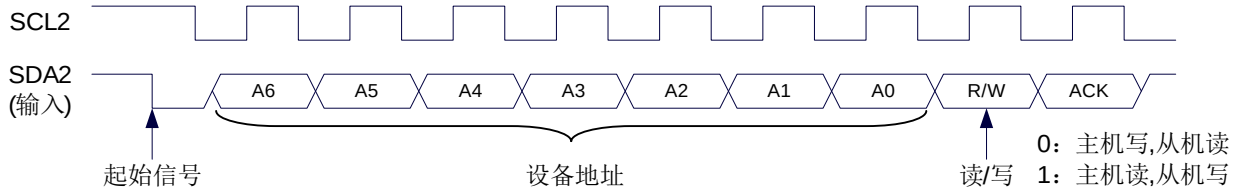
I2C2TXIF：从机模式时发送完成 1 字节的数据后的中断请求位。从机模式的 I2C2 控制器发送完成 1 字节的数据并成功接收到 1 位 ACK 信号后，在第 9 个时钟的下降沿时硬件会自动将此位置 1，并向 CPU 发请求中断，响应中断后 I2C2TXIF 位必须用软件清零。TXIF 被置 1 的时间点如下图所示：



I2C2STOIF：从机模式时接收到 STOP 信号后的中断请求位。从机模式的 I2C2 控制器接收到 STOP 信号后，硬件会自动将此位置 1，并向 CPU 发请求中断，响应中断后 I2C2STOIF 位必须用软件清零。I2C2STOIF 被置 1 的时间点如下图所示：



I2C2SLACKI：从机模式时，接收到的 ACK 数据。
I2C2SLACKO：从机模式时，准备将要发送出去的 ACK 信号。



27.2.10 I2C2 从机地址寄存器（I2C2SLADR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2SLADR	7EF865H	I2C2SLADR[6:0]							I2C2MA

I2C2SLADR[6:0]：从机设备地址

当 I2C2 控制器处于从机模式时，控制器在接收到 START 信号后，会继续检测接下来主机发送出的设备地址数据以及读/写信号。当主机发送出的设备地址与 SLADR[6:0]中所设置的从机设备地址相匹配时，控制器才会向 CPU 发出中断求，请求 CPU 处理 I2C2 事件；否则若设备地址不匹配，I2C2 控制器继续继续监控，等待下一个起始信号，对下一个设备地址继续匹配。

I2C2MA：从机设备地址匹配控制

0：设备地址必须与 I2C2SLADR[6:0]继续匹配

1：忽略 I2C2SLADR 中的设置，匹配所有的设备地址

27.2.11 I2C2 数据寄存器（I2C2TXD，I2C2RXD）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2TXD	7EF866H								
I2C2RXD	7EF867H								

I2C2TXD 是 I2C2 发送数据寄存器，存放将要发送的 I2C2 数据

I2C2RXD 是 I2C2 接收数据寄存器，存放接收完成的 I2C2 数据

27.2.12 I2C2 从机超时控制寄存器（I2C2TOCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2TOCR	7EF86CH	I2C2ENTO	I2C2ENTOI	I2C2SCALE	-	-	-	-	-

I2C2ENTO：I2C2 从机超时功能控制位

0：禁止 I2C2 从机超时功能

1：使能 I2C2 从机超时功能（**注：I2C2 的主机模式不要使能接收超时功能**）

I2C2ENTOI：I2C2 从机超时中断控制位

0：禁止 I2C2 从机超时中断

1：使能 I2C2 从机超时中断

I2C2SCALE：I2C2 超时计数时钟源选择

0：1us 时钟（1MHz 时钟）。（注意：如需要使用此时钟源，用户必须先正确设置 IAP_TPS 寄存器。例如：若系统时钟为 12MHz，则需要将 IAP_TPS 设置为 12；若系统时钟为 22.1184MHz，则需要将 IAP_TPS 设置为 22；其他频率以此类推。特别注意，此 1us 的时钟并不是精准时间）

1：系统时钟

27.2.13 I2C2 从机超时状态寄存器（I2C2TOSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2TOSR	7EF86DH	I2C2CTOIF	-	-	-	-	-	-	I2C2TOIF

I2C2CTOIF：写“1”清除 I2C2 超时中断标志位 I2C2TOIF。（只写）

I2C2TOIF：I2C2 超时中断请求标志位。（只读）

当发生 I2C2 超时时，I2C2TOIF 会被硬件置“1”。如果 I2C2ENTOI 为 1 时会产生 I2C2 中断，中断入口地址为原 I2C2 的中断入口地址。标志位需要软件向 I2C2CTOIF 位写“1”清零

27.2.14 I2C2 从机超时长度控制寄存器（I2C2TOTE/H/L）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2C2TOTE	7EF86BH	I2C2TM[23:16]							
I2C2TOTH	7EF86EH	I2C2TM[15:8]							
I2C2TOTL	7EF86FH	I2C2TM[7:0]							

I2C2TM[23:0]：I2C2 超时时间控制位。

当 I2C2 处于接收空闲状态时，内部超时计数器根据 SCALE 寄存器所选择的时钟源进行计数，当计数时间达到 I2C2TM 所设置的超时时间时，便会产生超时中断。当 I2C2 接收数据完成时，复位内部超时计数器，重新进行超时计数。

注：

1、如果需要使能接收超时中断功能，则 I2C2TM 不可设置为 0，且 I2C2TOTL、I2C2TOTH、I2C2TOTE 寄存器的设置必须先设置 I2C2TOTL 和 I2C2TOTH，最后设置 I2C2TOTE

2、必须使能 I2C2 为从机模式才有接收超时功能

3、接收超时功能必须在正确接收到一字节数据后才能触发

4、正在接收过程中，无接收超时功能

28 六组 16 位高级 PWM 定时器，支持正交编码

产品线	高级 PWM
STC32G144K246 系列	**六组**

STC32G144K246 系列的单片机内部集成了 6 组高级 PWM 定时器，分别命名为 PWMA、PWMC、PWME 和 PWMB、PWMD、PWMF，可分别单独设置。第 1/3/5 组 PWM（PWMA、PWMC、PWME）可配置成 4 组互补/对称/死区控制的 PWM 或捕捉外部信号，第 2/4/6 组 PWM（PWMB、PWMD、PWMF）可配置成 4 路 PWM 输出或捕捉外部信号。

第 1 组 PWM 定时器，PWMA 有 4 个通道（PWMA1P/PWMA1N、PWMA2P/PWMA2N、PWMA3P/PWMA3N、PWMA4P/PWMA4N），每个通道都可独立实现 PWM 输出（可设置带死区的互补对称 PWM 输出）、捕获和比较功能；

第 2 组 PWM 定时器，PWMB 有 4 个通道（PWMB5、PWMB6、PWMB7、PWMB8），每个通道也可独立实现 PWM 输出、捕获和比较功能。

第 3 组 PWM 定时器，PWMC 有 4 个通道（PWMC1P/PWMC1N、PWMC2P/PWMC2N、PWMC3P/PWMC3N、PWMC4P/PWMC4N），每个通道都可独立实现 PWM 输出（可设置带死区的互补对称 PWM 输出）、捕获和比较功能；

第 4 组 PWM 定时器，PWMD 有 4 个通道（PWMD5、PWMD6、PWMD7、PWMD8），每个通道也可独立实现 PWM 输出、捕获和比较功能。

第 5 组 PWM 定时器，PWME 有 4 个通道（PWME1P/PWME1N、PWME2P/PWME2N、PWME3P/PWME3N、PWME4P/PWME4N），每个通道都可独立实现 PWM 输出（可设置带死区的互补对称 PWM 输出）、捕获和比较功能；

第 6 组 PWM 定时器，PWMF 有 4 个通道（PWMF5、PWMF6、PWMF7、PWMF8），每个通道也可独立实现 PWM 输出、捕获和比较功能。

PWMA/PWMC/PWME 功能完全相同，可输出带死区的互补对称 PWM

PWMB/PWMD/PWMF 功能完全相同，只能输出单端的 PWM

下面关于高级 PWM 定时器的介绍只以第一组为例进行说明。

当使用 PWMA / PWMC / PWME 定时器输出 PWM 波形时，可单独使能 PWMx1P / PWMx2P / PWMx3P / PWMx4P 输出，也可单独使能 PWMx1N / PWMx2N / PWMx3N / PWMx4N 输出。例如：若单独使能了 PWMx1P 输出，则 PWMx1N 就不能再独立输出，除非 PWMx1P 和 PWMx1N 组成一组互补对称输出。PWMA / PWMC / PWME 的 4 路输出是可分别独立设置的，例如：可单独使能 PWMx1P 和 PWMx2N 输出，也可单独使能 PWMx2N 和 PWMx3N 输出。若需要使用 PWM 定时器进行捕获功能或者测量脉宽时，输入信号只能从每路的正端输入，即只有 PWMx1P / PWMx2P / PWMx3P / PWMx4P 才有捕获功能和测量脉宽功能。

6 组高级 PWM 定时器对外部信号进行捕获时，可选择上升沿捕获或者下降沿捕获。如果需要同时捕获上升沿和下降沿，则可将输入信号同时接入到两路 PWM，使能其中一路捕获上升沿，另外一路捕获下降沿即可。**更强悍的是，将外部输入信号同时接入到两路 PWM 时，可同时捕获信号的周期值和占空比值**。

三种硬件 PWM 比较：

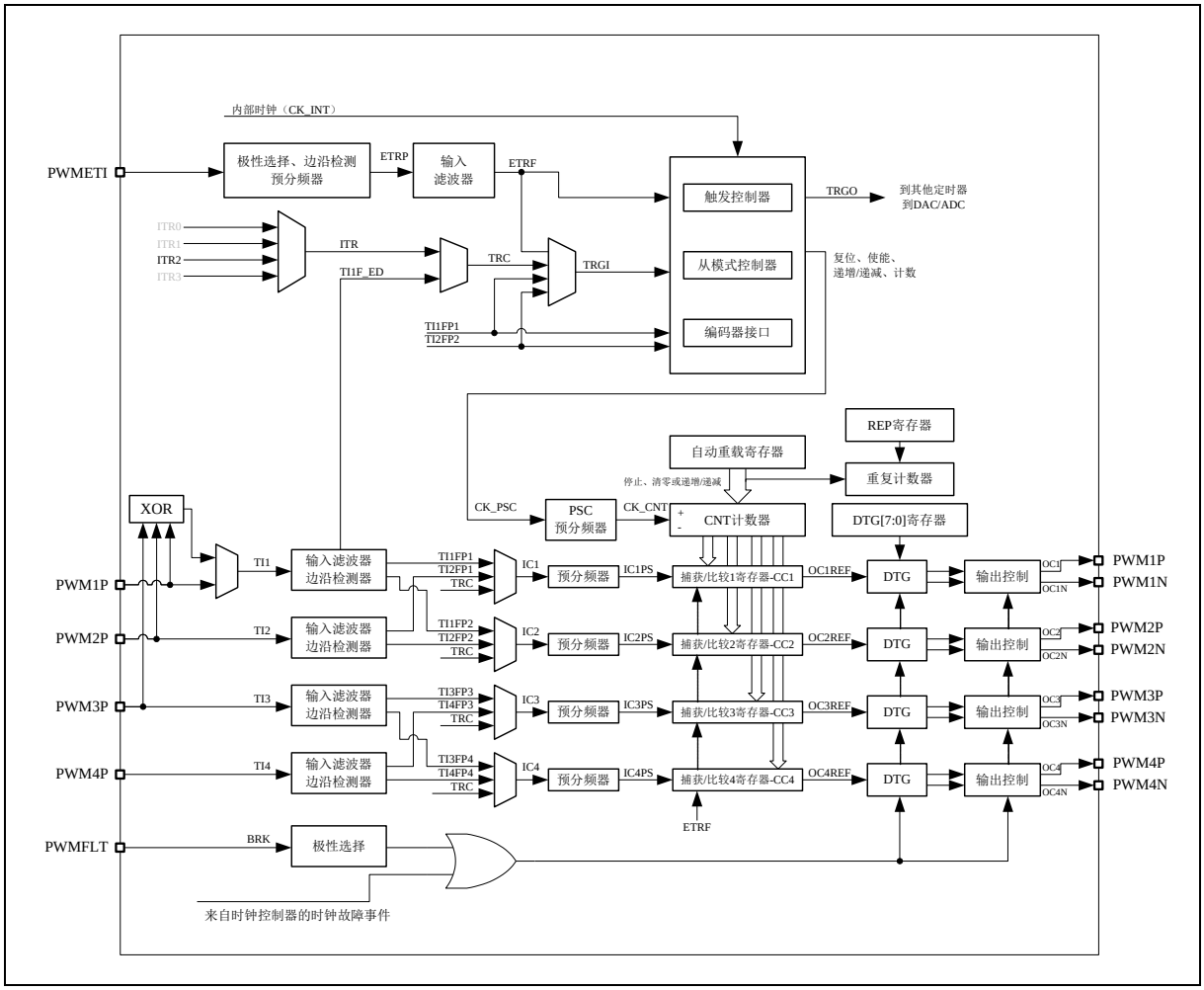
兼容传统 8051 的 PCA/CCP/PWM：可输出 PWM 波形、捕获外部输入信号以及输出高速脉冲。可对外输出 6 位/7 位/8 位/10 位的 PWM 波形，6 位 PWM 波形的频率为 PCA 模块时钟源频率/64；7 位 PWM 波形的频率为 PCA 模块时钟源频率/128；8 位 PWM 波形的频率为 PCA 模块时钟源频率/256；10 位 PWM 波形的频率为 PCA 模块时钟源频率/1024。捕获外部输入信号，可捕获上升沿、下降沿或者同时捕获上升沿和下降沿。

AI32G 系列的 15 位增强型 PWM：只能对外输出 PWM 波形，无输入捕获功能。对外输出 PWM 的频率以及占空比均可任意设置。通过软件干预，可实现多路互补/对称/带死区的 PWM 波形。有外部异常检测功能以及实时触发 ADC 转换功能。

AI32G/AI8H/STC32G144K246 系列的 16 位高级 PWM 定时器：是目前功能最强的 PWM，可对外输出任意频率以及任意占空比的 PWM 波形。无需软件干预即可输出互补/对称/带死区的 PWM 波形。能捕获外部输入信号，可捕获上升沿、下降沿或者同时捕获上升沿和下降沿，测量外部波形时，可同时测量波形的周期值和占空比值。有正交编码功能、外部异常检测功能以及实时触发 ADC 转换功能。

下面的说明中，　　PWMA 代表第 1 组 PWM 定时器，
PWMB 代表第 2 组 PWM 定时器，
PWMC 代表第 3 组 PWM 定时器，
PWMD 代表第 4 组 PWM 定时器，
PWME 代表第 5 组 PWM 定时器，
PWMF 代表第 6 组 PWM 定时器

28.1 高级 PWMA/PWMC/PWME 定时器内部结构框图



TI1： 外部时钟输入信号 1（PWM1P 管脚信号或者 PWM1P/PWM2P/PWM3P 相异或后的信号）
TI1F： 经过 IC1F 数字滤波后的 TI1 信号
TI1FP： 经过 CC1P/CC2P 边沿检测器后的 TI1F 信号
TI1F_ED：TI1F 的边沿信号
TI1FP1： 经过 CC1P 边沿检测器后的 TI1F 信号
TI1FP2： 经过 CC2P 边沿检测器后的 TI1F 信号
IC1： 通过 CC1S 选择的通道 1 的捕获输入信号
OC1REF：输出通道 1 输出的参考波形（中间波形）
OC1： 通道 1 的主输出信号（经过 CC1P 极性处理后的 OC1REF 信号）
OC1N： 通道 1 的互补输出信号（经过 CC1NP 极性处理后的 OC1REF 信号）
CC1： 通道 1 的捕获/比较寄存器

TI2： 外部时钟输入信号 2（PWM2P 管脚信号）
TI2F： 经过 IC2F 数字滤波后的 TI2 信号
TI2F_ED：TI2F 的边沿信号
TI2FP： 经过 CC1P/CC2P 边沿检测器后的 TI2F 信号
TI2FP1： 经过 CC1P 边沿检测器后的 TI2F 信号
TI2FP2： 经过 CC2P 边沿检测器后的 TI2F 信号
IC2： 通过 CC2S 选择的通道 2 的捕获输入信号

OC2REF：输出通道 2 输出的参考波形（中间波形）
OC2：　通道 2 的主输出信号（经过 CC2P 极性处理后的 OC2REF 信号）
OC2N：　通道 2 的互补输出信号（经过 CC2NP 极性处理后的 OC2REF 信号）
CC2：　通道 2 的捕获/比较寄存器

TI3：　外部时钟输入信号 3（PWM3P 管脚信号）
TI3F：　经过 IC3F 数字滤波后的 TI3 信号
TI3F_ED：TI3F 的边沿信号
TI3FP：　经过 CC3P/CC4P 边沿检测器后的 TI3F 信号
TI3FP3：　经过 CC3P 边沿检测器后的 TI3F 信号
TI3FP4：　经过 CC4P 边沿检测器后的 TI3F 信号
IC3：　通过 CC3S 选择的通道 3 的捕获输入信号
OC3REF：输出通道 3 输出的参考波形（中间波形）
OC3：　通道 3 的主输出信号（经过 CC3P 极性处理后的 OC3REF 信号）
OC3N：　通道 3 的互补输出信号（经过 CC3NP 极性处理后的 OC3REF 信号）
CC3：　通道 3 的捕获/比较寄存器

TI4：　外部时钟输入信号 4（PWM4P 管脚信号）
TI4F：　经过 IC4F 数字滤波后的 TI4 信号
TI4F_ED：TI4F 的边沿信号
TI4FP：　经过 CC3P/CC4P 边沿检测器后的 TI4F 信号
TI4FP3：　经过 CC3P 边沿检测器后的 TI4F 信号
TI4FP4：　经过 CC4P 边沿检测器后的 TI4F 信号
IC4：　通过 CC4S 选择的通道 4 的捕获输入信号
OC4REF：输出通道 4 输出的参考波形（中间波形）
OC4：　通道 4 的主输出信号（经过 CC4P 极性处理后的 OC4REF 信号）
OC4N：　通道 4 的互补输出信号（经过 CC4NP 极性处理后的 OC4REF 信号）
CC4：　通道 4 的捕获/比较寄存器

ITR1：　内部触发输入信号 1
ITR2：　内部触发输入信号 2

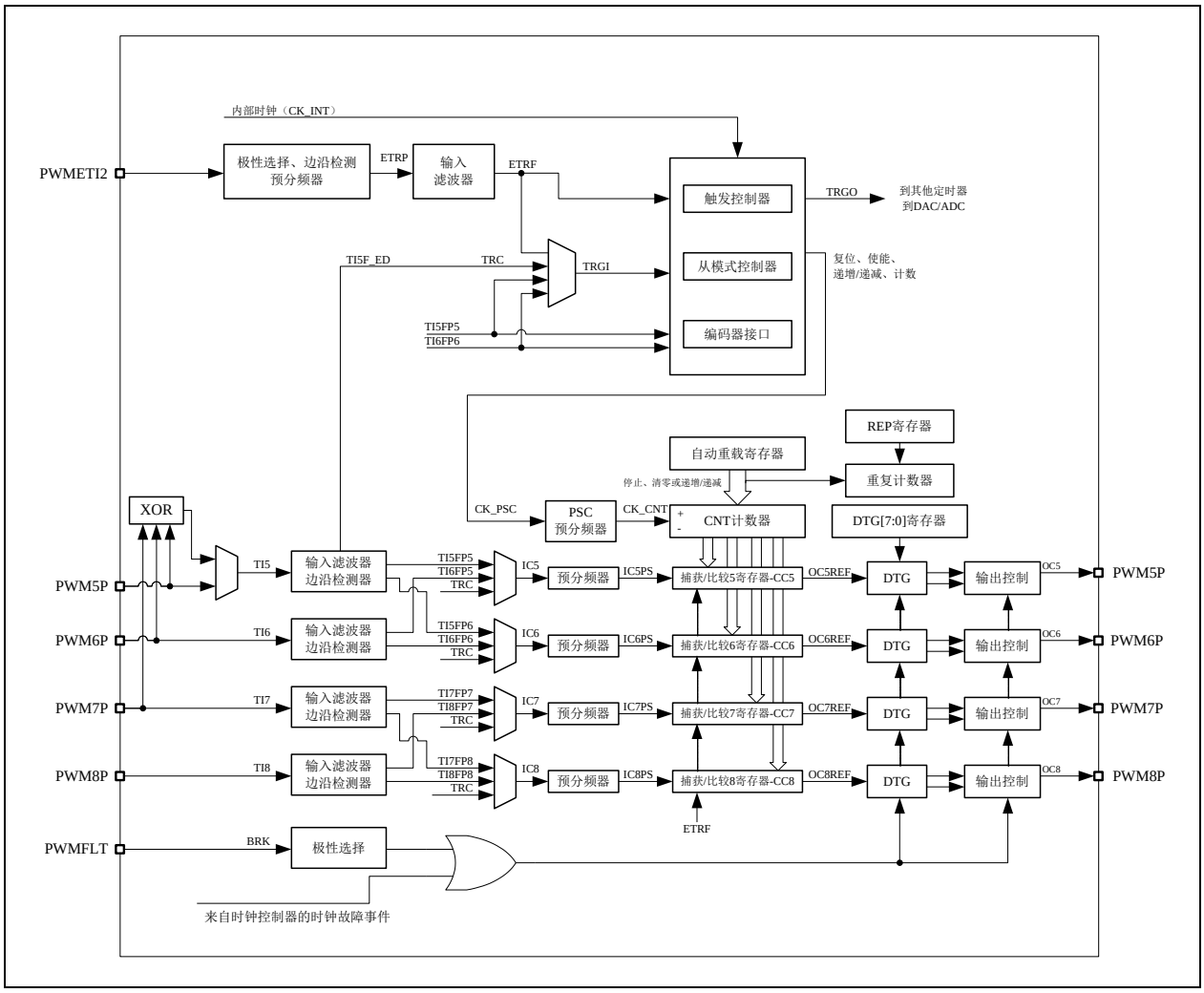
TRC：　固定为 TI1_ED
TRGI：　经过 TS 多路选择器后的触发输入信号
TRGO：　经过 MMS 多路选择器后的触发输出信号

ETR：　外部触发输入信号（PWMETI1 管脚信号）
ETRP：　经过 ETP 边沿检测器以及 ETPS 分频器后的 ETR 信号
ETRF：　经过 ETF 数字滤波后的 ETRP 信号

BRK：　刹车输入信号（PWMFLT）

CK_PSC：预分频时钟，PWMA_PSCR 预分频器的输入时钟
CK_CNT：PWMA_PSCR 预分频器的输出时钟，PWM 定时器的时钟

28.2 高级 PWMB/PWMD/PWMF 定时器内部结构框图



TI5：　外部时钟输入信号 5（PWM5 管脚信号或者 PWM5/PWM6/PWM7 相异或后的信号）
TI5F：　经过 IC5F 数字滤波后的 TI5 信号
TI5FP：　经过 CC5P/CC6P 边沿检测器后的 TI5F 信号
TI5F_ED：TI5F 的边沿信号
TI5FP5：　经过 CC5P 边沿检测器后的 TI5F 信号
TI5FP6：　经过 CC6P 边沿检测器后的 TI5F 信号
IC5：　通过 CC5S 选择的通道 5 的捕获输入信号
OC5REF：输出通道 5 输出的参考波形（中间波形）
OC5：　通道 5 的主输出信号（经过 CC5P 极性处理后的 OC5REF 信号）
CC5：　通道 5 的捕获/比较寄存器

TI6：　外部时钟输入信号 6（PWM6 管脚信号）
TI6F：　经过 IC6F 数字滤波后的 TI6 信号
TI6F_ED：TI6F 的边沿信号
TI6FP：　经过 CC5P/CC6P 边沿检测器后的 TI6F 信号
TI6FP5：　经过 CC5P 边沿检测器后的 TI6F 信号
TI6FP6：　经过 CC6P 边沿检测器后的 TI6F 信号
IC6：　通过 CC6S 选择的通道 6 的捕获输入信号
OC6REF：输出通道 6 输出的参考波形（中间波形）

OC6：　通道 6 的主输出信号（经过 CC6P 极性处理后的 OC6REF 信号）
CC6：　通道 6 的捕获/比较寄存器

TI7：　外部时钟输入信号 7（PWM7 管脚信号）
TI7F：　经过 IC7F 数字滤波后的 TI7 信号
TI7F_ED：TI7F 的边沿信号
TI7FP：　经过 CC7P/CC8P 边沿检测器后的 TI7F 信号
TI7FP7：经过 CC7P 边沿检测器后的 TI7F 信号
TI7FP8：经过 CC8P 边沿检测器后的 TI7F 信号
IC7：　通过 CC7S 选择的通道 7 的捕获输入信号
OC7REF：输出通道 7 输出的参考波形（中间波形）
OC7：　通道 7 的主输出信号（经过 CC7P 极性处理后的 OC7REF 信号）
CC7：　通道 7 的捕获/比较寄存器

TI8：　外部时钟输入信号 8（PWM8 管脚信号）
TI8F：　经过 IC8F 数字滤波后的 TI8 信号
TI8F_ED：TI8F 的边沿信号
TI8FP：　经过 CC7P/CC8P 边沿检测器后的 TI8F 信号
TI8FP7：经过 CC7P 边沿检测器后的 TI8F 信号
TI8FP8：经过 CC8P 边沿检测器后的 TI8F 信号
IC8：　通过 CC8S 选择的通道 8 的捕获输入信号
OC8REF：输出通道 8 输出的参考波形（中间波形）
OC8：　通道 8 的主输出信号（经过 CC8P 极性处理后的 OC8REF 信号）
CC8：　通道 8 的捕获/比较寄存器

28.3 简介

PWMB/PWMD/PWMF 与 PWMA/PWMC/PWME 唯一的区别是 PWMA/PWMC/PWME 可输出带死区的互补对称 PWM，而 PWMB/PWMD/PWMF 则只能输出单端的 PWM，其他功能完全相同。下面关于高级 PWM 的介绍只以 PWMA 为例进行说明。

PWMA 由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。

PWMA 适用于许多不同的用途：

- 基本的定时
- 测量输入信号的脉冲宽度（输入捕获）
- 产生输出波形（输出比较，PWM 和单脉冲模式）
- 对应与不同事件（捕获，比较，溢出，刹车，触发）的中断
- 与 PWMB 或者外部信号（外部时钟，复位信号，触发和使能信号）同步

PWMA 广泛的适用于各种控制应用中，包括那些需要中间对齐模式 PWM 的应用，该模式支持互补输出和死区时间控制。

PWMA 的时钟源可以是内部时钟，也可以是外部的信号，可以通过配置寄存器来进行选择。

28.4 主要特性

PWMA 的特性包括：

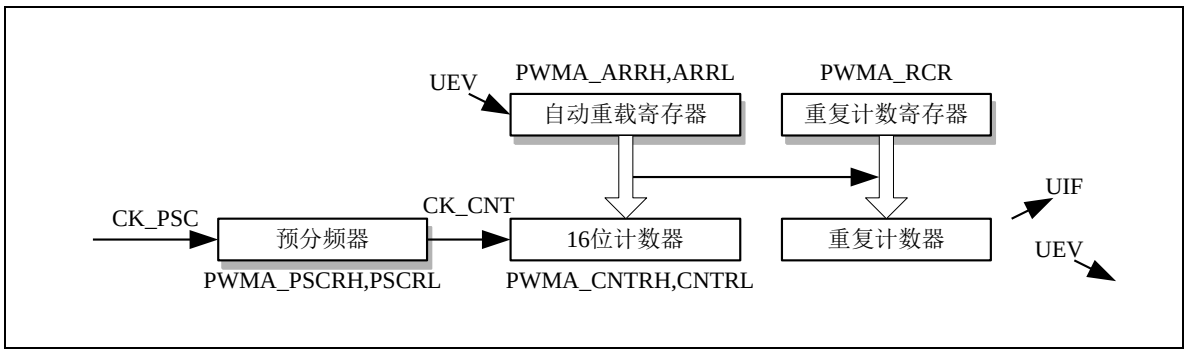
- 16 位向上、向下、向上/下自动装载计数器
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 16 位可编程（可以实时修改）预分频器，计数器时钟频率的分频系数为 1~65535 之间的任意数值
- 同步电路，用于使用外部信号控制定时器以及定时器互联
- 多达 4 个独立通道可以配置成：
  - 输入捕获
  - 输出比较
  - PWM 输出（边缘或中间对齐模式）
  - 六步 PWM 输出
  - 单脉冲模式输出
  - 支持 4 个死区时间可编程的通道上互补输出
- 刹车输入信号（PWMFLT）可以将定时器输出信号置于复位状态或者一个确定状态
- 外部触发输入引脚（PWMETI/PWMETR）
- 产生中断的事件包括：
  - 更新：计数器向上溢出/向下溢出，计数器初始化（通过软件或者内部/外部触发）
  - 触发事件（计数器启动、停止、初始化或者由内部/外部触发计数）
  - 输入捕获
  - 输出比较
  - 刹车信号输入

28.5 时基单元

PWMA 的时基单元包含：

- 16 位向上/向下计数器
- 16 位自动重载寄存器
- 重复计数器
- 预分频器

PWMA 时基单元



16 位计数器、预分频器、自动重载寄存器和重复计数器寄存器都可以通过软件进行读写操作。

自动重载寄存器由预装载寄存器和影子寄存器组成。

可在在两种模式下写自动重载寄存器：

- 自动预装载已使能（PWMA_CR1 寄存器的 ARPE 位为 1）。

在此模式下，写入自动重载寄存器的数据将被保存在预装载寄存器中，并在下一个更新事件（UEV）时传送到影子寄存器。

- 自动预装载已禁止（PWMA_CR1 寄存器的 ARPE 位为 0）。

在此模式下，写入自动重载寄存器的数据将立即写入影子寄存器。

更新事件的产生条件：

- 计数器向上或向下溢出。
- 软件置位了 PWMA_EGR 寄存器的 UG 位。
- 时钟/触发控制器产生了触发事件。

在预装载使能时（ARPE=1），如果发生了更新事件，预装载寄存器中的数值（PWMA_ARR）将写入影子寄存器中，并且 PWMA_PSCR 寄存器中的值将写入预分频器中。

置位 PWMA_CR1 寄存器的 UDIS 位将禁止更新事件（UEV）。

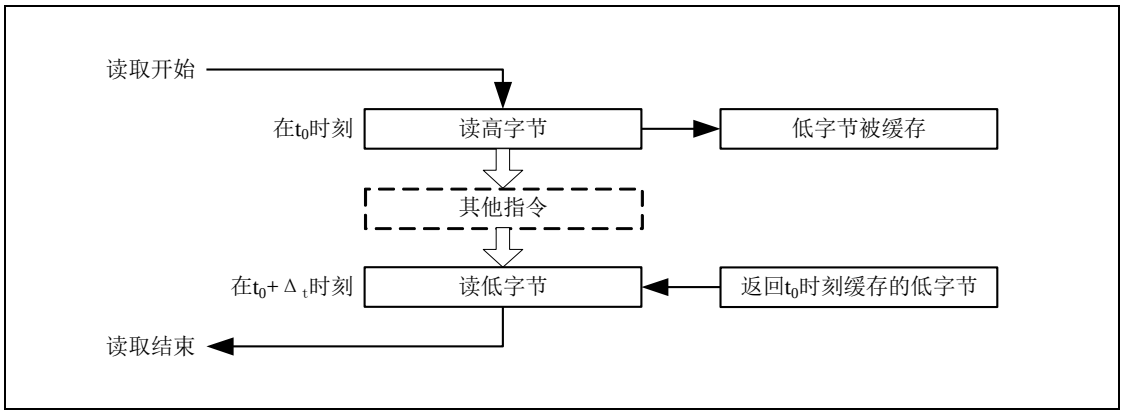
预分频器的输出 CK_CNT 驱动计数器，而 CK_CNT 仅在 IM1_CR1 寄存器的计数器使能位（CEN）被置位时才有效。

注意：实际的计数器在 CEN 位使能的一个时钟周期后才开始计数。

28.5.1 读写 16 位计数器

写计数器的操作没有缓存，在任何时候都可以写 PWMA_CNTRH 和 PWMA_CNTRL 寄存器，因此为避免写入了错误的数值，一般建议不要在计数器运行时写入新的数值。

读计数器的操作带有 8 位的缓存。用户必须先读定时器的高字节，在用户读了高字节后，低字节将被自动缓存，缓存的数据将会一直保持直到 16 位数据的读操作完成。



28.5.2 16 位 PWMA_ARR 寄存器的写操作

预装载寄存器中的值将写入 16 位的 PWMA_ARR 寄存器中，此操作由两条指令完成，每条指令写 入 1 个字节。必须先写高字节，后写低字节。

影子寄存器在写入高字节时被锁定，并保持到低字节写完。

28.5.3 预分频器

预分频器的实现：

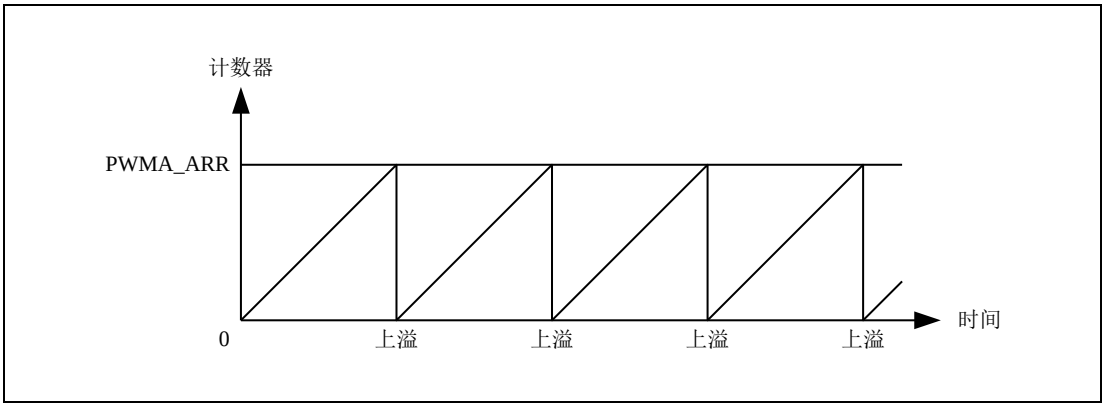
PWMA 的预分频器基于一个由 16 位寄存器（PWMA_PSCR）控制的 16 位计数器。由于这个控制寄存器带有缓冲器，因此它能够在运行时被改变。预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。预分频器的值由预装载寄存器写入，保存了当前使用值的影子寄存器在低字节写入时被载入。由于需两次单独的写操作来写 16 位寄存器，因此必须保证高字节先写入。新的预分频器的值在下一次更新事件到来时被采用。对 PWMA_PSCR 寄存器的读操作通过预装载寄存器完成。

计数器的频率计算公式：$f_{CK_CNT} = f_{CK_PSC} / (PSCR[15:0] + 1)$

28.5.4 向上计数模式

在向上计数模式中，计数器从 0 计数到用户定义的比较值（PWMA_ARR 寄存器的值），然后重新从 0 开始计数并产生一个计数器溢出事件，此时如果 PWMA_CR1 寄存器的 UDIS 位是 0，将会产生一个更新事件（UEV）。

向上计数模式的计数器



通过软件方式或者通过使用触发控制器置位 PWMA_EGR 寄存器的 UG 位同样也可以产生一个更新事件。

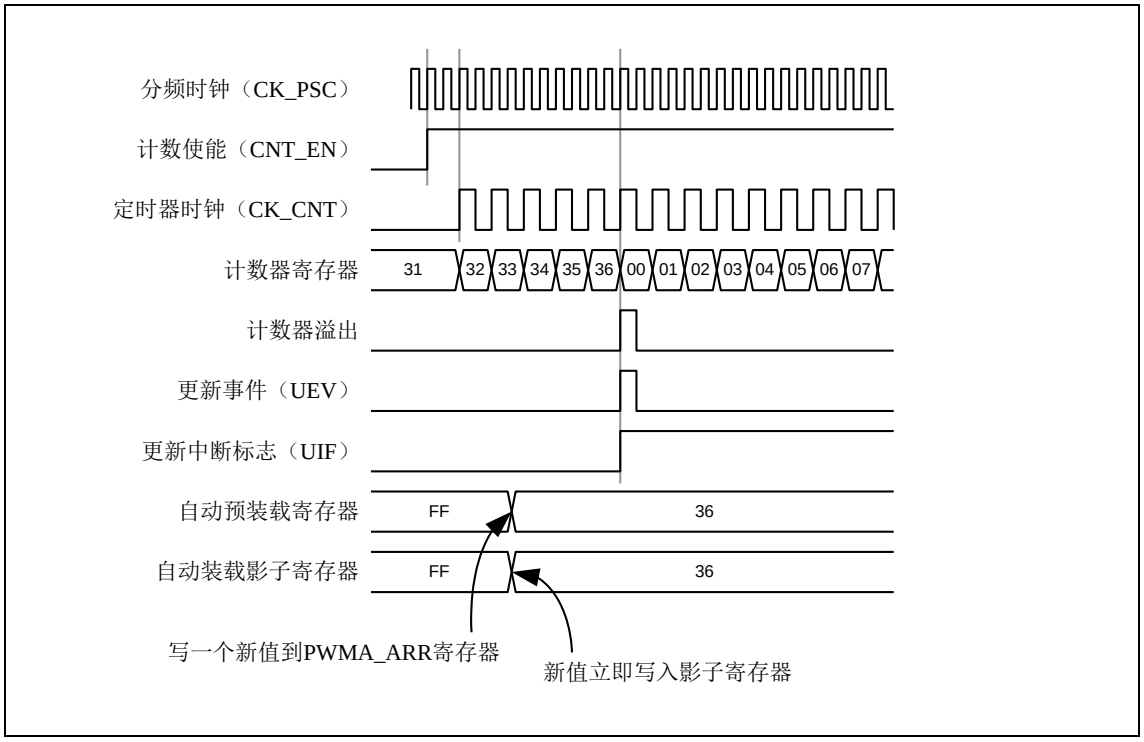
使用软件置位 PWMA_CR1 寄存器的 UDIS 位，可以禁止更新事件，这样可以避免在更新预装载寄存器时更新影子寄存器。在 UDIS 位被清除之前，将不产生更新事件。但是在应该产生更新事件时，计数器仍会被清 0，同时预分频器的计数也被清 0（但预分频器的数值不变）。此外，如果设置了 PWMA_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志（即不产生中断请求）。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件依据 URS 位同时设置更新标志位（PWMA_SR 寄存器的 UIF 位）：

- 自动装载影子寄存器被重新置入预装载寄存器的值（PWMA_ARR）。
- 预分频器的缓存器被置入预装载寄存器的值（PWMA_PSC 寄存器的内容）。

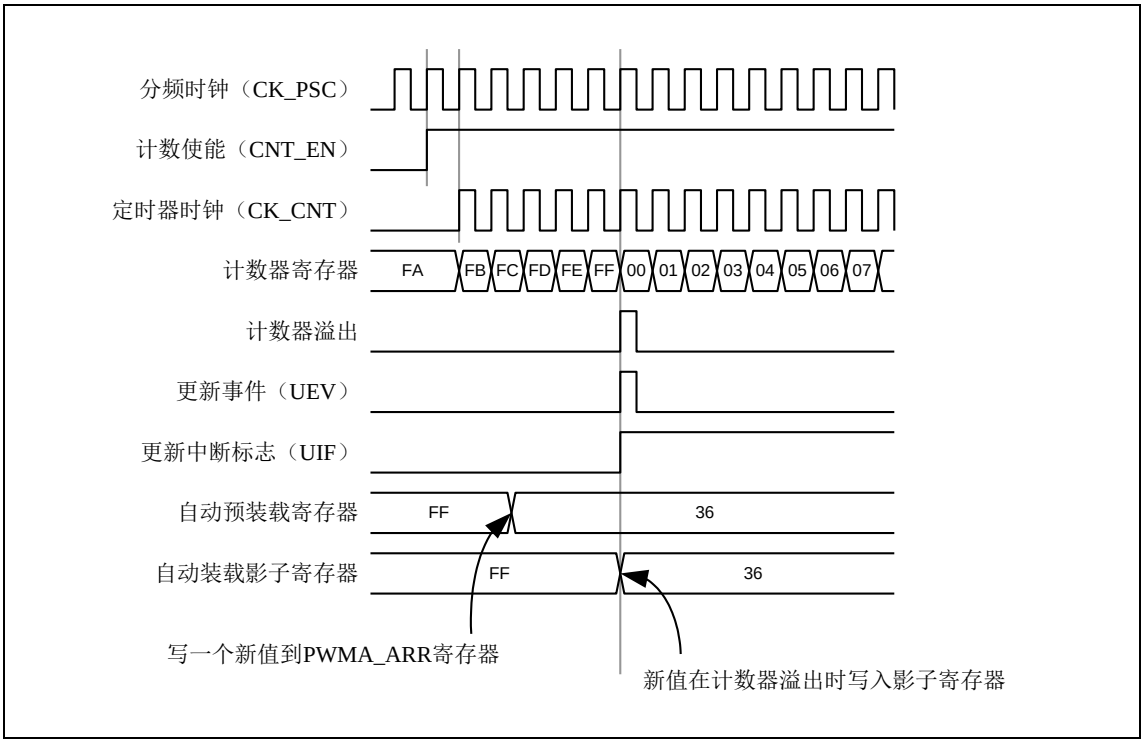
下图给出一些例子，说明当 PWMA_ARR=0x36 时，计数器在不同时钟频率下的动作。图中预分频为 2，因此计数器的时钟（CK_CNT）频率是预分频时钟（CK_PSC）频率的一半。图中禁止了自动装载功能（ARPE=0），所以在计数器达到 0x36 时，计数器溢出，影子寄存器立刻被更新，同时产生一个更新事件。

当 ARPE=0（ARR 不预装载），预分频为 2 时的计数器更新：



下图的预分频为 1，因此 CK_CNT 的频率与 CK_PSC 一致。图中使能了自动重载（ARPE=1），所以在计数器达到 0xFF 产生溢出。0x36 将在溢出时被写入，同时产生一个更新事件。

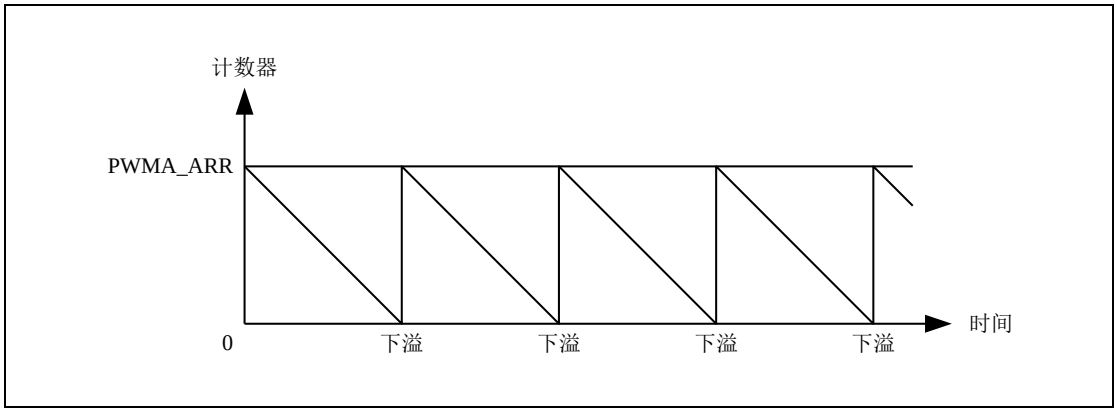
ARPE=1(PWMA_ARR 预装载) 预分频为 1 时的计数器更新：



28.5.5 向下计数模式

在向下模式中，计数器从自动装载的值（PWMA_ARR 寄存器的值）开始向下计数到 0，然后再从自动装载的值重新开始计数，并产生一个计数器向下溢出事件。如果 PWMA_CR1 寄存器的 UDIS 位被清除，还会产生一个更新事件（UEV）。

向下计数模式的计数器



通过软件方式或者通过使用触发控制器置位 PWMA_EGR 寄存器的 UG 位同样也可以产生一个更新事件。

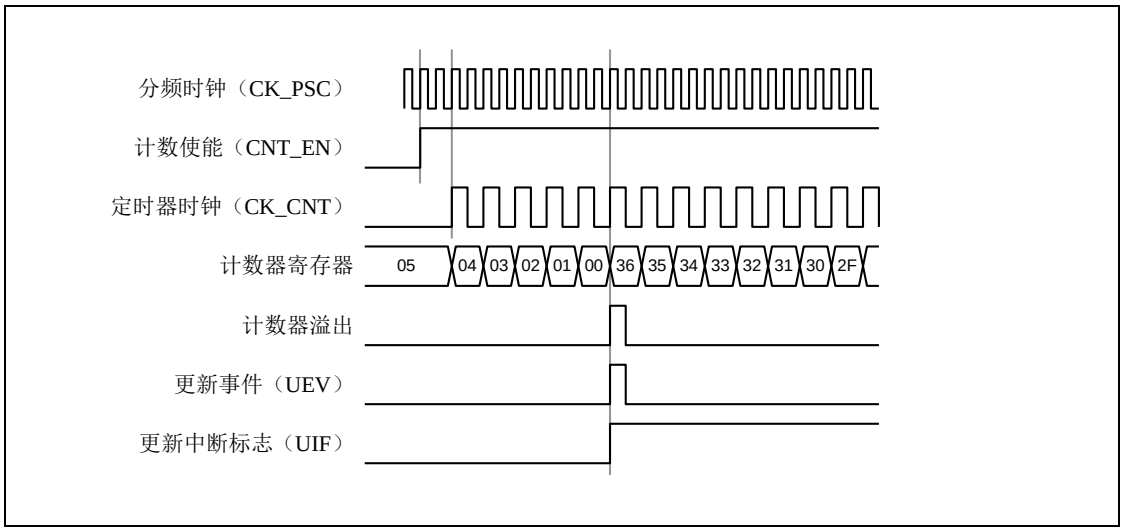
置位 PWMA_CR1 寄存器的 UDIS 位可以禁止 UEV 事件。这样可以避免在更新预装载寄存器时更新影子寄存器。因此 UDIS 位清除之前不会产生更新事件。然而，计数器仍会从当前自动加载值重新开始计数，并且预分频器的计数器重新从 0 开始（但预分频器不能被修改）。此外，如果设置了 PWMA_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，硬件依据 URS 位同时设置更新标志位（PWMA_SR 寄存器的 UIF 位）：

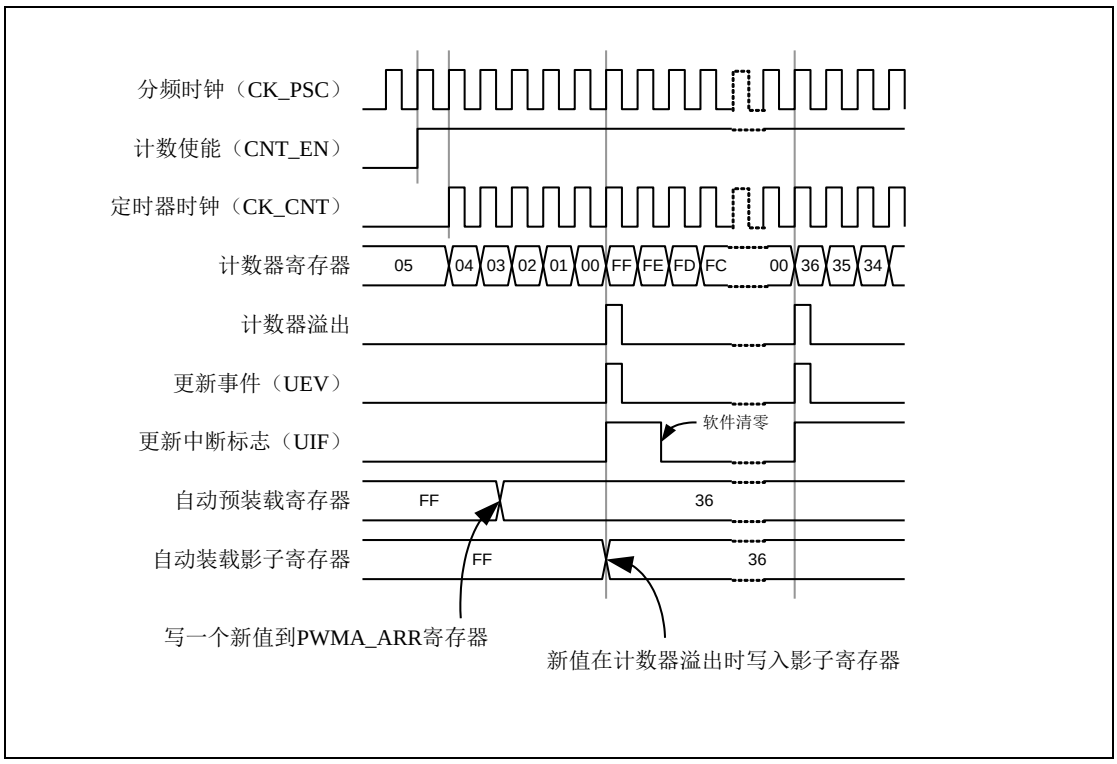
- 自动装载影子寄存器被重新置入预装载寄存器的值（PWMA_ARR）。
- 预分频器的缓存器被置入预装载寄存器的值（PWMA_PSC 寄存器的内容）。

以下是一些当 PWMA_ARR=0x36 时，计数器在不同时钟频率下的图表。下图描述了在向下计数模式下，预装载不使能时新的数值在下个周期时被写入。

ARPE=0（ARR 不预装载），预分频为 2 时的计数器更新：



ARPE=1（ARR 预装载），预分频为 1 时的计数器更新

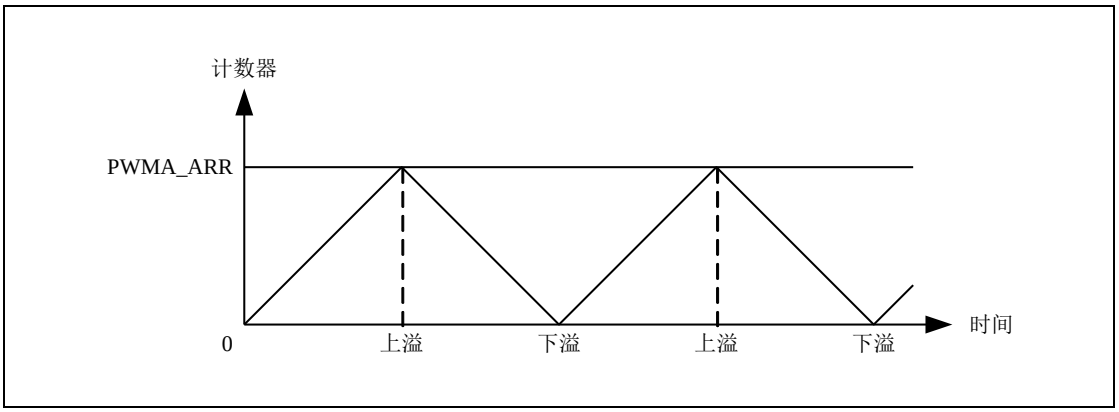


28.5.6 中间对齐模式（向上/向下计数）

在中央对齐模式，计数器从 0 开始计数到 PWMA_ARR 寄存器的值-1，产生一个计数器上溢事件，然后从 PWMA_ARR 寄存器的值向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。

在此模式下，不能写入 PWMA_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

中央对齐模式的计数器



如果定时器带有重复计数器，在重复了指定次数（PWMA_RCR[15:0]的值）的向上和向下溢出之后会产生更新事件（UEV）。否则每一次的向上向下溢出都会产生更新事件。

通过软件方式或者通过使用触发控制器置位 PWMA_EGR 寄存器的 UG 位同样也可以产生一个更新事件。此时，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。

设置 PWMA_CR1 寄存器中的 UDIS 位可以禁止 UEV 事件。这样可以避免在更新预装载寄存器时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。如果定时器带有重复计数器，由于重复寄存器没有双重的缓冲，新的重复数值将立刻生效，因此在修改时需要小心。此外，如果设置了 PWMA_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

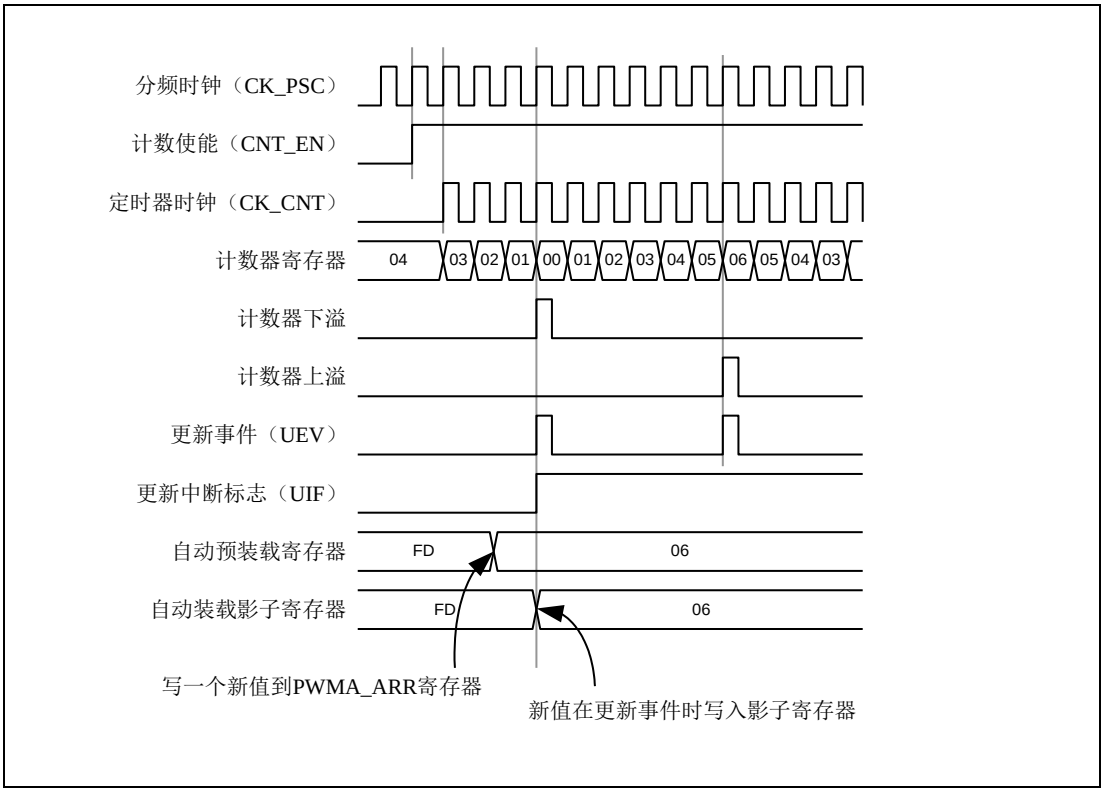
当发生更新事件时，所有的寄存器都被更新，硬件依据 URS 位更新标志位（PWMA_SR 寄存器中的 UIF 位）：

- 预分频器的缓存器被加载为预装载（PWMA_PSC 寄存器）的值。
- 当前的自动加载寄存器被更新为预装载值（PWMA_ARR 寄存器中的内容）。

要注意到如果因为计数器溢出而产生更新，自动重装载寄存器将在计数器重载入之前被更新，因此下一个周期才是预期的值（计数器被装载为新的值）。

以下是一些计数器在不同时钟频率下的操作的例子：

内部时钟分频因子为 1，PWMA_ARR=0x6，ARPE＝1



使用中央对齐模式的提示：

- 启动中央对齐模式时，计数器将按照原有的向上/向下的配置计数。也就是说 PWMA_CR1 寄存器中的 DIR 位将决定计数器是向上还是向下计数。此外，软件不能同时修改 DIR 位和 CMS 位的值。
- 不推荐在中央对齐模式下，计数器正在计数时写计数器的值，这将导致不能预料的后果。具体的说：
  - 向计数器写入了比自动装载值更大的数值时（PWMA_CNT>PWMA_ARR），但计数器的计数方向不发生改变。例如计数器已经向上溢出，但计数器仍然向上计数。
  - 向计数器写入了 0 或者 PWMA_ARR 的值，但更新事件不发生。
- 安全使用中央对齐模式的计数器的方法是在启动计数器之前先用软件（置位 PWMA_EGR 寄存器的 UG 位）产生一个更新事件，并且不在计数器计数时修改计数器的值。

28.5.7 重复计数器

时基单元解释了计数器向上/向下溢出时更新事件（UEV）是如何产生的，然而事实上它只能在重复计数器的值达到 0 的时候产生。这个特性对产生 PWM 信号非常有用。

这意味着在每 N 次计数上溢或下溢时，数据从预装载寄存器传输到影子寄存器（PWMA_ARR 自动重载入寄存器，PWMA_PSC 预装载寄存器，还有在比较模式下的捕获/比较寄存器 PWMA_CCRx），N 是 PWMA_RCR[15:0]重复计数寄存器中的值。

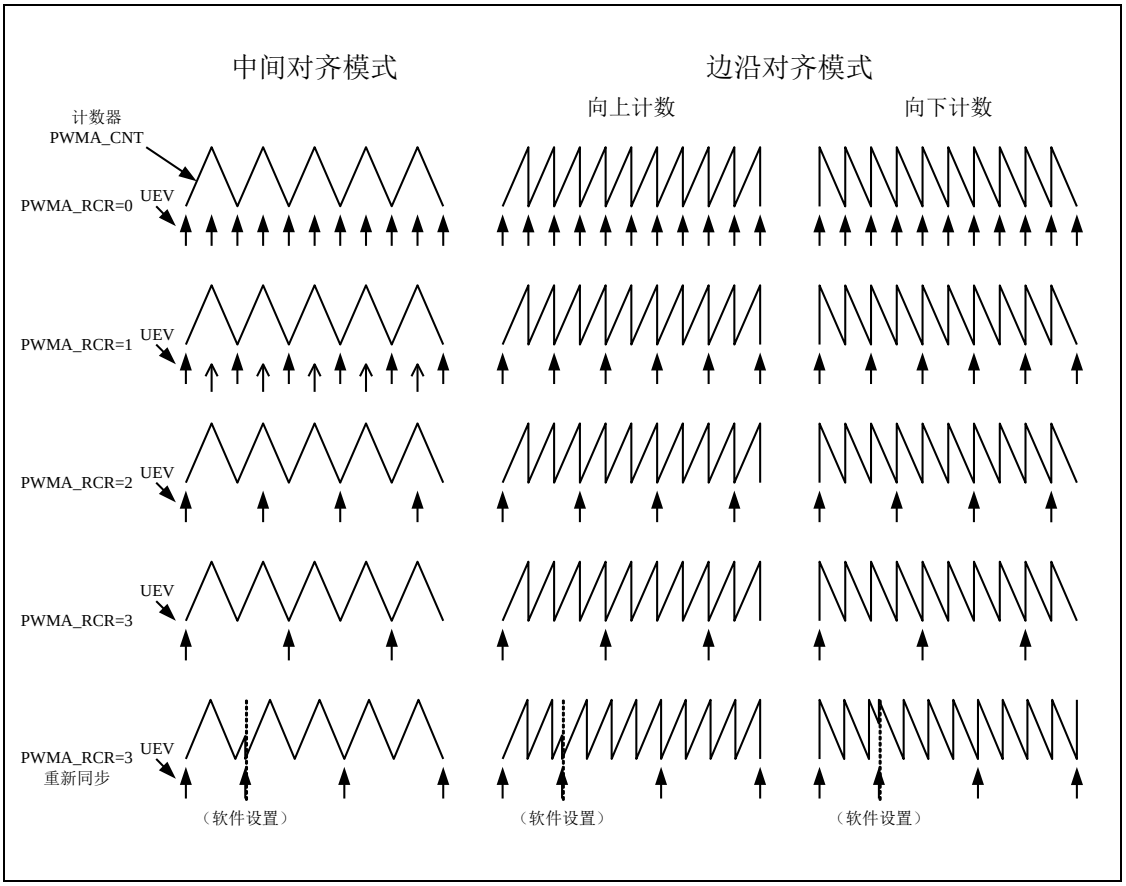
重复计数器在下述任一条件成立时递减：

- 向上计数模式下每次计数器向上溢出时
- 向下计数模式下每次计数器向下溢出时
- 中央对齐模式下每次上溢和每次下溢时。

虽然这样限制了 PWM 的最大循环周期为 128，但它能够在每个 PWM 周期 2 次更新占空比。在中央对齐模式下，因为波形是对称的，如果每个 PWM 周期中仅刷新一次比较寄存器，则最大的分辨率为 2*tCK_PSC。

重复计数器是自动加载的，重复速率由 PWMA_RCR[15:0]寄存器的值定义。当更新事件由软件产生（通过设置 PWMA_EGR 中的 UG 位）或者通过硬件的时钟/触发控制器产生，则无论重复计数器的值是多少，立即发生更新事件，并且 PWMA_RCR[15:0]寄存器中的内容被重载入到重复计数器。

不同模式下更新速率的例子，及 PWMA_RCR[15:0]的寄存器设置



28.6 时钟/触发控制器

时钟/触发控制器允许用户选择计数器的时钟源，输入触发信号和输出信号，

28.6.1 预分频时钟（CK_PSC）

时基单元的预分频时钟（CK_PSC）可以由以下源提供：

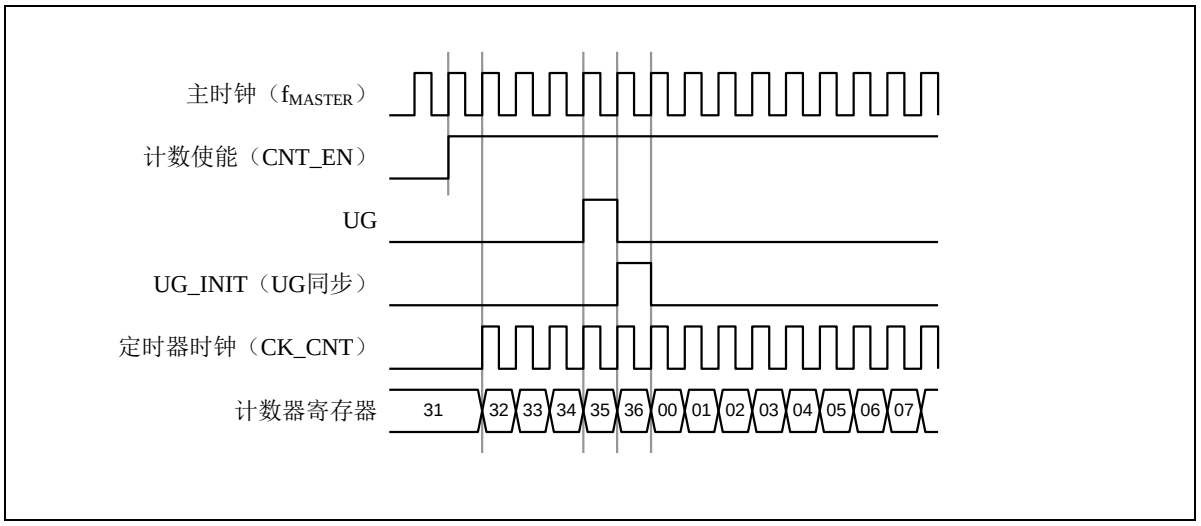
- 内部时钟（fMASTER）
- 外部时钟模式 1：外部时钟输入（TIx）
- 外部时钟模式 2：外部触发输入 ETR
- 内部触发输入（ITRx）：使用一个定时器做为另一个定时器的预分频时钟。

28.6.2 内部时钟源（fMASTER）

如果同时禁止了时钟/触发模式控制器和外部触发输入（PWMA_SMCR 寄存器的 SMS=000，PWMA_ETR 寄存器的 ECE＝0），则 CEN、DIR 和 UG 位是实际上的控制位，并且只能被软件修改（UG 位仍被自动清除）。一旦 CEN 位被写成 1，预分频器的时钟就由内部时钟提供。

下图描述了控制电路和向上计数器在普通模式下，不带预分频器时的操作。

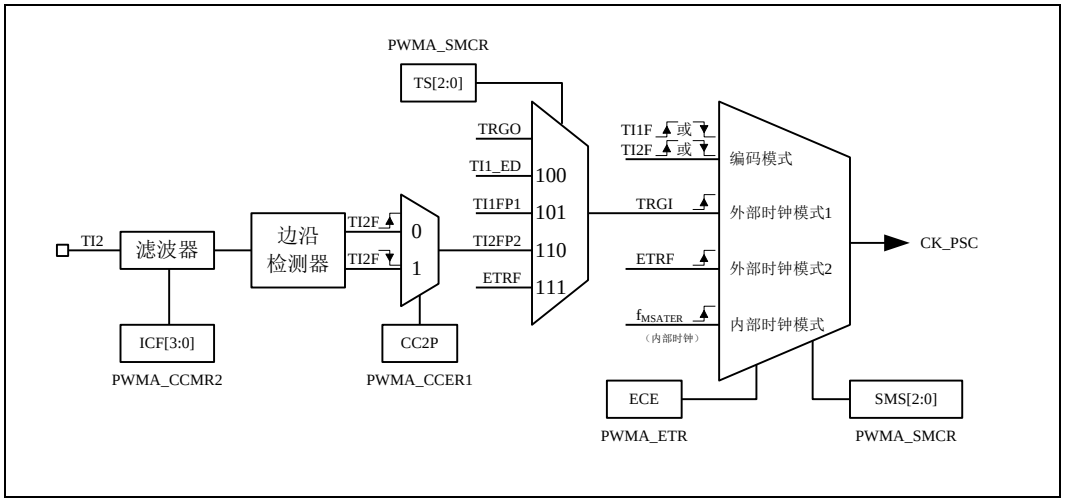
普通模式下的控制电路，f_{MASTER} 分频因子为 1



28.6.3 外部时钟源模式 1

当 PWMA_SMCR 寄存器的 SMS=111 时，此模式被选中。计数器可以在选定输入端的每个上升沿或下降沿计数。

TI2 外部时钟连接例子



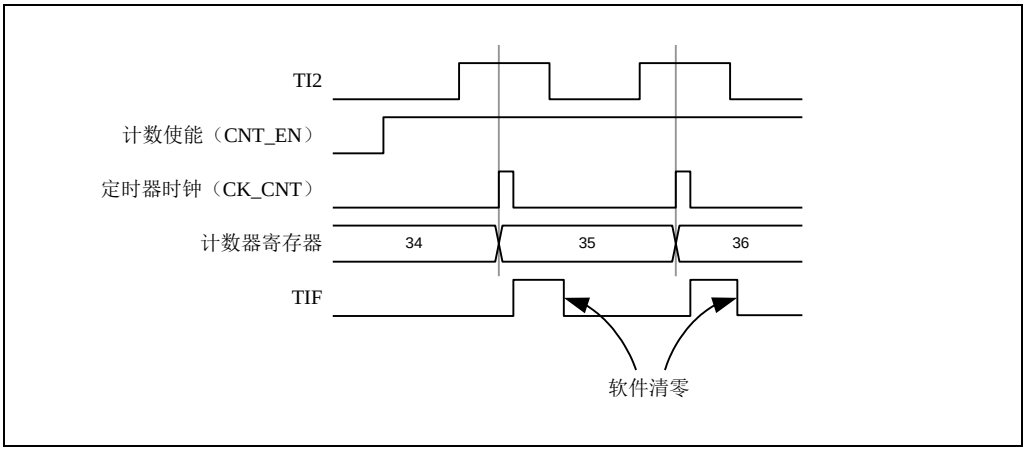
例如，要配置向上计数器在 TI2 输入端的上升沿计数，使用下列步骤：

1. 配置 PWMA_CCMR2 寄存器的 CC2S=01，使用通道 2 检测 TI2 输入
2. 配置 PWMA_CCMR2 寄存器的 IC2F[3:0]位，选择输入滤波器带宽（如果不需要滤波器，保持 IC2F=0000）注：捕获预分频器不用作触发，所以不需要对它进行配置，同样也不需要配置 TI2S 位，他们仅用来选择输入捕获源。
3. 配置 PWMA_CCER1 寄存器的 CC2P=0，选定上升沿极性
4. 配置 PWMA_SMCR 寄存器的 SMS=111，配置计数器使用外部时钟模式 1
5. 配置 PWMA_SMCR 寄存器的 TS=110，选定 TI2 作为输入源
6. 设置 PWMA_CR1 寄存器的 CEN=1，启动计数器

当上升沿出现在 TI2，计数器计数一次，且触发标识位（PWMA_SR1 寄存器的 TIF 位）被置 1，如果使能了中断（在 PWMA_IER 寄存器中配置）则会产生中断请求。

在 TI2 的上升沿和计数器实际时钟之间的延时取决于在 TI2 输入端的重新同步电路。

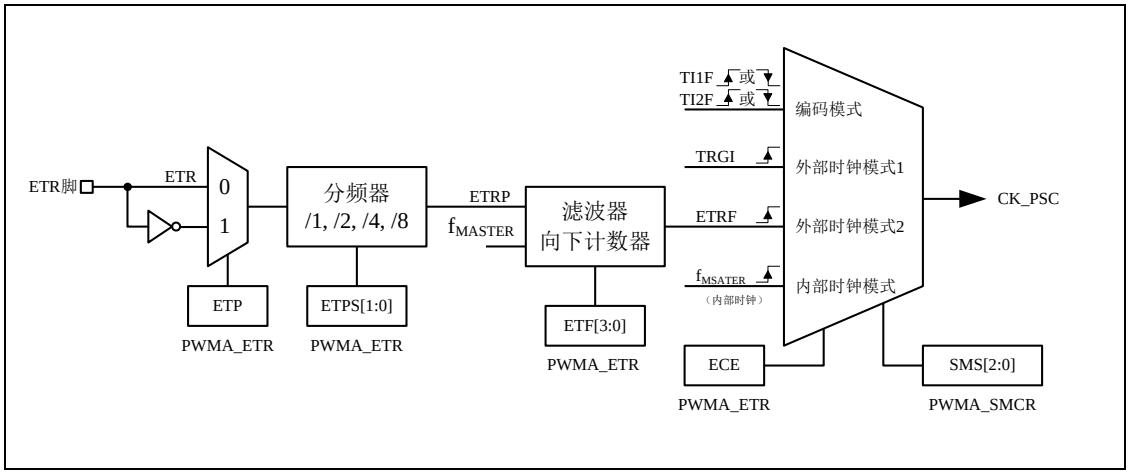
外部时钟模式 1 下的控制电路



28.6.4 外部时钟源模式 2

计数器能够在外部触发输入 ETR 信号的每一个上升沿或下降沿计数。将 PWMA_ETR 寄存器的 ECE 位写 1，即可选定此模式。

外部触发输入的总体框图：

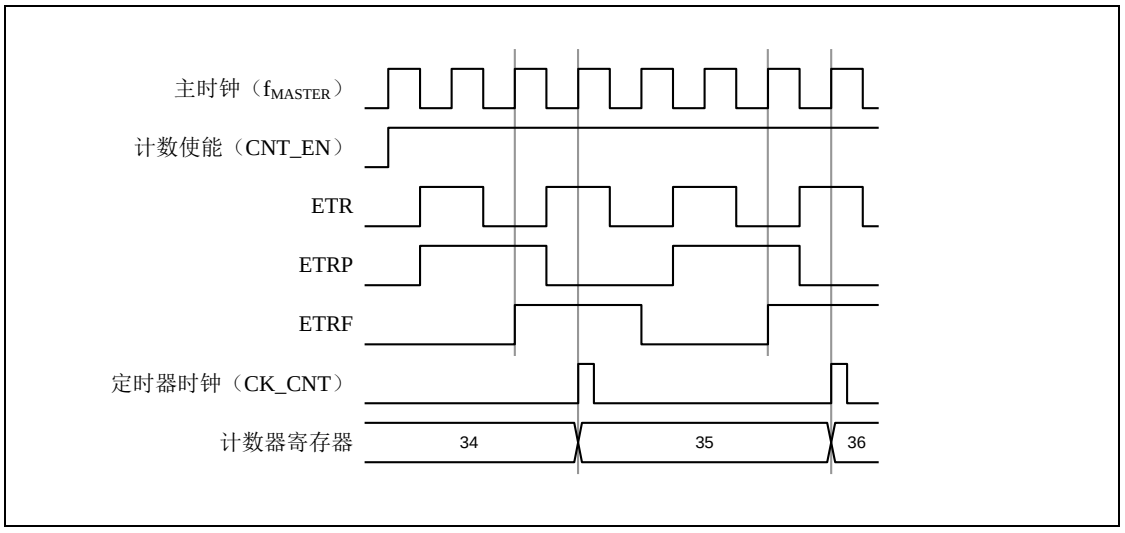


例如，要配置计数器在 ETR 信号的每 2 个上升沿时向上计数一次，需使用下列步骤：

1. 本例中不需要滤波器，配置 PWMA_ETR 寄存器的 ETF[3:0]=0000
2. 设置预分频器，配置 PWMA_ETR 寄存器的 ETPS[1:0]=01
3. 选择 ETR 的上升沿检测，配置 PWMA_ETR 寄存器的 ETP=0
4. 开启外部时钟模式 2，配置 PWMA_ETR 寄存器中的 ECE=1
5. 启动计数器，写 PWMA_CR1 寄存器的 CEN=1

计数器在每 2 个 ETR 上升沿计数一次。

外部时钟模式 2 下的控制电路



28.6.5 触发同步

PWMA 的计数器使用三种模式与外部的触发信号同步：

- 标准触发模式
- 复位触发模式
- 门控触发模式

标准触发模式

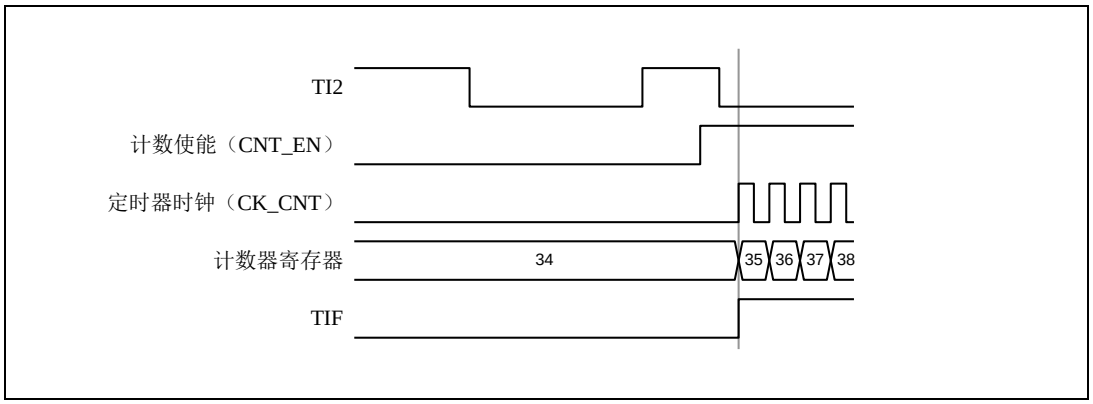
计数器的使能（CEN）依赖于选中的输入端上的事件。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

1. 配置 PWMA_CCER1 寄存器的 CC2P＝0，选择 TI2 的上升沿做为触发条件。
2. 配置 PWMA_SMCR 寄存器的 SMS=110，选择计数器为触发模式。配置 PWMA_SMCR 寄存器的 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时置位 TIF 标志。TI2 上升沿和计数器启动计数之间的延时取决于 TI2 输入端的重同步电路。

标准触发模式的控制电路



复位触发模式

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化。同时，如果 PWMA_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV，然后所有的预装载寄存器（PWMA_ARR，PWMA_CCRx）都会被更新。

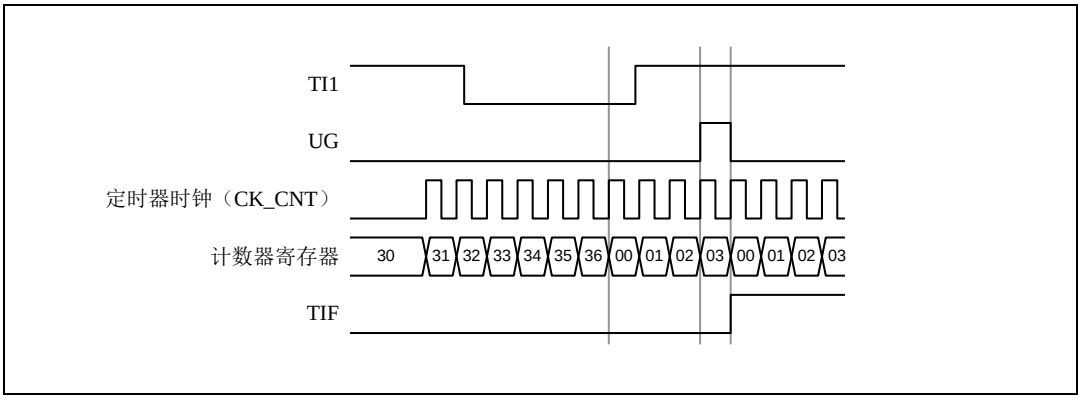
在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

1. 配置 PWMA_CCER1 寄存器的 CC1P=0 来选择 TI1 的极性（只检测 TI1 的上升沿）。
2. 配置 PWMA_SMCR 寄存器的 SMS=100，选择定时器为复位触发模式。配置 PWMA_SMCR 寄存器的 TS=101，选择 TI1 作为输入源。
3. 配置 PWMA_CR1 寄存器的 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常计数直到 TI1 出现一个上升沿。此时，计数器被清零然后从 0 重新开始计数。同时，触发标志（PWMA_SR1 寄存器的 TIF 位）被置位，如果使能了中断（PWMA_IER 寄存器的 TIE 位），则产生一个中断请求。

下图显示当自动重装载寄存器 PWMA_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

复位触发模式下的控制电路



门控触发模式

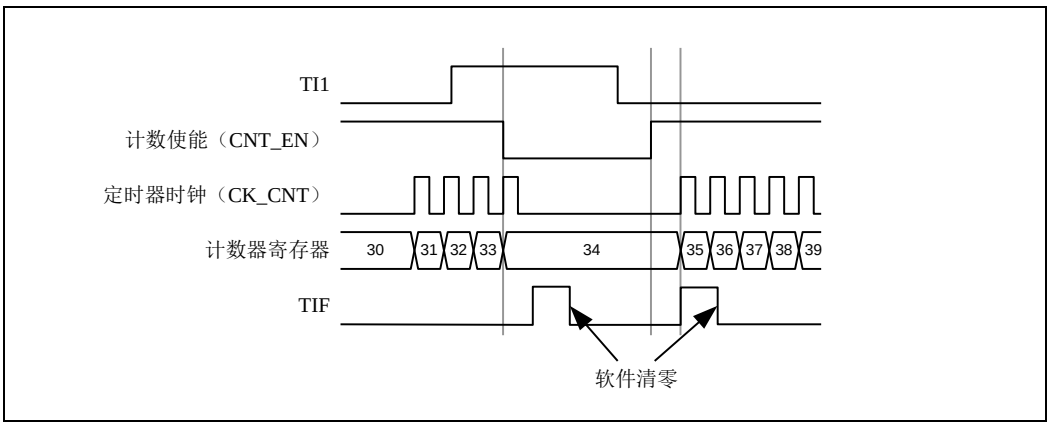
计数器由选中的输入端信号的电平使能。

在如下的例子中，计数器只在 TI1 为低时向上计数：

1. 配置 PWMA_CCER1 寄存器的 CC1P=1 来确定 TI1 的极性（只检测 TI1 上的低电平）。
2. 配置 PWMA_SMCR 寄存器的 SMS=101，选择定时器为门控触发模式，配置 PWMA_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
3. 配置 PWMA_CR1 寄存器的 CEN=1，启动计数器（在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何）。

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 TI1 变高则停止计数。当计数器开始或停止时 TIF 标志位都会被置位。TI1 上升沿和计数器实际停止之间的延时取决于 TI1 输入端的重同步电路。

门控触发模式下的控制电路



外部时钟模式 2 联合触发模式

外部时钟模式 2 可以与另一个输入信号的触发模式一起使用。例如，ETR 信号被用作外部时钟的输入，另一个输入信号可用作触发输入（支持标准触发模式，复位触发模式和门控触发模式）。注意不能通过 PWMA_SMCR 寄存器的 TS 位把 ETR 配置成 TRGI。

在下面的例子中，一旦在 TI1 上出现一个上升沿，计数器即在 ETR 的每一个上升沿向上计数一次：

1. 通过 PWMA_ETR 寄存器配置外部触发输入电路。配置 ETPS=00 禁止预分频，配置 ETP=0 监测 ETR 信号的上升沿，配置 ECE=1 使能外部时钟模式 2。
2. 配置 PWMA_CCER1 寄存器的 CC1P=0 来选择 TI1 的上升沿触发。
3. 配置 PWMA_SMCR 寄存器的 SMS=110 来选择定时器为触发模式。配置 PWMA_SMCR 寄存器的

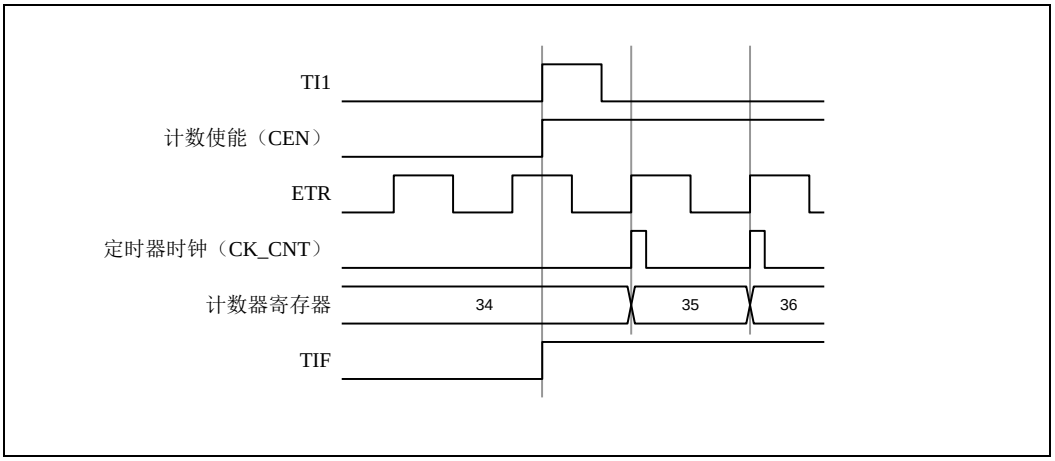
TS=101 来选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 ETR 的上升沿计数。

TI1 信号的上升沿和计数器实际时钟之间的延时取决于 TI1 输入端的重同步电路。

ETR 信号的上升沿和计数器实际时钟之间的延时取决于 ETRP 输入端的重同步电路。

外部时钟模式 2+触发模式下的控制电路



28.6.6 与 PWMB 同步

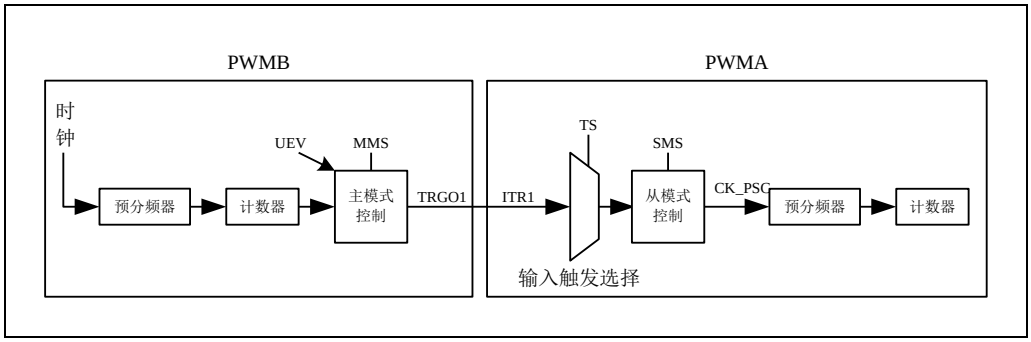
在芯片中，定时器在内部互相联结，用于定时器的同步或链接。当某个定时器配置成主模式时，可以输出触发信号（TRGO）到那些配置为从模式的定时器来完成复位操作、启动操作、停止操作或者作为那些定时器的驱动时钟。

使用 PWMB 的 TRGO 作为 PWMA 的预分频时钟

例如，用户可以配置 PWMB 作为 PWMA 的预分频时钟，需进行如下配置：

1. 配置 PWMB 为主模式，使得在每个更新事件（UEV）时输出周期性的触发信号。配置 PWMB_CR2 寄存器的 MMS=010，使每个更新事件时 TRGO 能输出一个上升沿。
2. PWMB 输出的 TRGO 信号链接到 PWMA。PWMA 需要配置成触发从模式，使用 ITR2 作为输入触发信号。以上操作可以通过配置 PWMA_SMCR 寄存器的 TS=010 实现。
3. 配置 PWMA_SMCR 寄存器的 SMS=111 将时钟/触发控制器设置为外部时钟模式 1。此操作将使 PWMB 输出的周期性触发信号 TRGO 的上升沿驱动 PWMA 的时钟。
4. 最后，置位 PWMB 的 CEN 位（PWMB_CR1 寄存器中），使能两个 PWM。

主/触发从模式的定时器例子



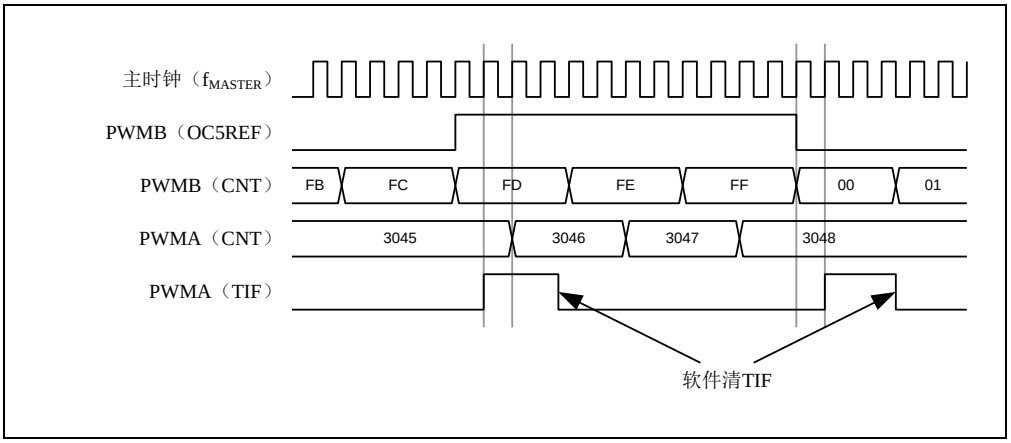
使用 PWMB 使能 PWMA

在本例中，我们用 PWMB 的比较输出使能 PWMA。PWMA 仅在 PWMB 的 OC1REF 信号为高时按照自己的驱动时钟计数。两个 PWM 都使用 4 分频的 f_{MASTER} 为时钟（$f_{CK_CNT} = f_{MASTER}/4$）。

1. 配置 PWMB 为主模式，将比较输出信号（OC5REF）作为触发信号输出。（配置 PWMB_CR2 寄存器的 MMS=100）。
2. 配置 PWMB 的 OC5REF 信号的波形（PWMB_CCMR1 寄存器）。
3. 配置 PWMA 把 PWMB 的输出作为自己的触发输入信号（配置 PWMA_SMCR 寄存器的 TS=010）。
4. 配置 PWMA 为门控触发模式（配置 PWMA_SMCR 寄存器的 SMS=101）。
5. 置位 CEN 位（PWMA_CR1 寄存器），使能 PWMA。
6. 置位 CEN 位（PWMB_CR1 寄存器），使能 PWMB。

注意：两个 PWM 的时钟并不同步，但仅影响 PWMA 的使能信号。

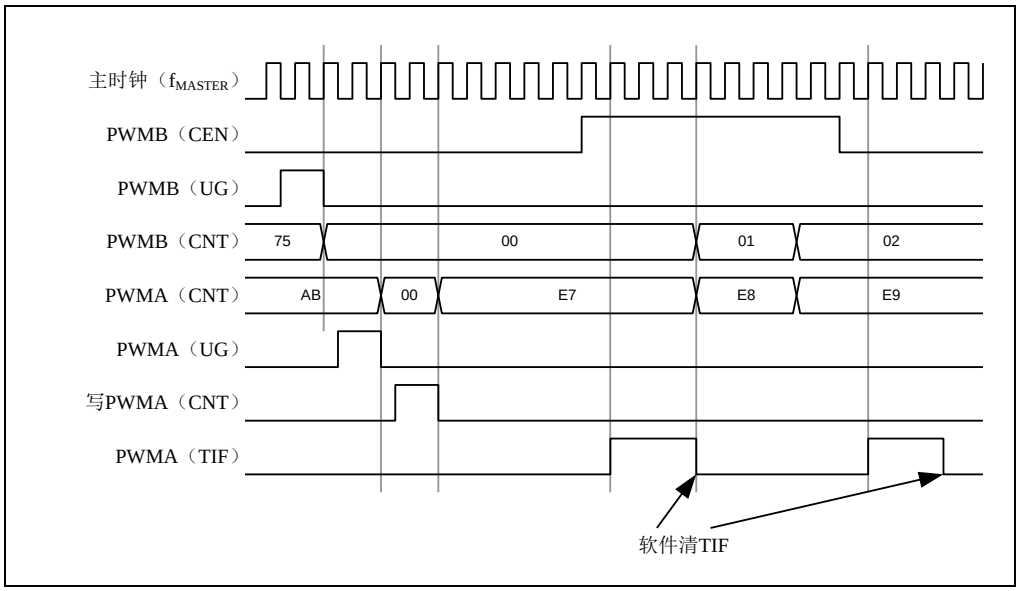
PWMB 的输出门控触发 PWMA



上图中，PWMA 的计数器和预分频器都没有在启动前初始化，所以都是从现有值开始计数的。如果在启动 PWMB 之前复位两个定时器，用户就可以写入期望的数值到 PWMA 的计数器，使之从指定值开始计数。对 PWMA 的复位操作可以通过软件写 PWMA_EGR 寄存器的 UG 位实现。

在下面这个例子中，我们使 PWMB 和 PWMA 同步。PWMB 为主模式并从 0 启动计数。PWMA 为触发从模式，并从 0xE7 启动计数。两个 PWM 采用相同的分频系数。当清除 PWMB_CR1 寄存器的 CEN 位时，PWMB 被禁止，同时 PWMA 停止计数。

1. 配置 PWMB 为主模式，将比较输出信号（OC5REF）作为触发信号输出。（配置 PWMB_CR2 寄存器的 MMS=100）。
2. 配置 PWMB 的 OC5REF 信号的波形（PWMB_CCMR1 寄存器）。
3. 配置 PWMA 把 PWMB 的输出作为自己的触发输入信号（配置 PWMA_SMCR 寄存器的 TS=010）。
4. 配置 PWMA 为门控触发模式（配置 PWMA_SMCR 寄存器的 SMS=101）。
5. 通过对 UG 位（PWMB_EGR 寄存器）写 1，复位 PWMB。
6. 通过对 UG 位（PWMA_EGR 寄存器）写 1，复位 PWMA。
7. 将 0xE7 写入 PWMA 的计数器中（PWMA_CNTRL），初始化 PWMA。
8. 通过对 CEN 位（PWMA_CR1 寄存器）写 1，使能 PWMA。
9. 通过对 CEN 位（PWMB_CR1 寄存器）写 1，启动 PWMB。
10. 通过对 CEN 位（PWMB_CR1 寄存器）写 0，停止 PWMB。



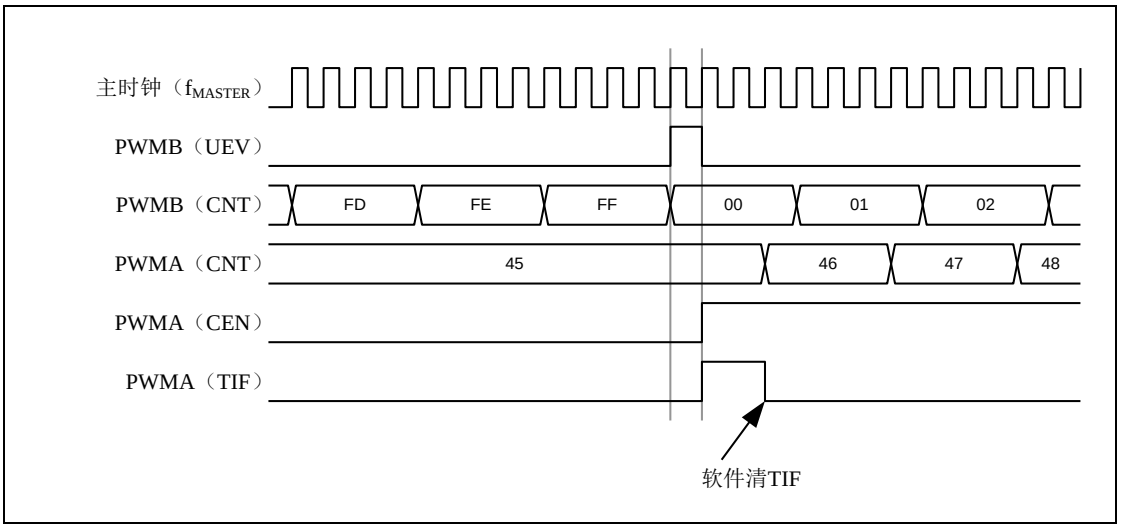
使用 PWMB 启动 PWMA

在本例中，我们用 PWMB 的更新事件来启动 PWMA。

PWMA 在 PWMB 发生更新事件时按照 PWMA 自己的驱动时钟从它的现有值开始计数（可以是非 0 值）。PWMA 在收到触发信号后自动使能 CEN 位，并开始计数，一直持续到用户向 PWMA_CR1 寄存器的 CEN 位写 0。两个 PWM 都使用 4 分频的 f_{MASTER} 作为驱动时钟（$f_{CK_CNT} = f_{MASTER}/4$）。

1. 配置 PWMB 为主模式，输出更新信号（UEV）。（配置 PWMB_CR2 寄存器的 MMS=010）。
2. 配置 PWMB 的周期（PWMB_ARR 寄存器）。
3. 配置 PWMA 用 PWMB 的输出作为输入的触发信号（配置 PWMA_SMCR 寄存器的 TS=010）。
4. 配置 PWMA 为触发模式（配置 PWMA_SMCR 寄存器的 SMS=110）。
5. 置位 CEN 位（PWMB_CR1 寄存器）启动 PWMB。

PWMB 的更新事件（PWMB-UEV）触发 PWMA



如同前面的例子，用户也可以在启动计数器前对它们初始化。

用外部信号同步的触发两个 PWM

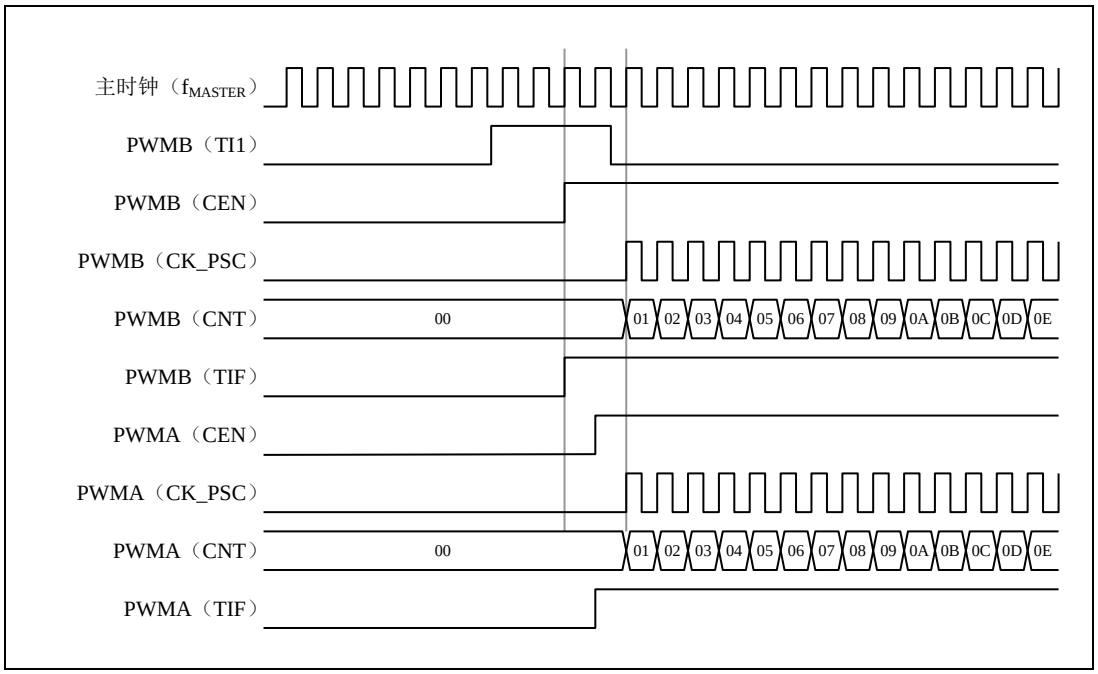
在本例中，使用 TI1 的上升沿使能 PWMB，并同时使能 PWMA。为了保持定时器的对齐，PWMB 需要配置成主/从模式（对于 TI1 信号为从模式，对于 PWMA 为主模式）。

1. 配置 PWMB 为主模式，以输出使能信号作为 PWMA 的触发（配置 PWMB_CR2 寄存器的 MMS=001）。
2. 配置 PWMB 为从模式，把 TI1 信号作为输入的触发信号（配置 PWMB_SMCR 寄存器的 TS=100）。
3. 配置 PWMB 的触发模式（配置 PWMB_SMCR 寄存器的 SMS=110）。
4. 配置 PWMB 为主/从模式（配置 PWMB_SMCR 寄存器的 MSM=1）。
5. 配置 PWMA 以 PWMB 的输出为输入触发信号（配置 PWMA_SMCR 寄存器的 TS=010）。
6. 配置 PWMA 的触发模式（配置 PWMA_SMCR 寄存器的 SMS=110）。

当 TI1 上出现上升沿时，两个定时器同步的开始计数，并且 TIF 位都被置起。

注意：在本例中，两个定时器在启动前都进行了初始化（设置 UG 位），所以它们都从 0 开始计数，但是用户也可以通过修改计数器寄存器（PWMA_CNT）来插入一个偏移量，这样的话，在 PWMB 的 CK_PSC 信号和 CNT_EN 信号间会插入延时。

PWMB 的 TI1 信号触发 PWMB 和 PWMA

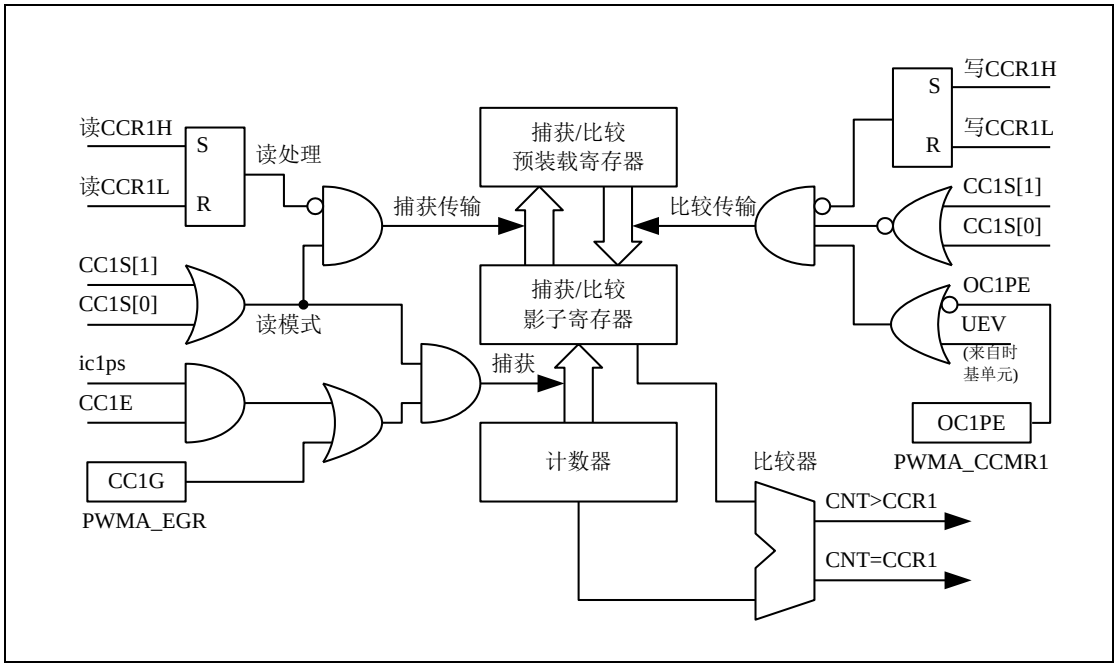


28.7 捕获/比较通道

PWM1P、PWM2P、PWM3P、PWM4P 可以用作输入捕获，PWM1P/PWM1N、PWM2P/PWM2N、PWM3P/PWM3N、PWM4P/PWM4N 可以输出比较，这个功能可以通过配置捕获/比较通道模式寄存器（PWMA_CCMRi）的 CCiS 通道选择位来实现，此处的 i 代表 1~4 的通道数。

每一个捕获/比较通道都是围绕着一个捕获/比较寄存器（包含影子寄存器）来构建的，包括捕获的输入部分（数字滤波、多路复用和预分频器）和输出部分（比较器和输出控制）。

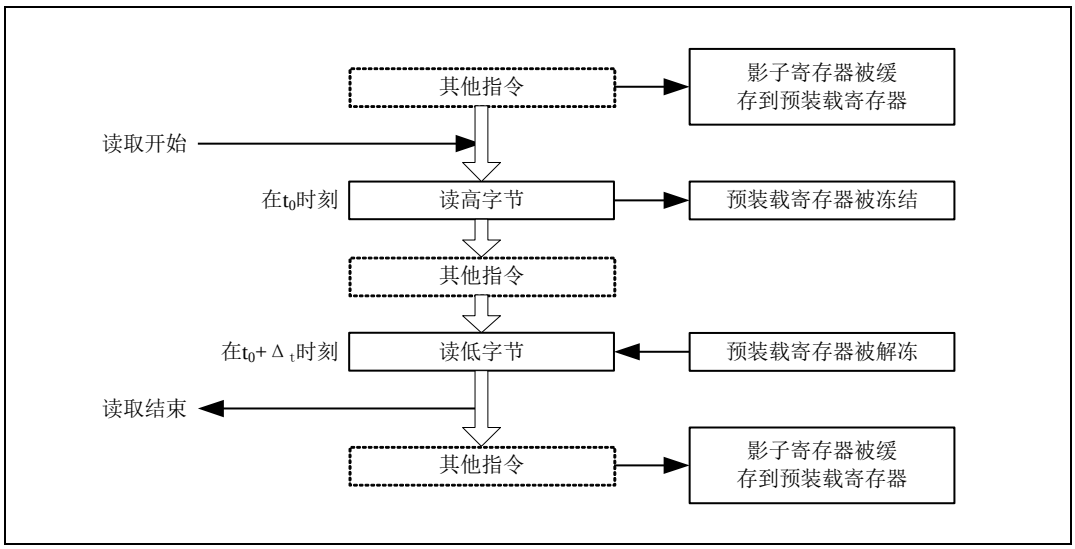
捕获/比较通道 1 的主要电路（其他通道与此类似）



捕获/比较模块由一个预装载寄存器和一个影子寄存器组成。读写过程仅操作预装载寄存器。在捕获模式下，捕获发生在影子寄存器上，然后再复制到预装载寄存器中。在比较模式下，预装载寄存器的内容被复制到影子寄存器中，然后影子寄存器的内容和计数器进行比较。

当通道被配置成输出模式时（PWMA_CCMRi 寄存器的 CCiS=0），可以随时访问 PWMA_CCRi 寄存器。

当通道被配置成输入模式时，对 PWMA_CCRi 寄存器的读操作类似于计数器的读操作。当捕获发生时，计数器的内容被捕获到 PWMA_CCRi 影子寄存器，随后再复制到预装载寄存器中。在读操作进行中，预装载寄存器是被冻结的。



上图描述了 16 位的 CCRi 寄存器的读操作流程，被缓存的数据将保持不变直到读流程结束。

在整个读流程结束后，如果仅仅读了 PWMA_CCRiL 寄存器，返回计数器数值的低位（LS）。

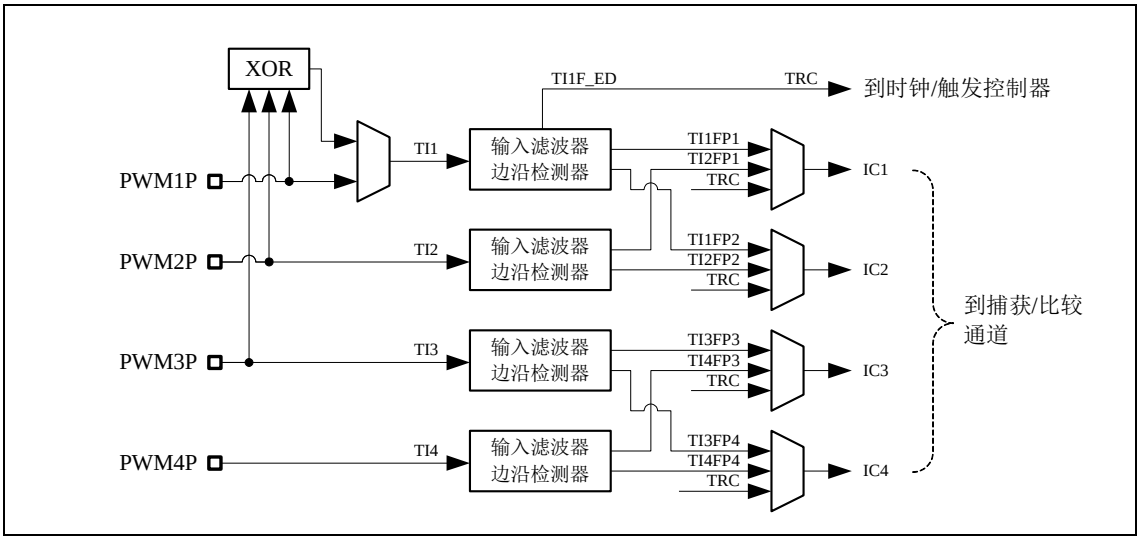
如果在读了低位（LS）数据以后再读高位（MS）数据，将不再返回同样的低位数据。

28.7.1 16 位 PWMA_CCRi 寄存器的写流程

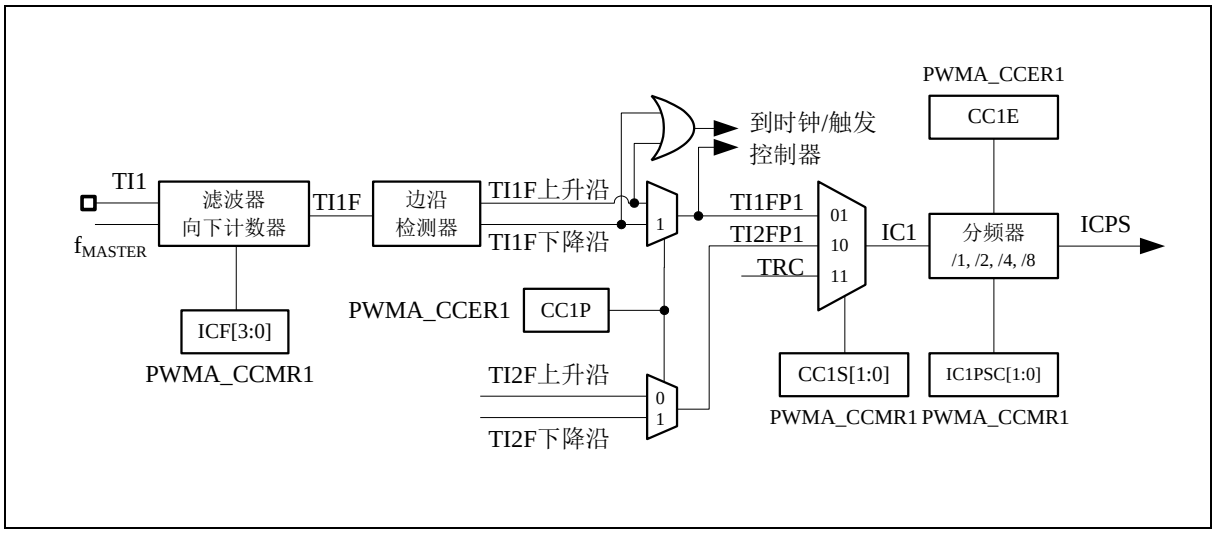
16 位 PWMA_CCRi 寄存器的写操作通过预装载寄存器完成。必需使用两条指令来完成整个流程，一条指令对应一个字节。必需先写高位字节（MS）。在写高位字节（MS）时，影子寄存器的更新被禁止直到低位字节（LS）的写操作完成。

28.7.2 输入模块

输入模块的框图



如图，输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边缘监测器产生一个信号（TIxFPx），它可以作为触发模式控制器的输入触发或者作为捕获控制。该信号通过预分频后进入捕获寄存器（ICxPS）。



28.7.3 输入捕获模式

在输入捕获模式下，当检测到 ICi 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器（PWMA_CCRx）中。当发生捕获事件时，相应的 CCiIF 标志（PWMA_SR 寄存器）被置 1。

如果 PWMA_IER 寄存器的 CCiIE 位被置位，也就是使能了中断，则将产生中断请求。如果发生捕获事件时 CCiIF 标志已经为高，那么重复捕获标志 CCiOF（PWMA_SR2 寄存器）被置 1。写 CCiIF=0 或读取存储在 PWMA_CCRiL 寄存器中的捕获数据都可清除 CCiIF。写 CCiOF=0 可清除 CCiOF。

PWM 输入信号上升沿时捕获

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 PWMA_CCR1 寄存器中，步骤如下：

1. 选择有效输入端：例如 PWMA_CCR1 连接到 TI1 输入，所以写入 PWMA_CCMR1 寄存器中的 CC1S=01，此时通道被配置为输入，并且 PWMA_CCR1 寄存器变为只读。
2. 根据输入信号 TIi 的特点，可通过配置 PWMA_CCMRi 寄存器中的 ICiF 位来设置相应的输入滤波器的滤波时间。假设输入信号在最多 5 个时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期;因此我们可以连续采样 8 次,以确认在 TI1 上一次真实的边沿变换,即在 TIMi_CCMR1 寄存器中写入 IC1F=0011，此时，只有连续采样到 8 个相同的 TI1 信号，信号才为有效（采样频率为 f_{MASTER}）。
3. 选择 TI1 通道的有效转换边沿，在 PWMA_CCER1 寄存器中写入 CC1P=0（上升沿）。
4. 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止（写 PWMA_CCMR1 寄存器的 IC1PS=00）。
5. 设置 PWMA_CCER1 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
6. 如果需要，通过设置 PWMA_IER 寄存器中的 CC1IE 位允许相关中断请求。

当发生一个输入捕获时：

- 当产生有效的电平转换时，计数器的值被传送到 PWMA_CCR1 寄存器。
- CC1IF 标志被设置。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除时，CC1OF 也被置 1。
- 如设置了 CC1IE 位，则会产生一个中断。

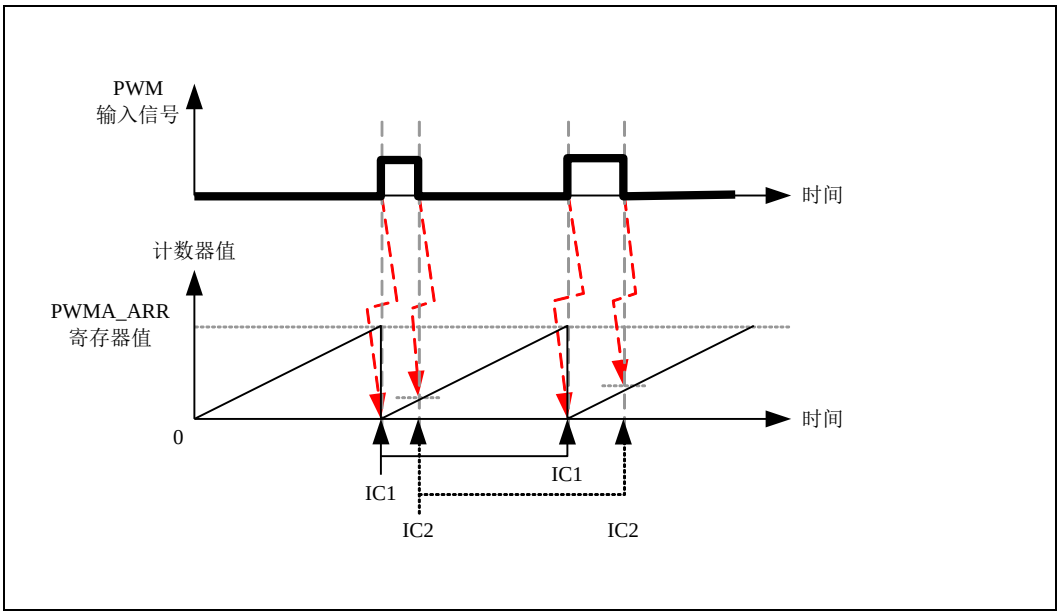
为了处理捕获溢出事件（CC1OF 位），建议在读出重复捕获标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的重复捕获信息。

注意：设置 PWMA_EGR 寄存器中相应的 CCiG 位，可以通过软件产生输入捕获中断。

PWM 输入信号测量

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

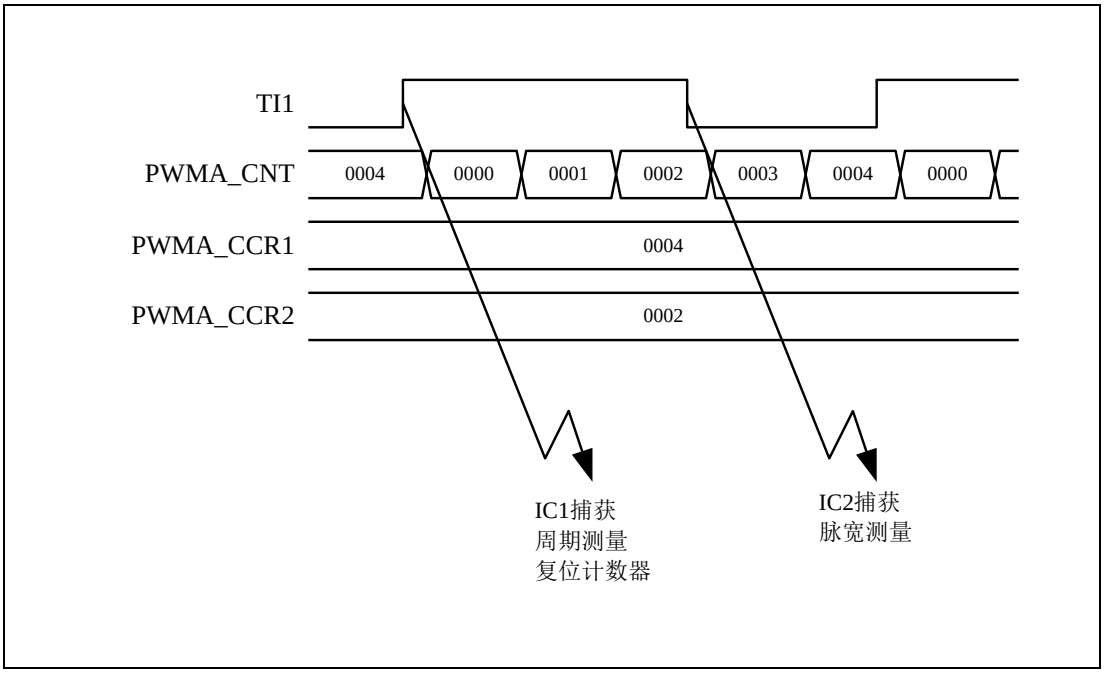
- 两个 ICi 信号被映射至同一个 TIi 输入。
- 这两个 ICi 信号的有效边沿的极性相反。
- 其中一个 TIiFP 信号被作为触发输入信号，而触发模式控制器被配置成复位触发模式。



例如，你可以用以下方式测量 TI1 上输入的 PWM 信号的周期（PWMA_CCR1 寄存器）和占空比（PWMA_CCR2 寄存器）。（具体取决于 f_{MASTER} 的频率和预分频器的值）

1. 选择 PWMA_CCR1 的有效输入：置 PWMA_CCMR1 寄存器的 CC1S=01（选中 TI1）。
2. 选择 TI1FP1 的有效极性（用来捕获数据到 PWMA_CCR1 中和清除计数器）：置 CC1P=0（上升沿有效）。
3. 选择 PWMA_CCR2 的有效输入：置 PWMA_CCMR2 寄存器的 CC2S=10（选中 TI1FP2）。
4. 选择 TI1FP2 的有效极性（捕获数据到 PWMA_CCR2）：置 CC2P=1（下降沿有效）。
5. 选择有效的触发输入信号：置 PWMA_SMCR 寄存器中的 TS=101（选择 TI1FP1）。
6. 配置触发模式控制器为复位触发模式：置 PWMA_SMCR 中的 SMS=100。
7. 使能捕获：置 PWMA_CCER1 寄存器中 CC1E=1，CC2E=1。

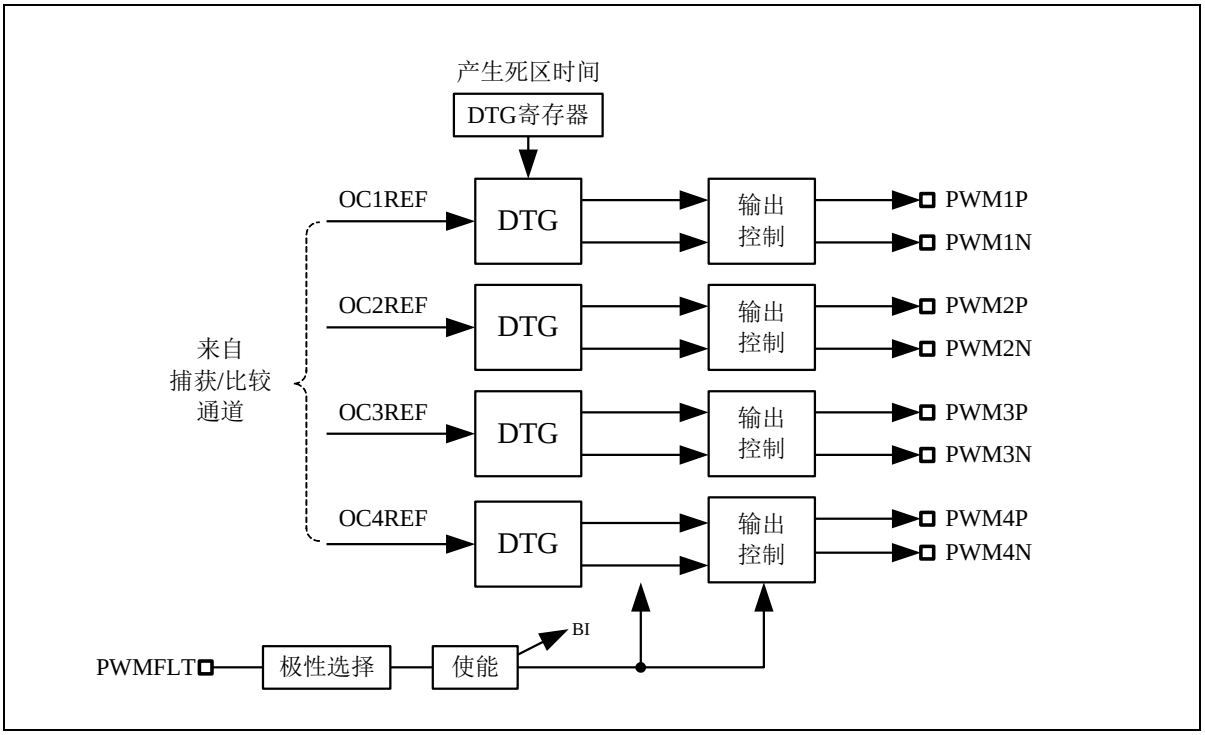
PWM 输入信号测量实例



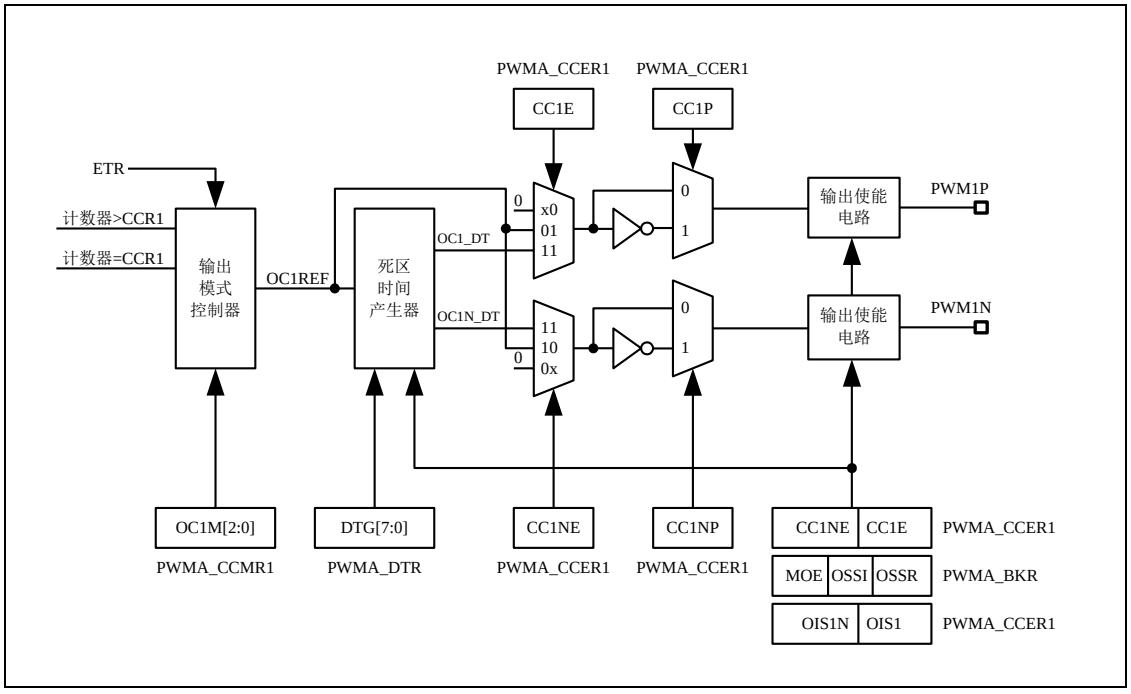
28.7.4 输出模块

输出模块会产生一个用来做参考的中间波形，称为 OCiREF（高有效）。刹车功能和极性的处理都在模块的最后处理。

输出模块框图



通道 1 详细的带互补输出的输出模块框图（其他通道类似）



28.7.5 强制输出模式

在输出模式下，输出比较信号能够直接由软件强制为高或低状态，而不依赖于输出比较寄存器和计数器间的比较结果。

置 PWMA_CCMRi 寄存器的 OCiM=101，可强制 OCiREF 信号为高。
置 PWMA_CCMRi 寄存器的 OCiM=100，可强制 OCiREF 信号为低。
OCi/OCiN 的输出是高还是低则取决于 CCiP/CCiNP 极性标志位。

该模式下，在 PWMA_CCRi 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改，也仍然会产生相应的中断。

28.7.6 输出比较模式

此模式用来控制一个输出波形或者指示一段给定的时间已经达到。

当计数器与捕获/比较寄存器的内容相匹配时，有如下操作：

- 根据不同的输出比较模式，相应的 OCi 输出信号：
  - 保持不变（OCiM=000）
  - 设置为有效电平（OCiM=001）
  - 设置为无效电平（OCiM=010）
  - 翻转（OCiM=011）
- 设置中断状态寄存器中的标志位（PWMA_SR1 寄存器中的 CCiIF 位）。
- 若设置了相应的中断使能位（PWMA_IER 寄存器中的 CCiIE 位），则产生一个中断。

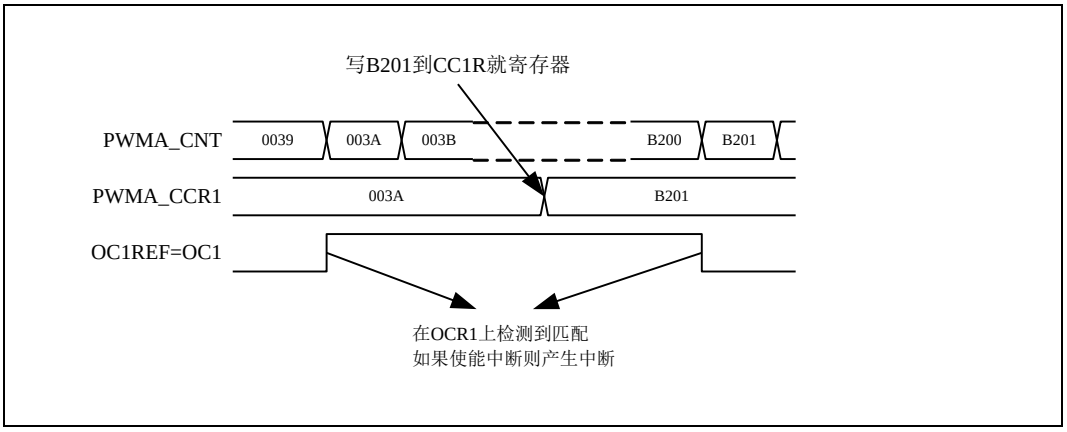
PWMA_CCMRi 寄存器的 OCiM 位用于选择输出比较模式，而 PWMA_CCMRi 寄存器的 CCiP 位用于选择有效和无效的电平极性。PWMA_CCMRi 寄存器的 OCiPE 位用于选择 PWMA_CCRi 寄存器是否需要使用预装载寄存器。在输出比较模式下，更新事件 UEV 对 OCiREF 和 OCi 输出没有影响。时间精度为计数器的一个计数周期。输出比较模式也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1. 选择计数器时钟（内部、外部或者预分频器）。
2. 将相应的数据写入 PWMA_ARR 和 PWMA_CCRi 寄存器中。
3. 如果要产生一个中断请求，设置 CCiIE 位。
4. 选择输出模式步骤：
   1. 设置 OCiM=011，在计数器与 CCRi 匹配时翻转 OCiM 管脚的输出
   2. 设置 OCiPE = 0，禁用预装载寄存器
   3. 设置 CCiP = 0，选择高电平为有效电平
   4. 设置 CCiE = 1，使能输出
   5. 设置 PWMA_CR1 寄存器的 CEN 位来启动计数器

PWMA_CCRi 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器（OCiPE=0），否则 PWMA_CCRi 的影子寄存器只能在发生下一次更新事件时被更新。

输出比较模式，翻转 OC1



28.7.7 PWM 模式

脉冲宽度调制（PWM）模式可以产生一个由 PWMA_ARR 寄存器确定频率，由 PWMA_CCRi 寄存器确定占空比的信号。

在 PWMA_CCMRi 寄存器中的 OCiM 位写入 110（PWM 模式 1）或 111（PWM 模式 2），能够独立地设置每个 OCi 输出通道产生一路 PWM。必须设置 PWMA_CCMRi 寄存器的 OCiPE 位使能相应的预装载寄存器，也可以设置 PWMA_CR1 寄存器的 ARPE 位使能自动重装载的预装载寄存器（在向上计数模式或中央对称模式中）。

由于仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 PWMA_EGR 寄存器的 UG 位来初始化所有的寄存器。

OCi 的极性可以通过软件在 PWMA_CCERi 寄存器中的 CCiP 位设置，它可以设置为高电平有效或低电平有效。OCi 的输出使能通过 PWMA_CCERi 和 PWMA_BKR 寄存器中的 CCiE、MOE、OISi、OSSR 和 OSSI 位的组合来控制。

在 PWM 模式（模式 1 或模式 2）下，PWMA_CNT 和 PWMA_CCRi 始终在进行比较，（依据计数器的计数方向）以确定是否符合 PWMA_CCRi≤PWMA_CNT 或者 PWMA_CNT≤PWMA_CCRi。

根据 PWMA_CR1 寄存器中 CMS 位域的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

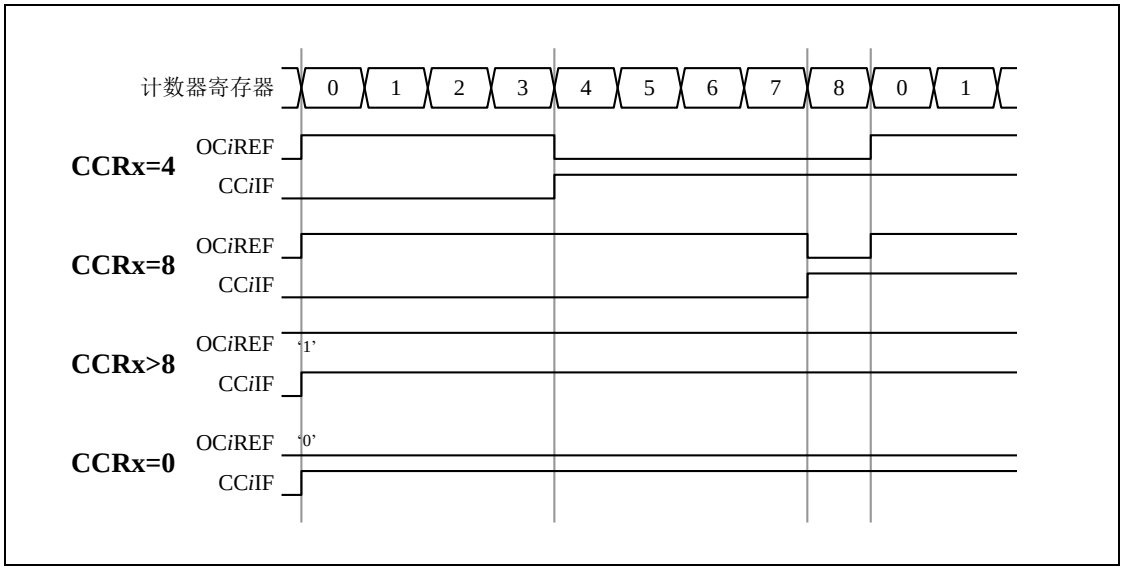
PWM 边沿对齐模式

向上计数配置

当 PWMA_CR1 寄存器中的 DIR 位为 0 时，执行向上计数。

下面是一个 PWM 模式 1 的例子。当 PWMA_CNT<PWMA_CCRi 时，PWM 参考信号 OCiREF 为高，否则为低。如果 PWMA_CCRi 中的比较值大于自动重装载值（PWMA_ARR），则 OCiREF 保持为’1’。如果比较值为 0，则 OCiREF 保持为’0’。

边沿对齐，PWM 模式 1 的波形（ARR=8）



向下计数的配置

当 PWMA_CR1 寄存器的 DIR 位为 1 时，执行向下计数。

在 PWM 模式 1 时，当 PWMA_CNT>PWMA_CCRi 时参考信号 OCiREF 为低，否则为高。如果 PWMA_CCRi 中的比较值大于 PWMA_ARR 中的自动重装载值，则 OCiREF 保持为'1'。该模式下不能产生 0%的 PWM 波形。

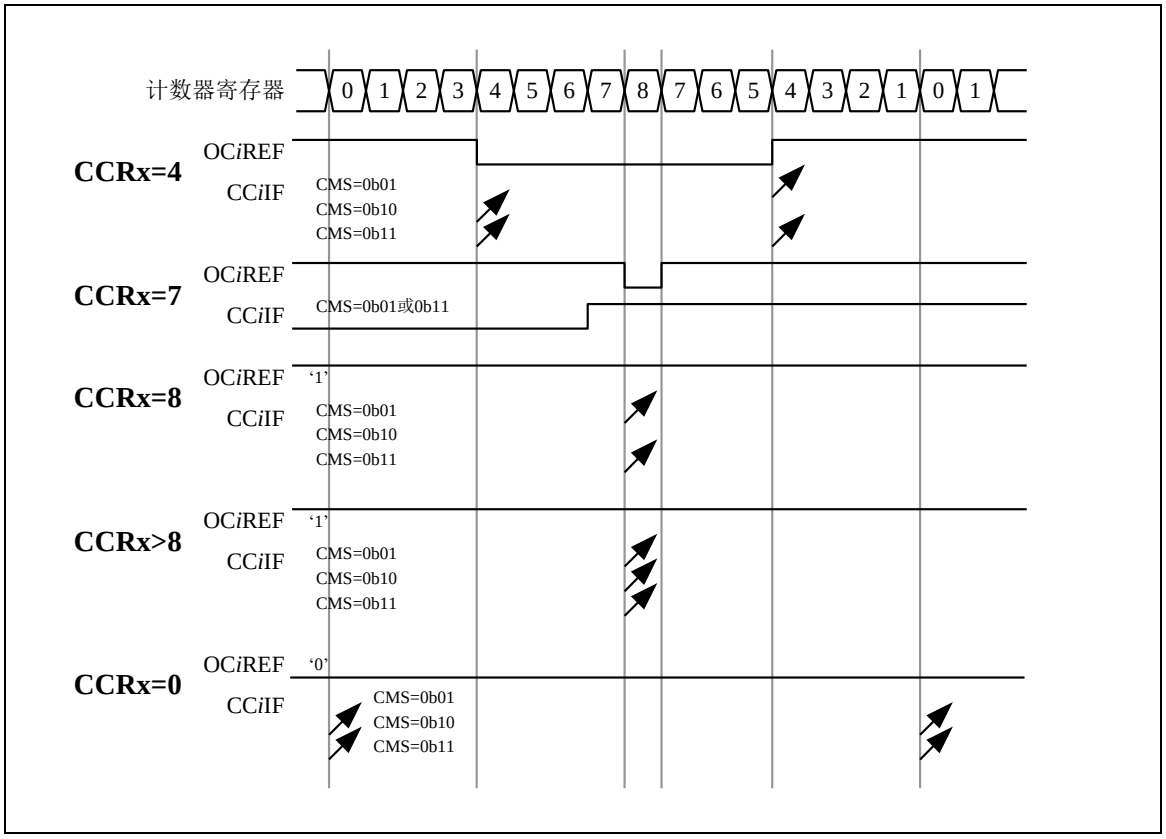
PWM 中央对齐模式

当 PWMA_CR1 寄存器中的 CMS 位不为'00'时为中央对齐模式（所有其他的配置对 OCiREF/OCi 信号都有相同的作用）。

根据不同的 CMS 位的设置，比较标志可以在计数器向上计数，向下计数，或向上和向下计数时被置 1。PWMA_CR1 寄存器中的计数方向位（DIR）由硬件更新，不要用软件修改它。

下面给出了一些中央对齐的 PWM 波形的例子：

- PWMA_ARR=8
- PWM 模式 1
- 标志位在以下三种情况下被置位：
  - 只有在计数器向下计数时（CMS=01）
  - 只有在计数器向上计数时（CMS=10）
  - 在计数器向上和向下计数时（CMS=11）
- 中央对齐的 PWM 波形（ARR=8）



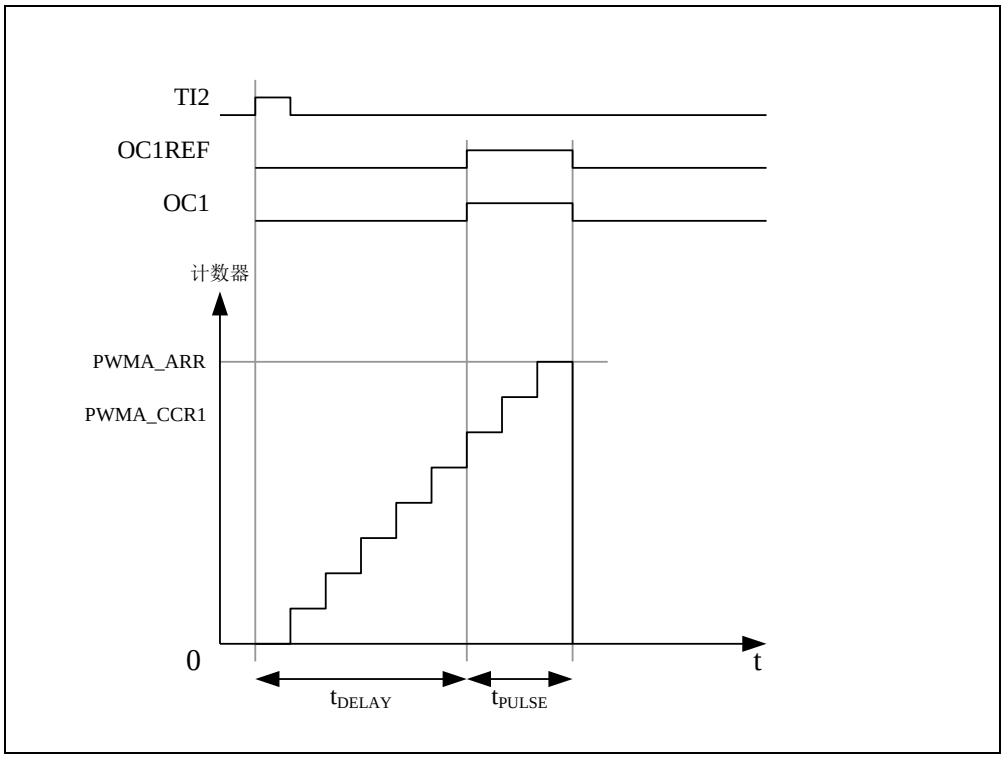
单脉冲模式

单脉冲模式（OPM）是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可控的脉冲。

可以通过时钟/触发控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 PWMA_CR1 寄存器的 OPM 位将选择单脉冲模式，此时计数器自动地在下一个更新事件 UEV 时停止。仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前（当定时器正在等待触发），必须如下配置：

- 向上计数方式：计数器 CNT < CCRi ≤ ARR，
- 向下计数方式：计数器 CNT > CCRi。

单脉冲模式图例



例如，在从 TI2 输入脚上检测到一个上升沿之后延迟 tDELAY，在 OC1 上产生一个 tPULSE 宽度的正脉冲：（假定 IC2 作为触发 1 通道的触发源）

- 置 PWMA_CCMR2 寄存器的 CC2S=01，把 IC2 映射到 TI2。
- 置 PWMA_CCER1 寄存器的 CC2P=0，使 IC2 能够检测上升沿。
- 置 PWMA_SMCR 寄存器的 TS=110，使 IC2 作为时钟/触发控制器的触发源（TRGI）。
- 置 PWMA_SMCR 寄存器的 SMS=110（触发模式），IC2 被用来启动计数器。OPM 的波形由写入比较寄存器的数值决定（要考虑时钟频率和计数器预分频器）。
- t_{DELAY} 由 PWMA_CCR1 寄存器中的值定义。
- t_{PULSE} 由自动装载值和比较值之间的差值定义（PWMA_ARR – PWMA_CCR1）。
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形，首先要置 PWMA_CCMR1 寄存器的 OCiM=111，进入 PWM 模式 2，根据需要有选择的设置 PWMA_CCMR1 寄存器的 OC1PE=1，置位 PWMA_CR1 寄存器中的 ARPE，使能预装载寄存器，然后在 PWMA_CCR1 寄存器中填写比较值，在 PWMA_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。

在这个例子中，PWMA_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以设置 PWMA_CR1 寄存器中的 OPM=1，在下一个更新事件（当计数器从自动装载值翻转到 0）时停止计数。

OCx 快速使能（特殊情况）

在单脉冲模式下，对 TIi 输入脚的边沿检测会设置 CEN 位以启动计数器，然后计数器和比较值间的比较操作产生了单脉冲的输出。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时 tDELAY。

如果要以最小延时输出波形，可以设置 PWMA_CCMRi 寄存器中的 OCiFE 位，此时强制 OCiREF（和 OCx）直接响应激励而不再依赖比较的结果，输出的波形与比较匹配时的波形一样。OCiFE 只在通道配

置为 PWMA 和 PWMB 模式时起作用。

互补输出和死区插入

PWMA 能够输出两路互补信号，并且能够管理输出的瞬时关断和接通，这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性（电平转换的延时、电源开关的延时等）来调整死区时间。

配置 PWMA_CCERi 寄存器中的 CCiP 和 CCiNP 位，可以为每一个输出独立地选择极性（主输出 OCi 或互补输出 OCiN）。

互补信号 OCi 和 OCiN 通过下列控制位的组合进行控制：PWMA_CCERi 寄存器的 CCiE 和 CCiNE 位，PWMA_BKR 寄存器中的 MOE、OISi、OISiN、OSSI 和 OSSR 位。特别的是，在转换到 IDLE 状态时（MOE 下降到 0）死区控制被激活。

同时设置 CCiE 和 CCiNE 位将插入死区，如果存在刹车电路，则还要设置 MOE 位。每一个通道都有一个 8 位的死区发生器。参考信号 OCiREF 可以产生 2 路输出 OCi 和 OCiN。

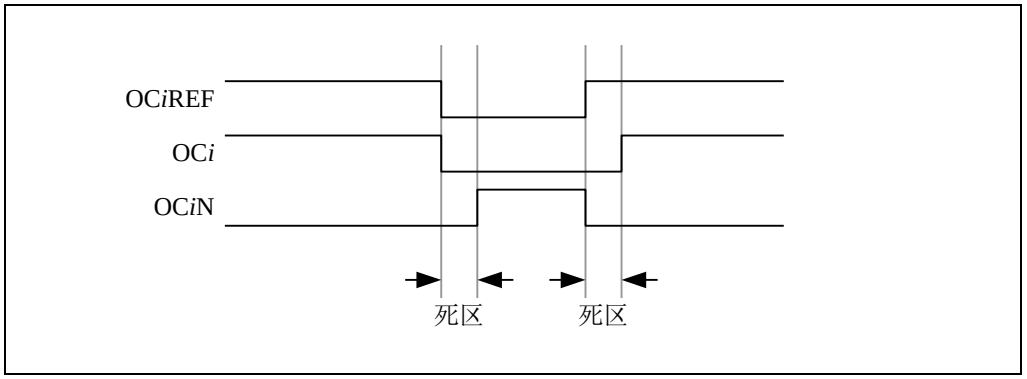
如果 OCi 和 OCiN 为高有效：

- OCi 输出信号与参考信号相同，只是它的上升沿相对于参考信号的上升沿有一个延迟。
- OCiN 输出信号与参考信号相反，只是它的上升沿相对于参考信号的下降沿有一个延迟。

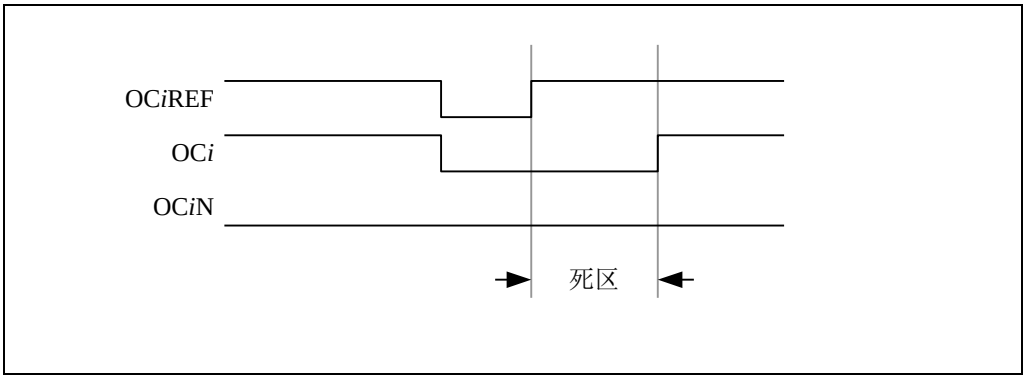
如果延迟大于当前有效的输出宽度（OCi 或者 OCiN），则不会产生相应的脉冲。

下列几张图显示了死区发生器的输出信号和当前参考信号 OCiREF 之间的关系。（假设 CCiP=0、CCiNP=0、MOE=1、CCiE=1 并且 CCiNE=1）

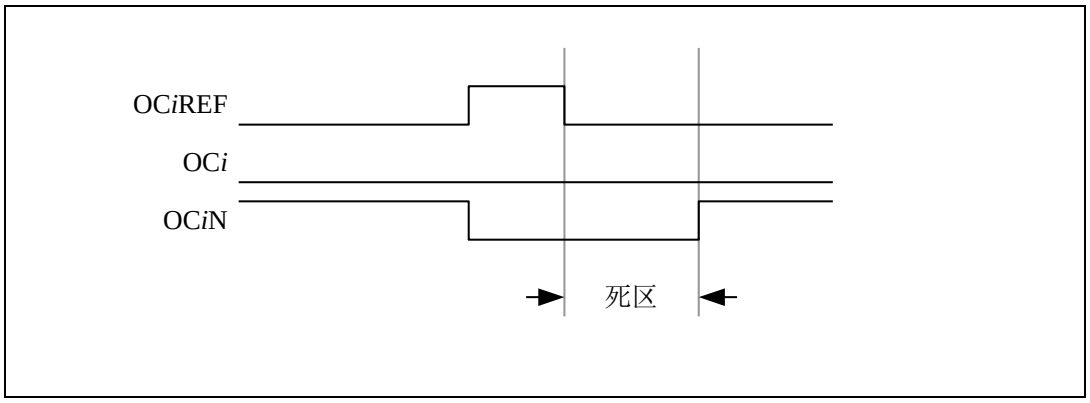
带死区插入的互补输出



死区波形延迟大于负脉冲



死区波形延迟大于正脉冲



每一个通道的死区延时都是相同的，是由 PWMA_DTR 寄存器中的 DTG 位编程配置。

重定向 OCiREF 到 OCi 或 OCiN

在输出模式下（强制输出、输出比较或 PWM 输出），通过配置 PWMA_CCERi 寄存器的 CCiE 和 CCiNE 位，OCiREF 可以被重定向到 OCi 或者 OCiN 的输出。

这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形（例如 PWM 或者静态有效电平）。另一个作用是，让两个输出同时处于无效电平，或同时处于有效电平（此时仍然是带死区的互补输出）。

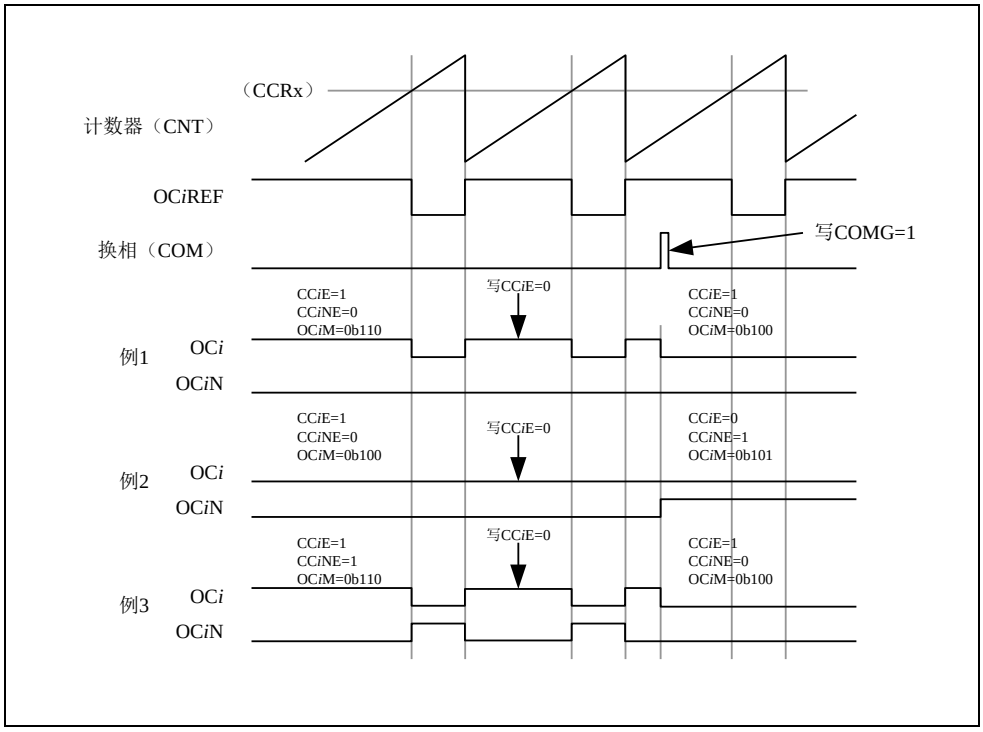
注：当只使能 OCiN（CCiE=0, CCiNE=1）时，它不会反相，而当 OCiREF 变高时立即有效。例如，如果 CCiNP=0，则 OCiN=OCiREF。另一方面，当 OCi 和 OCiN 都被使能时（CCiE=CCiNE=1），当 OCiREF 为高时 OCi 有效；而 OCiN 相反，当 OCiREF 低时 OCiN 变为有效。

针对马达控制的六步 PWM 输出

当在一个通道上需要互补输出时，预装载位有 OCiM、CCiE 和 CCiNE。在发生 COM 换相事件时，这些预装载位被传送到影子寄存器位。这样你就可以预先设置好下一步骤配置，并在同一个时刻同时修更改所有通道的配置。COM 可以通过设置 PWMA_EGR 寄存器的 COMG 位由软件产生，或在 TRGI 上升沿由硬件产生。

下图显示当发生 COM 事件时，三种不同配置下 OCx 和 OCxN 输出。

产生六步 PWM，使用 COM 的例子（OSSR=1）



28.7.8 使用刹车功能（PWMFLT）

刹车功能常用于马达控制中。当使用刹车功能时，依据相应的控制位（PWMA_BKR 寄存器中的 MOE、OSSI 和 OSSR 位），输出使能信号和无效电平都会被修改。

系统复位后，刹车电路被禁止，MOE 位为低。设置 PWMA_BKR 寄存器中的 BKE 位可以使能刹车功能。刹车输入信号的极性可以通过配置同一个寄存器中的 BKP 位选择。BKE 和 BKP 可以被同时修改。

MOE 下降沿相对于时钟模块可以是异步的，因此在实际信号（作用在输出端）和同步控制位（在 PWMA_BKR 寄存器中）之间设置了一个再同步电路。这个再同步电路会在异步信号和同步信号之间产生延迟。特别的，如果当它为低时写 MOE=1，则读出它之前必须先插入一个延时（空指令）才能读到正确的值。这是因为写入的是异步信号而读的是同步信号。

当发生刹车时（在刹车输入端出现选定的电平），有下述动作：

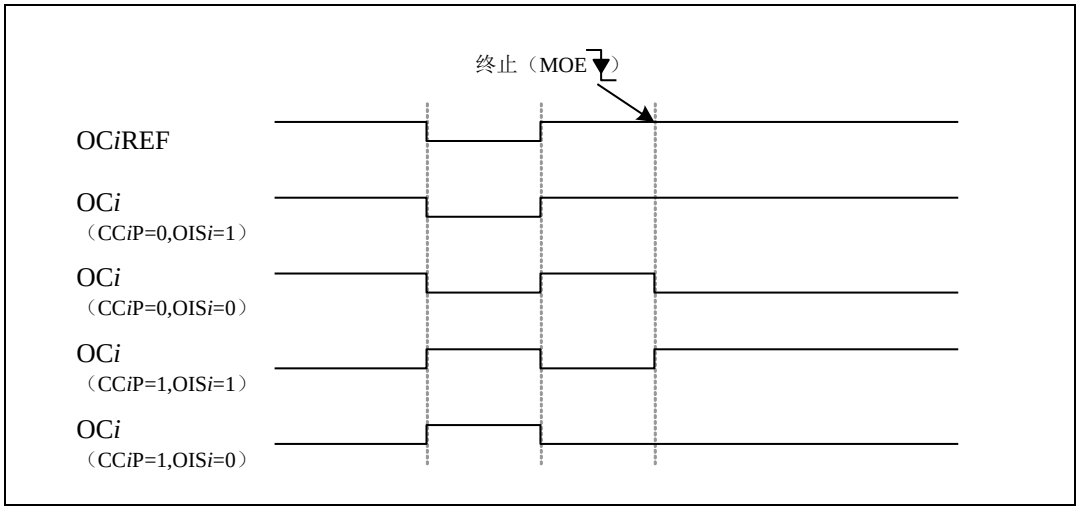
- MOE 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态（由 OSSI 位选择）。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 MOE=0，每一个输出通道输出由 PWMA_OISR 寄存器的 OISi 位设定的电平。如果 OSSI=0，则定时器不再控制输出使能信号，否则输出使能信号始终为高。
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态（取决于极性）。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISi 和 OISiN 位指示的电平驱动输出端口。即使在这种情况下，OCi 和 OCiN 也不能被同时驱动到有效的电平。注：因为重新同步 MOE，死区时间比通常情况下长一些（大约 2 个时钟周期）。
- 如果设置了 PWMA_IER 寄存器的 BIE 位，当刹车状态标志（PWMA_SR1 寄存器中的 BIF 位）为’1’时，则产生一个中断。
- 如果设置了 PWMA_BKR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位。

例如这可以用来进行波形控制，否则，MOE 始终保持低直到被再次置’1’。这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

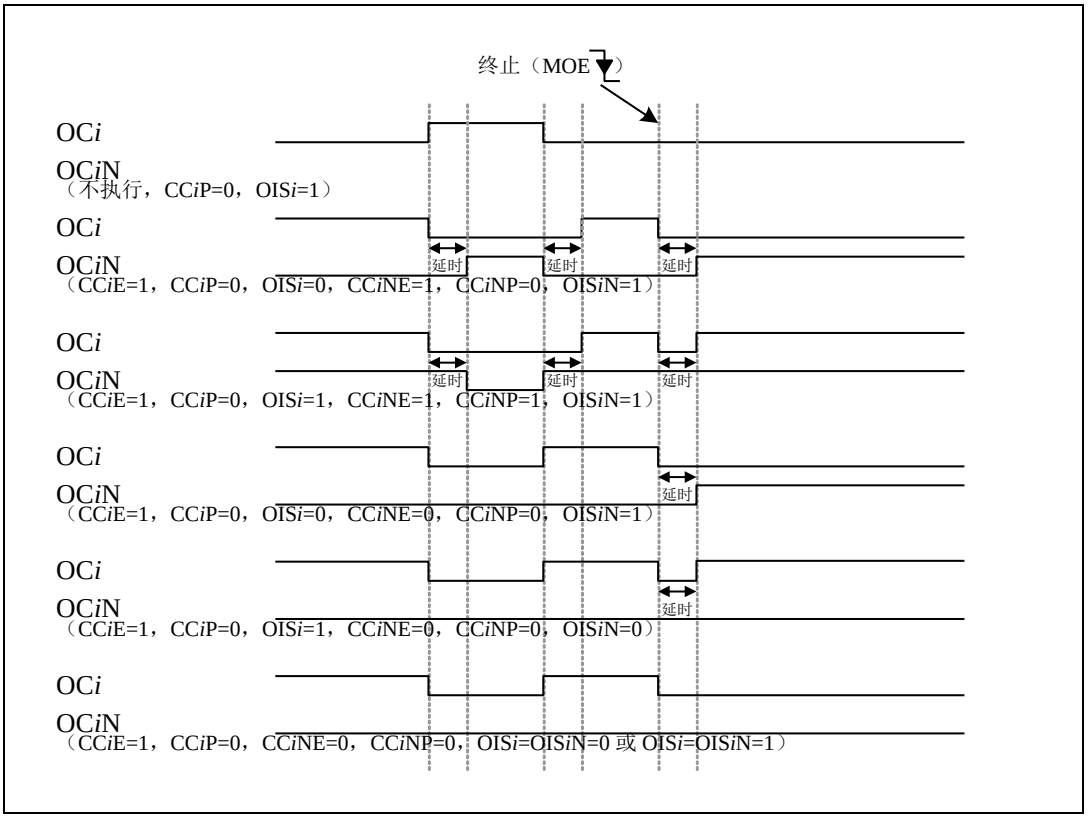
注：刹车输入为电平有效。所以，当刹车输入有效时，不能同时（自动地或者通过软件）设置 MOE。同时，状态标志 BIF 不能被清除。

刹车由 BRK 输入（BKIN）产生，它的有效极性是可编程的，且由 PWMA_BKR 寄存器的 BKE 位开启或禁止。除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数（OCi 极性和被禁止时的状态，OCiM 配置，刹车使能和极性）。用户可以通过 PWMA_BKR 寄存器的 LOCK 位，从三种级别的保护中选择一种。在 MCU 复位后 LOCK 位域只能被修改一次。

刹车响应的输出（不带互补输出的通道）



带互补输出的刹车响应的输出（PWMA 互补输出）



28.7.9 在外部事件发生时清除 OCiREF 信号

对于一个给定的通道，在 ETRF 输入端（设置 PWMA_CCMRi 寄存器中对应的 OCiCE 位为'1'）的高电平能够把 OCiREF 信号拉低，OCiREF 信号将保持为低直到发生下一次的更新事件 UEV。该功能只能用于输出比较模式和 PWM 模式，而不能用于强制模式。

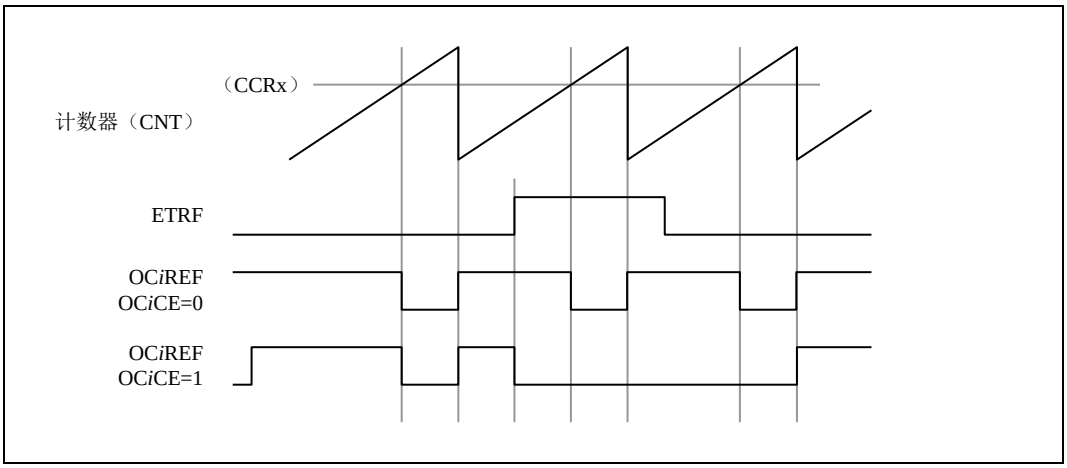
例如，OCiREF 信号可以联到一个比较器的输出，用于控制电流。这时，ETR 必须配置如下：

1. 外部触发预分频器必须处于关闭：PWMA_ETR 寄存器中的 ETPS[1:0]=00。
2. 必须禁止外部时钟模式 2：PWMA_ETR 寄存器中的 ECE=0。
3. 外部触发极性（ETP）和外部触发滤波器（ETF）可以根据需要配置。

下图显示了当 ETRF 输入变为高时，对应不同 OCiCE 的值，OCiREF 信号的动作。

在这个例子中，定时器 PWMA 被置于 PWM 模式。

ETR 清除 PWMA 的 OCiREF



28.7.10 编码器接口模式

编码器接口模式一般用于马达控制。

选择编码器接口模式的方法是：

- 如果计数器只在 TI2 的边沿计数，则置 PWMA_SMCR 寄存器中的 SMS=001；
- 如果只在 TI1 边沿计数，则置 SMS=010；
- 如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=011。

通过设置 PWMA_CCER1 寄存器中的 CC1P 和 CC2P 位，可以选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。假定计数器已经启动（PWMA_CR1 寄存器中的 CEN=1），则计数器在每次 TI1FP1 或 TI2FP2 上产生有效跳变时计数。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号。如果没有滤波和极性变换，则 TI1FP1=TI1，TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 PWMA_CR1 寄存器的 DIR 位进行相应的设置。不管计数器是依靠 TI1 计数、依靠 TI2 计数或者同时依靠 TI1 和 TI2 计数，在任一输入端（TI1 或者 TI2）的跳变都会重新计算 DIR 位。

编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 PWMA_ARR 寄存器的自动装载值之间连续计数（根据方向，或是 0 到 ARR 计数，或是 ARR 到 0 计数）。所以在开始计数之前必须配置 PWMA_ARR。在这种模式下捕获器、比较器、预分频器、重复计数器、触发输出特性等仍工作如常。编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。

编码器接口模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置，计数方向与相连的传感器旋转的方向对应。

下表列出了所有可能的组合（假设 TI1 和 TI2 不同时变换）。

计数方向与编码器信号的关系

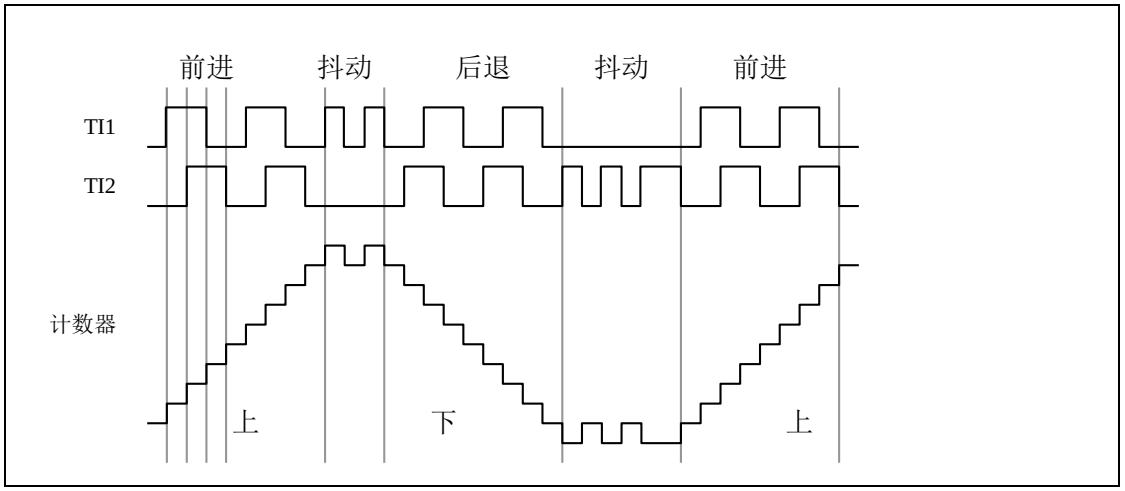
有效边沿	相对信号的电平 （TI1FP1 对应 TI2, TI2FP2 对应 TI1）	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在 TI2 计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
在 TI1 和 TI2 上计数	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般使用比较器将编码器的差分输出转换成数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

下面是一个计数器操作的实例，显示了计数信号的产生和方向控制。它还显示了当选择了双边沿时，输入抖动是如何被抑制的；抖动可能会在传感器的位置靠近一个转换点时产生。在这个例子中，我们假定配置如下：

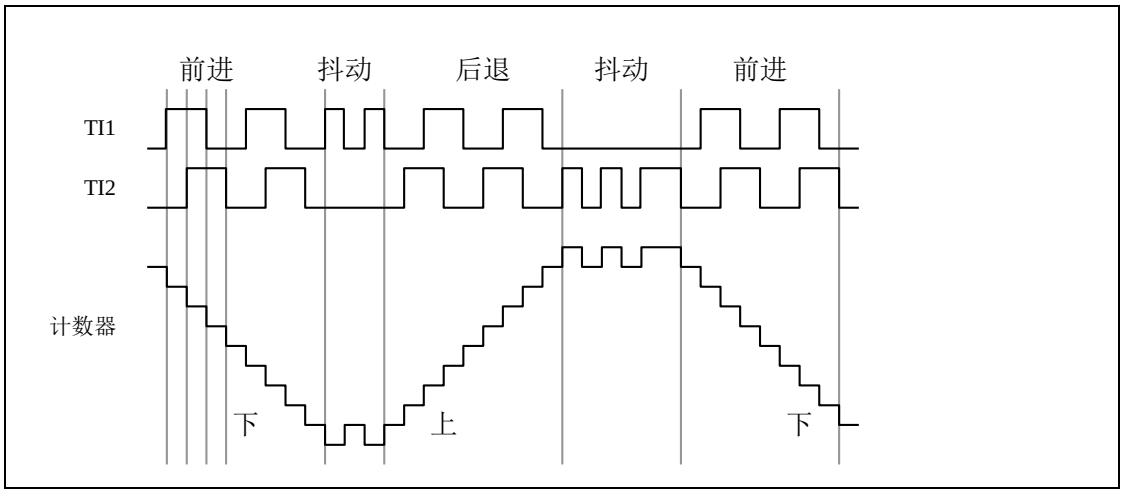
- CC1S=01（PWMA_CCMR1 寄存器，IC1FP1 映射到 TI1）
- CC2S=01（PWMA_CCMR2 寄存器，IC2FP2 映射到 TI2）
- CC1P=0（PWMA_CCER1 寄存器，IC1 不反相，IC1=TI1）
- CC2P=0（PWMA_CCER1 寄存器，IC2 不反相，IC2=TI2）
- SMS=011（PWMA_SMCR 寄存器，所有的输入均在上升沿和下降沿有效）.
- CEN=1（PWMA_CR1 寄存器，计数器使能）

编码器模式下的计数器操作实例



下图为当 IC1 极性反相时计数器的操作实例（CC1P=1，其他配置与上例相同）

IC1 反相的编码器接口模式实例



当定时器配置成编码器接口模式时，提供传感器当前位置的信息。使用另外一个配置在捕获模式下的定时器测量两个编码器事件的间隔，可以获得动态的信息（速度、加速度、减速度）。指示机械零点的编码器输出可被用做此目的。根据两个事件间的间隔，可以按照一定的时间间隔读出计数器。如果可能的话，你可以把计数器的值锁存到第三个输入捕获寄存器（捕获信号必须是周期的并且可以由另一个定时器产生）。

28.8 中断

PWMA 有 8 个中断请求源：

- 刹车中断
- 触发中断
- COM 事件中断
- 输入捕捉/输出比较 4 中断
- 输入捕捉/输出比较 3 中断

- 输入捕捉/输出比较 2 中断
- 输入捕捉/输出比较 1 中断
- 更新事件中断（如：计数器上溢，下溢及初始化）

为了使用中断特性，对每个被使用的中断通道，设置 PWMA_IER/PWMB_IER 寄存器中相应的中断使能位：即 BIE，TIE，COMIE，CCiIE，UIE 位。通过设置 PWMA_EGR/PWMB_EGR 寄存器中的相应位，也可以用软件产生上述各个中断源。

28.9 PWM 功能脚切换

28.9.1 PWMA 功能脚切换（PWMA_PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_PS	7EFEB2H	AC4PS[1:0]		AC3PS[1:0]		AC2PS[1:0]		AC1PS[1:0]	
PWMA_PS2	7EFEB8H	-		-		AC6PS[1:0]		AC5PS[1:0]	

AC1PS[1:0]：高级 PWMA 通道 1 输出脚选择位

AC1PS[1:0]	PWMA1P	PWMA1N
00	P1.0	P1.1
01	P2.0	P2.1
10	P6.0	P6.1
11	-	-

AC2PS[1:0]：高级 PWMA 通道 2 输出脚选择位

AC2PS[1:0]	PWMA2P	PWMA2N
00	P1.2	P1.3
01	P2.2	P2.3
10	P6.2	P6.3
11	-	-

AC3PS[1:0]：高级 PWMA 通道 3 输出脚选择位

AC3PS[1:0]	PWMA3P	PWM3AN
00	P1.4	P1.5
01	P2.4	P2.5
10	P6.4	P6.5
11	-	-

AC4PS[1:0]：高级 PWMA 通道 4 输出脚选择位

AC4PS[1:0]	PWMA4P	PWMA4N
00	P1.6	P1.7
01	P2.6	P2.7
10	P6.6	P6.7
11	P3.4	P3.3

AC5PS[1:0]：高级 PWMA 通道 5 输出脚选择位

AC5PS[1:0]	PWMA5
00	P4.6
01	P0.4
10	P7.2
11	P1.3

AC6PS[1:0]：高级 PWMA 通道 6 输出脚选择位

AC6PS[1:0]	PWMA6
00	P4.7
01	P0.5
10	P7.3
11	P1.5

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_ETRPS	7EFEB0H			CMPAPS[1:0]		BRKAPS[1:0]		ETRAPS[1:0]	

ETRAPS[1:0]：高级 PWMA 的外部触发脚 PWMAETR 选择位

ETRAPS [1:0]	PWMAETR
00	P3.2
01	P4.1
10	P7.3
11	-

BRKAPS[1:0]：高级 PWMA 的刹车脚 PWMAFLT 选择位

BRKAPS	PWMAFLT
00	P3.5
01	CMPAO
10	P4.5
11	-

CMPAPS[1:0]：高级 PWMA 的关联比较器选择位

CMPAPS	CMPAO
00	比较器 1 的输出
01	比较器 2 的输出
10	比较器 3 的输出
11	比较器 4 的输出

28.9.2 PWMB 功能脚切换（PWMB_PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMB_PS	7EFEB6H	BC8PS[1:0]		BC7PS[1:0]		BC6PS[1:0]		BC5PS[1:0]	

BC5PS[1:0]：高级 PWMB 通道 5 输出脚选择位

BC5PS[1:0]	PWMB5
00	P2.0
01	P1.7
10	P0.0
11	P7.4

BC6PS[1:0]：高级 PWMB 通道 6 输出脚选择位

BC6PS[1:0]	PWMB6
00	P2.1
01	P5.4
10	P0.1
11	P7.5

BC7PS[1:0]：高级 PWMB 通道 7 输出脚选择位

BC7PS[1:0]	PWMB7
00	P2.2
01	P3.3
10	P0.2
11	P7.6

BC8PS[1:0]：高级 PWM 通道 8 输出脚选择位

BC8PS[1:0]	PWMB8
00	P2.3
01	P3.4
10	P0.3
11	P7.7

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMB_ETRPS	7EFEB4H			CMPBPS[1:0]		BRKBPS[1:0]		ETRBPS[1:0]	

ETRBPS[1:0]：高级 PWMB 的外部触发脚 PWMBETR 选择位

ETRBPS [1:0]	PWMBETR
00	P3.2
01	P0.6
10	-
11	-

BRKBPS[1:0]：高级 PWMB 的刹车脚 PWMBFLT 选择位

BRKBPS	PWMBFLT
00	P3.5
01	CMPBO
10	P0.7
11	-

CMPBPS[1:0]：高级 PWMB 的关联比较器选择位

CMPBPS	CMPBO
00	比较器 1 的输出
01	比较器 2 的输出
10	比较器 3 的输出
11	比较器 4 的输出

28.9.3 PWMC 功能脚切换（PWMC_PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMC_PS	7EF8B2H	CC4PS[1:0]		CC3PS[1:0]		CC2PS[1:0]		CC1PS[1:0]	
PWMC_PS2	7EF8B8H	-		-		CC6PS[1:0]		CC5PS[1:0]	

CC1PS[1:0]：高级 PWMC 通道 1 输出脚选择位

CC1PS[1:0]	PWMC1P	PWMC1N
00	P6.0	P6.1
01	P8.0	P8.1
10	P4.0	P4.1
11	PB.0	PB.1

CC2PS[1:0]：高级 PWMC 通道 2 输出脚选择位

CC2PS[1:0]	PWMC2P	PWMC2N
00	P6.2	P6.3
01	P8.2	P8.3
10	P4.2	P4.3
11	PB.2	PB.3

CC3PS[1:0]：高级 PWMC 通道 3 输出脚选择位

CC3PS[1:0]	PWMC3P	PWM3AN
00	P6.4	P6.5
01	P8.4	P8.5
10	P4.4	P4.5
11	PB.4	PB.5

CC4PS[1:0]：高级 PWMC 通道 4 输出脚选择位

CC4PS[1:0]	PWMC4P	PWMC4N
00	P6.6	P6.7
01	P8.6	P8.7
10	P4.6	P4.7
11	PB.6	PB.7

CC5PS[1:0]：高级 PWMC 通道 5 输出脚选择位

CC5PS[1:0]	PWMC5
00	P1.1
01	P3.6
10	P9.6
11	P6.3

CC6PS[1:0]：高级 PWMC 通道 6 输出脚选择位

CC6PS[1:0]	PWMC6
00	P1.7
01	P3.7
10	P9.7
11	P6.5

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMC_ETRPS	7EF8B0H			CMPCPS[1:0]		BRKCPS[1:0]		ETRCPS[1:0]	

ETRCPS[1:0]：高级 PWMC 的外部触发脚 PWMCETR 选择位

ETRCPS [1:0]	PWMCETR
00	P3.2
01	P7.0
10	P9.4
11	-

BRKCPS[1:0]：高级 PWMC 的刹车脚 PWMCFLT 选择位

BRKCPS	PWMCFLT
00	P3.5
01	CMPCO
10	P7.1
11	P9.5

CMPCPS[1:0]：高级 PWMC 的关联比较器选择位

CMPCPS	CMPCO
00	比较器 1 的输出
01	比较器 2 的输出
10	比较器 3 的输出
11	比较器 4 的输出

28.9.4 PWMD 功能脚切换（PWMD_PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMD_PS	7EF8B6H	DC8PS[1:0]		DC7PS[1:0]		DC6PS[1:0]		DC5PS[1:0]	

DC5PS[1:0]：高级 PWMD 通道 5 输出脚选择位

DC5PS[1:0]	PWMD5
00	P2.4
01	P8.1
10	P5.0
11	PB.4

DC6PS[1:0]：高级 PWMD 通道 6 输出脚选择位

DC6PS[1:0]	PWMD6
00	P2.5
01	P8.3
10	P5.1
11	PB.5

DC7PS[1:0]：高级 PWMD 通道 7 输出脚选择位

DC7PS[1:0]	PWMD7
00	P2.6
01	P8.5
10	P5.2
11	PB.6

DC8PS[1:0]：高级 PWM 通道 8 输出脚选择位

DC8PS[1:0]	PWMD8
00	P2.7
01	P8.7
10	P5.3
11	PB.7

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMD_ETRPS	7EF8B4H			CMPDPS[1:0]		BRKDPS[1:0]		ETRDPS[1:0]	

ETRDPS[1:0]：高级 PWMD 的外部触发脚 PWMDETR 选择位

ETRDPS [1:0]	PWMDETR
00	P3.2
01	P7.0
10	PC.0
11	-

BRKDPS[1:0]：高级 PWMD 的刹车脚 PWMDFLT 选择位

BRKDPS	PWMDFLT
00	P3.5
01	CMPDO
10	P7.1
11	PC.1

CMPDPS[1:0]：高级 PWMD 的关联比较器选择位

CMPDPS	CMPDO
00	比较器 1 的输出
01	比较器 2 的输出
10	比较器 3 的输出
11	比较器 4 的输出

28.9.5 PWME 功能脚切换（PWME_PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWME_PS	7EF6B2H	EC4PS[1:0]		EC3PS[1:0]		EC2PS[1:0]		EC1PS[1:0]	
PWME_PS2	7EF6B8H	-		-		EC6PS[1:0]		EC5PS[1:0]	

EC1PS[1:0]：高级 PWME 通道 1 输出脚选择位

EC1PS[1:0]	PWME1P	PWME1N
00	P7.0	P7.1
01	P9.0	P9.1
10	PA.0	PA.1
11	-	-

EC2PS[1:0]：高级 PWME 通道 2 输出脚选择位

EC2PS[1:0]	PWME2P	PWME2N
00	P7.2	P7.3
01	P9.2	P9.3
10	PA.2	PA.3
11	-	-

EC3PS[1:0]：高级 PWME 通道 3 输出脚选择位

EC3PS[1:0]	PWME3P	PWM3AN
00	P7.4	P7.5
01	P9.4	P9.5
10	PA.4	PA.5
11	-	-

EC4PS[1:0]：高级 PWME 通道 4 输出脚选择位

EC4PS[1:0]	PWME4P	PWME4N
00	P7.6	P7.7
01	P9.6	P9.7
10	PA.6	PA.7
11	-	-

EC5PS[1:0]：高级 PWME 通道 5 输出脚选择位

EC5PS[1:0]	PWME5
00	P5.0
01	P4.3
10	PB.4
11	P9.5

EC6PS[1:0]：高级 PWME 通道 6 输出脚选择位

EC6PS[1:0]	PWME6
00	P5.1
01	P4.4
10	PB.5
11	P9.7

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWME_ETRPS	7EF6B0H			CMPEPS[1:0]		BRKEPS[1:0]		ETREPS[1:0]	

ETREPS[1:0]：高级 PWME 的外部触发脚 PWMEETR 选择位

ETREPS [1:0]	PWMEETR
00	P3.2
01	PC.2
10	P0.2
11	-

BRKEPS[1:0]：高级 PWME 的刹车脚 PWMEFLT 选择位

BRKEPS	PWMEFLT
00	P3.5
01	CMPEO
10	PC.3
11	P0.3

CMPEPS[1:0]：高级 PWME 的关联比较器选择位

CMPEPS	CMPEO
00	比较器 1 的输出
01	比较器 2 的输出
10	比较器 3 的输出
11	比较器 4 的输出

28.9.6 PWMF 功能脚切换（PWMF_PS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMF_PS	7EF6B6H	FC8PS[1:0]		FC7PS[1:0]		FC6PS[1:0]		FC5PS[1:0]	

FC5PS[1:0]：高级 PWMF 通道 5 输出脚选择位

FC5PS[1:0]	PWMF5
00	P7.0
01	PA.1
10	P9.0
11	-

FC6PS[1:0]：高级 PWMF 通道 6 输出脚选择位

FC6PS[1:0]	PWMF6
00	P7.1
01	PA.3
10	P9.1
11	-

FC7PS[1:0]：高级 PWMF 通道 7 输出脚选择位

FC7PS[1:0]	PWMF7
00	P7.2
01	PA.5
10	P9.2
11	-

FC8PS[1:0]：高级 PWM 通道 8 输出脚选择位

FC8PS[1:0]	PWMF8
00	P7.3
01	PA.7
10	P9.3
11	-

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMF_ETRPS	7EF6B4H			CMPFPS[1:0]		BRKFPS[1:0]		ETRFPS[1:0]	

ETRFPS[1:0]：高级 PWMF 的外部触发脚 PWMFETR 选择位

ETRFPS [1:0]	PWMFETR
00	P3.2
01	PB.1
10	P0.0
11	-

BRKFPS[1:0]：高级 PWMF 的刹车脚 PWMFFLT 选择位

BRKFPS	PWMFFLT
00	P3.5
01	CMPFO
10	PB.3
11	P0.1

CMPFPS[1:0]：高级 PWMF 的关联比较器选择位

CMPFPS	CMPFO
00	比较器 1 的输出
01	比较器 2 的输出
10	比较器 3 的输出
11	比较器 4 的输出

28.10 PWM 寄存器描述

28.10.1 输出使能寄存器（PWMn_ENO）

PWMA_ENO	7EFEB1H	ENOA4N	ENOA4P	ENOA3N	ENOA3P	ENOA2N	ENOA2P	ENOA1N	ENOA1P
PWMB_ENO	7EFEB5H	-	ENOB8P	-	ENOB7P	-	ENOB6P	-	ENOB5P
PWMC_ENO	7EF8B1H	ENOC4N	ENOC4P	ENOC3N	ENOC3P	ENOC2N	ENOC2P	ENOC1N	ENOC1P
PWMD_ENO	7EF8B5H	-	ENOD8P	-	ENOD7P	-	ENOD6P	-	ENOD5P
PWME_ENO	7EF6B1H	ENOE4N	ENOE4P	ENOE3N	ENOE3P	ENOE2N	ENOE2P	ENOE1N	ENOE1P
PWMF_ENO	7EF6B5H	-	ENOF8P	-	ENOF7P	-	ENOF6P	-	ENOF5P

ENOnxN：PWMnxN 输出控制位（n=A/B/C/D/E/F，x=1/2/3/4/5/6/7/8）

0：禁止 PWMnxN 输出

1：使能 PWMnxN 输出

ENOnxP：PWMnxP 输出控制位（n=A/B/C/D/E/F，x=1/2/3/4/5/6/7/8）

0：禁止 PWMnxP 输出

1：使能 PWMnxP 输出

28.10.2 输出附加使能寄存器（PWMn_IOAUX）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_IOAUX	7EFEB3H	AUXA4N	AUXA4P	AUXA3N	AUXA3P	AUXA2N	AUXA2P	AUXA1N	AUXA1P
PWMB_IOAUX	7EFEB7H	-	AUXB8P	-	AUXB7P	-	AUXB6P	-	AUXB5P
PWMC_IOAUX	7EF8B3H	AUXC4N	AUXC4P	AUXC3N	AUXC3P	AUXC2N	AUXC2P	AUXC1N	AUXC1P
PWMD_IOAUX	7EF8B7H	-	AUXD8P	-	AUXD7P	-	AUXD6P	-	AUXD5P
PWME_IOAUX	7EF6B3H	AUXE4N	AUXE4P	AUXE3N	AUXE3P	AUXE2N	AUXE2P	AUXE1N	AUXE1P
PWMF_IOAUX	7EF6B7H	-	AUXF8P	-	AUXF7P	-	AUXF6P	-	AUXF5P

AUXnxP：PWMnxP 输出控制位（n=A/B/C/D/E/F，x=1/2/3/4/5/6/7/8）

0：PWMnxP 的输出直接由 ENOnxP 控制

1：PWMnxP 的输出由 ENOnxP 和 PWMn_BKR 共同控制

AUXnxN：PWMnxN 输出附加控制位（n=A/B/C/D/E/F，x=1/2/3/4/5/6/7/8）

0：PWMnxN 的输出直接由 ENOnxP 控制

1：PWMnxN 的输出由 ENOnxP 和 PWMn_BKR 共同控制

28.10.3 控制寄存器 1（PWMn_CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CR1	7EFEC0H	ARPEA	CMSA[1:0]		DIRA	OPMA	URSA	UDISA	CENA
PWMB_CR1	7EFEE0H	ARPEB	CMSB[1:0]		DIRB	OPMB	URSB	UDISB	CENB
PWMC_CR1	7EF8C0H	ARPEC	CMSC[1:0]		DIRC	OPMC	URSC	UDISC	CENC
PWMD_CR1	7EF8E0H	ARPED	CMSD[1:0]		DIRD	OPMD	URSD	UDISD	CEND
PWME_CR1	7EF6C0H	ARPEE	CMSE[1:0]		DIRE	OPME	URSE	UDISE	CENE
PWMF_CR1	7EF6E0H	ARPEF	CMSF[1:0]		DIRF	OPMF	URSF	UDISF	CENF

ARPEn：自动预装载允许位（n=A/B/C/D/E/F）

0：PWMn_ARR 寄存器没有缓冲，它可以被直接写入

1：PWMn_ARR 寄存器由预装载缓冲器缓冲

CMSn[1:0]：选择对齐模式（n=A/B/C/D/E/F）

CMSn[1:0]	对齐模式	说明
00	边沿对齐模式	计数器依据方向位（DIR）向上或向下计数
01	中央对齐模式1	计数器交替地向上和向下计数。 配置为输出的通道的输出比较中断标志位(CCnIF)只在计数器向下计数时被置1。
10	中央对齐模式2	计数器交替地向上和向下计数。 配置为输出的通道的输出比较中断标志位(CCnIF)只在计数器向上计数时被置1。
11	中央对齐模式3	计数器交替地向上和向下计数。 配置为输出的通道的输出比较中断标志位(CCnIF)在计数器向上和向下计数时均被置1。

注 1：在计数器开启时（CEN=1），不允许从边沿对齐模式转换到中央对齐模式。

注 2：在中央对齐模式下，编码器模式（SMS=001，010，011）必须被禁止。

DIRn：计数器的计数方向（n=A/B/C/D/E/F）

0：计数器向上计数；

1：计数器向下计数。

注：当计数器配置为中央对齐模式或编码器模式时，该位为只读。

OPMn：单脉冲模式（n=A/B/C/D/E/F）

0：在发生更新事件时，计数器不停止；

1：在发生下一次更新事件时，清除 CEN 位，计数器停止。

URSn：更新请求源（n=A/B/C/D/E/F）

0：如果 UDIS 允许产生更新事件，则下述任一事件产生一个更新中断：

- 寄存器被更新（计数器上溢/下溢）
- 软件设置 UG 位
- 时钟/触发控制器产生的更新

1：如果 UDIS 允许产生更新事件，则只有当下列事件发生时才产生更新中断，并 UIF 置 1：

- 寄存器被更新（计数器上溢/下溢）

UDISn：禁止更新（n=A/B/C/D/E/F）

0：一旦下列事件发生，产生更新（UEV）事件：

- 计数器溢出/下溢
- 产生软件更新事件
- 时钟/触发模式控制器产生的硬件复位 被缓存的寄存器被装入它们的预装载值。

1：不产生更新事件，影子寄存器（ARR、PSC、CCRx）保持它们的值。如果设置了 UG 位或时钟/触发控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。

CENn：允许计数器（n=A/B/C/D/E/F）

0：禁止计数器；

1：使能计数器。

注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。然而触发模式可以自动地通过硬件设置 CEN 位。

28.10.4 控制寄存器 2（PWMn_CR2），及实时触发 ADC

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CR2	7EFEC1H	TIA1S	MMSA[2:0]			-	COMSA	-	CCPCA
PWMB_CR2	7EFEE1H	TIB5S	MMSB[2:0]			-	COMSB	-	CCPCB
PWMC_CR2	7EF8C1H	TIC1S	MMSC[2:0]			-	COMSC	-	CCPCC
PWMD_CR2	7EF8E1H	TID5S	MMSD[2:0]			-	COMSD	-	CCPCD
PWME_CR2	7EF6C1H	TIE1S	MMSE[2:0]			-	COMSE	-	CCPCE
PWMF_CR2	7EF6E1H	TIF5S	MMSF[2:0]			-	COMSF	-	CCPCF

TIn1S：PWMn 的 TIn1 选择（n=A/C/E）

0：PWMn1P 输入管脚连到 TIn1（数字滤波器的输入）；

1：PWMn1P、PWMn2P 和 PWMn3P 管脚经异或后连到 TIn1。

TIn5S：PWMn 的 TIn5 选择（n=B/D/F）

0：PWMn5 输入管脚连到 TIn5（数字滤波器的输入）；

1：PWMn5、PWMn6 和 PWMn7 管脚经异或后连到 TIn5。

MMSn[2:0]：主模式选择（n=A/C/E）

MMSn[2:0]	主模式	说明
000	复位	PWMn_EGR寄存器的UG位被用于作为触发输出（TRGO）。如果触发输入（时钟/触发控制器配置为复位模式）产生复位，则TRGO上的信号相对实际的复位会有一个延迟
001	使能	计数器使能信号被用于作为触发输出（TRGO）。其用于启动ADC，以便控制在一段时间内使能ADC。计数器使能信号是通过CEN控制位和门控模式下的触发输入信号的逻辑或产生。除非选择了主/从模式，当计数器使能信号受控于触发输入时，TRGO上会有一个延迟。 **注：当需要使用PWM触发ADC转换时，需要先设置ADC_CONTR寄存器中的ADC_POWER、ADC_CHS以及ADC_EPWMT，当PWM产生TRGO内部信号时，系统会自动设置ADC_START来启动AD转换。**
010	更新	更新事件被选为触发输出（TRGO）
011	比较脉冲	一旦发生一次捕获或一次比较成功，当CCn1IF标志被置1时，触发输出送出一个正脉冲（TRGO）
100	比较	OCn1REF信号被用于作为触发输出（TRGO）
101	比较	OCn2REF信号被用于作为触发输出（TRGO）
110	比较	OCn3REF信号被用于作为触发输出（TRGO）
111	比较	OCn4REF信号被用于作为触发输出（TRGO）

MMSn[2:0]：主模式选择（n=B/D/F）

MMSn[2:0]	主模式	说明
000	复位	PWMn_EGR寄存器的UG位被用于作为触发输出（TRGO）。如果触发输入（时钟/触发控制器配置为复位模式）产生复位，则TRGO上的信号相对实际的复位会有一个延迟
001	使能	计数器使能信号被用于作为触发输出（TRGO）。其用于启动多个PWM，以便控制在一段时间内使能从PWM。计数器使能信号是通过CEN控制位和门控模式下的触发输入信号的逻辑或产生。除非选择了主/从模式，当计数器使能信号受控于触发输入时，TRGO上会有一个延迟。
010	更新	更新事件被选为触发输出（TRGO）
011	比较脉冲	一旦发生一次捕获或一次比较成功，当CCn5IF标志被置1时，触发输出送出一个正脉冲（TRGO）
100	比较	OCn5REF信号被用于作为触发输出（TRGO）
101	比较	OCn6REF信号被用于作为触发输出（TRGO）
110	比较	OCn7REF信号被用于作为触发输出（TRGO）
111	比较	OCn8REF信号被用于作为触发输出（TRGO）

COMSn：捕获/比较控制位的更新控制选择（n=A/B/C/D/E/F）

0：当 CCPCn=1 时，只有在 COMG 位置 1 的时候这些控制位才被更新

1：当 CCPCn=1 时，只有在 COMG 位置 1 或 TRGI 发生上升沿的时候这些控制位才被更新

CCPCn：捕获/比较预装载控制位（n=A/B/C/D/E/F）

0：CCIE，CCINE，CCiP，CCiNP 和 OCIM 位不是预装载的

1：CCIE，CCINE，CCiP，CCiNP 和 OCIM 位是预装载的；设置该位后，它们只在设置了 COMG 位后被更新。

注：该位只对具有互补输出的通道起作用。

28.10.5 从模式控制寄存器(PWMn_SMCR)

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_SMCR	7EFEC2H	MSMA	TSA[2:0]			-	SMSA[2:0]		
PWMB_SMCR	7EFEE2H	MSMB	TSB[2:0]			-	SMSB[2:0]		
PWMC_SMCR	7EF8C2H	MSMC	TSC[2:0]			-	SMSC[2:0]		
PWMD_SMCR	7EF8E2H	MSMD	TSD[2:0]			-	SMSD[2:0]		
PWME_SMCR	7EF6C2H	MSME	TSE[2:0]			-	SMSE[2:0]		
PWMF_SMCR	7EF6E2H	MSMF	TSF[2:0]			-	SMSF[2:0]		

MSMn：主/从模式（n=A/B/C/D/E/F）

0：无作用

1：触发输入（TRGI）上的事件被延迟了，以允许 PWMn 与它的从 PWM 间的完美同步（通过 TRGO）

TSn[2:0]：触发源选择（n=A/C/E）

TSn[2:0]	触发源
000	-
001	-
010	内部触发ITRn2
011	-
100	TIn1的边沿检测器（TIn1F_ED）
101	滤波后的定时器输入1（TIn1FP1）
110	滤波后的定时器输入2（TIn2FP2）
111	外部触发输入（ETRF）

TSn[2:0]：触发源选择（n=B/D/F）

TSn[2:0]	触发源
000	-
001	-
010	-
011	-
100	TIn5的边沿检测器（TIn5F_ED）
101	滤波后的定时器输入1（TIn5FP5）
110	滤波后的定时器输入2（TIn6FP6）
111	外部触发输入（ETRF）

注：这些位只能在 SMS=000 时被改变，以避免在改变时产生错误的边沿检测。

SMSn[2:0]：时钟/触发/从模式选择（n=A/C/E）

SMSn[2:0]	功能	说明
000	内部时钟模式	如果CENn=1，则预分频器直接由内部时钟驱动
001	编码器模式1	根据TIn1FP1的电平，计数器在TIn2FP2的边沿向上/下计数
010	编码器模式2	根据TIn2FP2的电平，计数器在TIn1FP1的边沿向上/下计数
011	编码器模式3	根据另一个输入的电平，计数器在TIn1FP1和TIn2FP2的边沿向上/下计数
100	复位模式	在选中的触发输入（TRGI）的上升沿时重新初始化计数器，并且产生一个更新 寄存器的信号
101	门控模式	当触发输入（TRGI）为高时，计数器的时钟开启。一旦触发输入变为低，则计 数器停止（但不复位）。计数器的启动和停止都是受控的
110	触发模式	计数器在触发输入TRGI的上升沿启动（但不复位），只有计数器的启动是受控 的
111	外部时钟模式1	选中的触发输入（TRGI）的上升沿驱动计数器。 注：如果TIn1F_ED被选为触发输入（TS=100）时，不要使用门控模式。这是因为TIn1F_ED在每次TIn1F变化时只是输出一个脉冲，然而门控模式是要检查触发输入的电平

SMSn[2:0]：时钟/触发/从模式选择（n=B/D/F）

SMSn[2:0]	功能	说明
000	内部时钟模式	如果CENn=1，则预分频器直接由内部时钟驱动
001	编码器模式1	根据TIn5FP5的电平，计数器在TIn6FP6的边沿向上/下计数
010	编码器模式2	根据TIn6FP6的电平，计数器在TIn5FP5的边沿向上/下计数
011	编码器模式3	根据另一个输入的电平，计数器在TIn5FP5和TIn6FP6的边沿向上/下计数
100	复位模式	在选中的触发输入（TRGI）的上升沿时重新初始化计数器，并且产生一个更新 寄存器的信号
101	门控模式	当触发输入（TRGI）为高时，计数器的时钟开启。一旦触发输入变为低，则计 数器停止（但不复位）。计数器的启动和停止都是受控的
110	触发模式	计数器在触发输入TRGI的上升沿启动（但不复位），只有计数器的启动是受控的
111	外部时钟模式1	选中的触发输入（TRGI）的上升沿驱动计数器。 注：如果TIn5F_ED被选为触发输入（TS=100）时，不要使用门控模式。这是因为TIn5F_ED在每次TIn5F变化时只是输出一个脉冲，然而门控模式是要检查触发输入的电平

28.10.6 外部触发寄存器(PWMn_ETR)

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_ETR	7EFEC3H	ETPA	ECEA	ETPSA[1:0]		ETFA[3:0]			
PWMB_ETR	7EFEE3H	ETPB	ECEB	ETPSB[1:0]		ETFB[3:0]			
PWMC_ETR	7EF8C3H	ETPC	ECEC	ETPSC[1:0]		ETFC[3:0]			
PWMD_ETR	7EF8E3H	ETPD	ECED	ETPSD[1:0]		ETFD[3:0]			
PWME_ETR	7EF6C3H	ETPE	ECEE	ETPSE[1:0]		ETFE[3:0]			
PWMF_ETR	7EF6E3H	ETPF	ECEF	ETPSF[1:0]		ETFF[3:0]			

ETPn：外部触发 ETR 的极性（n=A/B/C/D/E/F）

0：高电平或上升沿有效

1：低电平或下降沿有效

ECEn：外部时钟使能（n=A/B/C/D/E/F）

0：禁止外部时钟模式 2

1：使能外部时钟模式 2，计数器的时钟为 ETRF 的有效沿。

注 1：ECE 置 1 的效果与选择把 TRGI 连接到 ETRF 的外部时钟模式 1 相同（PWMn_SMCR 寄存器中，SMS=111，TS=111）。

注 2：外部时钟模式 2 可与下列模式同时使用：触发标准模式；触发复位模式；触发门控模式。但是，此时 TRGI 决不能与 ETRF 相连（PWMn_SMCR 寄存器中，TS 不能为 111）。

注 3：外部时钟模式 1 与外部时钟模式 2 同时使能，外部时钟输入为 ETRF。

ETPSn：外部触发预分频器外部触发信号 EPRP 的频率最大不能超过 fMASTER/4。可用预分频器来降低 ETRP 的频率，当 EPRP 的频率很高时，它非常有用：（n=A/B/C/D/E/F）

00：预分频器关闭

01：EPRP 的频率/2

02：EPRP 的频率/4

03：EPRP 的频率/8

ETFn[3:0]：外部触发滤波器选择，该位域定义了 ETRP 的采样频率及数字滤波器长度。（n=A/B/C/D/E/F）

ETFn[3:0]	时钟数	ETF[3:0]	时钟数
0000	1	1000	48
0001	2	1001	64
0010	4	1010	80
0011	8	1011	96
0100	12	1100	128
0101	16	1101	160
0110	24	1110	192
0111	32	1111	256

28.10.7 中断使能寄存器(PWMn_IER)

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_IER	7EFEC4H	BIEA	TIEA	COMIEA	CCA4IE	CCA3IE	CCA2IE	CCA1IE	UIEA
PWMB_IER	7EFEE4H	BIEB	TIEB	COMIEB	CCB8IE	CCB7IE	CCB6IE	CCB5IE	UIEB
PWMC_IER	7EF8C4H	BIEC	TIEC	COMIEC	CCC4IE	CCC3IE	CCC2IE	CCC1IE	UIEC
PWMD_IER	7EF8E4H	BIED	TIED	COMIED	CCD8IE	CCD7IE	CCD6IE	CCD5IE	UIED
PWME_IER	7EF6C4H	BIEE	TIEE	COMIEE	CCE4IE	CCE3IE	CCE2IE	CCE1IE	UIEE
PWMF_IER	7EF6E4H	BIEF	TIEF	COMIEF	CCF8IE	CCF7IE	CCF6IE	CCF5IE	UIEF

BIEn：允许刹车中断（n=A/B/C/D/E/F）

0：禁止刹车中断；

1：允许刹车中断。

TIE：触发中断使能（n=A/B/C/D/E/F）

0：禁止触发中断；

1：使能触发中断。

COMIE：允许 COM 中断（n=A/B/C/D/E/F）

0：禁止 COM 中断；

1：允许 COM 中断。

CCnxIE：允许捕获/比较 n 中断（n=A/B/C/D/E/F，x=1/2/3/4/5/6/7/8）

0：禁止捕获/比较 n 中断；

1：允许捕获/比较 n 中断。

UIEn：允许更新中断（n=A/B/C/D/E/F）

0：禁止更新中断；

1：允许更新中断。

28.10.8 状态寄存器 1(PWMn_SR1)

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_SR1	7EFEC5H	BIFA	TIFA	COMIFA	CCA4IF	CCA3IF	CCA2IF	CCA1IF	UIFA
PWMB_SR1	7EFEE5H	BIFB	TIFB	COMIFB	CCB8IF	CCB7IF	CCB6IF	CCB5IF	UIFB
PWMC_SR1	7EF8C5H	BIFC	TIFC	COMIFC	CCC4IF	CCC3IF	CCC2IF	CCC1IF	UIFA
PWMD_SR1	7EF8E5H	BIFD	TIFD	COMIFD	CCD8IF	CCD7IF	CCD6IF	CCD5IF	UIFB
PWME_SR1	7EF6C5H	BIFE	TIFE	COMIFE	CCE4IF	CCE3IF	CCE2IF	CCE1IF	UIFA
PWMF_SR1	7EF6E5H	BIFF	TIFF	COMIFF	CCF8IF	CCF7IF	CCF6IF	CCF5IF	UIFB

BIFn：刹车中断标记。一旦刹车输入有效，由硬件对该位置 1。

如果刹车输入无效，则该位可由软件清 0。（n=A/B/C/D/E/F）

0：无刹车事件产生

1：刹车输入上检测到有效电平

TIFn：触发器中断标记。当发生触发事件时由硬件对该位置 1。由软件清 0。（n=A/B/C/D/E/F）

0：无触发器事件产生

1：触发中断等待响应

COMIFn：COM 中断标记。一旦产生 COM 事件该位由硬件置 1。由软件清 0。（n=A/B/C/D/E/F）

0：无 COM 事件产生

1：COM 中断等待响应

CCnxIF：捕获/比较nx中断标记，参考CC1IF描述（n=A/B/C/D/E/F，x=1/2/3/4/5/6/7/8）

CC1IF：捕获/比较1中断标记。

如果通道CC1配置为输出模式：

当计数器值与比较值匹配时该位由硬件置1，但在中心对称模式下除外。它由软件清0。

0：无匹配发生；

1：PWMA_CNT 的值与 PWMA_CCR1 的值匹配。

注：在中心对称模式下，当计数器值为 0 时，向上计数，当计数器值为 ARR 时，向下计数（它从 0 向上计数到 ARR-1，再由 ARR 向下计数到 1）。因此，对所有的 SMS 位值，这两个值都不置标记。但是，如果 CCR1>ARR，则当 CNT 达到 ARR 值时，CC1IF 置 1。

如果通道CC1配置为输入模式：

当捕获事件发生时该位由硬件置1，它由软件清0或通过读PWMA_CCR1L清0。

0：无输入捕获产生

1：计数器值已被捕获至 PWMA_CCR1

UIFn：更新中断标记 当产生更新事件时该位由硬件置 1。它由软件清 0。（n=A/B/C/D/E/F）

0：无更新事件产生

1：更新事件等待响应。当寄存器被更新时该位由硬件置 1

- 若 PWMn_CR1 寄存器的 UDIS=0，当计数器上溢或下溢时
- 若 PWMn_CR1 寄存器的 UDIS=0、URS=0，当设置 PWMn_EGR 寄存器的 UG 位软件对计数器 CNT 重新初始化时
- 若 PWMn_CR1 寄存器的 UDIS=0、URS=0，当计数器 CNT 被触发事件重新初始化时

28.10.9 状态寄存器 2(PWMn_SR2)

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_SR2	7EFEC6H	-	-	-	CCA4OF	CCA3OF	CCA2OF	CCA1OF	-
PWMB_SR2	7EFEE6H	-	-	-	CCB8OF	CCB7OF	CCB6OF	CCB5OF	-
PWMC_SR2	7EF8C6H	-	-	-	CCC4OF	CCC3OF	CCC2OF	CCC1OF	-
PWMD_SR2	7EF8E6H	-	-	-	CCD8OF	CCD7OF	CCD6OF	CCD5OF	-
PWME_SR2	7EF6C6H	-	-	-	CCE4OF	CCE3OF	CCE2OF	CCE1OF	-
PWMF_SR2	7EF6E6H	-	-	-	CCF8OF	CCF7OF	CCF6OF	CCF5OF	-

CCnxOF：捕获/比较nx重复捕获标记。参见CC1OF描述。（n=A/B/C/D/E/F，x=1/2/3/4/5/6/7/8）

CC1OF：捕获/比较1重复捕获标记。仅当相应的通道被配置为输入捕获时，该标记可由硬件置1。写0可清除该位。

0：无重复捕获产生；

1：计数器的值被捕获到 PWMA_CCR1 寄存器时，CC1IF 的状态已经为 1。

28.10.10 事件产生寄存器（PWMx_EGR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_EGR	7EFEC7H	BGA	TGA	COMGA	CCA4G	CCA3G	CCA2G	CCA1G	UGA
PWMB_EGR	7EFEE7H	BGB	TGB	COMGB	CCB8G	CCB7G	CCB6G	CCB5G	UGB
PWMC_EGR	7EF8C7H	BGC	TGC	COMGC	CCC4G	CCC3G	CCC2G	CCC1G	UGC
PWMD_EGR	7EF8E7H	BGD	TGD	COMGD	CCD8G	CCD7G	CCD6G	CCD5G	UGD
PWME_EGR	7EF6C7H	BGE	TGE	COMGE	CC4EG	CCE3G	CCE2G	CCE1G	UGE
PWMF_EGR	7EF6E7H	BGF	TGF	COMGF	CCF8G	CCF7G	CCF6G	CCF5G	UGF

BGn：产生刹车事件。该位由软件置 1，用于产生一个刹车事件，由硬件自动清 0（n=A/B/C/D/E/F）

0：无动作

1：产生一个刹车事件。此时 MOE=0、BIF=1，若开启对应的中断（BIE=1），则产生相应的中断

TGn：产生触发事件。该位由软件置 1，用于产生一个触发事件，由硬件自动清 0（n=A/B/C/D/E/F）

0：无动作

1：TIF=1，若开启对应的中断（TIE=1），则产生相应的中断

COMGn：捕获/比较事件，产生控制更新。该位由软件置 1，由硬件自动清 0（n=A/B/C/D/E/F）

0：无动作

1：CCPC=1，允许更新 CCIE、CCINE、CCiP，CCiNP，OCIM 位。

注：该位只对拥有互补输出的通道有效

CCnxG：产生捕获/比较 nx 事件。参考 CC1G 描述（n=A/B/C/D/E/F，x=1/2/3/4/5/6/7/8）

CC1G：产生捕获/比较 1 事件。产生捕获/比较 1 事件。该位由软件置 1，用于产生一个捕获/比较事件，由硬件自动清 0。

0：无动作；

1：在通道 CC1 上产生一个捕获/比较事件。

若通道 CC1 配置为输出：设置 CC1IF=1，若开启对应的中断，则产生相应的中断。

若通道 CC1 配置为输入：当前的计数器值被捕获至 PWMA_CCR1 寄存器，设置 CC1IF=1，若开启对应的中断，则产生相应的中断。若 CC1IF 已经为 1，则设置 CC1OF=1。

UGn：产生更新事件 该位由软件置 1，由硬件自动清 0。（n=A/B/C/D/E/F）

0：无动作；

1：重新初始化计数器，并产生一个更新事件。

注意预分频器的计数器也被清 0（但是预分频系数不变）。若在中心对称模式下或 DIR=0（向上计数）则计数器被清 0；若 DIR=1（向下计数）则计数器取 PWMn_ARR 的值。

28.10.11 捕获/比较模式寄存器 1（PWMx_CCMR1）

通道可用于输入（捕获模式）或输出（比较模式），通道的方向由相应的 CCnS 位定义。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能，ICxx 描述了通道在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

通道配置为比较输出模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR1	7EFEC8H	OCA1CE	OCA1M[2:0]			OCA1PE	OCA1FE	CCA1S[1:0]	
PWMB_CCMR1	7EFEE8H	OCB5CE	OCB5M[2:0]			OCB5PE	OCB5FE	CCB5S[1:0]	
PWMC_CCMR1	7EF8C8H	OCC1CE	OCC1M[2:0]			OCC1PE	OCC1FE	CCC1S[1:0]	
PWMD_CCMR1	7EF8E8H	OCD5CE	OCD5M[2:0]			OCD5PE	OCD5FE	CCD5S[1:0]	
PWME_CCMR1	7EF6C8H	OCE1CE	OCE1M[2:0]			OCE1PE	OCE1FE	CCE1S[1:0]	
PWMF_CCMR1	7EF6E8H	OCF5CE	OCF5M[2:0]			OCF5PE	OCF5FE	CCF5S[1:0]	

OCnxCE：输出比较 n 清零使能。该位用于使能使用 PWMETR 引脚上的外部事件来清通道 n 的输出信号（OCnREF）（n=A/B/C/D/E/F，x=1/5）

0：OCnREF 不受 ETRF 输入的影响；

1：一旦检测到 ETRF 输入高电平，OCnREF=0。

OCnxM[2:0]：输出比较 n 模式。该 3 位定义了输出参考信号 OCnREF 的动作，而 OCnREF 决定了 OCn 的值。OCnREF 是高电平有效，而 OCn 的有效电平取决于 CCnP 位。（n=A/B/C/D/E/F，x=1/5）

OCnM[2:0]	模式	说明
000	冻结	PWMn_CCR1与PWMn_CNT间的比较对OCnREF不起作用
001	匹配时设置通道n的输出为有效电平	当PWMn_CCR1=PWMn_CNT时，OCnREF输出高
010	匹配时设置通道n的输出为无效电平	当PWMn_CCR1=PWMn_CNT时，OCnREF输出低
011	翻转	当PWMn_CCR1=PWMn_CNT时，翻转OCnREF
100	强制为无效电平	强制OCnREF为低
101	强制为有效电平	强制OCnREF为高
110	PWM模式1	在向上计数时，当PWMn_CNT<PWMn_CCR1时 OCnREF输出高，否则OCnREF输出低 在向下计数时，当PWMn_CNT>PWMn_CCR1时 OCnREF输出低，否则OCnREF输出高
111	PWM模式2	在向上计数时，当PWMn_CNT<PWMn_CCR1时 OCnREF输出低，否则OCnREF输出高 在向下计数时，当PWMn_CNT>PWMn_CCR1时 OCnREF输出高，否则OCnREF输出低

注 1：一旦 LOCK 级别设为 3（PWMn_BKR 寄存器中的 LOCK 位）并且 CCnS=00（该通道配置成输出）则该位不能被修改。

注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OCnREF 电平才改变。

注 3：在有互补输出的通道上，这些位是预装载的。如果 PWMn_CR2 寄存器的 CCPC=1，OCM 位只有在 COM 事件发生时，才从预装载位取新值。

OCnxPE：输出比较 n 预装载使能（n=A/B/C/D/E/F，x=1/5）

0：禁止 PWMn_CCR1 寄存器的预装载功能，可随时写入 PWMn_CCR1 寄存器，并且新写入的数值立即起作用。

1：开启 PWMn_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，PWMn_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。

注 1：一旦 LOCK 级别设为 3（PWMn_BKR 寄存器中的 LOCK 位）并且 CCnS=00（该通道配置成输出）则该位不能被修改。

注 2：为了操作正确，在 PWM 模式下必须使能预装载功能。但在单脉冲模式下（PWMn_CR1 寄存器的 OPM=1），它不是必须的。

注：OCn1PE 必须在通道打开时（PWMn_CCER1 寄存器的 CCn1E=1）才是可写的。

注：OCn5PE 必须在通道打开时（PWMn_CCER1 寄存器的 CCn5E=1）才是可写的。

OCnxFE：输出比较 n 快速使能。该位用于加快 CC 输出对触发输入事件的响应。（n=A/B/C/D/E/F，x=1/5）

0：根据计数器与 CCRn 的值，CCn 正常操作，即使触发器是打开的。当触发器的输入有一个有效沿时，激活 CCn 输出的最小延时为 5 个时钟周期。

1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。OCFE 只在通道被配置成 PWMA 或 PWMB 模式时起作用。

注：OCn1FE 必须在通道打开时（PWMn_CCER1 寄存器的 CCn1E=1）才是可写的。

注：OCn5FE 必须在通道打开时（PWMn_CCER1 寄存器的 CCn5E=1）才是可写的。

CCn1S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=A/C/E）

CCn1S[1:0]	方向	输入脚
00	输出	
01	输入	ICn1映射在TIn1FP1上
10	输入	ICn1映射在TIn2FP1上
11	输入	ICn1映射在TRC上。此模式仅工作在内部触发器输入被选中时（由PWMn_SMCR寄存器的TSn位选择）

CCn5S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=B/D/F）

CCn5S[1:0]	方向	输入脚
00	输出	
01	输入	ICn5映射在TI5FP5上
10	输入	ICn5映射在TI6FP5上
11	输入	ICn5映射在TRC上。此模式仅工作在内部触发器输入被选中时（由PWMn_SMCR寄存器的TSn位选择）

注：CCn1S 仅在通道关闭时（PWMn_CCER1 寄存器的 CCn1E=0）才是可写的。

注：CCn5S 仅在通道关闭时（PWMn_CCER1 寄存器的 CCn5E=0）才是可写的。

通道配置为捕获输入模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR1	7EFEC8H	ICA1F[3:0]				ICA1PSC[1:0]		CCA1S[1:0]	
PWMB_CCMR1	7EFEE8H	ICB5F[3:0]				ICB5PSC[1:0]		CCB5S[1:0]	
PWMC_CCMR1	7EF8C8H	ICC1F[3:0]				ICC1PSC[1:0]		CCC1S[1:0]	
PWMD_CCMR1	7EF8E8H	ICD5F[3:0]				ICD5PSC[1:0]		CCD5S[1:0]	
PWME_CCMR1	7EF6C8H	ICE1F[3:0]				ICE1PSC[1:0]		CCE1S[1:0]	
PWMF_CCMR1	7EF6E8H	ICF5F[3:0]				ICF5PSC[1:0]		CCF5S[1:0]	

ICnxF[3:0]：输入捕获 n 滤波器选择，该位域定义了 TIn 的采样频率及数字滤波器长度。

（n=A/B/C/D/E/F，x=1/5）

ICnxF[3:0]	时钟数	ICnxF[3:0]	时钟数
0000	1	1000	48
0001	2	1001	64
0010	4	1010	80
0011	8	1011	96
0100	12	1100	128
0101	16	1101	160
0110	24	1110	192
0111	32	1111	256

注：即使对于带互补输出的通道，该位域也是非预装载的，并且不会考虑 CCPC（PWMn_CR2 寄存器）的值

ICnxPSC[1:0]：输入/捕获 n 预分频器。这两位定义了 CCn 输入（IC1）的预分频系数。

（n=A/B/C/D/E/F，x=1/5）

00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获

01：每 2 个事件触发一次捕获

10：每 4 个事件触发一次捕获

11：每 8 个事件触发一次捕获

注：ICn1PSC 必须在通道打开时（PWMn_CCER1 寄存器的 CCn1E=1）才是可写的。

注：ICn5PSC 必须在通道打开时（PWMn_CCER1 寄存器的 CCn5E=1）才是可写的。

CCn1S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=A/C/E）

CCn1S[1:0]	方向	输入脚
00	输出	
01	输入	ICn1映射在TIn1FP1上
10	输入	ICn1映射在TIn2FP1上
11	输入	ICn1映射在TRC上。此模式仅工作在内部触发器输入被选中时（由PWMn_SMCR寄存器的TSn位选择）

CCn5S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=B/D/F）

CCn5S[1:0]	方向	输入脚
00	输出	
01	输入	ICn5映射在TIn5FP5上
10	输入	ICn5映射在TIn6FP5上
11	输入	ICn5映射在TRC上。此模式仅工作在内部触发器输入被选中时（由PWMn_SMCR寄存器的TSn位选择）

注：CCn1S 仅在通道关闭时（PWMn_CCER1 寄存器的 CCn1E=0）才是可写的。

注：CCn5S 仅在通道关闭时（PWMn_CCER1 寄存器的 CCn5E=0）才是可写的。

28.10.12 捕获/比较模式寄存器 2（PWMx_CCMR2）

通道配置为比较输出模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR2	7EFEC9H	OCA2CE	OCA2M[2:0]			OCA2PE	OCA2FE	CCA2S[1:0]	
PWMB_CCMR2	7EFEE9H	OCB6CE	OCB6M[2:0]			OCB6PE	OCB6FE	CCB6S[1:0]	
PWMC_CCMR2	7EF8C9H	OCC2CE	OCC2M[2:0]			OCC2PE	OCC2FE	CCC2S[1:0]	
PWMD_CCMR2	7EF8E9H	OCD6CE	OCD6M[2:0]			OCD6PE	OCD6FE	CCD6S[1:0]	
PWME_CCMR2	7EF6C9H	OCE2CE	OCE2M[2:0]			OCE2PE	OCE2FE	CCE2S[1:0]	
PWMF_CCMR2	7EF6E9H	OCF6CE	OCF6M[2:0]			OCF6PE	OCF6FE	CCF6S[1:0]	

OCnxCE：输出比较 n 清零使能。该位用于使能使用 PWMETR 引脚上的外部事件来清通道 n 的输出信号（OCnREF）（n=A/B/C/D/E/F，x=2/6）

0：OCnREF 不受 ETRF 输入的影响；

1：一旦检测到 ETRF 输入高电平，OCnREF=0。

OCnxM[2:0]：输出比较 2 模式，参考 OCA1M。（n=A/B/C/D/E/F，x=2/6）

OCnxPE：输出比较 2 预装载使能，参考 OCA1PE。（n=A/B/C/D/E/F，x=2/6）

CCn2S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=A/C/E）

CCn2S[1:0]	方向	输入脚
00	输出	
01	输入	ICn2映射在TIn2FP2上
10	输入	ICn2映射在TIn1FP2上
11	输入	ICn2映射在TRC上。

CCn6S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=B/D/F）

CCn6S[1:0]	方向	输入脚
00	输出	
01	输入	ICn6映射在TIn6FP6上
10	输入	ICn6映射在TIn5FP6上
11	输入	ICn6映射在TRC上。

通道配置为捕获输入模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR2	7EFEC9H	ICA2F[3:0]				ICA2PSC[1:0]		CCA2S[1:0]	
PWMB_CCMR2	7EFEE9H	ICB6F[3:0]				ICB6PSC[1:0]		CCB6S[1:0]	
PWMC_CCMR2	7EF8C9H	ICC2F[3:0]				ICC2PSC[1:0]		CCC2S[1:0]	
PWMD_CCMR2	7EF8E9H	ICD6F[3:0]				ICD6PSC[1:0]		CCD6S[1:0]	
PWME_CCMR2	7EF6C9H	ICE2F[3:0]				ICE2PSC[1:0]		CCE2S[1:0]	
PWMF_CCMR2	7EF6E9H	ICF6F[3:0]				ICF6PSC[1:0]		CCF6S[1:0]	

ICnxF[3:0]：输入捕获 n 滤波器选择，参考 ICA1F。（n=A/B/C/D/E/F，x=2/6）

ICnxPSC[1:0]：输入/捕获 n 预分频器，参考 ICA1PSC。（n=A/B/C/D/E/F，x=2/6）

CCn2S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=A/C/E）

CCn2S[1:0]	方向	输入脚
00	输出	
01	输入	ICn2映射在TIn2FP2上
10	输入	ICn2映射在TIn1FP2上
11	输入	ICn2映射在TRC上。

CCn6S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=B/D/F）

CCn6S[1:0]	方向	输入脚
00	输出	
01	输入	ICn6映射在TIn6FP6上
10	输入	ICn6映射在TIn5FP6上
11	输入	ICn6映射在TRC上。

28.10.13 捕获/比较模式寄存器 3（PWMx_CCMR3）

通道配置为比较输出模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR3	7EFECAH	OCA3CE	OCA3M[2:0]			OCA3PE	OCA3FE	CCA3S[1:0]	
PWMB_CCMR3	7EFEEAH	OCB7CE	OCB7M[2:0]			OCB7PE	OCB7FE	CCB7S[1:0]	
PWMC_CCMR3	7EF8CAH	OCC3CE	OCC3M[2:0]			OCC3PE	OCC3FE	CCC3S[1:0]	
PWMD_CCMR3	7EF8EAH	OCD7CE	ODC7M[2:0]			OCD7PE	OCD7FE	CCD7S[1:0]	
PWME_CCMR3	7EF6CAH	OCE3CE	OCE3M[2:0]			OCE3PE	OCE3FE	CCE3S[1:0]	
PWMF_CCMR3	7EF6EAH	OCF7CE	OCF7M[2:0]			OCF7PE	OCF7FE	CCF7S[1:0]	

OCnxCE：输出比较 n 清零使能。该位用于使能使用 PWMETR 引脚上的外部事件来清通道 n 的输出信号（OCnREF）（n=A/B/C/D/E/F，x=3/7）

0：OCnREF 不受 ETRF 输入的影响；

1：一旦检测到 ETRF 输入高电平，OCnREF=0。

OCnxM[2:0]：输出比较 3 模式，参考 OCA1M。（n=A/B/C/D/E/F，x=3/7）

OCnxPE：输出比较 3 预装载使能，参考 OCA1PE。（n=A/B/C/D/E/F，x=3/7）

CCn3S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=A/C/E）

CCn3S[1:0]	方向	输入脚
00	输出	
01	输入	ICn3映射在TIn3FP3上
10	输入	ICn3映射在TIn4FP3上
11	输入	ICn3映射在TRC上。

CCn7S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=B/D/F）

CCn7S[1:0]	方向	输入脚
00	输出	
01	输入	ICn7映射在TIn7FP7上
10	输入	ICn7映射在TIn8FP7上
11	输入	ICn7映射在TRC上。

通道配置为捕获输入模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR3	7EFECAH	ICA3F[3:0]				ICA3PSC[1:0]		CCA3S[1:0]	
PWMB_CCMR3	7EFEEAH	ICB7F[3:0]				ICB7PSC[1:0]		CCB7S[1:0]	
PWMC_CCMR3	7EF8CAH	ICC3F[3:0]				ICC3PSC[1:0]		CCC3S[1:0]	
PWMD_CCMR3	7EF8EAH	ICD7F[3:0]				ICD7PSC[1:0]		CCD7S[1:0]	
PWME_CCMR3	7EF6CAH	ICE3F[3:0]				ICE3PSC[1:0]		CCE3S[1:0]	
PWMF_CCMR3	7EF6EAH	ICF7F[3:0]				ICF7PSC[1:0]		CCF7S[1:0]	

ICnxF[3:0]：输入捕获 n 滤波器选择，参考 ICA1F。（n=A/B/C/D/E/F，x=3/7）

ICnxPSC[1:0]：输入/捕获 n 预分频器，参考 ICA1PSC。（n=A/B/C/D/E/F，x=3/7）

CCn3S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=A/C/E）

CCn3S[1:0]	方向	输入脚
00	输出	
01	输入	ICn3映射在TIn3FP3上
10	输入	ICn3映射在TIn4FP3上
11	输入	ICn3映射在TRC上。

CCn7S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=B/D/F）

CCn7S[1:0]	方向	输入脚
00	输出	
01	输入	ICn7映射在TIn7FP7上
10	输入	ICn7映射在TIn8FP7上
11	输入	ICn7映射在TRC上。

28.10.14 捕获/比较模式寄存器 4（PWMx_CCMR4）

通道配置为比较输出模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR4	7EFECBH	OCA4CE	OCA4M[2:0]			OCA4PE	OCA4FE	CCA4S[1:0]	
PWMB_CCMR4	7EFEEBH	OCB8CE	OCB8M[2:0]			OCB8PE	OCB8FE	CCB8S[1:0]	
PWMC_CCMR4	7EF8CBH	OCC4CE	OCC4M[2:0]			OCC4PE	OCC4FE	CCC4S[1:0]	
PWMD_CCMR4	7EF8EBH	OCD8CE	OCD8M[2:0]			OCD8PE	OCD8FE	CCD8S[1:0]	
PWME_CCMR4	7EF6CBH	OCE4CE	OCE4M[2:0]			OCE4PE	OCE4FE	CCE4S[1:0]	
PWMF_CCMR4	7EF6EBH	OCF8CE	OCF8M[2:0]			OCF8PE	OCF8FE	CCF8S[1:0]	

OCnxCE：输出比较 n 清零使能。该位用于使能使用 PWMETR 引脚上的外部事件来清通道 n 的输出信号（OCnREF）（n=A/B/C/D/E/F，x=4/8）

0：OCnREF 不受 ETRF 输入的影响；

1：一旦检测到 ETRF 输入高电平，OCnREF=0。

OCnxM[2:0]：输出比较 n 模式，参考 OCA1M。（n=A/B/C/D/E/F，x=4/8）

OCnxPE：输出比较 n 预装载使能，参考 OCA1PE。（n=A/B/C/D/E/F，x=4/8）

CCn4S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=A/C/E）

CCn4S[1:0]	方向	输入脚
00	输出	
01	输入	ICn4映射在TIn4FP4上
10	输入	ICn4映射在TIn3FP4上
11	输入	ICn4映射在TRC上。

CCn8S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=B/D/F）

CCn8S[1:0]	方向	输入脚
00	输出	
01	输入	ICn8映射在TIn8FP8上
10	输入	ICn8映射在TIn7FP8上
11	输入	ICn8映射在TRC上。

通道配置为捕获输入模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR4	7EFECBH	ICA4F[3:0]				ICA4PSC[1:0]		CCA4S[1:0]	
PWMB_CCMR4	7EFEEBH	ICB8F[3:0]				ICB8PSC[1:0]		CCB8S[1:0]	
PWMC_CCMR4	7EF8CBH	ICD4F[3:0]				ICD4PSC[1:0]		CCD4S[1:0]	
PWMD_CCMR4	7EF8EBH	ICD8F[3:0]				ICD8PSC[1:0]		CCD8S[1:0]	
PWME_CCMR4	7EF6CBH	ICE4F[3:0]				ICE4PSC[1:0]		CCE4S[1:0]	
PWMF_CCMR4	7EF6EBH	ICF8F[3:0]				ICF8PSC[1:0]		CCF8S[1:0]	

ICnxF[3:0]：输入捕获 n 滤波器选择，参考 ICA1F。（n=A/B/C/D/E/F，x=4/8）

ICnxPSC[1:0]：输入/捕获 n 预分频器，参考 ICA1PSC。（n=A/B/C/D/E/F，x=4/8）

CCn4S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=A/C/E）

CCn4S[1:0]	方向	输入脚
00	输出	
01	输入	ICn4映射在TIn4FP4上
10	输入	ICn4映射在TIn3FP4上
11	输入	ICn4映射在TRC上。

CCn8S[1:0]：捕获/比较 n 选择。这两位定义通道的方向（输入/输出），及输入脚的选择（n=B/D/F）

CCn8S[1:0]	方向	输入脚
00	输出	
01	输入	ICn8映射在TIn8FP8上
10	输入	ICn8映射在TIn7FP8上
11	输入	ICn8映射在TRC上。

28.10.15 捕获/比较使能寄存器 1（PWMx_CCER1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCER1	7EFECCH	CCA2NP	CCA2NE	CCA2P	CCA2E	CCA1NP	CCA1NE	CCA1P	CCA1E
PWMB_CCER1	7EFEECH	-	-	CCB6P	CCB6E	-	-	CCB5P	CCB5E
PWMC_CCER1	7EF8CCH	CCC2NP	CCC2NE	CCC2P	CCC2E	CCC1NP	CCC1NE	CCC1P	CCC1E
PWMD_CCER1	7EF8ECH	-	-	CCD6P	CCD6E	-	-	CCD5P	CCD5E
PWME_CCER1	7EF6CCH	CCE2NP	CCE2NE	CCE2P	CCE2E	CCE1NP	CCE1NE	CCE1P	CCE1E
PWMF_CCER1	7EF6ECH	-	-	CCF6P	CCF6E	-	-	CCF5P	CCF5E

CCnxP：OCn 输入捕获/比较输出极性。参考 CCA1P（n=A/B/C/D/E/F，x=1/2/5/6）

CCnxE：OCn 输入捕获/比较输出使能。参考 CCA1E（n=A/B/C/D/E/F，x=1/2/5/6）

CCnxNP：OCnN（OCn 通道负极）比较输出极性。参考 CCA1NP（n=A/B/C/D/E/F，x=1/2/5/6）

CCnxNE：OCnN（OCn 通道负极）比较输出使能。参考 CCA1NE（n=A/B/C/D/E/F，x=1/2/5/6）

CCA1NP：OCA1N（OCA1 通道负极）比较输出极性

0：高电平有效；

1：低电平有效。

注 1：一旦 LOCK 级别（PWMA_BKR 寄存器中的 LOCK 位）设为 3 或 2 且 CC1S=00（通道配置为输出），则该位不能被修改。

注 2：对于有互补输出的通道，该位是预装载的。如果 CCPC=1（PWMA_CR2 寄存器），只有在 COM 事件发生时，CC1NP 位才从预装载位中取新值。

CCA1NE：OCA1N（OCA1 通道负极）比较输出使能

0：关闭比较输出。

1：开启比较输出，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。

注：对于有互补输出的通道，该位是预装载的。如果 CCPC=1（PWMA_CR2 寄存器），只有在 COM 事件发生时，CC1NE 位才从预装载位中取新值。

CCA1P：OCA1（OCA1 通道正极）输入捕获/比较输出极性

CC1 通道配置为输出：

0：高电平有效

1：低电平有效

CC1 通道配置为输入或者捕获：

0：捕获发生在 TI1F 或 TI2F 的上升沿；

1：捕获发生在 TI1F 或 TI2F 的下降沿.

注 1：一旦 LOCK 级别（PWMA_BKR 寄存器中的 LOCK 位）设为 3 或 2，则该位不能被修改。

注 2：对于有互补输出的通道，该位是预装载的。如果 CCPC=1（PWMA_CR2 寄存器），只有在 COM 事件发生时，CC1P 位才从预装载位中取新值。

CCA1E：OCA1（OCA1 通道正极）输入捕获/比较输出使能

0：关闭输入捕获/比较输出；

1：开启输入捕获/比较输出。

注：对于有互补输出的通道，该位是预装载的。如果 CCPC=1（PWMA_CR2 寄存器），只有在 COM 事件发生时，CC1E 位才从预装载位中取新值。

带刹车功能的互补输出通道 OCi 和 OCiN 的控制位

控制位					输出状态	
MOE	OSSI	OSSR	CCiE	CCiNE	OCi 输出状态	OCiN 输出状态
1	X	0	0	0	输出禁止	输出禁止
		0	0	1	输出禁止	带极性的 OCiREF
		0	1	0	带极性的 OCiREF	输出禁止
		0	1	1	带极性和死区的 OCiREF	带极性和死区的反向 OCiREF
		1	0	0	输出禁止	输出禁止
		1	0	1	关闭状态 （输出使能且为无效电平） OCi=CCiP	带极性的 OCiREF
		1	1	0	带极性的 OCiREF	关闭状态 （输出使能且为无效电平） OCiN=CCiNP
		1	1	1	带极性和死区的 OCiREF	带极性和死区的反向 OCiREF
0	0	X	X	X	输出禁止	
	1				关闭状态（输出使能且为无效电平）异步地：OCi=CCiP，OCiN=CCiNP；然后，若时钟存在：经过一个死区时间后 OCi=OISi，OCiN=OISiN，假设 OISi 与 OISiN 并不都对应 OCi 和 OCiN 的有效电平。	

注：管脚连接到互补的 OCi 和 OCiN 通道的外部 I/O 管脚的状态，取决于 OCi 和 OCiN 通道状态和 GPIO 寄存器。

28.10.16 捕获/比较使能寄存器 2（PWMx_CCER2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCER2	7EFECDH	CCA4NP	CCA4NE	CCA4P	CCA4E	CCA3NP	CCA3NE	CCA3P	CCA3E
PWMB_CCER2	7EFEEDH	-	-	CCB8P	CCB8E	-	-	CCB7P	CCB7E
PWMC_CCER2	7EF8CDH	CCC4NP	CCC4NE	CCC4P	CCC4E	CCC3NP	CCC3NE	CCC3P	CCC3E
PWMD_CCER2	7EF8EDH	-	-	CCD8P	CCD8E	-	-	CCD7P	CCD7E
PWME_CCER2	7EF6CDH	CCE4NP	CCE4NE	CCE4P	CCE4E	CCE3NP	CCE3NE	CCE3P	CCE3E
PWMF_CCER2	7EF6EDH	-	-	CCF8P	CCF8E	-	-	CCF7P	CCF7E

CCnxP：OCn 输入捕获/比较输出极性。参考 CCA1P（n=A/B/C/D/E/F，x=3/4/7/8）

CCnxE：OCn 输入捕获/比较输出使能。参考 CCA1E（n=A/B/C/D/E/F，x=3/4/7/8）

CCnxNP：OCnN（OCn 通道负极）比较输出极性。参考 CCA1NP（n=A/B/C/D/E/F，x=3/4/7/8）

CCnxNE：OCnN（OCn 通道负极）比较输出使能。参考 CCA1NE（n=A/B/C/D/E/F，x=3/4/7/8）

28.10.17 16 位计数器（PWMx_CNTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CNTRH	7EFECEH	CNTA[15:8]							
PWMA_CNTRL	7EFECFH	CNTA[7:0]							
PWMB_CNTRH	7EFEEEH	CNTB[15:8]							
PWMB_CNTRL	7EFEEFH	CNTB[7:0]							
PWMC_CNTRH	7EF8CEH	CNTC[15:8]							
PWMC_CNTRL	7EF8CFH	CNTC[7:0]							
PWMD_CNTRH	7EF8EEH	CNTD[15:8]							
PWMD_CNTRL	7EF8EFH	CNTD[7:0]							
PWME_CNTRH	7EF6CEH	CNTE[15:8]							
PWME_CNTRL	7EF6CFH	CNTE[7:0]							
PWMF_CNTRH	7EF6EEH	CNTF[15:8]							
PWMF_CNTRL	7EF6EFH	CNTF[7:0]							

CNTn[15:0]：计数器的 16 位值（n=A/B/C/D/E/F）

28.10.18 16 位预分频器（PWMx_PSCR），输出频率计算公式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_PSCRH	7EFED0H	PSCA[15:8]							
PWMA_PSCRL	7EFED1H	PSCA[7:0]							
PWMB_PSCRH	7EFEF0H	PSCB[15:8]							
PWMB_PSCRL	7EFEF1H	PSCB[7:0]							
PWMC_PSCRH	7EF8D0H	PSCC[15:8]							
PWMC_PSCRL	7EF8D1H	PSCC[7:0]							
PWMD_PSCRH	7EF8F0H	PSCD[15:8]							
PWMD_PSCRL	7EF8F1H	PSCD[7:0]							
PWME_PSCRH	7EF6D0H	PSCE[15:8]							
PWME_PSCRL	7EF6D1H	PSCE[7:0]							
PWMF_PSCRH	7EF6F0H	PSCF[15:8]							
PWMF_PSCRL	7EF6F1H	PSCF[7:0]							

PSCn[15:0]：预分频器的 16 位值。（n=A/B/C/D/E/F）

预分频器用于对 CK_PSC 进行分频。计数器的时钟频率（fCK_CNT）等于 fCK_PSC/（PSCR[15:0]+1）。PSCR 包含了当更新事件产生时装入当前预分频器寄存器的值（更新事件包括计数器被 TIM_EGR 的 UG 位清 0 或被工作在复位模式的从控制器清 0）。这意味着为了使新的值起作用，必须产生一个更新事件。

PWM 输出频率计算公式

PWMA 和 PWMB 两组 PWM 的输出频率计算公式相同，且每组可设置不同的频率。

对齐模式	PWM输出频率计算公式
边沿对齐	$\text{PWM输出频率} = \dfrac{\text{系统工作频率SYSclk}}{(\text{PWMx_PSCR} + 1) \times (\text{PWMx_ARR} + 1)}$
中间对齐	$\text{PWM输出频率} = \dfrac{\text{系统工作频率SYSclk}}{(\text{PWMx_PSCR} + 1) \times \text{PWMx_ARR} \times 2}$

28.10.19 16 位自动重装载寄存器（PWMx_ARR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_ARRH	7EFED2H	ARRA[15:8]							
PWMA_ARRL	7EFED3H	ARRA[7:0]							
PWMB_ARRH	7EFEF2H	ARRB[15:8]							
PWMB_ARRL	7EFEF3H	ARRB[7:0]							
PWMC_ARRH	7EF8D2H	ARRC[15:8]							
PWMC_ARRL	7EF8D3H	ARRC[7:0]							
PWMD_ARRH	7EF8F2H	ARRD[15:8]							
PWMD_ARRL	7EF8F3H	ARRD[7:0]							
PWME_ARRH	7EF6D2H	ARRE[15:8]							
PWME_ARRL	7EF6D3H	ARRE[7:0]							
PWMF_ARRH	7EF6F2H	ARRF[15:8]							
PWMF_ARRL	7EF6F3H	ARRF[7:0]							

ARRn[15:0]：自动重装载 16 位值（n=A/B/C/D/E/F）

ARR 包含了将要装载入实际的自动重装载寄存器的值。当自动重装载的值为 0 时，计数器不工作。

28.10.20 16 位重复计数器寄存器（PWMx_RCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_RCRH	7EFEB9H	REPA[15:8]							
PWMA_RCR	7EFED4H	REPA[7:0]							
PWMB_RCRH	7EFEBAH	REPB[15:8]							
PWMB_RCR	7EFEF4H	REPB[7:0]							
PWMC_RCRH	7EF8BAH	REPC[15:8]							
PWMC_RCR	7EF8F4H	REPC[7:0]							
PWMD_RCRH	7EF8BAH	REPD[15:8]							
PWMD_RCR	7EF8F4H	REPD[7:0]							
PWME_RCRH	7EF6BAH	REPE[15:8]							
PWME_RCR	7EF6F4H	REPE[7:0]							
PWMF_RCRH	7EF6BAH	REPF[15:8]							
PWMF_RCR	7EF6F4H	REPF[7:0]							

REPn[15:0]：重复计数器值（n=A/B/C/D/E/F）

开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率（即周期性地从预装载寄存器 传输到当前寄存器）；如果允许产生更新中断，则会同时影响产生更新中断的速率。每次向下计数器 REP_CNT 达到 0，会产生一个更新事件并且计数器 REP_CNT 重新从 REP 值开 始计数。由于 REP_CNT 只有在周期更新事件 U_RC 发生时才重载 REP 值，因此对 PWMn_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。这意味着在 PWM 模式中，（REP+1）对应着：

- 在边沿对齐模式下，PWM 周期的数目；
- 在中心对称模式下，PWM 半周期的数目。

28.10.21 16 位捕获/比较寄存器 1/5（PWMx_CCR1/5）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCR1H	7EFED5H	CCRA1[15:8]							
PWMA_CCR1L	7EFED6H	CCRA1[7:0]							
PWMB_CCR5H	7EFEF5H	CCRB5[15:8]							
PWMB_CCR5L	7EFEF6H	CCRB5[7:0]							
PWMC_CCR1H	7EF8D5H	CCRC1[15:8]							
PWMC_CCR1L	7EF8D6H	CCRC1[7:0]							
PWMD_CCR5H	7EF8F5H	CCRD5[15:8]							
PWMD_CCR5L	7EF8F6H	CCRD5[7:0]							
PWME_CCR1H	7EF6D5H	CCRE1[15:8]							
PWME_CCR1L	7EF6D6H	CCRE1[7:0]							
PWMF_CCR5H	7EF6F5H	CCRF5[15:8]							
PWMF_CCR5L	7EF6F6H	CCRF5[7:0]							

CCRnx[15:0]：捕获/比较 n 的 16 值（n=A/B/C/D/E/F，x=1/5）

若 CCn 通道配置为输出：CCRn 包含了装入当前比较值（预装载值）。如果在 PWMn_CCMR1 寄存器（OCnPE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 n 寄存器中。当前比较值同计数器 PWMn_CNT 的值相比较，并在 OCn 端口上产生输出信号。

若 CCn 通道配置为输入：CCRn 包含了上一次输入捕获事件发生时的计数器值（此时该寄存器为只读）。

28.10.22 16 位捕获/比较寄存器 2/6（PWMx_CCR2/6）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCR2H	7EFED7H	CCRA2[15:8]							
PWMA_CCR2L	7EFED8H	CCRA2[7:0]							
PWMB_CCR6H	7EFEF7H	CCRB6[15:8]							
PWMB_CCR6L	7EFEF8H	CCRB6[7:0]							
PWMC_CCR2H	7EF8D7H	CCRC2[15:8]							
PWMC_CCR2L	7EF8D8H	CCRC2[7:0]							
PWMD_CCR6H	7EF8F7H	CCRD6[15:8]							
PWMD_CCR6L	7EF8F8H	CCRD6[7:0]							
PWME_CCR2H	7EF6D7H	CCRE2[15:8]							
PWME_CCR2L	7EF6D8H	CCRE2[7:0]							
PWMF_CCR6H	7EF6F7H	CCRF6[15:8]							
PWMF_CCR6L	7EF6F8H	CCRF6[7:0]							

CCRnx[15:0]：捕获/比较 n 的 16 位值（n=A/B/C/D/E/F，x=2/6）

28.10.23 16 位捕获/比较寄存器 3/7（PWMx_CCR3/7）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCR3H	7EFED9H	CCRA3[15:8]							
PWMA_CCR3L	7EFEDAH	CCRA3[7:0]							
PWMB_CCR7H	7EFEF9H	CCRB7[15:8]							
PWMB_CCR7L	7EFEFAH	CCRB7[7:0]							
PWMC_CCR3H	7EF8D9H	CCRC3[15:8]							
PWMC_CCR3L	7EF8DAH	CCRC3[7:0]							
PWMD_CCR7H	7EF8F9H	CCRD7[15:8]							
PWMD_CCR7L	7EF8FAH	CCRD7[7:0]							
PWME_CCR3H	7EF6D9H	CCRE3[15:8]							
PWME_CCR3L	7EF6DAH	CCRE3[7:0]							
PWMF_CCR7H	7EF6F9H	CCRF7[15:8]							
PWMF_CCR7L	7EF6FAH	CCRF7[7:0]							

CCRnx[15:0]：捕获/比较 n 的 16 位值（n=A/B/C/D/E/F，x=3/7）

28.10.24 16 位捕获/比较寄存器 4/8（PWMx_CCR4/8）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCR4H	7EFEDBH	CCRA4[15:8]							
PWMA_CCR4L	7EFEDCH	CCRA4[7:0]							
PWMB_CCR8H	7EFEFBH	CCRB8[15:8]							
PWMB_CCR8L	7EFEFCH	CCRB8[7:0]							
PWMC_CCR4H	7EF8DBH	CCRC4[15:8]							
PWMC_CCR4L	7EF8DCH	CCRD4[7:0]							
PWMD_CCR8H	7EF8FBH	CCRD8[15:8]							
PWMD_CCR8L	7EF8FCH	CCRD8[7:0]							
PWME_CCR4H	7EF6DBH	CCRE4[15:8]							
PWME_CCR4L	7EF6DCH	CCRE4[7:0]							
PWMF_CCR8H	7EF6FBH	CCRF8[15:8]							
PWMF_CCR8L	7EF6FCH	CCRF8[7:0]							

CCRnx[15:0]：捕获/比较 n 的 16 位值（n=A/B/C/D/E/F，x=4/8）

28.10.25 刹车寄存器（PWMx_BKR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_BKR	7EFEDDH	MOEA	AOEA	BKPA	BKEA	OSSRA	OSSIA	LOCKA[1:0]	
PWMB_BKR	7EFEFDH	MOEB	AOEB	BKPB	BKEB	OSSRB	OSSIB	LOCKB[1:0]	
PWMC_BKR	7EF8DDH	MOEC	AOEC	BKPC	BKEC	OSSRC	OSSIC	LOCKC[1:0]	
PWMD_BKR	7EF8FDH	MOED	AOED	BKPD	BKED	OSSRD	OSSID	LOCKD[1:0]	
PWME_BKR	7EF6DDH	MOEE	AOEE	BKPE	BKEE	OSSRE	OSSIE	LOCKE[1:0]	
PWMF_BKR	7EF6FDH	MOEF	AOEF	BKPF	BKEF	OSSRF	OSSIF	LOCKF[1:0]	

MOEn：主输出使能。一旦刹车输入有效，该位被硬件异步清 0。根据 AOE 位的设置值，该位可以由软件置 1 或被自动置 1。它仅对配置为输出的通道有效。（n=A/B/C/D/E/F）

0：禁止 OC 和 OCN 输出或强制为空闲状态

1：如果设置了相应的使能位（PWMn_CCERX 寄存器的 CCIE 位），则使能 OC 和 OCN 输出。

注：一旦 LOCK 级别（PWMn_BKR 寄存器中的 LOCK 位）设为 1，则该位不能被修改

AOEn：自动输出使能（n=A/B/C/D/E/F）

0：MOE 只能被软件置 1；

1：MOE 能被软件置 1 或在下一个更新事件被自动置 1（如果刹车输入无效）。

注：一旦 LOCK 级别（PWMn_BKR 寄存器中的 LOCK 位）设为 1，则该位不能被修改

BKPn：刹车输入极性（n=A/B/C/D/E/F）

0：刹车输入低电平有效

1：刹车输入高电平有效

注：一旦 LOCK 级别（PWMn_BKR 寄存器中的 LOCK 位）设为 1，则该位不能被修改

BKEn：刹车功能使能（n=A/B/C/D/E/F）

0：禁止刹车输入（BRK）

1：开启刹车输入（BRK）

注：一旦 LOCK 级别（PWMn_BKR 寄存器中的 LOCK 位）设为 1，则该位不能被修改。

OSSRn：运行模式下“关闭状态”选择。该位在 MOE=1 且通道设为输出时有效（n=A/B/C/D/E/F）

0：当 PWM 不工作时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）；

1：当 PWM 不工作时，一旦 CCiE=1 或 CCiNE=1，首先开启 OC/OCN 并输出无效电平，然后置 OC/OCN 使能输出信号=1。

注：一旦 LOCK 级别（PWMn_BKR 寄存器中的 LOCK 位）设为 2，则该位不能被修改。

OSSIn：空闲模式下“关闭状态”选择。该位在 MOE=0 且通道设为输出时有效。（n=A/B/C/D/E/F）

0：当 PWM 不工作时，禁止 OC/OCN 输出（OC/OCN 使能输出信号=0）；

1：当 PWM 不工作时，一旦 CCiE=1 或 CCiNE=1，OC/OCN 首先输出其空闲电平，然后 OC/OCN 使能输出信号=1。

注：一旦 LOCK 级别（PWMn_BKR 寄存器中的 LOCK 位）设为 2，则该位不能被修改。

LOCKn[1:0]：锁定设置。该位为防止软件错误而提供的写保护措施（n=A/B/C/D/E/F）

LOCKn[1:0]	保护级别	保护内容
00	无保护	寄存器无写保护
01	锁定级别1	不能写入PWMn_BKR寄存器的MOE、BKE、BKP、AOE位和PWMn_OISR寄存器的OISI位
10	锁定级别2	不能写入锁定级别1中的各位， 也不能写入CC极性位以及OSSR/OSSI位
11	锁定级别3	不能写入锁定级别2中的各位， 也不能写入CC控制位

注：由于 MOE、BKE、BKP、AOE、OSSR、OSSI 位可被锁定（依赖于 LOCK 位），因此在第一次写 PWMn_BKR 寄存器时必须对它们进行设置。

28.10.26 死区寄存器（PWMx_DTR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_DTR	7EFEDEH	DTGA[7:0]							
PWMB_DTR	7EFEFEH	DTGB[7:0]							
PWMC_DTR	7EF8DEH	DTGC[7:0]							
PWMD_DTR	7EF8FEH	DTGD[7:0]							
PWME_DTR	7EF6DEH	DTGE[7:0]							
PWMF_DTR	7EF6FEH	DTGF[7:0]							

DTGn[7:0]：死区发生器设置。（n=A/B/C/D/E/F）

这些位定义了插入互补输出之间的死区持续时间。（t_{CK_PSC} 为 PWMn 的时钟脉冲）

DTGn[7:5]	死区时间
000	DTGn[7:0] * t_{CK_PSC}
001	
010	
011	
100	(64 + DTGn[6:0]) * 2 * t_{CK_PSC}
101	
110	(32 + DTGn[5:0]) * 8 * t_{CK_PSC}
111	(32 + DTGn[4:0]) * 16 * t_{CK_PSC}

注：一旦 LOCK 级别（PWMx_BKR 寄存器中的 LOCK 位）设为 1、2 或 3 时，则该位不能被修改。

28.10.27 输出空闲状态寄存器（PWMx_OISR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_OISR	7EFEDFH	OISA4N	OISA4	OISA3N	OISA3	OISA2N	OISA2	OISA1N	OISA1
PWMB_OISR	7EFEFFH	-	OISB8	-	OISB7	-	OISB6	-	OISB5
PWMC_OISR	7EF8DFH	OISC4N	OISC4	OISC3N	OISC3	OISC2N	OISC2	OISC1N	OISC1
PWMD_OISR	7EF8FFH	-	OISD8	-	OISD7	-	OISD6	-	OISD5
PWME_OISR	7EF6DFH	OISE4N	OISE4	OISE3N	OISE3	OISE2N	OISE2	OISE1N	OISE1
PWMF_OISR	7EF6FFH	-	OISF8	-	OISF7	-	OISF6	-	OISF5

OISnx：空闲状态时 OCn 输出电平，参考 OISA1（n=A/B/C/D/E/F，x=1/2/3/4/5/6/7/8）

OISnxN：空闲状态时 OCnN 输出电平，参考 OISA1N（n=A/B/C/D/E/F，x=1/2/3/4/5/6/7/8）

OISA1N：空闲状态时 OC1N 输出电平

0：当 MOE=0 时，则在一个死区时间后，OC1N=0；

1：当 MOE=0 时，则在一个死区时间后，OC1N=1。

注：一旦 LOCK 级别（PWMx_BKR 寄存器中的 LOCK 位）设为 1、2 或 3 时，则该位不能被修改。

OISA1：空闲状态时 OC1 输出电平

0：当 MOE=0 时，如果 OC1N 使能，则在一个死区后，OC1=0；

1：当 MOE=0 时，如果 OC1N 使能，则在一个死区后，OC1=1。

注：一旦 LOCK 级别（PWMx_BKR 寄存器中的 LOCK 位）设为 1、2 或 3 时，则该位不能被修改。

29 高级 PWM-硬件移相

产品线	高级 PWM 新增硬件移相功能
STC32G144K246 系列	**3 组**

29.1 高级 PWM 硬件移相功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_PS2	7EFEB8H	-		-		AC6PS[1:0]		AC5PS[1:0]	

AC5PS[1:0]：高级 PWMA 通道 5 输出脚选择位

AC5PS[1:0]	PWMA5
00	P4.6
01	P0.4
10	P7.2
11	P1.3

AC6PS[1:0]：高级 PWMA 通道 6 输出脚选择位

AC6PS[1:0]	PWMA6
00	P4.7
01	P0.5
10	P7.3
11	P1.5

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMC_PS2	7EF8B8H	-		-		CC6PS[1:0]		CC5PS[1:0]	

CC5PS[1:0]：高级 PWMC 通道 5 输出脚选择位

CC5PS[1:0]	PWMC5
00	P1.1
01	P3.6
10	P9.6
11	P6.3

CC6PS[1:0]：高级 PWMC 通道 6 输出脚选择位

CC6PS[1:0]	PWMC6
00	P1.7
01	P3.7
10	P9.7
11	P6.5

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWME_PS2	7EF6B8H	-		-		EC6PS[1:0]		EC5PS[1:0]	

EC5PS[1:0]：高级 PWME 通道 5 输出脚选择位

EC5PS[1:0]	PWME5
00	P5.0
01	P4.3
10	PB.4
11	P9.5

EC6PS[1:0]：高级 PWME 通道 6 输出脚选择位

EC6PS[1:0]	PWME6
00	P5.1
01	P4.4
10	PB.5
11	P9.7

29.2 寄存器描述

29.2.1 输出使能寄存器 2（PWMn_ENO2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_ENO2	7EF930H	-	-	-	-	-	ENOA6	-	ENOA5
PWMC_ENO2	7EF950H	-	-	-	-	-	ENOC6	-	ENOC5
PWME_ENO2	7EF690H	-	-	-	-	-	ENOE6	-	ENOE5

ENOn5：PWMn5 输出控制位（n=A/C/E）

0：禁止 PWMn5 输出

1：使能 PWMn5 输出

ENOn6：PWMn6 输出控制位（n=A/C/E）

0：禁止 PWMn6 输出

1：使能 PWMn6 输出

29.2.2 输出附加使能寄存器 2（PWMn_IOAUX2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_IOAUX2	7EF931H	-	-	-	-	-	AUXA6	-	AUXA5
PWMC_IOAUX2	7EF951H	-	-	-	-	-	AUXC6	-	AUXC5
PWMAE_IOAUX2	7EF691H	-	-	-	-	-	AUXE6	-	AUXE5

AUXn5：PWMn5 输出控制位（n=A/C/E）

0：PWMn5 的输出直接由 ENOn5 控制

1：PWMn5 的输出由 ENOn5 和 PWMn_BKR 共同控制

AUXn6：PWMn6 输出控制位（n=A/C/E）

0：PWMn6 的输出直接由 ENOn6 控制

1：PWMn6 的输出由 ENOn6 和 PWMn_BKR 共同控制

29.2.3 控制寄存器 3（PWMn_CR3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CR3	7EF932H	MMSA[3:0]				-	OISA6	-	OISA5
PWMC_CR3	7EF952H	MMSC[3:0]				-	OISC6	-	OISC5
PWME_CR3	7EF692H	MMSE[3:0]				-	OISE6	-	OISE5

MMSn[3:0]：主模式选择 2（n=A/C/E）

MMSn[3:0]	模式	说明
0000	复位	PWMn_EGR寄存器的UG位被用于作为触发输出（TRGO2）。如果复位由触发输入生成（从模式控制器配置为复位模式），则TRGO2上的信号相比实际复位会有延迟。
0001	使能	计数器使能信号CNT_EN用作触发输出(TRGO2)。该触发输出可用于同时启动多个定时器，或者控制在一段时间内使能从定时器。计数器使能信号由CEN控制位与门控模式下的触发输入的逻辑或运算组合而成。当计数器使能信号由触发输入控制时，TRGO2上会存在延迟，选择主/从模式时除外（请参见PWMn_SMCR寄存器中MSMn位的说明）。
0010	更新	更新事件被选为触发输出（TRGO2）
0011	比较脉冲	CCn1IF 标志置1时（即使已为高），只要发生捕获或比较匹配，触发输出(TRGO2)都会发送一个正脉冲。
0100	比较	OCn1REF信号用作触发输出（TRGO2）
0101	比较	OCn2REF信号用作触发输出（TRGO2）
0110	比较	OCn3REF信号用作触发输出（TRGO2）
0111	比较	OCn4REF信号用作触发输出（TRGO2）
1000	比较	OCn5REF信号用作触发输出（TRGO2）
1001	比较	OCn6REF信号用作触发输出（TRGO2）
1010	比较脉冲	OCn4REF 上升沿或下降沿时，TRGO2 上生成脉冲
1011	比较脉冲	OCn6REF 上升沿或下降沿时，TRGO2 上生成脉冲
1100	比较脉冲	OCn4REF 或 OCn6REF 上升沿时，TRGO2 上生成脉冲
1101	比较脉冲	OCn4REF 上升沿或 OCn6REF 下降沿时，TRGO2 上生成脉冲
1110	比较脉冲	OCn5REF 或 OCn6REF 上升沿时，TRGO2 上生成脉冲
1111	比较脉冲	OC5REF 上升沿或 OC6REF 下降沿时，TRGO2 上生成脉冲

注：必须先使能从定时器或 ADC 的时钟，才能从主定时器接收事件；并且从主定时器接收触发信号时，不得实时更改从定时器或 ADC 的时钟。

OISn6：空闲状态时 OCn6 输出电平（请参见 OISA1 位）（n=A/C/E）

OISn5：空闲状态时 OCn5 输出电平（请参见 OISA1 位）（n=A/C/E）

29.2.4 状态寄存器 3(PWMn_SR3)

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_SR1	7EF933H	-	-	-	-	-	-	CCA6IF	CCA5IF
PWMC_SR1	7EF953H	-	-	-	-	-	-	CCC6IF	CCC5IF
PWME_SR1	7EF693H	-	-	-	-	-	-	CCE6IF	CCE5IF

CCn6IF：比较n中断标记，参考CCA1IF描述（注意：通道6只能配置为输出）（n=A/C/E）

CCn5IF：比较n中断标记，参考CCA1IF描述（注意：通道5只能配置为输出）（n=A/C/E）

29.2.5 捕获/比较使能寄存器 3（PWMn_CCER3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCER3	7EF934H	-	-	CCA6P	CCA6E	-	-	CCA5P	CCA5E
PWMC_CCER3	7EF954H	-	-	CCC6P	CCC6E	-	-	CCC5P	CCC5E
PWME_CCER3	7EF694H	-	-	CCE6P	CCE6E	-	-	CCE5P	CCE5E

CCn6P：OCn6 输入捕获/比较输出极性。参考 CCA1P（n=A/C/E）

CCn6E：OCn6 输入捕获/比较输出使能。参考 CCA1E（n=A/C/E）

CCn5P：OCn5 输入捕获/比较输出极性。参考 CCA1P（n=A/C/E）

CCn5E：OCn5 输入捕获/比较输出使能。参考 CCA1E（n=A/C/E）

29.2.6 捕获/比较模式扩展寄存器 1（PWMn_CCMR1X）

通道可用于捕获输入模式或比较输出模式，通道的方向由相应的 CCnS 位定义。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能，ICxx 描述了通道在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

通道配置为比较输出模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR1X	7EF938H	-	-	-	-	-	-	-	OCA1M[3]
PWMC_CCMR1X	7EF958H	-	-	-	-	-	-	-	OCC1M[3]
PWME_CCMR1X	7EF698H	-	-	-	-	-	-	-	OCE1M[3]

OCn1M[3]：和 PWMn_CCMR1 中的 OCn1M[2:0]合并为 4 位对输出模式进行扩展（n=A/C/E）

OCnM[3:0]	模式	说明
0000	冻结	PWMn_CCR1与PWMn_CNT间的比较对OCnREF不起作用
0001	匹配时设置通道n的输出为有效电平	当PWMn_CCR1=PWMn_CNT时，OCnREF输出高
0010	匹配时设置通道n的输出为无效电平	当PWMn_CCR1=PWMn_CNT时，OCnREF输出低
0011	翻转	当PWMn_CCR1=PWMn_CNT时，翻转OCnREF
0100	强制为无效电平	强制OCnREF为低
0101	强制为有效电平	强制OCnREF为高
0110	PWM模式1	在向上计数时，当PWMn_CNT<PWMn_CCR1时OCnREF输出高，否则OCnREF输出低 在向下计数时，当PWMn_CNT>PWMn_CCR1时OCnREF输出低，否则OCnREF输出高
0111	PWM模式2	在向上计数时，当PWMn_CNT<PWMn_CCR1时OCnREF输出低，否则OCnREF输出高 在向下计数时，当PWMn_CNT>PWMn_CCR1时OCnREF输出高，否则OCnREF输出低
1000	可再触发OPM模式1	在递增计数模式下，通道为有效状态，直至（在TRGI信号上）检测到触发事件。然后，在PWM模式1下进行比较，通道会在下一次更新时再次变为有效状态。在递减计数模式下，通道为无效状态，直至（在 TRGI 信号上）检测到触发事件。然后，在PWM模式1下进行比较，通道会在下一次更新时再次变为无效状态。
1001	可再触发OPM模式2	在递增计数模式下，通道为无效状态，直至（在TRGI信号上）检测到触发事件。然后，在PWM模式2下进行比较，通道会在下一次更新时再次变为无效状态。在递减计数模

OCnM[3:0]	模式	说明
		式下，通道为有效状态，直至（在TRGI信号上）检测到触发事件。然后，在PWM模式2下进行比较，通道会在下一次更新时再次变为有效状态。
1010	保留	
1011	保留	
1100	组合PWM模式1	OC1REF与在PWM模式1下的行为相同。OC1REFC是OC1REF 和OC2REF的逻辑或运算结果。
1101	组合PWM模式2	OC1REF与在PWM模式2下的行为相同。OC1REFC是OC1REF 和OC2REF的逻辑与运算结果。
1110	不对称PWM模式1	OC1REF与在PWM模式1下的行为相同。计数器递增计数时，OC1REFC输出OC1REF；计数器递减计数时，OC1REFC 输出OC2REF。
1111	不对称PWM模式2	OC1REF与在PWM模式2下的行为相同。计数器递增计数时，OC1REFC输出OC1REF；计数器递减计数时，OC1REFC 输出OC2REF。

注 1：一旦 LOCK 级别设为 3（PWMn_BKR 寄存器中的 LOCK 位）并且 CCnS=00（该通道配置成输出）则该位不能被修改。

注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OCnREF 电平才改变。

注 3：在有互补输出的通道上，这些位是预装载的。如果 PWMn_CR2 寄存器的 CCPC=1，OCM 位只有在 COM 事件发生时，才从预装载位取新值。

29.2.7 捕获/比较模式扩展寄存器 2（PWMn_CCMR2X）

通道配置为比较输出模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR2X	7EF939H	-	-	-	-	-	-	-	OCA2M[3]
PWMC_CCMR2X	7EF959H	-	-	-	-	-	-	-	OCC2M[3]
PWME_CCMR2X	7EF699H	-	-	-	-	-	-	-	OCE2M[3]

OCn2M[3]：和 PWMn_CCMR2 中的 OCn2M[2:0]合并为 4 位对输出模式进行扩展，详情参考 OCA1M。（n=A/C/E）

29.2.8 捕获/比较模式扩展寄存器 3（PWMn_CCMR3X）

通道配置为比较输出模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR3X	7EF93AH	-	-	-	-	-	-	-	OCA3M[3]
PWMC_CCMR3X	7EF95AH	-	-	-	-	-	-	-	OCC3M[3]
PWME_CCMR3X	7EF69AH	-	-	-	-	-	-	-	OCE3M[3]

OCn3M[3]：和 PWMn_CCMR3 中的 OCn3M[2:0]合并为 4 位对输出模式进行扩展，详情参考 OCA1M。（n=A/C/E）

29.2.9 捕获/比较模式扩展寄存器 4（PWMn_CCMR4X）

通道配置为比较输出模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR4X	7EF93BH	-	-	-	-	-	-	-	OCA4M[3]
PWMC_CCMR4X	7EF95BH	-	-	-	-	-	-	-	OCC4M[3]
PWME_CCMR4X	7EF969BH	-	-	-	-	-	-	-	OCE4M[3]

OCn4M[3]：和 PWMn_CCMR4 中的 OCn4M[2:0]合并为 4 位对输出模式进行扩展，详情参考 OCA1M。（n=A/C/E）

29.2.10 捕获/比较模式寄存器 5（PWMn_CCMR5）

通道配置为比较输出模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR5	7EF93CH	OCA5CE	OCA5M[2:0]			OCA5PE	-	-	-
PWMC_CCMR5	7EF95CH	OCC5CE	OCC5M[2:0]			OCC5PE	-	-	-
PWME_CCMR5	7EF69CH	OCE5CE	OCE5M[2:0]			OCE5PE	-	-	-

OCn5CE：输出比较 5 清零使能。该位用于使能使用 PWMnETR 引脚上的外部事件来清通道 5 的输出信号（OCn5REF）（n=A/C/E）

0：OCn5REF 不受 ETRF 输入的影响；

1：一旦检测到 ETRF 输入高电平，OCn5REF=0。

OCn5M[2:0]：输出比较 5 模式，参考 OCA1M。（n=A/C/E）

OCn5PE：输出比较 5 预装载使能，参考 OCA1PE。（n=A/C/E）

29.2.11 捕获/比较模式扩展寄存器 5（PWMn_CCMR5X）

通道配置为比较输出模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR5X	7EF93DH	-	-	-	-	-	-	-	OCA5M[3]
PWMC_CCMR5X	7EF95DH	-	-	-	-	-	-	-	OCC5M[3]
PWME_CCMR5X	7EF69DH	-	-	-	-	-	-	-	OCE5M[3]

OCn5M[3]：和 PWMn_CCMR5 中的 OCn5M[2:0]合并为 4 位对输出模式进行扩展，详情参考 OCA1M。（n=A/C/E）

29.2.12 捕获/比较模式寄存器 6（PWMn_CCMR6）

通道配置为比较输出模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR6	7EF93EH	OCA6CE	OCA6M[2:0]			OCA6PE	-	-	-
PWMC_CCMR6	7EF95EH	OCC6CE	OCC6M[2:0]			OCC6PE	-	-	-
PWME_CCMR6	7EF69EH	OCE6CE	OCE6M[2:0]			OCE6PE	-	-	-

OCn6CE：输出比较 6 清零使能。该位用于使能使用 PWMnETR 引脚上的外部事件来清通道 6 的输出信号（OCn6REF）（n=A/C/E）

0：OCn6REF 不受 ETRF 输入的影响；

1：一旦检测到 ETRF 输入高电平，OCn6REF=0。

OCn6M[2:0]：输出比较 6 模式，参考 OCA1M。（n=A/C/E）

OCn6PE：输出比较 6 预装载使能，参考 OCA1PE。（n=A/C/E）

29.2.13 捕获/比较模式扩展寄存器 6（PWMn_CCMR6X）

通道配置为比较输出模式

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCMR6X	7EF93FH	-	-	-	-	-	-	-	OCA6M[3]
PWMC_CCMR6X	7EF95FH	-	-	-	-	-	-	-	OCC6M[3]
PWME_CCMR6X	7EF69FH	-	-	-	-	-	-	-	OCE6M[3]

OCn6M[3]：和 PWMn_CCMR6 中的 OCn6M[2:0]合并为 4 位对输出模式进行扩展，详情参考 OCA1M。（n=A/C/E）

29.2.14 16 位捕获/比较寄存器 5（PWMn_CCR5）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCR5H	7EF940H	CCRA5[15:8]							
PWMA_CCR5L	7EF941H	CCRA5[7:0]							
PWMC_CCR5H	7EF960H	CCRC5[15:8]							
PWMC_CCR5L	7EF961H	CCRC5[7:0]							
PWME_CCR5H	7EF6A0H	CCRE5[15:8]							
PWME_CCR5L	7EF6A1H	CCRE5[7:0]							

CCRn5[15:0]：捕获/比较 5 的 16 位值（n=A/C/E）

CCRn5 包含了装入当前比较值（预装载值）。如果在 PWMn_CCMR5 寄存器（OCn5PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 5 寄存器中。当前比较值同计数器 PWMn_CNT 的值相比较，并在 OCn5 端口上产生输出信号。

29.2.15 捕获/比较扩展寄存器 5（PWMn_CCR5X）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCR5X	7EF942H	GCA5C3	GCA5C2	GCA5C1	-	-	-	-	-
PWMC_CCR5X	7EF962H	GCC5C3	GCC5C2	GCC5C1	-	-	-	-	-
PWME_CCR5X	7EF6A2H	GCE5C3	GCE5C2	GCE5C1	-	-	-	-	-

GCn5C3：PWMn 的通道 5 和通道 3 组（Group Channel 5 and Channel 3）。（n=A/C/E）

通道 3 输出上失真：

0：OCn5REF 对 OCn3REFC 无影响；

1：OCn3REFC 是 OCn3REFC 和 OCn5REF 的逻辑与运算结果。该位可以立即生效，也可预装载并在更新事件后执行（如果在 PWMn_CCMR3 中选择了预装载功能）。

注：也可在组合 PWM 信号上应用此失真。

GCn5C2：PWMn 的通道 5 和通道 2 组（Group Channel 5 and Channel 2）。（n=A/C/E）

通道 2 输出上失真：

0：OCn5REF 对 OCn2REFC 无影响；

1：OCn2REFC 是 OCn2REFC 和 OCn5REF 的逻辑与运算结果。该位可以立即生效，也可预装载并在更新事件后执行（如果在 PWMn_CCMR2 中选择了预装载功能）。

注：也可在组合 PWM 信号上应用此失真。

GCn5C1：PWMn 的通道 5 和通道 1 组（Group Channel 5 and Channel 1）。（n=A/C/E）

通道 1 输出上失真：

0：OCn5REF 对 OCn1REFC 无影响；

1：OCn1REFC 是 OCn1REFC 和 OCn5REF 的逻辑与运算结果。该位可以立即生效，也可预装载并在更新事件后执行（如果在 PWMn_CCMR1 中选择了预装载功能）。

注：也可在组合 PWM 信号上应用此失真。

29.2.16 16 位捕获/比较寄存器 6（PWMn_CCR6）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCR6H	7EF943H	CCRA6[15:8]							
PWMA_CCR6L	7EF944H	CCRA6[7:0]							
PWMC_CCR6H	7EF963H	CCRC6[15:8]							
PWMC_CCR6L	7EF964H	CCRC6[7:0]							
PWME_CCR6H	7EF6A3H	CCRE6[15:8]							
PWME_CCR6L	7EF6A4H	CCRE6[7:0]							

CCRn6[15:0]：捕获/比较 n 的 16 位值（n=A/C/E）

CCRn6 包含了装入当前比较值（预装载值）。如果在 PWMn_CCMR6 寄存器（OCn6PE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 6 寄存器中。当前比较值同计数器 PWMn_CNT 的值相比较，并在 OCn6 端口上产生输出信号。

29.3 移相 PWM 输出模式

29.3.1 不对称 PWM 模式

在不对称模式下，生成的两个中心对齐 PWM 信号间允许存在可编程相移。频率由 PWMA_ARR 寄存器的值确定，而占空比和相移则由一对 PWMA_CCRx 寄存器确定。两个寄存器分别控制递增计数和递减计数期间的 PWM，这样每半个 PWM 周期便会调节一次 PWM：

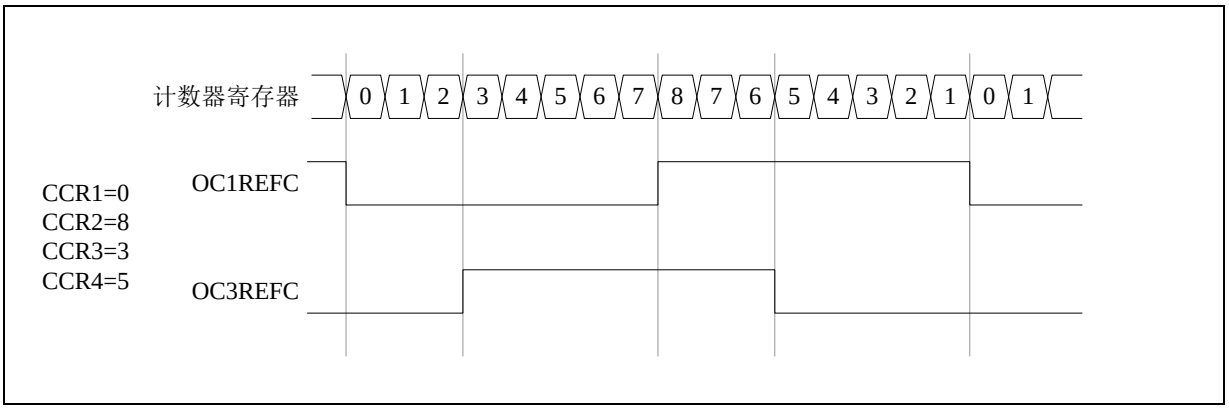
- OC1REFC（或 OC2REFC）由 PWMA_CCR1 和 PWMA_CCR2 控制
- OC3REFC（或 OC4REFC）由 PWMA_CCR3 和 PWMA_CCR4 控制

两个通道可以独立选择不对称 PWM 模式（每对 CCR 寄存器控制一个 OCx 输出），只需向 PWMA_CCMRx 寄存器的 OCxM 位写入“1110”（不对称 PWM 模式 1）或“1111”（不对称 PWM 模式 2）。
给定通道用作不对称 PWM 通道时，也可使用其互补通道。例如，如果通道 1 上产生 OC1REFC 信号（不对称 PWM 模式 1），则由于不对称 PWM 模式 1 的原因，通道 2 上可输出 OC2REF 信号或 OC2REFC 信号。

（注：出于兼容性原因，OCxM[3:0]位域分为两部分，最高有效位与最低有效的 3 位不相邻。）

下图显示了不对称 PWM 模式下可以产生的信号示例（通道 1 到通道 4 在不对称 PWM 模式 1 下配置）。与死区发生器配合使用时，这可控制相移全桥直流到直流转换器。

（产生 2 个 50%占空比的相移 PWM 信号）



29.3.2 组合 PWM 模式

在组合 PWM 模式下，生成的两个边沿或中心对齐 PWM 信号的各个脉冲间允许存在可编程延时和相移。频率由 PWMA_ARR 寄存器的值确定，而占空比和延时则由两个 PWMA_CCRx 寄存器确定。产生的信号 OCxREFC 由两个参考 PWM 的逻辑或运算或者逻辑与运算组合组成。

- OC1REFC（或 OC2REFC）由 PWMA_CCR1 和 PWMA_CCR2 控制
- OC3REFC（或 OC4REFC）由 PWMA_CCR3 和 PWMA_CCR4 控制

两个通道可以独立选择组合 PWM 模式（每对 CCR 寄存器控制一个 OCx 输出），只需向 PWMA_CCMRx 寄存器的 OCxM 位写入“1100”（组合 PWM 模式 1）或“1101”（组合 PWM 模式 2）。

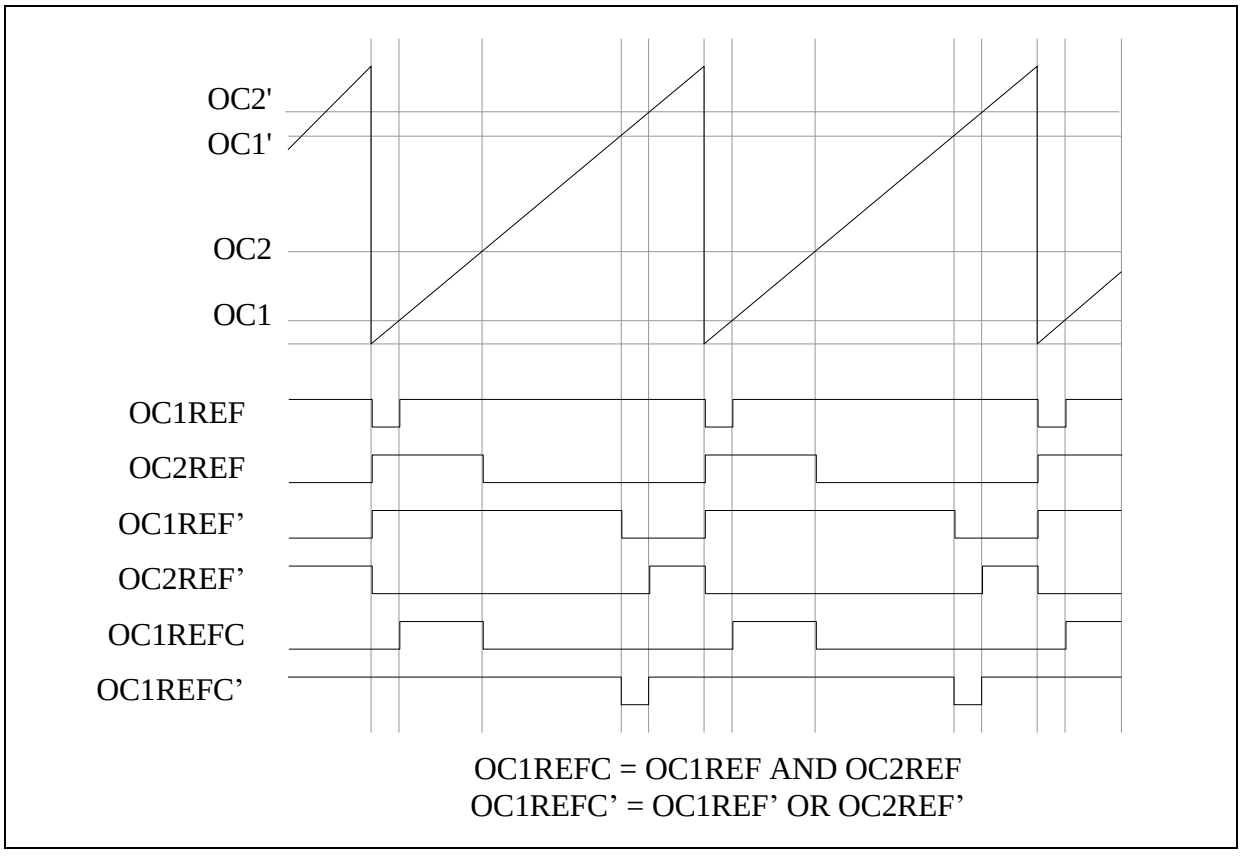
当给定通道用作组合 PWM 通道时，其互补通道必须在相反的 PWM 模式下配置。例如，一个通道在组合 PWM 模式 1 下配置，另一个通道在组合 PWM 模式 2 下配置。

（注：出于兼容性原因，OCxM[3:0]位域分为两部分，最高有效位与最低有效的 3 位不相邻。）

下图显示了不对称 PWM 模式下可以产生的信号示例，通过以下配置可获得这些信号：

- 通道 1 在组合 PWM 模式 2 下配置。
- 通道 2 在 PWM 模式 1 下配置。
- 通道 3 在组合 PWM 模式 2 下配置。
- 通道 4 在 PWM 模式 1 下配置。

（通道 1 和通道 3 上的组合 PWM 模式）



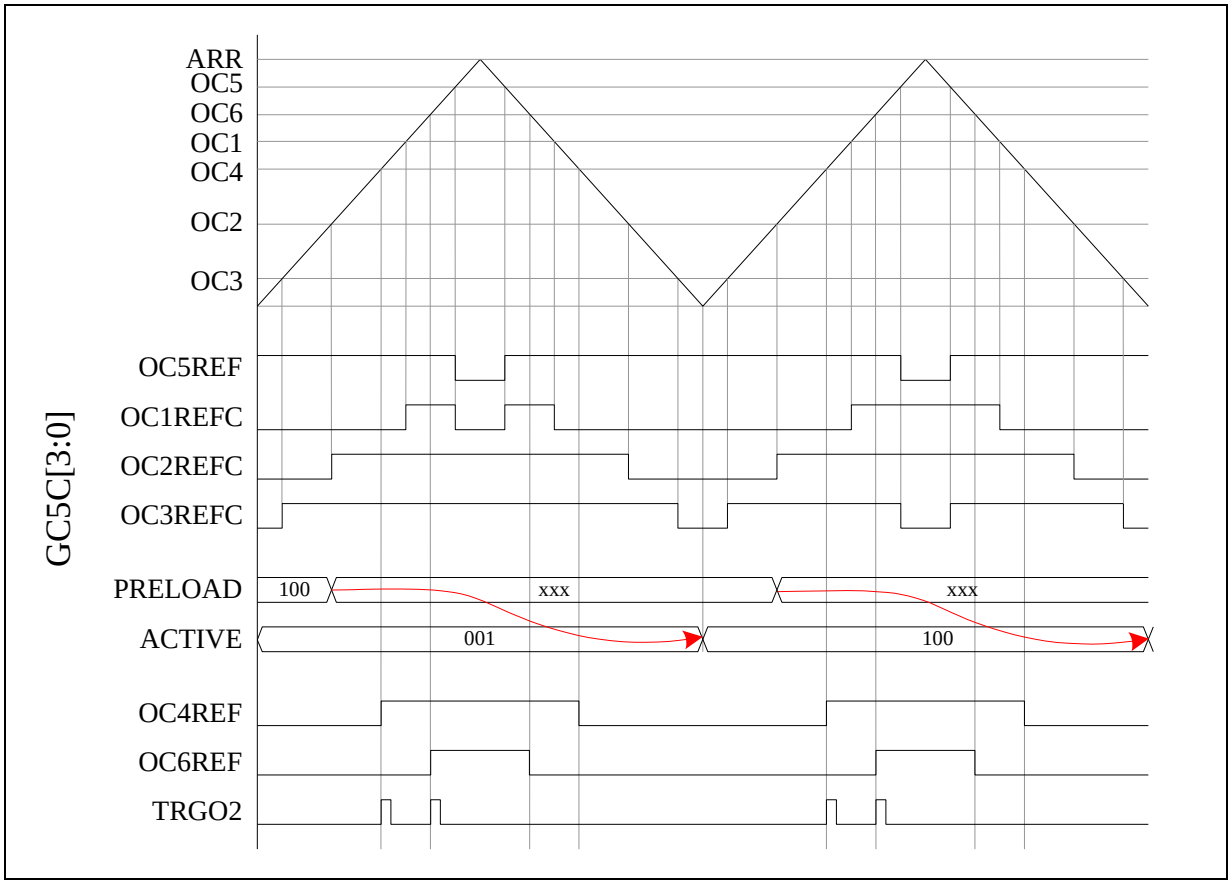
29.3.3 组合三相 PWM 模式

在组合三相 PWM 模式下，产生的一至三个中心对齐 PWM 信号与一个可编程信号间允许在脉冲中间进行逻辑与运算。OC5REF 信号用于定义产生的组合信号。凭借 PWMA_CCR5 中的 3 位 GC5C[3:1]，可以选择 OC5REF 与哪个参考信号组合。产生的信号 OCxREFC 由两个参考 PWM 的逻辑与运算组合组成。

- 如果 GC5C1 置 1，则 OC1REFC 由 PWMA_CCR1 和 PWMA_CCR5 控制
- 如果 GC5C2 置 1，则 OC2REFC 由 PWMA_CCR2 和 PWMA_CCR5 控制
- 如果 GC5C3 置 1，则 OC3REFC 由 PWMA_CCR3 和 PWMA_CCR5 控制

通道 1 到通道 3 可独立选择组合三相 PWM 模式，只需将 3 位 GC5C[3:1]中的至少一位置 1。

（三相组合 PWM 信号，每个周期多个触发脉冲）



（TRGO2 波形说明了如何根据给定的三相 PWM 信号同步 ADC）

29.4 利用不对称 PWM 实现高速正交编码信号输出

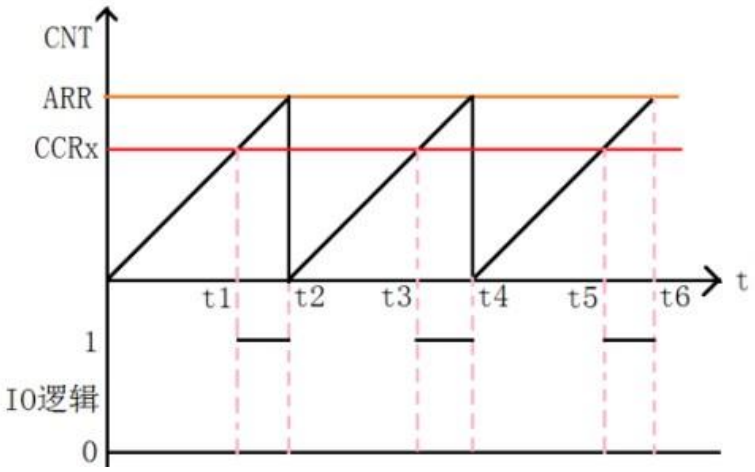
（热心网友冲哥提供）

前言：

众所周知，早早的就出了带硬件移相的 PWM，但是好多人还不理解这个硬件移相 PWM 有什么作用，这篇帖子带大家仔细研究下这个硬件移相 PWM 的一部分——不对称 PWM。

一、普通 PWM 模式

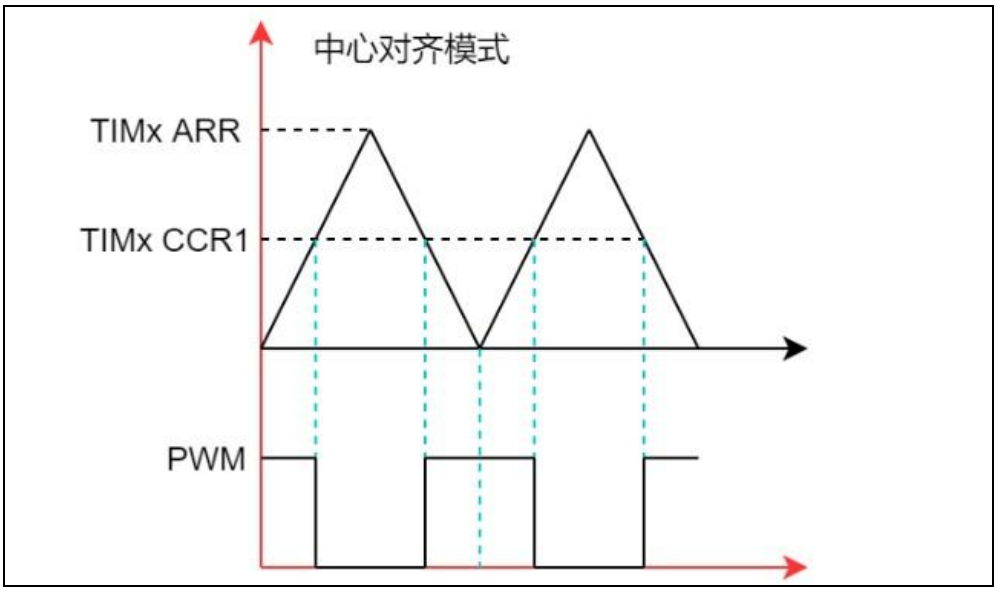
常见的 PWM 模式如下图所示，通过一个 ARR 寄存器设定最大计数值（向上计数模式），CNT 表示当前的计数，当计数超过 ARR 的时候被清 0，当 CNT 数值小于当前通道的 CCRx 的时候输出低电平，反之输出高电平，这个是常规的 PWM 模式 2（不开启极性反相的时候！）。



再来看下手册这部分，可以看到 PWM 模式 1 其实就是 PWM 模式 2 的反过来的波形

110	PWM模式1	在向上计数时，当PWMn_CNT<PWMn_CCR1时 OCnREF输出高，否则OCnREF输出低 在向下计数时，当PWMn_CNT>PWMn_CCR1时 OCnREF输出低，否则OCnREF输出高
111	PWM模式2	在向上计数时，当PWMn_CNT<PWMn_CCR1时 OCnREF输出低，否则OCnREF输出高 在向下计数时，当PWMn_CNT>PWMn_CCR1时 OCnREF输出高，否则OCnREF输出低

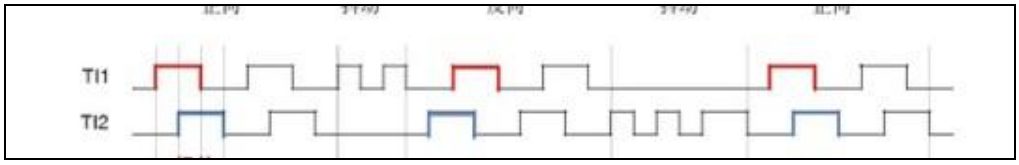
当然上面的都是边沿计数，还有一种中心对齐的计数



但是这种模式就是只能生成中心对称的波形，如果单纯的只想做几路不同占空比输出的波形，那确实没一点问题。可以思考下，像上面的几种模式，总的规划出来就分为四大类：

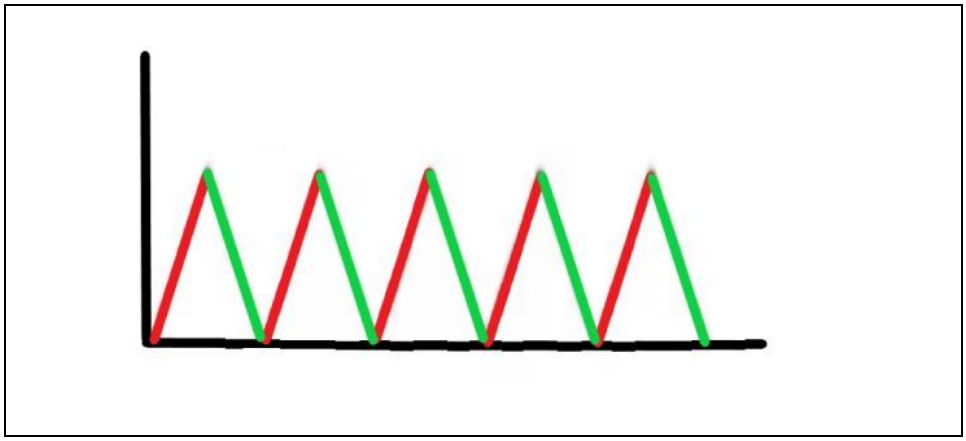
1.前面高电平后面低电平
2.前面低电平后面高电平
3.两边低电平中间高电平
4.两边高电平中间低电平

上面 1 和 2 是同一种计数方式，3 和 4 是一种计数方式，同一时间只能选择一种计数方式。但是这时候，我们如果想要生成一个如下图的正交编码信号的 PWM 波形该如何实现的？

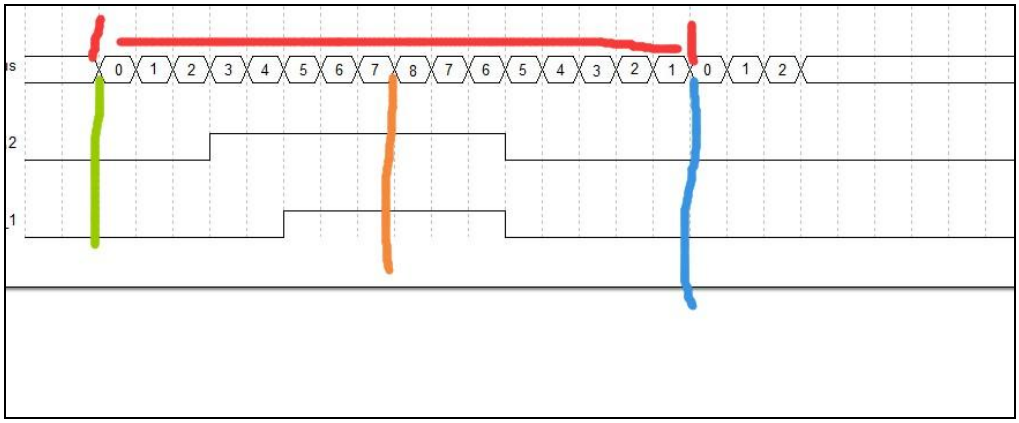


二、不对称互补 PWM

按照我个人的理解，不对称 PWM 的本质就是起始和结束电平不受约束，按照手册的说法他必须使用中心对齐计数模式，开始电平是可以在上升计数段里任意定义，结束电平是在下降计数段里任意定义的。在下面的图里，一个红色线和一个绿色线为一个中心计数模式的一个周期（从 0 计数到最大值 arr 再到 0 的过程），开始电平可以在红线里的任意位置，结束电平可以是绿线的任意位置，这样是不是就很好分配了。



再画个简单的示意图：



假设红色的是整个计数周期，将他对半的分为绿-橙区域和橙-蓝区域，开始的电平可以在这个绿-橙区域任意位置，结束电平也可以定义在这个橙-蓝区域的任意位置，此时你想要这个波形横移多少位置不就是分分钟的了。

当然在开始之前我们还需要了解一些基本知识：

24.3.1 不对称 PWM 模式

在不对称模式下，生成的两个中心对齐 PWM 信号间允许存在可编程相移。频率由 PWMA_ARR 寄存器的值确定，而占空比和相移则由一对 PWMA_CCRx 寄存器确定。两个寄存器分别控制递增计数和递减计数期间的 PWM，这样每半个 PWM 周期便会调节一次 PWM：

- OC1REFC（或 OC2REFC）由 PWMA_CCR1 和 PWMA_CCR2 控制
- OC3REFC（或 OC4REFC）由 PWMA_CCR3 和 PWMA_CCR4 控制

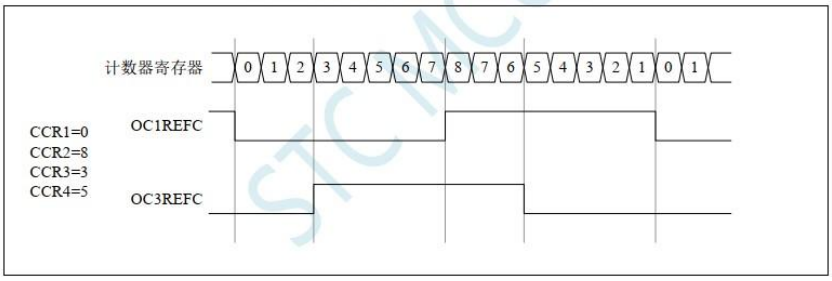
两个通道可以独立选择不对称 PWM 模式(每对 CCR 寄存器控制一个 OCx 输出)，只需向 PWMA_CCMRx 寄存器的 OCxM 位写入“1110”（不对称 PWM 模式 1）或“1111”（不对称 PWM 模式 2）。

给定通道用作不对称 PWM 通道时，也可使用其互补通道。例如，如果通道 1 上产生 OC1REFC 信号（不对称 PWM 模式 1），则由于不对称 PWM 模式 1 的原因，通道 2 上可输出 OC2REF 信号或 OC2REFC 信号。

（注：出于兼容性原因，OCxM[3:0]位域分为两部分，最高有效位与最低有效的 3 位不相邻。）

下图显示了不对称 PWM 模式下可以产生的信号示例（通道 1 到通道 4 在不对称 PWM 模式 1 下配置）。与死区发生器配合使用时，这可控制相移全桥直流到直流转换器。

（产生 2 个 50%占空比的相移 PWM 信号）



手册上关于这个模式的描述就这么一页，我给他归纳总结了一下：

1.需要在中心对齐计数模式下使用

2.必须手动设置为不对称模式才能使用这个输出（**切记这个寄存器的位置不连续，要重点注意**）

3.两个通道才能组合成为一个完整的不对称输出通道，且通道 1（又名 OC1REFC 或 OC2REFC）由 CCR1 和 CCR2 控制，通道 2（又名 OC3REFC 或 OC4REFC）由 CCR3 和 CCR4 控制。

4.1P 端口设置为不对称输出时起始电平时间点为 CCR1,结束时间点为 CCR2;　2P 端口设置为不对称输出时起始电平时间点为 CCR2,结束时间点为 CCR1;

5.别的配置都可以参考普通 PWM 的配置

6.一个通道会占用两个端口的 CCR 寄存器，其中一个作为不对称输出的时候，另一个还可以作为普通的 pwm 模式输出。

三、寄存器描述

1.设置为中心对齐计数

23.9.4 控制寄存器 1（PWMx_CR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CR1	FEC0H	ARPEA	CMSA[1:0]		DIRA	OPMA	URSA	UDISA	CENA
PWMB_CR1	FEE0H	ARPEB	CMSB[1:0]		DIRB	OPMB	URSB	UDISB	CENB

ARPEn：自动预装载允许位（n=A,B）

0：PWMn_ARR 寄存器没有缓冲，它可以被直接写入

1：PWMn_ARR 寄存器由预装载缓冲器缓冲

CMSn[1:0]：选择对齐模式（n= A,B）

CMSn[1:0]	对齐模式	说明
00	边沿对齐模式	计数器依据方向位（DIR）向上或向下计数
01	中央对齐模式1	计数器交替地向上和向下计数。 配置为输出的通道的输出比较中断标志位(CCnIF)只在计数器向下计数时被置1。
10	中央对齐模式2	计数器交替地向上和向下计数。 配置为输出的通道的输出比较中断标志位(CCnIF)只在计数器向上计数时被置1。
11	中央对齐模式3	计数器交替地向上和向下计数。 配置为输出的通道的输出比较中断标志位(CCnIF)在计数器向上和向下计数时均被置1。

注 1：在计数器开启时（CEN=1），不允许从边沿对齐模式转换到中央对齐模式。

2.手动使能不对称模式

		和OC2REF的逻辑与运算结果。
1110	不对称PWM模式1	OC1REF与在PWM模式1下的行为相同。计数器递增计数时，OC1REFC输出OC1REF；计数器递减计数时，OC1REFC输出OC2REF。
1111	不对称PWM模式2	OC1REF与在PWM模式2下的行为相同。计数器递增计数时，OC1REFC输出OC1REF；计数器递减计数时，OC1REFC输出OC2REF。

注 1：一旦 LOCK 级别设为 3（PWMn_BKR 寄存器中的 LOCK 位）并且 CCnS=00（该通道配置成

这里尤其要注意 OCxM[bit2:0]在 CCMx 寄存器，但是最高位 OCxM[bit3]在 CCMxX 寄存器（小写的 x 表示哪个端口）；

3.通道比较数值设置

23.9.25 捕获/比较寄存器 1/5 高 8 位（PWMx_CCR1H）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_CCR1H	FED5H	CCR1[15:8]							
PWMB_CCR5H	FEF5H	CCR5[15:8]							

CCRn[15:8]：捕获/比较 n 的高 8 位值（n=1,5）

若 CCn 通道配置为输出：CCRn 包含了装入当前比较值（预装载值）。如果在 PWMn_CCMR1 寄存器（OCnPE 位）中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 n 寄存器中。当前比较值同计数器 PWMn_CNT 的值相比较，并在 OCn 端口上产生输出信号。

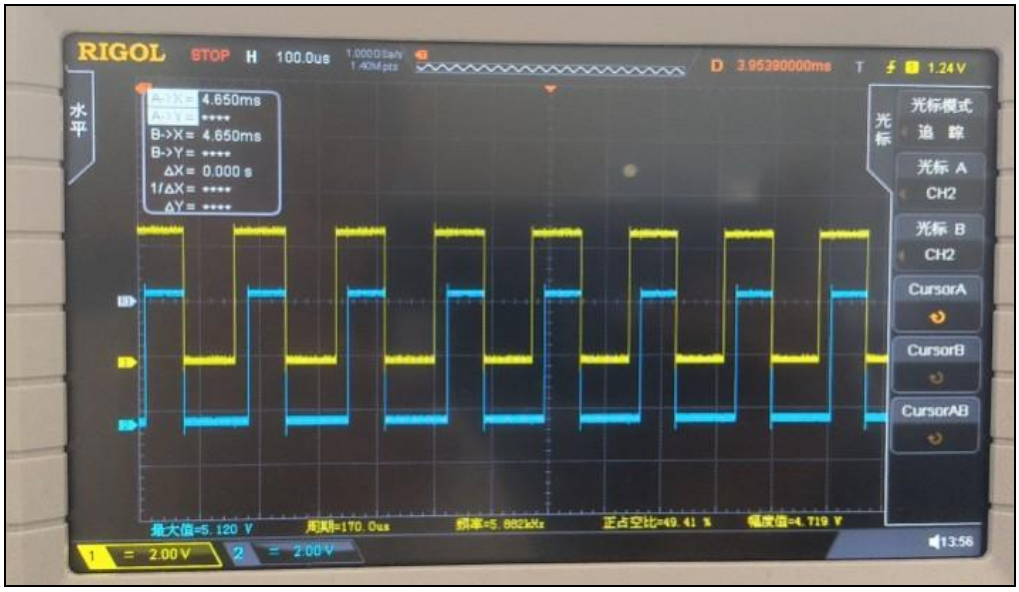
若 CCn 通道配置为输入：CCRn 包含了上一次输入捕获事件发生时的计数器值（此时该寄存器为只读）。

四、代码实战

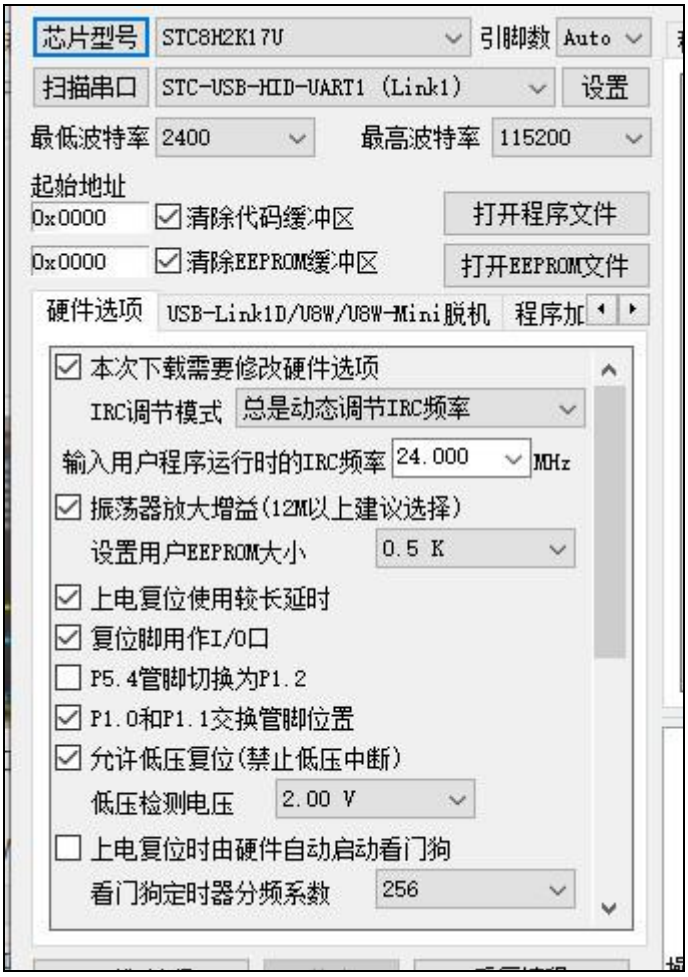
首先我们可以看下来验证一下第二章里的第四点 4.1P 端口设置为不对称输出时起始电平时间点为 CCR1, 结束时间点为 CCR2; 2P 端口设置为不对称输出时起始电平时间点为 CCR2,结束时间点为 CCR1）是什么意思：

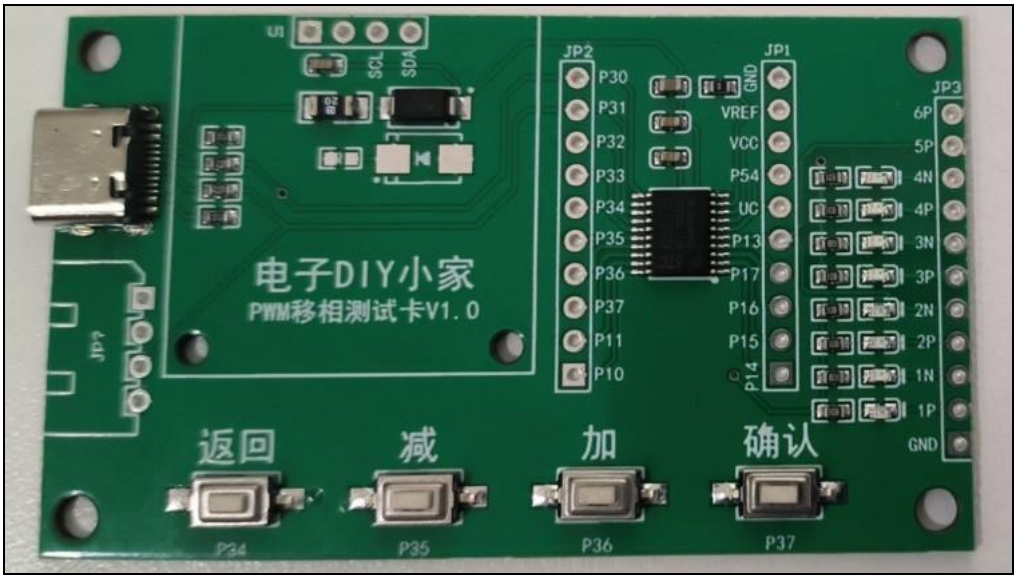
1）首先配置 1P 端口为不对称 PWM 模式 2 输出，2P 端口 PWM 模式 2 输出

```
PWM1_Duty = 3;
PWM2_Duty = 5;
PWMA_PSCRH = 0;
PWMA_PSCRL = 255;
PWMA_CCER1 = 0x00; //写 CCMRx 前必须先清零 CCxE 关闭通道
PWMA_CCER2 = 0x00;
PWMA_CCMR1X = 0x01;//端口 1:不对称 PWM2 模式
PWMA_CCMR1 = 0x70;
PWMA_CCMR2X = 0x00;//端口 2:PWM2 模式
PWMA_CCMR2 = 0x70;
PWMA_CCER1 = 0x55; //配置通道输出使能和极性
PWMA_CCER2 = 0x55;
PWMA_ARRH = (u8)(PWM_PERIOD >> 8); //设置周期时间
PWMA_ARRL = (u8)PWM_PERIOD;
PWMA_ENO = 0x00;
PWMA_ENO |= ENO1P; //使能输出
PWMA_ENO |= ENO2P; //使能输出
PWMA_PS = 0x00;  //高级 PWM 通道输出脚选择位
PWMA_PS |= PWM1_0; //选择 PWM1_0 通道
PWMA_PS |= PWM2_0; //选择 PWM2_0 通道
```



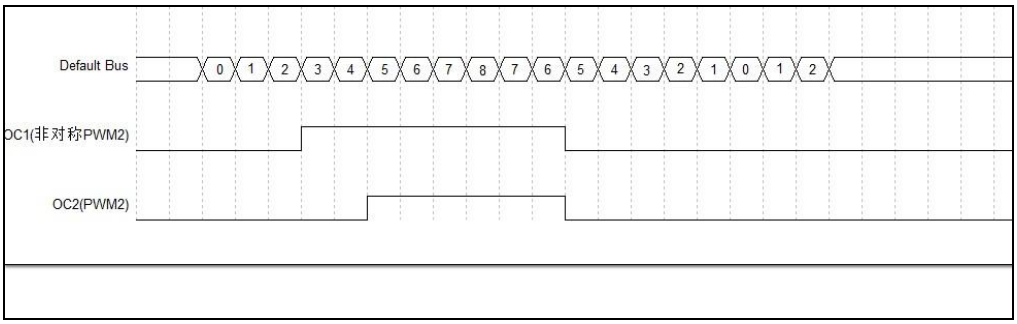
效果图如上，黄色线条为 1P 端口，即不对称 PWM2 模式。蓝色线条为 2P 端口，即 PWM2 模式。下载选项和使用的硬件如下图所示





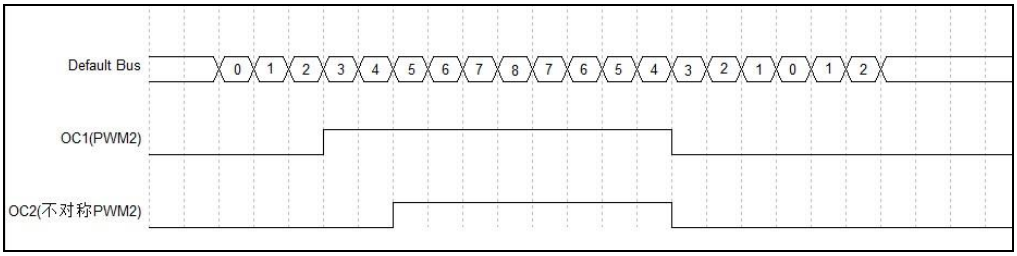
1.首先 PWM 频率 = 时钟/(分频+1)/ARR/2 = 24M / 256/8/2 ≈ 5.86Khz，和示波器测的频率基本一致

2.按照我们的分析，实际上端口 1 应该在上升计数为 3 的时候变成高电平，下降计数为 5 的时候变为低电平，占空比为 50%。端口 2 应该在上升计数为 5 的时候变成高电平，下降计数为 5 的时候变为低电平，例如下图所示，可以看到最终测量到的结果和这个一致，可见这个分析是正确的。

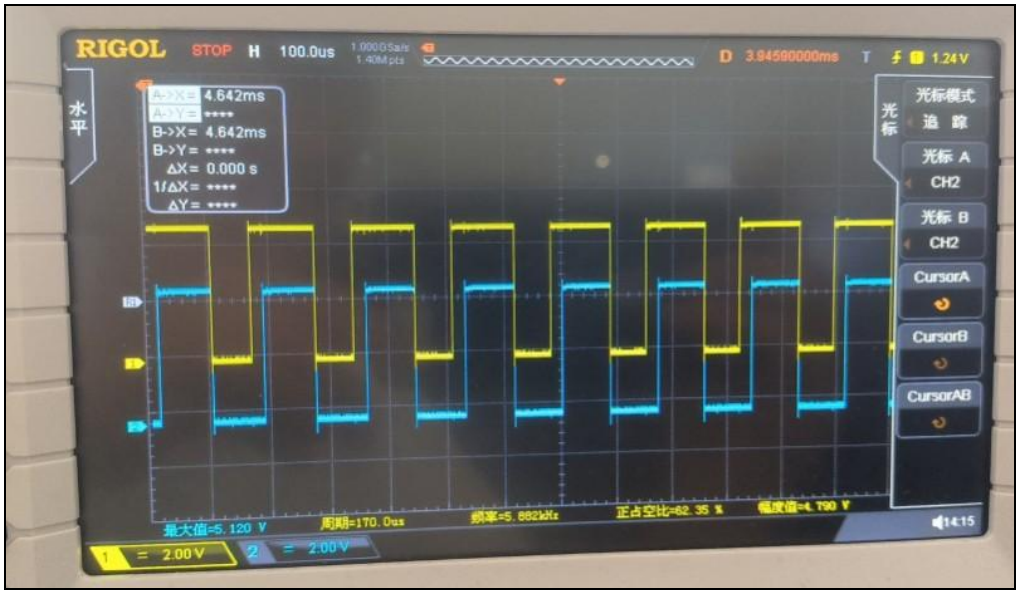


2）首先配置 1P 端口为不对称 PWM 模式 2 输出，2P 端口 PWM 模式 2 输出

```
PWM1_Duty = 3;
PWM2_Duty = 5;
PWMA_PSCRH = 0;
PWMA_PSCRL = 255;
PWMA_CCER1 = 0x00; //写  CCMRx  前必须先清零  CCxE  关闭通道
PWMA_CCER2 = 0x00;
PWMA_CCMR1X = 0x00;//端口 1:称 PWM2 模式
PWMA_CCMR1 = 0x70;
PWMA_CCMR2X = 0x01;//端口 2:不对 PWM2 模式
PWMA_CCMR2 = 0x70;
PWMA_CCER1 = 0x55; //配置通道输出使能和极性
PWMA_CCER2 = 0x55;
PWMA_ARRH = (u8)(PWM_PERIOD >> 8); //设置周期时间
PWMA_ARRL = (u8)PWM_PERIOD;
PWMA_ENO = 0x00;
PWMA_ENO |= ENO1P; //使能输出
PWMA_ENO |= ENO2P; //使能输出
PWMA_PS = 0x00;   //高级  PWM  通道输出脚选择位
PWMA_PS |= PWM1_0; //选择  PWM1_0  通道
PWMA_PS |= PWM2_0; //选择  PWM2_0  通道
```



我们下载进去实测一下：



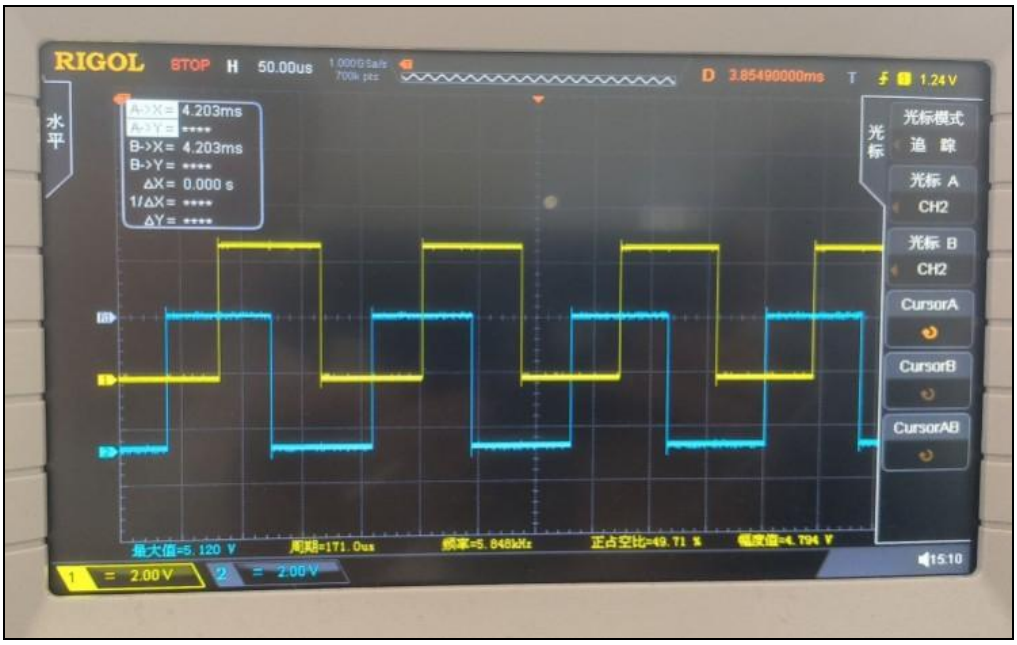
可以看到这个波形就是我们预期的波形

3）最后我们来实现一下正交编码信号的波形。
通道 1 使用 PWM2 不对称模式，通道 3 使用不对称 PWM2 模式

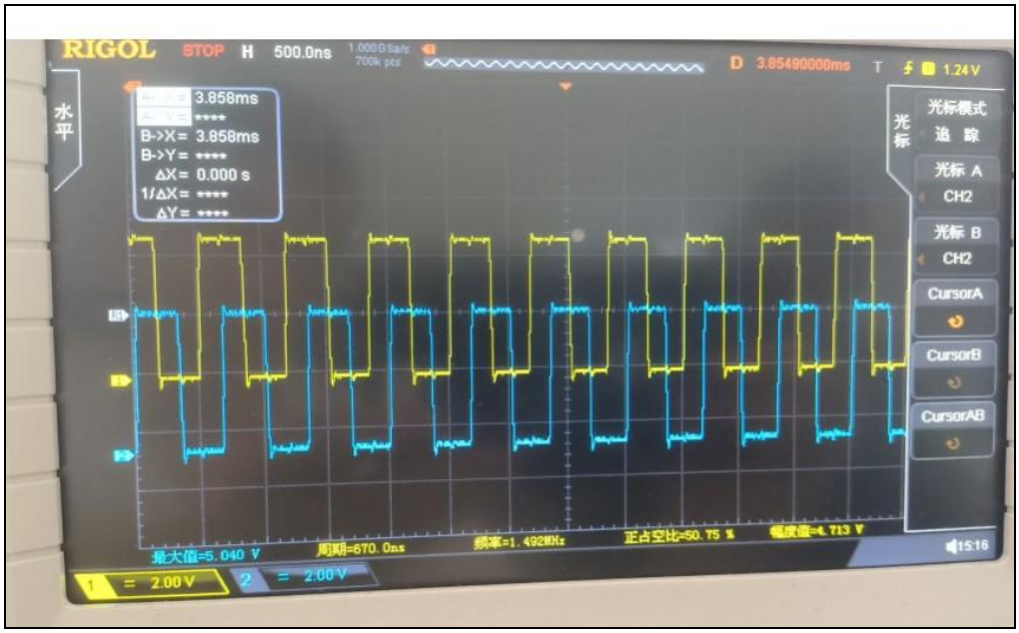
```
PWM1_Duty = 0;
PWM2_Duty = 8;
PWM3_Duty = 4;
PWM4_Duty = 4;
PWMA_PSCRH = 0;
PWMA_PSCRL = 255;
PWMA_CCER1 = 0x00; //写 CCMRx 前必须先清零 CCxE 关闭通道
PWMA_CCER2 = 0x00;
PWMA_CCMR1X = 0x01;//通道 1:PWM2 不对称模式
PWMA_CCMR1 = 0x70;
PWMA_CCMR2X = 0x00;//通道 2:PWM2 模式,极性反转
PWMA_CCMR2 = 0x70;
PWMA_CCMR3X = 0x01;//通道 3:不对称 PWM2 模式
PWMA_CCMR3 = 0x70;
PWMA_CCMR4X = 0x00;//通道 4:PWM2 模式
PWMA_CCMR4 = 0x70;
PWMA_CCER1 = 0x55; //配置通道输出使能和极性
PWMA_CCER2 = 0x55;
PWMA_ARRH = (u8)(PWM_PERIOD >> 8); //设置周期时间
PWMA_ARRL = (u8)PWM_PERIOD;
PWMA_ENO = 0x00;
PWMA_ENO |= ENO1P; //使能输出
PWMA_ENO |= ENO3P; //使能输出
PWMA_PS = 0x00;  //高级 PWM 通道输出脚选择位
PWMA_PS |= PWM1_0; //选择 PWM1_0 通道
```

```
PWMA_PS |= PWM2_0; //选择  PWM2_0  通道
PWMA_PS |= PWM3_0; //选择  PWM3_0  通道
PWMA_PS |= PWM4_0; //选择  PWM4_0  通道
```

最后下载到单片机看下最终的波形效果。



之所以这里要用硬件，就是频率可以很快，且不需要进中断即可实现。（代码里为了方便测量将 PSC 分频寄存器配置了 255，实际上这个改为 0，即不分频速度分分钟能到 Mhz 级别，效果如下）



30 高速高级 PWM（HSPWM），可以使用 PLL 高速时钟作为时钟源

产品线	高速高级 PWM
STC32G144K246 系列	**6 组**

STC32G144K246 系列单片机为 6 组高级 PWMA / PWMB / PWMC / PWMD / PWME / PWMF 提供了高速模式（HSPWMA / HSPWMB / HSPWMC / HSPWMD / HSPWME / HSPWMF）。高速高级 PWM 采用的是异步时钟设计，与 CPU 时钟不同步，一旦使能了 PWM 的高速模式，必须使用异步方式读写 PWM 的特殊功能寄存器。

当系统运行在较低工作频率时，高速高级 PWM 可工作在高达 230M 的频率下。从而可以达到降低内核功耗，提升外设性能的目的。

30.1 寄存器描述

30.1.1 HSPWM 配置寄存器（HSPWMn_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSPWMA_CFG	7EFBF0H	-	-	-	PWMA_EXTN	-	PWMA_INTEN	PWMA_ASYNCEN	1
HSPWMB_CFG	7EFBF4H	-	-	-	-	-	PWMB_INTEN	PWMB_ASYNCEN	1
HSPWMC_CFG	7EF970H	-	-	-	PWMC_EXTN	-	PWMC_INTEN	PWMC_ASYNCEN	1
HSPWMD_CFG	7EF974H	-	-	-	-	-	PWMD_INTEN	PWMD_ASYNCEN	1
HSPWME_CFG	7EF688H	-	-	-	-	-	PWME_INTEN	PWME_ASYNCEN	1
HSPWMF_CFG	7EF68CH	-	-	-	-	-	PWMF_INTEN	PWMF_ASYNCEN	1

PWMn_ASYNCEN：异步控制模式使能位（n=A/B/C/D/E/F）

0：关闭异步控制。

1：使能异步控制模式。

注：当关闭异步控制时，高级 PWMn 为传统模式，此时高级 PWM 会自动选择系统工作频率，PWM 工作频率与系统工作频率相同；若需要时 PWM 工作在高速模式，则需要使能异步控制模式，此时 PWM 时钟可选择高速 PLL 输出时钟

PWMn_INTEN：异步模式中断使能位（n=A/B/C/D/E/F）

0：关闭异步模式下的 PWMn 中断。

1：使能异步模式下的 PWMn 中断。

注：异步模式下，若需要响应高级 PWMn 的中断，则必须使能 PWMn_INTEN 位

PWMA_EXTN：PWMA 间接寻址的基地址选择

0：间接寻址的基地址，BASE[15:7] = 0x1fd，间接寻址范围为 FE80H～FEFFH

1：间接寻址的基地址，BASE[15:7] = 0x1f2，间接寻址范围为 F900H～F97FH

PWMC_EXTN：PWMC 间接寻址的基地址选择

0：间接寻址的基地址，BASE[15:7] = 0x1f1，间接寻址范围为 F880H～F8FFH

1：间接寻址的基地址，BASE[15:7] = 0x1f2，间接寻址范围为 F900H～F97FH

B0：固定位 1。

30.1.2 HSPWM 地址寄存器（HSPWMn_ADR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSPWMA_ADR	7EFBF1H	PWMA_RW	PWMA_ADDR[6:0]						
HSPWMB_ADR	7EFBF5H	PWMB_RW	PWMB_ADDR[6:0]						
HSPWMC_ADR	7EF971H	PWMC_RW	PWMC_ADDR[6:0]						
HSPWMD_ADR	7EF975H	PWMD_RW	PWMD_ADDR[6:0]						
HSPWME_ADR	7EF689H	PWME_RW	PWME_ADDR[6:0]						
HSPWMF_ADR	7EF68DH	PWMF_RW	PWMF_ADDR[6:0]						

PWMn_ADDR[6:0]：高级 PWMn 的特殊功能寄存器地址低 7 位（n=A/B/C/D/E/F）

PWMn_RW/BUSY：读写控制位、状态位（n=A/B/C/D/E/F）

写 0：异步方式**写** PWMn 的特殊功能寄存器。

写 1：异步方式**读** PWMn 的特殊功能寄存器。

读 0：异步读写已经完成

读 1：异步读写正在进行，处于忙状态

30.1.3 HSPWM 数据寄存器（HSPWMn_DAT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HSPWMA_DAT	7EFBF2H	PWMA_DATA[7:0]							
HSPWMB_DAT	7EFBF6H	PWMB_DATA[7:0]							
HSPWMC_DAT	7EF972H	PWMC_DATA[7:0]							
HSPWMD_DAT	7EF976H	PWMD_DATA[7:0]							
HSPWME_DAT	7EF68AH	PWME_DATA[7:0]							
HSPWMF_DAT	7EF68EH	PWMF_DATA[7:0]							

PWMn_DATA[7:0]：高级 PWMn 的特殊功能寄存器数据（n=A/B/C/D/E/F）

写：写数据到高级 PWMn 的特殊功能寄存器。

读：从高级 PWMn 的特殊功能寄存器读取数据。

异步读取 PWMA 的特殊功能寄存器步骤：（其他高速 PWM 类似）

1、读取 HSPWMA_ADR，等待 BUSY 为 0，确定前一个异步读写已完成

2、将 PWMA 的特殊功能寄存器的低 7 位写入 HSPWMA_ADR，同时置“1”HSPWMA_ADR.7

3、读取 HSPWMA_ADR，等待 BUSY 为 0

4、读取 HSPWMA_DAT

异步写 PWMA 的特殊功能寄存器步骤：（其他高速 PWM 类似）

1、读取 HSPWMA_ADR，等待 BUSY 为 0，确定前一个异步读写已完成

2、将需要写入 PWMA 的特殊功能寄存器的数据写入 HSPWMA_DAT

3、将 PWMA 的特殊功能寄存器的低 7 位写入 HSPWMA_ADR，同时清“0”HSPWMA_ADR.7

4、读取 HSPWMA_ADR，等待 BUSY 为 0。（可跳过此步继续执行其他代码，以提高系统效率）

特别注意：特殊功能寄存器 PWMA_PS / PWMA_PS2 / PWMB_PS / PWMC_PS / PWMC_PS2 / PWMD_PS / PWME_PS / PWME_PS2 / PWMF_PS 属于端口控制寄存器，不属于 PWM 寄存器组，所以无论是否启动 PWM 的异步控制模式，这些寄存器都只能使用普通的同步模式进行读写

30.2 高速高级 PWM + DMA（组合使用）特别注意事项

STC32G144K246 系列单片机的高级 PWMA 工作在普通模式和高速模式时，均支持 DMA 操作。

下表是 PWMA 进行 DMA 操作时的相关特殊功能寄存器。

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PWMA_DER	PWMA_DMA 事件使能寄存器	7EF948H	-	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE	0000,0000
PWMA_DBA	PWMA_DMA 基址寄存器	7EF949H	-	-	-	DBA[4:0]					xxx0,0000
PWMA_DBL	PWMA_DMA 基址长度寄存器	7EF94AH	-	-	-	DBL[4:0]					xxx0,0000
PWMA_DMACR	PWMA_DMA 控制寄存器	7EF94BH	-	-	-	DSKIP	DDIR	DMAEN	SIZE[1:0]		xxx0,0000

当 PWMA 工作在普通模式（非高速模式）时：

PWMA 的所有相关寄存器均使用直接寻址方式读写，DMA 操作时，上表中的 PWMA 的 DMA 相关寄存器也使用直接寻址方式读写。

当 PWMA 工作在高速模式时：

PWMA 的所有相关寄存器均使用间接寻址方式读写，**DMA 操作时，上表中的 PWMA 的 DMA 相关寄存器也必须使用间接寻址的方式进行读写（这里需要特别注意）**。

PWM 的 DMA 在操作的时候，触发 DMA 启动是有时序要求的：

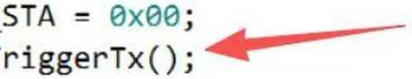
在 PWM-DMA 的完成中断中，打开下一次的 DMA 触发请求

这样，只需要关心对应数组中的变量更新即可

例如，在配置好 DMA 后，初始化中给一个触发启动 DMA

后续依靠 DMA 完成中断的 DMA 再次触发来循环执行，后续主循环中只需要更改对应的数组变量

```
5  void pwma_dma_isr(void) interrupt DMA_PWMAT_VECTOR
6  {
7      DMA_PWMAT_STA = 0x00;
8      DMA_PWMA_TriggerTx();
9  }
```



中断内触发下一次

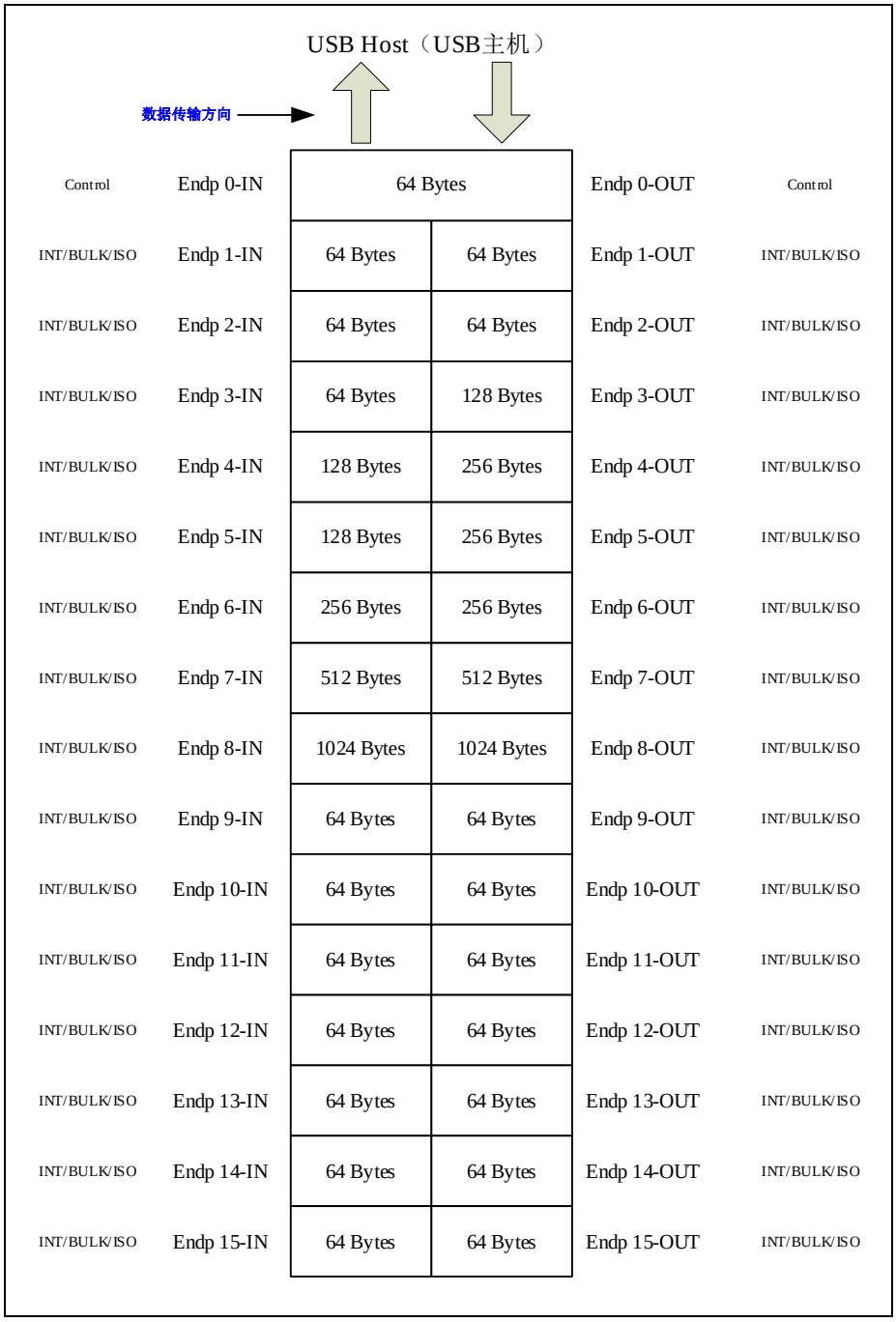
31 USB 2.0-FS 通用串行总线

产品线	USB
STC32G144K246 系列	●

STC32G144K246 系列单片机内部集成 USB2.0/USB1.1 兼容全速 USB，**16 个双向端点**，支持 4 种端点传输模式（控制传输、中断传输、批量传输和同步传输）。

USB 模块共有 **5760** 字节的 FIFO，结构如下

（特别注意：每个端点的 FIFO 大小固定为 64~1024 字节，非任意分配）：



Control	Endp 0-IN	64 Bytes		Endp 0-OUT	Control
INT/BULK/ISO	Endp 1-IN	64 Bytes	64 Bytes	Endp 1-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 2-IN	64 Bytes	64 Bytes	Endp 2-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 3-IN	64 Bytes	128 Bytes	Endp 3-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 4-IN	128 Bytes	256 Bytes	Endp 4-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 5-IN	128 Bytes	256 Bytes	Endp 5-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 6-IN	256 Bytes	256 Bytes	Endp 6-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 7-IN	512 Bytes	512 Bytes	Endp 7-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 8-IN	1024 Bytes	1024 Bytes	Endp 8-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 9-IN	64 Bytes	64 Bytes	Endp 9-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 10-IN	64 Bytes	64 Bytes	Endp 10-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 11-IN	64 Bytes	64 Bytes	Endp 11-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 12-IN	64 Bytes	64 Bytes	Endp 12-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 13-IN	64 Bytes	64 Bytes	Endp 13-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 14-IN	64 Bytes	64 Bytes	Endp 14-OUT	INT/BULK/ISO
INT/BULK/ISO	Endp 15-IN	64 Bytes	64 Bytes	Endp 15-OUT	INT/BULK/ISO

31.1 USB 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
USBCLK	USB 时钟控制寄存器	DCH	ENCKM	PCKI[1:0]		CRE	TST_USB	TST_PHY	PHYTST[1:0]		0010,0000
USBCON	USB 控制寄存器	F4H	ENUSB	ENUSBRST	PS2M	PUEN	PDEN	DFREC	DP	DM	0000,0000
USBADR	USB 地址寄存器	ECH	BUSY	AUTORD	UADR[5:0]						0000,0000
USBDAT	USB 数据寄存器	FCH									0000,0000

31.1.1 USB 控制寄存器（USBCON）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USBCON	F4H	ENUSB	ENUSBRST	PS2M	PUEN	PDEN	DFREC	DP	DM

ENUSB：USB 功能与 USB 时钟控制位

0：关闭 USB 功能与 USB 时钟

1：使能 USB 功能与 USB 时钟

ENUSBRST：USB 复位设置控制位

0：关闭 USB 复位设置

1：使能 USB 复位

PS2M：PS2 mode 功能控制位

0：关闭 PS2 mode 功能

1：使能 PS2 mode 功能

PUEN：DP/DM 端口上 1.5K 上拉电阻控制位

0：禁止上拉电阻

1：使能上拉电阻

PDEN：DP/DM 端口上 500K 下拉电阻控制位

0：禁止下拉电阻

1：使能下拉电阻

DFREC：差分接收状态位（只读）

0：当前 DP/DM 的差分状态为“0”

1：当前 DP/DM 的差分状态为“1”

DP：D+端口状态（PS2 为 0 时只读，PS2 为 1 时可读写）

0：当前 D+为逻辑 0 电平

1：当前 D+为逻辑 1 电平

DM：D-端口状态（PS2 为 0 时只读，PS2 为 1 时可读写）

0：当前 D-为逻辑 0 电平

1：当前 D-为逻辑 1 电平

31.1.2 USB 时钟控制寄存器（USBCLK）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USBCLK	DCH	ENCKM	PCKI[1:0]		CRE	TST_USB	TST_PHY	PHYTST[1:0]	

ENCKM：PLL 倍频控制

0：禁止 PLL 倍频

1：使能 PLL 倍频

PCKI[1:0]：PLL 时钟选择

PCKI[1:0]	PLL 时钟源
00	/1
01	/2
10	/4
11	/8

CRE：时钟追频控制位

0：禁止时钟追频

1：使能时钟追频

TST_USB: USB 测试模式

0：禁止 USB 测试模式

1：使能 USB 测试模式

TST_PHY: PHY 测试模式

0：禁止 PHY 测试模式

1：使能 PHY 测试模式

PHYTST[1:0]：USB PHY 测试

PHYTST[1:0]	方式	DP	DM
00	方式 0：正常	x	x
01	方式 1：强制“1”	1	0
10	方式 2：强制“0”	0	1
11	方式 3：强制单端“0”	0	0

31.1.3 USB 间址地址寄存器（USBADR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USBADR	FCH	BUSY	AUTORD	UADR[5:0]					

BUSY：USB 寄存器读忙标志位

写 0：无意义

写 1：启动 USB 间接寄存器的读操作，地址由 USBADR 设定

读 0：USBDAT 寄存器中的数据有效

读 1：USBDAT 寄存器中的数据无效，USB 正在读取间接寄存器

~~AUTORD：USB 寄存器自动读标志，用于 USB 的 FIFO 的块读取~~

~~写 0：每次读取间接 USB 寄存器都必须先写 BUSY 标志位~~

~~写 1：当软件读取 USBDAT 时，下一个 USB 间接寄存器的读取将自动启动（USBADR 不变）~~

UADR[5:0]：USB 间接寄存器的地址

31.1.4 USB 间址数据寄存器（USBDAT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
USBDAT	ECH	UDAT[7:0]							

UDAT[7:0]：用于间接读写 USB 寄存器

31.2 USB 控制器寄存器（SIE）

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
30H	UTRKCTL	UTRKSTS						
28H	FIFO8	FIFO9	FIFO10	FIFO11	FIFO12	FIFO13	FIFO14	FIFO15
20H	FIFO0	FIFO1	FIFO2	FIFO3	FIFO4	FIFO5	FIFO6	FIFO7
18H								
10H	INMAXP	CSR0 INCSR1	INCSR2	OUTMAXP	OUTCSR1	OUTCSR2	COUNT0 OUTCOUNT1	OUTCOUNT2
08H	INTRIN2E	INTROUT1E	INTROUT2E	INTRUSBE	FRAME1	FRAME2	INDEX	
00H	FADDR	POWER	INTRIN1	INTRIN2	INTROUT1	INTROUT2	INTRUSB	INTRIN1E

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
FADDR	USB 功能地址寄存器	00H	UPDATE	UADDR[6:0]							0000,0000
POWER	USB 电源管理寄存器	01H	ISOUD	-	-	-	USBRST	USBRSU	USBSUS	ENSUS	0xxx,0000
INTRIN1	USB 端点 IN 中断标志位	02H	EP7INIF	EP6INIF	EP5INIF	EP4INIF	EP3INIF	EP2INIF	EP1INIF	EP0IF	0000,0000
INTRIN2	USB 端点 IN 中断标志位	03H	EP15INIF	EP14INIF	EP13INIF	EP12INIF	EP11INIF	EP10INIF	EP9INIF	EP8IF	0000,0000
INTROUT1	USB 端点 OUT 中断标志位	04H	EP7OUTIF	EP6OUTIF	EP5OUTIF	EP4OUTIF	EP3OUTIF	EP2OUTIF	EP1OUTIF	-	0000,000x
INTROUT2	USB 端点 OUT 中断标志位	05H	EP15OUTIF	EP14OUTIF	EP13OUTIF	EP12OUTIF	EP11OUTIF	EP10OUTIF	EP9OUTIF	EP8OUTIF	0000,0000
INTRUSB	USB 电源中断标志位	06H	-	-	-	-	SOFIF	RSTIF	RSUIF	SUSIF	xxxx,0000
INTRIN1E	USB 端点 IN 中断允许位	07H	EP8INIE	EP9INIE	EP5INIE	EP4INIE	EP3INIE	EP2INIE	EP1INIE	EP0IE	1111,1111
INTRIN2E	USB 端点 IN 中断允许位	08H	EP15INIE	EP14INIE	EP13INIE	EP12INIE	EP11INIE	EP10INIE	EP9INIE	EP8IE	1111,1111
INTROUT1E	USB 端点 OUT 中断允许位	09H	EP7OUTIE	EP6OUTIE	EP5OUTIE	EP4OUTIE	EP3OUTIE	EP2OUTIE	EP1OUTIE	-	1111,111x
INTROUT2E	USB 端点 OUT 中断允许位	0AH	EP15OUTIE	EP14OUTIE	EP13OUTIE	EP12OUTIE	EP11OUTIE	EP10OUTIE	EP9OUTIE	EP8OUTIE	1111,1111
INTRUSBE	USB 电源中断允许位	0BH	-	-	-	-	SOFIE	RSTIE	RSUIE	SUSIE	xxxx,0110
FRAME1	USB 数据帧号低字节	0CH	FRAME[7:0]								0000,0000
FRAME2	USB 数据帧号高字节	0DH	-	-	-	-	-	FRAME[10:8]			xxxx,x000
INDEX	USB 端点索引寄存器	0EH	-	-	-	-	-	INDEX[2:0]			xxxx,x000
INMAXP	IN 端点的最大数据包大小	10H	INMAXP[7:0]								0000,0000
CSR0	端点 0 控制状态寄存器	11H	SSUEND	SOPRDY	SDSTL	SUEND	DATEND	STSTL	IPRDY	OPRDY	0000,0000
INCSR1	IN 端点控制状态寄存器 1	11H	-	CLRDT	STSTL	SDSTL	FLUSH	UNDRUN	FIFONE	IPRDY	x000,0000
INCSR2	IN 端点控制状态寄存器 2	12H	AUTOSET	ISO	MODE	ENDMA	FCDT	-	-	-	0010,0xxx
OUTMAXP	OUT 端点的最大数据包大小	13H	OUTMAXP[7:0]								0000,0000
OUTCSR1	OUT 端点控制状态寄存器 1	14H	CLRDT	STSTL	SDSTL	FLUSH	DATERR	OVRRUN	FIFOFUL	OPRDY	0000,0000
OUTCSR2	OUT 端点控制状态寄存器 2	15H	AUTOCLR	ISO	ENDMA	DMAMD	-	-	-	-	0000,xxxx
COUNT0	端点 0 的 OUT 长度	16H	-	OUTCNT0[6:0]							x000,0000
OUTCOUNT1	USB 端点 OUT 长度低字节	16H	OUTCNT[7:0]								0000,0000
OUTCOUNT2	USB 端点 OUT 长度高字节	17H	-	-	-	-	-	OUTCNT[10:8]			xxxx,x000
FIFO0	端点 0 的 FIFO 访问寄存器	20H	FIFO0[7:0]								0000,0000
FIFO1	端点 1 的 FIFO 访问寄存器	21H	FIFO1[7:0]								0000,0000
FIFO2	端点 2 的 FIFO 访问寄存器	22H	FIFO2[7:0]								0000,0000
FIFO3	端点 3 的 FIFO 访问寄存器	23H	FIFO3[7:0]								0000,0000
FIFO4	端点 4 的 FIFO 访问寄存器	24H	FIFO4[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
FIFO5	端点 5 的 FIFO 访问寄存器	25H	FIFO5[7:0]								0000,0000
FIFO6	端点 6 的 FIFO 访问寄存器	26H	FIFO6[7:0]								0000,0000
FIFO7	端点 7 的 FIFO 访问寄存器	27H	FIFO7[7:0]								0000,0000
FIFO8	端点 5 的 FIFO 访问寄存器	28H	FIFO8[7:0]								0000,0000
FIFO9	端点 9 的 FIFO 访问寄存器	29H	FIFO9[7:0]								0000,0000
FIFO10	端点 10 的 FIFO 访问寄存器	2AH	FIFO10[7:0]								0000,0000
FIFO11	端点 11 的 FIFO 访问寄存器	2BH	FIFO11[7:0]								0000,0000
FIFO12	端点 12 的 FIFO 访问寄存器	2CH	FIFO12[7:0]								0000,0000
FIFO13	端点 13 的 FIFO 访问寄存器	2DH	FIFO13[7:0]								0000,0000
FIFO14	端点 14 的 FIFO 访问寄存器	2EH	FIFO14[7:0]								0000,0000
FIFO15	端点 15 的 FIFO 访问寄存器	2FH	FIFO15[7:0]								0000,0000
UTRKCTL	USB 跟踪控制寄存器	30H	FTM1	FTM0	INTV[1:0]		ENST5	RES[2:0]			1011,1011
UTRKSTS	USB 跟踪状态寄存器	31H	INTVCNT[3:0]				STS[1:0]		TST_UTRK	UTRK_RDY	1111,00x0

31.2.1 USB 功能地址寄存器（FADDR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
FADDR	00H	UPDATE	UADDR[6:0]						

UPDATE：更新 USB 功能地址

0：最后的 UADDR 地址已生效

1：最后的 UADDR 地址还未生效

UADDR[6:0]：保存 USB 的 7 位功能地址

31.2.2 USB 电源控制寄存器（POWER）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
POWER	01H	ISOUD	-	-	-	USBRST	USBRSU	USBSUS	ENSUS

ISOUD（ISO Update）：ISO 更新

0：当软件向 IPRDY 写“1”时，USB 在收到下一个 IN 令牌后发送数据包

1：当软件向 IPRDY 写“1”时，USB 在收到 SOF 令牌后发送数据包，如果 SOF 令牌之前收到 IN 令牌，则 USB 发送长度为 0 的数据包

USBRST（USB Reset）：USB 复位控制位

向此位写“1”，可强制产生异步 USB 复位。读取此位可以获得当前总线上的复位状态信息

0：总线上没有检测到复位信号

1：总线上检测到了复位信号

USBRSU（USB Resume）：USB 恢复控制位

以软件方式在总线上强制产生恢复信号，以便将 USB 设备从挂起方式进行远程唤醒。当 USB 处于挂起模式（USBSUS=1）时，向此位写“1”，将强制在 USB 总线上产生恢复信号，软件应在 10-15ms 后向此位写“0”，以结束恢复信号。软件向 USBRSU 写入“0”后将产生 USB 恢复中断，此时硬件会自动将 USBSUS 清“0”

USBSUS（USB Suspend）：USB 挂起控制位

当 USB 进入挂起方式时，此位被硬件置“1”。当以软件方式在总线上强制产生恢复信号后或者在总线上检测到恢复信号时且在读取了 INTRUSB 寄存器后，硬件自动将此位清“0”。

ENSUS（Enable Suspend Detection）：使能 USB 挂起方式检测

0：禁止挂起检测，USB 将忽略总线上的挂起信号

1：使能挂起检测，当检测到总线上的挂起信号，USB 将进入挂起方式

31.2.3 USB 端点 IN 中断标志位（INTRIN1/2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTRIN1	02H	EP7INIF	EP6INIF	EP5INIF	EP4INIF	EP3INIF	EP2INIF	EP1INIF	EP0IF
INTRIN2	03H	EP15INIF	EP14INIF	EP13INIF	EP12INIF	EP11INIF	EP10INIF	EP9INIF	EP8INIF

EP0IF：端点 0 的 IN/OUT 中断标志位

0：端点 0 的 IN/OUT 中断无效

1：端点 0 的 IN/OUT 中断有效

EPnINIF：端点 n 的 IN 中断标志位（n=1 ~ 15）

0：端点 n 的 IN 中断无效

1：端点 n 的 IN 中断有效

在软件读取 INTRIN1/INTRIN2 寄存器后，硬件将自动清除 INTRIN1/INTRIN2 中的所有的中断标志

31.2.4 USB 端点 OUT 中断标志位（INTROUT1/2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTROUT1	04H	EP7OUTIF	EP6OUTIF	EP5OUTIF	EP4OUTIF	EP3OUTIF	EP2OUTIF	EP1OUTIF	-
INTROUT2	05H	EP15OUTIF	EP14OUTIF	EP13OUTIF	EP12OUTIF	EP11OUTIF	EP10OUTIF	EP9OUTIF	EP8OUTIF

EPnOUTIF：端点 n 的 OUT 中断标志位（n=1 ~ 15）

0：端点 n 的 OUT 中断无效

1：端点 n 的 OUT 中断有效

在软件读取 INTROUT1/INTROUT2 寄存器后，硬件将自动清除 INTROUT1/INTROUT2 中的所有的中断标志

31.2.5 USB 电源中断标志（INTRUSB）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTRUSB	06H	-	-	-	-	SOFIF	RSTIF	RSUIF	SUSIF

SOFIF：USB 帧起始信号中断标志

0：USB 帧起始信号中断无效

1：USB 帧起始信号中断有效

RSTIF：USB 复位信号中断标志

0：USB 复位信号中断无效

1：USB 复位信号中断有效

RSUIF：USB 恢复信号中断标志

0：USB 恢复信号中断无效

1：USB 恢复信号中断有效

SUSIF：USB 挂起信号中断标志

0：USB 挂起信号中断无效

1：USB 挂起信号中断有效

在软件读取 INTRUSB 寄存器后，硬件将自动清除 INTRUSB 中的所有的中断标志

31.2.6 USB 端点 IN 中断允许寄存器（INTRINnE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTRIN1E	07H	EP7INIE	EP6INIE	EP5INIE	EP4INIE	EP3INIE	EP2INIE	EP1INIE	EP0IE
INTRIN2E	08H	EP15INIE	EP14INIE	EP13INIE	EP12INIE	EP11INIE	EP10INIE	EP9INIE	EP8INIE

EP0IE：端点 0 的 IN/OUT 中断控制位

0：禁止端点 0 的 IN/OUT 中断

1：允许端点 0 的 IN/OUT 中断

EPnINIE：端点 n 的 IN 中断控制位（n=1 ~ 15）

0：禁止端点 n 的 IN 中断

1：允许端点 n 的 IN 中断

31.2.7 USB 端点 OUT 中断允许寄存器（INTROUT1/2E）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTROUT1E	09H	EP7OUTIE	EP6OUTIE	EP5OUTIE	EP4OUTIE	EP3OUTIE	EP2OUTIE	EP1OUTIE	-
INTROUT2E	0AH	EP15OUTIE	EP14OUTIE	EP13OUTIE	EP12OUTIE	EP11OUTIE	EP10OUTIE	EP9OUTIE	EP8OUTIE

EPnOUTIE：端点 n 的 OUT 中断控制位（n=1 ~ 15）

0：禁止端点 n 的 OUT 中断

1：允许端点 n 的 OUT 中断

31.2.8 USB 电源中断允许寄存器（INTRUSBE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INTRUSBE	0BH	-	-	-	-	SOFIE	RSTIE	RSUIE	SUSIE

SOFIE：USB 帧起始信号中断控制位

0：禁止 USB 帧起始信号中断

1：允许 USB 帧起始信号中断

RSTIE：USB 复位信号中断控制位

0：禁止 USB 复位信号中断

1：允许 USB 复位信号中断

RSUIE：USB 恢复信号中断控制位

0：禁止 USB 恢复信号中断

1：允许 USB 恢复信号中断

SUSIE：USB 挂起信号中断控制位

0：禁止 USB 挂起信号中断

1：允许 USB 挂起信号中断

31.2.9 USB 数据帧号寄存器（FRAMEn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
FRAME1	0CH	FRAME[7:0]							
FRAME2	0DH	-	-	-	-	-	FRAME[10:8]		

FRAME[10:0]：用于保存最后接收到的数据帧的 11 位帧号

31.2.10 USB 端点索引寄存器（INDEX）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INDEX	0EH	-	-	-	-	INDEX[3:0]			

INDEX[2:0]：选择 USB 端点

INDEX[3:0]	目标端点	INDEX[3:0]	目标端点
0000	端点 0	1000	端点 8
0001	端点 1	1001	端点 9
0010	端点 2	1010	端点 10
0011	端点 3	1011	端点 11
0100	端点 4	1100	端点 12
0101	端点 5	1101	端点 13
0110	端点 6	1110	端点 14
0111	端点 7	1111	端点 15

31.2.11 IN 端点的最大数据包大小（INMAXP）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INMAXP	10H	INMAXP[7:0]							

INMAXP[7:0]：设置 USB 的 IN 端点最大数据包的大小（**注意：数据包是以 8 字节为单位。例如若需要将端点的数据包设置为 64 字节，则需要将此寄存器设置为 8**）

当需要获取/设置此信息时，首先必须使用 INDEX 来选择目标端点 0 ~ 15

31.2.12 USB 端点 0 控制状态寄存器（CSR0）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CSR0	11H	SSUEND	SOPRDY	SDSTL	SUEND	DATEND	STSTL	IPRDY	OPRDY

SSUEND（Serviced Setup End）

SETUP 结束事件处理完成标志。当处理完成 SETUP 结束事件（SUEND）后，软件需要设置 SSUEND 标志位，硬件检测到 SSUEND 被写入“1”时自动将 SUEND 位清“0”。

SOPRDY（Serviced OPRDY）

OPRDY 事件处理完成标志。当处理完成从端点 0 接收到的数据包后，软件需要设置 SOPRDY 标志位，硬件检测到 SOPRDY 被写入“1”时自动将 OPRDY 位清“0”。

SDSTL（Send Stall）

当接收到错误的条件或者不支持的请求时，可以向此位写“1”来结束当前的数据传输。当 STALL 信号被发送后，硬件自动将此位清“0”。

SUEND（Setup End）

SETUP 安装包结束标志。当一次控制传输在软件向 DATAEND 位写“1”之前结束时，硬件将此只读位置“1”。当软件向 SSUEND 写‘1’后，硬件将该位清“0”。

DATEND（Data End）

数据结束。软件应在下列情况下将此位写“1”：

1、当发送最后一个数据包后，固件向 IPRDY 写“1”时；

2、当发送一个零长度数据包后，固件向 IPRDY 写“1”时；

3、当接收完最后一个数据包后，固件向 SOPRDY 写“1”时；

该位将被硬件自动清“0”

STSTL（Sent Stall）

STALL 信号发送完成标志。发送完成 STALL 信号后，硬件将该位置“1”。该位必须用软件清“0”。

IPRDY（IN Packet Ready）

IN 数据包准备完成标志。软件应在将一个要发送的数据包装入到端点 0 的 FIFO 后，将该位置“1”。在发生下列条件之一时，硬件将该位清“0”：

1、数据包已发送时；

2、数据包被一个 SETUP 包覆盖时；

3、数据包被一个 OUT 包覆盖时；

OPRDY（OUT Packet Ready）

OUT 数据包准备完成标志。当收到一个 OUT 数据包时，硬件将该只读位置“1”，并产生中断。该位只在软件向 SOPRDY 位写“1”时才被清“0”。

当需要获取/设置此信息时，首先必须使用 INDEX 来选择目标端点 0

31.2.13 IN 端点控制状态寄存器 1（INCSR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INCSR1	11H	-	CLRDT	STSTL	SDSTL	FLUSH	UNDRUN	FIFONE	IPRDY

CLRDT（Clear Data Toggle）：复位 IN 数据切换位。

当 IN 端点由于被重新配置或者被 STALL 而需要将数据切换位复位到“0”时，软件需要向此数据位写“1”。

STSTL（Sent Stall）：STALL 信号发送完成标志。

当 STALL 信号被发送完成后，硬件会将该位置“1”（此时 FIFO 被清空，IPRDY 位被清“0”）。该标志必须用软件清“0”。

SDSTL（Send Stall）：STALL 信号发送请求位。

软件应向该位写“1”以产生 STALL 信号作为对一个 IN 令牌的应答。软件应向该位写“0”以结束 STALL 信号。该位对 ISO 方式没有影响。

FLUSH（FIFO Flush）：清除 IN 端点的 FIFO 的下一个数据包。

向该位写“1”将从 IN 端点 FIFO 中清除待发送的下一个数据包。FIFO 指针被复位，IPRDY 位被清“0”。如果 FIFO 中包含多个数据包，软件必须对每个数据包向 FLUSH 写“1”。当 FIFO 清空完成后，硬件将 FLUSH 位清“0”。

UNDRUN（Data Underrun）：数据不足。

该位的功能取决于 IN 端点的方式：

ISO 方式：在 IPRDY 为“0”且收到一个 IN 令牌后发送了一个零长度数据包时，该位被置“1”。

中断/批量方式：当使用 NAK 作为对一个 IN 令牌的应答时，该位被置“1”。

该位必须用软件清“0”。

FIFONE（FIFO Not Empty）：IN 端点的 FIFO 非空标志

0：IN 端点的 FIFO 为空

1：IN 端点的 FIFO 包含有一个或者多个数据包

IPRDY（IN Packet Ready）：IN 数据包准备完成标志。

软件应在将一个要发送的数据包装入到端点的 FIFO 后，将该位置“1”。在发生下列条件之一时，硬件将该位清“0”：

1、数据包已发送时；

2、自动设置被使能（AUTOSET =‘1’）且端点 IN 的 FIFO 数据包达到 INMAXP 所设置的值；

3、如果端点处于同步方式且 ISOUD 为“1”，在收到下一个 SOF 之前 IPRDY 的读出值总是为 0。

需要获取/设置此信息时，首先必须使用 INDEX 来选择目标端点 1 ~ 15

31.2.14 IN 端点控制状态寄存器 2（INCSR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
INCSR2	12H	AUTOSET	ISO	MODE	ENDMA	FCDT	-	-	-

AUTOSET：自动设置 IPRDY 标志控制位。

0：禁止自动设置 IPRDY 标志

1：使能自动设置 IPRDY（必须向 IN FIFO 中装载的数据达到 INMAXP 所设置的值，否则 IPRDY 标志必须手动设置）

ISO：同步传输使能。

0：端点被配置为批量/中断传输方式

1：端点被配置为同步传输方式

MODE：端点方向选择位。

0：选择端点方向为 OUT

1：选择端点方向为 IN

~~ENDMA：IN 端点的 DMA 控制~~

~~0：禁止 IN 端点的 DMA 请求~~

~~1：使能 IN 端点的 DMA 请求~~

FCDT：强制 DATA0/DATA1 数据切换设置。

0：端点数据只在发送完一个数据包后且收到 ACK 时切换。

1：端点数据在每发送完一个数据包后被强制切换，不管是否收到 ACK。

需要获取/设置此信息时，首先必须使用 INDEX 来选择目标端点 1 ~ 15

31.2.15 OUT 端点的最大数据包大小（OUTMAXP）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
OUTMAXP	13H	OUTMAXP[7:0]							

OUTMAXP[7:0]：设置 USB 的 OUT 端点最大数据包的大小（**注意：数据包是以 8 字节为单位。例如若需要将端点的数据包设置为 64 字节，则需要将此寄存器设置为 8**）

需要获取/设置此信息时，首先必须使用 INDEX 来选择目标端点 1 ~ 15

31.2.16 OUT 端点控制状态寄存器 1（OUTCSR1）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
OUTCSR1	14H	CLRDT	STSTL	SDSTL	FLUSH	DATERR	OVRRUN	FIFOFUL	OPRDY

CLRDT（Clear Data Toggle）：复位 OUT 数据切换位。

当 OUT 端点由于被重新配置或者被 STALL 而需要将数据切换位复位到“0”时，软件需要向此数据位写“1”。

STSTL（Sent Stall）：STALL 信号发送完成标志。

当 STALL 信号被发送完成后，硬件会将该位置“1”。该标志必须用软件清“0”。

SDSTL（Send Stall）：STALL 信号发送请求位。

软件应向该位写“1”以产生 STALL 信号作为对一个 OUT 令牌的应答。软件应向该位写“0”以结束 STALL 信号。该位对 ISO 方式没有影响。

FLUSH（FIFO Flush）：清除 OUT 端点的 FIFO 的下一个数据包。

向该位写“1”将从 OUT 端点 FIFO 中清除下一个数据包。FIFO 指针被复位，OPRDY 位被清“0”。如果 FIFO 中包含多个数据包，软件必须对每个数据包向 FLUSH 写“1”。当 FIFO 清空完成后，硬件将 FLUSH 位清“0”。

DATAERR（Data Error）：数据错误。

在 ISO 方式，如果接收到的数据包有 CRC 或位填充错误，该位被硬件置‘1’。当软件清除 OPRDY 时，该位被清‘0’。该位只在 ISO 方式有效

OVRRUN（Data Overrun）：数据溢出。

当一个输入数据包不能被装入到 OUT 端点 FIFO 时，该位被硬件置“1”。该位只在 ISO 方式有效。该位必须用软件清“0”。

0：无数据溢出

1：自该标志最后一次被清除以来，因 FIFO 已满导致数据包丢失

FIFOFUL（FIFO Full）：OUT 端点的 FIFO 数据满标志。

0：OUT 端点的 FIFO 未满

1：OUT 端点的 FIFO 已满

OPRDY（OUT Packet Ready）：OUT 数据包接收完成标志。

当有数据包可用时硬件将该位置“1”。软件应在将每个数据包从 OUT 端点 FIFO 卸载后将该位清‘0’。

需要获取/设置此信息时，首先必须使用 INDEX 来选择目标端点 1 ~ 15

31.2.17 OUT 端点控制状态寄存器 2（OUTCSR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
OUTCSR2	15H	AUTOCLR	ISO	ENDMA	DMAMD	-	-	-	-

AUTOCLR：自动清除 OPRDY 标志控制位。

0：禁止自动清除 OPRDY 标志

1：使能自动清除 OPRDY（必须从 OUT FIFO 中下载的数据达到 OUTMAXP 所设置的值，否则 OPRDY 标志必须手动清除）

ISO：同步传输使能。

0：端点被配置为批量/中断传输方式

1：端点被配置为同步传输方式

ENDMA：OUT 端点的 DMA 控制

0：禁止 OUT 端点的 DMA 请求

1：使能 OUT 端点的 DMA 请求

DMAMD：设置 OUT 端点的 DMA 模式

需要获取/设置此信息时，首先必须使用 INDEX 来选择目标端点 1 ~ 15

31.2.18 USB 端点 0 的 OUT 长度（COUNT0）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
COUNT0	16H	-	OUTCNT0[6:0]						

OUTCNT0[6:0]：端点 0 的 OUT 字节长度

COUNT0 专用于保存端点 0 最后接收到的 OUT 数据包的数据长度（由于端点 0 数据包最长只能为 64 字节，所以只需要 7 位）。此长度值只在端点 0 的 OPRDY 位为“1”时才有效。

需要获取此长度信息时，首先必须使用 INDEX 来选择目标端点 0

31.2.19 USB 端点的 OUT 长度（OUTCOUNTn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
OUTCOUNT1	16H	OUTCNT[7:0]							
OUTCOUNT2	17H	-	-	-	-	-	OUTCNT[10:8]		

OUTCNT[10:0]：端点的 OUT 字节长度

OUTCOUNT1 和 OUTCOUNT2 联合组成一个 11 位的数，保存最后 OUT 数据包的数据长度，适用于端点 1 ~ 15。此长度值只在端点 1 ~ 15 的 OPRDY 位为“1”时才有效。

需要获取此长度信息时，首先必须使用 INDEX 来选择目标端点 1 ~ 15

31.2.20 USB 端点的 FIFO 数据访问寄存器（FIFOn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
FIFO0	20H	FIFO0[7:0]							
FIFO1	21H	FIFO1[7:0]							
FIFO2	22H	FIFO2[7:0]							
FIFO3	23H	FIFO3[7:0]							
FIFO4	24H	FIFO4[7:0]							
FIFO5	25H	FIFO5[7:0]							
FIFO6	26H	FIFO6[7:0]							
FIFO7	27H	FIFO7[7:0]							
FIFO8	28H	FIFO8[7:0]							
FIFO9	29H	FIFO9[7:0]							
FIFO10	2AH	FIFO10[7:0]							
FIFO11	2BH	FIFO11[7:0]							
FIFO12	2CH	FIFO12[7:0]							
FIFO13	2DH	FIFO13[7:0]							
FIFO14	2EH	FIFO14[7:0]							
FIFO15	2FH	FIFO15[7:0]							

FIFOn[7:0]：USB 各个端点的 IN/OUT 数据间接访问寄存器

31.2.21 USB 跟踪控制寄存器（UTRKCTL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UTRKCTL	30H	FTM1	FTM0	INTV[1:0]		ENST5	RES[2:0]		

FTM1：HFOSC 微调控制位

0：硬件使用粗调和微调为调整频率

1：硬件仅使用微调位调整 HFOSC

FTM0：FTM1 为 0 时有效

0：UTRK 使用全部 128 级微调

1：UTRK 禁止使用最大及最小 12 级之微调

INTV[1:0]：选择 UTRK 更新周期

INTV[1:0]	周期
00	2ms
01	4ms
10	8ms
11	16ms

ENST5：使能加 5 阶和减 5 阶

0：校准上下限为 10%

1：校准上下限为 20%

RES[2:0]：UTRK 自动调节分辨率设置

RES[2:0]	分辨率	调节值
000	8	0.067%
001	12	0.100%
010	16	0.133%
011	**24**	**0.200%**
100	28	0.233%
101	32	0.267%
110	48	0.4%
111	64	0.5%

31.2.22 USB 跟踪状态寄存器（UTRKSTS）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
UTRKSTS	31H	INTVCNT[3:0]				STS[1:0]		TST_UTRK	UTRK_RDY

INTVCNT[3:0]：USB 跟踪时的内部硬件计数器（软件只读）

STS[1:0]：UTRK 状态

STS[1:0]	状态
00	**INC 5**
01	DEC 5
10	INC 1
11	DEC 1

TST_UTRK：UTRK 测试模式控制

0：禁止 UTRK 测试模式

1：使能 UTRK 测试模式

UTRK_RDY：UTRK 校准完成状态位

31.3 USB 的 VID / PID，使用我们的 VID，可以向我们申请 PID

每个 USB 产品都必须有自己唯一的 VID&PID 组合，才能被电脑正确识别。若两个不同的 USB 产品对应的 VID&PID 组合相同，则可能出现电脑对 USB 产品的识别出现异常，从而无法正常使用 USB 产品。为避免出现这种情况，VID 和 PID 均需通过正规途径进行统一规划和分配。

目前已通过 USB-IF 组织取得了专用 USB 设备的 VID 编号 13503（十六进制：34BF）。客户使用 USB 芯片开发自己的 USB 产品时，若您已通过其它途径获取了您自己的 VID，则相应的 PID 可自行规划。若您的 USB 产品需要使用官方 VID，则产品的 PID 务必请您向我司申请。

31.4 如何软复位到系统区启动【USB 直接下载，不管 P3.2】

如果当前用户程序正在运行 USB,
请先【在用户区先关闭 USB】
再软复位到系统区，就是等待 USB 下载

```
CLKSEL = 0x00;          //切换为默认系统时钟
USBCON =0x00;           //清除USB设置
USBCLK &= 0xF0;
IRC48MCR = 0x00;        //停止48M IRC时钟

delay_ms(10);           //等待USB总线复位

IAP_CONTR =0x60;        //触发软件复位
```

31.5 使用 USB 时，CPU 如何跑 100MHz ~ 120MHz

使用官网最新版本的 USB 库文件，如果使用 USB 开放源码的话，在以下两个函数里面加 NOP

```
60  }
61  BYTE usb_read_reg(BYTE addr)
62  {
63    BYTE dat;
64
65    while (USBADR & 0x80);
66    USBADR = addr | 0x80;
67    NOP(30);
68    dat = USBDAT;
69
70    return dat;
71  }
72
73  void usb_write_reg(BYTE addr, BYTE dat)
74  {
75    while (USBADR & 0x80);
76    USBADR = addr & 0x7f;
77    NOP(30);
78    USBDAT = dat;
79  }
```

具体参考例子：STC32G144K246 要注意啥，使用 USB@120MHz，使用 CAN-FD@100MHz ~ 120MHz
https://www.stcaimcu.com/thread-22801-1-1.html

32 TFT 彩屏接口接口（8/16 位 i8080/M6800 接口）

产品线	TFT 彩屏接口
STC32G144K246 系列	●

STC32G144K246 系列的部分单片机内部集成了一个 TFT 彩屏接口控制器，可用于驱动目前流行的液晶显示屏模块。可驱动 i8080 接口和 M6800 接口彩屏，支持 8 位和 16 位数据宽度

32.1 TFT 彩屏接口功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LCMIFCFG	7EFE50H	LCMIFIE	-	LCMIFIP[1:0]		LCMIFDPS[1:0]		D16_D8	M68_I80
LCMIFCFG2	7EFE51H		LCMIFCPS[1:0]		SETUPT[2:0]			HOLDT[1:0]	

LCMIFCPS[1:0]：TFT 彩屏接口控制脚选择位

LCMIFCPS [1:0]	RS	i8080 的读信号 RD M6800 的使能信号 E	i8080 的写信号 WR M6800 的读写信号 RW
00	P4.5	P4.4	P4.2
01	P4.5	P3.7	P3.6
10	P4.0	P4.4	P4.2
11	P4.0	P3.7	P3.6

LCMIFDPS[1:0]：8 位数据位 LCM 接口数据脚选择位

LCMIFDPS [1:0]	D16_D8	数据字节 DAT[7:0]
00	0	P2[7:0]
01	0	P6[7:0]
10	0	P2[7:0]
11	0	P6[7:0]

LCMIFDPS[1:0]：16 位数据位 TFT 彩屏接口数据脚选择位

LCMIFDPS [1:0]	D16_D8	数据高字节 DAT[15:8]	数据低字节 DAT[7:0]
00	1	P2[7:0]	P0[7:0]
01	1	P6[7:0]	P2[7:0]
10	1	P2[7:0]	{P0[7:4],P4[7],P4[6],P4[3],P4[1]}
11	1	P6[7:0]	P7[7:0]

32.2 TFT 彩屏相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
LCMIFCFG	TFT 彩屏接口配置寄存器	7EFE50H	LCMIFIE	-	LCMIFIP[1:0]		LCMIFDPS[1:0]		D16_D8	M68_I80	0x00,0000
LCMIFCFG2	TFT 彩屏接口配置寄存器 2	7EFE51H	-	LCMIFCPS[1:0]		SETUPT[2:0]			HOLDT[1:0]		x000,0000
LCMIFCR	TFT 彩屏接口控制寄存器	7EFE52H	ENLCMIF	-	-	ST_ENDIAN	-	CMD[2:0]			0xx0,x000
LCMIFSTA	TFT 彩屏接口状态寄存器	7EFE53H	-	-	-	-	-	-	-	LCMIFIF	xxxx,xxx0
LCMIFDATL	TFT 彩屏接口低字节数据	7EFE54H	LCMIFDAT[7:0]								0000,0000
LCMIFDATH	TFT 彩屏接口高字节数据	7EFE55H	LCMIFDAT[15:8]								0000,0000
LCMIFPSCR	TFT 彩屏接口时钟预分频	7EFE56H	LCMIFPSCR[7:0]								0000,0000

32.2.1 TFT 彩屏接口配置寄存器（LCMIFCFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LCMIFCFG	7EFE50H	LCMIFIE	-	LCMIFIP[1:0]		LCMIFDPS[1:0]		D16_D8	M68_I80

LCMIFIE：TFT 彩屏接口中断使能控制位

0：禁止 TFT 彩屏接口中断

1：允许 TFT 彩屏接口中断

LCMIFIP[1:0]：TFT 彩屏接口中断优先级控制位

LCMIFIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

LCMIFDPS[1:0]：TFT 彩屏接口数据脚选择位

LCMIFDPS [1:0]	D16_D8	数据高字节 DAT[15:8]	数据低字节 DAT[7:0]
00	0	N/A	P2[7:0]
01	0	N/A	P6[7:0]
10	0	N/A	P2[7:0]
11	0	N/A	P6[7:0]
00	1	P2[7:0]	P0[7:0]
01	1	P6[7:0]	P2[7:0]
10	1	P2[7:0]	{P0[7:4],P4[7],P4[6],P4[3],P4[1]}
11	1	P6[7:0]	P7[7:0]

D16_D8：TFT 彩屏接口数据宽度控制位

0：8 位数据宽度

1：16 位数据宽度

M68_I80：TFT 彩屏接口模式选择位

0：i8080 模式

1：M6800 模式

32.2.2 TFT 彩屏接口配置寄存器 2（LCMIFCFG2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LCMIFCFG2	7EFE51H	-	LCMIFCPS[1:0]		SETUPT[2:0]			HOLDT[1:0]	

LCMIFCPS[1:0]：TFT 彩屏接口控制脚选择位

LCMIFCPS [1:0]	RS	i8080 的读信号 RD M6800 的使能信号 E	i8080 的写信号 WR M6800 的读写信号 RW
00	P4.5	P4.4	P4.2
01	P4.5	P3.7	P3.6
10	P4.0	P4.4	P4.2
11	P4.0	P3.7	P3.6

SETUPT[2:0]：TFT 彩屏接口通讯的数据建立时间控制位（详见后续章节的时序图）

HOLDT[1:0]：TFT 彩屏接口通讯的数据保持时间控制位（详见后续章节的时序图）

32.2.3 TFT 彩屏接口控制寄存器（LCMIFCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LCMIFCR	7EFE52H	ENLCMIF	-	-	ST_ENDIAN	-	CMD[2:0]		

ENLCMIF：TFT 彩屏接口使能控制位

0：禁止 TFT 彩屏接口功能

1：允许 TFT 彩屏接口功能

ST_ENDIAN：TFT_DMA 从 TFT 彩屏接口读取的 16 位命令或数据存储到 XRAM 的大小端顺序控制

0：大端模式（Big-Endian），即低地址存放高字节数据，高地址存放低字节数据

1：小端模式（Little-Endian），即低地址存放低字节数据，高地址存放高字节数据

注：此位仅在 DMA 对 TFT 进行读取操作时才有效，且必须是读取 16 位命令或数据（8 位数据不存在大小端的问题）

CMD[2:0]：TFT 彩屏接口触发命令

CMD[2:0]	触发命令
100	写命令
101	写数据
110	读命令/状态
111	读数据

32.2.4 TFT 彩屏接口状态寄存器（LCMIFSTA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LCMIFSTA	7EFE53H	-	-	-	-	-	-	-	LCMIFIF

LCMIFIF：TFT 彩屏接口中断请求标志，需软件清 0

32.2.5 TFT 彩屏接口数据寄存器（LCMIFDATL，LCMIFDATH）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LCMIFDATL	7EFE54H	LCMIFDAT[7:0]							
LCMIFDATH	7EFE55H	LCMIFDAT[15:8]							

LCMIFDAT：TFT 彩屏接口数据寄存器。

当数据宽度为 8 位数据时，只有 LCMDATL 数据有效；

当数据宽度为 16 位数据时，由 LCMDATL 和 LCMDATH 共同组合成 16 位数据

32.2.6 TFT 彩屏接口时钟预分频（LCMIFPSCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LCMIFPSCR	7EFE56H	LCMIFPSCR[7:0]							

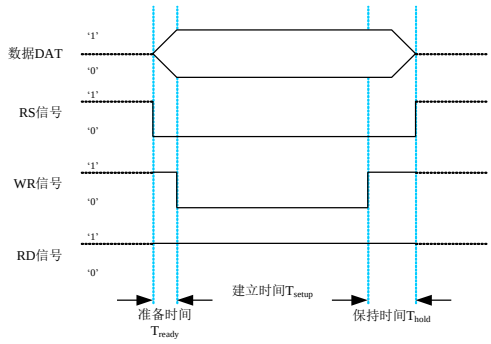
LCMIFPSCR[7:0]：LCMIF 预分频器。预分频器用于对系统时钟进行预分频，分频后的时钟提供给 LCMIF 模块。

LCMIF 时钟频率＝SYSclk /（LCMIFPSCR[7:0]+1）。

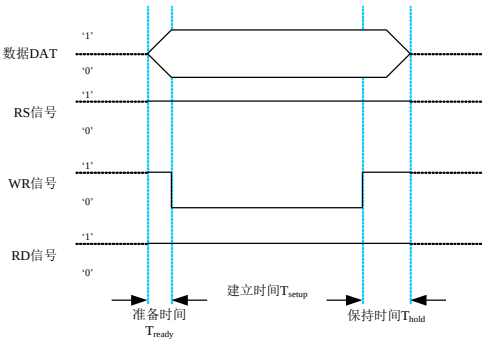
32.3 i8080/M6800 模式 TFT 彩屏接口时序图

注： T_{ready} ＝1 个系统时钟
T_{setup} ＝（SETUPT +1）个系统时钟
T_{hold} ＝（HOLDT+1）个系统时钟

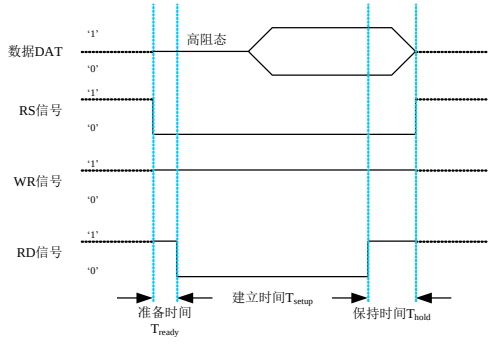
32.3.1 i8080 模式



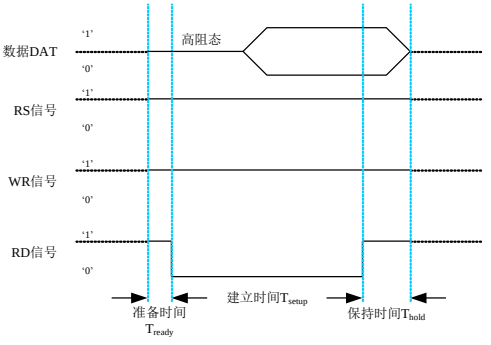
发送命令（CMD=100B）



发送数据（CMD=101B）

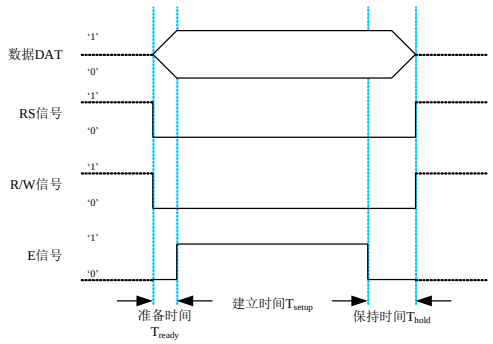


读取命令（CMD=110B）

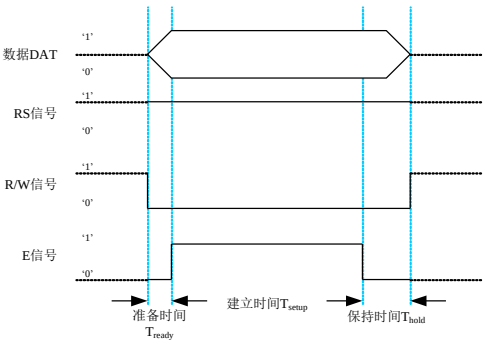


读取数据（CMD=111B）

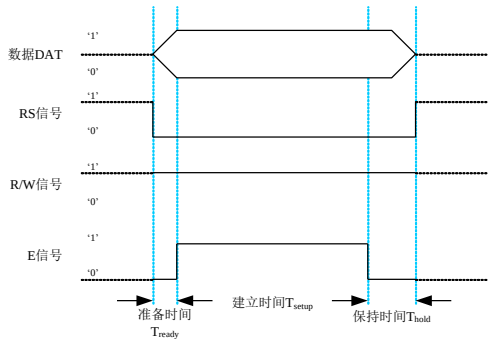
32.3.2 M6800 模式



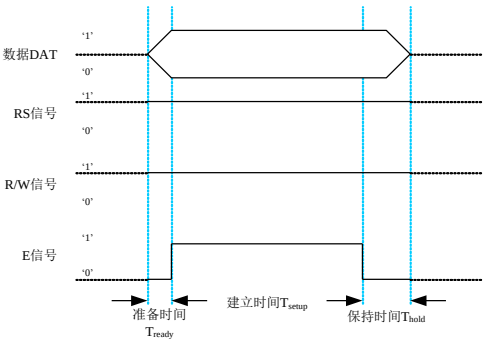
发送命令（CMD=100B）



发送数据（CMD=101B）



读取命令（CMD=110B）



读取数据（CMD=111B）

33 DMA，支持解放 CPU 的外设到外设传输（P2P）

产品线	DMA
STC32G144K246 系列	●

STC32G144K246 系列单片机支持批量数据存储功能，即传统的 DMA。

支持如下几种 DMA 操作：

- M2M_DMA：XRAM 存储器到 XRAM 存储器的数据读写
- ADC_DMA：自动扫描使能的 ADC 通道并将转换的 ADC 数据自动存储到 XRAM 中
- ADC2_DMA：自动扫描使能的 ADC2 通道并将转换的 ADC2 数据自动存储到 XRAM 中
- DAC_DMA：自动将 XRAM 中的数据通过 DAC 转换为模拟信号
- DAC2_DMA：自动将 XRAM 中的数据通过 DAC2 转换为模拟信号
- SPI_DMA：自动将 XRAM 中的数据和 SPI 外设之间进行数据交换
- SPI2_DMA：自动将 XRAM 中的数据和 SPI2 外设之间进行数据交换
- SPI3_DMA：自动将 XRAM 中的数据和 SPI3 外设之间进行数据交换
- UR1T_DMA：自动将 XRAM 中的数据通过串口 1 发送出去
- UR1R_DMA：自动将串口 1 接收到的数据存储到 XRAM 中
- UR2T_DMA：自动将 XRAM 中的数据通过串口 2 发送出去
- UR2R_DMA：自动将串口 2 接收到的数据存储到 XRAM 中
- UR3T_DMA：自动将 XRAM 中的数据通过串口 3 发送出去
- UR3R_DMA：自动将串口 3 接收到的数据存储到 XRAM 中
- UR4T_DMA：自动将 XRAM 中的数据通过串口 4 发送出去
- UR4R_DMA：自动将串口 4 接收到的数据存储到 XRAM 中
- UR5T_DMA：自动将 XRAM 中的数据通过串口 5 发送出去
- UR5R_DMA：自动将串口 5 接收到的数据存储到 XRAM 中
- UR6T_DMA：自动将 XRAM 中的数据通过串口 6 发送出去
- UR6R_DMA：自动将串口 6 接收到的数据存储到 XRAM 中
- UR7T_DMA：自动将 XRAM 中的数据通过串口 7 发送出去
- UR7R_DMA：自动将串口 7 接收到的数据存储到 XRAM 中
- UR8T_DMA：自动将 XRAM 中的数据通过串口 8 发送出去
- UR8R_DMA：自动将串口 8 接收到的数据存储到 XRAM 中
- TFT 彩屏 DMA：自动将 XRAM 中的数据和 TFT 彩屏设备之间进行数据交换
- I2CT_DMA：自动将 XRAM 中的数据通过 I2C 接口发送出去
- I2CR_DMA：自动将 I2C 接收到的数据存储到 XRAM 中
- I2C2T_DMA：自动将 XRAM 中的数据通过 I2C2 接口发送出去
- I2C2R_DMA：自动将 I2C2 接收到的数据存储到 XRAM 中
- I2ST_DMA：自动将 XRAM 中的数据通过 I2S 发送出去
- I2SR_DMA：自动将 I2S 接收到的数据存储到 XRAM 中
- I2S2T_DMA：自动将 XRAM 中的数据通过 I2S2 发送出去
- I2S2R_DMA：自动将 I2S2 接收到的数据存储到 XRAM 中
- QSPI_DMA：自动将 XRAM 中的数据和 QSPI 外设之间进行数据交换
- PWMAT_DMA：自动将 XRAM 中的数据通过 PWMA 接口发送出去
- PWMAR_DMA：自动将 PWMA 接收到的数据存储到 XRAM 中

- PWMCT_DMA：自动将 XRAM 中的数据通过 PWMC 接口发送出去
- PWMCR_DMA：自动将 PWMC 接收到的数据存储到 XRAM 中
- PWMET_DMA：自动将 XRAM 中的数据通过 PWME 接口发送出去
- PWMER_DMA：自动将 PWME 接收到的数据存储到 XRAM 中
- CANT_DMA：自动将 XRAM 中的数据通过 CAN-FD 接口发送出去
- CANR_DMA：自动将 CAN-FD 接收到的数据存储到 XRAM 中
- CAN2T_DMA：自动将 XRAM 中的数据通过 CAN-FD2 接口发送出去
- CAN2R_DMA：自动将 CAN-FD2 接收到的数据存储到 XRAM 中
- P2P：自动将源外设接收到的数据透传到目标外设（支持两组 P2P）

每次 DMA 数据传输最大数据量为 65535 字节,即将 AMT 设置为 0xFFFE；当将 AMT 设置为 0xFFFF 时，则 DMA 为**无限循环模式**。（存储器到存储器的 DMA 没有无限循环模式）

每种 DMA 对 XRAM 的读写操作都可设置 4 级访问优先级，硬件自动进行 XRAM 总线的访问仲裁，不会影响 CPU 的 XRAM 的访问。

33.1 DMA 使用注意事项

将 CKCON 设置为 0 时，xdata 访问速度设置为最快，写 xdata 为 2T，读 xdata 为 3T。此时如果硬件 DMA 和 CPU 同时访问 xdata，会出现总线竞争，需要进行总线仲裁。由于芯片内部的仲裁电路设计不够完美，会导致 CPU 访问 xdata 出错。

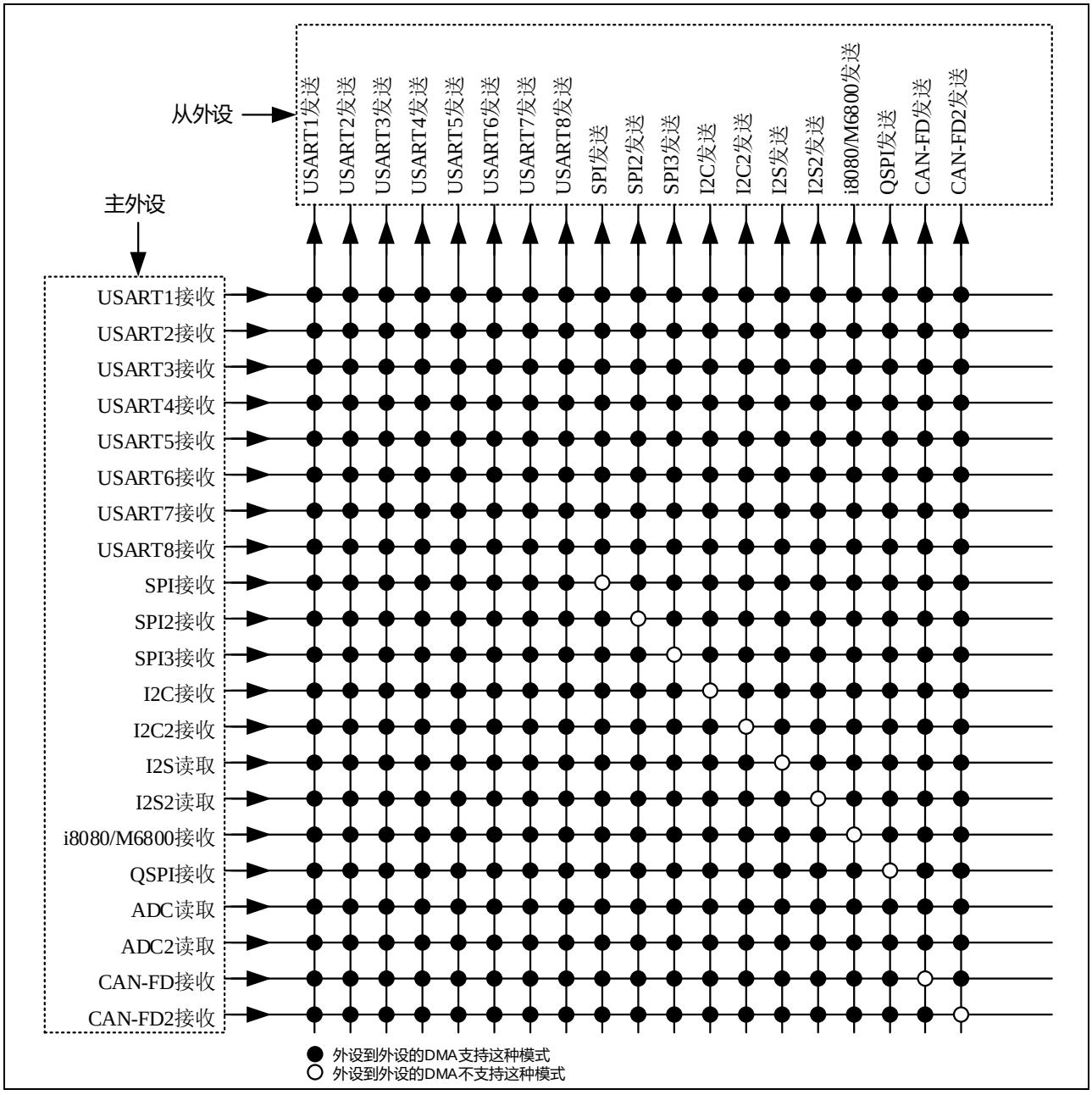
将 CKCON 设置为 1，可完美解决硬件 DMA 和 CPU 同时访问 xdata 出现的仲裁错误问题。CKCON 为 1 时，xdata 的访问时间会多 1T，写 xdata 为 3T，读 xdata 为 4T

解决硬件 DMA 时 CPU 访问 xdata 出错的问题的解决方案（下面两个方案任选其一即可）：

1、将 CKCON 设置为 1。此方案可完美解决问题，但访问 xdata 会多 1 个时钟

2、当硬件 DMA 访问 xdata 过程中，用户程序不得访问 xdata。用户程序只能在 DMA 不工作时才能读写 DMA 在 xdata 中的缓冲区。

33.2 外设到外设（P2P）DMA 矩阵



33.3 DMA 存储器读写地址选择（DMA_ARB_CFG）

33.3.1 DMA 仲裁配置寄存器（DMA_ARB_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_ARB_CFG	7EFAF8H	WTRREN	XBANK[1:0]		STASEL[4:0]				

XBANK[1:0]：DMA 存储器读写地址选择（DMA 的 xdata 区域 BANK 选择）

XBANK[1:0]	存储器范围	地址说明
00	01:0000H ~ 01:FFFFH	80251 体系结构中的 xdata 区域 （使用关键字 xdata 或者 far 定义）
01	02:0000H ~ 02:FFFFH	80251 体系结构中的 hdata 区域 （使用关键字 far 定义）
10	03:0000H ~ 03:0FFFH	80251 体系结构中的 hdata 区域 （4K 可执行代码的 RAM 未使能时可用）
11	保留	

【特别注意】：由于启动 DMA 后，DMA 的数据传输是硬件在后台自动进行的，所以在 DMA 进行数据传输完成前，XBANK 的设定值必须保持不变。如需更改 XBANK 的值，必须等待当前所有启动的 DMA 完成后，才可以修改。

33.4 存储器与存储器之间的数据读写（M2M_DMA）

33.4.1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_M2M_CFG	M2M_DMA 配置寄存器	7EFA00H	M2MIE	-	TXACO	RXACO	M2MIP[1:0]		M2MPTY[1:0]		0x00,0000
DMA_M2M_CR	M2M_DMA 控制寄存器	7EFA01H	ENM2M	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_M2M_STA	M2M_DMA 状态寄存器	7EFA02H	-	-	-	-	-	-	-	M2MIF	xxxx,xxx0
DMA_M2M_AMTH	M2M_DMA 传输总字节数	7EFA80H	M2M_AMT[15:8]								0000,0000
DMA_M2M_AMT	M2M_DMA 传输总字节数	7EFA03H	M2M_AMT[7:0]								0000,0000
DMA_M2M_DONEH	M2M_DMA 传输完成字节数	7EFA81H	M2M_DONE[15:8]								0000,0000
DMA_M2M_DONE	M2M_DMA 传输完成字节数	7EFA04H	M2M_DONE[7:0]								0000,0000
DMA_M2M_TXAH	M2M_DMA 发送高地址	7EFA05H	M2M_TXA[15:8]								0000,0000
DMA_M2M_TXAL	M2M_DMA 发送低地址	7EFA06H	M2M_TXA[7:0]								0000,0000
DMA_M2M_RXAH	M2M_DMA 接收高地址	7EFA07H	M2M_RXA[15:8]								0000,0000
DMA_M2M_RXAL	M2M_DMA 接收低地址	7EFA08H	M2M_RXA[7:0]								0000,0000

33.4.2 M2M_DMA 配置寄存器（DMA_M2M_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_M2M_CFG	7EFA00H	M2MIE	-	TXACO	RXACO	M2MIP[1:0]		M2MPTY[1:0]	

M2MIE：M2M_DMA 中断使能控制位

0：禁止 M2M_DMA 中断

1：允许 M2M_DMA 中断

TXACO：M2M_DMA 源地址（读取地址）改变方向

0：数据读取完成后地址自动递增

1：数据读取完成后地址自动递减

RXACO：M2M_DMA 目标地址（写入地址）改变方向

0：数据写入完成后地址自动递增

1：数据写入完成后地址自动递减

M2MIP[1:0]：M2M_DMA 中断优先级控制位

M2MIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

M2MPTY[1:0]：M2M_DMA 数据总线访问优先级控制位

M2MPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.4.3 M2M_DMA 控制寄存器（DMA_M2M_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_M2M_CR	7EFA01H	ENM2M	TRIG	-	-	-	-	-	-

ENM2M：M2M_DMA 功能使能控制位

0：禁止 M2M_DMA 功能

1：允许 M2M_DMA 功能

TRIG：M2M_DMA 数据读写触发控制位

0：写 0 无效

1：写 1 开始 M2M_DMA 操作，

33.4.4 M2M_DMA 状态寄存器（DMA_M2M_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_M2M_STA	7EFA02H	-	-	-	-	-	-	-	M2MIF

M2MIF：M2M_DMA 中断请求标志位，当 M2M_DMA 操作完成后，硬件自动将 M2MIF 置 1，若使能 M2M_DMA 中断则进入中断服务程序。标志位需软件清零

33.4.5 M2M_DMA 传输总字节寄存器（DMA_M2M_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_M2M_AMTH	7EFA80H	M2M_AMT[15:8]							
DMA_M2M_AMT	7EFA03H	M2M_AMT[7:0]							

M2M_AMT[15:0]：设置需要进行数据读写的字节数。

注：实际读写的字节数为（M2M_AMT+1），即当 M2M_AMT 设置为 0 时，读写 1 字节，当 M2M_AMT 设置 255 时，读写 256 字节。特别注意 M2M_AMT 不能设置为 FFFFH。

33.4.6 M2M_DMA 传输完成字节寄存器（DMA_M2M_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_M2M_DONEH	7EFA81H	M2M_DONE[15:8]							
DMA_M2M_DONE	7EFA04H	M2M_DONE[7:0]							

M2M_DONE[15:0]：当前已经读写完成的字节数。

33.4.7 M2M_DMA 发送地址寄存器（DMA_M2M_TXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_M2M_TXAH	7EFA05H	M2M_TXA[15:8]							
DMA_M2M_TXAL	7EFA06H	M2M_TXA[7:0]							

M2M_TXA[15:0]：设置进行数据读写时的源地址。执行 M2M_DMA 操作时会从这个地址开始读数据。

33.4.8 M2M_DMA 接收地址寄存器（DMA_M2M_RXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_M2M_RXAH	7EFA07H	M2M_RXA[15:8]							
DMA_M2M_RXAL	7EFA08H	M2M_RXA[7:0]							

M2M_RXA[15:0]：设置进行数据读写时目标地址。执行 M2M_DMA 操作时会从这个地址开始写入数据。

33.5 ADC1/2 数据自动存储（ADCn_DMA）

33.5.1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_ADC_CFG	ADC_DMA 配置寄存器	7EFA10H	ADCIE	-	-	-	ADCMIP[1:0]		ADCPTY[1:0]		0xxx,0000
DMA_ADC_CR	ADC_DMA 控制寄存器	7EFA11H	ENADC	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_ADC_STA	ADC_DMA 状态寄存器	7EFA12H	-	-	-	-	-	-	-	ADCIF	xxxx,xxx0
DMA_ADC_AMTH	ADC_DMA 传输总字节数	7EFA82H	ADC_AMT[15:8]								0000,0000
DMA_ADC_AMT	ADC_DMA 传输总字节数	7EFA13H	ADC_AMT[7:0]								0000,0000
DMA_ADC_DONEH	ADC_DMA 传输完成字节数	7EFA83H	ADC_DONE[15:8]								0000,0000
DMA_ADC_DONE	ADC_DMA 传输完成字节数	7EFA14H	ADC_DONE[7:0]								0000,0000
DMA_ADC_RXAH	ADC_DMA 接收高地址	7EFA17H	ADC_RXA[15:8]								0000,0000
DMA_ADC_RXAL	ADC_DMA 接收低地址	7EFA18H	ADC_RXA[7:0]								0000,0000
DMA_ADC_CFG2	ADC_DMA 配置寄存器 2	7EFA19H	-	-	-	-	CVTIMESEL[3:0]				xxxx,0000
DMA_ADC_CHSW0	ADC_DMA 通道使能	7EFA1AH	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0	0000,0001
DMA_ADC_CHSW1	ADC_DMA 通道使能	7EFA1BH	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	1000,0000
DMA_ADC_ITVH	ADC_DMA 时间间隔寄存器	7EFA1EH	ADC_ITV[15:8]								0000,0000
DMA_ADC_ITVL	ADC_DMA 时间间隔寄存器	7EFA1FH	ADC_ITV[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_ADC2_CFG	ADC2_DMA 配置寄存器	7EFAE0H	ADC2IE	-	-	-	ADC2MIP[1:0]		ADC2PTY[1:0]		0xxx,0000
DMA_ADC2_CR	ADC2_DMA 控制寄存器	7EFAE1H	ENADC2	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_ADC2_STA	ADC2_DMA 状态寄存器	7EFAE2H	-	-	-	-	-	-	-	ADC2IF	xxxx,xxx0
DMA_ADC2_CFG2	ADC2_DMA 配置寄存器 2	7EFAE3H	-	-	-	-	CVTIMESEL[3:0]				xxxx,0000
DMA_ADC2_AMTH	ADC2_DMA 传输总字节数	7EFAE4H	ADC2_AMT[15:8]								0000,0000
DMA_ADC2_AMT	ADC2_DMA 传输总字节数	7EFAE5H	ADC2_AMT[7:0]								0000,0000
DMA_ADC2_DONEH	ADC2_DMA 传输完成字节数	7EFAE6H	ADC2_DONE[15:8]								0000,0000
DMA_ADC2_DONE	ADC2_DMA 传输完成字节数	7EFAE7H	ADC2_DONE[7:0]								0000,0000
DMA_ADC2_RXAH	ADC2_DMA 接收高地址	7EFAE8H	ADC2_RXA[15:8]								0000,0000
DMA_ADC2_RXAL	ADC2_DMA 接收低地址	7EFAE9H	ADC2_RXA[7:0]								0000,0000
DMA_ADC2_CHSW0	ADC2_DMA 通道使能	7EFAEAH	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0	0000,0001
DMA_ADC2_CHSW1	ADC2_DMA 通道使能	7EFAEBH	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	1000,0000
DMA_ADC2_ITVH	ADC2_DMA 时间间隔寄存器	7EFAEEH	ADC2_ITV[15:8]								0000,0000
DMA_ADC2_ITVL	ADC2_DMA 时间间隔寄存器	7EFAEFH	ADC2_ITV[7:0]								0000,0000

33.5.2 ADCn_DMA 配置寄存器（DMA_ADCn_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_ADC_CFG	7EFA10H	ADCIE	-			ADCIP[1:0]		ADCPTY[1:0]	
DMA_ADC2_CFG	7EFAE0H	ADC2IE	-			ADC2IP[1:0]		ADC2PTY[1:0]	

ADCnIE：ADCn_DMA 中断使能控制位（n=1/2）

0：禁止 ADCn_DMA 中断

1：允许 ADCn_DMA 中断

ADCnIP[1:0]：ADCn_DMA 中断优先级控制位（n=1/2）

ADCnIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

ADCnPTY[1:0]：ADCn_DMA 数据总线访问优先级控制位（n=1/2）

ADCnPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.5.3 ADCn_DMA 控制寄存器（DMA_ADCn_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_ADC_CR	7EFA11H	ENADC	TRIG	-	-	-	-	-	-
DMA_ADC2_CR	7EFAE1H	ENADC2	TRIG2	-	-	-	-	-	-

ENADCn：ADCn_DMA 功能使能控制位（n=1/2）

0：禁止 ADCn_DMA 功能

1：允许 ADCn_DMA 功能

TRIGn：ADCn_DMA 操作触发控制位（n=1/2）

0：写 0 无效

1：写 1 开始 ADCn_DMA 操作，

33.5.4 ADCn_DMA 状态寄存器（DMA_ADCn_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_ADC_STA	7EFA12H	-	-	-	-	-	-	-	ADCIF
DMA_ADC2_STA	7EFAE2H	-	-	-	-	-	-	-	ADC2IF

ADCnIF：ADCn_DMA 中断请求标志位，当 ADCn_DMA 完成扫描所有使能的 ADCn 通道后后，硬件自动将 ADCnIF 置 1，若使能 ADCn_DMA 中断则进入中断服务程序。标志位需软件清零（n=1/2）

33.5.5 ADCn_DMA 循环扫描总次数寄存器（DMA_ADCn_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_ADC_AMTH	7EFA82H	ADC_AMT[15:8]							
DMA_ADC_AMT	7EFA13H	ADC_AMT[7:0]							
DMA_ADC2_AMTH	7EFAE4H	ADC2_AMT[15:8]							
DMA_ADC2_AMT	7EFAE5H	ADC2_AMT[7:0]							

ADCn_AMT[15:0]：设置需要循环扫描 ADCn 通道的总次数。（n=1/2）

注：将 DMA_ADCn_CHSW0 和 DMA_ADCn_CHSW1 所使能的 ADCn 通道全部扫描一遍为一次。实际扫描的次数为（ADCn_AMT+1），即当 ADCn_AMT 设置为 0 时，扫描 1 次，当 ADCn_AMT 设置 255 时，扫描 256 次。特别注意 ADCn_AMT 设置为 FFFFH 时表示无限循环扫描模式。

33.5.6 ADCn_DMA 完成扫描次数寄存器（DMA_ADCn_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_ADC_DONEH	7EFA83H	ADC_DONE[15:8]							
DMA_ADC_DONE	7EFA14H	ADC_DONE[7:0]							
DMA_ADC2_DONEH	7EFAE6H	ADC2_DONE[15:8]							
DMA_ADC2_DONE	7EFAE7H	ADC2_DONE[7:0]							

ADCn_DONE[15:0]：当前已经完成 ADC 扫描的次数。（n=1/2）

33.5.7 ADCn_DMA 接收地址寄存器（DMA_ADCn_RXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_ADC_RXAH	7EFA17H	ADC_RXA[15:8]							
DMA_ADC_RXAL	7EFA18H	ADC_RXA[7:0]							
DMA_ADC2_RXAH	7EFAE8H	ADC2_RXA[15:8]							
DMA_ADC2_RXAL	7EFAE9H	ADC2_RXA[7:0]							

ADCn_RXA[15:00]：设置进行 ADCn_DMA 操作时 ADC 转换数据的存储地址。（n=1/2）

33.5.8 ADCn_DMA 配置寄存器 2（DMA_ADCn_CFG2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_ADC_CFG2	7EFA19H	-	-	-	-	CVTIMESEL[3:0]			
DMA_ADC2_CFG2	7EFAE3H	-	-	-	-	CVTIMESEL[3:0]			

CVTIMESEL[3:0]：设置进行 ADCn_DMA 操作时，对每个 ADCn 通道进行 ADC 转换的次数（n=1/2）

CVTIMESEL[3:0]	转换次数	CVTIMESEL[3:0]	转换次数
0xxx	1 次		
1000	2 次	1100	32 次
1001	4 次	1101	64 次
1010	8 次	1110	128 次
1011	16 次	1111	256 次

33.5.9 ADCn_DMA 通道使能寄存器（DMA_ADCn_CHSWx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_ADC_CHSW0	7EFA1AH	ADCCH7	ADCCH6	ADCCH5	ADCCH4	ADCCH3	ADCCH2	ADCCH1	ADCCH0
DMA_ADC_CHSW1	7EFA1BH	ADCCH15	ADCCH14	ADCCH13	ADCCH12	ADCCH11	ADCCH10	ADCCH9	ADCCH8
DMA_ADC2_CHSW0	7EFAEAH	ADC2CH7	ADC2CH6	ADC2CH5	ADC2CH4	ADC2CH3	ADC2CH2	ADC2CH1	ADC2CH0
DMA_ADC2_CHSW1	7EFAEBH	ADC2CH15	ADC2CH14	ADC2CH13	ADC2CH12	ADC2CH11	ADC2CH10	ADC2CH9	ADC2CH8

ADCnCHx：设置 ADCn_DMA 操作时，自动扫描的 ADCn 通道。通道扫描总是从编号小的通道开始。（n=1/2，x=0~15）

33.5.10 ADCn_DMA 转换周期（DMA_ADCn_ITVx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_ADC_ITVH	7EFA1EH	ADC_ITV[15:8]							
DMA_ADC_ITVL	7EFA1FH	ADC_ITV[7:0]							
DMA_ADC2_ITVH	7EFAEEH	ADC2_ITV[15:8]							
DMA_ADC2_ITVL	7EFAEFH	ADC2_ITV[7:0]							

ADCn_ITV[15:0]：设置 ADCn 转换的间隔时间（基于系统时钟）。（n=1/2）

33.5.11 ADC_DMA 的数据存储格式

注：ADC 转换速度和转换结果的对齐方式均由 ADC 相关寄存器进行设置

XRAM[DMA_ADC_RXA+0] = 使能的第 1 通道的第 1 次 ADC 转换结果的高字节;
XRAM[DMA_ADC_RXA+1] = 使能的第 1 通道的第 1 次 ADC 转换结果的低字节;
XRAM[DMA_ADC_RXA+2] = 使能的第 1 通道的第 2 次 ADC 转换结果的高字节;
XRAM[DMA_ADC_RXA+3] = 使能的第 1 通道的第 2 次 ADC 转换结果的低字节;
...
XRAM[DMA_ADC_RXA+2n-2] = 使能的第 1 通道的第 n 次 ADC 转换结果的高字节;
XRAM[DMA_ADC_RXA+2n-1] = 使能的第 1 通道的第 n 次 ADC 转换结果的低字节;
XRAM[DMA_ADC_RXA+2n] = 第 1 通道的 ADC 通道号;
XRAM[DMA_ADC_RXA+2n+1] = 第 1 通道 n 次 ADC 转换结果取完平均值之后的余数;
XRAM[DMA_ADC_RXA+2n+2] = 第 1 通道 n 次 ADC 转换结果平均值的高字节;
XRAM[DMA_ADC_RXA+2n+3] = 第 1 通道 n 次 ADC 转换结果平均值的低字节;

XRAM[DMA_ADC_RXA+(2n+4)+0] = 使能的第 2 通道的第 1 次 ADC 转换结果的高字节;
XRAM[DMA_ADC_RXA+(2n+4)+1] = 使能的第 2 通道的第 1 次 ADC 转换结果的低字节;
XRAM[DMA_ADC_RXA+(2n+4)+2] = 使能的第 2 通道的第 2 次 ADC 转换结果的高字节;
XRAM[DMA_ADC_RXA+(2n+4)+3] = 使能的第 2 通道的第 2 次 ADC 转换结果的低字节;
...
XRAM[DMA_ADC_RXA+(2n+4)+2n-2] = 使能的第 2 通道的第 n 次 ADC 转换结果的高字节;
XRAM[DMA_ADC_RXA+(2n+4)+2n-1] = 使能的第 2 通道的第 n 次 ADC 转换结果的低字节;
XRAM[DMA_ADC_RXA+(2n+4)+2n] = 第 2 通道的 ADC 通道号;
XRAM[DMA_ADC_RXA+(2n+4)+2n+1] = 第 2 通道的 n 次 ADC 转换结果取完平均值之后的余数;
XRAM[DMA_ADC_RXA+(2n+4)+2n+2] =第 2 通道的 n 次 ADC 转换结果平均值的高字节;
XRAM[DMA_ADC_RXA+(2n+4)+2n+3] =第 2 通道的 n 次 ADC 转换结果平均值的低字节;
...
XRAM[DMA_ADC_RXA+(m-1)(2n+4)+0] = 使能的第 m 通道的第 1 次 ADC 转换结果的高字节;
XRAM[DMA_ADC_RXA+(m-1)(2n+4)+1] = 使能的第 m 通道的第 1 次 ADC 转换结果的低字节;
XRAM[DMA_ADC_RXA+(m-1)(2n+4)+2] = 使能的第 m 通道的第 2 次 ADC 转换结果的高字节;
XRAM[DMA_ADC_RXA+(m-1)(2n+4)+3] = 使能的第 m 通道的第 2 次 ADC 转换结果的低字节;
...
XRAM[DMA_ADC_RXA+(m-1)(2n+4)+2n-2] = 使能的第 m 通道的第 n 次 ADC 转换结果的高字节;
XRAM[DMA_ADC_RXA+(m-1)(2n+4)+2n-1] = 使能的第 m 通道的第 n 次 ADC 转换结果的低字节;
XRAM[DMA_ADC_RXA+(m-1)(2n+4)+2n] = 第 m 通道的 ADC 通道号;
XRAM[DMA_ADC_RXA+(m-1)(2n+4)+2n+1] = 第 m 通道的 n 次 ADC 转换结果取完平均值之后的余数;
XRAM[DMA_ADC_RXA+(m-1)(2n+4)+2n+2] =第 m 通道的 n 次 ADC 转换结果平均值的高字节;
XRAM[DMA_ADC_RXA+(m-1)(2n+4)+2n+3] =第 m 通道的 n 次 ADC 转换结果平均值的低字节;

表格形式如下：

ADC 通道	偏移地址	数据
第 1 通道	0	使能的第 1 通道的第 1 次 ADC 转换结果的高字节
	1	使能的第 1 通道的第 1 次 ADC 转换结果的低字节
	2	使能的第 1 通道的第 2 次 ADC 转换结果的高字节
	3	使能的第 1 通道的第 2 次 ADC 转换结果的低字节
	...	...
	2n-2	使能的第 1 通道的第 n 次 ADC 转换结果的高字节
	2n-1	使能的第 1 通道的第 n 次 ADC 转换结果的低字节
	2n	第 1 通道的 ADC 通道号
	2n+1	第 1 通道 n 次 ADC 转换结果取完平均值之后的余数
	2n+2	第 1 通道 n 次 ADC 转换结果平均值的高字节
	2n+3	第 1 通道 n 次 ADC 转换结果平均值的低字节
第 2 通道	(2n+4) + 0	使能的第 2 通道的第 1 次 ADC 转换结果的高字节
	(2n+4) + 1	使能的第 2 通道的第 1 次 ADC 转换结果的低字节
	(2n+4) + 2	使能的第 2 通道的第 2 次 ADC 转换结果的高字节
	(2n+4) + 3	使能的第 2 通道的第 2 次 ADC 转换结果的低字节
	...	...
	(2n+4) + 2n-2	使能的第 2 通道的第 n 次 ADC 转换结果的高字节
	(2n+4) + 2n-1	使能的第 2 通道的第 n 次 ADC 转换结果的低字节
	(2n+4) + 2n	第 2 通道的 ADC 通道号
	(2n+4) + 2n+1	第 2 通道 n 次 ADC 转换结果取完平均值之后的余数
	(2n+4) + 2n+2	第 2 通道 n 次 ADC 转换结果平均值的高字节
	(2n+4) + 2n+3	第 2 通道 n 次 ADC 转换结果平均值的低字节
	...	...
第 m 通道	(m-1)(2n+4) + 0	使能的第 m 通道的第 1 次 ADC 转换结果的高字节
	(m-1)(2n+4) + 1	使能的第 m 通道的第 1 次 ADC 转换结果的低字节
	(m-1)(2n+4) + 2	使能的第 m 通道的第 2 次 ADC 转换结果的高字节
	(m-1)(2n+4) + 3	使能的第 m 通道的第 2 次 ADC 转换结果的低字节
	...	...
	(m-1)(2n+4) + 2n-2	使能的第 m 通道的第 n 次 ADC 转换结果的高字节
	(m-1)(2n+4) + 2n-1	使能的第 m 通道的第 n 次 ADC 转换结果的低字节
	(m-1)(2n+4) + 2n	第 m 通道的 ADC 通道号
	(m-1)(2n+4) + 2n+1	第 m 通道 n 次 ADC 转换结果取完平均值之后的余数
	(m-1)(2n+4) + 2n+2	第 m 通道 n 次 ADC 转换结果平均值的高字节
	(m-1)(2n+4) + 2n+3	第 m 通道 n 次 ADC 转换结果平均值的低字节

33.6 SPI1/2/3 与存储器之间的数据交换（SPIn_DMA）

33.6.1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_SPI_CFG	SPI_DMA 配置寄存器	7EFA20H	SPIIE	ACT_TX	ACT_RX	-	SPIIP[1:0]		SPIPTY[1:0]		000x,0000
DMA_SPI_CR	SPI_DMA 控制寄存器	7EFA21H	ENSPI	TRIG_M	TRIG_S	-	-	-	-	CLRFIFO	000x,xxx0
DMA_SPI_STA	SPI_DMA 状态寄存器	7EFA22H	-	-	-	-	-	TXOVW	RXLOSS	SPIIF	xxxx,x000
DMA_SPI_AMTH	SPI_DMA 传输总字节数	7EFA84H	SPI_AMT[15:8]								0000,0000
DMA_SPI_AMT	SPI_DMA 传输总字节数	7EFA23H	SPI_AMT[7:0]								0000,0000
DMA_SPI_DONEH	SPI_DMA 传输完成字节数	7EFA85H	SPI_DONE[15:8]								0000,0000
DMA_SPI_DONE	SPI_DMA 传输完成字节数	7EFA24H	SPI_DONE[7:0]								0000,0000
DMA_SPI_TXAH	SPI_DMA 发送高地址	7EFA25H	SPI_TXA[15:8]								0000,0000
DMA_SPI_TXAL	SPI_DMA 发送低地址	7EFA26H	SPI_TXA[7:0]								0000,0000
DMA_SPI_RXAH	SPI_DMA 接收高地址	7EFA27H	SPI_RXA[15:8]								0000,0000
DMA_SPI_RXAL	SPI_DMA 接收低地址	7EFA28H	SPI_RXA[7:0]								0000,0000
DMA_SPI_CFG2	SPI_DMA 配置寄存器 2	7EFA29H	-	-	-	-	-	WRPSS	SSS[1:0]		xxxx,x000
DMA_SPI_ITVH	SPI_DMA 时间间隔寄存器	7EFA2EH	SPI_ITV[15:8]								0000,0000
DMA_SPI_ITVL	SPI_DMA 时间间隔寄存器	7EFA2FH	SPI_ITV[7:0]								0000,0011

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_SPI2_CFG	SPI2_DMA 配置寄存器	7EF810H	SPI2IE	ACT_TX	ACT_RX	-	SPI2IP[1:0]		SPI2PTY[1:0]		000x,0000
DMA_SPI2_CR	SPI2_DMA 控制寄存器	7EF811H	ENSPI2	TRIG_M	TRIG_S	-	-	-	-	CLRFIFO	000x,xxx0
DMA_SPI2_STA	SPI2_DMA 状态寄存器	7EF812H	-	-	-	-	-	TXOVW	RXLOSS	SPI2IF	xxxx,x000
DMA_SPI2_CFG2	SPI2_DMA 配置寄存器 2	7EF813H	-	-	-	-	-	WRPSS	SSS[1:0]		xxxx,x000
DMA_SPI2_AMTH	SPI2_DMA 传输总字节数	7EF814H	SPI2_AMT[15:8]								0000,0000
DMA_SPI2_AMT	SPI2_DMA 传输总字节数	7EF815H	SPI2_AMT[7:0]								0000,0000
DMA_SPI2_DONEH	SPI2_DMA 传输完成字节数	7EF816H	SPI2_DONE[15:8]								0000,0000
DMA_SPI2_DONE	SPI2_DMA 传输完成字节数	7EF817H	SPI2_DONE[7:0]								0000,0000
DMA_SPI2_TXAH	SPI2_DMA 发送高地址	7EF818H	SPI2_TXA[15:8]								0000,0000
DMA_SPI2_TXAL	SPI2_DMA 发送低地址	7EF819H	SPI2_TXA[7:0]								0000,0000
DMA_SPI2_RXAH	SPI2_DMA 接收高地址	7EF81AH	SPI2_RXA[15:8]								0000,0000
DMA_SPI2_RXAL	SPI2_DMA 接收低地址	7EF81BH	SPI2_RXA[7:0]								0000,0000
DMA_SPI2_ITVH	SPI2_DMA 时间间隔寄存器	7EF81EH	SPI2_ITV[15:8]								0000,0000
DMA_SPI2_ITVL	SPI2_DMA 时间间隔寄存器	7EF81FH	SPI2_ITV[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_SPI3_CFG	SPI3_DMA 配置寄存器	7EF830H	SPI3IE	ACT_TX	ACT_RX	-	SPI3IP[1:0]		SPI3PTY[1:0]		000x,0000
DMA_SPI3_CR	SPI3_DMA 控制寄存器	7EF831H	ENSPI3	TRIG_M	TRIG_S	-	-	-	-	CLRFIFO	000x,xxx0
DMA_SPI3_STA	SPI3_DMA 状态寄存器	7EF832H	-	-	-	-	-	TXOVW	RXLOSS	SPI3IF	xxxx,x000
DMA_SPI3_CFG2	SPI3_DMA 配置寄存器 2	7EF833H	-	-	-	-	-	WRPSS	SSS[1:0]		xxxx,x000
DMA_SPI3_AMTH	SPI3_DMA 传输总字节数	7EF834H	SPI3_AMT[15:8]								0000,0000
DMA_SPI3_AMT	SPI3_DMA 传输总字节数	7EF835H	SPI3_AMT[7:0]								0000,0000
DMA_SPI3_DONEH	SPI3_DMA 传输完成字节数	7EF836H	SPI3_DONE[15:8]								0000,0000
DMA_SPI3_DONE	SPI3_DMA 传输完成字节数	7EF837H	SPI3_DONE[7:0]								0000,0000
DMA_SPI3_TXAH	SPI3_DMA 发送高地址	7EF838H	SPI3_TXA[15:8]								0000,0000
DMA_SPI3_TXAL	SPI3_DMA 发送低地址	7EF839H	SPI3_TXA[7:0]								0000,0000
DMA_SPI3_RXAH	SPI3_DMA 接收高地址	7EF83AH	SPI3_RXA[15:8]								0000,0000
DMA_SPI3_RXAL	SPI3_DMA 接收低地址	7EF83BH	SPI3_RXA[7:0]								0000,0000
DMA_SPI3_ITVH	SPI3_DMA 间隔寄存器	7EF83EH	SPI3_ITV[15:8]								0000,0000
DMA_SPI3_ITVL	SPI3_DMA 间隔寄存器	7EF83FH	SPI3_ITV[7:0]								0000,0000

33.6.2 SPIn_DMA 配置寄存器（DMA_SPIn_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_SPI_CFG	7EFA20H	SPIIE	ACT_TX	ACT_RX	-	SPIIP[1:0]		SPIPTY[1:0]	
DMA_SPI2_CFG	7EF810H	SPI2IE	ACT_TX2	ACT_RX2	-	SPI2IP[1:0]		SPI2PTY[1:0]	
DMA_SPI3_CFG	7EF830H	SPI3IE	ACT_TX3	ACT_RX3	-	SPI3IP[1:0]		SPI3PTY[1:0]	

SPInIE：SPIn_DMA 中断使能控制位（n=1/2/3）

0：禁止 SPIn_DMA 中断

1：允许 SPIn_DMA 中断

ACT_TXn：SPIn_DMA 发送数据控制位（n=1/2/3）

0：禁止 SPIn_DMA 发送数据。主机模式时，SPIn 只发送时钟到 SCLKn 端口，但不从 XRAM 读取数据，也不向 MOSIn 端口上发送数据；从机模式时，SPIn 不从 XRAM 读取数据，也不向 MISOn 端口上发送数据

1：允许 SPIn_DMA 发送数据。主机模式时，SPIn 发送时钟到 SCLKn 端口，同时从 XRAM 读取数据，并将数据发送到 MOSIn 端口；从机模式时，SPIn 从 XRAM 读取数据，并将数据发送到 MISOn 端口

ACT_RXn：SPIn_DMA 接收数据控制位（n=1/2/3）

0：禁止 SPIn_DMA 接收数据。主机模式时，SPIn 只发送时钟到 SCLKn 端口，但不从 MISOn 端口读取数据，也不向 XRAM 写数据；从机模式时，SPIn 不从 MOSIn 端口读取数据，也不向 XRAM 写数据。

1：允许 SPIn_DMA 接收数据。主机模式时，SPIn 发送时钟到 SCLKn 端口，同时从 MISOn 端口读取数据，并将数据写入 XRAM；从机模式时，SPIn 从 MOSIn 端口读取数据，并写入 XRAM。

SPnIP[1:0]：SPIn_DMA 中断优先级控制位（n=1/2/3）

SPInIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

SPInPTY[1:0]：SPIn_DMA 数据总线访问优先级控制位（n=1/2/3）

SPInPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.6.3 SPIn_DMA 控制寄存器（DMA_SPIn_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_SPI_CR	7EFA21H	ENSPI	TRIG_M	TRIG_S	-	-	-	-	CLRFIFO
DMA_SPI2_CR	7EF811H	ENSPI2	TRIG_M2	TRIG_S2	-	-	-	-	CLRFIFO2
DMA_SPI3_CR	7EF831H	ENSPI3	TRIG_M3	TRIG_S3	-	-	-	-	CLRFIFO3

ENSPIn：SPIn_DMA 功能使能控制位（n=1/2/3）

0：禁止 SPIn_DMA 功能

1：允许 SPIn_DMA 功能

TRIG_Mn：SPIn_DMA 主机模式触发控制位（n=1/2/3）

0：写 0 无效

1：写 1 开始 SPIn_DMA 主机模式操作，

TRIG_Sn：SPIn_DMA 从机模式触发控制位（n=1/2/3）

0：写 0 无效

1：写 1 开始 SPIn_DMA 从机模式操作，

CLRFIFOn：清除 SPIn_DMA 接收 FIFO 控制位（n=1/2/3）

0：写 0 无效

1：开始 SPIn_DMA 操作前，先清空 SPIn_DMA 内置的 FIFO

33.6.4 SPIn_DMA 状态寄存器（DMA_SPIn_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_SPI_STA	7EFA22H	-	-	-	-	-	TXOVW	RXLOSS	SPIIF
DMA_SPI2_STA	7EF812H	-	-	-	-	-	TXOVW2	RXLOSS2	SPI2IF
DMA_SPI3_STA	7EF832H	-	-	-	-	-	TXOVW3	RXLOSS3	SPI3IF

SPInIF：SPIn_DMA 中断请求标志位，当 SPIn_DMA 数据交换完成后，硬件自动将 SPInIF 置 1，若使能 SPIn_DMA 中断则进入中断服务程序。标志位需软件清零（n=1/2/3）

RXLOSSn：SPIn_DMA 接收数据丢弃标志位。SPIn_DMA 操作过程中，当 XRAM 总线过于繁忙，来不及清空 SPIn_DMA 的接收 FIFO 导致 SPIn_DMA 接收的数据自动丢弃时，硬件硬件自动将 RXLOSSn 置 1。标志位需软件清零（n=1/2/3）

TXOVWn：SPIn_DMA 数据覆盖标志位。SPIn_DMA 正在数据传输过程中，主机模式的 SPnI 写 SPnDAT 寄存器再次触发 SPIn 数据传输时，会导致数据传输失败，此时硬件硬件自动将 TXOVWn 置 1。标志位需软件清零（n=1/2/3）

33.6.5 SPIn_DMA 传输总字节寄存器（DMA_SPIn_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_SPI_AMTH	7EFA84H	SPI_AMT[15:8]							
DMA_SPI_AMT	7EFA23H	SPI_AMT[7:0]							
DMA_SPI2_AMTH	7EF814H	SPI2_AMT[15:8]							
DMA_SPI2_AMT	7EF815H	SPI2_AMT[7:0]							
DMA_SPI3_AMTH	7EF834H	SPI3_AMT[15:8]							
DMA_SPI3_AMT	7EF835H	SPI3_AMT[7:0]							

SPIn_AMT[15:0]：设置需要进行数据读写的字节数。（n=1/2/3）

注：实际读写的字节数为（SPIn_AMT+1），即当 SPIn_AMT 设置为 0 时，读写 1 字节，当 SPIn_AMT 设置 255 时，读写 256 字节。特别注意 SPIn_AMT 设置为 FFFFH 时表示无限循环模式。

33.6.6 SPIn_DMA 传输完成字节寄存器（DMA_SPIn_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_SPI_DONEH	7EFA85H	SPI_DONE[15:8]							
DMA_SPI_DONE	7EFA24H	SPI_DONE[7:0]							
DMA_SPI2_DONEH	7EF816H	SPI2_DONE[15:8]							
DMA_SPI2_DONE	7EF817H	SPI2_DONE[7:0]							
DMA_SPI3_DONEH	7EF836H	SPI3_DONE[15:8]							
DMA_SPI3_DONE	7EF837H	SPI3_DONE[7:0]							

SPIn_DONE[15:0]：当前已经传输完成的字节数。（n=1/2/3）

33.6.7 SPIn_DMA 发送地址寄存器（DMA_SPIn_TXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_SPI_TXAH	7EFA25H	SPI_TXA[15:8]							
DMA_SPI_TXAL	7EFA26H	SPI_TXA[7:0]							
DMA_SPI2_TXAH	7EF818H	SPI2_TXA[15:8]							
DMA_SPI2_TXAL	7EF819H	SPI2_TXA[7:0]							
DMA_SPI3_TXAH	7EF838H	SPI3_TXA[15:8]							
DMA_SPI3_TXAL	7EF839H	SPI3_TXA[7:0]							

SPIn_TXA[15:0]：设置进行数据传输时的源地址。执行 SPIn_DMA 操作时会从这个地址开始读数据。（n=1/2/3）

33.6.8 SPIn_DMA 接收地址寄存器（DMA_SPIn_RXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_SPI_RXAH	7EFA27H	SPI_RXA[15:8]							
DMA_SPI_RXAL	7EFA28H	SPI_RXA [7:0]							
DMA_SPI2_RXAH	7EF81AH	SPI2_RXA[15:8]							
DMA_SPI2_RXAL	7EF81BH	SPI2_RXA [7:0]							
DMA_SPI3_RXAH	7EF83AH	SPI3_RXA[15:8]							
DMA_SPI3_RXAL	7EF83BH	SPI4_RXA [7:0]							

SPIn_RXA[15:0]：设置进行数据传输时的目标地址。执行 SPIn_DMA 操作时会从这个地址开始写入数据。（n=1/2/3）

33.6.9 SPIn_DMA 配置寄存器 2（DMA_SPIn_CFG2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_SPI_CFG2	7EFA29H	-	-	-	-	-	WRPSS	SSS[1:0]	
DMA_SPI2_CFG2	7EF813H	-	-	-	-	-	WRPSS2	SSS2[1:0]	
DMA_SPI3_CFG2	7EF833H	-	-	-	-	-	WRPSS3	SSS3[1:0]	

WRPSSn：SPInn_DMA 过程中使能 SSn 脚控制位（n=1/2/3）

0：SPIn_DMA 传输过程中，不自动控制 SSn 脚

1：SPIn_DMA 传输过程中，自动拉低 SSn 脚，传输完成后，自动恢复原始状态

SSSn[1:0]：SPIn_DMA 过程中，自动控制 SSn 选择位（n=1/2/3）

SSSn[1:0]	SS 脚	SS2 脚	SS3 脚
00	P1.2	P6.4	P2.2
01	P2.2	P0.2	P7.0
10	P5.4	PB.1	P8.4
11	P3.5	P8.0	P9.0

33.6.10 SPIn_DMA 传输周期（DMA_SPIn_ITVx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_SPI_ITVH	7EFA2EH	SPI_ITV[15:8]							
DMA_SPI_ITVL	7EFA2FH	SPI_ITV[7:0]							
DMA_SPI2_ITVH	7EF81EH	SPI2_ITV[15:8]							
DMA_SPI2_ITVL	7EF81FH	SPI2_ITV[7:0]							
DMA_SPI3_ITVH	7EF83EH	SPI3_ITV[15:8]							
DMA_SPI3_ITVL	7EF83FH	SPI3_ITV[7:0]							

SPIn_ITV[15:0]：设置 SPIn 传输的间隔时间（基于系统时钟）。（n=1/2/3）

SPIn_ITV[15:0]上电后的默认值为 3，如需要加快 DMA 速度，需要软件将间隔时间设置为 0。

33.6.11 如何提高 SPI+DMA 的传输效率

SPI 使用 DMA 传输数据时，可通过减少字节之间的间隔时间来提高 SPI+DMA 的传输效率。进行如下的设置可减少 DMA 字节之间的间隔时间：

1、配置 HSSPI_CFG 寄存器中的 SS_HLD 和 SS_SETUP，减少 SS 信号的 HOLD 时间和 SETUP 时间；

2、设置 HSSPI_CFG2 寄存器中的 FIFOEN 位，开启高速 SPI 的 FIFO 模式；

（上面的两点设置对于 SPI+DMA 和高速 SPI+DMA 均适用）

3、对于 STC32G144K246 型号，增加额外增加了 DMA_SPI_ITVH 和 DMA_SPI_ITVL 寄存器来控制 DMA 时字节间隔时间，上电默认值位 3，可软件将这两个寄存器设置位 0，可降低间隔时间。

测试程序和测试结果，请参考技术论坛：“SPI+DMA 传输间隔时间测量@STC32G144K246”

33.7 利用串口 DMA 的发送时间间隔寄存器可实现 1.5bit/2bit 停止位

STC32G144K246 系列单片机的 USART 串口硬件只支持 1 位停止位，若实际应用中不需要校验位，则可将第 9 位设置 1，并将串口设置为 9 位数据，则可将实现两位停止位。

如果实际应用中需要使用校验位，则可通过设置串口 DMA 的发送时间间隔寄存器来实现 1.5 位或者 2 位停止位。具体实现方法：

- 将寄存器 DMA_UR1_ITVH 和 DMA_UR1_ITVL 中的值设为 0.5 位串口数据位时间，则可实现 1.5 位停止位；
- 将寄存器 DMA_UR1_ITVH 和 DMA_UR1_ITVL 中的值设为 1 位串口数据位时间，则可实现 2 位停止位。

【特别注意】：使用此方法必须使用 DMA 方式发送串口数据才有效。

33.8 串口 1/2/3/4/5/6/7/8 与存储器之间的数据交换（URnT_DMA，URnR_DMA）

33.8.1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR1T_CFG	UR1T_DMA 配置寄存器	7EFA30H	UR1TIE	-	-	-	UR1TIP[1:0]		UR1TPTY[1:0]		0xxx,0000
DMA_UR1T_CR	UR1T_DMA 控制寄存器	7EFA31H	ENUR1T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR1T_STA	UR1T_DMA 状态寄存器	7EFA32H	-	-	-	-	-	TXOVW	-	UR1TIF	xxxx,x0x0
DMA_UR1T_AMTH	UR1T_DMA 传输总字节数	7EFA88H	UR1T_AMT[15:8]								0000,0000
DMA_UR1T_AMT	UR1T_DMA 传输总字节数	7EFA33H	UR1T_AMT[7:0]								0000,0000
DMA_UR1T_DONEH	UR1T_DMA 传输完成字节数	7EFA89H	UR1T_DONE[15:8]								0000,0000
DMA_UR1T_DONE	UR1T_DMA 传输完成字节数	7EFA34H	UR1T_DONE[7:0]								0000,0000
DMA_UR1T_TXAH	UR1T_DMA 发送高地址	7EFA35H	UR1T_TXA[15:8]								0000,0000
DMA_UR1T_TXAL	UR1T_DMA 发送低地址	7EFA36H	UR1T_TXA[7:0]								0000,0000
DMA_UR1_ITVH	UR1_DMA 时间间隔	7EFAC8H	UR1_ITV[15:8]								0000,0000
DMA_UR1_ITVL	UR1_DMA 时间间隔寄存器	7EFAC9H	UR1_ITV[7:0]								0000,0000
DMA_UR1R_CFG	UR1R_DMA 配置寄存器	7EFA38H	UR1RIE	-	-	-	UR1RIP[1:0]		UR1RPTY[1:0]		0xxx,0000
DMA_UR1R_CR	UR1R_DMA 控制寄存器	7EFA39H	ENUR1R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR1R_STA	UR1R_DMA 状态寄存器	7EFA3AH	-	-	-	-	-	-	RXLOSS	UR1RIF	xxxx,xx00
DMA_UR1R_AMTH	UR1R_DMA 传输总字节数	7EFA8AH	UR1R_AMT[15:8]								0000,0000
DMA_UR1R_AMT	UR1R_DMA 传输总字节数	7EFA3BH	UR1R_AMT[7:0]								0000,0000
DMA_UR1R_DONEH	UR1R_DMA 传输完成字节数	7EFA8BH	UR1R_DONE[15:8]								0000,0000
DMA_UR1R_DONE	UR1R_DMA 传输完成字节数	7EFA3CH	UR1R_DONE[7:0]								0000,0000
DMA_UR1R_RXAH	UR1R_DMA 接收高地址	7EFA3DH	UR1R_RXA[15:8]								0000,0000
DMA_UR1R_RXAL	UR1R_DMA 接收低地址	7EFA3EH	UR1R_RXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR2T_CFG	UR2T_DMA 配置寄存器	7EFA40H	UR2TIE	-	-	-	UR2TIP[1:0]		UR2TPTY[1:0]		0xxx,0000
DMA_UR2T_CR	UR2T_DMA 控制寄存器	7EFA41H	ENUR2T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR2T_STA	UR2T_DMA 状态寄存器	7EFA42H	-	-	-	-	-	TXOVW	-	UR2TIF	xxxx,x0x0
DMA_UR2T_AMTH	UR2T_DMA 传输总字节数	7EFA8CH	UR2T_AMT[15:8]								0000,0000
DMA_UR2T_AMT	UR2T_DMA 传输总字节数	7EFA43H	UR2T_AMT[7:0]								0000,0000
DMA_UR2T_DONEH	UR2T_DMA 传输完成字节数	7EFA8DH	UR2T_DONE[15:8]								0000,0000
DMA_UR2T_DONE	UR2T_DMA 传输完成字节数	7EFA44H	UR2T_DONE[7:0]								0000,0000
DMA_UR2T_TXAH	UR2T_DMA 发送高地址	7EFA45H	UR2T_TXA[15:8]								0000,0000
DMA_UR2T_TXAL	UR2T_DMA 发送低地址	7EFA46H	UR2T_TXA[7:0]								0000,0000
DMA_UR2_ITVH	UR2_DMA 时间间隔寄存器	7EFACAH	UR2_ITV[15:8]								0000,0000
DMA_UR2_ITVL	UR2_DMA 时间间隔寄存器	7EFACBH	UR2_ITV[7:0]								0000,0000
DMA_UR2R_CFG	UR2R_DMA 配置寄存器	7EFA48H	UR2RIE	-	-	-	UR2RIP[1:0]		UR2RPTY[1:0]		0xxx,0000
DMA_UR2R_CR	UR2R_DMA 控制寄存器	7EFA49H	ENUR2R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR2R_STA	UR2R_DMA 状态寄存器	7EFA4AH	-	-	-	-	-	-	RXLOSS	UR2RIF	xxxx,xx00
DMA_UR2R_AMTH	UR2R_DMA 传输总字节数	7EFA8EH	UR2R_AMT[15:8]								0000,0000
DMA_UR2R_AMT	UR2R_DMA 传输总字节数	7EFA4BH	UR2R_AMT[7:0]								0000,0000
DMA_UR2R_DONEH	UR2R_DMA 传输完成字节数	7EFA8FH	UR2R_DONE[15:8]								0000,0000
DMA_UR2R_DONE	UR2R_DMA 传输完成字节数	7EFA4CH	UR2R_DONE[7:0]								0000,0000
DMA_UR2R_RXAH	UR2R_DMA 接收高地址	7EFA4DH	UR2R_RXA[15:8]								0000,0000
DMA_UR2R_RXAL	UR2R_DMA 接收低地址	7EFA4EH	UR2R_RXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR3T_CFG	UR3T_DMA 配置寄存器	7EFA50H	UR3TIE	-	-	-	UR3TIP[1:0]		UR3TPTY[1:0]		0xxx,0000
DMA_UR3T_CR	UR3T_DMA 控制寄存器	7EFA51H	ENUR3T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR3T_STA	UR3T_DMA 状态寄存器	7EFA52H	-	-	-	-	-	TXOVW	-	UR3TIF	xxxx,x0x0
DMA_UR3T_AMTH	UR3T_DMA 传输总字节数	7EFA90H	UR3T_AMT[15:8]								0000,0000
DMA_UR3T_AMT	UR3T_DMA 传输总字节数	7EFA53H	UR3T_AMT[7:0]								0000,0000
DMA_UR3T_DONEH	UR3T_DMA 传输完成字节数	7EFA91H	UR3T_DONE[15:8]								0000,0000
DMA_UR3T_DONE	UR3T_DMA 传输完成字节数	7EFA54H	UR3T_DONE[7:0]								0000,0000
DMA_UR3T_TXAH	UR3T_DMA 发送高地址	7EFA55H	UR3T_TXA[15:8]								0000,0000
DMA_UR3T_TXAL	UR3T_DMA 发送低地址	7EFA56H	UR3T_TXA[7:0]								0000,0000
DMA_UR3_ITVH	UR3_DMA 时间间隔寄存器	7EFACCH	UR3_ITV[15:8]								0000,0000
DMA_UR3_ITVL	UR3_DMA 时间间隔寄存器	7EFACDH	UR3_ITV[7:0]								0000,0000
DMA_UR3R_CFG	UR3R_DMA 配置寄存器	7EFA58H	UR3RIE	-	-	-	UR3RIP[1:0]		UR3RPTY[1:0]		0xxx,0000
DMA_UR3R_CR	UR3R_DMA 控制寄存器	7EFA59H	ENUR3R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR3R_STA	UR3R_DMA 状态寄存器	7EFA5AH	-	-	-	-	-	-	RXLOSS	UR3RIF	xxxx,xx00
DMA_UR3R_AMTH	UR3R_DMA 传输总字节数	7EFA92H	UR3R_AMT[15:8]								0000,0000
DMA_UR3R_AMT	UR3R_DMA 传输总字节数	7EFA5BH	UR3R_AMT[7:0]								0000,0000
DMA_UR3R_DONEH	UR3R_DMA 传输完成字节数	7EFA93H	UR3R_DONE[15:8]								0000,0000
DMA_UR3R_DONE	UR3R_DMA 传输完成字节数	7EFA5CH	UR3R_DONE[7:0]								0000,0000
DMA_UR3R_RXAH	UR3R_DMA 接收高地址	7EFA5DH	UR3R_RXA[15:8]								0000,0000
DMA_UR3R_RXAL	UR3R_DMA 接收低地址	7EFA5EH	UR3R_RXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR4T_CFG	UR4T_DMA 配置寄存器	7EFA60H	UR4TIE	-	-	-	UR4TIP[1:0]		UR4TPTY[1:0]		0xxx,0000
DMA_UR4T_CR	UR4T_DMA 控制寄存器	7EFA61H	ENUR4T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR4T_STA	UR4T_DMA 状态寄存器	7EFA62H	-	-	-	-	-	TXOVW	-	UR4TIF	xxxx,x0x0
DMA_UR4T_AMTH	UR4T_DMA 传输总字节数	7EFA94H	UR4T_AMT[15:8]								0000,0000
DMA_UR4T_AMT	UR4T_DMA 传输总字节数	7EFA63H	UR4T_AMT[7:0]								0000,0000
DMA_UR4T_DONEH	UR4T_DMA 传输完成字节数	7EFA95H	UR4T_DONE[15:8]								0000,0000
DMA_UR4T_DONE	UR4T_DMA 传输完成字节数	7EFA64H	UR4T_DONE[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR4T_TXAH	UR4T_DMA 发送高地址	7EFA65H	UR4T_TXA[15:8]								0000,0000
DMA_UR4T_TXAL	UR4T_DMA 发送低地址	7EFA66H	UR4T_TXA[7:0]								0000,0000
DMA_UR4_ITVH	UR4_DMA 时间间隔寄存器	7EFACEH	UR4_ITV[15:8]								0000,0000
DMA_UR4_ITVL	UR4_DMA 时间间隔寄存器	7EFACFH	UR4_ITV[7:0]								0000,0000
DMA_UR4R_CFG	UR4R_DMA 配置寄存器	7EFA68H	UR4RIE	-	-	-	UR4RIP[1:0]		UR4RPTY[1:0]		0xxx,0000
DMA_UR4R_CR	UR4R_DMA 控制寄存器	7EFA69H	ENUR4R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR4R_STA	UR4R_DMA 状态寄存器	7EFA6AH	-	-	-	-	-	-	RXLOSS	UR4RIF	xxxx,xx00
DMA_UR4R_AMTH	UR4R_DMA 传输总字节数	7EFA96H	UR4R_AMT[15:8]								0000,0000
DMA_UR4R_AMT	UR4R_DMA 传输总字节数	7EFA6BH	UR4R_AMT[7:0]								0000,0000
DMA_UR4R_DONEH	UR4R_DMA 传输完成字节数	7EFA97H	UR4R_DONE[15:8]								0000,0000
DMA_UR4R_DONE	UR4R_DMA 传输完成字节数	7EFA6CH	UR4R_DONE[7:0]								0000,0000
DMA_UR4R_RXAH	UR4R_DMA 接收高地址	7EFA6DH	UR4R_RXA[15:8]								0000,0000
DMA_UR4R_RXAL	UR4R_DMA 接收低地址	7EFA6EH	UR4R_RXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR5T_CFG	UR5T_DMA 配置寄存器	7EF750H	UR5TIE	-	-	-	UR5TIP[1:0]		UR5TPTY[1:0]		0xxx,0000
DMA_UR5T_CR	UR5T_DMA 控制寄存器	7EF751H	ENUR5T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR5T_STA	UR5T_DMA 状态寄存器	7EF752H	-	-	-	-	-	TXOVW	-	UR5TIF	xxxx,x0x0
DMA_UR5T_AMTH	UR5T_DMA 传输总字节数	7EF754H	UR5T_AMT[15:8]								0000,0000
DMA_UR5T_AMT	UR5T_DMA 传输总字节数	7EF755H	UR5T_AMT[7:0]								0000,0000
DMA_UR5T_DONEH	UR5T_DMA 传输完成字节数	7EF756H	UR5T_DONE[15:8]								0000,0000
DMA_UR5T_DONE	UR5T_DMA 传输完成字节数	7EF757H	UR5T_DONE[7:0]								0000,0000
DMA_UR5T_TXAH	UR5T_DMA 发送高地址	7EF758H	UR5T_TXA[15:8]								0000,0000
DMA_UR5T_TXAL	UR5T_DMA 发送低地址	7EF759H	UR5T_TXA[7:0]								0000,0000
DMA_UR5_ITVH	UR5_DMA 时间间隔寄存器	7EF75EH	UR5_ITV[15:8]								0000,0000
DMA_UR5_ITVL	UR5_DMA 时间间隔寄存器	7EF75FH	UR5_ITV[7:0]								0000,0000
DMA_UR5R_CFG	UR5R_DMA 配置寄存器	7EF760H	UR5RIE	-	-	-	UR5RIP[1:0]		UR5RPTY[1:0]		0xxx,0000
DMA_UR5R_CR	UR5R_DMA 控制寄存器	7EF761H	ENUR5R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR5R_STA	UR5R_DMA 状态寄存器	7EF762H	-	-	-	-	-	-	RXLOSS	UR5RIF	xxxx,xx00
DMA_UR5R_AMTH	UR5R_DMA 传输总字节数	7EF764H	UR5R_AMT[15:8]								0000,0000
DMA_UR5R_AMT	UR5R_DMA 传输总字节数	7EF765H	UR5R_AMT[7:0]								0000,0000
DMA_UR5R_DONEH	UR5R_DMA 传输完成字节数	7EF766H	UR5R_DONE[15:8]								0000,0000
DMA_UR5R_DONE	UR5R_DMA 传输完成字节数	7EF767H	UR5R_DONE[7:0]								0000,0000
DMA_UR5R_RXAH	UR5R_DMA 接收高地址	7EF768H	UR5R_RXA[15:8]								0000,0000
DMA_UR5R_RXAL	UR5R_DMA 接收低地址	7EF769H	UR5R_RXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR6T_CFG	UR6T_DMA 配置寄存器	7EF780H	UR6TIE	-	-	-	UR6TIP[1:0]		UR6TPTY[1:0]		0xxx,0000
DMA_UR6T_CR	UR6T_DMA 控制寄存器	7EF781H	ENUR6T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR6T_STA	UR6T_DMA 状态寄存器	7EF782H	-	-	-	-	-	TXOVW	-	UR6TIF	xxxx,x0x0
DMA_UR6T_AMTH	UR6T_DMA 传输总字节数	7EF784H	UR6T_AMT[15:8]								0000,0000
DMA_UR6T_AMT	UR6T_DMA 传输总字节数	7EF785H	UR6T_AMT[7:0]								0000,0000
DMA_UR6T_DONEH	UR6T_DMA 传输完成字节数	7EF786H	UR6T_DONE[15:8]								0000,0000
DMA_UR6T_DONE	UR6T_DMA 传输完成字节数	7EF787H	UR6T_DONE[7:0]								0000,0000
DMA_UR6T_TXAH	UR6T_DMA 发送高地址	7EF788H	UR6T_TXA[15:8]								0000,0000
DMA_UR6T_TXAL	UR6T_DMA 发送低地址	7EF789H	UR6T_TXA[7:0]								0000,0000
DMA_UR6_ITVH	UR6_DMA 时间间隔寄存器	7EF78EH	UR6_ITV[15:8]								0000,0000
DMA_UR6_ITVL	UR6_DMA 时间间隔寄存器	7EF78FH	UR6_ITV[7:0]								0000,0000
DMA_UR6R_CFG	UR6R_DMA 配置寄存器	7EF790H	UR6RIE	-	-	-	UR6RIP[1:0]		UR6RPTY[1:0]		0xxx,0000
DMA_UR6R_CR	UR6R_DMA 控制寄存器	7EF791H	ENUR6R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR6R_STA	UR6R_DMA 状态寄存器	7EF792H	-	-	-	-	-	-	RXLOSS	UR6RIF	xxxx,xx00
DMA_UR6R_AMTH	UR6R_DMA 传输总字节数	7EF794H	UR6R_AMT[15:8]								0000,0000
DMA_UR6R_AMT	UR6R_DMA 传输总字节数	7EF795H	UR6R_AMT[7:0]								0000,0000
DMA_UR6R_DONEH	UR6R_DMA 传输完成字节数	7EF796H	UR6R_DONE[15:8]								0000,0000
DMA_UR6R_DONE	UR6R_DMA 传输完成字节数	7EF797H	UR6R_DONE[7:0]								0000,0000
DMA_UR6R_RXAH	UR6R_DMA 接收高地址	7EF798H	UR6R_RXA[15:8]								0000,0000
DMA_UR6R_RXAL	UR6R_DMA 接收低地址	7EF799H	UR6R_RXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR7T_CFG	UR7T_DMA 配置寄存器	7EF7B0H	UR7TIE	-	-	-	UR7TIP[1:0]		UR7TPTY[1:0]		0xxx,0000
DMA_UR7T_CR	UR7T_DMA 控制寄存器	7EF7B1H	ENUR7T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR7T_STA	UR7T_DMA 状态寄存器	7EF7B2H	-	-	-	-	-	TXOVW	-	UR7TIF	xxxx,x0x0
DMA_UR7T_AMTH	UR7T_DMA 传输总字节数	7EF7B4H	UR7T_AMT[15:8]								0000,0000
DMA_UR7T_AMT	UR7T_DMA 传输总字节数	7EF7B5H	UR7T_AMT[7:0]								0000,0000
DMA_UR7T_DONEH	UR7T_DMA 传输完成字节数	7EF7B6H	UR7T_DONE[15:8]								0000,0000
DMA_UR7T_DONE	UR7T_DMA 传输完成字节数	7EF7B7H	UR7T_DONE[7:0]								0000,0000
DMA_UR7T_TXAH	UR7T_DMA 发送高地址	7EF7B8H	UR7T_TXA[15:8]								0000,0000
DMA_UR7T_TXAL	UR7T_DMA 发送低地址	7EF7B9H	UR7T_TXA[7:0]								0000,0000
DMA_UR7_ITVH	UR7_DMA 时间间隔寄存器	7EF7BEH	UR7_ITV[15:8]								0000,0000
DMA_UR7_ITVL	UR7_DMA 时间间隔寄存器	7EF7BFH	UR7_ITV[7:0]								0000,0000
DMA_UR7R_CFG	UR7R_DMA 配置寄存器	7EF7C0H	UR7RIE	-	-	-	UR7RIP[1:0]		UR7RPTY[1:0]		0xxx,0000
DMA_UR7R_CR	UR7R_DMA 控制寄存器	7EF7C1H	ENUR7R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR7R_STA	UR7R_DMA 状态寄存器	7EF7C2H	-	-	-	-	-	-	RXLOSS	UR7RIF	xxxx,xx00
DMA_UR7R_AMTH	UR7R_DMA 传输总字节数	7EF7C4H	UR7R_AMT[15:8]								0000,0000
DMA_UR7R_AMT	UR7R_DMA 传输总字节数	7EF7C5H	UR7R_AMT[7:0]								0000,0000
DMA_UR7R_DONEH	UR7R_DMA 传输完成字节数	7EF7C6H	UR7R_DONE[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR7R_DONE	UR7R_DMA 传输完成字节数	7EF7C7H	UR7R_DONE[7:0]								0000,0000
DMA_UR7R_RXAH	UR7R_DMA 接收高地址	7EF7C8H	UR7R_RXA[15:8]								0000,0000
DMA_UR7R_RXAL	UR7R_DMA 接收低地址	7EF7C9H	UR7R_RXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_UR8T_CFG	UR8T_DMA 配置寄存器	7EF7E0H	UR8TIE	-	-	-	UR8TIP[1:0]		UR8TPTY[1:0]		0xxx,0000
DMA_UR8T_CR	UR8T_DMA 控制寄存器	7EF7E1H	ENUR8T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_UR8T_STA	UR8T_DMA 状态寄存器	7EF7E2H	-	-	-	-	-	TXOVW	-	UR8TIF	xxxx,x0x0
DMA_UR8T_AMTH	UR8T_DMA 传输总字节数	7EF7E4H	UR8T_AMT[15:8]								0000,0000
DMA_UR8T_AMT	UR8T_DMA 传输总字节数	7EF7E5H	UR8T_AMT[7:0]								0000,0000
DMA_UR8T_DONEH	UR8T_DMA 传输完成字节数	7EF7E6H	UR8T_DONE[15:8]								0000,0000
DMA_UR8T_DONE	UR8T_DMA 传输完成字节数	7EF7E7H	UR8T_DONE[7:0]								0000,0000
DMA_UR8T_TXAH	UR8T_DMA 发送高地址	7EF7E8H	UR8T_TXA[15:8]								0000,0000
DMA_UR8T_TXAL	UR8T_DMA 发送低地址	7EF7E9H	UR8T_TXA[7:0]								0000,0000
DMA_UR8_ITVH	UR8_DMA 时间间隔寄存器	7EF7EEH	UR8_ITV[15:8]								0000,0000
DMA_UR8_ITVL	UR8_DMA 时间间隔寄存器	7EF7EFH	UR8_ITV[7:0]								0000,0000
DMA_UR8R_CFG	UR8R_DMA 配置寄存器	7EF7F0H	UR8RIE	-	-	-	UR8RIP[1:0]		UR8RPTY[1:0]		0xxx,0000
DMA_UR8R_CR	UR8R_DMA 控制寄存器	7EF7F1H	ENUR8R	-	TRIG	-	-	-	-	CLRFIFO	0x0x,xxx0
DMA_UR8R_STA	UR8R_DMA 状态寄存器	7EF7F2H	-	-	-	-	-	-	RXLOSS	UR8RIF	xxxx,xx00
DMA_UR8R_AMTH	UR8R_DMA 传输总字节数	7EF7F4H	UR8R_AMT[15:8]								0000,0000
DMA_UR8R_AMT	UR8R_DMA 传输总字节数	7EF7F5H	UR8R_AMT[7:0]								0000,0000
DMA_UR8R_DONEH	UR8R_DMA 传输完成字节数	7EF7F6H	UR8R_DONE[15:8]								0000,0000
DMA_UR8R_DONE	UR8R_DMA 传输完成字节数	7EF7F7H	UR8R_DONE[7:0]								0000,0000
DMA_UR8R_RXAH	UR8R_DMA 接收高地址	7EF7F8H	UR8R_RXA[15:8]								0000,0000
DMA_UR8R_RXAL	UR8R_DMA 接收低地址	7EF7F9H	UR8R_RXA[7:0]								0000,0000

33.8.2 URnT_DMA 配置寄存器（DMA_URnT_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1T_CFG	7EFA30H	UR1TIE	-	-	-	UR1TIP[1:0]		UR1TPTY[1:0]	
DMA_UR2T_CFG	7EFA40H	UR2TIE	-	-	-	UR2TIP[1:0]		UR2TPTY[1:0]	
DMA_UR3T_CFG	7EFA50H	UR3TIE	-	-	-	UR3TIP[1:0]		UR3TPTY[1:0]	
DMA_UR4T_CFG	7EFA60H	UR4TIE	-	-	-	UR4TIP[1:0]		UR4TPTY[1:0]	
DMA_UR5T_CFG	7EF750H	UR5TIE	-	-	-	UR5TIP[1:0]		UR5TPTY[1:0]	
DMA_UR6T_CFG	7EF780H	UR6TIE	-	-	-	UR6TIP[1:0]		UR6TPTY[1:0]	
DMA_UR7T_CFG	7EF7B0H	UR7TIE	-	-	-	UR7TIP[1:0]		UR7TPTY[1:0]	
DMA_UR8T_CFG	7EF7E0H	UR8TIE	-	-	-	UR8TIP[1:0]		UR8TPTY[1:0]	

URnTIE：URnT_DMA 中断使能控制位（n=1/2/3/4/5/6/7/8）

0：禁止 URnT_DMA 中断

1：允许 URnT_DMA 中断

URnTIP[1:0]：URnT_DMA 中断优先级控制位（n=1/2/3/4/5/6/7/8）

URnTIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

URnTPTY[1:0]：URnT_DMA 数据总线访问优先级控制位（n=1/2/3/4/5/6/7/8）

URnTPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.8.3 URnT_DMA 控制寄存器（DMA_URnT_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1T_CR	7EFA31H	ENUR1T	TRIG1	-	-	-	-	-	-
DMA_UR2T_CR	7EFA41H	ENUR2T	TRIG2	-	-	-	-	-	-
DMA_UR3T_CR	7EFA51H	ENUR3T	TRIG3	-	-	-	-	-	-
DMA_UR4T_CR	7EFA61H	ENUR4T	TRIG4	-	-	-	-	-	-
DMA_UR5T_CR	7EF751H	ENUR5T	TRIG5	-	-	-	-	-	-
DMA_UR6T_CR	7EF781H	ENUR6T	TRIG6	-	-	-	-	-	-
DMA_UR7T_CR	7EF7B1H	ENUR7T	TRIG7	-	-	-	-	-	-
DMA_UR8T_CR	7EF7E1H	ENUR8T	TRIG8	-	-	-	-	-	-

ENURnT：URnT_DMA 功能使能控制位（n=1/2/3/4/5/6/7/8）

0：禁止 URnT_DMA 功能

1：允许 URnT_DMA 功能

TRIGn：URnT_DMA 发送触发控制位（n=1/2/3/4/5/6/7/8）

0：写 0 无效

1：写 1 开始 URnT_DMA 自动发送数据

33.8.4 URnT_DMA 状态寄存器（DMA_URnT_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1T_STA	7EFA32H	-	-	-	-	-	TXOVW	-	UR1TIF
DMA_UR2T_STA	7EFA42H	-	-	-	-	-	TXOVW2	-	UR2TIF
DMA_UR3T_STA	7EFA52H	-	-	-	-	-	TXOVW3	-	UR3TIF
DMA_UR4T_STA	7EFA62H	-	-	-	-	-	TXOVW4	-	UR4TIF
DMA_UR5T_STA	7EF752H	-	-	-	-	-	TXOVW5	-	UR5TIF
DMA_UR6T_STA	7EF782H	-	-	-	-	-	TXOVW6	-	UR6TIF
DMA_UR7T_STA	7EF7B2H	-	-	-	-	-	TXOVW7	-	UR7TIF
DMA_UR8T_STA	7EF7E2H	-	-	-	-	-	TXOVW8	-	UR8TIF

URnTIF：URnT_DMA 中断请求标志位，当 URnT_DMA 数据发送完成后，硬件自动将 URnTIF 置 1，若使能 URnT_DMA 中断则进入中断服务程序。标志位需软件清零（n=1/2/3/4/5/6/7/8）

TXOVWn：URnT_DMA 数据覆盖标志位。URnT_DMA 正在数据传输过程中，串口写 SnBUF 寄存器再次触发串口发送数据时，会导致数据传输失败，此时硬件硬件自动将 TXOVWn 置 1。标志位需软件清零（n=1/2/3/4/5/6/7/8）**【对串口 2~串口 8，此位无效】**

33.8.5 URnT_DMA 传输总字节寄存器（DMA_URnT_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1T_AMTH	7EFA88H	UR1T_AMT[15:8]							
DMA_UR1T_AMT	7EFA33H	UR1T_AMT[7:0]							
DMA_UR2T_AMTH	7EFA8CH	UR2T_AMT[15:8]							
DMA_UR2T_AMT	7EFA43H	UR2T_AMT[7:0]							
DMA_UR3T_AMTH	7EFA90H	UR3T_AMT[15:8]							
DMA_UR3T_AMT	7EFA53H	UR3T_AMT[7:0]							
DMA_UR4T_AMTH	7EFA94H	UR4T_AMT[15:8]							
DMA_UR4T_AMT	7EFA63H	UR4T_AMT[7:0]							
DMA_UR5T_AMTH	7EF754H	UR5T_AMT[15:8]							
DMA_UR5T_AMT	7EF755H	UR5T_AMT[7:0]							
DMA_UR6T_AMTH	7EF784H	UR6T_AMT[15:8]							
DMA_UR6T_AMT	7EF785H	UR6T_AMT[7:0]							
DMA_UR7T_AMTH	7EF7B4H	UR7T_AMT[15:8]							
DMA_UR7T_AMT	7EF7B5H	UR7T_AMT[7:0]							
DMA_UR8T_AMTH	7EF7E4H	UR8T_AMT[15:8]							
DMA_UR8T_AMT	7EF7E5H	UR8T_AMT[7:0]							

URnT_AMT[15:0]：设置需要进行数据读写的字节数。（n=1/2/3/4/5/6/7/8）

注：实际读写的字节数为（URnT_AMT+1），即当 URnT_AMT 设置为 0 时，读写 1 字节，当 URnT_AMT 设置 255 时，读写 256 字节。特别注意 URnT_AMT 设置为 FFFFH 时表示无限循环模式。

33.8.6 URnT_DMA 传输完成字节寄存器（DMA_URnT_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1T_DONEH	7EFA89H	UR1T_DONE[15:8]							
DMA_UR1T_DONE	7EFA34H	UR1T_DONE[7:0]							
DMA_UR2T_DONEH	7EFA8DH	UR2T_DONE[15:8]							
DMA_UR2T_DONE	7EFA44H	UR2T_DONE[7:0]							
DMA_UR3T_DONEH	7EFA91H	UR3T_DONE[15:8]							
DMA_UR3T_DONE	7EFA54H	UR3T_DONE[7:0]							
DMA_UR4T_DONEH	7EFA85H	UR4T_DONE[15:8]							
DMA_UR4T_DONE	7EFA64H	UR4T_DONE[7:0]							
DMA_UR5T_DONEH	7EF756H	UR5T_DONE[15:8]							
DMA_UR5T_DONE	7EF757H	UR5T_DONE[7:0]							
DMA_UR6T_DONEH	7EF786H	UR6T_DONE[15:8]							
DMA_UR6T_DONE	7EF787H	UR6T_DONE[7:0]							
DMA_UR7T_DONEH	7EF7B6H	UR7T_DONE[15:8]							
DMA_UR7T_DONE	7EF7B7H	UR7T_DONE[7:0]							
DMA_UR8T_DONEH	7EF7E6H	UR8T_DONE[15:8]							
DMA_UR8T_DONE	7E7E7H	UR8T_DONE[7:0]							

URnT_DONE[15:0]：当前已经发送完成的字节数。（n=1/2/3/4/5/6/7/8）

33.8.7 URnT_DMA 发送地址寄存器（DMA_URnT_TXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1T_TXAH	7EFA35H	UR1T_TXA[15:8]							
DMA_UR1T_TXAL	7EFA36H	UR1T_TXA[7:0]							
DMA_UR2T_TXAH	7EFA45H	UR2T_TXA[15:8]							
DMA_UR2T_TXAL	7EFA46H	UR2T_TXA[7:0]							
DMA_UR3T_TXAH	7EFA55H	UR3T_TXA[15:8]							
DMA_UR3T_TXAL	7EFA56H	UR3T_TXA[7:0]							
DMA_UR4T_TXAH	7EFA65H	UR4T_TXA[15:8]							
DMA_UR4T_TXAL	7EFA66H	UR4T_TXA[7:0]							
DMA_UR5T_TXAH	7EF758H	UR5T_TXA[15:8]							
DMA_UR5T_TXAL	7EF759H	UR5T_TXA[7:0]							
DMA_UR6T_TXAH	7EF788H	UR6T_TXA[15:8]							
DMA_UR6T_TXAL	7EF789H	UR6T_TXA[7:0]							
DMA_UR7T_TXAH	7EF7B8H	UR7T_TXA[15:8]							
DMA_UR7T_TXAL	7EF7B9H	UR7T_TXA[7:0]							
DMA_UR8T_TXAH	7EF7E8H	UR8T_TXA[15:8]							
DMA_UR8T_TXAL	7EF7E9H	UR8T_TXA[7:0]							

URnT_TXA[15:0]：设置自动发送数据的源地址。执行 URnT_DMA 操作时会从这个地址开始读数据。（n=1/2/3/4/5/6/7/8）

33.8.8 URn_DMA 传输周期（DMA_URn_ITVx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1_ITVH	7EFAC8H	UR1T_ITV[15:8]							
DMA_UR1_ITVL	7EFAC9H	UR1T_ITV[7:0]							
DMA_UR2_ITVH	7EFACAH	UR2T_ITV[15:8]							
DMA_UR2_ITVL	7EFACBH	UR2T_ITV[7:0]							
DMA_UR3_ITVH	7EFACCH	UR3T_ITV[15:8]							
DMA_UR3_ITVL	7EFACDH	UR3T_ITV[7:0]							
DMA_UR4_ITVH	7EFACEH	UR4T_ITV[15:8]							
DMA_UR4_ITVL	7EFACFH	UR4T_ITV[7:0]							
DMA_UR5_ITVH	7EF75EH	UR5T_ITV[15:8]							
DMA_UR5_ITVL	7EF75FH	UR5T_ITV[7:0]							
DMA_UR6_ITVH	7EF78EH	UR6T_ITV[15:8]							
DMA_UR6_ITVL	7EF78FH	UR6T_ITV[7:0]							
DMA_UR7_ITVH	7EF7BEH	UR7T_ITV[15:8]							
DMA_UR7_ITVL	7EF7BFH	UR7T_ITV[7:0]							
DMA_UR8_ITVH	7EF7EEH	UR8T_ITV[15:8]							
DMA_UR8_ITVL	7EF7EFH	UR8T_ITV[7:0]							

URn_ITV[15:0]：设置 UARTn 发送和接收的间隔时间（基于系统时钟）。（n=1/2/3/4/5/6/7/8）

33.8.9 URnR_DMA 配置寄存器（DMA_URnR_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1R_CFG	7EFA38H	UR1RIE	-	-	-	UR1RIP[1:0]		UR1RPTY[1:0]	
DMA_UR2R_CFG	7EFA48H	UR2RIE	-	-	-	UR2RIP[1:0]		UR2RPTY[1:0]	
DMA_UR3R_CFG	7EFA58H	UR3RIE	-	-	-	UR3RIP[1:0]		UR3RPTY[1:0]	
DMA_UR4R_CFG	7EFA68H	UR4RIE	-	-	-	UR4RIP[1:0]		UR4RPTY[1:0]	
DMA_UR5R_CFG	7EF760H	UR5RIE	-	-	-	UR5RIP[1:0]		UR5RPTY[1:0]	
DMA_UR6R_CFG	7EF790H	UR6RIE	-	-	-	UR6RIP[1:0]		UR6RPTY[1:0]	
DMA_UR7R_CFG	7EF7C0H	UR7RIE	-	-	-	UR7RIP[1:0]		UR7RPTY[1:0]	
DMA_UR8R_CFG	7EF7F0H	UR8RIE	-	-	-	UR8RIP[1:0]		UR8RPTY[1:0]	

URnRIE：URnR_DMA 中断使能控制位（n=1/2/3/4/5/6/7/8）

0：禁止 URnR_DMA 中断

1：允许 URnR_DMA 中断

URnRIP[1:0]：URnR_DMA 中断优先级控制位（n=1/2/3/4/5/6/7/8）

URnRIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

URnRPTY[1:0]：URnR_DMA 数据总线访问优先级控制位（n=1/2/3/4/5/6/7/8）

URnRPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.8.10 URnR_DMA 控制寄存器（DMA_URnR_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1R_CR	7EFA39H	ENUR1R	-	TRIG1	-	-	-	-	CLRFIFO1
DMA_UR2R_CR	7EFA49H	ENUR2R	-	TRIG2	-	-	-	-	CLRFIFO2
DMA_UR3R_CR	7EFA59H	ENUR3R	-	TRIG3	-	-	-	-	CLRFIFO3
DMA_UR4R_CR	7EFA69H	ENUR4R	-	TRIG4	-	-	-	-	CLRFIFO4
DMA_UR5R_CR	7EF761H	ENUR5R	-	TRIG5	-	-	-	-	CLRFIFO5
DMA_UR6R_CR	7EF791H	ENUR6R	-	TRIG6	-	-	-	-	CLRFIFO6
DMA_UR7R_CR	7EF7C1H	ENUR7R	-	TRIG7	-	-	-	-	CLRFIFO7
DMA_UR8R_CR	7EF7F1H	ENUR8R	-	TRIG8	-	-	-	-	CLRFIFO8

ENURnR：URnR_DMA 功能使能控制位（n=1/2/3/4/5/6/7/8）

0：禁止 URnR_DMA 功能

1：允许 URnR_DMA 功能

TRIG：URnR_DMA 接收触发控制位（n=1/2/3/4/5/6/7/8）

0：写 0 无效

1：写 1 开始 URnR_DMA 自动接收数据

CLRFIFOn：清除 URnR_DMA 接收 FIFO 控制位（n=1/2/3/4/5/6/7/8）

0：写 0 无效

1：开始 URnR_DMA 操作前，先清空 URnR_DMA 内置的 FIFO

33.8.11 URnR_DMA 状态寄存器（DMA_URnR_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1R_STA	7EFA3AH	-	-	-	-	-	-	RXLOSS1	UR1RIF
DMA_UR2R_STA	7EFA4AH	-	-	-	-	-	-	RXLOSS2	UR2RIF
DMA_UR3R_STA	7EFA5AH	-	-	-	-	-	-	RXLOSS3	UR3RIF
DMA_UR4R_STA	7EFA6AH	-	-	-	-	-	-	RXLOSS4	UR4RIF
DMA_UR5R_STA	7EF762H	-	-	-	-	-	-	RXLOSS5	UR5RIF
DMA_UR6R_STA	7EF792H	-	-	-	-	-	-	RXLOSS6	UR6RIF
DMA_UR7R_STA	7EF7C2H	-	-	-	-	-	-	RXLOSS7	UR7RIF
DMA_UR8R_STA	7EF7F2H	-	-	-	-	-	-	RXLOSS8	UR8RIF

URnRIF：URnR_DMA 中断请求标志位，当 URnR_DMA 接收数据完成后，硬件自动将 URnRIF 置 1，若使能 URnR_DMA 中断则进入中断服务程序。标志位需软件清零（n=1/2/3/4/5/6/7/8）

RXLOSSn：URnR_DMA 接收数据丢弃标志位。URnR_DMA 操作过程中，当 XRAM 总线过于繁忙，来不及清空 URnR_DMA 的接收 FIFO 导致 URnR_DMA 接收的数据自动丢弃时，硬件硬件自动将 RXLOSSn 置 1。标志位需软件清零（n=1/2/3/4/5/6/7/8）

33.8.12 URnR_DMA 传输总字节寄存器（DMA_URnR_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1R_AMTH	7EFA8AH	UR1R_AMT[15:8]							
DMA_UR1R_AMT	7EFA3BH	UR1R_AMT[7:0]							
DMA_UR2R_AMTH	7EFA8EH	UR2R_AMT[15:8]							
DMA_UR2R_AMT	7EFA4BH	UR2R_AMT[7:0]							
DMA_UR3R_AMTH	7EFA92H	UR3R_AMT[15:8]							
DMA_UR3R_AMT	7EFA5BH	UR3R_AMT[7:0]							
DMA_UR4R_AMTH	7EFA96H	UR4R_AMT[15:8]							
DMA_UR4R_AMT	7EFA6BH	UR4R_AMT[7:0]							
DMA_UR5R_AMTH	7EF764H	UR5R_AMT[15:8]							
DMA_UR5R_AMT	7EF765H	UR5R_AMT[7:0]							
DMA_UR6R_AMTH	7EF794H	UR6R_AMT[15:8]							
DMA_UR6R_AMT	7EF795H	UR6R_AMT[7:0]							
DMA_UR7R_AMTH	7EF7C4H	UR7R_AMT[15:8]							
DMA_UR7R_AMT	7EF7C5H	UR7R_AMT[7:0]							
DMA_UR8R_AMTH	7EF7F4H	UR8R_AMT[15:8]							
DMA_UR8R_AMT	7EF7F5H	UR8R_AMT[7:0]							

URnR_AMT[15:0]：设置需要进行数据读写的字节数。（n=1/2/3/4/5/6/7/8）

注：实际读写的字节数为（URnR_AMT+1），即当 URnR_AMT 设置为 0 时，读写 1 字节，当 URnR_AMT 设置 255 时，读写 256 字节。特别注意 URnR_AMT 设置为 FFFFH 时表示无限循环模式。

33.8.13 URnR_DMA 传输完成字节寄存器（DMA_URnR_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1R_DONEH	7EFA8BH	UR1R_DONE[15:8]							
DMA_UR1R_DONE	7EFA3CH	UR1R_DONE[7:0]							
DMA_UR2R_DONEH	7EFA8FH	UR2R_DONE[15:8]							
DMA_UR2R_DONE	7EFA4CH	UR2R_DONE[7:0]							
DMA_UR3R_DONEH	7EFA93H	UR3R_DONE[15:8]							
DMA_UR3R_DONE	7EFA5CH	UR3R_DONE[7:0]							
DMA_UR4R_DONEH	7EFA97H	UR4R_DONE[15:8]							
DMA_UR4R_DONE	7EFA6CH	UR4R_DONE[7:0]							
DMA_UR5R_DONEH	7EF766H	UR5R_DONE[15:8]							
DMA_UR5R_DONE	7EF767H	UR5R_DONE[7:0]							
DMA_UR6R_DONEH	7EF796H	UR6R_DONE[15:8]							
DMA_UR6R_DONE	7EF797H	UR6R_DONE[7:0]							
DMA_UR7R_DONEH	7EF7C6H	UR7R_DONE[15:8]							
DMA_UR7R_DONE	7EF7C7H	UR7R_DONE[7:0]							
DMA_UR8R_DONEH	7EF7F6H	UR8R_DONE[15:8]							
DMA_UR8R_DONE	7EF7F7H	UR8R_DONE[7:0]							

URnR_DONE[15:0]：当前已经接收完成的字节数。（n=1/2/3/4/5/6/7/8）

33.8.14 URnR_DMA 接收地址寄存器（DMA_URnR_RXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_UR1R_RXAH	7EFA3DH	UR1R_RXA[15:8]							
DMA_UR1R_RXAL	7EFA3EH	UR1R_RXA[7:0]							
DMA_UR2R_RXAH	7EFA4DH	UR2R_RXA[15:8]							
DMA_UR2R_RXAL	7EFA4EH	UR2R_RXA[7:0]							
DMA_UR3R_RXAH	7EFA5DH	UR3R_RXA[15:8]							
DMA_UR3R_RXAL	7EFA5EH	UR3R_RXA[7:0]							
DMA_UR4R_RXAH	7EFA6DH	UR4R_RXA[15:8]							
DMA_UR4R_RXAL	7EFA6EH	UR4R_RXA[7:0]							
DMA_UR5R_RXAH	7EF768H	UR5R_RXA[15:8]							
DMA_UR5R_RXAL	7EF769H	UR5R_RXA[7:0]							
DMA_UR6R_RXAH	7EF798H	UR6R_RXA[15:8]							
DMA_UR6R_RXAL	7EF799H	UR6R_RXA[7:0]							
DMA_UR7R_RXAH	7EF7C8H	UR7R_RXA[15:8]							
DMA_UR7R_RXAL	7EF7C9H	UR7R_RXA[7:0]							
DMA_UR8R_RXAH	7EF7F8H	UR8R_RXA[15:8]							
DMA_UR8R_RXAL	7EF7F9H	UR8R_RXA[7:0]							

URnR_RXA[15:0]：设置自动接收数据的目标地址。执行 URnR_DMA 操作时会从这个地址开始写数据。（n=1/2/3/4/5/6/7/8）

33.9 TFT 彩屏与存储器之间的数据读写（LCM_DMA）

33.9.1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_LCM_CFG	LCM_DMA 配置寄存器	7EFA70H	LCMIE	-	-	-	LCMIP[1:0]		LCMPTY[1:0]		0xxx,0000
DMA_LCM_CR	LCM_DMA 控制寄存器	7EFA71H	ENLCM	TRIGWC	TRIGWD	TRIGRC	TRIGRD	-	-	-	0000,0xxx
DMA_LCM_STA	LCM_DMA 状态寄存器	7EFA72H	-	-	-	-	-	-	TXOVW	LCMIF	xxxx,xx00
DMA_LCM_AMTH	LCM_DMA 传输总字节数	7EFA86H	LCM_AMT[15:8]								0000,0000
DMA_LCM_AMT	LCM_DMA 传输总字节数	7EFA73H	LCM_AMT[7:0]								0000,0000
DMA_LCM_DONEH	LCM_DMA 传输完成字节数	7EFA87H	LCM_DONE[15:8]								0000,0000
DMA_LCM_DONE	LCM_DMA 传输完成字节数	7EFA74H	LCM_DONE[7:0]								0000,0000
DMA_LCM_TXAH	LCM_DMA 发送高地址	7EFA75H	LCM_TXA[15:8]								0000,0000
DMA_LCM_TXAL	LCM_DMA 发送低地址	7EFA76H	LCM_TXA[7:0]								0000,0000
DMA_LCM_RXAH	LCM_DMA 接收高地址	7EFA77H	LCM_RXA[15:8]								0000,0000
DMA_LCM_RXAL	LCM_DMA 接收低地址	7EFA78H	LCM_RXA[7:0]								0000,0000
DMA_LCM_ITVH	LCM_DMA 时间间隔寄存器	7EFA7EH	LCM_ITV[15:8]								0000,0000
DMA_LCM_ITVL	LCM_DMA 时间间隔寄存器	7EFA7FH	LCM_ITV[7:0]								0000,0000

33.9.2 TFT 彩屏 DMA 配置寄存器（DMA_LCM_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_LCM_CFG	7EFA70H	LCMIE	-	-	-	LCMIP[1:0]		LCMPTY[1:0]	

LCMIE：TFT 彩屏 DMA 中断使能控制位

0：禁止 TFT 彩屏 DMA 中断

1：允许 TFT 彩屏 DMA 中断

LCMIP[1:0]：TFT 彩屏 DMA 中断优先级控制位

LCMIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

LCMPTY[1:0]：TFT 彩屏 DMA 数据总线访问优先级控制位

LCMPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.9.3 TFT 彩屏 DMA 控制寄存器（DMA_LCM_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_LCM_CR	7EFA71H	ENLCM	TRIGWC	TRIGWD	TRIGRC	TRIGRD	-	-	CLRFIFO

ENLCM：TFT 彩屏 DMA 功能使能控制位

0：禁止 TFT 彩屏 DMA 功能

1：允许 TFT 彩屏 DMA 功能

TRIGWC：TFT 彩屏 DMA 发送命令模式触发控制位

0：写 0 无效

1：写 1 开始 TFT 彩屏 DMA 发送命令模式操作

TRIGWD：TFT 彩屏 DMA 发送数据模式触发控制位

0：写 0 无效

1：写 1 开始 TFT 彩屏 DMA 发送数据模式操作

TRIGRC：TFT 彩屏 DMA 读取命令模式触发控制位

0：写 0 无效

1：写 1 开始 TFT 彩屏 DMA 读取命令模式操作

TRIGRD：TFT 彩屏 DMA 读取数据模式触发控制位

0：写 0 无效

1：写 1 开始 TFT 彩屏 DMA 读取数据模式操作

CLRFIFO：清除 TFT 彩屏 DMA 接收 FIFO 控制位

0：写 0 无效

1：开始 TFT 彩屏 DMA 操作前，先清空内置的 FIFO

33.9.4 TFT 彩屏 DMA 状态寄存器（DMA_LCM_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_LCM_STA	7EFA72H	-	-	-	-	-	-	TXOVW	LCMIF

LCMIF：TFT 彩屏 DMA 中断请求标志位，当 TFT 彩屏 DMA 数据交换完成后，硬件自动将 LCMIF 置 1，若使能 TFT 彩屏 DMA 中断则进入中断服务程序。标志位需软件清零

TXOVW：TFT 彩屏 DMA 数据覆盖标志位。TFT 彩屏 DMA 正在数据传输过程中，LCMIF 写 LCMIFDATL 和 LCMIFDATH 寄存器时，会导致数据传输失败，此时硬件硬件自动将 TXOVW 置 1。标志位需软件清零

33.9.5 TFT 彩屏 DMA 传输总字节寄存器（DMA_LCM_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_LCM_AMTH	7EFA86H	LCM_AMT[15:8]							
DMA_LCM_AMT	7EFA73H	LCM_AMT[7:0]							

LCM_AMT[15:0]：设置需要进行数据读写的字节数。

注：实际读写的字节数为（LCM_AMT+1），即当 LCM_AMT 设置为 0 时，读写 1 字节，当 LCM_AMT 设置 255 时，读写 256 字节。特别注意 LCM_AMT 设置为 FFFFH 时表示无限循环模式。

33.9.6 TFT 彩屏 DMA 传输完成字节寄存器（DMA_LCM_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_LCM_DONEH	7EFA87H	LCM_DONE[15:8]							
DMA_LCM_DONE	7EFA74H	LCM_DONE[7:0]							

LCM_DONE[15:0]：当前已经传输完成的字节数。

33.9.7 TFT 彩屏 DMA 发送地址寄存器（DMA_LCM_TXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_LCM_TXAH	7EFA75H	LCM_TXA[15:8]							
DMA_LCM_TXAL	7EFA76H	LCM_TXA[7:0]							

LCM_TXA[15:0]：设置进行数据传输时的源地址。执行 TFT 彩屏 DMA 操作时会从这个地址开始读数据。

33.9.8 TFT 彩屏 DMA 接收地址寄存器（DMA_LCM_RXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_LCM_RXAH	7EFA77H	LCM_RXA[15:8]							
DMA_LCM_RXAL	7EFA78H	LCM_RXA[7:0]							

LCM_RXA[15:0]：设置进行数据传输时的目标地址。执行 TFT 彩屏 DMA 操作时会从这个地址开始写入数据。

33.9.9 TFT 彩屏 DMA 传输周期（DMA_LCM_ITVx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_LCM_ITVH	7EFA7EH	LCM_ITV[15:8]							
DMA_LCM_ITVL	7EFA7FH	LCM_ITV[7:0]							

LCM_ITV[15:0]：设置数据传输时周期（基于系统时钟经过 LCMIFPSCR 分频器分频后的 LCMIF 时钟）。

33.10 I2C1/2 与存储器之间的数据交换（I2CnT_DMA，I2CnR_DMA）

33.10.1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_I2CT_CFG	I2CT_DMA 配置寄存器	7EFA98H	I2CTIE	-	-	-	I2CTIP[1:0]		I2CTPTY[1:0]		0xxx,0000
DMA_I2CT_CR	I2CT_DMA 控制寄存器	7EFA99H	ENI2CT	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_I2CT_STA	I2CT_DMA 状态寄存器	7EFA9AH	-	-	-	-	-	TXOVW	-	I2CTIF	xxxx,x0x0
DMA_I2CT_AMTH	I2CT_DMA 传输总字节数	7EFAA8H	I2CT_AMT[15:8]								0000,0000
DMA_I2CT_AMT	I2CT_DMA 传输总字节数	7EFA9BH	I2CT_AMT[7:0]								0000,0000
DMA_I2CT_DONEH	I2CT_DMA 传输完成字节数	7EFAA9H	I2CT_DONE[15:8]								0000,0000
DMA_I2CT_DONE	I2CT_DMA 传输完成字节数	7EFA9CH	I2CT_DONE[7:0]								0000,0000
DMA_I2CT_TXAH	I2CT_DMA 发送高地址	7EFA9DH	I2CT_TXA[15:8]								0000,0000
DMA_I2CT_TXAL	I2CT_DMA 发送低地址	7EFA9EH	I2CT_TXA[7:0]								0000,0000
DMA_I2C_ITVH	I2C_DMA 时间间隔寄存器	7EFAC4H	I2C_ITV[15:8]								0000,0000
DMA_I2C_ITVL	I2C_DMA 时间间隔寄存器	7EFAC5H	I2C_ITV[7:0]								0000,0000
DMA_I2CR_CFG	I2CR_DMA 配置寄存器	7EFAA0H	I2CRIE	-	-	-	I2CRIP[1:0]		I2CRPTY[1:0]		0xxx,0000
DMA_I2CR_CR	I2CR_DMA 控制寄存器	7EFAA1H	ENI2CR	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_I2CR_STA	I2CR_DMA 状态寄存器	7EFAA2H	-	-	-	-	-	-	RXLOSS	I2CRIF	xxxx,xx00
DMA_I2CR_AMTH	I2CR_DMA 传输总字节数	7EFAAAH	I2CR_AMT[15:8]								0000,0000
DMA_I2CR_AMT	I2CR_DMA 传输总字节数	7EFAA3H	I2CR_AMT[7:0]								0000,0000
DMA_I2CR_DONEH	I2CR_DMA 传输完成字节数	7EFAABH	I2CR_DONE[15:8]								0000,0000
DMA_I2CR_DONE	I2CR_DMA 传输完成字节数	7EFAA4H	I2CR_DONE[7:0]								0000,0000
DMA_I2CR_RXAH	I2CR_DMA 接收高地址	7EFAA5H	I2CR_RXA[15:8]								0000,0000
DMA_I2CR_RXAL	I2CR_DMA 接收低地址	7EFAA6H	I2CR_RXA[7:0]								0000,0000
DMA_I2C_CR	I2C_DMA 控制寄存器	7EFAADH	RDSEL	-	-	-	-	ACKERR	INTEN	BMMEN	0xxx,x000
DMA_I2C_ST1	I2C_DMA 状态寄存器	7EFAAEH	COUNT[7:0]								0000,0000
DMA_I2C_ST2	I2C_DMA 状态寄存器	7EFAAFH	COUNT[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_I2C2T_CFG	I2C2T_DMA 配置寄存器	7EF870H	I2C2TIE	-	-	-	I2C2TIP[1:0]		I2C2TPTY[1:0]		0xxx,0000
DMA_I2C2T_CR	I2C2T_DMA 控制寄存器	7EF871H	ENI2C2T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_I2C2T_STA	I2C2T_DMA 状态寄存器	7EF872H	-	-	-	-	-	TXOVW	-	I2C2TIF	xxxx,x0x0
DMA_I2C2T_AMTH	I2C2T_DMA 传输总字节数	7EF874H	I2C2T_AMT[15:8]								0000,0000
DMA_I2C2T_AMT	I2C2T_DMA 传输总字节数	7EF875H	I2C2T_AMT[7:0]								0000,0000
DMA_I2C2T_DONEH	I2C2T_DMA 传输完成字节数	7EF876H	I2C2T_DONE[15:8]								0000,0000
DMA_I2C2T_DONE	I2C2T_DMA 传输完成字节数	7EF877H	I2C2T_DONE[7:0]								0000,0000
DMA_I2C2T_TXAH	I2C2T_DMA 发送高地址	7EF878H	I2C2T_TXA[15:8]								0000,0000
DMA_I2C2T_TXAL	I2C2T_DMA 发送低地址	7EF879H	I2C2T_TXA[7:0]								0000,0000
DMA_I2C2_CR	I2C2_DMA 控制寄存器	7EF87BH	RDSEL	-	-	-	-	ACKERR	INTEN	BMMEN	0xxx,x000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_I2C2_ST1	I2C2_DMA 状态寄存器	7EF87CH	COUNT[7:0]								0000,0000
DMA_I2C2_ST2	I2C2_DMA 状态寄存器	7EF87DH	COUNT[15:8]								0000,0000
DMA_I2C2_ITVH	I2C2_DMA 时间间隔寄存器	7EF87EH	I2C2_ITV[15:8]								0000,0000
DMA_I2C2_ITVL	I2C2_DMA 时间间隔寄存器	7EF87FH	I2C2_ITV[7:0]								0000,0000
DMA_I2C2R_CFG	I2C2R_DMA 配置寄存器	7EF880H	I2C2RIE	-	-	-	I2C2RIP[1:0]		I2C2RPTY[1:0]		0xxx,0000
DMA_I2C2R_CR	I2C2R_DMA 控制寄存器	7EF881H	ENI2C2R	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_I2C2R_STA	I2C2R_DMA 状态寄存器	7EF882H	-	-	-	-	-	-	RXLOSS	I2C2RIF	xxxx,xx00
DMA_I2C2R_AMTH	I2C2R_DMA 传输总字节数	7EF884H	I2C2R_AMT[15:8]								0000,0000
DMA_I2C2R_AMT	I2C2R_DMA 传输总字节数	7EF885H	I2C2R_AMT[7:0]								0000,0000
DMA_I2C2R_DONEH	I2C2R_DMA 传输完成字节数	7EF886H	I2C2R_DONE[15:8]								0000,0000
DMA_I2C2R_DONE	I2C2R_DMA 传输完成字节数	7EF887H	I2C2R_DONE[7:0]								0000,0000
DMA_I2C2R_RXAH	I2C2R_DMA 接收高地址	7EF888H	I2C2R_RXA[15:8]								0000,0000
DMA_I2C2R_RXAL	I2C2R_DMA 接收低地址	7EF889H	I2C2R_RXA[7:0]								0000,0000

33.10.2 I2CnT_DMA 配置寄存器（DMA_I2CnT_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CT_CFG	7EFA98H	I2CTIE	-	-	-	I2CTIP[1:0]		I2CTPTY[1:0]	
DMA_I2C2T_CFG	7EF870H	I2C2TIE	-	-	-	I2C2TIP[1:0]		I2C2TPTY[1:0]	

I2CnTIE：I2CnT_DMA 中断使能控制位（n=1/2）

0：禁止 I2CnT_DMA 中断

1：允许 I2CnT_DMA 中断

I2CnTIP[1:0]：I2CnT_DMA 中断优先级控制位（n=1/2）

I2CnTIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

I2CnTPTY[1:0]：I2CnT_DMA 数据总线访问优先级控制位（n=1/2）

I2CnTPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.10.3 I2CnT_DMA 控制寄存器（DMA_I2CnT_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CT_CR	7EFA99H	ENI2CT	TRIG1	-	-	-	-	-	-
DMA_I2C2T_CR	7EF871H	ENI2C2T	TRIG2	-	-	-	-	-	-

ENI2CnT：I2CnT_DMA 功能使能控制位（n=1/2）

0：禁止 I2CnT_DMA 功能

1：允许 I2CnT_DMA 功能

TRIGn：I2CnT_DMA 发送触发控制位（n=1/2）

0：写 0 无效

1：写 1 开始 I2CnT_DMA 自动发送数据

33.10.4 I2CnT_DMA 状态寄存器（DMA_I2CnT_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CT_STA	7EFA9AH	-	-	-	-	-	TXOVW1	-	I2CTIF
DMA_I2C2T_STA	7EF872H	-	-	-	-	-	TXOVW2	-	I2C2TIF

I2CnTIF：I2CnT_DMA 中断请求标志位，当 I2CnT_DMA 数据发送完成后，硬件自动将 I2CnTIF 置 1，若使能 I2CnT_DMA 中断则进入中断服务程序。标志位需软件清零（n=1/2）

TXOVWn：I2CnT_DMA 数据覆盖标志位。I2CnT_DMA 正在数据传输过程中，写 I2C 数据寄存器 I2CnTXD 时，会导致数据传输失败，此时硬件硬件自动将 TXOVWn 置 1。标志位需软件清零（n=1/2）

33.10.5 I2CnT_DMA 传输总字节寄存器（DMA_I2CnT_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CT_AMTH	7EFAA8H	I2C_AMT[15:8]							
DMA_I2CT_AMT	7EFA9BH	I2C_AMT[7:0]							
DMA_I2C2T_AMTH	7EF874H	I2C2_AMT[15:8]							
DMA_I2C2T_AMT	7EF875H	I2C2_AMT[7:0]							

I2Cn_AMT[15:0]：设置需要进行数据读写的字节数。（n=1/2）

注：实际读写的字节数为（I2Cn_AMT+1），即当 I2Cn_AMT 设置为 0 时，读写 1 字节，当 I2Cn_AMT 设置 255 时，读写 256 字节。特别注意 I2Cn_AMT 设置为 FFFFH 时表示无限循环模式。

33.10.6 I2CnT_DMA 传输完成字节寄存器（DMA_I2CnT_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CT_DONEH	7EFAA9H	I2C_DONE[15:8]							
DMA_I2CT_DONE	7EFA9CH	I2C_DONE[7:0]							
DMA_I2C2T_DONEH	7EF876H	I2C2_DONE[15:8]							
DMA_I2C2T_DONE	7EF877H	I2C2_DONE[7:0]							

I2Cn_DONE[15:0]：当前已经发送完成的字节数。（n=1/2）

33.10.7 I2CnT_DMA 发送地址寄存器（DMA_I2CnT_TXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CT_TXAH	7EFA9DH	I2CT_TXA[15:8]							
DMA_I2CT_TXAL	7EFA9EH	I2CT_TXA[7:0]							
DMA_I2C2T_TXAH	7EF878H	I2C2T_TXA[15:8]							
DMA_I2C2T_TXAL	7EF879H	I2C2T_TXA[7:0]							

I2CnT_TXA：设置自动发送数据的源地址。执行 I2CnT_DMA 操作时会从这个地址开始读数据。（n=1/2）

33.10.8 I2Cn_DMA 传输周期（DMA_I2Cn_ITVx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2C_ITVH	7EFAC4H	I2C_ITV[15:8]							
DMA_I2C_ITVL	7EFAC5H	I2C_ITV[7:0]							
DMA_I2C2_ITVH	7EF87EH	I2C2_ITV[15:8]							
DMA_I2C2_ITVL	7EF87FH	I2C2_ITV[7:0]							

I2Cn_ITV[15:0]：设置 I2Cn 发送和接收的间隔时间（基于系统时钟）。（n=1/2）

33.10.9 I2CnR_DMA 配置寄存器（DMA_I2CnR_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CR_CFG	7EFAA0H	I2CRIE	-	-	-	I2CRIP[1:0]		I2CRPTY[1:0]	
DMA_I2C2R_CFG	7EF880H	I2C2RIE	-	-	-	I2C2RIP[1:0]		I2C2RPTY[1:0]	

I2CnRIE：I2CnR_DMA 中断使能控制位（n=1/2）

0：禁止 I2CnR_DMA 中断

1：允许 I2CnR_DMA 中断

I2CnRIP[1:0]：I2CnR_DMA 中断优先级控制位（n=1/2）

I2CnRIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

I2CnRPTY[1:0]：I2CnR_DMA 数据总线访问优先级控制位（n=1/2）

I2CnRPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.10.10 I2CnR_DMA 控制寄存器（DMA_I2CnR_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CR_CR	7EFAA1H	ENI2CR	TRIG1	-	-	-	-	-	CLRFIFO1
DMA_I2C2R_CR	7EF881H	ENI2C2R	TRIG2	-	-	-	-	-	CLRFIFO2

ENI2CnR：I2CnR_DMA 功能使能控制位（n=1/2）

0：禁止 I2CnR_DMA 功能

1：允许 I2CnR_DMA 功能

TRIGn：I2CnR_DMA 接收触发控制位（n=1/2）

0：写 0 无效

1：写 1 开始 I2CnR_DMA 自动接收数据

CLRFIFOn：清除 I2CnR_DMA 接收 FIFO 控制位（n=1/2）

0：写 0 无效

1：开始 I2CnR_DMA 操作前，先清空 I2CnR_DMA 内置的 FIFO

33.10.11 I2CnR_DMA 状态寄存器（DMA_I2CnR_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CR_STA	7EFAA2H	-	-	-	-	-	-	RXLOSS	I2CRIF
DMA_I2C2R_STA	7EF882H	-	-	-	-	-	-	RXLOSS2	I2C2RIF

I2CnRIF：I2CnR_DMA 中断请求标志位，当 I2CnR_DMA 接收数据完成后，硬件自动将 I2CnRIF 置 1，若使能 I2CnR_DMA 中断则进入中断服务程序。标志位需软件清零（n=1/2）

RXLOSSn：I2CnR_DMA 接收数据丢弃标志位。I2CnR_DMA 操作过程中，当 XRAM 总线过于繁忙，来不及清空 I2CnR_DMA 的接收 FIFO 导致 I2CnR_DMA 接收的数据自动丢弃时，硬件硬件自动将 RXLOSSn 置 1。标志位需软件清零（n=1/2）

33.10.12 I2CnR_DMA 传输总字节寄存器（DMA_I2CnR_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CR_AMTH	7EFAAAH	I2CR_AMT[15:8]							
DMA_I2CR_AMT	7EFAA3H	I2CR_AMT[7:0]							
DMA_I2C2R_AMTH	7EF884H	I2C2R_AMT[15:8]							
DMA_I22CR_AMT	7EF885H	I2C2R_AMT[7:0]							

I2CnR_AMT[15:0]：设置需要进行数据读写的字节数。（n=1/2）

注：实际读写的字节数为（I2CnR_AMT+1），即当 I2CnR_AMT 设置为 0 时，读写 1 字节，当 I2CnR_AMT 设置 255 时，读写 256 字节。特别注意 I2CnR_AMT 设置为 FFFFH 时表示无限循环模式。

33.10.13 I2CnR_DMA 传输完成字节寄存器（DMA_I2CnR_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CR_DONEH	7EFAABH	I2CR_DONE[15:8]							
DMA_I2CR_DONE	7EFAA4H	I2CR_DONE[7:0]							
DMA_I2C2R_DONEH	7EF886H	I2C2R_DONE[15:8]							
DMA_I2C2R_DONE	7EF887H	I2C2R_DONE[7:0]							

I2CnR_DONE[15:0]：当前已经接收完成的字节数。（n=1/2）

33.10.14 I2CnR_DMA 接收地址寄存器（DMA_I2CnR_RXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2CR_RXAH	7EFAA5H	I2CR_RXA[15:8]							
DMA_I2CR_RXAL	7EFAA6H	I2CR_RXA [7:0]							
DMA_I2C2R_RXAH	7EF888H	I2C2R_RXA [15:8]							
DMA_I2C2R_RXAL	7EF889H	I2C2R_RXA [7:0]							

I2CnR_RXA[15:0]：设置自动接收数据的目标地址。执行 I2CnR_DMA 操作时会从这个地址开始写数据。（n=1/2）

33.10.15 I2Cn_DMA 控制寄存器（DMA_I2Cn_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2C_CR	7EFAADH	RDSEL	-	-	-	-	ACKERR	ERRIE	BMMEN
DMA_I2C2_CR	7EF87BH	RDSEL2	-	-	-	-	ACKERR2	ERRIE2	BMMEN2

RDSELn：I2Cn_DMA_ST 寄存器读取功能选择（n=1/2）

ACKERRn：ACK 错误（n=1/2）

0：发送数据后收到的应答是 ACK

1：发送数据后收到的应答是 NAK（需软件清零）

ERRIEn：ACKERRn 中断使能控制位（n=1/2）

0：禁止 ACKERRn 中断

1：允许 ACKERRn 中断（ACKERRn 中断入口地址为 I2Cn 中断入口）

BMMENn：I2Cn 的 DMA 功能使能位（n=1/2）

0：禁止 I2Cn_DMA 功能

1：允许 I2Cn_DMA 功能

33.10.16 I2Cn_DMA 状态寄存器（DMA_I2Cn_ST）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2C_ST1	7EFAAEH	COUNT[7:0]							
DMA_I2C_ST2	7EFAAFH	COUNT[15:8]							
DMA_I2C2_ST1	7EF87CH	COUNT2[7:0]							
DMA_I2C2_ST2	7EF87DH	COUNT2[15:8]							

COUNTn[15:0]：I2Cn_DMA 传输字节控制（n=1/2）

写寄存器：设置 I2Cn_DMA 传输字节数

读寄存器：RDSELn＝0 时，COUNTn 为需要传输的字节数

RDSELn＝1 时，COUNTn 为已经传输完成的字节数

33.11 I2S1/2 与存储器之间的数据交换（I2SnT_DMA，I2SnR_DMA）

33.11.1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_I2ST_CFG	I2ST_DMA 配置寄存器	7EFAB0H	I2STIE	-	-	-	I2STIP[1:0]		I2STPTY[1:0]		0xxx,0000
DMA_I2ST_CR	I2ST_DMA 控制寄存器	7EFAB1H	ENI2ST	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_I2ST_STA	I2ST_DMA 状态寄存器	7EFAB2H	-	-	-	-	-	TXOVW	-	I2STIF	xxxx,x0x0
DMA_I2ST_AMTH	I2ST_DMA 传输总字节数	7EFAC0H	I2ST_AMT[15:8]								0000,0000
DMA_I2ST_AMT	I2ST_DMA 传输总字节数	7EFAB3H	I2ST_AMT[7:0]								0000,0000
DMA_I2ST_DONEH	I2ST_DMA 传输完成字节数	7EFAC1H	I2ST_DONE[15:8]								0000,0000
DMA_I2ST_DONE	I2ST_DMA 传输完成字节数	7EFAB4H	I2ST_DONE[7:0]								0000,0000
DMA_I2ST_TXAH	I2ST_DMA 发送高地址	7EFAB5H	I2ST_TXA[15:8]								0000,0000
DMA_I2ST_TXAL	I2ST_DMA 发送低地址	7EFAB6H	I2ST_TXA[7:0]								0000,0000
DMA_I2S_ITVH	I2S_DMA 时间间隔寄存器	7EFAC6H	I2S_ITV[15:8]								0000,0000
DMA_I2S_ITVL	I2S_DMA 时间间隔寄存器	7EFAC7H	I2S_ITV[7:0]								0000,0000
DMA_I2SR_CFG	I2SR_DMA 配置寄存器	7EFAB8H	I2SRIE	-	-	-	I2SRIP[1:0]		I2SRPTY[1:0]		0xxx,0000
DMA_I2SR_CR	I2SR_DMA 控制寄存器	7EFAB9H	ENI2SR	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_I2SR_STA	I2SR_DMA 状态寄存器	7EFABAH	-	-	-	-	-	-	RXLOSS	I2SRIF	xxxx,xx00
DMA_I2SR_AMTH	I2SR_DMA 传输总字节数	7EFAC2H	I2SR_AMT[15:8]								0000,0000
DMA_I2SR_AMT	I2SR_DMA 传输总字节数	7EFABBH	I2SR_AMT[7:0]								0000,0000
DMA_I2SR_DONEH	I2SR_DMA 传输完成字节数	7EFAC3H	I2SR_DONE[15:8]								0000,0000
DMA_I2SR_DONE	I2SR_DMA 传输完成字节数	7EFABCH	I2SR_DONE[7:0]								0000,0000
DMA_I2SR_RXAH	I2SR_DMA 接收高地址	7EFABDH	I2SR_RXA[15:8]								0000,0000
DMA_I2SR_RXAL	I2SR_DMA 接收低地址	7EFABEH	I2SR_RXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_I2S2T_CFG	I2S2T_DMA 配置寄存器	7EF890H	I2S2TIE	-	-	-	I2S2TIP[1:0]		I2S2TPTY[1:0]		0xxx,0000
DMA_I2S2T_CR	I2S2T_DMA 控制寄存器	7EF891H	ENI2S2T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_I2S2T_STA	I2S2T_DMA 状态寄存器	7EF892H	-	-	-	-	-	TXOVW	-	I2S2TIF	xxxx,x0x0
DMA_I2S2T_AMTH	I2S2T_DMA 传输总字节数	7EF894H	I2S2T_AMT[15:8]								0000,0000
DMA_I2S2T_AMT	I2S2T_DMA 传输总字节数	7EF895H	I2S2T_AMT[7:0]								0000,0000
DMA_I2S2T_DONEH	I2S2T_DMA 传输完成字节数	7EF896H	I2S2T_DONE[15:8]								0000,0000
DMA_I2S2T_DONE	I2S2T_DMA 传输完成字节数	7EF897H	I2S2T_DONE[7:0]								0000,0000
DMA_I2S2T_TXAH	I2S2T_DMA 发送高地址	7EF898H	I2S2T_TXA[15:8]								0000,0000
DMA_I2S2T_TXAL	I2S2T_DMA 发送低地址	7EF899H	I2S2T_TXA[7:0]								0000,0000
DMA_I2S2_ITVH	I2S2T_DMA 发送高地址	7EF89EH	I2S2T_TXA[15:8]								0000,0000
DMA_I2S2_ITVL	I2S2T_DMA 发送低地址	7EF89FH	I2S2T_TXA[7:0]								0000,0000
DMA_I2S2R_CFG	I2S2R_DMA 配置寄存器	7EF8A0H	I2S2RIE	-	-	-	I2S2RIP[1:0]		I2S2RPTY[1:0]		0xxx,0000
DMA_I2S2R_CR	I2S2R_DMA 控制寄存器	7EF8A1H	ENI2S2R	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_I2S2R_STA	I2S2R_DMA 状态寄存器	7EF8A2H	-	-	-	-	-	-	RXLOSS	I2S2RIF	xxxx,xx00
DMA_I2S2R_AMTH	I2S2R_DMA 传输总字节数	7EF8A4H	I2S2R_AMT[15:8]								0000,0000
DMA_I2S2R_AMT	I2S2R_DMA 传输总字节数	7EF8A5H	I2S2R_AMT[7:0]								0000,0000
DMA_I2S2R_DONEH	I2S2R_DMA 传输完成字节数	7EF8A6H	I2S2R_DONE[15:8]								0000,0000
DMA_I2S2R_DONE	I2S2R_DMA 传输完成字节数	7EF8A7H	I2S2R_DONE[7:0]								0000,0000
DMA_I2S2R_RXAH	I2S2R_DMA 接收高地址	7EF8A8H	I2S2R_RXA[15:8]								0000,0000
DMA_I2S2R_RXAL	I2S2R_DMA 接收低地址	7EF8A9H	I2S2R_RXA[7:0]								0000,0000

33.11.2 I2SnT_DMA 配置寄存器（DMA_I2SnT_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2ST_CFG	7EFAB0H	I2STIE	-	-	-	I2STIP[1:0]		I2STPTY[1:0]	
DMA_I2S2T_CFG	7EF890H	I2S2TIE	-	-	-	I2S2TIP[1:0]		I2S2TPTY[1:0]	

I2SnTIE：I2SnT_DMA 中断使能控制位（n=1/2）

0：禁止 I2SnT_DMA 中断

1：允许 I2SnT_DMA 中断

I2SnTIP[1:0]：I2SnT_DMA 中断优先级控制位（n=1/2）

I2SnTIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

I2SnTPTY[1:0]：I2SnT_DMA 数据总线访问优先级控制位（n=1/2）

I2SnTPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.11.3 I2SnT_DMA 控制寄存器（DMA_I2SnT_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2ST_CR	7EFAB1H	ENI2ST	TRIG1	-	-	-	-	-	-
DMA_I2S2T_CR	7EF891H	ENI2S2T	TRIG2	-	-	-	-	-	-

ENI2SnT：I2SnT_DMA 功能使能控制位（n=1/2）

0：禁止 I2SnT_DMA 功能

1：允许 I2SnT_DMA 功能

TRIGn：I2SnT_DMA 发送触发控制位（n=1/2）

0：写 0 无效

1：写 1 开始 I2SnT_DMA 自动发送数据

33.11.4 I2SnT_DMA 状态寄存器（DMA_I2SnT_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2ST_STA	7EFAB2H	-	-	-	-	-	TXOVW	-	I2STIF
DMA_I2S2T_STA	7EF892H	-	-	-	-	-	TXOVW2	-	I2S2TIF

I2SnTIF：I2SnT_DMA 中断请求标志位，当 I2SnT_DMA 数据发送完成后，硬件自动将 I2SnTIF 置 1，若使能 I2SnT_DMA 中断则进入中断服务程序。标志位需软件清零（n=1/2）

TXOVWn：I2SnT_DMA 数据覆盖标志位。I2SnT_DMA 正在数据传输过程中，写 I2Sn 数据寄存器 I2Sn_DRH 和 I2Snn_DRL 时，会导致数据传输失败，此时硬件硬件自动将 TXOVWn 置 1。标志位需软件清零（n=1/2）

33.11.5 I2SnT_DMA 传输总字节寄存器（DMA_I2SnT_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2ST_AMTH	7EFAC0H	I2ST_AMT[15:8]							
DMA_I2ST_AMT	7EFAB3H	I2ST_AMT[7:0]							
DMA_I2S2T_AMTH	7EF894H	I2S2T_AMT[15:8]							
DMA_I2S2T_AMT	7EF895H	I2S2T_AMT[7:0]							

I2SnT_AMT[15:0]：设置需要进行数据读写的字节数。（n=1/2）

注：实际读写的字节数为（I2SnT_AMT+1），即当 I2SnT_AMT 设置为 0 时，读写 1 字节，当 I2SnT_AMT 设置 255 时，读写 256 字节。特别注意 I2SnT_AMT 设置为 FFFFH 时表示无限循环模式。

33.11.6 I2SnT_DMA 传输完成字节寄存器（DMA_I2SnT_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2ST_DONEH	7EFAC1H	I2ST_DONE[15:8]							
DMA_I2ST_DONE	7EFAB4H	I2ST_DONE[7:0]							
DMA_I2S2T_DONEH	7EF896H	I2S2T_DONE[15:8]							
DMA_I2S2T_DONE	7EF897H	I2S2T_DONE[7:0]							

I2SnT_DONE[15:0]：当前已经发送完成的字节数。（n=1/2）

33.11.7 I2SnT_DMA 发送地址寄存器（DMA_I2SnT_TXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2ST_TXAH	7EFAB5H	I2ST_TXA [15:8]							
DMA_I2ST_TXAL	7EFAB6H	I2ST_TXA [7:0]							
DMA_I2S2T_TXAH	7EF898H	I2S2T_TXA [15:8]							
DMA_I2S2T_TXAL	7EF899H	I2S2T_TXA [7:0]							

I2SnT_TXA[15:0]：设置自动发送数据的源地址。执行 I2SnT_DMA 操作时会从这个地址开始读数据。（n=1/2）

33.11.8 I2Sn_DMA 传输周期（DMA_I2Sn_ITVx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2S_ITVH	7EFAC6H	I2S_ITV[15:8]							
DMA_I2S_ITVL	7EFAC7H	I2S_ITV[7:0]							
DMA_I2S2_ITVH	7EF89EH	I2S2_ITV[15:8]							
DMA_I2S2_ITVL	7EF89FH	I2S2_ITV[7:0]							

DMA_I2Sn_ITV[15:0]：设置 I2Sn 发送和接收的间隔时间（基于系统时钟）。（n=1/2）

33.11.9 I2SnR_DMA 配置寄存器（DMA_I2SnR_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2SR_CFG	7EFAB8H	I2SRIE	-	-	-	I2SRIP[1:0]		I2SRPTY[1:0]	
DMA_I2S2R_CFG	7EF8A0H	I2S2RIE	-	-	-	I2S2RIP[1:0]		I2S2RPTY[1:0]	

I2SnRIE：I2SnR_DMA 中断使能控制位（n=1/2）

0：禁止 I2SnR_DMA 中断

1：允许 I2SnR_DMA 中断

I2SnRIP[1:0]：I2SnR_DMA 中断优先级控制位（n=1/2）

I2SnRIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

I2SnRPTY[1:0]：I2SnR_DMA 数据总线访问优先级控制位（n=1/2）

I2SnRPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.11.10 I2SnR_DMA 控制寄存器（DMA_I2SnR_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2SR_CR	7EFAB9H	ENI2SR	TRIG1	-	-	-	-	-	CLRFIFO1
DMA_I2S2R_CR	7EF8A1H	ENI2S2R	TRIG2	-	-	-	-	-	CLRFIFO2

ENI2SnR：I2SnR_DMA 功能使能控制位（n=1/2）

0：禁止 I2SnR_DMA 功能

1：允许 I2SnR_DMA 功能

TRIGn：I2SnR_DMA 接收触发控制位（n=1/2）

0：写 0 无效

1：写 1 开始 I2SnR_DMA 自动接收数据

CLRFIFOn：清除 I2SnR_DMA 接收 FIFO 控制位（n=1/2）

0：写 0 无效

1：开始 I2SnR_DMA 操作前，先清空 I2SnR_DMA 内置的 FIFO

33.11.11 I2SnR_DMA 状态寄存器（DMA_I2SnR_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2SR_STA	7EFABAH	-	-	-	-	-	-	RXLOSS	I2SRIF
DMA_I2S2R_STA	7EF8A2H	-	-	-	-	-	-	RXLOSS2	I2SnRIF

I2SnRIF：I2SnR_DMA 中断请求标志位，当 I2SnR_DMA 接收数据完成后，硬件自动将 I2SnRIF 置 1，若使能 I2SnR_DMA 中断则进入中断服务程序。标志位需软件清零（n=1/2）

RXLOSSn：I2SnR_DMA 接收数据丢弃标志位。I2SnR_DMA 操作过程中，当 XRAM 总线过于繁忙，来不及清空 I2SnR_DMA 的接收 FIFO 导致 I2SnR_DMA 接收的数据自动丢弃时，硬件硬件自动将 RXLOSSn 置 1。标志位需软件清零（n=1/2）

33.11.12 I2SnR_DMA 传输总字节寄存器（DMA_I2SnR_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2SR_AMTH	7EFAC2H	I2SR_AMT[15:8]							
DMA_I2SR_AMT	7EFABBH	I2SR_AMT[7:0]							
DMA_I2S2R_AMTH	7EF8A4H	I2S2R_AMT[15:8]							
DMA_I2S2R_AMT	7EF8A5H	I2S2R_AMT[7:0]							

I2SnR_AMT[15:0]：设置需要进行数据读写的字节数。（n=1/2）

注：实际读写的字节数为（I2SnR_AMT+1），即当 I2SnR_AMT 设置为 0 时，读写 1 字节，当 I2SnR_AMT 设置 255 时，读写 256 字节。特别注意 I2SnR_AMT 设置为 FFFFH 时表示无限循环模式。

33.11.13 I2SnR_DMA 传输完成字节寄存器（DMA_I2SnR_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2SR_DONEH	7EFAC3H	I2SR_DONE[15:8]							
DMA_I2SR_DONE	7EFABCH	I2SR_DONE[7:0]							
DMA_I2S2R_DONEH	7EF8A6H	I2S2R_DONE[15:8]							
DMA_I2S2R_DONE	7EF8A7H	I2S2R_DONE[7:0]							

I2SnR_DONE[15:0]：当前已经接收完成的字节数。（n=1/2）

33.11.14 I2SnR_DMA 接收地址寄存器（DMA_I2SnR_RXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_I2SR_RXAH	7EFABDH	I2SR_RXA[15:8]							
DMA_I2SR_RXAL	7EFABEH	I2SR_RXA[7:0]							
DMA_I2S2R_RXAH	7EF8A8H	I2S2R_RXA[15:8]							
DMA_I2S2R_RXAL	7EF8A9H	I2S2R_RXA[7:0]							

I2SnR_RXA[15:0]：设置自动接收数据的目标地址。执行 I2SnR_DMA 操作时会从这个地址开始写数据。（n=1/2）

33.12 QSPI 与存储器之间的数据交换（QSPI_DMA）

33.12.1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_QSPI_CFG	QSPI_DMA 配置寄存器	7EFAD0H	QSPIIE	ACT_WR	ACT_RD	-	QSPIIP[1:0]		QSPIPTY[1:0]		000x,0000
DMA_QSPI_CR	QSPI_DMA 控制寄存器	7EFAD1H	ENQSPI	TRIG_W	TRIG_R	-	-	-	CLRWFIFO	CLRRFIFO	000x,xx00
DMA_QSPI_STA	QSPI_DMA 状态寄存器	7EFAD2H	-	-	-	-	-	-	-	QSPIIF	xxxx,xxx0
DMA_QSPI_AMTH	QSPI_DMA 传输总字节数	7EFADBH	QSPI_AMT[15:8]								0000,0000
DMA_QSPI_AMT	QSPI_DMA 传输总字节数	7EFAD3H	QSPI_AMT[7:0]								0000,0000
DMA_QSPI_DONEH	QSPI_DMA 传输完成字节数	7EFADCH	QSPI_DONE[15:8]								0000,0000
DMA_QSPI_DONE	QSPI_DMA 传输完成字节数	7EFAD4H	QSPI_DONE[7:0]								0000,0000
DMA_QSPI_TXAH	QSPI_DMA 发送高地址	7EFAD5H	QSPI_TXA[15:8]								0000,0000
DMA_QSPI_TXAL	QSPI_DMA 发送低地址	7EFAD6H	QSPI_TXA[7:0]								0000,0000
DMA_QSPI_RXAH	QSPI_DMA 接收高地址	7EFAD7H	QSPI_RXA[15:8]								0000,0000
DMA_QSPI_RXAL	QSPI_DMA 接收低地址	7EFAD8H	QSPI_RXA[7:0]								0000,0000
DMA_QSPI_ITVH	QSPI_DMA 时间间隔寄存器	7EFADEH	QSPI_ITV[15:8]								0000,0000
DMA_QSPI_ITVL	QSPI_DMA 时间间隔寄存器	7EFADFH	QSPI_ITV[7:0]								0000,0000

33.12.2 QSPI_DMA 配置寄存器（DMA_QSPI_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_QSPI_CFG	7EFAD0H	QSPIIE	ACT_WR	ACT_RD	-	QSPIIP[1:0]		QSPIPTY[1:0]	

QSPIIE：QSPI_DMA 中断使能控制位

0：禁止 QSPI_DMA 中断

1：允许 QSPI_DMA 中断

ACT_WR：QSPI_DMA 写数据控制位

0：禁止 QSPI_DMA 写数据。

1：允许 QSPI_DMA 写数据。

ACT_RD：QSPI_DMA 读数据控制位

0：禁止 QSPI_DMA 读数据。

1：允许 QSPI_DMA 读数据。

QSPIIP[1:0]：QSPI_DMA 中断优先级控制位

QSPIIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

QSPIPTY[1:0]：QSPI_DMA 数据总线访问优先级控制位

QSPIPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.12.3 QSPI_DMA 控制寄存器（DMA_QSPI_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_QSPI_CR	7EFAD1H	ENQSPI	TRIG_W	TRIG_R	-	-	-	CLRWFIFO	CLRRFIFO

ENQSPI：QSPI_DMA 功能使能控制位

0：禁止 QSPI_DMA 功能

1：允许 QSPI_DMA 功能

TRIG_W：QSPI_DMA 触发写控制位

0：写 0 无效

1：写 1 开始 QSPI_DMA 写操作，

TRIG_R：QSPI_DMA 触发读控制位

0：写 0 无效

1：写 1 开始 QSPI_DMA 读操作，

CLRWFIFO：清除 QSPI_DMA 写 FIFO 控制位

0：写 0 无效

1：开始 QSPI_DMA 操作前，先清空 QSPI_DMA 内置的 FIFO

CLRRFIFO：清除 QSPI_DMA 读 FIFO 控制位

0：写 0 无效

1：开始 QSPI_DMA 操作前，先清空 QSPI_DMA 内置的 FIFO

33.12.4 QSPI_DMA 状态寄存器（DMA_QSPI_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_QSPI_STA	7EFAD2H	-	-	-	-	-	-	-	QSPIIF

QSPIIF：QSPI_DMA 中断请求标志位，当 QSPI_DMA 数据交换完成后，硬件自动将 QSPIIF 置 1，若使能 QSPI_DMA 中断则进入中断服务程序。标志位需软件清零

33.12.5 QSPI_DMA 传输总字节寄存器（DMA_QSPI_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_QSPI_AMTH	7EFADBH	QSPI_AMT[15:8]							
DMA_QSPI_AMT	7EFAD3H	QSPI_AMT[7:0]							

QSPI_AMT[15:0]：设置需要进行数据读写的字节数。

注：实际读写的字节数为（QSPI_AMT+1），即当 QSPI_AMT 设置为 0 时，读写 1 字节，当 QSPI_AMT 设置 255 时，读写 256 字节。特别注意 QSPI_AMT 设置为 FFFFH 时表示无限循环模式。

33.12.6 QSPI_DMA 传输完成字节寄存器（DMA_QSPI_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_QSPI_DONEH	7EFADCH	QSPI_DONE[15:8]							
DMA_QSPI_DONE	7EFAD4H	QSPI_DONE[7:0]							

QSPI_DONE[15:0]：当前已经传输完成的字节数。

33.12.7 QSPI_DMA 发送地址寄存器（DMA_QSPI_TXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_QSPI_TXAH	7EFAD5H	QSPI_TXA[15:8]							
DMA_QSPI_TXAL	7EFAD6H	QSPI_TXA[7:0]							

QSPI_TXA[15:0]：设置进行数据传输时的源地址。执行 QSPI_DMA 操作时会从这个地址开始读数据。

33.12.8 QSPI_DMA 接收地址寄存器（DMA_QSPI_RXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_QSPI_RXAH	7EFAD7H	QSPI_RXA[15:8]							
DMA_QSPI_RXAL	7EFAD8H	QSPI_RXA[7:0]							

QSPI_RXA[15:0]：设置进行数据传输时的目标地址。执行 QSPI_DMA 操作时会从这个地址开始写入数据。

33.12.9 QSPI_DMA 传输周期（DMA_QSPI_ITVx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_QSPI_ITVH	7EFADEH	QSPI_ITV[15:8]							
DMA_QSPI_ITVL	7EFADFH	QSPI_ITV[7:0]							

QSPI_ITV[15:0]：设置 QSPI 传输的间隔时间（基于系统时钟）。

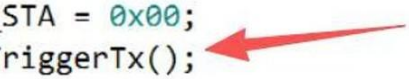
33.13 PWMA/C/E 与存储器之间的数据交换（PWMnT_DMA，PWMnR_DMA）

33.13.1 PWM 的 DMA 在操作时的注意事项

PWM 的 DMA 在操作的时候，触发 DMA 启动是有时序要求的：

在 PWM-DMA 的完成中断中，打开下一次的 DMA 触发请求
这样，只需要关心对应数组中的变量更新即可
例如，在配置好 DMA 后，初始化中给一个触发启动 DMA
后续依靠 DMA 完成中断的 DMA 再次触发来循环执行，后续主循环中只需要更改对应的数组变量

```
5  void pwma_dma_isr(void) interrupt DMA_PWMAT_VECTOR
6  {
7      DMA_PWMAT_STA = 0x00;
8      DMA_PWMA_TriggerTx();
9  }
```



中断内触发下一次

33.13.2 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PWMA_DER	PWMA_DMA 事件使能寄存器	7EF948H	-	TDEA	COMDEA	CCA4DE	CCA3DE	CCA2DE	CCA1DE	UDEA	x000,0000
PWMA_DBA	PWMA_DMA 基址寄存器	7EF949H	-	-	-	DBAA[4:0]					xxx0,0000
PWMA_DBL	PWMA_DMA 传输长度寄存器	7EF94AH	-	-	-	DBLA[4:0]					xxx0,0000
PWMA_DMACR	PWMA_DMA 控制寄存器	7EF94BH	-	-	-	DSKIPA	DDIRA	DMAENA	SIZEA[1:0]		xxx0,0000
DMA_PWMAT_CFG	PWMAT_DMA 配置寄存器	7EF980H	PWMATIE	-	-	-	PWMATIP[1:0]		PWMATPTY[1:0]		0xxx,0000
DMA_PWMAT_CR	PWMAT_DMA 控制寄存器	7EF981H	ENPWMAT	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_PWMAT_STA	PWMAT_DMA 状态寄存器	7EF982H	RDOV	-	-	-	-	TXOVW	-	PWMATIF	0xxx,x0x0
DMA_PWMAT_AMTH	PWMAT_DMA 传输总字节数	7EF984H	PWMAT_AMT[15:8]								0000,0000
DMA_PWMAT_AMT	PWMAT_DMA 传输总字节数	7EF985H	PWMAT_AMT[7:0]								0000,0000
DMA_PWMAT_DONEH	PWMAT_DMA 传输完成字节	7EF986H	PWMAT_DONE[15:8]								0000,0000
DMA_PWMAT_DONE	PWMAT_DMA 传输完成字节	7EF987H	PWMAT_DONE[7:0]								0000,0000
DMA_PWMAT_TXAH	PWMAT_DMA 发送高地址	7EF988H	PWMAT_TXA[15:8]								0000,0000
DMA_PWMAT_TXAL	PWMAT_DMA 发送低地址	7EF989H	PWMAT_TXA[7:0]								0000,0000
DMA_PWMA_ITVH	PWMA_DMA 时间间隔寄存器	7EF98EH	PWMA_ITV[15:8]								0000,0000
DMA_PWMA_ITVL	PWMA_DMA 时间间隔寄存器	7EF98FH	PWMA_ITV[7:0]								0000,0000
DMA_PWMAR_CFG	PWMAR_DMA 配置寄存器	7EF990H	PWMARIE	-	-	-	PWMARIP[1:0]		PWMARPTY[1:0]		0xxx,0000
DMA_PWMAR_CR	PWMAR_DMA 控制寄存器	7EF991H	ENPWMAR	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_PWMAR_STA	PWMAR_DMA 状态寄存器	7EF992H	WROV	-	-	-	-	-	RXLOSS	PWMARIF	0xxx,xx00
DMA_PWMAR_AMTH	PWMAR_DMA 传输总字节数	7EF994H	PWMAR_AMT[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_PWMAR_AMT	PWMAR_DMA 传输总字节数	7EF995H	PWMAR_AMT[7:0]								0000,0000
DMA_PWMAR_DONEH	PWMAR_DMA 传输完成字节	7EF996H	PWMAR_DONE[15:8]								0000,0000
DMA_PWMAR_DONE	PWMAR_DMA 传输完成字节	7EF997H	PWMAR_DONE[7:0]								0000,0000
DMA_PWMAR_RXAH	PWMAR_DMA 接收高地址	7EF998H	PWMAR_RXA[15:8]								0000,0000
DMA_PWMAR_RXAL	PWMAR_DMA 接收低地址	7EF999H	PWMAR_RXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PWMC_DER	PWMC_DMA 事件使能寄存器	7EF968H	-	TDEC	COMDEC	CCC4DE	CCC3DE	CCC2DE	CCC1DE	UDEC	x000,0000
PWMC_DBA	PWMC_DMA 基址寄存器	7EF969H	-	-	-	DBAC[4:0]					xxx0,0000
PWMC_DBL	PWMC_DMA 传输长度寄存器	7EF96AH	-	-	-	DBLC[4:0]					xxx0,0000
PWMC_DMACR	PWMC_DMA 控制寄存器	7EF96BH	-	-	-	DSKIPC	DDIRC	DMAENC	SIZEC[1:0]		xxx0,0000
DMA_PWMCT_CFG	PWMCT_DMA 配置寄存器	7EF9A0H	PWMCTIE	-	-	-	PWMCTIP[1:0]		PWMCTPTY[1:0]		0xxx,0000
DMA_PWMCT_CR	PWMCT_DMA 控制寄存器	7EF9A1H	ENPWMCT	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_PWMCT_STA	PWMCT_DMA 状态寄存器	7EF9A2H	RDOV	-	-	-	-	TXOVW	-	PWMCTIF	0xxx,x0x0
DMA_PWMCT_AMTH	PWMCT_DMA 传输总字节数	7EF9A4H	PWMCT_AMT[15:8]								0000,0000
DMA_PWMCT_AMT	PWMCT_DMA 传输总字节数	7EF9A5H	PWMCT_AMT[7:0]								0000,0000
DMA_PWMCT_DONEH	PWMCT_DMA 传输完成字节	7EF9A6H	PWMCT_DONE[15:8]								0000,0000
DMA_PWMCT_DONE	PWMCT_DMA 传输完成字节	7EF9A7H	PWMCT_DONE[7:0]								0000,0000
DMA_PWMCT_TXAH	PWMCT_DMA 发送高地址	7EF9A8H	PWMCT_TXA[15:8]								0000,0000
DMA_PWMCT_TXAL	PWMCT_DMA 发送低地址	7EF9A9H	PWMCT_TXA[7:0]								0000,0000
DMA_PWMC_ITVH	PWMC_DMA 时间间隔寄存器	7EF9AEH	PWMC_ITV[15:8]								0000,0000
DMA_PWMC_ITVL	PWMC_DMA 时间间隔寄存器	7EF9AFH	PWMC_ITV[7:0]								0000,0000
DMA_PWMCR_CFG	PWMCR_DMA 配置寄存器	7EF9B0H	PWMCRIE	-	-	-	PWMCRIP[1:0]		PWMCRPTY[1:0]		0xxx,0000
DMA_PWMCR_CR	PWMCR_DMA 控制寄存器	7EF9B1H	ENPWMCR	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_PWMCR_STA	PWMCR_DMA 状态寄存器	7EF9B2H	WROV	-	-	-	-	-	RXLOSS	PWMCRIF	0xxx,xx00
DMA_PWMCR_AMTH	PWMCR_DMA 传输总字节数	7EF9B4H	PWMCR_AMT[15:8]								0000,0000
DMA_PWMCR_AMT	PWMCR_DMA 传输总字节数	7EF9B5H	PWMCR_AMT[7:0]								0000,0000
DMA_PWMCR_DONEH	PWMCR_DMA 传输完成字节	7EF9B6H	PWMCR_DONE[15:8]								0000,0000
DMA_PWMCR_DONE	PWMCR_DMA 传输完成字节	7EF9B7H	PWMCR_DONE[7:0]								0000,0000
DMA_PWMCR_RXAH	PWMCR_DMA 接收高地址	7EF9B8H	PWMCR_RXA[15:8]								0000,0000
DMA_PWMCR_RXAL	PWMCR_DMA 接收低地址	7EF9B9H	PWMCR_RXA[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
PWME_DER	PWME_DMA 事件使能寄存器	7EF6A8H	-	TDEE	COMDEE	CCE4DE	CCE3DE	CCE2DE	CCE1DE	UDEE	x000,0000
PWME_DBA	PWME_DMA 基址寄存器	7EF6A9H	-	-	-	DBAE[4:0]					xxx0,0000
PWME_DBL	PWME_DMA 传输长度寄存器	7EF6AAH	-	-	-	DBLE[4:0]					xxx0,0000
PWME_DMACR	PWME_DMA 控制寄存器	7EF6ABH	-	-	-	DSKIPE	DDIRE	DMAENE	SIZEE[1:0]		xxx0,0000
DMA_PWMET_CFG	PWMET_DMA 配置寄存器	7EF668H	PWMETIE	-	-	-	PWMETIP[1:0]		PWMETPTY[1:0]		0xxx,0000
DMA_PWMET_CR	PWMET_DMA 控制寄存器	7EF669H	ENPWMET	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_PWMET_STA	PWMET_DMA 状态寄存器	7EF66AH	RDOV	-	-	-	-	TXOVW	-	PWMETIF	0xxx,x0x0

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_PWMET_AMTH	PWMET_DMA 传输总字节数	7EF66CH	PWMET_AMT[15:8]								0000,0000
DMA_PWMET_AMT	PWMET_DMA 传输总字节数	7EF66DH	PWMET_AMT[7:0]								0000,0000
DMA_PWMET_DONEH	PWMET_DMA 传输完成字节	7EF66EH	PWMET_DONE[15:8]								0000,0000
DMA_PWMET_DONE	PWMET_DMA 传输完成字节	7EF66FH	PWMET_DONE[7:0]								0000,0000
DMA_PWMET_TXAH	PWMET_DMA 发送高地址	7EF670H	PWMET_TXA[15:8]								0000,0000
DMA_PWMET_TXAL	PWMET_DMA 发送低地址	7EF671H	PWMET_TXA[7:0]								0000,0000
DMA_PWME_ITVH	PWME_DMA 时间间隔寄存器	7EF676H	PWME_ITV[15:8]								0000,0000
DMA_PWME_ITVL	PWME_DMA 时间间隔寄存器	7EF677H	PWME_ITV[7:0]								0000,0000
DMA_PWMER_CFG	PWMER_DMA 配置寄存器	7EF678H	PWMERIE	-	-	-	PWMERIP[1:0]		PWMERPTY[1:0]		0xxx,0000
DMA_PWMER_CR	PWMER_DMA 控制寄存器	7EF679H	ENPWMER	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_PWMER_STA	PWMER_DMA 状态寄存器	7EF67AH	WROV	-	-	-	-	-	RXLOSS	PWMERIF	0xxx,xx00
DMA_PWMER_AMTH	PWMER_DMA 传输总字节数	7EF67CH	PWMER_AMT[15:8]								0000,0000
DMA_PWMER_AMT	PWMER_DMA 传输总字节数	7EF67DH	PWMER_AMT[7:0]								0000,0000
DMA_PWMER_DONEH	PWMER_DMA 传输完成字节	7EF67EH	PWMER_DONE[15:8]								0000,0000
DMA_PWMER_DONE	PWMER_DMA 传输完成字节	7EF67FH	PWMER_DONE[7:0]								0000,0000
DMA_PWMER_RXAH	PWMER_DMA 接收高地址	7EF680H	PWMER_RXA[15:8]								0000,0000
DMA_PWMER_RXAL	PWMER_DMA 接收低地址	7EF681H	PWMER_RXA[7:0]								0000,0000

33.13.3 PWMnT_DMA 配置寄存器（DMA_PWMnT_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAT_CFG	7EF980H	PWMATIE	-	-	-	PWMATIP[1:0]		PWMATPTY[1:0]	
DMA_PWMCT_CFG	7EF9A0H	PWMCTIE	-	-	-	PWMCTIP[1:0]		PWMCTPTY[1:0]	
DMA_PWMET_CFG	7EF668H	PWMETIE	-	-	-	PWMETIP[1:0]		PWMETPTY[1:0]	

PWMnTIE：PWMnT_DMA 中断使能控制位（n=A/C/E）

0：禁止 PWMnT_DMA 中断

1：允许 PWMnT_DMA 中断

PWMnTIP[1:0]：PWMnT_DMA 中断优先级控制位（n=A/C/E）

PWMnTIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

PWMnTPTY[1:0]：PWMnT_DMA 数据总线访问优先级控制位（n=A/C/E）

PWMnTPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.13.4 PWMnT_DMA 控制寄存器（DMA_PWMnT_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAT_CR	7EF981H	ENPWMAT	TRIGA	-	-	-	-	-	-
DMA_PWMCT_CR	7EF9B1H	ENPWMCT	TRIGC	-	-	-	-	-	-
DMA_PWMET_CR	7EF669H	ENPWMET	TRIGE	-	-	-	-	-	-

ENPWMnT：PWMnT_DMA 功能使能控制位（n=A/C/E）

0：禁止 PWMnT_DMA 功能

1：允许 PWMnT_DMA 功能

TRIGn：PWMnT_DMA 发送触发控制位（n=A/C/E）

0：写 0 无效

1：写 1 开始 PWMnT_DMA 自动发送数据

33.13.5 PWMnT_DMA 状态寄存器（DMA_PWMnT_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAT_STA	7EF982H	RDOVA	-	-	-	-	TXOVWA	-	PWMATIF
DMA_PWMCT_STA	7EF9B2H	RDOVC	-	-	-	-	TXOVWC	-	PWMCTIF
DMA_PWMET_STA	7EF66AH	RDOVE	-	-	-	-	TXOVWE	-	PWMETIF

PWMnTIF：PWMnT_DMA 中断请求标志位，当 PWMnT_DMA 数据发送完成后，硬件自动将 PWMnTIF 置 1，若使能 PWMnT_DMA 中断则进入中断服务程序。标志位需软件清零（n=A/C/E）

TXOVWn：PWMnT_DMA 数据覆盖标志位。PWMnT_DMA 正在数据传输过程中，写 PWMn 数据寄存器 PWMnTXD 时，会导致数据传输失败，此时硬件硬件自动将 TXOVWn 置 1。标志位需软件清零（n=A/C/E）

RDOVn：当源 XRAM 地址超过当前 XRAM 所在地址的 64K 时，RDOVn 被置 1。（只读位）（n=A/C/E）

33.13.6 PWMnT_DMA 传输总字节寄存器（DMA_PWMnT_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAT_AMTH	7EF984H	PWMAT_AMT[15:8]							
DMA_PWMAT_AMT	7EF985H	PWMAT_AMT[7:0]							
DMA_PWMCT_AMTH	7EF9A4H	PWMCT_AMT[15:8]							
DMA_PWMCT_AMT	7EF9A5H	PWMCT_AMT[7:0]							
DMA_PWMET_AMTH	7EF66CH	PWMET_AMT[15:8]							
DMA_PWMET_AMT	7EF66DH	PWMET_AMT[7:0]							

PWMnT_AMT[15:0]：设置需要进行数据读写的字节数。（n=A/C/E）

注：实际读写的字节数为（PWMnT_AMT+1），即当 PWMnT_AMT 设置为 0 时，读写 1 字节，当 PWMnT_AMT 设置 255 时，读写 256 字节。特别注意 PWMnT_AMT 设置为 FFFFH 时表示无限循环模式。

33.13.7 PWMnT_DMA 传输完成字节寄存器（DMA_PWMnT_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAT_DONEH	7EF986H	PWMAT_DONE[15:8]							
DMA_PWMAT_DONE	7EF987H	PWMAT_DONE[7:0]							
DMA_PWMCT_DONEH	7EF9A6H	PWMCT_DONE[15:8]							
DMA_PWMCT_DONE	7EF9A7H	PWMCT_DONE[7:0]							
DMA_PWMET_DONEH	7EF66EH	PWMET_DONE[15:8]							
DMA_PWMET_DONE	7EF66FH	PWMET_DONE[7:0]							

PWMnT_DONE[15:0]：当前已经发送完成的字节数。（n=A/C/E）

33.13.8 PWMnT_DMA 发送地址寄存器（DMA_PWMnT_TXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAT_TXAH	7EF988H	PWMAT_TXA[15:8]							
DMA_PWMAT_TXAL	7EF989H	PWMAT_TXA[7:0]							
DMA_PWMCT_TXAH	7EF9A8H	PWMCT_TXA[15:8]							
DMA_PWMCT_TXAL	7EF9A9H	PWMCT_TXA[7:0]							
DMA_PWMET_TXAH	7EF670H	PWMET_TXA[15:8]							
DMA_PWMET_TXAL	7EF671H	PWMET_TXA[7:0]							

PWMnT_TXA[15:0]：设置自动发送数据的源地址。执行 PWMnT_DMA 操作时会从这个地址开始读数据。（n=A/C/E）

33.13.9 PWMn_DMA 传输周期（DMA_PWMn_ITVx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMA_ITVH	7EF98EH	PWMA_ITV[15:8]							
DMA_PWMA_ITVL	7EF98FH	PWMA_ITV[7:0]							
DMA_PWMC_ITVH	7EF9AEH	PWMC_ITV[15:8]							
DMA_PWMC_ITVL	7EF9AFH	PWMC_ITV[7:0]							
DMA_PWME_ITVH	7EF676H	PWME_ITV[15:8]							
DMA_PWME_ITVL	7EF677H	PWME_ITV[7:0]							

PWMn_ITV[15:0]：设置 PWMn 发送和接收的间隔时间（基于系统时钟）。（n=A/C/E）

33.13.10 PWMnR_DMA 配置寄存器（DMA_PWMnR_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAR_CFG	7EF990H	PWMARIE	-	-	-	PWMARIP[1:0]		PWMARPTY[1:0]	
DMA_PWMCR_CFG	7EF9B0H	PWMCRIE	-	-	-	PWMCRIP[1:0]		PWMCRPTY[1:0]	
DMA_PWMER_CFG	7EF678H	PWMERIE	-	-	-	PWMERIP[1:0]		PWMERPTY[1:0]	

PWMnRIE：PWMnR_DMA 中断使能控制位（n=A/C/E）

0：禁止 PWMnR_DMA 中断

1：允许 PWMnR_DMA 中断

PWMnRIP[1:0]：PWMnR_DMA 中断优先级控制位（n=A/C/E）

PWMnRIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

PWMnRPTY[1:0]：PWMnR_DMA 数据总线访问优先级控制位（n=A/C/E）

PWMnRPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.13.11 PWMnR_DMA 控制寄存器（DMA_PWMnR_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAR_CR	7EF991H	ENPWMAR	TRIGA	-	-	-	-	-	CLRFIFOA
DMA_PWMCR_CR	7EF9B1H	ENPWMCR	TRIGC	-	-	-	-	-	CLRFIFOC
DMA_PWMER_CR	7EF679H	ENPWMER	TRIGE	-	-	-	-	-	CLRFIFOE

ENPWMnR：PWMnR_DMA 功能使能控制位（n=A/C/E）

0：禁止 PWMnR_DMA 功能

1：允许 PWMnR_DMA 功能

TRIGn：PWMnR_DMA 接收触发控制位（n=A/C/E）

0：写 0 无效

1：写 1 开始 PWMnR_DMA 自动接收数据

CLRFIFOn：清除 PWMnR_DMA 接收 FIFO 控制位（n=A/C/E）

0：写 0 无效

1：开始 PWMnR_DMA 操作前，先清空 PWMnR_DMA 内置的 FIFO

33.13.12 PWMnR_DMA 状态寄存器（DMA_PWMnR_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAR_STA	7EF992H	WROVA	-	-	-	-	-	RXLOSSA	PWMARIF
DMA_PWMCR_STA	7EF9B2H	WROVC	-	-	-	-	-	RXLOSSC	PWMCRIF
DMA_PWMER_STA	7EF67AH	WROVE	-	-	-	-	-	RXLOSSE	PWMERIF

PWMnRIF：PWMnR_DMA 中断请求标志位，当 PWMnR_DMA 接收数据完成后，硬件自动将 PWMnRIF 置 1，若使能 PWMnR_DMA 中断则进入中断服务程序。标志位需软件清零（n=A/C/E）

RXLOSSn：PWMnR_DMA 接收数据丢弃标志位。PWMnR_DMA 操作过程中，当 XRAM 总线过于繁忙，来不及清空 PWMnR_DMA 的接收 FIFO 导致 PWMnR_DMA 接收的数据自动丢弃时，硬件硬件自动将 RXLOSSn 置 1。标志位需软件清零（n=A/C/E）

WROVn：当目标 XRAM 地址超过当前 XRAM 所在地址的 64K 时，WROVn 被置 1。（只读位）（n=A/C/E）

33.13.13 PWMnR_DMA 传输总字节寄存器（DMA_PWMnR_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAR_AMTH	7EF994H	PWMAR_AMT[15:8]							
DMA_PWMAR_AMT	7EF995H	PWMAR_AMT[7:0]							
DMA_PWMCR_AMTH	7EF9B4H	PWMCR_AMT[15:8]							
DMA_PWMCR_AMT	7EF9B5H	PWMCR_AMT[7:0]							
DMA_PWMER_AMTH	7EF67CH	PWMER_AMT[15:8]							
DMA_PWMER_AMT	7EF67DH	PWMER_AMT[7:0]							

PWMnR_AMT[15:0]：设置需要进行数据读写的字节数。（n=A/C/E）

注：实际读写的字节数为（PWMnR_AMT+1），即当 PWMnR_AMT 设置为 0 时，读写 1 字节，当 PWMnR_AMT 设置 255 时，读写 256 字节。特别注意 PWMnR_AMT 设置为 FFFFH 时表示无限循环模式。

33.13.14 PWMnR_DMA 传输完成字节寄存器（DMA_PWMnR_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAR_DONEH	7EF996H	PWMAR_DONE[15:8]							
DMA_PWMAR_DONE	7EF997H	PWMAR_DONE[7:0]							
DMA_PWMCR_DONEH	7EF9B6H	PWMCR_DONE[15:8]							
DMA_PWMCR_DONE	7EF9B7H	PWMCR_DONE[7:0]							
DMA_PWMER_DONEH	7EF67EH	PWMER_DONE[15:8]							
DMA_PWMER_DONE	7EF67FH	PWMER_DONE[7:0]							

PWMnR_DONE[15:0]：当前已经接收完成的字节数。（n=A/C/E）

33.13.15 PWMnR_DMA 接收地址寄存器（DMA_PWMnR_RXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_PWMAR_RXAH	7EF998H	PWMAR_RXA[15:8]							
DMA_PWMAR_RXAL	7EF999H	PWMAR_RXA[7:0]							
DMA_PWMCR_RXAH	7EF9B8H	PWMCR_RXA[15:8]							
DMA_PWMCR_RXAL	7EF9B9H	PWMCR_RXA[7:0]							
DMA_PWMER_RXAH	7EF680H	PWMER_RXA[15:8]							
DMA_PWMER_RXAL	7EF681H	PWMER_RXA[7:0]							

PWMnR_RXA[15:0]：设置自动接收数据的目标地址。执行 PWMnR_DMA 操作时会从这个地址开始写数据。（n=A/C/E）

33.13.16 PWMn_DMA 事件使能寄存器（PWMn_DER）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_DER	7EF948H	-	TDE	COMDEA	CC4DEA	CC3DEA	CC2DEA	CC1DEA	UDEA
PWMC_DER	7EF968H	-	TDE	COMDEC	CC4DEC	CC3DEC	CC2DEC	CC1DEC	UDEC
PWME_DER	7EF6A8H	-	TDEE	COMDEE	CC4DEE	CC3DEE	CC2DEE	CC1DEE	UDEE

TDEn：使能 PWMn 触发中断为 PWMn_DMA 触发事件（n=A/C/E）

COMDEn：使能 PWMn 的 COM 中断为 PWMn_DMA 触发事件（n=A/C/E）

CC4DEn：使能 PWMn 的捕获/比较 4 中断为 PWMn_DMA 触发事件（n=A/C/E）

CC3DEn：使能 PWMn 的捕获/比较 3 中断为 PWMn_DMA 触发事件（n=A/C/E）

CC2DEn：使能 PWMn 的捕获/比较 2 中断为 PWMn_DMA 触发事件（n=A/C/E）

CC1DEn：使能 PWMn 的捕获/比较 1 中断为 PWMn_DMA 触发事件（n=A/C/E）

UDEn：使能 PWMn 的更新中断为 PWMn_DMA 触发事件（n=A/C/E）

33.13.17 PWMn_DMA 基址寄存器（PWMn_DBA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_DBA	7EF949H	-	-	-	DBAA[4:0]				
PWMC_DBA	7EF969H	-	-	-	DBAC[4:0]				
PWME_DBA	7EF6A9H	-	-	-	DBAE[4:0]				

DBAn[4:0]：设置 PWMn_DMA 的寄存器基址。（n=A/C/E）

PWMn_DMA 基地址映射表（n=A/C/E）

基地址	Byte3	Byte2	Byte1	Byte0
0x00	-	-	-	PWMn_CR1
0x01	-	PWMn_CR3	PWMn_OISR	PWMn_CR2
0x02	-	-	PWMn_ETR	PWMn_SMCR
0x03	-	-	PWMn_DER	PWMn_IER
0x04	-	PWMn_SR3	PWMn_SR2	PWMn_SR1
0x05	-	-	-	PWMn_EGR
0x06	PWMn_CCMR2X	PWMn_CCMR1X	PWMn_CCMR2	PWMn_CCMR1
0x07	PWMn_CCMR4X	PWMn_CCMR3X	PWMn_CCMR4	PWMn_CCMR3
0x08	-	PWMn_CCER3	PWMn_CCER2	PWMn_CCER1
0x09	-	-	PWMn_CNTH	PWMn_CNTL
0x0A	-	-	PWMn_PSCRH	PWMn_PSCRL
0x0B	-	-	PWMn_ARRH	PWMn_ARRL
0x0C	-	-	-	PWMn_RCR
0x0D	-	-	PWMn_CCR1H	PWMn_CCR1L
0x0E	-	-	PWMn_CCR2H	PWMn_CCR2L
0x0F	-	-	PWMn_CCR3H	PWMn_CCR3L
0x10	-	-	PWMn_CCR4H	PWMn_CCR4L
0x11	-	-	PWMn_BKR	PWMn_DTR
0x12	-	-	PWMn_DBL	PWMn_DBA
0x13	-	-	-	-
0x14	-	-	-	-
0x15	PWMn_CCMR6X	PWMn_CCMR5X	PWMn_CCMR6	PWMn_CCMR5
0x16	PWMn_CCR5X	-	PWMn_CCR5H	PWMn_CCR5L
0x17	-	-	PWMn_CCR6H	PWMn_CCR6L
		-	-	-

33.13.18 PWMn_DMA 突发传输次数寄存器（PWMn_DBL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_DBL	7EF94AH	-	-	-	DBLA[4:0]				
PWMC_DBL	7EF96AH	-	-	-	DBLC[4:0]				
PWME_DBL	7EF6AAH	-	-	-	DBLE[4:0]				

DBLn[4:0]：设置 PWMn_DMA 的突发传输（Burst transmission）次数（n=A/C/E）

DBLn[4:0]	长度
00H	1 次
01H	2 次
02H	3 次
…	…
1FH	32 次

33.13.19 PWMn_DMA 控制寄存器（PWMn_DMACR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
PWMA_DMACR	7EF94BH	-	-	-	DSKIPA	DDIRA	DMAENA	SIZEA[1:0]	
PWMC_DMACR	7EF96BH	-	-	-	DSKIPC	DDIRC	DMAENC	SIZEC[1:0]	
PWME_DMACR	7EF6ABH	-	-	-	DSKIPE	DDIRE	DMAENE	SIZEE[1:0]	

DSKIPn：设置是否跳过保留的字节（n=A/C/E）

0：不跳过

1：跳过

DDIRn：PWMn_DMA 方向（n=A/C/E）

0：XRAM 到 PWMn（输出）

1：PWMn 到 XRAM（输入）

DMAENn：PWMn_DMA 使能位（n=A/C/E）

0：禁止 PWMn_DMA

1：使能 PWMn_DAM

SIZEn[1:0]：PWMn DMA 读写字节数（n=A/C/E）

00：读写 1 字节（Byte0）

01：读写 2 字节（Byte1，Byte0）

10：读写 3 字节（Byte2，Byte1，Byte0）

11：读写 4 字节（Byte3，Byte2，Byte1，Byte0）

例如：PWMA_DBA=0x06，PWMA_DBL=1，PWMA_DMACR.SIZE=3，DMA 缓冲区为字节数据 Data。

则当第 1 次 PWMA 的 DMA 更新请求时，

Data[0] 被传送到 PWMA_CCMR2X，

Data[1] 被传送到 PWMA_CCMR1X，

Data[2] 被传送到 PWMA_CCMR2，

Data[3] 被传送到 PWMA_CCMR1，

Data[4] 被传送到 PWMA_CCMR4X，

Data[5] 被传送到 PWMA_CCMR3X，

Data[6] 被传送到 PWMA_CCMR4，

Data[7] 被传送到 PWMA_CCMR3；

当第 2 次 PWMA 的 DMA 更新请求时，

Data[8] 被传送到 PWMA_CCMR2X，

Data[9] 被传送到 PWMA_CCMR1X，

Data[10] 被传送到 PWMA_CCMR2，

Data[11] 被传送到 PWMA_CCMR1，

Data[12] 被传送到 PWMA_CCMR4X，

Data[13] 被传送到 PWMA_CCMR3X，

Data[14] 被传送到 PWMA_CCMR4，

Data[15] 被传送到 PWMA_CCMR3

33.14 DAC1/2 数据自动存储（DACn_DMA）

33.14.1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_DAC1_CFG	DAC1_DMA 配置寄存器	7EF618H	DAC1IE	-	-	-	DAC1IP[1:0]		DAC1PTY[1:0]		0xxx,0000
DMA_DAC1_CR	DAC1_DMA 控制寄存器	7EF619H	ENDAC1	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_DAC1_STA	DAC1_DMA 状态寄存器	7EF61AH	-	-	-	-	-	-	-	DAC1IF	xxxx,x0x0
DMA_DAC1_AMTH	DAC1_DMA 传输总字节数	7EF61CH	DAC1_AMT[15:8]								0000,0000
DMA_DAC1_AMT	DAC1_DMA 传输总字节数	7EF61DH	DAC1_AMT[7:0]								0000,0000
DMA_DAC1_DONEH	DAC1_DMA 传输完成字节数	7EF61EH	DAC1_DONE[15:8]								0000,0000
DMA_DAC1_DONE	DAC1_DMA 传输完成字节数	7EF61FH	DAC1_DONE[7:0]								0000,0000
DMA_DAC1_TXAH	DAC1_DMA 发送高地址	7EF620H	DAC1_TXA[15:8]								0000,0000
DMA_DAC1_TXAL	DAC1_DMA 发送低地址	7EF621H	DAC1_TXA[7:0]								0000,0000
DMA_DAC1_ITVH	DAC1_DMA 时间间隔寄存器	7EF626H	DAC1_ITV[15:8]								0000,0000
DMA_DAC1_ITVL	DAC1_DMA 时间间隔寄存器	7EF627H	DAC1_ITV[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_DAC2_CFG	DAC2_DMA 配置寄存器	7EF630H	DAC2IE	-	-	-	DAC2IP[1:0]		DAC2PTY[1:0]		0xxx,0000
DMA_DAC2_CR	DAC2_DMA 控制寄存器	7EF631H	ENDAC2	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_DAC2_STA	DAC2_DMA 状态寄存器	7EF632H	-	-	-	-	-	-	-	DAC2IF	xxxx,x0x0
DMA_DAC2_AMTH	DAC2_DMA 传输总字节数	7EF634H	DAC2_AMT[15:8]								0000,0000
DMA_DAC2_AMT	DAC2_DMA 传输总字节数	7EF635H	DAC2_AMT[7:0]								0000,0000
DMA_DAC2_DONEH	DAC2_DMA 传输完成字节数	7EF636H	DAC2_DONE[15:8]								0000,0000
DMA_DAC2_DONE	DAC2_DMA 传输完成字节数	7EF637H	DAC2_DONE[7:0]								0000,0000
DMA_DAC2_TXAH	DAC2_DMA 发送高地址	7EF638H	DAC2_TXA[15:8]								0000,0000
DMA_DAC2_TXAL	DAC2_DMA 发送低地址	7EF639H	DAC2_TXA[7:0]								0000,0000
DMA_DAC2_ITVH	DAC2_DMA 时间间隔寄存器	7EF63EH	DAC2_ITV[15:8]								0000,0000
DMA_DAC2_ITVL	DAC2_DMA 时间间隔寄存器	7EF63FH	DAC2_ITV[7:0]								0000,0000

33.14.2 DACn_DMA 配置寄存器（DMA_DACn_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_DAC1_CFG	7EF618H	DAC1IE	-			DAC1IP[1:0]		DAC1PTY[1:0]	
DMA_DAC2_CFG	7EF630H	DAC2IE	-			DAC2IP[1:0]		DAC2PTY[1:0]	

DACnIE：DACn_DMA 中断使能控制位（n=1/2）

0：禁止 DACn_DMA 中断

1：允许 DACn_DMA 中断

DACnIP[1:0]：DACn_DMA 中断优先级控制位（n=1/2）

DACnIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

DACnPTY[1:0]：DACn_DMA 数据总线访问优先级控制位（n=1/2）

DACnPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

33.14.3 DACn_DMA 控制寄存器（DMA_DACn_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_DAC1_CR	7EF619H	ENDAC1	TRIG1	-	-	-	-	-	-
DMA_DAC2_CR	7EF631H	ENDAC2	TRIG2	-	-	-	-	-	-

ENDACn：DACn_DMA 功能使能控制位（n=1/2）

0：禁止 DACn_DMA 功能

1：允许 DACn_DMA 功能

TRIGn：DACn_DMA 操作触发控制位（n=1/2）

0：写 0 无效

1：写 1 开始 DACn_DMA 操作，

33.14.4 DACn_DMA 状态寄存器（DMA_DACn_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_DAC1_STA	7EF61AH	-	-	-	-	-	-	-	DAC1IF
DMA_DAC2_STA	7EF632H	-	-	-	-	-	-	-	DAC2IF

DACnIF：DACn_DMA 中断请求标志位，当 DACn_DMA 完成扫描所有使能的 DACn 通道后后，硬件自动将 DACnIF 置 1，若使能 DACn_DMA 中断则进入中断服务程序。标志位需软件清零（n=1/2）

33.14.5 DACn_DMA 循环扫描总次数寄存器（DMA_DACn_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_DAC1_AMTH	7EF61CH	DAC1_AMT[15:8]							
DMA_DAC1_AMT	7EF61DH	DAC1_AMT[7:0]							
DMA_DAC2_AMTH	7EF634H	DAC2_AMT[15:8]							
DMA_DAC2_AMT	7EF635H	DAC2_AMT[7:0]							

DACn_AMT[15:0]：设置需要进行数据读写的字节数。（n=1/2）

33.14.6 DACn_DMA 完成扫描次数寄存器（DMA_DACn_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_DAC1_DONEH	7EF61EH	DAC1_DONE[15:8]							
DMA_DAC1_DONE	7EF61FH	DAC1_DONE[7:0]							
DMA_DAC2_DONEH	7EF636H	DAC2_DONE[15:8]							
DMA_DAC2_DONE	7EF637H	DAC2_DONE[7:0]							

DACn_DONE[15:0]：当前已经完成 DAC 扫描的次数。（n=1/2）

33.14.7 DACn_DMA 发送地址寄存器（DMA_DACn_TXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_DAC1_TXAH	7EF620H	DAC1_TXA[15:8]							
DMA_DAC1_TXAL	7EF621H	DAC1_TXA[7:0]							
DMA_DAC2_TXAH	7EF638H	DAC2_TXA[15:8]							
DMA_DAC2_TXAL	7EF639H	DAC2_TXA[7:0]							

DACn_TXA[15:00]：设置进行 DACn_DMA 操作时 DAC 转换数据的存储地址。（n=1/2）

33.14.8 DACn_DMA 转换周期（DMA_DACn_ITVx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_DAC1_ITVH	7EF626H	DAC1_ITV[15:8]							
DMA_DAC1_ITVL	7EF627H	DAC1_ITV[7:0]							
DMA_DAC2_ITVH	7EF63EH	DAC2_ITV[15:8]							
DMA_DAC2_ITVL	7EF63FH	DAC2_ITV[7:0]							

DACn_ITV[15:0]：设置 DACn 转换的间隔时间（基于系统时钟）。（n=1/2）

33.15 CAN-FD1/2 与存储器之间的数据交换（CANnT_DMA，CANnR_DMA）

33.15.1 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_CAN1T_CFG	CAN1T_DMA 配置寄存器	7EF520H	CAN1TIE	-	TADRALIGN	TX_BW	CAN1TIP[1:0]		CAN1TPTY[1:0]		0xxx,0000
DMA_CAN1T_CR	CAN1T_DMA 控制寄存器	7EF521H	ENCAN1T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_CAN1T_STA	CAN1T_DMA 状态寄存器	7EF522H	-	-	-	-	-	TXOVW	-	CAN1TIF	xxxx,x0x0
DMA_CAN1T_AMTH	CAN1T_DMA 传输总字节数	7EF524H	CAN1T_AMT[15:8]								0000,0000
DMA_CAN1T_AMT	CAN1T_DMA 传输总字节数	7EF525H	CAN1T_AMT[7:0]								0000,0000
DMA_CAN1T_DONEH	CAN1T_DMA 传输完成字节数	7EF526H	CAN1T_DONE[15:8]								0000,0000
DMA_CAN1T_DONE	CAN1T_DMA 传输完成字节数	7EF527H	CAN1T_DONE[7:0]								0000,0000
DMA_CAN1T_TXAH	CAN1T_DMA 发送高地址	7EF528H	CAN1T_TXA[15:8]								0000,0000
DMA_CAN1T_TXAL	CAN1T_DMA 发送低地址	7EF529H	CAN1T_TXA[7:0]								0000,0000
DMA_CAN1T_FRM	CAN1T_DMA 发送帧寄存器	7EF52BH	CAN1T_FRM[7:0]								0000,0000
DMA_CAN1T_FCNTH	CAN1T_DMA 发送帧数寄存器	7EF52CH	CAN1T_FCNT[15:8]								0000,0000
DMA_CAN1T_FCNTL	CAN1T_DMA 发送帧数寄存器	7EF52DH	CAN1T_FCNT[7:0]								0000,0000
DMA_CAN1_ITVH	CAN1_DMA 时间间隔寄存器	7EF52EH	CAN1_ITV[15:8]								0000,0000
DMA_CAN1_ITVL	CAN1_DMA 时间间隔寄存器	7EF52FH	CAN1_ITV[7:0]								0000,0000
DMA_CAN1R_CFG	CAN1R_DMA 配置寄存器	7EF530H	CAN1RIE	RAMTALIGN	RADRALIGN	RX_BW	CAN1RIP[1:0]		CAN1RPTY[1:0]		0xxx,0000
DMA_CAN1R_CR	CAN1R_DMA 控制寄存器	7EF531H	ENCAN1R	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_CAN1R_STA	CAN1R_DMA 状态寄存器	7EF532H	-	-	-	-	-	-	RXLOSS	CAN1RIF	xxxx,xx00
CAN1_AUX_CR	CAN1 辅助控制寄存器	7EF533H	-	STBYEN	TRANMANU	DMATXEN	-	-	-	DMARXEN	x000,xxx0
DMA_CAN1R_AMTH	CAN1R_DMA 传输总字节数	7EF534H	CAN1R_AMT[15:8]								0000,0000
DMA_CAN1R_AMT	CAN1R_DMA 传输总字节数	7EF535H	CAN1R_AMT[7:0]								0000,0000
DMA_CAN1R_DONEH	CAN1R_DMA 传输完成字节数	7EF536H	CAN1R_DONE[15:8]								0000,0000
DMA_CAN1R_DONE	CAN1R_DMA 传输完成字节数	7EF537H	CAN1R_DONE[7:0]								0000,0000
DMA_CAN1R_RXAH	CAN1R_DMA 接收高地址	7EF538H	CAN1R_RXA[15:8]								0000,0000
DMA_CAN1R_RXAL	CAN1R_DMA 接收低地址	7EF539H	CAN1R_RXA[7:0]								0000,0000
DMA_CAN1R_FRM	CAN1T_DMA 接收帧寄存器	7EF53BH	CAN1R_FRM[7:0]								0000,0000
DMA_CAN1R_FCNTH	CAN1T_DMA 接收帧数寄存器	7EF53CH	CAN1R_FCNT[15:8]								0000,0000
DMA_CAN1R_FCNTL	CAN1T_DMA 接收帧数寄存器	7EF53DH	CAN1R_FCNT[7:0]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_CAN2T_CFG	CAN2T_DMA 配置寄存器	7EF540H	CAN2TIE	-	TADRALIGN	TX_BW	CAN2TIP[1:0]		CAN2TPTY[1:0]		0xxx,0000
DMA_CAN2T_CR	CAN2T_DMA 控制寄存器	7EF541H	ENCAN2T	TRIG	-	-	-	-	-	-	00xx,xxxx
DMA_CAN2T_STA	CAN2T_DMA 状态寄存器	7EF542H	-	-	-	-	-	TXOVW	-	CAN2TIF	xxxx,x0x0
DMA_CAN2T_AMTH	CAN2T_DMA 传输总字节数	7EF544H	CAN2T_AMT[15:8]								0000,0000
DMA_CAN2T_AMT	CAN2T_DMA 传输总字节数	7EF545H	CAN2T_AMT[7:0]								0000,0000
DMA_CAN2T_DONEH	CAN2T_DMA 传输完成字节数	7EF546H	CAN2T_DONE[15:8]								0000,0000

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_CAN2T_DONE	CAN2T_DMA 传输完成字节数	7EF547H	CAN2T_DONE[7:0]								0000,0000
DMA_CAN2T_TXAH	CAN2T_DMA 发送高地址	7EF548H	CAN2T_TXA[15:8]								0000,0000
DMA_CAN2T_TXAL	CAN2T_DMA 发送低地址	7EF549H	CAN2T_TXA[7:0]								0000,0000
DMA_CAN2T_FRM	CAN2T_DMA 发送帧寄存器	7EF54BH	CAN2T_FRM[7:0]								0000,0000
DMA_CAN2T_FCNTH	CAN2T_DMA 发送帧数寄存器	7EF54CH	CAN2T_FCNT[15:8]								0000,0000
DMA_CAN2T_FCNTL	CAN2T_DMA 发送帧数寄存器	7EF54DH	CAN2T_FCNT[7:0]								0000,0000
DMA_CAN2_ITVH	CAN2_DMA 时间间隔寄存器	7EF54EH	CAN2_ITV[15:8]								0000,0000
DMA_CAN2_ITVL	CAN2_DMA 时间间隔寄存器	7EF54FH	CAN2_ITV[7:0]								0000,0000
DMA_CAN2R_CFG	CAN2R_DMA 配置寄存器	7EF550H	CAN2RIE	RAMTALIGN	RADRALIGN	RX_BW	CAN2RIP[1:0]		CAN2RPTY[1:0]		0xxx,0000
DMA_CAN2R_CR	CAN2R_DMA 控制寄存器	7EF551H	ENCAN2R	TRIG	-	-	-	-	-	CLRFIFO	00xx,xxx0
DMA_CAN2R_STA	CAN2R_DMA 状态寄存器	7EF552H	-	-	-	-	-	-	RXLOSS	CAN2RIF	xxxx,xx00
CAN2_AUX_CR	CAN2 辅助控制寄存器	7EF553H	-	STBYEN	TRANMANU	DMATXEN	-	-	-	DMARXEN	x000,xxx0
DMA_CAN2R_AMTH	CAN2R_DMA 传输总字节数	7EF554H	CAN2R_AMT[15:8]								0000,0000
DMA_CAN2R_AMT	CAN2R_DMA 传输总字节数	7EF555H	CAN2R_AMT[7:0]								0000,0000
DMA_CAN2R_DONEH	CAN2R_DMA 传输完成字节数	7EF556H	CAN2R_DONE[15:8]								0000,0000
DMA_CAN2R_DONE	CAN2R_DMA 传输完成字节数	7EF557H	CAN2R_DONE[7:0]								0000,0000
DMA_CAN2R_RXAH	CAN2R_DMA 接收高地址	7EF558H	CAN2R_RXA[15:8]								0000,0000
DMA_CAN2R_RXAL	CAN2R_DMA 接收低地址	7EF559H	CAN2R_RXA[7:0]								0000,0000
DMA_CAN2R_FRM	CAN2T_DMA 接收帧寄存器	7EF55BH	CAN2R_FRM[7:0]								0000,0000
DMA_CAN2R_FCNTH	CAN2T_DMA 接收帧数寄存器	7EF55CH	CAN2R_FCNT[15:8]								0000,0000
DMA_CAN2R_FCNTL	CAN2T_DMA 接收帧数寄存器	7EF55DH	CAN2R_FCNT[7:0]								0000,0000

33.15.2 CANnT_DMA 配置寄存器（DMA_CANnT_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1T_CFG	7EF520H	CAN1TIE	-	TADRALIGN	TX_BW	CAN1TIP[1:0]		CAN1TPTY[1:0]	
DMA_CAN2T_CFG	7EF540H	CAN2TIE	-	TADRALIGN	TX_BW	CAN2TIP[1:0]		CAN2TPTY[1:0]	

CANnTIE：CANnT_DMA 中断使能控制位（n=1/2）

0：禁止 CANnT_DMA 中断

1：允许 CANnT_DMA 中断

CANnTIP[1:0]：CANnT_DMA 中断优先级控制位（n=1/2）

CANnTIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

CANnTPTY[1:0]：CANnT_DMA 数据总线访问优先级控制位（n=1/2）

CANnTPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

TADRALIGN：CANn 发送 DMA 数据对齐控制（n=1/2）

0：数据不对齐。DMA 缓冲区中的任意两帧数据之间没有空白，数据连续

1：数据对齐。DMA 缓冲区中的数据必须按 DMA_CANnT_FRM 寄存器设置值进行对齐。
例如：DMA_CANT_FRM 设置的值位 32，则 DMA 缓冲区中的每帧数据必须 32 字节对齐，即使某帧数据不足 32 字节，也必须填充 0，以达到 32 字节的长度。

TX_BW：CANn 发送 DMA 数据宽度（n=1/2）

0：8 位宽度（1 字节）

1：32 位宽度（4 字节）

DMA_CAN1T_CFG.5(TXALIGN)=0 时，缓冲区内每帧数据顺序排列；
DMA_CAN1T_CFG.5(TXALIGN)=1 时，缓冲区内每帧数据按照 DMA_CAN1T_FRM 指定长度排列，
例如：DMA_CAN1T_FRM=32，那么第一帧数据存放在缓冲区 BYTE[0]~BYTE[31]，第二帧数据存放在缓冲区 BYTE[32]~BYTE[63]...

DMA_CAN1T_CFG.4(TX_BW)=0 时，数据宽度 8 位，DMA 传输总数(AMT)按照 8 位计算，缓冲区内每帧数据顺序排列；
DMA_CAN1T_CFG.4(TX_BW)=1 时，数据宽度 32 位，DMA 传输总数(AMT)按照 32 位计算，缓冲区内每帧数据起始地址需要 32 位对齐，
例如：DMA 传输 4 帧数据，每帧数据长度分别是 16 字节、15 字节、14 字节、13 字节，
数据宽度 8 位时，(16+15+14+13)-1=57，DMA_CAN1T_AMT = 57;
数据宽度 32 位时，每帧数据长度占用 4 个 WORD，(4+4+4+4)-1=15，DMA_CAN1T_AMT = 15;

33.15.3 CANnT_DMA 控制寄存器（DMA_CANnT_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1T_CR	7EF521H	ENCAN1T	TRIG1	-	-	-	-	-	-
DMA_CAN2T_CR	7EF541H	ENCAN2T	TRIG2	-	-	-	-	-	-

ENCANnT：CANnT_DMA 功能使能控制位（n=1/2）

0：禁止 CANnT_DMA 功能

1：允许 CANnT_DMA 功能

TRIGn：CANnT_DMA 发送触发控制位（n=1/2）

0：写 0 无效

1：写 1 开始 CANnT_DMA 自动发送数据

33.15.4 CANnT_DMA 状态寄存器（DMA_CANnT_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1T_STA	7EF522H	-	-	-	-	-	TXOVW1	-	CAN1TIF
DMA_CAN2T_STA	7EF542H	-	-	-	-	-	TXOVW2	-	CAN2TIF

CANnTIF：CANnT_DMA 中断请求标志位，当 CANnT_DMA 数据发送完成后，硬件自动将 CANnTIF 置 1，若使能 CANnT_DMA 中断则进入中断服务程序。标志位需软件清零（n=1/2）

TXOVWn：CANnT_DMA 数据覆盖标志位。CANnT_DMA 正在数据传输过程中，写 CANn 数据寄存器时，会导致数据传输失败，此时硬件硬件自动将 TXOVW2 置 1。标志位需软件清零（n=1/2）

33.15.5 CANnT_DMA 传输总字节寄存器（DMA_CANnT_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1T_AMTH	7EF524H	CAN1T_AMT[15:8]							
DMA_CAN1T_AMT	7EF525H	CAN1T_AMT[7:0]							
DMA_CAN2T_AMTH	7EF544H	CAN2T_AMT[15:8]							
DMA_CAN2T_AMT	7EF545H	CAN2T_AMT[7:0]							

CANnT_AMT[15:0]：设置需要进行数据读写的字节数。（n=1/2）

注：实际读写的字节数为（CANnT_AMT+1），即当 CANnT_AMT 设置为 0 时，读写 1 字节，当 CANnT_AMT 设置 255 时，读写 256 字节。特别注意 CANnT_AMT 设置为 FFFFH 时表示无限循环模式。

33.15.6 CANnT_DMA 传输完成字节寄存器（DMA_CANnT_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1T_DONEH	7EF526H	CAN1T_DONE[15:8]							
DMA_CAN1T_DONE	7EF527H	CAN1T_DONE[7:0]							
DMA_CAN2T_DONEH	7EF546H	CAN2T_DONE[15:8]							
DMA_CAN2T_DONE	7EF547H	CAN2T_DONE[7:0]							

CANnT_DONE[15:0]：当前已经发送完成的字节数。（n=1/2）

33.15.7 CANnT_DMA 发送地址寄存器（DMA_CANnT_TXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1T_TXAH	7EF528H	CAN1T_TXA [15:8]							
DMA_CAN1T_TXAL	7EF529H	CAN1T_TXA [7:0]							
DMA_CAN2T_TXAH	7EF548H	CAN2T_TXA [15:8]							
DMA_CAN2T_TXAL	7EF549H	CAN2T_TXA [7:0]							

CANnT_TXA[15:0]：设置自动发送数据的源地址。执行 CANnT_DMA 操作时会从这个地址开始读数据。（n=1/2）

33.15.8 CANnT_DMA 发送帧大小寄存器（DMA_CANnT_FRM）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1T_FRM	7EF52BH	CAN1T_FRM[7:0]							
DMA_CAN2T_FRM	7EF54BH	CAN2T_FRM[7:0]							

CANnT_FRM[7:0]：设置 CANnT_DMA 发送缓冲区中数据帧对齐字节大小。本寄存器设置值只有在 DMA_CANnT_CFG.TADRALIGN 为 1 时才有效。（n=1/2）

33.15.9 CANnT_DMA 发送帧数寄存器（DMA_CANnT_FCNTx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1T_FCNTH	7EF52CH	CAN1T_FCNT[15:8]							
DMA_CAN1T_FCNTL	7EF52DH	CAN1T_FCNT[7:0]							
DMA_CAN2T_FCNTH	7EF54CH	CAN2T_FCNT[15:8]							
DMA_CAN2T_FCNTL	7EF54DH	CAN2T_FCNT[7:0]							

CANnT_FCNT[15:0]：CANnT_DMA 发送帧的数量（只读）（n=1/2）

33.15.10 CANn_DMA 传输周期（DMA_CANn_ITVx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1_ITVH	7EF52EH	CAN1_ITV[15:8]							
DMA_CAN1_ITVL	7EF52FH	CAN1_ITV[7:0]							
DMA_CAN2_ITVH	7EF54EH	CAN2_ITV[15:8]							
DMA_CAN2_ITVL	7EF54FH	CAN2_ITV[7:0]							

DMA_CANn_ITV[15:0]：设置 CANn 发送和接收的间隔时间（基于系统时钟）。（n=1/2）

33.15.11 CANnR_DMA 配置寄存器（DMA_CANnR_CFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1R_CFG	7EF530H	CAN1RIE	RAMTALIGN	RADRALIGN	RX_BW	CAN1RIP[1:0]		CAN1RPTY[1:0]	
DMA_CAN2R_CFG	7EF550H	CAN2RIE	RAMTALIGN	RADRALIGN	RX_BW	CAN2RIP[1:0]		CAN2RPTY[1:0]	

CANnRIE：CANnR_DMA 中断使能控制位（n=1/2）

0：禁止 CANnR_DMA 中断

1：允许 CANnR_DMA 中断

CANnRIP[1:0]：CANnR_DMA 中断优先级控制位（n=1/2）

CANnRIP[1:0]	中断优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

CANnRPTY[1:0]：CANnR_DMA 数据总线访问优先级控制位（n=1/2）

CANnRPTY [1:0]	总线优先级
00	最低级（0）
01	较低级（1）
10	较高级（2）
11	最高级（3）

RAMTALIGN：CANn 接收 DMA 总字节对齐控制（n=1/2）

0：当接收的数据总字节等于 DMA_CANnR_AMT 所设置的值时，DMA 接收完成

1：当接收的数据总字节大于或等于 DMA_CANnR_AMT 所设置的值，且当前帧的所有数据都已传输完成时，DMA 接收完成。

RADRALIGN：CANn 接收 DMA 数据对齐控制（n=1/2）

0：数据不对齐。DMA 缓冲区中的任意两帧数据之间没有空白，数据连续

1：数据对齐。DMA 缓冲区中的数据必须按 DMA_CANnR_FRM 寄存器设置值进行对齐。
例如：DMA_CANR_FRM 设置的值位 32，则 DMA 缓冲区中的每帧数据必须 32 字节对齐，即使某帧数据不足 32 字节，也必须填充 0，以达到 32 字节的长度。

RX_BW：CANn 接收 DMA 数据宽度（n=1/2）

0：8 位宽度（1 字节）

1：32 位宽度（4 字节）

33.15.12 CANnR_DMA 控制寄存器（DMA_CANnR_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1R_CR	7EF531H	ENCAN1R	TRIG1	-	-	-	-	-	CLRFIFO1
DMA_CAN2R_CR	7EF551H	ENCAN2R	TRIG2	-	-	-	-	-	CLRFIFO2

ENCANnR：CANnR_DMA 功能使能控制位（n=1/2）

0：禁止 CANnR_DMA 功能

1：允许 CANnR_DMA 功能

TRIGn：CANnR_DMA 接收触发控制位（n=1/2）

0：写 0 无效

1：写 1 开始 CANnR_DMA 自动接收数据

CLRFIFOn：清除 CANnR_DMA 接收 FIFO 控制位（n=1/2）

0：写 0 无效

1：开始 CANnR_DMA 操作前，先清空 CANnR_DMA 内置的 FIFO

33.15.13 CANnR_DMA 状态寄存器（DMA_CANnR_STA）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1R_STA	7EF532H	-	-	-	-	-	-	RXLOSS	CAN1RIF
DMA_CAN2R_STA	7EF552H	-	-	-	-	-	-	RXLOSS2	CANnRIF

CANnRIF：CANnR_DMA 中断请求标志位，当 CANnR_DMA 接收数据完成后，硬件自动将 CANnRIF 置 1，若使能 CANnR_DMA 中断则进入中断服务程序。标志位需软件清零（n=1/2）

RXLOSSn：CANnR_DMA 接收数据丢弃标志位。CANnR_DMA 操作过程中，当 XRAM 总线过于繁忙，来不及清空 CANnR_DMA 的接收 FIFO 导致 CANnR_DMA 接收的数据自动丢弃时，硬件硬件自动将 RXLOSSn 置 1。标志位需软件清零（n=1/2）

33.15.14 CANn_DMA 辅助控制寄存器（CANn_AUX_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN1_AUX_CR	7EF533H	-	STBYEN	TRANMANU	DMATXEN	-	-	-	DMARXEN
CAN2_AUX_CR	7EF553H	-	STBYEN	TRANMANU	DMATXEN	-	-	-	DMARXEN

STBYEN：CANn 模块的低功耗控制脚（CANn_STB）使能控制位（n=1/2）

0：禁止低功耗控制脚功能

1：允许低功耗控制脚功能

【说明】：允许低功耗控制脚功能后，CAN_TCMD.5 的 STBY=1 时 CANSTB 引脚输出高电平，STBY=0 时 CANSTB 引脚输出低电平。

TRANMANU：CANn 模块的 TPE 和 TSALL 控制位自动设置控制（n=1/2）

0：当一帧数据被 DMA 控制器写入到 CANn 模块的 TBUF 后，硬件自动设置 TPE 和 TSALL

1：硬件不自动设置 TPE 和 TSALL

DMATXEN：CANn 模块发送 DMA 使能控制（n=1/2）

0：禁止 CANn 模块发送 DMA 功能

1：使能 CANn 模块发送 DMA 功能

DMARXEN：CANn 模块接收 DMA 使能控制（n=1/2）

0：禁止 CANn 模块接收 DMA 功能

1：使能 CANn 模块接收 DMA 功能

33.15.15 CANnR_DMA 传输总字节寄存器（DMA_CANnR_AMT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1R_AMTH	7EF534H	CAN1R_AMT[15:8]							
DMA_CAN1R_AMT	7EF535H	CAN1R_AMT[7:0]							
DMA_CAN2R_AMTH	7EF554H	CAN2R_AMT[15:8]							
DMA_CAN2R_AMT	7EF555H	CAN2R_AMT[7:0]							

CANnR_AMT[15:0]：设置需要进行数据读写的字节数。（n=1/2）

注：实际读写的字节数为（CANnR_AMT+1），即当 CANnR_AMT 设置为 0 时，读写 1 字节，当 CANnR_AMT 设置 255 时，读写 256 字节。特别注意 CANnR_AMT 设置为 FFFFH 时表示无限循环模式。

33.15.16 CANnR_DMA 传输完成字节寄存器（DMA_CANnR_DONE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1R_DONEH	7EF536H	CAN1R_DONE[15:8]							
DMA_CAN1R_DONE	7EF537H	CAN1R_DONE[7:0]							
DMA_CAN2R_DONEH	7EF556H	CAN2R_DONE[15:8]							
DMA_CAN2R_DONE	7EF557H	CAN2R_DONE[7:0]							

CANnR_DONE[15:0]：当前已经接收完成的字节数。（n=1/2）

33.15.17 CANnR_DMA 接收地址寄存器（DMA_CANnR_RXAx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1R_RXAH	7EF538H	CAN1R_RXA[15:8]							
DMA_CAN1R_RXAL	7EF539H	CAN1R_RXA[7:0]							
DMA_CAN2R_RXAH	7EF558H	CAN2R_RXA[15:8]							
DMA_CAN2R_RXAL	7EF559H	CAN2R_RXA[7:0]							

CANnR_RXA[15:0]：设置自动接收数据的目标地址。执行 CANnR_DMA 操作时会从这个地址开始写数据。（n=1/2）

33.15.18 CANnR_DMA 接收帧大小寄存器（DMA_CANnR_FRM）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1R_FRM	7EF53BH	CAN1R_FRM[7:0]							
DMA_CAN2R_FRM	7EF55BH	CAN2R_FRM[7:0]							

CANnR_FRM[7:0]：设置 CANnR_DMA 接收缓冲区中数据帧对齐字节大小。本寄存器设置值只有在 DMA_CANnR_CFG.TADRALIGN 为 1 时才有效。（n=1/2）

33.15.19 CANnR_DMA 接收帧数寄存器（DMA_CANnR_FCNTx）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_CAN1R_FCNTH	7EF53CH	CAN1R_FCNT[15:8]							
DMA_CAN1R_FCNTL	7EF53DH	CAN1R_FCNT[7:0]							
DMA_CAN2R_FCNTH	7EF55CH	CAN2R_FCNT[15:8]							
DMA_CAN2R_FCNTL	7EF55DH	CAN2R_FCNT[7:0]							

CANnR_FCNT[15:0]：CANnR_DMA 接收的帧的数量。（只读）（n=1/2）

33.16 外设与外设之间的数据交换（P2P_DMA）

33.16.1 P2P 使用注意事项

目前测试发现 STC32G144K246 的 QSPI 到其他外设的 P2P 方式 DMA 会影响 XRAM 的数据，在使用 QSPI 到其他外设的 P2P 功能时，**必须将 QSPI 存储接收数据的 XRAM 起始地址设置到 XRAM 的后面部分。注意：由于 STC32G144K246 内部的 XRAM 有 128K+4K，当仅需要使用 QSPI 的 P2P 功能且不需要使用其他 DMA，可将 QSPI 的目标地址设置为额外的 4K 范围，则在进行 QSPI 的 P2P 功能时就不会影响其他 128K 的 XRAM。**

【例如】当使用 QSPI 到 i8080 彩屏接口的 P2P 功能时，可将{DMA_QSPI_RXAH, DMA_QSPI_RXAL}中的地址设置到 **0K** 开始的地方，并将 DMA_ARB_CFG 寄存器中的 XBANK 设置为 **10B**，则 QSPI 进行 P2P 时就只会影响额外的 4K 字节 XRAM，而不会影响前面的 128K 字节的 XRAM。

另外，寄存器{DMA_QSPI_RXAH, DMA_QSPI_RXAL}中设置的是每次 DMA 读取 XRAM 的起始地址，在 DMA 过程中{DMA_QSPI_RXAH, DMA_QSPI_RXAL}的值不会改变，所以只需要在初始化时设置好{DMA_QSPI_RXAH, DMA_QSPI_RXAL}中的初始值就可以了，不需要每次 DMA 时都设置。

33.16.2 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMA_P2P1_DEST	P2P1 DMA 目标外设选择	7EFAF0H	P2P1_DEST[7:0]								0000,0000
DMA_P2P1_SRC	P2P1 DMA 源外设选择	7EFAF1H	P2P1_SRC[7:0]								0000,0000
DMA_P2P2_DEST	P2P2 DMA 目标外设选择	7EFAF2H	P2P2_DEST[7:0]								0000,0000
DMA_P2P2_SRC	P2P2 DMA 源外设选择	7EFAF3H	P2P2_SRC[7:0]								0000,0000

33.16.3 P2Pn_DMA 目标外设选择寄存器（DMA_P2Pn_DEST）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_P2P1_DEST	7EFAF0H	P2P1_DEST[7:0]							
DMA_P2P2_DEST	7EFAF2H	P2P2_DEST[7:0]							

P2Pn_DEST[7:0]：选择 P2Pn_DMA 的目标外设（发送外设）（n=1/2）

P2Pn_DEST[3:0]	目标外设
0x00	无外设
0x01	串口 1（发送）
0x02	串口 2（发送）
0x03	串口 3（发送）
0x04	串口 4（发送）
0x05	SPI1（发送）
0x06	I2C1（发送）
0x07	LCM（发送）
0x08	QSPI（发送）
0x09	-
0x0A	I2S1（发送）
0x0B	串口 5（发送）
0x0C	串口 6（发送）
0x0D	串口 7（发送）
0x0E	串口 8（发送）
0x0F	SPI2（发送）
0x10	SPI3（发送）
0x11	-
0x12	I2C2（发送）
0x13	I2S2（发送）
0x14	-
0x15	-
0x16	CAN-FD1（发送）
0x17	CAN-FD2（发送）
0x18	-

33.16.4 P2Pn_DMA 源外设选择寄存器（DMA_P2Pn_SRC）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DMA_P2P1_SRC	7EFAF1H	P2P1_SRC[7:0]							
DMA_P2P2_SRC	7EFAF3H	P2P2_SRC[7:0]							

P2Pn_SRC[7:0]：选择 P2Pn_DMA 的源外设（接收外设）（n=1/2）

P2Pn_SRC[3:0]	源外设
0x00	无外设
0x01	串口 1（接收）
0x02	串口 2（接收）
0x03	串口 3（接收）
0x04	串口 4（接收）
0x05	SPI1（接收）
0x06	I2C1（接收）
0x07	LCM（接收）
0x08	QSPI（接收）
0x09	ADC1（接收）
0x0A	I2S1（接收）
0x0B	串口 5（接收）
0x0C	串口 6（接收）
0x0D	串口 7（接收）
0x0E	串口 8（接收）
0x0F	SPI2（接收）
0x10	SPI3（接收）
0x11	-
0x12	I2C2（接收）
0x13	I2S2（接收）
0x14	-
0x15	-
0x16	CAN-FD1（接收）
0x17	CAN-FD2（接收）
0x18	ADC2（接收）

34 两组 I2S 音频总线，可以使用 PLL 高速时钟作为时钟源

产品线	I2S
STC32G144K246 系列	**两组**

STC32G144K246 系列单片机内部集成两组 I2S 音频总线功能单元。

实际应用中，如果两组 I2S 在同一时间分别进行放音（主机模式）和录音（从机模式）时，主机模式的 I2S，MCLK/BCK/LRCK/Data 四根数据线都需要；作为从机模式的 I2S 只需要 Data 数据线就可以了，从机模式的 I2S 的 MCLK/BCK/LRCK 这 3 根线可用作 GPIO 或作其他功能使用。

34.1 I2S 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW3	BBH	**I2S_S[1:0]**		S2SPI_S[1:0]		S1SPI_S[1:0]		CAN2_S[1:0]	

I2S_S[1:0]：I2S 功能脚选择位

I2S_S[1:0]	I2SMCLK	I2SBCK	I2SLRCK	I2SData
	主时钟	位时钟	左右声道时钟	音频数据
00	P3.3	P3.2	P3.5	P3.4
01	P6.7	P6.6	P6.5	P6.4
10	P2.4	P2.5	P2.2	P2.3
11	P1.6	P4.3	P4.1	P4.0

I2SMCLK， I2S 主时钟
I2SBCK， I2S 位时钟
I2SLRCK， I2S 左右声道时钟
I2SData， I2S 音频数据

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX5	7EFD6DH	S8_S[1:0]		S7_S[1:0]		**I2S2_S[1:0]**		-	-

I2S2_S[1:0]：I2S2 功能脚选择位

I2S2_S[1:0]	I2S2MCLK	I2S2BCK	I2S2LRCK	I2S2Data
	主时钟	位时钟	左右声道时钟	音频数据
00	P5.2	P5.3	P5.0	P5.1
01	P6.2	P6.3	P6.0	P6.1
10	P8.2	P8.3	P8.0	P8.1
11	PB.3	PB.4	PB.1	PB.2

I2S2MCLK， I2S2 主时钟
I2S2BCK， I2S2 位时钟
I2S2LRCK， I2S2 左右声道时钟
I2S2Data， I2S2 音频数据

34.2 寄存器描述

34.2.1 I2S 控制寄存器（I2SnCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2SCR	7EFD98H	I2STXEIE	I2SRXNEIE	I2SERRIE	I2SFRF	-	-	I2STXDMAEN	I2SRXDMAEN
I2S2CR	7EFD90H	I2S2TXEIE	I2S2RXNEIE	I2S2ERRIE	I2S2FRF	-	-	I2S2TXDMAEN	I2S2RXDMAEN

I2SnTXEIE：发送缓冲区空中断允许位。（n=1/2）

0：禁止发送缓冲区空中断

1：允许发送缓冲区空中断

I2SnRXNEIE：接收缓冲区非空中断允许位。（n=1/2）

0：禁止接收缓冲区非空中断

1：允许接收缓冲区非空中断

I2SnERRIE：错误中断允许位。（n=1/2）

0：禁止错误中断

1：允许错误中断

I2SnFRF：帧格式（n=1/2）

0：Motorola 格式

1：TI 格式

I2SnTXDMAEN：发送DMA控制（n=1/2）

0：禁止发送 DMA

1：使能发送 DMA

I2SnRXDMAEN：接收DMA控制（n=1/2）

0：禁止接收 DMA

1：使能接收 DMA

34.2.2 I2S 状态寄存器（I2SnSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2SSR	7EFD99H	-	I2SFRE	I2SBUY	I2SOVR	I2SUDR	I2SCHSID	I2STXE	I2SRXNE
I2S2SR	7EFD91H	-	I2S2FRE	I2S2BUY	I2S2OVR	I2S2UDR	I2S2CHSID	I2S2TXE	I2S2RXNE

I2SnFRE：帧格式错误（n=1/2）

I2SnBUY：I2Sn忙状态。（n=1/2）（正在通讯中或者发送缓冲区非空）

I2SnOVR：发生上溢（n=1/2）

I2SnUDR：发生下溢（n=1/2）

I2SnCHSID：通道选择标志（n=1/2）

0：左通道正在发送或接收数据

1：右通道正在发送或接收数据

I2SnTXE：发送缓冲区空标志位（n=1/2）

I2SnRXNE：接收缓冲区非空标志位（n=1/2）

34.2.3 I2S 数据寄存器（I2SnDRH、I2SnDRL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2SDRH	7EFD9AH	I2SDR[15:8]							
I2SDRL	7EFD9BH	I2SDR[7:0]							
I2S2DRH	7EFD92H	I2S2DR[15:8]							
I2S2DRL	7EFD93H	I2S2DR[7:0]							

I2SnDR[15:0]：接收或发送的数据。写数据寄存器时应先写高字节I2SDRH，后写I2SDRL。读数据寄存器时应先读高字节I2SDRH，后读I2SDRL。（n=1/2）

34.2.4 I2Sn 时钟分频寄存器（I2Sn_CLKDIV）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2S_CLKDIV	7EFE94H	I2S_CLKDIV[7:0]							
I2S2_CLKDIV	7EFE94H	I2S2_CLKDIV[7:0]							

I2Sn_CLKDIV：I2Sn 时钟分频系数。(n=1,2)

I2Sn_CLKDIV	I2Sn 工作时钟频率
0	HSI2SCLK /1
1	HSI2SCLK /1
2	HSI2SCLK /2
…	…
255	HSI2SCLK /255

使用 I2S 一定要启动 PLL，选择 HPLL1/2 或 HPLL2/2。

I2S 时钟 = HPLL1 / 2 / I2S_CLKDIV。

推荐主频 2 个：

1、主频 24.576MHz，PLL 倍频系数为 40，则 HPLL 时钟 = 24.576MHz/4*2*40 = 491.52MHz

2、主频 24MHz，PLL 倍频系数为 41，则 HPLL 时钟 = 24/4 *2*41 = 492MHz，跟 491.52MHz 误差不到 0.1%。

推荐 I2S_CLKDIV=2，这样可以支持采样率为 8KHz ~ 96KHz，后面有推荐的采样率。

此时，I2S 时钟 = HPLL/2/2 = HPLL/4 = 491.52MHz/4 = 122.88MHz。

34.2.5 I2S 位时钟 BCLK 分频寄存器高字节（I2SnPRH）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2SPRH	7EFD9CH	-	-	-	-	-	-	I2SMCKOE	I2SODD
I2S2PRH	7EFD94H	-	-	-	-	-	-	I2S2MCKOE	I2S2ODD

I2SnMCKOE：I2Sn主时钟输出控制（n=1/2）

0：禁止 I2Sn 主时钟输出

1：使能 I2Sn 主时钟输出

I2SnODD：分频器的奇数因子控制（n=1/2）

0：实际分频值＝DIV*2

1：实际分频值＝DIV*2+1

34.2.6 I2S 位时钟 BCLK 分频寄存器低字节（I2SnPRL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2SPRL	7EFD9DH	I2SDIV[7:0]							
I2S2PRL	7EFD95H	I2S2DIV[7:0]							

I2SnDIV[7:0]：I2S的时钟分频器。DIV不可设置为0或者1。（n=1/2）

I2S 比特率＝每个通道的比特数×通道数×音频采样频率(FS)

对于 16 位音频、左声道和右声道，I2S 比特率计算如下：I2S 比特率＝16×2×FS

对于 32 位宽度的数据包长度，I2S 比特率＝32×2×FS。

当信道帧宽度为 16 位时，音频采样频率 FS＝I2S 时钟÷[(16×2)×(2×DIV+ODD)]

当信道帧宽度为 32 位时，音频采样频率 FS＝I2S 时钟÷[(32×2)×(2×DIV+ODD)]

位时钟分频系数 BCLK**DIV[8:0] =** 2×DIV+ODD

对于信道帧宽度为 16 位时，I2S 位时钟 BCLK = 16*2*FS = I2S 时钟 / BCLKDIV，

由此可计算出：BCLKDIV = I2S 时钟 / (32*FS)。

对于信道帧宽度为 32 位时，I2S 位时钟 BCLK = 32*2*FS = I2S 时钟 / BCLKDIV，

由此可计算出：BCLKDIV = I2S 时钟 / (64*FS)。

将 BCLKDIV 最低位 bit0 赋给 I2SPRH 的 bit0[I2SODD]，I2SPRL=BCLKDIV/2。

上述 I2S 时钟必须选择 PLL 时钟

34.2.7 I2S 配置寄存器高字节（I2SnCFGH）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2SCFGH	7EFD9EH	-	-	-	-	-	I2SE	I2SCFG[1:0]	
I2S2CFGH	7EFD96H	-	-	-	-	-	I2S2E	I2S2CFG[1:0]	

I2SnE：I2Sn模块控制（n=1/2）

0：禁止 I2S 功能

1：使能 I2S 功能

I2SnCFG[1:0]：I2Sn模式配置（n=1/2）

00：从机发送模式

01：从机接收模式

10：主机发送模式

11：主机接收模式

34.2.8 I2S 配置寄存器低字节（I2SnCFGL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2SCFGL	7EFD9FH	I2SPCMSYNC	-	I2SSTD[1:0]		I2SCKPOL	I2SDATLEN[1:0]		I2SCHLEN
I2S2CFGL	7EFD97H	I2S2PCMSYNC	-	I2S2STD[1:0]		I2S2CKPOL	I2S2DATLEN[1:0]		I2S2CHLEN

I2SnPCMSYNC：PCM帧同步（n=1/2）

0：短帧同步

1：长帧同步

I2SnSTD[1:0]：I2S标准选择（n=1/2）

00：I2S 飞利浦标准

01：MSB 左对齐标准

10：LSB 右对齐标准

11：PCM 标准

I2SnCKPOL：稳态时钟极性（n=1/2）

0：I2Sn 时钟稳定状态为低电平

1：I2Sn 时钟稳定状态为高电平

I2SnDATLEN[1:0]：数据长度（n=1/2）

00：16 位

01：24 位

10：32 位

11：保留

I2SnCHLEN：通道长度（每个音频通道的位数）（n=1/2）

0：16 位

1：32 位

34.2.9 I2S 主时钟 MCLK 分频寄存器（I2SnMCKDIV）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2SMCKDIV	7EFDA1H								
I2S2MCKDIV	7EFDA5H								

I2SnMCKDIV：I2Sn 主时钟分频系数。（n=1/2）

I2S 主时钟 MCLK = 256*FS = I2S 时钟 / I2SnCLKDIV，

由此可计算出：I2SnCLKDIV = I2S 时钟 / (256*FS)。

当 I2SnCLKDIV 为 0 时，I2SnCLKDIV 仍默认为 1。

34.2.10 I2S 输入信号选择寄存器（I2SnINSEL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
I2SINSEL	7EFDA2H	-	-	-	-	-	-	I2SINSEL[1:0]	
I2S2INSEL	7EFDA6H	-	-	-	-	-	-	I2S2INSEL[1:0]	

I2SINSEL[1:0]：I2S 输入信号选择

I2SINSEL[1:0]	I2S 位时钟	I2S 左右声道时钟
00	I2SBCK 管脚的输入时钟	I2SLRCK 管脚的输入时钟
01	I2S 输出的位时钟	I2S 输出的左右声道时钟
10	I2S2 输出的位时钟	I2S2 输出的左右声道时钟
11	-	-

I2S2INSEL[1:0]：I2S2 输入信号选择

I2S2INSEL[1:0]	I2S2 位时钟	I2S2 左右声道时钟
00	I2S2BCK 管脚的输入时钟	I2S2LRCK 管脚的输入时钟
01	I2S 输出的位时钟	I2S 输出的左右声道时钟
10	I2S2 输出的位时钟	I2S2 输出的左右声道时钟
11	-	-

35 TFPU（三角函数+单精度浮点运算器），可以使用 PLL 高速时钟作为时钟源

产品线	TFPU
STC32G144K246 系列	●

35.1 TFPU 浮点运算器简介

单精度浮点运算器（TFPU）提供了快速的单精度浮点算术运算。TFPU 支持单精度浮点数的加、减、乘、除、开方、比较和三角函数（正弦、余弦、正切和反正切）。同时支持整数类型和单精度浮点数之间的转换。输入的浮点数字格式符合 IEEE-754 标准。TFPU 由专用直接内存访问 DMA 控制。所有算术运算都是通过将运算指令写入称为 DMAIR 控制寄存器来启动的。TFPU 模块执行的所有算术运算的操作数（或其指针）和结果（或其指针）位于当前组的寄存器 R0-R7 中。

35.2 相关的特殊功能寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DMAIR	DMA 指令寄存器	EDH	DMAIR[7:0]								0000,0000

注：向 DMAIR 寄存器写入指令码，只能使用立即数寻址方式的指令“MOV DMAIR,#N”，使用其它指令会无法正常触发计算。

35.3 运算执行时间表

TFPU 运算	DMA 指令码		执行时钟数
	HEX	DEC	
浮点数加法	0x1C	28	31 ~ 40
浮点数减法	0x1D	29	31 ~ 40
浮点数乘法	0x1E	30	26 ~ 34
浮点数除法	0x1F	31	58 ~ 67
浮点数开方	0x20	32	50 ~ 54
浮点数比较	0x21	33	18
浮点数检测	0x22	34	15
正弦函数	0x2D	45	32 ~ 270
余弦函数	0x2E	46	32 ~ 270
正切函数	0x2F	47	58 ~ 258
反正切函数	0x30	48	62 ~ 175
浮点数转 8 位整数	0x23	35	19 ~ 30
浮点数转 16 位整数	0x24	36	19 ~ 30
浮点数转 32 位整数	0x25	37	23 ~ 39
8 位整数转浮点数	0x27	39	23 ~ 33
16 位整数转浮点数	0x28	40	23 ~ 33
32 位整数转浮点数	0x29	41	24 ~ 33
初始化协处理器	0x31	49	2
清除异常	0x32	50	4
读状态寄存器	0x33	51	4
写状态寄存器	0x34	52	4
读控制寄存器	0x35	53	4
写控制寄存器	0x36	54	4
TFPU 选择系统时钟	**0x3E**	**62**	**-**
TFPU 选择 PLL 高速时钟	**0x3F**	**63**	**-**

注：当使用 0x3E 和 0x3F 命令选择 TFPU 的时钟源后，再向 DMAIR 寄存器中写入其他运算指令不会改变 TFPU 的时钟源

35.4 TFPU 基本算术运算

本小节描述了 TFPU 模块使用 DMA 控制器可以执行的所有算术运算。所有操作数必须位于数据内存中。操作结果也存储在由 PSW（0xD0）位选择的 R0-R7 当前组的数据存储器空间中。

35.4.1 浮点数加法（+）

对两个浮点数进行加法运算。加数 BR 位于 R0~R3 寄存器中，被加数 AR 位于 R4~R7 寄存器中，计算结果和保存到 R4~R7 寄存器

和				=	被加数 AR				+	加数 BR			
MSB			LSB		MSB			LSB		MSB			LSB
R4	R5	R6	R7	=	R4	R5	R6	R7	+	R0	R1	R2	R3

指令码　0x1C(28)

执行时间（时钟数）　31 ~ 40

35.4.2 浮点数减法（－）

对两个浮点数进行减法运算。减数 BR 位于 R0~R3 寄存器中，被减数 AR 位于 R4~R7 寄存器中，计算结果差保存到 R4~R7 寄存器

差				=	被减数 AR				－	减数 BR			
MSB			LSB		MSB			LSB		MSB			LSB
R4	R5	R6	R7	=	R4	R5	R6	R7	－	R0	R1	R2	R3

指令码　0x1D(29)

执行时间（时钟数）　31 ~ 40

35.4.3 浮点数乘法（×）

对两个浮点数进行乘运算。乘数 BR 位于 R0~R3 寄存器中，被乘数 AR 位于 R4~R7 寄存器中，计算结果积保存到 R4~R7 寄存器

积				=	被乘数 AR				×	乘数 BR			
MSB			LSB		MSB			LSB		MSB			LSB
R4	R5	R6	R7	=	R4	R5	R6	R7	×	R0	R1	R2	R3

指令码　0x1E(30)

执行时间（时钟数）　26 ~ 34、

35.4.4 浮点数除法（÷）

对两个浮点数进行除运算。除数 BR 位于 R0~R3 寄存器中，被除数 AR 位于 R4~R7 寄存器中，计算结果商保存到 R4~R7 寄存器

商				=	被除数 AR				÷	除数 BR			
MSB			LSB		MSB			LSB		MSB			LSB
R4	R5	R6	R7	=	R4	R5	R6	R7	÷	R0	R1	R2	R3

指令码 0x1F(31)

执行时间（时钟数） 58 ~ 67

35.4.5 浮点数开方/平方根（sqrt）

对 1 个浮点数进行开方运算。被开方数 AR 位于 R4~R7 寄存器中，计算结果平方根保存到 R4~R7 寄存器

平方根				= sqrt	浮点数 AR			
MSB			LSB		MSB			LSB
R4	R5	R6	R7	= sqrt	R4	R5	R6	R7

指令码 0x20(32)

执行时间（时钟数） 50 - 54

35.4.6 浮点数比较（comp）

对两个浮点数进行算术比较运算。比较数 BR 位于 R0~R3 寄存器中，被比较数 AR 位于 R4~R7 寄存器中，比较结果保存到 R7 寄存器

R7.3	R7.2	R7.1	R7.0	比较结果	结果描述
0	0	0	0	0x0001	AR > BR
1	0	0	0	0x0000	AR = BR
0	0	0	1	0xFFFF	AR < BR
1	1	0	1	unchanged	Unordered

比较结果				=	被比较数 AR				?	比较数 BR			
			LSB		MSB			LSB		MSB			LSB
—	—	—	R7	=	R0	R1	R2	R3	?	R4	R5	R6	R7

指令码 0x21(33)

执行时间（时钟数） 18

35.4.7 浮点数检测（check）

对 1 个浮点数进行检测。被检测数 AR 位于 R4~R7 寄存器中，检测结果保存到 R7 寄存器

R7[7:4]	R7.3	R7.2	R7.1	R7.0	结果描述
0000	0	0	0	0	正非浮点数（+NaN）
0000	0	0	1	1	负非浮点数（-NaN）
0000	0	1	0	0	正规范浮点数
0000	0	1	1	0	负规范浮点数
0000	0	1	0	1	正无穷（+INF）
0000	0	1	1	1	负无穷（-INF）
0000	1	0	0	0	正零
0000	1	0	1	0	负零
0000	1	1	0	0	正非规范浮点数
0000	1	1	1	0	负非规范浮点数

检测结果				=	被检测数 AR			
			LSB		MSB			LSB
—	—	—	R7	=	R4	R5	R6	R7

指令码 0x22(34)

执行时间（时钟数） 15

35.5 TFPU 三角函数

注：所有三角函数的角度参数类型均为弧度。弧度与角度的转换公式：

$$角度 = \frac{180^\circ}{\pi} \times 弧度 \qquad 弧度 = \frac{\pi}{180^\circ} \times 角度$$

35.5.1 正弦函数（sin）

求一个单精度弧度浮点数的正弦值。弧度数 AR 位于 R4~R7 寄存器中，计算结果保存到 R4~R7 中

结果				= sin	弧度 AR			
MSB			LSB		MSB			LSB
R4	R5	R6	R7	= sin	R4	R5	R6	R7

指令码　　0x2D(45)

执行时间（时钟数）　32 - 270 clk

35.5.2 余弦函数（cos）

求一个单精度弧度浮点数的余弦值。弧度数 AR 位于 R4~R7 寄存器中，计算结果保存到 R4~R7 中

结果				= cos	弧度 AR			
MSB			LSB		MSB			LSB
R4	R5	R6	R7	= cos	R4	R5	R6	R7

指令码　　0x2E(46)

执行时间（时钟数）　32 - 270

35.5.3 正切函数（tan）

求一个单精度弧度浮点数的正切值。弧度数 AR 位于 R4~R7 寄存器中，计算结果保存到 R4~R7 中

结果				= tan	弧度 AR			
MSB			LSB		MSB			LSB
R4	R5	R6	R7	= tan	R4	R5	R6	R7

指令码　　0x2F(47)

执行时间（时钟数）　58 - 258

35.5.4 反正切函数（arctan）

求一个单精度弧度浮点数的反正切值。弧度数 AR 位于 R4~R7 寄存器中，计算结果保存到 R4~R7 中

result				= arctan	弧度 AR			
MSB			LSB		MSB			LSB
R4	R5	R6	R7	= arctan	R4	R5	R6	R7

指令码　　0x30(48)

执行时间（时钟数）　62 - 175

35.6 TFPU 数据转换操作

35.6.1 浮点数转 8 位整数（float → char）

将浮点数转换为 8 位整数（字符型 char）。浮点数位于 R4~R7 寄存器中，8 位整数保存到 R7 中

char				=	float			
			LSB		MSB			LSB
—	—	—	R7	=	R4	R5	R6	R7

指令码 0x23(35)

执行时间（时钟数） 19 - 30

35.6.2 浮点数转 16 位整数（float → short）

将浮点数转换为 16 位整数（短整型 short）。浮点数位于 R4~R7 寄存器中，16 位整数保存到 R6~R7 中

short				=	float			
		MSB	LSB		MSB			LSB
—	—	R6	R7	=	R4	R5	R6	R7

指令码 0x24(36)

执行时间（时钟数） 19 - 30

35.6.3 浮点数转 32 位整数（float → long）

将浮点数转换为 32 位整数（长整型 long）。浮点数 AR 位于 R4~R7 寄存器中，32 位整数保存到 R4~R7 中

long				=	float			
MSB			LSB		MSB			LSB
R4	R5	R6	R7	=	R4	R5	R6	R7

指令码 0x25(37)

执行时间（时钟数） 23 - 39

35.6.4 8 位整数转浮点数（char → float）

将 8 位整数（字符型 char）转换为浮点数。8 位整数位于 R7 寄存器中，浮点数保存到 R4~R7 中

float				=	char			
MSB			LSB					LSB
R4	R5	R6	R7	=	—	—	—	R7

指令码 0x27(39)

执行时间（时钟数） 23 - 33

35.6.5 16 位整数转浮点数（short → float）

将 16 位整数（短整型 short）转换为浮点数。16 位整数位于 R6~R7 寄存器中，浮点数保存到 R4~R7 中

float				=	short			
MSB			LSB				MSB	LSB
R4	R5	R6	R7	=	—	—	R6	R7

指令码 0x28(40)

执行时间（时钟数） 23 - 33

35.6.6 32 位整数转浮点数（long → float）

将 32 位整数（长整型 long）转换为浮点数。32 位整数位于 R4~R7 寄存器中，浮点数保存到 R4~R7 中

float				=	long			
MSB			LSB		MSB			LSB
R4	R5	R6	R7	=	R4	R5	R6	R7

指令码 0x29(41)

执行时间（时钟数） 24 - 33

35.7 TFPU 协处理器控制操作

35.7.1 初始化协处理器

初始化 TFPU 协处理器，初始化完成后会产生一个异常状态，需要软件清除。

指令码 0x31(49)

执行时间（时钟数） 2

35.7.2 清除异常

清除所有的异常状态

指令码 0x32(50)

执行时间（时钟数） 4

35.7.3 读状态寄存器

读取协处理器的状态寄存器。结果保存到 R7 中

指令码 0x33(51)

执行时间（时钟数） 4

35.7.4 写状态寄存器

写协处理器的状态寄存器。待写入的值位于 R7 寄存器中，完成后新的状态寄存器值保存到 R7 中

指令码 0x34(52)

执行时间（时钟数） 4

35.7.5 读控制寄存器

读取协处理器的控制寄存器。结果保存到 R7 中

指令码 0x35(53)

执行时间（时钟数） 4

35.7.6 写控制寄存器

写协处理器的控制寄存器。待写入的值位于 R7 寄存器中，完成后新的控制寄存器值保存到 R7 中

指令码 0x36(54)

执行时间（时钟数） 4

35.8 如何选择 TFPU 的时钟源

STC32G144K246 系列的 TFPU 可选择和 CPU 时钟同频，也可选择和 CPU 时钟异步且频率更高的 PLL 时钟源作为 TFPU 的运算时钟

35.8.1 选择系统时钟（和 CPU 时钟同步）作为 TFPU 时钟源

```
DMAIR = 0x3E;                    //此语句可放在系统初始化的地方
```

35.8.2 选择 PLL 时钟（和 CPU 时钟异步）作为 TFPU 时钟源

```
DMAIR = 0x3F;                    //此语句可放在系统初始化的地方
```

36 DSP32（DSP 指令，32 位及 64 位整数运算器）

注：DSP32 中的所有 32 位的乘法，结果一律为 64 位。（例如：64 位的乘加运算是两个 32 位整数相乘得到 64 位的积，然后和 64 位整数相加最后得到 64 位的结果）

产品线	DSP32
STC32G144K246 系列	●

36.1 相关的特殊功能寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
DPUST	DSP32 状态寄存器	86H	Z	DBZ	SC[5:0]						0000,0000
DPUOP	DSP32 指令寄存器	D8H	OPCODE[7:0]								0000,0000
DPUCFG	DSP32 配置寄存器	-	-	-	-	-	-	CDRS[2:0]			xxxx,x101

DSP32 状态寄存器

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DPUST	86H	Z	DBZ	SC[5:0]					

Z：零状态。

0：计算结果不为 0

1：计算结果为 0

DBZ：除数为0状态

0：除数不为 0

1：除数为 0

SC[5:0]：规格化时移动的位数。

运算过程中如果产生了进位标志和溢出标志，会直接修改到PSW中的CY和OV

DSP32 指令寄存器

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DPUOP	EDH	OP[7:0]							

DSP32的运算需要软件向寄存器DPUOP写入相应的指令码进行触发

DSP32 配置寄存器

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DPUCFG	-	-	-	-	-	-	CDRS[2:0]		

CDRS[2:0]：ECX和EDX寄存器组选择（详情参见：DSP32操作寄存器说明）

DPUCFG寄存器的修改方法：先将需要修改的值写入ACC寄存器中，然后DPUOP触发寄存器中写入0x80，触发硬件自动将ACC的值写入DPUCFG寄存器中。

36.2 DSP32 操作寄存器说明

PSW[4:3] / EAX/EBX		00	01	10	11	80251
EBX	BX2	DATA[0]	DATA[8]	DATA[10]	DATA[18]	DR0
		DATA[1]	DATA[9]	DATA[11]	DATA[19]	
	BX	DATA[2]	DATA[A]	DATA[12]	DATA[1A]	
		DATA[3]	DATA[B]	DATA[13]	DATA[1B]	
EAX	AX2	DATA[4]	DATA[C]	DATA[14]	DATA[1C]	DR4
		DATA[5]	DATA[D]	DATA[15]	DATA[1D]	
	AX	DATA[6]	DATA[E]	DATA[16]	DATA[1E]	
		DATA[7]	DATA[F]	DATA[17]	DATA[1F]	

CDRS[2:0] / ECX/EDX		000	001	010	011	101	110
EDX	DX2	DATA[0]	DATA[8]	DATA[10]	DATA[18]	DR16	DR24
		DATA[1]	DATA[9]	DATA[11]	DATA[19]		
	DX	DATA[2]	DATA[A]	DATA[12]	DATA[1A]		
		DATA[3]	DATA[B]	DATA[13]	DATA[1B]		
ECX	CX2	DATA[4]	DATA[C]	DATA[14]	DATA[1C]	DR20	DR28
		DATA[5]	DATA[D]	DATA[15]	DATA[1D]		
	CX	DATA[6]	DATA[E]	DATA[16]	DATA[1E]		
		DATA[7]	DATA[F]	DATA[17]	DATA[1F]		

36.3 运算执行时间表

DSP32 运算	指令码		执行时钟数	标志位			
	HEX	DEC		CY	OV	Z	DBZ
FILL_DPUCFG	0x80	128	1	–	–	–	–
BIN2BCD	0x81	129	36	–	–	–	–
BCD2BIN	0x82	130	13	–	–	–	–
NRM_BX	0x83	131	6	–	–	–	–
NRM_EBX	0x84	132	7	–	–	–	–
NRM_EABX	0x85	133	9	–	–	–	–
SWAP_EAB	0x86	134	3	–	–	–	–
SWAP_ECD	0x87	135	3	–	–	–	–
SWAP_EAC	0x88	136	3	–	–	–	–
SWAP_EBD	0x89	137	3	–	–	–	–
ADDC_EABX	0x90	144	3	✓	✓	✓	–
ADDC_ABX	0x91	145	3	✓	✓	✓	–
ADD_EABX	0x92	146	3	✓	✓	✓	–
ADD_ABX	0x93	147	3	✓	✓	✓	–
SUBB_EABX	0x94	148	3	✓	✓	✓	–
SUBB_ABX	0x95	149	3	✓	✓	✓	–
SUB_EABX	0x96	150	3	✓	✓	✓	–
SUB_ABX	0x97	151	3	✓	✓	✓	–
CMP_EABX	0x98	152	3	✓	✓	✓	–
CMP_ABX	0x99	153	3	✓	✓	✓	–
MULU_EABX	0x9A	154	8	–	–	–	–
MULS_EABX	0x9B	155	10	–	–	–	–
MULU_ABX	0x9C	156	3	–	–	–	–
MULS_ABX	0x9D	157	4	–	–	–	–
MULX_KEIL16	0x9E	158	3	–	–	–	–
DIVU_EABX	0x9F	159	19	–	–	–	✓
DIVS_EABX	0xA0	160	21	–	–	–	✓
DIVU_ABX	0xA1	161	11	–	–	–	✓
DIVU_KEIL16	0xA2	162	10	–	–	–	✓
DIVS_ABX	0xA3	163	13	–	–	–	✓
DIVS_KEIL16	0xA4	164	12	–	–	–	✓
DIVU_EABXD	0xA5	165	36	–	–	–	✓
DIVU_EAXB	0xA6	166	19	–	–	–	✓
SET0_EAX	0xB0	176	2	–	–	–	–
SET1_EAX	0xB1	177	2	–	–	–	–
SET0_EBX	0xB2	178	2	–	–	–	–
SET1_EBX	0xB3	179	2	–	–	–	–
SET0_ECX	0xB4	180	2	–	–	–	–
SET1_ECX	0xB5	181	2	–	–	–	–

DSP32 运算	指令码		执行时钟数	标志位			
	HEX	DEC		CY	OV	Z	DBZ
SET0_EDX	0xB6	182	2	–	–	–	–
SET1_EDX	0xB7	183	2	–	–	–	–
NEGS_EAX	0xB8	184	2	–	–	–	–
NEGS_EBX	0xB9	185	2	–	–	–	–
NEGS_AX	0xBA	186	2	–	–	–	–
NEGS_BX	0xBB	187	2	–	–	–	–
INC1_EAX	0xC0	192	3	✓	–	✓	–
INC1_EBX	0xC1	193	3	✓	–	✓	–
INC4_EAX	0xC2	194	3	✓	–	✓	–
INC4_EBX	0xC3	195	3	✓	–	✓	–
INC1_AX	0xC4	196	3	✓	–	✓	–
INC1_BX	0xC5	197	3	✓	–	✓	–
INC2_AX	0xC6	198	3	✓	–	✓	–
INC2_BX	0xC7	199	3	✓	–	✓	–
DEC1_EAX	0xC8	200	3	✓	–	✓	–
DEC1_EBX	0xC9	201	3	✓	–	✓	–
DEC4_EAX	0xCA	202	3	✓	–	✓	–
DEC4_EBX	0xCB	203	3	✓	–	✓	–
DEC1_AX	0xCC	204	3	✓	–	✓	–
DEC1_BX	0xCD	205	3	✓	–	✓	–
DEC2_AX	0xCE	206	3	✓	–	✓	–
DEC2_BX	0xCF	207	3	✓	–	✓	–
BAND_EABX	0xD0	208	3	–	–	✓	–
BAND_ABX	0xD1	209	3	–	–	✓	–
BOR_EABX	0xD2	210	3	–	–	✓	–
BOR_ABX	0xD3	211	3	–	–	✓	–
BXOR_EABX	0xD4	212	3	–	–	✓	–
BXOR_ABX	0xD5	213	3	–	–	✓	–
BCPL_EAX	0xD6	214	2	–	–	✓	–
BCPL_EBX	0xD7	215	2	–	–	✓	–
SHL_EAX	0xE0	224	7	✓	–	–	–
SHL_EBX	0xE1	225	7	✓	–	–	–
SHL_AX	0xE2	226	6	✓	–	–	–
SHL_BX	0xE3	227	6	✓	–	–	–
SHRU_EAX	0xE4	228	7	✓	–	–	–
SHRU_EBX	0xE5	229	7	✓	–	–	–
SHRU_AX	0xE6	230	6	✓	–	–	–
SHRU_BX	0xE7	231	6	✓	–	–	–
SHRS_EAX	0xE8	232	7	✓	–	–	–
SHRS_EBX	0xE9	233	7	✓	–	–	–
SHRS_AX	0xEA	234	6	✓	–	–	–

DSP32 运算	指令码		执行时钟数	标志位			
	HEX	DEC		CY	OV	Z	DBZ
SHRS_BX	0xEB	235	6	✓	–	–	–
ROR_EAX	0xEC	236	7	–	–	–	–
ROR_EBX	0xED	237	7	–	–	–	–
ROR_AX	0xEE	238	6	–	–	–	–
ROR_BX	0xEF	239	6	–	–	–	–
MMD32_EABX	0xF0	240	43	–	–	–	✓
MMD16_ABX	0xF1	241	21	–	–	–	✓
LTC32_EAX	0xF2	242	44	✓	✓	–	✓
LTC16_AX	0xF3	243	22	✓	✓	–	✓
MA32_EDX	0xF4	244	5	✓	✓	–	–
MA64_ECDX	0xF5	245	12	✓	✓	–	–

36.4 DSP32 指令说明

36.4.1 配置 DPU（FILL_DPUCFG）

FILL_DPUCFG

指令操作: DPUCFG[7:0] = ACC[7:0]

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] 80H

时钟数: 1

36.4.2 编码转换

36.4.2.1 BIN 转 BCD（BIN2BCD）

BIN2BCD

指令操作: { EAX, EBX } <= BCD(EBX)

指令说明: 将 EBX 中的 BIN 码转换为 BCD 码后写入{ EAX, EBX }

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] 81H

时钟数: 36

36.4.2.2 BCD 转 BIN（BCD2BIN）

BCD2BIN

指令操作: EAX <= BIN({EAX[6:0], EBX})

指令说明: 将{EAX[6:0], EBX}中的 BCD 码转换为 BIN 码后写入 EAX

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] 82H

时钟数: 13

36.4.3 规格化

36.4.3.1 16 位 AX 规格化（NRM_BX）

NRM_BX

指令操作: BX=NRM(BX); SC[5:0] <= Shifted#

指令说明: 规格化过程中移动的位数存放在 SC[5:0]中

影响标志位:

CY	OV	Z	DBZ
–	–	-	-

[指令码] 83H

时钟数: 6

36.4.3.2 32 位 EBX 规格化（NRM_EBX）

NRM_EBX

指令操作: EBX=NRM(EBX); SC[5:0] <= Shifted#

指令说明: 规格化过程中移动的位数存放在 SC[5:0]中

影响标志位:

CY	OV	Z	DBZ
–	–	-	-

[指令码] 84H

时钟数: 7

36.4.3.3 64 位｛EAX, EBX｝规格化（NRM_EABX）

NRM_EABX

指令操作: {EAX,EBX}=NRM({EAX,EBX}); SC[5:0] <= Shifted#

指令说明: 规格化过程中移动的位数存放在 SC[5:0]中

影响标志位:

CY	OV	Z	DBZ
–	–	-	-

[指令码] 85H

时钟数: 8

36.4.4 互换运算

36.4.4.1 EAX 与 EBX 内容互换（SWAP_EAB）

SWAP_EAB

指令操作: (EAX) ↔(EBX)

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] 86H

时钟数: 3

36.4.4.2 ECX 与 EDX 内容互换（SWAP_ECD）

SWAP_ECD

指令操作: (ECX) ↔(EDX)

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] 87H

时钟数: 3

36.4.4.3 EAX 与 ECX 内容互换（SWAP_EAC）

SWAP_EAC

指令操作: (EAX) ↔(ECX)

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] 88H

时钟数: 3

36.4.4.4 EBX 与 EDX 内容互换（SWAP_EBD）

SWAP_EBD

指令操作: (EBX) ↔(EDX)

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] 89H

时钟数: 3

36.4.5 加减乘除运算

36.4.5.1 32 位带进位加法（ADDC_EABX）

ADDC_EABX

指令操作: EAX=EAX + EBX+CY

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	✓	-

[指令码] 90H

时钟数: 3

36.4.5.2 16 位带进位加法（ADDC_ABX）

ADDC_ABX

指令操作: AX= AX+ BX + CY

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	✓	-

[指令码] 91H

时钟数: 3

36.4.5.3 32 位加法（ADD_EABX）

ADD_EABX

指令操作: EAX=EAX + EBX

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	✓	-

[指令码] 92H

时钟数: 3

36.4.5.4 16 位加法（ADD_ABX）

ADD_ABX

指令操作: AX=AX + BX

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	✓	-

[指令码] 93H

时钟数: 3

36.4.5.5 32 位带借位减法（SUBB_EABX）

SUBB_EABX

指令操作: EAX=EAX – EBX - CY

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	✓	-

[指令码] 94H

时钟数: 3

36.4.5.6 16 位带借位减法（SUBB_ABX）

SUBB_ABX

指令操作: AX=AX – BX - CY

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	✓	-

[指令码] 95H

时钟数: 3

36.4.5.7 32 位减法（SUB_EABX）

SUB_EABX

指令操作: EAX=EAX – EBX

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	✓	-

[指令码] 96H

时钟数: 3

36.4.5.8 16 位减法（SUB_ABX）

SUB_ABX

指令操作: AX=AX – BX

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	✓	-

[指令码] 97H

时钟数: 3

36.4.5.9 32 位无符号数乘法（MULU_EABX）

MULU_EABX

指令操作: { EDX, EAX } = EAX * EBX

指令说明: 结果 64 位，高 32 位在 EDX，低 32 位在 EAX

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] 9AH

时钟数: 8

36.4.5.10 32 位有符号数乘法（MULS_EABX）

MULS_EABX

指令操作: { EDX, EAX } = EAX * EBX

指令说明: 结果 64 位，高 32 位在 EDX，低 32 位在 EAX

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] 9BH

时钟数: 10

36.4.5.11 16 位无符号数乘法（MULU_ABX）

MULU_ABX

指令操作: EAX = AX * BX

指令说明: 结果 32 位，乘积低 16 位在 AX,高 16 位在 AX2,合并为 EAX

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] 9CH

时钟数: 3

36.4.5.12 16 位有符号数乘法（MULS_ABX）

MULS_ABX

指令操作: EAX = AX * BX

指令说明: 结果 32 位，乘积低 16 位在 AX,高 16 位在 AX2,合并为 EAX

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] 9DH

时钟数: 4

36.4.5.13 兼容 KEIL 编译器的 16 位无符号数乘法（MULX_KEIL16）

MULX_KEIL16

指令操作: AX=AX2 * AX

指令说明: 结果 16 位在 AX

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] 9EH

时钟数: 3

36.4.5.14 32 位无符号除法（DIVU_EABX）

DIVU_EABX

指令操作: EAX=EAX / EBX ;

EBX=EAX % EBX

指令说明: 结果 32 位商在 EAX，32 位余数在 EBX

影响标志位:

CY	OV	Z	DBZ
-	-	-	✓

[指令码] 9FH

时钟数: 19

36.4.5.15 32 位有符号除法（DIVS_EABX）

DIVS_EABX

指令操作: EAX=EAX / EBX ;

EBX=EAX % EBX

指令说明: 结果 32 位商在 EAX，32 位余数在 EBX

影响标志位:

CY	OV	Z	DBZ
-	-	-	✓

[指令码] A0H

时钟数: 21

36.4.5.16 16 位无符号除法（DIVU_ABX）

DIVU_ABX

指令操作: AX=AX / BX ;

BX=AX % BX

指令说明: 结果 16 位商在 AX，16 位余数在 BX

影响标志位:

CY	OV	Z	DBZ
-	-	-	✓

[指令码] A1H

时钟数: 11

36.4.5.17 兼容 KEIL 编译器的 16 位无符号除法（DIVU_KEIL16）

DIVU_KEIL16

指令操作: AX=AX / AX2 ;

AX2=AX % AX2

指令说明: 结果 16 位商在 AX，16 位余数在 AX2

影响标志位:

CY	OV	Z	DBZ
-	-	-	✓

[指令码] A2H

时钟数: 10

36.4.5.18 16 位有符号除法（DIVS_ABX）

DIVS_ABX

指令操作: AX=AX / BX ;

BX=AX % BX

指令说明: 结果 16 位商在 AX，16 位余数在 BX

影响标志位:

CY	OV	Z	DBZ
-	-	-	✓

[指令码] A3H

时钟数: 13

36.4.5.19 兼容 KEIL 编译器的 16 位有符号除法（DIVS_KEIL16）

DIVS_KEIL16

指令操作: AX=AX / AX2 ;

AX2=AX % AX2

指令说明: 结果 16 位商在 AX，16 位余数在 AX2

影响标志位:

CY	OV	Z	DBZ
-	-	-	✓

[指令码] A4H

时钟数: 12

36.4.5.20 64 位无符号除法（DIVU_EABXD）（64 位除以 32 位）

DIVU_EABXD

指令操作: {EAX, EBX} = ({EAX, EBX} / EDX)

EDX = ({EAX, EBX} % EDX)

指令说明: 结果 64 位商在{EAX, EBX}，32 位余数在 EDX

影响标志位:

CY	OV	Z	DBZ
-	-	-	✓

[指令码] A5H

时钟数: 36

36.4.5.21 32 位无符号除法（DIVU_EBXA）（32 位除以 16 位）

DIVU_EAXB

指令操作: EAX = (EAX / BX)

BX = (EAX % BX)

指令说明: 结果 32 位商在 EAX，16 位余数在 BX

影响标志位:

CY	OV	Z	DBZ
-	-	-	✓

[指令码] A6H

时钟数: 19

36.4.6 比较运算

36.4.6.1 32 位比较（CMP_EABX）

CMP_EABX

指令操作: EAX – EBX

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	✓	-

[指令码] 98H

时钟数: 3

36.4.6.2 16 位比较（CMP_ABX）

CMP_ABX

指令操作: AX – BX

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	✓	-

[指令码] 99H

时钟数: 3

36.4.7 清零/置位运算

36.4.7.1 清零 32 位 EAX（SET0_EAX）

SET0_EAX

指令操作: EAX=0

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] B0H

时钟数: 2

36.4.7.2 置位 32 位 EAX（SET1_EAX）

SET1_EAX

指令操作: EAX=0xFFFFFFFF

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] B1H

时钟数: 2

36.4.7.3 清零 32 位 EBX（SET0_EBX）

SET0_EBX

指令操作: EBX=0

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] B2H

时钟数: 2

36.4.7.4 置位 32 位 EBX（SET1_EBX）

SET1_EBX

指令操作: EBX=0xFFFFFFFF

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] B3H

时钟数: 2

36.4.7.5 清零 32 位 ECX（SET0_ECX）

SET0_ECX

指令操作: ECX=0

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] B4H

时钟数: 2

36.4.7.6 置位 32 位 ECX（SET1_ECX）

SET1_ECX

指令操作: ECX=0xFFFFFFFF

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] B5H

时钟数: 2

36.4.7.7 清零 32 位 EDX（SET0_EDX）

SET0_EDX

指令操作: EDX=0

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] B6H

时钟数: 2

36.4.7.8 置位 32 位 EDX（SET1_EDX）

SET1_EDX

指令操作: EDX=0xFFFFFFFF

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] B7H

时钟数: 2

36.4.8 取负数运算

36.4.8.1 32 位 EAX 取负数（NEGS_EAX）

NEGS_EAX

指令操作: EAX=-EAX

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] B8H

时钟数: 2

36.4.8.2 32 位 EBX 取负数（NEGS_EBX）

NEGS_EBX

指令操作: EBX=-EBX

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] B9H

时钟数: 2

36.4.8.3 16 位 AX 取负数（NEGS_AX）

NEGS_AX

指令操作: AX=-AX

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] BAH

时钟数: 2

36.4.8.4 16 位 BX 取负数（NEGS_BX）

NEGS_BX

指令操作: BX=-BX

指令说明:

影响标志位:

CY	OV	Z	DBZ
-	-	-	-

[指令码] BBH

时钟数: 2

36.4.9 累加运算

36.4.9.1 32 位 EAX 加 1 运算（INC1_EAX）

INC1_EAX

指令操作: EAX=EAX+1

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] C0H

时钟数: 3

36.4.9.2 32 位 EBX 加 1 运算（INC1_EBX）

INC1_EBX

指令操作: EBX=EBX+1

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] C1H

时钟数: 3

36.4.9.3 32 位 EAX 加 4 运算（INC4_EAX）

INC4_EAX

指令操作: EAX=EAX+4

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] C2H

时钟数: 3

36.4.9.4 32 位 EBX 加 4 运算（INC4_EBX）

INC4_EBX

指令操作: EBX=EBX+4

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] C3H

时钟数: 3

36.4.9.5 16 位 AX 加 1 运算（INC1_AX）

INC1_AX

指令操作: AX=AX+1

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] C4H

时钟数: 3

36.4.9.6 16 位 BX 加 1 运算（INC1_BX）

INC1_BX

指令操作: BX=BX+1

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] C5H

时钟数: 3

36.4.9.7 16 位 AX 加 2 运算（INC2_AX）

INC2_AX

指令操作: AX=AX+2

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] C6H

时钟数: 3

36.4.9.8 16 位 BX 加 2 运算（INC2_BX）

INC2_BX

指令操作: BX=BX+2

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] C7H

时钟数: 3

36.4.9.9 32 位 EAX 减 1 运算（DEC1_EAX）

DEC1_EAX

指令操作: EAX=EAX-1

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] C8H

时钟数: 3

36.4.9.10 32 位 EBX 减 1 运算（DEC1_EBX）

DEC1_EBX

指令操作: EBX=EBX-1

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] C9H

时钟数: 3

36.4.9.11 32 位 EAX 减 4 运算（DEC4_EAX）

DEC4_EAX

指令操作: EAX=EAX-4

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] CAH

时钟数: 3

36.4.9.12 32 位 EBX 减 4 运算（DEC4_EBX）

DEC4_EBX

指令操作: EBX=EBX-4

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] CBH

时钟数: 3

36.4.9.13 16 位 AX 减 1 运算（DEC1_AX）

DEC1_AX

指令操作: AX=AX-1

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] CCH

时钟数: 3

36.4.9.14 16 位 BX 减 1 运算（DEC1_BX）

DEC1_BX

指令操作: BX=BX-1

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] CDH

时钟数: 3

36.4.9.15 16 位 AX 减 2 运算（DEC2_AX）

DEC2_AX

指令操作: AX=AX-2

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] CEH

时钟数: 3

36.4.9.16 16 位 BX 减 2 运算（DEC2_BX）

DEC2_BX

指令操作: BX=BX-2

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	–	✓	-

[指令码] CFH

时钟数: 3

36.4.10 逻辑运算

36.4.10.1 32 位与运算（BAND_EABX）

BAND_EABX

指令操作: EBX=EBX & EAX

指令说明:

影响标志位:

CY	OV	Z	DBZ
–	–	✓	-

[指令码] D0H

时钟数: 3

36.4.10.2 16 位与运算（BAND_ABX）

BAND_ABX

指令操作: BX=BX & AX

指令说明:

影响标志位:

CY	OV	Z	DBZ
–	–	✓	-

[指令码] D1H

时钟数: 3

36.4.10.3 32 位或运算（BOR_EABX）

BOR_EABX

指令操作: EBX=EBX | EAX

指令说明:

影响标志位:

CY	OV	Z	DBZ
–	–	✓	-

[指令码] D2H

时钟数: 3

36.4.10.4 16 位或运算（BOR_ABX）

BOR_ABX

指令操作: BX=BX | AX

指令说明:

影响标志位:

CY	OV	Z	DBZ
–	–	✓	-

[指令码] D3H

时钟数: 3

36.4.10.5 32 位异或运算（BXOR_EABX）

BXOR_EABX

指令操作: EBX=EBX ^ EAX

指令说明:

影响标志位:

CY	OV	Z	DBZ
–	–	✓	-

[指令码] D4H

时钟数: 3

36.4.10.6 16 位异或运算（BXOR_ABX）

BXOR_ABX

指令操作: BX=BX ^ AX

指令说明:

影响标志位:

CY	OV	Z	DBZ
–	–	✓	-

[指令码] D5H

时钟数: 3

36.4.10.7 32 位取反运算（BCPL_EAX）

BCPL_EAX

指令操作: EAX=~EAX

指令说明:

影响标志位:

CY	OV	Z	DBZ
–	–	✓	-

[指令码] D6H

时钟数: 2

36.4.10.8 32 位取反运算（BCPL_EBX）

BCPL_EBX

指令操作: EBX=~EBX

指令说明:

影响标志位:

CY	OV	Z	DBZ
–	–	✓	-

[指令码] D7H

时钟数: 2

36.4.11 移位运算

36.4.11.1 32 位 EAX 逻辑左移 n 位（SHL_EAX）

SHL_EAX

指令操作: {CY, EAX}=EAX << ACC[4:0]

指令说明: 需要移动位数存放在 ACC[4:0]中,低位填充 0

影响标志位:

CY	OV	Z	DBZ
✓	–	–	-

[指令码] E0H

时钟数: 7

36.4.11.2 32 位 EBX 逻辑左移 n 位（SHL_EBX）

SHL_EBX

指令操作: (CY, EBX)=EBX << ACC[4:0]

指令说明: 需要移动位数存放在 ACC[4:0]中,低位填充 0

影响标志位:

CY	OV	Z	DBZ
✓	–	–	-

[指令码] E1H

时钟数: 7

36.4.11.3 16 位 AX 逻辑左移 n 位（SHL_AX）

SHL_AX

指令操作: {CY, AX}=AX << ACC[3:0]

指令说明: 需要移动位数存放在 ACC[3:0]中,低位填充 0

影响标志位:

CY	OV	Z	DBZ
✓	–	–	-

[指令码] E2H

时钟数: 6

36.4.11.4 16 位 BX 逻辑左移 n 位（SHL_BX）

SHL_BX

指令操作: {CY, BX}=BX << ACC[3:0]

指令说明: 需要移动位数存放在 ACC[3:0]中,低位填充 0

影响标志位:

CY	OV	Z	DBZ
✓	–	–	-

[指令码] E3H

时钟数: 6

36.4.11.5 32 位 EAX 逻辑右移 n 位（SHRU_EAX）

SHRU_EAX

指令操作: {EAX, CY}=EAX >> ACC[4:0]

指令说明: 无符号整数右移，需要移动位数存放在 ACC[4:0]中,高位填充 0

影响标志位:

CY	OV	Z	DBZ
✓	–	–	-

[指令码] E4H

时钟数: 7

36.4.11.6 32 位 EBX 逻辑右移 n 位（SHRU_EBX）

SHRU_EBX

指令操作: {EBX, CY}=EBX >> ACC[4:0]

指令说明: 无符号整数右移，需要移动位数存放在 ACC[4:0]中,高位填充 0

影响标志位:

CY	OV	Z	DBZ
✓	–	–	-

[指令码] E5H

时钟数: 7

36.4.11.7 16 位 AX 逻辑右移 n 位（SHRU_AX）

SHRU_AX

指令操作: {AX, CY}=AX >> ACC[3:0]

指令说明: 无符号整数右移，需要移动位数存放在 ACC[3:0]中,高位填充 0

影响标志位:

CY	OV	Z	DBZ
✓	–	–	-

[指令码] E6H

时钟数: 6

36.4.11.8 16 位 BX 逻辑右移 n 位（SHRU_BX）

SHRU_BX

指令操作: {BX, CY}=BX >> ACC[3:0]

指令说明: 无符号整数右移，需要移动位数存放在 ACC[3:0]中,高位填充 0

影响标志位:

CY	OV	Z	DBZ
✓	–	–	-

[指令码] E7H

时钟数: 6

36.4.11.9 32 位 EAX 算术右移 n 位（SHRS_EAX）

SHRS_EAX

指令操作: {EAX, CY}=EAX >> ACC[4:0]

指令说明: 有符号整数右移，需要移动位数存放在 ACC[4:0]中,高位填充符号位位

影响标志位:

CY	OV	Z	DBZ
✓	–	–	-

[指令码] E8H

时钟数: 7

36.4.11.10 32 位 EBX 算术右移 n 位（SHRS_EBX）

SHRS_EBX

指令操作: {EBX, CY}=EBX >> ACC[4:0]

指令说明: 有符号整数右移，需要移动位数存放在 ACC[4:0]中,高位填充符号位位

影响标志位:

CY	OV	Z	DBZ
✓	–	–	-

[指令码] E9H

时钟数: 7

36.4.11.11 16 位 AX 算术右移 n 位（SHRS_AX）

SHRS_AX

指令操作: {AX, CY}=AX >> ACC[3:0]

指令说明: 有符号整数右移，需要移动位数存放在 ACC[3:0]中,高位填充符号位位

影响标志位:

CY	OV	Z	DBZ
✓	–	–	-

[指令码] EAH

时钟数: 6

36.4.11.12 16 位 BX 算术右移 n 位（SHRS_BX）

SHRS_BX

指令操作: {BX, CY}=BX >> ACC[3:0]

指令说明: 有符号整数右移，需要移动位数存放在 ACC[3:0]中,高位填充符号位位

影响标志位:

CY	OV	Z	DBZ
✓	–	–	-

[指令码] EBH

时钟数: 6

36.4.11.13 32 位 EAX 循环右移 n 位（ROR_EAX）

ROR_EAX

指令操作: EAX=EAX >> ACC[4:0]

指令说明: 无符号整数循环右移，需要移动位数存放在 ACC[4:0]中

影响标志位:

CY	OV	Z	DBZ
–	–	–	-

[指令码] ECH

时钟数: 7

36.4.11.14 32 位 EBX 循环右移 n 位（ROR_EBX）

ROR_EBX

指令操作: EBX=EBX >> ACC[4:0]

指令说明: 无符号整数循环右移，需要移动位数存放在 ACC[4:0]中

影响标志位:

CY	OV	Z	DBZ
–	–	–	-

[指令码] EDH

时钟数: 7

36.4.11.15 16 位 AX 循环右移 n 位（ROR_AX）

ROR_AX

指令操作: AX=AX >> ACC[3:0]

指令说明: 无符号整数循环右移，需要移动位数存放在 ACC[3:0]中

影响标志位:

CY	OV	Z	DBZ
–	–	–	-

[指令码] EEH

时钟数: 6

36.4.11.16 16 位 BX 循环右移 n 位（ROR_BX）

ROR_BX

指令操作: BX=BX >> ACC[3:0]

指令说明: 无符号整数循环右移，需要移动位数存放在 ACC[3:0]中

影响标志位:

CY	OV	Z	DBZ
–	–	–	-

[指令码] EFH

时钟数: 6

36.4.12 复合运算

36.4.12.1 32 位乘除运算（MMD32_EABX）

MMD32_EABX

指令操作: {ECX,EAX}=EAX*EBX/EDX

指令说明: EAX、EBX、EDX 均为有符号数

影响标志位:

CY	OV	Z	DBZ
–	–	–	✓

[指令码] F0H

时钟数: 43

36.4.12.2 16 位乘除运算（MMD16_ABX）

MMD16_ABX

指令操作: EAX=AX*BX/DX

指令说明: AX、BX、DX 均为有符号数

影响标志位:

CY	OV	Z	DBZ
–	–	–	✓

[指令码] F1H

时钟数: 21

36.4.12.3 32 位线性标定（LTC32_EAX）

LTC32_EAX

指令操作: {ECX, EAX}=(EAX-EBX)*ECX/EDX

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	–	✓

[指令码] F2H

时钟数: 44

36.4.12.4 16 位线性标定（LTC16_AX）

LTC16_AX

指令操作: EAX=(AX-BX)*CX/DX

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	–	✓

[指令码] F3H

时钟数: 22

36.4.12.5 32 位乘加运算（MA32_EDX）

MA32_EDX

指令操作: EDX=EDX+AX*BX

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	–	–

[指令码] F4H

时钟数: 5

36.4.12.6 64 位乘加运算（MA64_EDX）

MA64_ECDX

指令操作: { ECX, EDX }={ECX, EDX}+EAX*EBX

指令说明:

影响标志位:

CY	OV	Z	DBZ
✓	✓	–	–

[指令码] F5H

时钟数: 12

37 硬件 CRC32/16（循环冗余校验）

产品线	CRC
STC32G144K246 系列	●

STC32G144K246 系列单片机内部集成了 32/16 位硬件循环冗余校验。CRC 是使用硬件电路进行循环冗余校验，软件只需要将需要校验的数据序列依次写入 CRC_DIN 数据输入寄存器，硬件会自动进行 CRC 计算，计算过程不需要额外的触发。由于是使用硬件组合逻辑电路进行计算，所以每次写入一个字节到 CRC_DIN 后，需要 1T 的系统时钟完成计算，所以对 CRC 校验结果的读取要等至少 1T 后再读取结果。

STC32G144K246 系列的 CRC 包含 CRC16 和 CRC32 两种校验模式：
CRC16 使用的是工业控制中常用的 CRC16-MODBUS 校验算法，CRC 多项式为：$(x^{16} + x^{15} + x^{2} + 1)$；
CRC32 使用的是适用于以太网 IEEE802.3 标准中算法，CRC 多项式为：$(x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1)$

37.1 相关的特殊功能寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
CRC_CR	CRC 控制寄存器	7EFB30H	RST	-	-	MODE	-	INVOUT	REVIN	REVOUT	0xx0,x011
CRC_DIN	CRC 数据输入寄存器	7EFB31H	DIN[7:0]								0000,0000
CRC_DOH	16 位 CRC 结果高字节	7EFB32H	DOUT_16[15:8]								1111,1111
CRC_DOL	16 位 CRC 结果低字节	7EFB33H	DOUT_16[7:0]								1111,1111
CRC_DO0	32 位 CRC 结果最高字节	7EFB32H	DOUT_32[31:24]								1111,1111
CRC_DO1	32 位 CRC 结果高字节	7EFB33H	DOUT_32[23:16]								1111,1111
CRC_DO2	32 位 CRC 结果低字节	7EFB34H	DOUT_32[15:8]								1111,1111
CRC_DO3	32 位 CRC 结果最字节	7EFB35H	DOUT_32[7:0]								1111,1111

37.1.1 CRC 控制寄存器（CRC_CR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CRC_CR	7EFB30H	RST	-	-	MODE	-	INVOUT	REVIN	REVOUT

RST：复位 CRC 模块。软件向此位写 1 触发复位 CRC 模块，复位完成后，硬件自动将此位清零。

MODE：选择 CRC 模式

0：CRC16，适用于 MODEBUS 协议

1：CRC32，适用于 IEEE802.3 标准以太网协议

INVOUT：计算结果取反控制

0：结果不取反

1：结果取反（计算结果与 FFFF 或 FFFFFFFF 异或）

REVIN：CRC 输入数据反转控制（数据的高位和低位交换，即 bit0 和 bit7 交换，bit1 和 bit6 交换，…）

0：输入数据不反转。写入 CRC_DIN 的数据直接参与 CRC 运算

1：数据数据反转。写入 CRC_DIN 的数据先反转，再参与 CRC 运算

REVOUT：CRC 的输出结果反转控制

0：结果不反转。内部计算的 CRC 结果直接输出到 CRC16_DOH 和 CRC16_DOL

1：结果反转。内部计算的 CRC 结果先反转，再输出到 CRC16_DOH 和 CRC16_DOL

37.1.2 CRC 数据输入寄存器（CRC_DIN）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CRC_DIN	7EFB31H	DIN[7:0]							

DIN[7:0]：参与 CRC 计算的字节数据

37.1.3 16 位 CRC 计算结果寄存器（CRC_DOH、CRC_DOL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CRC_DOH	7EFB32H	DOUT_16[15:8]							
CRC_DOL	7EFB33H	DOUT_16[7:0]							

DOUT_16[15:0]：16 位 CRC 的计算结果

37.1.4 32 位 CRC 计算结果寄存器（CRC_DO0/1/2/3）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CRC_DO0	7EFB32H	DOUT_32[31:24]							
CRC_DO1	7EFB33H	DOUT_32[23:16]							
CRC_DO2	7EFB34H	DOUT_32[15:8]							
CRC_DO3	7EFB35H	DOUT_16[7:0]							

DOUT_32[31:0]：32 位 CRC 的计算结果

38 CAN_FD，兼容 CAN2.0B/A

产品线	CAN-FD
STC32G144K246 系列	●

38.1 CAN-FD 简介

38.1.1 为什么开发 CAN-FD

CAN 网络在现有物理限制下已达到数据传输的实际最大值：

- 许多 CAN 总线的负载水平已高达 50%~95%，接近饱和状态，导致延迟增加和可靠性下降。
- 当前 CAN 总线的传输速率不超过 1Mbit/秒，这一上限限制了实时数据交换的能力，无法满足现代车载系统的高带宽需求。

受限于车载布线的物理特性：

- CAN 报文报头占比超过 50%，仅 40%~50%的带宽用于有效数据传输，造成大量带宽浪费。
- 标准 CAN 报文：每帧 129 比特，其中仅 64 比特承载有效数据，其余用于控制信息。
- 扩展 CAN 报文：每帧 154 比特，其中仅 64 比特承载有效数据，报头开销更大。

帧内响应（IFR）机制进一步制约 CAN 总线速率：

- CAN 控制器 ACK 生成延迟主要源于以下因素：

1) 收发器传播延时，影响信号转换时间。
2) 总线传播延时，取决于线缆长度和信号传输速度。

标准 CAN 的非破坏性仲裁机制设定了位速率上限（在特定总线长度下）：

- 该机制要求任意两个节点的相位切换时间小于 1/2 位时长，以确保仲裁过程中无数据冲突，但限制了最大可实现的比特率。

CAN-FD 是基于 CAN 2.0 物理层的串行通信协议，通过技术改进提升性能：

1. 帧结构优化：

1) 支持单报文双比特率，仲裁段保持标准 CAN 速率（如 500kbit/s），数据段采用更高比特率（如 2Mbit/s），有效提升吞吐量。
2) 支持扩展数据长度，单报文可收发最多 64 字节数据，相比标准 CAN 的 8 字节显著增加有效载荷容量。

2. CRC 校验机制升级：

1) 采用更长的 CRC 校验序列，根据数据长度动态选择 17 位或 21 位 CRC（标准 CAN 为 15 位），提升对长数据帧的错误检测能力，降低未检测到错误的概率。
2) 引入 CRC 分隔符（DEL）和固定填充位，CRC 场包含 6 个填充位，通过严格的位填充规则确保校验序列的独特性，避免与数据段混淆，增强校验可靠性。

3. 控制字段扩展：

1) 新增 FDF 位（FD Format，位于控制字段），显性位表示 FD 帧格式，隐性位表示传统 CAN 帧，使传统 CAN 节点可忽略 FD 帧，实现协议兼容性。
2) 引入 BRS 位（Bit Rate Switch），显性位时数据段维持仲裁段速率，隐性位时触发数据段比特率切换，通过硬件支持的相位补偿实现速率无缝切换。
3) 增加 ESI 位（Error State Indicator），由发送节点的错误状态决定（错误激活节点发送显性位，错误被动节点发送隐性位），使接收节点可实时监测发送节点健康状态。

4. 错误处理机制增强：

1) 优化错误帧结构，缩短错误标志长度，减少错误处理对总线带宽的占用，降低错误传播对正常通信的影响。

2) 支持错误状态动态调整，通过 ESI 位反馈错误状态，使网络管理节点可快速定位故障节点，提升系统容错能力和可维护性。

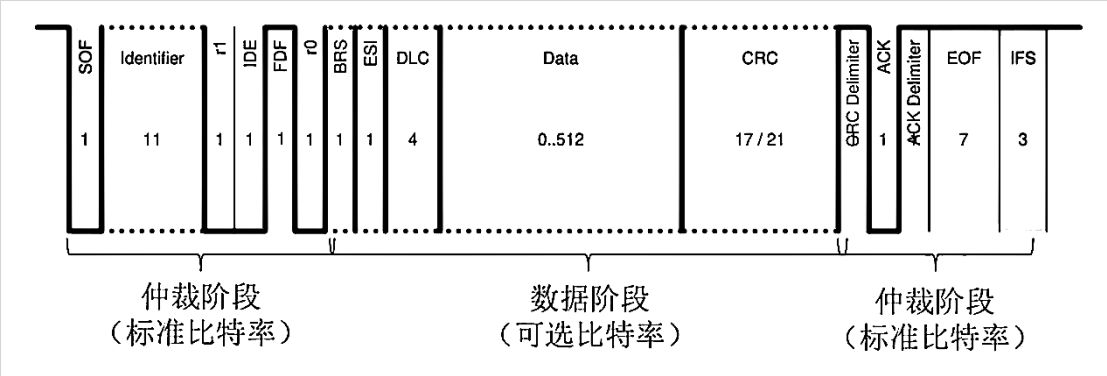
38.1.2 CAN-FD 帧格式

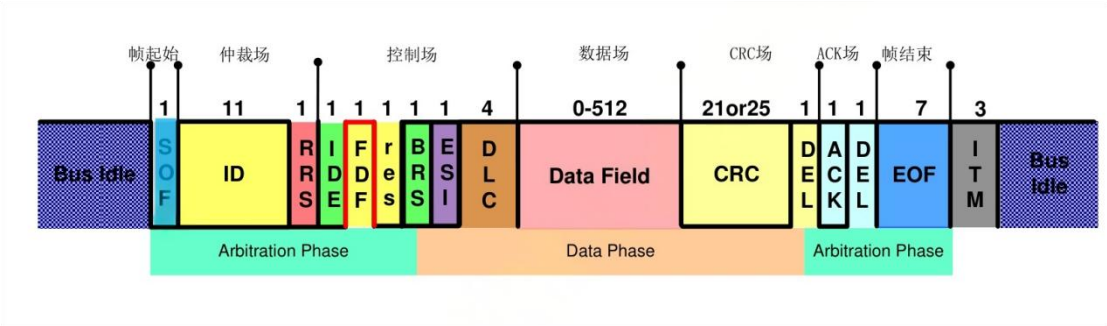
帧格式

主要由两个阶段组成——仲裁阶段和数据阶段

1）数据段开始于数据控制场，终止于 CRC 界定符；

2）数据段之外的部分都称为仲裁阶段



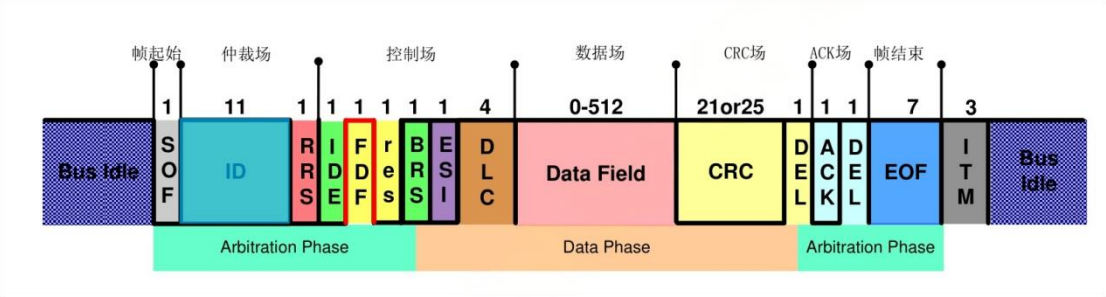


SOF：帧起始

标识一个数据帧的开始，用于同步

固定格式：一个显性位

只有在总线空闲期间，节点才能够发送 SOF



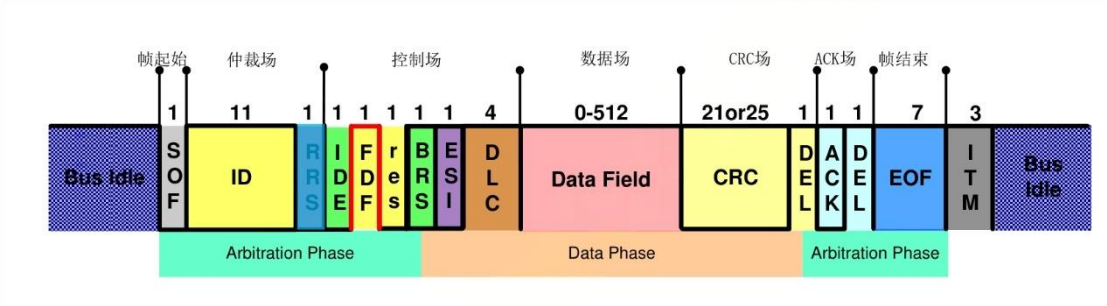
ID：标识符

唯一确定一条报文

表明报文的含义，可以包含报文的源地址和目标地址

确定报文的仲裁优先级，ID 数值越小，优先级越高

标准帧　11 位；扩展帧　29 位

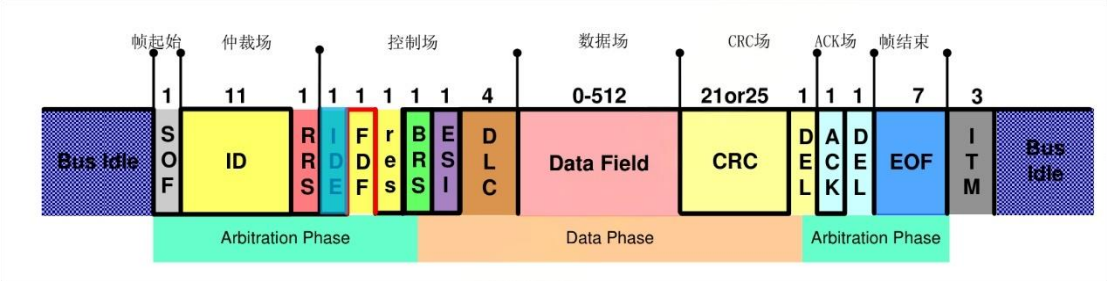


RRS

对应传统 CAN 报文 RTR 位

CAN FD 没有定义远程帧

发送节点发送显性位

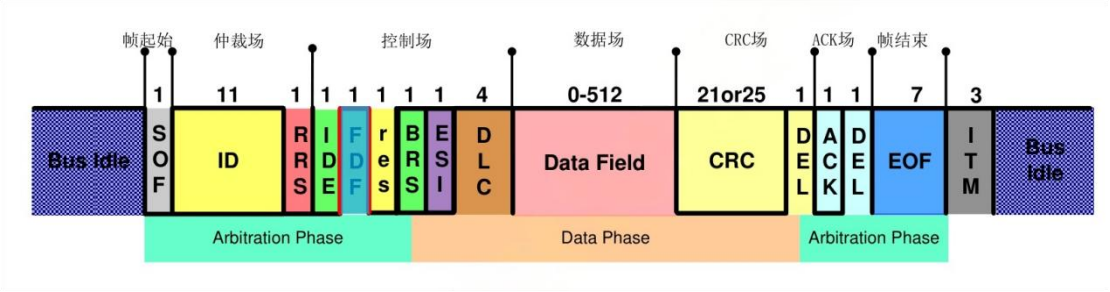


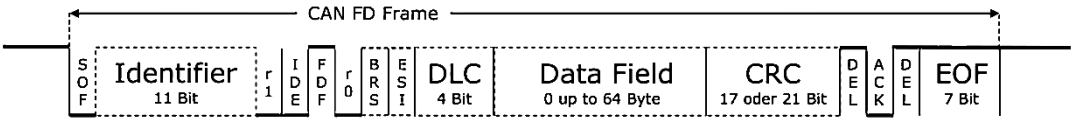
IDE

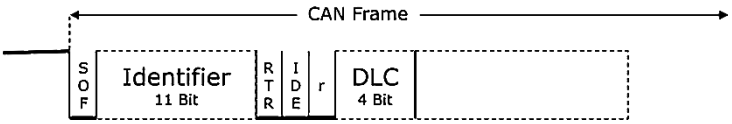
用于区分标准帧和扩展帧

标准帧，IDE=0（11 位 ID）

扩展帧，IDE=1（29 位 ID）







FDF =1：CAN-FD
FDF =0：标准CAN

FDF 位

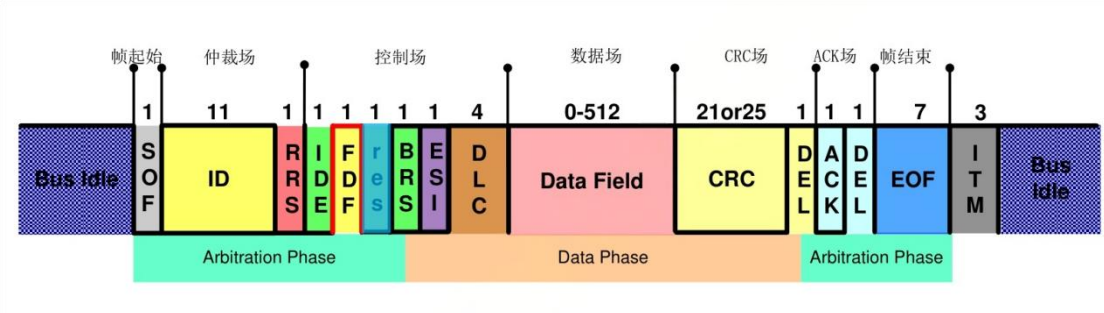
用于区分普通 CAN 报文和 CAN FD 报文

CAN 报文，FDF=0

CAN FD 报文，FDF=1

不用于仲裁

对于同一 ID 值只能分配给 Classic CAN 报文或者 FD 报文

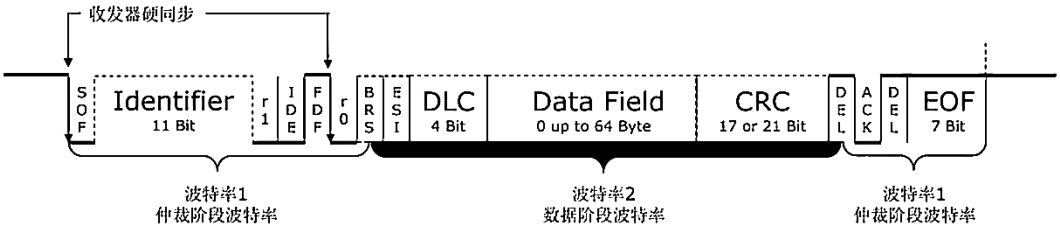


res

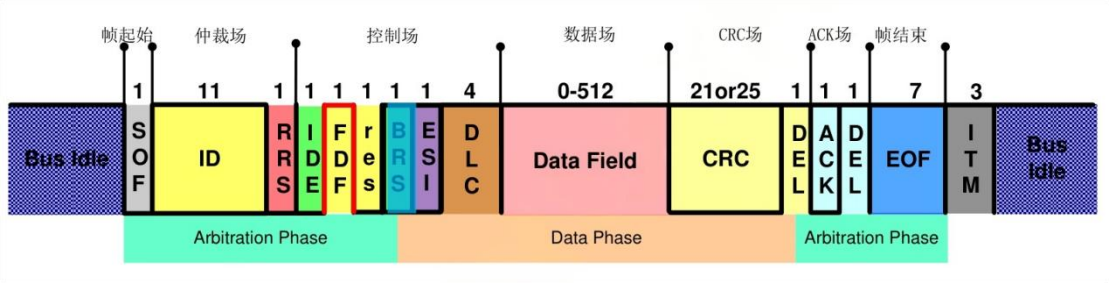
保留位

当前置 0

BRS 位



BRS =1：在数据阶段切换比特率
BRS =0：不切换比特率



BRS（Bit Rate Switch）

速率切换指示位

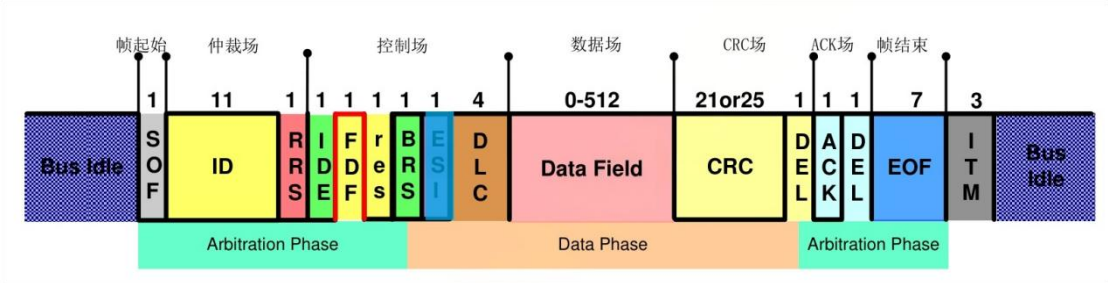
BRS=1，进行速率切换

BRS=0，不进行速率切换

ESI 位



ESI =0：节点处于错误活动状态
ESI =1：节点处于错误被动状态



ESI（Error State Indicator）

错误状态指示

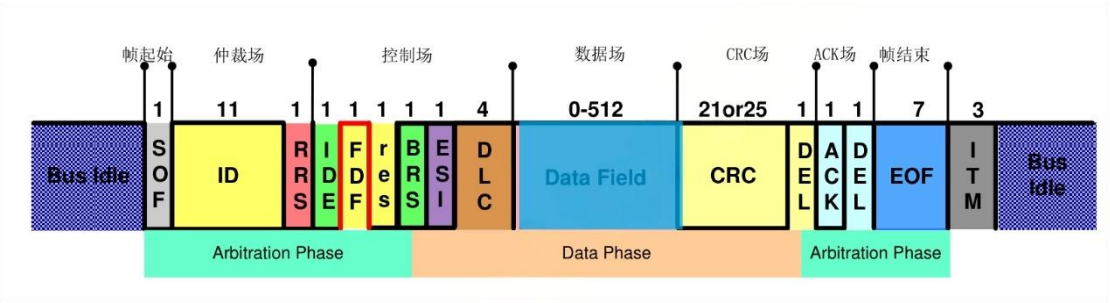
ESI=0，处于主动错误状态的节点

ESI=1，处于被动错误状态的节点

DLC 位



Frames	Data length code				Data bytes Num.
	DLC3	DLC2	DLC1	DLC0	
Classical Frames and FD Frames	0	0	0	0	0
	0	0	0	1	1
	0	0	1	0	2
	0	0	1	1	3
	0	1	0	0	4
	0	1	0	1	5
	0	1	1	0	6
	0	1	1	1	7
	1	0	0	0	8
Classical	1	0 or 1	0 or 1	0 or 1	8
FD Frames	1	0	0	1	12
	1	0	1	0	16
	1	0	1	1	20
	1	1	0	0	24
	1	1	0	1	32
	1	1	1	0	48
	1	1	1	1	64



数据场

具有 0-64 个字节长度，由 DLC 确定

包含 CAN 数据帧发送的内容

CRC 校验：

CRC Sequence

校验范围包括 Stuff Count 和 CRC 之前的填充位

长度为 17 位-DLC 小于等于 10（数据场小等于 16 个字节）

长度为 21 位-DLC 大于 10（数据场 20 到 64 个字节）



CAN: CRC_15 0xC599

$$(x^{15}+x^{14}+x^{10}+x^{8}+x^{7}+x^{4}+x^{3}+1) = (x+1)\cdot(x^{7}+x^{3}+1)\cdot(x^{7}+x^{3}+x^{2}+x+1)$$

CAN FD < 16 bytes of data: CRC_17 0x3685B

$$(x^{17}+x^{16}+x^{14}+x^{13}+x^{11}+x^{6}+x^{4}+x^{3}+x^{1}+1)$$
$$= (x+1)\cdot(x^{16}+x^{13}+x^{10}+x^{9}+x^{8}+x^{7}+x^{6}+x^{3}+1)$$

CAN FD > 16 bytes of data: CRC_21 0x302899

$$(x^{21}+x^{20}+x^{13}+x^{11}+x^{7}+x^{4}+x^{3}+1)$$
$$= (x+1)\cdot(x^{10}+x^{3}+1)\cdot(x^{10}+x^{3}+x^{2}+x^{1}+1)$$

位填充的区别：

CRC 填充机制

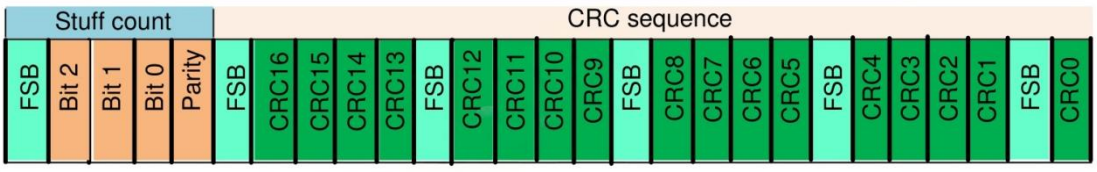
FSB（Fixed Stuff Bit）发送节点从 stuff count 前开始

插入以后每四位后填充一个极性相反的位

接收节点清除填充

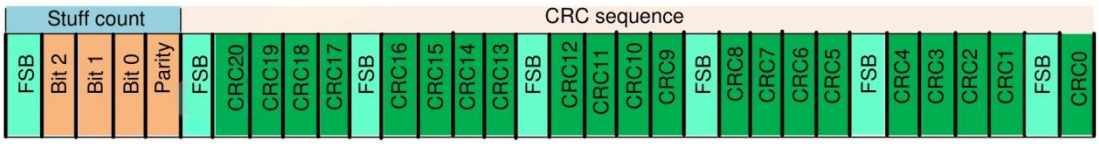
DLC 小于等于 10（数据场小等于 16 个字节）

17 CRC 位： CRC 场有 5 个填充位

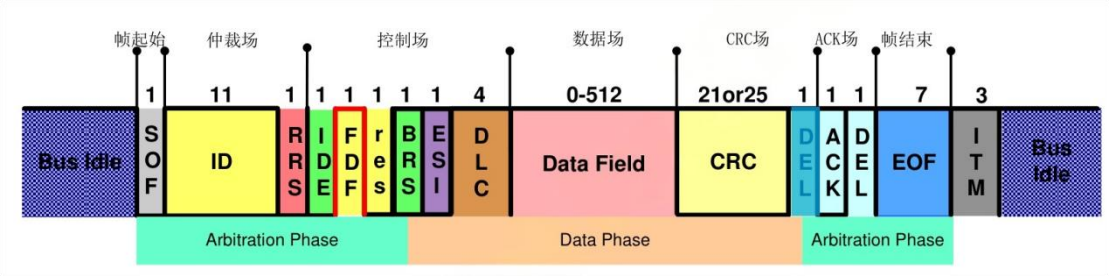


DLC 大于 10（数据场 20 到 64 个字节）

21 CRC 位： CRC 场有 6 个填充位



其他部分同标准 CAN 填充方式

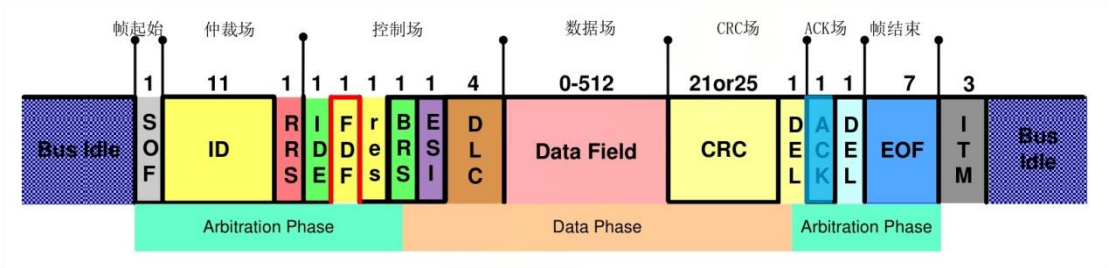


DEL：CRC 界定符

界定 CRC 序列

1~2 位（相位偏移引起）

在界定符的第一位进行速率切换



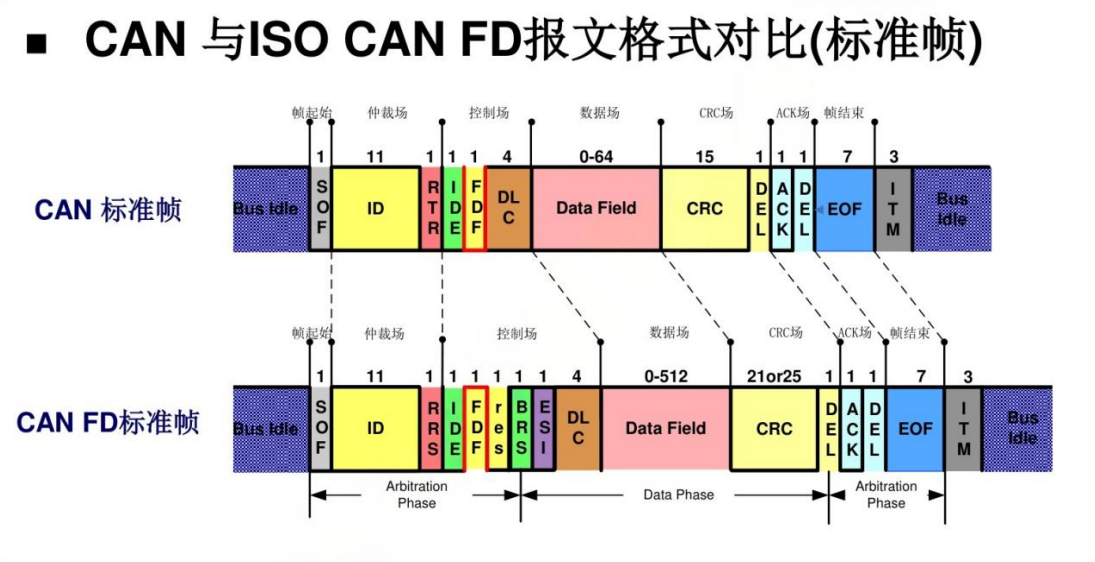
ACK

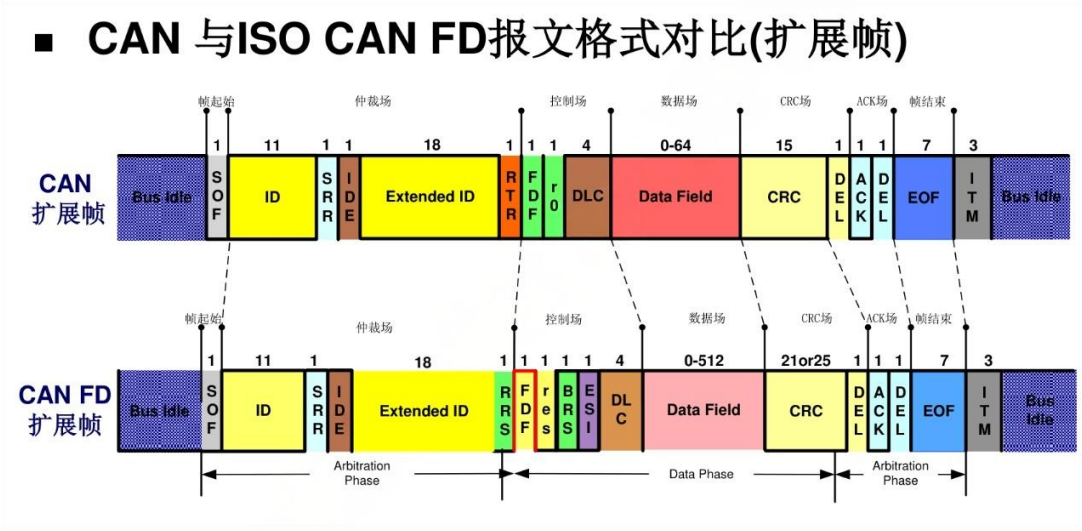
确定报文被至少一个节点正确接收

发送节点发送隐性位，接收节点正确接收后发送显性

CAN-FD 报文中所有节点最多可正确接收两位显性 ACK

38.1.3 CAN-FD 和标准 CAN 的区别





位时间上的区别

数据位时间与标称位时间一样由四个段组成：同步段，传播段和两个相位缓冲区段。 每个段由整数个数据时间段组成。

收发器延迟补偿

收发器环路延迟(Transceiver Loop Delay)

TxD 到 RxD 传输时间

不能超过 255ns（ISO11898-2:2016）规定

CAN 和 CAN-FD 共存

CAN-FD 补充 CAN 通信，它不替代现有的经典 CAN 帧格式。如果仅使用经典 CAN 帧，或者如果经典 CAN 收发器在 CAN-FD 格式的帧被发送时不激活，则经典 CAN 与 CAN-FD 将能够实现在相同网络中共存。

有 3 种 CAN 的实现形式：

1）FD enabled：发送和接收 CAN-FD 帧;

2）FD tolerant：不妨碍 CAN-FD 帧；

3）legacy CAN：只支持经典的 CAN。

38.2 CAN-FD 主要特性

- 支持 CAN 规范
  - 支持 CAN2.0A/CAN2.0B 规范（最大 8 字节有效载荷）
  - 支持 CAN-FD 规范（最高 64 字节有效载荷）
- 可配置的通信波特率
  - CAN2.0A/CAN2.0B 支持最高通信波特率可达 1Mbit/s
  - CAN-FD 支持最高通信波特率可达 8Mbit/s
- 可配置接收缓冲区大小
  - 通用参数用于选择缓冲区槽的数量。
  - 类似 FIFO 的行为
  - 收到的“不接受”或“错误”的消息不会覆盖已存在的内容存储的消息。
- 两个发送缓冲区
  - 一个主传输缓冲区（PTB）
  - 可配置次级发送缓冲区（STB）。STB 为可选参数。通用参数用于选择缓冲槽的数量。```
  - 以 FIFO 或优先级决策模式运行
- 1~16 组独立且可编程的内部 29 位接收过滤器
- 扩展功能
  - PTB 和 STB 均支持单次传输模式
  - 支持监听模式
  - 支持回环模式
  - 收发器待机模式
- 扩展状态和错误报告
  - 捕获最后一次发生的错误类型及仲裁丢失位置
  - 可编程误差警告限值
- 可配置中断源
- 时间戳
  - 支持 ISO 11898-4 时间触发 CAN
  - CiA 603 时间戳

38.3 CAN-FD 功能说明

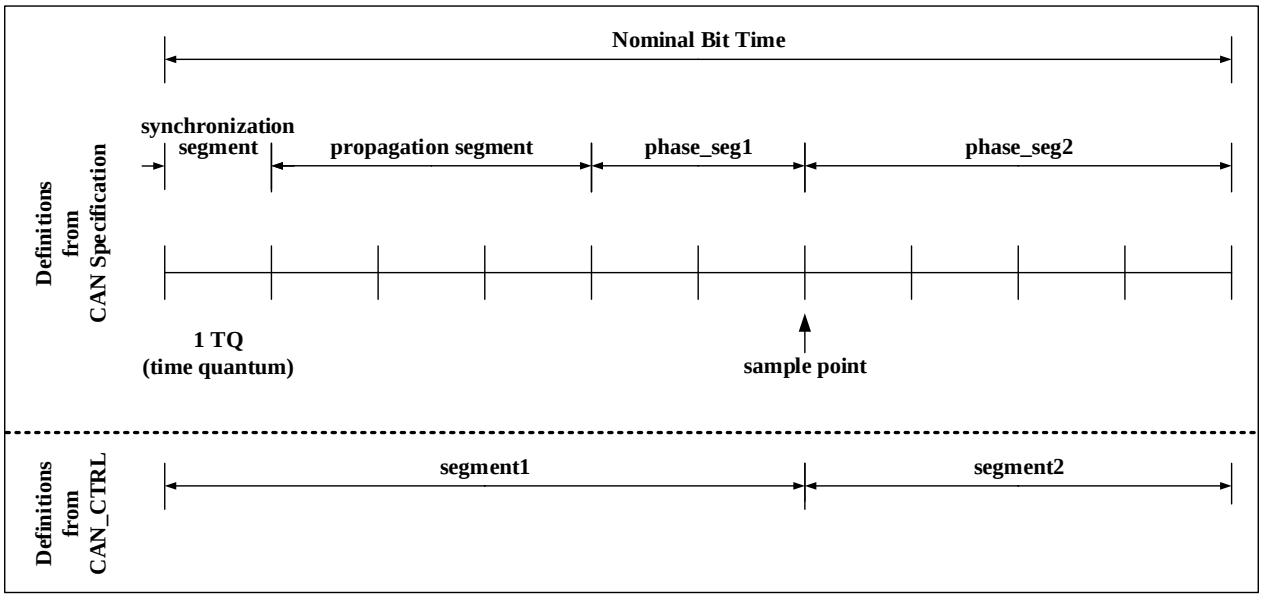
38.3.1 操作模式

CAN FD 控制器存在两个操作模式，复位模式（CAN_CFG_STAT.RESET=1）和正常模式（CAN_CFG_STAT.RESET=0）。模块初始化时，首先应该在复位模式中设定只能在复位模式下操作的寄存器（详见寄存器说明章节），然后退出复位模式，在正常模式中操作其余寄存器。

38.3.2 波特率设定

CAN 通信使用的时钟源为系统时钟。下图给出 CAN 位时间定义图，虚线上部分为 CAN 协议规定的位时间，虚线下部分为本 CAN 控制器（CAN-CTRL）定义的位时间。其中 segment1 和 segment2 可以通过寄存器 SBT 和 FBT 设定。SBT 寄存器和 FBT 寄存器只能在 CAN_CFG_STAT.RESET=1 即 CAN 软件复位时设定。SBT 寄存器用于 CAN2.0 和 CAN FD 的仲裁段，FBT 寄存器用于 CAN FD 数据段。

CAN 通信时，建议时钟源选取 20MHz/40MHz/80MHz。



CAN 位时间定义图

TQ 计算方法请参考以下公式，其中 PRESC 通过 SBT 寄存器的 S_PRESC 位或 FBT 寄存器的 F_PRESC 位设定。f_{can_clk} 为 CAN 通信时钟频率。

$$S_TQ = \frac{S_PRESC + 1}{f_{can_clk}}$$

$$F_TQ = \frac{F_PRESC + 1}{f_{can_clk}}$$

位时间计算方法请参考以下公式，其中 S_segment1 和 S_segment2 通过 SBT 寄存器的 S_SEG_1 位和 S_SEG_2 位设定，F_segment1 和 F_segment2 通过 FBT 寄存器的 F_SEG_1 位和 F_SEG_2 位设定。

$Slow_BT = t_{S_segment1} + t_{S_segment2} = ((S_SEG_1+2) + (S_SEG_2+1)) \times S_TQ$

$Fast_BT = t_{F_segment1} + t_{F_segment2} = ((F_SEG_1+2) + (F_SEG_2+1)) \times F_TQ$

CAN 位时间设定规则

位	设定范围	规则
SBT 寄存器的 S_SEG_1 位	[0..63] CAN2.0 bits (slow) [0..63] CAN FD nominal bits (slow)	SEG_1≥SEG_2+1 SEG_2≥SJW
SBT 寄存器的 S_SEG_2 位	[0....7] CAN2.0 bits (slow) [0..31] CAN FD nominal bits (slow)	
SBT 寄存器的 S_SJW 位	[0..15] CAN2.0 bits (slow) [0..15] CAN FD nominal bits (slow)	
FBT 寄存器的 F_SEG_1 位	[0..15] CAN FD data bits (fast)	SEG_1≥SEG_2 SEG_2≥SJW
FBT 寄存器的 F_SEG_2 位	[0…7] CAN FD data bits (fast)	
FBT 寄存器的 F_SJW 位	[0....7] CAN FD data bits (fast)	

以下给出 CAN FD 的波特率设定推荐，仅供参考。

PSP： Primary Sample Point

SSP： Secondary Sample Point

Seg 1： Segment 1

Seg 2： Segment 2

TDC： Transmitter Delay Compensation

20MHz 通信时钟时波特率设定建议

Bit Rate [Mbit/s]	PSP [%]	SSP [%]	Prescaler	Bit Time [TQ]	Seg 1 [TQ]	Seg 2 [TQ]	SJW [TQ]	TDC [CAN 通信时钟]
0.25 （仲裁）	80	-	1	80	64	16	16	-
0.5 （仲裁）	80	-	1	40	32	8	8	-
0.5	80	禁止	1	40	32	8	8	-
1	80	80	1	20	16	4	4	16
2	80	80	1	10	8	2	2	8
4	80	80	1	5	4	1	1	4
5	75	75	1	4	3	1	1	3

40MHz 通信时钟时波特率设定建议

Bit Rate [Mbit/s]	PSP [%]	SSP [%]	Prescaler	Bit Time [TQ]	Seg 1 [TQ]	Seg 2 [TQ]	SJW [TQ]	TDC [CAN 通信时钟]
0.25 （仲裁）	80	-	2	80	64	16	16	-
0.5 （仲裁）	80	-	1	80	64	16	16	-
0.5	80	禁止	2	40	32	8	8	-
1	80	80	1	40	32	8	8	32
2	80	80	1	20	16	4	4	16
4	80	80	1	10	8	2	2	8
5	75	75	1	8	6	2	2	6
8	80	80	1	5	4	1	1	4

80MHz 通信时钟时波特率设定建议

Bit Rate [Mbit/s]	PSP [%]	SSP [%]	Prescaler	Bit Time [TQ]	Seg 1 [TQ]	Seg 2 [TQ]	SJW [TQ]	TDC [CAN 通信时钟]
0.25 （仲裁）	80	-	4	80	64	16	16	-
0.5 （仲裁）	80	-	2	80	64	16	16	-
0.5	80	禁止	4	40	32	8	8	-
1	80	80	2	40	32	8	8	64
2	80	80	2	20	16	4	4	32
4	80	80	1	20	16	4	4	16
5	75	75	1	16	12	4	4	12
8	80	80	1	10	8	2	2	8

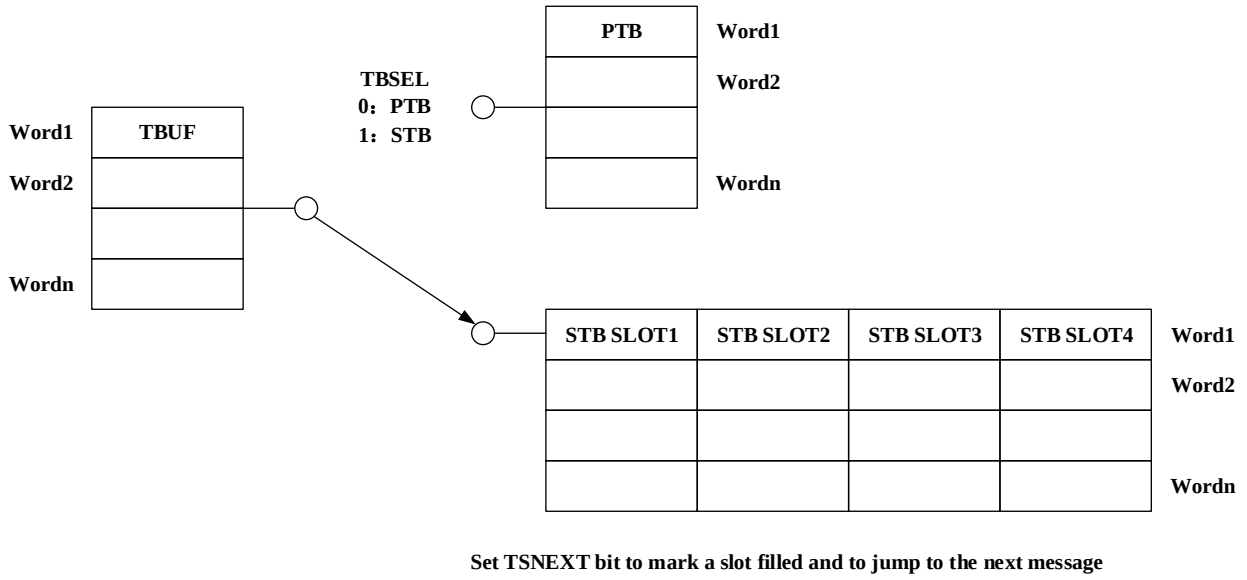
38.3.3 发送缓冲器

CAN_CTRL 提供两种发送缓冲器用于发送数据，主发送数据缓冲器 PTB 和副发送缓冲器 STB。PTB 具有最高的优先级，但只能缓冲一帧数据。STB 优先级比 PTB 低，但可以缓冲 4 帧数据，且 STB 内 4 帧数据可以工作在 FIFO 模式或者优先级仲裁模式。

STB 中的 4 帧数据可以通过 TCMD 寄存器的 TSALL 位设定为 1 全部发送，最先写入的数据先发送。

PTB 中的数据具有最高优先级，所以 PTB 发送能推迟 STB 发送，但是已经赢得仲裁并开始发送的 STB 不能够被 PTB 发送推迟。

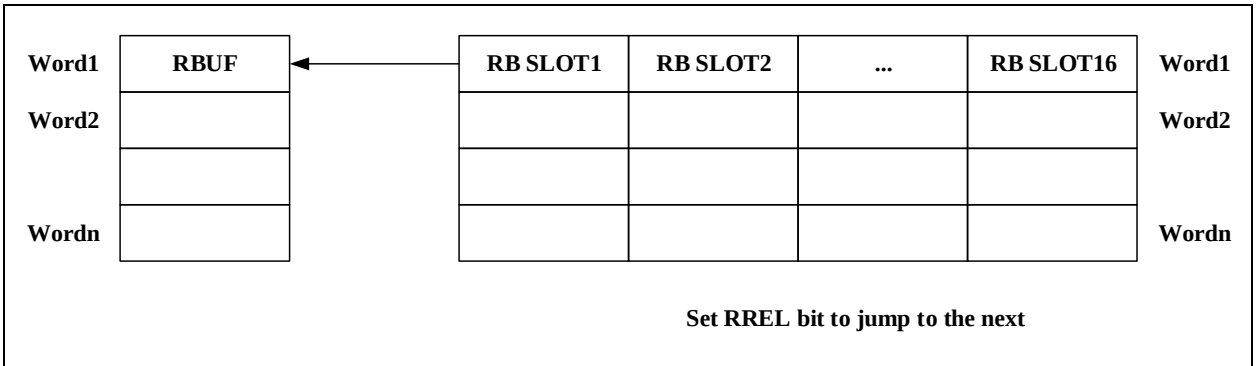
PTB 和 STB 可以通过 TBUF 寄存器进行访问。通过 TCMD 寄存器的 TBSEL 位选择 PTB 或者 STB，TBSEL=0，选择 PTB，TBSEL=1，选择 STB。通过 TCTRL 寄存器的 TSNEXT 位选择 STB 中的下一个 SLOT。对应关系如下图所示：



CAN FD TBUF 寄存器写发送缓冲器示意图

38.3.4 接收缓冲器

CAN_CTRL 提供 16 个 SLOT 的接收缓冲器用于存储接收到的数据，该 16 SLOT 的接收缓冲器工作在 FIFO 模式。RB SLOT 通过 RBUF 寄存器来读取接收到的数据，总是最先读取最早接收到的数据，并通过 RCTRL 寄存器的 RREL 设置为 1 释放已经读取的 RB SLOT，并指向下一个 RB SLOT。通过 RBUF 读取 RB SLOT 示意图如下。



CAN FD RBUF 寄存器读接收缓冲器示意图

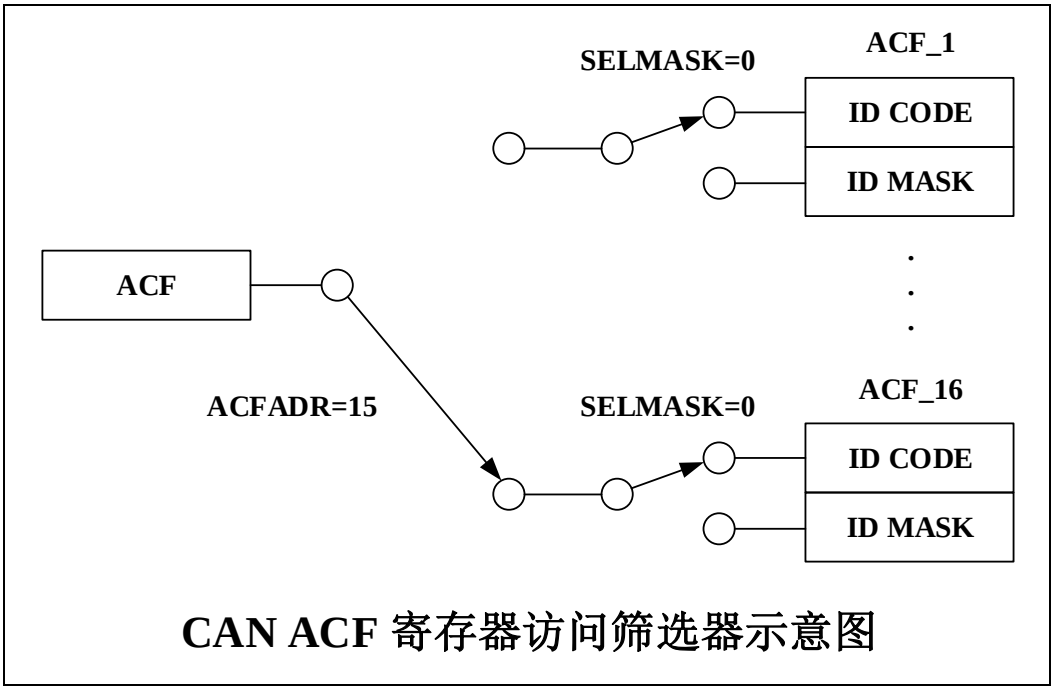
38.3.5 接收筛选寄存器组

CAN_CTRL 提供 16 组 32 位筛选器用于过滤接收到的数据从而降低 CPU 负荷，筛选器可以支持标准格式 11 位 ID 或者扩展格式 29 位 ID。每组筛选器有一个 32 位 ID CODE 寄存器和一个 32 位 ID MASK 寄存器，ID CODE 寄存器用于比较接收到 CAN ID，而 ID MASK 寄存器用于选择比较的 CAN ID 位。对应的 ID MASK 位为 1 时，不比较该位的 ID CODE。

接收到的数据只要通过 16 组筛选器的任意一组，则被接收，接收到的数据存储在 RB 中，否则数

据不被接收，也不被存储。

每组筛选器通过 ACFEN 寄存器使能或者禁止。ID CODE 和 ID MASK 通过 ACFCTRL 寄存器的 SELMASK 位设定，SELMASK=0 时，指向 ID CODE，SELMASK=0 时，指向 ID MASK。筛选器通过 ACFCTRL 寄存器的 ACFADR 位选择。ID CODE 和 ID MASK 通过 ACF 寄存器访问且只能在 CFG_STAT.RESET=1 即 CAN 复位模式时设定。ACF 寄存访问筛选寄存器组的方式请参考下图。



CAN ACF 寄存器访问筛选器示意图

38.3.6 数据发送

在开始发送前必须保证 PTB 或者 STB 待发送的数据已被填充完毕，然后再启动 PTB 或者 STB 发送。发送过程中不允许再次填充数据。

发送数据设定步骤如下：

1. 设定 TBSEL 从 PTB 和 STB 中选择发送 BUF。
2. 通过 TBUF 寄存器写需要发送的数据。
3. 如果选择的是 STB，设置 TSNEXT=1 以完成全部 STB SLOT 的装载。
4. 发送使能
   - PTB 发送使用 TPE
   - STB 发送使用 TSALL 或者 TSONE
5. 发送完成状态确认
   - PTB 发送完成使用 TPIF，TPIE 用于使能 TPIF
   - STB 采用 TSONE 发送完成时使用 TSIF，TSIE 用于使能 TSIF
   - STB 采用 TSALL 发送完成时使用 TSIF，此时需要设定的全部 STB SLOT 数据发送完成后，TSIF 才置位，TSIE 用于使能 TSIF

38.3.7 单次数据发送

不需要使用自动重新发送功能时，可以通过寄存器设定为单次发送模式，CFG_STAT 寄存器的 TPSS 位用于 PTB 的单次发送模式设定，TSSS 位用于 STB 的单次发送模式设定。数据成功发送时单次发送和正常发送模式时动作相同。但是数据没有成功发送时会出现以下结果：

- TPIF 置位（TPIE=1），对应的 BUF SLOT 数据会被清除。
- 有错误发送时，KOER 更新，BEIF 置位（BEIE=1）。
- 仲裁失败，ALIF 置位（ALIE=1）。

单次发送模式，不能单独依靠 TPIF 来判断发送完成，需要同 BEIF 和 ALIF 一起判断发送是否完成。

38.3.8 取消数据发送

可以通过 TPA 或者 TSA 取消已请求但还没有被执行的数据发送。取消数据发送会出现以下几种情况：

- 仲裁中
  - ✓ 节点仲裁失败，则取消数据发送。
  - ✓ 节点仲裁成功，则继续发送。
- 数据发送中
  - ✓ 成功发送数据且收到 ACK，对应的标志和状态正常置位。数据发送不取消。
  - ✓ 成功发送数据但没有收到 ACK，数据发送取消，错误计数器增加。
  - ✓ TSALL=1 设定的发送数据，正在发送的 STB SLOT 数据正常发送，没有开始发送的 STB SLOT 被取消。

取消数据发送的结果有以下两种情况。

- TPA 释放 PTB，且使 TPE=0。
- TSA 释放一个 STB SLOT 或者全部 STB SLOT 取决是 TSONE 还是 TSALL 使能的发送。

38.3.9 数据接收

接收筛选器组可以过滤掉不需要的接收数据，减少中断的发生和 RB 的读取，从而降低 CPU 负荷。接收数据设定步骤如下：

1. 设定筛选器组。
2. 设定 RFIE，RAFIE 和 AFWL。
3. 等待 RFIF 或者 RAFIF。
4. 通过 RBUF 从 RB FIFO 中读取最早接收到的数据。
5. 设置 RREL=1，选择下一个 RB SLOT。
6. 重复 4，5 直到通过 RSTAT 确认 RB 为空。

38.3.10 错误处理

CAN_CTRL 一方面可以自动处理部分错误，比如自动重发数据或者丢弃接收到含有错误的帧，另一方面通过中断将错误向 CPU 报告。

CAN 节点有以下三种错误状态：

- 主动错误：节点可正常通信，检测到错误时发送主动错误标志（6 个连续显性位），以破坏当前传输的报文。
- 被动错误：节点仍可通信，但检测到错误时仅发送被动错误标志（6 个连续隐性位），且发送后需等待额外时间才能重试。
- 节点关闭：节点无法收发任何报文，完全断开与总线的连接，需满足特定条件（如检测到 128 次 11 个连续隐性位）才能恢复。关闭状态下此节点不再影响整个 CAN 网络。

CAN_CTRL 提供 TECNT 和 RECNT 两个计数器用于计数错误。TECNT 和 RECNT 计数器按照 CAN 协议规定的规则进行增减。另外提供可编程的 CAN 错误警告 LIMIT 寄存器用于产生错误中断通知 CPU。

CAN 通信过程中有以下 5 种错误类型，错误类型可以通过 EALCAP 寄存器的 KOER 位识别。

- 位错误
- 形式错误
- 填充错误
- 应答错误
- CRC 错误

38.3.11 节点关闭

当发送错误数大于 255 时，CAN 节点自动进入节点关闭状态从而不参与 CAN 通信，直到返回到主动错误状态。可以通过 CFG_STAT 寄存器的 BUSOFF 位确认 CAN 节点关闭状态。BUSOFF 被置位的同时 EIF 中断产生。

CAN 从节点关闭状态恢复到主动错误状态有以下两种方法：

- MCU 复位
- 接收到连续 128 个 11 位的隐形位序列（恢复序列）

节点关闭状态下，TECNT 值保持不变，RECNT 用于计数恢复序列。从节点关闭状态恢复后，TECNT 和 RECNT 被复位为 0。

38.3.12 仲裁失败位置捕捉

CAN_CTRL 能够精确捕捉到仲裁失败位的位置并反映到 ALC 寄存器中。ALC 寄存器中保存着最近一次仲裁失败位的位置，如果节点赢得仲裁，则 ALC 位不更新。

ALC 值定义如下：

SOF 位后，第一个 ID 数据位 ALC 为 0，第二个 ID 数据位 ALC 为 1，依次类推。因为仲裁只发生在仲裁场内，所以 ALC 的最大值为 31。比如一个标准格式远程帧和一个扩展帧仲裁，扩展帧在 IDE 位失败，则 ALC=12。

38.3.13 回环模式

CAN_CTRL 支持以下两种回环模式：

- 内部回环
- 外部回环

两种回环模式都可以接收自己发出的数据帧，主要用于测试用途。

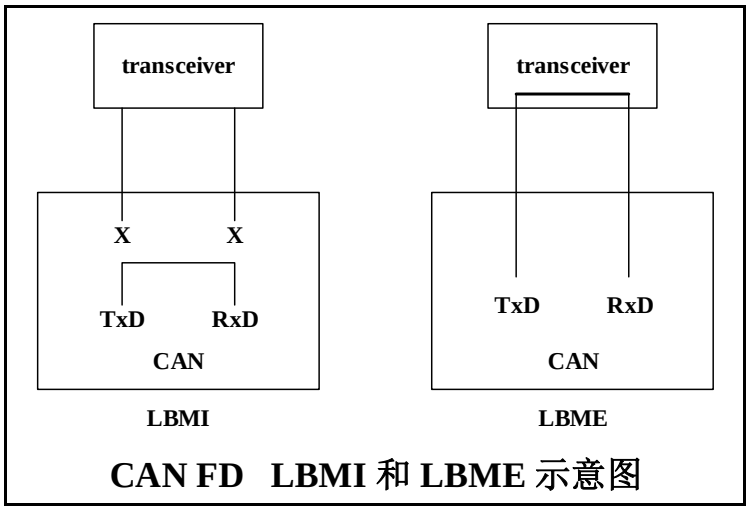
内部回环模式，模块内部将接收数据线连接到发送数据线，并且发送数据不输出。内部回环模式下，节点会生成自应答信号以避免 ACK 错误。

外部回环模式保持和收发器的连接因此发送的数据仍能出现在 CAN 总线上，在收发器的帮助下，CAN 能收到自己发送的数据。外部回环模式可以通过 RCTRL 寄存器的 SACK 位来决定是否生成自应答信号，SACK=0 时，不生成自应答信号，SACK=1 时，生成自应答信号。

外部回环模式，SACK=0 时，会出现以下两种情况：

- ✓ 其它节点也收到本节点发送的数据帧并发送应答信号，该情况下本节点能够成功收发数据。
- ✓ 如果没有其它节点返回应答信号，则会产生应答错误，会重新发送数据并增加错误计数器。此时推荐采用单次发送模式。

从回环模式返回到正常模式时，除了清除模式位以外，还需要软件复位 CAN_CTRL。



CAN FD LBMI 和 LBME 示意图

38.3.14 静默模式

静默模式可以用来监控 CAN 网络数据。在静默模式下，可以从 CAN 总线接收数据，不向总线发送任何数据。将 TCMD 寄存器中的 LOM 置 1，使 CAN 总线控制器进入静默模式，将其清 0 可以离开静默模式。

外部回环模式可以和静默模式组合成外部回环静默模式，此时 CAN 可以认为一个安静的接收者，但在有必要的时候可以发送数据。外部回环静默模式下，帧包含自应答信号允许被发送，但是该节点不会产生错误标志和过载帧。

38.3.15 软件复位功能

通过设定寄存器 CFG_STAT 寄存器的 RESET 位为 1，实现软件复位功能，软件复位功能的复位范围如下表所示。

软件复位范围表

寄存器位名	软件复位	备注	寄存器位名	软件复位	备注
ACFADR	否	-	F_SEG_1	是	只能在软件复位时可写
ACODE	否	只能在软件复位时可写	F_SEG_2	是	只能在软件复位时可写
AE_x	否	-	F_SJW	是	只能在软件复位时可写
AFWL	否	-	KOER	是	-
AIF	是	-	LBME	是	-
ALC	是	-	LBMI	是	-
ALIE	否	-	RACTIVE	是	接收立即停止，并不生成 ACK
ALIF	是	-	RAFIE	否	-
AMASK	否	只能在软件复位时可写	RAFIF	是	-
BEIE	否	-	RBALL	是	-
BEIF	是	-	RBUF	是	RB 被标记为空，数值不定
BUSOFF	否	通过写 1 清除	RECNT	否	通过 BUSOFF 写 1 清零
EIE	否	-	REF_ID	否	-

寄存器位名	软件复位	备注	寄存器位名	软件复位	备注
EIF	否	-	REF_IDE	否	-
EPASS	否	-	RFIE	否	-
EPIE	否	-	RFIF	是	-
EPIF	是	-	RIE	否	-
EWARN	否	-	RIF	是	-
EWL	是	-	ROIE	否	-
FD_ISO	否	只能在软件复位时可写	ROIF	是	-
F_FRESC	否	只能在软件复位时可写	ROM	否	
ROV	是	-	TSMODE	否	
RREL	是	-	TSNEXT	是	-
RSTAT	是	TSONE	是	-	
SACK	是	-	TPIE	否	-
SELMASK	否	-	TPIF	是	-
S_PRESC	否	只能在软件复位时可以写	TPSS	是	-
S_SEG_1	否	只能在软件复位时可以写	TSFF	是	所有 STB SLOT 被标记为空
S_SEG_2	否	只能在软件复位时可以写	TSIE	否	-
S_SJW	否	只能在软件复位时可以写	TSIF	是	-
SSPOFF	是	-	TSSS	是	-
TACTIVE	是	发送立即停止	TSSTAT	是	所有 STB SLOT 被标记为空
TBE	是	-	TTEN	是	-
TBF	否	-	TTIF	是	-
TBPTR	否	-	TTIE	否	-
TBSEL	是	-	TTPTR	否	-
TBUF	是	STB 被标记为空，指向 PTB	TTTBM	否	-
TDCEN	是	-	TTYPE	否	-
TECNT	否	可通过 BUSOFF=1 清除	TT_TRIG	否	-
TEIF	是	-	TT_WTRIG	否	-
TPA	是	-	T_PRESC	否	-
TPE	是	-	WTIE	否	-
TSA	是	-	WTIF	是	
TSALL	是	-			

38.3.16 向上兼容 CAN-FD 功能

CAN-CTRL 在 CAN FD 功能禁止时即使在包含 CAN FD 网络中收到 CAN FD 的帧，接收器会自动忽略这些帧，不返回 ACK，等到总线空闲时，再发送或者接收下一个 CAN2.0B 的帧。

38.3.17 时间触发 TTCAN

CAN-CTRL 为 ISO11898-4 规定的时间触发通信方式提供部分(lever 1)硬件支持。本章节从以下 5 个部分介绍 TTCAN 功能。

38.3.17.1 TTCAN 模式下的 TBUF 行为

TTTBM=1

TTTBM=1 时，PTB 和 STB SLOT 一样组成 TB SLOT，通过 TBPTR 寄存器指定发送 BUF，其中 TBPTR=0 时，指向 PTB，TBPTR=1 是指向 STB SLOT1，依次类推。主机可以通过 TBE 和 TBF 寄存器来标记发送 BUF SLOT。此时 TBSEL 和 TSNEXT 寄存器无任何意义从而可以被忽略。

TTTBM=1 时，PTB 不具有任何特殊的属性，和 STB SLOT 一样，传送完成标志也采用 TSIF。TTCAN 模式时，发送 BUF 没有 FIFO 模式和优先级仲裁模式，同时也只有一个选定的 SLOT 可以发送数据。

TTCAN 模式下，传输开始需要采用时间触发方式，TPE，TSONE，TSALL，TPSS 和 TPA 被固定为 0 且被忽略。

TTTBM=0

TTTBM=0 时，组合使用事件驱动通信和接收时间戳功能。在该模式下，PTB 和 STB 的功能和 TTEN=0 时一致，因此 PTB 始终具有最高的优先级，而 STB 可以工作在 FIFO 模式或者优先级仲裁模式。

38.3.17.2 TTCAN 功能描述

上电后，Time Master 需要根据 ISO 11898-4 协议进行初始化。一个 CAN 网络中，最多可以有 8 个潜在的 Time Master。每一个 Time Master 都具有自己的参考消息 ID（ID 最低 3 位）。这些潜在的 Time Master 根据自己的优先级发送各自的参考消息。

TTEN=1 后，16 位的计数器开始工作，当参考消息被成功接收或者 Time Master 成功发送参考消息时，CAN 控制器将 Sync_Mark 拷贝给 Ref_Mark，Ref_Mark 将 cycle time 设置为 0。成功接收参考消息置位 RIF 标志而成功发送参考消息置位 TPIF 标志或者 TSIF 标志。此时主机需要准备下一个动作的触发条件。

触发条件可以是接收触发。该触发仅触发中断可用于检测期待的消息没有被收到。

触发条件也可以是发送触发。该触发开始发送通过 TTPTR 寄存器指定的 TBUF SLOT 里的数据。如果选定的 TBUF SLOT 被标记为空，则不开始发送，但置位中断标志。

38.3.17.3 TTCAN 时序

CAN_CTRL 支持 ISO11898-4 level 1。包含的一个 16 位计数器工作在 S_PRESC，S_SEG_1，S_SEG_2 定义的位时间下。如果 TTEN=1，则有一个额外的预分频器 T_PRESC。

一帧数据的 SOF 时，计数器的值为 Sync_Mark。如果该帧数据为参考消息，则将 Sync_Mark 拷贝给 Ref_Mark。cycle time 等于计数器的值减去 Ref_Mark。该时间用作为接收消息的时间戳或者发送消息的触发时间基准。

38.3.17.4 TTCAN 触发方式

通过 TTYPE 寄存器定义 TTCAN 的触发方式，TTPTR 寄存器指定发送 SLOT，而 TT_TRIG 指定触发器的 cycle time。

包含以下五种触发方式：

- 立即触发
- 时间触发
- 单次发送触发
- 发送开始触发
- 发送停止触发

除了立即触发方式外，所有的触发器都使用 TTIF 标志。TTTBM=1 时，只支持时间触发方式。

立即触发

通过写 TT_TRIG 的高位（不在意写入的值），启动触发器。此模式下，TTPTR 选定的 TBUF SLOT 内的数据会立即发送。TTIF 不置位。

时间触发

时间触发方式仅通过置位 TTIF 标志产生中断，并无其他功能。如果一个节点期待在特定的时间窗口内收到期待的数据，则可以使用时间触发方式。如果 TT_TRIG 值小于实际的 cycle time，则 TEIF 置位且无其它动作。

单次发送触发

单次发送触发方式用于在执行时间窗口内发送数据。此时，忽略 TSSS 位。

通过 TEW 位设定 ISO11898-4 规定的最多 16 个 cycle time 的 Tick，设定范围为 1~16。如果在规定的发送使能时间窗口内数据没有开始发送，则帧被丢弃。对应的发送 BUF SLOT 被标记为空，并且置位 AIF，对应的发送 BUF 内的数据不会被改写，因此可以通过置位 TPF 再次发送。

如果 TT_TRIG 值小于实际的 cycle time，则 TEIF 置位且无其它动作。

发送开始触发

发送开始触发方式用于仲裁时间窗口内，参与仲裁。TSSS 用于决定是否自动重发或者单次发送模式。如果 TTPTR 寄存器指定的消息没有被成功发送，可以使用发送停止触发来停止该发送。

如果 TT_TRIG 值小于实际的 cycle time，则 TEIF 置位且无其它动作。

发送停止触发

发送停止触发方式用于停止通过发送开始触发方式已经开始的发送。如果发送被停止，则发送帧被舍弃，置位 AIF 并将选定的 TBUF SLOT 标记为空，但 TBUF SLOT 内的数据不会被改写，可以通过置位 TPF 再次发送。

如果 TT_TRIG 值小于实际的 cycle time 则 TEIF 置位且执行停止。

38.3.17.5 TTCAN 触发看门时间

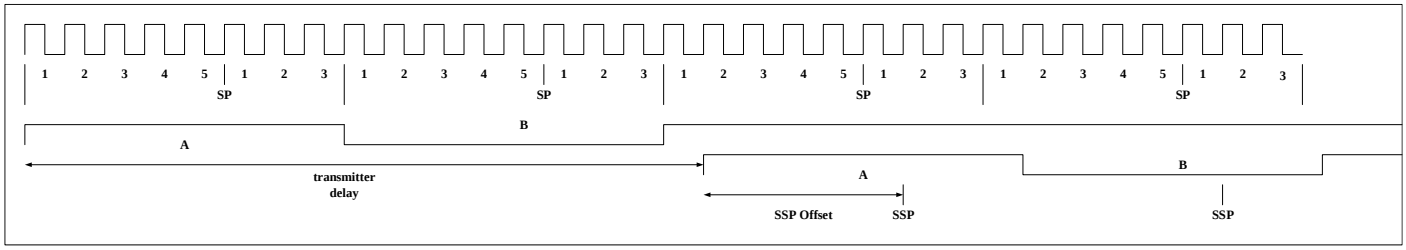
TTCAN 触发看门时间功能类似于看门狗功能，在 TTTBM=1 时使用。用来看门从上次成功接收到参考消息开始的时间。参考消息可以在周期 cycle time 中或者一个事件后被接收，应用程序应该根据具体情况设定合适的看门时间。

如果 cycle count 等于 TT_WTRIG，则置位 WTIF。通过 WTIE 写 0，关闭看门触发。

如果 TT_WTRIG 比实际的 cycle time 小，则 TEIF 置位。

38.3.18 TDC 和 RDC

CAN FD 通信时，数据的延迟可能超过一位时间，因此需要补偿。TDC（Transmitter Delay Compensation）和 RDC(Receiver Delay Compensation)用于数据延迟的补偿。其中 TDC 需要软件控制开关，而 RDC 不需要，自动有效。TDC 中采用辅助采样点 SSP（Secondary Sample Point）的方式去补偿数据延迟。如下图：



TDC功能示意图

软件使能 TDC 功能时，本控制器可以自动确定 transmitter delay，通过设定寄存器 TDC.SSPOFF 来设定 SSP Offset。建议 SSP Offset 设定值和 $t_{F_segment1}$ 设定值相同。

38.3.19 中断

CAN 中断表

中断标志	描述
RIF	接收中断
ROIF	接收上溢中断
RFIF	接收 BUF 满中断
RAFIF	接收 BUF 将满中断
TPIF	PTB 发送中断
TSIF	STB 发送中断
EIF	错误中断
AIF	取消发送中断
EPIE	错误被动中断
ALIF	仲裁失败中断
BEIF	总线错误中断
WTIF	触发看门中断
TEIF	触发错误中断
TTIF	时间触发中断

38.4 使用 CAN-FD / CAN 时，CPU 如何跑 100MHz ~ 120MHz

使用外部晶振提高主频的精度，加大 SJW 设置增加波特率的误差兼容，就可以在 100MHz ~ 120MHz 进行 CAN-FD 通讯。

具体参考例子：STC32G144K246 要注意啥，使用 USB@120MHz，使用 CAN-FD@100MHz ~ 120MHz
https://www.stcaimcu.com/thread-22801-1-1.html

38.5 CAN1 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW1	A2H	S1_S[1:0]		**CAN_S[1:0]**		SPI_S[1:0]		LIN_S[1:0]	
P_SWX1	7EFD69H	S4_S[1]	S3_S[1]	S2_S[1]	-	CAN2_S[2]	**CAN_S[2]**	-	SWP54P12

CAN_S[2:0]：CAN1 功能脚选择位

CAN_S[2:0]	CANRx	CANTx	CANSTB
000	P0.0	P0.1	P0.6
001	P5.0	P5.2	P2.2
010	P4.2	P4.5	P5.1
011	P7.0	P7.1	P4.5
100	P3.0	P3.1	P3.3
101	P3.6	P3.7	P1.2
110	P1.4	P1.5	P3.3
111	P4.3	P4.4	P7.4

38.6 CAN2 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW3	BBH	I2S_S[1:0]		S2SPI_S[1:0]		S1SPI_S[1:0]		**CAN2_S[1:0]**	
P_SWX1	7EFD69H	S4_S[1]	S3_S[1]	S2_S[1]	-	**CAN2_S[2]**	CAN_S[2]	-	SWP54P12

CAN2_S[2:0]：CAN2 功能脚选择位

CAN2_S[2:0]	CAN2Rx	CAN2Tx	CAN2STB
000	P0.2	P0.3	P0.7
001	P5.2	P5.3	P2.3
010	P4.6	P4.7	P5.3
011	P7.2	P7.3	P4.7
100	P1.0	P1.1	P0.7
101	P8.4	P8.5	P1.3
110	PA.0	PA.1	P2.3
111	-	-	-

38.7 CAN-FD 相关的特殊功能寄存器

38.7.1 CAN 接收缓冲区寄存器（CAN_RBUFn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_RBUF00	7EF400H	CAN_RBUF00[7:0]							
CAN_RBUF01	7EF401H	CAN_RBUF01[7:0]							
…	…	…							
CAN_RBUF79	7EF44FH	CAN_RBUF79[7:0]							

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN2_RBUF00	7EF300H	CAN2_RBUF00[7:0]							
CAN2_RBUF01	7EF301H	CAN2_RBUF01[7:0]							
…	…	…							
CAN2_RBUF79	7EF34FH	CAN2_RBUF79[7:0]							

CAN_RBUFn[7:0]：接收到的 CAN 邮箱数据

标准帧的接收数据格式如下：

地址	B7	B6	B5	B4	B3	B2	B1	B0	功能
CAN_RBUF00	ID[7:0]								ID
CAN_RBUF01	-	-	-	-	-	ID[10:8]			ID
CAN_RBUF02	-	-	-	-	-	-	-	-	ID
CAN_RBUF03	-	-	-	-	-	-	-	-	ID
CAN_RBUF04	IDE=0	RTR	FDF	BRS	DLC[3:0]				Control
CAN_RBUF05	KOER[2:0]			TX	-	-	-	-	Status
CAN_RBUF06	CYCLE_TIME[7:0]								TTCAN
CAN_RBUF07	CYCLE_TIME[15:8]								TTCAN
CAN_RBUF08	DATA1								Data
CAN_RBUF09	DATA2								Data
…	…								…
CAN_RBUF71	DATA64								Data

扩展帧的接收数据格式如下：

地址	B7	B6	B5	B4	B3	B2	B1	B0	功能
CAN_RBUF00	ID[7:0]								ID
CAN_RBUF01	ID[15:8]								ID
CAN_RBUF02	ID[23:16								ID
CAN_RBUF03	-	-	-	ID[28:24]					ID
CAN_RBUF04	IDE=1	RTR	FDF	BRS	DLC[3:0]				Control
CAN_RBUF05	KOER[2:0]			TX	-	-	-	-	Status
CAN_RBUF06	CYCLE_TIME[7:0]								TTCAN
CAN_RBUF07	CYCLE_TIME[15:8]								TTCAN
CAN_RBUF08	DATA1								Data
CAN_RBUF09	DATA2								Data

…	…	…
CAN_RBUF71	DATA64	Data

IDE：扩展帧标志位

0：标准帧

1：扩展帧

RTR：远程帧标志位

0：数据帧

1：远程帧（仅适用于 CAN2.0，CAN-FD 固定位 0）

FDF：CAN-FD 帧标志位

0：CAN2.0 帧

1：CAN-FD 帧

BRS：波特率切换标志位

0：整帧数据位低速波特率

1：数据和 CRC 位快速波特率（仅适用于 CAN-FD，CAN2.0 固定位 0）

DLC[3:0]：数据长度标志

DLC[3:0]	CAN2.0 数据长度	CAN-FD 数据长度
0000	0 字节	0 字节
0001	1 字节	1 字节
…	…	…
1000	8 字节	8 字节
1001		12 字节
1010		16 字节
1011		20 字节
1100		24 字节
1101		32 字节
1110		48 字节
1111		64 字节

KOER[2:0]：错误类型

KOER[2:0]	错误类型
000	无错误
001	位错误
010	形式错误
011	填充错误
100	应答错误
101	CRC 错误
110	其他错误
111	保留

TX：回环模式中接收自己发送的数据时此位为 1

38.7.2 CAN 发送缓冲区寄存器（CAN_TBUFn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_TBUF00	7EF450H	CAN_TBUF00[7:0]							
CAN_TBUF01	7EF451H	CAN_TBUF01[7:0]							
…	…	…							
CAN_TBUF71	7EF497H	CAN_TBUF71[7:0]							

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN2_TBUF00	7EF350H	CAN3_TBUF00[7:0]							
CAN2_TBUF01	7EF351H	CAN3_TBUF01[7:0]							
…	…	…							
CAN2_TBUF71	7EF397H	CAN3_TBUF71[7:0]							

CAN_TBUFn[7:0]：待发送的 CAN 邮箱数据

标准帧的发送数据格式如下：

地址	B7	B6	B5	B4	B3	B2	B1	B0	功能
CAN_TBUF00	ID[7:0]								ID
CAN_TBUF01	-	-	-	-	-	ID[10:8]			ID
CAN_TBUF02	-	-	-	-	-	-	-	-	ID
CAN_TBUF03	-	-	-	-	-	-	-	-	ID
CAN_TBUF04	IDE=0	RTR	FDF	BRS	DLC[3:0]				Control
CAN_TBUF05	-	-	-	-	-	-	-	-	Status
CAN_TBUF06	-	-	-	-	-	-	-	-	TTCAN
CAN_TBUF07	-	-	-	-	-	-	-	-	TTCAN
CAN_TBUF08	DATA1								Data
CAN_TBUF09	DATA2								Data
…	…								…
CAN_TBUF71	DATA64								Data

扩展帧的发送数据格式如下：

地址	B7	B6	B5	B4	B3	B2	B1	B0	功能
CAN_TBUF00	ID[7:0]								ID
CAN_TBUF01	ID[15:8]								ID
CAN_TBUF02	ID[23:16								ID
CAN_TBUF03	-	-	-	ID[28:24]					ID
CAN_TBUF04	IDE=1	RTR	FDF	BRS	DLC[3:0]				Control
CAN_TBUF05	-	-	-	-	-	-	-	-	Status
CAN_TBUF06	-	-	-	-	-	-	-	-	TTCAN
CAN_TBUF07	-	-	-	-	-	-	-	-	TTCAN
CAN_TBUF08	DATA1								Data
CAN_TBUF09	DATA2								Data
…	…								…
CAN_TBUF71	DATA64								Data

控制位含义请参考 CAN 接收缓冲区寄存器说明

38.7.3 CAN 传输时间戳寄存器（CAN_TTSn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_TTS00	7EF498H	CAN_TTS00[7:0]							
CAN_TTS01	7EF499H	CAN_TTS01[7:0]							
…	…	…							
CAN_TTS07	7EF49FH	CAN_TTS07[7:0]							

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN2_TTS00	7EF398H	CAN2_TTS00[7:0]							
CAN2_TTS01	7EF399H	CAN2_TTS01[7:0]							
…	…	…							
CAN2_TTS07	7EF39FH	CAN2_TTS07[7:0]							

CAN_TTSn[7:0]： TTS 存储 CiA603 时间戳功能中最后传输帧的时间戳。当 TTSEN=1 时，每帧新数据会覆盖 TTS。根据通用参数的设定，时间戳宽度可为 32 位或 64 位。未使用的位将强制置零。TTS 供时间主站使用，用于获取 SYNC 消息的时间戳。

38.7.4 CAN 配置状态寄存器（CAN_CFG_STAT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_CFG_STAT	7EF4A0H	RESET	LBME	LBMI	TPSS	TSSS	RACTIVE	TACTIVE	BUSOFF
CAN2_CFG_STAT	7EF3A0H	RESET	LBME	LBMI	TPSS	TSSS	RACTIVE	TACTIVE	BUSOFF

RESET：复位请求控制位

0：不请求局部复位

1：请求局部复位

LBME：外部回环模式使能位（通信中禁止设定此位）

0：禁止外部回环模式

1：使能外部回环模式

LBMI：内部回环模式使能位（通信中禁止设定此位）

0：禁止内部回环模式

1：使能内部回环模式

TPSS：PTB 单次传输模式使能位

0：禁止 PTB 单次传输模式

1：使能 PTB 单次传输模式

TSSS：STB 单次传输模式使能位

0：禁止 STB 单次传输模式

1：使能 STB 单次传输模式

RACTIVE：接收状态信号（只读）

0：非接收状态

1：正在接收中

TACTIVE：发送状态信号（只读）

0：非发送状态

1：正在发送中

BUSOFF：总线关闭状态

0：总线处于有效状态

1：总线处于关闭状态

38.7.5 CAN 命令寄存器（CAN_TCMD）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_TCMD	7EF4A1H	TBSEL	LOM	STBY	TPE	TPA	TSONE	TSALL	TSA
CAN2_TCMD	7EF3A1H	TBSEL	LOM	STBY	TPE	TPA	TSONE	TSALL	TSA

TBSEL：发送 BUF 选择位（当 TTEN==1 && TTTBM==1 时 TBSEL 被复位成复位值）

0：PTB

1：STB

LOM：静默模式使能位

0：禁止静默模式

1：使能静默模式

STBY：收发器待机模式

0：禁止待机模式

1：使能待机模式

【说明】：该寄存器位连接到 CANSTB 引脚，输出信号可用于控制收发器的待机模式。（需要设置寄存器 CANn_AUX_CR 的 STBYEN）

如果 TPE=1、TSONE=1 或 TSALL=1，则无法将 STBY 设置为 1。

如果主机将 STBY 设置为 0，则主机需要等待收发器启动所需的时间，然后才能请求新的传输。

STBY=1 时 CAN 控制器不能发送，但是可以接收，接收一次后 STBY 自动清零。

TPE：PTB 发送使能位

0：禁止 PTB 发送

1：使能 PTB 发送

TPA：PTB 发送取消位

0：不取消

1：取消已经通过 TPE 设置为 1 但还未开始的 PTB 发送（硬件自动清零）

TSONE：发送一帧 STB 数据设置位

0：不发送

1：发送一帧 STB 数据

TSALL：发送全部的 STB 数据设置位

0：不发送

1：发送全部的 STB 数据

TSA：STB 发送取消位

0：不取消

1：取消已经通过 TSONE 或者 TSALL 设置为 1 但还未开始的 STB 发送（硬件自动清零）

38.7.6 CAN 发送控制寄存器（CAN_TCTRL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_TCTRL	7EF4A2H	FD_ISO	TSNEXT	-	TTTBM	-	-	TSSTAT[1:0]	
CAN2_TCTRL	7EF3A2H	FD_ISO	TSNEXT	-	TTTBM	-	-	TSSTAT[1:0]	

FD_ISO：CAN-FD ISO 模式选择

0：Bosch CAN-FD 模式

1：11898-1:2015 CAN-FD 模式

TSNEXT：下一个 STB

0：无动作

1：当前的 STB SLOT 已填充，指向下一个 SLOT

TTTBM：TTCAN BUF 模式

0：TSMODE 决定 PTB 和 STB

1：通过 TBPTR 和 TTPTR 设定

TSTAT[1:0]：错误类型（只读）

TSTAT[1:0]	STB 状态	
	（TTEN==0）‖（TTEN==1 && TTTBM==0）	（TTEN==1 && TTTBM==1）
00	STB 空	PTB 和 STB 空
01	STB 小于等于半满	PTB 和 STB 非满
10	STB 大于半满	保留
11	STB 满	PTB 和 STB 满

38.7.7 CAN 接收控制寄存器（CAN_RCTRL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_RCTRL	7EF4A3H	SACK	ROM	ROV	RREL	RBALL	-	RSTAT[1:0]	
CAN2_RCTRL	7EF3A3H	SACK	ROM	ROV	RREL	RBALL	-	RSTAT[1:0]	

SACK：自应答

0：无应答

1：LBME==1 时，使能自应答功能

ROM：接收 BUF 上溢模式设置

0：最早接收的数据被覆盖

1：新接收的数据不存储

ROV：接收 BUF 上溢标志位（只读）

0：无上溢

1：发生上溢，需通过写 1 到 RREL 位来清零 ROV

RREL：释放接收 BUF

0：不释放

1：表示该接收 BUF 已经被读取过，RBUF 寄存器指向下一个 RB SLOT

RBALL：接收 BUF 数据存储所有的数据帧

0：正常模式

1：存储所有的数据包（包括有错误的数据）

RSTAT[1:0]：错误类型（只读）

RSTAT[1:0]	接收 BUF 状态
00	RBUF 空
01	RBUF 非空且小于 AFWL 的设定值
10	RBUF 非空且大于等于 AFWL 的设定值
11	RBUF 满（上溢时保持此值）

38.7.8 CAN 中断使能寄存器（CAN_RTIE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_RTIE	7EF4A4H	RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	TSFF
CAN2_RTIE	7EF3A4H	RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	TSFF

RIE：接收中断控制位

0：禁止

1：使能

ROIE：接收上溢中断控制位

0：禁止

1：使能

RFIE：接收 BUF 满中断控制位

0：禁止

1：使能

RAFIE：接收 BUF 将满中断控制位

0：禁止

1：使能

TPIE：PTB 发送中断控制位

0：禁止

1：使能

TSIE：STB 发送中断控制位

0：禁止

1：使能

EIE：错误中断控制位

0：禁止

1：使能

TSFF：发送 BUF 满标志

TTEN==0 || TTTBM==0 时 STB 满标志

0：STB SLOT 没有被全部填充

1：STB SLOT 被全部填充

TTEN==1 && TTTBM==1 时 TB 满标志

0：TBPTR 选的发送 BUF 没有被全部填充

1：TBPTR 选的发送 BUF 被全部填充

38.7.9 CAN 中断状态寄存器（CAN_RTIF）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_RTIF	7EF4A5H	RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF
CAN2_RTIF	7EF3A5H	RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF

RIF：接收中断标志位。需要软件写“1”清零

0：未接收到数据帧

1：接收到有效的数据帧或远程帧

ROIF：接收上溢中断标志位。需要软件写“1”清零

0：无 RB 被覆盖

1：至少有一个 RB 被覆盖

RFIF：接收 BUF 满中断标志位。需要软件写“1”清零

0：RB FIFO 未满

1：RB FIFO 满

RAFIF：接收 BUF 将满中断标志位。需要软件写“1”清零

0：被填充的 RB SLOT 数目小于 AFWL 设置的值

1：被填充的 RB SLOT 数目大于等于 AFWL 设置的值

TPIF：PTB 发送中断标志位。需要软件写“1”清零

0：没有 PTB 发送完成

1：请求的 PTB 发送完成

TSIF：STB 发送中断标志位。需要软件写“1”清零

0：没有 STB 发送完成

1：请求的 STB 发送完成

EIF：错误中断标志位。需要软件写“1”清零

0：BUSOFF 位未发生变化，或者错误计数器的值与 EWL 设置的值相对关系未发生变化

1：BUSOFF 位发生变化，或者错误计数器的值与 EWL 设置的值相对关系发生变化

AIF：取消发送中断标志位。需要软件写“1”清零

0：未取消发送数据

1：通过 TPA 和 TSA 请求的发送消息被成功取消

38.7.10 CAN 错误中断寄存器（CAN_ERRINT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_ERRINT	7EF4A6H	EWARN	EPASS	EPIE	EPIF	ALIE	ALIF	BEIE	BEIF
CAN2_ERRINT	7EF3A6H	EWARN	EPASS	EPIE	EPIF	ALIE	ALIF	BEIE	BEIF

EWARN：错误计数器的值状态（只读）

0：错误计数器（RECNT 或者 TECNT）的值小于 EWL 设置的值

1：错误计数器（RECNT 或者 TECNT）的值大于或等于 EWL 设置的值

EPASS：错误被动（只读）

0：节点是主动错误节点

1：节点是被动错误节点

EPIE：错误被动中断控制位

0：禁止

1：使能

EPIF：错误被动中断标志位。需要软件写“1”清零

0：未发生错误主动到错误被动或者错误被动到错误主动的变化

1：发生错误主动到错误被动或者错误被动到错误主动的变化

ALIE：仲裁失败中断控制位

0：禁止

1：使能

ALIF：仲裁失败中断标志位。需要软件写“1”清零

0：仲裁成功

1：仲裁失败

BEIE：总线错误中断控制位

0：禁止

1：使能

BEIF：总线错误中断标志位。需要软件写“1”清零

0：无总线错误

1：发生总线错误

38.7.11 CAN 警告限制寄存器（CAN_LIMIT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_LIMIT	7EF4A7H	AFWL[3:0]				EWL[3:0]			
CAN2_LIMIT	7EF3A7H	AFWL[3:0]				EWL[3:0]			

AFWL：接收 BUF 将满警告设定值。设定范围为 1~8，设置为 0 无意义，当作 ALWL 为 1 处理

EWL：接收 BUF 错误警告的编程值。错误警告值=（EWL+1）*8

38.7.12 CAN 慢速位段 1 时间设定寄存器（CAN_S_SEGn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_S_SEG1	7EF4A8H	S_SEG1[7:0]							
CAN2_S_SEG1	7EF3A8H	S_SEG1[7:0]							

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_S_SEG2	7EF4A9H	-	S_SEG2[6:0]						
CAN2_S_SEG2	7EF3A9H	-	S_SEG2[6:0]						

S_SEG1：慢速 CAN 波特率段位 1 时间设置

S_SEG2：慢速 CAN 波特率段位 2 时间设置

$$慢速CAN波特率 = \frac{CAN工作频率}{（S_PRESC + 1）*（（S_SEG1 + 2）+（S_SEG2 + 1））}$$

38.7.13 CAN 慢速同步补偿时间设定寄存器（CAN_S_SJW）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_S_SJW	7EF4AAH	-	S_SJW[6:0]						
CAN2_S_SJW	7EF3AAH	-	S_SJW[6:0]						

S_SJW：慢速 CAN 波特率再同步补偿宽度时间设置

补偿规则： S_SEG1 >= S_SEG2+1

S_SEG2 >= S_SJW

38.7.14 CAN 慢速预分频寄存器（CAN_S_PRESC）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_S_PRESC	7EF4ABH	S_PRESC[7:0]							
CAN2_S_PRESC	7EF3ABH	S_PRESC[7:0]							

S_PRESC：慢速 CAN 波特率预分频设置

38.7.15 CAN 快速位段 1 时间设定寄存器（CAN_F_SEGn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_F_SEG1	7EF4ACH	-	-	-	F_SEG1[4:0]				
CAN2_F_SEG1	7EF3ACH	-	-	-	F_SEG1[4:0]				

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_F_SEG2	7EF4ADH	-	-	-	-	F_SEG2[3:0]			
CAN2_F_SEG2	7EF3ADH	-	-	-	-	F_SEG2[3:0]			

F_SEG1：快速 CAN 波特率段位 1 时间设置

F_SEG2：快速 CAN 波特率段位 2 时间设置

$$\text{快速CAN波特率} = \frac{\text{CAN工作频率}}{(\text{F_PRESC}+1)*((\text{F_SEG1}+2)+(\text{F_SEG2}+1))}$$

38.7.16 CAN 快速同步补偿时间设定寄存器（CAN_F_SJW）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_F_SJW	7EF4AEH	-	-	-	-	F_SJW[3:0]			
CAN2_F_SJW	7EF3AEH	-	-	-	-	F_SJW[3:0]			

F_SJW：快速 CAN 波特率再同步补偿宽度时间设置

补偿规则： F_SEG1 >= F_SEG2

F_SEG2 >= F_SJW

38.7.17 CAN 快速预分频寄存器（CAN_F_PRESC）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_F_PRESC	7EF4AFH	F_PRESC[7:0]							
CAN2_F_PRESC	7EF3AFH	F_PRESC[7:0]							

F_PRESC：快速 CAN 波特率预分频设置

38.7.18 CAN 错误和仲裁丢失捕获寄存器（CAN_EALCAP）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_EALCAP	7EF4B0H	KOER[2:0]			ALC[4:0]				
CAN2_EALCAP	7EF3B0H	KOER[2:0]			ALC[4:0]				

KOER[2:0]：错误类型（只读）

KOER[2:0]	错误类型
000	无错误
001	位错误
010	形式错误
011	填充错误
100	应答错误
101	CRC 错误
110	其他错误
111	保留

ALC[4:0]：仲裁失败位置捕获。仲裁失败时 ALC 记录一帧数据中仲裁失败时的位置。（只读）

38.7.19 CAN 发送器延迟补偿寄存器（CAN_TDC）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_TDC	7EF4B1H	TDCEN	SSPOFF[6:0]						
CAN2_TDC	7EF3B1H	TDCEN	SSPOFF[6:0]						

TDCEN：发送器延迟补偿功能控制位（用于 CAN-FD BRS=1 的情况）

0：禁止

1：使能

SSPOFF[6:0]：TDC 功能辅助采样点位置设置

38.7.20 CAN 接收错误计数器（CAN_RECNT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_RECNT	7EF4B2H	RECNT[7:0]							
CAN2_RECNT	7EF3B2H	RECNT[7:0]							

RECNT[7:0]：接收错误计数器。计数器不会上溢，最大值为 255。（只读）

38.7.21 CAN 发送错误计数器（CAN_TECNT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_TECNT	7EF4B3H	TECNT[7:0]							
CAN2_TECNT	7EF3B3H	TECNT[7:0]							

RECNT[7:0]：发送错误计数器。计数器不会上溢，最大值为 255。（只读）

38.7.22 CAN 筛选器组控制寄存器（CAN_ACFCTRL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_ACFCTRL	7EF4B4H	-	-	SELMASK	-	ACFADR[3:0]			
CAN2_ACFCTRL	7EF3B4H	-	-	SELMASK	-	ACFADR[3:0]			

SELMASK：选择筛选器的屏蔽寄存器

0：ACF 指向筛选器 ID 寄存器

1：ACF 指向筛选器 MASK 寄存器

ACFADR[3:0]：筛选器地址

ACFADR[2:0]	筛选器	ACFADR[2:0]	筛选器
0000	指向 ACF_1	1000	指向 ACF_9
0001	指向 ACF_2	1001	指向 ACF_10
0010	指向 ACF_3	1010	指向 ACF_11
0011	指向 ACF_4	1011	指向 ACF_12
0100	指向 ACF_5	1100	指向 ACF_13
0101	指向 ACF_6	1101	指向 ACF_14
0110	指向 ACF_7	1110	指向 ACF_15
0111	指向 ACF_8	1111	指向 ACF_16

38.7.23 CAN 时间戳控制寄存器（CAN_TIMECFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_TIMECFG	7EF4B5H	-	-	-	-	-	-	TIMEPOS	TIMEEN
CAN2_TIMECFG	7EF3B5H	-	-	-	-	-	-	TIMEPOS	TIMEEN

TIMEPOS：时间戳位置

0：帧头（SOF）

1：帧尾（EOF）

TIMEEN：时间戳控制位

0：禁止

1：使能

38.7.24 CAN 筛选器组使能寄存器（CAN_ACF_ENn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_ACF_EN0	7EF4B6H	AE_7	AE_6	AE_5	AE_4	AE_3	AE_2	AE_1	AE_0
CAN_ACF_EN1	7EF4B7H	AE_15	AE_14	AE_13	AE_12	AE_11	AE_10	AE_9	AE_8

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN2_ACF_EN0	7EF3B6H	AE_7	AE_6	AE_5	AE_4	AE_3	AE_2	AE_1	AE_0
CAN2_ACF_EN1	7EF3B7H	AE_15	AE_14	AE_13	AE_12	AE_11	AE_10	AE_9	AE_8

AE_n：ACF_n 选择筛使能位（n=0~15）

0：禁止 ACF_n

1：使能 ACF_n

38.7.25 CAN 筛选器 code 和 mask 寄存器（CAN_ACFn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_ACF0	7EF4B8H	ACODE[7:0]/AMASK[7:0]							
CAN_ACF1	7EF4B9H	ACODE[15:8]/AMASK[15:8]							
CAN_ACF2	7EF4BAH	ACODE[23:16]/AMASK[23:16]							
CAN_ACF3	7EF4BBH	-	AIDEE	AIDE	ACODE[28:24]/AMASK[28:24]				

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN2_ACF0	7EF3B8H	ACODE[7:0]/AMASK[7:0]							
CAN2_ACF1	7EF3B9H	ACODE[15:8]/AMASK[15:8]							
CAN2_ACF2	7EF3BAH	ACODE[23:16]/AMASK[23:16]							
CAN2_ACF3	7EF3BBH	-	AIDEE	AIDE	ACODE[28:24]/AMASK[28:24]				

AIDEE：IDE 位比较使能

0：筛选器接收标准格式和扩展格式帧

1：筛选器接收 AIDE 位定义的标准格式或者扩展格式帧

AIDE：IDE 位 MASK

0：筛选器仅接收标准格式帧

1：筛选器仅接收扩展格式帧

ACODE[28:0]/AMASK[28:0]：筛选器 CODE/MASK，通过 ACFADR 指向具体的筛选器。

SELMASK=0 时，表示筛选器的 CODE；

SELMASK=1 时，表示筛选器的 MASK；

38.7.26 CAN 版本信息寄存器（CAN_VERn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_VER0	7EF4BCH	VER0[7:0]							
CAN_VER1	7EF4BDH	VER1[7:0]							

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN2_VER0	7EF3BCH	VER0[7:0]							
CAN2_VER1	7EF3BDH	VER1[7:0]							

VER：CAN 控制器的版本号

38.7.27 CANTB SLOT 指针寄存器（CAN_TBSLOT）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_TBSLOT	7EF4BEH	TBE	TBF	-	-	-	TBPTR[2:0]		
CAN2_TBSLOT	7EF3BFH	TBE	TBF	-	-	-	TBPTR[2:0]		

TBE：设置 TB 位空

0：无操作

1：被 TBPTR 选择的 SLOT 被标记为空

当 SLOT 被标记为空且 TSFF=0 时，TBE 自动复位为 0

TBF：设置 TB 位已填充

0：无操作

1：被 TBPTR 选择的 SLOT 被标记为已填充

当 SLOT 被标记为已填充且 TSFF=1 时，TBF 自动复位为 0

TBPTR[2:0]：TB SLOT 指针

TBPTR[2:0]	TB SLOT 指针
000	指向 PTB
001	指向 STB SLOT1
010	指向 STB SLOT2
011	指向 STB SLOT3
其他	保留

38.7.28 CAN 时间触发配置寄存器（CAN_TTCFG）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_TTCFG	7EF4BFH	WTIE	WTIF	TEIF	TTIE	TTIF	T_PRESC[1:0]		TTEN
CAN2_TTCFG	7EF3BFH	WTIE	WTIF	TEIF	TTIE	TTIF	T_PRESC[1:0]		TTEN

WTIE：触发看门中断使能

0：禁止

1：使能

WTIF：触发看门中断标志位。需要软件写“1”清零

TEIF：触发错误中断标志位。需要软件写“1”清零

TTIE：时间触发中断使能

0：禁止

1：使能

TTIF：时间触发中断标志位。需要软件写“1”清零

T_PRESC[1:0]：TTCAN 计数器预分频

T_PRESC[1:0]	SBT 寄存器设置的位时间分频
00	1 分频
01	2 分频
10	4 分频
11	8 分频

TTEN：TTCAN 使能

0：禁止

1：使能 TTCAN，计数器开始计数

38.7.29 CAN 参考消息寄存器（CAN_REF_MSGn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_REF_MSG0	7EF4C0H	REF_ID[7:0]							
CAN_REF_MSG1	7EF4C1H	REF_ID[15:8]							
CAN_REF_MSG2	7EF4C2H	REF_ID[23:16]							
CAN_REF_MSG3	7EF4C3H	REF_IDE	-	-	REF_ID[28:24]				

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN2_REF_MSG0	7EF3C0H	REF_ID[7:0]							
CAN2_REF_MSG1	7EF3C1H	REF_ID[15:8]							
CAN2_REF_MSG2	7EF3C2H	REF_ID[23:16]							
CAN2_REF_MSG3	7EF3C3H	REF_IDE	-	-	REF_ID[28:24]				

REF_IDE：参考消息的 IDE 位

0：标准格式

1：扩展格式

REF_ID：参考消息的 ID 位

当 REF_IDE=0 时，REF_ID[10:0]有效

当 REF_IDE=1 时，REF_ID[28:0]有效

38.7.30 CAN 触发配置寄存器（CAN_TRIG_CFGn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_TRIG_CFG0	7EF4C4H	-	-	TTPTR[5:0]					
CAN_TRIG_CFG1	7EF4C5H	TEW[3:0]				-	TTYPE[2:0]		

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN2_TRIG_CFG0	7EF3C4H	-	-	TTPTR[5:0]					
CAN2_TRIG_CFG1	7EF3C5H	TEW[3:0]				-	TTYPE[2:0]		

TEW：发送使能窗口。用于 TTCAN 的单次发送触发模式，可以设定 TEW+1 个 cycle time 的窗口，发送仅在此窗口内被允许。

TTYPE[2:0]：触发类型。（触发时间通过 TT_TRIG 寄存器设置，TB SLOT 通过 TTPTR 选择）

TTYPE[2:0]	触发类型
000	立即触发
001	时间触发
010	单次触发
011	发送开始触发
100	发送停止触发
其他	保留

TTPTR[2:0]：发送触发器 TB SLOT 指针

TTPTR[2:0]	发送触发器 TB SLOT 指针
000	指向 PTB
001	指向 STB SLOT1
010	指向 STB SLOT2
011	指向 STB SLOT3
其他	保留

38.7.31 CAN 触发时间寄存器（CAN_TT_TRIGn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_TT_TRIG0	7EF4C6H	TT_TRIG[7:0]							
CAN_TT_TRIG1	7EF4C7H	TT_TRIG[15:8]							

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN2_TT_TRIG0	7EF3C6H	TT_TRIG[7:0]							
CAN2_TT_TRIG1	7EF3C7H	TT_TRIG[15:8]							

TT_TRIG[15:0]：触发时间设置。需先写低字节，后写高字节

38.7.32 CAN 触发看门时间寄存器（CAN_TT_WTRIGn）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN_TT_WTRIG0	7EF4C8H	TT_WTRIG[7:0]							
CAN_TT_WTRIG1	7EF4C9H	TT_WTRIG[15:8]							

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
CAN2_TT_WTRIG0	7EF3C8H	TT_WTRIG[7:0]							
CAN2_TT_WTRIG1	7EF3C9H	TT_WTRIG[15:8]							

TT_WTRIG[15:0]：看门触发时间设置。需先写低字节，后写高字节

39 LIN 总线

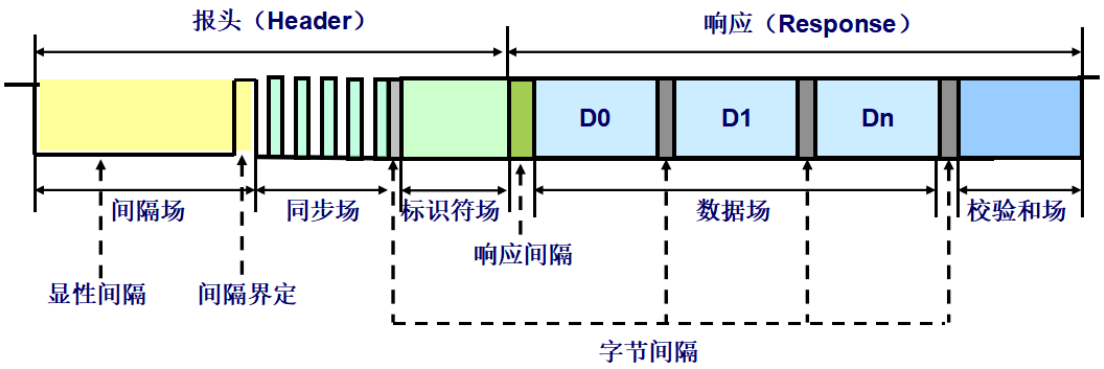
产品线	LIN
STC32G144K246 系列	2 组

STC32G144K246 系列单片机内部集成两组 LIN 总线功能单元，支持 LIN2.1 和 LIN1.3 协议协议。

主要功能如下：

- 帧头自动处理
- 可以在主、从两种模式之间切换
- 超时检测
- 错误分析

39.1 LIN 总线时序介绍



1、字节场

1）基于 UART/SCI 的通信格式；

2）发送一个字节需要 10 个位时间（TBIT）

2、间隔场

1）表示一帧报文的起始，由主节点发出；

2）间隔信号至少由 13 个显性位组成；

3）间隔界定符至少由 1 个隐形位组成；

4）间隔场是唯一一个不符合字节场格式的场；

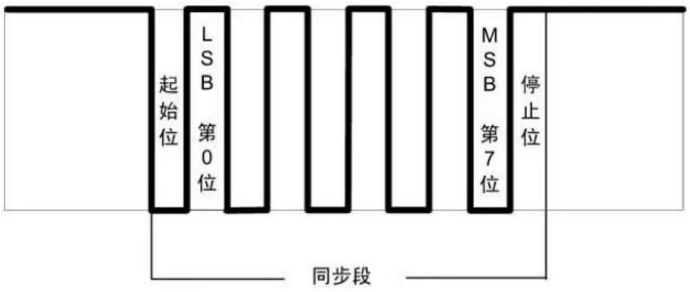
5）从节点需要检测到至少连续 11 个显性位才认为是间隔信号；



3、同步场

1）确保所有从节点使用与节点相同的波特率发送和接收数据；

2）一个字节，结构固定：0x55；



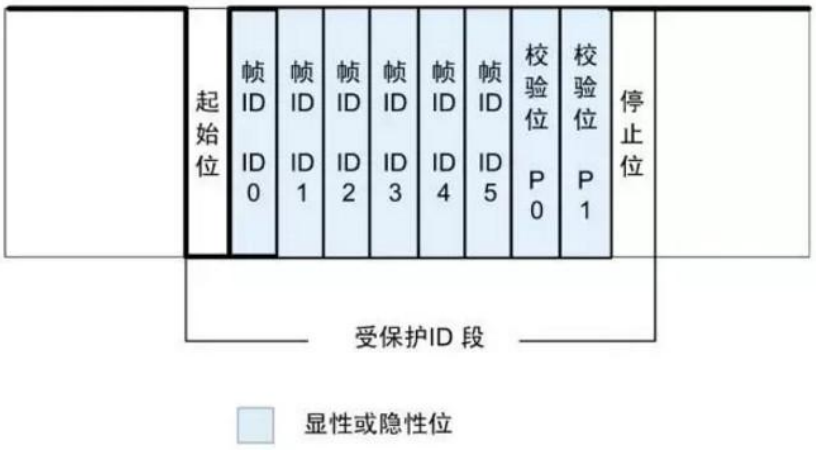
4、标识符场

1）ID 的范围从 0 到 63(0x3f)；

2）奇偶校验符(Parity)P0,P1

$$P0 = ID0 \oplus ID1 \oplus ID2 \oplus ID4$$

$$P1 = \neg (ID1 \oplus ID3 \oplus ID4 \oplus ID5)$$

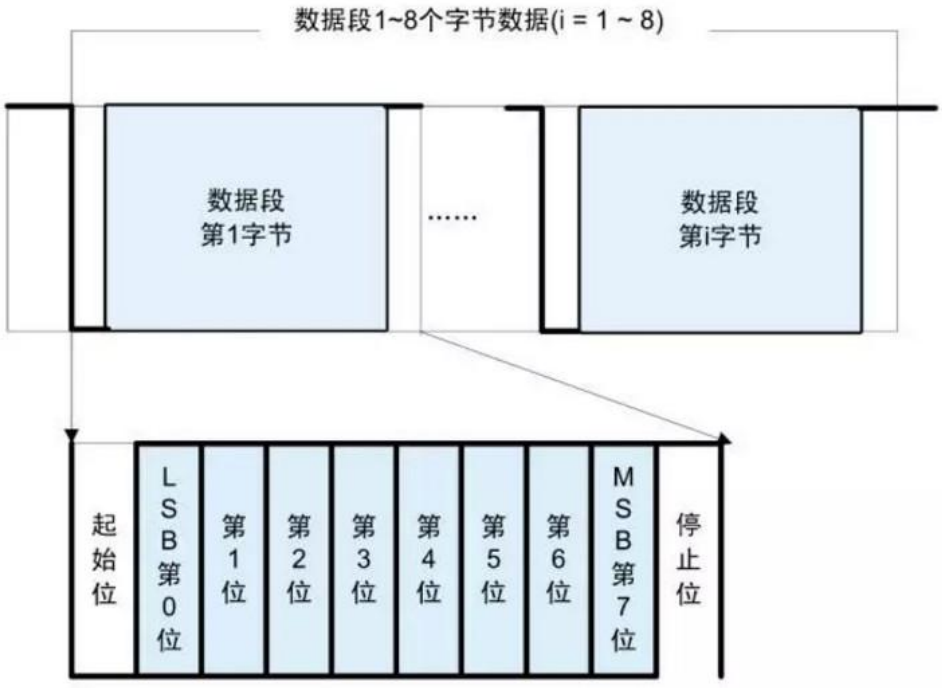


5、数据场

1）数据场长度 1 到 8 个字节；

2）低字节先发，低位先发；

3）如果某信号长度超过 1 个字节采用低位在前的方式发送（小端）；



6、校验和场

用于校验接收的数据是否正确

1）经典校验（Classic Checksum）仅校验数据场(LIN1.3)

2）增强校验（Enhance Checksum）校验标识符场与数据场内容(LIN2.0、LIN2.1)

标识符为 0x3C 和 0x3D 的帧只能使用经典校验

计算方法：反转 8 位求和



39.2 LIN1 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SW1	A2H	S1_S[1:0]		CAN_S[1:0]		SPI_S[1:0]		**LIN_S[1:0]**	

LIN_S[1:0]：LIN 功能脚选择位

LIN_S[1:0]	LIN_RX	LIN_TX
00	P0.2	P0.3
01	P5.2	P5.3
10	P4.6	P4.7
11	P7.2	P7.3

39.3 LIN2 功能脚切换

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
P_SWX2	A2H	-	-	**LIN2_S[1:0]**		S6_S[1:0]		S5_S[1:0]	

LIN2_S[1:0]：LIN2 功能脚选择位

LIN2_S[1:0]	LIN2_RX	LIN2_TX
00	P0.6	P0.7
01	P2.6	P2.7
10	P6.2	P6.3
11	P8.4	P8.5

39.4 LIN 相关的寄存器

符号	描述	地址	位地址与符号								复位值
			B7	B6	B5	B4	B3	B2	B1	B0	
LINICR	LINBUS 中断控制寄存器	F9H	PLIN2H	LIN2IF	LIN2IE	PLIN2L	PLINH	LINIF	LINIE	PLINL	0000,0000
LINAR	LINBUS 地址寄存器	FAH									0000,0000
LINDR	LINBUS 数据寄存器	FBH									0000,0000
AUXR2	辅助寄存器 2	97H	RAMTINY	CPUMODE	LINSEL	LIN2EN	CANEDIN	CAN2EN	CANEN	LINEN	0000,0000

39.4.1 辅助寄存器 2（AUXR2）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
AUXR2	97H	RAMTINY	CPUMODE	LINSEL	LIN2EN	CANEDIN	CAN2EN	CANEN	LINEN

LINEN：LIN 总线使能控制位

0：关闭 LIN 功能

1：使能 LIN 功能

LIN2EN：LIN2 总线使能控制位

0：关闭 LIN2 功能

1：使能 LIN2 功能

LINSEL：LIN 总线选择

0：选择第 1 组 LIN

1：选择第 2 组 LIN

39.4.2 LIN 总线中断控制寄存器（LINICR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LINICR	F9H	PLIN2H	LIN2IF	LIN2IE	PLIN2L	PLINH	LINIF	LINIE	PLINL

LINIE：LIN 总线中断使能控制位

0：关闭 LIN 中断

1：使能 LIN 中断

LINIF：LIN 总线中断请求标志位，硬件清零（读取 LSR 清零）。

PLINH，PLINL：LIN 中断优先级控制位

PLINH	PLINL	优先级
0	**0**	**0（最低）**
0	1	1
1	0	2
1	1	3（最高）

LIN2IE：LIN2 总线中断使能控制位

0：关闭 LIN2 中断

1：使能 LIN2 中断

LIN2IF：LIN2 总线中断请求标志位，硬件清零（读取 LSR 清零）。

PLIN2H，PLIN2L：LIN2 中断优先级控制位

PLIN2H	PLIN2L	优先级
0	**0**	**0（最低）**
0	1	1
1	0	2
1	1	3（最高）

39.4.3 LIN 总线地址寄存器（LINAR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LINAR	FAH								

39.4.4 LIN 总线数据寄存器（LINDR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LINDR	FBH								

对 LIN 内部功能寄存器进行读写均需要通过 LINAR 和 LINDR 进行间接访问

读 LIN 内部功能寄存器的方法：

1、将 LIN 内部功能寄存器的地址写入 LIN 总线地址寄存器 LINAR

2、读取 LIN 总线数据寄存器 LINDR

例如：需要读取 LIN 内部功能寄存器 LSR 的值

```
LINAR = 0x05;          //将 LSR 的地址写入 LINAR
dat = LINDR;           //读取 CANDR 以获得 LSR 的值
```

写 LIN 内部功能寄存器的方法：

1、将 LIN 内部功能寄存器的地址写入 LIN 总线地址寄存器 LINAR

2、将待写入的值写入 LIN 总线数据寄存器 LINDR

例如：需要将数据 0x5a 写入 LIN 内部功能寄存器 LBUF

```
LINAR = 0x00;          //将 LBUF 的地址写入 LINAR
LINDR = 0x5a;          //将待写入的值 0x5a 写入 LINDR
```

39.5 LIN 内部功能寄存器

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
08H	HDRL	HDRH	HDP					
00H	LBUF	LSEL	LID	LER	LIE	LSR	DLL	DLH

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LBUF	0x00	LBUF							
LSEL	0x01	ANIC				INDEX			
LID	0x02			ID					
LER	0x03		OVER	SYNCER	TOVER	CHKER	PER	BITER	FER
LIE	0x04					ABORTE	ERRE	RDYE	LIDE
LSR	0x05	LOG SIZE				ABORT	ERR	RDY	LID
LCR	0x05	MODE	LIN13	SIZE				CMD	
DLL	0x06	DLL							
DLH	0x07	SYNC	DLH						
HDRL	0x08	HDRL							
HDRH	0x09	HDRH							
HDP	0x0A			HDP					

39.5.1 LIN 数据寄存器（LBUF）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LBUF	0x00								

LBUF 是 LIN 模块内部数据 FIFO 的数据接口。

39.5.2 LIN 数据地址寄存器（LSEL）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LSEL	0x01	AINC	-	-	-	INDEX[3:0]			

AINC 地址自增

0：写入 LIN 内部 FIFO 的数据地址由 INDEX 决定。

1：数据地址自增，每操作一次 LBUF，LIN 内部 FIFO 的数据地址自动加一。

39.5.3 LIN 帧 ID 寄存器（LID）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LID	0x02			ID[5:0]					

LID 帧 ID，如果 LCR[5:2]=4’b1111，那么 LID[5:4]的组合代表数据帧中数据的数量：

00：2 字节

01：2 字节

10：4 字节

11：8 字节

39.5.4 LIN 错误寄存器（LER）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LER	0x03	-	OV	SYN	TOV	CHK	PER	BIT	FER

OV：over run error 。当 LIN 正在运行命令（RDY 为低）时，用户发送了新的命令到 LIN。

SYN：slave 模式下才有，LIN 接收帧同步时产生错误。

TOV：timeout 错误。在最大允许时间内，LIN 没有收到完整的数据帧。

CHK：checksum 错误。

PER：Parity error,没有正确接收到受保护的 ID 段。

BIT：位错误，标志 LIN 发送的数据位与总线监测到的状态不一致。

FER：帧错误，接收到的数据没有包含有效的停止位。

读取清除错误寄存器。

39.5.5 LIN 中断使能寄存器（LIE）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LIE	0x04	-	-	-	-	ABORTE	ERRE	RDYE	LIDE

ABORTE：使能终止中断。

ERRE：使能错误中断。

RDYE：使能 Ready 中断。

LIDE：使能 head 中断。

39.5.6 LIN 状态寄存器（只读寄存器）（LSR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LSR	0x05	LOG SIZE[3:0]				ABORT	ERR	RDY	LID

LOG SIZE：LOG 模式时，检测到的数据长度。

ABORT：终止命令正在执行中。

ERR：错误状态。

RDY：Ready 状态，可以执行新的命令。

LID：接收到了正确的 header。

Bit0~Bit3 读取后清零。

39.5.7 LIN 控制寄存器（只写寄存器）（LCR）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
LCR	0x05	MODE	LIN13	SIZE[3:0]				CMD[1:0]	

MODE：LIN 模式选择

0：从模式

1：主模式。

LIN13：校验和选择。

0：增强校验和，LIN2.1 协议。

1：普通校验和，LIN1.3 协议。

SIZE[3:0]：数据长度

设置	长度	设置	长度
0000	0 字节	1000	8 字节
0001	1 字节	1001	保留
0010	2 字节	1010	保留
0011	3 字节	1011	保留
0100	4 字节	1100	保留
0101	5 字节	1101	保留
0110	6 字节	1110	LOG MODE
0111	7 字节	1111	数据长度由 LID[5:4]决定

CMD[1:0]：LIN 命令。

00：Abort 命令。

01：主模式 Send header 命令。

10：TX response。

11：RX response。

39.5.8 LIN 波特率寄存器（DLL/DLH）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
DLL	0x06	DLL							
DLH	0x07	SYNC	DLH						

DLL 和 DLH 决定了 LIN 模块的波特率。波特率 = SYSclk/16/DL。

SYNC：同步模式，只有在从模式下，才有用。提高波特率误差兼容性。

例如，主机波特率 9600：

从机没有开启同步模式时，可以通信的波特率范围在 9500~9700 附近，误差容忍范围+/-1%左右；

从机开启同步模式后，可以通信的波特率范围在 8300 ~ 10000 附近，误差容忍范围-13.5% ~ +4.16%左右。

39.5.9 LIN 帧头延时计数寄存器（HDRL/HDRH）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HDRL	0x08	HDRL							
HDRH	0x09	HDRH							

延时(ms) = (HDR*1000)/ SYSclk。

39.5.10 LIN 帧头延时分频寄存器（HDP）

符号	地址	B7	B6	B5	B4	B3	B2	B1	B0
HDP	0x0A			HDP					

HDR 和 HDP 共同定义了一个帧头的延时时间，这个时间用于定义软件配置完发送帧头的命令与 LIN 模块实际发送 head 帧头的时间延时。

附录A 指令集

A.1 指令集简介

A.1.1 BINARY 模式和 SOURCE 模式

Binary 模式和 Source 模式是指令集提供操作码的两种方式。根据用户程序，Binary 模式或者 Source 模式可能会生成更高效的代码。Binary 模式是指 MCU51 的标准操作码。Source 模式是指 MCU251 特定的操作码集，它通过额外的操作和寻址模式扩展指令集。特殊助记符 0xA5 用于区分每种模式下的具体指令。所有未使用的操作码均已正确解码并作为 NOP 执行。

A.1.2 指令集标记

指令有五种不同的寻址模式：立即数寻址、直接寻址、寄存器寻址、间接寻址寻址和相对寻址。在立即寻址模式中，操作数包含在操作码中。对于直接寻址，8 位地址或 16 位地址是操作码的一部分；对于寄存器寻址，在操作码中选择一个寄存器用于操作。在间接寻址模式中，在操作码中选择一个寄存器来指向操作使用的地址。相对寻址模式用于跳转指令。下表提供了 Ai8051U 微控制器内核指令集的周期数。一个周期数等于一个系统时钟。表 1 和表 2 包含指令集表中使用的助记符说明。表 3 到表 7 标示了每条指令执行所需的十六进制代码、字节数和系统时钟数。

Rn	当前工作字节寄存器R0、R1、…、R7
N	字节寄存器索引0~7
rrr	n的二进制表示
Rm	字节寄存器R0、R1、…、R15
Rmd	目标寄存器
Rms	源数寄存器
m,md,ms	字节寄存器索引：m、md、ms = 0、1、…、15
ssss	m或者md的二进制表示
SSSS	ms的二进制表示
WR	字寄存器WR0、WR2、…、WR30
WRjd	目标寄存器
WRjs	源数寄存器
@WRj	由字寄存器寻址的间接内存位置（0x0000~0xFFFF）
@WRj+dis	由字寄存器寻址＋0到64KB偏移值的间接内存位置（0x0000~0xFFFF）
j, jd, js	字寄存器索引： j、jd、js = 0、2、…、30
tttt	j或者jd的二进制表示
TTTT	js的二进制表示
DRk	双字寄存器DR0、DR4、…、DR28、DR56、DR60
DRkd	目标寄存器
DRks	源数寄存器
@DRk	由双字寄存器寻址的间接内存位置（0x000000~0xFFFFFF）
@DRk+dis	由双字寄存器寻址＋0到64KB偏移值的间接内存位置（0x000000~0xFFFFFF）

k, kd, ks uuuu UUUU	双字寄存器索引： k、kd、ks ＝0、4、…、28、56、60 k或者kd的二进制表示 ks的二进制表示
dir8	128个内部内存位置，各种特殊功能寄存器
dir16	16位内存地址（0x000000~0x00FFFF）
@Ri	由寄存器R0或者R1寻址的间接内存位置（0x00~0xFF）
#data	8位立即数包含在指令中
#data16	16位立即数包含在指令中的第2和第3字节
#0data16	32位立即数；高位字填充零，低位字包含在指令中的第2和第3字节
#1data16	32位立即数；高位字填充1，低位字包含在指令中的第2和第3字节
#short vv	等于1、2或者4的常数，包含在指令中 #short 的二进制表示
bit	内存位置（0x20~0x7F）或者任何已定义的SFR中的直接寻址位
bit51	存储器或者SFR中的直接寻址位（位号 = 0x00~0xFF）。位0x00~0x7F是内部内存里字节位置为0x20~0x2F的128个位。 位0x80~0xFF是16个SFR中的128个位，其地址以0或者8结尾：0x80、0x88、0x90、…、0xF0、0xF8
A	累加器

表 1、数据寻址模式注释

addr24	24位目标地址可位于16MB地址空间的任意位置。它用于ECALL和EJMP指令。
addr16	LCALL和LJMP的目标地址可以是64KB程序存储器地址空间内的任何位置。
addr11	ACALL和AJMP的目标地址将与下一条指令的第一个字节位于相同的2 KB程序存储器页面内。
rel	SJMP 和所有条件跳转都包含一个8位偏移字节。范围是相对于下一条指令的第一个字节偏移+127到-128字节。

表 2、程序寻址模式注释

-	该指令不修改标志位。
√	该指令根据需要置位或者清零标志位。
1	该指令置位标志位。
0	该指令清零标志位。

表 3、标志位说明注释

A.1.3 指令表（功能排序）

指令执行总时间取决于 WTST（0xe9）的值。下面每个表标示了 WTST=0 时的时钟数。计算每条指令的总时钟数的通用公式是：

指令时钟数 = 时钟数 + nrPRGACS*WTST （nrPRGACS＝0 或 1）

如果指令访问 XDM 存储器，则应将 CKCON[2:0]值乘以 nrXDMACS（1 或者 2）来计算正确的周期数。

指令时钟数 = 时钟数 + nrPRGACS*WTST + nrXDMACS*CKCON

专用于二进制模式的指令标记为红色。这些指令在源代码模式执行时需要在操作码之前加上 0xA5 前缀（ESC）。专用于源代码模式的指令标记为蓝色。当以二进制模式执行时，这些指令在操作码之前需要加上 0xA5 前缀（ESC）。无论 CPU 模式如何，所有其它指令始终可用。

算术运算

助记符	描述	编码	字节	时钟数
ADD A,Rn	将寄存器加到累加器	0x28-0x2F	1	2
ADD A,dir8	将直接字节加到累加器	0x25	2	1
ADD A,@Ri	将间接内存加到累加器	0x26-0x27	1	2
ADD A,#data	将立即数加到累加器	0x24	2	1
ADD Rm,Rm	将字节寄存器加到字节寄存器	0x2C	2	1
ADD WRj,WRj	将字寄存器加到字寄存器	0x2D	2	1
ADD reg,op2(3)	将操作数加到 Rm、WRj 或者 DRk	0x2E	注1	注2
ADD DRk,DRk	将双字寄存器加到双字寄存器	0x2F	2	1
ADDC A,Rn	将寄存器加到带有进位标志位的累加器	0x38-0x3F	1	2
ADDC A,dir8	将直接字节加到带有进位标志的累加器 A	0x35	2	1
ADDC A,@Ri	将间接内存加到带有进位标志的累加器 A	0x36-0x37	1	2
ADDC A,#data	将立即数加到带有进位标志的累加器 A	0x34	2	1
SUBB A,Rn	从 A 借位再减去寄存器	0x98-0x9F	1	2
SUBB A,dir8	从 A 借位再减去直接字节	0x95	2	1
SUBB A,@Ri	从 A 借位再减去间接内存	0x96-0x97	1	2
SUBB A,#data	从 A 借位再减去立即数	0x94	2	1
SUB Rm,Rm	从字节寄存器中减去字节寄存器	0x9C	2	1
SUB WRj,WRj	从字寄存器中减去字寄存器	0x9D	2	1
SUB reg,op2(3)	从 Rm、WRj 或者 DRk 中减去操作数	0x9E	注1	注2
SUB DRk,DRk	从双字寄存器中减去双字寄存器	0x9F	2	1
CMP Rm,Rm	比较两个字节寄存器	0xBC	2	1
CMP WRj,WRj	比较两个字寄存器	0xBD	2	1
CMP reg,op2(3)	将 Rm、WRj 或者 DRk 与操作数进行比较	0xBE	注1	注2
CMP DRk,DRk	比较两个双字寄存器	0xBF	2	1
INC A	递增累加器	0x04	1	1
INC Rn	递增寄存器	0x08-0x0F	1	2

助记符	描述	编码	字节	时钟数
INC dir8	递增直接字节	0x05	2	1
INC @Ri	递增间接内存	0x06-0x07	1	2
INC reg,#short(3)	递增 Rm、WRj 或者 DRk	0x0B	2	注2
DEC A	递减累加器	0x14	1	1
DEC Rn	递减寄存器	0x18-0x1F	1	2
DEC dir8	递减直接字节	0x15	1	1
DEC @Ri	递减间接内存	0x16-0x17	2	2
DEC reg,#short(3)	递减 Rm、WRj 或者 DRk	0x1B	2	注2
INC DPTR	递增数据指针	0xA3	1	1
MUL A,B	将 A 乘以 B	0xA4	1	1
MUL Rm,Rm	字节寄存器相乘	0xAC	2	1
MUL WRj,WRj	字寄存器相乘	0xAD	2	1
DIV A,B	将 A 除以 B	0x84	1	6
DIV Rm,Rm	字节寄存器相除	0x8C	1	6
DIV WRj,WRj	字寄存器相除	0x8D	1	10
DA A	十进制调整累加器	0xD4	1	3

注 1：指令所需的字节数取决于由紧接着的字节确定的寻址模式。请参考指令集详解。

注 2：指令所需的周期取决于由紧接着的字节确定的寻址模式。请参考指令集详解。

注 3：操作数和寻址模式取决于紧接着的字节。指令集详解中描述了所有选项。

逻辑运算

助记符	描述	编码	字节	时钟数
ANL A,Rn	寄存器逻辑与累加器	0x58-0x5F	1	2
ANL A,dir8	直接字节逻辑与累加器	0x55	2	1
ANL A,@Ri	间接内存逻辑与累加器	0x56-0x57	1	2
ANL A,#data	立即数逻辑与累加器	0x54	2	1
ANL dir8,A	直接字节逻辑与累加器	0x52	2	1
ANL dir8,#data	直接数据逻辑与直接字节	0x53	3	1
ANL Rm,Rm	两个字节寄存器逻辑与	0x5C	2	1
ANL WRj,WRj	两个字寄存器逻辑与	0x5D	2	1
ANL reg,op2(3)	操作数逻辑与 Rm、WRj 或者 DRk	0x5E	注1	注2
ORL A,Rn	寄存器逻辑或累加器	0x48-0x4F	1	2
ORL A,dir8	直接字节逻辑或累加器	0x45	2	1
ORL A,@Ri	间接内存逻辑或累加器	0x46-0x47	1	2
ORL A,#data	立即数逻辑或累加器	0x44	2	1
ORL dir8,A	累加器直接字节	0x42	2	1
ORL dir8,#data	立即数逻辑或直接字节	0x43	3	1
ORL Rm,Rm	两个字节寄存器逻辑或	0x4C	2	1
ORL WRj,WRj	两个字寄存器逻辑或	0x4D	2	1

助记符	描述	编码	字节	时钟数
ORL reg,op2(3)	操作数逻辑或 Rm、WRj 或者 DRk	0x4E	注1	注2
XRL A,Rn	寄存器异或累加器	0x68-0x6F	1	2
XRL A,dir8	直接字节异或累加器	0x65	2	1
XRL A,@Ri	间接内存异或累加器	0x66-0x67	1	2
XRL A,#data	立即数异或累加器	0x64	2	1
XRL dir8,A	直接字节异或累加器	0x62	2	1
XRL dir8,#data	立即数到异或直接字节	0x63	3	1
XRL Rm,Rm	两个字节寄存器异或	0x6C	2	1
XRL WRj,WRj	两个字寄存器异或	0x6D	2	1
XRL reg,op2(3)	操作数异或 Rm、WRj 或者 DRk	0x6E	注1	注2
CLR A	累加器清零	0xE4	1	1
CPL A	累加器取反	0xF4	1	1
RL A	累加器向左循环移动	0x23	1	1
RLC A	累加器带进位向左循环移动	0x33	1	1
RR A	累加器向右循环移动	0x03	1	1
RRC A	累加器带进位向右循环移动	0x13	1	1
SRA reg(3)	通过 MSB 右移 Rm 或者 WRj	0x0E	2	1
SRL reg(3)	右移 Rm 或者 WRj	0x1E	2	1
SLL reg(3)	左移 Rm 或者 WRj	0x3E	2	1
SWAP A	累加器内的交换半字节	0xC4	1	1

注 1：指令所需的字节数取决于由紧接着的字节确定的寻址模式。请参考指令集详解。

注 2：指令所需的周期取决于由紧接着的字节确定的寻址模式。请参考指令集详解。

注 3：操作数和寻址模式取决于紧接着的字节。指令集详解中描述了所有选项。

布尔操作

助记符	描述	编码	字节	时钟数
CLR C	进位标志位清零	0xC3	1	1
CLR bit	直接位清零	0xC2	2	1
SETB C	进位标志位置位	0xD3	1	1
SETB bit	直接位置位	0xD2	2	1
CPL C	进位标志位取反	0xB3	1	1
CPL bit	直接位取反	0xB2	2	1
ANL C,bit	直接位逻辑与进位标志位	0x82	2	1
ANL C,/bit	直接位的非逻辑与进位标志位	0xB0	2	1
ORL C,bit	直接位逻辑或进位标志位	0x72	2	1
ORL C,/bit	直接位的非逻辑或进位标志位	0xA0	2	1
MOV C,bit	直接位搬运到进位标志位	0xA2	2	1
MOV bit,C	进位标志位搬运到直接位	0x92	2	1
Bit instr(1)	位指令集（MCU251 特有）	0xA9	3	1

数据传输

助记符	描述	编码	字节	时钟数
MOV A,Rn	将寄存器搬运到累加器	0xE8-0xEF	1	2
MOV A,dir8	将直接字节搬运到累加器	0xE5	2	1
MOV A,@Ri	将间接内存搬运到累加器	0xE6-0xE7	1	2
MOV A,#data	将立即数搬运到累加器	0x74	2	1
MOV Rn,A	将累加器搬运到寄存器	0xF8-0xFF	1	2
MOV Rn,dir8	将直接字节搬运到寄存器	0xA8-0xAF	2	1
MOV Rn,#data	将立即数搬运到寄存器	0x78-0x7F	2	1
MOV dir8,A	将累加器搬运到直接字节	0xF5	2	1
MOV dir8,Rn	将寄存器搬运到直接字节	0x88-0x8F	2	1
MOV dir8,dir8	将直接字节搬运到直接字节	0x85	3	1
MOV dir8,@Ri	将间接内存搬运到直接字节	0x86-0x87	2	2
MOV dir8,#data	将立即数搬运到直接字节	0x75	3	1
MOV @Ri,A	将累加器搬运到间接内存	0xF6-0xF7	1	2
MOV @Ri,dir8	将直接字节搬运到间接内存	0xA6-0xA7	2	2
MOV @Ri,#data	将立即数搬运到间接内存	0x76-0x77	2	2
MOV Rm,Rm	将字节寄存器搬运到字节寄存器	0x7C	2	1
MOV WRj,WRj	将字寄存器搬运到字寄存器	0x7D	2	1
MOV reg,op2(3)	将操作数搬运到 Rm、WRj 或者 DRk	0x7E	注1	注2
MOV DRk,DRk	将双字寄存器搬运到双字寄存器	0x7F	2	1
MOV WRj,@DRk	将间接（24 位）内存搬运到 WRj	0x0B	3	1
MOV @DRk,WRj	将 WRj 搬运到间接（24 位）内存	0x1B	3	1
MOV Rm, @WRj+dis	将 16 位偏移的间接（16 位）内存搬运到 Rm	0x09	4	1
MOV @WRj+dis,Rm	将 Rm 搬运到 16 位偏移的间接（16 位）内存	0x19	4	1
MOV Rm, @DRk+dis	将 16 位偏移的间接（24 位）内存搬运到 Rm	0x29	4	1
MOV @DRk+dis, Rm	将 Rm 搬运到 16 位偏移的间接（24 位）内存	0x39	4	1
MOV WRj,@WRj+dis	将 16 位偏移的间接（16 位）内存搬运到 WRj	0x49	4	1
MOV @WRj+dis,WRj	将 WRj 搬运到 16 位偏移的间接（16 位）内存	0x59	4	1
MOV WRj,@DRk+dis	将 16 位偏移的间接（24 位）内存搬运到 WRj	0x69	4	1
MOV @DRk+dis,WRj	将 WRj 搬运到 16 位偏移的间接（24 位）内存	0x79	4	1
MOV op1,reg(3)	将 Rm、WRj 或者 DRk 搬运到操作数	0x7A	注1	注2
MOVH DRk,#data16(4)	将 16 位立即数搬运到双字寄存器的高位字	0x7A	4	1
MOVZ WRj,Rm	将字节寄存器搬运到零扩展的字寄存器	0x0A	2	1
MOVS WRj,Rm	将字节寄存器搬运到带符号扩展的字寄存器	0x1A	2	1
MOV DPTR,#data16	将 16 位常数加载到活动的 DPTR	0x90	3	1
MOVC A,@A+DPTR	将代码字节搬运到 DPTR 偏移的累加器	0x93	1	4
MOVC A,@A+PC	将代码字节搬运到 PC 偏移的累加器	0x83	1	3
MOVX A,@Ri	将外部存储器（8 位地址）搬运到 A	0xE2-0xE3	1	3

助记符	描述	编码	字节	时钟数
MOVX A,@DPTR	将外部存储器（16 位地址）搬运到 A	0xE0	1	3
MOVX @Ri,A	将 A 搬运到外部存储器（8 位地址）	0xF2-0xF3	1	2
MOVX @DPTR,A	将 A 搬运到外部存储器（16 位地址）	0xF0	1	2
PUSH dir8	将直接字节压入 IDM 堆栈	0xC0	2	1
POP dir8	从 IDM 堆栈中弹出直接字节	0xD0	2	1
PUSH op1(3)	将操作数压入 IDM 堆栈	0xCA	注1	注2
POP op1(3)	从 IDM 堆栈弹出操作数	0xDA	注1	注2
XCH A,Rn	寄存器跟累加器交换	0xC8-0xCF	1	2
XCH A,dir8	直接字节跟累加器交换	0xC5	2	1
XCH A,@Ri	间接内存跟累加器交换	0xC6-0xC7	1	2
XCHD A,@Ri	间接内存的低位半字节跟 A 交换	0xD6-0xD7	1	2

注 1：指令所需的字节数取决于由紧接着的字节确定的寻址模式。请参考指令集详解。

注 2：指令所需的周期取决于由紧接着的字节确定的寻址模式。请参考指令集详解。

注 3：操作数和寻址模式取决于紧接着的字节。指令集详解中描述了所有选项。

注 4：MOVH 指令的操作码首字节与 MOV op1,reg 组相同。指令通过第二个字节的值来区分。

程序跳转

助记符	描述	编码	字节	时钟数
ACALL addr11	绝对子程序调用	0x11-0xF1	2	3
LCALL addr16	长子程序直接调用	0x12	3	3
ECALL addr24	扩展子程序直接调用	0x9A	4	3
ECALL @DRk	扩展子程序间接调用	0x99	2	3
LCALL @WRj	长子程序间接调用	0x99	2	3
RET	子程序返回	0x22	1	3
ERET	扩展子程序返回	0xAA	1	3
RETI	中断返回	0x32	1	3
AJMP addr11	绝对跳转	0x01-0xE1	2	3
LJMP addr16	直接长跳转	0x02	3	3
EJMP addr24	直接扩展跳转	0x8A	4	3
LJMP @WRj	间接长跳转	0x89	2	3
EJMP @DRk	间接扩展跳转	0x89	2	3
SJMP rel	短跳转（相对地址）	0x80	2	3
JMP @A+DPTR	DPTR 偏移的间接跳转	0x73	1	3
JZ rel	如果累加器为零则跳转	0x60	2	1/3
JNZ rel	如果累加器不为零则跳转	0x70	2	1/3
JC rel	如果进位标志位置位则跳转	0x40	2	1/3
JNC rel	如果进位标志位未置位则跳转	0x50	2	1/3
JB bit,rel	如果直接位置位则跳转	0x20	3	1/3
JNB bit,rel	如果直接位未置位则跳转	0x30	3	1/3

助记符	描述	编码	字节	时钟数
JBC bit,rel	如果直接位置位则跳转并清零位	0x10	3	1/3
JSLE rel	如果小于或者等于则跳转（有符号）	0x08	2	1/3
JSG rel	如果大于则跳转（有符号）	0x18	2	1/3
JLE rel	如果小于或者等于则跳转	0x28	2	1/3
JG rel	如果大于则跳转	0x38	2	1/3
JSL rel	如果小于则跳转（有符号）	0x48	2	1/3
JSGE rel	如果大于或者等于则跳转（有符号）	0x58	2	1/3
JE rel	如果相等则跳转	0x68	2	1/3
JNE rel	如果不相等则跳转	0x78	2	1/3
CJNE A,dir8,rel	将直接字节与 A 比较，如果不相等则跳转	0xB5	3	2/3
CJNE A,#data,rel	将立即数与 A 比较，如果不相等则跳转	0xB4	3	1/3
CJNE Rn,#data,rel	将立即数与寄存器比较。如果不相等则跳转	0xB8-0xBF	3	3/4
CJNE @Ri,#data,rel	将立即数与间接内存比较。如果不相等则跳转	0xB6-0xB7	3	3/4
DJNZ Rn,rel	寄存器递减，如果不为零则跳转	0xD8-0xDF	2	3/4
DJNZ dir8,rel	直接字节递减，如果不为零则跳转	0xD5	3	2/3
NOP	无操作	0x00	1	1

注意：未带条件的跳转在 1 个时钟周期内执行。

A.1.4 指令表（机器码排序）

BINARY 模式

操作码	助记符	操作码	助记符	操作码	助记符	操作码	助记符
00 H	NOP	40 H	JC rel	80 H	SJMP rel	C0 H	PUSH direct
01 H	AJMP addr11	41 H	AJMP addr11	81 H	AJMP addr11	C1 H	AJMP addr11
02 H	LJMP addr16	42 H	ORL direct,A	82 H	ANL C,bit	C2 H	CLR bit
03 H	RR A	43 H	ORL direct,#data	83 H	MOVC A,@A+PC	C3 H	CLR C
04 H	INC A	44 H	ORL A,#data	84 H	DIV AB	C4 H	SWAP A
05 H	INC direct	45 H	ORL A,direct	85 H	MOV direct,direct	C5 H	XCH A, direct
06 H	INC @R0	46 H	ORL A,@R0	86 H	MOV direct,@R0	C6 H	XCH A,@R0
07 H	INC @R1	47 H	ORL A,@R1	87 H	MOV direct,@R1	C7 H	XCH A,@R1
08 H	INC R0	48 H	ORL A,R0	88 H	MOV direct,R0	C8 H	XCH A,R0
09 H	INC R1	49 H	ORL A,R1	89 H	MOV direct,R1	C9 H	XCH A,R1
0A H	INC R2	4A H	ORL A,R2	8A H	MOV direct,R2	CA H	XCH A,R2
0B H	INC R3	4B H	ORL A,R3	8B H	MOV direct,R3	CB H	XCH A,R3
0C H	INC R4	4C H	ORL A,R4	8C H	MOV direct,R4	CC H	XCH A,R4
0D H	INC R5	4D H	ORL A,R5	8D H	MOV direct,R5	CD H	XCH A,R5
0E H	INC R6	4E H	ORL A,R6	8E H	MOV direct,R6	CE H	XCH A,R6
0F H	INC R7	4F H	ORL A,R7	8F H	MOV direct,R7	CF H	XCH A,R7
10 H	JBC bit,rel	50 H	JNC rel	90 H	MOV DPTR,#data16	D0 H	POP direct
11 H	ACALL addr11	51 H	ACALL addr11	91 H	ACALL addr11	D1 H	ACALL addr11
12 H	LCALL addr16	52 H	ANL direct,A	92 H	MOV bit,C	D2 H	SETB bit
13 H	RRC A	53 H	ANL direct,#data	93 H	MOVC A,@A+DPTR	D3 H	SETB C
14 H	DEC A	54 H	ANL A,#data	94 H	SUBB A,#data	D4 H	DA A
15 H	DEC direct	55 H	ANL A,direct	95 H	SUBB A,direct	D5 H	DJNZ direct, rel
16 H	DEC @R0	56 H	ANL A,@R0	96 H	SUBB A,@R0	D6 H	XCHD A,@R0
17 H	DEC @R1	57 H	ANL A,@R1	97 H	SUBB A,@R1	D7 H	XCHD A,@R1
18 H	DEC R0	58 H	ANL A,R0	98 H	SUBB A,R0	D8 H	DJNZ R0,rel
19 H	DEC R1	59 H	ANL A,R1	99 H	SUBB A,R1	D9 H	DJNZ R1,rel
1A H	DEC R2	5A H	ANL A,R2	9A H	SUBB A,R2	DA H	DJNZ R2,rel
1B H	DEC R3	5B H	ANL A,R3	9B H	SUBB A,R3	DB H	DJNZ R3,rel
1C H	DEC R4	5C H	ANL A,R4	9C H	SUBB A,R4	DC H	DJNZ R4,rel
1D H	DEC R5	5D H	ANL A,R5	9D H	SUBB A,R5	DD H	DJNZ R5,rel
1E H	DEC R6	5E H	ANL A,R6	9E H	SUBB A,R6	DE H	DJNZ R6,rel
1F H	DEC R7	5F H	ANL A,R7	9F H	SUBB A,R7	DF H	DJNZ R7,rel
20 H	JB bit,rel	60 H	JZ rel	A0 H	ORL C,bit	E0 H	MOVX A,@DPTR
21 H	AJMP addr11	61 H	AJMP addr11	A1 H	AJMP addr11	E1 H	AJMP addr11

操作码	助记符	操作码	助记符	操作码	助记符	操作码	助记符
22 H	RET	62 H	XRL direct,A	A2 H	MOV C,bit	E2 H	MOVX A,@R0
23 H	RL A	63 H	XRL direct,#data	A3 H	INC DPTR	E3 H	MOVX A,@R1
24 H	ADD A,#data	64 H	XRL A,#data	A4 H	MUL AB	E4 H	CLR A
25 H	ADD A,direct	65 H	XRL A,direct	A5 H	ESC	E5 H	MOV A, direct
26 H	ADD A,@R0	66 H	XRL A,@R0	A6 H	MOV @R0,direct	E6 H	MOV A,@R0
27 H	ADD A,@R1	67 H	XRL A,@R1	A7 H	MOV @R1,direct	E7 H	MOV A,@R1
28 H	ADD A,R0	68 H	XRL A,R0	A8 H	MOV R0,direct	E8 H	MOV A,R0
29 H	ADD A,R1	69 H	XRL A,R1	A9 H	MOV R1,direct	E9 H	MOV A,R1
2A H	ADD A,R2	6A H	XRL A,R2	AA H	MOV R2,direct	EA H	MOV A,R2
2B H	ADD A,R3	6B H	XRL A,R3	AB H	MOV R3,direct	EB H	MOV A,R3
2C H	ADD A,R4	6C H	XRL A,R4	AC H	MOV R4,direct	EC H	MOV A,R4
2D H	ADD A,R5	6D H	XRL A,R5	AD H	MOV R5,direct	ED H	MOV A,R5
2E H	ADD A,R6	6E H	XRL A,R6	AE H	MOV R6,direct	EE H	MOV A,R6
2F H	ADD A,R7	6F H	XRL A,R7	AF H	MOV R7,direct	EF H	MOV A,R7
30 H	JNB bit,rel	70 H	JNZ rel	B0 H	ANL C,bit	F0 H	MOVX @DPTR,A
31 H	ACALL addr11	71 H	ACALL addr11	B1 H	ACALL addr11	F1 H	ACALL addr11
32 H	RETI	72 H	ORL C,direct	B2 H	CPL bit	F2 H	MOVX @R0,A
33 H	RLC A	73 H	JMP @A+DPTR	B3 H	CPL C	F3 H	MOVX @R1,A
34 H	ADDC A,#data	74 H	MOV A,#data	B4 H	CJNE A,#data,rel	F4 H	CPL A
35 H	ADDC A,direct	75 H	MOV direct,#data	B5 H	CJNE A,direct,rel	F5 H	MOV direct, A
36 H	ADDC A,@R0	76 H	MOV @R0,#data	B6 H	CJNE @R0,#data,rel	F6 H	MOV @R0,A
37 H	ADDC A,@R1	77 H	MOV @R1,#data	B7 H	CJNE @R1,#data,rel	F7 H	MOV @R1,A
38 H	ADDC A,R0	78 H	MOV R0,#data	B8 H	CJNE R0,#data,rel	F8 H	MOV R0,A
39 H	ADDC A,R1	79 H	MOV R1,#data	B9 H	CJNE R1,#data,rel	F9 H	MOV R1,A
3A H	ADDC A,R2	7A H	MOV R2,#data	BA H	CJNE R2,#data,rel	FA H	MOV R2,A
3B H	ADDC A,R3	7B H	MOV R3,#data	BB H	CJNE R3,#data,rel	FB H	MOV R3,A
3C H	ADDC A,R4	7C H	MOV R4,#data	BC H	CJNE R4,#data,rel	FC H	MOV R4,A
3D H	ADDC A,R5	7D H	MOV R5,#data	BD H	CJNE R5,#data,rel	FD H	MOV R5,A
3E H	ADDC A,R6	7E H	MOV R6,#data	BE H	CJNE R6,#data,rel	FE H	MOV R6,A
3F H	ADDC A,R7	7F H	MOV R7,#data	BF H	CJNE R7,#data,rel	FF H	MOV R7,A

SOURCE 模式

操作码	助记符	操作码	助记符	操作码	助记符	操作码	助记符
00 H	NOP	40 H	JC rel	80 H	SJMP rel	C0 H	PUSH direct
01 H	AJMP addr11	41 H	AJMP addr11	81 H	AJMP addr11	C1 H	AJMP addr11
02 H	LJMP addr16	42 H	ORL direct,A	82 H	ANL C,bit	C2 H	CLR bit
03 H	RR A	43 H	ORL direct,#data	83 H	MOVC A,@A+PC	C3 H	CLR C
04 H	INC A	44 H	ORL A,#data	84 H	DIV AB	C4 H	SWAP A
05 H	INC direct	45 H	ORL A,direct	85 H	MOV direct,direct	C5 H	XCH A, direct
06 H	-	46 H	-	86 H	-	C6 H	-
07 H	-	47 H	-	87 H	-	C7 H	-
08 H	JSLE rel	48 H	JSL rel	88 H	-	C8 H	-
09 H	MOV Rm,@WRj+dis	49 H	MOV WRj,@WRj+dis	89 H	LJMP @WRj EJMP @DRk	C9 H	-
0A H	MOVZ WRj,Rm	4A H	-	8A H	EJMP addr24	CA H	PUSH op1
0B H	INC R,#short MOV WRj,@DRk	4B H	-	8B H	-	CB H	-
0C H	-	4C H	ORL Rm,Rm	8C H	DIV Rm,Rm	CC H	-
0D H	-	4D H	ORL WRj,WRj	8D H	DIV WRj,WRj	CD H	-
0E H	SRA reg	4E H	ORL reg,op2	8E H	-	CE H	-
0F H	-	4F H	-	8F H	-	CF H	-
10 H	JBC bit,rel	50 H	JNC rel	90 H	MOV DPTR,#data16	D0 H	POP direct
11 H	ACALL addr11	51 H	ACALL addr11	91 H	ACALL addr11	D1 H	ACALL addr11
12 H	LCALL addr16	52 H	ANL direct,A	92 H	MOV bit,C	D2 H	SETB bit
13 H	RRC A	53 H	ANL direct,#data	93 H	MOVC A,@A+DPTR	D3 H	SETB C
14 H	DEC A	54 H	ANL A,#data	94 H	SUBB A,#data	D4 H	DA A
15 H	DEC direct	55 H	ANL A,direct	95 H	SUBB A,direct	D5 H	DJNZ direct, rel
16 H	-	56 H	-	96 H	-	D6 H	-
17 H	-	57 H	-	97 H	-	D7 H	-
18 H	JSG rel	58 H	JSGE rel	98 H	-	D8 H	-
19 H	MOV @WRj+dis,Rm	59 H	MOV @WRj+dis,WRj	99 H	LCALL @WRj ECALL @DRk	D9 H	-
1A H	MOVS WRj,Rm	5A H	-	9A H	ECALL addr24	DA H	POP op1
1B H	DEC R,#short MOV @DRk, WRj	5B H	-	9B H	-	DB H	-
1C H	-	5C H	ANL Rm,Rm	9C H	SUB Rm,Rm	DC H	-
1D H	-	5D H	ANL WRj,WRj	9D H	SUB WRj,WRj	DD H	-
1E H	SRL reg	5E H	ANL reg,op2	9E H	SUB reg,op2	DE H	-
1F H	-	5F H	-	9F H	SUB DRk,DRk	DF H	-
20 H	JB bit,rel	60 H	JZ rel	A0 H	ORL C,bit	E0 H	MOVX A,@DPTR

操作码	助记符	操作码	助记符	操作码	助记符	操作码	助记符
21 H	AJMP addr11	61 H	AJMP addr11	A1 H	AJMP addr11	E1 H	AJMP addr11
22 H	RET	62 H	XRL direct,A	A2 H	MOV C,bit	E2 H	MOVX A,@R0
23 H	RL A	63 H	XRL direct,#data	A3 H	INC DPTR	E3 H	MOVX A,@R1
24 H	ADD A,#data	64 H	XRL A,#data	A4 H	MUL AB	E4 H	CLR A
25 H	ADD A,direct	65 H	XRL A,direct	A5 H	ESC	E5 H	MOV A, direct
26 H	-	66 H	-	A6 H	-	E6 H	-
27 H	-	67 H	-	A7 H	-	E7 H	-
28 H	JLE rel	68 H	JE rel	A8 H	-	E8 H	-
29 H	MOV Rm,@DRk+dis	69 H	MOV WRj,@DRk+dis	A9 H	Bit instructions	E9 H	-
2A H	-	6A H	-	AA H	ERET	EA H	-
2B H	-	6B H	-	AB H	-	EB H	-
2C H	ADD Rm,Rm	6C H	XRL Rm,Rm	AC H	MUL Rm,Rm	EC H	-
2D H	ADD WRj,WRj	6D H	XRL WRj,WRj	AD H	MUL WRj,WRj	ED H	-
2E H	ADD reg,op2	6E H	XRL reg,op2	AE H	-	EE H	-
2F H	ADD DRk,DRk	6F H	-	AF H	-	EF H	-
30 H	JNB bit,rel	70 H	JNZ rel	B0 H	ANL C,bit	F0 H	MOVX @DPTR,A
31 H	ACALL addr11	71 H	ACALL addr11	B1 H	ACALL addr11	F1 H	ACALL addr11
32 H	RETI	72 H	ORL C,direct	B2 H	CPL bit	F2 H	MOVX @R0,A
33 H	RLC A	73 H	JMP @A+DPTR	B3 H	CPL C	F3 H	MOVX @R1,A
34 H	ADDC A,#data	74 H	MOV A,#data	B4 H	CJNE A,#data,rel	F4 H	CPL A
35 H	ADDC A,direct	75 H	MOV direct,#data	B5 H	CJNE A,direct,rel	F5 H	MOV direct, A
36 H	-	76 H	-	B6 H	-	F6 H	-
37 H	-	77 H	-	B7 H	-	F7 H	-
38 H	JG rel	78 H	JNE rel	B8 H	-	F8 H	-
39 H	MOV @DRk+dis,Rm	79 H	MOV @DRk+dis,WRj	B9 H	TRAP	F9 H	-
3A H	-	7A H	MOVH DRk,#data16 MOV op1,reg	BA H	-	FA H	-
3B H	-	7B H	-	BB H	-	FB H	-
3C H	-	7C H	MOV Rm,Rm	BC H	CMP Rm,Rm	FC H	-
3D H	-	7D H	MOV WRj,WRj	BD H	CMP WRj,WRj	FD H	-
3E H	SLL reg	7E H	MOV reg,op2	BE H	CMP reg,op2	FE H	-
3F H	-	7F H	MOV DRk,DRk	BF H	CMP DRk,DRk	FF H	-

A.2 指令详解

ACALL <addr11>

功能: 绝对调用

说明: ACALL 无条件调用位于指定地址的子程序。该指令将 PC 递增两次以获得下一条指令的地址，然后将 16 位结果压入堆栈（低字节在前）并增加堆栈指针两次。目标地址是由 PC 递增后的 5 个高位、操作码第 7 到第 5 位以及指令的第二个字节依次串联得到的。因此调用的子程序必须位于 ACALL 之后指令的第一个字节相同的 2K 程序存储器块内开始。不影响标志位。

影响标志位:

CY	AC	OV	N	Z
—	—	—	—	—

Binary 模式	**Source 模式**
指令长度: 2	2
时钟数: 3	3
Hex: 指令码	指令码

[指令码] 11H, 31H, 51H, 71H, 91H, B1H, D1H, F1H

A10	A9	A8	1	0	0	0	1	A7	A6	A5	A4	A3	A2	A1	A0

指令操作: ACALL

(PC)	← (PC) + 2
(SP)	← (SP) + 1
((SP))	← (PC.7:0)
(SP)	← (SP) + 1
((SP))	← (PC.15:8)
(PC.10:0)	← 页地址

ADD <dest>,<src>

功能: 源数加到目标操作数。

说明: 将源操作数加到目标操作数，可以是寄存器或者累加器，将计算结果留在寄存器或者累加器中。如果第 7 位（CY）有进位，则 CY 标志位置位。

如果加了字节变量，并且如果第 3 位有进位（AC），则 AC 标志位置位。对于无符号整数的加法，CY 标志位指示发生了溢出。如果第 6 位进位但是第 7 位没有进位，或者第 7 位有进位但是第 6 位没有进位，则 OV 标志位置位。有符号整数相加时，OV 标志位指示两个正操作数之和出现了负数，或者两个负操作数之和出现了正数。本描述中的第 6 位和第 7 位指的是操作数的最高有效字节（8、16 或者 32 位）。结果会影响 N 和 Z 标志位。源操作数允许的寻址模式是寄存器、直接、寄存器间接和立即数寻址。

ADD A,Rn

指令操作:

(PC)	←(PC) + 1
(A)	←(A) + (Rn)

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 28H – 2FH

0	0	1	0	1	r	r	r

Binary 模式	**Source 模式**
指令长度: 1	2

时钟数: 1 2

Hex: 指令码 [A5]指令码

ADD A,Direct

指令操作: (PC) ←(PC) + 2

(A) ←(A) + (direct)

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 25H

0	0	1	0	0	1	0	1	直接地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

ADD A,@Ri

指令操作: (PC) ←(PC) + 1

(A) ←(A) + ((Ri))

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 26H, 27H

0	0	1	0	0	1	1	i

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

ADD A,#DATA

指令操作: (PC) ←(PC) + 2

(A) ←(A) + #data

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 24H

0	0	1	0	0	1	0	0	立即数

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

ADD Rmd,Rms

指令操作: (PC) ←(PC) + 2

(Rmd) ←(Rms) + (Rmd)

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 2CH

0	0	1	0	1	1	0	0	s	s	s	s	S	S	S	S

	Binary 模式	Source 模式
指令长度:	3	2
时钟数:	2	1
Hex:	[A5]指令码	指令码

ADD WRjd,WRjs

指令操作: (PC) ←(PC) + 2

(WRjd) ←(WRjs) + (WRjd)

影响标志位:

CY	AC	OV	N	Z
✓	-	✓	✓	✓

[指令码] 2DH

0	0	1	0	1	1	0	1	t	t	t	t	T	T	T	T

	Binary 模式	Source 模式
指令长度:	3	2
时钟数:	2	1
Hex:	[A5]指令码	指令码

ADD DRkd,DRks

指令操作: (PC) ←(PC) + 2

(DRkd) ←(DRks) + (DRkd)

影响标志位:

CY	AC	OV	N	Z
✓	-	✓	✓	✓

[指令码] 2FH

0	0	1	0	1	1	1	1	u	u	u	u	U	U	U	U

	Binary 模式	Source 模式
指令长度:	3	2
时钟数:	2	1
Hex:	[A5]指令码	指令码

ADD Rm,#DATA

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) + #data

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 2E(s)0H

0	0	1	0	1	1	1	0	s	s	s	s	0	0	0	0	立即数

	Binary 模式	Source 模式
指令长度:	4	3
时钟数:	2	1
Hex:	[A5]指令码	指令码

ADD WRj,#DATA16

指令操作: (PC) ←(PC) + 4

(WRj) ←(WRj) + #data16

影响标志位:

CY	AC	OV	N	Z
✓	-	✓	✓	✓

[指令码] 2E(t)4H

0	0	1	0	1	1	1	0	t	t	t	t	0	1	0	0	立即数高字节		立即数低字节	

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

ADD DRk,#0DATA16

指令操作: (PC) ←(PC) + 4

(DRk) ←(DRk) + #data16

影响标志位:

CY	AC	OV	N	Z
✓	-	✓	✓	✓

[指令码] 2E(u)8H

0	0	1	0	1	1	1	0	u	u	u	u	1	0	0	0	立即数高字节		立即数低字节	

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

ADD Rm,DIR8

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) + (dir8)

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 2E(s)1H

0	0	1	0	1	1	1	0	s	s	s	s	0	0	0	1	直接地址			

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

ADD WRj,DIR8

指令操作: (PC) ←(PC) + 3

(WRj) ←(WRj) + (dir8)

影响标志位:

CY	AC	OV	N	Z
✓	-	✓	✓	✓

[指令码] 2E(t)5H

0	0	1	0	1	1	1	0	t	t	t	t	0	1	0	1	直接地址			

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 (3 for SFR) 1 (2 for SFR)

Hex: [A5]指令码 指令码

ADD Rm,DIR16

指令操作: (PC) ←(PC) + 4

(Rm) ←(Rm) + (dir16)

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 2E(s)3H

																地址高字节	地址低字节
0	0	1	0	1	1	1	0	s	s	s	s	0	0	1	1		

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

ADD WRj,DIR16

指令操作: (PC) ←(PC) + 4

(WRj) ←(WRj) + (dir16)

影响标志位:

CY	AC	OV	N	Z
✓	-	✓	✓	✓

[指令码] 2E(t)7H

																地址高字节	地址低字节
0	0	1	0	1	1	1	0	t	t	t	t	0	1	1	1		

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

ADD Rm,@WRj

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) + ((WRj))

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 2E(t)9(s)0H

0	0	1	0	1	1	1	0	t	t	t	t	1	0	0	1	s	s	s	s	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 1 1

Hex: [A5]指令码 指令码

ADD Rm,@DRk

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) + ((DRk))

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 2E(u)B(s)0H

0	0	1	0	1	1	1	0	u	u	u	u	1	0	1	1	s	s	s	s	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

ADDC A,<src-byte>

功能: 将 A 和源操作数相加，如果 CY 已置位，则加一（1），并将结果放入 A。

说明: ADDC 同时将指定的字节变量、进位标志位和累加器内容相加，将结果留在累加器中。如果第 7 位或者第 3 位有进位，则分别将进位和辅助进位标志位置位，否则清零。添加无符号整数时，进位标志位指示发生溢出。如果第 6 位有进位但是第 7 位没有进位，或者第 7 位有进位但是第 6 位没有进位，则 OV 置位；否则 OV 清零。有符号整数相加时，OV 表示两个正操作数之和出现了负数，或者两个负操作数之和出现了正数。N 和 Z 标志位也会根据结果受到影响。源操作数允许四种寻址模式：寄存器、直接、寄存器间接或者立即数寻址。

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

ADDC A,Rn

指令操作: (PC) ←(PC) + 1

(A) ←(A) + (C) + (Rn)

[指令码] 38H – 3FH

0 0 1 1	1 r r r
Binary 模式	**Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

ADDC A,DIRECT

指令操作: (PC) ←(PC) + 2

(A) ←(A) + (C) + (direct)

[指令码] 35H

0 0 1 1	0 1 0 1	直接地址
Binary 模式	**Source 模式**	

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

ADDC A,@Ri

指令操作: (PC) ←(PC) + 1

(A) ←(A) + (C) + ((Ri))

[指令码] 36H, 37H

0 0 1 1	0 1 1 i
Binary 模式	**Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

ADDC A,#DATA

指令操作: (PC) ←(PC) + 2

(A) ←(A) + (C) + #data

[指令码] 34H

0 0 1 1	0 1 0 0	立即数

	Binary 模式	Source 模式
指令长度:	2	2
时钟数:	1	1
Hex:	指令码	指令码

AJMP addr11

功能: 绝对跳转

说明: AJMP 将程序执行转移到指定的地址，该地址是在运行时通过串联 PC 的高 5 位（在 PC 递增两次后）、操作码第 7 到第 5 位和指令的第二个字节而形成的。因此，目标必须与 AJMP 之后指令的第一个字节位于相同的 2K 程序存储器块内。

影响标志位:

CY	AC	OV	N	Z
—	—	—	—	—

指令操作: (PC) ←(PC) + 2

(PC10-0) ←页地址

[指令码] 01H, 21H, 41H, 61H, 81H, A1H, C1H, E1H

a10	a9	a8	0	0	0	0	1	a7	a6	a5	a4	a3	a2	a1	a0

	Binary 模式	Source 模式
指令长度:	2	2
时钟数:	3	3
Hex:	指令码	指令码

ANL <dest>,<src>

功能: 字节操作数的逻辑与

说明: 在指定变量之间执行按位逻辑与运算并将结果存储在目标变量中。这两个操作数允许 10 种寻址模式组合。当目标是寄存器或者累加器时，源数可以使用寄存器、直接、寄存器间接或者立即数寻址；当目标是直接地址时，源数可以是累加器或者立即数。N 和 Z 标志位会根据结果受到影响。

注意：当该指令用于修改输出端口时，用作原始端口数据的值将从输出数据锁存器中读取，而不是输入引脚。

ANL A,Rn

指令操作: (PC) ←(PC) + 1

(A) ←(A) and (Rn)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 58H – 5FH

0	1	0	1	1	r	r	r

	Binary 模式	Source 模式
指令长度:	1	2
时钟数:	1	2
Hex:	指令码	[A5]指令码

ANL A,Direct

指令操作: (PC) ←(PC) + 2

(A) ←(A) and (direct)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 55H

0 1 0 1	0 1 0 1	直接地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

ANL A,@Ri

指令操作: (PC) ←(PC) + 1

(A) ←(A) and ((Ri))

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 56H, 57H

0 1 0 1	0 1 1 i

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

ANL A,#DATA

指令操作: (PC) ←(PC) + 2

(A) ←(A) and #data

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 54H

0 1 0 1	0 1 0 0	立即数

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

ANL Direct,A

指令操作: (PC) ←(PC) + 2

(direct) ←(direct) and (A)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 52H

0 1 0 1	0 0 1 0	直接地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

ANL Direct,#DATA

指令操作: (PC) ←(PC) + 3

(direct) ←(direct) and #data

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 53H

0	1	0	1	0	0	1	1	直接地址				立即数			

Binary 模式 **Source 模式**

指令长度: 3 3

时钟数: 1 1

Hex: 指令码 指令码

ANL Rmd,Rms

指令操作: (PC) ←(PC) + 2

(Rmd) ←(Rms) and (Rmd)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 5CH

0	1	0	1	1	1	0	0	s	s	s	s	S	S	S	S

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

ANL WRjd,WRjs

指令操作: (PC) ←(PC) + 2

(WRjd) ←(WRjs) and (WRjd)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 5DH

0	1	0	1	1	1	0	1	t	t	t	t	T	T	T	T

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

ANL Rm,#DATA

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) and #data

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 5E(s)0H

0	1	0	1	1	1	1	0	s	s	s	s	0	0	0	0	立即数

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

ANL WRj,#DATA16

指令操作: (PC) ←(PC) + 4

(WRj) ←(WRj) and #data16

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 5E(t)4H

0	1	0	1	1	1	1	0	t	t	t	t	0	1	0	0	立即数高字节		立即数低字节	

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

ANL Rm,DIR8

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) and (dir8)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 5E(s)1H

0	1	0	1	1	1	1	0	s	s	s	s	0	0	0	1	直接地址			

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

ANL WRj,DIR8

指令操作: (PC) ←(PC) + 3

(WRj) ←(WRj) and (dir8)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 5E(t)5H

0	1	0	1	1	1	1	0	t	t	t	t	0	1	0	1	直接地址			

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 (3 for SFR) 1 (2 for SFR)

Hex: [A5]指令码 指令码

ANL Rm,DIR16

指令操作: (PC) ←(PC) + 4

(Rm) ←(Rm) and (dir16)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 5E(s)3H

0	1	0	1	1	1	1	0	s	s	s	s	0	0	1	1	地址高字节		地址低字节	

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

ANL WRj,DIR16

指令操作: (PC) ←(PC) + 4

(WRj) ←(WRj) and (dir16)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 5E(t)7H

0 1 0 1	1 1 1 0	t t t t	0 1 1 1	地址高字节	地址低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

ANL Rm,@WRj

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) and ((WRj))

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 5E(t)9(s)0H

0 1 0 1	1 1 1 0	t t t t	1 0 0 1	s s s s	0 0 0 0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

ANL Rm,@DRk

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) and ((DRk))

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 5E(u)B(s)0H

0 1 0 1	1 1 1 0	u u u u	1 0 1 1	s s s s	0 0 0 0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

ANL C,<src–bit>

功能: 位操作数的逻辑与

说明: 如果源位的布尔值为逻辑 0，则进位标志位清零；否则进位标志位保持当前状态。汇编语言中操作数前的斜杠（“/”）表示将寻址位的逻辑补码用作源数的值，但源位本身不受影响。不影响其它标志位。只允许直接位寻址作为源操作数。

ANL C,bit51

指令操作: (PC) ←(PC) + 2

(A) ←(C) and (bit51)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	-	-

[指令码] 82H

1 0 0 0	0 0 1 0	位地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

ANL C,/bit51

指令操作: (PC) ←(PC) + 2

(B) ←(C) and /(bit51)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	-	-

[指令码] B0H

1 0 1 1	0 0 0 0	位地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

ANL C,bit

指令操作: (PC) ←(PC) + 3

(A) ←(C) and (bit)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	-	-

[指令码] A98(y)H

1 0 1 0	1 0 0 1	1 0 0 0	0 y y y	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

ANL C,/bit

指令操作: (PC) ←(PC) + 3

(A) ←(C) and /(bit)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	-	-

[指令码] A9F(y)H

1 0 1 0	1 0 0 1	1 1 1 1	0 y y y	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

CJNE <dest-byte>,<src-byte>,rel

功能: 比较，如果不相等就跳转。

说明: CJNE 比较前两个操作数的大小，如果它们的值不相等则跳转。将 PC 加上指令最后一个字节中的有符号相对偏移来计算跳转目标，之后递增 PC 到下一条指令的开头。如果无符号整数 <dest-byte> 的值小于无符号整数 <src-byte> 的值，则进位标志位置位；否则进位清零。两个操作数都不受影响。前两个操作数允许四种寻址模式组合：累加器可以与任何直接寻址的字节或者立即数比较，任何间接内存位置或者工作寄存器都可以与立即数比较。影响 C、N 和 Z 标志位。

CJNE A,Direct,rel

指令操作:

	(PC)	←(PC) + 3
if	(A)	≠ (direct)
then	(PC)	←(PC) + 相对偏移
if	(A)	< (direct)
then	(C)	←1
else	(A)	←0

影响标志位:

CY	AC	OV	N	Z
✓	-	-	✓	✓

[指令码] B5H

1	0	1	1	0	1	0	1	直接地址	相对地址

Binary 模式 **Source 模式**

指令长度: 3 3

时钟数: 2/3 2/3

Hex: 指令码 指令码

CJNE A,#DATA,rel

指令操作:

	(PC)	←(PC) + 3
if	(A)	≠ data
then	(PC)	←(PC) + 相对偏移
if	(A)	< data
then	(C)	←1
else	(C)	←0

影响标志位:

CY	AC	OV	N	Z
✓	-	-	✓	✓

[指令码] B4H

1	0	1	1	0	1	0	0	立即数	相对地址

Binary 模式 **Source 模式**

指令长度: 3 3

时钟数: 1/3 1/3

Hex: 指令码 指令码

CJNE Rn,#DATA,rel

指令操作:

	(PC)	←(PC) + 3
if	(Rn)	≠ data
then	(PC)	←(PC) + 相对偏移
if	(Rn)	< data
then	(C)	←1
else	(C)	←0

影响标志位:

CY	AC	OV	N	Z

✓	-	-	✓	✓

[指令码] B8H - BFH

1	0	1	1	1	r	r	r	立即数	相对地址

Binary 模式 **Source 模式**

指令长度: 3 4

时钟数: 2/3 3/4

Hex: 指令码 [A5]指令码

CJNE @Ri,#DATA,rel

指令操作:

(PC) ←(PC) + 3
if ((Ri)) ≠ data
then (PC) ←(PC) + 相对偏移
if ((Ri)) < data
then (C) ←1
else (C) ←0

影响标志位:

CY	AC	OV	N	Z
✓	-	-	✓	✓

[指令码] B6H, B7H

1	0	1	1	0	1	1	i	立即数	相对地址

Binary 模式 **Source 模式**

指令长度: 3 4

时钟数: 2/3 3/4

Hex: 指令码 [A5]指令码

CLR A

功能: 累计器清零

说明: 累加器被清零（所有的位置位为零）。影响 N 和 Z 标志位。

指令操作:

(PC) ←(PC) + 1
(A) ←0

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] E4H

1	1	1	0	0	1	0	0

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

CLR bit51

功能: 位清零

说明: 指定的位被清零（重置为零）。不影响其它标志位。

指令操作:

(PC) ←(PC) + 2
bit ←0

影响标志位:

CY	AC	OV	N	Z
-	-	-	–	–

[指令码] C2H

1	1	0	0	0	0	1	0	位地址

	Binary 模式	Source 模式
指令长度:	2	2
时钟数:	1	1
Hex:	指令码	指令码

CLR C

功能: 进位清零

说明: 进位标志位被清零（重置为零）。不影响其它标志位。

指令操作: (PC) ←(PC) + 1

(C) ←0

影响标志位:

CY	AC	OV	N	Z
✓	-	-	–	–

[指令码] C3H

1	1	0	0	0	0	1	1

	Binary 模式	Source 模式
指令长度:	1	1
时钟数:	1	1
Hex:	指令码	指令码

CLR bit

功能: 位清零

说明: 指定的位被清零（重置为零）。不影响其它标志位。

指令操作: (PC) ←(PC) + 3

(bit) ←0

影响标志位:

CY	AC	OV	N	Z
-	-	-	–	–

[指令码] A9C(y)H

1	1	0	0	1	0	0	1	1	1	0	0	0	y	y	y	直接地址

	Binary 模式	Source 模式
指令长度:	4	3
时钟数:	2	1
Hex:	[A5]指令码	指令码

CMP <dest>,<src>

功能: 比较

说明: 从目标操作数中减去源操作数。结果不存储在目标操作数中。如果第 7 位需要借位，则 CY（借位）标志位置位；否则清零。当减去有符号整数时，OV 标志位指示正数减去负数出现了负结果，或者指示负数减去正数出现了正结果。本描述中的第 7 位是指操作数的最高有效字节（8、16 或者 32 位）。AC 仅受字节操作数影响。N 和 Z 标志位会根据结果受到影响。源操作数允许四种寻址模式：寄存器、直接、立即数和间接寻址。

CMP Rmd,Rms

指令操作: (PC) ←(PC) + 2

(Rmd) -(Rms)

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] BCH

1 0 1 1	1 1 0 0	s s s s	S S S S

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

CMP WRjd,WRjs

指令操作: (PC) ←(PC) + 2

(WRjd) -(WRjs)

影响标志位:

CY	AC	OV	N	Z
✓	–	✓	✓	✓

[指令码] BDH

1 0 1 1	1 1 0 1	t t t t	T T T T

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

CMP DRkd,DRks

指令操作: (PC) ←(PC) + 2

(DRkd) -(DRks)

影响标志位:

CY	AC	OV	N	Z
✓	–	✓	✓	✓

[指令码] BFH

1 0 1 1	1 1 1 1	u u u u	U U U U

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

CMP Rm,#DATA

指令操作: (PC) ←(PC) + 3

(Rm) -#data

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] BE(s)0H

1 0 1 1	1 1 1 0	s s s s	0 0 0 0	立即数

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

CMP WRj,#DATA16

指令操作: (PC) ←(PC) + 4

(WRj) -#data16

影响标志位:

CY	AC	OV	N	Z
✓	–	✓	✓	✓

[指令码] BE(t)4H

1 0 1 1	1 1 1 0	t t t t	0 1 0 0	立即数高字节	立即数低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

CMP DRk,#0DATA16

指令操作: (PC) ←(PC) + 4

(DRk) -#0data16

影响标志位:

CY	AC	OV	N	Z
✓	–	✓	✓	✓

[指令码] BE(u)8H

1 0 1 1	1 1 1 0	u u u u	1 0 0 0	立即数高字节	立即数低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

CMP DRk,#1DATA16

指令操作: (PC) ←(PC) + 4

(DRk) -#1data16

影响标志位:

CY	AC	OV	N	Z
✓	–	✓	✓	✓

[指令码] BE(u)CH

1 0 1 1	1 1 1 0	u u u u	1 1 0 0	立即数高字节	立即数低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

CMP Rm,Dir8

指令操作: (PC) ←(PC) + 3

(Rm) -(dir8)

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] BE(s)1H

1 0 1 1	1 1 1 0	s s s s	0 0 0 1	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

CMP WRj,Dir8

指令操作: (PC) ←(PC) + 3

(WRj) -(dir8)

影响标志位:

CY	AC	OV	N	Z
✓	–	✓	✓	✓

[指令码] BE(t)5H

1	0	1	1	1	1	1	0	t	t	t	t	0	1	0	1	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2(3 for SFR) 1(2 for SFR)

Hex: [A5]指令码 指令码

CMP Rm,Dir16

指令操作: (PC) ←(PC) + 4

(Rm) -(dir16)

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] BE(s)3H

1	0	1	1	1	1	1	0	s	s	s	s	0	0	1	1	地址高字节	地址低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

CMP WRj,Dir16

指令操作: (PC) ←(PC) + 4

(WRj) -(dir16)

影响标志位:

CY	AC	OV	N	Z
✓	–	✓	✓	✓

[指令码] BE(t)7H

1	0	1	1	1	1	1	0	t	t	t	t	0	1	1	1	地址高字节	地址低字节

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

CMP Rm,@WRj

指令操作: (PC) ←(PC) + 3

(Rm) -((WRj))

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] BE(t)9(s)0H

1	0	1	1	1	1	1	0	t	t	t	t	1	0	0	1	s	s	s	s	0	0	0	0

BE(WRj)9(Rm)0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码　　指令码

CMP Rm,@DRk

指令操作: (PC)　　←(PC) + 3

(Rm)　　-((DRk))

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] BE(u)B(s)0H

1 0 1 1	1 1 1 0	u u u u	1 0 1 1	s s s s	0 0 0 0

BE(DRk)B(Rm)0

Binary 模式　　**Source 模式**

指令长度: 4　　3

时钟数: 2　　1

Hex: [A5]指令码　　指令码

CPL A

功能: 累加器取反

说明: 累加器的每一位都逻辑取反（逐个取反）。先前为 1 的位将改变为 0，反之亦然。只影响 N 和 Z 标志位。

指令操作: (PC)　　←(PC) + 1

(A)　　←/(A)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] F4H

1 1 1 1	0 1 0 0

Binary 模式　　**Source 模式**

指令长度: 1　　1

时钟数: 1　　1

Hex: 指令码　　指令码

CPL Bit51

功能: 位取反

说明: 对指定的位变量取反。之前是 1 的位被改变为零，反之亦然。不影响其它标志位。

注意：当该指令用于修改输出引脚时，用作原始数据的值将从输出数据锁存器中读取，而不是输入引脚。

指令操作: (PC)　　←(PC) + 2

(bit51)　　←/(bit51)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] B2H

1 0 1 1	0 0 1 0	位地址

Binary 模式　　**Source 模式**

指令长度: 2　　2

时钟数: 1　　1

Hex: 指令码　　指令码

CPL C

功能: 进位取反

说明: 对进位标志位取反。之前是 1 的位被改变为零，反之亦然。

指令操作: (PC) ←(PC) + 1

(C) ←/(C)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	-	-

[指令码] B3H

1 0 1 1	0 0 1 1
Binary 模式	**Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

CPL Bit

功能: 位取反

说明: 对指定的位取反。

指令操作: (PC) ←(PC) + 3

(bit) ←/(bit)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] A9B(y)H

1 0 1 0	1 0 0 1	1 0 1 1	0 y y y	直接地址
Binary 模式			**Source 模式**	

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

DA A

功能: 十进制加法后调整累加器

说明: DA A 调整累加器中的 8 位值，该值由之前两个变量（每个变量为压缩 BCD 格式）相加生成，生成两个 4 位数字。任何 ADD 或者 ADDC 指令都可能用来执行加法。如果累加器第 3 到 0 位大于 9（xxxx1010-xxxx1111），或者如果 AC 标志位为 1，则将累加器加 6 使得低位半字节生成正确的 BCD 数字。如果低四位域的进位通过所有高位传送，则此内部加法将置位进位标志位，否则却不会清零进位标志位。如果现在置位了进位标志位，或者如果现在四个高位超过九（1010xxxx-1111xxxx），这些高位将加六，在高半字节中生成正确的 BCD 数字。同样，如果高位进位，将会置位进位标志位，但不会清零进位。因此，进位标志位指示原始的两个 BCD 变量的和是否大于 100，从而允许多种准确的十进制加法。OV 不受影响。所有这些都发生在一个指令周期内。本质上，该指令通过将累加器加 00H、06H、60H 或者 66H 来执行十进制转换，具体取决于初始累加器和 PSW 条件。影响 C、N 和 Z 标志位。

注意：DA A 不能简单地将累加器中的十六进制数转换为 BCD 表示法，DA A 也不适用于十进制减法。

指令操作:

(PC)←(PC) + 3

if [[(A3-0) > 9] ^ [(AC) = 1]]

then (A3-0) ← (A3-0) + 6

next if [[(A7-4) > 9] ^ [(C) = 1]]

then (A7-4) ← (A7-4) + 6

影响标志位:

CY	AC	OV	N	Z
✓	-	-	✓	✓

[指令码] B4H

1	0	1	1	0	1	0	0

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 3 3

Hex: 指令码 指令码

DEC byte

功能: 递减

说明: 指定的变量减 1。00H 的原始值将下溢到 0FFH。只影响 N 和 Z 标志位。允许使用以下寻址模式：累加器、寄存器、直接或者寄存器间接寻址。

注意：当该指令用于修改输出端口时，用作原始端口数据的值将从输出数据锁存器中读取，而不是输入引脚。

DEC A

指令操作: (PC) ←(PC) + 1

(A) ←(A) - 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 14H

0	0	0	1	0	1	0	0

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

DEC Rn

指令操作: (PC) ←(PC) + 1

(Rn) ←(Rn) - 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 18H - 1FH

0	0	0	1	1	r	r	r

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

DEC Direct

指令操作: (PC) ←(PC) + 2

(direct) ←(direct) - 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 15H

0	0	0	1	0	1	0	1	直接地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

DEC @Ri

指令操作: (PC) ←(PC) + 1

((Ri)) ←((Ri)) - 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 16H, 17H

0	0	0	1	0	1	1	i

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

DEC <dest>,<src>

功能: 目标值递减

说明: 将目标操作数所指定的变量递减 1、2 或者 4。原始值 00H 将下溢至 0FFH。影响 N 和 Z 标志位。递减的值编码如下：

vv	值
00	1
01	2
10	4

DEC Rm,#short

指令操作: (PC) ←(PC) + 2

(Rm) ←(Rm) - #short

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 1B(s)(00v)H

0	0	0	1	1	0	1	1	s	s	s	s	0	0	v	v

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

DEC WRj,#short

指令操作: (PC) ←(PC) + 2

(WRj) ←(WRj) - #short

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 1B(t)(01v)H

0	0	0	1	1	0	1	1	t	t	t	t	0	1	v	v

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

DEC DRk,#short

指令操作: (PC) ←(PC) + 2

(DRk) ←(DRk) - #short

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 1B(u)(11v)H

0	0	0	1	1	0	1	1	u	u	u	u	1	1	v	v

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

DIV AB

说明: DIV AB 将累加器中的无符号八位整数除以寄存器 B 中的无符号八位整数。累加器保存商的整数部分，寄存器 B 保存整数余数。进位和 OV 标志位将被清零。N 和 Z 标志位也会根据结果受到影响。

例外：如果 B 原来放入 00H，则返回到累加器和 B 寄存器的值为未定义的。此外，溢出标志位置位。进位标志位在任何情况下都会被清零。其它标志位未定义。

指令操作: (PC) ←(PC) + 1

(A15-8) ← (A) / (B) –结果的位 15..8

(A7-0) ← (A) / (B) –结果的位 7..0

影响标志位:

CY	AC	OV	N	Z
0	-	✓*	✓*	✓*

*) –如果 B 原来放入 00H，则 OV=1，N 和 Z 未定义

[指令码] 84H

1	0	0	0	0	1	0	0

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 6 6

Hex: 指令码 指令码

DIV <dest>,<src>

说明: 在寄存器寻址模式下将寄存器中的无符号整数除以无符号整数操作数，并清零 CY 和 OV 标志位。对于字节操作数（<dest>, <src> = Rmd,Rms），结果为 16 位。8 位的商存储在 Rmd 所在字的高字节中，8 位的余数存储在 Rmd 所在字的低字节中。例如：寄存器 1 放入 251 （0FBH），寄存器 5 放入 18（12H）。执行 DIV R1 指令后，R5 寄存器 1 放入 13（0DH 或者 00001101B）；寄存器 0 放入 17（11H 或者 00010001B），因为 251= (13×18) + 17； 同时 CY 和 OV 位被清零。CY 标志位被清零。如果商的最高有效位已置位，则 N 标志位置位。如果商为零，则 Z 标志位置位。

例外：如果 <src> 放入 00H，则两个操作数中返回的值都是未定义的；CY 标志位被清零， OV

标志位置位，其余标志位未定义。

DIV Rmd,Rms

指令操作:

(PC)	←(PC) + 2	
(Rmd)	余数(Rmd) / (Rms)	if <dest> md = 0,2,4,..,14
(Rmd+1)	商(Rmd) / (Rms)	
(Rmd-1)	余数(Rmd) / (Rms)	if <dest> md = 1,3,5,..,15
(Rmd)	商(Rmd) / (Rms)	

影响标志位:

CY	AC	OV	N	Z
0	-	✓*	✓*	✓*

*) –如果源数原来放入 0，则 OV=1，N 和 Z 未定义

[指令码] 8CH

1	0	0	0	1	1	0	0	s	s	s	s	S	S	S	S

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 7 6

Hex: [A5]指令码 指令码

DIV WRjd,WRjs

指令操作:

(PC)	←(PC) + 2	
(WRjd)	余数(WRjd) / (WRjs)	if <dest> jd = 0,4,8,..,28
(WRjd+1)	商(WRjd) / (WRjs)	
(WRjd-1)	余数(WRjd) / (WRjs)	if <dest> jd = 2,6,10,..,30
(WRjd)	商(WRjd) / (WRjs)	

对于字操作数（<dest>,<src> = WRjd,WRjs），16 位的商在 WR(jd+2)中，16 位的余数在 WRjd 中。例如，对于目标寄存器 WR4，假设商为 1122H，余数为 3344H。然后，将结果存储在这些寄存器文件位置：(4)->0x33、(5)->0x44、(6)->0x11、(7)->0x22。

影响标志位:

CY	AC	OV	N	Z
0	-	✓*	✓*	✓*

*) –如果源数原来放入 0，则 OV=1，N 和 Z 未定义

[指令码] 8DH

1	0	0	0	1	1	0	1	t	t	t	t	T	T	T	T

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 11 10

Hex: [A5]指令码 指令码

DJNZ <byte>,<rel–addr>

功能: 递减，如果不为零则跳转

说明: DJNZ 将指定位置的值递减 1，如果结果值不为零，则跳转到第二个操作数指定的地址。 00H 的原始值将下溢至 0FFH。只影响 N 和 Z 标志位。将 PC 加上指令最后一个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增到下一条指令的第一个字节。递减的位置可以是寄存器或者直接寻址的字节。

注意：当该指令用于修改输出端口时，用作原始端口数据的值将从输出数据锁存器中读取，而不是输入引脚。

DJNZ Rn,rel

指令操作:

(PC)←(PC) + 2
(Rn)←(Rn) - 1
if (Rn) ≠ 0
then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] D8H - DFH

1 1 0 1	1 r r r	相对地址

Binary 模式 **Source 模式**

指令长度: 2 3

时钟数: 2/3 3/4

Hex: 指令码 [A5]指令码

DJNZ Direct,rel

指令操作:

(PC)←(PC) + 3
(direct)←(direct) - 1
if (direct) ≠ 0
then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] D5H

1 1 0 1	0 1 0 1	直接地址	相对地址

Binary 模式 **Source 模式**

指令长度: 3 4

时钟数: 2/3 2/3

Hex: 指令码 [A5]指令码

ECALL <dest>

功能: 扩展调用

说明: 调用位于指定地址的子程序。该指令将程序计数器加 4 以生成下一条指令的地址，然后将 24 位结果压入堆栈（高字节在前），堆栈指针增加 3。然后将 PC 的 8 位高字和 16 位低字分别装载到 ECALL 指令的第二、第三和第四字节。程序继续执行该地址处的指令。因此，子程序可以在整个 16MB 存储空间中的任何位置开始。不影响标志位。

ECALL addr24

指令操作:

(PC) ←(PC) + 4
(SP) ←(SP) + 1
((SP)) ←(PC.23:16)
(SP) ←(SP) + 1
((SP)) ←(PC.15:8)
(SP) ←(SP) + 1
((SP)) ←(PC.7:0)
(PC) ←(addr.23:0)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 9AH

1	0	0	1	1	0	1	0	addr 23-16	addr 15-8	addr 7-0

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 4 3

Hex: [A5]指令码 指令码

ECALL @DRk

指令操作:

(PC) ←(PC) + 4

(SP) ←(SP) + 1

((SP)) ←(PC.23:16)

(SP) ←(SP) + 1

((SP)) ←(PC.15:8)

(SP) ←(SP) + 1

((SP)) ←(PC.7:0)

(PC) ←(DRk)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 99(u)8H

1	0	0	1	1	0	0	1	u	u	u	u	1	0	0	0

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 4 3

Hex: [A5]指令码 指令码

EJMP <dest>

功能: 扩展跳转

说明: 通过把指令的第二、第三和第四个字节加载到 PC 的 8 位高位字和 16 位低位字，进行无条件跳转到指定地址。因此，目标可以在整个 16MB 内存空间中的任何位置。不影响标志位。

EJMP addr24

指令操作: (PC) ←(addr.23:0)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 8AH

1	0	0	0	1	0	1	0	addr 23-16	addr 15-8	addr 7-0

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 4 3

Hex: [A5]指令码 指令码

EJMP @DRk

指令操作: (PC) ←(DRk)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 89(u)8H

1 0 0 0	1 0 0 1	u u u u	1 0 0 0

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 4 3

Hex: [A5]指令码 指令码

ESC

功能: 切换到相反的模式

说明: 以相反的模式继续执行紧接着的指令。除了 PC 之外，没有寄存器或者标志位受到影响。

指令操作: (PC) ←(PC) + 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] A5H

1 0 1 0	0 1 0 1

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

ERET

功能: 扩展返回

说明: 依次从堆栈中弹出 PC 的第 3 字节、第 2 字节、第 1 字节和第 0 字节，并将堆栈指针减 3。程序在结果地址处继续执行，通常是马上紧接着 ECALL 之后的指令。不影响标志位。

指令操作: (PC.7:0) ¨((SP))

(SP) ¨(SP) - 1

(PC.15:8) ¨((SP))

(SP) ¨(SP) - 1

(PC.23:16) ←((SP))

(SP) ←(SP) - 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] AAH

1 0 1 0	1 0 1 0

Binary 模式 **Source 模式**

指令长度: 2 1

时钟数: 4 3

Hex: [A5]指令码 指令码

INC byte

功能: 递增

说明: INC 将指定的变量加 1。原来值 0FFh 将溢出到 00h。只影响 N 和 Z 标志位。允许三种寻址模式：寄存器、直接或者寄存器间接寻址。

注意：当该指令用于修改输出端口时，用作原始端口数据的值将从输出数据锁存器中读取，而不是输入引脚。

INC A

指令操作: (PC) ←(PC) + 1

(A) ←(A) + 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 04H

0 0 0 0	0 1 0 0

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

INC Rn

指令操作: (PC) ←(PC) + 1

(Rn) ←(Rn) + 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 08H - 0FH

0 0 0 0	1 r r r

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

INC Direct

指令操作: (PC) ←(PC) + 1

(direct) ←(direct) + 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 05H

0 0 0 0	0 1 0 1	直接地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

INC @Ri

指令操作: (PC) ←(PC) + 1

((Ri)) ←((Ri)) + 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 06H, 07H

0 0 0 0	0 1 1 i

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

INC <dest>,<src>

功能: 递增

说明: 将指定变量递增 1、2 或者 4。原来值 0FFH 溢出到 00H。只影响 N 和 Z 标志位。递增值的编码如下：

vv	值
00	1
01	2
10	4

INC Rm,#short

指令操作: (PC) ←(PC) + 2

(Rm) ←(Rm) + #short

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 0B(s)(00v)H

0 0 0 0	1 0 1 1	s s s s	0 0 v v

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

INC WRj,#short

指令操作: (PC) ←(PC) + 2

(WRj) ←(WRj) + #short

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 0B(t)(01v)H

0 0 0 0	1 0 1 1	t t t t	0 1 v v

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

INC DRk,#short

指令操作: (PC) ←(PC) + 2

(DRk) ←(DRk) + #short

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 0B(u)(11v)H

0 0 0 0	1 0 1 1	u u u u	1 1 v v

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

INC DPTR

功能: 递增数据指针

说明: 将 16 位的数据指针加 1。执行 16 位递增（模 2^16）； 数据指针的低字节（DPL）从 0FFH 溢出到 00H 时会向高位字节（DPH）递增。高字节（DPH）溢出不会增加扩展数据指针的高字（DPX = DR56）。只影响 N 和 Z 标志位。

指令操作: (PC) ←(PC) + 1

(DPTR) ←(DPTR) + 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] A3H

1 0 1 0	0 0 1 1

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

JB

功能: 如果置位则跳转

说明: 如果指定的位为 1，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第三个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增到下一条指令的第一个字节。不修改被检测的位。不影响标志位。

JB Bit51,rel

指令操作: (PC)←(PC) + 3

if (bit51) = 1

then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 20H

0 0 1 0	0 0 0 0	位地址	相对地址

Binary 模式 **Source 模式**

指令长度: 3 3

时钟数: 1/3 1/3

Hex: 指令码 指令码

JB Bit,rel

指令操作: (PC)←(PC) + 3

if (bit) = 1

then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] A92(y)H

1 0 1 0	1 0 0 1	0 0 1 0	0 y y y	直接地址	相对地址

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2/5 1/3

Hex: [A5]指令码　　指令码

JBC

功能: 如果置位则跳转并清零位

说明: 如果指定的位为 1，则跳转到指定的地址；否则继续下一条指令。任何一种情况，都清零指定位。将 PC 加上指令第三个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增到下一条指令的第一个字节。不影响标志位。

注意：当该指令用于测试输出引脚时，用作原始数据的值将从输出数据锁存器中读取，而不是输入引脚。

JBC Bit51,rel

指令操作:

(PC)←(PC) + 3

if (bit51) = 1

then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 10H

0	0	0	1	0	0	0	0	位地址	相对地址

Binary 模式　　**Source 模式**

指令长度: 3　　3

时钟数: 1/3　　1/3

Hex: 指令码　　指令码

JBC Bit,rel

指令操作:

(PC)←(PC) + 3

if (bit) = 1

then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] A91(y)H

1	0	1	0	1	0	0	1	0	0	0	1	0	y	y	y	直接地址	相对地址

Binary 模式　　**Source 模式**

指令长度: 5　　4

时钟数: 2/5　　1/3

Hex: [A5]指令码　　指令码

JC

功能: 如果置位了进位则跳转

说明: 如果进位标志位被置位，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第二个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增两次。不修改被检测的位。不影响标志位。

指令操作:

(PC)←(PC) + 2

if (C) = 1

then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 40H

0 1 0 0	0 0 0 0	相对地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1/3 1/3

Hex: 指令码 指令码

JE

功能: 如果相等则跳转

说明: 如果 Z 标志位已置位，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第二个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增两次。

指令操作:

(PC)←(PC) + 2

if (Z) = 1

then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 68H

0 1 1 0	1 0 0 0	相对地址

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2/5 1/3

Hex: [A5]指令码 指令码

JG

功能: 如果大于则跳转

说明: 如果 Z 标志位和 CY 标志位都清零，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第二个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增两次。

指令操作:

(PC)←(PC) + 2

if (Z=0 and C=0) = 1

then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 38H

0 0 1 1	1 0 0 0	相对地址

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2/5 1/3

Hex: [A5]指令码 指令码

JLE

功能: 如果小于等于则跳转

说明: 如果 Z 标志位和 CY 标志位都已置位，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第二个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增两次。

指令操作:

(PC)←(PC) + 2

if (Z=1 and C=1) = 1

then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 28H

0	0	1	0	1	0	0	0	相对地址

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2/5 1/3

Hex: [A5]指令码 指令码

JMP @A+DPTR

功能: 间接跳转

说明: 把累加器的 8 位无符号内容和 16 位数据指针相加，并将结果和加载到程序计数器。这将是后续指令提取的地址。执行 16 位数加法（模 2^16）：低 8 位的进位传送到高位。累加器和数据指针都没有改变。不影响标志位。

指令操作: (PC) ←(A) + (DPTR)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 73H

0	1	1	1	0	0	1	1

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 4 4

Hex: 指令码 指令码

JNB

功能: 如果位未置位则跳转

说明: 如果指定的位为零，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第三个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增到下一条指令的第一个字节。不修改被检测的位。不影响标志位。

JNB Bit51,rel

指令操作: (PC)←(PC) + 3

if (bit51) = 0

then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 30H

0	0	1	1	0	0	0	0	位地址	相对地址

Binary 模式 **Source 模式**

指令长度: 3 3

时钟数: 1/3 1/3

Hex: 指令码 指令码

JNB Bit,rel

指令操作: (PC)←(PC) + 3

if (bit) = 0

then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] A93(y)H

1	0	1	0	1	0	0	1	0	0	1	1	0	y	y	y	直接地址	相对地址

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2/5 1/3

Hex: [A5]指令码 指令码

JNC

功能: 如果进位未置位则跳转

说明: 如果进位标志位为零，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第二个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增到下一条指令。

指令操作: (PC)←(PC) + 2

if (C) = 0

then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 50H

0	1	0	1	0	0	0	0	相对地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1/3 1/3

Hex: 指令码 指令码

JNE

功能: 如果不相等则跳转

说明: 如果 Z 标志位清零，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第二个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增两次。

指令操作: (PC)←(PC) + 2

if (Z) = 0

then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 78H

0	1	1	1	1	0	0	0	相对地址

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2/5 1/3

Hex: [A5]指令码 指令码

JNZ

功能: 如果累加器不为零则跳转

说明: 如果累加器的任何一位为 1，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第二个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增两次。

指令操作: (PC)←(PC) + 2
if (A) ≠ 0
then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 70H

0 1 1 1	0 0 0 0	相对地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1/3 1/3

Hex: 指令码 指令码

JSG

功能: 如果大于则跳转（有符号）

说明: 如果 Z 标志位清零并且 N 标志位和 OV 标志位具有相同的值，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第二个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增两次。

指令操作: (PC)←(PC) + 2
if (Z=0 and N=OV)
then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 18H

0 0 0 1	1 0 0 0	相对地址

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2/5 1/3

Hex: [A5]指令码 指令码

JSGE

功能: 大于等于跳转（有符号）

说明: 如果 N 标志位和 OV 标志位具有相同的值，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第二个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增两次。

指令操作: (PC)←(PC) + 2
if (N=OV)
then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 58H

0 1 0 1	1 0 0 0	相对地址

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2/5 1/3

Hex: [A5]指令码 指令码

JSL

功能: 如果小于则跳转（有符号）

说明: 如果 N 标志位和 OV 标志位具有不同的值，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第二个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增两次。

指令操作:

(PC)←(PC) + 2
if (N≠OV)
then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 48H

0 1 0 0	1 0 0 0	相对地址

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2/5 1/3

Hex: [A5]指令码 指令码

JSLE

功能: 如果小于等于则跳转（有符号）

说明: 如果 Z 标志位已置位或者如果 N 标志位和 OV 标志位具有不同的值，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第二个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增两次。

指令操作:

(PC)←(PC) + 2
if (Z=1 or (N≠OV))
then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 08H

0 0 0 0	1 0 0 0	相对地址

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2/5 1/3

Hex: [A5]指令码 指令码

JZ

功能: 如果累加器为零则跳转

说明: 如果累加器的所有位都为零，则跳转到指定的地址；否则继续下一条指令。将 PC 加上指令第二个字节中的有符号相对偏移来计算跳转目标，之后将 PC 递增两次。不修改累加器。 不影响标志位。

指令操作:

(PC)←(PC) + 2
if (A) = 0
then (PC)←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 60H

0 1 1 0	0 0 0 0	相对地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数:	1/3	1/3
Hex:	指令码	指令码

LCALL

功能: 长调用

说明: 调用位于指定地址的子程序。该指令将程序计数器加 3 以生成下一条指令的地址，然后将 16 位结果压入堆栈（先是低字节），堆栈指针加 2。然后将 PC 的高位和低位字节分别装载到 LCALL 指令的第二和第三字节。程序继续执行该地址处的指令。因此，子程序可以在整个 64KB 程序存储器地址空间中的任何位置开始。不影响标志位。

LCALL addr16

指令操作:

(PC) ←(PC) + 3
(SP) ←(SP) + 1
((SP)) ←(PC.7:0)
(SP) ←(SP) + 1
((SP)) ←(PC.15:8)
(PC) ←(addr.15:0)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 12H

0 0 0 1	0 0 1 0	地址高字节	地址低字节

Binary 模式	**Source 模式**
指令长度:	3	3
时钟数:	3	3
Hex:	指令码	指令码

LCALL @WRj

指令操作:

(PC) ←(PC) + 3
(SP) ←(SP) + 1
((SP)) ←(PC.7:0)
(SP) ←(SP) + 1
((SP)) ←(PC.15:8)
(PC) ←(WRj)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 99(t)4H

1 0 0 1	1 0 0 1	t t t t	0 1 0 0

Binary 模式	**Source 模式**
指令长度:	3	2
时钟数:	4	3
Hex:	[A5]指令码	指令码

LJMP

功能: 长跳转

说明: LJMP 通过把指令的第二和第三个字节分别加载到 PC 高位和低位字节，进行无条件跳转到指定地址。因此，目标可以在整个 64KB 程序存储器地址空间中的任何位置。不影响标志位。

LJMP addr16

指令操作: (PC) ←(addr.15:0)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 02H

0 0 0 0	0 0 1 0	地址高字节	地址低字节

Binary 模式 **Source 模式**

指令长度: 3 3

时钟数: 3 3

Hex: 指令码 指令码

LJMP @WRj

指令操作: (PC) ←(WRj)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 89(t)4H

1 0 0 0	1 0 0 1	t t t t	0 1 0 0

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 4 3

Hex: [A5]指令码 指令码

MOV

功能: 搬运变量

说明: 第二个操作数指定的变量被复制到第一个操作数指定的位置。源数字节不受影响。没有其它寄存器或者标志位受到影响。这是迄今为止最灵活的操作。允许源数和目标寻址模式的 24 种组合。

MOV A,Rn

指令操作: (PC) ←(PC) + 1

(A) ←(Rn)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] E8H - EFH

1 1 1 0	1 r r r

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

MOV A,Direct

指令操作: (PC) ←(PC) + 2

(A) ←(direct)

注：MOV A, ACC 是有效指令。

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] E5H

1 1 1 0	0 1 0 1	直接地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

MOV A,@Ri

指令操作: (PC) ←(PC) + 1

(A) ←((Ri))

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] E6H, E7H

1 1 1 0	0 1 1 i

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

MOV A,#DATA

指令操作: (PC) ←(PC) + 2

(A) ←#data

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 74H

0 1 1 1	0 1 0 0	立即数

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

MOV Rn,A

指令操作: (PC) ←(PC) + 1

(Rn) ←(A)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] F8H - FFH

1 1 1 1	1 r r r

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

MOV Rn,Direct

指令操作: (PC) ←(PC) + 2

(Rn) ←(direct)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] A8H - AFH

1 0 1 0	1 r r r	直接地址

Binary 模式 **Source 模式**

指令长度: 2 3

时钟数: 1 1

Hex: 指令码 [A5]指令码

MOV Rn,#DATA

指令操作: (PC) ←(PC) + 2

(Rn) ←#data

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 78H - 7FH

0 1 1 1	1 r r r	立即数

Binary 模式 **Source 模式**

指令长度: 2 3

时钟数: 1 1

Hex: 指令码 [A5]指令码

MOV Direct,A

指令操作: (PC) ←(PC) + 2

(direct) ←(A)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] F5H

1 1 1 1	0 1 0 1	直接地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

MOV Direct,Rn

指令操作: (PC) ←(PC) + 2

(direct) ←(Rn)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 88H - 8FH

1 0 0 0	1 r r r	直接地址

Binary 模式 **Source 模式**

指令长度: 2 3

时钟数: 1 1

Hex: 指令码 [A5]指令码

MOV Direct,Direct

指令操作: (PC) ←(PC) + 3

(direct) ←(direct)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 85H

1	0	0	0	0	1	0	1	源直接地址	目标直接地址

Binary 模式 **Source 模式**

指令长度: 3 3

时钟数: 1 1

Hex: 指令码 指令码

MOV Direct,@Ri

指令操作: (PC) ←(PC) + 2

(direct) ←((Ri))

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 86H, 87H

1	0	0	0	0	1	1	i	直接地址

Binary 模式 **Source 模式**

指令长度: 2 3

时钟数: 1 2

Hex: 指令码 [A5]指令码

MOV Direct,#DATA

指令操作: (PC) ←(PC) + 2

(direct) ←#data

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 75H

0	1	1	1	0	1	0	1	直接地址	立即数

Binary 模式 **Source 模式**

指令长度: 3 3

时钟数: 1 1

Hex: 指令码 指令码

MOV @Ri,A

指令操作: (PC) ←(PC) + 1

((Ri)) ←(A)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] F6H, F7H

1	1	1	1	0	1	1	i

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

MOV @Ri,Direct

指令操作: (PC) ←(PC) + 2

((Ri)) ←(direct)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] A6H, A7H

1 0 1 0	0 1 1 i	直接地址

Binary 模式 **Source 模式**

指令长度: 2 3

时钟数: 1 2

Hex: 指令码 [A5]指令码

MOV @Ri,#DATA

指令操作: (PC) ←(PC) + 2

((Ri)) ←#data

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 76H, 77H

0 1 1 1	0 1 1 i	立即数

Binary 模式 **Source 模式**

指令长度: 2 3

时钟数: 1 2

Hex: 指令码 [A5]指令码

MOV Rmd,Rms

指令操作: (PC) ←(PC) + 2

(Rmd) ←(Rms)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7CH

0 1 1 1	1 1 0 0	s s s s	S S S S

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV WRjd,WRjs

指令操作: (PC) ←(PC) + 2

(WRjd) ←(WRjs)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7DH

0 1 1 1	1 1 0 1	t t t t	T T T T

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV DRkd,DRks

指令操作: (PC) ←(PC) + 2

(DRkd) ←(DRks)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7FH

0	1	1	1	1	1	1	1	u	u	u	u	U	U	U	U

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV Rm,#DATA

指令操作: (PC) ←(PC) + 3

(Rm) ←#data

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7E(s)0H

0	1	1	1	1	1	1	0	s	s	s	s	0	0	0	0	立即数

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV WRj,#DATA16

指令操作: (PC) ←(PC) + 4

(WRj) ←#data16

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7E(t)4H

0	1	1	1	1	1	1	0	t	t	t	t	0	1	0	0	立即数高字节	立即数低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV DRk,#0DATA16

指令操作: (PC) ←(PC) + 4

(DRk) ←#0data16

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7E(u)8H

0	1	1	1	1	1	1	0	u	u	u	u	1	0	0	0	立即数高字节	立即数低字节

	Binary 模式	Source 模式
指令长度:	5	4
时钟数:	2	1
Hex:	[A5]指令码	指令码

MOV DRk,#1DATA16

指令操作: (PC) ←(PC) + 4

(DRk) ←#1data16

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7E(u)CH

0	1	1	1	1	1	1	0	u	u	u	u	1	1	0	0	立即数高字节	立即数低字节

	Binary 模式	Source 模式
指令长度:	5	4
时钟数:	2	1
Hex:	[A5]指令码	指令码

MOV Rm,Dir8

指令操作: (PC) ←(PC) + 3

(Rm) ←(dir8)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7E(s)1H

0	1	1	1	1	1	1	0	s	s	s	s	0	0	0	1	直接地址	

	Binary 模式	Source 模式
指令长度:	4	3
时钟数:	2	1
Hex:	[A5]指令码	指令码

MOV WRj,Dir8

指令操作: (PC) ←(PC) + 3

(WRj) ←(dir8)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7E(t)5H

0	1	1	1	1	1	1	0	t	t	t	t	0	1	0	1	直接地址	

	Binary 模式	Source 模式
指令长度:	4	3
时钟数:	2(3 for SFR)	1(2 for SFR)
Hex:	[A5]指令码	指令码

MOV DRk,Dir8

指令操作: (PC) ←(PC) + 3

(DRk) ←(dir8)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7E(u)DH

0 1 1 1	1 1 1 0	u u u u	1 1 0 1	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2(5 for SFR) 1(4 for SFR)

Hex: [A5]指令码 指令码

MOV Rm,Dir16

指令操作: (PC) ←(PC) + 4

(Rm) ←(dir16)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7E(s)3H

0 1 1 1	1 1 1 0	s s s s	0 0 1 1	地址高字节	地址低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV WRj,Dir16

指令操作: (PC) ←(PC) + 4

(WRj) ←(dir16)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7E(t)7H

0 1 1 1	1 1 1 0	t t t t	0 1 1 1	地址高字节	地址低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV DRk,Dir16

指令操作: (PC) ←(PC) + 4

(DRk) ←(dir16)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7E(u)FH

0 1 1 1	1 1 1 0	u u u u	1 1 1 1	地址高字节	地址低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV Rm,@WRj

指令操作: (PC) ←(PC) + 3

(Rm) ←((WRj))

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7E(t)9(s)0H

0	1	1	1	1	1	1	0	t	t	t	t	1	0	0	1	s	s	s	s	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV Rm,@DRk

指令操作: (PC) ←(PC) + 3

(Rm) ←((DRk))

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7E(u)B(s)0H

0	1	1	1	1	1	1	0	u	u	u	u	1	0	1	1	s	s	s	s	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV WRjd,@WRjs

指令操作: (PC) ←(PC) + 3

(WRjd) ←((WRjs))

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 0B(u)8(s)0H

0	0	0	0	1	0	1	1	u	u	u	u	1	0	0	0	s	s	s	s	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV WRj,@DRk

指令操作: (PC) ←(PC) + 3

(WRj) ←((DRk))

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 0B(u)A(t)0H

0	0	0	0	1	0	1	1	u	u	u	u	1	0	1	0	t	t	t	t	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV Dir8,Rm

指令操作: (PC) ←(PC) + 3

(dir8) ←(Rm)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7A(s)1H

0	1	1	1	1	0	1	0	s	s	s	s	0	0	0	1	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV Dir8,WRj

指令操作: (PC) ←(PC) + 3

(dir8) ←(WRj)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7A(t)5H

0	1	1	1	1	0	1	0	t	t	t	t	0	1	0	1	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV Dir8,DRk

指令操作: (PC) ←(PC) + 3

(dir8) ←(DRk)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7A(u)DH

0	1	1	1	1	0	1	0	u	u	u	u	1	1	0	1	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV Dir16,Rm

指令操作: (PC) ←(PC) + 4

(dir16) ←(Rm)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7A(s)3H

0	1	1	1	1	0	1	0	s	s	s	s	0	0	1	1	地址高字节	地址低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV Dir16,WRj

指令操作: (PC) ←(PC) + 4

(dir16) ←(WRj)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7A(t)7H

														地址高字节	地址低字节
0	1	1	1	1	0	1	0	t	t	t	t	0	1	1	1		

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV Dir16,DRk

指令操作: (PC) ←(PC) + 4

(dir16) ←(DRk)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7A(u)FH

														地址高字节	地址低字节
0	1	1	1	1	0	1	0	u	u	u	u	1	1	1	1		

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV @WRj,Rm

指令操作: (PC) ←(PC) + 3

((WRj)) ←(Rm)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7A(t)9(s)0H

0	1	1	1	1	0	1	0	t	t	t	t	1	0	0	1	s	s	s	s	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV @DRk,Rm

指令操作: (PC) ←(PC) + 3

((DRk)) ←(Rm)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7A(u)B(s)0H

0	1	1	1	1	0	1	0	u	u	u	u	1	0	1	1	s	s	s	s	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV @WRjd,WRjs

指令操作: (PC) ←(PC) + 3

((WRjd)) ←(WRjs)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 1B(t)8(T)0H

0	0	0	1	1	0	1	1	t	t	t	t	1	0	0	0	T	T	T	T	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV @DRk,WRj

指令操作: (PC) ←(PC) + 3

((DRk)) ←(WRj)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 1B(u)A(t)0H

0	0	0	1	1	0	1	1	u	u	u	u	1	0	1	0	t	t	t	t	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV Rm,@WRj+dis

指令操作: (PC) ←(PC) + 4

(Rm) ←((WRj)+dis)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 09H

																偏移高字节	偏移低字节
0	0	0	0	1	0	0	1	s	s	s	s	t	t	t	t		

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV WRjd,@WRjs+dis

指令操作: (PC) ←(PC) + 4

(WRjd) ←((WRjs)+dis)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 49H

																偏移高字节	偏移低字节
0	1	0	0	1	0	0	1	t	t	t	t	T	T	T	T		

	Binary 模式	Source 模式
指令长度:	5	4
时钟数:	2	1
Hex:	[A5]指令码	指令码

MOV Rm,@DRk+dis

指令操作: (PC) ←(PC) + 4

(Rm) ←((DRk)+dis)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 29H

0 0 1 0	1 0 0 1	s s s s	u u u u	偏移高字节	偏移低字节

	Binary 模式	Source 模式
指令长度:	5	4
时钟数:	2	1
Hex:	[A5]指令码	指令码

MOV WRj,@DRk+dis

指令操作: (PC) ←(PC) + 4

(WRj) ←((DRk)+dis)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 69H

0 1 1 0	1 0 0 1	t t t t	u u u u	偏移高字节	偏移低字节

	Binary 模式	Source 模式
指令长度:	5	4
时钟数:	2	1
Hex:	[A5]指令码	指令码

MOV @WRj+dis,Rm

指令操作: (PC) ←(PC) + 4

((WRj)+dis) ←(Rm)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 19H

0 0 0 1	1 0 0 1	s s s s	t t t t	偏移高字节	偏移低字节

	Binary 模式	Source 模式
指令长度:	5	4
时钟数:	2	1
Hex:	[A5]指令码	指令码

MOV @WRjd+dis,WRjs

指令操作: (PC) ←(PC) + 4

((WRjd)+dis) ←(WRjs)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 59H

0	1	0	1	1	0	0	1	T	T	T	T	t	t	t	t	偏移高字节		偏移低字节	

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV @DRk+dis,Rm

指令操作: (PC) ←(PC) + 4

((DRk)+dis) ←(Rm)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 39H

0	0	1	1	1	0	0	1	s	s	s	s	u	u	u	u	偏移高字节		偏移低字节	

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV @DRk+dis,WRj

指令操作: (PC) ←(PC) + 4

((DRk)+dis) ←(WRj)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 79H

0	1	1	1	1	0	0	1	t	t	t	t	u	u	u	u	偏移高字节		偏移低字节	

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV <dest–bit>,<src–bit>

功能: 搬运位数据

说明: 第二个操作数（可直接寻址的位）指定的布尔变量被复制到进位标志位中。没有其它寄存器或者标志位受到影响。

MOV C,Bit51

指令操作: (PC) ←(PC) + 2

(C) ←(bit51)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	-	-

[指令码] A2H

1	0	1	0	0	0	1	0	位地址											

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

MOV C,Bit

指令操作: (PC) ←(PC) + 3

(C) ←(bit)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	-	-

[指令码] A9A(y)H

1 0 1 0	1 0 0 1	1 0 1 0	y y y y	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV Bit51,C

指令操作: (PC) ←(PC) + 2

(bit51) ←(C)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 92H

1 0 0 1	0 0 1 0	位地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

MOV Bit,C

指令操作: (PC) ←(PC) + 3

(bit) ←(C)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	-	-

[指令码] A99(y)H

1 0 1 0	1 0 0 1	1 0 0 1	y y y y	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

MOV DPTR,#DATA16

功能: 数据指针加载 16 位常数

说明: 数据指针加载指定的 16 位常数。16 位常数加载到指令的第二个和第三个字节中。第二个字节（DPH）是高位字节，而第三个字节（DPL）是低位字节。不影响标志位。

指令操作: (PC) ←(PC) + 3

DPH ←立即数 15..8

DPL ←立即数 7..0

影响标志位:

CY	AC	OV	N	Z

-	-	-	-	-

[指令码] 90H

1 0 0 1	0 0 0 0	立即数高字节	立即数低字节

Binary 模式 **Source 模式**

指令长度: 3 3

时钟数: 1 1

Hex: 指令码 指令码

MOVC

功能: 搬运代码字节

说明: MOVC 指令将代码字节或者程序存储器中的常数加载到累加器中。取出的字节地址是原始的无符号 8 位累加器内容以及 16 位基址寄存器的内容之和，该基址寄存器可以是数据指针也可以是 PC。在后一种情况下，PC 会递增到下一条指令的地址再加上累加器；否则，基址寄存器不会改变。执行 16 位加法，因此低 8 位的进位可以传送到高位。不影响标志位。

MOVC A,@A+DPTR

指令操作: (PC) ←(PC) + 1

(A) ←((A) + (DPTR))

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 93H

1 0 0 1	0 0 1 1

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 4 4

Hex: 指令码 指令码

MOVC A,@A+PC

指令操作: (PC) ←(PC) + 1

(A) ←((A) + (PC))

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 83H

1 0 0 0	0 0 1 1

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 3 3

Hex: 指令码 指令码

MOVH DRk,#DATA16

功能: 将 16 位立即数搬运到 dword（双字）寄存器的高位字。

说明: 将 16 位立即数搬运到双字（32 位）寄存器的高位字。双字寄存器的低位字不变。

指令操作: (PC) ←(PC) + 4

(DRk).31-16 ←#data16

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 7A(u)CH

0	1	1	1	1	0	1	0	u	u	u	u	1	1	0	0	立即数高字节		立即数低字节	

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

MOVS WRj,Rm

功能: 将 8 位寄存器搬运到有符号扩展的 16 位寄存器

说明: 将 8 位寄存器的内容搬运到 16 位寄存器的低字节。16 位寄存器的高字节用符号扩展填充，符号是从 8 位源数寄存器的最高有效位获得的。

指令操作: (PC) ←(PC) + 2

(WRj).7-0 ←(Rm)

(WRj).15-8 ←符号扩展

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 1AH

0	0	0	1	1	0	1	0	t	t	t	t	s	s	s	s

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

MOVX

功能: 搬运外部的

说明: MOVX 指令在累加器和外部数据存储器的一个字节之间传输数据，因此 MOV 后附加 X。 有两种类型的指令，不同之处在于它们向外部数据内存提供 8 位还是 16 位间接地址。在第一种类型中，当前寄存器组 R0 或者 R1 中的内容提供了一个 8 位地址，在第二种 MOVX 指令的类型中，数据指针生成了一个 16 位地址。

MOVX A,@Ri

指令操作: (PC) ←(PC) + 1

(A) ←((Ri))

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] E2H, E3H

1	1	1	0	0	0	1	i

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 3* 3*

Hex: 指令码 指令码

MOVX A,@DPTR

指令操作: (PC) ←(PC) + 1

(A) ←((DPTR))

影响标志位:

CY	AC	OV	N	Z

	-	-	-	-	-

[指令码] E0H

1	1	1	0	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 3* 3*

Hex: 指令码 指令码

MOVX @Ri,A

指令操作: (PC) ←(PC) + 1

((Ri)) ←(A)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] F2H, F3H

1	1	1	1	0	0	1	i

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 2* 2*

Hex: 指令码 指令码

MOVX @DPTR,A

指令操作: (PC) ←(PC) + 1

((DPTR)) ←(A)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] F0H

1	1	1	1	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 2* 2*

Hex: 指令码 指令码

MOVZ WRj,Rm

功能: 将 8 位寄存器搬运到 16 位寄存器并扩展零

说明: 将 8 位寄存器的内容搬运到 16 位寄存器的低字节。16 位寄存器的高字节用零填充。

指令操作: (PC) ←(PC) + 2

(WRj).7-0 ←(Rm)

(WRj).15-8 ←0

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 0AH

0	0	0	0	1	0	1	0	t	t	t	t	s	s	s	s

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

MUL

功能: 相乘

说明: 将源数寄存器中的无符号整数与目标寄存器中的无符号整数相乘。只允许寄存器寻址。对于 8 位操作数，结果为 16 位。结果的最高有效字节存储在目标寄存器所在字的低字节中。最低有效字节存储在紧接着的字节寄存器中。如果乘积大于 255(0FFH)，则 OV 标志位置位；否则将被清零。对于 16 位操作数，结果为 32 位。最高有效字存储在目标寄存器所在的双字的低位字中。最低有效字存储在紧接着的字寄存器中。在此操作中，如果乘积大于 0FFFFH，则 OV 标志位置位，否则清零。CY 标志位总是被清零。当置位结果的最高有效字节时 N 标志位置位。当结果为零时 Z 标志位置位。

MUL AB

指令操作: (PC) ←(PC) + 1

(A) ←(A) × (B) -结果位 7..0

(B) ←(A) × (B) -结果位 15..8

影响标志位:

CY	AC	OV	N	Z
0	-	✓	✓	✓

[指令码] A4H

1 0 1 0	0 1 0 0
Binary 模式	**Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

MUL Rmd,Rms

指令操作: (PC)←(PC) + 2

if <dest>md = 0,2,4..,14

Rmd←Rmd × Rms 的高字节

Rmd+1←Rmd × Rms 的低字节

if <dest>md = 1,3,5..,15

Rmd-1←Rmd × Rms 的高字节

Rmd←Rmd × Rms 的低字节

影响标志位:

CY	AC	OV	N	Z
0	-	✓	✓	✓

[指令码] ACH

1 0 1 0	1 1 0 0	s s s s	S S S S
Binary 模式		**Source 模式**	

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

MUL WRjd,WRjs

指令操作: (PC)←(PC) + 2

if <dest>jd = 0,4,8..,28

WRjd←WRjd × WRjs 的高字节

WRjd+2←WRjd × WRjs 的低字节

if <dest>jd = 2,6,10..,30

WRjd-2←WRjd × WRjs 的高字节

WRjd←WRjd × WRjs 的低字节

影响标志位:

CY	AC	OV	N	Z
0	-	✓	✓	✓

[指令码] ADH

1	0	1	0	1	1	0	1	t	t	t	T	T	T	T	T

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

NOP

功能: 无操作

说明: 继续执行接下来的指令。除了 PC 之外，不影响任何寄存器或者标志位。

指令操作: (PC) ←(PC) + 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 00H

0	0	0	0	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

ORL

功能: 变量的逻辑或

说明: 在指定变量之间执行按位逻辑或运算，将结果存储在目标操作数中。目标操作数可以是寄存器、累加器或者直接地址。这两个操作数允许 12 种寻址模式组合。当目标是累加器时，源数可以是寄存器、直接、寄存器间接或者立即寻址；当目标是直接地址时，源数可以是累加器或者立即数。当目标是寄存器时，源数可以是寄存器、立即、直接和间接寻址。只影响 N 和 Z 标志位。注意：当该指令用于修改输出端口时，用作原始端口数据的值将从输出数据锁存器中读取，而不是输入引脚。

ORL A,Rn

指令操作: (PC) ←(PC) + 1

(A) ←(A) or (Rn)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 48H - 4FH

0	1	0	0	1	r	r	r

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

ORL A,Dir

指令操作: (PC) ←(PC) + 2

(A) ←(A) or (direct)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 45H

0 1 0 0	0 1 0 1	直接地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

ORL A,@Ri

指令操作: (PC) ←(PC) + 1

(A) ←(A) or ((Ri))

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 46H, 47H

0 1 0 0	0 1 1 i

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

ORL A,#DATA

指令操作: (PC) ←(PC) + 1

(A) ←(A) or #data

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 44H

0 1 0 0	0 1 0 0	立即数

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

ORL Dir,A

指令操作: (PC) ←(PC) + 1

(direct) ←(direct) or (A)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 42H

0 1 0 0	0 0 1 0	直接地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

ORL Dir,#DATA

指令操作: (PC) ←(PC) + 1

(direct) ←(direct) or #data

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 43H

0	1	0	0	0	0	1	1	直接地址		立即数	

Binary 模式 **Source 模式**

指令长度: 3 3

时钟数: 1 1

Hex: 指令码 指令码

ORL Rmd,Rms

指令操作: (PC) ←(PC) + 2

(Rmd) ←(Rms) or (Rmd)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 4CH

0	1	0	0	1	1	0	0	s	s	s	s	S	S	S	S

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

ORL WRjd,WRjs

指令操作: (PC) ←(PC) + 2

(WRjd) ←(WRjs) or (WRjd)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 4DH

0	1	0	0	1	1	0	1	t	t	t	t		S	S	S

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

ORL Rm,#DATA

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) or #data

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 4E(s)0H

0	1	0	0	1	1	1	0	s	s	s	s	0	0	0	0	立即数

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

ORL WRj,#DATA16

指令操作: (PC) ←(PC) + 4

(WRj) ←(WRj) or #data16

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 4E(t)4H

0 1 0 0	1 1 1 0	t t t t	0 1 0 0	立即数高字节	立即数低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

ORL Rm,Dir8

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) or (dir8)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 4E(s)1H

0 1 0 0	1 1 1 0	s s s s	0 0 0 1	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

ORL WRj,Dir8

指令操作: (PC) ←(PC) + 3

(WRj) ←(WRj) or (dir8)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 4E(t)5H

0 1 0 0	1 1 1 0	t t t t	0 1 0 1	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2(3 for SFR) 1(2 for SFR)

Hex: [A5]指令码 指令码

ORL Rm,Dir16

指令操作: (PC) ←(PC) + 4

(Rm) ←(Rm) or (dir16)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 4E(s)3H

---

0 1 0 0	1 1 1 0	s s s s	0 0 1 1	地址高字节	地址低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

ORL WRj,Dir16

指令操作: (PC) ←(PC) + 4

(WRj) ←(WRj) or (dir16)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 4E(t)7H

0 1 0 0	1 1 1 0	t t t t	0 1 1 1	地址高字节	地址低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

ORL Rm,@WRj

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) or ((WRj))

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 4E(t)9(s)0H

0 1 0 0	1 1 1 0	t t t t	1 0 0 1	s s s s	0 0 0 0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

ORL Rm,@DRk

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) or ((DRk))

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 4E(u)B(s)0H

0 1 0 0	1 1 1 0	u u u U	1 0 1 1	s s s s	0 0 0 0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

ORL CY,<src–bit>

功能: 位变量的逻辑或

说明: 如果布尔值为逻辑 1，则进位标志位置位；否则进位保持在当前状态。汇编语言中操作数前的斜杠（“/”）表示将寻址位的逻辑补码用作源数的值，但源位本身不受影响。不影响其它标志位。

ORL C,Bit51

指令操作: (PC) ←(PC) + 2

(C) ←(C) or (bit51)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	-	-

[指令码] 72H

0	1	1	1	0	0	1	0	位地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

ORL C,/Bit51

指令操作: (PC) ←(PC) + 2

(C) ←(C) or /(bit51)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	-	-

[指令码] A0H

1	0	1	0	0	0	0	0	位地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

ORL C,Bit

指令操作: (PC) ←(PC) + 2

(C) ←(C) or (bit)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	-	-

[指令码] A97(y)H

1	0	1	0	1	0	0	1	0	1	1	1	0	y	y	y	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

ORL C,/Bit

指令操作: (PC) ←(PC) + 2

(C) ←(C) or /(bit)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	-	-

[指令码] A9E(y)H

1	0	1	0	1	0	0	1	1	1	1	0	0	y	y	y	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

POP

功能: 弹出堆栈

说明: 读取由堆栈指针寻址的片上内存位置的内容，然后将堆栈指针减 1。在之前内存位置读取的值被传输到新寻址的位置，该值可以是 8 位或者 16 位。不影响标志位。

POP Dir8

指令操作: (PC) ←(PC) + 2

(dir8) ←((SP))

(SP) ←(SP) - 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] D0H

1 1 0 1	0 0 0 0	直接地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

POP Rm

指令操作: (PC) ←(PC) + 2

(Rm) ←((SP))

(SP) ←(SP) - 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] DA(s)8H

1 1 0 1	1 0 1 0	s s s s	1 0 0 0

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

POP WRj

指令操作: (PC) ←(PC) + 2

(WRj) ←((SP))

(SP) ←(SP) - 2

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] DA(t)9H

1 1 0 1	1 0 1 0	t t t t	1 0 0 1

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

POP DRk

指令操作: (PC) ←(PC) + 2
(DRk) ←((SP))
(SP) ←(SP) - 3

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] DA(u)BH

1 1 0 1	1 0 1 0	u u u u	1 0 1 1

Binary 模式 **Source 模式**

指令长度: 3 2
时钟数: 2 1
Hex: [A5]指令码 指令码

PUSH

功能: 压入堆栈
说明: 将堆栈指针加 1。然后将指定变量的内容复制到堆栈指针寻址的片上内存位置。不影响标志位。

PUSH Dir8

指令操作: (PC) ←(PC) + 2
(SP) ←(SP) + 1
((SP)) ←(dir8)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] C0H

1 1 0 0	0 0 0 0	直接地址

Binary 模式 **Source 模式**

指令长度: 2 2
时钟数: 1 1
Hex: 指令码 指令码

PUSH #DATA

指令操作: (PC) ←(PC) + 2
(SP) ←(SP) + 1
((SP)) ←#data

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] CA02H

1 1 0 0	1 0 1 0	0 0 0 0	0 0 1 0	立即数

Binary 模式 **Source 模式**

指令长度: 4 3
时钟数: 2 1
Hex: [A5]指令码 指令码

PUSH #DATA16

指令操作: (PC) ←(PC) + 2

(SP) ←(SP) + 1
((SP)) ←MSB #data
(SP) ←(SP) + 1
((SP)) ←LSB #data

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] CA06H

1 1 0 0	1 0 1 0	0 0 0 0	0 1 1 0	立即数高字节	立即数低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

PUSH Rm

指令操作: (PC) ←(PC) + 2
(SP) ←(SP) + 1
((SP)) ←(Rm)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] CA(s)8H

1 1 0 0	1 0 1 0	s s s s	1 0 0 0

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

PUSH WRj

指令操作: (PC) ←(PC) + 2
(SP) ←(SP) + 1
((SP)) ←(WRj)
(SP) ←(SP) + 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] CA(t)9H

1 1 0 0	1 0 1 0	t t t t	1 0 0 1

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

PUSH DRk

指令操作: (PC) ←(PC) + 2
(SP) ←(SP) + 1
((SP)) ←(DRk)
(SP) ←(SP) + 3

影响标志位:

CY	AC	OV	N	Z

-	-	-	-	-

[指令码] CA(u)BH

1 1 0 0	1 0 1 0	u u u u	1 0 1 1

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

RET

功能: 从子程序返回

说明: RET 依次从堆栈中弹出 PC 的高位和低位字节，堆栈指针减 2。程序在结果地址处继续执行，通常是紧跟在 ACALL 或者 LCALL 之后的指令。不影响标志位。

指令操作:

(PC15-8) ←((SP))

(SP) ←(SP) - 1

(PC7-0) ←((SP))

(SP) ←(SP) - 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 22H

0 0 1 0	0 0 1 0

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 3 3

Hex: 指令码 指令码

RETI

功能: 从中断返回

说明: 该指令从堆栈中弹出两个或者四个字节，具体取决于 CONFIG1 寄存器中的 INTR 位。如果 INTR = 0，RETI 将 PC 的高字节和低字节依次从堆栈中弹出，并用作 FF:区域的 16 位返回地址。堆栈指针减 2。不影响其它寄存器，PSW 和 PSW1 都不会自动恢复到中断前的状态。如果 INTR = 1，RETI 从堆栈中弹出四个字节：PSW1 和 PC 的三个字节。PC 的三个字节是返回地址，它可以是 16MB 内存空间中的任何位置。堆栈指针减四。PSW1 恢复到中断前状态，但 PSW 没有恢复到中断前状态。不影响其它寄存器。对于 INTR 的任意值，硬件都会恢复中断逻辑以接收跟刚处理的中断具有相同优先级的其它中断。程序在返回地址继续执行，该地址通常是检测到中断请求之后的指令。如果有相同或者较低优先级的中断在等待时执行了 RETI 指令，则在处理等待中断之前先执行该条指令。

指令操作:

INTR=1:		INTR=0:	
(PC15-8)	←((SP))	(PC15-8)	←((SP))
(SP)	←(SP) - 1	(SP)	←(SP) - 1
(PC7-0)	←((SP))	(PC7-0)	←((SP))
(SP)	←(SP) - 1	(SP)	←(SP) - 1
(PC23-16)	←((SP))		
(SP)	←(SP) - 1		
PSW1	←((SP))		
(SP)	←(SP) - 1		

影响标志位:

CY	AC	OV	N	Z

-	-	-	-	-

[指令码] 32H

0	0	1	1	0	0	1	0

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 3 3

Hex: 指令码 指令码

RL

功能: 累加器循环左移

说明: 累加器中的八位向左循环移动一位。第 7 位循环到第 0 位的位置。只影响 N 和 Z 标志位。

指令操作: (PC) ←(PC) + 1

(An + 1) ←(An) n = 0-6

(A0) ←(A7)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 23H

0	0	1	0	0	0	1	1

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

RLC

功能: 带进位标志位累加器循环左移

说明: 累加器中的八位和进位标志位一起向左循环移动一位。第 7 位移入进位标志位；进位标志位的之前状态移入第 0 位的位置。N 和 Z 标志位也会受到影响。

指令操作: (PC) ←(PC) + 1

(An + 1) ←(An) n = 0-6

(A0) ←(C)

(C) ←(A7)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	✓	✓

[指令码] 33H

0	0	1	1	0	0	1	1

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

RR

功能: 累加器循环右移

说明: 累加器中的八位向右循环移动一位。第 0 位循环到第 7 位的位置。只影响 N 和 Z 标志位。

指令操作: (PC) ←(PC) + 1

(An) ←(An + 1) n = 0-6

(A7) ←(A0)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 03H

0	0	0	0	0	0	1	1

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

RRC

功能: 带进位标志位累加器循环右移

说明: 累加器中的八位和进位标志位一起向右循环移动一位。第 0 位移入进位标志位；进位标志位的之前状态移入第 7 位的位置。N 和 Z 标志位也会受到影响。

指令操作: (PC) ←(PC) + 1

(An) ←(An + 1) n = 0-6

(A7) ←(C)

(C) ←(A0)

影响标志位:

CY	AC	OV	N	Z
✓	-	-	✓	✓

[指令码] 13H

0	0	0	1	0	0	1	1

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

SETB

功能: 置位

说明: SETB 将指定位置位为 1。SETB 可以对进位标志位或者任何可直接寻址的位进行操作。不影响其它标志位。

SETB C

指令操作: (PC) ←(PC) + 1

(C) ←1

影响标志位:

CY	AC	OV	N	Z
✓	-	-	-	-

[指令码] D3H

1	1	0	1	0	0	1	1

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

SETB Bit51

指令操作: (PC) ←(PC) + 2

(bit51) ←1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] D2H

1 1 0 1	0 0 1 0	位地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

SETB Bit

指令操作: (PC) ←(PC) + 3

(bit) ←1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] A9D(y)H

1 0 1 0	1 0 0 1	1 1 0 1	0 y y y	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

SJMP

功能: 短跳转

说明: 程序控制无条件地跳转到指定的地址。将 PC 加上指令第二个字节中的有符号相对偏移来计算跳转目标，然后 PC 递增两次。因此，允许的目标的范围是从该指令之前的 128 个字节到它之后的 127 个字节之间。不影响标志位。

指令操作: (PC) ←(PC) + 2

(PC) ←(PC) + rel

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] 80H

1 0 0 0	0 0 0 0	相对地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 3 3

Hex: 指令码 指令码

SLL

功能: 逻辑左移 1 位

说明: 将指定变量左移 1 位，将最低有效位替换为零。移出的位最高有效位（MSB）存储在 CY 位中。N 和 Z 标志位也受到影响。

SLL Rm

指令操作: (PC) ←(PC) + 2

(Rm).a + 1 ←(Rm).a

(Rm).0 ←0

CY ←(Rm).7

影响标志位:

CY	AC	OV	N	Z
✓	-	-	✓	✓

[指令码] 3E(s)0H

0 0 1 1	1 1 1 0	s s s s	0 0 0 0

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

SLL WRj

指令操作: (PC) ←(PC) + 2

(WRj).b + 1 ←(WRj).b

(WRj).0 ←0

CY ←(WRj).15

影响标志位:

CY	AC	OV	N	Z
✓	-	-	✓	✓

[指令码] 3E(t)4H

0 0 1 1	1 1 1 0	t t t t	0 1 0 0

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

SRA

功能: 算术右移 1 位（有符号）

说明: 将指定的变量进行算术右移 1 位。最高有效位不变。移出的位（LSB）存储在 CY 位中。N 和 Z 标志位也受到影响。

SRA Rm

指令操作: (PC) ←(PC) + 2

(Rm).7 ←(Rm).7

(Rm).a ←(Rm).a+1

CY ←(Rm).0

影响标志位:

CY	AC	OV	N	Z
✓	-	-	✓	✓

[指令码] 0E(s)0H

0 0 0 0	1 1 1 0	s s s s	0 0 0 0

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

SRA WRj

指令操作: (PC) ←(PC) + 2

(WRj).15 ←(WRj).15

(WRj).b ←(WRj).b + 1

CY ←(WRj).0

影响标志位:

CY	AC	OV	N	Z
✓	-	-	✓	✓

[指令码] 0E(t)4H

0 0 0 0	1 1 1 0	t t t t	0 1 0 0

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

SRL

功能: 逻辑右移 1 位

说明: SRL 将指定变量右移 1 位，将最高有效位替换为零。移出的位（LSB）存储在 CY 位中。N 和 Z 标志位也受到影响。

SRL Rm

指令操作: (PC) ←(PC) + 2

(Rm).7 ←(Rm).0

(Rm).a ←(Rm).a+1

CY ←(Rm).0

影响标志位:

CY	AC	OV	N	Z
✓	-	-	✓	✓

[指令码] 1E(s)0H

0 0 0 1	1 1 1 0	s s s s	0 0 0 0

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

SRL WRj

指令操作: (PC) ←(PC) + 2

(WRj).15 ←0

(WRj).b ←(WRj).b + 1

CY ←(WRj).0

影响标志位:

CY	AC	OV	N	Z
✓	-	-	✓	✓

[指令码] 1E(t)4H

0 0 0 1	1 1 1 0	t t t t	0 1 0 0

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

SUB

功能: 相减

说明: 从目标操作数中减去指定的变量，将结果留在目标操作数中。如果第 7 位需要借位，则 SUB 置位 CY 标志位（借位），否则清零 CY 位。当有符号整数相减时，OV 标志位表示正数减负数时出现了负数，或者负数减正数时出现了正结果。本说明中的第 7 位是指操作数的最高有效字节（8、16 或者 32 位）。源操作数允许四种寻址模式：立即、间接、寄存器和直接寻址。除了 AC，所有标志位都受到影响，它不影响字和双字减法。

SUB Rmd,Rms

指令操作: (PC) ←(PC) + 2

(Rmd) ←(Rmd) - (Rms)

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 9CH

1 0 0 1	1 1 0 0	s s s s	S S S S

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

SUB WRjd,WRjs

指令操作: (PC) ←(PC) + 2

(WRjd) ←(WRjd) - (WRjs)

影响标志位:

CY	AC	OV	N	Z
✓	-	✓	✓	✓

[指令码] 9DH

1 0 0 1	1 1 0 1	t t t t	T T T T

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

SUB DRkd,DRks

指令操作: (PC) ←(PC) + 2

(DRkd) ←(DRkd) - (DRks)

影响标志位:

CY	AC	OV	N	Z
✓	-	✓	✓	✓

[指令码] 9FH

1 0 0 1	1 1 1 1	u u u u	U U U U

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

SUB Rm,#DATA

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) - #data

影响标志位:

CY	AC	OV	N	Z

✓	✓	✓	✓	✓

[指令码] 9E(s)0H

1	0	0	1	1	1	1	0	s	s	s	s	0	0	0	0	立即数

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

SUB WRj,#DATA16

指令操作: (PC) ←(PC) + 4

(WRj) ←(WRj) - #data16

影响标志位:

CY	AC	OV	N	Z
✓	-	✓	✓	✓

[指令码] 9E(t)4H

1	0	0	1	1	1	1	0	t	t	t	t	0	1	0	0	立即数高字节	立即数低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

SUB DRk,#0DATA16

指令操作: (PC) ←(PC) + 4

(DRk) ←(DRk) - #data16

影响标志位:

CY	AC	OV	N	Z
✓	-	✓	✓	✓

[指令码] 9E(u)8H

1	0	0	1	1	1	1	0	u	u	u	u	1	0	0	0	立即数高字节	立即数低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

SUB Rm,Dir8

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) - (dir8)

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 9E(s)1H

1	0	0	1	1	1	1	0	s	s	s	s	0	0	0	1	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

SUB WRj,Dir8

指令操作: (PC) ←(PC) + 3

(WRj) ←(WRj) - (dir8)

影响标志位:

CY	AC	OV	N	Z
✓	-	✓	✓	✓

[指令码] 9E(t)5H

1	0	0	1	1	1	1	0	t	t	t	t	0	1	0	1	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2(3 for SFR) 1(2 for SFR)

Hex: [A5]指令码 指令码

SUB Rm,Dir16

指令操作: (PC) ←(PC) + 4

(Rm) ←(Rm) - (dir16)

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 9E(s)3H

1	0	0	1	1	1	1	0	s	s	s	s	0	0	1	1	地址高字节	地址低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

SUB WRj,Dir16

指令操作: (PC) ←(PC) + 4

(WRj) ←(WRj) - (dir16)

影响标志位:

CY	AC	OV	N	Z
✓	-	✓	✓	✓

[指令码] 9E(t)7H

1	0	0	1	1	1	1	0	t	t	t	t	0	1	1	1	地址高字节	地址低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

SUB Rm,@WRj

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) - ((WRj))

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 9E(t)9(s)0H

1	0	0	1	1	1	1	0	t	t	t	t	1	0	0	1	s	s	s	s	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

SUB Rm,@DRk

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) - ((DRk))

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 9E(u)B(s)0H

1 0 0 1	1 1 1 0	u u u U	1 0 1 1	s s s s	0 0 0 0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

SUBB A,<src-byte>

功能: 借位减法

说明: SUBB 从累加器中一起减去指定的变量和进位标志位，将结果留在累加器中。如果第 7 位需要借位，则 SUBB 置位进位（借位）标志位，否则清零 C。（如果在执行 SUBB 指令之前 C 已置位，这表示在多种准确的减法中的上一步需要借位，因此从累加器中减去进位以及源操作数）。如果第 3 位需要借位，则 AC 置位，否则清零。如果需要借位到第 6 位而不是第 7 位，或者需要借位到第 7 位而不是第 6 位，则 OV 置位。OV 标志位表示正数减负数时出现了负数，或者负数减正数时出现了正数。源操作数允许四种寻址模式：寄存器、直接、寄存器间接或者立即数寻址。所有标志位都会受到影响。

SUBB A,Rn

指令操作: (PC) ←(PC) + 1

(A) ←(A) - (C) - (Rn)

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 98H - 9FH

1 0 0 1	1 r r r

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

SUBB A,Direct

指令操作: (PC) ←(PC) + 2

(A) ←(A) - (C) - (direct)

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 95H

1 0 0 1	0 1 0 1	直接地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

SUBB A,@Ri

指令操作: (PC) ←(PC) + 1

(A) ←(A) - (C) - ((Ri))

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 96H, 97H

1	0	0	1	0	1	1	i

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

SUBB A,#DATA

指令操作: (PC) ←(PC) + 2

(A) ←(A) - (C) - #data

影响标志位:

CY	AC	OV	N	Z
✓	✓	✓	✓	✓

[指令码] 94H

1	0	0	1	0	1	0	0	立即数

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

SWAP A

功能: 累加器交换半字节

说明: SWAP A 交换累加器的低位和高位半字节（四位域）（第 3 到第 0 位和第 7 到第 4 位）。该操作也可以认为是一个四位循环指令。 只影响 N 和 Z 标志位。

指令操作: (PC) ←(PC) + 1

(A3-0) ←(A7-4)

(A7-4) ←(A3-0)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] C4H

1	1	0	0	0	1	0	0

Binary 模式 **Source 模式**

指令长度: 1 1

时钟数: 1 1

Hex: 指令码 指令码

TRAP

功能: 作为 NOP 执行

指令操作: (PC) ←(PC) + 1

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] B9H

1 0 1 1	1 0 0 1
Binary 模式	**Source 模式**

指令长度: 2 1

时钟数: 2 1

Hex: [A5]指令码 指令码

XCH A,<byte>

功能: 累加器交换字节变量

说明: XCH 将指定变量的内容加载到累加器，同时将之前累加器内容写入指定变量。源数、目标操作数可以使用寄存器、直接或者寄存器间接寻址。不影响标志位。

XCH A,Rn

指令操作: (PC) ←(PC) + 1

(A) ↔(Rn)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] C8H - CFH

1 1 0 0	1 r r r
Binary 模式	**Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

XCH A,Direct

指令操作: (PC) ←(PC) + 2

(A) ↔(direct)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] C5H

1 1 0 0	0 1 0 1	直接地址
Binary 模式	**Source 模式**	

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

XCH A,@Ri

指令操作: (PC) ←(PC) + 1

(A) ↔((Ri))

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] C6H, C7H

1 1 0 0	0 1 1 i
Binary 模式	**Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

XCHD A,@Ri

功能: 半字节交换数字

说明: XCHD 将累加器的低半字节（第 3 到第 0 位，通常表示十六进制或者 BCD 数字）与指定寄存器间接寻址的内部内存位置的半字节交换。每个寄存器的高位半字节（第 7 到第 4 位）不受影响。不影响标志位。

指令操作: (PC) ←(PC) + 1

(A3-0) ↔((Ri)3-0)

影响标志位:

CY	AC	OV	N	Z
-	-	-	-	-

[指令码] D6H, D7H

1 1 0 1	0 1 1 i

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

XRL

功能: 变量的逻辑异或

说明: 在指定变量之间执行按位逻辑异或运算，将结果存储在目标中。目标操作数可以是累加器、寄存器或者直接地址。 这两个操作数允许 12 种寻址模式组合。当目标是累加器或者寄存器时，源数可以是寄存器、直接、寄存器间接或者立即数寻址；当目标是直接地址时，源数可以是累加器或者立即数。只影响 N 和 Z 标志位。

注意：当该指令用于修改输出端口时，用作原始端口数据的值将从输出数据锁存器中读取，而不是输入引脚。

XRL A,Rn

指令操作: (PC) ←(PC) + 1

(A) ←(A) xor (Rn)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 68H - 6FH

0 1 1 0	1 r r r

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

XRL A,Direct

指令操作: (PC) ←(PC) + 2

(A) ←(A) xor (direct)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 65H

0 1 1 0	0 1 0 1	直接地址

Binary 模式 **Source 模式**

指令长度: 2 3

时钟数: 1 1

Hex: 指令码 指令码

XRL A,@Ri

指令操作: (PC) ←(PC) + 1

(A) ←(A) xor ((Ri))

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 66H, 67H

0	1	1	0	0	1	1	i

Binary 模式 **Source 模式**

指令长度: 1 2

时钟数: 1 2

Hex: 指令码 [A5]指令码

XRL A,#DATA

指令操作: (PC) ←(PC) + 2

(A) ←(A) xor #data

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 64H

0	1	1	0	0	1	0	1	立即数

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

XRL Direct,A

指令操作: (PC) ←(PC) + 2

(direct) ←(direct) xor (A)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 62H

0	1	1	0	0	0	1	0	直接地址

Binary 模式 **Source 模式**

指令长度: 2 2

时钟数: 1 1

Hex: 指令码 指令码

XRL Direct,#DATA

指令操作: (PC) ←(PC) + 3

(direct) ←(direct) xor #data

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 63H

---

0 1 1 0	0 0 1 1	直接地址	立即数

Binary 模式 **Source 模式**

指令长度: 3 3

时钟数: 1 1

Hex: 指令码 指令码

XRL Rmd,Rms

指令操作: (PC) ←(PC) + 2

(Rmd) ←(Rms) xor (Rmd)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 6CH

0 1 1 0	1 1 0 0	s s s s	S S S S

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

XRL WRjd,WRjs

指令操作: (PC) ←(PC) + 2

(WRjd) ←(WRjs) xor (WRjd)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 6DH

0 1 1 0	1 1 0 1	t t t t	T T T T

Binary 模式 **Source 模式**

指令长度: 3 2

时钟数: 2 1

Hex: [A5]指令码 指令码

XRL Rm,#DATA

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) xor #data

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 6E(s)0H

0 1 1 0	1 1 1 0	s s s s	0 0 0 0	立即数

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

XRL WRj,#DATA16

指令操作: (PC) ←(PC) + 4

(WRj) ←(WRj) xor #data16

影响标志位:

CY	AC	OV	N	Z

-	-	-	✓	✓

[指令码] 6E(t)4H

0 1 1 0	1 1 0 1	t t t t	0 1 0 0	立即数高字节	立即数低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

XRL Rm,Dir8

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) xor (dir8)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 6E(s)1H

0 1 1 0	1 1 1 0	s s s s	0 0 0 1	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

XRL WRj,Dir8

指令操作: (PC) ←(PC) + 3

(WRj) ←(WRj) xor (dir8)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 6E(t)5H

0 1 1 0	1 1 0 1	t t t t	0 1 0 1	直接地址

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2(3 for SFR) 1(2 for SFR)

Hex: [A5]指令码 指令码

XRL Rm,Dir16

指令操作: (PC) ←(PC) + 4

(Rm) ←(Rm) xor (dir16)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 6E(s)3H

0 1 1 0	1 1 1 0	s s s s	0 0 1 1	地址高字节	地址低字节

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

XRL WRj,Dir16

指令操作: (PC) ←(PC) + 4

(WRj) ←(WRj) xor (dir16)

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 6E(t)7H

0	1	1	0	1	1	0	1	t	t	t	t	0	1	1	1	地址高字节		地址低字节	

Binary 模式 **Source 模式**

指令长度: 5 4

时钟数: 2 1

Hex: [A5]指令码 指令码

XRL Rm,@WRj

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) xor ((WRj))

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 6E(t)9(s)0H

0	1	1	0	1	1	1	0	t	t	t	t	1	0	0	1	s	s	s	s	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

XRL Rm,@DRk

指令操作: (PC) ←(PC) + 3

(Rm) ←(Rm) xor ((DRk))

影响标志位:

CY	AC	OV	N	Z
-	-	-	✓	✓

[指令码] 6E(u)B(s)0H

0	1	1	0	1	1	1	0	u	u	u	u	1	0	1	1	s	s	s	s	0	0	0	0

Binary 模式 **Source 模式**

指令长度: 4 3

时钟数: 2 1

Hex: [A5]指令码 指令码

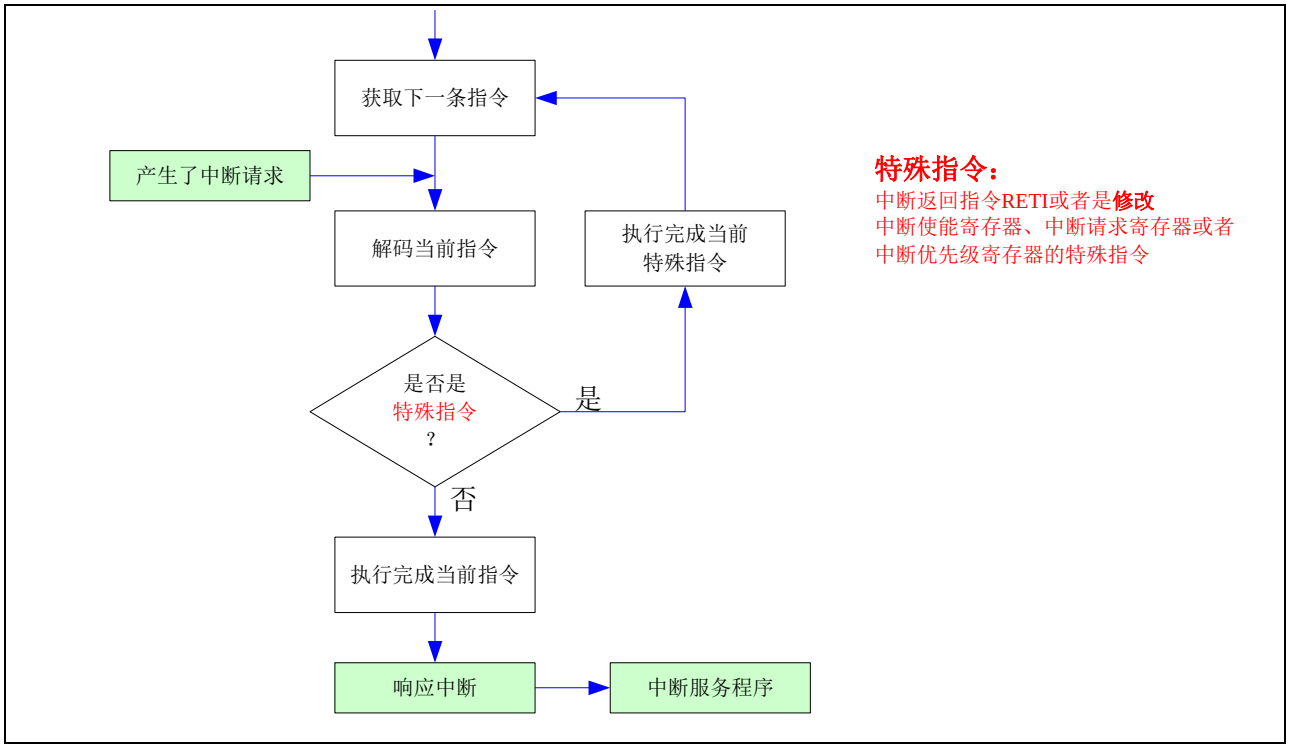
A.3 多级流水线内核的中断响应

增强型 8051（例如：AI32G/AI8H 系列）和 32 位 8051（例如：Ai8051U 系列）的 MCU 内核为多级流水线设计，在中断响应方面的设计和传统的 8051（例如：STC89C52 系列）略有差异。

对于传统的 8051（例如：STC89C52 系列）：

如果当前正在执行的指令是中断返回指令 RETI 或者是访问中断使能寄存器、中断请求寄存器或者中断优先级寄存器的特殊指令时，CPU 等待当前的这条特殊的指令执行完，再执行一条指令才能响应中断请求；

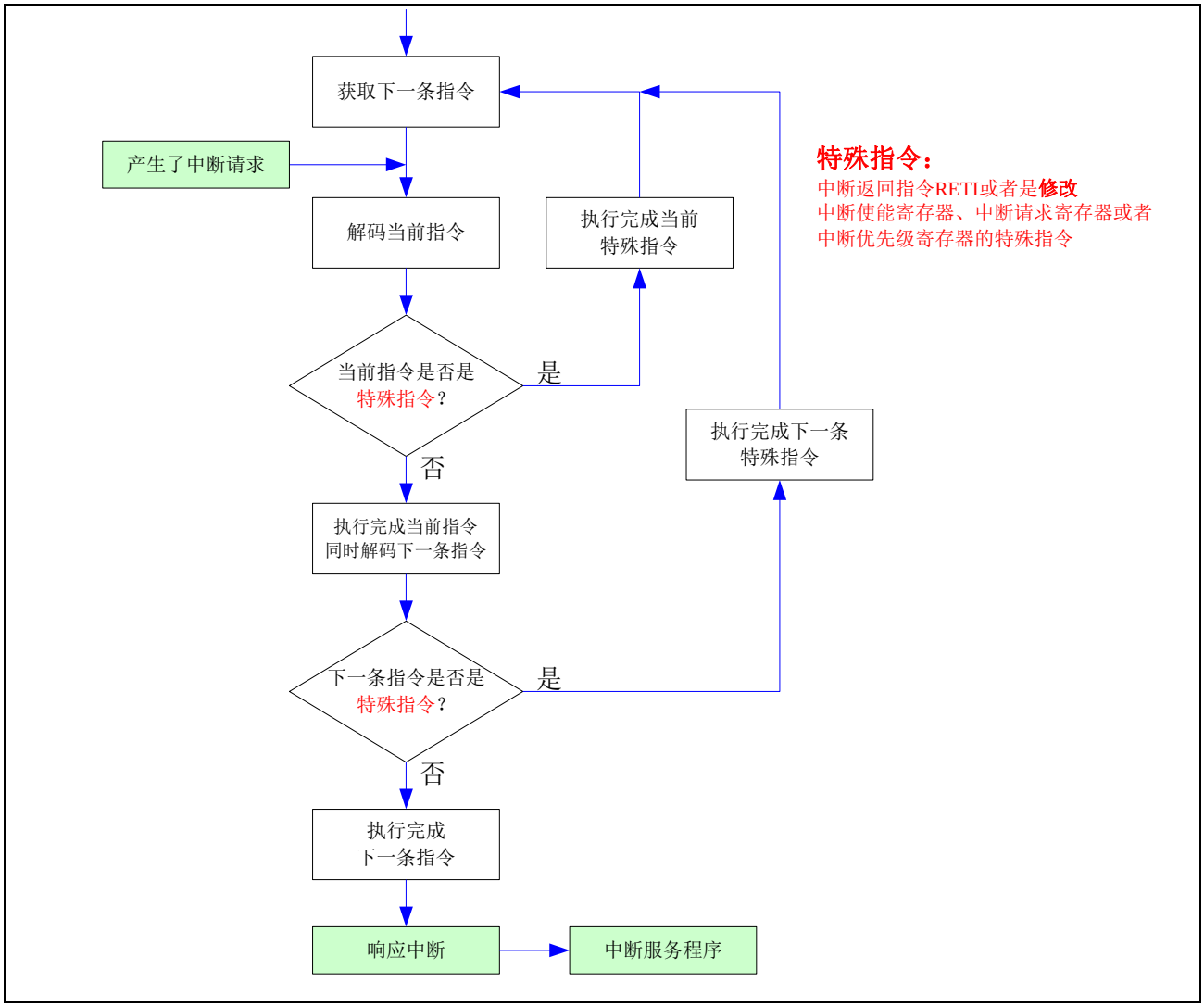
如果当前正在执行的指令不是上面所指的特殊指令，则等当前指令执行完成后就立即响应中断请求；



对于增强型 8051 单片机（例如：AI32G/AI8H 系列），由于是多级流水线设计，响应中断上会比传统的 8051（例如：STC89C52 系列）再多执行一条语句：

如果当前正在执行的指令是中断返回指令 RETI 或者是访问中断使能寄存器、中断请求寄存器或者中断优先级寄存器的特殊指令时，CPU 等待当前的这条特殊的指令执行完，同时解码下一条指令，直到下一条指令不是特殊指令，则等下一条指令执行完成才能响应中断请求；

如果当前正在执行的指令不是上面所指的特殊指令，则等当前指令执行完成后，同时会解码下一条指令，如果下一条也不是特殊指令，则会等下一条指令执行完成后再立即响应中断请求；


获取下一条指令
产生了中断请求
解码当前指令
执行完成当前
特殊指令
当前指令是否是
特殊指令?
是
否
执行完成下一条
特殊指令
执行完成当前指令
同时解码下一条指令
下一条指令是否是
特殊指令?
是
否
执行完成
下一条指令
响应中断
中断服务程序
特殊指令：
中断返回指令RETI或者是修改
中断使能寄存器、中断请求寄存器或者
中断优先级寄存器的特殊指令

附录B　重要说明

1、由于封装打线失误，批号为2534A670239.YZ 和 2544A670239.YZ的
STC32G144K246-LQFP64：

- **Pin8不是Core_Vcc，而是P6.3**
- **Pin41不是Vcc，而是P4.1**

2025年12月2日之后封装的芯片会将Pin8封装成Core_Vcc，Pin41封装成Vcc。

2、QSPI 的 P2P 模式会影响最后面的部分 XRAM 的数据。可以根据应用注意事项来使用，后续改版修正

3、CAN FD 相关的 XFR 寄存器的读取需要读取两次，后续改版时修正

4、INT2/INT3/INT4 新增的“上升沿+下降沿"模式不能产生中断，只支持 STOP 唤醒，后续改版时修正

5、关于 DMA 使用注意事项

将 CKCON 设置为 0 时，xdata 访问速度设置为最快，写 xdata 为 2T，读 xdata 为 3T。此时如果硬件 DMA 和 CPU 同时访问 xdata，会出现总线竞争，需要进行总线仲裁。由于芯片内部的仲裁电路设计不够完美，会导致 CPU 访问 xdata 出错。
将 CKCON 设置为 1，可完美解决硬件 DMA 和 CPU 同时访问 xdata 出现的仲裁错误问题。CKCON 为 1 时，xdata 的访问时间会多 1T，写 xdata 为 3T，读 xdata 为 4T
解决硬件 DMA 时 CPU 访问 xdata 出错的问题的解决方案（下面两个方案任选其一即可）：
1、将 CKCON 设置为 1。此方案可完美解决问题，但访问 xdata 会多 1 个时钟
2、当硬件 DMA 访问 xdata 过程中，用户程序不得访问 xdata。用户程序只能在 DMA 不工作时才能读写 DMA 在 xdata 中的缓冲区。

附录C 电气特性

绝对最大额定值

参数	最小值	最大值	单位	说明
存储温度	-55	+155	℃	
工作温度	-40	+85	℃	使用内部高速 IRC 和外部晶振， 工作频率建议控制在 100MHz 附近
	-40	+125	℃	当温度高于 85℃时请使用外部耐高温晶振， 且工作频率建议控制在 72MHz 及以下
工作电压	1.9	5.5	V	当工作温度低于-40℃时，工作电压不得低于 3.0V
VDD 对地电压	-0.3	+5.5	V	
I/O 口对地电压	-0.3	VDD+0.3	V	

Flash 可靠性特性

标号	参数	典型值	测试条件
N_{NED}	扇区擦除次数	10 万次	
T_{DR}	数据保持时间	100 年以上	@25℃
		20 年以上	@105℃
		15 年以上	@125℃
		平均数据保持时间 20 年以上	

内部 IRC 温漂特性（参考温度 25℃）

温度	范围		
	最小值	典型值	最大值
-20℃~65℃		-0.88%~+1.05%	
-40℃~85℃		-1.38%~+1.42%	
-40℃~125℃		-3%~+3%	

低压复位门槛电压（测试温度 25℃）

级别	电压		
	最小值	典型值	最大值
POR		1.9V	
LVR0		2.0V	
LVR1		2.4V	
LVR2		2.7V	
LVR3		3.0V	

直流特性（VSS=0V，VDD=5.0V，测试温度=25℃）（空白处后续会陆续补充）

标号	参数	范围				测试环境
		最小值	典型值	最大值	单位	
I_{PD}	掉电模式电流	-	9	-	uA	
I_{WKT}	掉电唤醒定时器	-	-	-	uA	
I_{LVD}	低压检测模块	-	-	-	uA	
I_{IDL}	空闲模式电流（6MHz）	-	-	-	mA	
	空闲模式电流（11.0592MHz）	-	-	-	mA	
	空闲模式电流（24MHz）	-	-	-	mA	
	空闲模式电流（内部 32KHz）	-	-	-	mA	
I_{NOR}	正常模式电流（6MHz）	-	-	-	mA	
	正常模式电流（11.0592MHz）	-	-	-	mA	
	正常模式电流（24MHz）	-	-	-	mA	
	正常模式电流（内部 32KHz）	-	-	-	mA	
I_{CC}	普通工作模式电流	-	-	-	mA	
V_{IL1}	输入低电平	-	-	-	V	打开施密特触发
		-	-	-	V	关闭施密特触发
V_{IH1}	输入高电平（普通 I/O）	-	-	-	V	打开施密特触发
		-	-	-	V	关闭施密特触发
I_{OL1}	输出低电平的灌电流	-	-	-	mA	详细数据参考附录 F.1：I/O 口驱动能力（驱动电流对应的 I/O 上的电压）
I_{OH1}	输出高电平电流（双向模式）	-	-	-	uA	
I_{OH2}	输出高电平电流（推挽模式）	-	-	-	mA	
I_{IL}	逻辑 0 输入电流	-	-	-	uA	
I_{TL}	逻辑 1 到 0 的转移电流	-	-	-	uA	
R_{PU}	I/O 口上拉电阻	-	-	-	KΩ	
I/O 速度	I/O 大电流驱动，I/O 快速转换	-	-		MHz	PxDR=0, PxSR=0
	I/O 小电流驱动，I/O 快速转换		-		MHz	PxDR=1, PxSR=0
	I/O 大电流驱动，I/O 慢速转换		-		MHz	PxDR=0, PxSR=1
	I/O 小电流驱动，I/O 慢速转换		-		MHz	PxDR=1, PxSR=1
比较器	最快速度		-		MHz	关闭所有模拟和数字滤波
	模拟滤波时间		-		us	
	数字滤波时间		-		系统时钟	LCDTY＝0
			-			LCDTY=n (n=1~63)
I_{PD2}	使能比较器时掉电模式功耗	-	-	-	uA	
I_{PD3}	使能 LVD 时掉电模式功耗	-	-	-	uA	

直流特性（VSS=0V，VDD=3.3V，测试温度=25℃）（空白处后续会陆续补充）

标号	参数	范围				测试环境
		最小值	典型值	最大值	单位	
I_{PD}	掉电模式电流	-	-	-	uA	
I_{WKT}	掉电唤醒定时器	-	-	-	uA	
I_{LVD}	低压检测模块	-	-	-	uA	
I_{IDL}	空闲模式电流（6MHz）	-	-	-	mA	
	空闲模式电流（11.0592MHz）	-	-	-	mA	
	空闲模式电流（24MHz）	-	-	-	mA	
	空闲模式电流（内部 32KHz）	-	-	-	mA	
I_{NOR}	正常模式电流（6MHz）	-	-	-	mA	
	正常模式电流（11.0592MHz）	-	-	-	mA	
	正常模式电流（24MHz）	-	-	-	mA	
	正常模式电流（内部 32KHz）	-	-	-	mA	
I_{CC}	普通工作模式电流	-	-	-	mA	
V_{IL1}	输入低电平	-	-	-	V	打开施密特触发
		-	-	-	V	关闭施密特触发
V_{IH1}	输入高电平（普通 I/O）	-	-	-	V	打开施密特触发
		-	-	-	V	关闭施密特触发
I_{OL1}	输出低电平的灌电流	-	-	-	mA	详细数据参考附录 F.1：I/O 口驱动能力（驱动电流对应的 I/O 上的电压）
I_{OH1}	输出高电平电流（双向模式）	-	-	-	uA	
I_{OH2}	输出高电平电流（推挽模式）	-	-	-	mA	
I_{IL}	逻辑 0 输入电流	-	-	-	uA	
I_{TL}	逻辑 1 到 0 的转移电流	-	-	-	uA	
R_{PU}	I/O 口上拉电阻	-	-	-	KΩ	
I/O 速度	I/O 大电流驱动，I/O 快速转换	-	-		MHz	PxDR=0, PxSR=0
	I/O 小电流驱动，I/O 快速转换		-		MHz	PxDR=1, PxSR=0
	I/O 大电流驱动，I/O 慢速转换		-		MHz	PxDR=0, PxSR=1
	I/O 小电流驱动，I/O 慢速转换		-		MHz	PxDR=1, PxSR=1
比较器	最快速度		-		MHz	关闭所有模拟和数字滤波
	模拟滤波时间		-		us	
	数字滤波时间		-		系统时钟	LCDTY＝0
			-			LCDTY=n (n=1~63)
I_{PD2}	使能比较器时掉电模式功耗	-	-	-	uA	
I_{PD3}	使能 LVD 时掉电模式功耗	-	-	-	uA	

C.1 I/O 口驱动能力（驱动电流对应的 I/O 上的电压）

（VSS=0V，VDD=5.0V，测试温度=25℃）

普通 I/O 推挽输出 1	一般推力	强推力
10mA	4.54V	4.74V
20mA	4.05V	4.48V
30mA	3.47V	4.21V
40mA	2.57V	3.92V
50mA	-	3.59V
60mA	-	3.15V
70mA	-	2.63V
80mA	-	1.52V
90mA	-	-

（VSS=0V，VDD=3.3V，测试温度=25℃）

普通 I/O 推挽输出 1	一般推力	强推力
10mA	2.64V	2.95V
20mA	1.57V	2.61V
30mA	-	2.19V
40mA	-	1.44V
50mA	-	-

（VSS=0V，VDD=5.0V，测试温度=25℃）

普通 I/O 推挽输出 0/准双向口输出 0/开漏模式 0	一般推力	强推力
10mA	0.35V	0.22V
20mA	0.70V	0.41V
30mA	1.12V	0.59V
40mA	1.80V	1.02V
50mA	-	1.32V
60mA	-	1.67V
70mA	-	-

（VSS=0V，VDD=3.3V，测试温度=25℃）

普通 I/O 推挽输出 0/准双向口输出 0/开漏模式 0	一般推力	强推力
10mA	0.49V	0.29V
20mA	1.13V	0.51V
30mA	-	0.77V
40mA	-	1.16V
50mA	-	-

C.2 ADC 静态特性

符号	描述	最小值	典型值	最大值	单位
RES	分辨率	-	12	-	Bits
E_T	整体误差	-	0.5	1	LSB
E_O	偏移误差	-	-0.1	1	LSB
E_G	增益误差	-	0	1	LSB
E_D	微分非线性误差	-	0.7	1.5	LSB
E_I	积分非线性误差	-	1	2	LSB
R_{AIN}	通道等效电阻	-	∞	-	ohm
R_{ESD}	采样保持电容前串接的抗静电电阻	-	700	-	ohm
C_{ADC}	内部采样保持电容	-	16.5	-	pF

C.3 DAC 电气特性

Symbol	Description	Condition	Min.	Typ.	Max.	Unit
VDDA50	Analog power supply voltage	-	1.8	5	5.5	V
VDDD15	Internal core power supply voltage	-	1.35	1.5	1.65	V
Temp	Operating ambient temperature	-	-40	25	125	℃
RES	Resolution	-		12	-	Bit
INL	Integral non-linearity error	-	-4	±1.0	4.0	LSB
DNL	Differential non-linearity error	-		±1	2	LSB
Offset	Offset error drift			+/-1.6		uV/℃
Gain error	Gain error			±0.5	±1	LSB
V_{refp}	Positive reference input voltage range		1.2		VDDA50	V
V_{refn}	Negative reference input voltage range			0		V
CLK	Clock frequency	-	0.2	2		MHz
$I_{_vrefp}$	Power of VREFP@ enable mode at 5V. Power of VREFP @ disable mode	-	-	0.6 10		mA nA
$I_{_VDDA50}$	Power of VDDA50 @ enable mode Power of VDDA50 @ disable mode.	-	-	1 5		uA nA
$I_{_VDDD15}$	PowerVDDD15 @ enable mode VDDD15 @ disable modePower	-	-	1 5		uA nA
C_{load}	Output Loading			10		fF
R_{out}	Output resistance			4		KOhm

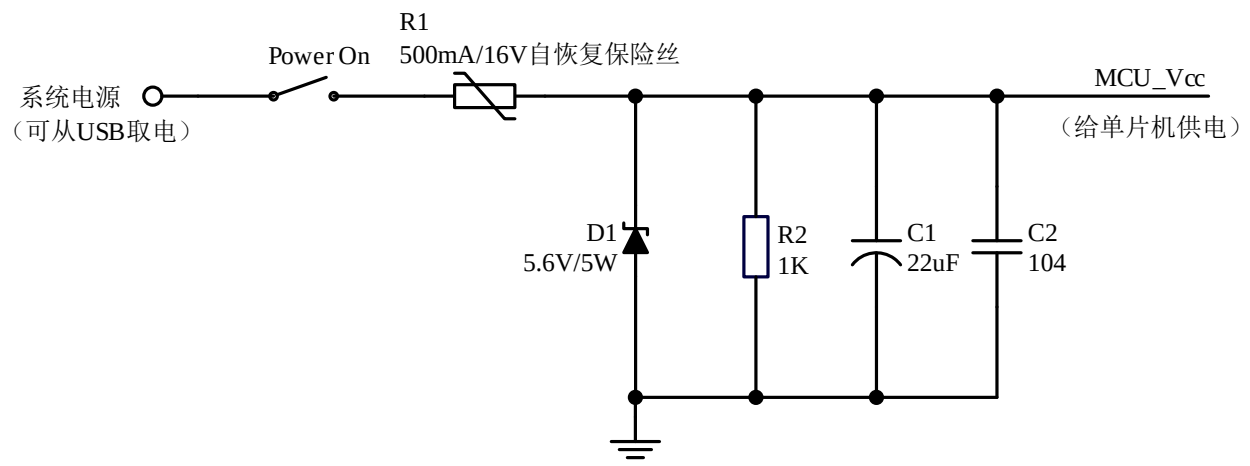
C.4 PGA 静态特性（IBS=0）

符号	描述	最小值	典型值	最大值	单位	测试条件
AV_{CC}	工作电压	2.4	5	5.5	V	
$CoreV_{CC}$	内核工作电压	1.35	-	1.65	V	
V_{OS}	输入偏移电压	-10	-	10	mV	V_{CM}= 0.1V ~ (AV_{CC}-0.1V)
V_{CM}	输入共模电压范围	0	-	AV_{CC}	V	
I_{DD_OA}	OPAMP 静态电流	-	127	-	uA	缓冲模式
I_{PD}	睡眠电流	-	0.02	-	uA	
I_{OUT}	输出电流	-4.5	-	4.5	mA	
C_{MRR}	共模抑制比	-	113	-	dB	C_L=50pF
P_{SRR+}	电源抑制比	-	99	-	dB	C_L=50pF
G_{BW}	增益带宽	-	2.5	-	MHz	C_L=50pF
A_{OL}	开环增益	-	113	-	dB	C_L=50pF
Φ_m	相位裕度	-	72	-	deg	C_L=50pF，R_L=2KΩ
T_{SR+}	正转换速率	-	1.5	-	V/us	C_L=50pF，step 1V
T_{settle}	稳定时间	-	0.8	-	us	0.1% final value，Gain=1，+1V step 1V
E_{NI}	ENI 输入噪声电压	-	38	-	uV	AV_{CC}=5V,V_{IN}=2.5V f=0.1~10Hz
E_{NI}	输入噪声密度	-	180	-	nV/√Hz	AV_{CC}=5V,V_{IN}=2.5V f=1KHz
		-	61	-		AV_{CC}=5V,V_{IN}=2.5V f=10KHz
反相 PGA 模式	GSEL[2:0]=0	-	N/A	-		AV_{CC}=5.0V, I_{OUT}=0mA
	GSEL[2:0]=1	0.98	1	1.02		
	GSEL[2:0]=2	1.96	2	2.04		
	GSEL[2:0]=3	3.92	4	4.08		
	GSEL[2:0]=4	7.84	8	8.16		
	GSEL[2:0]=5	15.68	16	16.32		
	GSEL[2:0]=6	31.36	32	32.64		
	GSEL[2:0]=7	47.04	48	48.96		
非反相 PGA 模式	GSEL[2:0]=0	-	N/A	-		AV_{CC}=5.0V, I_{OUT}=1mA
	GSEL[2:0]=1	1.96	2	2.04		
	GSEL[2:0]=2	2.94	3	3.06		
	GSEL[2:0]=3	4.9	5	5.1		
	GSEL[2:0]=4	8.82	9	9.18		
	GSEL[2:0]=5	16.66	17	17.34		
	GSEL[2:0]=6	32.36	33	33.66		
	GSEL[2:0]=7	48.02	49	49.98		

C.5 PLL500M 电气特性

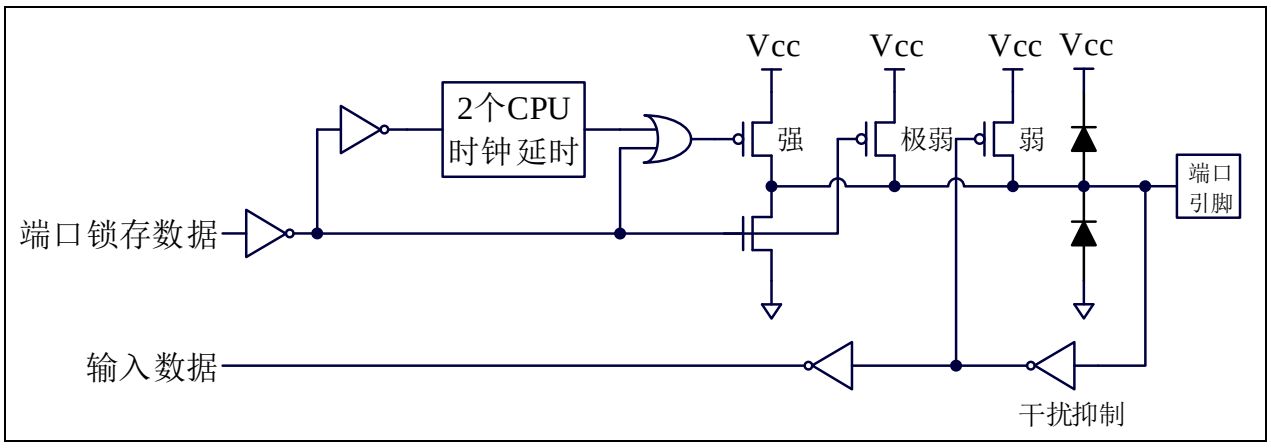
Symbol	Parameter	Conditions	Value			Unit
			Min	Typ	Max	
VDDL	Operating Voltage		1.35	1.5	1.65	V
CK_OUT	Frequency output range	TA = -40℃ ~ 125℃	312		492	MHz
IOP	Operating Current			360	600	uA
IPD	Power Down Current			0.1	1.03	uA
JPP	peak-to-peak period jitter			60		ps
FLOCK	Input clock range	TA = -40℃ ~ 125℃		6		MHz
TLOCK	Locking Time			20	40	us

附录D 单片机电源系统最简易自我保护电路

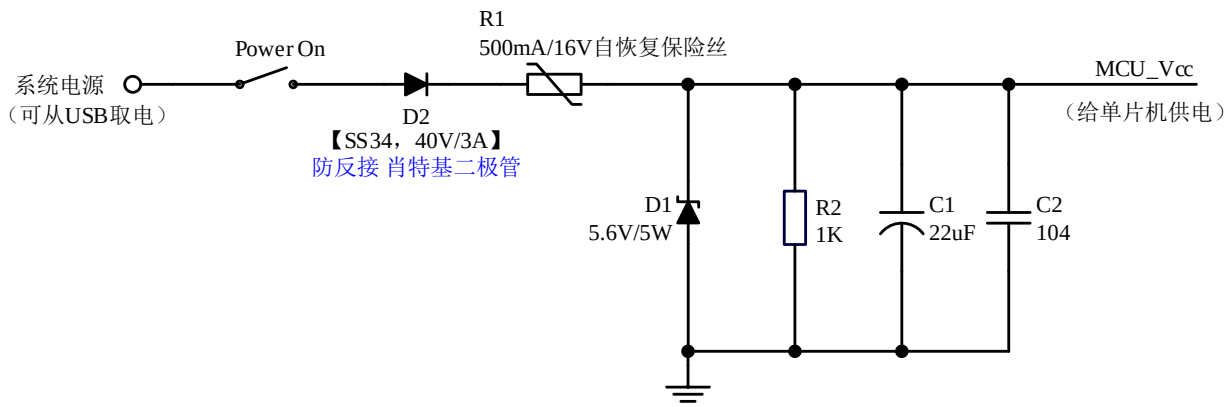


3.3V系统可使用4.1V稳压管

I/O 的内部结构图



附录E　防电源反接电路，汽车电子常用电路



3.3V系统可使用4.1V稳压管

附录F　更新记录

● 2026/06/18

1. 增加章节：SKDIP28 封装尺寸图
2. 更新：STC32G144K246 / STC32G96K246 / STC32G18K246 系列的多功能管脚切换表

● 2026/06/05

1. 增加章节：STC32G144K246 内部上拉电阻/下拉电阻，小结
2. 修正：个别笔误、乱码等

● 2026/05/26

1. 更新：LQFP32 封装尺寸图（9mm*9mm）-- 塑封体正面 3 个顶杆印、塑封体正面 1 个顶杆印
2. 更新：LQFP48 封装尺寸图（9mm*9mm）-- 塑封体正面 3 个顶杆印、塑封体正面 1 个顶杆印
3. 更新：电源控制寄存器（PCON）中 LVDF 和 POF 的说明
4. 修正：指令详解中个别笔误

● 2026/05/09

1. 增加章节：RTC 实战线路图
2. 增加章节：ADC 参考线路图 1 -- 有外部 专业 ADC 转换参考源 REF3025
3. 增加章节：ADC 参考线路图 3 -- 无外接 ADC 参考源，直接用内部 1.19V BandGap，反推外部输入电压
4. 更新：ADC 参考线路图 2 -- 有外部 普通 ADC 转换参考源 TL431B

● 2026/04/24

1. 更新：STC32G18K64-64MHz-LQFP32，TSSOP20 管脚图
2. 更新：LQFP32 封装尺寸图

● 2026/04/17

1. 增加章节：TL431/CD431 省电模式正规控制电路
2. 增加章节：DMA 使用注意事项
3. 增加章节：附录 -- 单片机电源系统最简易自我保护电路
4. 增加章节：附录 -- 防电源反接电路，汽车电子常用电路
5. 更新：ADC 参考线路图

● 2026/04/01

1. 更新：删除 DAC1 和 DAC2 的 P2P 功能（本系列 DAC 不支持 P2P 功能）

2026/03/19

1. 增加章节：STC32G18K64-64MHz-LQFP48/32，TSSOP20 系列单片机提前预告
2. 更新：STC32G96K246 系列单片机特性、管脚图及多功能管脚切换表
3. 更新：ADC 参考线路图

2026/03/06

1. 增加章节：使用 USB 时，CPU 如何跑 100MHz ~ 120MHz
2. 增加章节：使用 CAN-FD / CAN 时，CPU 如何跑 100MHz ~ 120MHz

2026/02/12

1. 增加章节：省电模式唤醒必须设置的寄存器，注意事项
2. 更新：EEPROM 操作时间
3. 更新：高速高级 PWM + DMA（组合使用）特别注意事项

2026/02/06

1. 增加章节：利用串口 DMA 的发送时间间隔寄存器可实现 1.5bit/2bit 停止位
2. 更新：STC32G96K246 系列单片机多功能管脚切换表

2026/02/04

1. 增加章节：P5.4 和 P1.2 脚功能切换寄存器（P_SWX1）
2. 增加：对 CANn_DMA 辅助控制寄存器（CANn_AUX_CR）位 STBYEN 增加使用说明
3. 增加：CAN 命令寄存器（CAN_TCMD）增加 STBY 位及其相关使用说明

2026/01/30

1. 增加：对 LIN 波特率寄存器（DLL/DLH）的 SYNC 功能增加说明
2. 更新：原“STC32G48K128 系列”更名为“STC32G96K246 系列”，并更新本章节全部内容

2026/01/27

1. 增加章节：附录---- PLL500M 电气特性
2. 增加：串口 n（n=1~8）的同步模式控制寄存器 5（USARTnCR5）（n=1~8）的 SnASYNC（n=1~8）功能增加“注意”
3. 更新：典型发光二极管控制电路

2026/01/16

1. 增加章节：SPI 实测波形图
2. 增加章节：QSPI 实测波形图
3. 增加章节：DAC 电气特性
4. 更新：I/O 口上、下拉电阻参数

5. 更新：附录---- I/O 口驱动能力（驱动电流对应的 I/O 上的电压）

2026/01/09

1. 更新：外设到外设（P2P）DMA 矩阵图
2. 更新：CAN_FD 的 STB 发送缓冲区最多可以缓冲 4 帧数据
3. 更新：CAN_FD 的接收缓冲区最多可以缓冲 16 帧数据
4. 更新：CAN 发送控制寄存器（CAN_TCTRL）内容

2026/01/05

1. 增加章节：DAC 范例程序--【DAC+OP】应用：【DAC1+OP1；DAC2+OP3；DAC2+OP4】同时输出 3 个正弦波
2. 增加章节：DAC 范例程序-- 录音/放音，ADC+DAC 应用：STC32G144K246-实验箱，复读机-A 率-存储在 SRAM
3. 增加章节：DAC 范例程序-- STC32G144K246，DAC-DMA 例程及注意事项
4. 更新：在“混合电压供电系统 3V/5V 器件 I/O 口互连”中“锗二极管”更新为“肖特基二极管”
5. 修正：个别笔误、乱码等

2025/12/26

1. 增加章节：I2Sn 时钟分频寄存器（I2Sn_CLKDIV）
2. 更新：I2S 音频采样频率计算公式

2025/12/22

1. 增加章节：CAN1 功能脚切换
2. 增加章节：CAN2 功能脚切换
3. 增加章节：LIN 总线

2025/12/19

1. 更新：关于 I/O 的注意事项
2. 更新：配置 I/O 口的“PnM0 与 PnM1 的组合方式表”
3. 更新：“【无弱上拉 准双向口】+上拉电阻 = 准双向口”章节

2025/12/18

1. 增加章节：【无弱上拉 准双向口】+上拉电阻 = 准双向口
2. 更新：“I/O 的结构图”章节中上拉/下拉电阻为：47K@5V，32K@3.3V
3. 删除原章节：新增上拉电阻和下拉电阻

2025/12/16

1. 增加章节：【USB 转双串口@Ai8H2K12U-SOP16】工具自动烧录，串口仿真+串口通讯
2. 增加章节：USB 转双串口芯片全自动停电/上电烧录，串口仿真+串口通讯，5V

3. 增加章节：USB 转双串口芯片全自动烧录，串口仿真+串口通讯，3.3V 原理图
4. 增加章节：USB 转双串口芯片进行自动烧录/仿真+串口通讯，5V/3.3V 跳线选择
5. 增加章节：通用 USB 转串口芯片全自动停电/上电烧录，串口仿真，5V 原理图
6. 增加章节：通用 USB 转串口芯片全自动停电/上电烧录，串口仿真，3.3V 原理图
7. 增加章节：USB 转串口芯片全自动停电/上电烧录/仿真/通信，5V/3.3V 跳线选择
8. 增加章节：USB 转串口芯片进行烧录/串口仿真，手动停电/上电，5V/3.3V 原理图
9. 增加章节：USB 转串口芯片进行烧录，串口仿真，手动停电/上电，3.3V 原理图
10. 增加章节：USB-Link1D 支持 脱机下载 说明
11. 增加章节：USB-Link1D 支持 脱机下载，如何免烧录环节
12. 增加章节：USB-Writer1A 编程器/烧录器 • 支持 • 插在 • 锁紧座上 • 烧录
13. 增加章节：USB-Writer1A 支持 自动烧录机，通信协议和接口
14. 增加章节：P2P 使用注意事项

● 2025/12/12

1. 更新：“特性及价格”章节中“工作温度”的描述
2. 更新：STC32G144K246-LQFP100/LQFP64/LQFP48/LQFP44 管脚图

● 2025/12/10

1. 增加章节：CAN-FD 功能说明
2. 增加章节：附录--指令集
3. 删除原章节：I2S 从模式控制寄存器（I2SnMD），并更新特殊功能寄存器表

● 2025/12/05

1. 增加章节：附录--电气特性
2. 增加：“特殊功能寄存器初始值”说明
3. 更新：附录--重要说明

● 2025/12/04

1. 增加：在“STC32G48K128 系列单片机--多功能管脚切换”中增加 DCMI 接口的管脚初步规划

● 2025/12/02

1. 增加章节：附录--重要说明

● 2025/11/26

1. 增加章节：DMA 存储器读写地址选择（DMA_ARB_CFG）

● 2025/11/25

1. 增加章节：STC32G48K128 系列单片机提前预告
2. 修正：STC32G144K246 系列“多功能管脚切换”表格中 USART1/U1SCLK、USART2/U2SCLK、

SPI/SCLK 的 I/O 口为 P4.3

2025/11/21

1. 更新：多功能管脚切换
2. 补充：“特殊功能寄存器列表”部分数据
3. 修正个别笔误

2025/11/11

1. 增加章节：关于 Keil 的双数据指针编译选项
2. 更新：可以唤醒省电模式/主时钟停振模式的中断资源
3. 更新：“外部中断类型”章节中 INTXIT 寄存器说明（INT2/3/4 只支持下降沿中断，可下降沿/上升沿/下降沿+上升沿唤醒）

2025/10/31

1. 修正：DSP32 章节中 DPUOP 寄存器地址为 D8H

2025/10/29

1. 增加章节：RAM 中运行程序@120MHz
2. 增加章节：CANn_DMA 辅助控制寄存器（CANn_AUX_CR）
3. 更新：存储器地址映像图
4. 更新：CANnR_DMA 配置寄存器（DMA_CANnR_CFG）的说明

2025/10/28

1. 更新：通信协议流程图

2025/10/24

1. 修正个别笔误、乱码

2025/10/22

1. 补充内容：扩展特殊功能寄存器列表（XFR：0x7EF400-0x7EF4FF）
2. 补充内容：扩展特殊功能寄存器列表（XFR：0x7EF300-0x7EF3FF）
3. 增加章节：CAN-FD 主要特性
4. 增加章节：CAN-FD 相关的特殊功能寄存器

2025/10/21

1. 增加章节：STC32G144K246 实验箱实物图及程序下载流程

2. 增加章节：STC32G144K246 仿真 -- SWD 硬件仿真
3. 增加章节：运放应用笔记及注意事项

● 2025/10/18

1. 增加章节：i-Cache 应用，如何让高速系统时钟和慢速 Flash 协同
2. 更新：增加 I/O 口上拉电阻和下拉电阻的参考值范围

● 2025/10/16

1. 增加章节：新建项目和项目设置，在 Keil 中
2. 增加章节：了解 File、Build、Debug 工具栏与仿真相关的常用操作，在 Keil 中
3. 增加章节：设置超 64K 程序代码的项目（Source 模式）
4. 增加章节：程序超 64K 时如何设置保留 EEPROM 空间

● 2025/10/14

1. 更新：STC32G144K246-LQPF100 管脚图

● 2025/10/13

1. 更新章节：STC32G144K246 管脚图
2. 增加章节：管脚说明

● 2025/10/10

1. 更新章节：中断系统及外部中断 INT0/1/2/3/4
2. 更新章节：USB 2.0-FS 通用串行总线
3. 增加章节：CAN_FD

● 2025/10/09

1. 《STC32G144K246 单片机原理及应用》初版（初稿框架，逐章校核及详细测试中）