



内置 EEPROM A/D Flash 单片机

HT66F0195

版本 : V1.20 日期 : 2020-09-11

www.holtek.com

目录

| | |
|----------------------------|-----------|
| 特性 | 6 |
| CPU 特性 | 6 |
| 周边特性 | 6 |
| 概述 | 7 |
| 方框图 | 7 |
| 引脚图 | 8 |
| 引脚说明 | 9 |
| 极限参数 | 13 |
| 直流电气特性 | 14 |
| 交流电气特性 | 16 |
| HIRC 电气特性 | 17 |
| A/D 转换器电气特性 | 17 |
| LVD/LVR 电气特性 | 18 |
| 内部参考电压电气特性 | 18 |
| 比较器电气特性 | 19 |
| 软件控制的 LCD 驱动器电气特性 | 19 |
| 上电复位电气特性 | 19 |
| 系统结构 | 20 |
| 时序和流水线结构 | 20 |
| 程序计数器 | 21 |
| 堆栈 | 21 |
| 算术逻辑单元 – ALU | 22 |
| Flash 程序存储器 | 22 |
| 结构 | 22 |
| 特殊向量 | 23 |
| 查表 | 23 |
| 查表范例 | 23 |
| 在线烧录 – ICP | 24 |
| 片上调试 – OCDS | 25 |
| 数据存储器 | 25 |
| 结构 | 25 |
| 通用数据存储器 | 26 |
| 特殊功能数据存储器 | 26 |
| 特殊功能寄存器 | 28 |
| 间接寻址寄存器 – IAR0, IAR1 | 28 |
| 存储器指针 – MP0, MP1 | 28 |
| 存储区指针 – BP | 28 |
| 累加器 – ACC | 29 |
| 程序计数器低字节寄存器 – PCL | 29 |

| | |
|--------------------------------|-----------|
| 表格寄存器 – TBLP, TBHP, TBLH | 29 |
| 状态寄存器 – STATUS | 29 |
| EEPROM 数据存储器 | 31 |
| EEPROM 数据存储器结构 | 31 |
| EEPROM 寄存器 | 31 |
| 从 EEPROM 中读取数据 | 32 |
| 写数据到 EEPROM | 32 |
| 写保护 | 33 |
| EEPROM 中断 | 33 |
| 编程注意事项 | 33 |
| 振荡器 | 34 |
| 振荡器概述 | 34 |
| 系统时钟配置 | 34 |
| 外部晶体 / 陶瓷振荡器 – HXT | 35 |
| 内部高速 RC 振荡器 – HIRC | 36 |
| 外部 32.768kHz 晶体振荡器 – LXT | 36 |
| 内部 32kHz 振荡器 – LIRC | 37 |
| 辅助振荡器 | 37 |
| 工作模式和系统时钟 | 37 |
| 系统时钟 | 37 |
| 系统工作模式 | 38 |
| 控制寄存器 | 39 |
| 快速唤醒 | 41 |
| 工作模式切换 | 41 |
| 待机电流的注意事项 | 45 |
| 唤醒 | 45 |
| 编程注意事项 | 45 |
| 看门狗定时器 | 46 |
| 看门狗定时器时钟源 | 46 |
| 看门狗定时器控制寄存器 | 46 |
| 看门狗定时器操作 | 47 |
| 复位和初始化 | 48 |
| 复位功能 | 48 |
| 复位初始状态 | 50 |
| 输入 / 输出端口 | 53 |
| 上拉电阻 | 53 |
| PA 口唤醒 | 54 |
| 输入 / 输出端口控制寄存器 | 54 |
| 输入 / 输出端口源电流控制 | 55 |
| 引脚重置功能 | 56 |
| 输入 / 输出引脚结构 | 57 |
| 编程注意事项 | 57 |
| 定时器模块 – TM | 58 |
| 简介 | 58 |

| | |
|---------------------------|------------|
| TM 操作 | 58 |
| TM 时钟源 | 58 |
| TM 中断 | 58 |
| TM 外部引脚 | 58 |
| TM 输入 / 输出引脚控制寄存器 | 59 |
| 编程注意事项 | 61 |
| 简易型 TM – CTM..... | 62 |
| 简易型 TM 操作 | 62 |
| 简易型 TM 寄存器介绍 | 62 |
| 简易型 TM 工作模式 | 66 |
| 标准型 TM – STM | 72 |
| 标准型 TM 操作 | 72 |
| 标准型 TM 寄存器介绍 | 72 |
| 标准型 TM 工作模式 | 76 |
| 周期型 TM – PTM | 84 |
| 周期型 TM 操作 | 84 |
| 周期型 TM 寄存器介绍 | 84 |
| 周期型 TM 工作模式 | 88 |
| A/D 转换器..... | 96 |
| A/D 简介 | 96 |
| A/D 转换寄存器介绍 | 96 |
| A/D 转换器参考电压 | 101 |
| A/D 转换器输入信号 | 101 |
| A/D 操作 | 102 |
| A/D 转换率及时序图 | 103 |
| A/D 转换步骤 | 103 |
| 编程注意事项 | 104 |
| A/D 转换功能 | 104 |
| A/D 转换应用范例 | 105 |
| 比较器 | 107 |
| 比较器操作 | 107 |
| 比较器中断 | 108 |
| 编程注意事项 | 108 |
| 串行接口模块 – SIM..... | 109 |
| SPI 接口 | 109 |
| I ² C 接口 | 116 |
| UART 接口 | 126 |
| UART 外部引脚..... | 126 |
| UART 数据传输方案..... | 127 |
| UART 状态和控制寄存器..... | 127 |
| 波特率发生器 | 131 |
| UART 模块的设置与控制..... | 132 |
| UART 发送器..... | 133 |
| UART 接收器..... | 134 |

| | |
|-----------------------------------|------------|
| 接收错误处理 | 135 |
| UART 模块中断结构..... | 136 |
| UART 模块暂停和唤醒..... | 137 |
| 带 SCOM 和 SSEG 功能的 LCD..... | 138 |
| LCD 操作 | 138 |
| LCD 控制寄存器 | 139 |
| 低电压检测 – LVD | 143 |
| LVD 寄存器 | 143 |
| LVD 操作 | 143 |
| 中断 | 144 |
| 中断寄存器 | 144 |
| 中断操作 | 149 |
| 外部中断 | 149 |
| 比较器中断 | 150 |
| 多功能中断 | 150 |
| A/D 转换器中断 | 150 |
| 时基中断 | 150 |
| 串行接口模块中断 | 151 |
| UART 传输中断..... | 152 |
| LVD 中断 | 152 |
| EEPROM 中断 | 152 |
| TM 中断 | 152 |
| 中断唤醒功能 | 152 |
| 编程注意事项 | 153 |
| 配置选项 | 153 |
| 应用电路 | 154 |
| 指令集 | 155 |
| 简介 | 155 |
| 指令周期 | 155 |
| 数据的传送 | 155 |
| 算术运算 | 155 |
| 逻辑和移位运算 | 155 |
| 分支和控制转换 | 156 |
| 位运算 | 156 |
| 查表运算 | 156 |
| 其它运算 | 156 |
| 指令集概要 | 157 |
| 惯例 | 157 |
| 指令定义 | 159 |
| 封装信息 | 170 |
| 24-pin SSOP (150mil) 外形尺寸 | 171 |
| 28-pin SSOP (150mil) 外形尺寸 | 172 |

特性

CPU 特性

- 工作电压：
 - ◆ $f_{SYS}=8\text{MHz}$: 2.2V~5.5V
 - ◆ $f_{SYS}=12\text{MHz}$: 2.7V~5.5V
 - ◆ $f_{SYS}=16\text{MHz}$: 3.3V~5.5V
- $V_{DD}=5\text{V}$, 系统时钟为 16MHz 时, 指令周期为 $0.25\mu\text{s}$
- 提供暂停和唤醒功能, 以降低功耗
- 振荡器类型：
 - ◆ 外部高速晶振 – HXT
 - ◆ 内部高速 RC – HIRC
 - ◆ 外部低速 32.768kHz 晶振 – LXT
 - ◆ 内部 32kHz RC – LIRC
- 多种工作模式: 正常、低速、空闲和休眠
- 内部集成的振荡器无需外接元件
- 所有指令都可在 1 或 2 个指令周期内完成
- 查表指令
- 61 条指令
- 8 层硬件堆栈
- 位操作指令

周边特性

- Flash 程序存储器: $8\text{K}\times 16$
- RAM 数据存储器: 512×8
- True EEPROM 存储器: 128×8
- 看门狗定时器功能
- 多达 26 个双向 I/O 口
- 两个与 I/O 口共用的外部中断引脚
- 可编程 I/O 口源电流可用于 LED 驱动应用
- 多个定时器模块用于时间测量、捕捉输入、比较匹配输出、PWM 输出及单脉冲输出
- 串行接口模块 – SIM, 用于 SPI 或 I²C 通信
- 全双工通用异步收发器接口 – UART
- 软件控制 6 个 SCOM/SSEG 口和 18 个 SSEG 口的 1/3 bias LCD 驱动
- 12 个外部通道 12-bit 分辨率 A/D 转换器
- 一个比较器功能
- 双时基功能用以产生固定的时间中断信号
- 低电压复位功能
- 低电压检测功能

- Flash 程序存储器烧录可达 10,000 次
- Flash 程序存储器数据可保存 10 年以上
- True EEPROM 数据存储器烧录可达 100,000 次
- True EEPROM 数据存储器数据可保存 10 年以上
- 多封装类型：24/28-pin SSOP

概述

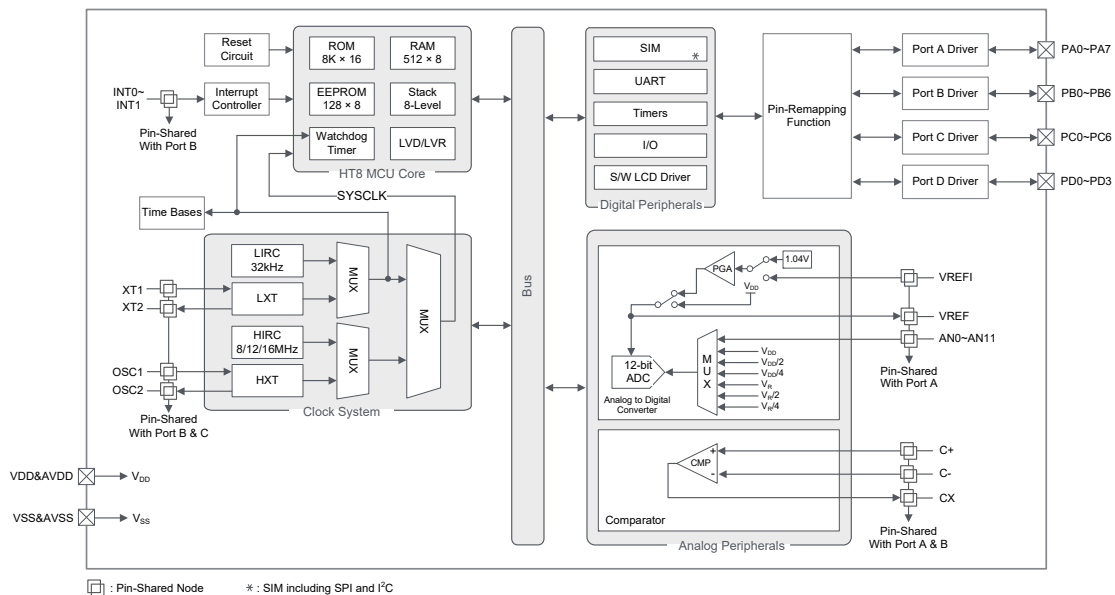
该单片机是一款具有 8 位高性能精简指令集的 Flash 单片机。此单片机具有一系列的功能和特性，其 Flash 存储器可多次编程的特性给用户提供了较大的方便。存储器方面，还包含了一个 RAM 数据存储器和一个可用于存储序号、校准数据等非易失性数据的 True EEPROM 存储器。

在模拟特性方面，该单片机包含一个多通道 12 位 A/D 转换器和一个比较器功能。还带有多个较为灵活的定时器模块，可提供定时功能、脉冲产生功能及 PWM 产生功能。内建完整的 SPI、I²C 和 UART 功能，这三种流行的接口为设计人员提供了易与外部硬件通信的方式。内部看门狗定时器、低电压复位和低电压检测等保护特性，外加优秀的抗干扰和 ESD 保护性能，确保单片机在恶劣的电磁干扰环境下可靠地运行。

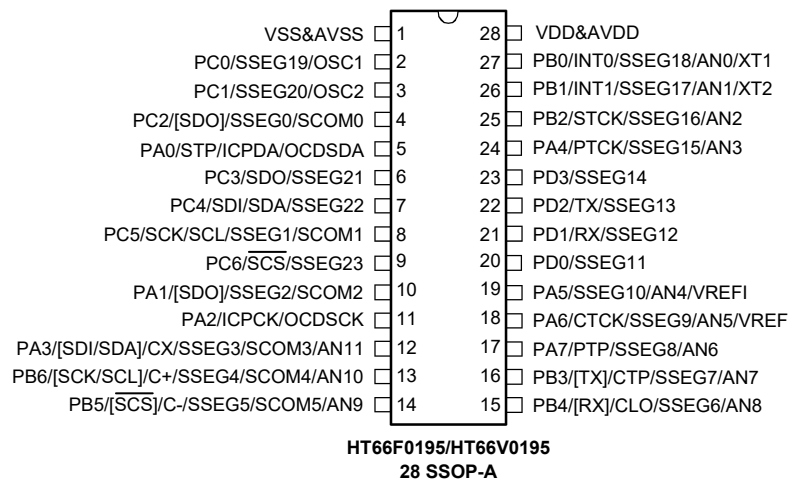
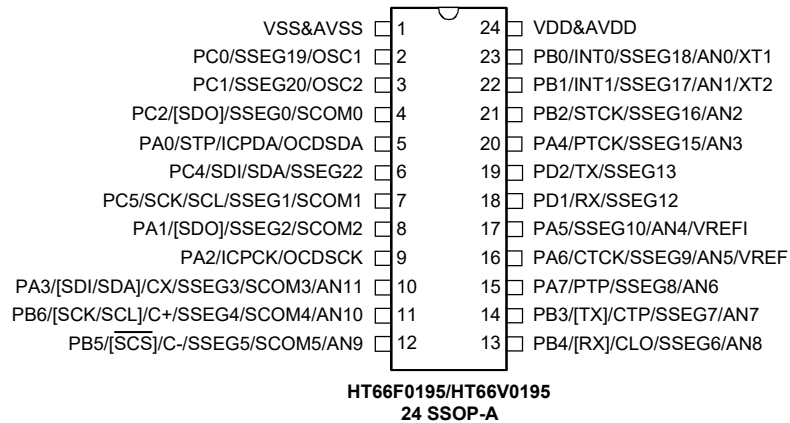
该单片机提供了丰富的内部和外部高速、低速振荡器功能选项，且内建完整的系统振荡器，无需外围元器件。其在不同工作模式之间动态切换的能力，为用户提供了一个优化单片机操作和减少功耗的手段。

外加 I/O 使用灵活、时基功能等其它特性，使该单片机可以广泛应用于各种产品中，例如电子测量仪器、环境监控、手持式测量工具、家庭应用、电子控制工具、马达控制等方面。

方框图



引脚图



- 注：1. 若共用的引脚同时有多种输出，“/”号右侧的引脚名具有更高的优先级。
 2. VDD&AVDD 指的是 VDD 和 AVDD 为同一引脚位置引出。VSS&AVSS 指的是 VSS 和 AVSS 为同一引脚位置引出。
 3. OCSDA 和 OCDSCK 引脚为 OCDS 专用引脚，仅适用于 HT66V0195 芯片，OCDS EV 芯片 HT66V0195 用于 HT66F0195 单片机仿真。
 4. 在较小封装类型中可能含有未引出的引脚，需合理设置其状态以避免输入浮空造成额外耗电，详见“待机电流注意事项”和“输入/输出端口”章节。

引脚说明

每个引脚的功能如下表所列。然而，引脚配置的详细内容请见规格书的其它章节。此引脚说明是针对最大封装芯片而言的，对于小封装的芯片可能不具有部分引脚和功能。

| 引脚名称 | 功能 | OPT | I/T | O/T | 说明 |
|---|--------|------------------|-----|------|----------------------------|
| PA0/STP/ICPDA/ OCSDSA | PA0 | PAWU PAPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能 |
| | STP | TMPC | ST | CMOS | STM 输入 / 输出 |
| | ICPDA | — | ST | CMOS | ICP 数据 / 地址 |
| | OCSDSA | — | ST | CMOS | OCDS 数据 / 地址，仅用于 EV 芯片 |
| PA1/[SDO]/SSEG2/ SCOM2 | PA1 | PAWU PAPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能 |
| | [SDO] | SIMC0 IFS | — | CMOS | SPI 数据输出 |
| | SSEG2 | SLCDC0 SLCDC1 | — | SSEG | 软件控制 LCD SEG 输出 |
| | SCOM2 | SLCDC0 SLCDC1 | — | SCOM | 软件控制 LCD COM 输出 |
| PA2/ICPCK/ OCDSCK | PA2 | PAWU PAPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能 |
| | ICPCK | — | ST | CMOS | ICP 时钟 |
| | OCDSCK | — | ST | — | OCDS 时钟，仅用于 EV 芯片 |
| PA3/[SDI]/[SDA]/CX/ SSEG3/SCOM3/ AN11 | PA3 | PAWU PAPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能 |
| | [SDI] | SIMC0 IFS | ST | — | SPI 数据输入 |
| | [SDA] | SIMC0 IFS | ST | NMOS | I ² C 地址 / 数据线 |
| | CX | CPC | — | CMOS | 比较器输出 |
| | SSEG3 | SLCDC0 SLCDC1 | — | SSEG | 软件控制 LCD SEG 输出 |
| | SCOM3 | SLCDC0 SLCDC1 | — | SCOM | 软件控制 LCD COM 输出 |
| | AN11 | ACERH | AN | — | A/D 转换器模拟输入 |
| PA4/PTCK/SSEG15/ AN3 | PA4 | PAWU PAPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能 |
| | PTCK | PTMC0 | ST | — | PTM 时钟输入 |
| | SSEG15 | SLCDC3 | — | SSEG | 软件控制 LCD SEG 输出 |
| | AN3 | ACERL | AN | — | A/D 转换器模拟输入 |
| PA5/SSEG10/AN4/ VREFI | PA5 | PAWU PAPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能 |
| | SSEG10 | SLCDC2 | — | SSEG | 软件控制 LCD SEG 输出 |
| | AN4 | ACERL | AN | — | A/D 转换器模拟输入 |
| | VREFI | SADC2 | AN | — | A/D 转换器 PGA 电压输入 |

| 引脚名称 | 功能 | OPT | I/T | O/T | 说明 |
|-----------------------------|--------|---------------------|-----|------|----------------------------|
| PA6/CTCK/SSEG9/ AN5/VREF | PA6 | PAWU PAPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能 |
| | CTCK | CTMC0 | ST | — | CTM 时钟输入 |
| | SSEG9 | SLCDC2 | — | SSEG | 软件控制 LCD SEG 输出 |
| | AN5 | ACERL | AN | — | A/D 转换器模拟输入 |
| | VREF | SADC2 | — | AN | A/D 转换器参考电压输出 |
| PA7/PTP/SSEG8/ AN6 | PA7 | PAWU PAPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能 |
| | PTP | TMPC | ST | CMOS | PTM 输入 / 输出 |
| | SSEG8 | SLCDC2 | — | SSEG | 软件控制 LCD SEG 输出 |
| | AN6 | ACERL | AN | — | A/D 转换器模拟输入 |
| PB0/INT0/SSEG18/ AN0/XT1 | PB0 | PBPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻 |
| | INT0 | INTEG INTC0 | ST | — | 外部中断 0 |
| | SSEG18 | SLCDC3 | — | SSEG | 软件控制 LCD SEG 输出 |
| | AN0 | ACERL | AN | — | A/D 转换器模拟输入 |
| | XT1 | CO | LXT | — | LXT 振荡器引脚 |
| PB1/INT1/SSEG17/ AN1/XT2 | PB1 | PBPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻 |
| | INT1 | INTEG INTC2 | ST | — | 外部中断 1 |
| | SSEG17 | SLCDC3 | — | SSEG | 软件控制 LCD SEG 输出 |
| | AN1 | ACERL | AN | — | A/D 转换器模拟输入 |
| | XT2 | CO | — | LXT | LXT 振荡器引脚 |
| PB2/STCK/SSEG16/ AN2 | PB2 | PBPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻 |
| | STCK | STMC0 | ST | — | STM 时钟输入 |
| | SSEG16 | SLCDC3 | — | SSEG | 软件控制 LCD SEG 输出 |
| | AN2 | ACERL | AN | — | A/D 转换器模拟输入 |
| PB3/[TX]/CTP/ SSEG7/AN7 | PB3 | PBPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻 |
| | [TX] | UCR1 UCR2 IFS | — | CMOS | UART TX 串行数据输出 |
| | CTP | TMPC | — | CMOS | CTM 输出 |
| | SSEG7 | SLCDC2 | — | SSEG | 软件控制 LCD SEG 输出 |
| | AN7 | ACERL | AN | — | A/D 转换器模拟输入 |

| 引脚名称 | 功能 | OPT | I/T | O/T | 说明 |
|---|-------------------------|-----------------------|-----|------|-----------------------|
| PB4/[RX]/CLO/ SSEG6/AN8 | PB4 | PBPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻 |
| | [RX] | UCR1 UCR2 IFS | ST | — | UART RX 串行数据输入 |
| | CLO | TMPC | — | CMOS | 系统时钟输出 |
| | SSEG6 | SLCDC2 | — | SSEG | 软件控制 LCD SEG 输出 |
| | AN8 | ACERH | AN | — | A/D 转换器模拟输入 |
| PB5/[SCS]/C-/ SSEG5/SCOM5/AN9 | PB5 | PBPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻 |
| | $\overline{\text{SCS}}$ | SIMC0 SIMC2 IFS | ST | CMOS | SPI 从机选择 |
| | C- | CPC | AN | — | 比较器负极输入 |
| | SSEG5 | SLCDC1 | — | SSEG | 软件控制 LCD SEG 输出 |
| | SCOM5 | SLCDC1 | — | SCOM | 软件控制 LCD COM 输出 |
| | AN9 | ACERH | AN | — | A/D 转换器模拟输入 |
| PB6/[SCK/SCL]/C+/ SSEG4/SCOM4/ AN10 | PB6 | PBPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻 |
| | [SCK] | SIMC0 IFS | ST | CMOS | SPI 串行时钟 |
| | [SCL] | SIMC0 IFS | ST | NMOS | I ² C 时钟线 |
| | C+ | CPC | AN | — | 比较器正极输入 |
| | SSEG4 | SLCDC1 | — | SSEG | 软件控制 LCD SEG 输出 |
| | SCOM4 | SLCDC1 | — | SCOM | 软件控制 LCD COM 输出 |
| | AN10 | ACERH | AN | — | A/D 转换器模拟输入 |
| PC0/SSEG19/OSC1 | PC0 | PCPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻 |
| | SSEG19 | SLCDC3 | — | SSEG | 软件控制 LCD SEG 输出 |
| | OSC1 | CO | HXT | — | HXT 振荡器引脚 |
| PC1/SSEG20/OSC2 | PC1 | PCPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻 |
| | SSEG20 | SLCDC3 | — | SSEG | 软件控制 LCD SEG 输出 |
| | OSC2 | CO | — | HXT | HXT 振荡器引脚 |
| PC2/[SDO]/SSEG0/ SCOM0 | PC2 | PCPU | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻 |
| | [SDO] | SIMC0 IFS | — | CMOS | SPI 数据输出 |
| | SSEG0 | SLCDC0 SLCDC1 | — | SSEG | 软件控制 LCD SEG 输出 |
| | SCOM0 | SLCDC0 SLCDC1 | — | SCOM | 软件控制 LCD COM 输出 |

| 引脚名称 | 功能 | OPT | I/T | O/T | 说明 |
|-------------------------------|------------------|--------------------|-----|------|------------------------|
| PC3/SDO/SSEG21 | PC3 | PCPU | ST | CMOS | 通用 I/O 口, 可通过寄存器设置上拉电阻 |
| | SDO | SIMC0 IFS | — | CMOS | SPI 数据输出 |
| | SSEG21 | SLCDC3 | — | SSEG | 软件控制 LCD SEG 输出 |
| PC4/SDI/SDA/SSEG22 | PC4 | PCPU | ST | CMOS | 通用 I/O 口, 可通过寄存器设置上拉电阻 |
| | SDI | SIMC0 IFS | ST | — | SPI 数据输入 |
| | SDA | SIMC0 IFS | ST | NMOS | I ² C 数据线 |
| | SSEG22 | SLCDC4 | — | SSEG | 软件控制 LCD SEG 输出 |
| PC5/SCK/SCL/SSEG1/SCOM1 | PC5 | PCPU | ST | CMOS | 通用 I/O 口, 可通过寄存器设置上拉电阻 |
| | SCK | SIMC0 IFS | ST | CMOS | SPI 串行时钟 |
| | SCL | SIMC0 IFS | ST | NMOS | I ² C 时钟线 |
| | SSEG1 | SLCDC0 SLCDC1 | — | SSEG | 软件控制 LCD SEG 输出 |
| | SCOM1 | SLCDC0 SLCDC1 | — | SCOM | 软件控制 LCD COM 输出 |
| PC6/ \overline{SCS} /SSEG23 | PC6 | PCPU | ST | CMOS | 通用 I/O 口, 可通过寄存器设置上拉电阻 |
| | \overline{SCS} | SIMC0 SIMC2 IFS | ST | CMOS | SPI 从机选择 |
| | SSEG23 | SLCDC4 | — | SSEG | 软件控制 LCD SEG 输出 |
| PD0/SSEG11 | PD0 | PDPU | ST | CMOS | 通用 I/O 口, 可通过寄存器设置上拉电阻 |
| | SSEG11 | SLCDC2 | — | SSEG | 软件控制 LCD SEG 输出 |
| PD1/RX/SSEG12 | PD1 | PDPU | ST | CMOS | 通用 I/O 口, 可通过寄存器设置上拉电阻 |
| | RX | UCR1 UCR2 IFS | ST | — | UART RX 串行数据输入 |
| | SSEG12 | SLCDC2 | — | SSEG | 软件控制 LCD SEG 输出 |
| PD2/TX/SSEG13 | PD2 | PDPU | ST | CMOS | 通用 I/O 口, 可通过寄存器设置上拉电阻 |
| | TX | UCR1 UCR2 IFS | — | CMOS | UART TX 串行数据输出 |
| | SSEG13 | SLCDC2 | — | SSEG | 软件控制 LCD SEG 输出 |

| 引脚名称 | 功能 | OPT | I/T | O/T | 说明 |
|------------|--------|--------|-----|------|-----------------------|
| PD3/SSEG14 | PD3 | PDPUP | ST | CMOS | 通用 I/O 口，可通过寄存器设置上拉电阻 |
| | SSEG14 | SLCDC3 | — | SSEG | 软件控制 LCD SEG 输出 |
| VDD&AVDD * | VDD | — | PWR | — | 正电源电压 |
| | AVDD | — | PWR | — | 模拟正电源电压 |
| VSS&AVSS * | VSS | — | PWR | — | 正电源电压，接地 |
| | AVSS | — | PWR | — | 模拟负电源电压，接地 |

注：I/T：输入类型； O/T：输出类型；
 OPT：通过配置选项 (CO) 或寄存器选项来配置；
 CO：配置选项； PWR：电源；
 AN：模拟信号； ST：施密特触发输入；
 CMOS：CMOS 输出； NMOS：NMOS 输出；
 SSEG：软件控制 LCD SEG； SCOM：软件控制 LCD COM；
 HXT：高频晶体振荡器； LXT：低频晶体振荡器；
 * AVDD 引脚与 VDD 引脚在内部是同一个引脚，AVSS 与 VSS 在内部是同一个引脚。

极限参数

| | |
|--------------------|----------------------------------|
| 电源供应电压 | $V_{SS}-0.3V \sim V_{SS}+6.0V$ |
| 输入电压 | $V_{SS}-0.3V \sim V_{DD}+0.3V$ |
| 储存温度 | $-50^{\circ}C \sim 125^{\circ}C$ |
| 工作温度 | $-40^{\circ}C \sim 85^{\circ}C$ |
| I_{OL} 总电流..... | 80mA |
| I_{OH} 总电流 | -80mA |
| 总功耗 | 500mW |

注：这里只强调额定功率，超过极限参数所规定的范围将对芯片造成损害，无法预期芯片在上述标示范围外的工作状态，而且若长期在标示范围外的条件下工作，可能影响芯片的可靠性。

直流电气特性

Ta=-40°C~85°C

| 符号 | 参数 | 测试条件 | | 最小 | 典型 | 最大 | 单位 |
|-----------------|-------------|-----------------|--|-----|-----|------|----|
| | | V _{DD} | 条件 | | | | |
| V _{DD} | 工作电压 (HXT) | — | f _{SYS} =f _{HXT} =8MHz | 2.2 | — | 5.5 | V |
| | | | f _{SYS} =f _{HXT} =12MHz | 2.7 | — | 5.5 | |
| | | | f _{SYS} =f _{HXT} =16MHz | 3.3 | — | 5.5 | |
| | 工作电压 (HIRC) | — | f _{SYS} =f _{HIRC} =8MHz | 2.2 | — | 5.5 | V |
| | | | f _{SYS} =f _{HIRC} =12MHz | 2.7 | — | 5.5 | |
| | | | f _{SYS} =f _{HIRC} =16MHz | 3.3 | — | 5.5 | |
| I _{DD} | 工作电流 (HXT) | 3V | f _{SYS} =f _H =f _{HXT} =8MHz, | — | 1.0 | 1.5 | mA |
| | | 5V | 无负载, 所有外设关闭 | — | 2.0 | 3.0 | |
| | | 3V | f _{SYS} =f _H =f _{HXT} =12MHz, | — | 1.5 | 2.75 | mA |
| | | 5V | 无负载, 所有外设关闭 | — | 3.0 | 4.5 | |
| | | 5V | f _{SYS} =f _H =f _{HXT} =16MHz, | — | 4.5 | 7.0 | mA |
| | | 5V | 无负载, 所有外设关闭 | — | 4.5 | 7.0 | |
| | 工作电流 (HIRC) | 3V | f _{SYS} =f _H =f _{HIRC} =8MHz, | — | 1.0 | 1.5 | mA |
| | | 5V | 无负载, 所有外设关闭 | — | 2.0 | 3.0 | |
| | | 3V | f _{SYS} =f _H =f _{HIRC} =12MHz, | — | 1.5 | 2.75 | mA |
| | | 5V | 无负载, 所有外设关闭 | — | 3.0 | 4.5 | |
| | | 5V | f _{SYS} =f _H =f _{HIRC} =16MHz, | — | 4.5 | 7.0 | mA |
| | | 5V | 无负载, 所有外设关闭 | — | 4.5 | 7.0 | |
| | 工作电流 (LXT) | 3V | f _{SYS} =f _{SUB} =f _{LXT} =32.768kHz, | — | 10 | 20 | μA |
| | | 5V | 无负载, 所有外设关闭 | — | 30 | 50 | |
| | 工作电流 (LIRC) | 3V | f _{SYS} =f _{SUB} =f _{LIRC} =32kHz, | — | 10 | 20 | μA |
| | | 5V | 无负载, 所有外设关闭 | — | 30 | 50 | |

| 符号 | 参数 | 测试条件 | | 最小 | 典型 | 最大 | 单位 |
|------------------|------------------------|-----------------|--|--------------------|------|--------------------|----|
| | | V _{DD} | 条件 | | | | |
| I _{STB} | 待机电流 (休眠模式 0) | 3V | 无负载, 所有外设关闭, | — | — | 1 | μA |
| | | 5V | WDT off | — | — | 2 | |
| | 待机电流 (休眠模式 1) | 3V | 无负载, 所有外设关闭, | — | — | 3 | μA |
| | | 5V | WDT on | — | — | 5 | |
| | 待机电流 (空闲模式 0) | 3V | 无负载, 所有外设关闭, | — | 3 | 5 | μA |
| | | 5V | f _{SUB} on | — | 5 | 10 | |
| | 待机电流 (空闲模式 1, HXT) | 3V | f _{sys} =f _{HXT} =8MHz, f _{SUB} on, | — | 0.5 | 1 | mA |
| | | 5V | 无负载, 所有外设关闭 | — | 1 | 2 | |
| | | 3V | f _{sys} =f _{HXT} =12MHz, f _{SUB} on, | — | 0.6 | 1.2 | mA |
| | | 5V | 无负载, 所有外设关闭 | — | 1.2 | 2.4 | |
| | 待机电流 (空闲模式 1, HIRC) | 3V | f _{sys} =f _{HIRC} =8MHz, f _{SUB} on, | — | 0.8 | 1.6 | mA |
| | | 5V | 无负载, 所有外设关闭 | — | 1.0 | 2.0 | |
| | | 3V | f _{sys} =f _{HIRC} =12MHz, f _{SUB} on, | — | 1.2 | 2.4 | mA |
| | | 5V | 无负载, 所有外设关闭 | — | 1.5 | 3.0 | |
| V _{IL} | 输入 / 输出口低电平输入电压 | 5V | — | 0 | — | 1.5 | V |
| | | — | — | 0 | — | 0.2V _{DD} | |
| V _{IH} | 输入 / 输出口高电平输入电压 | 5V | — | 3.5 | — | 5.0 | V |
| | | — | — | 0.8V _{DD} | — | V _{DD} | |
| I _{OL} | 输入 / 输出口灌电流 | 3V | V _{OL} =0.1V _{DD} | 16 | 32 | — | mA |
| | | 5V | | 32 | 64 | — | |
| I _{OH} | 输入 / 输出口源电流 | 3V | V _{OH} =0.9V _{DD} , SLEDCn[m+1, m]=00, | -1.0 | -2.0 | — | mA |
| | | 5V | n=0, 1; m=0, 2, 4, 6 | -2.0 | -4.0 | — | |
| | | 3V | V _{OH} =0.9V _{DD} , SLEDCn[m+1, m]=01, | -1.75 | -3.5 | — | mA |
| | | 5V | n=0, 1; m=0, 2, 4, 6 | -3.5 | -7.0 | — | |
| | | 3V | V _{OH} =0.9V _{DD} , SLEDCn[m+1, m]=10, | -2.5 | -5.0 | — | mA |
| | | 5V | n=0, 1; m=0, 2, 4, 6 | -5.0 | -10 | — | |
| R _{PH} | 输入 / 输出口上拉电阻 | 3V | — | 20 | 60 | 100 | kΩ |
| | | 5V | — | 10 | 30 | 50 | |

交流电气特性

Ta = -40°C~85°C，除非另有说明

| 符号 | 参数 | 测试条件 | | 最小 | 典型 | 最大 | 单位 |
|-------------------------|---|--|---|------|--------|----------------|-------------------|
| | | V _{DD} | 条件 | | | | |
| f _{SYS} | 系统时钟 (HXT) | 2.2V~5.5V | f _{SYS} =f _{HXT} =8MHz | 0.4 | — | 8 | MHz |
| | | 2.7V~5.5V | f _{SYS} =f _{HXT} =12MHz | 0.4 | — | 12 | MHz |
| | | 3.3V~5.5V | f _{SYS} =f _{HXT} =16MHz | 0.4 | — | 16 | MHz |
| | 系统时钟 (HIRC) | 2.2V~5.5V | f _{SYS} =f _{HIRC} =8MHz | — | 8 | — | MHz |
| | | 2.7V~5.5V | f _{SYS} =f _{HIRC} =12MHz | — | 12 | — | MHz |
| | | 3.3V~5.5V | f _{SYS} =f _{HIRC} =16MHz | — | 16 | — | MHz |
| | 系统时钟 (LXT) | 2.2V~5.5V | f _{SYS} =f _{LXT} =32.768kHz | — | 32.768 | — | kHz |
| 系统时钟 (LIRC) | 2.2V~5.5V | f _{SYS} =f _{LIRC} =32kHz | — | 32 | — | kHz | |
| f _{LIRC} | 内部低速 RC 振荡器 (LIRC) | 5V | Ta=25°C | -2% | 32 | +2% | kHz |
| | | 2.2V~5.5V | Ta=-40°C ~ 85°C | -10% | 32 | +10% | |
| t _{START} | LIRC 启动时间 | — | — | — | — | 100 | μs |
| t _{TCK} | xTCK 引脚最小输入脉宽 | — | — | 0.3 | — | — | μs |
| t _{INT} | 中断引脚最小输入脉宽 | — | — | 10 | — | — | μs |
| t _{SST} | 系统启动时间 (从 f _{SYS} Off 条件下唤醒) | — | f _{SYS} =f _{HXT} ~f _{HXT} /64 | 128 | — | — | t _{HXT} |
| | | — | f _{SYS} =f _{HIRC} ~f _{HIRC} /64 | 16 | — | — | t _{HIRC} |
| | | — | f _{SYS} =f _{LXT} | 128 | — | — | t _{LXT} |
| | | — | f _{SYS} =f _{LIRC} | 2 | — | — | t _{LIRC} |
| | 系统启动时间 (从 f _{SYS} On 条件下唤醒) | — | f _{SYS} =f _H ~f _H /64, f _H =f _{HXT} 或 f _{HIRC} | 2 | — | — | t _H |
| | | — | f _{SYS} =f _{LXT} 或 f _{LIRC} | 2 | — | — | t _{SUB} |
| 系统启动时间 (WDT 溢出硬件冷复位) | — | — | 0 | — | — | t _H | |
| t _{RSTD} | 系统复位延迟时间 (上电复位, LVR 硬件复位, LVRC/WDT 软件复位) | — | — | 25 | 50 | 100 | ms |
| | 系统复位延迟时间 (WDT 硬件复位) | — | — | 8.3 | 16.7 | 33.3 | ms |
| t _{EERD} | EEPROM 读周期 | — | — | — | — | 4 | t _{SYS} |
| t _{EEWR} | EEPROM 写周期 | — | — | — | 4 | 6 | ms |

注: t_{SYS}=1/f_{SYS}。

HIRC 电气特性

程序烧录时，烧录器会调整 HIRC 振荡器使其工作在用户选择的 HIRC 频率和工作电压 (3V 或 5V) 条件下。

| 符号 | 参数 | 测试条件 | | 最小 | 典型 | 最大 | 单位 |
|----------------------|----------------------|-----------------|------------|-------|-------|-------|-----|
| | | V _{DD} | 温度 | | | | |
| f _{HIRC} | 烧录器调整 HIRC 频率为 8MHz | 3V/5V | 25°C | -1% | 8 | +1% | MHz |
| | | | -40°C~85°C | -2% | 8 | +2% | |
| | | 2.2V~5.5V | 25°C | -2.5% | 8 | +2.5% | |
| | | | -40°C~85°C | -3% | 8 | +3% | |
| | 烧录器调整 HIRC 频率为 12MHz | 3V/5V | 25°C | -1% | 12 | +1% | MHz |
| | | | -40°C~85°C | -2% | 12 | +2% | |
| 2.7V~5.5V | | 25°C | -2.5% | 12 | +2.5% | | |
| | | -40°C~85°C | -3% | 12 | +3% | | |
| 烧录器调整 HIRC 频率为 16MHz | 5V | 25°C | -1% | 16 | +1% | MHz | |
| | | -40°C~85°C | -2% | 16 | +2% | | |
| | 3.3V~5.5V | 25°C | -2.5% | 16 | +2.5% | | |
| | | -40°C~85°C | -3% | 16 | +3% | | |

注：1. 烧录器可在 3V/5V 这两个可选的固定电压下对 HIRC 频率进行调整，在此提供 V_{DD}=3V/5V 时的参数值。

2. 3V/5V 表格下面提供的是全压条件下的参数值。对于电压范围在 2.2V~3.6V 的应用，建议调整电压固定在 3V；对于电压范围在 3.3V~5.5V 的应用，建议调整电压固定在 5V。

3. 表格中提供的最小和最大误差值仅在对应的烧录器调整 HIRC 振荡器频率下有效。

A/D 转换器电气特性

T_a = -40°C~85°C

| 符号 | 参数 | 测试条件 | | 最小 | 典型 | 最大 | 单位 |
|-------------------|----------------|-----------------|--|-----|-----|------------------|-----|
| | | V _{DD} | 条件 | | | | |
| V _{DD} | 工作电压 | — | — | 2.2 | — | 5.5 | V |
| V _{ADI} | 输入电压 | — | — | 0 | — | V _{REF} | V |
| V _{REF} | 参考电压 | — | — | 2 | — | V _{DD} | V |
| DNL | 非线性微分误差 | 3V | V _{REF} =V _{DD} , t _{ADCK} =0.5μs | -3 | — | +3 | LSB |
| | | | | | | | |
| | | 3V | V _{REF} =V _{DD} , t _{ADCK} =10μs | | | | |
| | | | | | | | |
| INL | 非线性积分误差 | 3V | V _{REF} =V _{DD} , t _{ADCK} =0.5μs | -4 | — | +4 | LSB |
| | | | | | | | |
| | | 3V | V _{REF} =V _{DD} , t _{ADCK} =10μs | | | | |
| | | | | | | | |
| I _{ADC} | A/D 转换器使能的额外电流 | 3V | 无负载, t _{ADCK} =0.5μs | — | 1.0 | 2.0 | mA |
| | | 5V | | — | 1.5 | 3.0 | |
| t _{ADCK} | 时钟周期 | — | — | 0.5 | — | 10 | μs |

| 符号 | 参数 | 测试条件 | | 最小 | 典型 | 最大 | 单位 |
|--------------------|-------------------------|-----------------|----|----|----|----|-------------------|
| | | V _{DD} | 条件 | | | | |
| t _{ADC} | A/D 转换时间 (包括采样和保持时间) | — | — | — | 16 | — | t _{ADCK} |
| t _{ON2ST} | A/D 转换器 On-to-Start 时间 | — | — | 4 | — | — | μs |

LVD/LVR 电气特性

T_a = -40°C ~ 85°C

| 符号 | 参数 | 测试条件 | | 最小 | 典型 | 最大 | 单位 |
|-------------------|-------------------------|-----------------|---------------------------------|-----|------|-----|----|
| | | V _{DD} | 条件 | | | | |
| V _{LVR} | 低电压复位电压 | — | LVR 使能, 电压选择 2.1V | -5% | 2.1 | +5% | V |
| | | | LVR 使能, 电压选择 2.55V | | 2.55 | | |
| | | | LVR 使能, 电压选择 3.15V | | 3.15 | | |
| | | | LVR 使能, 电压选择 3.8V | | 3.8 | | |
| V _{LVD} | 低电压检测电压 | — | LVD 使能, 电压选择 2.0V | -5% | 2.0 | +5% | V |
| | | | LVD 使能, 电压选择 2.2V | | 2.2 | | |
| | | | LVD 使能, 电压选择 2.4V | | 2.4 | | |
| | | | LVD 使能, 电压选择 2.7V | | 2.7 | | |
| | | | LVD 使能, 电压选择 3.0V | | 3.0 | | |
| | | | LVD 使能, 电压选择 3.3V | | 3.3 | | |
| | | | LVD 使能, 电压选择 3.6V | | 3.6 | | |
| LVD 使能, 电压选择 4.0V | 4.0 | | | | | | |
| I _{OP} | 工作电流 | 5V | LVD 使能, LVR 使能, VBGEN=0 | — | 20 | 25 | μA |
| | | 5V | LVD 使能, LVR 使能, VBGEN=1 | — | 180 | 200 | |
| t _{LVDS} | LVDO 稳定时间 | — | LVR 使能, VBGEN=0, LVD 关闭 → 打开 | — | — | 15 | μs |
| | | — | LVR 除能, VBGEN=0, LVD 关闭 → 打开 | — | — | 150 | |
| t _{LVR} | 产生 LVR 复位的低电压 最短保持时间 | — | — | 120 | 240 | 480 | μs |
| t _{LVD} | 产生 LVD 中断的低电压 最短保持时间 | — | — | 60 | 120 | 240 | μs |

内部参考电压电气特性

T_a = -40°C ~ 85°C

| 符号 | 参数 | 测试条件 | | 最小 | 典型 | 最大 | 单位 |
|------------------|------------------------|-----------------|-----|-----|------|-----|----|
| | | V _{DD} | 条件 | | | | |
| V _{BG} | Bandgap 参考电压 | — | — | -3% | 1.04 | +3% | V |
| t _{BGS} | V _{BG} 打开稳定时间 | — | 无负载 | — | — | 150 | μs |

注: V_{BG} 电压用作 A/D 转换器 PGA 输入。

比较器电气特性

Ta = -40°C~85°C

| 符号 | 参数 | 测试条件 | | 最小 | 典型 | 最大 | 单位 |
|------------------|-----------|-----------------|----------|-----------------|-----|----------------------|----|
| | | V _{DD} | 条件 | | | | |
| V _{DD} | 工作电压 | — | — | 2.2 | — | 5.5 | V |
| I _{CMP} | 比较器工作电流 | 3V | — | — | 37 | 56 | μA |
| | | 5V | — | — | 130 | 200 | |
| V _{OS} | 比较器输入失调电压 | 3V | — | -10 | — | 10 | mV |
| | | 5V | — | -10 | — | 10 | |
| V _{CM} | 比较器共模电压范围 | — | — | V _{SS} | — | V _{DD} -1.4 | V |
| A _{OL} | 开环增益 | 3V | — | 60 | 80 | — | dB |
| | | 5V | — | 60 | 80 | — | |
| V _{HYS} | 迟滞宽度 | 3V | CHYEN=0 | 0 | 0 | 5 | mV |
| | | 5V | | 0 | 0 | 5 | |
| | | 3V | CHYEN=1 | 40 | 60 | 80 | mV |
| | | 5V | | 40 | 60 | 80 | |
| t _{RP} | 响应时间 | 3V | 100mV 过载 | — | 370 | 560 | ns |
| | | 5V | | — | 370 | 560 | |

注：所有的测量都是在比较器正输入电压等于 (V_{DD}-1.4)/2 且保持不变的条件下。

软件控制的 LCD 驱动器电气特性

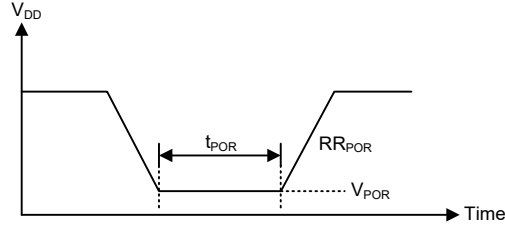
Ta=-40°C~85°C

| 符号 | 参数 | 测试条件 | | 最小 | 典型 | 最大 | 单位 |
|--------------------|--|-----------------|--------------|-------|------|-------|-----------------|
| | | V _{DD} | 条件 | | | | |
| I _{BIAS} | 偏置电流 | 5V | ISEL[1:0]=00 | 4.2 | 8.3 | 13 | μA |
| | | | ISEL[1:0]=01 | 8.3 | 16.7 | 25 | |
| | | | ISEL[1:0]=10 | 25 | 50 | 75 | |
| | | | ISEL[1:0]=11 | 50 | 100 | 150 | |
| V _{LCD_H} | (2/3)V _{DD} 电压用于 LCD SCOM/SSEG 输出 | 2.2V~5.5V | 无负载 | 0.645 | 0.67 | 0.695 | V _{DD} |
| V _{LCD_L} | (1/3)V _{DD} 电压用于 LCD SCOM/SSEG 输出 | 2.2V~5.5V | 无负载 | 0.305 | 0.33 | 0.355 | V _{DD} |

上电复位电气特性

Ta=25°C

| 符号 | 参数 | 测试条件 | | 最小 | 典型 | 最大 | 单位 |
|-------------------|--|-----------------|----|-------|----|-----|------|
| | | V _{DD} | 条件 | | | | |
| V _{POR} | 上电复位电压 | — | — | — | — | 100 | mV |
| RR _{POR} | 上电复位电压速率 | — | — | 0.035 | — | — | V/ms |
| t _{POR} | V _{DD} 保持为 V _{POR} 的最小时间 | — | — | 1 | — | — | ms |

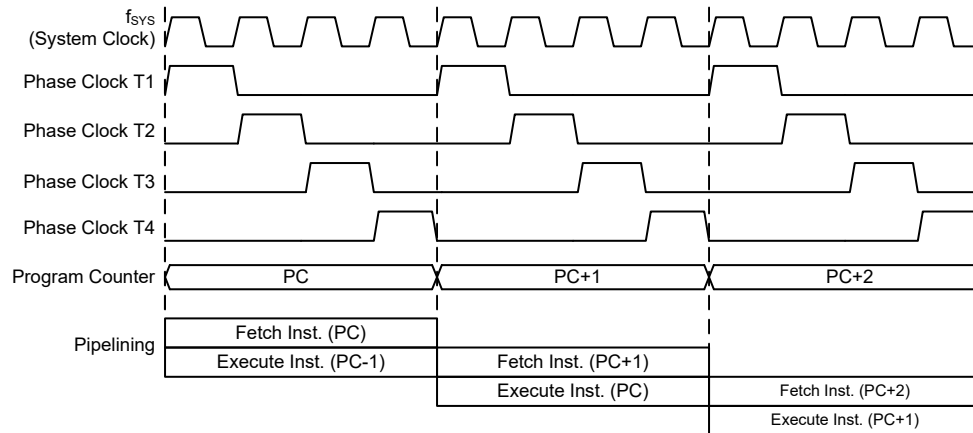


系统结构

内部系统结构是 Holtek 单片机具有良好性能的主要因素。由于采用 RISC 结构，该单片机具有高运算速度和高性能的特点。通过流水线的方式，指令的取得和执行同时进行，此举使得除了跳转和调用指令外，其它指令都能在一个指令周期内完成。8 位 ALU 参与指令集中所有的运算，它可完成算术运算、逻辑运算、移位、递增、递减和分支等功能，而内部的数据路径则是以通过累加器和 ALU 的方式加以简化。有些寄存器在数据存储中被实现，且可以直接或间接寻址。简单的寄存器寻址方式和结构特性，确保了在提供具有较大可靠性和灵活性的 I/O 和 A/D 控制系统时，仅需要少数的外部器件。使得该单片机适用于低成本和大量生产的控制应用。

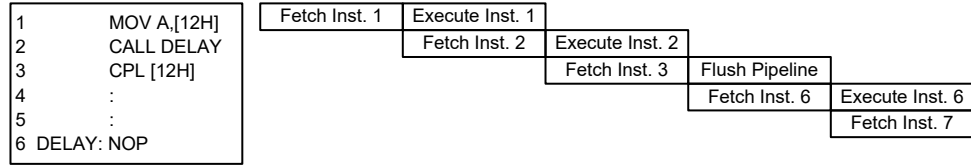
时序和流水线结构

主系统时钟由 HXT、LXT、HIRC 或 LIRC 振荡器提供，它被细分为 T1~T4 四个内部产生的非重叠时序。在 T1 时间，程序计数器自动加一并抓取一条新的指令。剩下的时间 T2~T4 完成译码和执行功能，因此，一个 T1~T4 时钟周期构成一个指令周期。虽然指令的抓取和执行发生在连续的指令周期，但单片机流水线结构会保证指令在一个指令周期内被有效执行。除非程序计数器的内容被改变，如子程序的调用或跳转，在这种情况下指令将需要多一个指令周期的时间去执行。



系统时序和流水线

如果指令牵涉到分支，例如跳转或调用等指令，则需要两个指令周期才能完成指令执行。需要一个额外周期的原因是程序先用一个周期取出实际要跳转或调用的地址，再用另一个周期去实际执行分支动作，因此用户需要特别考虑额外周期的问题，尤其是在执行时间要求较严格的时候。



指令捕捉

程序计数器

在程序执行期间，程序计数器用来指向下一个要执行的指令地址。除了“JMP”和“CALL”指令需要跳转到一个非连续的程序存储器地址之外，它会在每条指令执行完成以后自动加一。只有较低的 8 位，即所谓的程序计数器低字节寄存器 PCL，可以被用户直接读写。

当执行的指令要求跳转到不连续的地址时，如跳转指令、子程序调用、中断或复位等，单片机通过加载所需要的地址到程序寄存器来控制程序，对于条件跳转指令，一旦条件符合，在当前指令执行时取得的下一条指令将会被舍弃，而由一个空指令周期来取代。

| 程序计数器 | |
|----------|-----------|
| 程序计数器高字节 | PCL 寄存器 |
| PC12~PC8 | PCL7~PCL0 |

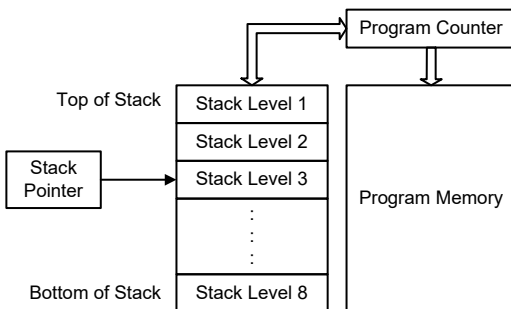
程序计数器

程序计数器的低字节，即程序计数器的低字节寄存器 PCL，可以通过程序控制，且它是可以读取和写入的寄存器。通过直接写入数据到这个寄存器，一个程序短跳转可直接执行，然而只有低字节的操作是有效的，跳转被限制在存储器的当前页中，即 256 个存储器地址范围内，当这样一个程序跳转要执行时，会插入一个空指令周期。程序计数器的低字节可由程序直接进行读取，PCL 的使用可能引起程序跳转，因此需要额外的指令周期。

堆栈

堆栈是一个特殊的存储空间，用来存储程序计数器中的内容。该单片机有多层堆栈，堆栈既不是数据部分也不是程序空间部分，而且它既不是可读取也不是可写入的。当前层由堆栈指针 (SP) 加以指示，同样也是不可读写的。在子程序调用或中断响应服务时，程序计数器的内容被压入到堆栈中。当子程序或中断响应结束时，返回指令 (RET 或 RETI) 使程序计数器从堆栈中重新得到它以前的值。当一个芯片复位后，堆栈指针将指向堆栈顶部。

如果堆栈已满，且有非屏蔽的中断发生，中断请求标志会被置位，但中断响应将被禁止。当堆栈指针减少 (执行 RET 或 RETI)，中断将被响应。这个特性提供程序设计者简单的方法来预防堆栈溢出。然而即使堆栈已满，CALL 指令仍然可以被执行，而造成堆栈溢出。使用时应避免堆栈溢出的情况发生，因为这可能导致不可预期的程序分支指令执行错误。若堆栈溢出，则首个存入堆栈的程序计数器数据将会丢失。



算术逻辑单元 – ALU

算术逻辑单元是单片机中很重要的部分，执行指令集中的算术和逻辑运算。ALU 连接到单片机的数据总线，在接收相关的指令码后执行需要的算术与逻辑操作，并将结果存储在指定的寄存器，当 ALU 计算或操作时，可能导致进位、借位或其它状态的变化，而相关的状态寄存器会因此更新内容以显示这些改变，ALU 所提供的功能如下：

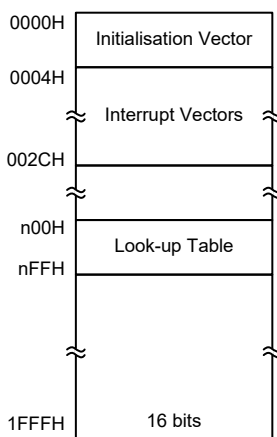
- 算术运算：ADD, ADDM, ADC, ADCM, SUB, SUBM, SBC, SBCM, DAA
- 逻辑运算：AND, OR, XOR, ANDM, ORM, XORM, CPL, CPLA
- 移位运算：RRA, RR, RRCA, RRC, RLA, RL, RLCA, RLC
- 递增和递减：INCA, INC, DECA, DEC
- 分支判断：JMP, SZ, SZA, SNZ, SIZ, SDZ, SIZA, SDZA, CALL, RET, RETI

Flash 程序存储器

程序存储器用来存放用户代码即储存程序。程序存储器为 Flash 类型意味着可以多次重复编程，方便用户使用同一芯片进行程序的修改。使用适当的单片机编程工具，此单片机提供用户灵活便利的调试方法和项目开发规划及更新。

结构

程序存储器的容量为 8K×16 位，程序存储器用程序计数器来寻址，其中也包含数据、表格和中断入口。数据表格可以设定在程序存储器的任何地址，由表格指针来寻址。



程序存储器结构

特殊向量

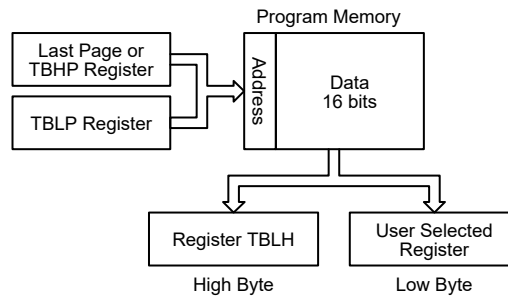
程序存储器内部某些地址保留用做诸如复位和中断入口等特殊用途。地址 0000H 是芯片复位后的程序起始地址。在芯片复位之后，程序将跳到这个地址并开始执行。

查表

程序存储器中的任何地址都可以定义成一个表格，以便储存固定的数据。使用表格时，表格指针必须先行设定，其方式是将表格的地址放在表格指针寄存器 TBLP 和 TBHP 中。这些寄存器定义表格总的地址。

在设定完表格指针后，表格数据可以使用“TABRD [m]”或“TABRDL [m]”指令分别从程序存储器查表读取。当这个指令执行时，程序存储器中表格数据低字节，将被传送到使用者所指定的数据存储器 [m]，程序存储器中表格数据的高字节，则被传送到 TBLH 特殊寄存器，而高字节中未使用的位将被读取为“0”。

下图是查表中寻址 / 数据流程：



查表范例

以下范例说明表格指针和表格数据如何被定义和执行。这个例子使用的表格数据用 ORG 伪指令储存在存储器中。ORG 指令的值“1F00H”指向的地址是 8K 程序存储器中最后一页的起始地址。表格指针低字节寄存器的初始值设为 06H，这可保证从数据表格读取的第一笔数据位于程序存储器地址 1F06H，即最后一页起始地址后的第六个地址。值得注意的是，假如“TABRD [m]”指令被使用，则表格指针指向 TBLP 和 TBHP 寄存器所指定的地址。在这个例子中，表格数据的高字节等于零，而当“TABRD [m]”指令被执行时，此值将会自动的被传送到 TBLH 寄存器。

TBLH 寄存器为只读寄存器，不能重新储存，若主程序和中断服务程序都使用表格读取指令，应该注意它的保护。使用表格读取指令，中断服务程序可能会改变 TBLH 的值，若随后在主程序中再次使用这个值，则会发生错误，因此建议避免同时使用表格读取指令。然而在某些情况下，如果同时使用表格读取指令是不可避免的，则在执行任何主程序的表格读取指令前，中断应该先除能，另外要注意的是所有与表格相关的指令，都需要两个指令周期去完成操作。

表格读取程序范例

```
tempreg1 db ?      ; temporary register #1
tempreg2 db ?      ; temporary register #2
:
:
mov a,06h          ; initialise low table pointer - note that this address
                  ; is referenced
mov tblp,a         ; to the last page or the page that tbhp pointed
```

```

mov a,1Fh          ; initialise high table pointer
mov tbhp,a
:
:
tabrd tempreg1     ; transfers value in table referenced by table pointer,
                  ; data at program memory address "1F06H" transferred to
                  ; tempreg1 and TBLH
dec tblp           ; reduce value of table pointer by one
tabrd tempreg2     ; transfers value in table referenced by table pointer,
                  ; data at program memory address "1F05H" transferred to
                  ; tempreg2 and TBLH
                  ; in this example the data "1AH" is transferred to
                  ; tempreg1 and data "0FH" to register tempreg2
                  ; the value "00H" will be transferred to the high byte
                  ; register TBLH
:
:
org 1F00h          ; sets initial address of program memory
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
:

```

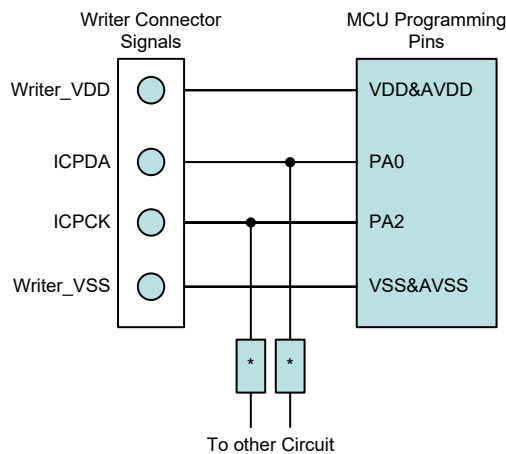
在线烧录 – ICP

Flash 型程序存储器提供用户便利地对同一芯片进行程序的更新和修改。另外，Holtek 单片机提供 4 线接口的在线烧录方式。用户可将进行过烧录或未经过烧录的单片机芯片连同电路板一起制成，最后阶段进行程序的更新和程序的烧录，在无需去除或重新插入芯片的情况下方便地保持程序为最新版。

| Holtek 烧录器引脚名称 | MCU 在线烧录引脚名称 | 引脚说明 |
|----------------|--------------|-------------|
| ICPDA | PA0 | 串行数据 / 地址烧录 |
| ICPCK | PA2 | 时钟烧录 |
| VDD | VDD&AVDD | 电源 |
| VSS | VSS&AVSS | 地 |

程序存储器可以通过 4 线的接口在线进行烧录。其中一个引脚用于数据串行下载或上传、另一个引脚用于串行时钟、两条用于提供电源。芯片在线烧写的详细使用说明超出此文档的描述范围，将由专门的参考文献提供。

烧录过程中，用户必须确保 ICPDA 和 ICPCK 这两个引脚没有连接至其它输出脚。



注：* 可能为电阻或电容。若为电阻则其值必须大于 1kΩ，若为电容则其必须小于 1nF。

片上调试 – OCDS

EV 芯片 HT66V0195 用于 HT66F0195 单片机仿真。此 EV 芯片提供片上调试功能 (OCDS) 用于开发过程中的单片机调试。除了片上调试功能方面，EV 芯片和实际 MCU 在功能上几乎是兼容的。用户可将 OCSDSA 和 OCDSCK 引脚连接至 Holtek HT-IDE 开发工具，从而实现 EV 芯片对实际 IC 的仿真。OCSDSA 引脚为 OCDS 数据 / 地址输入 / 输出脚，OCDSCK 引脚为 OCDS 时钟输入脚。当用户用 EV 芯片进行调试时，实际单片机 OCSDSA 和 OCDSCK 引脚上的其它共用功能无效。关于 OCDS 功能的详细描述，请参考“Holtek e-Link for 8-bit MCU OCDS 用户手册”文件。

| Holtek e-Link 引脚名称 | EV 芯片引脚名称 | 引脚说明 |
|--------------------|-----------|----------------------|
| OCSDSA | OCSDSA | 片上调试串行数据 / 地址输入 / 输出 |
| OCDSCK | OCDSCK | 片上调试时钟输入 |
| VDD | VDD&AVDD | 电源 |
| VSS | VSS&AVSS | 地 |

数据存储器的

数据存储器是内容可更改的 8 位 RAM 内部存储器，用来储存临时数据。

结构

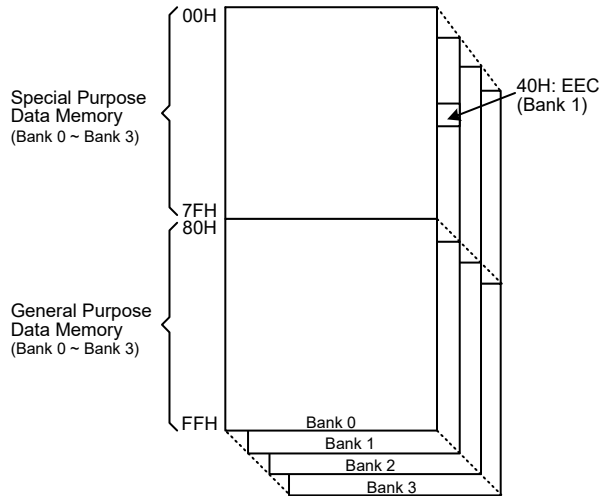
数据存储器分为两个部分，第一部分是特殊功能数据存储器。这些寄存器有固定的地址且与单片机的正确操作密切相关。大多特殊功能寄存器都可在程序控制下直接读取和写入，但有些被加以保护而不对用户开放。第二部分数据存储器是做一般用途使用，都可在程序控制下进行读取和写入。

总的存储器被分为四个 Bank，应用于 8 位宽的存储器。大部分特殊功能数据寄存器均可在所有 Bank 被访问，除了 EEC 寄存器只位于 Bank 1 的“40H”地址。切换不同区域可通过设置存储区指针 (BP) 实现。

单片机的数据寄存器的起始地址是“00H”。特殊功能数据存储器地址范围为 00H~7FH，而通用数据存储器地址范围为 80H~FFH。

| 特殊功能数据存储器 | 通用数据存储器 | |
|-----------|---------|--|
| Bank 位置 | 容量 | Bank: 地址 |
| 0~3 | 512×8 | 0: 80H~FFH 1: 80H~FFH 2: 80H~FFH 3: 80H~FFH |

数据存储器概要



数据存储器结构

通用数据存储器

所有的单片机程序需要一个读/写的存储区，让临时数据可以被储存和再使用，该 RAM 区域就是通用数据存储器。这个数据存储区可让使用者进行读取和写入的操作。使用位操作指令可对个别的位做置位或复位的操作，较大地方便了用户在数据存储器内进行位操作。

特殊功能数据存储器

这个区域的数据存储器是存放特殊寄存器的，这些寄存器与单片机的正确操作密切相关，大多数的寄存器可进行读取和写入，只有一些是被写保护而只能读取的，相关细节的介绍请参看有关特殊功能寄存器的部分。要注意的是，任何读取指令对存储器中未定义的地址进行读取将返回“00H”。

| Bank 0~3 | | Bank 0,2,3; Bank 1 | |
|----------|--------|--------------------|------------|
| 00H | IAR0 | 40H | EEC |
| 01H | MP0 | 41H | PC |
| 02H | IAR1 | 42H | PCC |
| 03H | MP1 | 43H | PCPU |
| 04H | BP | 44H | ACERL |
| 05H | ACC | 45H | SIMC0 |
| 06H | PCL | 46H | SIMC1 |
| 07H | TBLP | 47H | SIMD |
| 08H | TBLH | 48H | SIMC2/SIMA |
| 09H | TBHP | 49H | SIMTOC |
| 0AH | STATUS | 4AH | SLCDC0 |
| 0BH | SMOD | 4BH | SLCDC1 |
| 0CH | LVDC | 4CH | SLCDC2 |
| 0DH | INTEG | 4DH | SLCDC3 |
| 0EH | INTC0 | 4EH | SLCDC4 |
| 0FH | INTC1 | 4FH | SLEDC0 |
| 10H | INTC2 | 50H | SLEDC1 |
| 11H | MF10 | 51H | IFS |
| 12H | MF11 | 52H | PD |
| 13H | MF12 | 53H | PDC |
| 14H | PA | 54H | PDPU |
| 15H | PAC | 55H | USR |
| 16H | PAPU | 56H | UCR1 |
| 17H | PAWU | 57H | UCR2 |
| 18H | ACERH | 58H | BRG |
| 19H | TMPC | 59H | TXR_RXR |
| 1AH | WDTC | 5AH | |
| 1BH | TBC | | |
| 1CH | CTRL | | |
| 1DH | LVRC | | |
| 1EH | EEA | | |
| 1FH | EED | | |
| 20H | SADOL | | |
| 21H | SADOH | | |
| 22H | SADC0 | | |
| 23H | SADC1 | | |
| 24H | SADC2 | | |
| 25H | PB | | |
| 26H | PBC | | |
| 27H | PBPU | | |
| 28H | CTMC0 | | |
| 29H | CTMC1 | | |
| 2AH | CTMDL | | |
| 2BH | CTMDH | | |
| 2CH | CTMAL | | |
| 2DH | CTMAH | | |
| 2EH | CTMRP | | |
| 2FH | STMC0 | | |
| 30H | STMC1 | | |
| 31H | STMDL | | |
| 32H | STMDH | | |
| 33H | STMAL | | |
| 34H | STMAH | | |
| 35H | STM RP | | |
| 36H | | | |
| 37H | PTMC0 | | |
| 38H | PTMC1 | | |
| 39H | PTMDL | | |
| 3AH | PTMDH | | |
| 3BH | PTMAL | | |
| 3CH | PTMAH | | |
| 3DH | PTMRPL | | |
| 3EH | PTMRPH | | |
| 3FH | CPC | 7FH | |

□ : Unused, read as 00H

特殊功能数据存储结构

特殊功能寄存器

大部分特殊功能寄存器的细节将在相关功能章节描述，但有几个寄存器需在此章节单独描述。

间接寻址寄存器 – IAR0, IAR1

间接寻址寄存器 IAR0 和 IAR1 的地址虽位于数据存储区寄存器区域，但不同于普通寄存器，它们没有实际的物理地址。与定义实际存储器地址的直接存储器寻址不同，间接寻址是使用间接寻址寄存器和存储器指针来执行存储器数据操作。在间接寻址寄存器 IAR0 和 IAR1 上的任何动作，将对间接寻址指针 MP0 和 MP1 所指定的存储器地址产生对应的读 / 写操作。它们总是成对出现，IAR0 和 MP0 可以访问 Bank 0，而 IAR1 和 MP1 可以访问任何 Bank。因为这些间接寻址寄存器不是实际存在的，直接读取将返回“00H”的结果，而直接写入此寄存器则不做任何操作。

存储器指针 – MP0, MP1

该单片机提供两个存储器指针，即 MP0 和 MP1。由于这些指针在数据存储区中能像普通的寄存器一样被操作，因此提供了一个寻址和数据追踪的有效方法。当对间接寻址寄存器进行任何操作时，单片机指向的实际地址是由间接寻址指针所指定的地址。MP0、IAR0 用于访问 Bank 0，而 MP1 和 IAR1 可通过 BP 寄存器访问所有的 Bank。直接寻址仅可以用在 Bank 0 中，其它所有 Bank 只能使用 MP1 和 IAR1 进行间接寻址。

以下例子说明如何清除一个具有 4 RAM 地址的区块，它们已事先定义成地址 adres1 到 adres4。

间接寻址程序举例

```
data .section 'data'
adres1 db ?
adres2 db ?
adres3 db ?
adres4 db ?
block db ?
code .section at 0 'code'
org 00h
start:
    mov a,04h                ; set size of block
    mov block,a
    mov a,offset adres1     ; Accumulator loaded with first RAM address
    mov mp0,a               ; set memory pointer with first RAM address
loop:
    clr IAR0                ; clear the data at address defined by mp0
    inc mp0                 ; increment memory pointer
    sdz block               ; check if last memory location has been cleared
    jmp loop
continue:
:
```

在上面的例子中有一点值得注意，即并没有确定 RAM 地址。

存储区指针 – BP

数据存储区被分为四个 Bank，即 Bank 0 ~ Bank 3。可以通过设置存储区指针 (Bank Pointer) 值来访问不同的数据存储区。BP 指针的 bit 1~0 用于选择数据存储区的 Bank 0 ~ Bank 3。

复位后，数据存储器会初始化到 Bank 0，但是在休眠和空闲模式下的 WDT 溢出复位，不会改变通用数据存储器的存储区号。应该注意的是特殊功能数据存储器不受存储区的影响，也就是说，不论是在哪一个存储区，都能对特殊功能寄存器进行读写操作。数据存储器的直接寻址总是访问 Bank 0，不影响存储区指针的值。要访问 Bank 0 之外的存储区，则必须要使用间接寻址方式。

● **BP 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|---|---|-------|-------|
| Name | — | — | — | — | — | — | DMBP1 | DMBP0 |
| R/W | — | — | — | — | — | — | R/W | R/W |
| POR | — | — | — | — | — | — | 0 | 0 |

Bit 7~2 未定义，读为“0”

Bit 1~0 **DMBP1~DMBP0**: 数据存储区选择位
 00: Bank 0
 01: Bank 1
 10: Bank 2
 11: Bank 3

累加器 – ACC

对任何单片机来说，累加器是相当重要的，且与 ALU 所完成的运算有密切关系，所有 ALU 得到的运算结果都会暂时存在 ACC 累加器里。若没有累加器，ALU 必须在每次进行如加法、减法和移位的运算时，将结果写入到数据存储器，这样会造成程序编写和时间的负担。另外数据传送也常常牵涉到累加器的临时储存功能，例如在使用者定义的一个寄存器和另一个寄存器之间传送数据时，由于两寄存器之间不能直接传送数据，因此必须通过累加器来传送数据。

程序计数器低字节寄存器 – PCL

为了提供额外的程序控制功能，程序计数器低字节规划在数据存储器的特殊功能区域内，通过对此寄存器操作，便可直接跳转到其它程序地址。直接给 PCL 寄存器赋值将导致程序直接跳转到程序存储器的某一地址，然而由于寄存器只有 8 位长度，因此只允许在本页的程序存储器范围内进行跳转，而当执行此操作时，要注意会插入一个空指令周期。

表格寄存器 – TBLP, TBHP, TBLH

这三个特殊功能寄存器对存储在程序存储器中的表格进行操作。TBLP 和 TBHP 为表格指针，指向表格数据存储的地址。它们的值必须在任何表格读取指令执行前加以设定，由于它们的值可以被如“INC”或“DEC”的指令所改变，这就提供了一种简单的方法对表格数据进行读取。表格读取数据指令执行之后，表格数据高字节存储在 TBLH 中。其中要注意的是，表格数据低字节会被传送到使用者指定的地址。

状态寄存器 – STATUS

这 8 位的状态寄存器由零标志位 (Z)、进位标志位 (C)、辅助进位标志位 (AC)、溢出标志位 (OV)、暂停标志位 (PDF) 和看门狗定时器溢出标志位 (TO) 组成。这些算术 / 逻辑操作和系统运行标志位是用来记录单片机的运行状态。

除了 PDF 和 TO 标志外，状态寄存器中的位像其它大部分寄存器一样可以被改变。任何数据写入到状态寄存器将不会改变 TO 或 PDF 标志位。另外，执行不同的指令后，与状态寄存器有关的运算可能会得到不同的结果。TO 标志位只会

受系统上电、看门狗溢出或执行“CLR WDT”或“HALT”指令影响。PDF 标志位只会受执行“HALT”或“CLR WDT”指令或系统上电影响。

Z、OV、AC 和 C 标志位通常反映最近运算的状态。

- **C**: 当加法运算的结果产生进位, 或减法运算的结果没有产生借位时, 则 C 被置位, 否则 C 被清零, 同时 C 也会被带进位的移位指令所影响。
- **AC**: 当低半字节加法运算的结果产生进位, 或高半字节减法运算的结果没有产生借位时, AC 被置位, 否则 AC 被清零。
- **Z**: 当算术或逻辑运算结果是零时, Z 被置位, 否则 Z 被清零。
- **OV**: 当运算结果高两位的进位状态异或结果为 1 时, OV 被置位, 否则 OV 被清零。
- **PDF**: 系统上电或执行“CLR WDT”指令会清零 PDF, 而执行“HALT”指令则会置位 PDF。
- **TO**: 系统上电或执行“CLR WDT”或“HALT”指令会清零 TO, 而当 WDT 溢出则会置位 TO。

另外, 当进入一个中断程序或执行子程序调用时, 状态寄存器不会自动压入到堆栈保存。假如状态寄存器的内容是重要的且子程序可能改变状态寄存器的话, 则需谨慎的去做正确的储存。

● **STATUS 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|----|-----|-----|-----|-----|-----|
| Name | — | — | TO | PDF | OV | Z | AC | C |
| R/W | — | — | R | R | R/W | R/W | R/W | R/W |
| POR | — | — | 0 | 0 | x | x | x | x |

“x”：未知

- Bit 7~6 未定义, 读为“0”
- Bit 5 **TO**: 看门狗溢出标志位
0: 系统上电或执行“CLR WDT”或“HALT”指令后
1: 看门狗溢出发生
- Bit 4 **PDF**: 暂停标志位
0: 系统上电或执行“CLR WDT”指令后
1: 执行“HALT”指令
- Bit 3 **OV**: 溢出标志位
0: 无溢出
1: 运算结果高两位的进位状态异或结果为 1
- Bit 2 **Z**: 零标志位
0: 算术或逻辑运算结果不为 0
1: 算术或逻辑运算结果为 0
- Bit 1 **AC**: 辅助进位标志位
0: 无辅助进位
1: 在加法运算中低四位产生了向高四位进位, 或减法运算中低四位不发生从高四位借位
- Bit 0 **C**: 进位标志位
0: 无进位
1: 如果在加法运算中结果产生了进位, 或在减法运算中结果不发生借位
C 标志位也受循环移位指令的影响。

EEPROM 数据存储

该单片机的一个特性是内建 EEPROM 数据存储。由于其非易失的存储结构，即使在电源掉电的情况下存储器内的数据仍然保存完好。这种存储区扩展了存储器空间，对设计者来说增加了许多新的应用机会。EEPROM 可以用来存储产品编号、校准值、用户特定数据、系统配置参数或其它产品信息等。EEPROM 的数据读取和写入过程也会变的更简单。

EEPROM 数据存储结构

该单片机的 EEPROM 数据存储容量为 128×8 位。由于映射方式与程序存储器和数据存储器不同，因此不能像其它类型的存储器一样寻址。使用 Bank 0 中的一个地址寄存器和一个数据寄存器以及 Bank 1 中的一个控制寄存器，可以实现对 EEPROM 的单字节读写操作。

EEPROM 寄存器

有三个寄存器控制内部 EEPROM 数据存储总的操作。地址寄存器 EEA、数据寄存器 EED 及控制寄存器 EEC。EEA 和 EED 位于 Bank 0 中，它们能像其它特殊功能寄存器一样直接被访问。EEC 位于 Bank 1 中，不能被直接访问，仅能通过 MP1 和 IAR1 进行间接读取或写入。由于 EEC 控制寄存器位于 Bank 1 中的“40H”，在 EEC 寄存器上的任何操作被执行前，MP1 必须先设为“40H”，BP 被设为“01H”。

| 寄存器名称 | 位 | | | | | | | |
|-------|----|------|------|------|------|------|------|------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EEA | — | EEA6 | EEA5 | EEA4 | EEA3 | EEA2 | EEA1 | EEA0 |
| EED | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| EEC | — | — | — | — | WREN | WR | RDEN | RD |

EEPROM 寄存器列表

• EEA 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|------|------|------|------|------|------|------|
| Name | — | EEA6 | EEA5 | EEA4 | EEA3 | EEA2 | EEA1 | EEA0 |
| R/W | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | — | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7 未定义，读为“0”

Bit 6~0 **EEA6~ EEA0**: 数据 EEPROM 地址 bit 6~bit 0

• EED 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| Name | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **D7~D0**: EEPROM 数据 bit 7~bit 0

● EEC 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|------|-----|------|-----|
| Name | — | — | — | — | WREN | WR | RDEN | RD |
| R/W | — | — | — | — | R/W | R/W | R/W | R/W |
| POR | — | — | — | — | 0 | 0 | 0 | 0 |

Bit 7~4 未定义，读为“0”

Bit 3 **WREN**: 数据 EEPROM 写使能位

- 0: 除能
- 1: 使能

此位为数据 EEPROM 写使能位，向数据 EEPROM 写操作之前需将此位置高。将此位清零时，则禁止向数据 EEPROM 写操作。

Bit 2 **WR**: EEPROM 写控制位

- 0: 写周期结束
- 1: 写周期有效

此位为数据 EEPROM 写控制位，由应用程序将此位置高将激活写周期。写周期结束后，硬件自动将此位清零。当 WREN 未先置高时，此位置高无效。

Bit 1 **RDEN**: 数据 EEPROM 读使能位

- 0: 除能
- 1: 使能

此位为数据 EEPROM 读使能位，向数据 EEPROM 读操作之前需将此位置高。将此位清零时，则禁止向数据 EEPROM 读操作。

Bit 0 **RD**: EEPROM 读控制位

- 0: 读周期结束
- 1: 读周期有效

此位为数据 EEPROM 读控制位，由应用程序将此位置高将激活读周期。读周期结束后，硬件自动将此位清零。当 RDEN 未先置高时，此位置高无效。

注：1. 在同一条指令中 WREN、WR、RDEN 和 RD 不能同时置为“1”。WR 和 RD 不能同时置为“1”。

2. 确保 f_{SUB} 时钟在执行写动作前已稳定。

3. 确保写动作完成后才可改写 EEPROM 相关寄存器。

从 EEPROM 中读取数据

从 EEPROM 中读取数据，EEPROM 中读取数据的地址要先放入 EEA 寄存器中。EEC 寄存器中的读使能位 RDEN 置为高以使能读功能。若 EEC 寄存器中的 RD 位被置高，一个读周期将开始。若 RD 位已置为高而 RDEN 位还未被设置则不能开始读操作。若读周期结束，RD 位将自动清除为“0”，数据可以从 EED 寄存器中读取。数据在其它读或写操作执行前将一直保留在 EED 寄存器中。应用程序将轮询 RD 位以确定数据可以有效地被读取。

写数据到 EEPROM

写数据至 EEPROM，EEPROM 中写入数据的地址要先放入 EEA 寄存器中，然后将要写入的数据放入 EED 寄存器中。EEC 寄存器中的写使能位 WREN 先置为高以使能写功能，然后 EEC 寄存器中的 WR 位需立即置为高以开始写操作。这两条指令必须在两个指令周期内连续执行。在写操作执行之前需将总中断控制位 EMI 清零，在写周期开始后再置高。若 WR 位已置为高而 WREN 位还未被置高则不能开始写操作。由于控制 EEPROM 写周期是一个内部时钟，与单片机的系统时钟异步，所以数据写入 EEPROM 的时间将有所延迟。可通过轮询 EEC 寄存器中的 WR 位或判断 EEPROM 中断以侦测写周期是否完成。若写周期完成，WR 位将自动清除为“0”，通知用户数据已写入 EEPROM。因此，应用程序将轮询 WR 位以确定写周期是否结束。

写保护

防止误写入的写保护有以下几种。单片机上电后控制寄存器中的写使能位将被清除以杜绝任何写入操作。上电后 BP 将重置为“0”，这意味着数据存储区 Bank 0 被选中。由于 EEPROM 控制寄存器位于 Bank 1 中，这增加了对写操作的保护措施。在正常程序操作中确保控制寄存器中的写使能位被清除将能防止不正确的写操作。

EEPROM 中断

EEPROM 写周期结束后将产生 EEPROM 写中断，需先通过设置相关中断寄存器的 DEE 位使能 EEPROM 中断。由于 EEPROM 中断包含在多功能中断中，相应的多功能中断使能位也需被设置。当 EEPROM 写周期结束，DEF 请求标志位及其相关多功能中断请求标志位将被置位。若总中断、EEPROM 中断和多功能中断使能且堆栈未满的情况下将跳转到相应的多功能中断向量中执行。当中断被响应，只有多功能中断标志位将自动复位，而 EEPROM 中断标志将通过应用程序手动复位。

编程注意事项

必须注意的是数据不会无意写入 EEPROM。在没有写动作时写使能位被正常清零可以增强保护功能。BP 指针也可以正常清零以阻止进入 EEPROM 控制寄存器存在的 Bank 1。尽管没有必要，写一个简单的读回程序以检查新写入的数据是否正确还是应该考虑的。

当 WREN 位被置为高以后，写数据位 WR 必须立刻置为高，以确保写循环正确执行。总中断位 EMI 在写循环开始前应当被清零，写循环开始后将其使能。注意，单片机不应在 EEPROM 读或写操作完全完成之前进入空闲或休眠模式，否则 EEPROM 读或写操作将失败。

程序举例

从 EEPROM 中读取数据 – 轮询法

```
MOV A, EEPROM_ADRES      ; user defined address
MOV EEA, A
MOV A, 40H                ; set memory pointer MP1
MOV MP1, A                ; MP1 points to EEC register
MOV A, 01H                ; set bank pointer BP
MOV BP, A
SET IAR1.1                ; set RDEN bit, enable read operations
SET IAR1.0                ; start Read Cycle - set RD bit
BACK:
SZ IAR1.0                 ; check for read cycle end
JMP BACK
CLR IAR1                   ; disable EEPROM read if no more read operations
                           ; are required

CLR BP
MOV A, EED                 ; move read data to register
MOV READ_DATA, A
```

注：对于每一个读操作，即使地址是连续的，都必须重新设置地址寄存器，接着再将 RD 位置高开启一个读周期。

写数据到 EEPROM – 轮询法

```

MOV A, EEPROM_ADRES      ; user defined address
MOV EEA, A
MOV A, EEPROM_DATA      ; user defined data
MOV EED, A
MOV A, 40H               ; set memory pointer MP1
MOV MP1, A               ; MP1 points to EEC register
MOV A, 01H               ; set bank pointer BP
MOV BP, A
CLR EMI
SET IAR1.3                ; set WREN bit, enable write operations
SET IAR1.2                ; start Write Cycle - set WR bit
                          ; - executed immediately after set WREN bit

SET EMI
BACK:
SZ IAR1.2                ; check for write cycle end
JMP BACK
CLR BP
    
```

振荡器

不同的振荡器选择可以让使用者在不同的应用需求中实现更大范围的功能。振荡器的灵活性使得在速度和功耗方面可以达到较佳的优化。振荡器选择和操作是通过配置选项和相关控制寄存器共同完成的。

振荡器概述

振荡器除了作为系统时钟源，还作为看门狗定时器和时基中断的时钟源。外部振荡器需要一些外围器件，而集成的内部振荡器不需要任何外围器件。它们提供的高速和低速系统振荡器具有较宽的频率范围。所有振荡器选择通过配置选项选择。较高频率的振荡器提供更高的性能，但要求有更高的功率，反之亦然。动态切换快慢系统时钟的能力使单片机具有灵活而优化的性能 / 功耗比，此特性对功耗敏感的应用领域尤为重要。

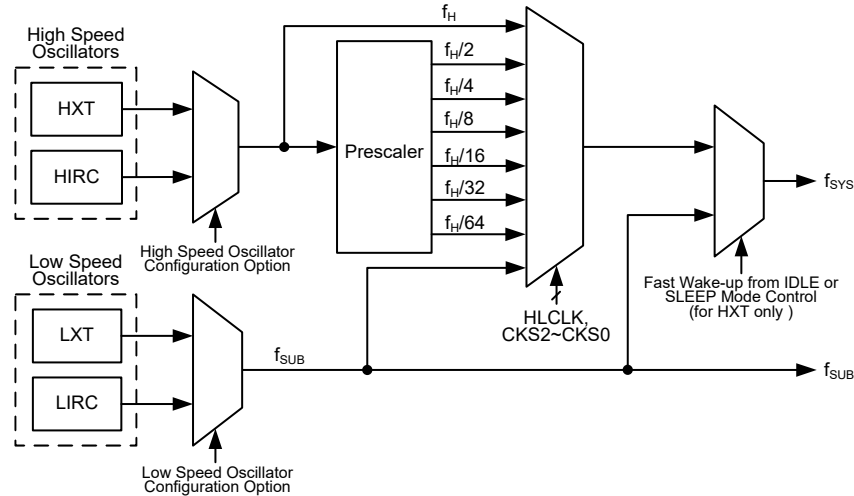
| 类型 | 名称 | 频率 | 引脚 |
|---------|------|--------------|-----------|
| 外部高速晶振 | HXT | 400kHz~16MHz | OSC1/OSC2 |
| 内部高速 RC | HIRC | 8/12/16MHz | — |
| 外部低速晶振 | LXT | 32.768kHz | XT1/XT2 |
| 内部低速 RC | LIRC | 32kHz | — |

振荡器类型

系统时钟配置

该单片机有四个系统振荡器，包括两个高速振荡器和两个低速振荡器。高速振荡器有外部晶体 / 陶瓷振荡器 HXT 和内部 8/12/16MHz RC 振荡器 HIRC，低速振荡器有内部 32kHz RC 振荡器 LIRC 和外部 32.768kHz 晶振 LXT。使用高速或低速振荡器作为系统时钟的选择是通过设置 SMOD 寄存器中的 HLCLK 位及 CKS2~CKS0 位决定的，系统时钟可动态选择。

高速或低速振荡器的实际时钟源经由配置选项选择。低速或高速系统时钟频率由 SMOD 寄存器的 HLCLK 位及 CKS2~CKS0 位决定的。请注意，两个振荡器必须做出选择，即一个高速和一个低速振荡器。

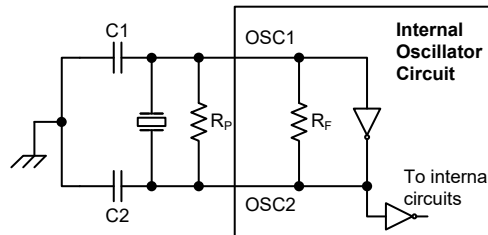


系统时钟配置

外部晶体 / 陶瓷振荡器 – HXT

外部晶体 / 陶瓷系统振荡器是一个高频振荡器，经由配置选项选择。对于晶体振荡器，只要简单地将晶体连接至 OSC1 和 OSC2，则会产生振荡所需的相移及反馈，而不需其它外部电容。为保证某些低频率的晶体振荡和陶瓷谐振器的振荡频率更精准，建议连接两个小容量电容 C1 和 C2 到 VSS，具体数值与客户选择的晶体 / 陶瓷晶振有关。

为了确保振荡器的稳定性及减少噪声和串扰的影响，晶体振荡器及其相关的电阻和电容以及它们之间的连线都应尽可能的接近单片机。



- Note: 1. R_P is normally not required. C1 and C2 are required.
2. Although not shown OSC1/OSC2 pins have a parasitic capacitance of around 7pF.

晶体 / 陶瓷振荡器 – HXT

| 晶体振荡器 C1 和 C2 值 | | |
|-----------------|-------|-------|
| 晶体频率 | C1 | C2 |
| 16MHz | 0pF | 0pF |
| 12MHz | 0pF | 0pF |
| 8MHz | 0pF | 0pF |
| 6MHz | 0pF | 0pF |
| 4MHz | 0pF | 0pF |
| 1MHz | 100pF | 100pF |

注：C1 和 C2 数值仅作参考用。

晶体振荡器电容推荐值

内部高速 RC 振荡器 – HIRC

内部 RC 振荡器是一个集成的系统振荡器，不需其它外部器件。内部 RC 振荡器具有三种固定的频率：8MHz、12MHz、16MHz，可通过配置选项选择。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡频率因电源电压、温度以及芯片制成工艺不同的影响较大程度地降低。如果选择了该内部时钟，无需额外的引脚。

外部 32.768kHz 晶体振荡器 – LXT

外部 32.768kHz 晶体系统振荡器是一个低频振荡器，经由配置选项选择。时钟频率固定为 32.768kHz，此时 XT1 和 XT2 间引脚必须连接 32.768kHz 的晶体振荡器。需要外部电阻和电容连接到 32.768kHz 晶振以帮助起振。对于那些要求精确频率的场合中，可能需要这些元件来对制程产生的误差提供频率补偿。在系统上电期间，LXT 振荡器启动需要一定的延时。

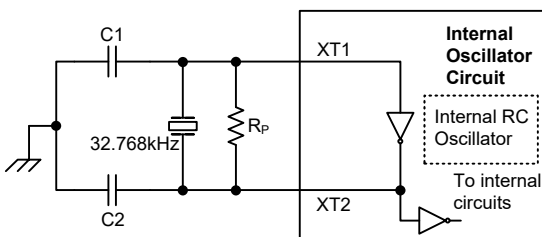
当系统进入空闲 / 休眠模式，系统时钟关闭以降低功耗。然而在某些应用，比如空闲 / 休眠模式下要保持内部定时器功能，必须提供额外的时钟，且与系统时钟无关。

然而，对于一些晶体，为了保证系统频率的启动与精度要求，需要外接两个小容量电容 C1 和 C2，具体数值与客户选择的晶体规格有关。外部并联的反馈电阻 R_P，是必需的。

引脚共用功能软件控制位决定是否 XT1/XT2 脚是用于 LXT 振荡器还是作为普通 I/O 口使用。

- 若 LXT 振荡器未被用于任何时钟源，XT1/XT2 脚能被用作一般 I/O 口或其它共用功能使用。
- 若 LXT 振荡器被用于一些时钟源，32.768kHz 晶体应被连接至 XT1/XT2 脚。

为了确保振荡器的稳定性及减少噪声和串扰的影响，晶体振荡器及其相关的电阻和电容以及它们之间的连线都应尽可能的接近单片机。



Note: 1. R_P, C1 and C2 are required.
2. Although not shown XT1/XT2 pins have a parasitic capacitance of around 7pF.

外部 LXT 振荡器

| LXT 振荡器 C1 和 C2 值 | | |
|-------------------|------|------|
| 晶振频率 | C1 | C2 |
| 32.768kHz | 10pF | 10pF |

注：1. C1 和 C2 数值仅作参考用
2. R_P 的建议值为 5MΩ~10MΩ

32.768kHz 振荡器电容推荐值

LXT 振荡器低功耗功能

LXT 振荡器可以工作在快速启动模式或低功耗模式，可通过设置 TBC 寄存器中的 LXTLP 位进行模式选择。

| LXTLP | LXT 工作模式 |
|-------|----------|
| 0 | 快速启动 |
| 1 | 低功耗 |

系统上电时会清零 LXTLP 位来快速启动 LXT 振荡器。在快速启动模式，LXT 振荡器将起振并快速稳定下来。LXT 振荡器完全起振后，可以通过设置 LXTLP 位为高进入低功耗模式。振荡器可以继续运行，其间耗电将少于快速启动模式。在功耗敏感的应用领域如电池应用方面，功耗必须尽可能小。为了降低功耗，建议系统上电 2 秒后，在应用程序中将 LXTLP 位设为“1”。应注意的是，无论 LXTLP 位是什么值，LXT 振荡器会一直运作，不同的只是在低功耗模式时启动时间更长。

内部 32kHz 振荡器 – LIRC

内部 32kHz 系统振荡器是一个低频振荡器，经由配置选项选择。它是一个完全集成 RC 振荡器，它在 5V 电压下运行的典型频率值为 32kHz 且无需外部元件。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡器因电源电压、温度及芯片制成工艺不同的影响较大程度地降低。

辅助振荡器

低速振荡器除了提供一个系统时钟源外，也用来为看门狗定时器和时基中断提供时钟来源。

工作模式和系统时钟

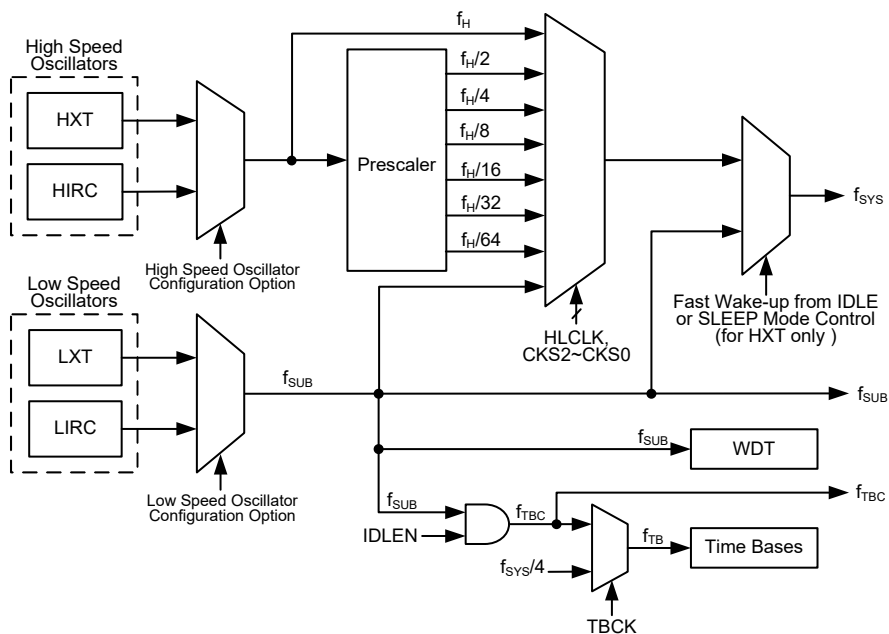
现今的应用要求单片机具有较高的性能及尽可能低的功耗，这种矛盾的要求在便携式电池供电的应用领域尤为明显。高性能所需要的高速时钟将增加功耗，反之亦然。该单片机提供高、低速两种时钟源，它们之间可以动态切换，用户可通过优化单片机操作来获得较佳性能 / 功耗比。

系统时钟

单片机为 CPU 和外围功能操作提供了多种不同的时钟源。用户使用配置选项和寄存器编程可获取多种时钟，进而使系统时钟获取较大的应用性能。

主系统时钟可来自高频时钟源 f_H 或低频时钟源 f_{SUB} ，通过 SMOD 寄存器中的 HLCLK 位及 CKS2~CKS0 位进行选择。高频时钟来自 HXT 或 HIRC 振荡器，经由配置选项选择。低频系统时钟源来自内部时钟 f_{SUB} ，低频时钟来自 LXT 或 LIRC 振荡器，经由配置选项选择。其它系统时钟还有高速系统振荡器的分频 $f_H/2 \sim f_H/64$ 。

另外两个内部时钟用于外围电路，次时钟源 f_{SUB} 和时基时钟 f_{TBC} 。这两个时钟来自 LXT 或 LIRC 振荡器，通过配置选项选择。快速唤醒发生后， f_{SUB} 为单片机提供一个次时钟。



系统时钟选项

注：当系统时钟源 f_{SYS} 由 f_H 到 f_{SUB} 转换时，高速振荡器将停止以节省耗电。因此，没有为外围电路提供 $f_H \sim f_H/64$ 的频率。

系统工作模式

单片机有 6 种不同的工作模式，每种有它自身的特性，根据应用中不同的性能和功耗要求可选择不同的工作模式。单片机正常工作有两种模式：正常模式和低速模式。剩余的 4 种工作模式：休眠模式 0、休眠模式 1、空闲模式 0 和空闲模式 1 用于单片机 CPU 关闭时以节省耗电。

| 工作模式 | 说明 | | | |
|--------|-----|-------------------|-----------|-----------|
| | CPU | f_{SYS} | f_{SUB} | f_{TBC} |
| 正常模式 | On | $f_H \sim f_H/64$ | On | On |
| 低速模式 | On | f_{SUB} | On | On |
| 空闲模式 0 | Off | Off | On | On |
| 空闲模式 1 | Off | On | On | On |
| 休眠模式 0 | Off | Off | Off | Off |
| 休眠模式 1 | Off | Off | On | Off |

正常模式

顾名思义，这是主要的工作模式之一，单片机的所有功能均可在此模式中实现且系统时钟由一个高速振荡器提供。该模式下单片机正常工作的时钟源来自 HXT 或 HIRC 振荡器。高速振荡器频率可被分为 1~64 的不等比率，实际的比率由 SMOD 寄存器中的 CKS2~CKS0 位及 HLCLK 位选择的。单片机使用高速振荡器分频作为系统时钟可减少工作电流。

低速模式

此模式的系统时钟虽为较低速时钟源，但单片机仍能正常工作。该低速时钟源可来自 LXT 或 LIRC 振荡器。单片机在此模式中运行所耗工作电流较低。在低速模式下， f_H 关闭。

休眠模式 0

在 HALT 指令执行后且 SMOD 寄存器中 IDLEN 位为低时，系统进入休眠模式 0。在休眠模式 0 中，CPU 及 f_{SUB} 停止运行，看门狗定时器功能除能。在该模式中 LV DEN 位需置为“0”，否则将不能进入休眠模式 0 中。

休眠模式 1

在 HALT 指令执行后且 SMOD 寄存器中 IDLEN 位为低时，系统进入休眠模式 1。在休眠模式 1 中，CPU 停止运行。然而若看门狗定时器功能使能， f_{SUB} 继续运行。

空闲模式 0

执行 HALT 指令后且 SMOD 寄存器中 IDLEN 位为高，CTRL 寄存器中 FSYSON 位为低时，系统进入空闲模式 0。在空闲模式 0 中，CPU 停止，但一些外围功能如看门狗定时器、TM 将继续工作。在空闲模式 0 中，系统振荡器停止。

空闲模式 1

执行 HALT 指令后且 SMOD 寄存器中 IDLEN 位为高，CTRL 寄存器中 FSYSON 位为高时，系统进入空闲模式 1。在空闲模式 1 中，CPU 停止，但会提供一个时钟源给一些外围功能如看门狗定时器、TM。在空闲模式 1 中，系统振荡器继续运行，该系统振荡器可以为高速或低速系统振荡器。

控制寄存器

寄存器 SMOD 和 CTRL 寄存器中 FSYSON 位用于控制单片机内部时钟。

• SMOD 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|------|------|-------|-----|-----|-------|-------|
| Name | CKS2 | CKS1 | CKS0 | FSTEN | LTO | HTO | IDLEN | HLCLK |
| R/W | R/W | R/W | R/W | R/W | R | R | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

Bit 7~5 **CKS2~CKS0**: 当 HLCLK 为“0”时系统时钟选择位

000: f_{SUB}
 001: f_{SUB}
 010: $f_H/64$
 011: $f_H/32$
 100: $f_H/16$
 101: $f_H/8$
 110: $f_H/4$
 111: $f_H/2$

这三位用于选择系统时钟源。除了 LXT 或 LIRC 振荡器提供的系统时钟源外，也可使用高频振荡器的分频作为系统时钟。

Bit 4 **FSTEN**: 快速唤醒控制位 (仅用于 HXT)

0: 除能
 1: 使能

此位为快速唤醒控制位，用于决定单片机被唤醒后 f_{SUB} 是否开始工作。当此位为高且 f_{SUB} 时钟可用，该时钟源用作临时系统时钟以提供快速唤醒时间。

- Bit 3 LTO:** 低速振荡器就绪标志位
0: 未就绪
1: 就绪
此位为低速系统振荡器就绪标志位，用于表明低速系统振荡器在系统上电复位或经唤醒后何时稳定下来。当系统处于 SLEEP0 模式时，该标志为低。若系统时钟来自 LXT 振荡器，系统唤醒后该位转换为高需 128 个时钟周期；若系统时钟来自 LIRC 振荡器，该位转换为高需 1~2 个时钟周期。
- Bit 2 HTO:** 高速振荡器就绪标志位
0: 未就绪
1: 就绪
此位为高速系统振荡器就绪标志位，用于表明高速系统振荡器何时稳定下来。此标志在系统上电后经硬件清零，高速系统振荡器稳定后变为高电平。因此，此位在单片机上电后由应用程序读取的总为“1”。该标志由休眠模式或空闲模式 0 中唤醒后会处于低电平状态，若使用 HXT 振荡器，该位将在 128 个时钟周期后变为高电平状态，若使用 HIRC 振荡器则只需 15~16 个时钟周期即可。
- Bit 1 IDLEN:** 空闲模式控制位
0: 除能
1: 使能
此位为空闲模式控制位，用于决定 HALT 指令执行后发生的动作。若此位为高，当指令 HALT 执行后，单片机进入空闲模式。若 FSYSON 位为高，在空闲模式 1 中 CPU 停止运行，系统时钟将继续工作以保持外围功能继续工作；若 FSYSON 为低，在空闲模式 0 中 CPU 和系统时钟都将停止运行。若此位为低，单片机将在 HALT 指令执行后进入休眠模式。
- Bit 0 HLCLK:** 系统时钟选择位
0: $f_H/2 \sim f_H/64$ 或 f_{SUB}
1: f_H
此位用于选择 f_H 或 $f_H/2 \sim f_H/64$ 还是 f_{SUB} 作为系统时钟。该位为高时选择 f_H 作为系统时钟，为低时则选择 $f_H/2 \sim f_H/64$ 或 f_{SUB} 作为系统时钟。当系统时钟由 f_H 时钟向 f_{SUB} 时钟转换时， f_H 将自动关闭以降低功耗。

● **CTRL 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|--------|---|---|---|---|------|-----|-----|
| Name | FSYSON | — | — | — | — | LVRF | LRF | WRF |
| R/W | R/W | — | — | — | — | R/W | R/W | R/W |
| POR | 0 | — | — | — | — | x | 0 | 0 |

“x”：未知

- Bit 7 FSYSON:** IDLE 模式时， f_{SYS} 控制位
0: 除能
1: 使能
该位用于控制系统时钟在空闲模式中是否开启，如果该位设为“0”，空闲模式中系统时钟关闭，如果该位设为“1”，空闲模式中系统时钟开启。
- Bit 6~3** 未定义，读为“0”
- Bit 2 LVRF:** LVR 复位标志位
详见低电压复位章节。
- Bit 1 LRF:** LVR 控制寄存器软件复位标志位
详见低电压复位章节。
- Bit 0 WRF:** WDT 控制寄存器软件复位标志位
详见看门狗定时器控制寄存器章节。

快速唤醒

单片机进入休眠模式或空闲模式 0 后，系统时钟将停止以降低功耗。然而单片机再次唤醒，原来的系统时钟重新起振、稳定且恢复正常工作需要一定的时间。为确保单片机能够尽快的开始工作，系统提供了一个快速唤醒功能。需提供一临时时钟源 f_{SUB} 先驱动系统直至原系统振荡器稳定，这个临时时钟可来自 LXT 或 LIRC 振荡器。快速启动功能的时钟源为 f_{SUB} ，该功能仅在休眠模式 1 和空闲模式 0 中有效。当单片机由休眠模式 0 唤醒时，因 f_{SUB} 已停止，故快速唤醒功能无效。快速唤醒功能使能/除能由 SMOD 寄存器中 FSTEN 位控制的。

若 HXT 振荡器作为正常模式的系统时钟，且快速唤醒功能使能，系统唤醒将需 1~2 个 t_{SUB} 时钟周期。系统开始在 f_{SUB} 时钟源下运行直至 512 个 HXT 时钟周期后 HTO 标志转换为高，系统将切换到 HXT 振荡器运行。

若系统振荡器选用 HIRC，将系统从休眠模式或空闲模式 0 中唤醒需 15~16 个时钟周期；若选用 LIRC，则需 1~2 个周期。快速唤醒位 FSTEN 在这些情况下不受影响。

| 系统振荡器 | FSTEN 位 | 唤醒时间 (休眠模式 0) | 唤醒时间 (休眠模式 1) | 唤醒时间 (空闲模式 0) | 唤醒时间 (空闲模式 1) |
|-------|---------|------------------|--|------------------|------------------|
| HXT | 0 | 128 个 HXT 周期 | 128 个 HXT 周期 | | 1~2 个 HXT 周期 |
| | 1 | 128 个 HXT 周期 | 1~2 个 f_{SUB} 周期 (系统在 f_{SUB} 下运行 512 个 HXT 周期后切换到 HXT 振荡器运行) | | 1~2 个 HXT 周期 |
| HIRC | × | 15~16 个 HIRC 周期 | 15~16 个 HIRC 周期 | | 1~2 个 HIRC 周期 |
| LIRC | × | 1~2 个 LIRC 周期 | 1~2 个 LIRC 周期 | | 1~2 个 LIRC 周期 |
| LXT | × | 128 个 LXT 周期 | 1~2 个 LXT 周期 | | 1~2 个 LXT 周期 |

“×”：无关

唤醒时间

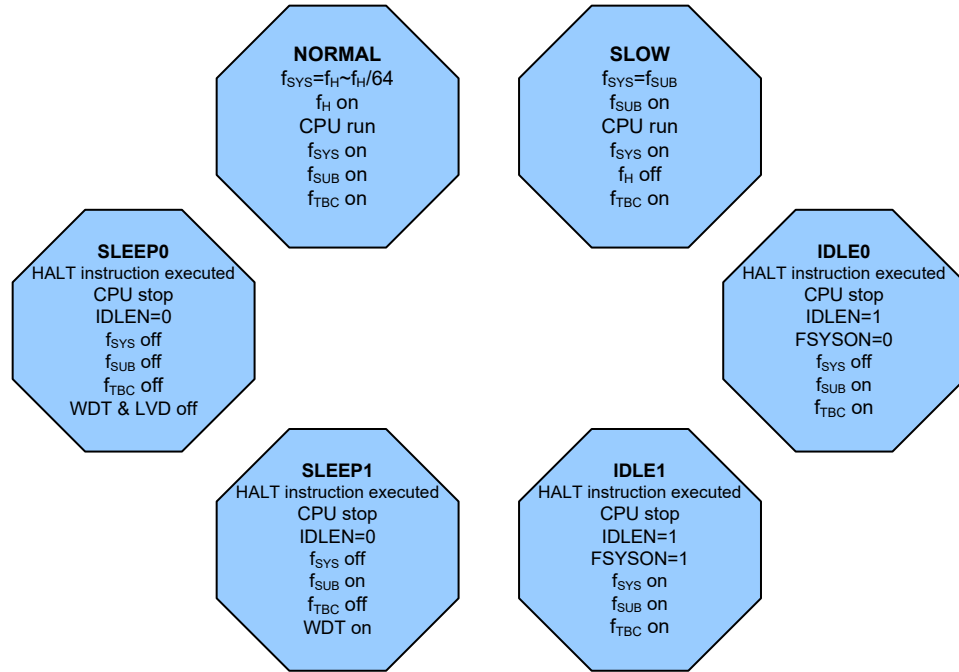
注：若看门狗定时器除能，意味着 LXT 和 LIRC 都关闭，当单片机由休眠模式 0 中唤醒时快速唤醒功能不可用。

工作模式切换

单片机可在各个工作模式间自由切换，使得用户可根据所需选择较佳的性能/功耗比。用此方式，对单片机工作的性能要求不高的情况下，可使用较低频时钟以减少工作电流，在便携式应用上延长电池的使用寿命。

简单来说，正常模式和低速模式间的切换仅需设置 SMOD 寄存器中的 HLCLK 位及 CKS2~CKS0 位即可实现，而正常模式/低速模式与休眠模式/空闲模式间的切换经由 HALT 指令实现。当 HALT 指令执行后，单片机是否进入空闲模式或休眠模式由 SMOD 寄存器中的 IDLEN 位和 CTRL 寄存器中的 FSYSON 位决定的。

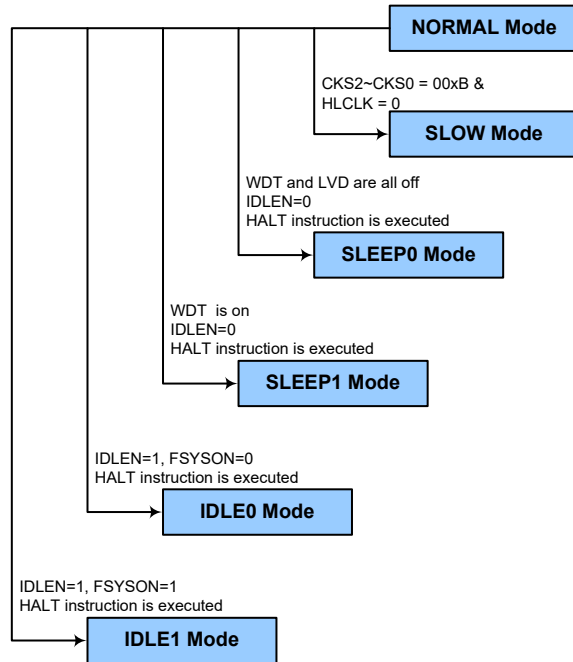
当 HLCLK 位变为低电平时，时钟源将由高速时钟源 f_H 转换成时钟源 $f_H/2 \sim f_H/64$ 或 f_{SUB} 。若时钟源来自 f_{SUB} ，高速时钟源将停止运行以节省耗电。此时须注意， $f_H/16$ 和 $f_H/64$ 内部时钟源也将停止运行，由此会影响到如 TMs 等内部功能的工作。所附流程图显示了单片机在不同工作模式间切换时的变化。



正常模式切换到低速模式

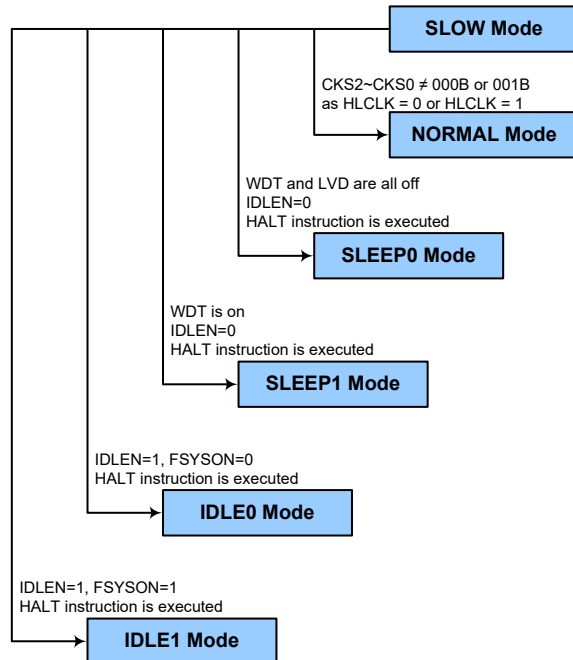
系统运行在正常模式时使用高速系统振荡器，因此较为耗电。可通过设置 SMOD 寄存器中的 HLCLK 位为“0”及 CKS2~CKS0 位为“000”或“001”使系统时钟切换至运行在低速模式下。此时将使用低速系统振荡器以节省耗电。用户可在对性能要求不高的操作中使用此方法以减少耗电。

低速模式的时钟源来自 LXT 或 LIRC 振荡器，且由配置选项决定。因此要求这个振荡器在所有模式切换动作发生前稳定下来。该动作由 SMOD 寄存器中 LTO 位控制。



低速模式切换到正常模式

在低速模式系统使用 LXT 或 LIRC 低速振荡器。切换到使用高速系统时钟振荡器的正常模式需设置 HLCLK 位为“1”，也可设置 HLCLK 位为“0”但 $CKS2\sim CKS0$ 需设为“010”、“011”、“100”、“101”、“110”或“111”。高频时钟需要一定的稳定时间，取决于使用哪个高速系统振荡器。



进入休眠模式 0

进入休眠模式 0 的方法仅有一种即应用程序中执行“HALT”指令前需设置寄存器 SMOD 中 IDLEN 位为“0”且 WDT 和 LVD 除能。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟、WDT 时钟和时基时钟停止运行，应用程序停止在“HALT”指令处。
- 数据存储器和寄存器的内容将保持当前值。
- 若 WDT 时钟源来自 LXT 或 LIRC 振荡器，WDT 将被清零并停止运行。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

进入休眠模式 1

进入休眠模式 1 的方法仅有一种即应用程序中执行“HALT”指令前需设置寄存器 SMOD 中 IDLEN 位为“0”且 WDT 使能。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟和时基时钟停止运行，应用程序停止在“HALT”指令处。WDT 继续运行，其时钟源来自 f_{SUB} 。
- 数据存储器和寄存器的内容将保持当前值。
- 若 WDT 使能，WDT 将被清零并重新开始计数。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

进入空闲模式 0

进入空闲模式 0 的方法仅有一种即应用程序中执行“HALT”指令前需设置寄存器 SMOD 中 IDLEN 位为“1”且 CTRL 寄存器中的 FSYSO 位为“0”。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟停止运行，应用程序停止在“HALT”指令处，时钟 f_{TBC} 和 f_{SUB} 将继续运行。
- 数据存储器和寄存器的内容将保持当前值。
- 若 WDT 使能，WDT 将被清零并重新开始计数。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

进入空闲模式 1

进入空闲模式 1 的方法仅有一种即应用程序中执行“HALT”指令前需设置寄存器 SMOD 中 IDLEN 位为“1”且 CTRL 寄存器中的 FSYSO 位为“1”。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟、时钟 f_{TBC} 和 f_{SUB} 开启，应用程序停止在“HALT”指令处。
- 数据存储器和寄存器的内容将保持当前值。
- 若 WDT 使能，WDT 将被清零并重新开始计数。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

待机电流的注意事项

由于单片机进入休眠或空闲模式的主要原因是将MCU的电流降低到尽可能低，可能到只有几个微安的级别（空闲模式1除外），所以如果要将电路的电流进一步降低，电路设计者还应有其它的考虑。应该特别注意的是单片机的输入/输出引脚。所有高阻抗输入脚都必须连接到固定的高或低电平，因为引脚浮空会造成内部振荡并导致耗电增加。这些引脚也必须设为输出或带有上拉电阻的输入。

另外还需注意单片机设为输出的I/O引脚上的负载。应将它们设置在有最小拉电流的状态或将它们和其它的CMOS输入一样接到没有拉电流的外部电路上。还应注意的是，如果选择LIRC或LXT振荡器，会导致耗电增加。

在空闲模式1中，系统时钟开启。若系统时钟来自高速系统振荡器，额外的待机电流也可能会有几百微安。

唤醒

单片机进入休眠模式或空闲模式后，系统时钟将停止以降低功耗。然而单片机再次唤醒，原来的系统时钟重新起振、稳定且恢复正常工作需要一定的时间。

系统进入休眠或空闲模式之后，可以通过以下几种方式唤醒：

- PA口下降沿
- 系统中断
- WDT溢出

单片机执行HALT指令，系统将进入空闲或休眠模式，PDF将被置1；系统上电或执行清除看门狗的指令，PDF将被清零。若由WDT溢出唤醒，则会发生看门狗定时器复位。看门狗计数器溢出将会置位TO标志并唤醒系统，这种复位会重置程序计数器和堆栈指针，其它标志保持原有状态。

PA口中的每个引脚都可以通过PAWU寄存器使能下降沿唤醒功能。PA端口唤醒后，程序将在“HALT”指令后继续执行。如果系统是通过中断唤醒，则有两种可能发生。第一种情况是：相关中断除能或是中断使能且堆栈已满，则程序会在“HALT”指令之后继续执行。这种情况下，唤醒系统的中断会等到相关中断使能或有堆栈层可以使用之后才执行。第二种情况是：相关中断使能且堆栈未滿，则中断可以马上执行。如果在进入休眠或空闲模式之前中断标志位已经被设置为“1”，则相关中断的唤醒功能将无效。

编程注意事项

高速振荡器和低速振荡器使用相同的SST计数器。例如，若系统从休眠模式0中唤醒，HIRC和LXT振荡器都需从关闭状态快速启动。HIRC振荡器结束其SST周期后，LXT振荡器才开始使用SST计数器。

- 若单片机从休眠模式0唤醒后进入正常模式，高速系统振荡器需要一个SST周期。在HTO为“1”后，单片机开始执行首条指令。此时，若 f_{SUB} 时钟来源于LXT振荡器，LXT振荡器可能不是稳定的，上电状态可能会发生类似情况，首条指令执行时LXT振荡器还未就绪。
- 若单片机从休眠模式1唤醒后进入正常模式，系统时钟源来自HXT振荡器且FSTEN为“1”，唤醒后，系统时钟可切换至LIRC振荡器。
- 一些外围功能，如WDT和TMs，采用系统时钟 f_{SYS} 时，在系统时钟源由 f_H 切换至 f_{SUB} 时，以上这些功能的时钟源也要随之改变。
- 当WDT时钟源选择为 f_{SUB} 时， f_{SUB} 和 f_S 的开启或关闭由WDT是否使能决定的。

看门狗定时器

看门狗定时器的功能在于防止如电磁的干扰等外部不可控制事件，所造成的程序不正常动作或跳转到未知的地址。

看门狗定时器时钟源

WDT 定时器时钟源来自于内部时钟 f_{SUB} ，由 LIRC 或 LXT 振荡器提供。LXT 振荡器由外部 32.768kHz 晶振提供。电压为 5V 时内部振荡器 LIRC 的周期大约为 32kHz。需要注意的是，这个特殊的内部时钟周期随 V_{DD} 、温度和制成的不同而变化。看门狗定时器的时钟源可分频为 $2^8 \sim 2^{18}$ 以提供更大的溢出周期，分频比由 WDTC 寄存器中的 WS2~WS0 位来决定。

看门狗定时器控制寄存器

WDTC 寄存器用于控制 WDT 功能的使能 / 除能及选择溢出周期。这个寄存器与看门狗定时器的所有操作相关。

• WDTC 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| Name | WE4 | WE3 | WE2 | WE1 | WE0 | WS2 | WS1 | WS0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |

Bit 7~3 **WE4~WE0**: WDT 功能控制

10101: 除能
01010: 使能
其它值: MCU 复位

若因外部环境噪声使这些位发生改变，单片机将复位。复位动作发生在 2~3 个 LIRC 时钟周期后，且 CTRL 寄存器的 WRF 位将置为 1。

Bit 2~0 **WS2~WS0**: WDT 溢出周期选择位

000: $2^8/f_{SUB}$
001: $2^{10}/f_{SUB}$
010: $2^{12}/f_{SUB}$
011: $2^{14}/f_{SUB}$
100: $2^{15}/f_{SUB}$
101: $2^{16}/f_{SUB}$
110: $2^{17}/f_{SUB}$
111: $2^{18}/f_{SUB}$

这三位控制 WDT 时钟源的分频比，从而实现对 WDT 溢出周期的控制。

• CTRL 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|--------|---|---|---|---|------|-----|-----|
| Name | FSYSON | — | — | — | — | LVRF | LRF | WRF |
| R/W | R/W | — | — | — | — | R/W | R/W | R/W |
| POR | 0 | — | — | — | — | x | 0 | 0 |

“x”：未知

Bit 7 **FSYSON**: IDLE 模式时， f_{SYS} 控制位
详见操作模式和系统时钟章节。

Bit 6~3 未定义，读为“0”

Bit 2 **LVRF**: LVR 复位标志位
详见低电压复位章节。

- Bit 1 **LRF**: LVR 控制寄存器软件复位标志位
 详见低电压复位章节。
- Bit 0 **WRF**: WDT 控制寄存器软件复位标志位
 0: 未发生
 1: 发生
 WDT 控制寄存器复位时, 该位被置为 1, 且通过应用程序清除。注意, 该位只能由应用程序清零。

看门狗定时器操作

当 WDT 溢出时, 看门狗定时器产生一个芯片复位的动作。这也就意味着正常工作期间, 用户需在应用程序中看门狗溢出前有策略地清看门狗定时器以防止其产生复位, 可使用清除看门狗指令实现。无论什么原因, 程序失常跳转到一个未知的地址或进入一个死循环, 此清除指令都不能被正确执行, 此种情况下, 看门狗将溢出以使单片机复位。WDTC 寄存器中的 WE4~WE0 位可提供使能/除能控制以及看门狗定时器复位操作。如果 WE4~WE0 设置为“10101B”, 则 WDT 除能; 如果 WE4~WE0 设置为“01010B”, 则 WDT 使能。如果 WE4~WE0 设置为除“10101B”和“01010B”以外的其它任意值, 则经过 2~3 个 f_{LIRC} 时钟周期后单片机复位。上电后这些位初始化为“01010B”。

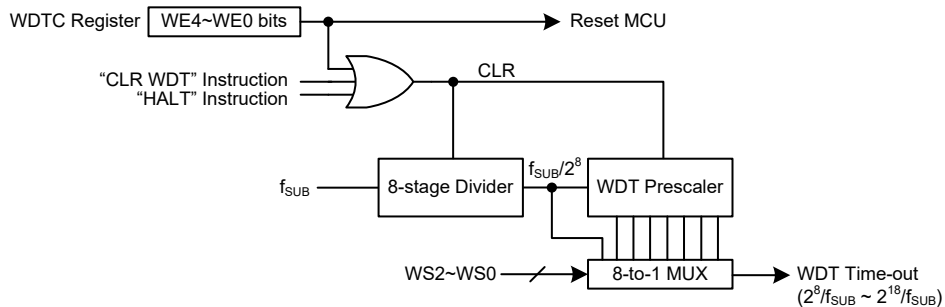
| WE4~WE0 位 | WDT 功能 |
|-----------|--------|
| 10101B | 除能 |
| 01010B | 使能 |
| 其它值 | MCU 复位 |

看门狗定时器使能 / 除能控制

程序正常运行时, WDT 溢出将导致芯片复位, 并置位状态标志位 TO。若系统处于休眠或空闲模式, 当 WDT 发生溢出时, 状态寄存器中的 TO 应置位, 仅 PC 和堆栈指针复位。有三种方法可以用来清除 WDT 的内容。第一种是 WDTC 软件复位, 即将 WE4~WE0 位设置成除了 01010B 和 10101B 外的任意值; 第二种是通过软件清除指令, 而第三种是通过“HALT”指令。

该单片机只使用一条清看门狗指令“CLR WDT”。因此只要执行“CLR WDT”便清除 WDT。

当设置分频比为 2^{18} 时, 溢出周期最大。例如, 时钟源为 32kHz LIRC 振荡器, 分频比为 2^{18} 时最大溢出周期约 8s, 分频比为 2^8 时最小溢出周期约 8ms。



看门狗定时器

复位和初始化

复位功能是在任何单片机中基本的部分，使得单片机可以设定一些与外部参数无关的先置条件。最重要的复位条件是在单片机首次上电以后，经过短暂的延迟，内部硬件电路使得单片机处于预期的稳定状态并开始执行第一条程序指令。上电复位以后，在程序执行之前，部分重要的内部寄存器将会被设定为预先设定的状态。程序计数器就是其中之一，它会被清除为零，使得单片机从最低的程序存储器地址开始执行程序。

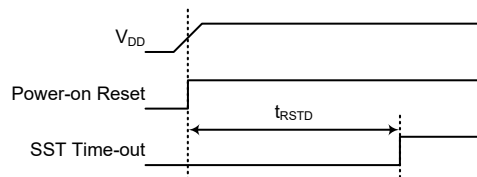
除上电复位以外，另一种复位为低电压复位即 LVR 复位，在电源供应电压低于 LVR 设定值时，系统会产生 LVR 复位。另一种复位为看门狗溢出单片机复位。不同方式的复位操作会对寄存器产生不同的影响。

复位功能

单片机有多种复位方式，描述如下：

上电复位

这是最基本且不可避免的复位，发生在单片机上电后。除了保证程序存储器从开始地址执行，上电复位也使得其它寄存器被设定在预设条件。所有的输入/输出端口控制寄存器在上电复位时会保持高电平，以确保上电后所有引脚被设定为输入状态。

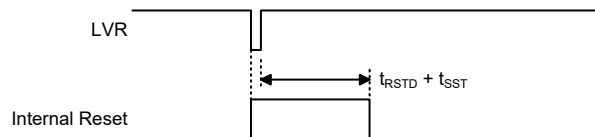


注： t_{RSTD} 为上电延迟时间，典型值为 50ms

上电复位时序图

低电压复位 – LVR

单片机具有低电压复位电路，用来监测它的电源电压。低电压复位功能总是使能于特定的电压值， V_{LVR} 。例如在更换电池的情况下，单片机供应的电压可能会落在 $0.9V \sim V_{LVR}$ 之间，这时 LVR 将会自动复位单片机，并且寄存器 CTRL 中的 LVRF 位将被自动置位为 1。LVR 包含以下的规格：有效的 LVR 信号，即在 $0.9V \sim V_{LVR}$ 的低电压状态的时间，必须超过 LVD/LVR 电气特性中的 t_{LVR} 参数的值。如果低电压存在不超过 t_{LVR} 参数的值，则 LVR 将会忽略它且不会执行复位功能。实际的 V_{LVR} 参数值可通过 LVRC 寄存器中的 LVS 位进行选择。如果由于不利的环境干扰因素使得 LVS7~LVS0 为其它值，LVR 将在 2~3 个 f_{LIRC} 时钟周期后复位单片机。此时，CTRL 寄存器中的 LRF 位被设置为 1。寄存器 LVRC 上电后初始化为 01010101B。当单片机进入休眠或空闲模式时 LVR 功能将自动除能。



注： t_{RSTD} 为上电延迟时间，典型值为 50ms

低电压复位时序图

• LVRC 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|------|------|------|------|------|------|------|
| Name | LVS7 | LVS6 | LVS5 | LVS4 | LVS3 | LVS2 | LVS1 | LVS0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

Bit 7~0 **LVS7~LVS0**: LVR 电压选择

01010101: 2.1V
00110011: 2.55V
10011001: 3.15V
10101010: 3.8V

其它值: 复位单片机 – 寄存器复位为 POR 值。

当上述定义的相应的低电压发生, 且此低电压保持时间超过 t_{LVR} 值, 则单片机复位。此复位后寄存器内容保持不变。

除上述四个已定义的寄存器值, 其它值也会导致单片机复位。复位动作将在 2~3 个 f_{LIRC} 时钟周期后执行。此复位后寄存器内容将恢复为 POR 值。

• CTRL 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|--------|---|---|---|---|------|-----|-----|
| Name | FSYSON | — | — | — | — | LVRF | LRF | WRF |
| R/W | R/W | — | — | — | — | R/W | R/W | R/W |
| POR | 0 | — | — | — | — | x | 0 | 0 |

“x”: 未知

Bit 7 **FSYSON**: IDLE 模式时, f_{SYS} 控制位
详见操作模式和系统时钟章节。

Bit 6~3 未定义, 读为 “0”

Bit 2 **LVRF**: LVR 复位标志位
0: 未发生
1: 发生

当特定的低电压复位条件发生时, 该位被置为 1。该位只能由应用程序清零。

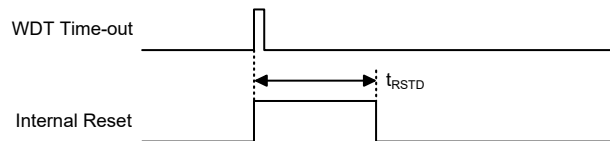
Bit 1 **LRF**: LVR 控制寄存器软件复位标志位
0: 未发生
1: 发生

如果 LVRC 控制寄存器包含任何非定义的 LVR 电压值, 该位被置为 1, 这类似于软件复位功能。该位只能由应用程序清零。

Bit 0 **WRF**: WDT 控制寄存器软件复位标志位
详见看门狗定时器控制寄存器章节。

正常运行时看门狗溢出复位

除了看门狗溢出标志位 TO 将被设为 “1” 之外, 正常运行时看门狗溢出复位和 LVR 复位相同。

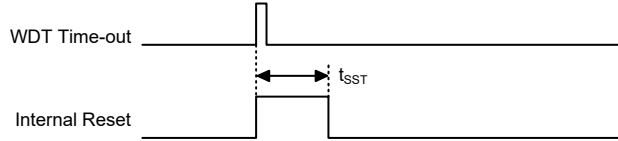


注: t_{RSTD} 为上电延迟时间, 典型值为 16.7ms

正常运行时看门狗溢出复位时序图

休眠或空闲时看门狗溢出复位

休眠或空闲时看门狗溢出复位和其它种类的复位有些不同。除了程序计数器与堆栈指针将被清“0”及 TO 位被设为“1”外，绝大部分的条件保持不变。图中 t_{SST} 的详细说明请参考交流电气特性。



休眠或空闲时看门狗溢出复位时序图

复位初始状态

不同的复位形式以不同的途径影响复位标志位。这些标志位，即 PDF 和 TO 存放在状态寄存器中，由休眠或空闲模式功能或看门狗计数器等几种控制器操作控制。复位标志位如下所示：

| TO | PDF | 复位条件 |
|----|-----|----------------------|
| 0 | 0 | 上电复位 |
| u | u | 正常模式或低速模式时的 LVR 复位 |
| 1 | u | 正常模式或低速模式时的 WDT 溢出复位 |
| 1 | 1 | 空闲或休眠模式时的 WDT 溢出复位 |

“u”代表不改变

在单片机上电复位之后，各功能单元初始化的情形，列于下表。

| 项目 | 复位后情况 |
|-----------|----------------|
| 程序计数器 | 清除为零 |
| 中断 | 所有中断被除能 |
| 看门狗定时器，时基 | 都清除，且 WDT 重新计数 |
| 定时模块 | 所有定时模块停止 |
| 输入 / 输出口 | I/O 口设为输入模式 |
| 堆栈指针 | 堆栈指针指向堆栈顶端 |

不同的复位形式对单片机内部寄存器的影响是不同的。为保证复位后程序能正常执行，了解寄存器在特定条件复位后的设置是非常重要的。下表即为不同方式复位后内部寄存器的状况。若芯片有多种封装类型，表格反映较大的封装的情况。

| 寄存器 | 上电复位 | LVR 复位 (正常操作) | WDT 溢出 (正常操作) | WDT 溢出 (空闲 / 休眠) |
|------|-----------|------------------|------------------|---------------------|
| IAR0 | xxxx xxxx | uuuu uuuu | xxxx xxxx | uuuu uuuu |
| MP0 | xxxx xxxx | xxxx xxxx | xxxx xxxx | uuuu uuuu |
| IAR1 | xxxx xxxx | uuuu uuuu | xxxx xxxx | uuuu uuuu |
| MP1 | xxxx xxxx | xxxx xxxx | xxxx xxxx | uuuu uuuu |
| BP | ---- --00 | ---- --00 | ---- --00 | ---- --uu |
| ACC | xxxx xxxx | uuuu uuuu | uuuu uuuu | uuuu uuuu |
| PCL | 0000 0000 | 0000 0000 | 0000 0000 | 0000 0000 |
| TBLP | xxxx xxxx | uuuu uuuu | uuuu uuuu | uuuu uuuu |
| TBLH | xxxx xxxx | uuuu uuuu | uuuu uuuu | uuuu uuuu |

| 寄存器 | 上电复位 | LVR 复位 (正常操作) | WDT 溢出 (正常操作) | WDT 溢出 (空闲/休眠) |
|--------|-----------|------------------|------------------|------------------------|
| TBHP | ---x xxxx | ---u uuuu | ---u uuuu | ---u uuuu |
| STATUS | --00 xxxx | --uu uuuu | --1u uuuu | --11 uuuu |
| SMOD | 0000 0011 | 0000 0011 | 0000 0011 | uuuu uuuu |
| LVDC | --00 -000 | --00 -000 | --00 -000 | --uu -uuu |
| INTEG | ---- 0000 | ---- 0000 | ---- 0000 | ---- uuuu |
| INTC0 | -000 0000 | -000 0000 | -000 0000 | -uuu uuuu |
| INTC1 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| INTC2 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| MFI0 | --00 --00 | --00 --00 | --00 --00 | --uu --uu |
| MFI1 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| MFI2 | --00 --00 | --00 --00 | --00 --00 | --uu --uu |
| PA | 1111 1111 | 1111 1111 | 1111 1111 | uuuu uuuu |
| PAC | 1111 1111 | 1111 1111 | 1111 1111 | uuuu uuuu |
| PAPU | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PAWU | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| ACERH | ---- 0000 | ---- 0000 | ---- 0000 | ---- uuuu |
| TMPC | 0--- -000 | 0--- -000 | 0--- -000 | u--- -uuu |
| WDTC | 0101 0011 | 0101 0011 | 0101 0011 | uuuu uuuu |
| TBC | 0011 0111 | 0011 0111 | 0011 0111 | uuuu uuuu |
| CTRL | 0--- -x00 | 0--- -000 | 0--- -uuu | u--- -uuu |
| LVRC | 0101 0101 | uuuu uuuu | 0101 0101 | uuuu uuuu |
| EEA | -000 0000 | -000 0000 | -000 0000 | -uuu uuuu |
| EED | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SADOL | xxxx ---- | xxxx ---- | xxxx ---- | uuuu ---- (ADRFS=0) |
| | | | | uuuu uuuu (ADRFS=1) |
| SADOH | xxxx xxxx | uuuu uuuu | uuuu uuuu | uuuu uuuu (ADRFS=0) |
| | | | | ---- uuuu (ADRFS=1) |
| SADC0 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SADC1 | 000- -000 | 000- -000 | 000- -000 | uuu- -uuu |
| SADC2 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PB | -111 1111 | -111 1111 | -111 1111 | -uuu uuuu |
| PBC | -111 1111 | -111 1111 | -111 1111 | -uuu uuuu |
| PBPU | -000 0000 | -000 0000 | -000 0000 | -uuu uuuu |
| CTMC0 | 0000 0--- | 0000 0--- | 0000 0--- | uuuu u--- |
| CTMC1 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| CTMDL | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| CTMDH | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |

| 寄存器 | 上电复位 | LVR 复位 (正常操作) | WDT 溢出 (正常操作) | WDT 溢出 (空闲 / 休眠) |
|------------|------------|------------------|------------------|---------------------|
| CTMAL | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| CTMAH | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| CTMRP | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| STMC0 | 0000 0--- | 0000 0--- | 0000 0--- | uuuu u--- |
| STMC1 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| STMDL | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| STMDH | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| STMAL | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| STMAH | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| STMRP | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PTMC0 | 0000 0--- | 0000 0--- | 0000 0--- | uuuu u--- |
| PTMC1 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PTMDL | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PTMDH | ----- --00 | ----- --00 | ----- --00 | ----- --uu |
| PTMAL | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PTMAH | ----- --00 | ----- --00 | ----- --00 | ----- --uu |
| PTMRPL | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PTMRPH | ----- --00 | ----- --00 | ----- --00 | ----- --uu |
| CPC | 1000 0001 | 1000 0001 | 1000 0001 | uuuu uuuu |
| PC | -111 1111 | -111 1111 | -111 1111 | -uuu uuuu |
| PCC | -111 1111 | -111 1111 | -111 1111 | -uuu uuuu |
| PCPU | -000 0000 | -000 0000 | -000 0000 | -uuu uuuu |
| ACERL | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SIMC0 | 111- 0000 | 111- 0000 | 111- 0000 | uuu- uuuu |
| SIMC1 | 1000 0001 | 1000 0001 | 1000 0001 | uuuu uuuu |
| SIMD | xxxx xxxx | xxxx xxxx | xxxx xxxx | uuuu uuuu |
| SIMA/SIMC2 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SIMTOC | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SLCDC0 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SLCDC1 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SLCDC2 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SLCDC3 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| SLCDC4 | ----- --00 | ----- --00 | ----- --00 | ----- --uu |
| SLEDC0 | 0101 0101 | 0101 0101 | 0101 0101 | uuuu uuuu |
| SLEDC1 | --01 0101 | --01 0101 | --01 0101 | --uu uuuu |
| IFS | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| PD | ----- 1111 | ----- 1111 | ----- 1111 | ----- uuuu |
| PDC | ----- 1111 | ----- 1111 | ----- 1111 | ----- uuuu |
| PDPU | ----- 0000 | ----- 0000 | ----- 0000 | ----- uuuu |
| USR | 0000 1011 | 0000 1011 | 0000 1011 | uuuu uuuu |
| UCR1 | 0000 00x0 | 0000 00x0 | 0000 00x0 | uuuu uuuu |

| 寄存器 | 上电复位 | LVR 复位 (正常操作) | WDT 溢出 (正常操作) | WDT 溢出 (空闲/休眠) |
|---------|-----------|------------------|------------------|-------------------|
| UCR2 | 0000 0000 | 0000 0000 | 0000 0000 | uuuu uuuu |
| BRG | xxxx xxxx | xxxx xxxx | xxxx xxxx | uuuu uuuu |
| TXR_RXR | xxxx xxxx | xxxx xxxx | xxxx xxxx | uuuu uuuu |
| EEC | ---- 0000 | ---- 0000 | ---- 0000 | ---- uuuu |

注：“u”表示不改变
“x”表示未知
“-”表示未定义

输入 / 输出端口

Holtek 单片机的输入 / 输出控制具有很大的灵活性。大部分引脚可在用户程序控制下被设定为输入或输出。所有引脚的上拉电阻设置以及指定引脚的唤醒设置也都由软件控制，这些特性也使得此单片机在广泛应用上都能符合开发的需求。

该单片机提供 PA~PD 双向输入 / 输出口。这些寄存器在数据存储器有特定的地址。所有 I/O 口用于输入输出操作。作为输入操作，输入引脚无锁存功能，也就是说输入数据必须在执行“MOV A, [m]”，T2 的上升沿准备好，m 为端口地址。对于输出操作，所有数据都是被锁存的，且保持不变直到输出锁存被重写。

| 寄存器 名称 | 位 | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|-------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PA | PA7 | PA6 | PA5 | PA4 | PA3 | PA2 | PA1 | PA0 |
| PAC | PAC7 | PAC6 | PAC5 | PAC4 | PAC3 | PAC2 | PAC1 | PAC0 |
| PAPU | PAPU7 | PAPU6 | PAPU5 | PAPU4 | PAPU3 | PAPU2 | PAPU1 | PAPU0 |
| PAWU | PAWU7 | PAWU6 | PAWU5 | PAWU4 | PAWU3 | PAWU2 | PAWU1 | PAWU0 |
| PB | — | PB6 | PB5 | PB4 | PB3 | PB2 | PB1 | PB0 |
| PBC | — | PBC6 | PBC5 | PBC4 | PBC3 | PBC2 | PBC1 | PBC0 |
| PBPU | — | PBPU6 | PBPU5 | PBPU4 | PBPU3 | PBPU2 | PBPU1 | PBPU0 |
| PC | — | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 |
| PCC | — | PCC6 | PCC5 | PCC4 | PCC3 | PCC2 | PCC1 | PCC0 |
| PCPU | — | PCPU6 | PCPU5 | PCPU4 | PCPU3 | PCPU2 | PCPU1 | PCPU0 |
| PD | — | — | — | — | PD3 | PCD2 | PD1 | PD0 |
| PDC | — | — | — | — | PDC3 | PDC2 | PDC1 | PDC0 |
| PDPU | — | — | — | — | PDPU3 | PDPU2 | PDPU1 | PDPU0 |

“—”：未定义，读为“0”

I/O 逻辑功能寄存器列表

上拉电阻

许多产品应用在端口处于输入状态时需要外加一个上拉电阻来实现上拉的功能。为了免去外部上拉电阻，当引脚规划为输入时(除 SCS 输入)，都可通过内部连接到一个上拉电阻。这些上拉电阻可通过相关上拉电阻控制寄存器 PAPU~PDPU 来设置，它用一个 PMOS 晶体管来实现上拉电阻功能。

● **PxPU 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|-------|-------|-------|-------|-------|-------|-------|
| Name | PxPU7 | PxPU6 | PxPU5 | PxPU4 | PxPU3 | PxPU2 | PxPU1 | PxPU0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

PxPUn: I/O Px 口上拉电阻控制位

- 0: 除能
- 1: 使能

PxPUn 位用于控制对应引脚的上拉电阻功能。这里的 x 可以是端口 A、B、C 或 D。但是，每个 I/O 端口实际有效位可能不同。

PA 口唤醒

当使用暂停指令“HALT”迫使单片机进入休眠或空闲模式，单片机的系统时钟将会停止以降低功耗，此功能对于电池及低功耗应用很重要。唤醒单片机有很多种方法，其中之一就是使 PA 口的其中一个引脚从高电平转为低电平。这个功能特别适合于通过外部开关来唤醒的应用。PA 口的每个引脚可以通过设置 PAWU 寄存器来单独选择是否具有唤醒功能。

● **PAWU 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|-------|-------|-------|-------|-------|-------|-------|
| Name | PAWU7 | PAWU6 | PAWU5 | PAWU4 | PAWU3 | PAWU2 | PAWU1 | PAWU0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **PAWU7~PAWU0:** PA7~PA0 唤醒功能控制位

- 0: 除能
- 1: 使能

输入 / 输出端口控制寄存器

每一个输入 / 输出口都具有各自的控制寄存器，即 PAC~PDC，用来控制输入 / 输出状态。从而每个 I/O 引脚都可以通过软件控制，动态的设置为 CMOS 输出或输入。所有的 I/O 端口的引脚都各自对应于 I/O 端口控制的某一位。若 I/O 引脚要实现输入功能，则对应的控制寄存器的位需要设置为“1”。这时程序指令可以直接读取输入脚的逻辑状态。若控制寄存器相应的位被设定为“0”，则此引脚被设置为 CMOS 输出。当引脚设置为输出状态时，程序指令读取的是输出口寄存器的内容。注意，如果对输出口做读取动作时，程序读取到的是内部输出数据锁存器中的状态，而不是输出引脚上实际的逻辑状态。

● **PxC 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|------|------|------|------|------|------|------|
| Name | PxC7 | PxC6 | PxC5 | PxC4 | PxC3 | PxC2 | PxC1 | PxC0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

PxCn: I/O Px 口输入 / 输出类型选择位

- 0: 输出
- 1: 输入

PxCn 位用于控制对应引脚的状态类型。这里的 x 可以是端口 A、B、C 或 D。但是，每个 I/O 端口实际有效位可能不同。

输入 / 输出端口源电流控制

该单片机的每个 I/O 口都支持不同的源电流驱动能力，通过相应的源电流选择位控制。每个输入 / 输出端口有 4 个级别的源电流驱动能力。用户可以参考直流电气特性部分选择所需的源电流用于不同应用。

| 寄存器名称 | 位 | | | | | | | |
|--------|-------|-------|-------|-------|-------|-------|-------|-------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SLEDC0 | PBPS3 | PBPS2 | PBPS1 | PBPS0 | PAPS3 | PAPS2 | PAPS1 | PAPS0 |
| SLEDC1 | — | — | PDPS1 | PDPS0 | PCPS3 | PCPS2 | PCPS1 | PCPS0 |

输入 / 输出端口源电流选择寄存器列表

• SLEDC0 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|-------|-------|-------|-------|-------|-------|-------|
| Name | PBPS3 | PBPS2 | PBPS1 | PBPS0 | PAPS3 | PAPS2 | PAPS1 | PAPS0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

Bit 7~6 **PBPS3~PBPS2**: PB6~PB4 引脚源电流选择位

- 00: 源电流 = Level 0 (最小)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (最大)

Bit 5~4 **PBPS1~PBPS0**: PB3~PB0 引脚源电流选择位

- 00: 源电流 = Level 0 (最小)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (最大)

Bit 3~2 **PAPS3~PAPS2**: PA7~PA4 引脚源电流选择位

- 00: 源电流 = Level 0 (最小)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (最大)

Bit 1~0 **PAPS1~PAPS0**: PA3~PA0 引脚源电流选择位

- 00: 源电流 = Level 0 (最小)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (最大)

• SLEDC1 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|-------|-------|-------|-------|-------|-------|
| Name | — | — | PDPS1 | PDPS0 | PCPS3 | PCPS2 | PCPS1 | PCPS0 |
| R/W | — | — | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | — | — | 0 | 1 | 0 | 1 | 0 | 1 |

Bit 7~6 未定义，读为“0”

Bit 5~4 **PDPS1~PDPS0**: PD3~PD0 引脚源电流选择位

- 00: 源电流 = Level 0 (最小)
- 01: 源电流 = Level 1
- 10: 源电流 = Level 2
- 11: 源电流 = Level 3 (最大)

- Bit 3~2 **PCPS3~PCPS2:** PC6~PC4 引脚源电流选择位
 00: 源电流 = Level 0 (最小)
 01: 源电流 = Level 1
 10: 源电流 = Level 2
 11: 源电流 = Level 3 (最大)
- Bit 1~0 **PCPS1~PCPS0:** PC3~PC0 引脚源电流选择位
 00: 源电流 = Level 0 (最小)
 01: 源电流 = Level 1
 10: 源电流 = Level 2
 11: 源电流 = Level 3 (最大)

引脚重置功能

引脚的多功能可以增加单片机应用的灵活性。有限的引脚个数将会限制设计者，而引脚的多功能将会解决很多此类问题。每个功能可单独选择所在的引脚，以及一个确定的优先级，使得引脚上多种功能可以同时使用。此外，一些引脚功能可以通过寄存器 **IFS** 进行设定。

封装中有限的引脚个数会对某些单片机功能造成影响。然而，引脚功能重置和引脚功能选择，使得小封装单片机具有更多不同的功能。如果共用引脚功能同时有多个输出，“/”标记的引脚名称拥有更高优先级。

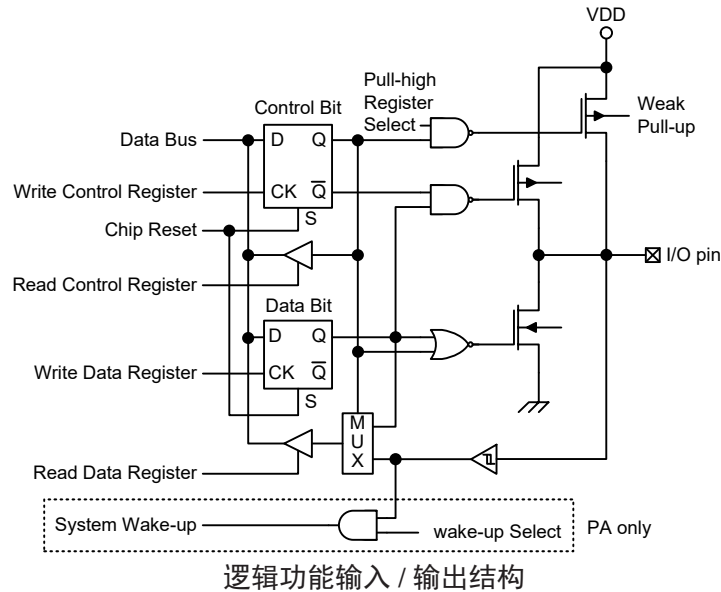
• IFS 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|--------|--------|-----------|-----------|--------|------|------|
| Name | RXCTL | SDOPS1 | SDOPS0 | SDI_SDAPS | SCK_SCLPS | SCSBPS | TXPS | RXPS |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Bit 7 **RXCTL:** RX 使能控制
 0: 除能
 1: 使能
- Bit 6~5 **SDOPS1~SDOPS0:** SDO 引脚重置选择位
 00: SDO on PC3
 01: SDO on PA1
 10: 未定义
 11: SDO on PC2
- Bit 4 **SDI_SDAPS:** SDI/SDA 引脚重置选择位
 0: SDI/SDA on PC4
 1: SDI/SDA on PA3
- Bit 3 **SCK_SCLPS:** SCK/SCL 引脚重置选择位
 0: SCK/SCL on PC5
 1: SCK/SCL on PB6
- Bit 2 **SCSBPS:** \overline{SCS} 引脚重置选择位
 0: \overline{SCS} on PC6
 1: \overline{SCS} on PB5
- Bit 1 **TXPS:** TX 引脚重置选择位
 0: TX on PD2
 1: TX on PB3
- Bit 0 **RXPS:** RX 引脚重置选择位
 0: RX on PD1
 1: RX on PB4

输入 / 输出引脚结构

下图为输入 / 输出引脚逻辑功能的内部结构图。输入 / 输出引脚的准确逻辑结构图可能与此图不同，这里只是为了方便对 I/O 引脚功能的理解提供的一个参考。由于存在诸多的引脚共用结构，在此不方便提供所有类型引脚功能结构图。



编程注意事项

在编程中，最先要考虑的是端口的初始化。复位之后，所有的输入 / 输出数据及端口控制寄存器都将被设为逻辑高。所有输入 / 输出引脚默认为输入状态，而其电平则取决于其它相连接电路以及是否选择了上拉电阻。如果端口控制寄存器某些引脚位被设定输出状态，这些输出引脚会有初始高电平输出，除非数据寄存器端口在程序中被预先设定。设置哪些引脚是输入及哪些引脚是输出，可通过设置正确的值到适当的端口控制寄存器，或使用指令“SET [m].i”及“CLR [m].i”来设定端口控制寄存器中个别的位。注意，当使用这些位控制指令时，系统即将产生一个读 - 修改 - 写的操作。单片机需要先读入整个端口上的数据，修改个别的位，然后重新把这些数据写入到输出端口。

PA 口的每个引脚都带唤醒功能。单片机处于休眠或空闲模式时，有很多方法可以唤醒单片机，其中之一就是通过 PA 任一引脚电平从高到低转换的方式，可以设置 PA 口一个或多个引脚具有唤醒功能。

定时器模块 – TM

控制和测量时间在任何单片机中都是一个很重要的部分。该单片机提供几个定时器模块 (简称 TM)，来实现和时间有关的功能。定时器模块是包括多种操作的定时单元，提供的操作有：定时 / 计数器，捕捉输入，比较匹配输出，单脉冲输出以及 PWM 输出等功能。每个定时器模块有两个独立中断。每个 TM 外加的输入输出引脚，扩大了定时器的灵活性，便于用户使用。

这里只介绍各种 TM 的共性，更多详细资料请参考简易型，标准型和周期型定时器章节。

简介

该单片机包含 3 个 TM，每个 TM 可被划分为一个特定的类型，即简易型 TM，标准型 TM 或周期型 TM。虽然性质相似，但不同 TM 特性复杂度不同。本章介绍简易型，标准型和周期型 TM 的共性，更多详细资料分别见后面各章。三种类型 TM 的特性和区别见下表。

| TM 功能 | CTM | STM | PTM |
|----------------|--------|--------|--------|
| 定时 / 计数器 | √ | √ | √ |
| 捕捉输入 | — | √ | √ |
| 比较匹配输出 | √ | √ | √ |
| PWM 通道数 | 1 | 1 | 1 |
| 单脉冲输出 | — | 1 | 1 |
| PWM 对齐方式 | 边沿对齐 | 边沿对齐 | 边沿对齐 |
| PWM 调节周期 & 占空比 | 占空比或周期 | 占空比或周期 | 占空比或周期 |

TM 功能概要

TM 操作

不同类型的 TM 提供从简单的定时操作到 PWM 信号产生等多种功能。理解 TM 操作的关键是比较 TM 内独立运行的计数器的值与内部比较器的预置值。当计数器的值与比较器的预置值相同时，则比较匹配，TM 中断信号产生，清零计数器并改变 TM 输出引脚的状态。用户选择内部时钟或外部时钟来驱动内部 TM 计数器。

TM 时钟源

驱动 TM 计数器的时钟源很多。通过设置 xTM 控制寄存器的 xTCK2~xTCK0 位，选择所需的时钟源，“x”代表 C、S 或者 P 类型的 TM。该时钟源来自系统时钟 f_{SYS} 或内部高速时钟 f_H 或 f_{SUB} 时钟源或外部 xTCK 引脚时钟的分频比。xTCK 引脚时钟源用于允许外部信号作为 TM 时钟源或用于事件计数。

TM 中断

简易型 TM，标准型 TM 或周期型 TM 都有两个内部中断，分别是内部比较器 A 或比较器 P，当比较匹配发生时产生 TM 中断。当 TM 中断产生时，计数器清零并改变 TM 输出引脚的状态。

TM 外部引脚

无论哪种类型的 TM，都有一个 TM 输入引脚 xTCK。xTCK 引脚可作为 xTM 时钟源输入脚，通过设置 xTMC0 寄存器中的 xTCK2~xTCK0 位进行选择。外部时钟源可通过该引脚来驱动内部 TM。xTCK 引脚可选择上升沿有效或下降沿

有效。STCK 和 PTCK 引脚也可用作 STM 和 PTM 单脉冲输出模式的外部触发引脚。

每个 TM 有一个输出引脚 xTP。当 TM 工作在比较匹配输出模式且比较匹配发生时，这些引脚会由 TM 控制切换到高电平或低电平或翻转。外部 xTP 输出引脚也被 TM 用来产生 PWM 输出波形。当 TM 工作在捕捉输入模式，xTP 引脚也可作为输入脚。

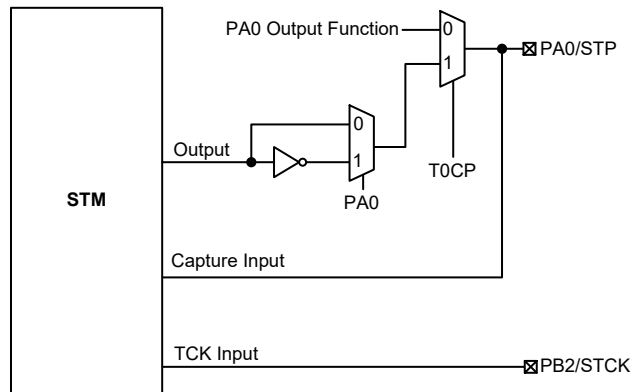
因为 TM 输出引脚与其它功能共用，TM 输出功能需要通过内部 TM 开启 / 关闭控制，工作模式选择，以及输入 / 输出控制寄存器的相应位的设置来使能或除能。当相应 TM 配置设置 xTP 用作外部输出脚，相应的引脚将作为外部 TM 输出引脚。当相应 TM 配置设置 xTP 为输入引脚，xTP 可作为输入引脚，且输入的信号来自于外部信号或其它共用功能的输出信号。当相应 TM 配置设置 xTP 不被使用，这些引脚将用作其它共用功能使用。不同类型 TM 的外部引脚详见下表。

| STM | PTM | CTM | 寄存器 |
|------------|-----------|-----------|-------------|
| STCK0; STP | PTCK; PTP | CTCK; CTP | xTMC0; TMPC |

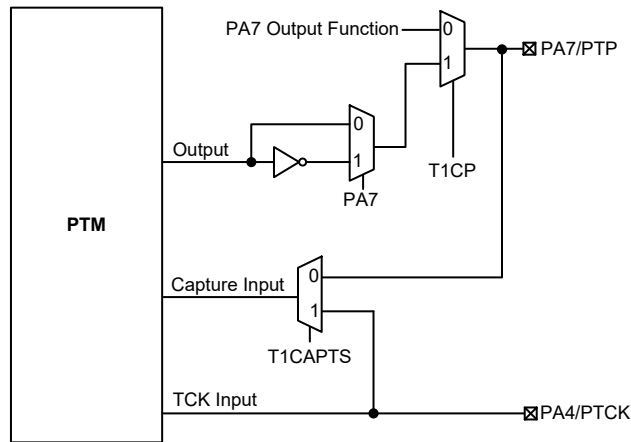
TM 外部引脚

TM 输入 / 输出引脚控制寄存器

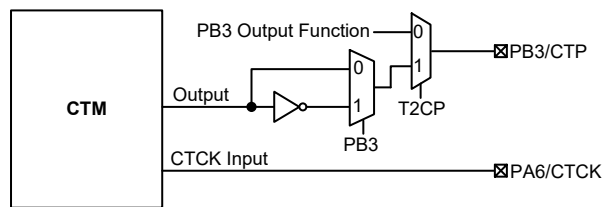
通过设置与 TM 输入 / 输出引脚相关的寄存器中的对应位，选择作为 TM 输入 / 输出功能或其它共用功能。设置位为高时，相关引脚用作 TM 输入 / 输出，如果重置为 0，则引脚将回到其原来的其它功能。



STM 功能引脚控制方框图



PTM 功能引脚控制方框图



CTM 功能引脚控制方框图

- 注：1. 上图所示输入 / 输出寄存器数据位用于 TM 输出反相控制。
2. 捕捉输入模式中，TM 引脚控制寄存器最多只能使能一个 TM 输入引脚。

● TMPC 寄存器

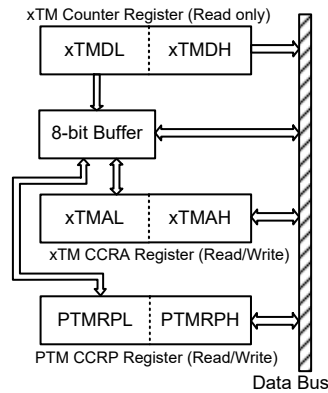
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|---|---|---|---|------|------|------|
| Name | CLOP | — | — | — | — | T2CP | T1CP | T0CP |
| R/W | R/W | — | — | — | — | R/W | R/W | R/W |
| POR | 0 | — | — | — | — | 0 | 0 | 0 |

- Bit 7 **CLOP**: CLO 引脚控制位
0: 除能
1: 使能
- Bit 6~3 未定义，读为“0”
- Bit 2 **T2CP**: CTP 引脚控制位
0: 除能
1: 使能
- Bit 1 **T1CP**: PTP 引脚控制位
0: 除能
1: 使能
- Bit 0 **T0CP**: STP 引脚控制位
0: 除能
1: 使能

编程注意事项

TM 计数寄存器和捕捉 / 比较寄存器 CCRA、CCRP，含有低字节和高字节结构。高字节可直接访问，低字节则仅能通过一个内部 8-bit 的缓存器进行访问。读写这些成对的寄存器需通过特殊的方式。值得注意的是 8-bit 缓存器的存取数据及相关低字节的读写操作仅在其相应的高字节读取操作执行时发生。

CCRA 和 CCRP 寄存器访问方式如下图所示，读写这些成对的寄存器需通过特殊的方式。建议使用“MOV”指令，通过以下步骤访问 CCRA 和 CCRP 低字节寄存器，即 xTMAL 和 PTMRPL。若不采用以下步骤访问 CCRA 和 CCRP 将导致不可预期的结果。

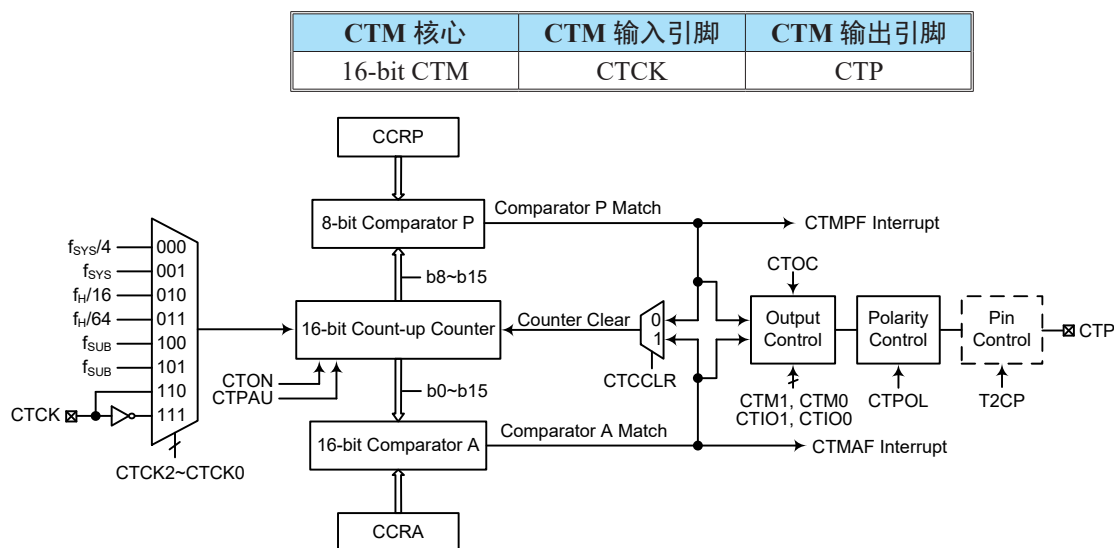


读写流程如下步骤所示：

- 写数据至 CCRA 或 CCRP
 - ◆ 步骤 1. 写数据至低字节寄存器 xTMAL 或 PTMRPL
 - 注意，此时数据仅写入 8-bit 缓存器。
 - ◆ 步骤 2. 写数据至高字节寄存器 xTMAH 或 PTMRPH
 - 注意，此时数据直接写入高字节寄存器，同时锁存在 8-bit 缓存器中的数据写入低字节寄存器。
- 由计数器寄存器和 CCRA 或 CCRP 中读取数据
 - ◆ 步骤 1. 由高字节寄存器 xTMDH, xTMAH 或 PTMRPH 读取数据
 - 注意，此时高字节寄存器中的数据直接读取，同时由低字节寄存器读取的数据锁存至 8-bit 缓存器中。
 - ◆ 步骤 2. 由低字节寄存器 xTMDL, xTMAL 或 PTMRPL 读取数据
 - 注意，此时读取 8-bit 缓存器中的数据。

简易型 TM – CTM

虽然简易型 TM 是这几种 TM 类型中最简单的形式，但仍然包括三种工作模式，即比较匹配输出、定时 / 事件计数器和 PWM 输出模式。简易型 TM 由一个外部输入脚控制并驱动一个外部输出脚。



16 位简易型 TM 方框图

简易型 TM 操作

简易型 TM 核心是一个由用户选择的内部或外部时钟源驱动的 16 位向上计数器，它还包括两个内部比较器即比较器 A 和比较器 P。这两个比较器将计数器的值与 CCRP 和 CCRA 寄存器中的值进行比较。CCRP 是 8 位的，与计数器的高 8 位比较；而 CCRA 是 16 位的，与计数器的所有位比较。

通过应用程序改变 16 位计数器值的唯一方法是使 CTON 位发生上升沿跳变清除计数器。此外，计数器溢出或比较匹配也会自动清除计数器。上述条件发生时，通常情况会产生 TM 中断信号。简易型 TM 可工作在不同的模式，可由包括来自输入脚的不同时钟源驱动，也可以控制输出脚。所有工作模式的设定都是通过设置相关寄存器来实现的。

简易型 TM 寄存器介绍

简易型 TM 的所有操作由一系列寄存器控制。一对只读寄存器用来存放 16 位计数器的值，一对读 / 写寄存器存放 16 位 CCRA 的值。CTMRP 寄存器存放 8 位 CCRP 的值。剩下两个控制寄存器设置不同的操作和控制模式。

| 寄存器名称 | 位 | | | | | | | |
|-------|-------|-------|-------|-------|-------|-------|-------|--------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CTMC0 | CTPAU | CTCK2 | CTCK1 | CTCK0 | CTON | — | — | — |
| CTMC1 | CTM1 | CTM0 | CTIO1 | CTIO0 | CTOC | CTPOL | CTDPX | CTCCLR |
| CTMDL | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| CTMDH | D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 |
| CTMAL | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| CTMAH | D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 |
| CTMRP | CTRP7 | CTRP6 | CTRP5 | CTRP4 | CTRP3 | CTRP2 | CTRP1 | CTRP0 |

16-bit 简易型 TM 寄存器列表

● CTMC0 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|-------|-------|-------|------|---|---|---|
| Name | CTPAU | CTCK2 | CTCK1 | CTCK0 | CTON | — | — | — |
| R/W | R/W | R/W | R/W | R/W | R/W | — | — | — |
| POR | 0 | 0 | 0 | 0 | 0 | — | — | — |

- Bit 7 **CTPAU**: CTM 计数器暂停控制位
 0: 运行
 1: 暂停
 通过设置此位为高可使计数器暂停，清零此位恢复正常计数器操作。当处于暂停条件时，CTM 保持上电状态并继续耗电。当此位由低到高转换时，计数器将保留其剩余值，直到此位再次改变为低电平，从此值开始继续计数。
- Bit 6~4 **CTCK2~CTCK0**: 选择 CTM 计数时钟位
 000: $f_{SYS}/4$
 001: f_{SYS}
 010: $f_H/16$
 011: $f_H/64$
 100: f_{SUB}
 101: f_{SUB}
 110: CTCK 上升沿时钟
 111: CTCK 下降沿时钟
 此三位用于选择 CTM 的时钟源。外部引脚时钟源能被选择在上升沿或下降沿有效。 f_{SYS} 是系统时钟， f_H 和 f_{SUB} 是其它的内部时钟源，细节方面请参考振荡器章节。
- Bit 3 **CTON**: CTM 计数器 On/Off 控制位
 0: Off
 1: On
 此位控制 CTM 的总开关功能。设置此位为高则使能计数器使其运行，清零此位则除能 CTM。清零此位将停止计数器并关闭 CTM 减少耗电。当此位经由低到高转换时，内部计数器将复位清零；当此位经由高到低转换时，内部计数器将保持其剩余值，直到此位再次改变为高电平。
 若 CTM 处于比较匹配输出或 PWM 输出模式时，当 CTON 位经由低到高转换时，CTM 输出脚将复位至 CTOC 位指定的初始值。
- Bit 2~0 未定义，读为“0”

• CTMC1 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|------|-------|-------|------|-------|-------|--------|
| Name | CTM1 | CTM0 | CTIO1 | CTIO0 | CTOC | CTPOL | CTDPX | CTCCLR |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~6 **CTM1~CTM0**: 选择 CTM 工作模式位

- 00: 比较匹配输出模式
- 01: 未定义模式
- 10: PWM 输出模式
- 11: 定时 / 计数器模式

这两位设置 CTM 需要的工作模式。为了确保操作可靠，CTM 应在 CTM1 和 CTM0 位有任何改变前先关掉。在定时 / 计数器模式，CTM 输出脚控制必须除能。

Bit 5~4 **CTIO1~CTIO0**: 选择 CTM 引脚输出功能位

比较匹配输出模式

- 00: 无变化
- 01: 输出低
- 10: 输出高
- 11: 输出翻转

PWM 输出模式

- 00: 强制无效状态
- 01: 强制有效状态
- 10: PWM 输出
- 11: 未定义

定时 / 计数器模式

未使用

此两位用于决定在一定条件达到时 CTM 输出脚如何改变状态。这两位值的选择取决于 CTM 运行在哪种模式下。

在比较匹配输出模式下，CTIO1 和 CTIO0 位决定当比较器 A 比较匹配输出发生时 CTM 输出脚如何改变状态。当比较器 A 比较匹配输出发生时 CTM 输出脚能设为切换高、切换低或翻转当前状态。若此两位同时为 0 时，这个输出将不会改变。CTM 输出脚的初始值通过 CTMC1 寄存器的 CTOC 位设置取得。注意，由 CTIO1 和 CTIO0 位得到的输出电平必须与通过 CTOC 位设置的初始值不同，否则当比较匹配发生时，CTM 输出脚将不会发生变化。在 CTM 输出脚改变状态后，通过 CTON 位由低到高电平的转换复位至初始值。

在 PWM 输出模式，CTIO1 和 CTIO0 用于决定比较匹配条件发生时怎样改变 CTM 输出脚的状态。PWM 输出功能通过这两位的变化进行更新。仅在 CTM 关闭时改变 CTIO1 和 CTIO0 位的值是很有必要的。若在 CTM 运行时改变 CTIO1 和 CTIO0 的值，PWM 输出的值是无法预料的。

Bit 3 **CTOC**: CTP 输出控制位

比较匹配输出模式

- 0: 初始低
- 1: 初始高

PWM 输出模式

- 0: 低有效
- 1: 高有效

这是 CTM 输出脚输出控制位。它取决于 CTM 此时正运行于比较匹配输出模式还是 PWM 输出模式。若 CTM 处于定时 / 计数器模式，则其不受影响。在比较匹配输出模式时，其决定比较匹配发生前 CTM 输出脚的逻辑电平值。在 PWM 输出模式时，其决定 PWM 信号是高有效还是低有效。

- Bit 2 **CTPOL**: CTM CTP 输出极性控制位
 0: 同相
 1: 反相
 此位控制 CTP 输出脚的极性。此位为高时 CTM 输出脚反相，为低时 CTM 输出脚同相。若 CTM 处于定时 / 计数器模式时其不受影响。
- Bit 1 **CTDPX**: CTM PWM 周期 / 占空比控制位
 0: CCRP – 周期; CCRA – 占空比
 1: CCRP – 占空比; CCRA – 周期
 此位决定 CCRA 与 CCRP 寄存器哪个被用于 PWM 波形的周期和占空比控制。
- Bit 0 **CTCCLR**: 选择 CTM 计数器清零条件位
 0: CTM 比较器 P 匹配
 1: CTM 比较器 A 匹配
 此位用于选择清除计数器的方法。简易型 TM 包括两个比较器 - 比较器 A 和比较器 P。这两个比较器每个都可以用作清除内部计数器。CTCCLR 位设为高，计数器在比较器 A 比较匹配发生时被清除；此位设为低，计数器在比较器 P 比较匹配发生或计数器溢出时被清除。计数器溢出清除的方法仅在 CCRP 被清除为 0 时才能生效。CTCCLR 位在 PWM 输出模式时未使用。

• **CTMDL 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|----|----|----|----|----|----|----|----|
| Name | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R | R | R | R | R | R | R | R |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **D7~D0**: CTM 计数器低字节寄存器 bit 7 ~ bit 0
 CTM 16-bit 计数器 bit 7 ~ bit 0

• **CTMDH 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|-----|-----|-----|-----|-----|----|----|
| Name | D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 |
| R/W | R | R | R | R | R | R | R | R |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~2 **D15~D8**: CTM 计数器高字节寄存器 bit 7 ~ bit 0
 CTM 16-bit 计数器 bit 15 ~ bit 8

• **CTMAL 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| Name | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **D7~D0**: CTM CCRA 低字节寄存器 bit 7 ~ bit 0
 CTM 16-bit CCRA bit 7 ~ bit 0

• CTMAH 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| Name | D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **D15~D8**: CTM CCRA 高字节寄存器 bit 7 ~ bit 0
CTM 16-bit CCRA bit 15 ~ bit 8

• CTMRP 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|-------|-------|-------|-------|-------|-------|-------|
| Name | CTRP7 | CTRP6 | CTRP5 | CTRP4 | CTRP3 | CTRP2 | CTRP1 | CTRP0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **CTRP7~CTRP0**: CTM CCRP 8 位寄存器，与 CTM 计数器 bit 15~bit 8 比较。
比较器 P 匹配周期
0: 65536 个 CTM 时钟
1~255: 256×(1~255) 个 CTM 时钟

此八位设定内部 CCRP 8-bit 寄存器的值，然后与内部计数器的高八位进行比较。如果 CTCCLR 位设为 0 并清除内部计数器。CTCCLR 位设为低，CCRP 比较匹配结果将重置内部计数器。由于 CCRP 只与计数器高八位比较，比较结果是 256 时钟周期的倍数。CCRP 被清零时，实际上会使得计数器在最大值溢出。

简易型 TM 工作模式

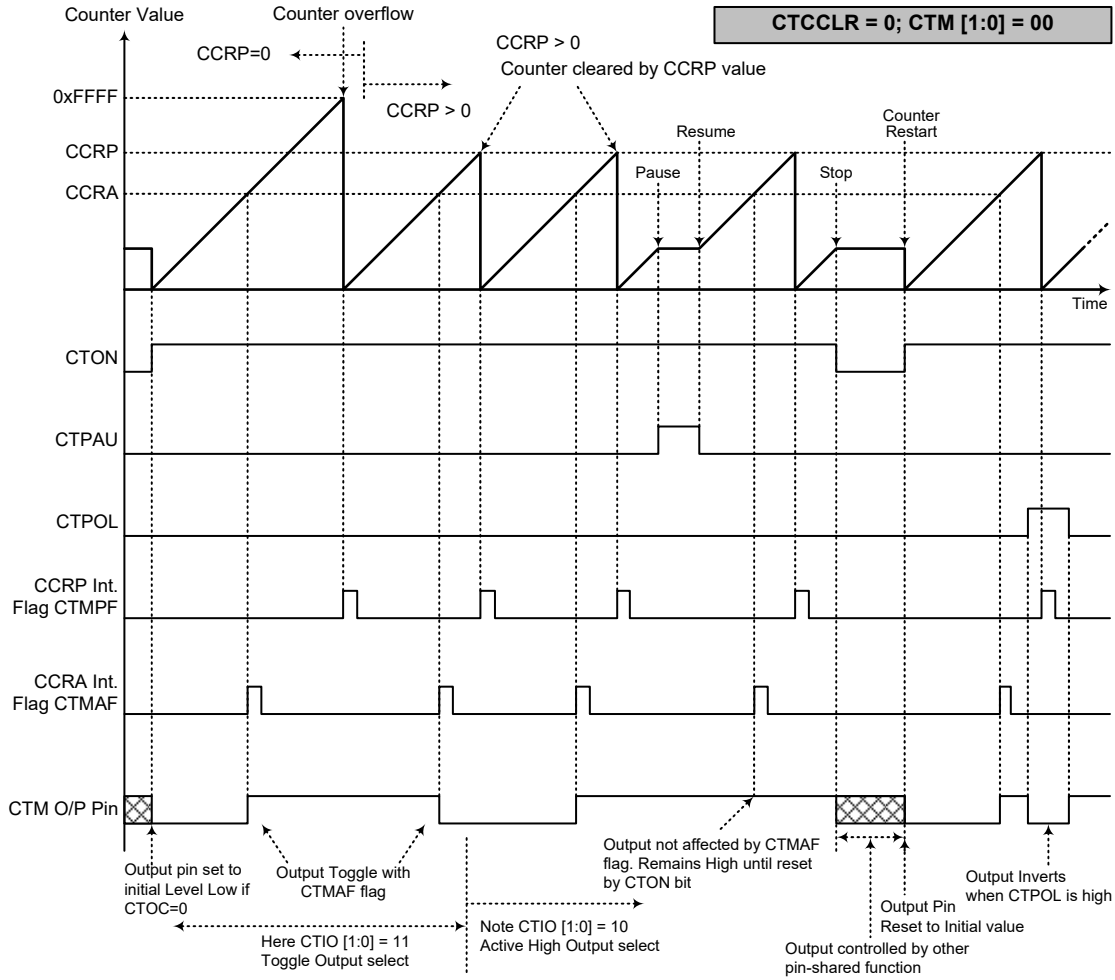
简易型 TM 有三种工作模式，即比较匹配输出模式，PWM 输出模式或定时 / 计数器模式。通过设置 CTMC1 寄存器的 CTM1 和 CTM0 位选择任意工作模式。

比较匹配输出模式

为使 CTM 工作在此模式，CTMC1 寄存器中的 CTM1 和 CTM0 位需要设置为“00”。当工作在该模式，一旦计数器使能并开始计数，有三种方法来清零，分别是：计数器溢出，比较器 A 比较匹配发生和比较器 P 比较匹配发生。CTCCLR 位为低，有两种方法清除计数器。一种是比较器 P 比较匹配发生，另一种是 CCRP 所有位设置为零并使得计数器溢出。此时，比较器 A 和比较器 P 的请求标志位 CTMAF 和 CTMPF 将分别置起。

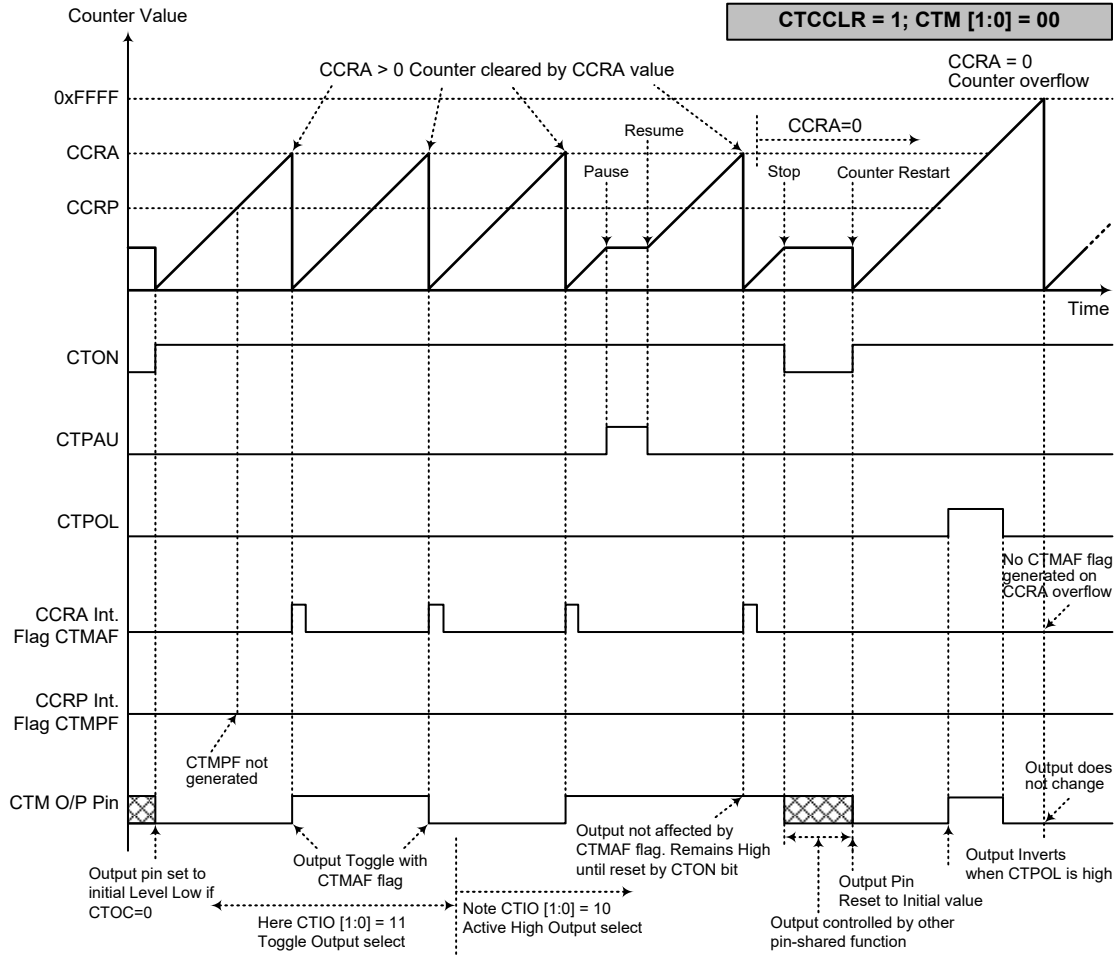
如果 CTMC1 寄存器的 CTCCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅 CTMAF 中断请求标志产生。所以当 CTCCLR 为高时，不产生 CTMPF 中断请求标志。如果 CCRA 被清零，当计数达到最大值 FFFFH 时，计数器溢出，而此时不产生 CTMAF 中断请求标志。

正如该模式名所言，当比较匹配发生后，CTM 输出脚状态改变。当比较器 A 比较匹配发生后 CTMAF 标志产生时，CTM 输出脚状态改变。比较器 P 比较匹配发生时产生的 CTMPF 标志不影响 CTM 输出脚。CTM 输出脚状态改变方式由 CTMC1 寄存器中 CTIO1 和 CTIO0 位决定。当比较器 A 比较匹配发生时，CTIO1 和 CTIO0 位决定 CTM 输出脚输出高，低或翻转当前状态。在 CTON 位由低到高电平的变化后，CTM 输出脚初始状态为 CTC位所指定的电平。注意，若 CTIO1 和 CTIO0 位同时为 0 时，引脚输出不变。



比较匹配输出模式 – CTCCLR=0

- 注：1. CTCCLR=0，比较器 P 匹配将清除计数器
2. CTM 输出脚仅由 CTMAF 标志位控制
3. 在 CTON 上升沿 CTM 输出脚复位至初始值



比较匹配输出模式 - CTCCLR=1

- 注：1. CTCCLR=1，比较器 A 匹配将清除计数器
 2. CTM 输出脚仅由 CTMAF 标志位控制
 3. 在 CTON 上升沿 CTM 输出脚复位至初始值
 4. 当 CTCCLR=1 时，CTMPF 标志位不会产生

定时 / 计数器模式

为使 CTM 工作在此模式，CTMC1 寄存器中的 CTM1 和 CTM0 位需要设置为“11”。定时 / 计数器模式与比较匹配输出模式操作方式相同，并产生同样的中断请求标志。不同的是，在定时 / 计数器模式下 CTM 输出脚未使用。因此，比较匹配输出模式中的描述和时序图可以适用于此功能。该模式中未使用的 CTM 输出脚用作普通 I/O 脚或其它功能。

PWM 输出模式

为使 CTM 工作在此模式，CTMC1 寄存器中的 CTM1 和 CTM0 位需要设置为“10”。CTM 的 PWM 功能在马达控制，加热控制，照明控制等方面十分有用。给 TMn 输出脚提供一个频率固定但占空比可调的信号，将产生一个有效值等于 DC 均方根的 AC 方波。

由于 PWM 波形的周期和占空比可调，其波形的选择就较为灵活。在 PWM 输出模式中，CTCCLR 位不影响 PWM 操作。CCRA 和 CCRP 寄存器决定 PWM 波形，一个用来清除内部计数器并控制 PWM 波形的频率，另一个用来控制占空比。哪个寄存器控制频率或占空比取决于 CTMC1 寄存器的 CTD PX 位。所以 PWM 波形频率和占空比由 CCRA 和 CCRP 寄存器共同决定。

当比较器 A 或比较器 P 比较匹配发生时，将产生 CCRA 或 CCRP 中断标志。CTMC1 寄存器中的 CTOC 位决定 PWM 波形的极性，CTIO1 和 CTIO0 位使能 PWM 输出或将 CTM 输出脚置为逻辑高或逻辑低。CTPOL 位对 PWM 输出波形的极性取反。

- 16-bit CTM, PWM 输出模式, 边沿对齐模式, CTD PX=0

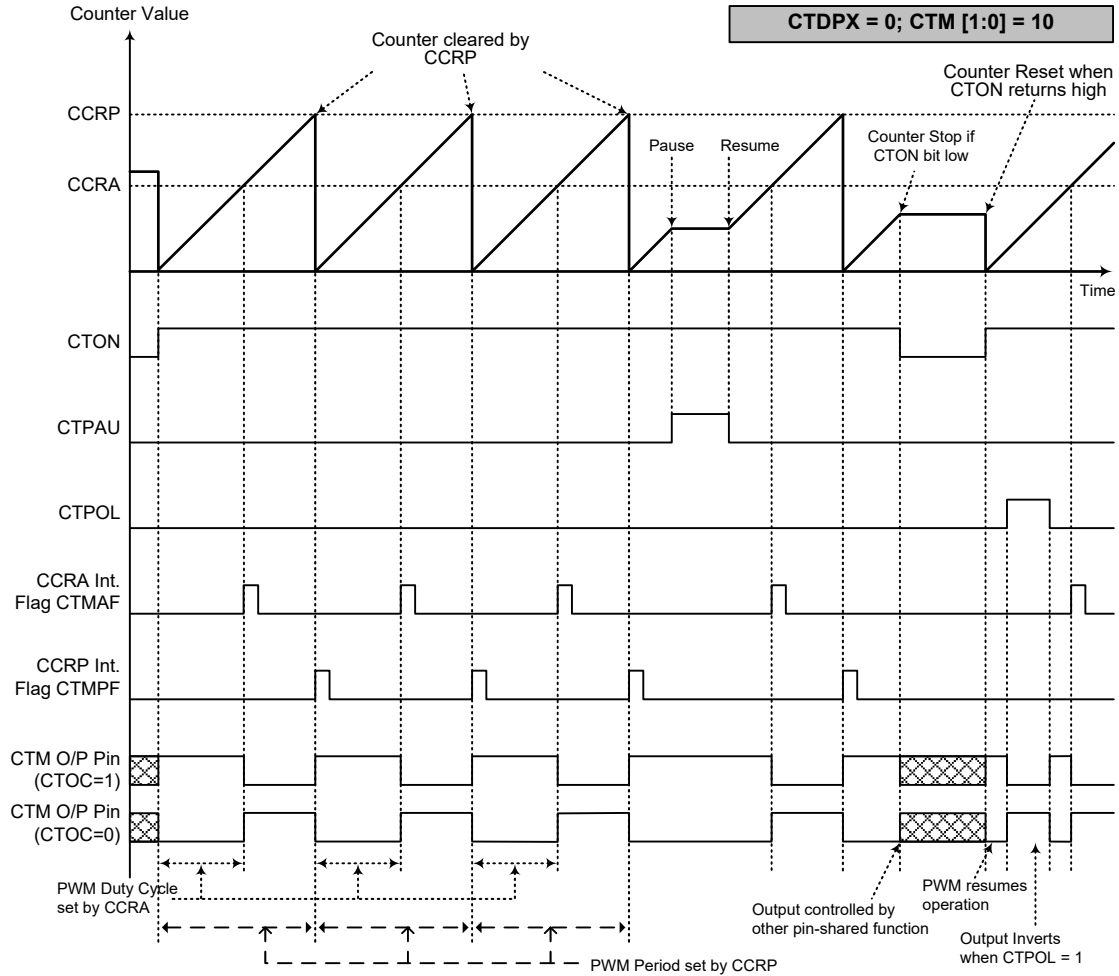
| CCRP | 1~255 | 0 |
|------|----------|-------|
| 周期 | CCRP×256 | 65536 |
| 占空比 | CCRA | |

若 $f_{sys}=8\text{MHz}$ ，CTM 时钟源选择 $f_{sys}/4$ ， $CCRP=2$ ， $CCRA=128$ ，
CTM PWM 输出频率 = $(f_{sys}/4)/(2 \times 256) = f_{sys}/2048 = 4\text{kHz}$ ，占空比 = $128/(2 \times 256) = 25\%$ ，
若由 CCRA 寄存器定义的占空比值等于或大于周期值，PWM 输出占空比为 100%。

- 16-bit CTM, PWM 输出模式, 边沿对齐模式, CTD PX=1

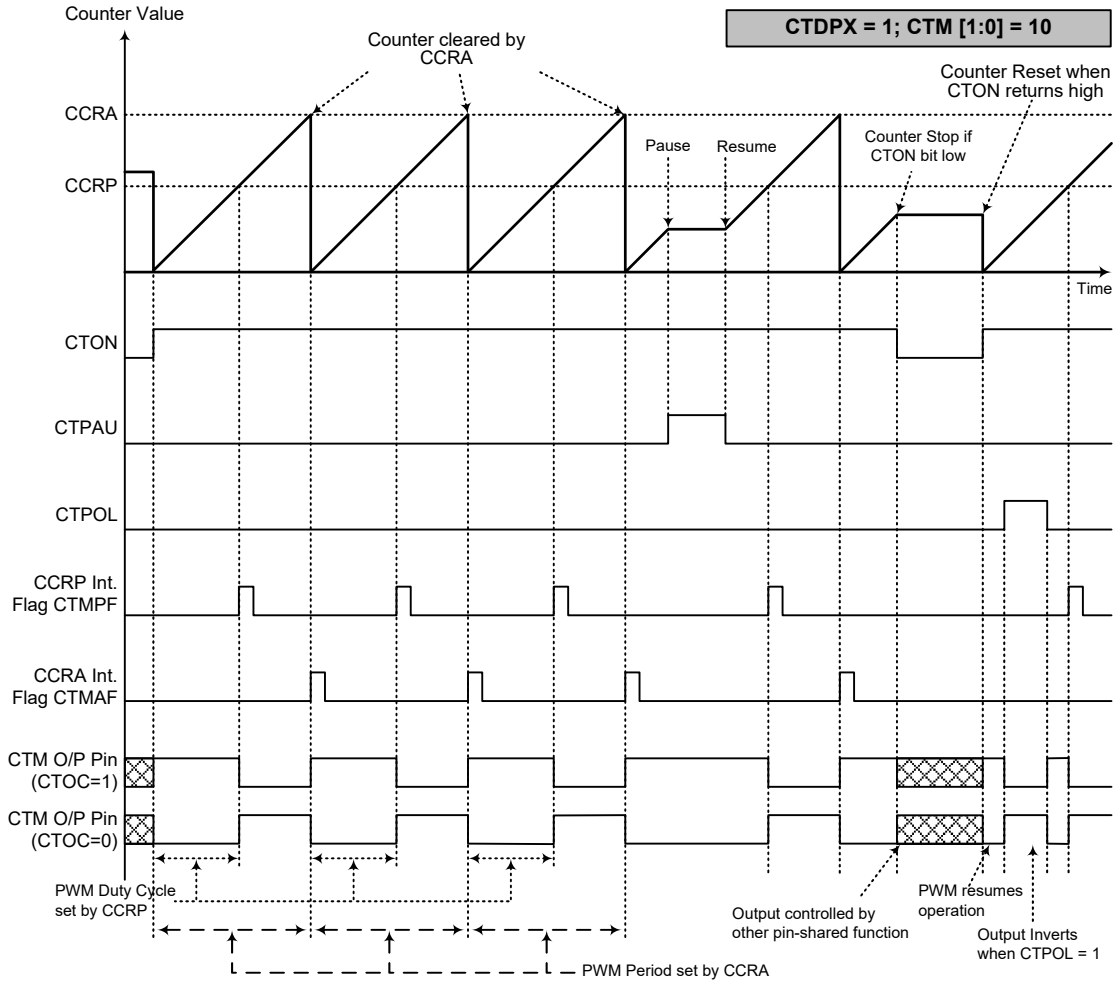
| CCRP | 1~255 | 0 |
|------|----------|-------|
| 周期 | CCRA | |
| 占空比 | CCRP×256 | 65536 |

PWM 的输出周期由 CCRA 寄存器的值与 CTM 的时钟共同决定，PWM 的占空比由 CCRP×256 (除了 CCRP 为“0”外) 的值决定。



PWM 输出模式 - CTD PX=0

- 注：1. CTD PX=0, CCRP 清除计数器
 2. 计数器清零并设置 PWM 周期
 3. 当 CTIO[1:0]=00 或 01, PWM 输出功能不变
 4. CTCCLR 位不影响 PWM 操作



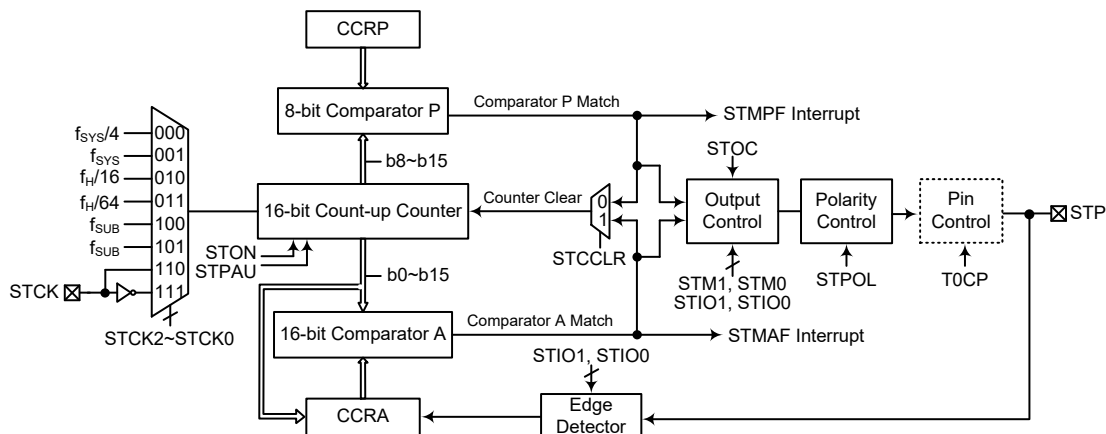
PWM 输出模式 - CTD PX=1

- 注：1. CTD PX=1, CCRA 清除计数器
2. 计数器清零并设置 PWM 周期
3. 当 CTIO[1:0]=00 或 01, PWM 输出功能不变
4. CTCCLR 位不影响 PWM 操作

标准型 TM – STM

标准型 TM 包括 5 种工作模式，即比较匹配输出、定时 / 事件计数器、捕捉输入、单脉冲输出和 PWM 输出模式。标准型 TM 由两个外部输入脚控制并驱动一个外部输出脚。

| STM 核心 | STM 输入引脚 | STM 输出引脚 |
|------------|-----------|----------|
| 16-bit STM | STCK, STP | STP |



16 位标准型 TM 方框图

标准型 TM 操作

标准型 TM 是 16 位宽度。核心是一个由用户选择的内部或外部时钟源驱动的 16 位向上计数器，它还包括两个内部比较器即比较器 A 和比较器 P。这两个比较器将计数器的值与 CCRP 和 CCRA 寄存器中的值进行比较。CCRP 是 8 位宽度，与计数器的高 8 位比较；而 CCRA 是 16 位的，与计数器的所有位比较。

通过应用程序改变 16 位计数器值的唯一方法是使 STON 位发生上升沿跳变清除计数器。此外，计数器溢出或比较匹配也会自动清除计数器。上述条件发生时，通常情况会产生 STM 中断信号。标准型 TM 可工作在不同的模式，可由包括来自输入脚的不同时钟源驱动，也可以控制输出脚。所有工作模式的设定都是通过设置相关寄存器来实现的。

标准型 TM 寄存器介绍

标准型 TM 的所有操作由一系列寄存器控制。一对只读寄存器用来存放 16 位计数器的值，一对读 / 写寄存器存放 16 位 CCRA 的值，STMRP 寄存器存放 8 位 CCRP 的值，剩下两个控制寄存器设置不同的操作和控制模式。

| 寄存器名称 | 位 | | | | | | | |
|-------|-------|-------|-------|-------|-------|-------|-------|--------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| STMC0 | STPAU | STCK2 | STCK1 | STCK0 | STON | — | — | — |
| STMC1 | STM1 | STM0 | STIO1 | STIO0 | STOC | STPOL | STDPX | STCCLR |
| STMDL | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| STMDH | D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 |
| STMAL | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| STMAH | D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 |
| STMRP | STRP7 | STRP6 | STRP5 | STRP4 | STRP3 | STRP2 | STRP1 | STRP0 |

16-bit 标准型 TM 寄存器列表

• STMC0 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|-------|-------|-------|------|---|---|---|
| Name | STPAU | STCK2 | STCK1 | STCK0 | STON | — | — | — |
| R/W | R/W | R/W | R/W | R/W | R/W | — | — | — |
| POR | 0 | 0 | 0 | 0 | 0 | — | — | — |

Bit 7 **STPAU**: STM 计数器暂停控制位

- 0: 运行
- 1: 暂停

通过设置此位为高可使计数器暂停，清零此位恢复正常计数器操作。当处于暂停条件时，STM 保持上电状态并继续耗电。当此位由低到高转换时，计数器将保留其剩余值，直到此位再次改变为低电平，并从此值开始继续计数。

Bit 6~4 **STCK2~STCK0**: 选择 STM 计数时钟位

- 000: $f_{SYS}/4$
- 001: f_{SYS}
- 010: $f_{H}/16$
- 011: $f_{H}/64$
- 100: f_{SUB}
- 101: f_{SUB}
- 110: STCK 上升沿时钟
- 111: STCK 下降沿时钟

此三位用于选择 STM 的时钟源。外部引脚时钟源能被选择在上升沿或下降沿有效。 f_{SYS} 是系统时钟， f_{H} 和 f_{SUB} 是其它的内部时钟源，详细请参考振荡器章节。

Bit 3 **STON**: STM 计数器 On/Off 控制位

- 0: Off
- 1: On

此位控制 STM 的总开关功能。设置此位为高则使能计数器使其运行，清零此位则除能 STM。清零此位将停止计数器并关闭 STM 减少耗电。当此位经由低到高转换时，内部计数器将复位清零；当此位经由高到低转换时，内部计数器将保持其剩余值，直到此位再次改变为高电平。若 STM 处于比较匹配输出模式时或 PWM 输出模式或单脉冲输出模式时，当 STON 位经由低到高转换时，STM 输出脚将复位至 STOC 位指定的初始值。

Bit 2~0 未定义，读为“0”

• STMC1 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|------|-------|-------|------|-------|--------|--------|
| Name | STM1 | STM0 | STIO1 | STIO0 | STOC | STPOL | STDPIX | STCCLR |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~6 **STM1~STM0**: 选择 STM 工作模式位

- 00: 比较匹配输出模式
- 01: 捕捉输入模式
- 10: PWM 输出模式或单脉冲输出模式
- 11: 定时 / 计数器模式

这两位设置 STM 需要的工作模式。为了确保操作可靠，STM 应在 STM1 和 STM0 位有任何改变前先关掉。在定时 / 计数器模式，STM 输出脚控制必须除能。

Bit 5~4 **STIO1~STIO0**: 选择 STP 引脚功能位

- 比较匹配输出模式
- 00: 无变化
 - 01: 输出低
 - 10: 输出高
 - 11: 输出翻转

PWM 输出模式 / 单脉冲输出模式

- 00: 强制无效状态
- 01: 强制有效状态
- 10: PWM 输出
- 11: 单脉冲输出

捕捉输入模式

- 00: 在 STP 上升沿输入捕捉
- 01: 在 STP 下降沿输入捕捉
- 10: 在 STP 双沿输入捕捉
- 11: 输入捕捉除能

定时 / 计数器模式

未使用

此两位用于决定在一定条件达到时 STM 输出脚如何改变状态。这两位值的选择取决于 STM 运行在何种模式下。

在比较匹配输出模式下，STIO1 和 STIO0 位决定当比较器 A 比较匹配输出发生时 STM 输出脚如何改变状态。当比较器 A 比较匹配输出发生时 STM 输出脚能设为切换高、切换低或翻转当前状态。若此两位同时为 0 时，这个输出将不会改变。STM 输出脚的初始值通过 STMC1 寄存器的 STOC 位设置取得。注意，由 STIO1 和 STIO0 位得到的输出电平必须与通过 STOC 位设置的初始值不同，否则当比较匹配发生时，STM 输出脚将不会发生变化。在 STM 输出脚改变状态后，通过 STON 位由低到高转换复位至初始值。在 PWM 输出模式，STIO1 和 STIO0 用于决定比较匹配条件发生时怎样改变 STM 输出脚的状态。PWM 输出功能通过这两位的变化进行更新。仅在 STM 关闭时改变 STIO1 和 STIO0 的值是很有必要的。若在 STM 运行时改变 STIO1 和 STIO0 值，PWM 输出的值是无法预料的。

Bit 3 **STOC: STM STP 输出控制位**

比较匹配输出模式

- 0: 初始低
- 1: 初始高

PWM 输出模式 / 单脉冲输出模式

- 0: 低有效
- 1: 高有效

这是 STM 输出脚输出控制位。它取决于 STM 此时正运行于比较匹配输出模式还是 PWM 输出模式，或单脉冲输出模式。若 STM 处于定时 / 计数器模式，则其不受影响。在比较匹配输出模式时，其决定比较匹配发生前 STM 输出脚的逻辑电平值。在 PWM 输出模式时，其决定 PWM 信号是高有效还是低有效。在单脉冲输出模式，其决定 STON 位由低变高时 STM 的逻辑电平。

Bit 2 **STPOL: STM STP 输出极性控制位**

- 0: 同相
- 1: 反相

此位控制 STP 输出脚的极性。此位为高时 STM 输出脚反相，为低时 STM 输出脚同相。若 STM 处于定时 / 计数器模式时其不受影响。

Bit 1 **STDPX: STM PWM 周期 / 占空比控制位**

- 0: CCRP – 周期; CCRA – 占空比
- 1: CCRP – 占空比; CCRA – 周期

此位决定 CCRA 与 CCRP 寄存器哪个被用于 PWM 波形的周期和占空比控制。

Bit 0 **STCCLR: 选择 STM 计数器清零条件位**

- 0: 比较器 P 匹配
- 1: 比较器 A 匹配

此位用于选择清除计数器的方法。标准型 TM 包括两个比较器 – 比较器 A 和比较器 P。这两个比较器每个都可以用作清除内部计数器。STCCLR 位设为高，计数器在比较器 A 比较匹配发生时被清除；此位设为低，计数器在比较器 P 比较匹配发生或计数器溢出时被清除。计数器溢出清除的方法仅在 CCRP 被清除为 0 时才能生效。STCCLR 位在 PWM 输出模式、单脉冲输出模式和输入捕捉模式时未使用。

• **STMDL 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|----|----|----|----|----|----|----|----|
| Name | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R | R | R | R | R | R | R | R |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **D7~D0**: STM 计数器低字节寄存器 bit 7 ~ bit 0
STM 16-bit 计数器 bit 7 ~ bit 0

• **STMDH 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|-----|-----|-----|-----|-----|----|----|
| Name | D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 |
| R/W | R | R | R | R | R | R | R | R |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **D15~D8**: STM 计数器高字节寄存器 bit 7 ~ bit 0
STM 16-bit 计数器 bit 15 ~ bit 8

• **STMAL 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| Name | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **D7~D0**: STM CCRA 低字节寄存器 bit 7 ~ bit 0
STM 16-bit CCRA bit 7 ~ bit 0

• **STMAH 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| Name | D15 | D14 | D13 | D12 | D11 | D10 | D9 | D8 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **D15~D8**: STM CCRA 高字节寄存器 bit 7 ~ bit 0
STM 16-bit CCRA bit 15 ~ bit 8

• **STM RP 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|-------|-------|-------|-------|-------|-------|-------|
| Name | STRP7 | STRP6 | STRP5 | STRP4 | STRP3 | STRP2 | STRP1 | STRP0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **STRP7~STRP0**: STM CCRP 8 位寄存器, 与 STM 计数器 bit 15 ~ bit 8 比较。

比较器 P 匹配周期 =

0: 65536 个 STM 时钟

1~255: (1~255)×256 个 STM 时钟

此八位设定内部 CCRP 8-bit 寄存器的值, 然后与内部计数器的高八位进行比较。如果 STCCLR 位设为 0 时, 比较结果为 0 并清除内部计数器。STCCLR 位设为低, CCRP 比较匹配结果将重置内部计数器。由于 CCRP 只与计数器高八位比较, 比较结果是 256 时钟周期的倍数。CCRP 被清零时, 实际上会使得计数器在最大值溢出。

标准型 TM 工作模式

标准型 TM 有五种工作模式，即比较匹配输出模式，PWM 输出模式，单脉冲输出模式，捕捉输入模式或定时 / 计数器模式。通过设置 STMC1 寄存器的 STM1 和 STM0 位选择任意模式。

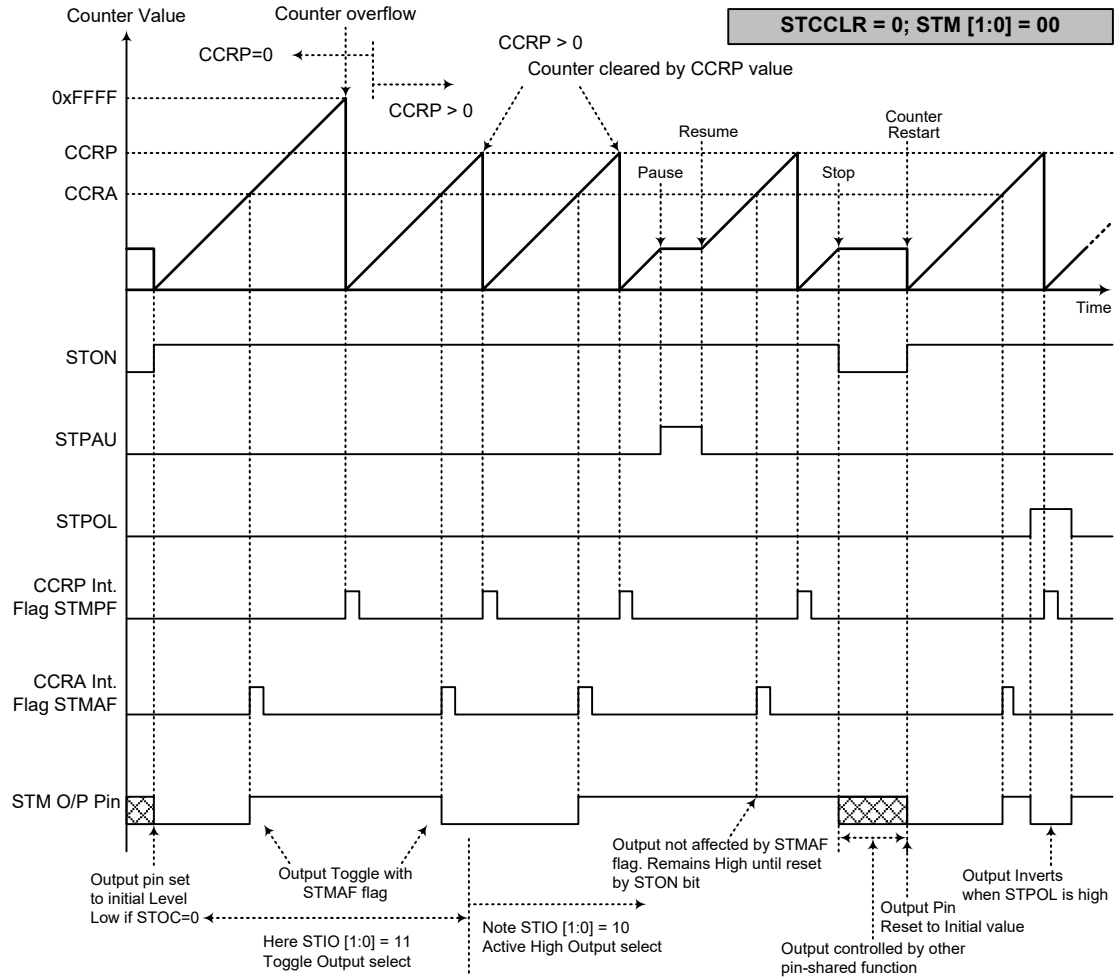
比较匹配输出模式

为使 STM 工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“00”。当工作在该模式，一旦计数器使能并开始计数，有三种方法来清零，分别是：计数器溢出，比较器 A 比较匹配发生和比较器 P 比较匹配发生。当 STCCLR 位为低，有两种方法清除计数器。一种是比较器 P 比较匹配发生，另一种是 CCRP 所有位设置为零并使得计数器溢出。此时，比较器 A 和比较器 P 的请求标志位 STMAF 和 STMPF 将分别置位。

如果 STMC1 寄存器的 STCCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅产生 STMAF 中断请求标志。所以当 STCCLR 为高时，不会产生 STMPF 中断请求标志。在比较匹配输出模式下，CCRA 不能设为“0”。

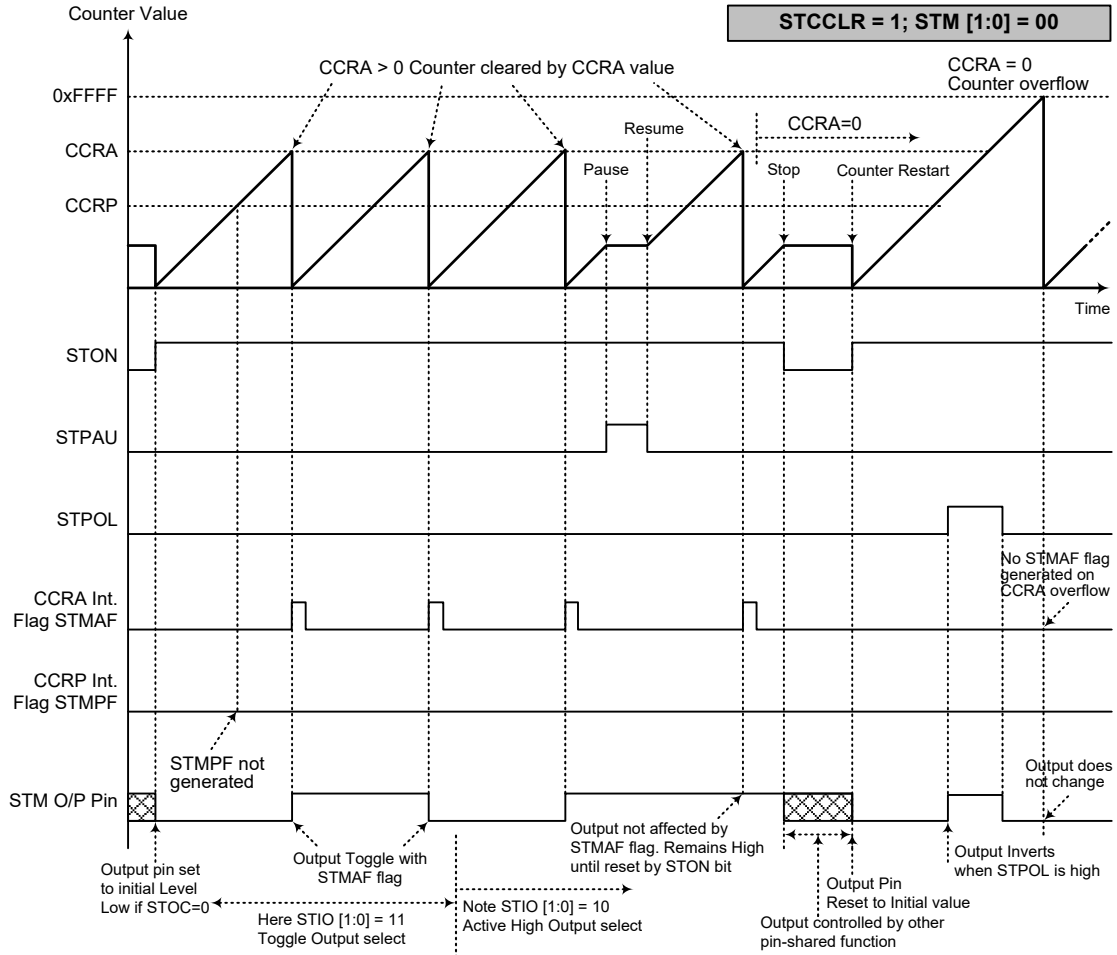
如果 CCRA 位都清除为零，当计数器的值达到 16 最大值 FFFF 时将溢出，但此时不会产生 STMAF 中断请求标志。

正如该模式名所言，当比较匹配发生后，STM 输出脚状态改变。当比较器 A 比较匹配发生后 STMAF 标志产生时，STM 输出脚状态改变。比较器 P 比较匹配发生时产生的 STMPF 标志不影响 STM 输出脚。STM 输出脚状态改变方式由 STMC1 寄存器中 STIO1 和 STIO0 位决定。当比较器 A 比较匹配发生时，TnIO1 和 TnIO0 位决定 STM 输出脚输出高，低或翻转当前状态。在 STON 位由低到高电平的变化后，STM 输出脚初始状态为 STOC 位所指定的电平。注意，若 STIO1 和 STIO0 位同时为 0 时，引脚输出不变。



比较匹配输出模式 – STCCLR=0

- 注：1. STCCLR=0，比较器 P 匹配将清除计数器
2. STM 输出脚仅由 STMAF 标志位控制
3. 在 STON 上升沿 STM 输出脚复位至初始值



比较匹配输出模式 – STCCLR=1

- 注：1. STCCLR=1，比较器 A 匹配将清除计数器
- 2. STM 输出脚仅由 STMAF 标志位控制
- 3. 在 STON 上升沿 STM 输出脚复位至初始值
- 4. 当 STCCLR=1 时，不会产生 STMPF 标志

定时 / 计数器模式

为使 STM 工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“11”。定时 / 计数器模式与比较输出模式操作方式相同，并产生同样的中断请求标志。不同的是，在定时 / 计数器模式下 STM 输出脚未使用。因此，比较匹配输出模式中的描述和时序图可以适用于此功能。该模式中未使用的 STM 输出脚用作普通 I/O 脚或其它功能。

PWM 输出模式

为使 STM 工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“10”，且 STIO1 和 STIO0 位也需要设置为“10”。STM 的 PWM 输出功能在马达控制，加热控制，照明控制等方面十分有用。给 STM 输出脚提供一个频率固定但占空比可调的信号，将产生一个有效值等于 DC 均方根的 AC 方波。

由于 PWM 波形的周期和占空比可调，其波形的选择就较为灵活。在 PWM 输出模式中，STCCLR 位不影响 PWM 周期。CCRA 和 CCRP 寄存器决定 PWM 波形，一个用来清除内部计数器并控制 PWM 波形的频率，另一个用来控制占空比。哪个寄存器控制频率或占空比取决于 STMC1 寄存器的 STDPX 位。所以 PWM 波形由 CCRA 和 CCRP 寄存器共同决定。

当比较器 A 或比较器 P 比较匹配发生时，将产生 CCRA 或 CCRP 中断标志。STMC1 寄存器中的 STOC 位决定 PWM 波形的极性，STIO1 和 STIO0 位使能 PWM 输出或将 STM 输出脚置为逻辑高或逻辑低。STPOL 位对 PWM 输出波形的极性取反。

● 16-bit STM, PWM 输出模式, 边沿对齐模式, STDPX=0

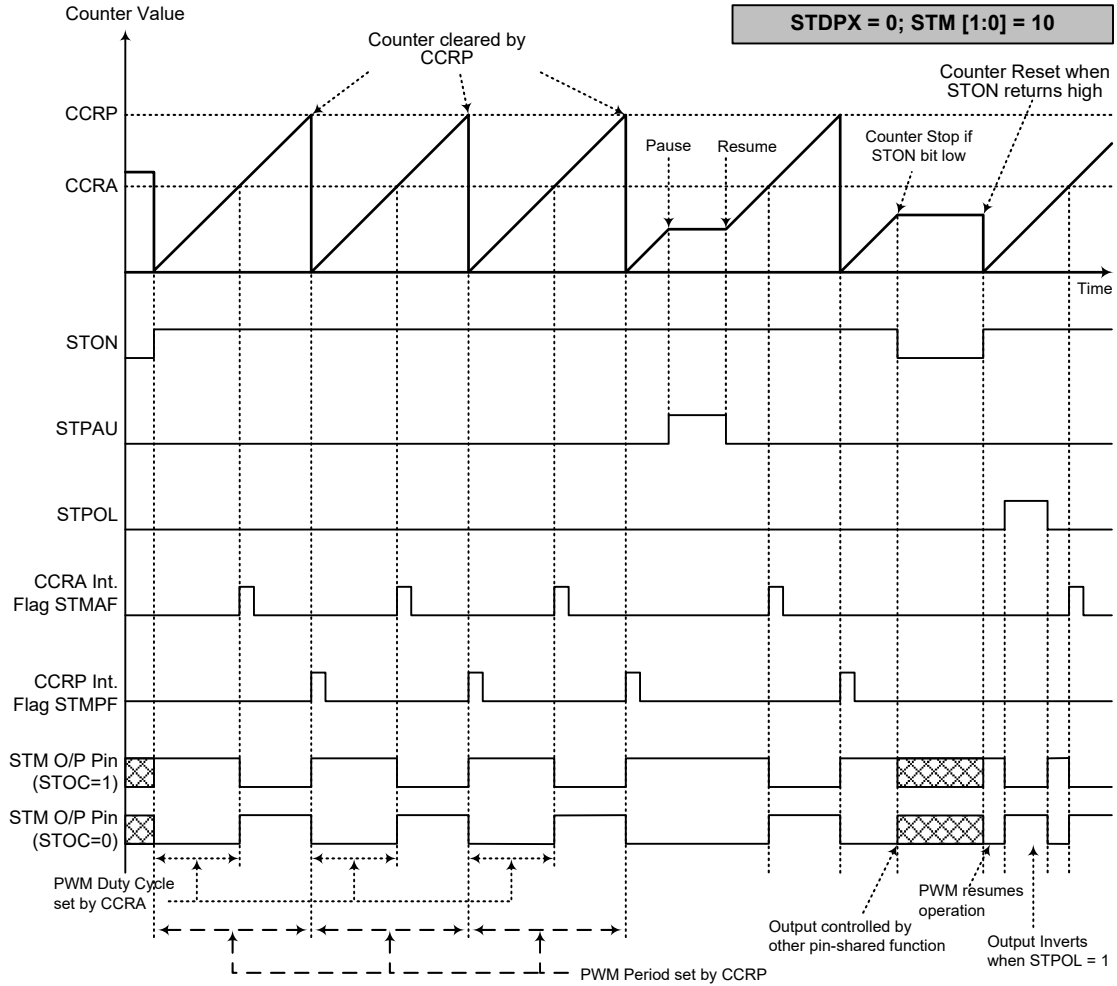
| CCRP | 1~255 | 0 |
|------|----------|-------|
| 周期 | CCRP×256 | 65536 |
| 占空比 | CCRA | |

若 $f_{sys}=8\text{MHz}$ ，STM 时钟源选择 $f_{sys}/4$ ， $CCRP=2$ ， $CCRA=128$ ，
STM PWM 输出频率 = $(f_{sys}/4)/(2 \times 256) = f_{sys}/2048 = 4\text{kHz}$ ，占空比 = $128/(2 \times 256) = 25\%$ ，
若由 CCRA 寄存器定义的占空比值等于或大于周期值，PWM 输出占空比为 100%。

● 16-bit STM, PWM 输出模式, 边沿对齐模式, STDPX=1

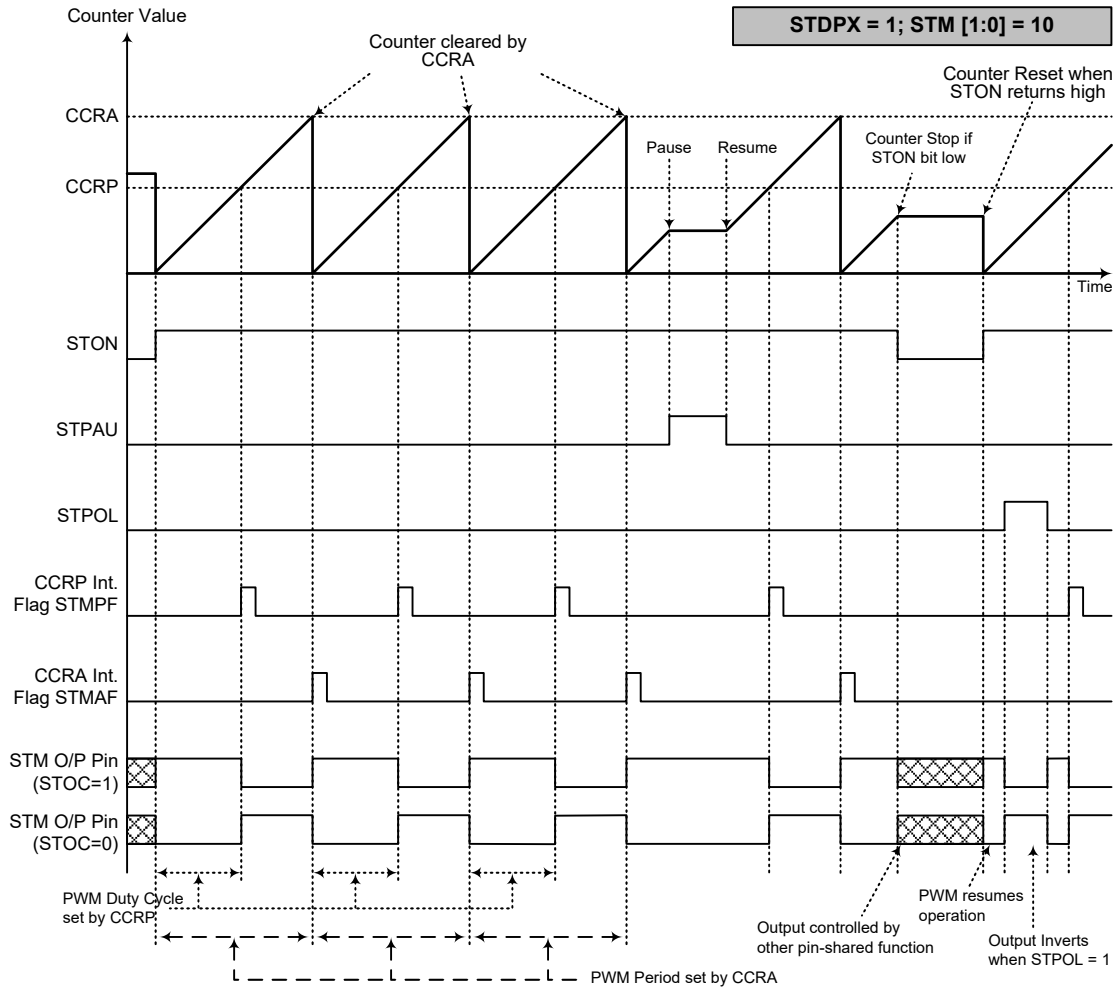
| CCRP | 1~255 | 0 |
|------|----------|-------|
| 周期 | CCRA | |
| 占空比 | CCRP×256 | 65536 |

PWM 的输出周期由 CCRA 寄存器的值与 STM 的时钟共同决定，PWM 的占空比由 $CCRP \times 256$ (除了 CCRP 为“0”外) 的值决定。



PWM 输出模式 – STDPX=0

- 注：1. STDPX=0, CCRP 清除计数器
 2. 计数器清零并设置 PWM 周期
 3. 当 STIO[1:0]=00 或 01, PWM 输出功能不变
 4. STCCLR 位不影响 PWM 操作



PWM 输出模式 – STDPX=1

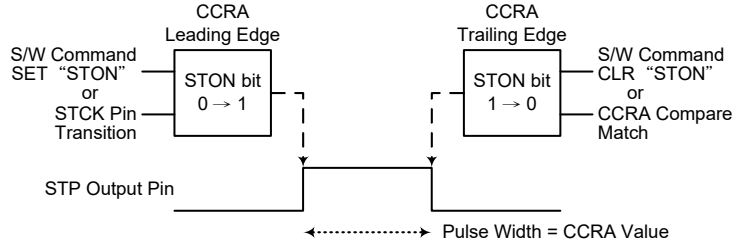
- 注：1. STDPX=1, CCRA 清除计数器
2. 计数器清零并设置 PWM 周期
3. 当 STIO[1:0]=00 或 01, PWM 输出功能不变
4. STCCLR 位不影响 PWM 操作

单脉冲输出模式

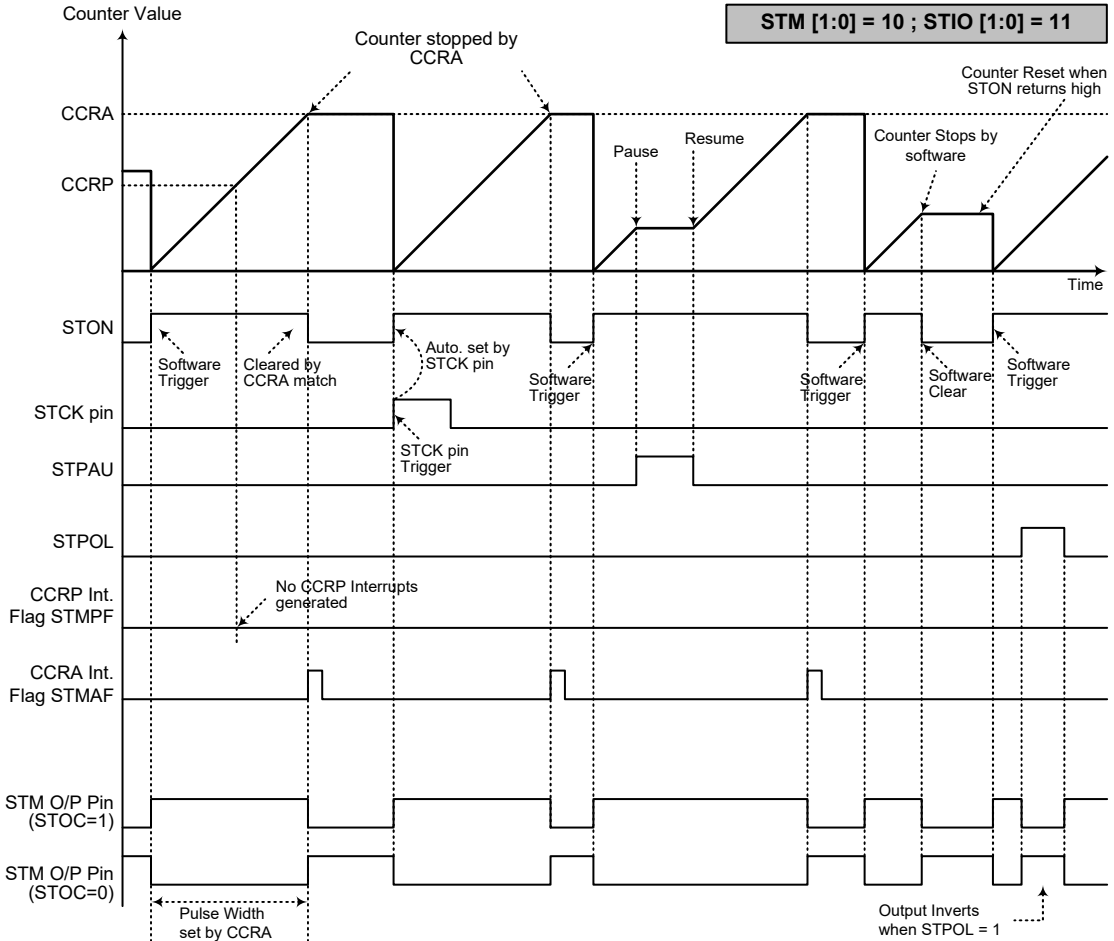
为使 STM 工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“10”，同时 STIO1 和 STIO0 位需要设置为“11”。正如模式名所言，单脉冲输出模式，在 STM 输出脚将产生一个脉冲输出。

通过应用程序控制 STON 位由低到高的转变来触发脉冲前沿输出。而处于单脉冲输出模式时，STON 位可在 STCK 脚发生有效边沿跳转时自动由低转变为高，进而开始单脉冲输出。当 STON 位转变为高电平时，计数器将开始运行，并产生脉冲前沿。当脉冲有效时 STON 位保持高电平。通过应用程序使 STON 位清零或比较器 A 比较匹配发生时，产生脉冲后沿。

然而，比较器 A 比较匹配发生时，会自动清除 STON 位并产生单脉冲输出边沿转换。CCRA 的值通过这种方式控制脉冲宽度。比较器 A 比较匹配发生时，也会产生 STM 中断。STON 位在计数器重启时会发生由低到高的转变，此时计数器才复位至零。在单脉冲模式中，CCRP 寄存器，STCCLR 和 STDPX 位未使用。



单脉冲产生示意图



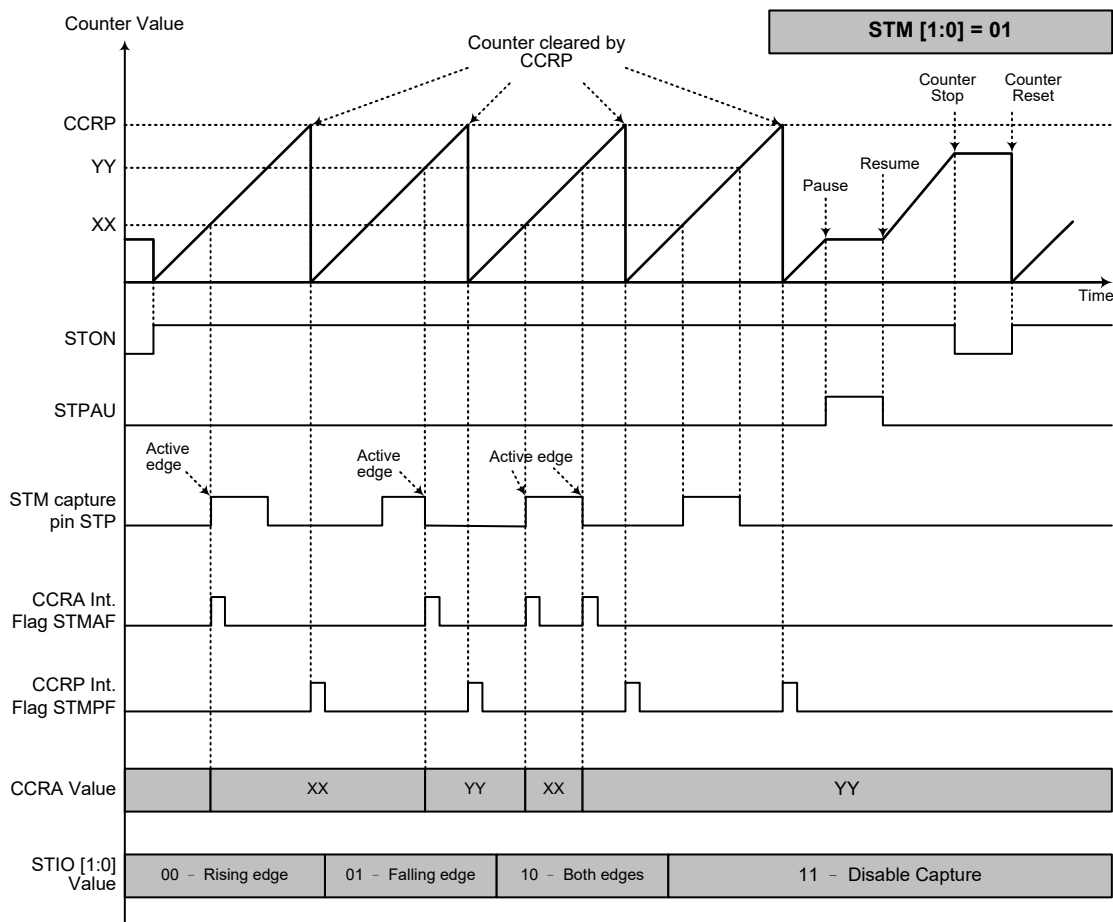
单脉冲输出模式

- 注：1. 通过 CCRA 匹配停止计数器
2. CCRP 未使用
3. 通过 STCK 脚或设置 STON 位为高来触发脉冲
4. STCK 脚有效沿会自动置位 STON
5. 单脉冲输出模式中，STIO[1:0] 需置位“11”，且不能更改。

捕捉输入模式

为使 STM 工作在此模式，STM1 寄存器中的 STM1 和 STM0 位需要设置为“01”。此模式使能外部信号捕捉并保存内部计数器当前值，因此被用于诸如脉冲宽度测量的应用中。STP 脚上的外部信号，通过设置 STMC1 寄存器的 STIO1 和 STIO0 位选择有效边沿类型，即上升沿，下降沿或双沿有效。计数器在 STON 位由低到高转变时启动并通过应用程序初始化。

当 STP 脚出现有效边沿转换时，计数器当前值被锁存到 CCRA 寄存器，并产生 STM 中断。无论 STP 引脚发生哪种边沿转换，计数器继续工作直到 STON 位发生下降沿跳变。当 CCRP 比较匹配发生时计数器复位至零；CCRP 的值通过这种方式控制计数器的最大值。当比较器 P CCRP 比较匹配发生时，也会产生 STM 中断。记录 CCRP 溢出中断信号的值可以测量长脉宽。通过设置 STIO1 和 STIO0 位选择 STP 引脚为上升沿，下降沿或双沿有效。如果 STIO1 和 STIO0 都设置为高，无论 STP 引脚发生哪种边沿转换都不会产生捕捉操作，但计数器继续运行。STCCLR 和 STDPX 位在此模式中未使用。

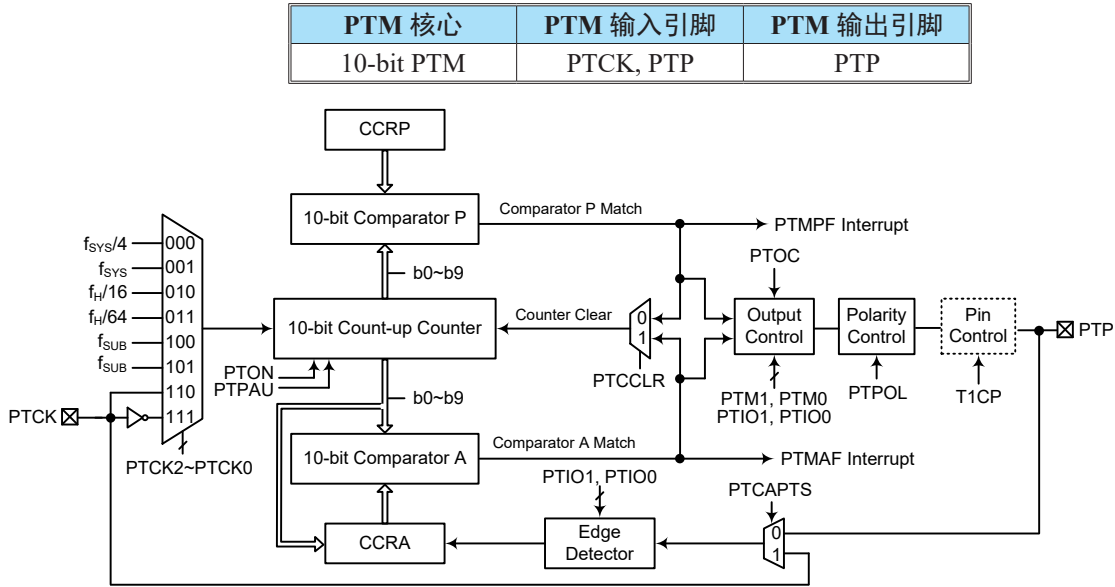


捕捉输入模式

- 注：1. STM[1:0]=01 并通过 STIO[1:0] 位设置有效边沿
2. STM 捕捉输入脚的有效边沿将计数器的值转移到 CCRA 中
3. STCCLR 位未使用
4. 无输出功能 – STOC 和 STPOL 位未使用
5. 计数器值由 CCRP 决定，在 CCRP 为“0”时，计数器计数值可达最大

周期型 TM – PTM

周期型 TM 包括 5 种工作模式，即比较匹配输出、定时 / 事件计数器、捕捉输入、单脉冲输出和 PWM 输出模式。周期型 TM 由两个外部输入脚控制并驱动一个外部输出脚。



10 位周期型 TM 方框图

周期型 TM 操作

周期型 TM 是 10 位宽度。周期型 TM 核心是一个由用户选择的内部或外部时钟源驱动的 10 位向上计数器，它还包括两个内部比较器即比较器 A 和比较器 P。这两个比较器将计数器的值与 CCRA 和 CCRP 寄存器中的值进行比较。CCRP 和 CCRA 是 10 位的，与计数器的所有位比较。

通过应用程序改变 10 位计数器值的唯一方法是使 PTON 位发生上升沿跳变清除计数器。此外，计数器溢出或比较匹配也会自动清除计数器。上述条件发生时，通常情况会产生 PTM 中断信号。周期型 TM 可工作在不同的模式，可由包括来自输入脚的不同时钟源驱动，也可以控制输出脚。所有工作模式的设定都是通过设置相关寄存器来实现的。

周期型 TM 寄存器介绍

周期型 TM 的所有操作由一系列寄存器控制。一对只读寄存器用来存放 10 位计数器的值，两对读 / 写寄存器存放 10 位 CCRA 和 CCRP 的值。剩下两个控制寄存器用来设置不同的操作和控制模式。

| 寄存器名称 | 位 | | | | | | | |
|--------|-------|-------|-------|-------|-------|-------|---------|--------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PTMC0 | PTPAU | PTCK2 | PTCK1 | PTCK0 | PTON | — | — | — |
| PTMC1 | PTM1 | PTM0 | PTIO1 | PTIO0 | PTOC | PTPOL | PTCAPTS | PTCCLR |
| PTMDL | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| PTMDH | — | — | — | — | — | — | D9 | D8 |
| PTMAL | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| PTMAH | — | — | — | — | — | — | D9 | D8 |
| PTMRPL | PTRP7 | PTRP6 | PTRP5 | PTRP4 | PTRP3 | PTRP2 | PTRP1 | PTRP0 |
| PTMRPH | — | — | — | — | — | — | PTRP9 | PTRP8 |

10 位周期型 TM 寄存器列表

● PTMC0 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|-------|-------|-------|------|---|---|---|
| Name | PTPAU | PTCK2 | PTCK1 | PTCK0 | PTON | — | — | — |
| R/W | R/W | R/W | R/W | R/W | R/W | — | — | — |
| POR | 0 | 0 | 0 | 0 | 0 | — | — | — |

Bit 7 **PTPAU**: PTM 计数器暂停控制位

0: 运行
1: 暂停

通过设置此位为高可使计数器暂停，清零此位恢复正常计数器操作。当处于暂停条件时，PTM 保持上电状态并继续耗电。当此位由低到高转变时，计数器将保留其剩余值，直到此位再次改变为低电平，并从此值开始继续计数。

Bit 6~4 **PTCK2~PTCK0**: 选择 PTM 计数时钟位

000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: f_{SUB}
110: PTCK 上升沿
111: PTCK 下降沿

此三位用于选择 PTM 的时钟源。外部引脚时钟源能被选择在上升沿或下降沿有效。 f_{SYS} 是系统时钟， f_H 和 f_{SUB} 是其它的内部时钟源，细节方面请参考振荡器章节。

Bit 3 **PTON**: PTM 计数器 On/Off 控制位

0: Off
1: On

此位控制 PTM 的总开关功能。设置此位为高则使能计数器使其运行，清零此位则除能 PTM。清零此位将停止计数器并关闭 PTM 减少耗电。当此位经由低到高转换时，内部计数器将复位清零；当此位经由高到低转换时，内部计数器将保持其剩余值，直到此位再次改变为高电平。

若 PTM 处于比较匹配输出模式或 PWM 输出模式或单脉冲输出模式时，当 PTON 位经由低到高转换时，PTM 输出脚将复位至 PTOC 位指定的初始值。

Bit 2~0 未定义，读为“0”

• PTMC1 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|------|-------|-------|------|-------|---------|--------|
| Name | PTM1 | PTM0 | PTIO1 | PTIO0 | PTOC | PTPOL | PTCAPTS | PTCCLR |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~6 **PTM1~PTM0**: 选择 PTM 工作模式位

- 00: 比较匹配输出模式
- 01: 捕捉输入模式
- 10: PWM 输出模式或单脉冲输出模式
- 11: 定时 / 计数器模式

这两位设置 PTM 需要的工作模式。为了确保操作可靠，PTM 应在 PTM1 和 PTM0 位有任何改变前先关掉。在定时 / 计数器模式，PTM 输出脚控制必须除能。

Bit 5~4 **PTIO1~PTIO0**: 选择 PTM 外部引脚功能位

- 比较匹配输出模式
 - 00: 无变化
 - 01: 输出低
 - 10: 输出高
 - 11: 输出翻转
- PWM 输出模式 / 单脉冲输出模式
 - 00: 强制无效状态
 - 01: 强制有效状态
 - 10: PWM 输出
 - 11: 单脉冲输出
- 捕捉输入模式
 - 00: 在 PTP 或 PTCK 上升沿输入捕捉
 - 01: 在 PTP 或 PTCK 下降沿输入捕捉
 - 10: 在 PTP 或 PTCK 双沿输入捕捉
 - 11: 输入捕捉除能
- 定时 / 计数器模式
 - 未使用

此两位用于决定在一定条件达到时 PTM 输出脚如何改变状态。这两位值的选择取决于 PTM 运行在何种模式下。

在比较匹配输出模式下，PTIO1 和 PTIO0 位决定当从比较器 A 比较匹配输出发生时 PTM 输出脚如何改变状态。当从比较器 A 比较匹配输出发生时 PTM 输出脚能设为切换高、切换低或翻转当前状态。若此两位同时为 0 时，这个输出将不会改变。PTM 输出脚的初始值通过 PTMC1 寄存器的 PTOC 位设置取得。注意，由 PTIO1 和 PTIO0 位得到的输出电平必须与通过 PTOC 位设置的初始值不同，否则当比较匹配发生时，PTM 输出脚将不会发生变化。在 PTM 输出脚改变状态后，通过 PTON 位由低到高电平的转换复位至初始值。

在 PWM 输出模式，PTIO1 和 PTIO0 用于决定比较匹配条件发生时怎样改变 PTM 输出脚的状态。PWM 输出功能通过这两位的变化进行更新。必须在 PTM 关闭时才能改变 PTIO1 和 PTIO0 位的值。若在 PTM 运行时改变 PTIO1 和 PTIO0 的值，PWM 输出的值是无法预料的。

Bit 3 **PTOC**: PTM PTP 输出控制位

- 比较匹配输出模式
 - 0: 初始低
 - 1: 初始高
- PWM 输出模式 / 单脉冲输出模式
 - 0: 低有效
 - 1: 高有效

这是 PTM 输出脚输出控制位。它取决于 PTM 此时正运行于比较匹配输出模式还是 PWM 输出模式 / 单脉冲输出模式。若 PTM 处于定时 / 计数器模式，则其不受影响。在比较匹配输出模式时，其决定比较匹配发生前 PTM 输出脚的逻辑

电平值。在 PWM 输出模式 / 单脉冲输出时，其决定 PWM 信号是高有效还是低有效。在单脉冲输出模式，其决定 PTON 位由低变高时 PTM 输出脚的逻辑电平。

Bit 2 **PTPOL**: PTM PTP 输出极性控制位

0: 同相
1: 反相

此位控制 PTP 输出脚的极性。此位为高时 PTM 输出脚反相，为低时 PTM 输出脚同相。若 PTM 处于定时 / 计数器模式时其不受影响。

Bit 1 **PTCAPTS**: 选择 PTM 捕捉触发源

0: 来自 PTP 引脚
1: 来自 PTCK 引脚

Bit 0 **PTCCLR**: 选择 PTM 计数器清零条件位

0: 比较器 P 匹配
1: 比较器 A 匹配

此位用于选择清除计数器的方法。周期型 TM 包括两个比较器 - 比较器 A 和比较器 P。这两个比较器每个都可以用作清除内部计数器。PTCCLR 位设为高，计数器在比较器 A 比较匹配发生时被清除；此位设为低，计数器在比较器 P 比较匹配发生或计数器溢出时被清除。计数器溢出清除的方法仅在 CCRP 被清除为 0 时才能生效。PTCCLR 位在 PWM，单脉冲或输入捕捉模式时未使用。

● PTMDL 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|----|----|----|----|----|----|----|----|
| Name | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R | R | R | R | R | R | R | R |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **D7~D0**: PTM 计数器低字节寄存器 bit 7 ~ bit 0

PTM 10-bit 计数器 bit 7 ~ bit 0

● PTMDH 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|---|---|----|----|
| Name | — | — | — | — | — | — | D9 | D8 |
| R/W | — | — | — | — | — | — | R | R |
| POR | — | — | — | — | — | — | 0 | 0 |

Bit 7~2 未定义，读为“0”

Bit 1~0 **D9~D8**: PTM 计数器高字节寄存器 bit 1 ~ bit 0

PTM 10-bit 计数器 bit 9 ~ bit 8

● PTMAL 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| Name | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **D7~D0**: PTM CCRA 低字节寄存器 bit 7 ~ bit 0

PTM 10-bit CCRA bit 7 ~ bit 0

● **PTMAH 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|---|---|-----|-----|
| Name | — | — | — | — | — | — | D9 | D8 |
| R/W | — | — | — | — | — | — | R/W | R/W |
| POR | — | — | — | — | — | — | 0 | 0 |

Bit 7~2 未定义，读为“0”

Bit 1~0 **D9~D8**: PTM CCRA 高字节寄存器 bit 1 ~ bit 0
PTM 10-bit CCRA bit 9 ~ bit 8

● **PTMRPL 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|-------|-------|-------|-------|-------|-------|-------|
| Name | PTRP7 | PTRP6 | PTRP5 | PTRP4 | PTRP3 | PTRP2 | PTRP1 | PTRP0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7~0 **PTRP7~PTRP0**: PTM CCRP 低字节寄存器 bit 7 ~ bit 0
PTM 10-bit CCRP bit 7 ~ bit 0

● **PTMRPH 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|---|---|-------|-------|
| Name | — | — | — | — | — | — | PTRP9 | PTRP8 |
| R/W | — | — | — | — | — | — | R/W | R/W |
| POR | — | — | — | — | — | — | 0 | 0 |

Bit 7~2 未定义，读为“0”

Bit 1~0 **PTRP9~PTRP8**: PTM CCRP 高字节寄存器 bit 1 ~ bit 0
PTM 10-bit CCRP bit 9 ~ bit 8

周期型 TM 工作模式

周期型 TM 有五种工作模式，即比较匹配输出模式、PWM 输出模式、单脉冲输出模式、捕捉输入模式或定时 / 计数器模式。通过设置 PTMC1 寄存器的 PTM1 和 PTM0 位选择任意模式。

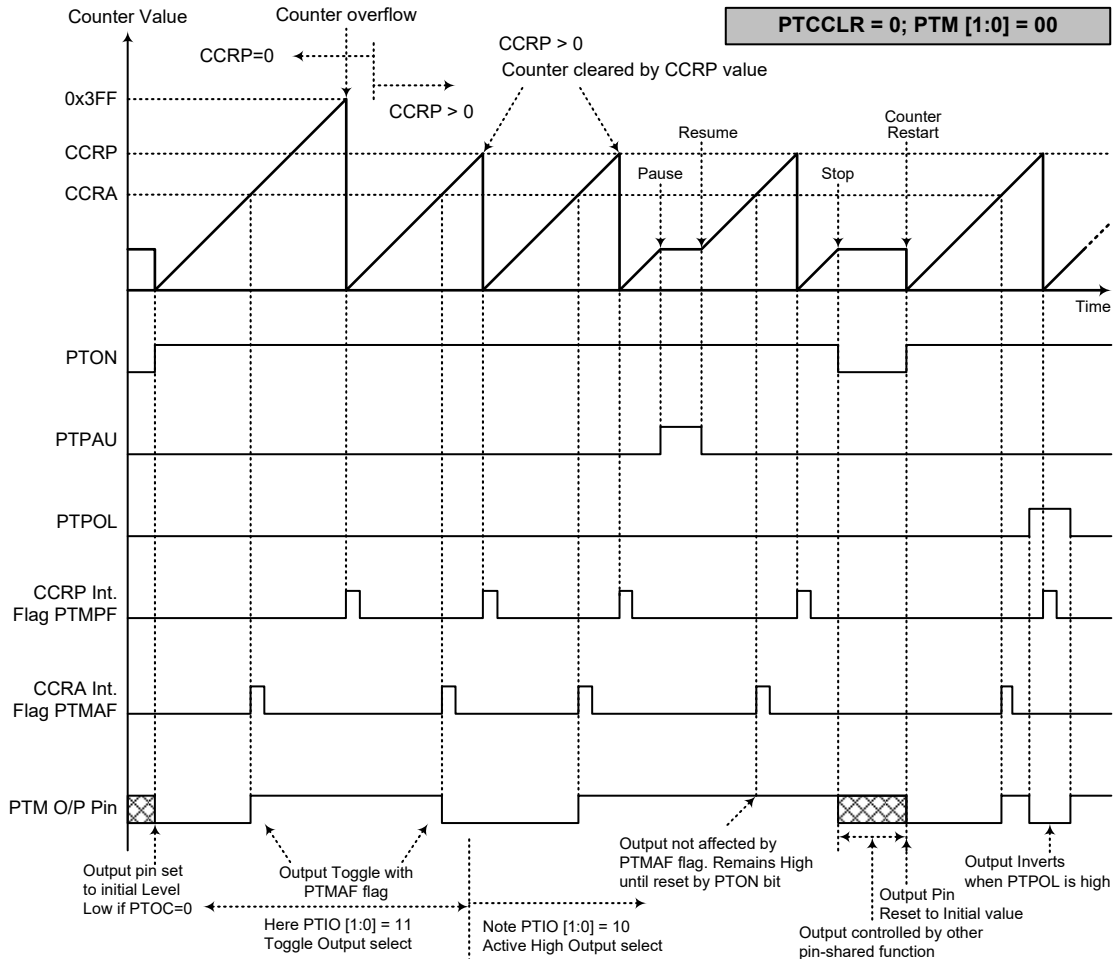
比较匹配输出模式

为使 PTM 工作在此模式，PTMC1 寄存器的 PTM1 和 PTM0 位需要设置为“00”。当工作在该模式，一旦计数器使能并开始计数，有三种方法来清零，分别是：计数器溢出，比较器 A 比较匹配发生和比较器 P 比较匹配发生。当 PTCCLR 位为低，有两种方法清除计数器。一种是比较器 P 比较匹配发生，另一种是 CCRP 所有位设置为零并使得计数器溢出。此时，比较器 A 和比较器 P 的请求标志位 PTMAF 和 PTMPF 将分别置起。

如果 PTMC1 寄存器的 PTCCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅 PTMAF 中断请求标志产生。所以当 PTCCLR 为高时，不会产生 PTMPF 中断请求标志。在比较匹配输出模式中，CCRA 寄存器值不能设为“0”。

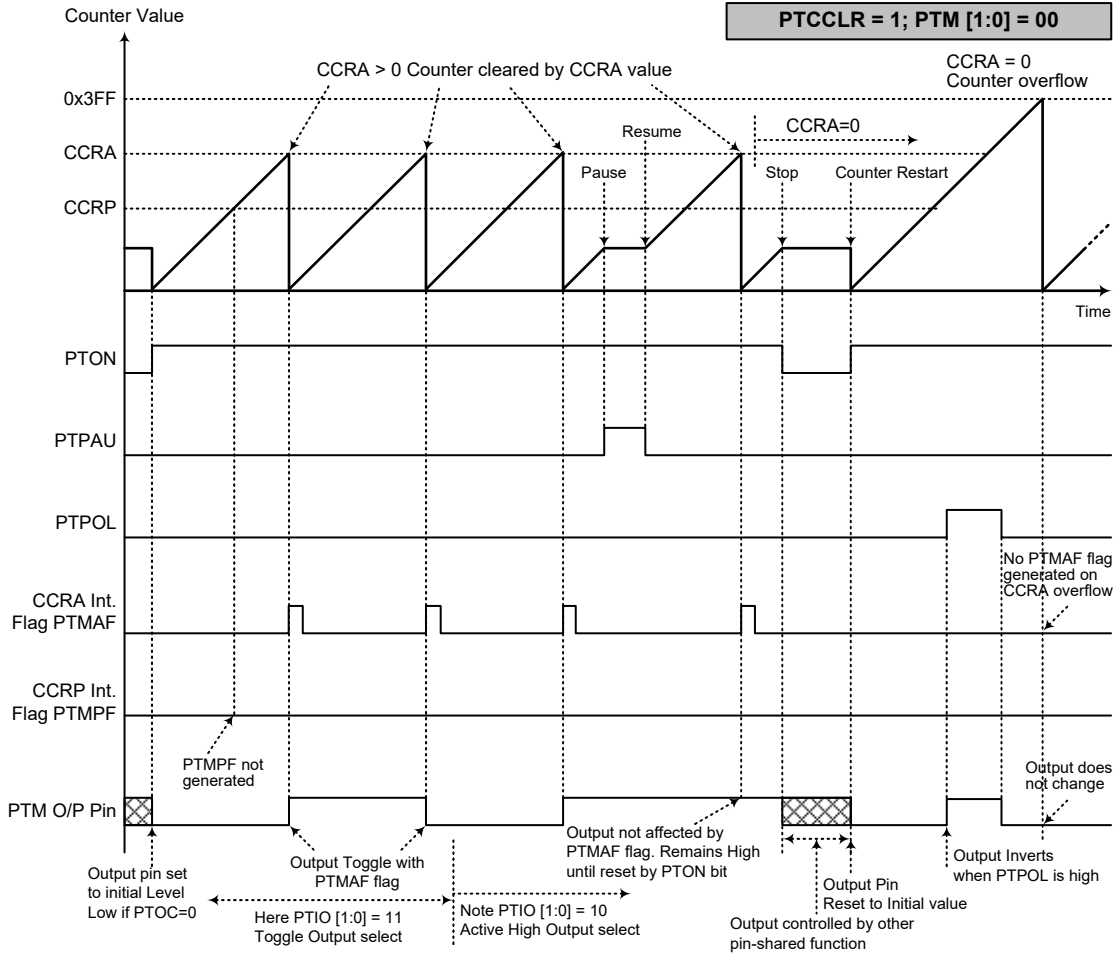
如果 CCRA 位都清除为零，当计数器的值达到 10 位最大值 3FFH 时将溢出，但此时不会产生 PTMAF 中断请求标志。

正如该模式名所言，当比较匹配发生后，PTM 输出脚状态改变。当比较器 A 比较匹配发生后 PTMAF 中断请求标志产生时，PTM 输出脚状态改变。比较器 P 比较匹配发生时产生的 PTMPF 标志不影响 PTM 输出脚。PTM 输出脚状态改变方式由 PTMC1 寄存器中 PTIO1 和 PTIO0 位决定。当比较器 A 比较匹配发生时，PTIO1 和 PTIO0 位决定 PTM 输出脚输出高，低或翻转当前状态。在 PTON 位由低到高电平的变化后，PTM 输出脚初始状态为 PTOC 位所指定的电平。注意，若 PTIO1 和 PTIO0 位同时为 0 时，引脚输出不变。



比较匹配输出模式 – PTCCLR=0

- 注：1. PTCCLR=0，比较器 P 匹配将清除计数器
2. PTM 输出脚仅由 PTMAF 标志位控制
3. 在 PTON 上升沿 PTM 输出脚复位至初始值



比较匹配输出模式 - PTCCLR=1

- 注：1. PTCCLR=1，比较器 A 匹配将清除计数器
 2. PTM 输出脚仅由 PTMAF 标志位控制
 3. 在 PTON 上升沿 PTM 输出脚复位至初始值
 4. 当 PTCCLR=1 时，不会产生 PTMPF 标志

定时 / 计数器模式

为使 PTM 工作在此模式，PTMC1 寄存器的 PTM1 和 PTM0 位需要设置为“11”。定时 / 计数器模式与比较输出模式操作方式相同，并产生同样的中断请求标志。不同的是，在定时 / 计数器模式下 PTM 输出脚未使用。因此，比较匹配输出模式中的描述和时序图可以适用于此功能。该模式中未使用的 PTM 输出脚用作普通 I/O 脚或其它功能。

PWM 输出模式

为使 PTM 工作在此模式，PTMC1 寄存器中的 PTM1 和 PTM0 位需要设置为“10”，且 PTIO1 和 PTIO0 位也需要设置为“10”。PTM 的 PWM 输出功能在马达控制，加热控制，照明控制等方面十分有用。给 PTM 输出脚提供一个频率固定但占空比可调的信号，将产生一个有效值等于 DC 均方根的 AC 方波。

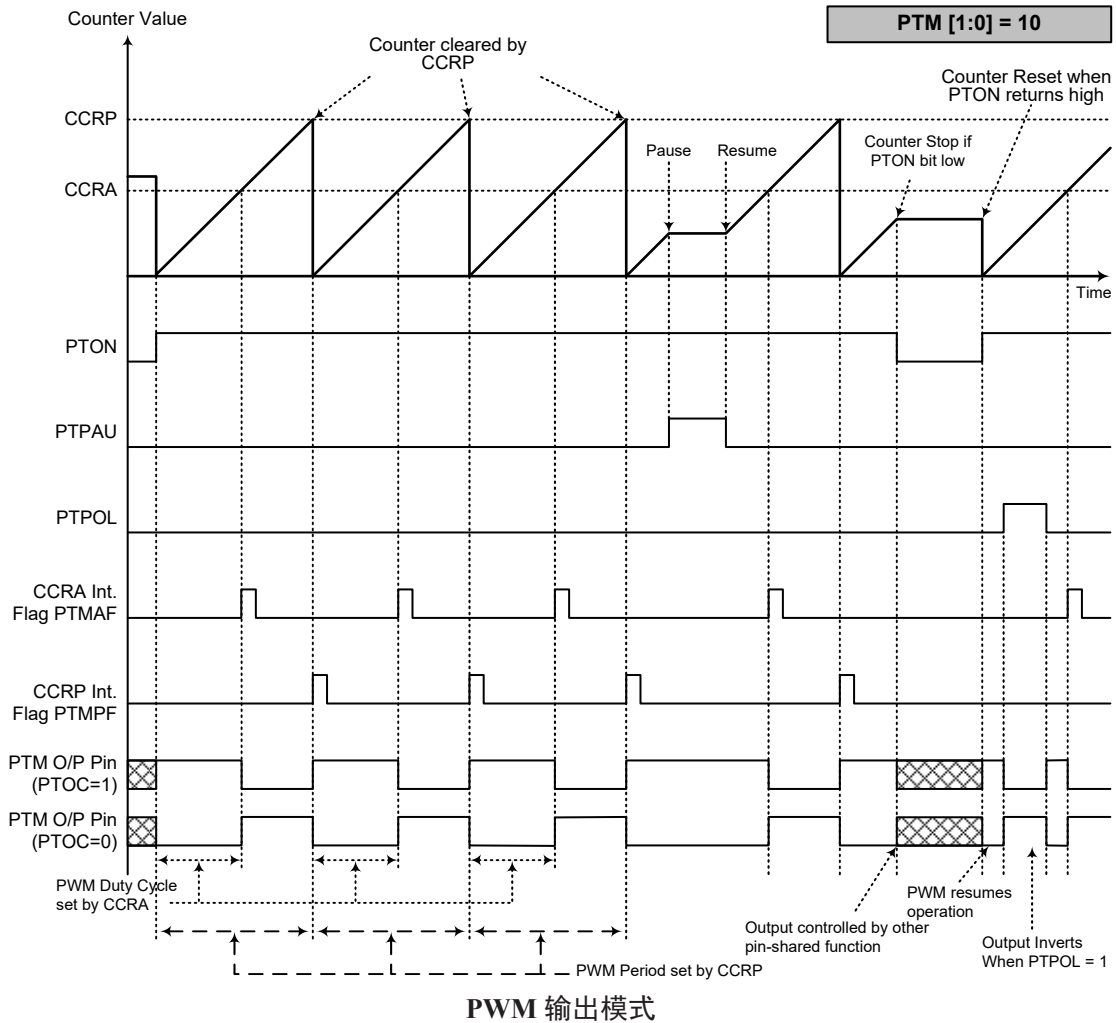
由于 PWM 波形的周期和占空比可调，其波形的选择就较为灵活。在 PWM 输出模式中，PTCCLR 位不影响 PWM 周期。CCRA 和 CCRP 寄存器决定 PWM 波形，一个用来清除内部计数器并控制 PWM 波形的频率，另一个用来控制占空比。PWM 波形的周期和占空比由 CCRP 和 CCRA 寄存器的值控制。

当比较器 A 或比较器 P 比较匹配发生时，将产生 CCRA 或 CCRP 中断标志。PTMC1 寄存器中的 PTOC 位决定 PWM 波形的极性，PTIO1 和 PTIO0 位使能 PWM 输出或将 PTM 输出脚置为逻辑高或逻辑低。PTPOL 位对 PWM 输出波形的极性取反。

● 10-bit PWM 输出模式，边沿对齐模式

| CCRP | 1~1023 | 0 |
|------|--------|------|
| 周期 | 1~1023 | 1024 |
| 占空比 | CCRA | |

若 $f_{SYS}=8\text{MHz}$ ，PTM 时钟源选择 $f_{SYS}/4$ ， $CCRP=512$ 且 $CCRA=128$ ，
PTM PWM 输出频率 = $(f_{SYS}/4)/512=f_{SYS}/2048=4\text{kHz}$ ，占空比 = $128/512=25\%$ ，
若由 CCRA 寄存器定义的占空比值等于或大于周期值，PWM 输出占空比为 100%。



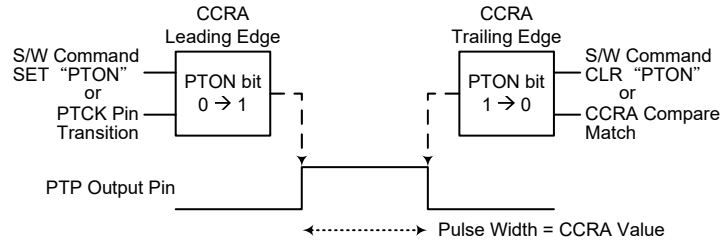
- 注：1. CCRP 清除计数器
2. 计数器清除并决定 PWM 周期
3. 当 PTIO[1:0]=00 或 01，PWM 输出功能不变
4. PTCCLR 位对 PWM 输出功能无影响

单脉冲输出模式

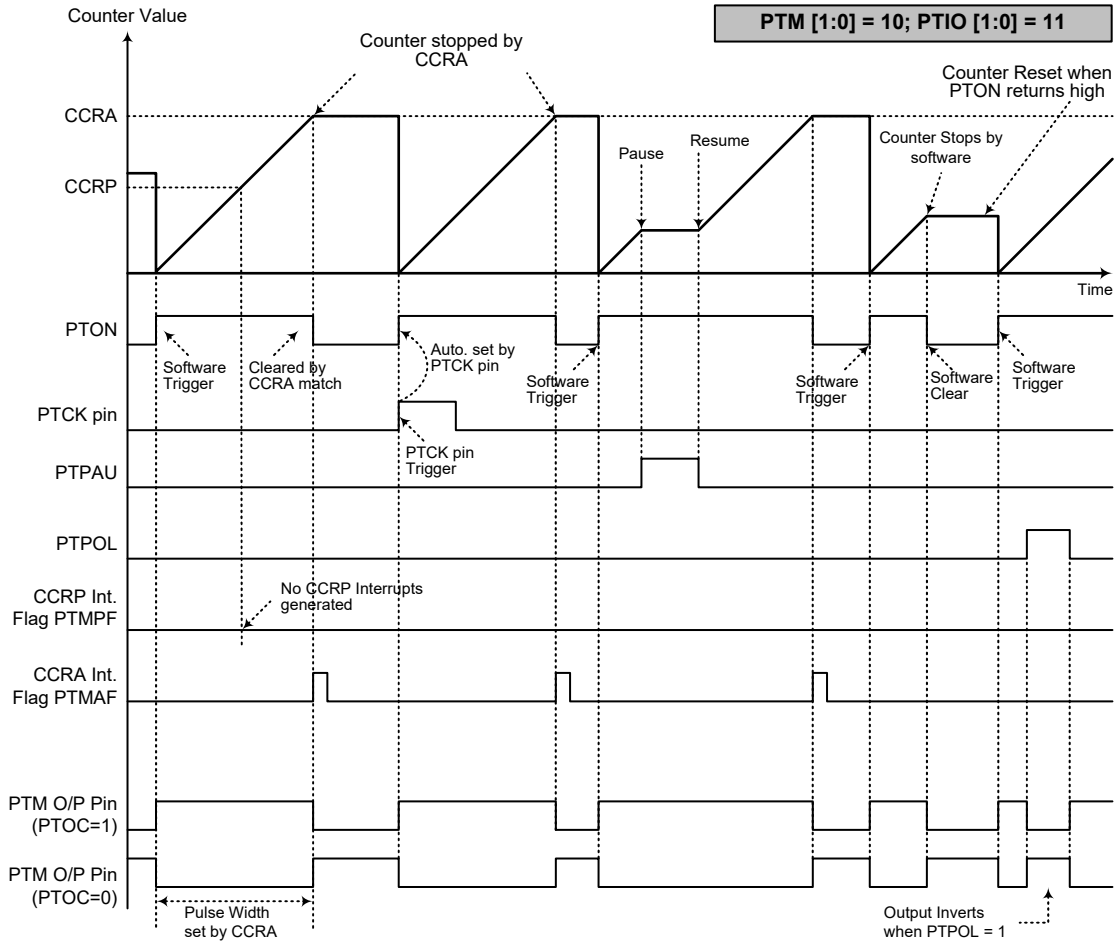
为使 TM 工作在此模式，PTMC1 寄存器中的 PTM1 和 PTM0 位需要设置为“10”，同时 PTIO1 和 PTIO0 位需要设置为“11”。正如模式名所言，单脉冲输出模式，在 PTM 输出脚将产生一个脉冲输出。

通过应用程序控制 PTON 位由低到高的转变来触发脉冲前沿输出。而处于单脉冲输出模式时，PTON 位可在 PTCK 脚发生有效边沿跳转时自动由低转变为高，进而开始单脉冲输出。当 PTON 位转变为高电平时，计数器将开始运行，并产生脉冲前沿。当脉冲有效时 PTON 位保持高电平。通过应用程序使 PTON 位清零或比较器 A 比较匹配发生时，产生脉冲后沿。

然而，比较器 A 比较匹配发生时，会自动清除 PTON 位并产生单脉冲输出边沿跳变。CCRA 的值通过这种方式控制脉冲宽度。比较器 A 比较匹配发生时，也会产生 PTM 中断。PTON 位在计数器重启时会发生由低到高的转变，此时计数器才复位至零。在单脉冲模式中，CCRP 寄存器，PTCCLR 位未使用。



单脉冲产生示意图



单脉冲输出模式

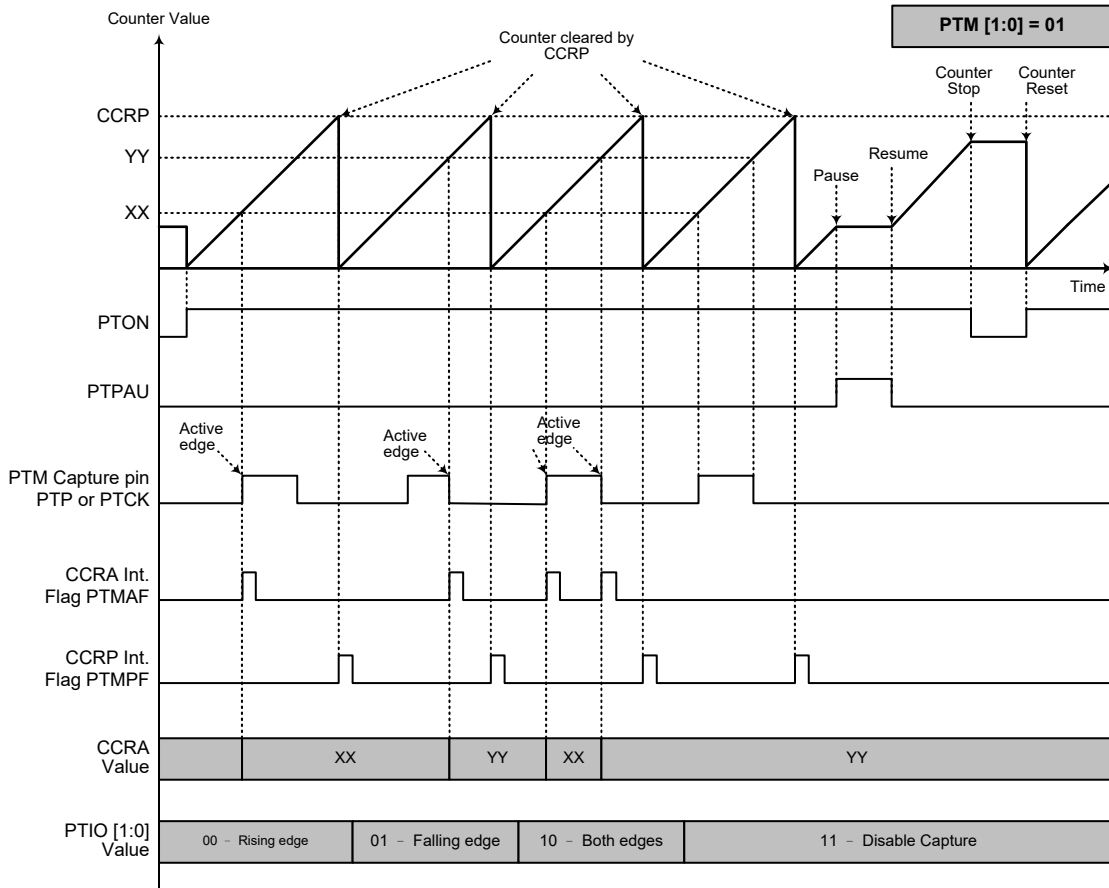
- 注：1. 通过 CCRA 匹配停止计数器
2. CCRP 未使用
3. 通过 PTCK 脚或设置 PTON 位为高来触发脉冲
4. PTCK 脚有效沿会自动设置 PTON 位为高
5. 单脉冲模式中，PTIO[1:0] 需置位“11”，且不能更改。

捕捉输入模式

为使TM工作在此模式，PTMC1寄存器的PTM1和PTM0位需要设置为“01”。此模式使能外部信号捕捉并保存内部计数器当前值，因此被用于诸如脉冲宽度测量的应用中。PTP或PTCK引脚上的外部信号，通过设置PTMC1寄存器的PTCAPTS位选择。可通过设置PTMC1寄存器的PTIO1和PTIO0位选择有效边沿类型，即上升沿，下降沿或双沿有效。通过应用程序将PTON位由低到高转变时，计数器启动。

当PTP或PTCK引脚出现有效边沿转换时，计数器当前值被锁存到CCRA寄存器，并产生PTM中断。无论PTP或PTCK引脚发生哪种边沿转换，计数器继续工作直到PTON位发生下降沿跳变。当CCRP比较匹配发生时计数器复位至零；CCRP的值通过这种方式控制计数器的最大值。当比较器P CCRP比较匹配发生时，也会产生PTM中断。记录CCRP溢出中断信号的值可以测量长脉宽。通过设置PTIO1和PTIO0位选择PTP或PTCK引脚为上升沿，下降沿或双沿有效。无论PTP或PTCK引脚发生哪种边沿转换，如果PTIO1和PTIO0位都设为高，不会产生捕捉操作，但计数器继续运行。

当PTP或PTCK引脚与其它功能共用，PTM工作在输入捕捉模式时需多加注意。这是因为如果引脚被设为输出，那么该引脚上的任何电平转变都可能执行输入捕捉操作。PTCCLR、PTOC和PTPOL位在此模式中未使用。



捕捉输入模式

- 注：1. PTM[1:0]=01 并通过 PTIO[1:0] 位设置有效边沿
2. PTM 捕捉输入脚的有效边沿将计数器的值转移到 CCRA 中
3. PTCCLR 位未使用
4. 无输出功能 – PTOC 和 PTPOL 位未使用
5. 计数器值由 CCRP 决定，在 CCRP 为“0”时，计数器计数值可达最大

A/D 转换器

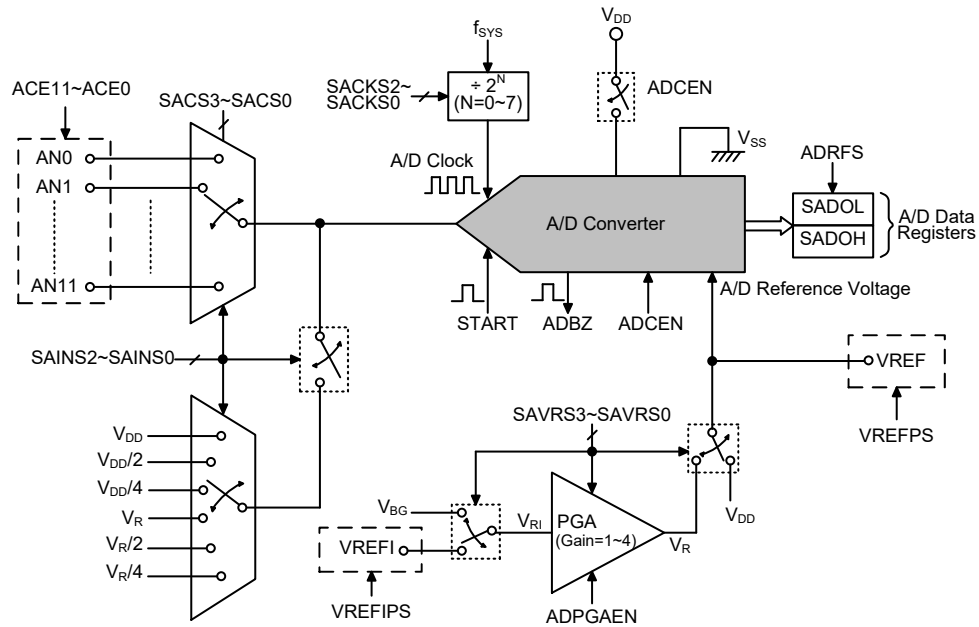
对于大多数电子系统而言，处理现实世界的模拟信号是共同的需求。为了完全由单片机来处理这些信号，首先需要通过 A/D 转换器将模拟信号转换成数字信号。将 A/D 转换器电路集成入单片机，可有效的减少外部器件，随之而来，具有降低成本和减少器件空间需求的优势。

A/D 简介

此单片机包含一个多通道的 A/D 转换器，它们可以直接接入外部模拟信号（来自传感器或其它控制信号）或内部模拟信号（如内部参考电压）并直接将这此信号转换成 12 位的数字量。选择转换外部或内部模拟信号由 SAINS 位和 SACS 位共同控制。注意，若使用 SAINS 位选择转换内部模拟信号时，已选的外部输入通道会自动断开以避免故障。关于 A/D 输入信号的详细描述请参考“A/D 转换寄存器介绍”和“A/D 输入引脚”两节内容。

| 外部输入通道 | 内部模拟信号 | A/D 信号选择 |
|--------------|---|----------------------------|
| 12: AN0~AN11 | 6: V_{DD} , $V_{DD}/2$, $V_{DD}/4$, V_R , $V_R/2$, $V_R/4$ | SAINS2~SAINS0, SACS3~SACS0 |

下图显示了 A/D 转换器内部结构和相关的寄存器。



A/D 转换器结构

A/D 转换寄存器介绍

A/D 转换器的所有工作由七个寄存器控制。一对只读寄存器来存放 12 位 A/D 转换数据的值。一个寄存器对 ACERL 和 ACERH 来配置外部模拟输入引脚功能。剩下三个控制寄存器设置 A/D 转换器的操作和控制功能。

| 寄存器名称 | 位 | | | | | | | |
|-----------------|---------|--------|---------|--------|--------|--------|--------|--------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SADOL (ADRFS=0) | D3 | D2 | D1 | D0 | — | — | — | — |
| SADOL (ADRFS=1) | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| SADOH (ADRFS=0) | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 |
| SADOH (ADRFS=1) | — | — | — | — | D11 | D10 | D9 | D8 |
| SADC0 | START | ADBZ | ADCEN | ADRFS | SACS3 | SACS2 | SACS1 | SACS0 |
| SADC1 | SAINS2 | SAINS1 | SAINS0 | — | — | SACKS2 | SACKS1 | SACKS0 |
| SADC2 | ADPGAEN | VBGEN | VREFIPS | VREFPS | SAVRS3 | SAVRS2 | SAVRS1 | SAVRS0 |
| ACERL | ACE7 | ACE6 | ACE5 | ACE4 | ACE3 | ACE2 | ACE1 | ACE0 |
| ACERH | — | — | — | — | ACE11 | ACE10 | ACE9 | ACE8 |

A/D 转换寄存器列表

A/D 转换器数据寄存器 – SADOL, SADOH

对于具有 12 位 A/D 转换器的芯片，需要两个数据寄存器存放转换结果，一个高字节寄存器 SADOH 和一个低字节寄存器 SADOL。在 A/D 转换完毕后，单片机可以直接读取这些寄存器以获得转换结果。由于寄存器只使用了 16 位中的 12 位，其数据存储格式由 SADC0 寄存器的 ADRFS 位控制，如下表所示。D0~D11 是 A/D 转换数据结果位。未使用的位读为“0”。当 A/D 转换器除能时，数据寄存器的值将保持不变。

| ADRFS | SADOH | | | | | | | | SADOL | | | | | | | |
|-------|-------|-----|----|----|-----|-----|----|----|-------|----|----|----|----|----|----|----|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | D11 | D10 | D9 | D8 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |

A/D 转换器数据寄存器

A/D 转换控制寄存器 – SADC0, SADC1, SADC2, ACERL, ACERH

寄存器 SADC0、SADC1、SADC2、ACERL 和 ACERH 用来控制 A/D 转换器的功能和操作。这些 8 位的寄存器定义包括选择连接至内部 A/D 转换器的模拟通道，数字化数据格式，A/D 时钟源，并控制和监视 A/D 转换器的开始和忙碌状态。由于此单片机只包含一个实际的模数转换电路，因此这些外部和内部模拟信号中的每一个都需要分别被发送到转换器。SADC0 寄存器中的 SACS 位用于选择哪个外部模拟输入通道被连接到内部 A/D 转换器。SADC1 寄存器中的 SAIN 位用于选择外部模拟输入通道或内部模拟信号被连接到内部 A/D 转换器。A/D 转换器还包含可编程增益放大器，PGA，用于生成 A/D 转换器内部参考电压。PGA 的总体运作是使用 SADC2 寄存器来控制的。

在 ACERL 和 ACERH 寄存器中的模拟输入引脚功能选择位用来定义 I/O 端口中的哪些引脚为 A/D 转换器的模拟输入，哪些引脚不作为 A/D 转换输入。当引脚作为 A/D 输入时，其原来的 I/O 或其它引脚共用功能消失，此外，其内部上拉电阻也将自动断开。

● **SADC0 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|------|-------|-------|-------|-------|-------|-------|
| Name | START | ADBZ | ADCEN | ADRF5 | SACS3 | SACS2 | SACS1 | SACS0 |
| R/W | R/W | R | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Bit 7 START:** 启动 A/D 转换位
0 → 1 → 0: 启动 A/D 转换
此位用于初始化 A/D 转换过程。通常此位为低，但如果设为高再被清零，将初始化 A/D 转换过程。
- Bit 6 ADBZ:** A/D 转换忙碌标志位
0: A/D 转换结束或未开始转换
1: A/D 转换中
此位用于表明 A/D 转换过程是否正在进行。当 START 位由低变为高再变为低时，ADBZ 位为高，表明 A/D 转换已初始化。A/D 转换结束后，此位被清零。
- Bit 5 ADCEN:** A/D 转换器使能控制位
0: 除能
1: 使能
此位控制 A/D 内部功能。该位被置高将使能 A/D 转换器。如果该位设为低将关闭 A/D 转换器以降低功耗。当 A/D 转换器除能时，A/D 数据寄存器 SADOH 和 SADOL 的内容将保持不变。
- Bit 4 ADRF5:** A/D 转换数据格式选择位
0: ADC 输出数据格式 → SADOH=D[11:4]; SADOL=D[3:0]
1: ADC 输出数据格式 → SADOH=D[11:8]; SADOL=D[7:0]
此位控制存放在两个 A/D 数据寄存器中的 12 位 A/D 转换结果的格式。细节方面请参考 A/D 转换数据寄存器章节。
- Bit 3~0 SACS3~SACS0:** A/D 转换器外部模拟通道输入选择位
0000: AN0
0001: AN1
0010: AN2
0011: AN3
0100: AN4
0101: AN5
0110: AN6
0111: AN7
1000: AN8
1001: AN9
1010: AN10
1011: AN11
1100~1111: 无定义，输入浮空

● **SADC1 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|--------|--------|--------|---|---|--------|--------|--------|
| Name | SAINS2 | SAINS1 | SAINS0 | — | — | SACKS2 | SACKS1 | SACKS0 |
| R/W | R/W | R/W | R/W | — | — | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | — | — | 0 | 0 | 0 |

- Bit 7~5 SAINS2~SAINS0:** A/D 输入信号选择位
000, 100: 外部信号 – 外部模拟通道输入, AN_n
001: 内部信号 – 内部 A/D 转换器电源电压 V_{DD}
010: 内部信号 – 内部 A/D 转换器电源电压 V_{DD}/2
011: 内部信号 – 内部 A/D 转换器电源电压 V_{DD}/4
101: 内部信号 – 内部参考电压 V_R

- 110: 内部信号 – 内部参考电压 $V_R/2$
 - 111: 内部信号 – 内部参考电压 $V_R/4$
- 当选择内部模拟信号时，无论 SACS 为何值，外部通道输入信号都会自动关闭。内部参考电压可通过 SADC2 寄存器中的 SAVRS3~SAVRS0 位选择不同的电压源。
- Bit 4~3 未定义，读为“0”
 - Bit 2~0 **SACKS2~SACKS0**: A/D 时钟源选择位
 - 000: f_{SYS}
 - 001: $f_{SYS}/2$
 - 010: $f_{SYS}/4$
 - 011: $f_{SYS}/8$
 - 100: $f_{SYS}/16$
 - 101: $f_{SYS}/32$
 - 110: $f_{SYS}/64$
 - 111: $f_{SYS}/128$

● **SADC2 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---------|-------|---------|--------|--------|--------|--------|--------|
| Name | ADPGAEN | VBGEN | VREFIPS | VREFPS | SAVRS3 | SAVRS2 | SAVRS1 | SAVRS0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Bit 7 **ADPGAEN**: A/D 转换器 PGA 功能使能控制位
 - 0: 除能
 - 1: 使能

该位用于控制 A/D 转换器的内部 PGA 功能以提供不同的参考电压。当该位被置高，内部参考电压 V_R 可作为 A/D 转换器的内部转换信号或参考电压。如果内部参考电压不用于 A/D 转换器，应正确配置 PGA 功能以减小功耗。
- Bit 6 **VBGEN**: 内部 Bandgap 参考电压使能控制位
 - 0: 除能
 - 1: 使能

该位用于控制 A/D 转换器的内部 Bandgap 电路的开启 / 关闭。当该位被置高，Bandgap 参考电压可用于 A/D 转换器。如果 Bandgap 参考电压不用于 A/D 转换器，且 LVD 或 LVR 功能关闭，单片机会自动关闭 Bandgap 参考电压以减小功耗。当 Bandgap 参考电压用于 A/D 转换器，在执行 A/D 转换之前需要一定的 t_{BGS} 时间以稳定 Bandgap 电路。
- Bit 5 **VREFIPS**: VREFI 引脚选择位
 - 0: 除能 – 不选择 VREFI 引脚
 - 1: 使能 – 选择 VREFI 引脚
- Bit 4 **VREFPS**: VREF 引脚选择位
 - 0: 除能 – 不选择 VREF 引脚
 - 1: 使能 – 选择 VREF 引脚
- Bit 3~0 **SAVRS3~SAVRS0**: A/D 转换器参考电压选择位
 - 0000: V_{DD}
 - 0001: V_{REFI}
 - 0010: $V_{REFI} \times 2$
 - 0011: $V_{REFI} \times 3$
 - 0100: $V_{REFI} \times 4$
 - 1001: 保留，未使用
 - 1010: $V_{BG} \times 2$
 - 1011: $V_{BG} \times 3$
 - 1100: $V_{BG} \times 4$
 - 1101~1111: V_{DD}

当 A/D 转换器参考电压选择 V_{DD} 或内部 V_{BG} 电压时，来自 VREFI 引脚的参考电压会自动关闭。

● **ACERL 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|------|------|------|------|------|------|------|
| Name | ACE7 | ACE6 | ACE5 | ACE4 | ACE3 | ACE2 | ACE1 | ACE0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Bit 7 **ACE7:** 定义 PB3 是否为 A/D 输入
0: 不是 A/D 输入
1: A/D 输入, AN7
- Bit 6 **ACE6:** 定义 PA7 是否为 A/D 输入
0: 不是 A/D 输入
1: A/D 输入, AN6
- Bit 5 **ACE5:** 定义 PA6 是否为 A/D 输入
0: 不是 A/D 输入
1: A/D 输入, AN5
- Bit 4 **ACE4:** 定义 PA5 是否为 A/D 输入
0: 不是 A/D 输入
1: A/D 输入, AN4
- Bit 3 **ACE3:** 定义 PA4 是否为 A/D 输入
0: 不是 A/D 输入
1: A/D 输入, AN3
- Bit 2 **ACE2:** 定义 PB2 是否为 A/D 输入
0: 不是 A/D 输入
1: A/D 输入, AN2
- Bit 1 **ACE1:** 定义 PB1 是否为 A/D 输入
0: 不是 A/D 输入
1: A/D 输入, AN1
- Bit 0 **ACE0:** 定义 PB0 是否为 A/D 输入
0: 不是 A/D 输入
1: A/D 输入, AN0

● **ACERH 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|-------|-------|------|------|
| Name | — | — | — | — | ACE11 | ACE10 | ACE9 | ACE8 |
| R/W | — | — | — | — | R/W | R/W | R/W | R/W |
| POR | — | — | — | — | 0 | 0 | 0 | 0 |

- Bit 7~4 未定义, 读为“0”
- Bit 3 **ACE11:** 定义 PA3 是否为 A/D 输入
0: 不是 A/D 输入
1: A/D 输入, AN11
- Bit 2 **ACE10:** 定义 PB6 是否为 A/D 输入
0: 不是 A/D 输入
1: A/D 输入, AN10
- Bit 1 **ACE9:** 定义 PB5 是否为 A/D 输入
0: 不是 A/D 输入
1: A/D 输入, AN9
- Bit 0 **ACE8:** 定义 PB4 是否为 A/D 输入
0: 不是 A/D 输入
1: A/D 输入, AN8

A/D 转换器参考电压

A/D 转换器参考电压来自正电源电压 V_{DD} 、外部参考源引脚 VREFI 或内部 Bandgap 参考源，所选的参考电压源除了来自 V_{DD} 的以外，都可通过可编程增益放大器进行放大。可通过 SADC2 寄存器中的 SAVRS3~SAVRS0 位及相关引脚功能选择位来选择。注意，所选的参考电压将会输出到 VREF 引脚，VREF 引脚与其它功能共用引脚，可通过 SADC2 寄存器中的 VREFPS 位设置。由于 VREFI 和 VREF 引脚都与其它功能共用，当选择 VREFI 或 VREF 引脚作为参考电压输入或输出引脚时，需合理设置相关引脚控制位选择 VREFI 或 VREF 引脚功能且除能其它共用引脚功能。然而，当 A/D 转换器电源电压或内部 Bandgap 参考信号被选作参考电压源时，则来自外部 VREFI 引脚的参考电压将自动被硬件关闭。在选择使用 V_{BG} 之前，应该首先使能内部 Bandgap 参考电路。为了使 Bandgap 电路变得稳定和准确，一个特定的启动时间是必要的。

注意，模拟输入信号值不能超过所选的 A/D 转换器参考电压的值。

| SAVRS[3:0] | 参考电压源 | 说明 |
|-----------------|------------|------------------------------------|
| 0000, 1101~1111 | V_{DD} | 内部 A/D 转换器电源电压 V_{DD} |
| 0001 | VREFI 引脚 | 外部 A/D 转换器参考电压 V_{REFI} |
| 0010 | | 外部 A/D 转换器参考电压 $V_{REFI} \times 2$ |
| 0011 | | 外部 A/D 转换器参考电压 $V_{REFI} \times 3$ |
| 0100 | | 外部 A/D 转换器参考电压 $V_{REFI} \times 4$ |
| 1010 | Bandgap 电路 | 内部 Bandgap 电压 $V_{BG} \times 2$ |
| 1011 | | 内部 Bandgap 电压 $V_{BG} \times 3$ |
| 1100 | | 内部 Bandgap 电压 $V_{BG} \times 4$ |

A/D 转换器参考电压选择

A/D 转换器输入信号

所有的 A/D 模拟输入引脚都与 I/O 引脚及其它功能共用。使用 ACERL 和 ACERH 寄存器中的相应引脚共用功能选择位，可以将它们设置为 A/D 转换器模拟输入脚或具有其它功能。如果对应的引脚作为 A/D 转换输入，那么它原来的引脚功能将除能。通过这种方式，引脚的功能可由程序来控制，灵活地切换引脚功能。如果将引脚设为 A/D 输入，则通过寄存器编程设置的所有上拉电阻会自动断开。请注意，端口控制寄存器不需要为使能 A/D 输入而先设定为输入模式，当 A/D 输入功能选择位使能 A/D 输入时，端口控制寄存器的状态将被重置。

在 SADC1 寄存器中，SAINS2~SAINS0 位被用来确定要转换的来自外部通道输入或内部模拟信号。SADC0 寄存器中的 SACS3~SACS0 位被用来决定选择哪个外部通道输入被转换。如果 SAINS2~SAINS0 位被设置为“000”或“100”，那么将选择外部通道输入进行转换，而 SACS 位可以决定选择哪个外部通道被转换。当 SAINS 字段被设置为“x01”、“x10”或“x11”的值时，将选择一个内部模拟信号进行转换。内部模拟信号来自 A/D 转换器电源 V_{DD} 或内部参考电压 V_R 、 $1/2 V_R$ 、 $1/4 V_R$ 。如果选择要转换内部模拟信号，外部通道信号输入将自动关闭，无论 SACS 位段为何值。

| SAINS[2:0] | SACS[3:0] | 输入信号 | 描述 |
|------------|-----------|--------------------|--------------------|
| 000, 100 | 0000~1011 | AN0~AN11 | 外部模拟通道输入 ANn |
| | 11xx | — | 输入浮空, 未选择外部通道 |
| 001 | xxxx | V _{DD} | 内部 A/D 转换器电源电压 |
| 010 | xxxx | V _{DD} /2 | 内部 A/D 转换器电源电压 / 2 |
| 011 | xxxx | V _{DD} /4 | 内部 A/D 转换器电源电压 / 4 |
| 101 | xxxx | V _R | 内部参考电压 |
| 110 | xxxx | V _R /2 | 内部参考电压 / 2 |
| 111 | xxxx | V _R /4 | 内部参考电压 / 4 |

“x”：任意值

A/D 转换器输入信号选择

A/D 操作

SADC0 寄存器中的 START 位, 用于打开 A/D 转换器。当单片机设置此位从逻辑低到逻辑高, 然后再到逻辑低, 就会开始一个模数转换周期。

SADC0 寄存器中的 ADBZ 位用于表明模数转换过程是否正在进行。A/D 转换成功启动后, ADBZ 位会被单片机自动置为 1。在转换周期结束后, ADBZ 位会自动置为 0。此外, 也会置位中断控制寄存器内相应的 A/D 中断请求标志位, 如果中断使能, 就会产生对应的内部中断信号。A/D 内部中断信号将引导程序跳转到相应的 A/D 内部中断地址。如果 A/D 内部中断被禁止, 可以让单片机轮询 SADC0 寄存器中的 ADBZ 位, 检查此位是否被清除, 作为另一种侦测 A/D 转换周期结束的方法。

A/D 转换器的时钟源为系统时钟 f_{sys} 或其分频, 而分频系数由 SADC1 寄存器中的 SACKS2~SACKS0 位决定。虽然 A/D 时钟源是由系统时钟 f_{sys} 和 SACKS2~SACKS0 位决定, 但可选择的 A/D 时钟源则有一些限制。由于允许的 A/D 时钟周期 t_{ADCK} 的范围为 0.5μs~10μs, 所以选择系统时钟速度时必须小心。如果系统时钟速度为 8MHz 时, SACKS2~SACKS0 位不能设为“000”、“001”或“111”。必须保证设置的 A/D 转换时钟周期不小于时钟周期的最小值或大于时钟周期的最大值, 否则将会产生不准确的 A/D 转换值。使用者可以参考下面的表格, 被标上星号 * 的数值是不允许的, 因为它们超出了 A/D 转换时钟周期规定的范围。

| f _{sys} | A/D 时钟周期 (t _{ADCK}) | | | | | | | |
|------------------|--------------------------------------|--|--|--|---|---|---|--|
| | SACKS[2:0] = 000 (f _{sys}) | SACKS[2:0] = 001 (f _{sys} /2) | SACKS[2:0] = 010 (f _{sys} /4) | SACKS[2:0] = 011 (f _{sys} /8) | SACKS[2:0] = 100 (f _{sys} /16) | SACKS[2:0] = 101 (f _{sys} /32) | SACKS[2:0] = 110 (f _{sys} /64) | SACKS[2:0] = 111 (f _{sys} /128) |
| 1MHz | 1μs | 2μs | 4μs | 8μs | 16μs* | 32μs* | 64μs* | 128μs* |
| 2MHz | 500ns | 1μs | 2μs | 4μs | 8μs | 16μs* | 32μs* | 64μs* |
| 4MHz | 250ns* | 500ns | 1μs | 2μs | 4μs | 8μs | 16μs* | 32μs* |
| 8MHz | 125ns* | 250ns* | 500ns | 1μs | 2μs | 4μs | 8μs | 16μs* |
| 12MHz | 83ns* | 167ns* | 333ns* | 667ns | 1.33μs | 2.67μs | 5.33μs | 10.67μs* |
| 16MHz | 62.5ns* | 125ns* | 250ns* | 500ns | 1μs | 2μs | 4μs | 8μs |

A/D 时钟周期范例

SADC0 寄存器中的 ADCEN 位用于控制 A/D 转换电路电源的开启和关闭。该位必须置高以开启 A/D 转换器电源。当设置 ADCEN 位为高开启 A/D 转换器内部电路时, 在 A/D 转换成功开启前需一段延时。即使选择无引脚作为 A/D 输入, 如果 ADCEN 设为“1”, 那么仍然会产生功耗。因此在功耗敏感的应用中, 当

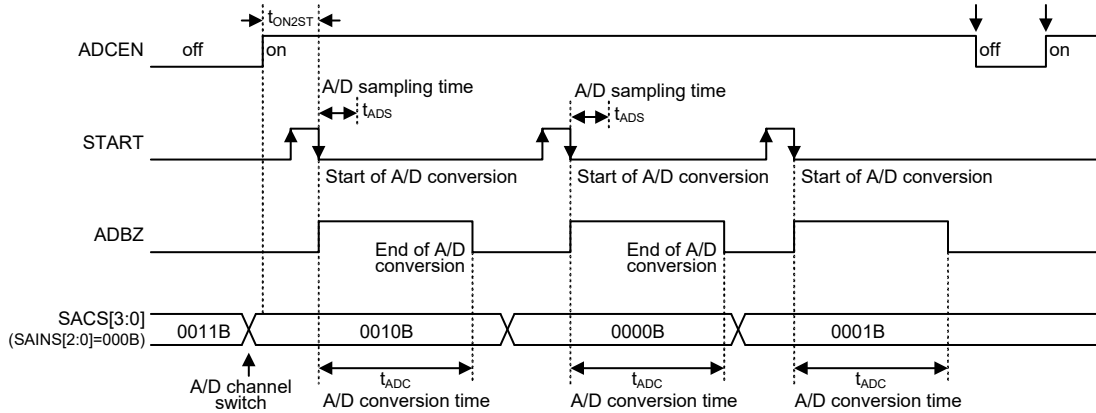
未使用 A/D 转换器功能时，建议设置 ADCEN 为低以减少功耗。

A/D 转换率及时序图

一个完整的 A/D 转换包含两部分，数据采样和数据转换。数据采样时间定义为 t_{ADS} ，需要 4 个 A/D 时钟周期，而数据转换需要 12 个 A/D 时钟周期。所以一个完整的 A/D 转换时间， t_{ADC} ，一共需要 16 个 A/D 时钟周期。

$$\text{最大 A/D 转换率} = \text{A/D 时钟周期} / 16 \quad (1)$$

下列时序图表示模数转换过程中不同阶段的图形与时序。由应用程序控制开始 A/D 转换过程后，单片机的内部硬件就会开始进行转换，在这个过程中，程序可以继续其它功能。A/D 转换时间为 $16t_{ADCK}$ ， t_{ADCK} 为 A/D 时钟周期。



A/D 转换时序图 - 外部通道输入

A/D 转换步骤

下面概述实现 A/D 转换过程的各个步骤。

- 步骤 1
通过 SADC1 寄存器中的 SACKS2~SACKS0 位，选择所需的 A/D 转换时钟。
- 步骤 2
将 SADC0 寄存器中的 ADCEN 位置高使能 A/D 转换器。
- 步骤 3
通过 SADC1 寄存器中的 SAINS 和 SACS 位段，选择连接至内部 A/D 转换器的信号。
若选择外部通道输入，接着执行步骤 4。
若选择内部模拟信号，接着执行步骤 5。
- 步骤 4
若已通过 SAINS 位选择 A/D 输入信号来自外部通道输入，接着应设置相关的引脚共用功能控制位将该引脚规划为 A/D 输入引脚。通过设置 SACS3~SACS0 位选择哪个外部通道接至 A/D 转换器。接着执行步骤 6。
- 步骤 5
若已通过 SAINS2~SAINS0 位选择 A/D 输入信号来自内部模拟信号，无论 SACS3~SACS0 为何值，外部通道输入都会自动断开。接着执行步骤 6。
- 步骤 6
通过在 SADC2 寄存器中配置 SAVRS3~SAVRS0 位来选择参考电压源。如果选择了 PGA 输出电压作为 A/D 转换器的参考电压，则需使能 PGA。如果

PGA 输入信号来自外部的 VREF1 引脚，则应该首先合理设置相应的引脚功能选择位。如果 A/D 转换器电源电压、内部 Bandgap 电路电压或其放大后电压被选择，则外部输入引脚 VREF1 的参考电压将自动关闭。

- 步骤 7
在 SADC0 寄存器中设置 ADRFS 位选择 A/D 转换器输出数据格式。
 - 步骤 8
如果要使用中断，则中断控制寄存器需要正确地设置，以确保 A/D 中断功能是激活的。总中断控制位 EMI 需要置位为“1”，以及 A/D 转换器中断位 ADE 也需要置位为“1”。
 - 步骤 9
现在可以通过设置 SADC0 寄存器中的 START 位从“0”到“1”再回到“0”，开始模数转换的过程。
 - 步骤 10
如果 A/D 转换正在进行中，ADBZ 位会被置为逻辑高。A/D 转换完成后，ADBZ 位会被置为逻辑低，并可从 SADOH 和 SADOL 寄存器中读取输出数据。如果 ADC 中断使能，并且堆栈没有满，则可以通过中断服务程序获得数据。
- 注：若使用轮询 SADC0 寄存器中 ADBZ 位的状态的方法来检查转换过程是否结束时，则中断使能的步骤可以省略。

编程注意事项

在编程时，如果 A/D 转换器未使用，通过设置 SADC0 寄存器中的 ADCEN 为低，关闭 A/D 内部电路以减少电源功耗。此时，不考虑输入脚的模拟电压，内部 A/D 转换器电路不产生功耗。如果 A/D 转换器输入脚用作普通 I/O 脚，必须特别注意，输入电压为无效逻辑电平也可能增加功耗。

A/D 转换功能

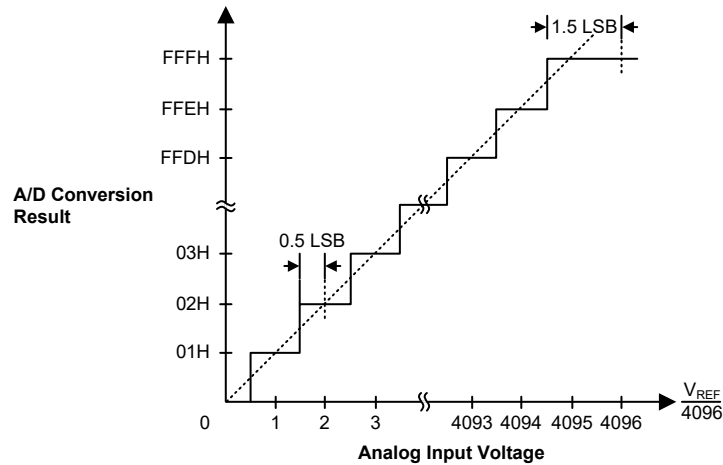
该单片机含有一组 12 位的 A/D 转换器，它们转换的最大值可达 FFFH。由于模拟输入最大值等于 V_{REF} 的电压值，因此每一位可表示 $V_{REF}/4096$ 的模拟输入值。

$$1 \text{ LSB} = V_{REF} \div 4096$$

通过下面的等式可估算 A/D 转换器输入电压值：

$$\text{A/D 输入电压} = \text{A/D 数字输出值} \times V_{REF} \div 4096$$

下图显示 A/D 转换器模拟输入值和数字输出值之间理想的转换功能。除了数字化数值 0，其后的数字化数值会在精确点之前的 0.5 LSB 处改变，而数字化数值的最大值将在 V_{REF} 之前的 1.5 LSB 处改变。注意， V_{REF} 电压是由 SAVRS 字段确定的实际的 A/D 转换器参考电压。



理想的 A/D 转换功能

A/D 转换应用范例

下面两个范例程序用来说明怎样使用 A/D 转换。第一个范例是轮询 SADC0 寄存器中的 ADBZ 位来判断 A/D 转换是否完成；第二个范例则使用中断的方式判断。

范例 1：使用查询 ADBZ 的方式来检测转换结束

```

clr ADE                ; disable ADC interrupt
mov a,03H              ; select fsys/8 as A/D clock and A/D input signal
                        ; comes from external channel

mov SADC1,a
mov a,00H              ; select VDD as A/D reference voltage source
mov SADC2,a
mov a,01h             ; setup ACERL to configure pin AN0
mov ACERL,a
mov a,20h             ; enable A/D converter and select AN0 external
                        ; channel input

mov SADC0,a
:
start_conversion:
clr START              ; high pulse on start bit to initiate conversion
set START              ; reset A/D converter
clr START              ; start A/D conversion
polling_EOC:
sz ADBZ                ; poll the SADC0 register ADBZ bit to detect end
                        ; of A/D conversion
                        ; continue polling
jmp polling_EOC
mov a,SADOL            ; read low byte conversion result value
mov SADOL_buffer,a    ; save result to user defined register
mov a,SADOH            ; read high byte conversion result value
mov SADOH_buffer,a    ; save result to user defined register
:
jmp start_conversion  ; start next A/D conversion
    
```

范例 2：使用中断的方式来检测转换结束

```

clr ADE                ; disable ADC interrupt
mov a,03H              ; select fsys/8 as A/D clock and A/D input signal
                        ; comes from external channel

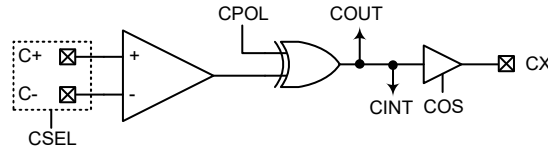
mov SADC1,a
mov a,00H              ; select VDD as A/D reference voltage source
mov SADC2,a
mov a,01h              ; setup ACERL to configure pin AN0
mov ACERL,a
mov a,20h              ; enable A/D converter and select AN0 external
                        ; channel input

mov SADC0,a
:
Start_conversion:
clr START              ; high pulse on START bit to initiate conversion
set START              ; reset A/D converter
clr START              ; start A/D conversion
clr ADF                ; clear ADC interrupt request flag
set ADE                ; enable ADC interrupt
set EMI                ; enable global interrupt
:
:
; ADC interrupt service routine
ADC_ISR:
mov acc_stack,a       ; save ACC to user defined memory
mov a,STATUS
mov status_stack,a    ; save STATUS to user defined memory
:
:
mov a,SADOL            ; read low byte conversion result value
mov SADOL_buffer,a    ; save result to user defined register
mov a,SADOH            ; read high byte conversion result value
mov SADOH_buffer,a    ; save result to user defined register
:
:
EXIT_INT_ISR:
mov a,status_stack
mov STATUS,a          ; restore STATUS from user defined memory
mov a,acc_stack       ; restore ACC from user defined memory
reti

```

比较器

该单片机包含一个模拟比较器。该比较器具有暂停、极性选择、迟滞等功能，可通过寄存器进行灵活配置。比较器的引脚与普通 I/O 引脚共用，当比较器功能未使用时，此引脚可做普通引脚使用而不浪费 I/O 资源。



比较器

比较器操作

该单片机包含一个比较器功能，用于比较两个模拟电压，基于它们的差值上提供一个输出。控制寄存器 CPC 可控制该内部比较器。比较器的输出可由寄存器的一位记录，并且在共用的 I/O 引脚上输出。此外，比较器功能有输出极性、迟滞功能和暂停控制。

当比较器使能时，连接到与比较器共用的输入引脚的上拉电阻将自动失效。当比较器输入接近其切换电压时，由于输入信号上升或下降速度较慢，比较器输出端可能会产生一些伪输出信号。通过选择迟滞功能提供少量正反馈给比较器可使此种情况的发生率降低。理想情况下正负输入信号在同一个电压点时比较器将发生开关动作，但是不可避免的输入失调电压会导致情况不确定。若迟滞功能使能，可将切换偏差值大大减小。

• CPC 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|-----|------|------|-----|--------|--------|-------|
| Name | CSEL | CEN | CPOL | COUT | COS | CMPEG1 | CMPEG0 | CHYEN |
| R/W | R/W | R/W | R/W | R | R/W | R/W | R/W | R/W |
| POR | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Bit7 **CSEL**: 比较器引脚或 I/O 引脚选择位

0: 输入 / 输出引脚

1: 比较器输入引脚 C+ 和 C-

此位为比较器输入引脚或输入 / 输出引脚选择位。为“1”时，比较器输入引脚使能。此时，引脚的输入 / 输出引脚功能失效，与比较器共用引脚的上拉电阻配置选项将自动失效。

Bit 6 **CEN**: 比较器开 / 关控制位

0: 关闭

1: 开启

此位为比较器开 / 关控制位。为“0”时，比较器关闭，即使其输入引脚上加有模拟电压也不会产生功耗。对功耗要求严格的应用中，当比较器未使用或单片机进入休眠或空闲模式之前，此位应清零。

Bit 5 **CPOL**: 比较器输出极性控制位

0: 同相

1: 反相

此位控制比较器输出极性。为“0”时，COUT 位反映比较器同相输出；为“1”时，COUT 位反映比较器反相输出。

| | |
|---------|---|
| Bit 4 | <p>COUT: 比较器输出位</p> <p>CPOL=0 0: $C+ < C-$ 1: $C+ > C-$</p> <p>CPOL=1 0: $C+ > C-$ 1: $C+ < C-$</p> <p>此位为比较器输出位。此位的极性由比较器输入电压和 CPOL 位的状态决定。</p> |
| Bit3 | <p>COS: 比较器输出路径选择位</p> <p>0: CX 引脚 (比较器输出到 CX 脚) 1: 输入 / 输出引脚 (比较器输出仅内部使用)</p> |
| Bit 2~1 | <p>CMPEG1~CMPEG0: 比较器输出中断触发边沿选择位</p> <p>00: 上升沿 → 如果 COUT 状态从 0 变为 1, 产生比较器中断触发信号 01: 下降沿 → 如果 COUT 状态从 1 变为 0, 产生比较器中断触发信号 1x: 双沿 → 如果 COUT 状态从 0 变为 1 或从 1 变为 0, 产生比较器中断触发信号</p> |
| Bit 0 | <p>CHYEN: 比较器迟滞功能控制位</p> <p>0: 关闭 1: 开启</p> <p>此位为迟滞控制位。为“1”时, 比较器有一定量迟滞, 具体见比较器电气特性表。滞后产生的正反馈将减少比较器门槛附近的伪开关效应的影响。</p> |

比较器中断

比较器具有中断功能。当输出状态改变时, 相应的中断标志将会置位, 若相关中断使能位被置位, 系统将跳转至相应的中断向量中执行。注意, 触发比较器产生中断的条件是 COUT 位状态的改变, 而非比较器输出引脚状态的改变。单片机处于休眠或空闲模式且比较器使能时, 若外部输入引脚导致比较器输出状态发生改变, 则由此产生的中断标志位也可产生一个唤醒动作。若要除能唤醒功能, 进入休眠或空闲模式前中断标志位应先置为高。

编程注意事项

若比较器使能, 当单片机进入休眠或空闲模式时其仍保持有效并会有一些的耗电, 用户可考虑在进入休眠或空闲模式前先关闭比较器。

由于比较器引脚与普通输入 / 输出脚共用, 若比较器功能使能时, 这些引脚的输入 / 输出寄存器将读为“0”(端口控制寄存器为“1”)或读为端口数据寄存器的值(端口控制寄存器为“0”)。

串行接口模块 – SIM

该单片机内有一个串行接口模块，包括两种易与外部设备通信的串行接口：四线 SPI 或两线 I²C 接口。这两种接口具有相当简单的通信协议，单片机可以通过这些接口与传感器、闪存或 EEPROM 内存等硬件设备通信。SIM 接口的引脚与 I/O 引脚共用，所以要使用 SIM 功能时应先通过 IFS 寄存器的引脚功能选择位和 SIMC0 寄存器中相关控制位 SIMEN 位选择 SIM 引脚功能。因为这两种接口共用引脚和寄存器，所以要通过一个 SIMC0 寄存器中的 SIM2~SIM0 位来选择 SPI 还是 I²C 通信接口。若 SIM 功能使能并且对应的引脚被用作 SIM 输入引脚，除了 SCS 引脚外，可通过上拉电阻控制寄存器选择与输入 / 输出口共用的 SIM 脚的上拉电阻。注意，如果 SCS 引脚功能使能，相应的内部上拉电阻将断开连接，不能使用上拉控制寄存器来控制。

SPI 接口

SPI 接口常用于与外部设备如传感器、闪存或 EEPROM 内存等通信。四线 SPI 接口最初是由摩托罗拉公司研制，是一个有相当简单的通信协议的串行数据接口，这个协议可以简化与外部硬件的编程要求。

SPI 通信模式为全双工模式，且能以主 / 从模式的工作方式进行通信，单片机既可以做为主机，也可以做为从机。虽然 SPI 接口理论上允许一个主机控制多个从机，但此处的 SPI 中只有一个片选信号引脚 SCS。若主机需要控制多个从机，可使用输入 / 输出引脚选择从机。

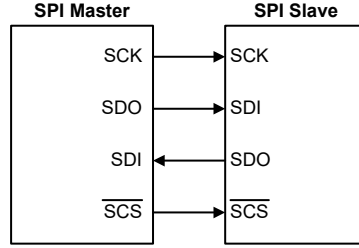
SPI 接口操作

SPI 接口是一个全双工串行数据传输器。SPI 接口的四线为：SDI、SDO、SCK 和 SCS。SDI 和 SDO 是数据的输入和输出线。SCK 是串行时钟线，SCS 是从机的选择线。SPI 的接口引脚与普通 I/O 口和 I²C 的功能脚共用。通过设定 SIMC0/SIMC2 寄存器的对应位和相关引脚重置控制位，来使能 SPI 接口引脚功能。SPI 可以通过 SIMC0 寄存器中的 SIMEN 位来除能或使能。连接到 SPI 接口的单片机以从主 / 从模式进行通信，且主机完成所有的数据传输初始化，并控制时钟信号。由于单片机只有一个 SCS 引脚，所以只能拥有一个从机设备。可通过软件控制 SCS 引脚使能与除能，设置 CSEN 位为“1”使能 SCS 功能，设置 CSEN 位为“0”，SCS 引脚将处于浮空状态。

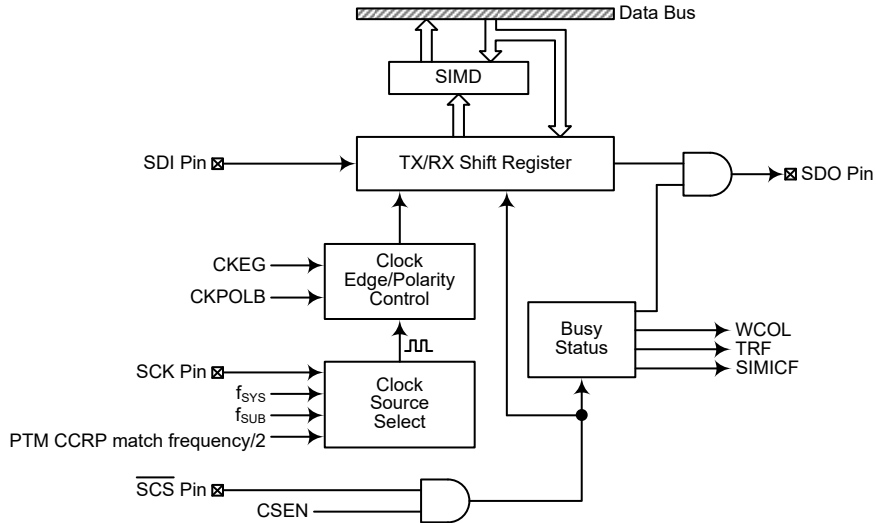
该单片机的 SPI 功能具有以下特点：

- 全双工同步数据传输
- 主从模式
- 最低有效位先传或最高有效位先传的数据传输模式
- 传输完成标志位
- 时钟源上升沿或下降沿有效

SPI 接口状态受很多因素的影响，如单片机处于主机或从机的工作模式和 CSEN，SIMEN 位的状态。



SPI 主 / 从机连接方式



SPI 方框图

SPI 寄存器

有三个内部寄存器用于控制 SPI 接口的所有操作，其中有一个数据寄存器 SIMD、两个控制寄存器 SIMC0 和 SIMC2。注意，SIMC1 寄存器仅用于 I²C 接口。

| 寄存器名称 | 位 | | | | | | | |
|-------|------|------|--------|------|---------|---------|-------|--------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SIMC0 | SIM2 | SIM1 | SIM0 | - | SIMDEB1 | SIMDEB0 | SIMEN | SIMICF |
| SIMC2 | D7 | D6 | CKPOLB | CKEG | MLS | CSEN | WCOL | TRF |
| SIMD | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |

SPI 寄存器列表

SIMD 数据寄存器

SIMD 寄存器用于存储发送和接收的数据。这个寄存器由 SPI 和 I²C 功能所共用。在单片机尚未将数据写入到 SPI 总线中时，要传输的数据应先存在 SIMD 中。SPI 总线接收到数据之后，单片机就可以从 SIMD 数据寄存器中读取。所有通过 SPI 传输或接收的数据都必须通过 SIMD 实现。

● **SIMD 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| Name | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | x | x | x | x | x | x | x | x |

“x”：未知

Bit 7~0 **D7~D0**: SIM 数据寄存器 bit 7 ~ bit 0

SIMD 控制寄存器

单片机中也有两个控制 SPI 接口功能的寄存器，SIMC0 和 SIMC2。应注意的是 SIMC2 与 I²C 接口功能中的寄存器 SIMA 是同一个寄存器。SPI 功能不会用到寄存器 SIMC1，SIMC1 只适用于 I²C。寄存器 SIMC0 用于控制使能 / 除能功能和设置数据传输的时钟频率。寄存器 SIMC2 用于其它的控制功能如 LSB/MSB 选择，写冲突标志位等。

● **SIMC0 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|------|------|---|---------|---------|-------|--------|
| Name | SIM2 | SIM1 | SIM0 | — | SIMDEB1 | SIMDEB0 | SIMEN | SIMICF |
| R/W | R/W | R/W | R/W | — | R/W | R/W | R/W | R/W |
| POR | 1 | 1 | 1 | — | 0 | 0 | 0 | 0 |

Bit 7~5 **SIM2~SIM0**: SIM 工作模式控制位

- 000: SPI 主机模式；SPI 时钟为 $f_{SYS}/4$
- 001: SPI 主机模式；SPI 时钟为 $f_{SYS}/16$
- 010: SPI 主机模式；SPI 时钟为 $f_{SYS}/64$
- 011: SPI 主机模式；SPI 时钟为 f_{SUB}
- 100: SPI 主机模式；SPI 时钟为 PTM CCRP 匹配频率 / 2
- 101: SPI 从机模式
- 110: I²C 从机模式
- 111: 未定义

这几位用于设置 SIM 功能的工作模式，用于选择 SPI 的主从模式和 SPI 的主机时钟频率及 I²C 或 SPI 功能。SPI 时钟源可来自于系统时钟也可以选择来自 PTM 和 f_{SUB} 。若选择的是作为 SPI 从机，则其时钟源从外部主机而得。

Bit 4 未定义，读为“0”

Bit 3~2 **SIMDEB1~SIMDEB0**: I²C 去抖时间选择位

这些位只有在 SIM 设置成 I²C 接口模式时才有效。请参考 I²C 寄存器部分。

Bit 1 **SIMEN**: SIM 控制位

- 0: 除能
- 1: 使能

此位为 SIM 接口的开 / 关控制位。此位为“0”时，SIM 接口除能，SDI、SDO、SCK 和 \overline{SCS} 或 SDA 和 SCL 脚将失去 SPI 或 I²C 功能，SIM 工作电流减小到最小值。此位为“1”时，SIM 接口使能。若 SIM 经由 SIM2~SIM0 位设置为工作在 SPI 接口，当 SIMEN 位由低到高转变时，SPI 控制寄存器中的设置不会发生变化，其首先应在应用程序中初始化。若 SIM 经由 SIM2~SIM0 位设置为工作在 I²C 接口，当 SIMEN 位由低到高转变时，I²C 控制寄存器中的设置，如 HTX 和 TXAK，将不会发生变化，其首先应在应用程序中初始化，此时相关 I²C 标志，如 HCF、HAAS、HBB、SRW 和 RXAK，将被设置为其默认状态。

Bit 0 **SIMICF**: SIM 未完成标志位

- 0: 未发生
- 1: 发生

此位仅当 SIM 配置在 SPI 从机模式时有效。如果 SPI 工作在从机模式且 SIMEN

和 CSEN 位都为“1”，但在 SPI 数据传输完全结束前 \overline{SCS} 线被外部主机拉高，SIMICF 和 TRF 位都会被置高。在这种情况下，如果相应的中断功能使能将产生一个中断。然而，如果 SIMICF 位是由软件应用程序设为 1，那么 TRF 位将不会置高。

● **SIMC2 寄存器**

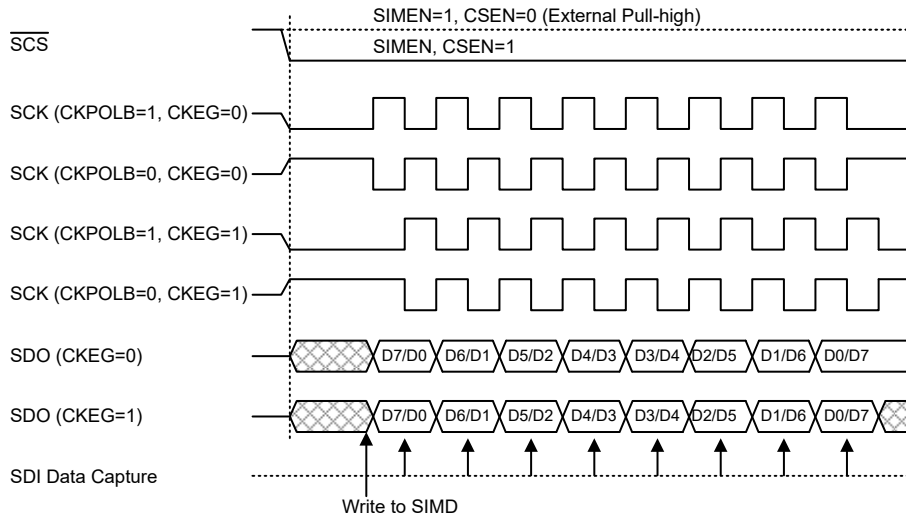
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|-----|--------|------|-----|------|------|-----|
| Name | D7 | D6 | CKPOLB | CKEG | MLS | CSEN | WCOL | TRF |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Bit 7~6 未定义位
用户可通过软件程序对这两位进行读写。
- Bit 5 **CKPOLB**: SPI 时钟线的基础状态位
0: 当时钟无效时, SCK 口为高电平
1: 当时钟无效时, SCK 口为低电平
此位决定了时钟线的基础状态, 当时钟无效时, 若此位为高, SCK 为低电平, 若此位为低, SCK 为高电平。
- Bit 4 **CKEG**: SPI 的 SCK 有效时钟边沿类型位
CKPOLB=0
0: SCK 为高电平且在 SCK 上升沿抓取数据
1: SCK 为高电平且在 SCK 下降沿抓取数据
CKPOLB=1
0: SCK 为低电平且在 SCK 下降沿抓取数据
1: SCK 为低电平且在 SCK 上升沿抓取数据
CKEG 和 CKPOLB 位用于设置 SPI 总线上时钟信号输入和输出方式。在执行数据传输前, 这两位必须被设置, 否则将产生错误的时钟边沿信号。CKPOLB 位决定时钟线的基本状态, 若时钟无效且此位为高, 则 SCK 为低电平, 若时钟无效且此位为低, 则 SCK 为高电平。CKEG 位决定有效时钟边沿类型, 取决于 CKPOLB 的状态。
- Bit 3 **MLS**: SPI 数据移位顺序位
0: LSB 优先
1: MSB 优先
数据移位选择位, 用于选择数据传输时高位优先传输还是低位优先传输。此位设置为高时高位优先传输, 为低时低位优先传输。
- Bit 2 **CSEN**: SPI \overline{SCS} 引脚控制位
0: 除能
1: 使能
CSEN 位用于 \overline{SCS} 引脚的使能 / 除能控制。此位为低时, \overline{SCS} 除能并处于浮空状态。此位为高时, \overline{SCS} 使能并作为选择脚。
- Bit 1 **WCOL**: SPI 写冲突标志位
0: 无冲突
1: 冲突
WCOL 标志位用于监测数据冲突的发生。此位为高时, 数据在传输时被写入 SIMD 寄存器。若数据正在被传输时, 此操作无效。此位可被应用程序清零。
- Bit 0 **TRF**: SPI 发送 / 接收结束标志位
0: 数据正在发送
1: 数据发送结束
TRF 位为发送 / 接收结束标志位, 当 SPI 数据传输结束时, 此位自动置为高, 但需通过应用程序设置为“0”。此位也可用于产生中断。

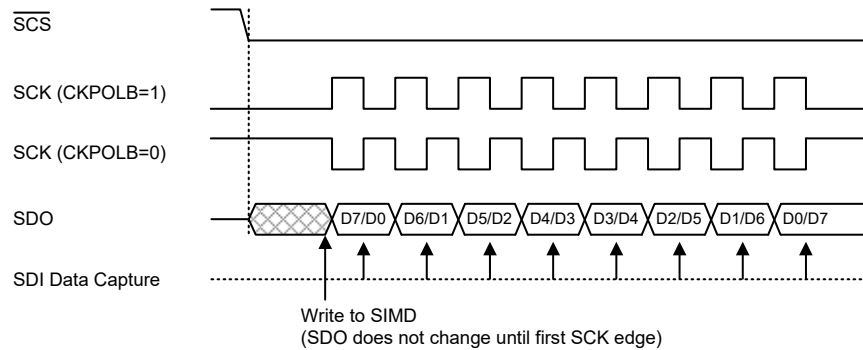
SPI 通信

将 SIMEN 设置为高，使能 SPI 功能之后，单片机处于主机模式，当数据写入到寄存器 SIMD 的同时传输 / 接收开始进行。数据传输完成时，TRF 位将自动被置位但清除只能通过应用程序完成。单片机处于从机模式时，收到主机发来的信号之后，会传输 SIMD 中的数据，而且在 SDI 引脚上的数据也会被移位到 SIMD 寄存器中。主机应在输出时钟信号之前先输出一个 SCS 信号以使能从机，从机的数据传输功能也应在与 SCK 信号相关的适当时候准备就绪，这由 CKPOLB 和 CKEG 位决定。所附时序图表明了了在 CKPOLB 和 CKEG 位各种设置情况下从机数据与 SCK 信号的关系。

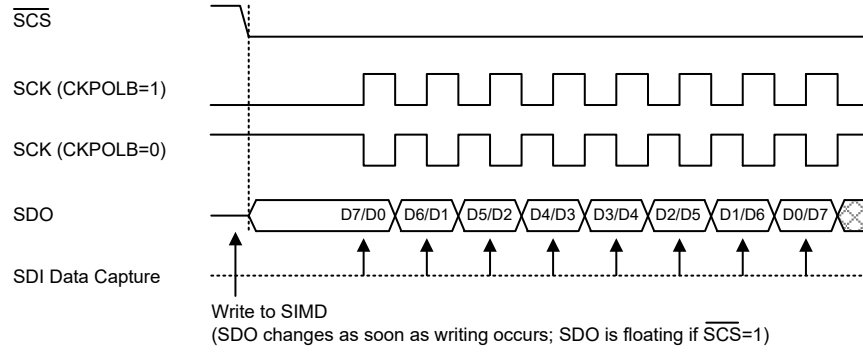
即使在单片机处于空闲模式，如果所选的 SPI 时钟源有效，SPI 主机功能仍将继续执行。



SPI 主机模式时序

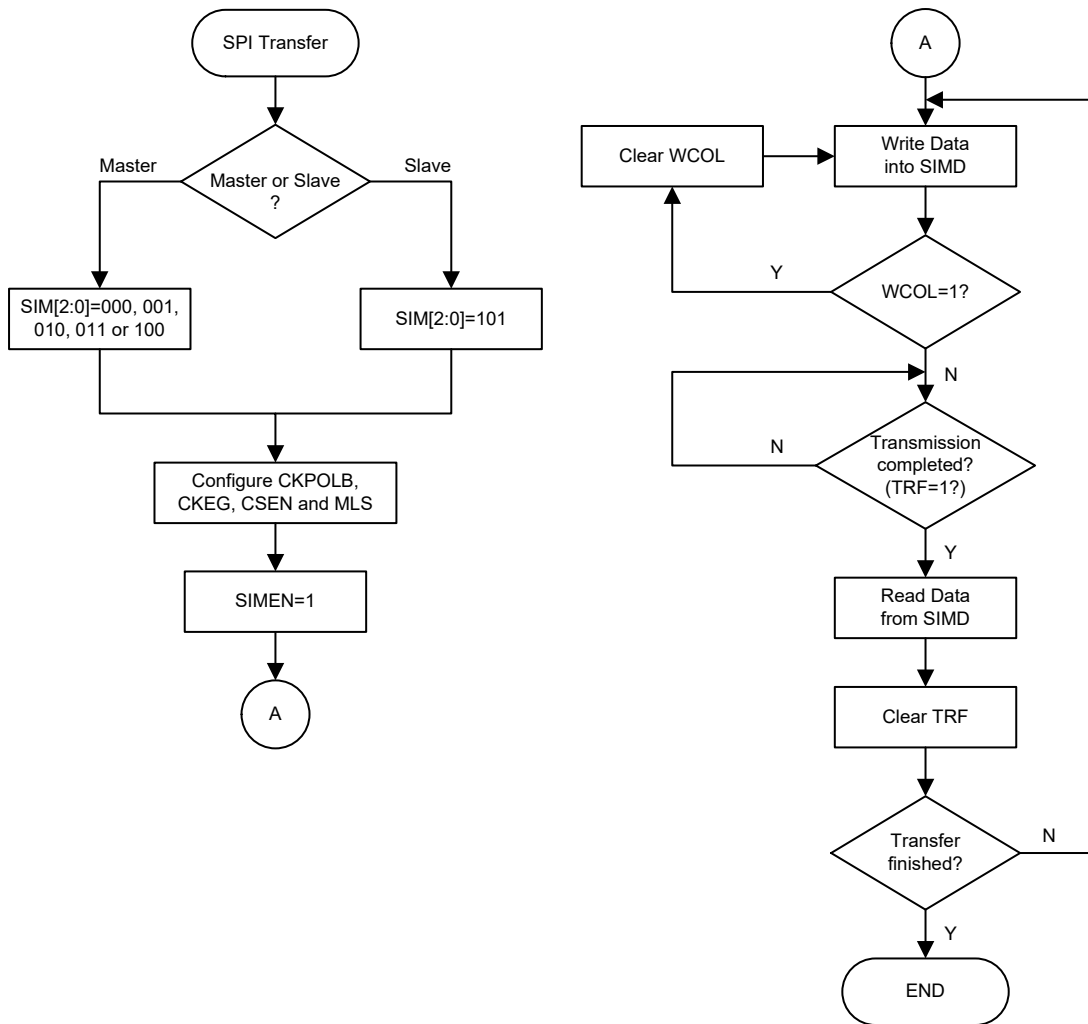


SPI 从机模式时序 - CKEG=0



Note: For SPI slave mode, if SIMEN=1 and CSEN=0, SPI is always enabled and ignores the \overline{SCS} level.

SPI 从机模式时序 - CKEG=1



SPI 传输控制流程图

SPI 使能 / 除能

设置 $\overline{\text{CS}}=1$ 、 $\overline{\text{SCS}}=0$ 将使能 SPI 总线，然后等待写数据到 SIMD 寄存器 (TXRX 缓存器)。单片机处于主机模式，数据写入 SIMD 寄存器后，自动开始数据传输或接收操作。数据传输完成时，TRF 位将自动被置位。单片机处于从机模式，SCK 引脚上收到脉冲信号之后，会传输 TXRX 中的数据，或将 SDI 引脚上的数据移入。

当 SPI 总线除能时，通过设置相应控制位，SCK、SDI、SDO、 $\overline{\text{SCS}}$ 可作为 I/O 口或其它功能引脚使用。

SPI 操作步骤

四线制 SPI 接口可完成所有主 / 从模式通信工作。

在 SIMC2 寄存器中，CSEN 位控制 SPI 接口的所有功能。设置此位为高， $\overline{\text{SCS}}$ 信号线有效将使能 SPI 接口。设置此位为低，SPI 接口将除能， $\overline{\text{SCS}}$ 信号线处于浮空状态因此不能控制 SPI 接口。CSEN 位和 SIMC0 寄存器中的 SIMEN 位设置为高，使得 SDI 信号线处于浮空状态且 SDO 信号线为高电平。主机模式中，如果 SCK 信号线为高还是低取决于 SIMC2 寄存器中的时钟极性选择位 CKPOLB。从机模式中，SCK 信号线处于浮空状态。如果 SIMEN 位设置为低，SPI 接口被除能，通过设置相应引脚共用控制位， $\overline{\text{SCS}}$ 、SDI、SDO 和 SCK 可作为 I/O 口或其它功能引脚使用。主机模式中，当数据被写入 SIMD 寄存器后，主机完成所有的数据传输初始化，并控制时钟信号。从机模式中，由外部主机发出数据传送 / 接收时钟信号。下面介绍主从模式中数据传输步骤。

主机模式

- 步骤 1
设置 SIMC0 控制寄存器中的 SIM2~SIM0 位，选择 SPI 主机模式和时钟源。
- 步骤 2
设置 CSEN 和 MLS 位，选择高位或低位数据优先传送，这必须与从机设备一致。
- 步骤 3
设置 SIMC0 控制寄存器中的 SIMEN 位，使能 SPI 接口功能。
- 步骤 4
对于写操作：写数据到 SIMD 寄存器，实际上此时数据会被存储在 TXRX 缓存器中。再使用 SCK 和 $\overline{\text{SCS}}$ 信号线将数据输出。跳至步骤 5。
对于读操作：从 SDI 信号线移入的数据将被存储在 TXRX 缓存器中，直到所有数据接收完毕，此时数据全部锁存至 SIMD 寄存器。
- 步骤 5
检测 WCOL 位，若此位为高，则发生数据冲突并跳回至步骤 4；若为低，则继续执行下面的步骤。
- 步骤 6
检测 TRF 位或等待 SIM SPI 串行总线中断发生。
- 步骤 7
从 SIMD 寄存器中读数据。
- 步骤 8
清除 TRF。
- 步骤 9
跳回至步骤 4。

从机模式

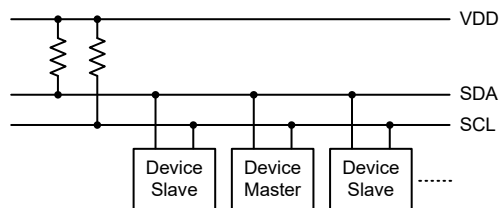
- 步骤 1
设置 SIMC0 控制寄存器中的 SIM2~SIM0 位，选择 SPI 从机模式。
- 步骤 2
设置 CSEN 和 MLS 位，选择高位或低位数据优先传送，这必须与主机设备一致。
- 步骤 3
设置 SIMC0 控制寄存器中的 SIMEN 位，使能 SPI 接口功能。
- 步骤 4
对于写操作：写数据到 SIMD 寄存器，实际上此时数据会被存储在 TXRX 缓存器中。等待主机时钟 SCK 信号和 \overline{SCS} 信号。跳至步骤 5。
对于读操作：从 SDI 信号线移入的数据将被存储在 TXRX 缓存器中，直到所有数据接收完毕，此时数据全部锁存至 SIMD 寄存器。
- 步骤 5
检测 WCOL 位，若此位为高，则发生数据冲突并跳回至步骤 4；若为低，则继续执行下面的步骤。
- 步骤 6
检测 TRF 位或等待 SIM SPI 串行总线中断发生。
- 步骤 7
从 SIMD 寄存器中读数据。
- 步骤 8
清除 TRF。
- 步骤 9
跳回至步骤 4。

错误侦测

SIMC2 寄存器中的 WCOL 位用于数据传输期间监测数据冲突的发生。此位由 SPI 串行接口设置为高，而由应用程序来清除为零。在数据传输期间如果写数据到 SIMD，此位被置高提示数据冲突发生，并阻止数据继续被写入。

I²C 接口

I²C 可以和传感器、EEPROM 内存等外部硬件接口进行通信。最初是由飞利浦公司研制，是适用于同步串行数据传输的双线式低速串行接口。I²C 接口具有两线通信，非常简单的通信协议和在同一总线上和多个设备进行通信的能力的优点，使之在很多的场合中大受欢迎。

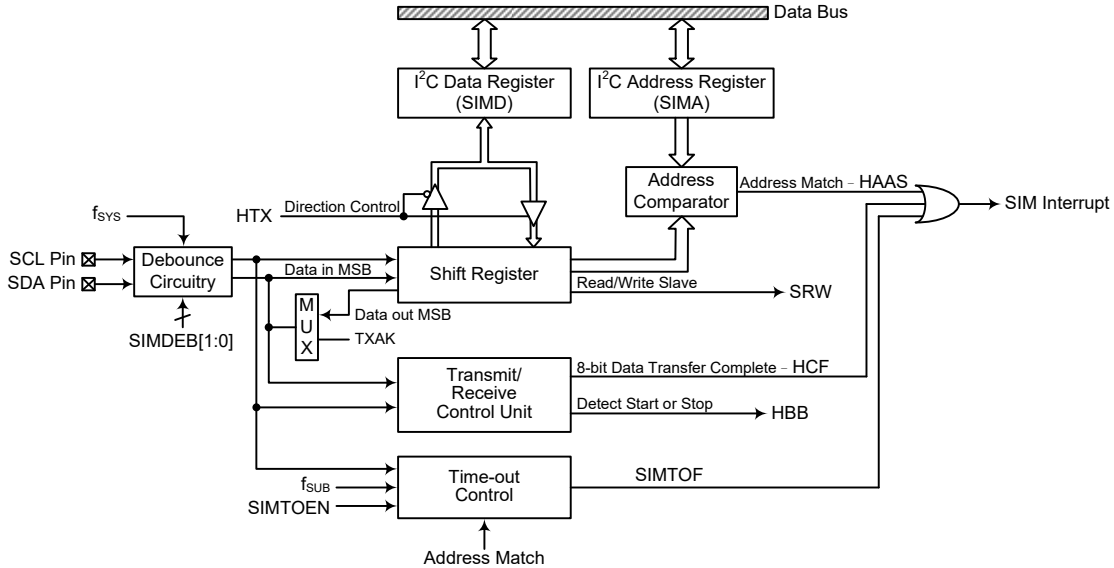


I²C 主从总线连接图

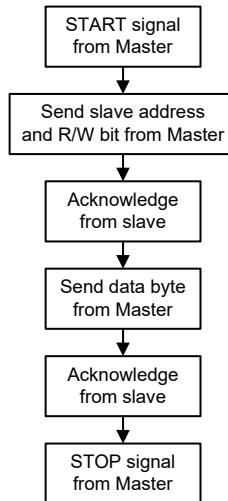
I²C 接口操作

I²C 串行接口是一个双线的接口，有一条串行数据线 SDA 和一条串行时钟线 SCL。由于可能有多个设备在同一条总线上相互连接，所以这些设备的输出都是开漏型输出。因此应在这些输出上都应加上拉电阻。应注意的是，I²C 总线上的每个设备都没有选择线，但分别与唯一的地址一一对应，用于 I²C 通信。

如果有两个设备通过双向的 I²C 总线进行通信，那么就存在一个主机和一个从机。主机和从机都可以用于发送和接收数据，但只有主机才可以控制总线动作。那些处于从机模式的设备，要在 I²C 总线上传输数据只有两种方式，一是从机发送模式，二是从机接收模式。



I²C 方框图



I²C 接口操作

SIMDEB1 和 SIMDEB0 位决定 I²C 接口的去抖时间。这个功能可以使用内部时钟在外部时钟上增加一个去抖间隔，会减小时钟线上毛刺发生的可能性，以避免单片机发生误动作。如果选择了这个功能，去抖时间可以选择 2 个或 4 个系

统时钟。为了达到需要的 I²C 数据传输速度，系统时钟 f_{SYS} 和 I²C 去抖时间之间存在一定的关系。I²C 标准模式或者快速模式下，用户需注意所选的系统时钟频率与标准匹配去抖时间的设置，其具体关系如下表所示。

| I ² C 去抖时间选择 | I ² C 标准模式 (100kHz) | I ² C 快速模式 (400kHz) |
|-------------------------|--------------------------------|--------------------------------|
| 无去抖时间 | $f_{SYS} > 2\text{MHz}$ | $f_{SYS} > 5\text{MHz}$ |
| 2 个系统时钟去抖时间 | $f_{SYS} > 4\text{MHz}$ | $f_{SYS} > 10\text{MHz}$ |
| 4 个系统时钟去抖时间 | $f_{SYS} > 8\text{MHz}$ | $f_{SYS} > 20\text{MHz}$ |

I²C 最小 f_{SYS} 频率要求

I²C 寄存器

I²C 总线有三个控制寄存器 SIMC0、SIMC1 和 SIMTOC，及一个从机地址寄存器 SIMA 和一个数据寄存器 SIMD。应注意的是 SIMA 寄存器也有另外一个名字，SIMC2，使用 SPI 功能时会用到。I²C 接口会用到寄存器 SIMC0 中的 SIMEN 位和 SIM2~SIM0 位。

| 寄存器名称 | 位 | | | | | | | |
|--------|---------|--------|---------|---------|---------|---------|---------|---------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SIMC0 | SIM2 | SIM1 | SIM0 | — | SIMDEB1 | SIMDEB0 | SIMEN | SIMICF |
| SIMC1 | HCF | HAAS | HBB | HTX | TXAK | SRW | IAMWU | RXAK |
| SIMD | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| SIMA | SIMA6 | SIMA5 | SIMA4 | SIMA3 | SIMA2 | SIMA1 | SIMA0 | D0 |
| SIMTOC | SIMTOEN | SIMTOF | SIMTOS5 | SIMTOS4 | SIMTOS3 | SIMTOS2 | SIMTOS1 | SIMTOS0 |

I²C 寄存器列表

I²C 数据寄存器

SIMD 用于存储发送和接收的数据。这个寄存器由 SPI 和 I²C 功能所共用。在单片机将数据写入到 I²C 总线之前，要传输的数据应先存在 SIMD 中。I²C 总线接收到数据之后，单片机就可以从 SIMD 数据寄存器中读取。所有通过 I²C 传输或接收的数据都必须通过 SIMD 实现。

• SIMD 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|-----|-----|-----|-----|-----|-----|-----|
| Name | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | x | x | x | x | x | x | x | x |

“x”：未知

Bit 7~0 **D7~D0**: SIM 数据寄存器 bit 7 ~ bit 0

I²C 地址寄存器

SIMA 寄存器也在 SPI 接口功能中使用，但其名称改为 SIMC2。SIMA 寄存器用于存放 7 位从机地址，寄存器 SIMA 中的 bit 7 ~ bit 1 是单片机的从机地址，bit 0 未定义。

如果接至 I²C 的主机发送出的地址和寄存器 SIMA 中存储的地址相符，那么就选中了这个从机。应注意的是寄存器 SIMA 和 SPI 接口使用的寄存器 SIMC2 共用同一个寄存器地址。

● SIMA 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|-------|-------|-------|-------|-------|-------|-----|
| Name | SIMA6 | SIMA5 | SIMA4 | SIMA3 | SIMA2 | SIMA1 | SIMA0 | D0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Bit 7~1 **SIMA6~SIMA0:** I²C 从机地址位
SIMA6~SIMA0 是 I²C 从机地址 bit 6 ~ bit 0
- Bit 0 **D0:** 保留位，此位可通过软件程序进行读写。

I²C 控制寄存器

单片机中有三个控制 I²C 接口功能的寄存器，SIMC0、SIMC1 和 SIMTOC。寄存器 SIMC0 用于控制使能 / 除能功能和设置数据传输的时钟频率。寄存器 SIMC1 包括多个用于表明 I²C 传输状态的相关标志位。SIMTOC 寄存器用于控制 I²C 总线超时功能，在 I²C 超时控制章节有描述。

● SIMC0 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|------|------|---|---------|---------|-------|--------|
| Name | SIM2 | SIM1 | SIM0 | — | SIMDEB1 | SIMDEB0 | SIMEN | SIMICF |
| R/W | R/W | R/W | R/W | — | R/W | R/W | R/W | R/W |
| POR | 1 | 1 | 1 | — | 0 | 0 | 0 | 0 |

- Bit 7~5 **SIM2~SIM0:** SIM 工作模式控制位
000: SPI 主机模式; SPI 时钟为 $f_{sys}/4$
001: SPI 主机模式; SPI 时钟为 $f_{sys}/16$
010: SPI 主机模式; SPI 时钟为 $f_{sys}/64$
011: SPI 主机模式; SPI 时钟为 f_{sub}
100: SPI 主机模式; SPI 时钟为 PTM CCRP 匹配频率 / 2
101: SPI 从机模式
110: I²C 从机模式
111: 未定义
这几位用于设置 SIM 功能的工作模式，用于选择 SPI 的主从模式和 SPI 的主机时钟频率及 I²C 或 SPI 功能。SPI 时钟源可来自于系统时钟也可以选择来自 PTM 和 f_{sub} 。若选择的是作为 SPI 从机，则其时钟源从外部主机而得。
- Bit 4 未定义，读为“0”
- Bit 3~2 **SIMDEB1~SIMDEB0:** I²C 去抖时间选择位
00: 无去抖时间
01: 2 个系统时钟去抖时间
1x: 4 个系统时钟去抖时间
当设置 SIM2~SIM0 位为“110”将 SIM 设置为 I²C 接口功能时，这两个位用于选择 I²C 去抖时间。
- Bit 1 **SIMEN:** SIM 控制位
0: 除能
1: 使能
此位为 SIM 接口的开 / 关控制位。此位为“0”时，SIM 接口除能，SDI、SDO、SCK 和 SCS 或 SDA 和 SCL 脚将失去 SPI 或 I²C 功能，SIM 工作电流减小到最小值。此位为“1”时，SIM 接口使能。若 SIM 经由 SIM2~SIM0 位设置为工作在 SPI 接口，当 SIMEN 位由低到高转变时，SPI 控制寄存器中的设置不会发生变化，其首先应在应用程序中初始化。若 SIM 经由 SIM2~SIM0 位设置为工作在 I²C 接口，当 SIMEN 位由低到高转变时，I²C 控制寄存器中的设置，如 HTX 和 TXAK，将不会发生变化，其首先应在应用程序中初始化，此时相关 I²C 标志，如 HCF、HAAS、HBB、SRW 和 RXAK，将被设置为其默认状态。

Bit 0 **SIMICF**: SIM 未完成标志位
此位仅当 SIM 配置在 SPI 从机模式时有效。请参考 SPI 寄存器部分。

● **SIMC1 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|------|-----|-----|------|-----|-------|------|
| Name | HCF | HAAS | HBB | HTX | TXAK | SRW | IAMWU | RXAK |
| R/W | R | R | R | R/W | R/W | R/W | R/W | R |
| POR | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Bit 7 **HCF**: I²C 总线数据传输结束标志位
0: 数据正在被传输
1: 8 位数据传输完成
数据正在传输时该位为低。当 8 位数据传输完成时，此位为高并产生一个中断。

Bit 6 **HAAS**: I²C 总线地址匹配标志位
0: 地址不匹配
1: 地址匹配
此标志位用于决定从机地址是否与主机发送地址相同。若地址匹配此位为高，否则此位为低。

Bit 5 **HBB**: I²C 总线忙标志位
0: I²C 总线闲
1: I²C 总线忙
当检测到 START 信号时 I²C 忙，此位变为高。当检测到 STOP 信号时 I²C 总线停止，该位变为低。

Bit 4 **HTX**: I²C 从机处于发送或接收模式选择位
0: 从机处于接收模式
1: 从机处于发送模式

Bit 3 **TXAK**: I²C 总线发送确认标志位
0: 从机发送确认标志
1: 从机没有发送确认标志
单片机接收 8 位数据之后会将该位在第九个时钟传到总线上。如果单片机想要接收更多的数据，则应在接收数据之前将此位设置为“0”。

Bit 2 **SRW**: I²C 从机读 / 写标志位
0: 从机应处于接收模式
1: 从机应处于发送模式
SRW 位是 I²C 从机读写标志位。决定主机是否希望发送或接收来自 I²C 总线的的数据。当传输地址和从机的地址相同时，HAAS 位会被设置为高，从机将检测 SRW 位来决定进入发送模式还是接收模式。如果 SRW 位为高时，主机会请求从总线上读数据，此时从机处于发送模式。当 SRW 位为低时，主机往总线上写数据，从机处于接收模式以读取该数据。

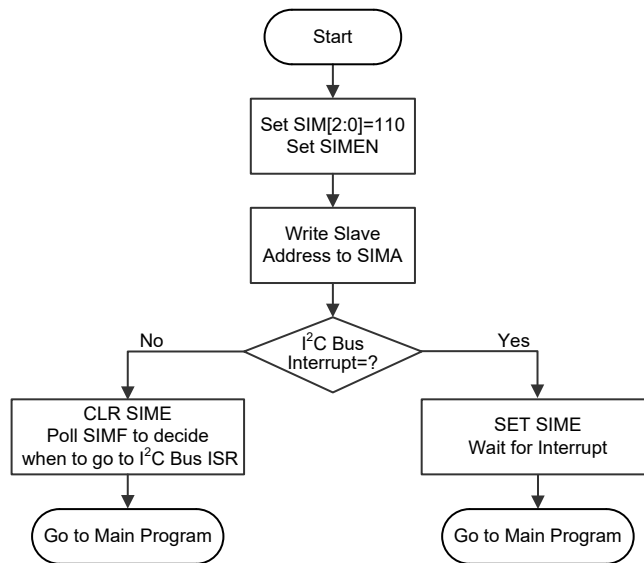
Bit 1 **IAMWU**: I²C 地址匹配唤醒控制位
0: 除能
1: 使能
此位应设置为“1”使能 I²C 地址匹配以使系统从休眠或空闲模式中唤醒。若进入休眠或空闲模式前 IAMWU 已经置高以使能 I²C 地址匹配唤醒功能，在系统唤醒后须应用程序清除此位以确保单片机正确地运行。

Bit 0 **RXAK**: I²C 总线接收确认标志位
0: 从机接收到确认标志
1: 从机没有接收到确认标志
RXAK 位是接收确认标志位。如果 RXAK 位为“0”即 8 位数据传输之后，从机在第九个时钟有接受到一个正确的确认位。如果从机处于发送状态，从机会检查 RXAK 位来判断主机是否愿意继续接收下一个字节。因此直到 RXAK 为“1”时，从机传输方停止发送数据。这时，从机传输方将释放 SDA 线，主机发出停止信号释放 I²C 总线。

I²C 总线通信

I²C 总线上的通信需要四步完成，一个起始信号，一个从机地址发送，一个数据传输，还有一个停止信号。当起始信号被写入 I²C 总线时，总线上的所有从机都会接收到这个起始信号并且被通知总线上会即将有数据到达。数据的前 7 位是从机地址，高位在前，低位在后。如果发出的地址和从机地址匹配，SIMC1 寄存器的 HAAS 位会被置位，同时产生 SIM 中断。进入中断服务程序后，系统要检测 HAAS 位和 SIMTOF 位，以判断 I²C 总线中断是来自从机地址匹配，还是来自 8 位数据传输完毕，或是来自 I²C 总线超时。在数据传输中，注意的是，在 7 位从机地址被发送后，接下来的一位，即第 8 位，是读 / 写控制位，该位的值会反映到 SRW 位中。从机通过检测 SRW 位以确定主控制器是要进入发送模式还是接收模式。在 I²C 总线开始传送数据前，需要先初始化 I²C 总线，初始化 I²C 总线步骤如下：

- 步骤 1
设置 SIMC0 寄存器中 SIM2~SIM0 位为“110”和 SIMEN 位为“1”，以使能 I²C 总线。
- 步骤 2
向 I²C 总线地址寄存器 SIMA 写入从机地址。
- 步骤 3
设置中断控制寄存器中的 SIME 中断使能位，以使能 SIM 中断。



I²C 总线初始化流程图

I²C 总线起始信号

起始信号只能由连接 I²C 总线的主机产生，而不是由从机产生。总线上的所有从机都可以侦测到起始信号。如果有从机侦测到起始信号，则表明 I²C 总线处于忙碌状态，并会置位 HBB。起始信号是指在 SCL 为高电平时，SDA 线上发生从高到低的电平变化。

I²C 从机地址

I²C 总线上的所有从机都会侦测由主机发出的起始信号。发送起始信号后，紧接着主机会发送从机地址以选择要进行数据传输的从机。所有在 I²C 总线上的从机接收到 7 位地址数据后，都会将其与各自内部的地址进行比较。如果从机从主机上接收到的地址与自身内部的地址相匹配，则会产生一个 SIM I²C 总线中断信号。地址位接下来的一位为读/写状态位（即第 8 位），将被保存到 SIMC1 寄存器的 SRW 位，随后发出一个低电平应答信号（即第 9 位）。当单片机从机的地址匹配时，会将状态标志位 HAAS 置位。

SIM I²C 总线有三个中断源，当程序运行至中断服务子程序时，通过检测 HAAS 位和 SIMTOF 位，以判断 I²C 总线中断是来自从机地址匹配，还是来自 8 位数据传递完毕，或是来自 I²C 总线超时。当是从机地址匹配发生中断时，则从机或是用于发送模式并将数据写进 SIMD 寄存器，或是用于接收模式并从 SIMD 寄存器中读取空值以释放 SCL 线。

I²C 总线读/写信号

SIMC1 寄存器的 SRW 位用来表示主机是要从 I²C 总线上读取数据还是要将数据写到 I²C 总线上。从机则通过检测该位以确定自己是作为发送方还是接收方。当 SRW 置“1”，表示主机要从 I²C 总线上读取数据，从机则作为发送方，将数据写到 I²C 总线；当 SRW 清“0”，表示主机要写数据到 I²C 总线上，从机则做为接收方，从 I²C 总线上读取数据。

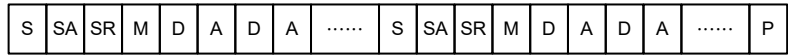
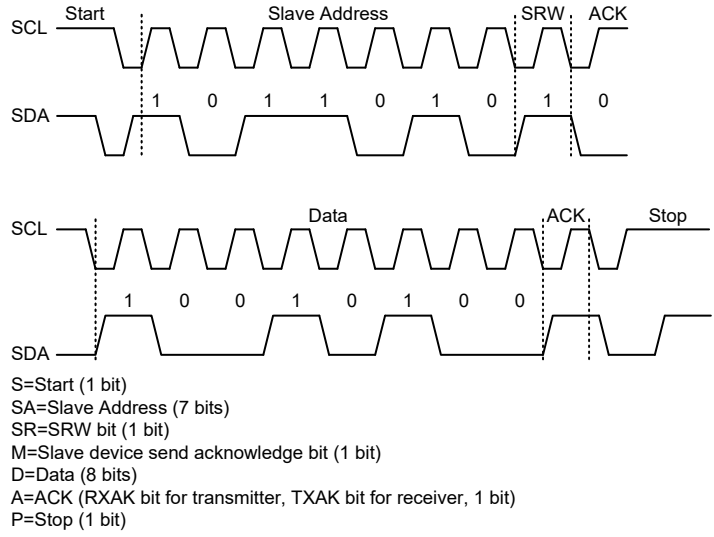
I²C 总线从机地址确认信号

主机发送呼叫地址后，当 I²C 总线上的任何从机内部地址与其匹配时，会发送一个应答信号。此应答信号会通知主机有从机已经接收到了呼叫地址。如果主机没有收到应答信号，则主机必须发送停止 (STOP) 信号以结束通信。当 HAAS 为高时，表示从机接收到的地址与自己内部地址匹配，则从机需检查 SRW 位，以确定自己是作为发送方还是作为接收方。如果 SRW 位为高，从机须设置成发送方，这样会置位 SIMC1 寄存器的 HTX 位。如果 SRW 位为低，从机须设置成接收方，这样会清零 SIMC1 寄存器的 HTX 位。

I²C 总线数据和确认信号

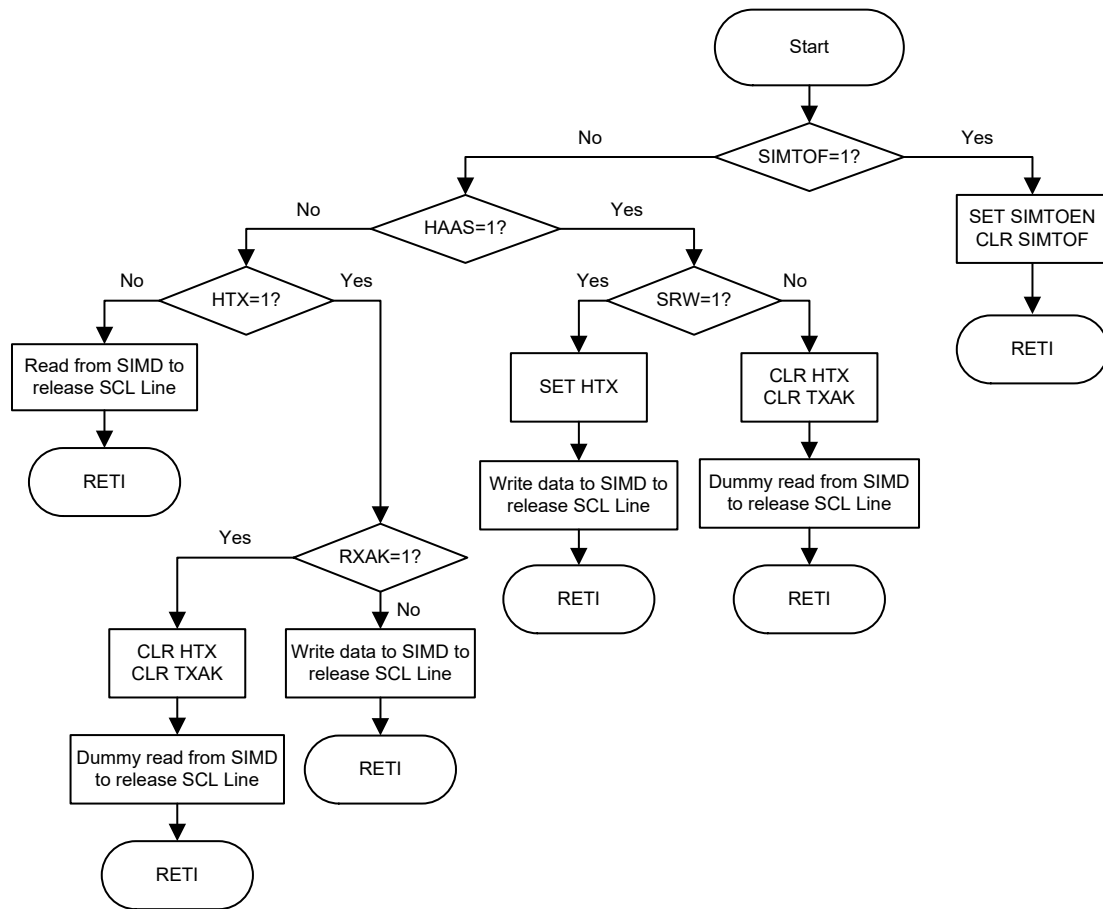
在从机确认接收到从机地址后，会进行 8 位宽度的数据传输。这个数据传输顺序是的高位在前，低位在后。接收方在接收到 8 位数据后必须发出一个应答信号（“0”）以继续接收下一个数据。如果发送方没接收到应答信号，发送方将释放 SDA 线，同时，主机将发出 STOP 信号以释放 I²C 总线。所传送的数据存储在 SIMD 寄存器中。如果设置成发送方，从机必须先将欲传输的数据写到 SIMD 寄存器中；如果设置成接收方，从机必须从 SIMD 寄存器读取数据。

当从机接收器想要继续接收下一个数据时，必须在第 9 个时钟发出应答信号 (TXAK)。被设为发送方的从机将检测寄存器 SIMC1 中的 RXAK 位以判断是否传输下一个字节的数据，如果单片机不传输下一个字节，那么它将释放 SDA 线并等待接收主机的停止信号。



I²C 通信时序图

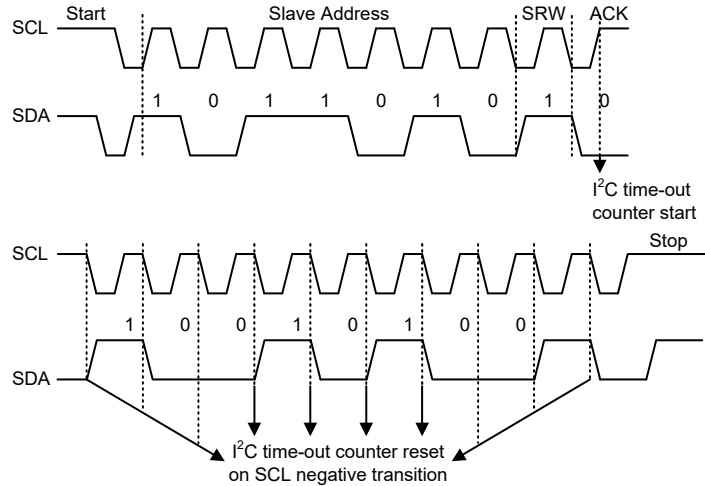
注：当从机地址匹配时，单片机必须选择设置为发送模式还是接收模式。若设置为发送模式，需写数据至 SIMD 寄存器；若设置为接收模式，需立即从 SIMD 寄存器中虚读数据以释放 SCL 线。



I²C 总线 ISR 流程图

I²C 超时控制

超时功能可减少 I²C 接收错误的时钟源而引起的锁死问题。如果连接到 I²C 总线的时钟源经过一段时间还未接收到，则在一定的超时周期后，I²C 电路和寄存器将复位。超时计数器在 I²C 总线“START”和“地址匹配”条件下开始计数，且在 SCL 下降沿清零。在下一个 SCL 下降沿到来之前，如果超时时间大于 SIMTOC 寄存器指定的超时周期，则超时发生。I²C “STOP”条件发生时超时功能终止。



I²C 超时时序图

当 I²C 超时计数器溢出时，计数器将停止计数，SIMTOEN 位被清零，且 SIMTOF 位被置高以表明超时计数器中断发生。超时计数器中断使用的也是 SIM 中断向量。当 I²C 超时发生时，I²C 内部电路会被复位，寄存器也将发生如下复位情况。

| 寄存器 | I ² C 超时发生后 |
|-------------------|------------------------|
| SIMD, SIMA, SIMC0 | 保持不变 |
| SIMC1 | 复位至 POR |

超时发生后的 I²C 寄存器

SIMTOF 标志位由应用程序清零。共有 64 个超时周期，可通过 SIMTOC 寄存器的 SIMTOS 位进行选择。超时周期可通过公式计算： $((1 \sim 64) \times (32 / f_{SUB}))$ 。由此可得超时周期范围为 1ms~64ms。

● SIMTOC 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---------|--------|---------|---------|---------|---------|---------|---------|
| Name | SIMTOEN | SIMTOF | SIMTOS5 | SIMTOS4 | SIMTOS3 | SIMTOS2 | SIMTOS1 | SIMTOS0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7 **SIMTOEN**: SIM I²C 超时功能控制位

- 0: 除能
- 1: 使能

Bit 6 **SIMTOF**: SIM I²C 超时标志位

- 0: 未发生
- 1: 发生

当超时发生时，此位设置为 1，并且只能通过应用程序将其清除为零。

Bit 5~0 **SIMTOS5~SIMTOS0**: SIM I²C 超时时间选择位

I²C 超时时钟源是 $f_{SUB}/32$

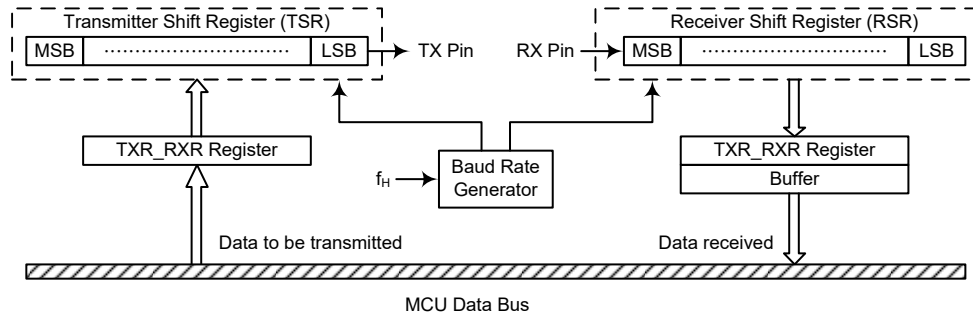
I²C 超时时间计算方法： $(SIMTOS[5:0]+1) \times (32 / f_{SUB})$

UART 接口

单片机具有一个全双工的异步串行通信接口 – UART，可以很方便的与其它具有串行接口的芯片通信。UART 具有许多功能特性，发送或接收串行数据时，将数据组成一个 8 位或 9 位的数据块，连同数据特征位一并传输。具有检测数据覆盖或帧错误等功能。UART 功能占用一个内部中断向量，当接收到数据或数据发送结束，触发 UART 中断。

内置的 UART 功能包含以下特性：

- 全双工通用异步接收器 / 发送器
- 8 位或 9 位传输格式
- 奇校验、偶校验或无校验
- 1 位或 2 位停止位
- 8 位预分频的波特率发生器
- 奇偶、帧、噪声和溢出检测
- 支持地址匹配中断 (最后一位 = 1)
- 独立的发送和接收使能
- 2-byte FIFO 接收缓冲器
- RX 引脚唤醒功能
- 发送和接收中断
- 中断可由下列条件初始化：
 - ◆ 发送器为空
 - ◆ 发送器空闲
 - ◆ 接收完成
 - ◆ 接收器溢出
 - ◆ 地址模式检测



UART 数据传输方框图

UART 外部引脚

内部 UART 有两个外部引脚 TX 和 RX，可与外部串行接口进行通信。TX 和 RX 分别为 UART 发送脚和接收脚，与 I/O 口或其它功能共用引脚。在使能 UART 功能之前，需先通过 IFS 寄存器选择 TX 和 RX 引脚。当 UARTEN、TXEN 和 RXEN 位置高时，将自动设置这些引脚作为 TX 输出和 RX 输入，并且除能 TX 和 RX 引脚上的上拉电阻功能。当 UARTEN、TXEN 或 RXEN 位清零除能 TX 或 RX 引脚功能后，TX 或 RX 引脚将处于浮空状态。这时 TX 或 RX 引脚是否连接内部上拉电阻可由相应的 I/O 上拉电阻控制位决定。

UART 数据传输方案

上图显示了 UART 接口的整体数据传输结构。需要发送的数据首先通过应用程序写入 TXR_RXR 寄存器，接着此数据被传输到发送移位寄存器 TSR 中，然后在波特率发生器的控制下将 TSR 寄存器中数据一位位地移到 TX 引脚上，低位在前。TXR_RXR 寄存器被映射到单片机的数据存储器中，而发送移位寄存器没有实际地址，所以发送移位寄存器不可直接操作。

数据在波特率发生器的控制下，低位在前高位在后，从外部引脚 RX 进入接收移位寄存器 RSR。当数据接收完成，数据从接收移位寄存器移入可被用户程序操作的 TXR_RXR 寄存器中。TXR_RXR 寄存器被映射到单片机数据存储器中，而接收移位寄存器没有实际地址，所以接收移位寄存器不可直接操作。

需要注意的是，发送和接收都是共用同一个地址的数据寄存器，即 TXR_RXR 寄存器。

UART 状态和控制寄存器

与 UART 功能相关的有五个寄存器，即控制 UART 模块整体功能的 USR、UCR1 和 UCR2 寄存器，控制波特率的 BRG 寄存器，管理发送和接收数据的数据寄存器 TXR_RXR。

| 寄存器名称 | 位 | | | | | | | |
|---------|--------|-------|-------|-------|-------|-------|-------|-------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| USR | PERR | NF | FERR | OERR | RIDLE | RXIF | TIDLE | TXIF |
| UCR1 | UARTEN | BNO | PREN | PRT | STOPS | TXBRK | RX8 | TX8 |
| UCR2 | TXEN | RXEN | BRGH | ADDEN | WAKE | RIE | THIE | TEIE |
| TXR_RXR | TXRX7 | TXRX6 | TXRX5 | TXRX4 | TXRX3 | TXRX2 | TXRX1 | TXRX0 |
| BRG | BRG7 | BRG6 | BRG5 | BRG4 | BRG3 | BRG2 | BRG1 | BRG0 |

UART 寄存器列表

• TXR_RXR 寄存器

TXR_RXR 是一个数据寄存器，用来存储 TX 引脚将要发送或 RX 引脚正在接收的数据。

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|-------|-------|-------|-------|-------|-------|-------|
| Name | TXRX7 | TXRX6 | TXRX5 | TXRX4 | TXRX3 | TXRX2 | TXRX1 | TXRX0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | x | x | x | x | x | x | x | x |

“x”：未知

Bit 7~0 TXRX7~TXRX0: UART 发送 / 接收数据位

● **USR 寄存器**

寄存器 USR 是 UART 的状态寄存器，可以通过程序读取。所有 USR 位是只读的。详细解释如下：

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|----|------|------|-------|------|-------|------|
| Name | PERR | NF | FERR | OERR | RIDLE | RXIF | TIDLE | TXIF |
| R/W | R | R | R | R | R | R | R | R |
| POR | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |

- Bit 7 PERR:** 奇偶校验出错标志位
 0: 奇偶校验正确
 1: 奇偶校验出错
 PERR 是奇偶校验出错标志位。若 PERR=0，奇偶校验正确；若 PERR=1，接收到的数据奇偶校验出错。只有使能了奇偶校验此位才有效。可使用软件清除该标志位，即先读取 USR 寄存器再读 TXR_RXR 寄存器来清除此位。
- Bit 6 NF:** 噪声干扰标志位
 0: 没有受到噪声干扰
 1: 受到噪声干扰
 NF 是噪声干扰标志位。若 NF=0，没有受到噪声干扰；若 NF=1，UART 接收数据时受到噪声干扰。它与 RXIF 在同周期内置位，但不会与溢出标志位同时置位。可使用软件清除该标志位，即先读取 USR 寄存器再读 TXR_RXR 寄存器将清除此标志位。
- Bit 5 FERR:** 帧错误标志位
 0: 无帧错误发生
 1: 有帧错误发生
 FREE 是帧错误标志位。若 FREE=0，没有帧错误发生；若 FREE=1，当前的数据发生了帧错误。可使用软件清除该标志位，即先读取 USR 寄存器再读 TXR_RXR 寄存器来清除此位。
- Bit 4 OERR:** 溢出错误标志位
 0: 无溢出错误发生
 1: 有溢出错误发生
 OERR 是溢出错误标志位，表示接收缓冲器是否溢出。若 OERR=0，没有溢出错误；若 OERR=1，发生了溢出错误，它将影响下一组数据的接收。可通过软件清除该标志位，即先读取 USR 寄存器再读 TXR_RXR 寄存器将清除此标志位。
- Bit 3 RIDLE:** 接收状态标志位
 0: 正在接收数据
 1: 接收器空闲
 RIDLE 是接收状态标志位。若 RIDLE=0，正在接收数据；若 RIDLE=1，接收器空闲。在接收到停止位和下一个数据的起始位之间，RIDLE 被置位，表明 UART 空闲，RX 脚处于逻辑高状态。
- Bit 2 RXIF:** 接收寄存器状态标志位
 0: TXR_RXR 寄存器为空
 1: TXR_RXR 寄存器含有有效数据
 RXIF 是接收寄存器状态标志位。当 RXIF=0，TXR_RXR 寄存器为空；当 RXIF=1，TXR_RXR 寄存器接收到新数据。当数据从移位寄存器加载到 TXR_RXR 寄存器中，如果 UCR2 寄存器中的 RIE=1，则会触发中断。当接收数据时检测到一个或多个错误时，相应的标志位 NF、FERR 或 PERR 会在同一周期内置位。读取 USR 寄存器再读 TXR_RXR 寄存器，如果 TXR_RXR 寄存器中没有新的数据，那么将清除 RXIF 标志。
- Bit 1 TIDLE:** 数据发送完成标志位
 0: 数据传输中
 1: 无数据传输
 TIDLE 是数据发送完成标志位。若 TIDLE=0，数据传输中。当 TXIF=1 且数据发送完毕或者暂停字被发送时，TIDLE 置位。TIDLE=1，TX 引脚空闲且处于逻

辑高状态。读取 USR 寄存器再写 TXR 寄存器将清除 TIDLE 位。数据字符或暂停字就绪时，不会产生该标志位。

- Bit 0 **TXIF**: 发送数据寄存器 TXR 状态位
 0: 数据还没有从缓冲器加载到移位寄存器中
 1: 数据已从缓冲器加载到移位寄存器中 (TXR 数据寄存器为空)
 TXIF 是发送数据寄存器为空标志位。若 TXIF=0, 数据还没有从缓冲器加载到移位寄存器中; 若 TXIF=1, 数据已从缓冲器中加载到移位寄存器中。读取 USR 寄存器再写 TXR 寄存器将清除 TXIF。当 TXEN 被置位, 由于发送缓冲器未滿, TXIF 也会被置位。

● **UCR1 寄存器**

UCR1 和 UCR2 是 UART 的两个控制寄存器, 用来定义各种 UART 功能, 例如 UART 的使能与除能、奇偶校验控制和传输数据的长度等等。详细解释如下:

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|--------|-----|------|-----|-------|-------|-----|-----|
| Name | UARTEN | BNO | PREN | PRT | STOPS | TXBRK | RX8 | TX8 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R | W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | x | 0 |

“x”: 未知

- Bit 7 **UARTEN**: UART 功能使能位
 0: UART 除能, TX 和 RX 脚作为其它功能共用引脚
 1: UART 使能, TX 和 RX 脚作为 UART 功能引脚
 此位为 UART 的使能位。UARTEN=0, UART 除能, RX 和 TX 作为其它功能共用引脚; UARTEN=1, UART 使能, TX 和 RX 将分别由 TXEN 和 RXEN 控制。当 UART 被除能将清除缓冲器, 所有缓冲器中的数据将被忽略, 另外波特率计数器、错误和状态标志位被复位, TXEN、RXEN、TXBRK、RXIF、OERR、FERR、PERR 和 NF 清零而 TIDLE、TXIF 和 RIDLE 置位, UCR1、UCR2 和 BRG 寄存器中的其它位保持不变。若 UART 工作时 UARTEN 清零, 所有发送和接收将停止, 模块也将复位成上述状态。当 UART 再次使能时, 它将在上次配置下重新工作。
- Bit 6 **BNO**: 发送数据位数选择位
 0: 8-bit 传输数据
 1: 9-bit 传输数据
 BNO 是发送数据位数选择位。BNO=1, 传输数据为 9 位; BNO=0, 传输数据为 8 位。若选择了 9 位数据传输格式, RX8 和 TX8 将分别存储接收和发送数据的第 9 位。
- Bit 5 **PREN**: 奇偶校验使能位
 0: 奇偶校验除能
 1: 奇偶校验使能
 此位为奇偶校验使能位。PREN=1, 使能奇偶校验; PREN=0, 除能奇偶校验。
- Bit 4 **PRT**: 奇偶校验选择位
 0: 偶校验
 1: 奇校验
 奇偶校验选择位。PRT=1, 奇校验; PRT=0, 偶校验。
- Bit 3 **STOPS**: 停止位的长度选择位
 0: 有一位停止位
 1: 有两位停止位
 此位用来设置停止位的长度。STOP=1, 有两位停止位; STOP=0, 只有一位停止位。
- Bit 2 **TXBRK**: 暂停字发送控制位
 0: 没有暂停字要发送
 1: 发送暂停字
 TXBRK 是暂停字发送控制位。TXBRK=0, 没有暂停字要发送, TX 引脚正常操

作：TXBRK=1，将会发送暂停字，发送器将发送逻辑“0”。若 TXBRK 为高，缓冲器中数据发送完毕后，发送器将至少保持 13 位宽的低电平直至 TXBRK 复位。

Bit 1 **RX8**: 接收 9-bit 数据传输格式中的第 9 位 (只读)
此位只有在传输数据为 9 位的格式中有效，用来存储接收数据的第 9 位。BNO 是用来控制传输位数是 8 位还是 9 位。

Bit 0 **TX8**: 发送 9-bit 数据传输格式中的第 9 位 (只写)
此位只有在传输数据为 9 位的格式中有效，用来存储发送数据的第 9 位。BNO 是用来控制传输位数是 8 位还是 9 位。

● **UCR2 寄存器**

UCR2 是 UART 的第二个控制寄存器，它的主要功能是控制发送器、接收器以及各种 UART 中断源的使能或除能。它也可用来控制波特率，使能接收唤醒和地址侦测。详细解释如下：

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|------|------|-------|------|-----|------|------|
| Name | TXEN | RXEN | BRGH | ADDEN | WAKE | RIE | TIIE | TEIE |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7 **TXEN**: UART 发送使能位
0: UART 发送除能
1: UART 发送使能
此位为发送使能位。TXEN=0，发送将被除能，发送器立刻停止工作。另外缓冲器将被复位，此时 TX 引脚将作为其它功能共用引脚。若 TXEN=1 且 UARTEN=1，则发送将被使能，TX 引脚将由 UART 来控制。在数据传输时清除 TXEN 将中止数据发送且复位发送器，此时 TX 引脚将作为其它功能共用引脚。

Bit 6 **RXEN**: UART 接收使能位
0: UART 接收除能
1: UART 接收使能
此位为接收使能位。RXEN=0，接收将被除能，接收器立刻停止工作。另外缓冲器将被复位，此时 RX 引脚将作为其它功能共用引脚。若 RXEN=1 且 UARTEN=1，则接收将被使能，RX 引脚将由 UART 来控制。在数据传输时清除 RXEN 将中止数据接收且复位接收器，此时 RX 引脚将作为其它功能共用引脚。

Bit 5 **BRGH**: 波特率发生器高低速选择位
0: 低速波特率
1: 高速波特率
此位为波特率发生器高低速选择位，它和 BRG 寄存器一起控制 UART 的波特率。BRGH=1，为高速模式；BRGH=0，为低速模式。

Bit 4 **ADDEN**: 地址检测使能位
0: 地址检测除能
1: 地址检测使能
此位为地址检测使能控制位。ADDEN=1，地址检测使能，此时数据的第 8 位 (BON=0) 或第 9 位 (BON=1) 为高，那么接到的是地址而非数据。若相应的中断使能且接收到的值最高位为 1，那么中断请求标志将会被置位，若最高位为 0，那么将不会产生中断且收到的数据也会被忽略。

Bit 3 **WAKE**: RX 脚下降沿唤醒 UART 功能使能位
0: RX 脚下降沿唤醒 UART 功能除能
1: RX 脚下降沿唤醒 UART 功能使能
此位用于控制 RX 引脚下降沿时是否唤醒 UART 功能。此位仅当 UART 时钟源 f_{clk} 关闭时有效。若 UART 时钟源 f_{clk} 还开启，则无 RX 引脚唤醒 UART 功能无效。若此位置高且 UART 时钟 f_{clk} 关闭，当 RX 引脚发生下降沿时会产生 UART 唤醒

请求。若相应的中断使能，将产生 RX 引脚唤醒 UART 的中断，以告知单片机使其通过应用程序开启 UART 时钟源 f_H ，从而唤醒 UART 功能。否则，若此位为低，即使 RX 引脚发生下降沿也无法恢复 UART 功能。

Bit 2 **RIE**: 接收中断使能位
0: 接收中断除能
1: 接收中断使能

此位为接收中断使能或除能位。若 RIE=1，当 OERR 或 RXIF 置位时，UART 的中断请求标志置位；若 RIE=0，UART 中断请求标志不受 OERR 和 RXIF 影响。

Bit 1 **TIIE**: 发送器空闲中断使能位
0: 发送器空闲中断除能
1: 发送器空闲中断使能

此位为发送器空闲中断的使能或除能位。若 TIIE=1，当 TIDLE 置位时，UART 的中断请求标志置位；若 TIIE=0，UART 中断请求标志不受 TIDLE 的影响。

Bit 0 **TEIE**: 发送寄存器为空中断使能位
0: 发送寄存器为空中断除能
1: 发送寄存器为空中断使能

此位为发送寄存器为空中断的使能或除能位。若 TEIE=1，当 TXIF 置位时，UART 的中断请求标志置位；若 TEIE=0，UART 中断请求标志不受 TXIF 的影响。

• BRG 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|------|------|------|------|------|------|------|
| Name | BRG7 | BRG6 | BRG5 | BRG4 | BRG3 | BRG2 | BRG1 | BRG0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | x | x | x | x | x | x | x | x |

“x”：未知

Bit 7~0 **BRG7~BRG0**: 波特率值

软件设置 BRGH 位 (设置波特率发生器的速度) 和 BRG 寄存器 (设置波特率的值)，一起控制 UART 的波特率。

注：若 BRGH = 0，波特率 = $f_H/[64 \times (N+1)]$ ；

若 BRGH = 1，波特率 = $f_H/[16 \times (N+1)]$ 。

波特率发生器

UART 自身具有一个波特率发生器，通过它可以设定数据传输速率。波特率是由一个独立的内部 8 位计数器产生，它由 BRG 寄存器和 UCR2 寄存器的 BRGH 位来控制。BRGH 是决定波特率发生器处于高速模式还是低速模式，从而决定计算公式的选用。BRG 寄存器的值 N 可根据下表中的公式计算，N 的范围是 0 到 255。

| UCR2 的 BRGH 位 | 0 | 1 |
|---------------|-------------------------|-------------------------|
| 波特率 (BR) | $f_H/[64 \times (N+1)]$ | $f_H/[16 \times (N+1)]$ |

为得到相应的波特率，首先需要设置 BRGH 来选择相应的计算公式从而算出 BRG 的值。由于 BRG 的值不连续，所以实际波特率和理论值之间有一个偏差。

下面举例怎样计算 BRG 寄存器中的值 N 和误差。

波特率和误差的计算

若选用 4MHz 时钟频率且 BRGH=0，若期望的波特率为 4800，计算它的 BRG 寄存器的值 N，实际波特率和误差。

根据上表，波特率 $BR = f_H / [64 (N+1)]$

转换后的公式 $N = [f_H / (BR \times 64)] - 1$

带入参数 $N = [4000000 / (4800 \times 64)] - 1 = 12.0208$
 取最接近的值，十进制 12 写入 BRG 寄存器，实际波特率如下
 $BR = 4000000 / [64 \times (12 + 1)] = 4808$
 因此，误差 = $(4808 - 4800) / 4800 = 0.16\%$

UART 模块的设置与控制

UART 采用标准的不归零码传输数据，这种方法通常被称为 NRZ 法。它由 1 位起始位，8 位或 9 位数据位和 1 位或者两位停止位组成。奇偶校验是由硬件自动完成的，可设置成奇校验、偶校验和无校验三种格式。常用的数据传输格式由 8 位数据位，1 位停止位，无校验组成，用 8、N、1 表示，它是系统上电的默认格式。数据位数、停止位数和奇偶校验由 UCR1 寄存器的 BNO、PRT、PREN 和 STOPS 设定。用于数据发送和接收的波特率由一个内部的 8 位波特率发送器产生，数据传输时低位在前高位在后。尽管 UART 发送器和接收器在功能上相互独立，但它们使用相同的数据传输格式和波特率，在任何情况下，停止位是必须的。

UART 的使能和除能

UART 是由 UCR1 寄存器的 UARTEN 位来使能和除能的。若 UARTEN、TXEN 和 RXEN 都为高，则 TX 和 RX 分别为 UART 的发送端口和接收端口。若没有数据发送，TX 引脚默认状态为高电平。

UARTEN 清零将除能 TX 和 RX，这两个引脚可用作普通 I/O 口或其它引脚共用功能。当 UART 被除能时将清空缓冲器，所有缓冲器中的数据将被忽略，另外一些使能控制、错误标志和状态标志将被复位，如 TXEN、RXEN、TXBRK、RXIF、OERR、FERR、PERR 和 NF 清零，而 TIDLE、TXIF 和 RIDLE 置位，UCR1、UCR2 和 BRG 寄存器中的其它位保持不变。若 UART 工作时 UARTEN 清零，所有发送和接收将停止，模块也将复位成上述状态。当 UART 再次使能时，它将在上次配置下重新工作。

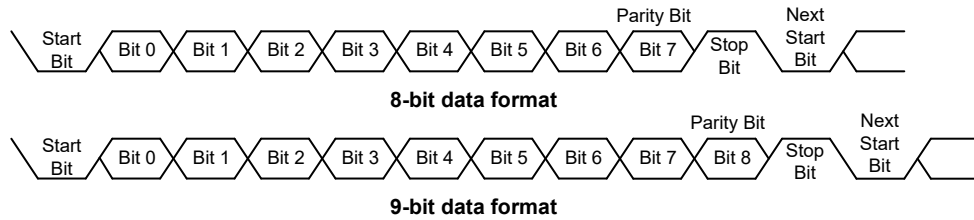
数据位、停止位位数以及奇偶校验的选择

数据传输格式由数据长度、是否校验、校验类型、地址位以及停止位长度组成。它们都是由 UCR1 寄存器的各个位控制的。BNO 决定数据传输是 8 位还是 9 位；PRT 决定校验类型；PREN 决定是否选择奇偶校验；而 STOPS 决定选用 1 位还是 2 位停止位。下表列出了各种数据传输格式。若地址检测功能使能，地址位，即数据字节的最高位，用来确定此帧是地址还是数据。停止位的长度和数据位的长度无关，且只有发送器需设置停止位长度。接收器只接收一个停止位。

| 起始位 | 数据位 | 地址位 | 校验位 | 停止位 |
|---------------|-----|-----|-----|-----|
| 8 位数据位 | | | | |
| 1 | 8 | 0 | 0 | 1 |
| 1 | 7 | 0 | 1 | 1 |
| 1 | 7 | 1 | 0 | 1 |
| 9 位数据位 | | | | |
| 1 | 9 | 0 | 0 | 1 |
| 1 | 8 | 0 | 1 | 1 |
| 1 | 8 | 1 | 0 | 1 |

发送和接收数据格式

下图是传输 8 位和 9 位数据的波形。



UART 发送器

UCR1 寄存器的 BNO 位是控制数据传输的长度。BNO=1 其长度为 9 位，第 9 位 MSB 存储在 UCR1 寄存器的 TX8 中。发送器的核心是发送移位寄存器 TSR，它的数据由发送寄存器 TXR_RXR 提供，应用程序只须将发送数据写入 TXR_RXR 寄存器。上组数据的停止位发出前，TSR 寄存器禁止写入。如果还有新的数据要发送，一旦停止位发出，待发数据将会从 TXR_RXR 寄存器加载到 TSR 寄存器。TSR 不像其它寄存器一样映射到数据存储区，所以应用程序不能对其进行读写操作。TXEN=1，发送使能，但若 TXR_RXR 寄存器没有数据或者波特率没有设置，发送器将不会工作。先写 TXR_RXR 寄存器再置高 TXEN 也会触发发送。当发送器使能，若 TSR 寄存器为空，数据写入 TXR_RXR 寄存器将会直接加载到 TSR 寄存器中。发送器工作时，TXEN 清零，发送器将立刻停止工作并且复位，此时 TX 引脚用作普通 I/O 口或其它引脚共用功能。

发送数据

当 UART 发送数据时，数据从移位寄存器中移到 TX 引脚上，其低位在前高位在后。在发送模式中，TXR_RXR 寄存器在内部总线和发送移位寄存器间形成一个缓冲。如果选择 9 位数据传输格式，最高位 MSB 取自 UCR1 寄存器的 TX8。

发送器初始化可由如下步骤完成：

- 正确地设置 BNO、PRT、PREN 和 STOPS 位以确定数据长度、校验类型和停止位长度。
- 设置 BRG 寄存器，选择期望的波特率。
- 置高 TXEN，使能 UART 发送器且使 TX 作为 UART 的发送端。
- 读取 USR 寄存器，然后将待发数据写入 TXR_RXR 寄存器。注意，此步骤会清除 TXIF 标志位。

如果要发送多个数据只需重复上一步骤。

当 TXIF=0 时，数据将禁止写入 TXR_RXR 寄存器。可以通过以下步骤来清除 TXIF：

1. 读取 USR 寄存器
2. 写 TXR_RXR 寄存器

只读标志位 TXIF 由 UART 硬件置位。若 TXIF=1，TXR_RXR 寄存器为空，其它数据可以写入而不会覆盖之前的数据。若 TEIE=1，TXIF 标志位会产生中断。

在数据传输时，写 TXR_RXR 指令会将待发数据暂存在 TXR_RXR 寄存器中，当前数据发送完毕后，待发数据被加载到发送移位寄存器中。当发送器空闲时，写 TXR_RXR 指令会将数据直接加载到 TSR 寄存器中，数据传输立刻开始且 TXIF 置位。当发送完停止位或暂停帧后，表示一帧数据已发送完毕，此时 TIDLE 位将被置位。

可以通过以下步骤来清除 TIDLE:

1. 读取 USR 寄存器
2. 写 TXR_RXR 寄存器

注意, 清除 TXIF 和 TIDLE 软件执行次序相同。

发送暂停字

若 TXBRK=1, 下一帧将会发送暂停字。它是由一个起始位、 $13 \times N$ ($N=1, 2, \dots$) 位逻辑 0 组成。置位 TXBRK 将会发送暂停字, 而清除 TXBRK 将产生停止位, 传输暂停字不会产生中断。需要注意的是, 暂停字至少 13 位宽。若 TXBRK 持续为高, 那么发送器会一直发送暂停字; 当应用程序将 TXBRK 清零后, 发送器结束最后一帧暂停字的发送后接着发送一位或两位停止位。最后一帧暂停字的结尾自动为高电平, 以确保下一帧数据起始位的检测。

UART 接收器

UART 接收器支持 8 位或者 9 位数据接收。若 BNO=1, 数据长度为 9 位, 而最高位 MSB 存放在 UCR1 寄存器的 RX8 中。接收器的核心是串行移位寄存器 RSR。RX 引脚上的数据送入数据恢复器中, 它在 16 倍波特率的频率下工作, 而串行移位器工作在正常波特率下。当在 RX 引脚上检测到停止位, 若 TXR_RXR 寄存器为空, 数据从 RSR 寄存器中加载到 TXR_RXR 寄存器。RX 引脚上的每一位数据会被采样三次以判断其逻辑状态。RSR 不像其它寄存器一样映射在数据存储区, 所以应用程序不能对其进行读写操作。

接收数据

当 UART 接收数据时, 数据低位在前高位在后, 连续地从 RX 引脚进入移位寄存器。TXR_RXR 寄存器在内部总线和接收移位寄存器间形成一个缓冲。TXR_RXR 寄存器是一个两层的 FIFO 缓冲器, 它能保存两帧数据的同时接收第三帧数据, 应用程序必须保证在接收完第三帧前读取 TXR_RXR 寄存器, 否则忽略第三帧数据并且发生溢出错误。

接收器的初始化可由如下步骤完成:

- 正确地设置 BNO、PRT 和 PREN 位以确定数据长度和校验类型。
- 设置 BRG 寄存器, 选择期望的波特率。
- 置高 RXEN, 使能 UART 发送器且使 RX 作为 UART 的接收端。

此时接收器被使能并检测起始位。

接收数据将会发生如下事件:

- 当 TXR_RXR 寄存器中包含有效数据时, USR 寄存器中的 RXIF 位将会置位, 溢出错误发生之前至多还有一帧数据可读。
- 若 RIE=1, 数据从 RSR 寄存器加载到 TXR_RXR 寄存器中将产生中断。
- 若接收器检测到帧错误、噪声干扰错误、奇偶出错或溢出错误, 那么相应的错误标志位置位。

可以通过如下步骤来清除 RXIF:

1. 读取 USR 寄存器
2. 读取 TXR_RXR 寄存器

接收暂停字

UART 接收任何暂停字都会当作帧错误处理。接收器只根据 BNO 位的设置外加

一个停止位来确定一帧数据的长度。若暂停字数大于 BNO 位指定的长度外加一个停止位，接收器认为接收已完毕，RXIF 和 FERR 置位，TXR_RXR 寄存器清 0，若相应的中断允许且 RIDLE 为高将会产生中断。暂停字只会被认为包含信息 0 且会置位 FERR 标志位。如果检测到较长的暂停信号，接收器会将此信号视为包含一个起始位、数据位和无效的停止位的数据帧并且置位 FERR 标志位。在下一个开始位到来之前，接收器必须等待一个有效的停止位。接收器不会假定线上的暂停信号是下一个开始位。暂停字将会加载到缓冲器中，在接收到停止位前不会再接收数据，需要注意的是，没有检测到停止位也会置位只读标志位 RIDLE。

UART 接收到暂停字会产生以下事件：

- 帧错误标志位 FERR 置位。
- TXR_RXR 寄存器清零。
- OERR、NF、PERR、RIDLE 或 RXIF 可能会置位。

空闲状态

当 UART 接收数据时，即在起初位和停止位之间，USR 寄存器的接收状态标志位 RIDLE 清零。在停止位和下一帧数据的起始位之间，RIDLE 被置位，表示接收器空闲。

接收中断

USR 寄存器的只读标志位 RXIF 由接收器的边沿触发置位。若 RIE=1，数据从移位寄存器 RSR 加载到 TXR_RXR 寄存器时产生中断，同样地，溢出也会产生中断。

接收错误处理

UART 会产生几种接收错误，下面部分将描述各错误以及怎样处理。

溢出 – OERR 标志

TXR_RXR 寄存器是一个两层的 FIFO 缓冲器，它能保存两帧数据的同时接收第三帧数据，应用程序必须保证在接收完第三帧前读取 TXR_RXR 寄存器，否则发生溢出错误。

产生溢出错误时将会发生以下事件：

- USR 寄存器中 OERR 被置位。
- TXR_RXR 寄存器中数据不会丢失。
- RSR 寄存器数据将会被覆盖。
- 若 RIE=1，将会产生中断。

先读取 USR 寄存器再读取 TXR_RXR 寄存器可将 OERR 清零。

噪声干扰 – NF 标志

数据恢复时多次采样可以有效的鉴别出噪声干扰。当检测到数据受到噪声干扰时将会发生以下事件：

- 在 RXIF 上升沿，USR 寄存器中只读标志位 NF 置位。
- 数据从 RSR 寄存器加载到 TXR_RXR 寄存器中。
- 不产生中断，但此位置位发生在 RXIF 置位产生中断的同周期内。

先读取 USR 寄存器再读取 TXR_RXR 寄存器可将 NF 清零。

帧错误 – FERR 标志

若在停止位上检测到 0，USR 寄存器中只读标志 FERR 置位。若选择两位停止位，此两位都必须为高，否则将置位 FERR。此标志位同接收的数据分别记录在 USR 寄存器和 TXR_RXR 寄存器中，此标志位可被任何复位清零。

奇偶校验错误 – PERR 标志

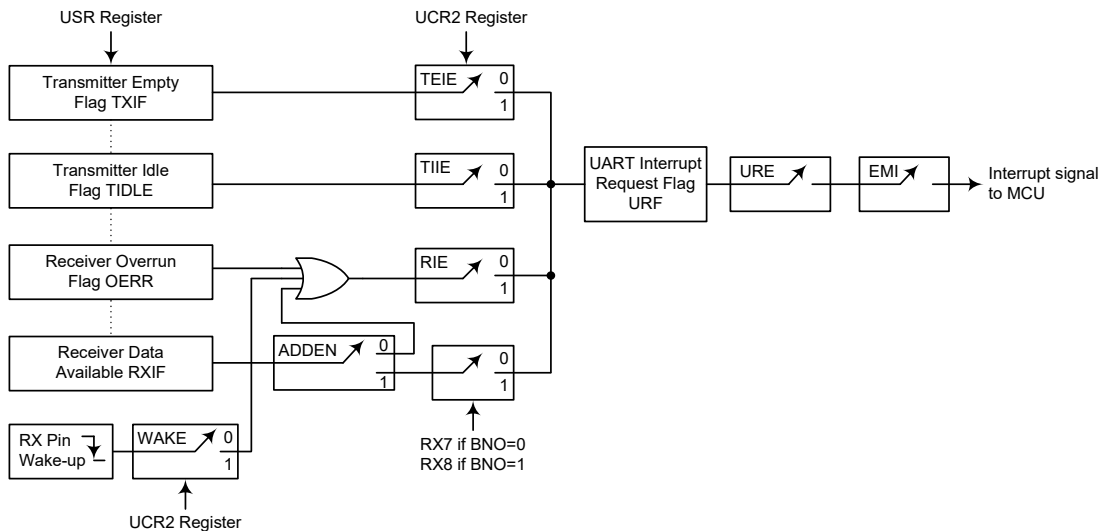
若接收到数据出现奇偶校验错误，USR 寄存器中只读标志 PERR 置位。只有使能了奇偶校验，选择了校验类型，此标志位才有效。此标志位同接收的数据分别记录在 USR 寄存器和 TXR_RXR 寄存器中，此标志位可被任何复位清零。注意，在读取相应的数据之前必须先访问 USR 寄存器中的 FERR 和 PERR 错误标志位。

UART 模块中断结构

几个独立的 UART 条件可以产生一个 UART 中断。当条件满足时，会产生一个低脉冲信号。发送寄存器为空、发送器空闲、接收器数据有效、溢出和地址检测和 RX 引脚唤醒都会产生中断。若总中断使能、UART 中断允许且堆栈未满，程序将会跳转到相应的中断向量执行中断服务程序，而后再返回主程序。其中四种情况，若其 UCR2 寄存器中相应中断允许位被置位，则 USR 寄存器中对应中断标志位将产生 UART 中断。发送器相关的两个中断情况有各自对应的中断允许位，而接收器相关的两个中断情况共用一个中断允许位。这些允许位可用于禁止个别的 UART 模式中断源。

地址检测也是 UART 模式的中断源，它没有相应的标志位，若 UCR2 寄存器中 ADDEN=1，当检测到地址将会产生 UART 中断。RX 引脚唤醒也可以产生 UART 中断，它没有相应的标志位，当 UART 时钟源 f_{clk} 关闭且 UCR2 中的 WAKE 和 RIE 位被置位，RX 引脚上有下降沿时会产生 UART 中断。

注意，USR 寄存器标志位为只读状态，软件不能对其进行设置，和其它一些中断一样，在进入相应中断服务程序时也不能清除这些标志位。这些标志位仅在 UART 特定动作发生时才会自动被清除，详细解释见 UART 寄存器章节。整体 UART 中断的使能或除能可由中断控制寄存器中的 UART 中断使能控制位控制，其中断请求由 UART 模块决定。



UART 中断框图

地址检测模式

置位 UCR2 寄存器中的 ADDEN 将启动地址检测模式。若此位为“1”，可产生接收数据有效中断，其请求标志位为 RXIF。若 ADDEN 有效，只有在接收到数据最高位为 1 才会产生中断，中断允许位 URE 和 EMI 也要使能才会产生中断。地址的最高位为第 9 位 (BNO=1) 或第 8 位 (BNO=0)，若此位为高，则接收到的是地址而非数据。只有接收的数据的最后一位为高才会产生中断。若 ADDEN 除能，每接收到一个有效数据便会置位 RXIF，而不用考虑数据的最后一位。地址检测和奇偶校验在功能上相互排斥，若地址检测模式使能，为了确保操作正确，必须将奇偶校验使能位清零以除能奇偶校验。

| ADDEN | Bit 9 (BNO=1) Bit 8 (BNO=0) | 产生 UART 中断 |
|-------|--------------------------------|------------|
| 0 | 0 | √ |
| | 1 | √ |
| 1 | 0 | × |
| | 1 | √ |

ADDEN 位功能

UART 模块暂停和唤醒

UART 时钟 f_{H} 关闭后 UART 模块将停止运行。当传送数据时 UART 时钟 f_{H} 关闭，发送将停止直到 UART 模块时钟再次使能。同样地，当接收数据时单片机进入空闲或休眠模式，数据接收也会停止。当单片机进入空闲或休眠模式，USR、UCR1、UCR2、接收 / 发送寄存器以及 BRG 寄存器都不会受到影响。建议在单片机进入空闲或休眠模式前先确保数据发送或接收已完成。

UART 功能中包括了 RX 引脚的唤醒功能，由 UCR2 寄存器中 WAKE 位控制。当 UART 时钟 f_{H} 关闭时，若 WAKE 位与 UART 允许位 UARTEN、接收器允许位 RXEN 和接收器中断允许位 RIE 都被置位，则 RX 引脚的下降沿可触发产生 RX 引脚唤醒 UART 的中断。唤醒后系统需延时一段时间才能正常工作，在此期间，RX 引脚上的任何数据将被忽略。

若要产生唤醒 UART 的中断，除了唤醒使能控制位和接收中断使能控制位需置位外，全局中断允许位 EMI 和 UART 中断使能控制位 URE 也必须置位；若这两控制位没有被置位，那么，可发生唤醒事件但不会产生中断。唤醒后系统需一定的延时才能正常工作，然后才会产生 UART 中断。

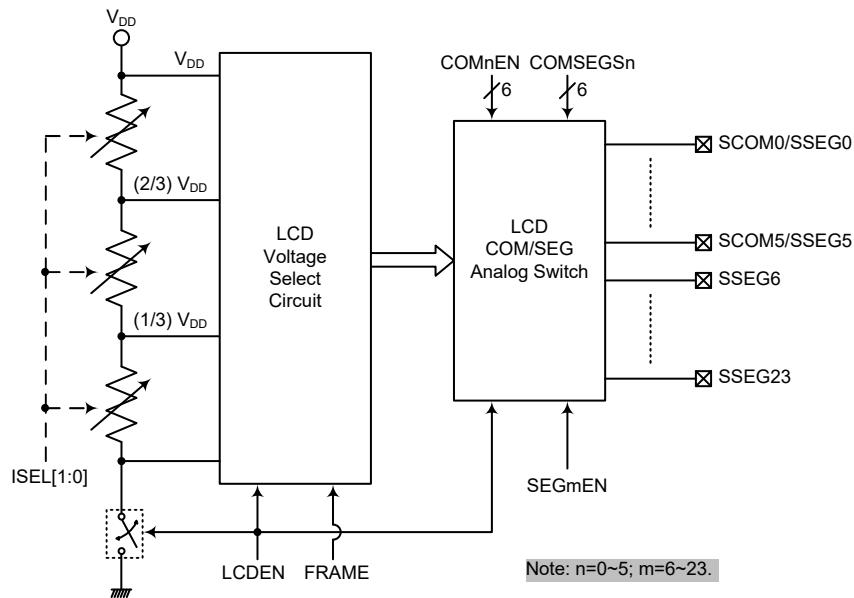
带 SCOM 和 SSEG 功能的 LCD

该单片机具有驱动外部 LCD 面板的能力。LCD 驱动的 COM 脚 SCOM0~SCOM5 和 SEG 脚 SSEG0~SSEG23 与 I/O 口共用。LCD 信号 COM 和 SEG 由应用程序实现。

LCD 操作

该单片机通过设置相关 I/O 引脚为 COM 引脚和 SEG 引脚，以驱动外部 LCD 面板。LCD 驱动功能是由几个 LCD 控制寄存器一起控制的，这些寄存器除了设置 LCD 的开启和关闭外，还控制 SCOM 和 SSEG 引脚的 R-type 偏压电流，使得 LCD 驱动器 COM 和 SEG 引脚输出 V_{SS} 、 $(1/3)V_{DD}$ 、 $(2/3)V_{DD}$ 和 V_{DD} 的电压，从而实现 1/3 bias LCD 的显示。

SLCDC0 寄存器中的 LCDEN 位是 LCD 驱动的主控制位，它与 COMnEN 和 SEGmEN 位搭配共同选择哪些输入/输出引脚用于 LCD 驱动。需注意的是，输入/输出端口控制寄存器不需要设置为输出以使能 LCD 驱动操作。



软件控制 LCD 驱动器结构

LCD Frames

一个完整的 LCD 波形周期包含两个 Frame，即 Frame 0 和 Frame 1。下面将做出详细解释：

Frame 0

当要输出 Frame 0 的波形，需将 SLCDC0 寄存器中的 FRAME 位设为 0。

在 Frame 0，COM 信号输出可以是 V_{DD} ，或是 $V_{BIAS}=(1/3)\times V_{DD}$ 。SEG 信号输出可以是 V_{SS} ，或是 $V_{BIAS}=(2/3)\times V_{DD}$ 。

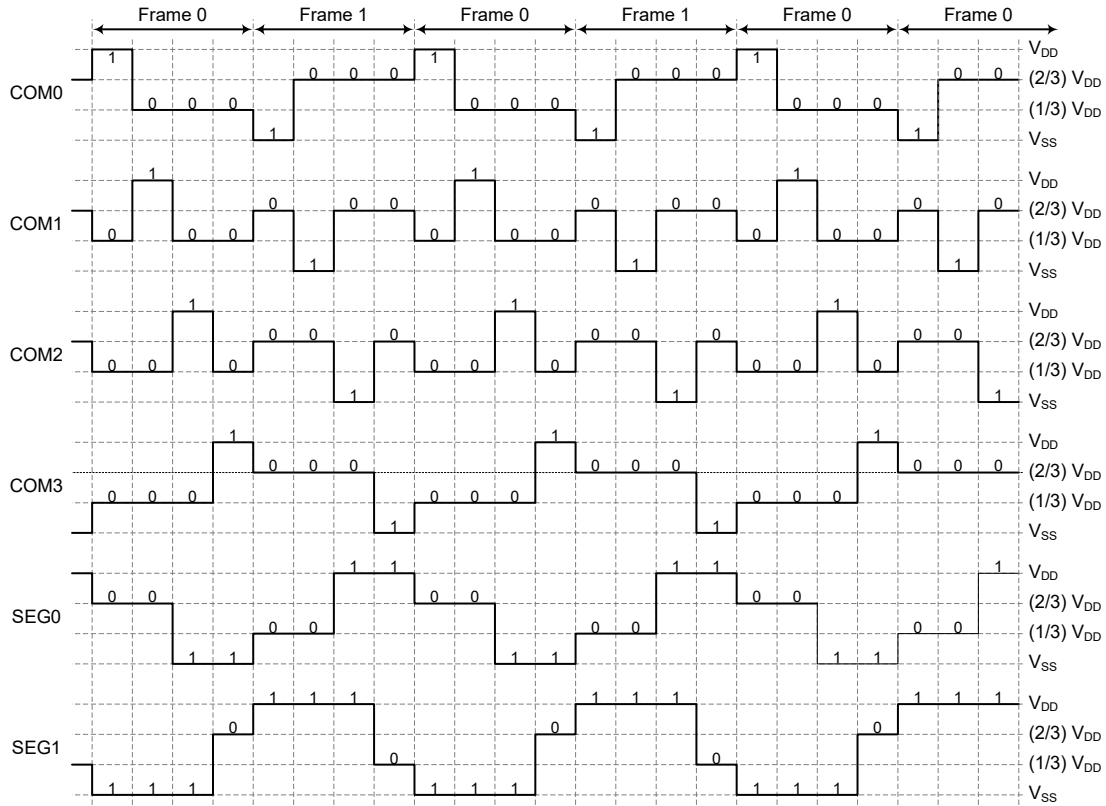
Frame 1

当要输出 Frame 1 的波形，需将 SLCDC0 寄存器中的 FRAME 位设为 1。

在 Frame 1，COM 信号输出可以是 V_{SS} ，或是 $V_{BIAS}=(2/3)\times V_{DD}$ 。SEG 信号输出可以是 V_{DD} ，或是 $V_{BIAS}=(1/3)\times V_{DD}$ 。

COM_n 的波形，由应用程序设定 SLCDC0 寄存器中的 FRAME 位控制，及相应的 I/O 共用引脚数据位决定 COM_n 引脚目前输出是 V_{DD}、V_{SS} 或 V_{BIAS}。SEG_m 的波形，由应用程序设定 FRAME 位控制，及相应的 I/O 共用引脚数据位决定 SEG_m 引脚目前输出是 V_{DD}、V_{SS} 或 V_{BIAS}。

典型的 1/3 bias LCD 波形由应用程序以及 LCD 电压选择电路产生，波形图如下。请注意，图中“1”代表点亮 LCD 像素。SCOM0~SCOM5 引脚的 COM 信号极性为“0”或“1”，由相应的 I/O 共用引脚数据位产生。



注：图形中的逻辑值即为相应 I/O 共用引脚数据位值

1/3 Bias LCD 波形 – 4-COM 和 2-SEG 应用

LCD 控制寄存器

LCD 驱动器 COM 和 SEG 口可以提供多种驱动电流选择以适应不同 LCD 面板的需求。通过设置 SLCDC0 寄存器中 ISEL0 位和 ISEL1 位可以配置不同的偏压电阻。所有 COM 和 SEG 引脚和 I/O 引脚共用，可分别通过 SLCDC_n 寄存器的相应引脚功能选择位选择 COM 和 SEG 引脚。

| 寄存器列表 | 位 | | | | | | | |
|--------|---------|---------|----------|----------|----------|----------|----------|----------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SLCDC0 | FRAME | ISEL1 | ISEL0 | LCDEN | COM3EN | COM2EN | COM1EN | COM0EN |
| SLCDC1 | COM5EN | COM4EN | COMSEGS5 | COMSEGS4 | COMSEGS3 | COMSEGS2 | COMSEGS1 | COMSEGS0 |
| SLCDC2 | SEG13EN | SEG12EN | SEG11EN | SEG10EN | SEG9EN | SEG8EN | SEG7EN | SEG6EN |
| SLCDC3 | SEG21EN | SEG20EN | SEG19EN | SEG18EN | SEG17EN | SEG16EN | SEG15EN | SEG14EN |
| SLCDC4 | — | — | — | — | — | — | SEG23EN | SEG22EN |

LCD 驱动器控制寄存器列表

● **SLCDC0 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|-------|-------|-------|--------|--------|--------|--------|
| Name | FRAME | ISEL1 | ISEL0 | LCDEN | COM3EN | COM2EN | COM1EN | COM0EN |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Bit 7 **FRAME**: SCOM/SSEG 输出 Frame 选择
0: Frame 0
1: Frame 1
- Bit 6~5 **ISEL1~ISEL0**: SCOM/SSEG 典型偏置电流选择 ($V_{DD}=5V$)
00: 8.3 μ A
01: 16.7 μ A
10: 50 μ A
11: 100 μ A
- Bit 4 **LCDEN**: SCOM/SSEG 模块使能控制
0: 除能
1: 使能
若 LCDEN 被置为 1, 可通过设置 COMnEN 和 SEGmEN 位使能 SCOMn 和 SSEGm 引脚功能。若 LCDEN 被清零, 则 SCOMn 和 SSEGm 将输出 V_{SS} 。
- Bit 3 **COM3EN**: SCOM3/SSEG3 或其它引脚功能选择
0: 其它引脚共用功能
1: SCOM3/SSEG3 功能
- Bit 2 **COM2EN**: SCOM2/SSEG2 或其它引脚功能选择
0: 其它引脚共用功能
1: SCOM2/SSEG2 功能
- Bit 1 **COM1EN**: SCOM1/SSEG1 或其它引脚功能选择
0: 其它引脚共用功能
1: SCOM1/SSEG1 功能
- Bit 0 **COM0EN**: SCOM0/SSEG0 或其它引脚功能选择
0: 其它引脚共用功能
1: SCOM0/SSEG0 功能

● **SLCDC1 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|--------|--------|----------|----------|----------|----------|----------|----------|
| Name | COM5EN | COM4EN | COMSEGS5 | COMSEGS4 | COMSEGS3 | COMSEGS2 | COMSEGS1 | COMSEGS0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Bit 7 **COM5EN**: SCOM5/SSEG5 或其它引脚功能选择
0: 其它引脚共用功能
1: SCOM5/SSEG5 功能
- Bit 6 **COM4EN**: SCOM4/SSEG4 或其它引脚功能选择
0: 其它引脚共用功能
1: SCOM4/SSEG4 功能
- Bit 5 **COMSEGS5**: SCOM5 或 SSEG5 引脚功能选择
0: SCOM5
1: SSEG5
- Bit 4 **COMSEGS4**: SCOM4 或 SSEG4 引脚功能选择
0: SCOM4
1: SSEG4

- Bit 3 **COMSEGS3:** SCOM3 或 SSEG3 引脚功能选择
0: SCOM3
1: SSEG3
- Bit 2 **COMSEGS2:** SCOM2 或 SSEG2 引脚功能选择
0: SCOM2
1: SSEG2
- Bit 1 **COMSEGS1:** SCOM1 或 SSEG1 引脚功能选择
0: SCOM1
1: SSEG1
- Bit 0 **COMSEGS0:** SCOM0 或 SSEG0 引脚功能选择
0: SCOM0
1: SSEG0

● **SLCDC2 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---------|---------|---------|---------|--------|--------|--------|--------|
| Name | SEG13EN | SEG12EN | SEG11EN | SEG10EN | SEG9EN | SEG8EN | SEG7EN | SEG6EN |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Bit 7 **SEG13EN:** SSEG13 引脚功能选择
0: 其它引脚共用功能
1: SSEG13 功能
- Bit 6 **SEG12EN:** SSEG12 引脚功能选择
0: 其它引脚共用功能
1: SSEG12 功能
- Bit 5 **SEG11EN:** SSEG11 引脚功能选择
0: 其它引脚共用功能
1: SSEG11 功能
- Bit 4 **SEG10EN:** SSEG10 引脚功能选择
0: 其它引脚共用功能
1: SSEG10 功能
- Bit 3 **SEG9EN:** SSEG9 引脚功能选择
0: 其它引脚共用功能
1: SSEG9 功能
- Bit 2 **SEG8EN:** SSEG8 引脚功能选择
0: 其它引脚共用功能
1: SSEG8 功能
- Bit 1 **SEG7EN:** SSEG7 引脚功能选择
0: 其它引脚共用功能
1: SSEG7 功能
- Bit 0 **SEG6EN:** SSEG6 引脚功能选择
0: 其它引脚共用功能
1: SSEG6 功能

● **SLCDC3 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---------|---------|---------|---------|---------|---------|---------|---------|
| Name | SEG21EN | SEG20EN | SEG19EN | SEG18EN | SEG17EN | SEG16EN | SEG15EN | SEG14EN |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Bit 7 **SEG21EN:** SSEG21 引脚功能选择
0: 其它引脚共用功能
1: SSEG21 功能
- Bit 6 **SEG20EN:** SSEG20 引脚功能选择
0: 其它引脚共用功能
1: SSEG20 功能
- Bit 5 **SEG19EN:** SSEG19 引脚功能选择
0: 其它引脚共用功能
1: SSEG19 功能
- Bit 4 **SEG18EN:** SSEG18 引脚功能选择
0: 其它引脚共用功能
1: SSEG18 功能
- Bit 3 **SEG17EN:** SSEG17 引脚功能选择
0: 其它引脚共用功能
1: SSEG17 功能
- Bit 2 **SEG16EN:** SSEG16 引脚功能选择
0: 其它引脚共用功能
1: SSEG16 功能
- Bit 1 **SEG15EN:** SSEG15 引脚功能选择
0: 其它引脚共用功能
1: SSEG15 功能
- Bit 0 **SEG14EN:** SSEG14 引脚功能选择
0: 其它引脚共用功能
1: SSEG14 功能

● **SLCDC4 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|---|---|---------|---------|
| Name | — | — | — | — | — | — | SEG23EN | SEG22EN |
| R/W | — | — | — | — | — | — | R/W | R/W |
| POR | — | — | — | — | — | — | 0 | 0 |

- Bit 7~2 未定义，读为“0”
- Bit 1 **SEG23EN:** SSEG23 引脚功能选择
0: 其它引脚共用功能
1: SSEG23 功能
- Bit 0 **SEG22EN:** SSEG22 引脚功能选择
0: 其它引脚共用功能
1: SSEG22 功能

低电压检测 – LVD

该单片机都具有低电压检测功能，即 LVD。该功能使能用于监测电源电压 V_{DD} ，若电源电压低于一定值可提供一个警告信号。此功能在电池类产品中非常有用，在电池电压较低时产生警告信号。低电压检测也可产生中断信号。

LVD 寄存器

低电压检测功能由 LVDC 寄存器控制。VLVD2~VLVD0 位用于选择 8 个固定的电压参考点。LVDO 位被置位时低电压情况发生，若 LVDO 位为低表明 V_{DD} 电压工作在当前所设置低电压水平值之上。LVDEN 位用于控制低电压检测功能的开启 / 关闭，设置此位为高使能此功能，反之，关闭内部低电压检测电路。低电压检测会有一些的功耗，在不使用时可考虑关闭此功能，此举在功耗要求严格的电池供电应用中值得考虑。

• LVDC 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|------|-------|---|-------|-------|-------|
| Name | — | — | LVDO | LVDEN | — | VLVD2 | VLVD1 | VLVD0 |
| R/W | — | — | R | R/W | — | R/W | R/W | R/W |
| POR | — | — | 0 | 0 | — | 0 | 0 | 0 |

Bit 7~6 未定义，读为“0”

Bit 5 **LVDO**: LVD 输出标志位
0: 未检测到低电压
1: 检测到低电压

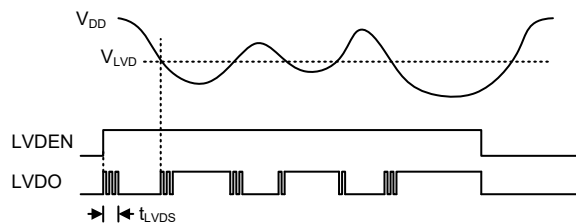
Bit 4 **LVDEN**: 低电压检测使能控制位
0: 除能
1: 使能

Bit 3 未定义，读为“0”

Bit 2~0 **VLVD2~VLVD0**: LVD 电压选择位
000: 2.0V
001: 2.2V
010: 2.4V
011: 2.7V
100: 3.0V
101: 3.3V
110: 3.6V
111: 4.0V

LVD 操作

低电压检测功能的工作原理是比较电源电压 V_{DD} 与 LVDC 寄存器定义的预置电压值。预置电压的范围为 2.0V~4.0V。当电源电压 V_{DD} 低于预置电压值时，LVDO 位被置为高，表明低电压产生。当单片机进入空闲或休眠模式，若 LVDEN 位为高，低电压检测器仍保持有效状态。低电压检测器使能后，读取 LVDO 位前，电路稳定需要一定的延时 t_{LVDs} 。注意， V_{DD} 电压可能上升或下降比较缓慢，在 V_{LVD} 电压值附近时，LVDO 位可能有多种变化。



LVD 操作

低电压检测器也有自己的中断功能，属于多功能中断的一种，它是除了轮询 LVDO 位之外的另一种检测低电压的方法。当低电压情况发生，即 V_{DD} 降至小于 LVD 预置电压值，LVDO 置位并延时 t_{LVDS} 后，中断才会产生。此时中断请求标志位 LVF 将被置位触发产生中断，单片机将从休眠或空闲模式中被唤醒。若不要求低电压检测的唤醒功能使能，在单片机进入休眠或空闲模式前应将 LVF 标志置为高。

中断

中断是单片机一个重要功能。当外部事件或内部功能如定时器模块或 A/D 转换器有效，并且产生中断时，系统会暂时中止当前的程序而转到执行相对应的中断服务程序。此单片机提供多个外部中断和内部中断功能，外部中断由 INT0 和 INT1 引脚动作产生，而内部中断由各种内部功能，如定时器模块、时基、LVD、EEPROM、SIM、UART 和 A/D 转换器等产生。

中断寄存器

中断控制基本上是在一定单片机条件发生时设置请求标志位，应用程序中中断使能位的设置是通过位于特殊功能数据存储器中的一系列寄存器控制的。寄存器总的分为三类。第一类是 INTC0~INTC2 寄存器，用于设置基本的中断；第二类是 MFIO~MFI2 寄存器，用于设置多功能中断；最后一种有 INTEG 寄存器，用于设置外部中断边沿触发类型。

寄存器中含有中断控制位和中断请求标志位。中断控制位用于使能或除能各种中断，中断请求标志位用于存放当前中断请求的状态。它们都按照特定的模式命名，前面表示中断类型的缩写，紧接着的字母“E”代表使能/除能位，“F”代表请求标志位。

| 功能 | 使能位 | 请求标志 | 注释 |
|------------|-------|-------|---------|
| 总中断 | EMI | — | — |
| INTn 脚 | INTnE | INTnF | n=0 或 1 |
| 比较器 | CPE | CPF | — |
| 多功能 | MFnE | MFnF | n=0~2 |
| A/D 转换器 | ADE | ADF | — |
| 时基 | TBnE | TBnF | n=0 或 1 |
| SIM | SIME | SIMF | — |
| UART | URE | URF | — |
| LVD | LVE | LVF | — |
| EEPROM 写操作 | DEE | DEF | — |
| CTM | CTMPE | CTMPF | — |
| | CTMAE | CTMAF | — |

| 功能 | 使能位 | 请求标志 | 注释 |
|-----|-------|-------|----|
| STM | STMPE | STMPF | — |
| | STMAE | STMAF | — |
| PTM | PTMPE | PTMPF | — |
| | PTMAE | PTMAF | — |

中断寄存器位命名模式

| 寄存器名称 | 位 | | | | | | | |
|-------|-------|-------|-------|-------|--------|--------|--------|--------|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| INTEG | — | — | — | — | INT1S1 | INT1S0 | INT0S1 | INT0S0 |
| INTC0 | — | MF0F | CPF | INT0F | MF0E | CPE | INT0E | EMI |
| INTC1 | TB0F | ADF | MF2F | MF1F | TB0E | ADE | MF2E | MF1E |
| INTC2 | URF | SIMF | INT1F | TB1F | URE | SIME | INT1E | TB1E |
| MF10 | — | — | STMAF | STMPF | — | — | STMAE | STMPE |
| MF11 | CTMAF | CTMPF | PTMAF | PTMPF | CTMAE | CTMPE | PTMAE | PTMPE |
| MF12 | — | — | DEF | LVF | — | — | DEE | LVE |

中断寄存器列表

● INTEG 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|--------|--------|--------|--------|
| Name | — | — | — | — | INT1S1 | INT1S0 | INT0S1 | INT0S0 |
| R/W | — | — | — | — | R/W | R/W | R/W | R/W |
| POR | — | — | — | — | 0 | 0 | 0 | 0 |

Bit 7~4 未定义，读为“0”

Bit 3~2 **INT1S1~INT1S0**: INT1 脚中断边沿控制位

00: 除能
01: 上升沿
10: 下降沿
11: 双沿

Bit 1~0 **INT0S1~INT0S0**: INT0 脚中断边沿控制位

00: 除能
01: 上升沿
10: 下降沿
11: 双沿

● INTC0 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|------|-----|-------|------|-----|-------|-----|
| Name | — | MF0F | CPF | INT0F | MF0E | CPE | INT0E | EMI |
| R/W | — | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | — | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7 未定义，读为“0”

Bit 6 **MF0F**: 多功能中断 0 请求标志位

0: 无请求
1: 中断请求

- Bit 5 **CPF:** 比较器中断请求标志位
0: 无请求
1: 中断请求
- Bit 4 **INT0F:** INT0 中断请求标志位
0: 无请求
1: 中断请求
- Bit 3 **MF0E:** 多功能中断 0 控制位
0: 除能
1: 使能
- Bit 2 **CPE:** 比较器中断控制位
0: 除能
1: 使能
- Bit 1 **INT0E:** INT0 中断控制位
0: 除能
1: 使能
- Bit 0 **EMI:** 总中断控制位
0: 除能
1: 使能

● **INTC1 寄存器**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|-----|------|------|------|-----|------|------|
| Name | TB0F | ADF | MF2F | MF1F | TB0E | ADE | MF2E | MF1E |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Bit 7 **TB0F:** 时基 0 中断请求标志位
0: 无请求
1: 中断请求
- Bit 6 **ADF:** A/D 转换器中断请求标志位
0: 无请求
1: 中断请求
- Bit 5 **MF2F:** 多功能中断 2 请求标志位
0: 无请求
1: 中断请求
- Bit 4 **MF1F:** 多功能中断 1 请求标志位
0: 无请求
1: 中断请求
- Bit 3 **TB0E:** 时基 0 中断控制位
0: 除能
1: 使能
- Bit 2 **ADE:** A/D 转换器中断控制位
0: 除能
1: 使能
- Bit 1 **MF2E:** 多功能中断 2 控制位
0: 除能
1: 使能
- Bit 0 **MF1E:** 多功能中断 1 控制位
0: 除能
1: 使能

• INTC2 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----|------|-------|------|-----|------|-------|------|
| Name | URF | SIMF | INT1F | TB1F | URE | SIME | INT1E | TB1E |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit 7 **URF**: UART 中断请求标志位
0: 无请求
1: 中断请求

Bit 6 **SIMF**: SIM 中断请求标志位
0: 无请求
1: 中断请求

Bit 5 **INT1F**: INT1 中断请求标志位
0: 无请求
1: 中断请求

Bit 4 **TB1F**: 时基 1 中断请求标志位
0: 无请求
1: 中断请求

Bit 3 **URE**: UART 中断控制位
0: 除能
1: 使能

Bit 2 **SIME**: SIM 中断控制位
0: 除能
1: 使能

Bit 1 **INT1E**: INT1 中断控制位
0: 除能
1: 使能

Bit 0 **TB1E**: 时基 1 中断控制位
0: 除能
1: 使能

• MF10 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|-------|-------|---|---|-------|-------|
| Name | — | — | STMAF | STMPF | — | — | STMAE | STMPE |
| R/W | — | — | R/W | R/W | — | — | R/W | R/W |
| POR | — | — | 0 | 0 | — | — | 0 | 0 |

Bit 7~6 未定义，读为“0”

Bit 5 **STMAF**: STM 比较器 A 匹配中断请求标志位
0: 无请求
1: 中断请求

Bit 4 **STMPF**: STM 比较器 P 匹配中断请求标志位
0: 无请求
1: 中断请求

Bit 3~2 未定义，读为“0”

Bit 1 **STMAE**: STM 比较器 A 匹配中断控制位
0: 除能
1: 使能

Bit 0 **STMPE**: STM 比较器 P 匹配中断控制位
0: 除能
1: 使能

● MF11 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-------|-------|-------|-------|-------|-------|-------|-------|
| Name | CTMAF | CTMPF | PTMAF | PTMPF | CTMAE | CTMPE | PTMAE | PTMPE |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

- Bit 7 **CTMAF**: CTM 比较器 A 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 6 **CTMPF**: CTM 比较器 P 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 5 **PTMAF**: PTM 比较器 A 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 4 **PTMPF**: PTM 比较器 P 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 3 **CTMAE**: CTM 比较器 A 匹配中断控制位
0: 除能
1: 使能
- Bit 2 **CTMPE**: CTM 比较器 P 匹配中断控制位
0: 除能
1: 使能
- Bit 1 **PTMAE**: PTM 比较器 A 匹配中断控制位
0: 除能
1: 使能
- Bit 0 **PTMPE**: PTM 比较器 P 匹配中断控制位
0: 除能
1: 使能

● MF12 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|-----|-----|---|---|-----|-----|
| Name | — | — | DEF | LVF | — | — | DEE | LVE |
| R/W | — | — | R/W | R/W | — | — | R/W | R/W |
| POR | — | — | 0 | 0 | — | — | 0 | 0 |

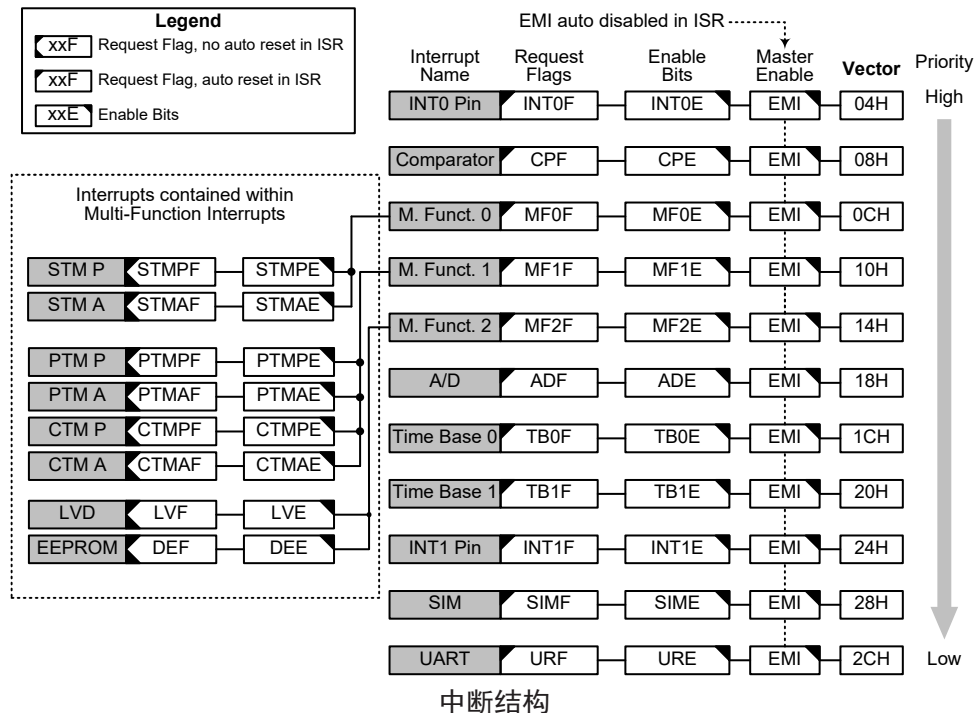
- Bit 7~6 未定义，读为“0”
- Bit 5 **DEF**: 数据 EEPROM 中断请求标志位
0: 无请求
1: 中断请求
- Bit 4 **LVF**: LVD 中断请求标志位
0: 无请求
1: 中断请求
- Bit 3~2 未定义，读为“0”
- Bit 1 **DEE**: 数据 EEPROM 中断控制位
0: 除能
1: 使能
- Bit 0 **LVE**: LVD 中断控制位
0: 除能
1: 使能

中断操作

若中断事件条件产生，如一个 TM 比较器 P、比较器 A 匹配或 A/D 转换结束等等，相关中断请求标志将置起。中断标志产生后程序是否会跳转至相关中断向量执行是由中断使能位的条件决定的。若使能位为“1”，程序将跳至相关中断向量中执行；若使能位为“0”，即使中断请求标志置起中断也不会发生，程序也不会跳转至相关中断向量执行。若总中断使能位为“0”，所有中断都将除能。当中断发生时，下条指令的地址将被压入堆栈。相应的中断向量地址加载至 PC 中。系统将从此向量取下条指令。中断向量处通常为“JMP”指令，以跳转到相应的中断服务程序。中断服务程序必须以“RETI”指令返回至主程序，以继续执行原来的程序。

各个中断使能位以及相应的请求标志位，以优先级的次序显示在下图。一些中断源有自己的向量，但是有些中断却共用多功能中断向量。一旦中断子程序被响应，系统将自动清除 EMI 位，所有其它的中断将被屏蔽，这个方式可以防止任何进一步的中断嵌套。其它中断请求可能发生在此期间，虽然中断不会立即响应，但是中断请求标志位会被记录。

如果某个中断服务子程序正在执行时，有另一个中断要求立即响应，那么 EMI 位应在程序进入中断子程序后置位，以允许此中断嵌套。如果堆栈已满，即使此中断使能，中断请求也不会被响应，直到 SP 减少为止。如果要求立刻动作，则堆栈必须避免成为储满状态。请求同时发生时，执行优先级如下流程图所示。所有被置起的中断请求标志都可把单片机从休眠或空闲模式中唤醒，若要防止唤醒动作发生，在单片机进入休眠或空闲模式前应将相应的标志置起。



外部中断

通过 INT0~INT1 引脚上的信号变化可控制外部中断。当触发沿选择位设置好触发类型，INT0~INT1 引脚的状态发生变化，外部中断请求标志 INT0F~INT1F 被置位时外部中断请求产生。若要跳转到相应中断向量地址，总中断控制位

EMI 和相应中断使能位 INT0E~INT1E 需先被置位。此外，必须使用 INTEG 寄存器使能外部中断功能并选择触发沿类型。外部中断引脚和普通 I/O 口共用，如果相应寄存器中的中断使能位被置位，非 IOM 架构，无引脚共用寄存器，此引脚将被作为外部中断脚使用。此时该引脚必须通过设置端口控制寄存器，将该引脚设置为输入口。当中断使能，堆栈未满并且外部中断脚状态改变，将调用外部中断向量子程序。当响应外部中断服务子程序时，中断请求标志位 INT0F~INT1F 会自动复位且 EMI 位会被清零以除能其它中断。注意，即使此引脚被用作外部中断输入，其上拉电阻仍保持有效。

寄存器 INTEG 被用来选择有效的边沿类型，来触发外部中断。可以选择上升沿还是下降沿或双沿触发都产生外部中断。注意 INTEG 也可以用来除能外部中断功能。

比较器中断

比较器中断由内部比较器控制。当比较器输出状态位改变，比较器中断请求标志 CPF 被置位，比较器中断请求产生。若要跳转到相应中断向量地址，总中断控制位 EMI 和比较器中断使能位 CPE 需先被置位。当中断使能，堆栈未满并且比较器输入产生比较器输出变化时，将调用比较器中断向量子程序。当响应中断服务子程序时，比较器中断请求标志位会自动复位且 EMI 位会被清零以除能其它中断。

多功能中断

此单片机中有多达 3 种多功能中断，与其它中断不同，它没有独立源，但由其它现有的中断源构成，即 TM 中断、LVD 中断和 EEPROM 写中断。

当多功能中断中任何一种中断请求标志 MF_nF 被置位，多功能中断请求产生。当中断使能，堆栈未满，包括在多功能中断中的任意一个中断发生时，将调用多功能中断向量中的一个子程序。当响应中断服务子程序时，相关的多功能请求标志位会自动复位且 EMI 位会自动清零以除能其它中断。

但必须注意的是，在中断响应时，虽然多功能中断标志会自动复位，但多功能中断源的请求标志位，即 TM 中断、LVD 中断、EEPROM 写中断的请求标志位不会自动复位，必须由应用程序清零。

A/D 转换器中断

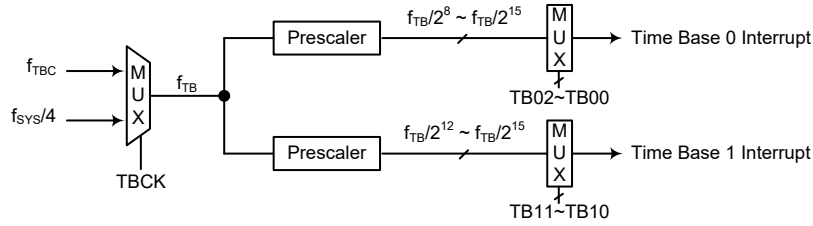
A/D 转换器中断由 A/D 转换动作的结束来控制。当 A/D 转换器中断请求标志 ADF 被置位，即 A/D 转换过程完成时，中断请求发生。若要跳转到相应中断向量地址，总中断控制位 EMI 和 A/D 转换器中断使能位 ADE 需先被置位。当中断使能，堆栈未满且 A/D 转换动作结束时，将调用 A/D 转换器中断向量子程序。当响应中断服务子程序时，相应的中断请求标志位 ADF 会自动复位且 EMI 位会被清零以除能其它中断。

时基中断

时基中断提供一个固定周期的中断信号，由各自的定时器功能产生溢出信号控制。当各自的中断请求标志 TB0F 或 TB1F 被置位时，中断请求发生。当总中断使能位 EMI 和时基使能位 TB0E 或 TB1E 被置位，允许程序跳转到各自的中断向量地址。当中断使能，堆栈未满且时基溢出时，将调用它们各自的中断向量子程序。当响应中断服务子程序时，相应的中断请求标志位 TB0F or TB1F 会自动复位且 EMI 位会被清零以除能其它中断。

时基中断的目的是提供一个固定周期的中断信号。其时钟源来自内部时钟源 f_{TB}。f_{TB} 输入时钟首先经过分频器，分频率由程序设置 TBC 寄存器相关位获取

合适的分频值以提供更长的时基中断周期。控制时基中断周期的时钟 f_{TB} ，可来自不同时钟源，这些时钟源详见系统工作模式章节。



时基中断

• TBC 寄存器

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|------|------|------|------|-------|------|------|------|
| Name | TBON | TBCK | TB11 | TB10 | LXTLP | TB02 | TB01 | TB00 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| POR | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |

- Bit 7 **TBON**: 时基功能使能控制位
0: 除能
1: 使能
- Bit 6 **TBCK**: 时基时钟源选择位
0: f_{TBC}
1: $f_{SYS}/4$
- Bit 5~4 **TB11~TB10**: 时基 1 溢出周期选择位
00: $2^{12}/f_{TB}$
01: $2^{13}/f_{TB}$
10: $2^{14}/f_{TB}$
11: $2^{15}/f_{TB}$
- Bit 3 **LXTLP**: LXT 低功耗控制位
0: 除能 - LXT 快速起振
1: 使能 - LXT 低功耗
- Bit 2~0 **TB02~TB00**: 时基 0 溢出周期选择位
000: $2^8/f_{TB}$
001: $2^9/f_{TB}$
010: $2^{10}/f_{TB}$
011: $2^{11}/f_{TB}$
100: $2^{12}/f_{TB}$
101: $2^{13}/f_{TB}$
110: $2^{14}/f_{TB}$
111: $2^{15}/f_{TB}$

串行接口模块中断

串行接口模块中断，即 SIM 中断，由 SPI 或 I²C 数据传输控制。当一个字节数据已由 SIM 接口接收或发送完，或 I²C 从机地址匹配，或 I²C 总线发生超时，中断请求标志 SIMF 被置位，SIM 中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI、串行接口中断使能位 SIME 需先被置位。当中断使能，堆栈未满足且以上任何一种情况发生时，可跳转至相关 SIM 中断向量子程序中执行。当串行接口中断响应，相应的中断请求标志位 SIMF 会自动复位且 EMI 位会被清零以除能其它中断。

UART 传输中断

UART 传输中断由几种 UART 传输条件来控制。当发送器为空、发送器空闲、接收器数据有效、接收器溢出、地址检测和 RX 引脚唤醒，UART 中断请求标志 URF 被置位，UART 中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI 和 UART 中断使能位 URE 需先被置位。当中断使能，堆栈未满足且以上任何一种情况发生时，将调用 UART 中断向量子程序。当响应中断服务子程序时，相应的中断请求标志位 URF 会自动复位且 EMI 位会被清零以除能其它中断。

LVD 中断

LVD 中断属于多功能中断。当低电压检测功能检测到一个低电压时，LVD 中断请求标志 LVF 被置位，LVD 中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI、低电压中断使能位 LVE 和相应多功能中断使能位需先被置位。当中断使能，堆栈未满足且低电压条件发生时，可跳转至相关多功能中断向量子程序中执行。当 LVD 中断响应，EMI 将被自动清零以除能其它中断，多功能中断请求标志也可自动清除，但 LVF 标志需在应用程序中手动清除。

EEPROM 中断

EEPROM 写中断属于多功能中断。当写周期结束，EEPROM 写中断请求标志 DEF 被置位，EEPROM 写中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI、EEPROM 写中断使能位 DEE 和相应多功能中断使能位需先被置位。当中断使能，堆栈未满足且 EEPROM 写周期结束时，可跳转至相关多功能中断向量子程序中执行。当 EEPROM 写中断响应，EMI 将被自动清零以除能其它中断，多功能中断请求标志也可自动清除，但 DEF 标志需在应用程序中手动清除。

TM 中断

简易型、标准型和周期型 TM 有两个中断，分别来自比较器 P、A 匹配，都属于多功能中断。所有类型的 TM 都有两个中断请求标志位及两个使能位。当 TM 比较器 P、A 匹配情况发生时，相应 TM 中断请求标志被置位，TM 中断请求产生。

若要程序跳转到相应中断向量地址，总中断控制位 EMI、相应 TM 中断使能位和相关多功能中断使能位 MF_nE 需先被置位。当中断使能，堆栈未满足且 TM 比较器匹配情况发生时，可跳转至相关多功能中断向量子程序中执行。当 TM 中断响应，EMI 将被自动清零以除能其它中断，相关 MF_nF 标志也可自动清除，但 TM 中断请求标志需在应用程序中手动清除。

中断唤醒功能

每个中断都具有将处于休眠或空闲模式的单片机唤醒的能力。当中断请求标志由低到高转换时唤醒动作产生，其与中断是否使能无关。因此，尽管单片机处于休眠或空闲模式且系统振荡器停止工作，如有外部中断脚上产生外部边沿跳变、低电压或比较器输入改变都可能导致其相应的中断标志被置位，由此产生中断，因此必须注意避免伪唤醒情况的发生。若中断唤醒功能被除能，单片机进入休眠或空闲模式前相应中断请求标志应被置起。中断唤醒功能不受中断使能位的影响。

编程注意事项

通过禁止相关中断使能位，可以屏蔽中断请求，然而，一旦中断请求标志位被设定，它们会被保留在中断控制寄存器内，直到相应的中断服务子程序执行或请求标志位被应用程序清除。

多功能中断中所含中断相应程序执行时，多功能中断请求标志 MF_nF 可以自动清零，但各自的请求标志需在应用程序中手动清除。

建议在中断服务子程序中不要使用“CALL 子程序”指令。中断通常发生在不可预料的情况或是需要立刻执行的某些应用。假如只剩下一层堆栈且没有控制好中断，当“CALL 子程序”在中断服务子程序中执行时，将破坏原来的控制序列。

所有中断在休眠或空闲模式下都具有唤醒功能，当中断请求标志发生由低到高的转变时都可产生唤醒功能。若要避免相应中断产生唤醒动作，在单片机进入休眠或空闲模式前需先将相应请求标志置为高。

当进入中断服务程序，系统仅将程序计数器的内容压入堆栈，如果中断服务程序会改变状态寄存器或其它的寄存器的内容而破坏控制流程，应事先将这些数据保存起来。

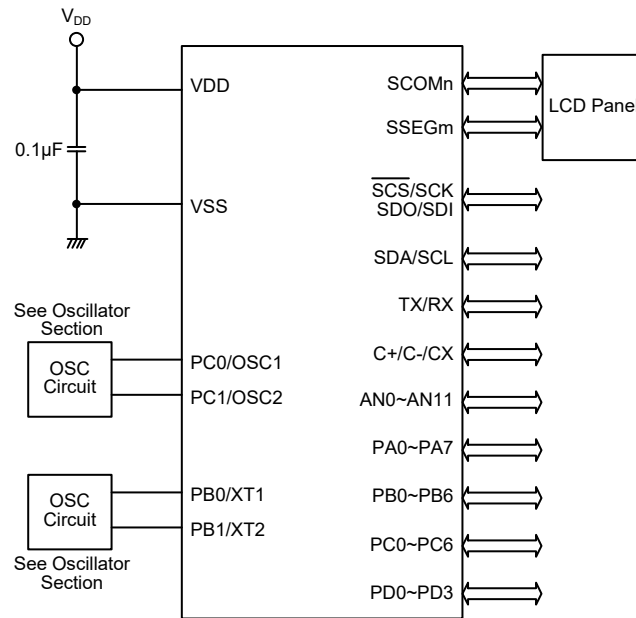
若从中断子程序中返回可执行 RET 或 RETI 指令。除了能返回至主程序外，RETI 指令还能自动设置 EMI 位为高，允许进一步中断。RET 指令只能返回至主程序，清除 EMI 位，除能进一步中断。

配置选项

配置选项在烧写程序时写入芯片。通过 HT-IDE 的软件开发环境，使用者在开发过程中可以选择配置选项。当配置选项烧入单片机后，无法再通过应用程序修改。所有位必须按系统的需要定义，具体内容可参考下表：

| 序号 | 选项 |
|----|--|
| 1 | 高速振荡器类型选择 - f _H : HXT 或 HIRC |
| 2 | 低速振荡器类型选择 - f _{SUB} : LXT 或 LIRC |
| 3 | HIRC 频率选择 -f _{HIRC} : 8MHz、12MHz 或 16MHz |

应用电路



指令集

简介

任何单片机成功运作的核心在于它的指令集，此指令集为一组程序指令码，用来指导单片机如何去执行指定的工作。在 Holtek 单片机中，提供了丰富且灵活的指令，共超过六十条，程序设计者可以事半功倍地实现它们的应用。

为了更加容易理解各种各样的指令码，接下来按功能分组介绍它们。

指令周期

大部分的操作均只需要一个指令周期来执行。分支、调用或查表则需要两个指令周期。一个指令周期相当于四个系统时钟周期，因此如果在 8MHz 的系统时钟振荡器下，大部分的操作将在 0.5 μ s 中执行完成，而分支或调用操作则将在 1 μ s 中执行完成。虽然需要两个指令周期的指令通常指的是 JMP、CALL、RET、RETI 和查表指令，但如果牵涉到程序计数器低字节寄存器 PCL 也将多花费一个周期去加以执行。即指令改变 PCL 的内容进而导致直接跳转至新地址时，需要多一个周期去执行，例如“CLR PCL”或“MOV PCL, A”指令。对于跳转指令必须注意的是，如果比较的结果牵涉到跳转动作将多花费一个周期，如果没有则需一个周期即可。

数据的传送

单片机程序中数据传送是使用最为频繁的操作之一，使用三种 MOV 的指令，数据不但可以从寄存器转移至累加器（反之亦然），而且能够直接移动立即数到累加器。数据传送最重要的应用之一是从输入端口接收数据或传送数据到输出端口。

算术运算

算术运算和数据处理是大部分单片机应用所必需具备的能力，在 Holtek 单片机内部的指令集中，可直接实现加与减的运算。当加法的结果超出 255 或减法的结果少于 0 时，要注意正确的处理进位和借位的问题。INC、INCA、DEC 和 DECA 指令提供了对一个指定地址的值加一或减一的功能。

逻辑和移位运算

标准逻辑运算例如 AND、OR、XOR 和 CPL 全都包含在 Holtek 单片机内部的指令集中。大多数牵涉到数据运算的指令，数据的传送必须通过累加器。在所有逻辑数据运算中，如果运算结果为零，则零标志位将被置位，另外逻辑数据运用形式还有移位指令，例如 RR、RL、RRC 和 RLC 提供了向左或向右移动一位的方法。不同的移位指令可满足不同的应用需要。移位指令常用于串行端口的程序应用，数据可从内部寄存器转移至进位标志位，而此位则可被检验，移位运算还可应用在乘法与除法的运算组成中。

分支和控制转换

程序分支是采取使用 JMP 指令跳转至指定地址或使用 CALL 指令调用子程序的形式，两者之不同在于当子程序被执行完毕后，程序必须马上返回原来的地址。这个动作是由放置在子程序里的返回指令 RET 来实现，它可使程序跳回 CALL 指令之后的地址。在 JMP 指令中，程序则只是跳到一个指定的地址而已，并不需如 CALL 指令般跳回。一个非常有用的分支指令是条件跳转，跳转条件是由数据存储器或指定位来加以决定。遵循跳转条件，程序将继续执行下一条指令或略过且跳转至接下来的指令。这些分支指令是程序走向的关键，跳转条件可能是外部开关输入，或是内部数据位的值。

位运算

提供数据存储器中单个位的运算指令是 Holtek 单片机的特性之一。这特性对于输出端口位的设置尤其有用，其中个别的位或端口的引脚可以使用“SET [m].i”或“CLR [m].i”指令来设定其为高位或低位。如果没有这特性，程序设计师必须先读入输出口的 8 位数据，处理这些数据，然后再输出正确的新数据。这种读入 - 修改 - 写出的过程现在则被位运算指令所取代。

查表运算

数据的储存通常由寄存器完成，然而当处理大量固定的数据时，它的存储量常常造成对个别存储器的不便。为了改善此问题，Holtek 单片机允许在程序存储器中建立一个表格作为数据可直接存储的区域，只需要一组简易的指令即可对数据进行查表。

其它运算

除了上述功能指令外，其它指令还包括用于省电的“HALT”指令和使程序在极端电压或电磁环境下仍能正常工作的看门狗定时器控制指令。这些指令的使用则请查阅相关的章节。

指令集概要

下表中说明了按功能分类的指令集，用户可以将该表作为基本的指令参考。

惯例

x: 立即数
m: 数据存储器地址
A: 累加器
i: 第 0~7 位
addr: 程序存储器地址

| 助记符 | 说明 | 指令周期 | 影响标志位 |
|--------------|------------------------------------|----------------|--------------|
| 算术运算 | | | |
| ADD A,[m] | ACC 与数据存储器相加，结果放入 ACC | 1 | Z, C, AC, OV |
| ADDM A,[m] | ACC 与数据存储器相加，结果放入数据存储器 | 1 ^注 | Z, C, AC, OV |
| ADD A, x | ACC 与立即数相加，结果放入 ACC | 1 | Z, C, AC, OV |
| ADC A,[m] | ACC 与数据存储器、进位标志相加，结果放入 ACC | 1 | Z, C, AC, OV |
| ADCM A,[m] | ACC 与数据存储器、进位标志相加，结果放入数据存储器 | 1 ^注 | Z, C, AC, OV |
| SUB A, x | ACC 与立即数相减，结果放入 ACC | 1 | Z, C, AC, OV |
| SUB A,[m] | ACC 与数据存储器相减，结果放入 ACC | 1 | Z, C, AC, OV |
| SUBM A,[m] | ACC 与数据存储器相减，结果放入数据存储器 | 1 ^注 | Z, C, AC, OV |
| SBC A,[m] | ACC 与数据存储器、进位标志相减，结果放入 ACC | 1 | Z, C, AC, OV |
| SBCM A,[m] | ACC 与数据存储器、进位标志相减，结果放入数据存储器 | 1 ^注 | Z, C, AC, OV |
| DAA [m] | 将加法运算中放入 ACC 的值调整为十进制数，并将结果放入数据存储器 | 1 ^注 | C |
| 逻辑运算 | | | |
| AND A,[m] | ACC 与数据存储器做“与”运算，结果放入 ACC | 1 | Z |
| OR A,[m] | ACC 与数据存储器做“或”运算，结果放入 ACC | 1 | Z |
| XOR A,[m] | ACC 与数据存储器做“异或”运算，结果放入 ACC | 1 | Z |
| ANDM A,[m] | ACC 与数据存储器做“与”运算，结果放入数据存储器 | 1 ^注 | Z |
| ORM A,[m] | ACC 与数据存储器做“或”运算，结果放入数据存储器 | 1 ^注 | Z |
| XORM A,[m] | ACC 与数据存储器做“异或”运算，结果放入数据存储器 | 1 ^注 | Z |
| AND A, x | ACC 与立即数做“与”运算，结果放入 ACC | 1 | Z |
| OR A, x | ACC 与立即数做“或”运算，结果放入 ACC | 1 | Z |
| XOR A, x | ACC 与立即数做“异或”运算，结果放入 ACC | 1 | Z |
| CPL [m] | 对数据存储器取反，结果放入数据存储器 | 1 ^注 | Z |
| CPLA [m] | 对数据存储器取反，结果放入 ACC | 1 | Z |
| 递增和递减 | | | |
| INCA [m] | 递增数据存储器，结果放入 ACC | 1 | Z |
| INC [m] | 递增数据存储器，结果放入数据存储器 | 1 ^注 | Z |
| DECA [m] | 递减数据存储器，结果放入 ACC | 1 | Z |
| DEC [m] | 递减数据存储器，结果放入数据存储器 | 1 ^注 | Z |
| 移位 | | | |
| RRA [m] | 数据存储器右移一位，结果放入 ACC | 1 | 无 |
| RR [m] | 数据存储器右移一位，结果放入数据存储器 | 1 ^注 | 无 |
| RRCA [m] | 带进位将数据存储器右移一位，结果放入 ACC | 1 | C |
| RRC [m] | 带进位将数据存储器右移一位，结果放入数据存储器 | 1 ^注 | C |
| RLA [m] | 数据存储器左移一位，结果放入 ACC | 1 | 无 |

| 助记符 | 说明 | 指令周期 | 影响标志位 |
|-------------|-------------------------------------|----------------|---------|
| RL [m] | 数据存储器左移一位，结果放入数据存储器 | 1 ^注 | 无 |
| RLCA [m] | 带进位将数据存储器左移一位，结果放入 ACC | 1 | C |
| RLC [m] | 带进位将数据存储器左移一位，结果放入数据存储器 | 1 ^注 | C |
| 数据传送 | | | |
| MOV A,[m] | 将数据存储器送至 ACC | 1 | 无 |
| MOV [m],A | 将 ACC 送至数据存储器 | 1 ^注 | 无 |
| MOV A, x | 将立即数送至 ACC | 1 | 无 |
| 位运算 | | | |
| CLR [m].i | 清除数据存储器的位 | 1 ^注 | 无 |
| SET [m].i | 置位数据存储器的位 | 1 ^注 | 无 |
| 转移 | | | |
| JMP addr | 无条件跳转 | 2 | 无 |
| SZ [m] | 如果数据存储器为零，则跳过下一条指令 | 1 ^注 | 无 |
| SZA [m] | 数据存储器送至 ACC，如果内容为零，则跳过下一条指令 | 1 ^注 | 无 |
| SZ [m].i | 如果数据存储器的第 i 位为零，则跳过下一条指令 | 1 ^注 | 无 |
| SNZ [m].i | 如果数据存储器的第 i 位不为零，则跳过下一条指令 | 1 ^注 | 无 |
| SIZ [m] | 递增数据存储器，如果结果为零，则跳过下一条指令 | 1 ^注 | 无 |
| SDZ [m] | 递减数据存储器，如果结果为零，则跳过下一条指令 | 1 ^注 | 无 |
| SIZA [m] | 递增数据存储器，将结果放入 ACC，如果结果为零，则跳过下一条指令 | 1 ^注 | 无 |
| SDZA [m] | 递减数据存储器，将结果放入 ACC，如果结果为零，则跳过下一条指令 | 1 ^注 | 无 |
| CALL addr | 子程序调用 | 2 | 无 |
| RET | 从子程序返回 | 2 | 无 |
| RET A, x | 从子程序返回，并将立即数放入 ACC | 2 | 无 |
| RETI | 从中断返回 | 2 | 无 |
| 查表 | | | |
| TABRD [m] | 读取特定页或当前页的 ROM 内容，并送至数据存储器 and TBLH | 2 ^注 | 无 |
| TABRDL [m] | 读取最后页的 ROM 内容，并送至数据存储器 and TBLH | 2 ^注 | 无 |
| 其它指令 | | | |
| NOP | 空指令 | 1 | 无 |
| CLR [m] | 清除数据存储器 | 1 ^注 | 无 |
| SET [m] | 置位数据存储器 | 1 ^注 | 无 |
| CLR WDT | 清除看门狗定时器 | 1 | TO, PDF |
| SWAP [m] | 交换数据存储器的高低字节，结果放入数据存储器 | 1 ^注 | 无 |
| SWAPA [m] | 交换数据存储器的高低字节，结果放入 ACC | 1 | 无 |
| HALT | 进入暂停模式 | 1 | TO, PDF |

- 注：1. 对跳转指令而言，如果比较的结果牵涉到跳转即需 2 个周期，如果没有发生跳转，则只需一个周期。
2. 任何指令若要改变 PCL 的内容将需要 2 个周期来执行。

指令定义

| | |
|--------------------|---|
| ADC A, [m] | Add Data Memory to ACC with Carry |
| 指令说明 | 将指定的数据存储器、累加器内容以及进位标志相加，结果存放到累加器。 |
| 功能表示 | $ACC \leftarrow ACC + [m] + C$ |
| 影响标志位 | OV、Z、AC、C |
| ADCM A, [m] | Add ACC to Data Memory with Carry |
| 指令说明 | 将指定的数据存储器、累加器内容和进位标志位相加，结果存放到指定的数据存储器。 |
| 功能表示 | $[m] \leftarrow ACC + [m] + C$ |
| 影响标志位 | OV、Z、AC、C |
| ADD A, [m] | Add Data Memory to ACC |
| 指令说明 | 将指定的数据存储器和累加器内容相加，结果存放到累加器。 |
| 功能表示 | $ACC \leftarrow ACC + [m]$ |
| 影响标志位 | OV、Z、AC、C |
| ADD A, x | Add immediate data to ACC |
| 指令说明 | 将累加器和立即数相加，结果存放到累加器。 |
| 功能表示 | $ACC \leftarrow ACC + x$ |
| 影响标志位 | OV、Z、AC、C |
| ADDM A, [m] | Add ACC to Data Memory |
| 指令说明 | 将指定的数据存储器和累加器内容相加，结果存放到指定的数据存储器。 |
| 功能表示 | $[m] \leftarrow ACC + [m]$ |
| 影响标志位 | OV、Z、AC、C |
| AND A, [m] | Logical AND Data Memory to ACC |
| 指令说明 | 将累加器中的数据和指定数据存储器内容做逻辑与，结果存放到累加器。 |
| 功能表示 | $ACC \leftarrow ACC \text{ "AND" } [m]$ |
| 影响标志位 | Z |

| | |
|--------------------|---|
| AND A, x | Logical AND immediate data to ACC |
| 指令说明 | 将累加器中的数据和立即数做逻辑与，结果存放到累加器。 |
| 功能表示 | $ACC \leftarrow ACC \text{ "AND" } x$ |
| 影响标志位 | Z |
| | |
| ANDM A, [m] | Logical AND ACC to Data Memory |
| 指令说明 | 将指定数据存储器内容和累加器中的数据做逻辑与，结果存放到数据存储器。 |
| 功能表示 | $[m] \leftarrow ACC \text{ "AND" } [m]$ |
| 影响标志位 | Z |
| | |
| CALL addr | Subroutine call |
| 指令说明 | 无条件地调用指定地址的子程序，此时程序计数器先加 1 获得下一个要执行的指令地址并压入堆栈，接着载入指定地址并从新地址继续执行程序，由于此指令需要额外的运算，所以为一个 2 周期的指令。 |
| 功能表示 | $Stack \leftarrow Program\ Counter + 1$ $Program\ Counter \leftarrow addr$ |
| 影响标志位 | 无 |
| | |
| CLR [m] | Clear Data Memory |
| 指令说明 | 将指定数据存储器的内容清零。 |
| 功能表示 | $[m] \leftarrow 00H$ |
| 影响标志位 | 无 |
| | |
| CLR [m].i | Clear bit of Data Memory |
| 指令说明 | 将指定数据存储器的第 i 位内容清零。 |
| 功能表示 | $[m].i \leftarrow 0$ |
| 影响标志位 | 无 |
| | |
| CLR WDT | Clear Watchdog Timer |
| 指令说明 | WDT 计数器、暂停标志位 PDF 和看门狗溢出标志位 TO 清零。 |
| 功能表示 | WDT cleared $TO \ \& \ PDF \leftarrow 0$ |
| 影响标志位 | TO、PDF |

| | |
|-----------------|--|
| CPL [m] | Complement Data Memory |
| 指令说明 | 将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或 0 变 1。 |
| 功能表示 | $[m] \leftarrow \overline{[m]}$ |
| 影响标志位 | Z |
| CPLA [m] | Complement Data Memory with result in ACC |
| 指令说明 | 将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或 0 变 1，而结果被储存回累加器且数据存储器中的内容不变。 |
| 功能表示 | $ACC \leftarrow \overline{[m]}$ |
| 影响标志位 | Z |
| DAA [m] | Decimal-Adjust ACC for addition with result in Data Memory |
| 指令说明 | 将累加器中的内容转换为 BCD (二进制转成十进制) 码。如果低四位的值大于“9”或 AC=1，那么 BCD 调整就执行对原值加“6”，否则原值保持不变；如果高四位的值大于“9”或 C=1，那么 BCD 调整就执行对原值加“6”。BCD 转换实质上是根据累加器和标志位执行 00H, 06H, 60H 或 66H 的加法运算，结果存放到数据存储器。只有进位标志位 C 受影响，用来指示原始 BCD 的和是否大于 100，并可以进行双精度十进制数的加法运算。 |
| 功能表示 | $[m] \leftarrow ACC + 00H$ 或 $[m] \leftarrow ACC + 06H$ 或 $[m] \leftarrow ACC + 60H$ 或 $[m] \leftarrow ACC + 66H$ |
| 影响标志位 | C |
| DEC [m] | Decrement Data Memory |
| 指令说明 | 将指定数据存储器内容减 1。 |
| 功能表示 | $[m] \leftarrow [m] - 1$ |
| 影响标志位 | Z |
| DECA [m] | Decrement Data Memory with result in ACC |
| 指令说明 | 将指定数据存储器的内容减 1，把结果存放回累加器并保持指定数据存储器内容不变。 |
| 功能表示 | $ACC \leftarrow [m] - 1$ |
| 影响标志位 | Z |

| | |
|-------------------|---|
| HALT | Enter power down mode |
| 指令说明 | 此指令终止程序执行并关掉系统时钟，RAM 和寄存器的内容保持原状态，WDT 计数器和分频器被清“0”，暂停标志位 PDF 被置位 1，WDT 溢出标志位 TO 被清 0。 |
| 功能表示 | TO ← 0 PDF ← 1 |
| 影响标志位 | TO、PDF |
| | |
| INC [m] | Increment Data Memory |
| 指令说明 | 将指定数据存储器的内容加 1。 |
| 功能表示 | [m] ← [m] + 1 |
| 影响标志位 | Z |
| | |
| INCA [m] | Increment Data Memory with result in ACC |
| 指令说明 | 将指定数据存储器的内容加 1，结果存放回累加器并保持指定的数据存储器内容不变。 |
| 功能表示 | ACC ← [m] + 1 |
| 影响标志位 | Z |
| | |
| JMP addr | Jump unconditionally |
| 指令说明 | 程序计数器的内容无条件地由被指定的地址取代，程序由新的地址继续执行。当新的地址被加载时，必须插入一个空指令周期，所以此指令为 2 个周期的指令。 |
| 功能表示 | Program Counter ← addr |
| 影响标志位 | 无 |
| | |
| MOV A, [m] | Move Data Memory to ACC |
| 指令说明 | 将指定数据存储器的内容复制到累加器。 |
| 功能表示 | ACC ← [m] |
| 影响标志位 | 无 |
| | |
| MOV A, x | Move immediate data to ACC |
| 指令说明 | 将 8 位立即数载入累加器。 |
| 功能表示 | ACC ← x |
| 影响标志位 | 无 |
| | |
| MOV [m], A | Move ACC to Data Memory |
| 指令说明 | 将累加器的内容复制到指定的数据存储器。 |
| 功能表示 | [m] ← ACC |
| 影响标志位 | 无 |

| | |
|-------------------|---|
| NOP | No operation |
| 指令说明 | 空操作，接下来顺序执行下一条指令。 |
| 功能表示 | $PC \leftarrow PC + 1$ |
| 影响标志位 | 无 |
| ORA, [m] | Logical OR Data Memory to ACC |
| 指令说明 | 将累加器中的数据和指定的数据存储器内容逻辑或，结果存放到累加器。 |
| 功能表示 | $ACC \leftarrow ACC \text{ "OR" } [m]$ |
| 影响标志位 | Z |
| ORA, x | Logical OR immediate data to ACC |
| 指令说明 | 将累加器中的数据和立即数逻辑或，结果存放到累加器。 |
| 功能表示 | $ACC \leftarrow ACC \text{ "OR" } x$ |
| 影响标志位 | Z |
| ORM A, [m] | Logical OR ACC to Data Memory |
| 指令说明 | 将存在指定数据存储器中的数据和累加器逻辑或，结果放到数据存储器。 |
| 功能表示 | $[m] \leftarrow ACC \text{ "OR" } [m]$ |
| 影响标志位 | Z |
| RET | Return from subroutine |
| 指令说明 | 将堆栈寄存器中的程序计数器值恢复，程序由取回的地址继续执行。 |
| 功能表示 | Program Counter ← Stack |
| 影响标志位 | 无 |
| RETA, x | Return from subroutine and load immediate data to ACC |
| 指令说明 | 将堆栈寄存器中的程序计数器值恢复且累加器载入指定的立即数，程序由取回的地址继续执行。 |
| 功能表示 | Program Counter ← Stack $ACC \leftarrow x$ |
| 影响标志位 | 无 |

| | |
|------------------|---|
| RETI | Return from interrupt |
| 指令说明 | 将堆栈寄存器中的程序计数器值恢复且中断功能通过设置 EMI 位重新使能。EMI 是控制中断使能的主控制位。如果在执行 RETI 指令之前还有中断未被响应，则这个中断将在返回主程序之前被响应。 |
| 功能表示 | Program Counter ← Stack |
| 影响标志位 | EMI ← 1 无 |
| | |
| RL [m] | Rotate Data Memory left |
| 指令说明 | 将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位。 |
| 功能表示 | [m].(i+1) ← [m].i (i=0~6) [m].0 ← [m].7 |
| 影响标志位 | 无 |
| | |
| RLA [m] | Rotate Data Memory left with result in ACC |
| 指令说明 | 将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位，结果送到累加器，而指定数据存储器的内容保持不变。 |
| 功能表示 | ACC.(i+1) ← [m].i (i=0~6) ACC.0 ← [m].7 |
| 影响标志位 | 无 |
| | |
| RLC [m] | Rotate Data Memory Left through Carry |
| 指令说明 | 将指定数据存储器的内容连同进位标志左移 1 位，第 7 位取代进位标志且原本的进位标志移到第 0 位。 |
| 功能表示 | [m].(i+1) ← [m].i (i=0~6) [m].0 ← C C ← [m].7 |
| 影响标志位 | C |
| | |
| RLC A [m] | Rotate Data Memory left through Carry with result in ACC |
| 指令说明 | 将指定数据存储器的内容连同进位标志左移 1 位，第 7 位取代进位标志且原本的进位标志移到第 0 位，移位结果送回累加器，但是指定数据寄存器的内容保持不变。 |
| 功能表示 | ACC.(i+1) ← [m].i (i=0~6) ACC.0 ← C C ← [m].7 |
| 影响标志位 | C |

| | |
|-------------------|--|
| RR [m] | Rotate Data Memory right |
| 指令说明 | 将指定数据存储器的内容循环右移 1 位且第 0 位移到第 7 位。 |
| 功能表示 | $[m].i \leftarrow [m].(i+1)$ (i=0~6) $[m].7 \leftarrow [m].0$ |
| 影响标志位 | 无 |
| RRA [m] | Rotate Data Memory right with result in ACC |
| 指令说明 | 将指定数据存储器的内容循环右移 1 位，第 0 位移到第 7 位，移位结果存放到累加器，而指定数据存储器的内容保持不变。 |
| 功能表示 | $ACC.i \leftarrow [m].(i+1)$ (i=0~6) $ACC.7 \leftarrow [m].0$ |
| 影响标志位 | 无 |
| RRC [m] | Rotate Data Memory right through Carry |
| 指令说明 | 将指定数据存储器的内容连同进位标志右移 1 位，第 0 位取代进位标志且原本的进位标志移到第 7 位。 |
| 功能表示 | $[m].i \leftarrow [m].(i+1)$ (i=0~6) $[m].7 \leftarrow C$ $C \leftarrow [m].0$ |
| 影响标志位 | C |
| RRCA [m] | Rotate Data Memory right through Carry with result in ACC |
| 指令说明 | 将指定数据存储器的内容连同进位标志右移 1 位，第 0 位取代进位标志且原本的进位标志移到第 7 位，移位结果送回累加器，但是指定数据寄存器的内容保持不变。 |
| 功能表示 | $ACC.i \leftarrow [m].(i+1)$ (i=0~6) $ACC.7 \leftarrow C$ $C \leftarrow [m].0$ |
| 影响标志位 | C |
| SBC A, [m] | Subtract Data Memory from ACC with Carry |
| 指令说明 | 将累加器减去指定数据存储器的内容以及进位标志的反，结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。 |
| 功能表示 | $ACC \leftarrow ACC - [m] - \bar{C}$ |
| 影响标志位 | OV、Z、AC、C、SC、CZ |

| | |
|--------------------|---|
| SBCM A, [m] | Subtract Data Memory from ACC with Carry and result in Data Memory |
| 指令说明 | 将累加器减去指定数据存储器的内容以及进位标志的反，结果存放到数据存储器。如果结果为负，C标志位清除为0，反之结果为正或0，C标志位设置为1。 |
| 功能表示 | $[m] \leftarrow ACC - [m] - \bar{C}$ |
| 影响标志位 | OV、Z、AC、C、SC、CZ |
| | |
| SDZ [m] | Skip if Decrement Data Memory is 0 |
| 指令说明 | 将指定的数据存储器的内容减1，判断是否为0，若为0则跳过下一条指令，由于取得下一个指令时会要求插入一个空指令周期，所以此指令为2个周期的指令。如果结果不为0，则程序继续执行下一条指令。 |
| 功能表示 | $[m] \leftarrow [m] - 1$ ，如果 $[m]=0$ 跳过下一条指令执行 |
| 影响标志位 | 无 |
| | |
| SDZA [m] | Decrement data memory and place result in ACC, skip if 0 |
| 指令说明 | 将指定数据存储器内容减1，判断是否为0，如果为0则跳过下一条指令，此结果将存放到累加器，但指定数据存储器内容不变。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为2个周期的指令。如果结果不为0，则程序继续执行下一条指令。 |
| 功能表示 | $ACC \leftarrow [m] - 1$ ，如果 $ACC=0$ 跳过下一条指令执行 |
| 影响标志位 | 无 |
| | |
| SET [m] | Set Data Memory |
| 指令说明 | 将指定数据存储器的每一位设置为1。 |
| 功能表示 | $[m] \leftarrow FFH$ |
| 影响标志位 | 无 |
| | |
| SET [m].i | Set bit of Data Memory |
| 指令说明 | 将指定数据存储器的第 i 位置位为1。 |
| 功能表示 | $[m].i \leftarrow 1$ |
| 影响标志位 | 无 |

| | |
|----------------------------|---|
| SIZ [m] 指令说明 | Skip if increment Data Memory is 0 将指定的数据存储器的内容加 1，判断是否为 0，若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。 |
| 功能表示 | $[m] \leftarrow [m] + 1$ ，如果 $[m]=0$ 跳过下一条指令执行 |
| 影响标志位 | 无 |
| SIZA [m] 指令说明 | Skip if increment Data Memory is zero with result in ACC 将指定数据存储器的内容加 1，判断是否为 0，如果为 0 则跳过下一条指令，此结果会被存放到累加器，但是指定数据存储器的内容不变。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。 |
| 功能表示 | $ACC \leftarrow [m] + 1$ ，如果 $ACC=0$ 跳过下一条指令执行 |
| 影响标志位 | 无 |
| SNZ [m].i 指令说明 | Skip if bit i of Data Memory is not 0 判断指定数据存储器的第 i 位，若不为 0，则程序跳过下一条指令执行。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果为 0，则程序继续执行下一条指令。 |
| 功能表示 | 如果 $[m].i \neq 0$ ，跳过下一条指令执行 |
| 影响标志位 | 无 |
| SUB A, [m] 指令说明 | Subtract Data Memory from ACC 将累加器的内容减去指定的数据存储器的数据，把结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。 |
| 功能表示 | $ACC \leftarrow ACC - [m]$ |
| 影响标志位 | OV、Z、AC、C、SC、CZ |
| SUBM A, [m] 指令说明 | Subtract Data Memory from ACC with result in Data Memory 将累加器的内容减去指定数据存储器的数据，结果存放到指定的数据存储器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。 |
| 功能表示 | $[m] \leftarrow ACC - [m]$ |
| 影响标志位 | OV、Z、AC、C、SC、CZ |

| | |
|---|--|
| <p>SUB A, x 指令说明</p> <p>功能表示</p> <p>影响标志位</p> | <p>Subtract immediate Data from ACC</p> <p>将累加器的内容减去立即数，结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。</p> <p>$ACC \leftarrow ACC - x$</p> <p>OV、Z、AC、C、SC、CZ</p> |
| <p>SWAP [m] 指令说明</p> <p>功能表示</p> <p>影响标志位</p> | <p>Swap nibbles of Data Memory</p> <p>将指定数据存储器的低 4 位和高 4 位互相交换。</p> <p>$[m].3 \sim [m].0 \leftrightarrow [m].7 \sim [m].4$</p> <p>无</p> |
| <p>SWAPA [m] 指令说明</p> <p>功能表示</p> <p>影响标志位</p> | <p>Swap nibbles of Data Memory with result in ACC</p> <p>将指定数据存储器的低 4 位与高 4 位互相交换，再将结果存放到累加器且指定数据寄存器的数据保持不变。</p> <p>$ACC.3 \sim ACC.0 \leftarrow [m].7 \sim [m].4$</p> <p>$ACC.7 \sim ACC.4 \leftarrow [m].3 \sim [m].0$</p> <p>无</p> |
| <p>SZ [m] 指令说明</p> <p>功能表示</p> <p>影响标志位</p> | <p>Skip if Data Memory is 0</p> <p>判断指定数据存储器的内容是否为 0，若为 0，则程序跳过下一条指令执行。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。</p> <p>如果 $[m]=0$，跳过下一条指令执行</p> <p>无</p> |
| <p>SZA [m] 指令说明</p> <p>功能表示</p> <p>影响标志位</p> | <p>Skip if Data Memory is 0 with data movement to ACC</p> <p>将指定数据存储器内容复制到累加器，并判断指定数据存储器的内容是否为 0，若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。</p> <p>$ACC \leftarrow [m]$，如果 $[m]=0$，跳过下一条指令执行</p> <p>无</p> |

| | |
|----------------------------|--|
| SZ [m].i 指令说明 | Skip if bit i of Data Memory is 0 判断指定数据存储器的第 i 位是否为 0，若为 0，则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。 |
| 功能表示 | 如果 [m].i=0，跳过下一条指令执行 |
| 影响标志位 | 无 |
| TABRD [m] 指令说明 | Read table (specific page or current page) to TBLH and Data Memory 将表格指针 (TBHP 和 TBLP，若无 TBHP 则仅 TBLP) 所指的程序代码低字节移至指定数据存储器且将高字节移至 TBLH。 |
| 功能表示 | [m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节) |
| 影响标志位 | 无 |
| TABRDL [m] 指令说明 | Read table (last page) to TBLH and Data Memory 将表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定数据存储器且将高字节移至 TBLH。 |
| 功能表示 | [m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节) |
| 影响标志位 | 无 |
| XOR A, [m] 指令说明 | Logical XOR Data Memory to ACC 将累加器的数据和指定的数据存储器内容逻辑异或，结果存放到累加器。 |
| 功能表示 | ACC ← ACC “XOR” [m] |
| 影响标志位 | Z |
| XORM A, [m] 指令说明 | Logical XOR ACC to Data Memory 将累加器的数据和指定的数据存储器内容逻辑异或，结果放到数据存储器。 |
| 功能表示 | [m] ← ACC “XOR” [m] |
| 影响标志位 | Z |
| XOR A, x 指令说明 | Logical XOR immediate data to ACC 将累加器的数据与立即数逻辑异或，结果存放到累加器。 |
| 功能表示 | ACC ← ACC “XOR” x |
| 影响标志位 | Z |

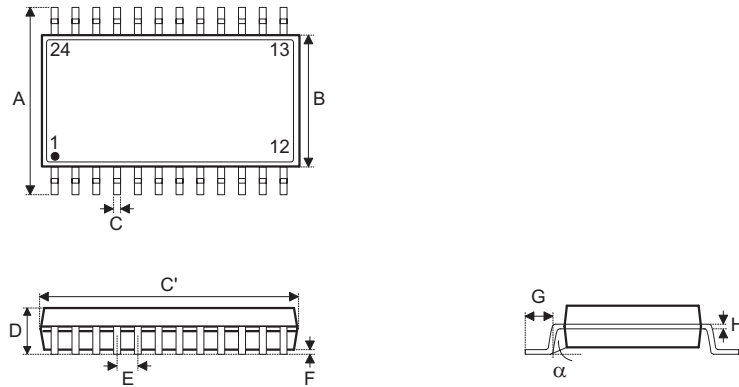
封装信息

请注意，这里提供的封装信息仅作为参考。由于这个信息经常更新，提醒用户咨询 [Holtek 网站](#) 以获取最新版本的[封装信息](#)。

封装信息的相关内容如下所示，点击可链接至 Holtek 网站相关信息页面。

- 封装信息 (包括外形尺寸、包装带和卷轴规格)
- 封装材料信息
- 纸箱信息

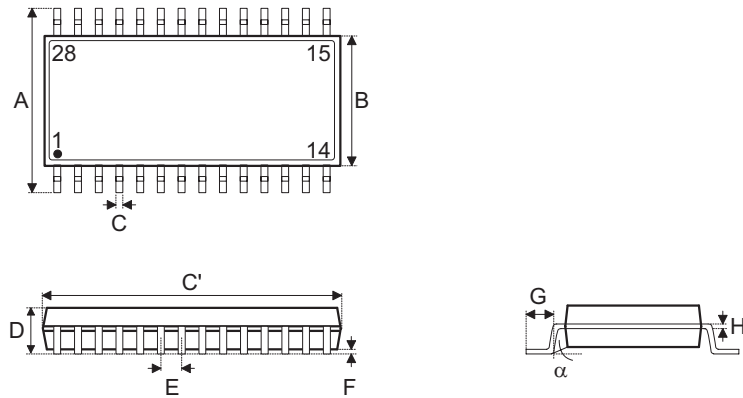
24-pin SSOP (150mil) 外形尺寸



| 符号 | 尺寸 (单位: inch) | | |
|----------|---------------|-----------|-------|
| | 最小值 | 典型值 | 最大值 |
| A | — | 0.236 BSC | — |
| B | — | 0.154 BSC | — |
| C | 0.008 | — | 0.012 |
| C' | — | 0.341 BSC | — |
| D | — | — | 0.069 |
| E | — | 0.025 BSC | — |
| F | 0.004 | — | 0.010 |
| G | 0.016 | — | 0.050 |
| H | 0.004 | — | 0.010 |
| α | 0° | — | 8° |

| 符号 | 尺寸 (单位: mm) | | |
|----------|-------------|-----------|------|
| | 最小值 | 典型值 | 最大值 |
| A | — | 6.00 BSC | — |
| B | — | 3.90 BSC | — |
| C | 0.20 | — | 0.30 |
| C' | — | 8.66 BSC | — |
| D | — | — | 1.75 |
| E | — | 0.635 BSC | — |
| F | 0.10 | — | 0.25 |
| G | 0.41 | — | 1.27 |
| H | 0.10 | — | 0.25 |
| α | 0° | — | 8° |

28-pin SSOP (150mil) 外形尺寸



| 符号 | 尺寸 (单位: inch) | | |
|----------|---------------|-----------|-------|
| | 最小值 | 典型值 | 最大值 |
| A | — | 0.236 BSC | — |
| B | — | 0.154 BSC | — |
| C | 0.008 | — | 0.012 |
| C' | — | 0.390 BSC | — |
| D | — | — | 0.069 |
| E | — | 0.025 BSC | — |
| F | 0.004 | — | 0.010 |
| G | 0.016 | — | 0.050 |
| H | 0.004 | — | 0.010 |
| α | 0° | — | 8° |

| 符号 | 尺寸 (单位: mm) | | |
|----------|-------------|-----------|------|
| | 最小值 | 典型值 | 最大值 |
| A | — | 6.00 BSC | — |
| B | — | 3.90 BSC | — |
| C | 0.20 | — | 0.30 |
| C' | — | 9.90 BSC | — |
| D | — | — | 1.75 |
| E | — | 0.635 BSC | — |
| F | 0.10 | — | 0.25 |
| G | 0.41 | — | 1.27 |
| H | 0.10 | — | 0.25 |
| α | 0° | — | 8° |

Copyright® 2020 by HOLTEK SEMICONDUCTOR INC.

使用指南中所出现的信息在出版当时相信是正确的，然而 **Holtek** 对于说明书的使用不负任何责任。文中提到的应用目的仅仅是用来做说明，**Holtek** 不保证或表示这些没有进一步修改的应用将是适当的，也不推荐它的产品使用在会由于故障或其它原因可能会对人身造成危害的地方。**Holtek** 产品不授权用于救生、维生从机或系统中做为关键从机。**Holtek** 拥有不事先通知而修改产品的权利，对于最新的信息，请参考我们的网址 <http://www.holtek.com/zh/>。

X-ON Electronics

Largest Supplier of Electrical and Electronic Components

Click to view similar products for [32-bit Microcontrollers - MCU category](#):

Click to view products by [Holtek manufacturer](#):

Other Similar products are found below :

[MCF51AC256AVFUE](#) [MCF51AC256BCFUE](#) [MCF51AC256BVFUE](#) [MB91F464AAPMC-GSE2](#) [R5S726B0D216FP#V0](#) [MB91F248PFV-GE1](#) [MB91243PFV-GS-136E1](#) [SAK-TC1782F-320F180HR BA](#) [TC364DP64F300WAAKXUMA1](#) [R5F566NNDDFP#30](#)
[R5F566NNDDFC#30](#) [R5F566NNDDBD#20](#) [MC96F8216ADBN](#) [A96G181HDN](#) [A96G140KNN](#) [A96G174FDN](#) [A31G213CL2N](#)
[A96G148KNN](#) [A96G174AEN](#) [AC33M3064TLBN-01](#) [V3s](#) [T3](#) [A40i-H](#) [V526](#) [A83T](#) [R11](#) [V851s](#) [A133](#) [V833](#) [F1C100S](#) [T3L](#) [T507](#) [A33](#)
[A63](#) [T113-i](#) [H616](#) [V853](#) [V533](#) [R16-J](#) [V536-H](#) [A64-H](#) [V831](#) [V3LP](#) [T113-S3](#) [F1C200S](#) [F133-A](#) [R128-S2](#) [D1-H](#) [ADUCM360BCPZ128-TR](#)
[APT32S003F8PT](#)