触控式 Sub-1GHz OOK/FSK RF 发射器 Flash 单片机

BC66F2235/BC66F2245/BC66F2255

版本：V1.20　日期：2024-01-08

www.holtek.com

目录

特性

CPU 特性

- 工作电压：
  - ♦ f_{SYS}=8MHz：2.0V~3.6V
  - ♦ f_{SYS}=16MHz：2.0V~3.6V
- V_{DD}=5V，系统时钟为 16MHz 时，指令周期为 0.25μs
- 提供省电和唤醒功能，以降低功耗
- 振荡器类型：
  - ♦ RF 外部高速晶振 – HXT
  - ♦ 内部高速 8MHz RC – HIRC
  - ♦ 内部低速 32kHz RC – LIRC
- 多种工作模式：快速、低速、空闲、休眠和深度休眠
- 完全集成的内部振荡器，无需外接元件
- 所有指令都可在 1~3 个指令周期内完成
- 查表指令
- 115 条指令
- 8 层堆栈
- 位操作指令

周边特性

- Flash 程序存储器：2K×16 ~ 8K×16
- RAM 数据存储器：256×8
- 触控按键数据存储器：96×8
- 多达 16 个触控按键功能 – 完全集成而无需外接元件
- 在线应用编程功能 – IAP
- 内部片上调试功能 – OCDS
- 看门狗定时器功能
- 多达 23 个双向 I/O 口
- 多达 2 个与 I/O 口复用的外部中断输入
- 多个定时器模块用于时间测量、比较匹配输出、PWM 输出及单脉冲输出
  - ♦ 2 个 10-bit 简易型 TM – CTM0~CTM1
  - ♦ 1 个 10-bit 周期性 TM – PTM0
- 通用串行接口模块 – USIM，用于 SPI、I^2C 或 UART 通信
- 双时基功能，用于产生固定时间的中断信号
- 多达 4 个外部通道 12-bit 分辨率的 A/D 转换器
- 内建 1.5V LDO
- 低电压复位功能
- 低电压检测功能
- 封装类型：16-pin NSOP-EP，24-pin SSOP-EP，32-pin QFN

RF 发射器特性

- 带传送相位低噪声的完整超高频 OOK/FSK 发射器
- 支持 315/433/868/915MHz 频带
- 小于 2kHz 分辨率的可编程通道设置
- OOK 符号传送速率 0.5Ksps~25Ksps，FSK 数据传送速率 0.5Kbps~100Kbps
- 输出功率高达 13dBm (可软件控制输出功率：0dBm，+10dBm，+13dBm)
- RF 外部晶振：16MHz

概述

该系列单片机是一款 A/D 型具有 8 位高性能精简指令集的 Flash 单片机，且完全集成触控按键功能和 RF 发射器，使其灵活地应用于一系列无线触控应用，如工业控制、消费类产品和子系统控制器等。触控按键功能完全集成于单片机内，无需外部元件，为各种触控按键的应用提供了可靠便捷的实现方法。

在存储器特性方面，Flash 存储器可多次编程的特性给用户提供了较大的方便。此外还包括 RAM 数据存储器。通过使用 Holtek 在线应用编程功能，用户可方便直接地将测量数据存储在 Flash 存储器中，且方便对应用程序进行更新。

在模拟特性方面，该系列单片机包含一个多通道 12-bit A/D 转换器。在内部定时器方面，具有多个使用灵活的定时器模块，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、I^2C 和 UART 接口功能，为设计者提供了一个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内部保护特性，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。

该系列单片机提供了丰富的外部高速和内部高低速振荡器功能选项。内部振荡器完全内建，无需外接元件。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。

集成的 RF 发射器可工作在 315MHz、433MHz、868MHz 和 915MHz 频带。仅需外接一个晶振和少量的外部元件即可构建一个完整通用的 RF 发射器系统。该系列单片机还提供了内部功率放大器，可为 50Ω 负载提供高达 +13dBm 的功率。这样的功率水平可使一个小型的发射器在接近最大传输规限条件下仍可工作。该系列单片机可与 OOK—On-Off Keying (开关键控) 和 FSK—Frequency Shift Keying (频移键控) 两种类型接收器搭配工作。FSK 数据传送速率高达 100Kbps，使单片机可支持更复杂的控制协议。

外加 I/O 使用灵活、时基功能和其它特性确保了该系列单片机可为多种远程无线应用提供一个高性价比的 Sub-1GHz RF 发射器方案。

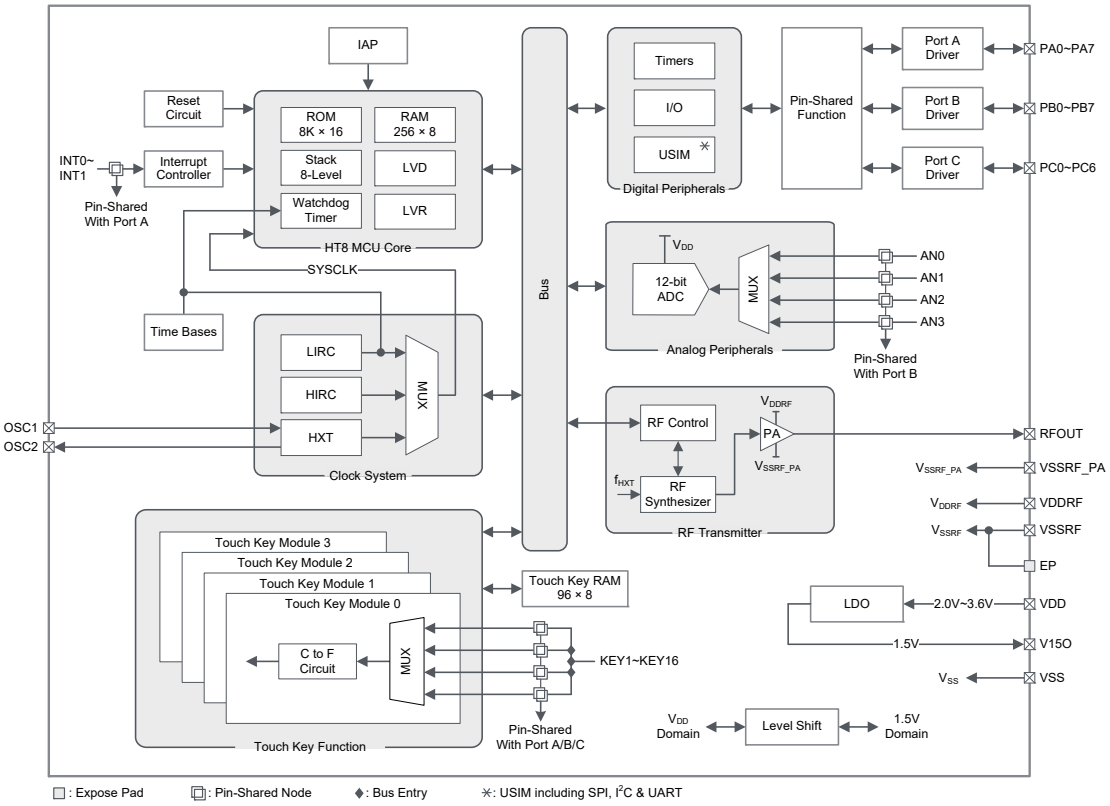
选型表

对此系列的芯片而言，大多数的特性参数都是一样的。主要差异在于程序存储器的容量、I/O 口数目、外部中断数、A/D 通道数、触控按键数和封装类型。下表列出了各单片机的主要特性。

型号	V_{DD}	ROM	RAM	触控按键 RAM	I/O	外部中断	A/D	时基	TM 模块
BC66F2235	2.0V~3.6V	2K×16	256×8	96×8	8	1	12-bit×1	2	10-bit CTM×2 10-bit PTM×1
BC66F2245		4K×16			15	2	12-bit×4		
BC66F2255		8K×16			23	2	12-bit×4		

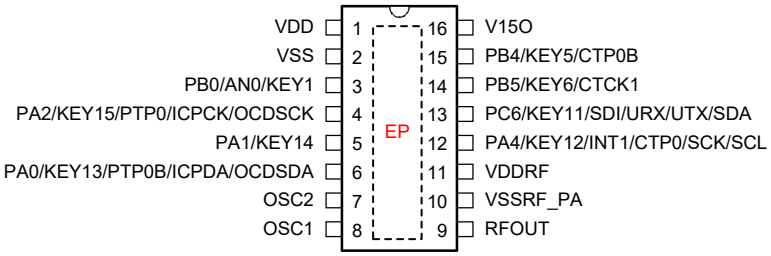
型号	集成 RF		触控按键	USIM (SPI+I²C+UART)	LDO	堆栈	封装
	频带	类型					
BC66F2235	315~915MHz	OOK/FSK TX	8	√	√	8	16NSOP-EP
BC66F2245			14				24SSOP-EP
BC66F2255			16				32QFN

方框图

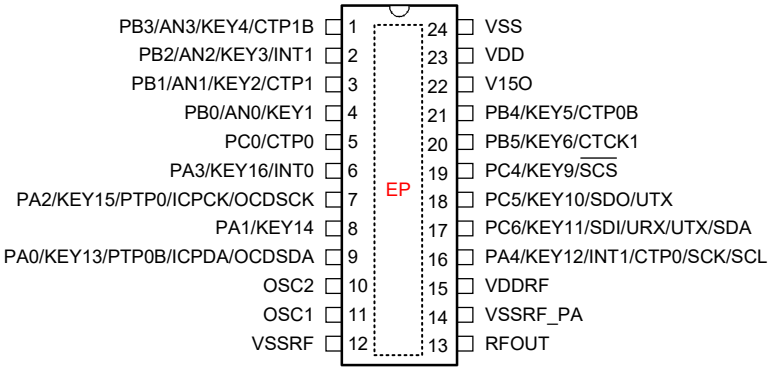


注：该图为此系列具有最大特性的单片机的方框图，单片机间的功能差异详见选型表。

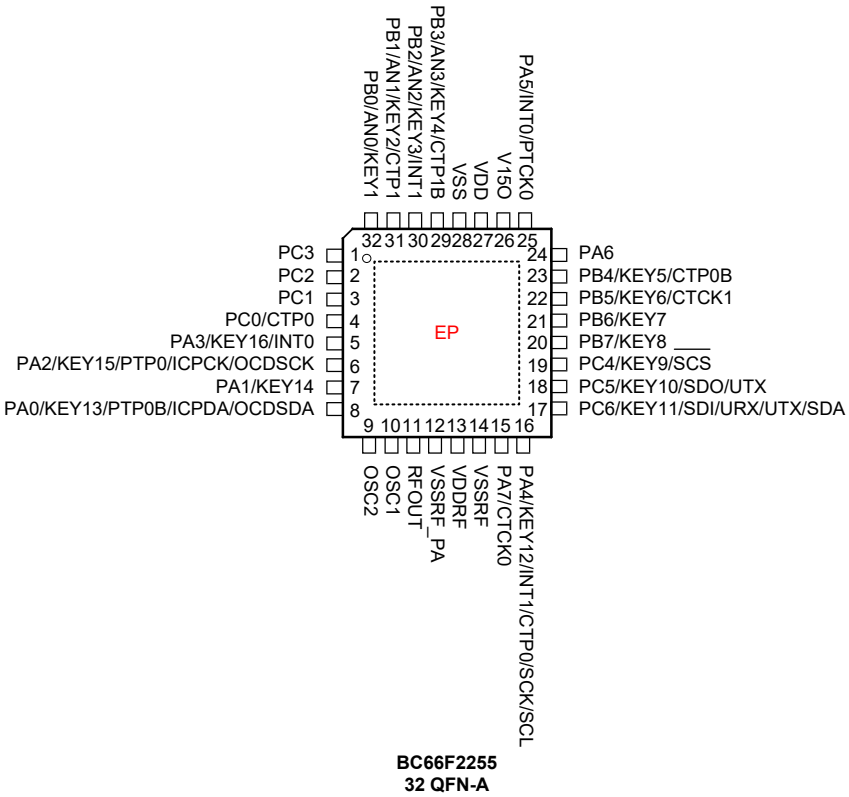
引脚图



BC66F2235
16 NSOP-EP-A



BC66F2245
24 SSOP-EP-A



注：1. 若共用脚同时有多种输出，所需引脚共用功能通过相应的软件控制位决定。
2. OCDSDA 和 OCDSCK 引脚为片上调试功能专用引脚。

引脚说明

每个引脚的功能如下表所述，而引脚配置的详细内容见规格书其它章节。

BC66F2235

引脚名称	功能	OPT	I/T	O/T	说明
PA0/KEY13/PTP0B/ICPDA/OCDSDA	PA0	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY13	PAS0	AN	—	触控按键输入
	PTP0B	PAS0	—	CMOS	PTM0 反相输出
	ICPDA	—	ST	CMOS	ICP 数据 / 地址引脚
	OCDSDA	—	ST	CMOS	OCDS 数据 / 地址引脚
PA1/KEY14	PA1	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY14	PAS0	AN	—	触控按键输入

引脚名称	功能	OPT	I/T	O/T	说明
PA2/KEY15/PTP0/ICPCK/OCDSCK	PA2	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY15	PAS0	AN	—	触控按键输入
	PTP0	PAS0	—	CMOS	PTM0 输出
	ICPCK	—	ST	—	ICP 时钟引脚
	OCDSCK	—	ST	—	OCDS 时钟引脚
PA4/KEY12/INT1/CTP0/SCK/SCL	PA4	PAPU PAWU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY12	PAS1	AN	—	触控按键输入
	INT1	PAS1 IFS INTEG INTC0	ST	—	外部中断 1 输入
	CTP0	PAS1	—	CMOS	CTM0 输出
	SCK	PAS1	ST	CMOS	SPI 串行时钟
	SCL	PAS1	ST	NMOS	I^2C 时钟线
PB0/AN0/KEY1	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN0	PBS0	AN	—	A/D 转换器外部输入通道
	KEY1	PBS0	AN	—	触控按键输入
PB4/KEY5/CTP0B	PB4	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY5	PBS1	AN	—	触控按键输入
	CTP0B	PBS1	—	CMOS	CTM0 反相输出
PB5/KEY6/CTCK1	PB5	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY6	PBS1	AN	—	触控按键输入
	CTCK1	PBS1	ST	—	CTM1 时钟输入
PC6/KEY11/SDI/URX/UTX/SDA	PC6	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY11	PCS1	AN	—	触控按键输入
	SDI	PCS1	ST	—	SPI 串行数据输入
	URX/ UTX	PCS1	ST	CMOS	UART 串行数据输入 (全双工通信)，UART 串行数据输入 / 输出 (单线通信模式)
	SDA	PCS1	ST	NMOS	I^2C 数据线
OSC1	OSC1	—	AN	—	RF 振荡器输入
OSC2	OSC2	—	—	AN	RF 振荡器输出
RFOUT	RFOUT	—	—	AN	RF 功率放大器输出
VDD	VDD	—	PWR	—	数字正电源
V15O	V15O	—	—	PWR	内部 1.5V LDO 输出
VSS	VSS	—	PWR	—	数字负电源

引脚名称	功能	OPT	I/T	O/T	说明
VDDRF	VDDRF	—	PWR	—	RF 正电源
VSSRF_PA	VSSRF_PA	—	PWR	—	RF 功率放大器负电源
EP	EP	—	PWR	—	裸露焊盘，必须接地

注：I/T：输入类型；
OPT：通过寄存器选项来配置；
ST：施密特触发输入；
NMOS：NMOS 输出；
O/T：输出类型；
PWR：电源；
CMOS：CMOS 输出；
AN：模拟信号。

BC66F2245

引脚名称	功能	OPT	I/T	O/T	说明
PA0/KEY13/PTP0B/ICPDA/OCDSDA	PA0	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY13	PAS0	AN	—	触控按键输入
	PTP0B	PAS0	—	CMOS	PTM0 反相输出
	ICPDA	—	ST	CMOS	ICP 数据 / 地址引脚
	OCDSDA	—	ST	CMOS	OCDS 数据 / 地址引脚
PA1/KEY14	PA1	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY14	PAS0	AN	—	触控按键输入
PA2/KEY15/PTP0/ICPCK/OCDSCK	PA2	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY15	PAS0	AN	—	触控按键输入
	PTP0	PAS0	—	CMOS	PTM0 输出
	ICPCK	—	ST	—	ICP 时钟引脚
	OCDSCK	—	ST	—	OCDS 时钟引脚
PA3/KEY16/INT0	PA3	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY16	PAS0	AN	—	触控按键输入
	INT0	PAS0 IFS INTEG INTC0	ST	—	外部中断 0 输入

引脚名称	功能	OPT	I/T	O/T	说明
PA4/KEY12/INT1/CTP0/SCK/SCL	PA4	PAPU PAWU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY12	PAS1	AN	—	触控按键输入
	INT1	PAS1 IFS INTEG INTC0	ST	—	外部中断 1 输入
	CTP0	PAS1	—	CMOS	CTM0 输出
	SCK	PAS1	ST	CMOS	SPI 串行时钟
	SCL	PAS1	ST	NMOS	I^2C 时钟线
PB0/AN0/KEY1	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN0	PBS0	AN	—	A/D 转换器外部输入通道
	KEY1	PBS0	AN	—	触控按键输入
PB1/AN1/KEY2/CTP1	PB1	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN1	PBS0	AN	—	A/D 转换器外部输入通道
	KEY2	PBS0	AN	—	触控按键输入
	CTP1	PBS0	—	CMOS	CTM1 输出
PB2/AN2/KEY3/INT1	PB2	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN2	PBS0	AN	—	A/D 转换器外部输入通道
	KEY3	PBS0	AN	—	触控按键输入
	INT1	PBS0 IFS INTEG INTC0	ST	—	外部中断 1 输入
PB3/AN3/KEY4/CTP1B	PB3	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN3	PBS0	AN	—	A/D 转换器外部输入通道
	KEY4	PBS0	AN	—	触控按键输入
	CTP1B	PBS0	—	CMOS	CTM1 反相输出
PB4/KEY5/CTP0B	PB4	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY5	PBS1	AN	—	触控按键输入
	CTP0B	PBS1	—	CMOS	CTM0 反相输出
PB5/KEY6/CTCK1	PB5	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY6	PBS1	AN	—	触控按键输入
	CTCK1	PBS1	ST	—	CTM1 时钟输入
PC0/CTP0	PC0	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CTP0	PCS0	—	CMOS	CTM0 输出

引脚名称	功能	OPT	I/T	O/T	说明
PC4/KEY9/$\overline{SCS}$	PC4	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY9	PCS1	AN	—	触控按键输入
	$\overline{SCS}$	PCS1	ST	CMOS	SPI 从机选择
PC5/KEY10/SDO/ UTX	PC5	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY10	PCS1	AN	—	触控按键输入
	SDO	PCS1	—	CMOS	SPI 串行数据输出
	UTX	PCS1	—	CMOS	UART 串行数据输出
PC6/KEY11/SDI/ URX/UTX/SDA	PC6	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY11	PCS1	AN	—	触控按键输入
	SDI	PCS1	ST	—	SPI 串行数据输入
	URX/ UTX	PCS1	ST	CMOS	UART 串行数据输入 (全双工通信), UART 串行数据输入 / 输出 (单线通信模式)
	SDA	PCS1	ST	NMOS	I^2C 数据线
OSC1	OSC1	—	AN	—	RF 振荡器输入
OSC2	OSC2	—	—	AN	RF 振荡器输出
RFOUT	RFOUT	—	—	AN	RF 功率放大器输出
VDD	VDD	—	PWR	—	数字正电源
V15O	V15O	—	—	PWR	内部 1.5V LDO 输出
VSS	VSS	—	PWR	—	数字负电源
VDDRF	VDDRF	—	PWR	—	RF 正电源
VSSRF	VSSRF	—	PWR	—	RF 负电源
VSSRF_PA	VSSRF_ PA	—	PWR	—	RF 功率放大器负电源
EP	EP	—	PWR	—	裸露焊盘，必须接地

注：I/T：输入类型；
OPT：通过寄存器选项来配置；
ST：施密特触发输入；
NMOS：NMOS 输出；
O/T：输出类型；
PWR：电源；
CMOS：CMOS 输出；
AN：模拟信号。

BC66F2255

引脚名称	功能	OPT	I/T	O/T	说明
PA0/KEY13/PTP0B/ICPDA/OCDSDA	PA0	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY13	PAS0	AN	—	触控按键输入
	PTP0B	PAS0	—	CMOS	PTM0 反相输出
	ICPDA	—	ST	CMOS	ICP 数据 / 地址引脚
	OCDSDA	—	ST	CMOS	OCDS 数据 / 地址引脚
PA1/KEY14	PA1	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY14	PAS0	AN	—	触控按键输入
PA2/KEY15/PTP0/ICPCK/OCDSCK	PA2	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY15	PAS0	AN	—	触控按键输入
	PTP0	PAS0	—	CMOS	PTM0 输出
	ICPCK	—	ST	—	ICP 时钟引脚
	OCDSCK	—	ST	—	OCDS 时钟引脚
PA3/KEY16/INT0	PA3	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY16	PAS0	AN	—	触控按键输入
	INT0	PAS0 IFS INTEG INTC0	ST	—	外部中断 0 输入
PA4/KEY12/INT1/CTP0/SCK/SCL	PA4	PAPU PAWU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	KEY12	PAS1	AN	—	触控按键输入
	INT1	PAS1 IFS INTEG INTC0	ST	—	外部中断 1 输入
	CTP0	PAS1	—	CMOS	CTM0 输出
	SCK	PAS1	ST	CMOS	SPI 串行时钟
	SCL	PAS1	ST	NMOS	I^2C 时钟线
PA5/INT0/PTCK0	PA5	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	INT0	IFS INTEG INTC0	ST	—	外部中断 0 输入
	PTCK0	—	ST	—	PTM0 时钟输入

引脚名称	功能	OPT	I/T	O/T	说明
PA6	PA6	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
PA7/CTCK0	PA7	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	CTCK0	—	ST	—	CTM0 时钟输入
PB0/AN0/KEY1	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN0	PBS0	AN	—	A/D 转换器外部输入通道
	KEY1	PBS0	AN	—	触控按键输入
PB1/AN1/KEY2/ CTP1	PB1	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN1	PBS0	AN	—	A/D 转换器外部输入通道
	KEY2	PBS0	AN	—	触控按键输入
	CTP1	PBS0	—	CMOS	CTM1 输出
PB2/AN2/KEY3/INT1	PB2	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN2	PBS0	AN	—	A/D 转换器外部输入通道
	KEY3	PBS0	AN	—	触控按键输入
	INT1	PBS0 IFS INTEG INTC0	ST	—	外部中断 1 输入
PB3/AN3/KEY4/ CTP1B	PB3	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN3	PBS0	AN	—	A/D 转换器外部输入通道
	KEY4	PBS0	AN	—	触控按键输入
	CTP1B	PBS0	—	CMOS	CTM1 反相输出
PB4/KEY5/CTP0B	PB4	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY5	PBS1	AN	—	触控按键输入
	CTP0B	PBS1	—	CMOS	CTM0 反相输出
PB5/KEY6/CTCK1	PB5	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY6	PBS1	AN	—	触控按键输入
	CTCK1	PBS1	ST	—	CTM1 时钟输入
PB6/KEY7	PB6	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY7	PBS1	AN	—	触控按键输入
PB7/KEY8	PB7	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY8	PBS1	AN	—	触控按键输入
PC0/CTP0	PC0	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CTP0	PCS0	—	CMOS	CTM0 输出

引脚名称	功能	OPT	I/T	O/T	说明
PC1~PC3	PCn	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
PC4/KEY9/$\overline{SCS}$	PC4	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY9	PCS1	AN	—	触控按键输入
	$\overline{SCS}$	PCS1	ST	CMOS	SPI 从机选择
PC5/KEY10/SDO/ UTX	PC5	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY10	PCS1	AN	—	触控按键输入
	SDO	PCS1	—	CMOS	SPI 串行数据输出
	UTX	PCS1	—	CMOS	UART 串行数据输出
PC6/KEY11/SDI/ URX/UTX/SDA	PC6	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY11	PCS1	AN	—	触控按键输入
	SDI	PCS1	ST	—	SPI 串行数据输入
	URX/ UTX	PCS1	ST	CMOS	UART 串行数据输入 (全双工通信), UART 串行数据输入 / 输出 (单线通信模式)
	SDA	PCS1	ST	NMOS	I^2C 数据线
OSC1	OSC1	—	AN	—	RF 振荡器输入
OSC2	OSC2	—	—	AN	RF 振荡器输出
RFOUT	RFOUT	—	—	AN	RF 功率放大器输出
VDD	VDD	—	PWR	—	数字正电源
V15O	V15O	—	—	PWR	内部 1.5V LDO 输出
VSS	VSS	—	PWR	—	数字负电源
VDDRF	VDDRF	—	PWR	—	RF 正电源
VSSRF	VSSRF	—	PWR	—	RF 负电源
VSSRF_PA	VSSRF_ PA	—	PWR	—	RF 功率放大器负电源
EP	EP	—	PWR	—	裸露焊盘，必须接地

注：I/T：输入类型；
OPT：通过寄存器选项来配置；
ST：施密特触发输入；
NMOS：NMOS 输出；
O/T：输出类型；
PWR：电源；
CMOS：CMOS 输出；
AN：模拟信号。

极限参数

电源供应电压 ..... V_{SS}-0.3V~3.6V
端口输入电压 ..... V_{SS}-0.3V~V_{DD}+0.3V
储存温度 ..... -60°C~150°C
工作温度 ..... -40°C~85°C
I_{OH} 总电流 ..... -80mA
I_{OL} 总电流..... 80mA
总功耗 500mW
ESD HBM ..... ±2kV
ESD MM ..... ±400V

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。

该系列单片机为 ESD 感应。人体模式 HBM (Human Body Mode) 基于标准 MIL-STD-883H Method 3015.8。机器模式 MM (Machine Mode) 基于 JEDEC EIA/JESD22-A115。

直流电气特性

以下表格中参数测量结果可能受多个因素影响，如振荡器类型、工作电压、工作频率、引脚负载状况、温度和程序指令等等。

工作电压特性

Ta=-40°C~85°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	温度				
V_{DD}	MCU 工作电压 – HXT (RF 外部晶振)	—	$f_{SYS}=f_{HXT}$=16MHz	2.0	—	3.6	V
	MCU 工作电压 – HIRC	—	$f_{SYS}=f_{HIRC}$=8MHz	2.0	—	3.6	V
	MCU 工作电压 – LIRC	—	$f_{SYS}=f_{LIRC}$=32kHz	2.0	—	3.6	V
V_{15O}	LDO 工作电压	2.0V~3.6V	—	-10%	1.5	+10%	V

工作电流特性

Ta=-40°C~85°C

符号	工作模式	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{DD}	低速模式 – LIRC	3V	$f_{SYS}=f_{LIRC}$=32kHz，RF TX 电源关闭，LCMD=0	—	42	70	μA
		3V	$f_{SYS}=f_{LIRC}$=32kHz，RF TX 电源关闭，LCMD=1	—	30	50	μA
	快速模式 – HIRC	3V	$f_{SYS}=f_{HIRC}$=8MHz，RF TX 电源关闭	—	0.76	1.50	mA
	快速模式 – HXT	3V	f_{SUB} on，$f_{SYS}=f_{HXT}$=16MHz，RF TX 电源关闭	—	1.1	1.6	mA

注：当使用该表格电气特性数据时，以下几点需注意：
1. 任何数字输入都设置为非浮空的状态。
2. 所有测量都在无负载且所有外围功能关闭的条件下进行。
3. 无直流电流路径。
4. 所有工作电流数值都是通过连续的 NOP 指令循环测得。

待机电流特性

Ta=25°C，除非另有说明

符号	待机模式	测试条件		最小	典型	最大	最大 @85°C	单位
		V_{DD}	条件					
I_{STB}	深度休眠模式 (LDO Off)	3V	f_{LIRC} off	—	0.4	1.0	1.0	μA
		3V	f_{LIRC} on	—	1.2	3.0	3.0	μA
	空闲模式 1 – HXT (LDO On)	3V	f_{SUB} on，$f_{SYS}=f_{HXT}$=16MHz	—	600	900	900	μA

注：当使用该表格电气特性数据时，以下几点需注意：
1. 任何数字输入都设置为非浮空的状态。
2. 所有测量都在无负载且所有外围功能关闭的条件下进行。
3. 无直流电流路径。
4. 所有待机电流数值都是在 HALT 指令执行后即停止执行所有指令后测得。

交流电气特性

以下表格中参数测量结果可能受多个因素影响，如振荡器类型、工作电压、工作频率和温度等等。

内部高速 RC 振荡器 HIRC 频率精准度

程序烧录时，烧录器会依据用户选择的 HIRC 频率和工作电压 (3V) 对 HIRC 进行频率精准度调整。

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	温度				
f_{HIRC}	通过烧录器调整后的 8MHz HIRC 频率	3V	25°C	-2%	8	+2%	MHz
		3V±0.1V	0°C~70°C	-5%	8	+5%	
		2.0V~3.6V	0°C~70°C	-7%	8	+7%	
		2.0V~3.6V	-40°C~85°C	-10%	8	+10%	

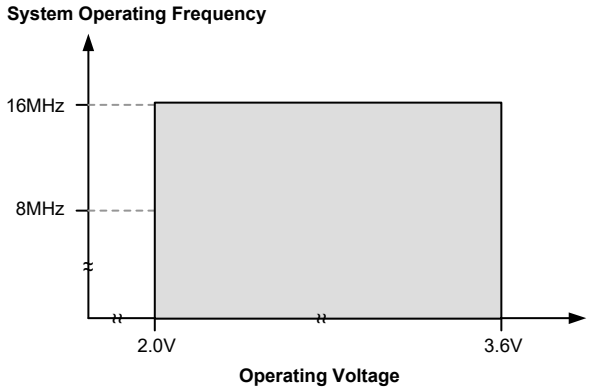
注：1. 烧录器可在 3V 这个固定电压下对 HIRC 频率进行调整，在此提供 V_{DD}=3V 时的参数值。

2. 3V 表格列下面提供的是全压条件下的参数值。对于电压范围在 2.0V~3.6V 的应用，建议烧录器电压固定在 3V。

内部低速振荡器 LIRC 电气特性

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	温度				
f_{LIRC}	LIRC 频率	3V	25°C	-10%	32	+10%	kHz
t_{START}	LIRC 启动时间	—	-40°C~85°C	—	—	500	μs

工作频率电气特性曲线图



系统上电时间电气特性

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				
t_{SST}	系统启动时间 (从 f_{SYS} off 的状态下唤醒)	—	$f_{SYS}=f_H\sim f_H/64$，$f_H=f_{HXT}$	—	128	—	t_{HXT}
			$f_{SYS}=f_H\sim f_H/64$，$f_H=f_{HIRC}$	—	16	—	t_{HIRC}
			$f_{SYS}=f_{SUB}=f_{LIRC}$	—	2	—	t_{LIRC}
	系统启动时间 (从 f_{SYS} on 的状态下唤醒)	—	$f_{SYS}=f_H\sim f_H/64$， $f_H=f_{HXT}$ 或 f_{HIRC}	—	2	—	t_H
			$f_{SYS}=f_{SUB}=f_{LIRC}$	—	2	—	t_{SUB}
	系统速度切换时间 (快速模式 → 低速模式或 低速模式 → 快速模式)	—	f_{HXT} off → on	—	1024	—	t_{HXT}
			f_{HIRC} off → on	—	16	—	t_{HIRC}
	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 (上电复位或 LVR 硬件复位)	—	RR_{POR}=5V/ms	25	50	100	ms
	系统复位延迟时间 (LVRC/WDTC/RSTC 软件复位)	—	—				
	系统复位延迟时间 (WDT 溢出复位)	—	—	8.3	16.7	33.3	
t_{SRESET}	软件复位最小脉宽	—	—	45	90	120	μs

注：1. 系统启动时间里提到的 f_{SYS} on/off 状态取决于工作模式类型以及所选的系统时钟振荡器。更多相关细节请参考系统工作模式章节。

2. t_{HIRC}、t_{SYS} 等符号所表示的时间单位，是对应频率值的倒数，相关频率值在前面表格有说明。例如，$t_{HIRC}=1/f_{HIRC}$，$t_{SYS}=1/f_{SYS}$ 等等。
3. 若 LIRC 被选择作为系统时钟源且在休眠模式或深度休眠模式下 LIRC 关闭，则上面表格中对应 t_{SST} 数值还需加上 LIRC 频率表格里提供的 LIRC 启动时间 t_{START}。
4. 系统速度切换时间实际上是指新使能的振荡器的启动时间。

输入 / 输出口电气特性

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_{IL}	I/O 口低电平输入电压	3V	—	0	—	0.9	V
		—	—	0	—	$0.3V_{DD}$	
V_{IH}	I/O 口高电平输入电压	3V	—	2.1	—	3.0	V
		—	—	$0.7V_{DD}$	—	V_{DD}	
I_{OL}	I/O 口灌电流	3V	$V_{OL}=0.1V_{DD}$	10	20	—	mA
I_{OH}	I/O 口源电流	3V	$V_{OH}=0.9V_{DD}$	-5	-10	—	mA
R_{PH}	I/O 口上拉电阻（注）	3V	—	20	30	60	kΩ
V_{OL}	I/O 口低电平输出电压	3V	I_{OL}=10mA	—	—	0.3	V
V_{OH}	I/O 口高电平输出电压	3V	I_{OH}=-5mA	2.7	—	—	V
t_{TCK}	TM 时钟输入引脚最小输入脉宽	—	—	0.3	—	—	μs
t_{INT}	外部中断引脚最小输入脉宽	—	—	10	—	—	μs

注：R_{PH} 内部上拉电阻值的计算方法是：将引脚接地并设置为输入且使能上拉电阻功能，然后在特定电源电压下测量该引脚上的电流，最后电压除以测量的电流值从而得到此上拉电阻值。

存储器电气特性

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				
Flash 程序存储器							
V_{DD}	读工作电压	—	—	1.9	—	3.6	V
	擦除 / 写工作电压	—	—	2.2	—	3.6	
t_{DEW}	擦除 / 写时间	—	—	—	2	3	ms
I_{DDPGM}	V_{DD} 电压下烧录 / 擦除电流	—	—	—	—	5.0	mA
E_P	存储单元耐受性	—	—	10K	—	—	E/W
t_{RETD}	ROM 数据保存时间	—	Ta=25°C	—	40	—	Year
RAM 数据存储器							
V_{DD}	读 / 写工作电压	—	—	V_{DDmin}	—	V_{DDmax}	V
V_{DR}	RAM 数据保存电压	—	—	1.0	—	—	V

注：“E/W”表示擦 / 写次数。

LVD/LVR 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_{LVR}	低电压复位电压	—	LVR 使能，电压选择 1.9V	Typ. -0.1	1.9	Typ. +0.1	V
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 2.9V		2.9		
		—	LVD 使能，电压选择 3.0V		3.0		
		—	LVD 使能，电压选择 3.3V		3.3		
I_{OP}	工作电流	3V	LVD 使能，LVR 使能	—	25	35	μA
t_{BGS}	V_{BG} 启动稳定时间	—	无负载	—	—	150	μs
t_{LVDS}	LVDO 稳定时间	—	LVR 使能，LVD off → on	—	—	15	μs
		—	LVR 除能，LVD off → on	—	—	150	μs
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

A/D 转换器电气特性

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}	A/D 转换器工作电压	—	—	2.0	—	3.6	V
V_{ADI}	A/D 转换器输入电压	—	—	0	—	V_{DD}	V
N_R	分辨率	—	—	—	—	12	Bit
DNL	A/D 非线性微分误差	—	t_{ADCK}=0.5μs	-3	—	3	LSB
INL	A/D 非线性积分误差	—	t_{ADCK}=0.5μs	-4	—	4	LSB
I_{ADC}	A/D 转换器使能的额外电流	2.0V	无负载，t_{ADCK}=0.5μs	—	400	500	μA
		3.0V		—	700	850	
		3.6V		—	850	1050	
t_{ADCK}	A/D 转换器时钟周期	—	—	0.5	—	10	μs
t_{ON2ST}	A/D 转换器 On-to-Start 时间	—	—	4	—	—	μs
t_{ADC}	A/D 转换时间 (包括采样和保持时间)	—	—	—	16	—	t_{ADCK}

RF 发射器电气特性

Ta=-40°C~85°C

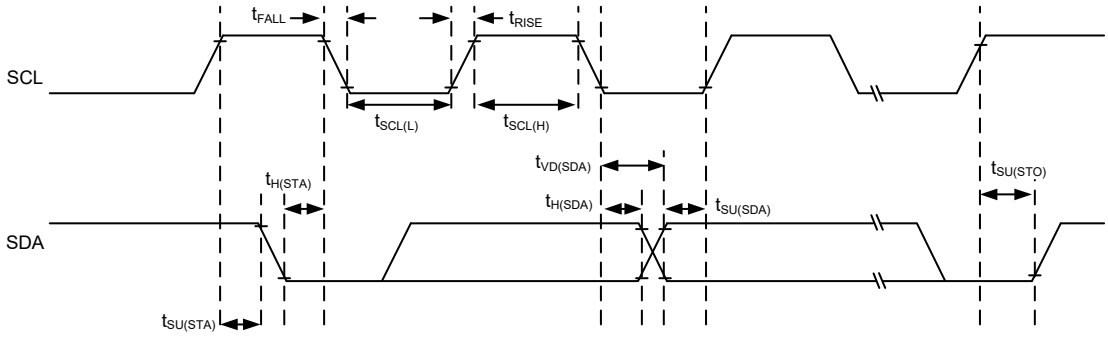
符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DDRF}	RF 工作电压	—	—	2.0	3.0	3.6	V
I_{STBRF}	RF 暂停电流	—	—	—	—	1	μA
f_{XTAL}	RF 工作晶振	—	—	—	16	—	MHz
f_{XTALV}	RF 工作晶振变化	—	—	-20	—	+20	ppm
f_{OP}	RF 工作频率	—	315/433/868/915MHz 频带	300	—	930	MHz
f_{FSK}	FSK 频率偏差	—	—	10	—	150	kHz
f_{STEP}	RF 频率步进	—	f_{OP}=315/433MHz	—	—	50	Hz
		—	f_{OP}=868/915MHz	—	—	100	Hz
R_{FSK}	无线数据传输速率 (FSK)	—	—	0.5	—	100	Kbps
R_{OOK}	无线符号传输速率 (OOK)	—	—	0.5	—	25	Ksps
P_{RF}	输出功率	3.0V	—	0	10	13	dBm
P_{RFV}	输出功率变化	2.0V~3.3V	P_{RF}=10dBm (f_{OP}=433MHz)	—	—	6	dBm
		2.5V~3.3V	P_{RF}=10dBm (f_{OP}=433MHz)	—	—	3	dBm
I_{DDTX}	发射器工作电流	3.0V	P_{RF}=0dBm (f_{OP}=315/433MHz)	—	10	—	mA
		3.0V	P_{RF}=10dBm (f_{OP}=315/433MHz)	—	17	—	mA
		3.0V	P_{RF}=13dBm (f_{OP}=315/433MHz)	—	23	—	mA
		3.0V	P_{RF}=0dBm (f_{OP}=868/915MHz)	—	11	—	mA
		3.0V	P_{RF}=10dBm (f_{OP}=868/915MHz)	—	20	—	mA
		3.0V	P_{RF}=13dBm (f_{OP}=868/915MHz)	—	26	—	mA
PN_{TX}	发射器相位噪声	—	P_{RF}=10dBm，100kHz 来自载波	—	—	-80	dBc/Hz
		—	P_{RF}=10dBm，1MHz 来自载波	—	—	-100	dBc/Hz
SE_{TX}	发射器杂散发射 (P_{RF}=10dBm，f_{OP}=433MHz)	—	f<1GHz	—	—	-36	dBm
		—	47MHz<f<74MHz，87.5MHz<f<118MHz，174MHz<f<230MHz，470MHz<f<790MHz	—	—	-54	dBm
		—	二次谐波	—	—	-30	dBm
		—	三次谐波	—	—	-30	dBm
		—	二次谐波 (其它频率)	—	—	-30	dBm
		—	三次谐波 (其它频率)	—	—	-30	dBm

I²C 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				
f_{I2C}	I²C 标准模式 (100kHz) 时的 f_{SYS} 频率（注）	—	无去抖时间	2	—	—	MHz
		—	2 个系统时钟时间去抖	4	—	—	MHz
		—	4 个系统时钟时间去抖	4	—	—	MHz
	I²C 快速模式 (400kHz) 时的 f_{SYS} 频率（注）	—	无去抖时间	4	—	—	MHz
		—	2 个系统时钟时间去抖	8	—	—	MHz
		—	4 个系统时钟时间去抖	8	—	—	MHz
f_{SCL}	SCL 时钟频率	3V	标准模式	—	—	100	kHz
			快速模式	—	—	400	
$t_{SCL(H)}$	SCL 时钟高电平时间	3V	标准模式	3.5	—	—	μs
			快速模式	0.9	—	—	
$t_{SCL(L)}$	SCL 时钟低电平时间	3V	标准模式	3.5	—	—	μs
			快速模式	0.9	—	—	
t_{FALL}	SCL 和 SDA 下降沿时间	3V	标准模式	—	—	1.3	μs
			快速模式	—	—	0.34	
t_{RISE}	SCL 和 SDA 上升沿时间	3V	标准模式	—	—	1.3	μs
			快速模式	—	—	0.34	
$t_{SU(SDA)}$	SDA 数据建立时间	3V	标准模式	0.25	—	—	μs
			快速模式	0.1	—	—	
$t_{H(SDA)}$	SDA 数据保持时间	3V	—	0.1	—	—	μs
$t_{VD(SDA)}$	SDA 数据有效时间	3V	—	—	—	0.6	μs
$t_{SU(STA)}$	START 条件建立时间	3V	标准模式	3.5	—	—	μs
			快速模式	0.6	—	—	
$t_{H(STA)}$	START 条件保持时间	3V	标准模式	4.0	—	—	μs
			快速模式	0.6	—	—	
$t_{SU(STO)}$	STOP 条件建立时间	3V	标准模式	3.5	—	—	μs
			快速模式	0.6	—	—	

注：使用去抖功能可使传输更稳定，降低受干扰影响通信失败的机率。

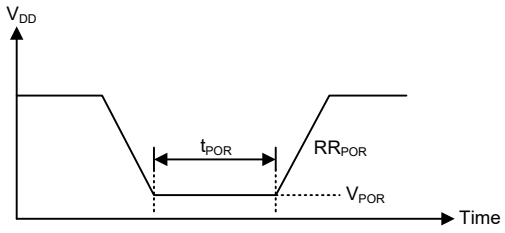


I²C 时序图

上电复位特性

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_{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



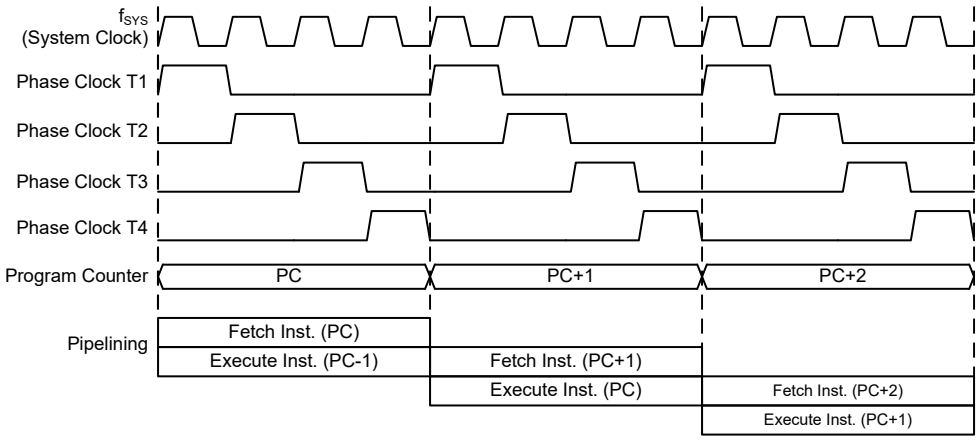
系统结构

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，该系列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需多一个指令周期外，其它大部分标准指令或扩展指令都能分别在一个或两个指令周期内完成。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器中被实现，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度和灵活性的 I/O 和 A/D 控制系统时，仅需要少数的外部器件。使得该系列单片机适用于低成本和大量生产的控制应用。

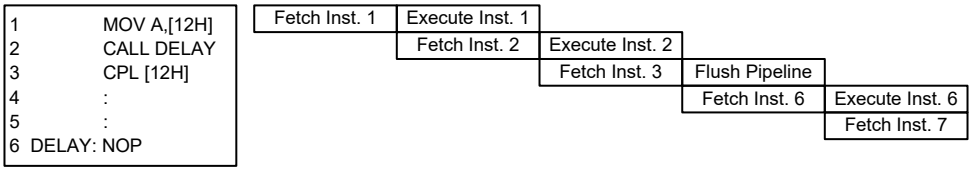
时序和流水线结构

主系统时钟由 HXT、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。

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。



系统时序和流水线



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。

单片机型号	程序计数器	
	高字节	低字节 (PCL)
BC66F2235	PC10~PC8	PCL7~PCL0
BC66F2245	PC11~PC8	PCL7~PCL0
BC66F2255	PC12~PC8	PCL7~PCL0

程序计数器

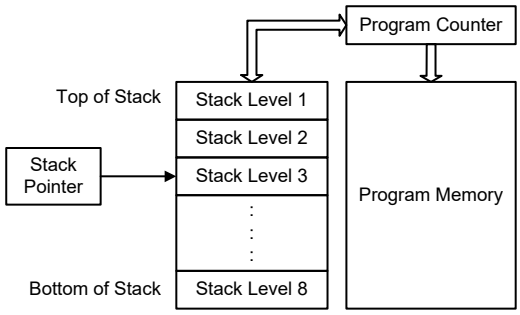
程序计数器的低字节，即程序计数器的低字节寄存器 PCL，可以通过程序控制，且它是可以读取和写入的寄存器。通过直接写入数据到这个寄存器，一个程序短跳转可直接执行，然而只有低字节的操作是有效的，跳转被限制在存储器的当前页中，即 256 个存储器地址范围内，当这样一个程序跳转要执行时，会插入一个空指令周期。程序计数器的低字节可由程序直接进行读取，PCL 的使用可能引起程序跳转，因此需要额外的指令周期。

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。该系列单片机有多层堆栈，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、借位或其它状态的改变，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，
  LADD，LADDM，LADC，LADCM，LSUB，LSUBM，LSBC，LSBCM，
  LDAA
- 逻辑运算：
  AND，OR，XOR，ANDM，ORM，XORM，CPL，CPLA，
  LAND，LANDM，LOR，LORM，LXOR，LXORM，LCPL，LCPLA
- 移位运算：
  RRA，RR，RRCA，RRC，RLA，RL，RLCA，RLC，
  LRR，LRRA，LRRCA，LRRC，LRLA，LRL，LRLCA，LRLC
- 递增和递减：
  INCA，INC，DECA，DEC，
  LINCA，LINC，LDECA，LDEC
- 分支判断：
  JMP，SZ，SZA，SNZ，SIZ，SDZ，SIZA，SDZA，CALL，RET，RETI，
  LSNZ，LSZ，LSZA，LSIZ，LSIZA，LSDZ，LSDZA

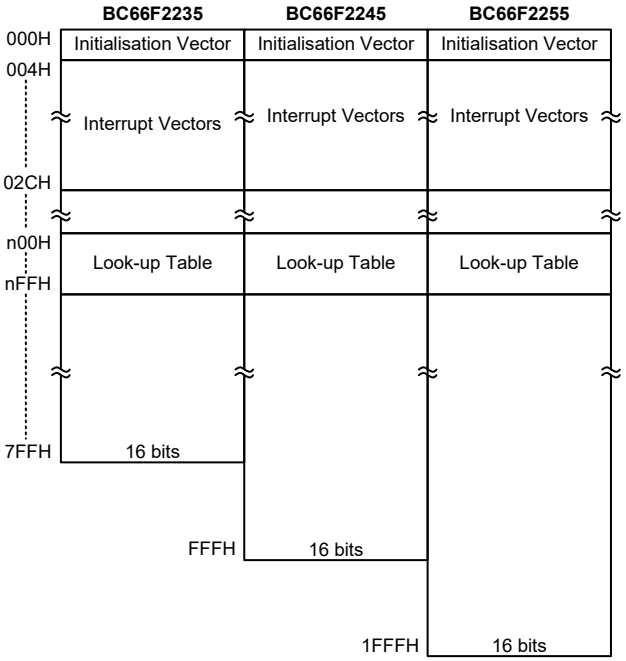
Flash 程序存储器

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，此系列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

该系列单片机的程序存储器容量为 2K×16 位 ~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，由表格指针寄存器来寻址。

单片机型号	容量
BC66F2235	2K×16
BC66F2245	4K×16
BC66F2255	8K×16



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 000H 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。

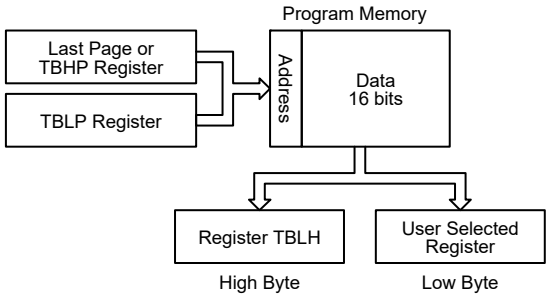
查表

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，当数据存储器 [m] 位于 Sector 0，表格数据可以使用如“TABRD [m]”或“TABRDL [m]”等指令分别从程序存储器查表读取。如果存储

器 [m] 位于其它 Sector，表格数据可以使用如“LTABRD [m]”或“LTABRDL [m]”等指令分别从程序存储器查表读取。当这些指令执行时，程序存储器中表格数据低字节，将被传送到用户指定的数据存储器 [m]，程序存储器中表格数据的高字节，则被传送到 TBLH 特殊功能寄存器。

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“700H”指向的地址是 BC66F2235 2K 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 706H，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]”或“LTABRD”指令被使用，则表格指针指向 TBHP 和 TBLP 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 寄存器为可读 / 可写寄存器，且能重复储存，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

```
ds .section 'data'
tempreg1 db?        ; temporary register #1
tempreg2 db?        ; temporary register #2
code0 .section 'code'
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,07h           ; initialise high table pointer
mov tbhp,a          ; it is not necessary to set tbhp if executing tabrdl
:
:
tabrd tempreg1      ; transfers value in table referenced by table pointer
                    ; data at program memory address "706H" 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 "705H" transferred to
```

```
                    ; tempreg2 and TBLH in this example the data "1AH" is
                    ; transferred to tempreg1 and data "0FH" to tempreg2
                    ; the value "00H" will be transferred to the high byte
                    ; register TBLH
:
:
org 700h            ; sets initial address of last page
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。

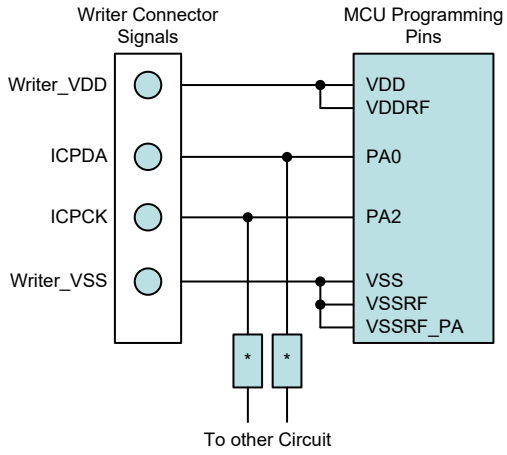
另外，Holtek 单片机提供 4 线接口的在线烧录方式。用户可将进行过烧录或未经过烧录的单片机芯片连同电路板一起制成，最后阶段进行程序的更新和程序的烧录，在无需去除或重新插入芯片的情况下方便地保持程序为最新版。

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 & VDDRF	电源
VSS	VSS & VSSRF & VSSRF_PA	地

注：使用 e-Socket 烧录该系列单片机时，使用者需自行在 V15O 引脚外加 1μF 的电容，以确保烧录的可靠性。

程序存储器可以通过 4 线的接口在线进行烧录。其中一条用于数据串行下载或上传、一条用于串行时钟、另外两条用于提供电源。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

该系列单片机还提供片上调试功能 (OCDS) 用于开发过程中的 MCU 调试。用户可将 OCDSDA 和 OCDSCK 引脚连接至 Holtek HT-IDE 开发工具，从而实现单片机的仿真。OCDSDA 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 OCDS 功能进行调试时，单片机 OCDSDA 和 OCDSCK 引脚上的其它共用功能无效。由于这两个 OCDS 引脚与 ICP 引脚

共用，因此在线烧录时仍用作 Flash 存储器烧录引脚。关于 OCDS 功能的详细描述，请参考“Holtek e-Link for 8-bit MCU OCDS 使用手册”文件。

Holtek e-Link 引脚名称	MCU OCDS 引脚名称	引脚说明
OCDSDA	OCDSDA	片上调试数据 / 地址输入 / 输出
OCDSCK	OCDSCK	片上调试时钟输入
VDD	VDD & VDDRF	电源
VSS	VSS & VSSRF & VSSRF_PA	地

在线应用编程 – IAP

Flash 型程序存储器便于用户在同一芯片上对程序进行更新和修改。单片机提供的 IAP 功能使用户可以方便地对 Flash 程序存储器进行多次编程。IAP 功能可以通过内部固件进行程序的更新，而无需外接烧录器或 PC。此外，IAP 接口通过 I/O 引脚可以设置为任何类型的通信协议，例如 UART。关于内部固件，用户可以选择 Holtek 提供的版本或创建自己的内部固件。以下章节说明了如何实现 IAP 固件程序。

Flash 存储器读 / 写大小

Flash 存储器以块为单位进行擦操作，以 4 字为单位进行写入操作，以字为单位进行读出操作。块的大小为 256 字。注意，在执行写入操作之前必须先执行擦除操作。

Flash 存储器擦 / 写功能成功使能时 CFWEN 位会被硬件置高，当该位被置高，便可写入数据到数据寄存器。FWT 位用于启动写入程序，并指示写入操作的状态。当该位由应用程序置高时将开始一个写入程序，当写入操作结束后该位将由硬件清零。

读出操作是通过一个特定的读出程序来执行的。FRDEN 位用于使能读出功能，由应用程序设置 FRD 位来启动读出程序，并指示读出操作的状态。当读出操作结束后该位将由硬件清零。

单片机型号	程序存储器大小	操作		
		擦除	写入	读出
BC66F2235	2K×16	256 字 / 块	4 字 / 次	1 字 / 次
BC66F2245	4K×16			
BC66F2255	8K×16			

IAP 操作格式

擦除块	写单元	FARH[2:0]	FARL[7:2]	FARL[1:0]
0	0	000	0000 00	xx
	1	000	0000 01	xx
	:	:	:	:
	62	000	1111 10	xx
	63	000	1111 11	xx
1	64	001	0000 00	xx
	65	001	0000 01	xx
	:	:	:	:
	126	001	1111 10	xx
	127	001	1111 11	xx
:	:	:	:	:
7	448	111	0000 00	xx
	449	111	0000 01	xx
	:	:	:	:
	510	111	1111 10	xx
	511	111	1111 11	xx

“x”：不相关

擦除块序号和选择 – **BC66F2235**

擦除块	写单元	FARH[3:0]	FARL[7:2]	FARL[1:0]
0	0	0000	0000 00	xx
	1	0000	0000 01	xx
	:	:	:	:
	62	0000	1111 10	xx
	63	0000	1111 11	xx
1	64	0001	0000 00	xx
	65	0001	0000 01	xx
	:	:	:	:
	126	0001	1111 10	xx
	127	0001	1111 11	xx
:	:	:	:	:

擦除块	写单元	FARH[3:0]	FARL[7:2]	FARL[1:0]
15	960	1111	0000 00	xx
	961	1111	0000 01	xx
	: :	: :	: :	: :
	1022	1111	1111 10	xx
	1023	1111	1111 11	xx

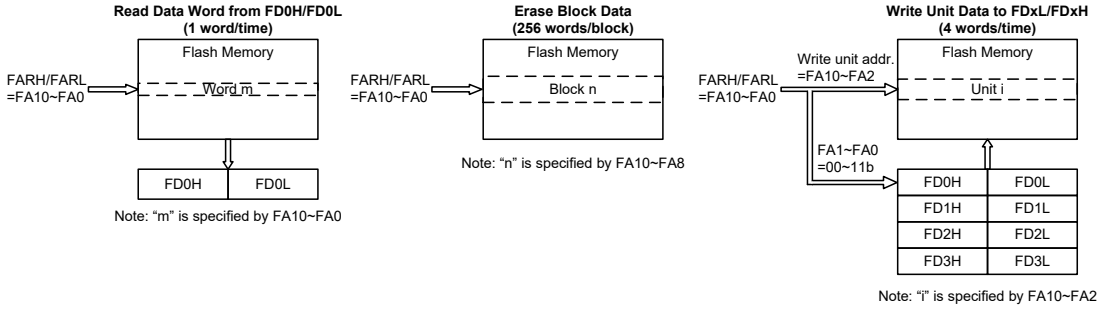
“x”：不相关

擦除块序号和选择 – **BC66F2245**

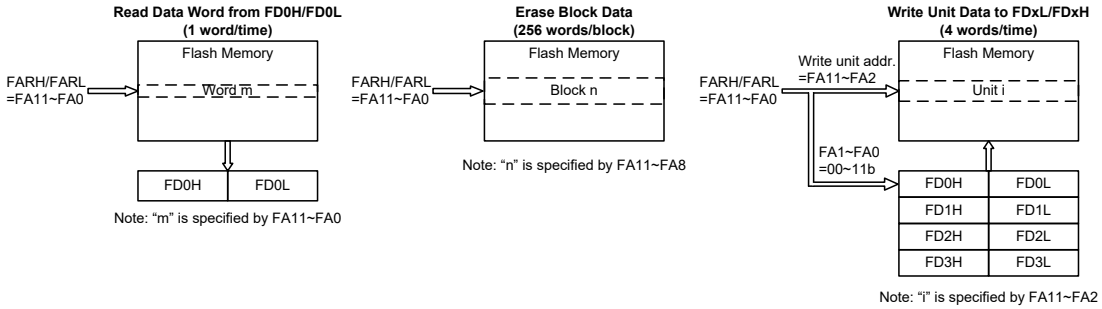
擦除块	写单元	FARH[4:0]	FARL[7:2]	FARL[1:0]
0	0	00000	0000 00	xx
	1	00000	0000 01	xx
	: :	: :	: :	: :
	62	00000	1111 10	xx
	63	00000	1111 11	xx
1	64	00001	0000 00	xx
	65	00001	0000 01	xx
	: :	: :	: :	: :
	126	00001	1111 10	xx
	127	00001	1111 11	xx
: :	: :	: :	: :	: :
31	1984	11111	0000 00	xx
	1985	11111	0000 01	xx
	: :	: :	: :	: :
	2046	11111	1111 10	xx
	2047	11111	1111 11	xx

“x”：不相关

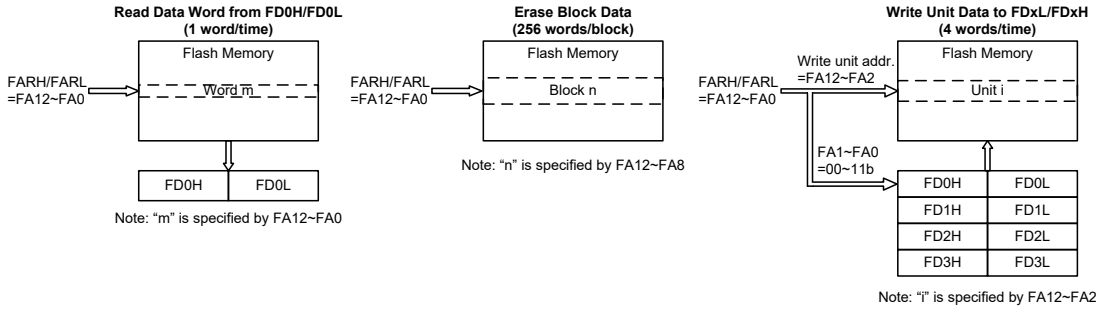
擦除块序号和选择 – **BC66F2255**



Flash 存储器 IAP 读 / 擦 / 写结构 – BC66F2235



Flash 存储器 IAP 读 / 擦 / 写结构 – BC66F2245



Flash 存储器 IAP 读 / 擦 / 写结构 – BC66F2255

IAP Flash 程序存储器寄存器

IAP Flash 程序存储器有两个地址寄存器、四对 16 位数据寄存器和两个控制寄存器。使用地址、数据和控制寄存器可以实现对 Flash 存储器的 16 位数据读写操作。这几个寄存器控制了内部 Flash 程序存储器所有操作，即地址寄存器 FARL 和 FARH，数据寄存器 FDnL 和 FDnH，控制寄存器 FC0 和 FC1。由于地址、数据寄控制寄存器都位于 Sector 1 中，只能通过扩展指令直接访问，或者通过存储器指针对 MP1H/MP1L 或 MP2H/MP2L 和间接寻址寄存器 IAR1 或 IAR2 进行间接读取或写入。

寄存器名称	位							
	7	6	5	4	3	2	1	0
FC0	CFWEN	FMOD2	FMOD1	FMOD0	FWPEN	FWT	FRDEN	FRD
FC1	D7	D6	D5	D4	D3	D2	D1	D0
FARL	FA7	FA6	FA5	FA4	FA3	FA2	FA1	FA0
FARH (BC66F2235)	—	—	—	—	—	FA10	FA9	FA8
FARH (BC66F2245)	—	—	—	—	FA11	FA10	FA9	FA8
FARH (BC66F2255)	—	—	—	FA12	FA11	FA10	FA9	FA8
FD0L	D7	D6	D5	D4	D3	D2	D1	D0
FD0H	D15	D14	D13	D12	D11	D10	D9	D8
FD1L	D7	D6	D5	D4	D3	D2	D1	D0
FD1H	D15	D14	D13	D12	D11	D10	D9	D8
FD2L	D7	D6	D5	D4	D3	D2	D1	D0

寄存器名称	位							
	7	6	5	4	3	2	1	0
FD2H	D15	D14	D13	D12	D11	D10	D9	D8
FD3L	D7	D6	D5	D4	D3	D2	D1	D0
FD3H	D15	D14	D13	D12	D11	D10	D9	D8

IAP 寄存器列表

- **FC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	CFWEN	FMOD2	FMOD1	FMOD0	FWPEN	FWT	FRDEN	FRD
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 **CFWEN**：Flash 存储器擦 / 写功能使能控制位
0：Flash 存储器擦 / 写功能除能
1：Flash 存储器擦 / 写功能已成功使能
当该位由应用程序清零时，Flash 存储器擦 / 写功能除能。注意该位不能由应用程序置高，对该位直接写“1”不会使能擦 / 写功能。此位用于指示 Flash 存储器擦 / 写功能的状态。硬件置高此位，表示 Flash 存储器擦 / 写功能成功使能，否则此位为 0，表示 Flash 存储器擦 / 写功能除能。

Bit 6~4 **FMOD2~FMOD0**：Flash 存储器模式选择位
000：写入模式
001：块擦除模式
010：保留
011：读出模式
100：保留
101：保留
110：Flash 存储器擦 / 写使能模式
111：保留
这几位用于选择 Flash 存储器的操作模式。注意在执行擦 / 写 Flash 存储器操作之前必须先成功使能“Flash 存储器擦 / 写使能模式”。

Bit 3 **FWPEN**：Flash 存储器擦 / 写使能程序触发控制位
0：擦 / 写使能程序未被触发或程序定时器溢出
1：擦 / 写使能程序被触发且程序定时器开始计时
该位用于启动 Flash 存储器擦 / 写使能程序和内部定时器。此位由应用程序置高，当内部定时器计时溢出后由硬件清零。需在 FWPEN 置高后尽快写入正确数据序列到 FD1L/FD1H、FD2L/FD2H 和 FD3L/FD3H 寄存器。

Bit 2 **FWT**：Flash 存储器写入控制位
0：不启动 Flash 存储器写入程序或写入程序已完成
1：启动 Flash 存储器写入程序
此位由软件置高，当 Flash 存储器写入程序结束后由硬件清零。

Bit 1 **FRDEN**：Flash 存储器读出使能位
0：除能
1：使能
此位为 Flash 存储器读出使能位，在执行 Flash 存储器读出操作之前需将此位置高。将此位清零则禁止 Flash 存储器读出操作。

Bit 0 **FRD**：Flash 存储器读出控制位
0：不启动 Flash 存储器读出程序或读出程序已完成
1：启动 Flash 存储器读出程序
此位由软件置高，当 Flash 存储器读出程序结束后由硬件清零。

注：1. 在同一条指令中 FWT、FRDEN 和 FRD 位不可同时设置为“1”。

2. 确保 f_{SUB} 时钟在执行擦或写动作前已稳定。
3. 当读、擦或写动作成功启动后，CPU 相关操作将停止。
4. 确保读、擦或写动作成功完成后才可执行其它操作。

- **FC1 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**：整个芯片复位
当用户写入特定值“55H”到该寄存器，将产生一个复位信号将整个单片机复位。

- **FARL 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	FA7	FA6	FA5	FA4	FA3	FA2	FA1	FA0
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 **FA7~FA0**：Flash 程序存储器地址 bit 7 ~ bit 0

- **FARH 寄存器 – BC66F2235**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	FA10	FA9	FA8
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 未定义，读为“0”
Bit 2~0 **FA10~FA8**：Flash 程序存储器地址 bit 10 ~ bit 8

- **FARH 寄存器 – BC66F2245**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	FA11	FA10	FA9	FA8
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 3~0 **FA11~FA8**：Flash 程序存储器地址 bit 11 ~ bit 8

- **FARH 寄存器 – BC66F2255**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	FA12	FA11	FA10	FA9	FA8
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	0	0	0	0	0

Bit 7~5 未定义，读为“0”
Bit 4~0 **FA12~FA8**：Flash 程序存储器地址 bit 12 ~ bit 8

- **FD0L 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**：第一个 Flash 存储器数据 bit 7 ~ bit 0

- **FD0H 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**：第一个 Flash 存储器数据 bit 15 ~ bit 8

- **FD1L 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**：第二个 Flash 存储器数据 bit 7 ~ bit 0

- **FD1H 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**：第二个 Flash 存储器数据 bit 15 ~ bit 8

- **FD2L 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**：第三个 Flash 存储器数据 bit 7 ~ bit 0

- **FD2H 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**：第三个 Flash 存储器数据 bit 15 ~ bit 8

- **FD3L 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**：第四个 Flash 存储器数据 bit 7 ~ bit 0

- **FD3H 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

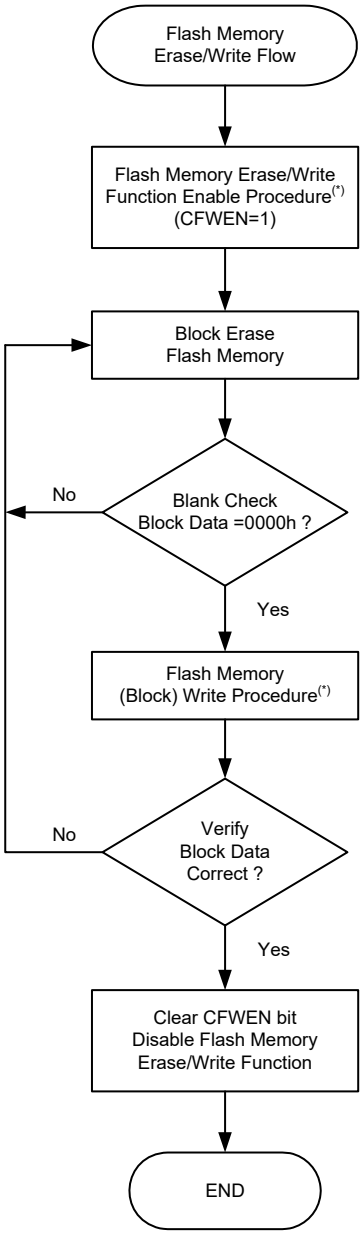
Bit 7~0 **D15~D8**：第四个 Flash 存储器数据 bit 15 ~ bit 8

Flash 存储器擦 / 写流程

在开始更新 Flash 存储器之前，先了解 Flash 存储器擦 / 写流程操作是很重要的，用户可参考下列步骤进行程序开发，以确保 IAP 功能擦 / 写 Flash 存储器内容更新正确。

Flash 存储器擦 / 写流程说明

1. 先启动“Flash 存储器擦 / 写使能程序”。当 Flash 存储器擦 / 写功能成功使能后，FC0 寄存器中的 CFWEN 位会由硬件自动置高，此时才可执行擦 / 写 Flash 存储器操作。详细内容请参考“Flash 存储器擦 / 写使能程序”。
2. 配置 Flash 存储器地址以指定要擦除的块，然后擦除此块。
3. 查空确认是否擦除成功，可采用 TABRD 指令进行读取并比对是否为“0000h”，如果擦除不成功返回步骤 2 再擦除一次。
4. 写入数据至该块，详细内容请参考“Flash 存储器写入程序”。
5. 采用 TABRD 指令进行读取并比对写入数据是否正确，如果读出的数据与写入数据不符，即写入不成功，返回步骤 2 再擦除一次。
6. 完成当前块擦 / 写步骤后，如果无需擦 / 写其它块，可清除 CFWEN 位来解除“Flash 存储器擦 / 写使能模式”。



Flash 存储器擦 / 写流程图

注：“Flash 存储器擦 / 写使能程序”及“Flash 存储器写入程序”详见后续章节介绍。

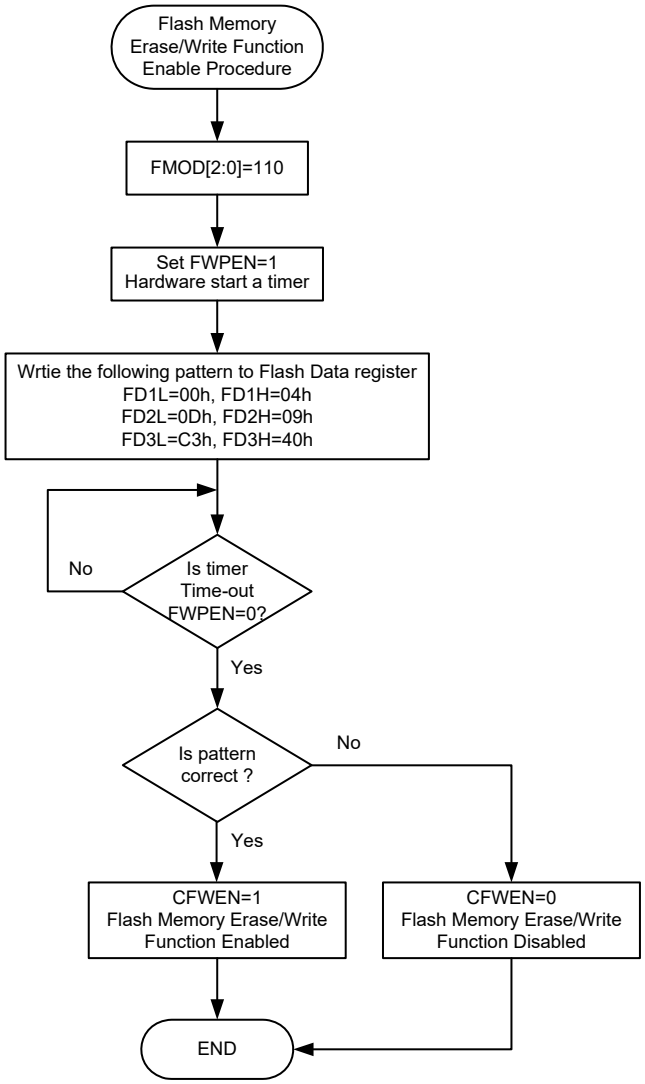
Flash 存储器擦 / 写使能程序

Flash 存储器擦 / 写使能模式是专门为保护 Flash 存储器内容不被轻易地修改而设计的。用户必须先使能 Flash 存储器擦 / 写功能，才能通过 IAP 控制寄存器来更改 Flash 存储器数据。

Flash 存储器擦 / 写使能程序说明

1. 写入数值“110”至 FC0 寄存器中的 FMOD[2:0] 位，选择 Flash 存储器擦 / 写使能模式。
2. 设置 FC0 寄存器中的 FWPEN 位为“1”，启动 Flash 存储器擦 / 写使能程序，此时内部硬件线路会启动一个内部定时器。
3. 使用者必须在 FWPEN 位置高后尽快填入正确数据序列至 FD1L~FD3L 和 FD1H~FD3H 寄存器中，数据序列为 FD1L=00h、FD1H=04h、FD2L=0Dh、FD2H=09h、FD3L=C3h、FD3H=40h。
4. 一旦定时器计时结束，无论写入的数据序列是否正确，FWPEN 位将由硬件自动清零。
5. 如果写入的数据序列不正确，表示 Flash 存储器擦 / 写功能没有成功使能，需重复以上步骤。如果写入的数据序列正确，表示 Flash 存储器擦 / 写功能成功使能。
6. 一旦 Flash 存储器擦 / 写功能成功使能，即可通过 IAP 控制寄存器进行块擦 / 写操作来更新 Flash 存储器内容。

将 FC0 寄存器中的 CFWEN 位清零，可除能 Flash 存储器擦 / 写功能，不必再执行以上步骤。



Flash 存储器擦 / 写使能程序

Flash 存储器写入程序

当 Flash 擦 / 写功能成功使能后，CFWEN 位会被硬件置高，此时要写入 Flash 存储器的数据才能填入到数据寄存器。在开始写入程序之前，应先正确配置 IAP 控制寄存器，将所选的 Flash 存储器块的数据擦除。

Flash 存储器块的大小为 256 个字，地址由 FA10~FA8、FA11~FA8 或 FA12~FA8 指定。

Flash 存储器写入程序说明

对于写入操作每次要写入数据单元的地址需先放入 FARL 和 FARH 寄存器中，要写入的数据需依序存入 FD0L/FD0H ~ FD3L/FD3H 寄存器对中。需注意的是写入操作每次写入 4 字，因此每次写入地址仅由 FARH 和 FARL 寄存器中的 FA10~FA2、FA11~FA2 或 FA12~FA2 位来指定，与 FARL 寄存器中的 FA1~FA0 值无关。

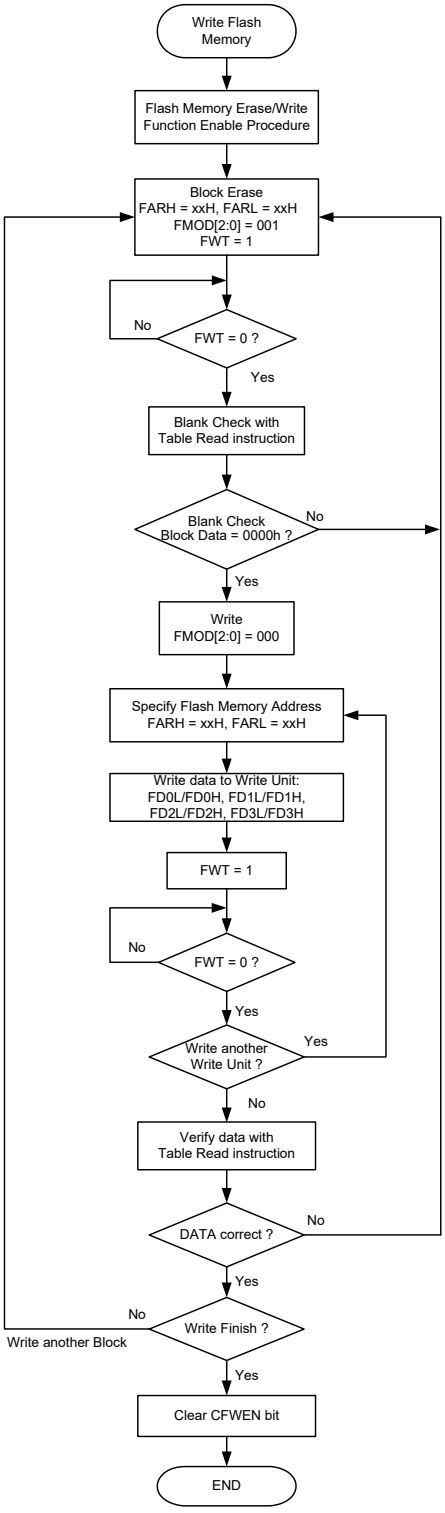
1. 启动“Flash 存储器擦 / 写使能程序”，确认 CFWEN 的值，如果 CFWEN 被硬件置高，表示可进行 IAP 擦 / 写操作。详细内容请参考“Flash 存储器擦 / 写使能程序”。
2. 设定 FMOD[2:0] 为“001”，选择擦除模式。设定 FWT 位为“1”，擦除 FARH 和 FARL 指定的目标块，直到 FWT 变为“0”。
3. 通过查表指令读出方式进行查空，以确保擦除操作已成功完成。

   如果擦除操作不成功则返回步骤 2。

   如果擦除操作成功则接着执行步骤 4。
4. 设定 FMOD[2:0] 为“000”，选择写入模式。
5. 先将目标起始地址写入 FARL 和 FARH 寄存器中，将要写入的数据依序写入 FD0L/FD0H ~ FD3L/FD3H 寄存器。
6. 设定 FWT 位为“1”，将数据寄存器的数据写入以 FARL[1:0]=00b 为起始地址的 4 个连续地址对应的 Flash 存储器中，直到 FWT 变为“0”。
7. 通过查表指令读出方式进行数据比对，以确保写入操作已成功完成。

   如果写入操作不成功则返回步骤 2。

   如果写入操作成功则接着执行步骤 8。
8. 将 CFWEN 位清零以除能 Flash 存储器擦 / 写功能。



Flash 存储器写入程序

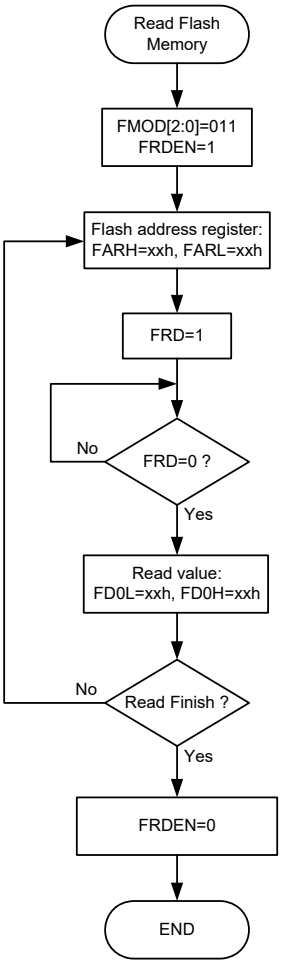
注：1. 当擦或写动作成功启动后，所有 CPU 相关操作将暂停。
2. FWT 位由高变低所需时间为 2.2ms (典型值)。
3. 当 FARH/FARL 的地址组合并非 4 的倍数时，数据无法正常写入依序的 4 笔地址中。

Flash 存储器写入操作注意事项

1. 要开始对 Flash 存储器进行 IAP 擦 / 写操作之前，必须先完成“Flash 存储器擦 / 写使能程序”。
2. Flash 存储器擦除操作以块作为擦除单位。
3. 数据写入 Flash 存储器后，必须以查表指令“TABRD”读出方式比对所写数据是否正确，若比对写入数据不正确时，清除对应的 Flash 存储器块，再重新写入，然后再比对，直到写入正确。
4. IAP 写入与数据比对时需与最高应用频率相同。

Flash 存储器读出程序

要启动 Flash 存储器读出程序，需将 FMOD[2:0] 位设为“011”选择 Flash 存储器读出模式，将 FRDEN 位设为“1”使能读出功能。将要读出的地址填入 FARH 和 FARL 地址寄存器中，并将 FRD 位设为“1”，然后便可开始 Flash 存储器读出操作。当 FRD 被硬件清为“0”时，则可在 FD0H 和 FD0L 寄存器中取得 Flash 存储器中该地址的数据。进行 Flash 存储器读出操作前，无需执行 Flash 存储器擦 / 写使能程序。



Flash 存储器读出程序

注：1. 当读动作成功启动后，所有 CPU 相关操作将暂停。
2. FRD 位由高变低所需时间为 3 个指令周期 (典型值)。

数据存储器

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。

数据存储器分为两种类型，第一种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。第二种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。

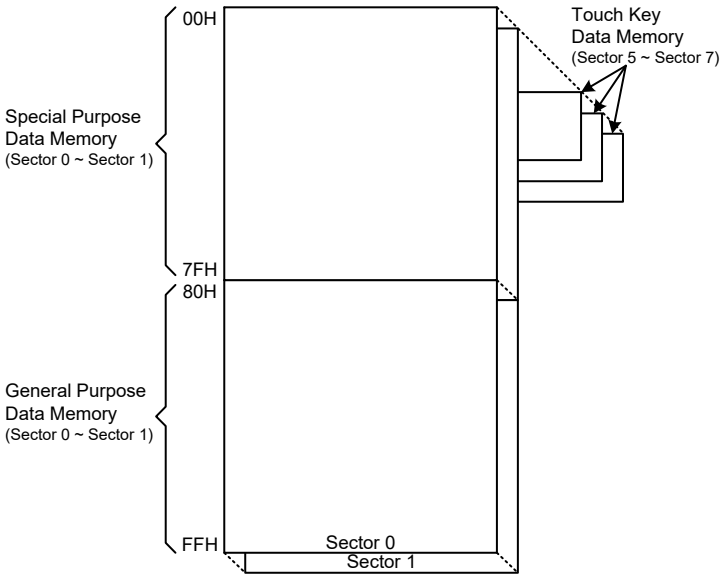
该系列单片机还提供了专门的存储区用于存放执行触控按键时使用到的数据。本章节仅对通用功能数据存储器和特殊功能寄存器存储器进行介绍。关于触控按键数据存储器的使用可参考相应章节内容。

结构

数据存储器被分为若干个 Sector，都位于 8 位存储器中。特殊功能数据寄存器地址范围为 00H~7FH，而通用数据存储器地址范围为 80H~FFH。触控按键数据存储器位于 Sector 5 ~ Sector 7，起始地址为 00H。若用间接寻址方式切换不同的数据存储器 Sector 可通过设置正确的存储器指针值实现。

单片机型号	特殊功能数据存储器	通用数据存储器		触控按键数据存储器	
	所在 Sector	容量	Sector：地址	容量	Sector：地址
BC66F2235	0，1	256×8	0：80H~FFH 1：80H~FFH	96×8	Sector 5：00H~1FH Sector 6：00H~1FH Sector 7：00H~1FH
BC66F2245					
BC66F2255					

数据存储器概要



数据存储器结构

数据存储器寻址

此系列单片机支持扩展指令架构，它并没有可用于数据存储器的存储区指针。对于数据存储器，使用间接寻址访问方式时所需的 Sector 是通过 MP1H 或 MP2H 寄存器指定，而所选 Sector 的某一数据存储器地址是通过 MP1L 或 MP2L 寄存器指定。

直接寻址可用于所有 Sector，通过扩展指令可以寻址所有可用的数据存储器空间。当所访问的数据存储器位于除 Sector 0 外的任何数据存储器 Sector 时，扩展指令可代替间接寻址方式用来访问数据存储器。标准指令和扩展指令的主要区别在于扩展指令中的数据存储器地址“m”有 9 个有效位，高字节表示 Sector，低字节表示指定的地址。

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

数据存储器的这个区域用于存放特殊寄存器，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，相关细节的介绍可参见相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”。

	Sector 0	Sector 1
00H	IAR0	
01H	MP0	
02H	IAR1	
03H	MP1L	
04H	MP1H	
05H	ACC	
06H	PCL	
07H	TBLP	
08H	TBLH	
09H	TBHP	
0AH	STATUS	
0BH		
0CH	IAR2	
0DH	MP2L	
0EH	MP2H	
0FH	RSTFC	
10H	INTC0	
11H	INTC1	
12H	INTC2	
13H		
14H	PA	
15H	PAC	
16H	PAPU	
17H	PAWU	
18H	PB	PC
19H	PBC	PCC
1AH	PBPU	PCPU
1BH	INTEG	
1CH	SCC	
1DH	HIRCC	
1EH	HXTC	
1FH		
20H	LVDC	SIMC0
21H	LVRC	SIMC1/UUCR1
22H	WDTC	SIMC2/SIMA/UUCR2
23H	RSTC	SIMD/UTXR_RXR
24H	PSCR0	SIMTOC/UBRG
25H	PSCR1	UUCR3
26H	PWRC	UUSR
27H	RF_PWR	
28H	MFI0	
29H	MFI1	
2AH	MFI2	
2BH	IFS	
2CH	SADC0	
2DH	SADC1	
2EH	SADOH	
2FH	SADOL	
30H	TB0C	
31H	TB1C	
32H	PTM0AH	
33H	PTM0AL	
34H	PTM0C0	
35H	PTM0C1	
36H	PTM0DH	
37H	PTM0DL	
38H	PTM0RPH	
39H	PTM0RPL	
3AH	CTM0C0	CTM1C0
3BH	CTM0C1	CTM1C1
3CH	CTM0DL	CTM1DL
3DH	CTM0DH	CTM1DH
3EH	CTM0AL	CTM1AL
3FH	CTM0AH	CTM1AH

	Sector 0	Sector 1
40H	PAS0	FC0
41H	PAS1	FC1
42H	PBS0	
43H	PBS1	FARL
44H	PCS0	FARH
45H	PCS1	FD0L
46H		FD0H
47H		FD1L
48H		FD1H
49H		FD2L
4AH		FD2H
4BH		FD3L
4CH		FD3H
4DH		
4EH		
4FH		
50H	RF_OPER	
51H	RF_CLK1	
52H	RF_CLK2	TKTMR
53H	RF_FIFO_CTRL1	TKC0
54H	RF_FIFO_CTRL2	TKC1
55H	RF_FIFO_CTRL3	TKC2
56H	RF_FIFO_CTRL4	TK16DL
57H	RF_MOD1	TK16DH
58H	RF_MOD2	TKM0C0
59H		TKM0C1
5AH	RF_MOD4	TKM0C2
5BH		TKM016DL
5CH		TKM016DH
5DH		TKM0ROL
5EH		TKM0ROH
5FH		TKM0TH16L
60H	RF_OPMOD	TKM0TH16H
61H	RF_SX1	TKM0THS
62H	RF_SX2	TKM1C0
63H	RF_SX3	TKM1C1
64H	RF_SX4	TKM1C2
65H		TKM116DL
66H		TKM116DH
67H	RF_CP3	TKM1ROL
68H	RF_OD1	TKM1ROH
69H		TKM1TH16L
6AH		TKM1TH16H
6BH	RF_VCO1	TKM1THS
6CH		TKM2C0
6DH		TKM2C1
6EH	RF_TX1	TKM2C2
6FH	RF_TX2	TKM216DL
70H	RF_DFC_CAL	TKM216DH
71H	RF_LDO	TKM2ROL
72H	RF_XO1	TKM2ROH
73H		TKM2TH16L
74H		TKM2TH16H
75H		TKM2THS
76H		TKM3C0
77H		TKM3C1
78H		TKM3C2
79H		TKM316DL
7AH		TKM316DH
7BH		TKM3ROL
7CH		TKM3ROH
7DH		TKM3TH16L
7EH		TKM3TH16H
7FH		TKM3THS

: Unused, read as 00H

: Reserved, cannot be changed

特殊功能数据存储器结构

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, IAR2

间接寻址寄存器 IAR0、IAR1 和 IAR2 的地址虽位于数据存储区，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、IAR1 和 IAR2 上的任何动作，将存储器指针 MP0、MP1L/MP1H 或 MP2L/MP2H 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 只可以访问 Sector 0，而 IAR1 和 MP1L/MP1H、IAR2 和 MP2L/MP2H 可以访问所有 Sector。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, MP1H/MP1L, MP2H/MP2L

该系列单片机提供五个存储器指针，即 MP0、MP1L、MP1H、MP2L 和 MP2H。由于这些指针在数据存储器中能像普通的寄存器一般被操作，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存储器指针所指定的地址。MP0、IAR0 仅可用于访问 Sector 0，而 MP1L/MP1H 和 IAR1、MP2L/MP2H 和 IAR2 可根据 MP1H 或 MP2H 寄存器访问所有的 Sector。使用扩展指令可对所有的数据 Sector 进行直接寻址。

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 1

```
data .section ´data´
adres1 db ?
adres2 db ?
adres3 db ?
adres4 db ?
block  db ?
code .section at 0 ´code´
org 00h
start:
    mov a, 04h              ; setup size of block
    mov block, a
    mov a, offset adres1    ; Accumulator loaded with first RAM address
    mov mp0, a              ; setup 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:
    :
```

间接寻址程序举例 2

```
data .section ´data´
adres1 db ?
adres2 db ?
adres3 db ?
```

```
adres4 db ?
block  db ?
code .section at 0 ´code´
org 00h
start:
    mov a, 04h            ; setup size of block
    mov block, a
    mov a, 01h            ; setup the memory sector
    mov mp1h, a
    mov a, offset adres1  ; Accumulator loaded with first RAM address
    mov mp1l, a           ; setup memory pointer with first RAM address
loop:
    clr IAR1              ; clear the data at address defined by MP1L
    inc mp1l              ; increment memory pointer MP1L
    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。

使用扩展指令直接寻址程序举例

```
data .section ´data´
temp db ?
code .section at 0 ´code´
org 00h
start:
    lmov a, [m]           ; move [m] data to acc
    lsub a, [m+1]         ; compare [m] and [m+1] data
    snz  c                ; [m]>[m+1]?
    jmp  continue         ; no
    lmov a, [m]           ; yes, exchange [m] and [m+1] data
    mov  temp, a
    lmov a, [m+1]
    lmov [m], a
    mov  a, temp
    lmov [m+1], a
continue:
    :
```

注：“m”是位于数据存储器任意 Sector 的某一地址。例如，m=1F0H 表示 Sector 1 中的地址 0F0H。

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，程序计数器低字节设置在数据存储器的特殊功能区域内，程序员可对此寄存器进行操作，很容易的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，而当使用这种运算时，要注意会插入一个空指令周期。

查表寄存器 – 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 SC 标志位、CZ 标志位、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、C、SC 和 CZ 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，或低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。
- CZ：不同指令不同标志位的操作结果。详细资料请参考寄存器定义部分。
- SC：当 OV 与当前指令操作结果 MSB 执行“XOR”所得结果。

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	SC	CZ	TO	PDF	OV	Z	AC	C
R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W
POR	x	x	0	0	x	x	x	x

"x"：未知

Bit 7 **SC**：当 OV 与当前指令操作结果 MSB 执行"XOR"所得结果

Bit 6 **CZ**：不同指令不同标志位的操作结果
对于 SUB/SUBM/LSUB/LSUBM 指令，CZ 等于 Z 标志位。
对于 SBC/SBCM/LSBC/LSBCM 指令，CZ 等于上一个 CZ 标志位与当前零标志位执行"AND"所得结果。对于其它指令，CZ 标志位无影响。

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 也受循环移位指令的影响。

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相关的控制寄存器完成的。

振荡器概述

振荡器除了作为系统时钟源，还作为 RF 电路、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部器件。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寄存器选择。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。

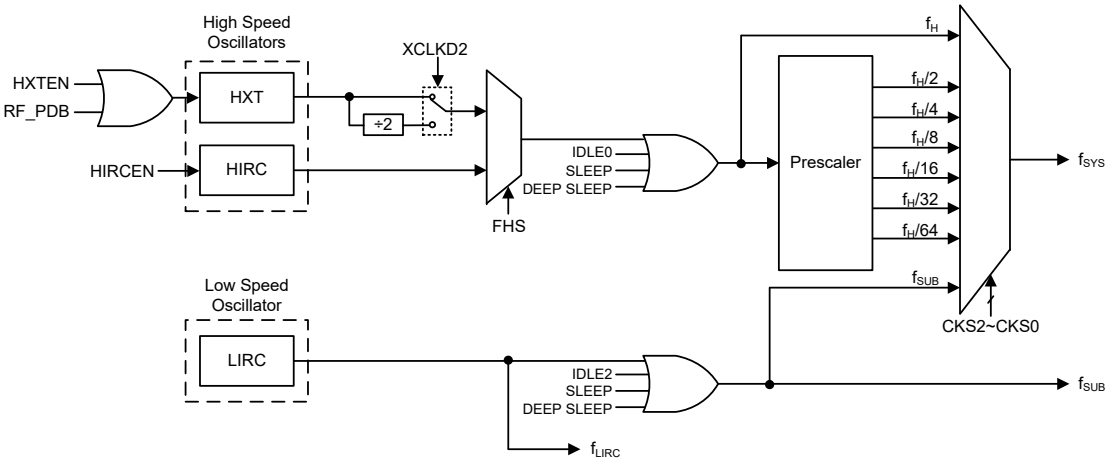
类型	名称	频率	引脚
外部高速晶振	HXT	16MHz	OSC1/OSC2
内部高速 RC	HIRC	8MHz	—
内部低速 RC	LIRC	32kHz	—

振荡器类型

系统时钟配置

该系列单片机有三个系统振荡器，包括两个高速振荡器和一个低速振荡器。高速振荡器有外部晶体 / 陶瓷振荡器 HXT 和内部 8MHz 高速振荡器 HIRC，低速振荡器有内部 32kHz 低速振荡器 LIRC。使用高速或低速振荡器作为系统时钟的选择是通过设置 SCC 寄存器中的 CKS2~CKS0 位决定的，系统时钟可动态选择。

高速振荡器的实际时钟源由 SCC 寄存器的 FHS 位选择。低速或高速系统时钟频率由 SCC 寄存器的 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。

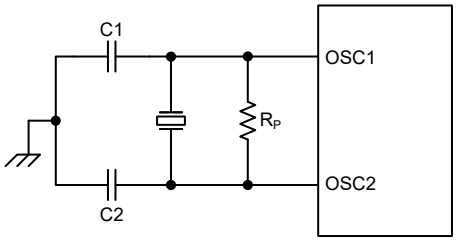


系统时钟配置

外部晶体振荡器 – HXT

外部晶体振荡器是高频振荡器之一，也为 RF 电路的时钟源。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。为保证某些低频率的晶体振荡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

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 振荡器具有固定的频率 8MHz。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，无需额外的外部引脚。

内部 32kHz 振荡器 – LIRC

内部 32kHz 振荡器是低频振荡器。它是一个完全集成 RC 振荡器，它在 3V 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。

工作模式和系统时钟

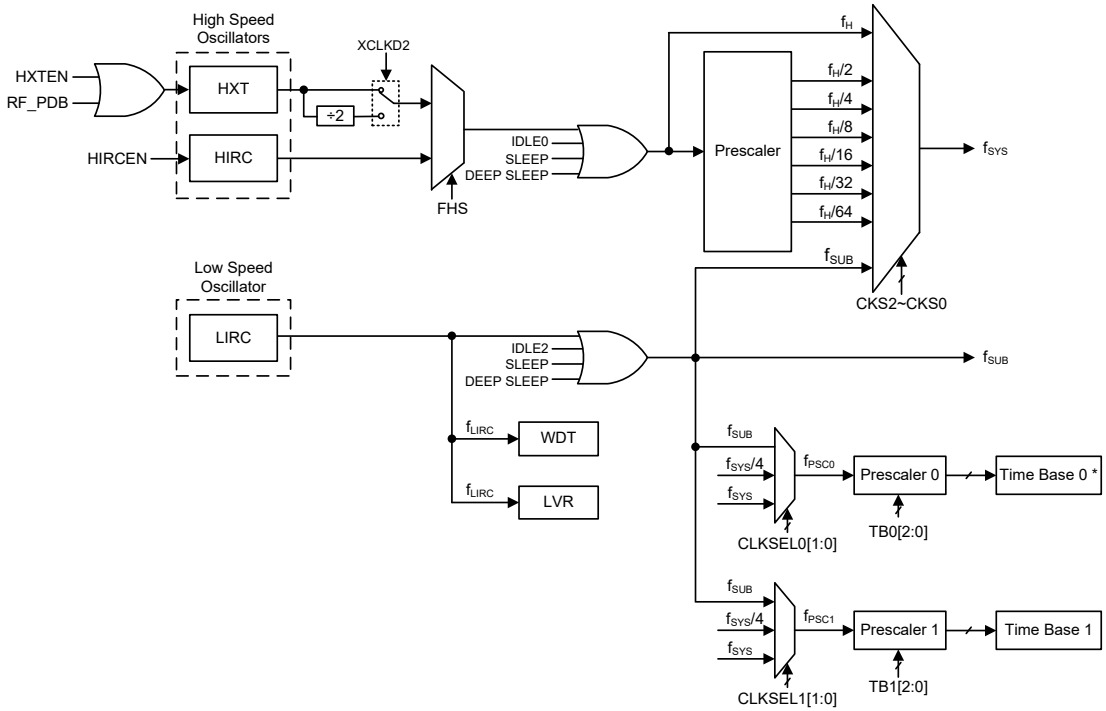
现今的应用要求单片机具有较高的性能及尽可能低的功耗，这种矛盾的要求在便携式电池供电的应用领域尤为明显。高性能所需要的高速时钟将增加功耗，反之亦然。此系列单片机提供高、低速两种时钟源，它们之间可以动态切换，用户可通过优化单片机操作来获得较佳性能 / 功耗比。

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 SCC 寄存器中的 CKS2~CKS0 位进行选择。高频时钟来自 HXT 或 HIRC 振荡器，可通过 SCC 寄存器中的 FHS 位选择。通过 HXTEN 和 RF_PDB 位使能 HXT 振荡器，然后通过 XCLKD2 位进行二分频。RF_PDB 和 XCLKD2 这两位的相关描述详见 RF 章节。

低频系统时钟源来自内部时钟 f_{SUB}，若 f_{SUB} 被选择，低频时钟来自 LIRC 振荡器。其它系统时钟还有高速系统振荡器的分频 $f_H/2$~$f_H/64$。



单片机时钟配置

注：1. 当系统时钟源 f_{SYS} 由 f_H 到 f_{SUB} 转换时，可以通过设置相应的高速振荡器使能控制位选择停止以节省耗电，或者继续振荡，为外围电路提供 f_H~$f_H/64$ 频率的时钟源。
2. 当系统进入深度休眠模式时，时基 0 时钟源为 f_{LIRC}，此时钟源的开启或关闭取决于看门狗定时器的使能 / 除能状态。

系统工作模式

单片机有 7 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：快速模式和低速模式。剩余的 5 种工作模式：深度休眠模式、休眠模式、空闲模式 0、空闲模式 1 和空闲模式 2 用于单片机 CPU 关闭时以节省耗电。

工作模式	LDO	CPU	寄存器设置				f_{SYS}	f_H	f_{SUB}	f_{LIRC}	f_{HXT}
			PWDN	FHIDEN	FSIDEN	CKS2~CKS0					
快速模式	On	On	0	x	x	000~110	f_H~f_H/64	On	On	On	On/Off(3)
低速模式	On	On	0	x	x	111	f_{SUB}	On/Off(1)	On	On	On/Off(3)
空闲模式 0	On	Off	0	0	1	000~110	Off	Off	On	On	On/Off(3)
						111	On				
空间模式 1	On	Off	0	1	1	xxx	On	On	On	On	On/Off(3)
空闲模式 2	On	Off	0	1	0	000~110	On	On	Off	On/Off(2)	On/Off(3)
						111	Off				
休眠模式	On	Off	0	0	0	xxx	Off	Off	Off	On/Off(2)	On/Off(3)
深度休眠模式	Off	Off	1	0	0	xxx	Off	Off	Off	On/Off(2)	Off

“x”：无关

注：1. 在低速模式中，f_H 开启或关闭由相应的振荡器使能位控制。
2. 在空闲模式 2、休眠模式和深度休眠模式中，f_{LIRC} 开启或关闭由 WDT 功能使能或除能控制。然而，在深度休眠模式中 WDT 不工作。
3. 除了深度休眠模式外，f_{HXT} 开启或关闭由 HXTC 寄存器中的 HXTEN 位控制。当 HXTEN 位置高时，f_{HXT} 会开启为 RF 发射器、时基等外围功能提供时钟源。
4. 当单片机进入深度休眠模式时，时基 0 时钟源为 f_{LIRC} (WDTC[7:3] ≠ 10101b)。当单片机从深度休眠模式唤醒后，时基 0 时钟源为 f_{PSC0} (INTC0 将复位)。
5. 当单片机从深度休眠模式唤醒后，大多数寄存器将被复位。因此，需要通过应用程序恢复系统设置。
6. 在 MCU 执行 HALT 指令前，需设定 RF_PDB=1，再设成 RF_PDB=0，以确保静态电流正常。

快速模式

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。在此模式中，1.5V LDO 开启且 PWRC 寄存器中的 PWDN 位为低。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 SCC 寄存器中的 CKS2~CKS0 位选择。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。在此模式中，1.5V LDO 开启且 PWRC 寄存器中的 PWDN 位为低。低速时钟源来自 f_{SUB}，而 f_{SUB} 来自 LIRC 振荡器。

休眠模式

执行 HALT 指令后且 SCC 寄存器中的 FHIDEN 和 FSIDEN 位都为低以及 PWRC 寄存器中的 PWDN 位为低时，系统进入休眠模式。在休眠模式中，CPU 停止运行，f_{SUB} 停止为外围功能提供时钟。若看门狗定时器功能使能，f_{LIRC} 继续运行。

空闲模式 0

执行 HALT 指令后且 SCC 寄存器中的 FHIDEN 位为低、FSIDEN 位为高以及 PWRC 寄存器中的 PWDN 位为低时，系统进入空闲模式 0。在空闲模式 0 中，CPU 停止，但低速振荡器会开启以驱动一些外围功能。

空闲模式 1

执行 HALT 指令后且 SCC 寄存器中的 FHIDEN 和 FSIDEN 位都为高以及 PWRC 寄存器中的 PWDN 位为低时，系统进入空闲模式 1。在空闲模式 1 中，

CPU 停止，但高速和低速振荡器都会开启以保持一些外围功能继续工作。

空闲模式 2

执行 HALT 指令后且 SCC 寄存器中的 FHIDEN 位为高、FSIDEN 位为低以及 PWRC 寄存器中的 PWDN 位为低时，系统进入空闲模式 2。在空闲模式 2 中，CPU 停止，但高速振荡器会开启以保持一些外围功能继续工作。

深度休眠模式

执行 HALT 指令后且 PWRC 寄存器中的 PWDN 位为高时，系统进入深度休眠模式。在深度休眠模式中，CPU 停止运行，f_{SUB} 停止为外围功能提供时钟。若看门狗定时器功能使能，f_{LIRC} 继续运行。然而，无论 WDT 功能是否使能，在深度休眠模式中 WDT 不工作。

控制寄存器

寄存器 SCC、HIRCC、HXTC 和 PWRC 用于控制系统时钟和相应的振荡器配置。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SCC	CKS2	CKS1	CKS0	—	FHS	D2	FHIDEN	FSIDEN
HIRCC	—	—	—	—	—	—	HIRCF	HIRCEN
HXTC	—	—	—	—	—	—	HXTF	HXTEN
PWRC	PA_WAKE	TK_WAKE	—	TB0_WAKE	POF33V	LCMD	IO_ISO_EN	PWDN

系统工作模式控制寄存器列表

- **SCC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	CKS2	CKS1	CKS0	—	FHS	D2	FHIDEN	FSIDEN
R/W	R/W	R/W	R/W	—	R/W	R/W	R/W	R/W
POR	0	0	1	—	0	0	0	0

Bit 7~5 **CKS2~CKS0**：系统时钟选择位
000：f_H
001：$f_H/2$
010：$f_H/4$
011：$f_H/8$
100：$f_H/16$
101：$f_H/32$
110：$f_H/64$
111：f_{SUB}
这三位用于选择系统时钟源。除了 f_H 或 f_{SUB} 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 未定义，读为“0”

Bit 3 **FHS**：高频时钟选择位
0：HIRC – 内部高频振荡器
1：HXT – 外部晶振
当该位置高时，即选择外部 HXT 振荡器作为系统时钟源，单片机实际时钟输入由连接到 OSC1 和 OSC2 引脚的晶体以及 RF_CLK1 寄存器中的 XCLKD2 位共同决定的。例如，如果 f_{HXT}=16MHz，XCLKD2=0，则 f_H=16MHz。

Bit 2 **D2**：保留位

Bit 1 **FHIDEN**：CPU 关闭时高频振荡器控制位
0：除能
1：使能
此位用来控制在 CPU 执行 HALT 指令关闭后高速振荡器是被激活还是停止。

Bit 0 **FSIDEN**：CPU 关闭时低频振荡器控制位
0：除能
1：使能
此位用来控制在 CPU 执行 HALT 指令关闭后低速振荡器是被激活还是停止。

注：使用 CKS2~CKS0 位或 FHS 位进行时钟切换设置之后，在相关时钟成功切换至目标时钟源之前需要一定的延时。因此，若接下来执行的操作需要目标时钟源立即响应，则在此之前必须规划适当的延迟时间。

时钟切换延迟时间 = $4\times t_{SYS}$ + [0 ~ $(1.5\times t_{Curr.}+0.5\times t_{Tar.})$]，其中 $t_{Curr.}$ 指代当前的时钟周期，$t_{Tar.}$ 指代目标时钟周期，t_{SYS} 指代当前系统时钟周期。

- **HIRCC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	HIRCF	HIRCEN
R/W	—	—	—	—	—	—	R	R/W
POR	—	—	—	—	—	—	0	1

Bit 7~2 未定义，读为“0”

Bit 1 **HIRCF**：HIRC 振荡器稳定标志位
0：HIRC 不稳定
1：HIRC 稳定
此位用于表明 HIRC 振荡器是否稳定。HIRCEN 位置高使能 HIRC 振荡器，HIRCF 位会先被清零，在 HIRC 稳定后会被置高。

Bit 0 **HIRCEN**：HIRC 振荡器使能控制位
0：除能
1：使能

- **HXTC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	HXTF	HXTEN
R/W	—	—	—	—	—	—	R	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义，读为“0”

Bit 1 **HXTF**：HXT 振荡器稳定标志位
0：HXT 不稳定
1：HXT 稳定
此位用于表明 HXT 振荡器是否稳定。HXTEN 位置高使能 HXT 振荡器后，HXTF 位会先被清零，在 HXT 稳定后会被置高。

Bit 0 **HXTEN**：HXT 振荡器使能控制位
0：除能
1：使能

- **PWRC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PA_WAKE	TK_WAKE	—	TB0_WAKE	POF33V	LCMD	IO_ISO_EN	PWDN
R/W	R/W	R/W	—	R/W	R/W	R/W	R/W	R/W
POR	0	0	—	0	1	0	0	0

Bit 7 **PA_WAKE**：端口 A 使单片机从深度休眠模式中唤醒标志位
0：未发生端口 A 使单片机从深度休眠模式中唤醒
1：发生端口 A 使单片机从深度休眠模式中唤醒
该位表明单片机是否通过端口 A 从深度休眠模式中唤醒。在单片机进入深度休眠模式前，应正确配置 PAWU 寄存器，并且将 PA_WAKE 位清零。

Bit 6 **TK_WAKE**：触控按键使单片机从深度休眠模式中唤醒标志位
0：未发生触控按键使单片机从深度休眠模式中唤醒
1：发生触控按键使单片机从深度休眠模式中唤醒
该位表明单片机是否通过触控按键从深度休眠模式中唤醒。在单片机进入深度休眠模式前，应正确配置触控按键相关寄存器，并且将 TK_WAKE 位清零。

Bit 5 未定义，读为“0”

Bit 4 **TB0_WAKE**：时基 0 使单片机从深度休眠模式中唤醒标志位
0：未发生时基 0 使单片机从深度休眠模式中唤醒
1：发生时基 0 使单片机从深度休眠模式中唤醒
该位表明单片机是否通过时基 0 中断从深度休眠模式中唤醒。在单片机进入深度休眠模式前，应正确配置 TB0C 寄存器，并且将 TB0_WAKE 位清零。

Bit 3 **POF33V**：3.3V 电源域发生上电复位标志位
0：3.3V 电源域未发生上电复位
1：3.3V 电源域发生上电复位
当 3.3V 电源域发生上电复位时，该位由硬件置高。可通过应用程序或执行“CLR WDT”指令将此位清零。

Bit 2 **LCMD**：1.5V LDO 小电流模式选择
0：1.5V LDO 正常模式
1：1.5V LDO 小电流模式

Bit 1 **IO_ISO_EN**：I/O 隔离模式选择
0：I/O 处于正常工作模式
1：I/O 处于隔离模式 (I/O 状态保持不变)
在系统进入深度休眠模式前，该位必须置高来锁存输入 / 输出端口，以使输入 / 输出端口状态在深度休眠模式下仍保持不变。当单片机从深度休眠模式中唤醒后，必须通过应用程序将该位清零来将输入 / 输出端口解除锁存。

Bit 0 **PWDN**：深度休眠模式控制
0：单片机正常工作
1：执行 HALT 指令后单片机进入深度休眠模式 (1.5V LDO 由硬件关闭)
如果该位置高，且执行 HALT 指令后，单片机进入深度休眠模式，这时 1.5V LDO 将由硬件关闭。在单片机进入深度休眠模式前，用户应预先通过应用程序将 1.5V 域系统设置储存到数据存储器，并锁存住 I/O。

有几点注意事项需在此强调：

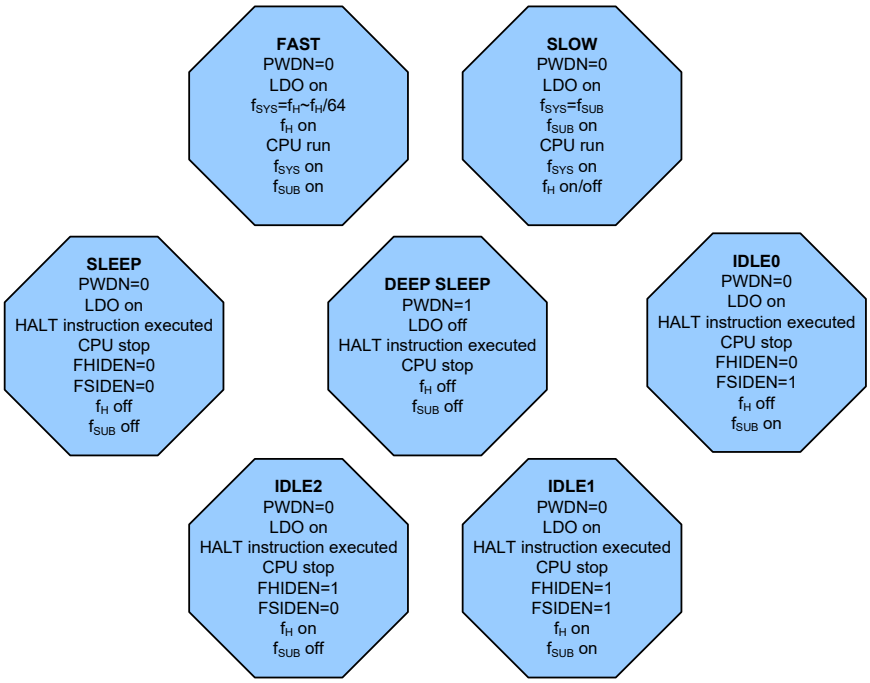
1. 在单片机进入深度休眠模式之前，需通过应用程序将系统设置预先备份。
2. 当 PWDN=1 且执行 HALT 指令后，单片机将进入深度休眠模式。
3. 当单片机进入深度休眠模式后，数据存储器将保持最新值。
4. 当单片机进入深度休眠模式后，RF_PWR 寄存器的 RF_PDB 位将由硬件自动清零。

5. 可通过以下情况将单片机从深度休眠模式中唤醒，唤醒后程序将从 0000h 开始执行。
   - PA 口下降沿
   - 触控按键
   - 时基 0 中断
6. 要想通过 PA 口下降沿将单片机从深度休眠模式中唤醒，则在单片机进入深度休眠模式之前应合理设置 PAWU 寄存器，并且将 PA_WAKE 位清零。
7. 要想通过 TB0 中断将单片机从深度休眠模式中唤醒，则在单片机进入深度休眠模式之前应合理设置 TB0C 寄存器，并且将 TB0_WAKE 位清零。注意，TB0 中断唤醒功能与该中断功能是否使能无关。
8. 要想通过触控按键将单片机从深度休眠模式中唤醒，则在单片机进入深度休眠模式之前应应合理设置触控按键相关寄存器，并且将 TK_WAKE 位清零。
9. 当单片机进入深度休眠模式时，无论分频器 0 时钟源选择如何设置，TB0 时钟源将固定来自 f_{LIRC}。
10. 单片机从深度休眠模式唤醒后，所有寄存器都将被复位。因此需通过应用程序恢复系统设置。

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。

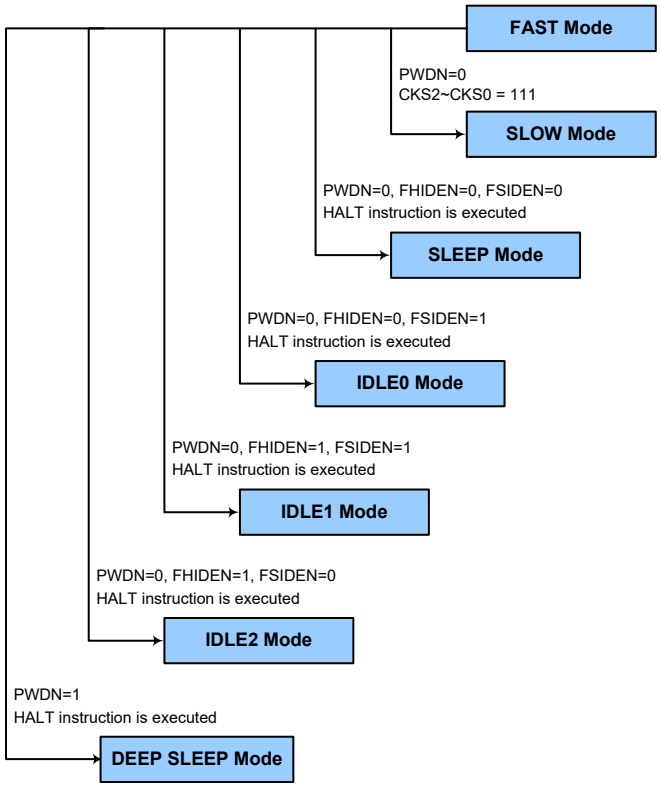
简单来说，快速模式和低速模式间的切换仅需设置 SCC 寄存器中的 CKS2~CKS0 位即可实现，而快速模式 / 低速模式与空闲模式 / 休眠模式 / 深度休眠模式间的切换经由 HALT 指令实现。当 HALT 指令执行后，单片机是否进入空闲模式、休眠模式或深度休眠模式由 SCC 寄存器中的 FHIDEN 和 FSIDEN 位以及 PWRC 寄存器中的 PWDN 位决定的。



快速模式切换到低速模式

系统运行在快速模式时使用高速系统振荡器，因此较为耗电。可通过设置 SCC 寄存器中的 CKS2~CKS0 位为“111”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。

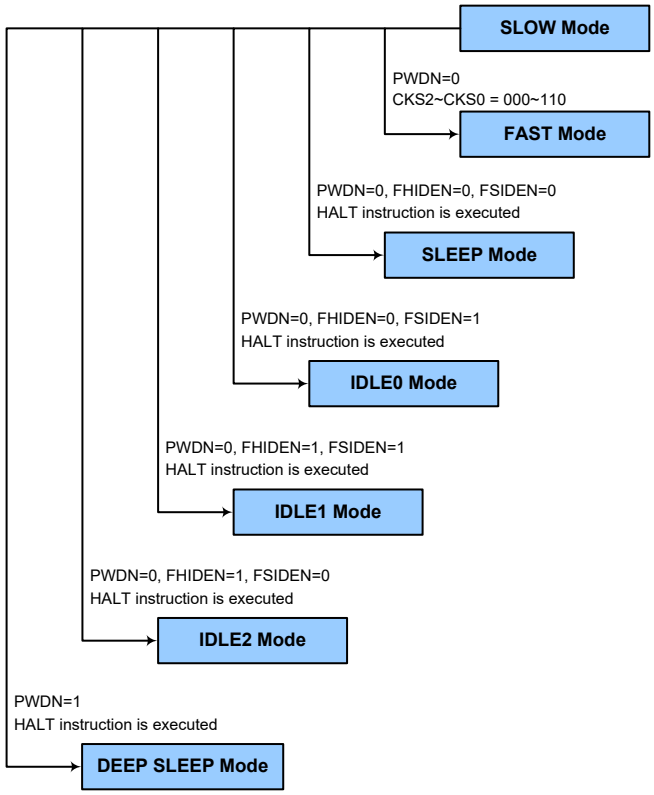
低速模式的时钟源来自 LIRC 振荡器，因此要求这个振荡器在所有模式切换动作发生前稳定下来。



低速模式切换到快速模式

在低速模式时系统时钟来自 f_{SUB}。切换回快速模式时，需设置 CKS2~CKS0 位为“000”~“110”使系统时钟从 f_{SUB} 切换到 f_H~$f_H/64$。

然而，如果在低速模式下 f_H 因未使用而关闭，那么从低速模式切换到快速模式时，它需要一定的时间来重新起振和稳定，可通过检测 HXTC 寄存器中的 HXTF 位或 HIRCC 寄存器中的 HIRCF 位进行判断，所需的高速系统振荡器稳定时间在系统上电时间电气特性中有说明。



进入休眠模式

进入休眠模式的方法仅有一种，即应用程序中执行“HALT”指令前需设置 SCC 寄存器中的 FHIDEN 和 FSIDEN 位都为“0”，且 PWRC 寄存器中的 PWDN 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。
- 数据存储器中的内容和寄存器将保持当前值。
- 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。
- 如果 WDT 功能使能，WDT 将被清零并重新开始计数。如果 WDT 功能除能，WDT 将被清零并停止计数。

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 SCC 寄存器中的 FHIDEN 位为“0”且 FSIDEN 位为“1”，且 PWRC 寄存器中的 PWDN 位为“0”。在上述条件下执行该指令后，将发生的情况如下：

- f_H 时钟停止运行，应用程序停止在“HALT”指令处，但 f_{SUB} 时钟将继续运行。
- 数据存储器中的内容和寄存器将保持当前值。
- 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。
- 如果 WDT 功能使能，WDT 将被清零并重新开始计数。如果 WDT 功能除能，WDT 将被清零并停止计数。

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 SCC 寄存器中的 FHIDEN 和 FSIDEN 位都为“1”，且 PWRC 寄存器中的 PWDN 位为“0”。在上述条件下执行该指令后，将发生的情况如下：

- f_H 和 f_{SUB} 时钟开启，应用程序停止在“HALT”指令处。
- 数据存储器中的内容和寄存器将保持当前值。
- 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。
- 如果 WDT 功能使能，WDT 将被清零并重新开始计数。如果 WDT 功能除能，WDT 将被清零并停止计数。

进入空闲模式 2

进入空闲模式 2 的方法仅有一种，即应用程序中执行“HALT”指令前需设置 SCC 寄存器中的 FHIDEN 位为“1”且 FSIDEN 位为“0”，且 PWRC 寄存器中的 PWDN 位为“0”。在上述条件下执行该指令后，将发生的情况如下：

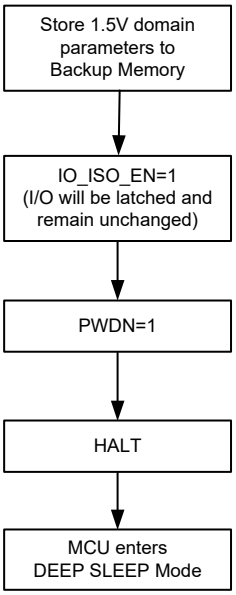
- f_H 时钟开启，f_{SUB} 时钟关闭，应用程序停止在“HALT”指令处。
- 数据存储器中的内容和寄存器将保持当前值。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。
- 如果 WDT 功能使能，WDT 将被清零并重新开始计数。如果 WDT 功能除能，WDT 将被清零并停止计数。

进入深度休眠模式

进入深度休眠模式的方法仅有一种，即应用程序中执行“HALT”指令前需设置 PWRC 寄存器中的 PWDN 位为“1”。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟停止运行，应用程序停止在“HALT”指令处。
- 数据存储器中的内容和寄存器将保持当前值。建议预先储存重要的 1.5V 域系统设置到数据存储器中。
- 如果 PWRC 寄存器中的 IO_ISO_EN 位置高，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。
- 无论 WDT 功能是否使能，WDT 将被清零并停止计数。

以下流程图显示了单片机进入深度休眠模式后的程序步骤。



待机电流的注意事项

由于单片机进入深度休眠、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 和空闲模式 2 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。这也应用于有不同封装的单片机，因为它们可能含有未引出的引脚，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 振荡器，会导致耗电增加。

在空闲模式 1 和空闲模式 2 中，高速振荡器开启。若外围功能时钟源来自高速振荡器，额外的待机电流也可能会有几百微安。

唤醒

单片机进入深度休眠模式、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

系统进入深度休眠模式之后，可以通过以下几种方式唤醒：

- PA 口下降沿
- 时基 0 中断
- 触控按键

单片机执行 HALT 指令，PDF 将被置位；系统上电或执行清除看门狗的指令，

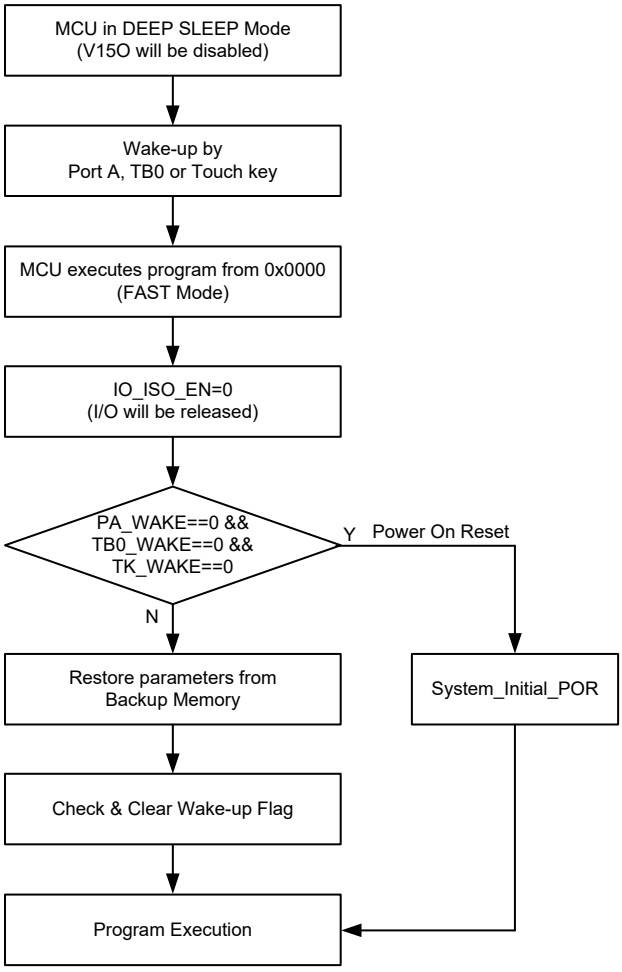
PDF 将被清零。看门狗计数器溢出将会置位 TO 标志并唤醒系统，这种复位会重置程序计数器和堆栈指针，其它标志保持原有状态。

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 端口使单片机系统从休眠或空闲模式唤醒后，程序将在“HALT”指令后继续执行。然而，当PA端口使单片机系统从深度休眠模式唤醒后，程序将从0000h继续执行。

如果系统是通过中断从休眠或空闲模式唤醒，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满，则中断可以马上执行。如果系统是通过时基 0 中断从深度休眠模式唤醒，程序将从 0000h 继续执行。

如果在进入深度休眠、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。

以下流程图显示了单片机从深度休眠模式唤醒后的程序步骤。



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_{LIRC}，而 f_{LIRC} 的时钟源由内部低速振荡器 LIRC 提供。内部振荡器 LIRC 的频率大约为 32kHz，这个特殊的内部时钟周期会随 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 2^8~2^{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 寄存器用于选择溢出周期、控制 WDT 功能的使能 / 除能以及软件复位单片机。

- **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 软件控制

如果配置选项中 WDT 设置为“始终使能”
10101：使能
01010：使能
其它值：单片机复位

如果配置选项中 WDT 设置为“由 WDT 控制寄存器设置”
10101：除能
01010：使能
其它值：单片机复位

通过 WDT 配置选项的选择决定 WE4~WE0 位如何设置。

如果由于不利的环境因素使这些位变为除 10101B 和 01010B 的其它值，单片机将复位。复位动作发生在一段延迟时间 t_{SRESET} 后，且 RSTFC 寄存器的 WRF 位将置为“1”。

注意，无论 WDT 功能是否使能，在深度休眠模式中 WDT 不工作。

Bit 2~0 **WS2~WS0**：WDT 溢出周期选择位

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

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。

- **RSTFC 寄存器**

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

“x”：未知

Bit 7~4　未定义，读为“0”

Bit 3　**RSTF**：复位控制寄存器软件复位标志位
具体描述见内部复位控制章节。

Bit 2　**LVRF**：LVR 复位标志位
具体描述见低电压复位章节。

Bit 1　**LRF**：LVR 控制寄存器软件复位标志位
具体描述见低电压复位章节。

Bit 0　**WRF**：WDT 控制寄存器软件复位标志位
0：未发生
1：发生
当 WDT 控制寄存器软件复位发生时，此位被置为“1”，且只能通过应用程序清零。

看门狗定时器操作

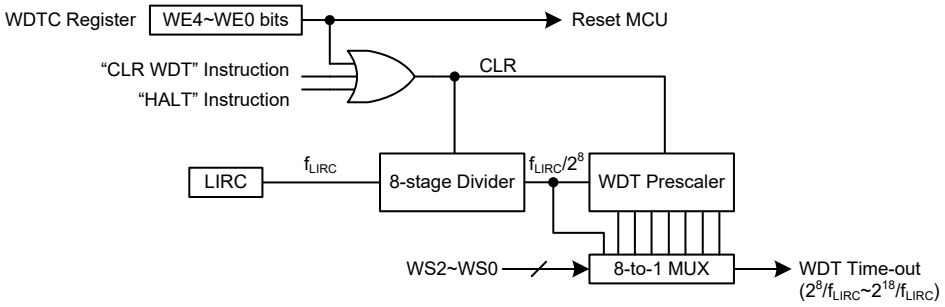
当 WDT 溢出时，它产生一个芯片复位的动作。这也就意味着正常工作期间，用户需在应用程序中看门狗溢出前有策略地清看门狗定时器以防止其产生复位，可使用清除看门狗指令实现。无论什么原因，程序失常跳转到一个未知的地址或进入一个死循环，这个清除指令都不能被正确执行，此种情况下，看门狗将溢出以使单片机复位。看门狗定时器控制寄存器 WDTC 中的 WE4~WE0 位可提供看门狗定时器使能 / 除能控制以及单片机复位操作。WDT 功能可由相关的配置选项以及下表中 WE4~WE0 位共同决定。如果 WE4~WE0 设置为除“01010B”和“10101B”以外的值时，单片机将在一段延迟时间 t_{SRESET} 后复位。上电后这些位初始化为“01010B”。

WDT 配置选项	WE4~WE0 位	WDT 功能
WDT 始终使能	01010B 或 10101B	使能
	其它值	单片机复位
WDT 控制寄存器控制	10101B	除能
	01010B	使能
	其它值	单片机复位

看门狗定时器功能控制

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。若系统处于深度休眠、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 WDT 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。该系列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。

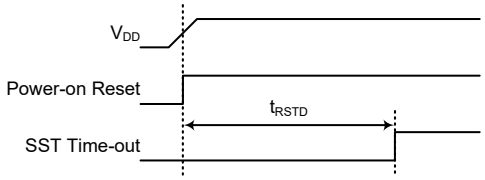
除了上电复位外，另一种复位为低电压复位即 LVR 复位，在电源供应电压低于 LVR 设定值时，系统会产生 LVR 复位。还有一种复位为看门狗溢出单片机复位，不同方式的复位操作会对寄存器产生不同的影响。

复位功能

单片机包含下面几种由内部事件触发的复位方式：

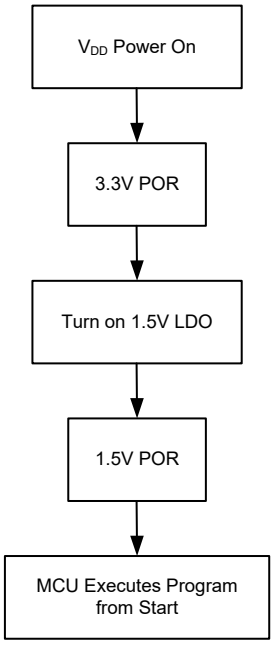
上电复位

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。



上电复位时序图

以下流程图显示了单片机上电后的程序步骤。



内部复位控制

内部复位控制寄存器 RSTC 用于为单片机在受到环境噪声干扰而异常工作时提供复位。如果 RSTC 寄存器的值被设置为除 01010101B 或 10101010B 以外的任何值，单片机会在一段延迟时间 t_{SRESET} 后发生复位。上电后寄存器的值为 01010101B。

RSTC7~RSTC0 位	复位功能
01010101B	无操作
10101010B	无操作
其它值	MCU 复位

内部复位功能控制

- **RSTC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	RSTC7	RSTC6	RSTC5	RSTC4	RSTC3	RSTC2	RSTC1	RSTC0
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 **RSTC7~RSTC0**：复位功能控制位
01010101：无操作
10101010：无操作
其它值：MCU 复位
如果由于不利的环境因素使这些位发生改变，单片机将复位。复位动作发生在一段延迟时间 t_{SRESET} 后，且 RSTFC 寄存器的 RSTF 位将置为“1”。

- **RSTFC 寄存器**

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

“x”：未知

Bit 7~4　未定义，读为“0”

Bit 3　**RSTF**：复位控制寄存器软件复位标志位
0：未发生
1：发生
当 RSTC 控制寄存器软件复位发生时，此位被置为“1”，且只能通过应用程序清零。

Bit 2　**LVRF**：LVR 复位标志位
具体描述见低电压复位章节。

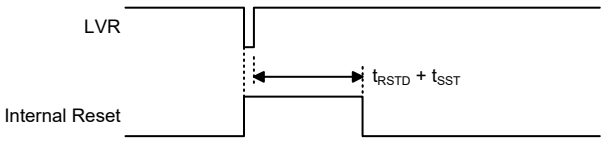
Bit 1　**LRF**：LVR 控制寄存器软件复位标志位
具体描述见低电压复位章节。

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

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。当电压低于一个预先定义的特定电压时，单片机会复位。

LVR 功能的使能 / 除能控制取决于相关的配置选项，且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在 0.9V~V_{LVR} 之间，这时 LVR 将会自动复位单片机且 RSTFC 寄存器中的 LVRF 标志位置位。LVR 包含以下的规格：有效的 LVR 信号，即在 0.9V~V_{LVR} 的低电压状态的时间，必须超过 LVD/LVR 电气特性中 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 V_{LVR} 参数值通过 LVRC 寄存器中的 LVS7~LVS0 位设置，固定为 1.9V。若由于受到干扰 LVS7~LVS0 变为其它值时，需经过一段延迟时间 t_{SRESET} 后才响应复位。此时 RSTFC 寄存器的 LRF 位被置位。上电后 LVRC 的初始值是 01010101B。需要注意的是，当单片机进入深度休眠、休眠或空闲模式，LVR 功能将自动关闭。



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
01010101：1.9V
00110011：1.9V

10011001：1.9V
10101010：1.9V
其它值：MCU 复位 – 寄存器复位为 POR 值

若低电压情况发生且满足以上定义的低电压复位值，则单片机复位。当低电压状况保持时间大于 t_{LVR} 后，响应复位。此时复位后的寄存器内容保持不变。

除了以上定义的低电压复位值外，其它值也能导致单片机复位。需要经过一段延迟时间 t_{SRESET} 才响应复位。但此时寄存器内容将复位为 POR 值。

- **RSTFC 寄存器**

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

“x”：未知

Bit 7~4　未定义，读为“0”

Bit 3　**RSTF**：复位控制寄存器软件复位标志位
具体描述见内部复位控制章节。

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，此位被置为“1”，且只能通过应用程序清零。

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 寄存器包含任何非定义的 LVR 电压值，此位被置为“1”，这类似于软件复位功能，且只能通过应用程序清零。

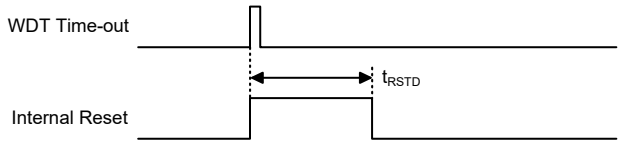
Bit 0　**WRF**：WDT 控制寄存器软件复位标志位
具体描述见看门狗定时器控制寄存器章节。

IAP 复位

当写值“55H”至 FC1 寄存器时，将产生一个复位信号将整个单片机复位。详见 IAP 章节。

正常运行时看门狗溢出复位

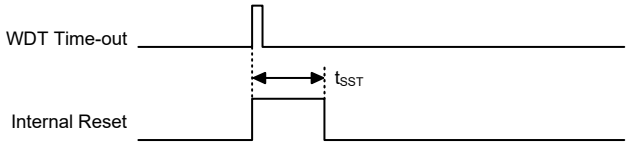
在快速或低速模式下正常运行时，看门狗溢出复位后 TO 将被设为“1”。



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 与 PDF 位被设为“1”外，绝大部分的条件保持不变。图中 t_{SST} 的详细说明请参考系统上电时间电气特性。



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 位存放在状态寄存器中，由深度休眠、休眠或空闲模式功能或看门狗定时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	上电复位或从深度休眠模式唤醒
u	u	快速模式或低速模式时的 LVR 复位
1	u	快速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	复位后清零，看门狗定时器重新计数
定时器模块	所有定时器模块停止
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。

寄存器	BC66F2235	BC66F2245	BC66F2255	上电复位	复位 (从深度休眠唤醒)	WDT 溢出 (正常运行)	WDT 溢出 (休眠 / 空闲)
IAR0	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP0	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
IAR1	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1L	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1H	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
ACC	●	●	●	xxxx xxxx	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	●	●	●	xxxx xxxx	xxxx xxxx	uuuu uuuu	uuuu uuuu
TBLH	●	●	●	xxxx xxxx	xxxx xxxx	uuuu uuuu	uuuu uuuu

寄存器	BC66F2235	BC66F2245	BC66F2255	上电复位	复位 (从深度休眠唤醒)	WDT 溢出 (正常运行)	WDT 溢出 (休眠 / 空闲)
TBHP	●			---- -xxx	---- -xxx	---- -uuu	---- -uuu
		●		---- xxxx	---- xxxx	---- uuuu	---- uuuu
			●	---x xxxx	---x xxxx	---u uuuu	---u uuuu
STATUS	●	●	●	xx00 xxxx	xx00 xxxx	uu1u uuuu	uu11 uuuu
IAR2	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2L	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2H	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
RSTFC	●	●	●	---- 0x00	---- 0x00	---- uuuu	---- 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
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
PB	●	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	●	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBPU	●	●	●	0000 0000	0000 0000	0000 0000	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
INTEG	●	●	●	---- 0000	---- 0000	---- 0000	---- uuuu
SCC	●	●	●	001- 0000	001- 0000	001- 0000	uuu- uuuu
HIRCC	●	●	●	---- --01	---- --01	---- --01	---- --uu
HXTC	●	●	●	---- --00	---- --00	---- --00	---- --uu
LVDC	●	●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
LVRC	●	●	●	0101 0101	0101 0101	0101 0101	uuuu uuuu
WDTC	●	●	●	0101 0011	0101 0011	0101 0011	uuuu uuuu
RSTC	●	●	●	0101 0101	0101 0101	0101 0101	uuuu uuuu
PSC0R	●	●	●	---- --00	---- --00	---- --00	---- --uu
PSC1R	●	●	●	---- --00	---- --00	---- --00	---- --uu
PWRC	●	●	●	00-0 1000	xx-x uuu0	uu-u uuuu	uu-u uuuu
RF_PWR	●	●	●	---- ---0	---- ---0	---- ---0	---- ---u
MFI0	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI1	●	●	●	--00 --00	--00 --00	--00 --00	--uu --uu
MFI2	●	●	●	-000 -000	-000 -000	-000 -000	-uuu -uuu
IFS	●	●	●	---- --00	---- --00	---- --00	---- --uu
SADC0	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu

寄存器	BC66F2235	BC66F2245	BC66F2255	上电复位	复位 (从深度休眠唤醒)	WDT 溢出 (正常运行)	WDT 溢出 (休眠 / 空闲)
SADC1	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADOH	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu (ADRFS=0) ---- uuuu (ADRFS=1)
SADOL	●	●	●	xxxx ----	xxxx ----	xxxx ----	uuuu ---- (ADRFS=0) uuuu uuuu (ADRFS=1)
TB0C	●	●	●	0--- -000	u--- -uuu	0--- -000	u--- -uuu
TB1C	●	●	●	0--- -000	0--- -000	0--- -000	u--- -uuu
PTM0AH	●	●	●	---- --00	---- --00	---- --00	---- --uu
PTM0AL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0C0	●	●	●	0000 0---	0000 0---	0000 0---	uuuu u---
PTM0C1	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0DH	●	●	●	---- --00	---- --00	---- --00	---- --uu
PTM0DL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0RPH	●	●	●	---- --00	---- --00	---- --00	---- --uu
PTM0RPL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM0C0	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM0C1	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM0DL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM0DH	●	●	●	---- --00	---- --00	---- --00	---- --uu
CTM0AL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM0AH	●	●	●	---- --00	---- --00	---- --00	---- --uu
PAS0	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS1	●	●	●	---- --00	---- --00	---- --00	---- --uu
PBS0	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBS1	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCS0	●	●	●	---- --00	---- --00	---- --00	---- --uu
PCS1	●	●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
RF_OPER	●	●	●	--00 ---0	--00 ---0	--00 ---0	--uu ---u
RF_CLK1	●	●	●	1000 ---0	1000 ---0	1000 ---0	uuuu ---u
RF_CLK2	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
RF_FIFO_CTRL1	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
RF_FIFO_CTRL2	●	●	●	0011 1111	0011 1111	0011 1111	uuuu uuuu
RF_FIFO_CTRL3	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
RF_FIFO_CTRL4	●	●	●	0--- 0001	0--- 0001	0--- 0001	u--- uuuu
RF_MOD1	●	●	●	1001 0000	1001 0000	1001 0000	uuuu uuuu

寄存器	BC66F2235	BC66F2245	BC66F2255	上电复位	复位 (从深度休眠唤醒)	WDT 溢出 (正常运行)	WDT 溢出 (休眠 / 空闲)
RF_MOD2	●	●	●	---- -001	---- -001	---- -001	---- -uuu
RF_MOD4	●	●	●	0000 1110	0000 1110	0000 1110	uuuu uuuu
RF_OPMOD	●	●	●	---- -000	---- -000	---- -000	---- -uuu
RF_SX1	●	●	●	-011 0110	-011 0110	-011 0110	-uuu uuuu
RF_SX2	●	●	●	0000 1010	0000 1010	0000 1010	uuuu uuuu
RF_SX3	●	●	●	1101 0111	1101 0111	1101 0111	uuuu uuuu
RF_SX4	●	●	●	---- 0011	---- 0011	---- 0011	---- uuuu
RF_CP3	●	●	●	1100 1010	1100 1010	1100 1010	uuuu uuuu
RF_OD1	●	●	●	0000 0100	0000 0100	0000 0100	uuuu uuuu
RF_VCO1	●	●	●	0001 0000	0001 0000	0001 0000	uuuu uuuu
RF_TX1	●	●	●	1000 1000	1000 1000	1000 1000	uuuu uuuu
RF_TX2	●	●	●	1101 -000	1101 -000	1101 -000	uuuu -uuu
RF_DFC_CAL	●	●	●	00-0 0000	00-0 0000	00-0 0000	uu-u uuuu
RF_LDO	●	●	●	0001 1000	0001 1000	0001 1000	uuuu uuuu
RF_XO1	●	●	●	0001 0101	0001 0101	0001 0101	---u uuuu
SIMC0	●	●	●	1110 0000	1110 0000	1110 0000	uuuu uuuu
SIMC1* (UMD=0)	●	●	●	1000 0001	1000 0001	1000 0001	uuuu uuuu
UUCR1* (UMD=1)	●	●	●	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
SIMC2/SIMA/ UUCR2	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
SIMD/UTXR_RXR	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SIMTOC* (UMD=0)	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
UBRG* (UMD=1)	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
UUCR3	●	●	●	---- ---0	---- ---0	---- ---0	---- ---u
UUSR	●	●	●	0000 1011	0000 1011	0000 1011	uuuu uuuu
CTM1C0	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM1C1	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM1DL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM1DH	●	●	●	---- --00	---- --00	---- --00	---- --uu
CTM1AL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTM1AH	●	●	●	---- --00	---- --00	---- --00	---- --uu
FC0	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC1	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARL	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARH	●			---- -000	---- -000	---- -000	---- -uuu
		●		---- 0000	---- 0000	---- 0000	---- uuuu
			●	---0 0000	---0 0000	---0 0000	---u uuuu
FD0L	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD0H	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu

寄存器	BC66F2235	BC66F2245	BC66F2255	上电复位	复位 (从深度休眠唤醒)	WDT 溢出 (正常运行)	WDT 溢出 (休眠 / 空闲)
FD1L	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD1H	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2L	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD2H	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3L	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD3H	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKTMR	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKC0	●	●	●	0000 0010	uuuu uuuu	0000 0010	uuuu uuuu
TKC1	●	●	●	0000 0011	uuuu uuuu	0000 0011	uuuu uuuu
TKC2	●	●	●	---- -001	---- -uuu	---- -001	---- -uuu
TK16DL	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TK16DH	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM0C0	●	●	●	--0- 0000	--u- uuuu	--0- 0000	--u- uuuu
TKM0C1	●	●	●	0-00 0000	u-uu uuuu	0-00 0000	u-uu uuuu
TKM0C2	●	●	●	1110 0100	uuuu uuuu	1110 0100	uuuu uuuu
TKM016DL	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM016DH	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM0ROL	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM0ROH	●	●	●	---- --00	---- --uu	---- --00	---- --uu
TKM0TH16L	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM0TH16H	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM0THS	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM1C0	●	●	●	--0- 0000	--u- uuuu	--0- 0000	--u- uuuu
TKM1C1	●	●	●	0-00 0000	u-uu uuuu	0-00 0000	u-uu uuuu
TKM1C2	●	●	●	1110 0100	uuuu uuuu	1110 0100	uuuu uuuu
TKM116DL	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM116DH	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM1ROL	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM1ROH	●	●	●	---- --00	---- --uu	---- --00	---- --uu
TKM1TH16L	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM1TH16H	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM1THS	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM2C0	●	●	●	--0- 0000	--u- uuuu	--0- 0000	--u- uuuu
TKM2C1	●	●	●	0-00 0000	u-uu uuuu	0-00 0000	u-uu uuuu
TKM2C2	●	●	●	1110 0100	uuuu uuuu	1110 0100	uuuu uuuu
TKM216DL	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM216DH	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM2ROL	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu

寄存器	BC66F2235	BC66F2245	BC66F2255	上电复位	复位 (从深度休眠唤醒)	WDT 溢出 (正常运行)	WDT 溢出 (休眠 / 空闲)
TKM2ROH	●	●	●	---- --00	---- --uu	---- --00	---- --uu
TKM2TH16L	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM2TH16H	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM2THS	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM3C0	●	●	●	--0- 0000	--u- uuuu	--0- 0000	--u- uuuu
TKM3C1	●	●	●	0-00 0000	u-uu uuuu	0-00 0000	u-uu uuuu
TKM3C2	●	●	●	1110 0100	uuuu uuuu	1110 0100	uuuu uuuu
TKM316DL	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM316DH	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM3ROL	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM3ROH	●	●	●	---- --00	---- --uu	---- --00	---- --uu
TKM3TH16L	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM3TH16H	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu
TKM3THS	●	●	●	0000 0000	uuuu uuuu	0000 0000	uuuu uuuu

注：“u”表示不改变
“x”表示未知
“-”表示未定义
“*”：UUCR1 和 SIMC1 寄存器共用同一个存储器地址，UBRG 和 SIMTOC 寄存器共用同一个存储器地址。复位发生后，通过应用程序手动设置 UMD 位为“1”后可获得 UUCR1 和 UBRG 寄存器的默认值。

输入 / 输出端口

Holtek 单片机的输入 / 输出口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，这些特性也使得此类单片机在广泛应用上都能符合开发的需求。

该系列单片机提供 PA~PC 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	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PBPU	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0

寄存器名称	位							
	7	6	5	4	3	2	1	0
PC	—	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	—	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PCPU	—	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0

“—”：未定义，读为“0”

I/O 逻辑功能寄存器列表

注：1. 此表反映的是最大封装的情况，每个单片机的实际有效位需参考对应封装，详见“引脚图”章节。
2. 对于小封装的单片机，这些寄存器中的无效位虽保留但仍需合理设置以避免输入浮空造成额外耗电，详见“待机电流注意事项”章节。

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，当引脚规划为数字输入时，可由内部连接到一个上拉电阻。这些上拉电阻可通过相应的上拉控制寄存器 PAPU~PCPU 来设置，它用一个 PMOS 晶体管来实现上拉电阻功能。

需要注意的是，当 I/O 引脚设为数字输入或 NMOS 输出时，上拉功能才会受 PxPU 控制开启，其它状态下上拉功能不可用。

- **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 口引脚上拉电阻控制位
0：除能
1：使能

PxPUn 位用于控制引脚上拉电阻功能。这里的 x 可以是端口 A、B 或 C。但是，各单片机的每个 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”迫使单片机进入深度休眠、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。

需要注意的是，只有当引脚选作通用输入类型且单片机处于深度休眠、休眠或空闲模式时，唤醒功能才会受 PAWU 控制开启，其它状态下此唤醒功能不可用。

- **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**：PA 口引脚唤醒功能控制位
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~PCC，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端口寄存器的内容。

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 口引脚类型选择位
0：输出
1：输入
PxCn 位用于选择引脚类型。这里的 x 可以是端口 A、B 或 C。但是，各单片机的每个 I/O 端口实际有效位可能不同。

引脚共用功能

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。此外，这些引脚功能可以通过一系列寄存器进行设定。

引脚共用功能选择寄存器

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，引脚功能共用和引脚功能选择，使得小封装单片机具有更多不同的功能。单片机包含端口“x”输出功能选择寄存器“n”，记为 PxSn，和输入功能选择寄存器，记为 IFS，这些寄存器可以用来选择共用引脚的特定功能。

要注意的最重要一点是，确保所需的引脚共用功能被正确地选择和取消。对于大部分共用功能，要选择所需的引脚共用功能，首先应通过相应的引脚共用控制寄存器正确地选择该功能，然后再配置相应的外围功能设置以使能外围功能。但是，在设置相关引脚控制位时，一些数字输入引脚如 INTn、xTCKn 等，与对应的通用 I/O 口共用同一个引脚共用设置选项。要选择这个引脚功能，除了上述的必要的引脚共用控制和外围功能设置外，还必须将其对应的端口控制寄

存器位设置为输入。要正确地取消引脚共用功能，首先应除能外围功能，然后再修改相应的引脚共用控制寄存器以选择其它的共用功能。

寄存器名称	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	—	—	—	—	—	—	PAS11	PAS10
PBS0	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
PBS1	PBS17	PBS16	PBS15	PBS14	PBS13	PBS12	PBS11	PBS10
PCS0	—	—	—	—	—	—	PCS01	PCS00
PCS1	—	—	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
IFS	—	—	—	—	—	—	INT1PS	INT0PS

引脚共用功能选择寄存器列表

- **PAS0 寄存器**

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
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 **PAS07~PAS06**：PA3 引脚共用功能选择
00：PA3/INT0
01：PA3/INT0
10：KEY16
11：KEY16
注：对于 BC66F2235，这两位为保留位且需固定为“00”。

Bit 5~4 **PAS05~PAS04**：PA2 引脚共用功能选择
00：PA2
01：PTP0
10：KEY15
11：KEY15

Bit 3~2 **PAS03~PAS02**：PA1 引脚共用功能选择
00：PA1
01：PA1
10：KEY14
11：KEY14

Bit 1~0 **PAS01~PAS00**：PA0 引脚共用功能选择
00：PA0
01：PTP0B
10：KEY13
11：KEY13

- **PAS1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PAS11	PAS10
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义，读为“0”

Bit 1~0 **PAS11~PAS10**：PA4 引脚共用功能选择
00：PA4/INT1
01：CTP0
10：SCL/SCK
11：KEY12

- **PBS0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
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 **PBS07~PBS06**：PB3 引脚共用功能选择
00：PB3
01：CTP1B
10：AN3
11：KEY4
注：对于 BC66F2235，这两位为保留位且需固定为“00”。

Bit 5~4 **PBS05~PBS04**：PB2 引脚共用功能选择
00：PB2/INT1
01：PB2/INT1
10：AN2
11：KEY3
注：对于 BC66F2235，这两位为保留位且需固定为“00”。

Bit 3~2 **PBS03~PBS02**：PB1 引脚共用功能选择
00：PB1
01：CTP1
10：AN1
11：KEY2
注：对于 BC66F2235，这两位为保留位且需固定为“00”。

Bit 1~0 **PBS01~PBS00**：PB0 引脚共用功能选择
00：PB0
01：PB0
10：AN0
11：KEY1

- **PBS1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PBS17	PBS16	PBS15	PBS14	PBS13	PBS12	PBS11	PBS10
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 **PBS17~PBS16**：PB7 引脚共用功能选择
00：PB7
01：PB7
10：KEY8
11：KEY8
注：对于 BC66F2235 和 BC66F2245，这两位为保留位且需固定为“00”。

Bit 5~4 **PBS15~PBS14**：PB6 引脚共用功能选择
00：PB6
01：PB6
10：KEY7
11：KEY7
注：对于 BC66F2235 和 BC66F2245，这两位为保留位且需固定为“00”。

Bit 3~2 **PBS13~PBS12**：PB5 引脚共用功能选择
00：PB5/CTCK1
01：PB5/CTCK1
10：KEY6
11：KEY6

Bit 1~0 **PBS11~PBS10**：PB4 引脚共用功能选择
00：PB4
01：CTP0B
10：KEY5
11：KEY5

- **PCS0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PCS01	PCS00
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义，读为“0”

Bit 1~0 **PCS01~PCS00**：PC0 引脚共用功能选择
00：PC0
01：CTP0
10：CTP0
11：CTP0
注：对于 BC66F2235，这两位为保留位且需固定为“00”。

- **PCS1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6　未定义，读为“0”

Bit 5~4　**PCS15~PCS14**：PC6 引脚共用功能选择
00：PC6
01：PC6
10：SDI/URX/UTX/SDA
11：KEY11

Bit 3~2　**PCS13~PCS12**：PC5 引脚共用功能选择
00：PC5
01：PC5
10：SDO/UTX
11：KEY10
注：对于 BC66F2235，这两位为保留位且需固定为“00”。

Bit 1~0　**PCS11~PCS10**：PC4 引脚共用功能选择
00：PC4
01：PC4
10：$\overline{\text{SCS}}$
11：KEY9
注：对于 BC66F2235，这两位为保留位且需固定为“00”。

- **IFS 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	INT1PS	INT0PS
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

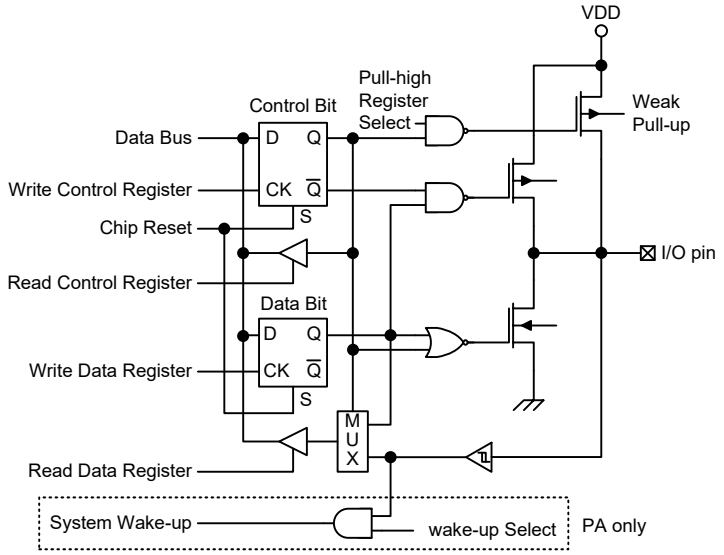
Bit 7~2　未定义，读为“0”

Bit 1　**INT1PS**：INT1 输入源引脚选择
BC66F2235：
0：PA4
1：保留
BC66F2245/BC66F2255：
0：PA4
1：PB2

Bit 0　**INT0PS**：INT0 输入源引脚选择
BC66F2245：
0：PA3
1：保留
BC66F2255：
0：PA3
1：PA5
注：对于 BC66F2235，这两位为保留位且需固定为“00”。

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 引脚逻辑功能的理解提供一个参考。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。



逻辑功能输入 / 输出结构

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将某些引脚设定为输出状态，这些输出引脚会有初始高电平输出，除非端口数据寄存器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对应的端口控制寄存器，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。单片机处于深度休眠、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。该系列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 / 事件计数器，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和周期型定时器章节。

简介

该系列单片机包含多个 TM，每个 TM 可被划分为一个特定的类型，即简易型 TM 和周期型 TM。虽然性质相似，但不同 TM 特性复杂度不同。本章介绍简易型和周期型 TM 的共性，更多详细资料分别见后面各章。这两种类型 TM 的特性和区别见下表。

TM 功能	CTM	PTM
定时 / 计数器	√	√
比较匹配输出	√	√
PWM 输出	√	√
单脉冲输出	—	√
PWM 对齐方式	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

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 xTMn 控制寄存器的 xTnCK2~xTnCK0 位，选择所需的时钟源，其中 x 代表 C 或 P 型 TM，n 代表具体 TM 编号。该时钟源来自系统时钟 f_{SYS} 的分频比或内部高速时钟 f_H 或 f_{SUB} 时钟源或外部 xTCKn 引脚。xTCKn 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

简易型和周期型 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

每个 TM 没有或有一个 TM 输入引脚 xTCKn。xTMn 输入引脚 xTCKn 作为 xTMn 时钟源输入脚，通过设置 xTMnC0 寄存器中的 xTnCK2~xTnCK0 位进行选择。外部时钟源可通过该引脚来驱动内部 TM。xTCKn 引脚可选择上升沿有效或下降沿有效。PTCKn 引脚还可用作 PTMn 单脉冲模式的外部触发引脚。

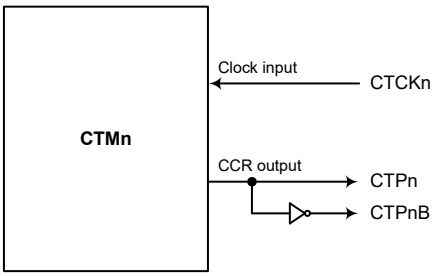
每个 TM 没有或有两个输出引脚 xTPn 和 xTPnB。当 TM 工作在比较匹配输出模式且比较匹配发生时，这些引脚会由 TM 控制切换到高电平或低电平或翻转。外部 xTPn 和 xTPnB 输出引脚也被 TM 用来产生 PWM 输出波形。

当 TM 输入和输出引脚与其它功能共用时，TM 输入和输出功能需要事先通过相关引脚共用功能选择寄存器先被设置。更多引脚共用功能选择详见引脚共用功能章节。

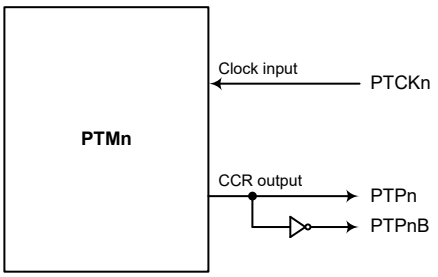
各 TM 引脚的数量不同，详见下表。

单片机型号	CTM		PTM	
	输入	输出	输入	输出
BC66F2235	—; CTCK1	CTP0, CTP0B; —	—	PTP0, PTP0B
BC66F2245	—; CTCK1	CTP0, CTP0B; CTP1, CTP1B	—	PTP0, PTP0B
BC66F2255	CTCK0; CTCK1	CTP0, CTP0B; CTP1, CTP1B	PTCK0	PTP0, PTP0B

TM 外部引脚



CTM 功能引脚方框图 (n=0~1)

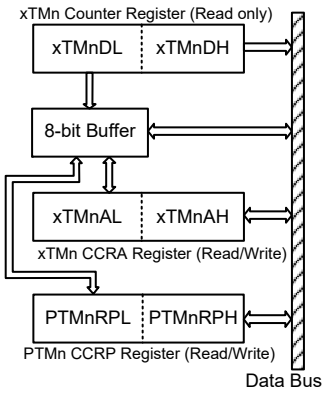


PTM 功能引脚方框图 (n=0)

编程注意事项

TM 计数寄存器和比较寄存器 CCRA 和 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。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”指令按照以下步骤访问 CCRA 和 CCRP 低字节寄存器，即 xTMnAL 和 PTMnRPL，否则可能导致无法预期的结果。



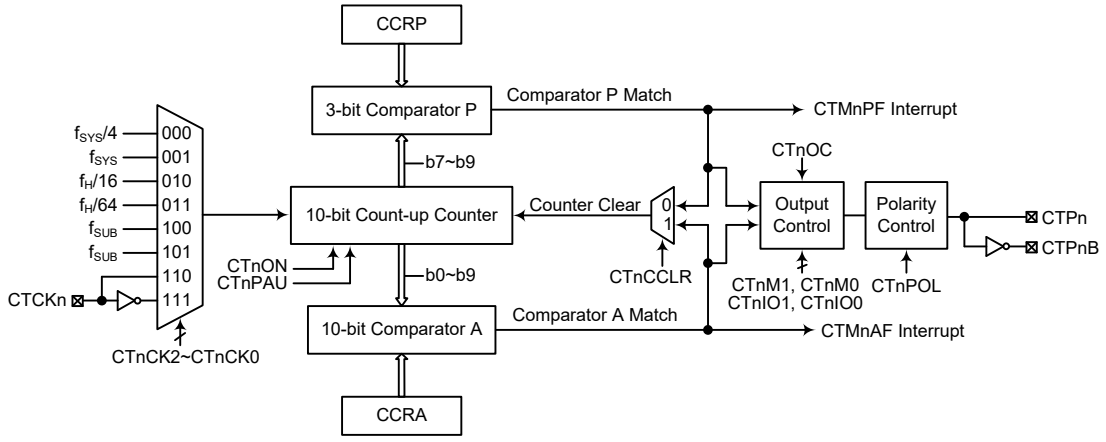
读写流程如下步骤所示：

- 写数据至 CCRA 或 CCRP
  - ♦ 步骤 1. 写数据至低字节寄存器 xTMnAL 或 PTMnRPL
    – 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 xTMnAH 或 PTMnRPH
    – 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 xTMnDH、xTMnAH 或 PTMnRPH 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 xTMnDL、xTMnAL 或 PTMnRPL 读取数据
    – 注意，此时读取 8-bit 缓存器中的数据。

简易型 TM – CTM

简易型 TM 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两个外部输出脚。

单片机型号	CTM 核心	CTM 输入引脚	CTM 输出引脚
BC66F2235	10-bit CTM (CTM0, CTM1)	—; CTCK1	CTP0, CTP0B; —
BC66F2245		—; CTCK1	CTP0, CTP0B; CTP1, CTP1B
BC66F2255		CTCK0; CTCK1	CTP0, CTP0B; CTP1, CTP1B



注：CTMn 外部引脚与其它功能共用引脚，因此在使用 CTMn 之前应该合理配置相关引脚共用功能选择寄存器以确保使能 CTMn 引脚功能。对于 CTCKn 引脚还需设置相应的端口控制寄存器，将该引脚设置为输入口。

10-bit 简易型 TM 方框图 (n=0~1)

简易型 TM 操作

简易型 TM 是 10 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 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 CCRP 和 CCRA 寄存器中的值进行比较。CCRP 是 3 位的，与计数器的高 3 位比较；而 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 CTnON 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 CTMn 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 10 位计数器的值，一对读 / 写寄存器存放 10 位 CCRA 的值，剩下两个控制寄存器设置不同的操作和控制模式以及 CCRP 的 3 个位。

寄存器名称	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CTMnC0	CTnPAU	CTnCK2	CTnCK1	CTnCK0	CTnON	CTnRP2	CTnRP1	CTnRP0
CTMnC1	CTnM1	CTnM0	CTnIO1	CTnIO0	CTnOC	CTnPOL	CTnDPX	CTnCCLR
CTMnDL	D7	D6	D5	D4	D3	D2	D1	D0
CTMnDH	—	—	—	—	—	—	D9	D8
CTMnAL	D7	D6	D5	D4	D3	D2	D1	D0
CTMnAH	—	—	—	—	—	—	D9	D8

10-bit 简易型 TM 寄存器列表 (n=0~1)

- **CTMnC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	CTnPAU	CTnCK2	CTnCK1	CTnCK0	CTnON	CTnRP2	CTnRP1	CTnRP0
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 **CTnPAU**：CTMn 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，CTMn 保持上电状态并继续耗电。当此位由低到高转换时，计数器将保持其当前计数值，直到此位再次改变为低电平，从此值开始继续计数。

Bit 6~4 **CTnCK2~CTnCK0**：CTMn 计数时钟选择位
000：$f_{SYS}/4$
001：f_{SYS}
010：$f_H/16$
011：$f_H/64$
100：f_{SUB}
101：f_{SUB}
110：CTCKn 上升沿时钟
111：CTCKn 下降沿时钟
此三位用于选择 CTMn 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，细节方面请参考工作模式和系统时钟章节。

Bit 3 **CTnON**：CTMn 计数器 On/Off 控制位
0：Off
1：On
此位控制 CTMn 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 CTMn。清零此位将停止计数器并关闭 CTMn 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当前计数值，直到此位再次改变为高电平。
若 CTMn 处于比较匹配输出模式或 PWM 输出模式，当 CTnON 位经由低到高转换时，CTMn 输出脚将复位至 CTnOC 位指定的初始值。

Bit 2~0 **CTnRP2~CTnRP0**：CTMn CCRP 3-bit 寄存器，与 CTMn 计数器 bit 9 ~ bit 7 比较比较器 P 匹配周期
000：1024 个 CTMn 时钟周期
001~111：(1~7) × 128 个 CTMn 时钟周期
此三位设定内部 CCRP 3-bit 寄存器的值，然后与内部计数器的高三位进行比较。如果 CTnCCLR 位设定为 0 时，此比较结果可用于清零内部计数器。CTnCCLR 位设为低，内部计数器在比较器 P 比较匹配发生时被重置；由于 CCRP 只与计数器高三位比较，比较结果是 128 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。

- **CTMnC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	CTnM1	CTnM0	CTnIO1	CTnIO0	CTnOC	CTnPOL	CTnDPX	CTnCCLR
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 **CTnM1~CTnM0**：CTMn 工作模式选择位
00：比较匹配输出模式
01：未定义
10：PWM 输出模式
11：定时 / 计数器模式
这两位设置 CTMn 需要的工作模式。为了确保操作可靠，CTMn 应在 CTnM1 和 CTnM0 位有任何改变前先关掉。在定时 / 计数器模式，CTMn 输出脚状态未定义。

Bit 5~4 **CTnIO1~CTnIO0**：CTMn 外部引脚功能选择位
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 CTMn 外部引脚如何改变状态。这两位值的选择取决于 CTMn 运行在哪种模式下。
在比较匹配输出模式下，CTnIO1 和 CTnIO0 位决定当比较器 A 比较匹配输出发生时 CTMn 输出脚如何改变状态。当比较器 A 比较匹配输出发生时 CTMn 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。CTMn 输出脚的初始值通过 CTMnC1 寄存器的 CTnOC 位设置取得。注意，由 CTnIO1 和 CTnIO0 位得到的输出电平必须与通过 CTnOC 位设置的初始值不同，否则当比较匹配发生时，CTMn 输出脚将不会发生变化。在 CTMn 输出脚改变状态后，通过 CTnON 位由低到高电平的转换复位至初始值。
在 PWM 输出模式，CTnIO1 和 CTnIO0 用于决定比较匹配条件发生时怎样改变 CTMn 输出脚的状态。PWM 输出功能通过这两位的变化进行更新。只可在 CTMn 关闭时改变 CTnIO1 和 CTnIO0 位的值。若在 CTMn 运行时改变 CTnIO1 和 CTnIO0 的值，PWM 输出的值是无法预料的。

Bit 3 **CTnOC**：CTMn CTPn 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 CTMn 输出脚输出控制位。它取决于 CTMn 此时正运行于比较匹配输出模式还是 PWM 输出模式。若 CTMn 处于定时 / 计数器模式，此位不起作用。在比较匹配输出模式时，比较匹配发生前其决定 CTMn 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 **CTnPOL**：CTMn CTPn 输出极性控制位
0：同相
1：反相
此位控制 CTPn 输出脚的极性。此位为高时 CTMn 输出脚反相，为低时 CTMn 输出脚同相。若 CTMn 处于定时 / 计数器模式时此位不起作用。

Bit 1 **CTnDPX**：CTMn 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 **CTnCCLR**：选择 CTMn 计数器清零条件位
0：CTMn 比较器 P 匹配
1：CTMn 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。CTnCCLR 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。CTnCCLR 位在 PWM 输出模式时未使用。

- **CTMnDL 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**：CTMn 计数器低字节寄存器 bit 7 ~ bit 0
CTMn 10-bit 计数器 bit 7 ~ bit 0

- **CTMnDH 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**：CTMn 计数器高字节寄存器 bit 1 ~ bit 0
CTMn 10-bit 计数器 bit 9 ~ bit 8

- **CTMnAL 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**：CTMn CCRA 低字节寄存器 bit 7 ~ bit 0
CTMn 10-bit CCRA bit 7 ~ bit 0

- **CTMnAH 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**：CTMn CCRA 高字节寄存器 bit 1 ~ bit 0
CTMn 10-bit CCRA bit 9 ~ bit 8

简易型 TM 工作模式

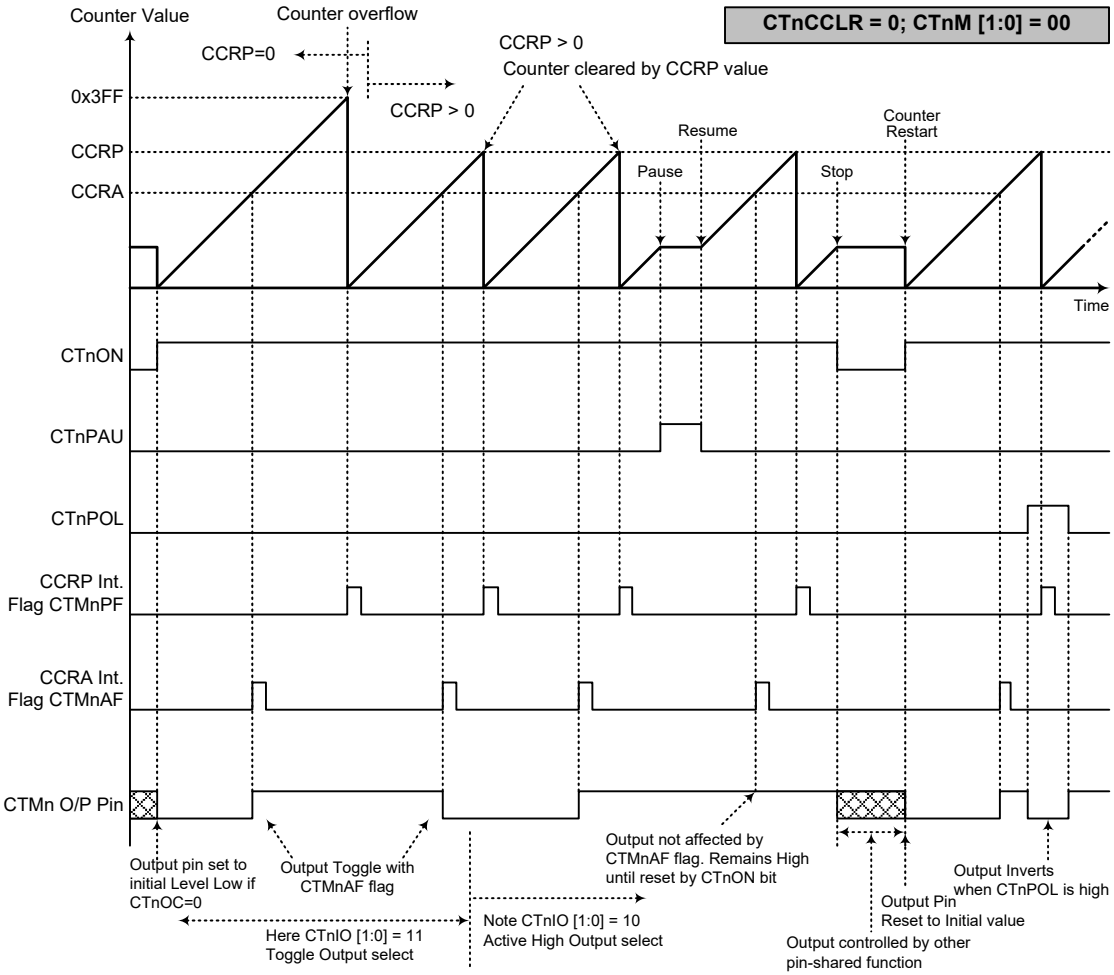
简易型 TM 有三种工作模式，即比较匹配输出模式，PWM 输出模式或定时 / 计数器模式。通过设置 CTMnC1 寄存器的 CTnM1 和 CTnM0 位选择任意工作模式。

比较匹配输出模式

为使 CTMn 工作在此模式，CTMnC1 寄存器中的 CTnM1 和 CTnM0 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 CTnCCLR 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 CTMnAF 和 CTMnPF 将分别置起。

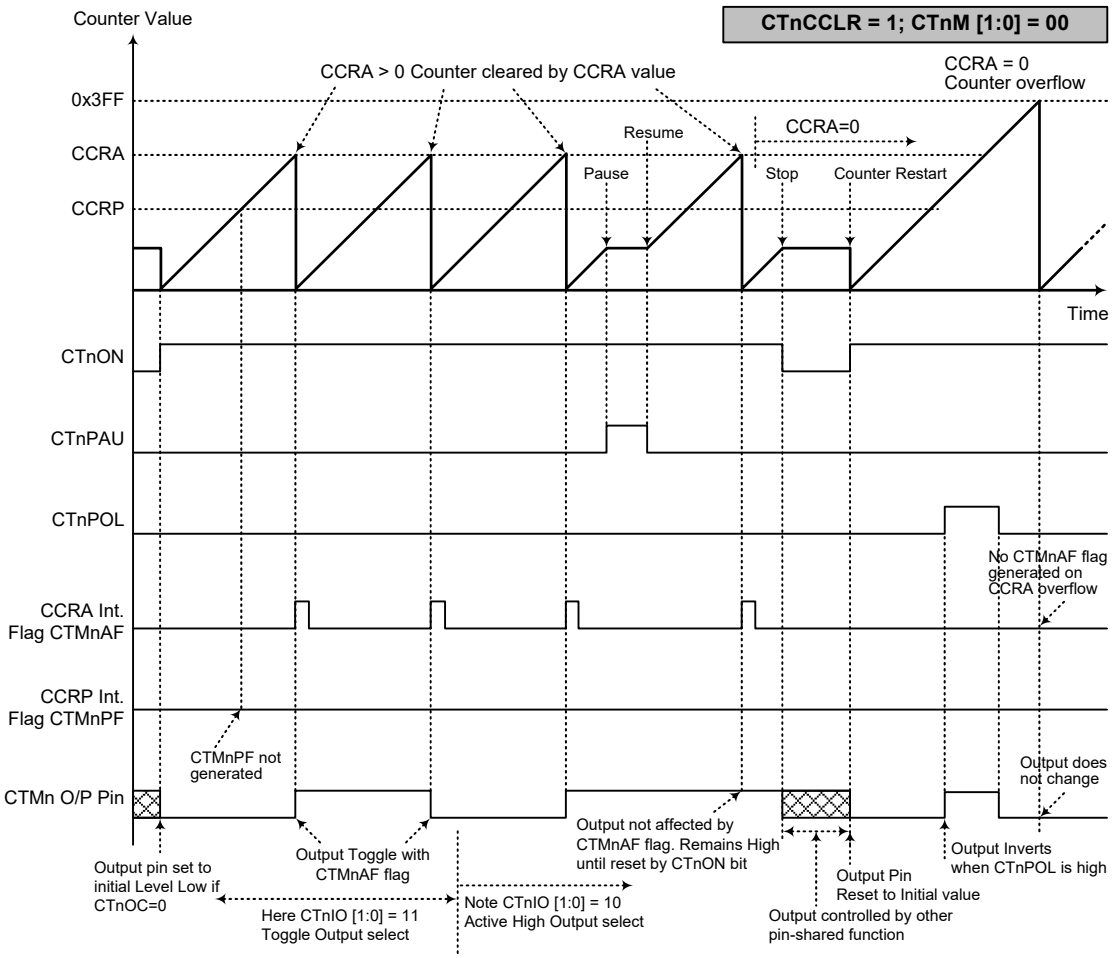
如果 CTMnC1 寄存器的 CTnCCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅 CTMnAF 中断请求标志产生。所以当 CTnCCLR 为高时，不产生 CTMnPF 中断请求标志。如果 CCRA 被清零，当计数达到 10 位最大值 3FFH 时，计数器溢出，而此时不产生 CTMnAF 请求标志。

正如该模式名所言，当比较匹配发生后，CTMn 输出脚状态改变。当比较器 A 比较匹配发生后 CTMnAF 标志产生时，CTMn 输出脚状态改变。比较器 P 比较匹配发生时产生的 CTMnPF 标志不影响 CTMn 输出脚。CTMn 输出脚状态改变方式由 CTMnC1 寄存器中 CTnIO1 和 CTnIO0 位决定。当比较器 A 比较匹配发生时，CTnIO1 和 CTnIO0 位决定 TM 输出脚输出高，低或翻转当前状态。在 CTnON 位由低到高电平的变化后，CTMn 输出脚初始状态为 CTnOC 位所指定的电平。注意，若 CTnIO1 和 CTnIO0 位同时为 0 时，引脚输出不变。



比较匹配输出模式 – CTnCCLR=0 (n=0~1)

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



比较匹配输出模式 – CTnCCLR=1 (n=0~1)

注：1. CTnCCLR=1，比较器 A 匹配将清除计数器
2. CTMn 输出脚仅由 CTMnAF 标志位控制
3. 在 CTnON 上升沿 CTMn 输出脚复位至初始值
4. 当 CTnCCLR=1 时，CTMnPF 标志位不会产生

定时 / 计数器模式

为使 CTMn 工作在此模式，CTMnC1 寄存器中的 CTnM1 和 CTnM0 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 CTMn 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 CTMn 输出脚用作普通 I/O 脚或其它功能。

PWM 输出模式

为使 CTMn 工作在此模式，CTMnC1 寄存器中的 CTnM1 和 CTnM0 位需要设置为“10”。CTMn 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 CTMn 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，CTnCCLR 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 CTMnC1 寄存器的 CTnDPX 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。CTMnC1 寄存器中的 CTnOC 位决定 PWM 波形的极性，CTnIO1 和 CTnIO0 位使能 PWM 输出或将 CTMn 输出脚置为逻辑高或逻辑低。CTnPOL 位对 PWM 输出波形的极性取反。

- **10-bit CTMn，PWM 输出模式，边沿对齐模式，CTnDPX=0**

CCRP	1~7	0
Period	CCRP×128	1024
Duty	CCRA	

若 f_{SYS}=16MHz，CTMn 时钟源选择 f_{SYS}/4，CCRP=4，CCRA=128，

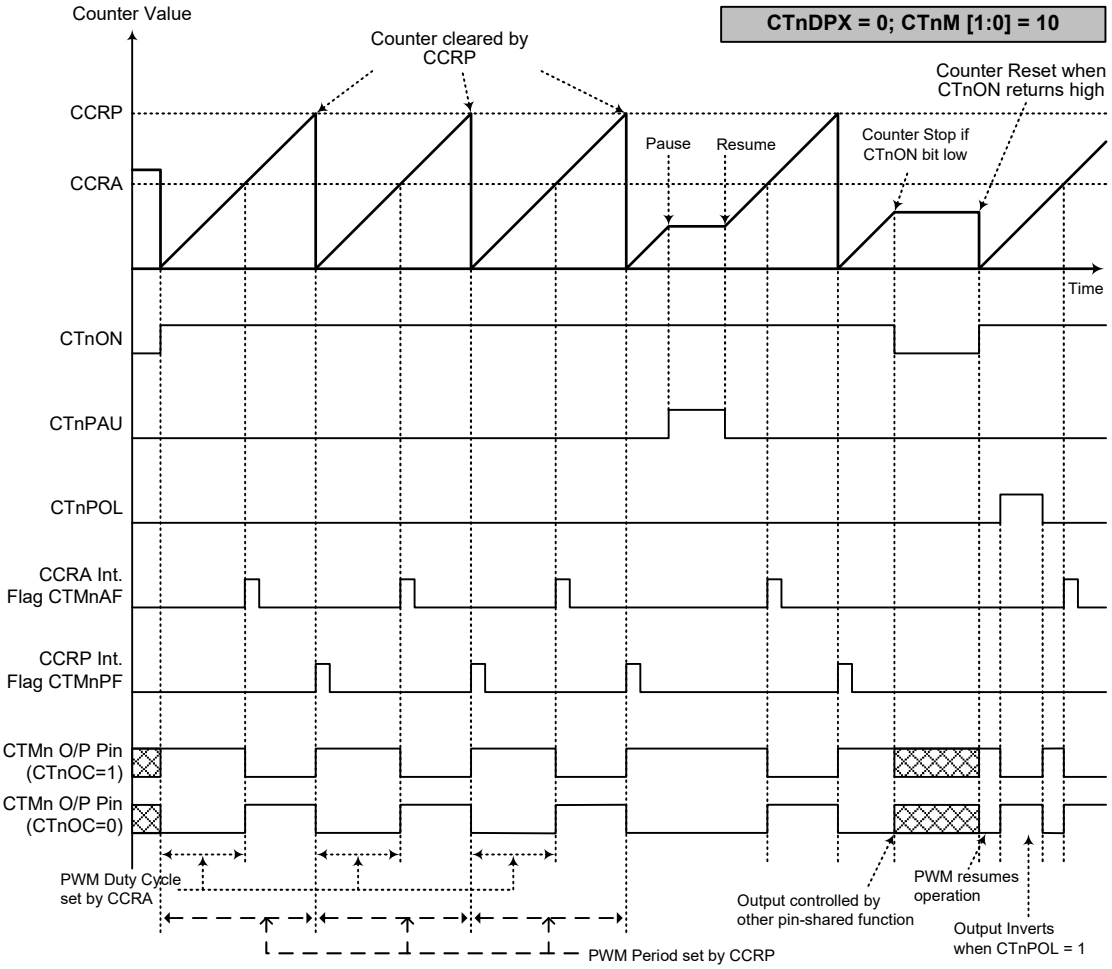
CTMn PWM 输出频率 = (f_{SYS}/4)/(4×128)=f_{SYS}/2048=8kHz，Duty=128/(4×128)=25%

若由 CCRA 寄存器定义的 Duty 值等于或大于 Period 值，PWM 输出占空比为 100%

- **10-bit CTMn，PWM 输出模式，边沿对齐模式，CTnDPX=1**

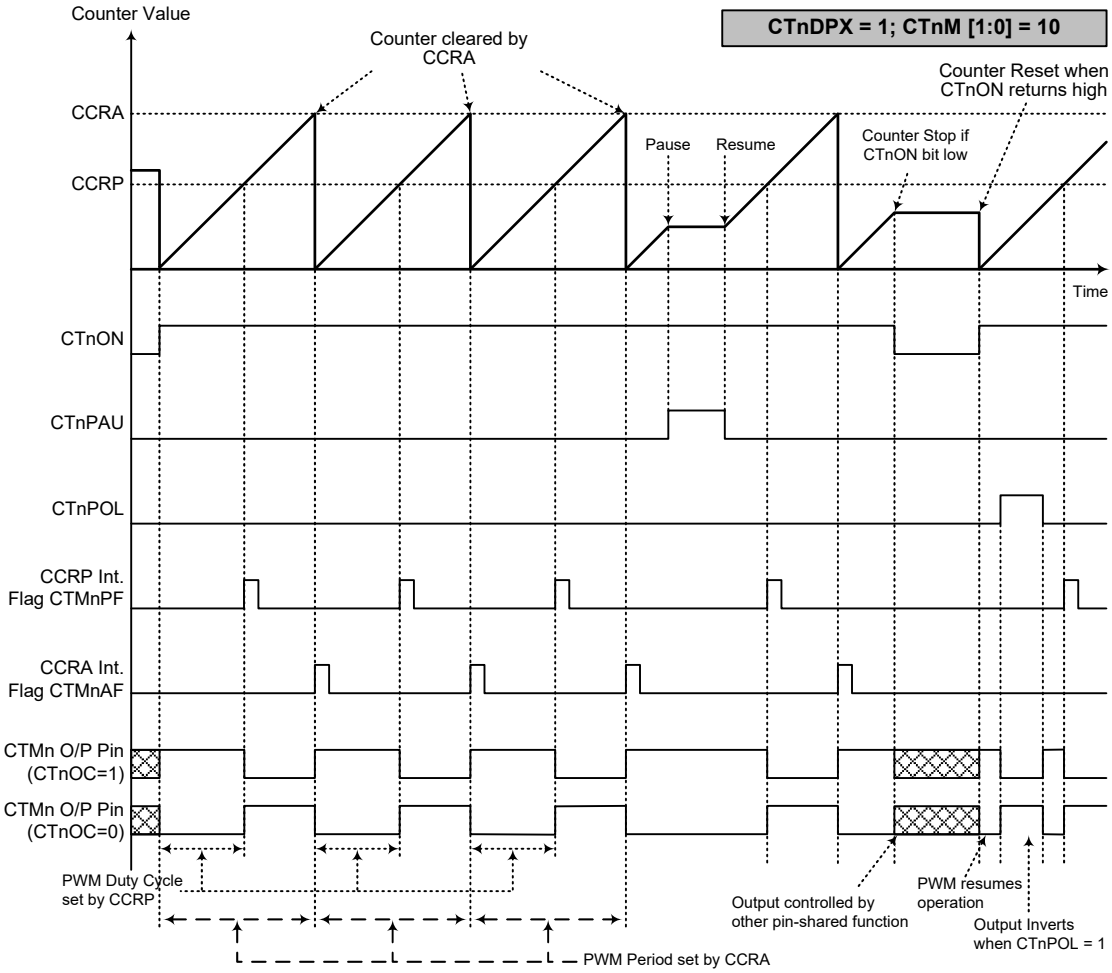
CCRP	1~7	0
Period	CCRA	
Duty	CCRP×128	1024

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



PWM 输出模式 – CTnDPX=0 (n=0~1)

注：1. CTnDPX=0，CCRP 清除计数器
2. 计数器清零并设置 PWM 周期
3. 当 CTnIO[1:0]=00 或 01，PWM 功能不变
4. CTnCCLR 位不影响 PWM 操作



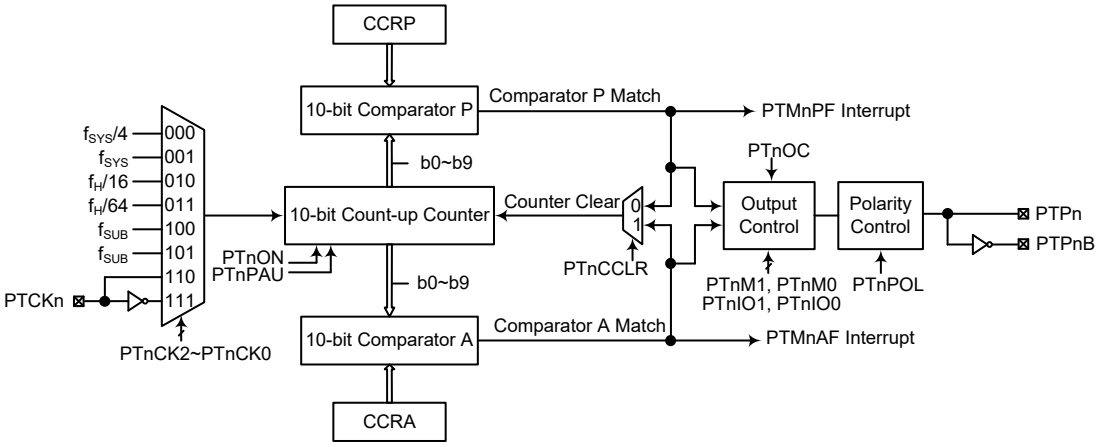
PWM 输出模式 – CTnDPX=1 (n=0~1)

注：1. CTnDPX=1，CCRA 清除计数器
2. 计数器清零并设置 PWM 周期
3. 当 CTnIO[1:0]=00 或 01，PWM 功能不变
4. CTnCCLR 位不影响 PWM 操作

周期型 TM – PTM

周期型 TM 包括四种工作模式，即比较匹配输出、定时 / 事件计数器、单脉冲输出和 PWM 输出模式。周期型 TM 由一个外部输入脚控制并驱动两个外部输出脚。

单片机型号	PTM 核心	PTM 输入引脚	PTM 输出引脚
BC66F2235	10-bit PTM (PTM0)	—	PTP0, PTP0B
BC66F2245		—	PTP0, PTP0B
BC66F2255		PTCK0	PTP0, PTP0B



注：PTMn 外部引脚与其它功能共用引脚，因此在使用 PTMn 功能前，需确保已通过相关的引脚共用功能选择寄存器选择了 PTMn 引脚功能。对于 PTCKn 引脚还需设置相应的端口控制寄存器，将该引脚设置为输入口。

10-bit 周期型 TM 方框图 (n=0)

周期型 TM 操作

周期型 TM 是 10 位宽度，核心是一个由用户选择的内部时钟源驱动的 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 PTnON 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 PTMn 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
PTMnC0	PTnPAU	PTnCK2	PTnCK1	PTnCK0	PTnON	—	—	—
PTMnC1	PTnM1	PTnM0	PTnIO1	PTnIO0	PTnOC	PTnPOL	D1	PTnCCLR
PTMnDL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnDH	—	—	—	—	—	—	D9	D8
PTMnAL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnAH	—	—	—	—	—	—	D9	D8
PTMnRPL	PTnRP7	PTnRP6	PTnRP5	PTnRP4	PTnRP3	PTnRP2	PTnRP1	PTnRP0
PTMnRPH	—	—	—	—	—	—	PTnRP9	PTnRP8

10-bit 周期型 TM 寄存器列表 (n=0)

- **PTMnC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PTnPAU	PTnCK2	PTnCK1	PTnCK0	PTnON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7 **PTnPAU**：PTMn 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。当处于暂停状态时，PTMn 保持上电状态并继续耗电。当此位由低到高转变时，计数器将保持其当前计数值，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 **PTnCK2~PTnCK0**：PTMn 计数时钟选择位
000：$f_{SYS}/4$
001：f_{SYS}
010：$f_H/16$
011：$f_H/64$
100：f_{SUB}
101：f_{SUB}
110：PTCKn 上升沿
111：PTCKn 下降沿
此三位用于选择 PTMn 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，细节方面请参考工作模式和系统时钟章节。

Bit 3 **PTnON**：PTMn 计数器 On/Off 控制位
0：Off
1：On
此位控制 PTMn 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 PTMn。清零此位将停止计数器并关闭 PTMn 减少耗电。当此位经由低到高转变时，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当前计数值，直到此位再次改变为高电平。
若 PTMn 处于比较匹配输出模式、PWM 输出模式或单脉冲输出模式，当 PTnON 位经由低到高转换时，PTMn 输出脚将复位至 PTnOC 位指定的初始值。

Bit 2~0 未定义，读为“0”

- **PTMnC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PTnM1	PTnM0	PTnIO1	PTnIO0	PTnOC	PTnPOL	D1	PTnCCLR
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 **PTnM1~PTnM0**：选择 PTMn 工作模式
00：比较匹配输出模式
01：未定义
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 PTMn 需要的工作模式。为了确保操作可靠，PTMn 应在 PTnM1 和 PTnM0 位有任何改变前先关掉。在定时 / 计数器模式，PTMn 输出脚状态为未知。

Bit 5~4 **PTnIO1~PTnIO0**：选择 PTMn 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
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 PTMn 外部引脚如何改变状态。这两位值的选择取决于 PTMn 运行在哪种模式下。
在比较匹配输出模式下，PTnIO1 和 PTnIO0 位决定当从比较器 A 比较匹配输出发生时 PTMn 输出脚如何改变状态。当从比较器 A 比较匹配输出发生时 PTMn 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。PTMn 输出脚的初始值通过 PTMnC1 寄存器的 PTnOC 位设置取得。注意，由 PTnIO1 和 PTnIO0 位得到的输出电平必须与通过 PTnOC 位设置的初始值不同，否则当比较匹配发生时，PTMn 输出脚将不会发生变化。在 PTMn 输出脚改变状态后，通过 PTnON 位由低到高电平的转换复位至初始值。
在 PWM 输出模式，PTnIO1 和 PTnIO0 用于决定比较匹配条件发生时怎样改变 PTMn 输出脚的状态。PWM 输出功能通过这两位的变化进行更新。只可在 PTMn 关闭时改变 PTnIO1 和 PTnIO0 位的值。若在 PTMn 运行时改变 PTnIO1 和 PTnIO0 的值，PWM 输出的值是无法预料的。

Bit 3 **PTnOC**：PTMn PTPn 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 PTMn 输出脚输出控制位。它取决于 PTMn 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 PTMn 处于定时 / 计数器模式，此位不起作用。在比较匹配输出模式时，其决定比较匹配发生前 PTMn 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 PTnON 位由低变高时 PTMn 输出脚的逻辑电平。

Bit 2 **PTnPOL**：PTMn PTPn 输出极性控制位
0：同相
1：反相
此位控制 PTPn 输出脚的极性。此位为高时 PTMn 输出脚反相，为低时 PTMn 输出脚同相。若 PTMn 处于定时 / 计数器模式时其不受影响。

Bit 1 **D1**：保留位，必须固定为“0”

Bit 0 **PTnCCLR**：选择 PTMn 计数器清零条件位
0：PTMn 比较器 P 匹配
1：PTMn 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，两者都可以用作清除内部计数器。PTnCCLR 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。PTnCCLR 位在 PWM 输出模式或单脉冲输出模式时未使用。

- **PTMnDL 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**：PTMn 计数器低字节寄存器 bit 7 ~ bit 0
PTMn 10-bit 计数器 bit 7 ~ bit 0

- **PTMnDH 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**：PTMn 计数器高字节寄存器 bit 1 ~ bit 0
PTMn 10-bit 计数器 bit 9 ~ bit 8

- **PTMnAL 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**：PTMn CCRA 低字节寄存器 bit 7 ~ bit 0
PTMn 10-bit CCRA bit 7 ~ bit 0

- **PTMnAH 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**：PTMn CCRA 高字节寄存器 bit 1 ~ bit 0
PTMn 10-bit CCRA bit 9 ~ bit 8

- **PTMnRPL 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PTnRP7	PTnRP6	PTnRP5	PTnRP4	PTnRP3	PTnRP2	PTnRP1	PTnRP0
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 **PTnRP7~PTnRP0**：PTMn CCRP 低字节寄存器 bit 7 ~ bit 0
PTMn 10-bit CCRP bit 7 ~ bit 0

- **PTMnRPH 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PTnRP9	PTnRP8
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义读为“0”

Bit 1~0 **PTnRP9~PTnRP8**：PTMn CCRP 高字节寄存器 bit 1 ~ bit 0
PTMn 10-bit CCRP bit 9 ~ bit 8

周期型 TM 工作模式

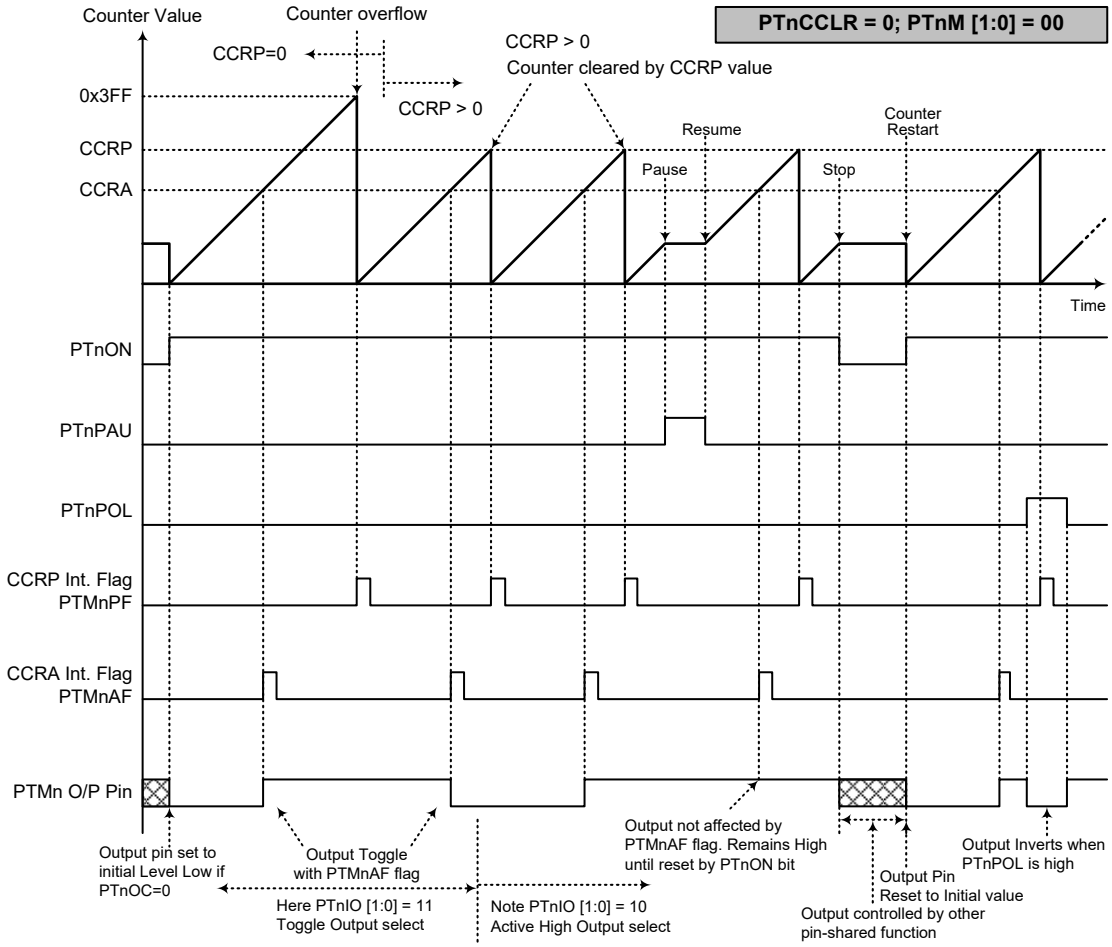
周期型 TM 有四种工作模式，即比较匹配输出模式、PWM 输出模式、单脉冲输出模式或定时 / 计数器模式。通过设置 PTMnC1 寄存器的 PTnM1 和 PTnM0 位选择任意模式。

比较匹配输出模式

为使 PTMn 工作在此模式，PTMnC1 寄存器的 PTnM1 和 PTnM0 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 PTnCCLR 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 PTMnAF 和 PTMnPF 将分别置起。

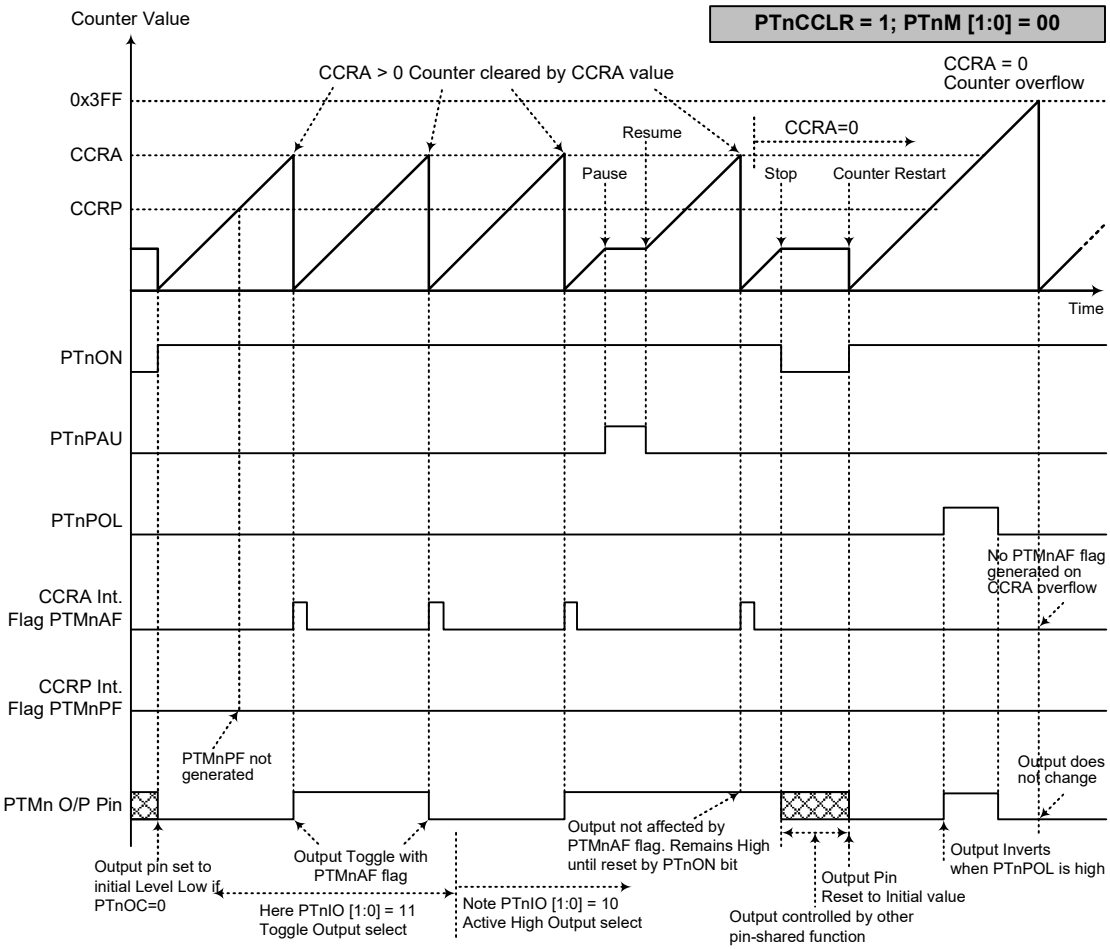
如果 PTMnC1 寄存器的 PTnCCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅 PTMnAF 中断请求标志产生。所以当 PTnCCLR 为高时，不会产生 PTMnPF 中断请求标志。在比较匹配输出模式中，CCRA 寄存器值不能设为“0”。如果 CCRA 位都清除为零，当计数器的值达到 10 位最大值 3FFH 时将溢出，但此时不会产生 PTMnAF 中断请求标志。

正如该模式名所言，当比较匹配发生后，PTMn 输出脚状态改变。当比较器 A 比较匹配发生后 PTMnAF 中断请求标志产生时，PTMn 输出脚状态改变。比较器 P 比较匹配发生时产生的 PTMnPF 标志不影响 PTMn 输出脚。PTMn 输出脚状态改变方式由 PTMnC1 寄存器中 PTnIO1 和 PTnIO0 位决定。当比较器 A 比较匹配发生时，PTnIO1 和 PTnIO0 位决定 PTMn 输出脚输出高，低或翻转当前状态。在 PTnON 位由低到高后，PTMn 输出脚初始状态为 PTnOC 位所指定的电平。注意，若 PTnIO1 和 PTnIO0 位同时为 0 时，引脚输出不变。



比较器匹配输出模式 – PTnCCLR=0 (n=0)

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



比较器匹配输出模式 – PTnCCLR=1 (n=0)

注：1. PTnCCLR=1，比较器 P 匹配将清除计数器
2. PTMn 输出脚仅由 PTMnAF 标志位控制
3. 在 PTnON 上升沿 PTMn 输出脚复位至初始值
4. 当 PTnCCLR=1 时，不会产生 PTMnPF 标志

定时 / 计数器模式

为使 PTMn 工作在此模式，PTMnC1 寄存器的 PTnM1 和 PTnM0 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 PTMn 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 PTMn 输出脚用作普通 I/O 脚或其它功能。

PWM 输出模式

为使 PTMn 工作在此模式，PTMnC1 寄存器的 PTnM1 和 PTnM0 位需要设置为“10”，且 PTnIO1 和 PTnIO0 位也需要设置为“10”。PTMn 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 PTMn 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，PTnCCLR 位对 PWM 周期无影响。CCRP 和 CCRA 寄存器都用于控制 PWM 方波。CCRP 寄存器通过清除内部计数从而控制 PWM 周期，CCRA 寄存器设置 PWM 的占空比。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，CCRA 和 CCRP 中断标志位分别产生。PTMnC1 寄存器的 PTnOC 位选择 PWM 波形的极性，PTnIO1 和 PTnIO0 位使能 PWM 输出或强制 PTMn 输出脚为高电平或低电平。PTnPOL 位用于 PWM 输出波形的极性反相控制。

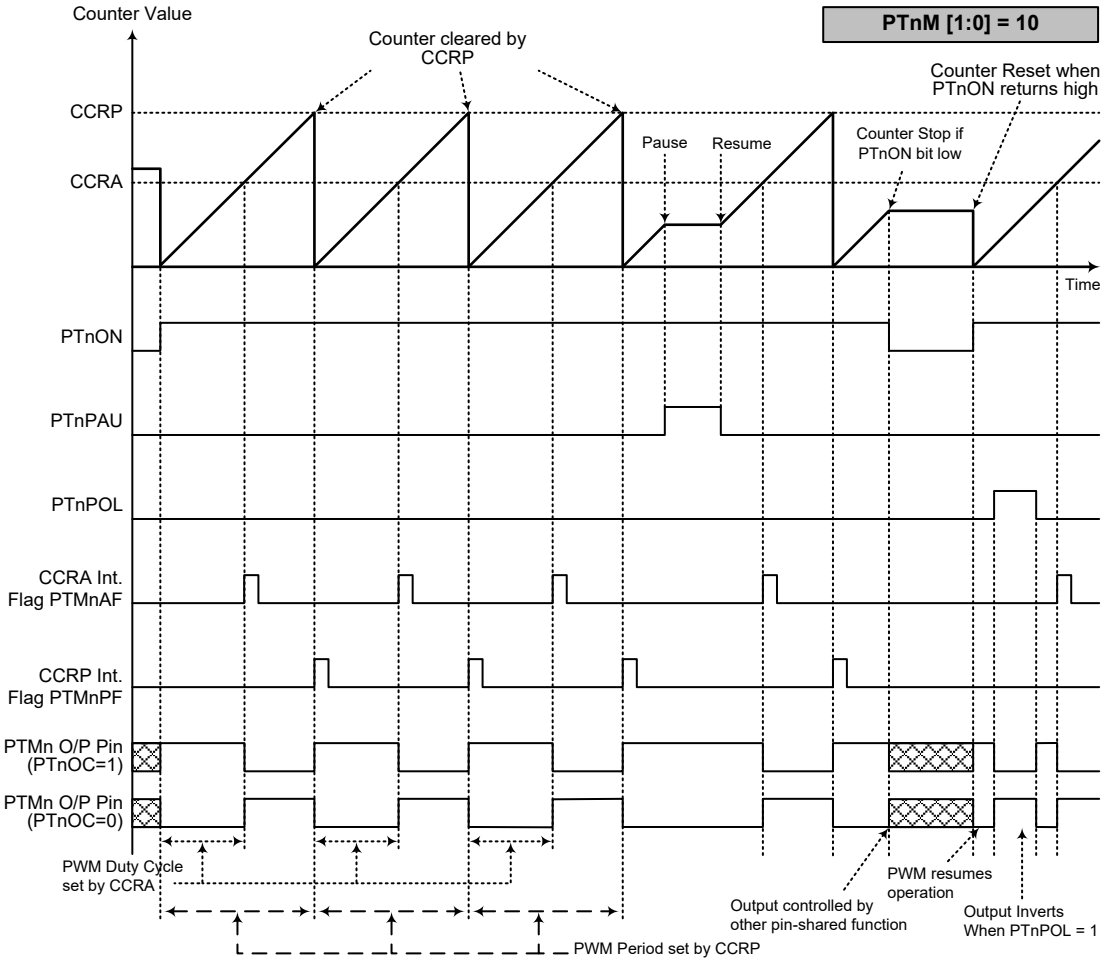
- **10-bit PTMn，PWM 输出模式，边沿对齐模式**

CCRP	1~1023	0
Period	1~1023	1024
Duty	CCRA	

若 f_{SYS}=16MHz，PTMn 时钟源选择 f_{SYS}/4，CCRP=512 且 CCRA=128，

PTMn PWM 输出频率 = (f_{SYS}/4)/512=f_{SYS}/2048=8kHz，duty=128/512=25%，

若由 CCRA 寄存器定义的 Duty 值等于或大于 Period 值，PWM 输出占空比为 100%。



PWM 输出模式 (n=0)

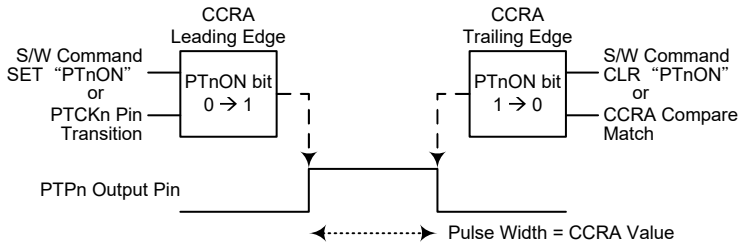
注：1. CCRP 清除计数器
2. 计数器清零并设置 PWM 周期
3. 当 PTnIO[1:0]=00 或 01，PWM 功能不变
4. PTnCCLR 位对 PWM 功能无影响

单脉冲输出模式

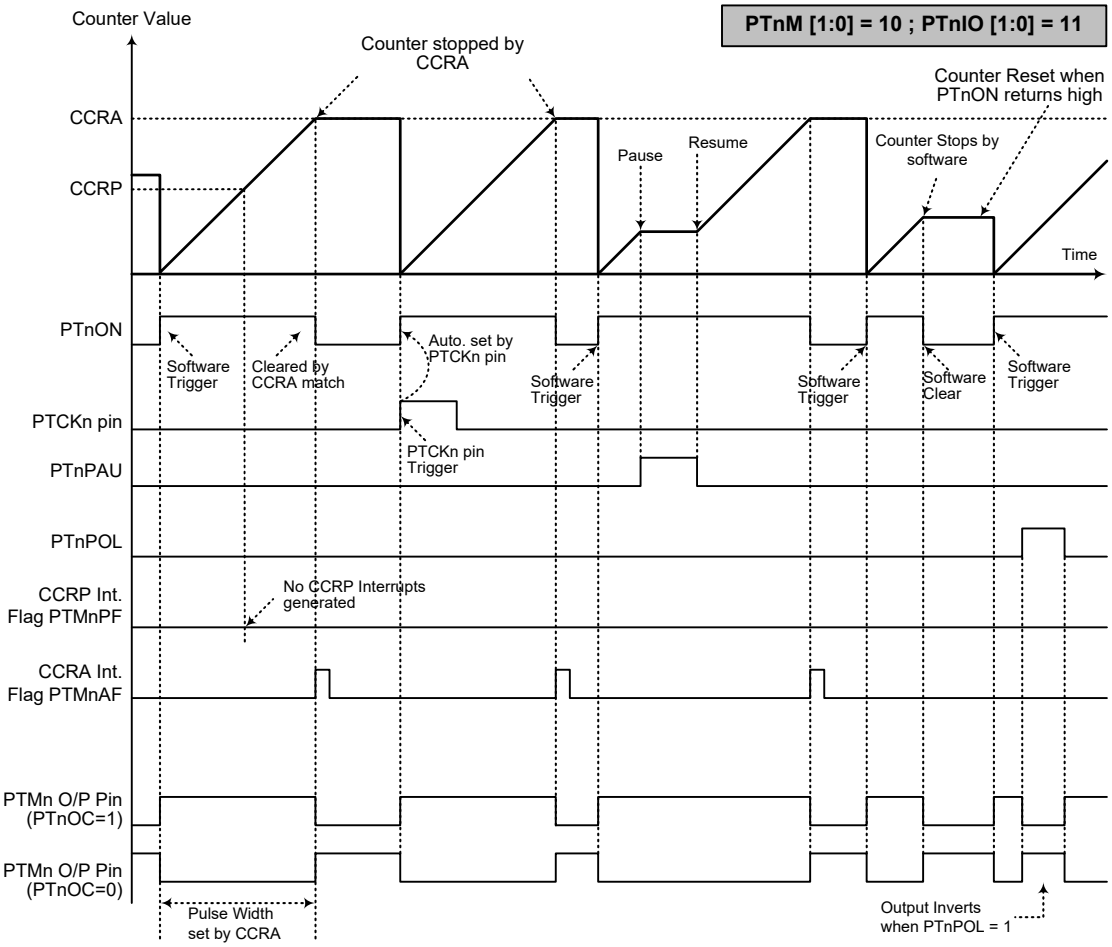
为使 PTMn 工作在此模式，PTMnC1 寄存器中的 PTnM1 和 PTnM0 位需要设置为“10”，并且相应的 PTnIO1 和 PTnIO0 需要设置为“11”。正如模式名所言，单脉冲输出模式，在 PTMn 输出脚将产生一个脉冲输出。

通过应用程序控制 PTnON 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，PTnON 位可在 PTCKn 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 PTnON 位转变为高电平时，计数器将开始运行，并产生脉冲前沿。通过应用程序使 PTnON 位清零或比较器 A 比较匹配发生时，产生脉冲后沿。

而比较器 A 比较匹配发生时，会自动清除 PTnON 位并产生单脉冲输出边沿跳转。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 PTMn 中断。PTnON 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。在单脉冲输出模式中，CCRP 寄存器和 PTnCCLR 位未使用。



单脉冲产生示意图 (n=0)



单脉冲输出模式 (n=0)

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

A/D 转换器

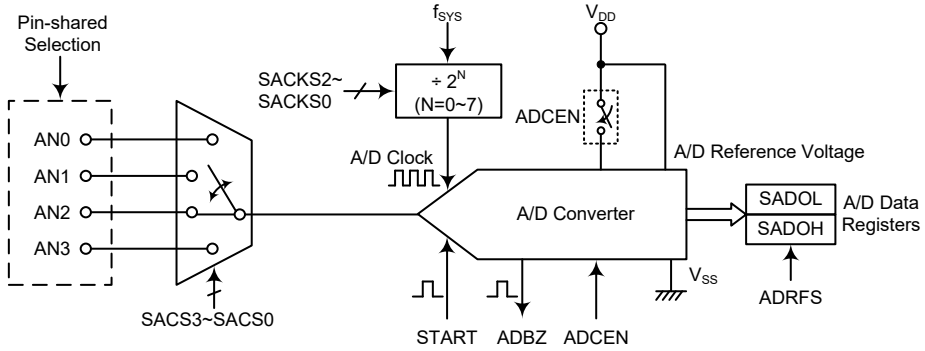
对于大多数电子系统而言，处理现实世界的模拟信号是共同的需求。为了完全由单片机来处理这些信号，首先需要通过 A/D 转换器将模拟信号转换成数字信号。将 A/D 转换器电路集成入单片机，可有效的减少外部器件，随之而来，具有降低成本和减少器件空间需求的优势。

A/D 转换器简介

此系列单片机包含一个多通道的 A/D 转换器，它可以直接接入外部模拟信号 (来自传感器或其它控制信号) 并直接将这些信号转换成 12 位的数字量。

单片机型号	外部输入通道	A/D 通道选择位
BC66F2235	1：AN0	SACS3~SACS0
BC66F2245 BC66F2255	4：AN0~AN3	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四个寄存器控制。一对只读寄存器用来存放 12 位 A/D 转换数据的值。剩下两个控制寄存器用于设置 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	D7	D6	D5	D4	D3	SACKS2	SACKS1	SACKS0

A/D 转换寄存器列表

A/D 转换器数据寄存器 – SADOL, SADOH

此 A/D 转换器每次转换值为 12 位，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。由于只使用到 16 位中的 12 位，转换结果的数据存储格式由 SADC0 寄存器的 ADRFS 位控制，如下表所示。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 转换器控制寄存器 – SADC0, SADC1

寄存器 SADC0 和 SADC1 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 转换器的忙碌状态。由于每个单片机只包含一个实际的模数转换电路，因此这些外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 SACS1~SACS0 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。

引脚共用功能选择寄存器的相关位用来定义 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	ADRFS	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
此位用于启动 A/D 转换过程。通常此位为低，但如果设为高再被清零，将启动 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 转换过程是否完成。当 START 位由低变为高再变为低时，ADBZ 位为高，表明 A/D 转换已开始。A/D 转换结束后，此位被清零。

Bit 5 **ADCEN**：A/D 转换器使能 / 除能控制位
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 的值保持不变。

Bit 4 **ADRFS**：A/D 转换数据格式选择位
0：A/D 转换数据格式 → SADOH=D[11:4]，SADOL=D[3:0]
1：A/D 转换数据格式 → 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 数据寄存器章节。

Bit 3~0 **SACS3~SACS0**：A/D 转换器输入通道选择位
BC66F2235：
0000：AN0
0001~1111：未定义，输入浮空

BC66F2245/BC66F2255：
0000：AN0
0001：AN1
0010：AN2
0011：AN3
0100~1111：未定义，输入浮空

- **SADC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	SACKS2	SACKS1	SACKS0
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 7~3 **D7~D3**：保留位，必须固定为“00000”

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：$f_{SYS}/128$
这三位用于选择 A/D 转换器的时钟源。

A/D 转换器参考电压

A/D 转换器参考电压来自 A/D 转换器电源电压 V_{DD}。模拟输入值一定不能超过参考电压值 V_{DD}。

A/D 转换器输入信号

所有的 A/D 模拟输入引脚都与 I/O 口及其它功能共用。使用 PBS0 寄存器中相应的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。

A/D 转换器操作

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 位会在下一个 ADCLK 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，检查此位是否被清零，作为另一种侦测 A/D 转换周期结束的方法。若 ADCLK 设定为较低的时间，可能会使轮询结果产生 0→1→0 的变化。

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就必须小心。例如，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不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。

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

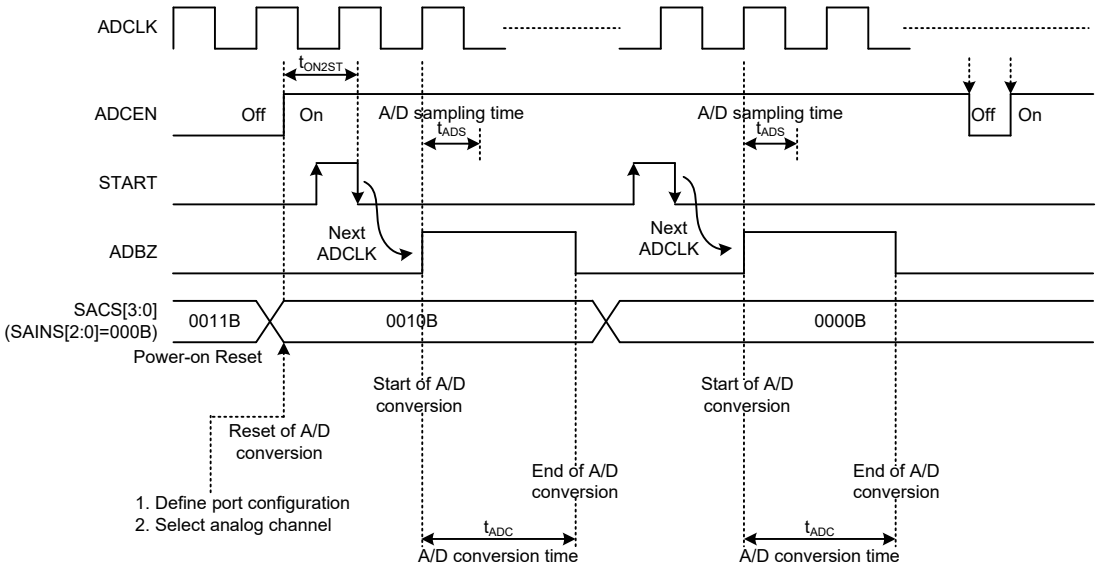
SADC0 寄存器中的 ADCEN 位用于控制 A/D 转换电路电源的开启和关闭。该位必须置高以开启 A/D 转换器电源。当设置 ADCEN 位为高开启 A/D 转换器内部电路时，在 A/D 转换成功开启前需一段延时。即使通过相关引脚共用控制位选择无引脚作为 A/D 输入，如果 ADCEN 设为“1”，那么仍然会产生功耗。因此在功耗敏感的应用中，当未使用 A/D 转换器功能时，建议设置 ADCEN 为低以减少功耗。

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。

最大 A/D 转换率 = 1/(A/D 时钟周期 × 16)

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 $16\times t_{ADCK}$，t_{ADCK} 为 A/D 时钟周期。



A/D 转换时序图

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 SADC0 寄存器中的 SACS3~SACS0 位选择连接至内部 A/D 转换器的外部通道。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。

- 步骤 4

  设置 SADC0 寄存器中的 ADRFS 位选择 A/D 转换器输出数据格式。

- 步骤 5

  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 6

  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 7

  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。

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

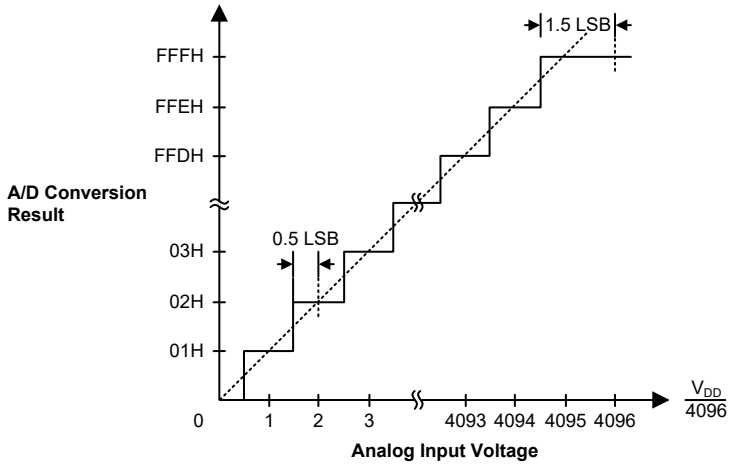
单片机含有一组 12 位的 A/D 转换器，它转换的最大值可达 FFFH。由于模拟输入最大值等于实际 A/D 转换器参考电压值，V_{DD}，因此每一位可表示 V_{DD}/4096 的模拟输入值。

$$1\ LSB=V_{DD}\div 4096$$

通过下面的等式可估算 A/D 转换器输入电压值：

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

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_{DD} 之前的 1.5 LSB 处改变。



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
mov SADC1,a
mov a,02h                  ; set PBS0 to configure pin AN0
mov PBS0,a
mov a,20h
mov SADC0,a                ; enable A/D and connect AN0 channel to A/D
                           ; converter
:
start_conversion:
```

```
clr START                   ; high pulse on start bit to initiate conversion
set START                   ; reset A/D
clr START                   ; start A/D
polling_EOC:
sz  ADBZ                    ; poll the SADC0 register ADBZ bit to detect end
                            ; of A/D conversion
jmp polling_EOC             ; continue polling
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
mov SADC1,a
mov a,02h                   ; set PBS0 to configure pin AN0
mov PBS0,a
mov a,20h
mov SADC0,a                 ; enable A/D and connect AN0 channel to A/D
                            ; converter
:
start_conversion:
clr START                   ; high pulse on START bit to initiate conversion
set START                   ; reset A/D
clr START                   ; start A/D
clr ADF                     ; clear ADC interrupt request flag
set ADE                     ; enable ADC interrupt
set EMI                     ; enable global interrupt
:
ADC_ISR:                    ; ADC interrupt service routine
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
```

通用串行接口模块 – USIM

此系列单片机内有一个通用串行接口模块，包括三种易与外部设备通信的串行接口：四线 SPI、两线 I^2C 或两线 / 单线 UART 接口。这三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、Flash 存储器或 EEPROM 存储器等硬件设备通信。因为 USIM 接口引脚是与其它 I/O 引脚共用，因此在使用 USIM 功能前，要先通过相应的引脚共用功能选择寄存器选定 USIM 引脚功能。因为这三种接口共用引脚和寄存器，所以要先通过 SIMC0 寄存器中的 UART 模式选择位 UMD 和 SPI/I^2C 工作模式控制位 SIM2~SIM0 选择哪一种通信接口。若 USIM 功能使能，可通过上拉电阻控制寄存器选择与输入 / 输出口共用的 USIM 脚的上拉电阻。

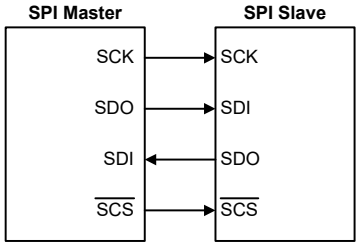
SPI 接口

SPI 接口常用于与外部设备如传感器、Flash 存储器或 EEPROM 存储器等通信。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 $\overline{SCS}$。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 $\overline{SCS}$。SDI 和 SDO 是数据的输入和输出线。SCK 是串行时钟线，$\overline{SCS}$ 是从机的选择线。SPI 的接口引脚与普通 I/O 口和 I^2C/UART 的功能脚共用。通过设定相关引脚共用选择位和 SIMC0/SIMC2 寄存器的对应位，来使能 SPI 接口。连接到 SPI 接口的单片机以从主 / 从模式进行通信，且所有的数据传输由主机发起，时钟信号也由主机控制。由于单片机只有一个 $\overline{SCS}$ 引脚，所以只能拥有一个从机设备。可通过软件控制 $\overline{SCS}$ 引脚使能与除能，设置 CSEN 位为“1”使能 $\overline{SCS}$ 功能，设置 CSEN 位为“0”，$\overline{SCS}$ 引脚将处于浮空状态。

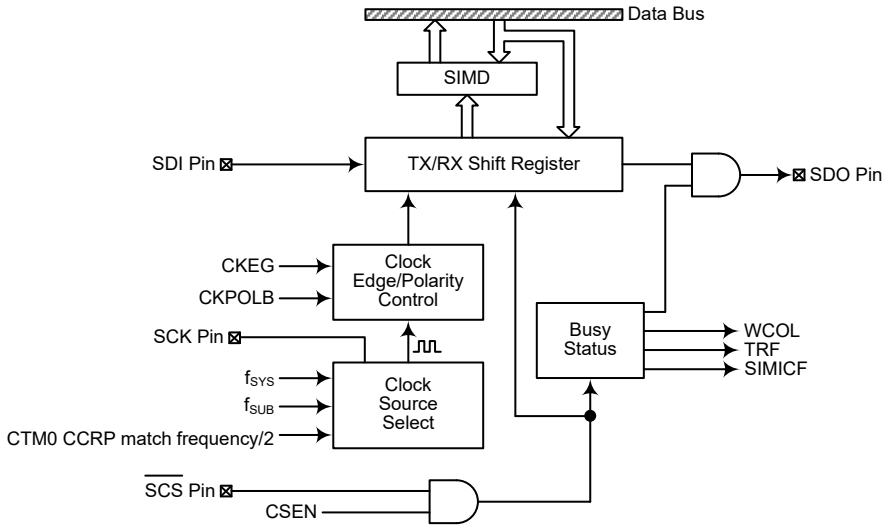


SPI 主 / 从机连接方式

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以及 CSEN、SIMEN 位的状态。



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，只有在合理设置 SIMC0 寄存器中的 UMD 位和 SIM2~SIM0 位选择 SPI 模式后，SIMC2 和 SIMD 寄存器以及它们的上电复位值才有效。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	UMD	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

SPI 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 I^2C 功能所共用。在单片机将数据写入到 SPI 总线之前，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**：USIM SPI/I^2C 数据寄存器位 bit 7 ~ bit 0

SPI 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。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	UMD	SIMDEB1	SIMDEB0	SIMEN	SIMICF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	0	0	0	0	0

Bit 7~5 **SIM2~SIM0**：USIM SPI/I²C 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 CTM0 CCRP 匹配频率 / 2
101：SPI 从机模式
110：I²C 从机模式
111：未使用模式
当 UMD 位清零时，这几位用于设置 USIM SPI/I²C 功能的工作模式，除了选择 USIM 模块的 I²C 或 SPI 功能，还可选择 SPI 的主从模式和 SPI 的主机时钟频率。SPI 时钟源可来自于系统时钟和 f_{SUB} 也可以选择来自 CTM0。若选择的是作为 SPI 从机，则其时钟源从外部主机而得。

Bit 4 **UMD**：UART 模式选择位
0：SPI 或 I²C 模式
1：UART 模式
此位为 UART 模式选择位。当此位清零时，选择 SPI 或 I²C 模式，而实际 SPI 或 I²C 模式是通过 SIM2~SIM0 位选择。

Bit 3~2 **SIMDEB1~SIMDEB0**：I²C 去抖时间选择位
这些位只有在 USIM 设置成 I²C 接口模式时才有效。请参考 I²C 寄存器部分。

Bit 1 **SIMEN**：USIM SPI/I²C 控制位
0：除能
1：使能
此位为 USIM SPI/I²C 接口的开 / 关控制位。此位为“0”时，USIM SPI/I²C 接口除能，SDI、SDO、SCK 和 $\overline{SCS}$ 或 SDA 和 SCL 脚将失去 SPI 或 I²C 功能，USIM 工作电流减小到最小值。此位为“1”时，USIM SPI/I²C 接口使能。若 USIM 经由 UMD 位和 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 USIM 经由 UMD 位和 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**：USIM SPI 未完成标志位
0：未发生
1：发生
此位仅当 USIM 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 **D7~D6**：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 引脚为高电平
1：当时钟无效时，SCK 引脚为低电平
此位决定了时钟线的基础状态，若此位为高，当时钟无效时 SCK 为低电平，若此位为低，当时钟无效时 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。这两位必须在执行数据传输前先被设置好，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 数据移位命令位
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，$\overline{SCS}$ 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，表示在传输过程中有数据被写入 SIMD 寄存器。若数据正在被传输时，此写操作无效。此位可被应用程序清零。

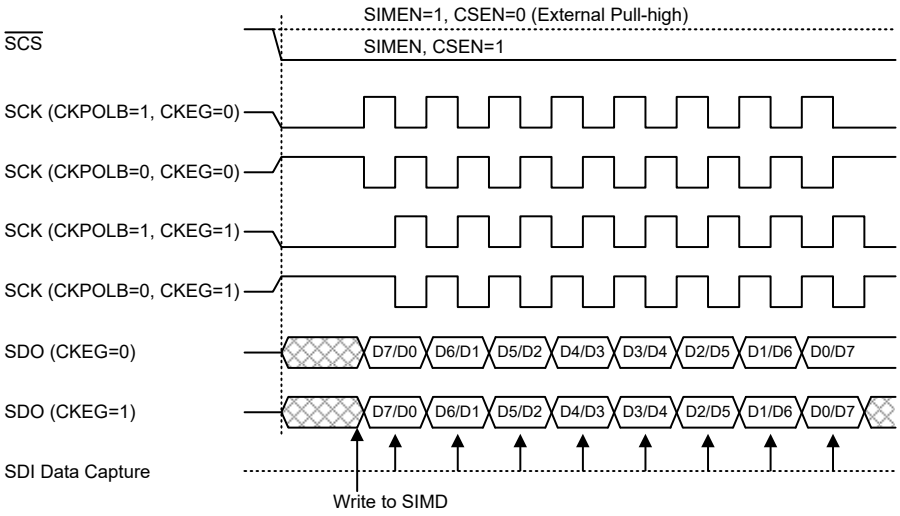
Bit 0 **TRF**：SPI 发送 / 接收结束标志位
0：数据正在发送
1：数据发送结束
TRF 位为发送 / 接收结束标志位，当 SPI 数据传输结束时，此位自动置为高，但须通过应用程序设置为“0”。此位也可用于产生中断。

SPI 通信

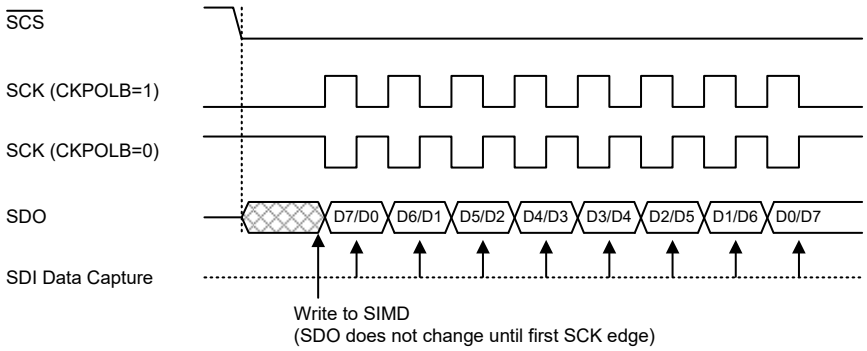
将 SIMEN 设置为高，使能 SPI 功能之后，若单片机处于主机模式，当数据写入到寄存器 SIMD 的同时传输 / 接收开始进行。数据传输完成时，TRF 位将自动被置位但清除只能通过应用程序完成。若单片机处于从机模式，收到主机发来的信号之后，会传输 SIMD 中的数据，而且在 SDI 引脚上的数据也会被移位

到 SIMD 寄存器中。主机应在输出时钟信号之前先输出一个 $\overline{SCS}$ 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。所附时序图表明了在 CKPOLB 和 CKEG 位各种设置情况下从机数据与 SCK 信号的关系。

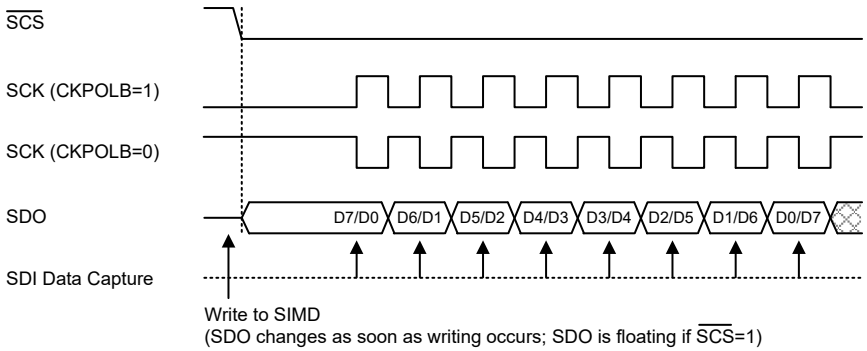
SPI 主机模式在 SPI 时钟运行情况下可以进行正常传输。



SPI 主机模式时序

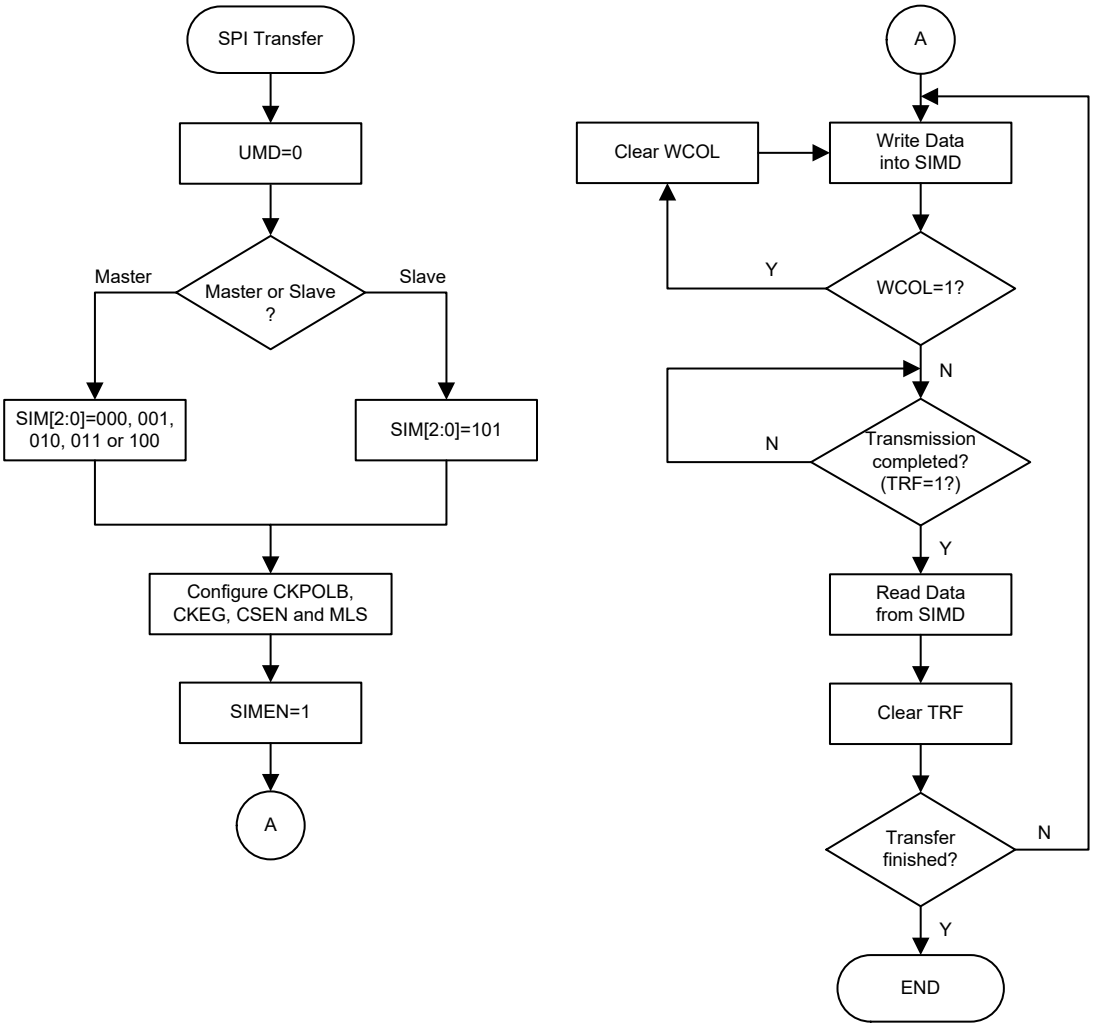


SPI 从机模式时序 – CKEG=0



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 CSEN=1、$\overline{SCS}$=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，通过设置相应引脚共用控制位，SCK、SDI、SDO、$\overline{SCS}$ 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，$\overline{SCS}$ 信号线有效将使能 SPI 接口。设置此位为低，SPI 接口将除能，$\overline{SCS}$ 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，$\overline{SCS}$、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，主机发起数据传输，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 UMD 和 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 SDO 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 USIM 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 UMD 和 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。

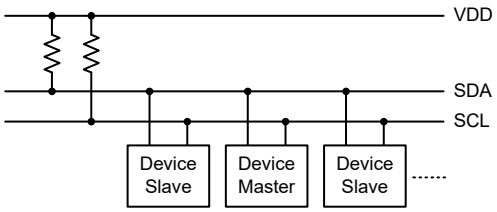
- 步骤 5
  检测 WCOL 位，若此位为高，则发生数据冲突并跳回至步骤 4；若为低，则继续执行下面的步骤。
- 步骤 6
  检测 TRF 位或等待 USIM SPI 串行总线中断发生。
- 步骤 7
  从 SIMD 寄存器中读数据。
- 步骤 8
  清除 TRF。
- 步骤 9
  跳回至步骤 4。

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，使之在很多的应用场合中大受欢迎。

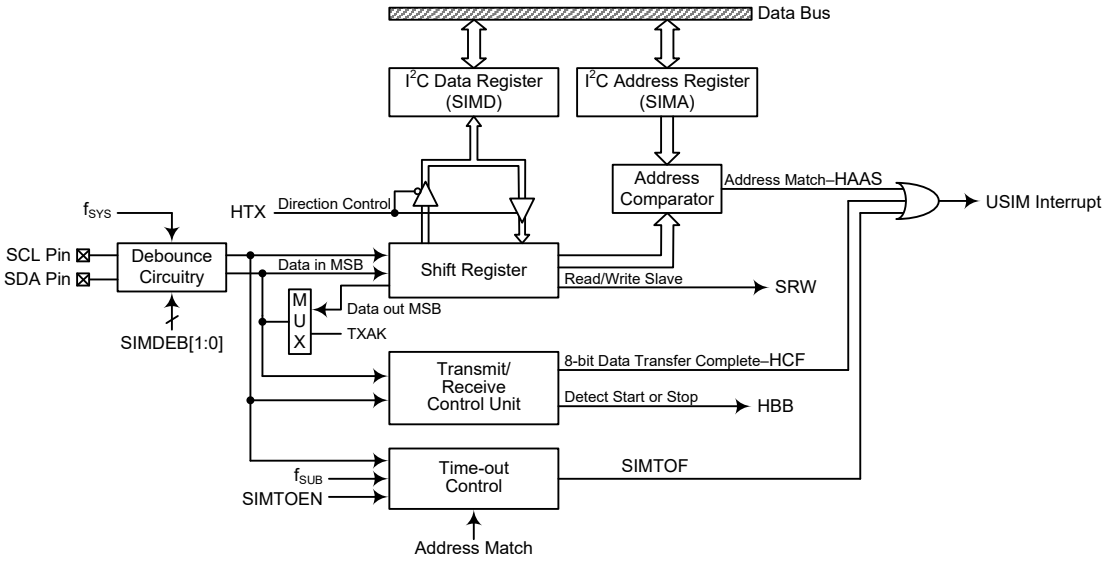


I²C 主从总线连接图

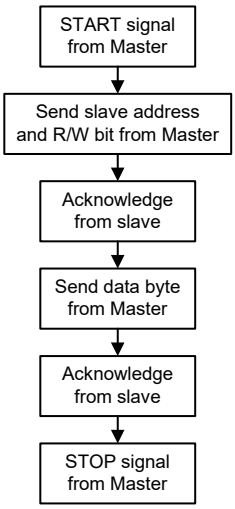
I²C 接口操作

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。因此应在这些输出口上都应加上拉电阻。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传输和接收数据，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。即使 I²C 设备被激活，与 SCL/SDA 引脚共用的 I/O 口上拉电阻控制功能仍有效，其上拉电阻功能由相应的上拉电阻控制寄存器控制。



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，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。

I²C 去抖时间选择	I²C 标准模式 (100kHz)	I²C 快速模式 (400kHz)
无去抖时间	f_{SYS} > 2MHz	f_{SYS} > 4MHz
2 个系统时钟去抖时间	f_{SYS} > 4MHz	f_{SYS} > 8MHz
4 个系统时钟去抖时间	f_{SYS} > 4MHz	f_{SYS} > 8MHz

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，一个地址寄存器 SIMA 以及一个数据寄存器 SIMD。注意，只有在合理设置 SIMC0 寄存器中的 UMD 位和 SIM2~SIM0 位选择 I²C 模式后，SIMC1、SIMD、SIMA 和 SIMTOC 寄存器以及它们的上电复位值才有效。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	UMD	SIMDEB1	SIMDEB0	SIMEN	SIMICF
SIMC1	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
SIMA	SIMA6	SIMA5	SIMA4	SIMA3	SIMA2	SIMA1	SIMA0	D0
SIMD	D7	D6	D5	D4	D3	D2	D1	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**：USIM SPI/I²C 数据寄存器位 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。

- **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 包括多个用于指示 I²C 传输状态的相关标志位。SIMTOC 寄存器用于控制 I²C 超时功能，此寄存器在 I²C 超时一节介绍。

- **SIMC0 寄存器**

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

Bit 7~5 **SIM2~SIM0**：USIM SPI/I²C 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 CTM0 CCRP 匹配频率 / 2
101：SPI 从机模式
110：I²C 从机模式
111：未使用模式
当 UMD 位清零时，这几位用于设置 USIM SPI/I²C 功能的工作模式，除了选择 USIM 模块的 I²C 或 SPI 功能，还可选择 SPI 的主从模式和 SPI 的主机时钟频率。SPI 时钟源可来自于系统时钟和 f_{SUB} 也可以选择来自 CTM0。若选择的是作为 SPI 从机，则其时钟源从外部主机而得。

Bit 4 **UMD**：UART 模式选择位
0：SPI 或 I²C 模式
1：UART 模式
此位为 UART 模式选择位。当此位清零时，选择 SPI 或 I²C 模式，而实际 SPI 或 I²C 模式是通过 SIM2~SIM0 位选择。

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 UMD 位为“0”、SIM2~SIM0 位为“110”将 USIM 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**：USIM SPI/I²C 控制位
0：除能
1：使能
此位为 USIM SPI/I²C 接口的开 / 关控制位。此位为“0”时，USIM SPI/I²C 接口除能，SDI、SDO、SCK 和 $\overline{SCS}$ 或 SDA 和 SCL 脚将失去 SPI 或 I²C 功能，USIM 工作电流减小到最小值。此位为“1”时，USIM SPI/I²C 接口使能。若 USIM 经由 UMD 位和 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 USIM 经由 UMD 位和 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**：USIM SPI 未完成标志位
此位仅当 USIM 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	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 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的地址相同。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，此位变为高电平。当检测到 STOP 信号时 I²C 总线空闲，该位变为低电平。

Bit 4 **HTX**：从机处于发送或接收模式标志位
0：从机处于接收模式
1：从机处于发送模式

Bit 3 **TXAK**：I²C 总线发送应答标志位
0：从机发送应答标志
1：从机没有发送应答标志
从机接收完 8 位数据之后，该位将在第九个从机时钟时被传到总线上。如果从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 / 写位
0：从机应处于接收模式
1：从机应处于发送模式
SRW 位是从机读写位。决定主机是否希望传输数据或接收来自 I²C 总线的数据。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，此时从机处于传输模式。当 SRW 位为“0”时，主机往总线上写数据，从机处于接收模式以读取数据。

Bit 1 **IAMWU**：I²C 地址匹配唤醒控制位
0：除能
1：使能
此位设置为“1”则使能 I²C 地址匹配使系统从休眠或空闲模式中唤醒的功能。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，在系统唤醒后须软件清除此位以确保单片机正确地运行。

Bit 0 **RXAK**：I²C 总线接收应答标志位
0：从机接收到应答标志
1：从机没有接收到应答标志
RXAK 位是接收应答标志位。如果 RXAK 位为“0”，即表示 8 位数据传输之后，从机在第九个时钟有接受到一个应答信号。如果从机处于发送状态，从机作为发送方会检查 RXAK 位来判断主机接收方是否愿意继续接收下一个字节。因此发送方会一直发送数据，直到 RXAK 为“1”时才停止发送数据。这时，发送方将释放 SDA 线，主机方可发出停止信号从而释放 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 USIM 中断。进入中断服务程序后，系统要检测 HAAS 位和 SIMTOF 位，以判断中断源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 超时。在数据传递中，要注意的是，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 位以确定自己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

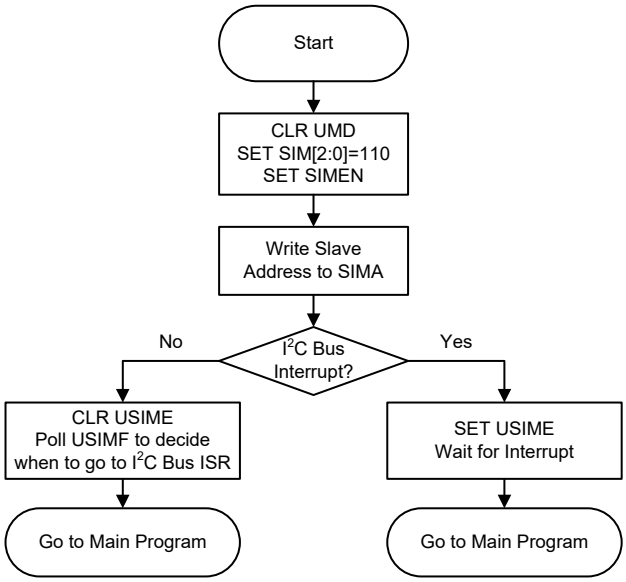
  设置 SIMC0 寄存器中 UMD 位为“0”、SIM2~SIM0 位为“110”和 SIMEN 位为“1”，以使能 I²C 总线。

- 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 USIME 位以使能 USIM 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

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 USIM I^2C 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，从机随后发出一个低电平应答信号 (即第 9 位)。当从机地址匹配时，从机会将状态标志位 HAAS 置位。

USIM I^2C 总线中断有三个中断源，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 USIM I^2C 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 I^2C 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。

I^2C 总线读 / 写信号

SIMC1 寄存器的 SRW 位用来表示主机是要从 I^2C 总线上读取数据还是要将数据写到 I^2C 总线上。从机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 I^2C 总线上读取数据，从机则作为发送方，将数据写到 I^2C 总线；当 SRW 清“0”，表示主机要写数据到 I^2C 总线上，从机则做为接收方，从 I^2C 总线上读取数据。

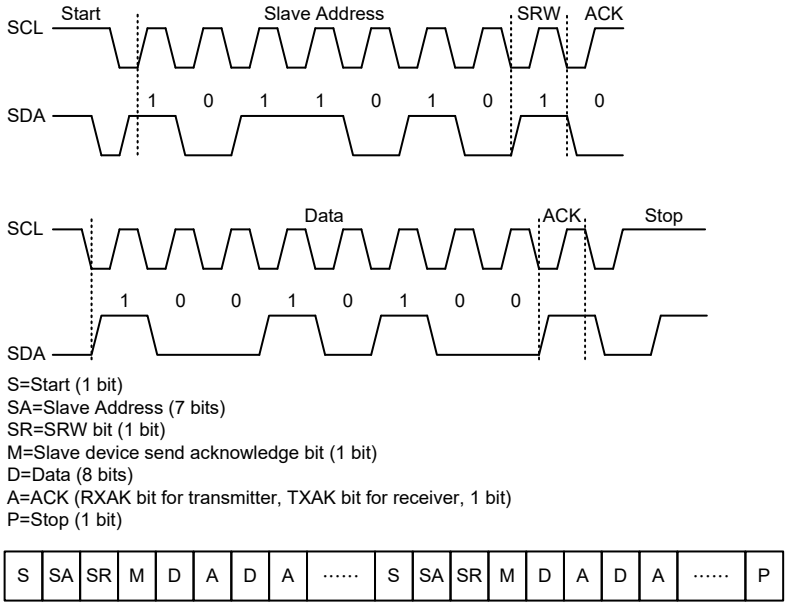
I^2C 总线从机地址应答信号

主机发送呼叫地址后，当 I^2C 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。

I^2C 总线数据和应答信号

在从机确认接收到从地址后，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。如果从机发送方没接收到来自主机接收方的应答信号，发送方将释放 SDA 线，此时主机方可发出 STOP 信号以释放 I^2C 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，如果从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。

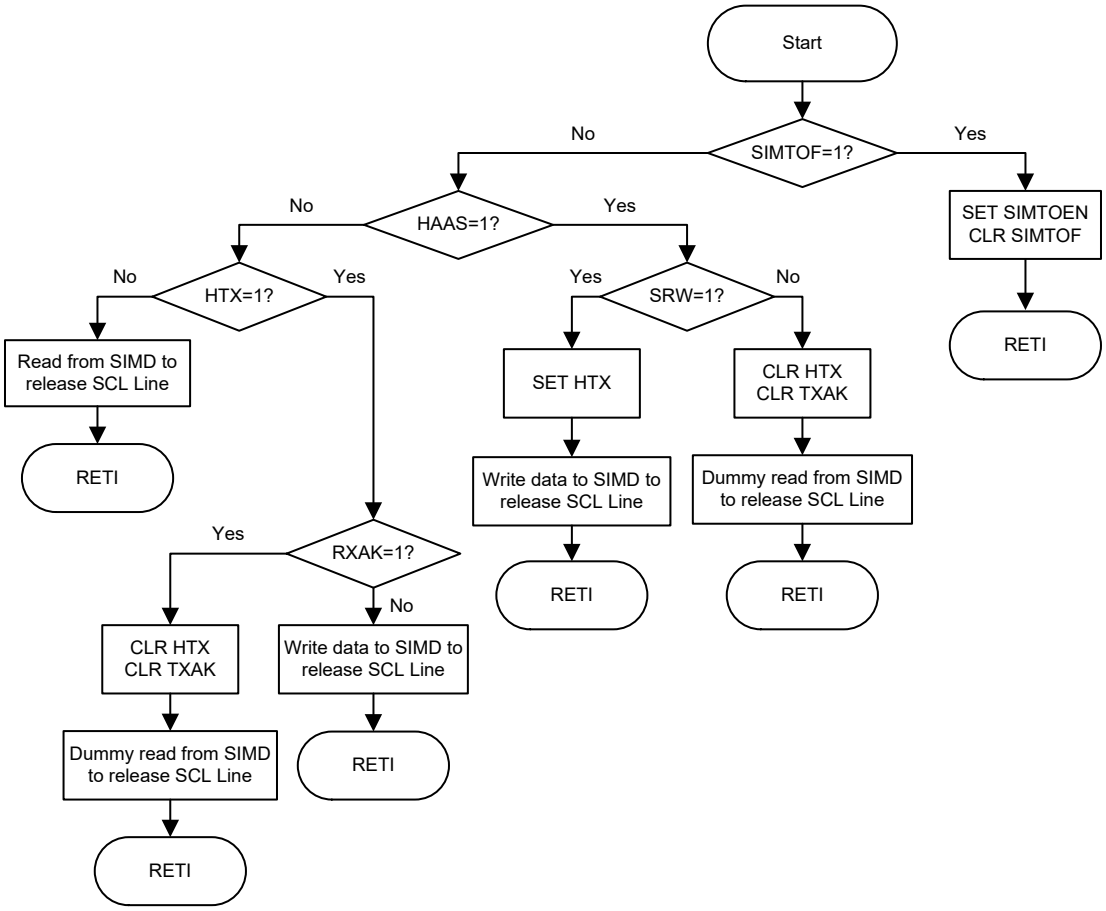


S=Start (1 bit)
SA=Slave Address (7 bits)
SR=SRW bit (1 bit)
M=Slave device send acknowledge bit (1 bit)
D=Data (8 bits)
A=ACK (RXAK bit for transmitter, TXAK bit for receiver, 1 bit)
P=Stop (1 bit)

S	SA	SR	M	D	A	D	A	·····	S	SA	SR	M	D	A	D	A	·····	P

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。

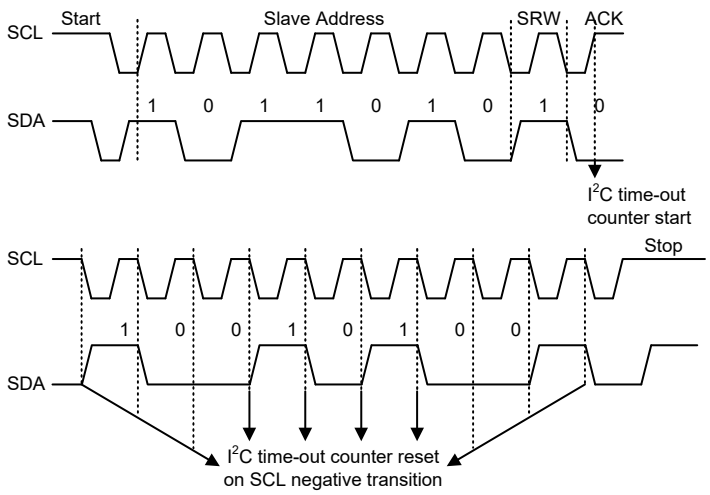
I^2C 通信时序图



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 USIM 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	I²C 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[5:0] 位进行选择。超时周期可通过公式计算：(1~64) × (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**：USIM I²C 超时控制位
0：除能
1：使能

Bit 6 **SIMTOF**：USIM I²C 超时标志位
0：超时未发生
1：超时发生
当发生超时，此位由硬件自动置位；此位必须通过应用程序清零。

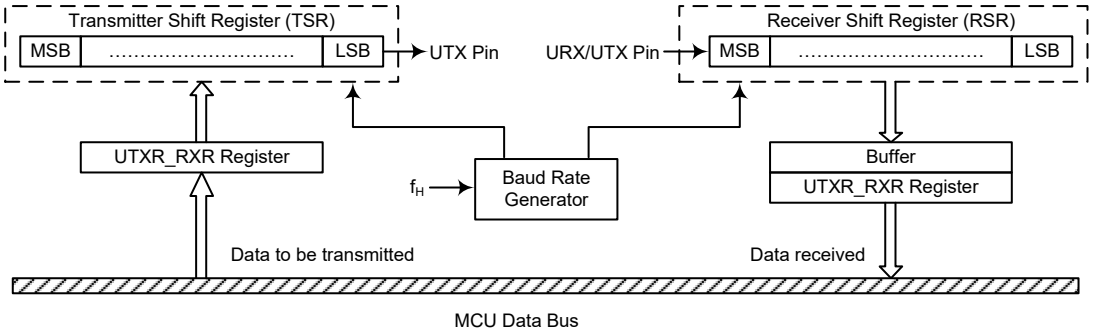
Bit 5~0 **SIMTOS5~SIMTOS0**：USIM I²C 超时时间选择位
I²C 超时时钟源是 f_{SUB}/32。
I²C 超时时间计算方法：(SIMTOS[5:0]+1) × (32/f_{SUB})。

UART 接口

该系列单片机具有一个全双工或半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 功能与 SPI 和 I^2C 接口共用一个内部中断向量，当接收到数据或数据发送结束，触发中断。

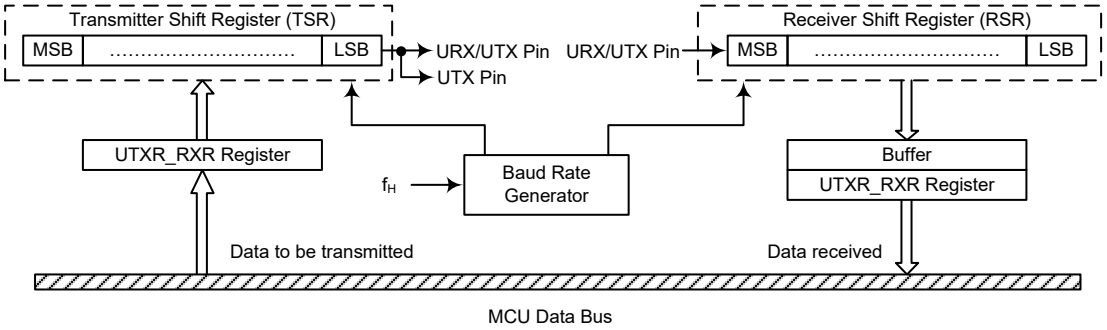
内置的 UART 功能包含以下特性：

- 全双工或半双工 (单线通信模式) 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
- URX/UTX 引脚唤醒功能
- 发送和接收中断
- 中断可由下列条件触发：
  - ♦ 发送器为空
  - ♦ 发送器空闲
  - ♦ 接收完成
  - ♦ 接收器溢出
  - ♦ 地址匹配



注：对于 BC66F2235，USWM=0 时 UART 功能不可用，只有在单线模式使能时才能正常运行。

UART 数据传输方框图 – USWM=0



UART 数据传输方框图 – USWM=1

UART 外部引脚

内部 UART 有两个外部引脚 UTX 和 URX/UTX，可与外部串行接口进行通信。UTX 和 URX/UTX 与 I/O 口或其它功能共用引脚。在使用 UART 功能前，应先通过相应的引脚共用功能选择寄存器，选择 UTX 和 URX/UTX 引脚功能。当 UMD、UREN、UTXEN 和 URXEN 位置高时，将自动设置这些 I/O 脚或其它共用功能脚作为发送输出和接收输入。此时，用作发送输出的引脚其内部上拉电阻会被除能，而用作接收输入的引脚其内部上拉电阻由相应的上拉电阻控制位控制。当 UMD、UREN、UTXEN 或 URXEN 位清零除能 UTX 或 URX/UTX 引脚功能后，UTX 或 URX/UTX 引脚将处于浮空状态。这时 UTX 或 URX/UTX 引脚是否连接内部上拉电阻是由相应的 I/O 上拉电阻控制位决定的。

UART 单线模式

UART 功能支持单线模式通信，通过 UUCR3 寄存器中的 USWM 位选择。当设置该位为高，UART 将工作在单线模式。在单线模式下，单个 URX/UTX 引脚通过相关控制位的不同设置即可完成数据的发送与接收。设置 URXEN 位为高，URX/UTX 引脚用作接收引脚。将 URXEN 位清零，同时设置 UTXEN 位为高，URX/UTX 引脚用作发送引脚。

在单线模式下建议不要将 URXEN 位和 UTXEN 位同时设置为高。若 URXEN 位和 UTXEN 位同时为高，URXEN 位具有更高的优先级，此时 UART 为接收器状态。

需特别注意的是，UART 章节所有内容是基于 UART 全双工通信来对 UART 功能进行描述，相关的说明除引脚的使用外，对半双工通信 (单线模式) 同样适用。在理解单线模式通信时，全双工通信中使用的 UTX 引脚需取代为 URX/UTX 引脚。

在单线模式下，通过合理的软件配置，数据也可以在 UTX 引脚发送。因此数据可通过 URX/UTX 和 UTX 引脚输出。

UART 数据传输方案

UART 数据传输方框图方框图显示了 UART 的整体结构。需要发送的数据首先写入 UTXR_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 UTX 引脚上，低位在前。UTXR_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 URX/UTX 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 UTXR_RXR 寄存器中。UTXR_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 UTXR_RXR 寄存器。

UART 状态和控制寄存器

与 UART 功能相关的有七个寄存器，SIMC0 寄存器中的 UMD 位用于选择 UART 接口功能。UUCR3 寄存器中的 USWM 位用于使能/除能 UART 单线模式。其它包括控制 UART 模块整体功能的 UUSR、UUCR1 和 UUCR2 寄存器，控制波特率的 UBRG 寄存器，管理发送和接收数据的数据寄存器 UTXR_RXR。注意，只有在 SIMC0 寄存器中的 UMD 位设置为“1”后，UART 相关的寄存器以及它们的上电复位值才有效。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM2	SIM0	UMD	SIMDEB1	SIMDEB0	SIMEN	SIMICF
UUSR	UPERR	UNF	UFERR	UOERR	URIDLE	URXIF	UTIDLE	UTXIF
UUCR1	UREN	UBNO	UPREN	UPRT	USTOPS	UTXBRK	URX8	UTX8
UUCR2	UTXEN	URXEN	UBRGH	UADDEN	UWAKE	URIE	UTIIE	UTEIE
UUCR3	—	—	—	—	—	—	—	USWM
UTXR_RXR	UTXRX7	UTXRX6	UTXRX5	UTXRX4	UTXRX3	UTXRX2	UTXRX1	UTXRX0
UBRG	UBRG7	UBRG6	UBRG5	UBRG4	UBRG3	UBRG2	UBRG1	UBRG0

UART 寄存器列表

- **SIMC0 寄存器**

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

Bit 7~5 **SIM2~SIM0**：USIM SPI/I²C 工作模式控制位
当 UMD 位清零时，这几位用于设置 USIM SPI/I²C 功能的工作模式。更多细节详见 SPI 或 I²C 寄存器章节。

Bit 4 **UMD**：UART 模式选择位
0：SPI 或 I²C 模式
1：UART 模式
此位为 UART 模式选择位。当此位清零时，选择 SPI 或 I²C 模式，而实际 SPI 或 I²C 模式是通过 SIM2~SIM0 位选择。当选择 SPI 或 I²C 模式时，此位必须清零。

Bit 3~2 **SIMDEB1~SIMDEB0**：I²C 去抖时间选择位
详见 I²C 寄存器章节。

Bit 1 **SIMEN**：USIM SPI/I²C 控制位
0：除能
1：使能
此位仅当 UMD 位设置为“0”选择 SPI 或 I²C 模式时才有效。详见 SPI 或 I²C 寄存器章节。

Bit 0 **SIMICF**：USIM SPI 未完成标志位
详见 SPI 寄存器章节。

- **UUSR 寄存器**

寄存器 UUSR 是 UART 的状态寄存器，可以通过程序读取以得知当前 UART 状态。所有 UUSR 位是只读的。详细解释如下：

Bit	7	6	5	4	3	2	1	0
Name	UPERR	UNF	UFERR	UOERR	URIDLE	URXIF	UTIDLE	UTXIF
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	1	0	1	1

Bit 7 **UPERR**：奇偶校验出错标志位
0：奇偶校验正确
1：奇偶校验出错
UPERR 是奇偶校验出错标志位。若 UPERR=0，奇偶校验正确；若 UPERR=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 UUSR 寄存器再读 UTXR_RXR 寄存器来清除此位。

Bit 6 **UNF**：噪声干扰标志位
0：没有受到噪声干扰
1：受到噪声干扰
UNF 是噪声干扰标志位。若 UNF=0，没有受到噪声干扰；若 UNF=1，UART 接收数据时受到噪声干扰。它与 URXIF 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 UUSR 寄存器再读 UTXR_RXR 寄存器将清除此标志位。

Bit 5 **UFERR**：帧错误标志位
0：无帧错误发生
1：有帧错误发生
UFERR 是帧错误标志位。若 UFERR=0，没有帧错误发生；若 UFERR=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 UUSR 寄存器再读 UTXR_RXR 寄存器来清除此位。

Bit 4 **UOERR**：溢出错误标志位
0：无溢出错误发生
1：有溢出错误发生
UOERR 是溢出错误标志位，表示接收缓冲器是否溢出。若 UOERR=0，没有溢出错误；若 UOERR=1，发生了溢出错误，它将禁止下一组数据的接收。可通过软件清除该标志位，即先读取 UUSR 寄存器再读 UTXR_RXR 寄存器将清除此标志位。

Bit 3 **URIDLE**：接收状态标志位
0：正在接收数据
1：接收器空闲
URIDLE 是接收状态标志位。若 URIDLE=0，正在接收数据；若 URIDLE=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，URIDLE 被置位，表明 UART 空闲，URX/UTX 引脚处于逻辑高状态。

Bit 2 **URXIF**：接收寄存器状态标志位
0：UTXR_RXR 寄存器为空
1：UTXR_RXR 寄存器含有有效数据
URXIF 是接收寄存器状态标志位。当 URXIF=0，UTXR_RXR 寄存器为空；当 URXIF=1，UTXR_RXR 寄存器接收到新数据。当数据从移位寄存器加载到 UTXR_RXR 寄存器中，如果 UUCR2 寄存器中的 URIE=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 UNF、UFERR 或 UPERR 会在同一周期内置位。读取 UUSR 寄存器再读 UTXR_RXR 寄存器，如果 UTXR_RXR 寄存器中没有新的数据，那么将清除 URXIF 标志。

Bit 1 **UTIDLE**：数据发送完成标志位
0：数据传输中
1：无数据传输
UTIDLE 是数据发送完成标志位。若 UTIDLE=0，数据传输中。当 UTXIF=1

且数据发送完毕或者暂停字被发送时，UTIDLE 置位。UTIDLE=1，UTX 引脚空闲且处于逻辑高状态。读取 UUSR 寄存器再写 UTXR_RXR 寄存器将清除 UTIDLE 位。数据字符或暂停字就绪时，不会产生该标志位。

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

- **UUCR1 寄存器**

UUCR1、UUCR2 和 UUCR3 是 UART 的三个控制寄存器，用来定义各种 UART 功能，例如 UART 的使能与除能、奇偶校验控制、传输数据的长度以及单线模式通信等等。详细解释如下：

Bit	7	6	5	4	3	2	1	0
Name	UREN	UBNO	UPREN	UPRT	USTOPS	UTXBRK	URX8	UTX8
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 **UREN**：UART 功能使能位
0：UART 除能，UTX 和 URX/UTX 引脚处于浮空状态
1：UART 使能，UTX 和 URX/UTX 引脚作为 UART 功能引脚
此位为 UART 的使能位。UREN=0，UART 除能，URX/UTX 和 UTX 处于浮空状态；UREN=1，若 UMD 位置高，UART 使能，UTX 和 URX/UTX 将分别由 USWM 模式选择位、UTXEN 和 URXEN 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，UTXEN、URXEN、UTXBRK、URXIF、UOERR、UFERR、UPERR 和 UNF 清零，而 UTIDLE、UTXIF 和 URIDLE 置位，UUCR1、UUCR2、UUCR3 和 UBRG 寄存器中的其它位保持不变。若 UART 工作时 UREN 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 **UBNO**：发送数据位数选择位
0：8-bit 传输数据
1：9-bit 传输数据
UBNO 是发送数据位数选择位。UBNO=1，传输数据为 9 位；UBNO=0，传输数据为 8 位。若选择了 9 位数据传输格式，URX8 和 UTX8 将分别存储接收和发送数据的第 9 位。

Bit 5 **UPREN**：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。UPREN=1，使能奇偶校验；UPREN=0，除能奇偶校验。

Bit 4 **UPRT**：奇偶校验选择位
0：偶校验
1：奇校验
奇偶校验选择位。UPRT=1，奇校验；UPRT=0，偶校验。

Bit 3 **USTOPS**：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。USTOP=1，有两位停止位；USTOP=0，只有一位停止位。

Bit 2 **UTXBRK**：暂停字发送控制位
0：没有暂停字要发送
1：发送暂停字
UTXBRK 是暂停字发送控制位。UTXBRK=0，没有暂停字要发送，UTX 引脚正常操作；UTXBRK=1，将会发送暂停字，发送器将发送逻辑“0”。若 UTXBRK 为高，缓冲器中数据发送完毕后，发送器输出将至少保持 13 位宽的低电平直至 UTXBRK 复位。

Bit 1 **URX8**：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。UBNO 是用来控制传输位数是 8 位还是 9 位。

Bit 0 **UTX8**：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。UBNO 是用来控制传输位数是 8 位还是 9 位。

- **UUCR2 寄存器**

UUCR2 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 USIM UART 模式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	UTXEN	URXEN	UBRGH	UADDEN	UWAKE	URIE	UTIIE	UTEIE
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 **UTXEN**：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。UTXEN=0，发送将被除能，发送器立刻停止工作。另外发送缓冲器将被复位，此时 UTX 引脚将处于浮空状态。若 UTXEN=1 且 UMD=1 及 UREN=1，则发送将被使能，UTX 引脚将由 UART 来控制。在数据传输时清除 UTXEN 将中止数据发送且复位发送器，此时 UTX 引脚将处于浮空状态。

Bit 6 **URXEN**：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。URXEN=0，接收将被除能，接收器立刻停止工作。另外接收缓冲器将被复位，此时 URX/UTX 引脚将处于浮空状态。若 URXEN=1 且 UMD=1 及 UREN=1，则接收将被使能，URX/UTX 引脚将由 UART 来控制。在数据传输时清除 URXEN 将中止数据接收且复位接收器，此时 URX/UTX 引脚将处于浮空状态。

Bit 5 **UBRGH**：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 UBRG 寄存器一起控制 UART 的波特率。UBRGH=1，为高速模式；UBRGH=0，为低速模式。

Bit 4 **UADDEN**：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和除能位。UADDEN=1，地址检测使能，此时数据的第 8 位 (UBNO=0) 或第 9 位 (UBNO=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，若地址检测功能使能且最高位为 0，那么将不会产生中断且收到的数据也会被忽略。

Bit 3 **UWAKE**：URX/UTX 引脚下降沿唤醒 UART 功能使能位
0：URX/UTX 引脚下降沿唤醒 UART 功能除能
1：URX/UTX 引脚下降沿唤醒 UART 功能使能
此位用于控制 URX/UTX 引脚下降沿时是否唤醒 UART 功能。此位仅当 UART 时钟源 f_H 关闭时有效。若 UART 时钟源 f_H 还开启，则 URX/UTX 引脚唤醒 UART 功能无效。若此位置高且 UART 时钟 f_H 关闭，当 URX/UTX 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 URX/UTX 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 URX/UTX 引脚发生下降沿也无法恢复 UART 功能。

Bit 2 **URIE**：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 URIE=1，当 UOERR 或 URXIF 置位时，USIM 的中断请求标志 USIMF 置位；若 URIE=0，USIM 中断请求标志 USIMF 不受 UOERR 和 URXIF 影响。

Bit 1 **UTIIE**：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 UTIIE=1，当发送器空闲触发 UTIDLE 置位时，USIM 的中断请求标志 USIMF 置位；若 UTIIE=0，USIM 中断请求标志 USIMF 不受 UTIDLE 的影响。

Bit 0 **UTEIE**：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 UTEIE=1，当发送器为空触发 UTXIF 置位时，USIM 的中断请求标志 USIMF 置位；若 UTEIE=0，USIM 中断请求标志 USIMF 不受 UTXIF 的影响。

- **UUCR3 寄存器**

UUCR3 寄存器用于使能 UART 单线模式通信。顾名思义，在单线模式下 UART 只需要使用一条线，URX/UTX，在 UUCR2 寄存器中的 URXEN 和 UTXEN 位控制下即可完成通信。

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	USWM
R/W	—	—	—	—	—	—	—	R/W
POR	—	—	—	—	—	—	—	0

Bit 7~1 未定义，读为“0”

Bit 0 **USWM**：单线模式使能控制
0：除能，URX/UTX 引脚仅用作 UART 接收功能
1：使能，URX/UTX 引脚在 URXEN 和 UTXEN 位控制下可用作接收或发送功能
需注意的是，单线模式使能时，若将 URXEN 和 UTXEN 位同时设置为高，URX/UTX 引脚用作接收功能。
注：对于 BC66F2235，当选择 UART 模式时，此位需置高。

- **UTXR_RXR 寄存器**

UTXR_RXR 是一个数据寄存器，用来存储 UTX 引脚将要发送或 URX/UTX 引脚正在接收的数据。

Bit	7	6	5	4	3	2	1	0
Name	UTXRX7	UTXRX6	UTXRX5	UTXRX4	UTXRX3	UTXRX2	UTXRX1	UTXRX0
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 **UTXRX7~UTXRX0**：UART 发送 / 接收数据位 Bit 7~Bit 0

- **UBRG 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	UBRG7	UBRG6	UBRG5	UBRG4	UBRG3	UBRG2	UBRG1	UBRG0
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 **UBRG7~UBRG0**：波特率值

软件设置 UUCR2 寄存器中的 UBRGH 位 (设置波特率发生器的速度) 和 UBRG 寄存器 (设置波特率的值)，一起控制 UART 的波特率。

注：若 UBRGH=0，波特率 = f_H/[64×(N+1)]；

若 UBRGH=1，波特率 = f_H/[16×(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 UBRG 寄存器和 UUCR2 寄存器的 UBRGH 位来控制。UBRGH 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。UBRG 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。

UUCR2 的 UBRGH 位	0	1
波特率 (BR)	f_H / [64 (N+1)]	f_H / [16 (N+1)]

为得到相应的波特率，首先需要设置 UBRGH 来选择相应的计算公式从而算出 UBRG 的值。由于 UBRG 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 UBRG 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 UBRGH=0，若期望的波特率为 4800，计算它的 UBRG 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×64)] - 1

带入参数 N = [4000000 / (4800×64)] - 1 = 12.0208

取最接近的值，十进制 12 写入 UBRG 寄存器，实际波特率如下

BR = 4000000 / [64 × (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 UUCR1 寄存器的 UBNO、UPRT、UPREN 和 USTOPS 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 UUCR1 寄存器的 UREN 位来使能和除能的。当 SIMC0 寄存器中的 UMD 位已设置为“1”选择 UART 模式，若 UREN、UTXEN 和 URXEN 都为高，则 UTX 和 URX/UTX 分别为 UART 的发送端口和接收端口。若没有数据发送，UTX 引脚默认状态为高电平。

UREN 清零将除能 UTX 和 URX/UTX，通过设置相关引脚共用控制位，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 UTXEN、URXEN、UTXBRK、URXIF、UOERR、UFERR、UPERR 和 UNF 清零，而 UTIDLE、UTXIF 和 URIDLE 置位，UUCR1、UUCR2、UUCR3 和 UBRG 寄存器中的其它位保持不变。若 UART 工作时 UREN 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。

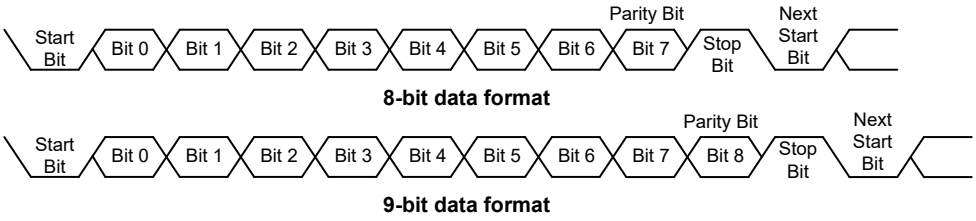
数据位、停止位位数以及奇偶校验的选择

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 UUCR1 寄存器的各个位控制的。UBNO 决定数据传输是 8 位还是 9 位；UPRT 决定校验类型；UPREN 决定是否选择奇偶校验；而 USTOPS 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

UUCR1 寄存器的 UBNO 位是控制数据传输的长度。UBNO=1 其长度为 9 位，第 9 位 MSB 存储在 UUCR1 寄存器的 UTX8 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 UTXR_RXR 提供，应用程序只须将发送数据写入 UTXR_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 UTXR_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器，所以应用程序不能对其进行读写操作。UTXEN=1，发送使能，但若 UTXR_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 UTXR_RXR 寄存器再置高 UTXEN 也会触发发送。当发送器使能，若 TSR 寄存器为空，数据写入 UTXR_RXR 寄存器将会直接加载到 TSR 寄存器中。发送器工作时，UTXEN 清零，发送器将立刻停止工作并且复位，此时通过设置相关引脚共用控制位，UTX 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 UTX 引脚上，其低位在前高位在后。在发送模式中，UTXR_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 UUCR1 寄存器的 UTX8。

发送器的启动可由如下步骤完成：

- 正确地设置 UBNO、UPRT、UPREN 和 USTOPS 位以确定数据长度、校验类型和停止位长度。
- 设置 UBRG 寄存器，选择期望的波特率。
- 置高 UTXEN，使能 UART 发送器且使 UTX 作为 UART 的发送端。
- 读取 UUSR 寄存器，然后将待发数据写入 UTXR_RXR 寄存器。注意，此步骤会清除 UTXIF 标志位。

如果要发送多个数据只需重复上一步骤。

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

1. 读取 UUSR 寄存器
2. 写 UTXR_RXR 寄存器

只读标志位 UTXIF 由 UART 硬件置位。若 UTXIF=1，UTXR_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 UTEIE=1，UTXIF 标志位会产生中断。在数据传输时，写 UTXR_RXR 指令会将待发数据暂存在 UTXR_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 UTXR_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 UTXIF 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 UTIDLE 位将被置位。

可以通过以下步骤来清除 UTIDLE：

1. 读取 UUSR 寄存器
2. 写 UTXR_RXR 寄存器

清除 UTXIF 和 UTIDLE 软件执行次序相同。

发送暂停字

若 UTXBRK=1 且此状态保持时间超过 [(BRG+1) × t_H] 且 UTIDLE=1，下一帧将会发送暂停字。它是由一个起始位、13×N (N=1, 2……) 位逻辑 0 组成。置位 UTXBRK 将会发送暂停字，而清除 UTXBRK 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 UTXBRK 持续为高，那么发送器会一直发送暂停字；当应用程序将 UTXBRK 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 UBNO=1，数据长度为 9 位，而最高位 MSB 存放在 UUCR1 寄存器的 URX8 中。接收器的核心是串行移位寄存器 RSR。URX/UTX 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 URX/UTX 引脚上检测到停止位，若 UTXR_RXR 寄存器为空，数据从 RSR 寄存器中加载到 UTXR_RXR 寄存器。URX/UTX 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器，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 URX/UTX 引脚进入移位寄存器。UTXR_RXR 寄存器在内部总线和接收移位寄存器间形成一个缓冲。UTXR_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 UTXR_RXR 寄存器，否则忽略第三帧数据并且发生溢出错误。

接收器的启动可由如下步骤完成：

- 正确地设置 UBNO、UPRT 和 UPREN 位以确定数据长度和校验类型。
- 设置 UBRG 寄存器，选择期望的波特率。
- 置高 URXEN，使能 UART 接收器且使 URX/UTX 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 UTXR_RXR 寄存器中包含有效数据时，UUSR 寄存器中的 URXIF 位将会置位，溢出错误发生之前至多还有一帧数据可读。
- 若 URIE=1，数据从 RSR 寄存器加载到 UTXR_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 URXIF：

1. 读取 UUSR 寄存器
2. 读取 UTXR_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 UBNO 位的设置外加一个停止位来确定一帧数据的长度。若暂停字位数大于 UBNO 位指定的长度外加一个停止位，接收器认为接收已完毕，URXIF 和 UFERR 置位，UTXR_RXR 寄存器清 0，若相应的中断允许且 URIDLE 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 UFERR 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 UFERR 标志位。在下个开始位到来之前，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，没有检测到停止位也会置位只读标志位 URIDLE。

UART 接收到暂停字会产生以下事件：

- 帧错误标志位 UFERR 置位。
- UTXR_RXR 寄存器清零。
- UOERR、UNF、UPERR、URIDLE 或 URXIF 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，UUSR 寄存器的接收状态标志位 URIDLE 清零。在停止位和下一帧数据的起始位之间，URIDLE 被置位，表示接收器空闲。

接收中断

UUSR 寄存器的只读标志位 URXIF 由接收器的边沿触发置位。若 URIE=1，数据从移位寄存器 RSR 加载到 UTXR_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 – UOERR 标志

UTXR_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 UTXR_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：

- UUSR 寄存器中 UOERR 被置位。
- UTXR_RXR 寄存器中数据不会丢失。
- RSR 寄存器数据将会被覆盖。
- 若 URIE=1，将会产生中断。

先读取 UUSR 寄存器再读取 UTXR_RXR 寄存器可将 UOERR 清零。

噪声干扰 – UNF 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 URXIF 上升沿，UUSR 寄存器中只读标志位 UNF 置位。
- 数据从 RSR 寄存器加载到 UTXR_RXR 寄存器中。

- 不产生中断，但此位置位发生在 URXIF 置位产生中断的同周期内。

先读取 UUSR 寄存器再读取 UTXR_RXR 寄存器可将 UNF 清零。

帧错误 – UFERR 标志

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

奇偶校验错误 – UPERR 标志

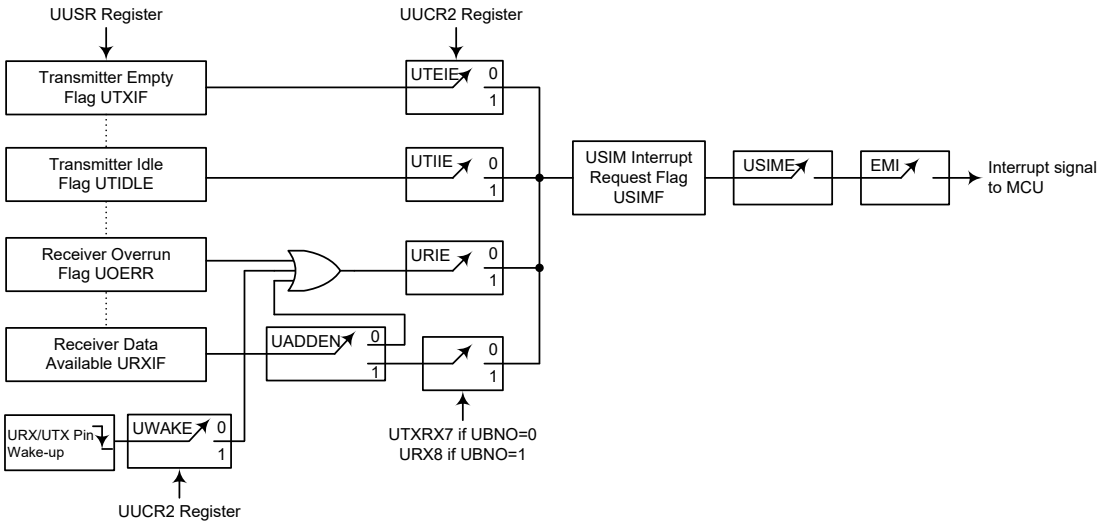
若接收到数据出现奇偶校验错误，UUSR 寄存器中只读标志 UPERR 置位。只有使能了奇偶校验，选择了校验类型，此标志位才有效。此标志位同接收的数据分别记录在 UUSR 寄存器和 UTXR_RXR 寄存器中，此标志位可被任何复位清零。注意，在读取相应的数据之前必须先访问 UUSR 寄存器中的 UFERR 和 UPERR 错误标志位。

UART 模块中断结构

几个独立的 UART 条件可以产生一个 USIM 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 URX/UTX 引脚唤醒都会产生中断。若总中断使能、USIM 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 UUCR2 寄存器中相应中断允许位被置位，则 UUSR 寄存器中对应中断标志位将产生 USIM 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 USIM UART 模式中断源。

地址检测也是 USIM UART 模式的中断源，它没有相应的标志位，若 UUCR2 寄存器中 UADDEN=1，当检测到地址将会产生 USIM 中断。URX/UTX 引脚唤醒也可以产生 USIM 中断，它没有相应的标志位，当 UART 时钟源 f_H 关闭且 UUCR2 中的 UWAKE 和 URIE 位被置位，URX/UTX 引脚上有下降沿时会产生 USIM 中断。应注意 URX/UTX 唤醒中断发生时将会有一定时间的延迟，即系统上电时间，从而允许振荡器在系统恢复正常操作之前重启并达到稳定。

注意，UUSR 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 USIM 中断控制寄存器中的相关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 UUCR2 寄存器中的 UADDEN 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 URXIF。若 UADDEN 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 USIME 和 EMI 也要使能才会产生中断。地址的最高位为第 9 位 (UBNO=1) 或第 8 位 (UBNO=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 UADDEN 除能，每接收到一个有效数据便会置位 URXIF，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。

UADDEN	9th Bit (UBNO=1) 8th Bit (UBNO=0)	产生 USIM 中断
0	0	√
	1	√
1	0	×
	1	√

UADDEN 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，UUSR、UUCR1、UUCR2、UUCR3、UTXR_RXR 以及 UBRG 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 URX/UTX 引脚的唤醒功能，由 UUCR2 寄存器中 UWAKE 位控制。当 UART 时钟 f_H 关闭时，若 UWAKE 位与 UART 模式选择位 UMD、UART 允许位 UREN、接收器使能位 URXEN 和接收器中断使能位 URIE 都被置位，则 URX/UTX 引脚的下降沿可触发产生 URX/UTX 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，URX/UTX 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 USIM 中断使能控制位 USIME 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 USIM 中断。

触控按键功能

该系列单片机提供多个触控按键功能。该触控按键功能完全内部集成不需外接元件，通过内部寄存器对其进行简单的操作。

触控按键结构

触控按键与 I/O 引脚共用，通过对应的引脚共用功能选择寄存器的位来选择此功能。按键被分成多组，每组为一个模块，编号为 M0~M3。每个模块为独立的一组，包含四个触控按键且每个按键有各自的振荡器。每个模块具有单独的控制逻辑电路和配套的寄存器系列。寄存器的名称和它相关的模块编号相对应。

单片机型号	触控按键总数	触控按键模块	触控按键
BC66F2235	8	M0	KEY1
		M1	KEY5~KEY6
		M2	KEY11~KEY12
		M3	KEY13~KEY15
BC66F2245	14	M0	KEY1~KEY4
		M1	KEY5~KEY6
		M2	KEY9~KEY12
		M3	KEY13~KEY16
BC66F2255	16	M0	KEY1~KEY4
		M1	KEY5~KEY8
		M2	KEY9~KEY12
		M3	KEY13~KEY16

触控按键结构

触控按键寄存器定义

每个触控按键模块包含四个触控按键功能，且都有其配套的寄存器。以下表格显示了每个触控按键模块的寄存器系列。寄存器名称里的 Mn 对应触控按键模块的序号，该系列单片机具有四个触控按键模块。

寄存器名称	说明
TKTMR	触控按键时隙 8-bit 计数器预载寄存器
TKC0	触控按键功能控制寄存器 0
TKC1	触控按键功能控制寄存器 1
TKC2	触控按键功能控制寄存器 2
TK16DL	触控按键功能 16-bit 计数器低字节
TK16DH	触控按键功能 16-bit 计数器高字节
TKMn16DL	触控按键模块 n 16-bit C/F 计数器低字节
TKMn16DH	触控按键模块 n 16-bit C/F 计数器高字节
TKMnROL	触控按键模块 n 参考振荡器电容选择低字节

寄存器名称	说明
TKMnROH	触控按键模块 n 参考振荡器电容选择高字节
TKMnC0	触控按键模块 n 控制寄存器 0
TKMnC1	触控按键模块 n 控制寄存器 1
TKMnC2	触控按键模块 n 控制寄存器 2
TKMnTH16L	触控按键模块 n 16-bit 阈值低字节
TKMnTH16H	触控按键模块 n 16-bit 阈值高字节
TKMnTHS	触控按键模块 n 阈值比较标志

触控按键功能寄存器定义 (n=0~3)

寄存器名称	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TKTMR	D7	D6	D5	D4	D3	D2	D1	D0
TKC0	TKRAMC	TKRCOV	TKST	TKCFOV	TK16OV	TKMOD1	TKMOD0	TKBUSY
TKC1	D7	D6	D5	TSCS	TK16S1	TK16S0	TKFS1	TKFS0
TKC2	—	—	—	—	—	TSC	ASMP1	ASMP0
TK16DL	D7	D6	D5	D4	D3	D2	D1	D0
TK16DH	D15	D14	D13	D12	D11	D10	D9	D8
TKMn16DL	D7	D6	D5	D4	D3	D2	D1	D0
TKMn16DH	D15	D14	D13	D12	D11	D10	D9	D8
TKMnROL	D7	D6	D5	D4	D3	D2	D1	D0
TKMnROH	—	—	—	—	—	—	D9	D8
TKMnC0	—	—	MnDFEN	—	MnSOFC	MnSOF2	MnSOF1	MnSOF0
TKMnC1	MnTSS	—	MnROEN	MnKOEN	MnK4EN	MnK3EN	MnK2EN	MnK1EN
TKMnC2	MnSK31	MnSK30	MnSK21	MnSK20	MnSK11	MnSK10	MnSK01	MnSK00
TKMnTH16L	D7	D6	D5	D4	D3	D2	D1	D0
TKMnTH16H	D15	D14	D13	D12	D11	D10	D9	D8
TKMnTHS	MnK4THF	MnK3THF	MnK2THF	MnK1THF	MnK4THS	MnK3THS	MnK2THS	MnK1THS

触控按键功能寄存器列表 (n=0~3)

- **TKTMR 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**：触控按键时隙 8-bit 计数器预载寄存器

触控按键时隙计数器预载寄存器用于确定触控按键时隙溢出时间。若时隙单位周期是通过一个 5-bit 计数器获得，则等于 32 个时隙时钟周期。若时隙单位周期是通过旁路一个 5-bit 计数器获得，则等于 1 个时隙时钟周期。因此，时隙计数器溢出时间可由下面的等式算出。

当 MnTSS=0 时，时隙计数器溢出时间 = (256-TKTMR[7:0])×32t_{TSC}；当 MnTSS=1 时，时隙计数器溢出时间 = (256-TKTMR[7:0])×t_{TSC}，其中 t_{TSC} 为时隙计数器时钟周期。

- **TKC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	TKRAMC	TKRCOV	TKST	TKCFOV	TK16OV	TKMOD1	TKMOD0	TKBUSY
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
POR	0	0	0	0	0	0	1	0

Bit 7 **TKRAMC**：触控按键数据存储器存取控制位
0：由 MCU 读 / 写
1：由触控按键模块读 / 写
该位用于确定触控按键模块数据存储器是由 MCU 还是触控按键模块使用。若触控按键模块工作在自动扫描模式或周期性自动扫描模式 (设置 TKMOD1~TKMOD0 位为 00、10 或 11，当 TKST 位由 0 变为 1 时可进入自动扫描模式或周期性自动扫描模式)，触控按键模块具有对该数据存储器进行存取的优先权。当自动扫描或周期性自动扫描操作完成后 (即 TKBUSY 位状态由 1 转为 0)，触控按键模块数据存储器的存取才可通过 TKRAMC 位控制。因此当触控按键模块工作在自动扫描模式或周期性自动扫描模式时，建议将 TKRAMC 位设置为“1”。否则，此数据存储器的内容可能会在触控按键模块配置完成后又被 MCU 的其它操作改动。当 TKRAMC 位设置为“1”时，即触控按键数据存储器由触控按键模块控制，此时通过 MCU 读取触控按键数据存储器得到的是不确定值。

Bit 6 **TKRCOV**：触控按键时隙计数器溢出标志位
0：无溢出
1：溢出
此位可通过应用程序读 / 写。当触控按键时隙计数器溢出将此位置为“1”时，相应的触控按键 TKRCOV 中断请求标志位也会同时置位。然而若是通过应用程序将此位设置为“1”时，相应的触控按键 TKRCOV 中断请求标志位不会受到影响。因此，该位不能通过应用程序置位，但必须通过应用程序清零。
在自动扫描模式时，如果模块 0 或所有模块 (由 TSCS 位选择) 的时隙计数器溢出，但自动扫描还未完成，TKRCOV 位将不会被置位，同时所有触控按键模块的 16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器将会自动清零，但 8-bit 时隙计数器将会从 8-bit 时隙计数器预载寄存器 (TKTMR 寄存器) 重新加载数据。当自动扫描工作结束，TKRCOV 位及触控按键 TKRCOV 中断请求标志位 TKRCOVF 将被置位，同时所有模块的按键振荡器和参考振荡器自动停止。所有模块的 16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器都会自动关闭。
在周期性自动扫描模式时，在自动扫描工作周期内，TKRCOV 位将被清零。仅在 TB0 溢出周期内的最后一次扫描结束时，16-bit C/F 计数器内容将被写入相应的触控按键数据存储器，TKRCOV 位将被硬件电路置高。除了以上提到的，该模式下的其它动作和自动扫描模式一样。
在手动扫描模式时，若模块 0 或所有模块 (通过 TSCS 位选择) 时隙计数器溢出，则 TKRCOV 位及触控按键 TKRCOV 中断请求标志位 TKRCOVF 将被置位，且所有模块的按键振荡器和参考振荡器自动停止。此时所有触控按键模块的 16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器也都会自动关闭。

Bit 5 **TKST**：触控按键检测开启控制位
0：停止或无操作
0→1：开始检测
当该位为“0”时，所有模块的 16-bit C/F 计数器、触控按键功能 16-bit 计数器和 5-bit 时隙单位周期计数器会自动清零，但 8-bit 可编程时隙计数器不会被清零。当该位由 0 到 1 转变时，触控按键模块 16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器都会自动开启，并使能按键振荡器和参考振荡器以驱动相应的计数器。

Bit 4 **TKCFOV**：触控按键模块 16-bit C/F 计数器溢出标志位
0：无溢出
1：溢出
该位由触控按键模块 16-bit C/F 计数器溢出置位，必须通过应用程序清零。

Bit 3 **TK16OV**：触控按键功能 16-bit 计数器溢出标志位
0：无溢出
1：溢出
该位由触控按键功能 16-bit 计数器溢出置位，必须通过应用程序清零。

Bit 2~1 **TKMOD1~TKMOD0**：触控按键扫描模式选择位
00：自动扫描模式
01：手动扫描模式
10/11：周期性自动扫描模式
在手动扫描模式时，应在扫描开始之前，合理的设置参考振荡器电容值，并在扫描结束后通过应用程序读取触控按键模块 16-bit C/F 计数器值。
在自动扫描模式时，上面所说的数据的读取或写入是通过硬件完成。有一个专用的触控按键模块数据存储区可存放所有被扫描按键的参考振荡器电容值及 16-bit C/F 计数器值。自动扫描按键的顺序是由 TKMnC2 寄存器的 MnSK3[1:0]~MnSK0[1:0] 位指定。直至扫描完所有计划扫描的按键后，自动扫描工作才会停止。
在周期性自动扫描模式时，触控按键扫描工作将周期性地自动进行，可由 TKC2 寄存器的 ASMP1~ASMP0 位决定。仅在 TB0 溢出周期内的最后一次扫描结束时，16-bit C/F 计数器内容将被写入计数器内容将被写入相应的触控按键数据存储器。另外，当 MnKmTHS 位为 0 时，任何一个按键 C/F 计数器值小于下限阈值，或 MnKmTHS 为 1 时，任何一个按键 C/F 计数器值大于上限阈值，TKTH 信号都将被置高。除了以上提到的，该模式下的其它动作和自动扫描模式一样。

Bit 0 **TKBUSY**：触控按键扫描忙碌标志位
0：空闲 – 没有在执行按键扫描或按键扫描已完成
1：忙碌 – 正在扫描中
该位用于指示触控按键扫描工作是否在执行中。当 TKST 位置高启动扫描工作，该位被置“1”。
在手动扫描模式中，当模块 0 或所有模块 (由 TSCS 位选择) 的触控时隙计数器溢出时，该位会自动清零。在自动扫描模式中，当触控按键扫描工作完成时，该位也会自动清零。在周期性自动扫描模式时，若在 TB0 溢出周期内完成最后一次扫描工作，或当 MnKmTHS 位为 0，任何一个按键 C/F 计数器值小于下限阈值时，或当 MnKmTHS 为 1，任何一个按键 C/F 计数器值大于上限阈值时，这些情况下，该位都会被清零。

- **TKC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	TSCS	TK16S1	TK16S0	TKFS1	TKFS0
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	1	1

Bit 7~5 **D7~D5**：测试数据位
这些位仅供内部测试使用，正常工作时，须固定为“000”。

Bit 4 **TSCS**：触控按键时隙计数器选择位
0：每个模块使用自己的时隙计数器
1：所有触控按键模块使用模块 0 的时隙计数器

Bit 3~2 **TK16S1~TK16S0**：触控按键功能 16-bit 计数器时钟源选择位
00：f_{SYS}
01：$f_{SYS}/2$
10：$f_{SYS}/4$
11：$f_{SYS}/8$

Bit 1~0 **TKFS1~TKFS0**：触控按键振荡器和参考振荡器频率选择位
00：1MHz
01：3MHz
10：7MHz
11：11MHz

- **TKC2 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	TSC	ASMP1	ASMP0
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	1

Bit 7~3 未定义，读为“0”

Bit 2 **TSC**：时隙控制
0：时隙 0~3 (默认)
1：仅时隙 0
该位用来配置时隙功能。若 TSC 位被置高，只有时隙 0 有效并执行触控按键相关操作，时隙 1~3 无效。在空闲、休眠或深度休眠模式下可以减少功耗并执行一键唤醒功能。在时隙 0 内用于唤醒 MCU 的按键由 TKMnC2 寄存器的 MnSK01~MnSK00 位选择。

Bit 1~0 **ASMP1~ASMP0**：周期性自动扫描模式周期选择位
00：$2^{14}/f_{PSC0}$
01：$2^{13}/f_{PSC0}$
10：$2^{12}/f_{PSC0}$
11：$2^{11}/f_{PSC0}$
这些位用来决定触控按键扫描周期，仅当触控按键功能被配置为工作在周期性自动扫描模式时有效。触控按键扫描次数可通过 TB0 溢出周期 t_{TB0} 得到，周期性自动扫描模式周期 t_{KEY} 可通过等式 $N=t_{TB0}/t_{KEY}$ 得到。TB0 溢出周期 t_{TB0} 可由 TB02~TB00 位选择。
例如：若将 TB0C[2:0] 位设为 100，选择 f_{PSC0} 为 f_{SUB} 且 TB0 溢出周期为 $2^{15}/f_{PSC0}$，t_{TB0} 等于 1s。因此，在一个 TB0 溢出周期内，当 ASMP[1:0] 位分别设为 00/01/10/11 时，触控按键扫描次数为 2/4/8/16 次。在应用上确保周期性自动扫描模式周期 t_{KEY} 不会超过 TB0 溢出周期 t_{TB0} 是非常重要的。

- **TK16DH/TK16DL – 触控按键功能 16-bit 计数器寄存器对**

寄存器	TK16DH								TK16DL							
位	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

该寄存器对用于存储触控按键功能 16-bit 计数器值。该 16-bit 计数器可用于校准参考振荡器或按键振荡器频率。手动扫描模式中，当触控按键时隙计数器溢出，此 16-bit 计数器将停止，计数器内容保持不变。自动扫描模式或周期性自动扫描模式中，在时隙 0、时隙 1 和时隙 2 扫描结束时，该 16-bit 计数器值会被清零，但在时隙 3 结束时，该 16-bit 计数器值不会改变。当 TKST 位为“0”时，该寄存器对将被清零。

- **TKMn16DH/TKMn16DL – 触控按键模块 n 16-bit C/F 计数器寄存器对**

寄存器	TKMn16DH								TKMn16DL							
位	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

该寄存器对用于存储触控按键模块 n 16-bit C/F 计数器值。手动扫描模式中，当触控按键时隙计数器溢出，该 16-bit C/F 计数器将被停止，其内容保持不变。自动扫描模式或周期性自动扫描模式中，在时隙 0、时隙 1 和时隙 2 扫描结束时，此 16-bit C/F 计数器内容会先被写入到对应的触控按键存储器，然后被清零，但在时隙 3 结束时，该 16-bit C/F 计数器值不会改变。当 TKST 位为“0”时，该寄存器对将被清零。

- **TKMnROH/TKMnROL – 触控按键模块 n 参考振荡器电容选择寄存器对**

寄存器	TKMnROH								TKMnROL							
位	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8	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	R/W	R/W
POR	—	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0

该寄存器对用于存储触控按键模块 n 参考振荡器电容值。自动扫描模式或周期性自动扫描模式中，该寄存器对会在当前时隙扫描结束后，从专用的触控按键模块数据存储器中载入下一时隙要扫描按键对应的参考振荡器电容值。

参考振荡器内部电容值 = (TKMnRO[9:0]×50pF)/1024

- **TKMnC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	MnDFEN	—	MnSOFC	MnSOF2	MnSOF1	MnSOF0
R/W	—	—	R/W	—	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　**MnDFEN**：触控按键模块 n 倍频功能控制位
0：除能
1：使能
此位用于控制触控按键振荡器的倍频功能。当此位置 1，按键振荡器频率将变为原来的两倍。

Bit 4　未定义，读为“0”

Bit 3　**MnSOFC**：触控按键模块 n C/F 振荡器跳频功能控制位
0：由 MnSOF2~MnSOF0 位控制
1：由硬件电路控制
该位用来选择触控按键振荡器跳频功能控制方式。当此位置 1，无论 MnSOF2~MnSOF0 位设置为何值，按键振荡器跳频功能由硬件电路控制。

Bit 2~0　**MnSOF2~MnSOF0**：触控按键模块 n 参考振荡器和按键振荡器跳频选择位 (MnSOFC=0)
000：1.020MHz
001：1.040MHz
010：1.059MHz

011：1.074MHz
100：1.085MHz
101：1.099MHz
110：1.111MHz
111：1.125MHz

这些位用于触控按键振荡器跳频功能的频率选择。注意，只有当 MnSOFC 位为零时，这些位的选择才有效。

上述频率仅适用于按键振荡器和参考振荡器频率选择 1MHz 的情况，这些数值会随着外部或内部电容值的不同而变化。用户选择其它按键振荡器和参考振荡器频率时，可依此比例进行调整。

- **TKMnC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	MnTSS	—	MnROEN	MnKOEN	MnK4EN	MnK3EN	MnK2EN	MnK1EN
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 **MnTSS**：触控按键模块 n 时隙计数器时钟源选择位
0：触控按键模块 n 参考振荡器
1：f_{LIRC}

MnTSS 位置高时，触控按键模块 n 时隙计数器时钟来自 f_{LIRC} 振荡器，即使在空闲、休眠或深度休眠模式下该时钟也继续运行。触控按键模块 n 时隙计数器时钟可旁路 5-bit 计数器以缩短溢出时间。在周期性自动扫描模式下，设置 MnTSS=1 且 TSC=1，可降低按键的待机功耗。

注：如果旁路了 5-bit 计数器，触控按键模块 n 参考和按键振荡器跳频频率由 MnSOF2~MnSOF0 位控制，无论 MnSOFC 位的设置为何值。

Bit 6 未定义，读为“0”

Bit 5 **MnROEN**：触控按键模块 n 参考振荡器使能控制位
0：除能
1：使能

在手动扫描模式中，该位用于使能 / 除能触控按键模块 n 参考振荡器。若需使用模块 n 参考振荡器，在将 TKST 位由低设置为高前，必须先使能此参考振荡器。当 TKBUSY 位从高变为低时，参考振荡器将自动除能。

在自动扫描模式或周期性自动扫描模式中，该位由硬件自动控制。对于模块 0，当 TKST 位由低到高时，M0ROEN 位会被自动置 1；对于其它模块 n (n≠0)，若 MnK4EN~MnK1EN≠0000B、MnTSS=0 且 TSCS=0，当 TKST 位由低到高时，MnROEN 位会被自动置 1。其它情况下，MnROEN 位不受 TKST 设置的影响。当 TKBUSY 位从高变为低时，MnROEN 位自动清零以除能参考振荡器。

Bit 4 **MnKOEN**：触控按键模块 n 按键振荡器使能控制位
0：除能
1：使能

在手动扫描模式中，该位用于使能 / 除能模块 n 按键振荡器。若相关按键已被使能且要被扫描，在设置 TKST 由低到高前，应先使能相应的模块 n 按键振荡器。当 TKBUSY 由高变为低时，按键振荡器自动除能。

在自动扫描模式或周期性自动扫描模式中，该位由硬件自动控制。当 TKST 位由低到高，MnKOEN 位会被自动置 1 以使能按键振荡器。若 TKBUSY 位发生由高到低的转变，MnKOEN 位将自动清零以除能按键振荡器。

Bit 3 **MnK4EN**：触控按键模块 n 的 Key 4 使能控制

MnK4EN	触控按键模块 n – Mn			
	M0	M1	M2	M3
0：除能	I/O 或其它功能			
1：使能	KEY4	KEY8	KEY12	KEY16
BC66F2235	—	—	√	—
BC66F2245	√	—	√	√
BC66F2255	√	√	√	√

注：“—”表示该型号对应的位为保留位且需固定为“0”。

Bit 2 **MnK3EN**：触控按键模块 n 的 Key 3 使能控制

MnK3EN	触控按键模块 n – Mn			
	M0	M1	M2	M3
0：除能	I/O 或其它功能			
1：使能	KEY3	KEY7	KEY11	KEY15
BC66F2235	—	—	√	√
BC66F2245	√	—	√	√
BC66F2255	√	√	√	√

注：“—”表示该型号对应的位为保留位且需固定为“0”。

Bit 1 **MnK2EN**：触控按键模块 n 的 Key 2 使能控制

MnK2EN	触控按键模块 n – Mn			
	M0	M1	M2	M3
0：除能	I/O 或其它功能			
1：使能	KEY2	KEY6	KEY10	KEY14
BC66F2235	—	√	—	√
BC66F2245	√	√	√	√
BC66F2255	√	√	√	√

注：“—”表示该型号对应的位为保留位且需固定为“0”。

Bit 0 **MnK1EN**：触控按键模块 n 的 Key 1 使能控制

MnK1EN	触控按键模块 n – Mn			
	M0	M1	M2	M3
0：除能	I/O 或其它功能			
1：使能	KEY1	KEY5	KEY9	KEY13
BC66F2235	√	√	—	√
BC66F2245	√	√	√	√
BC66F2255	√	√	√	√

注：“—”表示该型号对应的位为保留位且需固定为“0”。

- **TKMnC2 寄存器**

该寄存器用于选择触控按键模块 n 在时隙 0~3 时要被扫描的按键。应注意的是，如果任意时隙内对应的按键除能时，此时隙内的参考振荡器和按键振荡器均不振荡。

Bit	7	6	5	4	3	2	1	0
Name	MnSK31	MnSK30	MnSK21	MnSK20	MnSK11	MnSK10	MnSK01	MnSK00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	0	0	1	0	0

Bit 7~6 **MnSK31~MnSK30**：触控按键模块 n 时隙 3 扫描按键选择位
00：Key 1
01：Key 2
10：Key 3
11：Key 4
这两位用于选择在自动扫描模式或周期性自动扫描模式下时隙 3 时要被扫描的按键。这两位的设置在 TKMOD1~TKMOD0=01 或 TSC=1 时无效。

Bit 5~4 **MnSK21~MnSK20**：触控按键模块 n 时隙 2 扫描按键选择位
00：Key 1
01：Key 2
10：Key 3
11：Key 4
这两位用于选择在自动扫描模式或周期性自动扫描模式下时隙 2 时要被扫描的按键。这两位的设置在 TKMOD1~TKMOD0=01 或 TSC=1 时无效。

Bit 3~2 **MnSK11~MnSK10**：触控按键模块 n 时隙 1 扫描按键选择位
00：Key 1
01：Key 2
10：Key 3
11：Key 4
这两位用于选择在自动扫描模式或周期性自动扫描模式下时隙 1 时要被扫描的按键。这两位的设置在 TKMOD1~TKMOD0=01 或 TSC=1 时无效。

Bit 1~0 **MnSK01~MnSK00**：触控按键模块 n 时隙 0 扫描按键选择位
00：Key 1
01：Key 2
10：Key 3
11：Key 4
这两位用于选择在自动扫描模式或周期性自动扫描模式下时隙 0 时要被扫描的按键或在手动扫描模式时用作扫描按键选择的多路复用器。

- **TKMnTH16H/TKMnTH16L – 触控按键模块 n 16-bit 阈值寄存器对**

寄存器	TKMnTH16H								TKMnTH16L							
位	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	D15	D14	D13	D12	D11	D10	D9	D8	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	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	0	0	0	0	0	0	0	0

该寄存器对用于存储触控按键模块 n 16-bit 上限 / 下限阈值。当选中周期性自动扫描模式时，该寄存器对将在当前时隙结束时，由硬件自动从专用的触控按键数据存储器中载入相应的下一个 16-bit 上限 / 下限阈值。当触控按键模块 n 要扫描的按键 Key m (m=1~4) 完成扫描工作后，16-bit C/F 计数器内容 TKMn16DH/TKMn16DL 将和 TKMnTH16H/TKMnTH16L 值将通过硬件进行比较。当 MnKmTHS=0，该值小于下限阈值时或当 MnKmTHS=1，该值大于上限阈值，MnKmTHF 标志位将置高，中断信号将产生。

- **TKMnTHS 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	MnK4THF	MnK3THF	MnK2THF	MnK1THF	MnK4THS	MnK3THS	MnK2THS	MnK1THS
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 **MnK4THF**：触控按键模块 n Key 4 上限 / 下限阈值比较标志位
0：不小于下限阈值或不大于上限阈值
1：小于下限阈值或大于上限阈值

Bit 6 **MnK3THF**：触控按键模块 n Key 3 上限 / 下限阈值比较标志位
0：不小于下限阈值或不大于上限阈值
1：小于下限阈值或大于上限阈值

Bit 5 **MnK2THF**：触控按键模块 n Key 2 上限 / 下限阈值比较标志位
0：不小于下限阈值或不大于上限阈值
1：小于下限阈值或大于上限阈值

Bit 4 **MnK1THF**：触控按键模块 n Key 1 上限 / 下限阈值比较标志位
0：不小于下限阈值或不大于上限阈值
1：小于下限阈值或大于上限阈值

Bit 3 **MnK4THS**：触控按键模块 n Key 4 上限 / 下限阈值比较选择位
0：下限阈值比较
1：上限阈值比较

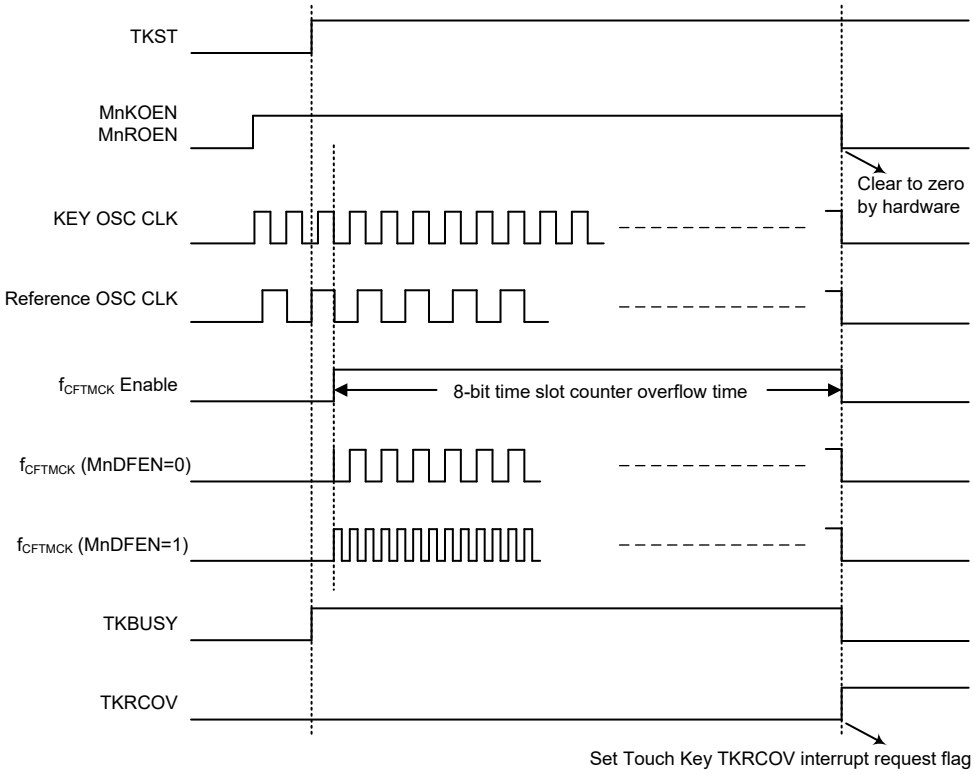
Bit 2 **MnK3THS**：触控按键模块 n Key 3 上限 / 下限阈值比较选择位
0：下限阈值比较
1：上限阈值比较

Bit 1 **MnK2THS**：触控按键模块 n Key 2 上限 / 下限阈值比较选择位
0：下限阈值比较
1：上限阈值比较

Bit 0 **MnK1THS**：触控按键模块 n Key 1 上限 / 下限阈值比较选择位
0：下限阈值比较
1：上限阈值比较

触控按键操作

手指接近或接触到触控面板时，面板的电容量会增大，电容量的变化会轻微改变内部感应振荡器的频率，通过测量频率的变化可以感知触控动作。参考时钟通过内部可编程分频器能够产生一个固定的时间周期。在这个时间周期内，通过在此固定时间周期内对感应振荡器产生的时钟周期计数，可确定触控按键的动作。



触控按键手动扫描模式时序图

每个触控按键模块包含四个与 I/O 引脚共用的触控按键输入，通过相关引脚共用控制寄存器位可设置相应引脚功能。每个触控按键具有独立的感应振荡器，因此每个模块包含四个感应振荡器。

在参考时钟固定的时间间隔内，感应振荡器产生的时钟周期数是可以测量的。测到的周期数可以用于判断触控动作是否有效发生。手动扫描模式中，在此固定的时间间隔结束后，会产生一个触控按键 TKRCOV 中断信号。

通过设置 TKC1 寄存器中的 TSCS 位可以选择模块 0 的时隙计数器作为所有模块的时隙计数器。所有的触控按键模块共用一个起始信号，即 TKC0 寄存器中的 TKST。在此位清零时，所有模块的 16-bit C/F 计数器、触控按键功能 16-bit 计数器和 5-bit 时隙单位周期计数器会自动清零，而 8-bit 可编程时隙计数器不清零，其溢出时间由用户设置。在 TKST 位由低变高时，16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器会自动开启。

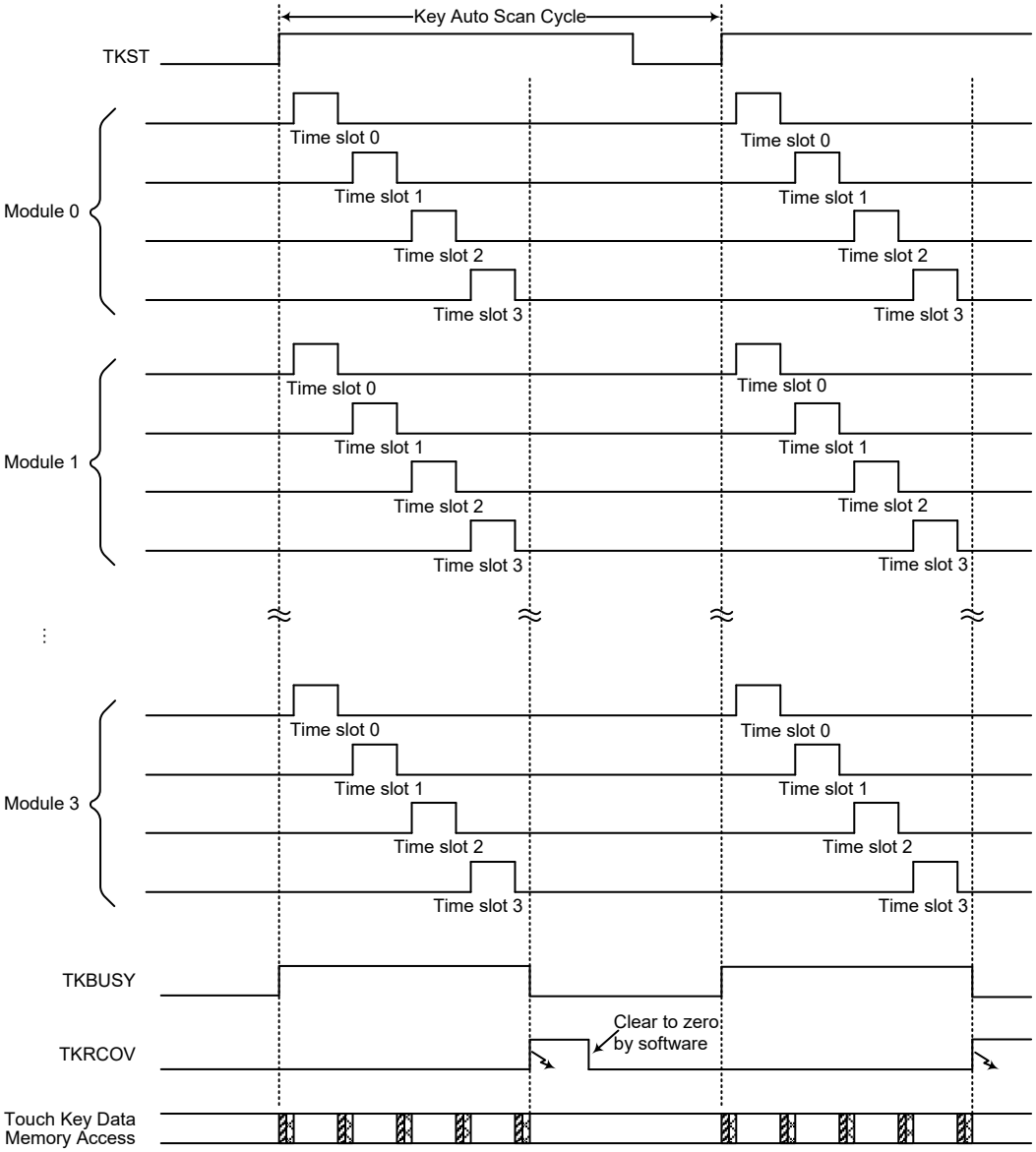
当时隙计数器溢出，所有模块的按键振荡器和参考振荡器都会自动停止且 16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器也会自动停止。时隙计数器时钟源可通过 TKMnC1 寄存器中的 MnTSS 位选择来自参考振荡器或 f_{LIRC}。通过设置 TKMnC1 寄存器中的 MnROEN 位和 MnKOEN 位为“1”，可使能参考振荡器和按键振荡器。

当所有触控按键模块的时隙计数器都溢出或模块 0 时隙计数器溢出时，将产生一个触控按键 TKRCOV 中断。这里所说的触控按键是指已使能的触控按键。

每四个按键为一个模块，所以 KEY1~KEY4 为模块 0，KEY5~KEY8 为模块 1，KEY9~KEY12 为模块 2，KEY13~KEY16 为模块 3，每个模块都是相同的架构。

自动扫描模式

触控按键功能包含三种按键扫描模式，即自动扫描模式、周期性自动扫描和手动扫描模式，可通过 TKC0 寄存器中的 TKMOD1~TKMOD0 位选择。自动扫描模式可以较大程度地减少程序负担并提高按键扫描执行效率。设置 TKMOD1~TKMOD0 位为 00 则选择自动扫描模式，在此模式中依指定顺序自动扫描每个模块的按键。扫描按键顺序由 TKMnC2 寄存器的 MnSK3[1:0]~MnSK0[1:0] 位决定。TKC2 寄存器的 TSC 位用于配置时隙。



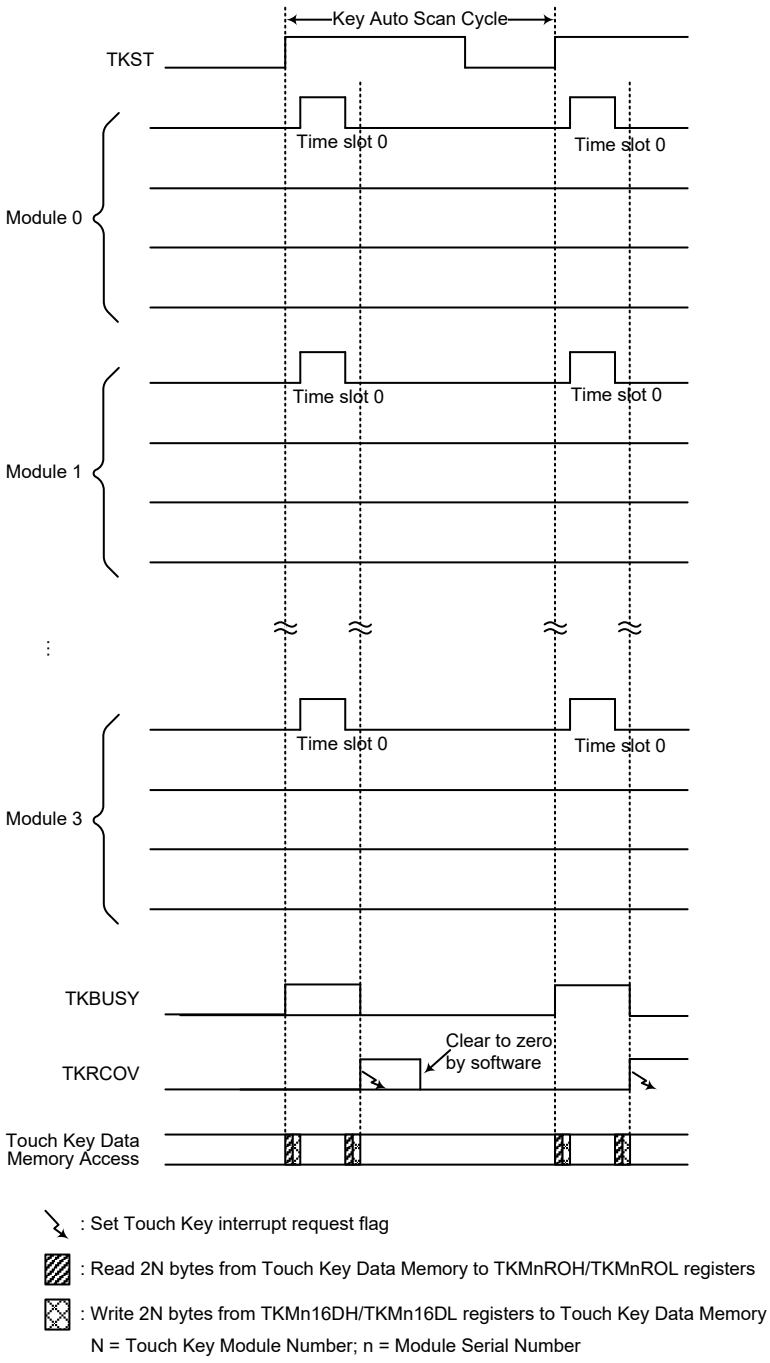
: Set Touch Key interrupt request flag

: Read 2N bytes from Touch Key Data Memory to TKMnROH/TKMnROL registers

: Write 2N bytes from TKMn16DH/TKMn16DL registers to Touch Key Data Memory

N = Touch Key Module Number; n = Module Serial Number

触控按键自动扫描模式时序图 – **TSC=0**



触控按键自动扫描模式时序图 – TSC=1

在自动扫描模式时，需要使用到的模块 n 按键振荡器和参考振荡器在 TKST 位由低变为高时自动使能，在 TKBUSY 位由高到低时自动除能。若 TSC 位为低，时隙 0~3 有效。在自动扫描模式中，当设置 TKST 位由低到高，硬件会自动先从专用触控按键数据存储器指定位置读取时隙 0 要扫描的按键对应的参考振荡器的电容值，并将此值写入相应的 TKMnROH/TKMnROL 寄存器对中。并将触控按键模块 n 的 16-bit C/F 计数器寄存器对中的值写入到上一轮时隙 3 扫描的按键对应的触控按键数据存储器位置。之后进入时隙 0 开始扫描所选的按键。

时隙 0 按键扫描结束时，硬件会自动从触控按键数据存储区读取下一个要扫描按键对应的参考振荡器电容值，并写入相应的 TKMnROH/TKMnROL 寄存器对中。并将当前 16-bit C/F 计数器的值写入到触控按键数据存储器对应位置。整个自动扫描操作会按上述方式从时隙 0 到时隙 3 有序地执行。时隙 3 按键扫描结束后，硬件会再次从触控按键数据存储区读取时隙 0 要扫描的按键对应的参考振荡器的电容值，并写入相应的 TKMnROH/TKMnROL 寄存器对中。16-bit C/F 计数器值也会被再次写回到时隙 3 扫描的按键对应的触控按键数据存储区位置。当所有按键都被扫描后，TKRCOV 位将被置高同时 TKBUSY 位被清零，意味着自动扫描工作已完成。若 TSC 位置高，仅时隙 0 有效并执行触控按键相关操作，而时隙 1~3 无效。

周期性自动扫描模式

除了在自动扫描模式中提到的动作，周期性自动扫描模式还提供了周期自动扫描和 C/F 计数器上限 / 下限阈值比较功能。设置 TKMOD1~TKMOD0 位为 10 或 11，可选择周期性自动扫描模式用来周期性地自动扫描模块按键。需注意的是，该模式通常在空闲模式下使用，用来监测触控按键状态，并尽量减少功耗。

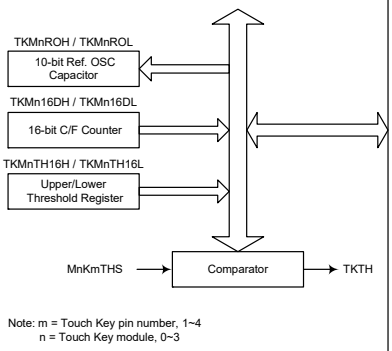
在周期性自动扫描模式时，触控按键扫描工作将周期地自动执行，扫描周期由 TKC2 寄存器的 ASMP1~ASMP0 位设定。触控按键扫描次数取决于 TB0 溢出周期和周期性自动扫描模式周期。每一次的自动扫描操作都会像自动扫描模式一样按指定的方式从时隙 0 到时隙 3 有序的执行。每个时隙选择扫描的按键对应的参考振荡器的电容值将会被自动从专用触控按键数据存储区指定位置读取出来，并写入相应的 TKMnROH/TKMnROL 寄存器对中。然而，仅在 TB0 溢出周期内最后一次扫描工作结束时，16-bit C/F 计数器的值才会被写回到专用触控按键数据存储区所有扫描的按键对应的位置。

另外，在进入时隙 0 开始扫描所选的按键之前，硬件会自动先从专用触控按键数据存储器指定位置读取时隙 0 要扫描的按键对应的上限 / 下限阈值，并写入相应的 TKMnTH16H/TKMnTH16L 寄存器对中。TKMnTH16H/TKMnTH16L 寄存器将在当前时隙结束时，由硬件自动从专用的触控按键数据存储器中载入相应的下一个 16-bit 上限 / 下限阈值。每个触控按键都有自己独立的上限 / 下限阈值比较器。在周期性自动扫描模式时，上限 / 下限阈值比较功能将自动使能。当 MnKmTHS=0，任何按键 C/F 计数器值小于下限阈值；或 MnKmTHS=1，按键 C/F 计数器值大于上限阈值，这表明触控按键状态发生了变化，MnKmTHF 标志位将被硬件置高，中断信号产生。应注意，因为在 16-bit C/F 计数器内容 TKMn16DH/TKMn16DL 与 TKMnTH16H/TKMnTH16L 值比较的同时 TKMnROH/TKMnROL 寄存器也在从专用的触控按键数据存储器中载入相应的下一个时隙电容值，所以触控按键模块 TKTH 中断发生时，将有 1-byte 数据写入到 TKMnROL 寄存器。

使用 TB0 计数器时钟进行周期性自动扫描可减少功耗，当 TB0ON 被清除时，TB0 计数器将停止，周期性自动扫描工作时间将会受到影响。

触控按键数据存储器

该系列单片机为触控按键自动扫描模式和周期性自动扫描模式提供了 3 个专用的数据存储器区域。第一个位于 Sector 5 用于存储触控按键模块 16-bit C/F 计数器值，第二个位于 Sector 6 用于存储触控按键模块参考振荡器内部电容值，最后一个位于 Sector 7 用于存储触控按键模块 16-bit 上限 / 下限阈值。

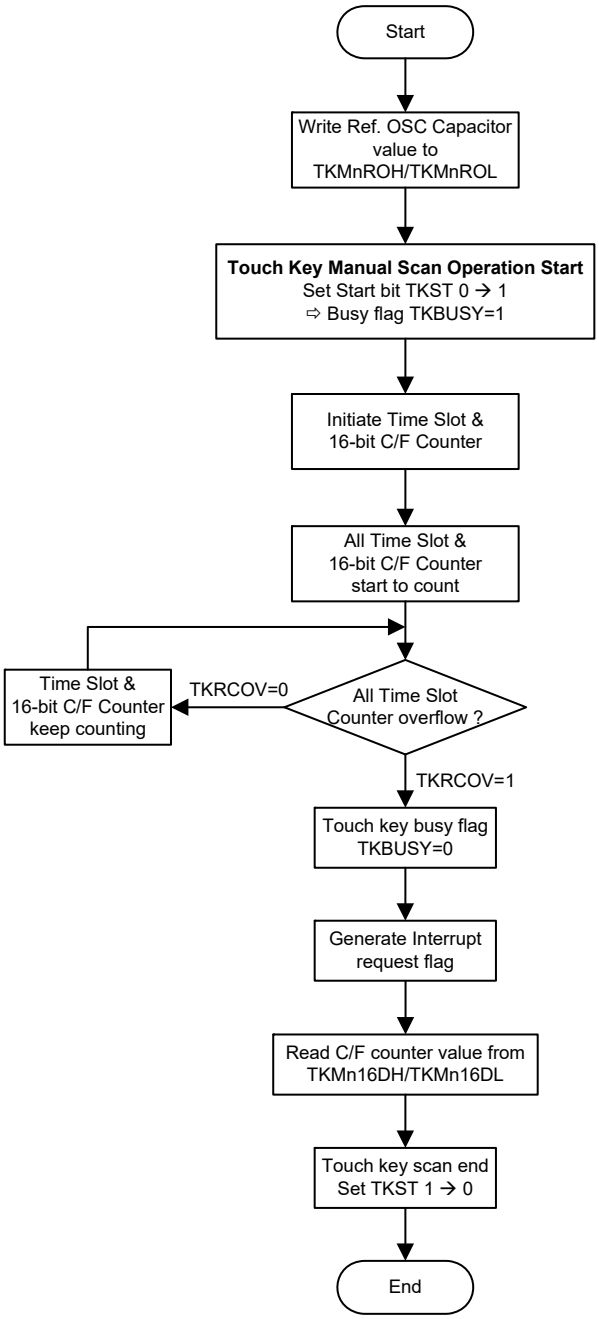


Note: m = Touch Key pin number, 1~4
n = Touch Key module, 0~3

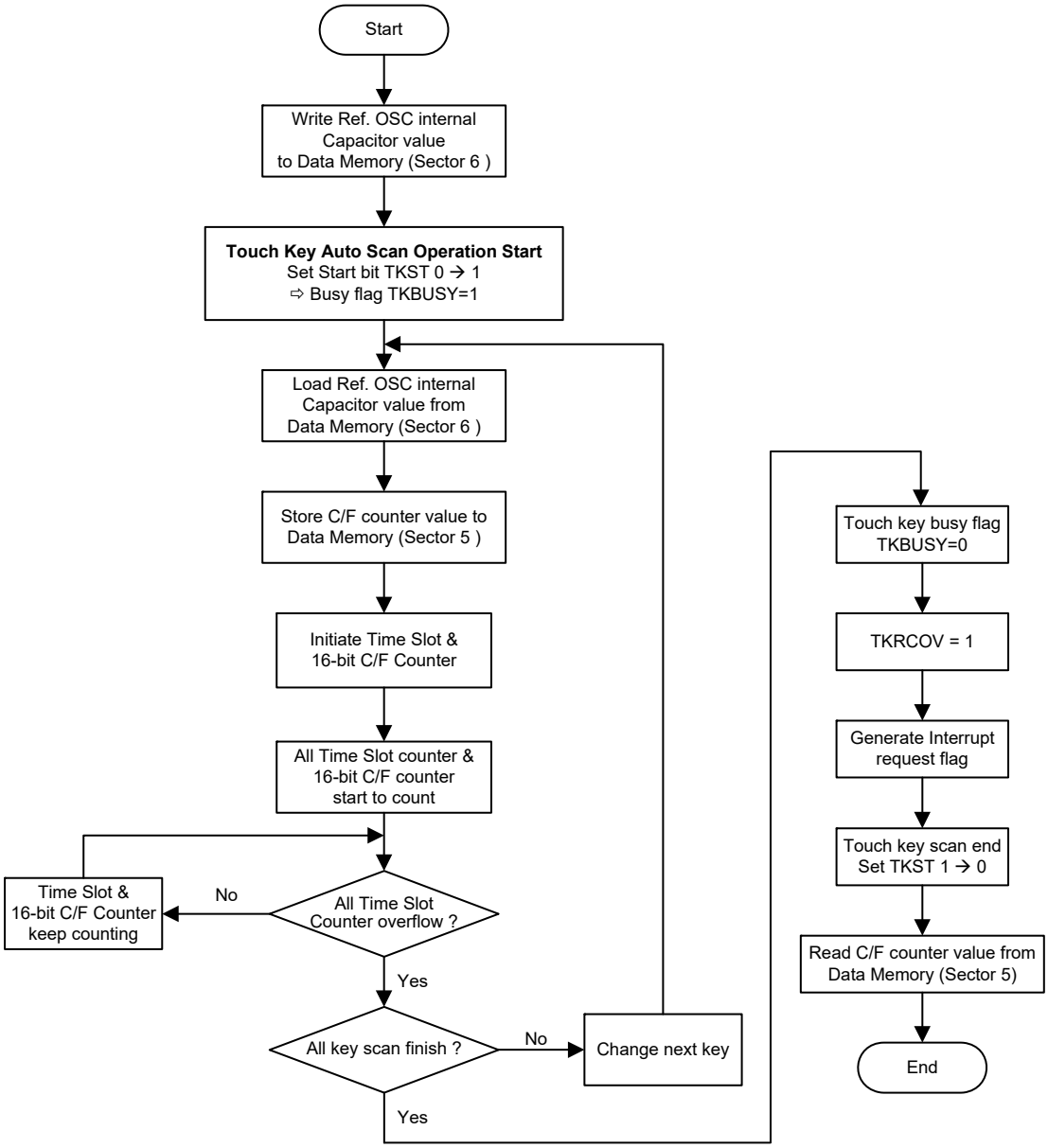
		16-bit C/F Counter Value (Sector 5)	Ref. OSC Capacitor Value (Sector 6)	Upper/Lower Threshold Value (Sector 7)
00H	Module 0	TKM016DL_K1	TKM0ROL_K1	TKM0TH16L_K1
01H		TKM016DH_K1	TKM0ROH_K1	TKM0TH16H_K1
02H		TKM016DL_K2	TKM0ROL_K2	TKM0TH16L_K2
		TKM016DH_K2	TKM0ROH_K2	TKM0TH16H_K2
		TKM016DL_K3	TKM0ROL_K3	TKM0TH16L_K3
		TKM016DH_K3	TKM0ROH_K3	TKM0TH16H_K3
		TKM016DL_K4	TKM0ROL_K4	TKM0TH16L_K4
		TKM016DH_K4	TKM0ROH_K4	TKM0TH16H_K4
	Module 1	TKM116DL_K1	TKM1ROL_K1	TKM1TH16L_K1
		TKM116DH_K1	TKM1ROH_K1	TKM1TH16H_K1
		TKM116DL_K2	TKM1ROL_K2	TKM1TH16L_K2
		TKM116DH_K2	TKM1ROH_K2	TKM1TH16H_K2
		TKM116DL_K3	TKM1ROL_K3	TKM1TH16L_K3
		TKM116DH_K3	TKM1ROH_K3	TKM1TH16H_K3
		TKM116DL_K4	TKM1ROL_K4	TKM1TH16L_K4
		TKM116DH_K4	TKM1ROH_K4	TKM1TH16H_K4
		⋮	⋮	⋮
	Module 3	TKM316DL_K1	TKM3ROL_K1	TKM3TH16L_K1
		TKM316DH_K1	TKM3ROH_K1	TKM3TH16H_K1
		TKM316DL_K2	TKM3ROL_K2	TKM3TH16L_K2
		TKM316DH_K2	TKM3ROH_K2	TKM3TH16H_K2
		TKM316DL_K3	TKM3ROL_K3	TKM3TH16L_K3
		TKM316DH_K3	TKM3ROH_K3	TKM3TH16H_K3
		TKM316DL_K4	TKM3ROL_K4	TKM3TH16L_K4
1FH		TKM316DH_K4	TKM3ROH_K4	TKM3TH16H_K4

触控按键数据存储器分布

触控按键扫描操作流程图



触控按键手动扫描模式流程图 – **TKMOD[1:0]=01, TSCS=0**



触控按键自动扫描模式流程图 – TKMOD[1:0]=00, TSCS=0

触控按键中断

触控按键有两个独立的中断，分别为触控按键 TKRCOV 中断和触控按键模块 TKTH 中断。在手动扫描模式中，当所有触控按键模块的时隙计数器都溢出时，或模块 0 时隙计数器溢出时，将会产生触控按键 TKRCOV 中断。在自动扫描模式中，当触控按键自动扫描操作结束时，触控按键 TKRCOV 中断请求标志位 TKRCOVF 将被置位。在周期性自动扫描模式中，只有在 TB0 溢出周期的最后一次扫描操作结束后，16-bit C/F 计数器内容被写入相应的触控按键数据存储器中，然后触控按键 TKRCOV 中断请求标志位 TKRCOVF 才会被置位。注意，此处提到的触控按键是指已被使能的触控按键。此时 16-bit C/F 计数器、16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器会自动清零。当

MnKmTHS=0，任何按键 C/F 计数器值小于下限阈值，或 MnKmTHS=1，按键 C/F 计数器值大于上限阈值，将产生触控按键模块 TKTH 中断。触控按键中断的详细说明见中断部分。

编程注意事项

相关寄存器设置后，将 TKST 位由低电平变为高电平会启动触控按键检测程序。此时所有相关的振荡器将使能并同步。在手动扫描模式中，当计数器溢出时，时隙计数器标志位 TKRCOV 将变为高电平，同时还会产生一个中断信号。在自动扫描模式中，若时隙计数器溢出但触控按键自动扫描操作未结束，TKRCOV 位将不会被置位。当触控按键自动扫描操作结束后，TKRCOV 位和触控按键 TKRCOV 中断请求标志位 TKRCOVF 将被置位。在周期性自动扫描模式中，自动扫描操作周期期间 TKRCOV 位为零。仅在 TB0 溢出周期内最后一次扫描操作结束时，16-bit C/F 计数器内容被写入相应的触控按键数据存储器中，然后 TKRCOV 才会将通过硬件电路被置高。当某一阈值比较条件发生时，阈值比较指示信号 TKTH 信号将变为高电平。当这种情况发生时，中断信号也会产生。由于 TKRCOV 标志位无法被自动清零，需通过应用程序将此位清零。

任何一个触控按键模块的 16-bit C/F 计数器溢出就会把 16-bit C/F 计数器溢出标志位 TKCFOV 置高。由于此标志位无法被自动清零，需通过应用程序将此位清零。16-bit 计数器溢出就会把其溢出标志位 TK16OV 置高。由于此标志位无法被自动清零，需通过应用程序将此位清零。

当外部触控按键的大小和布局确定时，其相关的电容值将决定感应振荡器的频率。

RF 发射器

RF 发射器是完全集成的发射器，能够使用频移键控 FSK 和开关键控 OOK 两种调制方式进行数据串流。它有两种工作模式，即突发模式和直接模式。RF 发射器工作在 315/433/868/915MHz 频段。

RF 发射器缩写注意事项

CP：充电泵

DFC：数字频率中心

FIFO：TX FIFO

MMD：多模分频器

PAD：功率放大驱动器

SX：合成器

TXD：来自 FIFO 或 DTXD 位的数据，取决于 DIR_EN 位

VCO：压控振荡器

XO：外部晶振输出

XCLK：RF 电路主要时钟。通过配置 XCLKD2 位控制，通过将 XCLK_EN 位置高来使能。

XCLK_MCU：单片机系统时钟。通过配置 XCLKD2 位控制，通过将寄存器 HXTC 的 HXTEN 位置高来使能。

RF 发射器控制寄存器

RF 发射器的所有操作都由一系列寄存器控制。这些寄存器控制 RF 所有功能，包括暂停控制、工作模式选择、时钟分频选择、FIFO 数据配置、调制器控制、小数 N 分频合成器控制、充电泵 (CP) 控制、多模分频器 (MMD) 控制、压控振荡器 (VCO) 控制、TX 功率微调、VCO DFC 校准、RF LDO 控制和 RF 外部晶振输出等。

寄存器名称	位							
	7	6	5	4	3	2	1	0
RF_PWR	—	—	—	—	—	—	—	RF_PDB
RF_OPER	—	—	FSK_EN	DIR_EN	—	—	—	TX_STROBE
RF_CLK1	XCLK_EN	XCLKINV	XCLKR_RDY	XCLKD2	—	—	—	RST_RF
RF_CLK2	DTR7	DTR6	DTR5	DTR4	DTR3	DTR2	DTR1	DTR0
RF_FIFO_CTRL1	FFDATA7	FFDATA6	FFDATA5	FFDATA4	FFDATA3	FFDATA2	FFDATA1	FFDATA0
RF_FIFO_CTRL2	FFLEN7	FFLEN6	FFLEN5	FFLEN4	FFLEN3	FFLEN2	FFLEN1	FFLEN0
RF_FIFO_CTRL3	RST_TX_FF	FFSA6	FFSA5	FFSA4	FFSA3	FFSA2	FFSA1	FFSA0
RF_FIFO_CTRL4	DTXD	—	—	—	TXFFLT	FFMG_EN	FFMG1	FFMG0
RF_MOD1	FDEV7	FDEV6	FDEV5	FDEV4	FDEV3	FDEV2	FDEV1	FDEV0
RF_MOD2	—	—	—	—	—	FDEV10	FDEV9	FDEV8
RF_MOD4	D7	D6	D5	OOKDT_TS2	OOKDT_TS1	OOKDT_TS0	OOKDT_POR	OOKDT_EN
RF_OPMOD	—	—	—	—	—	ACAL_EN	TX_EN	SX_EN
RF_SX1	—	DN6	DN5	DN4	DN3	DN2	DN1	DN0
RF_SX2	DK7	DK6	DK5	DK4	DK3	DK2	DK1	DK0
RF_SX3	DK15	DK14	DK13	DK12	DK11	DK10	DK9	DK8
RF_SX4	—	—	—	—	DK19	DK18	DK17	DK16
RF_CP3	DLY_SYN2	DLY_SYN1	DLY_SYN0	D4	D3	D2	D1	D0
RF_OD1	D7	D6	D5	D4	D3	D2	D1	D0
RF_VCO1	VCO_SWHB	D6	D5	D4	D3	D2	D1	D0
RF_TX1	DLY_PAD2	DLY_PAD1	DLY_PAD0	D4	CT_PARD2	CT_PARD1	CT_PARD0	PAD_EN
RF_TX2	D7	D6	D5	D4	D3	D2	D1	D0
RF_DFC_CAL	CT_MMDLDO1	CT_MMDLDO0	—	D4	D3	D2	D1	D0
RF_LDO	D7	D6	D5	D4	D3	D2	D1	D0
RF_XO1	—	—	—	XO_TRIM4	XO_TRIM3	XO_TRIM2	XO_TRIM1	XO_TRIM0

RF 发射器控制寄存器列表

大多数 RF 发射器控制寄存器都在此章节描述，但是一些各别寄存器分别在其它章节中描述。

- **RF_PWR 寄存器 – RF 暂停控制寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	RF_PDB
R/W	—	—	—	—	—	—	—	R/W
POR	—	—	—	—	—	—	—	0

Bit 7~1　未定义，读为“0”

Bit 0　**RF_PDB**：RF 暂停控制
0：RF 暂停模式
1：RF 有效模式
该位决定 RF 电路处于有效模式还是暂停模式。当单片机发生上电复位、LVR 复位或正常运行情况下发生 WDT 溢出，又或者通过将 PWRC 寄存器的 PWDN 位置 1 且执行 HALT 指令使单片机进入深度休眠模式时，此位都将被清零。

- **RF_CLK1 寄存器 – RF 时钟控制寄存器 1**

Bit	7	6	5	4	3	2	1	0
Name	XCLK_EN	XCLKINV	XCLKR_RDY	XCLKD2	—	—	—	RST_RF
R/W	R/W	R/W	R	R/W	—	—	—	R/W
POR	1	0	0	0	—	—	—	0

Bit 7　**XCLK_EN**：时钟自动选通用于 RF 内部数字电路
0：除能
1：使能
写数据到 FIFO 需要先使能 XCLK。XCLK_EN 位应该先清零，才能通过将 RF_PDB 位清零使 RF 电路进入暂停模式。这能避免 XCLK 时钟发生毛刺，从而保证不会因为时钟丢失而影响 XCLK 域寄存器。

Bit 6　**XCLKINV**：XCLK 时钟反相控制
0：XCLK 时钟同相
1：XCLK 时钟反相

Bit 5　**XCLKR_RDY**：XCLK 就绪标志位
0：XCLK 未就绪
1：XCLK 就绪
当 HXTEN=0 且 RF_PDB=1，即 MCU 采用 f_{HIRC} 时钟工作时读取该位。该位为只读位，用于指示 XCLK 时钟去抖是否完成并准备好工作。

Bit 4　**XCLKD2**：XCLK 时钟二分频控制
0：XCLK 时钟未二分频
1：XCLK 时钟二分频
建议将此位清零。

Bit 3~1　未定义，读为“0”

Bit 0　**RST_RF**：RF 控制寄存器复位控制
0：RF 控制寄存器已完成自动清零
1：RF 控制寄存器复位
当该位置 1 时，除了 RF_PWR 寄存器，其它所有 RF 相关的寄存器都将复位。这需要一个指令周期来完成自动清零使能动作。不要在两个连续的指令中设置 RST_RF 位为 1。

- **RF_CLK2 寄存器 – RF 时钟控制寄存器 2**

Bit	7	6	5	4	3	2	1	0
Name	DTR7	DTR6	DTR5	DTR4	DTR3	DTR2	DTR1	DTR0
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 **DTR7~DTR0**：RF 数据传输速率设置
RF 数据传输速率 = 100kHz/(DTR[7:0]+1)

- **RF_CP3 寄存器 – RF CP 控制寄存器 3**

Bit	7	6	5	4	3	2	1	0
Name	DLY_SYN2	DLY_SYN1	DLY_SYN0	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	0	0	1	0	1	0

Bit 7~5 **DLY_SYN2~DLY_SYN0**：合成器延迟就绪时间
000：16μs
001：20μs
010：24μs
011：28μs
100：32μs
101：36μs
110：40μs
111：100μs

Bit 4~0 **D4~D0**：保留位，请填入 11100b

- **RF_OD1 寄存器 – RF MMD 和 OD 控制寄存器 1**

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	1	0	0

Bit 7~0 **D7~D0**：多模分频器和输出分频

该寄存器用来控制 RF 电路的多模分频器和输出分频。需注意的是，针对不同的 RF 频带应用，该寄存器需写入不同设置值。建议的设置值详见下表。

RF 频带	315MHz	433MHz	868/915MHz
RF_OD1 设置值	A8H	A4H	00H

- **RF_VCO1 寄存器 – RF VCO 控制寄存器 1**

Bit	7	6	5	4	3	2	1	0
Name	VCO_SWHB	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	1	0	0	0	0

Bit 7 **VCO_SWHB**：VCO 2.5GHz 频带切换控制
0：其它频带
1：315MHz 频带
该位用来选择 VCO 315MHz 频带。如果该位置 1，则选择 VCO 315MHz。当该

位清 0 时，则选择除了 315MHz 外的其它 VCO 频带。

Bit 6~0 **D6~D0**：频带控制

需注意的是，针对不同的 RF 频带应用，该寄存器需写入不同设置值。建议的设置值详见下表。

RF 频带	315MHz	433MHz	868/915MHz
RF_VCO1 设置值	A8H	38H	38H

- **RF_TX1 寄存器 – RF TX 控制寄存器 1**

Bit	7	6	5	4	3	2	1	0
Name	DLY_PAD2	DLY_PAD1	DLY_PAD0	D4	CT_PARD2	CT_PARD1	CT_PARD0	PAD_EN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	0	0	0	1	0	0	0

Bit 7~5 **DLY_PAD2~DLY_PAD0**：TX PA 驱动器稳定时间 (仅用于 FSK 模式)
000：16μs
001：20μs
010：24μs
011：28μs
100：32μs
101：36μs
110：40μs
111：50μs

Bit 4 **D4**：保留位

Bit 3~1 **CT_PARD2~CT_PARD0**：PA Ramp-up/down 时间调整
000：5μs
001：9μs
010：13μs
011：17μs
100：21μs
101：25μs
110：29μs
111：33μs

Bit 0 **PAD_EN**：TX PA 驱动器使能控制
0：除能
1：使能

建议的设置值详见下表。

RF 调制模式	OOK	FSK
RF_TX1 设置值	86H	84H

- **RF_TX2 寄存器 – RF TX 控制寄存器 2**

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	1	0	0	0	0	0	1	0

Bit 7~0 **D7~D0**：保留位

建议的设置值详见下表。

RF 频带		315MHz	433MHz	868MHz/915MHz
RF_TX2 设置值	0dBm	3AH	3AH	3AH
	10dBm	8AH	8AH	8AH
	13dBm	B2H	AAH	BAH

- **RF_DFC_CAL 寄存器 – RF VCO DFC 校准控制寄存器**

Bit	7	6	5	4	3	2	1	0
Name	CT_MMDLDO1	CT_MMDLDO0	—	D4	D3	D2	D1	D0
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 7~6 **CT_MMDLDO1~CT_MMDLDO0**：MMD LDO 电压设置
00：1.35V
01：1.5V (建议设定值)
10：1.65V
11：1.8V

Bit 5 未定义，读为“0”

Bit 4~0 **D4~D0**：保留位，请填入 10000b

- **RF_LDO 寄存器 – RF LDO 控制寄存器**

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	1	1	0	0	0

Bit 7~0 **D7~D0**：RF 合成器和 VCO LDO 功能控制

该寄存器用来控制 RF 合成器和 VCO 电路的 LDO 整体功能。需注意的是，针对不同的 RF 频带应用，该寄存器需写入不同设置值。建议的设置值详见下表。

RF 频带	315MHz	433MHz	868/915MHz
RF_LDO 设置值	64H	74H	74H

- **RF_XO1 寄存器 – RF XO 控制寄存器 1**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	XO_TRIM4	XO_TRIM3	XO_TRIM2	XO_TRIM1	XO_TRIM0
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	1	0	1	0	1

Bit 7~5 未定义，读为“0”

Bit 4~0 **XO_TRIM4~XO_TRIM0**：晶体振荡器内部电容负载微调
每阶电容变化 = 0.15pF，XO_TRIM[4:0]=0~31
总电容负载 ≈ XO_TRIM[4:0]×0.15，单位：pF

调制模式和工作模式选择

调制模式

该系列单片机有两种 RF 调制模式，即 FSK 模式和 OOK 模式，可通过 FSK_EN 位选择。

在 OOK 调制模式中，RFOUT 引脚将输出 RF 载波信号，其频率 f_C 可通过通道代码 DN[6:0] 和 DK[19:0] 选择。RF 载波信号的开启和关闭由以所需的数据速率来传输的数据来控制。

在 FSK 调制模式中，RFOUT 引脚将输出 RF 信号，数据“1”的频率为 (f_C+f_{DEV})，而数据“0”的频率为 (f_C-f_{DEV})。RF 载波信号的开启和关闭由以确定的数据速率来传输的数据来控制。

工作模式

该系列单片机有两种 RF 工作模式，即突发模式和直接模式，可通过 DIR_EN 位选择。

在突发模式中，被传送的数据来自 FIFO，且通过状态机控制 RF 模块。用户只需将数据写入 FIFO，且通过将 TX_STROBE 置高来启动传送，直至传送完成。这是一个易用模式。

在直接模式中，被传送的数据由 RF_FIFO_CTRL4 寄存器中的 DTXD 位配置。RF 功能模块的启动和关闭时序可通过特定的程序序列控制，下文将一一介绍。该模式拥有较大的灵活性，但需要用户更多的关注模块控制和被传送的数据。

- **RF_OPER 寄存器 – RF 工作控制寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	FSK_EN	DIR_EN	—	—	—	TX_STROBE
R/W	—	—	R/W	R/W	—	—	—	R/W
POR	—	—	0	0	—	—	—	0

Bit 7~6　未定义，读为“0”

Bit 5　**FSK_EN**：RF 输出调制模式选择
0：OOK 模式，PAD 使能通过 TXD 直接控制
1：FSK 模式，PAD 强制使能，且 TXD 调制 TX 频率

Bit 4　**DIR_EN**：RF 输出工作模式选择
0：突发模式，可从 FIFO 读取 TX 数据，通过状态机控制 RF。若 FIFO 传送完成，RF TX 将进入待机模式
1：直接模式，TX 数据来自 RF_FIFO_CTRL4 寄存器的 DTXD 位，可通过 SX_EN 位 (第一层使能) 和 TX_EN 位 (第二层使能) 控制 RF 状态

Bit 3~1　未定义，读为“0”

Bit 0　**TX_STROBE**：TX Strobe 启动 RF 突发模式数据传输
0：TX 数据包已经发送完毕
1：启动 TX 传送
若该位置高，则自动启动 TX 传送步骤，此时所有寄存器设置都不能被改变。当一个 TX 数据包发送完毕时，该位将自动清零，此时会产生 BMTCF 中断请求。在此位返回 0 后，用户可启动下一次传送。通过对该位写入 0，用户可强制终止 TX 传送。

- **RF_MOD1 寄存器 – RF 调制器控制寄存器 1**

Bit	7	6	5	4	3	2	1	0
Name	FDEV7	FDEV6	FDEV5	FDEV4	FDEV3	FDEV2	FDEV1	FDEV0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	0	0	1	0	0	0	0

Bit 7~0 **FDEV7~FDEV0**：设置 FSK 模式下的频率偏差
这些位将和 RF_MOD2 寄存器中的 FDEV10~FDEV9 位一起用来设置 FSK 模式下的频率偏差。

- **RF_MOD2 寄存器 – RF 调制器控制寄存器 2**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	FDEV10	FDEV9	FDEV8
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	1

Bit 7~3 未定义，读为“0”

Bit 2~0 **FDEV10~FDEV8**：设置 FSK 模式下的频率偏差
这些位将和 RF_MOD1 寄存器中的 FDEV7~FDEV0 位一起用来设置 FSK 模式下的频率偏差。
$FDEV[10:0]=DEC2HEX\ (((2^{17}-1)/f_{XTAL})\times f_D)$
例如，若 f_{XTAL}=16MHz，f_D=50kHz，则 FDEV[10:0]=199H。
注：使用在 868/915MHz 频带时，若频率偏差为 150kHz，需注意安规特性。

- **RF_MOD4 寄存器 – RF 调制器控制寄存器 4**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	OOKDT_TS2	OOKDT_TS1	OOKDT_TS0	OOKDT_POR	OOKDT_EN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	1	1	1	0

Bit 7~5 **D7~D5**：保留位

Bit 4~2 **OOKDT_TS2~OOKDT_TS0**：OOK 占空比调谐时间选择
000：1μs
001：3μs
010：5μs
011：10μs
100：15μs
101：20μs
110：25μs
111：30μs

Bit 1 **OOKDT_POR**：OOK 占空比调谐极性
0：延长 0 个占空比
1：延长 1 个占空比

Bit 0 **OOKDT_EN**：OOK 占空比调谐使能控制
0：除能
1：使能

建议的设置值详见下表。

RF 调制模式	OOK	FSK
RF_MOD4 设置值	0DH	15H

- **RF_OPMOD 寄存器 – RF 工作模式控制寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	ACAL_EN	TX_EN	SX_EN
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3　未定义，读为“0”

Bit 2　**ACAL_EN**：自动校准使能控制
0：除能
1：使能
通过应用程序置位或复位此位来激活或终止自动校准功能。当完成自动校准时，该位会被硬件自动清零。

Bit 1　**TX_EN**：发射器控制 (仅在直接模式下有效)
0：除能
1：使能
该位仅用于直接模式，不可用于突发模式。

Bit 0　**SX_EN**：合成器 PLL 开启控制 (仅在直接模式下有效)
0：除能
1：使能
该位仅用于直接模式，不可用于突发模式。

突发模式下的 TX FIFO 模式

在突发模式中，要传送的数据来自 FIFO，由单片机预先写入。有三种 FIFO 模式，分别为简易型 FIFO 模式、模块型 FIFO 模式和扩展型 FIFO 模式，可用于支持不同应用。要在突发模式下使用 FIFO，可通过设置 RF_FIFO_CTRL3 寄存器的 RST_TX_FF 位为 1 来复位 FIFO 指针和缓冲器。此后，FIFO 处于与上电复位相同的初始状态。

简易型 FIFO 模式

此 FIFO 模式用于普通应用。数据长度不应超过 128 字节。为了采用简易型 FIFO 模式，单片机必须通过访问 RF_FIFO_CTRL1 寄存器的 FFDATA[7:0] 位将要传送的数据写入 FIFO。传送到发射器的传输序列须先入先出，且每个字节都是最高位先出。用户应先确定传输数据包格式，例如前导码、识别码和数据包编码，其中数据包编码包括 FEC (前向纠错)、CRC (循环冗余校验) 和数据白化等。当 FIFO 被完全写满后，将 RF_FIFO_CTRL3 寄存器的 FFSA[6:0] 位全清为 0，并通过设置 RF_FIFO_CTRL2 寄存器的 FFLEN[7:0] 位配置所需的传输长度 (单位：字节)。然后通过配置 FSK_EN、DIR_EN 和 TX_STROBE 位来启动传送。完成当前的传送后，数据将被保存在 FIFO 以等待下一次传送。

模块型 FIFO 模式

模块型 FIFO 模式用于支持多键编码应用。用户应预先将所有按键编码写入 FIFO。当按下按键时，单片机将检测到按键，并通过设置 RF_FIFO_CTRL3 寄存器的 FFSA[6:0] 位为目标按键码起始地址，然后通过设置 RF_FIFO_CTRL2 寄存器的 FFLEN[7:0] 位来表明按键码长度，最后通过设置 RF_OPER 寄存器的 TX_STROBE 位来启动传送。最大 FIFO 长度也应限制为 128 字节。

扩展型 FIFO 模式

扩展型 FIFO 模式用于长度可达 256 字节的大数据包。FIFO 的物理长度为 128 字节。为了在一个数据包里扩展可用的传输长度，则在单片机和 FIFO 之间需要一个握手机制。

通过设置 RF_FIFO_CTRL4 寄存器的 FFMG[1:0] 位来决定 FIFO 数据长度阈值，通过将 FFMG_EN 位置 1 使能阈值检测功能以提醒单片机。当收到提示信号时，单片机应尽可能快地将数据写入 FIFO，以避免 FIFO 数据长度为 0 从而导致传输终止。

- **RF_FIFO_CTRL1 寄存器 – RF FIFO 控制寄存器 1**

Bit	7	6	5	4	3	2	1	0
Name	FFDATA7	FFDATA6	FFDATA5	FFDATA4	FFDATA3	FFDATA2	FFDATA1	FFDATA0
R/W	W	W	W	W	W	W	W	W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **FFDATA7~FFDATA0**：FIFO 数据端口
写数据到该端口，即将数据写入 FIFO。从该端口读取数据，即从 FIFO 顶部读取数据。

- **RF_FIFO_CTRL2 寄存器 – RF FIFO 控制寄存器 2**

Bit	7	6	5	4	3	2	1	0
Name	FFLEN7	FFLEN6	FFLEN5	FFLEN4	FFLEN3	FFLEN2	FFLEN1	FFLEN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	1	1	1	1	1

Bit 7~0 **FFLEN7~FFLEN0**：TX 数据长度 (仅用于突发模式)
被传送的数据字节数为 FFLEN[7:0]+1。在简易型 FIFO 模式和模块型 FIFO 模式中，FFLEN[7:0] 应限制在 7Fh。用户设置寄存器时不可大于该值。

- **RF_FIFO_CTRL3 寄存器 – RF FIFO 控制寄存器 3**

Bit	7	6	5	4	3	2	1	0
Name	RST_TX_FF	FFSA6	FFSA5	FFSA4	FFSA3	FFSA2	FFSA1	FFSA0
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 **RST_TX_FF**：TX FIFO 复位控制
0：不复位 TX FIFO 到初始状态或复位动作已完成
1：复位 TX FIFO 到初始状态
该位置 1 时将复位 TX FIFO 到初始状态。复位动作完成后，该位将自动清零。

Bit 6~0 **FFSA6~FFSA0**：FIFO 发送数据的起始地址 – 仅用于模块型 FIFO 模式

- **RF_FIFO_CTRL4 寄存器 – RF FIFO 控制寄存器 4**

Bit	7	6	5	4	3	2	1	0
Name	DTXD	—	—	—	TXFFLT	FFMG_EN	FFMG1	FFMG0
R/W	R/W	—	—	—	R	R/W	R/W	R/W
POR	0	—	—	—	0	0	0	1

Bit 7 **DTXD**：直接模式 TX 数据设置

Bit 6~4 未定义，读为“0”

Bit 3 **TXFFLT**：TX FIFO 长度状态标志 (只读)
0：TX FIFO 长度大于 FFMG[1:0] 指定的长度阈值
1：TX FIFO 长度小于或等于 FFMG[1:0] 指定的长度阈值
若 FFMG_EN 位为 0，TXFFLT 位将始终保持为 0。若 FFMG_EN 位为 1，TXFFLT 位将表明 TX FIFO 的长度状态。当该位被硬件置高时，表明 TX FIFO 长度小于或等于 FFMG[1:0] 指定的长度阈值，此时，将产生一个 FFMGF 中断

请求。

Bit2 **FFMG_EN**：TX FIFO 长度阈值检测功能使能控制
0：除能
1：使能

Bit 1~0 **FFMG1~FFMG0**：TX FIFO 长度阈值 – 表示 FIFO 中剩余的数据字节数
00：4 个字节
01：8 个字节
10：16 个字节
11：32 个字节

RF 通道设置

RF 通道可通过四个寄存器 RF_SX1~RF_SX4 设置。

- **RF_SX1 寄存器 – RF 小数 N 分频合成器控制寄存器 1**

Bit	7	6	5	4	3	2	1	0
Name	—	DN6	DN5	DN4	DN3	DN2	DN1	DN0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	1	1	0	1	1	0

Bit 7 未定义，读为“0”

Bit 6~0 **DN6~DN0**：MMD 被除数的整数部分，支持 32~127
由于 MMD 数据来自 delta-sigma 调制器，DN 支持范围将受影响。实际范围将为 (32+3)~(127-4)，即 35~123。
设置初始值，即 XO=16MHz，TX 频带 = 433.92MHz：
例如，若晶振 = XO=16MHz，且 RF_OD1=04H；
VCO 频率 = TX 频率 × 4= 433.92MHz×4=1735.68MHz；
XO×(DN+DK/2^{20})=16MHz×(DN+DK/2^{20})=VCO 频率 / 2=1735.68MHz/2=867.84MHz；
(DN+DK/2^{20})=54.24，因此 DN[6:0]=36H=54，DK[19:0]=3D70AH=251658。

- **RF_SX2 寄存器 – RF 小数 N 分频合成器控制寄存器 2**

Bit	7	6	5	4	3	2	1	0
Name	DK7	DK6	DK5	DK4	DK3	DK2	DK1	DK0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	1	0	1	0

Bit 7~0 **DK7~DK0**：MMD 被除数的 20-bit 小数部分的低字节
设置初始值，实现 XO=16MHz，且 TX 频带 = 433.92MHz。

- **RF_SX3 寄存器 – RF 小数 N 分频合成器控制寄存器 3**

Bit	7	6	5	4	3	2	1	0
Name	DK15	DK14	DK13	DK12	DK11	DK10	DK9	DK8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	0	1	0	1	1	1

Bit 7~0 **DK15~DK8**：MMD 被除数的 20-bit 小数部分的中间字节
设置初始值，实现 XO=16MHz，且 TX 频带 = 433.92MHz。

- **RF_SX4 寄存器 – RF 小数 N 分频合成器控制寄存器 4**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	DK19	DK18	DK17	DK16
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	1	1

Bit 7~4　未定义，读为“0”

Bit 3~0　**DK19~DK16**：MMD 被除数的 20-bit 小数部分的高字节
设置初始值，实现 XO=16MHz，且 TX 频带 = 433.92MHz。

RF 通道设置说明 (VCO 工作频率：1.7GHz~1.9GHz, 2.5GHz)

RF_OD1 寄存器必须按照下面两个规则配置：

1. VCO 工作频率是否在规定范围内。
2. DN 和 DK 计算值在寄存器 RF_SX1~RF_SX4 的可用范围内。

当晶振频率 = XO=16MHz 时，以下为五个频率配置范例：

- 315MHz
  RF_OD1=08H
  VCO 频率 = TX 频率 × 8=315MHz×8=2520MHz
  XO×(DN+DK/2^{20})=16MHz×(DN+DK/2^{20})=VCO 频率 / 2=2520MHz/2=1260MHz
  (DN+DK/2^{20})=78.75，因此 DN[6:0]=4EH=78，DK[19:0]=C0000H=786432
  注：当 TX 频率为 315MHz 时，应通过应用程序将 VCO_SWHB 位置高，将 VCO 切换到 2.5GHz 频带。
- 433MHz
  RF_OD1=04H
  VCO 频率 = TX 频率 × 4=433MHz×4=1732MHz
  XO×(DN+DK/2^{20})=16MHz×(DN+DK/2^{20})=VCO 频率 / 2=1732MHz/2=866MHz
  (DN+DK/2^{20})=54.125，因此 DN[6:0]=36H=54，DK[19:0]=20000H=131072
- 433.92MHz (默认设置)
  RF_OD1=04H
  VCO 频率 = TX 频率 × 4=433.92MHz×4=1735.68MHz
  XO×(DN+DK/2^{20})=16MHz×(DN+DK/2^{20})=VCO 频率 /2=1735.68MHz/2=867.84MHz
  (DN+DK/2^{20})=54.24，因此 DN[6:0]=36H=54，DK[19:0]=3D70AH=251658
- 868MHz
  RF_OD1=00H
  VCO 频率 = TX 频率 × 2=868MHz×2=1736MHz
  XO×(DN+DK/2^{20})=16MHz×(DN+DK/2^{20})=VCO 频率 / 2=1736MHz/2=868MHz
  (DN+DK/2^{20})=54.25，因此 DN[6:0]=36H=54，DK[19:0]=40000H=262144
- 915MHz
  RF_OD1=00H
  VCO 频率 = TX 频率 × 2=915MHz×2=1830MHz
  XO×(DN+DK/2^{20})=16MHz×(DN+DK/2^{20})=VCO 频率 / 2=1830MHz/2=915MHz
  (DN+DK/2^{20})=57.1875，因此 DN[6:0]=39H=57，DK[19:0]=30000H=196608

软件编程指南

为了帮助用户实现所需的传送，此章节将介绍突发模式和直接模式下的参考设置步骤。这两个模式的差异在于被传送数据的产生、RF 模块开启和关闭控制，更多细节详见下文描述。

突发模式

1. 根据所选的晶体振荡器正确设置 XCLKD2 位。
2. 通过将 RF_PDB 位置 1，RF 电路将采用 XCLK 时钟开始工作。
3. 若用户选择 HXT 作为系统时钟，需检查 HXTF 位。若此位为 1，表明单片机时钟已经准备完毕，单片机可采用 XCLK_MCU 时钟作为它的主要时钟。
4. 设置 RF 相关配置寄存器。
5. 若第一次上电或工作环境变化很大，通过设置 RF_OPMOD 寄存器的 ACAL_EN 位来启动自动校准流程，轮询该位直至校准结束。
6. 对 FIFO 写入数据。通过 FFLEN[7:0] 位来设置所需的传输字节长度。通过 DTR[7:0] 位来设置数据传输速率。通过 FDEV[10:0] 位设置 FSK 模式下的频率偏差。
7. 通过设置 DIR_EN 位为 0 来选择突发模式，设置 FSK_EN 位为 0，则选择 OOK 调制方式，而设置 FSK_EN 位为 1，则选择 FSK 调制方式，设置 TX_STROBE 位为 1，启动传输。然后 FIFO 的数据将通过 FSK 或 OOK 调制方式自动转移到 RFOUT 引脚。
8. 轮询 TX_STROBE 位直至该位被硬件清零。当 TX_STROBE 位为 1 时，不允许对 RF 配置寄存器进行任何写入访问。
9. 若用户想要再发送先前的数据，只需检查 TX_STROBE 位。当该位为低，再设置为 1 将启动下一次传输。
10. 若 TX_STROBE 位为高，用户想强制终止此次传输，只需将 TX_STROBE 位设为低，而等待至少 1μs 后，状态机将关闭功率放大驱动器。用户可再次设置 TX_STROBE 位来启动新的传输。
11. 将 RF_PDB 清零即可关闭 RF 电路。

直接模式

1. 根据所选的晶体振荡器正确设置 XCLKD2 位。
2. 通过将 RF_PDB 位置 1，RF 电路将采用 XCLK 时钟开始工作。
3. 若用户选择 HXT 作为系统时钟，需检查 HXTF 位。若此位为 1，表明单片机时钟已经准备完毕，单片机可采用 XCLK_MCU 时钟作为它的主要时钟。
4. 设置 RF 相关配置寄存器。
5. 若第一次上电或工作环境变化很大，通过设置 RF_OPMOD 寄存器的 ACAL_EN 位来启动自动校准流程，轮询该位直至校准结束。
6. 写入要传送的第一个数据位至 DTXD 位。通过 FDEV[10:0] 位来设置 FSK 模式下的频率偏差。
7. 通过将 DIR_EN 位设为 1 来选择直接模式，并执行一段延迟时间 (约 20μs) 以等待相关电路就绪，设置 FSK_EN 位为 0，则选择 OOK 调制方式，而设置 FSK_EN 位为 1，则选择 FSK 调制方式。
8. 通过将 SX_EN 置位来使能所有合成器模块功能，执行一段延迟时间 (约 20μs) 以等待合成器稳定。
9. 通过将 TX_EN 置位来使能功率放大器模块功能，然后开始通过 RFOUT 引脚传输数据。
10. 按照所需数据传输速率持续更新 DTXD 位直至所有数据传送完成。
11. 将 TX_EN 位设为 0，延迟 20μs，然后也将 SX_EN 位为设 0 并延迟 20μs。
12. 将 RF_PDB 位清零来关闭 RF 电路。

注：在直接模式中，当完成数据传送时，为了使 RF 电路进入暂停模式以减少功耗，用户必须按照步骤 11 和步骤 12 所描述的正确顺序配置相关位，否则会产生不必要的待机电流。

低电压检测 – LVD

此系列单片机都具有低电压检测功能，即 LVD。该功能使能后用于监测电源电压 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 7 个固定电压中的一个参考点。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。ENLVD 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。低电压检测会有一定的功耗，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	ENLVD	D3	VLVD2	VLVD1	VLVD0
R/W	—	—	R	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	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

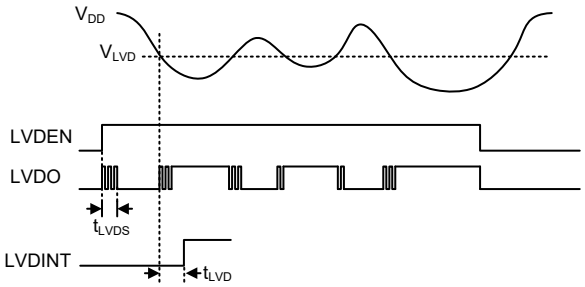
Bit 4　**ENLVD**：低电压检测控制位
0：除能
1：使能

Bit 3　**D3**：保留位

Bit 2~0　**VLVD2~VLVD0**：选择 LVD 电压位
000：2.4V
001：2.0V
010：2.2V
011：2.4V
100：2.7V
101：2.9V
110：3.0V
111：3.3V

LVD 操作

通过比较电源电压 V_{DD} 与存储在 LVDC 寄存器中的预置电压值的结果，低电压检测功能工作。其设置的范围为 2.0V~3.3V。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。当单片机进入休眠或深度休眠模式时，即使 ENLVD 位为高，低电压检测器也会自动除能。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_{LVD} 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请求，单片机将从空闲模式中被唤醒。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。此系列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 和时基等产生。

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专用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 MFI0~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 引脚	INTnE	INTnF	对于 BC66F2235，n=1 对于BC66F2245/BC66F2255，n=0~1
时基	TBnE	TBnF	n=0~1
多功能中断	MFnE	MFnF	n=0~2
LVD	LVE	LVF	—
RF TX FIFO 长度阈值检测	FFMGE	FFMGF	—
RF 突发模式传送完成	BMTCE	BMTCF	—
触控按键 TKRCOV	TKRCOVE	TKRCOVF	—

功能	使能位	请求标志	注释
触控按键模块 TKTH	TKTHE	TKTHF	—
CTM	CTMnPE	CTMnPF	n=0~1
	CTMnAE	CTMnAF	
PTM	PTMnPE	PTMnPF	n=0
	PTMnAE	PTMnAF	

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	—	TB0F	INT1F	INT0F	TB0E	INT1E	INT0E	EMI
INTC1	MF2F	MF1F	MF0F	TB1F	MF2E	MF1E	MF0E	TB1E
INTC2	ADF	USIMF	TKTHF	TKRCOVF	ADE	USIME	TKTHE	TKRCOVE
MFI0	CTM1AF	CTM1PF	CTM0AF	CTM0PF	CTM1AE	CTM1PE	CTM0AE	CTM0PE
MFI1	—	—	PTM0AF	PTM0PF	—	—	PTM0AE	PTM0PE
MFI2	—	LVF	FFMGF	BMTCF	—	LVE	FFMGE	BMTCE

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 引脚中断边沿控制位
00：除能
01：上升沿
10：下降沿
11：双沿

Bit 1~0 **INT0S1~INT0S0**：INT0 引脚中断边沿控制位
00：除能
01：上升沿
10：下降沿
11：双沿
注：对于 BC66F2235，这两位为保留位且需固定为“00”。

- **INTC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	TB0F	INT1F	INT0F	TB0E	INT1E	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 **TB0F**：时基 0 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 **INT0F**：INT0 中断请求标志位
0：无请求
1：中断请求
注：对于 BC66F2235，该位为保留位且需固定为“0”。

Bit 3 **TB0E**：时基 0 中断控制位
0：除能
1：使能

Bit 2 **INT1E**：INT1 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
注：对于 BC66F2235，该位为保留位且需固定为“0”。

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	MF2F	MF1F	MF0F	TB1F	MF2E	MF1E	MF0E	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 **MF2F**：多功能中断 2 请求标志位
0：无请求
1：中断请求

Bit 6 **MF1F**：多功能中断 1 请求标志位
0：无请求
1：中断请求

Bit 5 **MF0F**：多功能中断 0 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 **MF2E**：多功能中断 2 控制位
0：除能
1：使能

Bit 2 **MF1E**：多功能中断 1 控制位
0：除能
1：使能

Bit 1 **MF0E**：多功能中断 0 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

- **INTC2 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	ADF	USIMF	TKTHF	TKRCOVF	ADE	USIME	TKTHE	TKRCOVE
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 **ADF**：A/D 转换器中断请求标志位
0：无请求
1：中断请求

Bit 6 **USIMF**：USIM 中断请求标志位
0：无请求
1：中断请求

Bit 5 **TKTHF**：触控按键模块 TKTH 中断请求标志位
0：无请求
1：中断请求

Bit 4 **TKRCOVF**：触控按键 TKRCOV 中断请求标志位
0：无请求
1：中断请求

Bit 3 **ADE**：A/D 转换器中断控制位
0：除能
1：使能

Bit 2 **USIME**：USIM 中断控制位
0：除能
1：使能

Bit 1 **TKTHE**：触控按键模块 TKTH 中断控制位
0：除能
1：使能

Bit 0 **TKRCOVE**：触控按键 TKRCOV 中断控制位
0：除能
1：使能

- **MFI0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	CTM1AF	CTM1PF	CTM0AF	CTM0PF	CTM1AE	CTM1PE	CTM0AE	CTM0PE
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 **CTM1AF**：CTM1 比较器 A 匹配中断请求标志位
0：无请求
1：中断请求

Bit 6 **CTM1PF**：CTM1 比较器 P 匹配中断请求标志位
0：无请求
1：中断请求

Bit 5 **CTM0AF**：CTM0 比较器 A 匹配中断请求标志位
0：无请求
1：中断请求

Bit 4 **CTM0PF**：CTM0 比较器 P 匹配中断请求标志位
0：无请求
1：中断请求

Bit 3 **CTM1AE**：CTM1 比较器 A 匹配中断控制位
0：除能
1：使能

Bit 2 **CTM1PE**：CTM1 比较器 P 匹配中断控制位
0：除能
1：使能

Bit 1 **CTM0AE**：CTM0 比较器 A 匹配中断控制位
0：除能
1：使能

Bit 0 **CTM0PE**：CTM0 比较器 P 匹配中断控制位
0：除能
1：使能

- **MFI1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PTM0AF	PTM0PF	—	—	PTM0AE	PTM0PE
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 **PTM0AF**：PTM0 比较器 A 匹配中断请求标志位
0：无请求
1：中断请求

Bit 4 **PTM0PF**：PTM0 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 **PTM0AE**：PTM0 比较器 A 匹配中断控制位
0：除能
1：使能

Bit 0 **PTM0PE**：PTM0 比较器 P 匹配中断控制位
0：除能
1：使能

- **MFI2 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	LVF	FFMGF	BMTCF	—	LVE	FFMGE	BMTCE
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7 未定义，读为“0”

Bit 6 **LVF**：LVD 中断请求标志位
0：无请求
1：中断请求

Bit 5 **FFMGF**：RF TX FIFO 长度阈值检测中断请求标志位
0：无请求
1：中断请求

Bit 4 **BMTCF**：RF 突发模式传送完成中断请求标志位
0：无请求
1：中断请求

Bit 3 未定义，读为“0”

Bit 2 **LVE**：LVD 中断控制位
0：除能
1：使能

Bit 1 **FFMGE**：RF TX FIFO 长度阈值检测中断控制位
0：除能
1：使能

Bit 0 **BMTCE**：RF 突发模式传送完成中断控制位
0：除能
1：使能

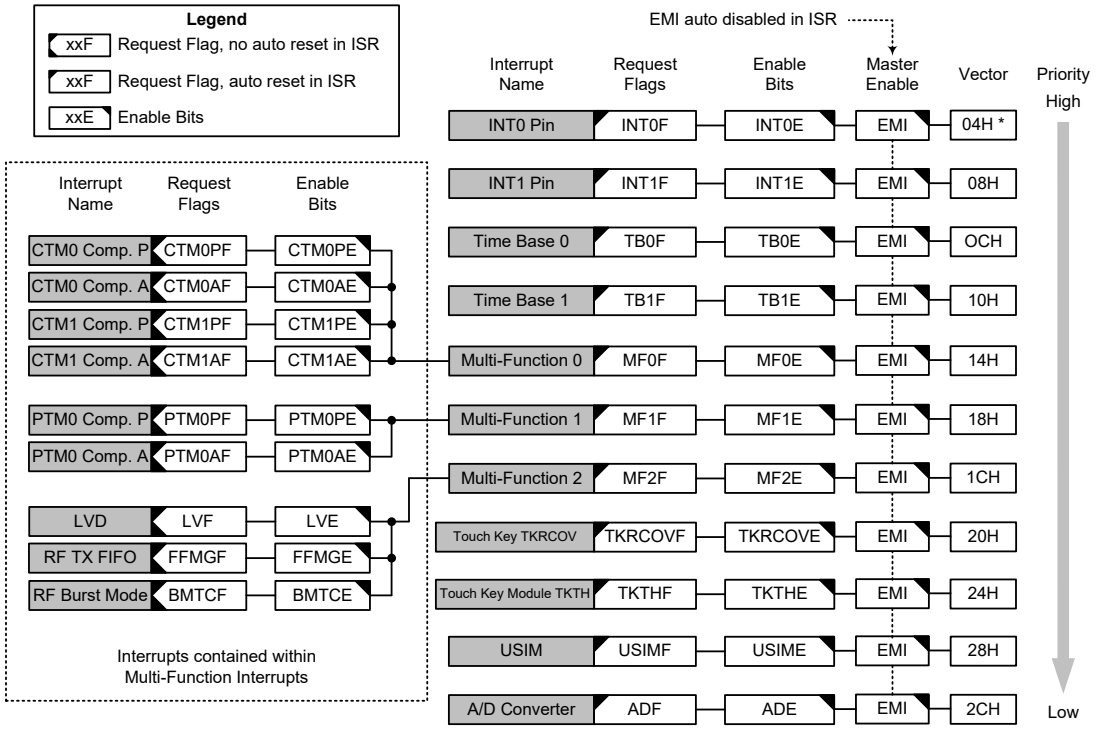
中断操作

若中断事件条件产生，如 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。



注：对于 BC66F2235，04H 的中断向量保留。

中断结构

外部中断

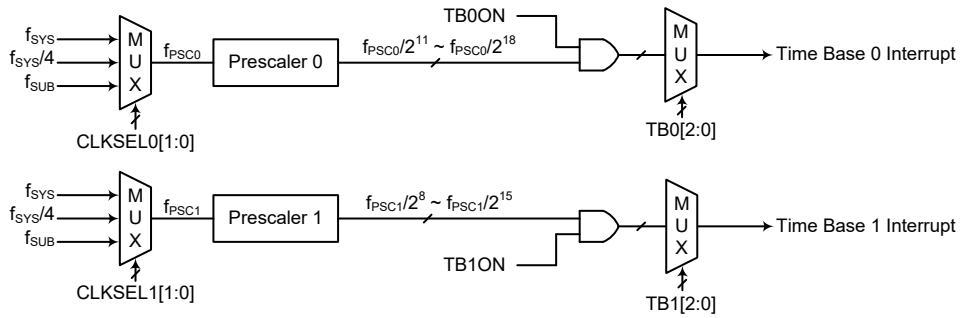
通过 INT0~INT1 引脚上的信号变化可控制外部中断。当触发沿选择位设置好触发类型，INT0~INT1 引脚的状态发生变化，外部中断请求标志 INT0F~INT1F 被置位时外部中断请求产生。若要跳转到相应中断向量地址，总中断控制位 EMI 和相应中断使能位 INT0E~INT1E 需先被置位。此外，必须使用 INTEG 寄存器使能外部中断功能并选择触发沿类型。外部中断引脚和普通 I/O 口共用，如果相应寄存器中的中断使能位被置位，并且通过引脚共用寄存器选择外部中断脚，此引脚将被作为外部中断脚使用。此时该引脚必须通过设置控制寄存器，将该引脚设置为输入口。当中断使能，堆栈未满并且外部中断引脚发生了所选择的边沿跳转，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。

时基中断

时基中断是以内部中断方式提供定时信号，由其定时器功能产生溢出信号来控制。当出现这种情况时其中断请求标志 TB0F 或 TB1F 被置位，中断请求发生。若要跳转到其相应的中断向量地址，总中断使能位 EMI 和时基使能位 TB0E 或 TB1E 需先被置位。当中断使能，堆栈未满且时基溢出时，将调用相应中断向量子程序。当响应中断服务子程序时，相应的中断请求标志位 TB0F 或 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 f_{PSC0} 或 f_{PSC1} 来自内部时钟源 f_{SYS}、$f_{SYS}/4$ 或 f_{SUB}，经过一个分频器，分频比可通过配置 TB0C 和 TB1C 寄存器中的相关位选择以获得更长的中断周期。控制时基中断周期的时钟源分别通过 PSC0R 和 PSC1R 寄存器中的 CLKSEL0[1:0] 和 CLKSEL1[1:0] 位进行选择。



时基中断

- **PSC0R 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CLKSEL01	CLKSEL00
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2　未定义，读为“0”

Bit 1~0　**CLKSEL01~CLKSEL00**：分频器 0 时钟源 f_{PSC0} 选择
00：f_{SYS}
01：$f_{SYS}/4$
1x：f_{SUB}
需要注意的是，当单片机进入深度休眠模式时，无论分频器 0 时钟源如何选择，时基 0 时钟源来自 f_{LIRC}。当单片机从深度休眠模式唤醒后，时基 0 时钟源来自 f_{PSC0}。

- **PSC1R 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CLKSEL11	CLKSEL10
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2　未定义，读为“0”

Bit 1~0　**CLKSEL11~CLKSEL10**：分频器 1 时钟源 f_{PSC1} 选择
00：f_{SYS}
01：$f_{SYS}/4$
1x：f_{SUB}

- **TB0C 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	TB0ON	—	—	—	—	TB02	TB01	TB00
R/W	R/W	—	—	—	—	R/W	R/W	R/W
POR	0	—	—	—	—	0	0	0

Bit 7 **TB0ON**：时基 0 控制位
0：除能
1：使能

Bit 6~3 未定义，读为“0”

Bit 2~0 **TB02~TB00**：时基 0 溢出周期选择位
000：$2^{11}/f_{PSC0}$
001：$2^{12}/f_{PSC0}$
010：$2^{13}/f_{PSC0}$
011：$2^{14}/f_{PSC0}$
100：$2^{15}/f_{PSC0}$
101：$2^{16}/f_{PSC0}$
110：$2^{17}/f_{PSC0}$
111：$2^{18}/f_{PSC0}$

- **TB1C 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	TB1ON	—	—	—	—	TB12	TB11	TB10
R/W	R/W	—	—	—	—	R/W	R/W	R/W
POR	0	—	—	—	—	0	0	0

Bit 7 **TB1ON**：时基 1 使能 / 除能控制位
0：除能
1：使能

Bit 6~3 未定义，读为“0”

Bit 2~0 **TB12~TB10**：时基 1 溢出周期选择位
000：$2^{8}/f_{PSC1}$
001：$2^{9}/f_{PSC1}$
010：$2^{10}/f_{PSC1}$
011：$2^{11}/f_{PSC1}$
100：$2^{12}/f_{PSC1}$
101：$2^{13}/f_{PSC1}$
110：$2^{14}/f_{PSC1}$
111：$2^{15}/f_{PSC1}$

多功能中断

此系列单片机中有 3 个多功能中断，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、RF FFMG 长度阈值检测中断和 RF 突发模式传送完成中断。

当多功能中断中任何一种中断请求标志 MFnF 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不会自动复位，必须由应用程序清零。

TM 中断

简易型和周期型 TM 各有两个中断，分别来自比较器 P 和比较器 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 或比较器 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 MFnE 需先被置位。当中断使能，堆栈未满且 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 MFnF 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。

LVD 中断

低电压检测中断属于多功能中断。当低电压检测功能检测到一个低电源电压时，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。

RF TX FIFO 长度阈值检测中断

RF TX FIFO 长度阈值检测中断属于多功能中断。当 TX FIFO 长度小于阈值长度时，TX FIFO 长度阈值检测中断请求标志位 FFMGF 被置位，TX FIFO 长度阈值检测中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI、TX FIFO 长度阈值检测中断使能位 FFMGE 和相应多功能中断使能位需先被置位。当中断使能，堆栈未满且前面提到的条件发生时，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 FFMGF 标志需在应用程序中手动清除。

RF 突发模式传送完成中断

RF 突发模式传送完成中断属于多功能中断。当突发模式下的 TX 数据包已经完全发送出去时，RF 突发模式传送完成中断请求标志位 BMTCF 被置位，RF 突发模式传送完成中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI、RF 突发模式传送完成中断使能位 BMTCE 和相应多功能中断使能位需先被置位。当中断使能，堆栈未满且前面提到的条件发生时，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 BMTCF 标志需在应用程序中手动清除。

触控按键 TKRCOV 中断

当触控按键 TKRCOV 中断请求标志位 TKRCOVF 被置位，即手动模式下时隙计数器溢出时，或自动模式下所有触控按键自动扫描完成时，或周期性自动扫描模式下在 TB0 溢出周期内最后一次扫描结束后 16-bit C/F 计数器的值被写入相应的触控按键数据存储器时，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 和触控按键 TKRCOV 中断使能位 TKRCOVE 需先被置位。当中断使能，堆栈未满且以上任何一种情况发生时，将调用触控按键 TKRCOV 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 TKRCOVF 会自动复位且 EMI 位会被清零以除能其它中断。

触控按键模块 TKTH 中断

当触控按键模块 TKTH 中断请求标志位 TKTHF 被置位，即当 MnKmTHS=0，触控按键模块 16 位 C/F 计数器值小于下限阈值时，或当 MnKmTHS=1，触控按键模块 16 位 C/F 计数器值大于上限阈值时，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 和触控按键模块 TKTH 中断请求标志位 TKTHF 需先被置位。当中断使能，堆栈未满且任何上述的阈值比较条件发生时，将调用触控按键模块 TKTH 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 TKTHF 会自动复位且 EMI 位会被清零以除能其它中断。

USIM 中断

通用串行接口模块中断，即 USIM 中断。当 USIM 接口中断标志位 USIMF 置位时，产生中断请求。由于 USIM 接口可工作在三个模式下：SPI 模式、I^2C 模式和 UART 模式，USIMF 标志位置位可由不同情况触发，取决于所选择的接口模式。

若选择 SPI 或 I^2C 模式，当一个字节数据已由 SPI/I^2C 接口接收或发送完，或 I^2C 从机地址匹配，或 I^2C 超时，中断请求标志 USIMF 被置位，USIM 中断请求产生。若选择 UART 模式，USIM 中断由几种 UART 传输条件控制。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 URX/UTX 引脚唤醒，USIM 中断请求标志 USIMF 被置位，USIM 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 和通用串行接口中断使能位 USIME 需先被置位。当中断使能，堆栈未满且以上任一种情况发生时，将调用相应的 USIM 中断向量子程序。当响应中断服务子程序时，通用串行接口中断标志位 USIMF 会自动复位且 EMI 将被自动清零以除能其它中断。

注意，当 USIM 中断是由 UART 接口触发产生的，当中断响应后，UUSR 寄存器里的标志位只有在对 UART 执行特定动作时才会被清零，详细参考 UART 章节。

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。

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或低电压都可能导致其相应的中断标志被置位，由此产生中断。

然而，当系统进入深度休眠模式时，只有时基 0 中断能够唤醒单片机，在深度休眠模式下时基 0 时钟源来自 f_{LIRC}，唤醒状态由 TB0_WAKE 指示。

必须注意避免伪唤醒情况的发生。如果要除能中断唤醒功能，单片机进入深度休眠、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 MFnF 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，时基 0 中断在深度休眠模式下也具有唤醒功能，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，在单片机进入深度休眠、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。

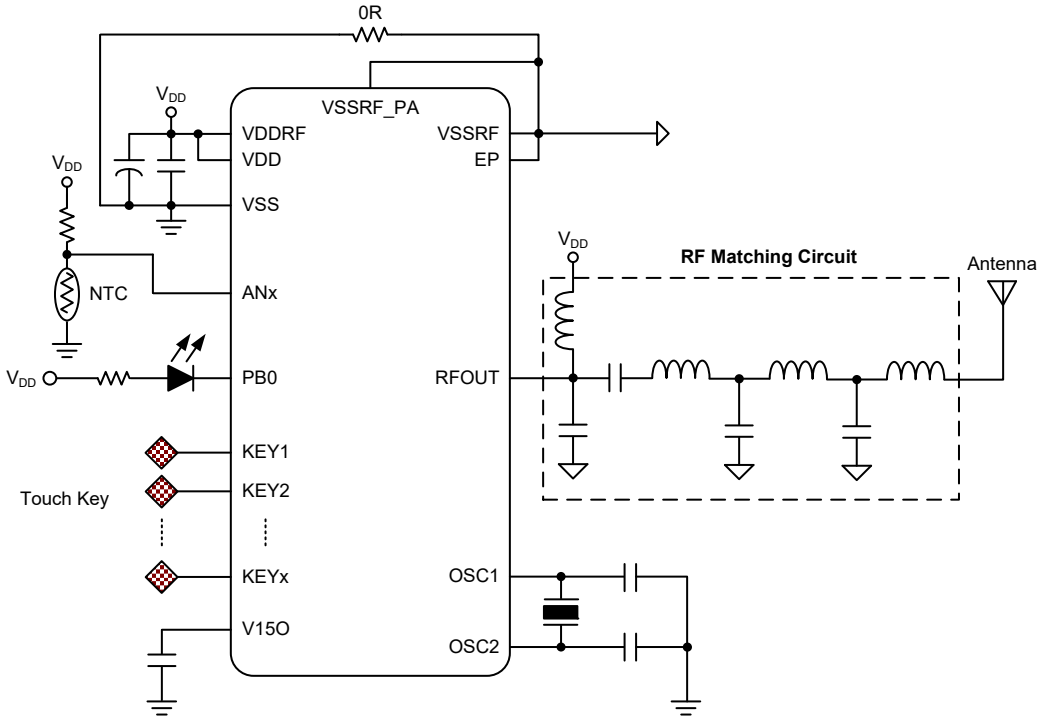
若从中断子程序中返回可执行 RET 或 RETI 指令。除了能返回至主程序外，RETI 指令还能自动设置 EMI 位为高，允许进一步中断。RET 指令只能返回至主程序，清除 EMI 位，除能进一步中断。

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
看门狗定时器选项	
1	看门狗定时器功能： 1. 始终使能 2. 由 WDTC 寄存器控制
LVR 选项	
2	LVR 功能： 1. 除能 2. 使能

应用电路


0R
VDD
VSSRF_PA
VDDRF
VDD
VSS
VSSRF
EP
NTC
ANx
PB0
RFOUT
KEY1
KEY2
Touch Key
KEYx
V15O
OSC1
OSC2
RF Matching Circuit
Antenna

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 0.5μs 中执行完成，而分支或调用操作则将在 1μs 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

当要操作的数据存储器位于数据存储器 Sector 0 时，下表说明了与数据存储器存取有关的指令。

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, SC
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, SC
ADD A, x	ACC 与立即数相加，结果放入 ACC	1	Z, C, AC, OV, SC
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, SC
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, SC
SUB A, x	ACC 与立即数相减，结果放入 ACC	1	Z, C, AC, OV, SC, CZ
SUB A,[m]	ACC 与数据存储器相减，结果放入 ACC	1	Z, C, AC, OV, SC, CZ
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, SC, CZ
SBC A, x	ACC 与立即数、进位标志相减，结果放入 ACC	1	Z, C, AC, OV, SC, CZ
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, SC, CZ
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, SC, CZ
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

助记符	说明	指令周期	影响标志位
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
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
SNZ [m]	如果数据存储器不为零，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]	读取特定页的 ROM 内容，并送至数据存储器和 TBLH	2[注]	无
TABRDL [m]	读取最后页的 ROM 内容，并送至数据存储器和 TBLH	2[注]	无
ITABRD [m]	读表指针 TBLP 自加，读取特定页的 ROM 内容，并送至数据存储器和 TBLH	2[注]	无
ITABRDL [m]	读表指针 TBLP 自加，读取最后页的 ROM 内容，并送至数据存储器和 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

助记符	说明	指令周期	影响标志位
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。

扩展指令集

扩展指令用来提供更大范围的数据存储器寻址。当被存取的数据存储器位于 Sector 0 之外的任何数据存储器 Sector，扩展指令可直接存取数据存储器而无需使用间接寻址，此举不仅可节省 Flash 存储器空间的使用，同时可提高 CPU 执行效率。

助记符	说明	指令周期	影响标志位
算术运算			
LADD A,[m]	ACC 与数据存储器相加，结果放入 ACC	2	Z, C, AC, OV, SC
LADDM A,[m]	ACC 与数据存储器相加，结果放入数据存储器	2注	Z, C, AC, OV, SC
LADC A,[m]	ACC 与数据存储器、进位标志相加，结果放入 ACC	2	Z, C, AC, OV, SC
LADCM A,[m]	ACC 与数据存储器、进位标志相加，结果放入数据存储器	2注	Z, C, AC, OV, SC
LSUB A,[m]	ACC 与数据存储器相减，结果放入 ACC	2	Z, C, AC, OV, SC, CZ
LSUBM A,[m]	ACC 与数据存储器相减，结果放入数据存储器	2注	Z, C, AC, OV, SC, CZ
LSBC A,[m]	ACC 与数据存储器、进位标志相减，结果放入 ACC	2	Z, C, AC, OV, SC, CZ
LSBCM A,[m]	ACC 与数据存储器、进位标志相减，结果放入数据存储器	2注	Z, C, AC, OV, SC, CZ
LDAA [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	2注	C
逻辑运算			
LAND A,[m]	ACC 与数据存储器做“与”运算，结果放入 ACC	2	Z
LOR A,[m]	ACC 与数据存储器做“或”运算，结果放入 ACC	2	Z
LXOR A,[m]	ACC 与数据存储器做“异或”运算，结果放入 ACC	2	Z
LANDM A,[m]	ACC 与数据存储器做“与”运算，结果放入数据存储器	2注	Z
LORM A,[m]	ACC 与数据存储器做“或”运算，结果放入数据存储器	2注	Z
LXORM A,[m]	ACC 与数据存储器做“异或”运算，结果放入数据存储器	2注	Z
LCPL [m]	对数据存储器取反，结果放入数据存储器	2注	Z
LCPLA [m]	对数据存储器取反，结果放入 ACC	2	Z
递增和递减			
LINCA [m]	递增数据存储器，结果放入 ACC	2	Z
LINC [m]	递增数据存储器，结果放入数据存储器	2注	Z
LDECA [m]	递减数据存储器，结果放入 ACC	2	Z
LDEC [m]	递减数据存储器，结果放入数据存储器	2注	Z
移位			
LRRA [m]	数据存储器右移一位，结果放入 ACC	2	无
LRR [m]	数据存储器右移一位，结果放入数据存储器	2注	无
LRRCA [m]	带进位将数据存储器右移一位，结果放入 ACC	2	C
LRRC [m]	带进位将数据存储器右移一位，结果放入数据存储器	2注	C
LRLA [m]	数据存储器左移一位，结果放入 ACC	2	无
LRL [m]	数据存储器左移一位，结果放入数据存储器	2注	无
LRLCA [m]	带进位将数据存储器左移一位，结果放入 ACC	2	C
LRLC [m]	带进位将数据存储器左移一位，结果放入数据存储器	2注	C
数据传送			
LMOV A,[m]	将数据存储器送至 ACC	2	无
LMOV [m],A	将 ACC 送至数据存储器	2注	无

助记符	说明	指令周期	影响标志位
位运算			
LCLR [m].i	清除数据存储器的位	2注	无
LSET [m].i	置位数据存储器的位	2注	无
转移			
LSZ [m]	如果数据存储器为零，则跳过下一条指令	2注	无
LSZA [m]	数据存储器送至 ACC，如果内容为零，则跳过下一条指令	2注	无
LSNZ [m]	如果数据存储器不为零，则跳过下一条指令	2注	无
LSZ [m].i	如果数据存储器的第 i 位为零，则跳过下一条指令	2注	无
LSNZ [m].i	如果数据存储器的第 i 位不为零，则跳过下一条指令	2注	无
LSIZ [m]	递增数据存储器，如果结果为零，则跳过下一条指令	2注	无
LSDZ [m]	递减数据存储器，如果结果为零，则跳过下一条指令	2注	无
LSIZA [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	2注	无
LSDZA [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	2注	无
查表			
LTABRD [m]	读取特定页的 ROM 内容，并送至数据存储器和 TBLH	3注	无
LTABRDL [m]	读取最后页的 ROM 内容，并送至数据存储器和 TBLH	3注	无
LITABRD [m]	读表指针 TBLP 自加，读取特定页的 ROM 内容，并送至数据存储器和 TBLH	3注	无
LITABRDL [m]	读表指针 TBLP 自加，读取最后页的 ROM 内容，并送至数据存储器和 TBLH	3注	无
其它指令			
LCLR [m]	清除数据存储器	2注	无
LSET [m]	置位数据存储器	2注	无
LSWAP [m]	交换数据存储器的高低字节，结果放入数据存储器	2注	无
LSWAPA [m]	交换数据存储器的高低字节，结果放入 ACC	2	无

注：1. 对扩展跳转指令而言，如果比较的结果牵涉到跳转即需 3 个周期，如果没有发生跳转，则只需两个周期。
2. 任何扩展指令若要改变 PCL 的内容将需要 3 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 ← ACC + [m] + C
影响标志位 OV、Z、AC、C、SC

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] ←ACC + [m] + C
影响标志位 OV、Z、AC、C、SC

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 ← ACC + [m]
影响标志位 OV、Z、AC、C、SC

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 ← ACC + x
影响标志位 OV、Z、AC、C、SC

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] ←ACC + [m]
影响标志位 OV、Z、AC、C、SC

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 ← ACC “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 ← ACC“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] ← ACC“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 ← Program Counter + 1 Program Counter ← 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] ← 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 ← 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 ← 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] ← $[\overline{m}]$
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←$[\overline{m}]$
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] ← ACC + 00H 或
[m] ← ACC + 06H 或
[m] ← ACC + 60H 或
[m] ← 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] ← [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的内容不变。
功能表示 ACC ← [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 无操作
影响标志位 无

OR A, [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 ← ACC“OR”[m]
影响标志位 Z

OR A, 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 ← ACC“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] ← ACC“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

RET A, 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←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
EMI ← 1

影响标志位 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 ← [m].(i+1) (i=0~6)
[m].7 ← [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 ← [m].(i+1) (i=0~6)
ACC.7 ← [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 ← [m].(i+1) (i=0~6)
[m].7← C
C ← [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 ← [m].(i+1) (i=0~6)
ACC.7 ← C
C ← [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 ← ACC – [m] – $\overline{C}$

影响标志位 OV、Z、AC、C、SC、CZ

SBC A, x Subtract immediate data from ACC with Carry
指令说明 将累加器减去立即数以及进位标志的反，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 ← ACC – [m] – $\overline{C}$
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] ← ACC – [m] – $\overline{C}$
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] ← [m] – 1，如果 [m]=0 跳过下一条指令执行
影响标志位 无

SDZA [m] Skip if decrement Data Memory is zero with result in ACC
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 ← [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] ← 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 ← 1
影响标志位 无

SIZ [m] Skip if increment Data Memory is 0
指令说明 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] ←[m] + 1，如果 [m]=0 跳过下一条指令执行
影响标志位 无

SIZA [m] Skip if increment Data Memory is zero with result in ACC
指令说明 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 ←[m] + 1，如果 ACC=0 跳过下一条指令执行
影响标志位 无

SNZ [m].i Skip if bit i of Data Memory is not 0
指令说明 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≠0，跳过下一条指令执行
影响标志位 无

SNZ [m] Skip if Data Memory is not 0
指令说明 指定数据存储器的内容会先被读出，后又被重新写入指定数据存储器内。判断指定存储器，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]≠0，跳过下一条指令执行
影响标志位 无

SUB A, [m] Subtract Data Memory from ACC

指令说明 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 ← ACC – [m]

影响标志位 OV、Z、AC、C、SC、CZ

SUBM A, [m] Subtract Data Memory from ACC with result in Data Memory

指令说明 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] ← ACC – [m]

影响标志位 OV、Z、AC、C、SC、CZ

SUB A, x Subtract immediate Data from ACC

指令说明 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。

功能表示 ACC ← ACC – x

影响标志位 OV、Z、AC、C、SC、CZ

SWAP [m] Swap nibbles of Data Memory

指令说明 将指定数据存储器的低 4 位和高 4 位互相交换。

功能表示 [m].3~[m].0 ↔ [m].7~[m].4

影响标志位 无

SWAPA [m] Swap nibbles of Data Memory with result in ACC

指令说明 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。

功能表示 ACC.3~ACC.0 ← [m].7~[m].4

ACC.7~ACC.4 ← [m].3~[m].0

影响标志位 无

SZ [m] Skip if Data Memory is 0

指令说明 指定数据存储器的内容会先被读出，后又被重新写入指定数据存储器内。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。

功能表示 如果 [m]=0，跳过下一条指令执行

影响标志位 无

SZA [m] Skip if Data Memory is 0 with data movement to ACC

指令说明 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。

功能表示 ACC ←[m]，如果 [m]=0，跳过下一条指令执行

影响标志位 无

SZ [m].i Skip if bit i of Data Memory is 0

指令说明 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] Read table (specific page) to TBLH and Data Memory

指令说明 将表格指针对 TBHP 和 TBLP 所指的程序代码低字节 (指定页) 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

TABRDL [m] Read table (last page) to TBLH and Data Memory

指令说明 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

ITABRD [m] Increment table pointer low byte first and read table (specific page) to TBLH and data memory

指令说明 自加表格指针低字节 TBLP，将表格指针对 TBHP 和 TBLP 所指的程序代码低字节 (指定页) 移至指定的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

ITABRDL [m] Increment table pointer low byte first and read table (last page) to TBLH and data memory

指令说明 自加表格指针低字节 TBLP，将表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定的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] Logical XOR Data Memory to ACC

指令说明 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] Logical XOR ACC to Data Memory

指令说明 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 Logical XOR immediate data to ACC

指令说明 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

扩展指令定义

扩展指令被用来直接存取存储在任何数据存储器 Sector 中的数据。

LADC 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 ← ACC + [m] + C
影响标志位 OV、Z、AC、C、SC

LADCM 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] ←ACC + [m] + C
影响标志位 OV、Z、AC、C、SC

LADD A, [m] Add Data Memory to ACC
指令说明 将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 ← ACC + [m]
影响标志位 OV、Z、AC、C、SC

LADDM 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] ←ACC + [m]
影响标志位 OV、Z、AC、C、SC

LAND 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 ← ACC“AND”[m]
影响标志位 Z

LANDM 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] ← ACC“AND”[m]
影响标志位 Z

LCLR [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] ← 00H
影响标志位 无

LCLR [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 ← 0
影响标志位 无

LCPL [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] ← $[\overline{m}]$
影响标志位 Z

LCPLA [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，结果被存放回累加器且数据寄存器的内容保持不变。
功能表示 ACC←$[\overline{m}]$
影响标志位 Z

LDAA [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 调整就执行对低四位加“6”，否则低四位保持不变；如果高四位的值大于“9”或C=1，那么BCD调整就执行对高四位加“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] ← ACC + 00H 或
[m] ← ACC + 06H 或
[m] ← ACC + 60H 或
[m] ← ACC + 66H
影响标志位 C

LDEC [m] Decrement Data Memory
指令说明 将指定数据存储器的内容减 1。
功能表示 [m] ← [m] – 1
影响标志位 Z

LDECA [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的内容不变。
功能表示 ACC ← [m] – 1
影响标志位 Z

LINC [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

LINCA [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

LMOV 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中。
功能表示 ACC← [m]
影响标志位 无

LMOV [m], A Move ACC to Data Memory
指令说明 将累加器的内容复制到指定数据存储器。
功能表示 [m] ← ACC
影响标志位 无

LOR A, [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 ← ACC“OR”[m]
影响标志位 Z

LORM 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] ← ACC“OR”[m]
影响标志位 Z

LRL [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

LRLA [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

LRLC [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

LRLC 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

LRR [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 ← [m].(i+1) (i=0~6)
[m].7 ← [m].0
影响标志位 无

LRRA [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 ← [m].(i+1) (i=0~6)
ACC.7 ← [m].0
影响标志位 无

LRRC [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 ← [m].(i+1) (i=0~6)
[m].7← C
C ← [m].0
影响标志位 C

LRRCA [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 ← [m].(i+1) (i=0~6)
ACC.7 ← C
C ← [m].0
影响标志位 C

LSBC 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 ← ACC – [m] – $\overline{C}$
影响标志位 OV、Z、AC、C、SC、CZ

LSBCM 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 $[m] \leftarrow ACC - [m] - \overline{C}$

影响标志位 OV、Z、AC、C、SC、CZ

LSDZ [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 3 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] ← [m] – 1，如果 [m]=0 跳过下一条指令执行

影响标志位 无

LSDZA [m] Skip if decrement Data Memory is zero with result in ACC

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 3 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 ← [m] – 1，如果 ACC=0 跳过下一条指令执行

影响标志位 无

LSET [m] Set Data Memory

指令说明 将指定数据存储器的每一个位置位为 1。

功能表示 [m] ← FFH

影响标志位 无

LSET [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 ← 1

影响标志位 无

LSIZ [m] Skip if increment Data Memory is 0

指令说明 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 3 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] ←[m] + 1，如果 [m]=0 跳过下一条指令执行

影响标志位 无

LSIZA [m] Skip if increment Data Memory is zero with result in ACC

指令说明 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 3 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 ←[m] + 1，如果 ACC=0 跳过下一条指令执行

影响标志位 无

LSNZ [m].i Skip if bit i of Data Memory is not 0

指令说明 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 3 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≠0，跳过下一条指令执行

影响标志位 无

LSNZ [m] Skip if Data Memory is not 0

指令说明 指定数据存储器的内容会先被读出，后又被重新写入指定数据存储器内。判断指定数据存储器，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 3 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]≠0，跳过下一条指令执行

影响标志位 无

LSUB A, [m] Subtract Data Memory from ACC

指令说明 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 ← ACC – [m]

影响标志位 OV、Z、AC、C、SC、CZ

LSUBM A, [m] Subtract Data Memory from ACC with result in Data Memory
指令说明 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] ← ACC – [m]
影响标志位 OV、Z、AC、C、SC、CZ

LSWAP [m] Swap nibbles of Data Memory
指令说明 将指定数据存储器的低 4 位和高 4 位互相交换。
功能表示 [m].3~[m].0 ↔ [m].7~[m].4
影响标志位 无

LSWAPA [m] Swap nibbles of Data Memory with result in ACC
指令说明 将指定数据存储器的低 4 位和高 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。
功能表示 ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0
影响标志位 无

LSZ [m] Skip if Data Memory is 0
指令说明 指定数据存储器的内容会先被读出，后又被重新写入指定数据存储器内。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 3 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]=0，跳过下一条指令执行
影响标志位 无

LSZA [m] Skip if Data Memory is 0 with data movement to ACC
指令说明 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 3 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 ← [m]，如果 [m]=0，跳过下一条指令执行
影响标志位 无

LSZ [m].i Skip if bit i of Data Memory is 0
指令说明 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 3 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

LTABRD [m] Move the ROM code (specific page) to TBLH and data memory
指令说明 将表格指针对 TBHP 和 TBLP 所指的程序代码低字节 (指定页) 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

LTABRDL [m] Read table (last page) to TBLH and Data Memory
指令说明 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

LITABRD [m] Increment table pointer low byte first and read table (specific page) to TBLH and data memory
指令说明 自加表格指针低字节 TBLP，将表格指针对 TBHP 和 TBLP 所指的程序代码低字节 (指定页) 移至指定的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

LITABRDL [m] Increment table pointer low byte first and read table (last page) to TBLH and data memory
指令说明 自加表格指针低字节 TBLP，将表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定的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

LXOR A, [m]	Logical XOR Data Memory to ACC
指令说明	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

LXORM A, [m]	Logical XOR ACC to Data Memory
指令说明	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

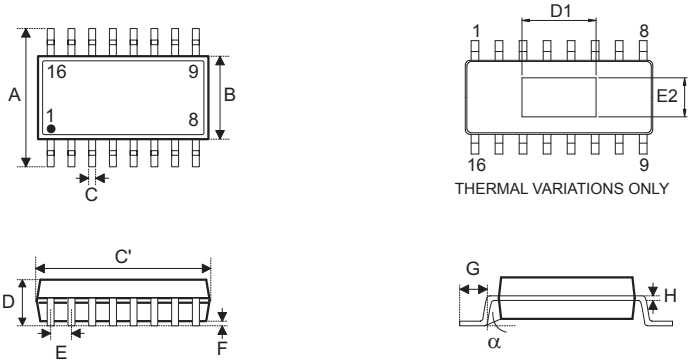
封装信息

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

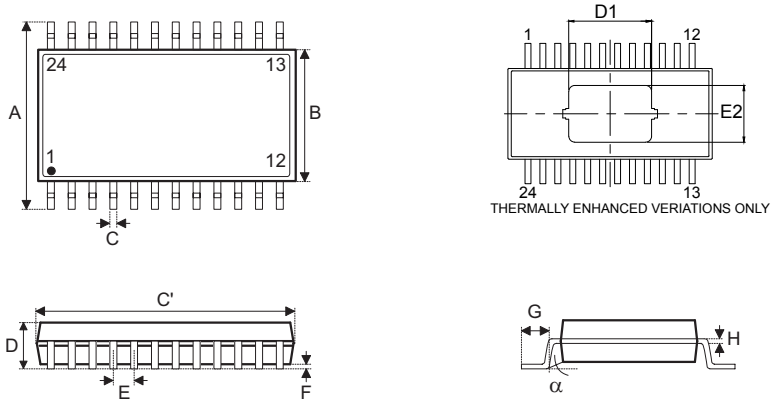
16-pin NSOP-EP (150mil) 外形尺寸



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	0.012	—	0.020
C’		0.390 BSC	
D	—	—	0.069
D1	0.152	—	0.186
E		0.050 BSC	
E2	0.066	—	0.101
F	0.000	—	0.006
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	0.31	—	0.51
C’		9.90 BSC	
D	—	—	1.75
D1	3.86	—	4.72
E		1.27 BSC	
E2	1.68	—	2.56
F	0.00	—	0.15
G	0.40	—	1.27
H	0.10	—	0.25
α	0°	—	8°

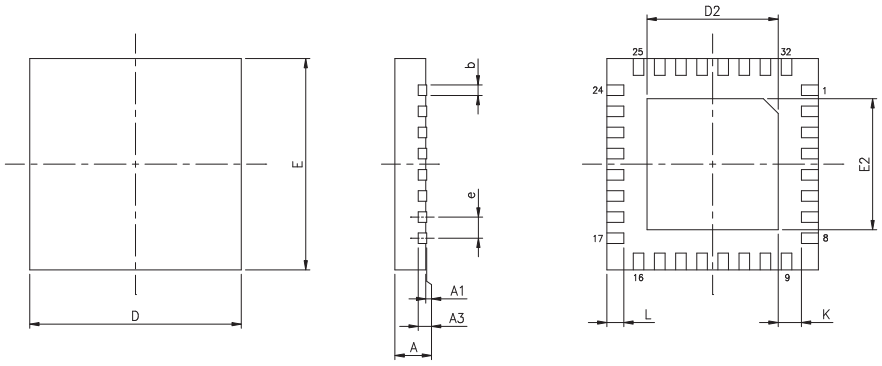
24-pin SSOP-EP (150mil) 外形尺寸



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
D1	0.119	—	0.146
E		0.025 BSC	
E2	0.081	—	0.102
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
D1	3.02	—	3.71
E		0.635 BSC	
E2	2.06	—	2.59
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°

SAW Type 32-pin QFN (4mm×4mm×0.75mm) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	典型值	最大值
A	0.028	0.030	0.031
A1	0.000	0.001	0.002
A3		0.008 REF	
b	0.006	0.008	0.010
D		0.157 BSC	
E		0.157 BSC	
e		0.016 BSC	
D2	0.100	—	0.108
E2	0.100	—	0.108
L	0.010	—	0.018
K	0.008	—	—

符号	尺寸 (单位：mm)		
	最小值	典型值	最大值
A	0.70	0.75	0.80
A1	0.00	0.02	0.05
A3		0.203 REF	
b	0.15	0.20	0.25
D		4.00 BSC	
E		4.00 BSC	
e		0.40 BSC	
D2	2.55	—	2.75
E2	2.55	—	2.75
L	0.25	—	0.45
K	0.20	—	—