A/D+LED 触控 Flash 单片机

BS66F340/BS66F350
BS66F360/BS66F370

版本 : V1.61　日期 : 2021-11-10

www.holtek.com

目 录

特性

CPU 特性

- 工作电压：
  - f_{SYS}=8MHz：2.2V ~ 5.5V
  - f_{SYS}=12MHz：2.7V ~ 5.5V
  - f_{SYS}=16MHz：3.3V ~ 5.5V
- V_{DD}=5V，系统时钟为 16MHz 时，指令周期为 0.25μs
- 提供暂停和唤醒功能，以降低功耗
- 振荡器类型：
  - 外部高频晶振 – HXT
  - 内部高频 RC – HIRC
  - 外部 32.768kHz 晶振 – LXT
  - 内部 32kHz RC – LIRC
- 内部集成 8/12/16MHz 振荡器，无需外接元件
- 多种工作模式：正常、低速、空闲和休眠
- 所有指令都可在 1 到 3 个指令周期内完成
- 查表指令
- 115 条指令
- 多达 16 层堆栈
- 位操作指令

周边特性

- Flash 程序存储器：可达 32K × 16
- RAM 数据存储器：可达 1536 × 8
- True EEPROM 存储器：128 × 8
- 集成触控按键功能 – 不需要增加外接元件
- 看门狗定时器功能
- 多达 60 个双向 I/O 口
- 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、捕捉输入、比较匹配输出、PWM 输出及单脉冲输出
- 串行接口模块 – SIM，用于 SPI 或 I^2C 通信
- 全双工异步通信接口模块 – UART
- 具有可编程源电流的 I/O 口
- 双时基功能，用于产生固定时间的中断信号
- 8 通道 12-bit 分辨精度的 A/D 转换器
- 内置温度传感器功能
- 支持在应用烧录功能 – IAP
- 低电压复位功能
- 低电压检测功能
- Flash 程序存储器可重复烧录达 100,000 次
- Flash 程序存储器数据可保存 10 年以上
- True EEPROM 数据存储器可重复烧录达 1,000,000 次
- True EEPROM 数据存储器数据可保存 10 年以上
- 多种封装类型

开发工具

为加快产品开发并简化单片机参数设置，Holtek 提供相关开发工具，用户可通过以下链接下载：

https://www.holtek.com.cn/esk-bs-210

概述

该系列单片机是 A/D 型具有 8 位高性能精简指令集的 Flash 单片机，具有完全集成的触控按键功能。其内部提供的触控按键功能以及 Flash 存储器可多次编程的特性给用户提供了可靠并易于实现的方式用于应用产品中触摸按键功能的执行。

触摸按键功能完全集成于单片机内部，使用较少的外部元件即可实现触控按键功能。存储器方面，除了 Flash 程序存储器，还包含了一个 RAM 数据存储器和一个可用于存储序列号、校准数据等非易失性数据的 True EEPROM 存储器。模拟特性方面包括一个带温度传感器的多通道 12-bit A/D 转换器。保护功能方面，包含内部看门狗定时器、低电压复位和低电压检测等特性，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。

该系列单片机提供丰富的高速和低速振荡器功能选项，且内建完整的系统振荡器，无需外围元器件。其在不同工作模式之间动态切换的能力，为用户提供了一个优化单片机操作和减少功耗的手段。内建完整的 UART、I^2C 及 SPI 功能，为设计者提供了一个易与外部硬件通信的接口。外加 I/O 使用灵活，定时器模块等其它特性，进一步增强了设备的功能性和灵活性。

该系列触摸按键单片机可以广泛应用于各种产品中，例如电子测量仪器、家用电器、电子控制工具等方面。

选型表

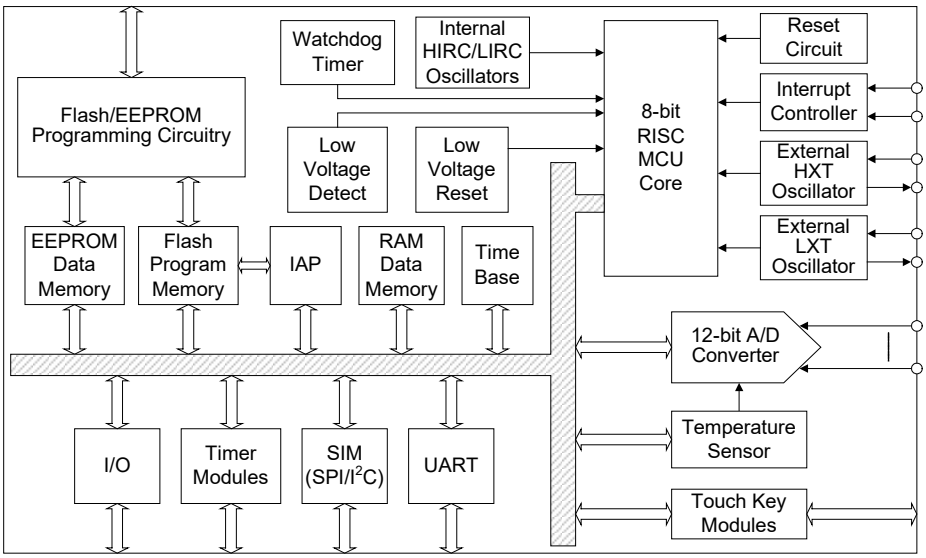
对此系列的芯片而言，大多数的特性参数都是一样的。主要差异在于存储器的容量、I/O 数量、TM 特性、触控模块、按键数据、堆栈层数和封装类型。下表列出了各单片机的主要特性。

型号	ROM	RAM	EEPROM	I/O	外部中断	A/D	温度传感器
BS66F340	4K×16	512×8	128×8	26	2	12-bit×8	√
BS66F350	8K×16	768×8	128×8	40	2	12-bit×8	√
BS66F360	16K×16	1024×8	128×8	46	2	12-bit×8	√
BS66F370	32K×16	1536×8	128×8	60	2	12-bit×8	√

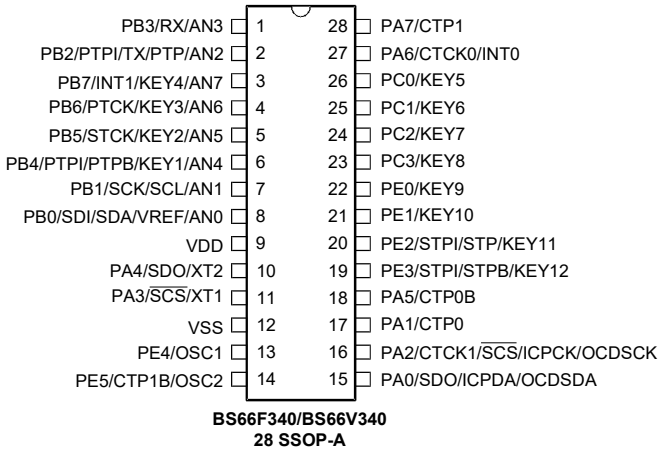
型号	TM 模块	时基	触控模块	触控按键	SIM	UART	堆栈	封装
BS66F340	10-bit CTM×2 10-bit PTM×1 16-bit STM×1	2	3	12	√	√	8	28SSOP
BS66F350	10-bit CTM×2 10-bit PTM×1 16-bit STM×1	2	5	20	√	√	8	44/48LQFP
BS66F360	10-bit CTM×2 10-bit PTM×1 16-bit STM×1	2	7	28	√	√	12	44/48LQFP
BS66F370	10-bit CTM×2 10-bit PTM×1 16-bit STM×1	2	9	36	√	√	16	44/48/64 LQFP

注：对于有不止一种封装形式的芯片，选型表反映较大封装的情况。

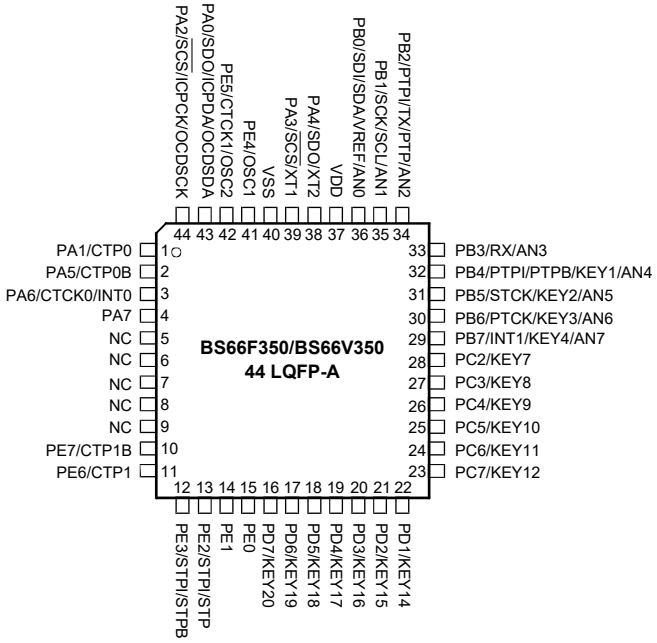
方框图

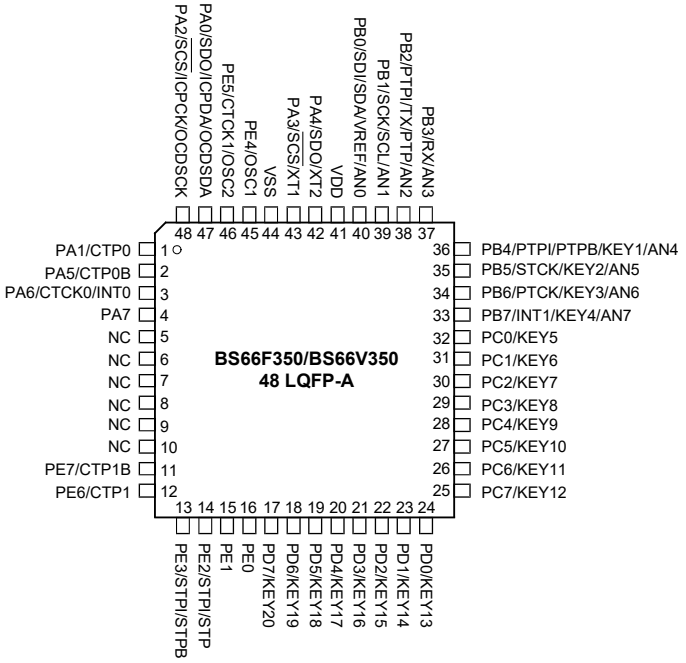


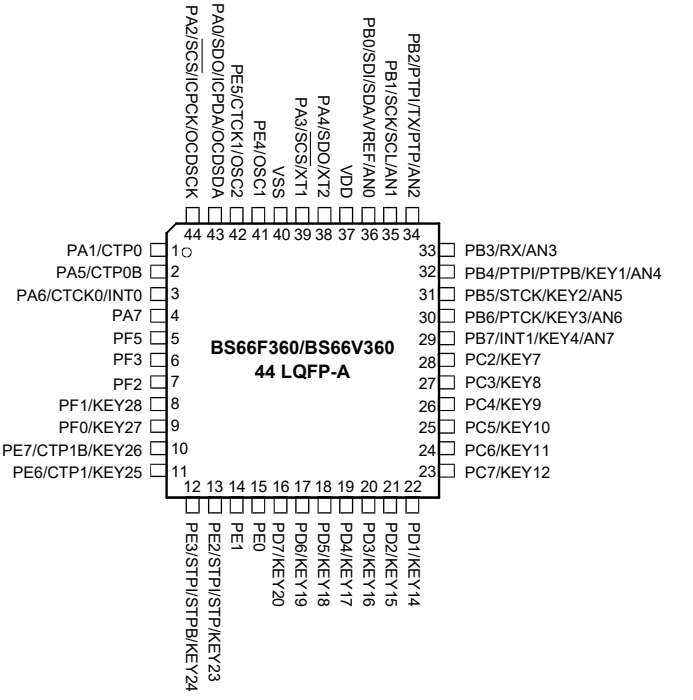
引脚图

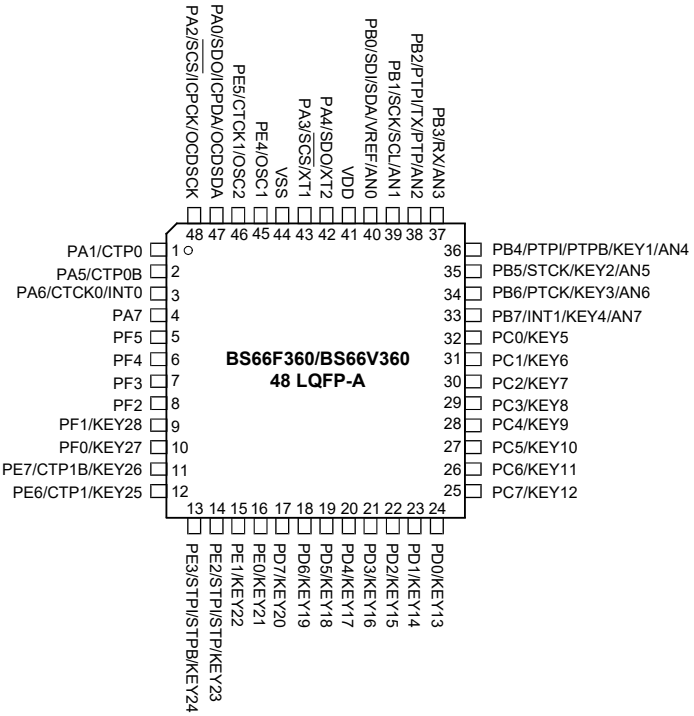


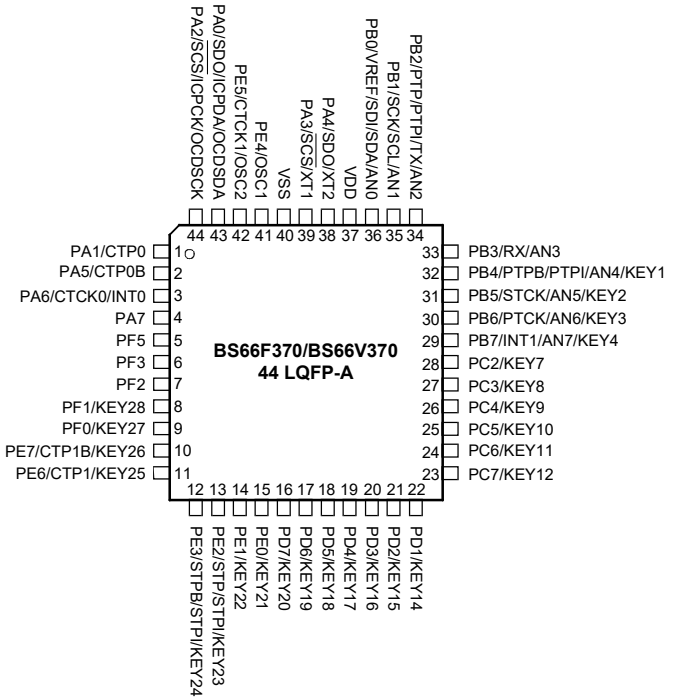
BS66F340/BS66V340
28 SSOP-A

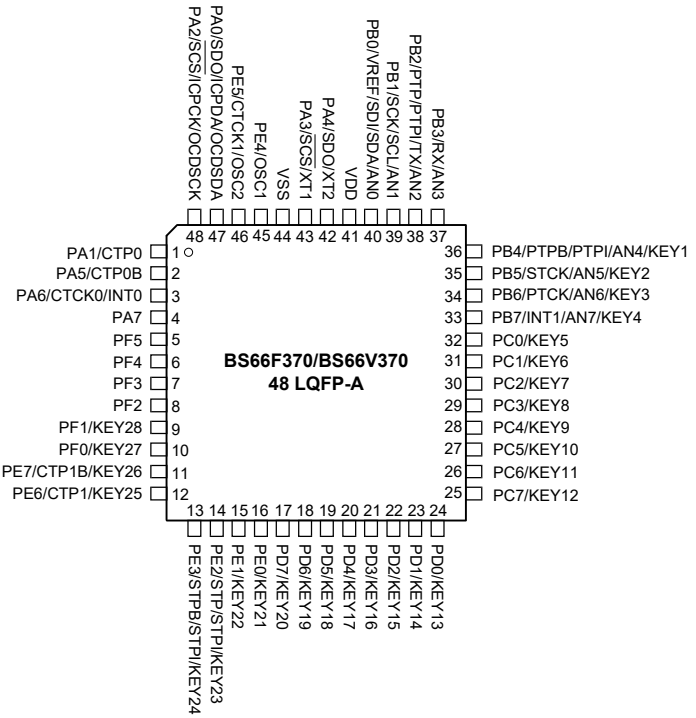

PA2/SCS/ICPCK/OCDSCK
PA0/SDO/ICPDA/OCDSDA
PE5/CTCK1/OSC2
PE4/OSC1
VSS
PA3/SCS/XT1
PA4/SDO/XT2
VDD
PB0/SDI/SDA/VREF/AN0
PB1/SCK/SCL/AN1
PB2/PTPI/TX/PTP/AN2
44 43 42 41 40 39 38 37 36 35 34
PA1/CTP0 1
PA5/CTP0B 2
PA6/CTCK0/INT0 3
PA7 4
NC 5
NC 6
NC 7
NC 8
NC 9
PE7/CTP1B 10
PE6/CTP1 11
33 PB3/RX/AN3
32 PB4/PTPI/PTPB/KEY1/AN4
31 PB5/STCK/KEY2/AN5
30 PB6/PTCK/KEY3/AN6
29 PB7/INT1/KEY4/AN7
28 PC2/KEY7
27 PC3/KEY8
26 PC4/KEY9
25 PC5/KEY10
24 PC6/KEY11
23 PC7/KEY12
12 13 14 15 16 17 18 19 20 21 22
PE3/STPI/STPB
PE2/STPI/STP
PE1
PE0
PD7/KEY20
PD6/KEY19
PD5/KEY18
PD4/KEY17
PD3/KEY16
PD2/KEY15
PD1/KEY14
BS66F350/BS66V350
44 LQFP-A

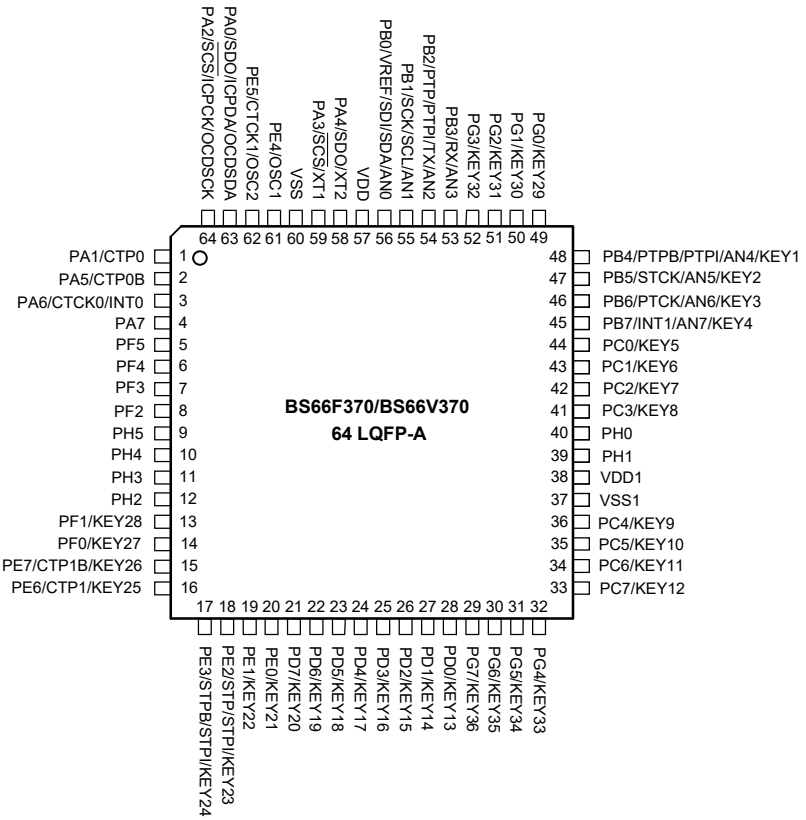

PA2/SCS/ICPCK/OCDSCK
PA0/SDO/ICPDA/OCDSDA
PE5/CTCK1/OSC2
PE4/OSC1
VSS
PA3/SCS/XT1
PA4/SDO/XT2
VDD
PB0/SDI/SDA/VREF/AN0
PB1/SCK/SCL/AN1
PB2/PTPI/TX/PTP/AN2
PB3/RX/AN3
48 47 46 45 44 43 42 41 40 39 38 37
PA1/CTP0 1
PA5/CTP0B 2
PA6/CTCK0/INT0 3
PA7 4
NC 5
NC 6
NC 7
NC 8
NC 9
NC 10
PE7/CTP1B 11
PE6/CTP1 12
36 PB4/PTPI/PTPB/KEY1/AN4
35 PB5/STCK/KEY2/AN5
34 PB6/PTCK/KEY3/AN6
33 PB7/INT1/KEY4/AN7
32 PC0/KEY5
31 PC1/KEY6
30 PC2/KEY7
29 PC3/KEY8
28 PC4/KEY9
27 PC5/KEY10
26 PC6/KEY11
25 PC7/KEY12
13 14 15 16 17 18 19 20 21 22 23 24
PE3/STPI/STPB
PE2/STPI/STP
PE1
PE0
PD7/KEY20
PD6/KEY19
PD5/KEY18
PD4/KEY17
PD3/KEY16
PD2/KEY15
PD1/KEY14
PD0/KEY13
BS66F350/BS66V350
48 LQFP-A


PA2/SCS/ICPCK/OCDSCK 44
PA0/SDO/ICPDA/OCDSDA 43
PE5/CTCK1/OSC2 42
PE4/OSC1 41
VSS 40
PA3/SCS/XT1 39
PA4/SDO/XT2 38
VDD 37
PB0/SDI/SDA/VREF/AN0 36
PB1/SCK/SCL/AN1 35
PB2/PTPI/TX/PTP/AN2 34
PA1/CTP0 1
PA5/CTP0B 2
PA6/CTCK0/INT0 3
PA7 4
PF5 5
PF3 6
PF2 7
PF1/KEY28 8
PF0/KEY27 9
PE7/CTP1B/KEY26 10
PE6/CTP1/KEY25 11
BS66F360/BS66V360
44 LQFP-A
33 PB3/RX/AN3
32 PB4/PTPI/PTPB/KEY1/AN4
31 PB5/STCK/KEY2/AN5
30 PB6/PTCK/KEY3/AN6
29 PB7/INT1/KEY4/AN7
28 PC2/KEY7
27 PC3/KEY8
26 PC4/KEY9
25 PC5/KEY10
24 PC6/KEY11
23 PC7/KEY12
12 PE3/STPI/STPB/KEY24
13 PE2/STPI/STP/KEY23
14 PE1
15 PE0
16 PD7/KEY20
17 PD6/KEY19
18 PD5/KEY18
19 PD4/KEY17
20 PD3/KEY16
21 PD2/KEY15
22 PD1/KEY14


PA2/SCS/ICPCK/OCDSCK 48
PA0/SDO/ICPDA/OCDSDA 47
PE5/CTCK1/OSC2 46
PE4/OSC1 45
VSS 44
PA3/SCS/XT1 43
PA4/SDO/XT2 42
VDD 41
PB0/SDI/SDA/VREF/AN0 40
PB1/SCK/SCL/AN1 39
PB2/PTPI/TX/PTP/AN2 38
PB3/RX/AN3 37
PA1/CTP0 1
PA5/CTP0B 2
PA6/CTCK0/INT0 3
PA7 4
PF5 5
PF4 6
PF3 7
PF2 8
PF1/KEY28 9
PF0/KEY27 10
PE7/CTP1B/KEY26 11
PE6/CTP1/KEY25 12
BS66F360/BS66V360
48 LQFP-A
36 PB4/PTPI/PTPB/KEY1/AN4
35 PB5/STCK/KEY2/AN5
34 PB6/PTCK/KEY3/AN6
33 PB7/INT1/KEY4/AN7
32 PC0/KEY5
31 PC1/KEY6
30 PC2/KEY7
29 PC3/KEY8
28 PC4/KEY9
27 PC5/KEY10
26 PC6/KEY11
25 PC7/KEY12
13 PE3/STPI/STPB/KEY24
14 PE2/STPI/STP/KEY23
15 PE1/KEY22
16 PE0/KEY21
17 PD7/KEY20
18 PD6/KEY19
19 PD5/KEY18
20 PD4/KEY17
21 PD3/KEY16
22 PD2/KEY15
23 PD1/KEY14
24 PD0/KEY13


PA2/SCS/ICPCK/OCDSCK
PA0/SDO/ICPDA/OCDSDA
PE5/CTCK1/OSC2
PE4/OSC1
VSS
PA3/SCS/XT1
PA4/SDO/XT2
VDD
PB0/VREF/SDI/SDA/AN0
PB1/SCK/SCL/AN1
PB2/PTP/PTPI/TX/AN2
44 43 42 41 40 39 38 37 36 35 34
PA1/CTP0 1
PA5/CTP0B 2
PA6/CTCK0/INT0 3
PA7 4
PF5 5
PF3 6
PF2 7
PF1/KEY28 8
PF0/KEY27 9
PE7/CTP1B/KEY26 10
PE6/CTP1/KEY25 11
BS66F370/BS66V370
44 LQFP-A
33 PB3/RX/AN3
32 PB4/PTPB/PTPI/AN4/KEY1
31 PB5/STCK/AN5/KEY2
30 PB6/PTCK/AN6/KEY3
29 PB7/INT1/AN7/KEY4
28 PC2/KEY7
27 PC3/KEY8
26 PC4/KEY9
25 PC5/KEY10
24 PC6/KEY11
23 PC7/KEY12
12 13 14 15 16 17 18 19 20 21 22
PE3/STPB/STPI/KEY24
PE2/STP/STPI/KEY23
PE1/KEY22
PE0/KEY21
PD7/KEY20
PD6/KEY19
PD5/KEY18
PD4/KEY17
PD3/KEY16
PD2/KEY15
PD1/KEY14


PA2/SCS/ICPCK/OCDSCK
PA0/SDO/ICPDA/OCDSDA
PE5/CTCK1/OSC2
PE4/OSC1
VSS
PA3/SCS/XT1
PA4/SDO/XT2
VDD
PB0/VREF/SDI/SDA/AN0
PB1/SCK/SCL/AN1
PB2/PTP/PTPI/TX/AN2
PB3/RX/AN3
48 47 46 45 44 43 42 41 40 39 38 37
PA1/CTP0 1
PA5/CTP0B 2
PA6/CTCK0/INT0 3
PA7 4
PF5 5
PF4 6
PF3 7
PF2 8
PF1/KEY28 9
PF0/KEY27 10
PE7/CTP1B/KEY26 11
PE6/CTP1/KEY25 12
BS66F370/BS66V370
48 LQFP-A
36 PB4/PTPB/PTPI/AN4/KEY1
35 PB5/STCK/AN5/KEY2
34 PB6/PTCK/AN6/KEY3
33 PB7/INT1/AN7/KEY4
32 PC0/KEY5
31 PC1/KEY6
30 PC2/KEY7
29 PC3/KEY8
28 PC4/KEY9
27 PC5/KEY10
26 PC6/KEY11
25 PC7/KEY12
13 14 15 16 17 18 19 20 21 22 23 24
PE3/STPB/STPI/KEY24
PE2/STP/STPI/KEY23
PE1/KEY22
PE0/KEY21
PD7/KEY20
PD6/KEY19
PD5/KEY18
PD4/KEY17
PD3/KEY16
PD2/KEY15
PD1/KEY14
PD0/KEY13



注：1. BS66V3x0 是 BS66F3x0 的 OCDS EV 芯片，OCDSCK 和 OCDSDA 引脚仅存在于 OCDS EV 芯片。
2. 在较小封装中可能含有未引出的引脚，需合理设置其状态以避免输入浮空造成额外耗电，详见“待机电流注意事项”和“输入 / 输出端口”章节。

引脚说明

除了电源引脚和一些发送器控制相关引脚外，该系列单片机的所有引脚都以它们的端口名称进行标注，例如 PA0、PA1 等，用于描述这些引脚的数字输入 / 输出功能。然而，这些引脚也与其它功能共用，如模数转换器，定时器模块引脚等。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。该章节的引脚描述是针对最大封装的单片机，这些引脚并非都存在于小封装的单片机内。

BS66F340

引脚名称	功能	OPT	I/T	O/T	说明
PA0/SDO/ICPDA/OCDSDA	PA0	PAWU PAPU 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
	SDO	PAS0	—	CMOS	SPI 数据输出
	ICPDA	—	ST	CMOS	ICP 数据 / 地址
	OCDSDA	—	ST	CMOS	OCDS 数据 / 地址，仅用于 EV 芯片
PA1/CTP0	PA1	PAWU PAPU 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
	CTP0	PAS0	—	CMOS	CTM0 输出
PA2/CTCK1/$\overline{\text{SCS}}$/ICPCK/OCDSCK	PA2	PAWU PAPU 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
	CTCK1	PAS0	ST	—	CTM1 时钟输入
	$\overline{\text{SCS}}$	PAS0 IFS	ST	CMOS	SPI 从机选择
	ICPCK	—	ST	CMOS	ICP 时钟引脚
	OCDSCK	—	ST	—	OCDS 时钟引脚，仅用于 EV 芯片
PA3/$\overline{\text{SCS}}$/XT1	PA3	PAWU PAPU 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
	$\overline{\text{SCS}}$	PAS0 IFS	ST	CMOS	SPI 从机选择
	XT1	PAS0	LXT	—	LXT 振荡器引脚
PA4/SDO/XT2	PA4	PAWU PAPU 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
	SDO	PAS1	—	CMOS	SPI 数据输出
	XT2	PAS1	—	LXT	LXT 振荡器引脚
PA5/CTP0B	PA5	PAWU PAPU 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
	CTP0B	PAS1	—	CMOS	CTM0 反向输出

引脚名称	功能	OPT	I/T	O/T	说明
PA6/CTCK0/INT0	PA6	PAWU PAPU 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
	CTCK0	PAS1	ST	—	CTM0 时钟输入
	INT0	PAS1 INTEG INTC0	ST	—	外部中断 0
PA7/CTP1	PA7	PAWU PAPU 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
	CTP1	PAS1	—	CMOS	CTM1 输出
PB0/SDI/SDA/VREF/AN0	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDI	PBS0	ST	—	SPI 数据输入
	SDA	PBS0	ST	NMOS	I^2C 数据线
	VREF	PBS0	—	AN	A/D 转换器参考电压输出
	AN0	PBS0	AN	—	A/D 转换器模拟输入
PB1/SCK/SCL/AN1	PB1	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCK	PBS0	ST	CMOS	SPI 串行时钟
	SCL	PBS0	ST	NMOS	I^2C 时钟线
	AN1	PBS0	AN	—	A/D 转换器模拟输入
PB2/PTPI/TX/PTP/AN2	PB2	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTPI	PBS0 IFS	ST	—	PTM 捕捉输入
	TX	PBS0	—	CMOS	UART TX 串行数据输出
	PTP	PBS0	—	CMOS	PTM 输出
	AN2	PBS0	AN	—	A/D 转换器模拟输入
PB3/RX/AN3	PB3	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	RX	PBS0	ST	—	UART RX 串行数据输入
	AN3	PBS0	AN	—	A/D 转换器模拟输入
PB4/PTPI/PTPB/KEY1/AN4	PB4	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTPI	PBS1 IFS	ST	—	PTM 捕捉输入
	PTPB	PBS1	—	CMOS	PTM 反向输出
	KEY1	PBS1	AN	—	触控按键输入
	AN4	PBS1	AN	—	A/D 转换器模拟输入

引脚名称	功能	OPT	I/T	O/T	说明
PB5/STCK/KEY2/AN5	PB5	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STCK	PBS1	ST	—	STM 时钟输入
	KEY2	PBS1	AN	—	触控按键输入
	AN5	PBS1	AN	—	A/D 转换器模拟输入
PB6/PTCK/KEY3/AN6	PB6	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTCK	PBS1	ST	—	PTM 时钟输入
	KEY3	PBS1	AN	—	触控按键输入
	AN6	PBS1	AN	—	A/D 转换器模拟输入
PB7/INT1/KEY4/AN7	PB7	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT1	PBS1 INTEG INTC0	ST	—	外部中断 1
	KEY4	PBS1	AN	—	触控按键输入
	AN7	PBS1	AN	—	A/D 转换器模拟输入
PC0/KEY5	PC0	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY5	PCS0	AN	—	触控按键输入
PC1/KEY6	PC1	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY6	PCS0	AN	—	触控按键输入
PC2/KEY7	PC2	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY7	PCS0	AN	—	触控按键输入
PC3/KEY8	PC3	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY8	PCS0	AN	—	触控按键输入
PE0/KEY9	PE0	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY9	PES0	AN	—	触控按键输入
PE1/KEY10	PE1	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY10	PES0	AN	—	触控按键输入
PE2/STPI/STP/KEY11	PE2	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STPI	PES0 IFS	ST	—	STM 捕捉输入
	STP	PES0	—	CMOS	STM 输出
	KEY11	PES0	AN	—	触控按键输入

引脚名称	功能	OPT	I/T	O/T	说明
PE3/STPI/ STPB/KEY12	PE3	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STPI	PES0 IFS	ST	—	STM 捕捉输入
	STPB	PES0	—	CMOS	STM 反向输出
	KEY12	PES0	AN	—	触控按键输入
PE4/OSC1	PE4	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	OSC1	PES1	HXT	—	HXT 振荡器引脚
PE5/CTP1B/ OSC2	PE5	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CTP1B	PES1	—	CMOS	CTM1 反向输出
	OSC2	PES1	—	HXT	HXT 振荡器引脚
VDD	VDD	—	PWR	—	正电源
VSS	VSS	—	PWR	—	负电源，接地

注：I/T：输入类型；
OPT：通过寄存器选择；
ST：施密特触发输入；
NMOS：NMOS 输出；
HXT：高频晶体振荡器；
O/T：输出类型；
PWR：电源；
AN：模拟信号；
CMOS：CMOS 输出；
LXT：低频晶体振荡器

BS66F350

引脚名称	功能	OPT	I/T	O/T	说明
PA0/SDO/ ICPDA /OCDSDA	PA0	PAWU PAPU 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
	SDO	PAS0	—	CMOS	SPI 数据输出
	ICPDA	—	ST	CMOS	ICP 数据 / 地址
	OCDSDA	—	ST	CMOS	OCDS 数据 / 地址，仅用于 EV 芯片
PA1/CTP0	PA1	PAWU PAPU 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
	CTP0	PAS0	—	CMOS	CTM0 输出
PA2/$\overline{\text{SCS}}$/ ICPCK/ OCDSCK	PA2	PAWU PAPU 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
	$\overline{\text{SCS}}$	PAS0 IFS	ST	CMOS	SPI 从机选择
	ICPCK	—	ST	CMOS	ICP 时钟引脚
	OCDSCK	—	ST	—	OCDS 时钟引脚，仅用于 EV 芯片

引脚名称	功能	OPT	I/T	O/T	说明
PA3/$\overline{SCS}$/XT1	PA3	PAWU PAPU 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
	$\overline{SCS}$	PAS0 IFS	ST	CMOS	SPI 从机选择
	XT1	PAS0	LXT	—	LXT 振荡器引脚
PA4/SDO/XT2	PA4	PAWU PAPU 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
	SDO	PAS1	—	CMOS	SPI 数据输出
	XT2	PAS1	—	LXT	LXT 振荡器引脚
PA5/CTP0B	PA5	PAWU PAPU 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
	CTP0B	PAS1	—	CMOS	CTM0 反向输出
PA6/CTCK0/ INT0	PA6	PAWU PAPU 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
	CTCK0	PAS1	ST	—	CTM0 时钟输入
	INT0	PAS1 INTEG INTC0	ST	—	外部中断 0
PA7	PA7	PAWU PAPU 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
PB0/SDI/SDA/ VREF/AN0	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDI	PBS0	ST	—	SPI 数据输入
	SDA	PBS0	ST	NMOS	I^2C 数据线
	VREF	PBS0	—	AN	A/D 转换器参考电压输出
	AN0	PBS0	AN	—	A/D 转换器模拟输入
PB1/SCK/ SCL/AN1	PB1	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCK	PBS0	ST	CMOS	SPI 串行时钟
	SCL	PBS0	ST	NMOS	I^2C 时钟线
	AN1	PBS0	AN	—	A/D 转换器模拟输入
PB2/PTPI/TX/ PTP/AN2	PB2	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTPI	PBS0 IFS	ST	—	PTM 捕捉输入
	TX	PBS0	—	CMOS	UART TX 串行数据输出
	PTP	PBS0	—	CMOS	PTM 输出
	AN2	PBS0	AN	—	A/D 转换器模拟输入

引脚名称	功能	OPT	I/T	O/T	说明
PB3/RX/AN3	PB3	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	RX	PBS0	ST	—	UART RX 串行数据输入
	AN3	PBS0	AN	—	A/D 转换器模拟输入
PB4/PTPI/ PTPB/KEY1/ AN4	PB4	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTPI	PBS1 IFS	ST	—	PTM 捕捉输入
	PTPB	PBS1	—	CMOS	PTM 反向输出
	KEY1	PBS1	AN	—	触控按键输入
	AN4	PBS1	AN	—	A/D 转换器模拟输入
PB5/STCK/ KEY2/AN5	PB5	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STCK	PBS1	ST	—	STM 时钟输入
	KEY2	PBS1	AN	—	触控按键输入
	AN5	PBS1	AN	—	A/D 转换器模拟输入
PB6/PTCK/ KEY3/AN6	PB6	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTCK	PBS1	ST	—	PTM 时钟输入
	KEY3	PBS1	AN	—	触控按键输入
	AN6	PBS1	AN	—	A/D 转换器模拟输入
PB7/INT1/ KEY4/AN7	PB7	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT1	PBS1 INTEG INTC0	ST	—	外部中断 1
	KEY4	PBS1	AN	—	触控按键输入
	AN7	PBS1	AN	—	A/D 转换器模拟输入
PC0/KEY5	PC0	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY5	PCS0	AN	—	触控按键输入
PC1/KEY6	PC1	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY6	PCS0	AN	—	触控按键输入
PC2/KEY7	PC2	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY7	PCS0	AN	—	触控按键输入
PC3/KEY8	PC3	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY8	PCS0	AN	—	触控按键输入
PC4/KEY9	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

引脚名称	功能	OPT	I/T	O/T	说明
PC5/KEY10	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
PC6/KEY11	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
PC7/KEY12	PC7	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY12	PCS1	AN	—	触控按键输入
PD0/KEY13	PD0	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY13	PDS0	AN	—	触控按键输入
PD1/KEY14	PD1	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY14	PDS0	AN	—	触控按键输入
PD2/KEY15	PD2	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY15	PDS0	AN	—	触控按键输入
PD3/KEY16	PD3	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY16	PDS0	AN	—	触控按键输入
PD4/KEY17	PD4	PDPU PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY17	PDS1	AN	—	触控按键输入
PD5/KEY18	PD5	PDPU PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY18	PDS1	AN	—	触控按键输入
PD6/KEY19	PD6	PDPU PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY19	PDS1	AN	—	触控按键输入
PD7/KEY20	PD7	PDPU PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY20	PDS1	AN	—	触控按键输入
PE0	PE0	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
PE1	PE1	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
PE2/STPI/STP	PE2	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STPI	PES0 IFS	ST	—	STM 捕捉输入
	STP	PES0	—	CMOS	STM 输出

引脚名称	功能	OPT	I/T	O/T	说明
PE3/STPI/STPB	PE3	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STPI	PES0 IFS	ST	—	STM 捕捉输入
	STPB	PES0	—	CMOS	STM 反向输出
PE4/OSC1	PE4	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	OSC1	PES1	HXT	—	HXT 振荡器引脚
PE5/CTCK1/ OSC2	PE5	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CTCK1	PES1	ST	—	CTM1 时钟输入
	OSC2	PES1	—	HXT	HXT 振荡器引脚
PE6/CTP1	PE6	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CTP1	PES1	—	CMOS	CTM1 输出
PE7/CTP1B	PE7	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CTP1B	PES1	—	CMOS	CTM1 反向输出
VDD	VDD	—	PWR	—	正电源
VSS	VSS	—	PWR	—	负电源，接地

注：I/T：输入类型；　　O/T：输出类型；
OPT：通过寄存器选择；　　PWR：电源；
ST：施密特触发输入；　　AN：模拟信号；
NMOS：NMOS 输出；　　CMOS：CMOS 输出；
HXT：高频晶体振荡器；　　LXT：低频晶体振荡器

BS66F360

引脚名称	功能	OPT	I/T	O/T	说明
PA0/SDO/ICPDA/OCDSDA	PA0	PAWU PAPU 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
	SDO	PAS0	—	CMOS	SPI 数据输出
	ICPDA	—	ST	CMOS	ICP 数据 / 地址
	OCDSDA	—	ST	CMOS	OCDS 数据 / 地址，仅用于 EV 芯片
PA1/CTP0	PA1	PAWU PAPU 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
	CTP0	PAS0	—	CMOS	CTM0 输出
PA2/$\overline{\text{SCS}}$/ICPCK/OCDSCK	PA2	PAWU PAPU 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
	$\overline{\text{SCS}}$	PAS0 IFS	ST	CMOS	SPI 从机选择
	ICPCK	—	ST	CMOS	ICP 时钟引脚
	OCDSCK	—	ST	—	OCDS 时钟引脚，仅用于 EV 芯片
PA3/$\overline{\text{SCS}}$/XT1	PA3	PAWU PAPU 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
	$\overline{\text{SCS}}$	PAS0 IFS	ST	CMOS	SPI 从机选择
	XT1	PAS0	LXT	—	LXT 振荡器引脚
PA4/SDO/XT2	PA4	PAWU PAPU 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
	SDO	PAS1	—	CMOS	SPI 数据输出
	XT2	PAS1	—	LXT	LXT 振荡器引脚
PA5/CTP0B	PA5	PAWU PAPU 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
	CTP0B	PAS1	—	CMOS	CTM0 反向输出
PA6/CTCK0/INT0	PA6	PAWU PAPU 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
	CTCK0	PAS1	ST	—	CTM0 时钟输入
	INT0	PAS1 INTEG INTC0	ST	—	外部中断 0
PA7	PA7	PAWU PAPU 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

引脚名称	功能	OPT	I/T	O/T	说明
PB0/SDI/SDA/VREF/AN0	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDI	PBS0	ST	—	SPI 数据输入
	SDA	PBS0	ST	NMOS	I^2C 数据线
	VREF	PBS0	—	AN	A/D 转换器参考电压输出
	AN0	PBS0	AN	—	A/D 转换器模拟输入
PB1/SCK/SCL/AN1	PB1	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCK	PBS0	ST	CMOS	SPI 串行时钟
	SCL	PBS0	ST	NMOS	I^2C 时钟线
	AN1	PBS0	AN	—	A/D 转换器模拟输入
PB2/PTPI/TX/PTP/AN2	PB2	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTPI	PBS0 IFS	ST	—	PTM 捕捉输入
	TX	PBS0	—	CMOS	UART TX 串行数据输出
	PTP	PBS0	—	CMOS	PTM 输出
	AN2	PBS0	AN	—	A/D 转换器模拟输入
PB3/RX/AN3	PB3	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	RX	PBS0	ST	—	UART RX 串行数据输入
	AN3	PBS0	AN	—	A/D 转换器模拟输入
PB4/PTPI/PTPB/KEY1/AN4	PB4	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTPI	PBS1 IFS	ST	—	PTM 捕捉输入
	PTPB	PBS1	—	CMOS	PTM 反向输出
	KEY1	PBS1	AN	—	触控按键输入
	AN4	PBS1	AN	—	A/D 转换器模拟输入
PB5/STCK/KEY2/AN5	PB5	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STCK	PBS1	ST	—	STM 时钟输入
	KEY2	PBS1	AN	—	触控按键输入
	AN5	PBS1	AN	—	A/D 转换器模拟输入
PB6/PTCK/KEY3/AN6	PB6	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTCK	PBS1	ST	—	PTM 时钟输入
	KEY3	PBS1	AN	—	触控按键输入
	AN6	PBS1	AN	—	A/D 转换器模拟输入

引脚名称	功能	OPT	I/T	O/T	说明
PB7/INT1/ KEY4/AN7	PB7	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT1	PBS1 INTEG INTC0	ST	—	外部中断 1
	KEY4	PBS1	AN	—	触控按键输入
	AN7	PBS1	AN	—	A/D 转换器模拟输入
PC0/KEY5	PC0	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY5	PCS0	AN	—	触控按键输入
PC1/KEY6	PC1	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY6	PCS0	AN	—	触控按键输入
PC2/KEY7	PC2	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY7	PCS0	AN	—	触控按键输入
PC3/KEY8	PC3	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY8	PCS0	AN	—	触控按键输入
PC4/KEY9	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
PC5/KEY10	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
PC6/KEY11	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
PC7/KEY12	PC7	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY12	PCS1	AN	—	触控按键输入
PD0/KEY13	PD0	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY13	PDS0	AN	—	触控按键输入
PD1/KEY14	PD1	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY14	PDS0	AN	—	触控按键输入
PD2/KEY15	PD2	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY15	PDS0	AN	—	触控按键输入
PD3/KEY16	PD3	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY16	PDS0	AN	—	触控按键输入

引脚名称	功能	OPT	I/T	O/T	说明
PD4/KEY17	PD4	PDPU PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY17	PDS1	AN	—	触控按键输入
PD5/KEY18	PD5	PDPU PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY18	PDS1	AN	—	触控按键输入
PD6/KEY19	PD6	PDPU PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY19	PDS1	AN	—	触控按键输入
PD7/KEY20	PD7	PDPU PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY20	PDS1	AN	—	触控按键输入
PE0/KEY21	PE0	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY21	PES0	AN	—	触控按键输入
PE1/KEY22	PE1	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY22	PES0	AN	—	触控按键输入
PE2/STPI/STP/ KEY23	PE2	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STPI	PES0 IFS	ST	—	STM 捕捉输入
	STP	PES0	—	CMOS	STM 输出
	KEY23	PES0	AN	—	触控按键输入
PE3/STPI/ STPB/KEY24	PE3	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STPI	PES0 IFS	ST	—	STM 捕捉输入
	STPB	PES0	—	CMOS	STM 反向输出
	KEY24	PES0	AN	—	触控按键输入
PE4/OSC1	PE4	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	OSC1	PES1	HXT	—	HXT 振荡器引脚
PE5/CTCK1/ OSC2	PE5	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CTCK1	PES1	ST	—	CTM1 时钟输入
	OSC2	PFS1	—	HXT	HXT 振荡器引脚
PE6/CTP1/ KEY25	PE6	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CTP1	PES1	—	CMOS	CTM1 输出
	KEY25	PES1	AN	—	触控按键输入

引脚名称	功能	OPT	I/T	O/T	说明
PE7/CTP1B/KEY26	PE7	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CTP1B	PES1	—	CMOS	CTM1 反向输出
	KEY26	PES1	AN	—	触控按键输入
PF0/KEY27	PF0	PFPU PFS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY27	PFS0	AN	—	触控按键输入
PF1/KEY28	PF1	PFPU PFS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY28	PFS0	AN	—	触控按键输入
PF2~PF5	PFn	PFPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
VDD	VDD	—	PWR	—	正电源
VSS	VSS	—	PWR	—	负电源，接地

注：I/T：输入类型；　　O/T：输出类型；
OPT：通过寄存器选择；　　PWR：电源；
ST：施密特触发输入；　　AN：模拟信号；
NMOS：NMOS 输出；　　CMOS：CMOS 输出；
HXT：高频晶体振荡器；　　LXT：低频晶体振荡器

BS66F370

引脚名称	功能	OPT	I/T	O/T	说明
PA0/SDO/ICPDA/OCDSDA	PA0	PAWU PAPU 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
	SDO	PAS0	—	CMOS	SPI 数据输出
	ICPDA	—	ST	CMOS	ICP 数据 / 地址
	OCDSDA	—	ST	CMOS	OCDS 数据 / 地址，仅用于 EV 芯片
PA1/CTP0	PA1	PAWU PAPU 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
	CTP0	PAS0	—	CMOS	CTM0 输出
PA2/$\overline{\text{SCS}}$/ICPCK/OCDSCK	PA2	PAWU PAPU 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
	$\overline{\text{SCS}}$	PAS0 IFS	ST	CMOS	SPI 从机选择
	ICPCK	—	ST	CMOS	ICP 时钟引脚
	OCDSCK	—	ST	—	OCDS 时钟引脚，仅用于 EV 芯片
PA3/$\overline{\text{SCS}}$/XT1	PA3	PAWU PAPU 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
	$\overline{\text{SCS}}$	PAS0 IFS	ST	CMOS	SPI 从机选择
	XT1	PAS0	LXT	—	LXT 振荡器引脚
PA4/SDO/XT2	PA4	PAWU PAPU 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
	SDO	PAS1	—	CMOS	SPI 数据输出
	XT2	PAS1	—	LXT	LXT 振荡器引脚
PA5/CTP0B	PA5	PAWU PAPU 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
	CTP0B	PAS1	—	CMOS	CTM0 反向输出
PA6/CTCK0/INT0	PA6	PAWU PAPU 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
	CTCK0	PAS1	ST	—	CTM0 时钟输入
	INT0	PAS1 INTEG INTC0	ST	—	外部中断 0
PA7	PA7	PAWU PAPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能

引脚名称	功能	OPT	I/T	O/T	说明
PB0/VREF/SDI/SDA/AN0	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDI	PBS0	ST	—	SPI 数据输入
	SDA	PBS0	ST	NMOS	I^2C 数据线
	VREF	PBS0	—	AN	A/D 转换器参考电压输出
	AN0	PBS0	AN	—	A/D 转换器模拟输入
PB1/SCK/SCL/AN1	PB1	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCK	PBS0	ST	CMOS	SPI 串行时钟
	SCL	PBS0	ST	NMOS	I^2C 时钟线
	AN1	PBS0	AN	—	A/D 转换器模拟输入
PB2/PTPI/TX/PTP/AN2	PB2	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTPI	PBS0 IFS	ST	—	PTM 捕捉输入
	TX	PBS0	—	CMOS	UART TX 串行数据输出
	PTP	PBS0	—	CMOS	PTM 输出
	AN2	PBS0	AN	—	A/D 转换器模拟输入
PB3/RX/AN3	PB3	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	RX	PBS0	ST	—	UART RX 串行数据输入
	AN3	PBS0	AN	—	A/D 转换器模拟输入
PB4/PTPI/PTPB/KEY1/AN4	PB4	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTPI	PBS1 IFS	ST	—	PTM 捕捉输入
	PTPB	PBS1	—	CMOS	PTM 反向输出
	KEY1	PBS1	AN	—	触控按键输入
	AN4	PBS1	AN	—	A/D 转换器模拟输入
PB5/STCK/KEY2/AN5	PB5	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STCK	PBS1	ST	—	STM 时钟输入
	KEY2	PBS1	AN	—	触控按键输入
	AN5	PBS1	AN	—	A/D 转换器模拟输入
PB6/PTCK/KEY3/AN6	PB6	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTCK	PBS1	ST	—	PTM 时钟输入
	KEY3	PBS1	AN	—	触控按键输入
	AN6	PBS1	AN	—	A/D 转换器模拟输入

引脚名称	功能	OPT	I/T	O/T	说明
PB7/INT1/ KEY4/AN7	PB7	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT1	PBS1 INTEG INTC0	ST	—	外部中断 1
	KEY4	PBS1	AN	—	触控按键输入
	AN7	PBS1	AN	—	A/D 转换器模拟输入
PC0/KEY5	PC0	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY5	PCS0	AN	—	触控按键输入
PC1/KEY6	PC1	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY6	PCS0	AN	—	触控按键输入
PC2/KEY7	PC2	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY7	PCS0	AN	—	触控按键输入
PC3/KEY8	PC3	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY8	PCS0	AN	—	触控按键输入
PC4/KEY9	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
PC5/KEY10	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
PC6/KEY11	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
PC7/KEY12	PC7	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY12	PCS1	AN	—	触控按键输入
PD0/KEY13	PD0	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY13	PDS0	AN	—	触控按键输入
PD1/KEY14	PD1	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY14	PDS0	AN	—	触控按键输入
PD2/KEY15	PD2	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY15	PDS0	AN	—	触控按键输入
PD3/KEY16	PD3	PDPU PDS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY16	PDS0	AN	—	触控按键输入

引脚名称	功能	OPT	I/T	O/T	说明
PD4/KEY17	PD4	PDPU PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY17	PDS1	AN	—	触控按键输入
PD5/KEY18	PD5	PDPU PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY18	PDS1	AN	—	触控按键输入
PD6/KEY19	PD6	PDPU PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY19	PDS1	AN	—	触控按键输入
PD7/KEY20	PD7	PDPU PDS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY20	PDS1	AN	—	触控按键输入
PE0/KEY21	PE0	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY21	PES0	AN	—	触控按键输入
PE1/KEY22	PE1	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY22	PES0	AN	—	触控按键输入
PE2/STPI/STP/ KEY23	PE2	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STPI	PES0 IFS	ST	—	STM 捕捉输入
	STP	PES0	—	CMOS	STM 输出
	KEY23	PES0	AN	—	触控按键输入
PE3/STPI/ STPB/KEY24	PE3	PEPU PES0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	STPI	PES0 IFS	ST	—	STM 捕捉输入
	STPB	PES0	—	CMOS	STM 反相输出
	KEY24	PES0	AN	—	触控按键输入
PE4/OSC1	PE4	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	OSC1	PES1	HXT	—	HXT 振荡器引脚
PE5/CTCK1/ OSC2	PE5	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CTCK1	PES1	ST	—	CTM1 时钟输入
	OSC2	PES1	—	HXT	HXT 振荡器引脚
PE6/CTP1/ KEY25	PE6	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CTP1	PES1	—	CMOS	CTM1 输出
	KEY25	PES1	AN	—	触控按键输入

引脚名称	功能	OPT	I/T	O/T	说明
PE7/CTP1B/KEY26	PE7	PEPU PES1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	CTP1B	PES1	—	CMOS	CTM1 反相输出
	KEY26	PES1	AN	—	触控按键输入
PF0/KEY27	PF0	PFPU PFS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY27	PFS0	AN	—	触控按键输入
PF1/KEY28	PF1	PFPU PFS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY28	PFS0	AN	—	触控按键输入
PF2	PF2	PFPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
PF3	PF3	PFPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
PF4	PF4	PFPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
PF5	PF5	PFPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
PG0/KEY29	PG0	PGPU PGS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY29	PGS0	AN	—	触控按键输入
PG1/KEY30	PG1	PGPU PGS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY30	PGS0	AN	—	触控按键输入
PG2/KEY31	PG2	PGPU PGS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY31	PGS0	AN	—	触控按键输入
PG3/KEY32	PG3	PGPU PGS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY32	PGS0	AN	—	触控按键输入
PG4/KEY33	PG4	PGPU PGS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY33	PGS1	AN	—	触控按键输入
PG5/KEY34	PG5	PGPU PGS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY34	PGS1	AN	—	触控按键输入
PG6/KEY35	PG6	PGPU PGS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY35	PGS1	AN	—	触控按键输入
PG7/KEY36	PG7	PGPU PGS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	KEY36	PGS1	AN	—	触控按键输入
PH0	PH0	PHPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻

引脚名称	功能	OPT	I/T	O/T	说明
PH1	PH1	PHPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
PH2	PH2	PHPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
PH3	PH3	PHPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
PH4	PH4	PHPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
PH5	PH5	PHPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
VDD，VDD1	VDD	—	PWR	—	正电源
VSS，VSS1	VSS	—	PWR	—	负电源，接地

注：I/T：输入类型；　　O/T：输出类型；
　　OPT：通过寄存器选择；　　PWR：电源；
　　ST：施密特触发输入；　　AN：模拟信号；
　　NMOS：NMOS 输出；　　CMOS：CMOS 输出；
　　HXT：高频晶体振荡器；　　LXT：低频晶体振荡器

极限参数

电源供应电压 ..... V_{SS}-0.3V ~ V_{SS}+6.0V
端口输入电压 ..... V_{SS}-0.3V ~ V_{DD}+0.3V
储存温度 ..... -50℃ ~ 125℃
工作温度 ..... -40℃ ~ 85℃
I_{OH}总电流 ..... -80mA
I_{OL}总电流 80mA
总功耗 500mW

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

直流电气特性

Ta=25℃

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DD}	工作电压 (HXT)	—	f_{SYS}=8MHz	2.2	—	5.5	V
			f_{SYS}=12MHz	2.7	—	5.5	V
			f_{SYS}=16MHz	3.3	—	5.5	V
	工作电压 (HIRC)	—	f_{SYS}=8MHz	2.2	—	5.5	V
			f_{SYS}=12MHz	2.7	—	5.5	V
			f_{SYS}=16MHz	3.3	—	5.5	V
I_{DD}	工作电流 (HXT)	3V	$f_{SYS}=f_H$=4MHz 无负载，所有外围功能关闭	—	500	750	μA
		5V		—	1.0	1.5	mA
		3V	$f_{SYS}=f_H$=8MHz 无负载，所有外围功能关闭	—	1.0	1.5	mA
		5V		—	2.0	3.0	mA
		3V	$f_{SYS}=f_H$=12MHz 无负载，所有外围功能关闭	—	1.5	2.75	mA
		5V		—	3.0	4.5	mA
		5V	$f_{SYS}=f_H$=16MHz 无负载，所有外围功能关闭	—	3.6	5.4	mA
	工作电流 (HIRC)	3V	$f_{SYS}=f_H$=8MHz 无负载，所有外围功能关闭	—	0.8	1.2	mA
		5V		—	1.6	2.4	mA
		3V	$f_{SYS}=f_H$=12MHz 无负载，所有外围功能关闭	—	1.2	1.8	mA
		5V		—	2.4	3.6	mA
		5V	$f_{SYS}=f_H$=16MHz 无负载，所有外围功能关闭	—	3.2	4.8	mA
	工作电流 (LXT)	3V	$f_{SYS}=f_{SUB}=f_{LXT}$=32.768kHz 无负载，所有外围功能关闭	—	10	20	μA
		5V		—	30	50	μA
	工作电流 (LIRC)	3V	$f_{SYS}=f_{SUB}=f_{LIRC}$=32kHz 无负载，所有外围功能关闭	—	10	20	μA
		5V		—	30	50	μA
I_{STB}	静态电流 (IDLE0 模式)	3V	f_{SYS} off，f_{SUB} on 无负载，所有外围功能关闭，WDT 使能	—	1.3	3.0	μA
		5V		—	2.4	5.0	μA
	静态电流 (IDLE1 模式)	3V	f_{SYS}=12MHz on，f_{SUB} on 无负载，所有外围功能关闭，WDT 使能	—	0.9	1.4	mA
		5V		—	1.4	2.1	mA
	静态电流 (SLEEP 模式)	3V	f_{SYS} off，f_{SUB} off 无负载，所有外围功能关闭，WDT 使能	—	1.2	1.8	μA
		5V		—	1.8	2.7	μA
V_{IL}	I/O 口或输入引脚的低电平输入电压	5V	—	0	—	1.5	V
		—	—	0	—	$0.2V_{DD}$	V
V_{IH}	I/O 口或输入引脚的高电平输入电压	5V	—	3.5	—	5.0	V
		—	—	$0.8V_{DD}$	—	V_{DD}	V
I_{OL}	I/O 口灌电流	3V	$V_{OL}=0.1V_{DD}$	17	34	—	mA
		5V		34	68	—	mA

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{OH}	I/O 口源电流	3V	V_{OH}=0.9V_{DD}	-5.5	-11.0	—	mA
		5V		-11.0	-22.0	—	mA
	I/O 口可编程源电流 *	3V	V_{OH}=0.9V_{DD}，PxPS[n+1:n]=00，x=A、B、C、G 或 H，n=0 或 2	-1.0	-2.0	—	mA
		5V		-2.0	-4.0	—	mA
		3V	V_{OH}=0.9V_{DD}，PxPS[n+1:n]=01，x=A、B、C、G 或 H，n=0 或 2	-1.75	-3.5	—	mA
		5V		-3.5	-7.0	—	mA
		3V	V_{OH}=0.9V_{DD}，PxPS[n+1:n]=10，x=A、B、C、G 或 H，n=0 或 2	-2.5	-5.0	—	mA
		5V		-5.0	-10.0	—	mA
		3V	V_{OH}=0.9V_{DD}，PxPS[n+1:n]=11，x=A、B、C、G 或 H，n=0 或 2	-5.5	-11.0	—	mA
		5V		-11.0	-22.0	—	mA
R_{PH}	I/O 口上拉电阻	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ

* 注：1. BS66F340 具有可编程源电流的 I/O 口包括 PA1、PA5~PA7、PB4~PB7、PC0~PC3。
2. BS66F350/BS66F360 具有可编程源电流的 I/O 口包括 PA1、PA5~PA7、PB4~PB7、PC0~PC7。
3. BS66F370 具有可编程源电流的 I/O 口包括 PA1、PA5~PA7、PB4~PB7、PC0~PC7、PG0~PG3、PH0~PH3。

交流电气特性

Ta=25℃

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
f_{SYS}	系统时钟 (HXT)	2.2V~5.5V	$f_{SYS}=f_{HXT}$=8MHz	—	8	—	MHz
		2.7V~5.5V	$f_{SYS}=f_{HXT}$=12MHz	—	12	—	MHz
		3.3V~5.5V	$f_{SYS}=f_{HXT}$=16MHz	—	16	—	MHz
	系统时钟 (HIRC)	2.2V~5.5V	$f_{SYS}=f_{HIRC}$=8MHz	—	8	—	MHz
		2.7V~5.5V	$f_{SYS}=f_{HIRC}$=12MHz	—	12	—	MHz
		3.3V~5.5V	$f_{SYS}=f_{HIRC}$=16MHz	—	16	—	MHz
	系统时钟 (LXT)	2.2V~5.5V	$f_{SYS}=f_{LXT}$=32.768kHz	—	32.768	—	kHz
	系统时钟 (LIRC)	2.2V~5.5V	$f_{SYS}=f_{LIRC}$=32kHz	—	32	—	kHz
f_{HIRC}	内部高速 RC 振荡器时钟 (HIRC) (12MHz trim at V_{DD}=3V)	3V	Ta=25℃	-2%	12	+2%	MHz
		3V±0.1V	Ta=0℃ ~ 70℃	-5%	12	+5%	MHz
		2.7V~5.5V	Ta=0℃ ~ 70℃	-7%	12	+7%	MHz
		2.7V~5.5V	Ta= -40℃ ~ 85℃	-10%	12	+10%	MHz
		3V	Ta=25℃	-20%	8	+20%	MHz
		3V	Ta=25℃	-20%	16	+20%	MHz
	内部高速 RC 振荡器时钟 (HIRC) (12MHz trim at V_{DD}=5V)	5V	Ta=25℃	-2%	12	+2%	MHz
		5V±0.1V	Ta=0℃ ~ 70℃	-5%	12	+5%	MHz
		2.7V~5.5V	Ta=0℃ ~ 70℃	-7%	12	+7%	MHz
		2.7V~5.5V	Ta= -40℃ ~ 85℃	-10%	12	+10%	MHz
		5V	Ta=25℃	-20%	8	+20%	MHz
		5V	Ta=25℃	-20%	16	+20%	MHz
f_{LIRC}	内部低速 RC 振荡器时钟 (LIRC)	5V	Ta=25℃	-10%	32	+10%	kHz
		2.2V~5.5V	Ta=-40℃ ~ 85℃	-40%	32	+40%	kHz
t_{TPI}	STPI 和 PTPI 最小输入脉宽	—	—	0.3	—	—	μs
t_{TCK}	CTCKn, STCK, PTCK 最小输入脉宽	—	—	0.3	—	—	μs
t_{INT}	中断引脚最小输入脉宽	—	—	10	—	—	μs

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 (从暂停模式唤醒，f_{SYS} off)	—	$f_{SYS}=f_H=f_{HXT} \sim f_{HXT}/64$	128	—	—	t_{HXT}
		—	$f_{SYS}=f_H=f_{HIRC} \sim f_{HIRC}/64$	16	—	—	t_{HIRC}
		—	$f_{SYS}=f_{SUB}=f_{LXT}$	1024	—	—	t_{LXT}
		—	$f_{SYS}=f_{SUB}=f_{LIRC}$	2	—	—	t_{LIRC}
	系统启动时间 (从暂停模式唤醒，f_{SYS} on)	—	$f_{SYS}=f_H \sim f_H/64$ $f_H=f_{HXT}$ 或 f_{HIRC}	2	—	—	t_H
		—	$f_{SYS}=f_{SUB}=f_{LXT}$ 或 f_{LIRC}	2	—	—	t_{SUB}
	系统启动时间 (低速模式 → 正常模式) (正常模式 → 低速模式)	—	f_{HXT} off → on (HXTF=1)	1024	—	—	t_{HXT}
		—	f_{HIRC} off → on (HIRCF=1)	16	—	—	t_{HIRC}
		—	f_{LXT} off → on (LXTF=1)	1024	—	—	t_{LXT}
	系统启动时间 (WDT 硬件复位)	—	—	0	—	—	t_{SYS}
t_{RSTD}	系统复位延迟时间 (上电复位，LVR 硬件复位，LVRC/WDTC/RSTC 软件复位)	—	—	25	50	100	ms
	系统复位延迟时间 (WDT 硬件复位)	—	—	8.3	16.7	33.3	ms
t_{EERD}	EEPROM 读周期	—	—	—	—	4	t_{SYS}
t_{EEWR}	EEPROM 写周期	—	—	—	4	6	ms

注： $t_{SYS}=1/f_{SYS}$

A/D 转换器电气特性

工作温度：-40℃～85℃，除非有特别说明

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	—	—	2.7	—	5.5	V
V_{ADI}	A/D 转换器输入电压	—	—	0	—	V_{REF}	V
V_{REF}	A/D 转换器参考电压	—	—	2	—	V_{DD}	V
DNL	A/D 非线性微分误差	3V	V_{REF}=V_{DD}， t_{ADCK} =0.5μs 或 10μs	—	—	±3	LSB
		5V					
INL	A/D 非线性积分误差	3V	V_{REF}=V_{DD}， t_{ADCK} =0.5μs 或 10μs	—	—	±4	LSB
		5V					
I_{ADC}	A/D 转换器使能的额外电流	3V	无负载，t_{ADCK} =0.5μs	—	1.0	2.0	mA
		5V		—	1.5	3.0	mA
t_{ADCK}	A/D 转换器时钟周期	—	AN ≠温度传感器输出	0.5	—	10	μs
		—	AN= 温度传感器输出	1	—	2	μs
t_{ADC}	A/D 转换时间 (包括采样和保持时间)	—	AN ≠温度传感器输出	—	16	—	t_{ADCK}
		—	AN= 温度传感器输出	—	56	—	t_{ADCK}
t_{ON2ST}	A/D 转换器 On-to-Start 时间	—	—	4	—	—	μs

温度传感器电气特性

Ta=25℃，工作温度：-40℃～85℃，除非有特别说明

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DD}	温度传感器工作电压	—	—	2.7	—	5.5	V
V_{TSVREF}	温度传感器参考电压	3V	Ta=25℃，Trim@V_{DD}=3V K_VPTAT=0, K_REFO=0	-5%	2.01	+5%	V
		5V					
$Code_{TS}$	A/D 转换器编码范围	3V	Ta=25℃，V_{REF}= V_{TSVREF}， G5XEN=1, AN= 温度传感器输出	1990	2250	2600	LSB
		5V					
		3V	Ta=90℃，V_{REF}= V_{TSVREF}， G5XEN=1, AN= 温度传感器输出	3400	3850	4250	LSB
		5V					
		3V	Ta=-40℃，V_{REF}= V_{TSVREF}， G5XEN=1, AN= 温度传感器输出	550	720	995	LSB
		5V					
t_{TSS}	温度传感器开启稳定时间	3V	—	—	—	5	μs
		5V					

LVD&LVR 电气特性

Ta=25℃

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{LVR}	低电压复位电压	—	LVR 使能，电压选择 2.1V	-5%	2.1	+5%	V
			LVR 使能，电压选择 2.55V		2.55		
			LVR 使能，电压选择 3.15V		3.15		
			LVR 使能，电压选择 3.8V		3.8		
V_{LVD}	低电压检测电压	—	LVD 使能，电压选择 2.0V	-5%	2.0	+5%	V
			LVD 使能，电压选择 2.2V		2.2		
			LVD 使能，电压选择 2.4V		2.4		
			LVD 使能，电压选择 2.7V		2.7		
			LVD 使能，电压选择 3.0V		3.0		
			LVD 使能，电压选择 3.3V		3.3		
			LVD 使能，电压选择 3.6V		3.6		
			LVD 使能，电压选择 4.0V		4.0		
V_{BG}	带隙参考电压	—	—	-5%	1.04	+5%	V
I_{OP}	LVD/LVR 工作电流	5V	LVD/LVR 使能，VBGEN=0	—	20	25	μA
		5V	LVD/LVR 使能，VBGEN=1	—	25	30	μA
t_{BGS}	V_{BG} 开启稳定时间	—	无负载	—	—	150	μs
t_{LVDS}	LVDO 稳定时间	—	LVR 使能，VBGEN=0， LVD off → on	—	—	15	μs
		—	LVR 除能，VBGEN=0， LVD off → on	—	—	150	μs
t_{LVR}	最小低电压复位时间	—	—	120	240	480	μs
t_{LVD}	最小低电压中断时间	—	—	60	120	240	μs

触控按键电气特性

Ta=25℃

触控按键 RC 振荡器 =500kHz

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{KEYOSC}	仅感应 (KEY) 振荡器工作电流	3V	*f_{SENOSC}=500kHz	—	30	60	μA
		5V		—	60	120	μA
I_{REFOSC}	仅参考振荡器工作电流	3V	*f_{REFOSC}=500kHz, MnTSS=0	—	30	60	μA
		5V		—	60	120	μA
		3V	*f_{REFOSC}=500kHz, MnTSS=1	—	30	60	μA
		5V		—	60	120	μA
C_{KEYOSC}	感应 (KEY) 振荡器外挂电容值	3V	*f_{SENOSC}=500kHz	3	10	30	pF
		5V		5	10	20	pF
C_{REFOSC}	参考振荡器内建电容值	3V	*f_{SENOSC}=500kHz	3	10	30	pF
		5V		5	10	20	pF
f_{KEYOSC}	感应 (KEY) 振荡器工作频率	3V	* C_{EXT}=7, 8, 9, 10, 11, 12, 13, 14, 15, ⋯ 50pF	100	500	1000	kHz
		5V		100	500	1000	kHz
f_{REFOSC}	参考振荡器工作频率	3V	* C_{INT}=7, 8, 9, 10, 11, 12, 13, 14, 15, ⋯ 50pF	100	500	1000	kHz
		5V		100	500	1000	kHz

注： *f_{SENOSC}=500kHz：调整 KEYn 上的外挂电容值，使得感应振荡器频率为 500kHz。
*f_{REFOSC}=500kHz：调整参考振荡器内建电容值，使得参考振荡器频率为 500kHz。

触控按键 RC 振荡器 =1000kHz

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{KEYOSC}	仅感应 (KEY) 振荡器工作电流	3V	*f_{SENOSC}=1000kHz	—	40	80	μA
		5V		—	80	160	μA
I_{REFOSC}	仅参考振荡器工作电流	3V	*f_{REFOSC}=1000kHz, MnTSS=0	—	40	80	μA
		5V		—	80	160	μA
		3V	*f_{REFOSC}=1000kHz, MnTSS=1	—	40	80	μA
		5V		—	80	160	μA
C_{KEYOSC}	感应 (KEY) 振荡器外挂电容值	3V	*f_{SENOSC}=1000kHz	3	10	25	pF
		5V		5	10	20	pF
C_{REFOSC}	参考振荡器内建电容值	3V	*f_{SENOSC}=1000kHz	3	10	25	pF
		5V		5	10	20	pF
f_{KEYOSC}	感应 (KEY) 振荡器工作频率	3V	* C_{EXT}=3, 4, 5, 6, 7, 8, 9, ⋯ 50pF	150	1000	2000	kHz
		5V		150	1000	2000	kHz
f_{REFOSC}	参考振荡器工作频率	3V	* C_{INT}=3, 4, 5, 6, 7, 8, 9, ⋯ 50pF	150	1000	2000	kHz
		5V		150	1000	2000	kHz

注： *f_{SENOSC}=1000kHz：调整 KEYn 上的外挂电容值，使得感应振荡器频率为 1000kHz。
*f_{REFOSC}=1000kHz：调整参考振荡器内建电容值，使得参考振荡器频率为 1000kHz。

触控按键 RC 振荡器 =1500kHz

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{KEYOSC}	仅感应 (KEY) 振荡器工作电流	3V	*f_{SENOSC}=1500kHz	—	60	120	μA
		5V		—	120	240	μA
I_{REFOSC}	仅参考振荡器工作电流	3V	*f_{REFOSC}=1500kHz, MnTSS=0	—	60	120	μA
		5V		—	120	240	μA
		3V	*f_{REFOSC}=1500kHz, MnTSS=1	—	60	120	μA
		5V		—	120	240	μA
C_{KEYOSC}	感应 (KEY) 振荡器外挂电容值	3V	*f_{SENOSC}=1500kHz	4	8	20	pF
		5V		5	10	20	pF
C_{REFOSC}	参考振荡器内建电容值	3V	*f_{SENOSC}=1500kHz	4	8	20	pF
		5V		5	10	20	pF
f_{KEYOSC}	感应 (KEY) 振荡器工作频率	3V	* C_{EXT}=3, 4, 5, 6, 7, 8, 9, ⋯ 50pF	150	1500	3000	kHz
		5V		150	1500	3000	kHz
f_{REFOSC}	参考振荡器工作频率	3V	* C_{EXT}=3, 4, 5, 6, 7, 8, 9, ⋯ 50pF	150	1500	3000	kHz
		5V		150	1500	3000	kHz

注：1. *f_{SENOSC}=1500kHz：调整 KEYn 上的外挂电容值，使得感应振荡器频率为 1500kHz。
2. *f_{REFOSC}=1500kHz：调整参考振荡器内建电容值，使得参考振荡器频率为 1500kHz。

触控按键 RC 振荡器 =2000kHz

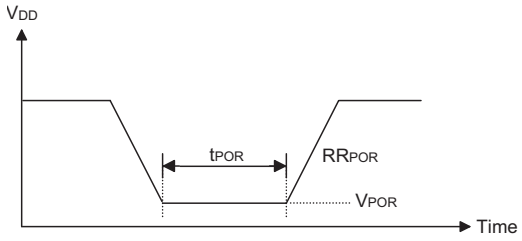
符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{KEYOSC}	仅感应 (KEY) 振荡器工作电流	3V	*f_{SENOSC}=2000kHz	—	80	160	μA
		5V		—	160	320	μA
I_{REFOSC}	仅参考振荡器工作电流	3V	*f_{REFOSC}=2000kHz, MnTSS=0,	—	80	160	μA
		5V		—	160	320	μA
		3V	*f_{REFOSC}=2000kHz, MnTSS=1,	—	80	160	μA
		5V		—	160	320	μA
C_{KEYOSC}	感应 (KEY) 振荡器外挂电容值	3V	*f_{SENOSC}=2000kHz	4	8	20	pF
		5V		5	10	20	pF
C_{REFOSC}	参考振荡器内建电容值	3V	*f_{SENOSC}=2000kHz	4	8	20	pF
		5V		5	10	20	pF
f_{KEYOSC}	感应 (KEY) 振荡器工作频率	3V	* C_{EXT}=3, 4, 5, 6, 7, 8, 9, ⋯ 50pF	150	2000	4000	kHz
		5V		150	2000	4000	kHz
f_{REFOSC}	参考振荡器工作频率	3V	* C_{EXT}=3, 4, 5, 6, 7, 8, 9, ⋯ 50pF	150	2000	4000	kHz
		5V		150	2000	4000	kHz

注：1. *f_{SENOSC}=2000kHz：调整 KEYn 上的外挂电容值，使得感应振荡器频率为 2000kHz。
2. *f_{REFOSC}=2000kHz：调整参考振荡器内建电容值，使得参考振荡器频率为 2000kHz。

上电复位特性

Ta=25℃

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



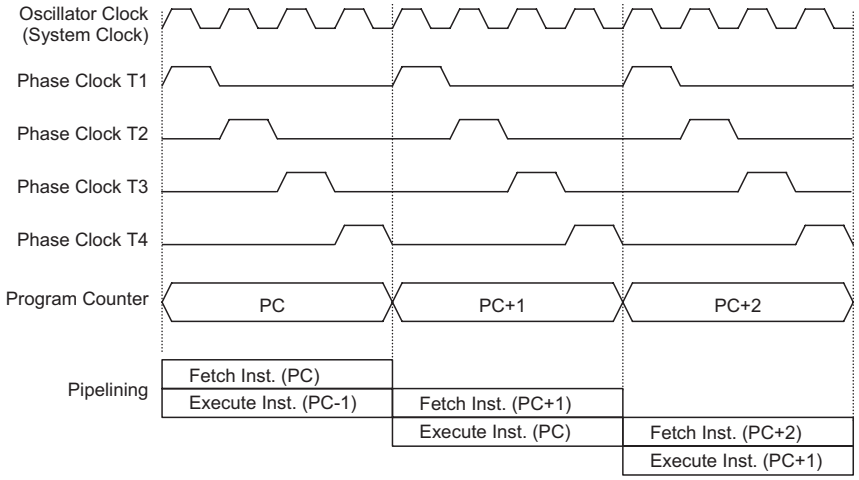
系统结构

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外，其它指令都能在一个指令周期内完成。8 位 ALU 参与指令集中所有的运算，它可完成算术运算、逻辑运算、移位、递增、递减和分支等功能，而内部的数据路径则是以通过累加器和 ALU 的方式加以简化。有些寄存器在数据存储器中被实现，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、LXT 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。

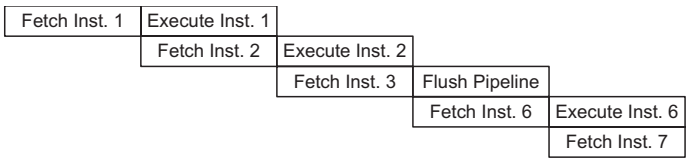
如果指令牵涉到分支，例如跳转或调用等指令，则需要两个指令周期才能完成指令执行。需要一个额外周期的原因是程序先用一个周期取出实际要跳转或调用的地址，再用另一个周期去实际执行分支动作，因此用户需要特别考虑额外周期的问题，尤其是在执行时间要求较严格的时候。



系统时序和流水线

```
1          MOV A,[12H]
2          CALL DELAY
3          CPL [12H]
4          :
5          :
6          NOP

DELAY:
```



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。对于存储器容量大于 8K 的单片机，程序存储器可分为几个程序存储区，通过程序存储区指针的 PBP0 或 PBP1 位来选择。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）
BS66F340	PC11~PC8	PC7~PC0
BS66F350	PC12~PC8	PC7~PC0
BS66F360	PBP0, PC12~PC8	PC7~PC0
BS66F370	PBP0, PBP1, PC12~PC8	PC7~PC0

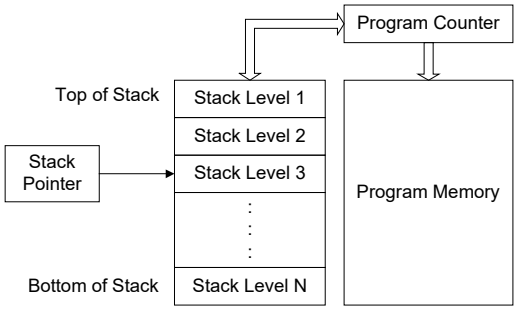
程序计数器的低字节，即程序计数器的低字节寄存器 PCL，可以通过程序控制，且它是可以读取和写入的寄存器。通过直接写入数据到这个寄存器，一个程序短跳转可直接执行，然而只有低字节的操作是有效的，跳转被限制在存储器的当前页中，即 256 个存储器地址范围内，当这样一个程序跳转要执行时，会插入一个空指令周期。程序计数器的低字节可由程序直接进行读取，PCL 的使用可能引起程序跳转，因此需要额外的指令周期。

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。



单片机型号	堆栈层数（N 值）
BS66F340/BS66F350	8
BS66F360	12
BS66F370	16

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，LOR，LXOR，LANDM，LORM，LXORM，LCPL，LCPLA
- 移位运算
  RRA，RR，RRCA，RRC，RLA，RL，RLCA，RLC
  LRRA，LRR，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
  LSZ，LSZA，LSNZ，LSIZ，LSDZ，LSIZA，LSDZA

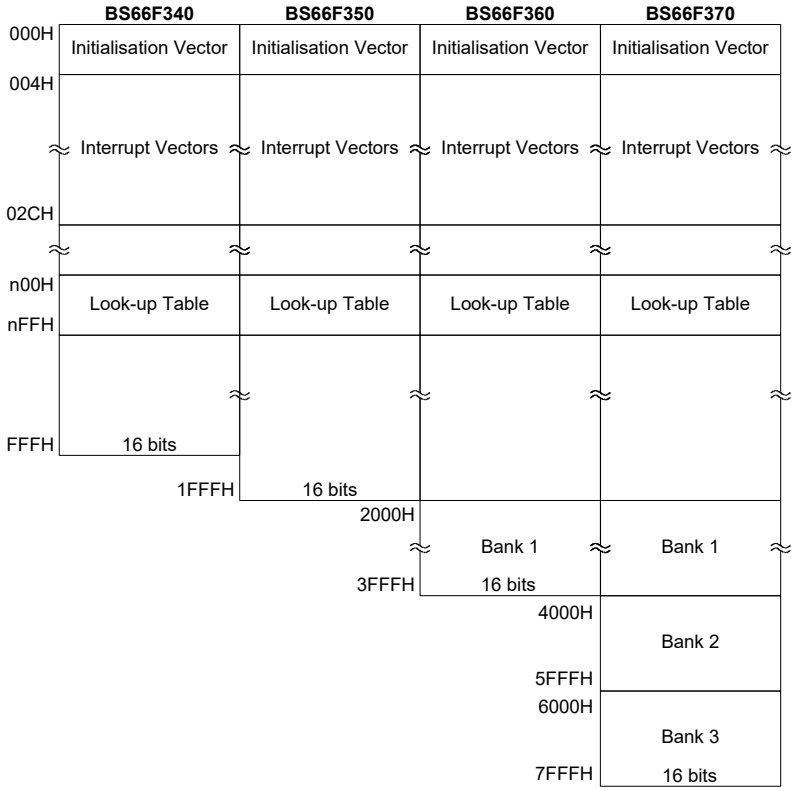
Flash 程序存储器

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。

单片机型号	容量	Banks
BS66F340	4K×16	—
BS66F350	8K×16	—
BS66F360	16K×16	0~1
BS66F370	32K×16	0~3

结构

程序存储器的容量为 4K×16~32K×16 位，程序存储器用程序计数器来寻址，其中也包含数据、表格和中断入口。数据表格可以设定在程序存储器的任何地址，由表格指针来寻址。对于 BS66F360/BS66F370，除需表格指针外，还要搭配程序存储区指针 PBP 进行寻址。



程序存储器结构

特殊向量

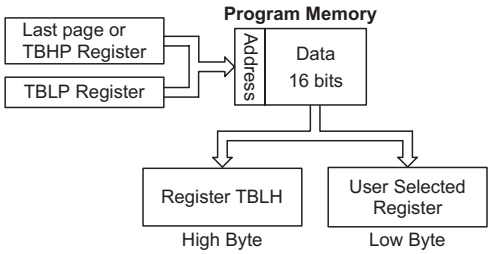
程序存储器内部某些地址保留用做诸如复位和中断入口等特殊用途。地址 0000H 是芯片复位后的程序起始地址。在芯片复位之后，程序将跳到这个地址并开始执行。

查表

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

在设定完表格指针后，若数据存储器 [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。高字节中未使用的位读为“0”。

下图是查表中寻址 / 数据流程：



查表范例

以下范例说明表格指针和表格数据如何被定义和执行。这个例子使用的表格数据用 ORG 伪指令储存在存储器中。ORG 指令的值“0F00H”指向的地址是 BS66F340 4K 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 0F06H，即最后一页起始地址后的第六个地址。值得注意的是，假如“TABRD [m]”指令被使用，则表格指针指向 TBHP 指定的页。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 / 写寄存器，可重复储存，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

```
tempreg1 db ?              ; temporary register #1
tempreg2 db ?              ; temporary register #2
:
mov a,06h                  ; initialise low table pointer - note that this
                           ; address is referenced
mov tblp,a                 ; to the last page or the page that tbhp pointed
mov a,0Fh                  ; initialise high table pointer
mov tbhp,a
:
tabrd      tempreg1        ; transfers value in table referenced by table
                           ; pointer data at program
                           ; memory address "0F06H" 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 "0F05H" transferred to tempreg2
                           ; and TBLH in this  example the data "1AH" is
                           ; transferred to tempreg1 and data "0FH" to
                           ; register tempreg2
:
org 0F00h                  ; sets initial address of program memory
dc         00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
:
```

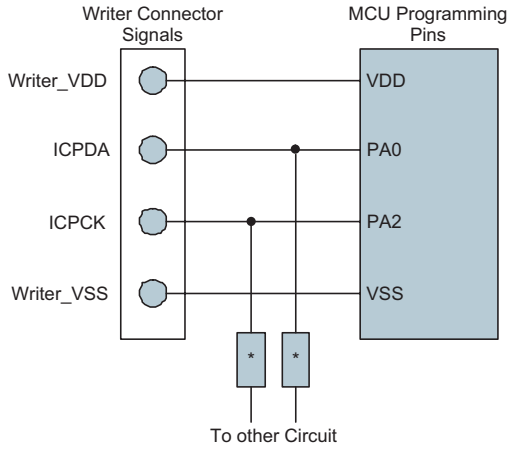
在线烧录 – ICP

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

Holtek 烧录器引脚名称	MCU 在线烧录引脚名称	功能
ICPDA	PA0	串行数据 / 地址烧录
ICPCK	PA2	时钟烧录
VDD	VDD	电源
VSS	VSS	地

程序存储器可以通过 4 线的接口在线进行烧录。其中 PA0 用于数据串行下载或上传、PA2 用于串行时钟、两条用于提供电源。芯片在线烧写的详细使用说明超出此文档的描述范围，将由专门的参考文献提供。

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



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

片上调试 – OCDS

EV 芯片 BS66V3x0 用于 Real MCU BS66F3x0 仿真。此 EV 芯片提供片上调试功能 (OCDS) 用于开发过程中的 Real MCU 调试。除了片上调试功能，EV 芯片和 Real MCU 在功能上几乎是兼容的。用户可将 OCDSDA 和 OCDSCK 引脚连接至 Holtek HT-IDE 开发工具，从而实现 EV 芯片对 Real MCU 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 EV 芯片 Real MCU 进行调试时，Real MCU OCDSDA 和 OCDSCK 引脚上的其它共用功能对 EV 芯片无效。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 User’s Guide”文件。

Holtek e-Link 引脚名称	EV 芯片引脚名称	功能
OCDSDA	OCDSDA	片上调试串行数据 / 地址输入 / 输出
OCDSCK	OCDSCK	片上调试时钟输入
VDD	VDD	电源
VSS	VSS	地

在应用烧录 – IAP

该系列单片机提供 IAP 功能来对 Flash ROM 进行数据和程序更新。用户可自行定义 IAP ROM 地址，但是用户在使用 IAP 功能时必须注意几个特点。注意，BS66F340 支持“块擦除”功能不支持“页擦除”功能。

BS66F340 IAP 配置	
块擦除	256 个字 / 块
写	4 个字 / 次
读	1 个字 / 次

BS66F350 IAP 配置	
页擦除	32 个字 / 页
写	32 个字 / 次
读	1 个字 / 次

BS66F360 IAP 配置	
页擦除	64 个字 / 页
写	64 个字 / 次
读	1 个字 / 次

BS66F370 IAP 配置	
页擦除	64 个字 / 页
写	64 个字 / 次
读	1 个字 / 次

IAP 控制寄存器

位于数据存储器 Sector 0 的地址寄存器 FARL/FARH 和数据寄存器 FD0L/FD0H、FD1L/FD1H、FD2L/FD2H 和 FD3L/FD3H 以及控制寄存器 FC0、FC1 和 FC2，都是与 IAP 相关的 Flash 存取寄存器。若使用间接寻址方式存取 FC0、FC1 和 FC2 寄存器，所有与这些寄存器相关的读写操作可以使用间接寻址寄存器 IAR1 或 IAR2，和存储器指针对 MP1L/MP1H 或 MP2L/MP2H 来执行。由于 FC0、FC1 和 FC2 控制寄存器位于地址 50H~52H 数据存储器 Sector 0 中，位于 50H~52H 地址范围的值必须首先被写入 MP1L 或 MP2L 存储器指针低字节，且“00”值也被写入 MP1H 或 MP2H 存储器指针高字节。

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
FC2 (BS66F350/ BS66F360/ BS66F370)	—	—	—	—	—	—	—	CLWB
FARL	A7	A6	A5	A4	A3	A2	A1	A0
FARH (BS66F340)	—	—	—	—	A11	A10	A9	A8
FARH (BS66F350)	—	—	—	A12	A11	A10	A9	A8
FARH (BS66F360)	—	—	A13	A12	A11	A10	A9	A8
FARH (BS66F370)	—	A14	A13	A12	A11	A10	A9	A8
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
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	1	1	1	0	0	0	0

Bit 7 **CFWEN**：Flash 存储器写使能控制位
0：Flash 存储器写功能除能
1：Flash 存储器写功能已成功使能
当此位由应用程序清零后，Flash 存储器写功能除能。注意，对此位直接写“1”不会使能写功能。此位只是用来指示 Flash 存储器写功能状态，即当此位由硬件置为“1”时，表明 Flash 存储器写功能已经成功使能，若为“0”，该功能除能。

Bit 6~4 **FMOD2~FMOD0**：模式选择
000：写程序存储器
001：块擦除 / 页擦除程序存储器
010：保留位
011：读程序存储器
100：保留位
101：保留位
110：FWEN 模式—Flash 存储器写功能使能模式
111：保留位
当这几位设置为“001”时，BS66F340 选择块擦除模式，而 BS66F350/BS66F360/BS66F370 选择页擦除模式。

Bit 3 **FWPEN**：Flash 存储器写功能使能步骤控制
0：除能
1：使能
当此位置为“1”且 FMOD2~FMOD0 为“110”时，IAP 控制器将执行“Flash 存储器写功能使能”步骤。一旦 Flash 存储器写功能成功使能，无需再设置 FWPEN 位。

Bit 2 **FWT**：Flash ROM 写开始控制位
0：未开始 Flash 存储器写或 Flash 存储器写过程已完成
1：开始 Flash 存储器写过程
此位由软件置“1”，当 Flash 存储器写过程完成，由硬件清零。

Bit 1 **FRDEN**：Flash 存储器读控制位
0：Flash 存储器读除能
1：Flash 存储器读使能

Bit 0 **FRD**：Flash 存储器读控制位
0：不初始化 Flash 存储器读或 Flash 存储器读过程已完成
1：初始化 Flash 存储器读过程
此位由软件置“1”，当 Flash 存储器读过程完成，由硬件清零。

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**：整个芯片复位控制位
当用户写特定值“55H”到该寄存器，将产生一个复位信号使整个单片机复位。

FC2 寄存器 – BS66F350/BS66F360/BS66F370

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

Bit 7~1　未定义，读为“0”

Bit 0　**CLWB**：Flash 存储器写缓冲清除控制位
0：未开始写缓冲区清除或写缓冲清除过程已完成
1：已开始写缓冲区清除过程
此位由软件置“1”，当写缓冲区清除过程完成，由硬件清零。

FARL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	A7	A6	A5	A4	A3	A2	A1	A0
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　Flash 存储器地址 [7:0]

FARH 寄存器 – BS66F340

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	A11	A10	A9	A8
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　Flash 存储器地址 [11:8]

FARH 寄存器 – BS66F350

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	A12	A11	A10	A9	A8
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　Flash 存储器地址 [12:8]

FARH 寄存器 – BS66F360

Bit	7	6	5	4	3	2	1	0
Name	—	—	A13	A12	A11	A10	A9	A8
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 5~0　Flash 存储器地址 [13:8]

FARH 寄存器 – BS66F370

Bit	7	6	5	4	3	2	1	0
Name	—	A14	A13	A12	A11	A10	A9	A8
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7　　未定义，读为“0”
Bit 6~0　　Flash 存储器地址 [14:8]

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　　第一个 Flash 存储器数据 [7:0]

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　　第一个 Flash 存储器数据 [15:8]

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　　第二个 Flash 存储器数据 [7:0]

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　　第二个 Flash 存储器数据 [15:8]

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　　第三个 Flash 存储器数据 [7:0]

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　　第三个 Flash 存储器数据 [15:8]

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　　第四个 Flash 存储器数据 [7:0]

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　　第四个 Flash 存储器数据 [15:8]

Flash 存储器写功能使能步骤

为使用户可以通过 IAP 控制寄存器来更改 Flash 存储器数据，用户必须先使能 Flash 存储器写操作，步骤如下：

步骤 1、写“110”到 FMOD2~FMOD0 位，选择 FWEN 模式。

步骤 2、FWPEN 置为“1”。步骤 1 和步骤 2 可同时执行。

步骤 3、模式数据序列 00H、04H、0DH、09H、C3H 和 40H 必须分别写入寄存器 FD1L、FD1H、FD2L、FD2H、FD3L 和 FD3H。

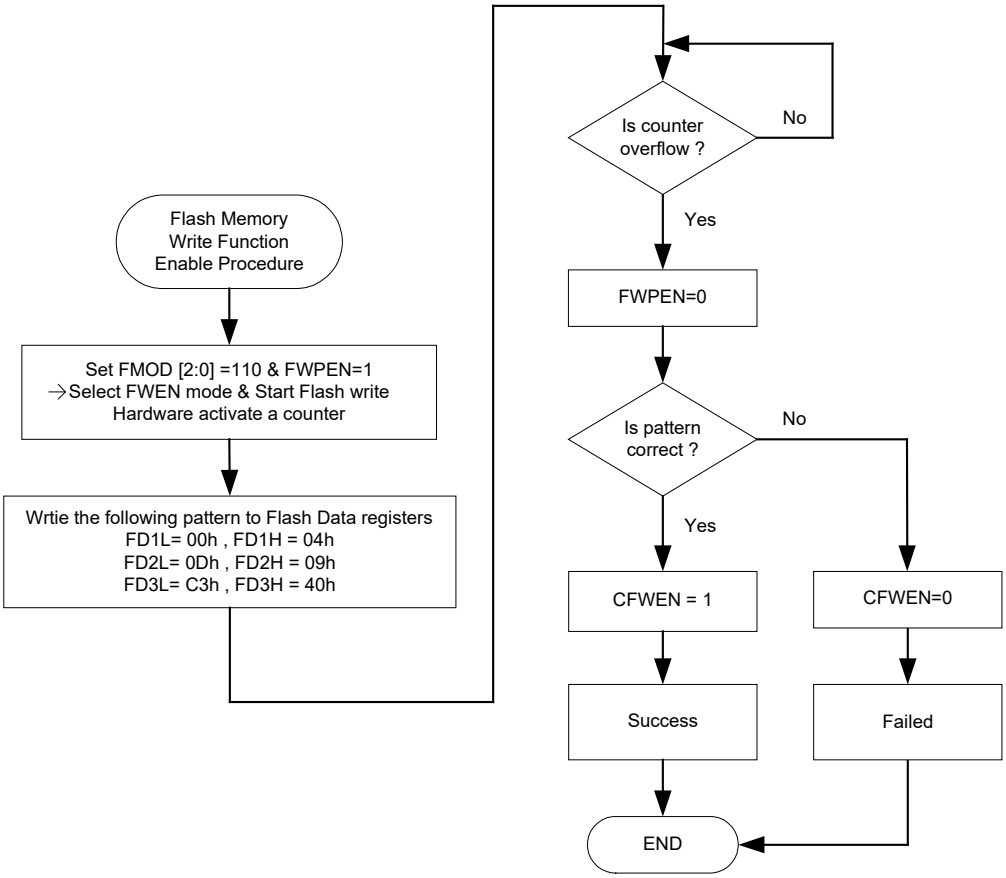
步骤 4、溢出周期为 300μs 的计数器将进行有效计时，此时允许用户将正确的数据序列写入 FD1L/FD1H~FD3L/FD3H 寄存器对。计数器时钟来自 LIRC 振荡器。

步骤 5、如果计数器溢出或模式数据不正确，Flash 存储器写操作不被使能，用户必须再次重复以上步骤。FWPEN 位将自动由硬件清零。

步骤 6、如果计数器溢出前模式数据正确，Flash 存储器写操作将使能且 FWPEN 位将自动由硬件清零。CFWEN 位也由硬件置为“1”，表明 Flash 存储器写操作成功使能。

步骤 7、一旦 Flash 存储器写操作使能，用户可通过 Flash 控制寄存器更改 Flash ROM 数据。

步骤 8、用户可以清零 CFWEN 位来除能 Flash 存储器写操作。



Flash 存储器写功能使能步骤

Flash 存储器读 / 写步骤

通过前面的 IAP 步骤成功使能 Flash 存储器写功能后，用户必须先擦除相应的 Flash 存储块或存储页，然后再开始 Flash 存储器写操作。对于 BS66F340，块擦除操作的数量是每块 256 个字，有效的块擦除地址只能由 FARH 寄存器指定，FARL 寄存器不用来指定块擦除地址。对于 BS66F350 或 BS66F360/BS66F370，页擦除操作的数量是每页 32 或 64 个字，有效的页擦除地址由 FARH 寄存器和 FARL[7:5] 或 FARL[7:6] 指定。

块擦除	FARH[3:0]	FARL[7:0]
0	0000	xxxx xxxx
1	0001	xxxx xxxx
2	0010	xxxx xxxx
3	0011	xxxx xxxx
4	0100	xxxx xxxx
5	0101	xxxx xxxx
6	0110	xxxx xxxx
7	0111	xxxx xxxx
8	1000	xxxx xxxx
9	1001	xxxx xxxx
10	1010	xxxx xxxx
11	1011	xxxx xxxx
12	1100	xxxx xxxx
13	1101	xxxx xxxx
14	1110	xxxx xxxx
15	1111	xxxx xxxx

“x”表示无关

BS66F340 块擦除数量和选择

页擦除	FARH	FARL[7:5]	FARL[4:0]
0	0000 0000	000	x xxxx
1	0000 0000	001	x xxxx
2	0000 0000	010	x xxxx
3	0000 0000	011	x xxxx
4	0000 0000	100	x xxxx
5	0000 0000	101	x xxxx
6	0000 0000	110	x xxxx
7	0000 0000	111	x xxxx
8	0000 0001	000	x xxxx
9	0000 0001	001	x xxxx
: :	: :	: :	: :
126	0000 1111	110	x xxxx
127	0000 1111	111	x xxxx
128	0001 0000	000	x xxxx
129	0001 0000	001	x xxxx
: :	: :	: :	: :
254	0001 1111	110	x xxxx
255	0001 1111	111	x xxxx

“x” 表示无关

BS66F350 页擦除数量和选择

页擦除	FARH	FARL[7:6]	FARL[5:0]
0	0000 0000	00	xx xxxx
1	0000 0000	01	xx xxxx
2	0000 0000	10	xx xxxx
3	0000 0000	11	xx xxxx
4	0000 0001	00	xx xxxx
5	0000 0001	01	xx xxxx
: :	: :	: :	: :
126	0001 1111	10	xx xxxx
127	0001 1111	11	xx xxxx
128	0010 0000	00	xx xxxx
129	0010 0000	01	xx xxxx
: :	: :	: :	: :
254	0011 1111	10	xx xxxx
255	0011 1111	11	xx xxxx

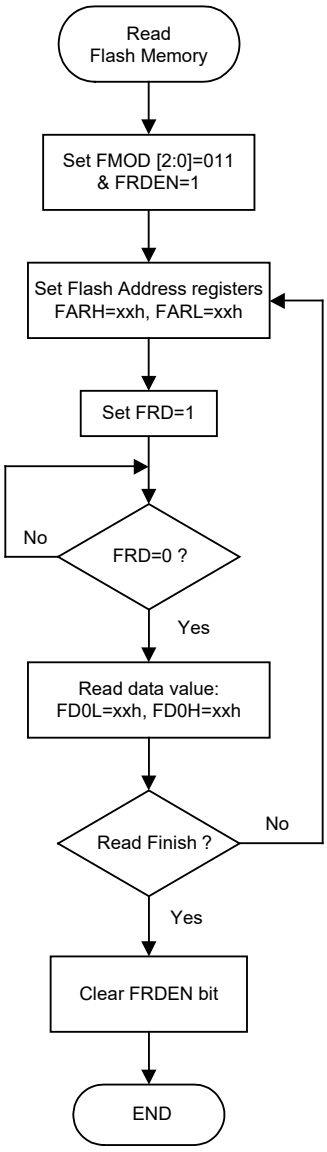
“x” 表示无关

BS66F360 页擦除数量和选择

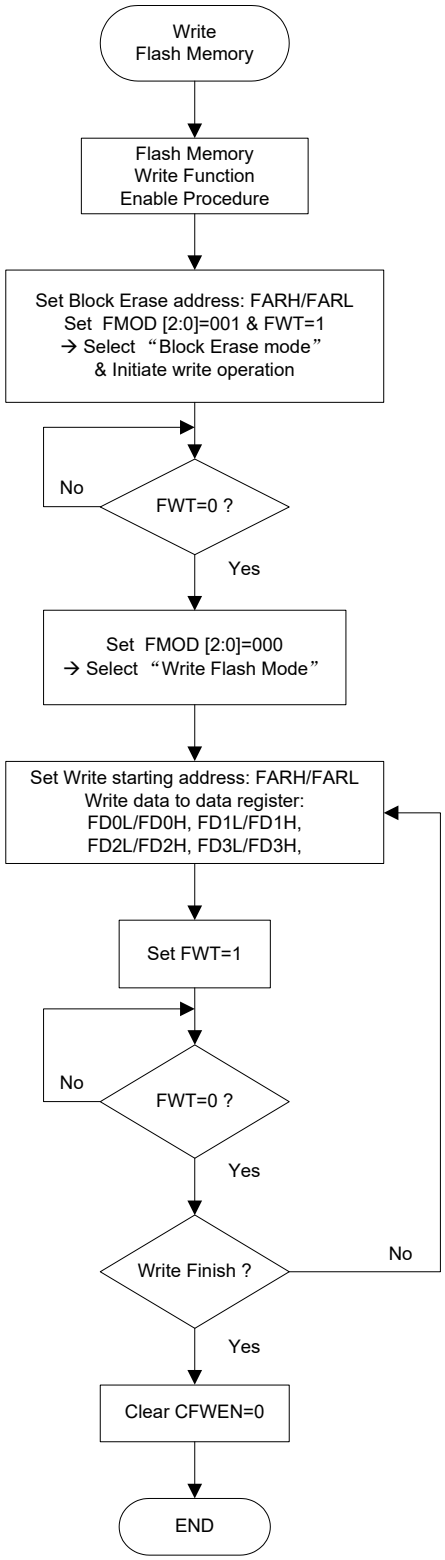
Erase Page	FARH	FARL [7:6]	FARL [5:0]
0	0000 0000	00	xx xxxx
1	0000 0000	01	xx xxxx
2	0000 0000	10	xx xxxx
3	0000 0000	11	xx xxxx
4	0000 0001	00	xx xxxx
5	0000 0001	01	xx xxxx
: :	: :	: :	: :
126	0001 1111	10	xx xxxx
127	0001 1111	11	xx xxxx
128	0010 0000	00	xx xxxx
129	0010 0000	01	xx xxxx
: :	: :	: :	: :
254	0011 1111	10	xx xxxx
255	0011 1111	11	xx xxxx
256	0100 0000	00	xx xxxx
257	0100 0000	01	xx xxxx
: :	: :	: :	: :
510	0111 1111	10	xx xxxx
511	0111 1111	11	xx xxxx

“x” 表示无关

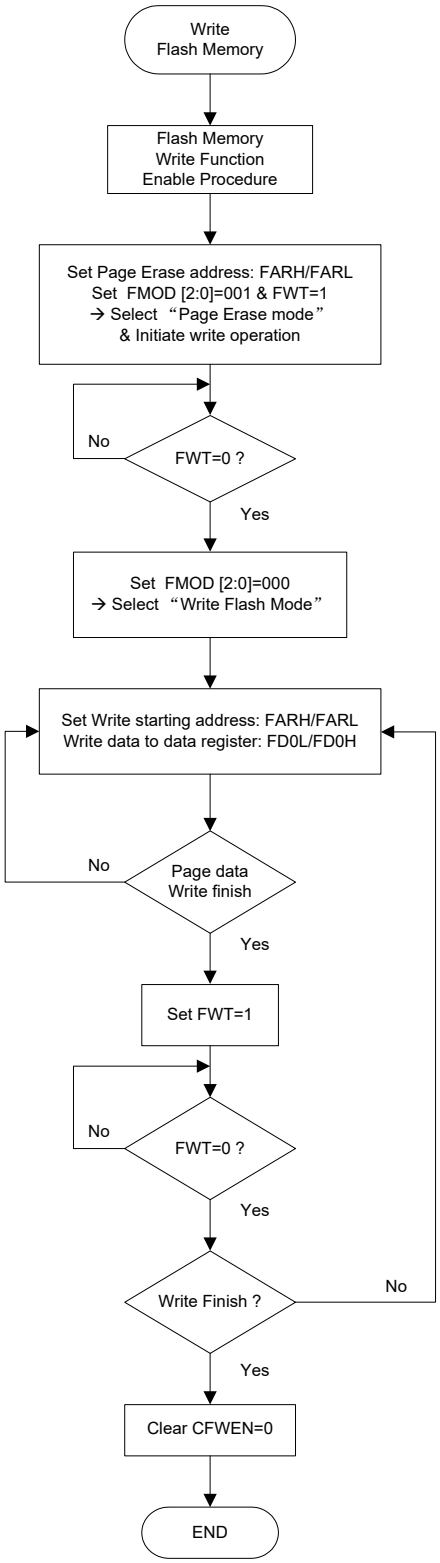
BS66F370 页擦除数量和选择



读 **Flash** 存储器步骤



写 Flash 存储器步骤 – BS66F340



写 Flash 存储器步骤 – BS66F350/BS66F360/BS66F370

注：当 FWT 或 FRD 位置为“1”，单片机停止工作。

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。

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。

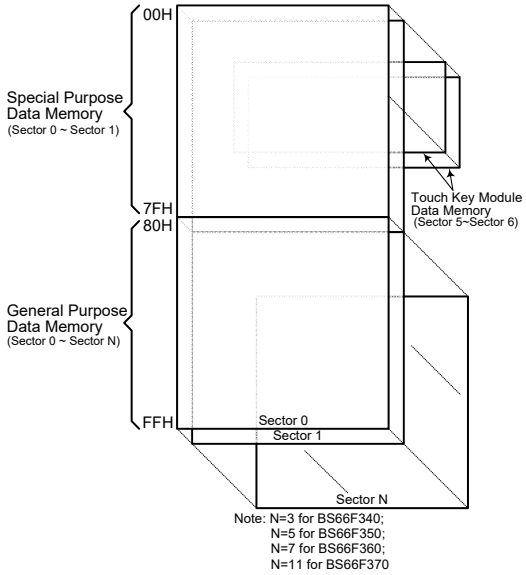
结构

数据存储器被分为多个 Sector，都位于 8 位存储器中。每个数据存储器 Sector 分为两类，特殊功能数据存储器和通用数据存储器。

特殊功能数据存储器地址范围为：00H~7FH，而通用数据存储器地址范围为：80H~FFH，触控按键模块数据存储区除外。触控按键模块数据存储区域位于 Sector 5 和 Sector 6，起始地址为“00H”。详细分布情况如下表格所示。

单片机型号	特殊功能数据存储器	通用数据存储器		触控按键模块数据存储器
	分布 Sectors	容量	Sector: 地址	Sector: 地址
BS66F340	0, 1	512×8	0: 80H~FFH 1: 80H~FFH 2: 80H~FFH 3: 80H~FFH	5: 00H~17H 6: 00H~17H
BS66F350	0, 1	768×8	0: 80H~FFH 1: 80H~FFH : 5: 80H~FFH	5: 00H~27H 6: 00H~27H
BS66F360	0, 1	1024×8	0: 80H~FFH 1: 80H~FFH : 7: 80H~FFH	5: 00H~37H 6: 00H~37H
BS66F370	0, 1	1536×8	0: 80H~FFH 1: 80H~FFH : 11: 80H~FFH	5: 00H~47H 6: 00H~47H

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。相比较于程序存储器 Bank 选择需使用程序存储区指针 PBP，数据存储器 Bank 的选择无需使用数据存储区指针。对整个数据存储器使用间接寻址方式时，可通过 MP1H 或 MP2H 寄存器指定所需 Sector，通过 MP1L 或 MP2L 寄存器指定所选 Sector 的具体地址。

直接寻址可用于所有 Sector，通过相应的指令可以寻址所有可用的数据存储器空间。所访问的数据存储器位于除 Sector 0 外的任何数据存储器 Sector，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”可以是 10~11 位，取决于所选的单片机，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

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

	Sector 0	Sector 1
00H	IAR0	TKTMR
01H	MP0	TKC0
02H	IAR1	TK16DL
03H	MP1L	TK16DH
04H	MP1H	TKC1
05H	ACC	TKM016DL
06H	PCL	TKM016DH
07H	TBLP	TKM0ROL
08H	TBLH	TKM0ROH
09H	TBHP	TKM0C0
0AH	STATUS	TKM0C1
0BH		TKM0C2
0CH	IAR2	TKM116DL
0DH	MP2L	TKM116DH
0EH	MP2H	TKM1ROL
0FH	RSTFC	TKM1ROH
10H	INTC0	TKM1C0
11H	INTC1	TKM1C1
12H	INTC2	TKM1C2
13H		TKM216DL
14H	PA	TKM216DH
15H	PAC	TKM2ROL
16H	PAPU	TKM2ROH
17H	PAWU	TKM2C0
18H	PB	TKM2C1
19H	PBC	TKM2C2
1AH	PBPU	
1BH	INTEG	
1CH	SCC	
1DH	HIRCC	
1EH	HXTC	
1FH	LXTC	
20H	LVDC	
21H	LVRC	
22H	WDTC	
23H	RSTC	
24H	PC	
25H	PCC	
26H	PCPU	
27H		
28H		
29H		
2AH	MFI0	
2BH	MFI1	
2CH	MFI2	
2DH	MFI3	
2EH	ADRL	
2FH	ADRH	
30H	ADCR0	
31H	ADCR1	
32H	PSCR0	
33H	TB0C	
34H	TB1C	
35H	SIMTOC	
36H	SIMC0	
37H	SIMC1	
38H	SIMD	
39H	SIMA/SIMC2	
3AH	CTM0C0	
3BH	CTM0C1	
3CH	CTM0DL	
3DH	CTM0DH	
3EH	CTM0AL	
3FH	CTM0AH	

	Sector 0	Sector 1
40H		EEC
41H	EEA	
42H		
43H	EED	
44H	PSCR1	IFS
45H		PAS0
46H	SLEDC	PAS1
47H		PBS0
48H	PTMC0	PBS1
49H	PTMC1	PCS0
4AH	PTMDL	
4BH	PTMDH	
4CH	PTMAL	
4DH	PTMAH	PES0
4EH	PTMRPL	PES1
4FH	PTMRPH	
50H	FC0	
51H	FC1	
52H		
53H	FARL	
54H	FARH	
55H	FD0L	
56H	FD0H	
57H	FD1L	
58H	FD1H	
59H	FD2L	
5AH	FD2H	
5BH	FD3L	
5CH	FD3H	
5DH	STMC0	
5EH	STMC1	
5FH	STMDL	
60H	STMDH	
61H	STMAL	
62H	STMAH	
63H	STMRP	
64H	CTM1C0	
65H	CTM1C1	
66H	CTM1DL	
67H	CTM1DH	
68H	CTM1AL	
69H	CTM1AH	
6AH	TSC0	
6BH	TSC1	
6CH	TSC2	
6DH	TSC3	
6EH		
6FH		
70H	PE	
71H	PEC	
72H	PEPU	
73H		
74H		
75H		
76H	USR	
77H	UCR1	
78H	UCR2	
79H	TXR_RXR	
7AH	BRG	
7BH		
≈	≈	≈
7FH		

: Unused, read as 00H

特殊功能数据存储器结构 – **BS66F340**

	Sector 0	Sector 1
00H	IAR0	TKTMR
01H	MP0	TKC0
02H	IAR1	TK16DL
03H	MP1L	TK16DH
04H	MP1H	TKC1
05H	ACC	TKM016DL
06H	PCL	TKM016DH
07H	TBLP	TKM0ROL
08H	TBLH	TKM0ROH
09H	TBHP	TKM0C0
0AH	STATUS	TKM0C1
0BH		TKM0C2
0CH	IAR2	TKM116DL
0DH	MP2L	TKM116DH
0EH	MP2H	TKM1ROL
0FH	RSTFC	TKM1ROH
10H	INTC0	TKM1C0
11H	INTC1	TKM1C1
12H	INTC2	TKM1C2
13H		TKM216DL
14H	PA	TKM216DH
15H	PAC	TKM2ROL
16H	PAPU	TKM2ROH
17H	PAWU	TKM2C0
18H	PB	TKM2C1
19H	PBC	TKM2C2
1AH	PBPU	TKM316DL
1BH	INTEG	TKM316DH
1CH	SCC	TKM3ROL
1DH	HIRCC	TKM3ROH
1EH	HXTC	TKM3C0
1FH	LXTC	TKM3C1
20H	LVDC	TKM3C2
21H	LVRC	TKM416DL
22H	WDTC	TKM416DH
23H	RSTC	TKM4ROL
24H	PC	TKM4ROH
25H	PCC	TKM4C0
26H	PCPU	TKM4C1
27H	PD	TKM4C2
28H	PDC	
29H	PDPU	
2AH	MFI0	
2BH	MFI1	
2CH	MFI2	
2DH	MFI3	
2EH	ADRL	
2FH	ADRH	
30H	ADCR0	
31H	ADCR1	
32H	PSCR0	
33H	TB0C	
34H	TB1C	
35H	SIMTOC	
36H	SIMC0	
37H	SIMC1	
38H	SIMD	
39H	SIMA/SIMC2	
3AH	CTM0C0	
3BH	CTM0C1	
3CH	CTM0DL	
3DH	CTM0DH	
3EH	CTM0AL	
3FH	CTM0AH	

	Sector 0	Sector 1
40H		EEC
41H	EEA	
42H		
43H	EED	
44H	PSCR1	IFS
45H		PAS0
46H	SLDEC	PAS1
47H		PBS0
48H	PTMC0	PBS1
49H	PTMC1	PCS0
4AH	PTMDL	PCS1
4BH	PTMDH	PDS0
4CH	PTMAL	PDS1
4DH	PTMAH	PES0
4EH	PTMRPL	PES1
4FH	PTMRPH	
50H	FC0	
51H	FC1	
52H	FC2	
53H	FARL	
54H	FARH	
55H	FD0L	
56H	FD0H	
57H	FD1L	
58H	FD1H	
59H	FD2L	
5AH	FD2H	
5BH	FD3L	
5CH	FD3H	
5DH	STMC0	
5EH	STMC1	
5FH	STMDL	
60H	STMDH	
61H	STMAL	
62H	STMAH	
63H	STMRP	
64H	CTM1C0	
65H	CTM1C1	
66H	CTM1DL	
67H	CTM1DH	
68H	CTM1AL	
69H	CTM1AH	
6AH	TSC0	
6BH	TSC1	
6CH	TSC2	
6DH	TSC3	
6EH		
6FH		
70H	PE	
71H	PEC	
72H	PEPU	
73H		
74H		
75H		
76H	USR	
77H	UCR1	
78H	UCR2	
79H	TXR_RXR	
7AH	BRG	
7BH		
≈	≈	≈
7FH		

: Unused, read as 00H

特殊功能数据存储器结构 – **BS66F350**

	Sector 0	Sector 1
00H	IAR0	TKTMR
01H	MP0	TKC0
02H	IAR1	TK16DL
03H	MP1L	TK16DH
04H	MP1H	TKC1
05H	ACC	TKM016DL
06H	PCL	TKM016DH
07H	TBLP	TKM0ROL
08H	TBLH	TKM0ROH
09H	TBHP	TKM0C0
0AH	STATUS	TKM0C1
0BH	PBP	TKM0C2
0CH	IAR2	TKM116DL
0DH	MP2L	TKM116DH
0EH	MP2H	TKM1ROL
0FH	RSTFC	TKM1ROH
10H	INTC0	TKM1C0
11H	INTC1	TKM1C1
12H	INTC2	TKM1C2
13H		TKM216DL
14H	PA	TKM216DH
15H	PAC	TKM2ROL
16H	PAPU	TKM2ROH
17H	PAWU	TKM2C0
18H	PB	TKM2C1
19H	PBC	TKM2C2
1AH	PBPU	TKM316DL
1BH	INTEG	TKM316DH
1CH	SCC	TKM3ROL
1DH	HIRCC	TKM3ROH
1EH	HXTC	TKM3C0
1FH	LXTC	TKM3C1
20H	LVDC	TKM3C2
21H	LVRC	TKM416DL
22H	WDTC	TKM416DH
23H	RSTC	TKM4ROL
24H	PC	TKM4ROH
25H	PCC	TKM4C0
26H	PCPU	TKM4C1
27H	PD	TKM4C2
28H	PDC	TKM516DL
29H	PDPU	TKM516DH
2AH	MFI0	TKM5ROL
2BH	MFI1	TKM5ROH
2CH	MFI2	TKM5C0
2DH	MFI3	TKM5C1
2EH	ADRL	TKM5C2
2FH	ADRH	TKM616DL
30H	ADCR0	TKM616DH
31H	ADCR1	TKM6ROL
32H	PSCR0	TKM6ROH
33H	TB0C	TKM6C0
34H	TB1C	TKM6C1
35H	SIMTOC	TKM6C2
36H	SIMC0	
37H	SIMC1	
38H	SIMD	
39H	SIMA/SIMC2	
3AH	CTM0C0	
3BH	CTM0C1	
3CH	CTM0DL	
3DH	CTM0DH	
3EH	CTM0AL	
3FH	CTM0AH	

	Sector 0	Sector 1
40H		EEC
41H	EEA	
42H		
43H	EED	
44H	PSCR1	IFS
45H		PAS0
46H	SLEDC	PAS1
47H		PBS0
48H	PTMC0	PBS1
49H	PTMC1	PCS0
4AH	PTMDL	PCS1
4BH	PTMDH	PDS0
4CH	PTMAL	PDS1
4DH	PTMAH	PES0
4EH	PTMRPL	PES1
4FH	PTMRPH	PFS0
50H	FC0	
51H	FC1	
52H	FC2	
53H	FARL	
54H	FARH	
55H	FD0L	
56H	FD0H	
57H	FD1L	
58H	FD1H	
59H	FD2L	
5AH	FD2H	
5BH	FD3L	
5CH	FD3H	
5DH	STMC0	
5EH	STMC1	
5FH	STMDL	
60H	STMDH	
61H	STMAL	
62H	STMAH	
63H	STMRP	
64H	CTM1C0	
65H	CTM1C1	
66H	CTM1DL	
67H	CTM1DH	
68H	CTM1AL	
69H	CTM1AH	
6AH	TSC0	
6BH	TSC1	
6CH	TSC2	
6DH	TSC3	
6EH		
6FH		
70H	PE	
71H	PEC	
72H	PEPU	
73H	PF	
74H	PFC	
75H	PFPU	
76H	USR	
77H	UCR1	
78H	UCR2	
79H	TXR_RXR	
7AH	BRG	
7BH		
≈	≈	≈
7FH		

: Unused, read as 00H

特殊功能数据存储器结构 – **BS66F360**

	Sector 0	Sector 1
00H	IAR0	TKTMR
01H	MP0	TKC0
02H	IAR1	TK16DL
03H	MP1L	TK16DH
04H	MP1H	TKC1
05H	ACC	TKM016DL
06H	PCL	TKM016DH
07H	TBLP	TKM0ROL
08H	TBLH	TKM0ROH
09H	TBHP	TKM0C0
0AH	STATUS	TKM0C1
0BH	PBP	TKM0C2
0CH	IAR2	TKM116DL
0DH	MP2L	TKM116DH
0EH	MP2H	TKM1ROL
0FH	RSTFC	TKM1ROH
10H	INTC0	TKM1C0
11H	INTC1	TKM1C1
12H	INTC2	TKM1C2
13H		TKM216DL
14H	PA	TKM216DH
15H	PAC	TKM2ROL
16H	PAPU	TKM2ROH
17H	PAWU	TKM2C0
18H	PB	TKM2C1
19H	PBC	TKM2C2
1AH	PBPU	TKM316DL
1BH	INTEG	TKM316DH
1CH	SCC	TKM3ROL
1DH	HIRCC	TKM3ROH
1EH	HXTC	TKM3C0
1FH	LXTC	TKM3C1
20H	LVDC	TKM3C2
21H	LVRC	TKM416DL
22H	WDTC	TKM416DH
23H	RSTC	TKM4ROL
24H	PC	TKM4ROH
25H	PCC	TKM4C0
26H	PCPU	TKM4C1
27H	PD	TKM4C2
28H	PDC	TKM516DL
29H	PDPU	TKM516DH
2AH	MFI0	TKM5ROL
2BH	MFI1	TKM5ROH
2CH	MFI2	TKM5C0
2DH	MFI3	TKM5C1
2EH	ADRL	TKM5C2
2FH	ADRH	TKM616DL
30H	ADCR0	TKM616DH
31H	ADCR1	TKM6ROL
32H	PSCR0	TKM6ROH
33H	TB0C	TKM6C0
34H	TB1C	TKM6C1
35H	SIMTOC	TKM6C2
36H	SIMC0	TKM716DL
37H	SIMC1	TKM716DH
38H	SIMD	TKM7ROL
39H	SIMA/SIMC2	TKM7ROH
3AH	CTM0C0	TKM7C0
3BH	CTM0C1	TKM7C1
3CH	CTM0DL	TKM7C2
3DH	CTM0DH	
3EH	CTM0AL	
3FH	CTM0AH	

	Sector 0	Sector 1
40H		EEC
41H	EEA	
42H		
43H	EED	
44H	PSCR1	IFS
45H		PAS0
46H	SLEDC	PAS1
47H	SLEDC1	PBS0
48H	PTMC0	PBS1
49H	PTMC1	PCS0
4AH	PTMDL	PCS1
4BH	PTMDH	PDS0
4CH	PTMAL	PDS1
4DH	PTMAH	PES0
4EH	PTMRPL	PES1
4FH	PTMRPH	PFS0
50H	FC0	
51H	FC1	PGS0
52H	FC2	PGS1
53H	FARL	
54H	FARH	
55H	FD0L	
56H	FD0H	
57H	FD1L	
58H	FD1H	
59H	FD2L	
5AH	FD2H	
5BH	FD3L	
5CH	FD3H	PH
5DH	STMC0	PHC
5EH	STMC1	PHPU
5FH	STMDL	
60H	STMDH	TKM816DL
61H	STMAL	TKM816DH
62H	STMAH	TKM8ROL
63H	STMRP	TKM8ROH
64H	CTM1C0	TKM8C0
65H	CTM1C1	TKM8C1
66H	CTM1DL	TKM8C2
67H	CTM1DH	
68H	CTM1AL	
69H	CTM1AH	
6AH	TSC0	
6BH	TSC1	
6CH	TSC2	
6DH	TSC3	
6EH		
6FH		
70H	PE	
71H	PEC	
72H	PEPU	
73H	PF	
74H	PFC	
75H	PFPU	
76H	USR	
77H	UCR1	
78H	UCR2	
79H	TXR_RXR	
7AH	BRG	
7BH	PG	
7CH	PGC	
7DH	PGPU	
7EH		
7FH		

: Unused, read as 00H

特殊功能数据存储器结构 – **BS66F370**

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，但其并没有实际的物理地址。间接寻址的方法准许使用间接寻址指针做数据操作，以取代定义实际存储器地址的直接存储器寻址方法。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。直接寻址通过相关的数据存储器寻址指令来访问所有的数据 Sector。

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

- **Example 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:
    :
```

- **Example 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。

程序存储区指针 – PBP

BS66F360 程序存储器被分为两个 Banks，BS66F370 程序存储器被分为四个 Banks，可以通过设置程序存储区指针 PBP 来访问不同的程序存储区。PBP 寄存器应在单片机使用“JMP”或“CALL”指令执行“分支”操作前正确地配置。在指令执行后会跳转到由程序存储区指针所选 bank 的一个非连续的程序存储器地址。

PBP 寄存器 – BS66F360

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	PBP0
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 **D7~D1**：一般数据位，可读 / 可写

Bit 0 **PBP0**：程序存储区选择位
0：Bank 0
1：Bank 1

PBP 寄存器 – BS66F370

Bit	7	6	5	4	3	2	1	0
Name	D6	D5	D4	D3	D2	D1	PBP1	PBP0
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 7~2 **D7~D1**：一般数据位，可读 / 可写

Bit 1~0 **PBP1~PBP0**：程序存储区选择位
00：Bank 0
01：Bank 1
10：Bank 2
11：Bank 3

累加器 – ACC

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

程序计数器低字节寄存器 – PCL

为了提供额外的程序控制功能，程序计数器低字节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。

表格寄存器 – TBLP, TBHP, TBLH

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

状态寄存器 – STATUS

这 8 位的状态寄存器由 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。

SC、CZ、Z、OV、AC 和 C 标志位通常反映最近运算的状态。

- SC：当 OV 与当前指令操作结果 MSB 执行“XOR”所得结果。
- 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。
- 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。

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	R	R	R	R/W	R/W	R/W	R/W
POR	x	x	0	0	x	x	x	x

“x”为未知

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。

EEPROM 数据存储器

该系列单片机内建 EEPROM 数据存储器。“Electrically Erasable Programmable Read Only Memory”为电可擦可编程只读存储器，由于其非易失的存储结构，即使在电源掉电的情况下存储器内的数据仍然保存完好。这种存储区扩展了 ROM 空间，对设计者来说增加了许多新的应用机会。EEPROM 可以用来存储产品编号、校准值、用户特定数据、系统配置参数或其它产品信息等。EEPROM 的数据读取和写入过程也会变的更简单。

单片机型号	容量	地址
BS66F340	128×8	00H~7FH
BS66F350		
BS66F360		
BS66F370		

EEPROM 数据存储器结构

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

EEPROM 寄存器

有三个寄存器控制内部 EEPROM 数据存储器总的操作。地址寄存器 EEA、数据寄存器 EED 及控制寄存器 EEC。EEA 和 EED 位于 Sector 0 中，它们能像其它特殊功能寄存器一样直接被访问。EEC 位于 Sector 1 中，可以通过 MP1L/MP1H 和 IAR1 或 MP2L/MP2H 和 IAR2 进行间接读取或写入。由于 EEC 控制寄存器位于 Sector 1 中的“40H”，在 EEC 寄存器上的任何操作被执行前，MP1L 或 MP2L 必须先设为“40H”，MP1H 或 MP2H 被设为“01H”。

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

EEPROM 寄存器列表

EEA 寄存器

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

Bit 7　　未定义，读为“0”

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

EED 寄存器

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

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

EEC 寄存器

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

Bit 7~4 未定义，读为“0”

Bit 3 **WREN**：数据 EEPROM 写使能位
0：除能
1：使能
此位为数据 EEPROM 写使能位，向数据 EEPROM 写操作之前需将此位置高。将此位清零时，则禁止向数据 EEPROM 写操作。

Bit 2 **WR**：EEPROM 写控制位
0：写周期结束
1：写周期有效
此位为数据 EEPROM 写控制位，由应用程序将此位置高将激活写周期。写周期结束后，硬件自动将此位清零。当 WREN 未先置高时，此位置高无效。

Bit 1 **RDEN**：数据 EEPROM 读使能位
0：除能
1：使能
此位为数据 EEPROM 读使能位，向数据 EEPROM 读操作之前需将此位置高。将此位清零时，则禁止向数据 EEPROM 读操作。

Bit 0 **RD**：EEPROM 读控制位
0：读周期结束
1：读周期有效
此位为数据 EEPROM 读控制位，由应用程序将此位置高将激活读周期。读周期结束后，硬件自动将此位清零。当 RDEN 未首先置高时，此位置高无效。

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

从 EEPROM 中读取数据

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

写数据到 EEPROM

写数据至 EEPROM，EEPROM 中写入数据的地址要先放入 EEA 寄存器中，写入的数据需存入 EED 寄存器中。EEC 寄存器中的写使能位 WREN 先置为高以使能写功能，然后 EEC 寄存器中的 WR 位需立即置高以开始写操作，这两条指令必须连续执行。总中断位 EMI 在写周期开始前应当被清零，写周期开始后再将其使能。若 WR 位已置为高而 WREN 位还未被设置则不能开始写操作。由于控制 EEPROM 写周期是一个内部时钟，与单片机的系统时钟异步，所以数据写入 EEPROM 的时间将有所延迟。可通过轮询 EEC 寄存器中的 WR 位或判断 EEPROM 写中断以侦测写周期是否完成。若写周期完成，WR 位将自动清除为“0”，通知用户数据已写入 EEPROM。因此，应用程序将轮询 WR 位以确定写周期是否结束。

写保护

防止误写入的写保护有以下几种。单片机上电后控制寄存器中的写使能位将被清除以杜绝任何写入操作。上电后存储器指针高字节寄存器 MP1H 及 MP2H 将重置为“0”，这意味着数据存储器 Sector 0 被选中。由于 EEPROM 控制寄存器位于 Sector 1 中，这增加了对写操作的保护措施。在正常程序操作中确保控制寄存器中的写使能位被清除将能防止不正确的写操作。

EEPROM 中断

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

编程注意事项

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

WREN 位置位后，EEC 寄存器中的 WR 位需立即置位，以确保写周期正确地执行。写周期执行前总中断位 EMI 应先清零，写周期开始执行后再将此位重新使能。注意，单片机不应在 EEPROM 读或写操作完全完成之前进入空闲或休眠模式，否则 EEPROM 读或写操作将失败。

程序举例

- **从 EEPROM 中读取数据 – 轮询法**

```
MOV A, EEPROM_ADRES              ; user defined address
MOV EEA, A
MOV A, 040H                      ; setup memory pointer low byte MP1L
MOV MP1L, A                      ; MP1L points to EEC register
MOV A, 01H                       ; setup Memory Pointer high byte MP1H
MOV MP1H, A
SET IAR1.1                       ; set RDEN bit, enable read operations
SET IAR1.0                       ; start Read Cycle - set RD bit
BACK:
SZ  IAR1.0                       ; check for read cycle end
JMP BACK
CLR IAR1                         ; disable EEPROM write
CLR MP1H
MOV A, EED                       ; move read data to register
MOV READ_DATA, A
```

- **写数据到 EEPROM – 轮询法**

```
MOV A, EEPROM_ADRES              ; user defined address
MOV EEA, A
MOV A, EEPROM_DATA               ; user defined data
MOV EED, A
MOV A, 040H                      ; setup memory pointer low byte MP1L
MOV MP1L, A                      ; MP1L points to EEC register
MOV A, 01H                       ; setup Memory Pointer high byte MP1H
MOV MP1H, A
CLR EMI
SET IAR1.3                       ; set WREN bit, enable write operations
SET IAR1.2                       ; start Write Cycle - set WR bit
SET EMI
BACK:
SZ  IAR1.2                       ; check for write cycle end
JMP BACK
CLR IAR1                         ; disable EEPROM write
CLR MP1H
```

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。振荡器选择是通过应用程序和相关的控制寄存器共同完成的。

振荡器概述

振荡器除了作为系统时钟源，还作为看门狗定时器和时基中断的时钟源。外部振荡器需要一些外围器件，而集成的内部振荡器不需要任何外围器件。它们提供的高速和低速系统振荡器具有较宽的频率范围。所有振荡器选择通过寄存器选择。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。

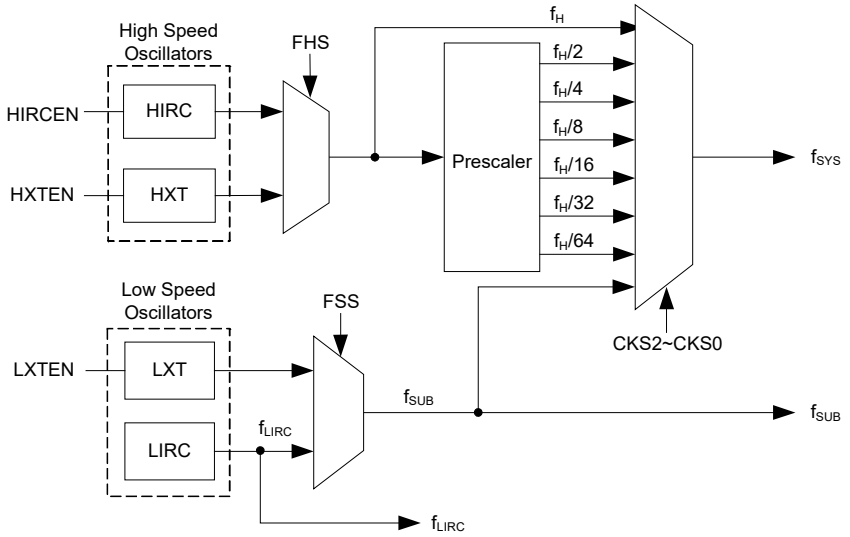
类型	名称	频率	引脚
外部高速晶振	HXT	400kHz~20MHz	OSC1/OSC2
内部高速 RC	HIRC	8/12/16MHz	—
外部低速晶振	LXT	32.768kHz	XT1/XT2
内部低速 RC	LIRC	32kHz	—

振荡器类型

系统时钟配置

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

低速振荡器的实际时钟源由 SCC 寄存器的 FSS 位选择，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。

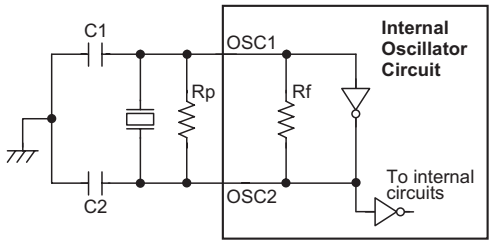


系统时钟配置

外部晶体 / 陶瓷振荡器 – HXT

外部高频晶体 / 陶瓷振荡器是一个高频振荡器，为上电后的默认振荡器时钟源。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，而不需其它外部电容。为保证某些低频率的晶体振荡和陶瓷谐振器的振荡频率更精准，建议连接两个小容量电容 C1 和 C2 到 VSS，具体数值与客户选择的晶体 / 陶瓷晶振有关。

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



Note: 1. Rp 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

晶体振荡器 C1 和 C2 值		
晶体频率	C1	C2
12MHz	0pF	0pF
8MHz	0pF	0pF
4MHz	0pF	0pF
1MHz	100pF	100pF
注：C1 和 C2 数值仅作参考用		

晶体振荡器电容推荐值

内部 RC 振荡器 – HIRC

内部 RC 振荡器是一个集成的系统振荡器，不需其它外部器件。内部 RC 振荡器具有三种固定的频率：8MHz，12MHz，16MHz。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 V_{DD}、温度以及芯片制成工艺不同的影响较大程度地降低。在电源电压为 3V 或 5V 及温度为 25℃ 的条件下，12MHz 频率的容差为 2%。如果选择了该内部时钟，无需额外的引脚；I/O 引脚可以作为通用 I/O 口使用。

外部 32.768kHz 晶体振荡器 – LXT

外部 32.768kHz 晶体振荡器是一个低频振荡器，由 FSS 控制位选择。时钟频率固定为 32.768kHz，此时 XT1 和 XT2 间引脚必须连接 32.768kHz 的晶体振荡器。需要外部电阻和电容连接到 32.768kHz 晶振以帮助起振。对于那些要求精确频率的场合中，可能需要这些元件来对由制程产生的误差提供频率补偿。LXTEN 位置高使能 LXT 振荡器后，LXT 振荡器启动需要一定的延时。

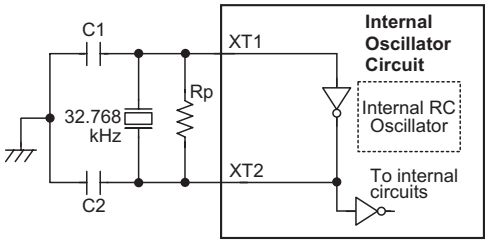
当系统进入空闲 / 休眠模式，系统时钟关闭以降低功耗。然而在某些应用，比如空闲 / 休眠模式下要保持内部定时器功能，必须提供额外的时钟，且与系统时钟无关。

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

引脚共用的软件控制位决定 XT1/XT2 脚是用于 LXT 还是作为普通 I/O 口或其它共用功能使用。

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

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



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

外部 LXT 振荡器

LXT 振荡器低功耗功能

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

LXTSP 位	LXT 工作模式
0	低功耗
1	快速启动

LXTSP 位置高会使能 LXT 快速启动模式。在快速启动模式，LXT 振荡器将起振并快速稳定下来。LXT 振荡器完全起振后，可以通过将 LXTSP 位清零进入低功耗模式。振荡器可以继续运行，其间耗电将少于快速启动模式。需要注意的是，在选择 LXT 振荡器时钟作为系统时钟源之前，必须适当地控制 LXT 工作模式的切换。一旦通过设置 SCC 寄存器中的 CKS2~CKS0 位和 FSS 位选择了 LXT 振荡器时钟作为系统时钟源，LXT 振荡器工作模式将不能改变。

应注意的是，无论 LXTSP 位是什么值，LXT 振荡器都会正常运作，不同的只是在低功耗模式时需要的启动时间更长。

内部 32kHz 振荡器 – LIRC

内部 32kHz 振荡器可通过 FSS 控制位选择作为系统低频振荡器。它是一个完全集成 RC 振荡器，它在 5V 电压下运行的典型频率值为 32kHz 且无需外部元件。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡器因电源电压、温度及芯片制成工艺不同的影响较大程度地降低。因此，内部 32kHz 振荡器频率在 25℃温度 5V 电压下的精度保持在 10% 以内。

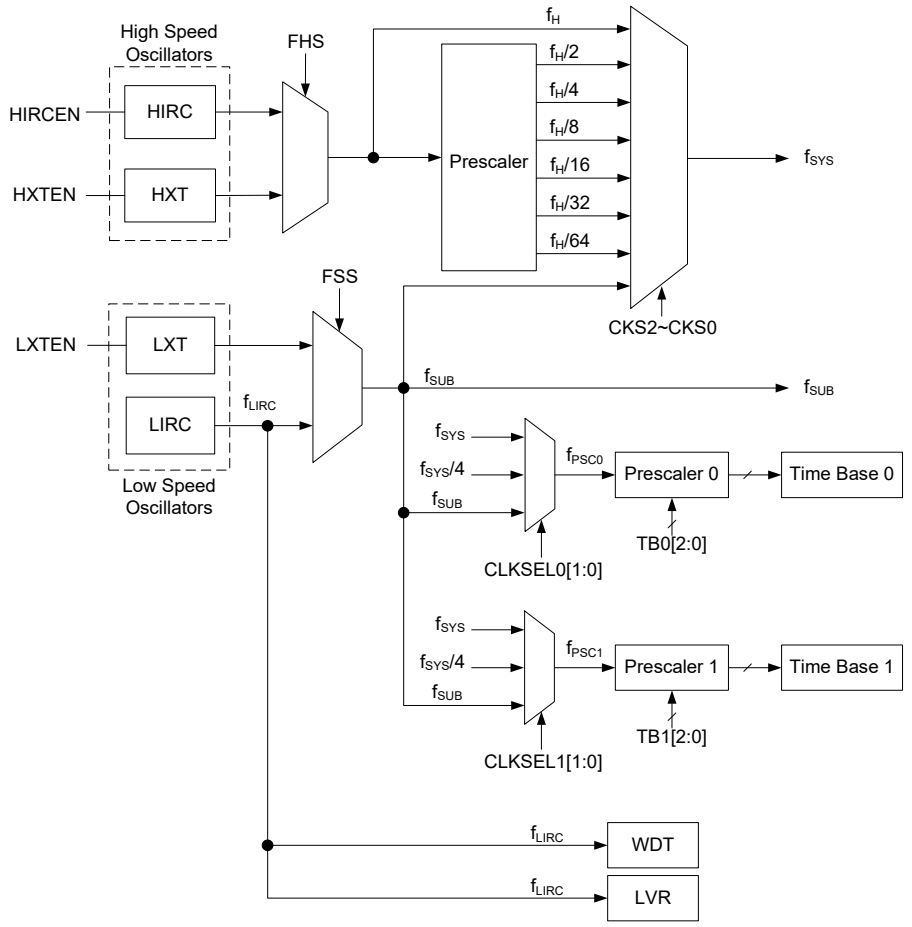
工作模式和系统时钟

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。低频系统时钟源来自 f_{SUB}，若 f_{SUB} 被选择，低频时钟来自 LXT 或 LIRC 振荡器，可通过 SCC 寄存器中的 FSS 位选择。其它系统时钟还有高速系统振荡器的分频 $f_H/2$~$f_H/64$。



驱动时钟配置

注：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。

系统工作模式

单片机有 6 种不同的工作模式，每种有它自身的特性，根据应用中不同的性能和功耗要求可选择不同的工作模式。单片机正常工作有两种模式：正常模式和低速模式。剩余的 4 种工作模式：休眠模式、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	CPU	寄存器设置			f_{SYS}	f_H	f_{SUB}	f_{LIRC}
		FHIDEN	FSIDEN	CKS2~CKS0				
正常模式	On	x	x	000~110	f_H~f_H/64	On	On	On
低速模式	On	x	x	111	f_{SUB}	On/Off(1)	On	On
空闲模式 0	Off	0	1	000~110	Off	Off	On	On
				111	On			
空闲模式 1	Off	1	1	xxx	On	On	On	On
空闲模式 2	Off	1	0	000~110	On	On	Off	On
				111	Off			
休眠模式	Off	0	0	xxx	Off	Off	Off	On(2)

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. 因 WDT 功能始终使能，f_{LIRC} 保持开启。

正常模式

顾名思义，这是主要的工作模式之一，单片机的所有功能均可在此模式中实现且系统时钟由一个高速振荡器提供。该模式下单片机正常工作的时钟源来自 HXT 或 HIRC 振荡器。高速振荡器频率可被分为 1~64 的不等比率，实际的比率由 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。该低速时钟源可来自 f_{SUB}，而 f_{SUB} 可来自于 LXT 或 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 0 时，系统进入休眠模式。在休眠模式中，CPU 停止运行。然而因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。

控制寄存器

寄存器 SCC、HIRCC、HXTC 和 LXTC 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	FSS	FHIDEN	FSIDEN
HIRCC	—	—	—	—	HIRC1	HIRC0	HIRCF	HIRCEN
HXTC	—	—	—	—	—	HXTM	HXTF	HXTEN
LXTC	—	—	—	—	—	LXTSP	LXTF	LXTEN

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	FSS	FHIDEN	FSIDEN
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 7~5 **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
1：HXT

Bit 2 **FSS**：低频时钟选择位

0：LIRC
1：LXT

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否停止。

Bit 0 **FSIDEN**：CPU 关闭时低频振荡器控制位

0：除能
1：使能

此位用来控制在执行 HALT 指令 CPU 关闭后低速振荡器是否停止。若 LIRC 被选择作为低速振荡器时钟源，则 LIRC 振荡器是由该位与 WDT 功能控制位共同控制的。但因 WDT 功能始终使能，即使该位被清零，LIRC 振荡器也将继续运行。

HIRCC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	HIRC1	HIRC0	HIRCF	HIRCEN
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	1

Bit 7~4 未定义，读为“0”

Bit 3~2 **HIRC1~HIRC0**：HIRC 频率选择位

00：8MHz
01：12MHz
10：16MHz
11：8MHz

当 HIRC 振荡器使能或通过应用程序改变 HIRC 频率选择位时，在 HIRCF 标志位置高后时钟频率会自动改变。

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，或者通过应用程序改变 HIRC 频率选择位时，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	—	—	—	—	—	HXTM	HXTF	HXTEN
R/W	—	—	—	—	—	R/W	R	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 未定义，读为“0”

Bit 2 **HXTM**：HXT 模式选择位

0：HXT 频率≦ 10MHz
1：HXT 频率＞ 10MHz

此位用于选择 HXT 振荡器的工作模式。注意，此位必须在 HXT 使能前正确地配置，在 HXTEN 位置高 HXT 振荡器使能后改变此位的值将无效。

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

LXTC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	LXTSP	LXTF	LXTEN
R/W	—	—	—	—	—	RW	R	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3　未定义，读为“0”

Bit 2　**LXTSP**：LXT 振荡器快速启动控制位
0：除能 – 低功耗模式
1：使能 – 快速启动模式
此位用来控制 LXT 振荡器工作在低功耗模式或快速启动模式。当 LXTSP 位被置高，LXT 振荡器的振荡加快，但功耗增加。如果 LXTSP 位被清零，LXT 振荡器功耗将减少，但需要较长时间才能稳定下来。需要注意的是，通过设置 SCC 寄存器中的 CKS2~CKS0 位和 FSS 位选择 LXT 振荡器作为系统时钟源后，该位不能改变。

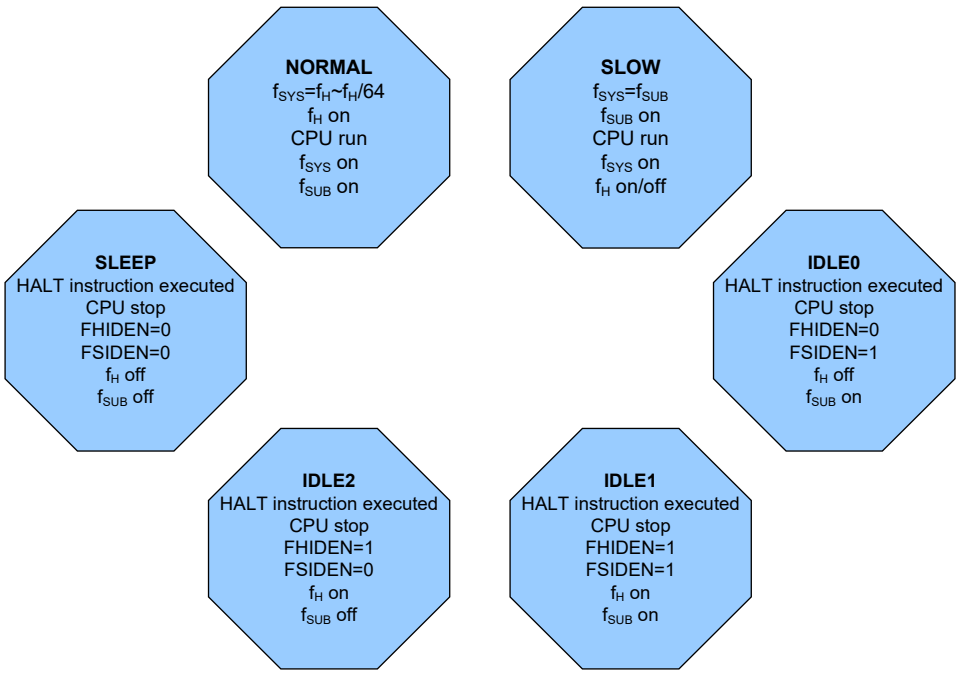
Bit 1　**LXTF**：LXT 振荡器稳定标志位
0：LXT 不稳定
1：LXT 稳定
此位用于表明 LXT 振荡器是否稳定。LXTEN 位置高使能 LXT 振荡器后，LXTF 位会先被清零，在 LXT 稳定后会被置高。

Bit 0　**LXTEN**：LXT 振荡器使能控制位
0：除能
1：使能

工作模式切换

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

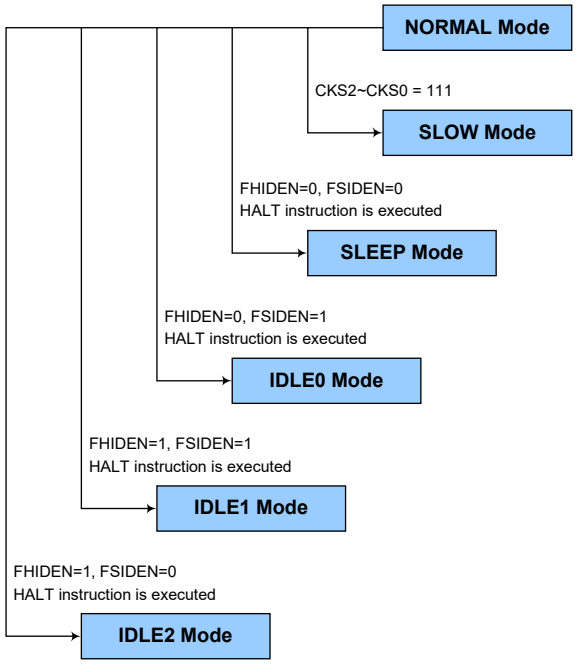
简单来说，正常模式和低速模式间的切换仅需设置 SCC 寄存器中的 CKS2~CKS0 位即可实现，而正常模式 / 低速模式与休眠模式 / 空闲模式间的切换经由 HALT 指令实现。当 HALT 指令执行后，单片机是否进入空闲模式或休眠模式由 SCC 寄存器中的 FHIDEN 和 FSIDEN 位决定的。



正常模式切换到低速模式

系统运行在正常模式时使用高速系统振荡器，因此较为耗电。可通过设置 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。

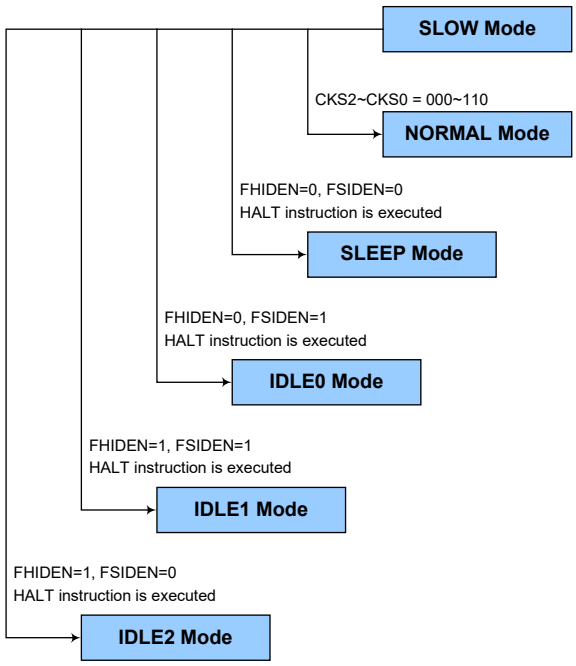
低速模式的时钟源来自 LXT 或 LIRC 振荡器，由 SCC 寄存器中的 FSS 位确定，因此要求这些振荡器在所有模式切换动作发生前稳定下来。



低速模式切换到正常模式

在低速模式时系统时钟来自 f_{SUB}。切换回正常模式时，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正常模式时，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指定在交流电气特性中。



进入休眠模式

进入休眠模式的方法仅有一种——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”。在这种模式下，除了 WDT 以外的所有时钟和功能都将关闭。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。
- 因 WDT 功能始终使能，WDT 将被清零并重新开始计数。

进入空闲模式 0

进入空闲模式 0 的方法仅有一种——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”。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。
- 因 WDT 功能始终使能，WDT 将被清零并重新开始计数。

进入空闲模式 1

进入空闲模式 1 的方法仅有一种——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”。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。
- 因 WDT 功能始终使能，WDT 将被清零并重新开始计数。

进入空闲模式 2

进入空闲模式 2 的方法仅有一种——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”。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。
- 因 WDT 功能始终使能，WDT 将被清零并重新开始计数。

静态电流的注意事项

由于单片机进入休眠或空闲模式的主要原因是将 MCU 的电流降低到尽可能低，可能到只有几个微安的级别（空闲模式 1 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，额外的静态电流也可能会有几百微安。

唤醒

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

系统进入休眠或空闲模式之后，可以通过以下几种方式唤醒：

- PA 口下降沿
- 系统中断
- WDT 溢出

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

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 定时器时钟源由内部 RC 振荡器 LIRC 提供。电压为 5V 时内部振荡器 LIRC 的频率大约为 32kHz，这个特殊的内部时钟周期随 V_{DD}、温度和制成的不同而变化。看门狗定时器的时钟源可分频为 2^8~2^{18} 以提供更大的溢出周期，分频比由 WDTC 寄存器中的 WS2~WS0 位来决定。

看门狗定时器控制寄存器

WDTC 寄存器用于控制 WDT 功能的使能 / 除能及选择溢出周期。该寄存器控制看门狗定时器的所有操作。

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
01010B/10101B：使能
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 2~3 个 LIRC 时钟周期后，且 RSTFC 寄存器的 WRF 位将置为“1”。

Bit 2~0 **WS2~WS0**：WDT 溢出周期选择位
000：$2^8/f_{LIRC}$
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”为未知

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
具体描述见其它章节。

Bit 2 **LVRF**：LVR 复位标志位
具体描述见其它章节。

Bit 1 **LRF**：LVR 控制寄存器软件复位标志位
具体描述见其它章节。

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，这些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 中有五位 WE4~WE0 可提供使能控制以及看门狗定时器复位操作。当 WE4~WE0 设置为“10101B”和“01010B”时使能 WDT 功能。如果 WE4~WE0 设置为除“01010B”和“10101B”以外的值时，单片机将在 2~3 个 f_{LIRC} 时钟周期后复位。上电后这些位初始化为“01010B”。

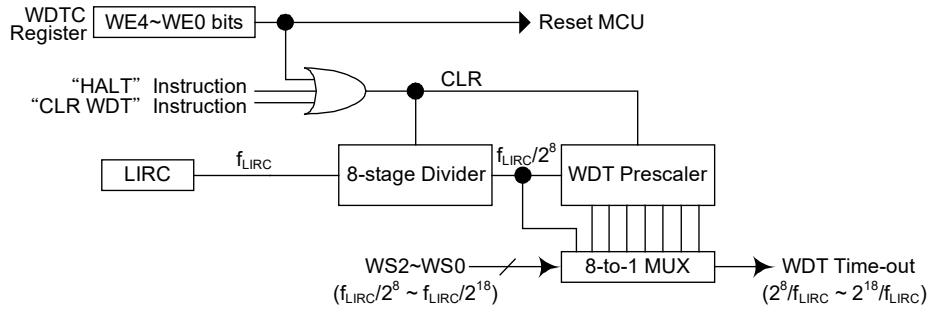
WE4~WE0 位	WDT 功能
10101B/01010B	使能
其它值	复位单片机

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 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 7.8ms。



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。

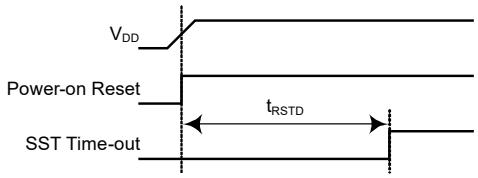
另一种复位为低电压复位即 LVR 复位，在电源供应电压低于 LVR 设定值时，系统会产生 LVR 复位。另一种复位为看门狗溢出单片机复位，不同方式的复位操作会对寄存器产生不同的影响。

复位功能

包括内部和外部事件触发复位，单片机共有五种复位方式：

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。



注：t_{RSTD} 为上电延迟时间，典型值为 50ms。

上电复位时序图

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 2~3 个 f_{LIRC} 时钟周期后发生复位。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 2~3 个 f_{LIRC} 时钟周期后，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”为未知

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。

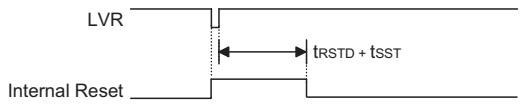
Bit 2 **LVRF**：LVR 复位标志位
具体描述见其它章节。

Bit 1 **LRF**：LVR 控制寄存器软件复位标志位
具体描述见其它章节。

Bit 0 **WRF**：WDT 控制寄存器软件复位标志位
具体描述见其它章节。

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。低电压复位功能总是使能于特定的电压值，V_{LVR}。例如在更换电池的情况下，单片机供应的电压可能会在 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。V_{LVR} 参数值可通过 LVRC 寄存器中的 LVS7~LVS0 位设置。若由于受到干扰 LVS7~LVS0 变为其它值时，需经过 2~3 个 f_{LIRC} 周期响应复位。此时 RSTFC 寄存器的 LRF 位被置位。上电后寄存器的值为 01010101B。正常执行时 LVR 会于休眠或空闲时自动除能关闭。



注：t_{RSTD} 为上电延迟时间，典型值为 50ms。

低电压复位时序图

- **LVRC 寄存器**

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

Bit 7~0 **LVS7~LVS0**：LVR 电压选择
01010101：2.1V
00110011：2.55V
10011001：3.15V
10101010：3.8V
其它值：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。需要经过 2~3 个 f_{LIRC} 时钟周期响应复位。此时复位后的寄存器内容保持不变。
除了以上定义的四个低电压复位值外，其它值也能导致单片机复位。需要经过 2~3 个 f_{LIRC} 时钟周期响应复位。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”为未知

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
具体描述见其它章节。

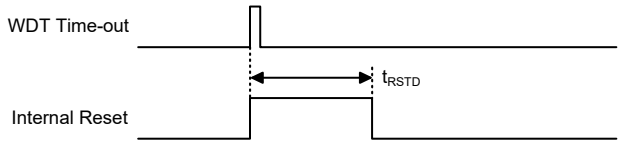
Bit 2 **LVRF**：LVR 复位标志位
0：未发生
1：发生
当特定的低电压复位条件发生时，此位被置为“1”，且只能通过应用程序清零。

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
具体描述见其它章节。

正常运行时看门狗溢出复位

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

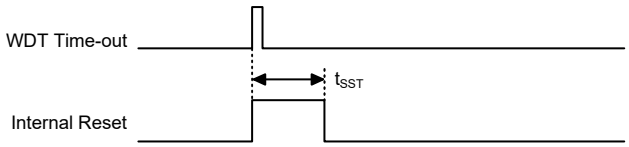


注：t_{RSTD} 为上电延迟时间，典型值为 16.7ms。

正常运行时看门狗溢出复位时序图

休眠或空闲时看门狗溢出复位

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



休眠或空闲时看门狗溢出复位时序图

复位初始状态

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

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

“u”代表不改变

在单片机上电复位之后，各功能单元初始化的情形，列于下表。

项目	复位后情况
程序计数器	清除为零
中断	所有中断被除能
看门狗定时器，时基	清零，WDT 清除并重新计数
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	BS66F340	BS66F350	BS66F360	BS66F370	上电复位	LVR 复位 (正常模式)	WDT 溢出 (正常模式)	WDT 溢出 (空闲 / 休眠模式)
IAR0	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP0	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
IAR1	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1L	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1H	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
ACC	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	●				---- xxxx	---- uuuu	---- uuuu	---- uuuu
TBHP		●			---x xxxx	---u uuuu	---u uuuu	---u uuuu
TBHP			●		--xx xxxx	--uu uuuu	--uu uuuu	--uu uuuu
TBHP				●	-xxx xxxx	-uuu uuuu	-uuu uuuu	-uuu uuuu
STATUS	●	●	●	●	xx00 xxxx	uuuu uuuu	xx1u uuuu	uu11 uuuu
PBP			●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
IAR2	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP2L	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2H	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
RSTFC	●	●	●	●	---- 0x00	---- u1uu	---- 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
INTEG	●	●	●	●	---- 0000	---- 0000	---- 0000	---- uuuu
SCC	●	●	●	●	000- 0000	000- 0000	000- 0000	uuu- uuuu
HIRCC	●	●	●	●	---- 0001	---- 0001	---- 0001	---- uuuu
HXTC	●	●	●	●	---- -000	---- -000	---- -000	---- -uuu
LXTC	●	●	●	●	---- -000	---- -000	---- -000	---- -uuu
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
PC	●				---- 1111	---- 1111	---- 1111	---- uuuu

寄存器	BS66F340	BS66F350	BS66F360	BS66F370	上电复位	LVR 复位 (正常模式)	WDT 溢出 (正常模式)	WDT 溢出 (空闲 / 休眠模式)
PC		●	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCC	●				---- 1111	---- 1111	---- 1111	---- uuuu
PCC		●	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCPU	●				---- 0000	---- 0000	---- 0000	---- uuuu
PCPU		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD		●	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC		●	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDPU		●	●	●	0000 0000	0000 0000	0000 0000	0uuuu uuuu
MFI0	●	●	●	●	--00 --00	--00 --00	--00 --00	--uu --uu
MFI1	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI2	●	●	●	●	-000 -000	-000 -000	-000 -000	-uuu -uuu
MFI3	●	●	●	●	--00 --00	--00 --00	--00 --00	--uu --uu
ADRL (ADRFS=0)	●	●	●	●	xxxx ----	xxxx ----	xxxx ----	uuuu ----
ADRL (ADRFS=1)	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
ADRH (ADRFS=0)	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
ADRH (ADRFS=1)	●	●	●	●	---- xxxx	---- uuuu	---- uuuu	---- uuuu
ADCR0	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADCR1	●	●	●	●	0-00 -000	0-00 -000	0-00 -000	u-uu -uuu
PSCR0	●	●	●	●	---- --00	---- --00	---- --00	---- --uu
TB0C	●	●	●	●	0--- -000	0--- -000	0--- -000	u--- -uuu
TB1C	●	●	●	●	0--- -000	0--- -000	0--- -000	u--- -uuu
SIMTOC	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
SIMC0	●	●	●	●	111- 0000	111- 0000	111- 0000	uuu- uuuu
SIMC1	●	●	●	●	1000 0001	1000 0001	1000 0001	uuuu uuuu
SIMD	●	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SIMA/SIMC2	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
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
EEA	●	●	●	●	-000 0000	-000 0000	-000 0000	-uuu uuuu
EED	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PSCR1	●	●	●	●	---- --00	---- --00	---- --00	---- --uu
SLEDC	●				--00 0000	--00 0000	--00 0000	--uu uuuu

寄存器	BS66F340	BS66F350	BS66F360	BS66F370	上电复位	LVR 复位 (正常模式)	WDT 溢出 (正常模式)	WDT 溢出 (空闲 / 休眠模式)
SLEDC		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
SLEDC1				●	---- 0000	---- 0000	---- 0000	---- uuuu
PTMC0	●	●	●	●	0000 0---	0000 0---	0000 0---	uuuu u---
PTMC1	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDH	●	●	●	●	---- --00	---- --00	---- --00	---- --uu
PTMAL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMAH	●	●	●	●	---- --00	---- --00	---- --00	---- --uu
PTMRPL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMRPH	●	●	●	●	---- --00	---- --00	---- --00	---- --uu
FC0	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC1	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FC2		●	●	●	---- ---0	---- ---0	---- ---0	---- ---u
FARL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FARH	●				---- 0000	---- 0000	---- 0000	---- uuuu
FARH		●			---0 0000	---0 0000	---0 0000	---u uuuu
FARH			●		--00 0000	--00 0000	--00 0000	--uu uuuu
FARH				●	-000 0000	-000 0000	-000 0000	-uuu uuuu
FD0L	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
FD0H	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
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
STMC0	●	●	●	●	0000 0---	0000 0---	0000 0---	uuuu u---
STMC1	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDH	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAH	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMRP	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
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

寄存器	BS66F340	BS66F350	BS66F360	BS66F370	上电复位	LVR 复位 (正常模式)	WDT 溢出 (正常模式)	WDT 溢出 (空闲 / 休眠模式)
TSC0	●	●	●	●	010- ----	010- ----	010- ----	uuu- ----
TSC1	●	●	●	●	000- ----	000- ----	000- ----	uuu- ----
TSC2		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TSC3	●	●	●	●	--0- ----	--0- ----	--0- ----	--u- ----
PE	●				--11 1111	--11 1111	--11 1111	--uu uuuu
PE		●	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEC	●				--11 1111	--11 1111	--11 1111	--uu uuuu
PEC		●	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PEPU	●				--00 0000	--00 0000	--00 0000	--uu uuuu
PEPU		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PF			●	●	--11 1111	--11 1111	--11 1111	--uu uuuu
PFC			●	●	--11 1111	--11 1111	--11 1111	--uu uuuu
PFPU			●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
USR	●	●	●	●	0000 1011	0000 1011	0000 1011	uuuu uuuu
UCR1	●	●	●	●	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
UCR2	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TXR_RXR	●	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
BRG	●	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PG				●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGC				●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PGPU				●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKTMR	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKC0	●	●	●	●	0000 0-00	0000 0-00	0000 0-00	uuuu u-uu
TK16DL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TK16DH	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKC1	●	●	●	●	0000 0011	0000 0011	0000 0011	uuuu uuuu
TKM016DL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM016DH	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM0ROL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM0ROH	●	●	●	●	---- --00	---- --00	---- --00	---- --uu
TKM0C0	●	●	●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
TKM0C1	●	●	●	●	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
TKM0C2	●	●	●	●	1110 0100	1110 0100	1110 0100	uuuu uuuu
TKM116DL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM116DH	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM1ROL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM1ROH	●	●	●	●	---- --00	---- --00	---- --00	---- --uu
TKM1C0	●	●	●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
TKM1C1	●	●	●	●	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu

寄存器	BS66F340	BS66F350	BS66F360	BS66F370	上电复位	LVR 复位 (正常模式)	WDT 溢出 (正常模式)	WDT 溢出 (空闲 / 休眠模式)
TKM1C2	●	●	●	●	1110 0100	1110 0100	1110 0100	uuuu uuuu
TKM216DL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM216DH	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM2ROL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM2ROH	●	●	●	●	---- --00	---- --00	---- --00	---- --uu
TKM2C0	●	●	●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
TKM2C1	●	●	●	●	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
TKM2C2	●	●	●	●	1110 0100	1110 0100	1110 0100	uuuu uuuu
TKM316DL		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM316DH		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM3ROL		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM3ROH		●	●	●	---- --00	---- --00	---- --00	---- --uu
TKM3C0		●	●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
TKM3C1		●	●	●	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
TKM3C2		●	●	●	1110 0100	1110 0100	1110 0100	uuuu uuuu
TKM416DL		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM416DH		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM4ROL		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM4ROH		●	●	●	---- --00	---- --00	---- --00	---- --uu
TKM4C0		●	●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
TKM4C1		●	●	●	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
TKM4C2		●	●	●	1110 0100	1110 0100	1110 0100	uuuu uuuu
TKM516DL			●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM516DH			●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM5ROL			●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM5ROH			●	●	---- --00	---- --00	---- --00	---- --uu
TKM5C0			●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
TKM5C1			●	●	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
TKM5C2			●	●	1110 0100	1110 0100	1110 0100	uuuu uuuu
TKM616DL			●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM616DH			●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM6ROL			●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM6ROH			●	●	---- --00	---- --00	---- --00	---- --uu
TKM6C0			●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
TKM6C1			●	●	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
TKM6C2			●	●	1110 0100	1110 0100	1110 0100	uuuu uuuu
TKM716DL				●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM716DH				●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM7ROL				●	0000 0000	0000 0000	0000 0000	uuuu uuuu

寄存器	BS66F340	BS66F350	BS66F360	BS66F370	上电复位	LVR 复位 (正常模式)	WDT 溢出 (正常模式)	WDT 溢出 (空闲 / 休眠模式)
TKM7ROH				●	---- --00	---- --00	---- --00	---- --uu
TKM7C0				●	--00 0000	--00 0000	--00 0000	--uu uuuu
TKM7C1				●	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
TKM7C2				●	1110 0100	1110 0100	1110 0100	uuuu uuuu
EEC	●	●	●	●	---- 0000	---- 0000	---- 0000	---- uuuu
IFS	●	●	●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
PAS0	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS1	●				00-- 0000	00-- 0000	00-- 0000	uu-- uuuu
PAS1		●	●	●	---- 0000	---- 0000	---- 0000	---- uuuu
PBS0	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBS1	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCS0	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCS1		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDS0		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDS1		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PES0	●		●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PES0		●			0000 ----	0000 ----	0000 ----	uuuu ----
PES1	●				---- 0000	---- 0000	---- 0000	---- uuuu
PES1		●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PFS0			●	●	---- 0000	---- 0000	---- 0000	---- uuuu
PGS0				●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PGS1				●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PH				●	--11 1111	--11 1111	--11 1111	--uu uuuu
PHC				●	--11 1111	--11 1111	--11 1111	--uu uuuu
PHPU				●	--00 0000	--00 0000	--00 0000	--uu uuuu
TKM816DL				●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM816DH				●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM8ROL				●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TKM8ROH				●	---- --00	---- --00	---- --00	---- --uu
TKM8C0				●	--00 0000	--00 0000	--00 0000	--uu uuuu
TKM8C1				●	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
TKM8C2				●	1110 0100	1110 0100	1110 0100	uuuu uuuu

注：“u”表示不改变
“x”表示未知
“-”表示未定义

输入 / 输出端口

Holtek 单片机的输入 / 输出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~PH 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
PC	—	—	—	—	PC3	PC2	PC1	PC0
PCC	—	—	—	—	PCC3	PCC2	PCC1	PCC0
PCPU	—	—	—	—	PCPU3	PCPU2	PCPU1	PCPU0
PE	—	—	PE5	PE4	PE3	PE2	PE1	PE0
PEC	—	—	PEC5	PEC4	PEC3	PEC2	PEC1	PEC0
PEPU	—	—	PEPU5	PEPU4	PEPU3	PEPU2	PEPU1	PEPU0

I/O 口寄存器列表 – BS66F340

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
PC	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	PCC7	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PCPU	PCPU7	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PDC	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PDPU	PDPU7	PDPU6	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
PE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PEC	PEC7	PEC6	PEC5	PEC4	PEC3	PEC2	PEC1	PEC0
PEPU	PEPU7	PEPU6	PEPU5	PEPU4	PEPU3	PEPU2	PEPU1	PEPU0

I/O 口寄存器列表 – BS66F350

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
PC	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	PCC7	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PCPU	PCPU7	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PDC	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PDPU	PDPU7	PDPU6	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
PE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PEC	PEC7	PEC6	PEC5	PEC4	PEC3	PEC2	PEC1	PEC0
PEPU	PEPU7	PEPU6	PEPU5	PEPU4	PEPU3	PEPU2	PEPU1	PEPU0
PF	—	—	PF5	PF4	PF3	PF2	PF1	PF0
PFC	—	—	PFC5	PFC4	PFC3	PFC2	PFC1	PFC0
PFPU	—	—	PFPU5	PFPU4	PFPU3	PFPU2	PFPU1	PFPU0

I/O 口寄存器列表 – BS66F360

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
PC	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	PCC7	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PCPU	PCPU7	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PDC	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PDPU	PDPU7	PDPU6	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
PE	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
PEC	PEC7	PEC6	PEC5	PEC4	PEC3	PEC2	PEC1	PEC0
PEPU	PEPU7	PEPU6	PEPU5	PEPU4	PEPU3	PEPU2	PEPU1	PEPU0
PF	—	—	PF5	PF4	PF3	PF2	PF1	PF0
PFC	—	—	PFC5	PFC4	PFC3	PFC2	PFC1	PFC0

寄存器名称	位							
	7	6	5	4	3	2	1	0
PFPU	—	—	PFPU5	PFPU4	PFPU3	PFPU2	PFPU1	PFPU0
PG	PG7	PG6	PG5	PG4	PG3	PG2	PG1	PG0
PGC	PGC7	PGC6	PGC5	PGC4	PGC3	PGC2	PGC1	PGC0
PGPU	PGPU7	PGPU6	PGPU5	PGPU4	PGPU3	PGPU2	PGPU1	PGPU0
PH	—	—	PH5	PH4	PH3	PH2	PH1	PH0
PHC	—	—	PHC5	PHC4	PHC3	PHC2	PHC1	PHC0
PHPU	—	—	PHPU5	PHPU4	PHPU3	PHPU2	PHPU1	PHPU0

I/O 口寄存器列表 – BS66F370

“—”：未定义，读为“0”

PAWUn：PA 口唤醒功能控制位

0：除能

1：使能

PAn/PBn/PCn/PDn/PEn/PFn/PGn/PHn：I/O 口数据位

0：数据 0

1：数据 1

PACn/PBCn/PCCn/PDCn/PECn/PFCn/PGCn/PHCn：I/O 口类型选择位

0：输出

1：输入

PAPUn/PBPUn/PCPUn/PDPUn/PEPUn/PFPUn/PGPUn/PHPUn：I/O 口上拉功能控制位

0：除能

1：使能

上拉电阻

许多产品应用在端口处于输入状态时需要外加一个上拉电阻来实现上拉的功能。为了免去外部上拉电阻，当引脚规划为输入时，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 引脚设为输入或 NMOS 输出时，上拉功能才会受 PxPU 控制开启，其他状态均不可用。

PA 口唤醒

当使用暂停指令“HALT”迫使单片机进入休眠或空闲模式，单片机的系统时钟将会停止以降低功耗，此功能对于电池及低功耗应用很重要。唤醒单片机有很多种方法，其中之一就是使 PA 口的其中一个引脚从高电平转为低电平。这个功能特别适合于通过外部开关来唤醒的应用。PA 口的每个引脚可以通过设置 PAWU 寄存器来单独选择是否具有唤醒功能。需要注意的是只有当引脚功能为通用 I/O 功能，且 MCU 处于 Power down 模式时，唤醒功能才会受 PAWU 控制开启，其他状态均不可用。

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~PHC，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。

输入 / 输出端口源电流控制寄存器

该系列单片机的每个输入 / 输出端口都具有不同的源电流驱动能力。通过选择寄存器 SLEDC 或 SLEDC1，特定 I/O 口可支持四种级别的源电流驱动能力。用户可参考直流电气特性章节来根据不同的应用选择相应的源电流。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SLEDC (BS66F340)	—	—	PCPS1	PCPS0	PBPS1	PBPS0	PAPS1	PAPS0
SLEDC (BS66F350/ BS66F360/ BS66F370)	PCPS3	PCPS2	PCPS1	PCPS0	PBPS1	PBPS0	PAPS1	PAPS0
SLEDC1 (BS66F370)	—	—	—	—	PHPS1	PHPS0	PGPS1	PGPS0

I/O 口源电流控制寄存器列表

SLEDC 寄存器 – BS66F340

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCPS1	PCPS0	PBPS1	PBPS0	PAPS1	PAPS0
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　**PCPS1~PCPS0**：PC3~PC0 源电流选择位
00：level 0（最小值）
01：level 1
10：level 2
11：level 3（最大值）

Bit 3~2　**PBPS1~PBPS0**：PB7~PB4 源电流选择位
00：level 0（最小值）
01：level 1
10：level 2
11：level 3（最大值）

Bit 1~0　**PAPS1~PAPS0**：PA7~PA5 和 PA1 源电流选择位
00：level 0（最小值）
01：level 1
10：level 2
11：level 3（最大值）

SLEDC 寄存器 – BS66F350/BS66F360/BS66F370

Bit	7	6	5	4	3	2	1	0
Name	PCPS3	PCPS2	PCPS1	PCPS0	PBPS1	PBPS0	PAPS1	PAPS0
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 **PCPS3~PCPS2**：PC7~PC4 源电流选择位
00：level 0（最小值）
01：level 1
10：level 2
11：level 3（最大值）

Bit 5~4 **PCPS1~PCPS0**：PC3~PC0 源电流选择位
00：level 0（最小值）
01：level 1
10：level 2
11：level 3（最大值）

Bit 3~2 **PBPS1~PBPS0**：PB7~PB4 源电流选择位
00：level 0（最小值）
01：level 1
10：level 2
11：level 3（最大值）

Bit 1~0 **PAPS1~PAPS0**：PA7~PA5 和 PA1 源电流选择位
00：level 0（最小值）
01：level 1
10：level 2
11：level 3（最大值）

SLEDC1 寄存器 – BS66F370

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	PHPS1	PHPS0	PGPS1	PGPS0
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 **PHPS1~PHPS0**：PH3~PH0 源电流选择位
00：Level 0 (最小值)
01：Level 1
10：Level 2
11：Level 3 (最大值)

Bit 1~0 **PGPS1~PGPS0**：PG3~PG0 源电流选择位
00：Level 0 (最小值)
01：Level 1
10：Level 2
11：Level 3 (最大值)

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。

当选择引脚共用输入功能，对应的输入和输出功能要进行合理设定。例如，I^2C SDA 端口被使用，对应的输出引脚共用功能则要通过寄存器 PxSn 设置为 SDI/SDA 功能。但是，如果选择外部中断功能，相关的输出引脚共用功能应选择作为 INTn 引脚，且也要选择中断输入信号。

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。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。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	PAS17	PAS16	—	—	PAS13	PAS12	PAS11	PAS10
PBS0	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
PBS1	PBS17	PBS16	PBS15	PBS14	PBS13	PBS12	PBS11	PBS10
PCS0	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
PES0	PES07	PES06	PES05	PES04	PES03	PES02	PES01	PES00
PES1	—	—	—	—	PES13	PES12	PES11	PES10
IFS	—	—	IFS5	IFS4	IFS3	IFS2	IFS1	IFS0

引脚共用功能选择寄存器列表 – BS66F340

寄存器名称	位							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	—	—	—	—	PAS13	PAS12	PAS11	PAS10
PBS0	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
PBS1	PBS17	PBS16	PBS15	PBS14	PBS13	PBS12	PBS11	PBS10
PCS0	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
PCS1	PCS17	PCS16	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
PDS0	PDS07	PDS06	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
PDS1	PDS17	PDS16	PDS15	PDS14	PDS13	PDS12	PDS11	PDS10
PES0	PES07	PES06	PES05	PES04	—	—	—	—
PES1	PES17	PES16	PES15	PES14	PES13	PES12	PES11	PES10
IFS	—	—	IFS5	IFS4	IFS3	IFS2	IFS1	IFS0

引脚共用功能选择寄存器列表 – BS66F350

寄存器名称	位							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	—	—	—	—	PAS13	PAS12	PAS11	PAS10
PBS0	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
PBS1	PBS17	PBS16	PBS15	PBS14	PBS13	PBS12	PBS11	PBS10
PCS0	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
PCS1	PCS17	PCS16	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
PDS0	PDS07	PDS06	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
PDS1	PDS17	PDS16	PDS15	PDS14	PDS13	PDS12	PDS11	PDS10
PES0	PES07	PES06	PES05	PES04	PES03	PES02	PES01	PES00
PES1	PES17	PES16	PES15	PES14	PES13	PES12	PES11	PES10
PFS0	—	—	—	—	PFS03	PFS02	PFS01	PFS00
IFS	—	—	IFS5	IFS4	IFS3	IFS2	IFS1	IFS0

引脚共用功能选择寄存器列表 – **BS66F360**

寄存器名称	位							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	—	—	—	—	PAS13	PAS12	PAS11	PAS10
PBS0	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
PBS1	PBS17	PBS16	PBS15	PBS14	PBS13	PBS12	PBS11	PBS10
PCS0	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
PCS1	PCS17	PCS16	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
PDS0	PDS07	PDS06	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
PDS1	PDS17	PDS16	PDS15	PDS14	PDS13	PDS12	PDS11	PDS10
PES0	PES07	PES06	PES05	PES04	PES03	PES02	PES01	PES00
PES1	PES17	PES16	PES15	PES14	PES13	PES12	PES11	PES10
PFS0	—	—	—	—	PFS03	PFS02	PFS01	PFS00
PGS0	PGS07	PGS06	PGS05	PGS04	PGS03	PGS02	PGS01	PGS00
PGS1	PGS17	PGS16	PGS15	PGS14	PGS13	PGS12	PGS11	PGS10
IFS	—	—	IFS5	IFS4	IFS3	IFS2	IFS1	IFS0

引脚共用功能选择寄存器列表 – **BS66F370**

- **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 功能选择
00/11：PA3
01：$\overline{SCS}$
10：XT1

Bit 5~4 **PAS05~PAS04**：PA2 功能选择

PAS0[5:4]	BS66F340	BS66F350	BS66F360	BS66F370
00	PA2/CTCK1	PA2	PA2	PA2
01	$\overline{SCS}$	$\overline{SCS}$	$\overline{SCS}$	$\overline{SCS}$
10	PA2/CTCK1	PA2	PA2	PA2
11	PA2/CTCK1	PA2	PA2	PA2

Bit 3~2 **PAS03~PAS02**：PA1 功能选择
00/10/11：PA1
01：CTP0

Bit 1~0 **PAS01~PAS00**：PA0 功能选择
00/10/11：PA0
01：SDO

- **PAS1 寄存器 – BS66F340**

Bit	7	6	5	4	3	2	1	0
Name	PAS17	PAS16	—	—	PAS13	PAS12	PAS11	PAS10
R/W	R/W	R/W	—	—	R/W	R/W	R/W	R/W
POR	0	0	—	—	0	0	0	0

Bit 7~6 **PAS17~PAS16**：PA7 功能选择
00/10/11：PA7
01：CTP1

Bit 5~4 未定义，读为“0”

Bit 3~2 **PAS13~PAS12**：PA5 功能选择
00/10/11：PA5
01：CTP0B

Bit 1~0 **PAS11~PAS10**：PA4 功能选择
00/11：PA4
01：SDO
10：XT2

- **PAS1 寄存器 – BS66F350/BS66F360/BS66F370**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	PAS13	PAS12	PAS11	PAS10
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 **PAS13~PAS12**：PA5 功能选择
00/10/11：PA5
01：CTP0B

Bit 1~0 **PAS11~PAS10**：PA4 功能选择
00/11：PA4
01：SDO
10：XT2

- **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 功能选择
00/10：PB3
01：RX
11：AN3

Bit 5~4 **PBS05~PBS04**：PB2 功能选择
00：PB2/PTPI
01：TX
10：PTP
11：AN2

Bit 3~2 **PBS03~PBS02**：PB1 功能选择
00/10：PB1
01：SCK/SCL
11：AN1

Bit 1~0 **PBS01~PBS00**：PB0 功能选择
00：PB0
01：SDI/SDA
10：VREF
11：AN0

- **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 功能选择
00/01：PB7/INT1
10：KEY4
11：AN7

Bit 5~4 **PBS15~PBS14**：PB6 功能选择
00/01：PB6/PTCK
10：KEY3
11：AN6

Bit 3~2 **PBS13~PBS12**：PB5 功能选择
00/01：PB5/STCK
10：KEY2
11：AN5

Bit 1~0 **PBS11~PBS10**：PB4 功能选择
00：PB4/PTPI
01：PTPB
10：KEY1
11：AN4

- **PCS0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
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 **PCS07~PCS06**：PC3 功能选择
00：PC3
01：PC3
10：KEY8
11：PC3

Bit 5~4 **PCS05~PCS04**：PC2 功能选择
00：PC2
01：PC2
10：KEY7
11：PC2

Bit 3~2 **PCS03~PCS02**：PC1 功能选择
00：PC1
01：PC1
10：KEY6
11：PC1

Bit 1~0 **PCS01~PCS00**：PC0 功能选择
00：PC0
01：PC0
10：KEY5
11：PC0

- **PCS1 寄存器 – BS66F350/BS66F360/BS66F370**

Bit	7	6	5	4	3	2	1	0
Name	PCS17	PCS16	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
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 **PCS17~PCS16**：PC7 功能选择
00：PC7
01：PC7
10：KEY12
11：PC7

Bit 5~4 **PCS15~PCS14**：PC6 功能选择
00：PC6
01：PC6
10：KEY11
11：PC6

Bit 3~2 **PCS13~PCS12**：PC5 功能选择
00：PC5
01：PC5
10：KEY10
11：PC5

Bit 1~0 **PCS11~PCS10**：PC4 功能选择
00：PC4
01：PC4
10：KEY9
11：PC4

- **PDS0 寄存器 – BS66F350/BS66F360/BS66F370**

Bit	7	6	5	4	3	2	1	0
Name	PDS07	PDS06	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
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 **PDS07~PDS06**：PD3 功能选择
00：PD3
01：PD3
10：KEY16
11：PD3

Bit 5~4 **PDS05~PDS04**：PD2 功能选择
00：PD2
01：PD2
10：KEY15
11：PD2

Bit 3~2 **PDS03~PDS02**：PD1 功能选择
00：PD1
01：PD1
10：KEY14
11：PD1

Bit 1~0 **PDS01~PDS00**：PD0 功能选择
00：PD0
01：PD0
10：KEY13
11：PD0

- **PDS1 寄存器 – BS66F350/BS66F360/BS66F370**

Bit	7	6	5	4	3	2	1	0
Name	PDS17	PDS16	PDS15	PDS14	PDS13	PDS12	PDS11	PDS10
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 **PDS17~PDS16**：PD7 功能选择
00：PD7
01：PD7
10：KEY20
11：PD7

Bit 5~4 **PDS15~PDS14**：PD6 功能选择
00：PD6
01：PD6
10：KEY19
11：PD6

Bit 3~2 **PDS13~PDS12**：PD5 功能选择
00：PD5
01：PD5
10：KEY18
11：PD5

Bit 1~0 **PDS11~PDS10**：PD4 功能选择
00：PD4
01：PD4
10：KEY17
11：PD4

- **PES0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PES07	PES06	PES05	PES04	PES03	PES02	PES01	PES00
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 **PES07~PES06**：PE3 功能选择

PES0[7:6]	BS66F340	BS66F350	BS66F360	BS66F370
00	PE3/STPI	PE3/STPI	PE3/STPI	PE3/STPI
01	STPB	STPB	STPB	STPB
10	KEY12	PE3/STPI	KEY24	KEY24
11	PE3/STPI	PE3/STPI	PE3/STPI	PE3/STPI

Bit 5~4 **PES05~PES04**：PE2 功能选择

PES0[5:4]	BS66F340	BS66F350	BS66F360	BS66F370
00	PE2/STPI	PE2/STPI	PE2/STPI	PE2/STPI
01	STP	STP	STP	STP
10	KEY11	PE2/STPI	KEY23	KEY23
11	PE2/STPI	PE2/STPI	PE2/STPI	PE2/STPI

Bit 3~2 **PES03~PES02**：PE1 功能选择

PES0[3:2]	BS66F340	BS66F350	BS66F360	BS66F370
00	PE1	—	PE1	PE1
01	PE1	—	PE1	PE1
10	KEY10	—	KEY22	KEY22
11	PE1	—	PE1	PE1

未定义，读为“0”– BS66F350

Bit 1~0 **PES01~PES00**：PE0 功能选择

PES0[1:0]	BS66F340	BS66F350	BS66F360	BS66F370
00	PE0	—	PE0	PE0
01	PE0	—	PE0	PE0
10	KEY9	—	KEY21	KEY21
11	PE0	—	PE0	PE0

未定义，读为“0”– BS66F350

- **PES1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PES17	PES16	PES15	PES14	PES13	PES12	PES11	PES10
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 **PES17~PES16**：PE7 功能选择

PES1[7:6]	BS66F340	BS66F350	BS66F360	BS66F370
00	—	PE7	PE7	PE7
01	—	CTP1B	CTP1B	CTP1B
10	—	PE7	KEY26	KEY26
11	—	PE7	PE7	PE7

未定义，读为“0”– BS66F340

Bit 5~4 **PES15~PES14**：PE6 功能选择

PES1[5:4]	BS66F340	BS66F350	BS66F360	BS66F370
00	—	PE6	PE6	PE6
01	—	CTP1	CTP1	CTP1
10	—	PE6	KEY25	KEY25
11	—	PE6	PE6	PE6

未定义，读为“0”– BS66F340

Bit 3~2 **PES13~PES12**：PE5 功能选择

PES1[3:2]	BS66F340	BS66F350	BS66F360	BS66F370
00	PE5	PE5/CTCK1	PE5/CTCK1	PE5/CTCK1
01	CTP1B	PE5/CTCK1	PE5/CTCK1	PE5/CTCK1
10	OSC2	OSC2	OSC2	OSC2
11	PE5	PE5/CTCK1	PE5/CTCK1	PE5/CTCK1

Bit 1~0 **PES11~PES10**：PE4 功能选择
00：PE4
01：PE4
10：OSC1
11：PE4

- **PFS0 寄存器 – BS66F360/BS66F370**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	PFS03	PFS02	PFS01	PFS00
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 **PFS03~PFS02**：PF1 功能选择
00/01/11：PF1
10：KEY28
Bit 1~0 **PFS01~PFS00**：PF0 功能选择
00/01/11：PF0
10：KEY27

- **PGS0 寄存器 – BS66F370**

Bit	7	6	5	4	3	2	1	0
Name	PGS07	PGS06	PGS05	PGS04	PGS03	PGS02	PGS01	PGS00
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 **PGS07~PGS06**：PG3 引脚选择
00：PG3
01：PG3
10：KEY32
11：PG3
Bit 5~4 **PGS05~PGS04**：PG2 引脚选择
00：PG2
01：PG2
10：KEY31
11：PG2
Bit 3~2 **PGS03~PGS02**：PG1 引脚选择
00：PG1
01：PG1
10：KEY30
11：PG1

Bit 1~0 **PGS01~PGS00**：PG0 引脚选择
00：PG0
01：PG0
10：KEY29
11：PG0

- **PGS1 寄存器 – BS66F370**

Bit	7	6	5	4	3	2	1	0
Name	PGS17	PGS16	PGS15	PGS14	PGS13	PGS12	PGS11	PGS10
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 **PGS17~PGS16**：PG7 引脚选择
00：PG7
01：PG7
10：KEY36
11：PG7

Bit 5~4 **PGS15~PGS14**：PG6 引脚选择
00：PG6
01：PG6
10：KEY35
11：PG6

Bit 3~2 **PGS13~PGS12**：PG5 引脚选择
00：PG5
01：PG5
10：KEY34
11：PG5

Bit 1~0 **PGS11~PGS10**：PG4 引脚选择
00：PG4
01：PG4
10：KEY33
11：PG4

- **IFS 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	IFS5	IFS4	IFS3	IFS2	IFS1	IFS0
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”

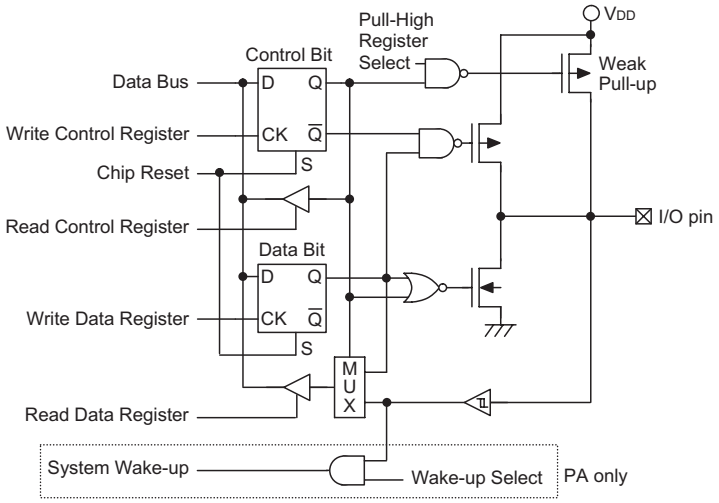
Bit 5~4 **IFS5~IFS4**：$\overline{\text{SCS}}$ 输入源引脚选择
00/10：PA2
01/11：PA3

Bit 3~2 **IFS3~IFS2**：PTPI 输入源引脚选择
00/10：PB2
01/11：PB4

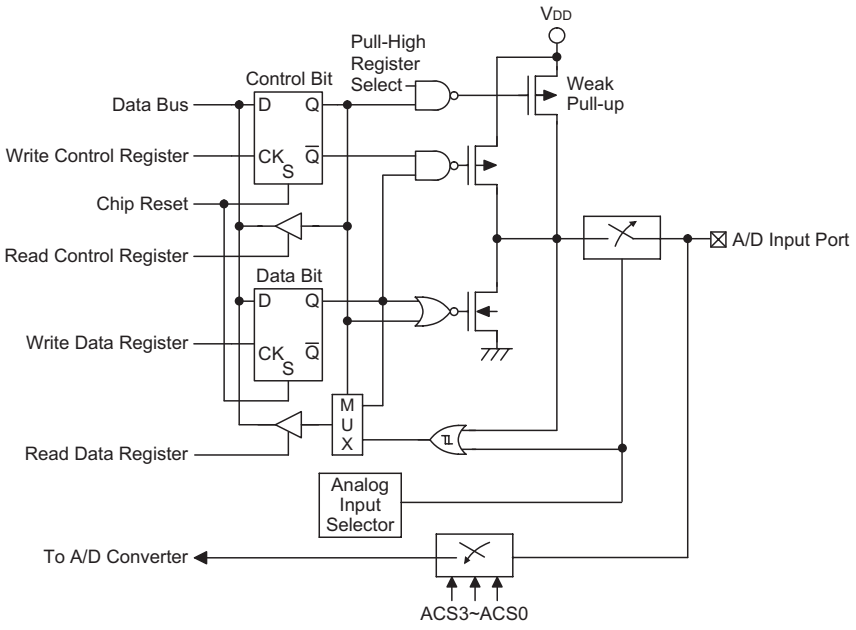
Bit 1~0 **IFS1~IFS0**：STPI 输入源引脚选择
00/01：PE2
01/11：PE3

输入 / 输出引脚结构

下图为输入 / 输出引脚的内部结构图。输入 / 输出引脚的准确逻辑结构图可能与此图不同，这里只是为了方便对 I/O 引脚功能的理解提供的一个参考。



通用输入 / 输出端口



A/D 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 PAC~PFC，某些引脚位被设定输出状态，这些输出引脚会有初始高电平输出，除非数据寄存器端口 PA~PF 在程序中被预先设定。设置哪些引脚是输入及哪些引脚是输出，可通过设置正确的值到适当的端口控制寄存器，或使用指令“SET [m].i”及“CLR [m].i”来设定端口控制寄存器中个别的位。注意，当使用这些位控制指令时，系统即将产生一个读 - 修改 - 写的操作。单片机需要先读入整个端口上的数据，修改个别的位，然后重新把这些数据写入到输出端口。

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

定时器模块 – TM

控制和测量时间在任何单片机中都是一个很重要的部分。该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，捕捉输入，比较匹配输出，单脉冲输出以及 PWM 输出等功能。每个定时器模块有两个独立中断。每个 TM 外加的输入输出引脚，扩大了定时器的灵活性，便于用户使用。

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

简介

该系列单片机包含 4 个 TM，每个 TM 可被划分为一个特定的类型，即简易型 TM、标准型 TM 或周期型 TM。虽然性质相似，但不同 TM 特性复杂度不同。本章介绍简易型、标准型和周期型 TM 的共性，更多详细资料分别见后面各章。三种类型 TM 的特性和区别见下表。

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

TM 功能概要

TM 操作

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

TM 时钟源

驱动 TM 计数器的时钟源很多。通过设置 xTMn 控制寄存器的 xTnCK2~xTnCK0 位，选择所需的时钟源，其中 x 代表 C、S 或 P，n 代表具体 TM 编号。由于该系列单片机分别只有一个 STM 和 PTM，所以 STM 和 PTM 的相关引脚或控制位无相关编号“n”。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、标准型和周期型 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

无论哪种类型的 TM，都有一个或两个 TM 输入引脚 xTCKn 和 xTPnI。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 输入脚可选择上升沿有效或下降沿有效。STCK 和 PTCK 引脚还可分别用作 STM 和 PTM 单脉冲模式的外部触发引脚。

另一种 xTM 输入引脚 STPI 或 PTPI 作为捕捉输入脚，其有效边沿有上升沿、下降沿和双沿，通过设置 STMC1 寄存器中的 STIO1~STIO0 位或 PTMC1 寄存器中的 PTIO1~PTIO0 位来选择有效边沿类型。除了 PTPI 引脚外，PTCK 引脚也可用作 PTM 捕捉输入模式的外部触发引脚。

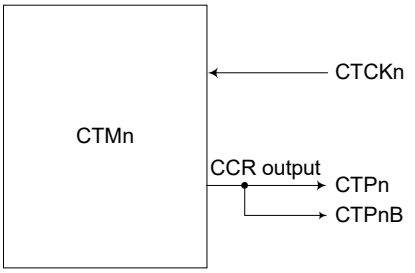
每个 TM 都有两个输出引脚 xTPn 和 xTPnB。xTPnB 信号为 xTPn 输出的反向信号。当 TM 工作在比较匹配输出模式且比较匹配发生时，这些引脚会由 TM 控制切换到高电平或低电平或翻转。外部 xTPn 和 xTPnB 输出引脚也被 TM 用来产生 PWM 输出波形。当 TM 输出引脚与其它功能共用时，TM 输出功能需要通过相关引脚共用功能选择寄存器先被设置。

单片机型号	CTM		STM		PTM	
	输入引脚	输出引脚	输入引脚	输出引脚	输入引脚	输出引脚
BS66F340 BS66F350 BS66F360 BS66F370	CTCK0 CTCK1	CTP0,CTP0B CTP1,CTP1B	STCK, STPI	STP, STPB	PTCK, PTPI	PTP, PTPB

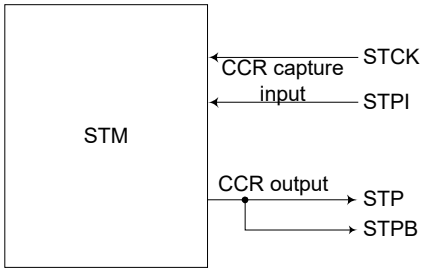
TM 外部引脚

TM 输入 / 输出引脚选择

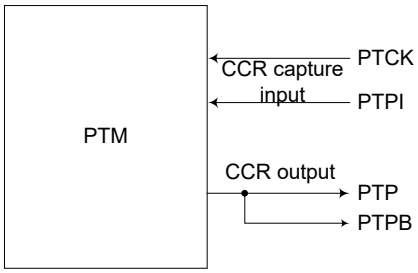
通过设置与 TM 输入 / 输出引脚相关的寄存器的位，选择作为 TM 输入 / 输出功能或其它共用功能。正确设置选择位将相应的引脚用作 TM 输入 / 输出。更多引脚共用功能选择详见引脚共用功能章节。



CTM 功能引脚控制框图 –n=0 或 1



STM 功能引脚控制框图

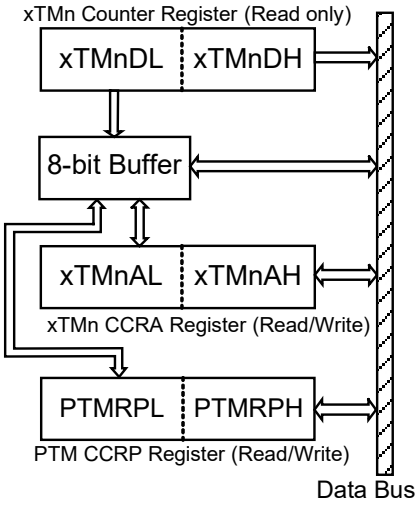


PTM 功能引脚控制框图

编程注意事项

TM 计数寄存器和捕捉 / 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 PTMRPL，否则可能导致无法预期的结果。



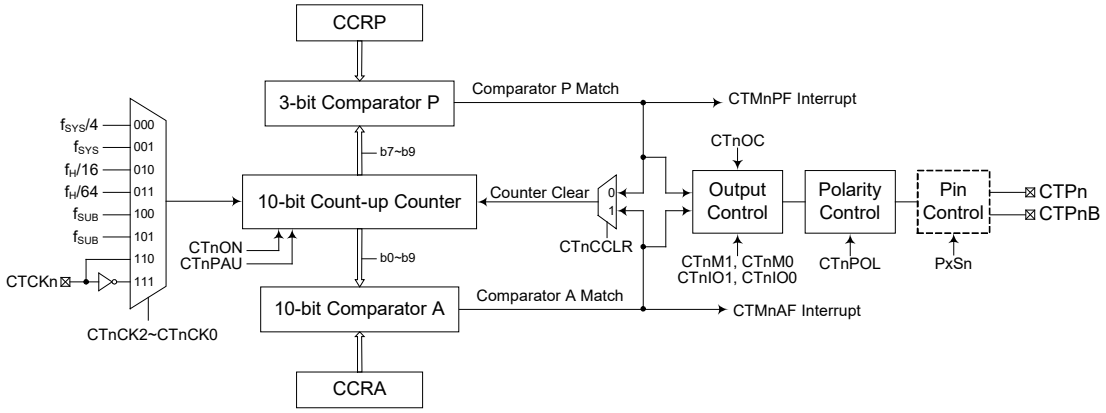
读写流程如下步骤所示：

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

简易型 TM – CTM

虽然简易型 TM 是这几种 TM 类型中最简单的形式，但仍然包括三种工作模式，即比较匹配输出、定时 / 事件计数器和 PWM 输出模式。简易型 TM 由一个外部输入脚控制并驱动两个外部输出脚。

单片机型号	CTM 核心	CTM 输入脚	CTM 输出脚	注释
BS66F340 BS66F350 BS66F360 BS66F370	10-bit CTM (CTM0， CTM1)	CTCK0 CTCK1	CTP0，CTP0B CTP1, CTP1B	n=0~1



简易型 TM 方框图 –n=0 或 1

简易型 TM 操作

简易型 TM 核心是一个由用户选择的内部或外部时钟源驱动的 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 或 1

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

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，计数器将保留其剩余值，直到此位再次改变为低电平，从此值开始继续计数。

Bit 6~4　**CTnCK2~CTnCK0**：选择 CTMn 计数时钟位
000：$f_{SYS}/4$
001：f_{SYS}
010：$f_H/16$
011：$f_H/64$
100：f_{SUB}
101：f_{SUB}
110：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，细节方面请参考振荡器章节。

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剩余值，直到此位再次改变为高电平。
若 CTMn 处于比较匹配输出模式时，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，对应于 CTMn 计数器 bit 9 ~ bit 7 比较器 P 匹配周期

000：1024 个 CTMn 时钟周期
001：128 个 CTMn 时钟周期
010：256 个 CTMn 时钟周期
011：384 个 CTMn 时钟周期
100：512 个 CTMn 时钟周期
101：640 个 CTMn 时钟周期
110：768 个 CTMn 时钟周期
111：896 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，比较结果为 0 并清除内部计数器。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**：选择 CTMn 工作模式位

00：比较匹配输出模式
01：未定义模式
10：PWM 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 输出脚控制必须除能。

Bit 5~4 **CTnIO1~CTnIO0**：选择 CTMn 外部引脚 CTPn 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 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 CTPn 输出脚如何改变状态。这两位值的选择决定 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 CTPn 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 CTPn 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 CTPn 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 CTPn 将不会发生变化。在 CTMn 输出脚 CTPn 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 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 CTPn 的状态。PWM 输出功能通过这两位的变化进行更新。仅在 CTMn 关闭时改变 CTnIO1 和 CTnIO0 位的值是很有必要的。若在 CTMn 运行时改变 CTnIO1 和 CTnIO0 的值，PWM 输出的值是无法预料的。
另一输出脚 CTPnB 信号与 CTPn 输出反向。

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 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 模式。若 CTMn 处于定时 / 计数器模式，则其不受影响。在比较匹配输出模式时，比较匹配发生前其决定 CTMn 输出脚的逻辑电平值。在 PWM 模式时，其决定 PWM 信号是高有效还是低有效。另一输出脚 CTPnB 信号与 CTPn 输出反向。

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 CTPn 输出脚反相，为低时 CTPn 输出脚同相。另一输出脚 CTPnB 信号与 CTPn 输出反向。若 CTMn 处于定时 / 计数器模式时其不受影响。

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 模式时未使用。

简易型 TM 工作模式

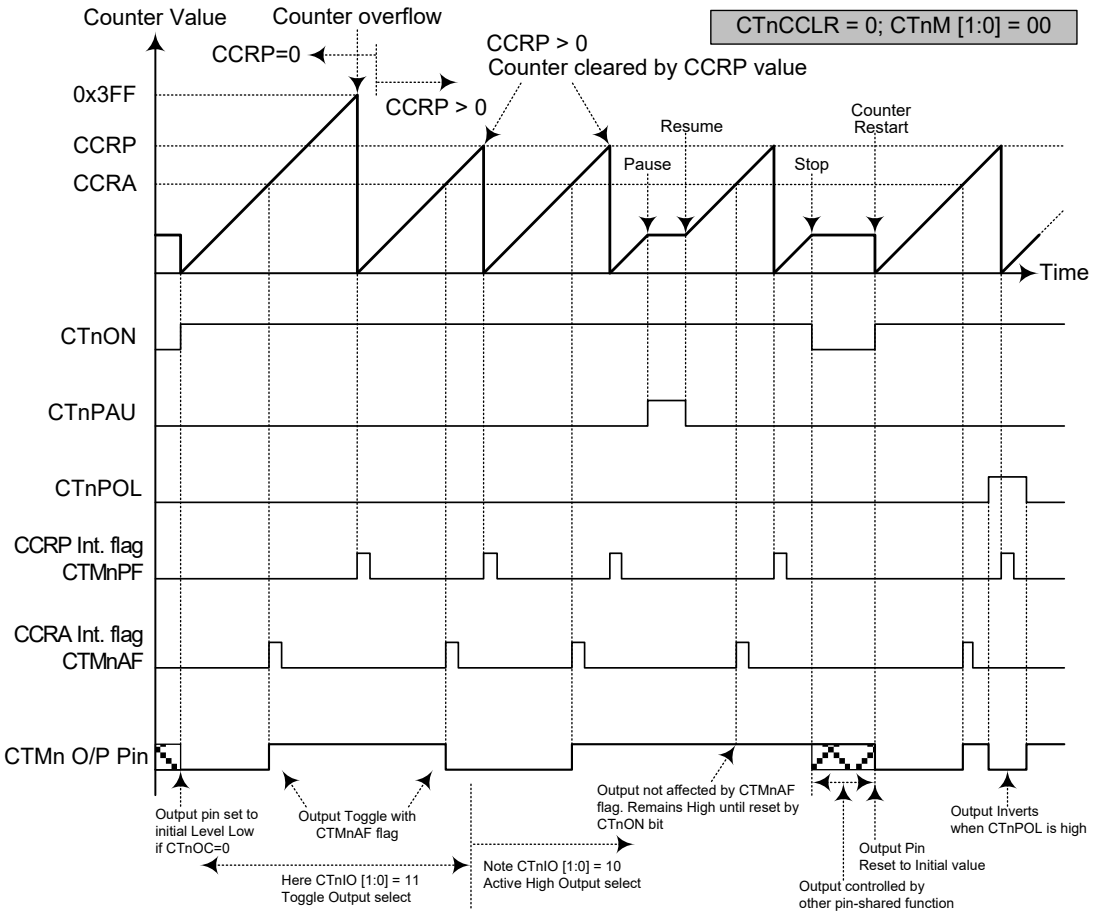
简易型 TM 有三种工作模式，即比较匹配输出模式，PWM 模式或定时 / 计数器模式。通过设置 CTMnC1 寄存器的 CTnM1 和 CTnM0 位选择任意工作模式。

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。

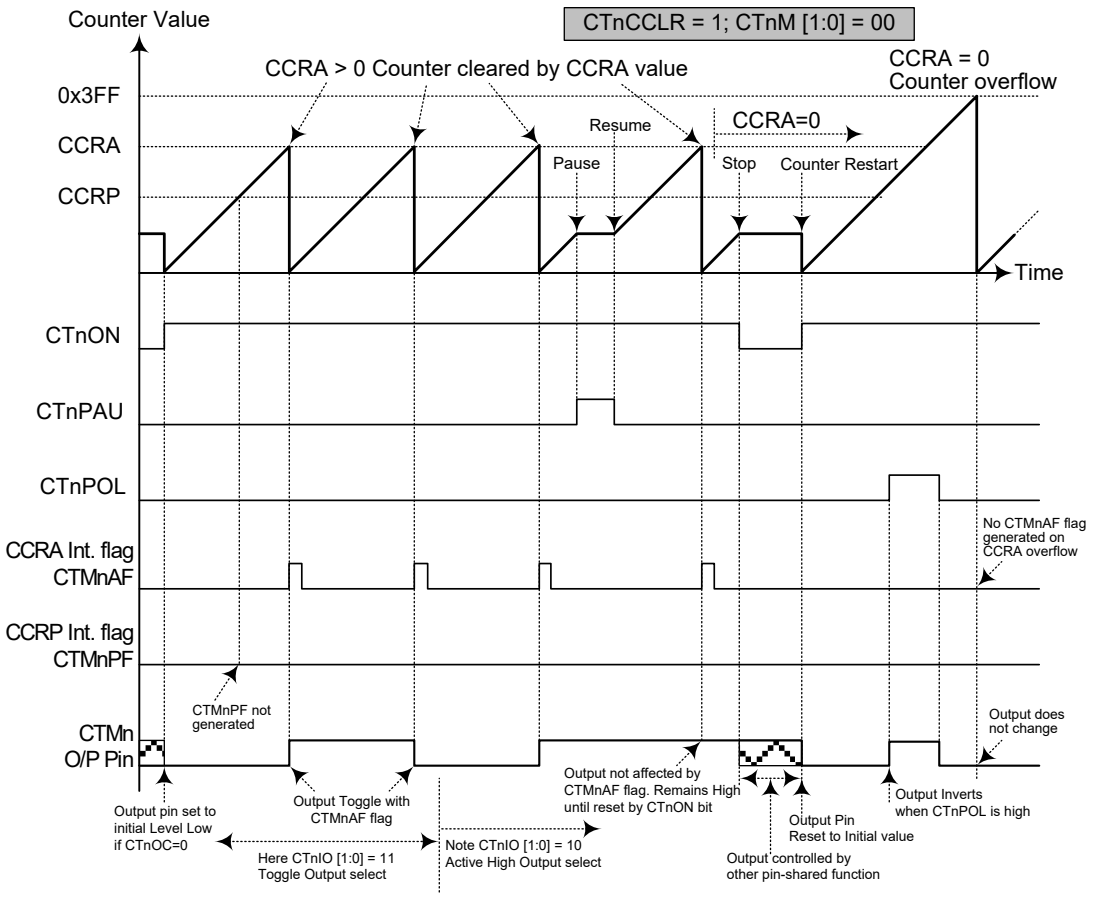
如果 CTMnC1 寄存器的 CTnCCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅 CTMnAF 中断请求标志产生。所以当 CTnCCLR 为高时，不产生 CTMnPF 中断请求标志。如果 CCRA 被清零，当计数达到最大值 3FFH 时，计数器溢出，而此时不产生 CTMnAF 请求标志。

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。CTMn 输出脚初始值，在 CTnON 位由低到高电平的变化后通过 CTnOC 位设置。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

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
4. n=0 或 1



比较匹配输出模式 – CTnCCLR=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
5. n=0 或 1

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 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 模式，边沿对齐模式，CTnDPX=0**

CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	128	256	384	512	640	768	896	1024
Duty	CCRA							

若 f_{SYS}=16MHz，CTMn 时钟源选择 $f_{SYS}/4$，CCRP=2，CCRA=128，

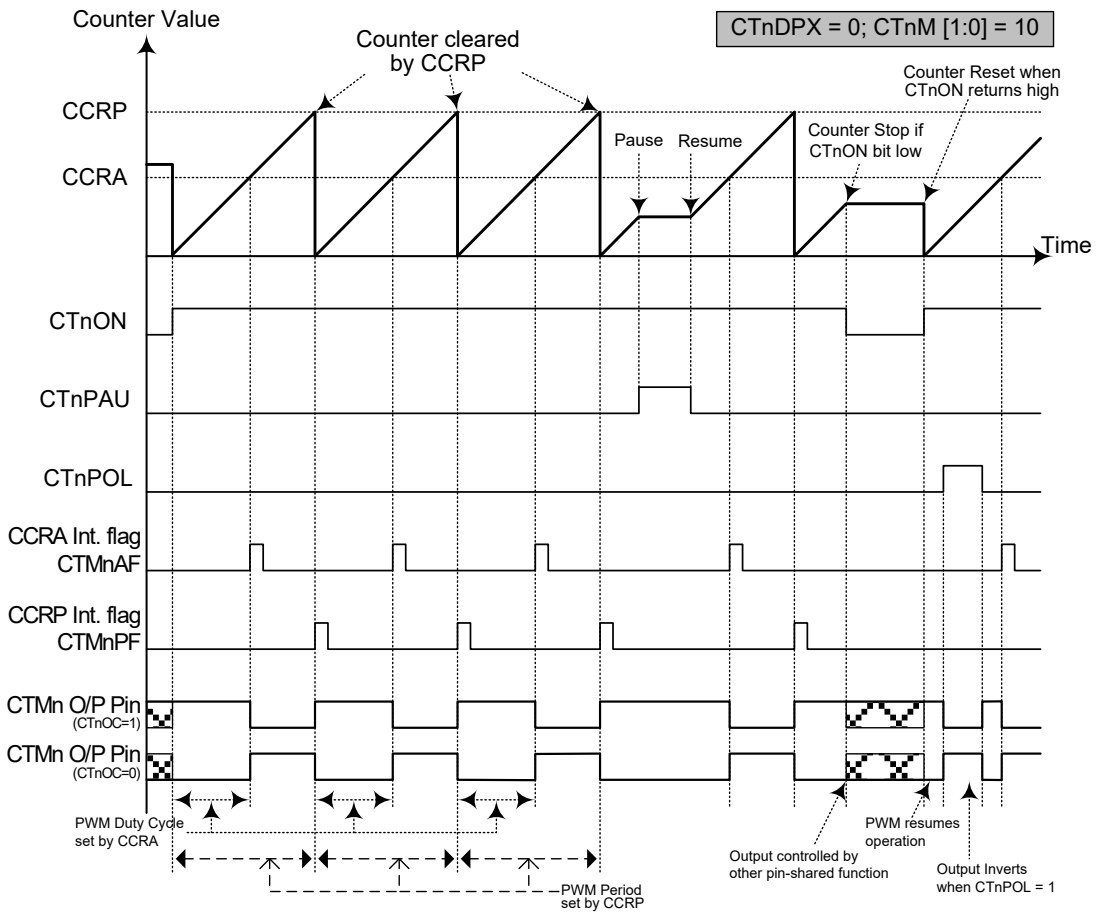
CTMn PWM 输出频率 =($f_{SYS}/4$)/256=$f_{SYS}/1024$=15.625kHz，duty=128/256=50%。

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 模式，边沿对齐模式，CTnDPX=1**

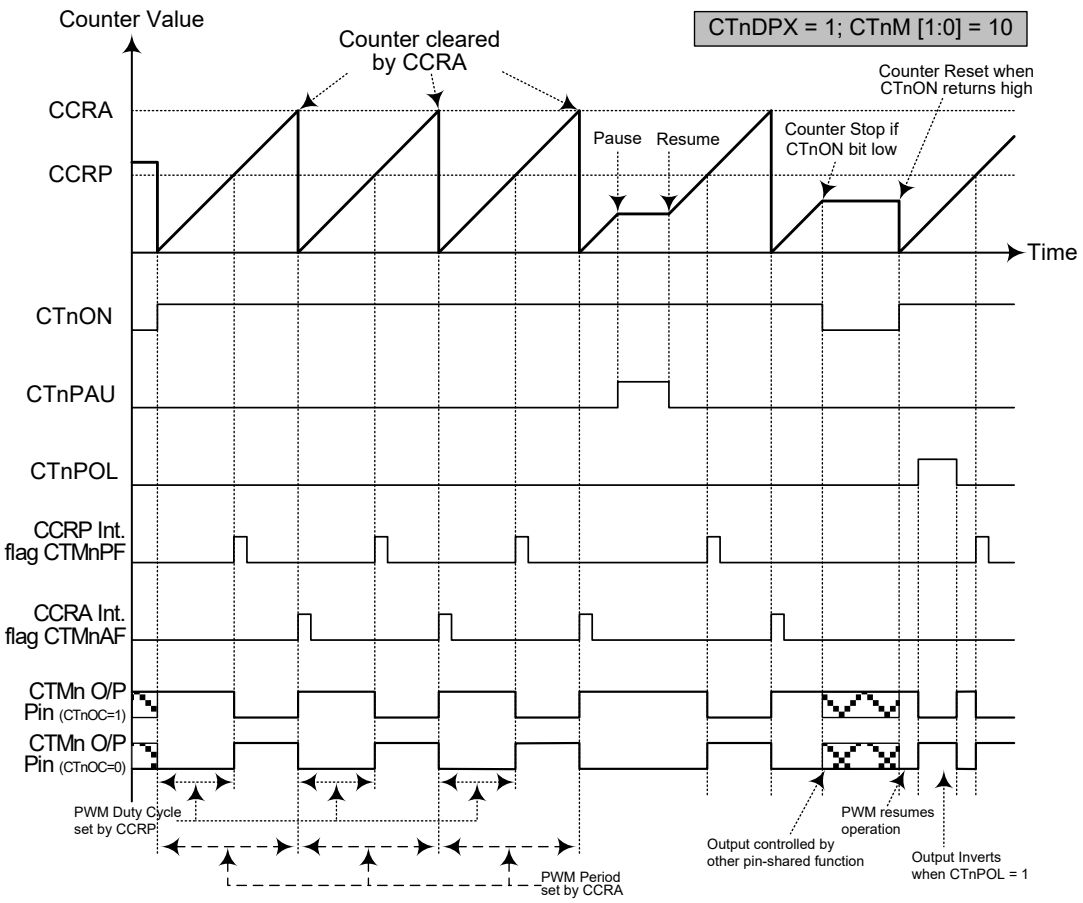
CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	CCRA							
Duty	128	256	384	512	640	768	896	1024

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 寄存器的值决定。



PWM 模式 – CTnDPX=0

注： 1. CTnDPX=0，CCRP 清除计数器
2. 计数器清零并设置 PWM 周期
3. 当 CTnIO1，CTnIO0=00 或 01，PWM 功能不变
4. CTnCCLR 位不影响 PWM 操作
5. n=0 或 1



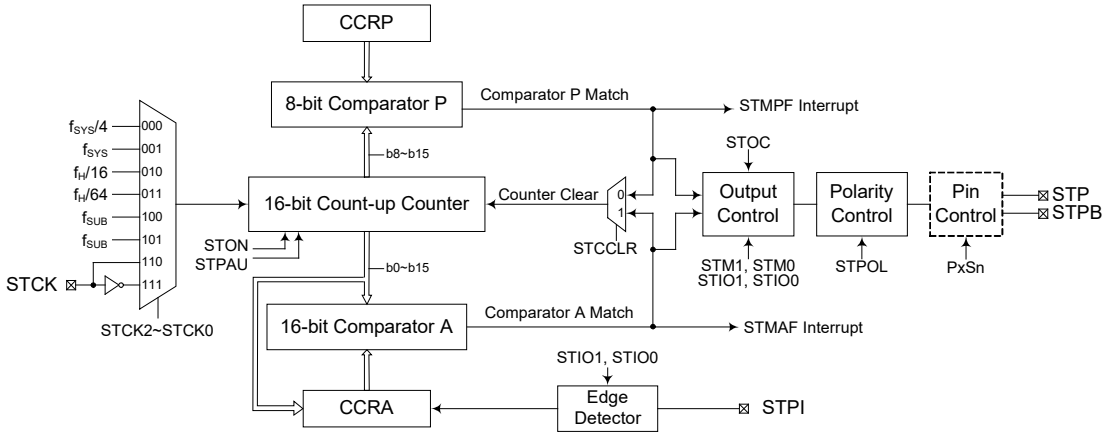
PWM 模式 – CTnDPX=1

注：1. CTnDPX=1，CCRA 清除计数器
2. 计数器清零并设置 PWM 周期
3. 当 CTnIO1，CTnIO0=00 或 01，PWM 功能不变
4. CTnCCLR 位不影响 PWM 操作
5. n=0 或 1

标准型 TM – STM

标准型 TM 包括 5 种工作模式，即比较匹配输出，定时 / 事件计数器，捕捉输入，单脉冲输出和 PWM 输出模式。标准型 TM 由两个外部输入脚控制并驱动两个外部输出脚。

单片机型号	STM 核心	STM 输入脚	STM 输出脚
BS66F340 BS66F350 BS66F360 BS66F370	16-bit STM	STCK, STPI	STP, STPB



标准型 TM 框图

标准型 TM 操作

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

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

标准型 TM 寄存器介绍

标准型 TM 的所有工作模式由一系列寄存器控制。一对只读寄存器用来存放 16 位计数器的值，一对读 / 写寄存器存放 16 位 CCRA 的值，一个读 / 写寄存器存放 8 位 CCRP 的值。剩下两个控制寄存器设置不同的操作和控制模式。

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

16-bit 标准型 TM 寄存器列表

STMDL 寄存器

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

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

STMDH 寄存器

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

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

STMAL 寄存器

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

Bit 7~0　STM CCRA 低字节寄存器 bit 7~bit 0
STM 16-bit CCRA bit 7~bit 0

STMAH 寄存器

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

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

STMC0 寄存器

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

Bit 7 **STPAU**：STM 计数器暂停控制位
0：运行
1：暂停
通过设置此位为高可使计数器暂停，清零此位恢复正常计数器操作。当处于暂停条件时，STM 保持上电状态并继续耗电。当此位由低到高转换时，计数器将保留其剩余值，直到此位再次改变为低电平，并从此值开始继续计数。

Bit 6~4 **STCK2~STCK0**：选择 STM 计数时钟位
000：$f_{SYS}/4$
001：f_{SYS}
010：$f_H/16$
011：$f_H/64$
100：f_{SUB}
101：f_{SUB}
110：STCK 上升沿时钟
111：STCK 下降沿时钟
此三位用于选择 TM 的时钟源。外部引脚时钟源能被选择在上升沿或下降沿有效。f_{SYS} 是系统时钟，f_H 和 f_{SUB} 是其它的内部时钟源，细节方面请参考振荡器章节。

Bit 3 **STON**：STM 计数器 On/Off 控制位
0：Off
1：On
此位控制 TM 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 TM。清零此位将停止计数器并关闭 TM 减少耗电。当此位经由高到低转换时，内部计数器将保持其剩余值，直到此位再次改变为高电平。
若 STM 处于比较匹配输出模式时，当 STON 位经由低到高的转换时，STM 输出脚将复位至 STOC 位指定的初始值。

Bit 2~0 未定义，读为“0”

STMC1 寄存器

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

Bit 7~6 **STM1~STM0**：选择 STM 工作模式位
00：比较匹配输出模式
01：捕捉输入模式
10：PWM 模式或单脉冲输出模式
11：定时 / 计数器模式
这两位设置 STM 需要的工作模式。为了确保操作可靠，STM 应在 STM1 和 STM0 位有任何改变前先关掉。在定时 / 计数器模式，TM 输出脚控制必须除能。

Bit 5~4 **STIO1~STIO0**：选择 STM 输出引脚（STP）或 STM 输入引脚（STPI）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 模式 / 单脉冲输出模式
00：强制无效状态
01：强制有效状态
10：PWM 输出
11：单脉冲输出
捕捉输入模式
00：在 STPI 上升沿输入捕捉
01：在 STPI 下降沿输入捕捉
10：在 STPI 双沿输入捕捉
11：输入捕捉除能
定时 / 计数器模式
未使用
此两位用于决定在一定条件达到时 STM 外部引脚 STP 或 STPI 如何改变状态。这两位值的选择取决于 STM 运行在哪种模式下。
在比较匹配输出模式下，STIO1 和 STIO0 位决定当从比较器 A 比较匹配输出发生时 STM 输出脚 STP 如何改变状态。当从比较器 A 比较匹配输出发生时 STP 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。STP 输出脚的初始值通过 STMC1 寄存器的 STOC 位设置取得。注意，由 STIO1 和 STIO0 位得到的输出电平必须与通过 STOC 位设置的初始值不同，否则当比较匹配发生时，STP 输出脚将不会发生变化。在 STP 输出脚改变状态后，通过 STON 位由低到高电平的转换复位至初始值。
在 PWM 模式，STIO1 和 STIO0 决定比较匹配条件发生时怎样改变 STP 输出脚的状态。PWM 输出功能通过这两位的变化进行更新。仅在 STM 关闭时改变 STIO1 和 STIO0 位的值是很有必要的。若在 STM 运行时改变 STIO1 和 STIO0 的值，PWM 输出的值将无法预料。
另一输出脚 STPB 信号与 STP 输出反向。

Bit 3 **STOC**：STM 输出脚 STP 输出控制位

比较匹配输出模式

0：初始低

1：初始高

PWM 模式 / 单脉冲输出模式

0：低有效

1：高有效

这是 STM 输出脚输出控制位。它取决于 STM 此时正运行于比较匹配输出模式还是 PWM 模式 / 单脉冲输出模式。若 STM 处于定时 / 计数器模式，则其不受影响。在比较匹配输出模式时，比较匹配发生前其决定 STM 输出脚 STP 的逻辑电平值。在 PWM 模式时，其决定 PWM 信号是高有效还是低有效。另一输出脚 STPB 信号与 STP 输出反向。

Bit 2 **STPOL**：STP 输出脚输出极性控制位

0：同相

1：反相

此位控制 STP/STPB 输出脚的极性。此位为高时 STP 输出脚反相，为低时 STP 输出脚同相。另一输出脚 STPB 信号与 STP 输出反向。若 STM 处于定时 / 计数器模式时其不受影响。

Bit 1 **STDPX**：STM PWM 周期 / 占空比控制位

0：CCRP - 周期；CCRA - 占空比

1：CCRP - 占空比；CCRA - 周期

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

Bit 0 **STCCLR**：选择 STM 计数器清零条件位

0：STM 比较器 P 匹配

1：STM 比较器 A 匹配

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

STMRP 寄存器

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

Bit 7~0 **STRP7~STRP0**：STM CCRP 8-bit 寄存器

STM CCRP 8 位寄存器，与 STM 计数器 bit 15~bit 8 比较。比较器 P 匹配周期

0：65536 个 STM 时钟周期

1~255：256×(1~255) 个 STM 时钟周期

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

标准型 TM 工作模式

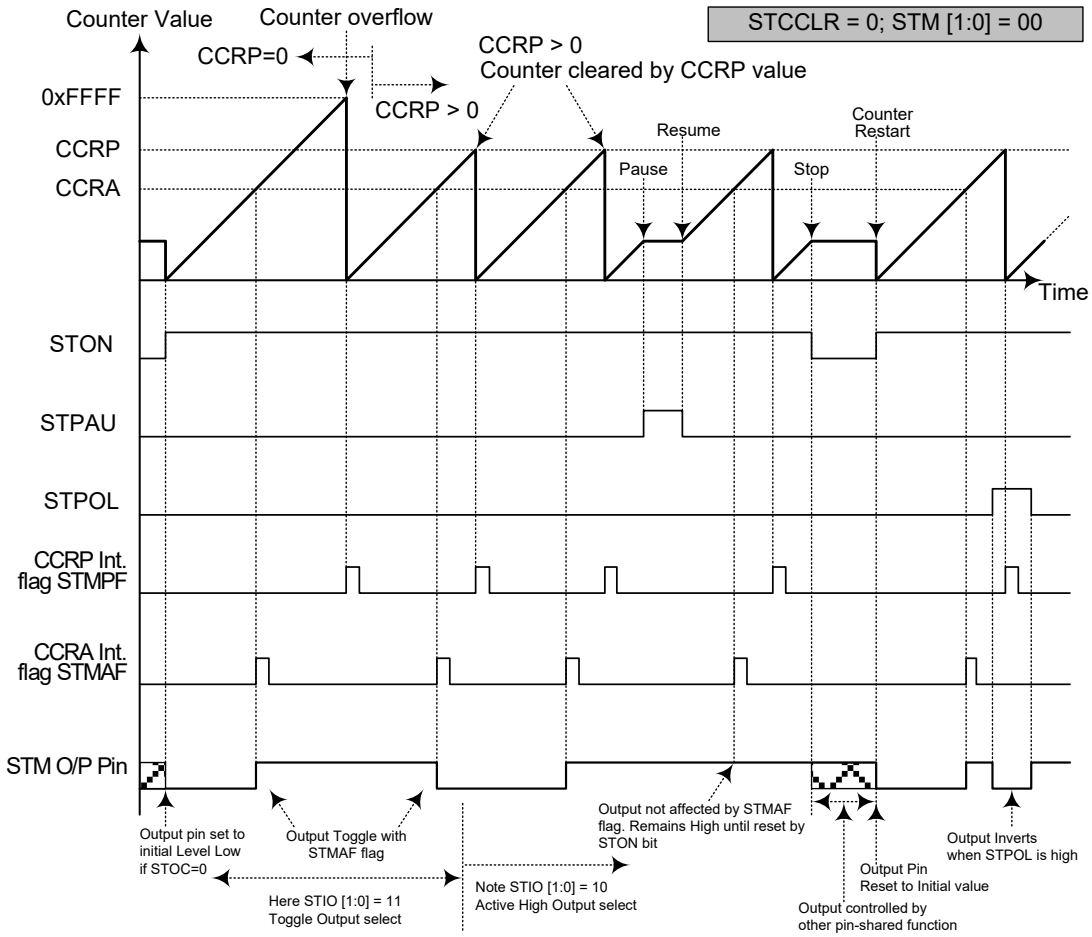
标准型 TM 有五种工作模式，即比较匹配输出模式，PWM 输出模式，单脉冲输出模式，捕捉输入模式或定时 / 计数器模式。通过设置 STMC1 寄存器的 STM1 和 STM0 位选择任意模式。

比较匹配输出模式

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

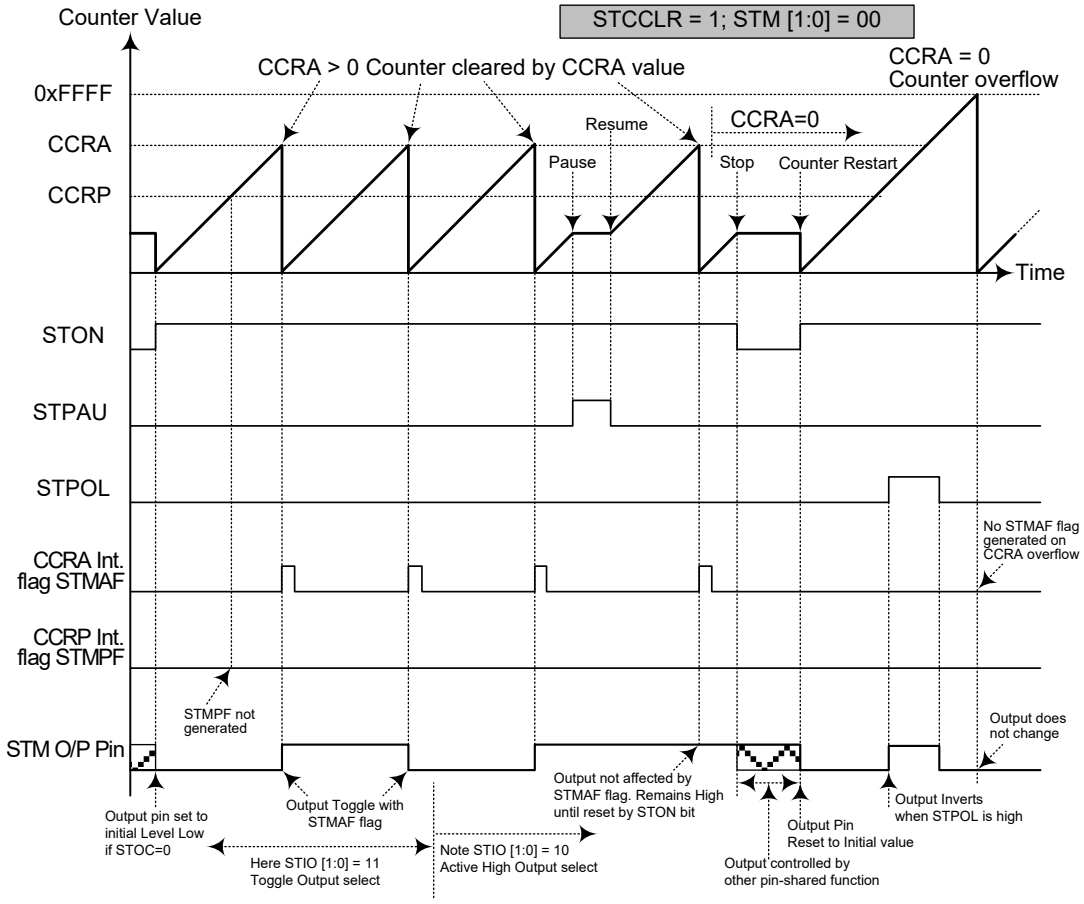
如果 STMC1 寄存器的 STCCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅产生 STMAF 中断请求标志。所以当 STCCLR 为高时，不会产生 STMPF 中断请求标志。在比较匹配输出模式下，CCRA 不能设为“0”。

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



比较匹配输出模式 – STCCLR=0

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



比较匹配输出模式 – STCCLR=1

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

定时 / 计数器模式

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

PWM 输出模式

为使 STM 工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“10”，且 STIO1 和 STIO0 位也需要设置为“10”。STM 的 PWM 功能在马达控制、加热控制、照明控制等方面十分有用。给 TM 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 模式中，STCCLR 位不影响 PWM 周期。CCRA 和 CCRP 寄存器决定 PWM 波形，一个用来清除内部计数器并控制 PWM 波形的频率，另一个用来控制占空比。哪个寄存器控制频率或占空比取决于 STMC1 寄存器的 STDPX 位。所以 PWM 波形由 CCRA 和 CCRP 寄存器共同决定。

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

- **16-bit STM，PWM 模式，边沿对齐模式，STDPX=0**

CCRP	1~255	0
Period	CCRP×256	65536
Duty	CCRA	

若 f_{SYS}=16MHz，STM 时钟源选择 f_{SYS}/4，CCRP=2，CCRA=128，

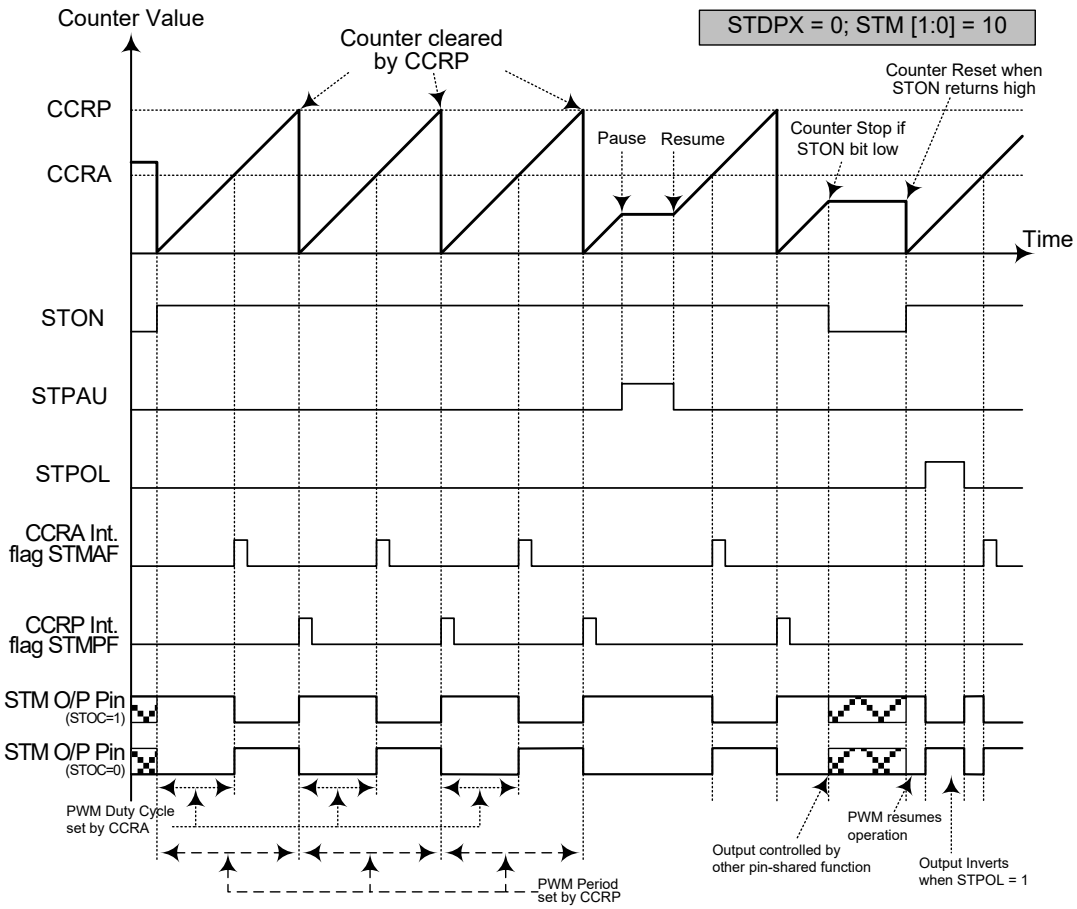
STM PWM 输出频率 =(f_{SYS}/4)/(2×256)= f_{SYS}/2048=7.8125kHz，duty=128/(2×256)=25%

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%

- **16-bit STM，PWM 模式，边沿对齐模式，STDPX=1**

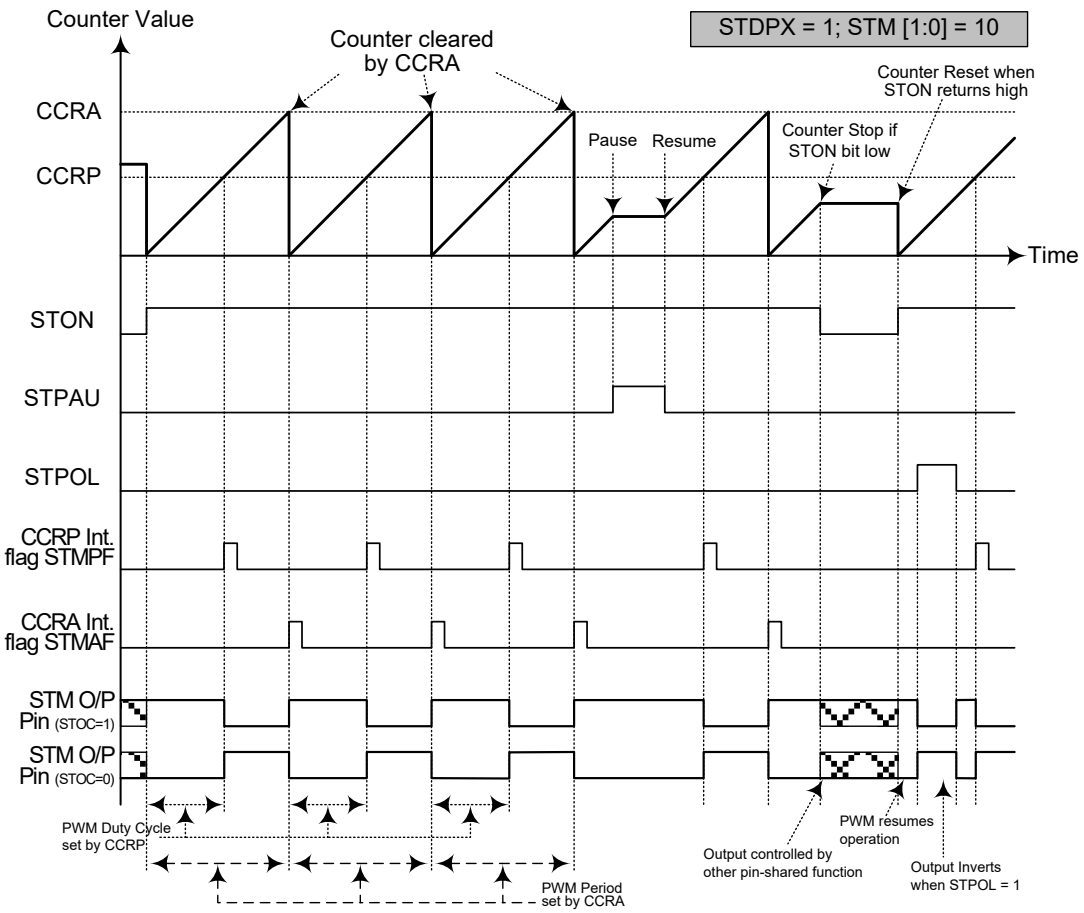
CCRP	1~255	0
Period	CCRA	
Duty	CCRP×256	65536

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



PWM 模式 – STDPX=0

注： 1. STDPX=0，CCRP 清除计数器
2. 计数器清零并设置 PWM 周期
3. 当 STIO1，STIO0=00 或 01，PWM 功能不变
4. STCCLR 位不影响 PWM 操作



PWM 模式 – STDPX=1

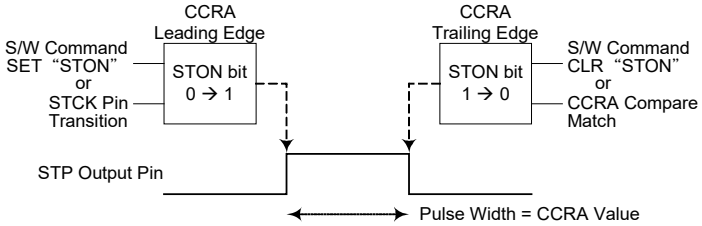
注： 1. STDPX=1，CCRA 清除计数器
2. 计数器清零并设置 PWM 周期
3. 当 STIO1，STIO0=00 或 01，PWM 功能不变
4. STCCLR 位不影响 PWM 操作

单脉冲模式

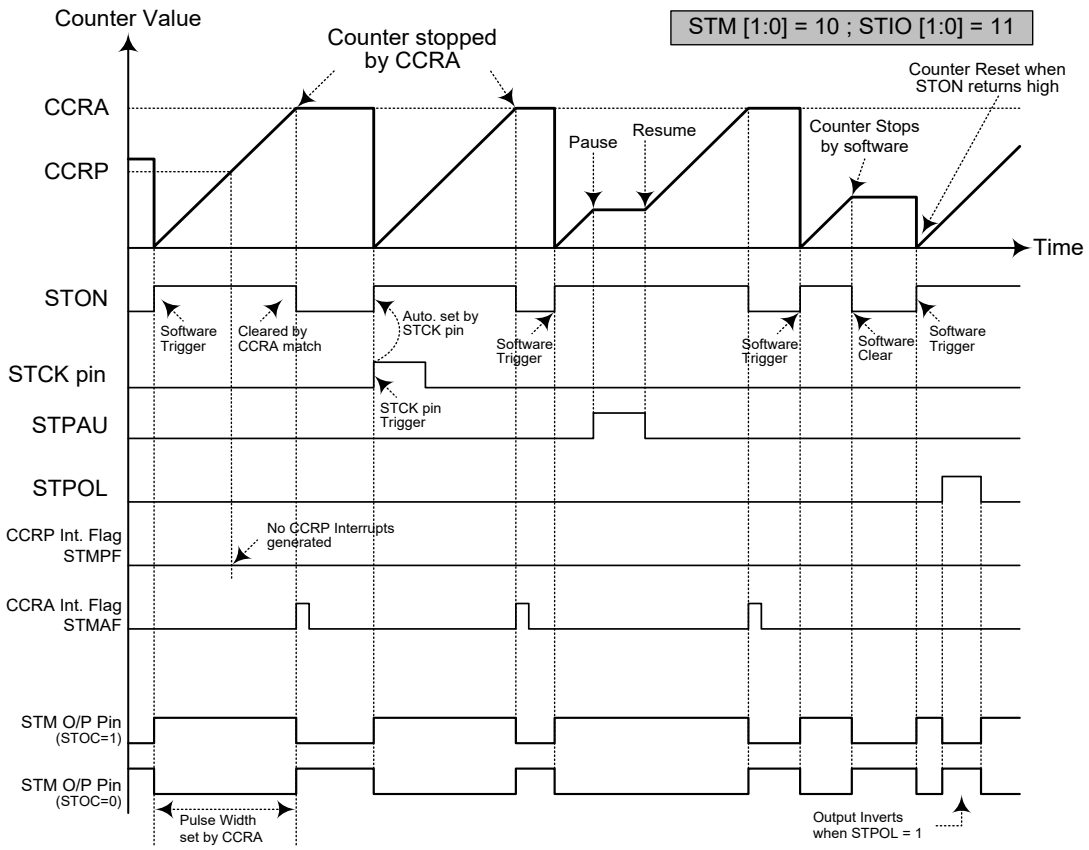
为使 STM 工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“10”，同时 STIO1 和 STIO0 位需要设置为“11”。正如模式名所言，单脉冲输出模式，在 TM 输出脚将产生一个脉冲输出。

脉冲输出可以通过应用程序控制 STON 位由低到高的转变来触发。而处于单脉冲模式时，STON 位可由 STCK 脚自动由低转变为高，进而初始化单脉冲输出状态。当 STON 位转变为高电平时，计数器将开始运行，并产生脉冲前沿。当脉冲有效时 STON 位保持高电平。通过应用程序使 STON 位清零或比较器 A 比较匹配发生时，产生脉冲下降沿。

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



单脉冲产生示意图



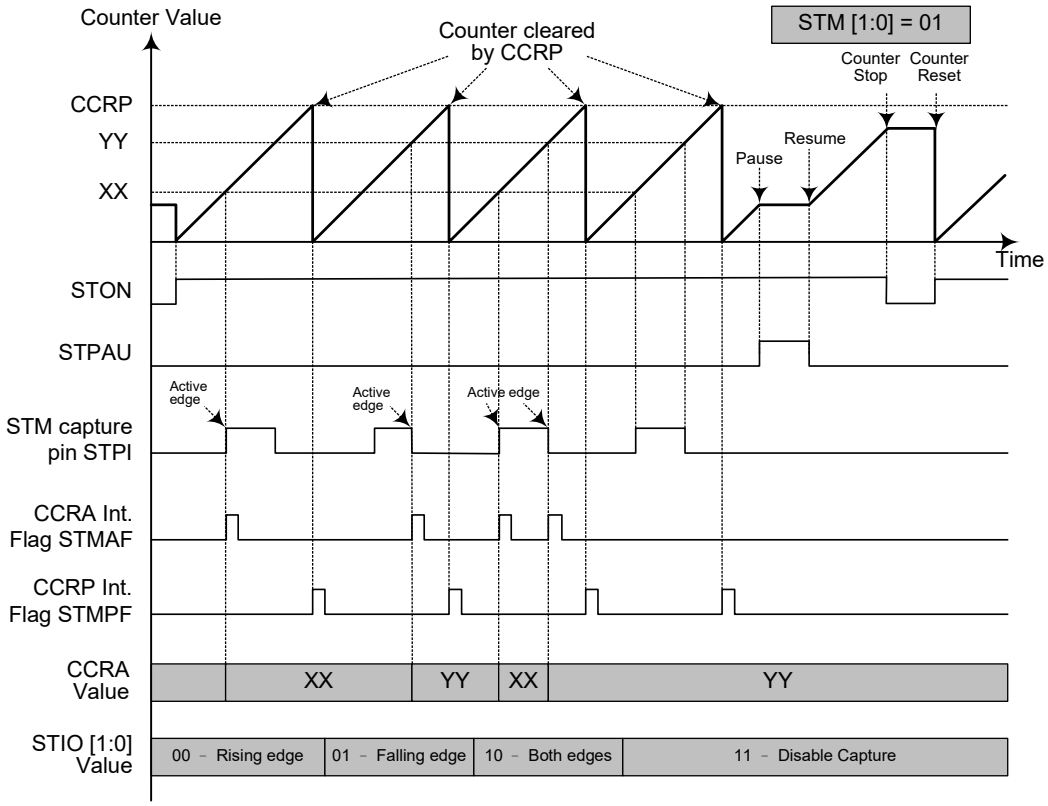
单脉冲模式

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

捕捉输入模式

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

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



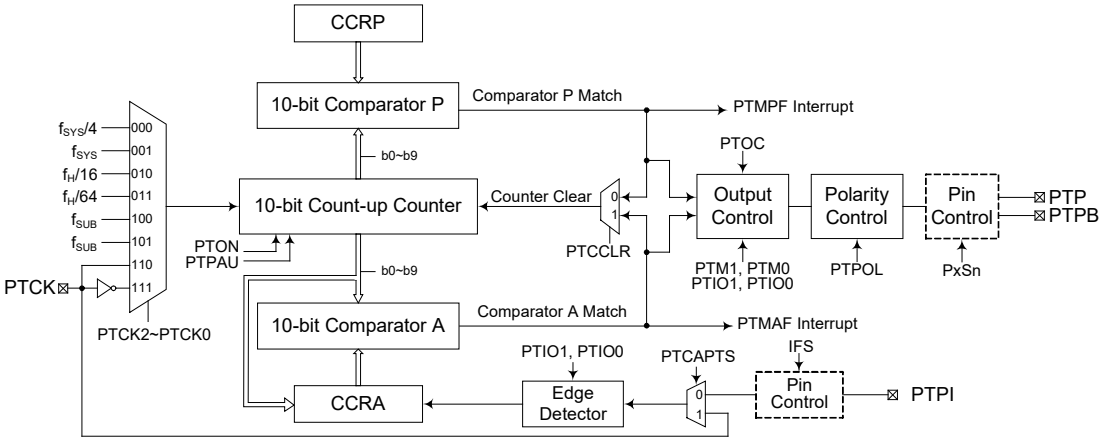
捕捉输入模式

注： 1. STM1，STM0=01 并通过 STIO1 和 STIO0 位设置有效边沿
2. STM 捕捉输入脚的有效边沿将计数器的值转移到 CCRA 中
3. STCCLR 位未使用
4. 无输出功能 – STOC 和 STPOL 位未使用
5. 计数器值由 CCRP 决定，在 CCRP 为“0”时，计数器计数值可达最大

周期型 TM – PTM

周期型 TM 包括 5 种工作模式，即比较匹配输出、定时 / 事件计数器、捕捉输入、单脉冲输出和 PWM 输出模式。周期型 TM 由两个外部输入脚控制并驱动两个外部输出脚。

单片机型号	PTM 核心	PTM 输入脚	PTM 输出脚
BS66F340 BS66F350 BS66F360 BS66F370	10-bit PTM	PTCK，PTPI	PTP, PTPB



周期型 TM 方框图

周期型 TM 操作

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

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

周期型 TM 寄存器介绍

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

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

10-bit 周期型 TM 寄存器列表

PTMDL 寄存器

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

Bit 7~0　PTM 计数器低字节寄存器 bit 7 ~ bit 0
PTM 10-bit 计数器 bit 7 ~ bit 0

PTMDH 寄存器

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

Bit 7~2　未定义，读为“0”

Bit 1~0　PTM 计数器高字节寄存器 bit 1 ~ bit 0
PTM 10-bit 计数器 bit 9 ~ bit 8

PTMAL 寄存器

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

Bit 7~0　PTM CCRA 低字节寄存器 bit 7 ~ bit 0
PTM 10-bit CCRA bit 7 ~ bit 0

PTMAH 寄存器

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

Bit 7~2 未定义，读为“0”
Bit 1~0 PTM CCRA 高字节寄存器 bit 1 ~ bit 0
PTM 10-bit CCRA bit 9 ~ bit 8

PTMRPL 寄存器

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

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

PTMRPH 寄存器

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

Bit 7~2 未定义，读为“0”
Bit 1~0 PTRP9~PTRP8：PTM CCRP 高字节寄存器 bit 1 ~ bit 0
PTM 10-bit CCRP bit 9 ~ bit 8

PTMC0 寄存器

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

Bit 7 **PTPAU**：PTM 计数器暂停控制位
0：运行
1：暂停
通过设置此位为高可使计数器暂停，清零此位恢复正常计数器操作。当处于暂停条件时，PTM 保持上电状态并继续耗电。当此位由低到高转变时，计数器将保留其剩余值，直到此位再次改变为低电平，并从此值开始继续计数。

Bit 6~4 **PTCK2~PTCK0**：选择 PTM 计数时钟位
000：$f_{SYS}/4$
001：f_{SYS}
010：$f_H/16$
011：$f_H/64$
100：f_{SUB}
101：f_{SUB}
110：PTCK 上升沿
111：PTCK 下降沿
此三位用于选择PTM的时钟源。外部引脚时钟源能被选择在上升沿或下降沿有效。f_{SYS} 是系统时钟，f_H 和 f_{SUB} 是其它的内部时钟源，细节方面请参考振荡器章节

Bit 3 **PTON**：PTM 计数器 On/Off 控制位
0：Off
1：On
此位控制 PTM 的总开关功能。设置此位为高则使能计数器使其运行，清零此位则除能 PTM。清零此位将停止计数器并关闭 PTM 减少耗电。当此位经由低到高转变时，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剩余值，直到此位再次改变为高电平。
若 PTM 处于比较匹配输出模式时，当 PTON 位经由低到高转换时，PTM 输出脚将复位至 PTOC 位指定的初始值。

Bit 2~0 未定义，读为“0”

PTMC1 寄存器

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

Bit 7~6 **PTM1~PTM0**：选择 PTM 工作模式位
00：比较匹配输出模式
01：捕捉输入模式
10：PWM 模式或单脉冲输出模式
11：定时 / 计数器模式
这两位设置 PTM 需要的工作模式。为了确保操作可靠，PTM 应在 PTM1 和 PTM0 位有任何改变前先关掉。在定时 / 计数器模式，PTM 输出脚控制必须除能。

Bit 5~4 **PTIO1~PTIO0**：选择 PTM 输出引脚（PTP）或 PTM 输入引脚（PTCK/PTPI）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 模式 / 单脉冲输出模式
00：强制无效状态
01：强制有效状态
10：PWM 输出
11：单脉冲输出
捕捉输入模式
00：在 PTPI 或 PTCK 上升沿输入捕捉
01：在 PTPI 或 PTCK 下降沿输入捕捉
10：在 PTPI 或 PTCK 双沿输入捕捉
11：输入捕捉除能
定时 / 计数器模式
未使用
此两位用于决定在一定条件达到时 PTM 输出脚如何改变状态。这两位值的选择取决于 PTM 运行在哪种模式下。
在比较匹配输出模式下，PTIO1 和 PTIO0 位决定当从比较器 A 比较匹配输出发生时 PTM 输出脚如何改变状态。当从比较器 A 比较匹配输出发生时 PTM 输出脚能设为切换高、切换低或翻转当前状态。若此两位同时为 0 时，这个输出将不会改变。PTM 输出脚的初始值通过 PTMC1 寄存器的 PTOC 位设置取得。注意，由 PTIO1 和 PTIO0 位得到的输出电平必须与通过 PTOC 位设置的初始值不同，否则当比较匹配发生时，PTM 输出脚将不会发生变化。在 PTM 输出脚改变状态后，可通过 PTON 位由低到高电平的转换将其复位至 PTOC 指定的初始值。

在 PWM 模式，PTIO1 和 PTIO0 用于决定比较匹配条件发生时怎样改变 PTM 输出脚的状态。PWM 输出功能通过这两位的变化进行更新。仅在 PTM 关闭时改变 PTIO1 和 PTIO0 位的值是很有必要的。若在 PTM 运行时改变 PTIO1 和 PTIO0 的值，PWM 输出的值是无法预料的。另一输出脚 PTPB 信号与 PTP 输出反向。

Bit 3 **PTOC**：PTM PTP 输出控制位

比较匹配输出模式

0：初始低

1：初始高

PWM 模式 / 单脉冲输出模式

0：低有效

1：高有效

这是 PTM 输出脚输出控制位。它取决于 PTM 此时正运行于比较匹配输出模式还是 PWM 模式 / 单脉冲输出模式。若 PTM 处于定时 / 计数器模式，则其不受影响。在比较匹配输出模式时，比较匹配发生前其决定 PTM 输出脚的逻辑电平值。在 PWM 模式时，其决定 PWM 信号是高有效还是低有效。另一输出脚 PTPB 信号与 PTP 输出反向。

Bit 2 **PTPOL**：PTM PTP 输出极性控制位

0：同相

1：反相

此位控制 PTP 输出脚的极性。此位为高时 PTM 输出脚反相，为低时 PTM 输出脚同相。另一输出脚 PTPB 信号与 PTP 输出反向。若 PTM 处于定时 / 计数器模式时其不受影响。

Bit 1 **PTCAPTS**：选择 PTM 捕捉触发源

0：来自 PTPI 引脚

1：来自 PTCK 引脚

Bit 0 **PTCCLR**：选择 PTM 计数器清零条件位

0：PTM 比较器 P 匹配

1：PTM 比较器 A 匹配

此位用于选择清除计数器的方法。周期型 PTM 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。PTCCLR 位设为高，计数器在比较器 A 比较匹配发生时被清除；此位设为低，计数器在比较器 P 比较匹配发生或计数器溢出时被清除。计数器溢出清除的方法仅在 CCRP 被清除为 0 时才能生效。PTCCLR 位在 PWM 模式、单脉冲或输入捕捉模式时未使用。

周期型 TM 工作模式

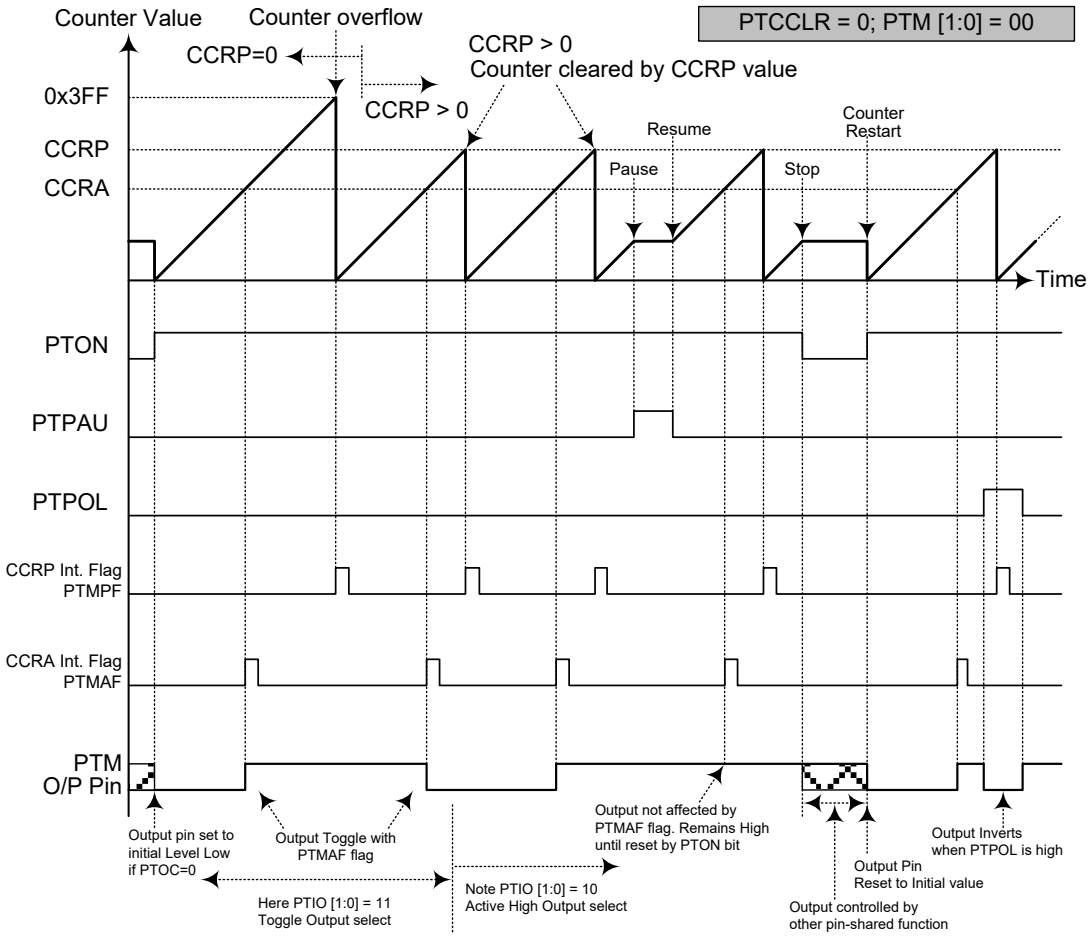
周期型 TM 有五种工作模式，即比较匹配输出模式、PWM 输出模式、单脉冲输出模式、捕捉输入模式或定时 / 计数器模式。通过设置 PTMC1 寄存器的 PTM1 和 PTM0 位选择任意模式。

比较匹配输出模式

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

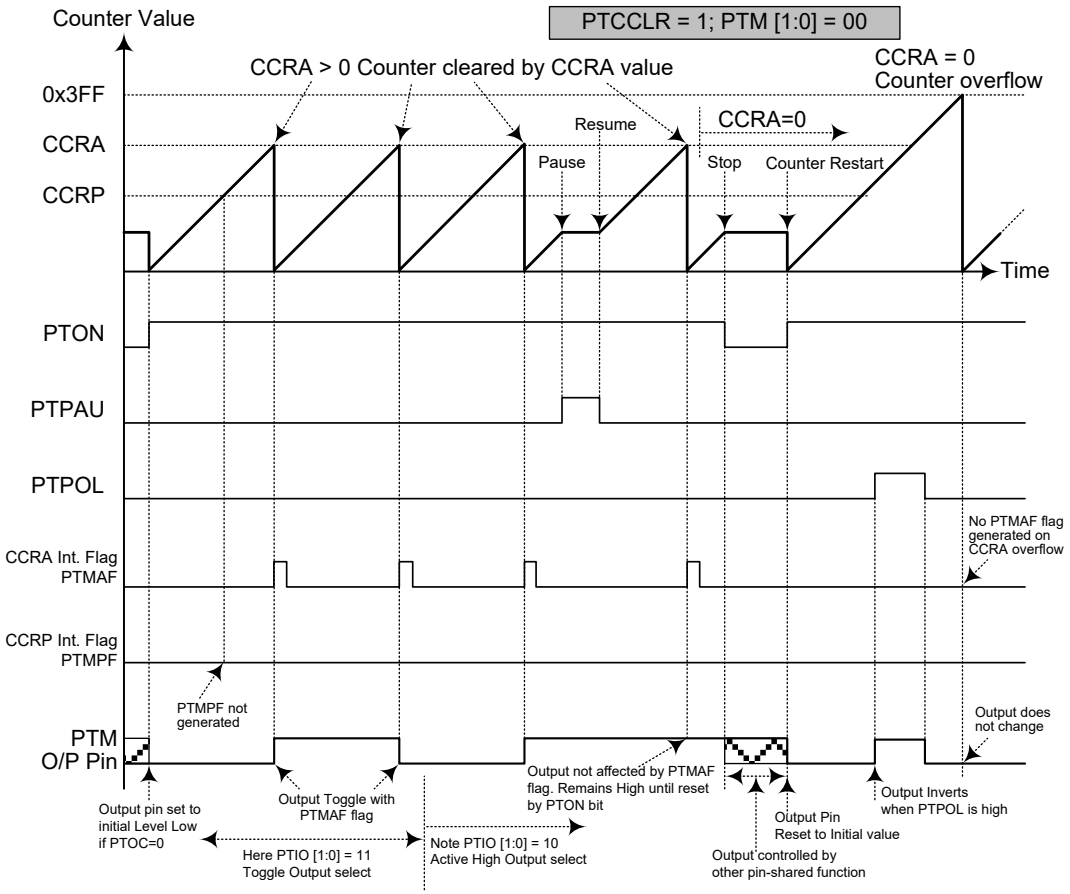
如果 PTMC1 寄存器的 PTCCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅 PTMAF 中断请求标志产生。所以当 PTCCLR 为高时，不会产生 PTMPF 中断请求标志。在比较匹配输出模式中，CCRA 寄存器值不能设为“0”。

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



比较器匹配输出模式 – PTCCLR=0

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



比较器匹配输出模式 – PTCCLR=1

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

定时 / 计数器模式

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

PWM 输出模式

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

由于 PWM 波形的周期和占空比可调，其波形的选择就较为灵活。在 PWM 模式中，PTCCLR 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。PTMC1 寄存器的 PTOC 位选择 PWM 波形的极性，PTIO1 和 PTIO0 位使能 PWM 输出或强制 PTM 输出脚为高电平或低电平。PTPOL 位用于 PWM 输出波形的极性反相控制。

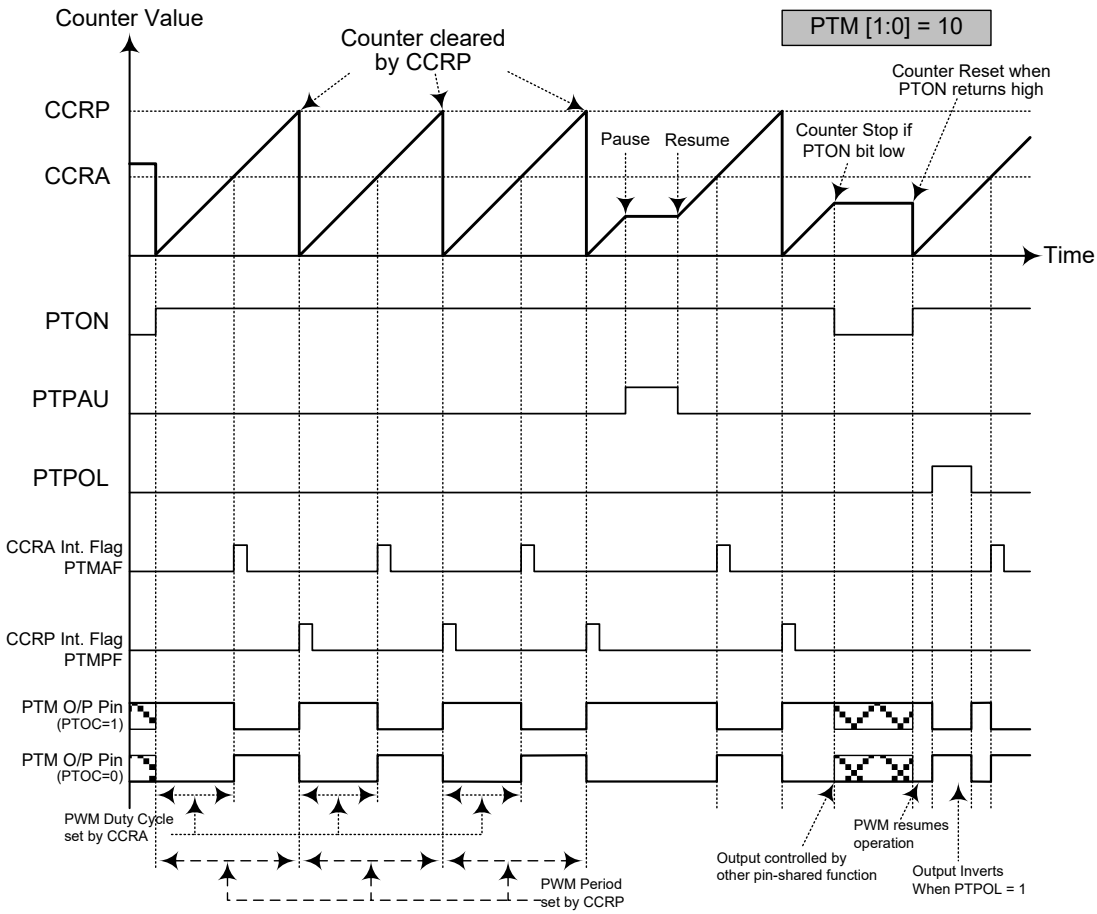
- **10-bit PTM，PWM 模式**

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

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

PTM PWM 输出频率 =(f_{SYS}/4) / 512=f_{SYS}/2048=7.8125kHz，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 模式

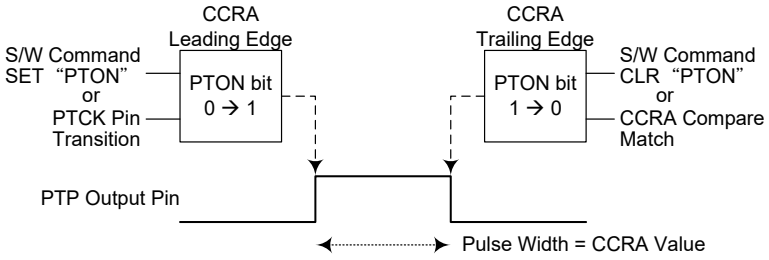
注：1. CCRP 清除计数器
2. 计数器清除并决定 PWM 周期
3. 当 PTIO[1:0]=00 或 01，PWM 功能不变
4. PTCCLR 位对 PWM 功能无影响

单脉冲输出模式

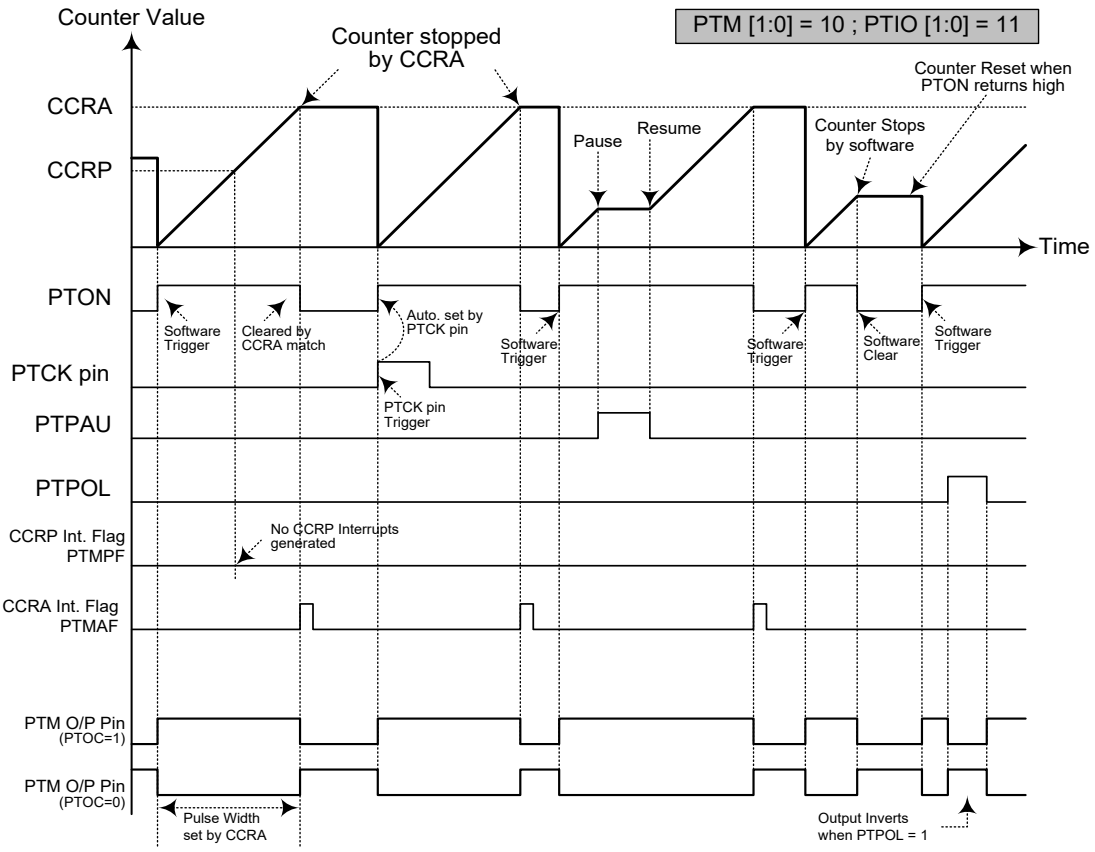
为使 PTM 工作在此模式，PTMC1 寄存器中的 PTM1 和 PTM0 位需要设置为“10”，并且相应的 PTIO1 和 PTIO0 需要设置为“11”。正如模式名所言，单脉冲输出模式，在 PTM 输出脚将产生一个脉冲输出。

通过应用程序控制 PTON 位由低到高的转变来触发脉冲前沿输出。而处于单脉冲模式时，PTON 位可由 PTCK 脚自动由低转变为高，进而依次初始化单脉冲输出。当 PTON 位转变为高电平时，计数器将开始运行，并产生脉冲前沿。通过应用程序使 PTON 位清零或比较器 A 比较匹配发生时，产生脉冲下降沿。

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



单脉冲产生示意图



单脉冲模式

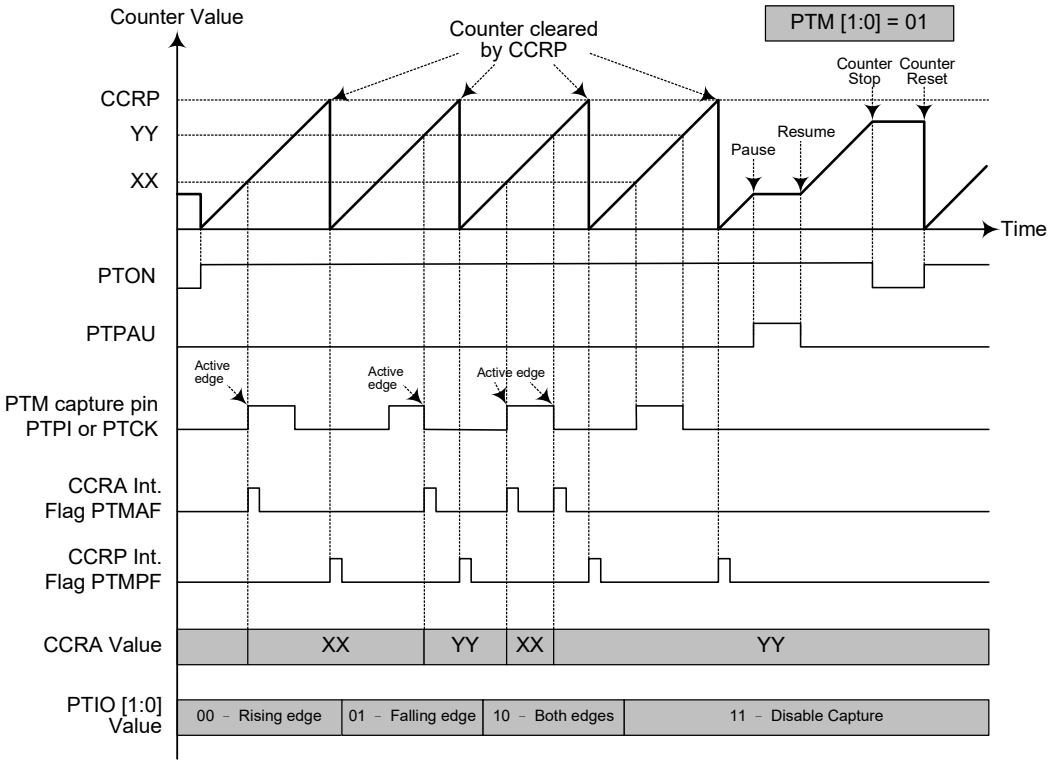
注：1. 通过 CCRA 匹配停止计数器
2. CCRP 未使用
3. 通过 PTCK 脚或设置 PTON 位为高来触发脉冲
4. PTCK 脚有效沿会自动置位 PTON
5. 单脉冲模式中，PTIO[1:0] 需置位“11”，且不能更改。

捕捉输入模式

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

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

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



捕捉输入模式

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

A/D 转换器

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

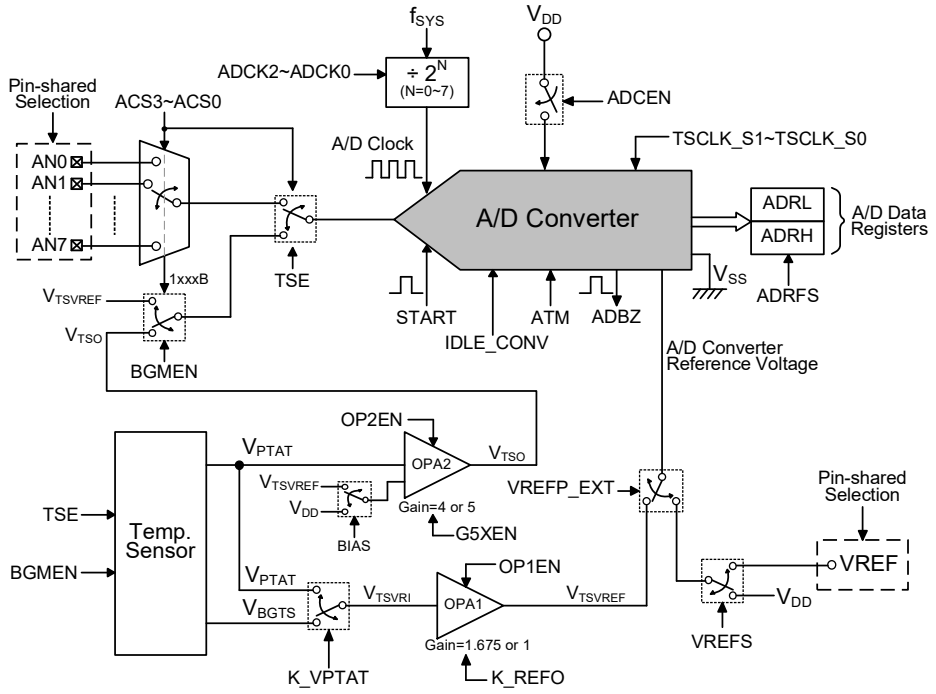
A/D 转换器简介

此系列单片机包含一个多通道的 A/D 转换器，它们可以直接接入外部模拟信号（来自传感器或其它控制信号）并直接将这些信号转换成 12 位的数字量。也可对内部模拟信号（来自温度传感器输出或温度传感器参考电压）进行 A/D 转换。选择转换外部或内部模拟信号由 TSE、BGMEN 位及 ACS3~ACS0 位共同控制。若要转换外部模拟信号时，需要先正确设置相关的引脚共用功能选择寄存器，然后再通过 ACS3~ACS0 位选择所需的外部通道输入。

此系列芯片内置的 A/D 转换器包含了一个温度传感器电路。温度传感器电路由一个温度传感器、运算放大器以及一个内部参考电压构成。温度传感器可感测温度，并将其输出为与温度成比例的电压信号。输出的电压信号通过 OPA 放大后，连接到 A/D 转换器转换为 12-bit 的数字信号。

下图显示了包含温度传感器的 A/D 转换器内部结构和相关的寄存器和控制位。

单片机型号	外部输入通道	内部信号	A/D 输入选择位
BS66F340 BS66F350 BS66F360 BS66F370	8: AN0~AN7	2: V_{TSO}, V_{TSVERF}	ACS3~ACS0 TSE, BGMEN



带温度传感器的 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八个寄存器控制。一对只读寄存器来存放 12 位 A/D 转换数据的值，两个控制寄存器，ADCR0 和 ADCR1，设置 A/D 转换器的操作和控制功能，剩下四个为温度传感器控制寄存器，用于选择要转换的温度传感器信号、参考电压源以及温度传感器转换时钟周期等。

寄存器名称	位							
	7	6	5	4	3	2	1	0
ADRL (ADRFS=0)	D3	D2	D1	D0	—	—	—	—
ADRL (ADRFS=1)	D7	D6	D5	D4	D3	D2	D1	D0
ADRH (ADRFS=0)	D11	D10	D9	D8	D7	D6	D5	D4
ADRH (ADRFS=1)	—	—	—	—	D11	D10	D9	D8
ADCR0	START	ADBZ	ADCEN	ADRFS	ACS3	ACS2	ACS1	ACS0
ADCR1	ATM	—	IDLE_CONV	VREFS	—	ADCK2	ADCK1	ADCK0
TSC0	BGMEN	G5XEN	K_REFO	—	—	—	—	—
TSC1	TSE	OP2EN	OP1EN	—	—	—	—	—
TSC2	VREFP_EXT	BIAS	D5	D4	D3	D2	TSCLK_S1	TSCLK_S0
TSC3	—	—	K_VPTAT	—	—	—	—	—

带温度传感器的 A/D 转换器寄存器列表

A/D 转换器数据寄存器 – ADRL, ADRH

对于具有 12 位 A/D 转换器的芯片，需要两个数据寄存器存放转换结果，一个高字节寄存器 ADRH 和一个低字节寄存器 ADRL。在 A/D 转换完毕后，单片机可以直接读取这些寄存器以获得转换结果。由于寄存器只使用了 16 位中的 12 位，其数据存储格式由 ADCR0 寄存器的 ADRFS 位控制，如下表所示。D0~D11 是 A/D 转换数据结果位。未使用的位读为“0”。注意，当通过设置 ADCR1 寄存器的 ATM 位为“1”，使能自动转换模式后，A/D 转换器数据寄存器的值只有进入 A/D 转换完成中断子程序才能读取。当 A/D 转换器除能时，数据寄存器的值将被清零。

ADRFS	ADRH								ADRL							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0
1	0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

A/D 转换器数据寄存器

A/D 转换控制寄存器 – ADCR0, ADCR1

寄存器 ADCR0 和 ADCR1 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，因此这些外部和内部模拟信号中的每一个都需要被分别发送到转换器。ADCR0 寄存器中的 ACS3~ACS0 位和 TSC1、TSC0 寄存器中的 TSE、BGMEN 位用于选择某个外部模拟输入还是内部温度传感器信号被连接到内部 A/D 转换器。当选择内部温度传感器模拟信号时，需要设置 ACS3~ACS0 位为“1xxx”并正确设置 TSE 和 BGMEN 位。

TSE	BGMEN	ACS[3:0]	输入信号	描述
0	x	x000~x111	AN0~AN7	外部模拟通道输入
1	0	1xxx	V_{TSO}	温度传感器输出电压
1	1	1xxx	V_{TSVREF}	温度传感器参考电压

A/D 转换器输入信号选择

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，其原来的通用 I/O 功能或其它引脚共用功能移除，此外，其内部上拉电阻也将自动断开。

- **ADCR0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	START	ADBZ	ADCEN	ADRFS	ACS3	ACS2	ACS1	ACS0
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
此位用于初始化 A/D 转换过程。通常此位为低，但如果设为高再被清零，将初始化 A/D 转换过程。

Bit 6 **ADBZ**：A/D 转换忙碌标志位
0：A/D 转换结束或未开始转换
1：A/D 转换中
此只读标志位用于表明 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 转换已初始化。A/D 转换结束后，此位被清零。

Bit 5 **ADCEN**：A/D 转换器使能 / 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 ADRH 和 ADRL 的内容将被清零。

Bit 4 **ADRFS**：A/D 转换数据格式选择位
0：A/D 转换数据格式 → ADRH=D[11:4]; ADRL=D[3:0]
1：A/D 转换数据格式 → ADRH=D[11:8]; ADRL=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 **ACS3~ACS0**：A/D 转换器模拟输入选择位
0000：AN0
0001：AN1
0010：AN2
0011：AN3
0100：AN4
0101：AN5
0110：AN6
0111：AN7
1xxx：来自温度传感器的内部信号－温度传感器输出电压或参考电压
只有当 TSE 位被置为“1”，“1xxx”的选择才是有效的。即若要选择温度传感器信号作为 A/D 转换器的输入，必须要将 TSE 位置高，同时设置 ACS3~ACS0 位为 1xxx。若没有将 TSE 位置高，而设置 ACS3~ACS0 位的值为“1xxx”，则会自动视 ACS3 的值为 0，根据“0xxx”选择连接到外部通道。

- **ADCR1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	ATM	—	IDLE_CONV	VREFS	—	ADCK2	ADCK1	ADCK0
R/W	R/W	—	R/W	R/W	—	R/W	R/W	R/W
POR	0	—	0	0	—	0	0	0

Bit 7 **ATM**：A/D 自动转换模式控制位
0：除能
1：使能
当此位置为“1”时，进入自动转换模式，A/D 转换器在完成当前转换后，无需通过应用程序配置“START”位，即可继续执行数据转换。

Bit 6 未定义，读为“0”

Bit 5 **IDLE_CONV**：CUP 空闲转换模式控制位
0：除能
1：使能
当此位置为“1”时，A/D 转换时 CPU 空闲模式使能。当设置 IDLE_CONV 位为“1”且 A/D 转换器未完成转换时，CUP 不工作。

Bit 4 **VREFS**：A/D 转换器参考电压选择位
0：A/D 转换器电源，V_{DD}
1：外部 VREF 引脚
此位用于选择 A/D 转换器的参考电压，但前提是 TSC2 寄存器中的 VREFP_EXT 位被置“1”。建议当选择内部温度传感器信号作为 A/D 转换输入时，保持 VREFP_EXT 为低状态。

Bit 3 未定义，读为“0”

Bit 2~0 **ADCK2~ADCK0**：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。建议在温度传感器使能且选择内部温度传感器信号作为 A/D 转换器输入时，正确配置 ADCK2~ADCK0 位，使得 A/D 转换时钟频率在 500kHz~1MHz 范围内。

- **TSC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	BGMEN	G5XEN	K_REFO	—	—	—	—	—
R/W	R/W	R/W	R/W	—	—	—	—	—
POR	0	1	0	—	—	—	—	—

Bit 7 **BGMEN**：温度传感器参考电压输出功能控制位
0：除能
1：使能
该位用于控制内部温度传感器参考电压输出功能，该位选择有效前提是 TSC1 寄存器中的 TSE 位被置“1”。当设置 ACS3~ACS0 位为“1xxx”且 TSE 和 BGMEN 位为“1”时，内部温度传感器参考电压作为 A/D 转换器输入。然而若 ACS3~ACS0 位为“1xxx” 且 TSE 位为“1”但 BGMEN 位被清零，将会是温度传感器输出电压作为 A/D 转换器输入。

Bit 6 **G5XEN**：OPA2 增益选择位
0：4 倍增益
1：5 倍增益
该位用于控制 OPA2 增益选择。针对不同应用温度范围，要合理设置此位，以避免出现饱和码。

Bit 5 **K_REFO**：OPA1 增益选择位
0：1.675 倍增益
1：1 倍增益
该位用于选择 OPA1 增益进而决定温度传感器参考电压输出值

Bit 4~0 未定义，读为“0”

- **TSC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	TSE	OP2EN	OP1EN	—	—	—	—	—
R/W	R/W	R/W	R/W	—	—	—	—	—
POR	0	0	0	—	—	—	—	—

Bit 7 **TSE**：温度传感器电路使能控制位
0：除能
1：使能
该位用于控制内部温度传感器电路。当设置 TSE 位为“1”使能温度传感器电路后，执行相关的温度传感器操作前，需要一段时间 t_{SS} 用于电路稳定。

Bit 6 **OP2EN**：温度传感器 OPA2 使能控制位
0：除能
1：使能

Bit 5 **OP1EN**：温度传感器 OPA1 使能控制位
0：除能
1：使能

Bit 4~0 未定义，读为“0”

- **TSC2 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	VREFP_EXT	BIAS	D5	D4	D3	D2	TSCLK_S1	TSCLK_S0
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 **VREFP_EXT**：A/D 转换器正极参考电压选择位
0：温度传感器参考电压－ V_{TSVREF}
1：由 VREFS 位选择
该位用于选择 A/D 转换器正极参考电压。当该位置“1”时，A/D 转换器参考电压由 ADCR1 寄存器中的 VREFS 位选择。若要选择 V_{TSVREF} 电压作为 A/D 转换器参考电压，应设置 VREFP_EXT 位为“0”同时也要合理配置 OPA2 输入信号及增益。

Bit 6 **BIAS**：OPA2 偏置电压选择位
0：V_{TSVREF}
1：A/D 转换器电源

Bit 5~2 **D5~D2**：内部使用位
这些位需保持为“0”且不能改变。

Bit 1~0 **TSCLK_S1~ TSCLK_S0**：温度传感器时钟源（t_{TSCLK}）选择位
00：$t_{TSCLK} = t_{ADCK}/4$
01：$t_{TSCLK} = t_{ADCK}/8$
10：$t_{TSCLK} = t_{ADCK}/16$
11：$t_{TSCLK} = t_{ADCK}/16$
温度传感器信号转换时间可通过下面公式计算得到：
温度传感器信号转换时间＝ $(5*N+1+16) * t_{ADCK}$
上述公式中的“N”表示分频率（4、8 或 16），由 TSCLK_S1~ TSCLK_S0 位选择。

- **TSC3 寄存器**

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

Bit 7~6 未定义，读为“0”

Bit 5 **K_VPTAT**：OPA1 输入电压选择位
0：V_{BG}
1：V_{PTAT}
该位用于选择 OPA1 输入电压从而得到内部温度传感器参考电压 V_{TSVREF}。

Bit 4~0 未定义，读为“0”

A/D 转换器操作

ADCR0 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。

ADCR0 寄存器中的 ADBZ 位用于表明模数转换过程是否正在进行。A/D 转换成功启动后，ADBZ 位会被单片机自动置为“1”。在转换周期结束后，ADBZ 位会自动清“0”。此外，还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 内部中断除能，可以让单片机轮询 ADCR0 寄存器中的 ADBZ 位，检查此位是否被清除，作为另一种侦测 A/D 转换周期结束的方法。

A/D 转换器的时钟源为系统时钟 f_{SYS} 或其分频，而分频系数由 ADCR1 寄存器中的 ADCK2~ADCK0 位决定。虽然 A/D 时钟源是由系统时钟 f_{SYS} 和 ADCK2~ADCK0 位决定，但可选择的最大 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。如果系统时钟速度为 8MHz 时，ADCK2~ADCK0 位不能设为“000”、“001”或“111”。必须保证设置的 A/D 转换时钟周期不小于时钟周期的最小值或大于时钟周期的最大值，否则将会产生不准确的 A/D 转换值。使用者可以参考下面的表格，被标上星号 * 的数值是不允许的，因超出了规定的范围。

若 A/D 转换器要转换的信号为温度传感器输出电压或参考电压，建议的 A/D 时钟周期值范围 1μs~2μs。

f_{SYS}	A/D 时钟周期 (t_{ADCK})							
	ADCK[2:0] = 000 (f_{SYS})	**ADCK[2:0] = 001 (f_{SYS}/2)**	**ADCK[2:0] = 010 (f_{SYS}/4)**	**ADCK[2:0] = 011 (f_{SYS}/8)**	**ADCK[2:0] = 100 (f_{SYS}/16)**	**ADCK[2:0] = 101 (f_{SYS}/32)**	**ADCK[2:0] = 110 (f_{SYS}/64)**	**ADCK[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
20MHz	50ns *	100ns *	200ns *	400ns *	800ns	1.6μs	3.2μs	6.4μs

A/D 时钟周期范例

ADCR0 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，如时序图所示。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 转换器参考电压

A/D 转换器参考电压来自正电源电压引脚 VDD、外部参考源引脚 VREF 或内部温度传感器参考电压 V_{TSVREF}。内部温度传感器参考电压又可来自内部 V_{BG} 或 V_{PTAT} 电压，可通过 TSC3 寄存器中的 K_VPTAT 位选择，并通过可编程增益放大器进行放大，PGA 增益可以为 1.675 或 1，通过 TSC0 寄存器中的 K_REFO 位来选择。注意由于 VREF 引脚都与其它功能共用，当选择 VREF 参考电压时，需合理设置相关引脚共用控制位选择 VREF 引脚功能且除能其它共用引脚功能。

A/D 输入引脚

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 PxS0 和 PxS1 寄存器中的相应位，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 转换器有自己的参考电压引脚 VREF，而通过设置 TSC2 和 ADCR1 寄存器中的 VREFP_EXT 和 VREFS 位，参考电压也可以选择来自电源电压引脚或内部温度传感器电路。需注意的是模拟输入值一定不能超过选择的参考电压值。

A/D 转换率及时序图

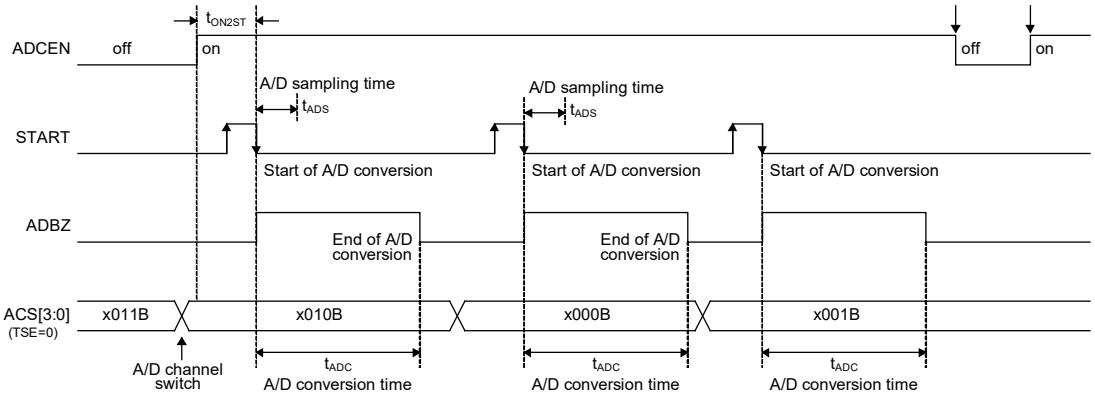
对于外部通道输入信号，一个完整的 A/D 转换包含两部分，数据采样和数据转换。数据采样时间定义为 t_{ADS}，需要 4 个 A/D 时钟周期，而数据转换需要 12 个 A/D 时钟周期。所以一个完整的 A/D 转换时间，t_{ADC}，一共需要 16 个 A/D 时钟周期。

对于内部温度传感器信号，一个完整的 A/D 转换时间一共需要 56 个时钟周期。

最大 A/D 转换率 =A/D 时钟周期 ÷16（外部通道输入信号）

最大 A/D 转换率 =A/D 时钟周期 ÷56（内部温度传感器信号）

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



A/D 转换时序图－外部通道输入

A/D 转换步骤

下面概述实现 A/D 转换过程的各个步骤。

- 步骤 1
  通过 ADCR1 寄存器中的 ADCK2~ADCK0 位，选择所需的 A/D 转换时钟。
- 步骤 2
  将 ADCR0 寄存器中的 ADCEN 位置高使能 A/D。
- 步骤 3
  通过 ADCR1 寄存器中的 ACS3~ACS0 位，选择连接至内部 A/D 转换器的信号。
  若选择外部通道输入，设置 TSE 为 0，ACS3~ACS0 为 x000~x111。
  若选择内部温度传感器相关信号，设置 TSE 为 1，ACS3~ACS0 为 1xxx。
- 步骤 4
  通过 K_VPTAT、K_REFO 和 VREFS 位，选择参考电压源。
- 步骤 5
  设置 ADRFS 位选择 A/D 转换器输出数据格式。
- 步骤 6
  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 7
  现在可以通过设置 ADCR0 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 8
  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 ADRH 和 ADRL 寄存器中读取输出数据。

注：若使用轮询 ADCR0 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。但若使能自动转换模式，必须使用中断方式用于查询转换结束及读取对应的转换值。

编程注意事项

在编程时，如果A/D转换器未使用，通过设置ADCR0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

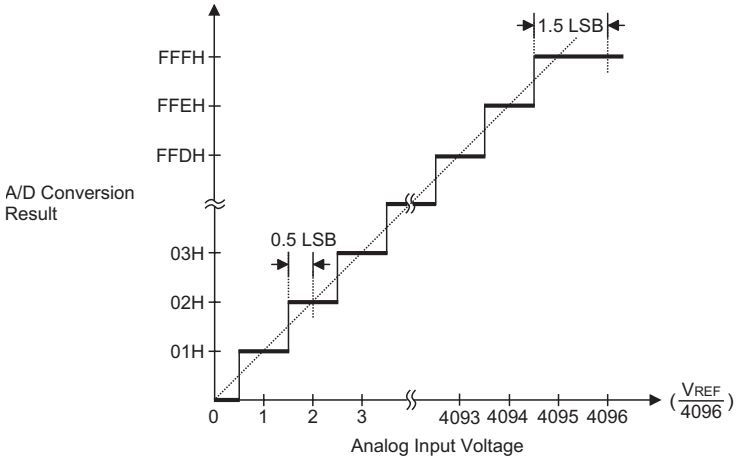
单片机含有一组 12 位的 A/D 转换器，它们转换的最大值可达 FFFH。由于模拟输入最大值等于 V_{REF} 的电压值，因此每一位可表示 V_{REF}/4096 的模拟输入值。

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

通过下面的等式可估算 A/D 转换器输入电压值：

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

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



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 ADCR0 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
SET VREFP_EXT       ; deselect the temperature sensor reference voltage
mov a,03H           ; select fSYS/8 as A/D clock and A/D internal power
                    ; supply
mov ADCR1,a         ; as reference voltage
set ADCEN
mov a,03H           ; setup PBS0 to configure pin AN0
mov PBS0,a
mov a, 20H
mov ADCR0,a         ; enable 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 ADCR0 register ADBZ bit to detect end of A/D
                    ; conversion
jmp polling_EOC     ; continue polling
:
mov a,ADRL          ; read low byte conversion result value
mov ADRL_buffer,a   ; save result to user defined register
mov a,ADRH          ; read high byte conversion result value
mov ADRH_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
set VREFP_EXT       ; deselect the temperature sensor reference voltage
mov a,03H           ; select fSYS/8 as A/D clock and A/D internal power
                    ; supply
mov ADCR1,a         ; as reference voltage
set ADCEN
mov a,03H           ; setup PBS0 to configure pin AN0
mov PBS0,a
mov a, 20H
mov ADCR0,a         ; enable 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, ADRL         ; read low byte conversion result value
mov ADRL_buffer,a   ; save result to user defined register
mov a, ADRH         ; read high byte conversion result value
mov ADRH_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
```

SIM 串行接口模块

此系列单片机内有一个串行接口模块，包括两种易与外部设备通信的串行接口：四线 SPI 或两线 I^2C 接口。这两种接口具有相当简单的通信协议，单片机可以通过这些接口与传感器、闪存或 EEPROM 内存等硬件设备通信。因 SIM 接口引脚是与其它 I/O 引脚共用，因此在使用 SIM 功能前，要先通过相应的引脚共用功能选择寄存器选定 SIM 引脚功能。因为这两种接口 SPI 和 I^2C 共用引脚和寄存器，所以要通过一个 SIMC0 寄存器中的 SIM2~SIM0 位来选择哪一种通信接口。若 SIM 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 SIM 输入引脚的上拉电阻。

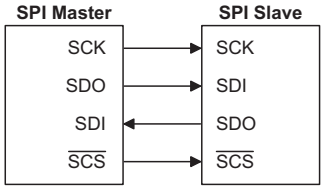
SPI 接口

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

SPI 通信模式为全双工模式，且能以主 / 从模式的工作方式进行通信，单片机既可以做为主机，也可以做为从机。虽然 SPI 接口理论上允许一个主机控制多个从机，但此处的 SPI 中只有一个片选信号引脚 $\overline{SCS}$。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 的功能脚共用。通过设定 SIMC0/SIMC2 寄存器的对应位，来使能 SPI 接口。SPI 可以通过 SIMC0 寄存器中的 SIMEN 位来除能或使能。连接到 SPI 接口的单片机以从主 / 从模式进行通信，且主机完成所有的数据传输初始化，并控制时钟信号。由于单片机只有一个 $\overline{SCS}$ 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。

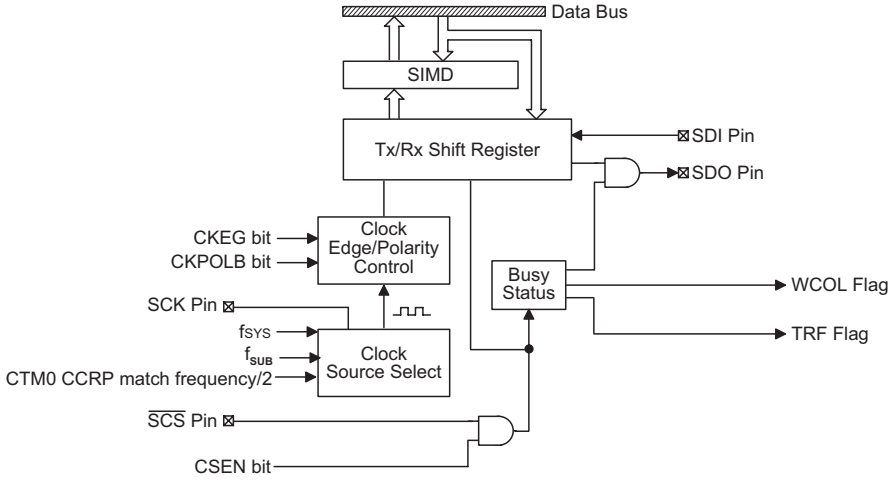


SPI 主 / 从机连接方式

该系列单片机的 SPI 功能具有以下特点：

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

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



SPI 方框图

SPI 寄存器

有三个内部寄存器用于控制 SPI 接口的所有操作，其中有一个数据寄存器 SIMD、两个控制寄存器 SIMC0 和 SIMC2。注意，SIMC1 寄存器仅用于 I^2C 接口。

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

SPI 寄存器列表

SIMD 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。在单片机尚未将数据写入到 SPI 总线中时，要传输的数据应先存在 SIMD 中。SPI 总线接收到数据之后，单片机就可以从 SIMD 数据寄存器中读取。所有通过 SPI 传输或接收的数据都必须通过 SIMD 实现。

- **SIMD 寄存器**

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

“x”为未知

单片机中也有两个控制 SPI 接口功能的寄存器，SIMC0 和 SIMC2。应注意的是 SIMC2 与 I^2C 接口功能中的寄存器 SIMA 是同一个寄存器。SPI 功能不会用到寄存器 SIMC1，SIMC1 只适用于 I^2C 中。寄存器 SIMC0 用于控制使能 / 除能功能和设置数据传输的时钟频率。寄存器 SIMC2 用于其它的控制功能如 LSB/MSB 选择，写冲突标志位等。

- **SIMC0 寄存器**

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

Bit 7~5 **SIM2~SIM0**：SIM 工作模式控制位
000：SPI 主机模式；SPI 时钟为 $f_{SYS}/4$
001：SPI 主机模式；SPI 时钟为 $f_{SYS}/16$
010：SPI 主机模式；SPI 时钟为 $f_{SYS}/64$
011：SPI 主机模式；SPI 时钟为 f_{SUB}
100：SPI 主机模式；SPI 时钟为 CTM0 CCRP 匹配频率 /2
101：SPI 从机模式
110：I^2C 从机模式
111：非 SIM 功能
这几位用于设置 SIM 功能的工作模式，用于选择 SPI 的主从模式和 SPI 的主机时钟频率及 I^2C 或 SPI 功能。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 未定义，读为“0”

Bit 3~2 **SIMDEB1~SIMDEB0**：I^2C 去抖时间选择位
详见其它章节

Bit 1 **SIMEN**：SIM 控制位
0：除能
1：使能
此位为 SIM 接口的开 / 关控制位。此位为“0”时，SIM 接口除能，SDI、SDO、SCK 和 SCS 或 SDA 和 SCL 脚将失去 SPI 或 I^2C 功能，SIM 工作电流减小到最小值。此位为“1”时，SIM 接口使能。若 SIM 经由 SIM2~SIM0 位设置为工作在 SPI 接口，当 SIMEN 位由低到高转变时，SPI 控制寄存器中的设置不会发生变化，其首先应在应用程序中初始化。若 SIM 经由 SIM2~SIM0 位设置为工作在 I^2C 接口，当 SIMEN 位由低到高转变时，I^2C 控制寄存器中的设置，如 HXT 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 I^2C 标志，如 HCF、HAAS、HBB、SRW 和 RXAK，将被设置为其默认状态。

Bit 0 **SIMICF**：SIM 未完成标志位
0：未发生
1：发生
此位仅当 SIM 配置在 SPI 从机模式时有效。如果 SPI 工作在从机模式且 SIMEN 和 CSEN 位都为“1”，但在 SPI 数据传输完全结束前 SCS 线被外部主机拉高，SIMICF 和 TRF 位都会被置高。在这种情况下，如果相应的中断功能使能将产生一个中断。然而，如果 SIMICF 位是由软件应用程序设为 1，那么 TRF 位将不会置高。

- **SIMC2 寄存器**

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

Bit 7~6 未定义位
用户可通过软件程序对这两位进行读写。

Bit 5 **CKPOLB**：时钟线的基础状态位
0：当时钟无效时，SCK 口为高电平
1：当时钟无效时，SCK 口为低电平
此位决定了时钟线的基础状态，当时钟无效时，若此位为高，SCK 为低电平，若此位为低，SCK 为高电平。

Bit 4 **CKEG**：SPI 的 SCK 有效时钟边沿类型位
CKPOLB=0
0：SCK 为高电平且在 SCK 上升沿抓取数据
1：SCK 为高电平且在 SCK 下降沿抓取数据
CKPOLB=1
0：SCK 为低电平且在 SCK 下降沿抓取数据
1：SCK 为低电平且在 SCK 上升沿抓取数据
CKEG 和 CKPOLB 位用于设置 SPI 总线上时钟信号输入和输出方式。在执行数据传输前，这两位必须被设置，否则将产生错误的时钟边沿信号。CKPOLB 位决定时钟线的基本状态，若时钟无效且此位为高，则 SCK 为低电平，若时钟无效且此位为低，则 SCK 为高电平。CKEG 位决定有效时钟边沿类型，取决于 CKPOLB 的状态。

Bit 3 **MLS**：SPI 数据移位命令位
0：LSB
1：MSB
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 $\overline{\text{SCS}}$ 引脚控制位
0：除能
1：使能
CSEN 位用于 $\overline{\text{SCS}}$ 引脚的使能 / 除能控制。此位为低时，$\overline{\text{SCS}}$ 除能并处于浮空状态。此位为高时，SCS 作为选择脚。

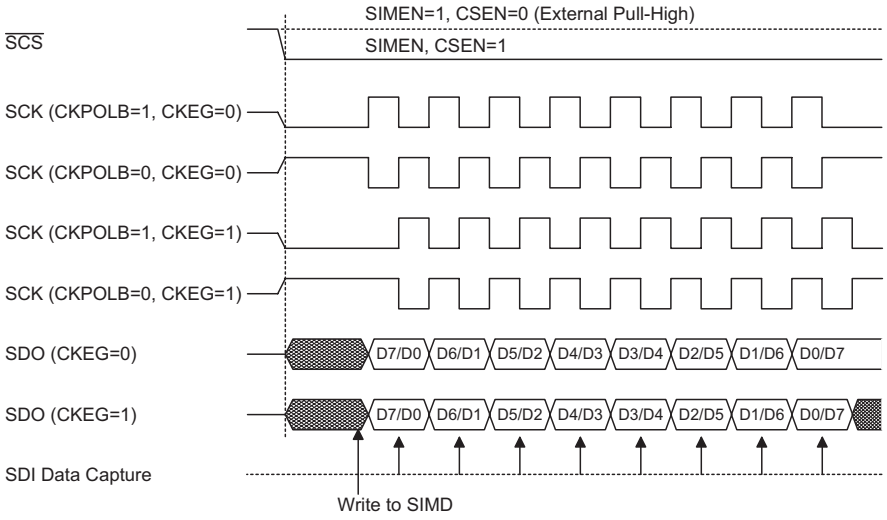
Bit 1 **WCOL**：SPI 写冲突标志位
0：无冲突
1：冲突
WCOL 标志位用于监测数据冲突的发生。此位为高时，数据在传输时被写入 SIMD 寄存器。若数据正在被传输时，此操作无效。此位可被应用程序清零。

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

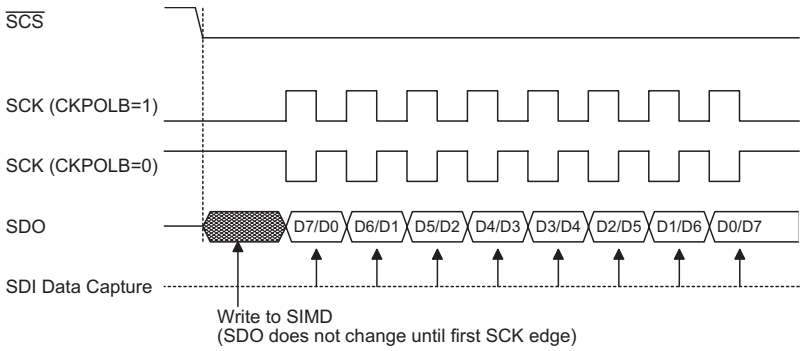
SPI 通信

将 SIMEN 设置为高，使能 SPI 功能之后，单片机处于主机模式，当数据写入到寄存器 SIMD 的同时传输 / 接收开始进行。数据传输完成时，TRF 位将自动被置位但清除只能通过应用程序完成。单片机处于从机模式时，收到主机发来的信号之后，会传输 SIMD 中的数据，而且在 SDI 引脚上的数据也会被移位到 SIMD 寄存器中。主机应在输出时钟信号之前先输出一个 $\overline{SCS}$ 信号以使能从机，从机的数据传输功能也应在与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 $\overline{SCS}$ 信号的关系。

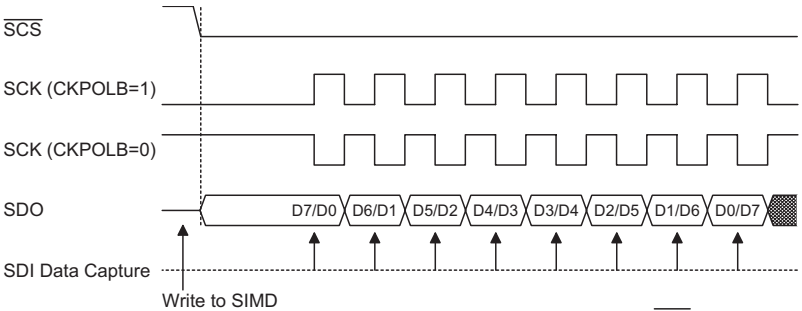
即使在单片机处于空闲模式，SPI 功能仍将继续执行。



SPI 主机模式时序

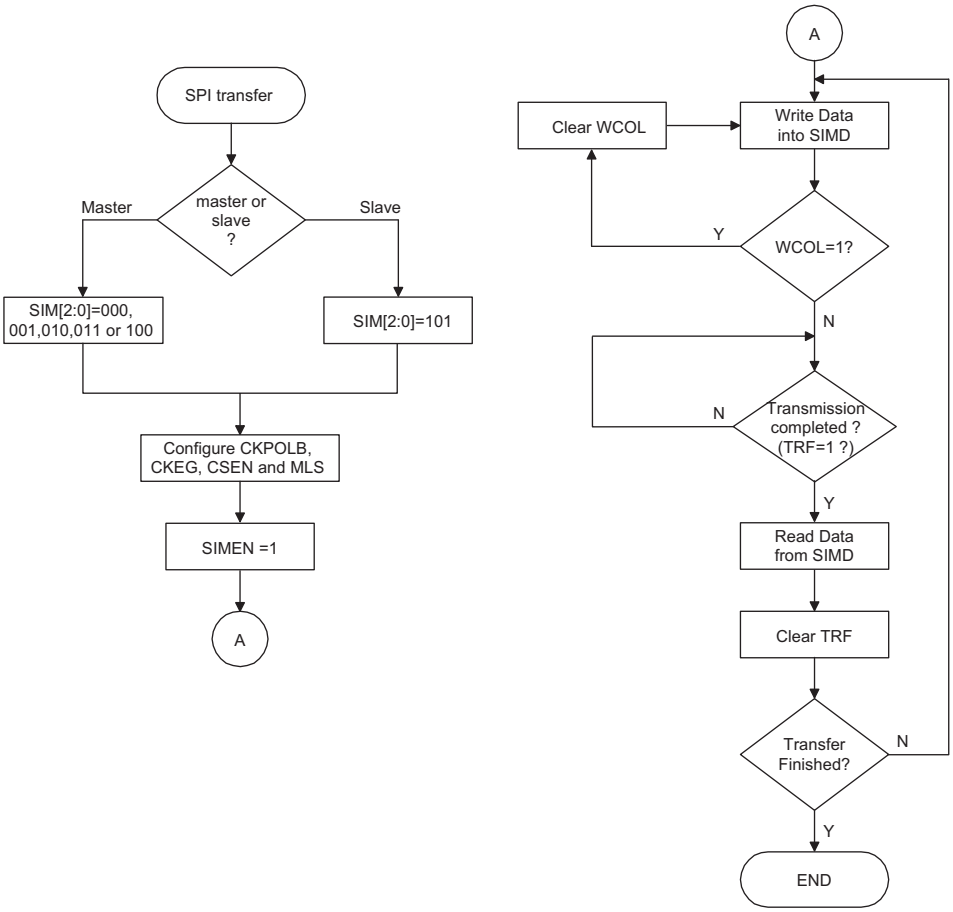


SPI 从机模式时序 – CKEG=0



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

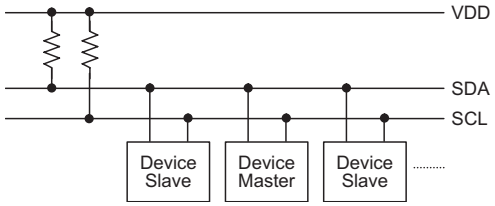
SPI 从机模式时序 – CKEG=1



SPI 传输控制流程图

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。

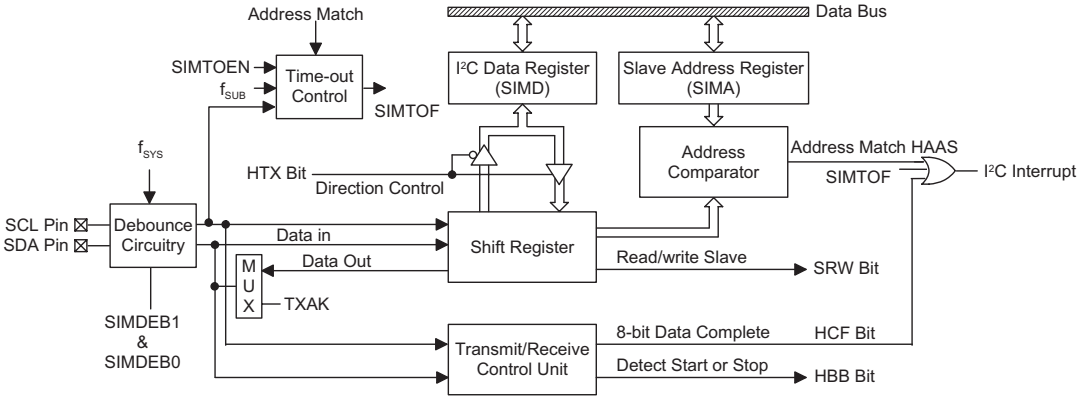


I²C 主从总线连接图

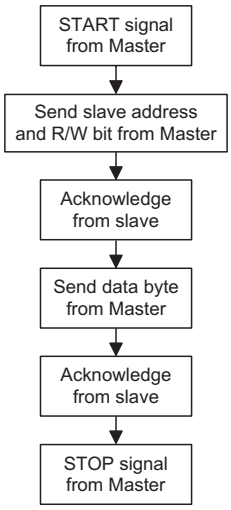
I²C 接口操作

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，上拉电阻控制功能和 SCL/SDA 引脚功能仍有效，其上拉电阻功能由相关上拉电阻控制寄存器控制。



I²C 方框图



SIMDEB1 和 SIMDEB0 位决定 I^2C 接口的去抖时间。这个功能可以使用内部时钟在外部时钟上增加一个去抖间隔，会减小时钟线上毛刺发生的可能性，以避免单片机发生误动作。如果选择了这个功能，去抖时间可以选择 2 个或 4 个系统时钟。为了达到需要的 I^2C 数据传输速度，系统时钟 f_{SYS} 和 I^2C 去抖时间之间存在一定的关系。I^2C 标准模式或者快速模式下，用户需注意所选的系统时钟频率与标准匹配去抖时间的设置，其具体关系如下表所示。

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

I^2C 最小 f_{SYS} 频率

I^2C 寄存器

I^2C 总线有三个控制寄存器 SIMC0、SIMC1 和 SIMA，及一个数据寄存器 SIMD。SIMD 寄存器，SPI 章节中已有介绍，用于存储正在传输和接收的数据。应注意的是 SIMA 也有另外一个名字，SIMC2，使用 SPI 功能时会用到。I^2C 接口会用到寄存器 SIMC0 中的 SIMEN 位和 SIM2~SIM0 位。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	SIMICF
SIMC1	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
SIMA	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	D0
SIMD	D7	D6	D5	D4	D3	D2	D1	D0
SIMTOC	SIMTOEN	SIMTOF	SIMTOS5	SIMTOS4	SIMTOS3	SIMTOS2	SIMTOS1	SIMTOS0

I^2C 寄存器列表

- **SIMD 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。在单片机尚未将数据写入到 I^2C 总线中时，要传输的数据应存在 SIMD 中。I^2C 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 I^2C 传输或接收的数据都必须通过 SIMD 实现。

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”为未知

- **SIMA 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 I^2C 的主机发送处的地址和寄存器 SIMA 中存储的地址相符，那么就选中了这个从机。应注意的是寄存器 SIMA 和 SPI 接口使用的寄存器 SIMC2 是同一个寄存器。

Bit	7	6	5	4	3	2	1	0
Name	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	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 **IICA6~IICA0**：I^2C 从机地址位
IICA6~IICA0 是从机地址 bit 6 ~ bit 0。

Bit 0 无定义
此位可通过软件程序进行读写。

单片机中也有两个控制 I^2C 接口功能的寄存器，SIMC0 和 SIMC1。寄存器 SIMC0 用于控制使能 / 除能功能和设置数据传输的时钟频率。寄存器 SIMC1 包括多个用于表明 I^2C 传输状态的相关标志位。

- **SIMC0 寄存器**

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

Bit 7~5 **SIM2~SIM0**：SIM 工作模式控制位
000：SPI 主机模式；SPI 时钟为 $f_{SYS}/4$
001：SPI 主机模式；SPI 时钟为 $f_{SYS}/16$
010：SPI 主机模式；SPI 时钟为 $f_{SYS}/64$
011：SPI 主机模式；SPI 时钟为 f_{SUB}
100：SPI 主机模式；SPI 时钟为 CTM0 CCRP 匹配频率 /2
101：SPI 从机模式
110：I^2C 从机模式
111：未使用模式
这几位用于设置 SIM 功能的工作模式，用于选择 SPI 的主从模式和 SPI 的主机时钟频率及 I^2C 或 SPI 功能。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　未定义，读为“0”

Bit 3~2　**SIMDEB1~SIMDEB0**：I^2C 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

Bit 1　**SIMEN**：SIM 控制位
0：除能
1：使能
此位为 SIM 接口的开 / 关控制位。此位为“0”时，SIM 接口除能，SDI、SDO、SCK 和 SCS 或 SDA 和 SCL 脚将失去 SPI 或 I^2C 功能，SIM 工作电流减小到最小值。此位为“1”时，SIM 接口使能。若 SIM 经由 SIM2~SIM0 位设置为工作在 SPI 接口，当 SIMEN 位由低到高转变时，SPI 控制寄存器中的设置不会发生变化，其首先应在应用程序中初始化。若 SIM 经由 SIM2~SIM0 位设置为工作在 I^2C 接口，当 SIMEN 位由低到高转变时，I^2C 控制寄存器中的设置，如 HXT 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 I^2C 标志，如 HCF、HAAS、HBB、SRW 和 RXAK，将被设置为其默认状态。

Bit 0　**SIMICF**：SIM 未完成标志位
详见其他章节

- **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**：I^2C 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**：I^2C 地址匹配标志位
0：地址不匹配
1：地址匹配
此标志位用于决定从机地址是否与主机发送地址相同。若地址匹配此位为高，否则此位为低。

Bit 5　**HBB**：I^2C 总线忙标志位
0：I^2C 总线闲
1：I^2C 总线忙
当检测到 START 信号时 I^2C 忙，此位变为高电平。当检测到 STOP 信号时 I^2C 总线停止，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**：I^2C 总线发送确认标志位
0：从机发送确认标志
1：从机没有发送确认标志
单片机接收 8 位数据之后会将该位在第九个时钟传到总线上。如果单片机想要接收更多的数据，则应在接收数据之前将此位设置为“0”。

Bit 2 **SRW**：I²C 从机读 / 写位
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，主机将检测 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，此时设备处于传输模式。当 SRW 位为“0”时，主机往总线上写数据，设备处于接收模式以读取该数据。

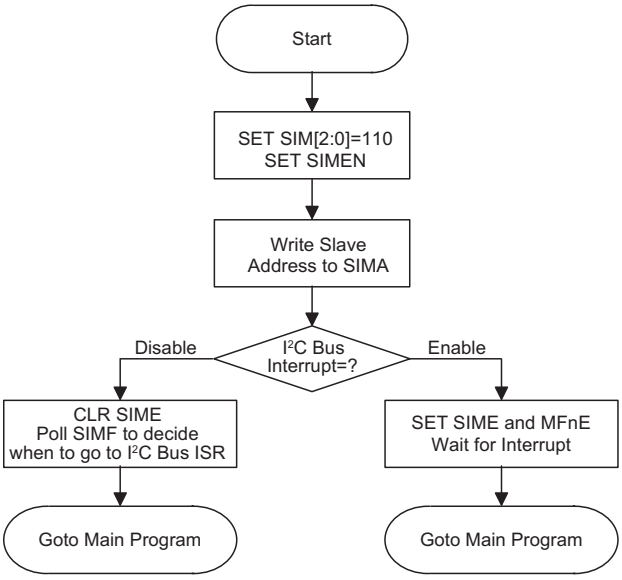
Bit 1 **IAMWU**：I²C 地址匹配唤醒控制位
0：除能
1：使能
此位应设置为“1”使能 I²C 地址匹配以使系统从休眠或空闲模式中唤醒。若进入休眠或空闲模式前 IAMWU 已经设置以使能 I²C 地址匹配唤醒功能，在系统唤醒后须软件清除此位以确保单片机正确地运行。

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

I²C 总线通信

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

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



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 总线主机产生，而不是由只做从机的 MCU 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。

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 I²C 总线中断信号。地址位接下来的一位为读 / 写状态位（即第 8 位），将被保存到 SIMC1 寄存器的 SRW 位，随后发出一个低电平应答信号（即第 9 位）。当单片机从机的地址匹配时，会将状态标志位 HAAS 置位。

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

I²C 总线读 / 写信号

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

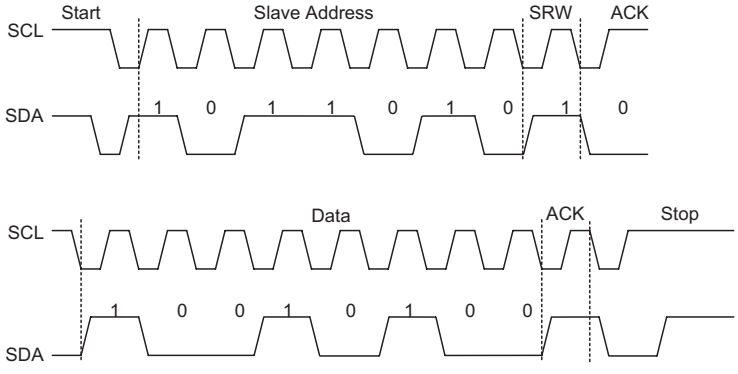
I²C 总线从机地址确认信号

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

I²C 总线数据和确认信号

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

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

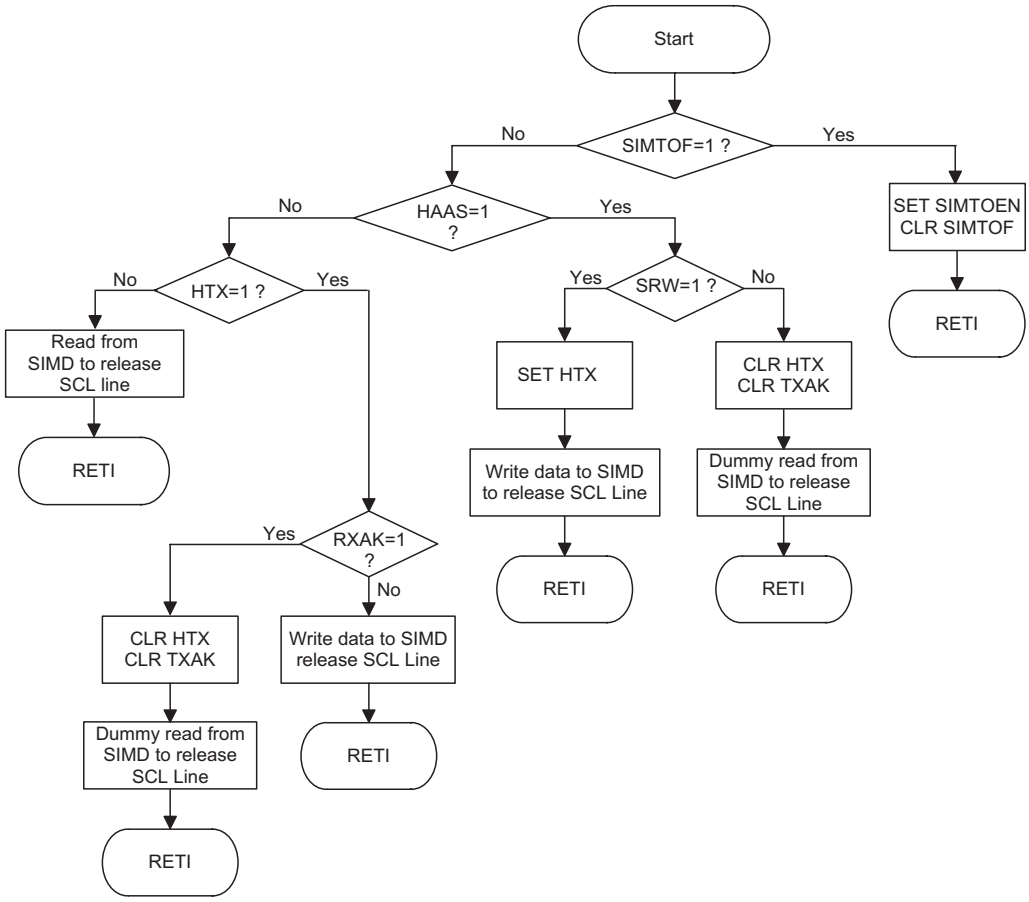


S=Start (1 bit)
SA=Slave Address (7 bits)
SR=SRW bit (1 bit)
M=Slave device send acknowledge bit (1 bit)
D=Data (8 bits)
A=ACK (RXAK bit for transmitter, TXAK bit for receiver 1 bit)
P=Stop (1 bit)

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。

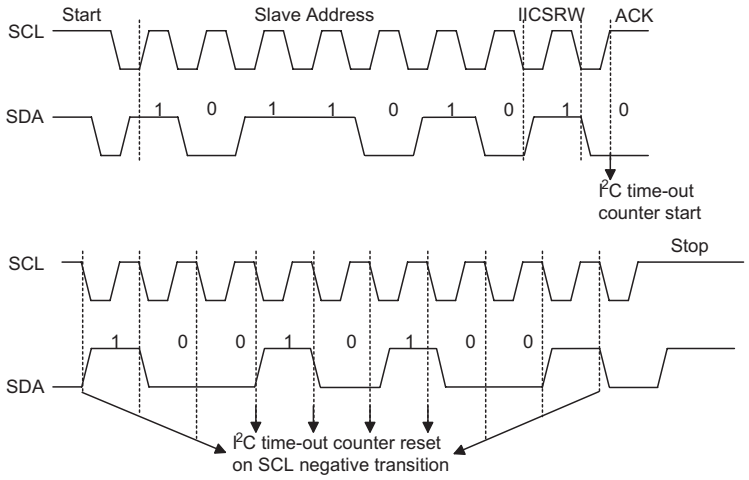
I²C 通信时序图



I^2C 总线 ISR 流程图

I^2C 超时控制

超时功能可减少 I^2C 接收错误的时钟源而引起的锁死问题。如果连接到 I^2C 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，I^2C 电路和寄存器将复位。超时计数器在 I^2C 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。I^2C“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 I²C 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**：I²C 超时控制位
0：除能
1：使能

Bit 6 **SIMTOF**：I²C 超时标志位
0：未发生
1：发生

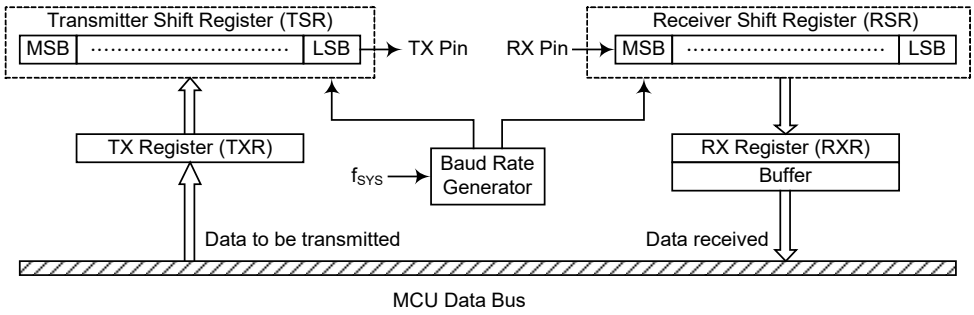
Bit 5~0 **SIMTOS5~SIMTOS0**：I²C 超时时间选择位
I²C 超时时钟源是 f_{SUB}/32
I²C 超时时间计算方法：([SIMTOS[5:0]+1) × (32/f_{SUB})

UART 模块串行接口

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

UART 模块特性如下：

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



UART 数据传输方框图

UART 外部引脚接口

内部 UART 有两个外部引脚 TX 和 RX，可与外部串行接口进行通信。TX 和 RX 引脚相应的为 UART 接口的发送和接收引脚。在使用 UART 功能之前，要先正确设置相关的引脚共用功能选择寄存器选择 RX 和 TX 引脚功能。若 UARTEN 位和 TXEN/RXEN 位均为“1”时，将会自动设置 TX 脚为输出，RX 为输入状态，同时断开 RX 和 TX 引脚上的上拉电阻。若 UARTEN 位或 TXEN/RXEN 位为“0”时，TX 和 RX 引脚功能除能，若未选择其它共用的引脚功能，这两个引脚将处于浮空状态。同时，这两个引脚是否连接上拉电阻由对应的 PxPUn 位控制。

UART 数据传输方案

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

数据在波特率发生器的控制下，低位在前高位在后，从外部引脚 RX 进入接收移位寄存器 RSR。当数据接收完成，数据从接收移位寄存器移入可被用户程序操作的 RXR 寄存器中。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，上述发送寄存器 TXR 和接收寄存器 RXR，其实是共享一个地址的数据寄存器 TXR_RXR 寄存器。

UART 状态和控制寄存器

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

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

UART 寄存器列表

TXR_RXR 寄存器

TXR_RXR 寄存器为数据寄存器，用来存储发送到 TX 引脚或从 RX 引脚接收到的数据。

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

“x”表示未知

Bit 7~0　　**TXRX7~TXRX0**：UART 发送 / 接收数据位 Bit 7~Bit 0

USR 寄存器

寄存器USR是UART的状态寄存器，可以通过程序读取从判断当前UART状态。USR 寄存器中所有位是只读的。详细解释如下：

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

Bit 7 **PERR**：奇偶校验出错标志位

0: 奇偶校验正确

1: 奇偶校验出错

PERR 是奇偶校验出错标志位。若 PERR=0，奇偶校验正确；若 PERR=1，接收到的数据奇偶校验出错。只有使能了奇偶校验此位才有效。可使用软件清除该标志位，即先读取 USR 寄存器再读 RXR 寄存器来清除此位。

Bit 6 **NF**：噪声干扰标志位

0: 没有受到噪声干扰

1: 受到噪声干扰

NF 是噪声干扰标志位。若 NF=0，没有受到噪声干扰；若 NF=1，UART 接收数据时受到噪声干扰。它与 RXIF 在同周期内置位，但不会与溢出标志位同时置位。可使用软件清除该标志位，即先读取 USR 寄存器再读 RXR 寄存器将清除此标志位。

Bit 5 **FERR**：帧错误标志位

0: 无帧错误发生

1: 有帧错误发生

FREE 是帧错误标志位。若 FREE=0，没有帧错误发生；若 FREE=1，当前的数据发生了帧错误。可使用软件清除该标志位，即先读取 USR 寄存器再读 RXR 寄存器来清除此位。

Bit 4 **OERR**：溢出错误标志位

0: 无溢出错误发生

1: 有溢出错误发生

OERR 是溢出错误标志位，表示接收缓冲器是否溢出。若 OERR=0，没有溢出错误；若 OERR=1，发生了溢出错误，它将影响下一组数据的接收。可通过软件清除该标志位，即先读取 USR 寄存器再读 RXR 寄存器将清除此标志位。

Bit 3 **RIDLE**：接收状态标志位

0：正在接收数据

1：接收器空闲

RIDLE 是接收状态标志位。若 RIDLE=0，表明正在接收数据；若 RIDLE=1，表明接收器空闲。在接收到停止位，还未接收到下一个数据的起始位之间，RIDLE 被置位，表明 UART 空闲，RX 脚处于逻辑高状态。

Bit 2 **RXIF**：接收寄存器状态标志位

0：RXR 寄存器为空

1：RXR 寄存器含有有效数据

RXIF 是接收寄存器状态标志位。当 RXIF=0，RXR 寄存器为空；当 RXIF=1，RXR 寄存器接收到新数据。当数据从移位寄存器加载到 RXR 寄存器中，如果 UCR2 寄存器中的 RIE=1，则会触发中断。当接收数据时检测到一个或多个错误时，相应的标志位 NF、FERR 或 PERR 会在同一周期内置位。读取 USR 寄存器再读 RXR 寄存器，如果 RXR 寄存器中没有新的数据，那么将清除 RXIF 标志。

Bit 1 **TIDLE**：数据发送完成标志位
0：数据传输中
1：无数据传输
TIDLE 是数据发送完成标志位。若 TIDLE=0，数据传输中。当 TXIF=1 且数据发送完毕或者暂停字被发送时，TIDLE 置位。TIDLE=1，TX 引脚空闲且处于逻辑高状态。读取 USR 寄存器再写 TXR 寄存器将清除 TIDLE 位。数据字符或暂停字就绪时，不会产生该标志位。

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

UCR1 寄存器

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

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

“x”表示未知

Bit 7 **UARTEN**：UART 功能使能位
0: UART 除能，TX 和 RX 脚处于浮空状态
1: UART 使能，TX 和 RX 脚作为 UART 功能引脚
此位为 UART 的使能位。UARTEN=0，UART 除能，RX 和 TX 处于浮空状态；UARTEN=1，UART 使能，TX 和 RX 将分别由 TXEN 和 RXEN 控制。当 UART 被除能将清除缓冲器，所有缓冲器中的数据将被忽略，另外波特率计数器、错误和状态标志位被复位，TXEN、RXEN、TXBRK、RXIF、OERR、FERR、PERR 和 NF 清零而 TIDLE、TXIF 和 RIDLE 置位，UCR1、UCR2 和 BRG 寄存器中的其它位保持不变。若 UART 工作时 UARTEN 清零，所有发送和接收将停止，模块也将复位成上述状态。当 UART 再次使能时，它将在上次配置下重新工作。

Bit 6 **BNO**：发送数据位数选择位
0: 8-bit 传输数据
1: 9-bit 传输数据
BNO 是发送数据位数选择位。BNO=1，传输数据为 9 位；BNO=0，传输数据为 8 位。若选择了 9 位数据传输格式，RX8 和 TX8 将分别存储接收和发送数据的第 9 位。

Bit 5 **PREN**：奇偶校验功能控制位
0: 除能
1: 使能
此位为奇偶校验功能控制位。PREN=1，使能奇偶校验；PREN=0，除能奇偶校验。

Bit 4 **PRT**：奇偶校验类型选择位
0: 偶校验
1: 奇校验
奇偶校验类型选择位。PRT=1，奇校验；PRT=0，偶校验。

Bit 3 **STOPS**：停止位的长度选择位
0：有一位停止位
1：有两位停止位
此位用来设置停止位的长度。STOP=1，有两位停止位；STOP=0，只有一位停止位。

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

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

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

UCR2 寄存器

UCR2 是 UART 的另外一个控制寄存器，它的主要功能是控制发送器、接收器以及各种 UART 中断源的使能或除能。它也可用来控制波特率，使能接收唤醒和地址侦测。详细解释如下：

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

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

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

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

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

Bit 3 **WAKE**：RX 脚下降沿唤醒功能使能位
0：RX 脚下降沿唤醒功能除能
1：RX 脚下降沿唤醒功能使能
此位为接收唤醒功能的使能和除能位。若 WAKE=1 且在 IDLE0 或 SLEEP 模式下，RX 引脚的下降沿将唤醒单片机。若 WAKE=0 且在 IDLE 或 SLEEP 模式下，RX 引脚的任何边沿都不能唤醒单片机。

Bit 2 **RIE**：接收中断使能位
0：接收中断除能
1：接收中断使能
此位为接收中断使能或除能位。若 RIE=1，且 OERR 或 RXIF 置位时，UART 的中断请求标志置位；若 RIE=0，UART 中断请求标志不受 OERR 和 RXIF 影响。

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

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

波特率发生器

UART 自身具有一个波特率发生器，通过它可以设置数据传输速率。波特率是由一个独立的内部 8 位计数器产生，它由 BRG 寄存器和 UCR2 寄存器的 BRGH 位一起控制。BRGH 是决定波特率发生器处于高速模式还是低速模式，从而决定计算公式的选用。BRG 寄存器的值 N 可根据下表中的公式计算，N 的范围是 0 到 255。

UCR2 的 BRGH 位	0	1
波特率 (BR)	$\frac{f_{SYS}}{[64(N+1)]}$	$\frac{f_{SYS}}{[16(N+1)]}$

为得到相应的波特率，首先需要设置 BRGH 来选择相应的计算公式从而算出 BRG 的值。由于 BRG 的值不连续，所以实际波特率和理论值之间有一个偏差。下面举例怎样计算 BRG 寄存器中的值 N 和误差。

BRG 寄存器

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

“x”未知

Bit 7~0 **BRG7~BRG0**：波特率值
软件设置 UCR2 寄存器的 BRGH 位（设置波特率发生器的速度）和 BRG 寄存器（设置波特率的值），一起控制 UART 的波特率。

波特率和误差的计算

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

根据上表，波特率 BR= $\frac{f_{SYS}}{[64(N+1)]}$

转换后的公式 N= $\frac{f_{SYS}}{(BR\times 64)}-1$

带入参数 N= $\frac{4000000}{(4800\times 64)}-1=12.0208$

取最接近的值，十进制 12 写入 BRG 寄存器，实际波特率如下

BR= $\frac{4000000}{[64(12+1)]}=4808$

因此，误差 = $\frac{4808-4800}{4800}$ =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，无校验，1 位停止位组成，用 8、N、1 表示，它是系统上电的默认格式。数据位数、停止位数和奇偶校验由 UCR1 寄存器的 BNO、PRT、PREN 和 STOPS 设置。用于数据发送和接收的波特率由一个内部的 8 位波特率发送器产生，数据传输时低位在前高位在后。尽管 UART 发送器和接收器在功能上相互独立，但它们使用相同的数据传输格式和波特率，在任何情况下，停止位是必须的。

UART 的使能和除能

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

UARTEN 清零将除能 TX 和 RX，若还未选择其它的引脚功能，这两个引脚处于浮空状态。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，使能控制、错误和状态标志位被复位，TXEN、RXEN、TXBRK、RXIF、OERR、FERR、PERR 和 NF 清零而 TIDLE、TXIF 和 RIDLE 置位，UCR1、UCR2 和 BRG 寄存器中的其它位保持不变。若 UART 工作时，UARTEN 清零，所有发送和接收将停止，模块也将复位成上述状态。当 UART 再次使能时，它将在上次配置下重新工作。

数据位、停止位以及奇偶校验的选择

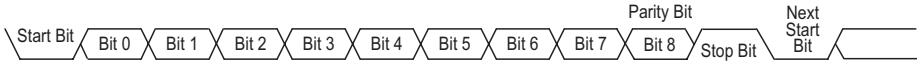
数据传输格式由数据长度、是否校验、校验类型、地址位以及停止位长度组成。它们都是由 UCR1 寄存器的各个位控制的。BNO 决定数据传输是 8 位还是 9 位；PRT 决定校验类型；PRTEN 决定是否选择奇偶校验；而 STOPS 决定选用 1 位还是 2 位停止位。下表列出了各种数据传输格式。地址位用来确定此帧是否为地址。停止位的长度和数据位的长度无关。

起始位	数据位	地址位	校验位	停止位
8 位数据格式				
1	8	0	0	1
1	7	0	1	1
1	7	1	0	1
9 位数据格式				
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。

8-Bit Data Format



9-Bit Data Format

UART 发送器

通过设置 UCR1 寄存器的 BNO 位，可以选择 8 位或 9 位数据字长度。BNO=1 其长度为 9 位，第 9 位 MSB 存储在 UCR1 寄存器的 TX8 中。发送器的核心是发送移位寄存器 TSR，它的数据由发送寄存器 TXR 提供，应用程序只须将发送数据写入 TXR 寄存器。上组数据的停止位发出前，TSR 寄存器禁止写入。如果还有新的数据要发送，一旦停止位发出，待发数据将会从 TXR 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。TXEN=1，发送使能，但若 TXR 寄存器没有数据或者波特率没有设置，发送器将不会工作。先写 TXR 寄存器再置高 TXEN 也会触发发送。当发送器使能，若 TSR 寄存器为空，数据写入 TXR 寄存器将会直接加载到 TSR 寄存器中。发送器工作时，TXEN 清零，发送器将立刻停止工作并且复位，此时 TX 引脚作为 I/O 口或其他引脚共用功能。

发送数据

当 UART 发送数据时，数据从移位寄存器中移到 TX 引脚上，其低位在前高位在后。在发送模式中，TXR 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 存储在 UCR1 寄存器的 TX8 中。

发送器初始化可由如下步骤完成：

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

如果要发送多个数据只需重复上述步骤。当 TXIF=0 时，数据将禁止写入 TXR 寄存器。可以通过以下步骤来清除 TXIF：

1. 读取 USR 寄存器
2. 写 TXR 寄存器

只读标志位 TXIF 由 UART 硬件置位。若 TXIF=1，TXR 寄存器为空，其它数据可以写入而不会覆盖以前的数据。若 TEIE=1，TXIF 标志位置位时中断产生。在数据传输时，写 TXR 指令会将待发数据暂存在 TXR 寄存器中，当前数据发送完毕后，待发数据被加载到发送移位寄存器中。当发送器空闲时，写 TXR 指令会将数据直接加载到 TSR 寄存器中，数据传输立刻开始且 TXIF 置位。当一帧数据发送完毕，TIDLE 将被置位。

可以通过以下步骤来清除 TIDLE：

1. 读取 USR 寄存器
2. 写 TXR 寄存器

清除 TXIF 和 TIDLE 软件执行次序相同。

发送暂停字

若 TXBRK=1，下一帧将会发送暂停字。它是由一个起始位、13×N（N=1，2……）位逻辑 0 组成。置位 TXBRK 将会发送暂停字，而清除 TXBRK 将产生停止位，传输暂停字不会产生中断。需要注意思的是，暂停字至少 13 位宽。若 TXBRK 持续为高，那么发送器会一直发送暂停字；当应用程序清除了 TXBRK，发送器将传输最后一帧暂停字再加上一位或者两位停止位。暂停字后的高电平保证下一帧数据起始位的检测。

UART 接收器

UART 接收器支持 8 位或者 9 位数据接收。若 BNO=1，数据长度为 9 位，而最高位 MSB 存放在 UCR1 寄存器的 RX8 中。接收器的核心是串行移位寄存器 RSR。RX 引脚上的数据送入数据恢复器中，它在 16 倍波特率的频率下工作，而串行移位器工作在正常波特率下。当在 RX 引脚上检测到停止位，数据从 RSR 寄存器中加载到 RXR 寄存器。RX 引脚上的每一位数据会被采样三次以判断其逻辑状态。RSR 不像其它寄存器一样映射在数据存储器，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 RX 引脚进入。RXR 寄存器在内部总线和接收移位寄存器间形成一个缓冲。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 RXR 寄存器，否则忽略第三帧数据并且发生溢出错误。

接收器的初始化可由如下步骤完成：

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

此时接收器被使能并检测起始位。

接收数据将会发生如下事件：

- 当 RXR 寄存器中有一帧以上的数据时，USR 寄存器中的 RXIF 位将会置位。
- 若 RIE=1，数据从 RSR 寄存器加载到 RXR 寄存器中将产生中断。
- 若接收器检测到帧错误、噪声干扰错误、奇偶出错或溢出错误，那么相应的错误标志位置位。

可以通过如下步骤来清除 RXIF：

1. 读取 USR 寄存器
2. 读取 RXR 寄存器

接收暂停字

UART 接收任何暂停字都会当作帧错误处理。接收器只根据 BNO 和 STOPS 位确定一帧数据的长度。若暂停字位数大于 BNO 和 STOPS 位指定的长度，接收器认为接收已完毕，RXIF 和 FERR 置位，RXR 寄存器清 0，若相应的中断允许且 RIDLE 为高将会产生中断。若暂停字较长，接收器收到起始位、数据位将会置位 FERR 标志，且在下一起始位前必须检测到有效的停止位。暂停字只会被认为包含信息 0 且会置位 FERR 标志。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 RIDLE。

UART 接收到暂停字会产生以下事件：

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

空闲状态

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

接收中断

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

接收错误处理

UART 会产生几种接收错误，下面部分将描述各错误以及怎样处理。

溢出——OERR 标志

RXR 寄存器是一个两个字节深度的 FIFO 数据缓冲器，它能保存两个字节数据的同时接收第三个字节数据，应用程序必须保证在接收完第三个字节数据前读取 RXR 寄存器，否则发生溢出错误。

产生溢出错误时将会发生以下事件：

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

先读取 USR 寄存器再读取 RXR 寄存器可将 OERR 清零。

噪声干扰——NF 标志

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

- 在 RXIF 上升沿，USR 寄存器中只读标志位 NF 置位。
- 数据从 RSR 寄存器加载到 RXR 寄存器中。
- 不产生中断，此位置位的同时由 RXIF 请求中断。

先读取 USR 寄存器再读取 RXR 寄存器可将 NF 清零。

帧错误——FERR 标志

若在停止位上检测到 0，USR 寄存器中只读标志 FERR 置位。若选择两位停止位，只侦测第一个停止位且此位必须为高，否则将置位 FERR。它同数据一起存储在缓冲器中，可被任何复位清零。

奇偶校验错误——PERR 标志

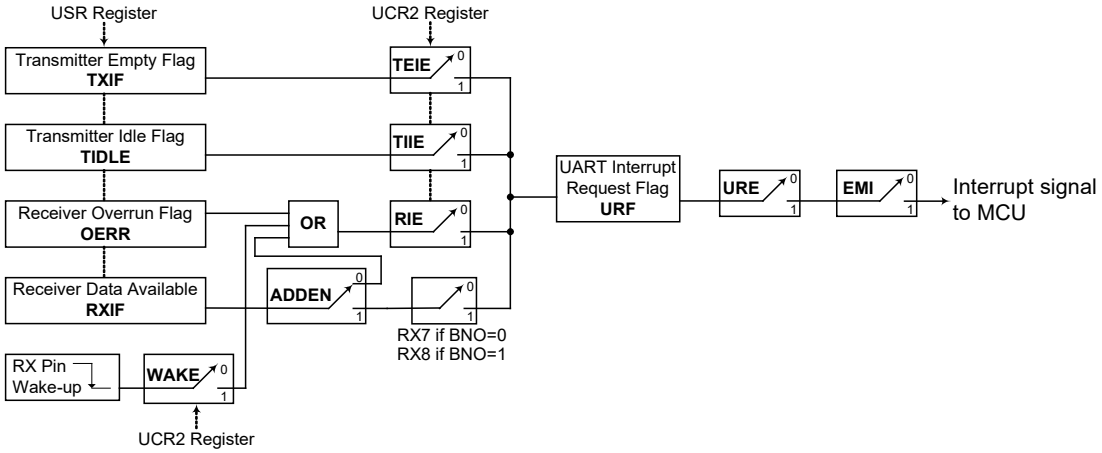
若接收到数据出现奇偶校验错误，USR 寄存器中只读标志 PERR 置位。只有使能了奇偶校验，选择了校验类型，此标志位才有效。它同数据一起存储在缓冲器中，可被任何复位清除。注意，FERR 和 PERR 与相应的数据一起存储于缓冲器中，在读取数据之前必须先访问错误标志位。

UART 模块中断结构

几个独立的 UART 条件可产生一个 UART 中断。当存在这些条件，一个低脉冲产生，进而提醒单片机注意。这些条件包括：发送寄存器为空、发送器空闲、接收器数据有效、接收器溢出、地址检测和 RX 引脚唤醒。当其中任何一种情况发生时，若其对应的中断控制位使能、 整体 UART 中断允许且堆栈未满，程序将会跳转到相应的中断向量执行中断服务程序，而后再返回主程序。这其中的四种情况在 USR 寄存器中有相关的标志位，若 UCR2 寄存器中相应中断允许位被置位，USR 寄存器中标志位将会产生中断。发送器有两个相应的中断允许位而接收器共用一个中断允许位。这些允许位可用于禁止个别的 UART 中断源。

地址检测也是 UART 的中断源，它没有相应的标志位，若 UCR2 寄存器中 ADDEN=1，当检测到地址将会产生 UART 中断。RX 引脚唤醒也可以产生 UART 中断，它没有相应的标志位，当 UCR2 中的 WAKE 和 RIE 位被置位，RX 引脚上有下降沿可以唤醒单片机。应注意，RX 唤醒中断发生时，系统必须延时 t_{SST} 才能正常工作。

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



UART 中断结构

地址检测模式

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

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

ADDEN 位功能

UART 模块暂停和唤醒

当 f_{SYS} 关闭时，UART 模块将暂停运行，接至 UART 模块的时钟源将除能。若正在传送数据时关闭 f_{SYS}，发送将停止直到 UART 模块时钟源再次使能。同样地，当接收数据时 UART 进入暂停模式，数据接收也会停止。当 UART 电路进入暂停模式，USR、UCR1、UCR2 、接收 / 发送寄存器以及 BRG 寄存器都不会受到影响。建议在 MCU 进入暂停模式前先确保数据发送或接收已完成。

UART 功能中包括了 RX 引脚的唤醒功能，由 UCR2 寄存器中 WAKE 位控制。进入 IDLE0 或 SLEEP 模式前，若该标志位 WAKE 与 UART 允许位 UARTEN、接收器允许位 RXEN 和接收器中断位 RIE 都被置位，则 RX 引脚的下降沿可将单片机从 IDLE0 或 SLEEP 模式唤醒。唤醒后系统需延时 t_{SST} 才能正常工作，在此期间，RX 引脚上的任何数据将被忽略。

若要唤醒并产生 UART 中断，除了唤醒使能控制位和接收中断使能控制位需置位外，全局中断允许位 EMI 和 UART 中断使能控制位 URE 也必须置位；若这两控制位没有被置位，那么，单片机将可以被唤醒但不会产生中断。同样唤醒后系统需一定的延时才能正常工作，然后才会产生 UART 中断。

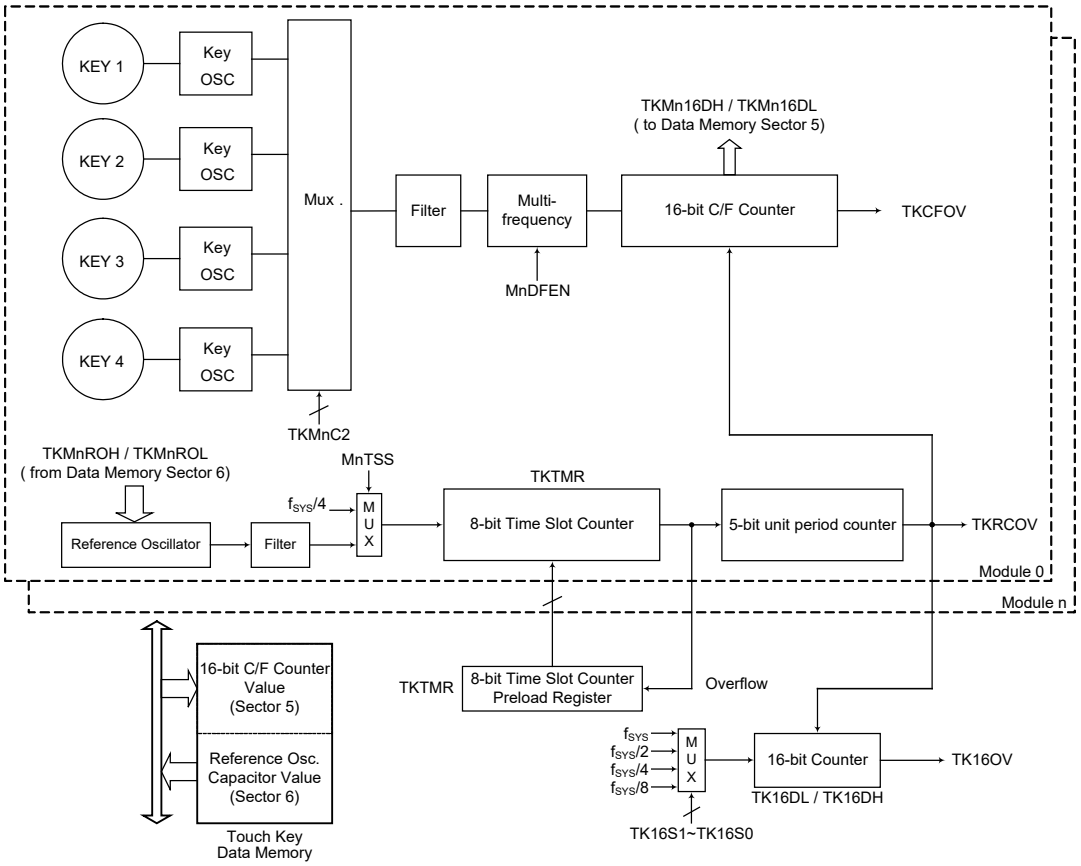
触控按键功能

该系列单片机提供多触控按键功能。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

触控按键引脚与 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 n 个部分，即 M0~Mn 模块。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系列寄存器。寄存器的名称和它相关的模块编号相对应。

单片机型号	按键总数	触控按键模块	触控按键
BS66F370	36	M0	KEY1~KEY4
		M1	KEY5~KEY8
		M2	KEY9~KEY12
		M3	KEY13~KEY16
		M4	KEY17~KEY20
		M5	KEY21~KEY24
		M6	KEY25~KEY28
		M7	KEY29~KEY32
		M8	KEY33~KEY36
BS66F360	28	M0	KEY1~KEY4
		M1	KEY5~ KEY8
		M2	KEY9~ KEY12
		M3	KEY13~KEY16
		M4	KEY17~ KEY20
		M5	KEY21~ KEY24
		M6	KEY25~KEY28
BS66F350	20	M0	KEY1~KEY4
		M1	KEY5~ KEY8
		M2	KEY9~ KEY12
		M3	KEY13~KEY16
		M4	KEY17~ KEY20
BS66F340	12	M0	KEY1~KEY4
		M1	KEY5~ KEY8
		M2	KEY9~ KEY12



注：虚线部分是每个触控按键模块都单独有的部分

触控按键功能方框图

触控按键寄存器

每个触控按键模块包含四个触控按键功能，且都有配套的寄存器。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 n 对应触控按键模块的序号。该系列单片机高达七个触控按键模块，均取决于所选单片机。BS66F340 触控按键模块序号为 n=0~2，BS66F350 触控按键模块序号为 n=0~4，BS66F360 触控按键模块序号为 n=0~6，BS66F370 触控按键模块序号为 n=0~8。

名称	作用
TKTMR	触控按键功能 8 位时隙定时计数器预置寄存器
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
TK16DL	触控按键功能 16 位计数器低字节内容
TK16DH	触控按键功能 16 位计数器高字节内容
TKMn16DL	触控按键模块 n 16 位 C/F 计数器低字节内容
TKMn16DH	触控按键模块 n 16 位 C/F 计数器高字节内容
TKMnROL	触控按键模块 n 参考振荡器电容值低字节内容
TKMnROH	触控按键模块 n 参考振荡器电容值高字节内容
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

触控按键寄存器列表

寄存器名称	Bit							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TKTMR	D7	D6	D5	D4	D3	D2	D1	D0
TKC0	TKRAMC	TKRCOV	TKST	TKCFOV	TK16OV	—	TKMOD	TKBUSY
TKC1	D7	D6	D5	TSCS	TK16S1	TK16S0	TKFS1	TKFS0
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	D4	MnSOFC	MnSOF2	MnSOF1	MnSOF0
TKMnC1	MnTSS	—	MnROEN	MnKOEN	MnK4EN	MnK3EN	MnK2EN	MnK1EN
TKMnC2	MnSK31	MnSK30	MnSK21	MnSK20	MnSK11	MnSK10	MnSK01	MnSK00

触控按键寄存器列表

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 触控按键功能 8 位时隙定时计数器预置寄存器

此寄存器用于控制触控按键时隙溢出时间。时隙周期由一个 5 位时隙单位周期计数器控制，等于 32 个 8 位时隙定时计数器周期。时隙计数器溢出时间可通过下述公式计算：

时隙计数器溢出时间＝(256-TKTMR[7:0]) × $32t_{TSC}$，其中 t_{TSC} 为时隙计数器时钟。

TKC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	TKRAMC	TKRCOV	TKST	TKCFOV	TK16OV	—	TKMOD	TKBUSY
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 **TKRAMC**：触控按键 RAM 存取控制位

0：由 MCU 读 / 写

1：由触控按键模块读 / 写

该位用于控制触控按键模块 RAM 是通过 MCU 还是触控按键模块存取。但若触控按键模块工作在自动扫描模式（当 TKMOD 位为低时，TKST 位由 0 转为 1 可进入自动扫描模式），由触控按键模块可优先存取 RAM 内容。当结束自动扫描后（如 TKBUSY 位状态由 1 转为 0），触控按键模块 RAM 存取才可通过 TKRAMC 位控制。因此建议当触控按键模块工作在自动扫描模式时，将 TKRAMC 位置为“1”。否则，若选择 MCU 存取 RAM，在自动扫描阶段，RAM 内容可能会被改动。

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，相应的中断请求标志位不会受到影响。

在自动扫描模式时，如果时隙计数器溢出，但自动扫描还未完成，TKRCOV 位将不会被置位，同时触控按键模块 n16 位 C/F 计数器、触控按键功能 16 位计数器、5 位时隙单位周期计数器将会自动清零，但 8 位时隙定时计数器将会从 8 位时隙定时计数器预置寄存器（TKTMR 寄存器）重新加载数据。当自动扫描工作结束，TKRCOV 位及触控按键中断请求标志位 TKMF 将被置位同时所有模块按键振荡器和参考振荡器自动停止。所有模块的 16 位 C/F 计数器、触控按键功能 16 位计数器、5 位时隙单计数器和 8 位时隙定时计数器都会自动关闭。

在手动扫描模式时，如果时隙计数器溢出，TKRCOV 位及触控按键中断请求标志位 TKMF 将被置位同时所有模块按键振荡器和参考振荡器自动停止。所有模块的 16 位 C/F 计数器、触控按键功能 16 位计数器、5 位时隙单位周期计数器和 8 位时隙定时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：检测停止或无操作

0 → 1：启动检测

当该位为“0”时，所有模块的 16 位 C/F 计数器、触控按键功能 16 位计数器和 5 位时隙单位周期计数器会自动清零但 8 位可编程时隙定时计数器不会被清零。当该位由 0 → 1 时，16 位 C/F 计数器、触控按键功能 16 位计数器、5 位时隙单位周期计数器和 8 位时隙定时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 位 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 位 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 位计数器溢出标志位
0：无溢出
1：溢出
该位由触控按键功能 16 位计数器溢出置位，必须通过应用程序清零。

Bit 2 未定义，读为“0”

Bit 1 **TKMOD**：触控按键扫描模式选择位
0：自动扫描模式
1：手动扫描模式
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 位 C/F 计数器值。
在自动扫描模式时，上述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 位 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] 位设置。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。

Bit 0 **TKBUSY**：触控按键扫描忙碌标志位
0：不忙碌－没有执行按键扫描或按键扫描已完成
1：忙碌－正在扫描中
该位作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”，当触控按键时隙计数器溢出，该位被清零。

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
这些位供内部测试使用，正常工作时，要保持 D7~D5 为“000”

Bit 4 **TSCS**：触控按键时隙计数器使用选择位
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 位计数器时钟选择位
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
00：500kHz
01：1000 kHz
10：1500 kHz
11：2000 kHz

TK16DH/TK16KDL – 触控按键功能 16 位计数器寄存器对

寄存器	TK16DH								TK16DL							
Bit	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

该寄存器对用于放置触控按键功能 16 位计数器值。此 16 位计数器用于校准触控按键振荡器和参考振荡器频率。在手动模式扫描时，如果触控按键时隙计数器溢出，16 位计数器停止但不会改变当前值。在自动扫描模式时，时隙 0，时隙 1 和时隙 2 扫描结束时，16 位计数器值会被清零，但在时隙 3 结束时，16 位计数器值不会改变。当 TKST 位为“0”时，该寄存器对被清零。

TKMn16DH/TKMn16KDL – 触控按键模块 n 16 位 C/F 计数器寄存器对

寄存器	TKMn16DH								TKMn16DL							
Bit	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

该寄存器对用于放置触控按键模块 n 16 位 C/F 计数器值。在手动模式扫描时，如果触控按键时隙计数器溢出，此 16 位 C/F 计数器停止但不会改变当前值。在自动扫描模式时，时隙 0，时隙 1 和时隙 2 扫描结束时，此 16 位 C/F 计数器值会被清零，但在时隙 3 结束时，16 位 C/F 计数器值不会改变。当 TKST 位为“0”时，该寄存器对被清零。

TKMnROH/TKMnROL – 触控按键模块 n 参考振荡器电容值寄存器对

寄存器	TKMnROH								TKMnROL							
Bit	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

该寄存器对用于放置触控按键模块 n 参考振荡器电容值。在自动扫描模式时，当前时隙扫描结束后，此寄存器会从专用的触控按键模块数据存储区加载下一时隙要扫描按键对应的参考振荡器电容值。

$$\text{参考振荡器内接电容值} = \frac{\text{TKMn}[9:0]\text{x}50\text{pF}}{1024}$$

TKMnC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	MnDFEN	D4	MnSOFC	MnSOF2	MnSOF1	MnSOF0
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　**MnDFEN**：触控按键模块 n 倍频功能控制位
0：除能
1：使能
该位用于控制触控按键振荡器频率加倍功能，当此位设置为“1”时，按键振荡器频率加倍。

Bit 4　**D4**：测试数据位
该位供内部测试使用，正常工作时，要保持 D4 为“0”

Bit 3　**MnSOFC**：触控按键模块 n C/F 振荡器跳频功能控制选择位
0：由 MnSOF2~MnSOF0 位控制
1：由硬件电路控制
该位用于选择 C/F 振荡器跳频功能控制方式。当此位置 1 时，按键振荡器跳频功能由硬件电路控制，MnSOF2~MnSOF0 位的设置无效。

Bit 2~0　**MnSOF2~MnSOF0**：触控按键模块 n 按键振荡器和参考振荡器跳频选择位 (MnSOFC=0)
000：f_{HOP0} – 最小跳频
001：f_{HOP1}
010：f_{HOP2}
011：f_{HOP3}
100：f_{HOP4} – 触控按键振荡器频率
101：f_{HOP5}
110：f_{HOP6}
111：f_{HOP7} – 最大跳频

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：来自触控按键模块 n 参考振荡器
1：$f_{SYS}/4$

Bit 6　未定义，读为“0”

Bit 5　**MnROEN**：触控按键模块 n 参考振荡器控制位
0：除能
1：使能
该位用于使能 / 除能触控按键模块 n 参考振荡器。在自动扫描模式，若选择参考振荡器作为时隙计数器时钟源，当 TKST 位置高后，设置 MnROEN 位 1 即可自动使能参考振荡器。MnTSS、TSCS 及 MnK4EN~MnK1EN 位共同决定参考振荡器是否被使用。若 TKBUSY 位发生由高到低的转变，MnROEN 位将自动变为“0”以除能参考振荡器。

Bit 4 **MnKOEN**：触控按键模块 n 按键振荡器控制位
0：除能
1：使能
该位用于使能 / 除能触控按键模块 n 按键振荡器。在自动扫描模式，当 TKST 位由低到高，则通过置位 MnKOEN 位即可自动使能参考振荡器。若 TKBUSY 位发生由高到低的转变，MnKOEN 位将自动变为“0”以除能按键振荡器。
在手动扫描模式，若相关按键功能已被使能扫描，在设置 TKST 由低到高前，应先使能按键振荡器。当 TKBUSY 由高变为低时，按键振荡器除能。

Bit 3 **MnK4EN**：触控按键模块 n key4 功能控制位
0：除能
1：使能

Bit 2 **MnK3EN**：触控按键模块 n key3 功能控制位
0：除能
1：使能

Bit 1 **MnK2EN**：触控按键模块 n key2 功能控制位
0：除能
1：使能

Bit 0 **MnK1EN**：触控按键模块 n key1 功能控制位
0：除能
1：使能

TKMnC2 寄存器

Bit	7	6	5	4	3	2	1	0
Name	MnSK31	MnSK30	MnSK21	MnSK10	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
该位用于选择在时隙 3 时要被扫描的按键，只有在自动扫描模式时用效。

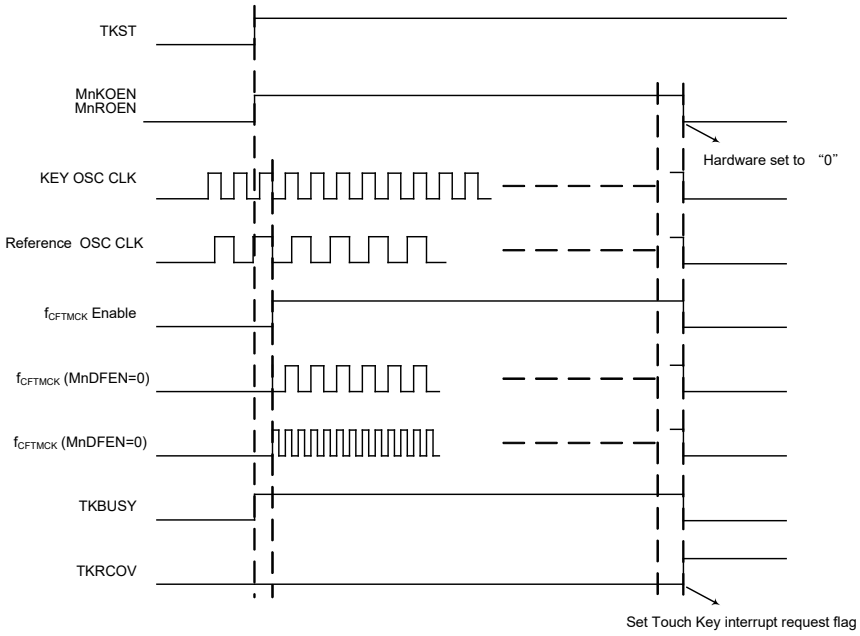
Bit 5~4 **MnSK21~MnSK20**：触控按键模块 n 时隙 2 扫描按键选择位
00：Key 1
01：Key 2
10：Key 3
11：Key 4
该位用于选择在时隙 2 时要被扫描的按键，只有在自动扫描模式时用效。

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
该位用于选择在时隙 1 时要被扫描的按键，只有在自动扫描模式时用效。

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
该位用于选择在自动扫描模式时时隙 0 时要被扫描的按键或用作在手动模式时选择要扫描按键。

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。通过寄存器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。在最后一个时间间隔后，会产生一个触控按键中断信号。

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，在 TKC0 寄存器中的 TKST 被清零时，所有模块的 16 位 C/F 计数器、16 位计数器和 5 位时隙单位周期计数器会自动清零，而 8 位可编程时隙计数器不清零，由用户设置溢出时间。TKST 位置高时，16 位 C/F 计数器、16 位计数器、5 位时隙单位周期计数器和 8 位时隙定时计数器会自动开启。

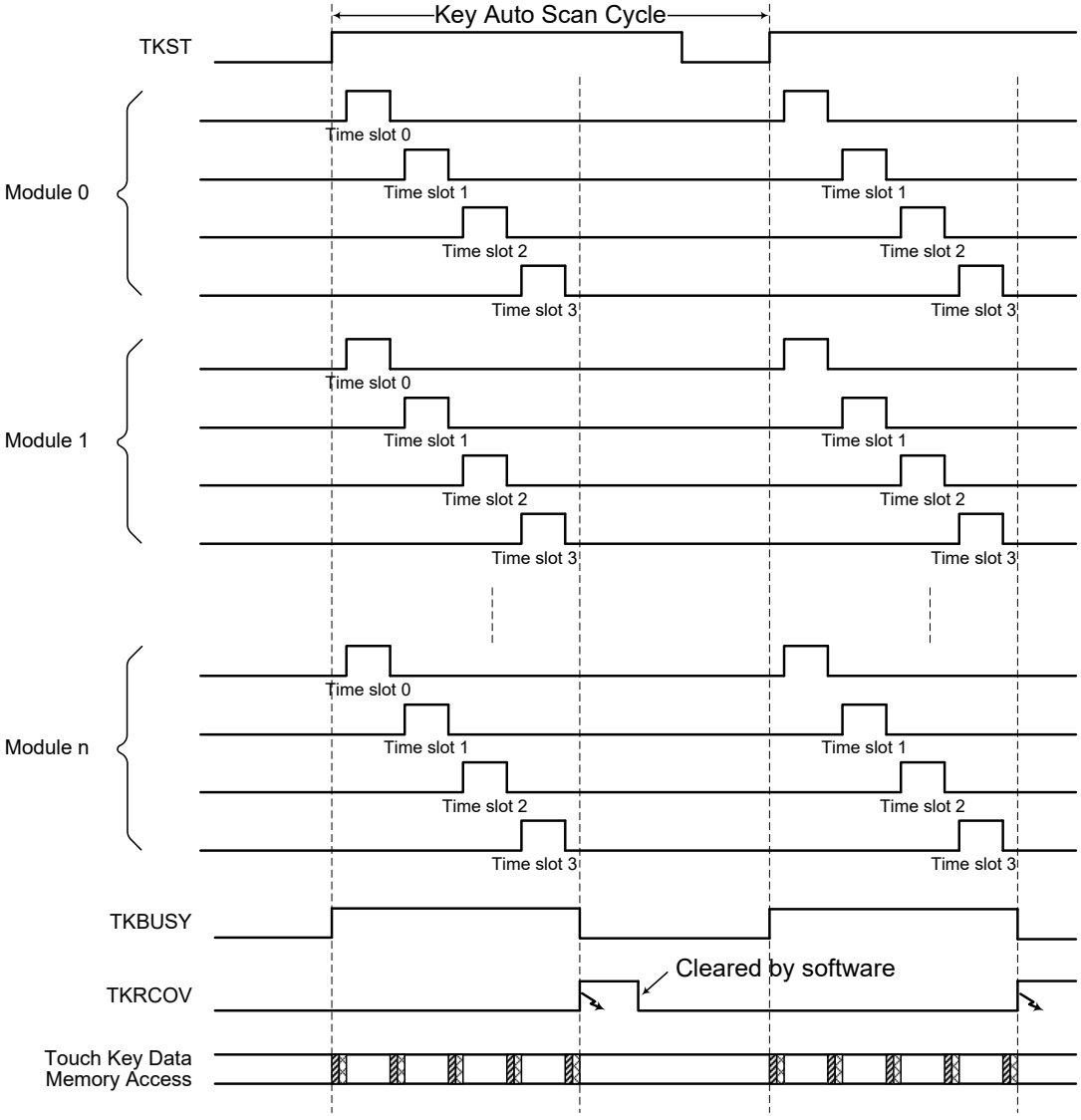
时隙计数器溢出，所有模块的按键振荡器和参考振荡器都会自动停止且 16 位 C/F 计数器、16 位计数器、5 位时隙单位周期计数器和 8 位时隙定时计数器会自动停止。时隙计数器时钟源可通过 TKMnC1 寄存器 MnTSS 位选择来自参考振荡器或 $f_{SYS}/4$。通过设置 TKMnC1 寄存器中的 MnROEN 位和 MnKOEN 为“1”，可使能参考振荡器和按键振荡器。

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，产生触控按键中断。这里所有的触控按键是指已使能的触控按键。

每 4 个按键为一个模块，所以 Key 1 ~ Key 4 为模块 0，Key 5 ~ Key 8 为模块 1，Key 9 ~ Key 12 为模块 2，以此类推。每个模块都是相同的架构。

自动扫描模式

触控按键功能包含两种按键扫描模式，即自动扫描模式和手动扫描模式。自动扫描模式可以较大程度地减少程序负担并能提高按键扫描执行效率。通过 TKC0 寄存器 TKMOD 位选择自动或手动扫描模式。设置 TKMOD 位为 0 可选择自动扫描模式，在此模式下以指定顺序自动扫描每个模块的按键。扫描顺序及按键由 TKMnC2 寄存器的 MnSK3[1:0]~ MnSK0[1:0] 决定。



↘ : 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

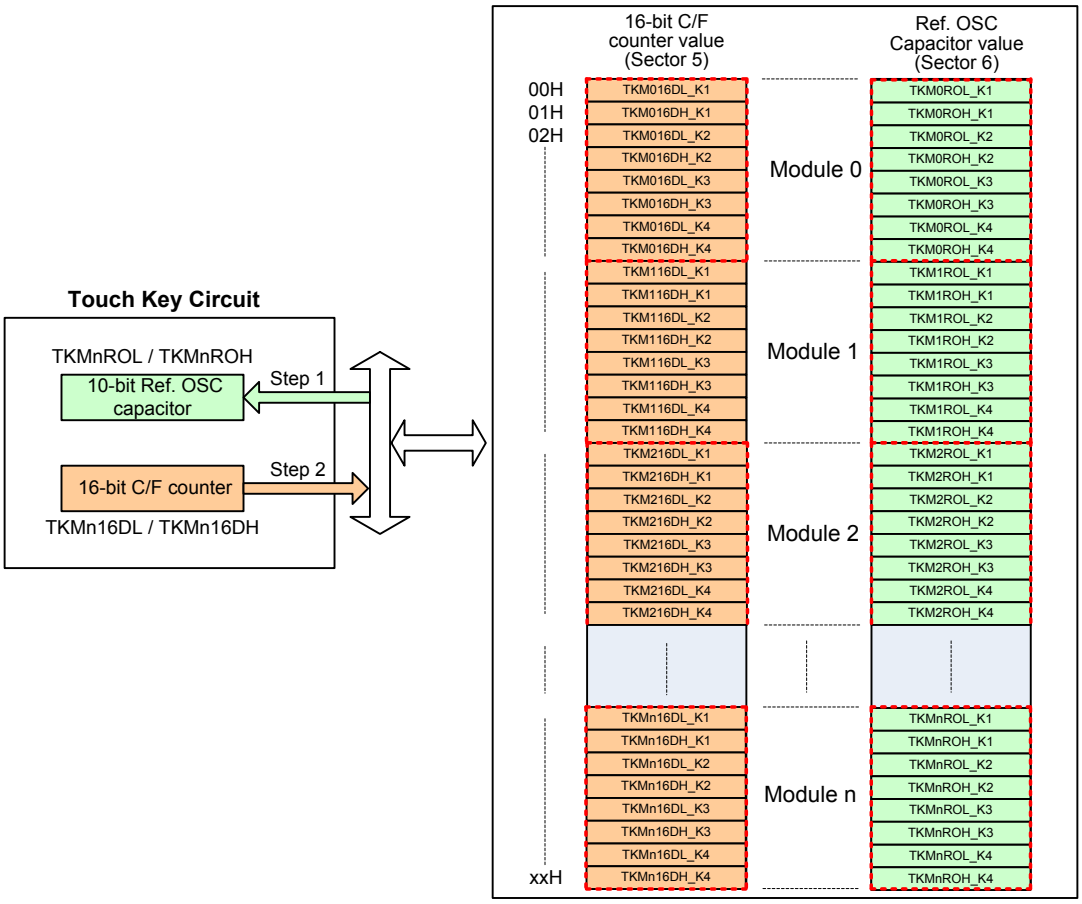
N = Touch Key Module Number; n = Module Serial Number

触控按键自动扫描时序图

当选择了自动扫描模式，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。在自动扫描模式时且已设置 TKST 位由低到高，硬件会自动从专用触控按键数据存储区指定位置读取第一个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。并将 TKMn16DH / TKMn16DL 的内容写回到专用触控按键数据存储区最后一个扫描的按键对应的位置。之后开始扫描选择的时隙 0 的按键。时隙 0 按键扫描结束后，硬件会自动从专用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 TKMn16DH / TKMn16DL 的内容写回到专用触控按键数据存储区对应位置，……，以此类推，完成时隙 0 至时隙 3 所有按键自动扫描过程。完成后，TKRCOV 位将被置高同时 TKBUSY 位拉低。自动扫描结束后，硬件会再次从专用触控按键数据存储区读取第一个扫描按键的参考振荡器电容值并写入相应的 TKMnROH/TKMnROL 寄存器对中。16 位 C/F 计数器值 TKMn16DH / TKMn16DL 也会被再次写回到专用触控按键数据存储区。

触控按键模块数据存储区

该系列单片机为触控按键模块提供 2 个专用的数据存储器区域。一个位于 Sector 5 用于存储触控按键模块 0~n 16 位 C/F 计数器值，另一个位于 Sector 6 用于存储触控按键模块 0~n 参考振荡器内部电容值。

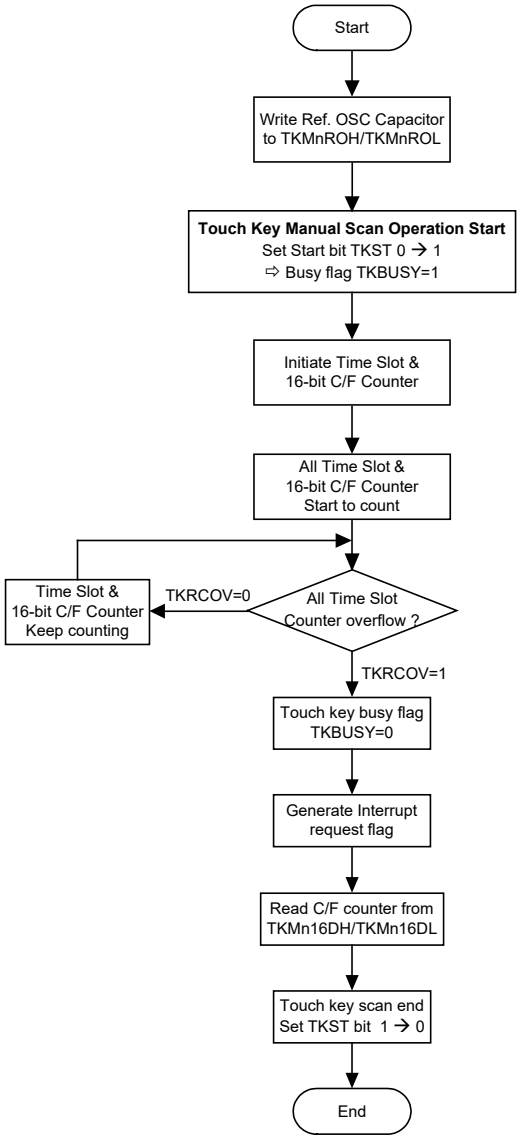


触控按键模块数据存储器分布

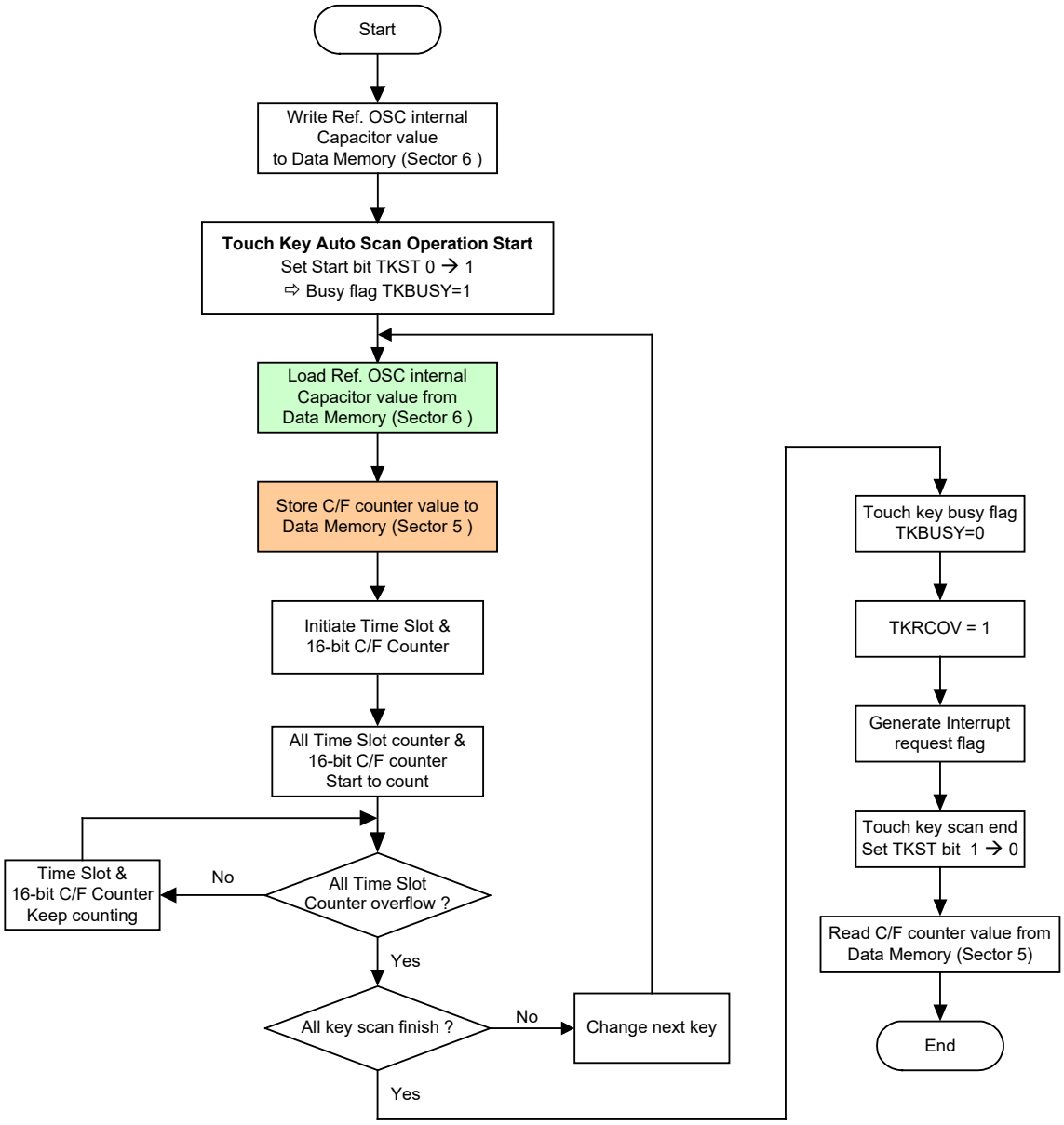
单片机型号	触控按键数据存储器 **Sector: 地址**
BS66F340	5: 00H~1FH 6: 00H~1FH
BS66F350	5: 00H~27H 6: 00H~27H
BS66F360	5: 00H~37H 6: 00H~37H
BS66F370	5: 00H~47H 6: 00H~47H

触控按键数据存储器概要

触控按键扫描流程图



手动扫描模式流程图－ **TKMOD=1, TSCS=0**



自动扫描模式流程图－ **TKMOD=0, TSCS=0**

触控按键中断

所有触控按键只有一个中断，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，才产生中断，此时所有模块的 16 位 C/F 计数器、16 位计数器，5 位时隙单位周期计数器和 8 位时隙定时计数器会自动清零。这里提到的触控按键都指已使能的触控按键。

编程注意事项

相关寄存器设置后，TKST 位由低电平变为高电平，触控按键检测程序初始化。此时所有相关的振荡器将使能并同步。当计数器溢出时，时隙计数器标志位 TKRCOV 将变为高电平。计数器溢出发生时，会产生一个中断信号。

当外部触控按键的大小和布局确定时，其相关的电容将决定感应振荡器的频率。

低电压检测 – LVD

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

LVD 寄存器

低电压检测功能由 LVDC 寄存器控制。VLVD2~VLVD0 位用于选择 8 个固定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。LVDEN 位用于控制低电压检测功能的开启 / 关闭，设置此位为高使能此功能，反之，关闭内部低电压检测电路。低电压检测会有一定的功耗，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	LVDEN	VBGEN	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

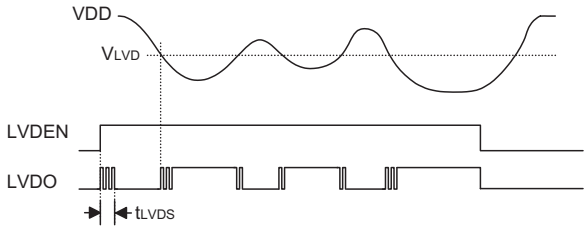
Bit 4　**LVDEN**：低电压检测控制位
0：除能
1：使能

Bit 3　**VBGEN**：Bandgap 电压输出控制位
0：除能
1：使能

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

LVD 操作

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~4.0V。当电源电压 V_{DD} 低于预置电压值时，LVDO 位被置为高，表明低电压产生。低电压检测功能由一个自动使能的参考电压提供。若 LVDEN 位为高，当单片机掉电时低电压检测器保持有效状态。低电压检测器使能后，读取 LVDO 位前，电路稳定需要一定的延时 t_{LVDS}。注意，V_{DD} 电压可能上升或下降比较缓慢，在 V_{LVD} 电压值附近时，LVDO 位可能有多种变化。



LVD 操作

低电压检测器也有自己的中断功能，也是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。中断条件产生置位 LVDO 并延时 t_{LVD} 后，中断产生。若 LVDEN 位为高，当单片机掉电时低电压检测器保持有效状态。此种情况下，若 V_{DD} 降至小于 LVD 预置电压值时，中断请求标志位 LVF 将被置位，中断产生，单片机将从休眠或空闲模式中被唤醒。若不要求低电压检测的唤醒功能使能，在单片机进入休眠或空闲模式前应将 LVF 标志置为高。

中断

中断是单片机一个重要功能。当外部事件或内部功能如定时器模块或 A/D 转换器有效，并且产生中断时，系统会暂时中止当前的程序而转到执行相对应的中断服务程序。此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、触控按键、LVD、EEPROM、SIM、UART 和 A/D 转换器等产生。

中断寄存器

中断控制基本上是在一定单片机条件发生时设置请求标志位，应用程序中中断使能位的设置是通过位于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~MFI3 寄存器，用于设置多功能中断；最后一种有 INTEG 寄存器，用于设置外部中断边沿触发类型。

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

功能	使能位	请求标志	注释
总中断	EMI	—	—
INTn 脚	INTnE	INTnF	n=0~1
触控按键模块	TKME	TKMF	—
UART	URE	URF	—
多功能	MFnE	MFnF	n=0~3
A/D 转换器	ADE	ADF	—
时基	TBnE	TBnF	n=0~1
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
SIM	SIME	SIMF	—
CTM	CTMnPE	CTMnPF	n=0~1
	CTMnAE	CTMnAF	
PTM	PTMPE	PTMPF	—
	PTMAE	PTMAF	
STM	STMPE	STMPF	—
	STMAE	STMAF	

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	—	TKMF	INT1F	INT0F	TKME	INT1E	INT0E	EMI
INTC1	ADF	MF1F	MF0F	URF	ADE	MF1E	MF0E	URE
INTC2	MF3F	TB1F	TB0F	MF2F	MF3E	TB1E	TB0E	MF2E
MFI0	—	—	CTM0AF	CTM0PF	—	—	CTM0AE	CTM0PE
MFI1	STMAF	STMPF	CTM1AF	CTM1PF	STMAE	STMPE	CTM1AE	CTM1PE
MFI2	—	SIMF	PTMAF	PTMPF	—	SIME	PTMAE	PTMPE
MFI3	—	—	DEF	LVF	—	—	DEE	LVE

中断寄存器列表

INTEG 寄存器

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

Bit 7~4　未定义，读为“0”

Bit 3~2　**INT1S1~INT1S0**：INT1 脚中断边沿控制位
00：除能
01：上升沿
10：下降沿
11：双沿

Bit 1~0　**INT0S1~INT0S0**：INT0 脚中断边沿控制位
00：除能
01：上升沿
10：下降沿
11：双沿

INTC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	TKMF	INT1F	INT0F	TKME	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　**TKMF**：触控按键模块中断请求标志位
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

Bit 3 **TKME**：触控按键模块中断控制位
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

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	ADF	MF1F	MF0F	URF	ADE	MF1E	MF0E	URE
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 **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 **URF**：UART 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 **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 **URE**：UART 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	MF3F	TB1F	TB0F	MF2F	MF3E	TB1E	TB0E	MF2E
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 **MF3F**：多功能中断 3 请求标志位
0：无请求
1：中断请求

Bit 6 **TB1F**：时基 1 中断请求标志位
0：无请求
1：中断请求

Bit 5 **TB0F**：时基 0 中断请求标志位
0：无请求
1：中断请求

Bit 4 **MF2F**：多功能中断 2 请求标志位
0：无请求
1：中断请求

Bit 3 **MF3E**：多功能中断 3 控制位
0：除能
1：使能

Bit 2 **TB1E**：时基 1 中断控制位
0：除能
1：使能

Bit 1 **TB0E**：时基 0 中断控制位
0：除能
1：使能

Bit 0 **MF2E**：多功能中断 2 控制位
0：除能
1：使能

MFI0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	CTM0AF	CTM0PF	—	—	CTM0AE	CTM0PE
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 **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 3~2 未定义，读为“0”

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	STMAF	STMPF	CTM1AF	CTM1PF	STMAE	STMPE	CTM1AE	CTM1PE
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 **STMAF**：STM 比较器 A 匹配中断请求标志位
0：无请求
1：中断请求

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

Bit 5 **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 4 **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 3 **STMAE**：STM 比较器 A 匹配中断控制位
0：除能
1：使能

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

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

Bit 0 **CTM1PE**：CTM1 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	—	SIMF	PTMAF	PTMPF	—	SIME	PTMAE	PTMPE
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 **SIMF**：SIM 请求标志位
0：无请求
1：中断请求

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

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

Bit 3 未定义，读为“0”

Bit 2 **SIME**：SIM 中断控制位
0：除能
1：使能

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

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

MFI3 寄存器

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

Bit 7~6 未定义，读为“0”

Bit 5 **DEF**：数据 EEPROM 中断请求标志位
0：无请求
1：中断请求

Bit 4 **LVF**：LVD 中断请求标志位
0：无请求
1：中断请求

Bit 3~2 未定义，读为“0”

Bit 1 **DEE**：数据 EEPROM 中断控制位
0：除能
1：使能

Bit 0 **LVE**：LVD 中断控制位
0：除能
1：使能

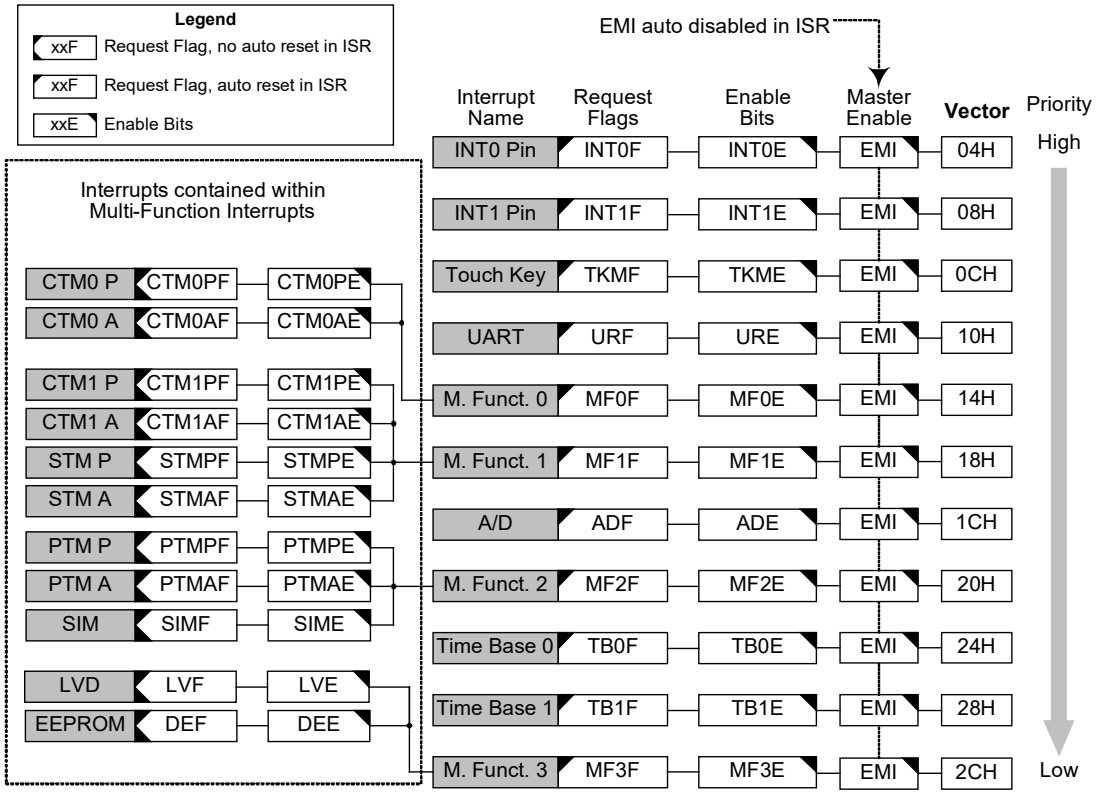
中断操作

若中断事件条件产生，如一个 TM 比较器 P、比较器 A 匹配或 A/D 转换结束等等，相关中断请求标志将置起。中断标志产生后程序是否会跳转至相关中断向量执行是由中断使能位的条件决定的。若使能位为“1”，程序将跳至相关中断向量中执行；若使能位为“0”，即使中断请求标志置起中断也不会发生，程序也不会跳转至相关中断向量执行。若总中断使能位为“0”，所有中断都将除能。

当中断发生时，下条指令的地址将被压入堆栈。相应的中断向量地址加载至 PC 中。系统将从此向量取下条指令。中断向量处通常为“JMP”指令，以跳转到相应的中断服务程序。中断服务程序必须以“RETI”指令返回至主程序，以继续执行原来的程序。

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

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



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脚状态改变，将调用外部中断向量子程序。当响应外部中断服务子程序时，中断请求标志位 INT0F~INT1F 会自动复位且 EMI 位会被清零以除能其它中断。注意，即使此引脚被用作外部中断输入，其上拉电阻仍保持有效。

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

Touch Key 中断

触控按键中断由时隙计数器溢出来控制。当触控按键模块中断请求标志位 TKMF 被置位，即触控按键中的时隙计数器溢出，中断请求发生。若要跳转到相应中断向量地址，总中断使能位 EMI 和触控按键模块使能位 TKME 必须先被置位，中断使能，堆栈未满且触控按键时隙计数器溢出时，将调用位于触控按键中断向量处的子程序。当响应中断服务子程序时，中断请求标志位 TKMF 会被自动复位且 EMI 位会被清零以除能其它中断。

UART 传输中断

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

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。

多功能中断

此系列单片机中有多达 4 种多功能中断，与其它中断不同，它没有独立源，但由其它现有的中断源构成，即 TM 中断、LVD 中断、EEPROM 写中断和 SIM 中断。

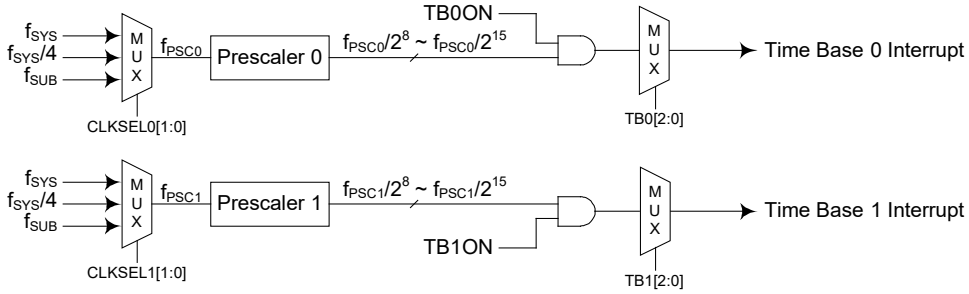
当多功能中断中任何一种中断请求标志 MFnF 被置位，多功能中断请求产生。当中断使能，堆栈未满，包括在多功能中断中的任意一个中断发生时，将调用多功能中断向量中的一个子程序。当响应中断服务子程序时，相关的多功能请求标志位会自动复位且 EMI 位会自动清零以除能其它中断。

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，即 TM 中断、LVD 中断、EEPROM 写中断和 SIM 中断的请求标志位不会自动复位，必须由应用程序清零。

时基中断

时基中断提供一个固定周期的中断信号，由各自的定时器功能产生溢出信号控制。当各自的中断请求标志 TBnF 被置位时，中断请求发生。当总中断使能位 EMI 和时基使能位 TBnE 被置位，允许程序跳转到各自的中断向量地址。当中断使能，堆栈未满且时基溢出时，将调用它们各自的中断向量子程序。当响应中断服务子程序时，相应的中断请求标志位 TBnF 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}。f_{PSC0} 或 f_{PSC1} 输入时钟首先经过分频器，分频率由程序设置 TB0C 和 TB1C 寄存器相关位获取合适的分频值以提供更长的时基中断周期。相应的控制时基中断周期的时钟源可通过 PSCR0 或 PSCR1 寄存器的 CLKSEL1[1:0] 和 CLKSEL0[1:0] 位选择。



时基中断

PSCR0 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 时钟源选择
00：f_{SYS}
01：$f_{SYS}/4$
1x：f_{SUB}

PSCR1 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 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　**TB02~TB00**：选择时基 0 溢出周期位
000：$2^8/f_{PSC0}$
001：$2^9/f_{PSC0}$
010：$2^{10}/f_{PSC0}$
011：$2^{11}/f_{PSC0}$
100：$2^{12}/f_{PSC0}$
101：$2^{13}/f_{PSC0}$
110：$2^{14}/f_{PSC0}$
111：$2^{15}/f_{PSC0}$

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**：选择时基 1 溢出周期位
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：$2^{15}/f_{PSC1}$

串行接口模块中断

串行接口模块中断，即 SIM 中断，属于多功能中断。当一个字节数据已由 SIM 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 SIMF 被置位，SIM 中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI、串行接口中断使能位 SIME 和多功能中断使能位需先被置位。当中断使能，堆栈未满且以上任一种情况发生时，可跳转至相关多功能中断向量子程序中执行。当串行接口中断响应，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 SIMF 标志需在应用程序中手动清除。

LVD 中断

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

EEPROM 中断

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

TM 中断

简易型、标准型和周期型 TM 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。

中断唤醒功能

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

编程注意事项

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

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。

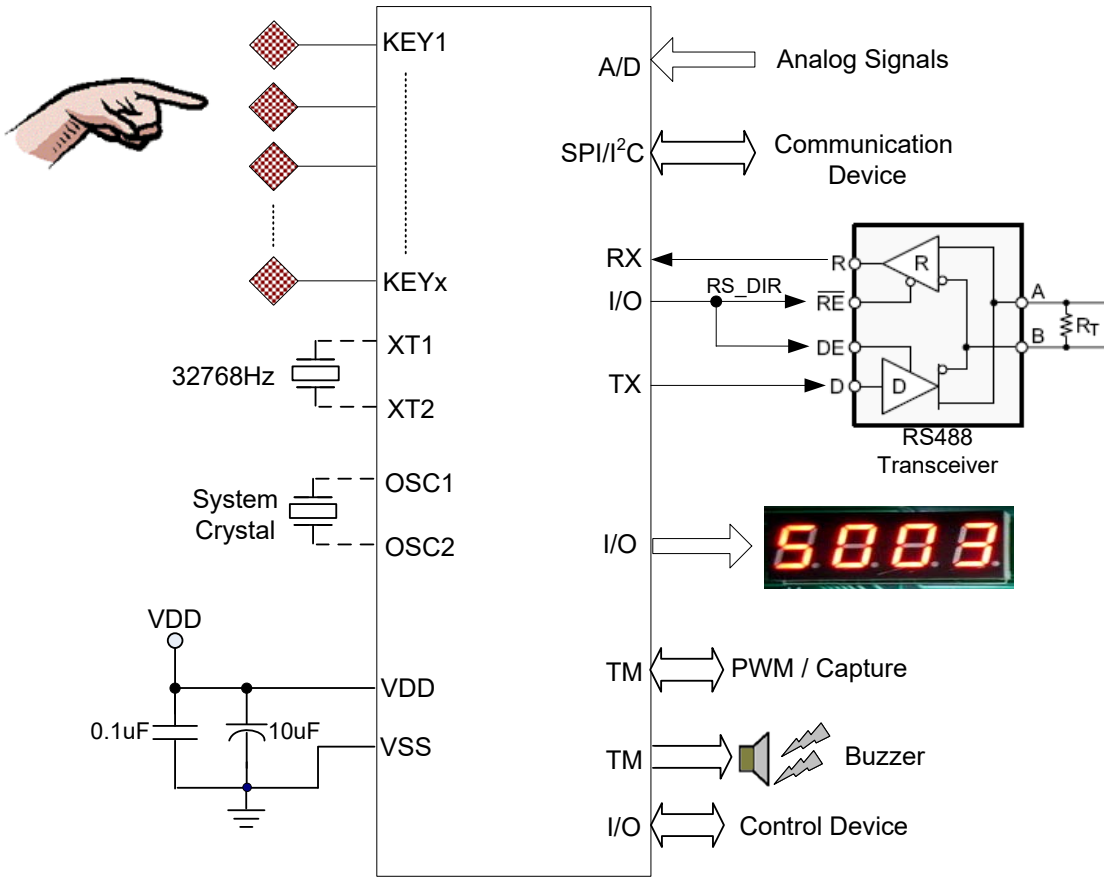
建议在中断服务子程序中不要使用“CALL 子程序”指令。中断通常发生在不可预料的情况或是需要立刻执行的某些应用。假如只剩下一层堆栈且没有控制好中断，当“CALL 子程序”在中断服务子程序中执行时，将破坏原来的控制序列。

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

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

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

应用电路


KEY1
KEYx
XT1
32768Hz
XT2
System
Crystal
OSC1
OSC2
VDD
0.1uF
10uF
VDD
VSS
A/D
Analog Signals
SPI/I2C
Communication
Device
RX
I/O
RS_DIR
TX
R
RE
DE
D
A
B
RT
RS488
Transceiver
I/O
TM
PWM / Capture
TM
Buzzer
I/O
Control Device
BS66F3x0

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他们的应用。

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，使用几种 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] ← 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

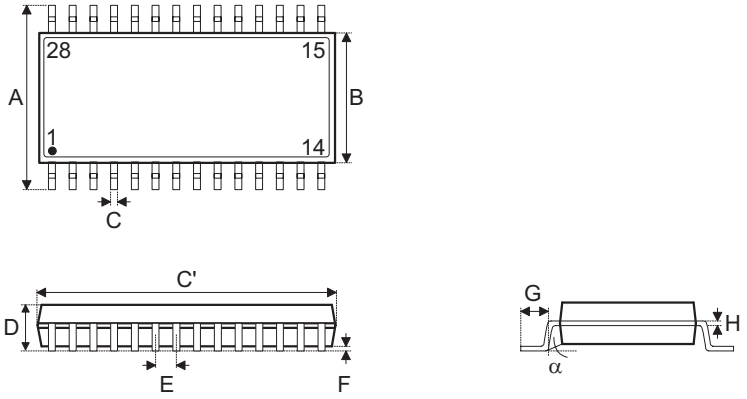
封装信息

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

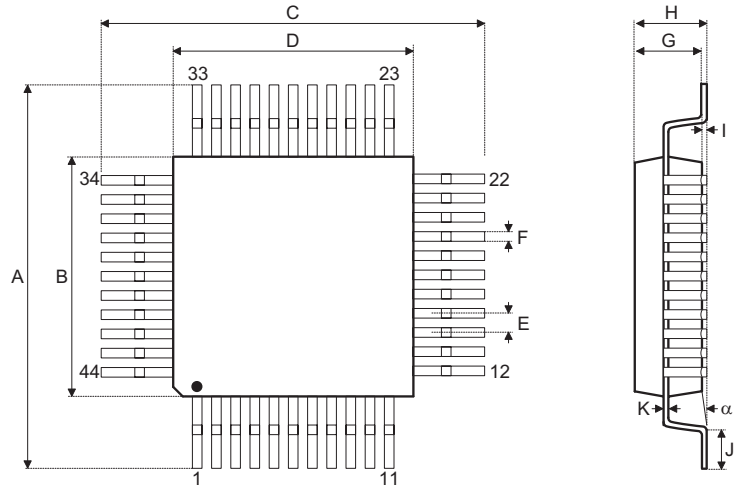
28-pin SSOP (150mil) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	正常值	最大值
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.008	—	0.012
C’	—	0.390 BSC	—
D	—	—	0.069
E	—	0.025 BSC	—
F	0.004	—	0.0098
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

符号	尺寸 (单位：mm)		
	最小值	正常值	最大值
A	—	6.0 BSC	—
B	—	3.9 BSC	—
C	0.20	—	0.30
C’	—	9.9 BSC	—
D	—	—	1.75
E	—	0.635 BSC	—
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°

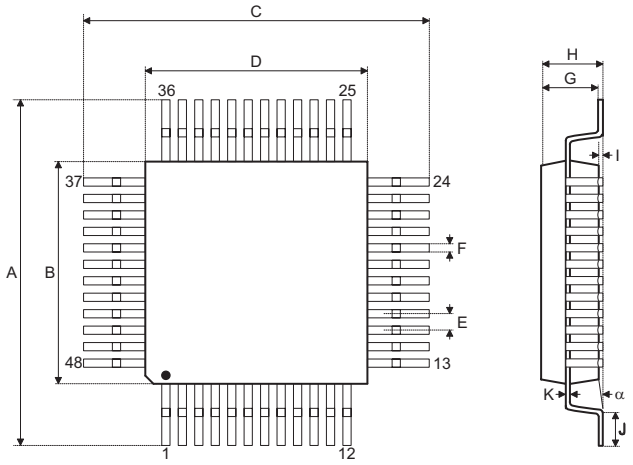
44-pin LQFP (10mm×10mm) (FP2.0mm) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	正常值	最大值
A	—	0.472 BSC	—
B	—	0.394 BSC	—
C	—	0.472 BSC	—
D	—	0.394 BSC	—
E	—	0.032 BSC	—
F	0.012	0.015	0.018
G	0.053	0.055	0.057
H	—	—	0.063
I	0.002	—	0.006
J	0.018	0.024	0.030
K	0.004	—	0.008
α	0°	—	7°

符号	尺寸 (单位：mm)		
	最小值	正常值	最大值
A	—	12.00 BSC	—
B	—	10.00 BSC	—
C	—	12.00 BSC	—
D	—	10.00 BSC	—
E	—	0.80 BSC	—
F	0.30	0.37	0.45
G	1.35	1.40	1.45
H	—	—	1.60
I	0.05	—	0.15
J	0.45	0.60	0.75
K	0.09	—	0.20
α	0°	—	7°

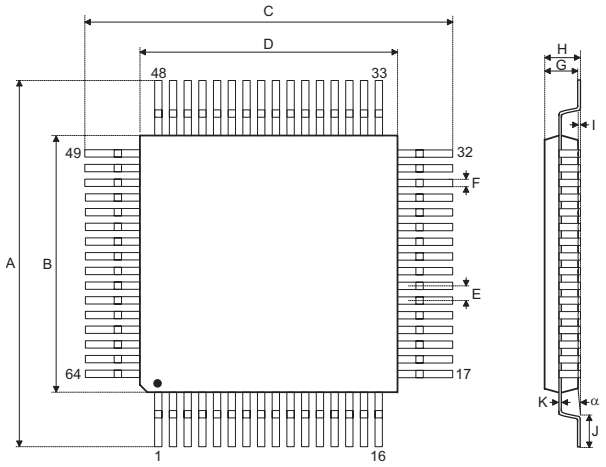
48-pin LQFP (7mm×7mm) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	正常值	最大值
A	—	0.354 BSC	—
B	—	0.276 BSC	—
C	—	0.354 BSC	—
D	—	0.276 BSC	—
E	—	0.020 BSC	—
F	0.007	0.009	0.011
G	0.053	0.055	0.057
H	—	—	0.063
I	0.002	—	0.006
J	0.018	0.024	0.030
K	0.004	—	0.008
α	0°	—	7°

符号	尺寸 (单位：mm)		
	最小值	正常值	最大值
A	—	9.00 BSC	—
B	—	7.00 BSC	—
C	—	9.00 BSC	—
D	—	7.00 BSC	—
E	—	0.50 BSC	—
F	0.17	0.22	0.27
G	1.35	1.40	1.45
H	—	—	1.60
I	0.05	—	0.15
J	0.45	0.60	0.75
K	0.09	—	0.20
α	0°	—	7°

64-pin LQFP (7mm×7mm) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	正常值	最大值
A	—	0.354 BSC	—
B	—	0.276 BSC	—
C	—	0.354 BSC	—
D	—	0.276 BSC	—
E	—	0.016 BSC	—
F	0.005	0.007	0.009
G	0.053	0.055	0.057
H	—	—	0.063
I	0.002	—	0.006
J	0.018	0.024	0.030
K	0.004	—	0.008
α	0°	—	7°

符号	尺寸 (单位：mm)		
	最小值	正常值	最大值
A	—	9.00 BSC	—
B	—	7.00 BSC	—
C	—	9.00 BSC	—
D	—	7.00 BSC	—
E	—	0.40 BSC	—
F	0.13	0.18	0.23
G	1.35	1.40	1.45
H	—	—	1.60
I	0.05	—	0.15
J	0.45	0.60	0.75
K	0.09	—	0.20
α	0°	—	7°