内置运算放大器 & 比较器的 TinyPower™ 8-bit Flash 单片机

HT45F23A/HT45F24A

版本 : V1.10　日期 : 2014-05-22

www.holtek.com

目录

特性

CPU 特性

- 工作电压：
  - f_{SYS}=32.768kHz：2.2V~5.5V
  - f_{SYS}=910kHz：2.2V~5.5V
  - f_{SYS}=2MHz：2.2V~5.5V
  - f_{SYS}=4MHz：2.2V~5.5V
  - f_{SYS}=8MHz：3.3V~5.5V
- 低功耗架构（TinyPowerTM 技术）
- 提供暂停和唤醒功能，以降低功耗
- 振荡器类型：
  - 外部晶振 -- HXT
  - 外部 32.768kHz 晶振 -- LXT
  - 外部 RC -- ERC
  - 内部 RC -- HIRC
  - 内部 32kHz RC – LIRC
  - 外部时钟 – EC
- 多种工作模式：正常、低速、空闲和休眠
- 内部集成 32kHz，910kHz，2MHz，4MHz 和 8MHz 振荡器，无需外接元件
- 提供外部系统时钟选择
- 所有指令都可在 1 或 2 个指令周期内完成
- 查表指令
- 61 或 63 条指令
- 6 层堆栈
- 位操作指令

周边特性

- Flash 程序存储：2K×15~4K×16
- RAM 数据存储：128×8~192×8
- EEPROM 存储器：64×8
- 看门狗定时器功能
- 多达 26 个双向 I/O 口
- 4 个软件控制 SCOM 口 1/2 bias LCD 驱动
- 多个引脚与外部中断口共用
- 一个 8-bit 和一个 16-bit 具有溢出中断功能的可编程定时 / 计数器
- 双时基功能
- 串行接口模块 -- SIM，用于 SPI 或 I^2C 通信
- 两个比较器
- 两个运算放大器
- 运算放大器输出可连接到内部 2 通道 12-bit 的 A/D 转换器
- 多通道 12-bit 的 A/D 转换器
- 2 通道 8-bit PWM
- 12-bit 音频 D/A 转换器输出
- PFD/Buzzer 用于产生音频频率
- 内置 2.4V/3.3V LDO
- 低电压复位功能
- 低电压检测功能
- 16-pin NSOP，20/24/28-pin SSOP 封装类型

概述

HT45F23A 和 HT45F24A 是具有 A/D 功能及 HOLTEK 低功耗架构的 8 位高性能精简指令集的 Flash 型单片机，专门为需要直接接模拟信号的应用产品所设计。单片机具有更高的增益带宽，更适合较高频率的应用。

单片机具有一系列功能和特性，其 Flash 存储器可多次编程的特性给用户提供了极大的方便。存储器方面，还包含了一个 RAM 数据存储器和一个可用于存储序号、校准数据等非易失性数据的 EEPROM 存储器。

在模拟特性方面，单片机包含一个多通道 12-bit A/D 转换器、两路 PWM 输出、两个运算放大器、两个比较器、一个内置 2.4V 或 3.3V LDO 电压调节器和一个用于语音输出的 12-bit D/A 转换器电路。内建完整的 SPI 和 I^2C 功能，为设计者提供了一个易与外部硬件通信的接口。内部看门狗定时器、低电压复位和低电压检测等内部保护特性，外加优秀的抗干扰和 ESD 保护性能，确保单片机在恶劣的电磁干扰环境下可靠地运行。

单片机内建完整的系统振荡器，无需外围元器件。提供了丰富的振荡器功能选项，包括内部、外部、低速和高速振荡器功能。盛群半导体独特低功耗技术也使单片机具有极低电流功耗的特性，该特性对于要求低功耗的电池电源应用方面极为重要。秉承 HOLTEK 单片机的一般特性，单片机带有暂停和唤醒功能，振荡器选项，可编程分频器等丰富的功能选项，只需使用极少的外部元器件便可实现用户的需求。

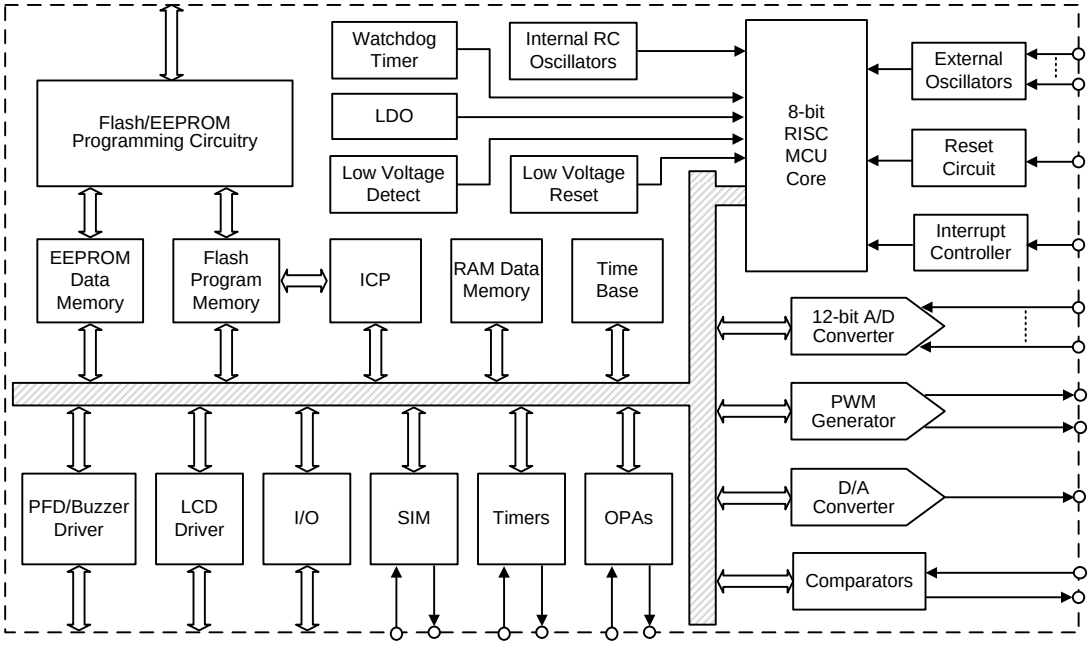
外加时基功能、I/O 使用灵活等其它特性，使单片机可以广泛应用于各种产品中，例如烟感器、电子测量仪器、环境监控、手持式测量工具、家庭应用、电子控制工具、马达控制、家庭安全系统以及其它方面。

选型表

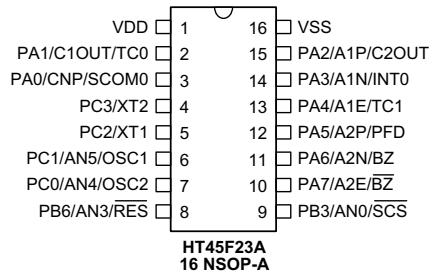
型号	程序存储器	数据存储器	EEPROM 存储器	I/O	外部中断
HT45F23A	2K×15	128×8	64×8	22	2
HT45F24A	4K×16	192×8	64×8	26	2

型号	A/D	定时器	时基	堆栈	封装
HT45F23A	12-bit×6	8-bit×1 16-bit×1	2	6	16 NSOP 20/24 SSOP
HT45F24A	12-bit×8	8-bit×1 16-bit×1	2	6	20/24/28 SSOP

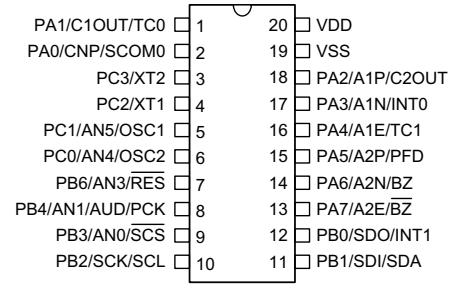
方框图



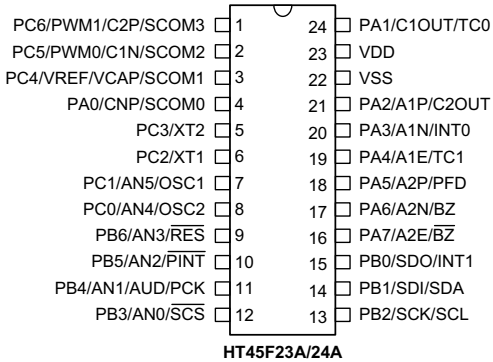
引脚图



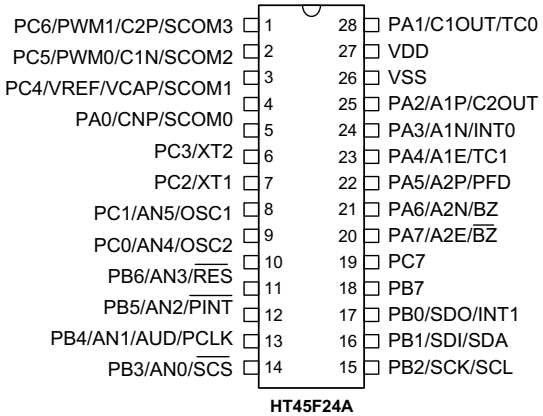
HT45F23A
16 NSOP-A



HT45F23A/24A
20 SSOP-A



HT45F23A/24A
24 SSOP-A



HT45F24A
28 SSOP-A

引脚说明

引脚名称	功能	OPT	I/T	O/T	说明
PA0/CNP/SCOM0	PA0	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	CNP	CMP1C1	CMPI	—	比较器输入脚
	SCOM0	LCDC	—	SCOM	软件控制 1/2 bias LCD COM
PA1/C1OUT/TC0	PA1	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	C1OUT	CMP1C1	—	CMPO	比较器 1 输出脚
	TC0	—	ST	—	外部定时器 0 时钟输入脚
PA2/A1P/C2OUT	PA2	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	A1P	OPA1C1	OPAI	—	OPA1 同相输入脚
	C2OUT	CMP2C1	—	CMPO	比较器 2 输出脚
PA3/A1N/INT0	PA3	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	A1N	OPA1C1	OPAI	—	OPA1 反相输入脚
	INT0	—	ST	—	外部中断 0 输入脚
PA4/A1E/TC1	PA4	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	A1E	OPA1C1	—	OPAO	OPA1 输出脚
	TC1	—	ST	—	外部定时器 1 时钟输入脚
PA5/A2P/PFD	PA5	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	A2P	OPA2C1	OPAI	—	OPA2 同相输入脚
	PFD	MISC	—	CMOS	PFD 输出
PA6/A2N/BZ	PA6	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	A2N	OPA2C1	OPAI	—	OPA2 反相输入脚
	BZ	BPCTL	—	CMOS	蜂鸣器输出脚
PA7/A2E/$\overline{BZ}$	PA7	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	A2E	OPA2C1	—	OPAO	OPA2 输出脚
	$\overline{BZ}$	BPCTL	—	CMOS	蜂鸣器互补输出脚
PB0/SDO/INT1	PB0	PBPU MISC	ST	CMOS NMOS	通用 I/O 口，可通过寄存器设置上拉电阻，具有 NMOS 输出结构
	SDO	—	—	CMOS	SPI 数据输出脚
	INT1	—	ST	—	外部中断 1 输入脚
PB1/SDI/SDA	PB1	PBPU MISC	ST	CMOS NMOS	通用 I/O 口，可通过寄存器设置上拉电阻，具有 NMOS 输出结构
	SDI	—	ST	—	SPI 数据输入脚
	SDA	—	ST	NMOS	I^2C 数据线

引脚名称	功能	OPT	I/T	O/T	说明
PB2/SCK/SCL	PB2	PBPU MISC	ST	CMOS NMOS	通用 I/O 口，可通过寄存器设置上拉电阻，具有 NMOS 输出结构
	SCK	—	ST	—	SPI 串行时钟线
	SCL	—	ST	NMOS	I^2C 时钟线
PB3/AN0/$\overline{SCS}$	PB3	PBPU MISC	ST	CMOS NMOS	通用 I/O 口，可通过寄存器设置上拉电阻，具有 NMOS 输出结构
	AN0	ADCR	AN	—	A/D 通道 0
	$\overline{SCS}$	—	ST	—	SPI 从机片选脚
PB4/AN1/AUD /PCK	PB4	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN1	ADCR	AN	CMOS	A/D 通道 1
	AUD	DACTRL	—	—	D/A 输出脚
	PCK	—	—	CMOS	外围时钟输出
PB5/AN2/$\overline{PINT}$	PB5	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN2	ADCR	AN	—	A/D 通道 2
	$\overline{PINT}$	—	ST	—	外围中断
PB6/AN3/$\overline{RES}$	PB6	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN3	ADCR	AN	—	A/D 通道 3
	$\overline{RES}$	CO	ST	—	复位脚
PB7	PB7	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
PC0/AN4/OSC2	PC0	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN4	ADCR	AN	—	A/D 通道 4
	OSC2	CO	—	HXT	HXT 脚
PC1/AN5/OSC1	PC1	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN5	ADCR	AN	—	A/D 通道 5
	OSC1	CO	HXT	—	HXT/ERC 脚
PC2/XT1	PC2	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	XT1	CO	LXT	—	LXT 脚
PC3/XT2	PC3	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	XT2	CO	—	LXT	LXT 脚
PC4/VREF/VCAP /SCOM1	PC4	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	VREF	ACSR	AN	—	A/D 转换器参考电压输入
	VCAP	LDOC	—	—	LDO 输出脚，连接一个 0.1μF 电容到地。
	SCOM1	LCDC	—	SCOM	软件控制 1/2 bias LCD COM
PC5/PWM0/C1N /SCOM2	PC5	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PWM0	BPCTL	—	CMOS	PWM0 输出脚
	C1N	CMP1C1	CMPI	—	比较器 1 反相输入脚
	SCOM2	LCDC	—	SCOM	软件控制 1/2 bias LCD COM
PC6/PWM1/C2P /SCOM3	PC6	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PWM1	BPCTL	—	CMOS	PWM1 输出脚
	C2P	CMP2C1	CMPI	—	比较器 2 同相输入脚
	SCOM3	LCDC	—	SCOM	软件控制 1/2 bias LCD COM
PC7	PC7	PCPU	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
PD0/AN6	PD0	PDPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN6	ADCR	AN	—	A/D 通道 6
PD1/AN7	PD1	PDPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN7	ADCR	AN	—	A/D 通道 7
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：通过配置选项（CO）或者寄存器选项来配置
PWR：电源；　　CO：配置选项
ST：施密特触发输入；　　DAO：D/A 输出
CMOS：CMOS 输出；　　NMOS：NMOS 输出
SCOM：软件控制的 LCD COM
HXT：高频晶体振荡器；　　LXT：低频晶体振荡器
OPAI：运算放大器输入脚；　　OPAO：运算放大器输出脚
CMPI：比较器输入脚；　　CMPO：比较器输出脚
AN：模拟输入
PB7、PC7、PD0/AN6 和 PD1/AN7 仅存在于 HT45F24A 中

极限参数

电源供应电压 ..... V_{SS}-0.3V~V_{SS}+6.0V
储存温度 ..... -50˚C~125˚C
端口输入电压 ..... V_{SS}-0.3V~V_{DD}+0.3V
工作温度 .....-40˚C~85˚C
I_{OL} 总电流.....100mA
I_{OH} 总电流 .....-100mA
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°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DD}	工作电压	—	f_{SYS}=910kHz (HXT/ERC/HIRC)	2.2	—	5.5	V
			f_{SYS}=2MHz (HXT/ERC/HIRC)	2.2	—	5.5	V
			f_{SYS}=4MHz (HXT/ERC/HIRC/EC)	2.2	—	5.5	V
			f_{SYS}=8MHz (HXT/ERC/HIRC/EC)	3.3	—	5.5	V
I_{DD1}	工作电流 (HXT, ERC)	3.3V	无负载，$f_{SYS} = f_M$ = 455kHz，ADC off，LVR off，Comparator off，OPAs off	—	70	110	μA
			无负载，$f_{SYS} = f_M$ =455kHz，ADC off，LVR on，Comparator on，OPAs off	—	100	150	μA
I_{DD2}	工作电流 (ERC, HIRC)	3.3V	无负载，f_M= 910kHz $f_{SYS}=f_{SLOW}$= 455kHz, ADC off，LVR off，Comparator off，OPAs off	—	90	135	μA
			无负载，f_M= 910kHz $f_{SYS}=f_{SLOW}$= 455kHz, ADC off，LVR on，Comparator on，OPAs off	—	120	180	μA
I_{DD3}	工作电流 (ERC, HIRC)	3.3V	无负载，$f_{SYS} = f_M$ =910kHz，ADC off，LVR off，Comparator off，OPAs off	—	110	170	μA
			无负载，$f_{SYS} = f_M$ =910kHz，ADC off，LVR on，Comparator on，OPAs off	—	160	240	μA
I_{DD4}	工作电流 (HXT, ERC)	3.3V	无负载，$f_{SYS} = f_M$ =1MHz，ADC off，LVR off，Comparator off，OPAs off	—	120	180	μA
			无负载，$f_{SYS} = f_M$ =1MHz，ADC off，LVR on，Comparator on，OPAs off	—	170	260	μA

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{DD5}	工作电流 (HXT, ERC, HIRC)	3.3V	无负载， $f_{SYS} = f_M$ =2MHz， ADC off，LVR off， Comparator off，OPAs off	—	170	260	μA
			无负载， $f_{SYS} = f_M$ =2MHz， ADC off，LVR on， Comparator on，OPAs off	—	200	300	μA
I_{DD6}	工作电流 (HXT, ERC, HIRC)	3V	无负载， $f_{SYS} = f_M$ =4MHz，ADC off	—	420	630	μA
		5V		—	700	1000	μA
I_{DD7}	工作电流 (EC)	3V	无负载， $f_{SYS} = f_M$ =4MHz，ADC off	—	330	500	μA
		5V		—	550	820	μA
I_{DD8}	工作电流 (HXT, ERC, HIRC)	5V	无负载， $f_{SYS} = f_M$ =8MHz，ADC off	—	1.5	3.0	mA
I_{DD9}	工作电流 (低速模式 , f_M=4MHz) (HXT, ERC, HIRC)	3V	无负载，ADC off $f_{SYS}=f_{SLOW}$= 1MHz，	—	200	300	μA
		5V		—	400	600	μA
I_{DD10}	工作电流 (低速模式 , f_M=4MHz) (HXT, ERC, HIRC)	3V	无负载，ADC off $f_{SYS}=f_{SLOW}$= 2MHz，	—	250	375	μA
		5V		—	560	840	μA
I_{DD11}	工作电流 (低速模式 , f_M=8MHz) (HXT, ERC, HIRC)	3V	无负载，ADC off $f_{SYS}=f_{SLOW}$= 2MHz，	—	300	450	μA
		5V		—	680	1020	μA
I_{DD12}	工作电流 (低速模式 ,f_M=8MHz) (HXT, ERC, HIRC)	3V	无负载，ADC off $f_{SYS}=f_{SLOW}$= 4MHz，	—	450	800	μA
		5V		—	1000	1500	μA
I_{DD13}	工作电流 ($f_{SYS}=f_{LXT}$ 注 1 或 f_{LIRC})	3V	无负载，WDT off， ADC off	—	10	20	μA
		5V		—	20	35	μA
I_{STB1}	静态电流 (休眠模式) (f_{SYS}, f_{SUB}, f_S, f_{WDT}=off)	3V	无负载，WDT off, 系统进入 HALT	—	0.1	1.0	μA
		5V		—	0.2	2.0	μA
I_{STB2}	静态电流 (休眠模式) (f_{SYS} off, f_S on, $f_{WDT}=f_{SUB}=f_{LXT}$ 注 1 或 f_{LIRC})	3V	无负载，WDT on, 系统进入 HALT	—	1.5	3.0	μA
		5V		—	2.5	5.0	μA
I_{STB3}	静态电流 (空闲模式) (f_{SYS} off, f_{WDT} off, $f_S=f_{SUB}= f_{LXT}$ 注 1 或 f_{LIRC})	3V	无负载，WDT off, 系统进入 HALT	—	4	6	μA
		5V		—	6	9	μA
I_{STB4}	静态电流 (空闲模式) (f_{SYS} on, f_{WDT} off; $f_{SYS} = f_M$ =4MHz; $f_S=f_{SUB}= f_{LXT}$ 注 1 或 f_{LIRC})	3V	无负载，WDT off， 系统进入 HALT， SPI 或 I^2C on，PCK on， $f_{PCK}=f_{SYS}/8$	—	260	350	μA
		5V		—	350	660	μA

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{IL1}	输入 / 输出口、TCn 脚及 INTn 脚的低电平输入电压	—	—	0	—	$0.3V_{DD}$	V
V_{IH1}	输入 / 输出口、TCn 脚及 INTn 脚的高电平输入电压	—	—	$0.7V_{DD}$	—	V_{DD}	V
V_{IL2}	低电平输入电压 ($\overline{RES}$)	—	—	0	—	$0.4V_{DD}$	V
V_{IH2}	高电平输入电压 ($\overline{RES}$)	—	—	$0.9V_{DD}$	—	V_{DD}	V
V_{IL3}	低电平输入电压 (PB1~PB3)	5V	—	—	—	1	V
V_{IH3}	高电平输入电压 (PB1~PB3)	5V	—	2	—	—	V
V_{LVR1}	低电压复位电压	—	V_{LVR} =2.10V	-5%	2.10	+5%	V
V_{LVR2}			V_{LVR} =2.55V		2.55		
V_{LVR3}			V_{LVR} =3.15V		3.15		
V_{LVR4}			V_{LVR} =4.20V		4.20		
V_{LVD1}	低电压检测电压	—	LVDEN=1, V_{LVD}=2.0V	-5%	2.0	+5%	V
V_{LVD2}			LVDEN=1, V_{LVD}=2.2V		2.2		
V_{LVD3}			LVDEN=1, V_{LVD}=2.4V		2.4		
V_{LVD4}			LVDEN=1, V_{LVD}=2.7V		2.7		
V_{LVD5}			LVDEN=1, V_{LVD}=3.0V		3.0		
V_{LVD6}			LVDEN=1, V_{LVD}=3.3V		3.3		
V_{LVD7}			LVDEN=1, V_{LVD}=3.6V		3.6		
V_{LVD8}			LVDEN=1, V_{LVD}=4.4V		4.4		
I_{OL}	输入 / 输出口灌电流	3V	V_{OL}=0.1V_{DD}	6	12	—	mA
		5V		10	25	—	mA
I_{OH}	输入 / 输出口源电流	3V	V_{OH}=0.9V_{DD}	-2	-4	—	mA
		5V		-5	-8	—	mA
R_{PH}	上拉电阻	3V	—	40	60	80	kΩ
		5V	—	10	30	50	kΩ
AV_{DD}	ADC 工作电压	—	—	2.7	—	5.5	V
V_{AD}	A/D 输入电压	—	—	0	—	V_{REF}	V
V_{REF}	A/D 输入参考电压范围	—	AV_{DD}=5V	2	—	V_{DD}	V
DNL	A/D 非线性微分误差	3V	V_{REF}=V_{DD}, t_{AD}=1μs	—	±1	±2	LSB
		5V					
INL	A/D 非线性积分误差	3V	V_{REF}=V_{DD}, t_{AD}=1μs	—	±2	±4	LSB
		5V					
I_{ADC}	打开 A/D 增加的功耗	3V	—	—	0.5	1.0	mA
		5V		—	1.5	3.0	mA
V_{BG}	带隙参考缓冲电压	—	—	-3%	1.25	+3%	V
I_{LVR}	LVD 或 LVR 开启时的 DC 电流	3V	—	—	10	15	μA
		5V		—	20	30	

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{SCOM}	LCD COM V_{DD}/2 电压	5V	无负载	0.475	0.500	0.525	V_{DD}
	LCD COM V_{LDO}/2 电压	5V	无负载	0.475	0.500	0.525	V_{LDO}
I_{OUT}	输出电流	5V	ISEL=0, LCDBUF=0	—	10	—	μA
			ISEL=1, LCDBUF=0	—	25	—	μA
			ISEL=0, LCDBUF=1	—	2	—	mA
			ISEL=1, LCDBUF=1	—	2	—	mA

注：1. t_{SYS} =1/f_{SYS}；t_{SUB} =1/f_{SUB}
2. 细节请参考 LDO 章节

交流电气特性

Ta=25˚C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
f_{SYS1}	系统时钟 (HXT，ERC，HIRC)	—	2.2V~5.5V	400	—	4000	kHz
			3.3V~5.5V	400	—	8000	kHz
f_{SYS2}	8MHz HIRC	3.3V	Ta=25˚C	-2%	8	+2%	MHz
		3.3V	Ta=-40˚C~85˚C	-5%	8	+5%	MHz
		2.7V~5.5V	Ta=-40˚C~85˚C	-10%	8	+10%	MHz
f_{SYS3}	4MHz HIRC	3.3V	Ta=25˚C	-2%	4	+2%	MHz
		3.3V	Ta=-40˚C~85˚C	-5%	4	+5%	MHz
		2.7V~5.5V	Ta=-40˚C~85˚C	-10%	4	+10%	MHz
f_{SYS4}	2MHz HIRC	3.3V	Ta=25˚C	-2%	2	+2%	MHz
		3.3V	Ta=-40˚C~85˚C	-5%	2	+5%	MHz
		2.7V~5.5V	Ta=-40˚C~85˚C	-10%	2	+10%	MHz
f_{SYS5}	910kHz HIRC	3.3V	Ta=25˚C	-2%	0.91	+2%	MHz
		3.3V	Ta=-40˚C~85˚C	-5%	0.91	+5%	MHz
		2.7V~5.5V	Ta=-40˚C~85˚C	-10%	0.91	+10%	MHz
f_{LXT}	系统时钟 (LXT)	—	—	—	32.768	—	kHz
f_{ERC}	4MHz ERC 注 3	3.3V	R=150kΩ，Ta=25˚C	-2%	4	+2%	MHz
		3.3V	R=150kΩ，Ta=-40˚C~85˚C	-8%	4	+8%	MHz
		2.7V~5.5V	R=150kΩ，Ta=-40˚C~85˚C	-15%	4	+15%	MHz
t_{LIRC}	32kHz RC 周期	3V	—	28.10	31.25	34.40	μs
t_{RES}	外部复位低电平脉宽	—	—	1	—	—	μs
t_{LVR}	低压复位时间	—	—	60	120	240	μs
t_{LVD}	低压中断时间	—	—	60	120	240	μs

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
t_{LVDS}	LVDO 稳定的时间	5V	LVR 除能， LVD 使能， V_{BG} 就绪	—	—	100	μs
t_{RSTD}	复位延迟时间	—	—	—	100	—	ms
t_{SST1}	系统启动时间 (W/O 快速启动， HXT/TBC)	—	上电或由休眠模式中唤醒	—	1024	—	t_{SYS}* 注 1
t_{SST2}	系统启动时间 (ERC，HIRC，EC)	—	上电或由 HALT (空闲或休眠模式) 中唤醒	—	1	2	t_{SYS}
t_{SST3}	系统启动时间 (快速启动，HXT/TBC)	—	由空闲模式中唤醒 (f_{SL}=f_{TBC})	—	1	2	t_{TBC} 注 2
t_{SST4}	系统启动时间 (快速启动，HXT/TBC)	—	由空闲模式中唤醒 (f_{SL}=f_{LIRC})	—	1	2	t_{LIRC}
t_{INT}	中断脉宽	—	—	1	—	—	μs
t_{AD}	A/D 时钟周期	—	—	0.5	—	—	μs
t_{ADC}	A/D 转换时间注 4	—	—	—	16	—	t_{AD}
t_{ON2ST}	A/D 开启到 A/D 开始工作的时间	—	2.2V~5.5V	2	—	—	μs

注：1. t_{SYS} =1/f_{SYS}；t_{SUB} =1/ f_{SUB}

2. t_{TBC} 是 LXT 或 LIRC 振荡器的周期，在空闲模式下，该振荡器继续运行。

3. 对于 f_{ERC}，由于电阻精度会影响频率，建议使用精密电阻。

4. A/D 转换时间 (t_{AD})=n (ADC 位数)+4 (采样时间)，每位的转换时间需要 1 个 ADC 时钟 (t_{AD})。

放大器电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
DC 电气特性							
V_{DD}	工作电压	—	—	2.2	—	5.5	V
I_{DD}	静态电流	5V	无负载，A1OEN/A2OEN 固定为 0	—	200	350	μA
V_{OPOS}	输入失调电压	5V	—	-2	—	+2	mV
I_{OPOS}	输入失调电流	—	V_{DD}=5V，V_{CM}=1/2V_{DD}，Ta=-40°C~85°C	—	10	—	nA
V_{CM}	共模电压范围	—	—	V_{SS}	—	V_{DD}-1.4V	V
PSRR	电源电压抑制比	—	—	58	80	—	dB
CMRR	共模抑制比	—	V_{DD}=5V，V_{CM}=0~V_{DD}-1.4V	58	80	—	dB
AC 电气特性							
A_{OL}	开环增益	—	—	60	80	—	dB
SR	电压转换速率	—	无负载	—	0.1	—	V/μs
GBW	增益带宽	—	R_L=1MΩ，C_L=100pF	1	2	—	MHz

比较器电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DDC}	比较器工作电压	—	—	2.2	—	5.5	V
I_{DDC}	比较器工作电流	3V	—	—	20	40	μA
		5V		—	30	60	μA
V_{CPOS1}	比较器输入失调电压	5V	CxOF[4:0]=10000	-10	—	10	mV
V_{CPOS2}	比较器输入失调电压	5V	校准后	-4	—	4	mV
V_{CM}	比较器共模电压范围	—	—	V_{SS}	—	V_{DD}-1.4V	V
A_{OL}	比较器开环增益	—	—	60	80	—	dB
t_{PD1}	比较器响应时间	—	2mV 偏置	—	—	2	μs
t_{PD2}	比较器响应时间	—	10mV 偏置	—	—	1.5	μs

LDO 2.4V

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DDIN}	电源电压	—	—	2.7	—	5.5	V
V_{DDOUT}	输出电压	—	—	2.28	2.40	2.52	V
I_{DD}	耗电流	—	启动后，无负载	—	50	70	μA
I_{OUT}	输出电流	5V	V_{CAP}=1μF	200	—	1100	μA

LDO 3.3V

Ta=25°C

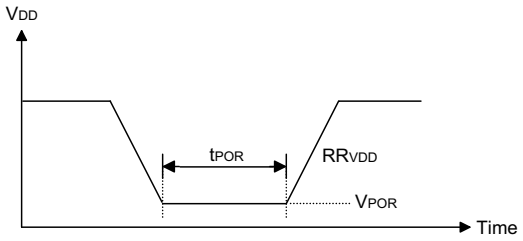
符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DDIN}	电源电压	—	—	3.6	—	5.5	V
V_{DDOUT}	输出电压	—	—	3.13	3.30	3.46	V
I_{DD}	耗电流	—	启动后，无负载	—	50	70	μA
I_{OUT}	输出电流	5V	V_{CAP}=1μF	200	—	1100	μA

注：1. 该 LDO 可为带 10μF 电容的 PIR 传感器提供一个稳定的电源电压。
2.VREF 脚应连接一个 10μF 电容为 A/D 转换器和 PIR 传感器提供参考电压。

上电复位特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{POR}	上电复位电压	—	—	—	—	100	mV
RR_{VDD}	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

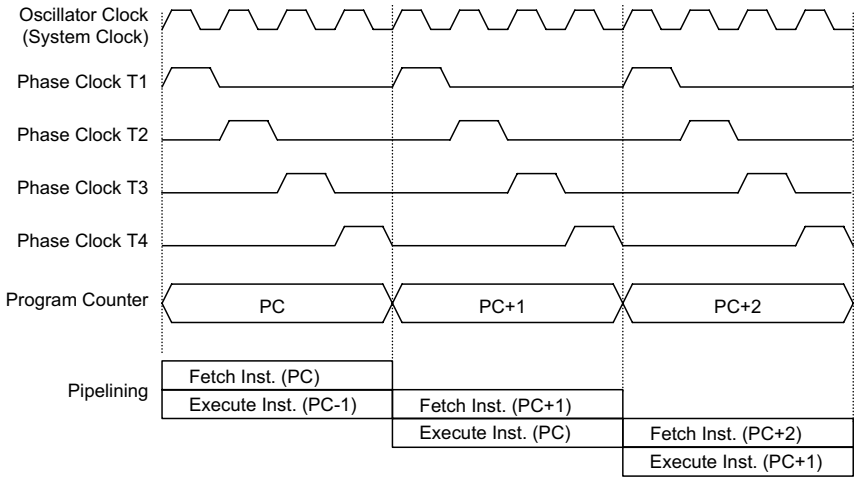


系统结构

内部系统结构是盛群单片机具有良好性能的主要因素。由于采用 RISC 结构，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，确保了在提供具有最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。使得这些单片机适用于低成本和大批量的控制应用。

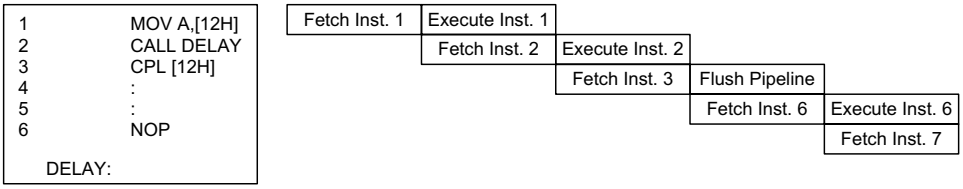
时序和流水线结构

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



系统时序和流水线

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



指令捕捉

程序计数器

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

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

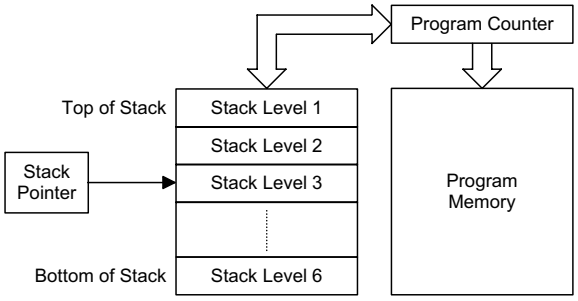
程序计数器		
单片机	程序计数器高字节	PCL 寄存器
HT45F23A	PC10~PC8	PCL7~PCL0
HT45F24A	PC11~PC8	PCL7~PCL0

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。单片机有 6 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。

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

Flash 程序存储器

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

结构

程序存储器的容量为 2K×15~4K×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。

特殊向量

程序存储器内部某些地址保留，被用作诸如复位和中断入口等特殊用途。

地址 000H

该地址为程序初始化保留。系统复位后，程序总是从 000H 开始执行。

地址 004H

该地址为外部中断服务程序保留。如果 INT0 输入脚有效，中断允许且堆栈未满，则程序会跳转到 004H 地址开始执行。

地址 008H

该地址为外部中断服务程序保留。如果 INT1 输入脚有效，中断允许且堆栈未满，则程序会跳转到 008H 地址开始执行。

地址 00CH

该地址为定时 / 计数器 0 中断服务程序保留。当定时 / 计数器 0 溢出，如果中断允许且堆栈未满，则程序会跳转到 00CH 地址开始执行。

地址 010H

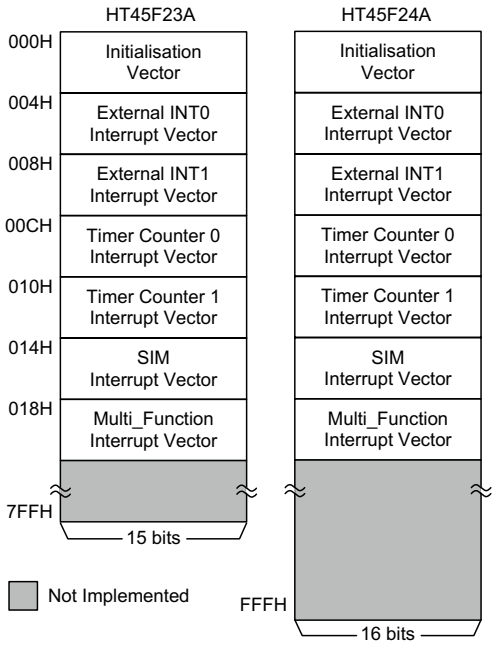
该地址为定时 / 计数器 1 中断服务程序保留。当定时 / 计数器 1 溢出，如果中断允许且堆栈未满，则程序会跳转到 010H 地址开始执行。

地址 014H

该地址为 SIM 中断服务程序保留。若 SIM 中断发生，需要数据传输，中断允许且堆栈未满，则程序会跳转到 014H 地址开始执行。

地址 018H

该地址为多功能中断服务程序保留。当时基溢出，A/D 转换完成，外围设备中断引脚出现上升沿，比较器输出中断，EEPROM 读写周期完成中断或 LVD 检测中断，当这些中断中的任意一个发生，如果中断允许且堆栈未满，则程序会跳转到 018H 地址开始执行。



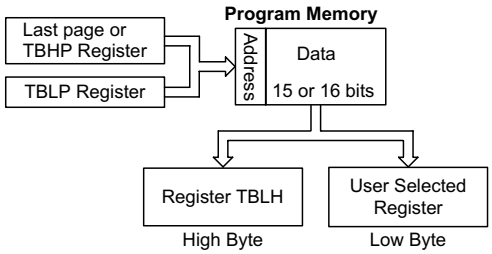
程序存储器结构

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。这个寄存器定义表格总的地址。

在设定完表格指针后，表格数据可以使用“TABRDC [m]”或“TABRDL [m]”指令从当前的程序所在的存储器页或存储器最后一页中来查表读取。当这些指令执行时，程序存储器中表格数据低字节，将被传送到使用者所指定的数据存储器 [m]，程序存储器中表格数据的高字节，则被传送到 TBLH 特殊寄存器，而高字节中未使用的位将被读取为“0”。

下图是查表中寻址 / 数据流程：



指令	表格地址											
	b11	**b10**	**b9**	**b8**	**b7**	**b6**	**b5**	**b4**	**b3**	**b2**	**b1**	**b0**
TABRDC [m]	PC11	PC10	PC9	PC8	@7	@6	@5	@4	@3	@2	@1	@0
TABRDL [m]	1	1	1	1	@7	@6	@5	@4	@3	@2	@1	@0

表格存储单元

注：@7~@0：表格指针位
PC11~PC8：当前程序计数器位
对于 HT45F23A，表格地址为 11 位，从 b10~b0
对于 HT45F24A，表格地址为 12 位，从 b11~b0

查表范例

以下范例说明在 HT45F24A 单片机中，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的最后一页。ORG 指令“0F00H”指向的地址是 4K 程序存储器中最后一页的起始地址。表格指针的初始值设为“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“TABRDC [m]”指令被使用，则表格指针指向当前页。在这个例子中，表格数据的高字节等于零，而当“TABRDL [m]”指令被执行时，此值将会自动的被传送到 TBLH 寄存器。

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

表格读取程序范例

```
tempreg1    db ?    ; temporary register #1
tempreg2    db ?    ; temporary register #2
    :
    :
mov a,06h           ; initialize table pointer – note that this address
                    ; is referenced
mov tblp,a          ; to the last page or present page
mov a,0Fh           ; initialise high table pointer
mov tbhp,a
    :
    :
Tabrdl tempreg1     ; transfers value in table referenced by table pointer
                    ; to tempreg1 data at prog. memory address 0F06H
                    ; transferred to tempreg1 and TBLH
dec tblp            ; reduce value of table pointer by one
tabrdl tempreg2     ; transfers value in table referenced by table pointer
                    ; to tempreg2
                    ; data at prog. memory address 0F05H transferred to
                    ; tempreg2 and TBLH
                    ; in this example the data ²1AH² is transferred to
                    ; to tempreg1 and data  ²0FH² to register tempreg2
                    ; the value “00H” will be transferred to the high byte
                    ; register TBLH
    :
    :
org 0F00h           ; sets initial address of last page
dc  00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
    :
    :
```

在线烧录

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 5 线接口的在线编程方式。用户可将进行过编程或未经过编程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写，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。

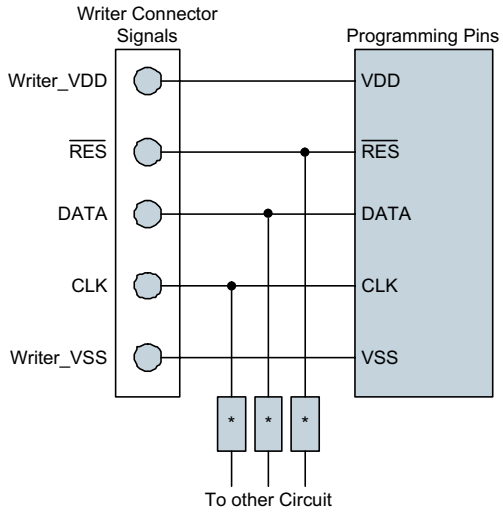
在线烧录引脚名称	功能
DATA	串行数据输入 / 输出
CLK	串行时钟
$\overline{RES}$	复位
VDD	电源
VSS	地

芯片内部程序存储器和 EEPROM 存储器都可以通过 5 线的接口在线进行烧录。其中一个引脚用于数据串行下载或上传、一个引脚用于串行时钟、两条用于提供电源，另外一条用于复位。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。

在烧录过程中，烧录器会将 $\overline{RES}$ 脚一直拉低以除能单片机工作，并控制 PA0 和 PA2 脚进行数据和时钟烧录，用户必须确保这两个引脚没有连接至其它输出脚。

在线烧录引脚	**MCU 引脚**
$\overline{RES}$	PB6
DATA	PA0
CLK	PA2

在线编程和 MCU 引脚



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。

数据存储器

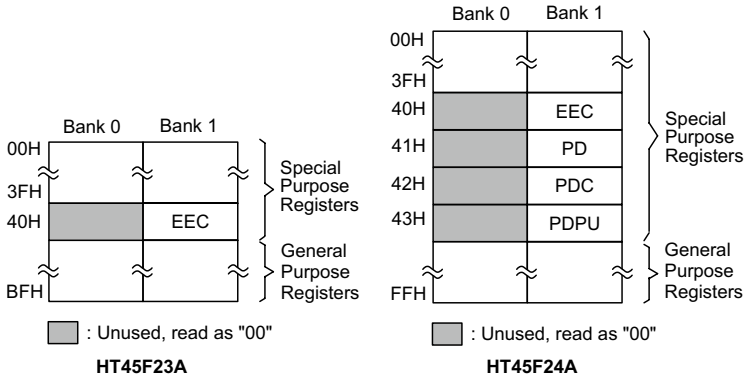
数据存储器是内容可更改的 8 位 RAM 内部存储器，用来储存临时数据。

结构

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

总的数据存储器被分为两个区。大部分特殊功能数据寄存器均可在所有 Bank 被访问，处于“40H”地址的 EEC 寄存器却只能在 Bank 1 中被访问到。切换不同区域可通过设置区域指针（BP）实现。所有单片机的数据存储器的起始地址都是“00H”。

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用户进行读取和写入的操作。使用“SET [m].i”和“CLR [m].i”指令可对个别的位做置位或复位的操作，方便用户在数据存储器内进行位操作。



数据存储器结构

注：除少数位外，大部分的数据存储器位可以通过“SET [m].i”和“CLR [m].i”位操作指令对各个位直接进行操作。也可以通过存储器指针寄存器 MP0 和 MP1 对数据存储器进行寻址。

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。

	Bank 0, 1
00H	IAR0
01H	MP0
02H	IAR1
03H	MP1
04H	BP
05H	ACC
06H	PCL
07H	TBLP
08H	TBLH
09H	TBC
0AH	STATUS
0BH	INTC0
0CH	Unused
0DH	TMR0
0EH	TMR0C
0FH	TMR1H
10H	TMR1L
11H	TMR1C
12H	PA
13H	PAC
14H	PB
15H	PBC
16H	PC
17H	PCC
18H	LVDC
19H	DACTRL
1AH	PWM0
1BH	PWM1
1CH	BPCTL
1DH	WDTC
1EH	INTC1
1FH	OPA1C0
20H	OPA1C1

	Bank 0 | Bank 1	
21H	OPA2C0	
22H	OPA2C1	
23H	OPA2C2	
24H	ADRL	
25H	ADRH	
26H	ADCR	
27H	ACSR	
28H	SMOD	
29H	PAWU	
2AH	PAPU	
2BH	PBPU	
2CH	PCPU	
2DH	ADPCR	
2EH	INTEDGE	
2FH	CMP1C0	
30H	CMP1C1	
31H	CMP2C0	
32H	CMP2C1	
33H	MISC	
34H	MFIC0	
35H	MFIC1	
36H	SIMC0	
37H	SIMC1	
38H	SIMD	
39H	SIMA/SIMC2	
3AH	EEA	
3BH	EED	
3CH	LCDC	
3DH	LDOC	
3EH	DAL	
3FH	DAH	
40H	Unused	EEC

HT45F23A

	Bank 0, 1
00H	IAR0
01H	MP0
02H	IAR1
03H	MP1
04H	BP
05H	ACC
06H	PCL
07H	TBLP
08H	TBLH
09H	TBC
0AH	STATUS
0BH	INTC0
0CH	TBHP
0DH	TMR0
0EH	TMR0C
0FH	TMR1H
10H	TMR1L
11H	TMR1C
12H	PA
13H	PAC
14H	PB
15H	PBC
16H	PC
17H	PCC
18H	LVDC
19H	DACTRL
1AH	PWM0
1BH	PWM1
1CH	BPCTL
1DH	WDTC
1EH	INTC1
1FH	OPA1C0
20H	OPA1C1

	Bank 0 | Bank 1	
21H	OPA2C0	
22H	OPA2C1	
23H	OPA2C2	
24H	ADRL	
25H	ADRH	
26H	ADCR	
27H	ACSR	
28H	SMOD	
29H	PAWU	
2AH	PAPU	
2BH	PBPU	
2CH	PCPU	
2DH	ADPCR	
2EH	INTEDGE	
2FH	CMP1C0	
30H	CMP1C1	
31H	CMP2C0	
32H	CMP2C1	
33H	MISC	
34H	MFIC0	
35H	MFIC1	
36H	SIMC0	
37H	SIMC1	
38H	SIMD	
39H	SIMA/SIMC2	
3AH	EEA	
3BH	EED	
3CH	LCDC	
3DH	LDOC	
3EH	DAL	
3FH	DAH	
40H	Unused	EEC
41H	Unused	PD
42H	Unused	PDC
43H	Unused	PDPU

HT45F24A

注：EEC、PD、PDC 和 PDPU 寄存器地址仅在 Bank 1 中。

特殊数据存储器

间接寻址寄存器 – IAR0，IAR1

间接寻址寄存器 IAR0 和 IAR1 的地址虽位于数据存储区，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 和 IAR1）上的任何动作，将对间接寻址指针 (MP0 和 MP1) 所指定的存储器地址产生对应的读 / 写操作。它们总是成对出现，可以共同访问数据存储器。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。

间接寻址指针 – MP0，MP1

单片机提供两个间接寻址指针，即 MP0 和 MP1。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间接寻址指针所指定的地址。MP0，IAR0 用于访问 Bank 0，而 MP1 和 IAR1 可通过 BP 寄存器访问所有的 Bank。只有 Bank 0 可使用直接寻址，其它所有 Bank 必须使用 MP1 和 IAR1 进行间接寻址。

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

间接寻址程序举例

```
data .section ´data´
adres1      db ?
adres2      db ?
adres3      db ?
adres4      db ?
block       db ?
code .section at 0 ´code´
org 00h
start:
    mov a,04h              ; 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:
```

在上面的例子中有一点值得注意，即并没有确定 RAM 地址。

存储区指针 – BP

数据存储器被分为两个部分，即 Bank 0 和 Bank 1。可以通过设置存储区指针（Bank Pointer）值来访问不同的数据存储区。只有 Bank 0 可使用直接寻址，Bank1 必须使用 MP1 和 IAR1 进行间接寻址。MP0，IAR0 用于访问 Bank 0，而 MP1 和 IAR1 可通过 BP 寄存器访问所有的 Bank。

复位后，数据存储器会初始化到 Bank 0，但是在空闲 / 休眠模式下的 WDT 溢出复位，不会改变通用数据存储器的存储区号。应该注意的是特殊功能数据存储器不受存储区的影响，也就是说，不论是在哪一个存储区，都能对特殊功能寄存器进行读写操作。数据存储器的直接寻址总是访问 Bank 0，不影响存储区指针的值。要访问 Bank 1，则必须要使用间接寻址方式。

BP 寄存器

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

Bit 7~1　未使用，读为“0”

Bit 0　**DMBP0**：数据存储区选择位
0：Bank 0
1：Bank 1

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

除了 PDF 和 TO 标志外，状态寄存器中的位像其它大部分寄存器一样可以被改变。任何数据写入到状态寄存器将不会改变 TO 或 PDF 标志位。另外，执行不同的指令后，与状态寄存器有关的运算可能会得到不同的结果。TO 标志位只会受系统上电、看门狗溢出或执行“CLR WDT”或“HALT”指令影响。PDF 标志位只会受执行“HALT”或“CLR WDT”指令或系统上电影响。

Z、OV、AC 和 C 标志位通常反映最近运算的状态。

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。注意，STATUS 寄存器中 bit 0~3 是读 / 写位 .

STATUS 寄存器

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

“×”为未知

Bit 7~6 未使用，读为“0”

Bit 5 **TO**：看门狗溢出标志位
0：系统上电或执行“CLR WDT”或“HALT”指令后
1：看门狗溢出发生

Bit 4 **PDF**：暂停标志位
0：系统上电或执行“CLR WDT”指令后
1：执行“HALT”指令

Bit 3 **OV**：溢出标志位
0：无溢出
1：运算结果高两位的进位状态异或结果为 1

Bit 2 **Z**：零标志位
0：算术或逻辑运算结果不为 0
1：算术或逻辑运算结果为 0

Bit 1 **AC**：辅助进位标志位
0：无辅助进位
1：在加法运算中低四位产生了向高四位进位，或减法运算中低四位不发生从高四位借位

Bit 0 **C**：进位标志位
0：无进位
1：如果在加法运算中结果产生了进位，或在减法运算中结果不发生借位
C 也受循环移位指令的影响。

EEPROM 数据存储器

单片机的一个特性是内建 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。

EEPROM 数据存储器结构

EEPROM 数据存储器容量为 64×8。由于映射方式与程序存储器和数据存储器不同，因此不能像其它类型的存储器一样寻址。使用 Bank 0 中的一个地址寄存器和一个数据寄存器以及 Bank 1 中的一个控制寄存器，可以实现对 EEPROM 的单字节读写操作。

EEPROM 寄存器

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

寄存器名称	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EEA	—	—	D5	D4	D3	D2	D1	D0
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	—	—	D5	D4	D3	D2	D1	D0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	×	×	×	×	×	×

“×”为未知

Bit 7~6　未定义，读为“0”

Bit 5~0　数据 EEPROM 地址
数据 EEPROM 地址 Bit 5~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	×	×	×	×	×	×	×	×

“×”为未知

Bit 7~0 数据 EEPROM 数据
数据 EEPROM 数据 Bit 7~Bit 0

EEC 寄存器

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

Bit 7~4 未定义，读为“0”

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

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

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

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

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。写数据至 EEPROM，EEC 寄存器中的写使能位 WREN 先置为高以使能写功能。在此之后，EEC 寄存器中 WR 位必须被立刻置为高，一个写周期将开始。这两条指令必须连续执行。执行任何写操作以前总中断控制位 EMI 首先被清零，当写周期开始以后又被置为高。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 中断以侦测写周期是否完成。若写周期完成，WR 位将自动清除为“0”，通知用户数据已写入 EEPROM。因此，应用程序将轮询 WR 位以确定写周期是否结束。

写保护

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

EEPROM 中断

EEPROM 写或读周期结束后将产生 EEPROM 写或读中断，需先通过设置相关中断寄存器的 EE2I 位使能 EEPROM 中断。然而，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，E2F 请求标志位及其相关多功能中断请求标志位将被置位。若 EEPROM 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。更多细节将在中断章节讲述。

编程注意事项

必须注意的是数据不会无意写入 EEPROM。在没有写动作时写使能位被正常清零可以增强保护功能。BP 指针也可以正常清零以阻止进入 EEPROM 控制寄存器存在的 Bank 1。尽管没有必要，写一个简单的读回程序以检查新写入的数据是否正确还是应该考虑的。WREN 位置位后，写数据 WR 位需立即置位，来确保写周期正确执行。总中断 EMI 应当在写周期执行前被清零，并且在写周期执行后又被再次使能。注意，只有当 EEPROM 读或写操作全部完成单片机才能进入空闲或休眠模式。否则 EEPROM 读或写操作将不能被执行。

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 MP1
MOV MP1, A                       ; MP1 points to EEC register
MOV A, 01H                       ; setup Bank Pointer
MOV BP, A
SET IAR1.1                       ; set RDEN bit, enable read operations
SET IAR1.0                       ; start Read Cycle - set RD bit
BACK:
SZ  IAR1.0                       ; check for read cycle end
JMP BACK
CLR IAR1                         ; disable EEPROM read/write
CLR BP
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 MP1
MOV MP1, A                       ; MP1 points to EEC register
MOV A, 01H                       ; setup Bank Pointer
MOV BP, 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 read/write
CLR BP
```

振荡器

不同的振荡器选择可以让使用者在不同的应用需求中实现更大范围的功能。振荡器的灵活性使得在速度和功耗方面可以达到最优化。振荡器选择是通过配置选项和寄存器共同完成的。

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功能的时钟源。外部振荡器需要一些外围器件，而集成的两个内部振荡器不需要任何外围器件。它们提供的高速和低速系统振荡器具有较宽的频率范围。所有振荡器选择通过配置选项选择。较高频率的振荡器提供更高的性能，但要求有更高的功率，反之亦然。动态切换快慢系统时钟的能力使单片机具有灵活而优化的性能 / 功耗比，此特性对功耗敏感的应用领域尤为重要。

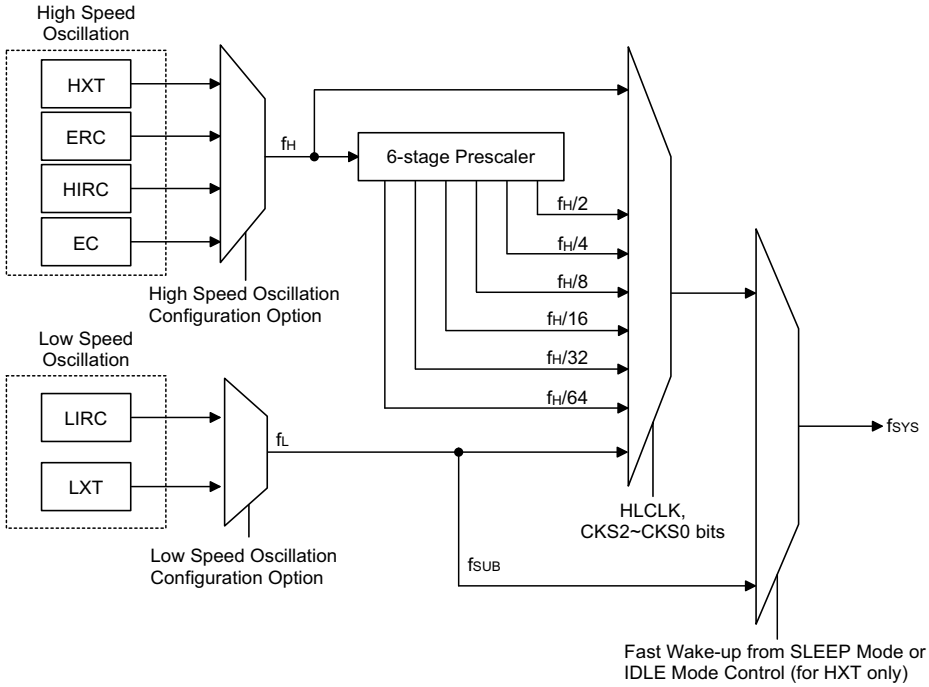
类型	名称	频率	引脚
外部晶振	HXT	400kHz~8MHz	OSC1/OSC2
外部 RC	ERC	4MHz	OSC1
内部高速 RC	HIRC	910kHz，2/4/8MHz	—
外部时钟	EC	400kHz~8MHz	OSC1
外部低速晶振	LXT	32.768kHz	XT1/ XT2
内部低速 RC	LIRC	32kHz	—

振荡器类型

系统时钟配置

单片机有六个系统振荡器，包括四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，外部 RC 振荡器，外部时钟和内部 910kHz、2MHz、4MHz 或 8MHz RC 振荡器。两个低速振荡器包括外部 32.768kHz 振荡器和内部 32kHz 振荡器。

使用高速或低速振荡器作为系统时钟的选择是通过设置 SMOD 寄存器中的 HLCLK 位及 CKS2~CKS0 位决定的，系统时钟可动态选择。高速或低速振荡器的实际时钟源经由配置选项选择。低速或高速系统时钟频率由 SMOD 寄存器的 HLCLK 位及 CKS2~CKS0 位决定的。请注意，两个振荡器必须做出选择，即一个高速和一个低速振荡器。

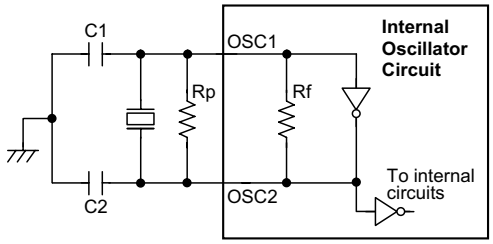


系统时钟配置

外部晶体 / 陶瓷振荡器 -- HXT

外部高频晶体 / 陶瓷振荡器可通过配置选项选择。对于晶体振荡器，只要简单地将晶体连接至 OSC1 和 OSC2，则会产生振荡所需的相移及反馈，而不需其它外部器件。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 -- HXT

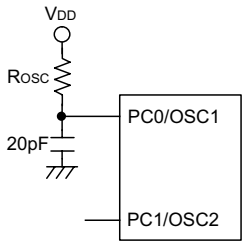
晶体振荡器 C1 和 C2 值		
晶体频率	C1	C2
8 MHz	0pF	0pF
4 MHz	0pF	0pF
1 MHz	100pF	100pF
注：C1 和 C2 数值仅作参考用		

晶体振荡器电容推荐值

外部 RC 振荡器 -- ERC

ERC 振荡器只需要在 OSC1 和 VDD 之间连接一个阻值约在 56kΩ 到 2.4MΩ 之间的电阻，OSC1 与 VSS 之间连接一个电容。系统频率由外部所接电阻的大小决定，外部电容并不会影响振荡器的频率值，在这里只起到稳定的作用。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 VDD、温度以及芯片制成工艺不同的影响减至最低程度。这里，提供一个电阻 / 频率的参考：使用外部 150kΩ 电阻连接到 5V 电源电压，在 25°C 下，振荡器的频率为 4MHz，容差 2%。外部 RC 振荡器仅使用 OSC1 引脚，OSC1 与 PC1 引脚共用，此时 PC1 引脚可以作为普通的 I/O 口使用。

为了确保振荡器的稳定性及减少噪声和串扰的影响 , 需将电容及电阻尽可能的接近单片机。



外部 RC 振荡器 -- ERC

外部振荡器 -- EC

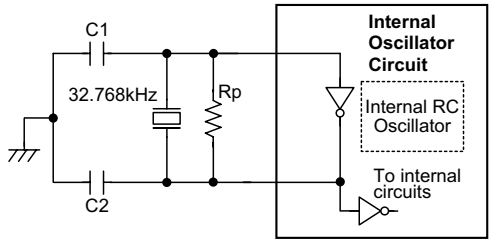
单片机的系统时钟也可来源于外部时钟，这样就给用户提供了一种将外部硬件和单片机同步运行的选择。通过配置选项选择，将外部时钟源接至 OSC1 引脚即可。如果使用的是外部振荡器提供的时钟源，则 OSC2 引脚应处于浮空状态。内部振荡电路中有一个滤波电路，它用于减少振荡器引脚的噪声引起单片机误动作的可能性，然而这个滤波电路有一定的功耗，可以通过振荡器配置选项来关闭这个滤波器。在功耗要求严格，高集成度的外部时钟源和外部低阻抗时钟源的应用中，不使用内部滤波器。

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 振荡器具有四种固定的频率：910kHz，2MHz，4MHz 和 8MHz。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡频率因 V_{DD}、温度以及芯片制成工艺不同的影响减至最低程度。在电源电压为 3V 或 5V 及温度为 25°C 的条件下，910kHz，2MHz，4MHz，8MHz 这四个固定频率的容差为 2%。如果选择了该内部时钟，无需额外的引脚；PC0 和 PC1 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，经由配置选项选择。时钟频率固定为 32.768kHz，此时 XT1 和 XT2 间引脚必须连接 32.768kHz 的晶体振荡器。需要外部电阻和电容连接到 32768Hz 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。在系统上电期间，LXT 振荡器启动需要一定的延时。



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

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。

一些配置选项决定是否 XT1/XT2 脚是用于 LXT 还是作为普通 I/O 口使用。

- 若 LXT 振荡器未被用于任何时钟源，XT1/XT2 脚能被用作一般 I/O 口使用。
- 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。

LXT 振荡器 C1 和 C2 值		
晶体频率	C1	C2
32.768kHz	10pF	10pF
注：1、C1 和 C2 数值仅作参考用 2、R_P 的建议值为 5M~10MΩ		

32.768kHz 振荡器电容推荐值

LXT 振荡器低功耗功能

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

LXTLP 位	LXT 模式
0	快速启动
1	低功耗

系统上电时会清零 LXTLP 位来快速启动 LXT 振荡器。在快速启动模式，LXT 振荡器将起振并快速稳定下来。LXT 振荡器完全起振后，可以通过设置 LXTLP 位为高进入低功耗模式。振荡器可以继续运行，其间耗电将少于快速启动模式。在功耗敏感的应用领域如电池应用方面，功耗必须限制为一个最小值。为了降低功耗，建议系统上电 2 秒后，在应用程序中将 LXTLP 位设为“1”。应注意的是，无论 LXTLP 位是什么值，LXT 振荡器会一直运作，不同的只是在低功耗模式时启动时间更长。

内部 32kHz 振荡器 -- LIRC

内部 32kHz 系统振荡器也是一个低频振荡器，经由配置选项选择。有一个完全集成 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减至最低。因此，内部 32kHz 振荡器频率在 25°C 温度 5V 电压下的精度保持在 10% 以内。

辅助振荡器

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

工作模式和系统时钟

现今的应用要求单片机具有较高的性能及尽可能低的功耗，这种矛盾的要求在便携式电池供电的应用领域尤为明显。高性能所需要的高速时钟将增加功耗，反之亦然。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最佳性能 / 功耗比。

系统时钟

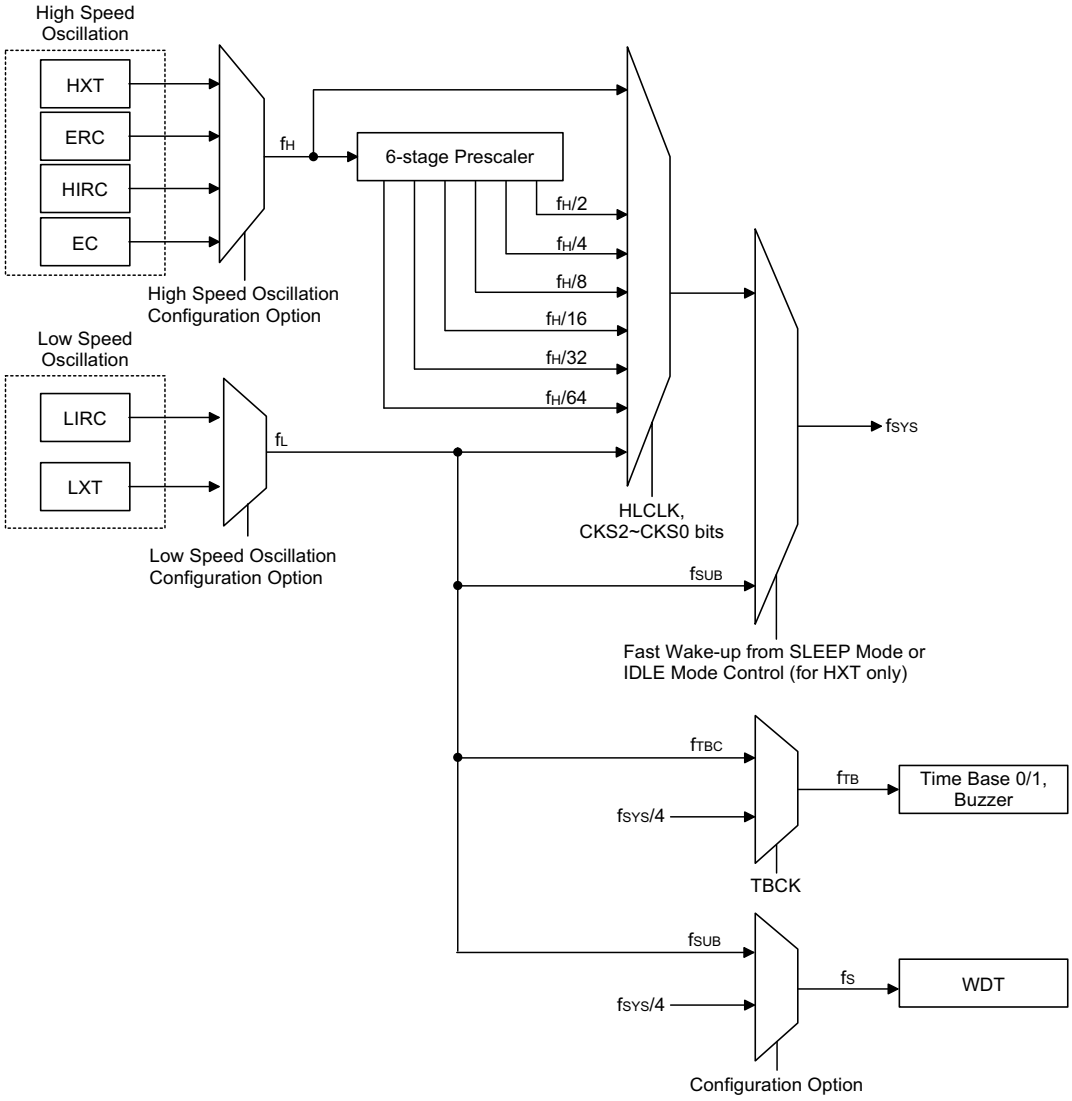
单片机为 CPU 和外围功能操作提供了多种不同的时钟源。用户使用配置选项和寄存器编程可获取多种时钟，进而使系统时钟获取最大的应用性能。

主系统时钟可来自高频时钟源 f_H 或低频时钟源 f_L，通过 SMOD 寄存器中的 HLCLK 位及 CKS2~CKS0 位进行选择。高频时钟来自 HXT、ERC、HIRC 振荡器或 EC，可通过配置选项选择，低频系统时钟源来自内部时钟 f_L，若 f_L 被选择，可通过配置选项设定为 LXT 或 LIRC 振荡器。其它系统时钟还有高速系统振荡器的分频 $f_H/2$~$f_H/64$。

另外两个内部时钟用于外围电路，次时钟源 f_{SUB} 和周期时钟 f_{TBC}。这两个时钟源来自 LXT 或 LIRC 振荡器，通过配置选项选择。快速唤醒发生后，f_{SUB} 为单片机提供一个次时钟，使系统能够在更短的时间内快速唤醒。f_{SUB} 和 $f_{SYS}/4$ 用于看门狗定时器的的时钟源。f_{TB} 用于时基 0/1 中断功能的时钟源。

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 0、休眠模式 1、空闲模式 0 和空闲模式 1 用于单片机 CPU 关闭时以节省耗电。



系统时钟选项

注：当系统时钟源 f_{SYS} 由 f_H 到 f_L 转换时，高速振荡器将停止以节省耗电。因此，没有为外围电路提供 $f_H/2$~$f_H/64$ 的频率。

工作模式	说明				
	CPU	f_{SYS}	f_{SUB}	f_S	f_{TBC}
正常模式	On	f_H~f_H/64	On	On	On
低速模式	On	f_L	On	On	On
空闲模式 0	Off	Off	On	On/Off	On
空闲模式 1	Off	On	On	On	On
休眠模式 0	Off	Off	Off	Off	Off
休眠模式 1	Off	Off	On	On	Off

正常模式

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

低速模式

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

休眠模式 0

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

休眠模式 1

在 HALT 指令执行后且 SMOD 寄存器中 IDLEN 位为低时，系统进入休眠模式。在休眠模式 1 中，CPU 停止运行。然而当其时钟源经配置选项选择为 f_{SUB} 时，若 LVDEN 位为“1”或看门狗定时器功能使能，f_{SUB} 及 f_S 继续运行。

空闲模式 0

执行 HALT 指令后且 SMOD 寄存器中 IDLEN 位为高，WDTC 寄存器中 FSYSON 位为低时，系统进入空闲模式 0。在空闲模式 0 中，CPU 停止，但一些外围功能如看门狗定时器、时基 0 和 SIM 将继续工作。在空闲模式 0 中，系统振荡器停止，看门狗定时器时钟 f_S 开启或关闭由 f_S 所选时钟源决定。若时钟源为 f_{SYS}/4，f_S 关闭；若时钟源为 f_{SUB}，f_S 开启。

空闲模式 1

执行 HALT 指令后且 SMOD 寄存器中 IDLEN 位为高，WDTC 寄存器中 FSYSON 位为高时，系统进入空闲模式 1。在空闲模式 1 中，CPU 停止，但会提供一个时钟源给一些外围功能如看门狗定时器、时基 0 和 SIM。在空闲模式 1 中，系统振荡器继续运行，该系统振荡器可以为高速或低速系统振荡器。在该模式中看门狗定时器时钟 f_S 开启。若时钟源为 f_{SYS}/4 或 f_{SUB}，f_S 开启。

控制寄存器

寄存器 SMOD 用于控制单片机内部时钟。

SMOD 寄存器

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

Bit 7~5 **CKS2~CKS0**：当 HLCLK 为“0”时系统时钟选择位
000：f_L（f_{LXT} 或 f_{LIRC}）
001：f_L（f_{LXT} 或 f_{LIRC}）
010：$f_H/64$
011：$f_H/32$
100：$f_H/16$
101：$f_H/8$
110：$f_H/4$
111：$f_H/2$
这三位用于选择系统时钟源。除了 LXT 或 LIRC 振荡器提供的系统时钟源外，也可使用高频振荡器的分频作为系统时钟。

Bit 4 **FSTEN**：快速唤醒控制位（仅用于 HXT）
0：除能
1：使能
此位为快速唤醒控制位，用于决定单片机被唤醒后 f_{SUB} 是否开始工作。若此位为高，当 f_{SUB} 有效时，此时钟源可作为暂用系统时钟以提供一个较快的唤醒时间。

Bit 3 **LTO**：低速振荡器就绪标志位
0：未就绪
1：就绪
此位为低速系统振荡器就绪标志位，用于表明低速系统振荡器在系统上电复位或经唤醒后何时稳定下来。当系统处于休眠 0 模式时，该标志为低。若系统时钟来自 LXT 振荡器，系统唤醒后该位转换为高需 1024 个时钟周期；若系统时钟来自 LIRC 振荡器，该位转换为高需 1~2 个时钟周期。

Bit 2 **HTO**：高速振荡器就绪标志位
0：未就绪
1：就绪
此位为高速系统振荡器就绪标志位，用于表明高速系统振荡器何时稳定下来。此标志在系统上电后经硬件清零，高速系统振荡器稳定后变为高电平。因此，此位在单片机上电后由应用程序读取的总为“1”。该标志由休眠模式或空闲模式 0 中唤醒后会处于低电平状态，若使用 HXT 振荡器，该位将在 1024 个时钟周期后变为高电平状态，若使用 ERC 或 HIRC 振荡器则只需 15~16 个时钟周期即可。

Bit 1 **IDLEN**：空闲模式控制位
0：除能
1：使能
此位为空闲模式控制位，用于决定 HALT 指令执行后发生的动作。若此位为高，当指令 HALT 执行后，单片机进入空闲模式。若 FSYSON 位为高，在空闲模式 1 中 CPU 停止运行，系统时钟将继续工作以保持外围功能继续工作；若 FSYSON 为低，在空闲模式 0 中 CPU 和系统时钟都将停止运行。若此位为低，单片机将在 HALT 指令执行后进入休眠模式。

Bit 0 **HLCLK**：系统时钟选择位
0：$f_H/2$~ $f_H/64$ 或 f_L
1：f_H
此位用于选择 f_H 或 $f_H/2$~ $f_H/64$ 还是 f_L 作为系统时钟。该位为高时选择 f_H 作为系统时钟，为低时则选择 $f_H/2$~ $f_H/64$ 或 f_L 作为系统时钟。当系统时钟由 f_H 时钟向 f_L 时钟转换时，f_H 将自动关闭以降低功耗。

快速唤醒

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

若 HXT 振荡器作为正常模式的系统时钟，且快速唤醒功能使能，系统唤醒将需 1~2 个 t_{SUB} 时钟周期。系统开始在 f_{SUB} 时钟源下运行直至 1024 个 HXT 时钟周期后 HTO 标志转换为高，系统将切换到 HXT 振荡器运行。

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

系统振荡器	FSTEN 位	唤醒时间（休眠模式 0）	唤醒时间 (休眠模式 1)	唤醒时间 (空闲模式 0)	唤醒时间（空闲模式 1）
HXT	0	1024 个 HXT 周期	1024 个 HXT 周期		1~2 个 HXT 周期
	1	1024 个 HXT 周期	1~2 个 f_{SUB} 周期（系统在 f_{SUB} 下运行 1024 个 HXT 周期后切换到 HXT 振荡器运行）		1~2 个 HXT 周期
ERC	×	15~16 个 ERC 周期	15~16 个 ERC 周期		1~2 个 ERC 周期
EC	×	15~16 个 EC 周期	15~16 个 EC 周期		1~2 个 EC 周期
HIRC	×	15~16 个 HIRC 周期	15~16 个 HIRC 周期		1~2 个 HIRC 周期
LIRC	×	1~2 个 LIRC 周期	1~2 个 LIRC 周期		1~2 个 LIRC 周期
LXT	×	1024 个 LXT 周期	1024 个 LXT 周期		1~2 个 LXT 周期

唤醒时间

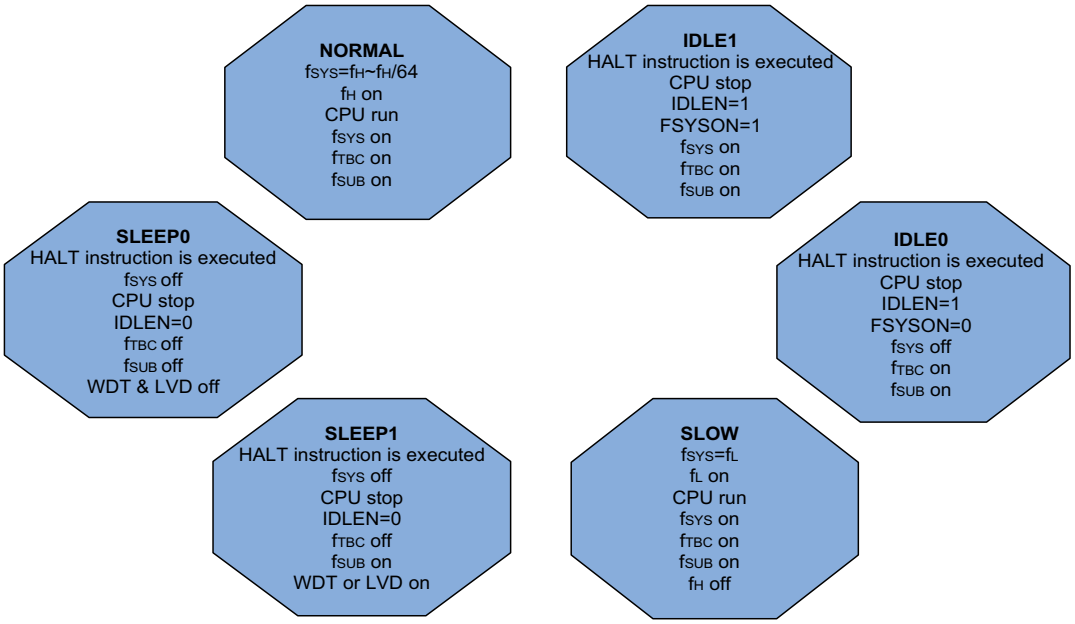
注：若看门狗定时器除能，意味着 LXT 和 LIRC 都关闭，当单片机由休眠模式 0 中唤醒时快速唤醒功能不可用。

工作模式切换和唤醒

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最佳的性能 / 功耗比。用此方式，对单片机工作的性能要求不高的情况下，可使用较低频时钟以减少工作电流，在便携式应用上延长电池的使用寿命。

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

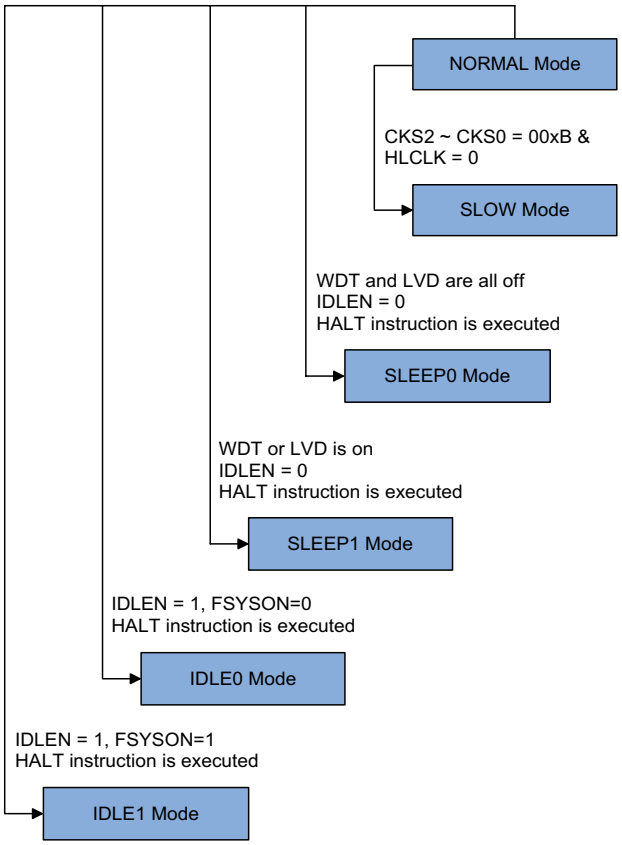
当 HLCLK 位变为低电平时，时钟源将由高速时钟源 f_H 转换成时钟源 $f_H/2$~$f_H/64$ 或 f_L。若时钟源来自 f_L，高速时钟源将停止运行以节省耗电。此时须注意，$f_H/16$ 和 $f_H/64$ 内部时钟源也将停止运行，由此会影响到如 SIM 等内部功能的工作。所附流程图显示了单片机在不同工作模式间切换时的变化。



正常模式切换到低速模式

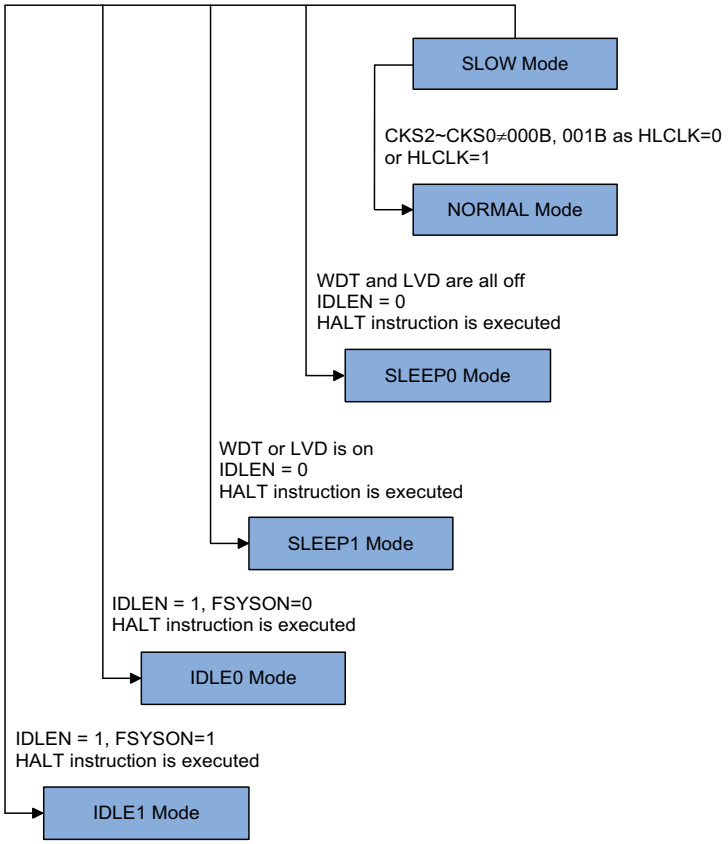
系统运行在正常模式时使用高速系统振荡器，因此较为耗电。可通过设置 SMOD 寄存器中的 HLCLK 位为“0”及 CKS2~CKS0 位为“000”或“001”使系统时钟切换至运行在低速模式下。此时将使用低速系统振荡器以节省耗电。用户可在对性能要求不高的操作中使用此方法以减少耗电。

低速模式的时钟源来自 LXT 或 LIRC 振荡器，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。该动作由 SMOD 寄存器中 LTO 位控制。



低速模式切换到正常模式

在低速模式系统使用 LXT 或 LIRC 低速振荡器。切换到使用高速系统时钟振荡器的正常模式需设置 HLCLK 位为“1”，也可设置 HLCLK 位为“0”但 CKS2~CKS0 需设为“010”、“011”、“100”、“101”、“110”或“111”。高频时钟需要一定的稳定时间，通过检测 HTO 位的状态可进行判断。高速振荡器的稳定时间由所使用高速系统振荡器的类型决定。



进入休眠模式 0

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

- 系统时钟、WDT 时钟和时基时钟停止运行，应用程序停止在“HALT”指令处。
- 数据存储器中的内容和寄存器将保持当前值。
- 无论 WDT 时钟源来自 f_{SUB} 时钟或系统时钟，WDT 都将被清除并停止运行。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

进入休眠模式 1

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

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

进入空闲模式 0

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

- 系统时钟停止运行，应用程序停止在“HALT”指令处，时基时钟和 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。
- 若 WDT 使能且其时钟源来自 f_{SUB}，则 WDT 将被清零并重新开始计数；若其时钟源来自系统时钟，则 WDT 将停止运行。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

进入空闲模式 1

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

- 系统时钟、时基时钟和 f_{SUB} 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。
- 若 WDT 使能，无论 WDT 时钟源来自 f_{SUB} 或是系统时钟，则 WDT 将被清零并重新开始计数。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

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 除外），所以如果要将电路的电流降到最低，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，如果使能配置选项中的 LXT 或 LIRC 振荡器，会导致耗电增加。

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

唤醒

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：

- 外部复位
- PA 口下降沿
- 系统中断
- WDT 溢出

若由外部 $\overline{\text{RES}}$ 引脚唤醒，系统会经过完全复位的过程；若由 WDT 溢出唤醒，则会发生看门狗定时器复位。这两种唤醒方式都会使系统复位，可以通过状态寄存器中 TO 和 PDF 位来判断它的唤醒源。系统上电或执行清除看门狗的指令，会清零 PDF；执行 HALT 指令，PDF 将被置位。看门狗计数器溢出将会置位 TO 标志并唤醒系统，这种复位会重置程序计数器和堆栈指针，其它标志保持原有状态。

PA 口中的每个引脚都可以通过 PAWU 寄存器使能下降沿唤醒功能。PA 端口唤醒后，程序将在“HALT”指令后继续执行。如果系统是通过中断唤醒，则有两种可能发生。第一种情况是：相关中断除能或是中断使能且堆栈已满，则程序会在“HALT”指令之后继续执行。这种情况下，唤醒系统的中断会等到相关中断使能或有堆栈层可以使用之后才执行。第二种情况是：相关中断使能且堆栈未满，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。

编程注意事项

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

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

看门狗定时器

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

看门狗定时器时钟源

WDT 定时器时钟源来自于内部时钟 f_S，而 f_S 的时钟源又是通过配置选项从 f_{SUB} 和 $f_{SYS}/4$ 中选择。f_{SUB} 时钟由 LXT 或 LIRC 振荡器提供，可通过配置选项设置。看门狗定时器的时钟源可分频为 2^{13}~2^{20} 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。电压为 5V 时内部振荡器 LIRC 的周期大约为 32kHz。

需要注意的是，这个特殊的内部时钟周期随 V_{DD}、温度和制成的不同而变化。LXT 振荡器由一个外部 32.768kHz 晶振提供。另一个看门狗定时器时钟源选项为 $f_{SYS}/4$。看门狗定时器时钟源可来自内部 LIRC 振荡器、LXT 振荡器或 $f_{SYS}/4$。

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。寄存器结合配置选项控制看门狗定时器的工作。

WDTC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	WS2	WS1	WS0	WDTEN3	WDTEN2	WDTEN1	WDTEN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	1	1	1	0	1	0

Bit 7 **FSYSON**：f_{SYS} 在空闲模式下的控制位
0：除能
1：使能

Bit 6~4 **WS2，WS1，WS0**：WDT 溢出周期选择位
000：$2^{13}/f_S$
001：$2^{14}/f_S$
010：$2^{15}/f_S$
011：$2^{16}/f_S$
100：$2^{17}/f_S$
101：$2^{18}/f_S$
110：$2^{19}/f_S$
111：$2^{20}/f_S$
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。

Bit 3~0 **WDTEN3，WDTEN2，WDTEN1，WDTEN0**：WDT 软件控制位
1010：除能
其它：使能

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。通过配置选项选择看门狗定时器的一些选项，如使能 / 除能、时钟源选择及清除指令类型。除了配置选项使能 / 除能看门狗定时器外，WDTC 寄存器中的 WDTEN3~WDTEN0 位也可用来除能看门狗定时器，此时需设置 WDTEN3~WDTEN0 为“1010”。若使用看门狗定时器功能，推荐设置这四位为“0101”，提供最大可能的防干扰能力。注意，若看门狗定时器被除能，相关操作的任何指令都不会工作。

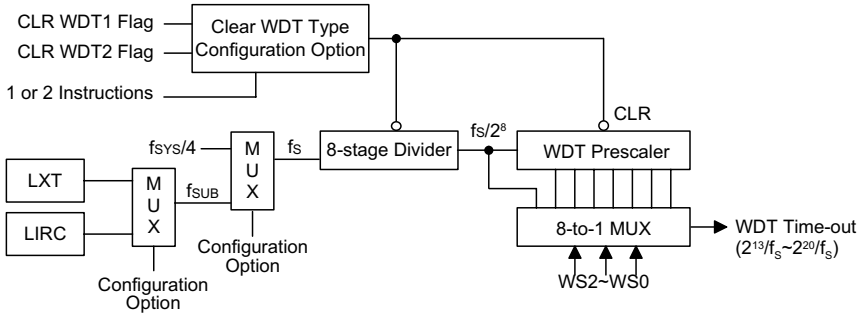
WDT 配置选项	WDTEN3~WDTEN0 位	WDT 功能
WDT 使能	××××	使能
WDT 除能	除 1010 外其它值	使能
WDT 除能	1010	除能

看门狗定时器使能 / 除能控制

程序正常运行时，WDT 溢出将导致芯片复位，并置位状态标志位 TO。若系统处于休眠或空闲模式，当 WDT 发生溢出时，状态寄存器中的 TO，程序计数器 PC 和堆栈指针 SP 将被置位。有三种方法可以用来清除 WDT 的内容。第一种是外部硬件复位 (RES 引脚低电平)，第二种是通过软件清除指令，而第三种是通过“HALT”指令。

软件指令有两种用于清除看门狗寄存器，需通过配置选项选择。第一种选择是使用一条“CLR WDT”指令，而第二种是使用“CLR WDT1”和“CLR WDT2”两个指令。对于第一种选择，只要执行“CLR WDT”便清除 WDT。而第二种选择，需要交替执行“CLR WDT1”和“CLR WDT2”两者才能成功的清除 WDT。关于第二种选择，如果“CLR WDT1”正被使用来清除 WDT，接着再执行这条指令将是无效的，只有执行“CLR WDT2”指令才能清除 WDT。同样的“CLR WDT2”指令已经执行后，只有接着执行“CLR WDT1”指令才可以清除看门狗定时器。

当设置分频比为 2^{20} 时，溢出周期最大。例如，时钟源为 32.768kHz LXT 振荡器，分频比为 2^{20} 时最大溢出周期约 32s，分频比为 2^{13} 时最小溢出周期约 250ms。如果 $f_{SYS}/4$ 作为看门狗定时器时钟源，需要注意，当系统工作在休眠或空闲模式 0 时，系统时钟停止工作，看门狗失去保护作用。如果系统工作在干扰大的环境中，强烈建议使用 f_{SUB} 作为时钟源。



复位和初始化

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

除上电复位以外，即使单片机处于正常工作状态，有些情况的发生也会迫使单片机复位。譬如当单片机上电后已经开始执行程序，$\overline{\text{RES}}$ 脚被强制拉为低电平。这种复位为正常操作复位，单片机中只有一些寄存器受影响，而大部分寄存器不会改变，在复位引脚恢复至高电平后，单片机可以正常运行。

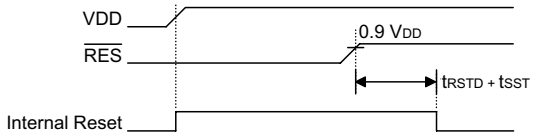
另一种复位为看门狗溢出单片机复位。不同方式的复位操作会对寄存器产生不同的影响。另一种复位为低电压复位即 LVR 复位，在电源供应电压低于 LVR 设定值时，系统会产生 LVR 复位，这种复位与与 $\overline{\text{RES}}$ 脚拉低复位方式相似。

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 100ms

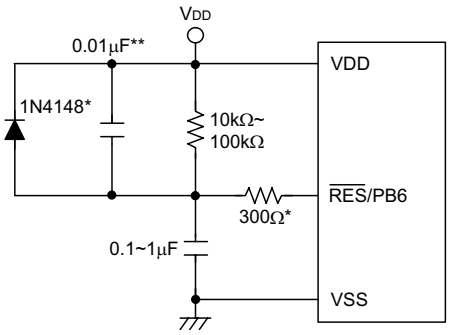
上电复位时序图

$\overline{\text{RES}}$ 引脚复位

由于复位引脚与 PB.6 共用，复位功能必须使用配置选项选择。虽然单片机有一个内部 RC 复位功能，如果电源上升缓慢或上电时电源不稳定，内部 RC 振荡可能导致芯片复位不良，所以推荐使用和 $\overline{\text{RES}}$ 引脚连接的外部 RC 电路，由 RC 电路所造成的时间延迟使得 $\overline{\text{RES}}$ 引脚在电源供应稳定前的一段延长周期内保持在低电平。在这段时间内，单片机的正常操作是被禁止的。$\overline{\text{RES}}$ 引脚达到一定电压值后，再经过延迟时间 t_{RSTD} 单片机可以开始进行正常操作。下图中 SST 是系统延迟周期 System Start-up Timer 的缩写。

在许多应用场合，可以在 VDD 和 $\overline{\text{RES}}$ 之间接入一个电阻，在 VSS 与 $\overline{\text{RES}}$ 之间接入一个电容作为外部复位电路。与 $\overline{\text{RES}}$ 脚上所有相连接的线段必须尽量短以减少噪声干扰。

当系统在较强干扰的场合工作时，建议使用增强型的复位电路，如下图所示。

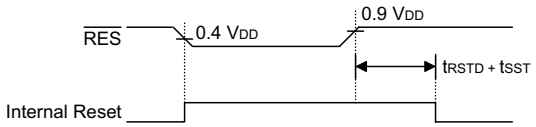


注：“*”表示建议加上此元件以加强静电保护。
“**”表示建议在电源有较强干扰场合加上此元件。

外部 RES 电路

欲知有关外部复位电路的更多信息可参考 HOLTEK 网站上的应用范例 HA0075S。

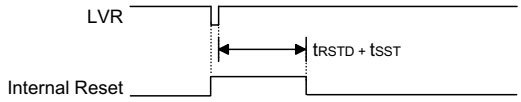
RES 引脚通过外部硬件强迫拉至低电平时，此种复位形式即会发生。这种复位方式和其它的复位方式一样，程序计数器会被清除为零且程序从头开始执行。



RES 复位时序图

低电压复位 -- LVR

单片机具有低电压复位电路，用来监测它的电源电压，可通过配置选项进行选择。例如在更换电池的情况下，单片机供应的电压可能会落在 0.9V~V_{LVR} 的范围内，这时 LVR 将会自动复位单片机。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，必须超过交流电气特性中 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} 参数值可通过配置选项进行设定。

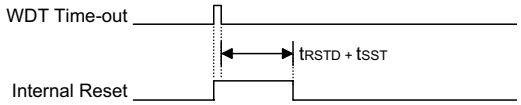


注：t_{RSTD} 为上电延迟时间，典型值为 100ms。

低电压复位时序图

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 RES 复位相同。

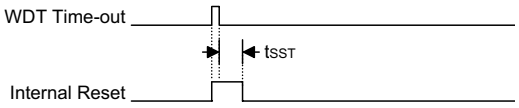


注：t_{RSTD} 为上电延迟时间，典型值为 100ms。

正常运行时看门狗溢出时序图

休眠或空闲时看门狗溢出复位

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



注：如果系统时钟源为 ERC 或 HIRC 时，t_{SST} 为 15~16 个时钟周期；
如果系统时钟源为 HXT 或 LXT，则 t_{SST} 为 1024 个时钟周期；
如果系统时钟源为 LIRC，则 t_{SST} 为 1~2 个时钟周期。

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 $\overline{\text{RES}}$ 复位或 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

注：“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	WDT 清除并重新计数
定时 / 计数器	所有定时 / 计数器停止
输入 / 输出口	I/O 口设为输入模式
堆栈指针	堆栈指针指向堆栈顶端

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

Register	HT45F23A	HT45F24A	Power-on Reset	$\overline{\text{RES}}$ or LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (Idle/Sleep)
PCL	●	●	0000 0000	0000 0000	0000 0000	0000 0000
MP0	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
BP	●	●	- - - - - - - 0	- - - - - - - 0	- - - - - - - 0	- - - - - - - u
ACC	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLP	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	●		- xxx xxxx	- uuu uuuu	- uuu uuuu	- uuu uuuu
TBLH		●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	●		- - - - - xxx	- - - - - uuu	- - - - - uuu	- - - - - uuu
TBHP		●	- - - - xxxx	- - - - uuuu	- - - - uuuu	- - - - uuuu
STATUS	●	●	- - 00 xxxx	- - uu uuuu	- - 1u uuuu	- - 11 uuuu
SMOD	●	●	0000 0011	0000 0011	0000 0011	uuuu uuuu
LVDC	●	●	- - 00 - 000	- - 00 - 000	- - 00 - 000	- -uu - uuu
INTEDGE	●	●	- - - - 0000	- - - - 0000	- - - - 0000	- - - - uuuu
WDTC	●	●	0111 1010	0111 1010	0111 1010	uuuu uuuu
INTC0	●	●	- 000 0000	- 000 0000	- 000 0000	- uuu uuuu
INTC1	●	●	- 000 - 000	- 000 - 000	- 000 - 000	- uuu - uuu
MFIC0	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFIC1	●	●	- 000 - 000	- 000 - 000	- 000 - 000	- uuu - uuu
PA	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	●		- 111 1111	- 111 1111	- 111 1111	- uuu uuuu
PB		●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	●		- 111 1111	- 111 1111	- 111 1111	- uuu uuuu
PBC		●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PC	●		- 111 1111	- 111 1111	- 111 1111	- uuu uuuu
PC		●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCC	●		- 111 1111	- 111 1111	- 111 1111	- uuu uuuu
PCC		●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PD		●	- - - - - -11	- - - - - -11	- - - - - -11	- - - - - -uu
PDC		●	- - - - - -11	- - - - - -11	- - - - - -11	- - - - - -uu
PAWU	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPU	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBPU	●		- 000 0000	- 000 0000	- 000 0000	- uuu uuuu
PBPU		●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCPU	●		- 000 0000	- 000 0000	- 000 0000	- uuu uuuu
PCPU		●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDPU		●	- - - - - -00	- - - - - -00	- - - - - -00	- - - - - -uu
PWM0	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWM1	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu

Register	HT45F23A	HT45F24A	Power-on Reset	$\overline{\text{RES}}$ or LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (Idle/Sleep)
MISC	●	●	0000 - - 00	0000 - - 00	0000 - - 00	uuuu - - uu
ADPCR	●		- -00 0000	- -00 0000	- -00 0000	- -uu uuuu
ADPCR		●	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADRL	●	●	xxxx - - - -	xxxx - - - -	xxxx - - - -	uuuu - - - -
ADRH	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCR	●		01 - - -000	01 - - -000	01 - - -000	uu - - -uuu
ADCR		●	01 - - 0000	01 - - 0000	01 - - 0000	uu - - uuuu
ACSR	●	●	100 - - 000	100 - - 000	100 - - 000	uuu - - uuu
SIMC0	●	●	1110 000 -	1110 000 -	1110 000 -	uuuu 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
TMR0	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR0C	●	●	00- 0 1000	00- 0 1000	00- 0 1000	uu- u uuuu
TMR1L	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1H	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1C	●	●	0000 1- - -	0000 1- - -	0000 1- - -	uuuu u- - -
EEA	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
EED	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
EEC	●	●	- - - - 0000	- - - - 0000	- - - - 0000	- - - - uuuu
LCDC	●	●	- 000 0000	- 000 0000	- 000 0000	- uuu uuuu
LDOC	●	●	- - -0 0000	- - -0 0000	- - -0 0000	- - -u uuuu
DACTRL	●	●	000 - - - -0	000 - - - -0	000 - - - -0	uuu - - - -u
DAL	●	●	0000 - - - -	0000 - - - -	0000 - - - -	uuuu - - - -
DAH	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
CMP1C0	●	●	0001 0000	0001 0000	0001 0000	uuuu uuuu
CMP1C1	●	●	1- - - 0010	1- - - 0010	1- - - 0010	1- - - uuuu
CMP2C0	●	●	0001 0000	0001 0000	0001 0000	uuuu uuuu
CMP2C1	●	●	00 - - 0010	00 - - 0010	00 - - 0010	uu- - uuuu
OPA1C0	●	●	0- - - - - - -	0- - - - - - -	0- - - - - - -	u- - - - - - -
OPA1C1	●	●	0000 1100	0000 1100	0000 1100	uuuu uuuu
OPA2C0	●	●	0- - - - - - -	0- - - - - - -	0- - - - - - -	u- - - - - - -
OPA2C1	●	●	0000 1100	0000 1100	0000 1100	uuuu uuuu
OPA2C2	●	●	00 - - 0000	00 - - 0000	00 - - 0000	uu - - uuuu
TBC	●	●	0011 0111	0011 0111	0011 0111	uuuu uuuu
BPCTL	●	●	0000 0000	0000 0000	0000 0000	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

盛群单片机的输入 / 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。

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

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寄存器 PAPU~PDPU 来设置，它用一个 PMOS 晶体管来实现上拉电阻功能。

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。

输入 / 输出端口控制寄存器

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

PAWU, PAPU, PA, PAC, PBPU, PB, PBC, PCPU, PC, PCC, PD, PDC 寄存器

- **HT45F23A**

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

“—”：未定义，读为“0”

PAWUn：PA 唤醒功能控制位
0：除能
1：使能

PAPUn/PBPUn/PCPUn：上拉电阻控制位
0：除能
1：使能

PACn/PBCn/PCCn：输入 / 输出控制位
0：输出
1：输入

● **HT45F24A**

寄存器名称	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PBPU	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PCPU	PCPU7	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PC	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	PCC7	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PDPU	—	—	—	—	—	—	PDPU1	PDPU0
PD	—	—	—	—	—	—	PD1	PD0
PDC	—	—	—	—	—	—	PDC1	PDC0

“—”：未定义，读为“0”

PAWUn：PA 唤醒功能控制位
0：除能
1：使能

PAPUn/PBPUn/PCPUn/PDPUn：上拉电阻控制位
0：除能
1：使能

PACn/PBCn/PCCn/PDCn：输入 / 输出控制位
0：输出
1：输入

PB 口 NMOS 开漏极控制寄存器

PB 端口中 PB0~PB3 可通过 MISC 寄存器的 ODE0~ODE3 位设置为开漏极结构。

MISC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	ODE3	ODE2	ODE1	ODE0	—	—	PFDSEL	PFDEN
R/W	R/W	R/W	R/W	R/W	—	—	R/W	R/W
POR	0	0	0	0	—	—	0	0

Bit 7 **ODE3**：PB3 开漏极使能位
0：除能
1：使能

Bit 6 **ODE2**：PB2 开漏极使能位
0：除能
1：使能

Bit 5 **ODE1**：PB1 开漏极使能位
0：除能
1：使能

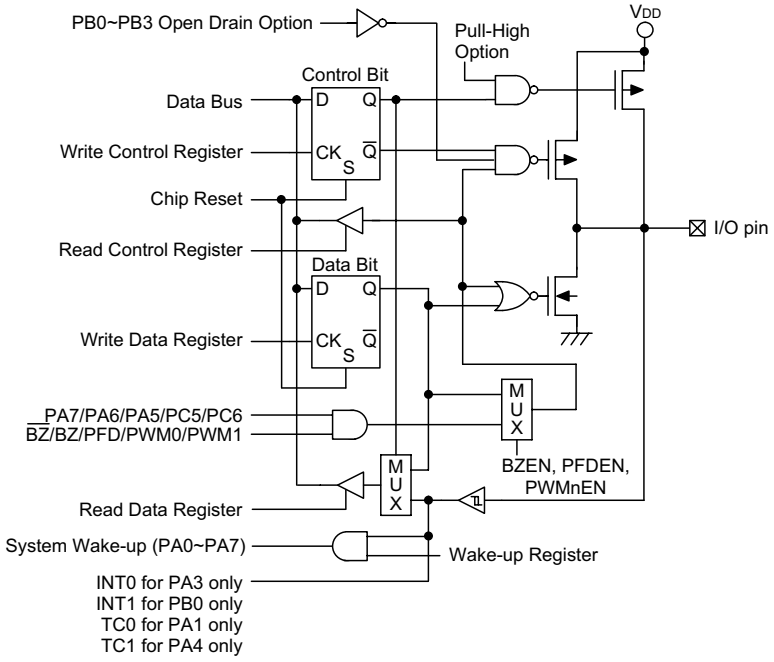
Bit 4 **ODE0**：PB0 开漏极使能位
0：除能
1：使能

Bit 3~2 未定义，读为“0”

Bit 1~0 **PFDSEL，PFDEN**：PFD 相关控制位，详细描述见其它章节

输入 / 输出引脚结构

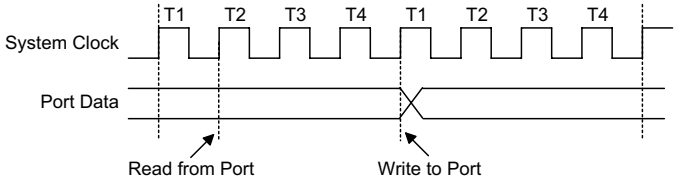
下图为输入 / 输出引脚的内部结构图。输入 / 输出引脚的准确逻辑结构图可能与此图不同，这里只是为了方便对 I/O 引脚功能的理解提供的一个参考。不允许引脚共用结构的所有类型被显示。



通用输入 / 输出端口

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~PDC，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~PD 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。



读写时序

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。此外，部分 PB 引脚结构也可为开漏极 I/O 形式，可通过特定的寄存器设置控制。

定时 / 计数器

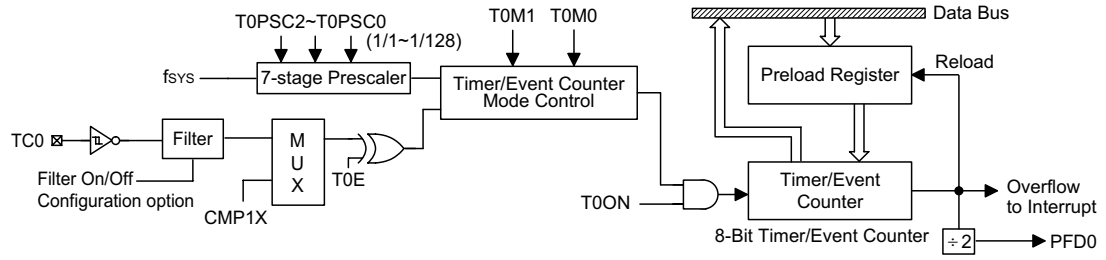
定时 / 计数器在任何单片机中都是一个很重要的部分，提供程序设计者一种实现和时间有关功能的方法。单片机提供一个 8 位和一个 16 位的向上定时 / 计数器。每个定时 / 计数器有四种不同的工作模式，可以被当作普通定时器、外部事件计数器、用于比较器的内部事件计数器或者脉冲宽度测量器使用。8-bit 定时器里的预分频器可扩大定时的范围。

有两类和定时 / 计数器相关的寄存器。第一类是存储实际的计数值，赋值给此寄存器可以设定初始计数值，读取此类寄存器可获得定时 / 计数器的内容。而第二类是定时 / 计数器的控制寄存器，此类寄存器设置定时 / 计数器的选项，控制定时 / 计数器的工作模式。定时 / 计数器的时钟源可配置设置来自内部系统时钟或外部引脚输入。

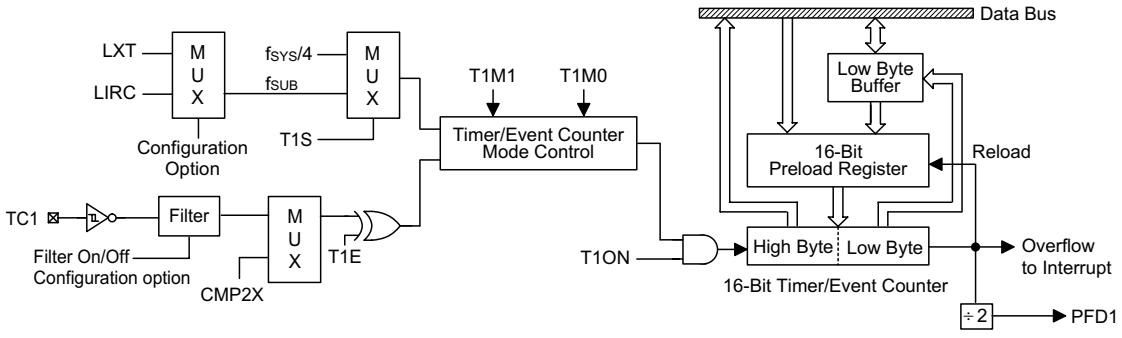
配置定时 / 计数器输入时钟源

内部定时 / 计数器的时钟源可以来自各种时钟源。当定时 / 计数器在定时器模式或者在脉冲宽度测量模式时，使用系统时钟作为计时来源。定时 / 计数器 0 的内部时钟为 f_{SYS}，其还可以通过预分频器进行分频，预分频值由定时 / 计数器控制寄存器 TMR0C 中的 T0PSC0~ T0PSC2 三位决定。定时 / 计数器 1 的内部时钟可通过内部时钟的配置选项及 TMR1C 寄存器的 T1S 位共同决定。

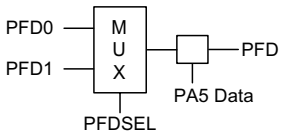
定时 / 计数器在事件计数器模式时使用外部时钟源，时钟源由外部定时 / 计数器引脚 TC0 或 TC1 提供。根据 T0E 或 T1E 位的状态，每次外部引脚由高电平到低电平或者由低电平到高电平跳变时，相应的计数器值加一。



8 位定时 / 计数器结构



16 位定时 / 计数器结构



注：1. PFD 的时钟源 PFD0 或 PFD1，分别来自 Timer0 或 Timer1，选择由 MISC 寄存器中的 PFDSEL 位决定。
2. PFD 的输出由 PA5 的状态决定。
3. CMP1X 是比较器 1 输出。
4. CMP2X 是比较器 2 输出。

定时 / 计数寄存器 – TMR0，TMR1L，TMR1H

定时 / 计数器是一种特殊功能寄存器，用于储存实际定时器值。8 位的定时 / 计数器为 TMR0，而 16 位的定时 / 计数器为 TMR1L 和 TMR1H。在收到一内部计时脉冲时或外部定时 / 计数器引脚状态发生跳变时，此寄存器的值将会加一。定时器将从预置寄存器所载入的值开始计数，直到计满 FFH（8 位定时 / 计数器）或 FFFFH（16 位定时 / 计数器），此时定时器溢出且会产生一个内部中断信号。定时器自动重新加载计数器初值并继续计数。

为了得到定时 / 计数器 FFH（8 位定时 / 计数器）或 FFFFH（16 位定时 / 计数器）的最大计数范围，预置寄存器必须先清除为“0”。注意的是，定时 / 计数器在 OFF 条件下，如果把数据写入预置寄存器，这数据将被立即写入实际的定时器。而如果定时 / 计数器已经 ON 且正在计数，在这个周期内写入到预置寄存器的任何新数据将保留在预置寄存器，只有在下一个溢出发生时才被写入实际定时器。

对于 16 位定时 / 计数器，它有低字节与高字节两个定时 / 计数寄存器，访问这些寄存器需要以指定方式进行。必须要注意的是当使用指令载入数据到低字节寄存器，即 TMR1L 时，数据只被载入到低字节缓冲器而不是直接送到低字节寄存器。当数据写入相应高字节寄存器，即 TMR1H 时，低字节缓冲器中的数据才真正被写入低字节寄存器。换句话说，写入数据到高字节定时 / 计数寄存器时，数据会被直接写入到高字节寄存器。同时在低字节缓冲器里的数据将被写入相应低字节寄存器。所以当写数据到 16 位定时 / 计数寄存器时，低字节数据应该先写入。另外要注意的是读取低字节寄存器的内容时，必须先读取高字节寄存器的内容，相应低字节寄存器中的内容就会载入低字节缓冲器中并被锁存。在此动作执行之后，低字节寄存器中的内容可使用一般的方式读取。请注意，读取定时 / 计数器低字节寄存器实际是读取先前锁存在低字节缓冲器中的内容，而非定时 / 计数器低字节寄存器的实际内容。

定时 / 计数控制寄存器 –TMR0C，TMR1C

HOLTEK 单片机有灵活的定时 / 计数器，定时 / 计数器能工作在四种不同的模式，至于选择工作在哪一种模式则是由控制寄存器的内容决定。

定时 / 计数器控制寄存器和其相应的定时 / 计数器寄存器控制定时 / 计数器的全部操作。在使用定时器之前，必须正确地设定定时 / 计数控制寄存器，以便保证定时器能正确操作，而这个过程通常在程序初始化期间完成。

为了确定定时器工作在哪一种模式，定时器模式、外部事件计数模式或脉冲宽度测量模式，定时 / 计数器中的 T0M1/T0M0 或 T1M1/T1M0 位必须设定到要求的逻辑电平。定时器打开位 T0ON 或 T1ON，即定时 / 计数控制寄存器的第 4 位，是定时器控制的开关，设定为逻辑高时，计数器开始计数，而清零时则停止计数。定时 / 计数器 0 控制寄存器的第 0~2 位决定其输入定时预分频器中的分频系数。如果使用外部计时源，预分频器的位将不起作用。如果定时器工作在事件计数或脉冲宽度测量模式，T0E 或 T1E 的逻辑电平，即控制寄存器的第 3 位，将可用来选择上升沿或下降沿触发。TMR1C 寄存器中的 T1S 位用于选择定时 / 计数器 1 的时钟源。

TMR0C 寄存器

Bit	7	6	5	4	3	2	1	0
Name	T0M1	T0M0	—	T0ON	T0E	T0PSC2	T0PSC1	T0PSC0
R/W	R/W	R/W	—	R/W	R/W	R/W	R/W	R/W
POR	0	0	—	0	1	0	0	0

Bit 7~6 **T0M1~T0M0**：选择定时 / 计数器 0 的工作模式
00：事件计数模式（来自比较器 1 的输出信号）
01：事件计数模式（来自 TC0 引脚的输入信号）
10：定时模式
11：脉冲宽度测量模式

Bit 5 未使用，读为“0”

Bit 4 **T0ON**：定时 / 计数器 0 使能 / 除能控制位
0：除能
1：使能

Bit 3 **T0E**：有效边沿选择位
事件计数模式有效边沿选择
0：在上升沿计数
1：在下降沿计数
脉冲宽度测量模式有效边沿选择
0：在下降沿计数，上升沿停止计数
1：在上升沿计数，下降沿停止计数

Bit 2~0 **T0PSC2~T0PSC0**：定时 / 计数器 0 预分频级数选择位
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：$f_{SYS}/128$

TMR1C 寄存器

Bit	7	6	5	4	3	2	1	0
Name	T1M1	T1M0	T1S	T1ON	T1E	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	1	—	—	—

Bit 7~6 **T1M1~T1M0**：选择定时 / 计数器 1 的工作模式
00：事件计数模式（来自比较器 2 的输出信号）
01：事件计数模式（来自 TC1 引脚的输入信号）
10：定时模式
11：脉冲宽度测量模式

Bit 5 **T1S**：定时 / 计数器 1 时钟源选择位
0：$f_{SYS}/4$
1：f_{SUB}，LXT 或 LIRC

Bit 4 **T1ON**：定时 / 计数器 1 使能 / 除能控制位
0：除能
1：使能

Bit 3 **T1E**：有效边沿选择位
事件计数模式有效边沿选择
0：在上升沿计数
1：在下降沿计数
脉冲宽度测量模式有效边沿选择
0：在下降沿计数，上升沿停止计数
1：在上升沿计数，下降沿停止计数

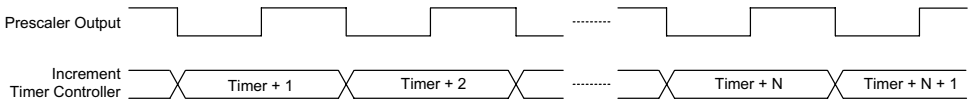
Bit 2~0 未使用，读为“0”

配置定时器模式

在这个模式，定时 / 计数器可以用来测量固定时间间距，当定时器发生溢出时，就会提供一个内部中断信号。要工作在这个模式，定时器工作模式选择位 TnM1/TnM0 必须设置正确的值。如下所示：

第 7 位	第 6 位
1	0

在这个模式下，8 位定时 / 计数器 0 的时钟源来自内部时钟 f_{SYS}，16 位定时 / 计数器 1 的时钟源来自 f_{SUB} 或 $f_{SYS}/4$。8 位定时 / 计数器 0 的输入计时频率是 f_{SYS} 除以定时器预分频器的值，这个值由定时器控制寄存器的 T0PSC2~T0PSC0 位决定。定时 / 计数器使能位 T0ON 或 T1ON 位必须被设为逻辑高，才能使定时器工作。每次内部时钟由高到低的电平跳变都会使定时器值增加一。当定时器已满即溢出时，会产生中断信号且定时器会重新载入已经载入到预置寄存器的值，然后继续向上计数。若定时器中断允许，将会产生一个中断信号。定时器中断可以通过清除 INTC0 寄存器中定时 / 计数器中断使能位禁止。



定时器模式时序图

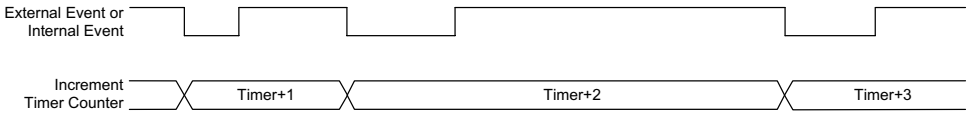
配置事件计数器模式

在这个模式下，发生在内部比较器输出和 TCn 引脚的逻辑事件改变的次数，可以通过内部计数器来记录。为使定时 / 计数器工作在内部事件计数器模式，控制寄存器中 TnM1/TnM0 位设置如下所示：

第 7 位	第 6 位
0	0/1

在这种模式下，定时 / 计数器的时钟源由比较器输出 CMP1X 或 CMP2X 提供或由 TCn 引脚提供，且该时钟源不能被内部预分频器分频。定时 / 计数器中的其它位设置好后，计数器打开位 TnON 必须设为逻辑高，使计数器开始计数。当 TnE 为逻辑低时，每次定时 / 计数器接收到由低到高的电平跳变将使计数器值加一。而当 TnE 为逻辑高时，每次定时 / 计数器接收到由高到低的电平跳变将使计数器值加一。当计数器计满时，计数器将溢出且自动从预置寄存器中加载初值并继续计数。若定时 / 计数器中断允许，将会产生一个中断信号。定时 / 计数器中断可以通过清除 INTC0 寄存器中定时 / 计数器中断使能位禁止。

值得注意的是，在事件计数模式下，即使系统进入暂停状态，定时 / 计数器仍可记录内部比较器输出逻辑事件的改变和外部输入引脚的变化。因此当定时 / 计数器溢出将会唤醒系统，若中断允许将会产生一个定时 / 计数器中断信号。



事件计数模式时序图（TnE=1）

配置脉冲宽度测量模式

在这个模式下，可以测量定时 / 计数器引脚上的外部脉冲宽度。在脉冲宽度测量模式中，TnM0 和 TnM1 位则必须都设为逻辑高。如下所示：

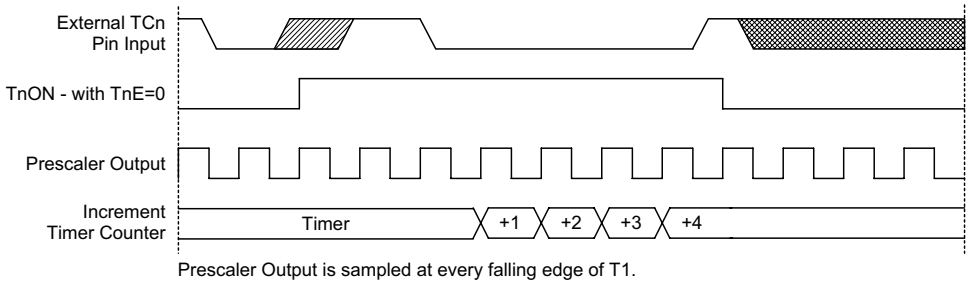
第 7 位	第 6 位
1	1

8 位定时 / 计数器 0 的时钟源由内部时钟 f_{SYS} 提供，而 16 位定时 / 计数器 1 的时钟源为 f_{SUB} 或 $f_{SYS}/4$。8 位定时 / 计数器 0 的时钟源可由预分频器进行分频，分频值则由控制寄存器 TMR0C 中的 T0PSC2~T0PSC0 位决定。定时 / 计数器中的其它位设置好后，计数器打开位 TnON 必须设为逻辑高，在外部 TCn 引脚接收到有效边沿触发信号后，计数器开始计数。

如果 TnE 位是逻辑低，当外部 TCn 引脚接收到一个由高到低的电平跳变时，定时 / 计数器将开始计数直到外部 TCn 引脚回到它原来的高电平。此时 TnON 位将自动清除为零，且定时 / 计数器停止计数。而如果 TnE 位是逻辑高，则当外部 TCn 引脚接收到一个由低到高的电平跳变时，定时 / 计数器开始计数直到外部 TCn 引脚回到原来的低电平。如上所述，TnON 位将自动清除为 0，且定时 / 计数器停止计数。注意，在脉冲宽度测量模式中，当外部定时器引脚上的外部控制信号回到它原来的电平时，TnON 位将自动地清除为 0。而在其它两种模式下，TnON 位只能在程序控制下才会被清除为 0。此时定时 / 计数器中的值可被程序读取，并由此得知外部定时 / 计数器引脚接收到的脉冲的长度。当 TnON 位被清除时，任何在外部定时 / 计数器引脚的电平跳变将被忽略。直到 TnON 位再次被程序设定为逻辑高，定时 / 计数器才开始脉冲宽度测量，用这种方法可完成单个脉冲的测量。

注意的是在这种模式下，定时 / 计数器是通过外部定时 / 计数器引脚上的逻辑跳变来控制，而不是通过逻辑电平。当定时 / 计数器计满产生溢出，定时 / 计数器将清零并载入预置寄存器的值。若定时器中断允许，将会产生一个内部中断信号。定时器中断可以通过清除 INTC0 寄存器中定时 / 计数器中断使能位禁止。

为了确保 TCn 工作在脉冲宽度测量模式，必须注意两点：首先是要将 TnM0 与 TnM1 位设定在脉冲宽度测量模式；其次是确定此引脚的输入 / 输出端口控制寄存器对应位被设定为输入状态。



脉冲宽度测量模式时序图（TnE=0）

可编程分频器 – PFD

可编程分频器可以产生可变的频率输出，适用于需要精确频率的应用场合。PFD 输出引脚与 I/O 引脚 PA5 共用。这个功能通过 MISC 寄存器中的 PFDEN 位控制，如果不选择该功能，则这个引脚就是作为普通的 I/O 引脚使用。

MISC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	ODE3	ODE2	ODE1	ODE0	—	—	PFDSEL	PFDEN
R/W	R/W	R/W	R/W	R/W	—	—	R/W	R/W
POR	0	0	0	0	—	—	0	0

Bit 7 **ODE3**：PB3 开漏极使能位
0：除能
1：使能

Bit 6 **ODE2**：PB2 开漏极使能位
0：除能
1：使能

Bit 5 **ODE1**：PB1 开漏极使能位
0：除能
1：使能

Bit 4 **ODE0**：PB0 开漏极使能位
0：除能
1：使能

Bit 3~2 未定义，读为“0”

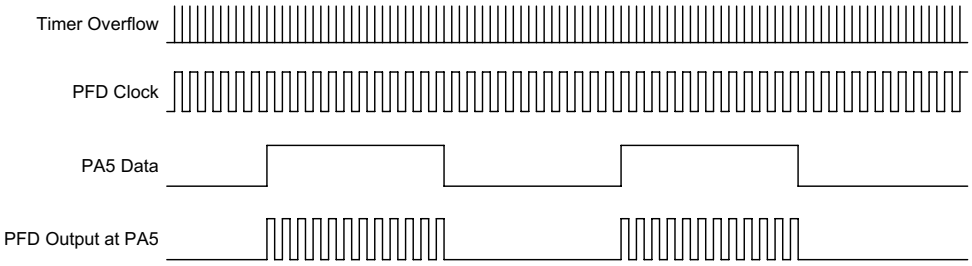
Bit 1 **PFDSEL**：PFD 时钟选择位
0：定时器 0 输出
1：定时器 1 输出

Bit 0 **PFDEN**：PFD 功能控制位
0：PFD 除能
1：PFD 使能

PFD 的时钟源可以是定时 / 计数器 0 或者是定时 / 计数器 1 中任意一个溢出信号，由 MISC 寄存器中的 PFDSEL 位决定。载入合适的值到定时寄存器和定时预分频器 (Prescaler) 中，可以产生需要的时钟输出频率。定时器会从预置寄存器中的预设值开始向上计数，直至溢出，此时单片机会产生一个溢出信号，导致 PFD 输出状态的改变。然后定时器会自动地重新载入预设值并继续向上计数。

为了让 PFD 正常运作，应记得要将端口控制寄存器 PAC 的第 5 位设置为输出模式。如果把它设置为输入，则 PFD 输出不工作，该引脚仍是作为普通的输入引脚使用。只有在 PA5 设置为“1”时，PFD 输出才开始有效。所以这个数据输出位就可作为 PFD 输出的开关控制位。应注意的是，若将 PA5 输出数据位设置为“0”，则 PFD 输出则为低。

假如系统时钟使用晶体振荡器，使用这种方法可以产生非常精确的频率值。



PFD 输出控制

预分频器

TMR0C 控制寄存器的 T0PSC0~T0PSC2 可以用来定义定时 / 计数器 0 中内部时钟源的预分频级数。定时 / 计数器溢出信号可用来驱动 PFD 或产生定时 / 计数器中断。

输入 / 输出接口

当工作在事件计数或脉冲宽度测量模式时，定时 / 计数器需要使用外部引脚以确保正确的动作。由于此端口为共用引脚，必须将其设置为定时 / 计数器的外部输入而不是普通的输入 / 输出，这需要设置定时 / 计数器控制寄存器内的相应的位为事件计数或脉宽测量模式。另外相应的端口控制寄存器位必须为高，以保证被设为输入引脚。即使定时 / 计数器使用了此引脚，其上拉仍有效。

定时 / 计数器引脚内部滤波器

外部定时 / 计数器引脚连接一个滤波器可降低由外部定时 / 计数器引脚上噪声或尖峰信号引起的计数出错和脉宽测量出错的可能性。内部滤波电路会产生一定的功耗，通过配置选项可关闭该滤波器，这在功耗要求严格的应用中十分有用，但其输入信号则为原始信号。值得注意的是滤波开 / 关的配置选项可套用到外部定时 / 计数器引脚和外部中断输入引脚，单独的定时 / 计数器引脚和外部中断输入引脚不具备滤波开 / 关功能。

编程注意事项

当定时 / 计数器运行在定时器模式时，定时器的时钟源是使用内部系统时钟，与单片机整体运行情况同步。在这个模式下，当定时寄存器溢出时，单片机将产生一个内部中断信号，使程序进入相应的内部中断。对于脉冲宽度测量模式，定时器的时钟源也是使用内部系统时钟，但定时器只有在正确的逻辑条件出现在定时器输入引脚时才执行动作。由于这个外部事件没有和内部定时器时钟同步，只有当下一个定时器时钟脉冲到达时，单片机才会检测到这个外部事件，因此在测量值上可能有小的差异，需要程序设计者在程序应用时加以注意。同样的情况发生在定时器配置为外部事件计数器模式时，它的时钟来源是外部事件，和内部系统时钟或者定时器时钟不同步。

当读取定时 / 计数器或写数据到预置寄存器时，计数时钟会被阻隔以避免错误发生，但这样做可能会导致计数错误，所以程序设计者应该考虑到这点。在第一次使用定时 / 计数器之前，要仔细确认有没有正确地设定初始值。中断控制寄存器中的定时器使能位必须正确的设置，否则相应定时 / 计数器内部中断仍然无效。定时 / 计数控制寄存器中的触发边沿选择位、定时 / 计数器工作模式和时钟源控制位也必须正确的设定，以确保定时 / 计数器按照应用需求而正确的配置。在定时 / 计数器打开之前，必须确保先载入定时 / 计数寄存器的初始值；这是因为在上电后，定时 / 计数寄存器中的初始值是未知的。定时 / 计数器初始化后，可以使用定时 / 计数控制寄存器中的使能位来打开或关闭定时器。注意，必须先正确地设定定时器模式，然后再去设置使能位为高打开定时器。写定时 / 计数控制寄存器，如果同时改变模式并设置使能位为高，可能会导致错误结果。

当发生定时 / 计数器溢出，相应的中断请求标志将置位。若中断允许，将会产生一个中断信号。不管中断是否允许，在 HALT 状态下，定时 / 计数器的溢出也会产生唤醒，这种情况可能发生在外部信号变化的计数模式中，定时 / 计数器向上计数直至溢出并唤醒系统。若在 HALT 模式下，不需要定时器中断唤醒系统，可以在进入 HALT 前将相应中断请求标志位置位。

定时 / 计数器应用范例

这个例子说明了如何设置定时 / 计数器的寄存器，如何设置和控制中断。另外还需注意的是，怎样通过寄存器的第 4 位来启动 / 停止定时 / 计数器。此应用范例设置定时 / 计数为定时模式，时钟来源于内部的系统时钟。

```
org 04h             ; external interrupt vector
org 0Ch             ; Timer/Event Counter 0 interrupt vector
jmp tmrint          ; jump here when the Timer/Event Counter 0 overflows
:
org 20h             ; main program internal Timer/Event Counter 0 interrupt
                    ; routine
tmrint:
:
                    ; Timer/Event Counter 0 main program placed here
:
reti
:
begin:
                    ; setup Timer 0 registers
mov a, 09bh         ; setup Timer 0 preload value
mov tmr0,a;
mov a, 081h         ; setup Timer 0 control register
mov tmr0c, a        ; timer mode and prescaler set to 1/2 setup interrupt
                    ; register
mov a, 009h         ; enable master interrupt and timer interrupt
mov intc0, a
set tmr0c.4         ; start Timer/Event Counter 0 - note mode bits must be
                    ; previously setup
```

脉冲宽度调制器

单片机提供 2 通道 8-bit 脉冲宽度调制（PWM）输出。这在马达速度控制等应用中十分有用，通过给相应的 PWM 寄存器设置特定的值，PWM 功能可提供占空比可调而频率固定的波形。

PWM 操作

在数据存储器中，单片机为每一个 PWM 都指定了对应的寄存器。此寄存器为 8 位，表示输出波形中每个调制周期的占空比。为了提高 PWM 调制频率，每一个调制周期被调制两个或四个独立的调制子区段，即分别是 7+1 或 6+2 模式。可以通过设置 BPCTL 寄存器来选择每个 PWM 通道的除能 / 使能及使用哪种模式。注意的是，当使用 PWM 时，只要将所需的值写入相应的 PWM 寄存器内，使用 BPCTL 寄存器设置需要的模式及相应 PWM 的使能 / 除能，单片机的内部硬件会自动地将波形细分为子调制周期。

PWM 时钟源就是系统时钟 f_{SYS}。将原始调制周期分成 2 个或 4 个子周期的方法，使产生更高的 PWM 频率成为可能，这样可以提供更广泛的应用。只要产生的 PWM 脉冲周期小于负载的时间常数，PWM 输出就比较合适，这是因为长时间常数负载将会平均 PWM 输出的脉冲。使用者必须理解 PWM 频率与 PWM 调制频率的不同之处。当 PWM 时钟为系统时钟 f_{SYS}，而 PWM 值为 8 位时，整个 PWM 周期的频率为 $f_{SYS}/256$。然而工作在 7+1 模式时，PWM 调制频率将会是 $f_{SYS}/128$，而工作在 6+2 模式时，PWM 调制频率将会是 $f_{SYS}/64$。

PWM 调制频率	PWM 频率	PWM 占空比
（6+2）位模式 $f_{SYS}/64$ （7+1）位模式 $f_{SYS}/128$	$f_{SYS}/256$	[PWM]/256

BPCTL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PMODE	PWM1EN	PWM0EN	BC1	BC0	BZ2	BZ1	BZ0
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 **PMODE**：PWM 模式选择位
0：7+1 模式
1：6+2 模式

Bit 6 **PWM1EN**：PWM1 或其他引脚共用功能
0：其他引脚共用功能
1：PWM1

Bit 5 **PWM0EN**：PWM0 或其他引脚共用功能
0：其他引脚共用功能
1：PWM0

Bit 4~0 蜂鸣器功能相关控制选择位，具体内容请参考蜂鸣器章节

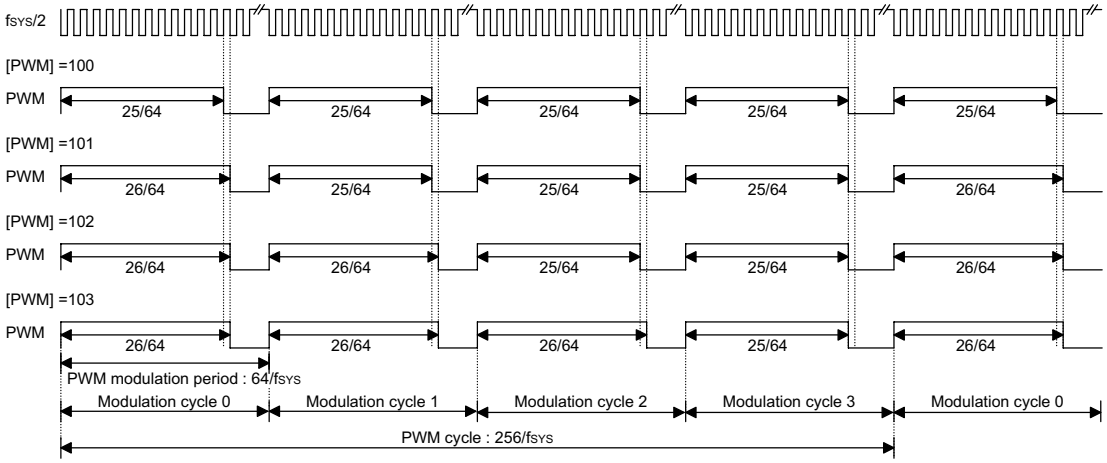
6+2 PWM 模式

通过一个 8 位的 PWM 寄存器控制，每个完整的 PWM 周期由 256 个时钟周期组成。在 6+2 PWM 模式中，每个 PWM 周期又被分成四个独立的子周期，称为调制周期 0~ 调制周期 3，在表格中以 “i” 表示。四个子周期各包含 64 个时钟周期。在这个模式下，得到以 4 为因数增加的调制频率。8 位的 PWM 寄存器被分成两个部分，这个寄存器的值表明整个 PWM 波形的占空比。第一部分包括第 2 位 ~ 第 7 位，表示 DC 值，第二部分为第 0 位 ~ 第 1 位，表示 AC 值。在 6+2 PWM 模式中，四个调制子周期的占空比，分别如下表所示。

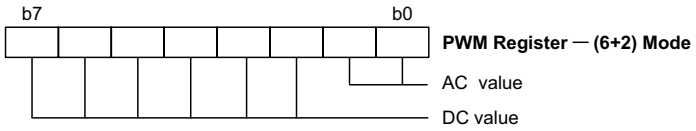
参数	AC (0~3)	DC (占空比)
调制周期 i (i=0~3)	i<AC	(DC+1)/64
	i ≥ AC	DC/64

6+2 模式调制周期值

下图表示 6+2 模式下 PWM 输出的波形。请特别注意单个的 PWM 周期是如何分成四个独立的调制周期以及 AC 值与 PWM 值的关系。



6+2 PWM 模式



6+2 模式 PWM 寄存器

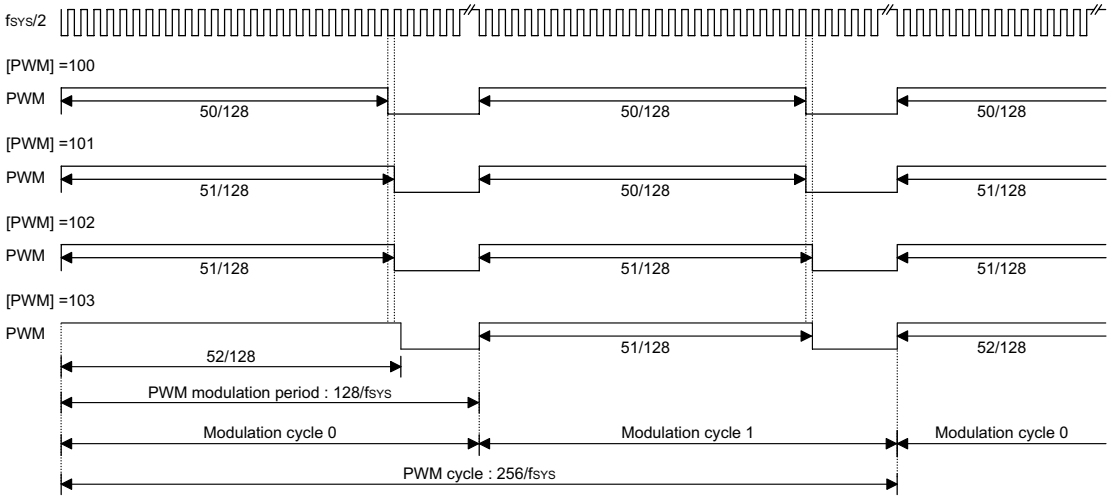
7+1 PWM 模式

通过一个 8 位的 PWM 寄存器控制，每个完整的 PWM 周期由 256 个时钟周期组成。在 7+1 PWM 模式中，每个 PWM 周期又被分成两个独立的子周期，称为调制周期 0~ 调制周期 1，在表格中以“i”表示。两个子周期各包含 128 个时钟周期。在这个模式下，得到以 2 为因数增加的调制频率。8 位的 PWM 寄存器被分成两个部分，这个寄存器的值表明整个 PWM 波形的占空比。第一部分包括第 1 位 ~ 第 7 位，表示 DC 值，第二部分为第 0 位，表示 AC 值。在 7+1 PWM 模式中，两个调制子周期的占空比，分别如下表所示。

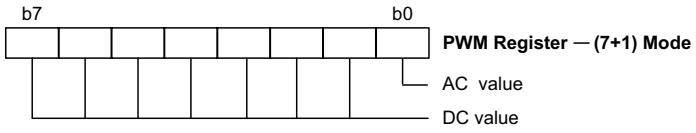
参数	AC (0~1)	DC (占空比)
调制周期 i (i=0~1)	i<AC	(DC+1)/128
	i ≥ AC	DC/128

7+1 模式调制周期值

下图表示 7+1 模式下 PWM 输出的波形。请特别注意单个的 PWM 周期是如何分成四个独立的调制周期以及 AC 值与 PWM 值的关系。



7+1 PWM 模式



7+1 模式 PWM 寄存器

PWM 输出控制

单片机的 PWM 输出与 PC5 和 PC6 端口引脚共用。要使引脚作为 PWM 输出而非普通的 I/O 引脚，必须正确设置 BPCTL 寄存器中有关 PWM 选项的位。根据所选择的 PWM 需置位相应的 PWM0EN 或 PWM1EN 位。I/O 端口控制寄存器中的相应位也必须写“0”，以确保所需要的 PWM 输出引脚设置为输出状态。在完成这两个初始化步骤，以及将所要求的 PWM 值写入 PWM 寄存器之后，将“1”写入到输出数据寄存器的相应位，则 PWM 数据就会出现在引脚上。将“0”写入到 PC 输出数据寄存器的相应位，则会除能 PWM 输出功能并强制输出低电平。通过这种方式，PC 数据输出寄存器作为 PWM 功能的开 / 关控制来使用。假如 BPCTL 寄存器已经选择 PWM 功能，但是在 PCC 控制寄存器中相应的位又写入“1”，使其成为输入引脚，则此引脚仍是作为带上拉电阻的正常输入端使用。

PWM 应用范例

下面的范例程序表明如何设置和控制 PWM0 输出。

```
mov a,64h          ; setup PWM value of decimal 100
mov pwm0, a
clr bpctl.7        ; select the 7+1 PWM mode
set bpctl.5        ; select PWM0
clr pcc.5          ; setup pin PC5
set pc.5           ; enable the PWM output
::
clr pc.5           ; disable the PWM output pin,PC5 forced low
```

A/D 转换器

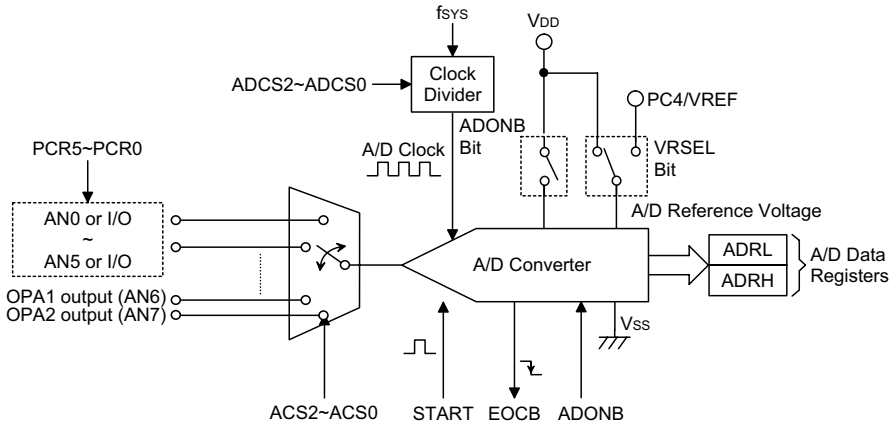
对于大多数电子系统而言，处理现实世界的模拟信号是共同的需求。为了完全由单片机来处理这些信号，首先需要通过 A/D 转换器将模拟信号转换成数字信号。将 A/D 转换器电路集成入单片机，可有效的减少外部器件，随之而来，具有降低成本和减少器件空间需求的优势。

A/D 简介

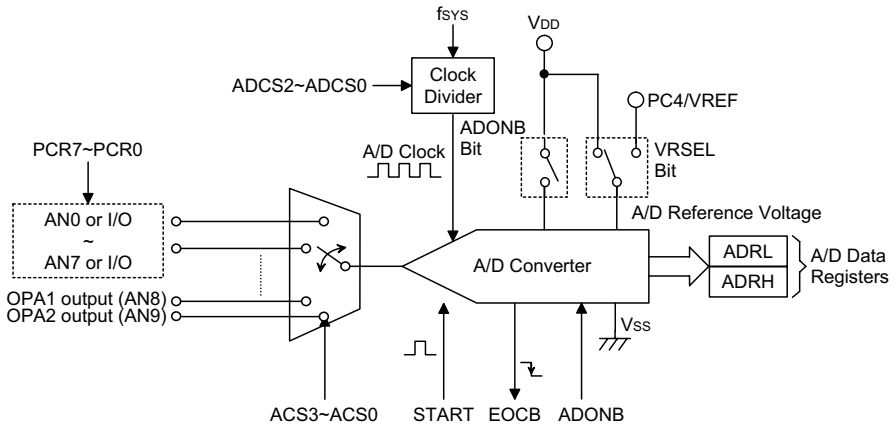
单片机都包含一个多通道的 A/D 转换器，它们可以直接接入外部模拟信号（来自传感器或其它控制信号）并直接将这些信号转换成 12 位的数字量。

型号	输入通道	A/D 通道选择位	输入引脚
HT45F23A	6+2（来自放大器输出脚）	ACS2~ACS0	AN0~AN7
HT45F24A	8+2（来自放大器输出脚）	ACS3~ACS0	AN0~AN9

下图显示了 A/D 转换器内部结构和相关的寄存器。



HT45F23A A/D 转换器结构



HT45F24A 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五个寄存器控制。一对只读寄存器来存放 12 位 ADC 数据的值。剩下三个控制寄存器设置 A/D 转换器的操作和控制功能。

寄存器名称	位							
	7	6	5	4	3	2	1	0
ADCR	START	EOCB	—	—	—	ACS2	ACS1	ACS0
ACSR	—	ADONB	VRSEL	—	—	ADCS2	ADCS1	ADCS0
ADPCR	—	—	PCR5	PCR4	PCR3	PCR2	PCR1	PCR0

A/D 转换寄存器列表 – HT45F23A

寄存器名称	位							
	7	6	5	4	3	2	1	0
ADCR	START	EOCB	—	—	ACS3	ACS2	ACS1	ACS0
ACSR	—	ADONB	VRSEL	—	—	ADCS2	ADCS1	ADCS0
ADPCR	PCR7	PCR6	PCR5	PCR4	PCR3	PCR2	PCR1	PCR0

A/D 转换寄存器列表 – HT45F24A

A/D 转换器数据寄存器 – ADRL，ADRH

对于具有 12 位 A/D 转换器的单片机，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。D0~D11 是 A/D 转换数据结果位。未使用的位读为“0”。

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

A/D 数据寄存器

A/D 转换器控制寄存器 – ADCR，ACSR，ADPCR

寄存器 ADCR，ACSR 和 ADPCR 用来控制 A/D 转换器的功能和操作。这些 8 位的寄存器定义包括选择连接至内部 A/D 转换器的模拟通道，数字化数据格式，A/D 时钟源，并控制和监视 A/D 转换器的开始和转换结束状态。寄存器 ADCR 的 ACS3~ACS0 位定义 A/D 转换器输入通道编号。由于每个单片机只包含一个实际的模数转换电路，对于 HT45F24A, 因此这 10 个模拟输入（8 个外部 A/D 通道和 2 个 OPA 输出）中的每一个都需要分别被发送到转换器。ACS3~ACS0 位的功能决定选择哪个模拟输入通道被连接到内部 A/D 转换器。

ADPCR 控制寄存器中的 PCR7~PCR0 位，用来定义 PB 口、PC 口和 PD 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。相应位设为高将选择 A/D 输入功能，清零将选择 I/O 或其它引脚共用功能。当引脚作为 A/D 输入时，其原来的 I/O 或其它引脚共用功能消失，此外，其内部上拉电阻也将自动断开。

ADCR 寄存器

- **HT45F23A**

Bit	7	6	5	4	3	2	1	0
Name	START	EOCB	—	—	—	ACS2	ACS1	ACS0
R/W	R/W	R	—	—	—	R/W	R/W	R/W
POR	0	1	—	—	—	0	0	0

Bit 7 **START**：启动 A/D 转换位
0 → 1 → 0：启动
0 → 1：重置 A/D 转换，并且设置 EOCB 为“1”
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。当此位为高，将重置 A/D 转换器。

Bit 6 **EOCB**：A/D 转换结束标志
0：A/D 转换结束
1：A/D 转换中
此位用于表明 A/D 转换过程的完成。当转换正在进行时，此位为高。

Bit 5~3 未使用，读为“0”

Bit 2~0 **ACS2~ACS0**：选择 A/D 通道
000：AN0
001：AN1
010：AN2
011：AN3
100：AN4
101：AN5
110：AN6，放大器 1 输出脚 A1E
111：AN7，放大器 2 输出脚 A2E
这些位是 A/D 通道选择控制位。由于只包含一个内部 A/D 转换电路，因此通过这些位将 8 个 A/D 输入连接到转换器。

● **HT45F24A**

Bit	7	6	5	4	3	2	1	0
Name	START	EOCB	—	—	ACS3	ACS2	ACS1	ACS0
R/W	R/W	R	—	—	R/W	R/W	R/W	R/W
POR	0	1	—	—	0	0	0	0

Bit 7 **START**：启动 A/D 转换位
0 → 1 → 0：启动
0 → 1：重置 A/D 转换，并且设置 EOCB 为“1”
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。当此位为高，将重置 A/D 转换器。

Bit 6 **EOCB**：A/D 转换结束标志
0：A/D 转换结束
1：A/D 转换中
此位用于表明 A/D 转换过程的完成。当转换正在进行时，此位为高。

Bit 5~4 未使用，读为“0”

Bit 3~0 **ACS3~ACS0**：选择 A/D 通道
0000：AN0
0001：AN1
0010：AN2
0011：AN3
0100：AN4
0101：AN5
0110：AN8，放大器 1 输出脚 A1E
0111：AN9，放大器 2 输出脚 A2E
1000：AN6
1001：AN7
这些位是 A/D 通道选择控制位。由于只包含一个内部 A/D 转换电路，因此通过这些位将 10 个 A/D 输入连接到转换器。

● **ACSR 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	ADONB	VRSEL	—	—	ADCS2	ADCS1	ADCS0
R/W	—	R/W	R/W	—	—	R/W	R/W	R/W
POR	1	0	0	—	—	0	0	0

Bit 7　未使用，读为“1”

Bit 6　**ADONB**：A/D 转换器模块电源开启 / 关闭控制位
0：A/D 转换器模块电源开启
1：A/D 转换器模块电源关闭
此位控制 A/D 内部功能的电源。该位被清零将使能 A/D 转换器。如果该位设为高将关闭 A/D 转换器以降低功耗。由于 A/D 转换器在不执行转换动作时都会产生一定的功耗，所以这在电源敏感的电池应用中需要多加注意。
注：1. 建议在进入空闲 / 休眠模式之前，设置 ADONB=1 以减小功耗。
2. ADONB=1 将关闭 A/D 转换器模块的电源。

Bit 5　**VRSEL**：A/D 转换器参考电压选择位
0：内部 ADC 电源
1：VREF 引脚或 LDO 输出（2.4V/3.3V）
此位用于选择 A/D 转换器的参考电压。如果该位设为高，A/D 转换器参考电压来源于外部 VREF
引脚或 LDO 输出（2.4V/3.3V）。如果该位设为低，内部参考电压来源于电源电压 V_{DD}。

Bit 4~3　未使用，读为“0”

Bit 2~0　**ADCS2~ADCS0**：A/D 转换器时钟源选择位
000：$f_{SYS}/2$
001：$f_{SYS}/8$
010：$f_{SYS}/32$
011：未使用
100：f_{SYS}
101：$f_{SYS}/4$
110：$f_{SYS}/16$
111：未使用
这三位用于选择 A/D 转换器的时钟源。

ADPCR 寄存器

- **HT45F23A**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCR5	PCR4	PCR3	PCR2	PCR1	PCR0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6　未使用，读为“0”

Bit 5　**PCR5**：定义 PC1 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN5

Bit 4　**PCR4**：定义 PC0 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN4

Bit 3　**PCR3**：定义 PB6 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN3

Bit 2　**PCR2**：定义 PB5 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN2

Bit 1　**PCR1**：定义 PB4 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN1

Bit 0　**PCR0**：定义 PB3 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN0

● **HT45F24A**

Bit	7	6	5	4	3	2	1	0
Name	PCR7	PCR6	PCR5	PCR4	PCR3	PCR2	PCR1	PCR0
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 **PCR7**：定义 PD1 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN7

Bit 6 **PCR6**：定义 PD0 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN6

Bit 5 **PCR5**：定义 PC1 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN5

Bit 4 **PCR4**：定义 PC0 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN4

Bit 3 **PCR3**：定义 PB6 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN3

Bit 2 **PCR2**：定义 PB5 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN2

Bit 1 **PCR1**：定义 PB4 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN1

Bit 0 **PCR0**：定义 PB3 是否为 A/D 输入
0：非 A/D 输入
1：A/D 输入，AN0

A/D 操作

ADCR 寄存器中的 START 位，用于打开和复位 A/D 转换器。当单片机设定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。当 START 位从逻辑低到逻辑高，但不再回到逻辑低时，ADCR 寄存器中的 EOCB 位置“1”，复位模数转换器。START 位用于控制内部模数换转器的开启动作。

ADCR 寄存器中的 EOCB 位用于表明模数转换过程的完成。在转换周期结束后，EOCB 位会被单片机自动地置为“0”。此外，也会置位中断控制寄存器内相应的 A/D 中断请求标志位，如果中断使能，就会产生对应的内部中断信号。A/D 内部中断信号将引导程序到相应的 A/D 内部中断入口。如果 A/D 内部中断被禁止，可以让单片机轮询 ADCR 寄存器中的 EOCB 位，检查此位是否被清除，以作为另一种侦测 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} 分频，而分频系数由 ACSR 寄存器中的 ADCS2~ADCS0 位决定。

虽然 A/D 时钟源是由系统时钟 f_{SYS}，ADCS2~ADCS0 位决定，但可选择的最大 A/D 时钟源则有一些限制。允许的 A/D 时钟周期 t_{AD} 的最小值为 0.5μs，当系统时钟速度等于或超过 4MHz 时就必须小心。如果系统时钟速度为 4MHz 时，ADCS2~ADCS0 位不能设为“100”。必须保证设定的 A/D 转换时钟周期不小于时钟周期的最小值，否则将会产生不准确的 A/D 转换值。使用者可以参考下面的表格，被标上星号 * 的数值是不允许的，因为它们的 A/D 转换时钟周期小于规定的最小值。

f_{SYS}	A/D 时钟周期 (t_{AD})						
	ADCS2, ADCS1, ADCS0 =100 (f_{SYS})	ADCS2, ADCS1, ADCS0 =000 ($f_{SYS}/2$)	ADCS2, ADCS1, ADCS0 =101 ($f_{SYS}/4$)	ADCS2, ADCS1, ADCS0 =001 ($f_{SYS}/8$)	ADCS2, ADCS1, ADCS0 =110 ($f_{SYS}/16$)	ADCS2, ADCS1, ADCS0 =010 ($f_{SYS}/32$)	ADCS2, ADCS1, ADCS0 =111 =011
1MHz	1μs	2μs	4μs	8μs	16μs	32μs	未定义
2MHz	500ns	1μs	2μs	4μs	8μs	16μs	未定义
4MHz	250ns*	500ns	1μs	2μs	4μs	8μs	未定义
8MHz	125ns*	250ns*	500ns	1μs	2μs	4μs	未定义
12MHz	83ns*	167ns*	333ns*	667ns	1.33μs	2.67μs	未定义

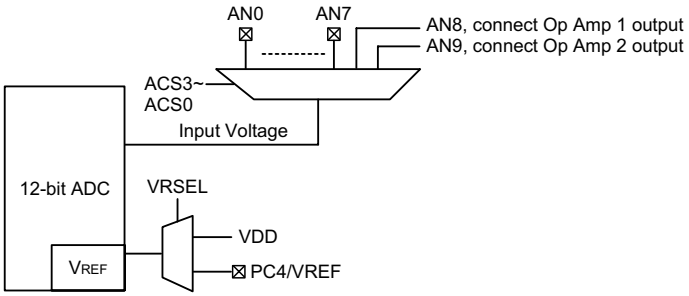
A/D 时钟周期范例

ACSR 寄存器的 ADONB 位用于控制 A/D 转换电路电源的开 / 关。该位必须清零以开启 A/D 转换器电源。即使通过清 ADPCR 寄存器的 PCR7~PCR0 位，选择无引脚作为 A/D 输入，如果 ADONB 设为“0”，那么仍然会产生功耗。因此当未使用 A/D 转换器功能时，在功耗敏感的应用中建议设置 ADONB 为高以减少功耗。

A/D 转换器参考电压来自正电源电压 VDD 或外部参考源引脚 VREF，可通过 VRSEL 位来选择。由于 VREF 引脚与其它功能共用，当 VRSEL 设为高，选择 VREF 引脚或 LDO 输出功能且其它引脚功能将自动除能。

A/D 输入引脚

所有的 A/D 模拟输入引脚都与 PB、PC 和 PD 端口的 I/O 引脚及其它功能共用。使用 ADPCR 寄存器中的 PCR7~PCR0 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。如果引脚的对应位 PCR7~PCR0 设为高，那么该引脚作为 A/D 转换输入且原引脚功能除能。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，PBC 或 PCC 或 PDC 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 PCR7~PCR0 位使能 A/D 输入时，端口控制寄存器的状态将被重置。



A/D 输入结构

A/D 转换器有自己的参考电压引脚 VREF，而通过设置 ACSR 寄存器的 VRSEL 位，参考电压也可以选择来自电源电压引脚。模拟输入值一定不能超过 V_{REF} 值。

A/D 转换步骤

下面概述实现 A/D 转换过程的各个步骤。

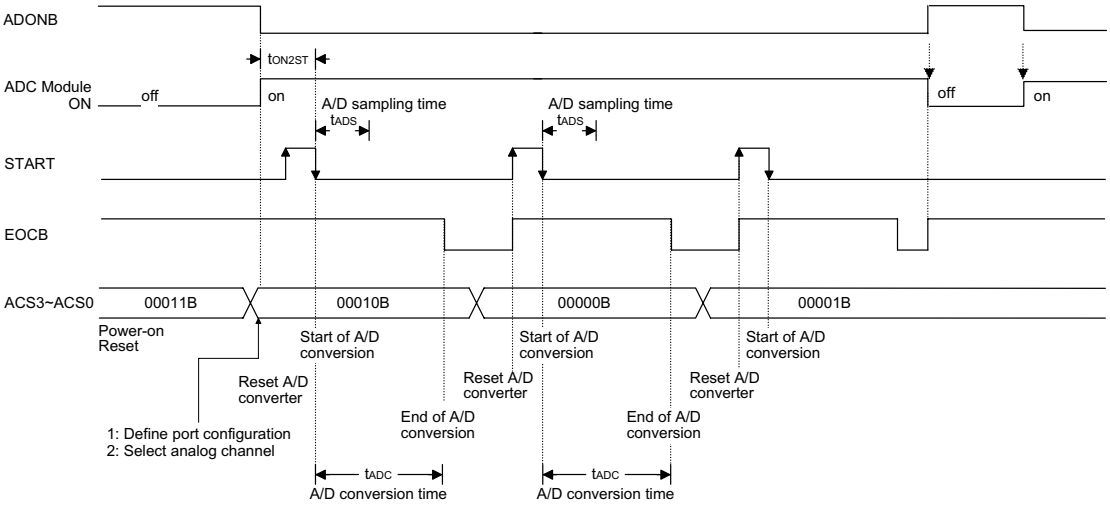
- 步骤 1
  通过 ACSR 寄存器中的 ADCS2~ADCS0 位，选择所需的 A/D 转换时钟。
- 步骤 2
  清零 ACSR 寄存器中的 ADONB 位使能 A/D。
- 步骤 3
  通过 ADCR 寄存器中的 ACS3~ACS0 位，选择连接至内部 A/D 转换器的通道。
- 步骤 4
  通过 ADPCR 寄存器中的 PCR7~PCR0 位，选择哪些引脚规划为 A/D 输入引脚。
- 步骤 5
  如果要使用中断，则中断控制寄存器需要正确地设置，以确保 A/D 转换功能是激活的。总中断控制位 EMI 需要置位为“1”，以及 A/D 转换器中断位 EADI 也需要置位为“1”。
- 步骤 6
  现在可以通过设定 ADCR 寄存器中的 START 位从“0”到“1”再回到“0”，开始模数转换的过程。注意，该位需初始化为“0”。

- 步骤 7

  可以轮询 ADCR 寄存器中的 EOCB 位，检查模数转换过程是否完成。当此位成为逻辑低时，表示转换过程已经完成。转换完成后，可读取 A/D 数据寄存器 ADRL 和 ADRH 获得转换后的值。另一种方法是，若中断使能且堆栈未满，则程序等待 A/D 中断发生。

  注：若使用轮询 ADCR 寄存器中 EOCB 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。

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 $16t_{AD}$，t_{AD} 为 A/D 时钟周期。



A/D 转换时序图

注意事项

在编程时，如果 A/D 转换器未使用，通过设置 ACSR 寄存器中的 ADONB 为高，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

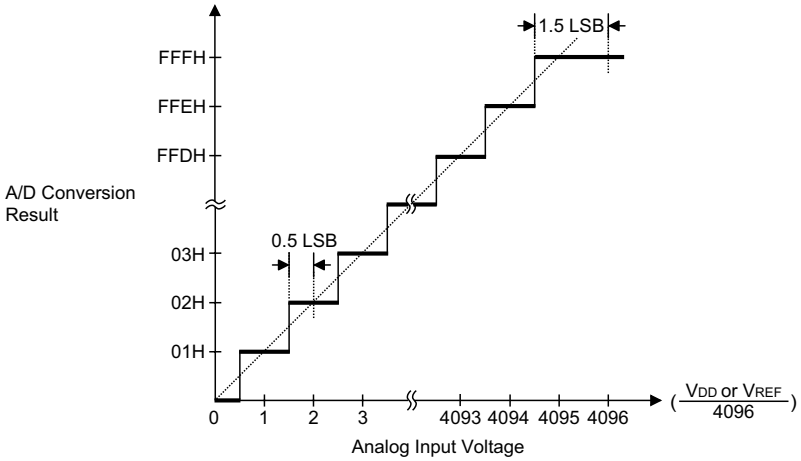
单片机含有一组 12 位的 A/D 转换器，它们转换的最大值可达 FFFH。由于模拟输入最大值等于 V_{DD} 或 V_{REF} 的电压值，因此每一位可表示 V_{DD} 或 $V_{REF}/4096$ 的模拟输入值。

$$1\ LSB=(V_{DD} 或 V_{REF})\div 4096$$

通过下面的等式可估算 A/D 转换器输入电压值：

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

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_{DD} 或 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 ADCR 寄存器中的 EOCB 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：使用查询 EOCB 的方式来检测转换结束

```
clr  EADI                    ; disable ADC interrupt
mov  a, 01H
mov  ACSR, a                 ; select fSYS/8 as A/D clock
                             ; Select VDD as ADC reference voltage and turn
                             ; on ADONB bit
mov  a, FFh                  ; setup ADPCR to configure pins AN0~AN7
mov  ADPCR, a
mov  a, 00h
mov  ADCR,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   EOCB                    ; poll the ADCR0 register EOCB 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：使用中断的方式来检测转换结束

```
clr  EADI                    ; disable ADC interrupt
mov  a, 01H
mov  ACSR, a                 ; select fSYS/8 as A/D clock
                             ; select VDD as ADC reference voltage and turn
                             ; on ADONB bit
mov  a, FFh                  ; setup ADPCR to configure pins AN0~AN7
mov  ADPCR, a
mov  a,00h
mov  ADCR, 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  EADI                    ; enable ADC interrupt
set  EMFI                    ; enable Multi-function interrupt
set  EMI                     ; enable global interrupt
:
                             ; ADC interrupt service routine
ADC_ISR:
mov  acc_stack, a            ; save ACC to user defined memory
mov  a, STATUS
mov  status_stack,a          ; save STATUS to user defined memory
:
mov  a, 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
clr  ADF                     ; clear ADC interrupt request flag
reti
```

串行接口模块 -- SIM

单片机内有一个串行接口模块，包括两种易与外部设备通信的串行接口：四线 SPI 或两线 I²C 接口。这两种接口具有相当简单的通信协议，单片机可以通过这些接口与传感器、闪存或 EEPROM 内存等硬件设备通信。SIM 接口的引脚与 PB0~PB3 引脚共用，所以要使用 SIM 功能时应先在配置选项中选中 SIM 功能。因为这两种接口共用引脚和寄存器，所以要通过一个 SIMC0 寄存器中的 SIM2~SIM0 位来选择哪一种通信接口。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 脚的上拉电阻。

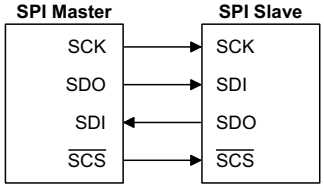
SPI 接口

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。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

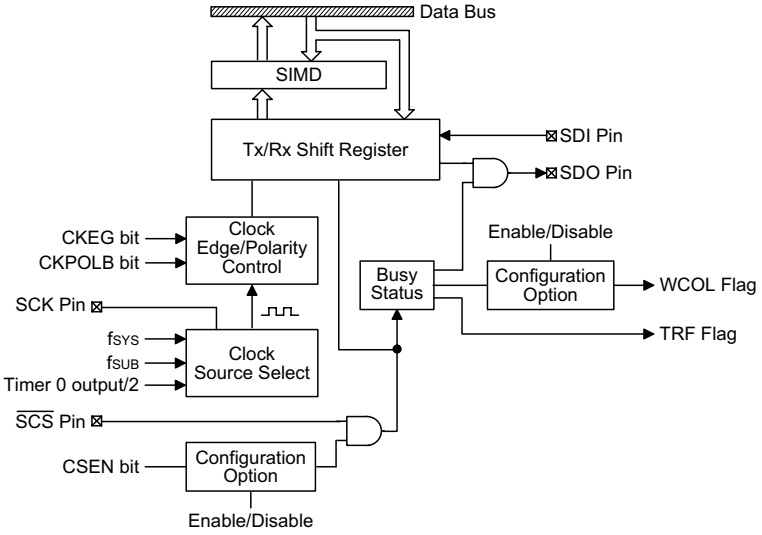
SPI 接口是一个全双工串行数据传输器。SPI 接口的四线为：SDI、SDO、SCK 和 $\overline{SCS}$。SDI 和 SDO 是数据的输入和输出线。SCK 是串行时钟线，$\overline{SCS}$ 是从机的选择线。SPI 的接口引脚与普通 I/O 口和 I²C 的功能脚共用。通过设定 SIM 配置选项和 SIMC0/SIMC2 寄存器的对应位，来使能 SPI 接口。SPI 配置选项设定好后，还可以通过 SIMC0 寄存器中的 SIMEN 位来除能或使能。连接到 SPI 接口的单片机以主 / 从模式进行通信，且主机完成所有的数据传输初始化，并控制时钟信号。由于单片机只有一个 $\overline{SCS}$ 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

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 LSB 或最高有效位先传 MSB 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
- WCOL 和 CSEN 位使能或除能选择



SPI 方框图

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。

配置选项中有几项与 SPI 接口功能相关。其中一项为使能 SIM 功能，共用引脚选择为 SIM 脚而非普通输入 / 输出脚。注意，若配置选项未选择 SIM 功能，SIMC0 寄存器中的 SIMEN 位的状态不会产生影响。另外两个 SPI 配置选项决定 CSEN 和 WCOL 位是否有用。

SPI 寄存器

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

寄存器名称	位							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	—
SIMD	D7	D6	D5	D4	D3	D2	D1	D0
SIMC2	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF

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

SIMD 寄存器

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

“×”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 I²C 接口功能中的的寄存器 SIMA 是同一个寄存器。SPI 功能不会用到寄存器 SIMC1，SIMC1 只适用于 I²C 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	PCKEN	PCKP1	PCKP0	SIMEN	—
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 Timer0 输出 /2（PFD0）
101：SPI 从机模式
110：I²C 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 I²C 或 SPI 功能。SPI 时钟源可来自于系统时钟也可以选择来自 Timer0。若选择的是作为 SPI 从机，则其时钟源从外部主机而得。

Bit 4 **PCKEN**：PCK 输出脚控制位
0：除能
1：使能

Bit 3~2 **PCKP1，PCKP0**：选择 PCK 输出脚的频率位
00：f_{SYS}
01：$f_{SYS}/4$
10：$f_{SYS}/8$
11：Timer0 输出 /2（PFD）

Bit 1 **SIMEN**：SIM 控制位
0：除能
1：使能
此位为 SIM 接口的开 / 关控制位。此位为“0”时，SIM 接口除能，SDI、SDO、SCK 和 $\overline{SCS}$ 或 SDA 和 SCL 脚处于浮空状态，SIM 工作电流减小到最小值。此位为“1”时，SIM 接口使能。配置选项中首先将 SIM 接口使能才能使此位有效。若 SIM 经由 SIM2~SIM0 位设置为工作在 SPI 接口，当 SIMEN 位由低到高转变时，SPI 控制寄存器中的设置不会发生变化，其首先应在应用程序中初始化。若 SIM 经由 SIM2~SIM0 位设置为工作在 I²C 接口，当 SIMEN 位由低到高转变时，I²C 控制寄存器中的设置，如 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 I²C 标志，如 HCF、HAAS、HBB、SRW 和 RXAK，将被设置为其默认状态。

Bit 0 未使用，读为“0”

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 $\overline{SCS}$ 引脚控制位
0：除能
1：使能
CSEN 位用于 $\overline{SCS}$ 引脚的使能 / 除能控制。此位为低时，$\overline{SCS}$ 除能并处于浮空状态。此位为高时，$\overline{SCS}$ 作为选择脚。注意，CSEN 位的使能 / 除能可通过配置选项设置。

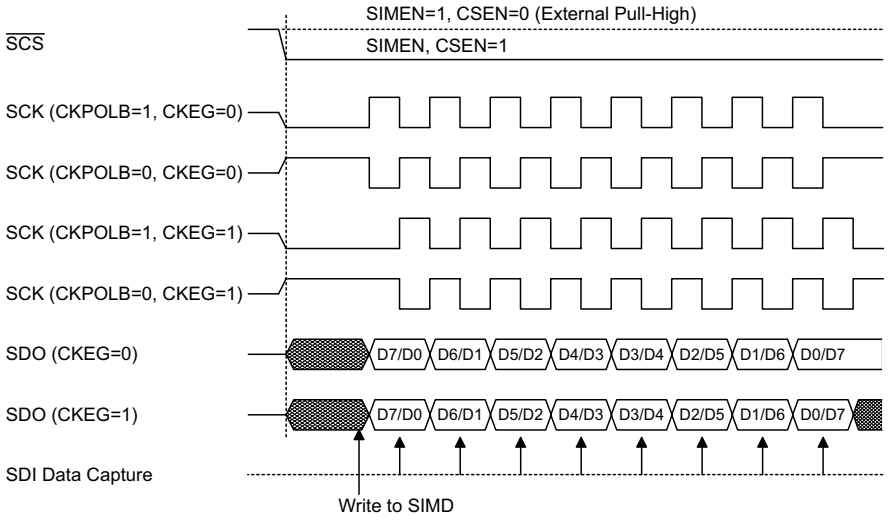
Bit 1 **WCOL**：SPI 写冲突标志位
0：无冲突
1：冲突
WCOL 标志位用于监测数据冲突的发生。此位为高时，数据在传输时被写入 SIMD 寄存器。若数据正在被传输时，此操作无效。此位可被应用程序清零。注意，WCOL 位的使能 / 除能可通过配置选项设置。

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。

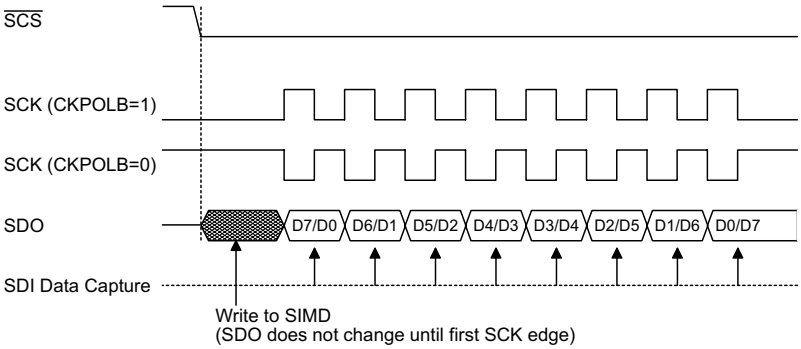
SPI 通信

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 $\overline{SCS}$ 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。

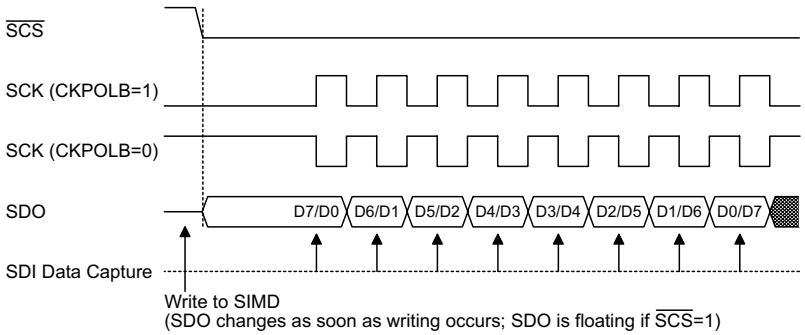
即使在单片机处于空闲模式，SPI 功能仍将继续执行。



SPI 主机模式时序

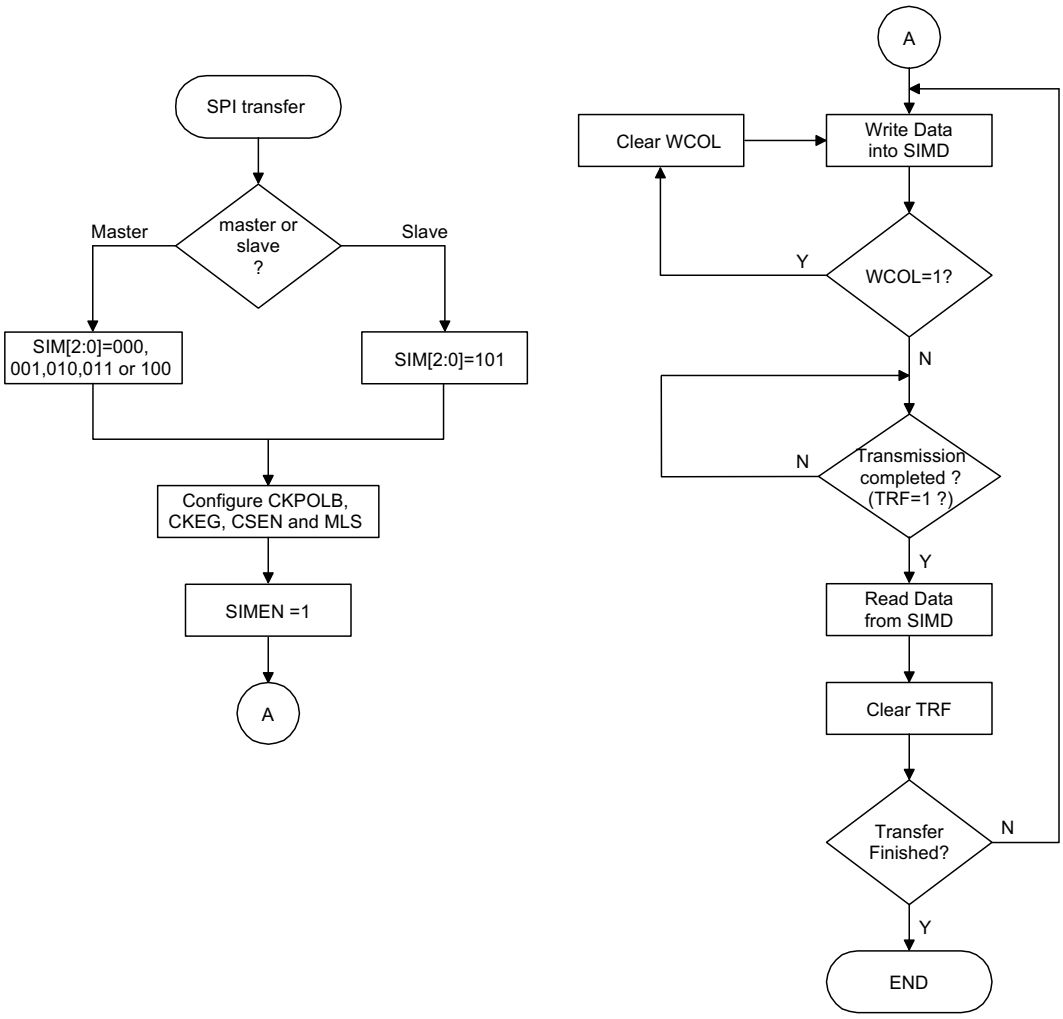


SPI 从机模式时序 -- CKEG=0



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

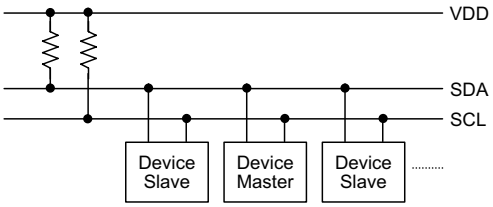
SPI 从机模式时序 -- CKEG=1



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。



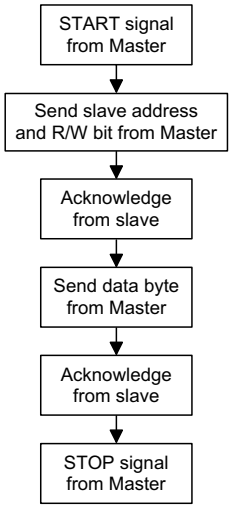
I²C 主从总线连接图

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。

单片机中有几个和 I²C 接口相关的配置选项，其中之一是使能 I²C 功能，选择 SIM 引脚而非普通 I/O 口。应注意的是如果在配置选项中没有将 SIM 功能使能，那么寄存器 SIMC0 中的 SIMEN 位不起作用。配置选项中有一个选项允许使用其它的时钟来驱动 I²C 接口。另外有一个配置选项用于控制 I²C 接口的去抖时间大小。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 1 个或 2 个系统时钟。



I²C 寄存器

I²C 总线的三个控制寄存器是 SIMC0 和 SIMC1，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，当单片机将数据写入 I²C 总线之前，实际将被传输的数据存放在寄存器 SIMD 中。从 I²C 总线接收到数据之后，单片机就可以从寄存器 SIMD 中得到这个数据。I²C 总线上的所有传输或接收到的数据都必须通过 SIMD。应注意的是 SIMA 也有另外一个名字，SIMC2，使用 SPI 功能时会用到。I²C 接口会用到寄存器 SIMC0 中的 SIMEN 位和 SIM0~SIM2 位。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	—
SIMC1	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
SIMD	D7	D6	D5	D4	D3	D2	D1	D0
SIMA	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	—

I²C 寄存器列表

SIMC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	—
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 Timer0 输出 /2（PFD0）
101：SPI 从机模式
110：I²C 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 I²C 或 SPI 功能。SPI 时钟源可来自于系统时钟也可以选择来自 Timer0。若选择的是作为 SPI 从机，则其时钟源从外部主机而得。

Bit 4 **PCKEN**：PCK 输出脚控制位
0：除能
1：使能

Bit 3~2 **PCKP1，PCKP0**：选择 PCK 输出脚的频率位
00：f_{SYS}
01：$f_{SYS}/4$
10：$f_{SYS}/8$
11：Timer0 输出 /2（PFD0）

Bit 1 **SIMEN**：SIM 控制位
0：除能
1：使能
此位为 SIM 接口的开 / 关控制位。此位为“0”时，SIM 接口除能，SDI、SDO、SCK 和 $\overline{SCS}$ 或 SDA 和 SCL 脚处于浮空状态，SIM 工作电流减小到最小值。此位为“1”时，SIM 接口使能。配置选项中首先将 SIM 接口使能才能使此位有效。若 SIM 经由 SIM2~SIM0 位设置为工作在 SPI 接口，当 SIMEN 位由低到高转变时，SPI 控制寄存器中的设置不会发生变化，其首先应在应用程序中初始化。若 SIM 经由 SIM2~SIM0 位设置为工作在 I²C 接口，当 SIMEN 位由低到高转变时，I²C 控制寄存器中的设置，如 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 I²C 标志，如 HCF、HAAS、HBB、SRW 和 RXAK，将被设置为其默认状态。

Bit 0 未使用，读为“0”

SIMC1 寄存器

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

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

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

Bit 5 **HBB**：I²C 总线忙标志位
0：I²C 总线闲
1：I²C 总线忙
HBB 是 I²C 总线忙标志位。当检测到 START 信号时 I²C 忙，此位变为高电平。当检测到 STOP 信号时 I²C 总线停止，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**：I²C 总线发送确认标志位
0：从机发送确认标志
1：从机没有发送确认标志
单片机接收 8 位数据之后会将该位在第九个时钟传到总线上。如果单片机想要接收更多的数据，则应在接收数据之前将此位设置为“0”。

Bit 2 **SRW**：I²C 从机读 / 写位
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。

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。

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

SIMD 寄存器

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

“×”为未知

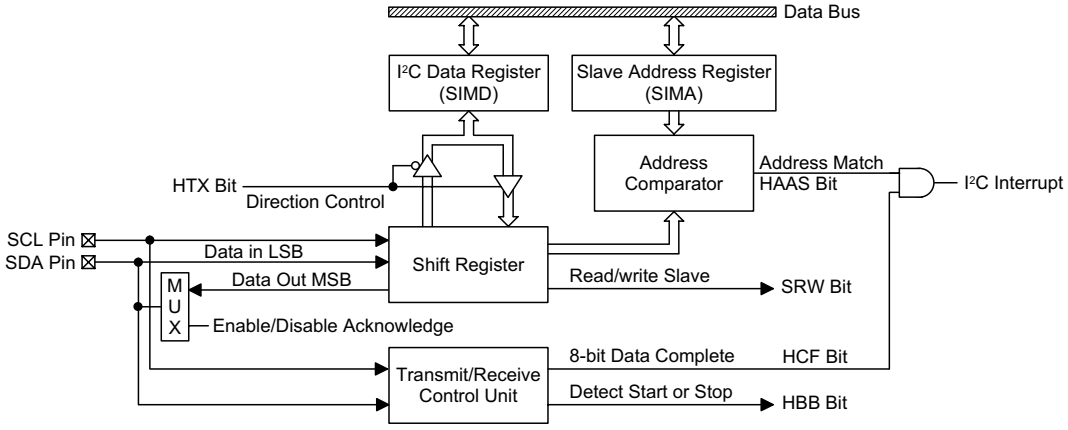
SIMA 寄存器

Bit	7	6	5	4	3	2	1	0
Name	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	—
POR	×	×	×	×	×	×	×	—

“×”为未知

Bit 7~1 **IICA6~IICA0**：I²C 从机地址位
IICA6~IICA0 是从机地址对应的 6~0 位。此寄存器也在 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 中的第 7~1 位是单片机的从机地址，位 0 未定义。如果接至 I²C 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 0 无定义
此位可通过软件程序进行读写。



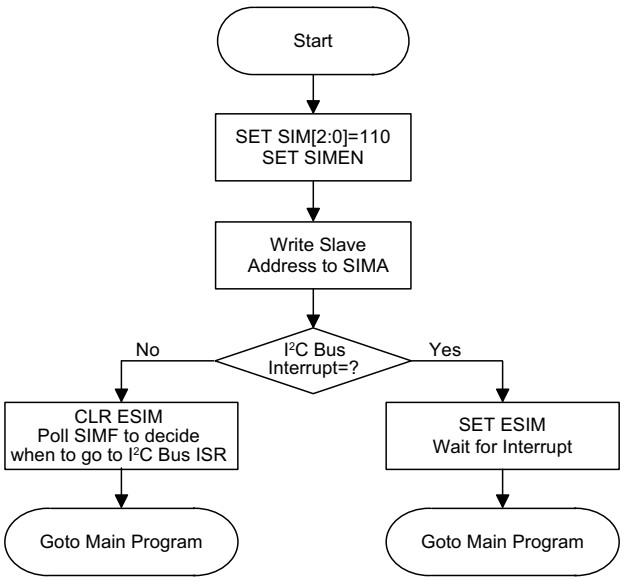
I²C 方框图

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，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。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 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 ESIM 位以使能 SIM 中断。



I²C 总线初始化流程图

I²C 总线起始信号

起始信号只能由连接 I²C 总线主机产生，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 总线有两个中断源，当程序运行至中断服务子程序时，通过检测 HAAS 位以确定 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。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。

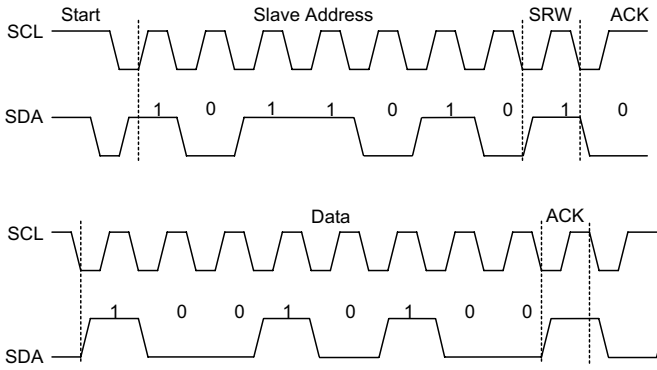
I²C 总线从机地址确认信号

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。

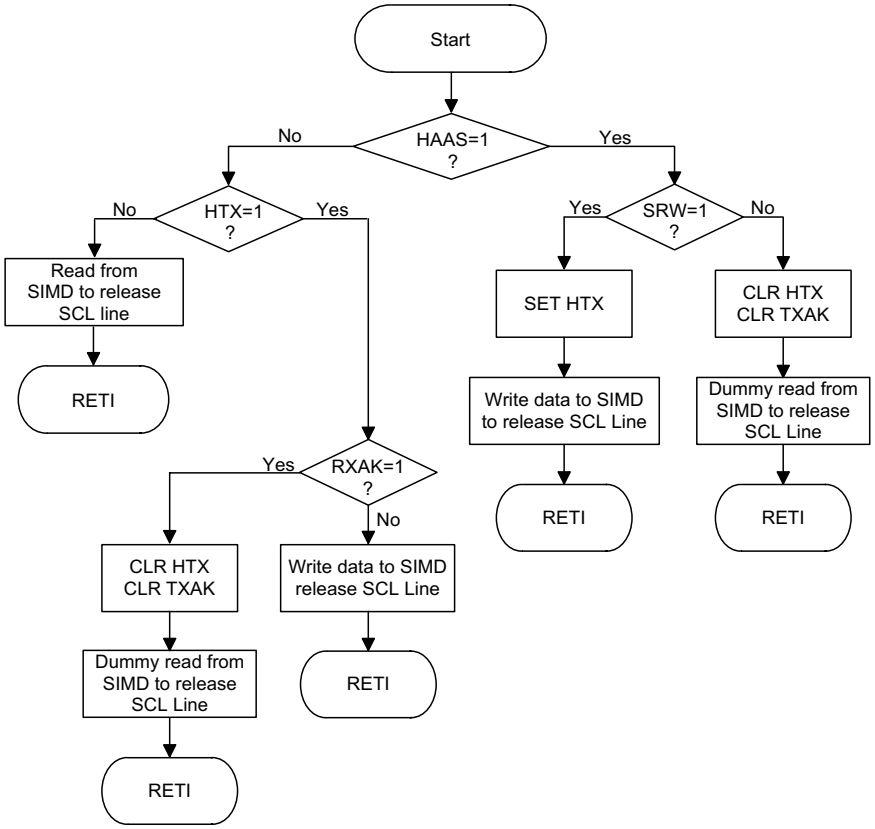


S=Start (1 bit)
SA=Slave Address (7 bits)
SR=SRW bit (1 bit)
M=Slave device send acknowledge bit (1 bit)
D=Data (8 bits)
A=ACK (RXAK bit for transmitter, TXAK bit for receiver 1 bit)
P=Stop (1 bit)

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



I²C 总线 ISR 流程图

外围时钟输出

外围时钟输出功能使单片机能够为外部硬件提供和单片机时钟同步的时钟信号。

外围时钟操作

外围时钟输出引脚 PCK 与 PB4 脚共用，可以通过 SIMC0 寄存器的 PCKEN 位来选择引脚功能。外围时钟功能由 SIMC0 寄存器控制。外围时钟输出的时钟源来自 Timer0 输出频率 /2 或内部系统时钟分频。SIMC0 寄存器的 PCKEN 位是总的开 / 关控制位，当该位为高时使能外围时钟，为低时除能外围时钟。系统时钟所需要的分频比由同一个寄存器中的 PCKP0 和 PCKP1 位来选择。如果系统进入休眠模式，将除能外围时钟输出功能。

SIMC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	PCKEN	PCKP1	PCKP0	SIMEN	—
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 Timer0 输出 /2（PFD0）
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也可以选择来自 Timer0。若选择的是作为 SPI 从机，则其时钟源从外部主机而得。

Bit 4 **PCKEN**：PCK 输出脚控制位
0：除能
1：使能

Bit 3~2 **PCKP1，PCKP0**：选择 PCK 输出脚的频率位
00：f_{SYS}
01：$f_{SYS}/4$
10：$f_{SYS}/8$
11：Timer0 输出 /2（PFD）

Bit 1 **SIMEN**：SIM 控制位
0：除能
1：使能
此位为 SIM 接口的开 / 关控制位。此位为“0”时，SIM 接口除能，SDI、SDO、SCK 和 $\overline{SCS}$ 或 SDA 和 SCL 脚处于浮空状态，SIM 工作电流减小到最小值。此位为“1”时，SIM 接口使能。配置选项中首先将 SIM 接口使能才能使此位有效。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 未使用，读为“0”

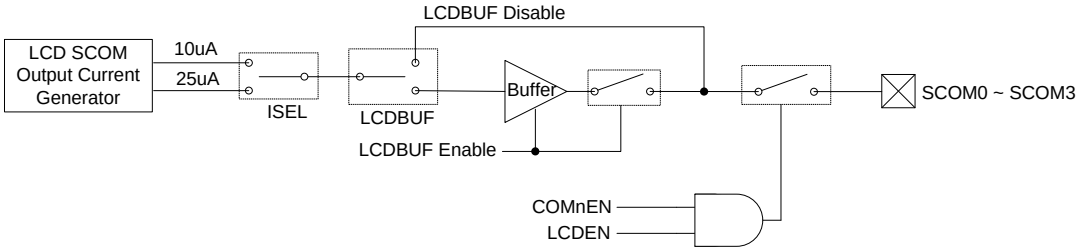
带 SCOM 功能的 LCD

单片机具有驱动外部 LCD 面板的能力。LCD 驱动的 COM 脚 SCOM0~SCOM3 与 PA0、PC4~PC6 引脚共用。LCD 控制信号 (COM & SEG) 由软件编程实现。

LCD 操作

单片机通过设置 PA0、PC4~PC6 作为 COM 引脚，其它输出口作为 SEG 引脚，以驱动外部的液晶面板。LCD 驱动功能是由 LCDC 寄存器来控制，另外，该寄存器可设置 LCD 的开启和关闭以及输出偏压值等功能，使得 COM 口输出 $V_{DD}/2$ 的电压，从而实现 1/2 bias LCD 的显示。

LCDC 寄存器中的 LCDEN 位是 LCD 驱动的主控制位，它与 COMnEN 位搭配共同设置 I/O 端口是否用于 LCD 驱动。注意，作为 LCD 驱动时，端口控制寄存器不需要设置为输出，即可使能 LCD 驱动操作。下图为 LCD COM 的功能结构图。



LCD 电路图

LCDEN	COMnEN	引脚功能	输出电平
0	×	I/O	高或低
1	0	I/O	高或低
1	1	SCOMn	VM

输出控制

LCD 偏压控制

LCD 驱动器提供两种驱动选择以适应所使用 LCD 面板的需求。通过设置 LCDC 寄存器中 ISEL 位可以配置不同的偏压电阻。

LCDC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	LCDBUF	ISEL	LCDEN	COM3EN	COM2EN	COM1EN	COM0EN
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　未使用，读为“0”

Bit 6　**LCDBUF: LCD 缓冲控制位**
0：除能
1：使能

Bit 5　**ISEL**：选择 SCOM 工作电流 (V_{DD}=5V)
0：10μA
1：25μA

Bit 4　**LCDEN**：LCD 控制位
0：除能
1：使能
如果 LCDEN=1，SCOMn 由 COMnEN 使能

Bit 3　**COM3EN: 选择 PC6 或者 SCOM3**
0：GPIO
1：SCOM3

Bit 2　**COM2EN**：选择 PC5 或者 SCOM2
0：GPIO
1：SCOM2

Bit 1　**COM1EN**：选择 PC4 或者 SCOM1
0：GPIO
1：SCOM1

Bit 0　**COM0EN**：选择 PA0 或者 SCOM0
0：GPIO
1：SCOM0

注：单片机提供 LCD 缓存功能以防止 LCD 面板受到干扰，由 LCDBUF 位控制。使用该缓存可为 OPA 和比较器提供更为稳定的参考电压 VH0/1、VL0/1。必须注意的是，LCD SCOM 电源电压来自 V_{LDO} 或 LCD 面板尺寸大于 LCD 缓存，系统将消耗更高的驱动电流。因此，开启此缓存将更加耗电。

LDO 功能

单片机内部集成一个 CMOS 结构的 LDO 稳压器，用于产生一个稳定的电压。此 LDO 电路可提供 2.4V 和 3.3V 两个输出电压，可由寄存器 LDOC 控制选择。LDO 输出的选项可由寄存器控制，为 LCD 偏置电压、OPA 参考电压、A/D 转换器参考电压及外部器件的供电电源提供一个稳定的参考。

LDOC 控制寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	VLOE	REN1	VRES	VSEL	LDOEN
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	0	0	0	0	0

Bit 7~5　未使用，读为“0”

Bit 4　**VLOE**：LDO 输出电压控制位
0：除能
1：使能
如果 VLOE 和 LDOEN 均设置为“1”，LDO 将输出 2.4V 或 3.3V 的电压到引脚，且除能其对应的 I/O 功能。

Bit 3　**REN1**：偏压分压电阻控制位
0：除能
1：使能
若 REN1 被置为“1”，电阻 DC 路径开启，可产生偏置电压，用于运算放大器或 LCD SCOM 功能。

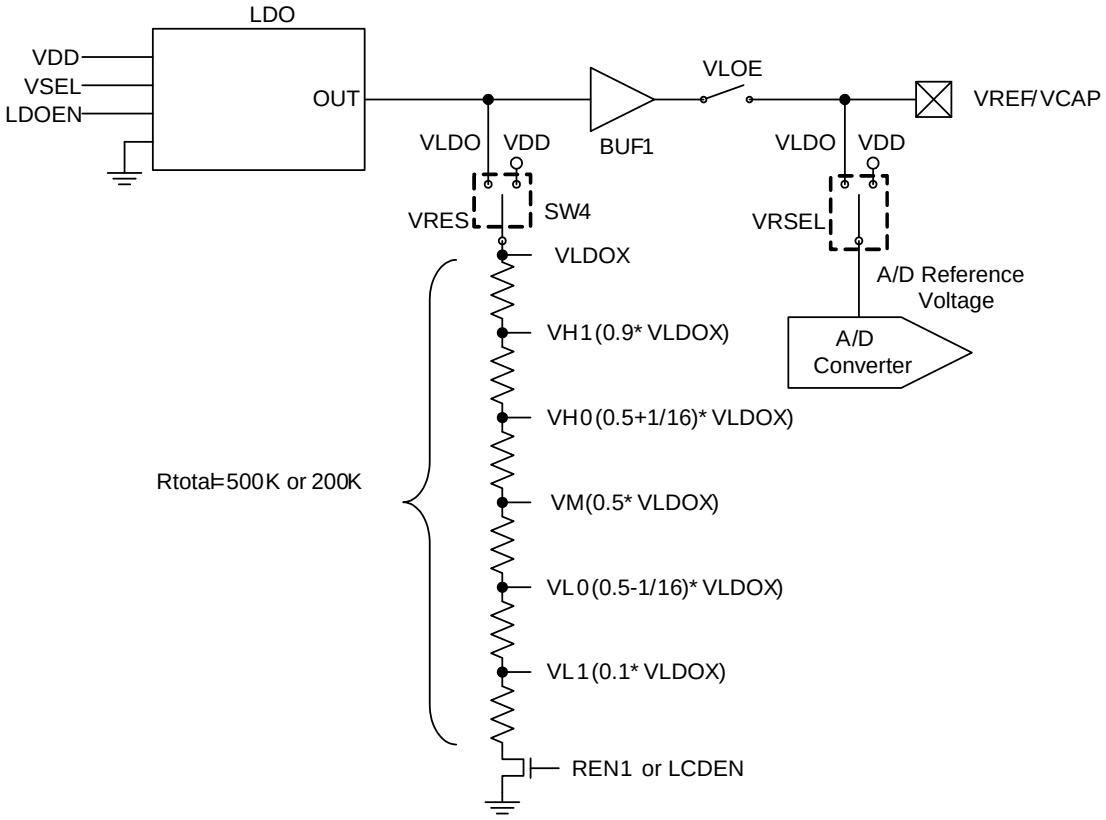
Bit 2　**VRES**：分压电阻电压来源选择位
0：VDD
1：VLDO

Bit 1　**VSEL**：LDO 输出电压选择位
0：2.4V
1：3.3V

Bit 0　**LDOEN**：LDO 控制位
0：除能
1：使能

注：1. 分压电阻的总阻值—500kΩ 或 200kΩ，由 LCDC 寄存器中的 ISEL 位控制。
2. 无论 LDO 输出电压控制位 VLOE 是否使能，若除能 LDO 功能，BUF1 也将关闭。
3. 若 LDO 输出作为 A/D 转换器的参考电压，VCAP 将连接一个 0.1μF 的电容至地。
4. 若 LDO 除能，即 LDOEN=0，无论 VRES 的状态为何，SW4 都将转至 VDD。

LDO 功能模块及分压电阻方框图如下：



LDO 功能方框图

运算放大器 OPA

单片机内部集成两个运算放大器，OPA1 和 OPA2，可用于用户特定的模拟信号处理。它们的使能或除能只能通过软件设置来实现。通过控制特殊寄存器，OPA 相关的应用可以很容易的实现，例如单位增益缓冲器，同相放大器，反相放大器和各种各样的滤波器等。

运算放大器寄存器

内部运算放大器的控制寄存器有：OPA1C0，OPA1C1，OPA2C0，OPA2C1 和 OPA2C2。这些寄存器用来控制运算放大器的使能 / 除能功能、输入路径选择、增益控制和极性。

OPA1C0 控制寄存器

Bit	7	6	5	4	3	2	1	0
Name	A1X	—	—	—	—	—	—	—
R/W	R	—	—	—	—	—	—	—
POR	0	—	—	—	—	—	—	—

Bit 7 **A1X**：运算放大器输出脚，正逻辑电平。此为只读位。

Bit 6~0 未使用，读为“0”

OPA1C1 控制寄存器

Bit	7	6	5	4	3	2	1	0
Name	A1O2CIN	A1O2N	A1PSEL1	A1PSEL0	A1PS	A1NS	A1OEN	A1EN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	1	1	0	0

Bit 7 **A1O2CIN**：OPA1 输出作比较器输入功能控制位
0：除能
1：使能

Bit 6 **A1O2N**：OPA1 输出作 OPA1 反相输入功能控制位
0：除能
1：使能

Bit 5~4 **A1PSEL1~ A1PSEL0**：OPA1 同相输入端选择位
00：无选择
01：来自 VH1 ($0.9\times V_{LDO}$)
10：来自 VM ($0.5\times V_{DD}$ 或 $0.5\times V_{LDO}$)
11：来自 VL1 ($0.1\times V_{DD}$ 或 $0.1\times V_{LDO}$)

Bit 3 **A1PS**：A1P 脚作为 OPA1 同相输入端选择位
0：无连接
1：A1P 脚作为 OPA1 同相输入端

Bit 2 **A1NS**：A1N 脚作为 OPA1 反相输入端选择位
0：无连接
1：A1N 脚作为 OPA1 反相输入端

Bit 1 **A1OEN**：OPA1 输出控制位
0：除能
1：使能
注 : 如果 OPA1 使能且 A1OEN 设为 1，单片机的 DC 功耗将增加（100μA~200μA）

Bit 0 **A1EN**：OPA1 功能控制位
0：除能
1：使能

OPA2C0 控制寄存器

Bit	7	6	5	4	3	2	1	0
Name	A2X	—	—	—	—	—	—	—
R/W	R	—	—	—	—	—	—	—
POR	0	—	—	—	—	—	—	—

Bit 7 **A2X**：运算放大器输出脚，正逻辑电平。此为只读位。

Bit 6~0 未使用，读为“0”

OPA2C1 控制寄存器

Bit	7	6	5	4	3	2	1	0
Name	A2O2CIN	A2O2N	A2PSEL1	A2PSEL0	A2PS	A2NS	A2OEN	A2EN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	1	1	0	0

Bit 7 **A2O2CIN**：OPA2 输出作比较器输入功能控制位
0：除能
1：使能

Bit 6 **A2O2N**：OPA2 输出作 OPA2 反相输入功能控制位
0：除能
1：使能

Bit 5~4 **A2PSEL1~ A2PSEL0**：OPA2 同相输入端选择位
00：无选择
01：来自 VH1 ($0.9\times V_{LDO}$)
10：来自 VM（$0.5\times V_{DD}$ 或 $0.5\times V_{LDO}$）
11：来自 VL1 ($0.1\times V_{DD}$ 或 $0.1\times V_{LDO}$)

Bit 3 **A2PS**：A2P 脚作为 OPA2 同相输入端选择位
0：无连接
1：A2P 脚作为 OPA2 同相输入端

Bit 2 **A2NS**：A2N 脚作为 OPA2 反相输入端选择位
0：无连接
1：A2N 脚作为 OPA2 反相输入端

Bit 1 **A2OEN**：OPA2 输出控制位
0：除能
1：使能
注：如果 OPA2 使能且 A2OEN 设为 1，单片机的 DC 功耗将增加（100μA~200μA）

Bit 0 **A2EN**：OPA2 功能控制位
0：除能
1：使能

OPA2C2 控制寄存器

Bit	7	6	5	4	3	2	1	0
Name	A1O2A2N	A1O2A2P	—	—	PGAEN	PGA2	PGA1	PGA0
R/W	R/W	R/W	—	—	R/W	R/W	R/W	R/W
POR	0	0	—	—	0	0	0	0

Bit 7 **A1O2A2N**：OPA1 输出作 OPA2 反相输入功能控制位
0：除能
1：使能

Bit 6 **A1O2A2P**：OPA1 输出作 OPA2 同相输入功能控制位
0：除能
1：使能

Bit 5~4 未使用，读为“0”

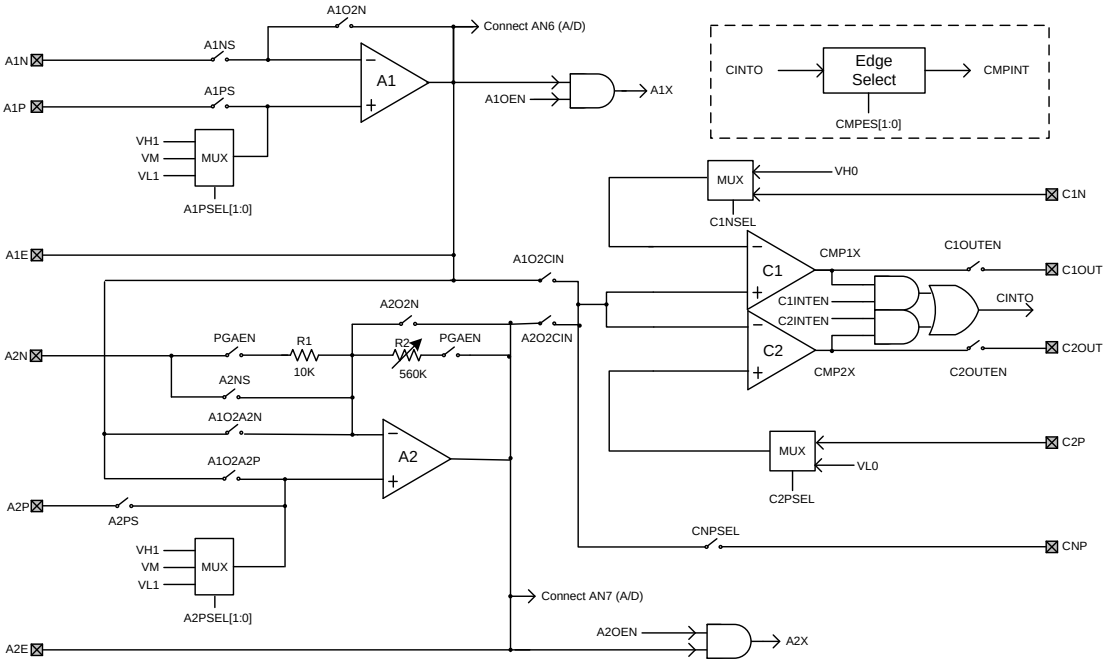
Bit 3 **PGAEN**：OPA2 PGA 增益控制位
0：除能
1：使能

Bit 2~0 **PGA2~PGA0**：增益比控制位
000：1
001：8
010：16
011：24
100：32
101：40
110：48
111：56

运算放大器操作

单片机中的运算放大器有多重开关和输入路径选项、各种参考电压选择、多达 8 种内部可编程增益控制、用于低功耗模式的暂停控制等多种优点，使得这两个运算放大器可以适用于较广应用领域。

单片机中运算放大器和比较器功能模块结构如下图所示。



运算放大器和比较器功能模块结构图

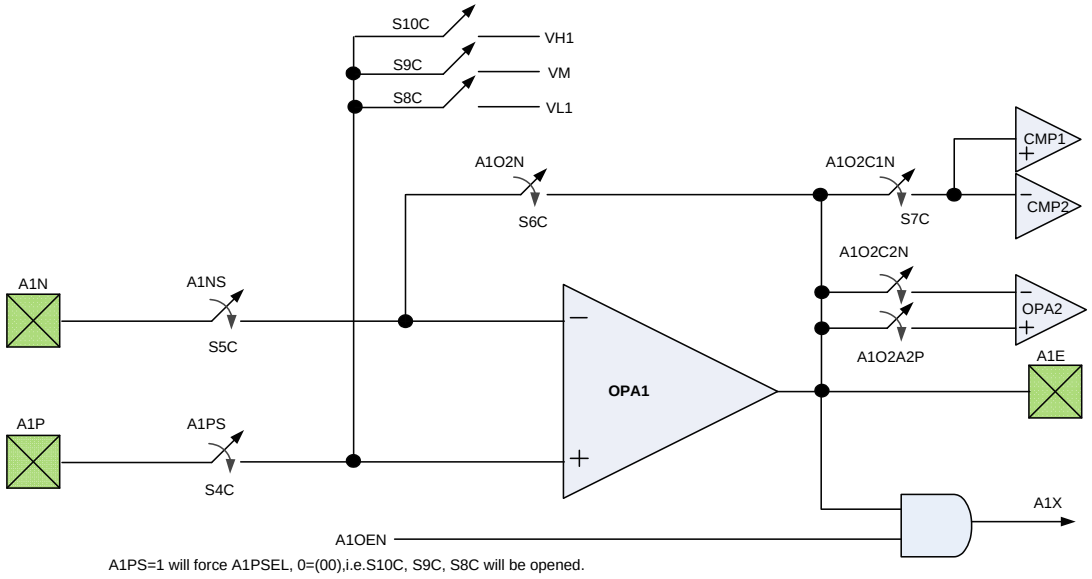
运算放大器功能

两个运算放大器内部可以特殊的方式连接，其输出也可以由内部连接至比较器。每个运算放大器都有自己的控制寄存器 OPA1C0、OPA1C1、OPA2C0、OPA2C1 及 OPA2C2，用于控制放大器使能 / 除能、校准程序及 OPA2 的可编程增益功能。

OPA1 开关控制

下图和表格为 OPA1 开关控制设置和相应的连接。注意，一些开关由硬件自动控制。例如：

- A1O2CIN=1 时 S7C 闭合，反之 S7C 断开
- A1PS=1 时，A1PSEL[1:0]=00，即 S10C、S9C、S8C 将处于断开状态
- A1EN=0 时，S6C 将由硬件断开，相关 I/O 脚用于其它功能



OPA1 开关控制

注：A1PS=1 时，A1PSEL[1:0]=00，即 S10C、S9C、S8C 将处于断开状态。

下表内容为 OPA1 控制寄存器设置与开关状态的关系。

OPA1C0、OPA1C1 寄存器中的 OPA1 控制位				开关描述				结果
A1PS	**A1NS**	**A1PSEL1, 0**	**A1O2N**	**S4C**	**S5C**	**S6C**	**S8C~S10C**	**OPA1 连接**
1	1	00	0	ON	ON	OFF	OFF	输入 =A1N, A1P
0	1	01	0	OFF	ON	OFF	S10C ON	输入 =A1N, VH1
0	1	10	0	OFF	ON	OFF	S9C ON	输入 =A1N, VM
0	1	11	0	OFF	ON	OFF	S8C ON	输入 =A1N, VL1
1	1	00	1	ON	ON	ON	OFF	输入 =A1N, A1P， 连接 A1N, A1E
1	0	00	1	ON	OFF	ON	OFF	输入 =A1P， OPA1 作单元增益缓冲器
0	1	01	0	OFF	ON	OFF	S10C ON	输入 =A1N, VH1
0	1	10	0	OFF	ON	OFF	S9C ON	输入 =A1N, VM
0	1	11	0	OFF	ON	OFF	S8C ON	输入 =A1N, VL1

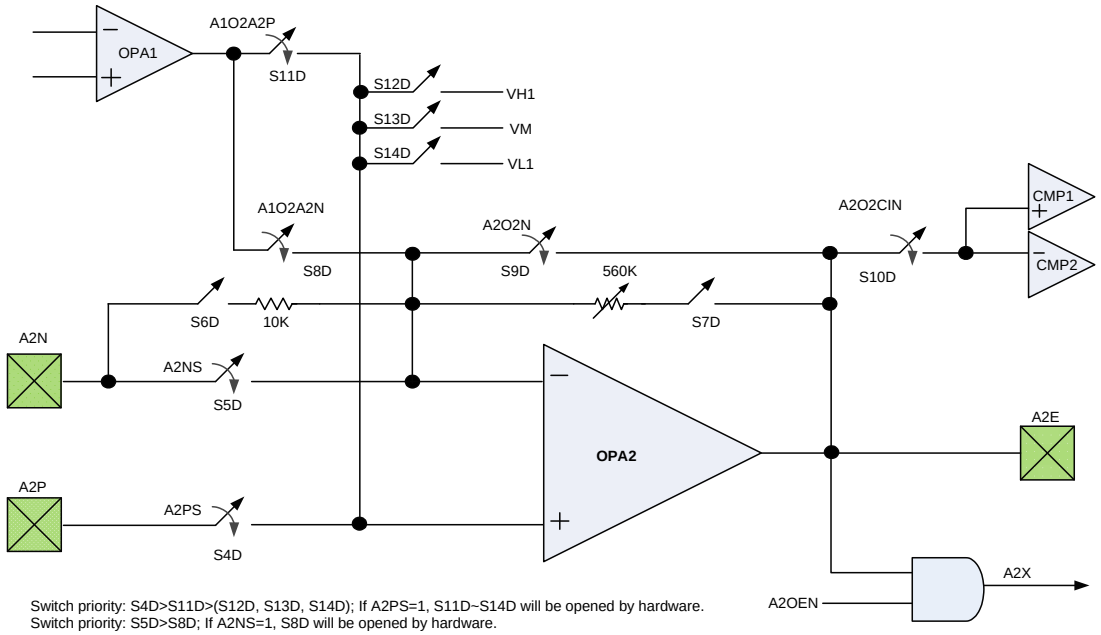
下表内容为 OPA1 I/O 设置及状态。

A1EN	A1NS	A1PS	描述
0	×	×	PA2、PA3、PA4 为普通 I/O 口
1	0	0	PA2、PA3 为普通 I/O 口，PA4 是 OPA1 A1E 输出脚
1	0	1	PA3 为普通 I/O 口，PA2 是 OPA1 A1P 输入脚， PA4 是 OPA1 A1E 输出脚
1	1	0	PA2 为普通 I/O 口，PA3 是 OPA1 A1N 输入脚， PA4 是 OPA1 A1E 输出脚
1	1	1	PA2 是 OPA1 A1P 输入，PA3 是 OPA1 A1N 输入脚， PA4 是 OPA1 A1E 输出脚

OPA2 开关控制

下图和表格为 OPA2 开关控制设置和相应的连接。注意，一些开关由硬件自动控制。例如：

- PGAEN=1 时，S6D、S7D 闭合，反之 S6D、S7D 断开。
- A2EN=0 时，S6D、S7D、S9D 将由硬件断开，相关 I/O 脚用于其它功能。



OPA2 开关控制

注：开关优先级：S4D>S11D>（S12D，S13D，S14D）；若 A2PS=1，S11D~S14D 将由硬件断开。
开关优先级：S5D>S8D；若 A2NS=1，S8D 将由硬件断开。

下表内容为 OPA2 控制寄存器设置与开关状态的关系。

OPA2C0、OPA2C1、OPA2C2 寄存器中的 OPA2 控制位							开关描述							结果
A2PS	A2NS	A2PSEL 1, 0	A1O2A2P	A1O2A2N	PGAEN	A2O2N	S4D	S5D	S6/ 7D	S8D	S9D	S11D	S12D ~S14D	OPA2 连接
1	1	00	0	0	0	0	ON	ON	OFF	OFF	OFF	OFF	OFF	正常模式，输入 =A2N, A2P
0	1	01	0	0	0	0	OFF	ON	OFF	OFF	OFF	OFF	S12D ON	输入 =A2N, VH1
0	1	10	0	0	0	0	OFF	ON	OFF	OFF	OFF	OFF	S13D ON	输入 =A2N, VM
0	1	11	0	0	0	0	OFF	ON	OFF	OFF	OFF	OFF	S14D ON	输入 =A2N, VL1
0	1	00	1	0	0	0	OFF	ON	OFF	OFF	OFF	ON	OFF	输入 =A2N, A1E
1	0	00	0	1	0	0	ON	OFF	OFF	ON	OFF	OFF	OFF	输入 =A1E, A2P
1	1	00	0	0	1	0	ON	ON	ON	OFF	OFF	OFF	OFF	输入 =A2N, A2P
1	0	00	0	0	1	0	ON	OFF	ON	OFF	OFF	OFF	OFF	输入 =A2N, A2P
1	0	00	0	0	0	1	ON	OFF	OFF	OFF	ON	OFF	OFF	输入 =A2P，OPA2 作缓冲器
0	0	01	0	0	0	1	OFF	OFF	OFF	OFF	OFF	OFF	S12D ON	输入 =VH1，OPA2 作缓冲器
0	0	10	0	0	0	1	OFF	OFF	OFF	OFF	OFF	OFF	S13D ON	输入 =VM，OPA2 作缓冲器
0	0	11	0	0	0	1	OFF	OFF	OFF	OFF	OFF	OFF	S14D ON	输入 =VL1，OPA2 作缓冲器

下表内容为 OPA2 I/O 设置及状态。

A2EN	PGAEN	A2NS	A2PS	描述
0	×	×	×	PA5、PA6、PA7 为普通 I/O 口
1	0	0	0	PA5、PA6 为普通 I/O 口，PA7 是 OPA2 A2E 输出脚
1	0	0	1	PA6 为普通 I/O 口，PA5 是 OPA2 A2P 输入脚，PA7 是 OPA2 A2E 输出脚
1	0	1	0	PA5 为普通 I/O 口，PA6 是 OPA2 A2N 输入脚，PA7 是 OPA2 A2E 输出脚
1	0	1	1	PA5 是 OPA2 A2P 输入脚，PA6 是 OPA2 A2N 输入脚，PA7 是 OPA2 A2E 输出脚
1	1	0	0	PA5 为普通 I/O 口，PA6 是 OPA2 A2N 输入脚，PA7 是 OPA2 A2E 输出脚
1	1	0	1	PA5 是 OPA2 A2P 输入脚，PA6 是 OPA2 A2N 输入脚，PA7 是 OPA2 A2E 输出脚
1	1	1	0	PA5 为普通 I/O 口，PA6 是 OPA2 A2N 输入脚且旁路电阻 R1 是 10kΩ，PA7 是 OPA2 A2E 输出脚
1	1	1	1	PA5 是 OPA2 A2P 输入脚，PA6 是 OPA2 A2N 输入脚且旁路电阻 R1 是 10kΩ，PA7 是 OPA2 A2E 输出脚

OPA2 输入输出控制表

比较器

单片机中含有两个模拟比较器。它们具有暂停、中断等功能，可通过寄存器进行灵活配置。比较器的引脚与普通 I/O 引脚共用，当比较器功能未使用时，此引脚可做普通引脚使用而不浪费 I/O 资源。除此之外，单片机还提供了比较器失调电压校准功能。

比较器操作

单片机包含两个比较器功能，用于比较两个模拟电压，基于它们的差值上提供一个输出。控制寄存器 CMP1C0、CMP1C1、CMP2C0 和 CMP2C1 可分别控制相应的内部比较器。比较器的输出可由各自寄存器的一位记录，并且在共用的 I/O 口上输出。此外，比较器功能还包括暂停控制。

比较器寄存器

比较器由控制寄存器 CMP1C0、CMP1C1、CMP2C0 和 CMP2C1 控制。这些寄存器用于控制比较器的使能 / 除能、选择输入路径、控制中断边沿和失调电压校准等功能。

CMP1C0 控制寄存器

Bit	7	6	5	4	3	2	1	0
Name	CMP1X	C1OFM	C1RS	C1OF4	C1OF3	C1OF2	C1OF1	C1OF0
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	1	0	0	0	0

Bit 7 **CMP1X**：比较器输出脚；正逻辑电平。此位为只读位。

Bit 6 **C1OFM**：输入失调电压校准模式 / 比较器模式选择位
0：比较器模式
1：输入失调电压校准模式
C1OFM=1 时，比较器输入端连接至 I/O 脚，即 CNPSEL 和 C1NSEL 强制置位为“1”，断开与放大器输出端的连接。

Bit 5 **C1RS**：比较器输入失调电压校准参考选择位
0：选择 C1N 作为参考输入脚
1：选择 CNP 作为参考输入脚

Bit 4~0 **C1OF4~C1OF0**：比较器输入失调电压校准控制位

CMP1C1 控制寄存器

Bit	7	6	5	4	3	2	1	0
Name	CNPSEL	—	—	—	C1INTEN	C1OUTEN	C1NSEL	CMP1EN
R/W	R/W	—	—	—	R/W	R/W	R/W	R/W
POR	1	—	—	—	0	0	1	0

Bit 7 **CNPSEL**：比较器同相输入控制位
0：选择 OPA 输出作为比较器输入
1：选择 CNP 脚作为比较器输入

Bit 6~4 未使用，读为“0”

Bit 3 **C1INTEN**：比较器 1 中断控制位
0：除能
1：使能

Bit 2 **C1OUTEN**：比较器 1 输出脚控制位
0：除能
1：使能

Bit 1 **C1NSEL**：比较器 1 反相输入端选择位
0：选择 VH0 作为输入端
1：选择 C1N 脚作为输入端

Bit 0 **CMP1EN**：比较器 1 使能 / 除能控制位
0：除能
1：使能

CMP2C0 控制寄存器

Bit	7	6	5	4	3	2	1	0
Name	CMP2X	C2OFM	C2RS	C2OF4	C2OF3	C2OF2	C2OF1	C2OF0
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	1	0	0	0	0

Bit 7 **CMP2X**：比较器输出脚；正逻辑电平。此为只读位。

Bit 6 **C2OFM**：输入失调电压校准模式 / 比较器模式选择位
0：比较器模式
1：输入失调电压校准模式
C2OFM=1 时，比较器输入端连接至 I/O 脚，CNPSEL 和 C2PSEL 强制置位为“1”，断开与放大器输出端的连接。

Bit 5 **C2RS**：比较器输入失调电压校准参考选择位
0：选择 C2P 作为参考输入端
1：选择 CNP 作为参考输入端

Bit 4~0 **C2OF4~C2OF0**：比较器输入失调电压校准控制位

CMP2C1 控制寄存器

Bit	7	6	5	4	3	2	1	0
Name	CMPES1	CMPES0	—	—	C2INTEN	C2OUTEN	C2PSEL	CMP2EN
R/W	R/W	R/W	—	—	R/W	R/W	R/W	R/W
POR	0	0	—	—	0	0	1	0

Bit 7~6 **CMPES1~ CMPES0**：中断边沿控制位
00：除能
01：上升沿触发
10：下降沿触发
11：双沿触发

Bit 5~4 未使用，读为“0”

Bit 3 **C2INTEN**：比较器 2 中断控制位
0：除能
1：使能

Bit 2 **C2OUTEN**：比较器 2 输出控制位
0：除能
1：使能

Bit 1 **C2PSEL**：比较器 2 反相输入选择位
0：选择 VL0 作为输入端
1：选择 C2P 作为输入端

Bit 0 **CMP2EN**：比较器 2 使能 / 除能控制位
0：除能
1：使能

比较器功能

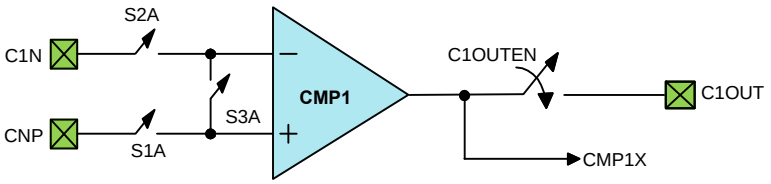
这两个比较器可以与运算放大器一起工作，也可单独使用。

每个比较器都可有其输入失调电压的共模校准方式。校准步骤如下：

1. 设置 C1OFM=1，进入失调电压校准模式（S3A 闭合）
2. 设置 C1RS，选择哪个引脚作为参考电压输入脚（S1A 或 S2A 闭合）
3. 调整 C1OF0~C1OF4 直到输出状态改变
4. 设置 C1OFM=0，进入比较器模式
5. 重复 (1)~(4) 步设置比较器 2

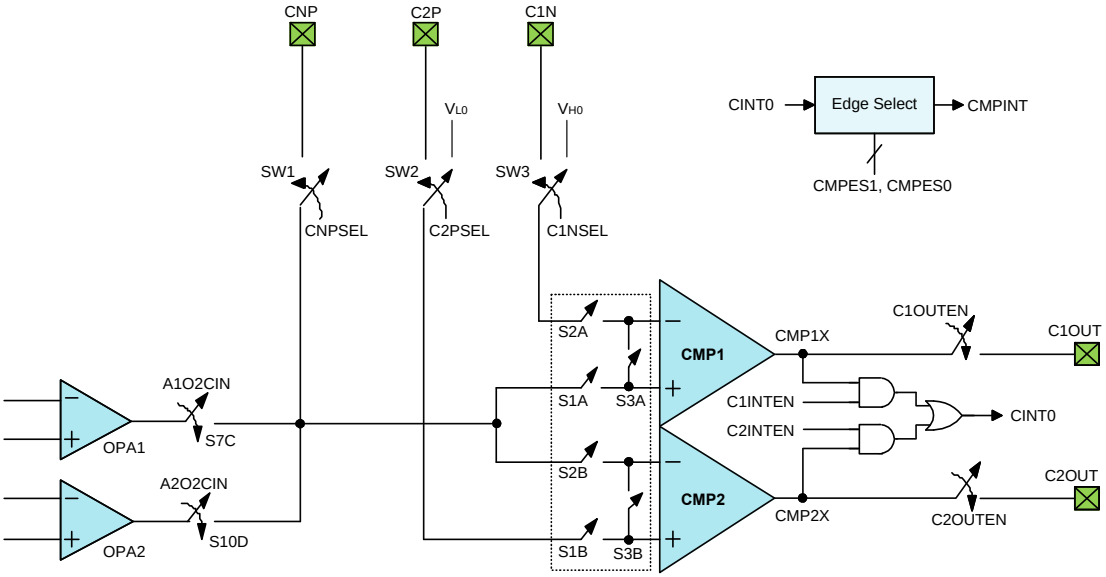
C1OFM	C1RS	S1A	S2A	S3A
0	x	ON	ON	OFF
1	0	OFF	ON	ON
1	1	ON	OFF	ON

"x" : don´t care



下图和表格为比较器开关控制设置和相应的连接。注意，一些开关由硬件自动控制。例如：

- 若比较器 1 处于校准模式（C1OFM=1）时，SW1、SW3 闭合。CNPSEL 和 C1NSEL 位由硬件置位为“1”且读出为“1”。失调电压校准后，CNPSEL 和 C1NSEL 位回到初始值状态
- 若比较器 2 处于校准模式（C2OFM=1）时，SW1、SW2 闭合。CNPSEL 和 C2PSEL 位由硬件置位为“1”且读出为“1”。失调电压校准后，CNPSEL 和 C2PSEL 位回到初始值状态
- 若 CNPSEL=1，A1O2CIN 和 A2O2CIN 将强制为“0”，即 SW1 闭合，S7C 和 S10D 断开
- 若 CNPSEL=0 和 A1O2CIN=1，A2O2CIN 将强制为“0”，即 S7C 闭合，S10D 断开



比较器开关控制

下表内容为比较器 1 I/O 设置。

CMP1EN	C1OUTEN	CNPSEL	C1NSEL	描述
0	×	×	×	PC5、PA0 和 PA1 为普通 I/O 口
1	0	0	0	CNP 来自 OPA1 或 OPA2 输出脚，C1N 来自 VH0 输入脚，PA1 为普通 I/O 口
1	0	0	1	CNP 来自 OPA1 或 OPA2 输出脚，C1N 来自 PC5 输入脚，PA1 为普通 I/O 口
1	0	1	0	CNP 来自 PA0 输入脚，C1N 来自 VH0 输入脚，PA1 为普通 I/O 口
1	0	1	1	CNP 来自 PA0 输入脚，C1N 来自 PC5 输入脚，PA1 为普通 I/O 口
1	1	0	0	CNP 来自 OPA1 或 OPA2 输出脚，C1N 来自 VH0 输入脚，PA1 为比较器输出口
1	1	0	1	CNP 来自 OPA1 或 OPA2 输出脚，C1N 来自 PC5 输入脚，PA1 为比较器输出口
1	1	1	0	CNP 来自 PA0 输入脚，C1N 来自 VH0 输入脚，PA1 是比较器输出脚
1	1	1	1	CNP 来自 PA0 输入脚，C1N 来自 PC5 输入脚，PA1 是比较器输出脚

下表内容为比较器 2 I/O 设置。

CMP2EN	C2OUTEN	CNPSEL	C2PSEL	描述
0	×	×	×	PC6、PA0 和 PA2 为普通 I/O 口
1	0	0	0	CNP 来自 OPA1 或 OPA2 输出脚，C2P 来自 VL0 输入脚，PA2 为普通 I/O 口
1	0	0	1	CNP 来自 OPA1 或 OPA2 输出脚，C2P 来自 PC6 输入脚，PA2 为普通 I/O 口
1	0	1	0	CNP 来自 PA0 输入脚，C2P 来自 VL0 输入脚，PA2 为普通 I/O 口
1	0	1	1	CNP 来自 PA0 输入脚，C2P 来自 PC6 输入脚，PA2 为普通 I/O 口
1	1	0	0	CNP 来自 OPA1 或 OPA2 输出脚，C2P 来自 VL0 输入脚，PA2 为比较器输出脚
1	1	0	1	CNP 来自 OPA1 或 OPA2 输出脚，C2P 来自 PC6 输入脚，PA2 为比较器输出脚
1	1	1	0	CNP 来自 PA0 输入脚，C2P 来自 VL0 输入脚，PA2 是比较器输出脚。
1	1	1	1	CNP 来自 PA0 输入脚，C2P 来自 PC6 输入脚，PA2 是比较器输出脚。

下表内容为比较器控制寄存器设置与开关状态的关系。

比较器 1、比较器 2 控制位					开关描述								结果
CNPSEL	**C2PSEL**	**C1NSEL**	**C1OFM**	**C1RS**	**SW1**	**SW2**	**SW3**	**S1A**	**S2A**	**S3A**	**S7C**	**S10D**	**连接**
1 (强制为 1)	×	1 (强制为 1)	1	1	ON CNP	×	ON C1N	ON	OFF	ON	OFF	OFF	共模输入 =CNP
1 (强制为 1)	×	1 (强制为 1)	1	0	ON CNP	×	ON C1N	OFF	ON	ON	OFF	OFF	共模输入 =C1N
1	0	1	0	×	ON	×	C1N	ON	ON	OFF	OFF	OFF	输入 =CNP, C1N
1	0	0	0	×	ON	×	VH	ON	ON	OFF	OFF	OFF	输入 =CNP, VH1
0	0	1	0	×	OFF	×	C1N	ON	ON	OFF	ON	OFF	输入 =A1E, C1N
0	0	1	0	×	OFF	×	C1N	ON	ON	OFF	OFF	ON	输入 =A2E, C1N

比较器开关控制

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。单片机提供多个外部中断和内部中断功能，外部中断由 INT0、INT1 和 $\overline{\text{PINT}}$ 引脚动作产生，而内部中断由各种内部功能，如定时 / 计数器溢出、时基、SIM、A/D 转换器、比较器、LVD 和 EEPROM 等产生。

中断寄存器

所有中断允许和请求标志均由数据存储器内 INTC0、INTC1、MFIC0 和 MFIC1 寄存器控制。通过控制相应的中断允许位可以使能或禁止相关的中断。如果有中断发生，相应的中断请求标志将被置位且 EMI 被清零将禁止其它中断。

INTC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	T0F	EIF1	EIF0	ET0I	EEI1	EEI0	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　未使用，读为“0”

Bit 6　**T0F**：定时 / 计数器 0 中断请求标志位
0：无效
1：有效

Bit 5　**EIF1**：外部中断 1 请求标志位
0：无效
1：有效

Bit 4　**EIF0**：外部中断 0 请求标志位
0：无效
1：有效

Bit 3　**ET0I**：定时 / 计数器 0 中断控制位
0：除能
1：使能

Bit 2　**EEI1**：外部中断 1 控制位
0：除能
1：使能

Bit 1　**EEI0**：外部中断 0 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	—	MFF	SIMF	T1F	—	EMFI	ESIM	ET1I
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7　未使用，读为“0”

Bit 6　**MFF**：多功能中断请求标志位
0：无效
1：有效

Bit 5　**SIMF**：SIM 请求标志位
0：无效
1：有效

Bit 4　**T1F**：定时 / 计数器 1 请求标志位
0：无效
1：有效

Bit 3　未使用，读为“0”

Bit 2　**EMFI**：多功能中断控制位
0：除能
1：使能

Bit 1　**ESIM**：SIM 中断控制位
0：除能
1：使能

Bit 0　**ET1I**：定时 / 计数器 1 控制位
0：除能
1：使能

MFIC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PEF	TB1F	TB0F	ADF	EPI	TB1E	TB0E	EADI
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 **PEF**：外围中断请求标志位
0：无效
1：有效

Bit 6 **TB1F**：时基中断 1 请求标志位
0：无效
1：有效

Bit 5 **TB0F**：时基中断 0 请求标志位
0：无效
1：有效

Bit 4 **ADF**：A/D 转换中断请求标志位
0：无效
1：有效

Bit 3 **EPI**：外围中断控制位
0：除能
1：使能

Bit 2 **TB1E**：时基中断 1 控制位
0：除能
1：使能

Bit 1 **TB0E**：时基中断 0 控制位
0：除能
1：使能

Bit 0 **EADI**：A/D 转换中断控制位
0：除能
1：使能

MFIC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	LVDF	E2F	CF	—	ELVDI	EE2I	ECI
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7　未使用，读为“0”

Bit 6　**LVDF**：LVD 中断请求标志位
0：无效
1：有效

Bit 5　**E2F**：EEPROM 中断请求标志位
0：无效
1：有效

Bit 4　**CF**：比较器中断请求标志位
0：无效
1：有效

Bit 3　未使用，读为“0”

Bit 2　**ELVDI**：LVD 中断控制位
0：除能
1：使能

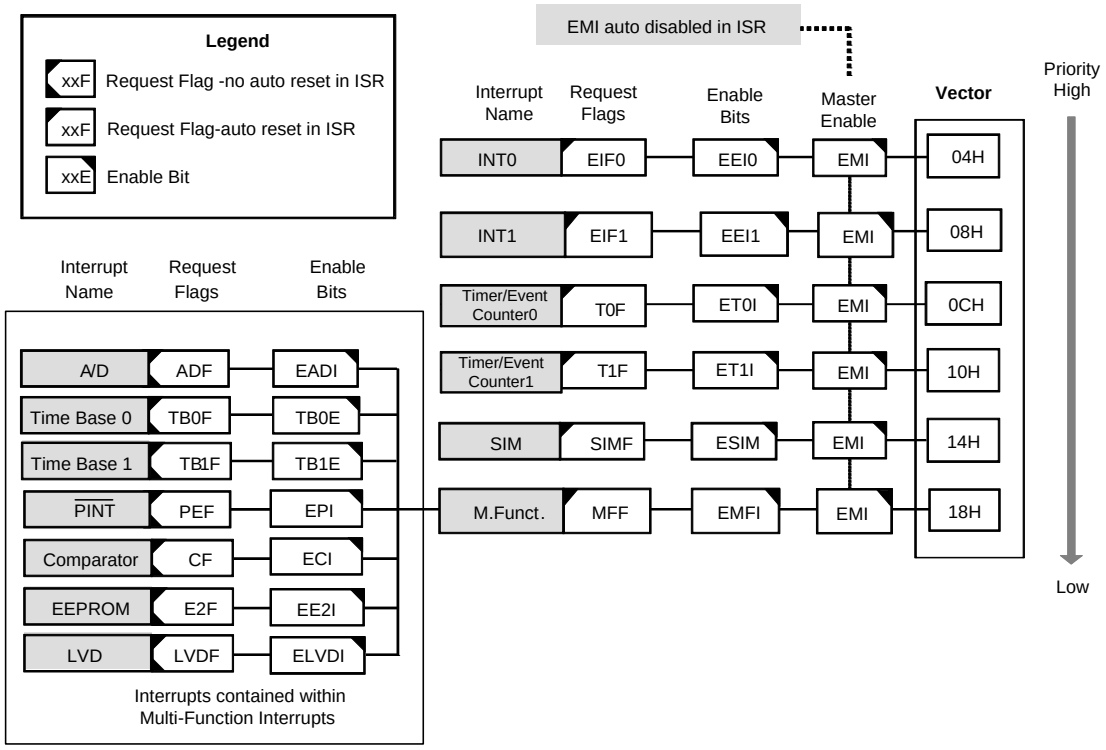
Bit 1　**EE2I**：EEPROM 中断控制位
0：除能
1：使能

Bit 0　**ECI**：比较器中断控制位
0：除能
1：使能

中断操作

如果相应的中断允许，定时 / 计数器溢出、时基 0/1、SIM 数据传输完成、A/D 转换结束、外部中断引脚触发、比较器输出、EEPROM 写 / 读周期完成或者 LVD 侦测到低电压都会通过设置相应的中断标志产生一个中断请求。当发生中断请求并且相应的中断使能位置位，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。中断向量处通常为跳转指令，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。



中断结构图

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。

中断优先级

中断发生在两个连续的 T2 脉冲上升沿之间，如果相应的中断请求被允许，中断将在后一个 T2 脉冲响应。下表指出在同时提出请求的情况下所提供的优先级。

中断源	优先级	向量
外部中断 0	1	04H
外部中断 1	2	08H
定时 / 计数器 0 溢出	3	0CH
定时 / 计数器 1 溢出	4	10H
SIM 中断	5	14H
多功能中断	6	18H

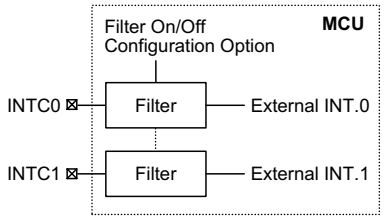
中断类型

A/D 转换中断、时基中断、外围中断、比较器中断、EEPROM 中断和 LVD 中断是同一个中断向量 18H。每个中断都有自己的中断标志位，但也有共同的中断标志 MFF。一旦多功能中断响应，MFF 标志位被硬件清为零，然而单独的中断触发多功能中断后，需通过软件清零其中断请求标志。

外部中断

要使能外部中断，总中断控制位 EMI、外部中断使能位 EEI0~EEI1 必须先被置位。此外，必须使用 INTEDGE 寄存器使能外部中断功能并选择触发沿类型。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，外部中断请求标志位 EIF0~EIF1 被置位时，将发生外部中断。外部中断引脚和普通 I/O 口共用，如果寄存器 INTC0 中相应的中断使能位被置位，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相应的位，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 EIF0~EIF1 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 INTEDGE 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者双沿触发都产生外部中断。注意 INTEDGE 也可以用来除能外部中断功能。



外部中断引脚与内部滤波器连接，以减少由外部中断引脚上的噪声和干扰产生的无用的中断。由于内部滤波电路会消耗一定的电量，可以通过配置选项关掉滤波功能。内部滤波电路会产生一定的功耗，通过配置选项可关闭该滤波器，这在功耗要求严格的应用中十分有用，但其输入信号则为原始信号。需要注意滤波器的开启 / 关闭配置选项，因为它不仅用于外部中断引脚，也应用于定时 / 计数器外部输入脚。单独的外部中断或定时 / 计数器引脚没有滤波器开启 / 关闭功能。

INTEDGE 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　未使用，读为“0”

Bit 3~2　**INT1S1~INT1S0**：外部中断 1 边沿触发类型选择位
00：除能
01：上升沿触发
10：下降沿触发
11：双沿触发

Bit 1~0　**INT0S1~INT0S0**：外部中断 0 边沿触发类型选择位
00：除能
01：上升沿触发
10：下降沿触发
11：双沿触发

外围中断

外围中断和外部中断工作方式类似，属于多功能中断。要使外围中断发生，总中断控制位 EMI，外围中断使能位 EPI 和多功能中断使能位 EMFI 必须先被置位。当 $\overline{\text{PINT}}$ 引脚出现一个下降沿跳变，此时外围中断请求标志位 PEF 被置位，外围发生中断。外围中断引脚与 PB5 脚共用，当中断使能，堆栈未满且外围中断脚出现一个下降沿跳变，单片机将调用位于多功能中断向量 18H 处的子程序。当响应外围中断服务子程序时，中断请求标志位 MFF 会被复位且 EMI 位会被清零以除能其它中断。PEF 标志位不会自动复位，由应用程序清零。

定时 / 计数器中断

要使定时 / 计数器 0 或者定时 / 计数器 1 中断发生，总中断控制位 EMI 和相应的内部中断使能位 ET0I 或 ET1I 必须先被置位。当定时 / 计数器溢出，相应的中断请求标志位 T0F 或 T1F 将置位并触发定时 / 计数器中断。中断使能，堆栈未满，当定时 / 计数器溢出发生中断时，将调用位于地址 0CH 或 10H 的子程序。当响应中断服务子程序时，中断请求标志位 T0F 或 T1F 会被自动复位且 EMI 位会被清零以除能其它中断。

SIM 中断

要使 SIM 中断发生，总中断控制位 EMI 和相应的中断使能位 ESIM 必须先被置位。当 SIM 接口完成传送或接收一个字节数据，相应的 SIM 中断请求标志位 SIMF 被置位，将会发生 SIM 中断。当中断使能，堆栈未满，SIM 接口完成传送或接收一个字节数据发生时，将调用位于地址 14H 的 SIM 中断子程序。当响应 SIM 中断服务子程序时，SIMF 会被自动复位且 EMI 位会被清零以除能其它中断。

多功能中断

这里提供了附加的中断，即多功能中断，与其它中断不同，它没有独立源，但由另外几个现有的中断源构成，即 A/D 转换中断、时基中断、外围中断、比较器中断、EEPROM 中断和 LVD 中断。

要使多功能中断发生，总中断控制位 EMI 和相应的中断使能位 EMFI 必须先被置位。当时基溢出，A/D 转换完成，外围中断、比较器输出状态改变、EEPROM 写 / 读周期结束或低电压条件发生时，相应的中断标志位 MFF 被置位，将会发生多功能中断。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，将调用位于地址 18H 处的子程序。当响应中断服务子程序时，多功能请求标志位 MFF 会被复位且 EMI 位会被清零以除能其它中断。但必须注意的是，来自多功能中断的源头的请求标志位，即时基中断，A/D 转换中断、比较器中断、EEPROM 中断、LVD 中断或外围中断的请求标志位不会自动复位，必须由应用程序清零。

A/D 中断

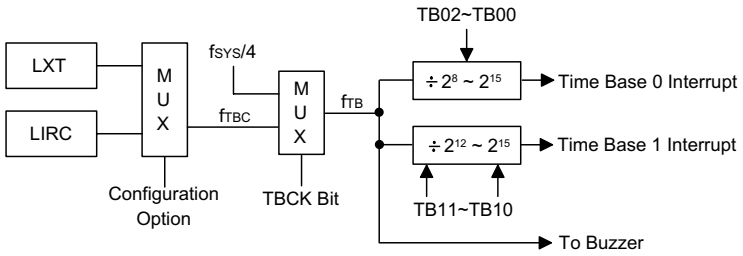
A/D 中断包含在多功能中断中。要使 A/D 转换中断发生，总中断控制位 EMI，A/D 中断使能位 EADI 和多功能中断使能位 EMFI 必须先被置位。当 A/D 转换结束，A/D 中断请求标志 ADF 被置位，将会发生 A/D 中断。当中断使能，堆栈未满和 A/D 转换结束，将调用位于地址 18H 的多功能中断子程序。当响应 A/D 转换中断服务子程序时，中断请求标志位 MFF 会被复位且 EMI 位会被清零以除能其它中断。由于 ADF 标志位不能自动复位，由应用程序清零。

时基中断

时基中断也包含在多功能中断中。要使时基中断发生，总中断控制位 EMI，多功能中断使能位 EMFI 和时基中断使能位 TB0E~TB1E 必须先被置位。当时基溢出，相应的时基中断请求标志位 TB0F~TB1F 被置位，则会发生时基中断。当中断使能，堆栈未满和时基溢出，将会调用位于地址 018H 处的多功能中断向量子程序。当响应时基中断子程序，EMI 位会被清零以除能其它中断，但是只有 MFF 中断请求标志位会被复位。TB0F~TB1F 标志位不会自动复位，必须通过应用程序来清零。

时基中断的目的是提供一个固定周期的中断信号，时钟源来自内部时钟源 f_{TB}。f_{TB} 输入时钟首先经过分频器，分频率由 TBC 寄存器选择以提供更长的时基中断周期。产生 f_{TB} 的时钟源可轮流控制时基中断周期，其由几种不同时钟源产生，具体请参考系统工作模式章节。

时基中断也可以作为可编程定时器，当时基溢出时，将置位时基中断标志，并通过多功能中断向量产生中断请求。



时基中断

TBC 寄存器

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

Bit 7 **TBON**：TB0 和 TB1 控制位
0：除能
1：使能

Bit 6 **TBCK**：f_{TB} 时钟源选择位
0：f_{TBC}
1：$f_{SYS}/4$

Bit 5~4 **TB11~TB10**：Time Base 1 溢出周期选择位
00：$4096/f_{TB}$
01：$8192/f_{TB}$
10：$16384/f_{TB}$
11：$32768/f_{TB}$

Bit 3 **LXTLP**：LXT 低功耗控制位
0：除能
1：使能

Bit 2~0 **TB02~TB00**：Time Base 0 溢出周期选择位
000：$256/f_{TB}$
001：$512/f_{TB}$
010：$1024/f_{TB}$
011：$2048/f_{TB}$
100：$4096/f_{TB}$
101：$8192/f_{TB}$
110：$16384/f_{TB}$
111：$32768/f_{TB}$

比较器中断

比较器中断包含于多功能中断中。比较器中断由内部两个比较器控制产生。要使比较器中断发生，总中断控制位 EMI，比较器中断使能位 ECI 和多功能中断使能位 EMFI 必须先被置位。当比较器输出状态发生改变时，相应的比较器中断请求标志位 CF 被置位，则会发生比较器中断。当中断使能、堆栈未满、比较器输出状态发生改变时，将会调用位于地址 018H 处的多功能中断向量子程序。当响应比较器中断子程序，EMI 位会被清零以除能其它中断，但是只有 MFF 中断请求标志位会被复位。CF 标志位不会自动复位，必须通过应用程序来清零。

EEPROM 中断

EEPROM 中断包含于多功能中断中。要使 EEPROM 中断发生，总中断控制位 EMI，EEPROM 中断使能位 EE2I 和多功能中断使能位 EMFI 必须先被置位。当 EEPROM 写 / 读周期完成时，相应的 EEPROM 中断请求标志位 E2F 被置位，则会发生 EEPROM 中断。当中断使能、堆栈未满、EEPROM 读或写周期完成时，将会调用位于地址 018H 处的多功能中断向量子程序。当响应 EEPROM 中断子程序，EMI 位会被清零以除能其它中断，但是只有 MFF 中断请求标志位会被复位。E2F 标志位不会自动复位，必须通过应用程序来清零。

LVD 中断

LVD 中断包含于多功能中断中。要使 LVD 中断发生，总中断控制位 EMI，LVD 中断使能位 ELVDI 和多功能中断使能位 EMFI 必须先被置位。当低电压侦测器检测到一个低电源电压时，相应的 LVD 中断请求标志位 LVDF 被置位，则会发生 LVD 中断。当中断使能、堆栈未满、低电压条件发生时，将会调用位于地址 018H 处的多功能中断向量子程序。当响应 LVD 中断子程序，EMI 位会被清零以除能其它中断，但是只有 MFF 中断请求标志位会被复位。LVDF 标志位不会自动复位，必须通过应用程序来清零。

中断唤醒功能

单片机处于空闲或休眠模式时，每种中断都具有唤醒单片机的功能。当中断请求标志由“0”变为“1”时，单片机即可被唤醒，与中断使能位的状态无关。因此，即使单片机处于空闲或休眠模式，系统振荡器停止运行，若有如外部中断脚上的中断边沿信号产生、低电压条件发生或比较器输出状态改变时，其相应的中断请求标志位置位，单片机也可以被唤醒。值得注意的是，应避免伪唤醒条件的产生。若要除能相应中断的唤醒功能，需在单片机进入空闲或休眠模式前使其中断请求标志位置位。中断唤醒功能与中断使能位的状态无关。

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。

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。

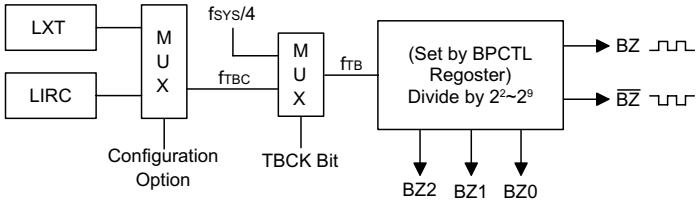
所有中断在休眠或空闲模式下都具有唤醒功能，当进入中断服务程序，系统仅将程序计数器的内容压入堆栈，如果中断服务程序会改变状态寄存器或其它的寄存器的内容而破坏控制流程，应事先将这些数据保存起来。

蜂鸣器

与可编程分频器的作用相似，单片机中蜂鸣器的功能在于提供可变频率的输出，以适用于像压电蜂鸣器驱动或其它需要精确频率输出的电路。引脚 BZ 和 $\overline{BZ}$ 形成互补对且和输入 / 输出引脚 PA6 和 PA7 共用。BPCTL 寄存器可以选择三种蜂鸣器选项的其中一个。第一个选项为 PA6 和 PA7 引脚都作为一般的输入 / 输出引脚使用，第二个选项为两个引脚都作为 BZ 和 $\overline{BZ}$ 蜂鸣器引脚使用，第三个选项为只有 PA6 引脚作为 BZ 蜂鸣器引脚使用而 PA7 引脚则保持一般的输入 / 输出引脚功能。要注意的是 $\overline{BZ}$ 引脚是 BZ 引脚的反向输出，可以提供更大的功率给外部接口，如蜂鸣器。

蜂鸣器由内部时钟源驱动，然后通过一个分频器，分频率由 BPCTL 寄存器的 BZ2~BZ0 位选择，可以选择范围从 $f_{TB}/2^2$ ~ $f_{TB}/2^9$ 的蜂鸣器频率。产生 f_{TB} 的时钟源控制着蜂鸣器的频率，这个时钟源有三种不同的来源，它们分别是 LXT 振荡器、LIRC 振荡器或系统振荡器的四分频，选择哪种 f_{TB} 时钟源由 f_{TB} 时钟源选项决定。注意，蜂鸣器的频率由 BPCTL 寄存器控制。

如果 BPCTL 寄存器把 PA6 和 PA7 设置为蜂鸣器输出功能即 BZ 和 $\overline{BZ}$，若要蜂鸣器正确工作，必须将 PAC 端口控制寄存器的 PAC.6 和 PAC.7 位设置为 0 以保证两个引脚作为输出。PA 数据寄存器的 PA.6 位必须置为高电平使蜂鸣器输出有效，如果置为低电平则 PA6 和 PA7 将保持低电平。这样，PA 数据寄存器的 PA.6 位用来作为蜂鸣器输出引脚 BZ 和 $\overline{BZ}$ 的一个开 / 关控制。PA 寄存器的 PA.7 数据位不能控制 $\overline{BZ}$ 引脚。



蜂鸣器功能

BPCTL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PMODE	PWM1EN	PWM0EN	BC1	BC0	BZ2	BZ1	BZ0
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 7~5　PWM 功能相关控制位，具体请参考脉冲宽度调节器章节

Bit 4~3　**BC1~BC0**：蜂鸣器或 I/O 选择控制位
00：PA7 为普通 I/O 口，PA6 为普通 I/O 口
01：PA7 为普通 I/O 口，PA6 为 BZ
10：保留
11：PA7 为 $\overline{BZ}$，PA6 为 BZ

Bit 2~0　**BZ2~BZ0**：蜂鸣器输出频率选择位
000：$f_{TB}/2^2$
001：$f_{TB}/2^3$
010：$f_{TB}/2^4$
011：$f_{TB}/2^5$
100：$f_{TB}/2^6$
101：$f_{TB}/2^7$
110：$f_{TB}/2^8$
111：$f_{TB}/2^9$

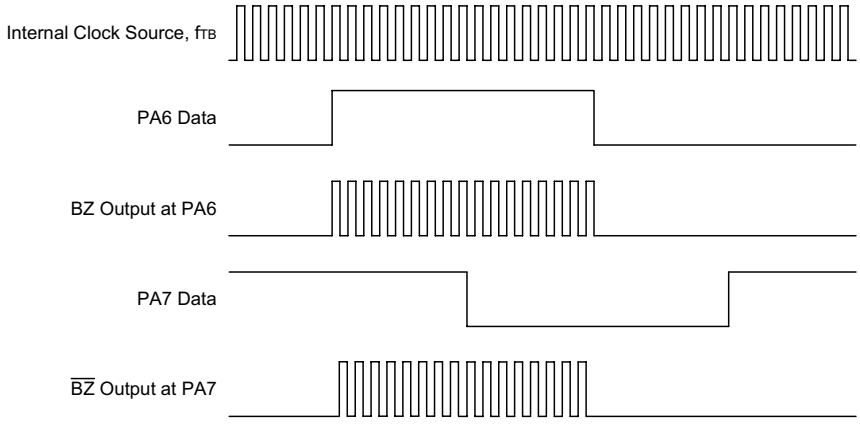
PAC 寄存器 PAC.6	PAC 寄存器 PAC.7	PA 数据寄存器 PA.6	PA 数据寄存器 PA.7	输出功能
0	0	1	×	PA6=BZ PA7=$\overline{BZ}$
0	0	0	×	PA6=“0” PA7=“0”
0	1	1	×	PA6=BZ PA7 作输入口
0	1	0	×	PA6=“0” PA7 作输入口
1	0	×	D	PA6 作输入口 PA7= D
1	1	×	×	PA6 作输入口 PA7 作输入口

注：“×”不相关，“D”数据“0”或“1”

PA6/PA7 引脚功能控制

如果仅选择 PA6 引脚功能为 BZ，而 PA7 为正常输入 / 输出引脚。因为 PA6 功能为 BZ，所以必须把 PAC 端口控制寄存器 PAC.6 位置为“0”以保证 PA6 作为输出口。PA 数据寄存器的 PA.6 位必须设置为高电平使得蜂鸣器输出有效，如果设置为低电平，即使已被设置为 BZ 蜂鸣器输出，PA6 扔将保持低电平。PA.6 位可作为 BZ 引脚的开 / 关控制。如果 PAC 端口控制寄存器的 PAC.6 位置为高电平，PA6 仍作为输入。

注意，如果端口控制寄存器已设置引脚为输入功能，则 BPCTL 设置的蜂鸣器选项将无效，即强制该引脚一直为输入状态。这使得该引脚既作为蜂鸣器引脚又是输入引脚，所以与 BPCTL 选项设置无关；引脚的实际功能可通应用程序设计适当的端口寄存器位而动态地改变。



蜂鸣器输出引脚控制

注：上面图表示 PA6 和 PA7 通过 BPCTL 寄存器设置为 BZ 和 $\overline{BZ}$ 蜂鸣器输出引脚的情形。两个引脚的端口控制寄存器必须已被设置为输出。对 PA7 引脚的数据设置不影响蜂鸣器的输出。

低电压检测 – LVD

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 个固定的电压参考点。LVDO 位被置位时低电压情况发生，若 LVDO 位为低表明 V_{DD} 电压工作在当前所设置低电压水平值之上。LVDEN 位用于控制低电压检测功能的开启 / 关闭，设置此位为高使能此功能，反之，关闭内部低电压检测电路。低电压检测会有一定的功耗，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	—	VLVD2	VLVD1	VLVD0
R/W	—	—	R	R/W	—	R/W	R/W	R/W
POR	—	—	0	0	—	0	0	0

Bit 7~6　未使用，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

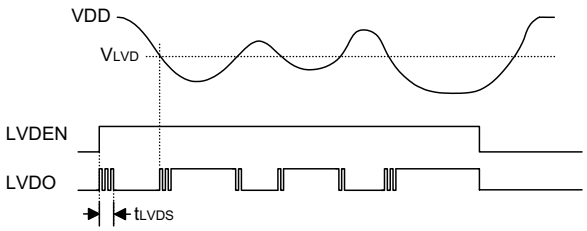
Bit 3　未使用，读为“0”

Bit 2~0　**VLVD2~VLVD0**：选择 LVD 电压
000：2.0V
001：2.2V
010：2.4V
011：2.7V
100：3.0V
101：3.3V
110：3.6V
111：4.4V

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~4.4V。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。

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 LVDF 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 LVDF 标志置为高。



LVD 操作

语音输出

语音控制

DAC 电路由语音控制寄存器控制。若 DAC 电路未开启，任何 DAH/DAL 输出均是无效的。通过控制 DACEN 位的高 / 低来使能 / 除能 D/A 转换电路。

语音输出和音量控制寄存器 – DAL，DAH，DACTRL

语音输出数据为 12 位，高 8 位存储在 DAH 寄存器中，低 4 位存储在 DAL 寄存器的高 4 位中，DAL 的低 4 位未使用，读为“0”。

音量有 8 级，由 DACTRL 寄存器中的高 3 位设置控制。DACEN 位为高时，D/A 转换器输出到 I/O 脚，其共用的 I/O 脚功能将除能。

DAL 数据寄存器

Bit	7	6	5	4	3	2	1	0
Name	D3	D2	D1	D0	—	—	—	—
R/W	R/W	R/W	R/W	R/W	—	—	—	—
POR	0	0	0	0	—	—	—	—

Bit 7~4 **D3~D0**：语音输出低 4 位

Bit 3~0 未使用，读为“0”

DAH 数据寄存器

Bit	7	6	5	4	3	2	1	0
Name	D11	D10	D9	D8	D7	D6	D5	D4
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 **D11~D4**：语音输出高 8 位

DACTRL 控制寄存器

Bit	7	6	5	4	3	2	1	0
Name	VOL2	VOL1	VOL0	—	—	—	—	DACEN
R/W	R/W	R/W	R/W	—	—	—	—	R/W
POR	0	0	0	—	—	—	—	0

Bit 7~5 **VOL2~VOL0**：D/A 转换器音量控制数据位
000：最小音量
111：最大音量

Bit 4~1 未使用，读为“0”

Bit 0 **DACEN**：D/A 转换器使能 / 除能
0：除能
1：使能

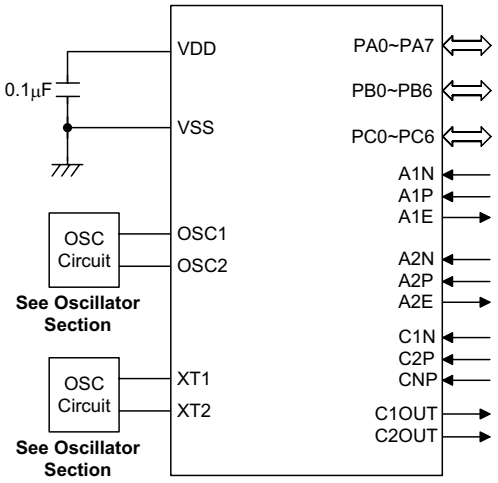
注：DACEN=1 时，D/A 转换器信号输出到 I/O 脚，此时与该引脚共用的 I/O 功能除能。

配置选项

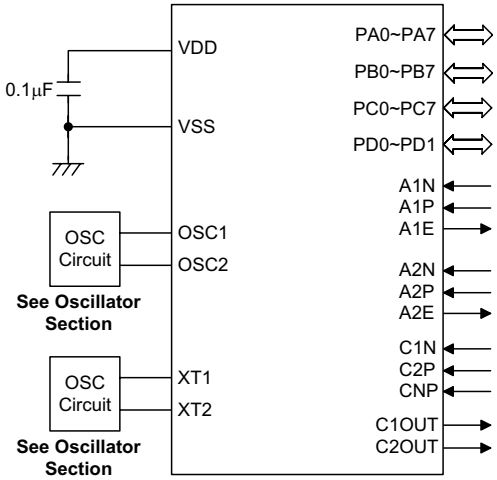
配置选项在烧写程序时写入芯片。通过 HT-IDE 的软件开发环境，使用者在开发过程中可以选择配置选项。当配置选项烧入单片机后，无法再通过应用程序修改。所有位需要按系统的需求定义，具体内容可参考下表：

编号	选项
振荡器选项	
1	OSC 类型选择：ERC/ 晶振 /HIRC/EC(外部时钟) （1）HXT(Filter ON) （2）ERC(Filter ON) （3）HIRC(Filter OFF) （4）EC(Filter OFF)
2	低速系统振荡器选择 – f_L：LXT、LIRC
3	高频振荡器 HIRC 时钟选择：910kHz、2MHz、4MHz、8MHz
4	f_S 时钟选择：f_{SUB} 或 $f_{SYS}/4$
5	HXT 模式选择：455kHz 或 1M~8MHz
看门狗选项	
6	WDT 使能或除能
7	CLR WDT 指令：1 或 2 条指令
LVR/LVD 选项	
8	LVR 功能：使能或除能
9	LVR 电压：2.1V/2.55V/3.15V/4.2V
RC filter	
10	RC filter(用于 TMR0/1&INT0/1)，使能或除能
SPI	
11	SIM：使能 / 除能
12	SPI_WCOL：使能 / 除能
13	SPI_CSEN：使能 / 除能 (1/0)，由软件控制 CSEN 功能
I^2C	
14	I^2C 去抖时间：无去抖，1 个系统时钟，2 个系统时钟
$\overline{RES}$	
15	I/O 或 $\overline{RES}$ 功能
锁选项	
16	所有锁
17	部分锁

应用电路



HT45F23A



HT45F24A

指令集

简介

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

为了更加容易理解各种各样的指令码，接下来按功能分组介绍它们。

指令周期

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

数据的传送

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

算术运算

算术运算和数据处理是大部分单片机应用所必需具备的能力，在盛群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 全都包含在盛群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盛群单片机的特性之一。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，盛群单片机允许在程序存储器中建立一个表格作为数据可直接存储的区域，只需要一组简易的指令即可对数据进行查表。

其它运算

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

指令集概要

下表中说明了按功能分类的指令集，用户可以将该表作为基本的指令参考。

惯例

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

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

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
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
TABRDC [m]	读取当前页的 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
其它指令			
NOP	空指令	1	无
CLR [m]	清除数据存储器	1注	无
SET [m]	置位数据存储器	1注	无
CLR WDT	清除看门狗定时器	1	TO, PDF

助记符	说明	指令周期	影响标志位
CLR WDT1	预清除看门狗定时器	1	TO, PDF
CLR WDT2	预清除看门狗定时器	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。
3. 对于“CLR WDT1”或“CLR WDT2”指令而言，TO 和 PDF 标志位也许会受执行结果影响，“CLR WDT1”和“CLR WDT2”被连续地执行后，TO 和 PDF 标志位会被清除，否则 TO 和 PDF 标志位保持不变

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

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

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

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

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

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 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

CLR WDT1	Preclear Watchdog Timer
指令说明	PDF 和 TO 标志位都被清 0。必须配合 CLR WDT2 一起使用清除 WDT 计时器。当程序仅执行 CLR WDT1，而没有执行 CLR WDT2 时，PDF 与 TO 保留原状态不变。
功能表示	WDT ← 00H TO & PDF ← 0
影响标志位	TO、PDF

CLR WDT2	Preclear Watchdog Timer
指令说明	PDF 和 TO 标志位都被清 0。必须配合 CLR WDT1 一起使用清除 WDT 计时器。当程序仅执行 CLR WDT2，而没有执行 CLR WDT1 时，PDF 与 TO 保留原状态不变。
功能表示	WDT ← 00H 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	PC ← PC+1
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相应，则这个中断将在返回主程序之前被相应。
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

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] Decrement data memory and place result in ACC,skip if 0

指令说明 将指定数据存储器内容减 1，判断是否为 0，如果为 0 则跳过下一条指令，此结果将存放到累加器，但指定数据存储器内容不变。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。

功能表示 ACC ← [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

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	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

TABRDC [m]	Read table (current 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（当前页）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

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

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

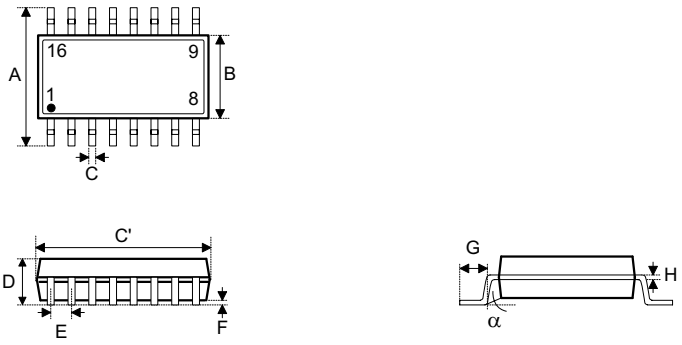
封装信息

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

封装信息的相关内容如下所示，点击可链接至 Holtek 网站相关信息页面。

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

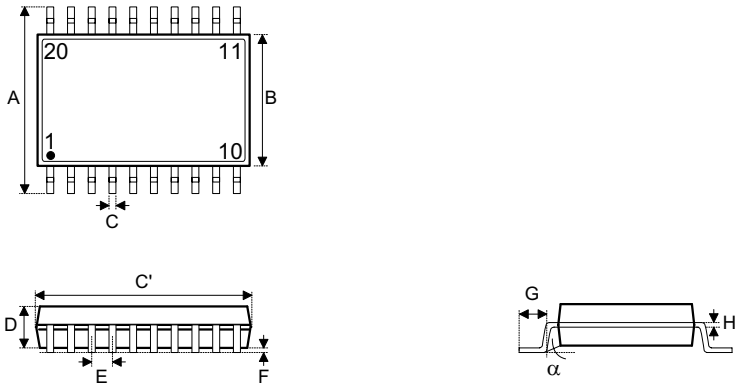
16-pin NSOP (150mil) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	典型值	最大值
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.012	—	0.020
C’	—	0.390 BSC	—
D	—	—	0.069
E	—	0.050 BSC	—
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

符号	尺寸 (单位：mm)		
	最小值	典型值	最大值
A	—	6.0 BSC	—
B	—	3.9 BSC	—
C	0.31	—	0.51
C’	—	9.9 BSC	—
D	—	—	1.75
E	—	1.27 BSC	—
F	0.10	—	0.25
G	0.40	—	1.27
H	0.10	—	0.25
α	0°	—	8°

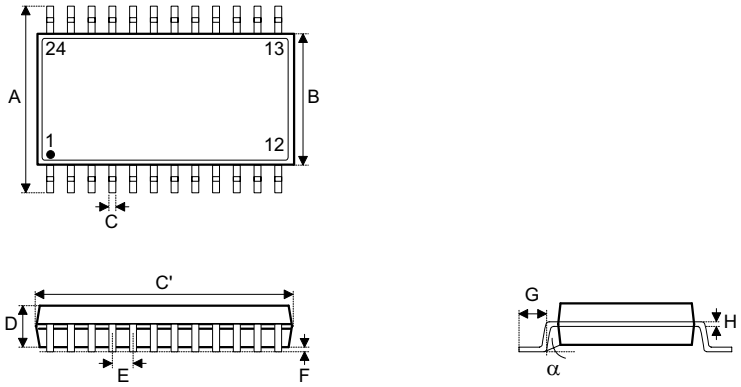
20-pin SSOP (150mil) 外形尺寸



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

符号	尺寸 (单位：mm)		
	最小值	典型值	最大值
A	—	6.0 BSC	—
B	—	3.9 BSC	—
C	0.20	—	0.30
C’	—	8.66 BSC	—
D	—	—	1.75
E	—	0.635 BSC	—
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°

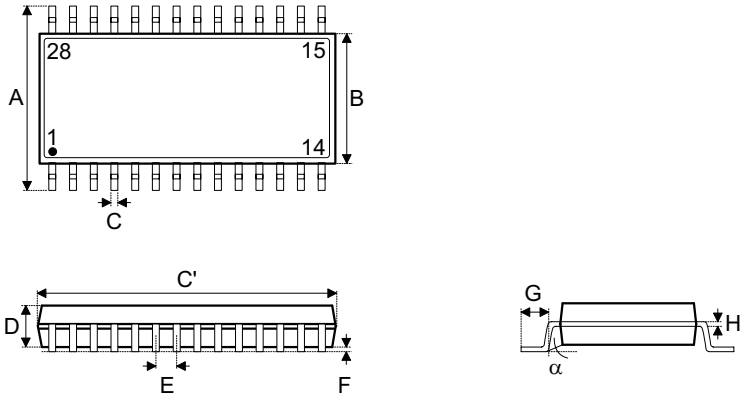
24-pin SSOP (150mil) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	典型值	最大值
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.008	—	0.012
C’	—	0.341 BSC	—
D	—	—	0.069
E	—	0.025 BSC	—
F	0.004	—	0.0098
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

符号	尺寸 (单位：mm)		
	最小值	典型值	最大值
A	—	6.0 BSC	—
B	—	3.9 BSC	—
C	0.20	—	0.30
C’	—	8.66 BSC	—
D	—	—	1.75
E	—	0.635 BSC	—
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°

28-pin SSOP (150mil) 外形尺寸



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

符号	尺寸 (单位：mm)		
	最小值	典型值	最大值
A	—	6.0 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°