可燃气体探测器 **Flash** 单片机

BA45F0096

版本 : V1.11　日期 : 2021-10-25

www.holtek.com

目录

特性

CPU 特性

- 工作电压：
  - ♦ f_{SYS}=8MHz：2.2V ~ 5.5V
- V_{DD}=5V，系统时钟为 8MHz 时，指令周期为 0.5μs
- 提供暂停和唤醒功能，以降低功耗
- 振荡器类型：
  - ♦ 内部高频 RC – HIRC
  - ♦ 内部 32kHz RC – LIRC
- 内部集成振荡器，无需外部元件
- 多种工作模式：正常、低速、空闲和休眠
- 所有指令都可在 1 或 2 个指令周期内完成
- 查表指令
- 61 条指令
- 2 层堆栈
- 位操作指令

周边特性

- Flash 程序存储器：1K×14
- RAM 数据存储器：64×8
- True EEPROM 存储器：32×8
- 看门狗定时器功能
- 14 个双向 I/O 口
- 1 个与 I/O 口共用的外部中断输入
- 多个定时器模块用于时间测量、捕捉输入、比较匹配输出、PWM 输出及单脉冲输出
- 双时基功能，可提供固定时间的中断信号
- 4 个外部通道 12-bit 分辨精度的 A/D 转换器
- 低电压复位功能
- Flash 程序存储器烧录可达 10,000 次
- Flash 程序存储器数据可保存 10 年以上
- True EEPROM 数据存储器烧录可达 100,000 次
- True EEPROM 数据存储器数据可保存 10 年以上
- 封装类型：16-pin NSOP

开发工具

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

https://www.holtek.com.cn/code-generator-for-gas-detector-ba45f0096

概述

该单片机是一款 A/D 型具有 8 位高性能精简指令集的 Flash 单片机，专门为可燃气体探测器产品而设计。此单片机可搭配使用安禧普公司的“可燃气体探测器参数平台”(Code Generator for Gas Detector)，开发者于此平台上勾选产品方案所需功能及参数，便可生成此单片机所需韧体，搭配 PCB 板便可快速进行产品功能验证与调校，可大幅缩短产品开发时间。

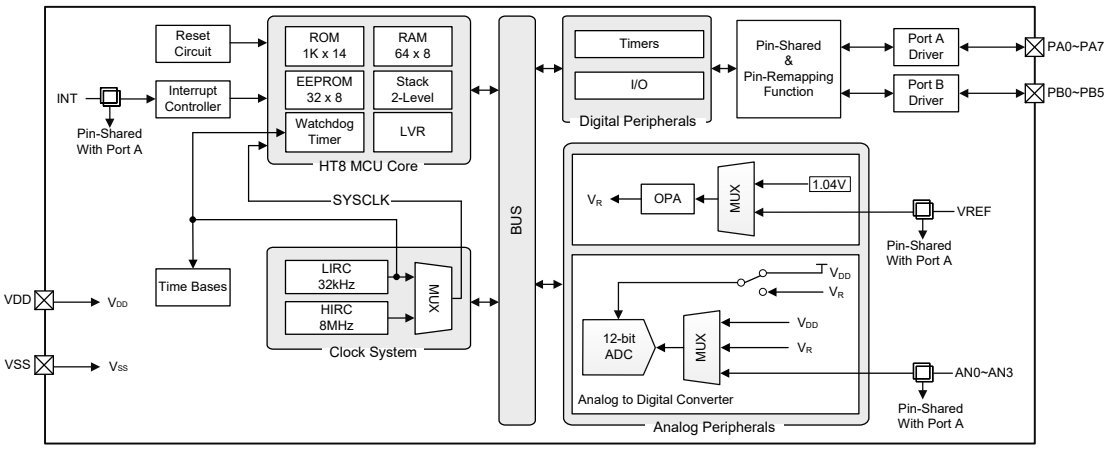
此单片机具有一系列功能和特性，其 Flash 存储器可多次编程的特性给用户提供了较大的方便。存储器方面，还包含了一个 RAM 数据存储器和一个可用于存储序号、校准数据等非易失性数据的 True EEPROM 存储器。

在模拟特性方面，这款单片机包含一个 4 通道 12 位 A/D 转换器 . 还带有多个使用灵活的定时器模块，可提供定时功能、脉冲产生功能、PWM 产生功能、捕捉输入及比较匹配输出功能。内部看门狗定时器和低电压复位等内部保护特性，外加优秀的抗干扰和 ESD 保护性能，确保单片机在恶劣的电磁干扰环境下可靠地运行。

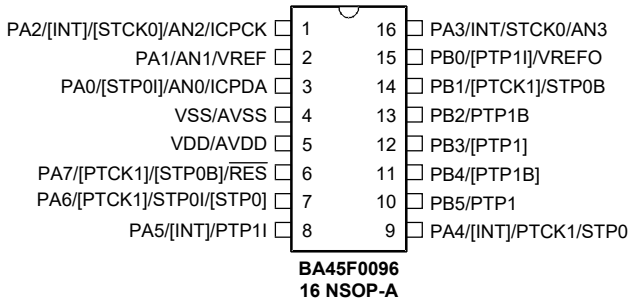
该单片机提供了高速和低速振荡器功能选项，且内建完整的系统振荡器，无需外围元器件。其在不同工作模式之间动态切换的能力，为用户提供了一个优化单片机操作和减少功耗的手段。

外加时基功能、I/O 使用灵活等其它特性，使这款单片机很适合可燃气体探测器的产品应用。

方框图



引脚图



注：括号内的引脚为可编程改变位置的引脚。

引脚说明

除了电源引脚及一些相关的变压控制引脚外，该单片机的所有引脚都以它们的端口名称进行标注，例如 PA0、PA1 等，用于描述这些引脚的数字输入 / 输出功能。然而，这些引脚也与其它功能共用，如模数转换器，定时器模块等。每个引脚的功能如下表所述，而引脚配置的详细内容见规格书其它章节。

引脚名称	功能	OPT	I/T	O/T	说明
PA0/[STP0I]/AN0/ICPDA	PA0	PAWU PAPU PASR	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	STP0I	PASR IFS0	ST	—	STM 输入
	AN0	PASR	AN	—	ADC 输入通道 0
	ICPDA	—	ST	CMOS	在线烧录数据 / 地址引脚
PA1/AN1/VREF	PA1	PAWU PAPU PASR	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	AN1	PASR	AN	—	ADC 输入通道 1
	VREF	PASR	AN	—	ADC 参考电压输入
PA2/[INT]/[STCK0]/AN2/ICPCK	PA2	PAWU PAPU PASR	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	INT	PASR IFS0	ST	—	外部中断输入
	STCK0	IFS0	ST	—	STM 时钟输入
	AN2	PASR	AN	—	ADC 输入通道 2
	ICPCK	—	ST	CMOS	在线烧录时钟引脚
PA3/INT/STCK0/AN3	PA3	PAWU PAPU PASR	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	INT	PASR IFS0	ST	—	外部中断输入
	STCK0	IFS0	ST	—	STM 时钟输入
	AN3	PASR	AN	—	ADC 输入通道 3

引脚名称	功能	OPT	I/T	O/T	说明
PA4/[INT]/PTCK1/STP0	PA4	PAWU PAPU PASR	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	INT	PASR IFS0	ST	—	外部中断输入
	PTCK1	PASR IFS0	ST	—	PTM 时钟输入
	STP0	PASR	—	CMOS	STM 输出
PA5/[INT]/PTP1I	PA5	PAWU PAPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	INT	PASR IFS0	ST	—	外部中断输入
	PTP1I	IFS0	ST	—	PTM 输入
PA6/[PTCK1]/STP0I/ [STP0]	PA6	PAWU PAPU PASR	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	PTCK1	PASR IFS0	ST	—	PTM 时钟输入
	STP0I	PASR IFS0	ST	—	STM 输入
	STP0	PASR	—	CMOS	STM 输出
PA7/[PTCK1]/[STP0B]/ $\overline{\text{RES}}$	PA7	PAWU PAPU PASR	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	PTCK1	PASR IFS0	ST	—	PTM 时钟输入
	STP0B	PASR	—	CMOS	STM 反相输出
	$\overline{\text{RES}}$	RSTC	ST	—	外部复位引脚
PB0/[PTP1I]/VREFO	PB0	PBPU PBSR	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTP1I	PBSR IFS0	ST	—	PTM 输入
	VREFO	PBSR	—	AN	ADC 参考电压输出
PB1/[PTCK1]/STP0B	PB1	PBPU PBSR	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTCK1	PBSR IFS0	ST	—	PTM 时钟输入
	STP0B	—	ST	CMOS	STM 反相输出
PB2/PTP1B	PB2	PBPU PBSR	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTP1B	PBSR	—	CMOS	PTM 反相输出
PB3/[PTP1]	PB3	PBPU PBSR	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTP1	PBSR	—	CMOS	PTM 输出

引脚名称	功能	OPT	I/T	O/T	说明
PB4/[PTP1B]	PB4	PBPU PBSR	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTP1B	PBSR	—	CMOS	PTM 反相输出
PB5/PTP1	PB5	PBPU PBSR	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTP1	PBSR	—	CMOS	PTM 输出
VDD/AVDD	VDD	—	PWR	—	数字电源电压
	AVDD	—	PWR	—	模拟电源电压
VSS/AVSS	VSS	—	PWR	—	数字地
	AVSS	—	PWR	—	模拟地

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

极限参数

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

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 (HIRC)	—	f_{SYS}=8MHz	2.2	—	5.5	V
I_{DD1}	工作电流，正常模式 $f_{SYS}=f_H$ (HIRC)	3V	无负载 , f_H=8MHz, ADC off, WDT 使能 , LVR 使能	—	1.0	2.0	mA
		5V		—	2.0	3.0	mA
I_{DD2}	工作电流，低速模式 $f_{SYS}=f_L$=LIRC	3V	无负载 , f_{SYS}=LIRC, ADC off, WDT 使能 , LVR 使能	—	20	30	μA
		5V		—	30	60	μA
I_{DD3}	工作电流，正常模式 f_H=8MHz (HIRC)	3V	无负载 , $f_{SYS}=f_H/2$, ADC off, WDT 使能 , LVR 使能	—	1.0	1.5	mA
		5V		—	1.5	2.0	mA
		3V	无负载 , $f_{SYS}=f_H/4$, ADC off, WDT 使能 , LVR 使能	—	0.9	1.3	mA
		5V		—	1.3	1.8	mA
		3V	无负载 , $f_{SYS}=f_H/8$, ADC off, WDT 使能 , LVR 使能	—	0.8	1.1	mA
		5V		—	1.1	1.6	mA
		3V	无负载 , $f_{SYS}=f_H/16$, ADC off, WDT 使能 , LVR 使能	—	0.7	1.0	mA
		5V		—	1.0	1.4	mA
		3V	无负载 , $f_{SYS}=f_H/32$, ADC off, WDT 使能 , LVR 使能	—	0.6	0.9	mA
		5V		—	0.9	1.2	mA
		3V	无负载 , $f_{SYS}=f_H/64$, ADC off, WDT 使能 , LVR 使能	—	0.5	0.8	mA
		5V		—	0.8	1.1	mA
I_{IDLE0}	IDLE0 模式，待机电流 (LIRC on)	3V	无负载 , ADC off, WDT 使能，LVR 除能	—	1.3	3.0	μA
		5V		—	5.0	10	μA
I_{IDLE1}	IDLE1 模式，待机电流 (HIRC)	3V	无负载 , ADC off, WDT 使能 , f_{SYS}=8MHz on	—	0.8	1.6	mA
		5V		—	1.0	2.0	mA
I_{SLEEP0}	SLEEP0 模式，待机电流 (LIRC off)	3V	无负载 , ADC off, WDT 除能 , LVR 除能	—	0.1	1.0	μA
		5V		—	0.3	2.0	μA
I_{SLEEP1}	SLEEP1 模式，待机电流 (LIRC on)	3V	无负载 , ADC off, WDT 使能 , LVR 除能	—	1.3	5.0	μA
		5V		—	2.2	10	μA
V_{IL1}	输入 / 输出口或除 $\overline{RES}$ 脚以外的输入引脚低电平输入电压	5V	—	0	—	1.5	V
		—	—	0	—	$0.2V_{DD}$	V
V_{IH1}	输入 / 输出口或除 $\overline{RES}$ 脚以外的输入引脚高电平输入电压	5V	—	3.5	—	5.0	V
		—	—	$0.8V_{DD}$	—	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
I_{OL}	I/O 口灌电流	3V	$V_{OL}=0.1V_{DD}$	18	36	—	mA
		5V	$V_{OL}=0.1V_{DD}$	40	80	—	mA
I_{OH}	I/O 口源电流	3V	$V_{OH}=0.9V_{DD}$	-3	-6	—	mA
		5V	$V_{OH}=0.9V_{DD}$	-7	-14	—	mA

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
R_{PH}	I/O 口上拉电阻	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ

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_{CPU}	工作时钟	2.2~5.5V	—	DC	—	8	MHz
f_{HIRC}	系统时钟 (HIRC)	3V/5V	Ta = 25°C	-2%	8	+2%	MHz
		3V/5V	Ta = 0°C~70°C	-5%	8	+5%	MHz
		2.2V~5.5V	Ta = 0°C~70°C	-8%	8	+8%	MHz
		2.2V~5.5V	Ta = -40°C~85°C	-12%	8	+12%	MHz
f_{LIRC}	系统时钟 (LIRC)	2.2V~5.5V	Ta = -40°C~85°C	8	32	50	kHz
t_{TIMER}	xTCKn, xTPnI 输入脉冲宽度	—	—	0.3	—	—	μs
t_{RES}	外部复位低电平脉宽	—	—	10	—	—	μs
t_{INT}	中断脉宽	—	—	0.3	—	—	μs
t_{EERD}	EEPROM 读周期	—	—	—	2	4	t_{SYS}
t_{EEWR}	EEPROM 写周期	—	—	—	2	5	ms
t_{SST}	系统启动时间 (从 HALT 中唤醒，HALT 状态下 f_{SYS} 关闭)	—	f_{SYS} = HIRC	16	—	—	t_{SYS}
			f_{SYS} = LIRC	2	—	—	
t_{RSTD}	系统复位延迟时间 (上电复位，LVR 复位，WDTC 软件复位)	—	—	25	50	100	ms
	系统复位延迟时间 (WDT 溢出复位，$\overline{RES}$ 复位)	—	—	8.3	16.7	33.3	ms

注： 1. t_{SYS}= 1/f_{SYS}

2. 为了保证 HIRC 振荡器的频率精度，VDD 与 VSS 间连接一个 0.1μF 的去耦电容，并尽可能接近芯片。

A/D 转换器电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
AV_{DD}	A/D 转换器工作电压	—	—	2.2	—	5.5	V
V_{ADI}	A/D 转换器输入电压	—	—	0	—	AV_{DD} / V_{REF}	V
V_{REF}	A/D 转换器参考电压	—	—	2	—	AV_{DD}	V
DNL	非线性微分误差	2.2V~2.7V	$V_{REF}=AV_{DD}=V_{DD}$ $t_{ADCK}=8\mu s$	—	±15	—	LSB
		2.7V~5.5V	$V_{REF}=AV_{DD}=V_{DD}$ $t_{ADCK}=0.5\mu s$	-3	—	+3	LSB
INL	非线性积分误差	2.2V~2.7V	$V_{REF}=AV_{DD}=V_{DD}$ $t_{ADCK}=8\mu s$	—	±16	—	LSB
		2.7V~5.5V	$V_{REF}=AV_{DD}=V_{DD}$ $t_{ADCK}=0.5\mu s$	-4	—	+4	LSB
I_{ADC}	打开 ADC 增加的功耗	3V	无负载 (t_{ADCK} =0.5μs)	—	1.0	2.0	mA
		5V	无负载 (t_{ADCK} =0.5μs)	—	1.5	3.0	mA
t_{ADCK}	ADC 时钟周期	2.2V~2.7V	—	8	—	10	μs
		2.7V~5.5V		0.5	—	10	μs
t_{ADC}	ADC 转换时间 (包括采样和保持时间)	—	12-bit A/D 转换	16	—	20	t_{ADCK}
t_{ADS}	ADC 采样时间	—	—	—	4	—	t_{ADCK}
t_{ON2ST}	ADC On-to-Start 时间	—	—	4	—	—	μs

运算放大器电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
AV_{DD}	OPA 工作电压	—	—	2.7	—	5.5	V
I_{OPA}	OPA 工作电流	5V	无负载	—	200	350	μA
V_{OPOS1}	OPA 输入失调电压	5V	—	-15	—	15	mV
V_{CM}	共模电压	5V	—	0.2	—	V_{DD}-1.4	V
PSRR	电源电压抑制比	5V	—	60	80	—	dB
CMRR	共模抑制比	5V	—	60	80	—	dB
SR	电压转换速率 +, 电压转换速率 -	5V	—	0.8	1.5	—	V/μs
GBW	增益带宽	5V	—	500	—	—	kHz
ERRG	OPA 增益错误	5V	Gain=1/2/3/4 若 OPA 输入电压 ≥0.2V	-5	Gain	+5	%

LVR 电气特性

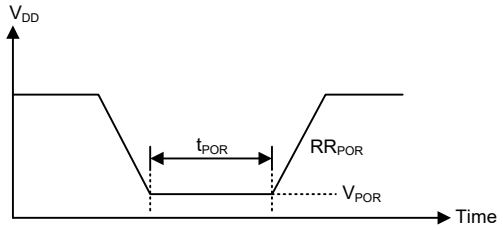
Ta=25°C

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	—	—	1.9	—	5.5	V
V_{LVR}	低电压复位电压	—	LVR 使能，选择 2.1V	-5%	2.1	+5%	V
V_{BG}	Reference Output with Buffer	—	T_J = +25°C@3.15V	-5%	1.04	+5%	V
t_{LVR}	产生 LVR 复位的低电压最短保持时间	—	—	160	320	640	μs

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_{POR}	上电复位电压速率	—	—	0.035	—	—	V/ms
t_{POR}	V_{DD} 保持为 V_{POR} 的最小时间	—	—	1	—	—	ms

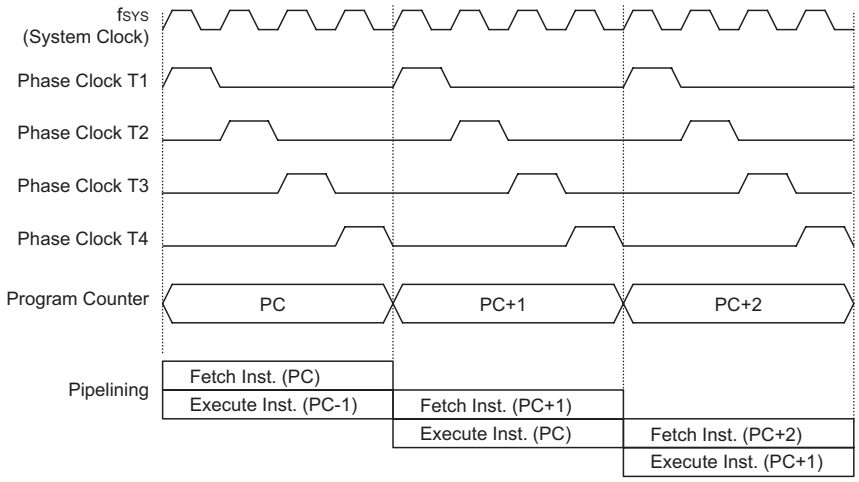


系统结构

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

时序和流水线结构

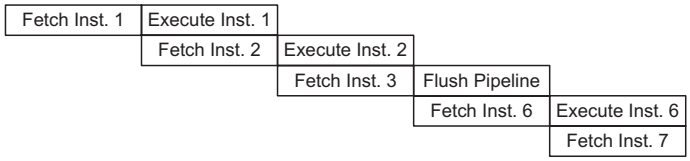
主系统时钟由 HIRC 或者 LIRC 振荡器提供，它被细分为 T1~T4 四个内部产生的非重叠时序。在 T1 时间，程序计数器自动加一并抓取一条新的指令。剩下的时间 T2~T4 完成译码和执行功能，因此，一个 T1~T4 时钟周期构成一个指令周期。虽然指令的抓取和执行发生在连续的指令周期，但单片机流水线结构会保证指令在一个指令周期内被有效执行。除非程序计数器的内容被改变，如子程序的调用或跳转，在这种情况下指令将需要多一个指令周期的时间去执行。



系统时序和流水线

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

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



指令捕捉

程序计数器

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

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

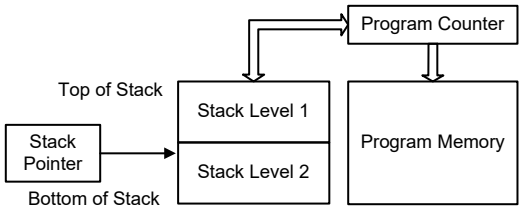
程序计数器	
程序计数器高字节	PCL 寄存器
PC9~PC8	PCL7~PCL0

程序计数器

程序计数器的低字节，即程序计数器的低字节寄存器 PCL，可以通过程序控制，且它是可以读取和写入的寄存器。通过直接写入数据到这个寄存器，一个程序短跳转可直接执行，然而只有低字节的操作是有效的，跳转被限制在存储器的当前页中，即 256 个存储器地址范围内，当这样一个程序跳转要执行时，会插入一个空指令周期。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。堆栈既不是数据部分也不是程序空间部分，而且它既不是可读取也不是可写入的。当前层由堆栈指针 (SP) 加以指示，同样也是不可读写的。在子程序调用或中断响应服务时，程序计数器的内容被压入到堆栈中。当子程序或中断响应结束时，返回指令 (RET 或 RETI) 使程序计数器从堆栈中重新得到它以前的值。当一个芯片复位后，堆栈指针将指向堆栈顶部。



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

若堆栈溢出，则首个存入堆栈的程序计数器数据将会丢失。

算术逻辑单元 – ALU

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

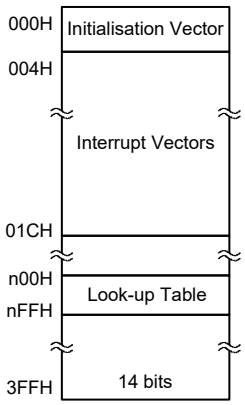
- 算术运算：ADD，ADDM，ADC，ADCM，SUB，SUBM，SBC，SBCM，DAA
- 逻辑运算：AND，OR，XOR，ANDM，ORM，XORM，CPL，CPLA
- 移位运算：RRA，RR，RRCA，RRC，RLA，RL，RLCA，RLC
- 递增和递减：INCA，INC，DECA，DEC
- 分支判断：JMP，SZ，SZA，SNZ，SIZ，SDZ，SIZA，SDZA，CALL，RET，RETI

Flash 程序存储器

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

结构

程序存储器的容量为 1K×14 位，程序存储器用程序计数器来寻址，其中也包含数据、表格和中断入口。数据表格可以设定在程序存储器的任何地址，由表格指针来寻址。



程序存储器结构

特殊向量

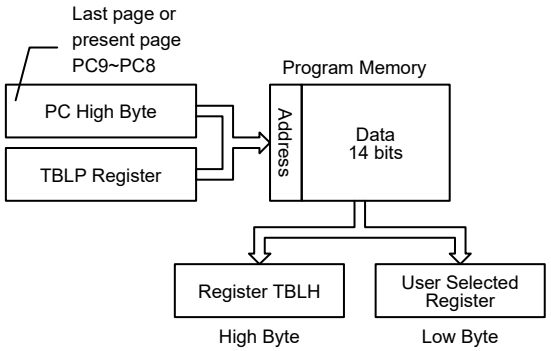
程序存储器内部某些地址保留用做诸如复位和中断入口等特殊用途。地址 000H 是芯片复位后的程序起始地址。在芯片复位之后，程序将跳到这个地址并开始执行。

查表

程序存储器中的任何地址都可以定义成一个表格，以便储存固定的数据。使用表格时，表格指针必须先行设定，其方式是将表格的地址放在表格指针寄存器 TBLP 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]”指令分别从程序存储器当前页或最后页查表读取。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：



查表范例

以下范例说明表格指针和表格数据如何被定义和执行。这个例子使用的表格数据用 ORG 伪指令储存在存储器中。ORG 指令的值“300H”指向的地址是 1K 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 306H，即最后一页起始地址后的第六个地址。值得注意的是，假如“TABRDL [m]”指令被使用，则表格指针指向最后一页由 TBLP 指定的地址。在这个例子中，表格数据的高字节等于零，而当“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。

表格读取程序举例

```
tempreg1 db ?       ; temporary register #1
tempreg2 db ?       ; temporary register #2
:
:
mov a,06h           ; initialise low table pointer - note that this address
                    ; is referenced
mov tblp,a          ; to the last page or present page
:
:
tabrdl tempreg1     ; transfers value in table referenced by table pointer
                    ; data at program memory address “306H” transferred to
                    ; tempreg1 and TBLH
dec tblp            ; reduce value of table pointer by one
tabrdl tempreg2     ; transfers value in table referenced by table pointer
                    ; data at program memory address “305H” transferred to
                    ; tempreg2 and TBLH in this example the data “1AH” is
                    ; transferred to tempreg1 and data “0FH” to register
                    ; tempreg2
:
:
org 300h            ; sets initial address of program memory
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
:
:
```

在线烧录

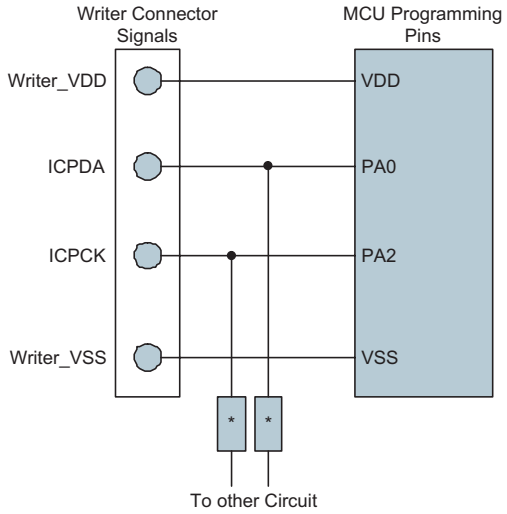
Flash 型程序存储器提供用户便利地对同一芯片进行程序的更新和修改。

另外，Holtek 单片机提供 4 线接口的在线烧录方式。用户可将进行过烧录或未经过烧录的单片机芯片连同电路板一起制成，最后阶段进行程序的更新和程序的烧写，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。

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

芯片内部程序存储器可以通过 4 线的接口在线进行烧录。其中 PA0 用于数据串行下载或上传、PA2 用于串行时钟、另外两条用于提供电源。芯片在线烧写的详细使用说明超出此文档的描述范围，将由专门的参考文献提供。

在烧录过程中，烧录器会控制 ICPDA 和 ICPCK 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。



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

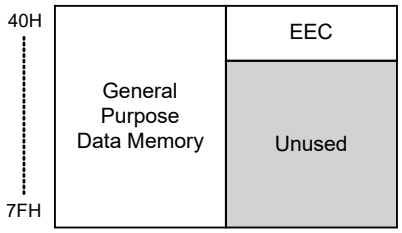
数据存储器分为 2 个部分，第一部分是特殊功能数据存储器。这些寄存器有固定的地址且与单片机的正确操作密切相关。大多特殊功能寄存器都可在程序控制下直接读取和写入，但有些被加以保护而不对用户开放。第二部分数据存储器是做一般用途使用，都可在程序控制下进行读取和写入。

总的数据存储器被分为 2 个区。大部分特殊功能数据寄存器均可在所有 Bank 被访问，除了 EEC 寄存器只位于 Bank 1 的“40H”地址。切换不同区域可通过设置区域指针实现。所有单片机的数据存储器的起始地址都是“00H”。

容量	Bank0	Bank1
64×8	40H~7FH	仅有 EEC 寄存器

通用功能数据存储器

所有单片机的程序需要一个读 / 写的存储区，让临时数据可以被储存和再使用。该区域就是通用数据存储器。此单片机的通用功能数据存储器的容量为 64×8 字节，分布在 Bank0。用户可对此区域进行读取和写入操作。使用“SET [m].i”和“CLR [m].i”指令可对个别位进行设置或复位的操作，方便用户在数据存储器中进行位操作。



通用功能数据存储器

特殊功能数据存储器

这个区域的数据存储器是存放特殊寄存器的，它和单片机的正确操作密切相关。大多数寄存器是可以读取和写入，只有一些是被写保护而只可读取的，相关的介绍请参考特殊功能寄存器的部分。需注意，任何读取指令对于未定义的地址读取将返回“00H”的值。

	Bank0 & Bank1		Bank0 & Bank1
00H	IAR0	20H	SADOL
01H	MP0	21H	SADOH
02H	IAR1	22H	SADC0
03H	MP1	23H	SADC1
04H	BP	24H	SADC2
05H	ACC	25H	RSTC
06H	PCL	26H	PASR
07H	TBLP	27H	PBSR
08H	TBLH	28H	STM0C0
09H	Unused	29H	STM0C1
0AH	STATUS	2AH	STM0DL
0BH	SMOD	2BH	STM0DH
0CH	Unused	2CH	STM0AL
0DH	INTEG	2DH	STM0AH
0EH	INTC0	2EH	Unused
0FH	INTC1	2FH	PB
10H	Unused	30H	PBC
11H	MFI0	31H	PBPU
12H	MFI1	32H	PTM1C0
13H	Unused	33H	PTM1C1
14H	PA	34H	PTM1DL
15H	PAC	35H	PTM1DH
16H	PAPU	36H	PTM1AL
17H	PAWU	37H	PTM1AH
18H	IFS0	38H	PTM1RPL
19H	WDTC	39H	PTM1RPH
1AH	Unused	3AH ~ 3FH	Unused
1BH	TBC		
1CH	SMOD1		
1DH	Unused		
1EH	EEA		
1FH	EED		

: Unused, read as “00”

特殊功能数据存储器结构

特殊功能寄存器

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

间接寻址寄存器 – IAR0, IAR1

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，IAR0 和 MP0 可以访问 Bank 0，而 IAR1 和 MP1 可以访问所有 Bank。因为这些间接寻址寄存器不是实际存在的，直接读取将返回“00H”的结果，而直接写入此寄存器则不做任何操作。

存储器指针 – MP0, MP1

该单片机提供两个存储器指针，即 MP0 和 MP1。由于这些指针在数据存储器中能像普通的寄存器一般被操作，因此提供了一个寻址和数据追踪的有效方法。当对间接寻址寄存器进行任何操作时，单片机指向的实际地址是由存储器指针所指定的地址。MP0，IAR0 用于访问 Bank 0，而 MP1 和 IAR1 可通过 BP 寄存器访问所有的 Bank。直接寻址仅可以用在 Bank 0 中，其它所有 Bank 只能使用 MP1 和 IAR1 进行间接寻址。

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

间接寻址程序举例

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

对于此单片机，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。BP 指针的第 0 位用于选择数据存储器的 Bank 0 或 Bank 1。

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

- **BP 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	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, TBLH

这两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 为表格指针，指向表格数据存储的地址。它的值必须在任何表格读取指令执行前加以设定，由于它的值可以被如“INC”或“DEC”的指令所改变，这就提供了一种简单的方法对表格数据进行读取。表格读取数据指令执行之后，表格数据高字节存储在 TBLH 中。其中要注意的是，表格数据低字节会被传送到使用者指定的地址。

状态寄存器 – STATUS

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

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

Z、OV、AC 和 C 标志位通常反映最近运算的状态。

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

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

- **STATUS 寄存器**

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

“x”：未知

Bit 7~6　未使用，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 标志位也受循环移位指令的影响。

EEPROM 数据存储器

此单片机的一个特性是内建 EEPROM 数据存储器。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。这种存储区扩展了数据存储器空间，对设计者来说增加了许多新的应用机会。EEPROM 可以用来存储产品编号、校准值、用户特定数据、系统配置参数或其它产品信息等。EEPROM 的数据读取和写入过程也会变的更简单。

EEPROM 数据存储器结构

此单片机 EEPROM 数据存储器容量为 32×8。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。使用一个地址和数据寄存器以及仅位于 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 位于所有 Bank 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	—	—	—	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	—	—	—	D4	D3	D2	D1	D0
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	0	0	0	0	0

Bit 7~5 未定义，读为“0”

Bit 4~0 数据 EEPROM 地址
数据 EEPROM 地址 Bit4 ~Bit 0

● **EED 寄存器**

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

Bit 7~0 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

写数据至 EEPROM，EEPROM 中写入数据的地址要先放入 EEA 寄存器中，写入的数据需存入 EED 寄存器中。EEC 寄存器中的写使能位 WREN 先置为高以

使能写功能，然后 EEC 寄存器中的 WR 位需立即置高以开始写操作，这两条指令必须在两个指令周期内连续执行。总中断位 EMI 在写周期开始前应当被清零，写周期开始后再将其使能。若 WR 位已置为高而 WREN 位还未被设置则不能开始写操作。由于控制 EEPROM 写周期是一个内部时钟，与单片机的系统时钟异步，所以数据写入 EEPROM 的时间将有所延迟。可通过轮询 EEC 寄存器中的 WR 位或判断 EEPROM 中断以侦测写周期是否完成。若写周期完成，WR 位将自动清除为“0”，通知用户数据已写入 EEPROM。因此，应用程序将轮询 WR 位以确定写周期是否结束。

写保护

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

EEPROM 中断

EEPROM 写周期结束后将产生 EEPROM 写中断，需先通过设置相关中断寄存器的 DEE 位使能 EEPROM 中断。当 EEPROM 写周期结束，DEF 请求标志位将被置位。若 EEPROM 和总中断使能且堆栈未满的情况下将跳转到相应的中断向量中执行。当中断被响应，EEPROM 中断标志位 DEF 将自动清零，总中断 EMI 也将被自动清零以除能其它中断。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。

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

程序举例

从 EEPROM 中读取数据 — 轮询法

```
MOV A, EEPROM_ADRES        ; user defined address
MOV EEA, A
MOV A, 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
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-executed
                           ; immediately after set WREN bit
SET EMI
BACK:
SZ IAR1.2                  ; check for write cycle end
JMP BACK
CLR IAR1                   ; disable EEPROM 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较佳的优化。振荡器选择和操作是通过寄存器完成的。

振荡器概述

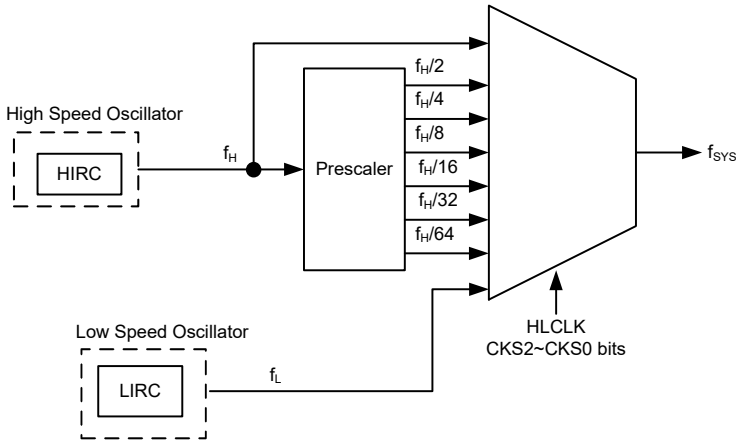
振荡器除了作为系统时钟源，还作为看门狗定时器和时基功能的时钟源。集成的两个内部振荡器不需要任何外围器件。它们提供的高速和低速系统振荡器具有较宽的频率范围。较高频率的振荡器提供更高的性能，但要求有更高的功率，反之亦然。动态切换快慢系统时钟的能力使单片机具有灵活而优化的性能 / 功耗比，此特性对功耗敏感的应用领域尤为重要。

类型	名称	频率
内部高速 RC	HIRC	8MHz
内部低速 RC	LIRC	32kHz

振荡器类型

系统时钟配置

此单片机有两个系统振荡器，包括一个高速振荡器和一个低速振荡器。高速振荡器为内部 8MHz RC 振荡器，低速振荡器为内部 32kHz 低速振荡器。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。



系统时钟配置

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 振荡器具有固定的 8MHz 频率。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡频率因 V_{DD}、温度以及芯片制成工艺不同的影响较大程度地降低。在电源电压为 5V 及温度为 25°C 的条件下， 此 8MHz 固定频率的容差为 2%。

内部 32kHz 振荡器 – LIRC

内部 32kHz 系统振荡器是一个低频振荡器。这种单片机有一个完全集成 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较大程度地降低。

辅助振荡器

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

工作模式和系统时钟

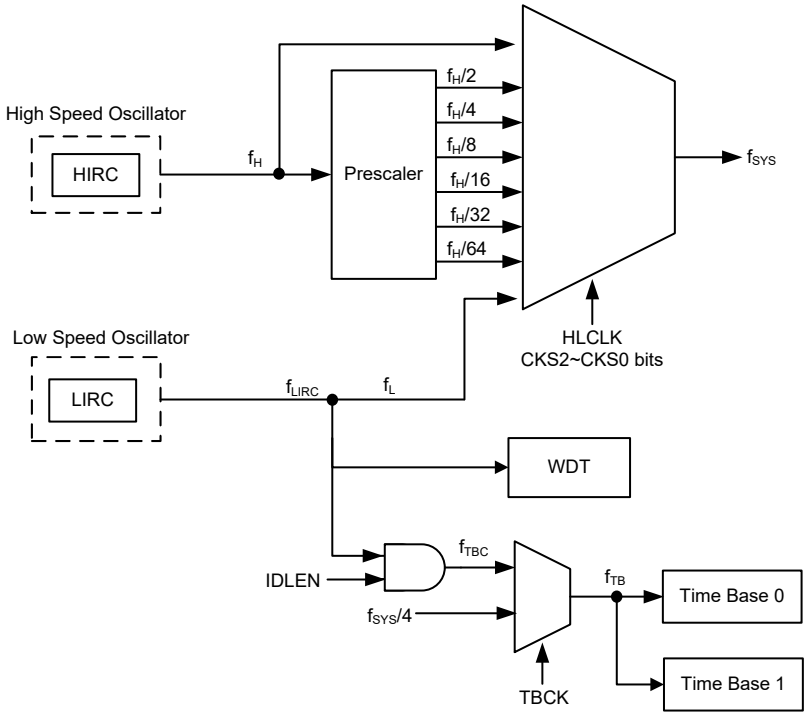
现今的应用要求单片机具有较高的性能及尽可能低的功耗，这种矛盾的要求在便携式电池供电的应用领域尤为明显。高性能所需要的高速时钟将增加功耗，反之亦然。此单片机提供高、低速两种时钟源，它们之间可以动态切换，用户可通过优化单片机操作来获得较佳性能 / 功耗比。

系统时钟

单片机为 CPU 和外围功能操作提供了两种不同的时钟源。用户使用寄存器编程可获取多种时钟，进而使系统时钟获取较大的应用性能。

主系统时钟可来自高频时钟源 f_H 或低频时钟源 f_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 HIRC 振荡器，低频系统时钟源来自内部时钟 f_L，低频时钟来自 LIRC 振荡器。其它系统时钟还有高速系统振荡器的分频 $f_H/2$~$f_H/64$。

另外一个内部时钟用于外围电路，时基时钟 f_{TBC}。此时钟来自 LIRC 振荡器，用于时基中断功能和 TMs 的时钟源。



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~$f_H/64$ 的频率。

系统工作模式

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

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

正常模式

顾名思义，这是主要的工作模式之一，单片机的所有功能均可在此模式中实现且系统时钟由一个高速振荡器提供。该模式下单片机正常工作的时钟源来自 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 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 及 f_{LIRC} 停止运行，看门狗定时器功能除能。

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。然而若看门狗定时器功能使能，f_{LIRC} 继续运行。

空闲模式 0

执行 HALT 指令后且 SMOD 寄存器中 IDLEN 位为高，SMOD1 寄存器中 FSYSON 位为低时，系统进入空闲模式 0。在空闲模式 0 中，CPU 停止，但一些外围功能如看门狗定时器、TM 将继续工作。在空闲模式 0 中，系统振荡器停止。

空闲模式 1

执行 HALT 指令后且 SMOD 寄存器中 IDLEN 位为高，SMOD1 寄存器中 FSYSON 位为高时，系统进入空闲模式 1。在空闲模式 1 中，CPU 停止，但会提供一个时钟源给一些外围功能如看门狗定时器、TM。在空闲模式 1 中，系统振荡器继续运行，该系统振荡器可以为高速或低速系统振荡器。在该模式中看门狗定时器时钟 f_{LIRC} 开启。

控制寄存器

寄存器 SMOD 和 SMOD1 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	—	LTO	HTO	IDLEN	HLCLK
R/W	R/W	R/W	R/W	—	R	R	R/W	R/W
POR	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_{LIRC})
001：f_L (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 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 未使用，读为“0”

Bit 3 **LTO**：低速振荡器就绪标志位
0：未就绪
1：就绪
此位为低速系统振荡器就绪标志位，用于表明低速系统振荡器在系统上电复位或经唤醒后何时稳定下来。当系统处于 SLEEP0 模式时，该标志为低。若系统时钟来自 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”。

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。

● **SMOD1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	—	—	—	D3	LVRF	—	WRF
R/W	R/W	—	—	—	R/W	R/W	—	R/W
POR	0	—	—	—	0	x	—	0

“x”：未知

Bit 7 **FSYSON**：IDLE 模式下 f_{SYS} 控制位
0：除能
1：使能

Bit 6~4 未使用，读为“0”

Bit 3 **D3**：保留位

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

Bit 1 未使用，读为“0”

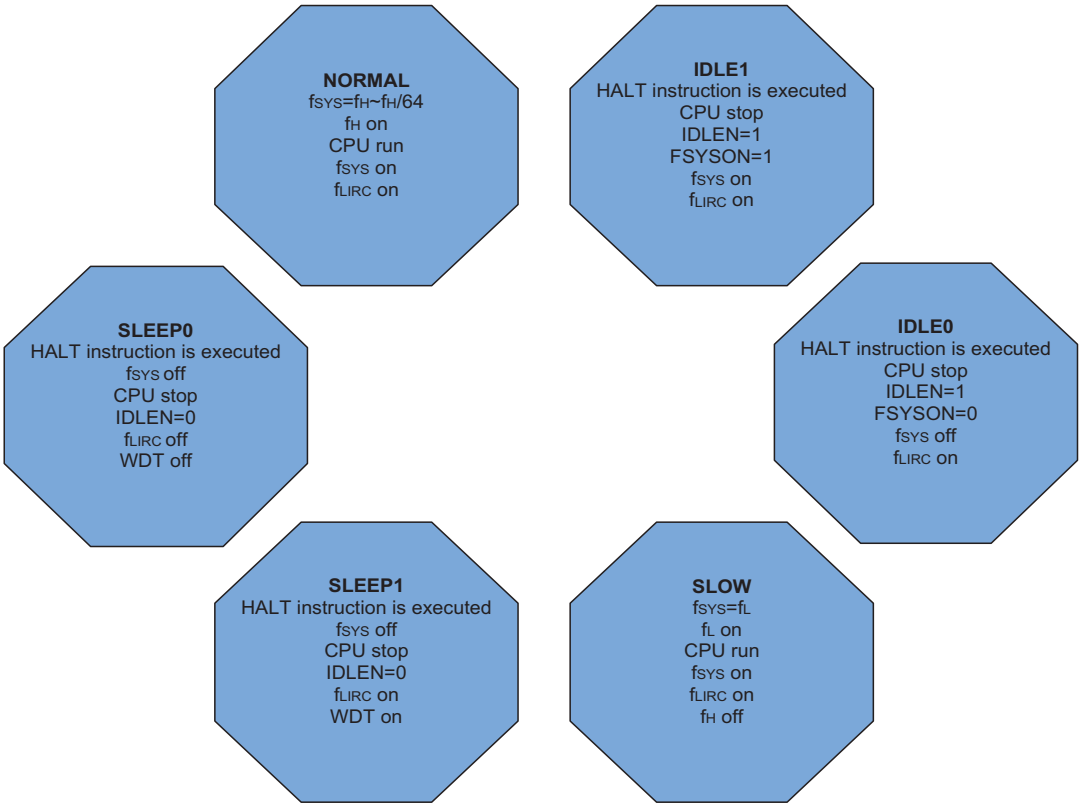
Bit 0 **WRF**：WDTC 控制的复位标志位
0：未发生
1：发生
WDT 控制寄存器复位时，该位被置为“1”，且通过应用程序清除。注意，该位只能由应用程序清零。

工作模式切换

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

简单来说，正常模式和低速模式间的切换仅需设置 SMOD 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 SMOD1 寄存器中的 FSYSON 位决定的。

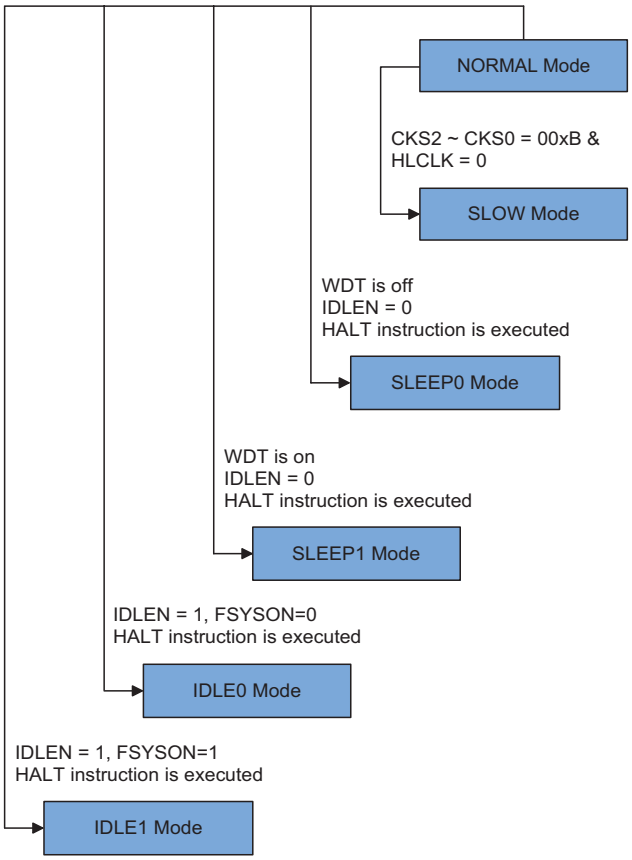
当 HLCLK 位变为低电平时，时钟源将由高速时钟源 f_H 转换成时钟源 $f_H/2$~$f_H/64$ 或 f_L。若时钟源来自 f_L，高速时钟源将停止运行以节省耗电。此时须注意，$f_H/16$ 和 $f_H/64$ 内部时钟源也将停止运行，由此会影响到如 TM 等内部功能的工作。所附流程图显示了单片机在不同工作模式间切换时的变化。


NORMAL
fSYS=fH~fH/64
fH on
CPU run
fSYS on
fLIRC on
IDLE1
HALT instruction is executed
CPU stop
IDLEN=1
FSYSON=1
fSYS on
fLIRC on
SLEEP0
HALT instruction is executed
fSYS off
CPU stop
IDLEN=0
fLIRC off
WDT off
IDLE0
HALT instruction is executed
CPU stop
IDLEN=1
FSYSON=0
fSYS off
fLIRC on
SLEEP1
HALT instruction is executed
fSYS off
CPU stop
IDLEN=0
fLIRC on
WDT on
SLOW
fSYS=fL
fL on
CPU run
fSYS on
fLIRC on
fH off

正常模式切换到低速模式

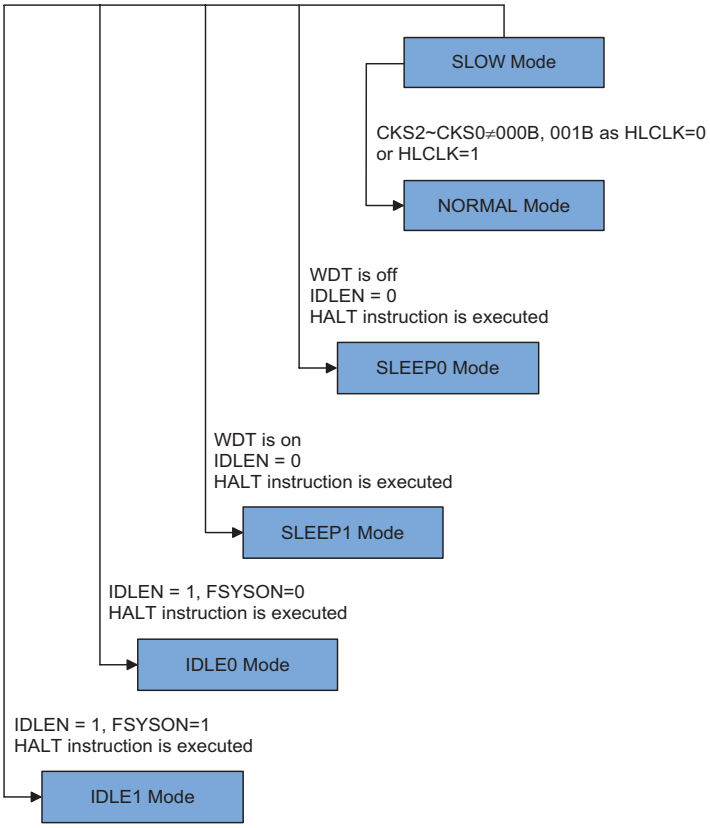
系统运行在正常模式时使用高速系统振荡器，因此较为耗电。可通过设置 SMOD 寄存器中的 HLCLK 位为“0”及 CKS2~CKS0 位为“000”或“001”使系统时钟切换至运行在低速模式下。此时将使用低速系统振荡器以节省耗电。用户可在对性能要求不高的操作中使用此方法以减少耗电。

低速模式的时钟源来自 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 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。



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 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 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。
- WDT 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 功能使能。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟停止运行，应用程序停止在“HALT”指令处。WDT 继续运行，其时钟源来自 f_{LIRC}。

- 数据存储器中的内容和寄存器将保持当前值。
- 若 WDT 使能，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 SMOD1 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，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 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 SMOD1 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：

- 系统时钟和时基时钟开启，应用程序停止在“HALT”指令处。
- 数据存储器中的内容和寄存器将保持当前值。
- 若 WDT 使能，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。

待机电流的注意事项

由于单片机进入休眠或空闲模式的主要原因是将 MCU 的电流降低到尽可能低，可能到只有几个微安的级别 (空闲模式 1 除外)，所以如果要将电路的电流进一步降低，电路设计者还应有其它的考虑。应该特别注意的是单片机的输入 / 输出引脚。所有高阻抗输入脚都必须连接到固定的高或低电平，因为引脚浮空会造成内部振荡并导致耗电增加。这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。在空闲模式 1 中，系统时钟开启。若系统时钟来自高速系统振荡器，额外的待机电流也可能会有几百微安。

唤醒

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。

看门狗定时器

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

看门狗定时器时钟源

WDT 定时器时钟源来自于内部时钟 f_{LIRC}，由 LIRC 振荡器提供。电压为 5V 时内部振荡器 LIRC 的周期大约为 32kHz。需要注意的是，这个特殊的内部时钟周期随 V_{DD}、温度和制成的不同而变化。看门狗定时器的时钟源可分频为 2^8~2^{15} 以提供更大的溢出周期，分频比由 WDTC 寄存器中的 WS2~WS0 位来决定。

看门狗定时器控制寄存器

WDTC 寄存器用于控制 WDT 功能的使能 / 除能及选择溢出周期。SMOD1 寄存器中的 WRF 为 WDT 软件复位标志位，这些寄存器与看门狗定时器的所有操作相关。

- **WDTC 寄存器**

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

Bit 7~3 **WE4~WE0**：WDT 软件控制位
01010：使能
10101：除能
其它值：MCU 复位
如果由于不利的环境因素使这些位发生改变，单片机将复位。复位动作发生在 2~3 个 f_{LIRC} 时钟周期后，且 SMOD1 寄存器的 WRF 位将置为“1”。

Bit 2~0 **WS2~WS0**：WDT 溢出周期选择位
000：$2^8/f_{LIRC}$
001：$2^9/f_{LIRC}$
010：$2^{10}/f_{LIRC}$
011：$2^{11}/f_{LIRC}$(默认)
100：$2^{12}/f_{LIRC}$
101：$2^{13}/f_{LIRC}$
110：$2^{14}/f_{LIRC}$
111：$2^{15}/f_{LIRC}$
这三位控制 WDT 时钟源的分频比，从而实现对 WDT 溢出周期的控制。

● **SMOD1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	—	—	—	D3	LVRF	—	WRF
R/W	R/W	—	—	—	R/W	R/W	—	R/W
POR	0	—	—	—	0	x	—	0

“x”：未知

Bit 7 **FSYSON**：IDLE 模式下 f_{SYS} 控制位
详见其它章节

Bit 6~4 未使用，读为“0”

Bit 3 **D3**：保留位

Bit 2 **LVRF**：LVR 复位标志位
详见其它章节

Bit 1 未使用，读为“0”

Bit 0 **WRF**：WDTC 控制的复位标志位
0：未发生
1：发生
WDT 控制寄存器复位时，该位被置为“1”，且通过应用程序清除。注意，该位只能由应用程序清零。

看门狗定时器操作

当 WDT 溢出时，它产生一个芯片复位的动作。这也就意味着正常工作期间，用户需在应用程序中看门狗溢出前有策略地清看门狗定时器以防止其产生复位，可使用清除看门狗指令实现。无论什么原因，程序失常跳转到一个未知的地址或进入一个死循环，这些清除指令都不能被正确执行，此种情况下，看门狗将溢出以使单片机复位。看门狗定时器控制寄存器 WDTC 中有五位 WE4~WE0 可提供使能 / 除能控制以及控制看门狗定时器复位操作。当 WE4~WE0 设置为“10101B”时除能 WDT 功能，而当设置为“01010B”时使能 WDT 功能。如果 WE4~WE0 设置为除“01010B”和“10101B”以外的值时，单片机将在 2~3 个 f_{LIRC} 时钟周期后复位。上电后这些位初始化为“01010B”。

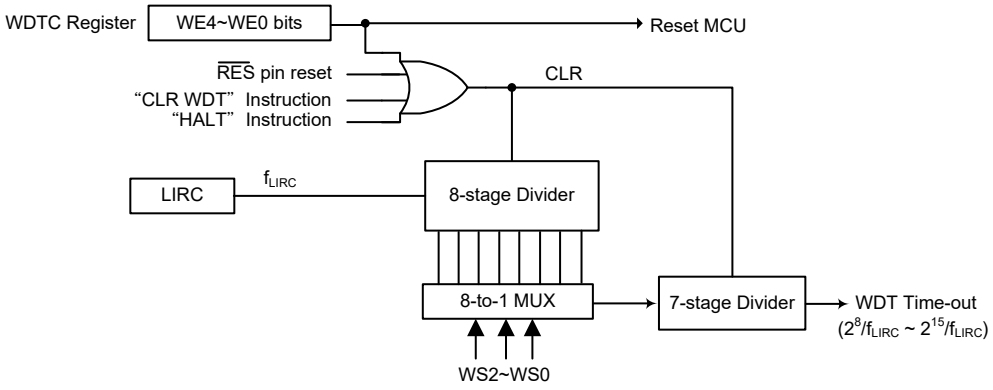
WE4~WE0 位	WDT 功能
10101B	除能
01010B	使能
其它值	复位单片机

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 将被置位，仅程序计数器和堆栈指针复位。有四种方法可以用来清除 WDT 的内容。第一种是 WDTC 软件复位，即写入除 01010B 和 10101B 外任何值到 WE4~WE0 位，第二种是外部硬件复位 ($\overline{RES}$ 引脚低电平)，第三种是通过软件清除指令，而第四种是通过“HALT”指令。

只有一种软件指令用于清除看门狗寄存器。因此只要执行“CLR WDT”便清除 WDT。

当设置分频比为 2^{15} 时，溢出周期最大。例如，时钟源为 32kHz LIRC 振荡器，分频比为 2^{15} 时最大溢出周期约 1s，分频比为 2^{8} 时最小溢出周期约 7.8ms。



看门狗定时器

复位和初始化

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

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。

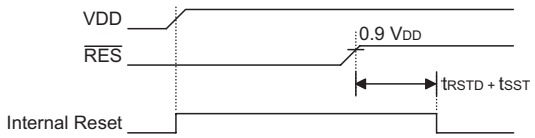
另一种复位为看门狗溢出单片机复位。不同方式的复位操作会对寄存器产生不同的影响。另一种复位为低电压复位即 LVR 复位，在电源供应电压低于 LVR 设定值时，系统会产生 LVR 复位，这种复位与 $\overline{\text{RES}}$ 脚拉低复位方式相似。

复位功能

包括内部和外部事件触发复位，单片机复位方式有以下几种：

上电复位

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



注：t_{RSTD} 为上电延迟时间，典型值为 50ms

上电复位时序图

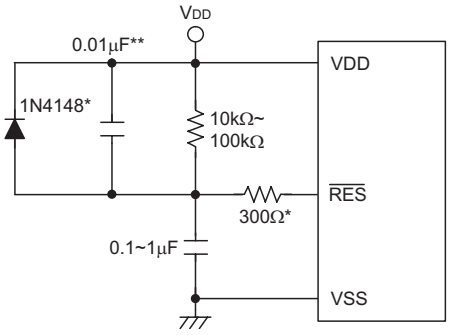
$\overline{\text{RES}}$ 引脚复位

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

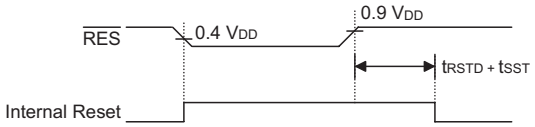
当系统在较强干扰的场合工作时，建议使用增强型的复位电路，如下图所示。



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 $\overline{RES}$ 电路

$\overline{RES}$ 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。



注：t_{RSTD} 为上电延迟时间，典型值为 16.7ms。

$\overline{RES}$ 复位时序图

- **RSTC 外部复位寄存器**

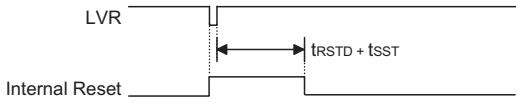
Bit	7	6	5	4	3	2	1	0
Name	RSTC7	RSTC6	RSTC5	RSTC4	RSTC3	RSTC2	RSTC1	RSTC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	0	1	0	1	0	1

Bit 7~3 **RSTC7~RSTC0**：PA7/$\overline{RES}$ 引脚控制位
01010101：PA7 或其它引脚功能
10101010：$\overline{RES}$ 引脚
其它：禁止使用
除 WDT 溢出复位外，其它所有复位方式如上电复位一样会将此寄存器复位。

低电压复位 – LVR

单片机具有低电压复位电路，用来监测它的电源电压。低电压复位功能总是使能于特定的电压值，V_{LVR}。例如在更换电池的情况下，单片机供应的电压可能会落在 0.9V~V_{LVR} 的范围内，这时 LVR 将会自动复位单片机，并且寄存器 SMOD1 中的 LVRF 位将被自动置位为 1。LVR 包含以下的规格：有效的 LVR

信号，即在 0.9V~V_{LVR} 的低电压状态的时间，必须超过 LVR 电气特性中 t_{LVR} 参数的值。如果低电压存在不超过 t_{LVR} 参数的值，则 LVR 将会忽略它且不会执行复位功能。实际的 V_{LVR} 参数固定为 2.1V。LVR 将在 2~3 个 LIRC 时钟周期后复位单片机。当单片机进入暂停模式时 LVR 功能将自动除能。



注：t_{RSTD} 为上电延迟时间，典型值为 50ms。

低电压复位时序图

- **SMOD1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	—	—	—	D3	LVRF	—	WRF
R/W	R/W	—	—	—	R/W	R/W	—	R/W
POR	0	—	—	—	0	x	—	0

“x”：未知

Bit 7 **FSYSON**：IDLE 模式下 f_{SYS} 控制位
详见其它章节

Bit 6~4 未使用，读为“0”

Bit 3 **D3**：保留位

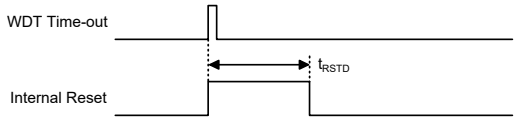
Bit 2 **LVRF**：LVR 复位标志位
0：未发生
1：发生
当特定的低电压复位条件发生时，该位被置为“1”。该位只能由应用程序清零。

Bit 1 未使用，读为“0”

Bit 0 **WRF**：WDTC 控制的复位标志位
详见其它章节

正常运行时看门狗溢出复位

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

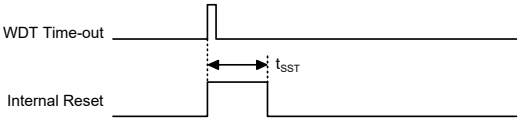


注：t_{RSTD} 为上电延迟时间，典型值为 16.7ms。

正常运行时看门狗溢出时序图

休眠或空闲时看门狗溢出复位

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



休眠或空闲时看门狗溢出复位时序图

复位初始状态

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

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

“u”代表不改变

在单片机上电复位之后，各功能单元初始化的情形，列于下表。

项目	复位后情况
程序计数器	清除为零
中断	所有中断被除能
看门狗定时器，时基	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，表格反映较大的封装的情况。

寄存器	上电复位	WDT 溢出 (正常模式)	$\overline{\text{RES}}$ 复位 (正常模式)	$\overline{\text{RES}}$ 复位 (HALT)	WDT 溢出 (HALT)*
PC	000H	000H	000H	000H	000H
MP0	1xxx xxxx	1xxx xxxx	1xxx xxxx	1xxx xxxx	1uuu uuuu
MP1	1xxx xxxx	1xxx xxxx	1xxx xxxx	1xxx xxxx	1uuu uuuu
BP	---- ---0	---- ---0	---- ---0	---- ---0	---- ---u
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	--xx xxxx	--uu uuuu	--uu uuuu	--uu uuuu	--uu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
SMOD	000- 0011	000- 0011	000- 0011	000- 0011	uuu- uuuu
INTEG	---- --00	---- --00	---- --00	---- --00	---- --uu
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI0	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
MFI1	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu

寄存器	上电复位	WDT 溢出 (正常模式)	$\overline{RES}$ 复位 (正常模式)	$\overline{RES}$ 复位 (HALT)	WDT 溢出 (HALT)*
PAPU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAWU	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS0	0000 0-00	0000 0-00	0000 0-00	0000 0-00	uuuu u-uu
WDTC	0101 0011	0101 0011	0101 0011	0101 0011	uuuu uuuu
TBC	0011 -111	0011 -111	0011 -111	0011 -111	uuuu –uuu
SMOD1	0--- 0x-0	0--- 0x-0	0--- 0x-0	0--- 0x-0	u--- uu-u
EEA	---0 0000	---0 0000	---0 0000	---0 0000	---u uuuu
EED	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADOL (ADRFS=0)	xxxx ----	xxxx ----	xxxx ----	xxxx ----	uuuu ----
SADOL (ADRFS=1)	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SADOH (ADRFS=0)	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
SADOH (ADRFS=1)	---- xxxx	---- xxxx	---- xxxx	---- xxxx	---- uuuu
SADC0	0000 --00	0000 --00	0000 --00	0000 --00	uuuu --uu
SADC1	000- -000	000- -000	000- -000	000- -000	uuu- -uuu
SADC2	00-- 0000	00-- 0000	00-- 0000	00-- 0000	uu-- uuuu
RSTC	0101 0101	0101 0101	0101 0101	0101 0101	uuuu uuuu
PASR	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBSR	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
STM0C0	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM0C1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM0DL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM0DH	---- --00	---- --00	---- --00	---- --00	---- --uu
STM0AL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
STM0AH	---- --00	---- --00	---- --00	---- --00	---- --uu
PTM1C0	0000 0---	0000 0---	0000 0---	0000 0---	uuuu u---
PTM1C1	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1DL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1DH	---- --00	---- --00	---- --00	---- --00	---- --uu
PTM1AL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1AH	---- --00	---- --00	---- --00	---- --00	---- --uu
PTM1RPL	0000 0000	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1RPH	---- --00	---- --00	---- --00	---- --00	---- --uu
PB	--11 1111	--11 1111	--11 1111	--11 1111	--uu uuuu
PBC	--11 1111	--11 1111	--11 1111	--11 1111	--uu uuuu
PBPU	--00 0000	--00 0000	--00 0000	--00 0000	--uu uuuu
EEC	---- 0000	---- 0000	---- 0000	---- 0000	---- uuuu

注：“*”表示热复位
“u”表示不改变
“x”表示未知
“-”表示未定义

输入 / 输出端口

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

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

寄存器名称	位							
	7	6	5	4	3	2	1	0
PA	D7	D6	D5	D4	D3	D2	D1	D0
PAC	D7	D6	D5	D4	D3	D2	D1	D0
PAPU	D7	D6	D5	D4	D3	D2	D1	D0
PAWU	D7	D6	D5	D4	D3	D2	D1	D0
PB	—	—	D5	D4	D3	D2	D1	D0
PBC	—	—	D5	D4	D3	D2	D1	D0
PBPU	—	—	D5	D4	D3	D2	D1	D0
PASR	PAS7	PAS6	PAS5	PAS4	PAS3	PAS2	PAS1	PAS0
PBSR	—	—	PBS5	PBS4	PBS3	PBS2	PBS1	PBS0
IFS0	PTCK1PS1	PTCK1PS0	STCK0PS	STP0IPS	PTP1IPS	—	INTPS1	INTPS0

输入 / 输出寄存器列表

上拉电阻

许多产品应用在端口处于输入状态时需要外加一个上拉电阻来实现上拉的功能。为了免去外部上拉电阻，当引脚规划为输入时，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~PBPU 来设置，它用一个 PMOS 晶体管来实现上拉电阻功能。

- **PAPU 寄存器**

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

Bit 7~0　PA 口 bit 7~bit 0 上拉电阻控制位
0：除能
1：使能

● **PBPU 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	—	—	0	0	0	0	0	0

Bit 7~6　未使用，读为“0”

Bit 5~0　PB 口 bit 5~bit 0 上拉电阻控制位
0：除能
1：使能

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。

● **PAWU 寄存器**

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

Bit 7~0　PA 口 bit 7~bit 0 唤醒功能控制位
0：除能
1：使能

输入 / 输出端口控制寄存器

每一个输入 / 输出口都具有各自的控制寄存器，即 PAC~PBC，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。

● **PAC 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	1	1	1	1	1	1	1	1

Bit 7~0　PA 口 bit 7~bit 0 输入 / 输出控制位
0：输出
1：输入

● **PBC 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	—	—	1	1	1	1	1	1

Bit 7~6　未使用，读为“0”

Bit 5~0　PB 口 bit 5~bit 0 输入 / 输出控制位
0：输出
1：输入

引脚共用功能

引脚的多功能可以增加单片机应用的灵活性。有限的引脚个数将会限制设计者，而引脚的多功能将会解决很多此类问题。每个功能可单独选择所在的引脚，以及一个确定的优先级，使得引脚上多种功能可以同时使用。此外，一些引脚功能可以通过寄存器 PASR 和 PBSR 进行设定。总之来说，模拟功能要比数字功能拥有更高的优先级。但是，如有含有两个以上的模拟功能都使能，且模拟信号来自同一个外部引脚，则由此引脚输入的模拟信号将通过内部连接到所有有效的模拟功能模块。

引脚共用功能选择寄存器

封装中有限的引脚个数会对某些单片机功能造成影响。然而，引脚功能共用可较大地扩大单片机的功能，并通过引脚共用功能选择寄存器灵活选择所需的引脚功能。

● **PASR 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PAS7	PAS6	PAS5	PAS4	PAS3	PAS2	PAS1	PAS0
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　**PAS7**：PA7 功能选择
0：PA7/PTCK1
1：STP0B
只有当 RSTC=55H 时，PAS7 选择位才是有效的。

Bit 6　**PAS6**：PA6 功能选择
0：PA6/PTCK1/STP0I
1：STP0

Bit 5　**PAS5**：PA4 功能选择
0：PA4/INT/PTCK1
1：STP0

Bit 4　**PAS4**：PA3 功能选择
0：PA3/INT/STCK0
1：AN3

Bit 3　**PAS3**：PA2 功能选择
0：PA2/INT/STCK0
1：AN2

Bit 2~1　**PAS2~PAS1**：PA1 功能选择
00：PA1
01：PA1
10：VREF
11：AN1

Bit 0 **PAS0**：PA0 功能选择
0：PA0/STP0I
1：AN0

● **PBSR 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PBS5	PBS4	PBS3	PBS2	PBS1	PBS0
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 **PBS5**：PB5 功能选择
0：PB5
1：PTP1

Bit 4 **PBS4**：PB4 功能选择
0：PB4
1：PTP1B

Bit 3 **PBS3**：PB3 功能选择
0：PB3
1：PTP1

Bit 2 **PBS2**：PB2 功能选择
0：PB2
1：PTP1B

Bit 1 **PBS1**：PB1 功能选择
0：PB1/PTCK1
1：STP0B

Bit 0 **PBS0**：PB0 功能选择
0：PB0/PTP1I
1：VREFO

● **IFS0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PTCK1PS1	PTCK1PS0	STCK0PS	STP0IPS	PTP1IPS	—	INTPS1	INTPS0
R/W	R/W	R/W	R/W	R/W	R/W	—	R/W	R/W
POR	0	0	0	0	0	—	0	0

Bit 7~6 **PTCK1PS1, PTCK1PS0**：PTCK1 引脚重置控制
00：PTCK1 on PA4 (默认)
01：PTCK1 on PA6
10：PTCK1 on PA7
11：PTCK1 on PB1

Bit 5 **STCK0PS**：STCK0 引脚重置控制
0：STCK0 on PA3 (默认)
1：STCK0 on PA2

Bit 4 **STP0IPS**：STP0I 引脚重置控制
0：STP0I on PA6 (默认)
1：STP0I on PA0

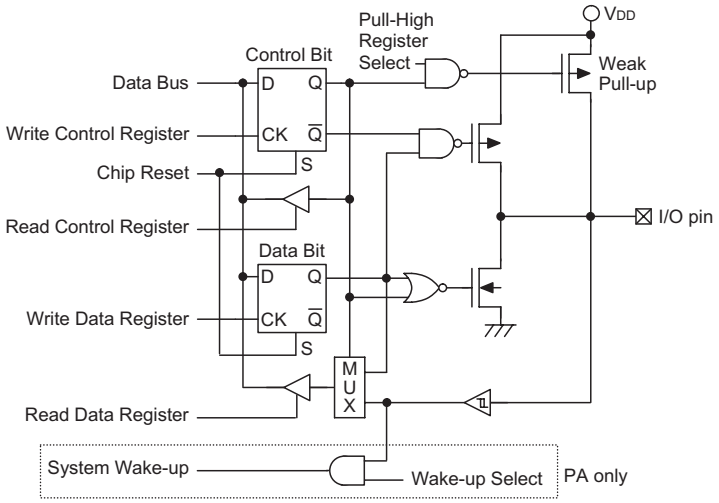
Bit 3 **PTP1IPS**：PTP1I 引脚重置控制
0：PTP1I on PA5(默认)
1：PTP1I on PB0

Bit 2 未使用，读为“0”

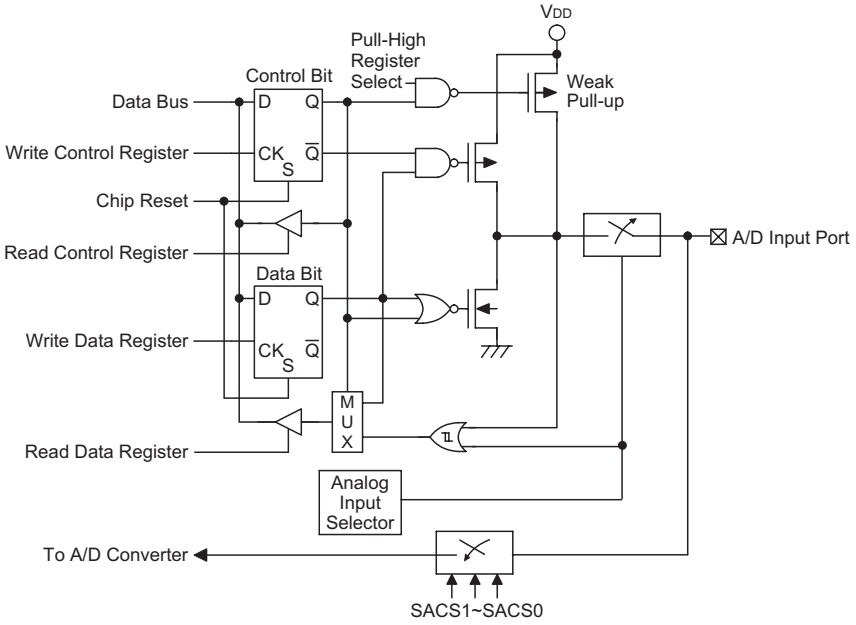
Bit 1~0 **INTPS1, INTPS0**：INT 引脚重置控制
00：INT on PA3 (默认)
01：INT on PA2
10：INT on PA4
11：INT on PA5

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。图中的引脚共用结构并非针对所有单片机。



通用输入 / 输出端口



A/D 转换器输入输出端口

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~PBC，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~PB 在程序中被预先设定。设置哪些引脚是输入及哪些引脚是输出，可通过设置正确的值到适当的端口控制寄存器，或使用指令“SET [m].i”及“CLR [m].i”来设定端口控制寄存器中个别的位。注意，当使用这些位控制指令时，系统即将产生一个读 - 修改 - 写的操作。单片机需要先读入整个端口上的数据，修改个别的位，然后重新把这些数据写入到输出端口。

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

定时器模块 – TM

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

在这里只介绍各种 TM 的共性，更多详细资料请分别参考标准型和周期型定时器章节。

简介

该单片机包含两个 TM。每个 TM 可被划分为一个特定的类型，即标准型 TM – STM 或周期型 TM – PTM。虽然性质相似，但不同 TM 特性复杂度不同。本章介绍周期型和标准型 TM 的共性，更多详细资料分别见后面各章。此两种类型 TM 的特性和区别见下表。

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

TM 功能概要

TM 操作

两种不同类型的 TM 提供从简单的定时操作到 PWM 信号产生等多种功能。理解 TM 操作的关键是比较 TM 内独立运行的计数器的值与内部比较器的预置值。当计数器的值与比较器的预置值相同时，则比较匹配，TM 中断信号产生，清零计数器并改变 TM 输出引脚的状态。用户可选择内部时钟或外部时钟来驱动内部 TM 计数器。

TM 时钟源

驱动 TM 计数器的时钟源很多。通过设置 xTMnC0 控制寄存器的 xTnCK2~xTnCK0 位，选择所需的时钟源。该时钟源来自系统时钟 f_{SYS} 或内部高速时钟 f_H 或 f_{TBC} 时钟源或外部 xTCKn 引脚。xTCKn 引脚时钟源用于允许外部信号作为 TM 时钟源或用于事件计数。

TM 中断

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

TM 外部引脚

无论哪种类型的 TM，都有两个 TM 输入引脚 xTCKn 和 xTPnI。对于输入引脚 xTCKn，可通过设置 xTMnC0 寄存器中的 xTnCK2~xTnCK0 位，选择 TM 功能并将该引脚作为 TM 时钟源输入脚。外部时钟源可通过该引脚来驱动内部 TM。外部 TM 输入脚也与其它功能共用，但是，如果设置适当值给 xTnCK2~xTnCK0，该引脚会连接到内部 TM。TM 引脚可选择上升沿有效或下降沿有效。

对于另外一个输入引脚 xTPnI，可作为捕捉输入引脚。通过 xTMnC1 寄存器中的 xTnIO1~xTnIO0 可设置为上升沿，下降沿或双边沿有效。

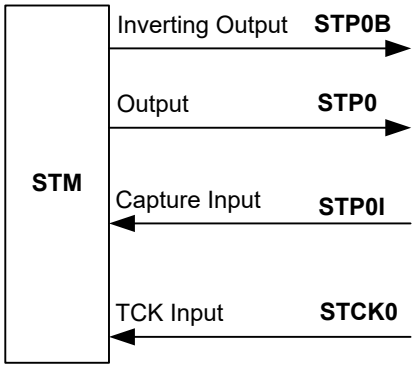
每个 TM 有两个输出引脚 xTPn 和 xTPnB。当 TM 工作在比较匹配输出模式且比较匹配发生时，这些引脚会由 TM 控制切换到高电平或低电平或翻转。外部 xTPn 输出引脚也被 TM 用来产生 PWM 输出波形。当 TM 输出引脚与其它功能共用时，TM 输出功能需要通过寄存器先被设置。寄存器中的相应位用于决定其相关引脚用于外部 TM 输出还是用于其它功能。

STM	PTM
STCK0, STP0I; STP0, STP0B	PTCK1, PTP1I, PTP1, PTP1B

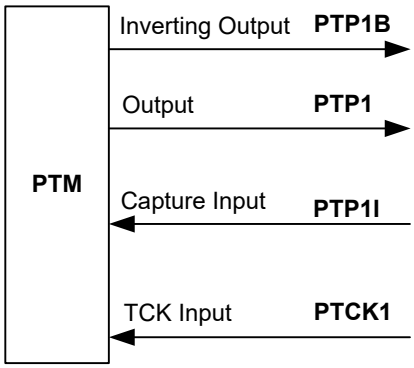
TM 输入 / 输出引脚

TM 输入 / 输出引脚控制寄存器

通过设置相应的引脚共用控制寄存器，选择作为 TM 输入 / 输出功能或其它共用功能。合理的设置寄存器相应位，相关引脚可用作 TM 输入 / 输出。具体的描述请参考引脚共用功能选择寄存器描述。



STM 功能引脚框图

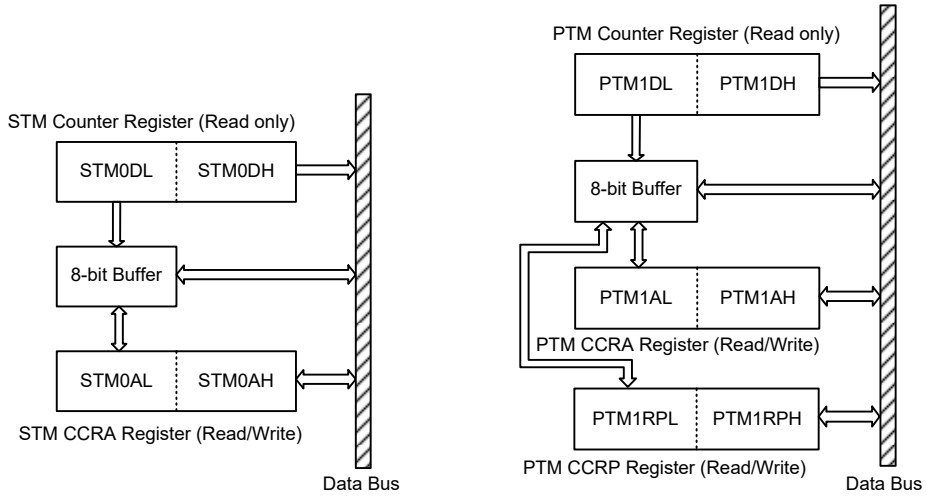


PTM 功能引脚框图

编程注意事项

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

CCRA 和 CCRP 寄存器访问方式如下图所示，读写这些成对的寄存器需通过特殊的方式。建议使用“MOV”指令按照以下步骤访问 CCRA 或 CCRP 低字节寄存器，否则可能导致无法预期的结果。



读写流程如下步骤所示：

- 写数据至 CCRA 或 PTM CCRP
  - ♦ 步骤 1. 写数据至低字节寄存器 xTMnAL 或 PTM1RPL
    – 注意，此时数据仅写入 8-bit 缓存器。
  - ♦ 步骤 2. 写数据至高字节寄存器 xTMnAH 或 PTM1RPH
    – 注意，此时数据直接写入高字节寄存器，同时锁存在 8-bit 缓存器中的数据写入低字节寄存器。
- 由计数器寄存器和 CCRA 或 PTM CCRP 中读取数据
  - ♦ 步骤 1. 由高字节寄存器 xTMnDH、xTMnAH 或 PTM1RPH 读取数据
    – 注意，此时高字节寄存器中的数据直接读取，同时由低字节寄存器读取的数据锁存至 8-bit 缓存器中。

♦ 步骤 2. 由低字节寄存器 xTMnDL、xTMnAL 或 PTM1RPL 读取数据
– 注意，此时读取 8-bit 缓存器中的数据。

标准型 TM – STM

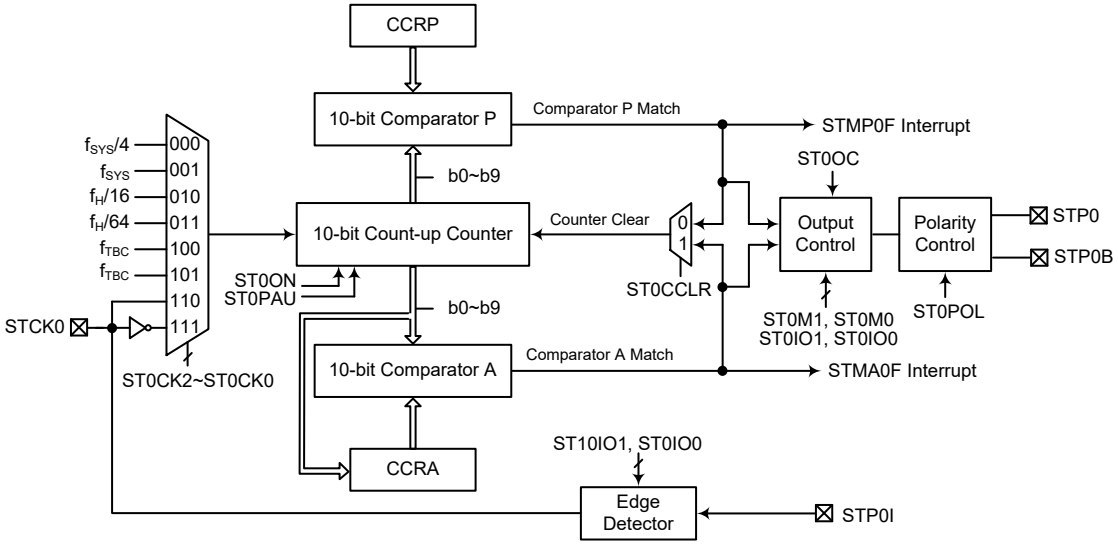
标准型 TM 包括 5 种工作模式，即比较匹配输出，定时 / 事件计数器，捕捉输入，单脉冲输出和 PWM 输出模式。标准型 TM 由两个外部输入脚控制并驱动两个外部输出脚。

TM 类型	TM 输入引脚	TM 输出引脚
10-bit STM	STCK0, STP0I	STP0, STP0B

标准型 TM 操作

标准型 TM 核心是一个由用户选择的内部或外部时钟源驱动的 10 位向上计数器，它还包括两个内部比较器即比较器 A 和比较器 P。这两个比较器将计数器的值与 CCRP 和 CCRA 寄存器中的值进行比较。CCRP 是 3 位宽度，与计数器的高 3 位比较；而 CCRA 是 10 位的，与计数器的所有位比较。

通过应用程序改变 10 位计数器值的唯一方法是使 ST0ON 位发生上升沿跳变清除计数器。此外，计数器溢出或比较匹配也会自动清除计数器。上述条件发生时，通常情况会产生 TM 中断信号。标准型 TM 可工作在不同的模式，可由包括来自输入脚的不同时钟源驱动，也可以控制输出脚。所有工作模式的设定都是通过设置相关寄存器来实现的。



标准型 TM 框图

标准型 TM 寄存器介绍

标准型 TM 的所有工作模式由一系列寄存器控制。一对只读寄存器用来存放 10 位计数器的值，一对读 / 写寄存器存放 10 位 CCRA 的值。剩下两个控制寄存器设置工作模式，以及 CCRP 的 3 个位。

寄存器名称	位							
	7	6	5	4	3	2	1	0
STM0C0	ST0PAU	ST0CK2	ST0CK1	ST0CK0	ST0ON	ST0RP2	ST0RP1	ST0RP0
STM0C1	ST0M1	ST0M0	ST0IO1	ST0IO0	ST0OC	ST0POL	ST0DPX	ST0CCLR
STM0DL	D7	D6	D5	D4	D3	D2	D1	D0
STM0DH	—	—	—	—	—	—	D9	D8
STM0AL	D7	D6	D5	D4	D3	D2	D1	D0
STM0AH	—	—	—	—	—	—	D9	D8

10-bit 标准型 TM 寄存器列表

- **STM0C0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	ST0PAU	ST0CK2	ST0CK1	ST0CK0	ST0ON	ST0RP2	ST0RP1	ST0RP0
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 **ST0PAU**：STM 计数器暂停控制位
0：运行
1：暂停
通过设置此位为高可使计数器暂停，清零此位恢复正常计数器操作。当处于暂停条件时，TM 保持上电状态并继续耗电。当此位由低到高转换时，计数器将保留其剩余值，直到此位再次改变为低电平，并从此值开始继续计数。

Bit 6~4 **ST0CK2~ST0CK0**：选择 STM 计数时钟位
000：$f_{SYS}/4$
001：f_{SYS}
010：$f_H/16$
011：$f_H/64$
100：f_{TBC}
101：f_{TBC}
110：STCK0 上升沿时钟
111：STCK0 下降沿时钟
此三位用于选择 TM 的时钟源。外部引脚时钟源能被选择在上升沿或下降沿有效。f_{SYS} 是系统时钟，f_H 和 f_{TBC} 是其它的内部时钟源，细节方面请参考振荡器章节。

Bit 3 **ST0ON**：STM 计数器 On/Off 控制位
0：Off
1：On
此位控制 TM 的总开关功能。设置此位为高则使能计数器使其运行，清零此位则除能 TM。清零此位将停止计数器并关闭 TM 减少耗电。当此位经由高到低转换时，内部计数器将保持其剩余值，直到此位再次改变为高电平。若 STM 处于比较匹配输出模式、PWM 输出模式或单脉冲输出模式，当 ST0ON 位经由低到高的转换时，STM 输出脚 STP 将复位至 ST0OC 位指定的初始值。

Bit 2~0 **ST0RP2~ST0RP0**：STM CCRP 3-bit 寄存器，与 STM 计数器 bit 9~bit 7 进行比较比较器 P 匹配周期

000：1024 个 STM 时钟周期
001：128 个 STM 时钟周期
010：256 个 STM 时钟周期
011：384 个 STM 时钟周期
100：512 个 STM 时钟周期
101：640 个 STM 时钟周期
110：768 个 STM 时钟周期
111：896 个 STM 时钟周期

此三位设置内部 CCRP 3-bit 寄存器的值，然后与内部计数器的高三位进行比较。如果 ST0CCLR 位设置为“0”时，此比较结果可清除内部计数器。ST0CCLR 位设为低，内部计数器在比较器 P 比较匹配发生时被重置；由于 CCRP 只与计数器高三位比较，比较结果是 128 时钟周期的倍数。CCRP 被清零时，实际上会使得计数器在最大值溢出。

● STM0C1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	ST0M1	ST0M0	ST0IO1	ST0IO0	ST0OC	ST0POL	ST0DPX	ST0CCLR
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 **ST0M1~ST0M0**：选择 STM 工作模式位

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

这两位设置 TM 需要的工作模式。为了确保操作可靠，TM 应在 ST0M1 和 ST0M0 位有任何改变前先关掉。在定时 / 计数器模式，TM 输出脚控制必须除能。

Bit 5~4 **ST0IO1~ST0IO0**：选择 STM 输出功能位

比较匹配输出模式
00：无变化
01：输出低
10：输出高
11：输出翻转

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

捕捉输入模式
00：在 STP0I 上升沿输入捕捉
01：在 STP0I 下降沿输入捕捉
10：在 STP0I 双沿输入捕捉
11：输入捕捉除能

定时 / 计数器模式
未使用

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

在比较匹配输出模式下，ST0IO1 和 ST0IO0 位决定当从比较器 A 比较匹配输出发生时 TM 输出脚如何改变状态。当从比较器 A 比较匹配输出发生时 TM 输出脚能设为切换高、切换低或翻转当前状态。若此两位同时为 0 时，这个输出将不会改变。TM 输出脚的初始值通过 STM0C1 寄存器的 ST0OC 位设置取得。注意，由 ST0IO1 和 ST0IO0 位得到的输出电平必须与通过 ST0OC 位设置的初始值不同，否则当比较匹配发生时，TM 输出脚将不会发生变化。在 TM 输出脚改变状态后，通过 ST0ON 位由低到高电平的转换复位至初始值。

在 PWM 输出模式，ST0IO1 和 ST0IO0 决定比较匹配条件发生时怎样改变 STM 输出脚的状态。PWM 输出通过这两位的变化进行更新。尽量在 STM 关闭时改变 STIO1 和 STIO0 位的值。若在 STM 运行时改变 STIO1 和 STIO0 的值，PWM 输出的值将无法预料。

Bit 3 **ST0OC**：STM 输出控制位

比较匹配输出模式
- 0：初始低
- 1：初始高

PWM 模式 / 单脉冲输出模式
- 0：低有效
- 1：高有效

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

Bit 2 **ST0POL**：STM 输出极性控制位
- 0：同相
- 1：反相

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

Bit 1 **ST0DPX**：STM PWM 周期 / 占空比控制位
- 0：CCRP - 周期；CCRA - 占空比
- 1：CCRP - 占空比；CCRA - 周期

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

Bit 0 **ST0CCLR**：选择 STM 计数器清零条件位
- 0：STM 比较器 P 匹配
- 1：STM 比较器 A 匹配

此位用于选择清除计数器的方法。标准型 TM 包括两个比较器 - 比较器 A 和比较器 P。这两个比较器每个都可以用作清除内部计数器。ST0CCLR 位设为高，计数器在比较器 A 比较匹配发生时被清除；此位设为低，计数器在比较器 P 比较匹配发生或计数器溢出时被清除。计数器溢出清除的方法仅在 CCRP 被清除为 0 时才能生效。ST0CCLR 位在 PWM，单脉冲或输入捕捉模式时未使用。

● **STM0DL 寄存器**

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

Bit 7~0 STM 计数器低字节寄存器 bit 7~bit 0
STM0 10-bit 计数器 bit 7~bit 0

● **STM0DH 寄存器**

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

Bit 7~2 未使用，读为“0”
Bit 1~0 STM 计数器高字节寄存器 bit 1~bit 0
STM 10-bit 计数器 bit 9~bit 8

● **STM0AL 寄存器**

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

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

● **STM0AH 寄存器**

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

Bit 7~2 未使用，读为“0”
Bit 1~0 STM CCRA 高字节寄存器 bit 1~bit 0
STM 10-bit CCRA bit 9~bit 8

标准型 TM 工作模式

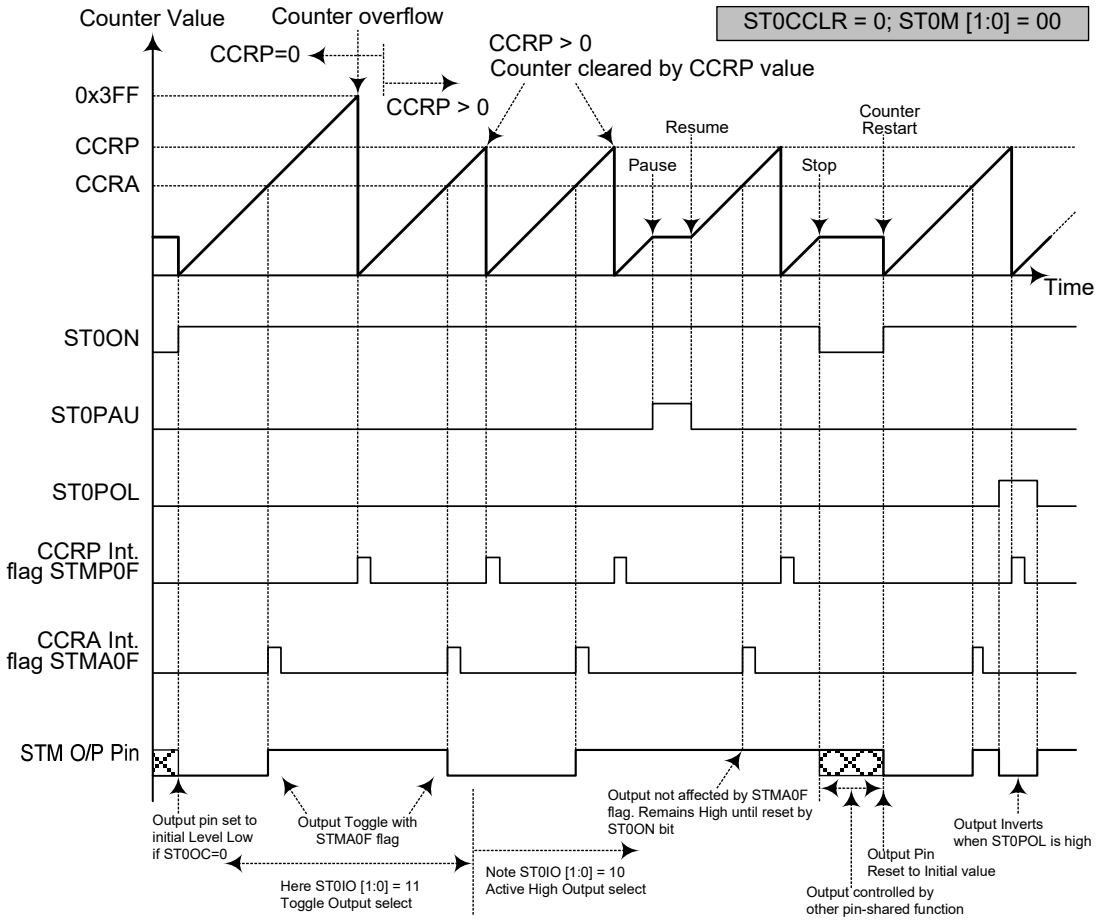
标准型 TM 有五种工作模式，即比较匹配输出模式，PWM 输出模式，单脉冲输出模式，捕捉输入模式或定时 / 计数器模式。通过设置 STM0C1 寄存器的 ST0M1 和 ST0M0 位选择任意模式。

比较匹配输出模式

为使 TM 工作在此模式，STM0C1 寄存器中的 ST0M1 和 ST0M0 位需要设置为“00”。当工作在该模式，一旦计数器使能并开始计数，有三种方法来清零，分别是：计数器溢出，比较器 A 比较匹配发生和比较器 P 比较匹配发生。当 STCCLR 位为低，有两种方法清除计数器。一种是比较器 P 比较匹配发生，另一种是 CCRP 所有位设置为零并使得计数器溢出。此时，比较器 A 和比较器 P 的请求标志位 STMA0F 和 STMP0F 将分别置位。

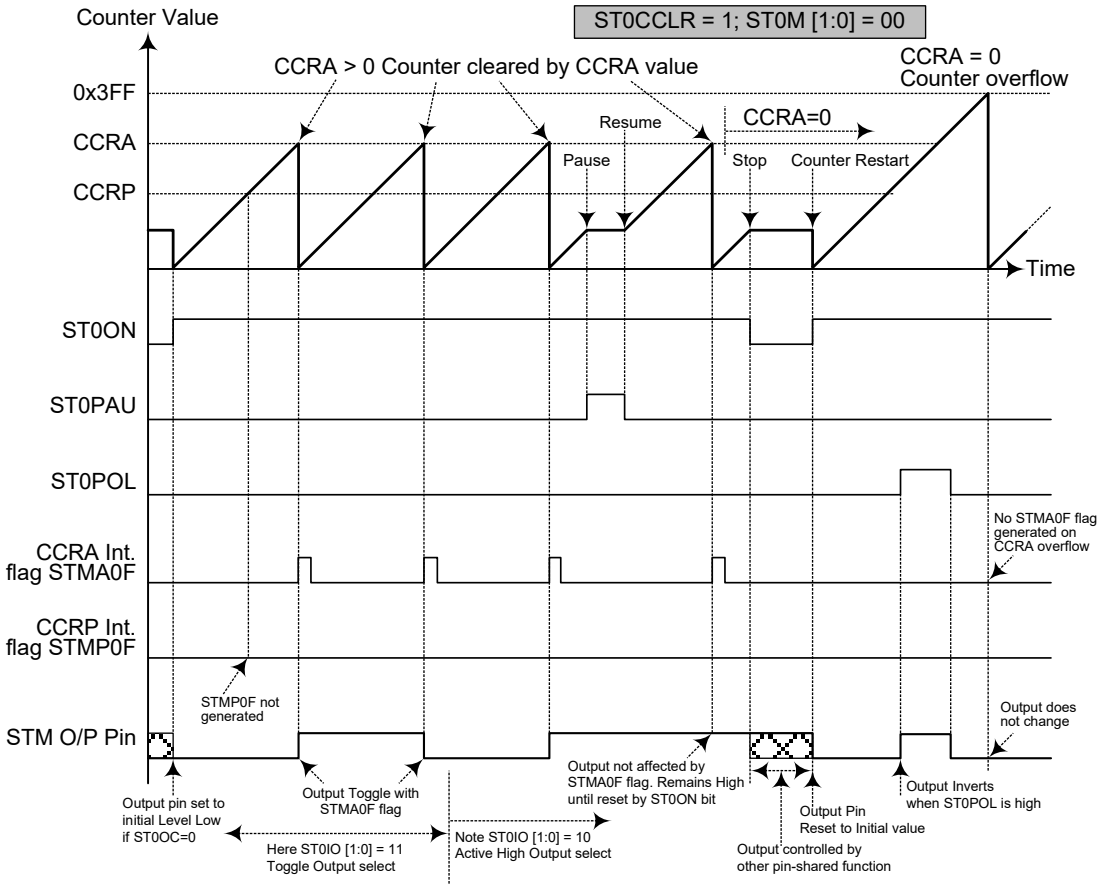
如果 STM0C1 寄存器的 ST0CCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅产生 STMA0F 中断请求标志。所以当 ST0CCLR 为高时，不会产生 STMP0F 中断请求标志。在比较匹配输出模式下，CCRA 不能设为“0”。如果 CCRA 为“0”，当 CCRA 达到最大值 0x3FF 时，计数器将溢出，不会产生 STMA0F 中断请求标志。

正如该模式名所言，当比较匹配发生后，TM 输出脚状态改变。当比较器 A 比较匹配发生后 STMA0F 标志产生时，TM 输出脚状态改变。比较器 P 比较匹配发生时产生的 STMP0F 标志不影响 TM 输出脚。TM 输出脚状态改变方式由 STM0C1 寄存器中 ST0IO1 和 ST0IO0 位决定。当比较器 A 比较匹配发生时，ST0IO1 和 ST0IO0 位决定 TM 输出脚输出高，低或翻转当前状态。TM 输出脚初始值，在 ST0ON 位由低到高电平的变化后通过 ST0OC 位设置。注意，若 ST0IO1 和 ST0IO0 位同时为 0 时，引脚输出不变。



比较匹配输出模式 – ST0CCLR=0

注：1. ST0CCLR=0，比较器 P 匹配将清除计数器
2. TM 输出脚仅由 STMA0F 标志位控制
3. 在 ST0ON 上升沿 TM 输出脚复位至初始值



比较匹配输出模式 – ST0CCLR=1

注：1. ST0CCLR=1，比较器 A 匹配将清除计数器
2. TM 输出脚仅由 STMA0F 标志位控制
3. 在 ST0ON 上升沿 TM 输出脚复位至初始值
4. 当 ST0CCLR=1 时，不会产生 STMP0F 标志

定时 / 计数器模式

为使 TM 工作在此模式，STM0C1 寄存器中的 ST0M1 和 ST0M0 位需要设置为“11”。定时 / 计数器模式与比较输出模式操作方式相同，并产生同样的中断请求标志。不同的是，在定时 / 计数器模式下 TM 输出脚未使用。因此，比较匹配输出模式中的描述和时序图可以适用于此功能。该模式中未使用的 TM 输出脚通过引脚共用功能选择寄存器设置用作普通 I/O 脚或其它功能。

PWM 输出模式

为使 TM 工作在此模式，STM0C1 寄存器中的 ST0M1 和 ST0M0 位需要设置为“10”，且 ST0IO1 和 ST0IO0 位也需要设置为“10”。TM 的 PWM 功能在马达控制，加热控制，照明控制等方面十分有用。给 TM 输出脚提供一个频率固定但占空比可调的信号，将产生一个有效值等于 DC 均方根的 AC 方波。

由于 PWM 波形的周期和占空比可调，其波形的选择就较为灵活。在 PWM 模式中，ST0CCLR 位不影响 PWM 周期。CCRA 和 CCRP 寄存器决定 PWM 波形，一个用来清除内部计数器并控制 PWM 波形的频率，另一个用来控制占空比。哪个寄存器控制频率或占空比取决于 STM0C1 寄存器的 ST0DPX 位。所以 PWM 波形由 CCRA 和 CCRP 寄存器共同决定。

当比较器 A 或比较器 P 比较匹配发生时，将产生 CCRA 或 CCRP 中断标志。STM0C1 寄存器中的 ST0OC 位决定 PWM 波形的极性，ST0IO1 和 ST0IO0 位使能 PWM 输出或将 TM 输出脚置为逻辑高或逻辑低。ST0POL 位对 PWM 输出波形的极性取反。

- **10-bit STM，PWM 模式，边沿对齐模式，ST0DPX=0**

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

若 f_{SYS}=16MHz，TM 时钟源选择 f_{SYS}/4，CCRP=100b，CCRA=128，

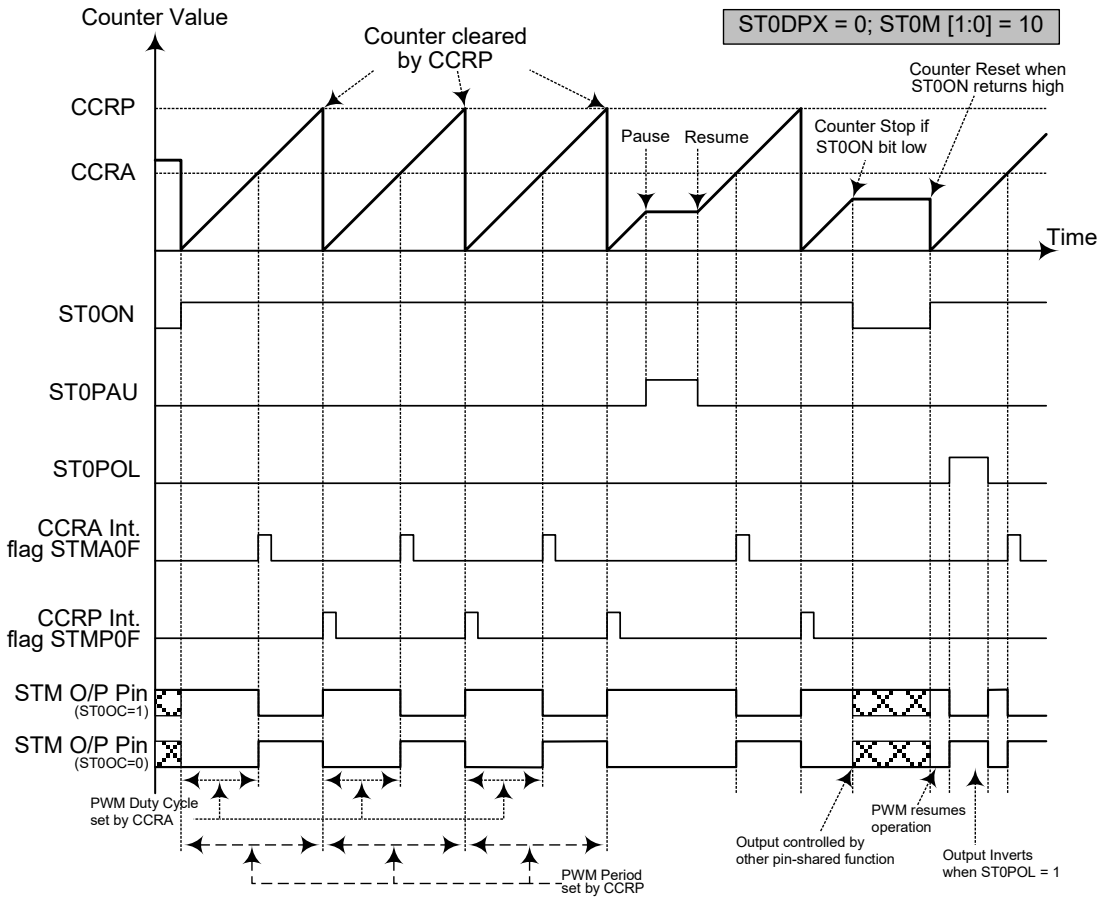
STM PWM 输出频率 = (f_{SYS}/4)/512=f_{SYS}/2048=7.8125kHz，duty=128/(2×256)=25%。

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

- **10-bit STM，PWM 模式，边沿对齐模式，ST0DPX=1**

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

PWM 的输出周期由 CCRA 寄存器的值与 TM 的时钟共同决定，PWM 的占空比由 CCRP×128 (除了 CCRP 为“0”外) 的值决定。



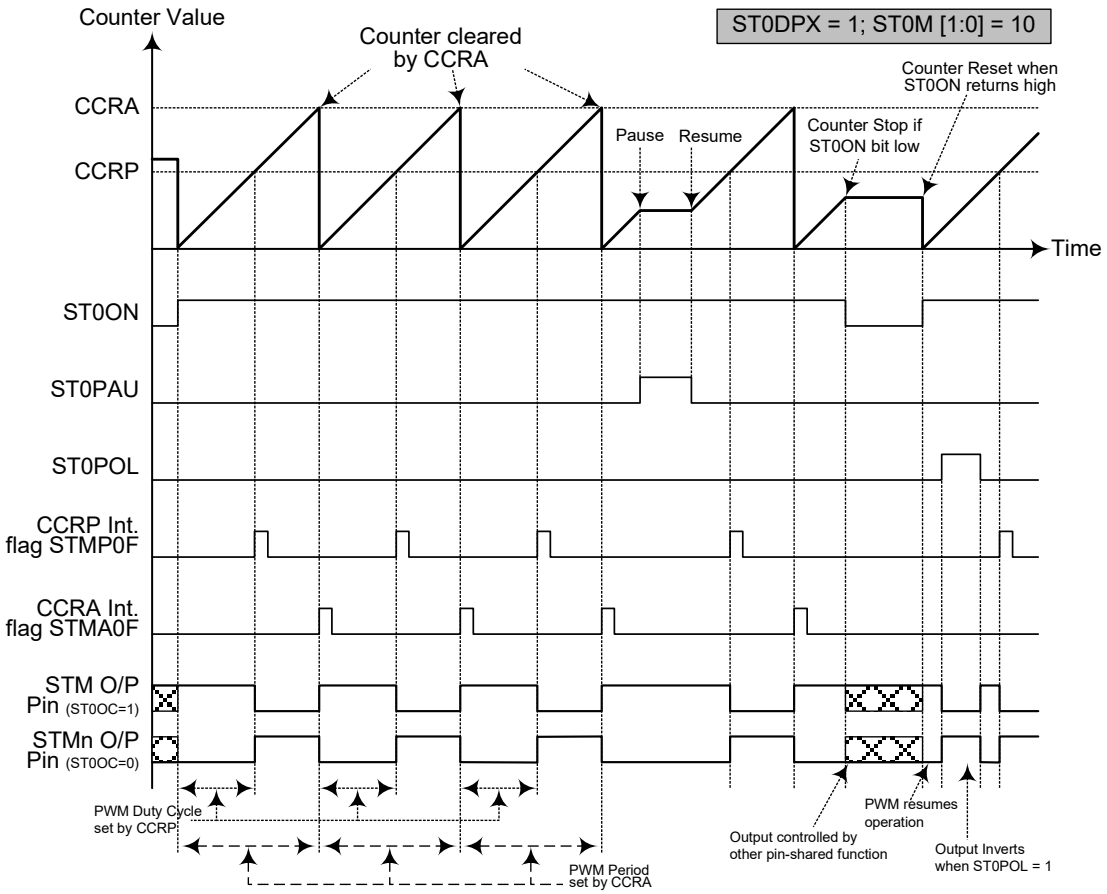
PWM 输出模式 – ST0DPX=0

注：1. ST0DPX=0，CCRP 清除计数器

2. 计数器清零并设置 PWM 周期

3. 当 ST0IO1，ST0IO0=00 或 01，PWM 功能不变

4. ST0CCLR 位不影响 PWM 操作



PWM 输出模式 – ST0DPX=1

注：1. ST0DPX=1，CCRA 清除计数器

2. 计数器清零并设置 PWM 周期

3. 当 ST0IO1，ST0IO0=00 或 01，PWM 功能不变

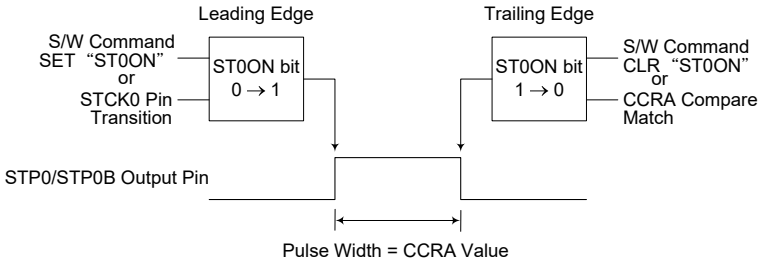
4. ST0CCLR 位不影响 PWM 操作

单脉冲模式

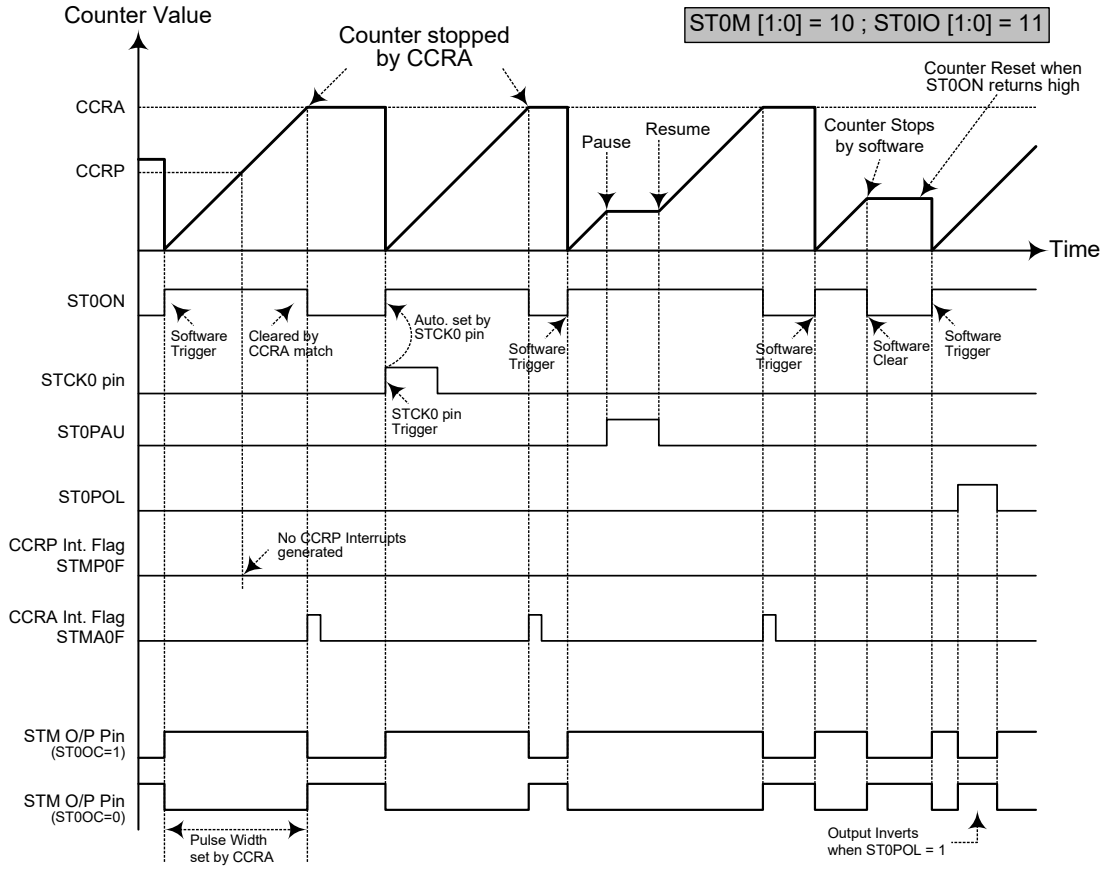
为使 TM 工作在此模式，STM0C1 寄存器中的 ST0M1 和 ST0M0 位需要设置为“10”，同时 ST0IO1 和 ST0IO0 位需要设置为“11”。正如模式名所言，单脉冲输出模式，在 TM 输出脚将产生一个脉冲输出。

脉冲输出可以通过应用程序控制 ST0ON 位由低到高的转变来触发。而处于单脉冲模式时，STON 位可在 STCK 脚发生有效边沿跳转时自动由低转变为高，进而初始化单脉冲输出状态。当 ST0ON 位转变为高电平时，计数器将开始运行，并产生脉冲前沿。当脉冲有效时 ST0ON 位保持高电平。通过应用程序使 ST0ON 位清零或比较器 A 比较匹配发生时，产生脉冲后沿。

然而，比较器 A 比较匹配发生时，会自动清除 ST0ON 位并产生单脉冲输出后沿。CCRA 的值通过这种方式控制脉冲宽度。比较器 A 比较匹配发生时，也会产生 TM 中断。ST0ON 位在计数器重启时会发生由低到高的转变，此时计数器才复位至零。在单脉冲模式中，CCRP 寄存器，ST0CCLR 和 ST0DPX 位未使用。



单脉冲产生示意图



单脉冲模式

注：1. 通过 CCRA 匹配停止计数器

2. CCRP 未使用

3. 通过设置 ST0ON 位为高来触发脉冲

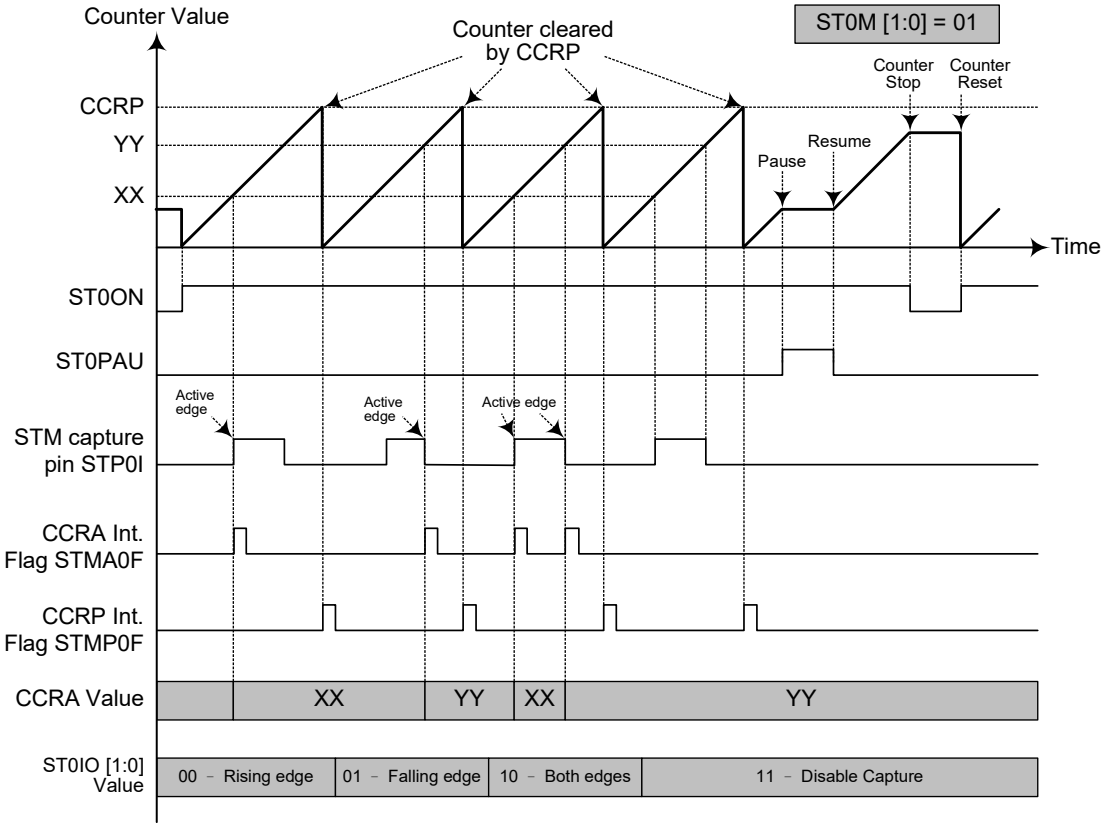
4. 单脉冲模式中，ST0IO[1:0] 需置位“11”，且不能更改。

捕捉输入模式

为使 TM 工作在此模式，STM0C1 寄存器中的 ST0M1 和 ST0M0 位需要设置为“01”。此模式使能外部信号捕捉并保存内部计数器当前值，因此被用于诸如脉冲宽度测量的应用中。STP0I 脚上的外部信号，通过设置 STM0C1 寄存器的 ST0IO1 和 ST0IO0 位选择有效边沿类型，即上升沿，下降沿或双沿有效。通过应用程序将 ST0ON 位由低置为高时，计数器启动。

当 STP0I 脚出现有效边沿转换时，计数器当前值被锁存到 CCRA 寄存器，并产生 TM 中断。不考虑 STP0I 引脚事件，计数器继续工作直到 ST0ON 位发生下降沿跳变。当 CCRP 比较匹配发生时计数器复位至零；CCRP 的值通过这种方式控制计数器的最大值。当比较器 P CCRP 比较匹配发生时，也会产生 TM 中断。记录 CCRP 溢出中断信号的值可以测量脉宽。通过设置 ST0IO1 和 ST0IO0 位选择 STP0I 引脚为上升沿，下降沿或双沿有效。如果 ST0IO1 和 ST0IO0 位设置为高，无论 STP0I 引脚发生哪种边沿转换，不会产生捕捉操作，但计数器继续运行。

当 STP0I 引脚与其它功能共用，TM 工作在输入捕捉模式时需多加注意。这是因为如果引脚被设为输入，那么该引脚上的任何电平转变都可能执行输入捕捉操作。ST0CCLR 和 ST0DPX 位在此模式中未使用。



捕捉输入模式

注：1. ST0M1，ST0M0=01 并通过 ST0IO1 和 ST0IO0 位设置有效边沿
2. TM 捕捉输入脚的有效边沿将计数器的值转移到 CCRA 中
3. ST0CCLR 和 ST0DPX 位未使用
4. 无输出功能 – ST0OC 和 ST0POL 位未使用
5. 计数器值由 CCRP 决定，在 CCRP 为“0”时，计数器计数值可达最大

周期型 TM – PTM

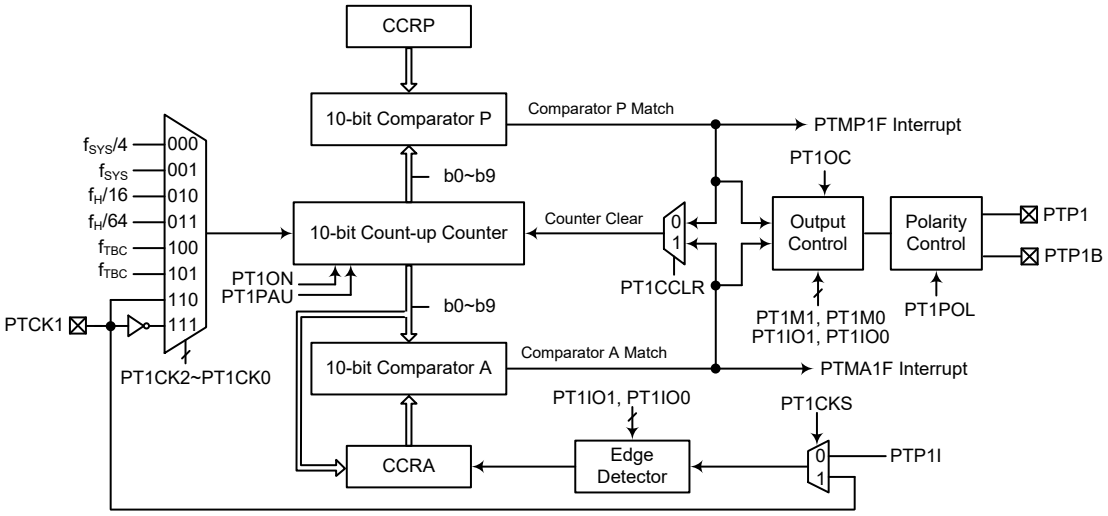
周期型 TM 包括 5 种工作模式，即比较匹配输出、定时 / 事件计数器、捕捉输入、单脉冲输出和 PWM 输出模式。周期型 TM 也由两个外部输入脚控制并驱动两个外部输出脚。

TM 类型	TM 输入引脚	TM 输出引脚
10-bit PTM1	PTCK1, PTP1I	PTP1, PTP1B

周期型 TM 操作

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

通过应用程序改变 10 位计数器值的唯一方法是使 PT1ON 位发生上升沿跳变清除计数器。此外，计数器溢出或比较匹配也会自动清除计数器。上述条件发生时，通常情况会产生 TM 中断信号。周期型 TM 可工作在不同的模式，可由包括来自输入脚的时钟源驱动，也可以控制输出脚。所有工作模式的设定都是通过设置相关寄存器来实现的。



周期型 TM 方框图

周期型 TM 寄存器介绍

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

寄存器名称	位							
	7	6	5	4	3	2	1	0
PTM1C0	PT1PAU	PT1CK2	PT1CK1	PT1CK0	PT1ON	—	—	—
PTM1C1	PT1M1	PT1M0	PT1IO1	PT1IO0	PT1OC	PT1POL	PT1CKS	PT1CCLR
PTM1DL	D7	D6	D5	D4	D3	D2	D1	D0
PTM1DH	—	—	—	—	—	—	D9	D8
PTM1AL	D7	D6	D5	D4	D3	D2	D1	D0
PTM1AH	—	—	—	—	—	—	D9	D8
PTM1RPL	D7	D6	D5	D4	D3	D2	D1	D0
PTM1RPH	—	—	—	—	—	—	D9	D8

10-bit 周期型 TM 寄存器列表

- **PTM1C0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PT1PAU	PT1CK2	PT1CK1	PT1CK0	PT1ON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7 **PT1PAU**：PTM1 计数器暂停控制位
0：运行
1：暂停
通过设置此位为高可使计数器暂停，清零此位恢复正常计数器操作。当处于暂停条件时，PTM1 保持上电状态并继续耗电。当此位由低到高转变时，计数器将保留其剩余值，直到此位再次改变为低电平，并从此值开始继续计数。

Bit 6 ~ 4 **PT1CK2~PT1CK0**：选择 PTM1 计数时钟位
000：$f_{SYS}/4$
001：f_{SYS}
010：$f_H/16$
011：$f_H/64$
100：f_{TBC}
101：f_{TBC}
110：PTCK1 上升沿
111：PTCK1 下降沿
此三位用于选择 PTM1 的时钟源。外部引脚时钟源能被选择在上升沿或下降沿有效。f_{SYS} 是系统时钟，f_H 和 f_{TBC} 是其它的内部时钟源，细节方面请参考振荡器章节。

Bit 3 **PT1ON**：PTM1 计数器 On/Off 控制位
0：Off
1：On
此位控制 PTM1 的总开关功能。设置此位为高则使能计数器使其运行，清零此位则除能 PTM1。清零此位将停止计数器并关闭 PTM1 减少耗电。当此位经由低到高转变时，内部计数器将复位清零；当此位经由高到低转换时，内部计数器将保持其剩余值，直到此位再次改变为高电平。
若 PTM1 处于比较匹配输出模式或 PWM 输出模式或单脉冲输出模式时，当 PT1ON 位经由低到高的转变时，PTM1 输出脚将复位至 PT1OC 位指定的初始值。

Bit 2 ~ 0 未使用，读为“0”

● **PTM1C1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PT1M1	PT1M0	PT1IO1	PT1IO0	PT1OC	PT1POL	PT1CKS	PT1CCLR
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 **PT1M1~PT1M0**：选择 PTM1 工作模式位
00：比较匹配输出模式
01：捕捉输入模式
10：PWM 输出模式或单脉冲输出模式
11：定时 / 计数器模式
这两位设置 PTM1 需要的工作模式。为了确保操作可靠，PTM1 应在 PT1M1 和 PT1M0 位有任何改变前先关掉。在定时 / 计数器模式，PTM1 输出脚状态未定义。

Bit 5 ~ 4 **PT1IO1~PT1IO0**：选择 PTM 输出功能位
比较匹配输出模式
00：无变化
01：输出低
10：输出高
11：输出翻转
PWM 模式 / 单脉冲输出模式
00：强制无效状态
01：强制有效状态
10：PWM 输出
11：单脉冲输出
捕捉输入模式
00：在 PTP1I 或 PTCK1 上升沿输入捕捉
01：在 PTP1I 或 PTCK1 下降沿输入捕捉
10：在 PTP1I 或 PTCK1 双沿输入捕捉
11：输入捕捉除能
定时 / 计数器模式
未使用
此两位用于决定在一定条件达到时 PTM1 输出脚如何改变状态。这两位值的选择取决于 PTM1 运行在哪种模式下。
在比较匹配输出模式下，PT1IO1 和 PT1IO0 位决定当从比较器 A 比较匹配输出发生时 PTM1 输出脚如何改变状态。当从比较器 A 比较匹配输出发生时 PTM1 输出脚能设为切换高、切换低或翻转当前状态。若此两位同时为 0 时，这个输出将不会改变。PTM1 输出脚的初始值通过 PTM1C1 寄存器的 PT1OC 位设置取得。注意，由 PT1IO1 和 PT1IO0 位得到的输出电平必须与通过 PT1OC 位设置的初始值不同，否则当比较匹配发生时，PTM1 输出脚将不会发生变化。在 PTM1 输出脚改变状态后，通过 PT1ON 位由低到高电平的转换复位至初始值。
在 PWM 输出模式，PT1IO1 和 PT1IO0 用于决定比较匹配条件发生时怎样改变 PTM1 输出脚的状态。PWM 输出功能通过这两位的变化进行更新。必须在 PTM1 关闭时改变 PT1IO1 和 PT1IO0 位的值。若在 PTM1 运行时改变 PT1IO1 和 PT1IO0 的值，PWM 输出的值是无法预料的。

Bit 3 **PT1OC**：PTP1/PTP1B 输出控制位
比较匹配输出模式
0：初始低
1：初始高
PWM 模式 / 单脉冲输出模式
0：低有效
1：高有效
这是 PTM1 输出脚输出控制位。它取决于 PTM1 此时正运行于比较匹配输出模式还是 PWM 模式 / 单脉冲输出模式。若 PTM1 处于定时 / 计数器模式，则其不受影响。在比较匹配输出模式时，比较匹配发生前其决定 TM 输出脚的逻辑电平值。在 PWM 模式时，其决定 PWM 信号是高有效还是低有效。

Bit 2 **PT1POL**：PTP1/PTP1B 输出极性控制位
0：同相
1：反相
此位控制 PTP1/PTP1B 输出脚的极性。此位为高时输出脚反相，为低时输出脚同相。若 PTM1 处于定时 / 计数器模式时其不受影响。

Bit 1 **PT1CKS**：选择 PTM1 捕捉触发源
0：来自 PTP1I 引脚
1：来自 PTCK1 引脚

Bit 0 **PT1CCLR**：选择 PTM1 计数器清零条件位
0：PTM1 比较器 P 匹配
1：PTM1 比较器 A 匹配
此位用于选择清除计数器的方法。周期型 TM 包括两个比较器 – 比较器 A 和比较器 P，两者都可以用作清除内部计数器。PT1CCLR 位设为高，计数器在比较器 A 比较匹配发生时被清除；此位设为低，计数器在比较器 P 比较匹配发生或计数器溢出时被清除。计数器溢出清除的方法仅在 CCRP 被清除为 0 时才能生效。PT1CCLR 位在 PWM 模式、单脉冲或输入捕捉模式时未使用。

● PTM1DL 寄存器

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

Bit 7 ~ 0 **PTM1DL**：PTM1 计数器低字节寄存器 bit 7~bit 0
PTM1 10-bit 计数器 bit 7~bit 0

● PTM1DH 寄存器

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

Bit 7 ~ 2 未使用，读为“0”

Bit 1 ~ 0 **PTM1DH**：PTM1 计数器高字节寄存器 bit 1~bit 0
PTM1 10-bit 计数器 bit 9~bit 8

● PTM1AL 寄存器

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

Bit 7 ~ 0 **PTM1AL**：PTM1 CCRA 低字节寄存器 bit 7~bit 0
PTM1 10-bit CCRA bit 7~bit 0

● **PTM1AH 寄存器**

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

Bit 7 ~ 2　未使用，读为“0”

Bit 1 ~ 0　**PTM1AH**：PTM1 CCRA 高字节寄存器 bit1~bit 0
PTM1 10-bit CCRA bit 9~bit 8

● **PTM1RPL 寄存器**

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

Bit 7 ~ 0　**PTM1RPL**：PTM1 CCRP 低字节寄存器 bit 7~bit 0
PTM1 10-bit CCRP bit 7~bit 0

● **PTM1RPH 寄存器**

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

Bit 7 ~ 2　未使用，读为“0”

Bit 1 ~ 0　**PTM1RPH**：PTM1 CCRP 高字节寄存器 bit 1~bit 0
PTM1 10-bit CCRP bit 9~bit 8

周期型 TM 工作模式

周期型 TM 有五种工作模式，即比较匹配输出模式、PWM 输出模式、单脉冲输出模式、捕捉输入模式或定时 / 计数器模式。通过设置 PTM1C1 寄存器的 PT1M1 和 PT1M0 位选择任意模式。

比较匹配输出模式

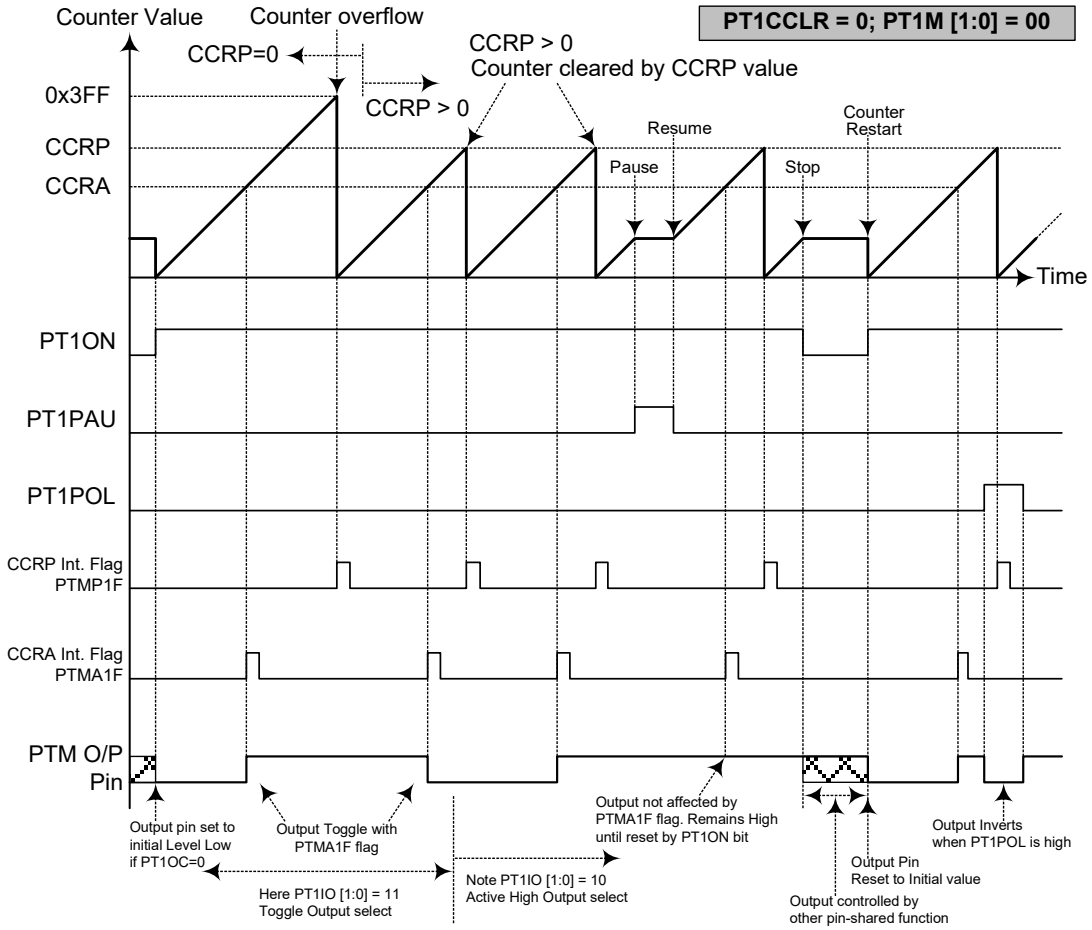
为使 TM 工作在此模式，PTM1C1 寄存器的 PT1M1 和 PT1M0 位需要设置为“00”。当工作在该模式，一旦计数器使能并开始计数，有三种方法来清零，分别是：计数器溢出，比较器 A 比较匹配发生和比较器 P 比较匹配发生。当 PT1CCLR 位为低，有两种方法清除计数器。一种是比较器 P 比较匹配发生，另一种是 CCRP 所有位设置为零并使得计数器溢出。此时，比较器 A 和比较器 P 的请求标志位 PTMA1F 和 PTMP1F 将分别置起。

如果 PTM1C1 寄存器的 PT1CCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅 PTMA1F 中断请求标志产生。所以当 PT1CCLR 为高时，不会产生 PTMP1F 中断请求标志。在比较匹配输出模式中，CCRA 寄存器值不能设为“0”。

如果 CCRA 为“0”，当 CCRA 达到最大值 0x3FF 时，计数器将溢出，不会产生 PTMA1F 中断请求标志。

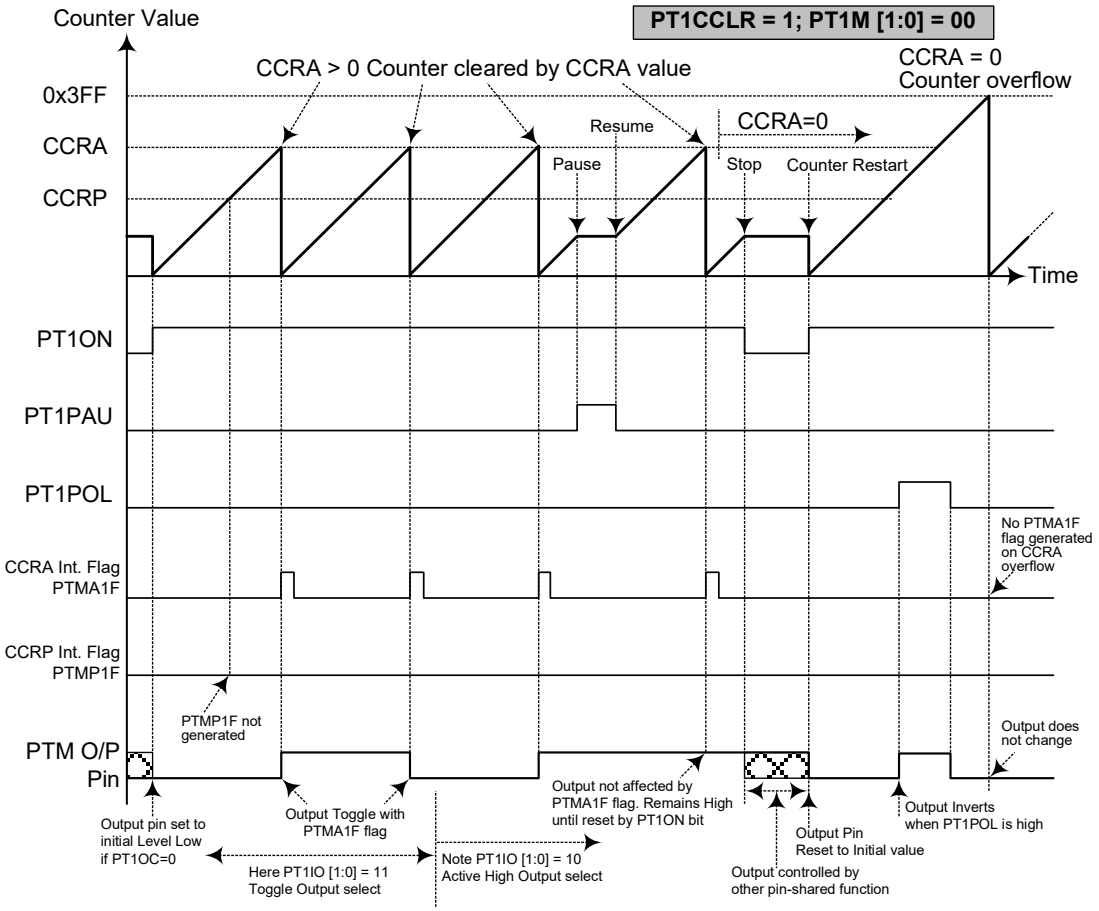
正如该模式名所言，当比较匹配发生后，TM 输出脚状态改变。当比较器 A 比较匹配发生后 PTMA1F 中断请求标志产生时，TM 输出脚状态改变。比较器 P

比较匹配发生时产生的 PTMP1F 标志不影响 TM 输出脚。TM 输出脚状态改变方式由 PTM1C1 寄存器中 PT1IO1 和 PT1IO0 位决定。当比较器 A 比较匹配发生时，PT1IO1 和 PT1IO0 位决定 TM 输出脚输出高，低或翻转当前状态。TM 输出脚初始值，在 PT1ON 位由低到高电平的变化后通过 PT1OC 位设置。注意，若 PT1IO1 和 PT1IO0 位同时为 0 时，引脚输出不变。



比较器匹配输出模式 – PT1CCLR = 0

注：1. PT1CCLR=0，比较器 P 匹配将清除计数器
2. TM 输出脚仅由 PTMA1F 标志位控制
3. 在 PT1ON 上升沿 TM 输出脚复位至初始值



比较器匹配输出模式 – PT1CCLR = 1

注：1. PT1CCLR=1，比较器 A 匹配将清除计数器
2. TM 输出脚仅由 PTMA1F 标志位控制
3. 在 PT1ON 上升沿 TM 输出脚复位至初始值
4. 当 PT1CCLR=1 时，不会产生 PTMP1F 标志

定时 / 计数器模式

为使 TM 工作在此模式，PTM1C1 寄存器的 PT1M1 和 PT1M0 位需要设置为“11”。定时 / 计数器模式与比较输出模式操作方式相同，并产生同样的中断请求标志。不同的是，在定时 / 计数器模式下 TM 输出脚未使用。因此，比较匹配输出模式中的描述和时序图可以适用于此功能。该模式中未使用的 TM 输出脚通过引脚共用功能选择寄存器设置用作普通 I/O 脚或其它功能。

PWM 输出模式

为使 TM 工作在此模式，PTM1C1 寄存器的 PT1M1 和 PT1M0 位需要设置为“10”，且 PT1IO1 和 PT1IO0 位也需要设置为“10”。TM 的 PWM 功能在马达控制，加热控制，照明控制等方面十分有用。给 TM 输出脚提供一个频率固定但占空比可调的信号，将产生一个有效值等于 DC 均方根的 AC 方波。

由于 PWM 波形的周期和占空比可调，其波形的选择就较为灵活。在 PWM 模式中，PT1CCLR 位对 PWM 周期无影响。CCRP 和 CCRA 寄存器都用于控制 PWM 方波。CCRP 寄存器通过清除内部计数从而控制 PWM 周期，CCRA 寄存器设置 PWM 的占空比。PWM 波形的周期和占空比由 CCRP 和 CCRA 寄存器的值控制。

当比较器 A 或比较器 P 比较匹配发生时，CCRA 和 CCRP 中断标志位分别产生。PTM1C1 寄存器的 PT1OC 位选择 PWM 波形的极性，PT1IO1 和 PT1IO0 位使能 PWM 输出或强制 TM 输出脚为高电平或低电平。PT1POL 位用于 PWM 输出波形的极性反相控制。

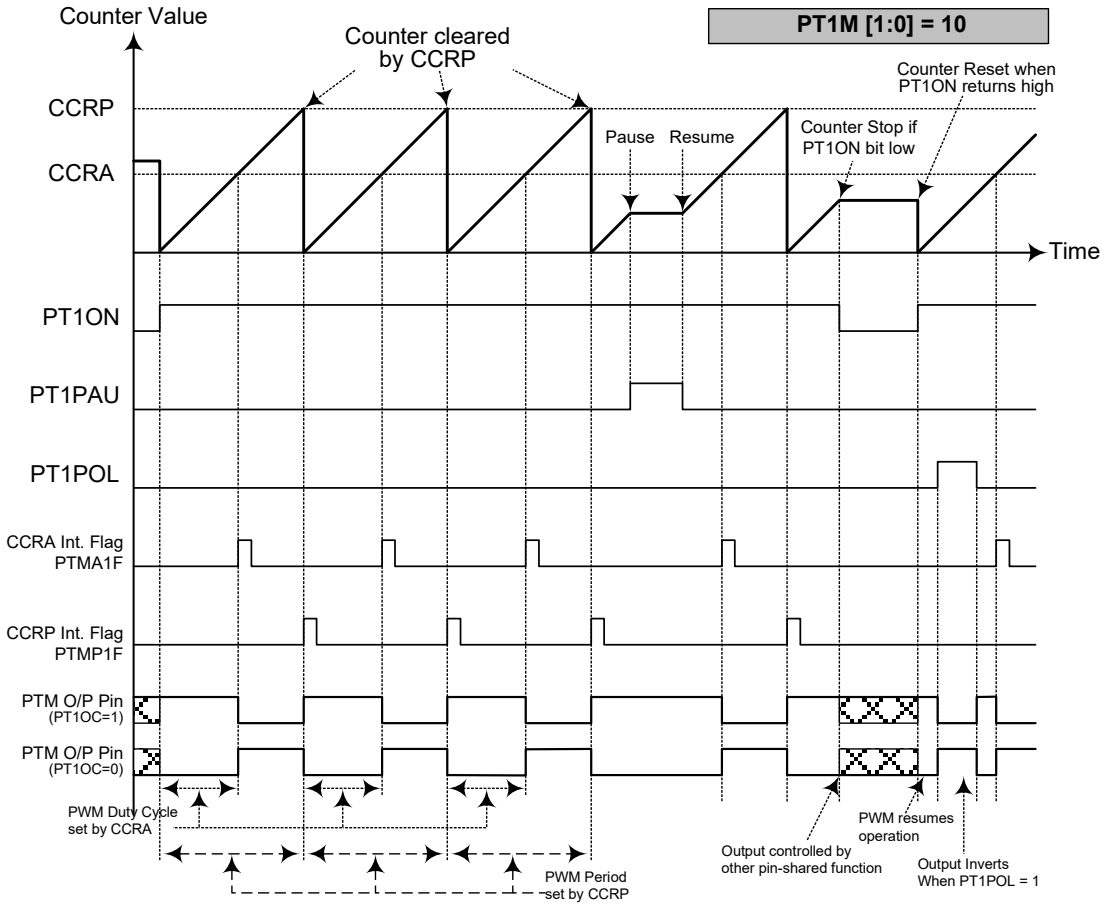
- **10-bit PTM1，PWM 模式，边沿对齐模式**

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

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

PTM1 PWM 输出频率 =(f_{SYS}/4)/512=f_{SYS}/2048=7.8125kHz，duty=128/512=25%，

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



PWM 输出模式

注：1. CCRP 清除计数器

2. 计数器清零并设置 PWM 周期

3. 当 PT1IO[1:0]=00 或 01，PWM 功能不变

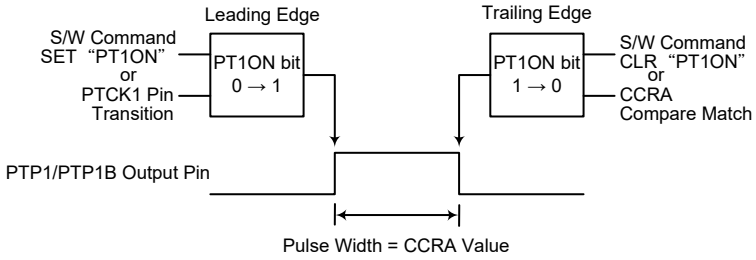
4. PT1CCLR 位对 PWM 功能无影响

单脉冲输出模式

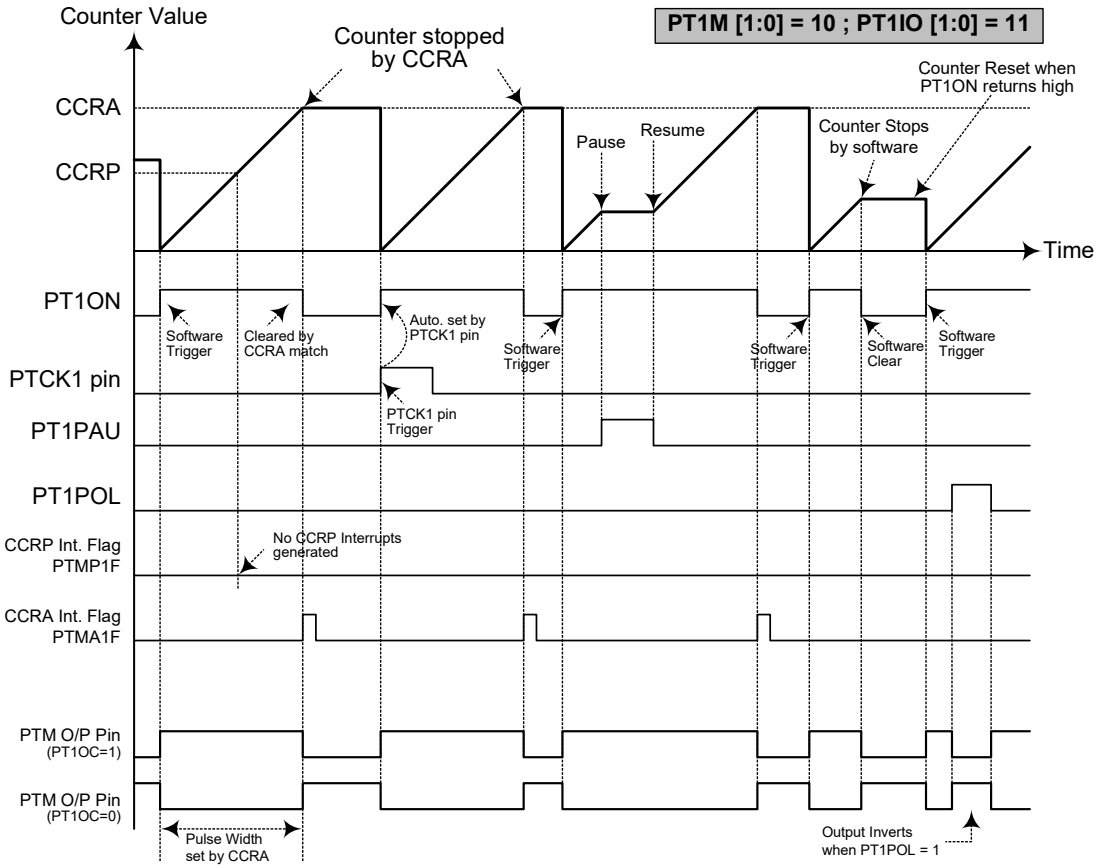
为使 TM 工作在此模式，PT1M1 和 PT1M0 位需要设置为“10”，并且相应的 PT1IO1 和 PT1IO0 需要设置为“11”。正如模式名所言，单脉冲输出模式，在 TM 输出脚将产生一个脉冲输出。

通过应用程序控制 PT1ON 位由低到高的转变来触发脉冲前沿输出。而处于单脉冲模式时，PT1ON 位可在 PTCK1 脚发生有效边沿跳转时自动由低转变为高，进而开始单脉冲输出。当 PT1ON 位转变为高电平时，计数器将开始运行，并产生脉冲前沿。通过应用程序使 PT1ON 位清零或比较器 A 比较匹配发生时，产生脉冲后沿。

而比较器 A 比较匹配发生时，会自动清除 PT1ON 位并产生单脉冲输出后沿。CCRA 的值通过这种方式控制脉冲宽度。比较器 A 比较匹配发生时，也会产生 TM 中断。PT1ON 位在计数器重启时会发生由低到高的转变，此时计数器才复位至零。在单脉冲模式中，CCRP 寄存器和 PT1CCLR 位未使用。



单脉冲产生示意图



单脉冲模式

注：1. 通过 CCRA 匹配停止计数器

2. CCRP 未使用

3. 通过 PTCK1 脚或设置 PT1ON 位为高来触发脉冲

4. PTCK1 脚有效沿会自动置位 PT1ON

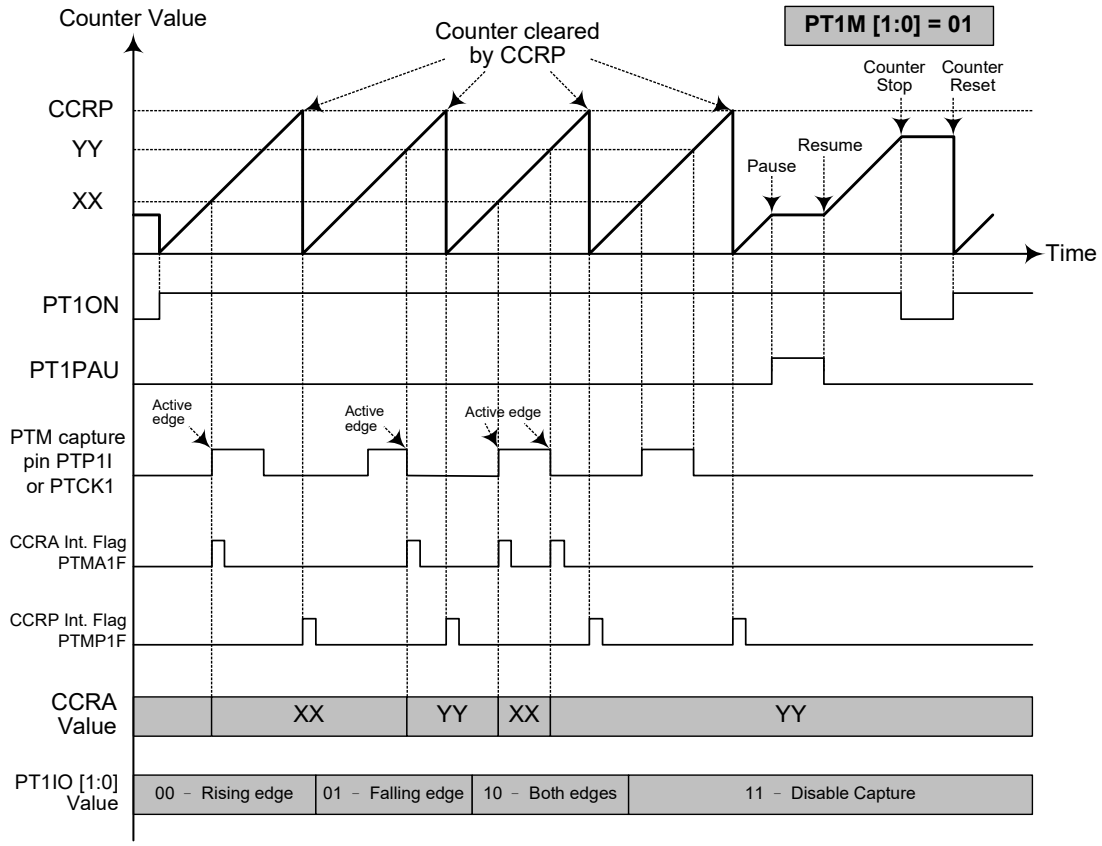
5. 单脉冲模式中，PT1IO[1:0] 需置位“11”，且不能更改。

捕捉输入模式

为使 TM 工作在此模式，PTM1C1 寄存器的 PT1M1 和 PT1M0 位需要设置为“01”。此模式使能外部信号捕捉并保存内部计数器当前值，因此被用于诸如脉冲宽度测量的应用中。PTP1I 或 PTCK1 引脚上的外部信号，通过设置 PTM1C1 寄存器的 PT1CKS 位选择。可通过设置 PTM1C1 寄存器的 PT1IO1 和 PT1IO0 位选择有效边沿类型，即上升沿，下降沿或双沿有效。通过应用程序将 PT1ON 位由低到高转变时，计数器启动。

当 PTP1I 或 PTCK1 引脚出现有效边沿转换时，计数器当前值被锁存到 CCRA 寄存器，并产生 TM 中断。不考虑 PTP1I 或 PTCK1 引脚事件，计数器继续工作直到 PT1ON 位发生下降沿跳变。当 CCRP 比较匹配发生时计数器复位至零；CCRP 的值通过这种方式控制计数器的最大值。当比较器 P CCRP 比较匹配发生时，也会产生 TM 中断。记录 CCRP 溢出中断信号的值可以测量长脉宽。通过设置 PT1IO1 和 PT1IO0 位选择 PTP1I 或 PTCK1 引脚为上升沿，下降沿或双沿有效。不考虑 PTP1I 或 PTCK1 引脚事件，如果 PT1IO1 和 PT1IO0 位都设为高，不会产生捕捉操作，但计数器继续运行。

当 PTP1I 或 PTCK1 引脚与其它功能共用，TM 工作在输入捕捉模式时需多加注意。这是因为如果引脚被设为输入，那么该引脚上的任何电平转变都可能执行输入捕捉操作。PT1CCLR，PT1OC 和 PT1POL 位在此模式中未使用。



捕捉输入模式

注：1. PT1M[1:0]=01 并通过 PT1IO[1:0] 位设置有效边沿
2. TM 捕捉输入脚的有效边沿将计数器的值转移到 CCRA 中
3. PT1CCLR 位未使用
4. 无输出功能，PT1OC 和 PT1POL 位未使用
5. 计数器值由 CCRP 决定，在 CCRP 为“0”时，计数器计数值可达最大

A/D 转换器

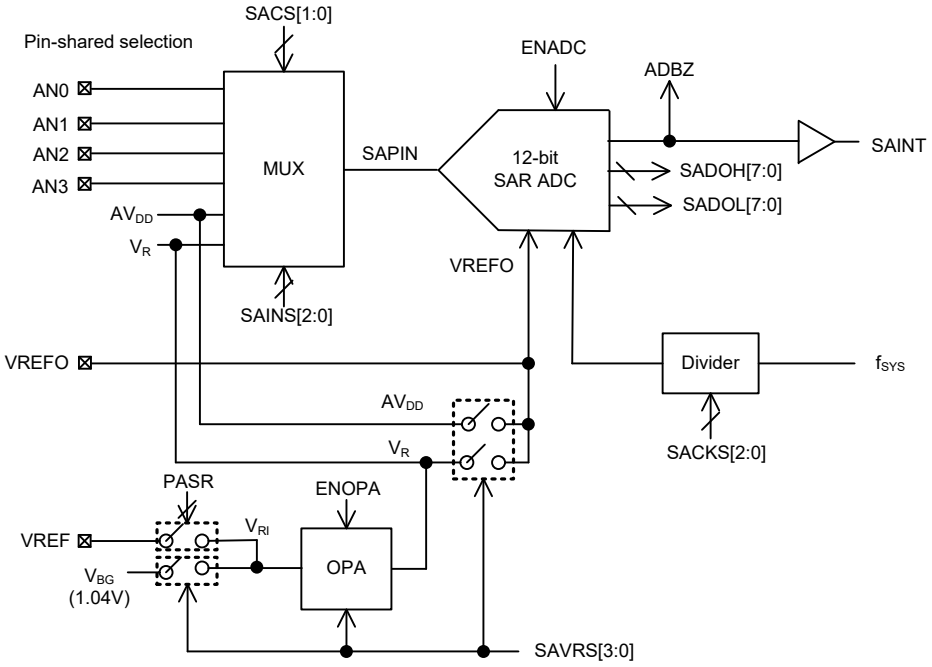
对于大多数电子系统而言，处理现实世界的模拟信号是共同的需求。为了完全由单片机来处理这些信号，首先需要通过 A/D 转换器将模拟信号转换成数字信号。将 A/D 转换器电路集成入单片机，可有效的减少外部器件，随之而来，具有降低成本和减少器件空间需求的优势。

A/D 简介

此单片机包含一个 4 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。可通过正确的设置 SAINS2~SAINS0 位及 SACS1~SACS0 位选择外部或内部模拟信号作为 A/D 转换输入。需要注意的是若选择内部信号作为 A/D 输入，需额外再正确设置引脚共用功能控制寄存器相关位，防止外部信号和内部信号发生冲突。具体请参考相应寄存器介绍以及“A/D 输入信号”部分内容。

外部输入通道数	A/D 通道选择位	外部信号输入引脚
4	SAINS2~SAINS0 SACS1~SACS0	AN0~AN3

下图显示了 A/D 转换器内部结构和相关的寄存器。



A/D 转换器结构

A/D 转换寄存器介绍

A/D 转换器的所有工作由五个寄存器控制。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
SADOL (ADRFS=0)	D3	D2	D1	D0	—	—	—	—
SADOL (ADRFS=1)	D7	D6	D5	D4	D3	D2	D1	D0
SADOH (ADRFS=0)	D11	D10	D9	D8	D7	D6	D5	D4
SADOH (ADRFS=1)	—	—	—	—	D11	D10	D9	D8
SADC0	START	ADBZ	ENADC	ADRFS	—	—	SACS1	SACS0
SADC1	SAINS2	SAINS1	SAINS0	—	—	SACKS2	SACKS1	SACKS0
SADC2	ENOPA	VBGEN	—	—	SAVRS3	SAVRS2	SAVRS1	SAVRS0

A/D 转换寄存器列表

A/D 转换器数据寄存器 – SADOL, SADOH

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

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

A/D 转换器数据寄存器

A/D 转换器控制寄存器 – SADC0, SADC1, SADC2, PASR

寄存器 SADC0、SADC1 和 SADC2 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，哪些引脚作为模拟输入，哪些作为 I/O 口，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 SADC0 的 SACS1~SACS0 位用于选择哪个外部通道作为 ADC 输入通道。寄存器 SADC1 的 SAINS2~SAINS0 位用于选择使用来自外部或内部的信号作为 A/D 转换的输入。设置 SAINS2~SAINS0 为 000，可选择外部信号作为 A/D 转换的输入，再设置 SACS1~SACS0 位来选择外部通道编号。设置 SAINS2~SAINS0 为其它值，可选择来自内部的信号作为 A/D 转换输入。具体可参考下面寄存器的具体描述。若选择 V_{REF} 或 V_{BG} 作为 ADC 输入或 ADC 参考电压，则需先置高 ENOPA 位使能 OPA 功能。

需注意的是，若程序选择同时将内部和外部信号作为 A/D 转换的输入，硬件将会自动选择内部信号。另外，若将外部参考电压 V_{REF} 及内部参考电压 V_{BG} 同时设为 ADC 参考电压，硬件只会选择内部参考电压 V_{BG} 作为此参考电压。

所有 A/D 模拟输入引脚都与 PA 端口的 I/O 引脚及其它功能共用。使用引脚共用功能选择寄存器 PASR 的相应位，可以将它们设置为 A/D 转换器模拟输入脚或具有其它功能。如果将引脚设置为 A/D 输入，其它引脚功能将会失效，且引脚的上拉电阻会自动断开。

● SADC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	START	ADBZ	ENADC	ADRFS	—	—	SACS1	SACS0
R/W	R/W	R	R/W	R/W	—	—	R/W	R/W
POR	0	0	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 ADBZ 为“0”
1→0：启动 A/D 转换，并且设置 ADBZ 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　**ADBZ**：A/D 转换忙碌标志位
0：A/D 转换结束或未开始转换
1：A/D 转换中
此位用于表明 A/D 转换过程是否完成。当 START 位由低变为高再变为低时，ADBZ 位为高，表明 A/D 转换已初始化。A/D 转换结束后，此位被清零。

Bit 5　**ENADC**：A/D 转换器除能 / 使能控制位
0：除能
1：使能
当 A/D 转换除能时，A/D 数据寄存器 SADOH 和 SADOL 的内容将不变。

Bit 4　**ADRFS**：A/D 数据格式控制位
0：A/D 数据格式 → SADOH=D[11:4]; SADOL=D[3:0]
1：A/D 数据格式 → SADOH=D[11:8]; SADOL=D[7:0]

Bit 3 ~2　未使用，读为“0”

Bit 1 ~ 0　**SACS1 ~ SACS0**：A/D 外部模拟通道输入选择位
00：AN0
01：AN1
10：AN2
11：AN3

● SADC1 寄存器

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

Bit 7~5　**SAINS2~SAINS0**：A/D 输入信号选择位
000：外部模拟通道输入
001：AV_{DD}
010：$AV_{DD}/2$
011：$AV_{DD}/4$
100：外部模拟通道输入
101：V_R
110：$V_R/2$
111：$V_R/4$
注：V_R 为 OPA 输出电压，可取值：V_{REF}, $V_{REF}\times2$, $V_{REF}\times3$, $V_{REF}\times4$, $V_{BG}\times2$, $V_{BG}\times3$, $V_{BG}\times4$

Bit 4~3　未使用，读为“0”

Bit 2~0　**SACKS2 ~ SACKS0**：选择 A/D 时钟源
000：f_{SYS}
001：$f_{SYS}/2$
010：$f_{SYS}/4$

011：$f_{SYS}/8$
100：$f_{SYS}/16$
101：$f_{SYS}/32$
110：$f_{SYS}/64$
111：$f_{SYS}/128$

● **SADC2 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	ENOPA	VBGEN	—	—	SAVRS3	SAVRS2	SAVRS1	SAVRS0
R/W	R/W	R/W	—	—	R/W	R/W	R/W	R/W
POR	0	0	—	—	0	0	0	0

Bit 7 **ENOPA**：OPA 使能控制位
0：除能
1：使能

Bit 6 **VBGEN**：Bandgap 控制位
0：除能
1：使能

Bit 5~4 未使用，读为“0”

Bit 3~0 **SAVRS3 ~ SAVRS0**：ADC 参考电压选择位
0000：AV_{DD}
0001：V_{REF}
0010：$V_{REF}\times2$
0011：$V_{REF}\times3$
0100：$V_{REF}\times4$
1001：禁止使用
1010：$V_{BG}\times2$
1011：$V_{BG}\times3$
1100：$V_{BG}\times4$
11xx：AV_{DD}

注：1. 若选择 V_{REF}，$V_{REF}\times2$，$V_{REF}\times3$，$V_{REF}\times4$ 作为 ADC 参考电压，则：需设置引脚共用功能选择位 PAS2、PAS1 分别为 1、0，选择 VREF 引脚功能
2. V_{BG} = 1.04V
3. 若 SAVRS3=1，OPA 将选择 V_{BG} 作为输入
4. 如果同时选择外部参考电压 V_{REF} 和内部参考电压 V_{BG} 作为 ADC 参考电压，硬件将会选择内部 V_{BG} 作为输入参考电压。

A/D 操作

SADC0 寄存器中的 START 位，用于打开和复位 A/D 转换器。当单片机设置此位从逻辑低到逻辑高，然后再到逻辑低，就会开始一个模数转换周期。当 START 位从逻辑低到逻辑高，但不再回到逻辑低时，SADC0 寄存器中的 ADBZ 位为“0”，复位模数转换器。START 位用于控制内部模数转换器的开启动作。

SADC0 寄存器中的 ADBZ 位用于表明模数转换过程是否完成。当 START 位从逻辑低到逻辑高时，A/D 转换器复位，ADBZ 位被清零。A/D 转换成功启动后，ADBZ 位被自动置为“1”。在转换周期结束后，ADBZ 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跳转到相应的 A/D 内部中断地址。如果 A/D 内部中断被禁止，可以让单片机轮询 SADC0 寄存器中的 ADBZ 位，检查此位是否被清除，作为另一种侦测 A/D 转换周期结束的方法。

A/D 转换器的时钟源为系统时钟 f_{SYS} 或其分频，而分频系数由 SADC1 寄存器中的 SACKS2~SACKS0 位决定。

虽然 A/D 时钟源是由系统时钟 f_{SYS} 和 SACKS2~SACKS0 位决定，但可选择的最大 A/D 时钟源则有一些限制。由于允许的 A/D 时钟周期 t_{ADCK} 的范围为 0.5μs ~10μs，所以选择系统时钟速度时就必须小心。如果系统时钟速度为 4MHz 时，SACKS2~SACKS0 位不能设为“000B”或“11xB”。必须保证设置的 A/D 转换时钟周期不小于时钟周期的最小值或大于时钟周期的最大值，否则将会产生不准确的 A/D 转换值。

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

A/D 转换器参考电压来自 A/D 电源电压 AV_{DD} 或外部参考源引脚 VREF 也可选择来自 V_{BG}，可通过 SAVRS3~SAVRS0 位来选择。若选择参考电压来自 VREF 引脚。由于 VREF 引脚与其它功能共用，当选择 VREF 参考电压时，需合理设置相关引脚共用控制位选择 VREF 引脚功能且除能其它共用引脚功能。若选择 V_{REF} 或 V_{BG} 作为 ADC 输入或 ADC 参考电压，需通过设置 ENOPA 为 1，使能 OPA 功能。

参考电压	SAVRS[3:0]]	描述
AV_{DD}	0000	A/D 转换器参考电压来自内部 AV_{DD}
V_{REF}	0001	A/D 转换器参考电压来自外部 V_{REF}
V_{REF}×2	0010	A/D 转换器参考电压来自外部 V_{REF}×2
V_{REF}×3	0011	A/D 转换器参考电压来自外部 V_{REF}×3
V_{REF}×4	0100	A/D 转换器参考电压来自外部 V_{REF}×4
V_{BG}×2	1010	A/D 转换器参考电压来自 V_{BG}×2
V_{BG}×3	1011	A/D 转换器参考电压来自 V_{BG}×3
V_{BG}×4	1100	A/D 转换器参考电压来自 V_{BG}×4

A/D 转换器参考电压选择

A/D 输入引脚

所有的 A/D 模拟输入引脚都与 I/O 引脚及其它功能共用。使用 PASR 寄存器中的相关控制位可以将它们设置为 A/D 转换器模拟输入脚或其它功能。如果引脚的对应控制位选择 A/D 输入功能，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，PAC 端口控制寄存器不需要为使能 A/D 输入而先设置为输入模式，当引脚共用控制位使能 A/D 输入时，端口控制寄存器的状态将被重置。

A/D 转换器有一个外部参考电压输入引脚，VREF，也可选择内部电压作为参考电压，通过设置 SADC2 寄存器的 SAVRS3~SAVRS0 位进行设置。需注意的是输入的模拟信号大小不能超过参考电压值。

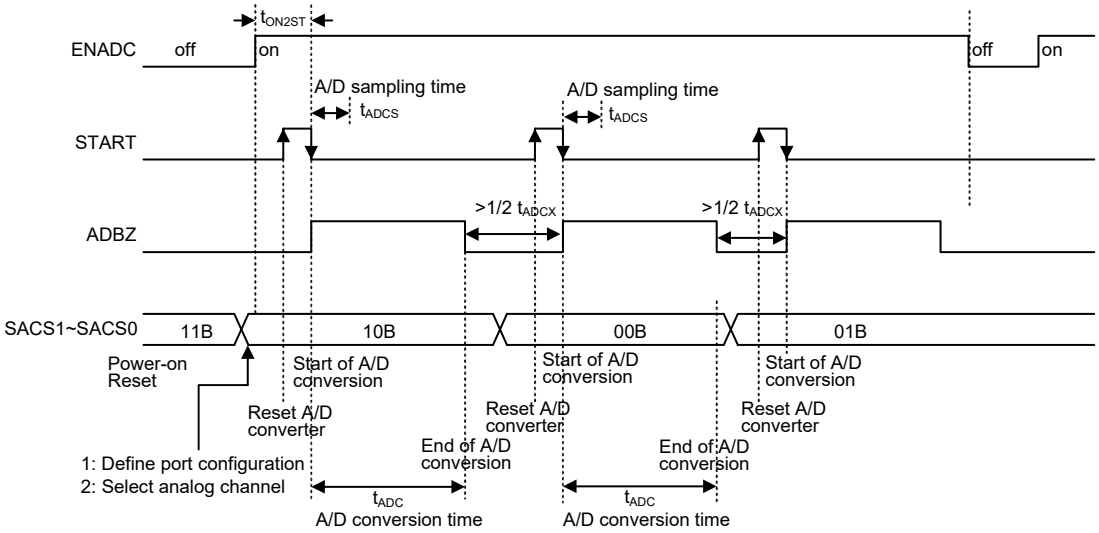
A/D 转换率及时序图

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

完整的 A/D 转换时间，t_{ADC}，一共需要 16 个 A/D 时钟周期。

最大 A/D 转换率 = A/D 时钟周期 ÷ 16

当前转换结束至下一轮转换开始有一段时间限制。当前 A/D 转换结束后，转换后的值将被存在 A/D 数据寄存器并锁存，耗时约半个 A/D 时钟周期。若 A/D 转换结束后半个时钟周期内把 START 设为“1”，则存在 A/D 数据寄存器的转换值将被改写。因此，当前转换结束至下一轮 A/D 转换开始的时间间隔应大于半个 A/D 时钟周期，且应在 START 位再次置位前读取 A/D 转换结果。



A/D 转换时序图

A/D 转换步骤

下面概述实现 A/D 转换过程的各个步骤。

- 步骤 1
  通过 SADC1 寄存器中的 SACKS2~SACKS0 位，选择所需的 A/D 转换时钟。
- 步骤 2
  将 SADC0 寄存器中的 ENADC 位置高以使能 A/D。
- 步骤 3
  选择连接至内部 A/D 转换器的信号。
- 步骤 4
  若选择外部通道输入，设置 SAINS2~SAINS0 为 000，接着设置 SACS 选择将要连接的外部通道。
  若选择内部模拟信号，设置 SAINS2~SAINS0 选择需要的内部信号。
- 步骤 5
  通过 SAVRS3~SAVRS0 位选择参考电压。
  注：若选择 V_{REF}，需设置位 PAS2、PAS1 为 1、0。
- 步骤 6
  设置 ADRFS 位选择 A/D 转换器输出数据格式。
- 步骤 7
  如果要使用中断，则中断控制寄存器需要正确地设置，以确保 A/D 中断功能是激活的。总中断控制位 EMI 需要置位为“1”，以及 A/D 转换器中断位 ADE 也需要置位为“1”。

- 步骤 8

  现在可以通过设置 SADC0 寄存器中的 START 位从“0”到“1”再回到“0”，开始模数转换的过程。

- 步骤 9

  可以轮询 SADC0 寄存器中的 ADBZ 位，检查模数转换过程是否完成。当此位为逻辑高时，表示转换正在进行中。当此位为逻辑低时，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 SADOL 和 SADOH 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 SADC0 寄存器中 ADBZ 位的状态的方法来检查转换过程是否结束时，则中断使能的步骤可以省略。

编程注意事项

在编程时，如果 A/D 转换器未使用，通过设置 SADC0 寄存器中的 ENADC 为低，关闭 A/D 内部电路以减少电源功耗。此时，无论输入脚的模拟电压为何，内部 A/D 转换器电路不产生功耗。当 A/D 输入引脚用作普通 I/O 口时，请注意，若输入电压为无效的逻辑电平，则可能增加功耗。

A/D 转换功能

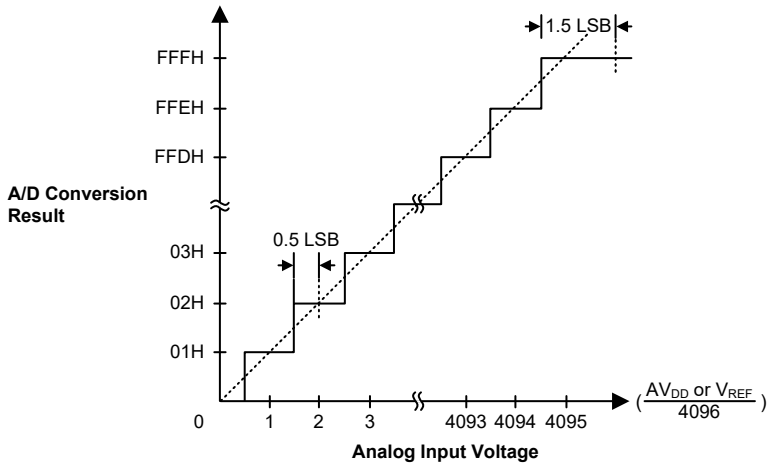
单片机含有一组 12 位的 A/D 转换器，它们转换的最大值可达 FFFH。由于模拟输入最大值等于 AV_{DD} 或 V_{REF} 的电压值，因此每一位可表示 AV_{DD} 或 V_{REF}/4096 的模拟输入值。

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

通过下面的等式可估算 A/D 转换器输入电压值：

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

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 AV_{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 SADC0 寄存器中的 ADBZ 位来判断 A/D 转换是否完成；第二个范例则使用中断的方式判断。

范例 : 使用轮询 ADBZ 的方式来检测转换结束

```
clr  ADE                    ; disable ADC interrupt
mov  a,0BH
mov  SADC1,a                ; select fSYS/8 as A/D clock and switch off the
                            ; bandgap reference voltage
set  ENADC
mov  a,03h                  ; setup PASR to configure pins AN0
mov  PASR,a
mov  a,20h
mov  SADC0,a                ; enable and connect AN0 channel to A/D converter
:
start_conversion:
clr  START                  ; high pulse on start bit to initiate conversion
set  START                  ; reset A/D
clr  START                  ; start A/D
polling_EOC:
sz   ADBZ                   ; poll the SADC0 register ADBZ bit to detect end
                            ; of A/D conversion
jmp  polling_EOC            ; continue polling
mov  a,SADOL                ; read low byte conversion result value
mov  SADOL_buffer,a         ; save result to user defined register
mov  a,SADOH                ; read high byte conversion result value
mov  SADOH_buffer,a         ; save result to user defined register
:
:
jmp  start_conversion       ; start next a/d conversion
```

范例 : 使用中断的方式来检测转换结束

```
clr  ADE                    ; disable ADC interrupt
mov  a,0BH
mov  SADC1,a                ; select fSYS/8 as A/D clock and switch off VBG
set  ENADC
mov  a,03h                  ; setup PASR to configure pins AN0
mov  PASR,a
mov  a,20h
mov  SADC0,a                ; enable and connect AN0 channel to A/D converter
Start_conversion:
clr  START                  ; high pulse on START bit to initiate conversion
set  START                  ; reset A/D
clr  START                  ; start A/D
clr  ADF                    ; clear ADC interrupt request flag
set  ADE                    ; enable ADC interrupt
set  EMI                    ; enable global interrupt
:
:
                            ; ADC interrupt service routine
ADC_ISR:
mov  acc_stack,a            ; save ACC to user defined memory
mov  a,STATUS
mov  status_stack,a         ; save STATUS to user defined memory
:
```

```
:
mov  a,SADOL                ; read low byte conversion result value
mov  SADOL_buffer,a         ; save result to user defined register
mov  a,SADOH                ; read high byte conversion result value
mov  SADOH_buffer,a         ; save result to user defined register
:
:
EXIT_INT_ISR:
mov  a,status_stack
mov  STATUS,a               ; restore STATUS from user defined memory
mov  a,acc_stack            ; restore ACC from user defined memory
reti
```

中断

中断是单片机一个重要功能。当外部事件或内部功能如定时器模块有效，并且产生中断时，系统会暂时中止当前的程序而转到执行相对应的中断服务程序。此单片机提供 1 个外部中断和多个内部中断功能，外部中断由 INT 引脚动作产生，而内部中断由各种内部功能，如定时器模块、时基、EEPROM 和 A/D 转换器等产生。

中断寄存器

中断控制基本上是在一定单片机条件发生时设置请求标志位，应用程序中中断使能位的设置是通过位于专用数据存储器中的一系列寄存器控制的。寄存器总的分为三类。第一类是 INTC0~INTC1 寄存器，用于设置基本的中断；第二类是 MFI0~MFI1 寄存器，用于设置多功能中断；最后一种有 INTEG 寄存器，用于设置外部中断边沿触发类型。

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

功能	使能位	请求标志	注释
总中断	EMI	—	—
INT 脚	INTE	INTF	—
A/D 转换器	ADE	ADF	—
多功能	MFnE	MFnF	n=0 或 1
时基	TBnE	TBnF	n=0 或 1
EEPROM	DEE	DEF	—
STM	STMA0E	STMA0F	—
	STMP0E	STMP0F	—
PTM	PTMA1E	PTMA1F	—
	PTMP1E	PTMP1F	—

中断寄存器位命名模式

寄存器名称	位							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	—	—	INT0S1	INT0S0
INTC0	—	TB1F	TB0F	INTF	TB1E	TB0E	INTE	EMI
INTC1	MF1F	ADF	DEF	MF0F	MF1E	ADE	DEE	MF0E
MFI0	—	—	STMA0F	STMP0F	—	—	STMA0E	STMP0E
MFI1	—	—	PTMA1F	PTMP1F	—	—	PTMA1E	PTMP1E

中断寄存器列表

- **INTEG 寄存器**

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

Bit 7~2 未使用，读为“0”

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

- **INTC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	TB1F	TB0F	INTF	TB1E	TB0E	INTE	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 **TB1F**：时基 1 中断请求标志位
0：无请求
1：中断请求

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

Bit 4 **INTF**：INT 中断请求标志位
0：无请求
1：中断请求

Bit 3 **TB1E**：时基 1 中断控制位
0：除能
1：使能

Bit 2 **TB0E**：时基 0 中断控制位
0：除能
1：使能

Bit 1 **INTE**：INT 中断控制位
0：除能
1：使能

Bit 0 **EMI**：总中断控制位
0：除能
1：使能

● **INTC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	MF1F	ADF	DEF	MF0F	MF1E	ADE	DEE	MF0E
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 **MF1F**：多功能中断 1 请求标志位
0：无请求
1：中断请求

Bit 6 **ADF**：A/D 转换器中断请求标志位
0：无请求
1：中断请求

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

Bit 4 **MF0F**：多功能中断 0 请求标志位
0：无请求
1：中断请求

Bit 3 **MF1E**：多功能中断 1 控制位
0：除能
1：使能

Bit 2 **ADE**：A/D 转换器中断控制位
0：除能
1：使能

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

Bit 0 **MF0E**：多功能中断 0 控制位
0：除能
1：使能

● **MFI0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	STMA0F	STMP0F	—	—	STMA0E	STMP0E
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 未使用，读为“0”

Bit 5 **STMA0F**：STM 比较器 A 匹配中断请求标志位
0：无请求
1：中断请求

Bit 4 **STMP0F**：STM 比较器 P 匹配中断请求标志位
0：无请求
1：中断请求

Bit 3~2 未使用，读为“0”

Bit 1 **STMA0E**：STM 比较器 A 匹配中断控制位
0：除能
1：使能

Bit 0 **STMP0E**：STM 比较器 P 匹配中断控制位
0：除能
1：使能

● **MFI1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PTMA1F	PTMP1F	—	—	PTMA1E	PTMP1E
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6　未使用，读为“0”

Bit 5　**PTMA1F**：PTM1 比较器 A 匹配中断请求标志位
0：无请求
1：中断请求

Bit 4　**PTMP1F**：PTM1 比较器 P 匹配中断请求标志位
0：无请求
1：中断请求

Bit 3~2　未使用，读为“0”

Bit 1　**PTMA1E**：PTM1 比较器 A 匹配中断控制位
0：除能
1：使能

Bit 0　**PTMP1E**：PTM1 比较器 P 匹配中断控制位
0：除能
1：使能

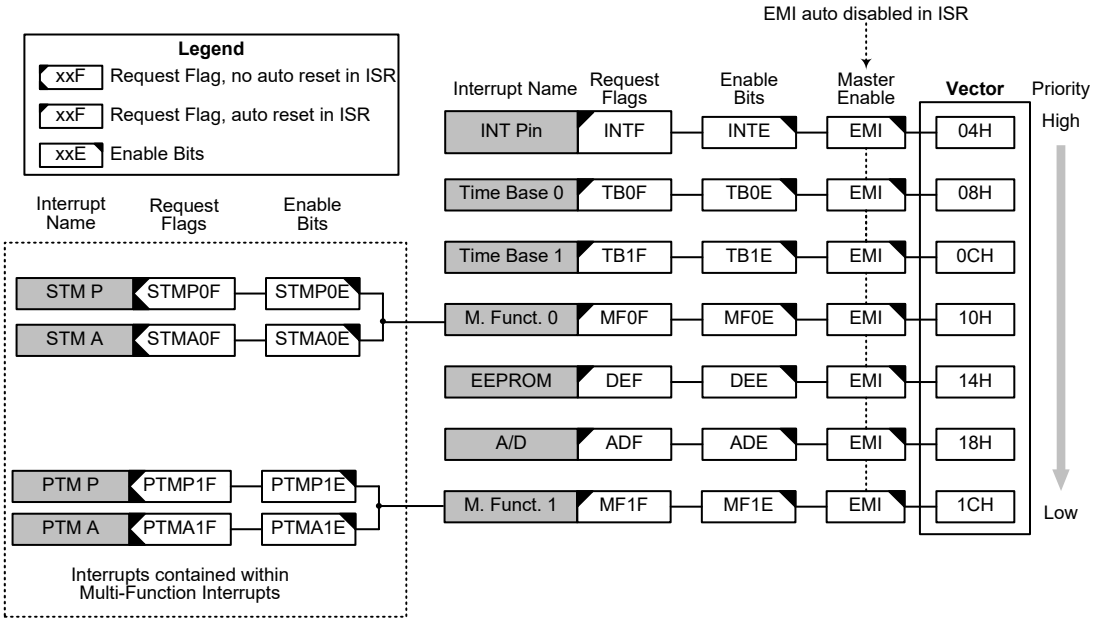
中断操作

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

当中断发生时，下条指令的地址将被压入堆栈。相应的中断向量地址加载至 PC 中。系统将从此向量取下条指令。中断向量处通常为跳转指令，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。一些中断源有自己的向量，但是有些中断却共用多功能中断向量。一旦中断子程序被响应，系统将自动清除 EMI 位，所有其它的中断将被屏蔽，这个方式可以防止任何进一步的中断嵌套。其它中断请求可能发生在此期间，虽然中断不会立即响应，但是中断请求标志位会被记录。

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



中断结构

外部中断

通过 INT 引脚上的信号变化可控制外部中断。当触发沿选择位设置好触发类型，INT 引脚的状态发生变化，外部中断请求标 INTF 被置位时外部中断请求产生。若要跳转到相应中断向量地址，总中断控制位 EMI 和相应中断使能位 INTE 需先被置位。此外，必须使用 INTEG 寄存器使能外部中断功能并选择触发沿类型。外部中断引脚和普通 I/O 口共用，可通过引脚共用功能选择寄存器设置，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相应的控制寄存器，将该引脚设置为输入口。当中断使能，堆栈未满并且外部中断脚状态改变，将调用外部中断向量子程序。当响应外部中断服务子程序时，中断请求标志位 INTF 会自动复位且 EMI 位会被清零以除能其它中断。注意，即使此引脚被用作外部中断输入，其上拉电阻仍保持有效。寄存器 INTEG 被用来选择有效的边沿类型，来触发外部中断。可以选择上升沿还是下降沿或双沿触发都产生外部中断。注意 INTEG 也可以用来除能外部中断功能。

多功能中断

此单片机中有 2 个多功能中断，与其它中断不同，它没有独立源，但由其它现有的中断源构成，即 TM 中断。

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

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

A/D 转换器中断

A/D 转换器中断由 A/D 转换动作的结束来控制。当 A/D 转换器中断请求标志被置位，即 A/D 转换过程完成时，中断请求发生。当总中断使能位 EMI 和 A/D

中断使能位 ADE 被置位，允许程序跳转到各自的中断向量地址。当中断使能，堆栈未满且 A/D 转换动作结束时，将调用它们各自的中断向量子程序。当响应中断服务子程序时，相应的中断请求标志位 ADF 会自动清零。EMI 位也会被清零以除能其它中断。

时基中断

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

时基中断的目的是提供一个固定周期的中断信号，其时钟源 f_{TB} 来自内部时钟源 f_{TBC} 或 $f_{SYS}/4$。f_{TB} 输入时钟首先经过分频器，分频率由程序设置 TBC 寄存器相关位获取合适的分频值以提供更长的时基中断周期。相应的控制时基中断周期的时钟源可通过 TBC 寄存器的一个位选择。

- **TBC 寄存器**

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

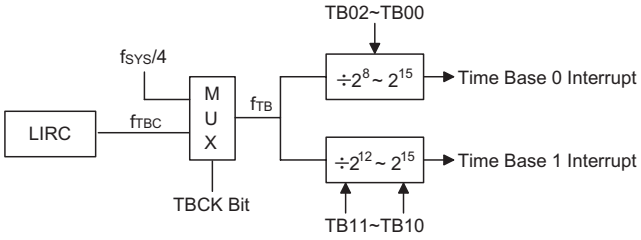
Bit 7 **TBON**：TB0 和 TB1 控制位
0：除能
1：使能

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

Bit 5~4 **TB11~TB10**：选择时基 1 溢出周期位
00：$2^{12}/f_{TB}$
01：$2^{13}/f_{TB}$
10：$2^{14}/f_{TB}$
11：$2^{15}/f_{TB}$

Bit 3 未使用，读为“0”

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



时基中断

EEPROM 中断

当写周期结束，EEPROM 中断请求标志 DEF 被置位，EEPROM 中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI 和 EEPROM 中断使能位 DEE 需先被置位。当中断使能，堆栈未满且 EEPROM 写周期结束时，可跳转至相应的 EEPROM 中断向量中执行。当 EEPROM 中断响应，EMI 将被自动清零以除能其它中断，EEPROM 中断请求标志 DEF 也可自动清除。

TM 中断

标准型和周期型 TM 各有两个中断，都属于多功能中断。每种类型 TM 有两个中断请求标志位 xTMPnF、xTMAnF 及两个使能位 xTMPnE、xTMAnE。当 TM 比较器 P 或 A 匹配情况发生时，相应 TM 中断请求标志被置位，TM 中断请求产生。

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

中断唤醒功能

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

编程注意事项

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

多功能中断中所含中断相应程序执行时，多功能中断请求标志 MF0F~MF1F 可以自动清零，但各自的请求标志需在应用程序中手动清除。

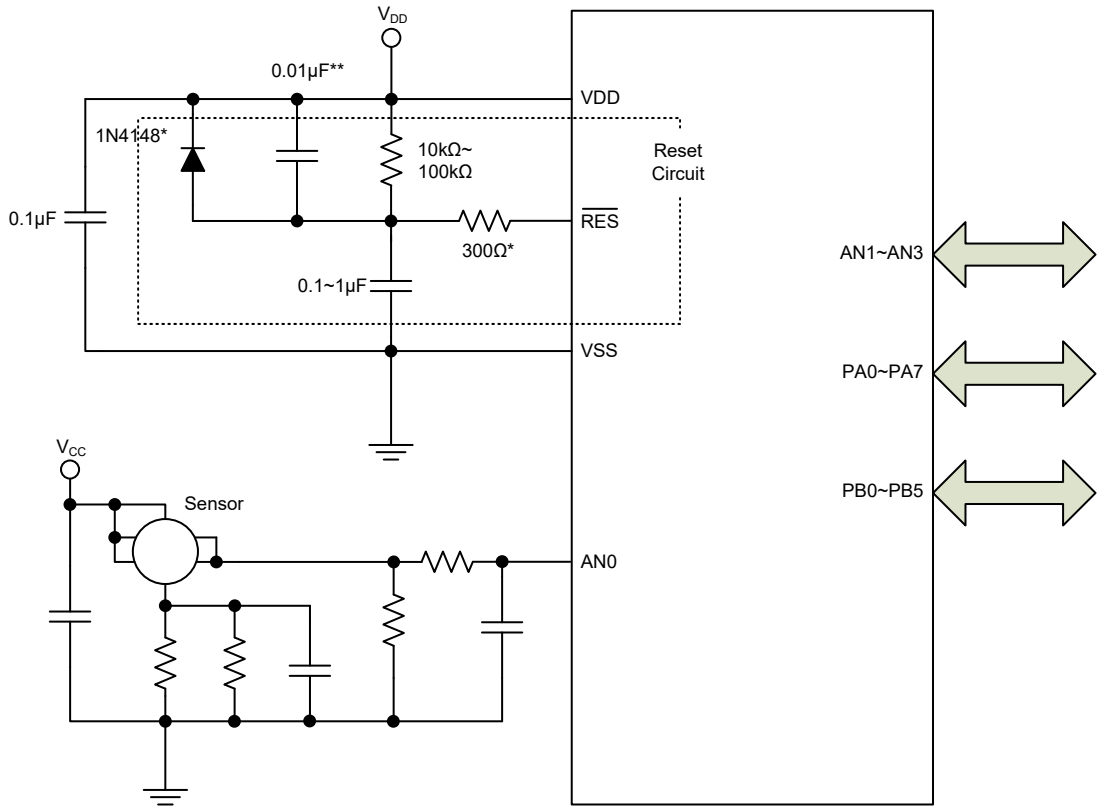
建议在中断服务子程序中不要使用“CALL 子程序”指令。中断通常发生在不可预料的情况或是需要立刻执行的某些应用。假如只剩下一层堆栈且没有控制好中断，当“CALL 子程序”在中断服务子程序中执行时，将破坏原来的控制序列。

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

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

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

应用电路



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。

指令集

简介

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

为了更加容易理解各种各样的指令码，接下来按功能分组介绍它们。

指令周期

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

数据的传送

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

算术运算

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

逻辑和移位运算

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

分支和控制转换

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

位运算

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

查表运算

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

其它运算

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

指令集概要

下表中说明了按功能分类的指令集，用户可以将该表作为基本的指令参考。

惯例

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

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

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

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

指令定义

ADC A, [m] Add Data Memory to ACC with Carry
指令说明 将指定的数据存储器、累加器内容以及进位标志相加，结果存放到累加器。
功能表示 ACC ← 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第 i 位内容清零。
功能表示	[m].i ← 0
影响标志位	无

CLR WDT	Clear Watchdog Timer
指令说明	WDT 计数器、暂停标志位 PDF 和看门狗溢出标志位 TO 清零。
功能表示	WDT cleared TO & PDF ← 0
影响标志位	TO、PDF

CPL [m] Complement Data Memory
指令说明 将指定数据存储器中的每一位取逻辑反，
相当于从 1 变 0 或 0 变 1。
功能表示 [m] ← $[\overline{m}]$
影响标志位 Z

CPLA [m] Complement Data Memory with result in ACC
指令说明 将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或 0 变 1，而结果被储存回累加器且数据存储器中的内容不变。
功能表示 ACC←$[\overline{m}]$
影响标志位 Z

DAA [m] Decimal-Adjust ACC for addition with result in Data Memory
指令说明 将累加器中的内容转换为 BCD (二进制转成十进制) 码。如果低四位的值大于“9”或 AC=1，那么 BCD 调整就执行对原值加“6”，否则原值保持不变；如果高四位的值大于“9”或 C=1，那么 BCD 调整就执行对原值加“6”。BCD 转换实质上是根据累加器和标志位执行 00H，06H，60H 或 66H 的加法运算，结果存放到数据存储器。只有进位标志位 C 受影响，用来指示原始 BCD 的和是否大于 100，并可以进行双精度十进制数的加法运算。
功能表示 [m] ← ACC + 00H 或
[m] ← ACC + 06H 或
[m] ← ACC + 60H 或
[m] ← ACC + 66H
影响标志位 C

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

DECA [m] Decrement Data Memory with result in ACC
指令说明 将指定数据存储器的内容减 1，把结果存放回累加器并保持指定数据存储器的内容不变。
功能表示 ACC ← [m] – 1
影响标志位 Z

HALT Enter power down mode
指令说明 此指令终止程序执行并关掉系统时钟，RAM 和寄存器的内容保持原状态，WDT 计数器和分频器被清“0”，暂停标志位 PDF 被置位 1，WDT 溢出标志位 TO 被清 0。
功能表示 TO ← 0
PDF ← 1
影响标志位 TO、PDF

INC [m] Increment Data Memory
指令说明 将指定数据存储器的内容加 1。
功能表示 [m] ← [m] + 1
影响标志位 Z

INCA [m] Increment Data Memory with result in ACC
指令说明 将指定数据存储器的内容加 1，结果存放回累加器并保持指定的数据存储器内容不变。
功能表示 ACC ← [m] + 1
影响标志位 Z

JMP addr Jump unconditionally
指令说明 程序计数器的内容无条件地由被指定的地址取代，程序由新的地址继续执行。当新的地址被加载时，必须插入一个空指令周期，所以此指令为 2 个周期的指令。
功能表示 Program Counter ← addr
影响标志位 无

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

MOV A, x Move immediate data to ACC
指令说明 将 8 位立即数载入累加器。
功能表示 ACC ← x
影响标志位 无

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

NOP	No operation
指令说明	空操作，接下来顺序执行下一条指令。
功能表示	无操作
影响标志位	无

OR A, [m]	Logical OR Data Memory to ACC
指令说明	将累加器中的数据和指定的数据存储器内容逻辑或， 结果存放到累加器。
功能表示	ACC ← ACC“OR”[m]
影响标志位	Z

OR A, x	Logical OR immediate data to ACC
指令说明	将累加器中的数据和立即数逻辑或，结果存放到累加器。
功能表示	ACC ← ACC“OR”x
影响标志位	Z

ORM A, [m]	Logical OR ACC to Data Memory
指令说明	将存在指定数据存储器中的数据和累加器逻辑或， 结果放到数据存储器。
功能表示	[m] ← ACC“OR”[m]
影响标志位	Z

RET	Return from subroutine
指令说明	将堆栈寄存器中的程序计数器值恢复， 程序由取回的地址继续执行。
功能表示	Program Counter←Stack
影响标志位	无

RET A, x	Return from subroutine and load immediate data to ACC
指令说明	将堆栈寄存器中的程序计数器值恢复且累加器载入指定的 立即数，程序由取回的地址继续执行。
功能表示	Program Counter ← Stack ACC←x
影响标志位	无

RETI	Return from interrupt
指令说明	将堆栈寄存器中的程序计数器值恢复且中断功能通过设置 EMI 位重新使能。EMI 是控制中断使能的主控制位。如果在执行 RETI 指令之前还有中断未被响应，则这个中断将在返回主程序之前被响应。
功能表示	Program Counter ←Stack EMI ← 1
影响标志位	无

RL [m]	Rotate Data Memory left
指令说明	将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位。
功能表示	[m].(i+1) ← [m].i (i=0~6) [m].0 ← [m].7
影响标志位	无

RLA [m]	Rotate Data Memory left with result in ACC
指令说明	将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位，结果送到累加器，而指定数据存储器的内容保持不变。
功能表示	ACC.(i+1) ← [m].i (i=0~6) ACC.0 ←[m].7
影响标志位	无

RLC [m]	Rotate Data Memory Left through Carry
指令说明	将指定数据存储器的内容连同进位标志左移 1 位， 第 7 位取代进位标志且原本的进位标志移到第 0 位。
功能表示	[m].(i+1) ← [m].i (i=0~6) [m].0 ← C C ← [m].7
影响标志位	C

RLC A [m]	Rotate Data Memory left through Carry with result in ACC
指令说明	将指定数据存储器的内容连同进位标志左移 1 位，第 7 位取代进位标志且原本的进位标志移到第 0 位，移位结果送回累加器，但是指定数据寄存器的内容保持不变。
功能表示	ACC.(i+1) ← [m].i (i=0~6) ACC.0 ← C C ← [m].7
影响标志位	C

RR [m] Rotate Data Memory right

指令说明 将指定数据存储器的内容循环右移 1 位且第 0 位移到第 7 位。

功能表示 [m].i ← [m].(i+1) (i=0~6)
[m].7 ← [m].0

影响标志位 无

RRA [m] Rotate Data Memory right with result in ACC

指令说明 将指定数据存储器的内容循环右移 1 位，第 0 位移到第 7 位，移位结果存放到累加器，而指定数据存储器的内容保持不变。

功能表示 ACC.i ← [m].(i+1) (i=0~6)
ACC.7 ← [m].0

影响标志位 无

RRC [m] Rotate Data Memory right through Carry

指令说明 将指定数据存储器的内容连同进位标志右移 1 位，第 0 位取代进位标志且原本的进位标志移到第 7 位。

功能表示 [m].i ← [m].(i+1) (i=0~6)
[m].7← C
C ← [m].0

影响标志位 C

RRCA [m] Rotate Data Memory right through Carry with result in ACC

指令说明 将指定数据存储器的内容连同进位标志右移 1 位，第 0 位取代进位标志且原本的进位标志移到第 7 位，移位结果送回累加器，但是指定数据寄存器的内容保持不变。

功能表示 ACC.i ← [m].(i+1) (i=0~6)
ACC.7 ← C
C ← [m].0

影响标志位 C

SBC A, [m] Subtract Data Memory from ACC with Carry

指令说明 将累加器减去指定数据存储器的内容以及进位标志的反，结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。

功能表示 ACC ← ACC – [m] – $\overline{C}$

影响标志位 OV、Z、AC、C

SBCM A, [m] Subtract Data Memory from ACC with Carry and result in Data Memory
指令说明 将累加器减去指定数据存储器的内容以及进位标志的反，结果存放到数据存储器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示 $[m] \leftarrow ACC - [m] - \overline{C}$
影响标志位 OV、Z、AC、C

SDZ [m] Skip if Decrement Data Memory is 0
指令说明 将指定的数据存储器的内容减 1，判断是否为 0，若为 0 则跳过下一条指令，由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示 [m] ← [m] – 1，如果 [m]=0 跳过下一条指令执行
影响标志位 无

SDZA [m] Skip if decrement Data Memory is zero with result in ACC
指令说明 将指定数据存储器内容减 1，判断是否为 0，如果为 0 则跳过下一条指令，此结果将存放到累加器，但指定数据存储器内容不变。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示 ACC ← [m] – 1，如果 ACC=0 跳过下一条指令执行
影响标志位 无

SET [m] Set Data Memory
指令说明 将指定数据存储器的每一位设置为 1。
功能表示 [m] ← FFH
影响标志位 无

SET [m].i Set bit of Data Memory
指令说明 将指定数据存储器的第 i 位置位为 1。
功能表示 [m].i ← 1
影响标志位 无

SIZ [m] Skip if increment Data Memory is 0

指令说明 将指定的数据存储器的内容加 1，判断是否为 0，若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。

功能表示 [m] ←[m] + 1，如果 [m]=0 跳过下一条指令执行

影响标志位 无

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

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

功能表示 ACC ←[m] + 1，如果 ACC=0 跳过下一条指令执行

影响标志位 无

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

指令说明 判断指定数据存储器的第 i 位，若不为 0，则程序跳过下一条指令执行。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果为 0，则程序继续执行下一条指令。

功能表示 如果 [m].i≠0，跳过下一条指令执行

影响标志位 无

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

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

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

SWAP [m] Swap nibbles of Data Memory

指令说明 将指定数据存储器的低 4 位和高 4 位互相交换。

功能表示 [m].3~[m].0 ↔ [m].7~[m].4

影响标志位 无

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

指令说明 将指定数据存储器的低 4 位与高 4 位互相交换，再将结果存放到累加器且指定数据寄存器的数据保持不变。

功能表示 ACC.3~ACC.0 ← [m].7~[m].4

ACC.7~ACC.4 ← [m].3~[m].0

影响标志位 无

SZ [m] Skip if Data Memory is 0

指令说明 指定数据存储器的内容会先被读出，后又被重新写入指定数据存储器内。判断指定数据存储器的内容是否为 0，若为 0，则程序跳过下一条指令执行。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。

功能表示 如果 [m]=0，跳过下一条指令执行

影响标志位 无

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

指令说明 将指定数据存储器内容复制到累加器，并判断指定数据存储器的内容是否为 0，若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。

功能表示 ACC ←[m]，如果 [m]=0，跳过下一条指令执行

影响标志位 无

SZ [m].i	Skip if bit i of Data Memory is 0
指令说明	判断指定数据存储器的第 i 位是否为 0，若为 0，则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	如果 [m].i=0，跳过下一条指令执行
影响标志位	无
TABRD [m]	Read table (specific page or current page) to TBLH and Data Memory
指令说明	将表格指针 (TBHP 和 TBLP，若无 TBHP 则仅 TBLP) 所指的程序代码低字节移至指定数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
TABRDL [m]	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

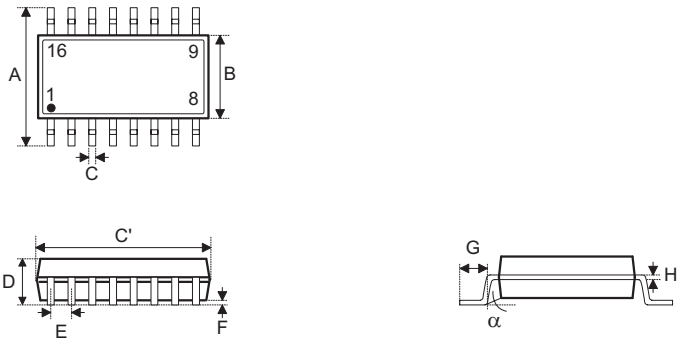
封装信息

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°

X-ON Electronics

Largest Supplier of Electrical and Electronic Components

Click to view similar products for 32-bit Microcontrollers - MCU ***category:***

Click to view products by Holtek ***manufacturer:***

Other Similar products are found below :

MCF51AC256AVFUE MCF51AC256BCFUE MCF51AC256BVFUE MB91F464AAPMC-GSE2 R5S726B0D216FP#V0 MB91F248PFV-GE1 MB91243PFV-GS-136E1 SAK-TC1782F-320F180HR BA TC364DP64F300WAAKXUMA1 R5F566NNDDFP#30 R5F566NNDDFC#30 R5F566NNDDBD#20 MC96F8216ADBN A96G181HDN A96G140KNN A96G174FDN A31G213CL2N A96G148KNN A96G174AEN AC33M3064TLBN-01 V3s T3 A40i-H V526 A83T R11 V851s A133 F1C100S T3L T507 A33 A63 T113-i H616 V853 V533 R16-J V536-H A64-H V831 V3LP F133-A R128-S2 D1-H ADUCM360BCPZ128-TR AT32F435VMT7 AT32F437ZMT7 AT32F421G8U7 AT32F421K8T7