A/D 型 Flash 单片机

HT66F0182

版本 : V1.10　日期 : 2017-08-29

www.holtek.com

目 录

特性 5
CPU 特性 5
周边特性 5
概述 6
方框图 6
引脚图 7
引脚描述 8
极限参数 9
直流电气特性 10
交流电气特性 11
A/D 转换器电气特性 12
LVR 电气特性 12
Bandgap 参考电压电气特性 – V_{BG} 13
上电复位电气特性 13
系统结构 14
时序与流水线结构 14
程序计数器 15
堆栈 15
算术逻辑单元 – ALU 16
Flash 程序存储器 17
结构 17
特殊向量 17
查表 17
查表范例 18
在线烧录 – ICP 18
片上调试 – OCDS 19
RAM 数据存储器 20
结构 20
通用数据存储器 20
特殊功能数据存储器 20
特殊功能寄存器描述 22
间接寻址寄存器 – IAR0，IAR1 22
存储器指针 – MP0，MP1 22
累加器 – ACC 23
程序计数器低字节寄存器 – PCL 23
查表寄存器 – TBLP，TBHP，TBLH 23
状态寄存器 – STATUS 23

振荡器 25
振荡器概述 25
系统时钟配置 25
内部 RC 振荡器 – HIRC 26
内部 32kHz 振荡器 – LIRC 26
辅助振荡器 26
工作模式与系统时钟 27
系统时钟 27
系统工作模式 28
控制寄存器 29
工作模式切换 30
待机电流注意事项 34
唤醒 34
看门狗定时器 35
看门狗定时器时钟源 35
看门狗定时器控制寄存器 35
看门狗定时器操作 36
复位和初始化 37
复位功能 37
复位初始条件 39
输入 / 输出端口 42
上拉电阻 42
PA 口唤醒 43
输入 / 输出端口控制寄存器 44
输入 / 输出引脚结构 45
编程注意事项 45
定时器模块 – TM 46
简介 46
TM 操作 46
TM 时钟源 46
TM 中断 46
TM 外部引脚 47
TM 输入 / 输出引脚控制寄存器 47
编程注意事项 49
标准型 TM – STM 50
标准型 TM 操作 50
标准型 TM 寄存器介绍 50
标准型 TM 工作模式 54
周期型 TM – PTM 63
周期型 TM 操作 63
周期型 TM 寄存器介绍 63
周期型 TM 工作模式 67

A/D 转换器 – ADC 76
A/D 转换器简介 76
A/D 转换器寄存器介绍 76
A/D 转换器操作 80
A/D 转换器参考电压 80
A/D 转换器输入信号 81
转换率和时序图 81
A/D 转换步骤 82
编程注意事项 82
A/D 转换功能 83
A/D 转换程序范例 84
中断 86
中断寄存器 86
中断操作 90
外部中断 91
多功能中断 91
A/D 转换器中断 91
时基中断 92
TM 中断 93
中断唤醒功能 93
编程注意事项 93
应用电路 94
指令集 95
简介 95
指令周期 95
数据的传送 95
算术运算 95
逻辑和移位运算 95
分支和控制转换 96
位运算 96
查表运算 96
其它运算 96
指令集概要 97
惯例 97
指令定义 100
封装信息 112
16-pin NSOP (150mil) 外形尺寸 113
20-pin NSOP (150mil) 外形尺寸 114

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 RC – HIRC
  - ♦ 内部 32kHz RC – LIRC
- 多种工作模式：正常、低速、空闲和休眠
- 高度集成内部 8MHz 振荡器，无需外部元器件
- 所有指令都可在 1 个指令周期内完成
- 查表指令
- 63 条功能强大的指令
- 6 层堆栈
- 位操作指令

周边特性

- Flash 程序存储器：4K×16
- RAM 数据存储器：128×8
- 看门狗定时器功能
- 18 个双向输入 / 输出口
- 2 个引脚与外部中断口共用
- 多个定时器模块用于时间测量、捕捉输入、比较匹配输出、PWM 输出及单脉冲输出
- 双时基功能用以产生固定的中断信号
- 8 个外部通道的 12-bit A/D 转换器
- 低电压复位功能
- 封装类型：16-pin/20-pin NSOP

概述

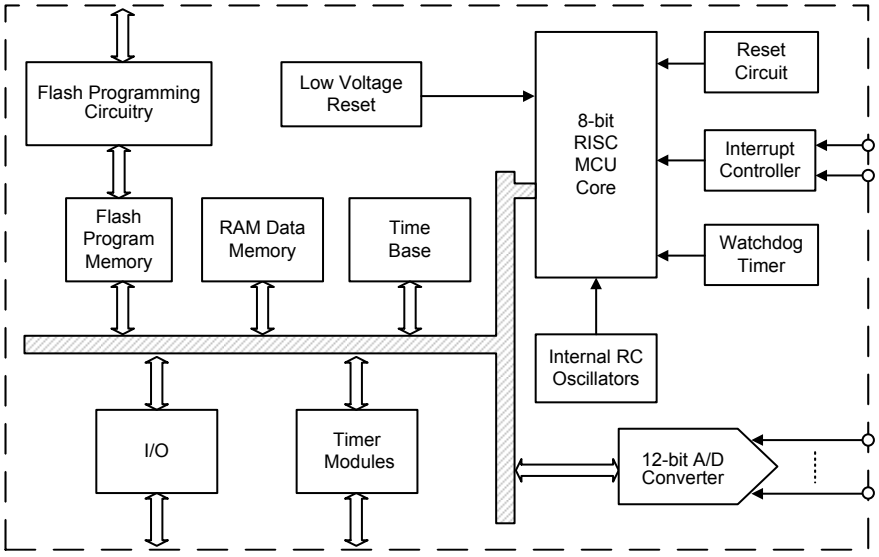
该单片机是一款 8 位具有高性能精简指令集的 Flash 型单片机。该单片机还具有一系列功能和特性，其 Flash 存储器可多次编程的特性给用户提供了极大的方便。存储器方面还包含了一个 RAM 数据存储器。

在模拟特性方面，该单片机包含了一个多通道的 12-bit A/D 转换器功能。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、捕捉输入、比较匹配输出及 PWM 产生功能。内部看门狗定时器、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。

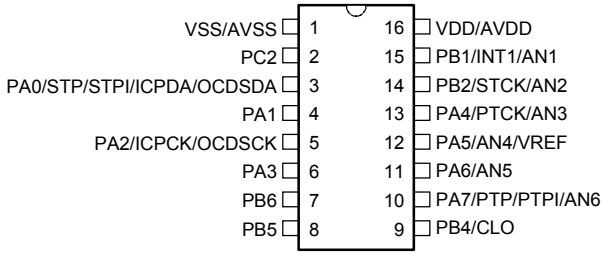
此单片机内建了高速和低速振荡器功能选项，可供不同的应用使用且无需外部元器件。其利用不同的时钟源在不同工作模式之间动态切换的能力，为用户提供了一个优化单片机操作和减少功耗的手段。

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

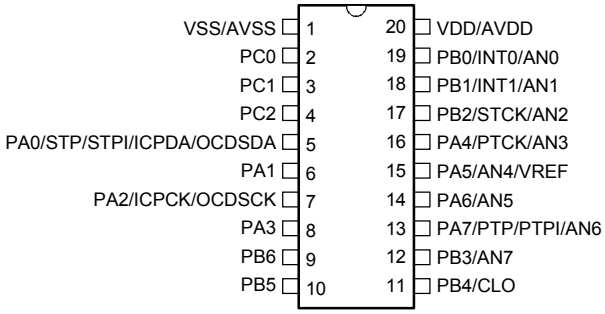
方框图



引脚图



HT66F0182/HT66V0182
16 NSOP-A



HT66F0182/HT66V0182
20 NSOP-A

注：1. 若引脚共用功能同时有多种输出则在“/”右侧的引脚名具有更高的优先级。
2. VDD/AVDD 表明 VDD 与 AVDD 相连接。VSS/AVSS 表明 VSS 与 AVSS 相连接。
3. OCDSDA 与 OCDSCK 引脚为 OCDS 专用引脚，仅用于 HT66V0182。HT66V0182 为 HT66F0182 单片机的 OCDS EV 芯片。

引脚描述

除了电源引脚外，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，定时器模块引脚等。每个引脚的功能如下表所述，而引脚配置的详细内容见规格书其它章节。

引脚名称	功能	OPT	I/T	O/T	描述
PA0/STP/STPI/ICPDA/OCDSDA	PA0	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉和唤醒。
	STP	TMPC	—	CMOS	STM 输出
	STPI	TMPC	ST	—	STM 比较输入
	ICPDA	—	ST	CMOS	ICP 地址 / 数据
	OCDSDA	—	ST	CMOS	OCDS 地址 / 数据，仅用于 EV 芯片
PA1	PA1	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉和唤醒。
PA2/ICPCK/OCDSCK	PA2	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉和唤醒。
	ICPCK	—	ST	—	ICP 时钟
	OCDSCK	—	ST	—	OCDS 时钟，仅用于 EV 芯片
PA3	PA3	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉和唤醒。
PA4/PTCK/AN3	PA4	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉和唤醒。
	PTCK	PTMC0	ST	—	PTM 时钟输入
	AN3	ACERL	AN	—	A/D 转换器输入通道 3
PA5/AN4/VREF	PA5	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉和唤醒。
	AN4	ACERL	AN	—	A/D 转换器输入通道 4
	VREF	ADCR1	AN	—	A/D 转换器参考电压输入引脚
PA6/AN5	PA6	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉和唤醒。
	AN5	ACERL	AN	—	A/D 转换器输入通道 5
PA7/PTP/PTPI/AN6	PA7	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉和唤醒。
	PTP	TMPC	—	CMOS	PTM 输出
	PTPI	TMPC	ST	—	PTM 捕捉输入
	AN6	ACERL	AN	—	A/D 转换器输入通道 6
PB0/INT0/AN0	PB0	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉。
	INT0	INTC0 INTEG	ST	—	外部中断 0
	AN0	ACERL	AN	—	A/D 转换器输入通道 0
PB1/INT1/AN1	PB1	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉。
	INT1	INTC2 INTEG	ST	—	外部中断 1
	AN1	ACERL	AN	—	A/D 转换器输入通道 1

引脚名称	功能	OPT	I/T	O/T	描述
PB2/STCK/AN2	PB2	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉。
	STCK	STMC0	ST	—	STM 时钟输入
	AN2	ACERL	AN	—	A/D 转换器输入通道 2
PB3/AN7	PB3	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉。
	AN7	ACERL	AN	—	A/D 转换器输入通道 7
PB4/CLO	PB4	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉。
	CLO	TMPC	ST	CMOS	系统时钟输出
PB5	PB5	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉。
PB6	PB6	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉。
PC0 ~ PC2	PC0 ~ PC2	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉。
VDD*	VDD	—	PWR	—	电源供电
AVDD*	AVDD	—	PWR	—	A/D 转换器电源供电
VSS**	VSS	—	PWR	—	接地脚
AVSS**	AVSS	—	PWR	—	A/D 转换器接地脚

注：I/T：输入类型；　　O/T：输出类型；
OPT：通过寄存器选项来配置；
PWR：电源；　　ST：施密特触发输入；
CMOS：CMOS 输出；　　AN：模拟信号。
*：VDD 为单片机电源供电而 AVDD 为 A/D 转换器电源供电。AVDD 引脚与 VDD 内部相连。
**：VSS 为单片机接地引脚而 AVSS 为 A/D 转换器接地引脚。AVSS 引脚与 VSS 内部相连。

极限参数

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

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

直流电气特性

Ta=25℃

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DD}	工作电压 (HIRC)	—	$f_{SYS}=f_{HIRC}$=8MHz	2.2	—	5.5	V
I_{DD}	工作电流 (HIRC)	3V	无负载，所有外设关闭，$f_{SYS}=f_{HIRC}$ =8MHz	—	0.8	1.2	mA
		5V		—	1.6	2.4	mA
	工作电流 (LIRC)	3V	无负载，所有外设关闭，$f_{SYS}=f_{LIRC}$ =32kHz	—	10	20	μA
		5V		—	30	50	μA
I_{STB}	待机电流 (SLEEP0 模式)	3V	无负载，所有外设关闭，WDT off	—	0.2	0.8	μA
		5V		—	0.5	1	μA
	待机电流 (SLEEP1 模式)	3V	无负载，所有外设关闭，WDT on	—	1.5	3	μA
		5V		—	3	5	μA
	待机电流 (IDLE0 模式)	3V	无负载，所有外设关闭，f_{SUB} on	—	3	5	μA
		5V		—	5	10	μA
	待机电流 (IDLE1 模式，HIRC)	3V	无负载，所有外设关闭，f_{SUB} on，$f_{SYS}=f_{HIRC}$=8MHz	—	360	500	μA
		5V		—	600	800	μA
I_{OL}	I/O 口灌电流	3V	$V_{OL}=0.1V_{DD}$	16	32	—	mA
		5V	$V_{OL}=0.1V_{DD}$	32	64	—	mA
I_{OH}	I/O 口源电流	3V	$V_{OH}=0.9V_{DD}$	-3.75	-7.5	—	mA
		5V	$V_{OH}=0.9V_{DD}$	-7.5	-15	—	mA
R_{PH}	I/O 口上拉电阻	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ
V_{IL}	I/O 口低电平输入电压	5V	—	0	—	1.5	V
		—	—	0	—	$0.2V_{DD}$	V
V_{IH}	I/O 口高电平输入电压	5V	—	3.5	—	5	V
		—	—	$0.8V_{DD}$	—	V_{DD}	V

交流电气特性

Ta=25℃

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
f_{SYS}	系统时钟 (HIRC)	2.2V~ 5.5V	$f_{SYS}=f_{HIRC}$=8MHz	—	8	—	MHz
f_{SYS}	系统时钟 (LIRC)	2.2V~ 5.5V	$f_{SYS}=f_{LIRC}$=32kHz	—	32	—	kHz
f_{HIRC}	高速内部 RC 振荡器 (HIRC)	3V/5V	Ta=25℃	-2%	8	+2%	MHz
		3V/5V	Ta=0℃ ~ 70℃	-5%	8	+5%	MHz
		2.2V~ 5.5V	Ta=0℃ ~ 70℃	-7%	8	+7%	MHz
		2.2V~ 5.5V	Ta= -40℃ ~ 85℃	-10%	8	+10%	MHz
f_{LIRC}	低速内部 RC 振荡器 (LIRC)	3V	Ta=25℃	-10%	32	+10%	kHz
		3V ± 0.3V	Ta=-40℃ ~ 85℃	-40%	32	+40%	kHz
		2.2V~ 5.5V	Ta=-40℃ ~ 85℃	-50%	32	+60%	kHz
		5V	Ta=25℃	-10%	32	+10%	kHz
		5V± 0.5V	Ta= -40℃ ~ 85℃	-40%	32	+40%	kHz
		2.2V~ 5.5V	Ta= -40℃ ~ 85℃	-50%	32	+60%	kHz
t_{START} (LIRC)	系统启动时间	5V	—	—	—	500	μs
t_{TCK}	STCK 与 PTCK 引脚输入最小脉宽	—	—	0.3	—	—	μs
t_{TPI}	STPI 与 PTPI 引脚输入最小脉宽	—	—	0.3	—	—	μs
t_{CPW}	TM 捕捉输入最小脉宽	—	—	2	—	—	t_{TMCLK}
t_{RSTD}	系统复位延迟时间 (POR 复位，LVR 硬件复位，LVR 软件复位，WDT 软件复位)	—	—	25	50	100	ms
	系统复位延迟时间 (WDT 溢出硬件冷复位)	—	—	8.3	16.7	33.3	ms
t_{SST}	系统启动时间 (从 HALT 唤醒，HALT 状态下 f_{SYS} off)	—	$f_{SYS}=f_{HIRC}$ ~ f_{HIRC} / 64	16	—	—	t_{HIRC}
		—	$f_{SYS}=f_{LIRC}$	2	—	—	t_{LIRC}
	系统启动时间 (低速模式 ↔ 正常模式)	—	f_{HIRC} off → on (HTO=1)	16	—	—	t_{HIRC}
	系统启动时间 (从 HALT 唤醒，HALT 状态下 f_{SYS} on)	—	$f_{SYS}=f_{HIRC}$ ~ f_{HIRC} / 64	2	—	—	t_{HIRC}
		—	$f_{SYS}=f_{LIRC}$	2	—	—	t_{LIRC}
	系统启动时间 (WDT 溢出硬件冷复位)	—	—	0	—	—	t_H

注：1. $t_{HIRC}=1/f_{HIRC}$；

2. 为保持内部 HIRC 振荡器频率的精确度，需在 VDD 和 VSS 之间连接一个 0.1μF 的去耦电容并尽可能靠近单片机。

A/D 转换器电气特性

Ta=25℃

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DD}	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	—	V_{REF}	V
V_{REF}	A/D 转换器参考电压	—	—	2	—	V_{DD}	V
DNL	非线性微分误差	—	V_{REF}=V_{DD}，t_{ADCK}=0.5μs Ta= -40°C~85°C	-3	—	+3	LSB
INL	非线性积分误差	—	V_{REF}=V_{DD}，t_{ADCK}=0.5μs Ta= -40°C~85°C	-4	—	+4	LSB
I_{ADC}	使能 A/D 转换器的额外电流	2.2V	无负载 (t_{ADCK}=0.5μs)	—	1.0	2.0	mA
		3V		—	1.0	2.0	mA
		5V		—	1.5	3.0	mA
t_{ADCK}	A/D 转换器时钟周期	—	—	0.5	—	10	μs
t_{ON2ST}	A/D 转换器 On-to-Start 时间	—	—	4	—	—	μs
t_{ADS}	A/D 转换器采样时间	—	—	—	4	—	t_{ADCK}
t_{ADC}	A/D 转换时间 (包含采样与保持时间)	—	—	—	16	—	t_{ADCK}

注：A/D 转换时间 (t_{ADC})=n(位 A/D 转换器) + 4(采样时间)，每个位的转换需要一个 A/D 转换器时钟 (t_{ADCK})。

LVR 电气特性

Ta=25℃

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{LVR}	低压复位电压	—	LVR 使能，2.1V	-5%	2.1	+5%	V
I_{LVRBG}	工作电流	5V	LVR 使能，V125EN=0	—	20	25	μA
			LVR 使能，V125EN=1	—	180	200	μA
t_{LVR}	低压复位最小脉宽	—	—	120	240	480	μs

Bandgap 参考电压电气特性 – V_{BG}

Ta=25℃

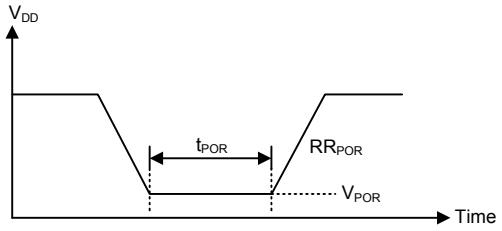
符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{BG}	Bandgap 参考电压	—	—	- 3%	1.25	+ 3%	V
t_{BGS}	V_{BG} 启动稳定时间	—	无负载	—	—	150	μs
I_{BG}	带缓冲器 V_{BG} 的额外电流	—	LVR 除能	—	—	220	μA

注：V_{BG} 电压可用于 A/D 转换器内部信号输入。

上电复位电气特性

Ta=25℃

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{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

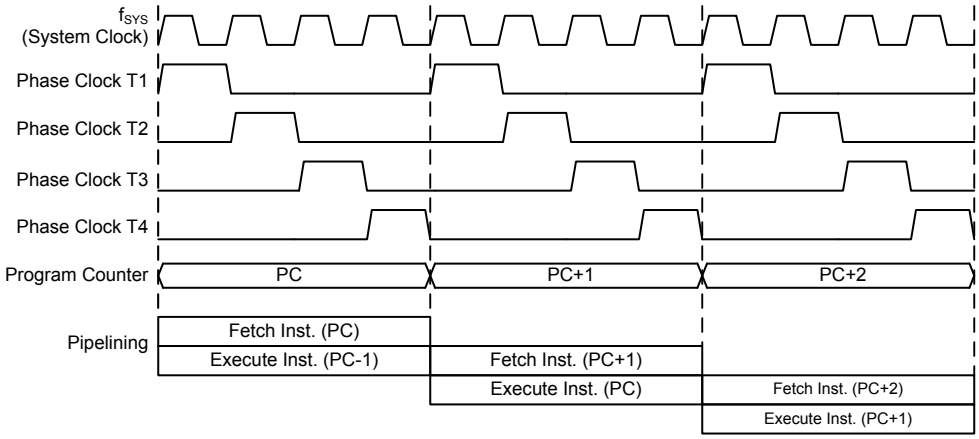


系统结构

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需要多一个指令周期外，大部分指令能在一个指令周期内完成。8-bit ALU 参与指令集中所有的运算，它可完成算术运算、逻辑运算、移位、递增、递减和分支等功能，而内部的数据路径则是以通过累加器和 ALU 的方式加以简化。有些寄存器在数据存储器中被实现，且可以直接或间接寻址。简单的寄存器寻址方式和结构特性，确保了在提供具有最大可靠度和灵活性的 I/O 和 A/D 控制系统时，仅需要少数的外部器件。这些使得此单片机适用于低成本和批量生产的控制应用。

时序与流水线结构

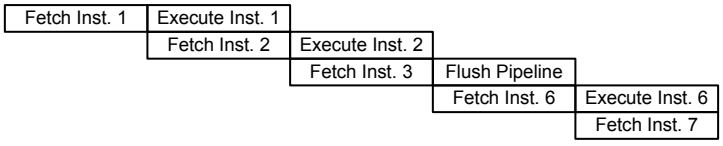
主系统时钟由 HIRC 或 LIRC 振荡器提供，它被细分为 T1~T4 四个内部产生的非重叠时序。在 T1 时间，程序计数器自动加一并取得一条新的指令。剩下的时间 T2~T4 完成译码和执行功能，因此，一个 T1~T4 时钟周期构成了一个指令周期。尽管指令的获取和执行发生在连续的指令周期中，但单片机流水线结构会保证指令在一个指令周期被有效执行。除非程序计数器的内容被改变，如子程序的调用和跳转，在这种情况下指令将需要多一个指令周期的时间去执行。



系统时序与流水线

如果指令牵涉到分支，例如跳转或调用等指令，则需要两个机器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，尤其是在应用对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
PC11~PC8	PCL7~PCL0

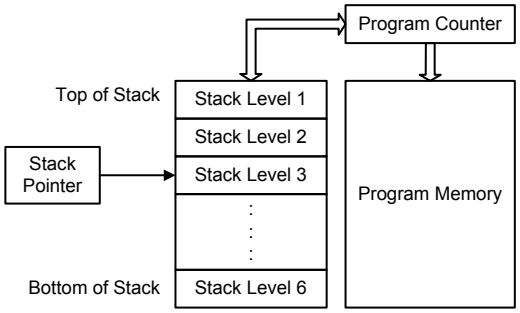
程序计数器的低字节，即程序计数器低字节寄存器 PCL，可以通过程序控制，且它是可读可写的寄存器。通过直接写入数据到这个寄存器，程序短跳转可直接执行，但由于只有低字节的操作是有效的，跳转被限制在存储器的当前页中，即 256 个存储器地址范围内。应注意的是当这样的一个程序跳转要执行时会插入一个空指令周期。PCL 的操作可能引起程序分支，因此需要额外的指令周期。

堆栈

堆栈是一个特殊的存储空间，只用来存储程序计数器中的内容。该单片机有 6 层堆栈。堆栈既不是数据部分也不是程序空间部分，而且它既不可读取也不可写入。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。当芯片复位后，堆栈指针将指向堆栈顶部。

如果堆栈已满，且有已使能的中断发生，中断请求标志会被记录下来，但中断响应将被禁止。当堆栈指针通过执行 RET 或 RETI 而使指针减少，中断服务将被响应。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。使用时应注意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相应地更新内容以显示这些改变，ALU 所提供的功能如下：

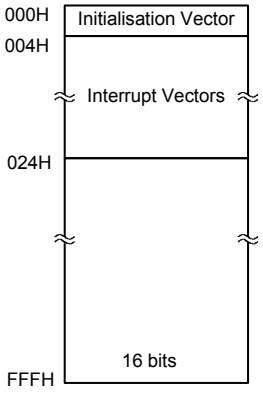
- 算术运算：ADD，ADDM，ADC，ADCM，SUB，SUBM，SBC，SBCM，DAA
- 逻辑运算：AND，OR，XOR，ANDM，ORM，XORM，CPL，CPLA
- 移位运算：RRA，RR，RRCA，RRC，RLA，RL，RLCA，RLC
- 递增和递减：INCA，INC，DECA，DEC
- 分支判断：JMP，SZ，SZA，SNZ，SIZ，SDZ，SIZA，SDZA，CALL，RET，RETI

Flash 程序存储器

程序存储器用来存放用户代码及储存程序。此单片机的程序存储器为 Flash 类型意味着可以多次重复烧录，方便用户使用同一芯片进行程序代码的修改。使用适当的单片机烧录工具，该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 4K×16。程序存储器用程序计数器来寻址，其中也包含数据、表格信息和中断入口。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

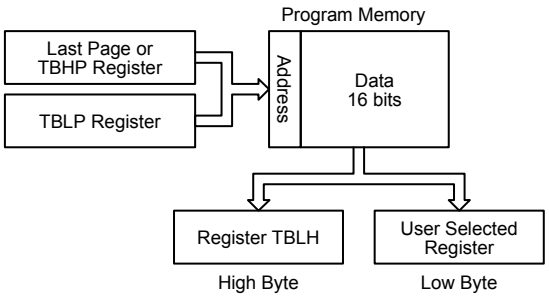
程序存储器内部某些地址保留用作诸如复位和中断入口等特殊用途。地址 000H 是芯片复位后的程序起始地址。在芯片复位之后，程序将跳到这个地址并开始执行。

查表

程序存储器中的任何地址都可以定义成一个表格，以便储存固定的数据。使用表格时，表格指针必须先行设定，其方式是将表格的地址放在表格指针寄存器 TBLP 和 TBHP 中。这两个寄存器定义表格总的地址。

在设置完表格指针后，表格数据可以使用“TABRD [m]”或“TABRDL [m]”指令分别从程序存储器查表读取。当这些指令执行时，程序存储器中表格数据低字节，将被传送到使用者在指令中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。所发送的高字节中未用到的位读为“0”。

下图为查表中寻址 / 数据流程：



查表范例

以下范例说明如何定义并从单片机中获取表格指针和表格数据。这个例子使用的是利用“ORG”指令储存在程序存储器中的一个原始数据表。ORG 指令的值“F00H”位于单片机中 4K 程序存储器内最后一页的起始地址。表格指针设置的初始值为“06H”，以确保从数据表格读取的第一笔数据位于程序存储器地址“F06H”，或是最后一页起始地址后的第六个地址。值得注意的是，若“TABRD [m]”指令被使用，则表格指针指向 TBHP 和 TBLP 寄存器所指定的起始地址，否则表格指针指向当前页的起始地址。在这个例子中，表格数据的高字节等于零，而当“TABRD [m]”指令被执行时，此值将会自动的被传送到 TBLH 寄存器。

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

表格读取程序范例

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

在线烧录 – ICP

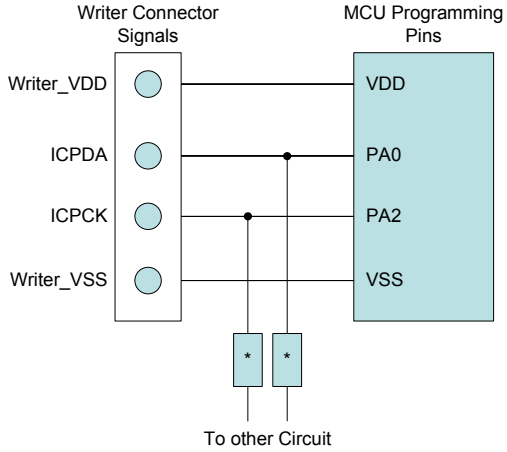
Flash 型程序存储器的提供使得用户可以方便简单地在同一芯片上进行程序的更新和修改。另外，Holtek 单片机提供 4 线接口的在线烧录方式。用户可将进行过烧录或未经过烧录的单片机芯片连同电路板一起制成，最后阶段进行程序的更新和程序的烧写，在无需去除或重新插入芯片的情况下方便地保持程序为最新版。

Holtek Flash 型单片机对应的烧录器引脚如下表所示：

Holtek 烧录器引脚	MCU 在线烧录引脚	引脚功能
ICPDA	PA0	烧写串行数据 / 地址
ICPCK	PA2	烧写时钟
VDD	VDD	电源
VSS	VSS	地

单片机内部程序存储器可以通过 4 线的接口在线进行烧录。其中一个单独的引脚用于数据串行下载或上传、一条用于时钟、另外两条用于提供电源。芯片在线烧录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，用户必须注意 ICPDA 用于烧写数据而 ICPCK 用于烧写时钟，确保这两个引脚没有被连接到其它输出。



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

片上调试 – OCDS

EV 芯片 HT66V0182 用于 HT66F0182 单片机的仿真。此 EV 芯片提供片上调试功能 (OCDS) 用于开发过程中 HT66F0182 单片机的调试。除了片上调试功能，EV 芯片和实际 MCU 在功能上几乎是兼容的。用户可将 OCDSDA 和 OCDSCK 引脚连接至 Holtek HT-IDE 开发工具，用 EV 芯片来仿真实际单片机芯片的性能。OCDSDA 引脚为 OCDS 数据 / 地址输入 / 输出脚，OCDSCK 引脚为 OCDS 时钟输入脚。当用户用 EV 芯片进行调试时，OCDSDA 和 OCDSCK 引脚上的其它共用功能对 EV 芯片无效。这两个 OCDS 引脚与 ICP 引脚共用，因此在线烧录时仍用作 Flash 存储器烧录引脚。关于 OCDS 功能的详细描述，参考“Holtek e-Link for 8-bit MCU OCDS User's Guide”文档。

Holtek e-Link 引脚	EV 芯片引脚	引脚描述
OCDSDA	OCDSDA	片上调试数据 / 地址输入 / 输出
OCDSCK	OCDSCK	片上调试时钟输入
VDD	VDD	电源
GND	VSS	地

RAM 数据存储器

数据存储器是内容可更改的 8 位 RAM 内部存储器，用来储存临时数据。

结构

数据存储器分为两部分，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。第二部分是通用数据存储器，该存储器内所有地址都可在程序的控制下进行读取和写入。数据存储器起始地址为 00H。

通用数据存储器

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

特殊功能数据存储器

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

00H	IAR0	40H	PC
01H	MP0	41H	PCC
02H	IAR1	42H	PCPU
03H	MP1	43H	
04H	Unused	44H	
05H	ACC	45H	
06H	PCL	46H	
07H	TBLP	47H	
08H	TBLH	48H	
09H	TBHP	49H	
0AH	STATUS	4AH	
0BH	SMOD	4BH	
0CH	Unused	4CH	
0DH	INTEG	4DH	
0EH	INTC0	4EH	
0FH	INTC1	4FH	
10H	INTC2	50H	
11H	MFI0	51H	
12H	MFI1	52H	
13H	Unused	53H	
14H	PA	54H	
15H	PAC	55H	
16H	PAPU	56H	
17H	PAWU	57H	
18H	Unused	58H	
19H	TMPC	59H	
1AH	WDTC	5AH	
1BH	TBC	5BH	
1CH	CTRL	5CH	
1DH	LVRC	5DH	
1EH	Unused	5EH	
1FH		5FH	
20H	ADRL	60H	
21H	ADRH	61H	Unused
22H	ADCR0	62H	
23H	ADCR1	63H	
24H	ACERL	64H	
25H	PB	65H	
26H	PBC	66H	
27H	PBPU	67H	
28H	Unused	68H	
29H		69H	
2AH		6AH	
2BH		6BH	
2CH		6CH	
2DH		6DH	
2EH		6EH	
2FH	STMC0	6FH	
30H	STMC1	70H	
31H	STMDL	71H	
32H	STMDH	72H	
33H	STMAL	73H	
34H	STMAH	74H	
35H	Unused	75H	
36H	PTMC0	76H	
37H	PTMC1	77H	
38H	PTMDL	78H	
39H	PTMDH	79H	
3AH	PTMAL	7AH	
3BH	PTMAH	7BH	
3CH	PTMRPL	7CH	
3DH	PTMRPH	7DH	
3EH	Unused	7EH	
3FH		7FH	

: Unused, read as 00H

特殊功能数据存储器结构

特殊功能寄存器描述

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寄存器和存储器指针做数据操作，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 或 MP1 所指定的存储器地址产生对应的读 / 写操作。它们总是成对出现，IAR0 和 MP0 可以访问 Bank 0，而 IAR1 和 MP1 可以访问任何 Bank。因为这些间接寻址寄存器不是实际存在的，直接读取将返回“00H”的结果，而直接写入此寄存器则不做任何操作。

存储器指针 – MP0，MP1

该单片机提供两个存储器指针，即 MP0 和 MP1。由于这些指针在数据存储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相关的存储器指针所指定的地址。MP0 和间接寻址寄存器 IAR0 用于访问 Bank 0，而 MP1 和 IAR1 根据 BP 寄存器可以访问所有的 Bank。直接寻址只能在 Bank 0 中使用，使用 MP1 和 IAR1 可间接访问所有的数据 Bank。

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

间接寻址程序范例

```
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□
```

需注意，在例中并没有确定 RAM 地址。

累加器 – ACC

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

程序计数器低字节寄存器 – PCL

为了提供额外的程序控制功能，程序计数器低字节设置在数据存储器的特殊功能区域内，程序员可对此寄存器进行操作，很容易的直接跳转到其它程序地址。直接给 PCL 寄存器赋值将导致程序直接跳转到程序存储器的某一地址，然而由于寄存器只有 8 位长度，因此只允许在本页的程序存储器范围内进行跳转，而当使用这种运算时，要注意会插入一个空指令周期。

查表寄存器 – TBLP，TBHP，TBLH

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

状态寄存器 – STATUS

该 8-bit 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 TO 和 PDF 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 标志位通常反映最新运算的状态。

- 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，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　未定义，读为“0”

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

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

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

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

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

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

振荡器

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

振荡器概述

振荡器除了作为系统时钟源，还作为看门狗定时器和时基中断的时钟源。高度集成的内部振荡器不需要任何外围器件。它们提供的高速和低速系统振荡器具有较宽的频率范围。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。

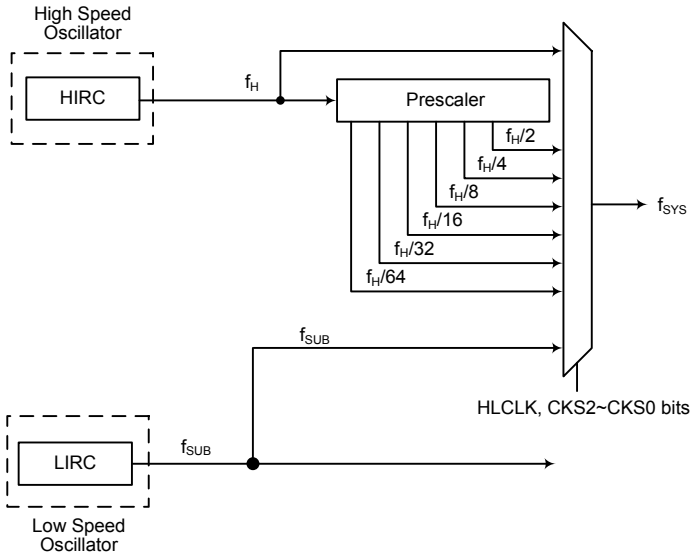
类型	名称	频率
内部高速 RC	HIRC	8MHz
内部低速 RC	LIRC	32kHz

振荡器类型

系统时钟配置

该单片机有两个系统振荡器，即一个高速内部 RC 振荡器和一个低速内部 RC 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 位和 CKS2~CKS0 位决定的，系统时钟可动态选择。

通过寄存器选择采用高速或低速振荡器作为实际时钟源。利用 SMOD 寄存器中的 HLCLK 位和 CKS2~CKS0 位段来确定低速或高速系统时钟的频率。需注意两个振荡器必须做出选择，即选择高速或选择低速系统振荡器。



系统时钟配置

内部 RC 振荡器 – HIRC

此内部高速 RC 振荡器是一个高度集成的系统振荡器，无需其它外部器件。内部 RC 振荡器有一个 8MHz 的频率。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡频率因电源电压、温度以及芯片制成工艺不同的影响减至最低程度。在供电电压为 3V/5V 且温度为 25℃时，固定振荡频率 8MHz 的容限为 2%. 需注意，若选中此内部系统时钟选项，则无需用到外部引脚，I/O 口引脚保持原功能。

内部 32kHz 振荡器 – LIRC

内部 32kHz 系统振荡器是一个低频振荡器。这是一个完全集成 RC 振荡器，它在 5V 电压下运行的典型频率值为 32kHz 且无需外部元件。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡器因电源电压、温度及芯片制成工艺不同的影响减至最低。在供电电压为 5V 且温度为 25℃时，固定振荡频率 32kHz 的容限为 10%.

辅助振荡器

低速振荡器除了提供系统时钟源外还用于为看门狗定时器和时基中断提供一个时钟源。

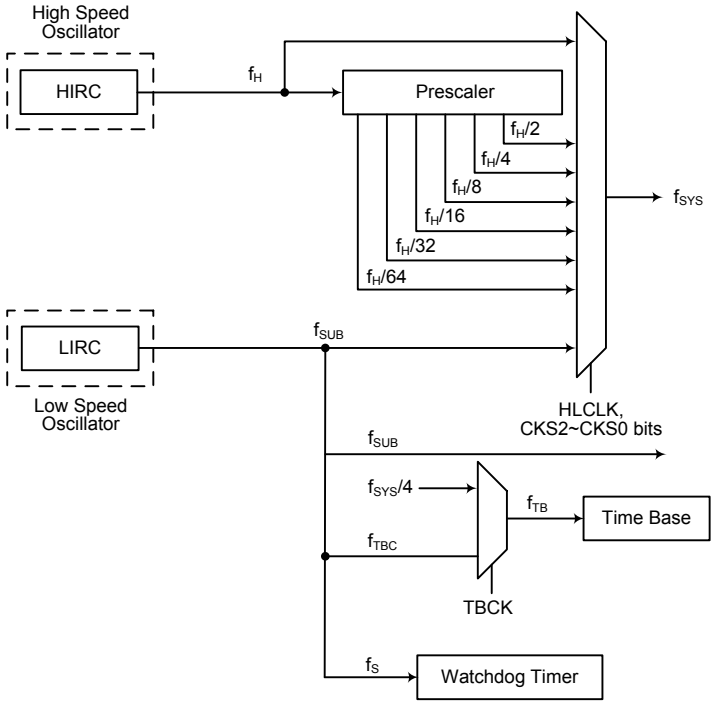
工作模式与系统时钟

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。Holtek 单片机了提供高、低速两种时钟源，它们之间可以动态切换，用户可通过优化单片机操作来获得最佳性能 / 功耗比。

系统时钟

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

主系统时钟可来自高频时钟源 f_H 或低频时钟源 f_{SUB}，通过 SMOD 寄存器中的 HLCLK 位和 CKS2~CKS0 位进行选择。高频系统时钟来自 HIRC 振荡器，低频系统时钟源来自 LIRC 振荡器。其它系统时钟还有高速系统振荡器的分频 $f_H/2$~$f_H/64$。



单片机时钟设置

注：当系统时钟源 f_{SYS} 由 f_H 切换到 f_{SUB}，高速振荡器将停止以节省耗电。因此没有 f_H~$f_H/64$ 的时钟可供外部电路使用。

系统工作模式

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

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

正常模式

顾名思义，这是主要的工作模式之一，单片机的所有功能均可在此模式中实现且系统时钟由一个高速振荡器提供。该模式下单片机正常工作的时钟源来自 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 HLCLK 位和 CKS2~CKS0 位进行选择。单片机使用高速振荡器分频作为系统时钟可减少工作电流。

低速模式

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

休眠模式 0

在 HALT 指令执行后且 SMOD 寄存器中的 IDLEN 位为低时，系统进入休眠模式。在休眠模式 0 中，CPU 停止运行，f_{SUB} 和 f_S 时钟也将停止运行，看门狗定时器功能除能。

休眠模式 1

在 HALT 指令执行后且 SMOD 寄存器中的 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，若看门狗定时器使能且其时钟源由寄存器选择为 f_{SUB}，则 f_{SUB} 和 f_S 时钟将继续运行。

空闲模式 0

执行 HALT 指令后且 SMOD 寄存器中的 IDLEN 位为高，CTRL 寄存器中的 FSYSON 位为低时，系统进入空闲模式 0。在空闲模式 0 中，CPU 停止，系统振荡器停止，但 f_{SUB} 时钟将开启以驱动一些外围功能如看门狗定时器和 TM。

空闲模式 1

执行 HALT 指令后且 SMOD 寄存器中的 IDLEN 位为高，CTRL 寄存器中的 FSYSON 位为高时，系统进入空闲模式 1。在空闲模式 1 中，CPU 停止，但系统振荡器将提供一个时钟源以驱动看门狗定时器、TM 等功能。在空闲模式 1 中，高速或低速振荡器都将继续运行。

控制寄存器

SMOD 与 CTRL 寄存器用于控制单片机的内部时钟。

- **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**：HLCLK 为“0”时系统时钟选择位
000：f_{SUB}(f_{LIRC})
001：f_{SUB}(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 可作为系统时钟源外，高速系统振荡器的分频比也可作为系统时钟源。

Bit 4 未定义，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唤醒后何时稳定下来。当从休眠模式 0 中被唤醒，唤醒后该位转换为高需 1~2 个周期。

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，用于表明唤醒后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值为“1”。在休眠模式或空闲模式 0 中，该标志会处于低电平状态，若唤醒后该标志位将在 15~16 个时钟周期后变为高电平状态。

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_{SUB}
1：f_H
此位用于选择 f_H 或 f_H/2~ f_H/64 或 f_{SUB} 作为系统时钟。该位为高时选择 f_H 作为系统时钟，为低时则选择 f_H/2~ f_H/64 或 f_{SUB} 作为系统时钟。当系统时钟由 f_H 时钟向 f_{SUB} 时钟转换时，f_H 将自动关闭以降低功耗。

- **CTRL 寄存器**

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

“x”：未知

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

Bit 6~3 未使用，读为“0”

Bit 2 **LVRF**：LVR 复位标志位
详见其它章节

Bit 1 **LRF**：LVRC 控制寄存器软件复位标志位
详见其它章节

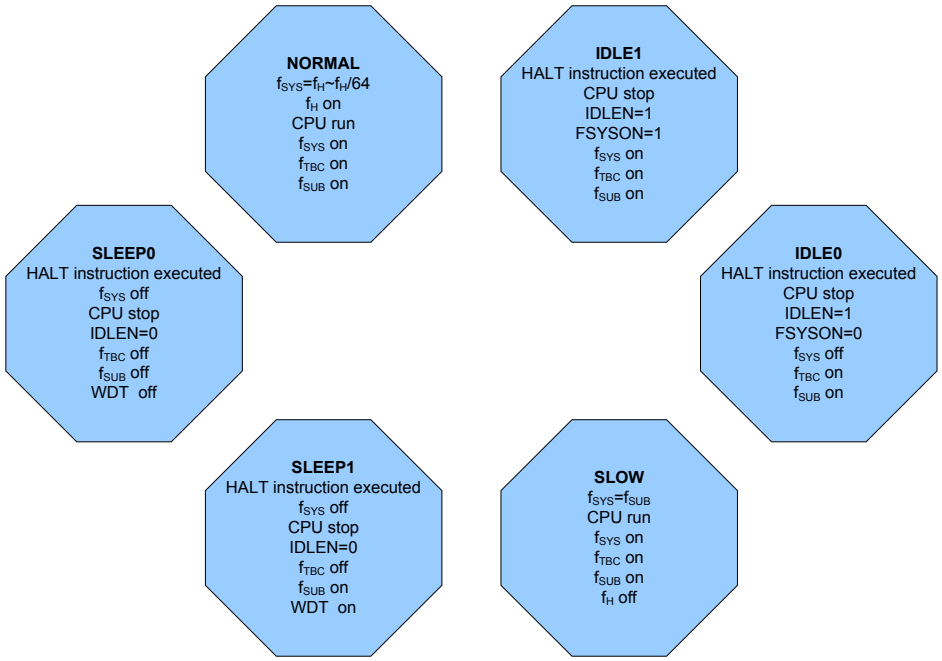
Bit 0 **WRF**：WDTC 控制寄存器软件复位标志位
详见其它章节

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

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

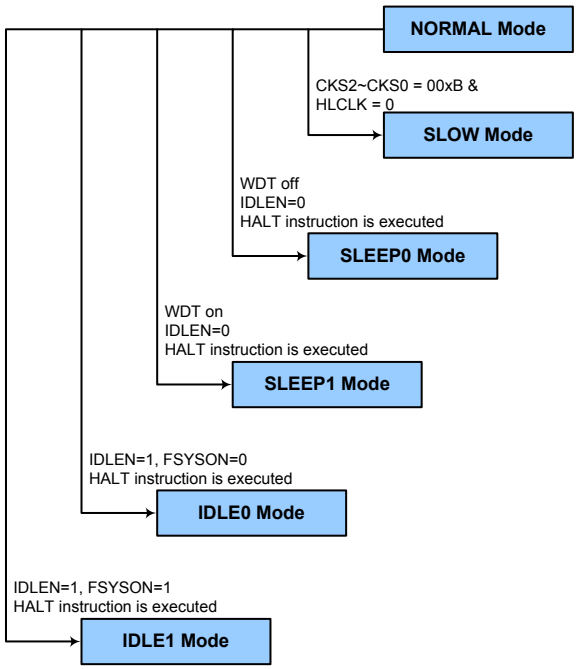
当 HLCLK 位变为低电平时，时钟源将由高速时钟源 f_H 转换成时钟源 $f_H/2$~$f_H/64$ 或 f_{SUB}。若时钟源来自 f_{SUB}，高速时钟源将停止运行以节省耗电。此时必须注意内部时钟源 $f_H/16$~$f_H/64$ 也将停止运行，这可能会影响到其它内部功能如 TM 的操作。所附流程图显示了单片机在不同工作模式间切换时的变化。



正常模式切换到低速模式

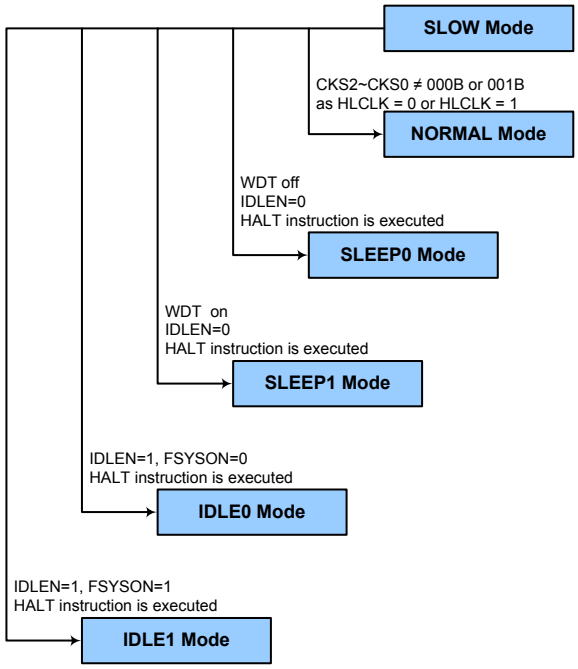
系统运行在正常模式时使用高速系统振荡器，因此较为耗电。可通过设置 SMOD 寄存器中的 HLCLK 位为“0”及 CKS2~CKS0 位为“000”或“001”使系统时钟切换至运行在低速模式下。此时将使用低速系统振荡器以节省耗电。用户可在对性能要求不高的操作中使用此方法以减少耗电。

低速模式的时钟源来自 LIRC 振荡器，因此要求该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 位监测。



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。切换回使用高速系统时钟振荡器的正常模式需设置 HLCLK 位为“1”，也可设置 HLCLK 位为“0”但 CKS2~CKS0 需设为“010”、“011”、“100”、“101”、“110”或“111”。高频时钟需要一定的稳定时间，通过检测 HTO 位的状态可进行判断。高速振荡器稳定需要多少的延时取决于所选的高速振荡器类型。



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 除能。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟、WDT 时钟与时基时钟将停止运行，应用程序停止在“HALT”指令处。
- 数据存储器中的内容和寄存器将保持当前值。
- 无论 WDT 时钟源是来自 f_{SUB} 或来自系统时钟，WDT 将被清零并停止运行。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

进入休眠模式 1

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

- 系统时钟与时基时钟将停止运行，应用程序停止在“HALT”指令处，而 WDT 继续运行，其时钟源来自 f_{SUB}。
- 数据存储器中的内容和寄存器将保持当前值。
- 若看门狗定时器功能使能，则 WDT 将被清零并重新开始计数；若看门狗定时器功能除能，WDT 将被清零并停止计数。
- 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 CTRL 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，但时基时钟与 f_{SUB} 时钟将继续运行。
- 数据存储器中的内容和寄存器将保持当前值。
- 若看门狗定时器功能使能，则 WDT 将被清零并重新开始计数；若看门狗定时器功能除能，WDT 将被清零并停止计数。
- 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 CTRL 寄存器中的 FSYSON 位为“1”。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟、时基时钟和 f_{SUB} 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。
- 若看门狗定时器功能使能，则 WDT 将被清零并重新开始计数；若看门狗定时器功能除能，WDT 将被清零并停止计数。
- 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单片机的电流降低到尽可能低，可能到只有几个微安的级别（空闲模式 1 除外），所以如果要将电路的电流降到最低，电路设计者还应有其它的考虑。应该特别注意的是单片机的输入 / 输出引脚。所有高阻抗输入脚都必须连接到固定的高或低电平，因为引脚浮空会造成内部振荡并导致耗电增加。这也应用于有不同封装的单片机，因为它们可能含有未引出的引脚，这些引脚也必须设为输出或带有上拉电阻的输入。

另外还需注意单片机设为输出的 I/O 引脚上的负载。应将它们设置在有最小拉电流的状态或将它们和其它的 CMOS 输入一样接到没有拉电流的外部电路上。

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：

- PA 口下降沿
- 系统中断
- WDT 溢出

若系统由 WDT 溢出唤醒，将会发生看门狗定时器复位。以上这些方法都可以启动复位，可以通过检测状态寄存器中 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_S。f_S 的时钟源由 LIRC 振荡器提供。LIRC 内部振荡器在 5V 供电下有一个约为 32kHz 的频率，应注意此指定的内部时钟可随 V_{DD}，温度以及制成工艺的不同而变化。看门狗定时器的时钟源可分频为 2^8~2^{18} 以提供更大的溢出周期，分频比由 WDTC 寄存器中的 WS2~WS0 位来决定。

看门狗定时器控制寄存器

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

- **WDTC 寄存器**

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

Bit 7~3 **WE4~WE0**：WDT 功能软件控制位
10101：除能
01010：使能
其它值：MCU 复位
若由于不利的环境因素或通过软件设置使这些位发生改变，单片机将复位。复位操作将在 2~3 个 LIRC 时钟周期后执行，且 CTRL 寄存器中的 WRF 位将置高。

Bit 2~0 **WS2~WS0**：WDT 溢出周期选择位
000：$2^8/f_S$
001：$2^{10}/f_S$
010：$2^{12}/f_S$
011：$2^{14}/f_S$
100：$2^{15}/f_S$
101：$2^{16}/f_S$
110：$2^{17}/f_S$
111：$2^{18}/f_S$

- **CTRL 寄存器**

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

“x”：未知

Bit7 **FSYSON**：空闲模式下 f_{SYS} 控制位
详见其它章节

Bit 6~3 未定义，读为“0”

Bit 2 **LVRF**：LVR 复位标志位
详见其它章节

Bit 1 **LRF**：LVRC 控制寄存器软件复位标志位
详见其它章节

Bit 0 **WRF**：WDTC 控制寄存器软件复位标志位
0：未发生
1：发生
当 WDTC 控制寄存器发生复位时，此位置为“1”，通过应用程序清零。应注意的是该位只能通过应用程序清零。

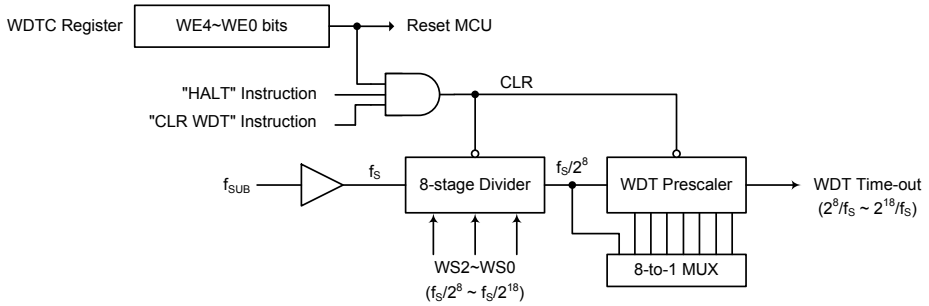
看门狗定时器操作

当 WDT 溢出时，它产生一个芯片复位的动作。这也就意味着正常工作期间，用户需在应用程序中看门狗溢出前有策略地清看门狗定时器以防止其产生复位，可使用清除看门狗指令实现。无论什么原因，程序失常跳转到一个未知的地址或进入一个死循环，这些清除指令都不能被正确执行，此种情况下，看门狗将溢出以使单片机复位。

看门狗定时器控制寄存器 WDTC 中的 WE4~WE0 位段可用于看门狗定时器的使能 / 除能以及复位控制。当设置 WE4~WE0 位段的值为 10101B 时将除能 WDT 功能。若 WE4~WE0 位段的值为 01010B 则 WDT 功能使能。当此位段在环境噪声或软件设置的影响下设为除 01010B 和 10101B 以外的值，则单片机将在 2~3 个 LIRC 时钟周期后复位。上电后此位段的值为 01010B。

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 被置位，程序计数器 PC 和堆栈指针 SP 将被复位。有三种方法可以用来清除 WDT 的内容。第一种是 WDT 复位，即写入除 01010B 和 10101B 外的任何值到 WE4~WE0 位段，而第二种是通过 WDT 软件清除指令，第三种是通过“HALT”指令。只有一种软件指令用于清除看门狗定时器。因此只要执行“CLR WDT”便清除 WDT。

当设置分频比为 2^{18} 时，溢出周期最大。例如，时钟源为 32kHz LIRC 振荡器，分频比为 2^{18} 时最大溢出周期约 8s，分频比为 2^8 时最小溢出周期约 7.8ms。



看门狗定时器

复位和初始化

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

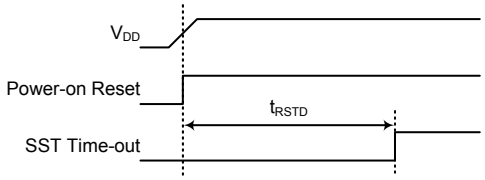
另一种复位为看门狗溢出单片机复位。不同方式的复位操作会对寄存器产生不同的影响。此外，还有一种复位为低电压复位即 LVR 复位，在电源供应电压低于一定的阈值时，系统会产生完全复位。

复位功能

通过内部事件触发，此单片机有以下几种复位方式：

上电复位

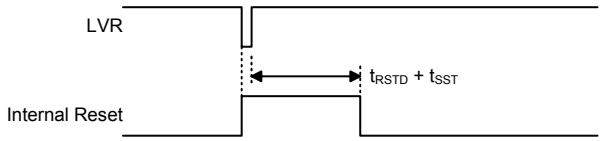
这是最基本且不可避免的复位，发生在单片机初次上电后。除了保证程序存储器从开始地址执行，上电复位也使得其它寄存器被设定在预设条件。所有的输入 / 输出端口控制寄存器在上电复位时会保持高电平，以确保上电后所有引脚被设定为输入状态。



上电复位时序图

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。LVR 功能始终使能，其指定 LVR 电压为 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且 CTRL 寄存器的 LVRF 标志位会被置位。一个有效的 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。V_{LVR} 参数值固定为 2.1V。若由于受到干扰 LVS7~LVS0 变为其它值时，单片机将在 2~3 个 LIRC 时钟周期后复位。此时 CTRL 寄存器的 LRF 位被置位。上电后 LVRC 寄存器的值为 01010101B。应注意当单片机进入休眠或空闲模式时 LVR 功能将自动除能。



低电压复位时序图

- **LVRC 寄存器**

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

Bit 7~0 **LVS7 ~ LVS0**：LVR 电压选择位
01010101：2.1V
00110011：2.1V
10011001：2.1V
10101010：2.1V
其它值：MCU 复位 – 寄存器复位至 POR 值
当发生低电压情况时，将产生 MCU 复位。复位操作将在 2~3 个 LIRC 时钟周期后执行。这种复位后寄存器的内容将保持不变。
如果 LVRC 寄存器写入除上面 4 个值以外的其它值也会引起单片机复位。复位操作将在 2~3 个 LIRC 时钟周期后执行。此种情况复位后寄存器值将恢复到 POR 值。

- **CTRL 寄存器**

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

“x”：未知

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

Bit 6~3 未使用，读为“0”

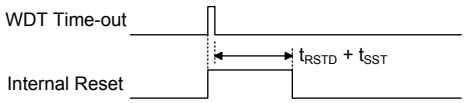
Bit 2 **LVRF**：LVR 复位标志位
0：未发生
1：发生
当 LVR 复位情况发生时，该位被置为“1”，此位只能通过应用程序清零。

Bit 1 **LRF**：LVRC 控制寄存器软件复位标志位
0：未发生
1：发生
当 LVRC 寄存器包含未定义的 LVR 电压值时，该位被置为“1”，此复位为软件复位功能。该位只能通过应用程序清零。

Bit 0 **WRF**：WDTC 控制寄存器复位标志位
详见其它章节

正常运行时看门狗溢出复位

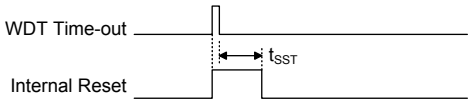
除了看门狗溢出标志位 TO 将被置高外，正常运行时的看门狗溢出复位与 LVR 复位相同。



正常运行时 WDT 溢出复位时序图

休眠或空闲模式时看门狗溢出复位

休眠或空闲时看门狗溢出复位和其它类型的复位有些不同。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。



休眠或空闲模式时 WDT 溢出复位时序图

复位初始条件

不同的复位形式以不同的方式影响复位标志位。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

注：“u”表示不改变

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	复位后清零，WDT 重新开始计数
定时器模块	所有定时器模块停止
输入 / 输出口	I/O 口设为输入模式
堆栈指针	堆栈指针指向堆栈顶端

不同的复位形式对单片机内部寄存器的影响是不同的。为保证复位后程序能正常执行，了解寄存器在特定条件复位后的设置是非常重要的。下表即为不同方式复位后内部寄存器的状况。注意若单片机存在多种封装类型，则该表反应较大封装类型的情况。

寄存器	上电复位	WDT 溢出 (正常操作)	WDT 溢出 (HALT)*
MP0	xxxx xxxx	xxxx xxxx	uuuu uuuu
MP1	xxxx xxxx	xxxx xxxx	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu
TBHP	---- xxxx	---- uuuu	---- uuuu
STATUS	--00 xxxx	--1u uuuu	--11 uuuu
SMOD	000- 0011	000- 0011	uuu- uuuu
INTEG	---- 0000	---- 0000	---- uuuu
INTC0	-0-0 0-00	-0-0 0-00	-u-u u-uu
INTC1	00-0 00-0	00-0 00-0	uu-u uu-u
INTC2	--00 --00	--00 --00	--uu --uu
MFI0	--00 --00	--00 --00	--uu --uu
MFI1	--00 --00	--00 --00	--uu --uu
PA	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	uuuu uuuu
PAPU	0000 0000	0000 0000	uuuu uuuu
PAWU	0000 0000	0000 0000	uuuu uuuu
TMPC	0--- --00	0--- --00	u--- --uu
WDTC	0101 0011	0101 0011	uuuu uuuu
TBC	0011 -111	0011 -111	uuuu -uuu
CTRL	0--- -x00	0--- -000	u--- -uuu
LVRC	0101 0101	0101 0101	uuuu uuuu
ADRL(ADRFS=0)	xxxx ----	xxxx ----	uuuu ----
ADRL(ADRFS=1)	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH(ADRFS=0)	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH(ADRFS=1)	---- xxxx	---- xxxx	---- uuuu
ADCR0	0110 -000	0110 -000	uuu- -uuu
ADCR1	00-0 -000	00-0 -000	uu-u -uuu
ACERL	1111 1111	1111 1111	uuuu uuuu
PB	-111 1111	-111 1111	-uuu uuuu
PBC	-111 1111	-111 1111	-uuu uuuu
PBPU	-000 0000	-000 0000	-uuu uuuu
STMC0	0000 0000	0000 0000	uuuu uuuu
STMC1	0000 0000	0000 0000	uuuu uuuu
STMDL	0000 0000	0000 0000	uuuu uuuu

寄存器	上电复位	WDT 溢出 (正常操作)	WDT 溢出 (HALT)*
STMDH	---- --00	---- --00	---- --uu
STMAL	0000 0000	0000 0000	uuuu uuuu
STMAH	---- --00	---- --00	---- --uu
PTMC0	0000 0---	0000 0---	uuuu u---
PTMC1	0000 0000	0000 0000	uuuu uuuu
PTMDL	0000 0000	0000 0000	uuuu uuuu
PTMDH	---- --00	---- --00	---- --uu
PTMAL	0000 0000	0000 0000	uuuu uuuu
PTMAH	---- --00	---- --00	---- --uu
PTMRPL	0000 0000	0000 0000	uuuu uuuu
PTMRPH	---- --00	---- --00	---- --uu
PC	---- -111	---- -111	---- -uuu
PCC	---- -111	---- -111	---- -uuu
PCPU	---- -000	---- -000	---- -uuu

注：“-”表示未定义
“u”表示未改变
“x”表示未知

输入 / 输出端口

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

该单片机提供 PA~PC 双向输入 / 输出口。这些 I/O 端口映射到 RAM 数据存储器中的特定地址上，如特殊功能数据存储器表所示。所有 I/O 口可用于输入输出操作。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]”时 T2 的上升沿准备好，m 为端口地址。对于输出操作，所有数据都是被锁存的，且保持不变直到输出锁存被重写。

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

输入 / 输出逻辑功能寄存器列表

上拉电阻

许多产品应用在端口处于输入状态时，通常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，当引脚配置为输入时，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~PCPU 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	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
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 **PAPU7~PAPU0**：PA 口 bit 7~bit 0 上拉控制位
0：除能
1：使能

- **PBPU 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7　　未定义，读为“0”

Bit 6~0　　**PBPU6~PBPU0**：PB 口 bit 6~bit 0 上拉控制位
0：除能
1：使能

- **PCPU 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	PCPU2	PCPU1	PCPU0
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3　　未定义，读为“0”

Bit 2~0　　**PCPU2~PCPU0**：PC 端口 bit 3~bit 0 上拉控制位
0：除能
1：使能

PA 口唤醒

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

- **PAWU 寄存器**

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

Bit 7~0　　**PAWU7~PAWU0**：PA 口 bit 7~bit 0 唤醒控制位
0：除能
1：使能

输入 / 输出端口控制寄存器

每一个输入 / 输出口都具有各自的控制寄存器，即 PAC~PCC，用来控制输入 / 输出配置。从而每个 I/O 引脚都可以在软件控制下，动态地设置为 CMOS 输出或输入。I/O 端口的每个引脚都映射到其相关端口控制寄存器中的一个位。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。但应注意，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	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
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 **PAC7~PAC0**：PA 口 bit 7~bit 0 输入 / 输出类型选择位
0：输出
1：输入

- **PBC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	1	1	1	1	1	1	1

Bit 7 未定义，读为“0”

Bit 6~0 **PBC6~PBC0**：PB 口 bit 6~bit 0 输入 / 输出类型选择位
0：输出
1：输入

- **PCC 寄存器**

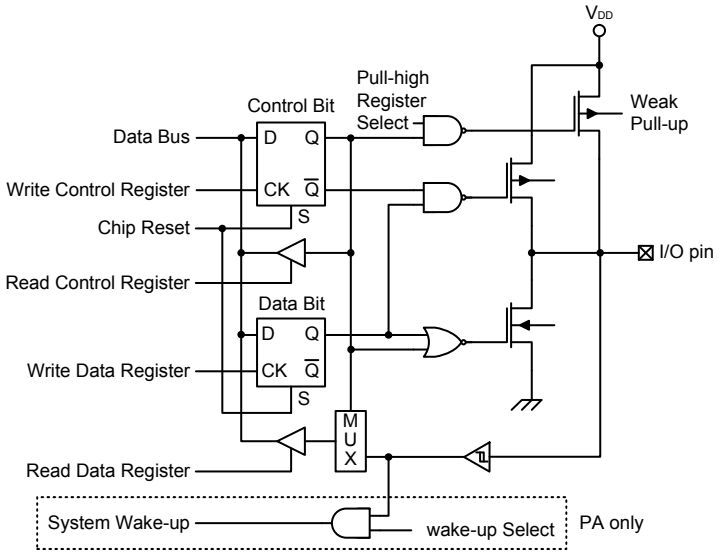
Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	PCC2	PCC1	PCC0
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	1	1	1

Bit 7~3 未定义，读为“0”

Bit 2~0 **PCC2~PCC0**：PC 口 bit 2~bit 0 输入 / 输出类型选择位
0：输出
1：输入

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。该单片机有大量的引脚共用结构，在此无法显示所有类型。



逻辑功能输入 / 输出结构

编程注意事项

在编程中，最先要考虑的是端口的初始化。复位之后，所有的输入 / 输出数据及端口控制寄存器都将被设为逻辑高。所有输入 / 输出引脚默认为输入状态，而其电平则取决于其它相连接电路以及是否选择了上拉电阻。如果端口控制寄存器 PAC~PCC，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~PC 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。

A/D 转换器控制寄存器的上电复位条件使得任意的 A/D 输入引脚 (始终与其他 I/O 口功能共用) 在复位后将被设置为模拟输入，但此时 A/D 转换器并不启动。因此必须注意的是若需将这些引脚用作 I/O 数字输入引脚或其他功能，必须正确配置 A/D 转换器控制寄存器以移除 A/D 功能。还应注意的是，由于 A/D 通道使能，内部连接的上拉电阻将被移除。

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

定时器模块 – TM

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

这里只介绍各种 TM 的共性，更多详细资料请参考标准型和周期型定时器章节。

简介

该单片机含有一个 10-bit 标准型 TM–STM 以及一个 10-bit 周期型 TM–PTM。虽然性质相似，但不同 TM 特性复杂度不同。本章介绍标准型和周期型 TM 的共性，更多详细资料见后面各相应章节。两种类型 TM 的特性和区别见下表。

功能	STM	PTM
定时 / 计数器	√	√
输入捕捉	√	√
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

这两种不同类型的 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 STM 与 PTM 控制寄存器的 STCK2~STCK0 与 PTCK2~PTCK0 位段可选择所需的时钟源。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 STCK 与 PTCK 引脚。STCK 与 PTCK 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

标准型和周期型 TM 各有两个内部中断，即内部比较器 A 和比较器 P，当发生比较匹配时比较器将产生一个 TM 中断。中断产生时将清除计数器，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，分别为 STCK、PTCK 与 STPI、PTPI 引脚。STCK、PTCK 作为 TM 的时钟源输入脚，通过设置 STMC0 或 PTMC0 寄存器中的 STCK2~STCK0 或 PTCK2~PTCK0 位进行选择。外部时钟源可通过该引脚来驱动内部 TM。TM 输入引脚可以选择上升沿或下降沿有效。

另一个 TM 输入引脚 STPI 与 PTPI 为捕捉输入引脚，可通过分别配置 STMC1 与 PTMC1 寄存器中的 STIO1~STIO0 与 PTIO1~PTIO0 位选择上升沿、下降沿或双沿为有效边沿转换类型。此外在 PTM 捕捉输入模式中，除 PTPI 引脚外，PTCK 也可用作外部触发输入源。

每个 TM 有一个输出引脚即 STP 与 PTP。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 STP 与 PTP 输出引脚也被 TM 用来产生 PWM 输出波形。由于 TM 输入或输出引脚与其它功能共用，TM 输出功能需要通过寄存器先被设置。寄存器中的单个位决定了其相关引脚是用作外部 TM 输出引脚还是其它功能。当相应的 TM 配置 STP 或 PTP 为输出引脚，则相应的引脚硬背设置为外部 TM 输出引脚。若 TM 配置 STP 或 PTP 为输入引脚，则供给相应引脚的信号可来自外部或是其他功能共享输出功能。若 TM 配置 STP 或 PTP 功能未使用，则相应引脚由其他共享功能控制。

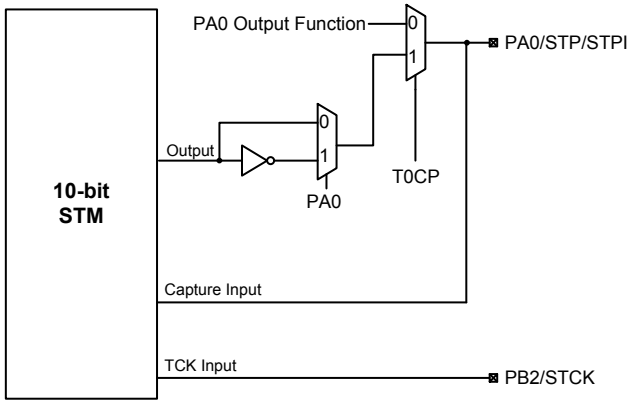
关于每种 TM 类型的引脚细节见下表。

STM		PTM	
Input	**Output**	**Input**	**Output**
STCK，STPI	STP	PTCK，PTPI	PTP

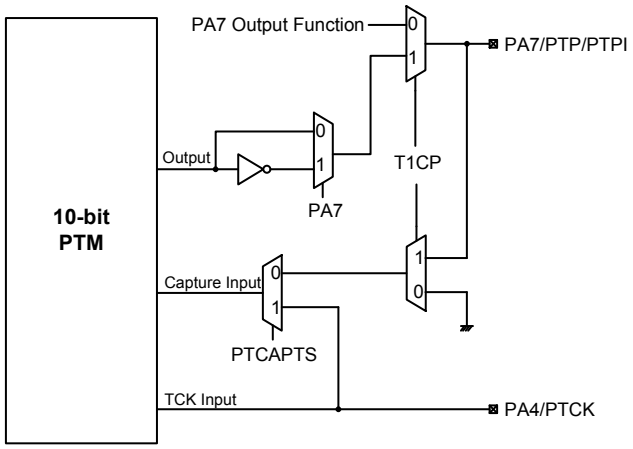
TM 外部引脚

TM 输入 / 输出引脚控制寄存器

选择作为 TM 输入 / 输出或其它共用功能可通过配置一个寄存器中与 TM 输入 / 输出引脚相对应的单个位实现。设置此位为高则相应引脚功能为 TM 输入 / 输出，若此位清零则该引脚具有其它功能。



STM 功能引脚控制方框图



PTM 功能引脚控制方框图

- **TMPC 寄存器**

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

Bit 7 **CLOP**：CLO 引脚控制位
0：除能
1：使能

Bit 6~2 未定义，读为“0”

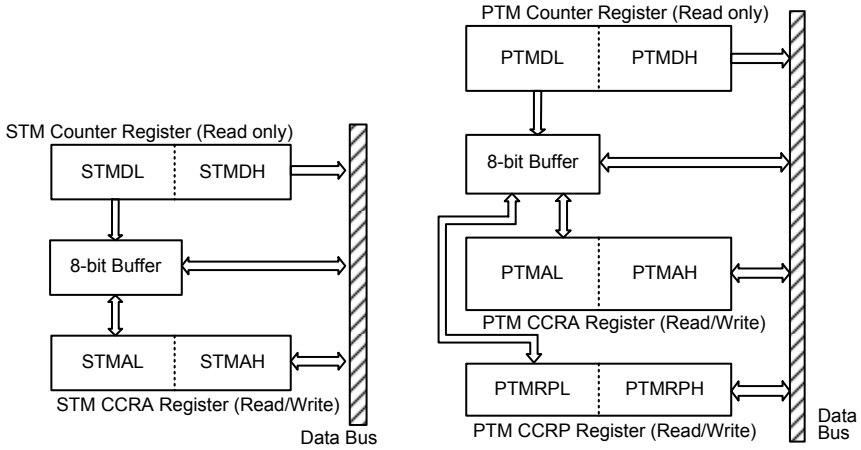
Bit 1 **T1CP**：PTP 引脚控制位
0：除能
1：使能

Bit 0 **T0CP**：STP 引脚控制位
0：除能
1：使能

编程注意事项

TM 计数器寄存器和捕捉 / 比较 CCRA 和 CCRP 寄存器，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。读写这些成对的寄存器需通过特殊的方式。值得注意的是 8-bit 缓存器的存取数据及相关低字节的读写操作仅在其相应的高字节读取操作执行时发生。

由于 CCRA 和 PTM CCRP 寄存器访问方式如下图所示，读写这些成对的寄存器需通过特殊的方式。建议使用“MOV”指令，通过以下步骤访问 CCRA 和 PTM CCRP 低字节寄存器，STMAL、PTMAL 与 PTMRPL。否则将导致不可预期的结果。

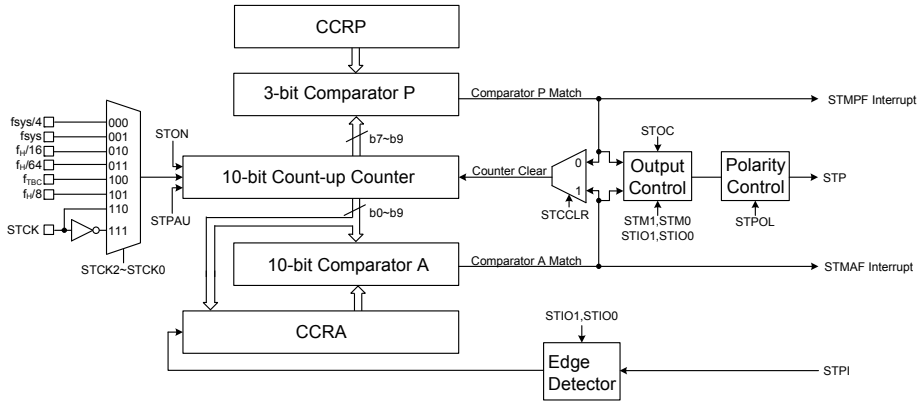


读写流程如下步骤所示：

- 写数据至 CCRA 或 PTM CCRP
  - ♦ 步骤 1. 写数据至低字节寄存器 STMAL、PTMAL 或 PTMRPL
    - – 注意，此时数据仅写入 8-bit 缓存器。
  - ♦ 步骤 2. 写数据至高字节寄存器 STMAH、PTMAH 或 PTMRPH
    - – 注意，此时数据直接写入高字节寄存器，同时锁存在 8-bit 缓存器中的数据写入低字节寄存器。
- 由计数器寄存器和 CCRA 或 PTM CCRP 中读取数据
  - ♦ 步骤 1. 从高字节寄存器 STMDH、PTMDH、STMAH、PTMAH 或 PTMRPH 中读取数据
    - – 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. 从低字节寄存器 STMDL、PTMDL、STMAL、PTMAL 或 PTMRPL 中读取数据
    - – 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 可由两个外部输入脚控制并驱动一个外部输出脚。



标准型 TM 方框图

标准型 TM 操作

标准型 TM 的核心是一个由用户选择的内部时钟源驱动的 10-bit 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-bit 计数器值的唯一方法是使 STON 位发生上升沿跳变清除计数器。此外，计数器溢出或比较匹配也会自动清除计数器。上述条件发生时，通常情况会产生 STM 中断信号。标准型 TM 可工作在不同的模式，可由包括来自输入脚的不同时钟源驱动，也可以控制输出脚。所有工作模式的设定都是通过设置相关寄存器来实现的。

标准型 TM 寄存器介绍

标准型 TM 的所有操作由一系列寄存器控制。一对只读寄存器用来存放 10-bit 计数器的值，一对读 / 写寄存器存放 10-bit CCRA 的值，剩下两个控制寄存器设置不同的操作和控制模式以及 3 位 CCRP 的值。

寄存器名称	位							
	7	6	5	4	3	2	1	0
STMC0	STPAU	STCK2	STCK1	STCK0	STON	STRP2	STRP1	STRP0
STMC1	STM1	STM0	STIO1	STIO0	STOC	STPOL	STDPX	STCCLR
STMDL	D7	D6	D5	D4	D3	D2	D1	D0
STMDH	—	—	—	—	—	—	D9	D8
STMAL	D7	D6	D5	D4	D3	D2	D1	D0
STMAH	—	—	—	—	—	—	D9	D8

10-bit 标准型 TM 寄存器列表

- **STMC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	STPAU	STCK2	STCK1	STCK0	STON	STRP2	STRP1	STRP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **STPAU**：STM 计数器暂停控制位

0：运行

1：暂停

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

Bit 6~4 **STCK2~STCK0**：STM 计数器时钟选择位

000：$f_{SYS}/4$

001：f_{SYS}

010：$f_H/16$

011：$f_H/64$

100：f_{TBC}

101：$f_H/8$

110：STCK 上升沿时钟

111：STCK 下降沿时钟

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

Bit 3 **STON**：STM 计数器 On/Off 控制位

0：Off

1：On

此位控制 STM 的总开关功能。设置此位为高则使能计数器使其运行，清零此位则除能 STM。清零此位将停止计数器并关闭 STM 以减少耗电。当此位经由低到高转换时，内部计数器将复位清零，当此位由高到低转换时，内部计数器将保持其剩余值直到此位再次变高。若 STM 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 STON 位经由低到高转换时，STM 输出脚将复位至 STOC 位指定的初始值。

Bit 2~0 **STRP2~STRP0**：STM CCRP 3-bit 寄存器，与 STM 计数器 bit 9~bit 7 比较比较器 P 匹配周期

000：1024 个 STM 时钟

001：128 个 STM 时钟

010：256 个 STM 时钟

011：384 个 STM 时钟

100：512 个 STM 时钟

101：640 个 STM 时钟

110：768 个 STM 时钟

111：896 个 STM 时钟

此三位设定内部 CCRP 3-bit 寄存器的值，然后与内部计数器的高三位进行比较。如果 STCCLR 位设定为 0 时，选中该比较结果清除内部计数器。STCCLR 位设为 0，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，会使得计数器在最大值溢出。

- **STMC1 寄存器**

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

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

Bit 5~4 **STIO1~STIO0**：选择 STM 功能位
比较匹配输出模式
00：无变化
01：输出低
10：输出高
11：输出翻转
PWM 输出模式 / 单脉冲输出模式
00：PWM 输出无效状态
01：PWM 输出有效状态
10：PWM 输出
11：单脉冲输出
捕捉输入模式
00：在 STPI 上升沿输入捕捉
01：在 STPI 下降沿输入捕捉
10：在 STPI 双沿输入捕捉
11：输入捕捉除能
定时 / 计数器模式
未使用
此两位用于决定在一定条件达到时 STM 输出脚如何改变状态。这两位值的选择取决于 STM 运行在哪种模式下。
在比较匹配输出模式下，STIO1 和 STIO0 位决定了当比较器 A 比较匹配发生时 STM 输出脚如何改变状态。当比较器 A 比较匹配发生时 STM 输出脚能设为切换高、切换低或翻转当前状态。若此两位同时为 0 时，这个输出将不会改变。STM 输出脚的初始值通过 STOC 位设置取得。注意，由 STIO1 和 STIO0 位得到的输出电平必须与通过 STOC 位设置的初始值不同，否则当比较匹配发生时，STM 输出脚将不会发生变化。在 STM 输出脚改变状态后，通过 STON 位由低到高电平的转换复位至初始值。
在 PWM 输出模式，STIO1 和 STIO0 用于决定比较匹配条件发生时怎样改变 STM 输出脚的状态。PWM 输出功能通过这两位的变化进行更新。仅在 STM 关闭时改变 STIO1 和 STIO0 位的值是很有必要的。若在 STM 运行时改变 STIO1 和 STIO0 的值，PWM 输出的值是无法预料的。

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

Bit 2 **STPOL**：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 STM 输出脚反相，为低时 STM 输出脚同相。若 STM 处于定时 / 计数器模式时其不受影响。

Bit 1 **STDPX**：STM PWM 周期 / 占空比控制位
0：CCRP - 周期；CCRA - 占空比
1：CCRP - 占空比；CCRA - 周期
此位决定 CCRA 与 CCRP 寄存器哪个被用于 PWM 波形的周期和占空比控制。

Bit 0 **STCCLR**：选择 STM 计数器清零条件位
0：STM 比较器 P 匹配
1：STM 比较器 A 匹配
此位用于选择清除计数器的方法。标准型 TM 包括两个比较器 - 比较器 A 和比较器 P。这两个比较器每个都可以用作清除内部计数器。STCCLR 位设为高，计数器在比较器 A 比较匹配发生时被清除；此位设为低，计数器在比较器 P 比较匹配发生或计数器溢出时被清除。计数器溢出清除的方法仅在 CCRP 被清除为 0 时才能生效。STCCLR 位在 PWM 输出、单脉冲或输入捕捉模式时未使用。

- **STMDL 寄存器**

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

Bit 7~0 **D7~D0**：STM 计数器低字节寄存器 bit 7 ~ bit 0
STM 10-bit 计数器 bit 7 ~ bit 0

- **STMDH 寄存器**

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

Bit 7~2 未定义，读为“0”

Bit 1~0 **D9~D8**：STM 计数器高字节寄存器 bit 1~bit 0
STM 10-bit 计数器 bit 9~bit 8

- **STMAL 寄存器**

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

Bit 7~0 **D7~D0**：STM CCRA 低字节寄存器 bit 7 ~ bit 0
STM 10-bit CCRA bit 7 ~ bit 0

- **STMAH 寄存器**

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

Bit 7~2 未定义，读为“0”

Bit 1~0 **D9~D8**：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

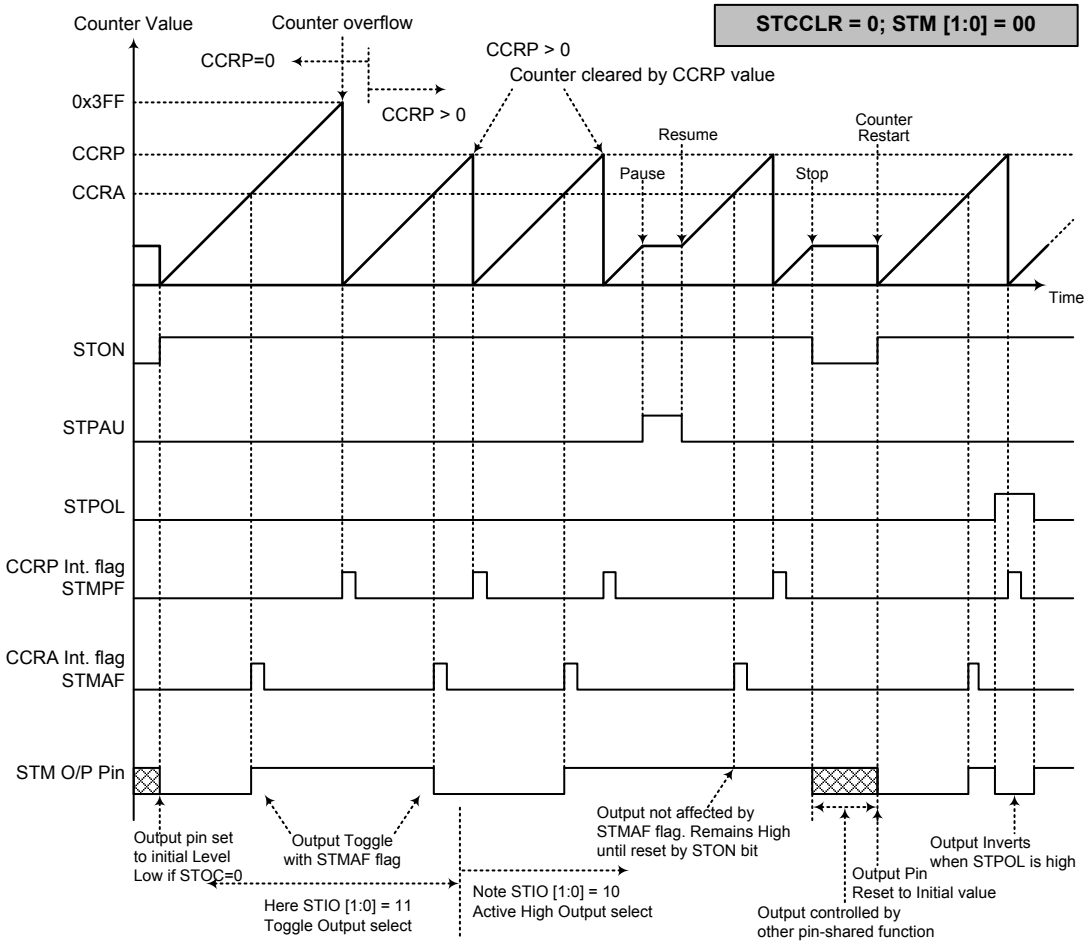
标准型 TM 有五种工作模式，即比较匹配输出模式，PWM 输出模式，单脉冲输出模式，捕捉输入模式或定时 / 计数器模式。通过设置 STMC1 寄存器的 STM1 和 STM0 位选择任意模式。

比较匹配输出模式

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

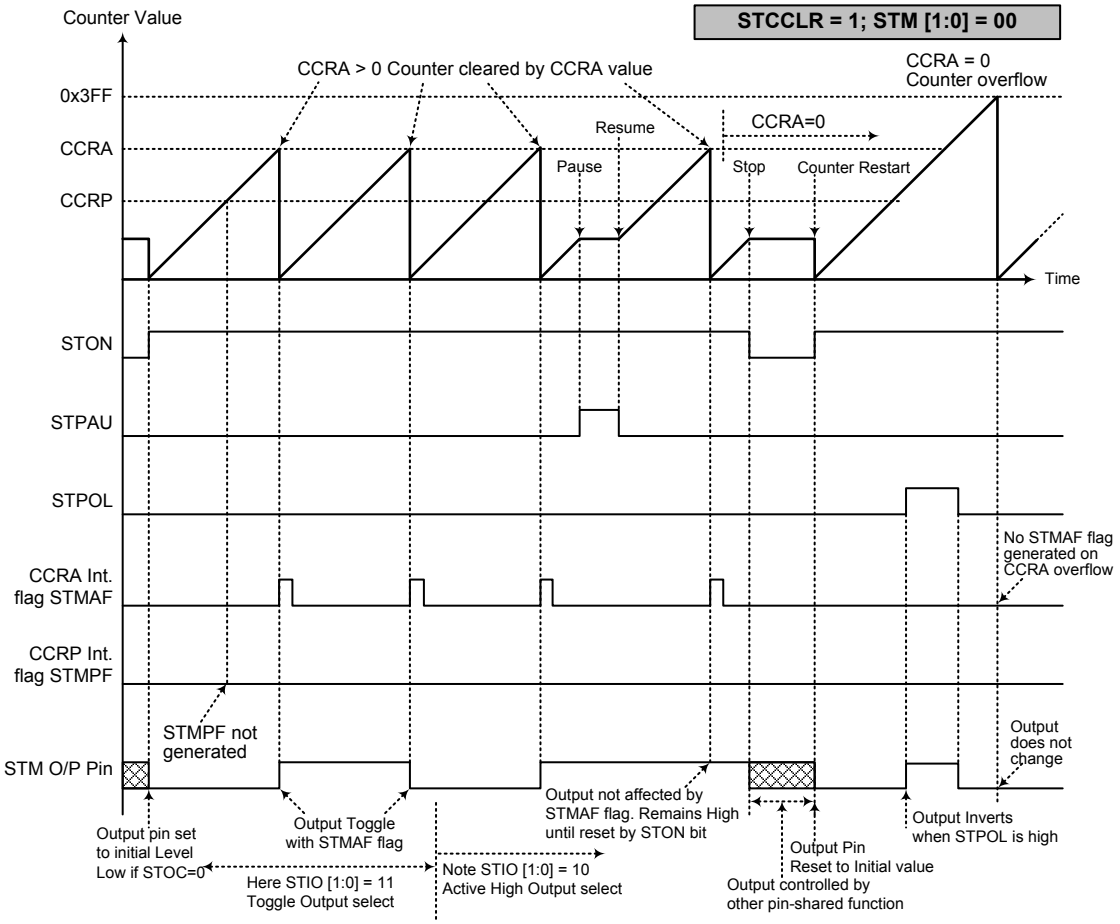
如果 STMC1 寄存器的 STCCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅产生 STMAF 中断请求标志。所以当 STCCLR 为高时，不会产生 STMPF 中断请求标志。如果 CCRA 被清零，当计数达到最大值 3FFH 时，计数器溢出，而此时不产生 STMAF 请求标志。在比较匹配输出模式中，CCRA 不可置为“0”。

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



比较匹配输出模式 – **STCCLR=0**

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



比较匹配输出模式 – **STCCLR=1**

注：1. STCCLR=1，比较器 A 匹配将清除计数器
2. TM 输出引脚仅由 STMAF 标志位控制
3. 输出引脚在 STON 上升沿复位至初始值
4. 当 STCCLR=1 时，不产生 STMPF 标志位

定时 / 计数器模式

要工作在此模式，STMC1 寄存器中的 STM1 和 STM0 应设置为 11。定时 / 计数器模式操作方式与比较匹配输出模式相同，产生相同的中断标志位。唯一不同的就是在定时 / 计数器模式中未使用 STM 输出引脚。因此，比较匹配输出模式中的描述和时序图可以帮助理解此功能。该模式中未使用的 STM 输出脚用作普通 I/O 脚或其它功能。

PWM 输出模式

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

由于 PWM 波形的周期和占空比可调，其波形的选择就极其灵活。在 PWM 模式中，STCCLR 位不影响 PWM 周期。CCRA 和 CCRP 寄存器决定 PWM 波形，一个用来清除内部计数器并控制 PWM 波形的频率，另一个用来控制占空比。哪个寄存器控制频率或占空比取决于 STMC1 寄存器的 STDPX 位。因此，PWM 波形的频率和占空比受 CCRA 和 CCRP 寄存器中的值控制。

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

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

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

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

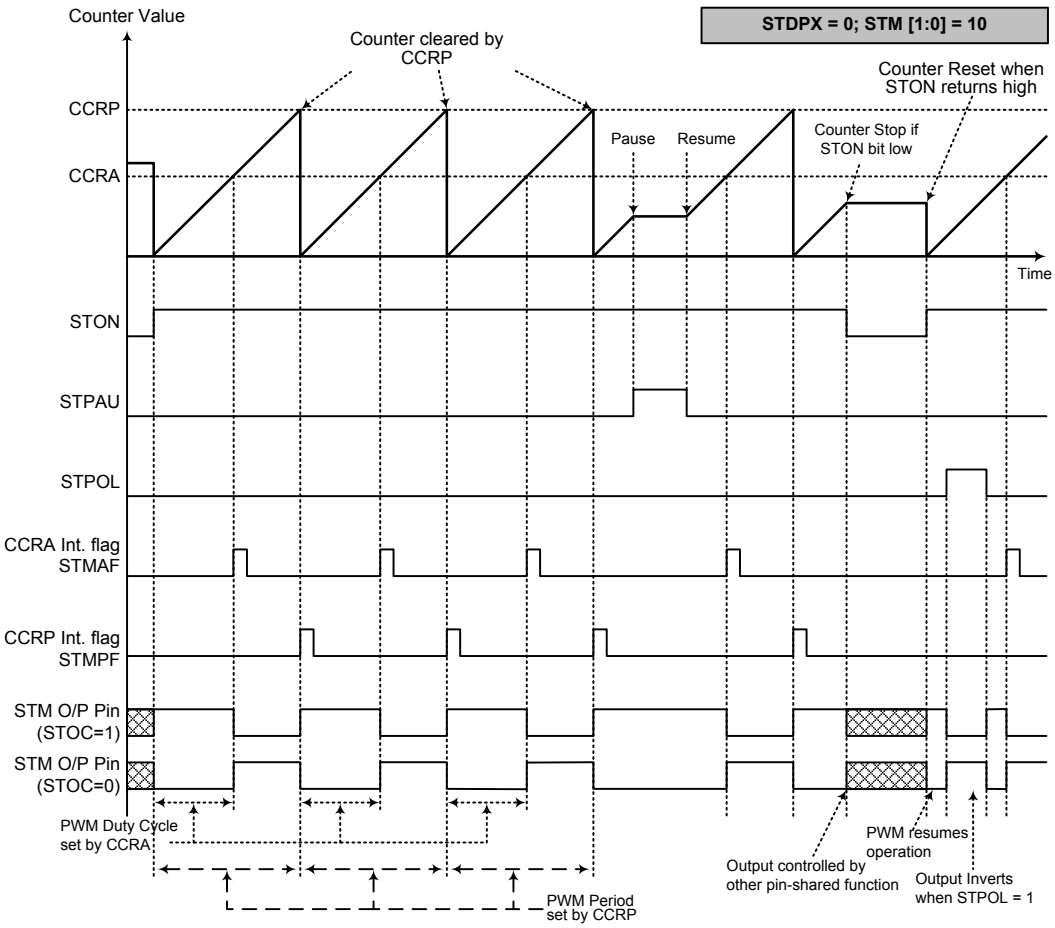
STM PWM 输出频率 =(f_{SYS}/4)/(2×128)=f_{SYS}/1024=16kHz，周期 =128/(2×128)=50%.

若 CCRA 所定义的 Duty 值等于或大于 Period 值，则 PWM 输出占空比为 100%.

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

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

PWM 的输出周期由 CCRA 寄存器的值与 TM 的时钟共同决定，PWM 的占空比由 CCRP 的值决定。



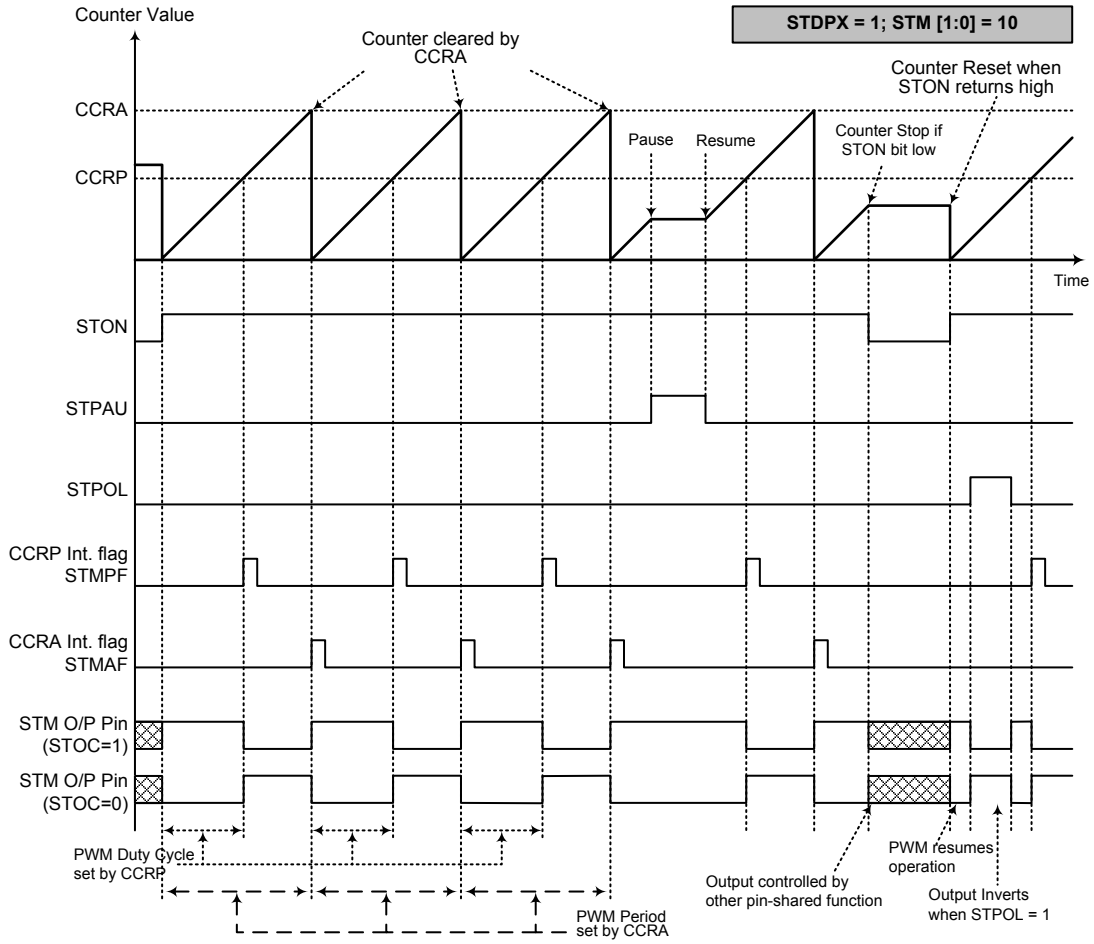
PWM 输出模式 – STDPX=0

注：1. STDPX=0 – 计数器由 CCRP 清除

2. 计数器清除设置 PWM 周期

3. 即使当 STIO[1:0]=00 或 01 时，内部 PWM 功能继续运行

4. STCCLR 对 PWM 操作无影响



PWM 输出模式 – STDPX=1

注：1. STDPX=1 – 计数器由 CCRA 清除

2. 计数器清除设置 PWM 周期

3. 即使当 STIO[1:0]=00 或 01 时，内部 PWM 功能继续运行

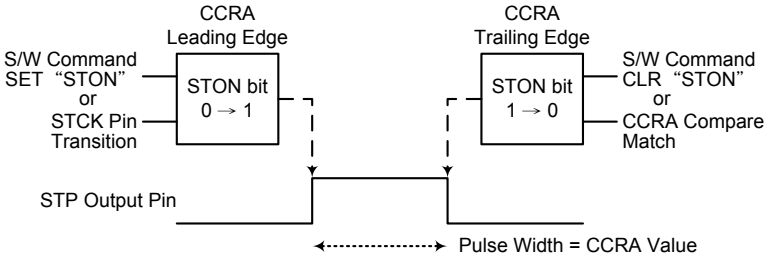
4. STCCLR 对 PWM 操作无影响

单脉冲模式

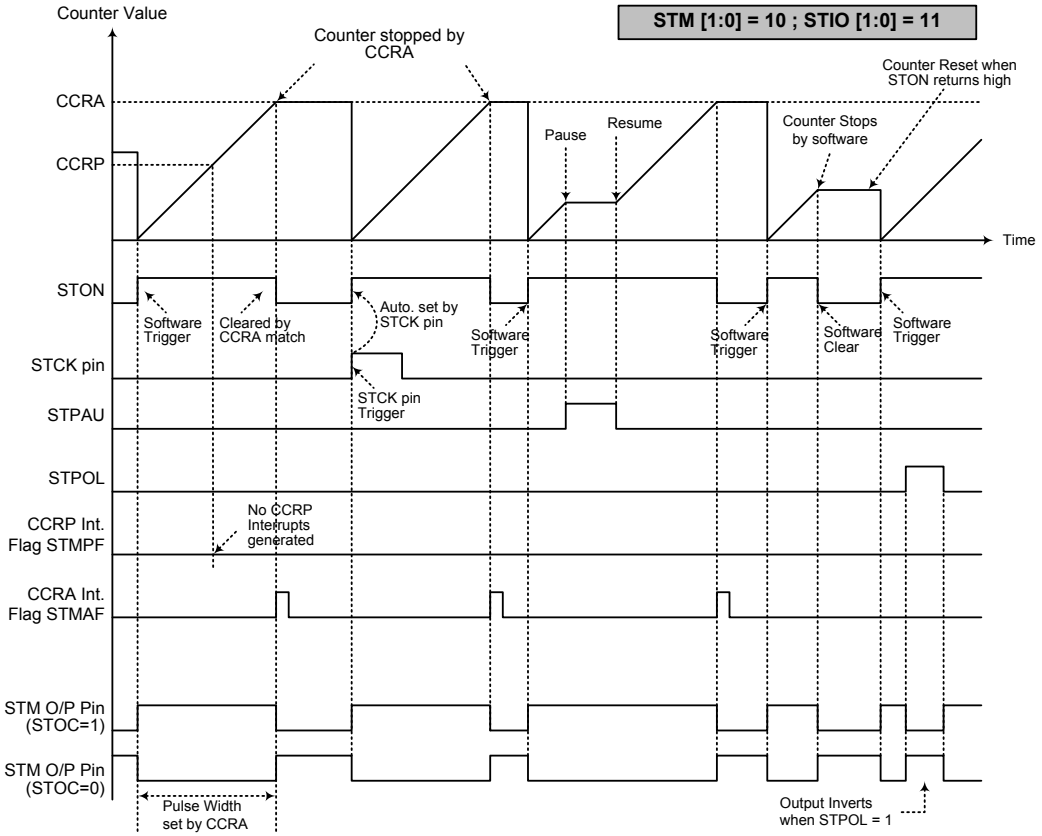
要工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“10”，同时 STIO1 和 STIO0 位需要设置为“11”。正如模式名所言，单脉冲输出模式，在 STM 输出脚将产生一个单脉冲输出。

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

但比较器 A 的比较匹配也会自动清除 STON 位，从而产生单脉冲输出下降沿。此时 CCRA 的值可用于控制脉冲宽度。比较器 A 的比较匹配也能产生一个 STM 中断信号。当计数器重新启动，STON 位从低到高转换时，计数器将被复位回 0。在单脉冲模式下，CCRP 寄存器，STCCLR 位和 STDPX 位未使用。



单脉冲产生示意图



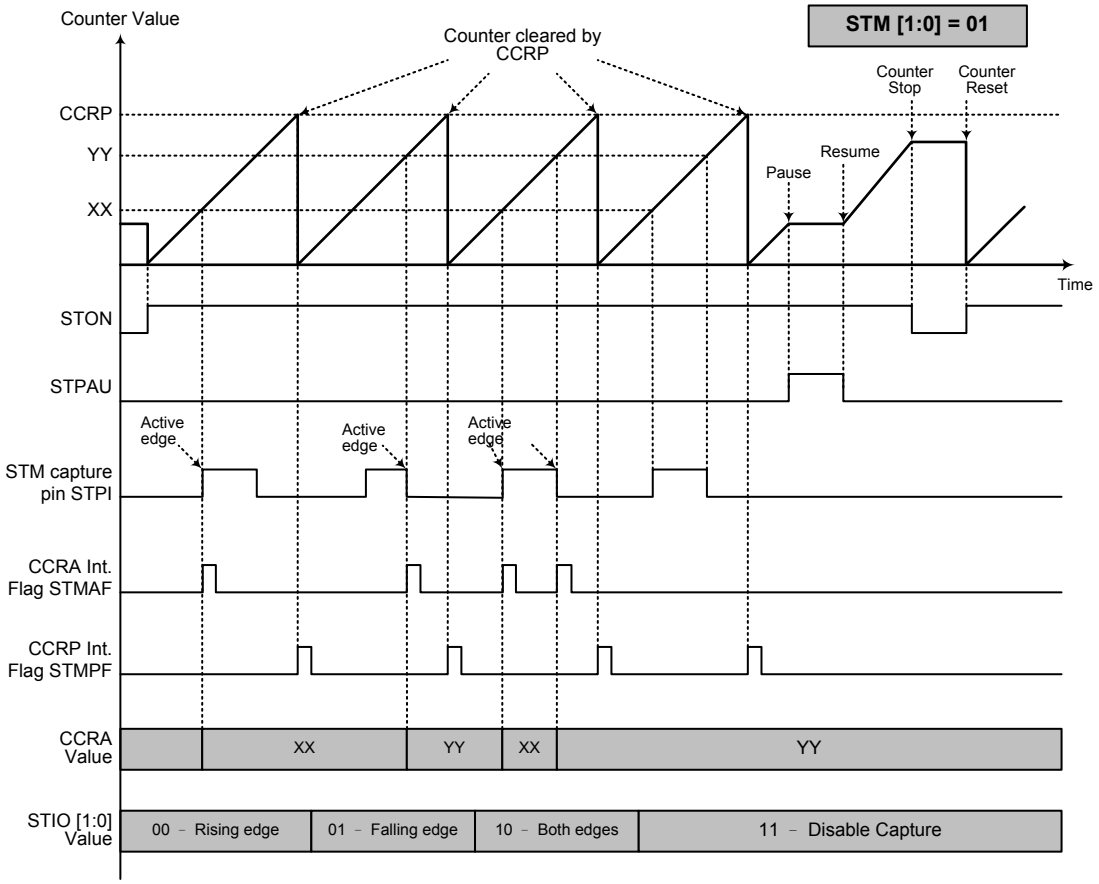
单脉冲模式

注：1. 通过 CCRA 匹配停止计数器
2. CCRP 未使用
3. 通过 STCK 引脚或设置 STON 位为高触发脉冲
4. STCK 引脚有效边沿将自动将 STON 位置高
5. 在单脉冲模式下，STIO[1:0] 必须设为“11”且不能被改变。

捕捉输入模式

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

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

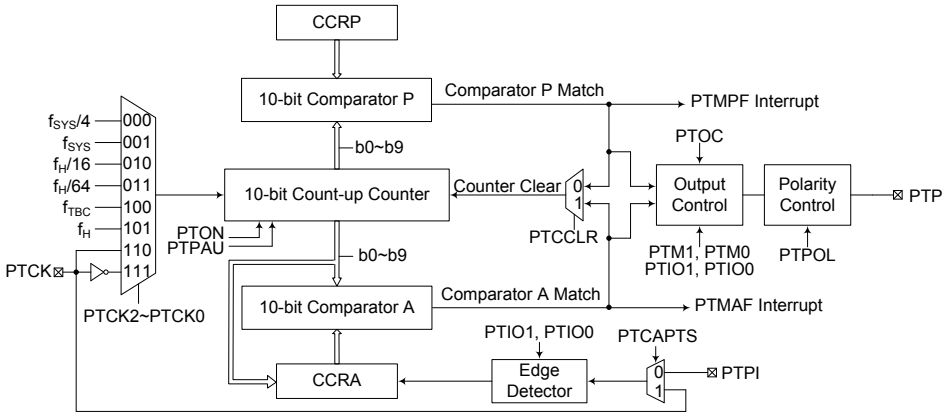


捕捉输入模式

注：1. STM[1:0]=01，有效边沿通过 STIO[1：0] 位段设置
2. TM 捕捉输入引脚有效边沿将计数器的值传到 CCRA 中
3. STCCLR 位未使用
4. 无输出功能 – STOC 和 STPOL 位未使用
5. CCRP 决定计数器的值，且当 CCRP 等于 0 时计数器有一个最大计数值

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一个外部输出脚。



周期型 TM 方框图

周期型 TM 操作

周期型 TM 的核心为 10-bit 向上计数器，由用户选择的内部或外部时钟源驱动。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。

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

周期型 TM 寄存器介绍

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

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

10-bit 周期型 TM 寄存器列表

- **PTMC0 寄存器**

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

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

Bit 6~4 **PTCK2~ PTCK0**：选择 PTM 计数器时钟位
000：$f_{SYS}/4$
001：f_{SYS}
010：$f_H/16$
011：$f_H/64$
100：f_{TBC}
101：f_H
110：PTCK 上升沿时钟
111：PTCK 下降沿时钟
此三位用于选择 PTM 的时钟源。外部引脚时钟源可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 **PTON**：PTM 计数器 On/Off 控制位
0：Off
1：On
此位控制 PTM 的总开关功能。设置此位为高则使能计数器使其运行，清零此位则除能 PTM。清零此位将停止计数器并关闭 PTM 减少耗电。当此位经由低到高转换时，内部计数器将复位清零，当此位由高到低转换时，内部计数器将保持其剩余值直到此位再次转高。
若 PTM 处于比较匹配输出模式、PWM 输出模式或单脉冲输出模式，当 PTON 位经由低到高转换时，PTM 输出脚将复位至 PTOC 位指定的初始值。

Bit 2~0 未定义，读为“0”

- **PTMC1 寄存器**

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

Bit 7~6 **PTM1~ PTM0**：PTM 工作模式选择位
00：比较匹配输出模式
01：捕捉输入模式
10：PWM 模式或单脉冲输出模式
11：定时 / 计数器模式
这两位为 PTM 设置所需工作模式。为确保操作可靠，应在 PTM1 和 PTM0 位发生任何改变之前关闭 PTM。在定时 / 计数器模式中，PTM 输出引脚状态未定义。

Bit 5~4 **PTIO1~ PTIO0**：PTP 或 PTCK 功能选择位

比较匹配输出模式

00：无改变
01：输出低
10：输出高
11：输出翻转

PWM 输出模式 / 单脉冲输出模式

00：PWM 输出无效状态
01：PWM 输出有效状态
10：PWM 输出
11：单脉冲输出

捕捉输入模式

00：在 PTPI 或 PTCK 上升沿输入捕捉
01：在 PTPI 或 PTCK 下降沿输入捕捉
10：在 PTPI 或 PTCK 双沿输入捕捉
11：输入捕捉除能

定时 / 计数器模式

未使用

这两位用于确定在达到一定条件时 PTM 输出脚如何改变状态。此位段所选择的功能取决于 PTM 当下的工作模式。

在比较匹配输出模式中，当比较器 A 发生比较匹配时，PTIO1 ~ PTIO0 决定了 PTM 输出脚的状态应如何改变。当比较器 A 发生比较匹配时，PTM 输出脚可以设置为切换高，切换低或翻转当前状态。当这两位都为零，则输出将不会发生改变。PTM 输出脚的初始值应通过 PTMC1 寄存器中的 PTOC 位进行设置。注意，由 PTIO1 和 PTIO0 位所确定的输出电平必须与利用 PTOC 位所设置的初始值不同，否则在发生比较匹配时 PTM 输出脚上将不发生改变。在 PTM 输出脚改变状态后，可通过改变 PTON 位的电平从低变高将其复位至初始值。

在 PWM 输出模式中，当发生某种比较匹配情况时，PTIO1~ PTIO0 决定了 PTM 输出脚的状态应如何改变。PWM 输出功能通过改变这两位进行更新。在 PTM 关闭后再来改变 PTIO1~ PTIO0 的值是很有必要的。若在 PTM 仍运行时改变 PTIO1~ PTIO0 的值，所得的 PWM 输出是不可预期的。

Bit 3 **PTOC**：PTP 输出控制位

比较匹配输出模式

0：初始低
1：初始高

PWM 模式 / 单脉冲输出模式

0：低有效
1：高有效

此位为 PTM 输出脚的输出控制位。其操作取决于 PTM 被使用在比较匹配输出模式还是 PWM 输出模式 / 单脉冲输出模式。若 PTM 处于定时 / 计数器模式下则不受此位影响。在比较匹配输出模式中，此位决定了在比较匹配发生之前 PTM 输出脚的逻辑电平。在 PWM 模式中，此位决定了 PWM 信号是为高有效还是低有效。

Bit 2 **PTPOL**：PTP 输出极性控制位

0：同相
1：反相

此位控制 PTP 输出引脚的极性。当此位置高，PTM 输出脚将被反相，而此位为零时则为同相。若 PTM 处于定时 / 计数器模式则对其无影响。

Bit 1 **PTCAPTS**：PTM 输入捕捉触发源选择位

0：来自 PTPI
1：来自 PTCK

Bit 0 **PTCCLR**：PTM 计数器清除条件选择位
0：PTM 比较器 P 匹配
1：PTM 比较器 A 匹配
此位用于选择清除计数器的方法。记住周期型 TM 包含两个比较器，即比较器 A 和比较器 P，两者中的任意一个都可以清除内部计数器。当 PTCCLR 位置高，在比较器 A 发生比较匹配时计数器将被清除。当此位为低，计数器将在比较器 P 发生比较匹配或是计数器溢出时被清除。只有在 CCRP 位都被清零时才可执行计数器溢出清除方式。PTCCLR 位在 PWM 输出，单脉冲或输入捕捉模式中未使用。

- **PTMDL 寄存器**

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

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

- **PTMDH 寄存器**

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

Bit 7~2 未定义，读为“0”
Bit 1~0 **PTMDH**：PTM 计数器高字节寄存器 bit 1 ~ bit 0
PTM 10-bit 计数器 bit 9 ~ bit 8

- **PTMAL 寄存器**

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

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

- **PTMAH 寄存器**

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

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

- **PTMRPL 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	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 **PTMRPL**：PTM CCRP 低字节寄存器 bit 7 ~ bit 0
PTM 10-bit CCRP bit 7 ~ bit 0

- **PTMRPH 寄存器**

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

Bit 7~2 未定义，读为“0”

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

周期型 TM 工作模式

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

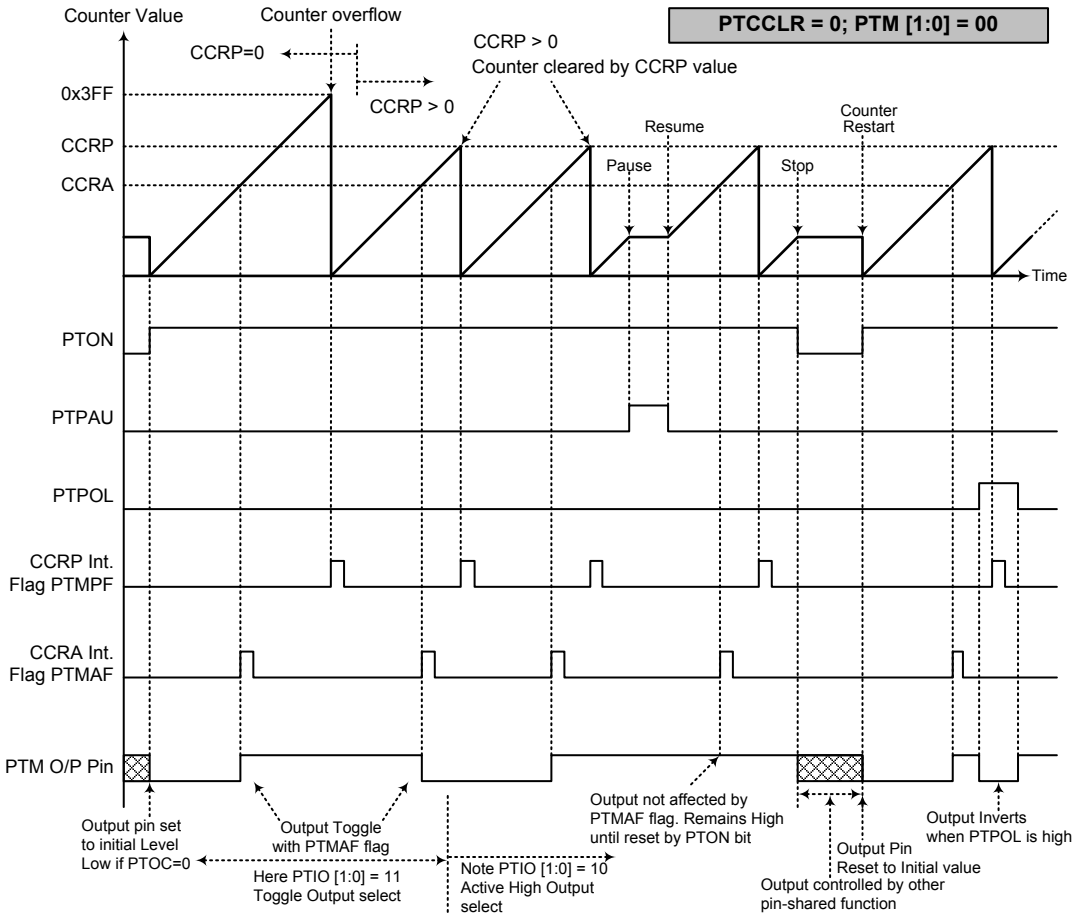
比较匹配输出模式

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

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

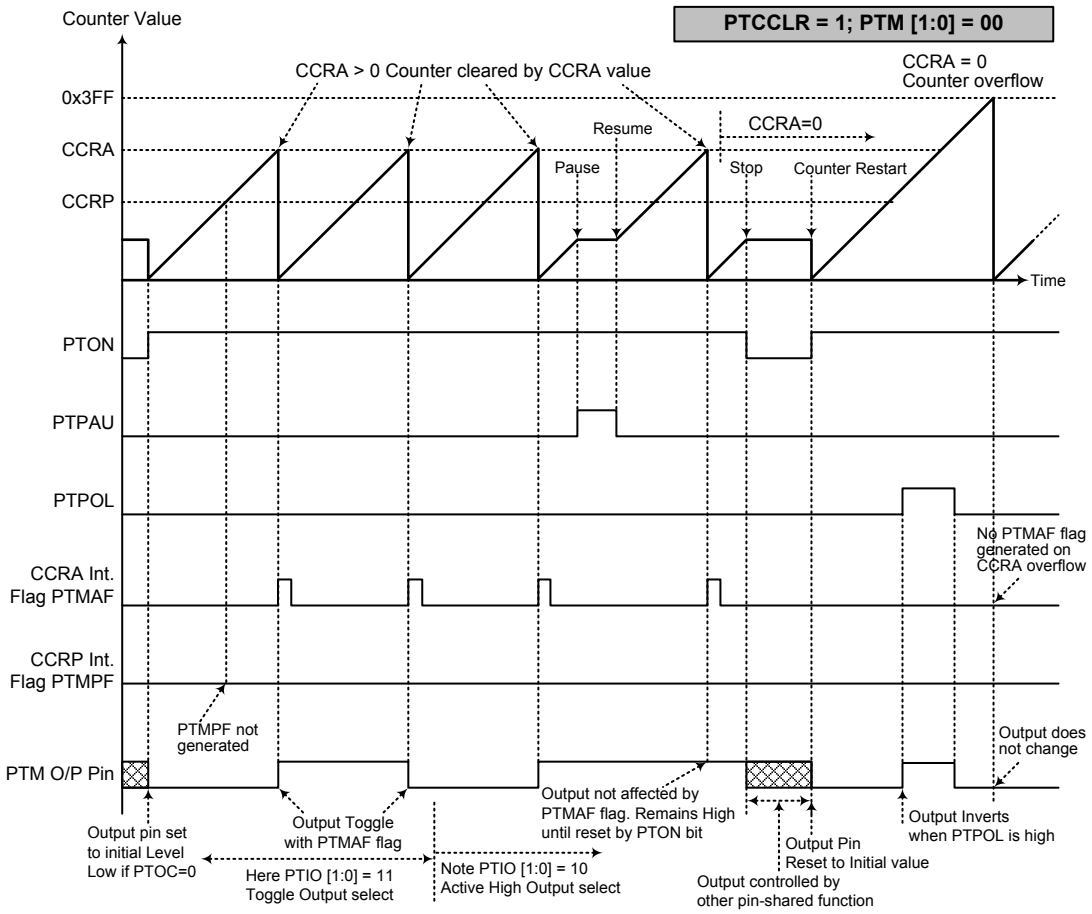
如果 CCRA 被清零，当计数达到最大值 3FFH 时，计数器溢出，而此时不产生 PTMAF 请求标志。

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



比较匹配输出模式 – **PTCCLR=0**

注：1. PTCCLR=0，比较器 P 匹配将清除计数器
2. TM 输出引脚仅由 PTMAF 标志位控制
3. 输出引脚通过 PTON 上升沿复位至初始值



比较匹配输出模式 – PTCCLR =1

注：1. PTCCLR=1，比较器 A 匹配将清除计数器
2. TM 输出引脚仅由 PTMAF 标志位控制
3. 输出引脚通过 PTON 上升沿复位至初始值
4. 当 PTCCLR=1 时不产生 PTMPF 标志位

定时 / 计数器模式

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

PWM 输出模式

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

由于 PWM 波形的周期和占空比可调，其波形的选择就极其灵活。在 PWM 模式中，PTCCLR 位对 PWM 周期无效。CCRA 和 CCRP 寄存器共同产生 PWM 波形，一个用来清除内部计数器从而控制 PWM 波形频率，另一个则用于控制占空比。因此，PWM 波形频率和占空比可由 CCRA 和 CCRP 寄存器中的值来控制。

当比较器 A 或比较器 P 发生比较匹配时，将产生 CCRA 和 CCRP 的中断标志位。PTMC1 寄存器的 PTOC 位用于选择 PWM 波形所需极性，而 PTIO1 和 PTIO0 位用于使能 PWM 输出或将 PTM 输出脚强制固定为高或低电平。PTPOL 位用于取反 PWM 输出波形的极性。

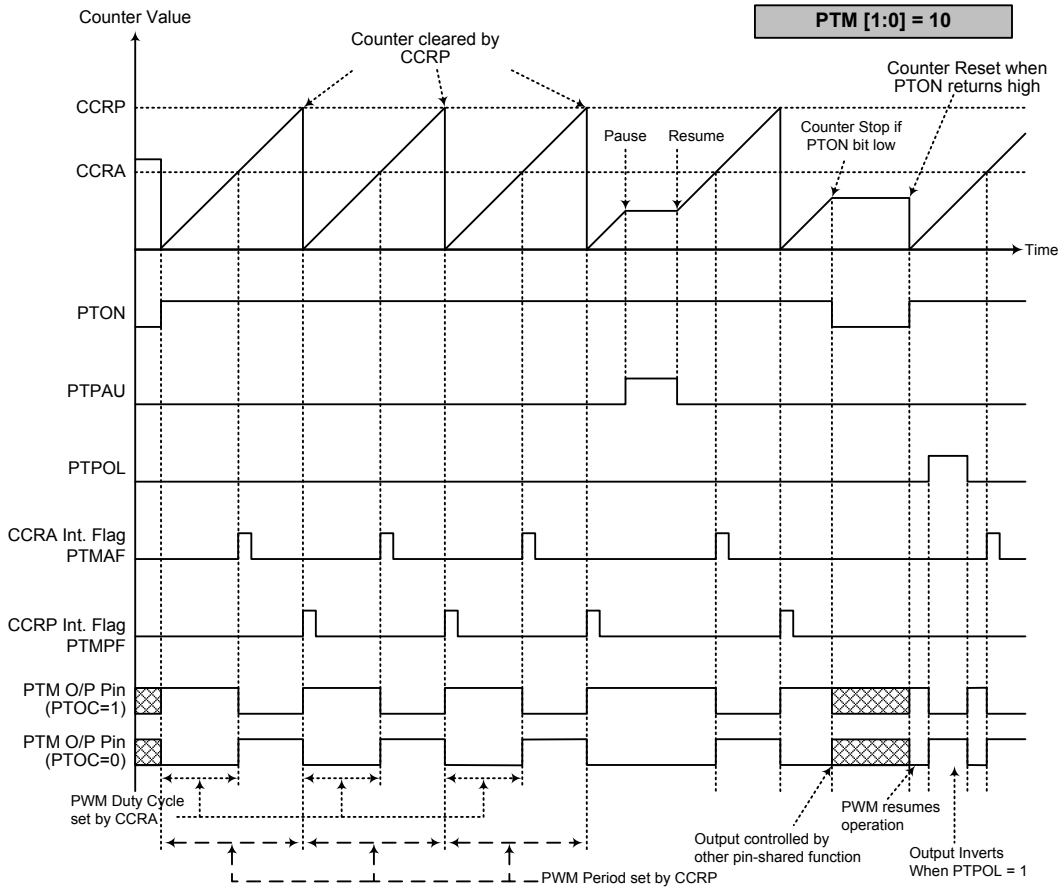
- **10-bit PTM，PWM 模式，边沿对齐模式**

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

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

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

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



PWM 输出模式

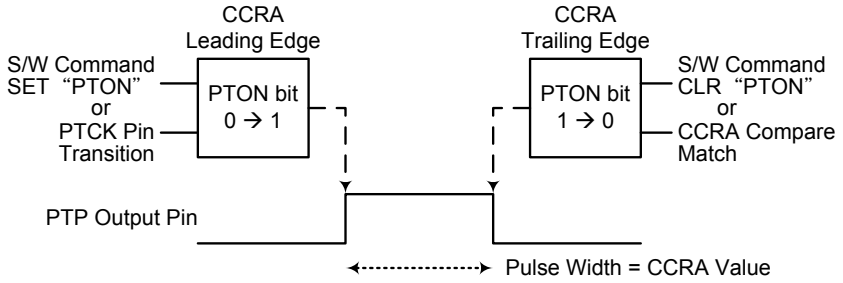
注：1. 计数器由 CCRP 清除
2. 计数器清除设置 PWM 周期
3. 当 PTIO[1:0]=00 或 01，内部 PWM 功能继续运行
4. PTCCLR 位对 PWM 操作无影响

单脉冲模式

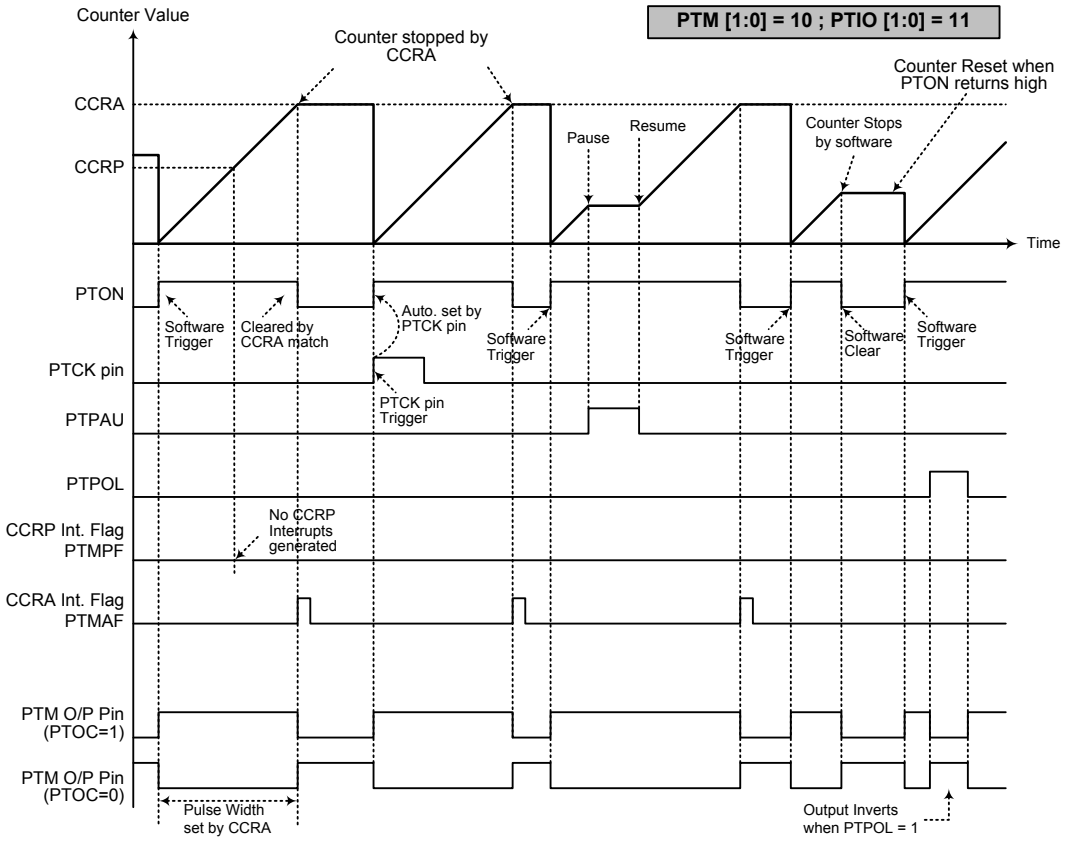
要在此模式，PTMC1 寄存器的 PTM1 和 PTM0 位的值需设置为“10”，且 PTIO1 和 PTIO0 位的值也需设置为“11”。正如模式名所言，单脉冲输出模式，在 PTM 输出脚将产生一个脉冲输出。

通过应用程序控制 PTON 位由低到高的转变来触发脉冲前沿输出。但在单脉冲模式中，PTON 位可利用外部 PTCK 脚而自动由低转变为高，进而启动单脉冲输出。当 PTON 位转变为高电平时，计数器将开始运行，并产生脉冲前沿。当脉冲有效时 PTON 位保持高电平。通过应用程序使 PTON 位清零或比较器 A 比较匹配发生时，产生脉冲下降沿。

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



单脉冲产生示意图



单脉冲模式

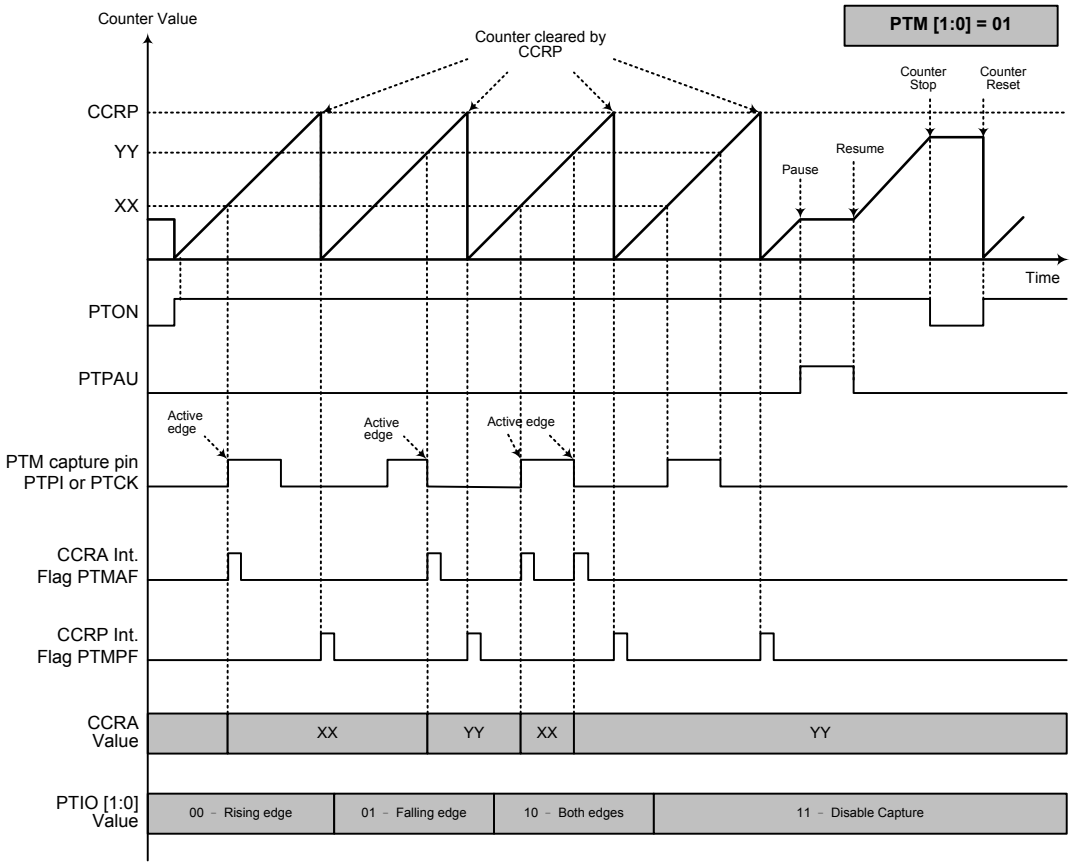
注：1. 通过 CCRA 匹配停止计数器
2. CCRP 未使用
3. 通过 PTCK 引脚或设置 PTON 位为高触发脉冲
4. PTCK 脚有效边沿自动将 PTON 位置高
5. 单脉冲模式下，PTIO[1:0] 必须设为“11”且不能改变

捕捉输入模式

要工作在此模式，PTMC1 寄存器的 PTM1 和 PTM0 位的值需设置为“01”。此模式使能外部信号捕捉并保存内部计数器当前值，因此被用于诸如脉冲宽度测量的应用中。PTPI 和 PTCK 引脚上的外部信号可通过设置 PTMC1 寄存器的 PTCAPTS 位进行选择。输入引脚有效边沿可为上升沿，下降沿或双沿有效。有效边沿转换类型可通过设置 PTMC1 寄存器的 PTIO1 和 PTIO0 位进行选择。通过应用程序将 PTON 位由低到高转变时，计数器启动。

当 PTPI 和 PTCK 引脚出现所需边沿转换时，计数器当前值被锁存到 CCRA 寄存器，并产生 PTM 中断。无论 PTPI 和 PTCK 引脚发生什么事件，计数器继续工作直到 PTON 位发生下降沿跳变。当 CCRP 比较匹配发生时计数器复位至零; CCRP 的值通过这种方式控制计数器的最大值。当比较器 P CCRP 比较匹配发生时，也会产生 PTM 中断。计数 CCRP 溢出中断信号的值可以测量长脉宽。PTIO1 和 PTIO0 位可选择 PTPI 和 PTCK 引脚上的上升沿，下降沿或双沿为有效触发边沿类型。如果 PTIO1 和 PTIO0 位都设为高，则无论 PTPI 和 PTCK 引脚发生什么事件都不会产生捕捉操作，但必须注意，计数器将继续运行。

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



捕捉输入模式

注：1. PTM[1:0]=01，有效边沿通过 PTIO[1：0] 位段设置
2. TM 捕捉输入引脚有效边沿将计数器的值传到 CCRA 中
3. PTCCLR 位未使用
4. 无输出功能 – PTOC 和 PTPOL 位未使用
5. CCRP 决定计数器的值，且当 CCRP 等于 0 时计数器有一个最大计数值

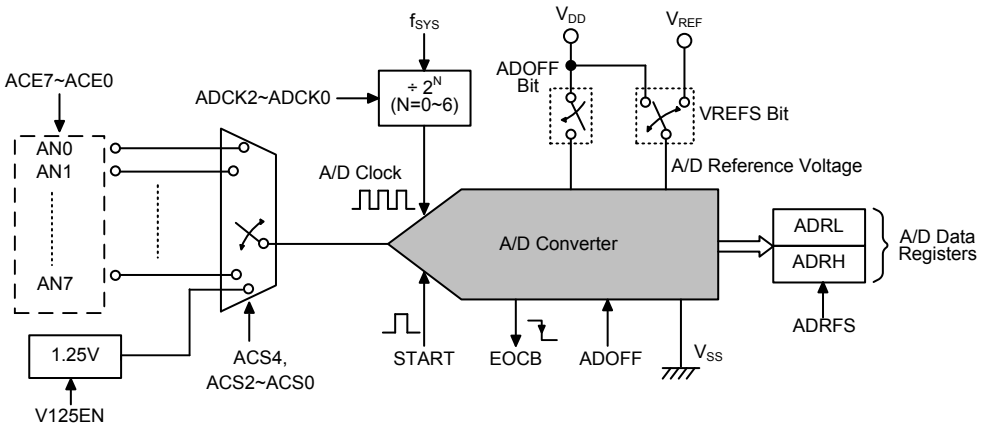
A/D 转换器 – ADC

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

A/D 转换器简介

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

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 转换器寄存器介绍

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 -bit A/D 转换数据的值，剩下的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
ADRL(ADRFS=0)	D3	D2	D1	D0	—	—	—	—
ADRL(ADRFS=1)	D7	D6	D5	D4	D3	D2	D1	D0
ADRH(ADRFS=0)	D11	D10	D9	D8	D7	D6	D5	D4
ADRH(ADRFS=1)	—	—	—	—	D11	D10	D9	D8
ADCR0	START	EOCB	ADOFF	ADRFS	—	ACS2	ACS1	ACS0
ADCR1	ACS4	V125EN	—	VREFS	—	ADCK2	ADCK1	ADCK0
ACERL	ACE7	ACE6	ACE5	ACE4	ACE3	ACE2	ACE1	ACE0

A/D 转换器寄存器列表

A/D 转换器数据寄存器 – ADRL，ADRH

对于具有 12-bit A/D 转换器的单片机，需要两个数据寄存器存放转换结果，一个高字节寄存器 ADRH 和一个低字节寄存器 ADRL。在 A/D 转换完毕后，单片机可以直接读取这些寄存器以获得转换结果。由于寄存器只使用了 16 位中的 12 位，其数据存储格式由 ADCR0 寄存器的 ADRFS 位控制，如下表所示。D0~D11 是 A/D 换转数据结果位。未使用的位读为“0”。

ADRFS	ADRH								ADRL							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	—	—	—	—
1	—	—	—	—	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 – ADCR0，ADCR1，ACERL

控制寄存器 ADCR0、ADCR1 和 ACERL 用于控制 A/D 转换器的功能和操作。这些 8-bit 寄存器定义包括选择连接至内部 A/D 转换器的模拟通道，数字化数据格式，A/D 时钟源，并控制和监测 A/D 转换器的转换结束状态。寄存器 ADCR0 的 ACS2~ACS0 位和 ADCR1 寄存器中的 ACS4 位定义 A/D 转换器外部输入通道编号。由于单片机只包含一个模数转换器电路，每 8 位模拟输入必须按路线发送至转换器。ACS4 和 ACS2~ACS0 位段用于决定模拟信号是来自外部通道输入引脚或内部 1.25V 信号。

ACERL 控制寄存器所包含的 ACE7~ACE0 位段用于确定用作 A/D 转换器模拟输入的引脚。相应的位置高将选中 A/D 转换器输入功能，清零此位则选中 I/O 功能或其他引脚共用功能。当引脚被选择为 A/D 转换器输入，其原始功能除能。此外，这些引脚上所连接的内部上拉电阻将自动移除。

- **ADCR0 寄存器**

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

Bit 7 **START**：启动 A/D 转换位
0 → 1 → 0：启动 A/D 转换
0 → 1：复位 A/D 转换器，EOCB 位置高
此位用于启动 A/D 转换过程。通常此位为低，但如果设为高再被清零，将启动 A/D 转换过程。当此位置高则 A/D 转换器将被复位。

Bit 6 **EOCB**：A/D 转换结束标志位
0：A/D 转换结束
1：A/D 转换中
该只读标志位用于表明 A/D 转换过程是否完成。当转换正在进行，此位置高。

Bit 5 **ADOFF**：A/D 转换器模块 On/Off 控制位
0：A/D 转换器模块开启
1：A/D 转换器模块关闭
此位控制 A/D 内部功能。该位被清零将使能 A/D 转换器。如果该位置高将关闭 A/D 转换器以降低功耗。由于 A/D 转换器在不进行转换时也会消耗一定功率，所以在对功率敏感的电池供电应用中应特别考虑。
注：1. 建议在进入 IDLE/SLEEP 模式之前将 ADOFF 置高以节省功率。
2. ADOFF=1 将关闭 A/D 转换器模块。

Bit 4 **ADRFS**：A/D 转换器数据格式控制位
0：A/D 转换器数据 MSB 为 ADRH bit 7，LSB 为 ADRL bit 4
1：A/D 转换器数据 MSB 为 ADRH bit 3，LSB 为 ADRL bit 0
此位控制存放在两个 A/D 数据寄存器中的 12-bit A/D 转换结果的格式。细节方面请参考 A/D 转换器数据寄存器章节。

Bit 3 未定义，读为“0”

Bit 2 ~ 0 **ACS2 ~ ACS0**：A/D 转换器外部模拟输入通道选择位 (ACS4 为“0”)
000：AN0
001：AN1
010：AN2
011：AN3
100：AN4
101：AN5
110：AN6
111：AN7

- **ADCR1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	ACS4	V125EN	—	VREFS	—	ADCK2	ADCK1	ADCK0
R/W	R/W	R/W	—	R/W	—	R/W	R/W	R/W
POR	0	0	—	0	—	0	0	0

Bit 7 **ACS4**：选择内部 1.25V 作为 A/D 转换器输入控制位
0：除能
1：使能
此位用于使能 1.25V，将其连接至 A/D 转换器。V125EN 位必须预先置位以使能 Bandgap 电路中用于 A/D 转换器的 1.25V 电压。当 ACS4 位置高，Bandgap 1.25V 电压将按线路发送至 A/D 转换器且其他的 A/D 输入通道断开连接。

Bit 6 **V125EN**：内部 1.25V 控制位
0：除能
1：使能
此位用于控制内部 Bandgap 电路是否连接至 A/D 转换器。当此位置高，Bandgap 电压 1.25V 可用于 A/D 转换器。若 1.25V 未用于 A/D 转换器且 LVR 功能除能，则 Bandgap 参考电路将自动关闭以节省功率。当 1.25V 开启并用于 A/D 转换器，在执行 A/D 转换之前需延时 t_{BGS} 的时间使 Bandgap 电路达到稳定状态。

Bit 5 未定义，读为 “0”

Bit 4 **VREFS**：A/D 转换器参考电压选择位
0：内部 A/D 转换器电源
1：VREF 引脚
此位用于为 A/D 转换器选择参考电压。若此位置高，则由外部 VREF 引脚提供 A/D 转换器参考电压。若此位为低，则内部参考由电源供电引脚 VDD 提供。当 A/D 转换器参考电压来自于其他功能共用引脚的 VREF 引脚时，该引脚上的其他功能都将除能。

Bit 3 未定义，读为“0”

Bit 2~0 **ADCK2~ADCK0**：A/D 转换器时钟源选择位
000：f_{SYS}
001：$f_{SYS}/2$
010：$f_{SYS}/4$
011：$f_{SYS}/8$
100：$f_{SYS}/16$
101：$f_{SYS}/32$
110：$f_{SYS}/64$
111：未定义
此位段用于为 A/D 转换器选择时钟源。

- **ACERL 寄存器**

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

Bit 7 **ACE7**：定义 PB3 是否为 A/D 输入
0：不是 A/D 输入
1：A/D 输入，AN7

Bit 6 **ACE6**：定义 PA7 是否为 A/D 输入
0：不是 A/D 输入
1：A/D 输入，AN6

Bit 5 **ACE5**：定义 PA6 是否为 A/D 输入
0：不是 A/D 输入
1：A/D 输入，AN5

Bit 4 **ACE4**：定义 PA5 是否为 A/D 输入
0：不是 A/D 输入
1：A/D 输入，AN4

Bit 3 **ACE3**：定义 PA4 是否为 A/D 输入
0：不是 A/D 输入
1：A/D 输入，AN3

Bit 2 **ACE2**：定义 PB2 是否为 A/D 输入
0：不是 A/D 输入
1：A/D 输入，AN2

Bit 1 **ACE1**：定义 PB1 是否为 A/D 输入
0：不是 A/D 输入
1：A/D 输入，AN1

Bit 0 **ACE0**：定义 PB0 是否为 A/D 输入
0：不是 A/D 输入
1：A/D 输入，AN0

A/D 转换器操作

ADCR0 寄存器中的 START 位用于启动 A/D 转换。当单片机设定此位从逻辑低到逻辑高，然后再到逻辑低，就会开始一个模数转换周期。当 START 位由低转高后未被再次置低，则 ADCR0 寄存器中的 EOCB 位将置高，A/D 转换器将复位。故 START 位用于控制内部 A/D 转换器的启动功能。

ADCR0 寄存器中的 EOCB 位用于表明模数转换过程是否完成。A/D 转换结束后，EOCB 位被自动清零。此外，中断控制寄存器内相应的 A/D 中断请求标志位也会置位，如果中断使能，就会产生对应的内部中断信号。A/D 内部中断信号将引导程序到相应的 A/D 内部中断入口。如果 A/D 内部中断被禁止，可以让单片机轮询 ADCR0 寄存器中的 EOCB 位，检测此位是否被清除，以作为另一种侦测 A/D 转换周期结束的方法。

A/D 转换器时钟源可选择来自系统时钟 f_{SYS} 或 f_{SYS} 的分频频率。分频值由 ADCR1 寄存器中的 ADCK2~ADCK0 位段决定。

虽然 A/D 时钟源是由系统时钟 f_{SYS} 和 ADCK2~ADCK0 位段决定，但可选择的最大 A/D 时钟源还有一些限制。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，ADCK2~ADCK0 位段不能设为“000”或“110”。否则所得的 A/D 时钟周期将小于最小时钟周期或大于最大时钟周期，导致所得 A/D 转换结果不准确。

以下表为参考范例，必须特别注意标了“*”的值，这些值可能会小于所指定的最小 A/D 时钟周期。

f_{SYS}	A/D 时钟周期 (t_{ADCK})							
	ADCK [2：0]= 000 (f_{SYS})	ADCK [2：0]= 001 (f_{SYS}/2)	ADCK [2：0]= 010 (f_{SYS}/4)	ADCK [2：0]= 011 (f_{SYS}/8)	ADCK [2：0]= 100 (f_{SYS}/16)	ADCK [2：0]= 101 (f_{SYS}/32)	ADCK [2：0]= 110 (f_{SYS}/64)	ADCK [2：0]= 111
1MHz	1μs	2μs	4μs	8μs	16μs *	32μs *	64μs *	未定义
2MHz	500ns	1μs	2μs	4μs	8μs	16μs *	32μs *	未定义
4MHz	250ns *	500ns	1μs	2μs	4μs	8μs	16μs *	未定义
8MHz	125ns *	250ns *	500ns	1μs	2μs	4μs	8μs	未定义

A/D 时钟周期范例

ADCR0 寄存器的 ADOFF 位用于控制 A/D 转换电路电源的开启 / 关闭。该位必须清零以开启 A/D 转换器电源。当清零 ADOFF 位以开启 A/D 转换器内部电路时，在 A/D 转换成功初始化前需一段延时。如时序图所示。即使清除寄存器 ACERL 中的 ACE7~ACE0 位段，选择无引脚作为 A/D 输入，如果 ADOFF 设为“0”，那么仍然会产生功耗。因此当未使用 A/D 转换器功能时，在功耗敏感的应用中建议设置 ADOFF 为低以减少功耗。

A/D 转换器参考电压

A/D 转换器的参考电压可以来自正电源引脚 VDD 或来自 VREF 引脚上的外部参考源。通过配置 VREFS 位进行选择。由于 VREF 引脚与其它功能共用，当 VREFS 位置高。VREF 引脚选作参考电压源引脚时，其它引脚功能自动除能。

A/D 转换器输入信号

A/D 转换器的所有模拟输入引脚与 I/O 引脚及其它功能共用。ACERL 寄存器中的 ACE7~ACE0 位决定了输入引脚是设置为 A/D 转换器模拟通道输入还是其它功能。若 ACE7~ACE0 位段的位置高，则相应引脚设置为 A/D 转换器模拟通道输入，该引脚原引脚功能除能。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，端口控制寄存器不需要为使能 A/D 输入而先设定为输入模式，当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。

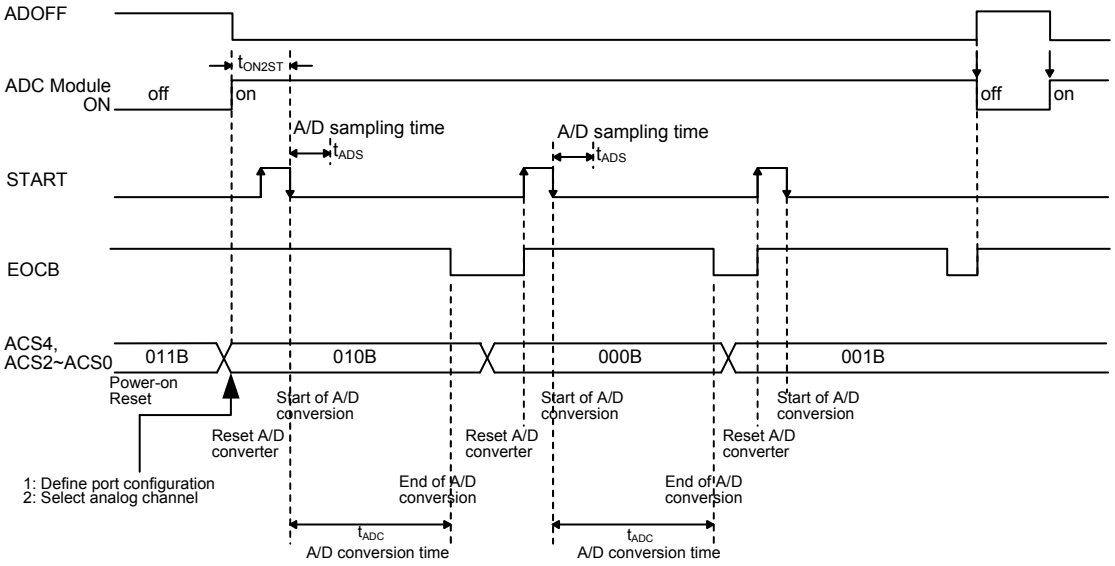
A/D 转换器有其独立的参考电压引脚，即 VREF 引脚，但参考电压也可来自电源供电引脚，可通过 ADCR1 寄存器中的 VREFS 位进行选择。模拟输入值必须小于参考电压值 V_{REF}。

转换率和时序图

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 4 个 A/D 时钟周期，而数据转换需 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

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



A/D 转换时序

A/D 转换步骤

下面概述实现 A/D 转换过程的各个步骤。

- 步骤 1
  通过 ADCR1 寄存器中的 ADCK2~ADCK0 位，选择所需的 A/D 转换器转换时钟。
- 步骤 2
  将 ADCR0 寄存器中的 ADOFF 位清零以使能 A/D 转换器。
- 步骤 3
  通过 ADCR1 寄存器中的 ACS4 位和 ADCR0 寄存器中的 ACS2~ACS0 位段，选择连接至内部 A/D 转换器的信号。
- 步骤 4
  选中作为 A/D 输入的引脚，并正确设置 ACERL 寄存器中的 ACE7~ACE0 位段以配置这些引脚。
- 步骤 5
  若使用中断，则必须正确配置中断控制寄存器以确保 A/D 转换器中断功能有效。主中断控制位 EMI 与 A/D 转换器中断位 ADE 都将置高。
- 步骤 6
  现在可通过设定 ADCR0 寄存器中的 START 位从低到高再到低，启动模拟 / 数字转换进程。应注意此位原始状态应为“0”。
- 步骤 7
  可轮询 ADCR0 寄存器中的 EOCB 位以检测模数转换过程是否完成。当此位变低时表明转换过程结束。此时可从 A/D 数据寄存器 ADRL 和 ADRH 中读取到输出数据。若中断使能且堆栈未满，程序可等待 A/D 中断的发生，这也可作为检测转换是否结束的一种方法。

注：若使用轮询 ADCR0 寄存器中的 EOCB 位的方法以检测模数转换过程是否完成，上述中断使能步骤可以忽略。

编程注意事项

在单片机工作时，如果 A/D 转换器未使用，通过设置 ADCR0 寄存器中的 ADOFF 为高，关闭 A/D 内部电路以减少电源功耗。此时，不考虑输入脚的模拟电压，内部 A/D 转换器电路不产生功耗。如果 A/D 转换器输入脚用作普通 I/O 脚，必须特别注意，输入电压为无效逻辑电平也可能增加功耗。

A/D 转换器控制寄存器的上电复位条件将共用功能的引脚设置为 A/D 转换器输入。若 A/D 转换器的任一输入引脚需要工作在其他功能，则需适当配置 A/D 转换器控制寄存器的位来除能 A/D 输入配置。

A/D 转换功能

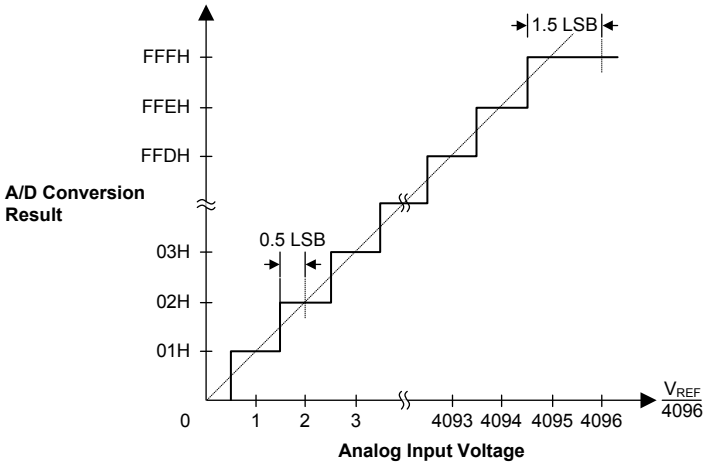
该单片机含有一个 12-bit A/D 转换器，它们转换的最大值可达 FFFH。由于模拟输入最大值等于实际 A/D 转换器参考电压 V_{REF}，因此每一位可表示 $V_{REF}/4096$ 的模拟输入值。

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

通过下面的等式可估算 A/D 转换器输入电压值：

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

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



理想的 A/D 转换功能

A/D 转换程序范例

下面两个范例程序用来说明怎样设置和实现 A/D 转换。第一个范例是轮询 ADCR0 寄存器中的 EOCB 位来判断 A/D 转换何时完成，而第二个范例则使用 A/D 中断的方式判断。

范例 1：使用轮询 EOCB 的方式来检测转换结束

```
clr  ADE                    ; disable A/D converter interrupt
mov  a,03H
mov  ADCR1,a                ; select fSYS/8 as A/D clock and switch off 1.25V
clr  ADOFF
mov  a,0FH                  ; setup ACERL to configure pins AN0~AN3
mov  ACERL,a
mov  a,00H
mov  ADCR0,a                ; enable and connect AN0 channel to the A/D converter
:
start_conversion:
clr  START
set  START                  ; reset A/D
clr  START                  ; start A/D
polling_EOC:
sz   EOCB                   ; poll the ADCR0 register EOCB bit to detect end
                            ; of A/D conversion
jmp  polling_EOC            ; continue polling
mov  a,ADRL                 ; read low byte conversion result value
mov  adrl_buffer,a          ; save result to user defined register
mov  a,ADRH                 ; read high byte conversion result value
mov  adrh_buffer,a          ; save result to user defined register
:
jmp  start_conversion       ; start next A/D conversion
```

注：要关闭 A/D 转换器，必须将 ADOFF 位置高。

范例 2：使用中断的方式来检测转换结束

```
clr ADE                    ; disable A/D converter interrupt
mov a,03H
mov ADCR1,a                ; select fSYS/8 as A/D clock and switch off 1.25V
clr ADOFF
mov a,0FH                  ; setup ACERL to configure pins AN0~AN3
mov ACERL,a
mov a,00H
mov ADCR0,a                ; enable and connect AN0 to A/D convertor
:
start_conversion:
clr START
set START                  ; reset A/D
clr START                  ; start A/D
clr ADF                    ; clear A/D converter interrupt request flag
set ADE                    ; enable A/D converter interrupt
set EMI                    ; enable global interrupt
:
; A/D Converter interrupt service routine ADC_ISR:
mov acc_stack,a            ; save ACC to user defined memory
mov a,STATUS
mov status_stack,a         ; save STATUS to user defined memory
:
mov a,ADRL                 ; read low byte conversion result value
mov adrl_buffer,a          ; save result to user defined register
mov a,ADRH                 ; read high byte conversion result value
mov adrh_buffer,a          ; save result to user defined register
:
EXIT _ISR:
mov a,status_stack
mov STATUS,a               ; restore STATUS from user defined memory
mov a,acc_stack            ; restore ACC from user defined memory
clr ADF                    ; clear A/D converter interrupt flag
reti
```

注：要关闭 A/D 转换器，必须将 ADOFF 位置高。

中断

中断是单片机的一个重要功能。当外部事件或内部功能如定时器模块或 A/D 转换器有效，并且产生中断时，系统会暂时中止当前的程序而转到执行相对应的中断服务程序。该单片机提供多个外部中断和内部中断功能，外部中断由 INTn 引脚动作产生，而内部中断由各种内部功能，如定时器模块、时基和 A/D 转换器等产生。

中断寄存器

中断控制基本上是在一定单片机条件发生时设置请求标志位，应用程序中中断使能位的设置是通过位于特殊功能数据存储器中的一系列寄存器控制的，如下表所示。中断寄存器分为三类。第一类是 INTC0~INTC2 寄存器，用于设置基本的中断；第二类是 MFI0~MFI1 寄存器，用于设置多功能中断；最后一类是 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，紧接着是中断的序号（可选），后面跟着的字母“E”代表使能 / 除能位，“F”代表请求标志位。

功能	使能位	请求标志位	注释
总中断	EMI	—	—
INTn 引脚中断	INTnE	INTnF	n=0 或 1
多功能中断	MFnE	MFnF	n=0~1
A/D 转换器中断	ADE	ADF	—
时基中断	TBnE	TBnF	n=0 或 1
定时器模块	STMPE	STMPF	—
	STMAE	STMAF	—
	PTMPE	PTMPF	—
	PTMAE	PTMAF	—

中断寄存器命名模式

寄存器名称	位							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	MF0F	—	INT0F	MF0E	—	INT0E	EMI
INTC1	TB0F	ADF	—	MF1F	TB0E	ADE	—	MF1E
INTC2	—	—	INT1F	TB1F	—	—	INT1E	TB1E
MFI0	—	—	STMAF	STMPF	—	—	STMAE	STMPE
MFI1	—	—	PTMAF	PTMPF	—	—	PTMAE	PTMPE

中断寄存器列表

- **INTEG 寄存器**

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

Bit 7~4 未定义，读为“0”

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

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

- **INTC0 寄存器**

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

Bit 7 未定义，读为“0”

Bit 6 **MF0F**：多功能中断 0 请求标志位
0：无请求
1：中断请求

Bit 5 未定义，读为“0”

Bit 4 **INT0F**：INT0 中断请求标志位
0：无请求
1：中断请求

Bit 3 **MF0E**：多功能中断 0 控制位
0：除能
1：使能

Bit 2 未定义，读为“0”

Bit 1 **INT0E**：INT0 中断控制位
0：除能
1：使能

Bit 0 **EMI**：总中断控制位
0：除能
1：使能

- **INTC1 寄存器**

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

Bit 7 **TB0F**：时基 0 中断请求标志位
0：无请求
1：中断请求

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

Bit 5 未定义，读为“0”

Bit 4 **MF1F**：多功能中断 1 请求标志位
0：无请求
1：中断请求

Bit 3 **TB0E**：时基 0 中断控制位
0：除能
1：使能

Bit 2 **ADE**：A/D 转换器中断控制位
0：除能
1：使能

Bit 1 未定义，读为“0”

Bit 0 **MF1E**：多功能中断 1 中断控制位
0：除能
1：使能

- **INTC2 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	INT1F	TB1F	—	—	INT1E	TB1E
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 未定义，读为“0”

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

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

Bit 3~2 未定义，读为“0”

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

Bit 0 **TB1E**：时基 1 中断控制位
0：除能
1：使能

- **MFI0 寄存器**

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

Bit 7~6 未定义，读为“0”

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

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

Bit 3~2 未定义，读为“0”

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

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

- **MFI1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PTMAF	PTMPF	—	—	PTMAE	PTMPE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 未定义，读为“0”

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

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

Bit 3~2 未定义，读为“0”

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

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

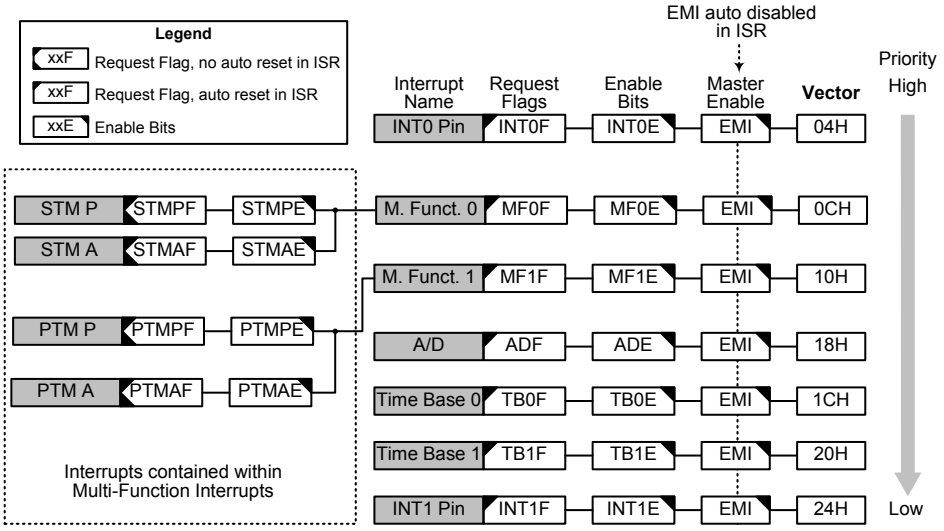
中断操作

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

当中断发生时，PC 中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。系统将从此向量取得下条指令。中断向量处通常为“JMP”指令，以跳转到相应的中断服务程序，此处程序存放着相应中断的代码。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 INTn 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，INTn 引脚的状态发生变化，外部中断请求标志 INTnF 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 和相应外部中断使能位 INTnE 需先被置位。此外，必须使用 INTEG 寄存器使能外部中断功能并选择触发沿类型。外部中断引脚和普通 I/O 口共用，如果相应寄存器中的中断使能位被置位，并且通过引脚共用寄存器选择外部中断脚，此引脚将被作为外部中断脚使用。此时必须通过设置端口控制寄存器和相关引脚共用选择位，将该引脚设置为输入口。当中断使能，堆栈未满并且外部中断脚状态改变，将调用外部中断向量子程序。当响应外部中断服务子程序时，中断请求标志位 INTnF 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 被用于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。注意寄存器 INTEG 也可以用来除能外部中断功能。

多功能中断

此单片机中有多达 2 种多功能中断，与其它中断不同，这些没有独立源，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。当所包含的任一功能产生中断请求标志，多功能中断标志将置位。若要跳转到相应的中断向量地址，当多功能中断使能，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 转换动作的结束控制 A/D 转换器中断。当 A/D 转换器中断请求标志 ADF 被置位，即 A/D 转换过程完成时，中断请求发生。若要跳转到相应中断向量地址，总中断控制位 EMI、A/D 转换器中断使能位 ADE 需先被置位。当中断使能，堆栈未满且 A/D 转换动作结束时，将调用 A/D 转换器中断向量子程序。当 A/D 转换器中断响应时，ADF 标志将自动清除，EMI 将被自动清零以除能其它中断。

时基中断

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

时基中断的目的是提供一个固定周期的中断信号。其时钟源来自内部时钟源 f_{TB}。f_{TB} 输入时钟首先经过分频器，分频率由程序设置 TBC 寄存器相关位获取合适的分频值以提供更长的时基中断周期。相应的控制时基中断周期的时钟源可为多个不同的源，如系统工作模式章节所述。

- **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

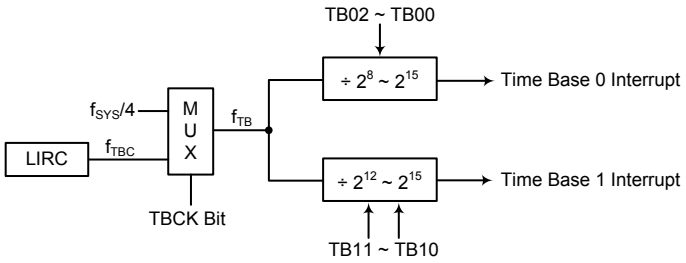
Bit 7 **TBON**：TB0 和 TB1 控制位
0：除能
1：使能

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

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

Bit 3 未定义，读为“0”

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



时基中断

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。标准型和周期型 TM 都有两个中断请求标志位 xTMPF 和 xTMAF 及两个使能位 xTMPE 和 xTMAE。当 TM 比较器 P、A 匹配情况发生时，相应 TM 中断请求标志被置位，TM 中断请求产生。

若要程序跳转到相应中断向量地址，总中断控制位 EMI、相应 TM 中断使能位和相关多功能中断使能位 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、低电压都可能导致其相应的中断标志被置位，由此产生中断，因此必须注意避免伪唤醒情况的发生。若中断唤醒功能被除能，单片机进入休眠或空闲模式前相应中断请求标志应被置起。中断唤醒功能不受中断使能位的影响。

编程注意事项

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

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

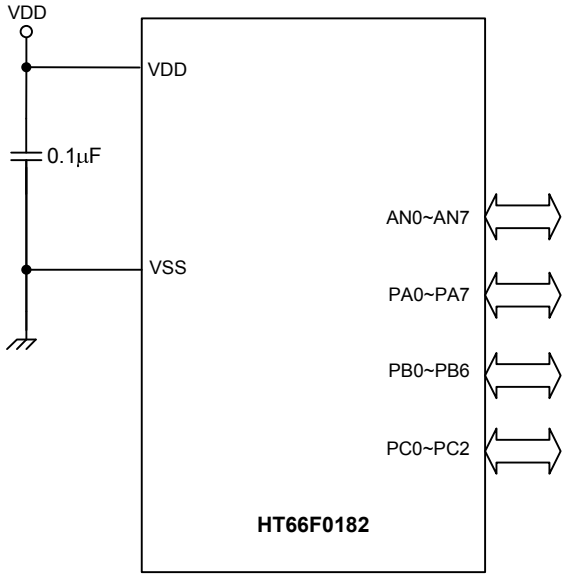
建议在中断服务子程序中不要使用“CALL 子程序”指令。中断通常发生在不可预料的情况或是需要立刻执行的某些应用。假如只剩下一层堆栈且没有控制好中断，当“CALL 子程序”在中断服务子程序中执行时，将破坏原来的控制序列。

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

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

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

应用电路



指令集

简介

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

为了更加容易理解各种各样的指令码，接下来按功能分组介绍它们。

指令周期

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

数据的传送

单片机程序中数据传送是使用最为频繁的操作之一，使用三种 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、进位标志的反相减，结果放入 ACC	1	Z, C, AC, OV
SBCM A,[m]	ACC 与数据存储器、进位标志相减，结果放入数据存储器	1 注	Z, C, AC, OV
DAA [m]	将加法运算中放入 ACC 的值调整为十进制数，并将结果放入数据存储器	1 注	C
逻辑运算			
AND A,[m]	ACC 与数据存储器做“与”运算，结果放入 ACC	1	Z
OR A,[m]	ACC 与数据存储器做“或”运算，结果放入 ACC	1	Z
XOR A,[m]	ACC 与数据存储器做“异或”运算，结果放入 ACC	1	Z
ANDM A,[m]	ACC 与数据存储器做“与”运算，结果放入数据存储器	1 注	Z
ORM A,[m]	ACC 与数据存储器做“或”运算，结果放入数据存储器	1 注	Z
XORM A,[m]	ACC 与数据存储器做“异或”运算，结果放入数据存储器	1 注	Z
AND A, x	ACC 与立即数做“与”运算，结果放入 ACC	1	Z
OR A, x	ACC 与立即数做“或”运算，结果放入 ACC	1	Z
XOR A, x	ACC 与立即数做“异或”运算，结果放入 ACC	1	Z
CPL [m]	对数据存储器取反，结果放入数据存储器	1 注	Z
CPLA [m]	对数据存储器取反，结果放入 ACC	1	Z
递增和递减			
INCA [m]	递增数据存储器，结果放入 ACC	1	Z
INC [m]	递增数据存储器，结果放入数据存储器	1 注	Z
DECA [m]	递减数据存储器，结果放入 ACC	1	Z
DEC [m]	递减数据存储器，结果放入数据存储器	1 注	Z

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

助记符	说明	指令周期	影响标志位
CLR WDT1	预清除看门狗定时器	1	TO, PDF
CLR WDT2	预清除看门狗定时器	1	TO, PDF
SWAP [m]	交换数据存储器的高低字节，结果放入数据存储器	1 注	无
SWAPA [m]	交换数据存储器的高低字节，结果放入 ACC	1	无
HALT	进入暂停模式	1	TO, PDF

注：1. 对跳转指令而言，如果比较的结果牵涉到跳转即需 2 个周期，如果没有发生跳转，则只需一个周期。
2. 任何指令若要改变 PCL 的内容将需要 2 个周期来执行。
3. 对于“CLR WDT1”或“CLR WDT2”指令而言，TO 和 PDF 标志位也许会受执行结果影响，“CLR WDT1”和“CLR WDT2”被连续地执行后，TO 和 PDF 标志位会被清除，否则 TO 和 PDF 标志位保持不变

指令定义

ADC A，[m] Add Data Memory to ACC with Carry
指令说明 将指定的数据存储器、累加器内容以及进位标志相加，结果存放到累加器。
功能表示 ACC ← ACC + [m] + C
影响标志位 OV、Z、AC、C

ADCM A，[m] Add ACC to Data Memory with Carry
指令说明 将指定的数据存储器、累加器内容和进位标志位相加，结果存放到指定的数据存储器。
功能表示 [m] ← ACC + [m] + C
影响标志位 OV、Z、AC、C

ADD A，[m] Add Data Memory to ACC
指令说明 将指定的数据存储器和累加器内容相加，结果存放到累加器。
功能表示 ACC ← ACC + [m]
影响标志位 OV、Z、AC、C

ADD A，x Add immediate data to ACC
指令说明 将累加器和立即数相加，结果存放到累加器。
功能表示 ACC ← ACC + x
影响标志位 OV、Z、AC、C

ADDM A，[m] Add ACC to Data Memory
指令说明 将指定的数据存储器和累加器内容相加，结果存放到指定的数据存储器。
功能表示 [m] ← ACC + [m]
影响标志位 OV、Z、AC、C

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

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

ANDM A，[m]	Logical AND ACC to Data Memory
指令说明	将指定数据存储器内容和累加器中的数据做逻辑与， 结果存放到数据存储器。
功能表示	[m] ← ACC“AND”[m]
影响标志位	Z

CALL addr	Subroutine call
指令说明	无条件地调用指定地址的子程序，此时程序计数器先加 1 获得下一个要执行的指令地址并压入堆栈，接着载入指定 地址并从新地址继续执行程序，由于此指令需要额外的运 算，所以为一个 2 周期的指令。
功能表示	Stack ← Program Counter + 1 Program Counter ← addr
影响标志位	无

CLR [m]	Clear Data Memory
指令说明	将指定数据存储器的内容清零。
功能表示	[m] ← 00H
影响标志位	无

CLR [m].i	Clear bit of Data Memory
指令说明	将指定数据存储器的 i 位内容清零。
功能表示	[m].i ← 0
影响标志位	无

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

CLR WDT1 Preclear Watchdog Timer

指令说明 PDF 和 TO 标志位都被清 0。必须配合 CLR WDT2 一起使用清除 WDT 计时器。当程序仅执行 CLR WDT1，而没有执行 CLR WDT2 时，PDF 与 TO 保留原状态不变。

功能表示 WDT ← 00H
TO & PDF ← 0

影响标志位 TO、PDF

CLR WDT2 Preclear Watchdog Timer

指令说明 PDF 和 TO 标志位都被清 0。必须配合 CLR WDT1 一起使用清除 WDT 计时器。当程序仅执行 CLR WDT2，而没有执行 CLR WDT1 时，PDF 与 TO 保留原状态不变。

功能表示 WDT ← 00H
TO & PDF ← 0

影响标志位 TO、PDF

CPL [m] Complement Data Memory

指令说明 将指定数据存储器中的每一位取逻辑反，
相当于从 1 变 0 或 0 变 1。

功能表示 [m] ← $[\overline{m}]$

影响标志位 Z

CPLA [m] Complement Data Memory with result in ACC

指令说明 将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或 0 变 1，而结果被储存回累加器且数据存储器中的内容不变。

功能表示 ACC ← $[\overline{m}]$

影响标志位 Z

DAA [m] Decimal-Adjust ACC for addition with result in Data Memory

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

功能表示 [m] ← ACC + 00H 或
[m] ← ACC + 06H 或
[m] ← ACC + 60H 或
[m] ← ACC + 66H

影响标志位 C

DEC [m] Decrement Data Memory

指令说明 将指定数据存储器内容减 1。

功能表示 [m] ← [m] – 1

影响标志位 Z

DECA [m] Decrement Data Memory with result in ACC

指令说明 将指定数据存储器的内容减 1，把结果存放回累加器并保持指定数据存储器的内容不变。

功能表示 ACC ← [m] – 1

影响标志位 Z

HALT Enter power down mode

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

功能表示 TO ← 0
PDF ← 1

影响标志位 TO、PDF

INC [m] Increment Data Memory

指令说明 将指定数据存储器的内容加 1。

功能表示 [m] ← [m] + 1

影响标志位 Z

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

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

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

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

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

NOP No operation
指令说明 空操作，接下来顺序执行下一条指令。
功能表示 PC ← PC+1
影响标志位 无

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

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

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

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

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

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

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

RLA [m] Rotate Data Memory left with result in ACC

指令说明 将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位，结果送到累加器，而指定数据存储器的内容保持不变。

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

影响标志位 无

RLC [m] Rotate Data Memory Left through Carry

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

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

影响标志位 C

RLC A [m] Rotate Data Memory left through Carry with result in ACC

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

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

影响标志位 C

RR [m] Rotate Data Memory right

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

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

影响标志位 无

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

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

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

影响标志位 无

RRC [m] Rotate Data Memory right through Carry

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

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

影响标志位 C

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

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

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

影响标志位 C

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

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

功能表示 ACC ← ACC – [m] – $\overline{C}$

影响标志位 OV、Z、AC、C、SC、CZ

SBCM A，[m] Subtract Data Memory from ACC with Carry and result in Data Memory

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

功能表示 [m] ← ACC – [m] – $\overline{C}$

影响标志位 OV、Z、AC、C、SC、CZ

SDZ [m] Skip if Decrement Data Memory is 0

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

功能表示 [m] ← [m] -1，如果 [m]=0 跳过下一条指令执行

影响标志位 无

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

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

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

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

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

SNZ [m].i Skip if bit i of Data Memory is not 0
指令说明 判断指定数据存储器的第 i 位，若不为 0，则程序跳过下一条指令执行。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果为 0，则程序继续执行下一条指令。
功能表示 如果 [m].i≠0，跳过下一条指令执行
影响标志位 无

SUB A，[m] Subtract Data Memory from ACC
指令说明 将累加器的内容减去指定的数据存储器的数据，把结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示 ACC ← ACC – [m]
影响标志位 OV、Z、AC、C、SC、CZ

SUBM A，[m] Subtract Data Memory from ACC with result in Data Memory
指令说明 将累加器的内容减去指定数据存储器的数据，结果存放到指定的数据存储器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示 [m] ← ACC – [m]
影响标志位 OV、Z、AC、C、SC、CZ

SUB A，x Subtract immediate Data from ACC
指令说明 将累加器的内容减去立即数，结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示 ACC ← ACC – x
影响标志位 OV、Z、AC、C、SC、CZ

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

SWAPA [m] Swap nibbles of Data Memory with result in ACC
指令说明 将指定数据存储器的低 4 位与高 4 位互相交换，再将结果存放到累加器且指定数据寄存器的数据保持不变。
功能表示 ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0
影响标志位 无

SZ [m] Skip if Data Memory is 0

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

功能表示 如果 [m]=0, 跳过下一条指令执行

影响标志位 无

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

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

功能表示 ACC ← [m]，如果 [m]=0，跳过下一条指令执行

影响标志位 无

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

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

功能表示 如果 [m].i=0，跳过下一条指令执行

影响标志位 无

TABRD [m] Read table (specific page) to TBLH and Data Memory

指令说明 将表格指针对 TBHP 和 TBLP 所指的程序代码低字节 (指定页) 移至指定数据存储器且将高字节移至 TBLH。

功能表示 [m] ← 程序代码 (低字节)
TBLH ← 程序代码 (高字节)

影响标志位 无

TABRDC [m] Read table (current page) to TBLH and Data Memory

指令说明 将表格指针 TBLP 所指的程序代码低字节（当前页）移至指定的数据存储器且将高字节移至 TBLH。

功能表示 [m] ← 程序代码 (低字节)
TBLH ← 程序代码 (高字节)

影响标志位 无

TABRDL [m]	Read table (last page) to TBLH and Data Memory
指令说明	将表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无

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

XORM A，[m]	Logical XOR ACC to Data Memory
指令说明	将累加器的数据和指定的数据存储器内容逻辑异或，结果放到数据存储器。
功能表示	[m] ← ACC“XOR”[m]
影响标志位	Z

XOR A，x	Logical XOR immediate data to ACC
指令说明	将累加器的数据与立即数逻辑异或，结果存放到累加器。
功能表示	ACC ← ACC“XOR”x
影响标志位	Z

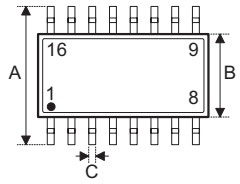
封装信息

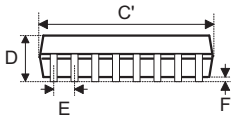
请注意，这里提供的封装信息仅作为参考。由于这个信息经常更新，提醒用户咨询 Holtek 网站以获取最新版本的封装信息。

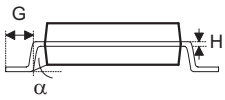
封装信息的相关内容如下所示，点击可链接至 Holtek 网站相关信息页面。

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

16-pin NSOP (150mil) 外形尺寸



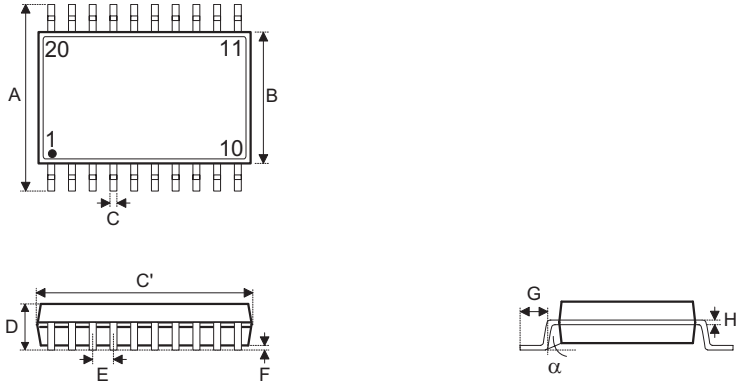




符号	尺寸 (单位：inch)		
	最小	正常	最大
A	—	0.236 BSC	—
B	—	0.154 BSC	—
C	0.012	—	0.020
C’	—	0.390 BSC	—
D	—	—	0.069
E	—	0.050 BSC	—
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

符号	尺寸 (单位：mm)		
	最小	正常	最大
A	—	6 BSC	—
B	—	3.9 BSC	—
C	0.31	—	0.51
C’	—	9.9 BSC	—
D	—	—	1.75
E	—	1.27 BSC	—
F	0.10	—	0.25
G	0.40	—	1.27
H	0.10	—	0.25
α	0°	—	8°

20-pin NSOP (150mil) 外形尺寸



符号	尺寸 (单位：inch)		
	最小	正常	最大
A	0.228	0.236	0.244
B	0.146	0.154	0.161
C	0.009	—	0.012
C’	0.382	0.390	0.398
D	—	—	0.069
E	—	0.032 BSC	—
F	0.002	—	0.009
G	0.020	—	0.031
H	0.008	—	0.010
α	0°	—	8°

符号	尺寸 (单位：mm)		
	最小	正常	最大
A	5.80	6.00	6.20
B	3.70	3.90	4.10
C	0.23	—	0.30
C’	9.70	9.90	10.10
D	—	—	1.75
E	—	0.8 BSC	—
F	0.05	—	0.23
G	0.50	—	0.80
H	0.21	—	0.25
α	0°	—	8°