



内置 EEPROM 增强型 Flash 单片机

HT66F016/HT66F017

HT68F016/HT68F017

HT66F016R/HT66F017R

HT68F016R/HT68F017R

版本：V1.60 日期：2019-11-13

www.holtek.com

目录

特性	6
CPU 特性	6
周边特性	6
概述	7
选型表	7
方框图	8
引脚图	9
引脚说明	10
极限参数	12
直流电气特性	12
交流电气特性	14
ADC 特性.....	15
比较器电气特性	15
上电复位特性	16
BandGap 电压参考 (V_{BG}) 特性曲线	16
系统结构	17
时序和流水线结构	17
程序计数器	18
堆栈	18
算术逻辑单元 – ALU	19
Flash 程序存储器	19
结构	19
特殊向量	20
查表	20
查表范例	20
在线编程	21
数据存储器	22
结构	22
特殊功能寄存器	25
间接寻址寄存器 – IAR0, IAR1	25
间接寻址指针 – MP0, MP1.....	25
存储区指针 – BP	26
累加器 – ACC	26
程序计数器低字节寄存器 – PCL.....	26
表格寄存器 – TBLP, TBHP, TBLH.....	26
状态寄存器 – STATUS.....	27
EEPROM 数据寄存器	29
EEPROM 数据寄存器结构	29

EEPROM 寄存器	29
从 EEPROM 中读取数据	31
写数据到 EEPROM	31
写保护	31
EEPROM 中断	31
编程注意事项	32
振荡器	33
振荡器概述	33
系统时钟配置	33
外部晶体 / 陶瓷振荡器 – HXT	34
内部 RC 振荡器 – HIRC	35
内部 32kHz 振荡器 – LIRC	35
辅助振荡器	35
工作模式和系统时钟	36
系统时钟	36
系统工作模式	38
控制寄存器	39
快速唤醒	40
工作模式切换和唤醒	41
静态电流的注意事项	45
唤醒	45
编程注意事项	45
看门狗定时器	46
看门狗定时器时钟源	46
看门狗定时器控制寄存器	46
看门狗定时器操作	47
看门狗定时器使能 / 除能控制	48
复位和初始化	48
复位功能	48
复位初始状态	51
输入 / 输出端口	54
上拉电阻	54
PA 口唤醒	55
输入 / 输出端口控制寄存器	55
引脚重置功能	56
引脚重置寄存器	56
输入 / 输出引脚结构	57
编程注意事项	58
定时器模块 – TM	58
简介	58
TM 操作	59
TM 时钟源	59
TM 中断	59

TM 外部引脚	59
TM 输入 / 输出引脚控制寄存器	60
编程注意事项	61
简易型 – CTM	62
简易型 TM 操作	62
简易型 TM 寄存器介绍	63
简易型 TM 工作模式	67
标准型 TM – STM	73
标准型 TM 操作	73
标准型 TM 寄存器介绍	73
标准型 TM 工作模式	77
A/D 转换器.....	87
A/D 简介	87
A/D 转换寄存器介绍	87
A/D 操作	90
A/D 输入引脚	91
A/D 转换步骤	92
编程注意事项	93
A/D 转换功能	93
A/D 转换应用范例	94
比较器	96
比较器操作	96
比较器寄存器	96
比较器中断	97
编程注意事项	97
中断	98
中断寄存器	98
中断操作	104
外部中断	105
比较器中断	105
多功能中断	106
A/D 转换器中断	106
时基中断	106
EEPROM 中断	107
LVD 中断	108
TM 中断	108
中断唤醒功能	108
编程注意事项	108
低电压检测 – LVD	109
LVD 寄存器	109
LVD 操作	109
配置选项	110
应用电路	110

指令集 111

 简介 111

 指令周期 111

 数据的传送 111

 算术运算 111

 逻辑和移位运算 111

 分支和控制转换 112

 位运算 112

 查表运算 112

 其它运算 112

指令集概要 113

 惯例 113

指令定义 116

封装信息 128

 16-pin NSOP (150mil) 外形尺寸 129

 16-pin SSOP (150mil) 外形尺寸 130

特性

CPU 特性

- 工作电压：
 - ◆ $f_{\text{SYS}}=8\text{MHz}$: 2.2V~5.5V
 - ◆ $f_{\text{SYS}}=12\text{MHz}$: 2.7V~5.5V
 - ◆ $f_{\text{SYS}}=16\text{MHz}$: 3.3V~5.5V
 - ◆ $f_{\text{SYS}}=20\text{MHz}$: 4.5V~5.5V
- $V_{\text{DD}}=5\text{V}$, 系统时钟为 20MHz 时, 指令周期为 $0.2\mu\text{s}$
- 提供暂停和唤醒功能, 以降低功耗
- 三种振荡模式:
 - ◆ 外部晶振 – HXT
 - ◆ 内部 RC – HIRC
 - ◆ 内部 32kHz RC – LIRC
- 多种工作模式: 正常、低速、空闲和休眠
- 内部集成 8MHz 振荡器, 无需外接元件
- 所有指令都可在 1 或 2 个指令周期内完成
- 查表指令
- 63 条指令
- 最多可达 8 层堆栈
- 位操作指令

周边特性

- Flash 程序存储: 最大可达 $2\text{K}\times 16$
- RAM 数据存储: 最大可达 128×8
- True EEPROM 存储器: 64×8
- 看门狗定时器功能
- 14 个双向 I/O 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
- 比较器功能
- 双时基功能, 可提供固定时间的中断信号
- 低电压复位功能
- 低电压检测功能
- 4 通道 12 位分辨精度的 A/D 转换器
- 封装类型: 16-pin NSOP/SSOP
- Flash 程序存储器烧录可达 10,000 次
- Flash 程序存储器数据可保存 10 年以上
- True EEPROM 程序存储器烧录可达 100,000 次
- True EEPROM 程序存储器数据可保存 10 年以上

概述

HT6xF01x 系列单片机是具有 8 位高性能精简指令集的 Flash 单片机。该系列单片机具有一系列功能和特性，其 Flash 存储器可多次编程的特性给用户提供了较大的方便。存储器方面，还包含了一个 RAM 数据存储器和一个可用于存储序号、校准数据等非易失性数据的 True EEPROM 存储器。

在模拟特性方面，此系列单片机包含都一个多通道 12 位 A/D 转换器和比较器功能。以及多个使用灵活的定时器模块，可提供定时功能、脉冲产生功能及 PWM 产生功能。内部看门狗定时器、低电压复位和低电压检测等内部保护特性，外加优秀的抗干扰和 ESD 保护性能，确保单片机在恶劣的电磁干扰环境下可靠地运行。

此系列单片机提供了丰富的 HXT、HIRC 和 LIRC 振荡器功能选项，且内建完整的系统振荡器，无需外围元器件。其不同在工作模式之间动态切换的能力，为用户提供了一个优化单片机操作和减少功耗的手段。

外加时基功能、I/O 使用灵活等其它特性，使这款单片机可以广泛应用于各种产品中，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、马达控制等方面。

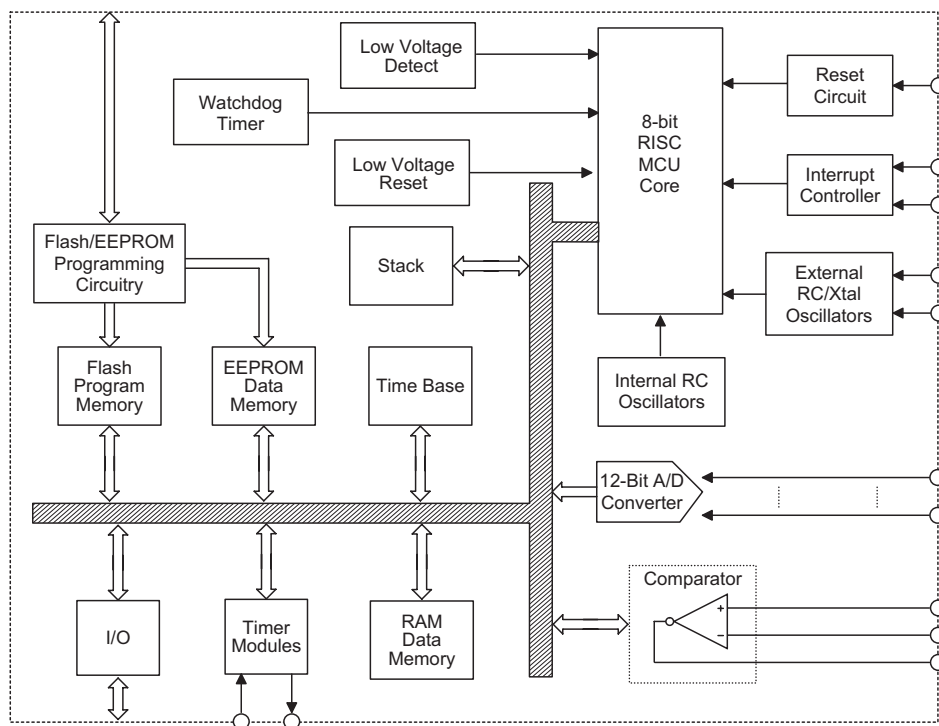
选型表

对此系列的芯片而言，大多数的特性参数都是一样的。主要差异在于 A/D 转换功能。下表列出了各单片机的主要特性。

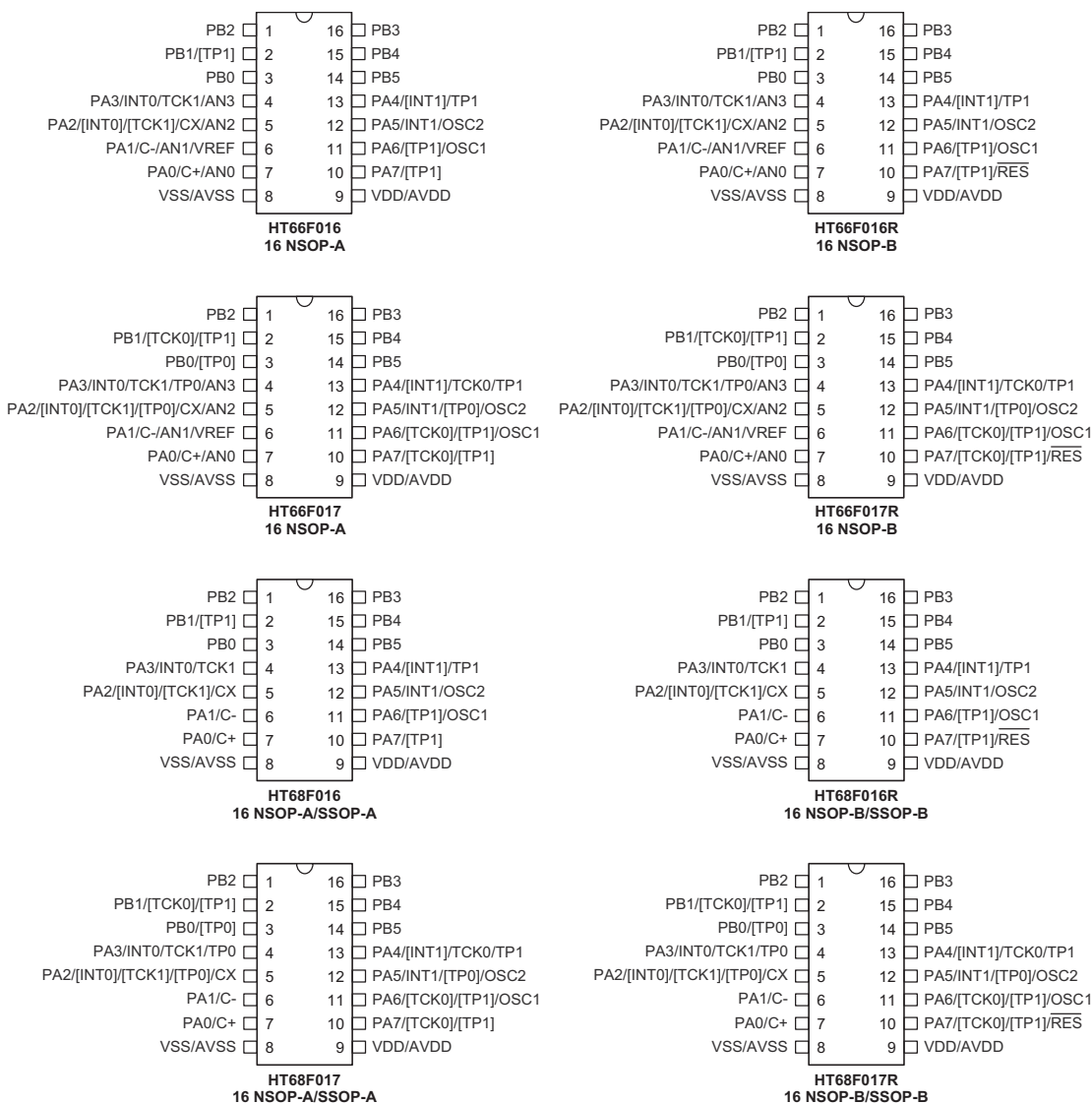
型号	ROM	RAM	EEPROM	I/O	A/D	16-bit TM	时基	比较器	堆栈	封装类型
HT68F016	1K×16	64×8	64×8	14	—	STM×1	2	1	4	16NSOP/SSOP
HT68F017	2K×16	128×8	64×8	14	—	CTM×1 STM×1	2	1	8	16NSOP/SSOP
HT66F016	1K×16	64×8	64×8	14	12-bit×4	STM×1	2	1	4	16NSOP
HT66F017	2K×16	128×8	64×8	14	12-bit×4	CTM×1 STM×1	2	1	8	16NSOP

注：HT66F016R/HT66F017R/HT68F016R/HT68F017R 提供外部复位引脚。

方框图



引脚图



- 注：1、括号内的引脚为可编程改变位置的引脚。
2、若共用脚同时有多种输出，“/”号右侧的引脚名具有更高的优先级。
3、VDD&AVDD 指的是 VDD 和 AVDD 为同一个引脚。

引脚说明

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

下表只介绍直接与 MCU 相关的引脚。其它的用于外设功能的引脚将在之后的章节相应外设功能模块中进行介绍。

HT66F016/HT68F016

引脚名称	功能	OP	I/T	O/T	共用引脚映射
PA0~PA7	端口 A	PAWU PAPU	ST	CMOS	—
PB0~PB5	端口 B	PBPU	ST	CMOS	—
AN0~AN3	ADC 输入	ACERL	AN	—	PA0~PA3
VREF	ADC 参考输入	ADCR1	AN	—	PA1
C-	比较器输入	CPC	AN	—	PA1
C+	比较器输入		AN	—	PA0
CX	比较器输出		—	CMOS	PA2
TCK1	TM1 输入	PRM	ST	—	PA2, PA3
TP1	TM1 输入 / 输出	PRM TMPC	ST	CMOS	PA4, PA6, PA7, PB1
INT0	外部中断 0	PRM INTC0 INTEG	ST	—	PA3, PA2
INT1	外部中断 1	PRM INTC2 INTEG	ST	—	PA5, PA4
OSC1	HXT 脚	CO	HXT	—	PA6
OSC2	HXT 脚	CO	—	HXT	PA5
RES	复位输入脚	CO	ST	—	PA7
VDD	电源电压	—	PWR	—	—
VSS	地	—	PWR	—	—

注：I/T：输入类型；O/T：输出类型

OP：通过配置选项（CO）或寄存器选项来设置

PWR：电源；CO：配置选项；ST：斯密特触发输入

CMOS：CMOS 输出

AN：模拟输入脚

HXT：高速晶体振荡器

HT68F016 没有 AN0~AN3，VREF 引脚。

HT66F017/HT68F017

引脚名称	功能	OP	I/T	O/T	共用引脚映射
PA0~PA7	端口 A	PAWU PAPU	ST	CMOS	—
PB0~PB5	端口 B	PBPU	ST	CMOS	—
AN0~AN3	ADC 输入	ACERL	AN	—	PA0~PA3
VREF	ADC 参考输入	ADCR1	AN	—	PA1
C-	比较器输入	CPC	AN	—	PA1
C+	比较器输入		AN	—	PA0
CX	比较器输出		—	CMOS	PA2
TCK0	TM0 输入	PRM	ST	—	PA4, PA6, PA7, PB1
TCK1	TM1 输入	PRM	ST	—	PA2, PA3
TP0	TM0 输入 / 输出	PRM TMPC	ST	CMOS	PA2, PA3, PA5, PB0
TP1	TM1 输入 / 输出	PRM TMPC	ST	CMOS	PA4, PA6, PA7, PB1
INT0	外部中断 0	PRM INTC0 INTEG	ST	—	PA3, PA2
INT1	外部中断 1	PRM INTC2 INTEG	ST	—	PA5, PA4
OSC1	HXT 脚	CO	HXT	—	PA6
OSC2	HXT 脚	CO	—	HXT	PA5
RES	复位输入脚	CO	ST	—	PA7
VDD	电源电压 *	—	PWR	—	—
AVDD	ADC 电源电压 *	—	PWR	—	—
VSS	地 **	—	PWR	—	—
AVSS	ADC 地 **	—	PWR	—	—

注：I/T：输入类型；O/T：输出类型

OP：通过配置选项（CO）或寄存器选项来设置

PWR：电源；CO：配置选项；ST：斯密特触发输入

CMOS：CMOS 输出；

HXT：高速晶体振荡器

*：VDD 是单机电源电压，而 AVDD 是 ADC 电源电压。AVDD 与 VDD 在内部是同一个引脚。

**：VSS 是单片机地引脚，而 AVSS 是 ADC 地引脚。AVSS 与 VSS 在内部是同一个引脚。

HT68F017 没有 AN0~AN3, VREF 引脚。

极限参数

电源供应电压	$V_{SS}-0.3V \sim V_{SS}+6.0V$
端口输入电压	$V_{SS}-0.3V \sim V_{DD}+0.3V$
储存温度	$-50^{\circ}C \sim 125^{\circ}C$
工作温度	$-40^{\circ}C \sim 85^{\circ}C$
I_{OL} 总电流	80mA
I_{OH} 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

$T_a=25^{\circ}C$

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DD}	工作电压 (HXT)	—	$f_{SYS}=8MHz$	2.2	—	5.5	V
			$f_{SYS}=12MHz$	2.7	—	5.5	V
			$f_{SYS}=16MHz$	3.3	—	5.5	V
			$f_{SYS}=20MHz$	4.5	—	5.5	V
	工作电压 (HIRC)	—	$f_{SYS}=8MHz$	2.2	—	5.5	V
			$f_{SYS}=12MHz$	2.7	—	5.5	V
			$f_{SYS}=16MHz$	3.3	—	5.5	V
I_{DD1}	工作电流 (HXT) ($f_{SYS}=f_H$, $f_S=f_{SUB}=f_{LIRC}$)	3V	无负载, $f_H=8MHz$,	—	1.0	1.5	mA
		5V	ADC off, WDT 使能	—	2.5	4.0	mA
		3V	无负载, $f_H=12MHz$,	—	1.5	2.5	mA
		5V	ADC off, WDT 使能	—	3.5	5.5	mA
		3.3V	无负载, $f_H=16MHz$,	—	2.0	3.0	mA
		5V	ADC off, WDT 使能	—	4.5	7.0	mA
		5V	无负载, $f_H=20MHz$, ADC off, WDT 使能	—	5.5	8.5	mA
I_{DD2}	工作电流 (HIRC) ($f_{SYS}=f_H$, $f_S=f_{SUB}=f_{LIRC}$)	3V	无负载, $f_H=8MHz$,	—	1.4	2.1	mA
		5V	ADC off, WDT 使能	—	2.4	3.6	mA
I_{DD3}	工作电流 (LIRC) ($f_{SYS}=f_L=f_{LIRC}$, $f_S=f_{SUB}=f_{LIRC}$)	3V	无负载,	—	10	20	μA
		5V	ADC off, WDT 使能	—	30	50	μA
I_{IDLE0}	IDLE0 模式静态电流 (LIRC on)	3V	无负载,	—	1.5	3.0	μA
		5V	ADC off, WDT 使能	—	3.0	6.0	μA

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
I _{IDLE1}	IDLE1 模式静态电流 (HXT)	3V	无负载, ADC off, WDT 使能 f _{sys} =12MHz on	—	0.6	1.0	mA
		5V		—	1.3	2.0	mA
	IDLE1 模式静态电流 (HIRC)	3V	无负载, ADC off, WDT 使能 f _{sys} =8MHz on	—	0.9	1.4	mA
		5V		—	1.2	1.8	mA
I _{SLEEP0}	SLEEP0 模式静态电流 (LIRC off)	3V	无负载, ADC off, WDT 除能	—	—	1	μA
		5V		—	—	2	μA
I _{SLEEP1}	SLEEP1 模式静态电流 (LIRC on)	3V	无负载, ADC off, WDT 使能	—	1.5	3.0	μA
		5V		—	2.5	5.0	μA
V _{IL1}	输入 / 输出或除 RES 脚以外的低电平输入电压	—	—	0	—	0.2V _{DD}	V
		5V	—	0	—	1.5	
V _{IH1}	输入 / 输出或除 RES 脚以外高电平输入电压	—	—	0.8V _{DD}	—	V _{DD}	V
		5V	—	3.5	—	5.0	
V _{IL2}	低电平输入电压 (RES)	—	—	0	—	0.4V _{DD}	V
V _{IH2}	高电平输入电压 (RES)	—	—	0.9V _{DD}	—	V _{DD}	V
V _{LVR1}	低电压复位电压	—	LVR 使能, 选择 2.10V	-5%	2.10	+5%	V
V _{LVR2}			LVR 使能, 选择 2.55V	-5%	2.55	+5%	V
V _{LVR3}			LVR 使能, 选择 3.15V	-5%	3.15	+5%	V
V _{LVR4}			LVR 使能, 选择 3.80V	-5%	3.80	+5%	V
V _{LVD1}	低电压检测电压	—	LVDEN=1, V _{LVD} =2.0V	-5%	2.00	+5%	V
V _{LVD2}			LVDEN=1, V _{LVD} =2.2V	-5%	2.20	+5%	V
V _{LVD3}			LVDEN=1, V _{LVD} =2.4V	-5%	2.40	+5%	V
V _{LVD4}			LVDEN=1, V _{LVD} =2.7V	-5%	2.70	+5%	V
V _{LVD5}			LVDEN=1, V _{LVD} =3.0V	-5%	3.00	+5%	V
V _{LVD6}			LVDEN=1, V _{LVD} =3.3V	-5%	3.30	+5%	V
V _{LVD7}			LVDEN=1, V _{LVD} =3.6V	-5%	3.60	+5%	V
V _{LVD8}			LVDEN=1, V _{LVD} =4.0V	-5%	4.00	+5%	V
I _{LVD1}	使用 LVR 和 LVD 的额外功耗	—	LVR 除能, LVDEN=1	—	75	120	μA
I _{LVD2}			LVR 使能, LVDEN=1	—	90	150	μA
V _{OL}	输入 / 输出输出低电平	3V	I _{OL} =9mA	—	—	0.3	V
		5V	I _{OL} =20mA	—	—	0.5	V
V _{OH}	输入 / 输出输出高电平	3V	I _{OH} =-3.2mA	2.7	—	—	V
		5V	I _{OH} =-7.4mA	4.5	—	—	V
R _{PH}	输入 / 输出上拉电阻	3V	—	20	60	100	kΩ
		5V		10	30	50	kΩ
V _{I25}	1.25V 参考电压	—	—	-3%	1.25	+3%	V
I _{I25}	使用 1.25V 参考电压的额外功耗	—	—	—	200	300	μA

交流电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
f _{CPU}	工作时钟	—	2.2V~5.5V	DC	—	8	MHz
			2.7V~5.5V	DC	—	12	MHz
			3.3V~5.5V	DC	—	16	MHz
			4.5V~5.5V	DC	—	20	MHz
f _{SYS}	系统时钟 (HXT)	—	2.2V~5.5V	0.4	—	8	MHz
			2.7V~5.5V	0.4	—	12	MHz
			3.3V~5.5V	0.4	—	16	MHz
			4.5V~5.5V	0.4	—	20	MHz
f _{HIRC}	系统时钟 (HIRC)	3.0V~5.5V	Ta=-40°C~85°C	-6%	8	+6%	MHz
		4.5V~5.5V	Ta=-40°C~85°C	-2%	8	+2%	MHz
f _{LIRC}	系统时钟 (LIRC)	5V	Ta=25°C	-10%	32	+10%	kHz
t _{INT}	中断脉宽	—	—	10	—	—	μs
t _{LVR}	低压复位时间	—	—	120	240	480	μs
t _{LVD}	低压中断时间	—	—	20	45	90	μs
t _{LVDS}	LVDO 稳定的时间	—	对于所有的 V _{LVD} , LVR 除能	200	—	—	μs
t _{BGS}	V _{BG} 打开稳定时间	—	—	200	—	—	μs
t _{TIMER}	TCKn 输入脉宽	—	—	0.3	—	—	μs
t _{SRESET}	软件复位时间	—	—	45	90	120	μs
t _{RSTD}	系统复位时间 (上电复位)	—	—	25	50	100	ms
	系统复位时间 (除上电复位)	—	—	8.3	16.7	33.3	ms
t _{SST}	系统启动时间 (从 HALT 中唤醒, HALT 时, f _{SYS} off)	—	f _{SYS} =HXT	128	—	—	t _{SYS}
			f _{SYS} =HIRC	16	—	—	
			f _{SYS} =LIRC	2	—	—	
	系统启动时间 (从 HALT 中唤醒, HALT 时, f _{SYS} on)	—	—	2	—	—	t _{SYS}
t _{EEIRD}	EEPROM 读周期	—	—	—	2	4	t _{SYS}
t _{EEWR}	EEPROM 写周期	—	—	—	2	4	ms

注：1、t_{SYS}=1/f_{SYS}

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

ADC 特性

Ta=25°C

符号	参数	测试条件		最小值	典型值	最大值	单位
		V _{DD}	条件				
AV _{DD}	ADC 工作电压	—	V _{REF} =AV _{DD}	2.7	—	5.5	V
V _{ADI}	ADC 输入电压	—	—	0	—	V _{REF}	V
V _{REF}	ADC 参考电压	—	—	2	—	AV _{DD}	V
DNL	A/D 非线性微分误差	5V	t _{ADCK} =1.0μs	—	±1	±2	LSB
INL	A/D 非线性积分误差	5V	t _{ADCK} =1.0μs	—	±2	±4	LSB
I _{ADC}	打开 A/D 增加的功耗	3V	无负载, t _{ADCK} =0.5μs	—	0.90	1.35	mA
		5V	无负载, t _{ADCK} =0.5μs	—	1.20	1.80	mA
t _{ADCK}	A/D 时钟周期	2.2~5.5V	—	0.5	—	10	μs
t _{ADC}	A/D 转换时间 (包括 A/D 采样和保持时间)	2.2~5.5V	12-bit ADC	—	16	—	t _{ADCK}
t _{ADS}	A/D 采样时间	2.2~5.5V	—	—	4	—	t _{ADCK}
T _{ON2ST}	A/D 转换启动到执行时间	2.2~5.5V	—	2	—	—	μs

比较器电气特性

Ta=25°C

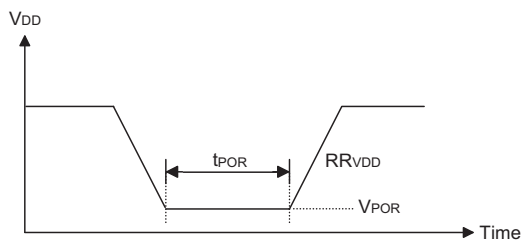
符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{CMP}	比较器工作电压	—	—	2.2	—	5.5	V
I _{CMP}	比较器工作电流	3V	—	—	37	56	μA
		5V	—	—	130	200	μA
V _{CMPOS}	比较器输入失调电压	—	—	-10	—	10	mV
V _{HYS}	迟滞宽度	—	—	20	40	60	mV
V _{CM}	比较器共模电压范围	—	—	V _{SS}	—	V _{DD} -1.4V	V
A _{OL}	比较器开环增益	—	—	60	80	—	dB
t _{PD}	比较器响应时间	—	100mV 偏置 (注)	—	370	560	ns

注：测量方式为：当一只输入脚的输入电压为 V_{CM}=(V_{DD}-1.4)/2 时，另一只输入脚的输入电压从 V_{SS} 到 (V_{CM}+100mV) 或从 V_{DD} 到 (V_{CM}-100mV) 转变。

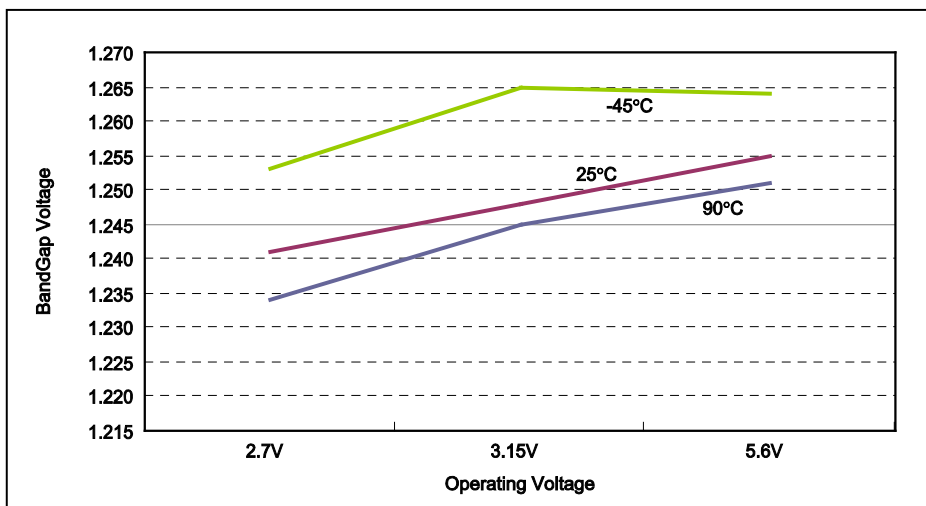
上电复位特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{POR}	上电复位电压	—	—	—	—	100	mV
RR _{VDD}	上电复位电压速率	—	—	0.035	—	—	V/ms
t _{POR}	V _{DD} 保持为 V _{POR} 的最小时间	—	—	1	—	—	ms



BandGap 电压参考 (V_{BG}) 特性曲线

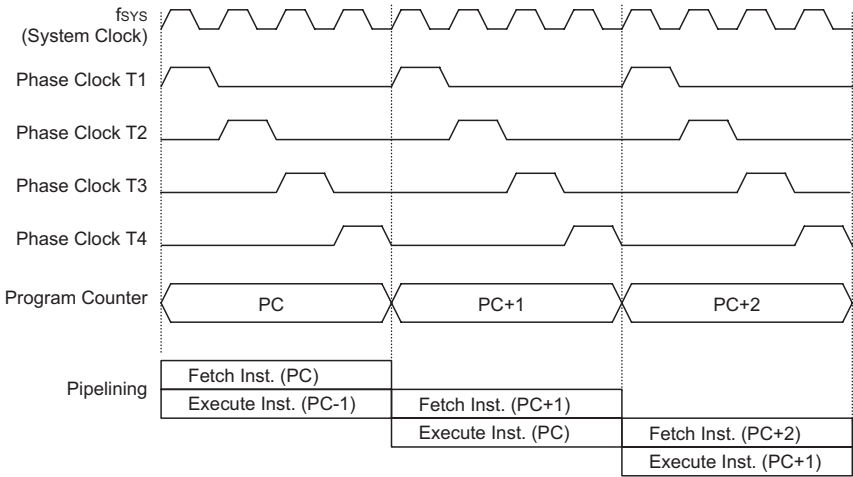


系统结构

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

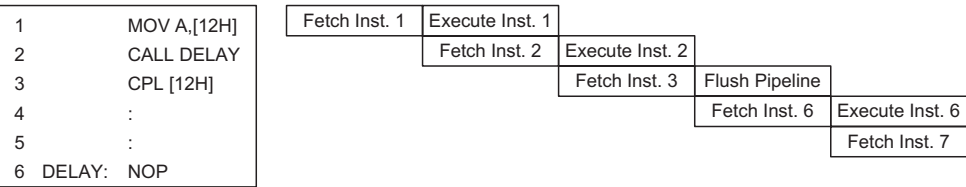
时序和流水线结构

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



系统时序和流水线

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



指令捕捉

程序计数器

在程序执行期间，程序计数器用来指向下一个要执行的指令地址。除了“JMP”和“CALL”指令需要跳转到一个非连续的程序存储器地址之外，它会在每条指令执行完成以后自动加一。选择不同型号的单片机，程序寄存器的宽度会因程序存储器的容量的不同而不同。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。

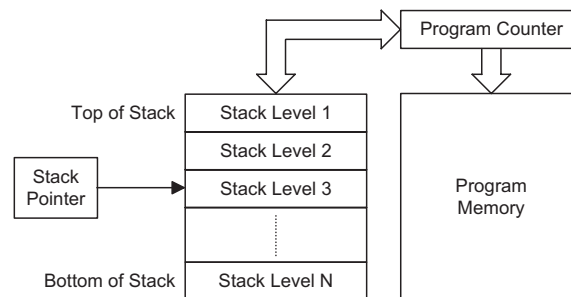
单片机型号	程序计数器	
	程序计数器高字节	PCL 寄存器
HT66F016	PC9~PC8	PCL7~PCL0
HT68F016		
HT66F017	PC10~PC8	
HT68F017		

程序计数器

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

堆栈

堆栈是一个特殊的存储空间，用来存储程序计数器中的内容。各单片机有不同的堆栈层数，堆栈既不是数据部分也不是程序空间部分，而且它既不是可读取也不是可写入的。当前层由堆栈指针 (SP) 加以指示，同样也是不可读写的。在子程序调用或中断响应服务时，程序计数器的内容被压入到堆栈中。当子程序或中断响应结束时，返回指令 (RET 或 RETI) 使程序计数器从堆栈中重新得到它以前的值。当一个芯片复位后，堆栈指针将指向堆栈顶部。



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

若堆栈溢出，则首个存入堆栈的程序计数器数据将会丢失。

单片机型号	堆栈层数
HT66F016/ HT68F016	4
HT66F017/ HT68F017	8

算术逻辑单元 – ALU

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

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

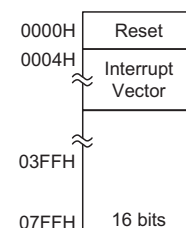
Flash 程序存储器

程序存储器用来存放用户代码即储存程序。程序存储器为 FLASH 类型意味着可以多次重复编程，方便用户使用同一芯片进行程序的修改。使用适当的单片机编程工具，此系列所有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

此系列单片机的程序存储器的容量为 1K×16~2K×16 位，程序存储器用程序计数器来寻址，其中也包含数据、表格和中断入口。数据表格可以设定在程序存储器的任何地址，由表格指针来寻址。

单片机型号	容量
HT66F016/HT68F016	1K×16
HT66F017/HT68F017	2K×16



程序存储器结构

特殊向量

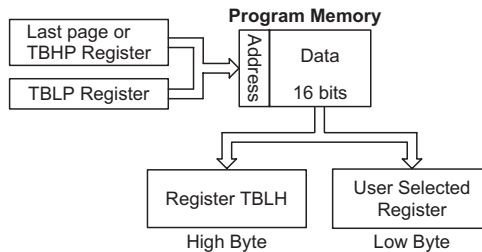
程序存储器内部某些地址保留用做诸如复位和中断入口等特殊用途。地址 000H 是芯片复位后的程序起始地址。在芯片复位之后，程序将跳到这个地址并开始执行。

查表

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

在设定完表格指针后，表格数据可以使用“TABRD [m]”或“TABRDL [m]”指令分别从程序存储器中查表读取。当这些指令执行时，程序存储器中表格数据低字节，将被传送到使用者所指定的数据存储器 [m]，程序存储器中表格数据的高字节，则被传送到 TBLH 特殊寄存器，而高字节中未使用的位将被读取为“0”。

下图是查表中寻址 / 数据流程：



查表范例

以下范例说明表格指针和表格数据如何被定义和执行。这个例子使用的表格数据用 ORG 伪指令储存在存储器中。此系列芯片中 ORG 指令的值“700H”指向的地址是 2K 程序存储器中最后一页的起始地址。表格指针的初始值设为 06H，这可保证从数据表格读取的第一笔数据位于程序存储器地址 706H，即最后一页起始地址后的第六个地址。值得注意的是，假如“TABRD [m]”指令被使用，则表格指针指向当前页。在这个例子中，表格数据的高字节等于零，而当“TABRD [m]”指令被执行时，此值将会自动的被传送到 TBLH 寄存器。

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

表格读取程序举例

```
tempreg1 db ?      ; temporary register #1
tempreg2 db ?      ; temporary register #2
:
:
mov a,06h           ; initialise low table pointer - note that this address
mov tblp,a          ; is referenced
mov a,07h           ; initialise high table pointer
mov tbhp,a
:
:
tabrd tempreg1      ; transfers value in table referenced by table pointer
                    ; data at program memory address "706H" transferred to
                    ; tempreg1 and TBLH
dec tblp            ; reduce value of table pointer by one
tabrd tempreg2      ; transfers value in table referenced by table pointer
                    ; data at program memory address "705H" transferred to
                    ; tempreg2 and TBLH in this example the data "1AH" is
                    ; transferred to tempreg1 and data "0FH" to register
                    ; tempreg2
:
:
org 700h            ; sets initial address of program memory
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
:
:
```

在线编程

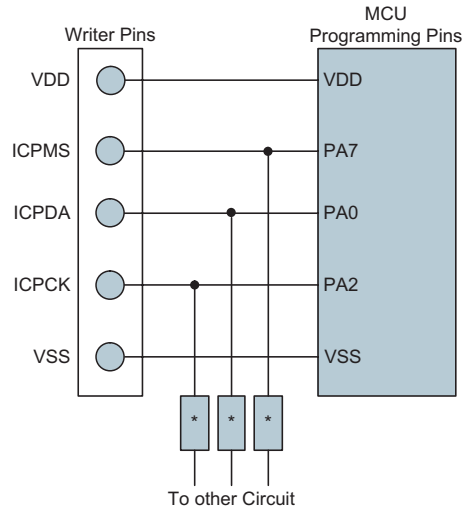
Flash 型程序存储器提供用户便利地对同一芯片进行程序的更新和修改。另外，Holtek 单片机提供 5 线接口的在线编程方式。用户可将进行过编程或未经过编程的单片机芯片连同电路板一起制成，最后阶段进行程序的更新和程序的烧写，在无需去除或重新插入芯片的情况下方便地保持程序为最新版。

Holtek Writer 引脚名称	MCU 在线编程引脚名称	引脚描述
ICPMS	PA7/RES	模式选择
ICPDA	PA0	串行数据输入 / 输出
ICPCK	PA2	串行时钟
VDD	VDD	电源
VSS	VSS	地

编程器和 MCU 引脚

芯片内部程序存储器可以通过 5 线的接口在线进行编程。其中 PA0 用于数据串行下载或上传、PA2 用于串行时钟、两条用于提供电源，另外一条用于复位。芯片在线烧写的详细使用说明超出此文档的描述范围，将由专门的参考文献提供。

在编程过程中，编程器会将 ICPMS 引脚一直拉低以除能单片机工作，并控制 PA0 和 PA2 脚进行数据和时钟编程，用户必须确保这两个引脚没有连接至其它输出脚。



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

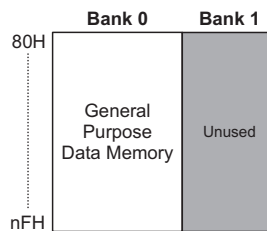
数据存储器

数据存储器是内容可更改的 8 位 RAM 内部存储器，用来储存临时数据。

结构

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

单片机型号	容量	Bank0	Bank1
HT66F016 HT68F016	64 $\times$ 8	80H~BFH	未使用
HT66F017 HT68F017	128 $\times$ 8	80H~FFH	未使用



Note: General Purpose Data Memory space:
 80H~BFH for HT66F016/HT68F016
 80H~FFH for HT66F017/HT68F017

通用功能数据存储器

第二部分数据存储器是做一般用途使用，都可在程序控制下进行读取和写入。
总的数据存储器被分为两个区。大部分特殊功能数据寄存器均可在所有 Bank 被访问，处于“40H”地址的 EEC 寄存器却只能在 Bank1 中被访问到。切换不同区域可通过设置区域指针 (BP) 实现。所有单片机的数据存储器的起始地址都是“00H”。

Bank 0		Bank 1	Bank 0		Bank 1
00H	IAR0		20H	ADRL	
01H	MP0		21H	ADRH	
02H	IAR1		22H	ADCR0	
03H	MP1		23H	ADCR1	
04H	BP		24H	ACERL	
05H	ACC		25H	CPC	
06H	PCL		26H	CTRL	
07H	TBLP		27H	LVRCL	
08H	TBLH		28H	Unused	
09H	TBHP		29H	Unused	
0AH	STATUS		2AH	Unused	
0BH	SMOD		2BH	Unused	
0CH	LVDC		2CH	Unused	
0DH	INTEG		2DH	Unused	
0EH	INTC0		2EH	Unused	
0FH	INTC1		2FH	TM1C0	
10H	INTC2		30H	TM1C1	
11H	MFI0		31H	TM1DL	
12H	MFI1		32H	TM1DH	
13H	MFI2		33H	TM1AL	
14H	PA		34H	TM1AH	
15H	PAC		35H	TM1RP	
16H	PAPU		36H	Unused	
17H	PAWU		37H	Unused	
18H	PRM		38H	Unused	
19H	TMPC		39H	Unused	
1AH	WDTC		3AH	Unused	
1BH	TBC		3BH	Unused	
1CH	Unused		3CH	PB	
1DH	Unused		3DH	PBC	
1EH	EEA		3EH	PBC	
1FH	EED		3FH	PBC	
			40H	Unused	EEC
			41H	Unused	
			42H	Unused	
			43H	Unused	
			44H	Unused	
			45H	Unused	
			46H	Unused	
			47H	Unused	
			48H	Unused	
			49H	Unused	
			4AH	Unused	
			4BH	Unused	
			4CH	Unused	
			4DH	Unused	
			4EH	Unused	
			4FH	Unused	
			50H	Unused	
			51H	Unused	
			52H	Unused	
			53H	Unused	
			54H	Unused	
			55H	Unused	
			56H	Unused	
			57H	Unused	
			58H	Unused	
			59H	Unused	
			5AH	Unused	
			5BH	Unused	
			5CH	Unused	
			5DH	Unused	
			5EH	Unused	
			5FH	Unused	
			60H	Unused	
			61H	Unused	
			62H	Unused	
			63H	Unused	
			64H	Unused	
			65H	Unused	
			66H	Unused	
			67H	Unused	
			68H	Unused	
			69H	Unused	
			6AH	Unused	
			6BH	Unused	
			6CH	Unused	
			6DH	Unused	
			6EH	Unused	
			6FH	Unused	
			70H	Unused	
			71H	Unused	
			72H	Unused	
			73H	Unused	
			74H	Unused	
			75H	Unused	
			76H	Unused	
			77H	Unused	
			78H	Unused	
			79H	Unused	
			7AH	Unused	
			7BH	Unused	
			7CH	Unused	
			7DH	Unused	
			7EH	Unused	
			7FH	Unused	

Unused, read as "00"

注：HT68F016 20H~24H 未被使用

HT66F016/HT68F016 特殊功能数据存储器

Bank 0 Bank 1		Bank 0 Bank 1	
00H	IAR0	20H	ADRL
01H	MP0	21H	ADRH
02H	IAR1	22H	ADCR0
03H	MP1	23H	ADCR1
04H	BP	24H	ACERL
05H	ACC	25H	CPC
06H	PCL	26H	CTRL
07H	TBLP	27H	LVRC
08H	TBLH	28H	TM0C0
09H	TBHP	29H	TM0C1
0AH	STATUS	2AH	TM0DL
0BH	SMOD	2BH	TM0DH
0CH	LVDC	2CH	TM0AL
0DH	INTEG	2DH	TM0AH
0EH	INTC0	2EH	TM0RP
0FH	INTC1	2FH	TM1C0
10H	INTC2	30H	TM1C1
11H	MF10	31H	TM1DL
12H	MF11	32H	TM1DH
13H	MF12	33H	TM1AL
14H	PA	34H	TM1AH
15H	PAC	35H	TM1RP
16H	PAPU	36H	Unused
17H	PAWU	37H	
18H	PRM	38H	
19H	TMPC	39H	
1AH	WDTC	3CH	PB
1BH	TBC	3DH	PBC
1CH	Unused	3EH	PBC
1DH	Unused	3FH	PBPU
1EH	EEA	40H	Unused EEC
1FH	EED	41H	Unused
		42H	
		43H	
		44H	
		45H	Unused
		46H	
		47H	
		48H	
		49H	Unused
		4AH	
		4BH	
		4CH	
		4DH	Unused
		4EH	
		4FH	
		50H	
		51H	Unused
		52H	
		53H	
		54H	
		55H	Unused
		56H	
		57H	
		58H	
		59H	Unused
		5AH	
		5BH	
		5CH	
		5DH	Unused
		5EH	
		5FH	
		60H	
		61H	Unused
		62H	
		63H	
		64H	
		65H	Unused
		66H	
		67H	
		68H	
		69H	Unused
		6AH	
		6BH	
		6CH	
		6DH	Unused
		6EH	
		6FH	
		70H	
		71H	Unused
		72H	
		73H	
		74H	
		75H	Unused
		76H	
		77H	
		78H	
		79H	Unused
		7AH	
		7BH	
		7CH	
		7DH	Unused
		7EH	
		7FH	
		80H	

Unused, read as "00"

注：HT68F017 20H~24H 未被使用

HT66F017/HT68F017 特殊功能数据存储器

特殊功能寄存器

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

间接寻址寄存器 – IAR0, IAR1

间接寻址寄存器 IAR0 和 IAR1 的地址虽位于数据存储区，但其并没有实际的物理地址。间接寻址的方法准许使用间接寻址指针做数据操作，以取代定义实际存储器地址的直接存储器寻址方法。在间接寻址寄存器 (IAR0 和 IAR1) 上的任何动作，将对间接寻址指针 (MP0 和 MP1) 所指定的存储器地址产生对应的读 / 写操作。它们总是成对出现，IAR0 和 MP0 可以访问 Bank 0，而 IAR1 和 MP1 可以访问 Bank 0 和 Bank 1。因为这些间接寻址寄存器不是实际存在的，直接读取将返回 “00H” 的结果，而直接写入此寄存器则不做任何操作。

间接寻址指针 – MP0, MP1

该系列单片机提供两个间接寻址指针，即 MP0 和 MP1。由于这些指针在数据存储区中能像普通的寄存器一般被操作，因此提供了一个寻址和数据追踪的有效方法。当对间接寻址寄存器进行任何操作时，单片机指向的实际地址是由间接寻址指针所指定的地址。MP0, IAR0 用于访问 Bank 0，而 MP1 和 IAR1 可通过 BP 寄存器访问所有的 Bank。Bank 0 仅可使用直接寻址，其它所有 Bank 都可使用 MP1 和 IAR1 进行间接寻址。注意，MP0 和 MP1 为 8 位寄存器，通常与 IAR0 和 IAR1 一起对相应的数据存储区进行访问。

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

间接寻址程序举例

```
data .section 'data'
adres1      db ?
adres2      db ?
adres3      db ?
adres4      db ?
block       db ?
code .section at 0 'code'
org 00h
start:
    mov a,04h                ; setup size of block
    mov block,a
    mov a,offset adres1      ; Accumulator loaded with first RAM address
    mov mp0,a                ; setup memory pointer with first RAM address
loop:
    clr IAR0                  ; clear the data at address defined by mp0
    inc mp0                   ; increment memory pointer
    sdz block                  ; check if last memory location has been cleared
    jmp loop
continue:
```

在上面的例子中有一点值得注意，即并没有确定 RAM 地址。

存储区指针 – BP

数据存储器被分为两个部分。可以通过设置 Bank 指针（Bank Pointer）值来访问不同的数据存储区。BP 指针的第 0 位用于选择数据存储区 0~1。

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

BP 寄存器

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

Bit 7~1 未使用，读为“0”

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

累加器 – ACC

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

程序计数器低字节寄存器 – PCL

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

表格寄存器 – TBLP, TBHP, TBLH

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

状态寄存器 – STATUS

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

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

Z、OV、AC 和 C 标志位通常反映最近运算的状态。

- **C**: 当加法运算的结果产生进位，或减法运算的结果没有产生借位时，则 C 被置位，否则 C 被清零，同时 C 也会被带进位的移位指令所影响。
- **AC**: 当低半字节加法运算的结果产生进位，或高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	×	×	×	×

“×”：未知

Bit 7~6 未使用，读为“0”

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

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

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

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

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

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

EEPROM 数据寄存器

此系列所有单片机的一个特性是内建 EEPROM 数据存储器。“Electrically Erasable Programmable Read Only Memory”为电可擦可编程只读存储器，由于其非易失的存储结构，即使在电源掉电的情况下存储器内的数据仍然保存完好。这种存储区扩展了 ROM 空间，对设计者来说增加了许多新的应用机会。EEPROM 可以用来存储产品编号、校准值、用户特定数据、系统配置参数或其它产品信息等。EEPROM 的数据读取和写入过程也会变的更简单。

EEPROM 数据寄存器结构

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

单片机型号	容量	地址
所有单片机	64×8	00H~3FH

EEPROM 寄存器

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

EEPROM 寄存器列表

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

EEA 寄存器

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

“×”；未知

- Bit 7~6 未定义，读为“0”
- Bit 5~0 数据 EEPROM 地址
- 数据 EEPROM 地址 Bit 5~Bit 0

EEC 寄存器

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

Bit 7~4 未定义，读为“0”

Bit 3 **WREN**: 数据 EEPROM 写使能位

0: 除能

1: 使能

此位为数据 EEPROM 写使能位，向数据 EEPROM 写操作之前需将此位置高。将此位清零时，则禁止向数据 EEPROM 写操作。

Bit 2 **WR**: EEPROM 写控制位

0: 写周期结束

1: 写周期有效

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

Bit 1 **RDEN**: 数据 EEPROM 读使能位

0: 除能

1: 使能

此位为数据 EEPROM 读使能位，向数据 EEPROM 读操作之前需将此位置高。将此位清零时，则禁止向数据 EEPROM 读操作。

Bit 0 **RD**: EEPROM 读控制位

0: 读周期结束

1: 读周期有效

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

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

EED 寄存器

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

“×”：未知

Bit 7~0 数据 EEPROM 地址

数据 EEPROM 地址 Bit 7~Bit 0

从 EEPROM 中读取数据

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

写数据到 EEPROM

写数据至 EEPROM，EEPROM 中写入数据的地址要先放入 EEA 寄存器中，写入的数据需存入 EED 寄存器中。EEC 寄存器中的写使能位 WREN 先置为高以使能写功能，若 EEC 寄存器中 WR 位被置为高，一个内部写周期将开始。这两个指令必须连续进行。在进行写操作前，要先将总的中断控制位 EMI 清零，当写周期开始后再将其置为 1。需注意的是若 WR 位已置为高而 WREN 位还未被设置则不能开始写操作。由于控制 EEPROM 写周期是一个内部时钟，与单片机的系统时钟异步，所以数据写入 EEPROM 的时间将有所延迟。可通过轮询 EEC 寄存器中的 WR 位或判断 EEPROM 中断以侦测写周期是否完成。若写周期完成，WR 位将自动清除为“0”，通知用户数据已写入 EEPROM。因此，应用程序将轮询 WR 位以确定写周期是否结束。

写保护

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

EEPROM 中断

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

编程注意事项

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

写数据时，WREN 位置为“1”后，WR 须立即设置为高，以确保正确地执行写周期。写周期执行前总中断位 EMI 应先清零，写周期开始执行后再将此位重新使能。

程序举例

从 EEPROM 中读取数据 – 轮询法

```
MOV A, EEPROM_ADRES      ; user defined address
MOV EEA, A
MOV A, 040H               ; setup memory pointer MP1
MOV MP1, A                ; MP1 points to EEC register
MOV A, 01H                ; setup Bank Pointer
MOV BP, A
SET IAR1.1                ; set RDEN bit, enable read operations
SET IAR1.0                ; start Read Cycle - set RD bit
BACK:
SZ IAR1.0                 ; check for read cycle end
JMP BACK
CLR IAR1                  ; disable EEPROM read/write
CLR BP
MOV A, EED                ; move read data to register
MOV READ_DATA, A
```

写数据到 EEPROM – 轮询法

```
CLR EMI
MOV A, EEPROM_ADRES      ; user defined address
MOV EEA, A
MOV A, EEPROM_DATA       ; user defined data
MOV EED, A
MOV A, 040H               ; setup memory pointer MP1
MOV MP1, A                ; MP1 points to EEC register
MOV A, 01H                ; setup Bank Pointer
MOV BP, A
SET IAR1.3                ; set WREN bit, enable write operations
SET IAR1.2                ; start Write Cycle - set WR bit - executed
                        ; immediately after set WREN bit

SET EMI
BACK:
SZ IAR1.2                 ; check for write cycle end
JMP BACK
CLR IAR1                  ; disable EEPROM read/write
CLR BP
```


振荡器

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

振荡器概述

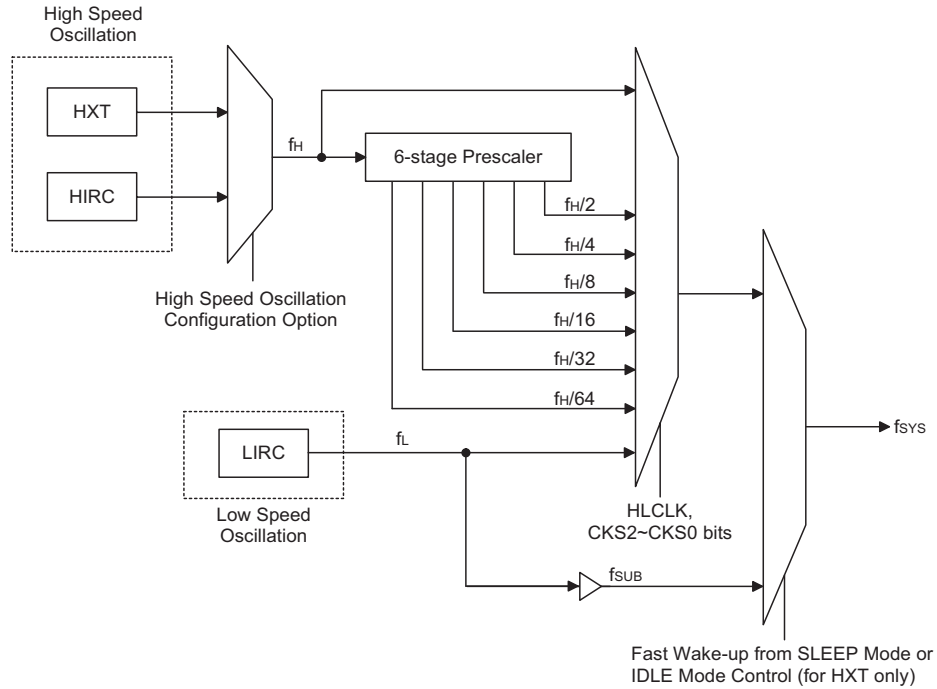
振荡器除了作为系统时钟源，还作为看门狗定时器和时基功能的时钟源。外部振荡器需要一些外围器件，而集成的两个内部振荡器不需要任何外围器件。它们提供的高速和低速系统振荡器具有较宽的频率范围。所有振荡器选择通过配置选项选择。较高频率的振荡器提供更高的性能，但要求有更高的功率，反之亦然。动态切换快慢系统时钟的能力使单片机具有灵活而优化的性能 / 功耗比，此特性对功耗敏感的应用领域尤为重要。

类型	名称	频率	引脚
外部晶振	HXT	400kHz~20MHz	OSC1/OSC2
内部高速 RC	HIRC	8 MHz	—
内部低速 RC	LIRC	32kHz	—

振荡器类型

系统时钟配置

此系列的单片机有三个系统振荡器，包括两个高速振荡器和一个低速振荡器。高速振荡器有外部晶体 / 陶瓷振荡器 – HXT 和内部 8MHz RC 振荡器 – HIRC。低速振荡器为内部 32kHz 振荡器 – LIRC。使用高速或低速振荡器作为系统时钟的选择是通过设置 SMOD 寄存器中的 HLCLK 位及 CKS2~CKS0 位决定的，系统时钟可动态选择。高速或低速振荡器的实际时钟源经由配置选项选择。低速或高速系统时钟频率由 SMOD 寄存器的 HLCLK 位及 CKS2~CKS0 位决定的。请注意，两个振荡器必须做出选择，即一个高速和一个低速振荡器。选择外部晶振时，OSC1 和 OSC2 脚与外部元件连接。

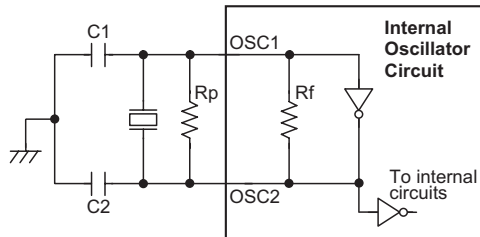


系统时钟配置

外部晶体 / 陶瓷振荡器 – HXT

外部高频晶体 / 陶瓷振荡器可通过配置选项选择。对于晶体振荡器，只要简单地将晶体连接至 OSC1 和 OSC2，则会产生振荡所需的相移及反馈，而不需其它外部器件。为保证某些低频率的晶体振荡和陶瓷谐振器的振荡频率更精准，建议连接两个小容量电容 C1 和 C2 到 VSS，具体数值与客户选择的晶体 / 陶瓷晶振有关。

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



Note: 1. R_p is normally not required. C1 and C2 are required.
2. Although not shown OSC1/OSC2 pins have a parasitic capacitance of around 7pF.

晶体 / 陶瓷振荡器 – HXT

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

晶体振荡器电容推荐值

内部 RC 振荡器 – HIRC

内部 RC 振荡器是一个集成的系统振荡器，不需其它外部器件。内部 RC 振荡器具有固定的频率：8MHz。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡频率因 V_{DD} 、温度以及芯片制成工艺不同的影响较大程度地降低。在电源电压为 5V 及温度为 $-40^{\circ}\text{C}\sim 85^{\circ}\text{C}$ 的条件下，8MHz 频率的容差为 2%。如果选择了该内部时钟，无需额外的引脚；PA6 和 PA5 可以作为通用 I/O 口使用。

内部 32kHz 振荡器 – LIRC

内部 32kHz 系统振荡器也是一个低频振荡器，经由配置选项选择。这种单片机有一个完全集成 RC 振荡器，它在 5V 电压下运行的典型频率值为 32kHz 且无需外部元件。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡器因电源电压、温度及芯片制成工艺不同的影响较大程度地降低。因此，内部 32kHz 振荡器频率在 25°C 温度 5V 电压下的精度保持在 10% 以内。

辅助振荡器

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

工作模式和系统时钟

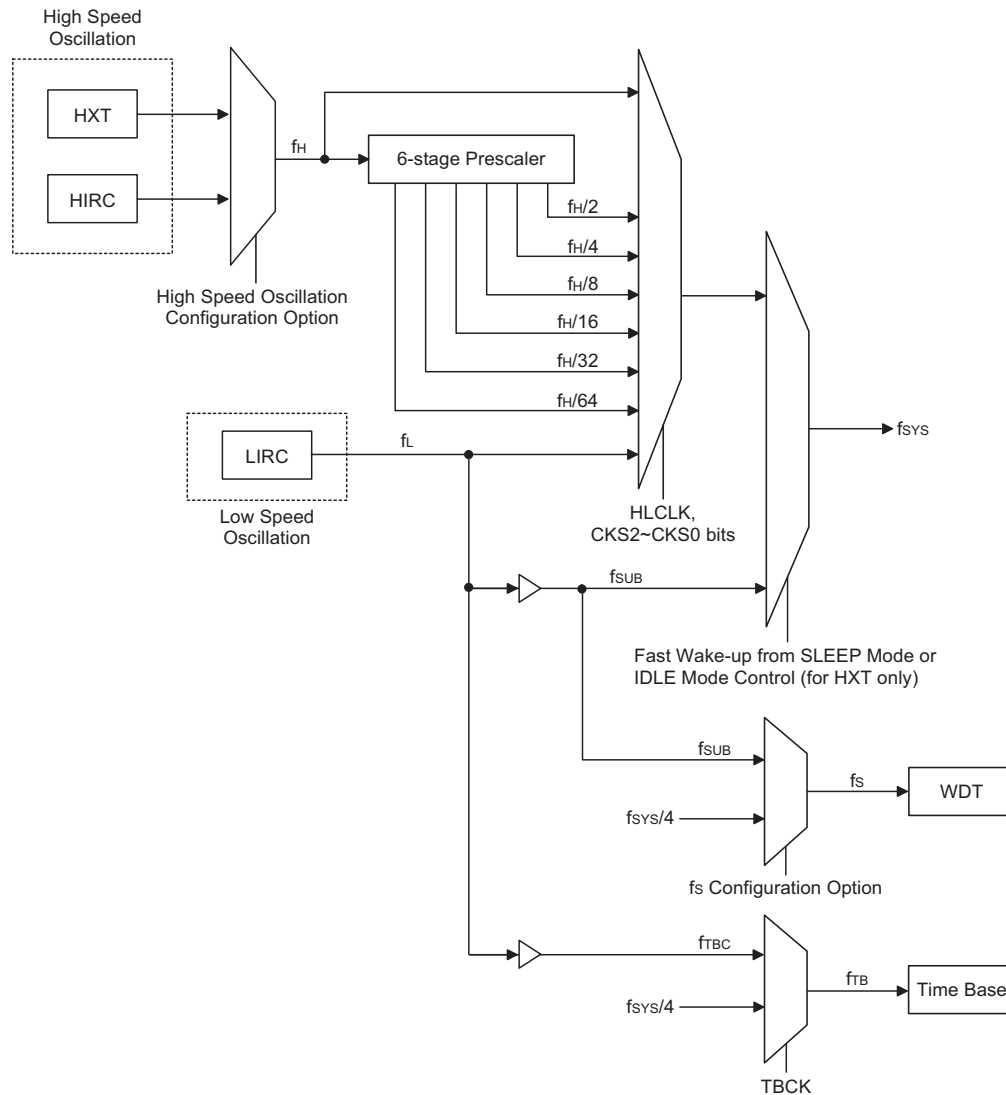
现今的应用要求单片机具有较高的性能及尽可能低的功耗，这种矛盾的要求在便携式电池供电的应用领域尤为明显。高性能所需要的高速时钟将增加功耗，反之亦然。此单片机提供高、低速两种时钟源，它们之间可以动态切换，用户可通过优化单片机操作来获得较佳性能 / 功耗比。

系统时钟

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

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

另外两个内部时钟用于外围电路，次时钟源 f_{SUB} 和时基时钟 f_{TBC} 。这两个时钟来自 LIRC 振荡器。快速唤醒发生后， f_{SUB} 为单片机提供一个次时钟。 f_{SUB} 和 $f_{SYS}/4$ 用于看门狗定时器的时钟源。 f_{TBC} 用于时基中断功能和 TM 的时钟源。



系统时钟选项

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

系统工作模式

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

工作模式	说明				
	CPU	f _{sys}	f _{sub}	f _s	f _{tbc}
正常模式	On	f _H ~f _H /64	On	On	On
低速模式	On	f _L	On	On	On
空闲模式 0	Off	Off	On	On	On
空闲模式 1	Off	On	On	On	On
休眠模式 0	Off	Off	Off	Off	Off
休眠模式 1	Off	Off	On	On	Off

- 正常模式

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

- 低速模式

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

- 休眠模式 0

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

- 休眠模式 1

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

- 空闲模式 0

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

- 空闲模式 1

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

控制寄存器

寄存器 SMOD 用于控制单片机内部时钟。

SMOD 寄存器

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

- Bit 7~5 **CKS2~CKS0:** 当 HLCLK 为“0”时系统时钟选择位
000: f_L (f_{LIRC})
001: f_L (f_{LIRC})
010: $f_H/64$
011: $f_H/32$
100: $f_H/16$
101: $f_H/8$
110: $f_H/4$
111: $f_H/2$
这三位用于选择系统时钟源。除了 LIRC 振荡器提供的系统时钟源外，也可使用高频振荡器的分频作为系统时钟源。
- Bit 4 **FSTEN:** 快速唤醒控制位（仅用于 HXT）
0: 除能
1: 使能
此位为快速唤醒控制位，用于决定单片机被唤醒后 f_{SUB} 是否开始工作。若 f_{SUB} 有效，当此位为高时， f_{SUB} 可暂时作为系统时钟，用于快速唤醒。
- Bit 3 **LTO:** 低速振荡器就绪标志位
0: 未就绪
1: 就绪
此位为低速系统振荡器就绪标志位，用于表明低速系统振荡器在系统上电复位或经唤醒后何时稳定下来。当系统处于 SLEEP0 模式时，该标志为低。若系统时钟来自 LIRC 振荡器，系统唤醒后该位转换为高需 1~2 个时钟周期。
- Bit 2 **HTO:** 高速振荡器就绪标志位
0: 未就绪
1: 就绪
此位为高速系统振荡器就绪标志位，用于表明高速系统振荡器何时稳定下来。此标志在系统上电后经硬件清零，高速系统振荡器稳定后变为高电平。因此，此位在单片机上电后由应用程序读取的总为“1”。该标志由休眠模式或空闲模式 0 中唤醒后会处于低电平状态，若使用 HXT 振荡器，该位将在 128 个时钟周期后变为高电平状态，若使用 HIRC 振荡器则只需 15~16 个时钟周期即可。
- Bit 1 **IDLEN:** 空闲模式控制位
0: 除能
1: 使能
此位为空闲模式控制位，用于决定 HALT 指令执行后发生的动作。若此位为高，当指令 HALT 执行后，单片机进入空闲模式。若 FSYSON 位为高，在空闲模式 1 中 CPU 停止运行，系统时钟将继续工作以保持外围功能继续工作；若 FSYSON 为低，在空闲模式 0 中 CPU 和系统时钟都将停止运行。若此位为低，单片机将在 HALT 指令执行后进入休眠模式。
- Bit 0 **HLCLK:** 系统时钟选择位
0: $f_H/2 \sim f_H/64$ 或 f_L
1: f_H
此位用于选择 f_H 或 $f_H/2 \sim f_H/64$ 还是 f_L 作为系统时钟。该位为高时选择 f_H 作为系统时钟，为低时则选择 $f_H/2 \sim f_H/64$ 或 f_L 作为系统时钟。当系统时钟由 f_H 时钟向 f_L 时钟转换时， f_H 将自动关闭以降低功耗。

快速唤醒

单片机进入休眠模式或空闲模式 0 后，系统时钟将停止以降低功耗。然而单片机再次唤醒，原来的系统时钟重新起振、稳定且恢复正常工作需要一定的时间。为确保单片机能够尽快的开始工作，系统提供了一个快速唤醒功能。需提供一临时时钟源 f_{SUB} 先驱动系统直至原系统振荡器稳定，这个临时时钟来自 LIRC 振荡器。快速启动功能的时钟源为 f_{SUB} ，该功能仅在休眠模式 1 和空闲模式 0 中有效。当单片机由休眠模式 0 唤醒时，因 f_{SUB} 已停止，故快速唤醒功能不受影响。快速唤醒功能使能 / 除能由 SMOD 寄存器中 FSTEN 位控制的。

若 HXT 振荡器作为正常模式的系统时钟，且快速唤醒功能使能，系统唤醒将需 1~2 个 t_{SUB} 时钟周期。系统开始在 f_{SUB} 时钟源下运行直至 128 个 HXT 时钟周期后 HTO 标志转换为高，系统将切换到 HXT 振荡器运行。

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

系统振荡器	FSTEN 位	唤醒时间 (休眠模式 0)	唤醒时间 (休眠模式 1)	唤醒时间 (空闲模式 0)	唤醒时间 (空闲模式 1)
HXT	0	128 个 HXT 周期	128 个 HXT 周期	1~2 个 HXT 周期	1~2 个 HXT 周期
	1	128 个 HXT 周期	1~2 个 f_{SUB} 周期（系统在 f_{SUB} 下运行 128 个 HXT 周期后切换到 HXT 振荡器运行）	1~2 个 HXT 周期	1~2 个 HXT 周期
HIRC	×	15~16 个 HIRC 周期	15~16 个 HIRC 周期	1~2 个 HIRC 周期	1~2 个 HIRC 周期
LIRC	×	1~2 个 LIRC 周期	1~2 个 LIRC 周期	1~2 个 LIRC 周期	1~2 个 LIRC 周期

“×”：未知

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

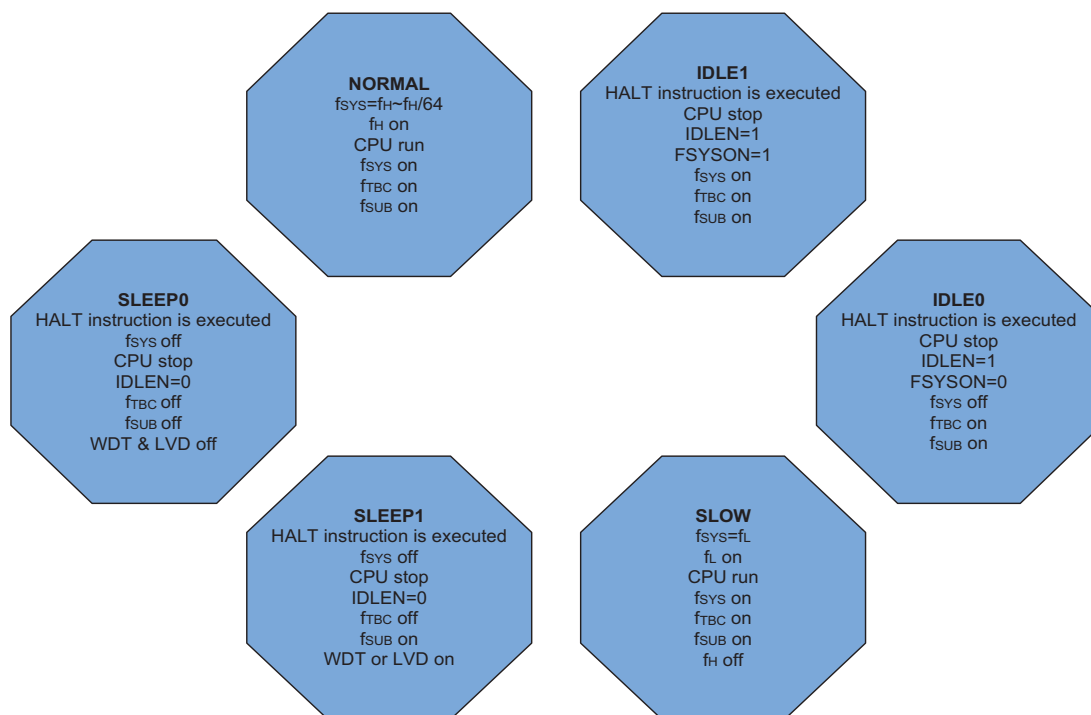
唤醒时间

工作模式切换和唤醒

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

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

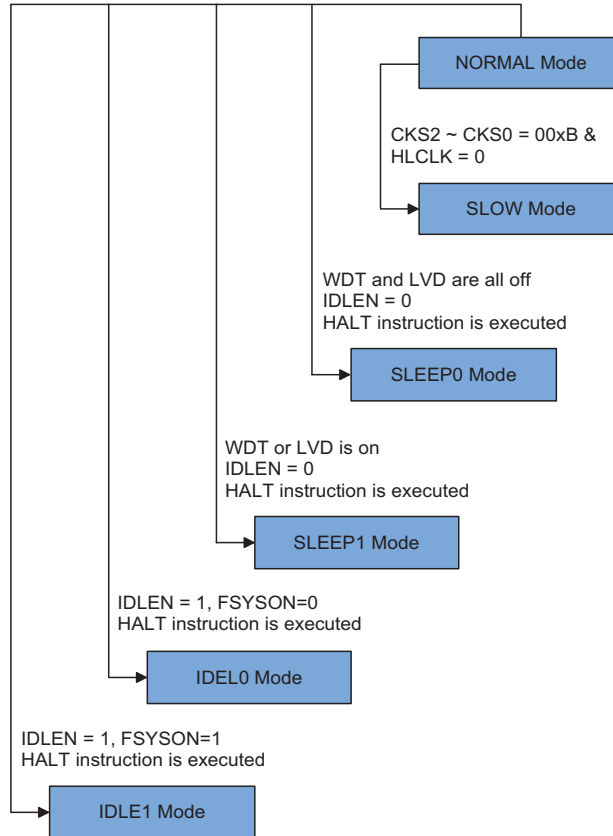
当 HLCLK 位变为低电平时，时钟源将由高速时钟源 f_H 转换成时钟源 $f_H/2 \sim f_H/64$ 或 f_L 。若时钟源来自 f_L ，高速时钟源将停止运行以节省耗电。此时须注意， $f_H/16$ 和 $f_H/64$ 内部时钟源也将停止运行，由此会影响到如 TMs 等内部功能的工作。所附流程图显示了单片机在不同工作模式间切换时的变化。



正常模式切换到低速模式

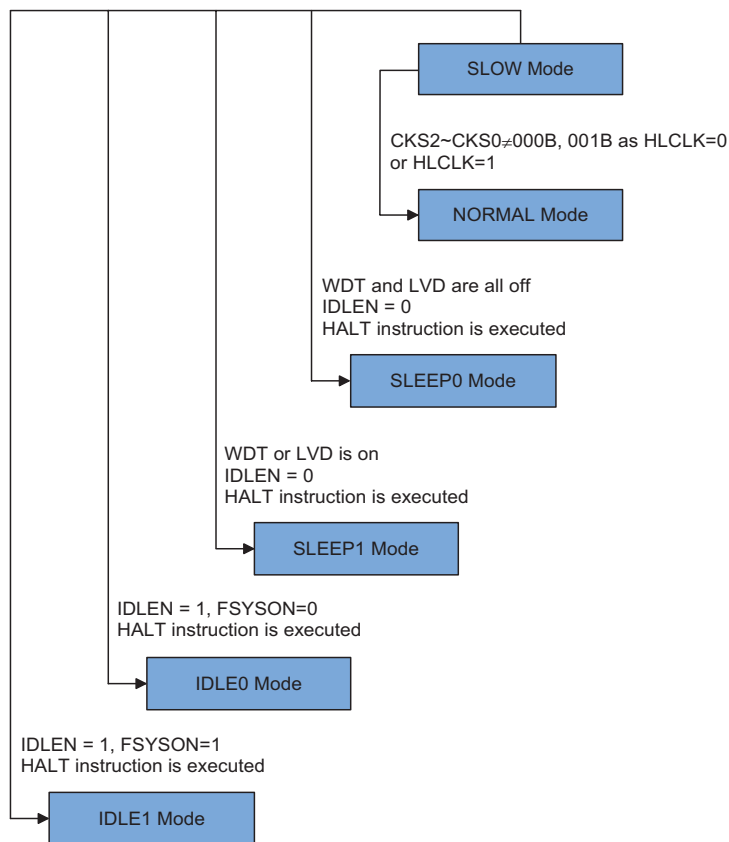
系统运行在正常模式时使用高速系统振荡器，因此较为耗电。可通过设置 SMOD 寄存器中的 HLCLK 位为“0”及 CKS2~CKS0 位为“000”或“001”使系统时钟切换至运行在低速模式下。此时将使用低速系统振荡器以节省耗电。用户可在对性能要求不高的操作中使用此方法以减少耗电。

低速模式的时钟源来自 LIRC 振荡器，因此要求这些振荡器在所有模式切换动作发生前稳定下来。该动作由 SMOD 寄存器中 LTO 位控制。



低速模式切换到正常模式

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



进入休眠模式 0

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

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

进入休眠模式 1

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

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

进入空闲模式 0

进入空闲模式 0 的方法仅有一种 —— 应用程序中执行“HALT”指令前需设置寄存器 SMOD 中 IDLEN 位为“1”且 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，时基时钟和 f_{SUB} 时钟将继续运行。
- 数据存储器中的内容和寄存器将保持当前值。
- 若 WDT 使能且其时钟源来自 f_{SUB} ，则 WDT 将被清零并重新开始计数；若其时钟源来自系统时钟，则 WDT 将停止运行。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

进入空闲模式 1

进入空闲模式 1 的方法仅有一种 —— 应用程序中执行“HALT”指令前需设置寄存器 SMOD 中 IDLEN 位为“1”且 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} 开启，应用程序停止在“HALT”指令处。
- 数据存储器中的内容和寄存器将保持当前值。
- 若 WDT 使能，无论 WDT 时钟源来自 f_{SUB} 或是系统时钟，则 WDT 将被清零并重新开始计数。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。

静态电流的注意事项

由于单片机进入休眠或空闲模式的主要原因是将 MCU 的电流降低到尽可能低，可能到只有几个微安的级别（空闲模式 1 除外），所以如果要将电路的电流进一步降低，电路设计者还应有其它的考虑。应该特别注意的是单片机的输入/输出引脚。所有高阻抗输入脚都必须连接到固定的高或低电平，因为引脚浮空会造成内部振荡并导致耗电增加。这也应用于有不同封装的单片机，因为它们可能含有未引出的引脚，这些引脚也必须设为输出或带有上拉电阻的输入。

另外还需注意单片机设为输出的 I/O 引脚上的负载。应将它们设置在有最小拉电流的状态或将它们和其它的 CMOS 输入一样接到没有拉电流的外部电路上。还应注意的是，如果使能配置选项中的 LIRC 振荡器，会导致耗电增加。在空闲模式 1 中，系统时钟开启。若系统时钟来自高速系统振荡器，额外的静态电流也可能会有几百微安。

唤醒

系统进入休眠或空闲模式之后，可以通过以下几种方式唤醒：

- 外部复位
- PA 口下降沿
- 系统中断
- WDT 溢出

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

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

编程注意事项

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

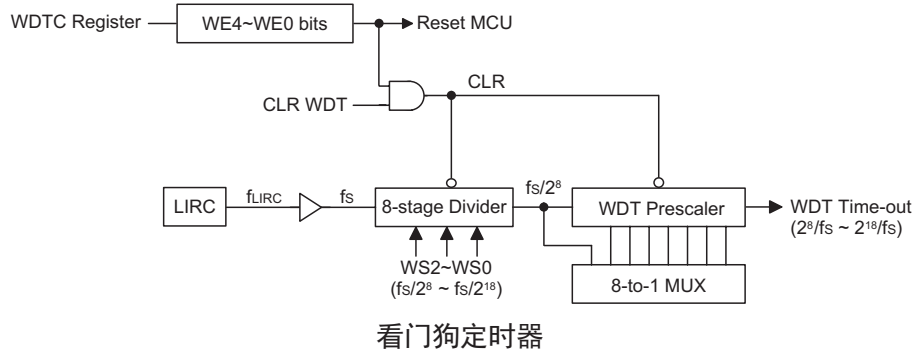
- 若单片机从休眠模式 0 唤醒后进入正常模式，高速系统振荡器需要一个 SST 周期。在 HTO 为“1”后，单片机开始执行首条指令。此时，LIRC 振荡器可能不是稳定的，上电状态可能会发生类似情况，首条指令执行时 LIRC 振荡器还未就绪。
- 若单片机从休眠模式 1 唤醒后进入正常模式，系统时钟源来自 HXT 振荡器且 FSTEN 为“1”，唤醒后，系统时钟可切换至 LIRC 振荡器。
- 一些外围功能，如 WDT 和 TMs 采用系统时钟 f_{SYS} 时，在系统时钟源由 f_H 切换至 f_L 时，以上这些功能的时钟源也要随之改变。
- 当 WDT 时钟源选择为 f_{SUB} 时，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

看门狗定时器

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

看门狗定时器时钟源

WDT 定时器时钟源来自于内部时钟 f_s ，而 f_s 的时钟源由 LIRC 振荡器提供。电压为 5V 时内部振荡器 LIRC 的周期大约为 32kHz。需要注意的是，这个特殊的内部时钟周期随 V_{DD} 、温度和制成的不同而变化。看门狗定时器的时钟源可分频为 $2^8 \sim 2^{18}$ 以提供更大的溢出周期，分频比由 WDTC 寄存器中的 WS2~WS0 位来决定。



看门狗定时器控制寄存器

WDTC 寄存器用于控制 WDT 功能的使能 / 除能及选择溢出周期。寄存器结合配置选项控制看门狗定时器的的工作。

WDTC 寄存器

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

Bit 7~3

WE4~WE0: WDT 软件控制位

如果配置选项中 WDT 设置为“始终使能”

10101 或 01010: 使能

其它值: 单片机复位

如果配置选项中 WDT 设置为“由 WDT 控制寄存器设置”

10101: 除能

01010: 使能

其它值: 单片机复位

当 WE4~WE0 通过环境噪声或软件设置被改动从而单片机复位，复位操作将会在 2~3 个 LIRC 时钟周期后执行同时 CTRL 寄存器 WRF 位将被设置为“1”。

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	—	—	—	—	×	0	0

“×”：未知

- Bit 7 **FSYSON**：空闲模式 f_{SYS} 控制位
 0：除能
 1：使能
- Bit 6~3 未定义，读为“0”
- Bit 2 **LVRF**：LVR 功能复位标志位
 0：未发生
 1：发生
 当某一特定的低电压复位情况发生时，LVRF 位将会设为“1”。该位只能通过应用程序清零。
- Bit 1 **LRF**：LVR 控制寄存器软件复位标志位
 0：未发生
 1：发生
 当 LVRC 寄存器包含任一未定义的 LVR 电压值，LRF 位将会设为“1”，这相当于软件复位功能。该位只能通过应用程序清零。
- Bit 0 **WRF**：WDT 控制寄存器软件复位标志位
 0：未发生
 1：发生
 该位通过 WDT 控制寄存器软件复位功能设为“1”，通过应用程序清零。注意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。通过配置选项选择看门狗定时器的一些选项，如始终使能及清除指令类型。除了配置选项外，WDTC 寄存器中的 WE4~WE0 位也可用来使能 / 除能看门狗定时器以及控制看门狗定时器复位操作。如果配置选项中 WDT 始终使能，此时需设置 WE4~WE0 为“10101”或“01010”才能开启 WDT 功能。若因环境噪声或软件设置修改为其它值，那么单片机将在 2~3 个 LIRC 时钟周期后复位。如果配置选项中设置 WDT 的使能 / 除能由 WDT 控制寄存器控制，那么 WE4~WE0 可以设置 WDT 的操作模式。若这五位被设为“10101”，WDT 除能。若为“01010”WDT 使能。若因环境噪声或软件设置修改为除这两个值以外的其它值，那么单片机将在 2~3 个 LIRC 时钟周期后复位。上电复位后，这些位的初始值为“01010”。

看门狗定时器使能 / 除能控制

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

程序正常运行时，WDT 溢出将导致芯片复位，并置位状态标志位 TO。若系统处于休眠或空闲模式，当 WDT 发生溢出时，状态寄存器中的 TO，程序计数器 PC 和堆栈指针 SP 将被置位。有三种方法可以用来清除 WDT 的内容。第一种是 WDT 复位。将 WE4~WE0 写入除“01010B”和“10101B”外的任意值。第二种是通过 WDT 软件清除指令，而第三种是通过“HALT”指令。

只有一种软件指令用来清除看门狗寄存器，即“CLR WDT”指令。

当设置分频比为 2^{18} 时，溢出周期最大。例如，时钟源为 32 kHz 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。

复位和初始化

复位功能是整个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。

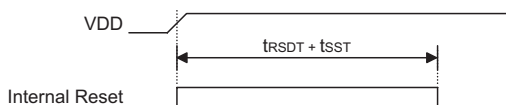
另一种复位为看门狗溢出单片机复位。不同方式的复位操作会对寄存器产生不同的影响。还有一种复位为低电压复位即 LVR 复位，在电源供应电压低于 LVR 设定值时，系统会产生 LVR 复位。

复位功能

包括内部和外部事件触发复位，单片机共有四种复位方式：

上电复位

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



注： $trSDT$ 为上电延迟时间，典型值为 50ms

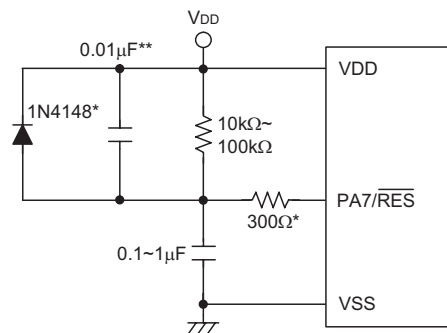
上电复位时序图

RES 引脚复位

由于复位引脚与 PA.7 共用，复位功能必须使用配置选项选择。虽然单片机有一个内部 RC 复位功能，如果电源上升缓慢或上电时电源不稳定，内部 RC 振荡可能导致芯片复位不良，所以推荐使用和 RES 引脚连接的外部 RC 电路，由 RC 电路所造成的时间延迟使得 RES 引脚在电源供应稳定前的一段延长周期内保持在低电平。在这段时间内，单片机的正常操作是被禁止的。RES 引脚达到一定电压值后，再经过延迟时间 t_{RSTD} 单片机可以开始进行正常操作。下图中 SST 是系统延迟周期 System Start-up Timer 的缩写。

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

当系统在较强干扰的场合工作时，建议使用增强型的复位电路，如下图所示。



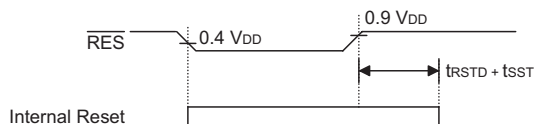
注：“*”表示建议加上此元件以加强静电保护。

“**”表示建议在电源有较强干扰场合加上此元件。

外部 RES 电路

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

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



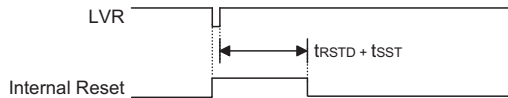
注： t_{RSTD} 为上电延迟时间，典型值为 100ms

RES 复位时序图

低电压复位 – LVR

单片机具有低电压复位电路，用来监测它的电源电压，LVR 功能始终使能，并有特定的 LVR 电压值 V_{LVR} 。例如在更换电池的情况下，单片机供应的电压可能会落在 $0.9V \sim V_{LVR}$ 的范围内，这时 LVR 将会自动复位单片机。同时 CTRL 寄存器中 LVRF 值设置为 1。同时包含以下要求：有效的 LVR 信号，即在 $0.9V \sim V_{LVR}$ 的低电压状态的时间，必须超过交流电气特性中 t_{LVR} 参数的值。如果低电压存在不超过 t_{LVR} 参数的值，则 LVR 将会忽略它且不会执行复位功能。 V_{LVR} 参数值可通过 LVRC 寄存器进行设置。若因环境噪声或软件设置寄存器中的值发生改变，LVR 将在 2~3 个 LIRC 时钟周期后复位单片机。同时 CTRL 寄存器 LVRF 位将被置“1”。上电复位后，寄存器的初始值为“01010101B”。

注意若单片机进入休眠或空闲模式，LVR 功能将自动被除能。



注： t_{RSTD} 为上电延迟时间，典型值为 16.7ms

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 电压选择控制位

01010101: 2.1V

00110011: 2.55V

10011001: 3.15V

10101010: 3.8V

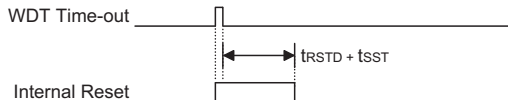
其它值：单片机复位

当相应的低电压出现后，将会产生 MCU 复位。复位操作将会在 2~3 个 LIRC 时钟周期后执行。MCU 复位后寄存器中的值与复位前保持不变。

除上述的四个值后，其它值都会产生 MCU 复位。复位操作将会在 2~3 个 LIRC 时钟周期后执行。MCU 复位后寄存器中的值将会被重置。

正常运行时看门狗溢出复位

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

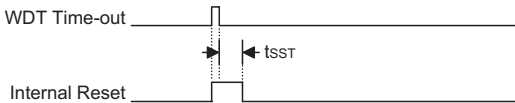


注： t_{RSTD} 为上电延迟时间，典型值为 16.7ms。

正常运行时看门狗溢出时序图

休眠或空闲时看门狗溢出复位

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



注：如果系统时钟源为HIRC时，t_{SST}为15~16个时钟周期。
如果系统时钟源为HXT，则t_{SST}为128个时钟周期。
如果系统时钟源为LIRC，则t_{SST}为1~2个时钟周期。

休眠或空闲时看门狗溢出复位时序图

复位初始状态

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

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

“u”代表不改变

在单片机上电复位之后，各功能单元初始化的情形，列于下表。

项目	复位后情况
程序计数器	清除为零
中断	所有中断被除能
看门狗定时器	WDT清除并重新计数
定时/计数器	所有定时/计数器停止
输入/输出口	I/O口设为输入模式，AN0~AN3作为A/D输入脚。
堆栈指针	堆栈指针指向堆栈顶端

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

寄存器	HT66F016	HT66F017	HT68F016	HT68F017	上电复位	$\overline{\text{RES}}$ 或 LVR 复位	WDT 溢出 (正常模式)	WDT 溢出 (空闲模式)
MP0	●	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
MP1	●	●	●	●	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
BP	●	●	●	●	-----0	-----0	-----0	-----u
ACC	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	●	●	●	●	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	●	●	●	●	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	—	●	—	●	-----xxx	-----uuu	-----uuu	-----uuu
	●	—	●	—	-----xx	-----uu	-----uu	-----uu
STATUS	●	●	●	●	--00 xxxx	--uu uuuu	--1u uuuu	--1l uuuu
SMOD	●	●	●	●	0000 0011	0000 0011	0000 0011	uuuu uuuu
LVDC	●	●	●	●	--00 -000	--00 -000	--00 -000	--uu -uuu
INTEG	●	●	●	●	-----0000	-----0000	-----0000	-----uuuu
INTC0	—	●	—	●	-000 0000	-000 0000	-000 0000	-uuu uuuu
	●	—	●	—	-00 -000	-00 -000	-00 -000	-uu -uuu
INTC1	●	●	—	—	0000 0000	0000 0000	0000 0000	uuuu uuuu
	—	—	●	●	0-00 0-00	0-00 0-00	0-00 0-00	u-u u-u
INTC2	●	●	●	●	--00 -00	--00 -00	--00 -00	--uu -uu
MFI0	—	●	—	●	--00 -00	--00 -00	--00 -00	--uu -uu
MFI1	●	●	●	●	--00 -00	--00 -00	--00 -00	--uu -uu
MFI2	●	●	●	●	--00 -00	--00 -00	--00 -00	--uu -uu
PA	●	●	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	●	●	●	●	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAPU	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAWU	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
PB	●	●	●	●	--11 1111	--11 1111	--11 1111	--uu uuuu
PBC	●	●	●	●	--11 1111	--11 1111	--11 1111	--uu uuuu
PBPU	●	●	●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
PRM	●	●	●	●	0101 0000	0101 0000	0101 0000	uuuu uuuu
TMPC	—	●	—	●	-----00	-----00	-----00	-----uu
	●	—	●	—	-----0-	-----0-	-----0-	-----u-
WDTC	●	●	●	●	0101 0011	0101 0011	0101 0011	uuuu uuuu
TBC	●	●	●	●	0011 -111	0011 -111	0011 -111	uuuu -uuu
EEA	●	●	●	●	--00 0000	--00 0000	--00 0000	--uu uuuu
EED	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
EEC	●	●	●	●	-----0000	-----0000	-----0000	-----uuuu
ADRL (ADRFs=0)	●	●	—	—	xxxx - - - -	xxxx - - - -	xxxx - - - -	uuuu - - - -

寄存器	HT66F016	HT66F017	HT68F016	HT68F017	上电复位	$\overline{\text{RES}}$ 或 LVR 复位	WDT 溢出 (正常模式)	WDT 溢出 (空闲模式)
ADRL (ADRFs=1)	●	●	—	—	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH (ADRFs=0)	●	●	—	—	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH (ADRFs=1)	●	●	—	—	---- xxxx	---- xxxx	---- xxxx	---- uuuu
ADCR0	●	●	—	—	0110 - -00	0110 - -00	0110 - -00	uuuu - -uu
ADCR1	●	●	—	—	00 -0 - 000	00 -0 - 000	00 -0 - 000	uu -u - uuu
ACERL	●	●	—	—	---- 1111	---- 1111	---- 1111	---- uuuu
CPC	●	●	●	●	1000 0 - -1	1000 0 - -1	1000 0 - -1	uuuu u - -u
CTRL	●	●	●	●	0- - - -x00	u- - - -uuu	u- - - -uuu	u- - - -uuu
LVRC	●	●	●	●	0101 0101	0101 0101	0101 0101	uuuu uuuu
TM0C0	—	●	—	●	0000 0- - -	0000 0- - -	0000 0- - -	uuuu u- - -
TM0C1	—	●	—	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0DL	—	●	—	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0DH	—	●	—	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0AL	—	●	—	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0AH	—	●	—	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM0RP	—	●	—	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1C0	●	●	●	●	0000 0- - -	0000 0- - -	0000 0- - -	uuuu u- - -
TM1C1	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1DL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1DH	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1AL	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1AH	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu
TM1RP	●	●	●	●	0000 0000	0000 0000	0000 0000	uuuu uuuu

注：“u”表示不改变
“x”表示未知
“-”表示未定义

输入 / 输出端口

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

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

寄存器名称	位							
	7	6	5	4	3	2	1	0
PA	D7	D6	D5	D4	D3	D2	D1	D0
PAC	D7	D6	D5	D4	D3	D2	D1	D0
PAPU	D7	D6	D5	D4	D3	D2	D1	D0
PAWU	D7	D6	D5	D4	D3	D2	D1	D0
PB	—	—	D5	D4	D3	D2	D1	D0
PBC	—	—	D5	D4	D3	D2	D1	D0
PBPU	—	—	D5	D4	D3	D2	D1	D0
PRM	PRML3	PRML2	PRML1	PRML0	PRMS3	PRMS2	PRMS1	PRMS0

输入 / 输出寄存器列表

上拉电阻

许多产品应用在端口处于输入状态时需要外加一个上拉电阻来实现上拉的功能。为了免去外部上拉电阻，当引脚规划为输入时，可由内部连接到一个上拉电阻。这些上拉电阻可通过寄存器 PAPU 和 PBPU 来设置，它用一个 PMOS 晶体管来实现上拉电阻功能。

PAPU 寄存器

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

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

PBPU 寄存器

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

Bit 7~6 未定义，读为“0”
Bit 5~0 PB 口 bit 5~bit 0 上拉电阻控制位
0: 除能
1: 使能

PA 口唤醒

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

PAWU 寄存器

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

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

输入 / 输出端口控制寄存器

每一个输入 / 输出口都具有各自的控制寄存器，即 PAC 和 PBC，用来控制输入 / 输出状态。从而每个 I/O 引脚都可以通过软件控制，动态的设置为 CMOS 输出或输入。所有的 I/O 端口的引脚都各自对应于 I/O 端口控制的某一位。若 I/O 引脚要实现输入功能，则对应的控制寄存器的位需要设置为“1”。这时程序指令可以直接读取输入脚的逻辑状态。若控制寄存器相应的位被设定为“0”，则此引脚被设置为 CMOS 输出。当引脚设置为输出状态时，程序指令读取的是输出端口寄存器的内容。注，如果对输出口做读取动作时，程序读取到的是内部输出数据锁存器中的状态，而不是输出引脚上实际的逻辑状态。

PAC 寄存器

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

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

PBC 寄存器

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

Bit 7~6 未定义，读为“0”

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

引脚重置功能

引脚的多功能可以增加单片机应用的灵活性。有限的引脚个数将会限制设计者，而引脚的多功能将会解决很多此类问题。每个功能可单独选择所在的引脚，以及一个确定的优先级，使得引脚上多种功能可以同时使用。此外，一些引脚功能可以通过寄存器 PRM 进行设定。大致上讲，模拟信号功能比数字信号功能具有更高的优先级。但是如果在同一外部引脚有两个模拟功能同时使能，那么输入的模拟信号将在内部同时连接到所有的有效的功能模块。

引脚重置寄存器

封装中有限的引脚个数会对某些单片机功能造成影响。然而，引脚功能重置和引脚功能选择，使得小封装单片机具有更多不同的功能。

PRM 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PRML3	PRML2	PRML1	PRML0	PRMS3	PRMS2	PRMS1	PRMS0
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	0	0

Bit 7~4 **PRML3~PRML0:** 引脚重置功能锁存控制位（默认值：0101）
1010: PRM 寄存器写操作使能
其它值: PRM 寄存器写操作除能

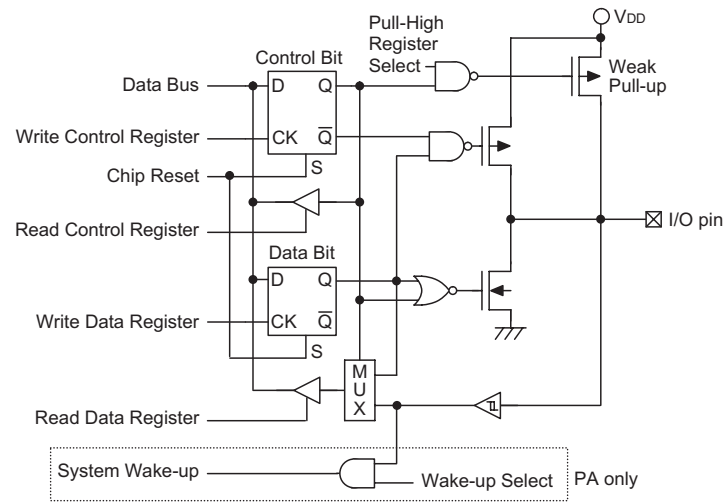
Bit 3 **PRMS3:** INT1 引脚功能重置选择位
0: INT1 on PA5
1: INT1 on PA4

Bit 2 **PRMS2:** INT0/TCK1 引脚重置选择位
0: INT0 on PA3, TCK1 on PA3
1: INT0 on PA2, TCK1 on PA2

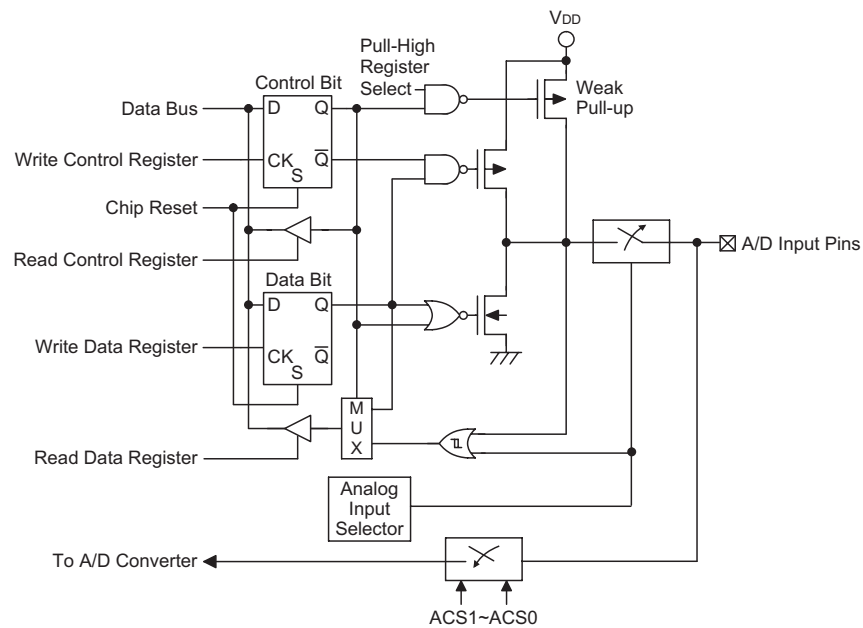
Bit 1~0 **PRMS1~PRMS0:** 引脚重置选择位
For HT66F017, HT68F017
00: TP0 on PA3, TP1/TCK0 on PA4
01: TP0 on PB0, TP1/TCK0 on PB1
10: TP0 on PA5, TP1/TCK0 on PA6
11: TP0 on PA2, TP1/TCK0 on PA7
For HT66F016, HT68F016
00: TP1 on PA4
01: TP1 on PB1
10: TP1 on PA6
11: TP1 on PA7

输入 / 输出引脚结构

下图为输入 / 输出引脚的内部结构图。输入 / 输出引脚的准确逻辑结构图可能与此图不同，这里只是为了方便对 I/O 引脚功能的理解提供的一个参考。图中的引脚共用结构并非针对所有单片机。



通用输入 / 输出端口



A/D 输入 / 输出端口

编程注意事项

在编程中，最先要考虑的是端口的初始化。复位之后，所有的输入 / 输出数据及端口控制寄存器都将被设为逻辑高。所有输入 / 输出引脚默认为输入状态，而其电平则取决于其它相连接电路以及是否选择了上拉电阻。如果端口控制寄存器 PAC 和 PBC，某些引脚位被设定输出状态，这些输出引脚会有初始高电平输出，除非数据寄存器端口 PA 和 PB 在程序中被预先设定。设置哪些引脚是输入及哪些引脚是输出，可通过设置正确的值到适当的端口控制寄存器，或使用指令“SET [m].i”及“CLR [m].i”来设定端口控制寄存器中个别的位。注意，当使用这些位控制指令时，系统即将产生一个读 - 修改 - 写的操作。单片机需要先读入整个端口上的数据，修改个别的位，然后重新把这些数据写入到输出端口。

A/D 转换控制寄存器的上电初始状态默认 PA0~PA3 为模拟信号输入引脚，但 A/D 转换功能并没自动开启。因些需注意若要将 PA0~PA3 用作数字信号输入引脚，或其它功能，需在程序中修改 A/D 转换控制寄存器值以关闭 A/D 功能。另外需注意 A/D 通道使能，内部上拉电阻将自动断开。

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

这里只介绍各种 TM 的共性，更多详细资料请参考简易型和标准型定时器章节。

简介

该系列单片机包含 2 个 TM 分别命名为 TM0 和 TM1。每个 TM 可被划分为一个特定的类型，即简易型 TM (CTM) 或标准型 TM (STM)。虽然性质相似，但不同 TM 特性复杂度不同。本章介绍简易型和标准型 TM 的共性，更多详细资料分别见后面各章。两种类型 TM 的特性和区别见下表。

功能	CTM	STM
定时 / 计数器	√	√
捕捉输入	—	√
比较匹配输出	√	√
PWM 通道数	1	1
单脉冲输出	—	1
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

HT66F017 和 HT68F017 单片机包括一个简易型和一个标准型 TM 单元，分别命名为 TM0 和 TM1，HT66F016 和 HT68F016 单片机仅包含一个标准型 TM，如下表所示。

单片机	TM0	TM1
HT66F016/HT68F016	—	16-bit STM
HT66F017/HT68F017	16-bit CTM	16-bit STM

TM 名称 / 类型参考

TM 操作

两种不同类型的 TM 提供从简单的定时操作到 PWM 信号产生等多种功能。理解 TM 操作的关键是比较 TM 内独立运行的计数器的值与内部比较器的预置值。当计数器的值与比较器的预置值相同时，则比较匹配，TM 中断信号产生，清零计数器并改变 TM 输出引脚的状态。用户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 TM 控制寄存器的 TnCK2~TnCK0 位，选择所需的时钟源。该时钟源来自系统时钟 f_{SYS} 或内部高速时钟 f_H 或 f_{TBC} 时钟源或外部 TCKn 引脚时钟的分频比。注意：设置 TnCK2~TnCK0 为 101，将选择 TM 预设时钟输入，有效切断 TM 时钟源。TCKn 引脚时钟源用于允许外部信号作为 TM 时钟源或用于事件计数。

TM 中断

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

TM 外部引脚

无论哪种类型的 TM，都有一个 TM 输入引脚 TCKn。通过设置 TMnCO 寄存器中的 TnCK2~TnCK0 位，选择 TM 功能并将该引脚作为 TM 时钟源输入脚。外部时钟源可通过该引脚来驱动内部 TM。外部 TM 输入脚也与其它功能共用，但是，如果设置适当值给 TnCK2~TnCK0，该引脚会连接到内部 TM。TM 引脚可选择上升沿有效或下降沿有效。

每个 TM 有一个输出引脚 TPn。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 TPn 输出引脚也被 TM 用来产生 PWM 输出波形。当 TM 工作在捕捉输入模式，TPn 引脚也可作为输入脚。因为 TM 输出引脚与其它功能共用，TM 输出功能需要通过内部 TM 开启 / 关闭控制，操作模式选择，以及输入 / 输出控制寄存器的相应位的设置来使能 / 除能。当相应 TM 配置设置 TPn 用作外部输出脚，相应的引脚将作为外部 TM 输出引脚。当相应 TM 配置设置 TPn 为输入引脚，TPn 可作为输入引脚，且输入的信号来自于外部信号或其它共用功能的输出信号。当相应 TM 配置设置 TPn 不被使用，这些引脚将用作其它共用功能使用。每个单片机和不同类型 TM 中输出引脚的具体情况详见下表。

单片机	CTM	STM	寄存器
HT66F016/HT68F016	—	TP1	TMPC
HT66F017/HT68F017	TP0	TP1	TMPC

TM 输出引脚

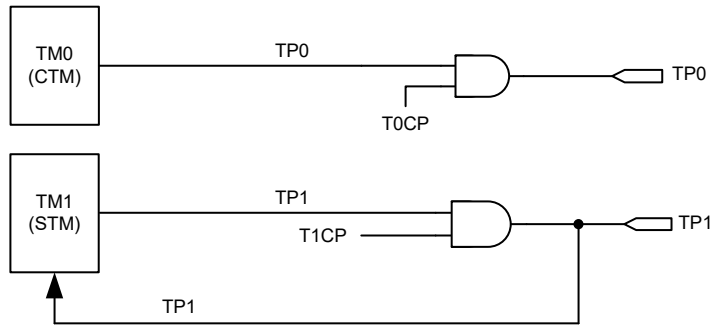
TM 输入 / 输出引脚控制寄存器

通过设置一个与 TM 输入 / 输出引脚相关的寄存器的一位，选择作为 TM 输入 / 输出功能或其它共用功能。设定为高时，相关引脚用作 TM 输入 / 输出，清零时将保持原来的功能。

TMPC 寄存器

单片机	寄存器	位							
		7	6	5	4	3	2	1	0
HT66F016 HT68F016	TMPC	—	—	—	—	—	—	T1CP	—
HT66F017 HT68F017	TMPC	—	—	—	—	—	—	T1CP	T0CP

TM 输入 / 输出引脚控制寄存器列表



TM 功能引脚控制方框图

TMPC 寄存器

• HT66F016/HT68F016

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

Bit 7~2 未定义，读为“0”

Bit 1 **T1CP**: TP1 引脚控制位
0: 除能
1: 使能

Bit 0 未定义，读为“0”

● HT66F017/HT68F017

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

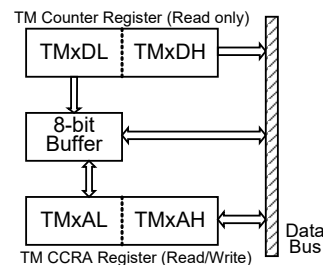
Bit 7~2 未定义，读为“0”

Bit 1 **T1CP**: TP1 引脚控制位
0: 除能
1: 使能

Bit 0 **T0CP**: TP0 引脚控制位
0: 除能
1: 使能

编程注意事项

TM 计数寄存器和捕捉 / 比较寄存器 CCRA 为 16-bit 的寄存器，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。CCRA 寄存器的操作如下所示，同时因为这些成对的寄存器用上述所说的特殊的方式进行读写。因此建议使用“MOV”指令用如下方式对 CCRA 低字节寄存器 TMxAL 操作。若不用下述过程进行操作，将会导致不可预知的结果。



读写流程如下步骤所示：

● 写数据至 CCRA

- ◆ 步骤 1. 写数据至低字节寄存器 TMxAL
 - 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 TMxAH
 - 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。

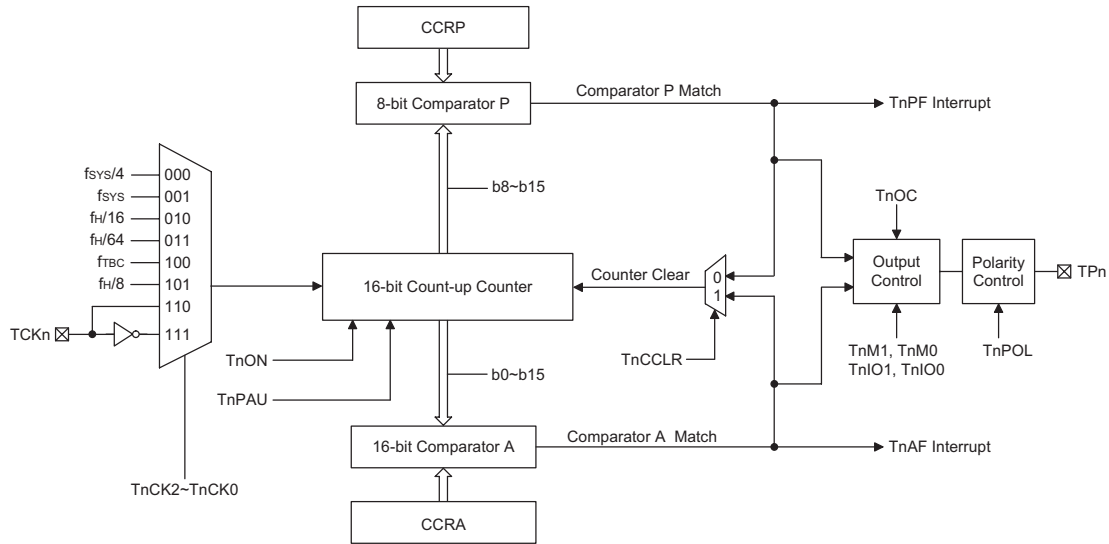
● 由计数器寄存器和 CCRA 中读取数据

- ◆ 步骤 1. 由高字节寄存器 TMxDH, TMxAH 读取数据
 - 注意，此时高字节寄存器中的数据直接读取，同时由低字节寄存器读取的数据锁存至 8-bit 缓存器中。
- ◆ 步骤 2. 由低字节寄存器 TMxDL、TMxAL 读取数据
 - 注意，此时读取 8-bit 缓存器中的数据。

简易型 – CTM

虽然简易型 TM 是这两种 TM 类型中最简单的形式，但仍然包括三种工作模式，即比较匹配输出，定时 / 事件计数器和 PWM 输出模式。简易型 TM 也由外部输入脚控制并驱动一个外部输出脚。

CTM	名称	TM 编号	TM 输入引脚	TM 输出引脚
HT66F017/HT68F017	16-bit CTM	0	TCK0	TP0



简易型 TM 方框图

简易型 TM 操作

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

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

简易型 TM 寄存器介绍

简易型 TM 的所有操作由一组寄存器控制。每组中包含一对只读寄存器用来存放 16 位计数器的值，一对读 / 写寄存器存放 16 位 CCRA 的值，一个寄存器用来存放内部 8 位 CCRP 的值。剩下两个控制寄存器设置不同的操作和控制模式。

寄存器名称	位							
	7	6	5	4	3	2	1	0
TM0C0	T0PAU	T0CK2	T0CK1	T0CK0	T0ON	—	—	—
TM0C1	T0M1	T0M0	T0IO1	T0IO0	T0OC	T0POL	T0DPX	T0CCLR
TM0DL	D7	D6	D5	D4	D3	D2	D1	D0
TM0DH	D15	D14	D13	D12	D11	D10	D9	D8
TM0AL	D7	D6	D5	D4	D3	D2	D1	D0
TM0AH	D15	D14	D13	D12	D11	D10	D9	D8
TM0RP	T0RP7	T0RP6	T0RP5	T0RP4	T0RP3	T0RP2	T0RP1	T0RP0

简易型 TM 寄存器列表

TM0DL 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 **TM0DL**: TM0 计数器低字节寄存器 bit 7~bit 0
TM0 16-bit 计数器 bit 7~bit 0

TM0DH 寄存器

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

Bit 7~0 **TM0DH**: TM0 计数器高字节寄存器 bit 7~bit 0
TM0 16-bit 计数器 bit 15~bit 8

TM0AL 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 **TM0AL**: TM0 CCRA 低字节寄存器 bit 7~bit 0
TM0 16-bit CCRA bit 7~bit 0

TM0AH 寄存器

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

Bit 7~0 **TM0AH**: TM0 CCRA 高字节寄存器 bit 7~bit 0
TM0 16-bit CCRA bit 15~bit 8

TM0C0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	T0PAU	T0CK2	T0CK1	T0CK0	T0ON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7 **T0PAU**: TM0 计数器暂停控制位
0: 运行
1: 暂停

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

Bit 6~4 **T0CK2~T0CK0**: 选择 TM0 计数时钟位
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_{H}/16$
011: $f_{H}/64$
100: f_{TBC}
101: $f_{H}/8$
110: TCK0 上升沿时钟
111: TCK0 下降沿时钟

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

Bit 3 **T0ON**: TM0 计数器 On/Off 控制位
0: Off
1: On

此位控制 TM 的总开关功能。设置此位为高则使能计数器使其运行, 清零此位则除能 TM。清零此位将停止计数器并关闭 TM 减少耗电。当此位经由低到高转换时, 内部计数器将保持其剩余值。

若 TM 处于比较匹配输出模式时 (通过 TnOC 位指定), 当 TnON 位经由低到高的转换时, TM 输出脚将重置其初始值。

Bit 2~0 未定义, 读为 0

TM0C1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	T0M1	T0M0	T0IO1	T0IO0	T0OC	T0POL	T0DPX	T0CCLR
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 **T0M1~T0M0**: 选择 TM0 工作模式位

- 00: 比较匹配输出模式
- 01: 未定义
- 10: PWM 模式
- 11: 定时 / 计数器模式

这两位设置 TM 需要的工作模式。为了确保操作可靠，TM 应在 T0M1 和 T0M0 位有任何改变前先关掉。在定时 / 计数器模式，TM 输出脚控制必须除能。

Bit 5~4 **T0IO1~T0IO0**: 选择 TP0 输出功能位

比较匹配输出模式

- 00: 无变化
- 01: 输出低
- 10: 输出高
- 11: 输出翻转

PWM 模式

- 00: 强制无效状态
- 01: 强制有效状态
- 10: PWM 输出
- 11: 未定义

定时 / 计数器模式
未使用

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

在比较匹配输出模式下，T0IO1 和 T0IO0 位决定当比较器 A 比较匹配输出发生时 TM 输出脚如何改变状态。当比较器 A 比较匹配输出发生时 TM 输出脚能设为切换高、切换低或翻转当前状态。若此两位同时为 0 时，这个输出将不会改变。TM 输出脚的初始值通过 TM0C1 寄存器的 T0OC 位设置取得。注意，由 T0IO1 和 T0IO0 位得到的输出电平必须与通过 T0OC 位设置的初始值不同，否则当比较匹配发生时，TM 输出脚将不会发生变化。在 TM 输出脚改变状态后，通过 T0ON 位由低到高电平的转换复位至初始值。

在 PWM 模式，T0IO1 和 T0IO0 用于决定比较匹配条件发生时怎样改变 TM 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 TM0 关闭时改变 T0IO1 和 T0IO0 位的值是很有必要的。若在 TM 运行时改变 T0IO1 和 T0IO0 的值，PWM 输出的值是无法预料的。

Bit 3 **T0OC**: TP0 输出控制位

比较匹配输出模式

- 0: 初始低
- 1: 初始高

PWM 模式

- 0: 低有效
- 1: 高有效

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

- Bit 2 **T0POL:** TP0 输出极性控制位
0: 同相
1: 反相
此位控制 TP0 输出脚的极性。此位为高时 TM 输出脚反相，为低时 TM 输出脚同相。若 TM 处于定时 / 计数器模式时其不受影响。
- Bit 1 **T0DPX:** TM0 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 **T0CCLR:** 选择 TM0 计数器清零条件位
0: TM0 比较器 P 匹配
1: TM0 比较器 A 匹配
此位用于选择清除计数器的方法。简易型 TM 包括两个比较器 - 比较器 A 和比较器 P。这两个比较器每个都可以用作清除内部计数器。T0CCLR 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。T0CCLR 位在 PWM 模式时未使用。

TM0RP 寄存器

Bit	7	6	5	4	3	2	1	0
Name	T0RP7	T0RP6	T0RP5	T0RP4	T0RP3	T0RP2	T0RP1	T0RP0
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 **T0RP7~T0RP0:** TM0 CCRP 寄存器 bit 7~bit 0
TM0 CCRP 8 位寄存器，与 TM0 计数器 bit 15~bit 8 比较。比较器 P 匹配周期
0: 65536 个 TM0 时钟周期
1~255: $256 \times (1 \sim 255)$ 个 TM0 时钟周期
此八位设定内部 CCRP 8-bit 寄存器的值，然后与内部计数器的高八位进行比较。如果 T0CCLR 位设为 0 时，比较结果为 0 并清除内部计数器。T0CCLR 位设为低，CCRP 比较匹配结果将重置内部计数器。由于 CCRP 只与计数器高八位比较，比较结果是 256 时钟周期的倍数。CCRP 被清零时，实际上会使得计数器在最大值溢出。

简易型 TM 工作模式

简易型 TM 有三种工作模式，即比较匹配输出模式，PWM 模式或定时 / 计数器模式。通过设置 TMnC1 寄存器的 TnM1 和 TnM0 位选择任意工作模式。

比较匹配输出模式

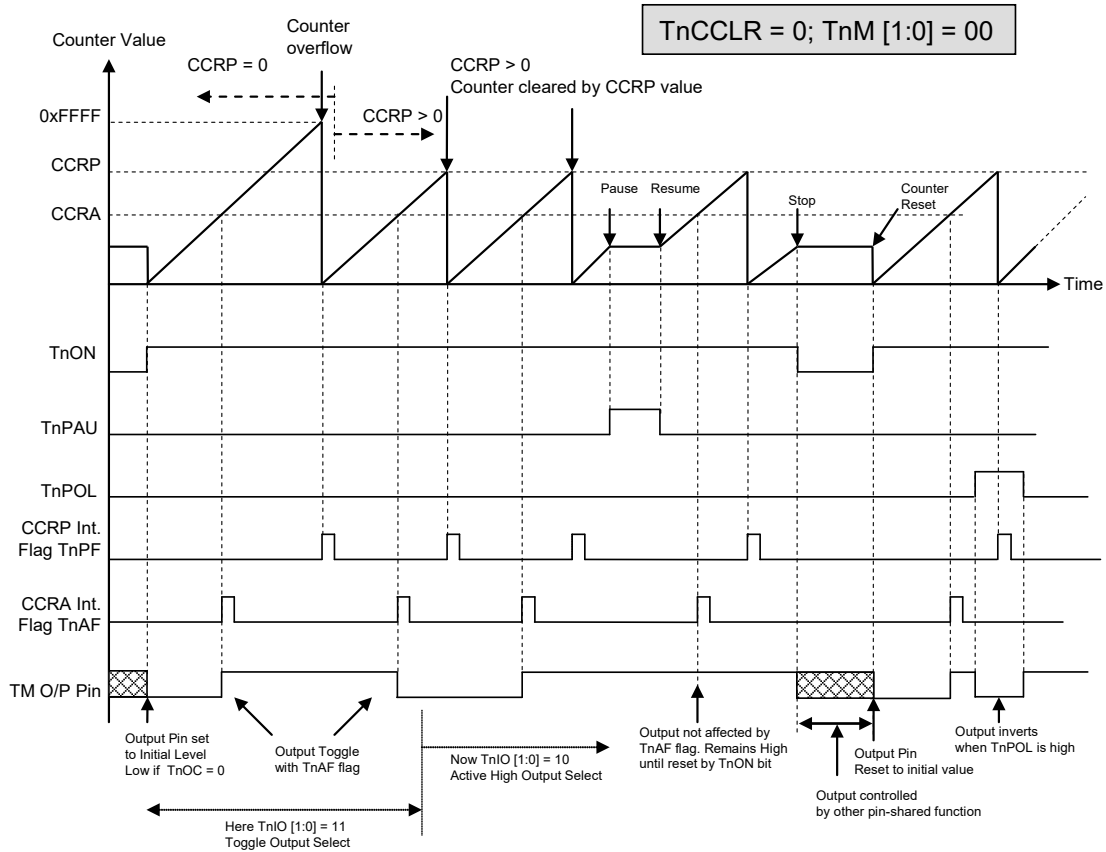
为使 TM 工作在此模式，TMnC1 寄存器中的 TnM1 和 TnM0 位需要设置为“00”。当工作在该模式，一旦计数器使能并开始计数，有三种方法来清零，分别是：计数器溢出，比较器 A 比较匹配发生和比较器 P 比较匹配发生。当 TnCCLR 位为低，有两种方法清除计数器。一种是比较器 P 比较匹配发生，另一种是 CCRP 所有位设置为零并使得计数器溢出。此时，比较器 A 和比较器 P 的请求标志位 TnAF 和 TnPF 将分别置起。

如果 TMnC1 寄存器的 TnCCLR 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 TnAF 中断请求标志产生。所以当 TnCCLR 为高时，不产生 TnPF 中断请求标志。如果 CCRA 被清零，当计数达到最大值 FFFH 时，计数器溢出，而此时不产生 TnAF 请求标志。

正如该模式名所言，当比较匹配发生后，TM 输出脚状态改变。当比较器 A 比较匹配发生后 TnAF 标志产生时，TM 输出脚状态改变。比较器 P 比较匹配发生时产生的 TnPF 标志不影响 TM 输出脚。TM 输出脚状态改变方式由 TMnC1 寄存器中 TnIO1 和 TnIO0 位决定。当比较器 A 比较匹配发生时，TnIO1 和 TnIO0 位决定 TM 输出脚输出高，低或翻转当前状态。TM 输出脚初始值，既可以通过 TnON 位由低到高电平的变化设置，也可以由 TnOC 位设置。注意，若 TnIO1 和 TnIO0 位同时为 0 时，引脚输出不变。

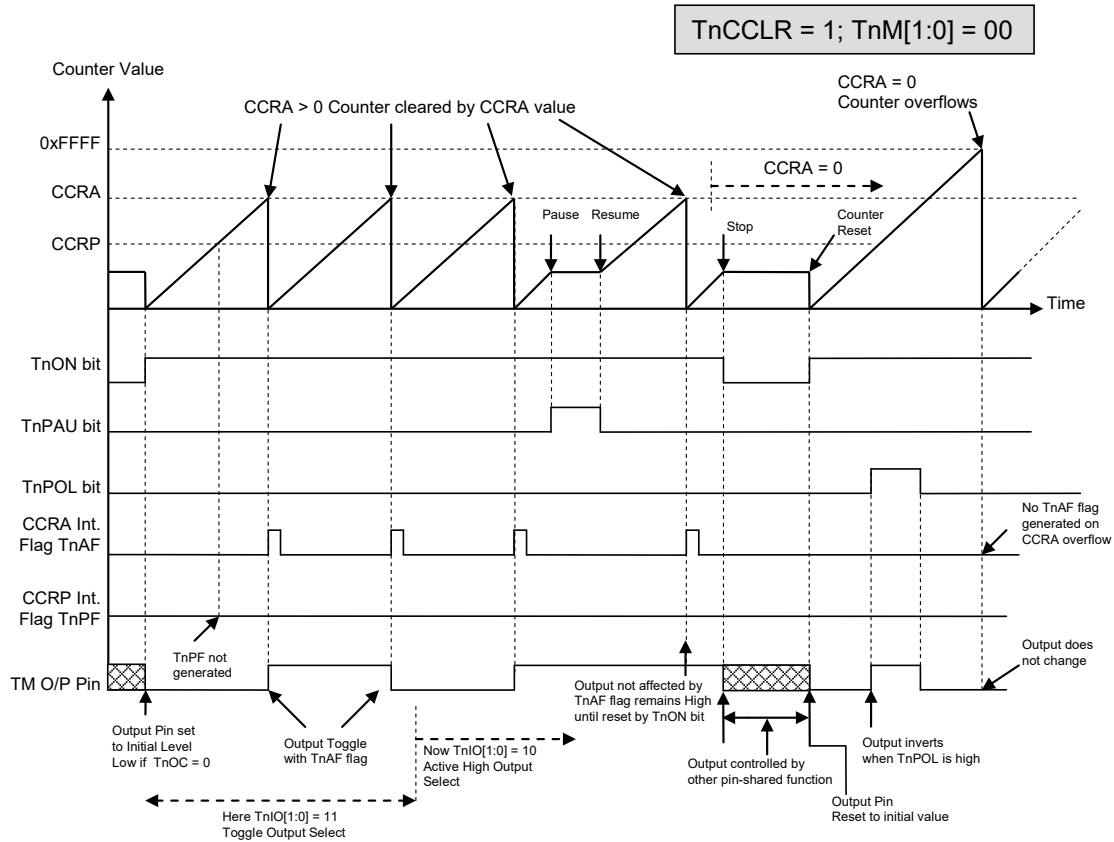
定时 / 计数器模式

为使 TM 工作在此模式，TMnC1 寄存器中的 TnM1 和 TnM0 位需要设置为“11”。定时 / 计数器模式与比较输出模式操作方式相同，并产生同样的中断请求标志。不同的是，在定时 / 计数器模式下 TM 输出脚未使用。因此，比较匹配输出模式中的描述和时序图可以适用于此功能。该模式中未使用的 TM 输出脚用作普通 I/O 脚或其它功能。



比较匹配输出模式 – TnCCLR=0

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



比较匹配输出模式 – TnCCLR=1

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

PWM 输出模式

为使 TM 工作在此模式，TMnC1 寄存器中的 TnM1 和 TnM0 位需要设置为“10”。TM 的 PWM 功能在马达控制，加热控制，照明控制等方面十分有用。给 TM 输出脚提供一个频率固定但占空比可调的信号，将产生一个有效值等于 DC 均方根的 AC 方波。

由于 PWM 波形的周期和占空比可调，其波形的选择就较为灵活。在 PWM 模式中，TnCCLR 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 TMnC1 寄存器的 TnDPX 位。所以 PWM 波形由 CCRA 和 CCRP 寄存器共同决定。

当比较器 A 或比较器 P 比较匹配发生时，将产生 CCRA 或 CCRP 中断标志。TMnC1 寄存器中的 TnOC 位决定 PWM 波形的极性，TnIO1 和 TnIO0 位使能 PWM 输出或将 TM 输出脚置为逻辑高或逻辑低。TnPOL 位对 PWM 输出波形的极性取反。

•CTM, PWM 模式，边沿对齐模式，TnDPX=0

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

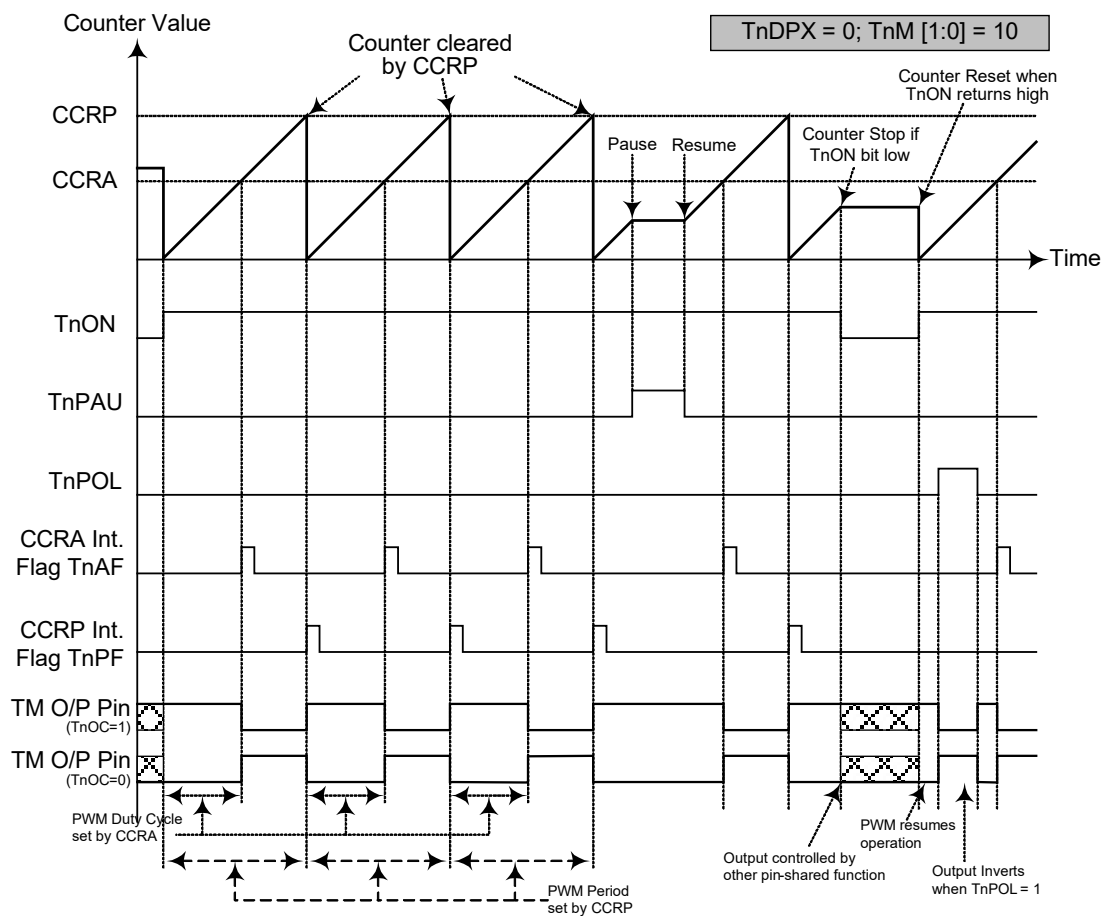
若 $f_{SYS}=16\text{MHz}$ ，TM 时钟源选择 $f_{SYS}/4$ ，CCRP=2，CCRA=128，CTM PWM 输出频率 $=(f_{SYS}/4)/=f_{SYS}/2048=7.8125\text{kHz}$ ， $duty=128/(2\times 256)=25\%$ 。

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

•CTM, PWM 模式，边沿对齐模式，TnDPX=1

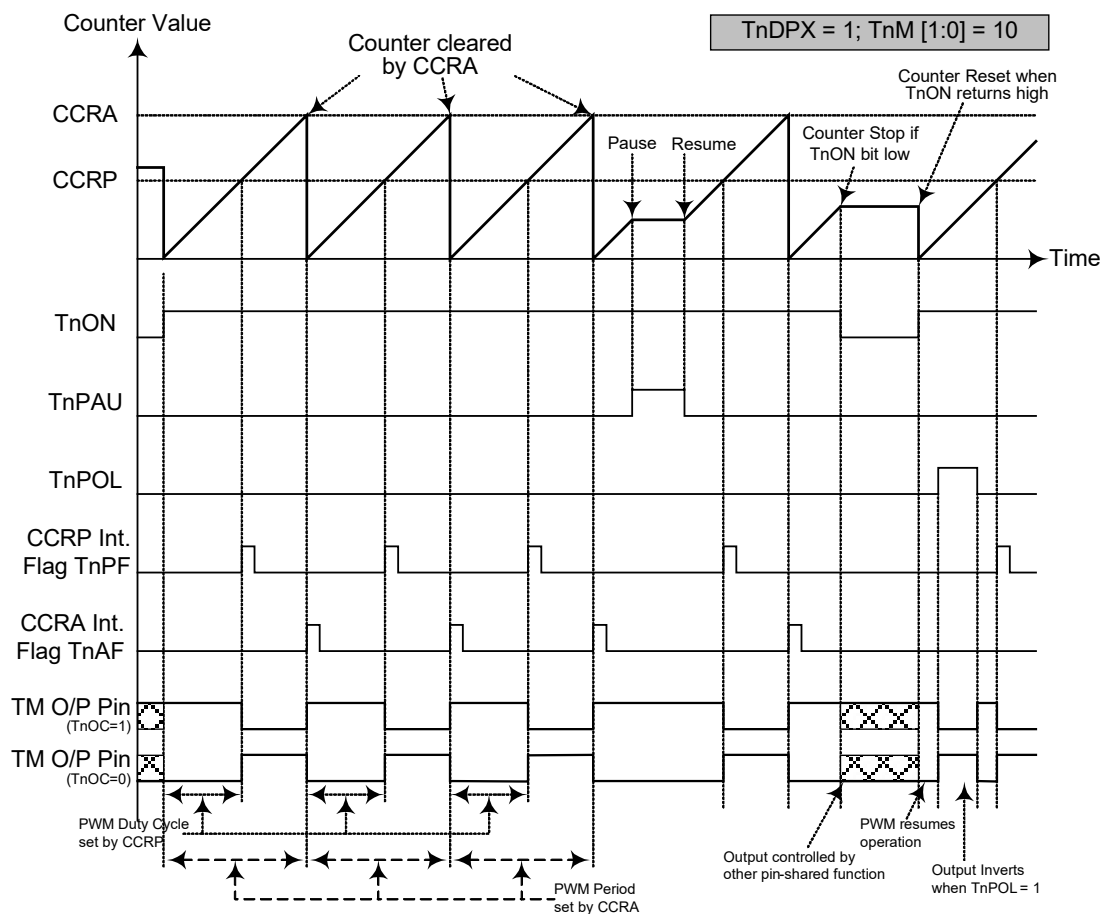
CCRP	1~255	0
Period	CCRA	
Duty	CCRP×256	65536

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



PWM 输出模式 – TnDPX=0

- 注：1. TnDPX=0，CCRP 清除计数器
2. 计数器清零并设置 PWM 周期
3. 当 TnIO1，TnIO0=00 或 01，PWM 功能不变
4. TnCCLR 位不影响 PWM 操作



PWM 输出模式 – TnDPX=1

- 注：1. TnDPX=1，CCRA 清除计数器
2. 计数器清零并设置 PWM 周期
3. 当 TnIO1，TnIO0=00 或 01，PWM 功能不变
4. TnCCLR 位不影响 PWM 操作

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 也由外部输入脚控制并驱动一个外部输出脚。

名称	TM 编号	TM 输入引脚	TM 输出引脚
16-bit STM	1	TCK1	TP1

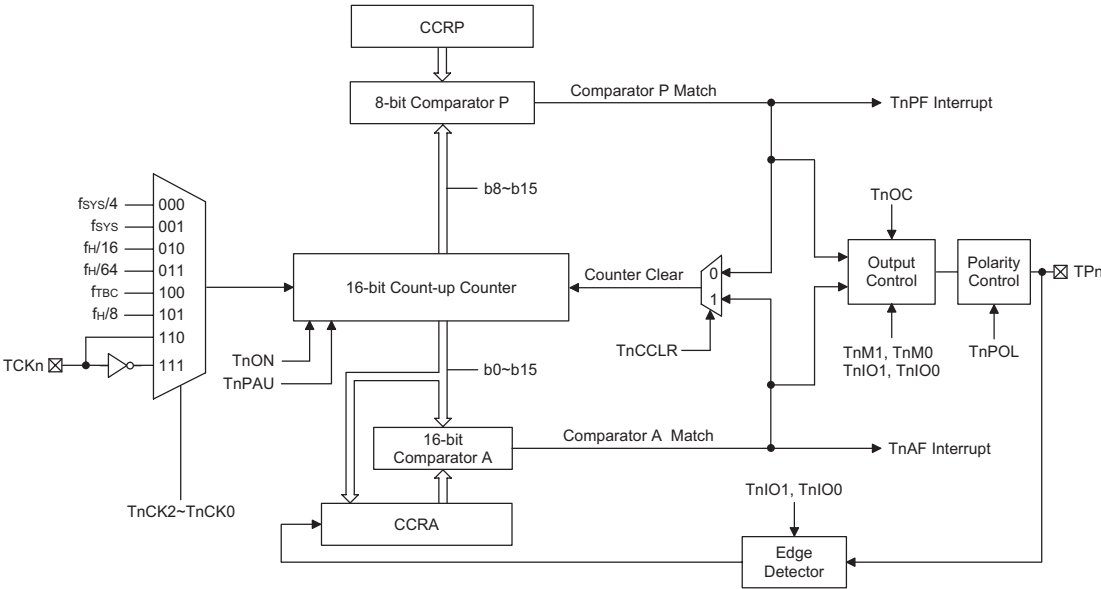
标准型 TM 操作

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

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

标准型 TM 寄存器介绍

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



标准型 TM 框图

寄存器名称	位							
	7	6	5	4	3	2	1	0
TM1C0	T1PAU	T1CK2	T1CK1	T1CK0	T1ON	—	—	—
TM1C1	T1M1	T1M0	T1IO1	T1IO0	T1OC	T1POL	T1DPX	T1CCLR
TM1DL	D7	D6	D5	D4	D3	D2	D1	D0
TM1DH	D15	D14	D13	D12	D11	D10	D9	D8
TM1AL	D7	D6	D5	D4	D3	D2	D1	D0
TM1AH	D15	D14	D13	D12	D11	D10	D9	D8
TM1RP	T1RP7	T1RP6	T1RP5	T1RP4	T1RP3	T1RP2	T1RP1	T1RP0

标准型 STM 寄存器列表

TM1C0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	T1PAU	T1CK2	T1CK1	T1CK0	T1ON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

- Bit 7 T1PAU:** TM1 计数器暂停控制位
0: 运行
1: 暂停
通过设置此位为高可使计数器暂停，清零此位恢复正常计数器操作。当处于暂停条件时，TM 保持上电状态并继续耗电。当此位由低到高转换时，计数器将保留其剩余值，直到此位再次改变为低电平，并从此值开始继续计数。
- Bit 6~4 T1CK2~T1CK0:** 选择 TM1 计数时钟位
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{TBC}
101: $f_H/8$
110: TCK1 上升沿时钟
111: TCK1 下降沿时钟
此三位用于选择 TM 的时钟源。外部引脚时钟源能被选择在上升沿或下降沿有效。 f_{SYS} 是系统时钟， f_H 和 f_{TBC} 是其它的内部时钟源，细节方面请参考振荡器章节。
- Bit 3 T1ON:** TM1 计数器 On/Off 控制位
0: Off
1: On
此位控制 TM 的总开关功能。设置此位为高则使能计数器使其运行，清零此位则除能 TM。清零此位将停止计数器并关闭 TM 减少耗电。当此位经由低到高转换时，内部计数器将保持其剩余值，直到此位再次改变为高电平。
若 TM 处于比较匹配输出模式时 (通过 T1OC 位指定)，当 T1ON 位经由低到高的转换时，TM 输出脚将重置其初始值。
- Bit 2~0** 未定义，读为 “0”

TM1C1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	T1M1	T1M0	T1IO1	T1IO0	T1OC	T1POL	T1DPX	T1CCLR
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 **T1M1~T1M0**: 选择 TM1 工作模式位

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

这两位设置 TM 需要的工作模式。为了确保操作可靠, TM 应在 T1M1 和 T1M0 位有任何改变前先关掉。在定时 / 计数器模式, TM 输出脚控制必须除能。

Bit 5~4 **T1IO1~T1IO0**: 选择 TP1 输出功能位

比较匹配输出模式

- 00: 无变化
- 01: 输出低
- 10: 输出高
- 11: 输出翻转

PWM 模式 / 单脉冲输出模式

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

捕捉输入模式

- 00: 在 TP1 上升沿输入捕捉
- 01: 在 TP1 下降沿输入捕捉
- 10: 在 TP1 双沿输入捕捉
- 11: 输入捕捉除能

定时 / 计数器模式

未使用

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

在比较匹配输出模式下, T1IO1 和 T1IO0 位决定当从比较器 A 比较匹配输出发生时 TM 输出脚如何改变状态。当从比较器 A 比较匹配输出发生时 TM 输出脚能设为切换高、切换低或翻转当前状态。若此两位同时为 0 时, 这个输出将不会改变。TM 输出脚的初始值通过 TM1C1 寄存器的 T1OC 位设置取得。注意, 由 T1IO1 和 T1IO0 位得到的输出电平必须与通过 T1OC 位设置的初始值不同, 否则当比较匹配发生时, TM 输出脚将不会发生变化。在 TM 输出脚改变状态后, 通过 T1ON 位由低到高电平的转换复位至初始值。

在 PWM 模式, T1IO1 和 T1IO0 用于决定比较匹配条件发生时怎样改变 TM 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 TM1 关闭时改变 T1IO1 和 T1IO0 位的值是很有必要的。若在 TM 运行时改变 T1IO1 和 T1IO0 的值, PWM 输出的值是无法预料的。

Bit 3 **T1OC**: TP1 输出控制位

比较匹配输出模式

- 0: 初始低
- 1: 初始高

PWM 模式 / 单脉冲输出模式

- 0: 低有效
- 1: 高有效

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

- Bit 2 **T1POL:** TP1 输出极性控制位
0: 同相
1: 反相
此位控制 TP1 输出脚的极性。此位为高时 TM 输出脚反相，为低时 TM 输出脚同相。若 TM 处于定时 / 计数器模式时其不受影响。
- Bit 1 **T1DPX:** TM1 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 **T1CCLR:** 选择 TM1 计数器清零条件位
0: TM1 比较器 P 匹配
1: TM1 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。T1CCLR 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。T1CCLR 位在 PWM，单脉冲或输入捕捉模式时未使用。

TM1DL 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 **TM1DL:** TM1 计数器低字节寄存器 bit 7~bit 0
TM1 16-bit 计数器 bit 7~bit 0

TM1DH 寄存器

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

- Bit 7~0 **TM1DH:** TM1 计数器高字节寄存器 bit 7~bit 0
TM1 16-bit 计数器 bit 15~bit 8

TM1AL 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 **TM1AL:** TM1 CCRA 低字节寄存器 bit 7~bit 0
TM1 16-bit CCRA bit 7~bit 0

TM1AH 寄存器

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

Bit 7~0 **TM1AH:** TM1 CCRA 高字节寄存器 bit 7~bit 0
TM1 16-bit CCRA bit 15~bit 8

TM1RP 寄存器

Bit	7	6	5	4	3	2	1	0
Name	T1RP7	T1RP6	T1RP5	T1RP4	T1RP3	T1RP2	T1RP1	T1RP0
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 **T1RP7~T1RP0:** TM1 CCRP 寄存器 bit 7~bit 0
TM1 CCRP 8 位寄存器，与 TM1 计数器 bit 15~bit 8 比较。比较器 P 匹配周期
0: 65536 个 TM1 时钟周期
1~255: $256 \times (1 \sim 255)$ 个 TM1 时钟周期
此八位设定内部 CCRP 8-bit 寄存器的值，然后与内部计数器的高八位进行比较。
如果 TICCLR 位设为 0 时，比较结果为 0 并清除内部计数器。TICCLR 位设为低，
CCRP 比较匹配结果将重置内部计数器。由于 CCRP 只与计数器高八位比较，
比较结果是 256 时钟周期的倍数。CCRP 被清零时，实际上会使得计数器在最大
值溢出。

标准型 TM 工作模式

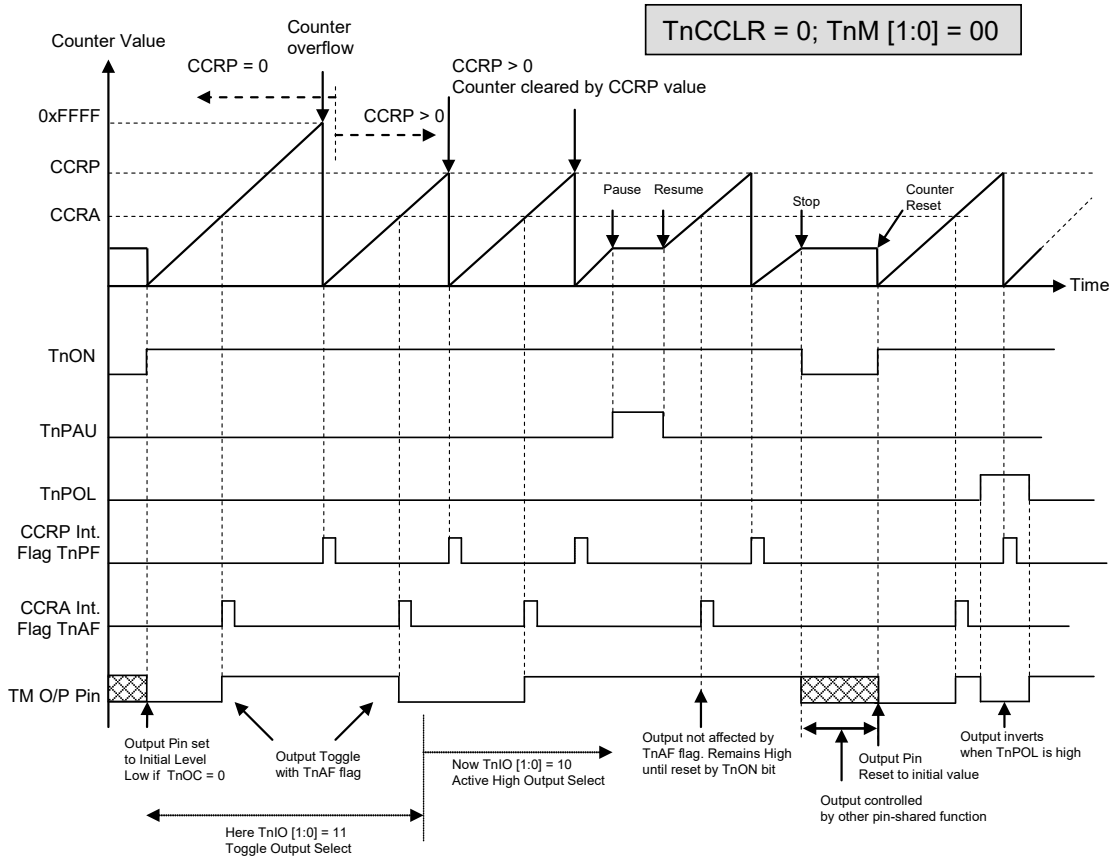
标准型 TM 有五种工作模式，即比较匹配输出模式，PWM 输出模式，单脉冲输出模式，捕捉输入模式和定时 / 计数器模式。通过设置 TMnC1 寄存器的 TnM1 和 TnM0 位选择任意模式。

比较匹配输出模式

为使 TM 工作在此模式，TMnC1 寄存器中的 TnM1 和 TnM0 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 TnCCLR 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 TnAF 和 TnPF 将分别置位。

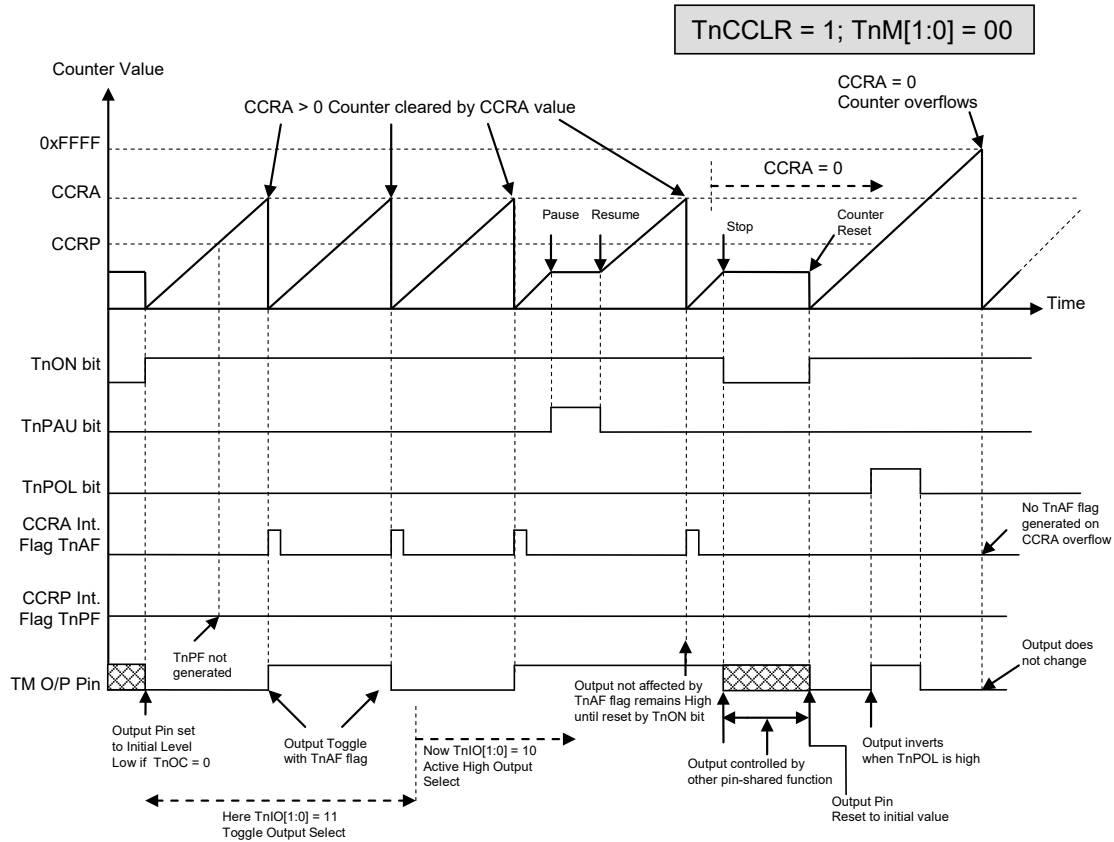
如果 TMnC1 寄存器的 TnCCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅产生 TnAF 中断请求标志。所以当 TnCCLR 为高时，不会产生 TnPF 中断请求标志。在比较匹配输出模式下，CCRA 不能设为“0”。

正如该模式名所言，当比较匹配发生后，TM 输出脚状态改变。当比较器 A 比较匹配发生后 TnAF 标志产生时，TM 输出脚状态改变。比较器 P 比较匹配发生时产生的 TnPF 标志不影响 TM 输出脚。TM 输出脚状态改变方式由 TMnC1 寄存器中 TnIO1 和 TnIO0 位决定。当比较器 A 比较匹配发生时，TnIO1 和 TnIO0 位决定 TM 输出脚输出高，低或翻转当前状态。TM 输出脚初始值，既可以通过 TnON 位由低到高电平的变化设置，也可以由 TnOC 位设置。注意，若 TnIO1 和 TnIO0 位同时为 0 时，引脚输出不变。



比较匹配输出模式 – TnCCLR=0

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



比较匹配输出模式 – TnCCLR=1

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

定时 / 计数器模式

为使 TM 工作在此模式，TMnC1 寄存器中的 TnM1 和 TnM0 位需要设置为“11”。定时 / 计数器模式与比较输出模式操作方式相同，并产生同样的中断请求标志。不同的是，在定时 / 计数器模式下 TM 输出脚未使用。因此，比较匹配输出模式中的描述和时序图可以适用于此功能。该模式中未使用的 TM 输出脚用作普通 I/O 脚或其它功能。

PWM 输出模式

为使 TM 工作在此模式，TMnC1 寄存器中的 TnM1 和 TnM0 位需要设置为“10”。TM 的 PWM 功能在马达控制，加热控制，照明控制等方面十分有用。给 TM 输出脚提供一个频率固定但占空比可调的信号，将产生一个有效值等于 DC 均方根的 AC 方波。

由于 PWM 波形的周期和占空比可调，其波形的选择就较为灵活。在 PWM 模式中，TnCCLR 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 TMnC1 寄存器的 TnDPX 位。所以 PWM 波形由 CCRA 和 CCRP 寄存器共同决定。

当比较器 A 或比较器 P 比较匹配发生时，将产生 CCRA 或 CCRP 中断标志。TMnC1 寄存器中的 TnOC 位决定 PWM 波形的极性，TnIO1 和 TnIO0 位使能 PWM 输出或将 TM 输出脚置为逻辑高或逻辑低。TnPOL 位对 PWM 输出波形的极性取反。

● 16-bit STM, PWM 模式，边沿对齐模式，TnDPX=0

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

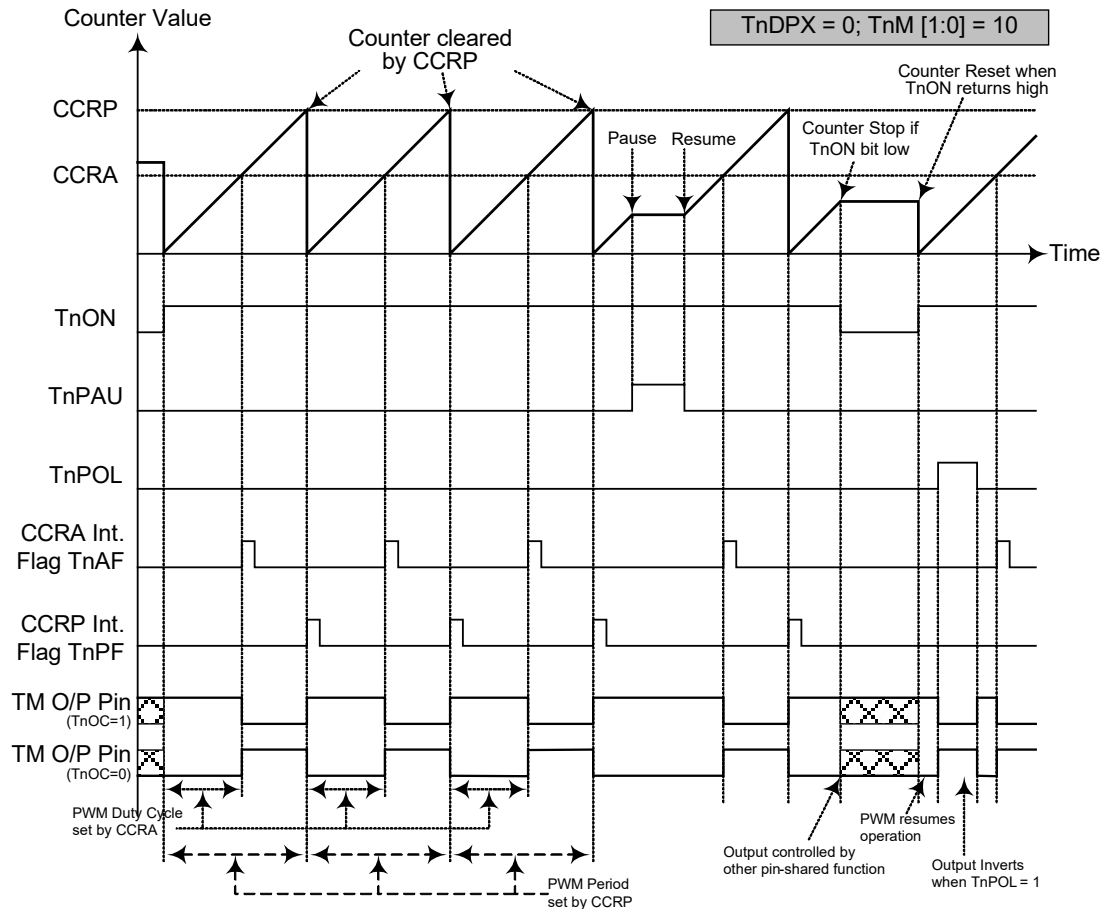
若 $f_{sys}=16\text{MHz}$ ，TM 时钟源选择 $f_{sys}/4$ ，CCRP=2，CCRA=128，STM PWM 输出频率 $= (f_{sys}/4)/(2 \times 256) = f_{sys}/2048 = 7.8125\text{kHz}$ ，duty=128/(2×256)=25%。

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

● 16-bit STM, PWM 模式，边沿对齐模式，TnDPX=1

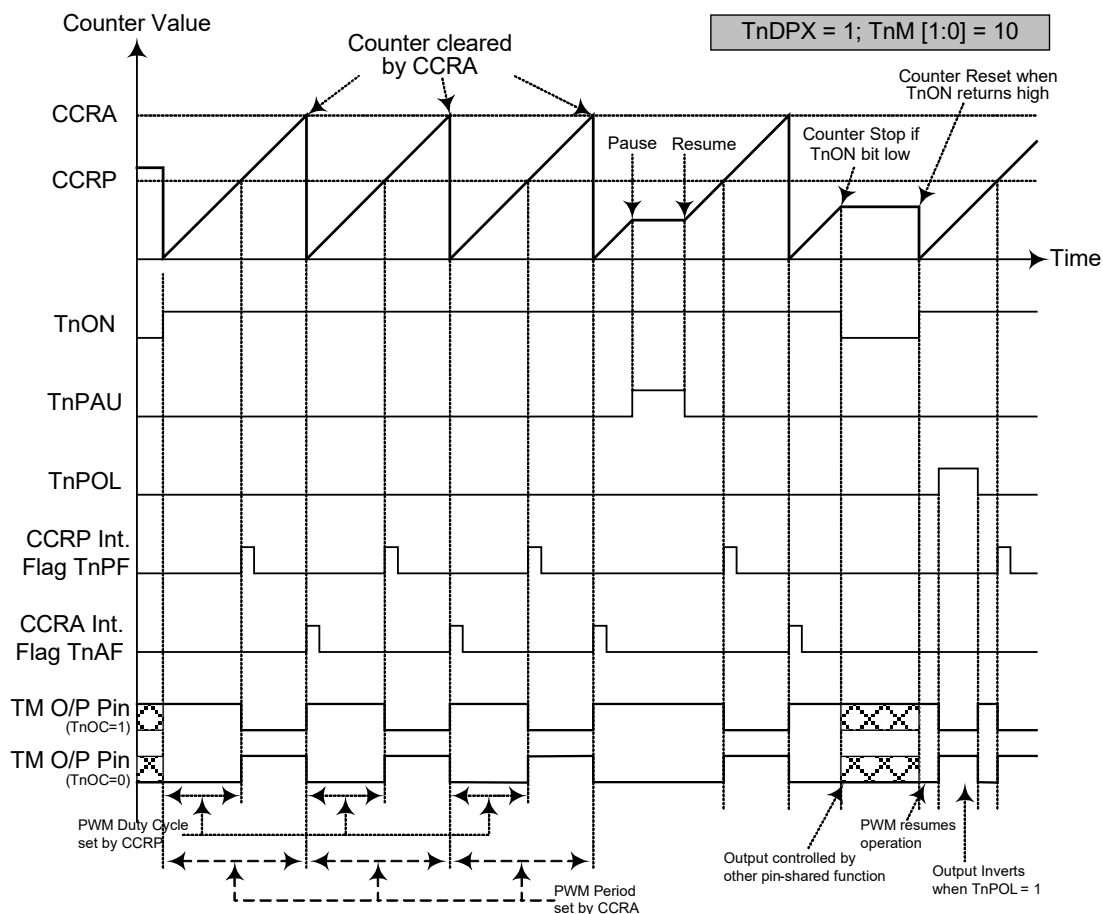
CCRP	1~255	0
Period	CCRA	
Duty	CCRP×256	65536

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



PWM 输出模式 – TnDPX=0

- 注：1. TnDPX=0, CCRP 清除计数器
2. 计数器清零并设置 PWM 周期
3. 当 TnIO1, TnIO0=00 或 01, PWM 功能不变
4. TnCCLR 位不影响 PWM 操作



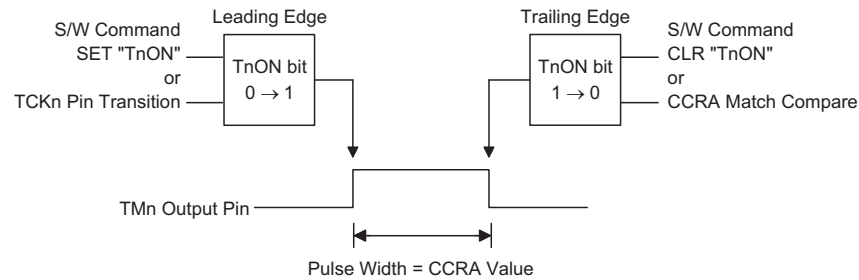
PWM 输出模式 – TnDPX=1

- 注：1. TnDPX=1，CCRA 清除计数器
2. 计数器清零并设置 PWM 周期
3. 当 TnIO1, TnIO0=00 或 01，PWM 功能不变
4. TnCCLR 位不影响 PWM 操作

单脉冲模式

为使 TM 工作在此模式，TMnC1 寄存器中的 TnM1 和 TnM0 位需要设置为“10”，同时 TnIO1 和 TnIO0 位需要设置为“11”。正如模式名所言，单脉冲输出模式，在 TM 输出脚将产生一个脉冲输出。

脉冲输出可以通过应用程序控制 TnON 位由低到高的转变来触发。而处于单脉冲模式时，TnON 位在 TCKn 脚自动由低转变为高，进而初始化单脉冲输出状态。当 TnON 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 TnON 位保持高电平。通过应用程序使 TnON 位清零或比较器 A 比较匹配发生时，产生脉冲下降沿。



单脉冲产生示意图



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

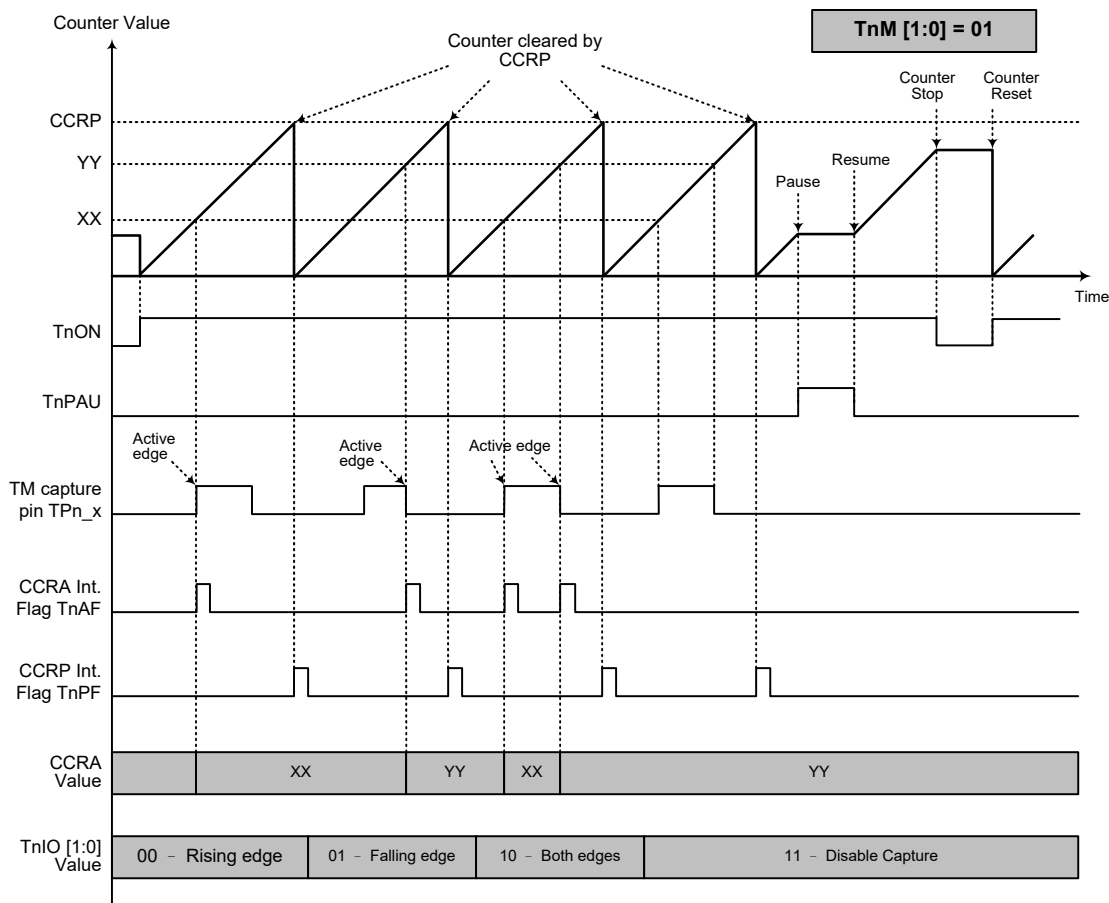
然而，比较器 A 比较匹配发生时，会自动清除 TnON 位并产生单脉冲输出下降沿。CCRA 的值通过这种方式控制脉冲宽度。比较器 A 比较匹配发生时，也会产生 TM 中断。TnON 位在计数器重启时会发生由低到高的转变，此时计数器才复位至零。在单脉冲模式中，CCRP 寄存器，TnCCLR 和 TnDPX 位未使用。

捕捉输入模式

为使 TM 工作在此模式，TMnC1 寄存器中的 TnM1 和 TnM0 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。TPn 脚上的外部信号，通过设置 TMnC1 寄存器的 TnIO1 和 TnIO0 位选择有效边沿类型，即上升沿，下降沿或双沿有效。计数器在 TnON 位由低到高转变时启动并通过应用程序初始化。

当 TPn 出现有效边沿转换时，计数器当前值被锁存到 CCRA 寄存器，并产生 TM 中断。不考虑 TPn 引脚事件，计数器继续工作直到 TnON 位发生下降沿跳变。当 CCRP 比较匹配发生时计数器复位至零；CCRP 的值通过这种方式控制计数器的最大值。当比较器 P CCRP 比较匹配发生时，也会产生 TM 中断。记录 CCRP 溢出中断信号的值可以测量脉宽。通过设置 TnIO1 和 TnIO0 位选择 TPn 引脚为上升沿，下降沿或双沿有效。不考虑 TPn 引脚事件，如果 TnIO1 和 TnIO0 位设置为高，不会产生捕捉操作，但计数器继续运行。

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



捕捉输入模式

- 注：1. TnM1, TnM0=01 并通过 TnIO1 和 TnIO0 位设置有效边沿
2. TM 捕捉输入脚的有效边沿将计数器的值转移到 CCRA 中
3. TnCCLR 位未使用
4. 无输出功能 – TnOC 和 TnPOL 位未使用
5. 计数器值由 CCRP 决定，在 CCRP 为 “0” 时，计数器计数值可达最大

A/D 转换器

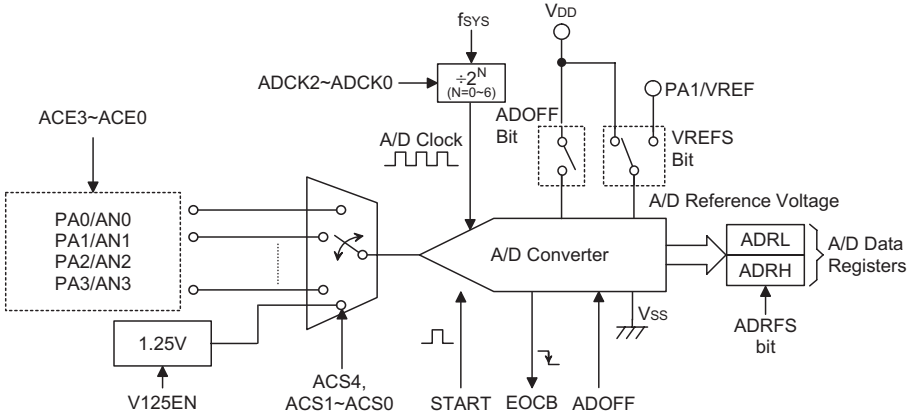
对于大多数电子系统而言，处理现实世界的模拟信号是共同的需求。为了完全由单片机来处理这些信号，首先需要通过 A/D 转换器将模拟信号转换成数字信号。将 A/D 转换器电路集成入单片机，可有效的减少外部器件，随之而来，具有降低成本和减少器件空间需求的优势。

A/D 简介

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

单片机	输入通道数	A/D 通道选择位	输入引脚
HT66F016 HT66F017	4	ACS4, ACS1~ACS0	AN0~AN3
HT68F016 HT68F017	—	—	—

下图显示了 A/D 转换器内部结构和相关的寄存器。



A/D 转换器结构

A/D 转换寄存器介绍

A/D 转换器的所有工作由五个寄存器控制。一对只读寄存器来存放 12 位 ADC 数据的值。剩下三个控制寄存器设置 A/D 转换器的操作和控制功能。

寄存器名称	位							
	7	6	5	4	3	2	1	0
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	—	—	ACS1	ACS0
ADCR1	ACS4	V125EN	—	VREFS	—	ADCK2	ADCK1	ADCK0
ACERL	—	—	—	—	ACE3	ACE2	ACE1	ACE0

A/D 转换寄存器列表

A/D 转换器数据寄存器 – ADRL, ADRLH

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

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

A/D 数据寄存器

A/D 转换控制寄存器 – ADCR0, ADCR1, ACERL

寄存器 ADCR0, ADCR1 和 ACERL 用来控制 A/D 转换器的功能和操作。这些 8 位的寄存器定义包括选择连接至内部 A/D 转换器的模拟通道，数字化数据格式，A/D 时钟源，并控制和监视 A/D 转换器的开始和转换结束状态。寄存器 ADCR0 的 ACS1~ACS0 位和 ADCR1 的 ACS4 位定义 ADC 输入通道编号。由于每个单片机只包含一个实际的模数转换电路，因此这 4 个模拟输入中的每一个都需要分别被发送到转换器。ACS1~ACS0 位以及 ACS4 位的功能决定选择哪个模拟输入通道或内部 1.25V 电路是否被连接到内部 A/D 转换器。

ACERL 控制寄存器中的 ACE3~ACE0 位，用来定义 PA0~PA3 的哪些引脚为 A/D 转换器的模拟输入，哪些引脚不作为 A/D 转换输入。相应位设为高将选择 A/D 输入功能，清零将选择 I/O 或其它引脚共用功能。当引脚作为 A/D 输入时，其原来的 I/O 或其它引脚共用功能消失，此外，其内部上拉电阻也将自动断开。

• ADCR0 寄存器

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

- Bit 7 **START**: 启动 A/D 转换位
0→1→0: 启动
0→1: 重置 A/D 转换，并且设置 EOCB 为“1”
此位用于初始化 A/D 转换过程。通常此位为低，但如果设为高再被清零，将初始化 A/D 转换过程。当此位为高，将重置 A/D 转换器。
- Bit 6 **EOCB**: A/D 转换结束标志
0: A/D 转换结束
1: A/D 转换中
此位用于表明 A/D 转换过程的完成。当转换正在进行时，此位为高。
- Bit 5 **ADOFF**: ADC 模块电源开 / 关控制位
0: ADC 模块电源开
1: ADC 模块电源关
此位控制 A/D 内部功能的电源。该位被清零将使能 A/D 转换器。如果该位设为高将关闭 A/D 转换器以降低功耗。由于 A/D 转换器在不执行转换动作时都会产生一定的功耗，所以这在电源敏感的电池应用中需要多加注意。
注: 1. 建议在进入空闲 / 休眠模式前，设置 ADOFF=1 以减少功耗。
2. ADOFF=1 将关闭 ADC 模块的电源。
- Bit 4 **ADRFS**: ADC 数据格式控制位

0: ADC 数据高字节是 ADRH 的 bit 7~bit 0, 低字节是 ADRL 的 bit 7~bit 4
1: ADC 数据高字节是 ADRH 的 bit 3~bit 0, 低字节是 ADRL 的 bit 7~bit 0
此位控制存放在两个 A/D 数据寄存器中的 12 位 A/D 转换结果的格式。细节方面请参考 A/D 数据寄存器章节。

Bit 3~2 未定义, 读为 “0”

Bit 1~0 **ACS1, ACS0**: 选择 A/D 通道 (ACS4 为 “0”) 位

00: AN0

01: AN1

10: AN2

11: AN3

这三位是 A/D 通道选择控制位。由于只包含一个内部 A/D 转换电路, 因此通过这些位将 4 个 A/D 输入连接到转换器。如果 ADCR1 寄存器中的 ACS4 设为高, 内部 1.25V 电路将被连接到内部 A/D 转换器。

● 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 ADC 输入控制位

0: 除能

1: 使能

此位使能 1.25V 连接到 A/D 转换器。V125EN 位必须先被置位使能 1.25V 电压能隙电路被用于 A/D 转换器。当 ACS4 设为高, 1.25V 能隙电压将连接到 A/D 转换器, 其它 A/D 输入通道断开。

Bit 6 **V125EN**: 内部 1.25V 控制位

0: 除能

1: 使能

此位控制连接到 A/D 转换器的内部充电泵电路开 / 关功能。当此位设为高, 充电泵电压 1.25V 连接至 A/D 转换器。如果 1.25V 未连接至 A/D 转换器且 LVR/LVD 除能, 充电泵参考电压电路自动关闭以减少功耗。当 1.25V 打开连接至 A/D 转换器, 在 A/D 转换动作执行前, 充电泵电路稳定需一段时间 t_{BG} 。

Bit 5 未定义, 读为 “0”

Bit 4 **VREFS**: 选择 ADC 参考电压

0: 内部 ADC 电源

1: VREF 引脚

此位用于选择 A/D 转换器的参考电压。如果该位设为高, A/D 转换器参考电压来源于外部 VREF 引脚。如果该位设为低, 内部参考电压来源于电源电压 VDD。若参考电压来源于外部 VREF 引脚, 刚引脚的其它共用功能除能。

Bit 3 未定义, 读为 “0”

Bit 2~0 **ADCK2, ADCK1, ADCK0**: 选择 ADC 时钟源

000: f_{SYS}

001: $f_{SYS}/2$

010: $f_{SYS}/4$

011: $f_{SYS}/8$

100: $f_{SYS}/16$

101: $f_{SYS}/32$

110: $f_{SYS}/64$

111: 未定义

这三位用于选择 A/D 转换器的时钟源。

● ACERL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	ACE3	ACE2	ACE1	ACE0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	1	1	1	1

- Bit 7~4 未定义，读为“0”
- Bit 3 **ACE3**：定义 PA3 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 PA2 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 PA1 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 PA0 是否为 A/D 输入
0：不是 A/D 输入
1：A/D 输入，AN0

A/D 操作

ADCR0 寄存器中的 START 位，用于打开和复位 A/D 转换器。当单片机设定此位从逻辑低到逻辑高，然后再到逻辑低，就会开始一个模数转换周期。当 START 位从逻辑低到逻辑高，但不再回到逻辑低时，ADCR0 寄存器中的 EOCB 位置“1”，复位模数转换器。START 位用于控制内部模数转换器的开启动作。

ADCR0 寄存器中的 EOCB 位用于表明模数转换过程的完成。在转换周期结束后，EOCB 位会被单片机自动地置为“0”。此外，也会置位中断控制寄存器内相应的 A/D 中断请求标志位，如果中断使能，就会产生对应的内部中断信号。A/D 内部中断信号将引导程序到相应的 A/D 内部中断入口。如果 A/D 内部中断被禁止，可以让单片机轮询 ADCR0 寄存器中的 EOCB 位，检查此位是否被清除，以作为另一种侦测 A/D 转换周期结束的方法。

A/D 转换器的时钟源为系统时钟 f_{SYS} 分频，而分频系数由 ADCR1 寄存器中的 ADCK2~ADCK0 位决定。

虽然 A/D 时钟源是由系统时钟 f_{SYS} ，ADCK2~ADCK0 位决定，但可选择的最大 A/D 时钟源则有一些限制。建议 A/D 时钟周期 t_{ADCK} 的允许范围是 $0.5\mu s \sim 10\mu s$ ，因此应该慎重选择系统时钟频率。如果系统时钟频率为 4MHz 时，ADCK2~ADCK0 位不能设为“000”或“110”。必须保证设定的 A/D 转换时钟周期不小于时钟周期的最小值，否则将会产生不准确的 A/D 转换值。使用者可以参考下面的表格，被标上星号 * 的数值是不允许的，因为它们的 A/D 转换时钟周期小于规定的最小值。

f _{sys}	A/D 时钟周期 (t _{ADCK})							
	ADCK2, ADCK1, ADCK0 =000 (f _{sys})	ADCK2, ADCK1, ADCK0 =001 (f _{sys} /2)	ADCK2, ADCK1, ADCK0 =010 (f _{sys} /4)	ADCK2, ADCK1, ADCK0 =011 (f _{sys} /8)	ADCK2, ADCK1, ADCK0 =100 (f _{sys} /16)	ADCK2, ADCK1, ADCK0 =101 (f _{sys} /32)	ADCK2, ADCK1, ADCK0 =110 (f _{sys} /64)	ADCK2, ADCK1, ADCK0 =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
12MHz	83ns*	167ns*	333ns*	667ns	1.33μs	2.67μs	5.33μs	未定义

A/D 时钟周期范例

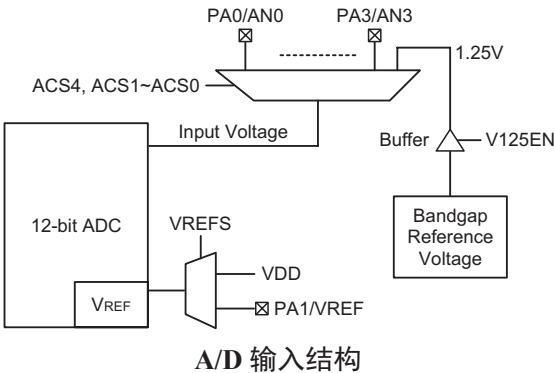
ADCR0 寄存器的 ADOFF 位用于控制 A/D 转换电路电源的开 / 关。该位必须清零以开启 A/D 转换器电源。即使通过清除 ACERL 寄存器的 ACE3~ACE0 位，选择无引脚作为 A/D 输入，如果 ADOFF 设为“0”，那么仍然会产生功耗。因此当未使用 A/D 转换器功能时，在功耗敏感的应用中建议设置 ADOFF 为高以减少功耗。

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

A/D 输入引脚

所有的 A/D 模拟输入引脚都与 PA3~PA0 引脚及其它功能共用。使用 ACERL 寄存器中的 ACE3~ACE0 位，可以将它们设置为 A/D 转换器模拟输入脚或具有其它功能。如果引脚的对应位 ACE3~ACE0 设为高，那么该引脚作为 A/D 转换输入且原引脚功能除能。通过这种方式，引脚的功能可由程序来控制，灵活地切换引脚功能。如果将引脚设为 A/D 输入，则通过寄存器编程设置的所有上拉电阻会自动断开。请注意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 ACE3~ACE0 位使能 A/D 输入时，端口控制寄存器的状态将被重置。

A/D 转换器有自己的参考电压引脚 VREF，而通过设置 ADCR1 寄存器的 VREFS 位，参考电压也可以选择来自电源电压引脚。模拟输入值一定不能超过 V_{REF} 值。

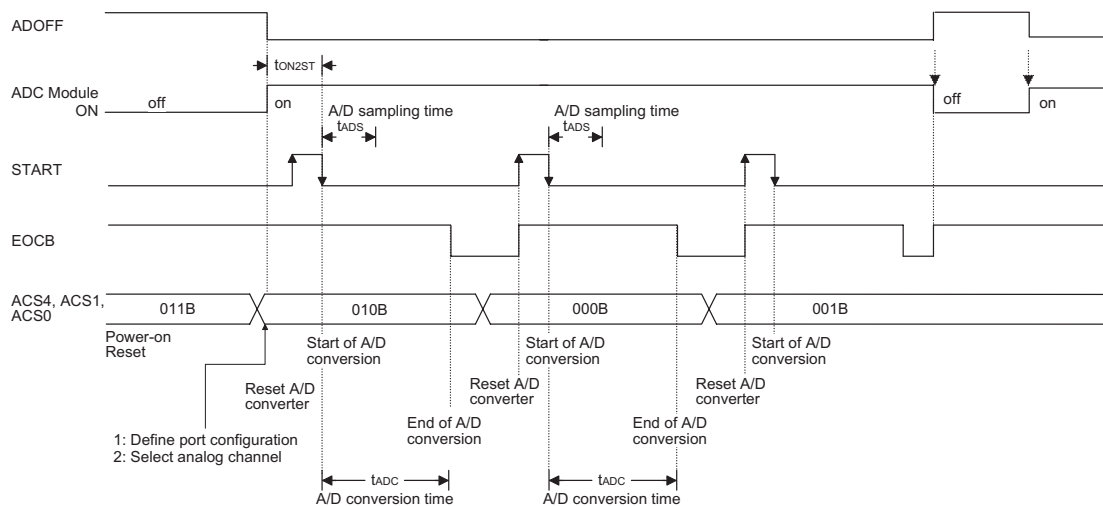


A/D 转换步骤

下面概述实现 A/D 转换过程的各个步骤。

- 步骤 1
通过 ADCR1 寄存器中的 ADCK2~ADCK0 位，选择所需的 A/D 转换时钟。
- 步骤 2
清零 ADCR0 寄存器中的 ADOFF 位使能 A/D。
- 步骤 3
通过 ADCR1 和 ADCR0 寄存器中的 ACS4， ACS1~ACS0 位，选择连接至内部 A/D 转换器的通道。
- 步骤 4
通过 ACERL 寄存器中的 ACE3~ACE0 位，选择哪些引脚规划为 A/D 输入引脚。
- 步骤 5
如果要使用中断，则中断控制寄存器需要正确地设置，以确保 A/D 转换功能是激活的。总中断控制位 EMI 需要置位为“1”，以及 A/D 转换器中断位 ADE 也需要置位为“1”。
- 步骤 6
现在可以通过设定 ADCR0 寄存器中的 START 位从“0”到“1”再回到“0”，开始模数转换的过程。注意，该位需初始化为“0”。
- 步骤 7
可以轮询 ADCR0 寄存器中的 EOCB 位，检查模数转换过程是否完成。当此位成为逻辑低时，表示转换过程已经完成。转换完成后，可读取 A/D 数据寄存器 ADRL 和 ADRH 获得转换后的值。另一种方法是，若中断使能且堆栈未满，则程序等待 A/D 中断发生。
注：若使用轮询 ADCR0 寄存器中 EOCB 位的状态的方法来检查转换过程是否结束时，则中断使能的步骤可以省略。

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



A/D 转换时序图

编程注意事项

在编程时，如果 A/D 转换器未使用，通过设置 ADCR0 寄存器中的 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 转换功能

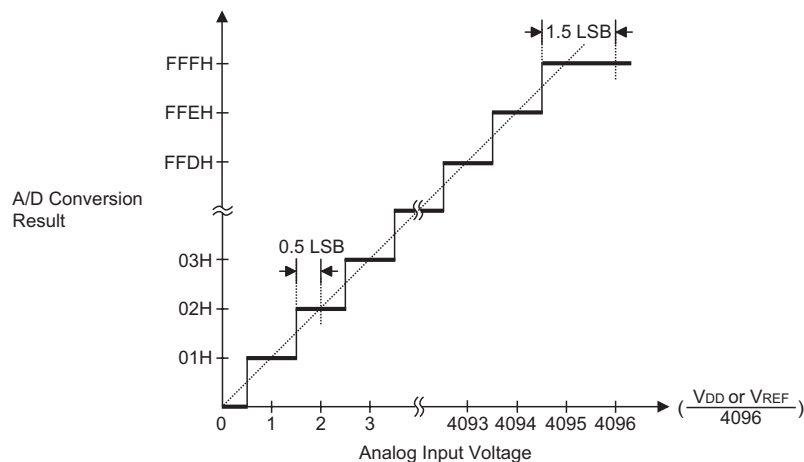
单片机含有一组 12 位的 A/D 转换器，它们转换的最大值可达 FFFH。由于模拟输入最大值等于 V_{DD} 或 V_{REF} 的电压值，因此每一位可表示 V_{DD} 或 $V_{REF}/4096$ 的模拟输入值。

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

通过下面的等式可估算 A/D 转换器输入电压值：

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

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



理想的 A/D 转换功能

A/D 转换应用范例

下面两个范例程序用来说明怎样使用 A/D 转换。第一个范例是轮询 ADCR0 寄存器中的 EOCB 位来判断 A/D 转换是否完成；第二个范例则使用中断的方式判断。

范例：使用查询 EOCB 的方式来检测转换结束

```

clr ADE                      ; disable ADC 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 and ACERH to configure pins AN0~AN3
mov ACERL,a
mov a,00h
mov ADCR0,a                  ; enable and connect AN0 channel to A/D converter
:
start_conversion:
clr START                    ; high pulse on start bit to initiate conversion
set START                    ; reset A/D
clr START                    ; start A/D
polling_EOC:
sz EOCB                      ; poll the ADCR0 register EOCB bit to detect end
                                ; of A/D conversion
jmp polling_EOC              ; continue polling
mov a,ADRL                   ; read low byte conversion result value
mov ADRL_buffer,a            ; save result to user defined register
mov a,ADRH                   ; read high byte conversion result value
mov ADRH_buffer,a            ; save result to user defined register
:
:
jmp start_conversion          ; start next A/D conversion

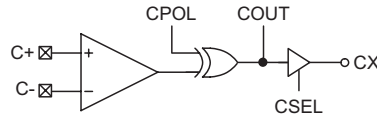
```

范例：使用中断的方式来检测转换结束

```
clr ADE                      ; disable ADC 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 A/D converter
Start_conversion:
clr START                    ; high pulse on start bit to initiate conversion
set START                    ; reset A/D
clr START                    ; start A/D
clr ADF                      ; clear ADC interrupt request flag
set ADE                      ; enable ADC interrupt
set EMI                      ; enable global interrupt
:
:
                                ; ADC interrupt service routine
ADC_ISR:
mov acc_stack,a              ; save ACC to user defined memory
mov a,STATUS
mov status_stack,a           ; save STATUS to user defined memory
:
:
mov a,ADRL                   ; read low byte conversion result value
mov adrl_buffer,a            ; save result to user defined register
mov a,ADRH                   ; read high byte conversion result value
mov adrh_buffer,a            ; save result to user defined register
:
:
EXIT_INT_ISR:
mov a,status_stack
mov STATUS,a                 ; restore STATUS from user defined memory
mov a,acc_stack              ; restore ACC from user defined memory
reti
```

比较器

该系列芯片中含有一个独立的模拟比较器。它们具有暂停、极性选择、迟滞等功能，可通过寄存器进行灵活配置。比较器的引脚与普通 I/O 引脚共用，当比较器功能未使用时，此引脚可做普通引脚使用而不浪费 I/O 资源。



比较器

比较器操作

此系列单片机包含一个比较器功能，用于比较两个模拟电压，基于它们的差值上提供一个输出。控制寄存器 CPC 可控制相应的内部比较器。比较器的输出可由寄存器的一位记录，并且在共用的 I/O 口上输出。此外，比较器功能有输出极性，迟滞功能和暂停控制。

当比较器使能时，连接到与比较器共用的输入引脚的上拉电阻将自动失效。当比较器输入接近其切换电压时，由于输入信号上升或下降速度较慢，比较器输出端可能会产生一些伪输出信号。通过选择迟滞功能提供少量正反馈给比较器可使此种情况的发生率较大程度地降低。理想情况下正负输入信号在同一个电压点时比较器将发生开关动作，但是不可避免的输入失调电压会导致情况不确定。若迟滞功能使能，也可增加切换偏差值。

比较器寄存器

CPC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	CSEL	CEN	CPOL	COUT	COS	—	—	CHYEN
R/W	R/W	R/W	R/W	R	R/W	—	—	R/W
POR	1	0	0	0	0	—	—	1

Bit 7 **CSEL:** 比较器引脚或 I/O 引脚选择位

- 0: 输入 / 输出引脚
- 1: 比较器引脚

此位为比较器引脚或输入 / 输出引脚选择位。为高时，比较器输入引脚使能。此时，这两个引脚的输入 / 输出功能失效，任何与比较器共用引脚的上拉电阻配置选项将自动失效。

Bit 6 **CEN:** 比较器开 / 关控制位

- 0: 关闭
- 1: 开启

此位为比较器开 / 关控制位。为“0”时，比较器关闭，即使其引脚上加有模拟电压也不会产生功耗。对功耗要求严格的应用中，当比较器未使用或单片机进入休眠或空闲模式之前，此位应清零。

Bit 5 **CPOL:** 比较器输出极性位

- 0: 输出同相
- 1: 输出反相

此位决定比较器极性。为“0”时，COUT 位与比较器输出条件同相；为“1”时，COUT 位与比较器输出条件反相。

Bit 4	COUT: 比较器输出位 CPOL=0 0: $C+ < C-$ 1: $C+ > C-$ CPOL=1 0: $C+ > C-$ 1: $C+ < C-$ 此位为比较器输出位。此位的极性由比较器输入电压和 CPOL 位的状态决定。
Bit 3	COS: 输出路径选择位 0: CX 引脚 1: 内部使用 此位为比较器输出路径选择控制位。当此位为“0”且 CSEL 位为“1”时，比较器输出连接到外部 CX 引脚。当此位为“1”或 CSEL 位为“0”时，比较器输出信号仅供单片机内部使用，且共用输出引脚作为普通输入 / 输出引脚使用。
Bit 2~1	未使用，读为“0”
Bit 0	CHYEN: 迟滞控制位 0: 关闭 1: 开启 此位为迟滞控制位。为“1”时，比较器有一定量迟滞，具体见比较器电气特性表。滞后产生的正反馈将减少比较器门槛附近的伪开关效应的影响。

比较器中断

比较器有自己的中断功能。当任何一个状态改变时，其对应的中断标志将会置位，若应答中断使能位被置位，系统将跳转至相应的中断向量中执行。注意，发生状态改变的是 COUT 位而不是产生中断的输出脚。单片机处于休眠或空闲模式且比较器使能时，若为外部输入线导致比较器输出状态发生的改变，则由此产生的中断标志也可产生一个唤醒动作。若除能唤醒功能，进入休眠或空闲模式前中断标志应先置为高。

编程注意事项

若比较器使能，当单片机进入休眠或空闲模式时其仍保持有效并会有一定的耗电，用户可考虑在进入休眠或空闲模式前先关闭比较器。

由于比较器引脚与普通输入 / 输出脚共用，若比较器功能使能时，这些引脚的输入 / 输出寄存器将读为“0”（端口控制寄存器读为“1”）或读为端口数据寄存器的值（端口控制寄存器读为“0”）。

中断

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

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。寄存器的数量由所选单片机的型号决定，但总的分为三类。第一类是 INTC0~INTC2 寄存器，用于设置基本的中断；第二类是 MFI0~MFI2 寄存器，用于设置多功能中断；最后一种有 INTEG 寄存器，用于设置外部中断边沿触发类型。

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

功能	使能位	请求标志	注释
总中断	EMI	—	—
INTn 脚	INTnE	INTnF	n=0~1
比较器	CPE	CPF	—
多功能	MFnE	MFnF	n=0~2
A/D 转换器	ADE	ADF	只有 HT66F016, HT66F017
时基	TBnE	TBnF	n=0~1
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
TM	TnPE	TnPF	n=0~1
	TnAE	TnAF	

中断寄存器位命名模式

寄存器名称	位							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	—	CPF	INT0F	—	CPE	INT0E	EMI
INTC1	TB0F	ADF	MF2F	MF1F	TB0E	ADE	MF2E	MF1E
INTC2	—	—	INT1F	TB1F	—	—	INT1E	TB1E
MFI0	—	—	—	—	—	—	—	—
MFI1	—	—	T1AF	T1PF	—	—	T1AE	T1PE
MFI2	—	—	DEF	LVF	—	—	DEE	LVE

注：HT68F016 没有 ADE 和 ADF 位

中断寄存器内容 – HT66F016/HT68F016

寄存器名称	位							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	MF0F	CPF	INT0F	MF0E	CPE	INT0E	EMI
INTC1	TB0F	ADF	MF2F	MF1F	TB0E	ADE	MF2E	MF1E
INTC2	—	—	INT1F	TB1F	—	—	INT1E	TB1E
MFI0	—	—	T0AF	T0PF	—	—	T0AE	T0PE
MFI1	—	—	T1AF	T1PF	—	—	T1AE	T1PE
MFI2	—	—	DEF	LVF	—	—	DEE	LVE

注：HT68F017 没有 ADE 和 ADF 位

中断寄存器内容 – HT66F017/HT68F017

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 未使用，读为“0”

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

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

INTC0 寄存器

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

注：HT66F016/HT68F016 没有 MF0F，MF0E 位

- Bit 7 未使用，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 **CPF**：比较器中断请求标志位
0：无请求
1：中断请求
- Bit 4 **INT0F**：INT0 中断请求标志位
0：无请求
1：中断请求
- Bit 3 **MF0E**：多功能中断 0 控制位
0：除能
1：使能
- Bit 2 **CPE**：比较器中断控制位
0：除能
1：使能
- Bit 1 **INT0E**：INT0 中断控制位
0：除能
1：使能
- Bit 0 **EMI**：总中断控制位
0：除能
1：使能

INTC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	TB0F	ADF	MF2F	MF1F	TB0E	ADE	MF2E	MF1E
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

注：HT68F016/HT68F017 没有 ADE，ADF 位

- 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 **MF2F**: 多功能中断 2 请求标志位
 0: 无请求
 1: 中断请求
- 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 **MF2E**: 多功能中断 2 控制位
 0: 除能
 1: 使能
- Bit 0 **MF1E**: 多功能中断 1 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 未使用，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 请求标志位
0: 无请求
1: 中断请求
- Bit 3~2 未使用，读为“0”
- Bit 1 **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

MFIO 寄存器

• HT66F017/HT68F017

Bit	7	6	5	4	3	2	1	0
Name	—	—	T0AF	T0PF	—	—	T0AE	T0PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

- Bit 7~6 未使用，读为“0”
- Bit 5 **T0AF**: TM0 比较器 A 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 4 **T0PF**: TM0 比较器 P 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 3~2 未使用，读为“0”
- Bit 1 **T0AE**: TM0 比较器 A 匹配中断控制位
0: 除能
1: 使能
- Bit 0 **T0PE**: TM0 比较器 P 匹配中断控制位
0: 除能
1: 使能

MF11 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	T1AF	T1PF	—	—	T1AE	T1PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

- Bit 7~6 未使用，读为“0”
- Bit 5 **T1AF**: TM1 比较器 A 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 4 **T1PF**: TM1 比较器 P 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 3~2 未使用，读为“0”
- Bit 1 **T1AE**: TM1 比较器 A 匹配中断控制位
0: 除能
1: 使能
- Bit 0 **T1PE**: TM1 比较器 P 匹配中断控制位
0: 除能
1: 使能

MF12 寄存器

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

- Bit 7~6 未使用，读为“0”
- Bit 5 **DEF**: 数据 EEPROM 中断请求标志位
0: 无请求
1: 中断请求
- Bit 4 **LVF**: LVD 中断请求标志位
0: 无请求
1: 中断请求
- Bit 3~2 未使用，读为“0”
- Bit 1 **DEE**: 数据 EEPROM 中断控制位
0: 除能
1: 使能
- Bit 0 **LVE**: LVD 中断控制位
0: 除能
1: 使能

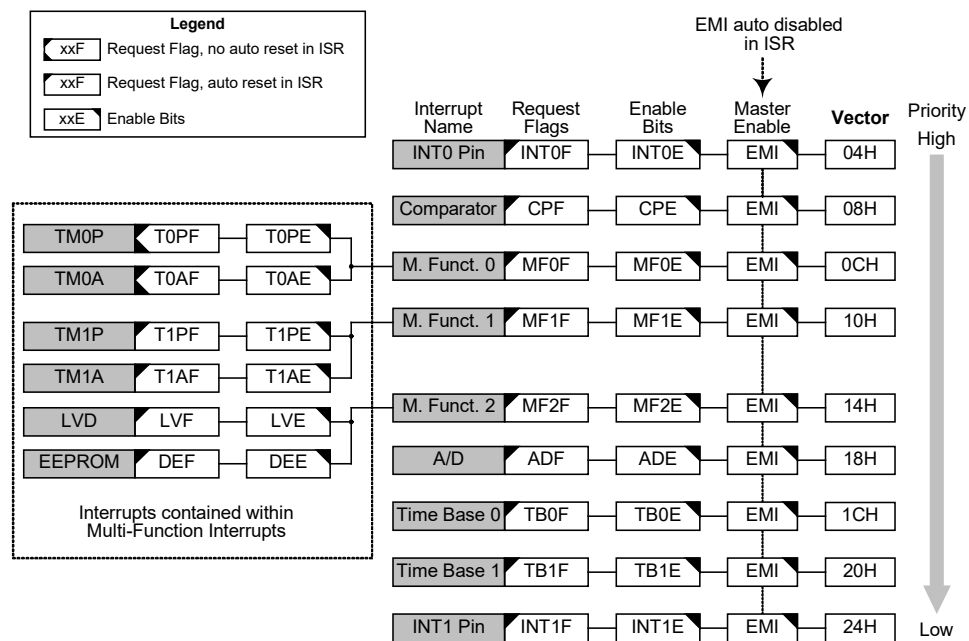
中断操作

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

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

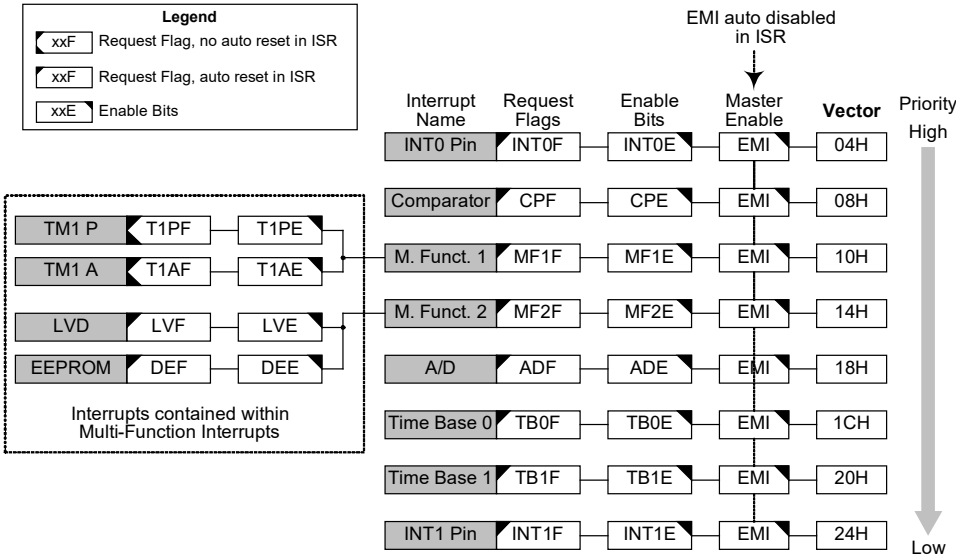
各个中断使能位以及相应的请求标志位，以优先级的次序显示在下图。一些中断源有自己的向量，但是有些中断却共用多功能中断向量。一旦中断子程序被响应，系统将自动清除 EMI 位，所有其它的中断将被屏蔽，这个方式可以防止任何进一步的中断嵌套。其它中断请求可能发生在此期间，虽然中断不会立即响应，但是中断请求标志位会被记录。

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



注：HT68F017 没有 ADE，ADF 位

中断结构 – HT66F017/HT68F017



注：HT68F016 没有 ADE，ADF 位

中断结构 – HT66F016/HT68F016

外部中断

通过 INT0~INT1 引脚上的信号变化可控制外部中断。当触发沿选择位设置好触发类型，INT0~INT1 引脚的状态发生变化，外部中断请求标志 INT0F~INT1F 被置位时外部中断请求产生。若要跳转到相应中断向量地址，总中断控制位 EMI 和相应中断使能位 INT0E~INT1E 需先被置位。此外，必须使用 INTEG 寄存器使能外部中断功能并选择触发沿类型。外部中断引脚和普通 I/O 口共用，如果相应寄存器中的中断使能位被置位，此引脚将被作为外部中断脚使用。此时该引脚必须通过设置控制寄存器，将该引脚设置为输入口。当中断使能，堆栈未满并且外部中断脚状态改变，将调用外部中断向量子程序。当响应外部中断服务子程序时，中断请求标志位 INT0F~INT1F 会自动复位且 EMI 位会被清零以除能其它中断。注意，即使此引脚被用作外部中断输入，其配置选项中的上拉电阻仍保持有效。寄存器 INTEG 被用来选择有效的边沿类型，来触发外部中断。可以选择上升沿还是下降沿或双沿触发都产生外部中断。注意 INTEG 也可以用来除能外部中断功能。

比较器中断

比较器中断由内部比较器控制。当比较器输出状态改变，比较器中断请求标志 CPF 被置位，比较器中断请求产生。若要跳转到相应中断向量地址，总中断控制位 EMI 和比较器中断使能位 CPE 需先被置位。当中断使能，堆栈未满并且比较器输入产生一个比较器输出变化时，将调用比较器中断向量子程序。当响应中断服务子程序时，外部中断请求标志位会自动复位且 EMI 位会被清零以除能其它中断。

多功能中断

此系列单片机中有多达三种多功能中断，与其它中断不同，它没有独立源，但由其它现有的中断源构成，即 TM 中断，LVD 中断和 EEPROM 中断。

当多功能中断中任何一种中断请求标志 MF0F~MF2F 被置位，多功能中断请求产生。当中断使能，堆栈未满，包括在多功能中断中的任意一个中断发生时，将调用多功能中断向量中的一个子程序。当响应中断服务子程序时，相关多功能请求标志位会自动复位且 EMI 位会自动清零以除能其它中断。

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

A/D 转换器中断

A/D 转换器中断由 A/D 转换动作的结束来控制。当 A/D 转换器中断请求标志被置位，即 A/D 转换过程完成时，中断请求发生。当总中断使能位 EMI 和 A/D 中断使能位 ADE 被置位，允许程序跳转到各自的中断向量地址。当中断使能，堆栈未满且 A/D 转换动作结束时，将调用它们各自的中断向量子程序。当响应中断服务子程序时，相应的中断请求标志位 ADF 会自动清零。EMI 位也会被清零以除能其它中断。

时基中断

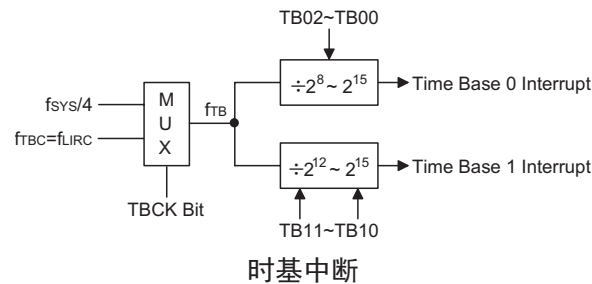
时基中断提供一个固定周期的中断信号，由各自的定时器功能产生溢出信号控制。当各自的中断请求标志 TB0F 或 TB1F 被置位时，中断请求发生。当总中断使能位 EMI 和时基使能位 TB0E 或 TB1E 被置位，允许程序跳转到各自的中断向量地址。当中断使能，堆栈未满且时基溢出时，将调用它们各自的中断向量子程序。当响应中断服务子程序时，相应的中断请求标志位 TB0F 或 TB1F 会自动复位且 EMI 位会被清零以除能其它中断。

时基中断的目的是提供一个固定周期的中断信号，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。控制时基中断频率 f_{TB} 的时钟源有几种，如在系统工作模式章节所示。

TBC 寄存器

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

- Bit 7 **TBON**: TB0 和 TB1 控制位
0: 除能
1: 使能
- Bit 6 **TBCK**: 选择 f_{TB} 时钟位
0: f_{TBC}
1: $f_{SYS}/4$
- Bit 5~4 **TB11~TB10**: 选择时基 1 溢出周期位
00: $4096/f_{TB}$
01: $8192/f_{TB}$
10: $16384/f_{TB}$
11: $32768/f_{TB}$
- Bit 3 未使用, 读为 “0”
- Bit 2~0 **TB02~TB00**: 选择时基 0 溢出周期位
000: $256/f_{TB}$
001: $512/f_{TB}$
010: $1024/f_{TB}$
011: $2048/f_{TB}$
100: $4096/f_{TB}$
101: $8192/f_{TB}$
110: $16384/f_{TB}$
111: $32768/f_{TB}$



EEPROM 中断

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

LVD 中断

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

TM 中断

简易型和标准型 TM 各有两个中断。所有的 TM 中断也属于多功能中断。简易型和标准型 TM 各有两个中断请求标志位 TnPF、TnAF 及两个使能位 TnPE、TnAE。当 TM 比较器 P、A 匹配情况发生时，任意 TM 中断请求标志被置位，TM 中断请求产生。

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

中断唤醒功能

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

编程注意事项

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

多功能中断中所含中断相应程序执行时，多功能中断请求标志 MF0F~MF2F 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”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”指令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高到低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。

低电压检测 – LVD

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

LVD 寄存器

低电压检测功能由 LVDC 寄存器控制。VLVD2~VLVD0 位用于选择 8 个固定的电压参考点。LVDO 位被置位时低电压情况发生，若 LVDO 位为低表明 V_{DD} 电压工作在当前所设置低电压水平值之上。LVDEN 位用于控制低电压检测功能的开启 / 关闭，设置此位为高使能此功能，反之，关闭内部低电压检测电路。低电压检测会有一定的功耗，在不使用时可考虑关闭此功能，此举在功耗要求严格的电池供电应用中值得考虑。

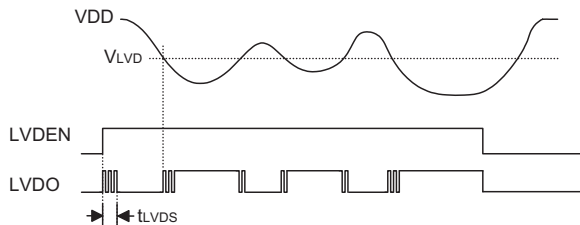
LVDC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	LVDO	LVDEN	—	VLVD2	VLVD1	VLVD0
R/W	—	—	R	R/W	—	R/W	R/W	R/W
POR	—	—	0	0	—	0	0	0

- Bit 7~6 未定义，读为 “0”
- Bit 5 **LVDO**: LVD 输出标志位
0: 未检测到低电压
1: 检测到低电压
- Bit 4 **LVDEN**: 低电压检测控制位
0: 除能
1: 使能
- Bit 3 未定义，读为 “0”
- Bit 2~0 **VLVD2~VLVD0**: 选择 LVD 电压位
000: 2.0V
001: 2.2V
010: 2.4V
011: 2.7V
100: 3.0V
101: 3.3V
110: 3.6V
111: 4.2V

LVD 操作

通过比较电源电压 V_{DD} 与存储在 LVDC 寄存器中的预置电压值的结果，低电压检测功能工作。其设置的范围为 2.0V~4.2V。当电源电压 V_{DD} 低于预置电压值时，LVDO 位被置为高，表明低电压产生。低电压检测功能由一个自动使能的参考电压提供。若 LVDEN 位为高，当单片机掉电时低电压检测器保持有效状态。低电压检测器使能后，读取 LVDO 位前，电路稳定需要一定的延时 t_{LVDs} 。注意， V_{DD} 电压可能上升或下降比较缓慢，在 V_{LVD} 电压值附近时，LVDO 位可能有多种变化。



LVD 操作

低电压检测器也有自己的中断功能，也是属于多功能中断的一种，它是除了轮询 LVDO 位之外的另一种检测低电压的方法。中断条件产生置位 LVDO 并延时 t_{LVD} 后，中断产生。若 LVDS 位为高，当单片机掉电时低电压检测器保持有效状态。此种情况下，若 V_{DD} 降至小于 LVD 预置电压值时，中断请求标志位 LVF 将被置位，中断产生，单片机将被从休眠或空闲模式中唤醒。若不要求低电压检测的唤醒功能使能，在单片机进入休眠或空闲模式前应将 LVF 标志置为高。

配置选项

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

序号	选项
振荡器选项	
1	高速振荡器类型选择 – f_{osc} : 1. HXT 2. HIRC
复位脚选项	
2	PA7/RES 引脚选择: 1. RES 脚 2. I/O
看门狗选项	
3	看门狗定时器功能: 1. 始终使能 2. 通过 S/W 控制

应用电路



注：“*”表示建议加上此元件以加强静电保护。

“**”表示建议在电源有较强干扰场合加上此元件。

指令集

简介

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

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 0.5 μ s 中执行完成，而分支或调用操作则将在 1 μ s 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入口的 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 and TBLH	2 注	无
TABRDC [m]	读取当前页的 ROM 内容，并送至数据存储器 and TBLH	2 注	无
TABRDL [m]	读取最后页的 ROM 内容，并送至数据存储器 and 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多达 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	$ACC \leftarrow ACC + [m] + C$
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	$[m] \leftarrow ACC + [m] + C$
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	$ACC \leftarrow ACC + [m]$
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	$ACC \leftarrow ACC + x$
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	$[m] \leftarrow ACC + [m]$
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	$ACC \leftarrow ACC \text{ “AND” } [m]$
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	$ACC \leftarrow ACC \text{ "AND" } x$
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	$[m] \leftarrow ACC \text{ "AND" } [m]$
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	$Stack \leftarrow Program\ Counter + 1$ $Program\ Counter \leftarrow addr$
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	$[m] \leftarrow 00H$
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	$[m].i \leftarrow 0$
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 $TO \ \& \ PDF \leftarrow 0$
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 $\leftarrow$ 00H TO & PDF $\leftarrow$ 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 $\leftarrow$ 00H TO & PDF $\leftarrow$ 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	$[m] \leftarrow \overline{[m]}$
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	$ACC \leftarrow \overline{[m]}$
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，结果存放和数据存储器。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	$[m] \leftarrow ACC + 00H$ 或 $[m] \leftarrow ACC + 06H$ 或 $[m] \leftarrow ACC + 60H$ 或 $[m] \leftarrow ACC + 66H$
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	$[m] \leftarrow [m] - 1$
影响标志位	Z
DECA [m]	Decrement Data Memory with result in ACC
指令说明	将指定数据存储器内容减 1，把结果存放回累加器并保持指定数据存储器内容不变。
功能表示	$ACC \leftarrow [m] - 1$
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	$TO \leftarrow 0$ $PDF \leftarrow 1$
影响标志位	TO、PDF
INC [m]	Increment Data Memory
指令说明	将指定数据存储器内容加 1。
功能表示	$[m] \leftarrow [m] + 1$
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	$ACC \leftarrow [m] + 1$
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 $\leftarrow$ 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	$ACC \leftarrow [m]$
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	$ACC \leftarrow x$
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	$[m] \leftarrow ACC$
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	$PC \leftarrow PC+1$
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	$ACC \leftarrow ACC \text{ "OR" } [m]$
影响标志位	Z

ORA, 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	$ACC \leftarrow ACC \text{ "OR" } x$
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	$[m] \leftarrow ACC \text{ "OR" } [m]$
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	$Program\ Counter \leftarrow Stack$
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	$Program\ Counter \leftarrow Stack$ $ACC \leftarrow x$
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	$Program\ Counter \leftarrow Stack$ $EMI \leftarrow 1$
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	$[m].(i+1) \leftarrow [m].i \ (i=0\sim6)$ $[m].0 \leftarrow [m].7$
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	$ACC.(i+1) \leftarrow [m].i \ (i=0\sim6)$ $ACC.0 \leftarrow [m].7$
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	$[m].(i+1) \leftarrow [m].i \ (i=0\sim6)$ $[m].0 \leftarrow C$ $C \leftarrow [m].7$
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	$ACC.(i+1) \leftarrow [m].i \ (i=0\sim6)$ $ACC.0 \leftarrow C$ $C \leftarrow [m].7$
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	$[m].i \leftarrow [m].(i+1) \ (i=0\sim6)$ $[m].7 \leftarrow [m].0$
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	$ACC.i \leftarrow [m].(i+1) \ (i=0\sim6)$ $ACC.7 \leftarrow [m].0$
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	$[m].i \leftarrow [m].(i+1) \ (i=0\sim6)$ $[m].7 \leftarrow C$ $C \leftarrow [m].0$
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	$ACC.i \leftarrow [m].(i+1) \ (i=0\sim6)$ $ACC.7 \leftarrow C$ $C \leftarrow [m].0$
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	$ACC \leftarrow ACC - [m] - \bar{C}$
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	$[m] \leftarrow ACC - [m] - \bar{C}$
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	$[m] \leftarrow [m] - 1$ ，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	$ACC \leftarrow [m]-1$ ，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	$[m] \leftarrow FFH$
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	$[m].i \leftarrow 1$
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	$[m] \leftarrow [m]+1$ ，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	$ACC \leftarrow [m]+1$ ，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	$ACC \leftarrow ACC - [m]$
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	$[m] \leftarrow ACC - [m]$
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	$ACC \leftarrow ACC - x$
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	$[m].3 \sim [m].0 \leftrightarrow [m].7 \sim [m].4$
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	$ACC.3 \sim ACC.0 \leftarrow [m].7 \sim [m].4$ $ACC.7 \sim ACC.4 \leftarrow [m].3 \sim [m].0$
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	$[m] \leftarrow \text{程序代码 (低字节)}$ $TBLH \leftarrow \text{程序代码 (高字节)}$
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	$ACC \leftarrow ACC \text{ "XOR" } [m]$
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	$[m] \leftarrow ACC \text{ "XOR" } [m]$
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	$ACC \leftarrow ACC \text{ "XOR" } x$
影响标志位	Z

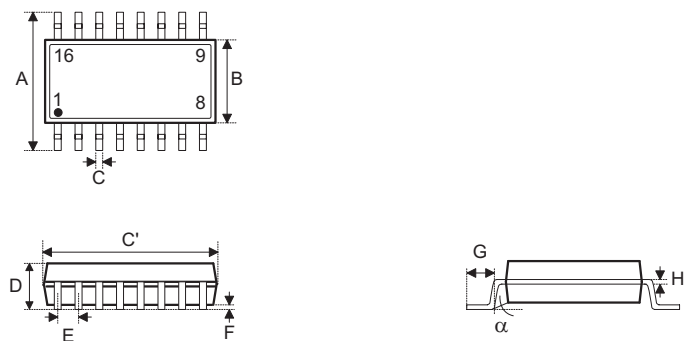
封装信息

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

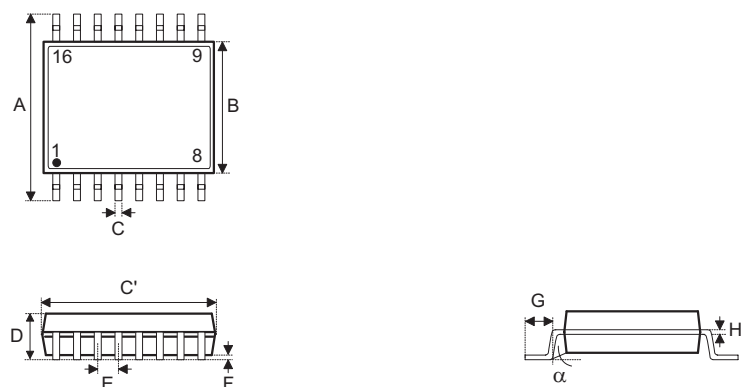
16-pin NSOP (150mil) 外形尺寸



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	—	6.00 BSC	—
B	—	3.90 BSC	—
C	0.31	—	0.51
C'	—	9.90 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°

16-pin SSOP (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	0.008	—	0.012
C'	—	0.193 BSC	—
D	—	—	0.069
E	—	0.025 BSC	—
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

符号	尺寸 (单位: mm)		
	最小	正常	最大
A	—	6.00 BSC	—
B	—	3.90 BSC	—
C	0.20	—	0.30
C'	—	4.90 BSC	—
D	—	—	1.75
E	—	0.635 BSC	—
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°

Copyright© 2019 by HOLTEK SEMICONDUCTOR INC.

使用指南中所出现的信息在出版当时相信是正确的，然而 Holtek 对于说明书的使用不负任何责任。文中提到的应用目的仅仅是用来做说明，Holtek 不保证或表示这些没有进一步修改的应用将是适当的，也不推荐它的产品使用在会由于故障或其它原因可能会对人身造成危害的地方。Holtek 产品不授权使用于救生、维生从机或系统中做为关键从机。Holtek 拥有不事先通知而修改产品的权利，对于最新的信息，请参考我们的网址 <http://www.holtek.com/zh/>