



接近感应 2-Key 触控 Flash 单片机

BS45F3332

版本: V1.01 日期: 2023-07-07

www.holtek.com

目录

特性	6
CPU 特性	6
周边特性	6
开发工具	7
概述	7
方框图	7
引脚图	8
引脚说明	9
极限参数	10
直流电气特性	11
工作电压特性	11
工作电流特性	11
待机电流特性	11
交流电气特性	12
内部高速 RC 振荡器 HIRC 频率精准度	12
内部低速振荡器 – LIRC – 频率精准度	12
工作频率电气特性曲线图	12
系统上电时间电气特性	13
输入 / 输出口电气特性	13
存储器电气特性	14
LVR 电气特性	15
内部参考电压特性	15
A/D 转换器电气特性	15
运算放大器电气特性	16
灌电流发生器电气特性	17
上电复位特性	18
系统结构	19
时序和流水线结构	19
程序计数器	20
堆栈	20
算术逻辑单元 – ALU	21
Flash 程序存储器	21
结构	21
特殊向量	22
查表	22
查表范例	22
在线烧录 – ICP	23
片上调试 – OCDS	24

数据存储器	25
结构	25
通用数据存储器	25
特殊功能数据存储器	25
特殊功能寄存器	27
间接寻址寄存器 – IAR0, IAR1	27
存储器指针 – MP0, MP1	27
累加器 – ACC	27
程序计数器低字节寄存器 – PCL	28
表格寄存器 – TBLP, TBHP, TBLH	28
状态寄存器 – STATUS	28
存储区指针 – BP	29
Option 存储器映射寄存器 – ORMC	30
EEPROM 数据存储器	30
EEPROM 数据存储器结构	30
EEPROM 寄存器	30
从 EEPROM 中读取数据	32
写数据到 EEPROM	32
写保护	32
EEPROM 中断	32
编程注意事项	32
振荡器	34
振荡器概述	34
系统时钟配置	34
内部高速 RC 振荡器 – HIRC	34
内部 32kHz 振荡器 – LIRC	35
工作模式和系统时钟	35
系统时钟	35
系统工作模式	36
控制寄存器	37
工作模式切换	38
待机电流注意事项	41
唤醒	41
看门狗定时器	42
看门狗定时器时钟源	42
看门狗定时器控制寄存器	42
看门狗定时器操作	43
复位和初始化	44
复位功能	44
复位初始状态	46
输入 / 输出端口	49
上拉电阻	49
PA 口唤醒	50
输入 / 输出端口控制寄存器	50

输入 / 输出端口源电流控制	51
引脚共用功能	51
输入 / 输出引脚结构	54
编程注意事项	54
定时器模块 – TM	55
简介	55
TM 操作	55
TM 时钟源	55
TM 中断	55
TM 外部引脚	55
编程注意事项	56
简易型 TM – CTM.....	57
简易型 TM 操作	57
简易型 TM 寄存器介绍	57
简易型 TM 工作模式	61
接近感应电路	67
接近感应电路操作	67
接近感应电路寄存器	67
失调校准步骤	70
灌电流发生器	71
灌电流发生器寄存器	71
A/D 转换器.....	73
A/D 简介	73
A/D 转换寄存器介绍	74
A/D 转换器操作	76
A/D 转换器参考电压	77
A/D 转换器输入信号	77
A/D 转换率及时序图	78
A/D 转换步骤概述	78
编程注意事项	79
A/D 转换功能	79
A/D 转换应用范例	80
触控按键功能	82
触控按键结构	82
触控按键寄存器定义	82
触控按键操作	86
触控按键中断	87
编程注意事项	87
中断	87
中断寄存器	87
中断操作	90
外部中断	91
触控按键中断	91
A/D 转换器中断	92

TM 中断	92
EEPROM 中断	92
时基中断	92
中断唤醒功能	93
编程注意事项	93
应用电路	94
8-pin SOP	94
16-pin NSOP	95
指令集	96
简介	96
指令周期	96
数据的传送	96
算术运算	96
逻辑和移位运算	96
分支和控制转换	97
位运算	97
查表运算	97
其它运算	97
指令集概要	98
惯例	98
指令定义	100
封装信息	111
8-pin SOP (150mil) 外形尺寸	112
16-pin NSOP (150mil) 外形尺寸	113

特性

CPU 特性

- 工作电压:
 - ◆ $f_{SYS}=8MHz$: 1.8V~5.5V
- $V_{DD}=5V$, 系统时钟为 8MHz 时, 指令周期为 $0.5\mu s$
- 提供暂停和唤醒功能, 以降低功耗
- 振荡器类型:
 - ◆ 内部高速 8MHz RC – HIRC
 - ◆ 内部低速 32kHz RC – LIRC
- 多种工作模式: 快速、低速、空闲和休眠
- 内部集成的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
- 61 条功能强大的指令系统
- 4 层堆栈
- 位操作指令

周边特性

- Flash 程序存储器: $2K \times 15$
- 数据存储器: 128×8
- True EEPROM 存储器: 32×8
- 2 个触控按键功能 – 完全集成而无需外接元件
- 看门狗定时器功能
- 13 个双向 I/O 口
- 1 个与 I/O 口共用的外部中断引脚
- 定时器模块用于时间测量、比较匹配输出及 PWM 输出功能
- 双时基功能, 可提供固定时间的中断信号
- 4 个外部通道 10-bit 分辨精度的 A/D 转换器, 带可编程内部参考电压 V_{VR}
- 双通道灌电流发生器提供红外线 LED 恒定驱动电流可达 320mA
- 接近感应电路
 - ◆ 一个运算放大器
- 低电压复位功能
- 封装类型: 8-pin SOP, 16-pin NSOP

开发工具

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

<https://www.holtek.com.cn/infrared-ir-proximity-sensing-workshop>

概述

该单片机是一款 A/D 型具有 8 位高性能精简指令集的 Flash 单片机, 内置接近感应电路和触控按键功能。

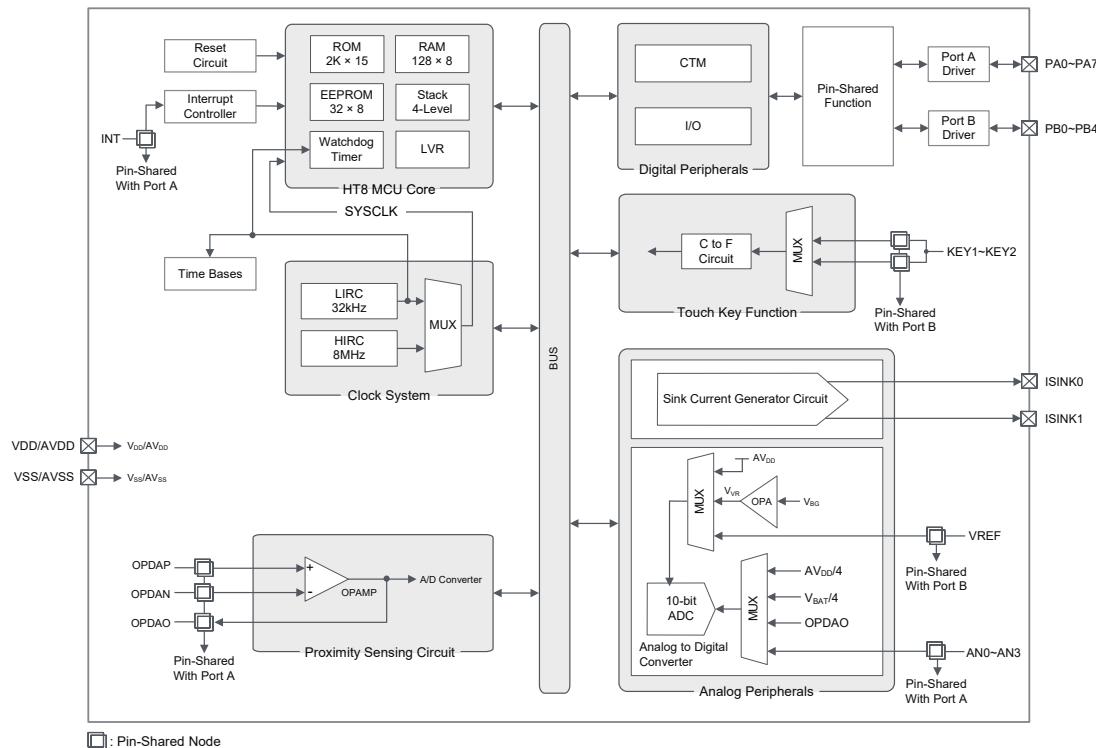
在存储器特性方面, Flash 存储器可多次编程的特性给用户提供了较大的方便。此外, 还包含了一个 RAM 数据存储器和一个可用于存储序号、校准数据等不易失性数据的 True EEPROM 存储器。

在模拟特性方面, 该单片机包含一个多通道 10-bit A/D 转换器, 一个接近感应电路主要包含一个运算放大器, 以及一个双通道灌电流发生器。其具有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。内部看门狗定时器和低电压复位等内部保护特性, 外加优秀的抗干扰和 ESD 保护性能, 确保单片机在恶劣的电磁干扰环境下可靠地运行。

该单片机提供内部高速和低速振荡器功能选项, 可灵活应用于不同程序。其在不同工作模式之间动态切换的能力, 为用户提供了一个优化单片机操作和减少功耗的手段。

外加 I/O 使用灵活, 时基功能和触控按键等其它特性, 使该单片机可以广泛应用于接近感应应用产品中。

方框图



引脚图

PB2/KEY1/VREF	1	8	PA1/OPDAO/AN1/CTP
ISINK0 & ISINK1	2	7	PA3/OPDAP
VSS/AVSS	3	6	PA0/INT/AN0/CTP/ICPDA/OCDSDA
VDD/AVDD	4	5	PA2/OPDAN/VBAT/ICPCK/OCDSDCK

BS45F3332/BS45V3332
8 SOP-A

PB1/CTPB	1	16	PA4/CTP
PB0/CTP	2	15	PA1/OPDAO/AN1/CTP
PB2/KEY1/VREF	3	14	PA2/OPDAN/VBAT/ICPCK/OCDSDCK
PB3/KEY2	4	13	PA3/OPDAP
ISINK0 & ISINK1	5	12	PA0/INT/AN0/CTP/ICPDA/OCDSDA
VSS/AVSS	6	11	PA6/OPDAP/AN2/CTPB
VDD/AVDD	7	10	PA5/CTCK
PA7/AN3	8	9	PA4/VBAT

BS45F3332/BS45V3332
16 NSOP-A

PB1/CTPB	1	16	PA4/CTP
PB0/CTP	2	15	PA1/OPDAO/AN1/CTP
PB2/KEY1/VREF	3	14	PA2/OPDAN/VBAT/ICPCK/OCDSDCK
ISINK1	4	13	PA3/OPDAP
ISINK0	5	12	PA0/INT/AN0/CTP/ICPDA/OCDSDA
VSS/AVSS	6	11	PA6/OPDAP/AN2/CTPB
VDD/AVDD	7	10	PA3/KEY2
PA7/AN3	8	9	PA4/VBAT

BS45F3332/BS45V3332
16 NSOP-B

- 注：
1. 若共用引脚同时有多种输出，所需引脚共用功能通过相应的软件控制位决定。
 2. OCDSDA 和 OCDSDCK 引脚为片上调试功能专用引脚，仅存在于 BS45F3332 的 OCDS EV 芯片 BS45V3332。
 3. 当 ISINK0 和 ISINK1 引脚打在同一位置 (8 SOP-A 或 16 NSOP-A) 或被用户自行外部接在一起 (16 NSOP-B)，使用时对应的内部开关和电流控制寄存器须合理设置以避免漏电流情况发生。详细请参考“灌电流发生器”章节。
 4. 在较小封装中可能含有未引出的引脚，需合理设置其状态以避免输入浮空造成额外耗电，详见“待机电流注意事项”和“输入 / 输出端口”章节。

引脚说明

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。由于该单片机存在不止一种封装，该表格反映的是较大封装类型的情况。

引脚名称	功能	OPT	I/T	O/T	说明
PA0/INT/AN0/CTP/ICPDA/OCDSDA	PA0	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	INT	PAS0 INTC0 INTEG	ST	—	外部中断输入
	AN0	PAS0	AN	—	A/D 转换器外部输入通道 0
	CTP	PAS0	—	CMOS	CTM 输出
	ICPDA	—	ST	CMOS	ICP 数据 / 地址
	OCDSDA	—	ST	CMOS	OCDS 数据 / 地址，仅用于 EV 芯片
PA1/OPDAO/AN1/CTP	PA1	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	OPDAO	PAS0	—	AN	OPAMP 输出
	AN1	PAS0	AN	—	A/D 转换器外部输入通道 1
	CTP	PAS0	—	CMOS	CTM 输出
PA2/OPDAN/VBAT/ICPCK/OCDSCK	PA2	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	OPDAN	PAS0	AN	—	OPAMP 反相输入
	VBAT	PAS0	AN	—	A/D 转换器外部输入
	ICPCK	—	ST	—	ICP 时钟引脚
	OCDSCK	—	ST	—	OCDS 时钟引脚，仅用于 EV 芯片
PA3/OPDAP	PA3	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	OPDAP	PAS0	AN	—	OPAMP 正相输入
PA4/VBAT	PA4	PAPU PAWU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	VBAT	PAS1	AN	—	A/D 转换器外部输入
PA5/CTCK	PA5	PAPU PAWU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	CTCK	—	ST	—	CTM 时钟输入
PA6/OPDAP/AN2/CTPB	PA6	PAPU PAWU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	OPDAP	PAS1	AN	—	OPAMP 正相输入
	AN2	PAS1	AN	—	A/D 转换器外部输入通道 2
	CTPB	PAS1	—	CMOS	CTM 反相输出

引脚名称	功能	OPT	I/T	O/T	说明
PA7/AN3	PA7	PAPU PAWU PAS1	ST	CMOS	通用 I/O 口, 可通过寄存器设置上拉电阻和唤醒功能
	AN3	PAS1	AN	—	A/D 转换器外部输入通道 3
PB0/CTP	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口, 可通过寄存器设置上拉电阻
	CTP	PBS0	—	CMOS	CTM 输出
PB1/CTPB	PB1	PBPU PBS0	ST	CMOS	通用 I/O 口, 可通过寄存器设置上拉电阻
	CTPB	PBS0	—	CMOS	CTM 反相输出
PB2/KEY1/VREF	PB2	PBPU PBS0	ST	CMOS	通用 I/O 口, 可通过寄存器设置上拉电阻
	KEY1	PBS0	AN	—	触控按键输入
	VREF	PBS0	AN	—	A/D 转换器外部参考电压输入
PB3/KEY2	PB3	PBPU PBS0	ST	CMOS	通用 I/O 口, 可通过寄存器设置上拉电阻
	KEY2	PBS0	AN	—	触控按键输入
PB4/CTP	PB4	PBPU PBS1	ST	CMOS	通用 I/O 口, 可通过寄存器设置上拉电阻
	CTP	PBS1	—	CMOS	CTM 输出
ISINK0	ISINK0	—	—	AN	Sink0 电流源
ISINK1	ISINK1	—	—	AN	Sink1 电流源
VDD/AVDD	VDD	—	PWR	—	数字正电源
	AVDD	—	PWR	—	模拟正电源
VSS/AVSS	VSS	—	PWR	—	数字负电源, 接地
	AVSS	—	PWR	—	模拟负电源, 接地

注: I/T: 输入类型;

O/T: 输出类型;

OPT: 通过寄存器选项来配置;

PWR: 电源;

ST: 施密特触发输入;

CMOS: CMOS 输出;

AN: 模拟信号。

极限参数

电源供应电压	V _{SS} -0.3V~6.0V
输入电压	V _{SS} -0.3V~V _{DD} +0.3V
储存温度	-60°C~150°C
工作温度	-40°C~85°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

以下表格中参数测量结果可能受多个因素影响，如振荡器类型、工作电压、工作频率、引脚负载状况、温度和程序指令等等。

工作电压特性

T_a=-40°C~85°C

符号	参数	测试条件	最小	典型	最大	单位
V _{DD}	工作电压 – HIRC	f _{SYS} =f _{HIRC} =8MHz	1.8	—	5.5	V
	工作电压 – LIRC	f _{SYS} =f _{LIRC} =32kHz	1.8	—	5.5	V

工作电流特性

T_a=25°C

符号	工作模式	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
I _{DD}	低速模式 – LIRC	1.8V	f _{SYS} =32kHz	—	8	16	μA
		3V		—	10	20	
		5V		—	30	50	
	快速模式 – HIRC	1.8V	f _{SYS} =8MHz	—	0.6	1.0	mA
		3V		—	0.8	1.2	
		5V		—	1.6	2.4	

注：当使用该表格电气特性数据时，以下几点需注意：

1. 任何数字输入都设置为非浮空的状态。
2. 所有测量都在无负载且所有外围功能关闭的条件下进行。
3. 无直流电流路径。
4. 所有工作电流数值都是通过连续的 NOP 指令循环测得。

待机电流特性

T_a=25°C，除非另有说明

符号	待机模式	测试条件		最小	典型	最大	最大 @85°C	单位
		V _{DD}	条件					
I _{STB}	休眠模式	1.8V	WDT off	—	0.11	0.15	2.00	μA
		3V		—	0.11	0.15	2.00	
		5V		—	0.18	0.38	2.90	
	空闲模式 0 – LIRC	1.8V	f _{SUB} on	—	2.4	4.0	4.8	μA
		3V		—	3	5	6	
		5V		—	5	10	12	
	空闲模式 1 – HIRC	1.8V	f _{SUB} on, f _{SYS} =8MHz	—	288	400	480	μA
		3V		—	360	500	600	
		5V		—	600	800	960	

注：当使用该表格电气特性数据时，以下几点需注意：

1. 任何数字输入都设置为非浮空的状态。
2. 所有测量都在无负载且所有外围功能关闭的条件下进行。
3. 无直流电流路径。
4. 所有待机电流数值都是在 HALT 指令执行后即停止执行所有指令后测得。

交流电气特性

以下表格中参数测量结果可能受多个因素影响，如振荡器类型、工作电压、工作频率和温度等等。

内部高速 RC 振荡器 HIRC 频率精准度

程序烧录时，烧录器会依据用户选择的 HIRC 频率和工作电压 (3V 或 5V) 对 HIRC 进行频率精准度调整。

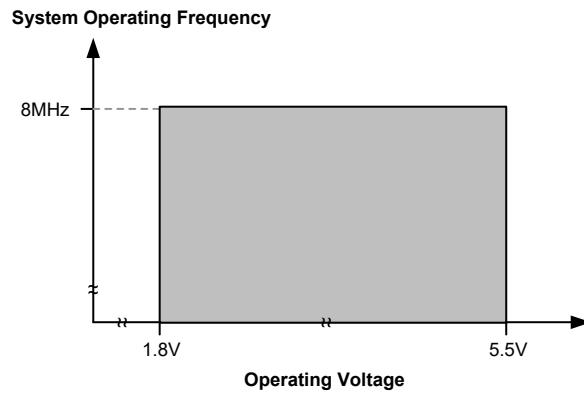
符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	温度				
f _{HIRC}	通过烧录器调整后的 8MHz HIRC 频率	3V/5V	25°C	-1%	8	+1%	MHz
			-40°C~85°C	-3%	8	+3%	
		2.2V~5.5V	25°C	-2.5%	8	+2.5%	
			-40°C~85°C	-4%	8	+4%	
		1.8V~5.5V	25°C	-5%	8	+5%	
			-40°C~85°C	-10%	8	+10%	

- 注：1. 烧录器可在 3V/5V 这两个可选的固定电压下对 HIRC 频率进行调整，在此提供 V_{DD}=3V/5V 时的参数值。
2. 3V/5V 表格列下面提供的是全压条件下的参数值。对于电压范围在 1.8V~3.6V 的应用，建议烧录器电压固定在 3V；对于电压范围在 3.3V~5.5V 的应用，建议烧录器电压固定在 5V。

内部低速振荡器 – LIRC – 频率精准度

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	温度				
f _{LIRC}	LIRC 频率	1.8V~5.5V	-40°C~85°C	-7%	32	+7%	kHz
t _{START}	LIRC 启动时间	—	-40°C~85°C	—	—	100	μs

工作频率电气特性曲线图



系统上电时间电气特性

Ta=-40°C~85°C

符号	参数	测试条件	最小	典型	最大	单位
t _{SST}	系统启动时间 (从 f _{SYS off} 的状态下唤醒)	f _{SYS} =f _H ~f _H /64, f _H =f _{HIRC}	—	16	—	t _{HIRC}
		f _{SYS} =f _{SUB} =f _{LIRC}	—	2	—	t _{LIRC}
	系统启动时间 (从 f _{SYS on} 的状态下唤醒)	f _{SYS} =f _H ~f _H /64, f _H =f _{HIRC}	—	2	—	t _H
	系统速度切换时间 (快速模式 → 低速模式或 低速模式 → 快速模式)	f _{HIRC off} → on	—	16	—	t _{HIRC}
t _{RSTD}	系统复位延迟时间 (上电复位或 LVR 硬件复位)	RR _{POR} =5V/ms	14	16	18	ms
	系统复位延迟时间 (LVRC/WDT 寄存器软件复位)	—				
	系统复位延迟时间 (WDT 溢出复位)	—		14	16	18
t _{SRESET}	软件复位最小延迟脉宽	—	45	90	120	μs

- 注：1. 系统启动时间里提到的 f_{SYS on/off} 状态取决于工作模式类型以及所选的系统时钟振荡器。更多相关细节请参考系统工作模式章节。
2. t_{HIRC} 等符号所表示的时间单位，是对应频率值的倒数，相关频率值在前面表格有说明。例如，
t_{HIRC}=1/f_{HIRC}, t_{SYS}=1/f_{SYS} 等等。
3. 若 LIRC 被选择作为系统时钟源且在休眠模式下 LIRC 关闭，则上面表格中对应 t_{SST} 数值还需加上
LIRC 频率表格里提供的 LIRC 启动时间 t_{START}。
4. 系统速度切换时间实际上是指新使能的振荡器的启动时间。

输入 / 输出口电气特性

Ta=-40°C~85°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{IL}	I/O 口低电平输入电压	5V	—	0	—	1.5	V
		—		0	—	0.2V _{DD}	
V _{IH}	I/O 口高电平输入电压	5V	—	3.5	—	5.0	V
		—		0.8V _{DD}	—	V _{DD}	
I _{OL}	I/O 口灌电流	3V	V _{OL} =0.1V _{DD}	16	32	—	mA
		5V		32	65	—	

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
I _{OH}	I/O 口源电流 (PB0~PB4)	3V	V _{OH} =0.9V _{DD}	-4	-8	—	mA
		5V		-8	-16	—	
	I/O 口源电流 (PA0~PA7)	3V	V _{OH} =0.9V _{DD} , SLEDC[m+1:m]=00B (m=0, 2, 4, 6)	-0.7	-1.5	—	mA
		5V		-1.5	-2.9	—	mA
		3V	V _{OH} =0.9V _{DD} , SLEDC[m+1:m]=01B (m=0, 2, 4, 6)	-1.3	-2.5	—	mA
		5V		-2.5	-5.1	—	mA
		3V	V _{OH} =0.9V _{DD} , SLEDC[m+1:m]=10B (m=0, 2, 4, 6)	-1.8	-3.6	—	mA
		5V		-3.6	-7.3	—	mA
		3V	V _{OH} =0.9V _{DD} , SLEDC[m+1:m]=11B (m=0, 2, 4, 6)	-4	-8	—	mA
		5V		-8	-16	—	mA
R _{PH}	I/O 口上拉电阻 (注)	3V	LVPU=0, PxPU=FFH (Px=PA, PB)	20	60	100	kΩ
		5V		10	30	50	
		3V	LVPU=1, PxPU=FFH (Px=PA, PB)	6.67	15.00	23.00	
		5V		3.5	7.5	12.0	
I _{LEAK}	输入漏电流	5V	V _{IN} =V _{DD} 或 V _{IN} =V _{SS}	—	—	±1	μA
t _{TCK}	CTCK 输入引脚最小脉宽	—	—	0.3	—	—	μs
t _{INT}	中断引脚最小脉宽	—	—	10	—	—	μs

注：R_{PH} 内部上拉电阻值的计算方法是：将引脚接地并设置为输入且使能上拉电阻功能，然后在特定电源电压下测量该引脚上的电流，最后电压除以测量的电流值从而得到此上拉电阻值。

存储器电气特性

T_a=-40°C~85°C，除非另有说明。

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DD}	写工作电压 – 数据 EEPROM 存储器	—	—	2.2	—	5.5	V
t _{DEW}	擦除 / 写周期时间 – Flash 程序存储器	—	—	—	2	3	ms
t _{ERD}	读周期时间	—	—	—	—	4	t _{SYS}
I _{DDPGM}	V _{DD} 电压下烧录 / 擦除电流	—	—	—	—	5.0	mA
E _P	存储单元耐久性 – Flash 程序存储器	—	—	10K	—	—	E/W
	存储单元耐久性 – 数据 EEPROM 存储器	—	—	100K	—	—	
t _{RETD}	程序存储器数据保存时间	—	T _a =25°C	—	40	—	Year

注：“E/W” 表示擦 / 写次数。

LVR 电气特性

Ta=-40°C~85°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DD}	工作电压	—	—	1.8	—	5.5	V
V _{LVR}	低电压复位电压	—	LVR 使能	-5%	1.7	+5%	V
I _{LVRBG}	工作电流	3V	LVR 使能, V _{LVR} =1.7V	—	—	15	μA
		5V		—	15	25	
t _{LVR}	产生 LVR 复位的低电压最短保持时间	—	—	120	240	480	μs
I _{LVR}	LVR 使能的额外电流	5V	VBGEN=0	—	—	25	μA

内部参考电压特性

Ta=-40°C~85°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
t _{BGS}	V _{BG} 启动稳定时间	—	无负载	—	—	50	μs
I _{BG}	Bandgap 参考使能的额外电流	—	VBGEN=1, LVR 除能	—	—	2	μA

注: V_{BG} 电压可用作 A/D 转换器内部信号输入。

A/D 转换器电气特性

Ta=-40°C~85°C, 除非另有说明

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DD}	工作电压	—	—	1.8	—	5.5	V
V _{ADI}	输入电压	—	—	0	—	V _{REF}	V
V _{REF}	参考电压	—	—	1.6	—	V _{DD}	V
N _R	分辨精度	—	—	—	—	10	Bit
DNL	非线性微分误差	—	V _{REF} =V _{DD} , t _{ADCK} =0.5μs	-1.5	—	1.5	LSB
INL	非线性积分误差	—	V _{REF} =V _{DD} , t _{ADCK} =0.5μs	-2	—	2	LSB
I _{ADC}	A/D 转换器使能的额外电流	1.8V	无负载, t _{ADCK} =0.5μs	—	300	420	μA
		3V		—	340	500	
		5V		—	500	700	
t _{ADCK}	时钟周期	—	—	0.5	—	10.0	μs
t _{ON2ST}	A/D 转换器 On-to-Start 时间	—	—	4	—	—	μs
t _{ADS}	采样时间	—	—	—	4	—	t _{ADCK}
t _{ADC}	A/D 转换时间 (包括采样和保持时间)	—	—	—	14	—	t _{ADCK}
GERR	A/D 转换增益误差	—	V _{REF} =V _{DD}	-2	—	2	LSB
OSRR	A/D 转换失调误差	—	V _{REF} =V _{DD}	-2	—	2	LSB

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
I _{PGA}	OPA 使能的额外电流	3V	无负载	—	390	550	μA
		5V		—	500	650	
V _{OR}	OPA 最大输出电压范围	3V	—	V _{SS} +0.1	—	V _{DD} -0.1	V
		5V		V _{SS} +0.1	—	V _{DD} -0.1	
V _{VR}	OPA 固定电压输出	1.8V~5.5V	Ta=25°C	-5%	1.6	+5%	V
		1.8V~5.5V	Ta=-40°C~85°C	-10%	1.6	+10%	
		3.2V~5.5V	Ta=25°C	-5%	3.0	+5%	
		3.2V~5.5V	Ta=-40°C~85°C	-10%	3.0	+10%	
		4.2V~5.5V	Ta=25°C	-5%	4.0	+5%	
		4.2V~5.5V	Ta=-40°C~85°C	-10%	4.0	+10%	

运算放大器电气特性

Ta=-40°C~85°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DD}	工作电压	—	—	1.8	5.0	5.5	V
I _{OPA}	OPA 使能的额外电流	5V	无负载, OPDABW=0	—	80	200	μA
			无负载, OPDABW=1	—	230	510	
V _{os}	输入失调电压	5V	未校准 (OPDAOF[5:0]=100000B)	-15	—	+15	mV
			已校准	-2	—	+2	
I _{os}	输入失调电流	5V	V _{IN} =1/2 V _{CM}	—	1	10	nA
V _{CM}	共模电压范围	—	—	V _{SS}	—	V _{DD} -1.4	V
PSRR	电源电压抑制比	5V	—	50	70	—	dB
CMRR	共模抑制比	5V	—	50	80	—	dB
A _{OL}	开环增益	—	—	60	80	—	dB
SR	转换速率	5V	R _{LOAD} =1MΩ, C _{LOAD} =60pF, OPDABW=0	180	500	—	V/ms
			R _{LOAD} =1MΩ, C _{LOAD} =60pF, OPDABW=1	600	1800	—	
GBW	增益带宽	5V	R _{LOAD} =1MΩ, C _{LOAD} =60pF, OPDABW=0	250	600	—	kHz
			R _{LOAD} =1MΩ, C _{LOAD} =60pF, OPDABW=1	800	2000	—	
V _{OR}	最大输出电压范围	—	R _{LOAD} =5kΩ 接到 V _{DD} /2 处	V _{SS} +120	—	V _{DD} -180	mV

灌电流发生器电气特性

Ta=-40°C~85°C, 除非另有说明

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DD}	工作电压	—	—	1.8	—	5.5	V
I _{SINK0}	I _{SINK0} 引脚灌电流	5V	(调整后) Ta=25°C, V _{I_{SINK0}} =3.0V, ISGDATA0[3:0]=0000B, ISST0=0	1.90	2.00	2.10	mA
		—	(调整后) Ta=-40°C~85°C, V _{I_{SINK0}} =1.0V~4.5V, ISGDATA0[3:0]=0000B, ISST0=0	1.64	2.00	2.36	
		—	(调整后) Ta=-40°C~85°C, V _{I_{SINK0}} =0.7V~1.0V, ISGDATA0[3:0]=0000B, ISST0=0	1.40	2.00	2.10	
		5V	(调整后) Ta=25°C, V _{I_{SINK0}} =3.0V, ISGDATA0[3:0]=1111B, ISST0=1	147	160	173	
		—	(调整后) Ta=-40°C~85°C, V _{I_{SINK0}} =1.0V~4.5V, ISGDATA0[3:0]=1111B, ISST0=1	128	160	189	
		—	(调整后) Ta=-40°C~85°C, V _{I_{SINK0}} =0.7V~1.0V, ISGDATA0[3:0]=1111B, ISST0=1	112	160	173	

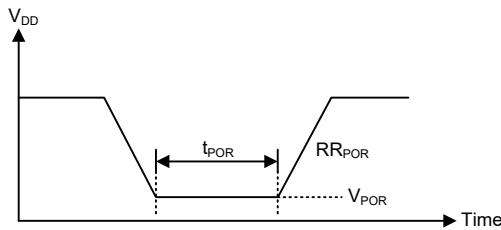
符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
ISINK1	ISINK1 引脚灌电流	5V	(调整后) Ta=25°C, V _{ISINK1} =3.0V, ISGDATA1[3:0]=0000B, ISST1=0	1.80	2.00	2.20	mA
		—	(调整后) Ta=-40°C~85°C, V _{ISINK1} =1.0V~4.5V, ISGDATA1[3:0]=000B, ISST1=0	1.64	2.00	2.36	
		—	(调整后) Ta=-40°C~85°C, V _{ISINK1} =0.7V~1.0V, ISGDATA1[3:0]=0000B, ISST1=0	1.40	2.00	2.20	
		5V	(调整后) Ta=25°C, V _{ISINK1} =3.0V, ISGDATA1[3:0]=1111B, ISST1=1	144	160	176	
		—	(调整后) Ta=-40°C~85°C, V _{ISINK1} =1.0V~4.5V, ISGDATA1[3:0]=1111B, ISST1=1	128	160	189	
		—	(调整后) Ta=-40°C~85°C, V _{ISINK1} =0.7V~1.0V, ISGDATA1[3:0]=1111B, ISST1=1	112	160	176	

注：当 ISINK0 和 ISINK1 引脚打在同一位置或被用户自行外部接在一起，使用时对应的内部开关和电流控制寄存器须合理设置以避免漏电流情况发生。详细请参考“灌电流发生器”章节。

上电复位特性

Ta=-40°C~85°C

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



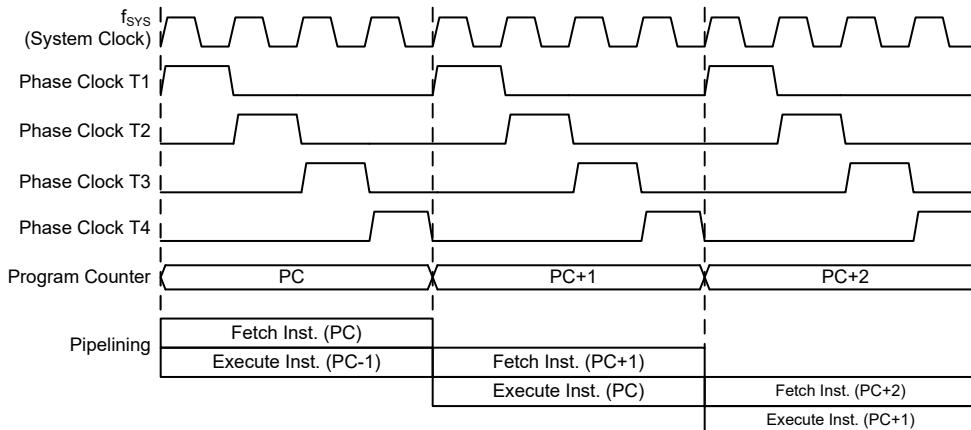
系统结构

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

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



系统时序和流水线

1	MOV A,[12H]	Fetch Inst. 1	Execute Inst. 1				
2	CALL DELAY	Fetch Inst. 2	Execute Inst. 2				
3	CPL [12H]		Fetch Inst. 3	Flush Pipeline			
4	:			Fetch Inst. 6	Execute Inst. 6		
5	:				Fetch Inst. 7		
6	DELAY: NOP						

指令捕捉

程序计数器

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

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

程序计数器	
程序计数器高字节	PCL 寄存器
PC10~PC8	PCL7~PCL0

程序计数器

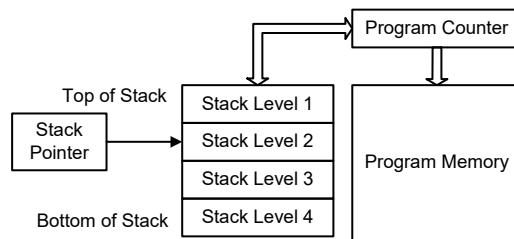
程序计数器的低字节，即程序计数器的低字节寄存器 PCL，可以通过程序控制，且它是可以读取和写入的寄存器。通过直接写入数据到这个寄存器，一个程序短跳转可直接执行，然而只有低字节的操作是有效的，跳转被限制在存储器的当前页中，即 256 个存储器地址范围内，当这样一个程序跳转要执行时，会插入一个空指令周期。PCL 的使用可能引起程序跳转，因此需要额外的指令周期。

堆栈

堆栈是一个特殊的存储空间，用来存储程序计数器中的内容。该单片机有 4 层堆栈，堆栈既不是数据部分也不是程序空间部分，而且它既不是可读取也不是可写入的。当前层由堆栈指针 (SP) 加以指示，同样也是不可读写的。在子程序调用或中断响应服务时，程序计数器的内容被压入到堆栈中。当子程序或中断响应结束时，返回指令 (RET 或 RETI) 使程序计数器从堆栈中重新得到它以前的值。当一个芯片复位后，堆栈指针将指向堆栈顶部。

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

若堆栈溢出，则首个存入堆栈的程序计数器数据将会丢失。



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，而相关状态寄存器会因此更新内容以显示这些改变，ALU 所提供的功能如下：

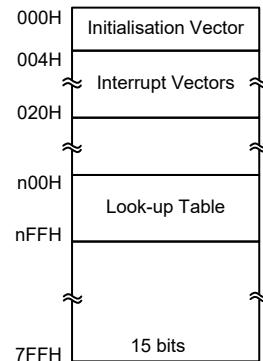
- 算术运算：
ADD, ADDM, ADC, ADCM, SUB, SUBM, SBC, SBCM, DAA
- 逻辑运算：
AND, OR, XOR, ANDM, ORM, XORM, CPL, CPLA
- 移位运算：
RRA, RR, RRCA, RRC, RLA, RL, RLCA, RLC
- 递增和递减：
INCA, INC, DECA, DEC
- 分支判断：
JMP, SZ, SZA, SNZ, SIZ, SDZ, SIZA, SDZA, CALL, RET, RETI

Flash 程序存储器

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

结构

程序存储器的容量为 $2K \times 15$ 位，程序存储器用程序计数器来寻址，其中也包含数据、表格和中断入口。数据表格可以设定在程序存储器的任何地址，由表格指针来寻址。



程序存储器结构

特殊向量

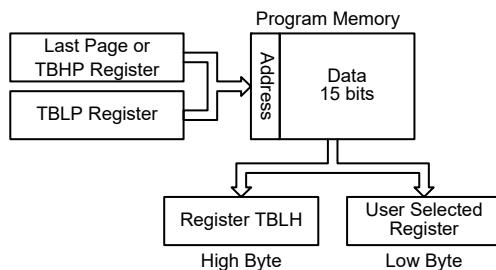
程序存储器内部某些地址保留用做诸如复位和中断入口等特殊用途。地址 000H 是芯片复位后的程序起始地址。在芯片复位之后，程序将跳到这个地址并开始执行。

查表

程序存储器中的任何地址都可以定义成一个表格，以便储存固定的数据。使用表格时，表格指针必须先行设定，其方式是将表格的地址放在表格指针寄存器 TBLP 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，表格数据可以使用如“TABRD [m]”或“TABRDL [m]”等指令分别从程序存储器查表读取。当这些指令执行时，程序存储器中表格数据低字节，将被传送到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。

下图是查表中寻址 / 数据流程：



查表范例

以下范例说明表格指针和表格数据如何被定义和执行。这个例子使用的表格数据用 ORG 伪指令储存在存储器中。ORG 指令的值“700H”指向的地址是 2K 程序存储器中最后一页的起始地址。表格指针低字节寄存器的初始值设为 06H，这可保证从数据表格读取的第一笔数据位于程序存储器地址 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 TBLP 和 TBHP 指定的地址。在这个例子中，表格数据的高字节等于零，而当“TABRD [m]”指令被执行时，此值将会自动的被传送到 TBLH 寄存器。

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

表格读取程序范例

```

tempreg1 db ?      ; temporary register #1
tempreg2 db ?      ; temporary register #2
:
:
mov a,06h          ; initialise low table pointer - note that this address
                    ; is referenced
mov tblp,a          ; to the last page or the page that tbhp pointed
mov a,0Fh          ; initialise high table pointer

```

```

mov tbhp, a
:
:
tabrd tempreg1      ; transfers value in table referenced by table pointer
; data at program memory address "706H" transferred to
; tempreg1 and TBLH
dec tbhp             ; 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 and the
; data "00H" is transferred to TBLH
:
:
org 700h             ; sets initial address of program memory
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
:
:

```

在线烧录 – ICP

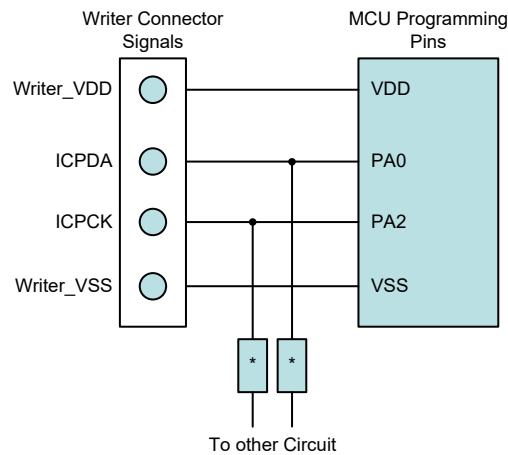
Flash 型程序存储器提供用户便利地对同一芯片进行程序的更新和修改。另外，Holtek 单片机提供 4 线接口的在线烧录方式。用户可将进行过烧录或未经过烧录的单片机芯片连同电路板一起制成，最后阶段进行程序的更新和程序的烧写，在无需拔出再重新插入芯片的情况下方便地保持程序为最新版。

Holtek Flash MCU 与烧录器引脚对应表如下：

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

程序存储器可以通过 4 线的接口在线进行烧录。其中一条线用于数据串行下载或上传、一条线用于串行时钟、剩下两条用于提供电源。芯片在线烧写的详细使用说明超出此文档的描述范围，将由专门的参考文献提供。

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



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

片上调试 – OCDS

EV 芯片用于单片机仿真。此 EV 芯片提供片上调试功能 (On-Chip Debug) 用于开发过程中的实际单片机调试。除了片上调试功能方面，EV 芯片和实际单片机在功能上几乎是兼容的。用户可将 OCDSDA 和 OCDSCK 引脚连接至 Holtek HT-IDE 开发工具，从而实现 EV 芯片对实际单片机的仿真。OCDSDA 引脚为 OCDS 数据 / 地址输入 / 输出脚，OCDSCK 引脚为 OCDS 时钟输入脚。当用户用 EV 芯片进行调试时，实际单片机 OCDSDA 和 OCDSCK 引脚上的其它共用功能无效。由于这两个 OCDS 引脚与 ICP 引脚共用，因此在线烧录时仍用作 Flash 存储器烧录引脚。关于 OCDS 功能的详细描述，请参考“Holtek e-Link for 8-bit MCU OCDS 使用手册”文件。

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

数据存储器

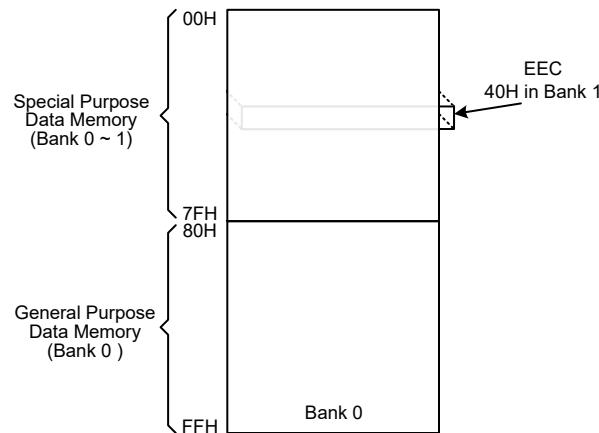
数据存储器是内容可更改的 8 位 RAM 内部存储器，用来储存临时数据。

结构

数据存储器分为两部分，第一部分是特殊功能数据存储器。这些寄存器有固定的地址且与单片机的正确操作密切相关。大多特殊功能寄存器都可在程序控制下直接读取和写入，但有些被加以保护而不对用户开放。第二部分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。

数据存储器被分为两个 Bank，都位于 8 位存储器中。切换不同区域可通过设置存储区指针 (BP) 实现。特殊功能数据存储器地址范围为 00H~7FH，而通用数据存储器地址范围为 80H~FFH。

特殊功能数据存储器		通用数据存储器	
所在 Bank	容量	Bank: 地址	
0: 00H~7FH	128×8	0: 80H~FFH	
1: 40H (仅 EEC)			



数据存储器结构

通用数据存储器

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

特殊功能数据存储器

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

Bank 0	Bank 1	Bank 0	Bank 1
00H	IAR0	40H	
01H	MP0	41H	OPSWA
02H	IAR1	42H	OPSWB
03H	MP1	43H	OPSWC
04H	BP	44H	OPDC
05H	ACC	45H	OPDACL
06H	PCL	46H	SADC0
07H	TBLP	47H	SADC1
08H	TBLH	48H	SAD0H
09H	TBHP	49H	SAD0L
0AH	STATUS	4AH	ORMC
0BH	SCC	4BH	
0CH	HIRCC		
0DH	LVRC		
0EH	INTEG		
0FH	RSTFC		
10H	INTC0		
11H	INTC1		
12H	INTC2		
13H			
14H	PA		
15H	PAC		
16H	PAPU		
17H	PAWU		
18H	PB		
19H	PBC		
1AH	PBPU		
1BH			
1CH			
1DH	SLEDC		
1EH	VBGC		
1FH	WDTC		
20H	PSC0R		
21H	TB0C		
22H	PSC1R		
23H	TB1C		
24H	EEA		
25H	EED		
26H	PAS0		
27H	PAS1		
28H	PBS0		
29H	PBS1		
2AH			
2BH	TKTMR		
2CH	TKC0		
2DH	TKC1		
2EH	TK16DL		
2FH	TK16DH		
30H	TKM0C0		
31H	TKM0C1		
32H	TKM016DL		
33H	TKM016DH		
34H	TKM0ROL		
35H	TKM0ROH		
36H	CTMC0		
37H	CTMC1		
38H	CTMDL		
39H	CTMDH		
3AH	CTMAL		
3BH	CTMAH		
3CH	ISGENC		
3DH	ISGDATA0		
3EH	ISGDATA1		
3FH	LVPUC		

■ : Unused, read as 00H

特殊功能数据存储器结构

特殊功能寄存器

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

间接寻址寄存器 – IAR0, IAR1

间接寻址寄存器 IAR0 和 IAR1 的地址虽位于数据存储区域，但不同于普通寄存器，它们没有实际的物理地址。与定义实际存储器地址的直接存储器寻址不同，间接寻址是使用间接寻址寄存器和存储器指针来执行存储器数据操作。在间接寻址寄存器 IAR0 和 IAR1 上的任何动作，将对存储器指针 MP0 或 MP1 所指定的存储器地址产生对应的读 / 写操作。它们总是成对出现，IAR0 和 MP0 可以访问 Bank 0，而 IAR1 和 MP1 可以访问任何 Bank。因为这些间接寻址寄存器不是实际存在的，直接读取将返回 “00H”的结果，而直接写入此寄存器则不做任何操作。

存储器指针 – MP0, MP1

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

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

间接寻址程序范例

```

data .section 'data'
adres1 db ?
adres2 db ?
adres3 db ?
adres4 db ?
block db ?
code .section at 0 'code'
org 00h
start:
    mov a, 04h           ; setup size of block
    mov block, a
    mov a, offset adres1 ; Accumulator loaded with first RAM address
    mov mp0, a            ; setup memory pointer with first RAM address
loop:
    clr IAR0             ; clear the data at address defined by MP0
    inc mp0               ; increase 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。

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、C、SC 和 CZ 标志位通常反映最近运算的状态。

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

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

- STATUS 寄存器

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

“x”：未知

Bit 7~6 未定义，读为“0”

Bit 5 **TO**：看门狗溢出标志位

0: 系统上电或执行“CLR WDT”或“HALT”指令后
1: 看门狗溢出发生

Bit 4 **PDF**：暂停标志位

0: 系统上电或执行“CLR WDT”指令后
1: 执行“HALT”指令

Bit 3 **OV**：溢出标志位

0: 无溢出
1: 运算结果高两位的进位状态异或结果为1

Bit 2 **Z**：零标志位

0: 算术或逻辑运算结果不为0
1: 算术或逻辑运算结果为0

Bit 1 **AC**：辅助进位标志位

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

Bit 0 **C**：进位标志位

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

存储区指针 – BP

数据存储器被分为两个 Bank，即 Bank 0 和 Bank 1。可以通过设置存储区指针 BP 寄存器的 Bit 0 来选择数据存储区。

复位后，数据存储器会初始化到 Bank 0，但是在空闲或休眠模式下的 WDT 溢出复位不会改变数据存储器的存储区号。数据存储器的直接寻址总是访问 Bank 0，不受存储区指针值的影响。要访问 Bank 1 则必须要使用间接寻址方式。

- BP 寄存器

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

Bit 7~1 未定义，读为“0”

Bit 0 **DMBPO**：数据存储区选择

0: Bank 0
1: Bank 1

Option 存储器映射寄存器 – ORMC

ORMC 寄存器用于使能 Option 存储器映射功能。Option 存储器的容量为 32 个字。当连续写入特定数据序列 55H 和 AAH 到该寄存器，Option 存储器映射功能将使能，通过使用查表指令即可读到 Option 存储器的内容，Option 存储器的 00H~1FH 地址会一一对应到程序存储器最后一页的 E0H~FFH 地址。

要成功使能 Option 存储器映射功能，该特定的数据序列 55H 和 AAH 必须在两个指令周期内连续写入。建议在写入该特定数据序列前应当先将总中断位 EMI 清零，在数据序列成功写入后，根据用户的需求在适当的时间再将其置高。当数据序列成功写入时会启动内部定时器， $4 \times t_{LIRC}$ 时间之后会自动结束映射。因此，用户需及时读出数据，否则需要重新启动 Option 存储器映射功能。每次 ORMC 寄存器被连续写入后，定时器都会重新计数。

当使用查表指令来读取 Option 存储器内容时，“TABRD [m]”和“TABRDL [m]”指令皆可使用。然而，若使用“TABRD [m]”指令来读取，必须配置 TBHP 寄存器将表格指针设定在最后一页。更多查表的描述请参考相关章节。

• ORMC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	ORMC7	ORMC6	ORMC5	ORMC4	ORMC3	ORMC2	ORMC1	ORMC0
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 **ORMC7~ORMC0:** Option 存储器映射特定数据序列

当将特定数据序列 55H 和 AAH 连续写入该寄存器，会使能 Option 存储器映射功能。需注意，单片机从空闲 / 休眠模式唤醒后，该寄存器的内容将被清除。

EEPROM 数据存储器

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

EEPROM 数据存储器结构

该单片机的 EEPROM 数据存储器容量为 32×8 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 中的一个地址寄存器和一个数据寄存器以及 Bank 1 中的一个控制寄存器，可以实现对 EEPROM 的单字节读写操作。

EEPROM 寄存器

有三个寄存器控制内部 EEPROM 数据存储器总的操作。地址寄存器 EEA、数据寄存器 EED 及控制寄存器 EEC。EEA 和 EED 位于 Bank 0 中，它们能像其它特殊功能寄存器一样直接被访问。EEC 位于 Bank 1 中，只能通过存储器指针 MP1 和间接寻址寄存器 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”，存储区指针寄存器 BP 被设为“01H”。

寄存器 名称	位							
	7	6	5	4	3	2	1	0
EEA	—	—	—	EEA4	EEA3	EEA2	EEA1	EEA0
EED	EED7	EED6	EED5	EED4	EED3	EED2	EED1	EED0
EEC	D7	—	—	—	WREN	WR	RDEN	RD

EEPROM 寄存器列表

● EEA 寄存器

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

Bit 7~5 未定义, 读为“0”

Bit 4~0 **EEA4~EEA0:** 数据 EEPROM 地址 Bit 4 ~ Bit 0

● EED 寄存器

Bit	7	6	5	4	3	2	1	0
Name	EED7	EED6	EED5	EED4	EED3	EED2	EED1	EED0
R/W								
POR	0	0	0	0	0	0	0	0

Bit 7~0 **EED7~EED0:** 数据 EEPROM 数据 Bit 7 ~ Bit 0

● EEC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	D7	—	—	—	WREN	WR	RDEN	RD
R/W	R/W	—	—	—	R/W	R/W	R/W	R/W
POR	0	—	—	—	0	0	0	0

Bit 7 **D7:** 保留位, 必须固定为“0”

Bit 6~4 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: 写周期开启

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: 读周期开启
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: 1. 在同一条指令中 WREN、WR、RDEN 和 RD 不能同时置为“1”。 2. 确保 f_{SUB} 时钟在执行写动作前已稳定。 3. 确保写动作完成后才可改写 EEPROM 相关寄存器。	

从 EEPROM 中读取数据

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

写数据到 EEPROM

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

写保护

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

EEPROM 中断

EEPROM 写周期结束后将产生 EEPROM 写中断, 需先通过设置相关中断寄存器的 DEE 位使能 EEPROM 中断。当 EEPROM 写周期结束, DEF 请求标志位将被置位。若总中断和 EEPROM 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, EEPROM 中断标志位将自动复位, 且 EMI 位将自动清零以除能其它中断。更多内容请参考“中断”章节。

编程注意事项

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

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

程序举例

从 EEPROM 中读取数据 – 轮询法

```

MOV A, EEPROM_ADRES      ; user defined address
MOV EEA, A
MOV A, 40H                 ; setup memory pointer low byte 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 if no more read operations
                           ; are required
CLR BP
MOV A, EED                  ; move read data to register
MOV READ_DATA, A

```

注：对于每一个读操作，即使地址是连续的，都必须重新设置地址寄存器，接着再将 RD 位置高开启一个读周期。

写数据到 EEPROM – 轮询法

```

MOV A, EEPROM_ADRES      ; user defined address
MOV EEA, A
MOV A, EEPROM_DATA        ; user defined data
MOV EED, A
MOV A, 40H                 ; setup memory pointer low byte MP1
MOV MP1, A                 ; MP1 points to EEC register
MOV A, 01H                 ; setup Bank Pointer
MOV BP, A
CLR EMI
SET IAR1.3                 ; set WREN bit, enable write operations
SET IAR1.2                 ; start Write Cycle - set WR bit - executed
                           ; immediately after set WREN bit
SET EMI
BACK:
SZ IAR1.2                 ; check for write cycle end
JMP BACK
CLR 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。振荡器选择是通过应用程序控制寄存器完成的。

振荡器概述

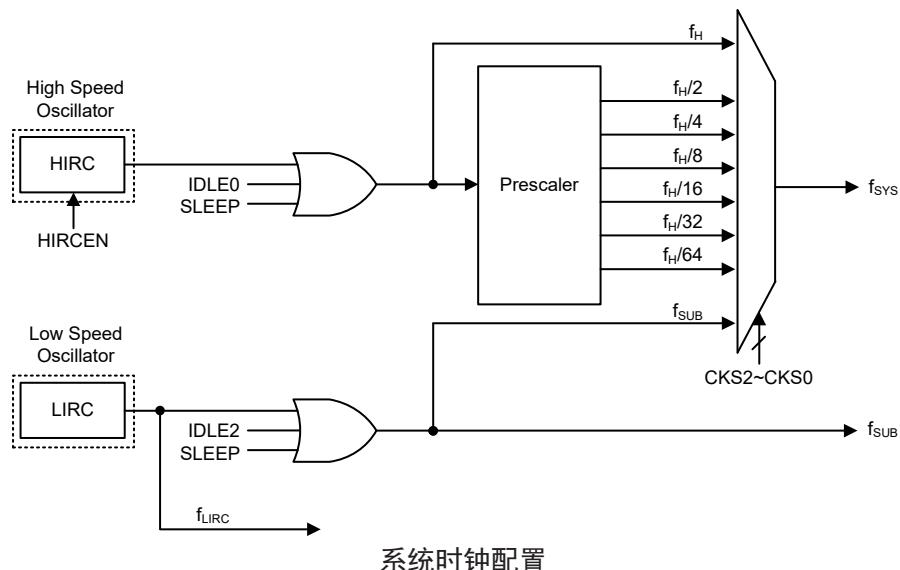
振荡器除了作为系统时钟源，还作为看门狗定时器和时基功能的时钟源。两个完全集成的内部振荡器不需要任何外围器件。它们提供的高速和低速系统振荡器具有较宽的频率范围。较高频率的振荡器提供更高的性能，但要求有更高的功率，反之亦然。动态切换快慢系统时钟的能力使单片机具有灵活而优化的性能 / 功耗比，此特性对功耗敏感的应用领域尤为重要。

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

振荡器类型

系统时钟配置

此单片机有两个系统振荡器，包括一个高速振荡器和一个低速振荡器。高速振荡器有内部 8MHz RC 振荡器 – HIRC，低速振荡器有内部 32kHz 低速振荡器 – 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 SCC 寄存器中的 CKS2~CKS0 位决定的，系统时钟可动态选择。



系统时钟配置

内部高速 RC 振荡器 – HIRC

内部 RC 振荡器是一个集成的系统振荡器，无需其它外部器件。内部 RC 振荡器固定频率为 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的降低。

内部 32kHz 振荡器 – LIRC

内部 32kHz 系统振荡器是一个完全集成 RC 振荡器，它的典型频率值为 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的降低。

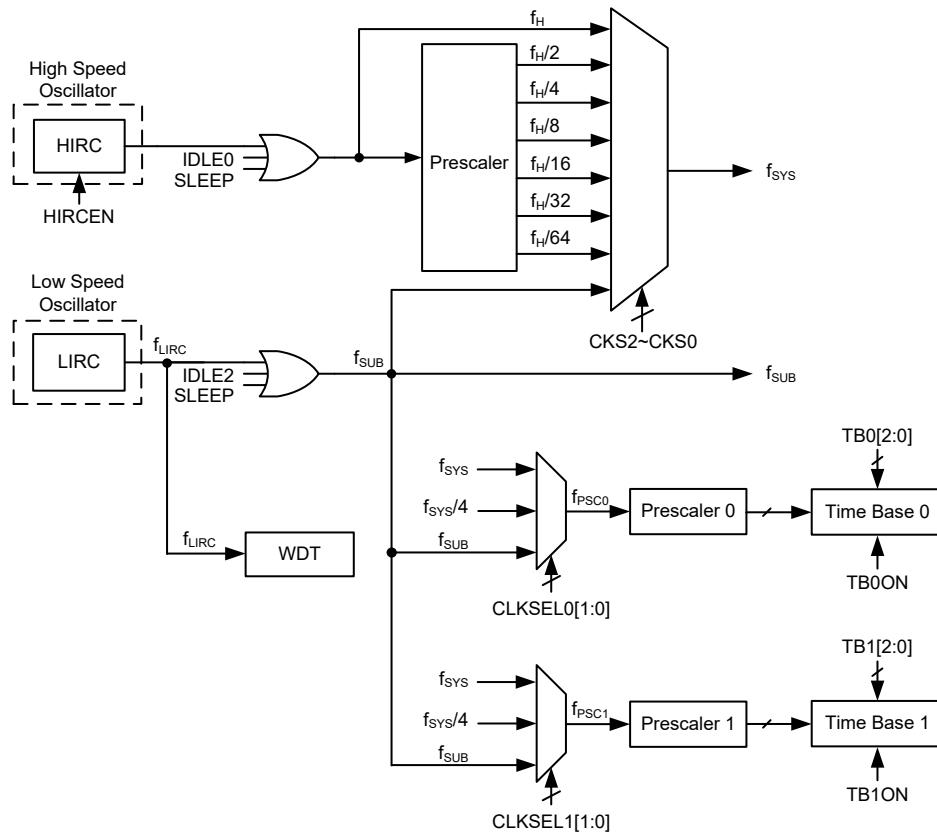
工作模式和系统时钟

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

系统时钟

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

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



单片机时钟配置

注：当系统时钟源 f_{SYS} 由 f_H 切换为 f_{SUB} 时，可以通过设置相应的高速振荡器使能控制位，选择停止以节省耗电，或者继续振荡，为外围电路提供 f_H ~ $f_H/64$ 频率的时钟源。

系统工作模式

单片机有 6 种不同的工作模式，每种有它自身的特性，根据应用中不同的性能和功耗要求可选择不同的工作模式。单片机正常工作有两种模式：快速模式和低速模式。剩余的 4 种工作模式：休眠模式、空闲模式 0、空闲模式 1 和空闲模式 2 用于单片机 CPU 关闭时以节省耗电。

工作模式	CPU	寄存器设置			f_{SYS}	f_H	f_{SUB}	f_{LIRC}
		FHIDEN	FSIDEN	CKS2~CKS0				
快速模式	On	x	x	000~110	$f_H \sim f_{H/64}$	On	On	On
低速模式	On	x	x	111	f_{SUB}	On/Off ⁽¹⁾	On	On
空闲模式 0	Off	0	1	000~110	Off	Off	On	On
				111	On			
空闲模式 1	Off	1	1	xxx	On	On	On	On
空闲模式 2	Off	1	0	000~110	On	On	Off	On
				111	Off			
休眠模式	Off	0	0	xxx	Off	Off	Off	On/off ⁽²⁾

“x”：无关

注：1. 在低速模式中， f_H 开启或关闭由相应的振荡器使能位控制。

2. 在休眠模式中， f_{LIRC} 时钟开启或关闭由 WDT 功能使能或除能控制。

快速模式

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

低速模式

此模式的系统时钟虽为较低速时钟源，但单片机仍能正常工作。该低速时钟源可来自 f_{SUB} ，而 f_{SUB} 来自 LIRC 振荡器。

休眠模式

在 HALT 指令执行后且 FHIDEN 位和 FSIDEN 位为低时，系统进入休眠模式。在休眠模式中，CPU 停止运行， f_{SUB} 停止为外围功能提供时钟。若看门狗定时器功能使能， f_{LIRC} 继续运行。

空闲模式 0

执行 HALT 指令后且 SCC 寄存器中的 FHIDEN 位为低、FSIDEN 位为高时，系统进入空闲模式 0。在空闲模式 0 中，CPU 停止，但低速振荡器会开启以驱动一些外围功能。

空闲模式 1

执行 HALT 指令后且 SCC 寄存器中的 FHIDEN 和 FSIDEN 位都为高时，系统进入空闲模式 1。在空闲模式 1 中，CPU 停止，但高速和低速振荡器都会开启以确保一些外围功能继续工作。

空闲模式 2

执行 HALT 指令后且 SCC 寄存器中的 FHIDEN 位为高、FSIDEN 位为低时，系统进入空闲模式 2。在空闲模式 2 中，CPU 停止，但高速振荡器会开启以确保一些外围功能继续工作。

控制寄存器

寄存器 SCC 和 HIRCC 用于控制系统时钟和 HIRC 振荡器配置。

寄存器 名称	位							
	7	6	5	4	3	2	1	0
SCC	CKS2	CKS1	CKS0	—	—	—	FHIDEN	FSIDEN
HIRCC	—	—	—	—	—	—	HIRCF	HIRCEN

系统工作模式控制寄存器列表

• SCC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	CKS2	CKS1	CKS0	—	—	—	FHIDEN	FSIDEN
R/W	R/W	R/W	R/W	—	—	—	R/W	R/W
POR	0	0	0	—	—	—	0	0

Bit 7~5 **CKS2~CKS0:** 系统时钟选择位

- 000: f_H
- 001: $f_H/2$
- 010: $f_H/4$
- 011: $f_H/8$
- 100: $f_H/16$
- 101: $f_H/32$
- 110: $f_H/64$
- 111: f_{SUB}

这三位用于选择系统时钟源。除了 f_H 或 f_{SUB} 提供的系统时钟源外，也可使用高频振荡器的分频作为系统时钟。

Bit 4~2 未定义，读为“0”

Bit 1 **FHIDEN:** CPU 关闭时高频振荡器控制位

- 0: 除能
- 1: 使能

此位用来控制在 CPU 执行 HALT 指令关闭后高速振荡器是被激活还是停止。

Bit 0 **FSIDEN:** CPU 关闭时低频振荡器控制位

- 0: 除能
- 1: 使能

此位用来控制在 CPU 执行 HALT 指令关闭后低速振荡器是被激活还是停止。

注：使用 CKS2~CKS0 位进行时钟切换设置之后，在相关时钟成功切换至目标时钟源之前需要一定的延时。因此，若接下来执行的操作需要目标时钟源立即响应，则在此之前必须规划适当的延迟时间。

时钟切换延迟时间 = $4 \times t_{SYS} + [0 \sim (1.5 \times t_{curr.} + 0.5 \times t_{tar.})]$ ，其中 $t_{curr.}$ 指代当前的时钟周期， $t_{tar.}$ 指代目标时钟周期， t_{SYS} 指代当前系统时钟周期。

- HIRCC 寄存器

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

Bit 7~2 未定义, 读为“0”

Bit 1 **HIRCF**: HIRC 振荡器稳定标志位

0: HIRC 未稳定

1: HIRC 稳定

此位用于表明 HIRC 振荡器是否稳定。HIRCEN 位置高使能 HIRC 振荡器, HIRCF 位会先被清零, 在 HIRC 稳定后会被置高。

Bit 0 **HIRCEN**: HIRC 振荡器使能控制位

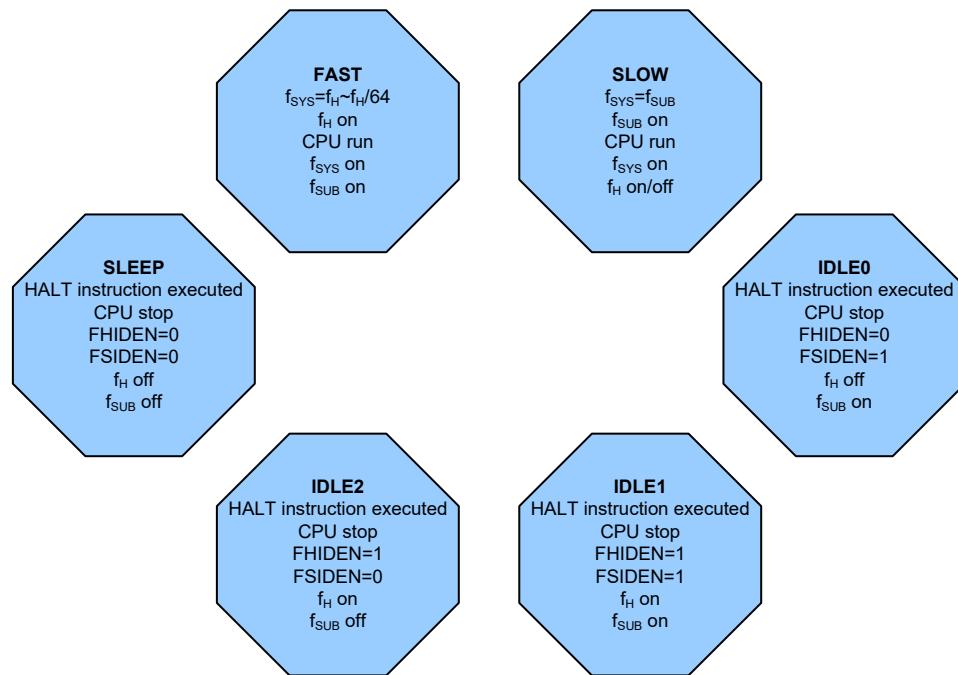
0: 除能

1: 使能

工作模式切换

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

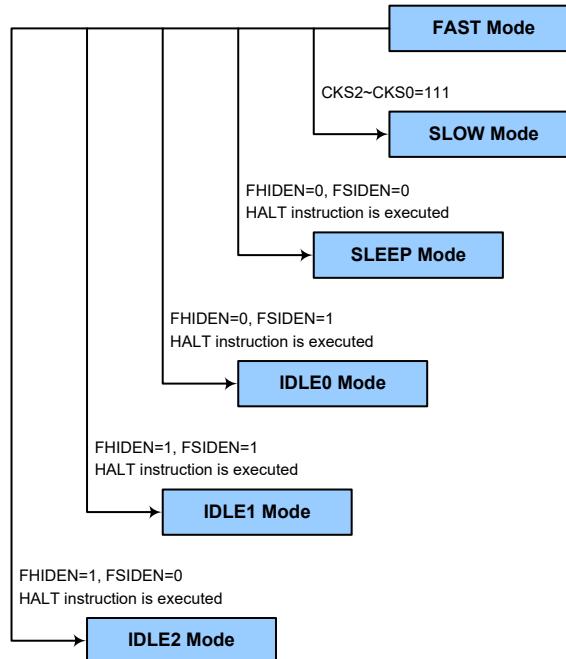
简单来说, 快速模式和低速模式间的切换仅需设置 SCC 寄存器中的 CKS2~CKS0 位即可实现, 而快速模式 / 低速模式与休眠模式 / 空闲模式间的切换经由 HALT 指令实现。当 HALT 指令执行后, 单片机是否进入空闲模式或休眠模式由 SCC 寄存器中的 FHIDEN 和 FSIDEN 位决定的。



快速模式切换到低速模式

系统运行在快速模式时使用高速系统振荡器，因此较为耗电。可通过设置 SCC 寄存器中的 CKS2~CKS0 位为“111”使系统时钟切换至运行在低速模式下。此时将使用低速系统振荡器以节省耗电。用户可在对性能要求不高的操作中使用此方法以减少耗电。

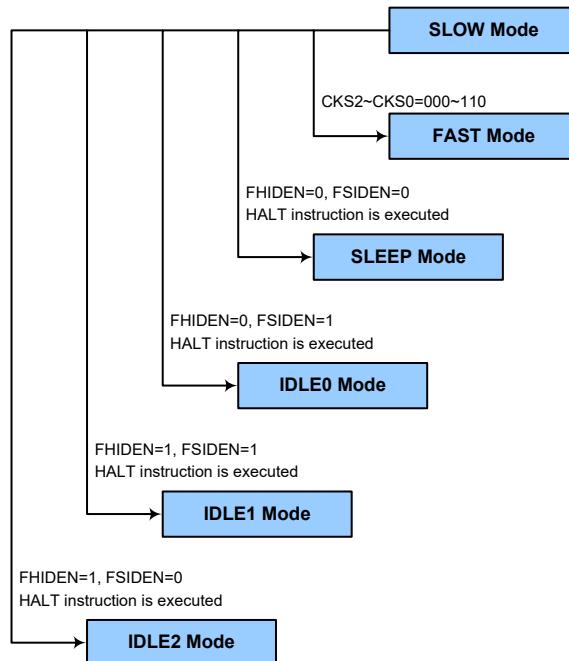
低速模式的时钟源来自 LIRC 振荡器，因此要求此振荡器在所有模式切换动作发生前稳定下来。



低速模式切换到快速模式

在低速模式时系统时钟来自 f_{SUB} 。切换回快速模式时，需设置 CKS2~CKS0 位为“000”~“110”使系统时钟从 f_{SUB} 切换到 $f_H \sim f_H/64$ 。

然而，如果在低速模式下 f_H 因未使用而关闭，那么从低速模式切换到快速模式时，它需要一定的时间来重新起振和稳定，可通过检测 HIRCC 寄存器中的 HIRCF 位进行判断，所需的高速系统振荡器稳定时间在系统上电时间电气特性中有说明。



进入休眠模式

进入休眠模式的方法仅有一种，即应用程序中执行“HALT”指令前需设置 SCC 寄存器中的 FHIDEN 和 FSIDEN 位都为“0”。在这种模式下，除了 WDT 以外的所有时钟和功能都将关闭。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟停止运行，应用程序停止在“HALT”指令处。
- 数据存储器中的内容和寄存器将保持当前值。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。
- 如果 WDT 功能使能，WDT 将被清零并重新开始计数。如果 WDT 功能除能，WDT 将被清零并停止计数。

进入空闲模式 0

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

- f_H 时钟停止运行，应用程序停止在“HALT”指令处，但 f_{SUB} 时钟将继续运行。
- 数据存储器中的内容和寄存器将保持当前值。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。
- 如果 WDT 功能使能，WDT 将被清零并重新开始计数。如果 WDT 功能除能，WDT 将被清零并停止计数。

进入空闲模式 1

进入空闲模式 1 的方法仅有一种，即应用程序中执行“HALT”指令前需设置 SCC 寄存器中的 FHIDEN 和 FSIDEN 位都为“1”。在上述条件下执行该指令后，将发生的情况如下：

- f_H 和 f_{SUB} 时钟开启，应用程序停止在“HALT”指令处。
- 数据存储器中的内容和寄存器将保持当前值。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。
- 如果 WDT 功能使能，WDT 将被清零并重新开始计数。如果 WDT 功能除能，WDT 将被清零并停止计数。

进入空闲模式 2

进入空闲模式 2 的方法仅有一种，即应用程序中执行“HALT”指令前需设置 SCC 寄存器中的 FHIDEN 位为“1”且 FSIDEN 位为“0”。在上述条件下执行该指令后，将发生的情况如下：

- f_H 时钟开启， f_{SUB} 时钟关闭，应用程序停止在“HALT”指令处。
- 数据存储器中的内容和寄存器将保持当前值。
- 输入 / 输出口将保持当前值。
- 状态寄存器中暂停标志 PDF 将被置起，看门狗溢出标志 TO 将被清除。
- 如果 WDT 功能使能，WDT 将被清零并重新开始计数。如果 WDT 功能除能，WDT 将被清零并停止计数。

待机电流注意事项

由于单片机进入休眠或空闲模式的主要原因是将单片机的电流降低到尽可能低，可能到只有几个微安的级别（空闲模式 1 和空闲模式 2 除外），所以如果要将电路的电流进一步降低，电路设计者还应有其它的考虑。应该特别注意的是单片机的输入 / 输出引脚。所有高阻抗输入脚都必须连接到固定的高或低电平，因为引脚浮空会造成内部振荡并导致耗电增加。这也应用于有不同封装的单片机，因为它们可能含有未引出的引脚，这些引脚也必须设为输出或带有上拉电阻的输入。

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

在空闲模式 1 和空闲模式 2 中高速振荡器开启。若外围功能时钟源来自高速振荡器，额外的待机电流也可能会有几百微安。

唤醒

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

系统进入休眠或空闲模式之后，可以通过以下几种方式唤醒：

- PA 口下降沿
- 系统中断
- WDT 溢出

单片机执行 HALT 指令，系统将进入空闲或休眠模式，PDF 将被置位；系统上电或执行清除看门狗的指令，PDF 将被清零。

若系统由看门狗定时器溢出唤醒，则会发生看门狗定时器复位，TO 将被置位。看门狗计数器溢出将会置位 TO 标志并唤醒系统，这种复位会重置程序计数器和堆栈指针，其它标志保持原有状态。

PA 口中的每个引脚都可以通过 PAWU 寄存器使能下降沿唤醒功能。PA 端口唤醒后，程序将在“HALT”指令后继续执行。

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

看门狗定时器

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

看门狗定时器时钟源

WDT 定时器时钟源由内部低速振荡器 LIRC 提供。内部振荡器 LIRC 的频率大约为 32kHz，这个特殊的内部时钟周期会随 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 $2^8 \sim 2^{15}$ 以提供更大的溢出周期，分频比由 WDTC 寄存器中的 WS2~WS0 位来决定。

看门狗定时器控制寄存器

WDTC 寄存器用于选择溢出周期、控制 WDT 功能的使能 / 除能和 MCU 复位操作。

● WDTC 寄存器

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

Bit 7~3 **WE4~WE0:** WDT 功能软件控制位

01010: 使能

10101: 除能

其它: MCU 复位

若因外部环境噪声或软件设置使单片机复位，复位动作发生在一段延迟时间 t_{RESET} 后，复位后 RSTFC 寄存器中的 WRF 标志位会被置位。

Bit 2~0 **WS2~WS0:** WDT 溢出周期选择位

000: $2^8/f_{LIRC}$

001: $2^9/f_{LIRC}$

010: $2^{10}/f_{LIRC}$

011: $2^{11}/f_{LIRC}$

100: $2^{12}/f_{LIRC}$

101: $2^{13}/f_{LIRC}$

110: $2^{14}/f_{LIRC}$

111: $2^{15}/f_{LIRC}$

这三位控制 WDT 时钟源的分频比，从而实现对 WDT 溢出周期的控制。

- RSTFC 寄存器

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

“x”：未知

Bit 7~3 未定义，读为“0”

Bit 2 **LVRF**: LVR 复位标志位

具体描述见低电压复位章节。

Bit 1 **LRF**: LVR 控制寄存器软件复位标志位

具体描述见低电压复位章节。

Bit 0 **WRF**: WDT 寄存器软件复位标志位

0: 未发生

1: 发生

当 WDT 控制寄存器软件复位发生时，此位被置为“1”，且只能通过应用程序清零。

看门狗定时器操作

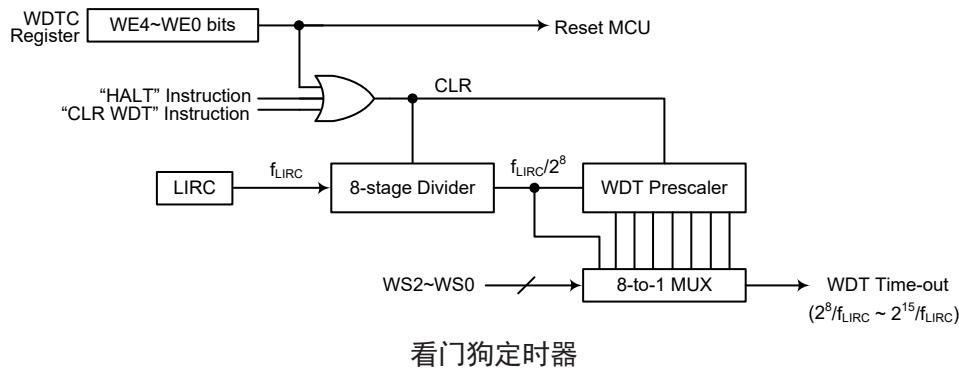
当 WDT 溢出时，它产生一个单片机复位的动作。这也就意味着正常工作期间，用户需在应用程序中看门狗溢出前有策略地清看门狗定时器以防止其产生复位，可使用清除看门狗指令实现。无论什么原因，程序失常跳转到一个未知的地址或进入一个死循环，这个清除指令不能被正确执行，此种情况下，看门狗将溢出以使单片机复位。看门狗定时器控制寄存器 WDTC 中的 WE4~WE0 位可提供使能 / 除能控制以及控制看门狗定时器复位单片机。当 WE4~WE0 设置为“10101B”时除能 WDT 功能，而当设置为 01010B 时使能 WDT。如果设置为“01010B”和“10101B”以外的值时，单片机将在一段延迟时间 t_{RESET} 后复位。上电后这些位初始化为“01010B”。

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

看门狗定时器使能 / 除能 / 复位控制

程序正常运行时，WDT 溢出将导致单片机复位，并置位状态标志位 TO。若系统处于休眠或空闲模式，当 WDT 发生溢出时，状态寄存器中的 TO 应置位，仅 PC 和堆栈指针复位。有三种方法可以用来清除 WDT 的内容。第一种是 WDTC 软件复位，即将 WE4~WE0 位设置成除了“01010B”和“10101B”外的任意值；第二种是通过 WDT 软件清除指令，而第三种是通过“HALT”指令。

该单片机只使用一条软件指令清看门狗。只要执行“CLR WDT”便清除 WDT。当设置分频比为 2^{15} 时，溢出周期最大。当时钟源为 32kHz LIRC 振荡器，分频比为 2^{15} 时最大溢出周期约 1s，分频比为 2^8 时最小溢出周期约 8ms。



复位和初始化

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

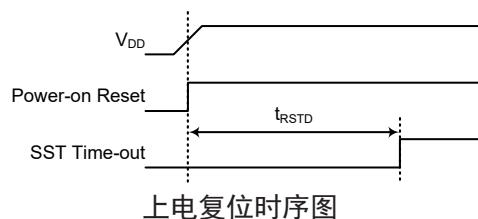
另一种复位为低电压复位即 LVR 复位，在电源供应电压低于 LVR 设定值时，系统会产生 LVR 复位。还有一种复位为看门狗溢出单片机复位，不同方式的复位操作会对寄存器产生不同的影响。

复位功能

单片机包含下面几种由内部事件触发的复位方式。

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。

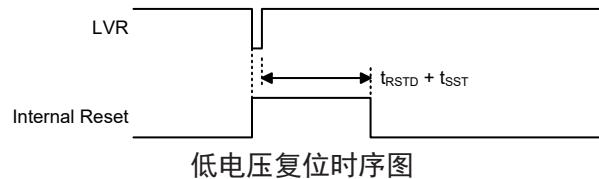


低电压复位 – LVR

单片机具有低电压复位电路，用来监测它的电源电压。当电源电压低于某一预定值时，它将复位单片机。如果在更换电池的情况下，单片机供应的电压可能会在 $0.9V \sim V_{LVR}$ 之间，这时 LVR 将会自动复位单片机且 RSTFC 寄存器中的 LVRF 标志位置位。所谓有效的 LVR 信号，即在 $0.9V \sim V_{LVR}$ 的低电压状态的时间，必须超过 LVR 电气特性中 t_{LVR} 参数的值。如果低电压存在不超过 t_{LVR} 参数的值，则 LVR 将会忽略它且不会执行复位功能。若 LVS7~LVS0 位设置为 01011010B，则 LVR 功能使能，且 V_{LVR} 值固定为 1.7V。若 LVS7~LVS0 位设置为 10100101B，则 LVR 功能除能。若由于受到干扰 LVS7~LVS0 变为其它值时，

将在 t_{SRESET} 时间后复位单片机。此时 RSTFC 寄存器的 LRF 位被置位。上电后寄存器的值为 01011010B。

需要注意的是，当单片机进入空闲或休眠模式，LVR 功能将自动除能。



低电压复位时序图

• LVRC 寄存器

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

Bit 7~0 **LVS7~LVS0:** LVR 电压选择

01011010: 1.7V

10100101: LVR 除能

其它值: 单片机复位 – 寄存器复位为 POR 值

当上述定义的相应的低电压出现，且低电压保持时间超过 t_{LVR} 值，则单片机复位。此时复位后的寄存器内容保持不变。

若将 LVRC 寄存器定义为 01011010B 和 10100101B 以外的其它值，将会产生单片机复位。复位操作会在 t_{SRESET} 时间后执行。注意的是此处单片机复位后，寄存器的值将恢复到上电复位值。

• VBGC 寄存器

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

Bit 7~4 未定义，读为“0”

Bit 3 **VBGEN:** Bandgap 控制

0: 除能

1: 使能

当 LVR 功能使能或当 VBGEN 位设置为“1”时，Bandgap 电路使能。

Bit 2~0 未定义，读为“0”

• RSTFC 寄存器

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

“x”：未知

Bit 7~3 未定义，读为“0”

Bit 2 **LVRF:** LVR 功能复位标志位

0: 未发生

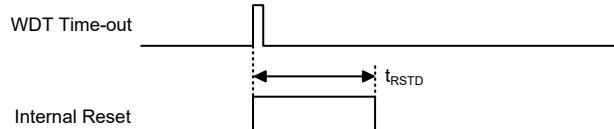
1: 发生

当特定的低电压复位条件发生时，此位被置为“1”，且只能通过应用程序清零。

- Bit 1 **LRF:** LVR 控制寄存器软件复位标志位
 0: 未发生
 1: 发生
 如果 LVRC 寄存器包含任何非定义的 LVR 电压值, 此位被置为“1”, 这类似于软件复位功能, 且只能通过应用程序清零。
- Bit 0 **WRF:** WDT 控制寄存器软件复位标志位
 具体描述见看门狗定时器控制寄存器章节。

正常运行时看门狗溢出复位

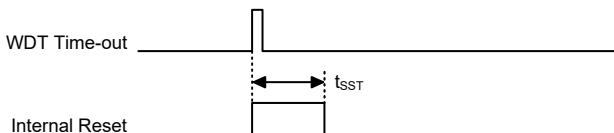
在正常运行即快速模式和低速模式下看门狗溢出复位后 TO 将被设为“1”。



正常运行时看门狗溢出复位时序图

休眠或空闲时看门狗溢出复位

休眠或空闲时看门狗溢出复位和其它种类的复位有些不同, 除了程序计数器与堆栈指针将被清零及 TO 位被设为 1 外, 绝大部份的条件保持不变。图中 t_{sst} 的详细说明请参考系统上电时间电气特性。



休眠或空闲时看门狗溢出复位时序图

复位初始状态

不同的复位形式以不同的途径影响复位标志位。这些标志位, 即存放在状态寄存器中的 PDF 和 TO 位, 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	快速模式或低速模式时的 LVR 复位
1	u	快速模式或低速模式时的 WDT 溢出复位
1	1	空闲或休眠模式时的 WDT 溢出复位

“u”代表不改变

在单片机上电复位之后, 各功能单元初始化的情形, 列于下表。

项目	复位后情况
程序计数器	清除为零
中断	所有中断被除能
看门狗定时器, 时基	都清零, WDT 清除并重新计数
定时器模块	定时器模块停止
输入 / 输出口	I/O 口设为输入模式
堆栈指针	堆栈指针指向堆栈顶端

不同的复位形式对单片机内部寄存器的影响是不同的。为保证复位后程序能正常执行，了解寄存器在特定条件复位后的设置是非常重要的。下表即为不同方式复位后内部寄存器的状况。注意的是，此单片机支持多种封装类型，该表将反映较大封装类型的情况。

寄存器	上电复位	WDT 溢出 (正常运行)	WDT 溢出 (空闲 / 休眠)
IAR0	xxxx xxxx	uuuu uuuu	uuuu uuuu
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu
IAR1	xxxx xxxx	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu
BP	---- - -0	---- - -0	---- - -u
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu
TBLH	-xxx xxxx	-uuu uuuu	-uuu uuuu
TBHP	---- -xx x	---- -uu u	---- -uu u
STATUS	--00 xxxx	--1u uuuu	--11 uuuu
SCC	000- --00	000- --00	uuu- --uu
HIRCC	---- --01	---- --01	---- --uu
LVRC	0101 1010	0101 1010	uuuu uuuu
INTEG	---- --00	---- --00	---- --uu
RSTFC	---- -x00	---- -uu u	---- -uu u
INTC0	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	uuuu uuuu
INTC2	---0 ---0	---0 ---0	---u ---u
PA	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	uuuu uuuu
PAPU	0000 0000	0000 0000	uuuu uuuu
PAWU	0000 0000	0000 0000	uuuu uuuu
PB	---1 1111	---1 1111	---u uuuu
PBC	---1 1111	---1 1111	---u uuuu
PBPU	---0 0000	---0 0000	---u uuuu
SLEDC	0000 0000	0000 0000	uuuu uuuu
VBGC	---- 0---	---- 0---	---- u---
WDTC	0101 0011	0101 0011	uuuu uuuu
PSC0R	---- --00	---- --00	---- --uu
TB0C	0--- -000	0--- -000	u--- -uuu
PSC1R	---- --00	---- --00	---- --uu
TB1C	0--- -000	0--- -000	u--- -uuu
EEA	---0 0000	---0 0000	---u uuuu
EED	0000 0000	0000 0000	uuuu uuuu
PAS0	0000 0000	0000 0000	uuuu uuuu
PAS1	0000 --00	0000 --00	uuuu --uu
PBS0	0000 0000	0000 0000	uuuu uuuu

寄存器	上电复位	WDT 溢出 (正常运行)	WDT 溢出 (空闲 / 休眠)
PBS1	-----00	-----00	-----uu
TKTMR	0000 0000	0000 0000	uuuu uuuu
TKC0	-000 0-00	-000 0-00	-uuu u-uu
TKC1	-----11	-----11	-----uu
TK16DL	0000 0000	0000 0000	uuuu uuuu
TK16DH	0000 0000	0000 0000	uuuu uuuu
TKM0C0	0000 0000	0000 0000	uuuu uuuu
TKM0C1	0-00 --00	0-00 --00	u-uu --uu
TKM016DL	0000 0000	0000 0000	uuuu uuuu
TKM016DH	0000 0000	0000 0000	uuuu uuuu
TKM0ROL	0000 0000	0000 0000	uuuu uuuu
TKM0ROH	-----00	-----00	-----uu
CTMC0	0000 0000	0000 0000	uuuu uuuu
CTMC1	0000 0000	0000 0000	uuuu uuuu
CTMDL	0000 0000	0000 0000	uuuu uuuu
CTMDH	-----00	-----00	-----uu
CTMAL	0000 0000	0000 0000	uuuu uuuu
CTMAH	-----00	-----00	-----uu
ISGENC	0---000	0---000	u---uu
ISGDATA0	---0 0000	---0 0000	---u uuuu
ISGDATA1	---0 0000	---0 0000	---u uuuu
LVPUC	-----0	-----0	-----uu
OPSWA	0000 0000	0000 0000	uuuu uuuu
OPSWB	0000 0000	0000 0000	uuuu uuuu
OPSWC	----0000	----0000	----uuuu
OPDC	----000	----000	----uuuu
OPDACL	0010 0000	0010 0000	uuuu uuuu
SADC0	0000 0000	0000 0000	uuuu uuuu
SADC1	0000 0000	0000 0000	uuuu uuuu
SADOH	xxxx xxxx	xxxx xxxx	uuuu uuuu (ADRFS=0)
			-----uu (ADRFS=1)
SADOL	xx-- -----	xx-- -----	uu-- ----- (ADRFS=0)
			uuuu uuuu (ADRFS=1)
ORMC	0000 0000	0000 0000	0000 0000
EEC	0---0000	0---0000	u---uuuu

注：“u”表示不改变

“x”表示未知

“-”表示未定义

输入 / 输出端口

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

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

寄存器 名称	位							
	7	6	5	4	3	2	1	0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PB	—	—	—	PB4	PB3	PB2	PB1	PB0
PBC	—	—	—	PBC4	PBC3	PBC2	PBC1	PBC0
PBPU	—	—	—	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
LVPUC	—	—	—	—	—	—	—	LVPU

“—”：未定义，读为“0”

输入 / 输出逻辑功能寄存器列表

上拉电阻

许多产品应用在端口处于输入状态时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，当引脚规划为数字输入时，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 PxPU 和 LVPUC 来设置，它用一个弱 PMOS 晶体管来实现上拉电阻功能。PxPU 寄存器用于决定是否使能上拉功能，LVPUC 寄存器用于选择低电压应用时的上拉电阻值。

需要注意的是当 I/O 引脚设为数字输入或 NMOS 输出时，上拉功能才会受 PxPU 控制开启，其它状态下上拉功能不可用。

• PxPU 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PxPU7	PxPU6	PxPU5	PxPU4	PxPU3	PxPU2	PxPU1	PxPU0
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

PxPUn: I/O Port x 引脚上拉功能控制位

0: 除能

1: 使能

PxPUn 位用于控制引脚上拉功能。此处的“x”可为 A 或 B。每个端口实际的有效位是不同的，具体信息可参考 I/O 逻辑功能寄存器列表。

● LVPUC 寄存器

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

Bit 7~1 未定义, 读为“0”

Bit 0 **LVPU**: 低电压时上拉电阻选择位

0: 所有引脚上拉电阻为 $60\text{k}\Omega$ @ 3V

1: 所有引脚上拉电阻为 $15\text{k}\Omega$ @ 3V

LVPU 位用于选择低电压应用时的上拉电阻值。此位只有在对应引脚的上拉功能使能时才有效。若对应上拉功能除能, 使用 LVPU 位选择上拉电阻值无效。

PA 口唤醒

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

需要注意的是只有当引脚被设置为通用 I/O 功能输入类型且单片机处于空闲或休眠模式时, 唤醒功能才会受 PAWU 控制开启, 其它状态下此唤醒功能不可用。

● PAWU 寄存器

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

Bit 7~0 **PAWU7~PAWU0**: PA7~PA0 唤醒功能控制位

0: 除能

1: 使能

输入 / 输出端口控制寄存器

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

- **PxC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PxC7	PxC6	PxC5	PxC4	PxC3	PxC2	PxC1	PxC0
R/W								
POR	1	1	1	1	1	1	1	1

PxCn: I/O Port x 引脚输入 / 输出类型选择位

0: 输出

1: 输入

PxCn 用于选择引脚输入 / 输出类型。此处的 “x” 可为 A 或 B。每个端口实际的有效位是不同的，具体信息可参考 I/O 逻辑功能寄存器列表。

输入 / 输出端口源电流控制

通过配置相应的源电流选择位，该单片机的每个 I/O 口可支持不同的源电流驱动能力。用户可参考输入 / 输出口电气特性章节为不同应用选择所需的源电流。

- **SLEDC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	SLEDC7	SLEDC6	SLEDC5	SLEDC4	SLEDC3	SLEDC2	SLEDC1	SLEDC0
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 **SLEDC7~SLEDC6:** PA7~PA6 引脚源电流选择位

00: 源电流 = Level 0 (最小)

01: 源电流 = Level 1

10: 源电流 = Level 2

11: 源电流 = Level 3 (最大)

Bit 5~4 **SLEDC5~SLEDC4:** PA5~PA4 引脚源电流选择位

00: 源电流 = Level 0 (最小)

01: 源电流 = Level 1

10: 源电流 = Level 2

11: 源电流 = Level 3 (最大)

Bit 3~2 **SLEDC3~SLEDC2:** PA3~PA2 引脚源电流选择位

00: 源电流 = Level 0 (最小)

01: 源电流 = Level 1

10: 源电流 = Level 2

11: 源电流 = Level 3 (最大)

Bit 1~0 **SLEDC1~SLEDC0:** PA1~PA0 引脚源电流选择位

00: 源电流 = Level 0 (最小)

01: 源电流 = Level 1

10: 源电流 = Level 2

11: 源电流 = Level 3 (最大)

引脚共用功能

引脚的多功能可以增加单片机应用的灵活性。有限的引脚个数将会限制设计者，而引脚的多功能将会解决很多此类问题。此外，这些引脚功能可以通过一系列寄存器进行设定。

引脚共用功能选择寄存器

封装中有限的引脚个数会对某些单片机功能造成影响。然而，引脚功能共用和引脚功能选择，使得小封装单片机具有更多不同的功能。该单片机包含端口引脚输出功能选择寄存器 PxSn，这些寄存器可以用来对引脚上的功能进行配置。

要注意的最重要一点是，确保所需的引脚共用功能被正确地选择和取消。对于大部分共用功能，要选择所需的引脚共用功能，首先应通过相应的引脚共用控制寄存器正确地选择该功能，然后再配置相应的外围功能设置以使能外围功能。但是，在设置相关引脚控制字段时，一些数字输入引脚如 INT、CTCK 等，与对应的通用 I/O 口共用同一个引脚共用设置选项。要选择这个引脚功能，除了上述的必要的引脚共用控制和外围功能设置外，还必须将其对应的端口控制寄存器位设置为输入。要正确地取消引脚共用功能，首先应除能外围功能，然后再修改相应的引脚共用控制寄存器以选择其它的共用功能。

寄存器 名称	位							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	PAS17	PAS16	PAS15	PAS14	—	—	PAS11	PAS10
PBS0	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
PBS1	—	—	—	—	—	—	PBS11	PBS10

引脚共用功能选择寄存器列表

• PAS0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PAS07~PAS06:** PA3 引脚共用功能选择

- 00: PA3
- 01: OPDAP
- 10: PA3
- 11: PA3

Bit 5~4 **PAS05~PAS04:** PA2 引脚共用功能选择

- 00: PA2
- 01: OPDAN
- 10: VBAT
- 11: PA2

注：由于 VBAT 引脚有 pin map 设计，建议优先使用 PA4 引脚，并避免同时使用 PA2/PA4 引脚。

Bit 3~2 **PAS03~PAS02:** PA1 引脚共用功能选择

- 00: PA1
- 01: OPDAO
- 10: AN1
- 11: CTP

Bit 1~0 **PAS01~PAS00:** PA0 引脚共用功能选择

- 00: PA0/INT
- 01: AN0
- 10: CTP
- 11: PA0/INT

- **PAS1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PAS17	PAS16	PAS15	PAS14	—	—	PAS11	PAS10
R/W	R/W	R/W	R/W	R/W	—	—	R/W	R/W
POR	0	0	0	0	—	—	0	0

Bit 7~6 **PAS17~PAS16:** PA7 引脚共用功能选择

00: PA7
01: AN3
10: PA7
11: PA7

Bit 5~4 **PAS15~PAS14:** PA6 引脚共用功能选择

00: PA6
01: OPDAP
10: AN2
11: CTPB

Bit 3~2 未定义, 读为“0”

Bit 1~0 **PAS11~PAS10:** PA4 引脚共用功能选择

00: PA4
01: VBAT
10: PA4
11: PA4

注: 由于 VBAT 引脚有 pin map 设计, 建议优先使用 PA4 引脚, 并避免同时使用 PA2/PA4 引脚。

- **PBS0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PBS07	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PBS07~PBS06:** PB3 引脚共用功能选择

00: PB3
01: KEY2
10: PB3
11: PB3

Bit 5~4 **PBS05~PBS04:** PB2 引脚共用功能选择

00: PB2
01: KEY1
10: VREF
11: PB2

Bit 3~2 **PBS03~PBS02:** PB1 引脚共用功能选择

00: PB1
01: CTPB
10: PB1
11: PB1

Bit 1~0 **PBS01~PBS00:** PB0 引脚共用功能选择

00: PB0
01: CTP
10: PB0
11: PB0

● PBS1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PBS11	PBS10
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义, 读为“0”

Bit 1~0 PBS11~PBS10: PB4 引脚共用功能选择

00: PB4

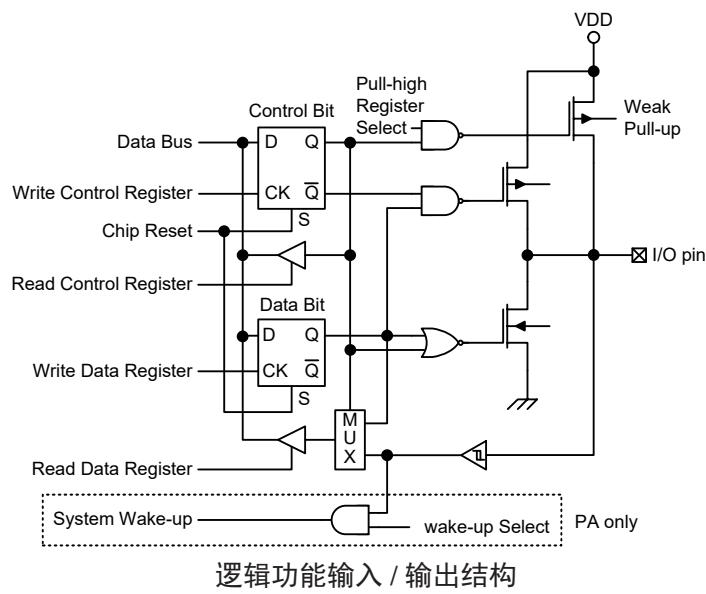
01: CTP

10: PB4

11: PB4

输入 / 输出引脚结构

下图为输入 / 输出引脚逻辑功能的内部结构图。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 引脚逻辑功能的理解提供一个参考。由于存在诸多的引脚共用结构, 在此不方便提供所有类型引脚功能结构图。



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

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

定时器模块 – TM

控制和测量时间在任何单片机中都是一个很重要的部分。该单片机提供定时器模块(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，比较匹配输出以及 PWM 输出等功能。定时器模块有两个独立中段，其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。

简介

该单片机包含一个简易型 TM，即 CTM。CTM 的主要特性见下表。

功能	CTM
定时 / 计数器	√
比较匹配输出	√
PWM 输出	√
PWM 对齐方式	边沿对齐
PWM 调节周期 & 占空比	占空比或周期

TM 功能概要

TM 操作

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 CTM 控制寄存器的 CTCK2~CTCK0 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 的分频比或 f_{SUB} 时钟源或外部 CTCK 引脚。CTCK 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 TM 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

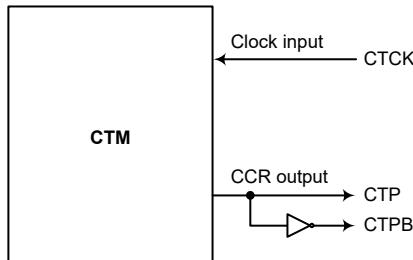
该 TM 有一个输入引脚，即 CTCK。CTM 输入引脚 CTCK 作为 CTM 时钟源输入脚，通过设置 CTMC0 寄存器中的 CTCK2~CTCK0 位进行选择。外部时钟源可通过该引脚来驱动内部 TM。该 TM 输入引脚可选择上升沿有效或下降沿有效。

该 TM 有两个输出引脚 CTP 和 CTPB。CTPB 信号为 CTP 输出的反相信号。当 TM 工作在比较匹配输出模式且比较匹配发生时，这些引脚会由 TM 控制切换到高电平或低电平或翻转。外部输出引脚也被 TM 用来产生 PWM 输出波形。

因 TM 输入和输出引脚与其它功能共用，TM 输入和输出功能需要事先通过相关引脚共用功能选择位进行设置。更多引脚共用功能选择详见引脚共用功能章节。

CTM	
输入	输出
CTCK	CTP, CTPB

TM 外部引脚

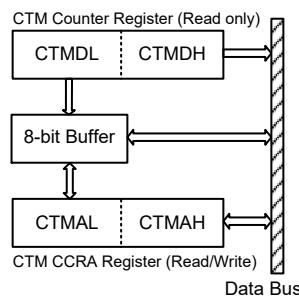


CTM 功能引脚方框图

编程注意事项

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

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

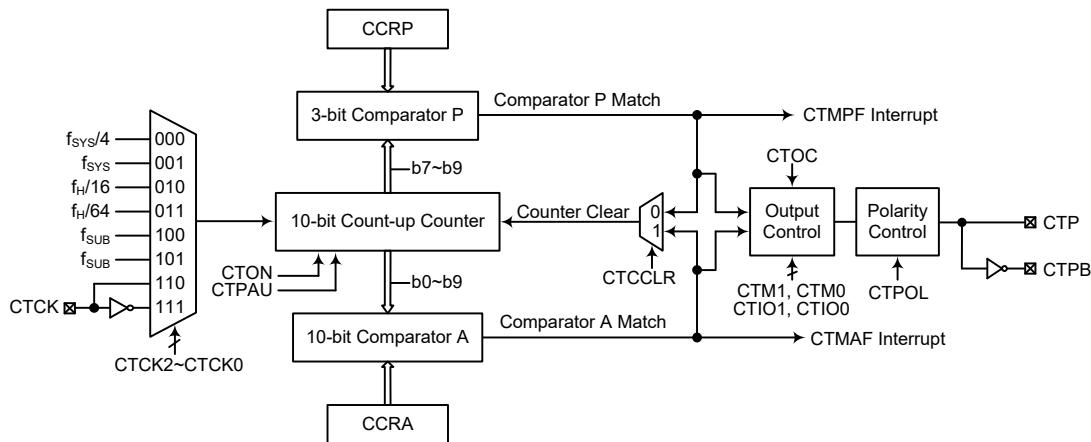


读写流程如下步骤所示：

- 写数据至 CCRA
 - ◆ 步骤 1. 写数据至低字节寄存器 CTMAL
 - 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 CTMAH
 - 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、CCRA 中读取数据
 - ◆ 步骤 1. 由高字节寄存器 CTMDH、CTMAH 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 CTMDL、CTMAL 读取数据
 - 注意，此时读取 8-bit 缓存器中的数据。

简易型 TM – CTM

简易型 TM 包括三种工作模式，即比较匹配输出、定时 / 事件计数器和 PWM 输出模式。简易型 TM 由一个外部输入脚控制并驱动两个外部输出脚。



注：1. CTM 外部引脚与其它功能共用引脚，因此，在使用这些引脚功能之前，相关引脚共用功能寄存器必须合理设置；CTCK 引脚除外，只有通过 CTM 相关控制寄存器选择 CTM 时钟源来自 CTCK 引脚时，此引脚功能才选中。
2. CTPB 为 CTP 的反相信号。

10-bit 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 10 位向上计数器，它还包括两个内部比较器即比较器 A 和比较器 P。这两个比较器将计数器的值与 CCRP 和 CCRA 寄存器中的值进行比较。CCRP 是 3 位的，与计数器的高 3 位比较；而 CCRA 是 10 位的，与计数器的所有位比较。

通过应用程序改变 10 位计数器值的唯一方法是使 CTON 位发生上升沿跳变清零计数器。此外，计数器溢出或比较匹配也会自动清零计数器。上述条件发生时，通常情况会产生 CTM 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，也可以控制两个输出脚。所有工作模式的设定都是通过设置相关寄存器来实现的。

简易型 TM 寄存器介绍

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

寄存器 名称	位							
	7	6	5	4	3	2	1	0
CTMC0	CTPAU	CTCCK2	CTCCK1	CTCCK0	CTON	CTRP2	CTRP1	CTRP0
CTMC1	CTM1	CTM0	CTIO1	CTIO0	CTOC	CTPOL	CTDPX	CTCCLR
CTMDL	D7	D6	D5	D4	D3	D2	D1	D0
CTMDH	—	—	—	—	—	—	D9	D8
CTMAL	D7	D6	D5	D4	D3	D2	D1	D0
CTMAH	—	—	—	—	—	—	D9	D8

10-bit 简易型 TM 寄存器列表

● CTMC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	CTPAU	CTCK2	CTCK1	CTCK0	CTON	CTRP2	CTRP1	CTRP0
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 **CTPAU:** CTM 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，CTM 保持上电状态并继续耗电。当此位由低到高转换时，计数器将保持其当前计数值，直到此位再次改变为低电平，从此值开始继续计数。

Bit 6~4 **CTCK2~CTCK0:** CTM 计数时钟选择位

000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: f_{SUB}
110: CTCK 上升沿时钟
111: CTCK 下降沿时钟

此三位用于选择 CTM 的时钟源。外部引脚时钟源能被选择在上升沿或下降沿有效。 f_{SYS} 是系统时钟， f_H 和 f_{SUB} 是其它的内部时钟源，细节方面请参考工作模式和系统时钟章节。

Bit 3 **CTON:** CTM 计数器 On/Off 控制位

0: Off
1: On

此位控制 CTM 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 CTM。清零此位将停止计数器并关闭 CTM 减少耗电。当此位经由低到高转换时，内部计数器将复位清零；当此位经由高到低转换时，内部计数器将保持其当前计数值，直到此位再次改变为高电平。

若 CTM 处于比较匹配输出模式或 PWM 输出模式，当 CTON 位经由低到高转换时，CTM 输出脚将复位至 CTOC 位指定的初始值。

Bit 2~0 **CTRP2~CTRP0:** CTM CCRP 3-bit 寄存器，与 CTM 计数器 Bit 9 ~ Bit 7 进行比较
比较器 P 匹配周期 =

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

此三位设定内部 CCRP 3-bit 寄存器的值，然后与内部计数器的高三位进行比较。如果 CTCCLR 位设定为 0 时，此比较结果可用于清零内部计数器。CTCCLR 位设为低，内部计数器在比较器 P 比较匹配发生时被重置；由于 CCRP 只与计数器高三位比较，比较结果是 128 时钟周期的倍数。CCRP 被清零时，实际上会使得计数器在最大值溢出。

- **CTMC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	CTM1	CTM0	CTIO1	CTIO0	CTOC	CTPOL	CTDPX	CTCCLR
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 **CTM1~CTM0:** CTM 工作模式选择位

00: 比较匹配输出模式

01: 未定义

10: PWM 输出模式

11: 定时 / 计数器模式

这两位设置 CTM 需要的工作模式。为了确保操作可靠, CTM 应在 CTM1 和 CTM0 位有任何改变前先关掉。在定时 / 计数器模式, CTM 输出引脚状态未定义。

Bit 5~4 **CTIO1~CTIO0:** CTM 外部引脚功能选择位

比较匹配输出模式

00: 无变化

01: 输出低

10: 输出高

11: 输出翻转

PWM 输出模式

00: PWM 输出无效状态

01: PWM 输出有效状态

10: PWM 输出

11: 未定义

定时 / 计数器模式

未使用

这两位用于决定在达到某些条件时 CTM 输出脚如何改变状态。这两位选择的功能取决于 CTM 运行在哪种模式下。

在比较匹配输出模式下, CTIO1 和 CTIO0 位决定当比较器 A 比较匹配输出发生时 CTM 输出脚如何改变状态。当比较器 A 比较匹配输出发生时 CTM 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。CTM 输出脚的初始值通过 CTMC1 寄存器的 CTOC 位设置。注意, 由 CTIO1 和 CTIO0 位设置的输出电平必须与 CTOC 位设置的初始值不同, 否则当比较匹配发生时, CTM 输出脚将不会发生变化。在 CTM 输出脚改变状态后, 通过 CTON 位由低到高电平的转换可复位至初始值。

在 PWM 输出模式, CTIO1 和 CTIO0 用于决定比较匹配条件发生时如何改变 CTM 输出脚的状态。PWM 输出功能通过这两位的变化进行更新。只可在 CTM 关闭时改变 CTIO1 和 CTIO0 位的值。若在 CTM 运行时改变 CTIO1 和 CTIO0 的值, PWM 输出的值是无法预料的。

Bit 3 **CTOC:** CTM CTP 输出控制位

比较匹配输出模式

0: 初始低

1: 初始高

PWM 输出模式

0: 低有效

1: 高有效

此位为 CTM 输出脚输出控制位。它取决于 CTM 此时正运行于比较匹配输出模式还是 PWM 输出模式。若 CTM 处于定时 / 计数器模式, 此位不起作用。在比较匹配输出模式时, 它决定比较匹配发生前 CTM 输出脚的逻辑电平值。在 PWM 输出模式时, 它决定 PWM 信号是高有效还是低有效。

Bit 2	CTPOL: CTM CTP 输出极性控制位 0: 同相 1: 反相 此位控制 CTP 输出脚的极性。此位为高时 CTM 输出脚反相, 为低时 CTM 输出脚同相。若 CTM 处于定时 / 计数器模式时此位不起作用。							
Bit 1	CTDPX: CTM 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	CTCCLR: CTM 计数器清零条件选择位 0: CTM 比较器 P 匹配 1: CTM 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。CTCCLR 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。CTCCLR 位在 PWM 输出模式时未使用。							

● **CTMDL 寄存器**

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

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

● **CTMDH 寄存器**

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

Bit 7~2 未定义, 读为 “0”
Bit 1~0 **D9~D8:** CTM 计数器高字节寄存器 bit 1 ~ bit 0
CTM 10-bit 计数器 bit 9 ~ bit 8

● **CTMAL 寄存器**

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

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

- CTMAH 寄存器

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

Bit 7~2 未定义, 读为“0”

Bit 1~0 **D9~D8:** CTM CCRA 高字节寄存器 bit 1 ~ bit 0
CTM 10-bit CCRA bit 9 ~ bit 8

简易型 TM 工作模式

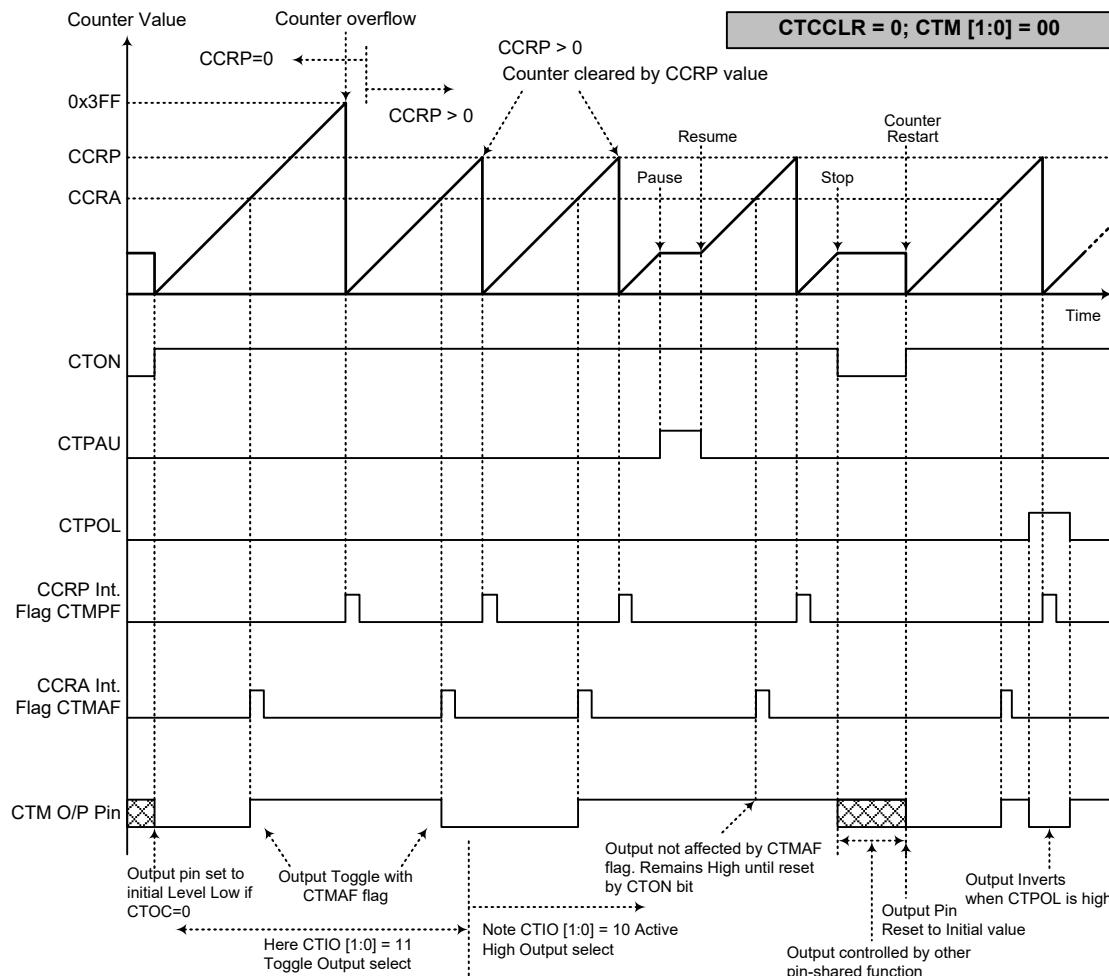
简易型 TM 有三种工作模式, 即比较匹配输出模式, PWM 输出模式或定时 / 计数器模式。通过设置 CTMC1 寄存器的 CTM1 和 CTM0 位选择任意工作模式。

比较匹配输出模式

为使 CTM 工作在此模式, CTMC1 寄存器中的 CTM1 和 CTM0 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 CTCCLR 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 CTMAF 和 CTMPF 将分别置起。

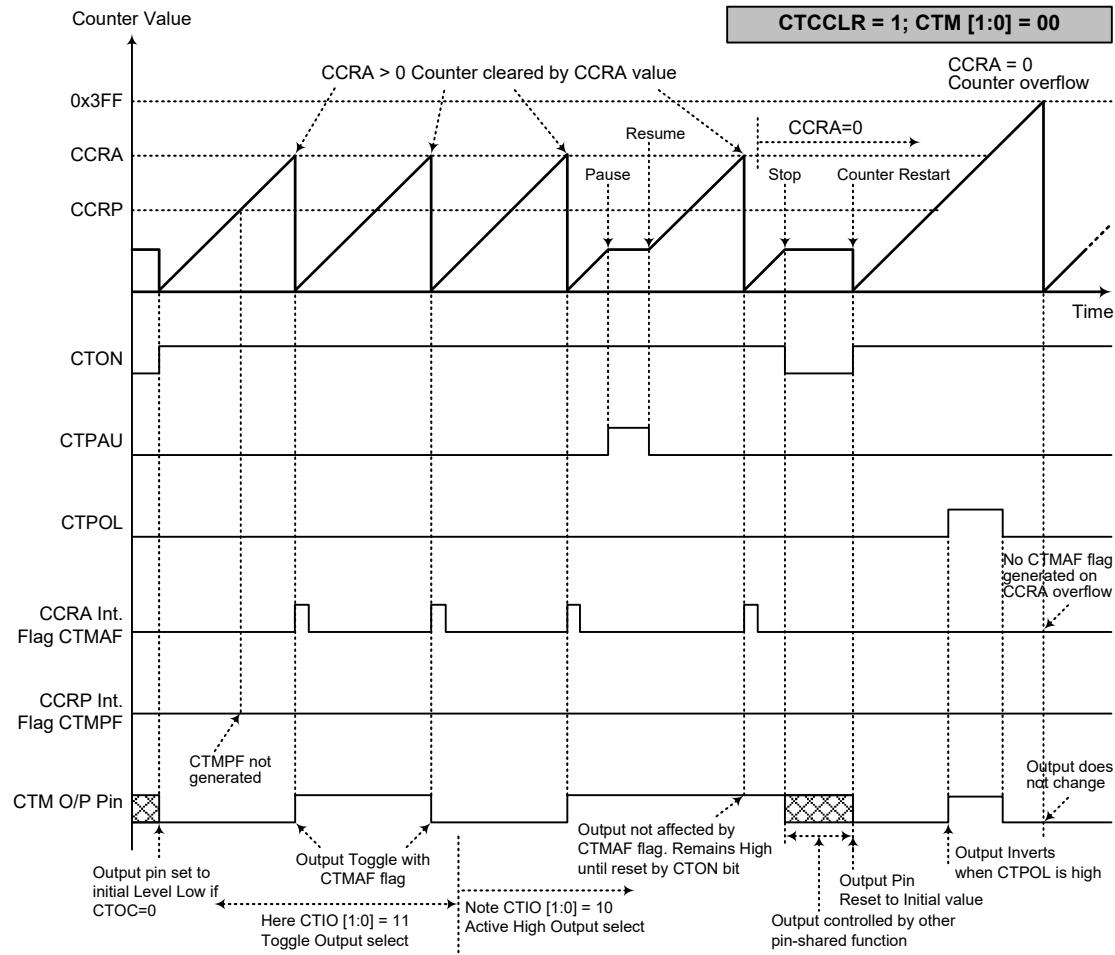
如果 CTMC1 寄存器的 CTCCLR 位设置为高, 当比较器 A 比较匹配发生时计数器被清零。此时, 即使 CCRP 寄存器的值小于 CCRA 寄存器的值, 仅 CTMAF 中断请求标志产生。所以当 CTCCLR 为高时, 不产生 CTMPF 中断请求标志。如果 CCRA 被清零, 当计数达到最大值 3FFH 时, 计数器溢出, 而此时不产生 CTMAF 请求标志。

正如该模式名所言, 当比较匹配发生后, CTM 输出脚状态改变。当比较器 A 比较匹配发生后 CTMAF 标志产生时, CTM 输出脚状态改变。比较器 P 比较匹配发生时产生的 CTMPF 标志不影响 CTM 输出脚。CTM 输出脚状态改变方式由 CTMC1 寄存器中 CTIO1 和 CTIO0 位决定。当比较器 A 比较匹配发生时, CTIO1 和 CTIO0 位决定 TM 输出脚输出高, 低或翻转当前状态。在 CTON 位由低到高电平的变化后, CTM 输出脚初始状态为 CTOC 位所指定的电平。注意, 若 CTIO1 和 CTIO0 位同时为 0 时, 引脚输出不变。



比较匹配输出模式 – CTCCLR=0

- 注: 1. CTCCLR=0, 比较器 P 匹配将清零计数器
 2. CTM 输出脚仅由 CTMAF 标志位控制
 3. 在 CTON 上升沿 CTM 输出脚复位至初始值



比较匹配输出模式 – CTCCLR=1

- 注:
1. CTCCLR=1, 比较器 A 匹配将清零计数器
 2. CTM 输出脚仅由 CTMAF 标志位控制
 3. 在 CTON 上升沿 CTM 输出脚复位至初始值
 4. 当 CTCCLR=1 时, CTMPF 标志位不会产生

定时 / 计数器模式

为使 CTM 工作在此模式, CTMC1 寄存器中的 CTM1 和 CTM0 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 CTM 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 CTM 输出脚可用作普通 I/O 引脚或其它功能。

PWM 输出模式

为使 CTM 工作在此模式, CTMC1 寄存器中的 CTM1 和 CTM0 位需要设置为“10”。CTM 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 CTM 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 输出模式中, CTCCLR 位不影响 PWM 操作。CCRA 和 CCRP 寄存器决定 PWM 波形, 一个用来清除内部计数器并控制 PWM 波形的频率, 另一个用来控制占空比。哪个寄存器控制频率或占空比取决于 CTMC1 寄存器的 CTDPX 位。所以 PWM 波形频率和占空比由 CCRA 和 CCRP 寄存器共同决定。

当比较器 A 或比较器 P 比较匹配发生时, 将产生 CCRA 或 CCRP 中断标志。CTMC1 寄存器中的 CTOC 位决定 PWM 波形的极性, CTIO1 和 CTIO0 位使能 PWM 输出或将 CTM 输出脚置为逻辑高或逻辑低。CTPOL 位对 PWM 输出波形的极性取反。

- **10-bit CTM, PWM 输出模式, 边沿对齐模式, CTDPX=0**

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

若 $f_{sys}=8MHz$, CTM 时钟源选择 $f_{sys}/4$, CCRP=2, CCRA=128,

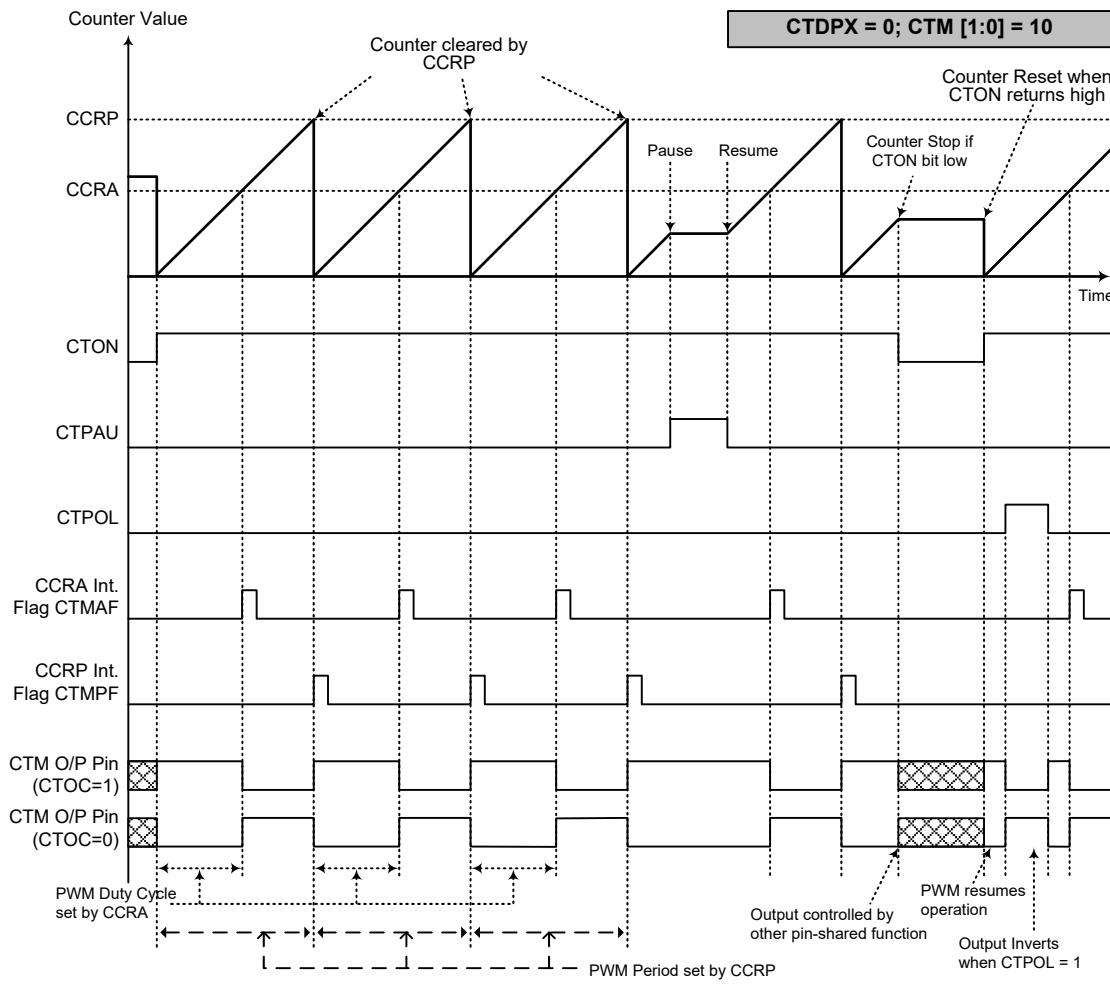
CTM PWM 输出频率 = $(f_{sys}/4)/(2 \times 128) = f_{sys}/1024 = 8kHz$, duty=128/(2×128)=25%。

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

- **10-bit CTM, PWM 输出模式, 边沿对齐模式, CTDPX=1**

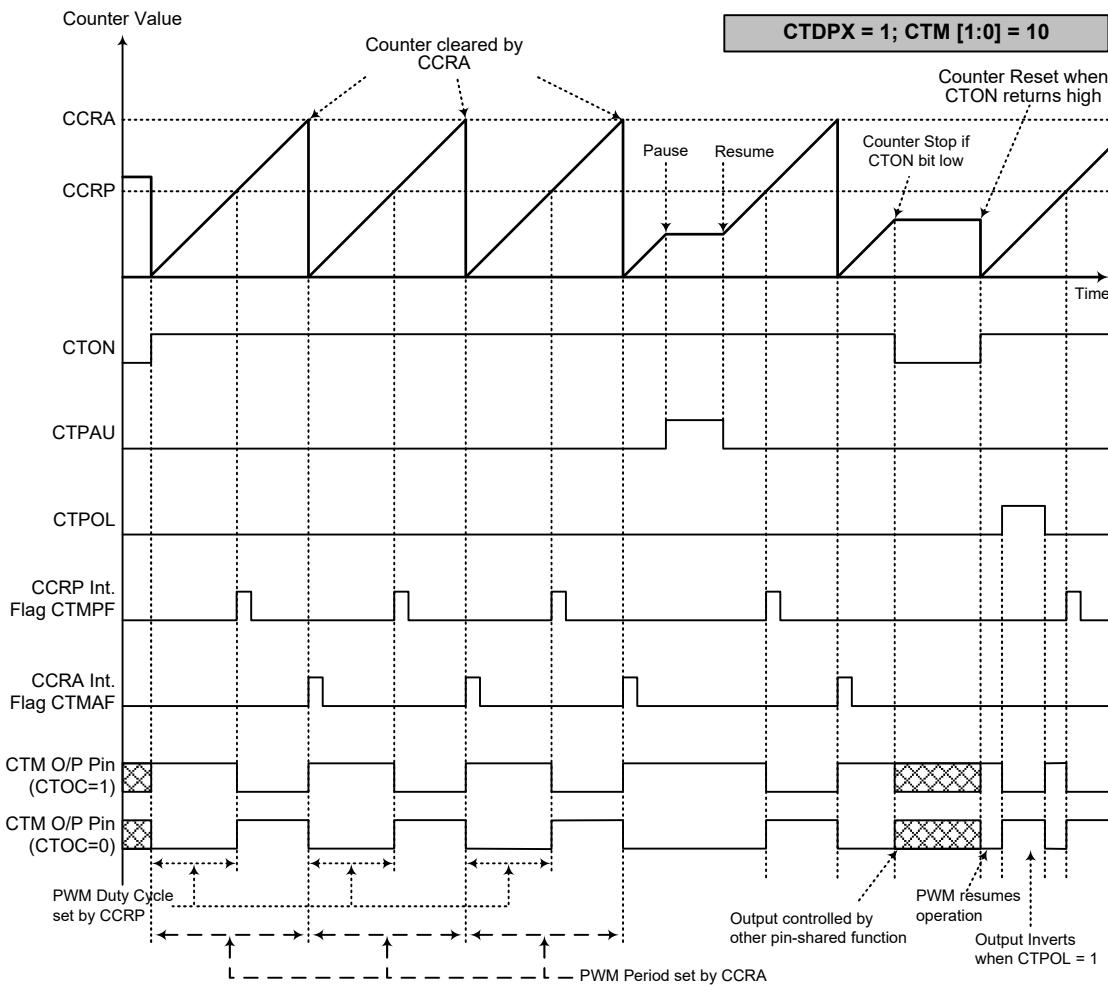
CCRP	1~7	0
Period	CCRA	
Duty	CCRP×128	1024

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



PWM 输出模式 – CTDPX=0

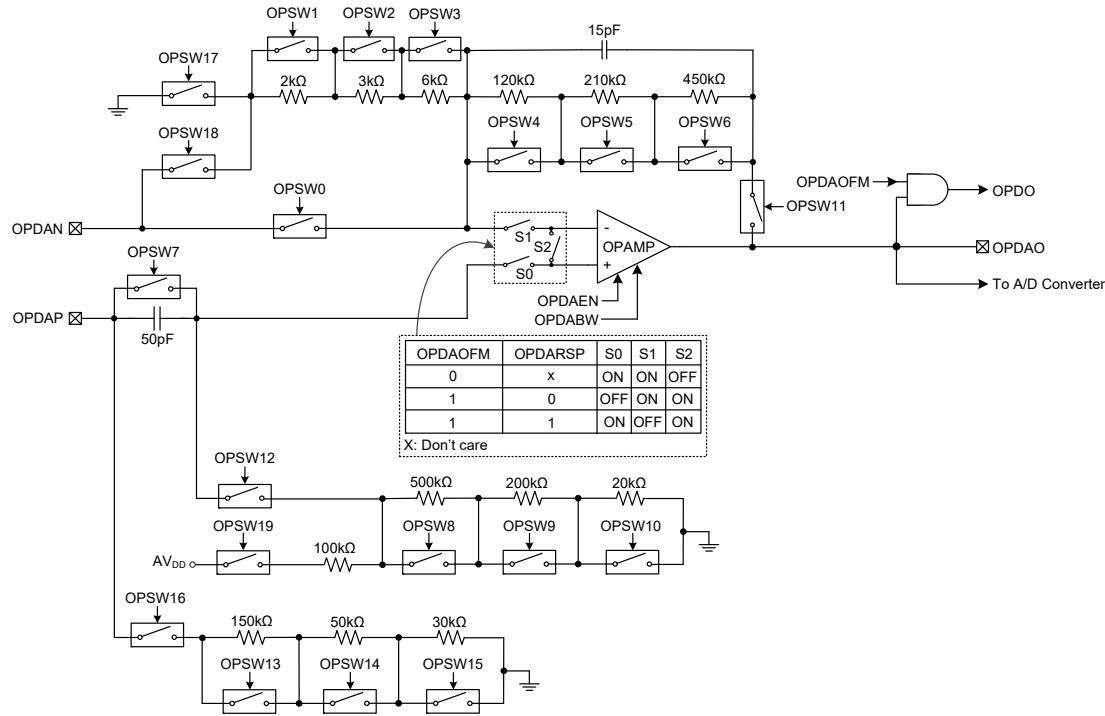
- 注:
1. CTDPX=0, CCRP 清零计数器
 2. 计数器清零并设置 PWM 周期
 3. 当 CTIO[1:0]=00 或 01, PWM 功能不变
 4. CTCCLLR 位不影响 PWM 操作



- 注:
1. CTDPX=1, CCRA 清零计数器
 2. 计数器清零并设置 PWM 周期
 3. 当 CTIO[1:0]=00 或 01, PWM 功能不变
 4. CTCCLR 位不影响 PWM 操作

接近感应电路

该单片机包括一个接近感应电路，由一个运算放大器组成。运算放大器电路可以将一个小信号放大，增益范围从 1 到 391。无论是反相放大还是同相放大，用户可以根据不同的应用程序选择不同的组合。最后，将放大的模拟信号转换成数字信号。



接近感应电路方框图

接近感应电路操作

该电路由 OPDAN 和/或 OPDAP 输入源电压信号。运算放大器通过开关 OPSW0~OPSW19 选择不同的工作模式。运算放大器具有 PGA 功能，其增益可以为正也可为负，取决于电压从正极还是负极输入。OPAM0 可通过 OPSW1~OPSW6 配置其增益为 1~391。

输入信号经 OPAMP 放大后可以直接输出到 OPDAO 引脚上，也可通过设置相关的寄存器将其由内部连接到 A/D 转换器，从而读取放大的输入电压。

接近感应电路寄存器

接近感应电路的所有操作由一系列寄存器控制。OPSWA~OPSWC 寄存器用于模拟开关控制。OPDC 寄存器用于控制运算放大器以及运算放大器低电流或高带宽的选择。OPDACAL 寄存器用于控制运算放大器输入失调电压校准功能。

寄存器 名称	位							
	7	6	5	4	3	2	1	0
OPSWA	OPSW7	OPSW6	OPSW5	OPSW4	OPSW3	OPSW2	OPSW1	OPSW0
OPSWB	OPSW15	OPSW14	OPSW13	OPSW12	OPSW11	OPSW10	OPSW9	OPSW8
OPSWC	—	—	—	—	OPSW19	OPSW18	OPSW17	OPSW16
OPDC	—	—	—	—	—	OPDAEN	OPDO	OPDABW

寄存器 名称	位							
	7	6	5	4	3	2	1	0
OPDACAL	OPDAOFM	OPDARSP	OPDAOF5	OPDAOF4	OPDAOF3	OPDAOF2	OPDAOF1	OPDAOF0

接近感应电路寄存器列表

- OPSWA 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OPSW7	OPSW6	OPSW5	OPSW4	OPSW3	OPSW2	OPSW1	OPSW0
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 **OPSW7:** OPSW7 开关 On/Off 控制

0: Off

1: On

Bit 6 **OPSW6:** OPSW6 开关 On/Off 控制

0: Off

1: On

Bit 5 **OPSW5:** OPSW5 开关 On/Off 控制

0: Off

1: On

Bit 4 **OPSW4:** OPSW4 开关 On/Off 控制

0: Off

1: On

Bit 3 **OPSW3:** OPSW3 开关 On/Off 控制

0: Off

1: On

Bit 2 **OPSW2:** OPSW2 开关 On/Off 控制

0: Off

1: On

Bit 1 **OPSW1:** OPSW1 开关 On/Off 控制

0: Off

1: On

Bit 0 **OPSW0:** OPSW0 开关 On/Off 控制

0: Off

1: On

- OPSWB 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OPSW15	OPSW14	OPSW13	OPSW12	OPSW11	OPSW10	OPSW9	OPSW8
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 **OPSW15:** OPSW15 开关 On/Off 控制

0: Off

1: On

Bit 6 **OPSW14:** OPSW14 开关 On/Off 控制

0: Off

1: On

Bit 5 **OPSW13:** OPSW13 开关 On/Off 控制

0: Off

1: On

Bit 4	OPSW12: OPSW12 开关 On/Off 控制 0: Off 1: On
Bit 3	OPSW11: OPSW11 开关 On/Off 控制 0: Off 1: On
Bit 2	OPSW10: OPSW10 开关 On/Off 控制 0: Off 1: On
Bit 1	OPSW9: OPSW9 开关 On/Off 控制 0: Off 1: On
Bit 0	OPSW8: OPSW8 开关 On/Off 控制 0: Off 1: On

- **OPSWC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	OPSW19	OPSW18	OPSW17	OPSW16
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	OPSW19: OPSW19 开关 On/Off 控制 0: Off 1: On
Bit 2	OPSW18: OPSW18 开关 On/Off 控制 0: Off 1: On
Bit 1	OPSW17: OPSW17 开关 On/Off 控制 0: Off 1: On
Bit 0	OPSW16: OPSW16 开关 On/Off 控制 0: Off 1: On

- **OPDC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	OPDAEN	OPDO	OPDABW
R/W	—	—	—	—	—	R/W	R	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3	未定义, 读为 “0”
Bit 2	OPDAEN: OPD OPAMP 使能 / 除能控制位 0: 除能 – OPDAO 高阻抗 1: 使能
Bit 1	OPDO: 失调校准模式下 OPAMP 输出
Bit 0	OPDABW: OPD OPAMP 低电流 / 高带宽选择位 0: 低电流 1: 高带宽

- OPDACL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OPDAOFM	OPDARSP	OPDAOF5	OPDAOF4	OPDAOF3	OPDAOF2	OPDAOF1	OPDAOF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	0	0	0	0	0

Bit 7 **OPDAOFM:** OPD OPAMP 正常操作或输入失调电压校准模式选择位

- 0: 正常操作
- 1: 失调校准模式

Bit 6 **OPDARSP:** OPD OPAMP 输入失调电压校准参考选择位

- 0: 输入参考电压来自反相端输入
- 1: 输入参考电压来自同相端输入

Bit 5~0 **OPDAOF5~OPDAOF0:** OPD OPAMP 输入失调电压校准控制位

这 6 位用于执行 OPD OPAMP 的输入失调校准操作，并重新储存 OPD OPAMP 的输入失调校准值。更多详细资料请参考“运算放大器输入失调校准”章节。

失调校准步骤

为使运算放大器工作在输入失调校准模式，应先将 OPDAOFM 位设置为 1，然后再设置 OPDARSP 位选择参考输入电压。应注意，由于运算放大器输入引脚与 I/O 引脚或其它功能共用，校准前，应先通过相关引脚共用寄存器将引脚配置为运算放大器输入功能。

运算放大器输入失调校准

- 步骤 1

设置 OPDAOFM=1 且 OPDARSP=1，使运算放大器工作于失调校准模式。为了确保校准后的 V_{os} 尽可能小，校准模式下的输入参考电压应该跟正常模式下的输入直流工作电压相同。

- 步骤 2

设置 OPDAOF[5:0]=000000，读取 OPDO 位。

- 步骤 3

使 OPDAOF[5:0]=OPDAOF[5:0]+1，读取 OPDO 位。

如果 OPDO 位状态不变，重复步骤 3 直到 OPDO 位状态改变。

如果 OPDO 位状态改变，记录此时的 OPDAOF[5:0] 值为 V_{os1} 然后转到步骤 4。

- 步骤 4

设置 OPDAOF[5:0]=111111，读取 OPDO 位。

- 步骤 5

使 OPDAOF[5:0]=OPDAOF[5:0]-1，读取 OPDO 位。

如果 OPDO 位状态不变，重复步骤 5 直到 OPDO 位状态改变。

如果 OPDO 位状态改变，记录此时的 OPDAOF[5:0] 值为 V_{os2} 然后转到步骤 6。

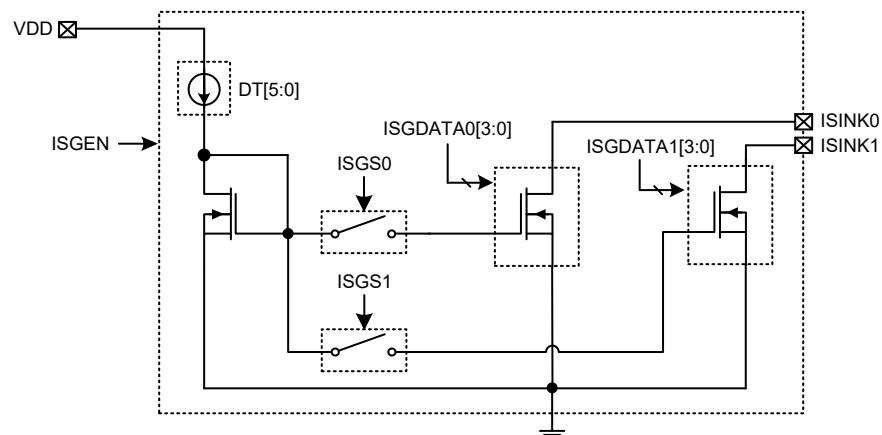
- 步骤 6

将运算放大器输入失调校准值 V_{os} 存入 OPDAOF[5:0] 位中，校准结束。

其中 $V_{os}=(V_{os1}+V_{os2})/2$ 。如果 $(V_{os1}+V_{os2})/2$ 不是整数，舍弃小数。

灌电流发生器

当 V_{ISINK} 电压为 0.7V~4.5V 范围中任意值, 灌电流发生器都可提供恒定的电流。恒流值由 ISGDATA0/ISGDATA1 寄存器控制, 灌电流范围为 2mA~160mA。



注: 在 8-pin SOP/16-pin NSOP-A 封装中 ISINK0 和 ISINK1 打在同一引脚位置, 或者在 16-pin NSOP-B 封装中, 这两个引脚被用户自行外部接在一起使用时, 若 ISGS0/ISGS1=0, 则对应的 ISGDATA0/ISGDATA1 寄存器必须清零, 以避免漏电流情况发生。

灌电流发生器方框图

灌电流发生器寄存器

灌电流发生器所有功能由一系列寄存器控制。

寄存器 名称	位							
	7	6	5	4	3	2	1	0
ISGENC	ISGEN	—	—	—	—	—	ISGS1	ISGS0
ISGDATA0	—	—	—	ISST0	D3	D2	D1	D0
ISGDATA1	—	—	—	ISST1	D3	D2	D1	D0

灌电流发生器寄存器列表

• ISGENC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	ISGEN	—	—	—	—	—	ISGS1	ISGS0
R/W	R/W	—	—	—	—	—	R/W	R/W
POR	0	—	—	—	—	—	0	0

Bit 7 **ISGEN**: 灌电流发生器使能控制

- 0: 除能
- 1: 使能

当 ISGEN 位清零除能灌电流发生器时, ISINK0 和 ISINK1 引脚的状态为: $V_{ISINK0/1} = \text{浮空}$, $I_{ISINK0/1} = 0$ 。

Bit 6~2 未定义, 读为“0”

Bit 1 **ISGS1**: ISINK1 开关 On/Off 控制

- 0: Off
- 1: On

Bit 0 **ISGS0:** ISINK0 开关 On/Off 控制
 0: Off
 1: On

● **ISGDATA0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	ISST0	D3	D2	D1	D0
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	0	0	0	0	0

Bit 7~5 未定义, 读为“0”

Bit 4 **ISST0:** 开关第一级或第二级电流控制

0: 第一级

1: 第二级

Bit 3~0 **D3~D0:** ISINK0 引脚的灌电流控制

第一级电流值 (mA) = $2+2\times D[3:0]$, 2mA/step

第二级电流值 (mA) = $32+8+8\times D[3:0]$, 8mA/step

最大电流 (mA) = $40mA+120mA = 160mA$

● **ISGDATA1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	ISST1	D3	D2	D1	D0
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	0	0	0	0	0

Bit 7~5 未定义, 读为“0”

Bit 4 **ISST1:** 开关第一级或第二级电流控制

0: 第一级

1: 第二级

Bit 3~0 **D3~D0:** ISINK1 引脚的灌电流控制

第一级电流值 (mA) = $2+2\times D[3:0]$, 2mA/step

第二级电流值 (mA) = $32+8+8\times D[3:0]$, 8mA/step

最大电流 (mA) = $40mA+120mA = 160mA$

A/D 转换器

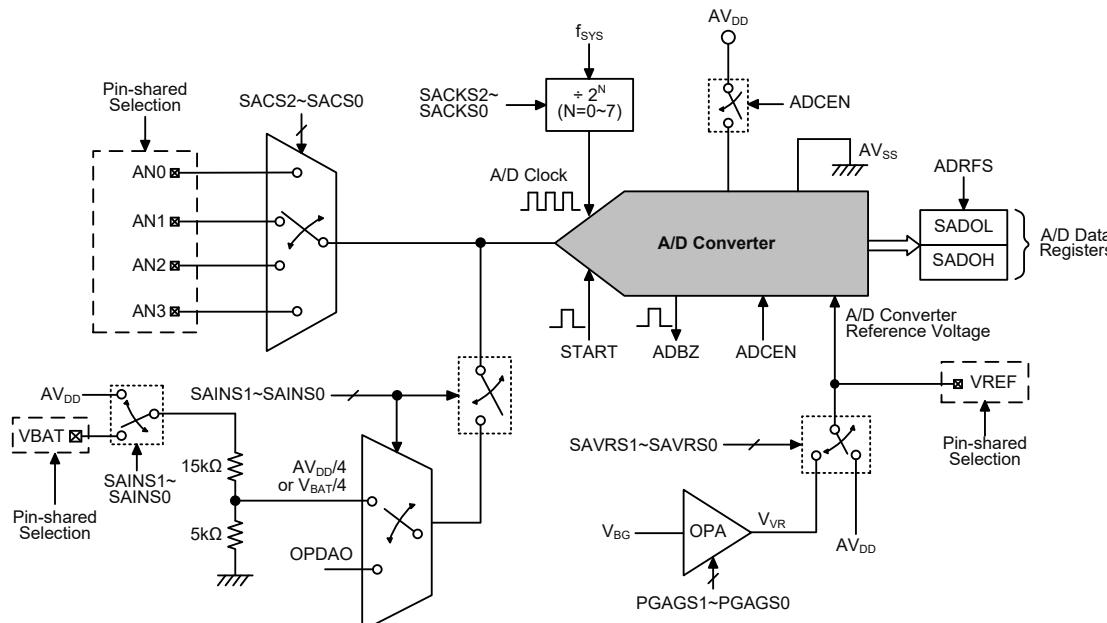
对于大多数电子系统而言，处理现实世界的模拟信号是共同的需求。为了完全由单片机来处理这些信号，首先需要通过 A/D 转换器将模拟信号转换成数字信号。将 A/D 转换器电路集成入单片机，可有效的减少外部器件，随之而来，具有降低成本和减少器件空间需求的优势。

A/D 简介

此单片机包含一个通道 10 位 A/D 转换器，它可以直接接入外部模拟信号（来自传感器或其它控制信号）或内部模拟信号（例如接近感应电路放大器输出）并直接将这些信号转换成 10 位的数字量。选择转换外部或内部模拟信号由 SAINS1~SAINS0 位和 SACS2~SACS0 位共同控制。关于 A/D 输入信号的详细描述请参考“A/D 转换器控制寄存器”和“A/D 转换器输入信号”两节内容。

外部输入通道	内部信号	通道选择位
4: AN0~AN3	3: AV _{DD} /4, V _{BAT} /4, OPDAO	SAINS1~SAINS0, SACS2~SACS0

图显示了 A/D 转换器整体的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一系列寄存器控制。一对只读寄存器来存放 10 位 A/D 转换数据的值。剩下两个控制寄存器设置 A/D 转换器的操作和控制功能。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SADOL (ADRFS=0)	D1	D0	—	—	—	—	—	—
SADOL (ADRFS=1)	D7	D6	D5	D4	D3	D2	D1	D0
SADOH (ADRFS=0)	D9	D8	D7	D6	D5	D4	D3	D2
SADOH (ADRFS=1)	—	—	—	—	—	—	D9	D8
SADC0	START	ADBZ	ADCEN	ADRFS	PGAGS0	SACS2	SACS1	SACS0
SADC1	SAINS1	SAINS0	PGAGS1	SAVRS1	SAVRS0	SACKS2	SACKS1	SACKS0

A/D 转换寄存器列表

A/D 转换器数据寄存器 – SADOL, SADOH

对于具有 10 位 A/D 转换器的芯片，需要两个数据寄存器存放转换结果，一个高字节寄存器 SADOH 和一个低字节寄存器 SADOL。在 A/D 转换完毕后，单片机可以直接读取这些寄存器以获得转换结果。由于寄存器只使用了 16 位中的 10 位，其数据存储格式由 SADC0 寄存器的 ADRFS 位控制，如下表所示。D0~D9 是 A/D 转换数据结果位。未使用的位读为“0”。应注意，当 A/D 转换器除能时，数据寄存器的内容不变。

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

A/D 数据寄存器

A/D 转换器控制寄存器 – SADC0, SADC1

寄存器 SADC0 和 SADC1 用来控制 A/D 转换器的功能和操作。这些 8-bit 的寄存器定义包括选择连接至内部 A/D 转换器的模拟通道，数字化数据格式，A/D 时钟源，并控制和监视 A/D 转换器的忙碌状态，等等。由于单片机只包含一个实际的模数转换电路，因此这些外部和内部模拟信号中的每一个都需要分别被发送到转换器。SADC0 寄存器中的 SACS2~SACS0 位用于选择哪个外部模拟输入通道被连接到内部 A/D 转换器。SADC1 寄存器中的 SAISNS1~SAISNS0 位用于选择外部模拟输入通道或内部模拟信号被连接到内部 A/D 转换器。

引脚共用功能选择寄存器的相关位用来定义 I/O 端口中的哪些引脚为 A/D 转换器的模拟输入，哪些引脚不作为 A/D 转换输入。当引脚作为 A/D 输入时，其原来的 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。

- **SADC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	START	ADBZ	ADCEN	ADRFS	PGAGS0	SACS2	SACS1	SACS0
R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

- Bit 7 **START:** 启动 A/D 转换位
0→1→0: 启动
此位用于启动 A/D 转换过程。通常此位为低，但如果设为高再被清零，将启动 A/D 转换过程。
- Bit 6 **ADBZ:** A/D 转换忙碌标志位
0: A/D 转换结束或未开始转换
1: A/D 转换中
此只读标志位用于表明 A/D 转换过程是否完成。当 START 位由低变为高再变为低时，ADBZ 位为高，表明 A/D 转换已启动。A/D 转换结束后，此位被清零。
- Bit 5 **ADCEN:** A/D 转换器使能 / 除能控制位
0: 除能
1: 使能
此位控制 A/D 内部功能。该位被置高将使能 A/D 转换器。如果该位设为低将关闭 A/D 转换器以降低功耗。当 A/D 转换器除能时，A/D 数据寄存器 SADOH 和 SADOL 的内容将保持不变。
- Bit 4 **ADRFS:** A/D 转换数据格式选择位
0: A/D 转换数据格式 → SADOH=D[9:2]; SADOL=D[1:0]
1: A/D 转换数据格式 → SADOH=D[9:8]; SADOL=D[7:0]
此位控制存放在两个 A/D 数据寄存器中的 10 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 **PGAGS0:** PGA 增益选择位
PGAGS[1:0]=
00: $V_{VR}=1.6V$
01: $V_{VR}=3.0V$
10: $V_{VR}=4.0V$
11: $V_{VR}=1.6V$
注: PGAGS1 位位于 SADC1 寄存器。
- Bit 2~0 **SACS2~SACS0:** A/D 外部模拟通道输入选择位
000: AN0
001: AN1
010: AN2
011: AN3
100~111: 未定义，输入浮空

- **SADC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	SAINS1	SAINS1	PGAGS1	SAVRS1	SAVRS0	SACKS2	SACKS1	SACKS0
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 **SAINS1~SAINS0:** A/D 输入信号选择位
00: 外部输入 – 外部模拟通道输入 AN_n
01: 内部输入 – 接近感应电路 OPAMP 输出，OPDAO
10: 内部输入 – A/D 转换器外部输入电压， $V_{BAT}/4$
11: 内部输入 – A/D 转换器电源电压， $AV_{DD}/4$
当 SAINS1~SAINS0 被设为 “01”~“11” 选择转换内部模拟信号时，需特别注意。
当选择内部模拟信号时，应设置 SACS2~SACS0 位为 “100”~“111”，避免外

部通道输入作为 A/D 输入信号。否则，外部输入通道会和内部模拟信号一起连接至内部 A/D 转换器，这将导致无法预期的损坏。

Bit 5 **PGAGS1:** PGA 增益选择位 1

详见 SADC0 寄存器。

Bit 4~3 **SAVRS1~SAVRS0:** A/D 转换器参考电压选择位

- 00: 外部 VREF 引脚
- 01: 内部 A/D 转换器电源 AV_{DD}
- 10: 内部 A/D 转换器 OPA 输出 V_{VR}
- 11: 外部 VREF 引脚

这两位用于选择 A/D 转换器参考电压。当 SAVRS1~SAVRS0 为被设为“01”或“10”选择内部参考电压时需特别注意。当选择 A/D 转换器电源或 OPA 输出作为 A/D 转换器参考电压时，外部 VREF 引脚需通过引脚共用控制位选择作为其它共用的引脚功能。否则，外部 VREF 输入电压会和内部参考电压一起连接至内部 A/D 转换器，可能会导致内部电路的损坏。

Bit 2~0 **SACKS2~SACKS0:** A/D 时钟源选择位

- 000: f_{sys}
- 001: f_{sys}/2
- 010: f_{sys}/4
- 011: f_{sys}/8
- 100: f_{sys}/16
- 101: f_{sys}/32
- 110: f_{sys}/64
- 111: f_{sys}/128

这三位用于选择 A/D 转换器的时钟源。

● VBGC 寄存器

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

Bit 7~4 未定义，读为“0”

Bit 3 **VBGEN:** V_{BG} Bandgap 参考控制

- 0: 除能
- 1: 使能

注意当 LVR 功能使能或 VBGEN 位置高时，Bandgap 电路使能。

Bit 2~0 未定义，读为“0”

A/D 转换器操作

SADC0 寄存器中的 START 位，用于开启 A/D 转换器。当单片机设置此位从逻辑低到逻辑高，然后再回到逻辑低，就会开始一个模数转换周期。

SADC0 寄存器中的 ADBZ 位用于表明模数转换过程是否正在进行。A/D 转换成功启动后，ADBZ 位会被单片机自动置为“1”。在转换周期结束后，ADBZ 位会自动置为“0”。此外，也会置位中断控制寄存器内相应的 A/D 中断请求标志位，如果中断使能，就会产生对应的内部中断信号。A/D 内部中断信号将引导程序跳转到相应的 A/D 内部中断地址。如果 A/D 内部中断除能，可以让单片机轮询 SADC0 寄存器中的 ADBZ 位，检查此位是否被清除，作为另一种侦测 A/D 转换周期结束的方法。

A/D 转换器的时钟源为系统时钟 f_{sys} 或其分频，而分频系数由 SADC1 寄存器中的 SACKS2~SACKS0 位决定。虽然 A/D 时钟源是由系统时钟 f_{sys} 和 SACKS2~SACKS0 位决定，但可选择的最大 A/D 时钟源则有一些限制。由于允许的 A/D

时钟周期 t_{ADCK} 的范围为 $0.5\mu s \sim 10\mu s$ ，所以选择系统时钟速度时就必须小心。例如，如果系统时钟速度为 8MHz 时，SACKS2~SACKS0 位不能设为“000”、“001”或“111”。必须保证设置的 A/D 转换时钟周期不小于时钟周期的最小值或不大于时钟周期的最大值。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，因为它们超出了 A/D 转换时钟周期规定的范围。

f_{SYS}	A/D 时钟周期 (t_{ADCK})							
	SACKS[2:0] =000 (f_{SYS})	SACKS[2:0] =001 ($f_{SYS}/2$)	SACKS[2:0] =010 ($f_{SYS}/4$)	SACKS[2:0] =011 ($f_{SYS}/8$)	SACKS[2:0] =100 ($f_{SYS}/16$)	SACKS[2:0] =101 ($f_{SYS}/32$)	SACKS[2:0] =110 ($f_{SYS}/64$)	SACKS[2:0] =111 ($f_{SYS}/128$)
1MHz	1 μs	2 μs	4 μs	8 μs	16 μs *	32 μs *	64 μs *	128 μs *
2MHz	500ns	1 μs	2 μs	4 μs	8 μs	16 μs *	32 μs *	64 μs *
4MHz	250ns *	500ns	1 μs	2 μs	4 μs	8 μs	16 μs *	32 μs *
8MHz	125ns *	250ns *	500ns	1 μs	2 μs	4 μs	8 μs	16 μs *

A/D 时钟周期范例

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

A/D 转换器参考电压

A/D 转换器参考电压来自正电源电压 AV_{DD} 、A/D 转换器 OPA 输出 V_{VR} 或外部参考源引脚 VREF，通过 SAVRS1 和 SAVRS0 位选择。当 SAVRS1~SAVRS0 位为“01”时，A/D 转换器参考电压来自 AV_{DD} 。当 SAVRS1~SAVRS0 位为“10”时，A/D 转换器参考电压来自 OPA 输出。否则，A/D 转换器参考电压来自 VREF 引脚。由于 VREF 引脚与其它功能共用，当选择 VREF 引脚作为参考电压源时，需先正确设置引脚共用选择位将 VREF 引脚配置为参考电压输入功能。然而，当内部 A/D 转换器电源或 OPA 输出被选作参考电压源时，相关的引脚共用控制位不可选择 VREF 参考电压输入功能，避免 VREF 引脚电压跟内部参考电压信号一起接入 A/D 转换器。

模拟输入值一定不能超过所选的参考电压值。

SAVRS[1:0]	参考源	说明
00, 11	VREF 引脚	外部 A/D 转换器参考引脚 VREF
01	AV_{DD}	内部 A/D 转换器电源电压
10	V_{VR}	内部 A/D 转换器 OPA 输出电压

A/D 转换器参考电压选择

A/D 转换器输入信号

所有的 A/D 模拟输入引脚都与 I/O 口及其它功能共用。使用 PxS0 和 PxS1 寄存器中的相应位，可以将它们设置为 A/D 转换器模拟输入脚或具有其它功能。如果对应的引脚作为 A/D 转换输入，那么它原来的引脚功能将除能。通过这种方式，引脚的功能可由程序来控制，灵活地切换引脚功能。如果将引脚设为 A/D 输入，则通过寄存器编程设置的所有上拉电阻会自动断开。请注意，端口控制寄存器不需要为使能 A/D 输入而先设定为输入模式，当 A/D 输入功能选择位使能 A/D 输入时，端口控制寄存器的状态将被重置。

另外还有几个内部模拟信号可作为 A/D 转换器的模拟输入信号，分别来自接近感应电路运算放大器输出、A/D 转换器电源电压或 A/D 转换器 OPA 输出，通过设置 SAINS1~SAINS0 位来选择。若 SAINS1~SAINS0 位为“00”，则选择转换外部模拟输入信号，具体通道编号由 SACS2~SACS0 位决定。若选择内部模拟信号时，应将 SACS2~SACS0 位设置为一个适当的值，以关闭外部模拟通道输入。否则，外部通道输入会与内部模拟信号一起接入从而导致错误。

SAINS[1:0]	SACS[2:0]	输入信号	说明
00	000~011	AN0~AN3	外部模拟通道输入
	100~111	—	未选择外部通道，输入浮空
01	100~111	OPDAO	接近感应电路 OPAMP 输出电压
10	100~111	$V_{BAT}/4$	A/D 转换器外部输入 $V_{BAT}/4$
11	100~111	$AV_{DD}/4$	A/D 转换器电源电压 $AV_{DD}/4$

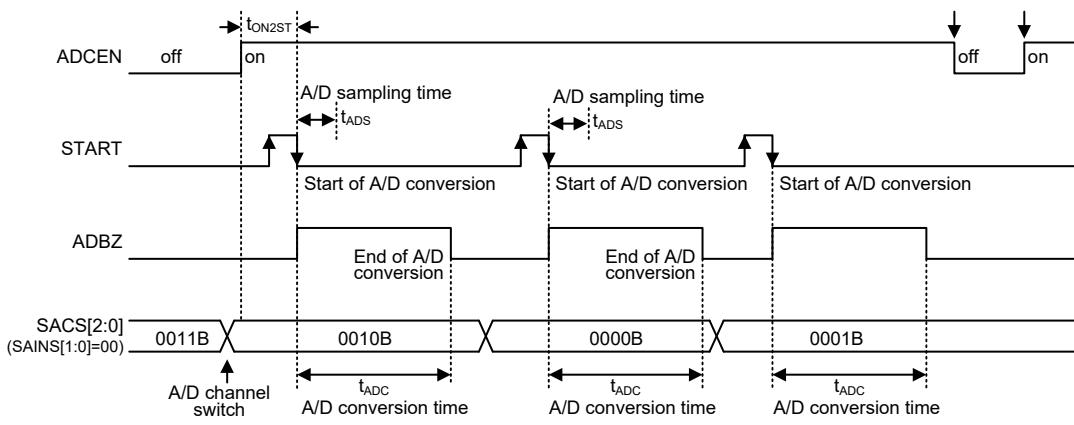
A/D 转换器输入信号选择

A/D 转换率及时序图

一个完整的 A/D 转换包含两部分，数据采样和数据转换。数据采样时间为 t_{ADS} ，需要 4 个 A/D 时钟周期，而数据转换需要 10 个 A/D 时钟周期。所以一个完整的 A/D 转换时间， t_{ADC} ，一共需要 14 个 A/D 时钟周期。

$$\text{最大 A/D 转换率} = 1/(\text{A/D 时钟周期} \times 14)$$

下列时序图表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 14 t_{ADCK} ， t_{ADCK} 为 A/D 时钟周期。



A/D 转换时序图 – 外部通道输入

A/D 转换步骤概述

下面概述实现 A/D 转换过程的各个步骤。

- 步骤 1
通过 SADC1 寄存器中的 SACKS2~SACKS0 位，选择所需的 A/D 转换时钟。
- 步骤 2
将 SADC0 寄存器中的 ADCEN 位置高使能 A/D 转换器。
- 步骤 3
通过配置 SAINS1~SAINS0 位，选择连接至内部 A/D 转换器的信号。

若选择外部通道输入，接着执行步骤 4。

若选择内部模拟信号，接着执行步骤 5。

- 步骤 4

若已通过 SAINS1~SAINS0 位选择 A/D 输入信号来自外部通道输入，接着应设置相关的引脚共用控制位将该引脚规划为 A/D 输入引脚。通过设置 SACS2~SACS0 位选择哪个外部通道接至 A/D 转换器。接着执行步骤 6。

- 步骤 5

选择内部模拟信号前，应正确设置 SACS2~SACS0 位，将外部通道输入切换到无通道输入。然后再设置 SAINS1~SAINS0 位选择所需的内部模拟信号。接着执行步骤 6。

- 步骤 6

通过 SADC1 寄存器中的 SAVRS1~SAVRS0 位选择参考电压。此步骤之注意事项请参考 A/D 转换器参考电压选择章节。

- 步骤 7

设置 SADC0 寄存器中的 ADRFS 位选择 A/D 转换器输出数据格式。

- 步骤 8

如果要使用中断，则中断控制寄存器需要正确地设置，以确保 A/D 中断功能是激活的。总中断控制位 EMI 需要置位为“1”，以及 A/D 转换器中断位 ADE 也需要置位为“1”。

- 步骤 9

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

- 步骤 10

如果 A/D 转换正在进行中，ADBZ 位会被置为逻辑高。A/D 转换完成后，ADBZ 位会被置为逻辑低，并可从 SADOH 和 SADOL 寄存器中读取输出数据。

注：若使用轮询 SADC0 寄存器中 ADBZ 位的状态的方法来检查转换过程是否结束时，则中断使能的步骤可以省略。

编程注意事项

在编程时，如果 A/D 转换器未使用，通过设置 SADC0 寄存器中的 ADCEN 为低，关闭 A/D 内部电路以减少电源功耗。此时，不考虑输入脚的模拟电压，内部 A/D 转换器电路不产生功耗。如果 A/D 转换器输入脚用作普通 I/O 脚，必须特别注意，输入电压为无效逻辑电平也可能增加功耗。

A/D 转换功能

单片机含有一组 10 位的 A/D 转换器，它们转换的最大值可达 3FFH。由于模拟输入最大值等于实际 A/D 转换器参考电压值 V_{REF} ，因此每一位可表示 $V_{REF}/1024$ 的模拟输入值。

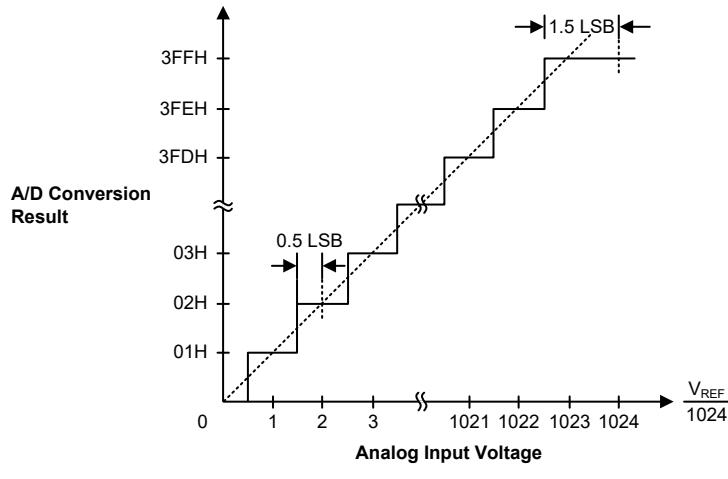
$$1 \text{ LSB} = V_{REF} \div 1024$$

通过下面的等式可估算 A/D 转换器输入电压值：

$$\text{A/D 输入电压} = \text{A/D 数字输出值} \times (V_{REF} \div 1024)$$

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

注意，这里的 V_{REF} 电压指的是通过 SAVRS 字段选择的实际输入 A/D 转换器的参考电压。



理想的 A/D 转换功能

A/D 转换应用范例

下面两个范例程序用来说明怎样使用 A/D 转换。第一个范例是轮询 SADC0 寄存器中的 ADBZ 位来判断 A/D 转换是否完成；第二个范例则使用中断的方式判断。

范例：使用查询 ADBZ 的方式来检测转换结束

```

clr ADE           ; disable ADC interrupt
mov a,03H
mov SADC1,a      ; select input signal from external channel
                  ; input, reference voltage from AVDD fsys/8 as A/D
                  ; clock
mov a,01h
mov PAS0,a        ; setup PAS0 register to configure pin AN0
mov a,20h
mov SADC0,a      ; enable A/D converter and connect AN0 channel
                  ; to A/D converter
:
start_conversion:
clr START          ; high pulse on start bit to initiate conversion
set START          ; reset A/D
clr START          ; start A/D
polling_EOC:
sz ADBZ           ; poll the SADC0 register ADBZ bit to detect end
                  ; of A/D conversion
jmp polling_EOC  ; continue polling
mov a,SADOL        ; read low byte conversion result value
mov SADOL_buffer,a
mov a,SADOH        ; read high byte conversion result value
mov SADOH_buffer,a
:
:
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 SADC1,a      ; select input signal from external channel
                  ; input, reference voltage from AVDD fsys/8 as A/D
                  ; clock
mov a,01h
mov PAS0,a
mov a,20h
mov SADC0,a      ; enable A/D converter and connect AN0 channel
                  ; to A/D converter

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

```

触控按键功能

该单片机提供多个触控按键功能。触控按键功能完全内部集成不需外接元件，可通过对内部寄存器的简单操作来实现此功能。

触控按键结构

触控按键引脚与 I/O 引脚共用，通过相应的引脚共用选择寄存器来选择此功能。按键被分成一组，即模块 0。此模块包含两个触控按键且每个按键有各自的振荡器。该模块具有单独的控制逻辑电路和配套的寄存器系列。

触控按键总数	触控按键	共用引脚
2	KEY1~KEY2	PB2~PB3

触控按键结构

触控按键寄存器定义

触控按键模块 0 包含 2 个触控按键功能，由一系列寄存器控制。以下表格列出了触控按键模块 0 的寄存器。

寄存器名称	说明
TKTMR	触控按键时隙 8-bit 计数器预载寄存器
TKC0	触控按键功能控制寄存器 0
TKC1	触控按键功能控制寄存器 1
TK16DL	触控按键功能 16-bit 计数器低字节
TK16DH	触控按键功能 16-bit 计数器高字节
TKM016DL	触控按键模块 0 16-bit C/F 计数器低字节
TKM016DH	触控按键模块 0 16-bit C/F 计数器高字节
TKM0ROL	触控按键模块 0 参考振荡器电容选择低字节
TKM0ROH	触控按键模块 0 参考振荡器电容选择高字节
TKM0C0	触控按键模块 0 控制寄存器 0
TKM0C1	触控按键模块 0 控制寄存器 1

触控按键功能寄存器定义

寄存器 名称	位							
	7	6	5	4	3	2	1	0
TKTMR	D7	D6	D5	D4	D3	D2	D1	D0
TKC0	—	TKRCOV	TKST	TKCFOV	TK16OV	—	TK16S1	TK16S0
TKC1	—	—	—	—	—	—	TKFS1	TKFS0
TK16DL	D7	D6	D5	D4	D3	D2	D1	D0
TK16DH	D15	D14	D13	D12	D11	D10	D9	D8
TKM016DL	D7	D6	D5	D4	D3	D2	D1	D0
TKM016DH	D15	D14	D13	D12	D11	D10	D9	D8
TKM0ROL	D7	D6	D5	D4	D3	D2	D1	D0
TKM0ROH	—	—	—	—	—	—	D9	D8
TKM0C0	M0MXS1	M0MXS0	M0DFEN	M0FILEN	M0SOF0	M0SOF2	M0SOF1	M0SOF0
TKM0C1	M0TSS	—	M0ROEN	M0KOEN	—	—	M0K2EN	M0K1EN

触控按键功能寄存器列表

- **TKTMR 寄存器**

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

Bit 7~0

D7~D0: 触控按键时隙 8-bit 计数器预载寄存器

触控按键时隙计数器预载寄存器用于确定触控按键时隙溢出时间。时隙单位周期通过一个 5-bit 计数器获得，等于 32 个时隙时钟周期。因此，时隙计数器溢出时间可由下面的等式算出。

时隙计数器溢出时间 = $(256 - \text{TKTMR}[7:0]) \times 32t_{\text{tsc}}$ ，其中 t_{tsc} 为时隙计数器时钟周期。

- **TKC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	TKRCOV	TKST	TKCFOV	TK16OV	—	TK16S1	TK16S0
R/W	—	R/W	R/W	R/W	R/W	—	R/W	R
POR	—	0	0	0	0	—	0	0

Bit 7 未定义，读为“0”

Bit 6 **TKRCOV:** 触控按键时隙计数器溢出标志位

- 0: 无溢出
- 1: 溢出

当模块 0 时隙计数器溢出将此位置为“1”时，相应的触控按键中断请求标志位 TKMF 也会同时置位，且所有模块按键振荡器和参考振荡器将自动停止。此时触控按键模块 16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器都会自动关闭。然而若是通过应用程序将此位设置为“1”时，相应的中断请求标志位不会受到影响。因此，此位不能通过应用程序置位但必须通过应用程序清零。

Bit 5 **TKST:** 触控按键检测开启控制位

- 0: 停止或无操作
- 0→1: 开始检测

当该位为“0”时，模块的 16-bit C/F 计数器、触控按键功能 16-bit 计数器和 5-bit 时隙单位周期计数器会自动清零，但 8-bit 可编程时隙计数器不会被清零。当该位由 0 到 1 转变时，16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器都会自动开启，并使能按键振荡器和参考振荡器以驱动相应的计数器。

Bit 4 **TKCFOV:** 触控按键模块 16-bit C/F 计数器溢出标志位

- 0: 无溢出
- 1: 溢出

该位由触控按键模块 16-bit C/F 计数器溢出置位，必须通过应用程序清零。

Bit 3 **TK16OV:** 触控按键功能 16-bit 计数器溢出标志位

- 0: 无溢出
- 1: 溢出

该位由触控按键功能 16-bit 计数器溢出置位，必须通过应用程序清零。

Bit 2 未定义，读为“0”

Bit 1~0 **TK16S1~TK16S0:** 触控按键功能 16-bit 计数器时钟选择位

- 00: f_{sys}
- 01: f_{sys}/2
- 10: f_{sys}/4
- 11: f_{sys}/8

- **TKC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	TKFS1	TKFS0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	1	1

Bit 7~2 未定义, 读为“0”

Bit 1~0 **TKFS1~TKFS0:** 触控按键振荡器和参考振荡器频率选择位

- 00: 1MHz
- 01: 3MHz
- 10: 7MHz
- 11: 11MHz

- **TK16DH/TK16DL – 触控按键功能 16-bit 计数器寄存器对**

寄存器	TK16DH								TK16DL							
	位	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1
Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

该寄存器对用于存储触控按键功能 16-bit 计数器值。该 16-bit 计数器可用于校准参考振荡器或按键振荡器频率。当触控按键时隙计数器溢出, 此 16-bit 计数器将停止, 计数器内容保持不变。当 TKST 位为“0”时, 该寄存器对将被清零。

- **TKM016DH/TKM016DL – 触控按键模块 0 16-bit C/F 计数器寄存器对**

寄存器	TKM016DH								TKM016DL							
	位	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1
Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

该寄存器对用于存储触控按键模块 0 16-bit C/F 计数器值。当触控按键时隙计数器溢出, 该 16-bit C/F 计数器将被停止, 其内容保持不变。当 TKST 位为“0”时, 该寄存器对将被清零。

- **TKM0ROH/TKM0ROL 寄存器 – 触控按键模块 0 参考振荡器电容选择寄存器对**

寄存器	TKM0ROH								TKM0ROL							
	位	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1
Name	—	—	—	—	—	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
R/W	—	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0

该寄存器对用于存储触控按键模块 0 参考振荡器电容值。

参考振荡器内部电容值 = $(\text{TKM0RO}[9:0] \times 30\text{pF}) / 1024$

- **TKM0C0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	M0MXS1	M0MXS0	M0DFEN	M0FILEN	M0SOF0	M0SOF2	M0SOF1	M0SOF0
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 **M0MXS1~M0MXS0:** 多路复用按键选择
 00: KEY1
 01: KEY2
 10: 保留
 11: 保留
- Bit 5 **M0DFEN:** 触控按键模块 0 倍频功能控制位
 0: 除能
 1: 使能
 此位用于控制触控按键振荡器的倍频功能。当此位置 1, 按键振荡器频率将为原来的两倍。
- Bit 4 **M0FILEN:** 触控按键模块 0 滤波器功能控制位
 0: 除能
 1: 使能
- Bit 3 **M0SOF0:** 触控按键模块 0 C/F 振荡器跳频功能控制位
 0: 由 M0SOF2~M0SOF0 位控制
 1: 由硬件电路控制
 该位用来选择触控按键振荡器跳频功能控制方式。当此位置 1, 按键振荡器跳频功能由硬件电路控制, MSOF2~MSOF0 位的设置无效。
- Bit 2~0 **M0SOF2~M0SOF0:** 触控按键模块 0 参考振荡器和按键振荡器跳频选择位 (当 M0SOF0=0)
 000: 1.020MHz
 001: 1.040MHz
 010: 1.059MHz
 011: 1.074MHz
 100: 1.085MHz
 101: 1.099MHz
 110: 1.111MHz
 111: 1.125MHz
 这些位用于触控按键振荡器跳频功能的频率选择。注意, 只有当 M0SOF0 位为零时, 这些位的选择才有效。
 上述频率会随着外部或内部电容值的不同而变化。若触控按键振荡器频率选择 1MHz, 用户选择其它频率时可依比例调整。

- **TKM0C1 寄存器**

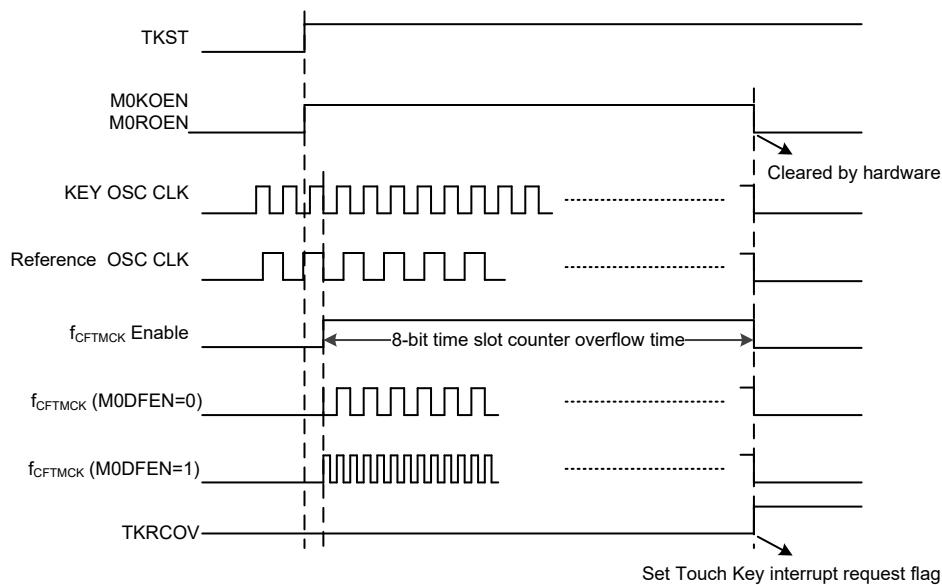
Bit	7	6	5	4	3	2	1	0
Name	M0TSS	—	M0ROEN	M0KOEN	—	—	M0K2EN	M0K1EN
R/W	R/W	—	R/W	R/W	—	—	R/W	R/W
POR	0	—	0	0	—	—	0	0

- Bit 7 **M0TSS:** 触控按键模块 0 时隙计数器时钟源选择位
 0: 触控按键模块 0 参考振荡器
 1: $f_{SYS}/4$
- Bit 6 未定义, 读为 “0”
- Bit 5 **M0ROEN:** 触控按键模块 0 参考振荡器使能控制位
 0: 除能
 1: 使能

Bit 4	M0KOEN: 触控按键模块 0 按键振荡器使能控制位 0: 除能 1: 使能
Bit 3~2	未定义, 读为“0”
Bit 1	M0K2EN: 触控按键模块 0 KEY2 使能控制 0: 除能 1: 使能
Bit 0	M0K1EN: 触控按键模块 0 KEY1 使能控制 0: 除能 1: 使能

触控按键操作

手指接近或接触到触控面板时, 面板的电容量会增大, 电容量的变化会轻微改变内部感应振荡器的频率, 通过测量频率的变化可以感知触控动作。参考时钟通过内部可编程分频器能够产生一个固定的时间周期。在这个时间周期内, 通过在此固定时间周期内对感应振荡器产生的时钟周期计数, 可确定触控按键的动作。



触控按键扫描模式时序图

触控按键模块 0 包含两个与 I/O 引脚共用的触控按键, KEY1~KEY2, 通过相关引脚共用控制寄存器位可选择相应引脚功能。每个触控按键具有独立的感应振荡器, 因此触控按键模块 0 包含两个感应振荡器。

在参考时钟固定的时间间隔内, 测量感应振荡器产生的时钟周期数。测到的周期数可以用于判断触控动作是否有效发生。在此固定的时间间隔结束后, 会产生一个触控按键中断信号。

当 TKC0 寄存器中的 TKST 位清零时, 模块 0 的 16-bit C/F 计数器、触控按键功能 16-bit 计数器和 5-bit 时隙单位周期计数器会自动清零, 而 8-bit 可编程时隙计数器不清零, 由用户设置溢出时间。在 TKST 位由低变高时, 16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器会自动开启。

当时隙计数器溢出，模块 0 的按键振荡器和参考振荡器都会自动停止且 16-bit C/F 计数器、触控按键功能 16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器也会自动停止。时隙计数器时钟源可通过 TKMC1 寄存器中的 MTSS 位选择来自参考振荡器或 $f_{SYS}/4$ 。通过设置 TKMC1 寄存器中的 MROEN 位和 MKOEN 为“1”，可使能参考振荡器和按键振荡器。

当模块 0 时隙计数器溢出时，将产生一个触控按键中断。这里所说的触控按键是指已使能的触控按键。

触控按键中断

触控按键只有一个中断，模块 0 时隙计数器溢出时，才产生中断，这里所说的触控按键是指已使能的触控按键。此时 16-bit C/F 计数器、16-bit 计数器、5-bit 时隙单位周期计数器和 8-bit 时隙计数器会自动清零。更多详细内容见规格书中断章节中的“触控按键中断”。

编程注意事项

相关寄存器设置后，将 TKST 位由低电平变为高电平会启动触控按键检测程序。此时所有相关的振荡器将使能并同步。当计数器溢出时，时隙计数器标志位 TKRCOV 将变为高电平，同时还会产生一个中断信号。由于 TKRCOV 标志位无法被自动清零，需通过应用程序将此位清零。

当触控按键模块的 16-bit C/F 计数器溢出就会把 16-bit C/F 计数器溢出标志位 TKCFOV 置高。由于此标志位无法被自动清零，需通过应用程序将此位清零。16-bit 计数器溢出就会把其溢出标志位 TK16OV 置高。由于此标志位无法被自动清零，需通过应用程序将此位清零。

当外部触控按键的大小和布局确定时，其相关的电容将决定感应振荡器的频率。

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、时基、EEPROM、触控按键和 A/D 转换器等产生。

中断寄存器

中断控制基本上是在一定单片机条件发生时设置请求标志位，应用程序中中断使能位的设置是通过位于特殊功能数据存储器中的一系列寄存器控制的。寄存器总的分为两类。第一类是 INTC0~INTC2 寄存器，用于设置基本的中断；第二类是 INTEG 寄存器，用于设置外部中断边沿触发类型。

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

功能	使能位	请求标志	注释
总中断	EMI	—	—
INT 引脚	INTE	INTF	—
触控按键	TKME	TKMF	—
A/D 转换器	ADE	ADF	—
CTM	CTMPE	CTMPF	—
	CTMAE	CTMAF	
EEPROM	DEE	DEF	—
时基	TBnE	TBnF	n=0~1

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	—	—	—	—	—	—	INTS1	INTS0
INTC0	—	ADF	TKMF	INTF	ADE	TKME	INTE	EMI
INTC1	CTMPF	TB1F	TB0F	DEF	CTMPE	TB1E	TB0E	DEE
INTC2	—	—	—	CTMAF	—	—	—	CTMAE

中断寄存器列表

● INTEG 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	INTS1	INTS0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义, 读为“0”

Bit 1~0 **INTS1~INTS0:** INT 引脚中断边沿控制位

- 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	—	ADF	TKMF	INTF	ADE	TKME	INTE	EMI
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7 未定义, 读为“0”

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 **TKMF:** 触控按键中断请求标志位

- 0: 无请求
- 1: 中断请求

Bit 4 **INTF:** INT 中断请求标志位

- 0: 无请求
- 1: 中断请求

Bit 3	ADE: A/D 转换器中断控制位 0: 除能 1: 使能
Bit 2	TKME: 触控按键中断控制位 0: 除能 1: 使能
Bit 1	INTE: INT 中断控制位 0: 除能 1: 使能
Bit 0	EMI: 总中断控制位 0: 除能 1: 使能

- **INTC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	CTMPF	TB1F	TB0F	DEF	CTMPE	TB1E	TB0E	DEE
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	CTMPF: CTM 比较器 P 匹配中断请求标志位 0: 无请求 1: 中断请求
Bit 6	TB1F: 时基 1 中断请求标志位 0: 无请求 1: 中断请求
Bit 5	TB0F: 时基 0 中断请求标志位 0: 无请求 1: 中断请求
Bit 4	DEF: 数据 EEPROM 中断请求标志位 0: 无请求 1: 中断请求
Bit 3	CTMPE: CTM 比较器 P 匹配中断控制位 0: 除能 1: 使能
Bit 2	TB1E: 时基 1 中断控制位 0: 除能 1: 使能
Bit 1	TB0E: 时基 0 中断控制位 0: 除能 1: 使能
Bit 0	DEE: 数据 EEPROM 中断控制位 0: 除能 1: 使能

- INTC2 寄存器

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

Bit 7~5 未定义, 读为“0”

Bit 4 **CTMAF**: CTM 比较器 A 匹配中断请求标志位

0: 无请求

1: 中断请求

Bit 3~1 未定义, 读为“0”

Bit 0 **CTMAE**: CTM 比较器 A 匹配中断控制位

0: 除能

1: 使能

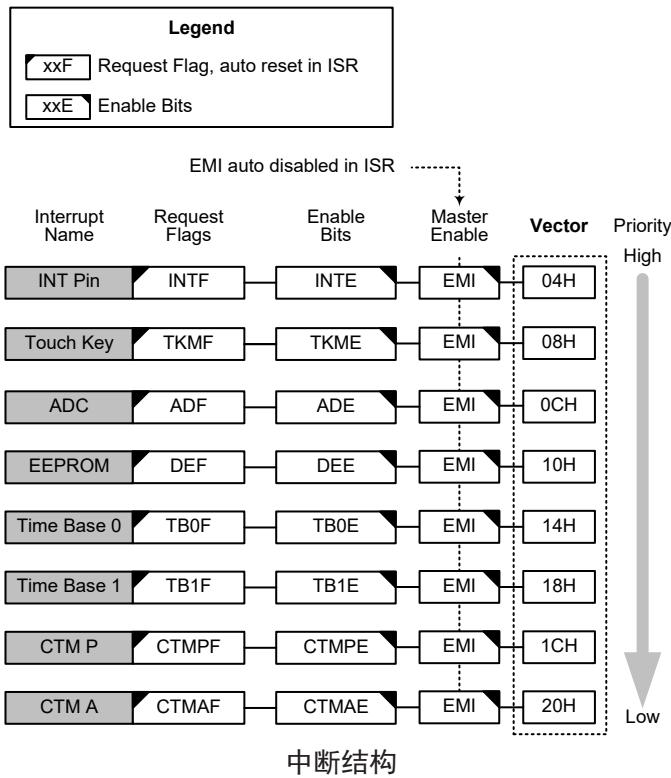
中断操作

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

当中断发生时, 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。所有中断源有自己的向量。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。



外部中断

通过 INT 引脚上的信号变化可控制外部中断。当触发沿选择位设置好触发类型, INT 引脚的状态发生变化, 外部中断请求标志 INTF 被置位时外部中断请求产生。若要跳转到相应中断向量地址, 总中断控制位 EMI 和相应中断使能位 INTE 需先被置位。此外, 必须使用 INTEG 寄存器使能外部中断功能并选择触发沿类型。外部中断引脚和普通 I/O 口共用, 如果相应寄存器中的中断使能位被置位, 并且通过引脚共用寄存器选择外部中断脚, 此引脚将被作为外部中断脚使用。此时该引脚必须通过设置控制寄存器, 将该引脚设置为输入口。当中断使能, 堆栈未满并且外部中断脚状态改变, 将调用外部中断向量子程序。当响应外部中断服务子程序时, 中断请求标志位 INTF 会自动复位且 EMI 位会被清零以除能其它中断。注意, 即使此引脚被用作外部中断输入, 其上拉电阻选项仍保持有效。

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。

触控按键中断

当触控按键模块 0 中的时隙计数器溢出时, 触摸按键中断请求标志 TKMF 将被置位, 触控按键中断产生。要使触控按键中断发生, 总中断控制位 EMI 和触摸按键中断使能位 TKME 必须先被置位。当中断使能, 堆栈未满且触控按键模块 0 时隙计数器溢出发生时, 将调用相应中断向量处的子程序。当响应中断服务子程序时, 中断请求标志位 TKMF 会被自动复位且 EMI 位会被清零以除能其它中断。

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 ADF 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 A/D 转换器中断向量程序。当响应中断服务子程序时，A/D 转换器中断请求位 ADF 会自动清零，EMI 位也会被清零以除能其它中断。

TM 中断

简易型 TM 有两个中断，分别来自比较器 P 和比较器 A 匹配。简易型 TM 有两个中断请求标志位，CTMAF 和 CTMPF，及两个使能位，CTMAE 和 CTMPE。当 TM 比较器 P、A 匹配情况发生时，相应 TM 中断请求标志被置位，TM 中断请求产生。

若要程序跳转到相应中断向量地址，总中断控制位 EMI 和相应的 TM 中断使能位 CTMAE 或 CTMPE 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，可跳转至相应的 TM 中断向量程序中执行。当 TM 中断响应，TM 中断请求标志位，CTMAF 或 CTMPF，会被自动清零，EMI 位也会被清零以除能其它中断。

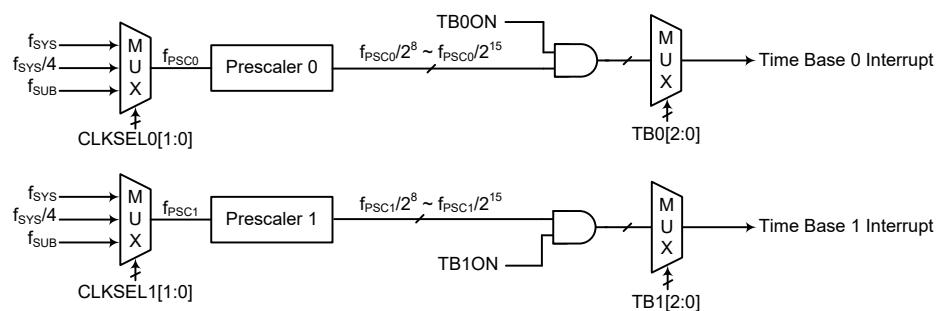
EEPROM 中断

当写周期结束，EEPROM 中断请求标志 DEF 被置位，EEPROM 中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI 和 EEPROM 中断使能位 DEE 需先被置位。当中断使能，堆栈未满且 EEPROM 写周期结束时，将调用 EEPROM 中断向量程序。当 EEPROM 中断响应，DEF 标志位会自动清零，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~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~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程序。当响应中断服务子程序时，相应的中断请求标志位 TB0F~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 f_{PSC0} 或 f_{PSC1} 来自内部时钟源 f_{SYS} 、 $f_{SYS}/4$ 或 f_{SUB} 。 f_{PSC0} 或 f_{PSC1} 输入时钟首先经过分频器，分频率由程序设置 TB0C 或 TB1C 寄存器相关位获取合适的分频值以提供更长的时基中断周期。相应的控制时基中断周期的时钟源可分别通过 PSC0R 和 PSC1R 寄存器的 CLKSEL0[1:0] 和 CLKSEL1[1:0] 位选择。



时基中断

- PSCnR 寄存器 (n=0~1)

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	CLKSELn1	CLKSELn0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义, 读为“0”

Bit 1~0 **CLKSELn1-CLKSELn0:** 预分频器 n 时钟源选择

00: f_{SYS}

01: f_{SYS}/4

1X: f_{SUB}

- TBnC 寄存器 (n=0~1)

Bit	7	6	5	4	3	2	1	0
Name	TBnON	—	—	—	—	TBn2	TBn1	TBn0
R/W	R/W	—	—	—	—	R/W	R/W	R/W
POR	0	—	—	—	—	0	0	0

Bit 7 **TBnON:** 时基 n 控制位

0: 除能

1: 使能

Bit 6~3 未定义, 读为“0”

Bit 2~0 **TBn2~TBn0:** 选择时基 n 溢出周期

000: 2⁸/f_{PSC}

001: 2⁹/f_{PSC}

010: 2¹⁰/f_{PSC}

011: 2¹¹/f_{PSC}

100: 2¹²/f_{PSC}

101: 2¹³/f_{PSC}

110: 2¹⁴/f_{PSC}

111: 2¹⁵/f_{PSC}

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。若要除能中断唤醒功能, 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。

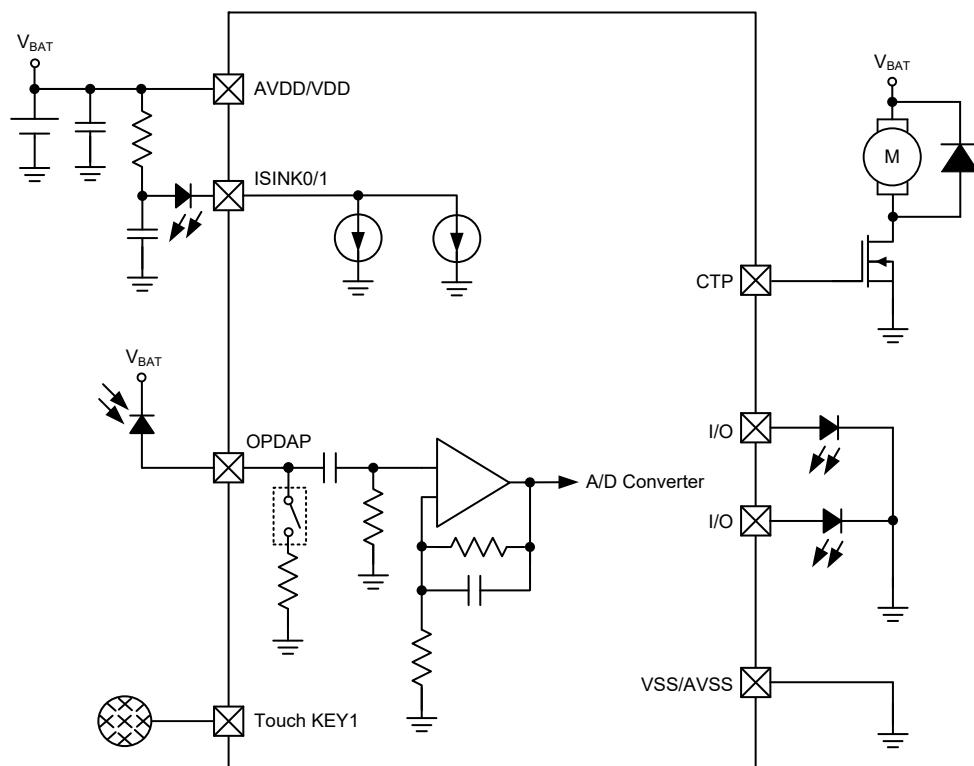
建议在中断服务子程序中不要使用“CALL 子程序”指令。中断通常发生在不可预料的情况或是需要立刻执行的某些应用。假如只剩下一层堆栈且没有控制好中断, 当“CALL 子程序”在中断服务子程序中执行时, 将破坏原来的控制序列。

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

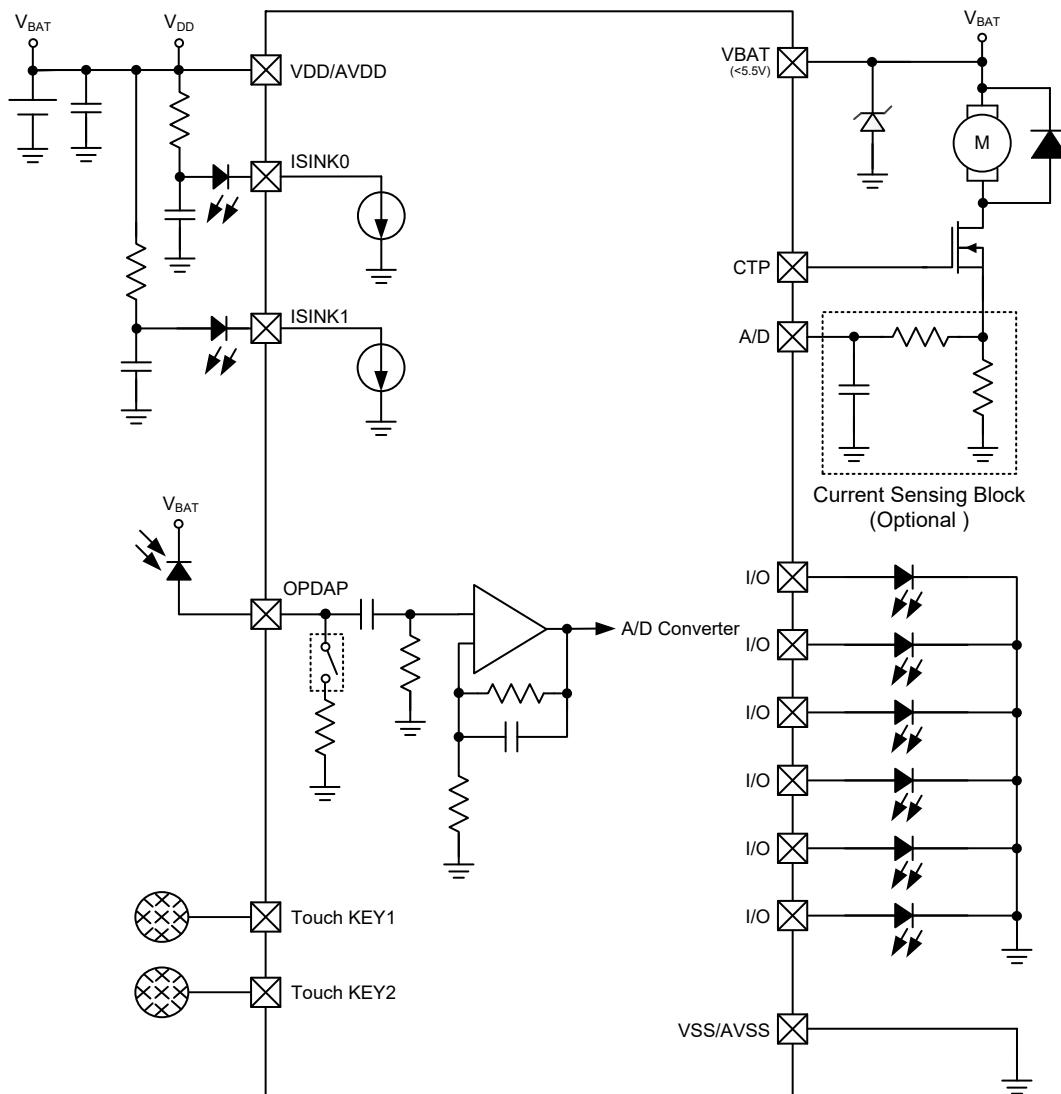
当进入中断服务程序，系统仅将程序计数器的内容压入堆栈，如果中断服务程序会改变状态寄存器或其它的寄存器的内容而破坏控制流程，应事先将这些数据保存起来。若从中断子程序中返回可执行 RET 或 RETI 指令。除了能返回至主程序外，RETI 指令还能自动设置 EMI 位为高，允许进一步中断。RET 指令只能返回至主程序，清除 EMI 位，除能进一步中断。

应用电路

8-pin SOP



16-pin NSOP



指令集

简介

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

为了更加容易理解各种各样的指令码，接下来按功能分组介绍它们。

指令周期

大部分的操作均只需要一个指令周期来执行。分支、调用或查表则需要两个指令周期。一个指令周期相当于四个系统时钟周期，因此如果在 8MHz 的系统时钟振荡器下，大部分的操作将在 $0.5\mu\text{s}$ 中执行完成，而分支或调用操作则将在 $1\mu\text{s}$ 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出口的 8 位数据，处理这些数据，然后再输出正确的数据。这种读入 - 修改 - 写出的过程现在则被位运算指令所取代。

查表运算

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

其它运算

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

指令集概要

下表中说明了按功能分类的指令集，用户可以将该表作为基本的指令参考。

惯例

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

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

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

注: 1. 对跳转指令而言, 如果比较的结果牵涉到跳转即需 2 个周期, 如果没有发生跳转, 则只需一个周期。

2. 任何指令若要改变 PCL 的内容将需要 2 个周期来执行。

指令定义

ADC A, [m]

指令说明

Add Data Memory to ACC with Carry

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]

指令说明

Add ACC to Data Memory with Carry

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]

指令说明

Add Data Memory to ACC

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

指令说明

Add immediate data to ACC

功能表示

$ACC \leftarrow ACC + x$

影响标志位

OV、Z、AC、C

ADDM A, [m]

指令说明

Add ACC to Data Memory

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]

指令说明

Logical AND Data Memory to ACC

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 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 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 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 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 将指定数据存储器的第 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 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 影响标志位 $TO \& PDF \leftarrow 0$ 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	$[m] \leftarrow [\bar{m}]$
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	$ACC \leftarrow [\bar{m}]$
影响标志位	Z
 DAA [m]	 Decimal-Adjust ACC for addition with result in Data Memory
指令说明	将累加器中的内容转换为 BCD (二进制转成十进制) 码。如果低四位的值大于“9”或 AC=1，那么 BCD 调整就执行对原值加“6”，否则原值保持不变；如果高四位的值大于“9”或 C=1，那么 BCD 调整就执行对原值加“6”。BCD 转换实质上是根据累加器和标志位执行 00H, 06H, 60H 或 66H 的加法运算，结果存放到数据存储器。只有进位标志位 C 受影响，用来指示原始 BCD 的和是否大于 100，并可以进行双精度十进制数的加法运算。
功能表示	$[m] \leftarrow ACC + 00H$ 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	将指定数据存储器的内容减 1，把结果存放回累加器并保持指定数据存储器的内容不变。
功能表示	$ACC \leftarrow [m] - 1$
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	$TO \leftarrow 0$
影响标志位	$PDF \leftarrow 1$
	TO、PDF
INC [m]	Increment Data Memory
指令说明	将指定数据存储器的内容加 1。
功能表示	$[m] \leftarrow [m] + 1$
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	无操作
影响标志位	无
 OR A, [m]	 Logical OR Data Memory to ACC
指令说明	将累加器中的数据和指定的数据存储器内容逻辑或, 结果存放到累加器。
功能表示	$ACC \leftarrow ACC \text{ "OR" } [m]$
影响标志位	Z
 OR A, x	 Logical OR immediate data to ACC
指令说明	将累加器中的数据和立即数逻辑或, 结果存放到累加器。
功能表示	$ACC \leftarrow ACC \text{ "OR" } x$
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	$Program\ Counter \leftarrow Stack$
影响标志位	$ACC \leftarrow x$ 无

RETI	Return from interrupt
指令说明	将堆栈寄存器中的程序计数器值恢复且中断功能通过设置 EMI 位重新使能。EMI 是控制中断使能的主控制位。如果 在执行 RETI 指令之前还有中断未被响应，则这个中断将 在返回主程序之前被响应。
功能表示	Program Counter ← Stack
	EMI ← 1
影响标志位	无
 RL [m]	 Rotate Data Memory left
指令说明	将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位。
功能表示	$[m].(i+1) \leftarrow [m].i$ ($i=0\sim6$) $[m].0 \leftarrow [m].7$
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	$[m].i \leftarrow [m].(i+1)$ ($i=0 \sim 6$) $[m].7 \leftarrow [m].0$
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	$ACC.i \leftarrow [m].(i+1)$ ($i=0 \sim 6$) $ACC.7 \leftarrow [m].0$
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	$[m].i \leftarrow [m].(i+1)$ ($i=0 \sim 6$) $[m].7 \leftarrow C$ $C \leftarrow [m].0$
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	$ACC.i \leftarrow [m].(i+1)$ ($i=0 \sim 6$) $ACC.7 \leftarrow C$ $C \leftarrow [m].0$
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

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

SUB A, x	Subtract immediate Data from ACC
指令说明	将累加器的内容减去立即数，结果存放于累加器。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
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	指定数据存储器的内容会先被读出，后又被重新写入指定数据存储器内。判断指定数据存储器的内容是否为 0，若为 0，则程序跳过下一条指令执行。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	如果 $[m]=0$ ，跳过下一条指令执行
影响标志位	无
SZA [m]	Skip if Data Memory is 0 with data movement to ACC
指令说明	将指定数据存储器内容复制到累加器，并判断指定数据存储器的内容是否为 0，若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	$ACC \leftarrow [m]$ ，如果 $[m]=0$ ，跳过下一条指令执行
影响标志位	无

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

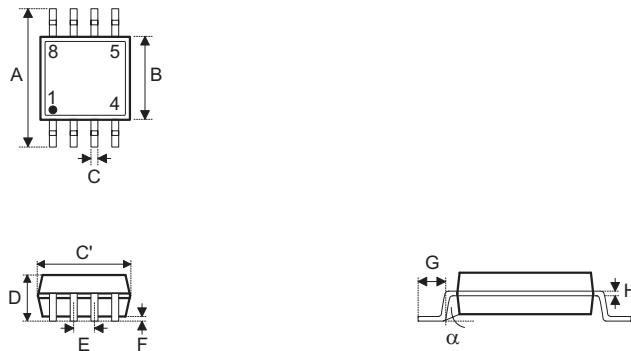
封装信息

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

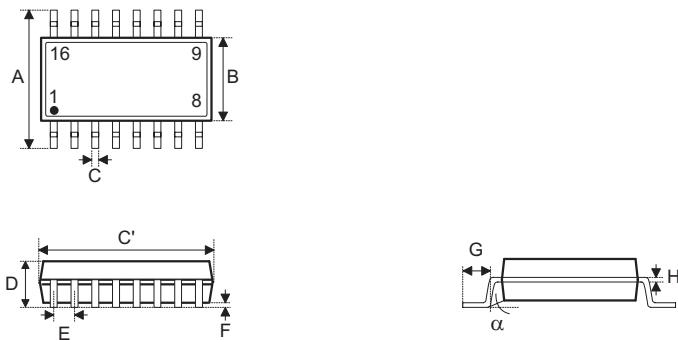
8-pin SOP (150mil) 外形尺寸



符号	尺寸 (单位: inch)		
	最小值	典型值	最大值
A		0.236 BSC	
B		0.154 BSC	
C	0.012	—	0.020
C'		0.193 BSC	
D	—	—	0.069
E		0.050 BSC	
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

符号	尺寸 (单位: mm)		
	最小值	典型值	最大值
A		6.00 BSC	
B		3.90 BSC	
C	0.31	—	0.51
C'		4.90 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 NSOP (150mil) 外形尺寸



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

符号	尺寸 (单位: mm)		
	最小值	典型值	最大值
A			6.00 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°

Copyright® 2023 by HOLTEK SEMICONDUCTOR INC. All Rights Reserved.

本文件出版时 HOLTEK 已针对所载信息为合理注意，但不保证信息准确无误。文中提到的信息仅是提供作为参考，且可能被更新取代。HOLTEK 不担保任何明示、默示或法定的，包括但不限于适合商品化、令人满意的质量、规格、特性、功能与特定用途、不侵害第三方权利等保证责任。HOLTEK 就文中提到的信息及该信息之应用，不承担任何法律责任。此外，HOLTEK 并不推荐将 HOLTEK 的产品使用在会由于故障或其他原因而可能会对人身安全造成危害的地方。HOLTEK 特此声明，不授权将产品使用于救生、维生或安全关键零部件。在救生 / 维生或安全应用中使用 HOLTEK 产品的风险完全由买方承担，如因该等使用导致 HOLTEK 遭受损害、索赔、诉讼或产生费用，买方同意出面进行辩护、赔偿并使 HOLTEK 免受损害。HOLTEK (及其授权方，如适用) 拥有本文件所提供信息 (包括但不限于内容、数据、示例、材料、图形、商标) 的知识产权，且该信息受著作权法和其他知识产权法的保护。HOLTEK 在此并未明示或暗示授予任何知识产权。HOLTEK 拥有不事先通知而修改本文件所载信息的权利。如欲取得最新的信息，请与我们联系。