



移动电源 Flash 单片机
BP45FH6N

版本 : V1.10 日期 : 2019-11-12

www.holtek.com

目录

特性	7
CPU 特性	7
周边特性	7
概述	8
方框图	9
引脚图	10
引脚说明	10
极限参数	13
直流电气特性	14
工作电压特性	14
待机电流特性	14
工作电流特性	15
交流电气特性	15
内部高速振荡器 – HIRC – 频率精确度	15
内部低速振荡器电气特性 – LIRC	16
工作频率电气特性曲线图	16
系统上电时间电气特性	16
输入 / 输出口电气特性	17
输入 / 输出口 (无多电源功能) 直流特性	17
输入 / 输出口 (带多电源功能) 直流特性	17
存储器电气特性	18
LVD/LVR 电气特性	19
A/D 转换器电气特性	19
参考电压电气特性	20
可编程增益放大器电气特性	20
过电流保护电气特性	21
过电压 / 欠电压保护电气特性	22
延迟锁相环电气特性	22
USB 自动检测电气特性	23
LDO 稳压器电气特性	24
高压输出电气特性	25
上电复位特性	26
系统结构	26
时序和流水线结构	26
程序计数器	27
堆栈	27
算术逻辑单元 – ALU	28

Flash 程序存储器	29
结构	29
特殊向量	29
查表	29
查表范例	30
在线烧录 – ICP	31
片上调试 – OCDS	31
数据存储器	32
结构	32
数据存储器寻址	32
通用数据存储器	33
特殊功能数据存储器	33
特殊功能寄存器	35
间接寻址寄存器 – IAR0, IAR1, IAR2	35
存储器指针 – MP0, MP1L/MP1H, MP2L/MP2H	35
累加器 – ACC	36
程序计数器低字节寄存器 – PCL	37
表格寄存器 – TBLP, TBHP, TBLH	37
状态寄存器 – STATUS	37
EEPROM 数据存储器	39
EEPROM 数据存储器结构	39
EEPROM 寄存器	39
从 EEPROM 中读取数据	40
写数据到 EEPROM	40
写保护	41
EEPROM 中断	41
编程注意事项	41
振荡器	42
振荡器概述	42
系统时钟配置	42
内部 RC 振荡器 – HIRC	43
内部 32kHz 振荡器 – LIRC	43
工作模式和系统时钟	43
系统时钟	43
系统工作模式	44
控制寄存器	45
工作模式切换	46
待机电流的注意事项	49
唤醒	49
看门狗定时器	50
看门狗定时器时钟源	50
看门狗定时器控制寄存器	50
看门狗定时器操作	51

复位和初始化	52
复位功能	52
复位初始状态	54
输入 / 输出端口	58
上拉电阻	59
PA 口唤醒	59
输入 / 输出端口控制寄存器	60
输入 / 输出端口电源控制	60
引脚共用功能	61
输入 / 输出引脚结构	64
编程注意事项	64
低压差稳压器 – LDO	65
高压输出端口	65
高压输出功能描述	66
高压输出寄存器描述	66
定时器模块 – TM	68
简介	68
TM 操作	68
TM 时钟源	68
TM 中断	68
TM 外部引脚	69
编程注意事项	69
标准型 TM – STM	71
标准型 TM 操作	71
标准型 TM 寄存器介绍	71
标准型 TM 工作模式	75
周期型 TM – PTM	84
周期型 TM 操作	84
周期型 TM 寄存器介绍	84
周期型 TM 工作模式	88
A/D 转换器	96
A/D 转换器简介	96
A/D 转换器寄存器介绍	97
A/D 转换器操作	100
A/D 转换器参考电压	101
A/D 转换器输入信号	101
转换率和时序图	102
A/D 转换步骤	102
编程注意事项	103
A/D 转换功能	103
A/D 转换程序范例	104
带延迟锁相环 / 死区时间的可自动调节的高精度 PWM	106
功能描述	106
H.R PWM 寄存器介绍	106

PWM 发生器.....	113
延迟锁相环	113
自动调节电路	115
死区时间插入	115
保护与反相控制	116
编程注意事项	117
过电流保护 – OCP	117
OCP 操作	117
OCP 寄存器介绍	118
输入电压范围	122
OCPn 运算放大器和比较器失调校准	122
过电压 / 欠电压保护 – OUV	123
OUVP 电路操作	123
OUVPn 寄存器介绍	124
OVPn 和 UVPn 比较器失调校准	128
USB 自动检测.....	130
D1+/D1- 和 D2+/D2- 自动检测	130
USB 自动检测寄存器	131
低电压检测 – LVD	133
LVD 寄存器	133
LVD 操作	134
中断	134
中断寄存器	134
中断操作	139
外部中断	140
过电流保护中断	141
过电压保护中断	141
欠电压保护中断	141
多功能中断	141
定时器模块中断	141
EEPROM 中断	142
A/D 转换器中断	142
时基中断	142
LVD 中断	143
中断唤醒功能	143
编程注意事项	143
配置选项	144
指令集	145
简介	145
指令周期	145
数据的传送	145
算术运算	145
逻辑和移位运算	145
分支和控制转换	146

位运算	146
查表运算	146
其它运算	146
指令集概要	147
惯例	147
扩展指令集	150
指令定义	152
扩展指令定义	164
封装信息	174
SAW Type 46-pin QFN (6.5mm×4.5mm) 外形尺寸	175

特性

CPU 特性

- 工作电压：
 - ◆ $f_{SYS}=8\text{MHz}$: 2.2V~5.5V
 - ◆ $f_{SYS}=12\text{MHz}$: 2.7V~5.5V
 - ◆ $f_{SYS}=16\text{MHz}$: 3.3V~5.5V
- $V_{DD}=5\text{V}$, 系统时钟为 16MHz 时, 指令周期为 0.25 μs
- 提供暂停和唤醒功能, 以降低功耗
- 振荡器类型:
 - ◆ 内部高速 8/12/16MHz RC – HIRC
 - ◆ 内部低速 32kHz RC – LIRC
- 多种工作模式: 快速、低速、空闲和休眠模式
- 完全集成内部振荡器无需外接元器件
- 所有指令都可在 1~3 个指令周期内完成
- 查表指令
- 115 条指令
- 8 层堆栈
- 位操作指令

周边特性

- Flash 程序存储器: 6K \times 16
- 数据存储器: 256 \times 8
- True EEPROM 存储器: 64 \times 8
- 看门狗定时器
- 28 个双向输入 / 输出口
- 8 个带短路保护功能的高压输出引脚
- 3 个引脚与外部中断口共用
- 多个定时器模块用于时间测量、捕捉输入、比较匹配输出、PWM 输出及单脉冲输出功能
 - ◆ 一个标准型 16-bit 定时器模块 – STM
 - ◆ 一个周期型 10-bit 定时器模块 – PTM
- 带延迟锁相环 / 死区时间功能的可自动调节的高精度 PWM
- 2 个可产生中断的过电流保护功能模块 – OCP
- 2 组可产生中断的过电压 / 欠电压保护功能模块 – OUVP
- USB 自动检测功能
- 双时基功能用以产生固定的中断信号
- 多通道 12-bit 分辨率 A/D 转换器, 具有可编程参考电压 V_R
- 低电压复位功能 – LVR
- 低电压检测功能 – LVD

- 内置低压差稳压器 – LDO
- 封装类型：46-pin QFN

概述

该单片机是针对移动电源应用开发的 8 位高性能精简指令集的 A/D Flash 单片机。该单片机具有一系列功能和特性，其 Flash 存储器可多次编程的特性给用户提供了极大的方便。存储器方面，还包含了一个 RAM 数据存储器和一个可用于存储序号、校准数据等非易失性数据的 True EEPROM 存储器。

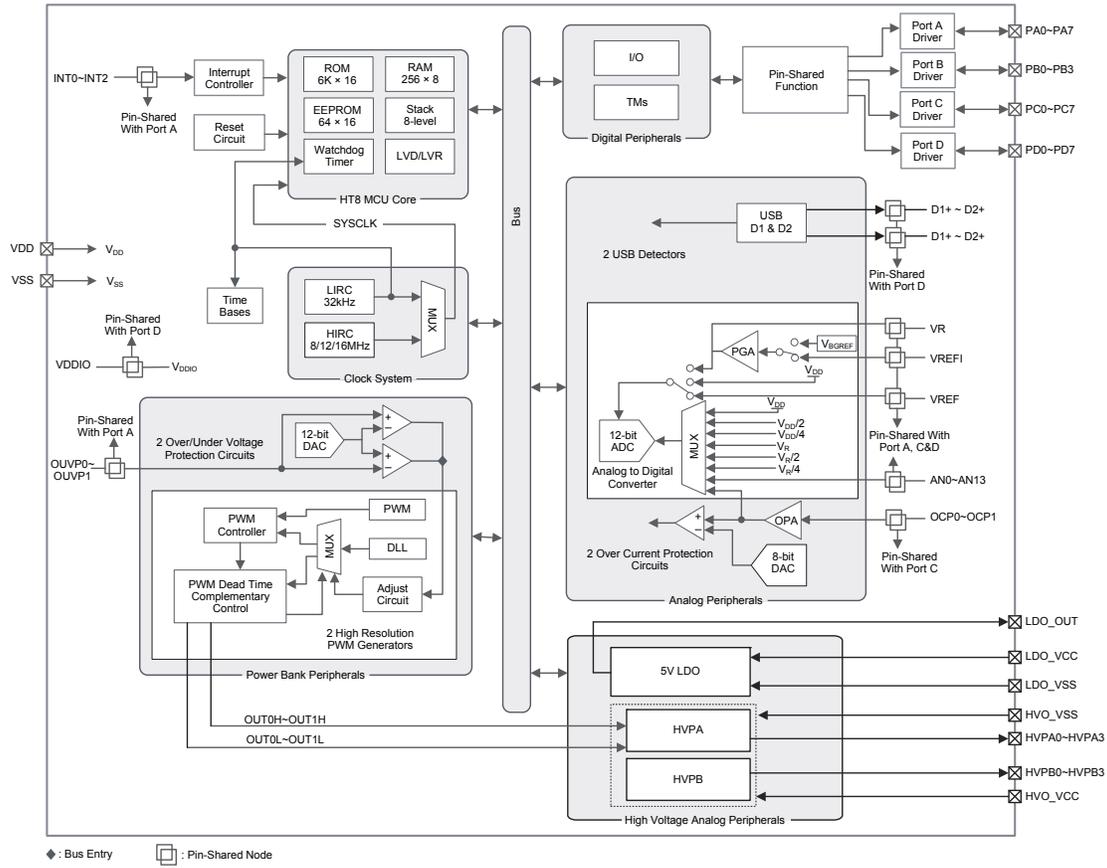
在模拟特性方面，该单片机包含一个多通道 12-bit A/D 转换器、一个用于稳压的 5V 低压差稳压器 (LDO)、两个过电流保护功能模块，两组过电压 / 欠电压保护功能模块，带延迟锁相环 / 死区时间功能的可自动调节的高精度 PWM 和 USB 设备自动检测功能。另外还带有多个使用灵活的定时器模块，可提供定时功能、脉冲产生功能及 PWM 产生功能。内部看门狗定时器、低电压复位和低电压检测等内部保护特性，外加优秀的抗干扰和 ESD 保护性能，确保单片机在恶劣的电磁干扰环境下可靠地运行。

该单片机提供了完全集成的内部高速和低速振荡器，无需外接元器件。其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。

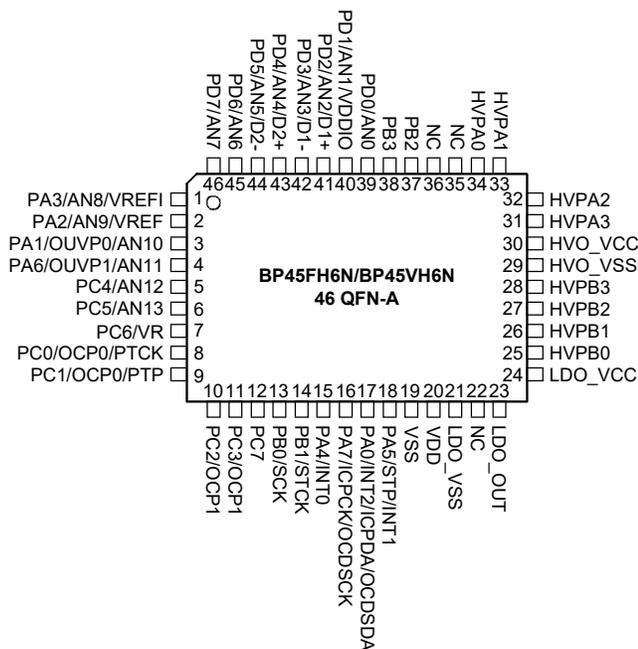
针对高压和大电流应用，该单片机内建了高压输出功能。外加时基功能、I/O 使用灵活等其它特性，使该单片机可以很好应用于各种移动电源产品应用中。

针对移动电源应用，该单片机内部集成多个功能电路，如过电压 / 欠电压保护电路、过电流保护电路及自动检测功能，确保仅需极少的外接元器件即可实现移动电源应用，从而大大减少元器件数量及电路板面积。

方框图



引脚图



- 注：1. 若共用脚同时有多种输出，所需的引脚功能可通过对应的引脚共用功能选择寄存器设置。
2. OCDSCK 和 OCSDA 引脚只存在于 BP45FH6N 的 OCDS EV 芯片，即 BP45VH6N。

引脚说明

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

引脚名称	功能	OPT	I/T	O/T	说明
PA0/INT2/ ICPDA/ OCSDA	PA0	PAWU PAPU	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻和唤醒
	INT2	INTEG INTC0	ST	—	外部中断 2 输入
	ICPDA	—	ST	CMOS	ICP 数据 / 地址
	OCSDA	—	ST	CMOS	OCDS 数据 / 地址 – 仅用于 EV 芯片
PA1/OUVP0/ AN10	PA1	PAWU PAPU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻和唤醒
	OUVP0	PAS0	AN	—	OVP/UVP 0 输入
	AN10	PAS0	AN	—	A/D 转换器外部信号输入通道 10

引脚名称	功能	OPT	I/T	O/T	说明
PA2/AN9/ VREF	PA2	PAWU PAPU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻和唤醒
	AN9	PAS0	AN	—	A/D 转换器外部信号输入通道 9
	VREF	PAS0	AN	—	A/D 转换器和 OVPn/OUVPn D/A 转换器外部参考输入
PA3/AN8/ VREFI	PA3	PAWU PAPU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻和唤醒
	AN8	PAS0	AN	—	A/D 转换器外部信号输入通道 8
	VREFI	PAS0	AN	—	A/D 转换器外部参考电压输入
PA4/INT0	PA4	PAWU PAPU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻和唤醒
	INT0	PAS1 INTEG INTC0	ST	—	外部中断 0 输入
PA5/STP/INT1	PA5	PAWU PAPU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻和唤醒
	STP	PAS1	ST	CMOS	STM 输出或 STM 捕捉输入
	INT1	PAS1 INTEG INTC0	ST	—	外部中断 1 输入
PA6/OUVP1/ AN11	PA6	PAWU PAPU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻和唤醒
	OUVP1	PAS1	AN	—	OVP/UVP 1 输入
	AN11	PAS1	AN	—	A/D 转换器外部信号输入通道 11
PA7/ICPCK/ OCDSCK	PA7	PAWU PAPU	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻和唤醒
	ICPCK	—	ST	—	ICP 时钟输入
	OCDSCK	—	ST	—	OCDS 时钟 – 仅用于 EV 芯片
PB0	PB0	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
PB1/STCK	PB1	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	STCK	—	ST	—	STM 输入
PB2~PB3	PB2~PB3	PBPU	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
PC0/OCP0/ PTCK	PC0	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	OCP0	PCS0	AN	—	OCP0 输入
	PTCK	PCS0	ST	—	PTM 输入

引脚名称	功能	OPT	I/T	O/T	说明
PC1/OCP0/PTP	PC1	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	OCP0	PCS0	AN	—	OCP0 输入
	PTP	PCS0	ST	CMOS	PTM 输出或 PTM 捕捉输入
PC2/OCP1	PC2	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	OCP1	PCS0	AN	—	OCP1 输入
PC3/OCP1	PC3	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	OCP1	PCS0	AN	—	OCP1 输入
PC4/AN12	PC4	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	AN12	PCS1	AN	—	A/D 转换器外部信号输入通道 12
PC5/AN13	PC5	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	AN13	PCS1	AN	—	A/D 转换器外部信号输入通道 13
PC6/VR	PC6	PCPU PCS1	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	VR	PCS1	—	AN	A/D 转换器外部参考电压输出
PC7	PC7	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
PD0/AN0	PD0	PDP PDS0	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	AN0	PDS0	AN	—	A/D 转换器外部信号输入通道 0
PD1/AN1/ VDDIO	PD1	PDP PDS0	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	AN1	PDS0	AN	—	A/D 转换器外部信号输入通道 1
	VDDIO	PDS0	PWR	—	PD2~PD5 I/O 引脚电源
PD2/AN2/D1+	PD2	PDP PDS0	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	AN2	PDS0	AN	—	A/D 转换器外部信号输入通道 2
	D1+	PDS0	—	AN	USB DAC0 输出引脚
PD3/AN3/D1-	PD3	PDP PDS0	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	AN3	PDS0	AN	—	A/D 转换器外部信号输入通道 3
	D1-	PDS0	—	AN	USB DAC1 输出引脚
PD4/AN4/D2+	PD4	PDP PDS1	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	AN4	PDS1	AN	—	A/D 转换器外部信号输入通道 4
	D2+	PDS1	—	AN	USB DAC2 输出引脚
PD5/AN5/D2-	PD5	PDP PDS1	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	AN5	PDS1	AN	—	A/D 转换器外部信号输入通道 5
	D2-	PDS1	—	AN	USB DAC3 输出引脚

引脚名称	功能	OPT	I/T	O/T	说明
PD6/AN6	PD6	PDPUPDS1	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	AN6	PDS1	AN	—	A/D 转换器外部信号输入通道 6
PD7/AN7	PD7	PDPUPDS1	ST	CMOS	通用 I/O 口，可通过寄存器使能上拉电阻
	AN7	PDS1	AN	—	A/D 转换器外部信号输入通道 7
HVPA0~HVPA3	HVPA0~HVPA3	HVPAEN	—	AN	高压输出
HVPB0~HVPB3	HVPB0~HVPB3	HVPBEN	—	AN	高压输出
VDD	VDD	—	PWR	—	数字正电源
VSS	VSS	—	PWR	—	数字负电源，接地
HVO_VCC	HVO_VCC	—	PWR	—	HVO 和电平转换器高压电源
HVO_VSS	HVO_VSS	—	PWR	—	高压负电源，接地
LDO_OUT	LDO_OUT	—	—	PWR	LDO 输出电压
LDO_VCC	LDO_VCC	—	PWR	—	LDO 高压电源
LDO_VSS	LDO_VSS	—	PWR	—	LDO 负电源，接地

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

极限参数

电源供应电压 (V_{LDO_VCC} 、 V_{HVO_VCC})	$V_{SS}-0.3V \sim 13.5V$
电源供应电压 (V_{DD})	$V_{SS}-0.3V \sim 6.0V$
高压输入电压	$V_{SS}-0.3V \sim V_{HVO_VCC}+0.3V$
输入电压	$V_{SS}-0.3V \sim V_{DD}+0.3V$
储存温度	$-50^{\circ}C \sim 125^{\circ}C$
工作温度	$-40^{\circ}C \sim 85^{\circ}C$
I_{OH} 总电流	-120mA
I_{OL} 总电流	120mA
总功耗	600mW

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。

工作电压特性

Ta=-40°C~85°C

符号	参数	测试条件	最小	典型	最大	单位
V _{DD}	工作电压 - HIRC	f _{SYS} =f _{HIRC} =8MHz	2.2	—	5.5	V
		f _{SYS} =f _{HIRC} =12MHz	2.7	—	5.5	
		f _{SYS} =f _{HIRC} =16MHz	3.3	—	5.5	
	工作电压 - LIRC	f _{SYS} =f _{LIRC} =32kHz	V _{LVR}	—	5.5	V

待机电流特性

Ta=-40°C~85°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	3V	WDT on	—	1.5	4.0	TBD	μA
		5V		—	3.0	5.0	TBD	
	空闲模式 0	3V	f _{SUB} on	—	3.0	5.0	TBD	μA
		5V		—	5.0	10.0	TBD	
	空闲模式 1 - HIRC	2.2V	f _{SUB} on, f _{SYS} =8MHz	—	TBD	TBD	TBD	μA
		3V		—	360	500	TBD	
		5V		—	600	800	TBD	
		2.7V	f _{SUB} on, f _{SYS} =12MHz	—	TBD	TBD	TBD	μA
		3V		—	540	750	TBD	
		5V		—	800	1200	TBD	
	3.3V	f _{SUB} on, f _{SYS} =16MHz	—	TBD	TBD	TBD	mA	
	5V		—	2.0	2.4	TBD		

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 指令执行后测得，因此 HALT 后停止执行所有指令。

工作电流特性

Ta=-40°C~85°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}	工作电流 – LIRC	3V	f _{SYS} =32kHz	—	10	20	μA
		5V		—	30	50	
	工作电流 – HIRC	2.2V	f _{SYS} =8MHz	—	TBD	TBD	mA
		3V		—	0.8	1.2	
		5V		—	1.6	2.4	
		3V	f _{SYS} =12MHz	—	1.2	1.8	mA
		5V		—	2.4	3.6	
		3.3V	f _{SYS} =16MHz	—	TBD	TBD	mA
	5V	—		3.6	4.8		

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. 无直流电流通路。
4. 所有的工作电流值都通过一个连续的 NOP 指令循环程序测得。

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。

内部高速振荡器 – HIRC – 频率精确度

程序烧录时，烧录器会调整 HIRC 振荡器使其工作在用户选择的 HIRC 频率和工作电压 (3V 或 5V) 条件下。

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	-2%	8	+2%	
		2.2V~5.5V	25°C	-2.5%	8	+2.5%	
			-40°C ~ 85°C	-3%	8	+3%	
	通过烧录器调整后的 12MHz HIRC 频率	3V/5V	25°C	-1%	12	+1%	MHz
			-40°C ~ 85°C	-2%	12	+2%	
		2.7V~5.5V	25°C	-2.5%	12	+2.5%	
			-40°C ~ 85°C	-3%	12	+3%	
	通过烧录器调整后的 16MHz HIRC 频率	5V	25°C	-1%	16	+1%	MHz
			-40°C ~ 85°C	-2%	16	+2%	
		3.3V~5.5V	25°C	-2.5%	16	+2.5%	
			-40°C ~ 85°C	-3%	16	+3%	

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。

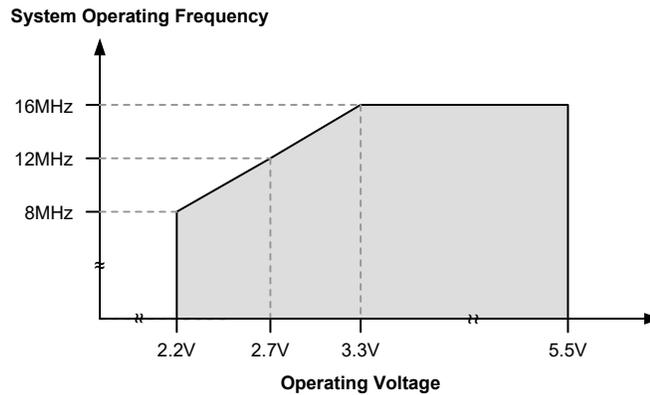
2. 3V/5V 表格列下面提供的是全压条件下的参数值。当应用电压范围是 2.2V~3.6V 时，建议烧录器电压固定在 3V；当应用电压范围是 3.3V~5.5V 时，建议烧录器电压固定在 5V。

3. 表格中提供的最小和最大误差值仅在对应的烧录器调整频率下有效。当烧录器已将 HIRC 调整为某一固定频率, 此后再通过程序中振荡器控制位将其频率改为其它值时, 频率误差范围将增加到 $\pm 20\%$ 。

内部低速振荡器电气特性 – LIRC

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	温度				
f _{LIRC}	振荡器频率	5V	25°C	-10%	32	+10%	kHz
		5V±0.5V	-40°C ~ 85°C	-40%	32	+40%	
		V _{LVR} ~5.5V	-40°C ~ 85°C	-50%	32	+60%	

工作频率电气特性曲线图



系统上电时间电气特性

T_a=-40°C ~ 85°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
t _{SSST}	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
		—	f _{sys} =f _{sub} =f _{LIRC}	2	—	—	t _{sub}
系统启动时间 (从 WDT 溢出硬件冷复位的 状态下唤醒)	—	—	0	—	—	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)	—	—	8.3	16.7	33.3	ms
	系统复位延迟时间 (LVRC/WDT 软件复位)	—	—				
	系统复位延迟时间 (WDT 溢出复位)	—	—	8.3	16.7	33.3	ms

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
t _{SRESET}	软件复位最小脉宽	—	—	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. 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

输入 / 输出口 (无多电源功能) 直流特性

T_a=25°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 端口低电平输入电压 (除 PD2~PD5 引脚外)	5V	—	0	—	1.5	V
		—	—	0	—	0.2V _{DD}	
V _{IH}	I/O 端口高电平输入电压 (除 PD2~PD5 引脚外)	5V	—	3.5	—	5.0	V
		—	—	0.8V _{DD}	—	V _{DD}	
I _{OL}	I/O 端口灌电流 (除 PD2~PD5 引脚外)	3V	V _{OL} =0.1V _{DD}	16	32	—	mA
		5V		32	65	—	
I _{OH}	I/O 端口源电流 (除 PD2~PD5 引脚外)	3V	V _{OH} =0.9V _{DD}	-4	-8	—	mA
		5V		-8	-16	—	
R _{PH}	I/O 端口上拉电阻 ^注 (除 PD2~PD5 引脚外)	3V	—	20	60	100	kΩ
		5V	—	10	30	50	
t _{TCK}	PTCK、STCK 时钟输入 最小脉宽	—	—	0.3	—	—	μs
t _{INT}	外部中断最小脉宽	—	—	10	—	—	μs

输入 / 输出口 (带多电源功能) 直流特性

T_a=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DD}	V _{DD} 引脚电源 (PD2~PD5 引脚)	—	—	2.2	5.0	5.5	V
V _{DDIO}	V _{DDIO} 引脚电源 (PD2~PD5 引脚)	—	—	1.8	—	V _{DD}	V
V _{IL}	低电平输入电压 (PD2~PD5 引脚)	5V	—	0	—	1.5	V
		—	引脚电源 = V _{DD} 或 V _{DDIO}	0	—	0.2 (V _{DD} /V _{DDIO})	
V _{IH}	高电平输入电压 (PD2~PD5 引脚)	5V	—	3.5	—	5.0	V
		—	引脚电源 = V _{DD} 或 V _{DDIO}	0.8V _{DD}	—	V _{DD} /V _{DDIO}	

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
I _{OL}	引脚灌电流 (PD2~PD5 引脚)	3V	V _{OL} =0.1(V _{DD} /V _{DDIO}), V _{DDIO} =V _{DD}	16	32	—	mA
		5V	V _{OL} =0.1(V _{DD} /V _{DDIO}), V _{DDIO} =V _{DD}	32	65	—	
			V _{OL} =0.1(V _{DD} /V _{DDIO}), V _{DDIO} =3V	20	40	—	
I _{OH}	引脚源电流 (PD2~PD5 引脚)	3V	V _{OH} =0.9(V _{DD} /V _{DDIO}), V _{DDIO} =V _{DD}	-4	-8	—	mA
		5V	V _{OH} =0.9(V _{DD} /V _{DDIO}), V _{DDIO} =V _{DD}	-8	-16	—	
			V _{OH} =0.9(V _{DD} /V _{DDIO}), V _{DDIO} =3V	-2.5	-5	—	
R _{PH}	上拉电阻 ^注 (PD2~PD5 引脚)	3V	V _{DDIO} =V _{DD} , LVPU=0, PDPU[5:2]=1111B	20	60	100	kΩ
		5V	V _{DDIO} =V _{DD} , LVPU=0, PDPU[5:2]=1111B	10	30	50	
		3V	V _{DDIO} =V _{DD} , LVPU=1, PDPU[5:2]=1111B	6.67	15	23	
		5V	V _{DDIO} =V _{DD} , LVPU=1, PDPU[5:2]=1111B	3.5	7.5	12	
I _{LEAK}	输入漏电流 (PD2~PD5 引脚)	5V	V _{IN} =V _{DD} 或 V _{IN} =V _{DDIO} 或 V _{IN} =V _{SS}	—	—	±1	μA

注: R_{PH} 内部上拉电阻值的计算方法是: 将其接地并使能输入引脚的上拉电阻选项, 然后在特定电源电压下测量引脚电流, 在此基础上测量上拉电阻上的分压从而得到此上拉电阻值。

存储器电气特性

T_a=-40°C~85°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{RW}	读 / 写工作电压	—	—	V _{DDmin}	—	V _{DDmax}	V
Flash 程序存储器 / 数据 EEPROM 存储器							
t _{DEW}	擦除 / 写时间 – Flash 程序存储器	—	—	—	2	3	ms
	擦除 / 写时间 – 数据 EEPROM 存储器	—	—	—	4	6	ms
I _{DDPGM}	V _{DD} 电压下烧录 / 擦除电流	—	—	—	—	5.0	mA
E _P	电容耐久性 – Flash 程序存储器	—	—	10K	—	—	E/W
	电容耐久性 – 数据 EEPROM 存储器	—	—	100K	—	—	
t _{RETD}	ROM 数据保存时间	—	T _a =25°C	—	40	—	Year
RAM 数据存储器							
V _{DR}	RAM 数据保存电压	—	单片机处于休眠模式	1.0	—	—	V

LVD/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 _{LVR}	低电压复位电压	—	LVR 使能	-5%	2.55	+5%	V
V _{LVD}	低电压检测电压	—	LVD 使能, V _{LVD} =2.7V	-5%	2.7	+5%	V
		—	LVD 使能, V _{LVD} =3.0V		3.0		
		—	LVD 使能, V _{LVD} =3.3V		3.3		
		—	LVD 使能, V _{LVD} =3.6V		3.6		
		—	LVD 使能, V _{LVD} =4.0V		4.0		
I _{LVR}	LVR 使能额外电流	—	LVD 除能	—	—	10	μA
I _{LVD}	LVD 使能额外电流	—	LVR 除能	—	60	90	μA
t _{LVDS}	LVDO 稳定时间	—	LVR 使能, LVD off → on	—	—	15	μs
t _{LVR}	产生 LVR 复位的低电压最短保持时间	—	—	120	240	480	μs
t _{LVD}	产生 LVD 中断低电压最短保持时间	—	—	60	120	240	μs

A/D 转换器电气特性

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}	A/D 转换器工作电压	—	—	2.2	—	5.5	V
V _{ADI}	A/D 转换器输入电压	—	—	0	—	V _{REF}	V
V _{REF}	A/D 转换器参考电压	—	—	2	—	V _{DD}	V
DNL	非线性微分误差	3V	SAINS[3:0]=0000B, SAVRS[1:0]=00B, V _{REF} =V _{DD} , t _{ADCK} =0.5μs	-3	—	+3	LSB
		5V	V _{REF} =V _{DD} , t _{ADCK} =0.5μs				
		3V	SAINS[3:0]=0000B, SAVRS[1:0]=00B, V _{REF} =V _{DD} , t _{ADCK} =10μs				
		5V	V _{REF} =V _{DD} , t _{ADCK} =10μs				
INL	非线性积分误差	3V	SAINS[3:0]=0000B, SAVRS[1:0]=00B, V _{REF} =V _{DD} , t _{ADCK} =0.5μs	-4	—	+4	LSB
		5V	V _{REF} =V _{DD} , t _{ADCK} =0.5μs				
		3V	SAINS[3:0]=0000B, SAVRS[1:0]=00B, V _{REF} =V _{DD} , t _{ADCK} =10μs				
		5V	V _{REF} =V _{DD} , t _{ADCK} =10μs				
I _{ADC}	A/D 转换器使能的额外电流	3V	无负载, t _{ADCK} =0.5μs	—	0.2	0.4	mA
		5V		—	0.3	0.6	
t _{ADCK}	A/D 转换器时钟周期	—	—	0.5	—	10	μs
t _{ON2ST}	A/D 转换器 On-to-Start 时间	—	—	4	—	—	μs
t _{ADS}	A/D 采样时间	—	—	—	4	—	t _{ADCK}
t _{ADC}	A/D 转换时间 (包括 A/D 采样和保持时间)	—	—	—	16	—	t _{ADCK}

参考电压电气特性

Ta=25°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	—	—	2.2	—	5.5	V
V _{BGREF}	Bandgap 参考电压	—	—	-1%	1.2	+1%	V
		—	Ta=-40°C~85°C	-2%	1.2	+2%	
I _{BGREF}	工作电流	5.5V	Ta=-40°C~85°C	—	25	40	μA
PSRR	电源抑制比	—	V _{RIPPLE} =1V _{P-P} , f _{RIPPLE} =100Hz	75	—	—	dB
En	输出噪声	—	f=0.1Hz ~ 10Hz	—	300	—	μV _{RMS}
I _{DRV}	缓冲器驱动能力	—	ΔV _{BGREF} =-1%	1	—	—	mA
I _{SD}	关闭电流	—	V _{BGREN} =0	—	—	0.1	μA
t _{START}	启动时间	2.2V~5.5V	—	—	—	400	μs

- 注：1. 以上参数均能在无负载的条件下测得，除非另有说明。
 2. V_{DD} 引脚需连接一个 0.1μF 的陶瓷电容到地。
 3. V_{BGREF} 电压可作为 A/D 转换器内部信号输入。

可编程增益放大器电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
I _{PGA}	PGA 使能的额外电流	3V	无负载	—	390	550	μA
		5V		—	500	650	
V _{OR}	PGA 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}	PGA 固定电压范围	2.2V~5.5V	V _{RI} =V _{BGREF} , (PGAIS=1)	-1%	2	+1%	V
		3.2V~5.5V	V _{RI} =V _{BGREF} , (PGAIS=1)	-1%	3	+1%	
		4.2V~5.5V	V _{RI} =V _{BGREF} , (PGAIS=1)	-1%	4	+1%	
I _{VR}	V _R 最大输出电流	5V	V _{VR} =4V, ΔV _{VR} =-1%	1.2	2.4	—	mA
V _{IR}	PGA 输入电压范围	3V	Gain=1, PGAIS=0, 相对增益, 增益误差 < ±5%	V _{SS} +0.1	—	V _{DD} -1.4	V
		5V		V _{SS} +0.1	—	V _{DD} -1.4	

过电流保护电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
I _{OCP}	OCPn 工作电流	3V	OCPnEN[1:0]=01B, DAC V _{REF} =2.5V	—	—	1000	μA
		5V		—	730	1250	
V _{OS_CMP}	比较器输入失调电压	3V	未校准 (OCPnCOF[4:0]=10000B)	-15	—	15	mV
		5V		-15	—	15	
		3V	校准后	-4	—	4	mV
		5V		-4	—	4	
V _{HYS}	迟滞电压	3V	—	10	40	60	mV
		5V		10	40	60	
V _{CM_CMP}	比较器共模电压范围	3V	—	V _{SS}	—	V _{DD} -1.4	V
		5V		V _{SS}	—	V _{DD} -1.4	
V _{OS_OPA}	OPA 输入失调电压	3V	校准后 (OCPnOOF[5:0]=100000B)	-15	—	15	mV
		5V		-15	—	15	
		3V	未校准	-4	—	4	mV
		5V		-4	—	4	
V _{CM_OPA}	OPA 共模电压范围	3V	—	V _{SS}	—	V _{DD} -1.4	V
		5V		V _{SS}	—	V _{DD} -1.4	
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	
Ga	PGA 增益精度	3V	所有增益	-5	—	5	%
		5V		-5	—	5	
R _o	R2R 输出电阻	3V	—	—	10	—	kΩ
		5V		—	10	—	
V _{REF}	DAC 参考电压	3V/5V	OCPnVRS[1:0]=01B	2	—	V _{DD}	V
DNL	非线性微分误差	3V	DAC V _{REF} =V _{DD}	-1.5	—	+1.5	LSB
		5V		-1	—	+1	
INL	非线性积分误差	3V	DAC V _{REF} =V _{DD}	-2	—	+2	LSB
		5V		-1.5	—	+1.5	

过电压 / 欠电压保护电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
I _{OVP}	OVPn 工作电流	3V	UVPnEN=1, OVPnEN=1, DAC V _{REF} =2.5V	—	—	420	μA
		5V		—	300	500	
V _{OS}	输入失调电压	3V	校准后	-4	—	4	mV
		5V		-4	—	4	
V _{HYS}	迟滞电压	3V	—	10	20	30	mV
		5V	—	10	20	30	
V _{CM}	共模电压范围	3V	—	V _{SS}	—	V _{DD} -1.4	V
		5V	—	V _{SS}	—	V _{DD} -1.4	
R _O	R2R 输出电阻	5V	—	—	10	—	kΩ
DNL	非线性微分误差	3V	DAC V _{REF} =V _{DD}	-1.5	—	+1.5	LSB
		5V		-1	—	+1	
INL	非线性积分误差	3V	DAC V _{REF} =V _{DD}	-2.0	—	+2.0	LSB
		5V		-1.5	—	+1.5	
t _{RP}	响应时间	3V	带 100mV 过载(注)	—	200	400	ns
		5V		—	200	400	

 注：负载情况：C_{LOAD}=50pF。

延迟锁相环电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
I _{DLL}	DLL 工作电流	3V	DLEN=1	—	0.9	1.3	mA
		5V		—	1.5	2.1	
f _{DLL}	DLL 工作频率	2.2V~5.5V	f _{HIRC} =8MHz	-10%	8	+10%	MHz
		2.7V~5.5V	f _{HIRC} =12MHz	-10%	12	+10%	
		3.0V~5.5V	f _{HIRC} =16MHz	-10%	16	+10%	
t _{DLLS}	DLL 稳定时间	2.2V~5.5V	DLEN=0 → 1	—	20	30	μs
t _{PD}	传播延迟时间	2.2V~5.5V	DLEN=1	1.5	2.5	3.3	ns

USB 自动检测电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
V _{DAC}	DAC 工作电压	—	—	2.2	—	5.5	V
I _{DAC}	DAC 工作电流	3V	无负载	—	0.6	0.9	mA
		5V		—	1.0	1.5	
I _{DACSD}	DAC 关闭电流	—	无负载	—	—	0.1	μA
N _R	DAC 分辨精度	—	—	—	8	—	bits
DNL	DAC 非线性微分误差	—	无负载, DAC V _{REF} =V _{DD}	-1	—	+1	LSB
INL	DAC 非线性积分误差	—	无负载, DAC V _{REF} =V _{DD}	-2	—	+2	LSB
V _{DACO}	DAC 输出电压范围	—	Code=0000H	V _{SS}	—	V _{SS} +0.2	V
		—	Code=0FFFH	V _{REF} -0.2	—	V _{REF}	
V _{REF}	DAC 参考电压	—	—	2	—	V _{DD}	V
t _{ST}	建立时间	3V	C _{LOAD} =50pF	—	—	5	μs
		5V		—	—	5	
R _O	R2R 输出电阻	3V	—	—	3	—	kΩ
		5V		—	5	—	
OSRR	失调误差	3V	V _{REF} =V _{DD} =3V Data word=128	—	—	50	mV
		5V	V _{REF} =V _{DD} =5V Data word=128	—	—	80	
GERR	增益误差	3V	V _{REF} =V _{DD} =3V Data word=128	—	—	50	mV
		5V	V _{REF} =V _{DD} =5V Data word=128	—	—	80	
I _{DACOL}	输出灌电流	3V	Data word=0000H, V _{DACO} =0.1V _{REF}	20	—	—	mA
		5V	—	40	—	—	
I _{DACOH}	输出源电流	3V	Data word=0FFFH, V _{DACO} =0.9V _{REF}	20	—	—	mA
		5V	—	40	—	—	
I _{SC}	输出短路电流	3V	Data word=0FFFH	0.25	—	—	mA
		5V		0.4	—	—	
R _{ON}	D1+/D2+ 与 D1-/D2- 之间的模拟开关电阻	5V	—	—	20	35	Ω
R _{PL}	D1+, D2+, D1-, D2- 下拉电阻	5V	—	15	20	30	kΩ

符号	参数	测试条件		最小	典型	最大	单位
		V _{DD}	条件				
ERR	D1+, D1-, D2+, D2- 输出电压误差	5V	DAC V _{REF} =V _{DD} , ADUDAn[7:0]=10010100B, D1+, D1-, D2+, D2- 连接 150kΩ 电阻到地	2.57	2.70	2.84	V
		5V	DAC V _{REF} =V _{DD} , ADUDAn[7:0]=01101110B, D1+, D1-, D2+, D2- 连接 150kΩ 电阻到地	1.9	2.0	2.1	V

LDO 稳压器电气特性

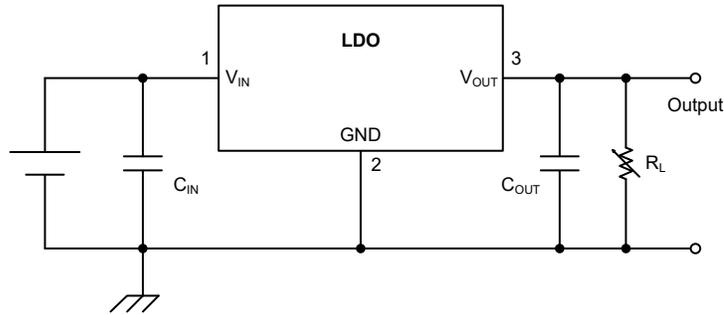
V_{IN}=V_{OUT}+1V, C_{LOAD}=10μF+0.1μF, T_a=-40°C~85°C, 除非另有说明

符号	参数	测试条件		最小	典型	最大	单位
		V _{IN}	条件				
V _{IN}	输入电压	—	—	V _{OUT} +0.1	6	13.5	V
V _{OUT}	输出电压	—	T _a =25°C I _{LOAD} =1mA, V _{OUT} =5.0V	-2%	5.0	2%	V
		—	I _{LOAD} =1mA, V _{OUT} =5.0V	-3%	5.0	3%	V
ΔV _{LOAD}	负载调整率 ⁽¹⁾	—	1mA≤I _{LOAD} ≤70mA	—	0.02	0.05	%/mA
V _{DROP}	电压差 ⁽²⁾	—	V _{OUT} =5V, ΔV _{OUT} =-2%, I _{LOAD} =50mA	—	150	300	mV
	旁路模式电压差	—	3V<V _{IN} <5.1V, I _{LOAD} =50mA	—	300	600	mV
I _{OUT}	输出电流	—	ΔV _{OUT} =-3%	70	—	—	mA
		—	V _{IN} =V _{OUT} +2V, ΔV _{OUT} =-5%	100	—	—	mA
I _Q	静态电流	12V	无负载	—	3	8	μA
ΔV _{LINE}	线性稳压率	—	V _{OUT} +1V≤V _{IN} ≤12V, I _{LOAD} =1mA	—	—	0.5	%/V
TC	温度系数	—	I _{LOAD} =10mA	—	±1.5	±2	mV/°C
ΔV _{OUT_RIPPLE}	输出电压纹波	6V	I _{LOAD} =10mA	—	—	10	mV
RR	纹波抑制比 ⁽³⁾	—	V _{IN} =10V _{DC} +2V _{P-P(AC)} , I _{LOAD} ≤50mA, f=120Hz	35	—	—	dB
I _{LIMIT}	电流限制值	6V	ΔV _{OUT} =-10%	150	—	—	mA
t _{START}	LDO 启动时间	6V	I _{LOAD} =1mA, V _{OUT} =5V±5%	—	—	10	ms

注：1. 负载调整在恒结温条件下使用一个低 ON 时间的脉冲测得，测量时确保达到最大的功耗。功耗由输入 / 输出差分电压和输出电流决定。确保的最大功耗不允许超出全输入 / 输出范围。任何环境温度下的最大可允许功耗为 P_D=(T_{J(MAX)}-T_a)/θ_{JA}。

2. 压降是指将输出电压维持在 2% 以内所需的输入电压 V_{IN} 与输出电压 V_{OUT} 的差值。

3. 纹波抑制比测量电路如下所示，RR=20×log(ΔV_{IN}/ΔV_{OUT})，C_{IN}=10μF/C_{OUT}=10μF+0.1μF。



4. 出于稳定性的考虑，LDO 负载电容选择的应用信息如下：

输出电容建议值

$T_a=25^{\circ}\text{C}$ ，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				
C_{LOAD}	输出负载电容	—	电容	4.7	10	—	μF

与大多数稳压器一样，LDO 需要在 V_{OUT} 和地 GND 之间外接一个电容以稳定稳压器。如果 ESR 小于 10Ω ，则电容值只需在 $4.7\mu\text{F}$ 以上，选择满足上述要求的铝电解电容即可。

为了获得很好的负载瞬态响应，可以在 V_{OUT} 上使用一个 $10\mu\text{F}$ 的 C_{LOAD} 和额外 $0.1\mu\text{F}$ 电容的组合。注意， $0.1\mu\text{F}$ 电容需要一直接在 V_{OUT} 上，强烈建议使用多层陶瓷电容。内部稳压器稳定设计的关键是需要如上推荐的输出滤波电容 C_{LOAD} 和 ESR。

高压输出电气特性

$T_a=25^{\circ}\text{C}$

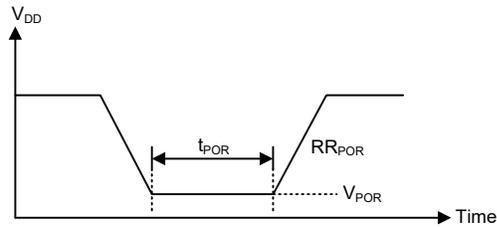
符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
HVPA0~HVPA3 引脚高压输出电气特性							
V_{IN}	输入电压	—	详见下方注释	3.0	—	13.5	V
I_{OH}	高压输出引脚源电流	—	$V_{OH}=0.9\times V_{IN}$, $V_{IN}=12\text{V}$	-300	-360	—	mA
I_{OL}	高压输出引脚灌电流	—	$V_{OL}=0.1\times V_{IN}$, $V_{IN}=12\text{V}$	300	360	—	mA
HVPB0~HVPB3 引脚高压输出电气特性							
V_{IN}	输入电压	—	详见下方注释	3.0	—	13.5	V
I_{OH}	高压输出引脚源电流	—	$V_{OH}=0.9\times V_{IN}$, $V_{IN}=12\text{V}$	-50	-90	—	mA
I_{OL}	高压输出引脚灌电流	—	$V_{OL}=0.1\times V_{IN}$, $V_{IN}=12\text{V}$	50	90	—	mA

注：HVPA_n 和 HVPB_n 引脚在电压为 3V 以下时只能作为一般输出高或输出低控制，高低电平切换速度不可超过 25Hz，否则会出现漏电流与波形失真问题。

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} 的最小时间	—	—	1	—	—	ms



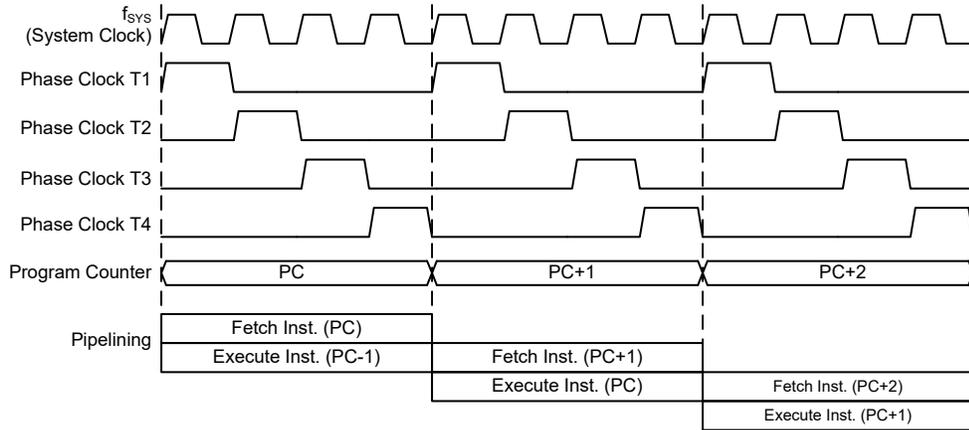
系统结构

内部系统结构是 Holtek 单片机具有良好性能的主要因素。由于采用 RISC 结构，此单片机具有高运算速度和高性能的特点。通过电流流水线的方式，指令的取得和执行同时进行，此举使得除了跳转和调用指令需多一个指令周期外，其它大部分标准指令或扩展指令都能分别在一个或两个指令周期内完成。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寄存器中被实现，且可以直接或间接寻址。简单的寄存器寻址方式和结构特性，确保了在提供具有最大可靠性和灵活性的 I/O 和 A/D 控制系统时，仅需要少数的外部器件。使得这些单片机适用于低成本和大量生产的控制应用。

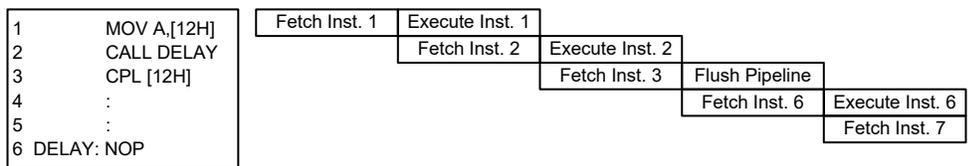
时序和流水线结构

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



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地址到程序寄存器来控制程序，对于条件跳转指令，一旦条件符合，在当前指令执行时取得的下一条指令将会被舍弃，而由一个空指令周期来取代。

程序计数器	
程序计数器高字节	PCL 寄存器
PC12~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。程序计数器的低字节可由程序直接进行读取，PCL 的使用可能引起程序跳转，因此需要额外的指令周期。

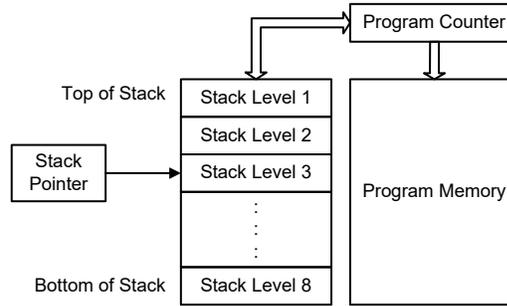
堆栈

堆栈是一个特殊的存储空间，用来存储程序计数器中的内容。堆栈是不可读取或写入的。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、借位或其它状态的变化，而相关的状态寄存器会因此更新内容以显示这些变化，ALU 所提供的功能如下：

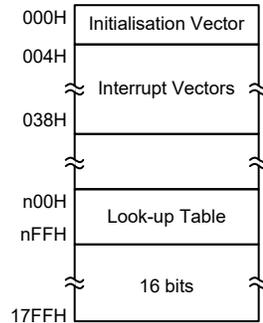
- 算术运算：
 - ADD, ADDM, ADC, ADCM, SUB, SUBM, SBC, SBCM, DAA
 - LADD, LADDM, LADC, LADCM, LSUB, LSUBM, LSBC, LSBCM, LDAA
- 逻辑运算：
 - AND, OR, XOR, ANDM, ORM, XORM, CPL, CPLA
 - LAND, LANDM, LOR, LORM, LXOR, LXORM, LCPL, LCPLA
- 移位运算：
 - RRA, RR, RRCA, RRC, RLA, RL, RLCA, RLC
 - LRRA, LRR, LRRCA, LRRC, LRLA, LRL, LRLCA, LRLC
- 递增和递减：
 - INCA, INC, DECA, DEC
 - LINCA, LINC, LDECA, LDEC
- 分支判断：
 - JMP, SZ, SZA, SNZ, SIZ, SDZ, SIZA, SDZA, CALL, RET, RETI
 - LSZ, LSZA, LSNZ, LSIZ, LSIZA, LSDZ, LSDZA

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，此所有单片机提供用户灵活便利的调试方法和项目开发规划及更新。

结构

程序存储器的容量为 6K×16 位，程序存储器用程序计数器来寻址，其中也包含数据、表格和中断入口。数据表格可以设定在程序存储器的任何地址，由表格指针来寻址。



程序存储器结构

特殊向量

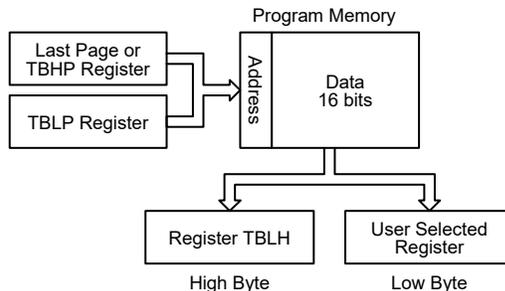
程序存储器内部某些地址保留用做诸如复位和中断入口等特殊用途。地址 000H 是芯片复位后的程序起始地址。在芯片复位之后，程序将跳到这个地址并开始执行。

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 TBLH 中。这些寄存器定义表格总的地址。

在设定完表格指针后，当数据存储器 [m] 位于 Sector 0，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。如果存储器 [m] 位于其它 Sector，表格数据可以使用如“LTABRD [m]”或“LTABRDL [m]”等扩展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，而高字节中未使用的位将被读取为“0”。

下图是查表中寻址 / 数据流程：



查表范例

以下范例说明表格指针和表格数据如何被定义和执行。这个例子使用的表格数据用 ORG 伪指令储存在存储器中。ORG 指令的值“1700H”指向的地址是 6K 程序存储器中最后一页的起始地址。表格指针的初始值设为“06H”，这可保证从数据表格读取的第一笔数据位于程序存储器地址“1706H”，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]”或“LTABRD[m]”指令被使用，则表格指针指向 TBHP 指定的页中 TBLP 指定的地址。在这个例子中，表格数据的高字节等于零，而当“TABRD [m]”或“LTABRD[m]”指令被执行时，此值将会自动的被传送到 TBLH 寄存器。

TBLH 寄存器为可读 / 可写寄存器，且能重复储存，若主程序和中断服务程序都使用表格读取指令，应注意对 TBLH 中数据的保护。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,17h          ; initialise high table pointer
mov tbhp,a
:
:
tabrd tempreg1     ; transfers value in table referenced by table pointer
                   ; data at program memory address "1706H" transferred to
                   ; tempreg1 and TBLH
dec tblp           ; reduce value of table pointer by one
tabrd tempreg2     ; transfers value in table referenced by table pointer
                   ; data at program memory address "1705H" transferred to
                   ; tempreg2 and TBLH
                   ; in this example the data "1AH" is transferred to
                   ; tempreg1 and data "0FH" to register tempreg2
:
:
org 1700h          ; 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。

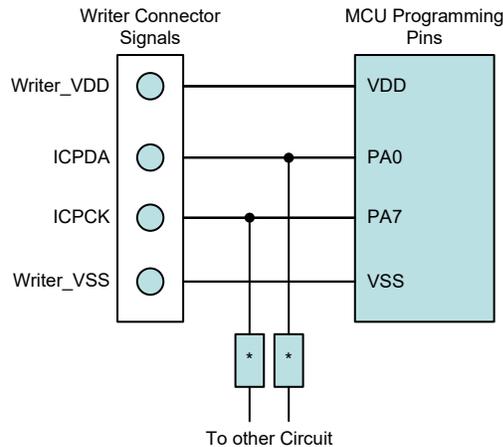
另外，Holtek 单片机提供 4 线接口的在线烧录方式。用户可将进行过烧录或未经过烧录的单片机芯片连同电路板一起制成，最后阶段进行程序的更新和程序的烧录，在无需去除或重新插入芯片的情况下方便地保持程序为最新版。

Holtek Flash 单片机与烧录器引脚对应表格如下：

Holtek 烧录器引脚名称	MCU 烧录引脚名称	功能
ICPDA	PA0	串行数据 / 地址烧录
ICPCK	PA7	时钟烧录
VDD	VDD	电源 (5V)
VSS	VSS	地

芯片内部程序存储器可以通过 4 线的接口在线进行烧录。其中 PA6 用于数据串行下载或上传、PA7 用于串行时钟、另外两条用于提供电源。芯片在线烧写的详细使用说明超出此文档的描述范围，将由专门的参考文献提供。

在烧录过程中，烧录器会控制 ICPDA 和 ICPCK 脚进行数据和时钟烧录，用户必须确保这两个引脚没有连接至其它输出脚。



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

片上调试 – OCDS

EV 芯片 BP45VH6N 用于单片机 BP45FH6N 仿真。此单片机的 EV 芯片提供片上调试功能 (OCDS – On-Chip Debug Support) 用于开发过程中的单片机调试。除了片上调试功能方面，EV 芯片和实际 MCU 在功能上几乎是兼容的。用户可将 OCSDA 和 OCDSCK 引脚连接至 Holtek HT-IDE 开发工具，从而实现 EV 芯片对实际 MCU 的仿真。OCSDA 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 OCSDA 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
OCSDA	OCSDA	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 (5V)
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。

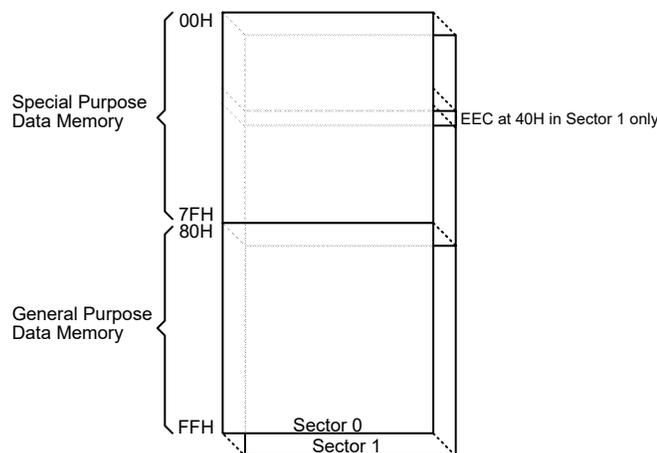
数据存储器分为两个部分，第一部分是特殊功能数据存储器。这些寄存器有固定的地址且与单片机的正确操作密切相关。大多特殊功能寄存器都可在程序控制下直接读取和写入，但有些被加以保护而不对用户开放。第二部分通用数据存储器是做一般用途使用，都可在程序控制下进行读取和写入。

结构

整个 8-bit 宽度的数据存储器被分为两个 Sector。每个 Sector 分为两部分，即特殊功能数据存储器 and 通用数据存储器。切换不同区域可通过设置存储器指针实现。此单片机的数据存储器的起始地址是“00H”。

特殊功能 数据存储器地址	通用数据存储器	
	容量	Sector: 地址
Sector 0: 00H~7FH	256×8	0: 80H~FFH
Sector 1: 00H~7FH		1: 80H~FFH

数据存储器概要



数据存储器结构

数据存储器寻址

此单片机支持扩展指令集，它并没有专门的寄存器用于选择不同的数据存储器 Sector。对于数据存储器，使用间接寻址访问方式时所需的 Sector 是通过 MP1H 或 MP2H 寄存器指定，而所选 Sector 的具体数据存储器地址的选择是通过 MP1L 或 MP2L 寄存器完成。

直接寻址可用于所有 Sector，通过扩展指令可以寻址所有可用的数据存储器空间。当所访问的数据存储器位于除 Sector 0 外的任何数据存储器 Sector 时，扩展指令可代替间接寻址方式用来访问数据存储器。标准指令和扩展指令的主要区别在于扩展指令中的数据存储器地址“m”有 9 个有效位，高字节表示选择的 Sector，低字节表示指定的地址。

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单独的位做置位或复位的操作，极大地方便了用户在数据存储器内进行位操作。

特殊功能数据存储器

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

	Sector 0	Sector 1		Sector 0	Sector 1
00H	IAR0		40H	EEA	EEC
01H	MP0		41H	EED	
02H	IAR1		42H		
03H	MP1L		43H		
04H	MP1H		44H	PWM1P	
05H	ACC		45H	PWM1D	
06H	PCL		46H	DLL1	
07H	TBLP		47H	PWM1C	
08H	TBLH		48H	DLLC	
09H	TBHP		49H	INTEG	
0AH	STATUS		4AH	INTC0	
0BH			4BH	INTC1	
0CH	IAR2		4CH	INTC2	
0DH	MP2L		4DH	INTC3	
0EH	MP2H		4EH	MFI0	
0FH	LVDC		4FH	MFI1	
10H	SMOD		50H	OCPPC	
11H	CTRL		51H	OUVPOPC	
12H		HVPAEN	52H	OUVP1PC	
13H	LVRC		53H	OUTPC	
14H	PA	HVPCA	54H	PMPS	
15H	PAC	HVPSA	55H	LVPUC	
16H	PAPU	HVPBEN	56H	OUVPOC0	
17H	PAWU		57H	OUVPOC1	
18H	PAS0	HVPCB	58H	OUVPOC2	
19H	PAS1	HVPSB	59H	OUVPOC3	
1AH			5AH	OVP0DAH	
1BH			5BH	OVP0DAL	
1CH	PCS0		5CH	UVP0DAH	
1DH	PCS1		5DH	UVP0DAL	
1EH	PDS0		5EH	OUVP1C0	
1FH	PDS1		5FH	OUVP1C1	
20H	PWM0P	TBC	60H	OUVP1C2	
21H	PWM0D		61H	OUVP1C3	
22H	DLL0		62H	OVP1DAH	
23H	PWM0C		63H	OVP1DAL	
24H	ADJ0DT	STMC0	64H	UVP1DAH	
25H	ADJ0S	STMC1	65H	UVP1DAL	
26H	ADJ0C	STMDL	66H	VBGRC	
27H	ADJ0MAXH	STMDH	67H		
28H	ADJ0MAXL	STMAL	68H		
29H	ADJ0MINH	STMAH	69H		
2AH	ADJ0MINL	STMRP	6AH	OCP0C0	
2BH	ADJ0BH	PTMC0	6BH	OCP0C1	
2CH	ADJ0BL	PTMC1	6CH	OCP0DA	
2DH	ADJ1DT	PTMDL	6DH	OCP0OCAL	
2EH	ADJ1S	PTMDH	6EH	OCP0CCAL	
2FH	ADJ1C	PTMAL	6FH	OCP1C0	
30H	ADJ1MAXH	PTMAH	70H	OCP1C1	
31H	ADJ1MAXL	PTMPRL	71H	OCP1DA	
32H	ADJ1MINH	PTMPRH	72H	OCP1OCAL	
33H	ADJ1MINL		73H	OCP1CCAL	
34H	ADJ1BH		74H	ADUDA0	
35H	ADJ1BL		75H	ADUDA1	
36H	WDTC		76H	ADUDA2	
37H	PC		77H	ADUDA3	
38H	PCC		78H	ADUC0	
39H	PCPU		79H		
3AH	PD		7AH	ADUC1	
3BH	PDC		7BH	SADC0	
3CH	PDPU		7CH	SADC1	
3DH	PB		7DH	SADC2	
3EH	PBC		7EH	SAD0H	
3FH	PBPU		7FH	SAD0L	

■ : Unused, read as 00H

▣ : Reserved, cannot be changed

特殊功能数据存储结构

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, IAR2

间接寻址寄存器 IAR0、IAR1 和 IAR2 的地址虽位于数据存储区，但其并没有实际的物理地址。间接寻址的方法准许使用间接寻址指针做数据操作，以取代定义实际存储器地址的直接存储器寻址方法。在间接寻址寄存器 IAR0、IAR1 和 IAR2 上的任何动作，将对间接寻址指针 MP0、MP1L/MP1H 或 MP2L/MP2H 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 Sector 0，而 IAR1 和 MP1L/MP1H、IAR2 和 MP2L/MP2H 可以访问任何 Sector。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, MP1L/MP1H, MP2L/MP2H

该单片机提供五个存储器指针，即 MP0、MP1L、MP1H、MP2L 和 MP2H。由于这些指针在数据存储区中能像普通的寄存器一般被操作，因此提供了一个寻址和数据追踪的有效方法。当对间接寻址寄存器进行任何操作时，单片机指向的实际地址是由存储器指针所指定的地址。MP0、IAR0 用于访问 Sector 0，而 MP1L/MP1H 和 IAR1、MP2L/MP2H 和 IAR2 可根据 MP1H 或 MP2H 寄存器访问所有的 Sector。使用扩展指令可对所有的数据存储区 Sector 进行直接寻址。

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

间接寻址程序举例

范例 1

```
data .section 'data'
adres1 db ?
adres2 db ?
adres3 db ?
adres4 db ?
block db ?
code .section at 0 code
org 00h
start:
    mov a,04h                ; setup size of block
    mov block,a
    mov a,offset adres1     ; Accumulator loaded with first RAM address
    mov mp0,a               ; setup memory pointer with first RAM address
loop:
    clr IAR0                 ; clear the data at address defined by MP0
    inc mp0                  ; increment memory pointer
    sdz block                ; check if last memory location has been cleared
    jmp loop
continue:
:
```

范例 2

```

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,01h                ; setup the memory sector
    mov mplh,a
    mov a,offset adres1     ; Accumulator loaded with first RAM address
    mov mp1l,a              ; setup memory pointer with first RAM address
loop:
    clr IAR1                 ; clear the data at address defined by MP1L
    inc mp1l                 ; increment memory pointer MP1L
    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。

使用扩展指令直接寻址程序举例

```

data .section 'data'
temp db ?
code .section at 0 code
org 00h
start:
    lmov a,[m]               ; move [m] data to acc
    lsub a,[m+1]             ; compare [m] and [m+1] data
    snz c                    ; [m]>[m+1]?
    jmp continue            ; no
    lmov a,[m]               ; yes, exchange [m] and [m+1] data
    mov temp,a
    lmov a,[m+1]
    mov [m],a
    mov a,temp
    lmov [m+1],a
continue:
    :

```

注：“m”是位于任何数据存储器 Sector 的某一地址。例如，m=1F0H 表示 Sector 1 中的地址 0F0H。

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中的特殊功能区域内，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

此 8 位的状态寄存器由 SC 标志位、CZ 标志位、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。

SC、CZ、Z、OV、AC 和 C 标志位通常反映最近运算的状态。

- C: 当加法运算的结果产生进位，或减法运算的结果没有产生借位时，则 C 被置位，否则 C 被清零，同时 C 也会被带进位的移位指令所影响。
- AC: 当低半字节加法运算的结果产生进位，或低半字节减法运算的结果没有产生借位时，AC 被置位，否则 AC 被清零。
- Z: 当算术或逻辑运算结果是零时，Z 被置位，否则 Z 被清零。
- OV: 当运算结果高两位的进位状态异或结果为 1 时，OV 被置位，否则 OV 被清零。
- PDF: 系统上电或执行“CLR WDT”指令会清零 PDF，而执行“HALT”指令则会置位 PDF。
- TO: 系统上电或执行“CLR WDT”或“HALT”指令会清零 TO，而当 WDT 溢出则会置位 TO。
- CZ: 不同指令不同标志位的操作结果。详细资料请参考寄存器定义部分。
- SC: 当 OV 与当前指令操作结果 MSB 执行“XOR”所得结果。

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	SC	CZ	TO	PDF	OV	Z	AC	C
R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W
POR	x	x	0	0	x	x	x	x

“x”：未知

- Bit 7 **SC**: 当 OV 与当前指令操作结果 MSB 执行“XOR”所得结果
- Bit 6 **CZ**: 不同指令不同标志位的操作结果。
对于 SUB/SUBM/LSUB/LSUBM 指令, CZ 等于 Z 标志位。
对于 SBC/SBCM/LSBC/LSBCM 指令, CZ 等于上一个 CZ 标志位与当前零标志位执行“AND”所得结果。对于其它指令, CZ 标志位无影响。
- 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
进位标志位 C 也受循环移位指令的影响。

EEPROM 数据存储

此单片机的一个特性是内建 EEPROM 数据存储，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 数据存储结构

EEPROM 数据存储容量为 64×8。由于映射方式与程序存储器和数据存储器不同，因此不能像其它类型的存储器一样寻址。使用一个地址和一个数据寄存器以及一个仅位于 Sector 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 Sector 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 Sector 1 中，仅可通过 MP1L/MP1H 和 IAR1 或 MP2L/MP2H 和 IAR2 进行间接读取或写入。由于 EEC 控制寄存器位于 Sector 1 中的“40H”，在 EEC 寄存器上的任何操作被执行前，MP1L 或 MP2L 必须先设为“40H”，MP1H 或 MP2H 被设为“01H”。

寄存器名称	位							
	7	6	5	4	3	2	1	0
EEA	—	—	EEA5	EEA4	EEA3	EEA2	EEA1	EEA0
EED	D7	D6	D5	D4	D3	D2	D1	D0
EEC	—	—	—	—	WREN	WR	RDEN	RD

EEPROM 寄存器列表

• EEA 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	EEA5	EEA4	EEA3	EEA2	EEA1	EEA0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 未定义，读为“0”

Bit 5~0 **EEA5~EEA0**: 数据 EEPROM 地址 bit 5~bit 0

• EED 寄存器

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

Bit 7~0 **D7~D0**: 数据 EEPROM 数据 bit 7~bit 0

• EEC 寄存器

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

Bit 7~4 未定义，读为“0”

Bit 3 **WREN**: 数据 EEPROM 写使能位

0: 除能
1: 使能

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

Bit 2 **WR**: EEPROM 写控制位

0: 写周期结束
1: 启动写周期

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: 启动读周期

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”。WR 和 RD 不能同时置为“1”。

2. 需确保 f_{SUB} 时钟运行稳定后才可执行写操作。

3. 需确保写操作已执行完毕后才可改动 EEC 寄存器内容。

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，接着将 EEC 寄存器中的读使能位 RDEN 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，要写入的数据要存入 EED 寄存器中。写数据至 EEPROM，EEC 寄存器中的写使能位 WREN 先置为高以使能写功能。之后将 WR 位置为高，初始化一个写周期。这两个指令必须在两个连续的指令周期内执行。在执行任何写操作之前，总中断位 EMI 要先清零，写周期开始后，再将 EMI 置为高。需要注意的是若 WR 位已置为高而 WREN 位还未被设置则不能开始写操作。由于控制 EEPROM 写周期是一个内部时钟，与单片机的系统时钟异步，所以数据写入 EEPROM 的时间将有所延迟。可通过轮询 EEC 寄存器中的 WR 位或判断 EEPROM 中断以侦测写周期是否完成。若写周期完成，WR 位将自动清除为“0”，通知用户数据已写入 EEPROM。因此，应用程序将轮询 WR 位以确定写周期是否结束。

写保护

防止误写入的写保护有以下几种。单片机上电后控制寄存器中的写使能位将被清除以杜绝任何写入操作。上电后存储区指针高字节寄存器 MP1H 和 MP2H 将重置为“0”，这意味着数据存储区 Sector 0 被选中。由于 EEPROM 控制寄存器位于 Sector 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 写中断。由于 EEPROM 写中断包含在多功能中断中，相应的多功能中断使能位需被设置。当 EEPROM 写周期结束，DEF 请求标志位及其相关多功能中断请求标志位将被置位。若 EEPROM 和多功能中断使能且堆栈未满的情况下将跳转到相应的多功能中断向量中执行。当中断被响应，只有多功能中断标志位将自动复位，而 EEPROM 写中断标志将通过应用程序手动复位。更多细节将在中断章节讲述。

编程注意事项

必须注意的是数据不会无意写入 EEPROM。在没有写动作时写使能位被正常清零可以增强保护功能。存储器指针高字节也可以正常清零以阻止进入 EEPROM 控制寄存器存在的 Sector 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。

写数据时，WREN 位置为“1”后，WR 需立即设置为高，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，EEPROM 读或写周期彻底完成前单片机不能进入空闲或休眠模式，否则将导致 EEPROM 读或写操作失败。

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 MP1L
MOV MP1L, A              ; MP1L points to EEC register
MOV A, 01H                ; setup Memory Pointer high byte MP1H
MOV MP1H, 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 MP1H
MOV A, EED                ; move read data to register
MOV READ_DATA, A
```

注：即使目标地址是连续的，每次执行读操作仍需重新定义地址寄存器，接着置位 RD 以启动一个读周期。

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 MP1L
MOV MP1L, A              ; MP1L points to EEC register
MOV A, 01H                ; setup Memory Pointer high byte MP1H
MOV MP1H, 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 MP1H
    
```

振荡器

不同的振荡器选择可以让使用者在不同的应用需求中实现更大范围的功能。振荡器的灵活性使得在速度和功耗方面可以达到最优化。振荡器的选择和操作是通过寄存器和配置选项共同实现的。

振荡器概述

振荡器除了作为系统时钟源，还作为看门狗定时器和时基功能的时钟源。集成的两个内部振荡器不需要任何外围器件。它们提供高速和低速系统振荡器。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。

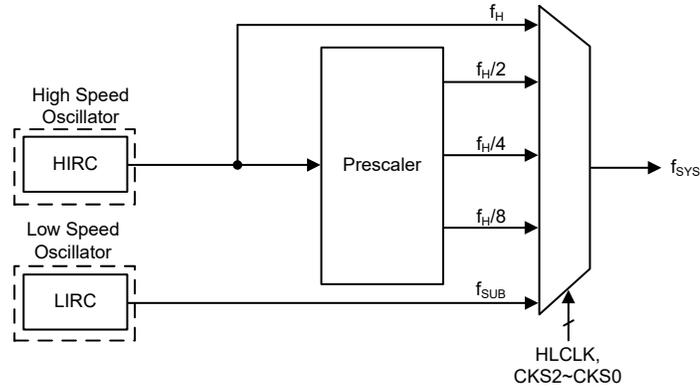
类型	名称	频率
内部高速 RC	HIRC	8/12/16MHz
内部低速 RC	LIRC	32kHz

振荡器类型

系统时钟配置

此单片机有两个系统振荡器，包括一个高速振荡器和一个低速振荡器。高速振荡器为内部 8/12/16MHz RC 振荡器 HIRC。低速振荡器为内部 32kHz RC 振荡器 LIRC。使用高速或低速振荡器作为系统时钟的选择是通过设置 SMOD 寄存器中的 HLCLK 位及 CKS2~CKS0 位决定的，系统时钟可动态选择。

高速振荡器的实际时钟由配置选项选择。低速或高速系统时钟频率是由 SMOD 寄存器的 HLCLK 位及 CKS2~CKS0 位决定的。请注意，两个振荡器必须做出选择，即一个高速和一个低速振荡器。



系统时钟配置

内部 RC 振荡器 – HIRC

内部 RC 振荡器是一个集成的系统振荡器，不需其它外部器件。内部 RC 振荡器具有三种固定频率：8MHz、12MHz 和 16MHz，需通过配置选项选择。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡频率因 V_{DD} 、温度以及芯片制成工艺不同的影响减至最低程度。因此，在电源为 3V 或 5V 且温度为 25°C 时，固定振荡器频率 8MHz、12MHz 或 16MHz 的容限将在 1% 以内。应注意如果选择了该内部时钟，无需额外的引脚。

内部 32kHz 振荡器 – LIRC

内部 32kHz 系统振荡器也是一个低频振荡器。该完全集成的 RC 振荡器在 5V 电压下运行的典型频率值为 32kHz 且无需外部元器件。因此，在电源为 5V 且温度为 25°C 时，固定振荡器频率 32kHz 的容限将在 10% 以内。

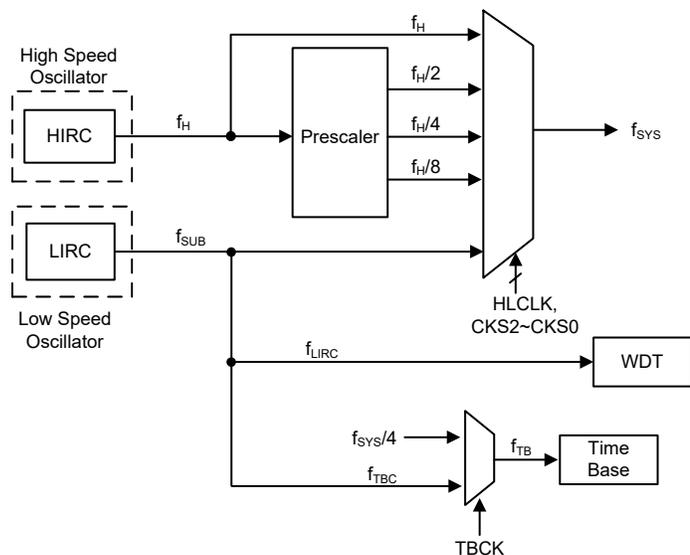
工作模式和系统时钟

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最佳性能 / 功耗比。

系统时钟

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

主系统时钟可来自高频时钟源 f_H 或低频时钟源 f_{SUB} ，通过 SMOD 寄存器中的 HLCLK 位及 CKS2~CKS0 位进行选择。高频时钟来自 HIRC 振荡器，低频系统时钟源来自内部时钟 LIRC 振荡器。其它系统时钟还有高速系统振荡器的分频 $f_H/2 \sim f_H/8$ 。



单片机时钟配置

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

系统工作模式

单片机有 5 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 3 种工作模式：休眠模式、空闲模式 0 和空闲模式 1 用于单片机 CPU 关闭时以节省耗电。

工作模式	说明			
	CPU	f_{SYS}	f_{LIRC}	f_{TBC}
快速模式	On	$f_H \sim f_H/8$	On	On
低速模式	On	f_{SUB}	On	On
空闲模式 0	Off	Off	On	On
空闲模式 1	Off	On	On	On
休眠模式	Off	Off	On	Off

注：由于 WDT 功能始终使能， f_{LIRC} 时钟将始终开启。

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 1~8 的不等比率，实际的比率由 SMOD 寄存器中的 CKS2~CKS0 位及 HLCLK 位选择的。单片机使用高速振荡器分频作为系统时钟可减少工作电流。

低速模式

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

休眠模式

在 HALT 指令执行后且 SMOD 寄存器中 IDLEN 位为低时，系统进入休眠模式。在休眠模式中，CPU 停止运行，然而 f_{LIRC} 会继续运行以保持看门狗功能使能。

空闲模式 0

执行 HALT 指令后且 SMOD 寄存器中 IDLEN 位为高，CTRL 寄存器中 FSYSON 位为低时，系统进入空闲模式 0。在空闲模式 0 中，CPU 停止，系统振荡器停止，低频时钟开启。

空闲模式 1

执行 HALT 指令后且 SMOD 寄存器中 IDLEN 位为高，CTRL 寄存器中 FSYSON 位为高时，系统进入空闲模式 1。在空闲模式 1 中，CPU 停止，系统振荡器继续运行，该系统振荡器可以为高速或低速系统振荡器。在空闲模式 1 中低频时钟开启。

注：若 LVDEN=1 时，MCU 进入休眠或空闲模式，LVD 和 bandgap 不会关掉，并且 f_{SUB} 也会被强制开启。

控制寄存器

寄存器 SMOD 和 CTRL 用于控制系统时钟和相应的振荡器设置。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SMOD	CKS2	CKS1	CKS0	—	LTO	HTO	IDLEN	HLCLK
CTRL	FSYSON	—	—	—	—	LVRF	LRF	WRF

系统工作模式控制寄存器列表

• SMOD 寄存器

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

Bit 7~5 **CKS2~CKS0**: 系统时钟选择位

000: f_{SUB}
001: f_{SUB}
010: 未定义
011: 未定义
100: 未定义
101: $f_H/8$
110: $f_H/4$
111: $f_H/2$

这三位用于选择系统时钟源。除了 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 未定义，读为“0”

Bit 3 **LTO**: LIRC 低速振荡器就绪标志位

0: 未就绪
1: 就绪

此位为低速系统振荡器就绪标志位，用于表明低速系统振荡器在系统上电复位或经唤醒后何时稳定下来。该位转换为高需 1~2 个时钟周期。

- Bit 2 **HTO:** HIRC 高速振荡器就绪标志位
 0: 未就绪
 1: 就绪
 此位为高速系统振荡器就绪标志位，用于表明高速系统振荡器唤醒后何时稳定下来。如果使用 HIRC 振荡器，该位转换为高需 15~16 个时钟周期。
- Bit 1 **IDLEN:** 空闲模式控制位
 0: 除能
 1: 使能
 此位为空闲模式控制位，用于决定 HALT 指令执行后发生的动作。若此位为高，当指令 HALT 执行后，单片机进入空闲模式。若 FSYSON 位为高，在空闲模式 1 中 CPU 停止运行，系统时钟将继续工作以保持外围功能继续工作；若 FSYSON 为低，在空闲模式 0 中 CPU 和系统时钟都将停止运行。若此位为低，单片机将在 HALT 指令执行后进入休眠模式。
- Bit 0 **HLCLK:** 系统时钟选择位
 0: $f_H/2 \sim f_H/8$ 或 f_{SUB}
 1: f_H
 此位用于选择 f_H 或 $f_H/2 \sim f_H/8$ 还是 f_{SUB} 作为系统时钟。该位为高时选择 f_H 作为系统时钟，为低时则选择 $f_H/2 \sim f_H/8$ 或 f_{SUB} 作为系统时钟。当系统时钟由 f_H 时钟向 f_{SUB} 时钟转换时， f_H 将自动关闭以降低功耗。

● **CTRL 寄存器**

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

“x”：未知

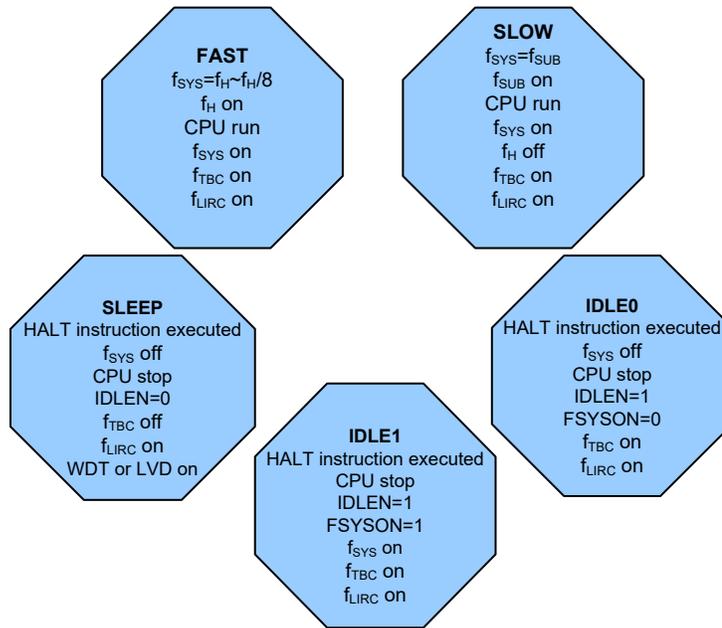
- Bit 7 **FSYSON:** 空闲模式下系统时钟 f_{SYS} 控制位
 0: 除能
 1: 使能
- Bit 6~3 未定义，读为“0”
- Bit 2 **LVRF:** LVR 硬件复位标志位
 具体描述见其它章节
- Bit 1 **LRF:** LVRC 寄存器软件复位标志位
 具体描述见其它章节
- Bit 0 **WRF:** WDTC 寄存器软件复位标志位
 具体描述见其它章节

工作模式切换

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

简单来说，快速模式和低速模式间的切换仅需设置 SMOD 中的 HLCLK 位及 CKS2~CKS0 位即可实现，而快速模式 / 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 SMOD 寄存器中的 IDLEN 位和 CTRL 寄存器中的 FSYSON 位决定的。

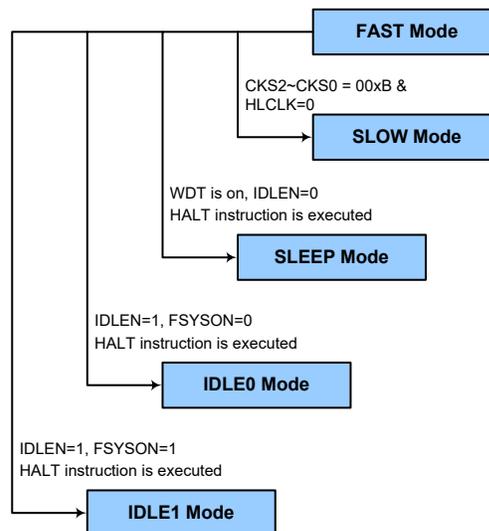
当 HLCLK 位变为低电平时，时钟源将由高速时钟源 f_H 转换成时钟源 $f_H/2 \sim f_H/8$ 或 f_{SUB} 。若时钟源来自 f_{SUB} ，高速时钟源将停止运行以节省耗电。所附流程图显示了单片机在不同工作模式间切换时的变化。



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

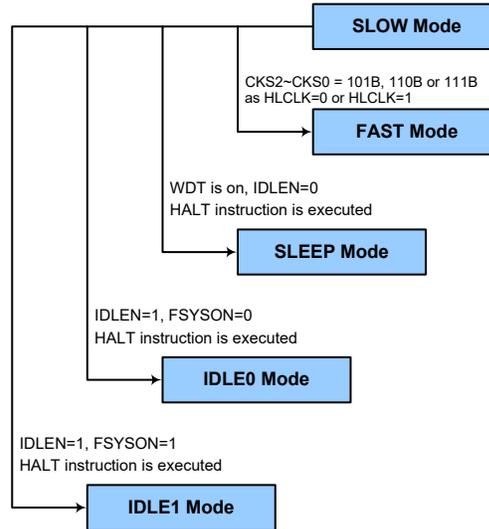
低速模式的时钟源来自 LIRC 振荡器，因此要求该振荡器在所有模式切换动作发生前稳定下来，可通过 SMOD 寄存器中的 LTO 位进行监测。



低速模式切换到快速模式

在低速模式系统时钟来自时钟 f_{SUB} 。切换到使用高速系统时钟振荡器的正常模式需设置 HLCLK 位为“1”，也可设置 HLCLK 位为“0”但 CKS2~CKS0 需设为“101”、“110”或“111”。

然而，若低速模式下时钟 f_H 未使用从而关闭，则从低速模式切换到快速模式时高频时钟重新起振直到稳定需要一定的时间，可通过 SMOD 寄存器中的 HTO 位进行监测。高速振荡器的稳定时间可参考系统上电时间电气特性表格。



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”指令前需设置寄存器 SMOD 中 IDLEN 位为“0”，且 WDT 或者 LVD 功能使能。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟和时基时钟停止运行，应用程序停止在“HALT”指令处。WDT 或 LVD 继续运行，其时钟源来自 f_{LIRC} 。
- 数据存储器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。
- 由于 WDT 功能始终使能，WDT 将被清零并重新开始计数。

进入空闲模式 0

进入空闲模式 0 的方法仅有一种，即在应用程序执行“HALT”指令前设置寄存器 SMOD 中 IDLEN 位为“1”且 CTRL 寄存器中的 FSYSON 位为“0”。在上述条件下执行 HALT 指令后，将发生的情况如下：

- 系统时钟停止运行，应用程序停止在“HALT”指令处，时基时钟 f_{TBC} 和低频时钟 f_{LIRC} 将继续运行。
- 数据存储器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。
- 由于 WDT 功能始终使能，WDT 将被清零并重新开始计数。

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”指令前需设置寄存器 SMOD 中 IDLEN 位为“1”且 CTRL 寄存器中的 FSYSON 位为“1”。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟 f_{SYS} 、时基时钟 f_{TBC} 和低频时钟 f_{LIRC} 开启，应用程序停止在“HALT”指令处。
- 数据存储器和寄存器的内容将保持当前值。
- 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。
- 由于 WDT 功能始终使能，WDT 将被清零并重新开始计数。

待机电流的注意事项

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

另外还需注意单片机设为输出的 I/O 引脚上的负载。应将它们设置在有最小拉电流的状态或将它们和其它的 CMOS 输入一样接到没有拉电流的外部电路上。还应注意的，如果选择 LIRC 振荡器使能，会导致耗电增加。

唤醒

单片机可进入休眠或空闲模式以节省功耗，此期间 MCU 将停止。然而，当单片机再次被唤醒，初始系统时钟的重启、稳定直到可为正常运行提供时钟源，这个过程需要一定的时间。

单片机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

当单片机执行 HALT 指令，PDF 将被置位。系统上电或执行清除看门狗的指令，会清零 PDF。若系统由 WDT 溢出唤醒，将会发生看门狗定时器复位。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滿，则中断可以马上执行。如果在进入休眠或空闲模式之前中断标志位已经被设置为“1”，则相关中断的唤醒功能将无效。

看门狗定时器

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

看门狗定时器时钟源

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

看门狗定时器控制寄存器

WDTC 寄存器用于选择溢出周期，控制 WDT 功能的使能及 MCU 软件复位操作。

• 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 功能使能控制

01010 或 10101: 使能

其它值: MCU 复位

如果由于不利的环境因素使这些位变为其它值，单片机将复位。复位动作发生在 t_{SRESET} 时间后，此复位完成后 CTRL 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

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

001: $2^{10}/f_{LIRC}$

010: $2^{12}/f_{LIRC}$

011: $2^{14}/f_{LIRC}$

100: $2^{15}/f_{LIRC}$

101: $2^{16}/f_{LIRC}$

110: $2^{17}/f_{LIRC}$

111: $2^{18}/f_{LIRC}$

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。

• CTRL 寄存器

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

“x”: 未知

Bit 7 **FSYSON**: 空闲模式下 f_{sys} 控制位

具体描述见其它章节

Bit 6~3 未定义，读为“0”

Bit 2 **LVRF**: LVR 复位标志

具体描述见其它章节

- Bit 1 **LRF**: LVRC 寄存器软件复位标志位
 具体描述见其它章节
- Bit 0 **WRF**: WDTC 寄存器软件复位标志位
 0: 未发生
 1: 发生
 当发生 WDTC 控制寄存器软件复位时，此位被置为“1”。此位只能通过程序清零。

看门狗定时器操作

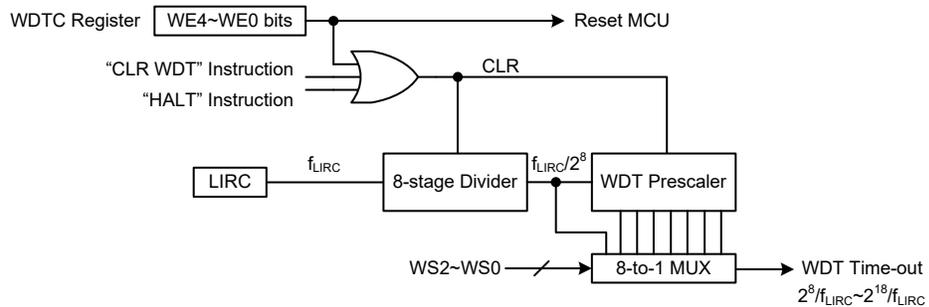
当 WDT 溢出时，看门狗定时器产生一个芯片复位的动作。这也就意味着正常工作期间，用户需在应用程序中看门狗溢出前有策略地清看门狗定时器以防止其产生复位，可使用清除看门狗指令实现。无论什么原因，程序失常跳转到一个未知的地址或进入一个死循环，这些清除指令都不能被正确执行，此种情况下，看门狗将溢出以使单片机复位。WDTC 寄存器中的 WE4~WE0 位可使能及复位看门狗定时器。如果 WE4~WE0 为 10101B 或 01010B，则 WDT 使能；如果由于受到干扰，WE4~WE0 变为其它任意值，则经过 t_{SRESET} 延迟时间后单片机复位。上电后 WE4~WE0 位的初始值为 01010B。

WE4 ~ WE0	WDT 功能
10101B 或 01010B	使能
其它值	MCU 复位

看门狗定时器使能 / 复位控制

程序正常运行时，WDT 溢出将导致芯片复位，并置位状态标志位 TO。若系统处于休眠或空闲模式，当 WDT 发生溢出时，状态寄存器中的 TO 将被置位，仅程序计数器 PC 和堆栈指针 SP 将被复位。有三种方法可以用来清除 WDT 的内容。第一种是 WDT 复位，即将 WE4~WE0 位设置成除了 01010B 和 10101B 外的任意值；第二种是通过软件清除指令，而第三种是通过“HALT”指令。该单片机只使用一条清看门狗指令“CLR WDT”。因此只要执行“CLR WDT”便清除 WDT。

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



看门狗定时器

复位和初始化

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

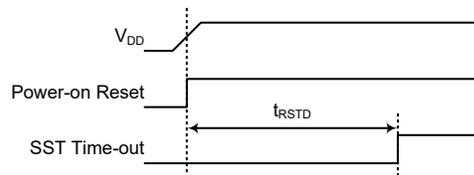
另一种复位为看门狗溢出单片机复位。不同方式的复位操作会对寄存器产生不同的影响。另一种复位为低电压复位即 LVR 复位，在电源供应电压低于 LVR 设定的值时，系统会产生 LVR 复位。

复位功能

此单片机提供多种内部事件触发复位。

上电复位

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



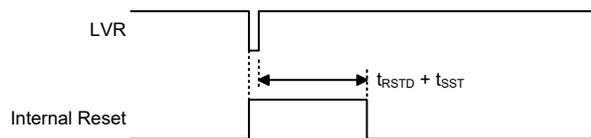
注： t_{RSTD} 为上电延迟时间，典型值为 16.7ms。

上电复位时序图

低电压复位 – LVR

单片机具有低电压复位电路，用来监测它的电源电压。在快速和低速模式下，低电压复位功能始终使能于特定的电压值， V_{LVR} 。例如在更换电池的情况下，单片机供应的电压可能会在 $0.9V \sim V_{LVR}$ 之间，这时 LVR 将会自动复位单片机且 CTRL 寄存器中的 LVRF 标志位置位。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 LVD/LVR 电气特性中的 t_{LVR} 参数值。如果低电压存在时间没超过 t_{LVR} 参数的值，则 LVR 将会忽略它且不会执行复位功能。

V_{LVR} 参数值固定为 2.55V。LVS7~LVS0 位仍可对 LVR 功能进行控制。若由于受到干扰 LVS7~LVS0 变为其它值时，需经过 t_{SRESET} 时间发生复位。此时 CTRL 寄存器的 LRF 位被置位。上电后寄存器的值为 01010101B。注意，当单片机进入空闲或休眠模式时，LVR 功能将自动除能。



注： t_{RSTD} 为上电延迟时间，典型值为 16.7ms。

低电压复位时序图

• LVRC 寄存器

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

Bit 7~0 **LVS7~LVS0**: LVR 电压选择

01010101: 2.55V

00110011: 2.55V

10011001: 2.55V

10101010: 2.55V

其它值: MCU 复位 – 寄存器复位为 POR 值

当相应的低电压出现后, 且此低电压保持时间超过 t_{LVR} 值, 将会产生 MCU 复位。

MCU 复位后寄存器中的值保持不变。

除上述定义的值外, 其它值都会产生 MCU 复位。复位操作将会在 t_{SRESET} 时间后执行。注意的是此处 MCU 复位后, 寄存器的值将恢复到上电复位值。

• CTRL 寄存器

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

“x”: 未知

Bit 7 **FSYSON**: 空闲模式下 f_{SYS} 控制位

具体描述见其它章节。

Bit 6~3 未定义, 读为 “0”

Bit 2 **LVRF**: LVR 复位标志

0: 未发生

1: 发生

当低电压复位情况发生时此位设置为 “1”。此位只能通过程序清零。

Bit 1 **LRF**: LVRC 寄存器软件复位标志位

0: 未发生

1: 发生

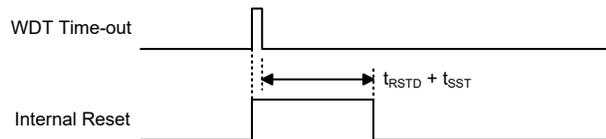
当 LVRC 寄存器包含任何未定义的 LVR 电压值时此位设置为 “1”, 相当于软件复位功能。此位只能通过程序清零。

Bit 0 **WRF**: WDTC 寄存器软件复位标志位

具体描述见其它章节。

正常运行时看门狗溢出复位

在快速和低速模式下正常运行时发生看门狗溢出复位, 看门狗溢出标志位 TO 将被设为 “1”。

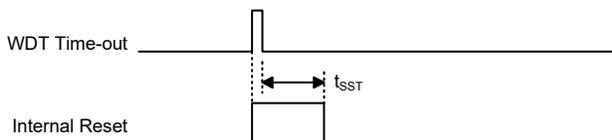


注: t_{RSTD} 为上电延迟时间, 典型值为 16.7ms

正常运行时看门狗溢出复位时序图

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“0”及 TO 位被设为“1”外，绝大部分的条件保持不变。图中 t_{SST} 的详细说明请参考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 PDF 和 TO 位存放在状态寄存器中，由休眠或空闲模式功能或看门狗计数器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”：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
WDT, 时基	复位后清除, WDT 重新计数
定时器模块	所有定时器模块除能
输入 / 输出口	I/O 口设置为输入类型
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。

寄存器	上电复位	LVR 复位 (正常运行)	WDT 溢出 (正常运行)	WDT 溢出 (空闲 / 休眠)
IAR0	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP0	0000 0000	0000 0000	0000 0000	uuuu uuuu
IAR1	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1L	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP1H	0000 0000	0000 0000	0000 0000	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu

寄存器	上电复位	LVR 复位 (正常运行)	WDT 溢出 (正常运行)	WDT 溢出 (空闲/休眠)
TBHP	---x xxxx	---u uuuu	---u uuuu	---u uuuu
STATUS	xx00 xxxx	xxuu uuuu	uu1u uuuu	uull uuuu
IAR2	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2L	0000 0000	0000 0000	0000 0000	uuuu uuuu
MP2H	0000 0000	0000 0000	0000 0000	uuuu uuuu
LVDC	--00 -000	--00 -000	--00 -000	--uu -uuu
SMOD	111- 0010	111- 0010	111- 0010	uuu- uuuu
CTRL	0--- -x00	0--- -1uu	0--- -uuu	u--- -uuu
LVRC	0101 0101	uuuu uuuu	0101 0101	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS0	0000 00--	0000 00--	0000 00--	uuuu uu--
PAS1	--00 0000	--00 0000	--00 0000	--uu uuuu
PCS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCS1	--00 0000	--00 0000	--00 0000	--uu uuuu
PDS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDS1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWM0P	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWM0D	0000 0000	0000 0000	0000 0000	uuuu uuuu
DLL0	0000 ----	0000 ----	0000 ----	uuuu ----
PWM0C	0000 0000	0000 0000	0000 0000	uuuu uuuu
AD0JDT	--00 0000	--00 0000	--00 0000	--uu uuuu
ADJ0S	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ0C	000- xxx-	000- xxx-	000- xxx-	uuu- uuu-
ADJ0MaxH	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ0MaxL	0000 ----	0000 ----	0000 ----	uuuu ----
ADJ0MinH	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ0MinL	0000 ----	0000 ----	0000 ----	uuuu ----
ADJ0BH	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ0BL	0000 ----	0000 ----	0000 ----	uuuu ----
AD1JDT	--00 0000	--00 0000	--00 0000	--uu uuuu
ADJ1S	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ1C	000- xxx-	000- xxx-	000- xxx-	uuu- uuu-
ADJ1MaxH	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ1MaxL	0000 ----	0000 ----	0000 ----	uuuu ----
ADJ1MinH	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADJ1MinL	0000 ----	0000 ----	0000 ----	uuuu ----
ADJ1BH	0000 0000	0000 0000	0000 0000	uuuu uuuu

寄存器	上电复位	LVR 复位 (正常运行)	WDT 溢出 (正常运行)	WDT 溢出 (空闲 / 休眠)
ADJ1BL	0000 ----	0000 ----	0000 ----	uuuu ----
WDTC	0101 0011	0101 0011	0101 0011	uuuu uuuu
PC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PD	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PB	---- 1111	---- 1111	---- 1111	---- uuuu
PBC	---- 1111	---- 1111	---- 1111	---- uuuu
PBPU	---- 0000	---- 0000	---- 0000	---- uuuu
EEA	--00 0000	--00 0000	--00 0000	--uu uuuu
EED	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWM1P	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWM1D	0000 0000	0000 0000	0000 0000	uuuu uuuu
DLL1	0000 ----	0000 ----	0000 ----	uuuu ----
PWM1C	0000 0000	0000 0000	0000 0000	uuuu uuuu
DLLC	10-- ---0	10-- ---0	10-- ---0	uu-- ---u
INTEG	--00 0000	--00 0000	--00 0000	--uu uuuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC2	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC3	-000 0000	-000 0000	-000 0000	-uuu uuuu
MFI0	-000 0000	-000 0000	-000 0000	-uuu uuuu
MFI1	-000 0000	-000 0000	-000 0000	-uuu uuuu
OCPPC	0000 0000	0000 0000	0000 0000	uuuu uuuu
OUPV0PC	0000 0000	0000 0000	0000 0000	uuuu uuuu
OUPV1PC	0000 0000	0000 0000	0000 0000	uuuu uuuu
OUTPC	0000 0000	0000 0000	0000 0000	uuuu uuuu
PMPS	---- --00	---- --00	---- --00	---- --uu
LVPUC	---- ---0	---- ---0	---- ---0	---- ---u
OUPV0C0	--00 0000	--00 0000	--00 0000	--uu uuuu
OUPV0C1	--00 0000	--00 0000	--00 0000	--uu uuuu
OUPV0C2	0001 0000	0001 0000	0001 0000	uuuu uuuu
OUPV0C3	0001 0000	0001 0000	0001 0000	uuuu uuuu
OVP0DAH	---- 0000	---- 0000	---- 0000	---- uuuu
OVP0DAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
UVP0DAH	---- 0000	---- 0000	---- 0000	---- uuuu
UVP0DAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
OUPV1C0	--00 0000	--00 0000	--00 0000	--uu uuuu

寄存器	上电复位	LVR 复位 (正常运行)	WDT 溢出 (正常运行)	WDT 溢出 (空闲/休眠)
OUIVPC1	--00 0000	--00 0000	--00 0000	--uu uuuu
OUIVPC2	0001 0000	0001 0000	0001 0000	uuuu uuuu
OUIVPC3	0001 0000	0001 0000	0001 0000	uuuu uuuu
OVPIDAH	---- 0000	---- 0000	---- 0000	---- uuuu
OVPIDAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
UVPIDAH	---- 0000	---- 0000	---- 0000	---- uuuu
UVPIDAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
VBGRC	---- ---0	---- ---0	---- ---0	---- ---u
OCP0C0	0000 0--0	0000 0--0	0000 0--0	uuuu u--u
OCP0C1	--00 0000	--00 0000	--00 0000	--uu uuuu
OCP0DA	0000 0000	0000 0000	0000 0000	uuuu uuuu
OCP0CAL	0010 0000	0010 0000	0010 0000	uuuu uuuu
OCP0CCAL	0001 0000	0001 0000	0001 0000	uuuu uuuu
OCP1C0	0000 0--0	0000 0--0	0000 0--0	uuuu u--u
OCP1C1	--00 0000	--00 0000	--00 0000	--uu uuuu
OCP1DA	0000 0000	0000 0000	0000 0000	uuuu uuuu
OCP1CAL	0010 0000	0010 0000	0010 0000	uuuu uuuu
OCP1CCAL	0001 0000	0001 0000	0001 0000	uuuu uuuu
ADUDA0	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADUDA1	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADUDA2	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADUDA3	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADUC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADUC1	--00 0000	--00 0000	--00 0000	--uu uuuu
SADC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADC1	0000 -000	0000 -000	0000 -000	uuuu -uuu
SADC2	0-00 0000	0-00 0000	0-00 0000	u-uu uuuu
SADOH	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu (ADRFS=0)
				---- uuuu (ADRFS=1)
SADOL	xxxx ----	xxxx ----	xxxx ----	uuuu ---- (ADRFS=0)
				uuuu uuuu (ADRFS=1)
HVPAEN	---- 0000	---- 0000	---- 0000	---- uuuu
HVPCA	---- 0000	---- 0000	---- 0000	---- uuuu
HVPSA	---- 0000	---- 0000	---- 0000	---- uuuu
HVPBEN	---- 0000	---- 0000	---- 0000	---- uuuu
HVPCB	---- 0000	---- 0000	---- 0000	---- uuuu
HVPSB	---- 0000	---- 0000	---- 0000	---- uuuu

寄存器	上电复位	LVR 复位 (正常运行)	WDT 溢出 (正常运行)	WDT 溢出 (空闲/休眠)
TBC	0011 -111	0011 -111	0011 -111	uuuu -uuu
STMC0	0000 0---	0000 0---	0000 0---	uuuu u---
STMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMDH	---- --00	---- --00	---- --00	---- --uu
STMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
STMAH	---- --00	---- --00	---- --00	---- --uu
STMRP	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMC0	---- --00	---- --00	---- --00	---- --uu
PTMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDH	---- --00	---- --00	---- --00	---- --uu
PTMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMAH	---- --00	---- --00	---- --00	---- --uu
PTMPRL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMPRH	---- --00	---- --00	---- --00	---- --uu
EEC	---- 0000	---- 0000	---- 0000	---- uuuu

注：“u”表示不改变
 “x”表示未知
 “-”表示未定义

输入 / 输出端口

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

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

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	—	—	—	—	PB3	PB2	PB1	PB0
PBC	—	—	—	—	PBC3	PBC2	PBC1	PBC0
PBPU	—	—	—	—	PBPU3	PBPU2	PBPU1	PBPU0
PC	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	PCC7	PCC6	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0

寄存器名称	位							
	7	6	5	4	3	2	1	0
PCPU	PCPU7	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PDC	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PDP	PDP7	PDP6	PDP5	PDP4	PDP3	PDP2	PDP1	PDP0
LVPUC	—	—	—	—	—	—	—	LVPU

“—”：未定义，读为“0”

I/O 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 LVPUC 和 P xPU~PDP 来设置，它用一个 PMOS 晶体管来实现上拉电阻功能。寄存器 P xPU 用于确定是否使能上拉功能，而寄存器 LVPUC 用于为低电压电源应用选择上拉电阻值。

需要注意的是，当 I/O 引脚设为数字输入或 NMOS 输出时，上拉功能才会受相应的上拉电阻控制寄存器控制开启，其它状态下上拉功能不可用。

• P xPU 寄存器

Bit	7	6	5	4	3	2	1	0
Name	P xPU7	P xPU6	P xPU5	P xPU4	P xPU3	P xPU2	P xPU1	P xPU0
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

P xPUn: I/O 端口 x 引脚上拉功能控制位

- 0: 除能
- 1: 使能

P xPUn 位用于控制引脚上拉功能。此处的“x”可为 A、B、C 和 D。然而，每个 I/O 端口的实际有效位可能不同。

• 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**: 低电压电源时上拉电阻选择

- 0: 所有引脚上拉电阻为 60kΩ @ 3V
- 1: 所有引脚上拉电阻为 15kΩ @ 3V

此位用于为低电压电源应用选择上拉电阻值。LVPU 位仅在对应该引脚上拉功能通过置高相关上拉控制位使能后有效。当上拉功能除能，此位无效。

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，只有当引脚功能选为通用 I/O 功能输入类型且单片机处于休眠或空闲模式时，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 引脚唤醒功能控制位

0: 除能
1: 使能

输入 / 输出端口控制寄存器

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

● 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 端口 x 引脚类型选择位

0: 输出
1: 输入

PxCn 位用于控制引脚类型选择。此处的“x”可为 A、B、C 和 D。然而，每个 I/O 端口的实际有效位可能不同。

输入 / 输出端口电源控制

该单片机为 PD2 和 PD5 引脚提供不同的 I/O 端口电流选项。端口电源可来自电源引脚 VDD 或 VDDIO，可通过 PMPS 寄存器中的 PMPS[1:0] 字段进行选择。若选择引脚电源来自 VDDIO 电源引脚，需通过相应的引脚共用功能选择位先将对应引脚选择为 VDDIO 功能。需注意 VDDIO 引脚上的输入电源电压必须等于或小于单片机电源电压 V_{DD}。

● PMPS 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PMPS1	PMPS0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未定义，读为“0”

Bit 1~0 **PMPS1~PMPS0**: PD2~PD5 引脚电源选择

0x: V_{DD}

1x: V_{DDIO}

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单片机某些功能造成影响。然而，引脚功能共用功能通过引脚功能选择，使得小封装单片机具有更多不同的功能。该单片机具有端口“x”输出功能选择寄存器“n”，即寄存器 P_xS_n，用于选择多功能共用引脚上的所需功能。

需要注意的一点是要确保所需引脚共用功能被正确地选中和取消。对于多数引脚共用功能，要正确地选择所需引脚共用功能，需通过对相应的引脚共用控制寄存器正确配置来实现。接着配置相应的外围功能设定从而使能这些外围功能。但是，在设置相关引脚控制字段时，一些数字输入引脚如 INT_n、xTCK 等，与对应的通用 I/O 口共用同一个引脚共用设置选项。要选择这些引脚功能，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。要正确地取消选择的引脚共用功能，应先除能外围功能，接着修改相应的引脚共用功能控制寄存器以选择其它引脚共用功能。

寄存器名称	位							
	7	6	5	4	3	2	1	0
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	—	—
PAS1	—	—	PAS15	PAS14	PAS13	PAS12	PAS11	PAS10
PCS0	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
PCS1	—	—	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
PDS0	PDS07	PDS06	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
PDS1	PDS17	PDS16	PDS15	PDS14	PDS13	PDS12	PDS11	PDS10

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	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	—	—
POR	0	0	0	0	0	0	—	—

Bit 7~6 **PAS07~PAS06:** PA3 引脚共用功能选择
00/11: PA3
01: AN8
10: VREFI

Bit 5~4 **PAS05~PAS04:** PA2 引脚共用功能选择
00/11: PA2
01: AN9
10: VREF

Bit 3~2 **PAS03~PAS02:** PA1 引脚共用功能选择
00/10/11: PA1
01: OUVPO/AN10

Bit 1~0 未定义, 读为“0”

● PAS1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	PAS15	PAS14	PAS13	PAS12	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 未定义, 读为“0”

Bit 5~4 **PAS15~PAS14:** PA6 引脚共用功能选择
00/10/11: PA6
01: OUVPI/AN11

Bit 3~2 **PAS13~PAS12:** PA5 引脚共用功能选择
00/10/11: PA5/INT1
01: STP

当此字段设置为“01”选择STP功能时, PAC寄存器中的PAC5位也需清为“0”。

Bit 1~0 **PAS11~PAS10:** PA4 引脚共用功能选择
00/01/10: PA4/INT0
11: 未定义

● PCS0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PCS07	PCS06	PCS05	PCS04	PCS03	PCS02	PCS01	PCS00
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 **PCS07~PCS06:** PC3 引脚共用功能选择
00/10/11: PC3
01: OCP1

注: 若 PCS0[7:4]=0101B, 则这两个引脚可同时作为 OCP1 输入。

Bit 5~4 **PCS05~PCS04:** PC2 引脚共用功能选择
00/10/11: PC2
01: OCP1

注: 若 PCS0[7:4]=0101B, 则这两个引脚可同时作为 OCP1 输入。

- Bit 3~2 **PCS03~PCS02:** PC1 引脚共用功能选择
00/11: PC1
01: OCP0
10: PTP
注: 若 PCS0[3:0]=0101B, 则这两个引脚可同时作为 OCP0 输入。
- Bit 1~0 **PCS01~PCS00:** PC0 引脚共用功能选择
00/10/11: PC0/PTCK
01: OCP0
注: 若 PCS0[3:0]=0101B, 则这两个引脚可同时作为 OCP0 输入。

● **PCS1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCS15	PCS14	PCS13	PCS12	PCS11	PCS10
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

- Bit 7~4 未定义, 读为 “0”
- Bit 5~4 **PCS15~PCS14:** PC6 引脚共用功能选择
00/10/11: PC6
01: VR
- Bit 3~2 **PCS13~PCS12:** PC5 引脚共用功能选择
00/10/11: PC5
01: AN13
- Bit 1~0 **PCS11~PCS10:** PC4 引脚共用功能选择
00/10/11: PC4
01: AN12

● **PDS0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PDS07	PDS06	PDS05	PDS04	PDS03	PDS02	PDS01	PDS00
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 **PDS07~PDS06:** PD3 引脚共用功能选择
00: PD3
01: AN3
10: D1-
11: AN3 + DAC1 输出
- Bit 5~4 **PDS05~PDS04:** PD2 引脚共用功能选择
00: PD2
01: AN2
10: D1+
11: AN2 + DAC0 输出
- Bit 3~2 **PDS03~PDS02:** PD1 引脚共用功能选择
00/11: PD1
01: AN1
10: VDDIO
- Bit 1~0 **PDS01~PDS00:** PD0 引脚共用功能选择
00/10/11: PD0
01: AN0

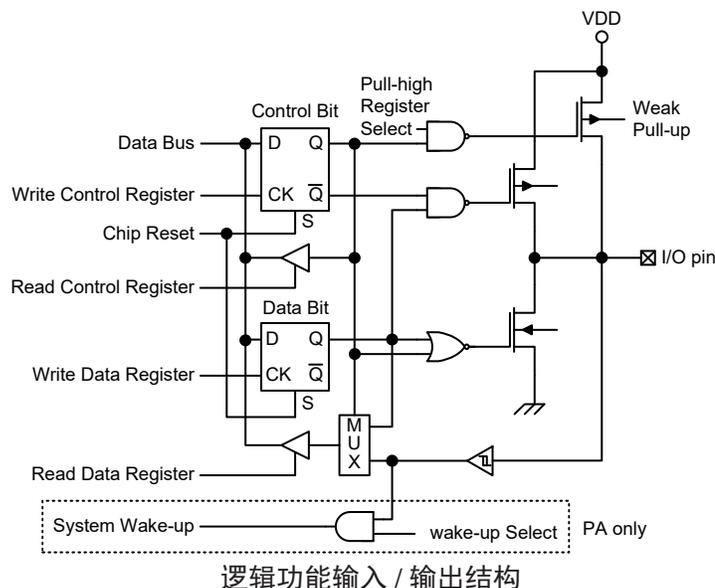
• PDS1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PDS17	PDS16	PDS15	PDS14	PDS13	PDS12	PDS11	PDS10
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 **PDS17~PDS16:** PD7 引脚共用功能选择
00/10/11: PD7
01: AN7
- Bit 5~4 **PDS15~PDS14:** PD6 引脚共用功能选择
00/10/11: PD6
01: AN6
- Bit 3~2 **PDS13~PDS12:** PD5 引脚共用功能选择
00: PD5
01: AN5
10: D2-
11: AN5+DAC3 输出
- Bit 1~0 **PDS11~PDS10:** PD4 引脚共用功能选择
00: PD4
01: AN4
10: D2+
11: AN4+DAC2 输出

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。具体输入 / 输出引脚的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

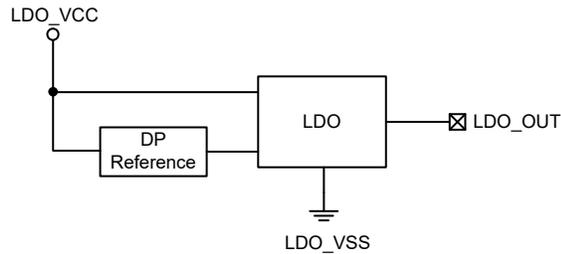
在编程中，最先要考虑的是端口的初始化。复位之后，所有的输入 / 输出数据及端口控制寄存器都将被设为逻辑高。所有输入 / 输出引脚默认为输入状态，而其电平则取决于其它相连接电路以及是否选择了上拉电阻。如果端口控制寄

寄存器某些引脚位被设定输出状态，这些输出引脚会有初始高电平输出，除非数据寄存器端口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。

低压差稳压器 – LDO

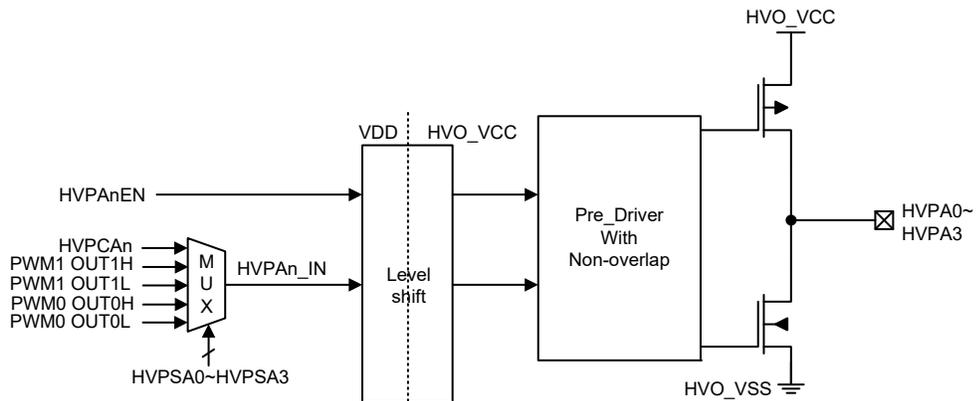
该单片机内置一个低压差稳压器 LDO。该 LDO 稳压器可以将由 LDO_VCC 引脚输入的约 5.1V~13.5V 的较高电压降低到稳定的 5V 电压，再由 LDO_OUT 引脚输出。当 LDO 输入电压 V_{IN} 在 3V 到 5.1V 范围内时，LDO 输出电压 V_{OUT} 跟随输入电压 V_{IN} 。这个较低的电压可以为内部电路或外部电路提供固定的电压源。



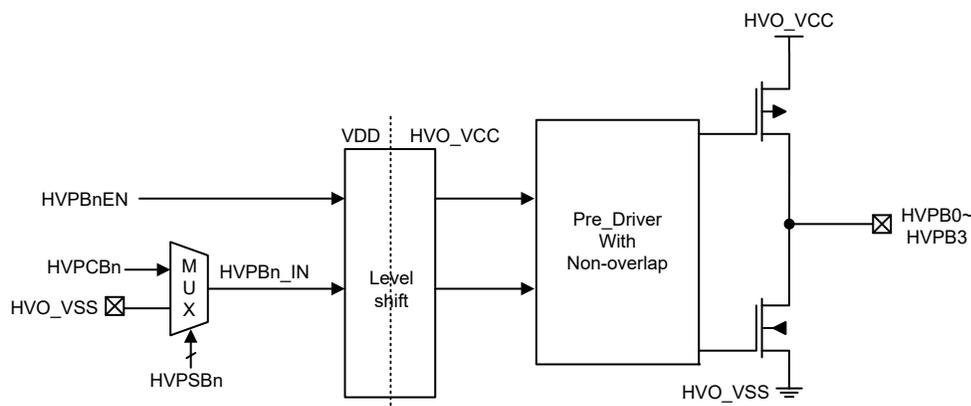
LDO 方框图

高压输出端口

该单片机提供几个 12V 高压输出脚，即 HVPxn，其中“x”表示端口 A 或 B，“n”表示具体的高压输出引脚编号。这些高压端口可转换 5V 逻辑输出信号为 12V 电压输出。



高压 HVPAn 方框图 (n=0~3)



高压 HVPB 方框图 (n=0~3)

注：HVPAn_IN/HVPBn_IN 为 HVPAn/HVPBn 输入的驱动信号。

高压输出功能描述

HVPxEN 寄存器用于控制 HVPxn 引脚为输出或浮空状态。HVPSx 寄存器用于选择 HVPSxn 输出功能。HVPSAn 输出功能可选择由 HVPCx 寄存器控制或来自内部 PWM1 OUT1H、PWM1 OUT1L、PWM0 OUT0H 或 PWM0 OUT0L 等输出信号，而 HVPSBn 输出功能可选择由 HVPCx 寄存器控制或连接到 HVO_VSS 引脚。若当前 HVPxEN 寄存器设置为输出模式，则可用于检测外部电路是否异常。

高压输出寄存器描述

高压输出端口的所有操作通过一系列寄存器控制。每个 HVPx 端口具有一组同样的寄存器，即寄存器 HVPxEN、HVPCx 和 HVPSx。

寄存器名称	位							
	7	6	5	4	3	2	1	0
HVPAEN	—	—	—	—	HVPA3EN	HVPA2EN	HVPA1EN	HVPA0EN
HVPCA	—	—	—	—	HVPCA3	HVPCA2	HVPCA1	HVPCA0
HVPSA	—	—	—	—	HVPSA3	HVPSA2	HVPSA1	HVPSA0
HVPBEN	—	—	—	—	HVPB3EN	HVPB2EN	HVPB1EN	HVPB0EN
HVPCB	—	—	—	—	HVPCB3	HVPCB2	HVPCB1	HVPCB0
HVPSB	—	—	—	—	HVPSB3	HVPSB2	HVPSB1	HVPSB0

高压输出寄存器列表

• HVPxEN 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	HVPx3EN	HVPx2EN	HVPx1EN	HVPx0EN
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

HVPxnEN: HVO 端口 x 引脚类型选择

0: 浮空

1: 输出

HVPxnEN 位用于控制高压引脚类型选择，其中“x”可以是 A 或 B。

• HVPCx 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	HVPCx3	HVPCx2	HVPCx1	HVPCx0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

HVPCxn: HVO 端口 x 引脚输出控制

0: 输出 HVO_VSS

1: 输出 HVO_VCC

HVPCxn 位用于控制引脚输出状态, 其中 “x” 可以是 A 或 B。

• HVPSA 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	HVPSA3	HVPSA2	HVPSA1	HVPSA0
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 **HVPSA3**: HVO HVPSA3 输出功能选择

0: 由 HVPCA3 控制

1: 来自 PWM1 OUT1L 输出

Bit 2 **HVPSA2**: HVO HVPSA2 输出功能选择

0: 由 HVPCA2 控制

1: 来自 PWM1 OUT1H 输出

Bit 1 **HVPSA1**: HVO HVPSA1 输出功能选择

0: 由 HVPCA1 控制

1: 来自 PWM0 OUT0L 输出

Bit 0 **HVPSA0**: HVO HVPSA0 输出功能选择

0: 由 HVPCA0 控制

1: 来自 PWM0 OUT0H 输出

• HVPSB 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	HVPSB3	HVPSB2	HVPSB1	HVPSB0
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 3~0 **HVPSB3~HVPSB0**: HVO HVPSB3~HVPSB0 输出功能选择

0: 由 HVPCBn 控制

1: 连接到 HVO_VSS 引脚

定时器模块 – TM

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

这里只介绍各种 TM 的共性，更多详细资料请参考独立的标准型和周期型定时器章节。

简介

该单片机包含两个 TM，即标准型 TM 和周期型 TM，分别命名为 STM 和 PTM。虽然性质相似，但不同 TM 特性复杂度不同。本章介绍标准型和周期型 TM 的共性，更多详细资料分别见后面各章。两种类型 TM 的特性和区别见下表。

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

TM 功能概要

STM	PTM
16-bit STM	10-bit PTM

TM 名称 / 类型参考

TM 操作

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

TM 时钟源

驱动 TM 计数器的时钟源很多。通过设置 xTM 控制寄存器的 xTCK2~xTCK0 位选择所需的时钟源，其中的“x”可为 S 或 P，代表不同的 TM 类型。该时钟源来自系统时钟 f_{sys} 或内部高速时钟 f_H 或 f_{TBC} 时钟源或外部 xTCK 引脚输入时钟。xTCK 引脚时钟源允许外部信号作为 TM 时钟源或用于事件计数。

TM 中断

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

TM 外部引脚

无论哪种类型的 TM，都有一个 TM 输入引脚 xTCK。通过设置 xTMC0 寄存器中的 xTCK2~xTCK0 位，选择 TM 功能并将该引脚作为 TM 时钟源输入脚。外部时钟源可通过该引脚来驱动内部 TM。TM 引脚可选择上升沿有效或下降沿有效。当 PTM 或 STM 工作在单脉冲输出模式，xTCK 引脚还可用作外部触发输入引脚。

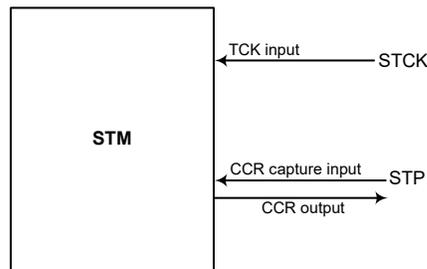
若 TM 工作在捕捉输入模式，另外一个 TM 引脚 xTP 也可作为捕捉输入引脚，并可通过设置 xTMC1 寄存器中的 xTIO1~xTIO0 位来选择有效边沿为上升沿，下降沿或双沿。对于周期型 TM，当 PTM 工作在捕捉输入模式，PTCK 引脚也可同 PTP 引脚一样通过寄存器选择作为捕捉输入引脚。

当设置 xTM 工作在其它模式(捕捉输入模式除外)，xTP 为 TM 功能的输出引脚。若工作在比较匹配输出模式且比较匹配发生时，xTP 引脚会由 TM 控制切换到高电平或低电平或翻转。若工作在 PWM 输出模式，xTP 引脚被 TM 用来产生 PWM 输出波形。

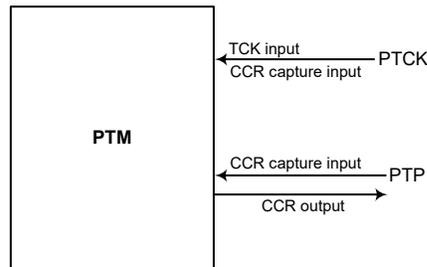
由于 TM 输入和输出引脚与其它功能共用同一引脚，在使用 TM 功能前，用户需要先通过引脚共用功能选择寄存器对相应位进行设置以选择 TM 引脚功能。更多引脚共用功能选择详见引脚共用功能章节。

STM		PTM	
输入	输出	输入	输出
STCK, STP	STP	PTCK, PTP	PTP

TM 外部引脚



STM 功能引脚控制方框图

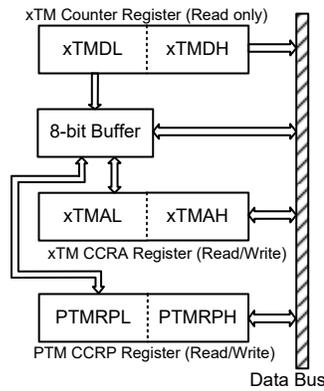


PTM 功能引脚控制方框图

编程注意事项

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

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

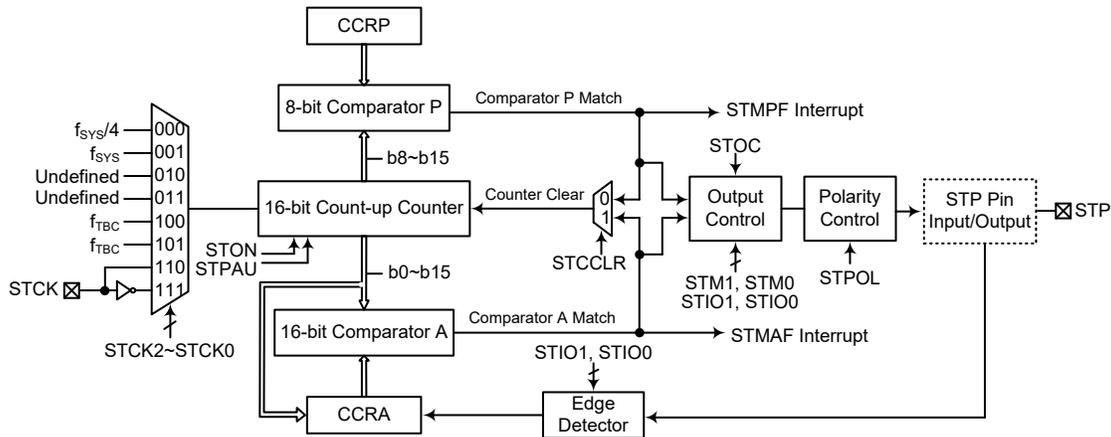


读写流程如下步骤所示：

- 写数据至 CCRA 或 PTM CCRP
 - ◆ 步骤 1. 写数据至低字节寄存器 xTMAL 或 PTMRPL
 - 注意，此时数据仅写入 8-bit 缓存器。
 - ◆ 步骤 2. 写数据至高字节寄存器 xTMAH 或 PTMRPH
 - 注意，此时数据直接写入高字节寄存器，同时锁存在 8-bit 缓存器中的数据写入低字节寄存器。
- 由计数器寄存器和 CCRA 或 PTM CCRP 中读取数据
 - ◆ 步骤 1. 由高字节寄存器 xTMDH、xTMAH 或 PTMRPH 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 xTMDL、xTMAL 或 PTMRPL 读取数据
 - 注意，此时读取 8-bit 缓存器中的数据。

标准型 TM – STM

标准型 TM 包括 5 种工作模式，即比较匹配输出、定时 / 事件计数器、捕捉输入、单脉冲输出和 PWM 输出模式。该标准型 TM 可由两个外部输入引脚控制且可驱动一个外部输出引脚。



标准型 TM 方框图

标准型 TM 操作

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

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

标准型 TM 寄存器介绍

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

寄存器名称	位							
	7	6	5	4	3	2	1	0
STMC0	STPAU	STCK2	STCK1	STCK0	STON	—	—	—
STMC1	STM1	STM0	STIO1	STIO0	STOC	STPOL	STDPX	STCCLR
STMDL	D7	D6	D5	D4	D3	D2	D1	D0
STMDH	D15	D14	D13	D12	D11	D10	D9	D8
STMAL	D7	D6	D5	D4	D3	D2	D1	D0
STMAH	D15	D14	D13	D12	D11	D10	D9	D8
STMRP	D7	D6	D5	D4	D3	D2	D1	D0

16-bit 标准型 TM 寄存器列表

• **STMC0 寄存器**

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

Bit 7 **STPAU**: STM 计数器暂停控制位

- 0: 运行
- 1: 暂停

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

Bit 6~4 **STCK2~STCK0**: 选择 STM 计数时钟位

- 000: $f_{SYS}/4$
- 001: f_{SYS}
- 010: 未定义
- 011: 未定义
- 100: f_{TBC}
- 101: f_{TBC}
- 110: STCK 上升沿时钟
- 111: STCK 下降沿时钟

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

Bit 3 **STON**: STM 计数器 On/Off 控制位

- 0: Off
- 1: On

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

Bit 2~0 未定义，读为“0”

• **STMC1 寄存器**

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

Bit 7~6 **STM1~STM0**: STM 工作模式选择位

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

这两位设置 STM 需要的工作模式。为了确保操作可靠，STM 应在 STM1 和 STM0 位有任何改变前先关掉。在定时 / 计数器模式，STM 输出引脚状态未定义。

Bit 5~4 **STIO1~STIO0**: STM 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 / 单脉冲输出模式

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

捕捉输入模式

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

定时 / 计数器模式

未使用

此两位用于决定在满足特定条件时 STM 外部引脚如何改变状态。这两位值的选择取决于 STM 运行在何种工作模式。

在比较匹配输出模式下，STIO1 和 STIO0 位决定当从比较器 A 比较匹配输出发生时 STM 输出脚 STP 如何改变状态。当从比较器 A 比较匹配输出发生时 STP 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。STP 输出脚的初始值通过 STMC1 寄存器的 STOC 位设置取得。注意，由 STIO1 和 STIO0 位得到的输出电平必须与通过 STOC 位设置的初始值不同，否则当比较匹配发生时，STP 输出脚将不会发生变化。在 STP 输出脚改变状态后，通过 STON 位由低到高电平的转换复位至初始值。

在 PWM 输出模式，STIO1 和 STIO0 决定比较匹配条件发生时怎样改变 STM 输出脚的状态。PWM 输出功能通过这两位的变化进行更新。仅在 STM 关闭时改变 STIO1 和 STIO0 位的值是很有必要的。若在 STM 运行时改变 STIO1 和 STIO0 的值，PWM 输出的值将无法预料。

Bit 3 **STOC: STM 输出脚 STP 输出控制位**

比较匹配输出模式

- 0: 初始低
- 1: 初始高

PWM 输出模式 / 单脉冲输出模式

- 0: 低有效
- 1: 高有效

这是 STM 输出脚输出控制位。它取决于 STM 此时正运行于比较匹配输出模式还是 PWM 输出模式 / 单脉冲输出模式。若 STM 处于定时 / 计数器模式，则其不受影响。在比较匹配输出模式时，其决定比较匹配发生前 STM 输出脚 STP 的逻辑电平值。在 PWM 输出模式，其决定输出信号是高有效还是低有效。在单脉冲输出模式时，其决定当 STON 位发生从低到高的转变时，STM 输出引脚的逻辑电平。

Bit 2 **STPOL: STP 输出脚输出极性控制位**

- 0: 同相
- 1: 反相

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

Bit 1 **STDPX: STM PWM 周期 / 占空比控制位**

- 0: CCRP – 周期；CCRA – 占空比
- 1: CCRP – 占空比；CCRA – 周期

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

Bit 0 **STCCLR: 选择 STM 计数器清零条件位**

- 0: STM 比较器 P 匹配
- 1: STM 比较器 A 匹配

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

• STMDL 寄存器

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

Bit 7~0 **D7~D0:** STM 计数器低字节寄存器 bit 7~bit 0
 STM 16-bit 计数器 bit 7~bit 0

• STMDH 寄存器

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

Bit 7~0 **D15~D8:** STM 计数器高字节寄存器 bit 7~bit 0
 STM 16-bit 计数器 bit 15~bit 8

• STMAL 寄存器

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

Bit 7~0 **D7~D0:** STM CCRA 低字节寄存器 bit 7~bit 0
 STM 16-bit CCRA bit 7~bit 0

• STMAH 寄存器

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

Bit 7~0 **D15~D8:** STM CCRA 高字节寄存器 bit 7~bit 0
 STM 16-bit CCRA bit 15~bit 8

• STMRP 寄存器

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

Bit 7~0 **D7~D0:** STM CCRP 8-bit 寄存器，与 STM 计数器 bit 15~bit 8 比较
 比较器 P 匹配周期 =
 0: 65536 个 STM 时钟周期
 1~255: (1~255)×256 个 STM 时钟周期

此八位设定内部 CCRP 8-bit 寄存器的值，然后与内部计数器的高八位进行比较。如果 STCCLR 位设为 0 时，此比较结果可用于清零内部计数器。STCCLR 位设为低，CCRP 比较匹配结果将重置内部计数器。由于 CCRP 只与计数器高八位比较，比较结果是 256 时钟周期的倍数。CCRP 被清零时，实际上会使得计数器在最大值溢出。

标准型 TM 工作模式

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

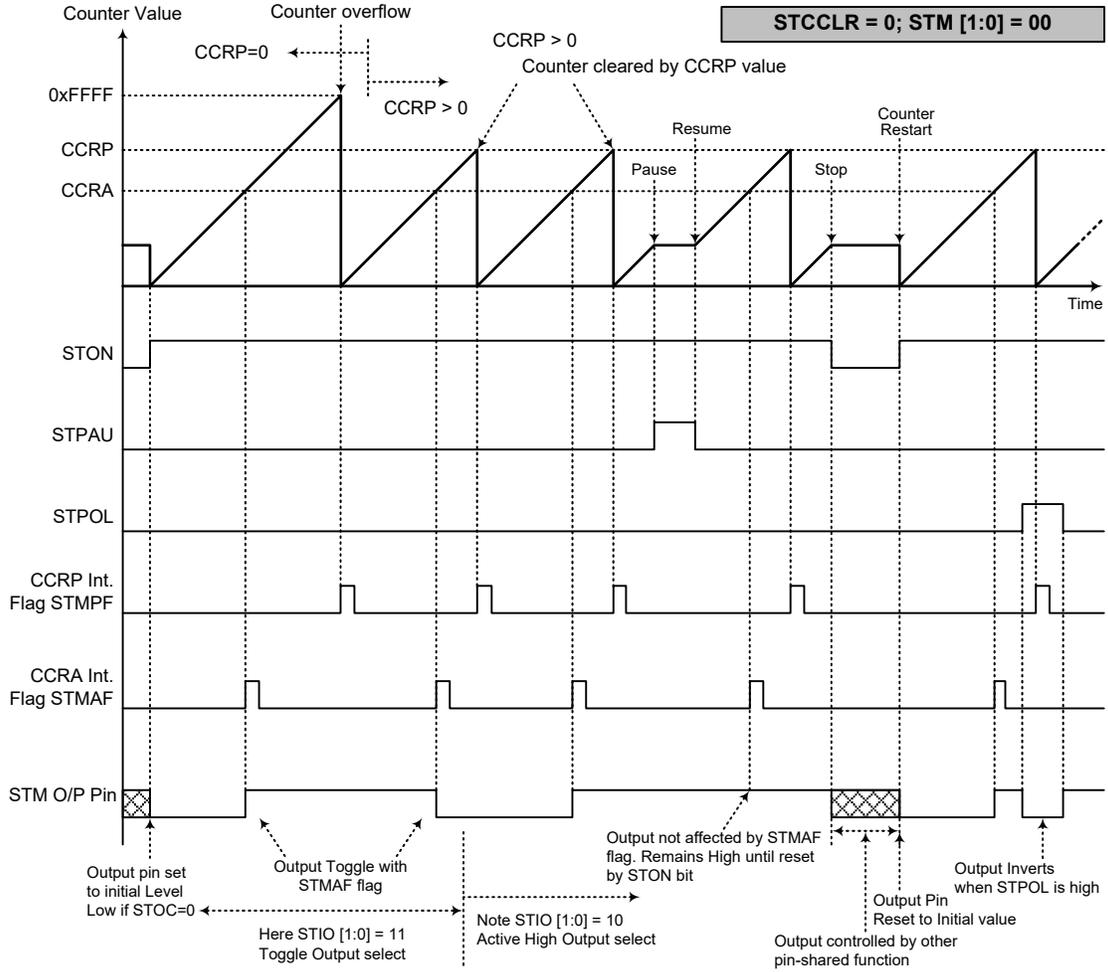
比较匹配输出模式

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

如果 STMC1 寄存器的 STCCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅产生 STMAF 中断请求标志。所以当 STCCLR 为高时，不会产生 STMPF 中断请求标志。在比较匹配输出模式下，CCRA 不能设为“0”。

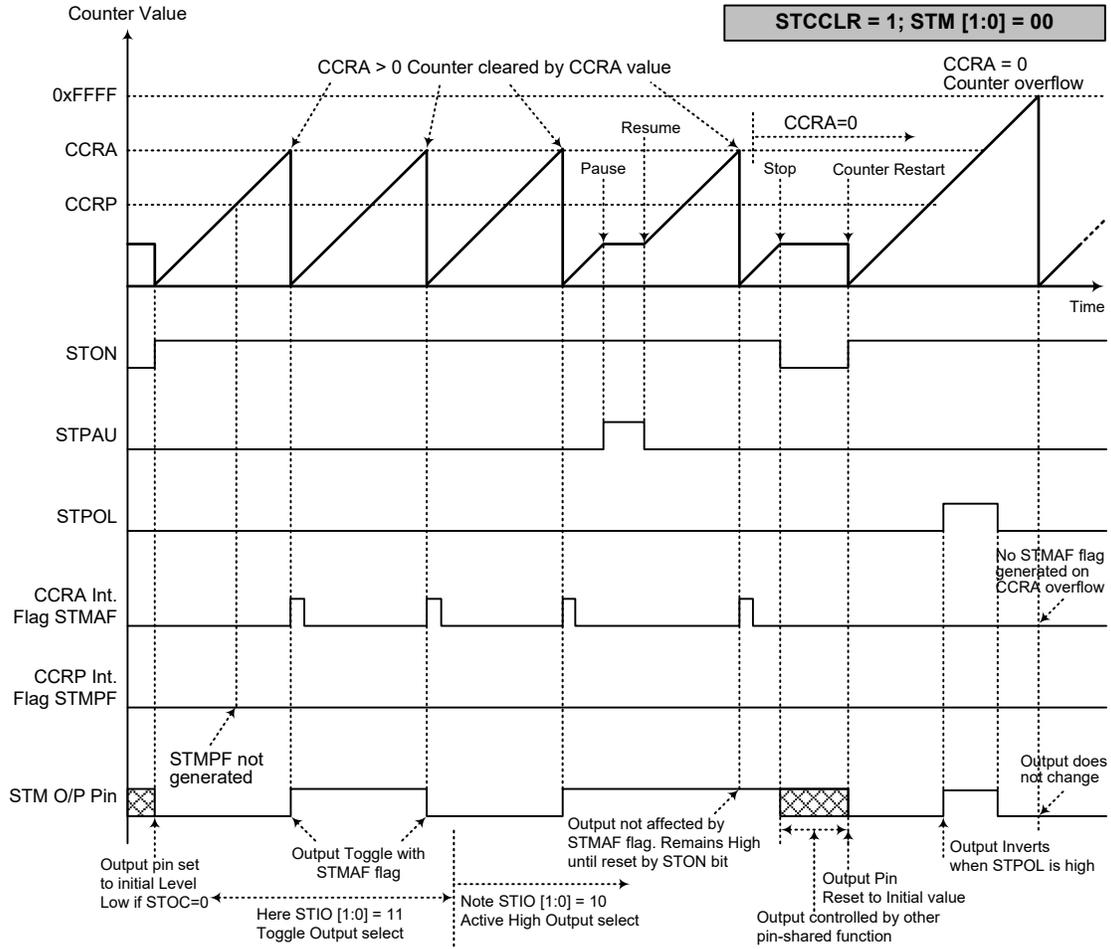
如果 CCRA 位都清除为零，当计数器的值达到 16 位最大值 FFFFH 时将溢出，但此时不会产生 STMAF 中断请求标志。

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



比较匹配输出模式 – STCCLR=0

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



比较匹配输出模式 – STCCLR=1

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

定时 / 计数器模式

为使 STM 工作在此模式，STMC1 寄存器中的 STM1 和 STM0 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 STM 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 STM 外部引脚，这些引脚可通过引脚共用功能选择寄存器相应位设置用作普通 I/O 或其它功能。

PWM 输出模式

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

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

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

● 16-bit STM, PWM 输出模式, 边沿对齐模式, STDPX=0

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

若 $f_{sys}=16\text{MHz}$ ，STM 时钟源选择 $f_{sys}/4$ ，CCRP=2，CCRA=128，

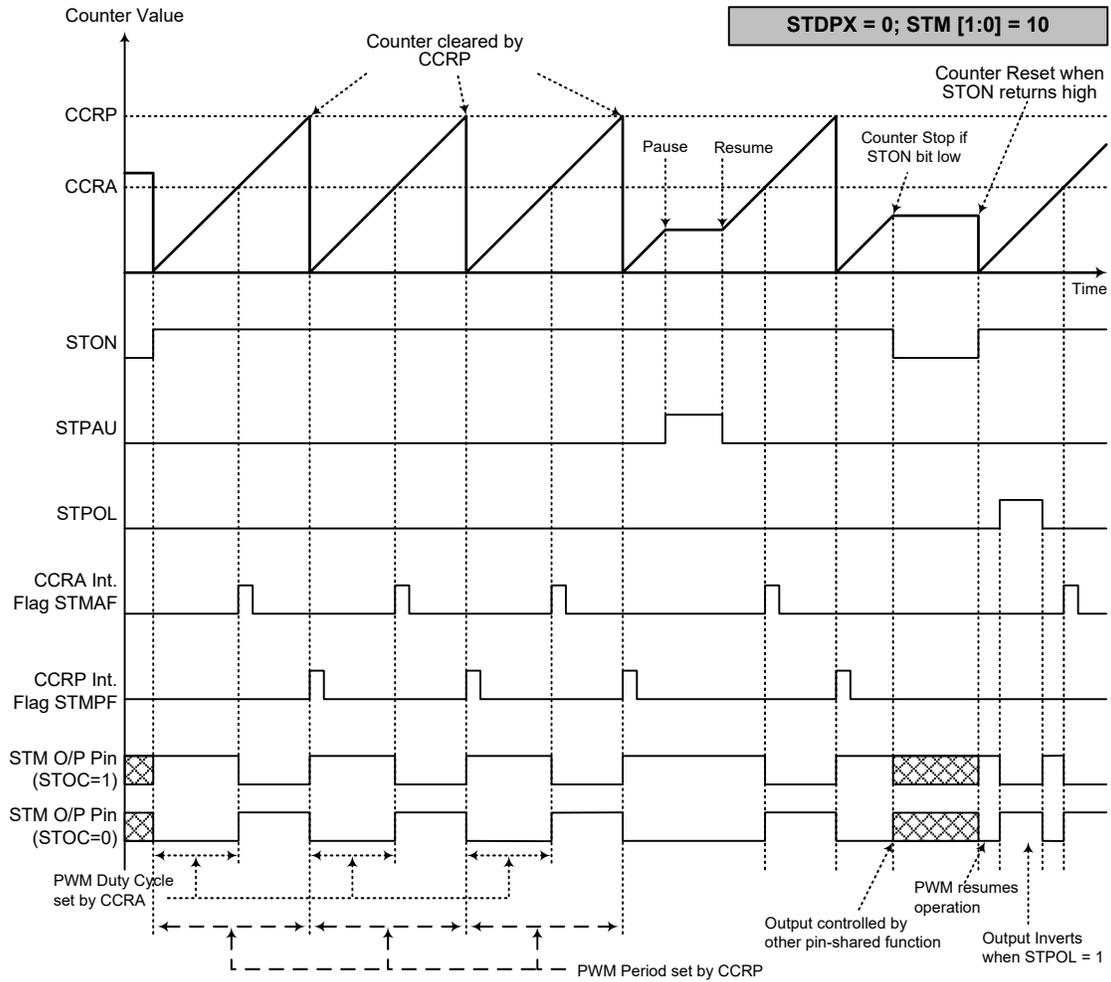
STM PWM 输出频率 = $(f_{sys}/4)/(2 \times 256) = f_{sys}/2048 = 7.8125\text{kHz}$ ， $duty = 128/(2 \times 256) = 25\%$ 。

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

● 16-bit STM, PWM 输出模式, 边沿对齐模式, STDPX=1

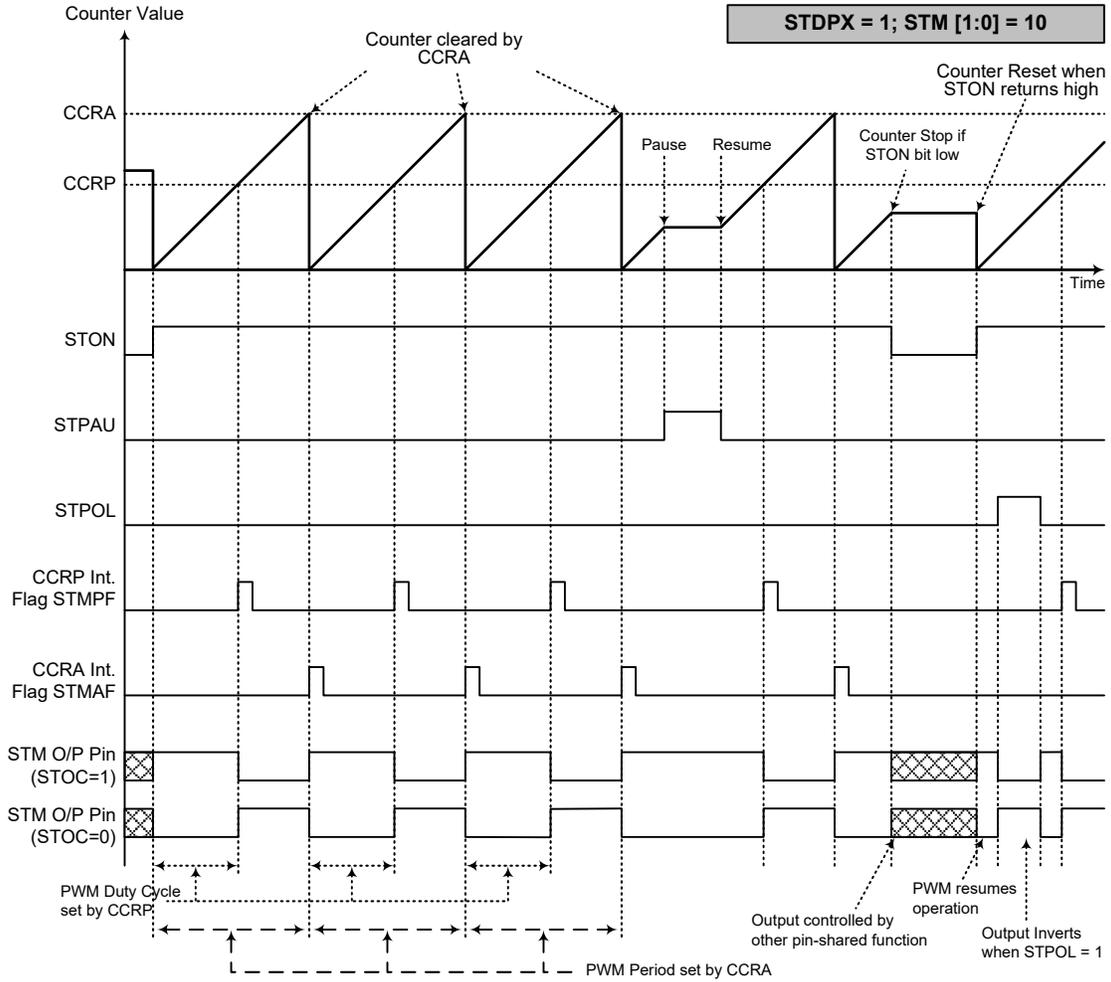
CCRP	1~255	0
Period	CCRA	
Duty	CCRP×256	65536

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



PWM 输出模式 – STDPX=0

- 注: 1. STDPX=0, 计数器由 CCRP 清除
 2. 计数器清零并设置 PWM 周期
 3. 即使当 STIO[1:0]=00 或 01, 内部 PWM 功能不变
 4. STCCLR 位对 PWM 操作无影响



PWM 输出模式 – STDPX=1

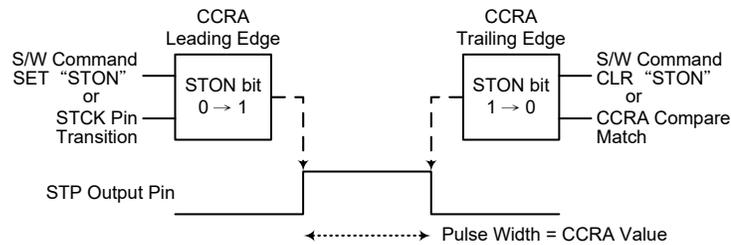
- 注: 1. STDPX=1, 计数器由 CCRA 清除
 2. 计数器清零设置 PWM 周期
 3. 即使当 STIO[1:0]=00 或 01, 内部 PWM 功能不变
 4. STCCLR 对 PWM 操作无影响

单脉冲输出模式

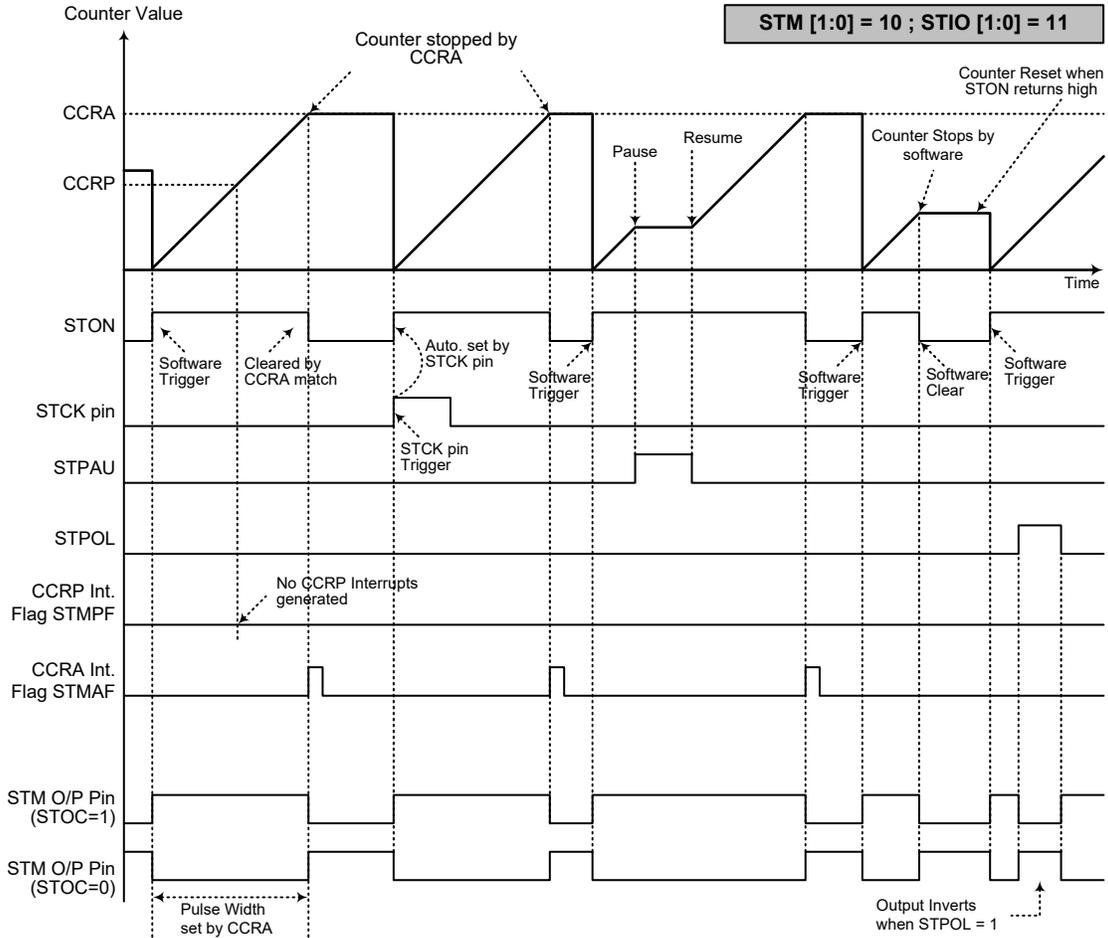
为使 STM 工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“10”，同时 STIO1 和 STIO0 位需要设置为“11”。正如模式名所言，单脉冲输出模式，在 STM 输出脚将产生一个脉冲输出。

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

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



单脉冲产生示意图



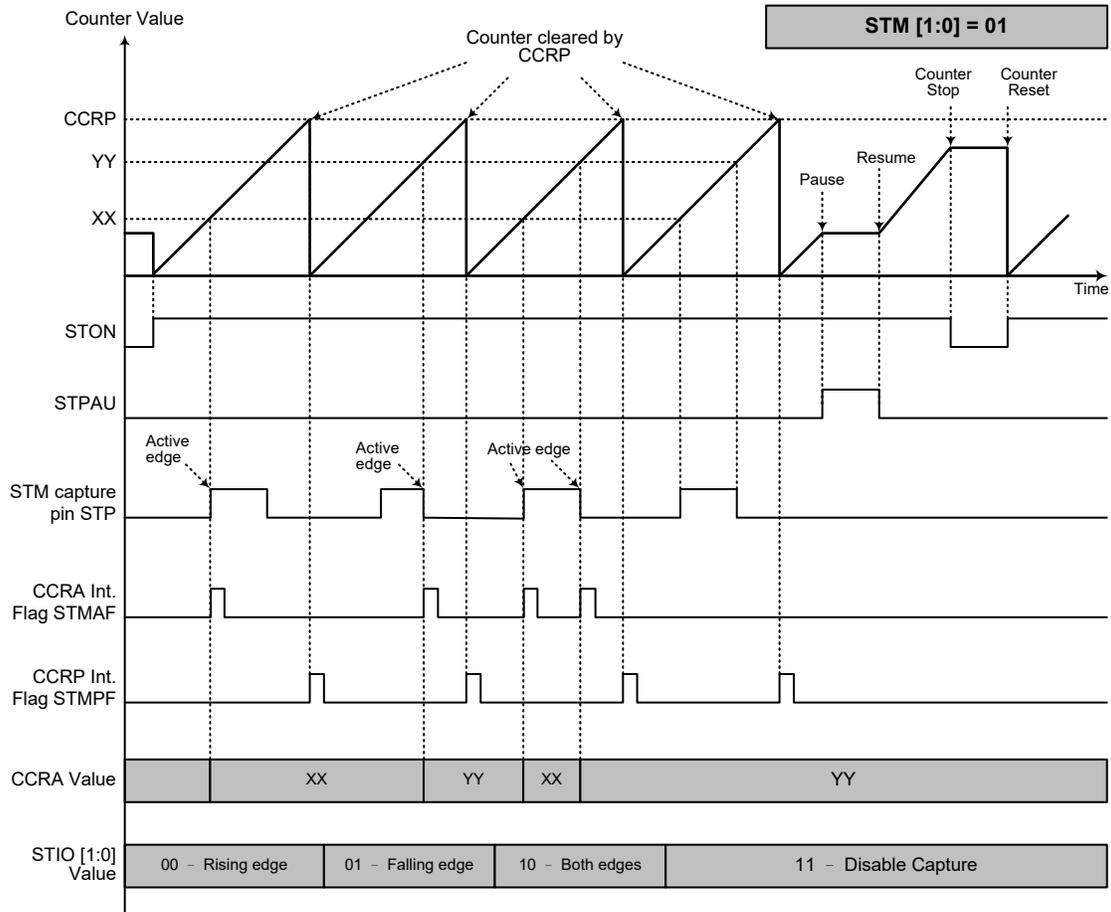
单脉冲输出模式

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

捕捉输入模式

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

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

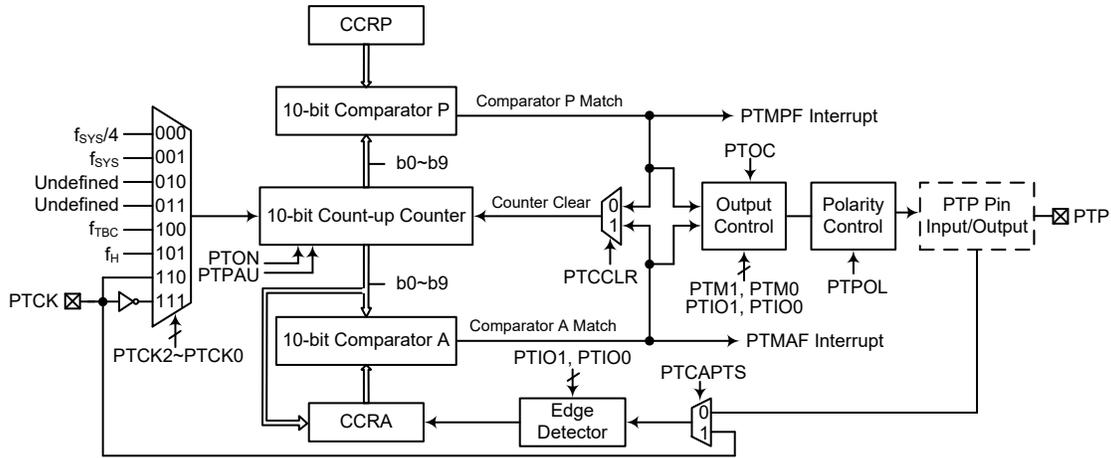


捕捉输入模式

- 注：1. STM1, STM0=01 并通过 STIO1 和 STIO0 位设置有效边沿
 2. STM 捕捉输入脚的有效边沿将计数器的值转移到 CCRA 中
 3. STCCLR 位未使用
 4. 无输出功能 – STOC 和 STPOL 位未使用
 5. 计数器值由 CCRP 决定，在 CCRP 为“0”时，计数器计数值可达最大

周期型 TM – PTM

周期型 TM 包括 5 种工作模式，即比较匹配输出、定时 / 事件计数器、捕捉输入、单脉冲输出和 PWM 输出模式。该周期型 TM 可由两个外部输入引脚控制且可驱动一个外部输出引脚。



周期型 TM 方框图

周期型 TM 操作

周期型 TM 是 10 位宽度。周期型 TM 核心是一个由用户选择的内部或外部时钟源驱动的 10 位向上计数器，它还包括两个内部比较器即比较器 A 和比较器 P。这两个比较器将计数器的值与 CCRA 和 CCRP 寄存器中的值进行比较。CCRP 和 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 PTON 位发生上升沿跳变清除计数器。此外，计数器溢出或比较匹配也会自动清除计数器。上述条件发生时，通常情况会产生 PTM 中断信号。周期型 TM 可工作在不同的模式，可由包括来自输入脚的不同时钟源驱动，也可以控制输出脚。所有工作模式的设定都是通过设置相关寄存器来实现的。

周期型 TM 寄存器介绍

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

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

10-bit 周期型 TM 寄存器列表

● PTMC0 寄存器

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

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

Bit 6~4 **PTCK2~PTCK0**: 选择 PTM 计数时钟位
 000: $f_{SYS}/4$
 001: f_{SYS}
 010: 未定义
 011: 未定义
 100: f_{TBC}
 101: f_H
 110: PTCK 上升沿
 111: PTCK 下降沿
 此三位用于选择 PTM 的时钟源。外部引脚时钟源能被选择在上升沿或下降沿有效。 f_{SYS} 是系统时钟, f_H 和 f_{TBC} 是其它的内部时钟源, 细节方面请参考振荡器章节。

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

Bit 2~0 未定义, 读为“0”

● PTMC1 寄存器

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

Bit 7~6 **PTM1~PTM0**: PTM 工作模式选择位
 00: 比较匹配输出模式
 01: 捕捉输入模式
 10: PWM 输出模式或单脉冲输出模式
 11: 定时 / 计数器模式
 这两位设置 PTM 需要的工作模式。为了确保操作可靠, PTM 应在 PTM1 和 PTM0 位有任何改变前先关掉。在定时 / 计数器模式, PTM 输出引脚未定义。

Bit 5~4 **PTIO1~PTIO0**: PTM 外部引脚 (PTP 或 PTCK) 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 / 单脉冲输出模式

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

捕捉输入模式

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

定时 / 计数器模式

未使用

此两位用于决定在一定条件达到时 PTM 输出如何改变状态。这两位值的选择取决于 PTM 运行在何种模式下。

在比较匹配输出模式下，PTIO1 和 PTIO0 位决定当从比较器 A 比较匹配输出发生时 PTM 输出脚如何改变状态。当从比较器 A 比较匹配输出发生时 PTM 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。PTM 输出脚的初始值通过 PTMC1 寄存器的 PTOC 位设置取得。注意，由 PTIO1 和 PTIO0 位得到的输出电平必须与通过 PTOC 位设置的初始值不同，否则当比较匹配发生时，PTM 输出脚将不会发生变化。在 PTM 输出脚改变状态后，通过 PTON 位由低到高电平的转换复位至初始值。

在 PWM 输出模式，PTIO1 和 PTIO0 用于决定比较匹配条件发生时怎样改变 PTM 输出脚的状态。PWM 输出功能通过这两位的变化进行更新。仅在 PTM 关闭时改变 PTIO1 和 PTIO0 位的值是很有必要的。若在 PTM 运行时改变 PTIO1 和 PTIO0 的值，PWM 输出的值是无法预料的。

Bit 3 **PTOC: PTM PTP 输出控制位**

比较匹配输出模式

- 0: 初始低
- 1: 初始高

PWM 输出模式 / 单脉冲输出模式

- 0: 低有效
- 1: 高有效

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

Bit 2 **PTPOL: PTM PTP 输出极性控制位**

- 0: 同相
- 1: 反相

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

Bit 1 **PTCAPTS: 选择 PTM 捕捉触发源**

- 0: 来自 PTP 引脚
- 1: 来自 PTCK 引脚

Bit 0 **PTCCLR: 选择 PTM 计数器清零条件位**

- 0: PTM 比较器 P 匹配
- 1: PTM 比较器 A 匹配

此位用于选择清除计数器的方法。周期型 PTM 包括两个比较器 – 比较器 A 和比较器 P，两者都可以用作清除内部计数器。PTCCLR 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 比较匹配发生时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。PTCCLR 位在 PWM 输出模式、单脉冲输出或输入捕捉模式时未使用。

• PTMDL 寄存器

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

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

• PTMDH 寄存器

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

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

• PTMAL 寄存器

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

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

• PTMAH 寄存器

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

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

• PTMRPL 寄存器

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

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

● PTMRPH 寄存器

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

Bit 7~2 未定义，读为“0”

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

周期型 TM 工作模式

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

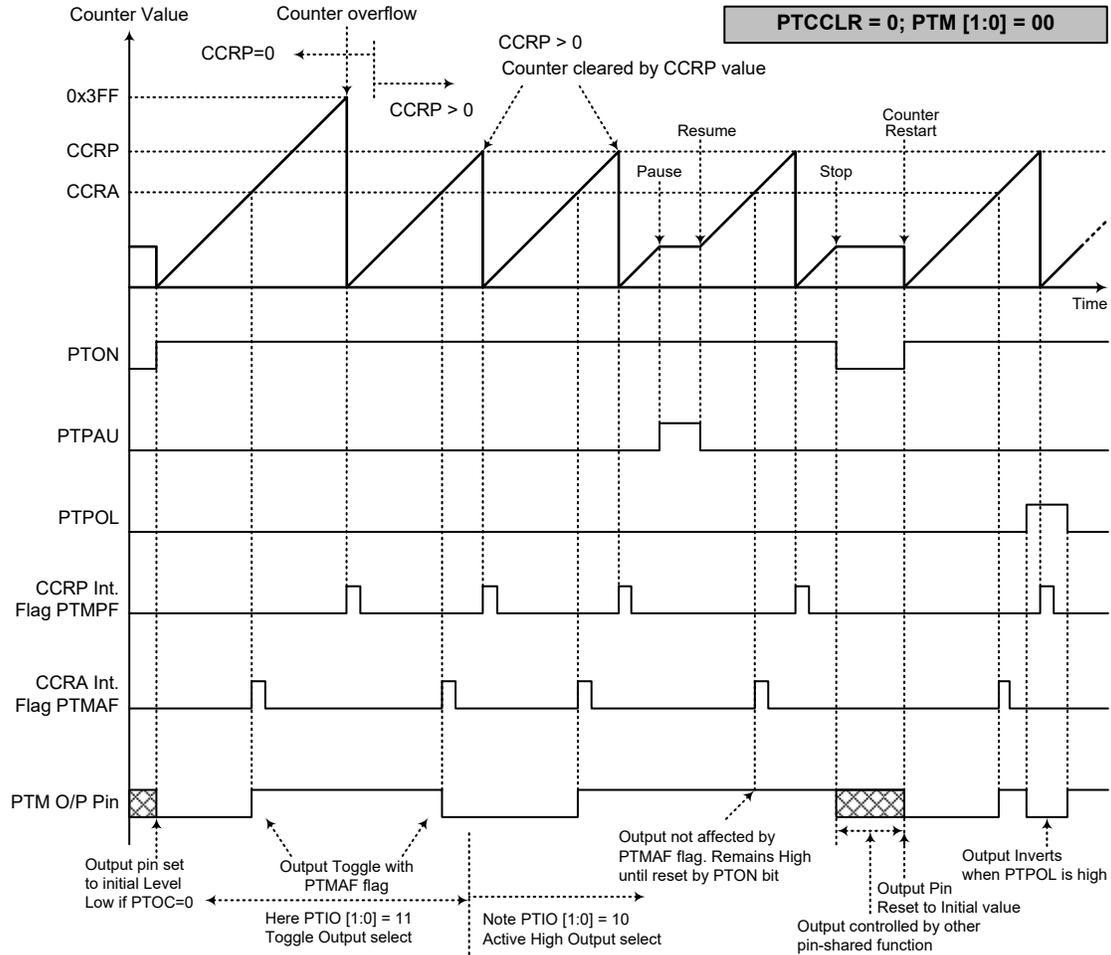
比较匹配输出模式

为使 PTM 工作在此模式，PTMC1 寄存器的 PTM1 和 PTM0 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 PTCCLR 位为低，有两种方法清除计数器。一种是比较器 P 比较匹配发生，另一种是 CCRP 所有位设置为零并使得计数器溢出。此时，比较器 A 和比较器 P 的请求标志位 PTMAF 和 PTMPF 将分别置起。

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

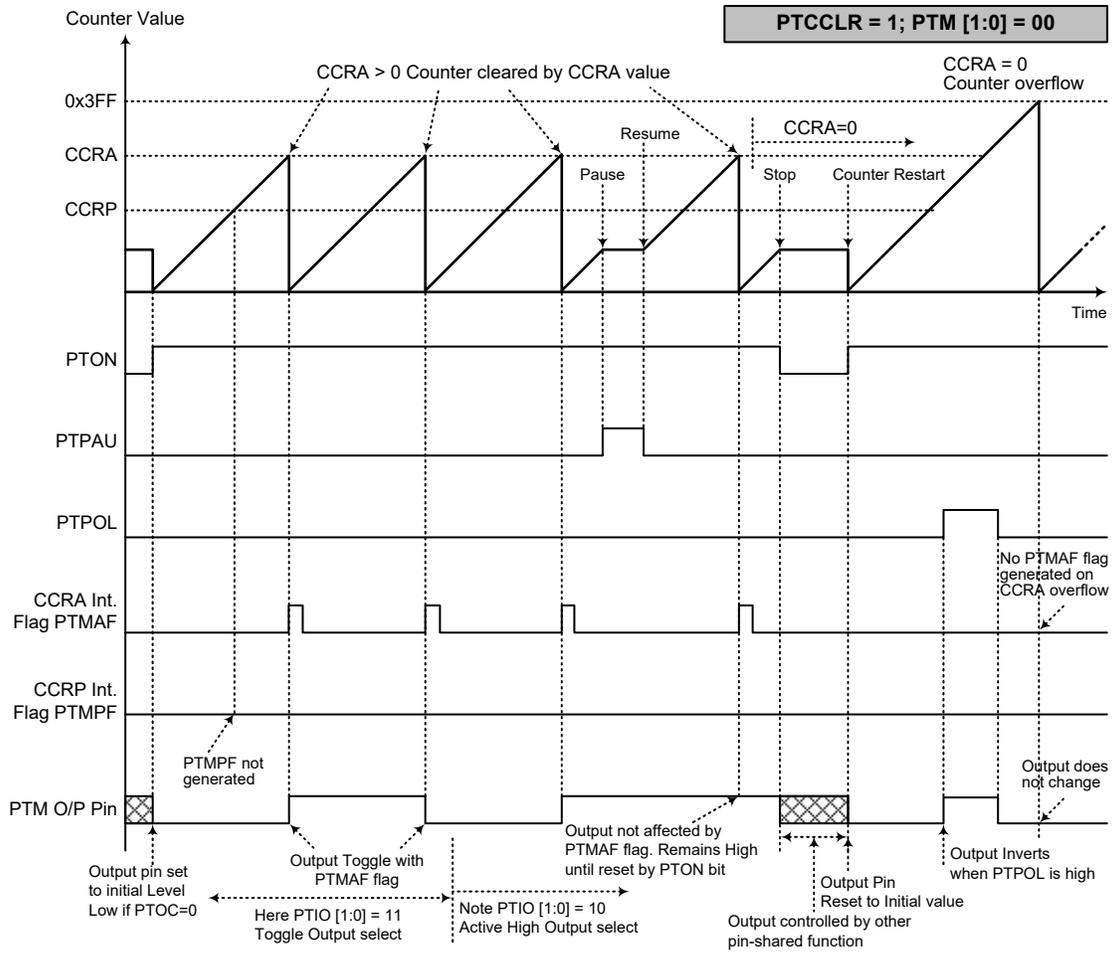
如果 CCRA 位都清除为零，当计数器的值达到 10 位最大值 3FFH 时将溢出，但此时不会产生 PTMAF 中断请求标志。

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



比较器匹配输出模式 - PTCCLR=0

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



比较器匹配输出模式 - PTCCCLR=1

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

定时 / 计数器模式

为使 PTM 工作在此模式，PTMC1 寄存器的 PTM1 和 PTM0 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 PTM 输出脚未使用。因此，比较匹配输出模式中的描述和时序图可以适用于此功能。该模式中未使用 PTM 外部引脚，这些引脚可通过引脚共用功能选择寄存器相应位设置用作普通 I/O 或其它功能。

PWM 输出模式

为使 PTM 工作在此模式，PTMC1 寄存器的 PTM1 和 PTM0 位需要设置为“10”，且 PTIO1 和 PTIO0 位也需要设置为“10”。PTM 的 PWM 功能在马达控制，加热控制，照明控制等方面十分有用。给 PTM 输出脚提供一个频率固定但占空比可调的信号，将产生一个有效值等于 DC 均方根的 AC 方波。

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

当比较器 A 或比较器 P 比较匹配发生时，CCRA 和 CCRP 中断标志位分别产生。PTMC1 寄存器的 PTOC 位选择 PWM 波形的极性，PTIO1 和 PTIO0 位使能 PWM 输出或强制 PTM 输出脚为高电平或低电平。PTPOL 位用于 PWM 输出波形的极性反相控制。

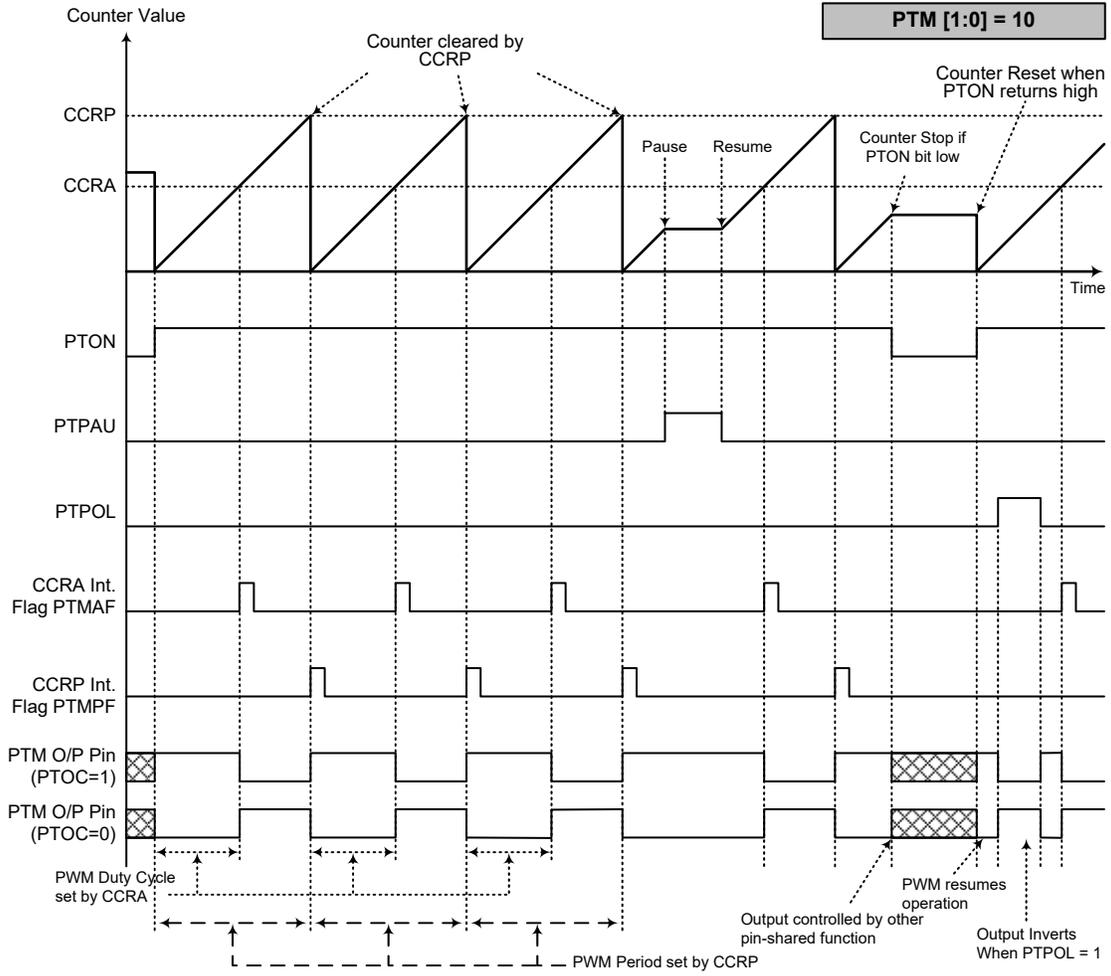
● 10-bit PTM, PWM 输出模式, 边沿对齐模式

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

若 $f_{sys}=16\text{MHz}$ ，PTM 时钟源选择 $f_{sys}/4$ ， $CCRP=512$ 且 $CCRA=128$ ，

PTM PWM 输出频率 = $(f_{sys}/4)/512=f_{sys}/2048=7.8125\text{kHz}$ ， $duty=128/512=25\%$ 。

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



PWM 输出模式

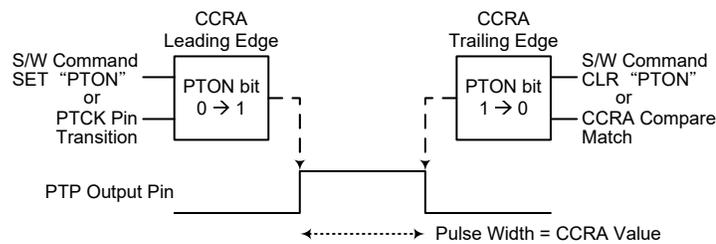
- 注：1. 计数器由 CCRP 清除
2. 计数器清零设置 PWM 周期
3. 当 PTIO[1:0]=00 或 01，PWM 功能不变
4. PTCCLR 位对 PWM 功能无影响

单脉冲输出模式

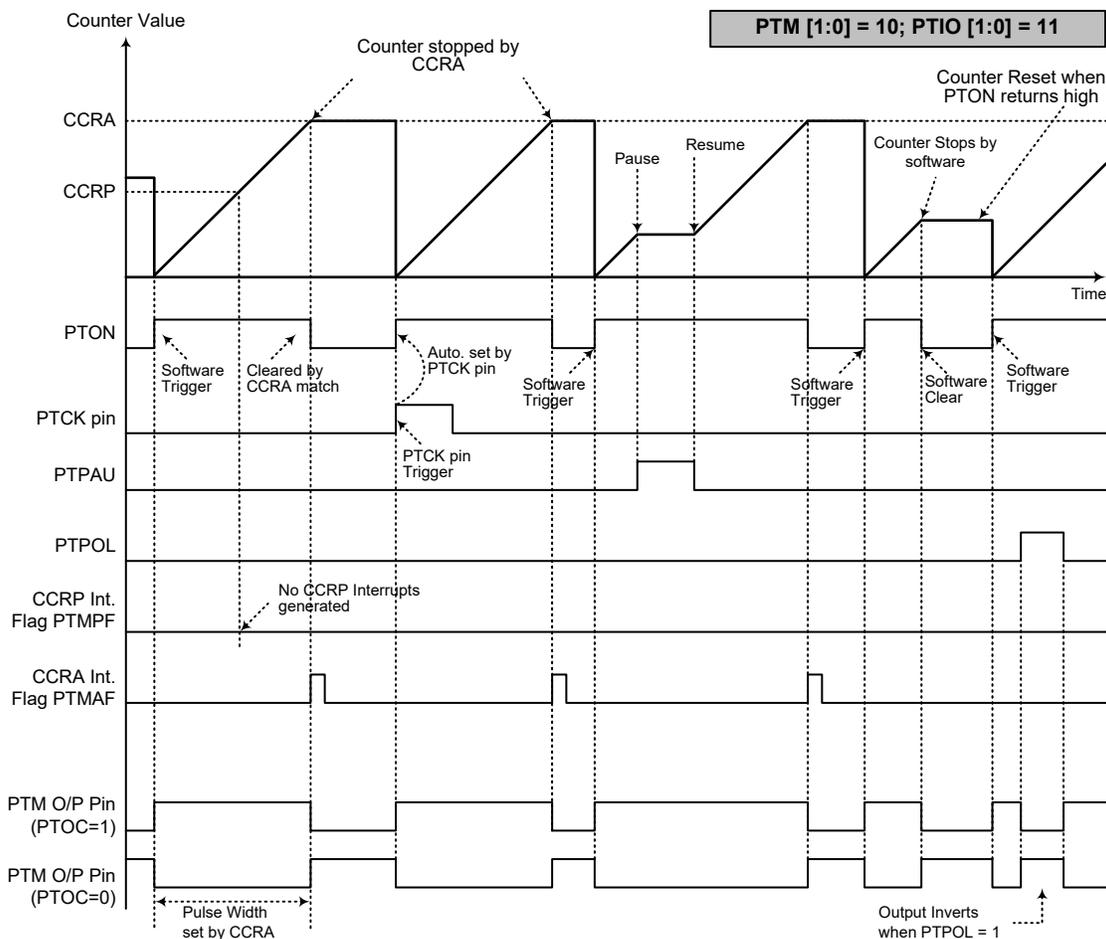
为使 PTM 工作在此模式，PTMC1 寄存器中的 PTM1 和 PTM0 位需要设置为“10”，并且相应的 PTIO1 和 PTIO0 需要设置为“11”。正如模式名所言，单脉冲输出模式，在 PTM 输出脚将产生一个脉冲输出。

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

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



单脉冲产生示意图



单脉冲输出模式

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

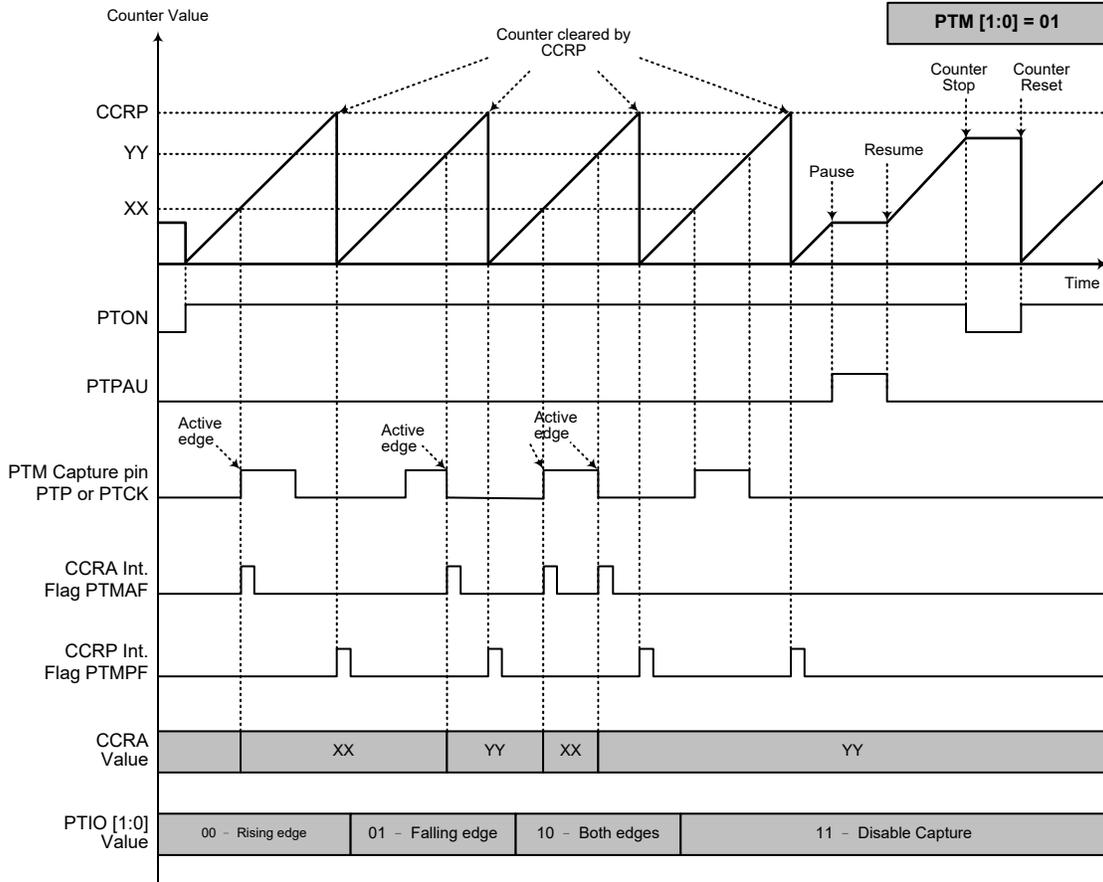
捕捉输入模式

为使 PTM 工作在此模式，PTMC1 寄存器的 PTM1 和 PTM0 位需要设置为“01”。此模式使能外部信号捕捉并保存内部计数器当前值，因此被用于诸如脉冲宽度测量的应用中。PTP 或 PTCK 引脚上的外部信号，通过设置 PTMC1 寄存器的 PTCAPTS 位选择。可通过设置 PTMC1 寄存器的 PTIO1 和 PTIO0 位选择有效边沿类型，即上升沿，下降沿或双沿有效。通过应用程序将 PTON 位由低到高转变时，计数器启动。

当 PTP 或 PTCK 引脚出现有效边沿转换时，计数器当前值被锁存到 CCRA 寄存器，并产生 PTM 中断。无论 PTP 或 PTCK 引脚发生哪种边沿转换，计数器将继续工作直到 PTON 位发生下降沿跳变。当 CCRP 比较匹配发生时计数器复位至零；CCRP 的值通过这种方式控制计数器的最大值。当比较器 P CCRP 比较匹配发生时，也会产生 PTM 中断。记录 CCRP 溢出中断信号的值可以测量长脉宽。通过设置 PTIO1 和 PTIO0 位选择 PTP 或 PTCK 引脚为上升沿，下降沿或双沿有效。如果 PTIO1 和 PTIO0 位都设置为高，无论 PTP 或 PTCK 引脚发

生哪种边沿转换都不会产生捕捉操作，但计数器仍会继续运行。

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



捕捉输入模式

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

A/D 转换器

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

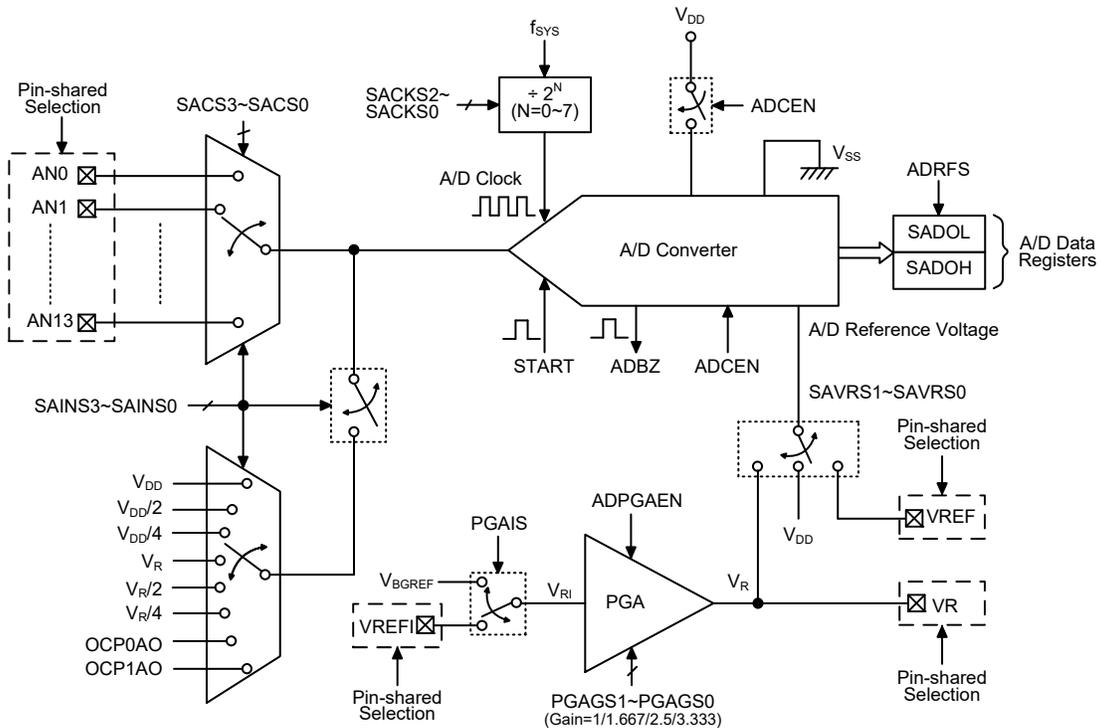
A/D 转换器简介

该单片机包含了多通道的 A/D 转换器，它们可以直接接入外部模拟信号，如来自传感器或其它控制信号，并直接将这些信号转换成 12-bit 数字量。也可对内部模拟信号，如 OCP0/OCP1 OPA 输出信号等进行 A/D 转换。

选择转换外部或内部模拟信号由 SAINS 字段和 SACS 字段控制。若选择转换外部模拟信号，相应的引脚共用控制位需先被设置，再通过设置 SAINS 字段和 SACS 字段选择所需的外部通道输入。应注意的是，若通过 SAINS 字段选择内部模拟信号，则外部通道模拟输入将被自动关闭。关于 A/D 输入信号的更多详细信息请参见“A/D 转换器控制寄存器”和“A/D 转换器输入信号”章节。

外部输入通道	内部信号	A/D 通道选择位
14: AN0~AN13	8: V _{DD} , V _{DD} /2, V _{DD} /4, V _R , V _R /2, V _R /4, OCP0AO, OCP1AO	SAINS3~SAINS0 SACS3~SACS0

下图显示了 A/D 转换器的内部结构以及相关的寄存器与控制位。



注：A/D 转换器 PGA 输出 V_R 可用于驱动 OCP_n、OVP_n 和 UVP_n 中的 DAC。

A/D 转换器结构

A/D 转换器寄存器介绍

A/D 转换器的所有操作由 6 个寄存器控制。一对只读寄存器用来存放 12-bit ADC 数据的值。余下三个控制寄存器用于设置 A/D 转换器的操作和控制功能。寄存器 VBGRC 则用于控制 bandgap 参考电压。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SADOH (ADRFS=0)	D11	D10	D9	D8	D7	D6	D5	D4
SADOL (ADRFS=0)	D3	D2	D1	D0	—	—	—	—
SADOH (ADRFS=1)	—	—	—	—	D11	D10	D9	D8
SADOL (ADRFS=1)	D7	D6	D5	D4	D3	D2	D1	D0
SADC0	START	ADBZ	ADCEN	ADRF5	SACS3	SACS2	SACS1	SACS0
SADC1	SAINS3	SAINS2	SAINS1	SAINS0	—	SACKS2	SACKS1	SACKS0
SADC2	ADPGAEN	—	—	PGAIS	SAVRS1	SAVRS0	PGAGS1	PGAGS0
VBGRC	—	—	—	—	—	—	—	VBGREN

A/D 转换器寄存器列表

A/D 转换器数据寄存器 – SADOL, SADOH

对于具有 12-bit A/D 转换器的单片机，需要两个数据寄存器存放转换结果，一个高字节寄存器 SADOH 和一个低字节寄存器 SADOL。在 A/D 转换完毕后，单片机可以直接读取这些寄存器以获得转换结果。由于寄存器只使用了 16 位中的 12 位，其数据存储格式由 SADC0 寄存器的 ADRFS 位控制，如下表所示。D0~D11 是 A/D 转换数据结果位。未使用的位读为“0”。应注意若 A/D 转换器除能则 A/D 数据寄存器的内容将保持不变。

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

A/D 转换器数据寄存器

A/D 转换器控制寄存器 – SADC0, SADC1, SADC2

寄存器 SADC0、SADC1 和 SADC2 用来控制 A/D 转换器的功能和操作。这些 8-bit 寄存器定义包括选择连接至内部 A/D 转换器的模拟通道，数字化数据格式，A/D 时钟源，并控制和监测 A/D 转换器的开始和转换忙状态。由于该单片机只包含一个实际的模数转换硬件电路，因此外部或内部模拟输入中的每一个都需要分别被发送到转换器。寄存器 SADC1 的 SAINS 字段和寄存器 SADC0 的 SACS 字段用来决定模拟信号是来自外部或内部信号。

相关引脚共用功能选择位用于确定 I/O 端口上的引脚是否用作 A/D 转换器输入引脚。当引脚功能选择为 A/D 输入，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。A/D 转换器还包含了一个可编程增益放大器 PGA 用于产生 A/D 转换器内部参考电压。PGA 的所有操作由 SADC2 寄存器控制。

此外该单片机还提供一个 VBGRC 寄存器，该寄存器内的 VBGREN 位用于 Bandgap 参考电压控制。

● SADC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	START	ADBZ	ADCEN	ADRF5	SACS3	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 A/D 转换
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 转换忙标志位
0: A/D 转换结束或未开始转换
1: A/D 转换中
该只读标志位用于表明 A/D 转换过程是否完成。当 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 转换器功能使能控制位
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 转换器功能除能时，A/D 数据寄存器 SADOH 和 SADOL 的内容将保持不变。
- Bit 4 **ADRF5**: A/D 转换器数据格式控制位
0: A/D 转换器数据格式 → SADOH=D[11:4]; SADOL=D[3:0]
1: A/D 转换器数据格式 → SADOH=D[11:8]; SADOL=D[7:0]
此位控制存放在两个 A/D 数据寄存器中的 12-bit A/D 转换结果的格式。细节方面请参考 A/D 转换器数据寄存器章节。
- Bit 3~0 **SACS3~SACS0**: A/D 转换器外部模拟输入通道选择位
0000: AN0
0001: AN1
:
:
1100: AN12
1101: AN13
1110~1111: 通道不存在，若选择此项则输入为浮空

● SADC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SAINS3	SAINS2	SAINS1	SAINS0	—	SACKS2	SACKS1	SACKS0
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 7~4 **SAINS3~SAINS0**: A/D 转换器输入信号选择
0000: 外部来源 – 外部模拟通道输入
0001: 内部来源 – 内部 A/D 转换器电源电压 V_{DD}
0010: 内部来源 – 内部 A/D 转换器电源电压 $V_{DD}/2$
0011: 内部来源 – 内部 A/D 转换器电源电压 $V_{DD}/4$
0100: 外部来源 – 外部模拟通道输入
0101: 内部来源 – 内部 A/D 转换器 PGA 输出电压 V_R
0110: 内部来源 – 内部 A/D 转换器 PGA 输出电压 $V_R/2$
0111: 内部来源 – 内部 A/D 转换器 PGA 输出电压 $V_R/4$

- 1000: 内部来源 – OCP0 OPA 输出, OCP0AO
- 1001: 内部来源 – OCP1 OPA 输出, OCP1AO
- 1010~1011: 保留, 接地
- 1100~1111: 外部来源 – 外部模拟通道输入

当内部模拟信号和外部信号同时被选择, 无论 SACS 字段的值设定为多少, 外部通道信号输入将自动关闭。

- Bit 3 未定义, 读为 “0”
- Bit 2~0 **SACKS2~SACKS0**: A/D 转换时钟源选择位
 - 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: $f_{SYS}/128$

此字段用于选择 A/D 转换器时钟源。

• SADC2 寄存器

Bit	7	6	5	4	3	2	1	0
Name	ADPGAEN	—	—	PGAIS	SAVRS1	SAVRS0	PGAGS1	PGAGS0
R/W	R/W	—	—	R/W	R/W	R/W	R/W	R/W
POR	0	—	—	0	0	0	0	0

- Bit 7 **ADPGAEN**: PGA 使能控制
 - 0: 除能
 - 1: 使能

当 PGA 输出 V_R 被选作 A/D 转换器输入信号或 A/D 转换器参考电压, PGA 需通过置位此位使能。否则需通过清零此位除能 PGA 以节省功耗。
- Bit 6~5 未定义, 读为 “0”
- Bit 4 **PGAIS**: PGA 输入电压 V_{RI} 选择
 - 0: 来自外部 VREFI 引脚
 - 1: 来自内部独立参考电压 V_{BREF}

当 VREFI 引脚上的外部电压和内部独立参考电压 V_{BREF} 同时被选作 PGA 输入, 硬件将自动选择内部电压 V_{BREF} 作为 PGA 输入。
- Bit 3~2 **SAVRS1~SAVRS0**: A/D 转换器参考电压选择
 - 00: 内部 A/D 转换器电源 V_{DD}
 - 01: 外部 VREF 引脚
 - 1x: 内部 PGA 输出电压 V_R

该字段用于选择 A/D 转换器参考电压源。当内部 A/D 转换器电源或内部 PGA 输出电压和 VREF 引脚上的外部输入电压同时被选中, 硬件将自动选择内部参考电压作为 A/D 转换器参考电压。
- Bit 1~0 **PGAGS1~PGAGS0**: PGA 增益选择
 - 00: Gain=1
 - 01: Gain=1.667 – $V_{RI}=V_{BREF}$ (PGAIS=1) 时 $V_R=2V$
 - 10: Gain=2.5 – $V_{RI}=V_{BREF}$ (PGAIS=1) 时 $V_R=3V$
 - 11: Gain=3.333 – $V_{RI}=V_{BREF}$ (PGAIS=1) 时 $V_R=4V$

• VBGRC 寄存器

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

Bit 7~1 未定义，读为“0”

Bit 0 **VBGREN**: 独立参考电压 Bandgap 使能 / 除能控制

0: 除能

1: 使能

当此位被清零， V_{BGRF} 将处于高阻态。

A/D 转换器操作

SADC0 寄存器中的 START 位用于启动 A/D 转换。当单片机设定此位从逻辑低到逻辑高，然后再到逻辑低，就会开始一个模数转换周期。

SADC0 寄存器中的 ADBZ 位用于表明模数转换是否正在进行。A/D 转换成功启动后，ADBZ 位被自动置为“1”。在转换周期结束后，ADBZ 位会被清零。此外，中断控制寄存器内相应的 A/D 中断请求标志位也会置位，如果中断使能，就会产生对应的内部中断信号。A/D 内部中断信号将引导程序到相应的 A/D 内部中断入口。如果 A/D 内部中断被禁止，可以让单片机轮询 SADC0 寄存器中的 ADBZ 位，检测此位是否被清除，以作为另一种侦测 A/D 转换周期结束的方法。

A/D 转换器时钟源来自系统时钟 f_{SYS} ，可选择为 f_{SYS} 或 f_{SYS} 的分频。分频比由 SADC1 寄存器中的 SACKS 字段确定。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 时钟源还有一些限制。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必须小心。如果系统时钟速度为 8MHz 时，SACKS2~SACKS0 位不能设为“000”，“001”或“111”。否则所得的 A/D 时钟周期将小于最小时钟周期或大于最大时钟周期，导致所得 A/D 转换结果不准确。参考下表，根据单片机的类型，表格中加星号“*”的值需多加注意，因为这些值可能小于或大于所指定的 A/D 时钟周期。

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 *
12MHz	83ns *	167ns *	333ns *	667ns	1.33 μs	2.67 μs	5.33 μs	10.67 μs *
16MHz	62.5ns *	125ns *	250ns *	500ns	1 μs	2 μs	4 μs	8 μs

A/D 时钟周期范例

SADC0 寄存器的 ADCEN 位用于控制 A/D 转换电路电源的开启 / 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 转换成功初始化前需一段延时，如时序图所示。即使选择无引脚作为 A/D 输入，若 ADCEN 设为“1”，仍然会产生功耗。因此当未使用 A/D 转换器功能时，在功耗敏感的应用中建议设置 ADCEN 为低以减少功耗。

A/D 转换器参考电压

A/D 转换器的参考电压可以来自正电源 V_{DD} 或来自 VREF 引脚上的外部参考源或是内部 PGA 输出电压 V_R ，通过配置 SADC2 寄存器中的 SAVRS1 和 SAVRS0 位进行选择。若 SAVRS 字段设置为“00”，A/D 转换器参考电压将来自 V_{DD} 电源，若设置为“01”则参考电压来自 VREF 引脚，否则参考电压将来自 PGA 输出电压 V_R 。由于 VREF 引脚与其它功能共用引脚，当这个引脚被选作参考电压引脚时，相应的引脚共用功能选择位应被预先设置以除能其它引脚共用功能。此外，若程序同时选择 VREF 引脚上的外部电压和内部参考电压 V_{DD} 或 V_R 作为 A/D 转换器参考电压，硬件将只选择内部参考电压作为 A/D 转换器参考电压输入。模拟输入值必须小于所选的参考电压值 V_{REF} 。

A/D 转换器具有另一个 VREFI 引脚，其为 PGA 的其中一个输入信号，用于提供 A/D 转换器参考电压。要选择此 PGA 输入信号，SADC2 寄存器中的 PGAIS 位必须清零且相关引脚功能共用控制位需正确配置。然而，PGA 输入也可由内部独立参考电压 V_{BGRREF} 提供。若程序同时选中 VREFI 引脚上的外部电压和内部电压 V_{BGRREF} 作为 PGA 输入，硬件将只选择内部电压 V_{BGRREF} 作为 PGA 输入。

SAVRS[1:0]	参考电压	说明
00	V_{DD}	内部 A/D 转换器电源电压
01	VREF 引脚	外部 A/D 转换器参考引脚 VREF
10 或 11	V_R	内部 A/D 转换器 PGA 输出电压

A/D 转换器参考电压选择

A/D 转换器输入信号

A/D 转换器的所有模拟输入引脚与 I/O 端口上的引脚及其它功能共用。PxS1 和 PXS0 寄存器中相应的引脚共用功能选择位决定了内部引脚是设置为 A/D 转换器模拟通道输入还是其它功能。如果相应引脚设置为 A/D 转换器模拟通道输入那么该引脚原引脚功能除能。通过这种方式，引脚的功能可由程序来控制，灵活地切换引脚功能。如果将引脚设为 A/D 输入，则通过寄存器编程设置的所有上拉电阻会自动断开。请注意，端口控制寄存器不需要为使能 A/D 输入而先设定为输入模式，当相关 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。

若 SAINS3~SAINS0 字段设为“0000”，“0100”或“1110~1111”，则所要转换的信号为外部模拟通道信号，SACS3~SACS0 字段可决定选择哪个外部通道的信号进行转换。若 SAINS3~SAINS0 位设置为“0001~0011”，则选择转换内部信号 V_{DD} 、 $V_{DD}/2$ 或 $V_{DD}/4$ ；若该字段设置为“0101~0111”则选择转换内部信号 V_R 、 $V_R/2$ 或 $V_R/4$ ；若该字段设置为“1000”则选择转换 OCP0 OPA 输出信号 OCP0AO；若该字段设置为“1001”则选择转换 OCP1 OPA 输出信号 OCP1AO。

注意当程序同时选中外部信号 (AN0~AN13) 和内部信号 (V_{DD} 、 $V_{DD}/2$ 、 $V_{DD}/4$ 、 V_R 、 $V_R/2$ 、 $V_R/4$ 、OCP0AO 或 OCP1AO) 作为 A/D 转换器输入信号，硬件将仅选择内部信号作为 A/D 转换器输入，外部通道信号输入将被自动关闭。

SAINS[3:0]	SACS[3:0]	输入信号	说明
0000, 0100, 1100~1111	0000~1101	AN0~AN13	外部通道模拟输入
	111x	—	通道不存在, 若选择此项则输入为浮空
0001	xxxx	V _{DD}	内部 A/D 转换器电源电压 V _{DD}
0010	xxxx	V _{DD} /2	内部 A/D 转换器电源电压 V _{DD} /2
0011	xxxx	V _{DD} /4	内部 A/D 转换器电源电压 V _{DD} /4
0101	xxxx	V _R	内部 A/D 转换器 PGA 输出 V _R
0110	xxxx	V _R /2	内部 A/D 转换器 PGA 输出 V _R /2
0111	xxxx	V _R /4	内部 A/D 转换器 PGA 输出 V _R /4
1000	xxxx	OCP0AO	OCP0 OPA 输出
1001	xxxx	OCP1AO	OCP1 OPA 输出
101x	xxxx	—	保留, 接地

“x”：无关

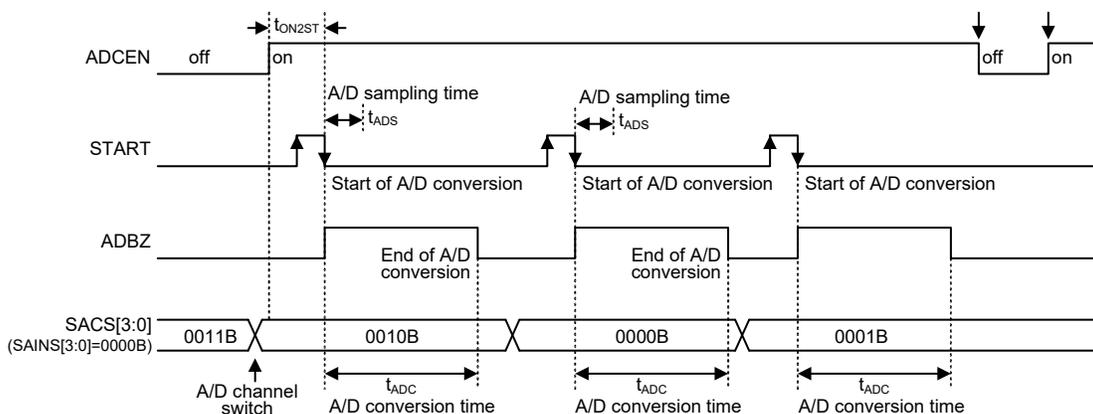
A/D 转换器输入信号选择

转换率和时序图

一个完整的 A/D 转换包含两部分, 数据采样和数据转换。数据采样时间定义为 t_{ADS} , 需 4 个 A/D 时钟周期, 而数据转换需 12 个 A/D 时钟周期。所以一个完整的 A/D 转换时间 t_{ADC} 一共需要 16 个 A/D 时钟周期。

$$\text{最大 A/D 转换率} = \text{A/D 时钟周期} / 16$$

下列时序图表示一个模数转换过程中不同阶段的图形与时序。由应用程序控制开始 A/D 转换过程后, 单片机的内部硬件就会开始进行转换, 在这个过程中, 程序可以继续其它功能。A/D 转换时间为 $16t_{ADCK}$, t_{ADCK} 为 A/D 时钟周期。



A/D 转换时序 - 外部通道输入

A/D 转换步骤

下面概述实现 A/D 转换过程的各个步骤。

- 步骤 1
通过 SADC1 寄存器中的 SACKS2~SACKS0 位, 选择所需的 A/D 转换器转换时钟。
- 步骤 2
将 SADC0 寄存器中的 ADCEN 位置高以使能 A/D 转换器。

- 步骤 3
通过 SADC1 寄存器中的 SAINS 字段，选择连接至内部 A/D 转换器的信号。
选择外部通道输入，前往步骤 4。
选择内部模拟信号，前往步骤 5。
 - 步骤 4
当 A/D 输入信号来自外部通道输入，相应引脚应先通过配置相关的引脚共用功能控制位选择。所需的外部通道输入由 SACS 字段确定。完成此步骤后，前往步骤 6。
 - 步骤 5
若选中相关内部模拟信号作为 A/D 输入信号，SAINS 字段需正确配置，此时无论 SACS 字段为何值，外部通道模拟输入将被自动断开。完成此步骤后，前往步骤 6。
 - 步骤 6
通过设置 SADC2 寄存器中的 SAVRS1~SAVRS0 位选择参考电压源。若选择 PGA 输出电压，则必须使能 PGA，并通过 SADC2 寄存器中的 PGAIS 位选择 PGA 输入源。
 - 步骤 7
通过配置 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 A/D 转换中断，则中断控制寄存器需要正确地设置，以确保 A/D 中断功能是有效的。总中断控制位 EMI 以及 A/D 转换器中断位 ADE 需要预先置位为“1”。
 - 步骤 9
现在可以通过设定 START 位从低到高再到低，开始 A/D 转换的过程。
 - 步骤 10
若 A/D 转换正在进行，ADBZ 标志将置高。A/D 转换结束后，ADBZ 标志置低，并可从 SADOH 和 SADOL 寄存器读取输出数据。
- 注：若使用轮询 SADC0 寄存器中的 ADBZ 位的方法以检测模数转换过程是否完成，上述中断使能步骤可以忽略。

编程注意事项

在单片机工作时，如果 A/D 转换器未使用，通过设置 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

该单片机含有一个 12-bit A/D 转换器，它们转换的最大值可达 0FFFH。由于模拟输入最大值等于实际 A/D 转换器参考电压 V_{REF} ，因此每一位可表示 $V_{REF}/4096$ 的模拟输入值。

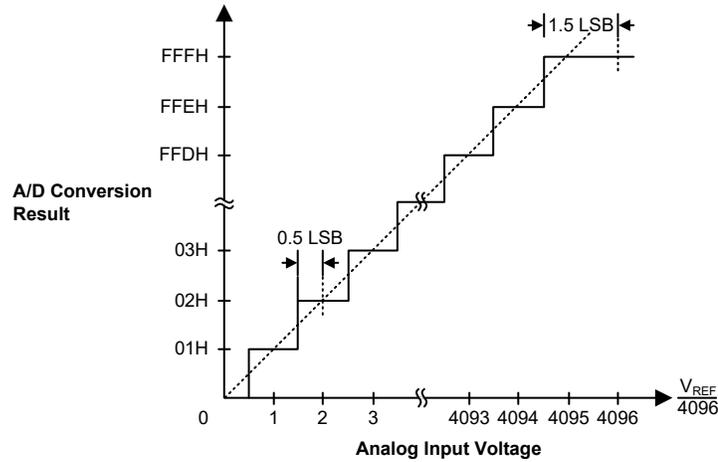
$$1 \text{ LSB} = V_{REF} \div 4096$$

通过下面的等式可估算 A/D 转换器输入电压值：

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

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

值的最大值将在 V_{REF} 之前的 1.5 LSB 处改变。应注意的是此处的 V_{REF} 电压指的是 A/D 转换器参考电压源，由 SAVRS 字段确定。



理想 A/D 转换功能

A/D 转换程序范例

下面两个范例程序用来说明怎样设置和实现 A/D 转换。第一个范例是轮询 SADC0 寄存器中的 ADBZ 位来判断 A/D 转换何时完成，而第二个范例则使用中断的方式判断。

范例 1：使用轮询 ADBZ 轮询的方式来检测转换结束

```

clr ADE                ; disable ADC interrupt
mov a,03H
mov SADC1,a            ; select fsys/8 as A/D clock
mov a,01h              ; setup PDS0 to configure pin AN0
mov PDS0,a
mov a,20h
mov SADC0,a
mov a,00h
mov SADC2,a           ; enable and connect AN0 channel to A/D converter
:
:
start_conversion:
clr START              ; high pulse on start bit to initiate conversion
set START              ; reset A/D
clr START              ; start A/D
polling_EOC:
sz ADBZ                ; poll the SADC0 register ADBZ bit to detect end
                       ; of A/D conversion
jmp polling_EOC        ; continue polling
mov a,SADOL             ; read low byte conversion result value
mov SADOL_buffer,a     ; save result to user defined register
mov a,SADOH             ; read high byte conversion result value
mov SADOH_buffer,a    ; save result to user defined register
:
:
jmp start_conversion   ; start next A/D conversion
    
```

范例 2：使用中断的方式来检测转换结束

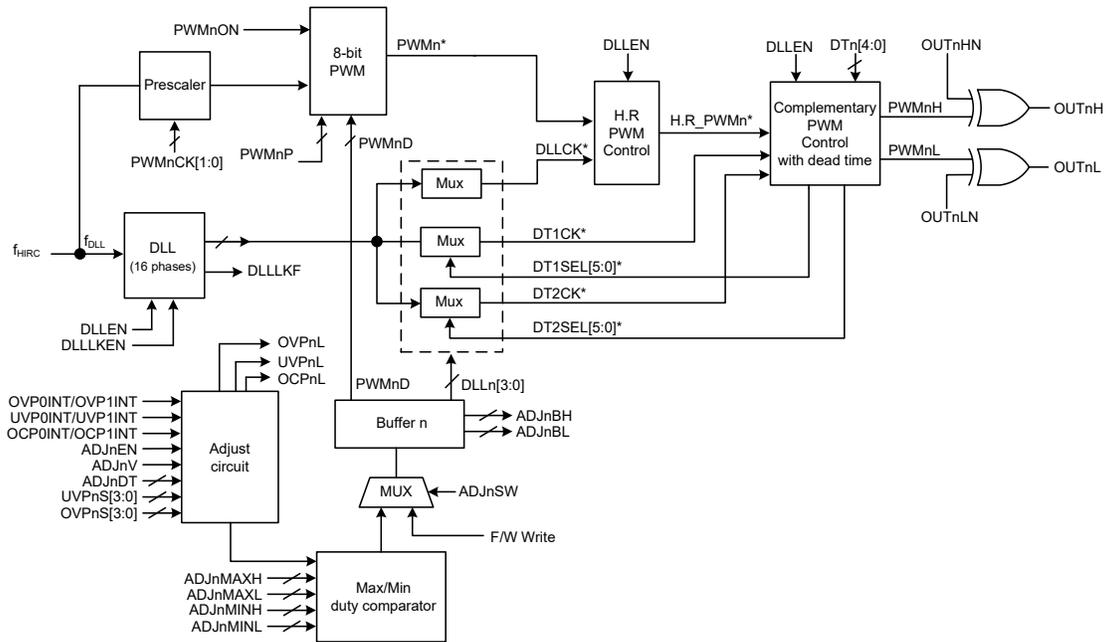
```
clr ADE                ; disable ADC interrupt
mov a,03H
mov SADC1,a           ; select fsys/8 as A/D clock
mov a,01h            ; setup PDS0 to configure pin AN0
mov PDS0,a
mov a,20h
mov SADC0,a
mov a,00h
mov SADC2,a          ; enable and connect AN0 channel to A/D converter
:
:
start_conversion:
clr START            ; high pulse on START bit to initiate conversion
set START            ; reset A/D
clr START            ; start A/D
clr ADF              ; clear ADC interrupt request flag
set ADE              ; enable ADC interrupt
set EMI              ; enable global interrupt
:
:
; ADC interrupt service routine
ADC_ISR:
mov acc_stack,a     ; save ACC to user defined memory
mov a,STATUS
mov status_stack,a ; save STATUS to user defined memory
:
:
mov a,SADOL          ; read low byte conversion result value
mov SADOL_buffer,a ; save result to user defined register
mov a,SADOH          ; read high byte conversion result value
mov SADOH_buffer,a ; save result to user defined register
:
:
EXIT_INT_ISR:
mov a,status_stack
mov STATUS,a        ; restore STATUS from user defined memory
mov a,acc_stack     ; restore ACC from user defined memory
reti
```

带延迟锁相环 / 死区时间的可自动调节的高精度 PWM

该单片机包含一个具有互补输出的高度集成多功能高精度 PWM 信号发生器，增强了应用的灵活性。

功能描述

该高精度 8-bit PWM 发生器电路由一个 PWM 发生器电路，延迟锁相环路及一个带死区时间插入的 PWM 互补输出电路组成，此外还支持高精度 PWM 输出的 PWM 占空比调节控制。



H.R PWM 输出方框图 (n=0~1)

- 注：1. “*” 为内部信号名称而非特殊功能寄存器位名称
 DT1SEL[5:0] 和 DT2SEL[5:0] 基于 DTn[4:0] 自动进行计算与选择。
 DT1CK 为 PWMnH 死区时间参考信号
 DT2CK 为 PWMnL 死区时间参考信号
 DLLCK 为 H.R_PWMn 参考信号
 2. H.R 为高精度 (High Resolution) 的缩写符号

H.R PWM 寄存器介绍

高精度 PWM 的基本操作通过一系列寄存器来控制。PWM 周期寄存器 PWMnP 用于存储所需的 8-bit PWM 周期值。PWM 占空比值则存储在 8-bit PWMnD 寄存器中。PWMnC 寄存器用于控制 PWMn 功能的使能 / 除能控制，PWMn 计数时钟源选择，DLL 电路以及死区时间周期等。DLLn 寄存器用于选择 DLL 电路相位。DLLC 寄存器用于 DLL 电路使能控制和失锁保护功能控制等。

PWM 自动调节功能的控制也可通过一些寄存器进行设置。ADJnC 寄存器用于控制 PWM 自动调节功能的使能 / 除能，PWMn 占空比调节操作以及存储 OUVpn (过电压 / 欠电压保护) 比较器的输出状态。ADJnS 寄存器用于选择当过电压或欠电压状况发生时的调节步数。ADJnDT 寄存器用于设置自动调节功能触发延迟时间。两组寄存器对，ADJnMAXH & ADJnMAXL 和 ADJnMINH & ADJnMINL 分别用于设置可允许的占空比数据的最大和最小值。一个寄存器

对 ADJnBH & ADJnBL 用于存储自动调节时 PWM 缓存的占空比数据。剩下的 OUTPC 寄存器用于 PWMn 输出反相控制。

寄存器名称	位							
	7	6	5	4	3	2	1	0
PWMnP	D7	D6	D5	D4	D3	D2	D1	D0
PWMnD	D7	D6	D5	D4	D3	D2	D1	D0
PWMnC	PWMnCK1	PWMnCK0	PWMnON	DTn4	DTn3	DTn2	DTn1	DTn0
DLLn	DLLn3	DLLn2	DLLn1	DLLn0	—	—	—	—
DLLC	DLLKEN	DLEN	—	—	—	—	—	DLLKF
ADJnC	ADJnEN	ADJnV	ADJnSW	—	OVpL	UVpL	OCpL	—
ADJnS	OVpS3	OVpS2	OVpS1	OVpS0	UVpS3	UVpS2	UVpS1	UVpS0
ADJnDT	—	—	D5	D4	D3	D2	D1	D0
ADJnMAXH	D11	D10	D9	D8	D7	D6	D5	D4
ADJnMAXL	D3	D2	D1	D0	—	—	—	—
ADJnMINH	D11	D10	D9	D8	D7	D6	D5	D4
ADJnMINL	D3	D2	D1	D0	—	—	—	—
ADJnBH	D11	D10	D9	D8	D7	D6	D5	D4
ADJnBL	D3	D2	D1	D0	—	—	—	—
OUTPC	OUT1HS	OUT1LS	OUT0HS	OUT0LS	OUT1HN	OUT1LN	OUT0HN	OUT0LN

H.R PWM 发生器 & 自动调节控制寄存器列表 (n=0~1)

• PWMnP 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**: 8-bit PWMn 周期寄存器
 $PWMn \text{ 周期} = PWMnP[7:0] + 1$

• PWMnD 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**: 8-bit PWMn 占空比寄存器
 PWMnP 和 PWMnD 两个寄存器用于 8-bit PWMn 周期与占空比控制。在设置过程中应注意以下几点:

1. 设置的 PWMnD 值应满足条件: $1 \leq PWMnD \leq (PWMnP - 1)$
2. $PWMnD \text{ (最小值)} = 1 + DLLn[3:0] - DTn[4:0]$ (其中 $DLLn[3:0] = 0000B$, $DTn[4:0] = 11111B$)
3. $PWMnD \text{ (最大值)} = PWMnP - 1 + DLLn[3:0] - DTn[4:0]$ (其中 $DLLn[3:0] = 11111B$, $DTn[4:0] = 00000B$)

• PWMnC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PWMnCK1	PWMnCK0	PWMnON	DTn4	DTn3	DTn2	DTn1	DTn0
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 **PWMnCK1~PWMnCK0**: PWMn 计数时钟源选择

00: f_{HRC}
01: $f_{HRC}/2$
10: $f_{HRC}/4$
11: $f_{HRC}/8$

Bit 5 **PWMnON**: PWMn 功能使能控制

0: 除能, PWM 计数值 = 0
1: 使能

Bit 4~0 **DTn[4:0]**: PWMn 死区时间选择

0000: 死区时间 = $t_{DLL} \times 0 \sim t_{DLL} \times 1$
00001: 死区时间 = $t_{DLL} \times 2 \sim t_{DLL} \times 3$
00010: 死区时间 = $t_{DLL} \times 4 \sim t_{DLL} \times 5$
00011: 死区时间 = $t_{DLL} \times 6 \sim t_{DLL} \times 7$
00100: 死区时间 = $t_{DLL} \times 8 \sim t_{DLL} \times 9$
00101: 死区时间 = $t_{DLL} \times 10 \sim t_{DLL} \times 11$
00110: 死区时间 = $t_{DLL} \times 12 \sim t_{DLL} \times 13$
00111: 死区时间 = $t_{DLL} \times 14 \sim t_{DLL} \times 15$
01000: 死区时间 = $t_{DLL} \times 16 \sim t_{DLL} \times 17$
...
11110: 死区时间 = $t_{DLL} \times 60 \sim t_{DLL} \times 61$
11111: 死区时间 = $t_{DLL} \times 62 \sim t_{DLL} \times 63$

注: $t_{DLL} = 1/(f_{HRC} \times 16)$ 。

• DLLn 寄存器

Bit	7	6	5	4	3	2	1	0
Name	DLLn3	DLLn2	DLLn1	DLLn0	—	—	—	—
R/W	R/W	R/W	R/W	R/W	—	—	—	—
POR	0	0	0	0	—	—	—	—

Bit 7~4 **DLLn3~DLLn0**: DLL 相位选择

0000: 将 H.R_PWMn 占空比下降沿微调至 DLL phase#0 的输出上升沿
0001: 将 H.R_PWMn 占空比下降沿微调至 DLL phase#1 的输出上升沿
0010: 将 H.R_PWMn 占空比下降沿微调至 DLL phase#2 的输出上升沿
...
1110: 将 H.R_PWMn 占空比下降沿微调至 DLL phase#14 的输出上升沿
1111: 将 H.R_PWMn 占空比下降沿微调至 DLL phase#15 的输出上升沿

Bit 3~0 未定义, 读为 “0”

• DLLC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	DLLKEN	DLEN	—	—	—	—	—	DLLKF
R/W	R/W	R/W	—	—	—	—	—	R/W
POR	1	0	—	—	—	—	—	0

- Bit 7 **DLLKEN**: DLL 电路失锁保护功能使能控制
 0: 除能
 1: 使能
 注: 当 DLL 功能使能且 DLLKEN=1, 若有失锁现象发生, 则 DLLKF 位被置高同时 DLL 会自动解除失锁状况。当 DLLKEN=0, 即使失锁现象发生, DLLKF 位也始终为零, 且 DLL 不会自动解除失锁状况。
- Bit 6 **DLEN**: DLL 及死区时间功能控制
 0: DLL 除能且不会插入死区时间
 1: DLL 使能且插入死区时间 (时间由 DTn[4:0] 决定)
 若此位为零则 PWMnCK[1:0] 位可通过软件设置为“00~11”中任一值, 且输出的 H.R_PWMn=PWMn, 无死区时间插入。若此位为 1, 则 PWMnCK[1:0] 位将被硬件强制设为“00”且不可改变, PWMn 经由 DLL 微调, 输出带死区时间插入的高精度 PWM。
- Bit 5~1 未定义, 读为“0”
- Bit 0 **DLLKF**: DLL 电路失锁标志位
 0: 没有发生失锁现象
 1: 发生失锁现象
 此位可通过软件清零, 但不能通过软件置 1。
 注: 当 DLL 功能使能且 DLLKEN=1, 若无失锁现象发生, 则 DLLKF 位为零, 若有失锁现象发生, 则 DLLKF 位被置 1。但若 DLLKEN=0 则无论有无失锁现象发生 DLLKF 位始终为零。

• ADJnDT 寄存器

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

- Bit 7~6 未定义, 读为“0”
- Bit 5~0 **D5~D0**: 自动调节 PWMn 延迟时间选择
 000000: 延迟时间 = PWMn 周期 × 2
 000001: 延迟时间 = PWMn 周期 × 4
 000010: 延迟时间 = PWMn 周期 × 8
 ...
 111111: 延迟时间 = PWMn 周期 × 128
 即: 延迟时间 = (ADJnDT[5:0]+1) × PWMn 周期 × 2

• ADJnS 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OVPnS3	OVPnS2	OVPnS1	OVPnS0	UVPnS3	UVPnS2	UVPnS1	UVPnS0
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 7~4 **OVPnS3~OVPnS0**: OVP0/OVP1 自动调节 PWMn 占空比步数选择
 0000: 0
 0001: 1
 ...
 1111: 15

Bit 3~0 **UVPnS3~UVPnS0**: UVP0/UVP1 自动调节 PWMn 占空比步数选择
 0000: 0
 0001: 1
 ...
 1111: 15

• ADJnC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	ADJnEN	ADJnV	ADJnSW	—	OVPnL	UVPnL	OCPnL	—
R/W	R/W	R/W	R/W	—	R	R	R	—
POR	0	0	0	—	x	x	x	—

“x”：未知

Bit 7 **ADJnEN**: PWM 占空比自动调节功能使能控制
 0: 除能
 1: 使能

Bit 6 **ADJnV**: PWMn 占空比自动调节动作控制
 0: OVPnL 增加占空比, UVPnL 减少占空比
 1: OVPnL 减少占空比, UVPnL 增加占空比

Bit 5 **ADJnSW**: PWMn 占空比软件自动调节控制
 0: 除能, 通过 F/W 将 PWMnD + DLLn 寄存器写入 Buffer
 1: 使能, 通过自动调节系统写入 Buffer

此位可通过软件清零, 但不能通过软件置位。当 ADJnEN=0 时, 自动调节功能关闭, 则 ADJnSW 位始终为 0; 当 ADJnEN=1 时, 自动调节功能开启。

Bit 4 未定义, 读为 “0”

Bit 3 **OVPnL**: OVP0/OVP1 比较器输出状态位
 0: 输出低 (无过电压状况发生)
 1: 输出高 (过电压状况发生)

Bit 2 **UVPnL**: UVP0/UVP1 比较器输出状态位
 0: 输出低 (无欠电压状况发生)
 1: 输出高 (欠电压状况发生)

Bit 1 **OCPnL**: OCP0/OCP1 比较器输出状态位
 0: 输出低 (无过电流状况发生)
 1: 输出高 (过电流状况发生)

Bit 0 未定义, 读为 “0”

OVPnL/ UVPnL/ OCPnL 与 ADJnSW 位的关系如下表所示。

OVPnL	UVPnL	OCPnL	注释	
			ADJnSW	说明
0	0	0	0	自动调节系统未被触发。
0	0	1	0	出现过电流状况，自动调节系统未被触发。
0	1	0	1	自动调节系统被触发，未出现过电流状况。
0	1	1	0	UVP=1，但由于出现过电流状况，所以自动调节系统无法被触发。
1	0	0	1	自动调节系统被触发，未出现过电流状况。
1	0	1	1	OVP=1，尽管出现过电流状况，自动调节系统仍被触发。
1	1	0	0	OVP、UVP=1，外部电路出错，自动调节系统无法被触发。
1	1	1	0	出现过电流状况，尽管 OVP、UVP=1，自动调节系统仍无法被触发。

● ADJnMAXH & ADJnMAXL 寄存器

寄存器	ADJnMAXH								ADJnMAXL							
	位	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1
Name	D11	D10	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	R/W	—	—	—	—
POR	0	0	0	0	0	0	0	0	0	0	0	0	—	—	—	—

“—”：未定义，读为“0”

D11~D4：自动调节 PWMn 最大占空比高字节

D3~D0：自动调节 PWMn 最大占空比低字节

D11~D4 对应于寄存器 PWMnD[7:0] 位，D3~D0 对应于寄存器 DLLn[3:0] 位。

● ADJnMINH & ADJnMINL 寄存器

寄存器	ADJnMINH								ADJnMINL							
	位	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1
Name	D11	D10	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	R/W	—	—	—	—
POR	0	0	0	0	0	0	0	0	0	0	0	0	—	—	—	—

“—”：未定义，读为“0”

D11~D4：自动调节 PWMn 最小占空比高字节

D3~D0：自动调节 PWMn 最小占空比低字节

D11~D4 对应于寄存器 PWMnD[7:0] 位，D3~D0 对应于寄存器 DLLn[3:0] 位。

● ADJnBH & ADJnBL 寄存器

寄存器	ADJnBH								ADJnBL							
位	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	D11	D10	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	R/W	—	—	—	—
POR	0	0	0	0	0	0	0	0	0	0	0	0	—	—	—	—

“—”：未定义，读为“0”

D11~D4: 自动调节 PWMn 缓存占空比高字节

D3~D0: 自动调节 PWMn 缓存占空比低字节

D11~D4 对应于寄存器 PWMnD[7:0] 位，D3~D0 对应于寄存器 DLLn[3:0] 位。

● OUTPC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OUT1HS	OUT1LS	OUT0HS	OUT0LS	OUT1HN	OUT1LN	OUT0HN	OUT0LN
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 **OUT1HS**: OCP/OUVP 发生时或 PWMn 除能时 OUT1H 输出状态控制
0: 输出 0
1: 输出 1

Bit 6 **OUT1LS**: OCP/OUVP 发生时或 PWMn 除能时 OUT1L 输出状态控制
0: 输出 0
1: 输出 1

Bit 5 **OUT0HS**: OCP/OUVP 发生时或 PWMn 除能时 OUT0H 输出状态控制
0: 输出 0
1: 输出 1

Bit 4 **OUT0LS**: OCP/OUVP 发生时或 PWMn 除能时 OUT0L 输出状态控制
0: 输出 0
1: 输出 1

Bit 3 **OUT1HN**: OUT1H 信号反相控制
0: 同相
1: 反相

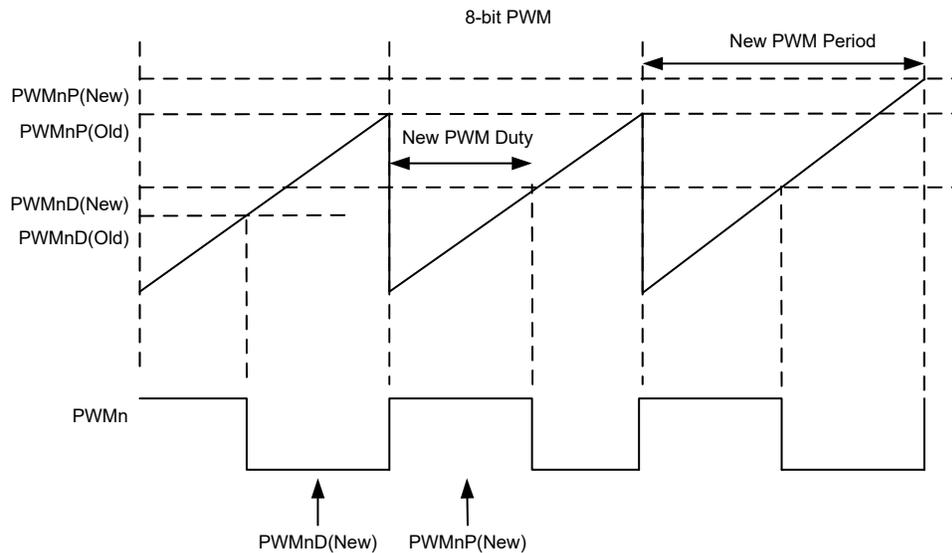
Bit 2 **OUT1LN**: OUT1L 信号反相控制
0: 同相
1: 反相

Bit 1 **OUT0HN**: OUT0H 信号反相控制
0: 同相
1: 反相

Bit 0 **OUT0LN**: OUT0L 信号反相控制
0: 同相
1: 反相

PWM 发生器

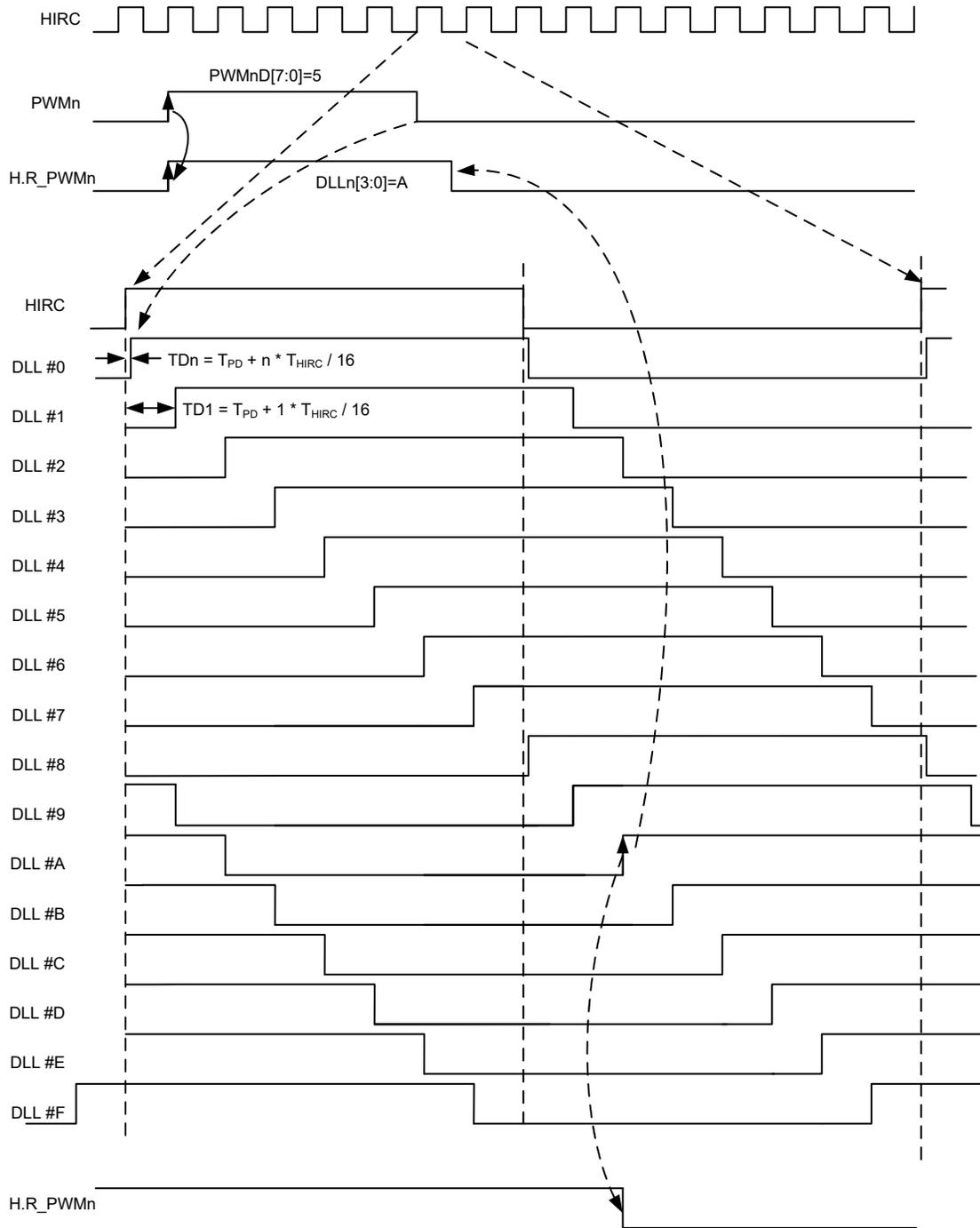
PWM 信号发生器由 HIRC 时钟驱动并产生 PWM_n 信号，通过配置 8-bit PWM_nP 和 PWM_nD 寄存器可调节生成的 PWM_n 信号的占空比和周期。PWM_n 信号周期与 PWM_nC 寄存器的 PWM_nCK[1:0] 位选择的 PWM_n 计数器时钟频率相关，其值由 PWM_nP 寄存器设置。PWM_n 信号占空比值由 PWM_nD 寄存器设置。



通过软件写入值到 DLL_n、DT_n[4:0]、PWM_nD 和 PWM_nP 寄存器后，硬件会等待到 PWM_n 计数器为零时，才会更新到新数据。

延迟锁相环

DLL 为延迟锁相环 (Delay Lock Loop) 的简写。DLL 可在一个 HIRC 时钟周期内产生 16 个相位输出。这 16 个相位输出用于控制 PWM_n 信号输出微调。若选择 PWM_n 时钟为 f_{HIRC} ，这表示 PWM_n 输出占空比精度为 $1/f_{HIRC}$ 。PWM_n 通过 DLL_n 相位选择与 H.R PWM_n 控制 (由 DLL_n 寄存器中的 DLL_n[3:0] 位控制) 电路后输出微调后的 PWM_n 信号，藉由 DLL_n 可将高精度 PWM_n 的占空比精度提高 4 位。



DLL 失锁保护说明

该单片机提供失锁保护电路，通过设置 DLLLKEN 位为高可启用此保护电路。

当单片机受到干扰时，DLL 电路可能会发生失锁状况，此时 DLL 所产生出来的一个相位时间会变成正常相位的 1.5 倍。如果 DLLLKEN 位为高，启动失锁保护电路，则此失锁相位会在 30 μ s 后恢复正常。除此之外，会将 DLLLKF 位置高，以告知用户发生过失锁现象。如果 DLLLKEN 位为低，关闭失锁保护电路，则若有失锁状况发生，DLL 产生的相位不会自动恢复到正常相位，且 DLLLKF 位将始终为低，因此用户也无法得知有失锁现象发生。

自动调节电路

为了提高 DC-DC 应用中的响应速度，该单片机中的 PWM 发生器功能另外提供了自动调节电路。以下总结了自动调节功能的设置步骤。

步骤 1. 清零 ADJnEN 位以关闭自动调节电路，并进行初始化设置：

- 通过 12-bit 的 ADJnMAXH&ADJnMAXL 和 ADJnMINH&ADJnMINL 寄存器对设置最大 / 最小占空比值。注：
 $1 < \text{Min. 值} < (\text{PWMnD} + \text{DLLn}) < \text{Max. 值}$
- 配置 6-bit ADJnDT 寄存器位设置每次触发的延迟时间。
- 设置 ADJnC 寄存器中的 ADJnV 位，选择占空比调节增减动作。
- 设置 ADJnS 寄存器选择 OUVp0/OUVp1 发生时，占空比调节的步数 (0~15)。
- 过电压 OVP0/OVP1 及欠电压 UVP0/UVP1 限定电压值设置。注：
 $\text{UVP0/UVP1} < \text{目标电压} < \text{OVP0/OVP1}$ 。

步骤 2. PWMn 启动：

- PWMn 初始化，包括 PWMnP 和 (PWMnD+DLLn) 初始化。
- 设置 PWMnON 为高以开启 PWMn 发生器。
- 读取 OUVp0/OUVp1 输出后调节占空比，确保输出电压 = 目标电压

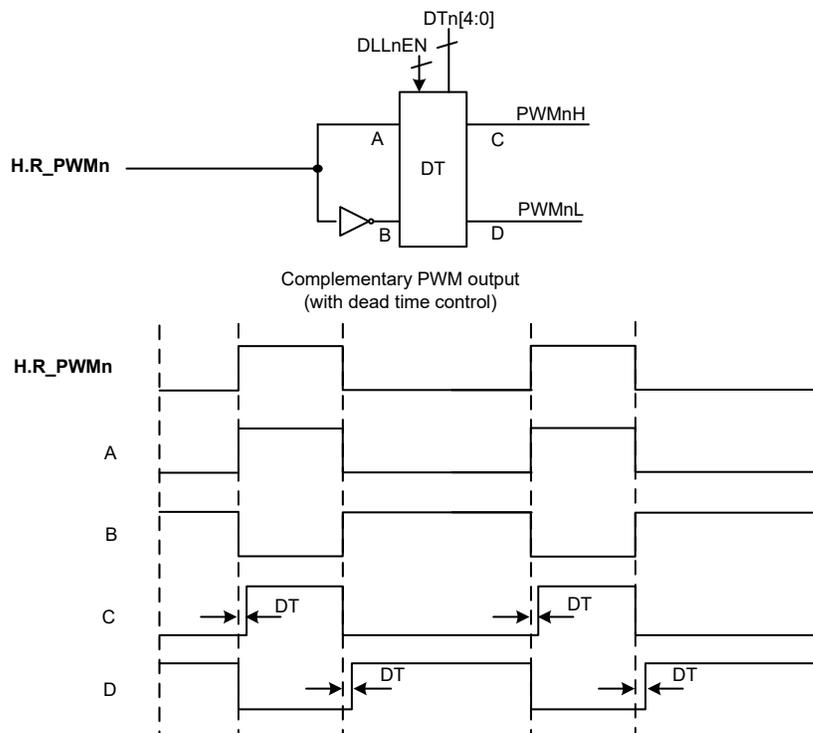
步骤 3. 设置 ADJnEN 位为高以开启自动调节电路，OVPnL/UVPnL 电平可触发自动调节。

步骤 4. 若有 UVP0/UVP1 中断发生，F/W 基于 UVP0/UVP1 和 OCP0/OCP1 比较器输出给出一定的延迟时间，以判断欠电压状况是由负载瞬间加重导致还是外部设备短路造成的。

步骤 5. 若有 OVP0/OVP1 中断发生，F/W 基于 OVP0/OVP1 比较器输出给出一定的延迟时间，以判断是过电压状况是由负载瞬间减少导致还是反馈电路异常造成的。

死区时间插入

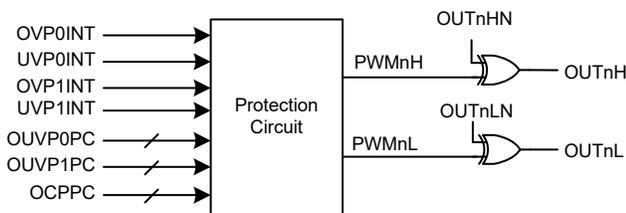
该单片机提供一对互补输出信号，可作为 PWMn 驱动器信号。该信号源自高精度 PWMn 输出信号，带 DLL 电路的 8-bit PWMn。PWM 输出为高有效信号。通过 DLLnEN 位可启用死区时间发生器并会插入一个死区时间以预防过大的直流电流，插入的死区时间长度可通过 PWMnC 寄存器中的 DTn[4:0] 进行设置，且将在输入信号的每一个上升沿插入一个死区时间。



注：上图中的 C 和 D 为 PWMn 互补信号插入死区时间后的输出信号。

保护与反相控制

在 H.R_PWM 互补信号对中插入了一个死区时间可以预防过大的直流电流，但这两个信号也可能由于误操作或电气噪声等一些非预期的原因而同时处于无效状态。该单片机提供了一个保护功能，即在 PWMnH 或 PWMnL 信号为无效状态时强制使这两个信号输出反相信号。反相控制电路通过设置 OUTPC 寄存器中的相应反相控制位 OUTnHN 或 OUTnLN 位决定是否将信号进行反相。



该单片机还提供了过电流保护与过电压 / 欠电压保护功能对 PWMn 的输出进行保护，具体操作可参考过电流保护章节中 OCPPC 寄存器与过电压 / 欠电压保护章节中 OUVpnPC 寄存器介绍。当有过电流，过电压或欠电压情况发生时，输出的 OUT1H&OUT1L 及 OUT0H&OUT0L 可被强制设置为无效状态，可通过 OUTPC 寄存器的 OUT1HS、OUT1LS、OUT0HS 及 OUT0LS 位进行设置。当有 OCP、OVP 或 UVP 状况发生时会产生中断以通知 MCU 进行处理。一旦 OCP、OVP 或 UVP 状况解除，PWM 将恢复输出。关于过电压保护及过电压 / 欠电压保护的更多内容，可参考“过电流保护”和“过电压 / 欠电压保护”章节。

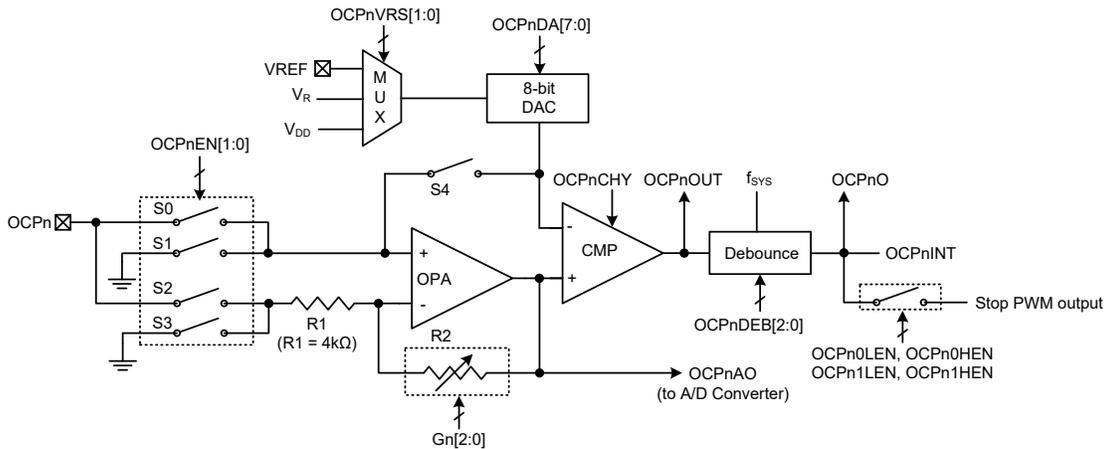
编程注意事项

读写流程如下步骤所示：

- 写数据至 DLLn/PWMnD
 - ◆ 步骤 1. 写数据至 DLLn
 - 注意，此时数据仅写入 4-bit 缓存器。
 - ◆ 步骤 2. 写数据至 PWMnD
 - 注意，此时数据直接写入 PWMnD 寄存器，同时锁存在 4-bit 缓存器的数据写入 DLLn 寄存器。
- 从 DLLn/PWMnD 寄存器中读取数据
 - ◆ 步骤 1. 从 PWMnD 中读取数据
 - 注意，此时 PWMnD 寄存器中的数据可直接读取，同时从 DLLn 读取的数据锁存在 4-bit 缓存器中。
 - ◆ 步骤 2. 从 DLLn 中读取数据
 - 注意，此时读取 4-bit 缓存器中的数据。

过电流保护 – OCP

该单片机中内建过电流保护功能，为电池充电及放电安全应用提供了保障。为防止电池充电或负载电流超出指定的值，过电流保护电路通过内部运算放大器将从 OCPn 引脚输入的电流转换为对应的电压值信号，输入到比较器端与内部 8-bit D/A 转换器输出的电压值进行比较，从而判断是否有过电流状况。当检测到过电流时，若对应的中断使能，将会产生 OCPn 中断。



注：V_R 为 A/D 转换器 PGA 输出，OCPnAO 可作为 A/D 转换器输入信号。

OCP 电路方框图 (n=0~1)

OCP 操作

OCPn 电路可对输入的电流进行监测，防止输入的电流超过指定的参考值。从 OCPn 引脚输入的电流先被转换为一个电压信号并通过 OCPn 电路内置的可编程增益运算放大器 PGA 进行放大 (1~50 倍增益，通过 OCPnC1 寄存器的 Gn2~Gn0 位设置)。通过设置 OCPnC0 寄存器中的 OCPnEN1 和 OCPnEN0 位，PGA 可工作在同相，反相或输入失调校准模式。放大后的电压信号输入到内置的比较器电路并与 8-bit D/A 转换器提供的电压进行比较。其中 D/A 转换器参考电压可通过 OCPnC0 寄存器的 OCPnVRS[1:0] 位选择来自 V_{DD}，V_R 或外部

输入电压 V_{REF} 。比较器输出的 OCPnCOU 先通过去抖电路处理 (去抖周期由 OCPnC1 寄存器的 OCPnDEB[2:0] 位选择), 之后输出 OCPnO 信号, 用于指明是否有过电流状况发生。OCPnO 输出为 1, 则表明有过电流情况, 若无过电流情况, 则 OCPnO 输出为 0。一旦发生过电流情况, 即 OCPn 输入电流转换得到的电压高于参考电压, 此时若相关中断使能, 则会产生过电流保护中断。

此单片机对高精度 PWM 发生器输出信号 OUT0H/OUT0L, OUT1H/OUT1L 提供了过电流保护控制, 此功能通过 OCPPC 寄存器设置。若 PWMn 输出过电流保护功能使能, 则当过电流情况发生时, 这些信号将被强制设置为无效状态。同时 OCPn 也会产生中断信号以告知 MCU。当过电流情况解除后, OUT0H/OUT0L, OUT1H/OUT1L 将会恢复正常输出并继续产生 PWM 信号。

关于 OCPn 发生时 OUTnH 和 OUTnL 信号极性和输出控制的详细内容将在 OCPPC 寄存器中描述。

OCP 寄存器介绍

OCP 功能的所有操作由一系列的寄存器控制。一个寄存器用于为 OCP 电路提供参考电压。两个寄存器用于运算放大器和比较器输入失调校准。两个控制寄存器用于控制 OCPn 功能, D/A 转换器参考电压选择, PGA 增益选择, 比较器去抖动时间以及迟滞功能。剩下的 OCPPC 寄存器用于设置在 OCPn 状况发生时是否将 PWMn 输出信号 OUT0H/OUT0L, OUT1H/OUT1L 强制设置为无效状态。

寄存器名称	位							
	7	6	5	4	3	2	1	0
OCPnC0	OCPnEN1	OCPnEN0	OCPnVRS1	OCPnVRS0	OCPnCHY	—	—	OCPnO
OCPnC1	—	—	Gn2	Gn1	Gn0	OCPnDEB2	OCPnDEB1	OCPnDEB0
OCPnDA	D7	D6	D5	D4	D3	D2	D1	D0
OCPnOCAL	OCPnOOFM	OCPnORSP	OCPnOOF5	OCPnOOF4	OCPnOOF3	OCPnOOF2	OCPnOOF1	OCPnOOF0
OCPnCCAL	OCPnCOU	OCPnCOFM	OCPnCRSP	OCPnCOF4	OCPnCOF3	OCPnCOF2	OCPnCOF1	OCPnCOF0
OCPPC	OCP11LEN	OCP11HEN	OCP10LEN	OCP10HEN	OCP01LEN	OCP01HEN	OCP00LEN	OCP00HEN

OCPn 寄存器列表 (n=0~1)

• OCPnC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OCPnEN1	OCPnEN0	OCPnVRS1	OCPnVRS0	OCPnCHY	—	—	OCPnO
R/W	R/W	R/W	R/W	R/W	R/W	—	—	R
POR	0	0	0	0	0	—	—	0

Bit 7~6 **OCPnEN1~OCPnEN0**: OCPn 工作模式选择
 00: OCPn 除能, S1 和 S3 on, S0 和 S2 off
 01: 同相模式, S0 和 S3 on, S1 和 S2 off
 10: 反相模式, S1 和 S2 on, S0 和 S3 off
 11: 校准模式, S1 和 S3 on, S0 和 S2 off

Bit 5~4 **OCPnVRS1~OCPnVRS0**: OCPn DAC 参考电压选择
 00/11: 来自 V_{DD}
 01: 来自 V_{REF} 引脚输入
 10: 来自 V_R

注: 若设置 OCPnVRS[1:0] 位为“10”选择 OCPn DAC 参考电压来自内部 V_R 。需注意因 V_R 为 A/D 转换器内部 PGA 输出, 因此需先设置 ADPGAEN 位为高以使能 PGA。

- Bit 3 **OCNCHY**: OCPn 比较器迟滞功能控制位
 0: 除能
 1: 使能
- Bit 2~1 未定义, 读为 “0”
- Bit 0 **OCN0**: OCPn 数字输出位
 0: 没有过电流情况发生
 1: 发生过电流情况

● **OCN1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	Gn2	Gn1	Gn0	OCnDEB2	OCnDEB1	OCnDEB0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 未定义, 读为 “0”

- Bit 5~3 **Gn2~Gn0**: R2/R1 比率选择
 000: 单位增益缓冲器 (同相模式) 或 R2/R1=1 (反相模式)
 001: R2/R1=5
 010: R2/R1=10
 011: R2/R1=15
 100: R2/R1=20
 101: R2/R1=30
 110: R2/R1=40
 111: R2/R1=50

这三位用于选择 R2/R1 比率以获得反相或同相模式时的增益值。反相和同相模式中的 OCPn PGA 增益计算公式请参见“输入电压范围”章节。

- Bit 2~0 **OCnDEB2~OCnDEB 0**: OCPn 输出滤波去抖动时间选择
 000: 旁路, 无去抖动
 001: $(1\sim2) \times t_{DEB}$
 010: $(3\sim4) \times t_{DEB}$
 011: $(7\sim8) \times t_{DEB}$
 100: $(15\sim16) \times t_{DEB}$
 101: $(31\sim32) \times t_{DEB}$
 110: $(63\sim64) \times t_{DEB}$
 111: $(127\sim128) \times t_{DEB}$

注: $t_{DEB}=1/f_{SYS}$ 。

● **OCnDA 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**: OCPn DAC 输出电压控制位
 OCP DAC 输出 = (DAC 参考电压 / 256) × D[7:0]

• OCPnOCAL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OCPnOOFM	OCPnORS	OCPnOOF5	OCPnOOF4	OCPnOOF3	OCPnOOF2	OCPnOOF1	OCPnOOF0
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 **OCPnOOFM:** OCPn 运算放大器正常工作或输入失调校准模式选择位
 0: 正常工作模式
 1: 输入失调校准模式
 此位用来选择 OCPn 运算放大器输入校准功能。在设置此位之前, 应先设置 OCPnEN[1:0] 位为“11”后才能设置 OCPnOOFM 为 1 以选择输入失调校准模式, 之后需将 OCPnCOFM 位清零, 运算放大器工作于输入失调校准模式。更具体操作参考“运算放大器输入失调校准”章节。
- Bit 6 **OCPnORS:** OCPn 运算放大器输入失调校准参考电压输入选择
 0: 选择负输入作为参考输入
 1: 选择正输入作为参考输入
- Bit 5~0 **OCPnOOF5~OCPnOOF0:** OCPn 运算放大器输入失调电压校准值
 这 6 位的寄存器位供运算放大器输入失调校准操作使用, 并用于重新储存 OCPn 运算放大器输入失调校准值。更多详细内容可参考“运算放大器输入失调校准”章节。

• OCPnCCAL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OCPnCOUT	OCPnCOFM	OCPnCRSP	OCPnCOF4	OCPnCOF3	OCPnCOF2	OCPnCOF1	OCPnCOF0
R/W	R	R/W						
POR	0	0	0	1	0	0	0	0

- Bit 7 **OCPnCOUT:** OCPn 比较器输出, 正逻辑 (只读)
 0: 正端输入电压 < 负端输入电压
 1: 正端输入电压 > 负端输入电压
 此位用于当 OCPn 工作在输入失调校准模式时, 显示正输入电压是否大于负输入电压。若 OCPnCOUT 设为“1”, 表示正输入电压大于负输入电压, 否则正输入电压小于负输入电压。
- Bit 6 **OCPnCOFM:** OCPn 比较器正常工作或输入失调校准模式选择位
 0: 正常工作模式
 1: 输入失调校准模式
 此位用于选择 OCPn 比较器输入失调校准功能。在设置此位之前, 应先设置 OCPnEN[1:0] 位为“11”然后才能设置 OCPnCOFM 为 1 以选择输入失调校准模式, 之后需将 OCPnOOFM 位清零, 比较器输入失调校准模式使能。更具体操作参考“比较器输入失调校准”章节。
- Bit 5 **OCPnCRSP:** OCPn 比较器输入失调校准参考电压输入选择
 0: 选择负输入作为参考输入
 1: 选择正输入作为参考输入
- Bit 4~0 **OCPnCOF4~OCPnCOF0:** OCPn 比较器输入失调校准值
 这 5 位的寄存器位供比较器输入失调校准操作时使用, 并用于重复储存校准值。更多详细内容可参考“比较器输入失调校准”章节。

• OCPPC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OCP11LEN	OCP11HEN	OCP10LEN	OCP10HEN	OCP01LEN	OCP01HEN	OCP00LEN	OCP00HEN
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 OCP11LEN:** OUT1L 过电流保护 1 使能控制
 0: 除能
 1: 使能
 此位用于控制 OUT1L 信号的过电流保护 1 功能。若清除此位为 0, 除能 OUT1L 的过电流保护 1 功能, 则当有 OCP1 状况发生时, OUT1L 的输出不受影响。若设置此位为 1, 使能 OUT1L 的过电流保护 1 功能, 则当有 OCP1 状况发生时, OUT1L 的输出将由 OUTPC 寄存器的 OUT1LS 位控制。
- Bit 6 OCP11HEN:** OUT1H 过电流保护 1 使能控制
 0: 除能
 1: 使能
 此位用于控制 OUT1H 信号的过电流保护 1 功能。若清除此位为 0, 除能 OUT1H 的过电流保护 1 功能, 则当有 OCP1 状况发生时, OUT1H 的输出不受影响。若设置此位为 1, 使能 OUT1H 的过电流保护 1 功能, 则当有 OCP1 状况发生时, OUT1H 的输出将由 OUTPC 寄存器的 OUT1HS 位控制。
- Bit 5 OCP10LEN:** OUT0L 过电流保护 1 使能控制
 0: 除能
 1: 使能
 此位用于控制 OUT0L 信号的过电流保护 1 功能。若清除此位为 0, 除能 OUT0L 的过电流保护 1 功能, 则当有 OCP1 状况发生时, OUT0L 的输出不受影响。若设置此位为 1, 使能 OUT0L 的过电流保护 1 功能, 则当有 OCP1 状况发生时, OUT0L 的输出将由 OUTPC 寄存器的 OUT0LS 位控制。
- Bit 4 OCP10HEN:** OUT0H 过电流保护 1 使能控制
 0: 除能
 1: 使能
 此位用于控制 OUT0H 信号的过电流保护 1 功能。若清除此位为 0, 除能 OUT0H 的过电流保护 1 功能, 则当有 OCP1 状况发生时, OUT0H 的输出不受影响。若设置此位为 1, 使能 OUT0H 的过电流保护 1 功能, 则当有 OCP1 状况发生时, OUT0H 的输出将由 OUTPC 寄存器的 OUT0HS 位控制。
- Bit 3 OCP01LEN:** OUT1L 过电流保护 0 使能控制
 0: 除能
 1: 使能
 此位用于控制 OUT1L 信号的过电流保护 0 功能。若清除此位为 0, 除能 OUT1L 的过电流保护 0 功能, 则当有 OCP0 状况发生时, OUT1L 的输出不受影响。若设置此位为 1, 使能 OUT1L 的过电流保护 0 功能, 则当有 OCP0 状况发生时, OUT1L 的输出将由 OUTPC 寄存器的 OUT1LS 位控制。
- Bit 2 OCP01HEN:** OUT1H 过电流保护 0 使能控制
 0: 除能
 1: 使能
 此位用于控制 OUT1H 信号的过电流保护 0 功能。若清除此位为 0, 除能 OUT1H 的过电流保护 0 功能, 则当有 OCP0 状况发生时, OUT1H 的输出不受影响。若设置此位为 1, 使能 OUT1H 的过电流保护 0 功能, 则当有 OCP0 状况发生时, OUT1H 的输出将由 OUTPC 寄存器的 OUT1HS 位控制。
- Bit 1 OCP00LEN:** OUT0L 过电流保护 0 使能控制
 0: 除能
 1: 使能
 此位用于控制 OUT0L 信号的过电流保护 0 功能。若清除此位为 0, 除能 OUT0L 的过电流保护 0 功能, 则当有 OCP0 状况发生时, OUT0L 的输出不受影响。若

设置此位为 1，使能 OUT0L 的过电流保护 0 功能，则当有 OCP0 状况发生时，OUT0L 的输出将由 OUTPC 寄存器的 OUT0LS 位控制。

Bit 0 **OCPO0HEN**: OUT0H 过电流保护 0 使能控制

0: 除能
1: 使能

此位用于控制 OUT0H 信号的过电流保护 0 功能。若清除此位为 0，除能 OUT0H 的过电流保护 0 功能，则当有 OCP0 状况发生时，OUT0H 的输出不受影响。若设置此位为 1，使能 OUT0H 的过电流保护 0 功能，则当有 OCP0 状况发生时，OUT0H 的输出将由 OUTPC 寄存器的 OUT0HS 位控制。

输入电压范围

为了操作的灵活性，在不同的 PGA 操作模式下，OCPn 引脚上的输入电压可以为正或者为负。正输入或负输入电压的 PGA 输出基于不同的公式计算如下：

- 输入电压 $V_{IN} > 0$ ，PGA 操作于同相模式下，PGA 输出可由以下公式计算：

$$V_{OUT} = (1 + R_2/R_1) \times V_{IN}$$

- 将 OCPnEN[1:0] 的值设为“01”使 PGA 工作在同相模式，并设置 Gn[2:0] 的值为“000”以选择单位增益，此时 PGA 将作为一个单位增益缓冲器，其输出与 V_{IN} 相等。

$$V_{OUT} = V_{IN}$$

- 输入电压 $0 > V_{IN} > -0.4V$ 时，工作在反相模式下的 PGA 以及 PGA 输出可由以下公式获得。

$$V_{OUT} = -(R_2/R_1) \times V_{IN}$$

注意，若输入电压为负，其值不可小于 -0.4V，否则会产生漏电流。

OCPn 运算放大器和比较器失调校准

OCPn 电路可工作于四种操作模式，通过 OCPnEN[1:0] 字段进行选择。其中一种就是校准模式。在校准模式下，可对运算放大器和比较器进行失调校准。

运算放大器输入失调校准

步骤 1. 设置 OCPnEN[1:0]=11，OCPnOOFM=1，OCPnCOFM=0，OCPn 将工作在运算放大器输入失调校准模式。

步骤 2. 设置 OCPnOOF[5:0]=000000B，此时开始读 OCPnCOUT 位。

步骤 3. OCPnOOF[5:0] 的值加一，读 OCPnCOUT 位。

若 OCPnCOUT 位状态不变，则重复步骤 3 直到 OCPnCOUT 位状态改变。

若 OCPnCOUT 位状态改变，则记录下 OCPnOOF 值为 V_{OOS1} ，前往步骤 4。

步骤 4. 设置 OCPnOOF[5:0]=111111B，此时开始读 OCPnCOUT 位。

步骤 5. OCPnOOF[5:0] 的值减一，读 OCPnCOUT 位。

若 OCPnCOUT 位状态未改变，重复步骤 5 直到 OCPnCOUT 位状态改变。

若 OCPnCOUT 位状态改变，则记录下 OCPnOOF 值为 V_{OOS2} ，前往步骤 6。

步骤 6. 将运算放大器输入失调校准值 V_{OOS} ，重新存入 OCPnOOF[5:0] 字段。此时失调校准步骤完成。

$$\text{其中 } V_{OOS} = (V_{OOS1} + V_{OOS2}) / 2$$

比较器输入失调校准

步骤 1. 设置 OCPnEN[1:0]=11，OCPnCOFM=1 且 OCPnOOFM=0，OCPn 将工作在比较器输入失调校准模式。S4 为 on (S4 用于校准模式，在正常模式操作下为 off)。

- 步骤 2. 设置 OCPnCOF[4:0]=00000B, 此时开始读 OCPnCOUT 位。
- 步骤 3. OCPnCOF[4:0] 的值加一, 读 OCPnCOUT 位。
若 OCPnCOUT 位状态不变, 则重复步骤 3 直到 OCPnCOUT 位状态改变。
若 OCPnCOUT 位状态改变, 则记录下 OCPnCOF 值为 V_{COS1} , 前往步骤 4。
- 步骤 4. 设置 OCPnCOF[4:0]=11111B, 此时开始读 OCPnCOUT 位。
- 步骤 5. OCPnCOF[4:0] 的值减一, 读 OCPnCOUT 位。
若 OCPnCOUT 位状态未改变, 重复步骤 5 直到 OCPnCOUT 位状态改变。
若 OCPnCOUT 位状态改变, 则记录下 OCPnCOF 值为 V_{COS2} , 前往步骤 6。
- 步骤 6. 将比较器输入失调校准值 V_{COS} , 重新存入 OCPnCOF[4:0] 字段。此时失调校准步骤完成。

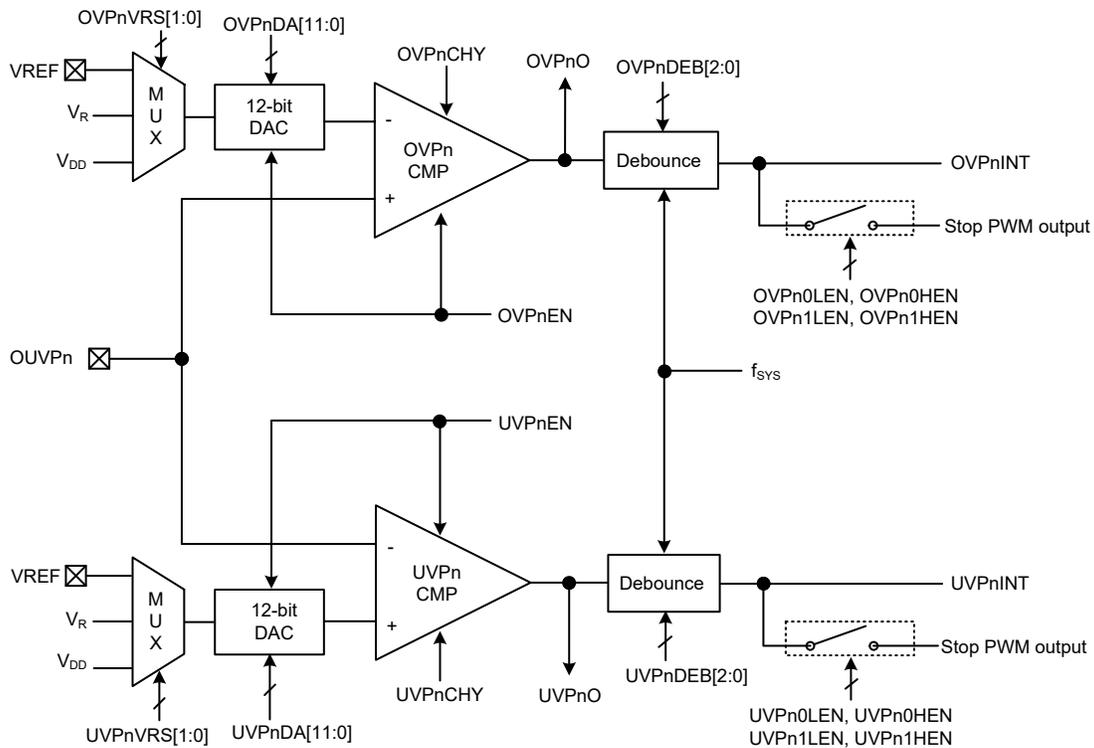
$$\text{其中 } V_{COS} = (V_{COS1} + V_{COS2}) / 2$$

过电压 / 欠电压保护 – OUVP

该单片机内置过电压 / 欠电压保护电路, 用于电池充 / 放电应用的安全防护。

OUVP 电路操作

OUVP 电路内建了过电压保护 (OVPn) 和欠电压 (UVPn) 保护功能。



注: V_R 为 A/D 转换器单位增益缓冲器输出。

过电压 / 欠电压保护电路方框图 (n=0~1)

过电压保护功能

为防止输出电压超过指定的电压值，过电压保护电路将 OVPn 输入电压与 12-bit D/A 转换器提供的参考电压进行比较。12-bit DAC 参考输入信号可通过 OVPnVRS[1:0] 位选择来自 V_{DD}，V_R 或外部 VREF 引脚输入。一旦 OVPn 输入电压值大于设定的参考电压值，OVPnO 的值将从 0 变为 1。OVPnINT 为 OVPnO 经过去抖处理后的输出，用于指示 OVPn 引脚上的输入电压是否超过了参考电压值。OVPnO 为 OVPn 输出，OVPnINT 则为 OVPn 中断触发信号。通过设置 OVPnCHY 位可启用 OVPn 电路的比较器迟滞功能。

欠电压保护功能

为防止输出电压低于指定的电压值，欠电压保护电路将 UVPn 输入电压与 12-bit D/A 转换器提供的参考电压进行比较。12-bit DAC 参考输入信号可通过 OVPnVRS[1:0] 位选择来自 V_{DD}，V_R 或外部 VREF 引脚输入。一旦 UVPn 输入电压值低于设定的参考电压值，UVPnO 的值将从 0 变为 1。UVPnINT 为 UVPnO 经过去抖处理后的输出，用于指示 UVPn 引脚上的输入电压是否低于参考电压值。UVPnO 为 UVPn 输出，UVPnINT 则为 UVPn 中断触发信号。通过设置 UVPnCHY 位可启用 UVPn 电路的比较器迟滞功能。

此单片机针对高精度 PWM 发生器输出 OUT0H/OUT0L，OUT1H/OUT1L 信号提供了过电压 / 欠电压保护控制，此功能通过 OUVpPC 寄存器设置。若 PWMn 输出过电压 / 欠电压保护功能使能，则当有过电压或欠电压情况发生时，这些信号将被强制设置为无效状态。OVPn/UVPn 同时也会产生中断信号以告知 MCU。当过电压或欠电压情况解除后，OUT0H/OUT0L，OUT1H/OUT1L 将会恢复正常并继续输出 PWM 信号。

当发生 OVPn 或 UVPn 状况时 OUTnH 和 OUTnL 信号极性和输出控制将在 OUVpPC 寄存器中描述。

OUVPn 寄存器介绍

过电压和欠电压保护的整个操作由一系列寄存器控制。

寄存器名称	位							
	7	6	5	4	3	2	1	0
OUVPnC0	—	—	OVPnEN	OVPnCHY	OVPnVRS1	OVPnVRS0	OVPnDEB1	OVPnDEB0
OUVPnC1	—	—	UVPnEN	UVPnCHY	UVPnVRS1	UVPnVRS0	UVPnDEB1	UVPnDEB0
OUVPnC2	OVPnO	OVPnCOFM	OVPnCRS	OVPnCOF4	OVPnCOF3	OVPnCOF2	OVPnCOF1	OVPnCOF0
OUVPnC3	UVPnO	UVPnCOFM	UVPnCRS	UVPnCOF4	UVPnCOF3	UVPnCOF2	UVPnCOF1	UVPnCOF0
OVPnDAH	—	—	—	—	D11	D10	D9	D8
OVPnDAL	D7	D6	D5	D4	D3	D2	D1	D0
UVPnDAH	—	—	—	—	D11	D10	D9	D8
UVPnDAL	D7	D6	D5	D4	D3	D2	D1	D0
OUVPnPC	UVPn1LEN	UVPn1HEN	UVPn0LEN	UVPn0HEN	OVPn1LEN	OVPn1HEN	OVPn0LEN	OVPn0HEN

OUVPn 寄存器列表 (n=0~1)

● OVPnDAH&OVPnDAL 寄存器

寄存器	OVPnDAH								OVPnDAL							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	—	—	—	—	D11	D10	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	R/W
POR	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

“—”：未定义，读为“0”

D11~D8: OVPn DAC 参考值选择高字节

D7~D0: OVPn DAC 参考值选择低字节

OVPn DAC 输出 = (OVPn DAC $V_{REF}/4096$) × D[11:0]

注：写 OVPnDAL 寄存器数据仅会被写入阴影寄存器，写 OVPnDAH 寄存器时阴影寄存器中的数据将被复制到 OVPnDAL 寄存器中。

● UVPnDAH&UVPnDAL 寄存器

寄存器	UVPnDAH								UVPnDAL							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	—	—	—	—	D11	D10	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	R/W
POR	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

“—”：未定义，读为“0”

D11~D8: UVPn DAC 参考值选择高字节

D7~D0: UVPn DAC 参考值选择低字节

UVPn DAC 输出 = (UVPn DAC $V_{REF}/4096$) × D[11:0]

注：写 UVPnDAL 寄存器数据仅会被写入阴影寄存器，写 UVPnDAH 寄存器时阴影寄存器中的数据将被复制到 UVPnDAL 寄存器中。

● OUVPnC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	OVPnEN	OVPnCHY	OVPnVRS1	OVPnVRS0	OVPnDEB1	OVPnDEB0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 未定义，读为“0”

Bit 5 **OVPnEN**: OVPn 功能使能控制

0: 除能
1: 使能

若 OVPnEN=0, 过电压保护 n 功能除能, 则不会产生功耗, 同时 OVPn 电路的比较器和 DAC 也会关闭。

Bit 4 **OVPnCHY**: OVPn 比较器迟滞功能控制

0: 除能
1: 使能

Bit 3~2 **OVPnVRS1~OVPnVRS0**: OVPn DAC 参考电压选择

00: 来自 V_{DD}
01: 来自 VREF 引脚输入
10: 来自内部 V_R
11: 来自 V_{DD}

注：若设置 OVPnVRS[1:0] 位为“10”选择 OVPn DAC 参考电压来自内部 V_R 。需注意因 V_R 为 A/D 转换器 PGA 输出, 因此需先设置 ADPGAEN 位为高以使能内部 PGA。

- Bit 1~0 **OVPnDEB1~OVPnDEB0:** OVPn 比较器去抖时间选择
 00: 无去抖
 01: $(7\sim 8) \times 1/f_{SYS}$
 10: $(15\sim 16) \times 1/f_{SYS}$
 11: $(31\sim 32) \times 1/f_{SYS}$

● **OUPnPnC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	UVPnEN	UVPnCHY	UVPnVRS1	UVPnVRS0	UVPnDEB1	UVPnDEB0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

- Bit 7~6 未定义，读为“0”
- Bit 5 **UVPnEN:** UVPn 功能使能控制
 0: 除能
 1: 使能
 若 UVPnEN=0, 欠电压保护 n 功能除能, 则不会产生功耗, 同时 UVPn 电路的比较器和 DAC 也会关闭。
- Bit 4 **UVPnCHY:** UVPn 比较器迟滞功能控制
 0: 除能
 1: 使能
- Bit 3~2 **UVPnVRS1~UVPnVRS0:** UVPn DAC 参考电压选择
 00: 来自 V_{DD}
 01: 来自 VREF 引脚输入
 10: 来自内部 V_R
 11: 来自 V_{DD}
 注: 若设置 UVPnVRS[1:0] 位为“10”选择 UVPn DAC 参考电压来自内部 V_R 。需注意因 V_R 为 A/D 转换器 PGA 输出, 因此需先设置 ADPGAEN 位为高以使能内部 PGA。
- Bit 1~0 **UVPnDEB1~UVPnDEB 0:** UVPn 比较器去抖时间选择
 00: 无去抖
 01: $(7\sim 8) \times 1/f_{SYS}$
 10: $(15\sim 16) \times 1/f_{SYS}$
 11: $(31\sim 32) \times 1/f_{SYS}$

● **OUPnPnC2 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	OVPnO	OVPnCOFM	OVPnCRS	OVPnCOF4	OVPnCOF3	OVPnCOF2	OVPnCOF1	OVPnCOF0
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	1	0	0	0	0

- Bit 7 **OVPnO:** OVPn 比较器输出位
 0: 正端输入电压 < 负端输入电压
 1: 正端输入电压 > 负端输入电压
- Bit 6 **OVPnCOFM:** OVPn 比较器正常工作或输入失调校准模式选择位
 0: 正常工作模式
 1: 输入失调校准模式
- Bit 5 **OVPnCRS:** OVPn 比较器输入失调校准参考电压输入选择
 0: 选择负输入作为参考输入
 1: 选择正输入作为参考输入
 OVPnCRS 位用于选择参考输入电压来自 OVPn D/A 转换器或是来自外部输入。注意仅在设置 OVPnCOFM 位为 1 选择 OVPn 比较器工作在输入失调校准模式时, 此位的选择才有效。
- Bit 4~0 **OVPnCOF4~OVPnCOF0:** OVPn 比较器输入失调校准值

• OUVpnC3 寄存器

Bit	7	6	5	4	3	2	1	0
Name	UVpNO	UVpNCOFM	UVpNCRS	UVpNCOF4	UVpNCOF3	UVpNCOF2	UVpNCOF1	UVpNCOF0
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	1	0	0	0	0

- Bit 7 **UVpNO:** UVpN 比较器输出位
0: 正端输入电压 < 负端输入电压
1: 正端输入电压 > 负端输入电压
- Bit 6 **UVpNCOFM:** UVpN 比较器正常工作或输入失调校准模式选择位
0: 正常工作模式
1: 输入失调校准模式
- Bit 5 **UVpNCRS:** UVpN 比较器输入失调校准参考电压输入选择
0: 选择负输入作为参考输入
1: 选择正输入作为参考输入
UVpNCRS 位用于选择参考输入电压来自 UVpN D/A 转换器或是来自外部输入。注意仅在设置 UVpNCOFM 位为 1 选择 UVpN 比较器工作在输入失调校准模式时，此位的选择才有效。
- Bit 4~0 **UVpNCOF4~UVpNCOF0:** UVpN 比较器输入失调校准值

• OUVpNPC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	UVpN1LEN	UVpN1HEN	UVpN0LEN	UVpN0HEN	OVpN1LEN	OVpN1HEN	OVpN0LEN	OVpN0HEN
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 **UVpN1LEN:** OUT1L 欠电压保护 n 使能控制
0: 除能
1: 使能
此位用于控制 OUT1L 信号的欠电压保护 n 功能。若清除此位为 0，除能 OUT1L 的欠电压保护 n 功能，则当有 UVpN 状况发生时，OUT1L 的输出不受影响。若设置此位为 1，使能 OUT1L 的欠电压保护 n 功能，则当有 UVpN 状况发生时，OUT1L 的输出将由 OUTPC 寄存器的 OUT1LS 位控制。
- Bit 6 **UVpN1HEN:** OUT1H 欠电压保护 n 使能控制
0: 除能
1: 使能
此位用于控制 OUT1H 信号的欠电压保护 n 功能。若清除此位为 0，除能 OUT1H 的欠电压保护 n 功能，则当有 UVpN 状况发生时，OUT1H 的输出不受影响。若设置此位为 1，使能 OUT1H 的欠电压保护 n 功能，则当有 UVpN 状况发生时，OUT1H 的输出将由 OUTPC 寄存器的 OUT1HS 位控制。
- Bit 5 **UVpN0LEN:** OUT0L 欠电压保护 n 使能控制
0: 除能
1: 使能
此位用于控制 OUT0L 信号的欠电压保护 n 功能。若清除此位为 0，除能 OUT0L 的欠电压保护 n 功能，则当有 UVpN 状况发生时，OUT0L 的输出不受影响。若设置此位为 1，使能 OUT0L 的欠电压保护 n 功能，则当有 UVpN 状况发生时，OUT0L 的输出将由 OUTPC 寄存器的 OUT0LS 位控制。
- Bit 4 **UVpN0HEN:** OUT0H 欠电压保护 n 使能控制
0: 除能
1: 使能
此位用于控制 OUT0H 信号的欠电压保护 n 功能。若清除此位为 0，除能 OUT0H 的欠电压保护 n 功能，则当有 UVpN 状况发生时，OUT0H 的输出不受

	影响。若设置此位为 1，使能 OUT0H 的欠电压保护 n 功能，则当有 UVPn 状况发生时，OUT0H 的输出将由 OUTPC 寄存器的 OUT0HS 位控制。
Bit 3	<p>OVPn1LEN: OUT1L 过电压保护 n 使能控制</p> <p>0: 除能 1: 使能</p> <p>此位用于控制 OUT1L 信号的过电压保护 n 功能。若清除此位为 0，除能 OUT1L 的过电压保护 n 功能，则当有 OVPn 状况发生时，OUT1L 的输出不受影响。若设置此位为 1，使能 OUT1L 的过电压保护 n 功能，则当有 OVPn 状况发生时，OUT1L 的输出将由 OUTPC 寄存器的 OUT1LS 位控制。</p>
Bit 2	<p>OVPn1HEN: OUT1H 过电压保护 n 使能控制</p> <p>0: 除能 1: 使能</p> <p>此位用于控制 OUT1H 信号的过电压保护 n 功能。若清除此位为 0，除能 OUT1H 的过电压保护 n 功能，则当有 OVPn 状况发生时，OUT1H 的输出不受影响。若设置此位为 1，使能 OUT1H 的过电压保护 n 功能，则当有 OVPn 状况发生时，OUT1H 的输出将由 OUTPC 寄存器的 OUT1HS 位控制。</p>
Bit 1	<p>OVPn0LEN: OUT0L 过电压保护 n 使能控制</p> <p>0: 除能 1: 使能</p> <p>此位用于控制 OUT0L 信号的过电压保护 n 功能。若清除此位为 0，除能 OUT0L 的过电压保护 n 功能，则当有 OVPn 状况发生时，OUT0L 的输出不受影响。若设置此位为 1，使能 OUT0L 的过电压保护 n 功能，则当有 OVPn 状况发生时，OUT0L 的输出将由 OUTPC 寄存器的 OUT0LS 位控制。</p>
Bit 0	<p>OVPn0HEN: OUT0H 过电压保护 n 使能控制</p> <p>0: 除能 1: 使能</p> <p>此位用于控制 OUT0H 信号的过电压保护 n 功能。若清除此位为 0，除能 OUT0H 的过电压保护 n 功能，则当有 OVPn 状况发生时，OUT0H 的输出不受影响。若设置此位为 1，使能 OUT0H 的过电压保护 n 功能，则当有 OVPn 状况发生时，OUT0H 的输出将由 OUTPC 寄存器的 OUT0HS 位控制。</p>

OVPn 和 UVPn 比较器失调校准

OVPn 和 UVPn 比较器都具有输入失调校准功能。在失调校准之前，应先设置 OVPnCHY 或 UVPnCHY 位为零，以除能 OVPn 或 UVPn 比较器迟滞功能。因 OUVpn 输入引脚与其它功能共用引脚，应正确设置相关的引脚共用功能选择寄存器以选择引脚功能为 OUVpn 比较器输入功能。以下为建议的 OVPn 和 UVPn 比较器失调校准步骤。

OVPn 比较器输入失调校准

- 步骤 1. 设置 OVPnCOFM=1，OVPnCRS=1 则 OVPn 比较器将工作在输入失调校准模式。为确保校准后， V_{OS} 值尽可能的小，失调校准时输入的参考电压应与比较器正常工作时输入的直流工作电压一致。
- 步骤 2. 设置 OVPnCOF[4:0]=00000B，此时开始读 OVPnO 位。
- 步骤 3. OVPnCOF[4:0] 的值加一，读 OVPnO 位。
若 OVPnO 位状态不变，则重复步骤 3 直到 OVPnO 位状态改变。
若 OVPnO 位状态改变，则记录下 OVPnCOF 值为 V_{OS1} ，前往步骤 4。
- 步骤 4. 设置 OVPnCOF[4:0]=11111B，此时开始读 OVPnO 位。
- 步骤 5. OVPnCOF[4:0] 的值减一，读 OVPnO 位。
若 OVPnO 位状态未改变，重复步骤 5 直到 OVPnO 位状态改变。
若 OVPnO 位状态改变，则记录下 OCPnCOF 值为 V_{OS2} ，前往步骤 6。

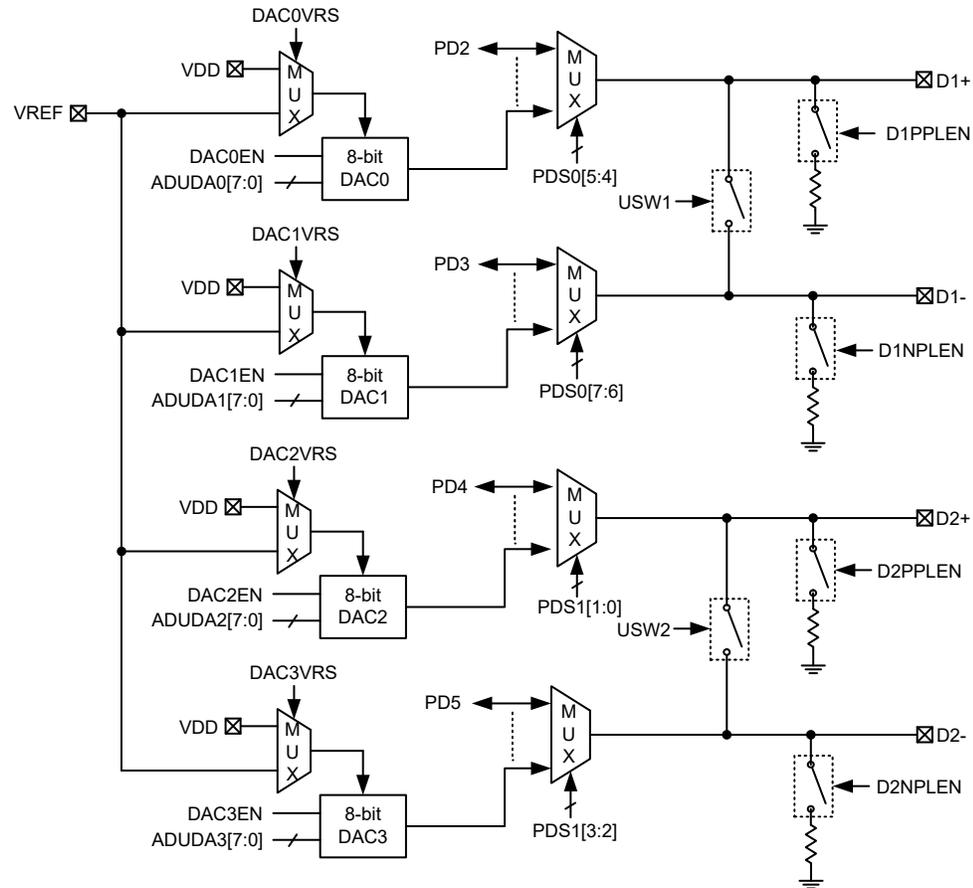
步骤 6. 将比较器输入失调校准值 $V_{OS}=(V_{OS1}+V_{OS2})/2$ ，重新存入 OVPnCOF[4:0] 字段。此时失调校准步骤完成。若 V_{OS} 的值不是整数，则去掉小数部分，保留 $V_{OS}=V_{OUT} - V_{IN}$ (1)

UVPn 比较器输入失调校准

- 步骤 1. 设置 UVPnCOFM=1，UVPnCRS=1 则 UVPn 比较器将工作在输入失调校准模式。为确保校准后， V_{OS} 值尽可能的小，失调校准时输入的参考电压应与比较器正常工作时输入的直流工作电压一致。
- 步骤 2. 设置 UVPnCOF[4:0]=00000B，此时开始读 UVPnO 位。
- 步骤 3. UVPnCOF[4:0] 的值加一，读 UVPnO 位。
若 UVPnO 位状态不变，则重复步骤 3 直到 UVPnO 位状态改变。
若 UVPnO 位状态改变，则记录下 UVPnCOF 值为 V_{OS1} ，前往步骤 4。
- 步骤 4. 设置 UVPnCOF[4:0]=11111B，此时开始读 UVPnO 位。
- 步骤 5. UVPnCOF[4:0] 的值减一，读 UVPnO 位。
若 UVPnO 位状态未改变，重复步骤 5 直到 UVPnO 位状态改变。
若 UVPnO 位状态改变，则记录下 UCPnCOF 值为 V_{OS2} ，前往步骤 6。
- 步骤 6. 将比较器输入失调校准值 $V_{OS}=(V_{OS1}+V_{OS2})/2$ ，重新存入 UVPnCOF[4:0] 字段。此时失调校准步骤完成。若 V_{OS} 的值不是整数，则去掉小数部分，保留 $V_{OS}=V_{OUT} - V_{IN}$ (2)

USB 自动检测

此单片机包含两组 USB 端口 D1+/D1- 和 D2+/D2-，用于执行充 / 放电设备自动检测功能。用户可以通过监测 USB 接口端的电压与电流，识别连接到 USB 端口的是专用的充电器，便携设备，一般的 USB 接口或是带 USB 接口的充电装置。



USB 自动检测功能方框图

D1+/D1- 和 D2+/D2- 自动检测

USB 自动检测电路中，包含四个 8-bit D/A 转换器即 DAC0~DAC3。这四个 DACn 可通过 ADUC0 寄存器中的 DACnEN 位使能。DAC 输出的电压由 ADUC0 寄存器中的 DACnRV 位选择的 DACn 参考电压及相应的 8-bit 寄存器 ADUDAn 决定。在 D1+ 和 D1- 引脚间有一个由 USW1 位控制的模拟开关 USW1，同样的，在 D2+ 和 D2- 间也有一个由 USW2 位控制的模拟开关 USW2。但在设置这些开关时需注意只有当 D2+/D2- 其中一个引脚或 D1+/D1- 其中一个引脚通过引脚共用功能选择寄存器设置为模拟或数字输入类型时，且设置 USW2 或 USW1 为高，才能使相应的开关闭合。D1+/D1- 和 D2+/D2- 脚可通过设置 ADUC1 寄存器中的 D1NPLEN/D1PPLEN 和 D2NPLEN/D2PPLEN 位选择是否分别连接下拉电阻到地。

USB 自动检测寄存器

USB 自动检测功能是通过一系列寄存器进行控制的。

寄存器名称	位							
	7	6	5	4	3	2	1	0
ADUC0	DAC3VRS	DAC2VRS	DAC1VRS	DAC0VRS	DAC3EN	DAC2EN	DAC1EN	DAC0EN
ADUC1	—	—	D2NPLEN	D2PPLEN	D1NPLEN	D1PPLEN	USW2	USW1
ADUDA0	D7	D6	D5	D4	D3	D2	D1	D0
ADUDA1	D7	D6	D5	D4	D3	D2	D1	D0
ADUDA2	D7	D6	D5	D4	D3	D2	D1	D0
ADUDA3	D7	D6	D5	D4	D3	D2	D1	D0

USB 自动检测寄存器列表

• ADUC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	DAC3VRS	DAC2VRS	DAC1VRS	DAC0VRS	DAC3EN	DAC2EN	DAC1EN	DAC0EN
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 **DAC3VRS:** DAC3 参考电压选择
0: VDD 引脚输入
1: VREF 引脚输入

Bit 6 **DAC2VRS:** DAC2 参考电压选择
0: VDD 引脚输入
1: VREF 引脚输入

Bit 5 **DAC1VRS:** DAC1 参考电压选择
0: VDD 引脚输入
1: VREF 引脚输入

Bit 4 **DAC0VRS:** DAC0 参考电压选择
0: VDD 引脚输入
1: VREF 引脚输入

Bit 3 **DAC3EN:** DAC3 使能控制
0: 除能
1: 使能

Bit 2 **DAC2EN:** DAC2 使能控制
0: 除能
1: 使能

Bit 1 **DAC1EN:** DAC1 使能控制
0: 除能
1: 使能

Bit 0 **DAC0EN:** DAC0 使能控制
0: 除能
1: 使能

• ADUC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	D2NPLEN	D2PPLEN	D1NPLEN	D1PPLEN	USW2	USW1
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 未定义，读为“0”

Bit 5 **D2NPLEN**: D2- 引脚下拉控制位
0: 除能
1: 使能

Bit 4 **D2PPLEN**: D2+ 引脚下拉控制位
0: 除能
1: 使能

Bit 3 **D1NPLEN**: D1- 引脚下拉控制位
0: 除能
1: 使能

Bit 2 **D1PPLEN**: D1+ 引脚下拉控制位
0: 除能
1: 使能

Bit 1 **USW2**: USW2 开关 on/off 控制
0: Off
1: On

此位用于控制 USW2 开关的 on/off。当 D2+ 或 D2- 其中有一个引脚功能通过引脚共用功能选择寄存器设置为模拟或数字输入类型时，设置此位为高，USW2 开关将为 on。

Bit 0 **USW1**: USW1 开关 on/off 控制
0: Off
1: On

此位用于控制 USW1 开关的 on/off。当 D1+ 或 D1- 其中有一个引脚功能通过引脚共用功能选择寄存器设置为模拟或数字输入类型时，设置此位为高，USW1 开关将为 on。

• ADUDA0 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**: 8-bit DAC0 输出控制位
DAC0 输出 = (DAC0 参考电压) × (ADUDA0[7:0])/256

• ADUDA1 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**: 8-bit DAC1 输出控制位
DAC1 输出 = (DAC1 参考电压) × (ADUDA1[7:0])/256

• ADUDA2 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**: 8-bit DAC2 输出控制位
 $\text{DAC2 输出} = (\text{DAC2 参考电压}) \times (\text{ADUDA2}[7:0]) / 256$

• ADUDA3 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**: 8-bit DAC3 输出控制位
 $\text{DAC3 输出} = (\text{DAC3 参考电压}) \times (\text{ADUDA3}[7:0]) / 256$

低电压检测 – LVD

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

LVD 寄存器

低电压检测功能由 LVDC 寄存器控制。VLVD2~VLVD0 位用于选择 5 个固定的电压参考点。LVDO 位被置位时表明低电压情况发生，若 LVDO 位为低则表明 V_{DD} 电压工作在当前所设置低电压水平值之上。LVDEN 位用于控制低电压检测功能的开启/关闭，设置此位为高使能此功能，反之，关闭内部低电压检测电路。低电压检测会有一些的功耗，在不使用时可考虑关闭此功能，此举在功耗要求严格的电池供电应用中值得考虑。

• LVDC 寄存器

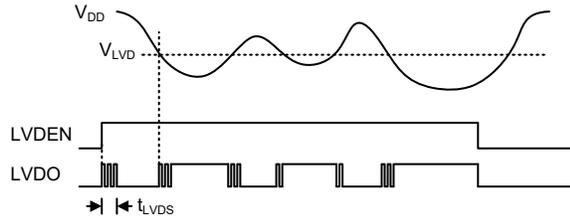
Bit	7	6	5	4	3	2	1	0
Name	—	—	LVDO	LVDEN	—	VLVD2	VLVD1	VLVD0
R/W	—	—	R	R/W	—	R/W	R/W	R/W
POR	—	—	0	0	—	0	0	0

Bit 7~6 未定义，读为“0”
 Bit 5 **LVDO**: LVD 输出标志位
 0: 未检测到低电压
 1: 检测到低电压
 Bit 4 **LVDEN**: 低电压检测功能控制位
 0: 除能
 1: 使能
 Bit 3 未定义，读为“0”
 Bit 2~0 **VLVD2~VLVD0**: LVD 电压选择
 000: 未定义
 001: 未定义
 010: 未定义

011: 2.7V
100: 3.0V
101: 3.3V
110: 3.6V
111: 4.0V

LVD 操作

通过比较电源电压 V_{DD} 与通过 LVDC 寄存器选择的预置电压值的结果，低电压检测功能工作。其设置的范围为 2.7V~4.0V。当电源电压 V_{DD} 低于预置电压值时，LVDO 位被置为高，表明低电压产生。若 LVDEN 位为高，则当单片机在空闲 / 休眠模式时低电压检测器保持有效状态。低电压检测器使能后，读取 LVDO 位前，电路稳定需要一定的延时 t_{LVDs} 。注意， V_{DD} 电压可能上升或下降比较缓慢，在 V_{LVD} 电压值附近时，LVDO 位可能有多种变化。



LVD 操作

低电压检测器也有自己的中断功能，它是除了轮询 LVDO 位之外的另一种检测低电压的方法。中断条件产生置位 LVDO 并延时 t_{LVD} 后，中断产生。此种情况下，若 V_{DD} 降至小于 LVD 预置电压值时，中断请求标志位 LVF 将被置位，中断产生，单片机将从休眠或空闲模式中被唤醒。若不要求低电压检测的唤醒功能使能，在单片机进入休眠或空闲模式前应将 LVF 标志置为高。

当 LVD 功能使能后，建议在使能 LVD 中断前先将 LVD 中断请求标志位清零以避免错误发生。

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 INT0~INT2 引脚动作产生，而内部中断由各种内部功能，如定时器模块 (TM)、时基、过电流保护功能 (OCP)、过电压保护功能 (OVP)、欠电压保护功能 (UVP)、LVD、EEPROM 和 A/D 转换器等产生。

中断寄存器

中断控制基本上是在一定单片机条件发生时设置请求标志位，应用程序中中断使能位的设置是通过位于专用数据存储中的一系列寄存器控制的。寄存器总的分为三类。第一类是 INTC0~INTC3 寄存器，用于设置基本的中断；第二类是 MFIO~MFII 寄存器，用于设置多功能中断；最后一种有 INTEG 寄存器，用于设置外部中断边沿触发类型。

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

功能	使能位	请求标志位	注释
总中断	EMI	—	—
INTn 引脚	INTnE	INTnF	n=0~2
过电流保护	OCnE	OCnF	n=0~1
欠电压保护	UVnE	UVnF	n=0~1
过电压保护	OVPnE	OVPnF	n=0~1
多功能	MFnE	MFnF	n=0~1
LVD	LVE	LVF	—
时基	TBnE	TBnF	n=0~1
A/D 转换器	ADE	ADF	—
EEPROM	DEE	DEF	—
STM	STMPE	STMPF	—
	STMAE	STMAF	
PTM	PTMPE	PTMPF	—
	PTMAE	PTMAF	

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	—	—	INT2S1	INT2S0	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	OVP0F	OC1F	OC0F	OVP0E	OC1E	OC0E	EMI
INTC1	INT0F	UVP1F	UVP0F	OVP1F	INT0E	UVP1E	UVP0E	OVP1E
INTC2	MF1F	MF0F	INT2F	INT1F	MF1E	MF0E	INT2E	INT1E
INTC3	—	TB1F	TB0F	LVF	—	TB1E	TB0E	LVE
MF10	—	DEF	STMAF	STMPF	—	DEE	STMAE	STMPE
MF11	—	ADF	PTMAF	PTMPF	—	ADE	PTMAE	PTMPE

中断寄存器列表

• INTEG 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	INT2S1	INT2S0	INT1S1	INT1S0	INT0S1	INT0S0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 未定义，读为“0”

Bit 5~4 **INT2S1~INT2S0**: INT2 脚中断边沿控制位

- 00: 除能
- 01: 上升沿
- 10: 下降沿
- 11: 双沿

Bit 3~2 **INT1S1~INT1S0**: INT1 脚中断边沿控制位

- 00: 除能
- 01: 上升沿
- 10: 下降沿
- 11: 双沿

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

● **INTC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	OVP0F	OCP1F	OCP0F	OVP0E	OCP1E	OCP0E	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 **OVP0F**: 过电压保护 0 中断请求标志位
 0: 无请求
 1: 中断请求
 Bit 5 **OCP1F**: 过电流保护 1 中断请求标志位
 0: 无请求
 1: 中断请求
 Bit 4 **OCP0F**: 过电流保护 0 中断请求标志位
 0: 无请求
 1: 中断请求
 Bit 3 **OVP0E**: 过电压保护 0 中断控制位
 0: 除能
 1: 使能
 Bit 2 **OCP1E**: 过电流保护 1 中断控制位
 0: 除能
 1: 使能
 Bit 1 **OCP0E**: 过电流保护 0 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	INT0F	UVP1F	UVP0F	OVP1F	INT0E	UVP1E	UVP0E	OVP1E
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 **INT0F**: INT0 中断请求标志位
 0: 无请求
 1: 中断请求
 Bit 6 **UVP1F**: 欠电压保护 1 中断请求标志位
 0: 无请求
 1: 中断请求
 Bit 5 **UVP0F**: 欠电压保护 0 中断请求标志位
 0: 无请求
 1: 中断请求

- Bit 4 **OVPIF**: 过电压保护 1 中断请求标志位
0: 无请求
1: 中断请求
- Bit 3 **INT0E**: INT0 中断控制位
0: 除能
1: 使能
- Bit 2 **UVPIE**: 欠电压保护 1 中断控制位
0: 除能
1: 使能
- Bit 1 **UVPOE**: 欠电压保护 0 中断控制位
0: 除能
1: 使能
- Bit 0 **OVPIE**: 过电压保护 1 中断控制位
0: 除能
1: 使能

● **INTC2 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	MF1F	MF0F	INT2F	INT1F	MF1E	MF0E	INT2E	INT1E
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

- Bit 7 **MF1F**: 多功能中断 1 中断请求标志位
0: 无请求
1: 中断请求
- Bit 6 **MF0F**: 多功能中断 0 中断请求标志位
0: 无请求
1: 中断请求
- Bit 5 **INT2F**: INT2 中断请求标志位
0: 无请求
1: 中断请求
- Bit 4 **INT1F**: INT1 中断请求标志位
0: 无请求
1: 中断请求
- Bit 3 **MF1E**: 多功能中断 1 中断控制位
0: 除能
1: 使能
- Bit 2 **MF0E**: 多功能中断 0 中断控制位
0: 除能
1: 使能
- Bit 1 **INT2E**: INT2 中断控制位
0: 除能
1: 使能
- Bit 0 **INT1E**: INT1 中断控制位
0: 除能
1: 使能

• INTC3 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	TB1F	TB0F	LVF	—	TB1E	TB0E	LVE
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

- Bit 7 未定义，读为“0”
- Bit 6 **TB1F**: 时基 1 中断请求标志位
0: 无请求
1: 中断请求
- Bit 5 **TB0F**: 时基 0 中断请求标志位
0: 无请求
1: 中断请求
- Bit 4 **LVF**: LVD 中断请求标志位
0: 无请求
1: 中断请求
- Bit 3 未定义，读为“0”
- 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 **LVE**: LVD 中断控制位
0: 除能
1: 使能

• MF10 寄存器

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

- Bit 7 未定义，读为“0”
- Bit 6 **DEF**: 数据 EEPROM 写中断请求标志位
0: 无请求
1: 中断请求
- Bit 5 **STMAF**: STM 比较器 A 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 4 **STMPF**: STM 比较器 P 匹配中断请求标志位
0: 无请求
1: 中断请求
- Bit 3 未定义，读为“0”
- Bit 2 **DEE**: 数据 EEPROM 写中断控制位
0: 除能
1: 使能
- Bit 1 **STMAE**: STM 比较器 A 匹配中断控制位
0: 除能
1: 使能

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

● **MF11 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	ADF	PTMAF	PTMPF	—	ADE	PTMAE	PTMPE
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	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 **PTMAF**: PTM 比较器 A 匹配中断请求标志位
0: 无请求
1: 中断请求

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

Bit 3 未定义，读为“0”

Bit 2 **ADE**: A/D 转换器中断控制位
0: 除能
1: 使能

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

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

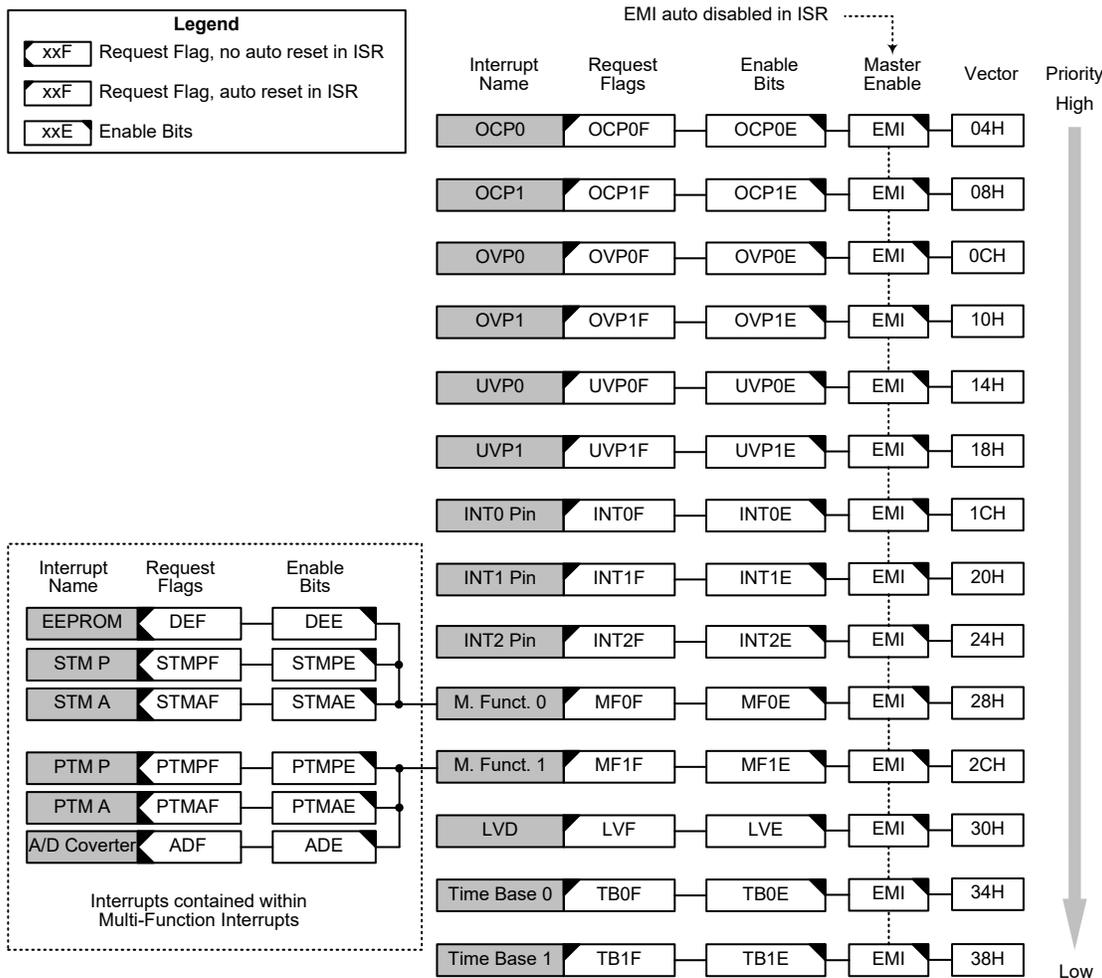
中断操作

若中断事件条件产生，如一个 TM 比较器 P、比较器 A 匹配或 A/D 转换结束等等，相关中断请求标志将置起。中断标志产生后程序是否会跳转至相关中断向量执行是由中断使能位的条件决定的。若使能位为“1”，程序将跳至相关中断向量中执行；若使能位为“0”，即使中断请求标志置起中断也不会发生，程序也不会跳转至相关中断向量执行。若总中断使能位为“0”，所有中断都将除能。当中断发生时，下条指令的地址将被压入堆栈。相应的中断向量地址加载至 PC 中。系统将从此向量取下条指令。中断向量处通常为跳转指令，以跳转到相应的中断服务程序。中断服务程序必须以“RETI”指令返回至主程序，以继续执行原来的程序。

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

如果某个中断服务子程序正在执行时，有另一个中断要求立即响应，那么 EMI 位应在程序进入中断子程序后置位，以允许此中断嵌套。如果堆栈已满，即使此中断使能，中断请求也不会被响应，直到 SP 减少为止。如果要求立刻动作，则堆栈必须避免成为储满状态。请求同时发生时，执行优先级如下流程图所示。

所有被置起的中断请求标志都可把单片机从休眠或空闲模式中唤醒，若要防止唤醒动作发生，在单片机进入休眠或空闲模式前应将相应的标志置起。



中断结构

外部中断

通过 INT0~INT2 引脚上的信号变化可控制外部中断。当触发沿选择位设置好触发类型，INT0~INT2 引脚的状态发生变化，外部中断请求标志 INT0F~INT2F 被置位时外部中断请求产生。若要跳转到相应中断向量地址，总中断控制位 EMI 和相应中断使能位 INT0E~INT2E 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，如果通过相应的引脚共用功能选择位选择外部中断引脚功能且相应寄存器中的中断使能位被置位，此引脚将被作为外部中断脚使用。需注意的是此时该引脚必须通过设置相应的 I/O 端口控制寄存器，将该引脚输入 / 输出类型设置为输入状态。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程序。当响应外部中断服务子程序时，中断请求标志位 INT0F~INT2F 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，其通过寄存器选择的上拉电阻仍保持有效。

寄存器 INTEG 被用来选择有效的边沿类型，来触发外部中断。可以选择上升沿

还是下降沿或双沿触发都产生外部中断。注意 INTEG 也可以用来除能外部中断功能。

过电流保护中断

过电流保护 OCPn 中断通过检测 OCPn 输入电流控制。当检测到过电流情况时，过电流保护中断请求标志位 OCPnF 被置位，OCPn 中断请求发生。当总中断使能位 EMI 和 OCPn 中断使能位 OCPnE 被置位，允许程序跳转到相应的中断向量地址。当中断使能，堆栈未满且过电流情况发生时，将调用 OCPn 中断向量子程序。当响应中断服务子程序时，相应的中断请求标志位 OCPnF 会自动清零。EMI 位也会被清零以除能其它中断。

过电压保护中断

过电压保护 OVPn 中断通过检测 OVPn 输入电压控制。当检测到过电压情况时，过电压保护中断请求标志位 OVPnF 被置位，OVPn 中断请求发生。当总中断使能位 EMI 和过电压保护中断使能位 OVPnE 被置位，允许程序跳转到相应的中断向量地址。当中断使能，堆栈未满且过电压情况发生时，将调用 OVPn 中断向量子程序。当响应中断服务子程序时，相应的中断请求标志位 OVPnF 会自动清零。EMI 位也会被清零以除能其它中断。

欠电压保护中断

欠电压保护 UVPn 中断通过检测 UVPn 输入电压控制。当检测到电欠电压情况发生时，欠电压保护中断请求标志位 UVPnF 被置位，UVPn 中断请求发生。当总中断使能位 EMI 和 UVPn 中断使能控制位 UVPnE 被置位，允许程序跳转到相应的中断向量地址。当中断使能，堆栈未满且相应的欠电压情况发生时，将调用它们中断向量子程序。当响应中断服务子程序时，相应的中断请求标志位 UVPnF 会自动清零。EMI 位也会被清零以除能其它中断。

多功能中断

此单片机中有两个多功能中断，与其它中断不同，它们没有独立的中断来源，而是由其它现有的中断源构成，即 TM 中断、EEPROM 中断和 A/D 转换器中断。当多功能中断中的任何一种中断请求标志位被置位，则相应的多功能中断请求标志位 MFnF 被置位，多功能中断请求产生。当中断使能，堆栈未满，包括在多功能中断中的任意一个中断发生时，将调用多功能中断向量中的一个子程序。当响应中断服务子程序时，相关的多功能请求标志位会自动复位且 EMI 位会自动清零以除能其它中断。

但必须注意的是，在中断响应时，虽然多功能中断请求标志位会自动复位，但多功能中断源的请求标志位，即 TM 中断、EEPROM 中断或 A/D 转换器中断的请求标志位不会自动复位。必须由应用程序清零。

定时器模块中断

标准型和周期型 TM 都有两个中断。所有的 TM 中断也属于多功能中断。标准型和周期型 TM 都有两个中断请求标志位 STMPF、STMAF 和 PTMPF、PTMAF 以及相应的各有两个使能位 STMPE、STMAE 和 PTMPE、PTMAE。当 TM 比较器 P 或 A 匹配情况发生时，任意 TM 中断请求标志被置位，TM 中断请求发生。

若要程序跳转到相应中断向量地址，总中断控制位 EMI、相应 TM 中断使能位和相关多功能中断使能位 MFnE 需先被置位。当中断使能，堆栈未满且 TM 比较器匹配情况发生时，可跳转至相关多功能中断向量子程序中执行。当 TM 中

断响应，EMI 将被自动清零以除能其它中断，相关 MF_nF 标志也可自动清除，但 TM 中断请求标志位需通过应用程序手动清零。

EEPROM 中断

EEPROM 中断也属于多功能中断。当写周期结束，EEPROM 中断请求标志 DEF 被置位，EEPROM 中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI、EEPROM 中断使能位 DEE 和相应多功能中断使能位 MF0E 需先被置位。当中断使能，堆栈未满足且 EEPROM 写周期结束时，可跳转至相关多功能中断向量子程序中执行。当 EEPROM 中断响应，EMI 将被自动清零以除能其它中断，多功能中断请求标志也可自动清除，但 DEF 标志需通过应用程序手动清零。

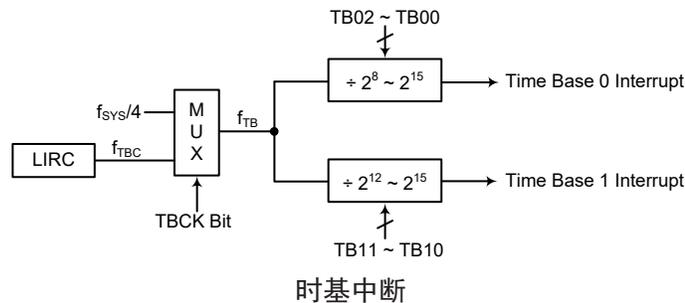
A/D 转换器中断

A/D 转换器中断也属于多功能中断，由 A/D 转换动作的结束来控制。即当 A/D 转换过程完成时，A/D 转换器中断请求标志被置位，中断请求发生。若要跳转到相应的中断向量地址，总中断使能位 EMI、A/D 中断使能位 ADE 和相应多功能中断使能位 MF1E 需先被置位。当中断使能，堆栈未满足且 A/D 转换动作结束时，可跳转至相关多功能中断向量子程序中执行。当响应中断服务子程序后，EMI 将被自动清零以除能其它中断，多功能中断请求标志也可自动清除，但 ADF 标志需通过应用程序手动清零。

时基中断

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

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



• TBC 寄存器

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

Bit 7 **TBON**: 时基 0 和时基 1 功能控制位
0: 除能
1: 使能

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

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

Bit 3 未定义, 读为“0”

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

LVD 中断

当低电压检测功能检测到一个低电压时, LVD 中断请求标志 LVF 被置位, LVD 中断请求产生。若要程序跳转到相应中断向量地址, 总中断控制位 EMI 和低电压中断使能位 LVE 需先被置位。当中断使能, 堆栈未满且低电压条件发生时, 可跳转至 LVD 中断向量程序执行。当低电压中断响应, LVD 中断请求标志可自动清除, EMI 也将被自动清零以除能其它中断。

中断唤醒功能

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

编程注意事项

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

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

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

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

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

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

配置选项

配置选项在烧写程序时写入芯片。通过 HT-IDE 的软件开发环境，使用者在开发过程中可以选择配置选项。由于使用的是硬件工具将配置选项烧入单片机，之后无法再通过应用程序修改。所有的选项必须按系统的需要定义，具体内容可参考下表：

No.	选项
振荡器选项	
1	HIRC 频率选择 – f_{HIRC} : 8MHz、12MHz 或 16MHz

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 μ s 中执行完成，而分支或调用操作则将在 1 μ s 中执行完成。虽然需要两个指令周期的指令通常指的是 JMP、CALL、RET、RETI 和查表指令，但如果牵涉到程序计数器低字节寄存器 PCL 也将多花费一个周期去加以执行。即指令改变 PCL 的内容进而导致直接跳转至新地址时，需要多一个周期去执行，例如“CLR PCL”或“MOV PCL, A”指令。对于跳转指令必须注意的是，如果比较的结果牵涉到跳转动作将多花费一个周期，如果没有则需一个周期即可。

数据的传送

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

算术运算

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

逻辑和移位运算

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

分支和控制转换

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

位运算

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

查表运算

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

其它运算

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

指令集概要

当要操作的数据存储器位于数据存储器 Sector 0 时，下表说明了与数据存储器存取有关的指令。

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

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

助记符	说明	指令周期	影响标志位
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多达 3 个周期，如果没有发生跳转，则只需一个周期。
2. 任何指令若要改变 PCL 的内容将需要 2 个周期来执行。
3. 对于“CLR WDT”指令而言，TO 和 PDF 标志位也许会受执行结果影响，“CLR WDT”被执行后，TO 和 PDF 标志位会被清除，否则 TO 和 PDF 标志位保持不变。

扩展指令集

扩展指令用来提供更大范围的数据存储器寻址。当被存取的数据存储器位于 Sector 0 之外的任何数据存储器 Sector，扩展指令可直接存取数据存储器而无需使用间接寻址，此举不仅可节省 Flash 存储器空间的使用，同时可提高 CPU 执行效率。

助记符	说明	指令周期	影响标志位
算术运算			
LADD A,[m]	ACC 与数据存储器相加，结果放入 ACC	2	Z, C, AC, OV, SC
LADDM A,[m]	ACC 与数据存储器相加，结果放入数据存储器	2 ^注	Z, C, AC, OV, SC
LADC A,[m]	ACC 与数据存储器、进位标志相加，结果放入 ACC	2	Z, C, AC, OV, SC
LADCM A,[m]	ACC 与数据存储器、进位标志相加，结果放入数据存储器	2 ^注	Z, C, AC, OV, SC
LSUB A,[m]	ACC 与数据存储器相减，结果放入 ACC	2	Z, C, AC, OV, SC, CZ
LSUBM A,[m]	ACC 与数据存储器相减，结果放入数据存储器	2 ^注	Z, C, AC, OV, SC, CZ
LSBC A,[m]	ACC 与数据存储器、进位标志的反相减，结果放入 ACC	2	Z, C, AC, OV, SC, CZ
LSBCM A,[m]	ACC 与数据存储器、进位标志相减，结果放入数据存储器	2 ^注	Z, C, AC, OV, SC, CZ
LDA A [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	2 ^注	C
逻辑运算			
LAND A,[m]	ACC 与数据存储器做“与”运算，结果放入 ACC	2	Z
LOR A,[m]	ACC 与数据存储器做“或”运算，结果放入 ACC	2	Z
LXOR A,[m]	ACC 与数据存储器做“异或”运算，结果放入 ACC	2	Z
LANDM A,[m]	ACC 与数据存储器做“与”运算，结果放入数据存储器	2 ^注	Z
LORM A,[m]	ACC 与数据存储器做“或”运算，结果放入数据存储器	2 ^注	Z
LXORM A,[m]	ACC 与数据存储器做“异或”运算，结果放入数据存储器	2 ^注	Z
LCPL [m]	对数据存储器取反，结果放入数据存储器	2 ^注	Z
LCPLA [m]	对数据存储器取反，结果放入 ACC	2	Z
递增和递减			
LINCA [m]	递增数据存储器，结果放入 ACC	2	Z
LINC [m]	递增数据存储器，结果放入数据存储器	2 ^注	Z
LDECA [m]	递减数据存储器，结果放入 ACC	2	Z
LDEC [m]	递减数据存储器，结果放入数据存储器	2 ^注	Z
移位			
LRRA [m]	数据存储器右移一位，结果放入 ACC	2	无
LRR [m]	数据存储器右移一位，结果放入数据存储器	2 ^注	无
LRRCA [m]	带进位将数据存储器右移一位，结果放入 ACC	2	C
LRRC [m]	带进位将数据存储器右移一位，结果放入数据存储器	2 ^注	C
LRLA [m]	数据存储器左移一位，结果放入 ACC	2	无
LRL [m]	数据存储器左移一位，结果放入数据存储器	2 ^注	无
LRLCA [m]	带进位将数据存储器左移一位，结果放入 ACC	2	C
LRLC [m]	带进位将数据存储器左移一位，结果放入数据存储器	2 ^注	C
数据传送			
LMOV A,[m]	将数据存储器送至 ACC	2	无
LMOV [m],A	将 ACC 送至数据存储器	2 ^注	无

助记符	说明	指令周期	影响标志位
位运算			
LCLR [m].i	清除数据存储器的位	2 ^注	无
LSET [m].i	置位数据存储器的位	2 ^注	无
转移			
LSZ [m]	如果数据存储器为零，则跳过下一条指令	2 ^注	无
LSZA [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
LSNZ [m]	如果数据存储器不为零，则跳过下一条指令	2 ^注	无
LSZ [m].i	如果数据存储器的第 i 位为零，则跳过下一条指令	2 ^注	无
LSNZ [m].i	如果数据存储器的第 i 位不为零，则跳过下一条指令	2 ^注	无
LSIZ [m]	递增数据存储器，如果结果为零，则跳过下一条指令	2 ^注	无
LSDZ [m]	递减数据存储器，如果结果为零，则跳过下一条指令	2 ^注	无
LSIZA [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	2 ^注	无
LSDZA [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	2 ^注	无
查表			
LTABRD [m]	读取当前页的 ROM 内容，并送至数据存储器 and TBLH	3 ^注	无
LTABRDL [m]	读取最后页的 ROM 内容，并送至数据存储器 and TBLH	3 ^注	无
LITABRD [m]	读表指针 TBLP 自加，读取当前页的 ROM 内容，并送至数据存储器 and TBLH	3 ^注	无
LITABRDL [m]	读表指针 TBLP 自加，读取最后页的 ROM 内容，并送至数据存储器 and TBLH	3 ^注	无
其它指令			
LCLR [m]	清除数据存储器	2 ^注	无
LSET [m]	置位数据存储器	2 ^注	无
LSWAP [m]	交换数据存储器的高低字节，结果放入数据存储器	2 ^注	无
LSWAPA [m]	交换数据存储器的高低字节，结果放入 ACC	2	无

注：1. 对扩展跳转指令而言，如果比较的结果牵涉到跳转即需多达 4 个周期，如果没有发生跳转，则只需两个周期。
2. 任何扩展指令若要改变 PCL 的内容将需要 3 个周期来执行。

指令定义

ADC A, [m]	Add Data Memory to ACC with Carry
指令说明	将指定的数据存储器、累加器内容以及进位标志相加，结果存放到累加器。
功能表示	$ACC \leftarrow ACC + [m] + C$
影响标志位	OV、Z、AC、C、SC
ADCM A, [m]	Add ACC to Data Memory with Carry
指令说明	将指定的数据存储器、累加器内容和进位标志位相加，结果存放到指定的数据存储器。
功能表示	$[m] \leftarrow ACC + [m] + C$
影响标志位	OV、Z、AC、C、SC
ADD A, [m]	Add Data Memory to ACC
指令说明	将指定的数据存储器和累加器内容相加，结果存放到累加器。
功能表示	$ACC \leftarrow ACC + [m]$
影响标志位	OV、Z、AC、C、SC
ADD A, x	Add immediate data to ACC
指令说明	将累加器和立即数相加，结果存放到累加器。
功能表示	$ACC \leftarrow ACC + x$
影响标志位	OV、Z、AC、C、SC
ADDM A, [m]	Add ACC to Data Memory
指令说明	将指定的数据存储器和累加器内容相加，结果存放到指定的数据存储器。
功能表示	$[m] \leftarrow ACC + [m]$
影响标志位	OV、Z、AC、C、SC
AND A, [m]	Logical AND Data Memory to ACC
指令说明	将累加器中的数据和指定数据存储器内容做逻辑与，结果存放到累加器。
功能表示	$ACC \leftarrow ACC \text{ "AND" } [m]$
影响标志位	Z

AND A, x	Logical AND immediate data to ACC
指令说明	将累加器中的数据和立即数做逻辑与，结果存放到累加器。
功能表示	$ACC \leftarrow ACC \text{ “AND” } x$
影响标志位	Z
ANDM A, [m]	Logical AND ACC to Data Memory
指令说明	将指定数据存储器内容和累加器中的数据做逻辑与，结果存放到数据存储器。
功能表示	$[m] \leftarrow ACC \text{ “AND” } [m]$
影响标志位	Z
CALL addr	Subroutine call
指令说明	无条件地调用指定地址的子程序，此时程序计数器先加 1 获得下一个要执行的指令地址并压入堆栈，接着载入指定地址并从新地址继续执行程序，由于此指令需要额外的运算，所以为一个 2 周期的指令。
功能表示	$Stack \leftarrow Program Counter + 1$ $Program Counter \leftarrow addr$
影响标志位	无
CLR [m]	Clear Data Memory
指令说明	将指定数据存储器的内容清零。
功能表示	$[m] \leftarrow 00H$
影响标志位	无
CLR [m].i	Clear bit of Data Memory
指令说明	将指定数据存储器的 i 位内容清零。
功能表示	$[m].i \leftarrow 0$
影响标志位	无
CLR WDT	Clear Watchdog Timer
指令说明	WDT 计数器、暂停标志位 PDF 和看门狗溢出标志位 TO 清零。
功能表示	WDT cleared $TO \ \& \ PDF \leftarrow 0$
影响标志位	TO、PDF

CPL [m]	Complement Data Memory
指令说明	将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或 0 变 1。
功能表示	$[m] \leftarrow \overline{[m]}$
影响标志位	Z
CPLA [m]	Complement Data Memory with result in ACC
指令说明	将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或 0 变 1，而结果被储存回累加器且数据存储器中的内容不变。
功能表示	$ACC \leftarrow \overline{[m]}$
影响标志位	Z
DAA [m]	Decimal-Adjust ACC for addition with result in Data Memory
指令说明	将累加器中的内容转换为 BCD (二进制转成十进制) 码。如果低四位的值大于“9”或 AC=1，那么 BCD 调整就执行对原值加“6”，否则原值保持不变；如果高四位的值大于“9”或 C=1，那么 BCD 调整就执行对原值加“6”。BCD 转换实质上是根据累加器和标志位执行 00H, 06H, 60H 或 66H 的加法运算，结果存放到数据存储器。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	$TO \leftarrow 0$ $PDF \leftarrow 1$
影响标志位	TO、PDF
INC [m]	Increment Data Memory
指令说明	将指定数据存储器的内容加 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	$PC \leftarrow PC + 1$
影响标志位	无
ORA, [m]	Logical OR Data Memory to ACC
指令说明	将累加器中的数据和指定的数据存储器内容逻辑或，结果存放到累加器。
功能表示	$ACC \leftarrow ACC \text{ "OR" } [m]$
影响标志位	Z
ORA, x	Logical OR immediate data to ACC
指令说明	将累加器中的数据和立即数逻辑或，结果存放到累加器。
功能表示	$ACC \leftarrow ACC \text{ "OR" } x$
影响标志位	Z
ORM A, [m]	Logical OR ACC to Data Memory
指令说明	将存在指定数据存储器中的数据和累加器逻辑或，结果放到数据存储器。
功能表示	$[m] \leftarrow ACC \text{ "OR" } [m]$
影响标志位	Z
RET	Return from subroutine
指令说明	将堆栈寄存器中的程序计数器值恢复，程序由取回的地址继续执行。
功能表示	Program Counter ← Stack
影响标志位	无
RET A, x	Return from subroutine and load immediate data to ACC
指令说明	将堆栈寄存器中的程序计数器值恢复且累加器载入指定的立即数，程序由取回的地址继续执行。
功能表示	Program Counter ← Stack ACC ← x
影响标志位	无

RETI	Return from interrupt
指令说明	将堆栈寄存器中的程序计数器值恢复且中断功能通过设置 EMI 位重新使能。EMI 是控制中断使能的主控制位。如果在执行 RETI 指令之前还有中断未被相应，则这个中断将在返回主程序之前被相应。
功能表示	Program Counter ← Stack EMI ← 1
影响标志位	无
RL [m]	Rotate Data Memory left
指令说明	将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位。
功能表示	[m].(i+1) ← [m].i (i=0~6) [m].0 ← [m].7
影响标志位	无
RLA [m]	Rotate Data Memory left with result in ACC
指令说明	将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位，结果送到累加器，而指定数据存储器的内容保持不变。
功能表示	ACC.(i+1) ← [m].i (i=0~6) ACC.0 ← [m].7
影响标志位	无
RLC [m]	Rotate Data Memory Left through Carry
指令说明	将指定数据存储器的内容连同进位标志左移 1 位，第 7 位取代进位标志且原本的进位标志移到第 0 位。
功能表示	[m].(i+1) ← [m].i (i=0~6) [m].0 ← C C ← [m].7
影响标志位	C
RLC A [m]	Rotate Data Memory left through Carry with result in ACC
指令说明	将指定数据存储器的内容连同进位标志左移 1 位，第 7 位取代进位标志且原本的进位标志移到第 0 位，移位结果送回累加器，但是指定数据寄存器的内容保持不变。
功能表示	ACC.(i+1) ← [m].i (i=0~6) ACC.0 ← C C ← [m].7
影响标志位	C

RR [m]	Rotate Data Memory right
指令说明	将指定数据存储器的内容循环右移 1 位且第 0 位移到第 7 位。
功能表示	$[m].i \leftarrow [m].(i+1)$ (i=0~6) $[m].7 \leftarrow [m].0$
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=0~6) $ACC.7 \leftarrow [m].0$
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=0~6) $[m].7 \leftarrow C$ $C \leftarrow [m].0$
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=0~6) $ACC.7 \leftarrow C$ $C \leftarrow [m].0$
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、SC、CZ

SBC A, x	Subtract immediate data from ACC with Carry
指令说明	将累加器减去立即数以及进位标志，结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示	$ACC \leftarrow ACC - [m] - \bar{C}$
影响标志位	OV、Z、AC、C、SC、CZ
SBCM A, [m]	Subtract Data Memory from ACC with Carry and result in Data Memory
指令说明	将累加器减去指定数据存储器的内容以及进位标志的反，结果存放到数据存储器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示	$[m] \leftarrow ACC - [m] - \bar{C}$
影响标志位	OV、Z、AC、C、SC、CZ
SDZ [m]	Skip if Decrement Data Memory is 0
指令说明	将指定的数据存储器的内容减 1，判断是否为 0，若为 0 则跳过下一条指令，由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	$[m] \leftarrow [m] - 1$ ，如果 $[m]=0$ 跳过下一条指令执行
影响标志位	无
SDZA [m]	Decrement data memory and place result in ACC, skip if 0
指令说明	将指定数据存储器内容减 1，判断是否为 0，如果为 0 则跳过下一条指令，此结果将存放到累加器，但指定数据存储器内容不变。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	$ACC \leftarrow [m] - 1$ ，如果 $ACC=0$ 跳过下一条指令执行
影响标志位	无
SET [m]	Set Data Memory
指令说明	将指定数据存储器的每一位设置为 1。
功能表示	$[m] \leftarrow FFH$
影响标志位	无

<p>SET [m].i</p> <p>指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Set bit of Data Memory</p> <p>将指定数据存储器的第 i 位置位为 1。</p> <p>$[m].i \leftarrow 1$</p> <p>无</p>
<p>SIZ [m]</p> <p>指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Skip if increment Data Memory is 0</p> <p>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。</p> <p>$[m] \leftarrow [m] + 1$，如果 $[m]=0$ 跳过下一条指令执行</p> <p>无</p>
<p>SIZA [m]</p> <p>指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Skip if increment Data Memory is zero with result in ACC</p> <p>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。</p> <p>$ACC \leftarrow [m] + 1$，如果 $ACC=0$ 跳过下一条指令执行</p> <p>无</p>
<p>SNZ [m].i</p> <p>指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Skip if bit i of Data Memory is not 0</p> <p>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。</p> <p>如果 $[m].i \neq 0$，跳过下一条指令执行</p> <p>无</p>
<p>SNZ [m]</p> <p>指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Skip if Data Memory is not 0</p> <p>判断指定存储器，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。</p> <p>如果 $[m] \neq 0$，跳过下一条指令执行</p> <p>无</p>

SUB A, [m]	Subtract Data Memory from ACC
指令说明	将累加器的内容减去指定的数据存储器的数据，把结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示	$ACC \leftarrow ACC - [m]$
影响标志位	OV、Z、AC、C、SC、CZ
SUBM A, [m]	Subtract Data Memory from ACC with result in Data Memory
指令说明	将累加器的内容减去指定数据存储器的数据，结果存放到指定的数据存储器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示	$[m] \leftarrow ACC - [m]$
影响标志位	OV、Z、AC、C、SC、CZ
SUB A, x	Subtract immediate Data from ACC
指令说明	将累加器的内容减去立即数，结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示	$ACC \leftarrow ACC - x$
影响标志位	OV、Z、AC、C、SC、CZ
SWAP [m]	Swap nibbles of Data Memory
指令说明	将指定数据存储器的低 4 位和高 4 位互相交换。
功能表示	$[m].3 \sim [m].0 \leftrightarrow [m].7 \sim [m].4$
影响标志位	无
SWAPA [m]	Swap nibbles of Data Memory with result in ACC
指令说明	将指定数据存储器的低 4 位与高 4 位互相交换，再将结果存放到累加器且指定数据寄存器的数据保持不变。
功能表示	$ACC.3 \sim ACC.0 \leftarrow [m].7 \sim [m].4$ $ACC.7 \sim ACC.4 \leftarrow [m].3 \sim [m].0$
影响标志位	无
SZ [m]	Skip if Data Memory is 0
指令说明	判断指定数据存储器的内容是否为 0，若为 0，则程序跳过下一条指令执行。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

<p>SZA [m] 指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Skip if Data Memory is 0 with data movement to ACC 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。</p> <p>ACC ←[m]，如果 [m]=0，跳过下一条指令执行</p> <p>无</p>
<p>SZ [m].i 指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Skip if bit i of Data Memory is 0 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。</p> <p>如果 [m].i=0，跳过下一条指令执行</p> <p>无</p>
<p>TABRD [m] 指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Read table (specific page) to TBLH and Data Memory 将表格指针对 TBHP 和 TBLP 所指的程序代码低字节 (指定页) 移至指定数据存储器且将高字节移至 TBLH。</p> <p>[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)</p> <p>无</p>
<p>TABRDL [m] 指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Read table (last page) to TBLH and Data Memory 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。</p> <p>[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)</p> <p>无</p>
<p>ITABRD [m] 指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Increment table pointer low byte first and read table to TBLH and data memory 将自加表格指针 TBLP 所指的程序代码低字节 (当前页) 移至指定的数据存储器且将高字节移至 TBLH。</p> <p>[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)</p> <p>无</p>

ITABRDL [m]	Increment table pointer low byte first and read table (last page) to TBLH and data memory
指令说明	将自加表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定的数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
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 to 累加器。
功能表示	ACC ← ACC “XOR” [m]
影响标志位	Z
XORM A, [m]	Logical XOR ACC to Data Memory
指令说明	将累加器的数据和指定的数据存储器内容逻辑异或, 结果放到数据存储器。
功能表示	[m] ← ACC “XOR” [m]
影响标志位	Z
XOR A, x	Logical XOR immediate data to ACC
指令说明	将累加器的数据与立即数逻辑异或, 结果存放 to 累加器。
功能表示	ACC ← ACC “XOR” x
影响标志位	Z

扩展指令定义

扩展指令被用来直接存取存储在任何数据存储器 Sector 中的数据。

LADC A, [m]	Add Data Memory to ACC with Carry
指令说明	将指定的数据存储器、累加器内容以及进位标志相加，结果存放到累加器。
功能表示	$ACC \leftarrow ACC + [m] + C$
影响标志位	OV、Z、AC、C、SC
LADCM A, [m]	Add ACC to Data Memory with Carry
指令说明	将指定的数据存储器、累加器内容和进位标志位相加，结果存放到指定的数据存储器。
功能表示	$[m] \leftarrow ACC + [m] + C$
影响标志位	OV、Z、AC、C、SC
LADD A, [m]	Add Data Memory to ACC
指令说明	将指定的数据存储器和累加器内容相加，结果存放到累加器。
功能表示	$ACC \leftarrow ACC + [m]$
影响标志位	OV、Z、AC、C、SC
LADDM A, [m]	Add ACC to Data Memory
指令说明	将指定的数据存储器和累加器内容相加，结果存放到指定的数据存储器。
功能表示	$[m] \leftarrow ACC + [m]$
影响标志位	OV、Z、AC、C、SC
LAND A, [m]	Logical AND Data Memory to ACC
指令说明	将累加器中的数据和指定数据存储器内容做逻辑与，结果存放到累加器。
功能表示	$ACC \leftarrow ACC \text{ "AND" } [m]$
影响标志位	Z
LANDM A, [m]	Logical AND ACC to Data Memory
指令说明	将指定数据存储器内容和累加器中的数据做逻辑与，结果存放到数据存储器。
功能表示	$[m] \leftarrow ACC \text{ "AND" } [m]$
影响标志位	Z

LCLR [m]	Clear Data Memory
指令说明	将指定数据存储器的内容清零。
功能表示	$[m] \leftarrow 00H$
影响标志位	无
LCLR [m].i	Clear bit of Data Memory
指令说明	将指定数据存储器的 i 位内容清零。
功能表示	$[m].i \leftarrow 0$
影响标志位	无
LCPL [m]	Complement Data Memory
指令说明	将指定数据存储器中的每一位取逻辑反， 相当于从 1 变 0 或 0 变 1。
功能表示	$[m] \leftarrow \overline{[m]}$
影响标志位	Z
LCPLA [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，结果被存放回累加器且数据寄存器的内容保持 不变。
功能表示	$ACC \leftarrow \overline{[m]}$
影响标志位	Z
LDAA [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 对低四位加“6”，否则低四位保持不变；如果高四位的 值大于“9”或 C=1，那么 BCD 调整就执行对高四位加“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

LDEC [m]	Decrement Data Memory
指令说明	将指定数据存储器的内容减 1。
功能表示	$[m] \leftarrow [m] - 1$
影响标志位	Z
LDECA [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
LINC [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
LINCA [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
LMOV 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中。
功能表示	$ACC \leftarrow [m]$
影响标志位	无
LMOV [m], A	Move ACC to Data Memory
指令说明	将累加器的内容复制到指定数据存储器。
功能表示	$[m] \leftarrow ACC$
影响标志位	无
LOR 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

LORMA, [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 and 累加器逻辑或，结果放到数据存储器。
功能表示	$[m] \leftarrow \text{ACC} \text{ "OR" } [m]$
影响标志位	Z
LRL [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
LRLA [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	$\text{ACC}.(i+1) \leftarrow [m].i \ (i=0\sim6)$ $\text{ACC}.0 \leftarrow [m].7$
影响标志位	无
LRLC [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
LRLCA [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	$\text{ACC}.(i+1) \leftarrow [m].i \ (i=0\sim6)$ $\text{ACC}.0 \leftarrow C$ $C \leftarrow [m].7$
影响标志位	C

LRR [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=0~6) $[m].7 \leftarrow [m].0$
影响标志位	无
LRRA [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=0~6) $ACC.7 \leftarrow [m].0$
影响标志位	无
LRRC [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=0~6) $[m].7 \leftarrow C$ $C \leftarrow [m].0$
影响标志位	C
LRRC A [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=0~6) $ACC.7 \leftarrow C$ $C \leftarrow [m].0$
影响标志位	C
LSBC A, [m]	Subtract Data Memory from ACC with Carry
指令说明	将累加器减去指定数据存储器的内容以及进位标志的反，结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示	$ACC \leftarrow ACC - [m] - \bar{C}$
影响标志位	OV、Z、AC、C、SC、CZ

LSBCMA, [m]	Subtract Data Memory from ACC with Carry and result in Data Memory
指令说明	将累加器减去指定数据存储器的内容以及进位标志的反，结果存放到数据存储器。如果结果为负，C标志位清除为0，反之结果为正或0，C标志位设置为1。
功能表示	$[m] \leftarrow ACC - [m] - \bar{C}$
影响标志位	OV、Z、AC、C、SC、CZ
LSDZ [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
LSDZA [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
LSET [m]	Set Data Memory
指令说明	将指定数据存储器的每一个位置位为1。
功能表示	$[m] \leftarrow FFH$
影响标志位	无
LSET [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

<p>LSIZ [m] 指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Skip if increment Data Memory is 0</p> <p>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。</p> <p>$[m] \leftarrow [m] + 1$，如果 $[m]=0$ 跳过下一条指令执行</p> <p>无</p>
<p>LSIZA [m] 指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Skip if increment Data Memory is zero with result in ACC</p> <p>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。</p> <p>$ACC \leftarrow [m] + 1$，如果 $ACC=0$ 跳过下一条指令执行</p> <p>无</p>
<p>LSNZ [m].i 指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Skip if bit i of Data Memory is not 0</p> <p>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。</p> <p>如果 $[m].i \neq 0$，跳过下一条指令执行</p> <p>无</p>
<p>LSNZ [m] 指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Skip if Data Memory is not 0</p> <p>判断指定数据存储器，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。</p> <p>如果 $[m] \neq 0$，跳过下一条指令执行</p> <p>无</p>
<p>LSUB A, [m] 指令说明</p> <p>功能表示</p> <p>影响标志位</p>	<p>Subtract Data Memory from ACC</p> <p>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。</p> <p>$ACC \leftarrow ACC - [m]$</p> <p>OV、Z、AC、C、SC、CZ</p>

LSUBM A, [m] 指令说明	Subtract Data Memory from ACC with result in Data Memory 将累加器的内容减去指定数据存储器中的数据，结果存放到指定的数据存储器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示	$[m] \leftarrow ACC - [m]$
影响标志位	OV、Z、AC、C、SC、CZ
LSWAP [m] 指令说明	Swap nibbles of Data Memory 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
LSWAPA [m] 指令说明	Swap nibbles of Data Memory with result in ACC 将指定数据存储器的低 4 位和高 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
LSZ [m] 指令说明	Skip if Data Memory is 0 判断指定数据存储器内容是否为 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
LSZA [m] 指令说明	Skip if Data Memory is 0 with data movement to ACC 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

LSZ [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
LTABRD [m]	Move the ROM code to TBLH and data memory
指令说明	将表格指针 TBLP 所指的程序代码低字节 (当前页) 移至指定数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
LTABRDL [m]	Read table (last page) to TBLH and Data Memory
指令说明	将表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
LITABRD [m]	Increment table pointer low byte first and read table to TBLH and data memory
指令说明	将自加表格指针 TBHP 和 TBLP 所指的程序代码低字节移至指定的数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
LITABRDL [m]	Increment table pointer low byte first and read table (last page) to TBLH and data memory
指令说明	将自加表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定的数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无

LXOR 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
LXORM 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

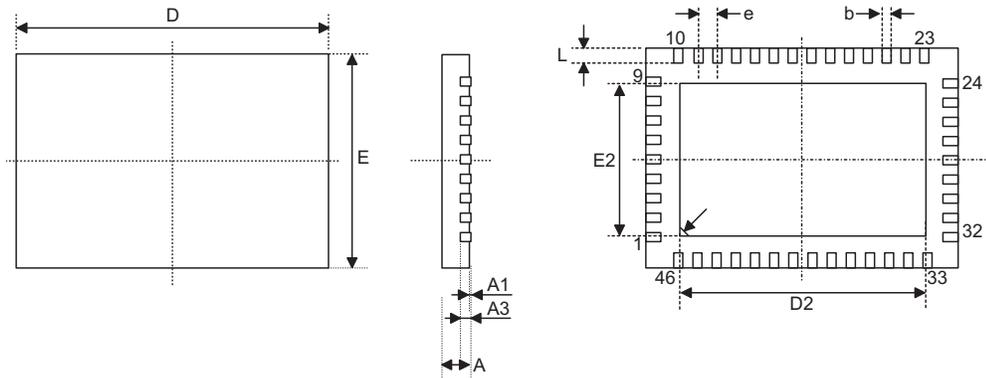
封装信息

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

SAW Type 46-pin QFN (6.5mm×4.5mm) 外形尺寸



符号	尺寸 (单位: inch)		
	最小值	典型值	最大值
A	0.031	0.033	0.035
A1	0.000	0.001	0.002
A3	—	0.008 BSC	—
b	0.006	0.008	0.010
D	0.254	0.256	0.258
E	0.175	0.177	0.179
e	—	0.016 BSC	—
D2	0.197	0.201	0.205
E2	0.118	0.122	0.126
L	0.012	0.016	0.020

符号	尺寸 (单位: mm)		
	最小值	典型值	最大值
A	0.80	0.85	0.90
A1	0.00	0.02	0.04
A3	—	0.20 BSC	—
b	0.15	0.20	0.25
D	6.45	6.50	6.55
E	4.45	4.50	4.55
e	—	0.40 BSC	—
D2	5.00	5.10	5.20
E2	3.00	3.10	3.20
L	0.30	0.40	0.50

Copyright© 2019 by HOLTEK SEMICONDUCTOR INC.

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