无刷直流马达控制 A/D 型 Flash 单片机

HT66FM5240

版本：V1.30　日期：2023-07-25

www.holtek.com

目录

特性

CPU 特性

- 工作电压：
  f_{SYS}=32kHz~20MHz：4.5V~5.5V
- V_{DD}=5V，系统时钟为 20MHz 时，指令周期为 0.2μs
- 提供暂停和唤醒功能，以降低功耗
- 两种振荡模式：
  内部 20MHz RC – HIRC
  内部 32kHz RC – LIRC
- 多种工作模式：正常，低速，空闲，休眠
- 所有指令都可在 1 或 2 个指令周期内完成
- 查表指令
- 61 条指令
- 多达 8 层堆栈
- 位操作指令

周边特性

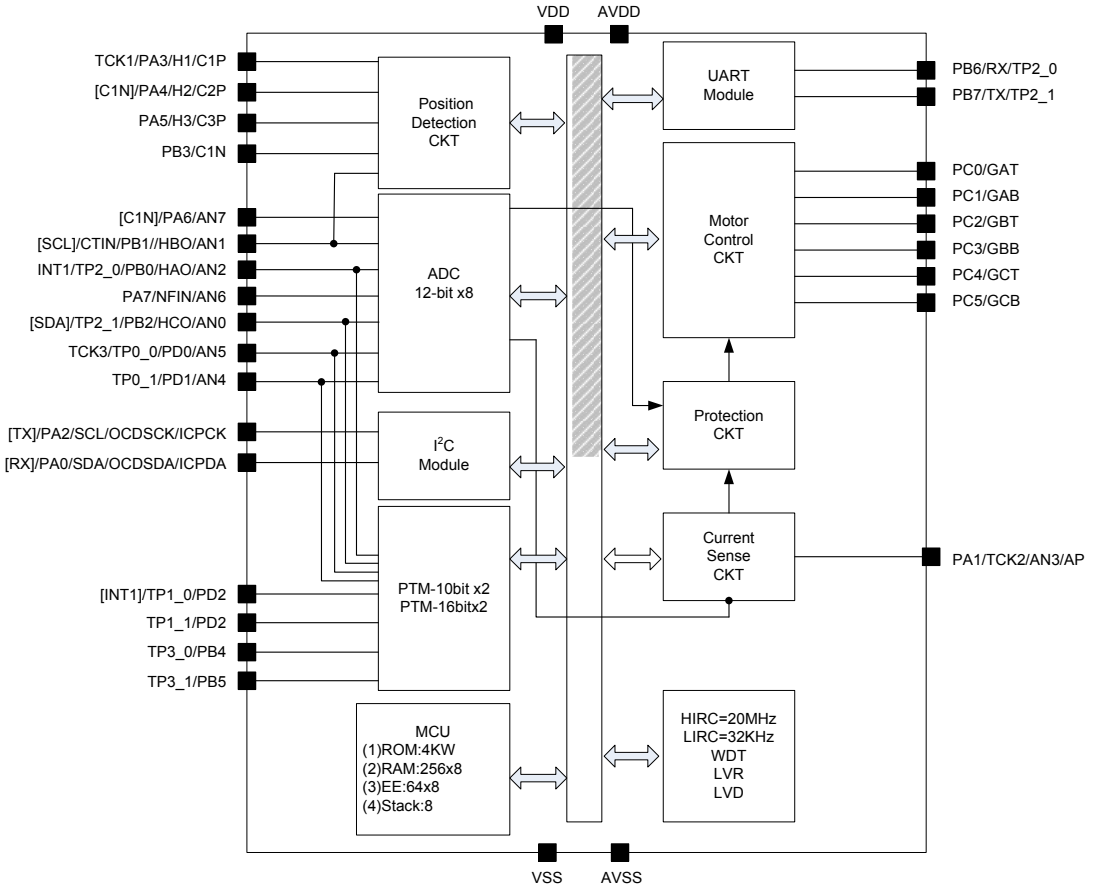
- Flash 程序存储：4K×16
- RAM 数据存储：256×8
- True EEPROM 存储器：64×8
- 看门狗定时器功能
- 多达 26 个双向 I/O 口
- 5 个引脚与外部中断口共用
- 2 个 16 位的 PTM
- 2 个 10 位的 PTM
- 1 个 16 位的 CAPTM 用于马达保护
- PWM 10-bit×3 (支持边沿 / 中心对齐模式)
- 8+1 通道 12-bit 分辨精度的 A/D 转换器
- 时基功能，可提供固定时间的中断信号
- 集成一个运算放大器，用于电流检测
- 集成 4 个比较器
- 1 个 8 位的 D/A 转换器
- I^2C 接口
- UART 接口
- 低电压复位功能
- 低电压检测功能
- 封装类型：20/28-pin SSOP

概述

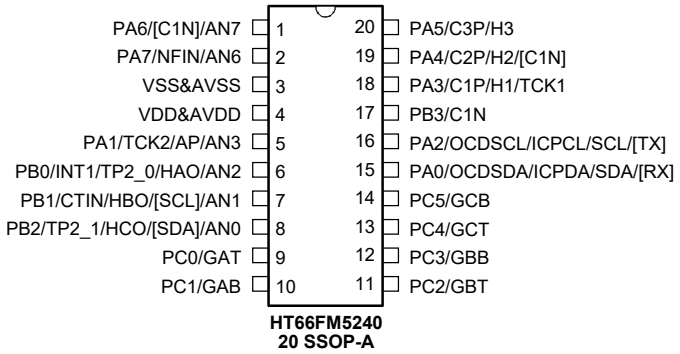
HT66FM5240 是一款 8 位高性能精简指令集 Flash 单片机，它包含一个高度集成有多种特性的主机，是专门为无刷直流电机应用而设计的。

此款单片机具有性能良好、功耗低、I/O 使用灵活的特点，外加集成有定时器模块、振荡类型可选、多通道 A/D、D/A 转换器、PWM、16 位捕捉定时器模块、比较器、马达保护模块、霍尔传感器位置检测功能、时基、LVD、True EEPROM、暂停和唤醒功能。与外界通信的全集成 I^2C 和 UART 接口功能，虽是一款专为无刷直流马达应用而设计的单片机，但其强大的功能使得它可以广泛应用于 A/D 产品中，例如传感器信号处理、马达驱动、工业控制、消费类产品和子系统控制等。

方框图



引脚图



注：1. VDD&AVDD 指的是 VDD 和 AVDD 为同一个引脚。
2. VSS&AVSS 指的是 VSS 和 AVSS 为同一个引脚。
3. 在较小封装中可能含有未引出的引脚，需合理设置其状态以避免输入浮空造成额外耗电，详见“待机电流注意事项”和“输入 / 输出端口”章节。

引脚说明

引脚名称	功能	OP	I/T	O/T	描述
PA0/OCDSDA/ ICPDA/SDA/[RX]	PA0	PAPU PAWU PAPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻和唤醒功能
	OCDSDA	—	ST	CMOS	片上调试数据 / 地址引脚
	ICPDA	—	ST	CMOS	在线烧录数据 / 地址引脚
	SDA	PAPS0 PRM	ST	NMOS	I^2C 数据
	RX	PAPS0 PRM	ST	—	UART 串行数据输入脚

引脚名称	功能	OP	I/T	O/T	描述
PA1/TCK2/AP/AN3	PA1	PAPU PAWU PAPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻和唤醒功能
	TCK2	PAPS0	ST	—	TM2 输入
	AP	PAPS0	ST	—	运算放大器输入
	AN3	PAPS0	AN	—	A/D 通道 3
PA2/OCDSCK/ICPCK/SCL/[TX]	PA2	PAPU PAWU PAPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻和唤醒功能
	OCDSCK	—	ST	—	片上调试时钟引脚
	ICPCK	—	ST	—	在线烧录时钟引脚
	SCL	PAPS0 PRM	ST	NMOS	I^2C 时钟
	TX	PAPS0 PRM	—	CMOS	UART 串行数据输出脚
PA3/C1P/H1/TCK1	PA3	PAPU PAWU PAPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻和唤醒功能
	C1P	PAPS0	AN	—	比较器 1 输入
	H1	PAPS0	ST	—	霍尔传感器输入
	TCK1	PAPS0	ST	—	TM1 输入
PA4/C2P/H2/[C1N]	PA4	PAPU PAWU PAPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻和唤醒功能
	C2P	PAPS1	AN	—	比较器 2 输入
	H2	PAPS1	ST	—	霍尔传感器输入
	C1N	PAPS1 PRM	AN	—	比较器 1 输入
PA5/C3P/H3	PA5	PAPU PAWU PAPS1	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻和唤醒功能
	H3	PAPS1	ST	—	霍尔传感器输入
	C3P	PAPS1	AN	—	比较器 3 输入
PA6/[C1N]/AN7	PA6	PAPU PAWU PAPS1	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻和唤醒功能
	C1N	PAPS1 PRM	AN	—	比较器 1 输入
	AN7	PAPS1	AN	—	A/D 通道 7
PA7/NFIN/AN6	PA7	PAPU PAWU PAPS1	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻和唤醒功能
	NFIN	PAPS1	ST	—	外部 Noise Filtered 输入
	AN6	PAPS1	AN	—	A/D 通道 6

引脚名称	功能	OP	I/T	O/T	描述
PB0/INT1/TP2_0/HAO/AN2	PB0	PBPU	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	INT1	PBPS0 PRM	ST	—	外部中断 1 输入
	TP2_0	PBPS0	ST	CMOS	TM2 输出脚
	HAO	PBPS0	—	CMOS	SA 测试引脚
	AN2	PBPS0	AN	—	A/D 通道 2
PB1/CTIN/HBO/[SCL]/AN1	PB1	PBPU PBPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	CTIN	PBPS0	—	—	CAPTM 输入
	HBO	PBPS0	—	CMOS	SB 测试引脚
	SCL	PBPS0 PRM	ST	NMOS	I²C 时钟
	AN1	PBPS0	AN	—	A/D 通道 1
PB2/TP2_1/HCO/[SDA]/AN0	PB2	PBPU PBPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	TP2_1	PBPS0	ST	CMOS	TM2 输出
	HCO	PBPS0	—	CMOS	SC 测试引脚
	SDA	PBPS0 PRM	ST	NMOS	I²C 数据
	AN0	PBPS0	AN	—	A/D 通道 0
PB3/C1N	PB3	PBPU PBPS1	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	C1N	PBPS1 PRM	AN	—	比较器 1 输入
PB4/TP3_0	PB4	PBPU PBPS1	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	TP3_0	PBPS1	ST	CMOS	TM3 输出
PB5/TP3_1	PB5	PBPU PBPS1	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	TP3_1	PBPS1	ST	CMOS	TM3 输出
PB6/TP2_0/RX	PB6	PBPU PBPS1	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	TP2_0	PBPS1	ST	CMOS	TM2 输出
	RX	PBPS1 PRM	ST	—	UART 串行数据输入脚
PB7/TP2_1/TX	PB7	PBPU PBPS2	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	TP2_1	PBPS2	ST	CMOS	TM2 输出
	TX	PBPS2 PRM	—	CMOS	UART 串行数据输出脚
PC0/GAT	PC0	PCPU PCPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	GAT	PCPS0	—	CMOS	PWM 互补输出

引脚名称	功能	OP	I/T	O/T	描述
PC1/GAB	PC1	PCPU PCPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	GAB	PCPS0	—	CMOS	PWM 互补输出
PC2/GBT	PC2	PCPU PCPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	GBT	PCPS0	—	CMOS	PWM 互补输出
PC3/GBB	PC3	PCPU PCPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	GBB	PCPS0	—	CMOS	PWM 互补输出
PC4/GCT	PC4	PCPU PCPS1	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	GCT	PCPS1	—	CMOS	PWM 互补输出
PC5/GCB	PC5	PCPU	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	GCB	PCPS1	—	CMOS	PWM 互补输出
PD0/TCK0/TCK3/ TP0_0/AN5	PD0	PDPU PDPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	TCK0	PDPS0	ST	—	TM0 输入
	TCK3	PDPS0	ST	—	TM3 输入
	TP0_0	PDPS0	ST	CMOS	TM0 输出
	AN5	PDPS0	AN	—	A/D 通道 5
PD1/TP0_1/AN4	PD1	PDPU PDPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	TP0_1	PDPS0	ST	CMOS	TM0 输出
	AN4	PDPS0	AN	—	A/D 通道 4
PD2/TP1_0/[INT1]	PD2	PDPU PDPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	TP1_0	PDPS0	ST	CMOS	TM1 输出
	INT1	PDPS0 PRM	ST	—	外部中断 1 输入
PD3/TP1_1	PD3	PDPU PDPS0	ST	CMOS	8 位双向输入 / 输出口 通过寄存器使能上拉电阻
	TP1_1	PDPS0	ST	CMOS	TM1 输出
VSS	VSS	—	PWR	—	负电源
AVSS	AVSS	—	PWR	—	A/D 转换器的接地引脚，封装形式的芯片上的 VSS 和 AVSS 相同
VDD	VDD	—	PWR	—	正电源
AVDD	AVDD	—	PWR	—	A/D 转换器的电源供应端，封装形式的芯片上的 VDD 和 AVDD 相同

注：I/T：输入类型；　　O/T：输出类型
OP：通过配置选项 (CO) 或寄存器选项来设置
PWR：电源；　　CO：配置选项；　ST：斯密特触发输入
CMOS：CMOS 输出；NMOS：NMOS 输出
AN：模拟输入脚

VDD 是单片机电源电压，而 AVDD 是 ADC 电源电压。
VSS 是单片机地引脚，而 AVSS 是 ADC 地引脚。
此引脚功能表是针对大封装芯片而言的，对于小封装的芯片可能不具有上述引脚和功能。

极限参数

电源供应电压 ..... V_{SS}-0.3V~V_{SS}+6.0V
端口输入电压 ..... V_{SS}-0.3V~V_{DD}+0.3V
储存温度 ..... -60°C~150°C
工作温度 ..... -40°C~85°C
I_{OH} 总电流 ..... -80mA
I_{OL} 总电流 80mA
总功耗 500mW

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

直流电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DD}	工作电压	—	f_{SYS}=32~20000kHz	4.5	—	5.5	V
I_{DD}	工作电流 (HIRC)	5V	无负载，f_H= 20MHz，ADC 关闭，WDT 使能，Motor_CTL 关闭，	—	9	12	mA
I_{STB}	静态电流	—	LIRC 和 LVR 开启，LVD 关闭，WDT 使能	—	60	100	μA
V_{IL}	输入 / 输出口、TCKn，INTn，H1，H2，H3 和 NFIN 的低电平输入电压	—	—	0	—	0.3V_{DD}	V
V_{IH}	输入 / 输出口、TCKn，INTn，H1，H2，H3 和 NFIN 的高电平输入电压	—	—	0.7V_{DD}	—	V_{DD}	V
V_{LVR}	低电压复位电压	—	LVR 使能：3.15V	-5%	3.15	+5%	V
V_{LVD}	低电压检测电压	—	LVDEN=1，V_{LVD}=3.6V	-5%	3.6	+5%	V
V_{OL}	输入 / 输出口的低电平输出电压	5V	I_{OL}=20mA	—	—	0.5	V
V_{OH}	输入 / 输出口的高电平输出电压	5V	I_{OH}=-7.4mA	4.5	—	—	V
R_{PH}	上拉电阻	5V	—	10	30	50	kΩ

交流电气特性

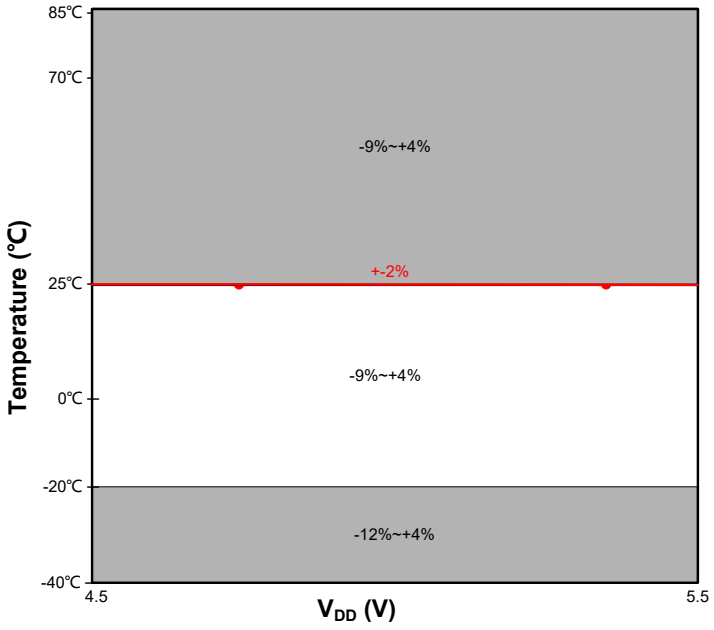
Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
f_{SYS}	系统时钟	—	4.5V~5.5V	32	—	20000	kHz
f_{HIRC}	系统时钟 (HIRC)	4.5V~5.5V	Ta=-40°C~85°C	-12%	20	+4%	MHz
			Ta=-20°C~85°C	-9%	20	+4%	MHz
			Ta=25°C	-2%	20	+2%	MHz
f_{TIMER}	定时器输入频率	—	—	—	—	4	f_{SYS}
t_{INT}	外部中断输入最小脉宽	—	—	1	5	10	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 中断的低电压最短保持时间	—	—	60	120	240	μs
t_{LVDS}	LVDO 稳定时间	—	—	—	—	15	μs
t_{EERD}	EEPROM 读时间	—	—	—	2	4	t_{SYS}
t_{EEWR}	EEPROM 写时间	—	—	—	2	4	ms
t_{SST}	系统启动时间 (从暂停模式中唤醒)	—	f_{SYS}=HIRC	—	15~16	—	t_{SYS}
t_{RSTD}	系统复位延迟时间 (上电复位)	—	—	25	50	100	ms
	系统复位延迟时间 (上电复位之外的复位)	—	—	8.3	16.7	33.3	ms

注：1. t_{SYS} =1/f_{SYS}

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

HIRC 频率准确性随此单片机的 V_{DD} 和温度变化



ADC 特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
AV_{DD}	ADC 工作电压	—	—	V_{LVR}	5.0	5.5	V
I_{OP}	ADC 工作电流	3V	—	—	0.5	—	mA
		5V	—	—	0.6	—	mA
I_{STBY}	ADC 静态电流	—	数字输入没改变	—	—	4	μA
V_{REF}	A/D 输入参考电压	—	—	2.0	AV_{DD}	AV_{DD}+0.1	V
t_{CONV}	转换时间	—	—	16			t_{ADCK}
DNL	A/D 非线性微分误差	4.5V 5.5V	V_{REF}=AV_{DD}=V_{DD}，无负载， t_{ADCK}=0.2μs，10-bit	-1	—	3	LSB
		4.5V 5.5V	V_{REF}=AV_{DD}=V_{DD}，无负载 t_{ADCK}=0.8μs，12-bit				
		4.5V 5.5V	V_{REF}=AV_{DD}=V_{DD}，无负载， t_{ADCK}=0.8μs，10-bit/12-bit				LSB
INL	A/D 非线性积分误差	4.5V 5.5V	V_{REF}=AV_{DD}=V_{DD}，无负载， t_{ADCK}=0.2μs，10-bit	-4	—	4	LSB
		4.5V 5.5V	V_{REF}=AV_{DD}=V_{DD}，无负载 t_{ADCK}=0.8μs，12-bit				
		4.5V 5.5V	V_{REF}=AV_{DD}=V_{DD}，无负载， t_{ADCK}=0.8μs，10-bit/12-bit				
G_{err}	增益出错	—	—	-10	—	+10	LSB
t_{ADCK}	ADCLK 周期	—	12-bit	0.8	—	12.8	μs
		—	10-bit	0.2	—	12.8	μs
t_{CKH}	ADCLK 高宽度	—	—	225	—	—	ns
t_{CKL}	ADCLK 低宽度	—	—	225	—	—	ns
t_{ST1}	ADON 设置时间	—	—	2	—	—	ns
t_{ST2}	START 锁定设置时间	—	—	2	—	—	ns
t_{STH}	START 高宽度	—	—	25	—	—	ns
t_{DEOC}	EOCB 输出延时	—	AV_{DD}=5V	—	3	—	ns
t_{DOUT}	输出延时	—	AV_{DD}=5V	—	3	—	ns
t_{ON}	A/D 唤醒时间	—	—	—	—	2	μs
t_{OFF}	A/D 睡眠时间	—	—	—	—	5	ns

DAC 特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DD}	D/A 工作电压	—	—	V_{LVR}	—	5.5	V
V_{DA}	D/A 输出电压	—	00h~FFh，无负载	0.01	—	0.99	V
t_{DAC}	D/A 转换时间	—	V_{DD}=5V，C_L=10pF	—	—	2	μs
R_O	D/A 输出电阻	—	—	—	10	—	kΩ

运算放大器特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{OPR1}	工作电压	3.3V	—	2.7	3.3	5.5	V
I_{OFF1}	暂停模式时的工作电流	3.3V	—	—	—	0.1	μA
V_{OPOS1}	输入偏置电压 1	3.3V	未校准，AOF[4:0]=10000B	-15	—	+15	mV
V_{OPOS2}	输入偏置电压 2	3.3V	通过校准	-2	—	+2	mV
V_{CM}	共模电压范围	3.3V	—	V_{SS}	—	V_{DD}-1.4V	V
PSRR	电源电压抑制比	3.3V	—	90	—	96	dB
CMRR	共模抑制比	3.3V	V_{CM} =0~V_{DD}-1.4V	—	106	—	dB
SR	转换速率 +，转换速率 -	3.3V	R_L=600Ω，C_L=100pF	1.29	2.18	2.5	V/μs
GBW	增益带宽	3.3V	R_L=600Ω，C_L=100pF	2.05	3.70	7.16	MHz
A_{OL}	开环增益	3.3V	R_L=600Ω，C_L=100pF	—	96	—	dB
PM	相位余量	3.3V	R_L=600Ω，C_L=100pF	—	90	—	—

比较器电气特性

Ta=25°C

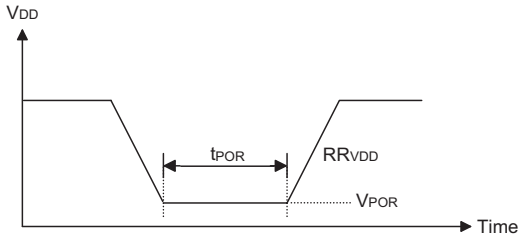
符号	参数	测试条件		最大	典型	最大	单位
		V_{DD}	条件				
V_{CMP}	比较器工作电压	5V	—	2.2	5.0	5.5	V
I_{CMP}	比较器工作电流	5V	—	—	300	450	μA
I_{OFF}	暂停模式时比较器的工作电流	5V	比较器除能	—	—	0.1	μA
V_{CMPOS}	比较器输入失调电压	5V	—	-10	—	+10	mV
V_{HYS0}	迟滞宽度	5V	比较器 0	TBC	100	TBC	mV
V_{HYS1}	迟滞宽度	5V	比较器 1，2，3	20	40	60	mV
V_{CM}	共模输入电压范围	—	—	V_{SS}	—	V_{DD}-1.4	V
AOL	比较器开环增益	—	—	100	120	—	dB
t_{PD}	比较器响应时间	5V	V_{CM}=0~(V_{DD}-1.4)V 10mV 偏置	—	—	1	μs
t_{PD}	比较器响应时间	5V	100mV 偏置（注）	—	—	200	ns

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

上电复位特性

Ta=25°C

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

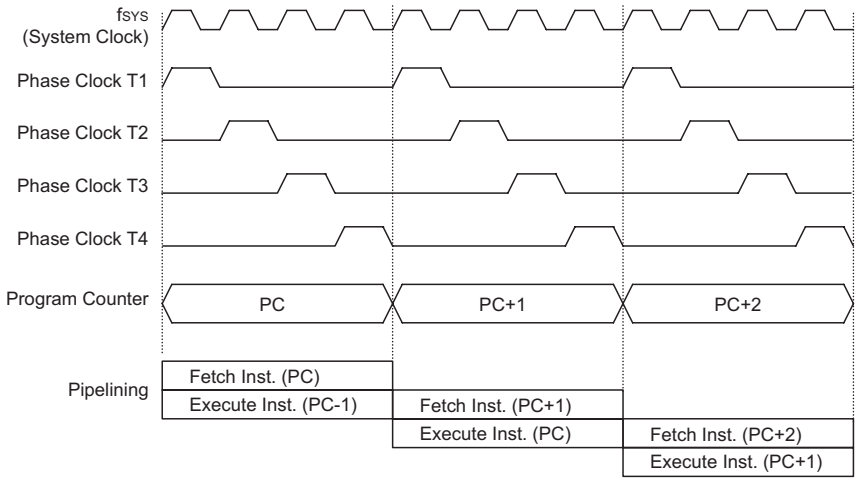


系统结构

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

时序和流水线结构

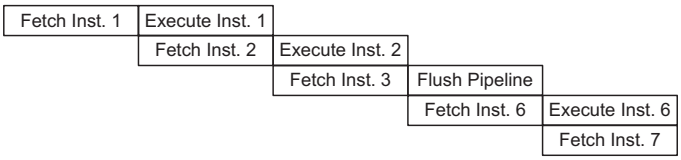
主系统时钟由 HIRC 或 LIRC 振荡器提供，它被细分为 T1~T4 四个内部产生的非重叠时序。在 T1 时间，程序计数器自动加一并抓取一条新的指令。剩下的时间 T2~T4 完成译码和执行功能，因此，一个 T1~T4 时钟周期构成一个指令周期。虽然指令的抓取和执行发生在连续的指令周期，但单片机流水线结构会保证指令在一个指令周期内被有效执行。除非程序计数器的内容被改变，如子程序的调用或跳转，在这种情况下指令将需要多一个指令周期的时间去执行。



系统时序和流水线

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

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



指令捕捉

程序计数器

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

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

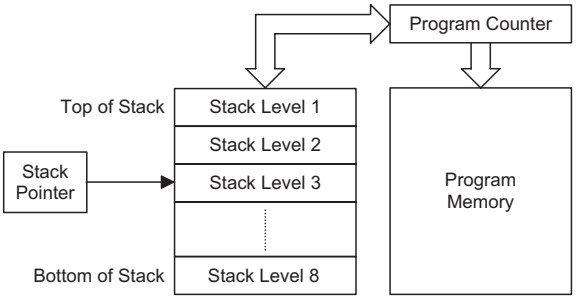
程序计数器	
程序计数器高字节	PCL 寄存器
PC11~PC8	PCL7~PCL0

程序计数器

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

堆栈

堆栈是一个特殊的存储空间，用来存储程序计数器中的内容。该单片机有 8 层堆栈，堆栈既不是数据部分也不是程序空间部分，而且它既不是可读取也不是可写入的。当前层由堆栈指针 (SP) 加以指示，同样也是不可擦写的。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)，中断将被回应。这个特性提供程序设计者简单的方法来预防堆栈溢出。然而即使堆栈已满，CALL 指令仍然可以被执行，而造成堆栈溢出。使用时应避免堆栈溢出的情况发生，因为这可能导致不可预期的程序分支指令执行错误。若堆栈溢出，则首个存入堆栈的程序计数器数据将会丢失。

算术逻辑单元 – ALU

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

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

Flash 程序存储器

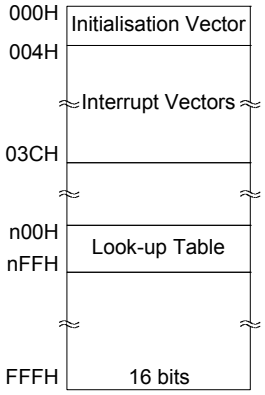
程序存储器用来存放用户代码即储存程序。程序存储器为 FLASH 类型意味着可以多次重复编程，方便用户使用同一芯片进行程序的修改。使用适当的单片机编程工具，此单片机提供用户灵活便利的调试方法和项目开发规划及更新。

结构

程序存储器的容量为 4K×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。

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。



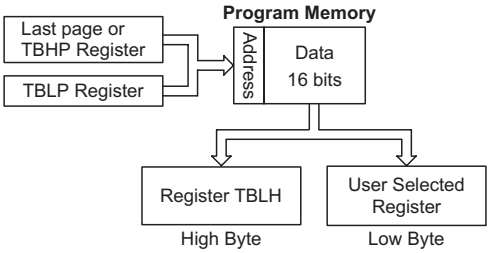
程序存储器结构

查表

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

在设定完表格指针后，表格数据可以使用“TABRD [m]”或“TABRDL [m]”指令分别从程序存储器查表读取。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的最后一页，在此 ORG 伪指令的值为“0F00H”指向的地址是 4K 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 F06H，即最后一页起始地址后的第六个地址。值得注意的是，假如“TABRD [m]”指令被使用，则表格指针指向特定页页。在这个例子中，表格数据的高字节等于零，而当“TABRD [m]”指令被执行时，此值将会自动的被传送到 TBLH 寄存器。

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

表格读取程序举例

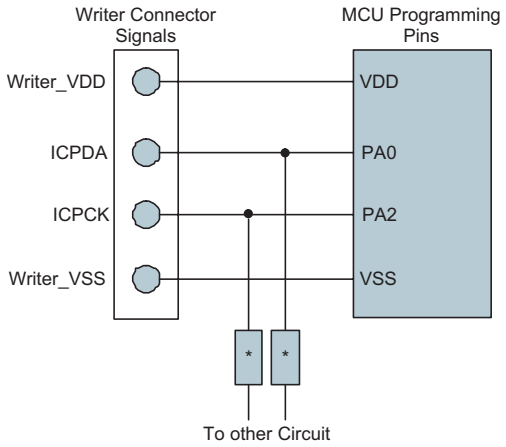
```
tempreg1 db ?       ; temporary register #1
tempreg2 db ?       ; temporary register #2
:
:
mov a,06h           ; initialise low table pointer - note that this address
mov tblp,a          ; is referenced
mov a,0Fh           ; initialise high table pointer
mov tbhp,a
:
:
tabrd tempreg1      ; transfers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

在线烧录

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

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

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



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

EV IC 用于单片机仿真。此 EV IC 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 IC 和实际 MCU 在功能上几乎是兼容的。用户可将 OCDSDA 和 OCDSCK 引脚连接至 Holtek HT-IDE 开发工具，从而实现 EV IC 对实际 IC 的仿真。OCDSDA 引脚为 OCDS 数据 / 地址输入 / 输出脚，OCDSCK 引脚为 OCDS 时钟输入脚。当用户用 EV IC 进行调试时，实际单片机 OCDSDA 和 OCDSCK 引脚上的其它共用功能无效。关于 OCDS 功能的详细描述，请参考“Holtek e-Link for 8-bit MCU OCDS User's Guide”文件。

Holtek e-Link 引脚名称	EV IC 引脚名称	功能
OCDSDA	OCDSDA	片上调试数据 / 地址输入 / 输出
OCDSCK	OCDSCK	片上调试时钟输入
VDD	VDD	电源
VSS	VSS	地

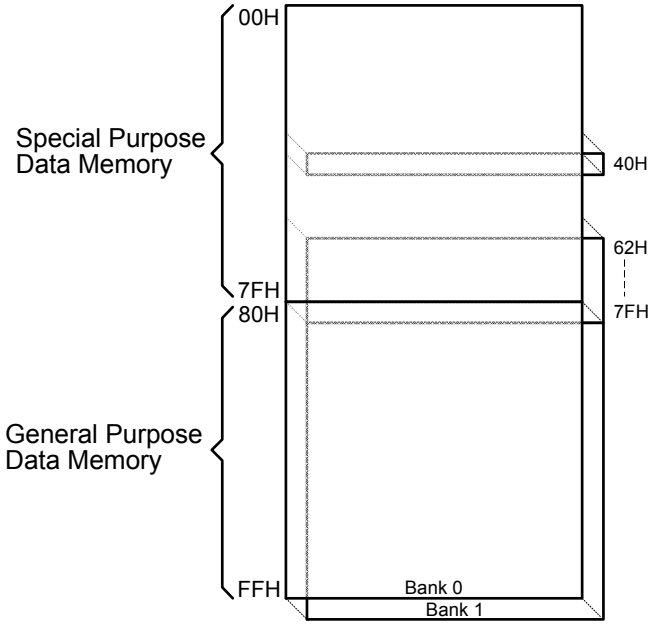
RAM 数据存储器

RAM 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。该单片机数据存储器的容量为 256×8。

结构

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

总的数据存储器被分为两个区。大部分特殊功能数据寄存器均可在所有 Bank 被访问，处于“40H”地址的 EEC 寄存器却只能在 Bank 1 中被访问到。切换不同区域可通过设置区域指针 (BP) 实现。所有单片机的数据存储器的起始地址都是“00H”。



数据存储器结构

	Bank 0 ~ Bank1
00H	IAR0
01H	MP0
02H	IAR1
03H	MP1
04H	BP
05H	ACC
06H	PCL
07H	TBLP
08H	TBLH
09H	TBHP
0AH	STATUS
0BH	SMOD
0CH	LVDC
0DH	LVRC
0EH	WDTC
0FH	TBC
10H	INTEG1
11H	INTC0
12H	INTC1
13H	INTC2
14H	INTC3
15H	MFI0
16H	MFI1
17H	MFI2
18H	MFI3
19H	MFI4
1AH	MFI5
1BH	MFI6
1CH	PAWU
1DH	PAPU
1EH	PA
1FH	PAC
20H	PBPU
21H	PB
22H	PBC
23H	PCPU
24H	PC
25H	PCC
26H	PDPU
27H	PD
28H	PDC
29H	CAPTC0
2AH	CAPTC1
2BH	CAPTMDL
2CH	CAPTMDH
2DH	CAPTMAL
2EH	CAPTMAH
2FH	CAPTMCL
30H	CAPTMCH
31H	ADRL
32H	ADRH
33H	ADCR0
34H	ADCR1
35H	ADCR2
36H	ADDL
37H	ADLVDL
38H	ADLVDH
39H	ADHVDL
3AH	ADHVDH
3BH	PBPS0
3CH	PBPS1
3DH	PBPS2
3EH	INTEG0
3FH	CTRL

	Bank 0	Bank 1
40H	Unused	EEC
41H	EEA	
42H	EED	
43H	HDCR	
44H	HDCD	
45H	MPTC1	
46H	MPTC2	
47H	PTM1C0	
48H	PTM1C1	
49H	PTM1DL	
4AH	PTM1DH	
4BH	PTM1AL	
4CH	PTM1AH	
4DH	PTM1RPL	
4EH	PTM1RPH	
4FH	PTM0C0	
50H	PTM0C1	
51H	PTM0DL	
52H	PTM0DH	
53H	PTM0AL	
54H	PTM0AH	
55H	PTM0RPL	
56H	PTM0RPH	
57H	PTM2C0	
58H	PTM2C1	
59H	PTM2DL	
5AH	PTM2DH	
5BH	PTM2AL	
5CH	PTM2AH	
5DH	PTM2RPL	
5EH	PTM2RPH	
5FH	PWMC	
60H	DUTR0L	
61H	DUTR0H	
62H	DUTR1L	HDCT0
63H	DUTR1H	HDCT1
64H	DUTR2L	HDCT2
65H	DUTR2H	HDCT3
66H	PRDRL	HDCT4
67H	PRDRH	HDCT5
68H	PWMRL	HDCT6
69H	PWMRH	HDCT7
6AH	PWMME	HDCT8
6BH	PWMMD	HDCT9
6CH	MCF	HDCT10
6DH	MCD	HDCT11
6EH	DTS	OPOMS
6FH	PLC	OPCM
70H	IICC0	OPACAL
71H	IICC1	PTM3C0
72H	IICD	PTM3C1
73H	IICA	PTM3DL
74H	I2CTOC	PTM3DH
75H	USR	PTM3AL
76H	UCR1	PTM3AH
77H	UCR2	PTM3RPL
78H	BRG	PTM3RPH
79H	Unused	
7AH	TXR_RXR	PAPS0
7BH	CPC	PAPS1
7CH	HCHK_NUM	PCPS0
7DH	HNF_MSEL	PCPS1
7EH	NF_VIH	PDPS0
7FH	NF_VIL	PRM

: Unused, read as 00H

特殊功能数据存储器结构

特殊功能寄存器

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

间接寻址寄存器 – IAR0, IAR1

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

存储器指针 – MP0, MP1

该单片机提供两个存储器指针，即 MP0 和 MP1。由于这些指针在数据存储器中能像普通的寄存器一般被操作，因此提供了一个寻址和数据追踪的有效方法。当对间接寻址寄存器进行任何操作时，单片机指向的实际地址是由存储器指针所指定的地址。使用 MP0 和 IAR0 可访问存储区 0，而使用 MP1 和 IAR1 可访问存储区 0 和存储区 1。

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

间接寻址程序举例

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

在上面的例子中有一点值得注意，即并没有确定 RAM 地址。

存储区指针 – BP

数据存储区被分为两个部分，可以通过设置存储区指针 (Bank Pointer) 值来访问不同的程序和数据存储区。BP 指针的第 0 位用于选择数据存储区 0 或 1。

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

BP 寄存器

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

Bit 7~1　未定义，读为“0”

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

累加器 – ACC

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

程序计数器低字节寄存器 – PCL

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

表格寄存器 – TBLP，TBHP，TBLH

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

状态寄存器 – STATUS

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

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

Z、OV、AC 和 C 标志位通常反映最近运算的状态。

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

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

STATUS 寄存器

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

“x”：未知

Bit 7~6　未定义，读为“0”

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

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

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

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

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

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

EEPROM 数据存储器

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

EEPROM 数据存储器结构

EEPROM 数据寄存器容量为 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 Bank 0 中的一个地址和数据寄存器以及 Bank 1 中的一个控制寄存器，可以实现对 EEPROM 的单字节读写操作。

EEPROM 寄存器

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

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

EEPROM 寄存器列表

EEA 寄存器

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

Bit 7~6 未定义，读为“0”

Bit 5~0 **D5~D0**：数据 EEPROM 地址
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 数据 EEPROM 数据
数据 EEPROM 数据 bit 7 ~ bit 0

EEC 寄存器

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

Bit 7~4 未定义，读为“0”

Bit 3 **WREN**：数据 EEPROM 写使能位
0：除能
1：使能
此位为数据 EEPROM 写使能位，向数据 EEPROM 写操作之前需将此位置高。将此位清零时，则禁止向数据 EEPROM 写操作。

Bit 2 **WR**：EEPROM 写控制位
0：写周期结束
1：写周期有效
此位为数据 EEPROM 写控制位，由应用程序将此位置高将启动写周期。写周期结束后，硬件自动将此位清零。当 WREN 未先置高时，此位置高无效。

Bit 1 **RDEN**：数据 EEPROM 读使能位
0：除能
1：使能
此位为数据 EEPROM 读使能位，向数据 EEPROM 读操作之前需将此位置高。将此位清零时，则禁止向数据 EEPROM 读操作。

Bit 0 **RD**：EEPROM 读控制位
0：读周期结束
1：读周期有效
此位为数据 EEPROM 读控制位，由应用程序将此位置高将启动读周期。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 EREN、ER、WREN、WR、RDEN 和 RD 不能同时置为“1”。
2. 确保 f_{SUB} 时钟在执行擦 / 写动作前已稳定。
3. 确保擦 / 写动作完成后才可改写 EEPROM 相关寄存器。

从 EEPROM 中读取数据

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

写数据到 EEPROM

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

写保护

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

EEPROM 中断

EEPROM 写周期结束后将产生 EEPROM 写中断，需先通过设置相关中断寄存器的 EPWE 位使能 EEPROM 中断。当 EEPROM 写周期结束，EPWF 请求标志位将被置位。若 EEPROM 中断使能且堆栈未满的情况下将跳转到相应的中断向量中执行。当中断被响应，只有 EEPROM 中断标志位将自动复位。更多细节将在中断章节讲述。

编程注意事项

必须注意的是数据不会无意写入 EEPROM。在没有写动作时写使能位被正常清零可以增强保护功能。BP 指标也可以正常清零以阻止进入 EEPROM 控制寄存器存在的 Bank 1。尽管没有必要，写一个简单的读回程序以检查新写入的数据是否正确还是应该考虑的。往 EEPROM 写数据时，WREN 位设为高时需立刻将 WR 位设为高以保证能正确的执行写周期。总中断使能位 EMI 也必须在执行写周期前清零然后在写周期开始前重新使能。在 IDLE0/IDLE1/SLEEP 模式下不可对 EEPROM 进行写入的动作 , 否则会写入失败。

程序举例

从 EEPROM 中读取数据—轮询法

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

写数据到 EEPROM—轮询法

```
MOV A, EEPROM_ADRES                ; user defined address
MOV EEA, A
MOV A, EEPROM_DATA                 ; user defined data
MOV EED, A
MOV A, 040H                        ; setup memory pointer MP1
MOV MP1, A                         ; MP1 points to EEC register
MOV A, 01H                         ; setup Bank Pointer
MOV BP, A
CLR  EMI
SET IAR1.3                         ; set WREN bit, enable write operations
SET IAR1.2                         ; start Write Cycle - set WR bit
SET  EMI
BACK:
SZ IAR1.2                          ; check for write cycle end
JMP BACK
CLR IAR1                           ; disable EEPROM write
CLR BP
```

振荡器

不同的振荡器选择可以让使用者在不同的应用需求中实现更大范围的功能。振荡器的灵活性使得在速度和功耗方面可以达到较佳的优化。通过寄存器选择振荡器及其工作。

振荡器概述

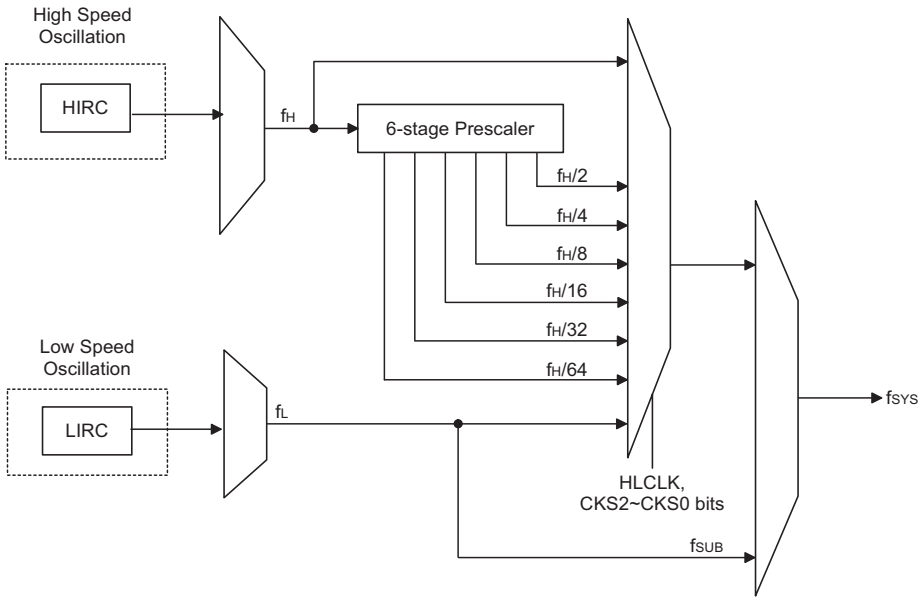
振荡器除了作为系统时钟源，还作为看门狗定时器和时基功能的时钟源。外部振荡器需要一些外围器件，而集成的两个内部振荡器不需要任何外围器件。它们提供的高速和低速系统振荡器具有较宽的频率范围。较高频率的振荡器提供更高的性能，但要求有更高的功率，反之亦然。动态切换快慢系统时钟的能力使单片机具有灵活而优化的性能 / 功耗比，此特性对功耗敏感的应用领域尤为重要。

类型	名称	频率
内部高速 RC	HIRC	20MHz
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，内部高速振荡器和内部低速振荡器。高速振荡器有内部 20MHz RC 振荡器，低速振荡器为内部 32kHz 振荡器。使用高速或低速振荡器作为系统时钟的选择是通过设置 SMOD 寄存器中的 HLCLK 位及 CKS2~CKS0 位决定的，系统时钟可动态选择。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 20MHz 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 振荡器具有一种固定的频率：20MHz。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡频率因 V_{DD}、温度以及芯片制成工艺不同的影响较大程度地降低。在电源电压为 5V 及温度为 25°C 的条件下，20MHz 这个固定频率的容差为 2%。

内部 32kHz 振荡器 – LIRC

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

辅助振荡器

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

工作模式和系统时钟

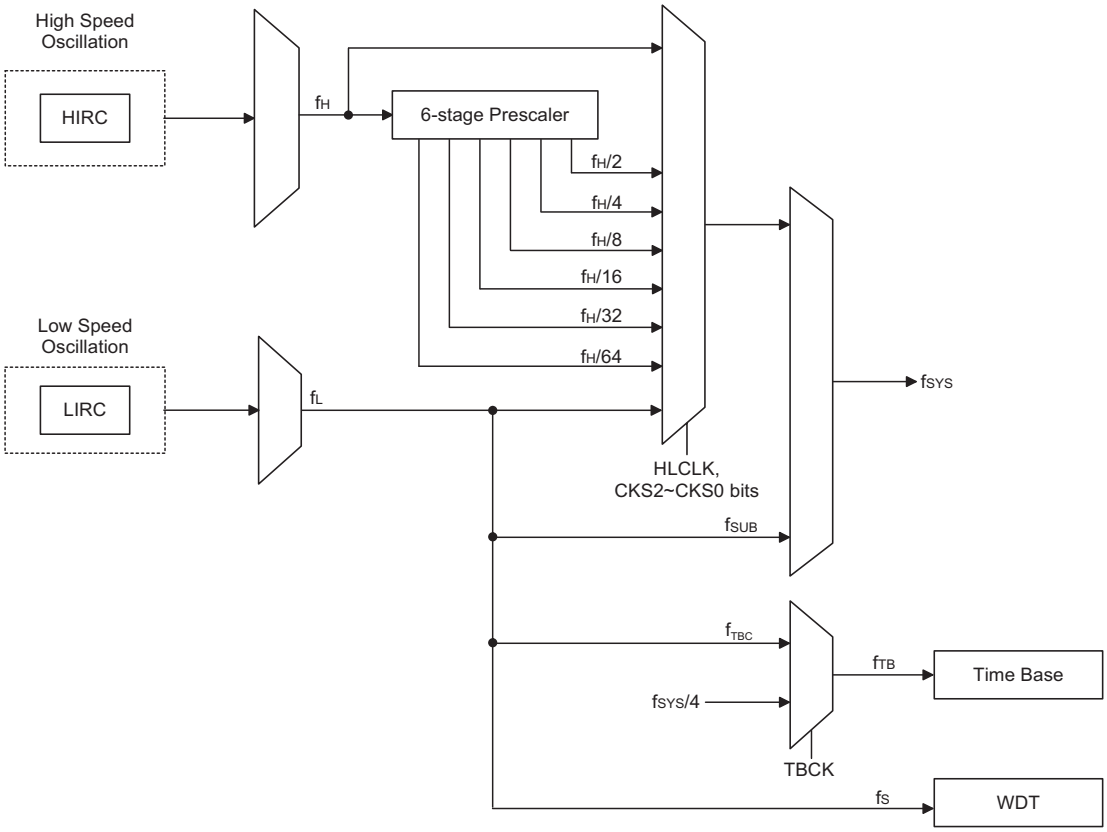
现今的应用要求单片机具有较高的性能及尽可能低的功耗，这种矛盾的要求在便携式电池供电的应用领域尤为明显。高性能所需要的高速时钟将增加功耗，反之亦然。此单片机提供高、低速两种时钟源，它们之间可以动态切换，用户可通过优化单片机操作来获得较佳性能 / 功耗比。

系统时钟

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

主系统时钟可来自高频时钟源 f_H 或低频时钟源 f_L，通过 SMOD 寄存器中的 HLCLK 位及 CKS2~CKS0 位进行选择。高频时钟来自 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 $f_H/2$~$f_H/64$。

另外两个内部时钟用于外围电路，次时钟源 f_{SUB} 和时基时钟 f_{TBC}。这两个时钟源来自 LIRC 振荡器。



系统时钟选项

注：当系统时钟源 f_{SYS} 由 f_H 到 f_L 转换时，高速振荡器将停止以节省耗电。因此，没有为外围电路提供 $f_H/2$~$f_H/64$ 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：正常模式和

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_{SUB}	f_S	f_{TBC}
正常模式	On	f_H~f_H/64	On	On	On
低速模式	On	f_L	On	On	On
空闲模式 0	Off	Off	On	On	On
空闲模式 1	Off	On	On	On	On
休眠模式	Off	Off	On	On	Off

正常模式

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

低速模式

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

休眠模式

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，而 f_L 时钟将继续运行。

空闲模式 0

执行 HALT 指令后且 SMOD 寄存器中 IDLEN 位为高，CTRL 寄存器中 FSYSON 位为低时，系统进入空闲模式 0。在空闲模式 0 中，CPU 停止，但一些外围功能如看门狗定时器，TMs 和 I^2C 将继续工作。在空闲模式 0 中，系统振荡器停止，看门狗定时器时钟 f_S 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，但会提供一个时钟源给一些外围功能如看门狗定时器和 TMs。在空闲模式 1 中，系统振荡器继续运行，该系统振荡器可以为高速或低速系统振荡器。在该模式中看门狗定时器时钟 f_S 开启。

控制寄存器

寄存器 SMOD 用于控制单片机内部时钟。

SMOD 寄存器

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

Bit 7~5 **CKS2~CKS0**：当 HLCLK 为“0”时系统时钟选择位
000：f_L (f_{LIRC})
001：f_L (f_{LIRC})
010：f_H/64
011：f_H/32
100：f_H/16
101：f_H/8
110：f_H/4
111：f_H/2
这三位用于选择系统时钟源。除了 LIRC 振荡器提供的系统时钟源外，也可使用高频振荡器的分频作为系统时钟。

Bit 4 未定义，读为“0”

Bit 3 **LTO**：LIRC 系统振荡器 SST 就绪标志位
0：未就绪
1：就绪
此位为低速系统振荡器 SST 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。若系统时钟来自 LIRC 振荡器，该位转换为高需 1~2 个时钟周期。

Bit 2 **HTO**：HIRC 系统振荡器 SST 就绪标志位
0：未就绪
1：就绪
此位为高速系统振荡器就绪标志位，用于表明高速系统振荡器何时稳定下来。此标志在系统上电后经硬件清零，高速系统振荡器稳定后变为高电平。因此，此位在单片机上电后由应用程序读取的总为“1”。该标志由休眠模式或空闲模式 0 中唤醒后会处于低电平状态，若 HIRC 振荡器则只需 15~16 个时钟周期即可。

Bit 1 **IDLEN**：空闲模式控制位
0：除能
1：使能
此位为空闲模式控制位，用于决定 HALT 指令执行后发生的动作。若此位为高，当指令 HALT 执行后，单片机进入空闲模式。若 FSYSON 位为高，在空闲模式 1 中 CPU 停止运行，系统时钟将继续工作以保持外围功能继续工作；若 FSYSON 为低，在空闲模式 0 中 CPU 和系统时钟都将停止运行。若此位为低，单片机将在 HALT 指令执行后进入休眠模式。

Bit 0 **HLCLK**：系统时钟选择位
0：f_H/2~ f_H/64 或 f_L
1：f_H
此位用于选择 f_H 或 f_H/2~ f_H/64 还是 f_L 作为系统时钟。该位为高时选择 f_H 作为系统时钟，为低时则选择 f_H/2~ f_H/64 或 f_L 作为系统时钟。当系统时钟由 f_H 时钟向 f_L 时钟转换时，f_H 将自动关闭以降低功耗。

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**：IDLE 模式下 f_{SYS} 控制位
0：除能
1：使能

Bit 6~3 未定义，读为“0”

Bit 2 **LVRF**：LVR 复位标志位
0：未发生
1：发生
当 LVR 复位情况发生时此位设置为“1”。此位只能通过程序清零。

Bit 1 **LRF**：LVR 控制寄存器 LVRC 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”，用作软件复位功能。此位只能通过程序清零。

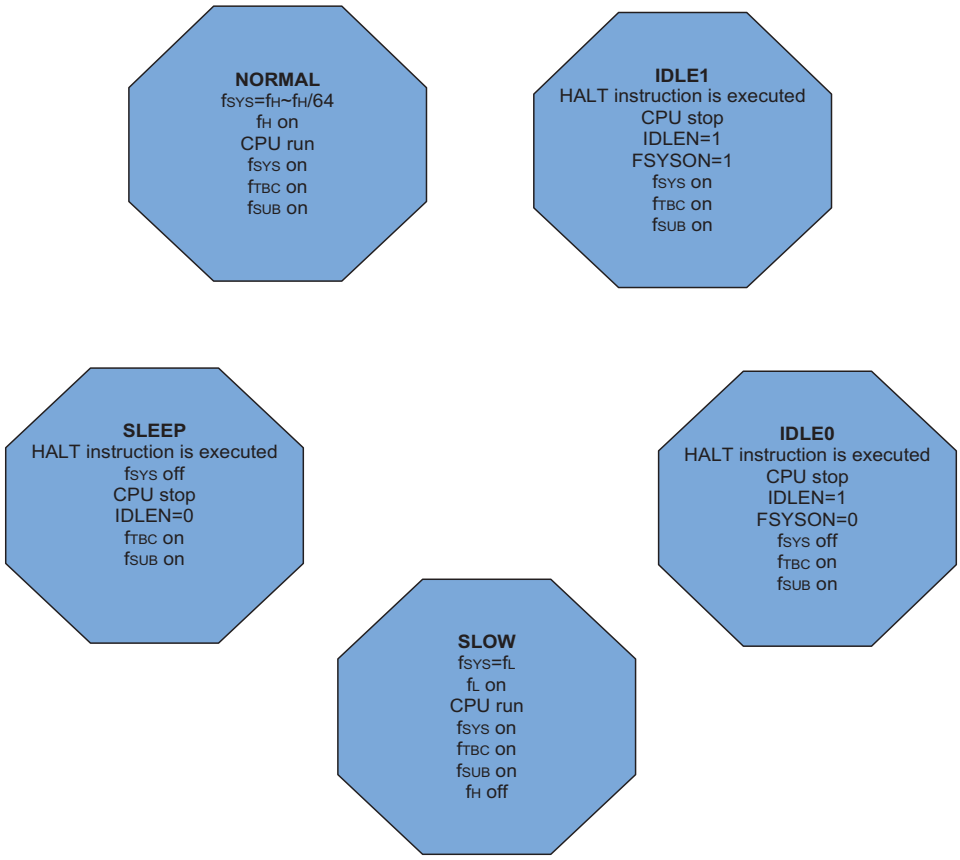
Bit 0 **WRF**：WDT 控制寄存器软件复位标志位
0：未发生
1：发生
当 WDT 控制寄存器软件复位时此位设置为“1”。此位只能通过程序清零。

工作模式切换和唤醒

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

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

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



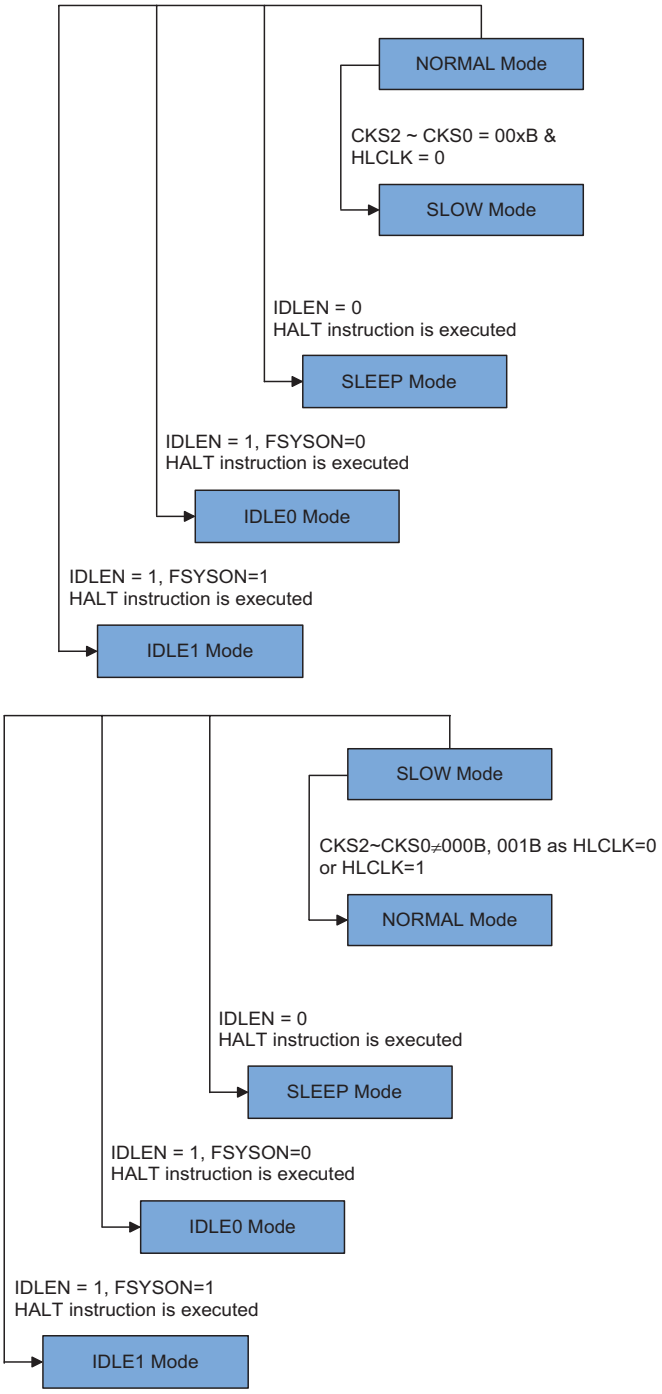
正常模式切换到低速模式

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

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

低速模式切换到正常模式

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


NORMAL Mode
CKS2 ~ CKS0 = 00xB &
HLCLK = 0
SLOW Mode
IDLEN = 0
HALT instruction is executed
SLEEP Mode
IDLEN = 1, FSYSON=0
HALT instruction is executed
IDLE0 Mode
IDLEN = 1, FSYSON=1
HALT instruction is executed
IDLE1 Mode
SLOW Mode
CKS2~CKS0≠000B, 001B as HLCLK=0
or HLCLK=1
NORMAL Mode
IDLEN = 0
HALT instruction is executed
SLEEP Mode
IDLEN = 1, FSYSON=0
HALT instruction is executed
IDLE0 Mode
IDLEN = 1, FSYSON=1
HALT instruction is executed
IDLE1 Mode

进入休眠模式

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

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

进入空闲模式 0

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

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

进入空闲模式 1

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

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

静态电流的注意事项

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

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

唤醒

系统进入休眠或空闲模式之后，可以通过以下几种方式唤醒：

- PA 口下降沿
- 系统中断
- WDT 溢出

若由 WDT 溢出唤醒，则会发生看门狗定时器复位。可以通过状态寄存器中 TO 和 PDF 位来判断它的唤醒源。系统上电或执行清除看门狗的指令，会清零 PDF；执行 HALT 指令，PDF 将被置位。看门狗计数器溢出将会置位 TO 标志并唤醒系统，这种复位会重置程序计数器和堆栈指针，其它标志保持原有状态。

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

看门狗定时器

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

看门狗定时器时钟源

WDT 定时器时钟源来自于内部时钟 f_S，而 f_S 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 2^8~2^{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。

WDT 总是使能，它可以通过 WDTC 寄存器来设置。

看门狗定时器控制寄存器

WDTC 寄存器用于控制 WDT 功能的使能 / 除能及选择溢出周期。任何单片机复位，WDTC 都为 01010011B，且这个值在暂停模式下 WDT 溢出也不会改变。

WDTC 寄存器

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

Bit 7~3 **WE4~WE0**：WDT 功能软件控制
10101B 或 01010B：使能
其它值：MCU 复位 (需要 1~2 个 LIRC 周期响应复位)
如果单片机复位且由 WDT 寄存器中的 WE[4:0] 引起，则 CTRL 寄存器中的 WRF 标志位会被置位。

Bit 2~0 **WS2~WS0**：WDT 溢出周期选择位
000：$2^8/f_S$
001：$2^{10}/f_S$
010：$2^{12}/f_S$
011：$2^{14}/f_S$
100：$2^{15}/f_S$
101：$2^{16}/f_S$
110：$2^{17}/f_S$
111：$2^{18}/f_S$
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**：IDLE 模式下 f_{SYS} 控制位
其它地方有描述。

Bit 6~3 未定义，读为“0”

Bit 2 **LVRF**：LVR 复位标志位
其它地方有描述。

Bit 1 **LRF**：LVR 控制寄存器 LVRC 软件复位标志位
其它地方有描述。

Bit 0 **WRF**：WDT 控制寄存器软件复位标志位
0：未发生
1：发生
当 WDT 控制寄存器软件复位时此位设置为“1”。此位只能通过程序清零。

看门狗定时器操作

该单片机的看门狗定时器时钟源来自 LIRC 振荡器，总是开启的。当 WDT 溢出时，它产生一个芯片复位的动作。这也就意味着正常工作期间，用户需在应用程序中看门狗溢出前有策略地清看门狗定时器以防止其产生复位，可使用清除看门狗指令实现。无论什么原因，程序失常跳转到一个未知的地址或进入一个死循环，这些清除指令都不能被正确执行，此种情况下，看门狗将溢出以使单片机复位。WDTC 寄存器中的 WE4~WE0 位可使能看门狗定时器。如果 WE4~WE0 为 10101B 或 01010B，则 WDT 总是使能，如果 WE4~WE0 为其它任意值，则经过 2~3 个 LIRC 时钟周期后单片机复位。

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

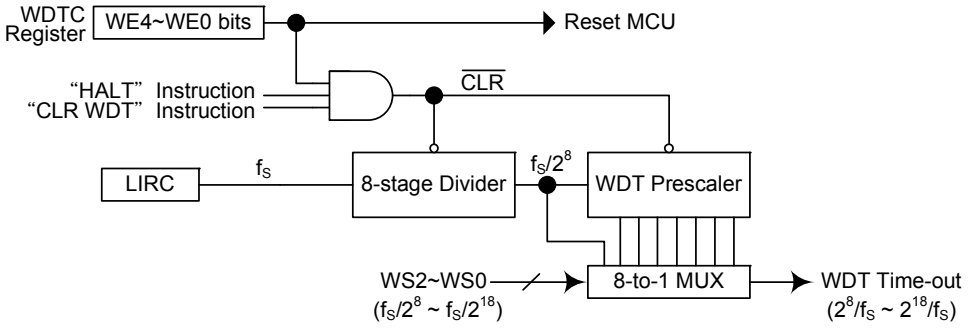
看门狗定时器使能 / 除能控制

程序正常运行时，WDT 溢出将导致芯片复位，并置位状态标志位 TO。若系统处于休眠或空闲模式，当 WDT 发生溢出时，状态寄存器中的 TO，程序计数器 PC 和堆栈指针 SP 将被置位。有三种方法可以用来清除 WDT 的内容。第一种是通过外部上电复位，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 LIRC 振荡器，分频比为 2^{18} 时最大溢出周期约 8s，分频比为 2^8 时最小溢出周期约 7.8ms。



看门狗定时器

复位和初始化

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

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

复位功能

包括内部和外部事件触发复位，单片机共有四种复位方式：

上电复位

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

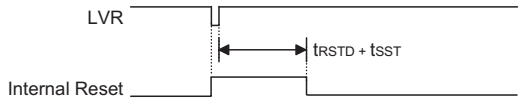
注：t_{RSTD} 为上电延迟时间，典型值为 50ms

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。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~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。LVR 包含以下的规格：有效的 LVR 信号，即在 0.9V~V_{LVR}

的低电压状态的时间，必须超过交流电气特性中 t_{LVR} 参数的值。如果低电压存在不超过 t_{LVR} 参数的值，则 LVR 将会忽略它且不会执行复位功能。V_{LVR} 参数值可通过 LVRC 寄存器中的 LVS 位设置固定为 3.15V。若由于受到干扰 LVS7~LVS0 变为其它值时，需经过 2~3 个 LIRC 周期响应复位。此时 CTRL 寄存器的 LRF 位被置位。上电后寄存器的值为 01010101B。



注：t_{RSTD} 为上电延迟时间，典型值为 16.7ms。

低电压复位时序图

● **LVRC 寄存器**

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

Bit 7~0 **LVS7~LVS0**：LVR 电压选择
01010101B：3.15V
00110011B：3.15V
10011001B：3.15V
10101010B：3.15V
其它值：MCU 复位 (需要 2~3 个 LIRC 周期响应复位)
注：可以通过 S/W 写 00H~FFH 来控制 LVR 电压，也可复位单片机。如果单片机复位且由 LVRC 寄存器引起，则在复位后 CTRL 寄存器中的 LRF 标志位会被置位。

● **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**：IDLE 模式下 f_{SYS} 控制位
其它地方有描述。

Bit 6~3 未定义，读为“0”

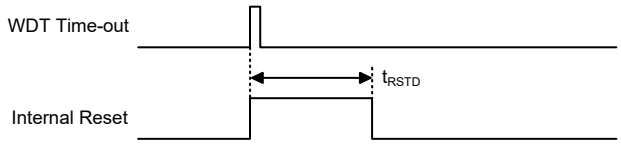
Bit 2 **LVRF**：LVR 复位标志位
0：未发生
1：发生
当 LVR 复位情况发生时此位设置为“1”。此位只能通过程序清零。

Bit 1 **LRF**：LVR 控制寄存器 LVRC 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”，用作软件复位功能。此位只能通过程序清零。

Bit 0 **WRF**：WDT 控制寄存器软件复位标志位
其它地方有描述。

正常运行时看门狗溢出复位

除了看门狗溢出标志位 TO 将被设为“1”之外。

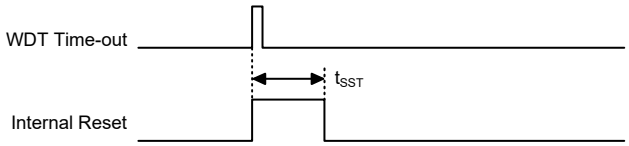


注：t_{RSTD} 为上电延迟时间，典型值为 16.7ms。

正常运行时看门狗溢出时序图

休眠或空闲时看门狗溢出复位

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



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

休眠或空闲时看门狗溢出复位时序图

复位初始状态

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

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

注：“u”代表不改变

在单片机上电复位之后，各功能单元初始化的情形，列于下表。

项目	复位后情况
程序计数器	清除为零
中断	所有中断被除能
看门狗定时器	WDT 清除并重新计数
定时器模块	所有定时器模块停止
输入 / 输出口	I/O 口设为输入模式，AN0~AN7 作为 A/D 输入脚。
堆栈指针	堆栈指针指向堆栈顶端

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

寄存器	上电复位	看门狗溢出 (正常操作)	低电压复位	看门狗溢出 (暂停)
MP0	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
MP1	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
BP	---- ---0	---- ---0	---- ---0	---- ---u
ACC	xxxx xxxx	uuuu uuuu	xxxx xxxx	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	xxxx xxxx	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	xxxx xxxx	uuuu uuuu
TBHP	---- xxxx	---- uuuu	---- xxxx	---- uuuu
STATUS	--00 xxxx	--1u uuuu	--uu xxxx	--11 uuuu
SMOD	0000 0011	0000 0011	0000 0011	uuuu uuuu
LVDC	--00 -000	--00 -000	--00 -000	--uu –uuu
LVRC	0101 0101	0101 0101	uuuu uuuu	uuuu uuuu
WDTC	0101 0011	0101 0011	0101 0011	uuuu uuuu
TBC	0011 ----	0011 ----	0011 ----	uuuu ----
INTEG1	---- --00	---- --00	---- --00	---- --uu
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	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI0	-000 -000	-000 -000	-000 -000	-uuu -uuu
MFI1	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI2	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI3	--00 --00	--00 --00	--00 --00	--uu --uu
MFI4	--00 --00	--00 --00	--00 --00	--uu --uu
MFI5	--00 --00	--00 --00	--00 --00	--uu --uu
MFI6	--00 --00	--00 --00	--00 --00	--uu --uu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCPU	--00 0000	--00 0000	--00 0000	--uu uuuu
PC	--11 1111	--11 1111	--11 1111	--uu uuuu
PCC	--11 1111	--11 1111	--11 1111	--uu uuuu
PDPU	---- 0000	---- 0000	---- 0000	---- uuuu
PDC	---- 1111	---- 1111	---- 1111	---- uuuu
PD	---- 1111	---- 1111	---- 1111	---- uuuu
CAPTC0	0000 0-00	0000 0-00	0000 0-00	uuuu u-uu
CAPTC1	0000 0000	0000 0000	0000 0000	uuuu uuuu

寄存器	上电复位	看门狗溢出 (正常操作)	低电压复位	看门狗溢出 (暂停)
CAPTMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
CAPTMDH	0000 0000	0000 0000	0000 0000	uuuu uuuu
CAPTMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
CAPTMAH	0000 0000	0000 0000	0000 0000	uuuu uuuu
CAPTMCL	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
CAPTMCH	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRL (ADRFS=0)	xxxx ----	xxxx ----	xxxx ----	uuuu ----
ADRL (ADRFS=1)	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH (ADRFS=0)	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADRH (ADRFS=1)	---- xxxx	---- xxxx	---- xxxx	---- uuuu
ADCR0	0110 0000	0110 0000	0110 0000	Uuuu uuuu
ADCR1	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADCR2	---- --00	---- --00	---- --00	---- --uu
ADDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADLVDL (ADRFS=0)	0000 ----	0000 ----	0000 ----	uuuu ----
ADLVDL (ADRFS=1)	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADLVDH (ADRFS=0)	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADLVDH (ADRFS=1)	---- 0000	---- 0000	---- 0000	---- uuuu
ADHVDL (ADRFS=0)	0000 ----	0000 ----	0000 ----	uuuu ----
ADHVDL (ADRFS=1)	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADHVDH (ADRFS=0)	0000 0000	0000 0000	0000 0000	uuuu uuuu
ADHVDH (ADRFS=1)	---- 0000	---- 0000	---- 0000	---- uuuu
PBPS0	-000 0000	-000 0000	-000 0000	-uuu uuuu
PBPS1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBPS2	---- --00	---- --00	---- --00	---- --uu
INTEG0	-000 0000	-000 0000	-000 0000	-uuu uuuu
CTRL	0--- -x00	0--- -x00	0--- -x00	u--- -uuu
EEC	---- 0000	---- 0000	---- 0000	---- uuuu
EEA	-xxx xxxx	-xxx xxxx	-xxx xxxx	-uuu uuuu
EED	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
HDCR	0001 0000	0001 0000	0001 0000	uuuu uuuu
HDCD	---- -000	---- -000	---- -000	---- -uuu
MPTC1	0000 00--	0000 00--	0000 00--	uuuu uu--
MPTC2	---0 0000	---0 0000	---0 0000	---U UUUU
HDCT0	--00 0000	--00 0000	--00 0000	--uu uuuu
HDCT1	--00 0000	--00 0000	--00 0000	--uu uuuu
HDCT2	--00 0000	--00 0000	--00 0000	--uu uuuu
HDCT3	--00 0000	--00 0000	--00 0000	--uu uuuu
HDCT4	--00 0000	--00 0000	--00 0000	--uu uuuu
HDCT5	--00 0000	--00 0000	--00 0000	--uu uuuu

寄存器	上电复位	看门狗溢出 (正常操作)	低电压复位	看门狗溢出 (暂停)
HDCT6	--00 0000	--00 0000	--00 0000	--uu uuuu
HDCT7	--00 0000	--00 0000	--00 0000	--uu uuuu
HDCT8	--00 0000	--00 0000	--00 0000	--uu uuuu
HDCT9	--00 0000	--00 0000	--00 0000	--uu uuuu
HDCT10	--00 0000	--00 0000	--00 0000	--uu uuuu
HDCT11	--00 0000	--00 0000	--00 0000	--uu uuuu
PTM1C0	0000 0---	0000 0---	0000 0---	uuuu u---
PTM1C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1DH	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1AH	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1RPL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM1RPH	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0C0	0000 0---	0000 0---	0000 0---	uuuu u---
PTM0C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0DH	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0AH	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0RPL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM0RPH	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM2C0	0000 0---	0000 0---	0000 0---	uuuu u---
PTM2C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM2DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM2DH	---- --00	---- --00	---- --00	---- --uu
PTM2AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM2AH	---- --00	---- --00	---- --00	---- --uu
PTM2RPL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM2RPH	---- --00	---- --00	---- --00	---- --uu
PTM3C0	0000 0---	0000 0---	0000 0---	uuuu u---
PTM3C1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM3DL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM3DH	---- --00	---- --00	---- --00	---- --uu
PTM3AL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM3AH	---- --00	---- --00	---- --00	---- --uu
PTM3RPL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTM3RPH	---- --00	---- --00	---- --00	---- --uu
PWMC	0000 0000	0000 0000	0000 0000	uuuu uuuu
DUTR0L	0000 0000	0000 0000	0000 0000	uuuu uuuu
DUTR0H	---- --00	---- --00	---- --00	---- --uu

寄存器	上电复位	看门狗溢出 (正常操作)	低电压复位	看门狗溢出 (暂停)
DUTR1L	0000 0000	0000 0000	0000 0000	uuuu uuuu
DUTR1H	---- --00	---- --00	---- --00	---- --uu
DUTR2L	0000 0000	0000 0000	0000 0000	uuuu uuuu
DUTR2H	---- --00	---- --00	---- --00	---- --uu
PRDRL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PRDRH	---- --00	---- --00	---- --00	---- --uu
PWMRL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWMRH	---- --00	---- --00	---- --00	---- --uu
PWMME	--00 0000	--00 0000	--00 0000	--uu uuuu
PWMMD	--00 0000	--00 0000	--00 0000	--uu uuuu
MCF	---- 0100	---- 0100	---- 0100	---- uuuu
MCD	--00 0111	--00 0111	--00 0111	--uu uuuu
DTS	0000 0000	0000 0000	0000 0000	uuuu uuuu
PLC	--00 0000	--00 0000	--00 0000	--uu uuuu
IICC0	---- 000-	---- 000-	---- 000-	---- uuu-
IICC1	1000 0001	1000 0001	1000 0001	uuuu uuuu
IICD	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
IICA	xxxx xxx-	xxxx xxx-	xxxx xxx-	uuuu uuu-
I2CTOC	0000 0000	0000 0000	0000 0000	uuuu uuuu
USR	0000 1011	0000 1011	0000 1011	uuuu uuuu
UCR1	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
UCR2	0000 0000	0000 0000	0000 0000	uuuu uuuu
BRG	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXR_RXR	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
CPC	0000 0000	0000 0000	0000 0000	uuuu uuuu
HCHK_NUM	---0 0000	---0 0000	---0 0000	---u uuuu
HNF_MSEL	---- 0000	---- 0000	---- 0000	---- uuuu
NF_VIH	00-11001	00-11001	00-11001	uu-u uuuu
NF_VIL	00-0 1010	00-0 1010	00-0 1010	UU-U UUUU
OPOMS	00-- -010	00-- -010	00-- -010	uu-- -uuu
OPCM	0000 0000	0000 0000	0000 0000	uuuu uuuu
OPACAL	-001 0000	-001 0000	-001 0000	-uuu uuuu
PAPS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPS1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCPS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCPS1	---- 0000	---- 0000	---- 0000	---- uuuu
PDPS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PRM	-000 0000	-000 0000	-000 0000	-uuu uuuu

注：“u”表示不改变
“x”表示未知
“-”表示未定义

输入 / 输出端口

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

此单片机提供 PA，PB，PC 和 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	D7	D6	D5	D4	D3	D2	D1	D0
PAC	D7	D6	D5	D4	D3	D2	D1	D0
PAPU	D7	D6	D5	D4	D3	D2	D1	D0
PAWU	D7	D6	D5	D4	D3	D2	D1	D0
PB	D7	D6	D5	D4	D3	D2	D1	D0
PBC	D7	D6	D5	D4	D3	D2	D1	D0
PBPU	D7	D6	D5	D4	D3	D2	D1	D0
PC	—	—	D5	D4	D3	D2	D1	D0
PCC	—	—	D5	D4	D3	D2	D1	D0
PCPU	—	—	D5	D4	D3	D2	D1	D0
PD	—	—	—	—	D3	D2	D1	D0
PDC	—	—	—	—	D3	D2	D1	D0
PDPU	—	—	—	—	D3	D2	D1	D0

输入 / 输出寄存器列表

上拉电阻

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

PAPU 寄存器

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

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

PBPU 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 PB 口 bit 7~bit 0 上拉电阻控制
0：除能
1：使能

PCPU 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 PC 口 bit 5~bit 0 上拉电阻控制
0：除能
1：使能

PDPU 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	D3	D2	D1	D0
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 PD 口 bit 3~bit 0 上拉电阻控制
0：除能
1：使能

PA 口唤醒

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

PAWU 寄存器

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

Bit 7~0 PA 口 bit 7~bit 0 唤醒控制
0：除能
1：使能

输入 / 输出端口控制寄存器

每一个输入 / 输出口都具有各自的控制寄存器，即 PAC~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。

PAC 寄存器

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

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

PBC 寄存器

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

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

PCC 寄存器

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

Bit 7~6 未定义，读为“0”
Bit 5~0 PC 口 bit 5~bit 0 输入 / 输出控制
0：输出
1：输入

PDC 寄存器

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

Bit 7~4 未定义，读为“0”
Bit 3~0 PD 口 bit 3~bit 0 输入 / 输出控制
0：输出
1：输入

引脚共用功能

寄存器名称	位							
	7	6	5	4	3	2	1	0
PAPS0	PA3S1	PA3S0	PA2S1	PA2S0	PA1S1	PA1S0	PA0S1	PA0S0
PAPS1	PA7S1	PA7S0	PA6S1	PA6S0	PA5S1	PA5S0	PA4S1	PA4S0
PBPS0	—	PB2S2	PB2S1	PB2S0	PB1S1	PB1S0	PB0S1	PB0S0
PBPS1	PB6S1	PB6S0	PB5S1	PB5S0	PB4S1	PB4S0	PB3S1	PB3S0
PBPS2	—	—	—	—	—	—	PB7S1	PB7S0
PCPS0	PC3S1	PC3S0	PC2S1	PC2S0	PC1S1	PC1S0	PC0S1	PC0S0
PCPS1	—	—	—	—	PC5S1	PC5S0	PC4S1	PC4S0
PDPS0	PD3S1	PD3S0	PD2S1	PD2S0	PD1S1	PD1S0	PD0S1	PD0S0

引脚共用寄存器列表

PAPS0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PA3S1	PA3S0	PA2S1	PA2S0	PA1S1	PA1S0	PA0S1	PA0S0
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 **PA3S1~PA3S0**：PA3 引脚共用设置
00：PA3/TCK1/H1
01：C1P
10：PA3/TCK1/H1
11：PA3/TCK1/H1

Bit 5~4 **PA2S1~PA2S0**：PA2 引脚共用设置
00：PA2
01：SCL
10：如果 TX 引脚重置功能使能，则为 TX
11：PA2

Bit 3~2 **PA1S1~PA1S0**：PA1 引脚共用设置
00：PA1/TCK2
01：AN3/AP
10：PA1/TCK2
11：PA1/TCK2

Bit 1~0 **PA0S1~PA0S0**：PA0 引脚共用设置
00：PA0
01：SDA
10：如果 RX 引脚重置功能使能，则为 RX
11：PA0

PAPS1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PA7S1	PA7S0	PA6S1	PA6S0	PA5S1	PA5S0	PA4S1	PA4S0
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 **PA7S1~PA7S0**：PA7 引脚共用设置
00：PA7/NFIN
01：AN6
10：PA7/NFIN
11：PA7/NFIN

Bit 5~4 **PA6S1~PA6S0**：PA6 引脚共用设置
00：PA6
01：如果 C1N 引脚重置功能使能，则为 C1N
10：AN7
11：PA6

Bit 3~2 **PA5S1~PA5S0**：PA5 引脚共用设置
00：PA5/H3
01：PA5/H3
10：C3P
11：PA5/H3

Bit 1~0 **PA4S1~PA4S0**：PA4 引脚共用设置
00：PA4/H2
01：PA4/H2
10：C2P
11：如果 C1N 引脚重置功能使能，则为 C1N

PBPS0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	PB2S2	PB2S1	PB2S0	PB1S1	PB1S0	PB0S1	PB0S0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7 未使用，读为“0”

Bit 6~4 **PB2S1~PB2S0**：PB2 引脚共用设置
000：PB2
001：HCO
010：AN0
011：TP2_1
100：如果 SDA 引脚重置功能使能，则为 SDA
101：PB2
110：PB2
111：PB2

Bit 3~2 **PB1S1~PB1S0**：PB1 引脚共用设置
00：PB1/CTIN
01：HBO
10：AN1
11：如果 SCL 引脚重置功能使能，则为 SCL

Bit 1~0 **PB0S1~PB0S0**：PB0 引脚共用设置
00：PB0/INT1
01：HAO
10：AN2
11：TP2_0

PBPS1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PB6S1	PB6S0	PB5S1	PB5S0	PB4S1	PB4S0	PB3S1	PB3S0
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 **PB6S1~PB6S0**：PB6 引脚共用设置
00：PB6
01：如果 RX 引脚重置功能除能，则为 RX
10：TP2_0 (此处需注意非引脚重置机制，可与 PB0 同时输出 TP2_0)
11：PB6

Bit 5~4 **PB5S1~PB5S0**：PB5 引脚共用设置
00：PB5
01：TP3_1
10：PB5
11：PB5

Bit 3~2 **PB4S1~PB4S0**：PB4 引脚共用设置
00：PB4
01：TP3_0
10：PB4
11：PB4

Bit 1~0 **PB3S1~PB30S0**：PB3 引脚共用设置
00：PB3
01：如果 C1N 引脚重置功能除能，则为 C1N
10：PB3
11：PB3

PBPS2 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PB7S1	PB7S0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 未使用，读为“0”

Bit 1~0 **PB7S1~PB7S0**：PB7 引脚共用设置
00：PB7
01：如果 TX 引脚重置功能除能，则为 TX
10：TP2_1(此处需注意非引脚重置机制，可与 PB2 同时输出 TP2_1)
11：PB7

PCPS0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PC3S1	PC3S0	PC2S1	PC2S0	PC1S1	PC1S0	PC0S1	PC0S0
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 **PC3S1~PC3S0**：PC3 引脚共用设置
00：PC3
01：PC3
10：GBB
11：PC3

Bit 5~4 **PC2S1~PC2S0**：PC2 引脚共用设置
00：PC2
01：PC2
10：GBT
11：PC2

Bit 3~2 **PC1S1~PC1S0**：PC1 引脚共用设置
00：PC1
01：PC1
10：GAB
11：PC1

Bit 1~0 **PC0S1~PC0S0**：PC0 引脚共用设置
00：PC0
01：PC0
10：GAT
11：PC0

PCPS1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	PC5S1	PC5S0	PC4S1	PC4S0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4 未定义，读为“0”

Bit 3~2 **PC5S1~PC5S0**：PC5 引脚共用设置
00：PC5
01：PC5
10：GCB
11：PC5

Bit 1~0 **PC4S1~PC4S0**：PC4 引脚共用设置
00：PC4
01：PC4
10：GCT
11：PC4

PDPS0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PD3S1	PD3S0	PD2S1	PD2S0	PD1S1	PD1S0	PD0S1	PD0S0
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 **PD3S1~PD3S0**：PD3 引脚共用设置
00：PD3
01：TP1_1
10：PD3
11：PD3

Bit 5~4 **PD2S1~PD2S0**：PD2 引脚共用设置
00：PD2/INT1，如果引脚重置功能使能
01：TP1_0
10：PD2
11：PD2

Bit 3~2 **PD1S1~PD1S0**：PD1 引脚共用设置
00：PD1
01：AN4
10：TP0_1
11：PD1

Bit 1~0 **PD0S1~PD0S0**：PD0 引脚共用设置
00：PD0/TCK3/TCK0
01：AN5
10：TP0_0
11：PD0/TCK3/TCK0

引脚重置功能

引脚的多功能可以增加单片机应用的灵活性。有限的引脚个数将会限制设计者，而引脚的多功能将会解决很多此类问题。每个功能可单独选择所在的引脚，以及一个确定的优先级，使得引脚上多种功能可以同时使用。此外，一些引脚功能可以通过寄存器 PRM 进行设定。

引脚重置寄存器

封装中有限的引脚个数会对某些单片机功能造成影响。然而，引脚功能重置和引脚功能选择，使得小封装单片机具有更多不同的功能。一些单片机的特定引脚功能可以通过寄存器 PRM 进行设定。

PRM 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	INT1PS	RXPS	TXPS	C1NPS1	C1NPS0	SDAPS	SCLPS
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 **INT1PS**：INT1 引脚重置控制位
0：INT1on PB0
1：INT1 on PD2

Bit 5 **RXPS**：RX 引脚重置控制位
0：RX on PB6
1：RX on PA0

Bit 4 **TXPS**：TX 引脚重置控制位
0：TX on PB7
1：TX on PA2

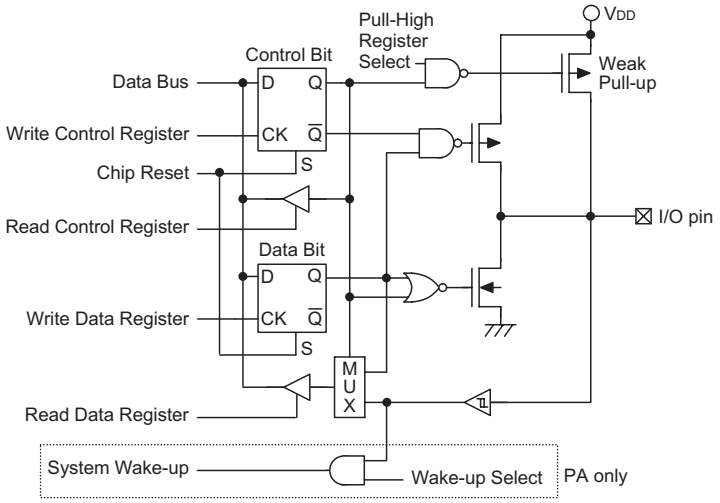
Bit 3~2 **C1NPS1~C1NPS0**：C1N 引脚重置控制位
00：C1N on PB3
01：C1N on PA6
10：C1N on PA4
11：保留位

Bit 1 **SDAPS**：SDA 引脚重置控制位
0：SDA on PA0
1：SDA on PB2

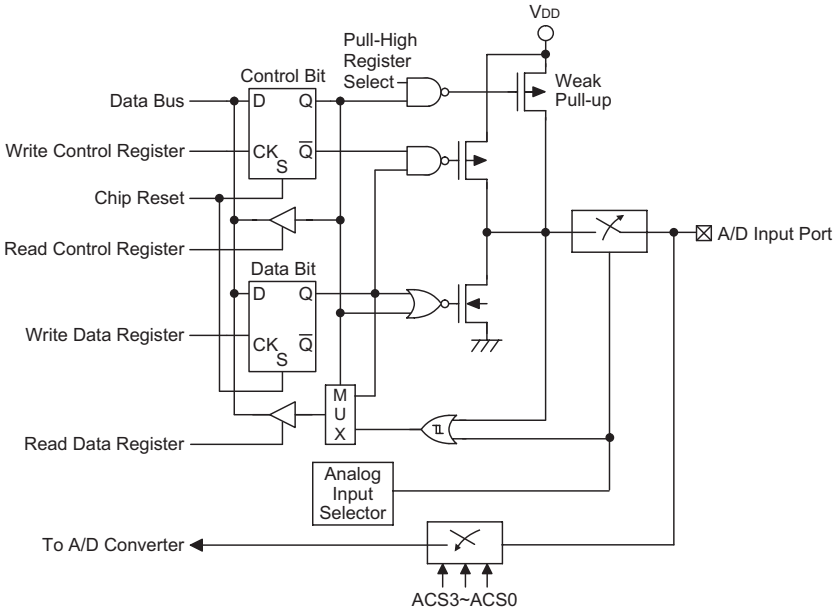
Bit 0 **SCLPS**：SCL 引脚重置控制位
0：SCL on PA2
1：SCL on PB1

输入 / 输出引脚结构

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



通用输入 / 输出端口



A/D 输入 / 输出端口

编程注意事项

在编程中，最先要考虑的是端口的初始化。复位之后，所有的输入 / 输出数据及端口控制寄存器都将被设为逻辑高。所有输入 / 输出引脚默认为输入状态，而其电平则取决于其它相连接电路以及是否选择了上拉电阻。如果端口控制寄存器 PAC~PDC，某些引脚位元被设定输出状态，这些输出引脚会有初始高电平输出，除非数据寄存器端口 PA~PD 在程序中被预先设定。设置哪些引脚是输入及哪些引脚是输出，可通过设置正确的值到适当的端口控制寄存器，或使用

指令“SET [m].i”及“CLR [m].i”来设定端口控制寄存器中个别的位。注意，当使用这些位控制指令时，系统即将产生一个读 - 修改 - 写的操作。单片机需要先读入整个端口上的数据，修改个别的位，然后重新把这些数据写入到输出端口。

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

定时器模块 – TM

控制和测量时间在任何单片机中都是一个很重要的部分。每个单片机提供几个定时器模块 (简称 TM)，来实现和时间有关的功能。定时器模块是包括多种操作的定时单元，提供的操作有：定时 / 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周期型定时器章节。

简介

该单片机包含 4 个 TM，其中 2 个 16 位 PTM、2 个 10 位 PTM 分别命名为 TM0、TM1、TM2 和 TM3，其特性见下表。

功能	PTM
定时 / 计数器	√
捕捉输入	√
比较匹配输出	√
PWM 通道数	2
单脉冲输出	√
PWM 对齐方式	边沿对齐
PWM 调节周期 & 占空比	占空比或周期

TM 功能概要

TM0	TM1	TM2	TM3
16-bit PTM	16-bit PTM	10-bit PTM	10-bit PTM

TM 名称 / 类型参考

TM 操作

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

TM 时钟源

驱动 TM 计数器的时钟源很多。通过设置 TM 控制寄存器的 TnCK2~TnCK0 位，选择所需的时钟源。该时钟源来自系统时钟 f_{SYS} 或内部高速时钟 f_H 或 f_{TBC} 时钟源或外部 TCKn 引脚时钟的分频比。TCKn 引脚时钟源用于允许外部信号作为 TM 时钟源或用于事件计数。TCKn 引脚也可作为捕捉输入引脚使用。

TM 中断

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

TM 外部引脚

无论哪种类型的 TM，都有一个 TM 输入引脚 TCKn。通过设置 PTMnC0 寄存器中的 PTnCK2~PTnCK0 位，选择 TM 功能并将该引脚作为 TM 时钟源输入脚。外部时钟源可通过该引脚来驱动内部 TM。外部 TM 输入脚也与其它功能共用，但是，如果设置适当值给 PTnCK2~PTnCK0，该引脚会连接到内部 TM。TM 引脚可选择上升沿有效或下降沿有效。

每个 TM 有两个输出引脚 TPn。当 TM 工作在比较匹配输出模式且比较匹配发生时，这些引脚会由 TM 控制切换到高电平或低电平或翻转。外部 TPn 输出引脚也被 TM 用来产生 PWM 输出波形。当 TM 输出引脚与其它功能共用时，TM 输出功能需要通过寄存器先被设置。寄存器中的一个单独位用于决定其相关引脚用于外部 TM 输出还是用于其它功能。每个单片机和不同类型 TM 中输出引脚的个数是不同的，详见下表。

所有 TM 输出引脚名称都带有“_n”后缀。引脚名称带“_0”或“_1”后缀表示来自多输出引脚的 TM。这允许 TM 产生一对互补输出，可通过 I/O 寄存器数据位选择。

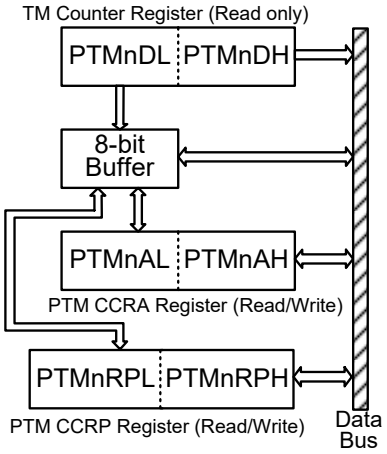
TM0	TM1	TM2	TM3
TCK0； TP0_0，TP0_1	TCK1； TP1_0，TP1_1	TCK2； TP2_0，TP2_1	TCK3； TP3_0，TP3_1

TM 外部引脚

编程注意事项

TM 计数寄存器和捕捉 / 比较寄存器 CCRA 和 CCRP 为 10-bit 或 16-bit 的寄存器，含有低字节和高字节结构。高字节可直接访问，低字节则仅能通过一个内部 8-bit 的缓存器进行访问。读写这些成对的寄存器需通过特殊的方式。值得注意的是 8-bit 缓存器的存取数据及相关低字节的读写操作仅在其相应的高字节读取操作执行时发生。

CCRA 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 和 CCRP 低字节寄存器，PTMnAL 和 PTMnPRL 否则可能导致无法预期的结果。



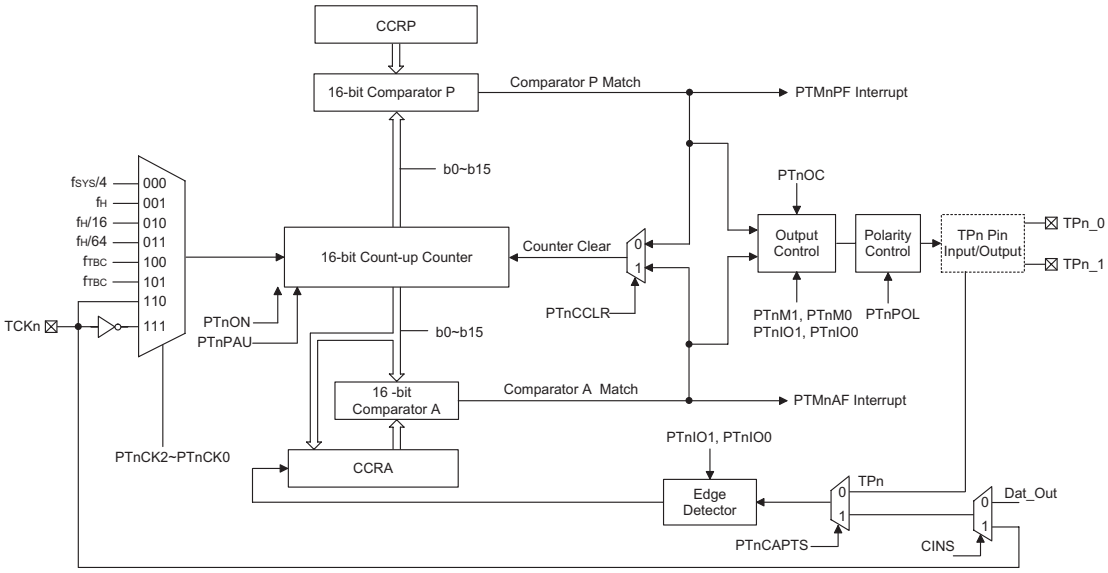
读写流程如下步骤所示：

- 写数据至 CCRA 或 CCRP
  - ♦ 步骤 1. 写数据至低字节寄存器 PTMnAL
    – 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 PTMnAH，PTMnRPH
    – 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 CCRP 中读取数据
  - ♦ 步骤 1. 由高字节寄存器 PTMnDH，PTMnAH，PTMnRPH 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 PTMnDL、PTMnAL 或 PTMnRPL 读取数据
    – 注意，此时读取 8-bit 缓存器中的数据。

周期型 TM – PTM

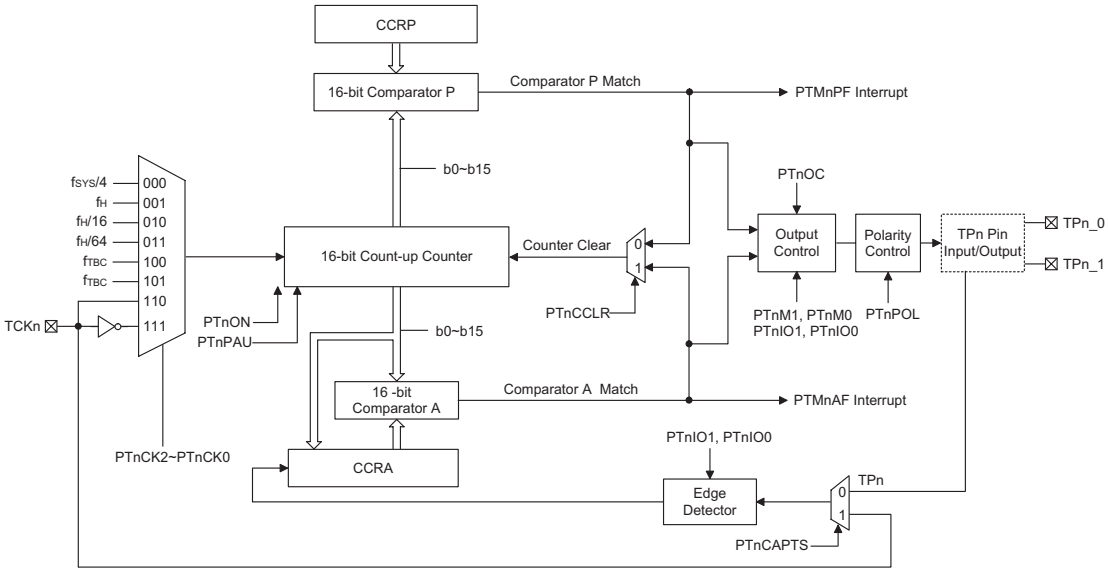
周期型 TM 包括 5 种工作模式，即比较匹配输出、定时 / 事件计数器、捕捉输入、单脉冲输出和 PWM 输出模式。周期型 TM 也由一个外部输入脚控制并驱动两个外部输出脚。

名称	TM 编号	TM 输入引脚	TM 输出引脚
16-bit PTM	0，1	TCK0，TCK1	TP0_0，TP0_1 TP1_0，TP1_1
10-bit PTM	2，3	TCK2，TCK3	TP2_0，TP2_1 TP3_0，TP3_1

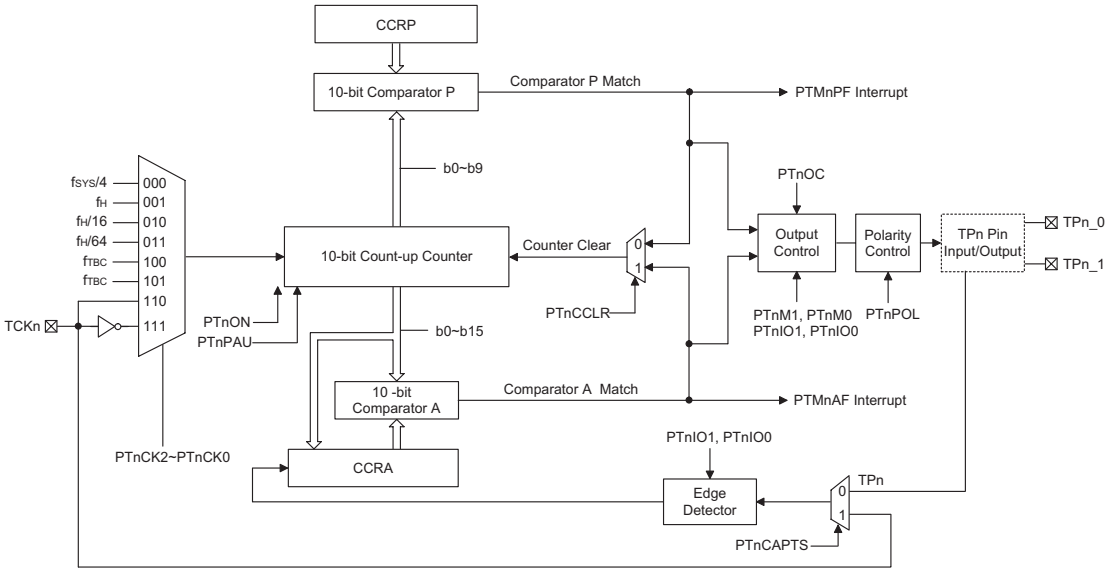


周期型 TM 框图 (n=0)

16 位 PTM(TM0) 可捕捉 Noise Filter Dat_Out 信号上下沿之间的时间。



周期型 TM 框图 (n=1)



周期型 TM 框图 (n=2, 3)

周期型 TM 操作

周期型 TM 的核心是一个由用户选择的内部或外部时钟源驱动的 10 位或 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 10 位或 16 位宽度。

通过应用程序改变 10 位或 16 位计数器值的唯一方法是使 PTnON 位发生上升沿跳变清除计数器。此外，计数器溢出或比较匹配也会自动清除计数器。上述条件发生时，通常情况会产生 TM 中断信号。周期型 TM 可工作在不同的模式，可由包括来自输入脚的不同时钟源驱动，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 的所有工作模式由一系列寄存器控制。一对只读寄存器用来存放 10 位或 16 位计数器的值，两对读 / 写寄存器存放 10 位或 16 位的 CCRA 和 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
PTMnC0	PTnPAU	PTnCK2	PTnCK1	PTnCK0	PTnON	—	—	—
PTMnC1	PTnM1	PTnM0	PTnIO1	PTnIO0	PTnOC	PTnPOL	PTnCAPTS	PTnCCLR
PTMnDL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnDH	D15	D14	D13	D12	D11	D10	D9	D8
PTMnAL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnAH	D15	D14	D13	D12	D11	D10	D9	D8
PTMnRPL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnRPH	D15	D14	D13	D12	D11	D10	D9	D8

16-bit 周期型 TM 寄存器列表 (n=0, 1)

寄存器名称	位							
	7	6	5	4	3	2	1	0
PTMnC0	PTnPAU	PTnCK2	PTnCK1	PTnCK0	PTnON	—	—	—
PTMnC1	PTnM1	PTnM0	PTnIO1	PTnIO0	PTnOC	PTnPOL	PTnCAPTS	PTnCCLR
PTMnDL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnDH	—	—	—	—	—	—	D9	D8
PTMnAL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnAH	—	—	—	—	—	—	D9	D8
PTMnRPL	D7	D6	D5	D4	D3	D2	D1	D0
PTMnRPH	—	—	—	—	—	—	D9	D8

10-bit 周期型 TM 寄存器列表 (n=2, 3)

PTMnC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PTnPAU	PTnCK2	PTnCK1	PTnCK0	PTnON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

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

Bit 6 ~ 4 **PTnCK2 ~ PTnCK0**：选择 TMn 计数时钟位
000：$f_{SYS}/4$
001：f_{SYS}
010：$f_H/16$
011：$f_H/64$
100：f_{TBC}
101：f_{TBC}
110：TCKn 上升沿时钟
111：TCKn 下降沿时钟
此三位用于选择 TM 的时钟源。外部引脚时钟源能被选择在上升沿或下降沿有效。f_{SYS} 是系统时钟，f_H 和 f_{TBC} 是其它的内部时钟源，细节方面请参考振荡器章节。

Bit 3 **PTnON**：TMn 计数器 On/Off 控制
0：Off
1：On
此位控制 TM 的总开关功能。设置此位为高则使能计数器使其运行，清零此位则除能 TM。清零此位将停止计数器并关闭 TM 减少耗电。当此位经由低到高转换时，内部计数器将复位清零；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 TM 处于比较匹配输出模式时 (通过 PTnOC 位指定)，当 PTnON 位经由低到高的转换时，TM 输出脚将重置其初始值。

Bit 2 ~ 0 未定义，读为“0”

PTMnC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PTnM1	PTnM0	PTnIO1	PTnIO0	PTnOC	PTnPOL	PTnCAPTS	PTnCCLR
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 **PTnM1~PTnM0**：选择 TMn 工作模式
00：比较匹配输出模式
01：捕捉输入模式
10：PWM 模式或单脉冲输出模式
11：定时 / 计数器模式
这两位设置 PTM 需要的工作模式。为了确保操作可靠，TM 应在 PTnM1 和 PTnM0 位有任何改变前先关掉。在定时 / 计数器模式，TM 输出脚控制必须除能。

Bit 5 ~ 4 **PTnIO1~PTnIO0**：选择 TPn_0 和 TPn_1 输出功能位
比较匹配输出模式
00：无变化
01：输出低
10：输出高
11：输出翻转
PWM 模式 / 单脉冲输出模式
00：强制无效状态
01：强制有效状态
10：PWM 输出
11：单脉冲输出
捕捉输入模式
00：在 TM 捕捉输入引脚上升沿输入捕捉
01：在 TM 捕捉输入引脚下降沿输入捕捉
10：在 TM 捕捉输入引脚双沿输入捕捉
11：输入捕捉除能
定时 / 计数器模式
未使用
此两位用于决定在一定条件达到时 TM 输出脚如何改变状态。这两位值的选择决定 TM 运行在哪种模式下。
在比较匹配输出模式下，PTnIO1 和 PTnIO0 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 PTMnC1 寄存器的 PTnOC 位设置取得。注意，由 PTnIO1 和 PTnIO0 位得到的输出电平必须与通过 PTnOC 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 PTnON 位由低到高电平的转换复位至初始值。
在 PWM 模式，PTnIO1 和 PTnIO0 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 PTMn 关闭时改变 PTnIO1 和 PTnIO0 位的值是很有必要的。若在 PTM 运行时改变 PTnIO1 和 PTnIO0 的值，PWM 输出的值是无法预料的。
在捕捉输入模式，TM1，TM2 和 TM3 的捕捉输入源由 PTMnC1 寄存器的 PTnCAPTS 位决定。然而，TM0 的捕捉输入源由 NF_VIH 寄存器的 CINS 位和 PTM0C1 寄存器的 PT0CAPTS 位共同决定。

Bit 3 **PTnOC**：TPn_0 和 TPn_1 输出控制位
比较匹配输出模式
0：初始低
1：初始高
PWM 模式 / 单脉冲输出模式
0：低有效
1：高有效

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

Bit 2 **PTnPOL**：TPn_0 和 TPn_1 输出极性控制位
0：同相
1：反相
此位控制 TPn_0 和 TPn_1 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 **PTnCAPTS**：TMn 捕捉输入触发源选择位
0：来自 TPn_0 或 TPn _1 引脚
1：来自 TCKn 引脚
注：对于 TM0，如下：
PT0CAPS=0 选 TP0_0(PD0)
PT0CAPS=1 选 TCK0，则还需经过 CINS 决定，如下：
PT0CAPS=1 的情况下 CINS=0，触发源 = 维持原路径
PT0CAPS=1 的情况下 CINS=1，触发源 = Noise Filter Dat_Out

Bit 0 **PTnCCLR**：选择 TMn 计数器清零条件位
0：TMn 比较器 P 匹配
1：TMn 比较器 A 匹配
此位用于选择清除计数器的方法。周期型 TM 包括两个比较器 - 比较器 A 和比较器 P。这两个比较器每个都可以用作清除内部计数器。PTnCCLR 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。PTnCCLR 位在 PWM，单脉冲或输入捕捉模式时未使用。

PTMnDL 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 **PTMnDL**：PTMn 计数器低字节寄存器 bit 7 ~ bit 0
PTMn 10-bit 计数器 bit 7 ~ bit 0

PTMnDH 寄存器 (n=0, 1)

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 **PTMnDH**：PTMn 计数器高字节寄存器 bit 7 ~ bit 0
PTMn 16-bit 计数器 bit 15 ~ bit 8

PTMnDH 寄存器 (n=2, 3)

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　**PTMnDH**：PTMn 计数器高字节寄存器 bit 1 ~ bit 0
PTMn 10-bit 计数器 bit 9 ~ bit 8

PTMnAL 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　**PTMnAL**：PTMn CCRA 低字节寄存器 bit 7 ~ bit 0
PTMn 10-bit CCRA bit 7 ~ bit 0

PTMnAH 寄存器 (n=0, 1)

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　**PTMnAH**：PTMn CCRA 高字节寄存器 bit 7 ~ bit 0
TMn 16-bit CCRA bit 15 ~ bit 8

PTMnAH 寄存器 (n=2, 3)

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　**PTMnAH**：PTMn CCRA 高字节寄存器
TMn 10-bit CCRA bit 9 ~ bit 8

PTMnRPL 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　**PTMnRPL**：TMn CCRP 低字节寄存器 bit 7~bit 0
TMn 10-bit CCRP bit 7~bit 0

PTMnRPH 寄存器 (n=0, 1)

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 **PTMnRPH**：PTMn CCRP 高字节寄存器 bit 7~bit 0
PTMn 16-bit CCRP bit 15~bit 8

PTMnRPH 寄存器 (n=2, 3)

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 **PTMnRPH**：TMn CCRP 高字节寄存器 bit 1~bit 0
TMn 10-bit CCRP bit 9~bit 8

周期型 TM 工作模式

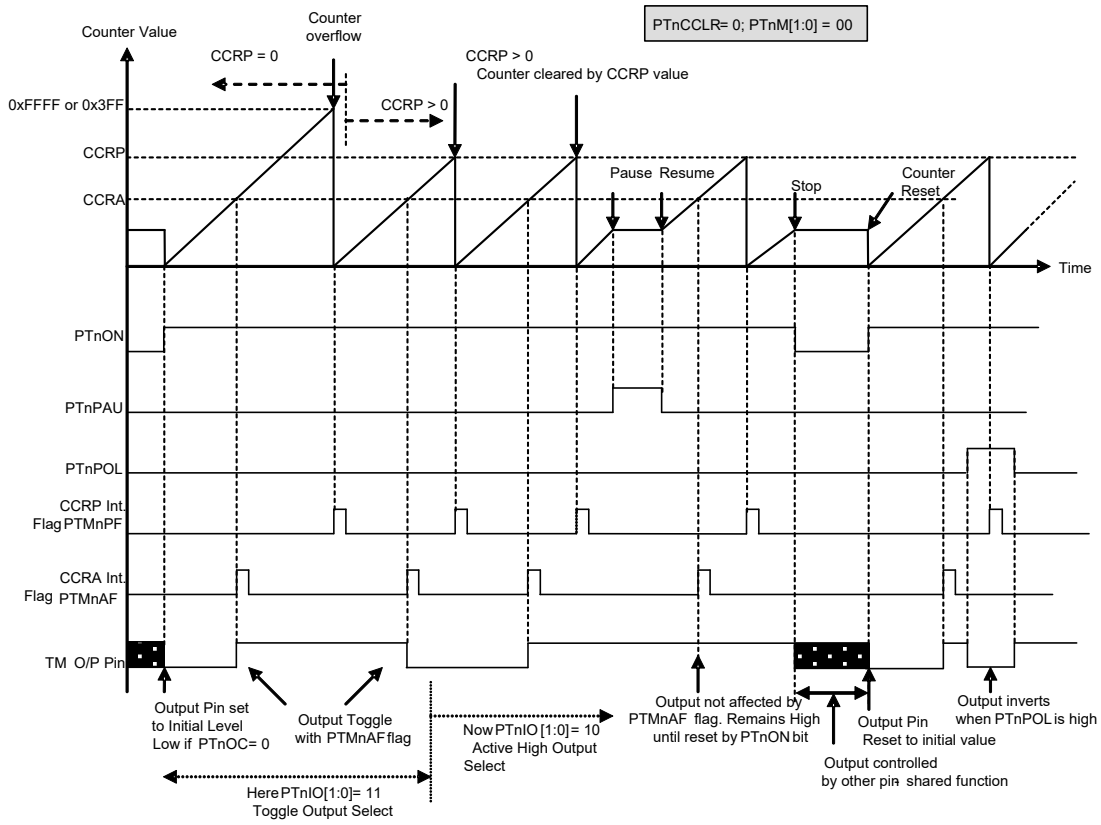
周期型 TM 有五种工作模式，即比较匹配输出模式、PWM 输出模式、单脉冲输出模式、捕捉输入模式或定时 / 计数器模式。通过设置 PTMnC1 寄存器的 PTnM1 和 PTnM0 位选择任意模式。

比较匹配输出模式

为使 TM 工作在此模式，PTMnC1 寄存器中的 PTnM1 和 PTnM0 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 PTnCCLR 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 PTMnAF 和 PTMnPF 将分别置位。

如果 PTMnC1 寄存器的 PTnCCLR 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 PTMnAF 中断请求标志。所以当 PTnCCLR 为高时，不会产生 PTMnPF 中断请求标志。在比较匹配输出模式下，CCRA 不能设为“n”。

正如该模式名所言，当比较匹配发生后，TM 输出脚状态改变。当比较器 A 比较匹配发生后 PTMnAF 标志产生时，TM 输出脚状态改变。比较器 P 比较匹配发生时产生的 PTMnPF 标志不影响 TM 输出脚。TM 输出脚状态改变方式由 PTMnC1 寄存器中 PTnIO1 和 PTnIO0 位决定。当比较器 A 比较匹配发生时，PTnIO1 和 PTnIO0 位决定 TM 输出脚输出高，低或翻转当前状态。TM 输出脚初始值，既可以通过 PTnON 位由低到高电平的变化设置，也可以由 PTnOC 位设置。注意，若 PTnIO1 和 PTnIO0 位同时为 0 时，引脚输出不变。



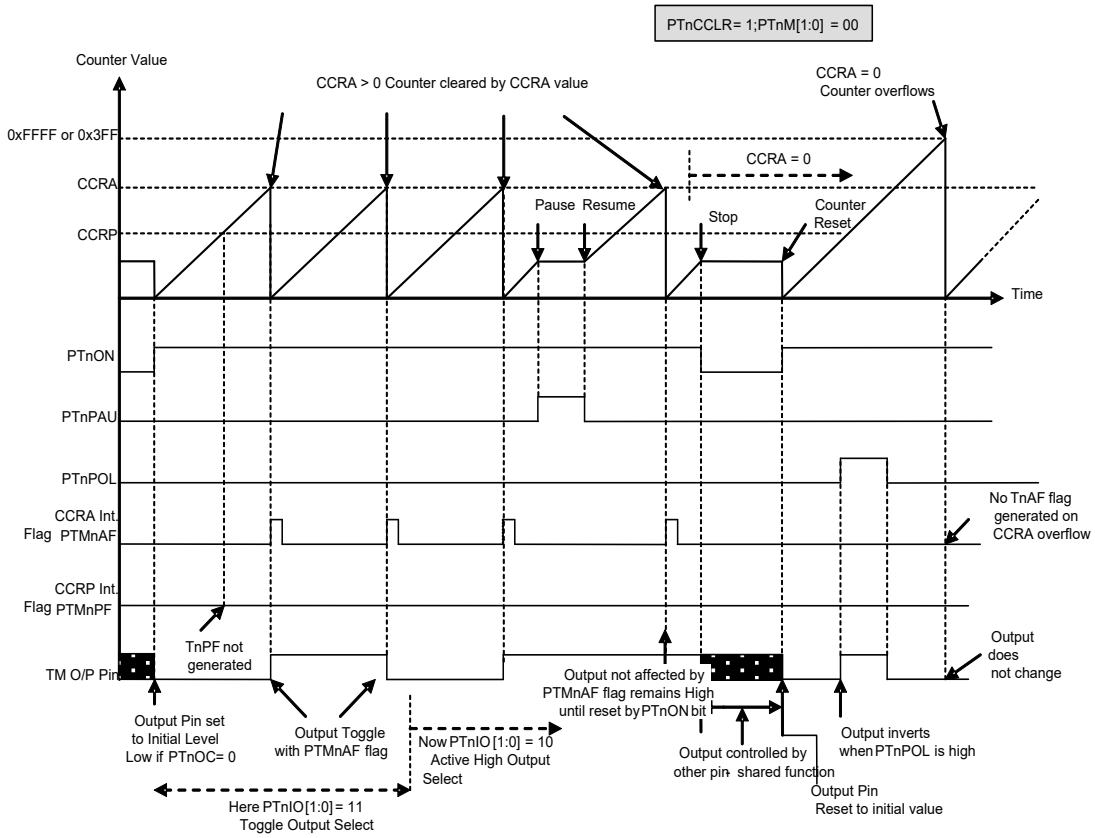
比较匹配输出模式 – PTnCCLR=0

注：1. PTnCCLR=0，比较器 P 匹配将清除计数器

2. TM 输出脚仅由 PTMnAF 标志位控制

3. 在 PTnON 上升沿 TM 输出脚复位至初始值

4. n=0~3



比较匹配输出模式 – PTnCCLR=1

注：1. PTnCCLR=1，比较器 A 匹配将清除计数器
2. TM 输出脚仅由 PTMnAF 标志位控制
3. 在 PTnON 上升沿 TM 输出脚复位至初始值
4. 当 PTnCCLR=1 时，不会产生 PTMnPF 标志
5. n=0~3

定时 / 计数器模式

为使 TM 工作在此模式，PTMnC1 寄存器中的 PTnM1 和 PTnM0 位需要设置为“11”。定时 / 计数器模式与比较输出模式操作方式相同，并产生同样的中断请求标志。不同的是，在定时 / 计数器模式下 TM 输出脚未使用。因此，比较匹配输出模式中的描述和时序图可以适用于此功能。该模式中未使用的 TM 输出脚用作普通 I/O 脚或其它功能。

PWM 输出模式

为使 TM 工作在此模式，PTMnC1 寄存器中的 PTnM1 和 PTnM0 位需要设置为“10”，且 PTnIO1 和 PTnIO0 位也需要设置为“10”。TM 的 PWM 功能在马达控制、加热控制、照明控制等方面十分有用。给 TM 输出脚提供一个频率固定但占空比可调的信号，将产生一个有效值等于 DC 均方根的 AC 方波。

由于 PWM 波形的周期和占空比可调，其波形的选择就较为灵活。在 PWM 模式中，PTnCCLR 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。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。PTMnC1 寄存器中的 PTnOC 位决定 PWM 波形的极性，PTnIO1 和 PTnIO0 位使能 PWM 输出或将 TM 输出脚置为逻辑高或逻辑低。PTnPOL 位对 PWM 输出波形的极性取反。

- **10-bit PTM，PWM 模式，边沿对齐模式**

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

若 f_{SYS}=8MHz，TM 时钟源选择 f_{SYS}/4，CCRP=512，CCRA=128，

PTM PWM 输出频率 =(f_{SYS}/4)/512= f_{SYS}/2048=3.90625kHz，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%。

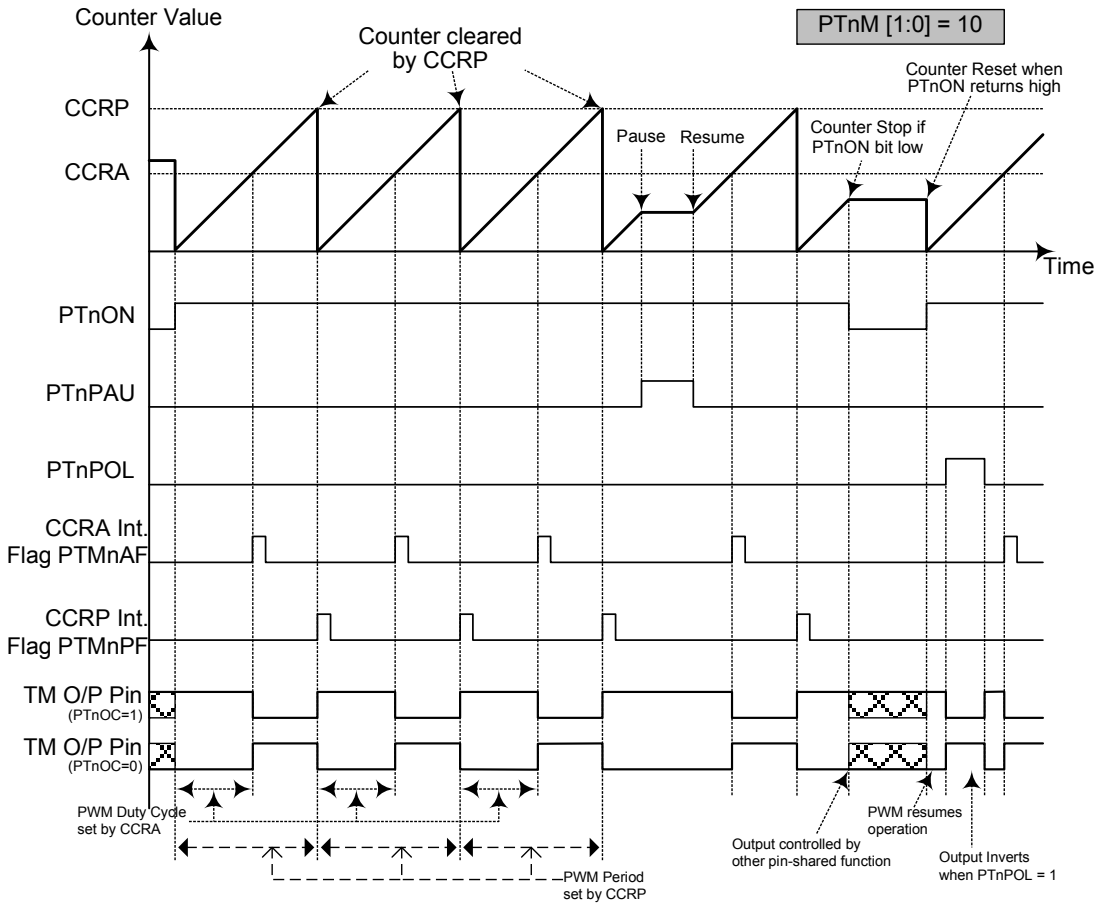
- **16-bit PTM，PWM 模式，边沿对齐模式**

CCRP	1~65535	0
Period	1~65535 clocks	65536 clocks
Duty	CCRA	

若 f_{SYS}=8MHz，TM 时钟源选择 f_{SYS}/4，CCRP=512，CCRA=128，

PTM PWM 输出频率 =(f_{SYS}/4)/512= f_{SYS}/2048=3.90625kHz，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 模式

注：1. CCRP 清除计数器

2. 计数器清零并设置 PWM 周期

3. 当 PTnIO1，PTnIO0=00 或 01，PWM 功能不变

4. PTnCCLR 位不影响 PWM 操作

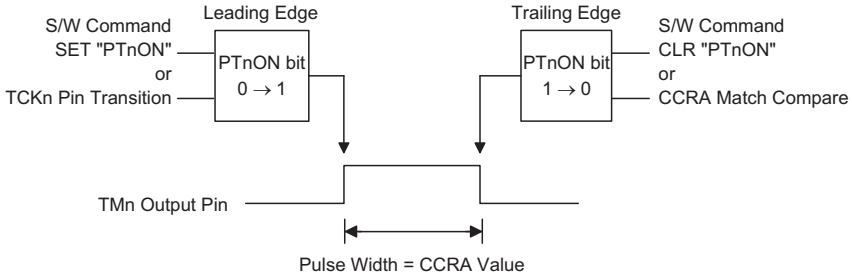
5. n=0~3

单脉冲模式

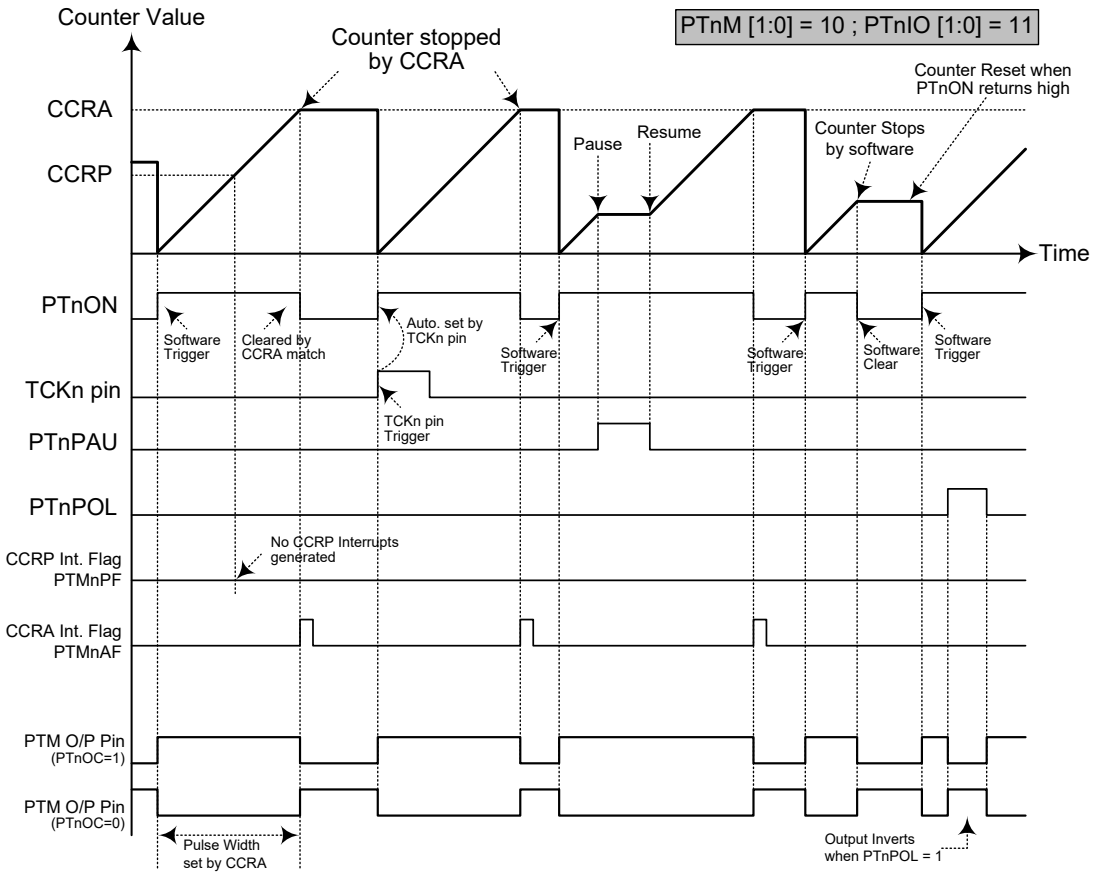
为使 TM 工作在此模式，PTMnC1 寄存器中的 PTnM1 和 PTnM0 位需要设置为“10”，同时 PTnIO1 和 PTnIO0 位需要设置为“11”。正如模式名所言，单脉冲输出模式，在 TM 输出脚将产生一个脉冲输出。

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

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



单脉冲产生示意图 (n=0~3)



单脉冲模式

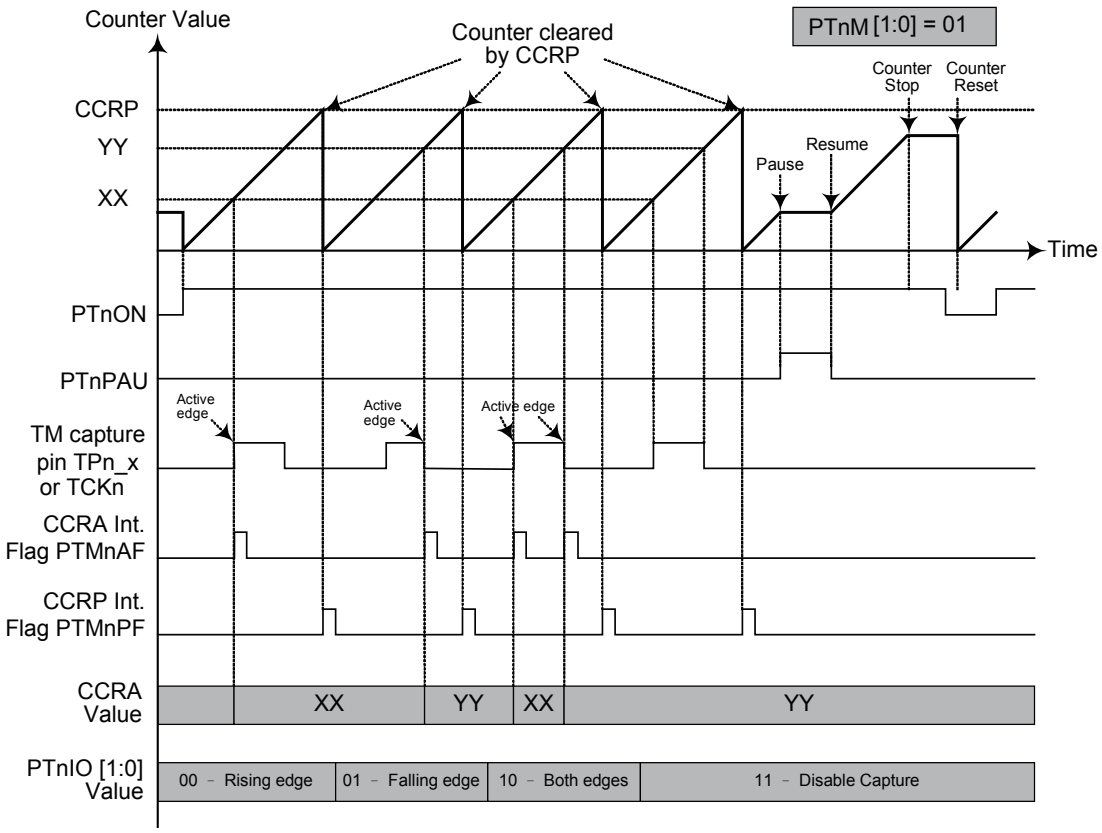
注：1. 通过 CCRA 匹配停止计数器
2. CCRP 未使用
3. 通过 TCKn 脚或设置 PTnON 位为高来触发脉冲
4. TCKn 脚有效沿会自动置位 PTnON
5. 单脉冲模式中，PTnIO[1:0] 需置位“11”，且不能更改。
6. n=0~3

捕捉输入模式

为使 TM 工作在此模式，PTMnC1 寄存器中的 PTnM1 和 PTnM0 位需要设置为“01”。此模式使能外部信号捕捉并保存内部计数器当前值，因此被用于诸如脉冲宽度测量的应用中。TPn_1、TPn_0 或 TCKn 脚上的外部信号，通过设置 PTMnC0 寄存器的 PTnCAPTS 位选择。可通过设置 PTMnC1 寄存器的 PTnIO1 和 PTnIO0 位选择有效边沿类型，即上升沿，下降沿或双沿有效。计数器在 PTnON 位由低到高转变时启动并通过应用程序初始化。

当 TPn_1、TPn_0 或 TCKn 引脚出现有效边沿转换时，计数器当前值被锁存到 CCRA 寄存器，并产生 TM 中断。不考虑 TPn_1、TPn_0 或 TCKn 引脚事件，计数器继续工作直到 PTnON 位发生下降沿跳变。当 CCRP 比较匹配发生时计数器复位至零；CCRP 的值通过这种方式控制计数器的最大值。当比较器 P CCRP 比较匹配发生时，也会产生 TM 中断。记录 CCRP 溢出中断信号的值可以测量脉宽。通过设置 PTnIO1 和 PTnIO0 位选择 TPn_1、TPn_0 或 TCKn 引脚为上升沿、下降沿或双沿有效。不考虑 TPn_1、TPn_0 或 TCKn 引脚引脚事件，如果 PTnIO1 和 PTnIO0 位设置为高，不会产生捕捉操作，但计数器继续运行。

当 TPn_1、TPn_0 或 TCKn 引脚与其它功能共用，TM 工作在输入捕捉模式时需多加注意。这是因为如果引脚被设为输出，那么该引脚上的任何电平转变都可能执行输入捕捉操作。PTnCCLR、PTnOC 和 PTnPOL 位在此模式中未使用。



捕捉输入模式

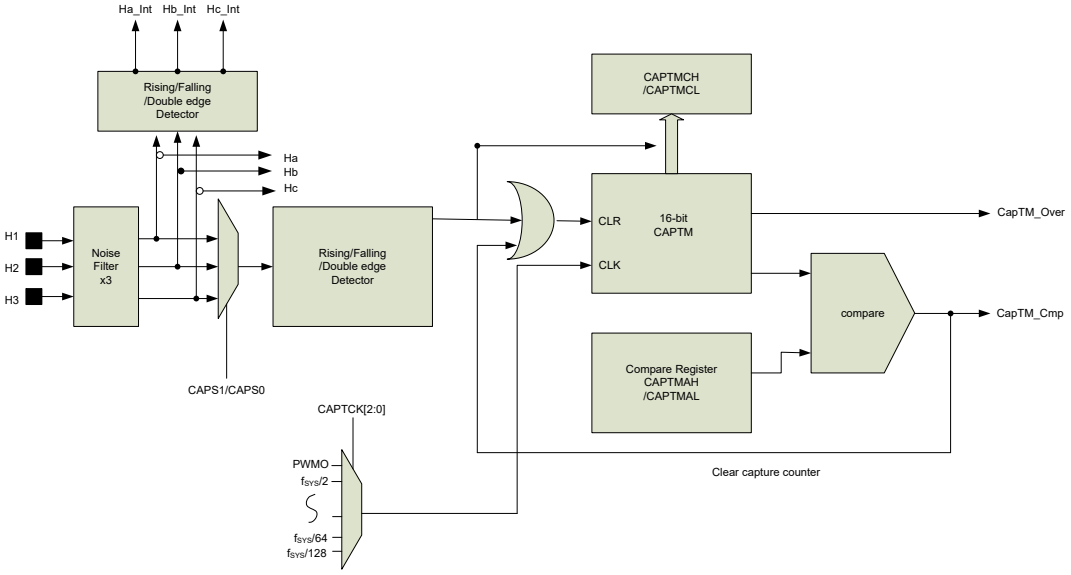
注：1. PTnM1，PTnM0=01 并通过 PTnIO1 和 PTnIO0 位设置有效边沿
2. TM 捕捉输入脚的有效边沿将计数器的值转移到 CCRA 中
3. PTnCCLR 位未使用
4. 无输出功能 -- PTnOC 和 PTnPOL 位未使用
5. 计数器值由 CCRP 决定，在 CCRP 为“0”时，计数器计数值可达最大
6. n=0~3

捕捉定时器模块 – CAPTM

捕捉定时器模块是一个专门用于马达控制的定时单元。CAPTM 由程序选择的时钟源和来自马达定位霍尔传感器的三个中断源控制。

捕捉定时器简介

捕捉定时器核心是一个由用户选择的内部时钟源驱动的 16 位向上计数器，时钟源可来自系统时钟或 PWM。它还包括一个内部比较器用于将计数器的值与存储在两个寄存器中的值进行比较。它有两个基本工作模式，即比较模式和捕捉模式，每一个都可以用来复位内部计数器。当比较匹配情况到达一个信号时，内部计数器将复位。当由三个外部中断源 H1，H2 和 H3 触发捕捉时，计数器也将被清除。



捕捉定时器方框图

捕捉定时器寄存器介绍

捕捉定时器的所有操作由 8 个寄存器控制。每组中包含一对只读寄存器用来存放 16 位计数器的值，一对读 / 写寄存器存放 16 位比较值，另外一组只读寄存器用来存放捕捉值，剩下两个控制寄存器设置不同的操作和控制模式。

寄存器名称	位							
	7	6	5	4	3	2	1	0
CAPTC0	CAPTPAU	CAPTCK2	CAPTCK1	CAPTCK0	CAPTON	—	CAPS1	CAPS0
CAPTC1	CAPEG1	CAPEG0	CAPEN	CAPNFT	CAPNFS	CAPFIL	CAPCLR	CAMCLR
CAPTMDL	D7	D6	D5	D4	D3	D2	D1	D0
CAPTMDH	D15	D14	D13	D12	D11	D10	D9	D8
CAPTMAL	D7	D6	D5	D4	D3	D2	D1	D0
CAPTMAH	D15	D14	D13	D12	D11	D10	D9	D8
CAPTMCL	D7	D6	D5	D4	D3	D2	D1	D0
CAPTMCH	D15	D14	D13	D12	D11	D10	D9	D8

捕捉定时器寄存器列表

CAPTC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	CAPTPAU	CAPTCK2	CAPTCK1	CAPTCK0	CAPTON	—	CAPS1	CAPS0
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 **CAPTPAU**：CAPTM 计数器暂停控制位
0：运行
1：暂停
将该位设置为高可以暂停计数器，清为零可以将计数器重启操作。当处于暂停状态时，CAPTM 将保持上电且继续消耗功耗。当该位从低到高的转变时，计数器将会保持原来的值；当该位又从高到低转变时，则计数器将从原来的值重新开始计数。

Bit 6~4 **CAPTCK2~CAPTCK0**：选择 CAPTM 计数器时钟
000：PWMO
001：$f_H/2$
010：$f_H/4$
011：$f_H/8$
100：$f_H/16$
101：$f_H/32$
110：$f_H/64$
111：$f_H/128$
这三位用于选择 CAPTM 时钟源 。时钟源 fH 为内部高速振荡器 -HIRC。

Bit 3 **CAPTON**：CAPTM 计数器开 / 关控制位
0：停止
1：开启
该位为 CAPTM 总的开 / 关控制位。将该位设为高，将使能计数器，清为零将除能 CAPTM。将该位清为零将停止计数器计数并关闭 CAPTM 以节省功耗。当该位从低到高的转变时，内部计数器值将复位到零；当该位从高到低时，内部计数器将保持原来的值。

Bit 2 未定义，读为“0”

Bit 1~0 **CAPS1~CAPS0**：捕捉源选择位
00：H1
01：H2
10：H3
11：CTIN

CAPTC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	CAPEG1	CAPEG0	CAPEN	CAPNFT	CAPNFS	CAPFIL	CAPCLR	CAMCLR
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 **CAPEG1~CAPEG0**：CAPTM 捕捉有效边沿选择位
00：除能 CAPTM 捕捉
01：上升沿捕捉
10：下降沿捕捉
11：双边沿捕捉

Bit 5 **CAPEN**：CAPTM 捕捉输入控制
0：除能
1：使能
此位用于控制 CAPTM 捕捉输入源的使能或除能

Bit 4 **CAPNFT**：CAPTM 噪声滤波取样次数
0：2 次
1：4 次
CAPTM 噪声滤波电路需连续取样 CAPTM 输入 2 次或 4 次都相同时，信号才会被承认，取样的时间由 CAPNFS 决定。

Bit 3 **CAPNFS**：捕捉 CAPTM 杂讯噪声滤波时钟源选择位
0：t_{SYS}
1：$4t_{SYS}$
CAPTM 噪声滤波器的时钟来自 f_{SYS} 或 $f_{SYS}/4$。

Bit 2 **CAPFIL**：CAPTM 捕捉输入滤波器控制
0：除能
1：使能
该位使能 / 除能 CAPTM 捕捉输入滤波器。

Bit 1 **CAPCLR**：CAPTM 计数器捕捉自动复位控制
0：除能
1：使能
当 H1/H2/H3 产生触发源时，该位使能 / 除能硬件自动复位 CAPTM 计数器，同时 CAPTMDL 和 CAPTMDH 的值传送到捕捉寄存器 CAPTMCL 和 CAPTMCH 后再自动重新计数 CAPTM 计数值。

Bit 0 **CAMCLR**：CAPTM 计数器比较匹配自动复位控制
0：除能
1：使能
比较匹配发生时，设置 CAMCLR 位可以使能 / 除能硬件自动复位 CAPTM 计数器。

CAPTMDL 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 CAPTM 计数器低字节寄存器 bit 7~bit 0
CAPTM 16-bit 计数器 bit 7~bit 0

CAPTMDH 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 CAPTM 计数器高字节寄存器 bit 7~bit 0
CAPTM 16-bit 计数器 bit 15~bit 8

CAPTMAL 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 CAPTM 比较低字节寄存器 bit 7~bit 0
CAPTM 16-bit 比较寄存器 bit 7~bit 0

CAPTMAH 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 CAPTM 比较高字节寄存器 bit 7~bit 0
CAPTM 16-bit 比较寄存器 bit 15~bit 8

CAPTMCL 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	x	x	x	x	x	x	x	x

“x”：未知

Bit 7~0 CAPTM 捕捉低字节寄存器 bit 7~bit 0
CAPTM 16-bit 捕捉寄存器 bit 7~bit 0

CAPTMCH 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	x	x	x	x	x	x	x	x

“x”：未知

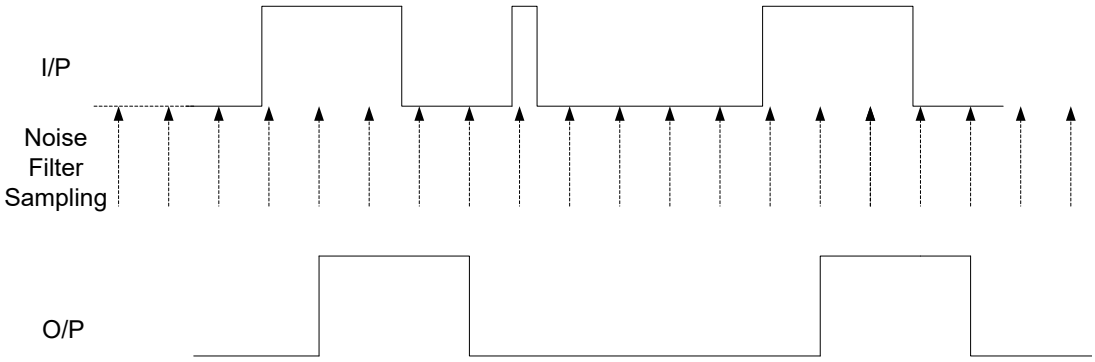
Bit 7~0 CAPTM 捕捉高字节寄存器 bit 7~bit 0
CAPTM 16-bit 捕捉寄存器 bit 15~bit 8

捕捉定时器操作

捕捉定时器模块可以用于检测和测量脉冲宽度和方波的周期。它可工作在捕捉模式或比较模式。定时器有 4 个捕捉输入 H1，H2，H3 和 CTIN。每个捕捉输入都有自己的有效边沿选择位，选择上升沿，下降沿或者双边沿捕捉。

CAPTON 位用来控制整个捕捉定时器使能 / 除能功能。若不使用捕捉定时器模块时，请除能以减少单片机耗电。通过 CAPEN 控制位来使能 / 除能捕捉输入。通过 CAPEG1 和 CAPEG0 设置触发边沿选项来选择为上升沿，下降沿或者双边沿捕捉。

该定时器还包括一个噪声滤波器，用来滤除 H1，H2，H3 和 CTIN 输入脚处不必要的干扰或脉宽。噪声滤波器可以通过 CAPFIL 位使能。如果噪声滤波器使能，CAPTM 噪声滤除电路需连续取样 CAPTM 输入 2 次或 4 次都相同时，信号才会被承认，取样的时间由 CAPNFS 决定。



噪声滤波器 (CAPNFT=0&CATNFS=0)

捕捉模式操作

捕捉定时器模块包含两个捕捉寄存器 CAPTMCL 和 CAPTMCH 用于存储捕捉到的数据。捕捉模块由 CAPEN 使能控制。如果 CAPEN 使能，外部引脚的每次触发，16 位计数器向上计数，存储于寄存器 CAPTMDL 和 CAPTMDH 内的值将被捕捉到捕捉寄存器 CAPTMCL 和 CAPTMCH 内。这个动作将引起中断标志位 CAPOF 置位。如果通过设置中断使能位 CAPOE 来使能中断，将会发生中断。如果置位 CAPCLR 位，捕捉事件发生后将自动复位 CAPTM 计数器。

比较模式操作

当定时器工作在比较模式时，寄存器 CAPTMAL 和 CAPTMAH 用来存储 16 位比较值。当 CAPTM 计数器向上计数达到 CAPTMAL 和 CAPTMAH 值时，如果中断使能，将会发生中断且 CAPCF 位将被置位，计数器重新开始计数。当比较匹配发生时，设置 CAMCLR 位为高，可以自动复位 CAPTM 计数器。CAPTM 内建边沿检测 / 捕捉 / 比较电路，设定比较寄存器，动态比对捕捉信号边沿转换时间，监控转子的移动速度，据此判断马达是否发生堵转情形，若有堵转情形，发出比较中断，可以关闭 PWM 马达驱动电路，以保护马达不会被烧毁。

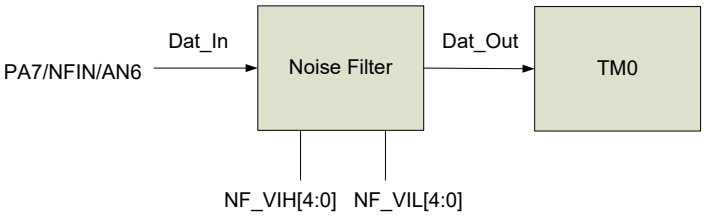
噪声滤波器

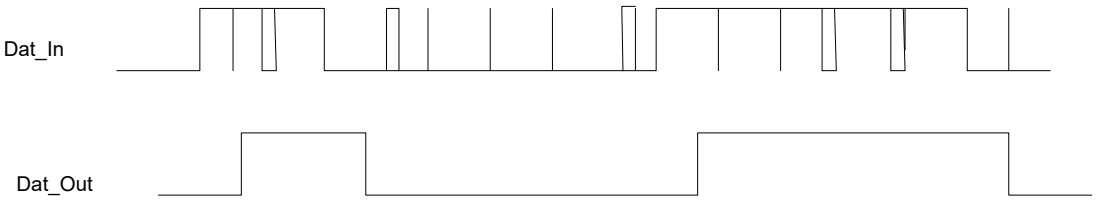
外部 NFIN 引脚连接到一个内部滤波器用来尽可能地减少不必要的事件计数或不准确的脉宽测量，然后将 NFIN 输入引脚上耦合的不良噪声或尖刺送至 16-bit PTM (TM0) 捕捉电路的 TCK0。这样可以确保马达控制电路正常地工作。

噪声滤波电路是一个输入 / 输出去电涌模拟电路，它可以滤除 μs 等级的尖锐型噪声。

抗噪脉宽能力最大值计算：(NF_VIH[4:0]-NF_VIL[4:0])×5μs，

其中 (NF_VIH[4:0]-NF_VIL[4:0])>1，允许 2，3，4，……。





噪声滤波器寄存器介绍

- **NF_VIH 寄存器**

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

Bit 7 **NF_BYPS**：旁路噪声滤波器使能位
0：除能
1：旁路模式，Dat_Out=Dat_In

Bit 6 **CINS**：TM0 捕捉源选择
0：不选择 Noise Filter Dat_Out (维持原 TM0 路径)
1：选择 Noise Filter Dat_Out

Bit 5 未定义，读为“0”

Bit 4~0 NF_VIH 寄存器 bit4~bit 0

- **NF_VIL 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	NFIS1	NFIS0	—	D4	D3	D2	D1	D0
R/W	R/W	R/W	—	R/W	R/W	R/W	R/W	R/W
POR	0	0	—	0	1	0	1	0

Bit 7~6 **NFIS1~ NFIS0**：NFIN 中断边沿控制位
00：除能
01：上升沿
10：下降沿
11：双沿

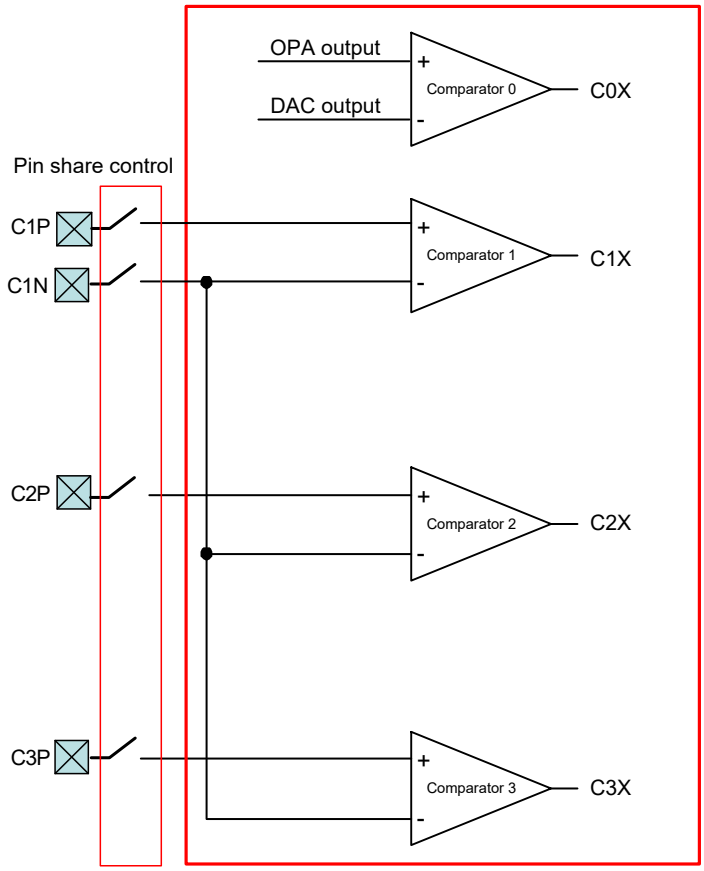
Bit 5 未定义，读为“0”

Bit 4~0 NF_VIL 寄存器 bit4~bit 0

比较器

该系列芯片中含有四个独立的模拟比较器。它们具有暂停、极性选择、迟滞等功能，可通过寄存器进行灵活配置。比较器的引脚与普通 I/O 引脚共用，当比较器功能未使用时，此引脚可做普通引脚使用而不浪费 I/O 资源。

比较器方框图



比较器操作

该单片机包含四个比较器功能，用于比较两个模拟电压，基于它们的差值上提供一个输出。控制寄存器 CPC 可控制内部比较器。

当比较器使能时，连接到与比较器共用的输入引脚的上拉电阻将自动失效。当比较器输入接近其切换电压时，由于输入信号上升或下降速度较慢，比较器输出端可能会产生一些伪输出信号。

通过选择迟滞功能提供少量正反馈给比较器可使此种情况的发生率较大程度地降低。理想情况下正负输入信号在同一个电压点时比较器将发生开关动作，但是不可避免的输入失调电压会导致情况不确定。若迟滞功能使能，也可增加切换偏差值。

CPC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	C3HYEN	C2HYEN	C1HYEN	C0HYEN	C3EN	C2EN	C1EN	C0EN
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 **C3HYEN**：比较器 3 迟滞控制位
0：关闭
1：开启
此位为迟滞控制位。该位为 1 时，比较器有一定量迟滞，具体见比较器电气特性表。滞后产生的正反馈将减少比较器门槛附近的伪开关效应的影响。

Bit 6 **C2HYEN**：比较器 2 迟滞控制位
0：关闭
1：开启
此位为迟滞控制位。该位为 1 时，比较器有一定量迟滞，具体见比较器电气特性表。滞后产生的正反馈将减少比较器门槛附近的伪开关效应的影响。

Bit 5 **C1HYEN**：比较器 1 迟滞控制位
0：关闭
1：开启
此位为迟滞控制位。该位为 1 时，比较器有一定量迟滞，具体见比较器电气特性表。滞后产生的正反馈将减少比较器门槛附近的伪开关效应的影响。

Bit 4 **C0HYEN**：比较器 0 迟滞控制位
0：关闭
1：开启
此位为迟滞控制位。该位为 1 时，比较器有一定量迟滞，具体见比较器电气特性表。滞后产生的正反馈将减少比较器门槛附近的伪开关效应的影响。

Bit 3 **C3EN**：比较器 3 开 / 关控制位
0：关闭
1：开启
此位为比较器开 / 关控制位。该位为 0 时，比较器关闭，即使其引脚上加有模拟电压也不会产生功耗。对功耗要求严格的应用中，当比较器未使用或单片机进入暂停模式之前，此位应清零。

Bit 2 **C2EN**：比较器 2 开 / 关控制位
0：关闭
1：开启
此位为比较器开 / 关控制位。该位为 0 时，比较器关闭，即使其引脚上加有模拟电压也不会产生功耗。对功耗要求严格的应用中，当比较器未使用或单片机进入暂停模式之前，此位应清零。

Bit 1 **C1EN**：比较器 1 开 / 关控制位
0：关闭
1：开启
此位为比较器开 / 关控制位。该位为 0 时，比较器关闭，即使其引脚上加有模拟电压也不会产生功耗。对功耗要求严格的应用中，当比较器未使用或单片机进入暂停模式之前，此位应清零。

Bit 0 **C0EN**：比较器 0 开 / 关控制位
0：关闭
1：开启
此位为比较器开 / 关控制位。该位为 0 时，比较器关闭，即使其引脚上加有模拟电压也不会产生功耗。对功耗要求严格的应用中，当比较器未使用或单片机进入暂停模式之前，此位应清零。

A/D 转换器

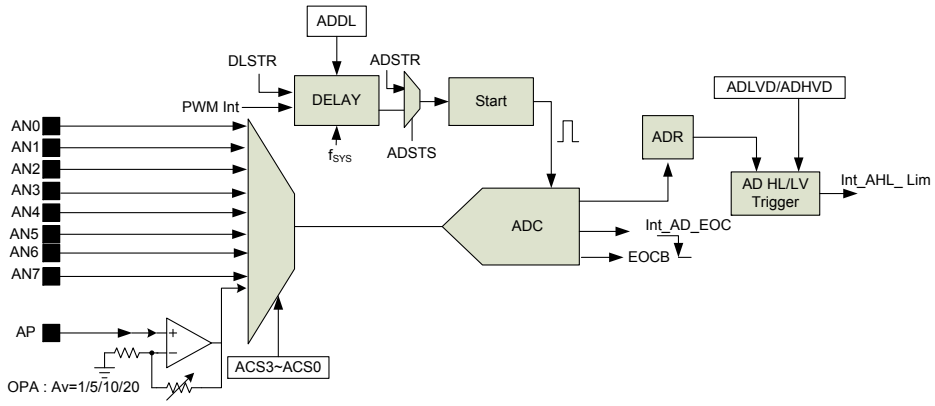
对于大多数电子系统而言，处理现实世界的模拟信号是共同的需求。为了完全由单片机来处理这些信号，首先需要通过 A/D 转换器将模拟信号转换成数字信号。将 A/D 转换器电路集成入单片机，可有效的减少外部器件，随之而来，具有降低成本和减少器件空间需求的优势。该单片机也包含一些特殊 A/D 特性专门应用于马达控制。

A/D 简介

此单片机包含一个 8+1 通道的 12 位 A/D 转换器，8 个通道可以直接接入外部模拟信号 (来自传感器或其它控制信号) 并直接将这些信号转换成 12 位的数字量，另一个由 AP(即 OPA 正端) 输入，经过放大器后输出给 A/D 转换器。一些命名为高低边界寄存器允许 A/D 转换器数字输出值分别和上下阈值作比较且产生相应的中断。另外，延迟功能允许一个延迟插入 PWM 触发 A/D 开始转换的过程中。当输出电源晶体管正在开启马达大电流，该延迟功能可用来尽可能减少错误的模拟值取样。

输入通道数	A/D 通道选择位	输入引脚
8+1	ACS3~ACS0	AN0~AN7，AP

下图显示了 A/D 转换器内部结构和相关的寄存器。



A/D 转换器结构

A/D 转换寄存器介绍

A/D 转换器的所有工作由一系列寄存器控制。ADRL/ADRH 这对只读寄存器用来存放 12 位 ADC 数据的值，ADLVDL/ADLVDH 和 ADHVDL/ADHVDH 寄存器用来存放 ADC 中断触发边界值，ADDL 寄存器用来存放 A/D 转换器启动延迟时间。剩下四个控制寄存器设置 A/D 转换器的操作和控制功能。

寄存器名称	位							
	7	6	5	4	3	2	1	0
ADRL(ADRFS=0)	D3	D2	D1	D0	—	—	—	—
ADRL(ADRFS=1)	D7	D6	D5	D4	D3	D2	D1	D0
ADRH(ADRFS=0)	D11	D10	D9	D8	D7	D6	D5	D4
ADRH(ADRFS=1)	—	—	—	—	D11	D10	D9	D8
ADCR0	ADSTR	EOCB	ADOFF	ADRFS	ACS3	ACS2	ACS1	ACS0

寄存器名称	位							
	7	6	5	4	3	2	1	0
ADCR1	ADSTS	DLSTR	PWIS	ADCHVE	ADCLVE	ADCK2	ADCK1	ADCK0
ADCR2	—	—	—	—	—	—	PWDIS1	PWDIS0
ADDL	D7	D6	D5	D4	D3	D2	D1	D0
ADLVDL	D7	D6	D5	D4	D3	D2	D1	D0
ADLVDH	—	—	—	—	D11	D10	D9	D8
ADHVDL	D7	D6	D5	D4	D3	D2	D1	D0
ADHVDH	—	—	—	—	D11	D10	D9	D8

A/D 转换寄存器列表

A/D 转换器数据寄存器 – ADRL，ADRH

对于具有 12 位 A/D 转换器的芯片，需要两个数据寄存器存放转换结果，转换结果存放格式依据 ADRFS 设定存放于一个高字节寄存器 ADRH 和一个低字节寄存器 ADRL。在 A/D 转换完毕后，单片机可以直接读取这些寄存器以获得转换结果。

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

A/D 转换数据寄存器

A/D 转换控制寄存器 – ADCR0，ADCR1，ADCR2，ADDL

寄存器 ADCR0，ADCR1 和 ADCR2 用来控制 A/D 转换器的功能和操作。这些 8 位的寄存器定义包括选择连接至内部 A/D 转换器的模拟信道，数字化数据格式，A/D 时钟源，并控制和监视 A/D 转换器的开始和转换结束状态。寄存器 ADCR0 的 ACS3~ACS0 位定义 ADC 输入通道编号。由于每个单片机只包含一个实际的模数转换电路，因此这 8 个模拟输入中的每一个都需要分别被发送到转换器。ACS3~ACS0 位的功能决定选择哪个模拟输入通道或 AP 引脚是否被连接到内部 A/D 转换器。

ADDL 寄存器用来存放 A/D 转换器启动延迟时间。

- **ADCR0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	ADSTR	EOCB	ADOFF	ADRFS	ACS3	ACS2	ACS1	ACS0
R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	1	0	0	0	0	0

Bit 7 **ADSTR**：启动 A/D 转换位
0→1→0：启动
0→1：重置 A/D 转换，并且设置 EOCB 为“1”
此位用于初始化 A/D 转换过程。通常此位为低，但如果设为高再被清零，将初始化 A/D 转换过程。当此位为高，将重置 A/D 转换器。

Bit 6 **EOCB**：A/D 转换结束标志
0：A/D 转换结束
1：A/D 转换中
此位用于表明 A/D 转换过程的完成。当转换正在进行时，此位为高。

Bit 5 **ADOFF**：ADC 模块电源开 / 关控制位
0：ADC 模块电源开
1：ADC 模块电源关
此位控制 A/D 内部功能的电源。该位被清零将使能 A/D 转换器。如果该位设为高将关闭 A/D 转换器以降低功耗。由于 A/D 转换器在不执行转换动作时都会产生一定的功耗，所以这在电源敏感的电池应用中需要多加注意。
注：1. 建议在进入空闲 / 休眠模式前，设置 ADOFF=1 以减少功耗。
2. ADOFF=1 将关闭 ADC 模块的电源。

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

Bit 3~0 **ACS3~ACS0**：选择 A/D 通道位
0000：AN0
0001：AN1
0010：AN2
0011：AN3
0100：AN4
0101：AN5
0110：OPA 输出
0111：AN6
1000：AN7
1001~1111：未定义
这三位是 A/D 通道选择控制位。由于只包含一个内部 A/D 转换电路，因此通过这些位将 8 个 A/D 输入连接到转换器。

- **ADCR1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	ADSTS	DLSTR	PWIS	ADCHVE	ADCLVE	ADCK2	ADCK1	ADCK0
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 **ADSTS**：选择 ADC 触发电路
0：选择 ADSTR 触发电路
1：选择 DELAY 触发电路

Bit 6 **DLSTR**：Delay start 功能控制
0：除能，但 ADDL 必须设置为 00H
1：使能，但 ADDL 必须设置为非 00H 值

Bit 5 **PWIS**：PWM 中断源选择位
0：选择 PWM 周期触发中断
1：选择 PWM 占空比触发中断

Bit 4~3 **ADCHVE，ADCLVE**：ADC 中断触发源选择位
00：ADLVD[9:0]<ADR[9:0]<ADHVD[9:0]
01：ADR[9:0]≤ADLVD[9:0]
10：ADR[9:0]≤ADHVD[9:0]
11：ADR[9:0]≤ADLVD[9:0] 或者 ADR[9:0]≥ADHVD[9:0]

Bit 2~0 **ADCK2~ADCK0**：ADC 时钟源选择位
000：f_{SYS}
001：$f_{SYS}/2$
010：$f_{SYS}/4$
011：$f_{SYS}/8$
100：$f_{SYS}/16$
101：$f_{SYS}/32$

110：$f_{SYS}/64$
111：未定义
这三位用于选择 A/D 转换器的时钟源。

● **ADCR2 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	PWDIS1	PWDIS0
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2　未定义，读为“0”

Bit 1~0　**PWDIS**：当 PWIS=1，PWMn 占空比触发中断源选择位
00：选择 PWM0 占空比触发中断源
01：选择 PWM1 占空比触发中断源
10：选择 PWM2 占空比触发中断源
11：保留位 (选择 PWM2 占空比触发中断源)

● **ADDL 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　ADC 延迟时间寄存器 bit 7~bit 0
ADC 延迟时间值 (由系统时钟计数)

A/D 转换器边界寄存器 – ADLVDL，ADLVDH，ADHVDL，ADHVDH

此芯片具有 12 位 A/D 转换器，每个边界值需要两个寄存器来存放触发中断值，存放格式依据 ADRFS，设定一对高边界寄存器 ADHVDH 与 ADHVDL 和一对低边界寄存器 ADLVDL 与 ADLVDH。

ADRFS	ADLVDH								ADLVDL							
	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

ADRFS	ADHVDH								ADHVDL							
	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 操作

通过 ADSTS 位选择有两种方法可以启动 ADC 转换周期。一种方法是通过 ADCR0 寄存器中的 ADSTR 位，用于打开和复位 A/D 转换器。当单片机设定此位从逻辑低到逻辑高，然后再到逻辑低，就会开始一个模数转换周期。当 ADSTR 位从逻辑低到逻辑高，但不再回到逻辑低时，ADCR0 寄存器中的 EOCB 位置 1，复位模数转换器。

第二种启动 ADC 转换的方法是使用 PWM 中断信号。通过 PWIS 位选择中断源是来自 PWM 周期还是占空比中断信号。如果中断源选择 PWM 占空比中断信

号，通过寄存器 ADCR2 中的 PWDIS1 和 PWDIS2 位选择中断触发源。DLSTR 位可启动延迟功能，在即将到来的 PWM 中断信号和实际 ADC 启动转换过程之间插入一个延迟时间，通过寄存器 ADDL 设置实际时间。实际延迟时间由 ADDL 寄存器的内容和系统时钟周期相乘计算的结果。当马达驱动晶体管瞬间释放大电流，该延迟可用来减少错误模拟取样发生的可能性。值得注意的是，如果通过 DLSTR 位选择无延迟，寄存器 ADDL 必须清零，反之亦然，如果选择为有延迟，寄存器 ADDL 必须写入非零值。

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

A/D 转换器的时钟源为系统时钟 f_{SYS} 或 f_{SYS} 分频，而分频系数由 ADCR1 寄存器中的 ADCK2~ADCK0 位决定。虽然 A/D 时钟源是由系统时钟 f_{SYS}，ADCK2~ADCK0 位决定，但可选择的最大 A/D 时钟源则有一些限制。允许的 A/D 时钟周期 t_{ADCK} 的最小值为 0.8μs，当系统时钟速度等于或超过 1.25MHz 时就必须小心。如果系统时钟速度为 5MHz 时，ADCK2~ADCK0 位不能设为“000”或“001”。必须保证设定的 A/D 转换时钟周期小于时钟周期的最小值，否则将会产生不准确的 A/D 转换值。

使用者可以参考下面的表格，被标上星号 * 的数值是不允许的，因为它们的 A/D 转换时钟周期小于规定的最小值。

f_{SYS}	A/D 时钟周期 (t_{ADCK})							
	ADCK2, ADCK1, ADCK0 =000 (f_{SYS})	ADCK2, ADCK1, ADCK0 =001 ($f_{SYS}/2$)	ADCK2, ADCK1, ADCK0 =010 ($f_{SYS}/4$)	ADCK2, ADCK1, ADCK0 =011 ($f_{SYS}/8$)	ADCK2, ADCK1, ADCK0 =100 ($f_{SYS}/16$)	ADCK2, ADCK1, ADCK0 =101 ($f_{SYS}/32$)	ADCK2, ADCK1, ADCK0 =110 ($f_{SYS}/64$)	ADCK2, ADCK1, ADCK0 =111
5MHz	200ns*	400ns*	800ns	1.6μs	3.2μs	6.4μs	12.8μs	未定义
10MHz	100ns*	200ns*	400ns*	800ns	1.6μs	3.2μs	6.4μs	未定义
20MHz	50ns*	100ns*	200ns*	400ns*	800ns	1.6μs	3.2μs	未定义

A/D 时钟周期范例

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

ADCR1 寄存器的 ADSTS 位用于选择 A/D 转换触发电路，如果该位设为“1”，则选择 DELAY 触发电路，设为“0”则为 ADSTR 触发电路。DELAY 触发电路可以由 DLSTR 来启动 Delay start 机制，也可以来自 PWM 中断。

边界寄存器组 ADHVDL/ADHVDH 和 ADLVDL/ADLVDH 包含预设值，该值与寄存器 ADRL/ADRH 存储的 A/D 转换后的值作比较。通过 ADCLVE 和 ADCHVE 位定义各种比较类型，且无论超过高边界值还是低边界值，系统都会产生中断。这个功能用来确保马达工作在安全限制电流内。

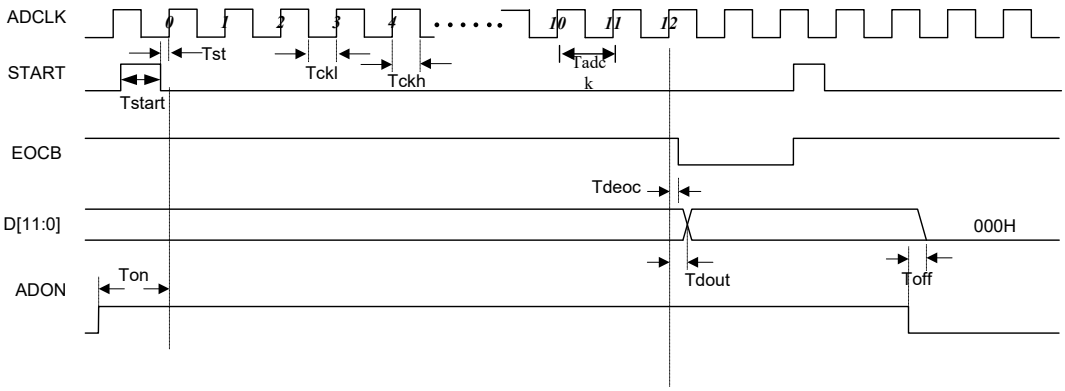
A/D 转换步骤

下面概述实现 A/D 转换过程的各个步骤。

- 步骤 1
  通过 ADCR1 寄存器中的 ADCK2~ADCK0 位，选择所需的 A/D 转换时钟。
- 步骤 2
  清零 ADCR0 寄存器中的 ADOFF 位使能 A/D。
- 步骤 3
  通过 ADCR0 寄存器中的 ACS3~ACS0 位，选择连接至内部 A/D 转换器的通道。
- 步骤 4
  引脚共用寄存器选择哪些引脚规划为 A/D 输入引脚。
- 步骤 5
  选择触发电路。可以通过设置 ADCR1 寄存器的 ADSTS 位，选择 ADSTR 触发电路还是 Delay 触发电路。
- 步骤 6
  如果要使用中断，则中断控制寄存器需要正确地设置，以确保 A/D 转换功能是启动的。总中断控制位 EMI 需要置位为“1”，A/D 转换器中断位 AEOCE 也需要置为 1。
- 步骤 7
  如果步骤 5 选择 ADSTR 触发电路，则可以通过设定 ADCR0 寄存器中的 ADSTR 位从 0 到 1 再回到 0，开始模数转换的过程。注意，该位需初始化为 0。如果步骤 5 选择 PWM 中断触发 Delay 电路，则需将 ADCR1 寄存器的 DLSTR 位置为 1，启动 Delay start 功能。
- 步骤 8
  可以轮询 ADCR0 寄存器中的 EOCB 位，检查模数转换过程是否完成。当此位成为逻辑低时，表示转换过程已经完成。转换完成后，可读取 A/D 数据寄存器 ADRL 和 ADRH 获得转换后的值。另一种方法是，若中断使能且堆栈未满，则程序等待 A/D 中断发生。

注：若使用轮询 ADCR0 寄存器中 EOCB 位的状态的方法来检查转换过程是否结束时，则中断使能的步骤可以省略。

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



A/D 转换时序图

编程注意事项

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

A/D 转换功能

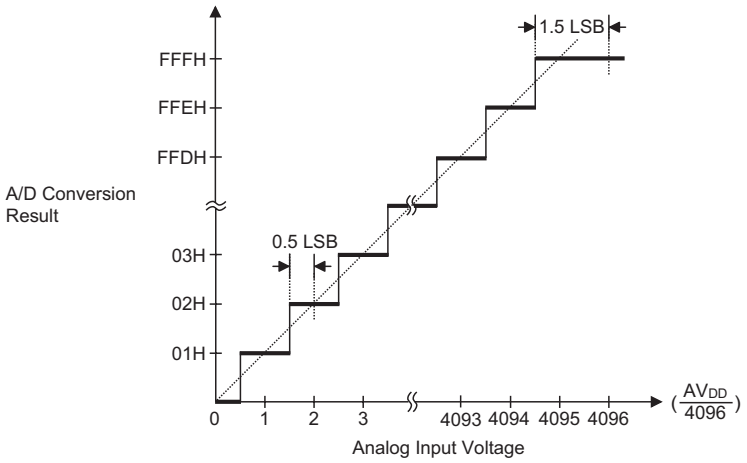
该单片机含有一组12位的A/D转换器，它们转换的最大值可达FFFH。由于模拟输入最大值等于 AV_{DD} 的电压值，因此每一位可表示 $AV_{DD}/4096$ 的模拟输入值。

$$1\ LSB=AV_{DD}/4096$$

通过下面的等式可估算A/D转换器输入电压值：

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

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



理想的 A/D 转换功能

A/D 转换应用范例

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

范例：使用查询 EOCB 的方式来检测转换结束

```
clr AEOCE                ; disable ADC interrupt
mov a,03H
mov ADCR1,a              ; select fSYS/8 as A/D clock
clr ADOFF
mov a,24h                ; setup PBPS0 to configure pins AN0~AN1
mov PBPS0,a
mov a,00h
mov ADCR0,a              ; enable and connect AN0 channel to A/D converter
:
start_conversion:
```

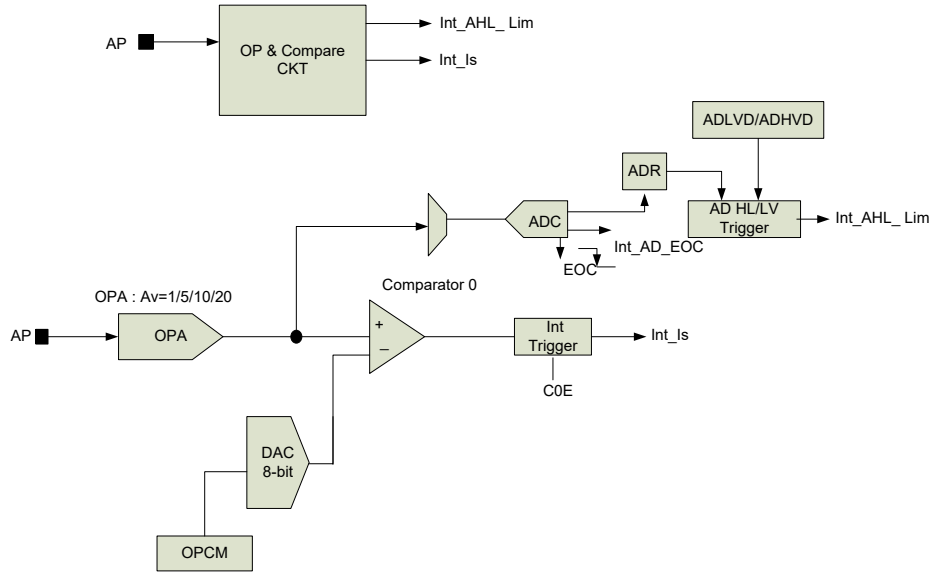
```
clr ADSTR                   ; high pulse on start bit to initiate conversion
set ADSTR                   ; reset A/D
clr ADSTR                   ; start A/D
polling_EOC:
sz EOCB                     ; poll the ADCR0 register EOCB bit to detect end
                            ; of A/D conversion
jmp polling_EOC             ; continue polling
mov a,ADRL                  ; read low byte conversion result value
mov ADRL_buffer,a           ; save result to user defined register
mov a,ADRH                  ; read high byte conversion result value
mov ADRH_buffer,a           ; save result to user defined register
:
:
jmp start_conversion        ; start next A/D conversion
```

范例：使用中断的方式来检测转换结束

```
clr MF1E                    ; disable ADC interrupt
clr AEOCE
mov a,03H
mov ADCR1,a                 ; select fSYS/8 as A/D clock
clr ADOFF
mov a,24h                   ; setup PBPS0 to configure pins AN0~AN1
mov PBPS0,a
mov a,00h
mov ADCR0,a                 ; enable and connect AN0 channel to A/D converter
Start_conversion:
clr ADSTR                   ; high pulse on start bit to initiate conversion
set ADSTR                   ; reset A/D
clr ADSTR                   ; start A/D
clr AEOCF                   ; clear ADC interrupt request flag
set AEOCE                   ; enable ADC interrupt
set MF1E                    ; enable Multi_interrupt 1
set EMI                     ; enable global interrupt
:
:
                            ; ADC interrupt service routine
ADC_ISR:
mov acc_stack,a             ; save ACC to user defined memory
mov a,STATUS
mov status_stack,a          ; save STATUS to user defined memory
:
:
mov a,ADRL                  ; read low byte conversion result value
mov adrl_buffer,a           ; save result to user defined register
mov a,ADRH                  ; read high byte conversion result value
mov adrh_buffer,a           ; save result to user defined register
:
:
EXIT_INT_ISR:
mov a,status_stack
mov STATUS,a                ; restore STATUS from user defined memory
mov a, acc_stack            ; restore ACC from user defined memory
reti
```

过流检测

该单片机包含一个完全集成的过流检测电路用于保护马达。



过流检测电路方框图

过流检测功能简介

过流检测功能模块包含一个放大器，12 位 A/D 转换器、8 位 D/A 转换器和比较器。如果检测到过流情况，应实时关闭马达外部驱动电路，以避免马达被烧毁。

两种中断用来检测过流情况：

- A/D 转换器中断 — Int_AHL_Lim
- 比较器 0 中断 — Int_Is

过流检测寄存器介绍

有三个寄存器 OPOMS，OPCM 和 OPACAL 用来控制整个过流检测电路的功能和操作。OPOMS 寄存器为 OPA 工作模式选择寄存器，OPCM 用于 OPA 比较的 8 位 DAC 寄存器，OPACAL 寄存器为 OPA 校准和比较寄存器。

寄存器名称	位							
	7	6	5	4	3	2	1	0
OPOMS	CMP0_EG1	CMP0_EG0	—	—	—	OPAVS2	OPAVS1	OPAVS0
OPCM	D7	D6	D5	D4	D3	D2	D1	D0
OPACAL	—	ARS	AOFM	AOF4	AOF3	AOF2	AOF1	AOF0

过电流检测寄存器列表

OPOMS 寄存器

Bit	7	6	5	4	3	2	1	0
Name	CMP0_EG1	CMP0_EG0	—	—	—	OPAVS2	OPAVS1	OPAVS0
R/W	R/W	R/W	—	—	—	R/W	R/W	R/W
POR	0	0	—	—	—	0	1	0

Bit 7~6 **CMP0_EG1~CMP0_EG0**：比较器有效边沿选择
00：除能比较器 0 和 DAC
01：上升沿
10：下降沿
11：双边沿

Bit 5~3 未定义，读为“0”

Bit 2~0 **OPAVS2~OPAVS0**：OPA Av 模式选择
000：除能 OPA
001：Av=5
010：Av=10
011：Av=20
111：Av=1

注：当使用 OPA 功能时，将引脚共用寄存器设为 I/O (PA1)，仍然可以作为 OPA 的输入引脚，但使用者需注意将 PA1 的上拉电阻除能，避免影响 OPA 输入。

OPCM 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 8 位 OPA 比较寄存器 bit 7~bit 0

OPACAL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	ARS	AOFM	AOF4	AOF3	AOF2	AOF1	AOF0
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	1	0	0	0	0

Bit 7 未定义，读为“0”

Bit 6 **ARS**：比较器输入偏置校准参考选择
0：比较器负输入 = 参考输入
1：比较器正输入 = 参考输入

Bit 5 **AOFM**：正常或校准模式选择
0：运算放大器或比较器模式
1：偏置校准模式

Bit 4~0 **AOF4~AOF0**：比较器输入偏置电压校准控制位
00000：最小值
10000：中间值
11111：最大值

BLDC 马达控制电路

该章节介绍单片机如何控制无刷直流马达，即 BLDC 马达。高度的功能集成和灵活性为马达提供更大范围的驱动特性。

功能简介

PWM 计数器电路：产生 PWMO，可线性调整不同占空比，用以调整马达的输出功率或转速，并可线性调整频率以调整适应电机特性，以提升马达效率或降低马达运作时所产生的共振或噪音。

内部 Mask 电路：Mask 电路决定 PWM 调制信号输出与否，以调整马达转速。可软件控制选择采用外部门驱动的晶体管对的上臂 (GAT/GBT/GCT) 或是下臂 (GAB/GBB/GCB) 来输出 PWM 调整信号。

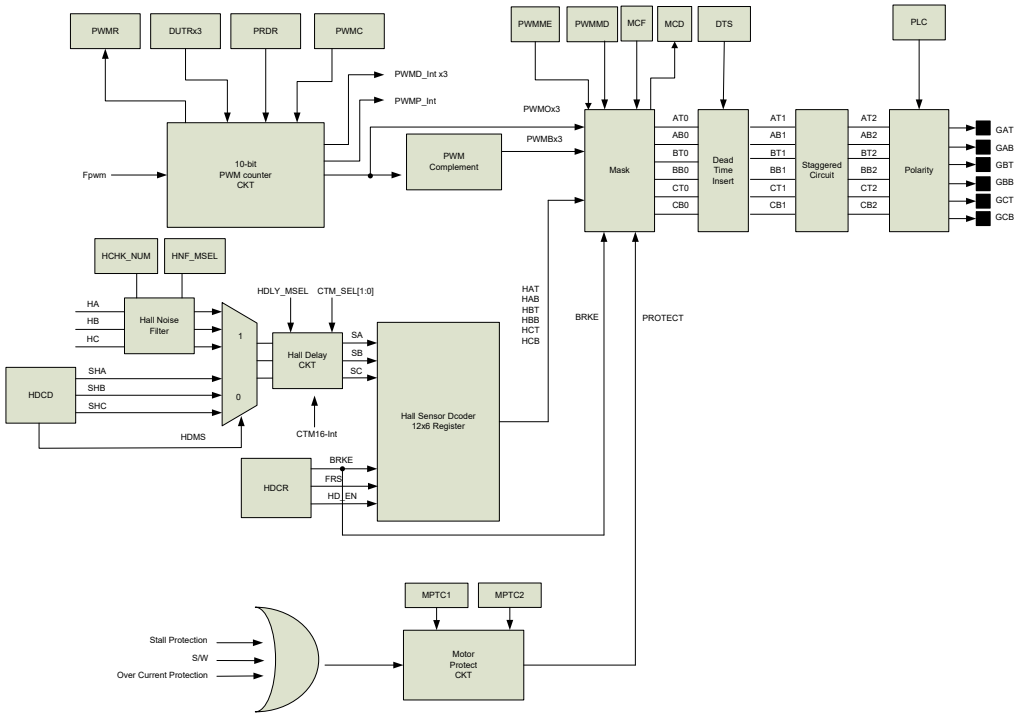
死区时间插入电路：确保外部门驱动电路晶体管对在转态时不会瞬间导通 (上下臂 MOS 都开启)，产生短路电流。死区时间由软件控制。

防呆电路：软件检测若有错误，或是外力因素如 ESD 问题，或外部门驱动晶体管对的顶端与底端的输出都是 MOS 开启的状态，防呆电路可强迫输出都是 MOS 关闭。

极性电路：调整 BLDC 输出控制接口的输出极性，以支持多种外部 MOS 门驱动电路组合。

马达保护电路包括多种检测电路用来检测堵转情况，过流保护，外部边沿触发暂停引脚，外部电平触发错误引脚等。

霍尔传感器译码电路：支持 6 步骤通信的马达方向控制的方法，通过 HDCD/HDCR 寄存器控制马达的转向前、转向后、刹车、空转。霍尔传感器译码电路的输入可以选择由霍尔传感器译码 (HA/HB/HC) 或 SHA/SHB/SHC。

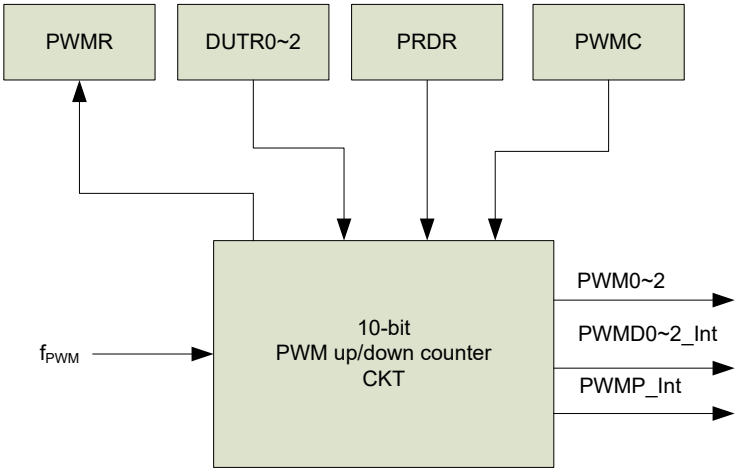


BLDC 马达控制方框图

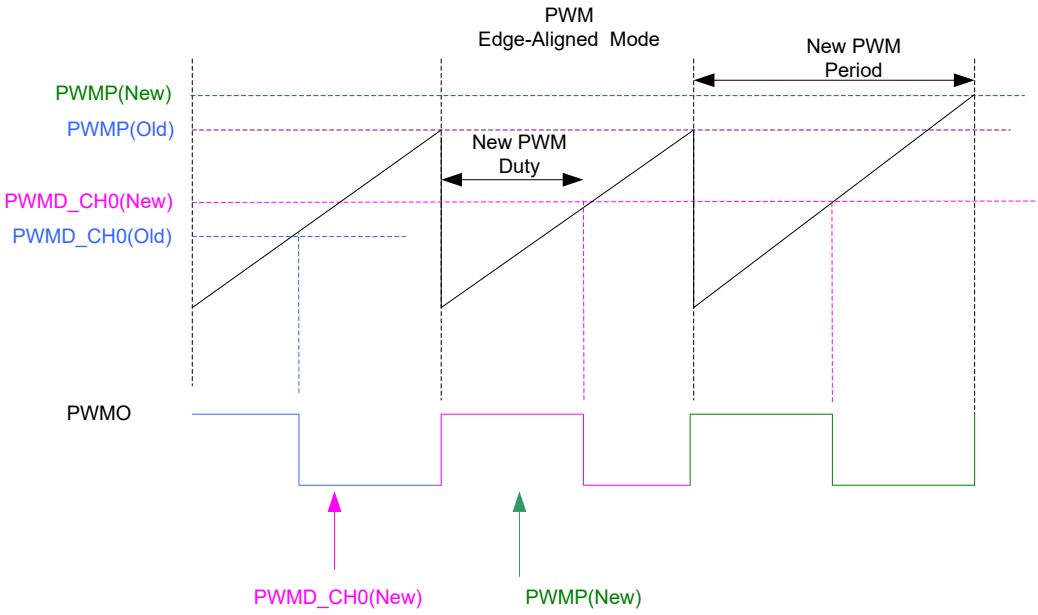
注：GAT, GAB, GBT, GBB, GCT, GCB == PWM0H, PWM0L, PWM1H, PWM1L, PWM2H, PWM2L

PWM 计数器控制电路

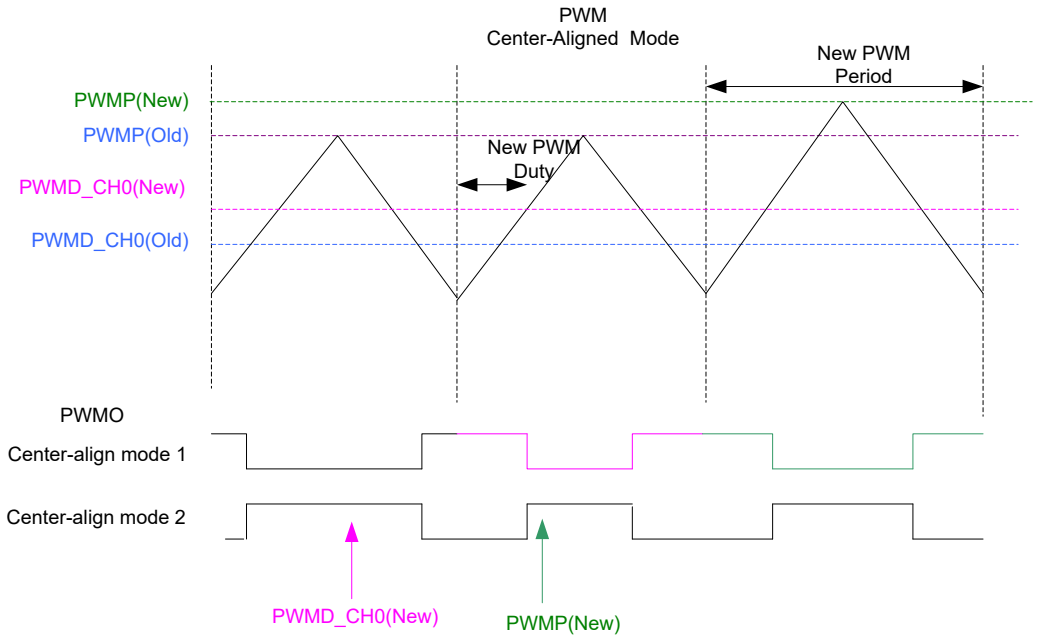
此单片机提供一个 10 位 PWM 发生器。通过给相应的 PWM 寄存器设置特定的 10 位值，PWM 功能可提供占空比和频率可调的波形。



PWM 方框图



PWM 边沿对齐模式时序图



PWM 中心对齐模式时序图

PWM 寄存器介绍

整个 PWM 操作由一系列寄存器控制。寄存器 DUTRnL/DUTRnH 用来改变 PWM 占空比以调整马达的输出功率。寄存器 PRDRL/PRDRH 组成 10 位值用来设置 PWM 周期以调整 PWM 频率。线性调整 PWM 频率，以适应马达的电机结构，如降低噪音或共振等特性。寄存器 PWMRL/PWMRH 用来监控 PWM 计数器的动态。寄存器 PWMC 的 PWMON 位用来控制 10 位 PWM 计数器的开 / 关。寄存器 PWMC 中的 PCKS1~PCKS0 位用来选择 PWM 计数器的时钟源。寄存器 PWMC 中的 PWMMS 位决定 PWM 对齐类型为边沿对齐还是中心对齐。应该注意的是，写入数据到 PWM 寄存器的顺序是先高字节后低字节。

寄存器名称	位							
	7	6	5	4	3	2	1	0
PWMC	PWMMS1	PWMMS0	PCKS1	PCKS0	PWMON	ITCMS1	ITCMS0	PWMLD
DUTR0L	D7	D6	D5	D4	D3	D2	D1	D0
DUTR0H	—	—	—	—	—	—	D9	D8
DUTR1L	D7	D6	D5	D4	D3	D2	D1	D0
DUTR1H	—	—	—	—	—	—	D9	D8
DUTR2L	D7	D6	D5	D4	D3	D2	D1	D0
DUTR2H	—	—	—	—	—	—	D9	D8
PRDRL	D7	D6	D5	D4	D3	D2	D1	D0
PRDRH	—	—	—	—	—	—	D9	D8
PWMRL	D7	D6	D5	D4	D3	D2	D1	D0
PWMRH	—	—	—	—	—	—	D9	D8

PWM 寄存器列表

● PWMC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PWMMS1	PWMMS0	PCKS1	PCKS0	PWMON	ITCMS1	ITCMS0	PWMLD
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 **PWMMS1~PWMMS0**：PWM 模式选择位
0x：边沿对齐模式
10：中心对齐模式 1
11：中心对齐模式 2

Bit 5~4 **PCKS1~PCKS0**：PWM 计数器时钟源选择
00：f_{PWM}，PWM 最小频率 =20kHz，f_{PWM} 基础值为 20MHz
01：$f_{PWM}/2$，PWM 最小频率 =10kHz
10：$f_{PWM}/4$，PWM 最小频率 =5kHz
11：$f_{PWM}/8$，PWM 最小频率 =2.5kHz

Bit 3 **PWMON**：PWM 电路开 / 关控制位
0：关闭
1：开启
该位是 PWM 功能总的开 / 关位。该位为高，则 PWM 计数器开启，清零则 PWM 除能。将该位清为零将会使正在计数的计数器停止计数且关闭 PWM 用来降低功耗。

Bit 2~1 **ITCMS1~ITCMS0**
00：除能中心对齐模式占空比中断
01：向上计数时发生的中心对齐模式占空比中断
10：向下计数时发生的中心对齐模式占空比中断
11：向上计数或向下计数时发生的中心对齐模式占空比中断

Bit 0 **PWMLD**：PWM PRDR&DUTRx (x=0~2) 寄存器更新位
0：PRDR 和 DUTRx (x=0~2) 的寄存器值不被载入计数器和比较器寄存器中。
1：计数器溢出后，PRDR 寄存器的值将被载入计数器寄存器，在下一个时钟周期硬件清除。

● DUTR0L 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 10 位 PWM0 占空比低字节寄存器
10 位 DUTR0 寄存器 bit 7~bit 0

● DUTR0H 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 10 位 PWM0 占空比高字节寄存器
10 位 DUTR0 寄存器 bit 9~bit 8

● **DUTR1L 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　10 位 PWM1 占空比低字节寄存器
10 位 DUTR1 寄存器 bit 7~bit 0

● **DUTR1H 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　10 位 PWM1 占空比高字节寄存器
10 位 DUTR1 寄存器 bit 9~bit 8

● **DUTR2L 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　10 位 PWM2 占空比低字节寄存器
10 位 DUTR2 寄存器 bit 7~bit 0

● **DUTR2H 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　10 位 PWM2 占空比高字节寄存器
10 位 DUTR2 寄存器 bit 9~bit 8

● **PRDRL 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　10 位 PWM 周期低字节寄存器
10 位 PRDR 寄存器 bit 7~bit 0

● **PRDRH 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　10 位 PWM 周期高字节寄存器
10 位 PRDR 寄存器 bit 9~bit 8

● **PWMRL 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　10 位 PWM 计数器低字节寄存器
10 位 PWM 计数器 bit 7~bit 0

● **PWMRH 寄存器**

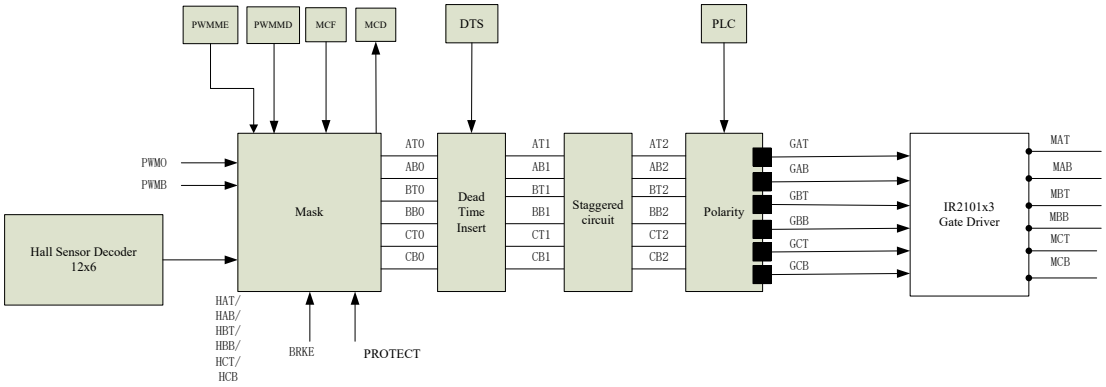
Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R	R
POR	—	—	—	—	—	—	0	0

Bit 7~2　未定义，读为“0”

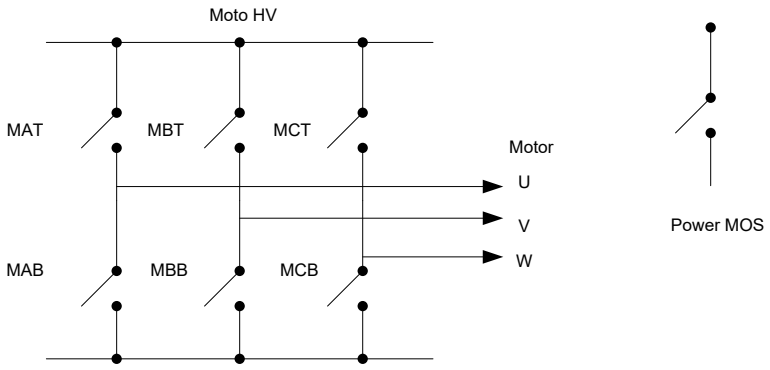
Bit 1~0　10 位 PWM 计数器高字节寄存器
10 位 PWM 计数器 bit 9~bit 8

Mask 功能

该单片机拥有马达控制 Mask 功能用来增加马达控制灵活性。



Mask 功能方框图



Mask 转换图

功能简介

内部 Mask 电路有三种工作模式：正常模式，刹车模式，马达保护模式。

- 正常模式

  正常模式下，通过寄存器 MCF 的 PWMS/MPWE 位决定马达调速的控制方式。
  - ♦ 当 PWMS=0 时，底端 PWM 输出选择晶体管对的下臂 (GAB/GBB/GCB)
  - ♦ 当 PWMS=1 时，顶端 PWM 输出选择晶体管对的上臂 (GAT/GBT/GCT)
  - ♦ 当 MPWE=0 时，PWM 输出除能 (AT0/BT0/CT0/AB0/BB0/CB0 全通)
  - ♦ 当 MPWE=1 时，PWM 输出使能 (AT0/BT0/CT0/AB0/BB0/CB0 可输出 PWM 调速)
  - ♦ 当 MPWMS=0 时，PWM 有一个互补式输出
  - ♦ 当 MPWMS=1 时，PWM 有一个非互补式输出
  - ♦ 当 MSKMS=0 时，通过 H/W 选择 Mask 模式
  - ♦ 当 MSKMS=1 时，通过 S/W 选择 Mask 模式

  MCF 与相对应的 AT0/AB0/BT0/BB0/CT0/CB0 真值表如下：

- H/W Mask 模式

  互补式控制：MPWMS=0

	HAT	HAB	AT0	AB0		HAT	HAB	AT0	AB0
PWMS=0	0	0	0	0	PWMS=1	0	0	0	0
	0	1	PWMB	PWMO		0	1	0	1
	1	0	1	0		1	0	PWMO	PWMB
	1	1	0	0		1	1	0	0

	HBT	HBB	BT0	BB0		HBT	HBB	BT0	BB0
PWMS=0	0	0	0	0	PWMS=1	0	0	0	0
	0	1	PWMB	PWMO		0	1	0	1
	1	0	1	0		1	0	PWMO	PWMB
	1	1	0	0		1	1	0	0

	HCT	HCB	CT0	CB0		HCT	HCB	CT0	CB0
PWMS=0	0	0	0	0	PWMS=1	0	0	0	0
	0	1	PWMB	PWMO		0	1	0	1
	1	0	1	0		1	0	PWMO	PWMB
	1	1	0	0		1	1	0	0

非互补式控制：MPWMS=1

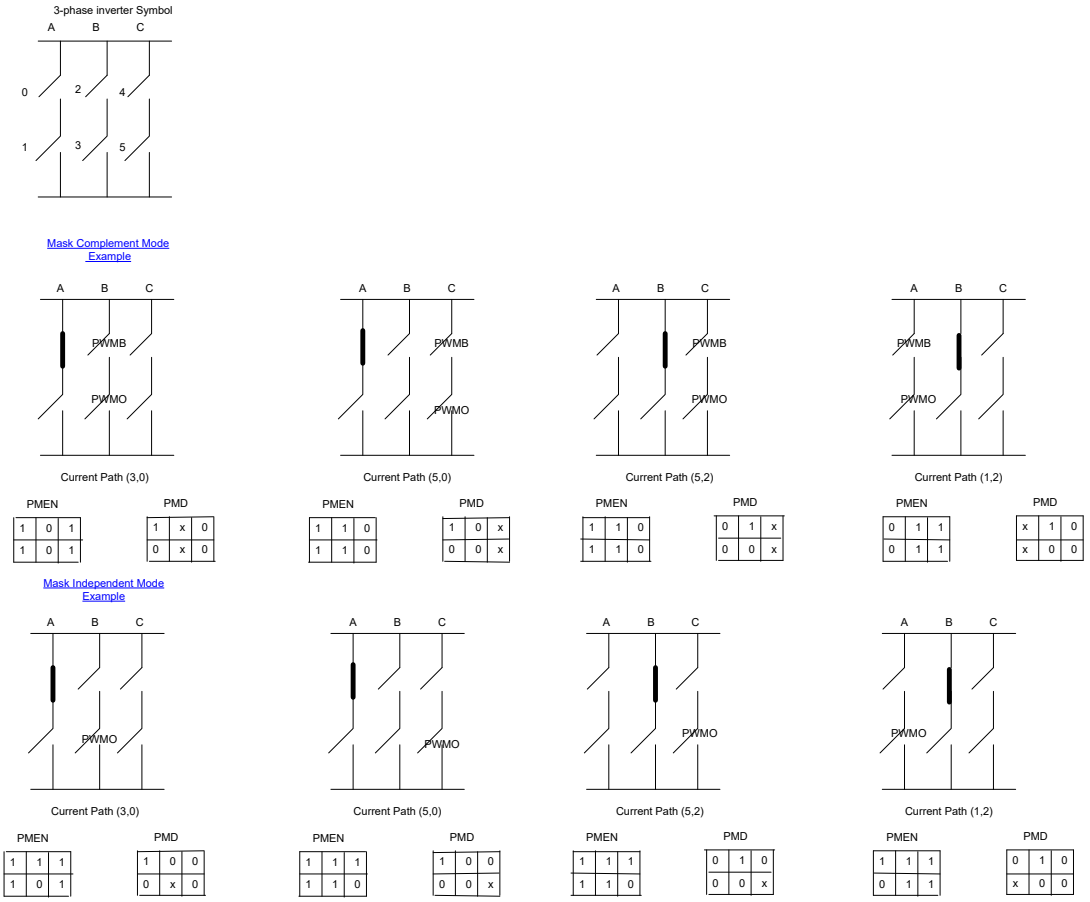
	HAT	HAB	AT0	AB0		HAT	HAB	AT0	AB0
PWMS=0	0	0	0	0	PWMS=1	0	0	0	0
	0	1	0	PWMO		0	1	0	1
	1	0	1	0		1	0	PWMO	0
	1	1	0	0		1	1	0	0

	HBT	HBB	BT0	BB0		HBT	HBB	BT0	BB0
PWMS=0	0	0	0	0	PWMS=1	0	0	0	0
	0	1	0	PWMO		0	1	0	1
	1	0	1	0		1	0	PWMO	0
	1	1	0	0		1	1	0	0

	HCT	HCB	CT0	CB0		HCT	HCB	CT0	CB0
PWMS=0	0	0	0	0	PWMS=1	0	0	0	0
	0	1	0	PWMO		0	1	0	1
	1	0	1	0		1	0	PWMO	0
	1	1	0	0		1	1	0	0

- S/W Mask 模式

  寄存器 PWMME，PWMMD，MCF 和 MCD 用来控制 Mask 电路。寄存器 PWMME 用来控制 PWM 信号，寄存器 PWMMD 用来决定 MOS 门驱动电路开或关。注意，设置 PWMS 和 MPWMS 位或任何与 PWM 功能相关的位只对 H/W 或 S/W 模式有效。



Mask S/W 模式电路

注：1. 在使能 mask 期间，当寄存器 PWMxH 和 PWMxL 同时 mask 时，PMD0 和 PMD1 不可同时为“1”，PMD2 和 PMD3 不可同时为“1”，PMD4 和 PMD5 不可同时为“1”；若是同时为“1”，开关 2n 和开关 2n+1 将输出“0”。

2. 若 PWM 和互补式 PWM 同时使能，则寄存器 PWMxH 和 PWMxL 其中一个为 PWM 输出，则另外一个不可 mask 为“1”，硬件将输出自动设成“0”。

3．若寄存器 PWMxH 和 PWMxL 被设置为 I/O 功能，则 PWM Mask 功能无效。

S/W Mask 寄存器 – PWMME，PWMMD

- **PWMME 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PME5	PME4	PME3	PME2	PME1	PME0
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　**PMEn**：PWM Mask 使能位 (n=0~5)
0：PWM 发生信号输出到下一级
1：PWM 发生信号被 mask，PMDn 输出到下一级。
当使能这个位，PWM 发生信号被 mask，相应的 PWMn 通道将输出 PMDn 数据。

- **PWMMD 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PMD5	PMD4	PMD3	PMD2	PMD1	PMD0
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　**PMDn**：PWM Mask 数据位 (n=0~5)
0：PWMn 输出逻辑低
1：PWMn 输出逻辑高
如果相应的 PMEn 为 1，该数据位可控制 PWMn 输出引脚的状态。

- 刹车模式

刹车模式有最高优先级。当刹车模式启动时，外部的门驱动晶体管对上臂关闭，下臂开启。刹车真相解码表如下所示。

BRKE=1	AT0	BT0	CT0	AB0	BB0	CB0
	0	0	0	1	1	1

- 马达保护模式

当马达保护模式启动时，PROTECT=1，外部的门驱动晶体管对可选择刹车 (上臂关闭，下臂开启) 或是空转 (上臂和下臂都关闭)。保护解码表如下所示。

PROTECT 解码表如下：

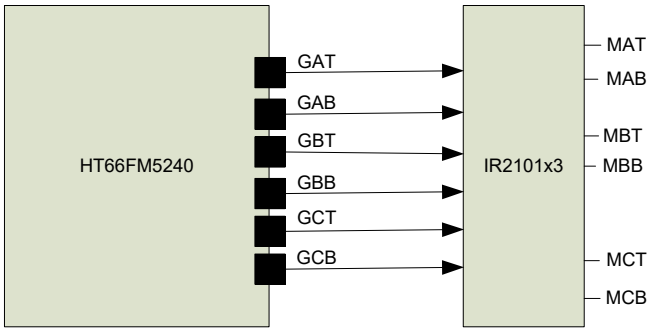
PROTECT=1	GAT	GBT	GCT	GAB	GBB	GCB
FMOS=0	0	0	0	0	0	0
FMOS=1	0	0	0	1	1	1

以 6 步通信机制为例，若导通 U 线和 W 线，则关闭 V 线。

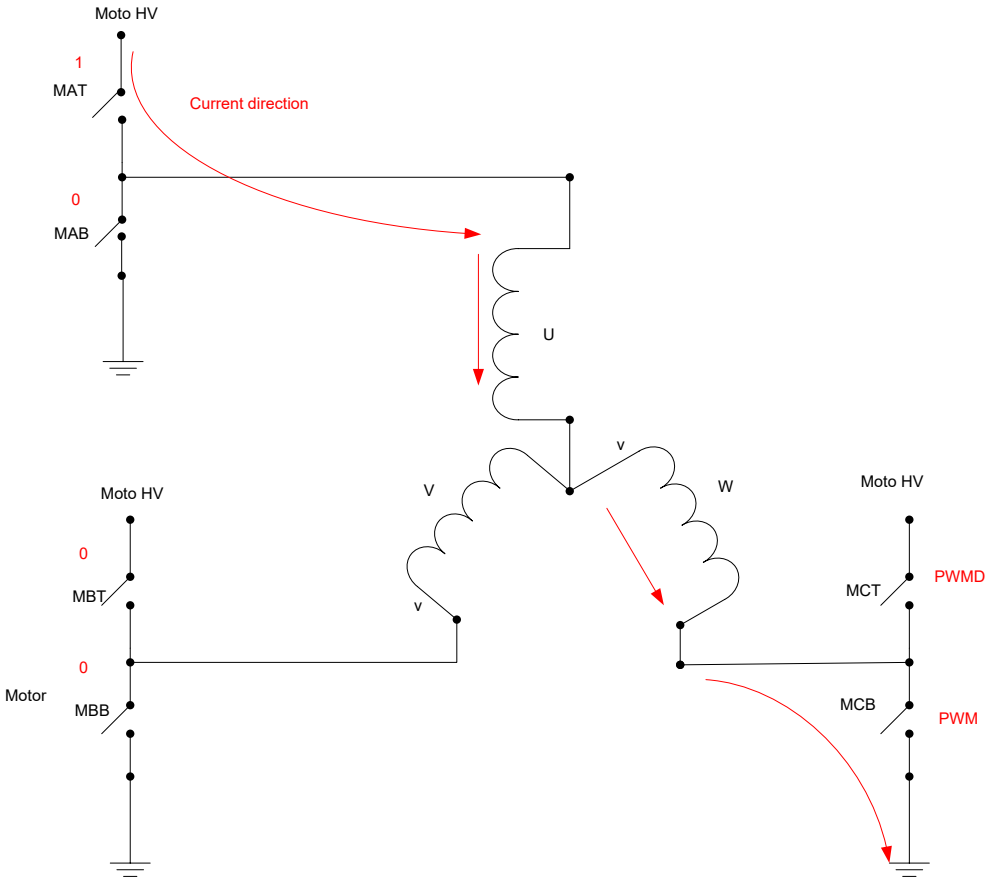
若 GAT=1，GAB=0，则打开 U 线。

若 GBT=0，GBB=0，则关闭 V 线。

若 GCT=PWMD，GCB=PWM 开启 W 线并以 PWM 引擎的 DUTR 寄存器来调整马达的输出功率，以调整转速。



驱动信号方框图



马达线连接图

寄存器介绍

该单片机有两个寄存器用来控制 Mask 功能。寄存器 MCF 用于控制 Mask 各种功能，寄存器 MCD 用于读取门驱动电路输出状态。

- **MCF 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	MSKMS	—	—	—	MPWMS	MPWE	FMOS	PWMS
R/W	R/W	—	—	—	R/W	R/W	R/W	R/W
POR	0	—	—	—	0	1	0	0

Bit 7 **MSKMS**：Mask 模式选择位
0：H/W Mask 模式
1：S/W Mask 模式

Bit 6~4 未定义，读为“0”

Bit 3 **MPWMS**：MASK PWM 模式选择
0：互补式输出
1：非互补式输出

Bit 2 **MPWE**：PWM 输出控制
0：PWM 输出除能 (AT0/BT0/CT0/AB0/BB0/CB0 不输出 PWM)
1：PWM 输出使能 (AT0/BT0/CT0/AB0/BB0/CB0 可输出 PWM 调速)

Bit 1 **FMOS**：PROTECT Mask 输出选择
0：AT0/BT0/CT0=0，AB0/BB0/CB0=0
1：AT0/BT0/CT0=0，AB0/BB0/CB0=1

Bit 0 **PWMS**：顶端 / 底端 PWM 选择
0：选择底端 PWM 输出
1：选择顶端 PWM 输出

- **MCD 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	GAT	GAB	GBT	FHC	FHB	FHA
R/W	—	—	R	R	R	R	R	R
POR	—	—	0	0	0	1	1	1

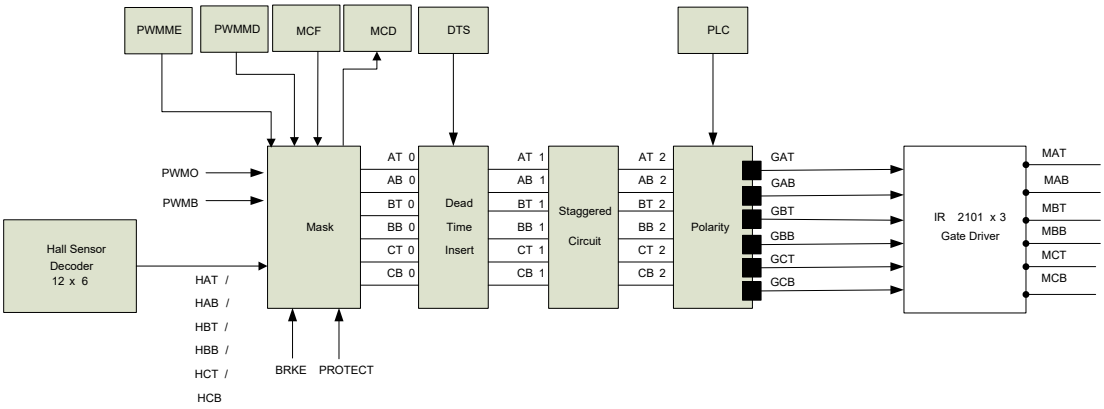
Bit 7~6 未定义，读为“0”

Bit 5~3 **GAT/GAB/GBT**：门驱动输出显示

Bit 2~0 **FHC/FHB/FHA**：HC/HB/HA 滤波输出
经过霍尔噪声滤波器滤波后的 HC/HB/HA 输出信号

其它功能

一些其它功能用来控制马达驱动信号的灵活性。它们包括死区时间功能，防呆功能和极性功能。



死区时间，防呆和极性功能方框图

死区时间功能

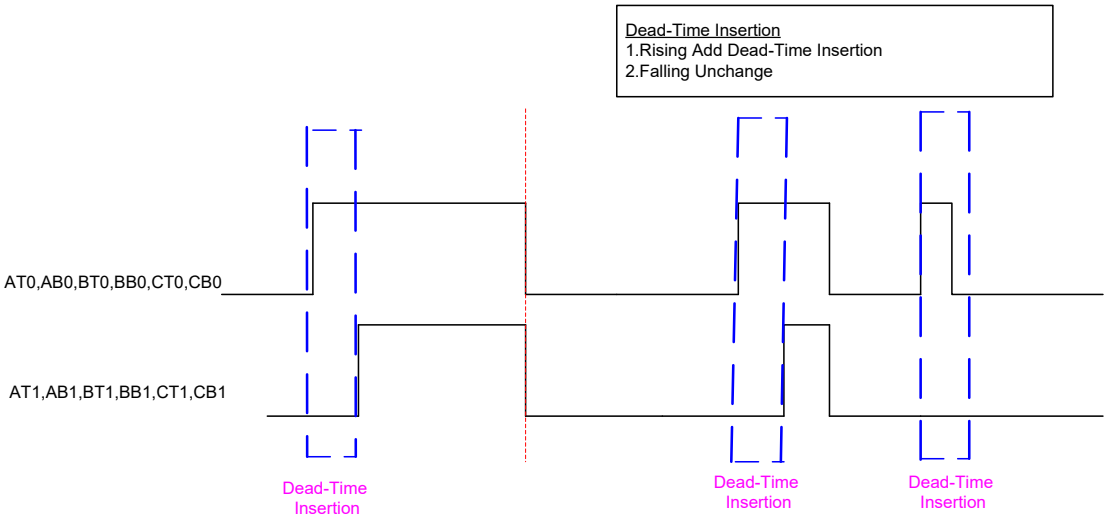
死区时间功能确保外部门驱动电路晶体管对在转态时不会瞬间导通 (上下臂 MOS 都开启)，以防产生短路电流。死区时间可通过应用程序控制在 0.3μs~5μs 左右。

死区时间插入电路要求 6 个单独的输出电路：

当 AT0/AB0/BT0/BB0/CT0/CB0 输出上升沿时，可插入死区时间。

当 AT0/AB0/BT0/BB0/CT0/CB0 输出下降时，输出不变。

死区时间插入电路只在马达控制时使用，可利用寄存器 DTS 中的 DTE 位来使能 / 除能死区时间功能。



死区时间插入时序图

寄存器 DTS 为死区时间选择寄存器。

● **DTS 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	DTCKS1	DTCKS0	DTE	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 7~6 **DTCKS1~DTCKS0**：死区时间时钟源选择
00：$f_{DT}=f_{SYS}$，f_{SYS} 基础值为 20MHz
00：$f_{DT}=f_{SYS}/2$
00：$f_{DT}=f_{SYS}/4$
00：$f_{DT}=f_{SYS}/8$

Bit 5 **DTE**：死区时间使能位
0：死区时间 =0
1：死区时间 =(DTS[4:0]+1)/f_{DT}

Bit 4~0 **D4~D0**：死区时间寄存器 bit 4 ~bit 0
死区时间计数器。死区时间单位：5 位死区时间值
死区时间 =(DTS[4:0]+1)/f_{DT}

防呆功能

若发生软件错误，或是外界因素如 ESD，造成外部门驱动晶体管对的顶端与底端的输出都是 MOS 开启的状态，防呆电路强迫输出都是 MOS 关闭。

AT1	AB1	AT2	AB2
0	0	0	0
0	1	0	1
1	0	1	0
1	1	0	0

注：BLDC 马达控制电路中都默认把外部门驱动晶体管对设计为 N 型晶体管对。以上表格中值为 1 代表晶体管开启，0 代表晶体管关闭。

极性功能

该功能用来设置外部门驱动晶体管开 / 关极性状态。寄存器 PLC 用于整个功能控制。

● **PLC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCBC	PCTC	PBBC	PBTC	PABC	PATC
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 **PCBC**：C 对底端门输出选择
0：正相输出
1：反相输出

Bit 4 **PCTC**：C 对顶端门输出选择
0：正相输出
1：反相输出

Bit 3 **PBBC**：B 对底端门输出选择
0：正相输出
1：反相输出

Bit 2 **PBTC**：B 对顶端门输出选择
0：正相输出
1：反相输出

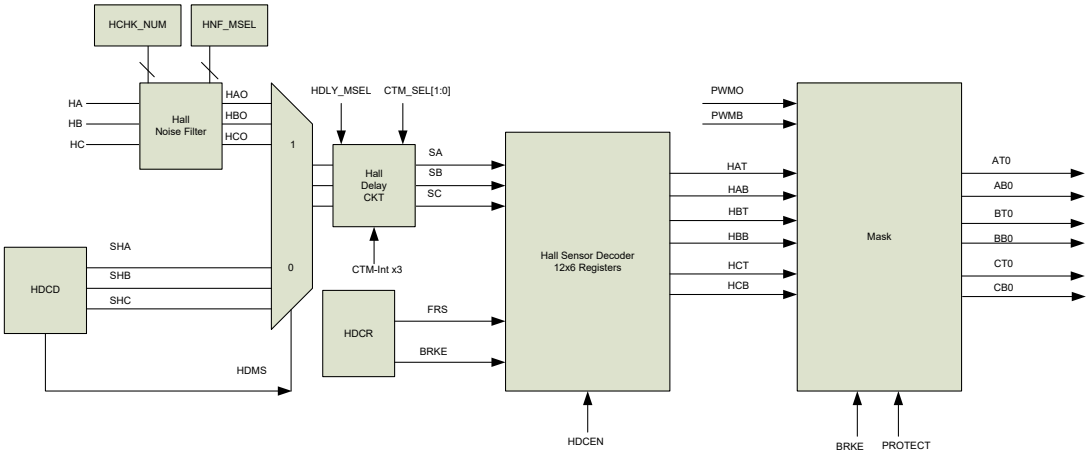
Bit 1 **PABC**：A 对底端门输出选择
0：正相输出
1：反相输出

Bit 0 **PATC**：A 对顶端门输出选择
0：正相输出
1：反相输出

注：默认输出引脚 GAT/GAB/GBT/GBB/GCT/GCB 状态为高阻抗。

霍尔传感器译码电路

该单片机包含一个完全集成的霍尔传感器译码电路连接到霍尔传感器用于 BLDC 马达的方向和转速控制。

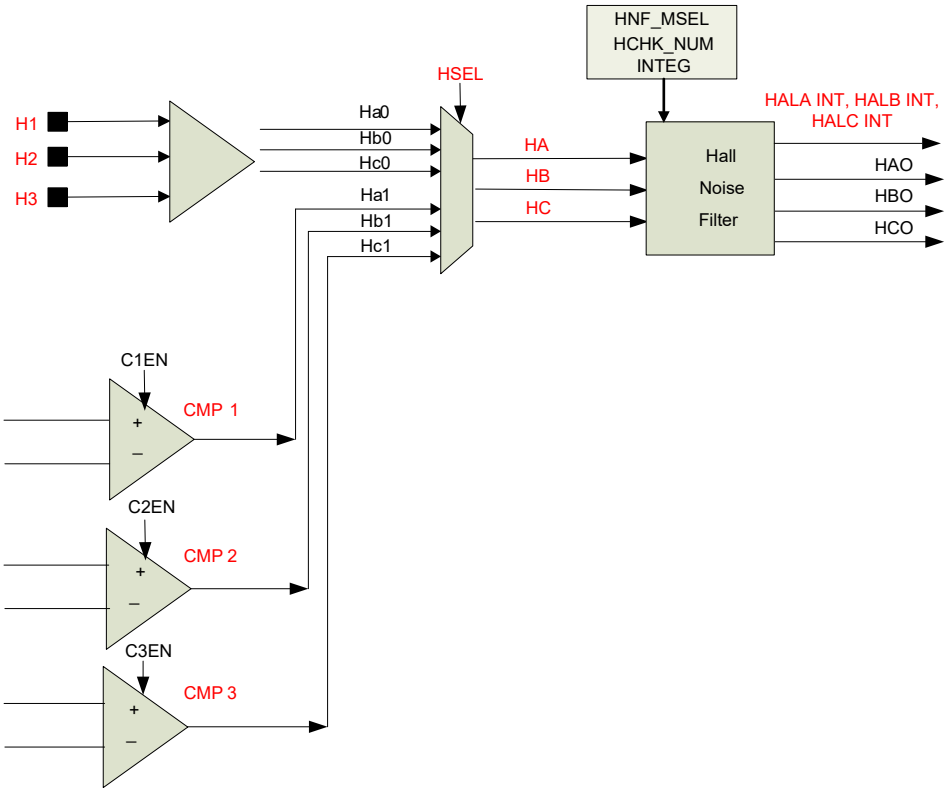


霍尔传感器译码电路图

通过设置 HDMS 位为高来选择实际的霍尔传感器信号输入。如果 HDMS 位为 0，则 SHA/SHB/SHC 取代实际的霍尔传感器信号输入。

霍尔传感器噪声滤波器

该单片机内建霍尔噪声滤波器以滤除马达驱动器因转换电流过大产生的噪声。该噪声干扰可能影响霍尔传感器输入 (H1/H2/H3)，造成霍尔传感器解码输出有误动作。



霍尔传感器噪声滤波器方框图

一些寄存器用来控制噪声滤波器。寄存器 HNF_MSEL 中的 HNF_EN 位用来使能 / 除能噪声滤波器。马达控制 Sensorless 应用若使用内建的 3 个比较器时，要将 CMP1/CMP2/CMP3 迟滞功能打开。

HNF_EN 位	状态
0	噪声滤波器关闭 — 不使用 HA/HB/HC
1	噪声滤波器开启

霍尔传感器噪声滤波器使能

通过 HFR_SEL[3:0] 位设置霍尔传感器噪声滤波器的取样频率。

HCHK_NUM[4:0] 位用来设置霍尔传感器输入比对次数。

HCHK_NUM[4:0]× 取样间距 = 抗噪声能力 = 霍尔延迟时间

要注意的是，霍尔延迟时间越长，转子速度回馈失真越严重。

霍尔传感器延迟功能

霍尔传感器电路内建霍尔延迟电路以支持霍尔相位超前 / 相位落后功能。开启霍尔译码器使能前，需先执行下列步骤来显示该功能如何运作。

- 步骤 1
  设定霍尔解码表，以决定相位超前或相位落后
- 步骤 2
  通过 CTM_SEL1~CTM_SEL0 位选择哪一个 PTM 来作延迟时间，并设定成比较匹配模式，以决定延迟时间。

- 步骤 3

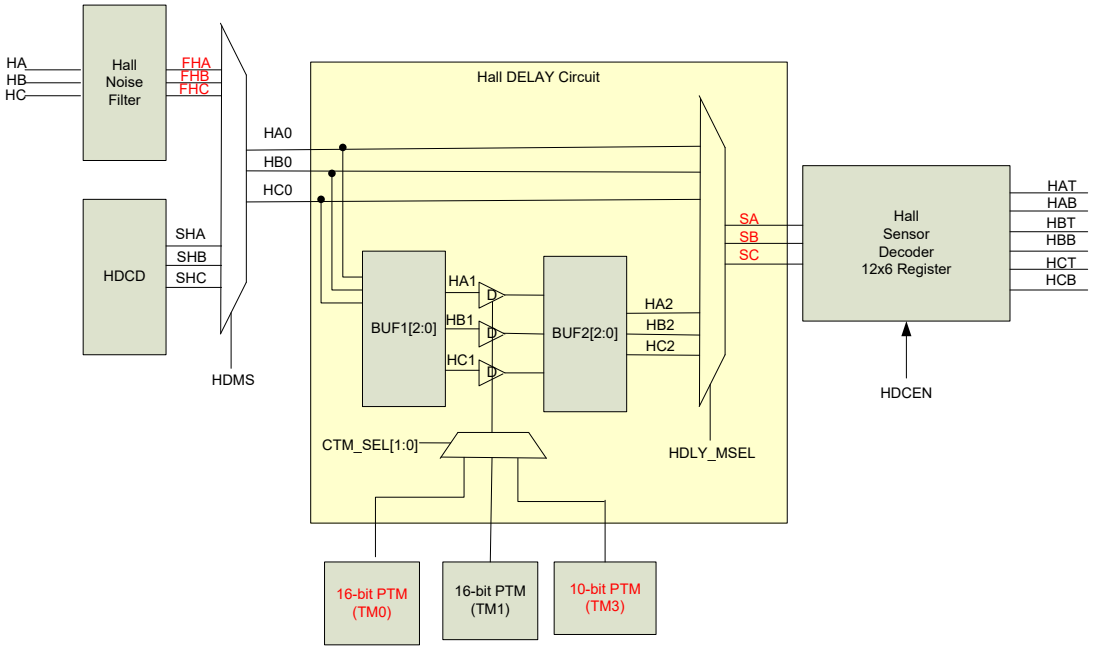
  通过 HDLY_MSEL 位选择霍尔延迟电路工作模式
  HDLY_MSEL 位的默认值为 0，霍尔延迟电路除能。
  如果 HDLY_MSEL 位置为高，霍尔延迟电路使能。

- 步骤 4

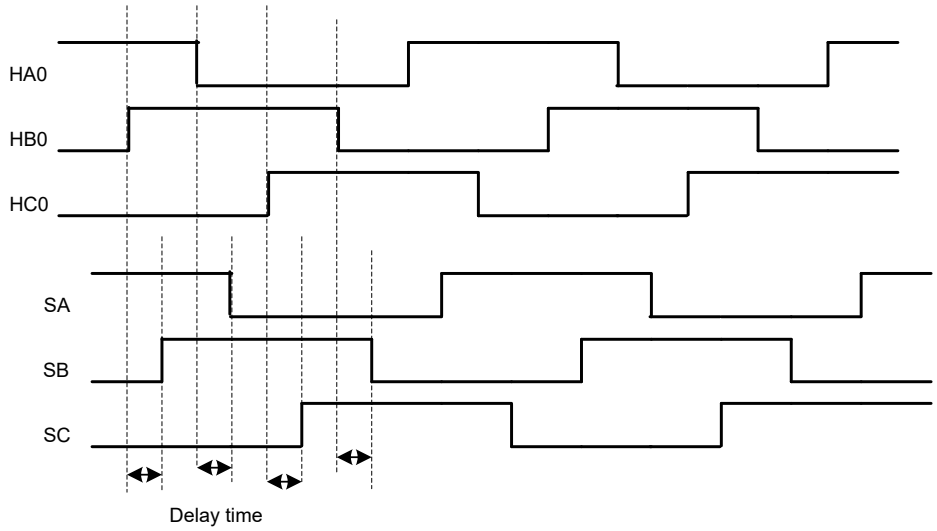
  通过 HDCEN 位开启霍尔译码器。
  关于 HDLY_MSEL 位必须注意以下几点：
  - ♦ 当 HDLY_MSEL 位为 0 时，BUF1[2:0]=0 和 BUF2[2:0] 都将清零。
  - ♦ 当 HDLY_MSEL 位为 0 时，PTM0/PTM1/PTM3 都为最初的 PTM 功能。
  - ♦ 当 HDLY_MSEL 位为 1，被延迟电路选到的 PTM，基本上将专门用于霍尔延迟电路计时用。

  原 PTM 功能仍然有效，除了 PTnON 位须由硬件来自动控制。
  在开启延迟功能前，用户须适当配置 PTM 功能。
  - ♦ 需先保持 PTnON=0 和 PTnPAU=0
  - ♦ 需将 PTM 设置为比较匹配模式
  - ♦ 设定 PTnCCLR=1。(即当比较器 A 比较匹配时即清除 PTM。)
  - ♦ 适当设定 PTMnA 与 PTCKn 以决定延迟时间。

  在使能延迟电路后，HDLY_MSEL 位将从低变为高。延迟时间不得大于霍尔输入的一步时间 (共 6 步)，否则输出无法预期，造成控制失步。



延迟功能方框图



延迟功能时序表

马达控制驱动信号

通过 HDCR 和 HDCD 寄存器和一系列 HDCT 寄存器 HDCT0~HDCT1 来控制 BLDC 马达的方向。当使用霍尔传感器译码功能时，分别通过 HDCR 寄存器 FRS 位和 BRAKE 位，决定马达的方向或者是否刹车。HDCT0~HDCT5 为马达向前转表；HDCT6~HDCT11 为马达向后转表。

霍尔传感器解码真值表如下。

	60 度			120 度			Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SA	SB	SC	SA	SB	SC	HAT	HAB	HBT	HBB	HCT	HCB
向前转 (HDCEN=1 FRS=0 BRKE=0)	1	0	0	1	0	0	HDCT0[5:0]					
	1	1	0	1	1	0	HDCT1[5:0]					
	1	1	1	0	1	0	HDCT2[5:0]					
	0	1	1	0	1	1	HDCT3[5:0]					
	0	0	1	0	0	1	HDCT4[5:0]					
	0	0	0	1	0	1	HDCT5[5:0]					

霍尔传感器译码器向前真值表

	60 度			120 度			Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SA	SB	SC	SA	SB	SC	HAT	HAB	HBT	HBB	HCT	HCB
向后转 (HDCEN=1 FRS=1 BRKE=0)	1	0	0	1	0	0	HDCT6[5:0]					
	1	1	0	1	1	0	HDCT7[5:0]					
	1	1	1	0	1	0	HDCT8[5:0]					
	0	1	1	0	1	1	HDCT9[5:0]					
	0	0	1	0	0	1	HDCT10[5:0]					
	0	0	0	1	0	1	HDCT11[5:0]					

霍尔传感器译码器向后真值表

下面三个表分别为刹车功能，霍尔译码器除能功能和霍尔译码器出错功能的真值表。

刹车 (BRKE=1 HDCEN=X FRS=X)	60 度			120 度			Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SA	SB	SC	SA	SB	SC	HAT	HAB	HBT	HBB	HCT	HCB
	V	V	V	V	V	V	0	1	0	1	0	1

刹车功能真值表

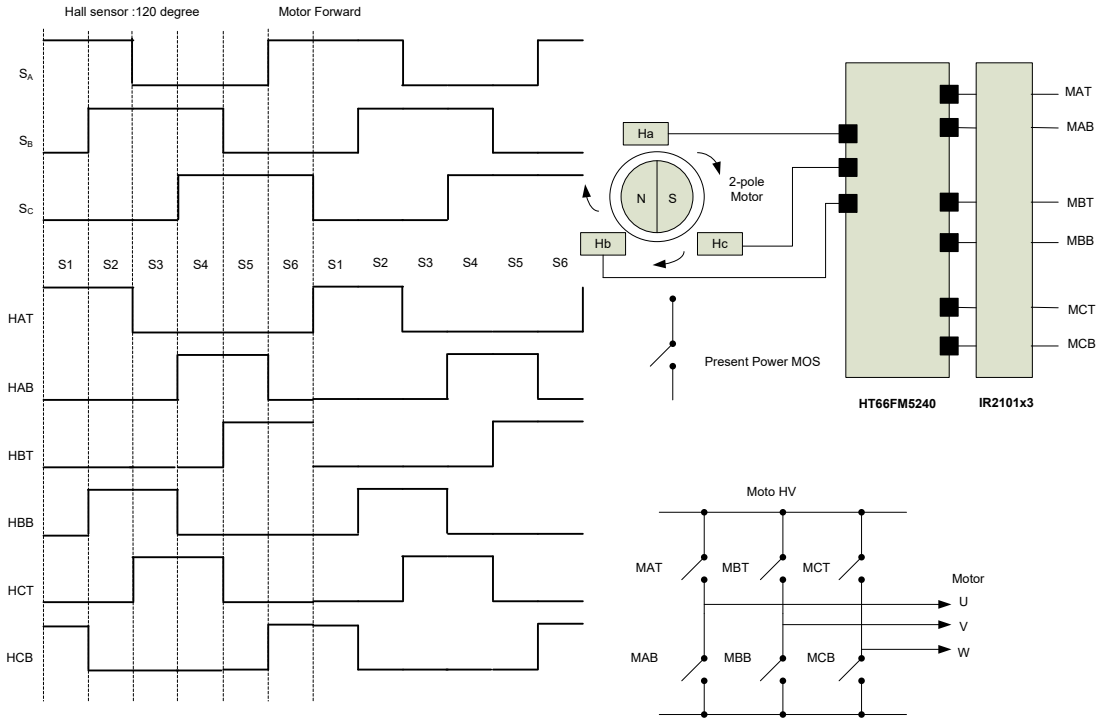
霍尔译码器除能 (HDCEN=0) 空转	60 度			120 度			Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SA	SB	SC	SA	SB	SC	HAT	HAB	HBT	HBB	HCT	HCB
	V	V	V	V	V	V	0	0	0	0	0	0

霍尔译码器除能功能真值表

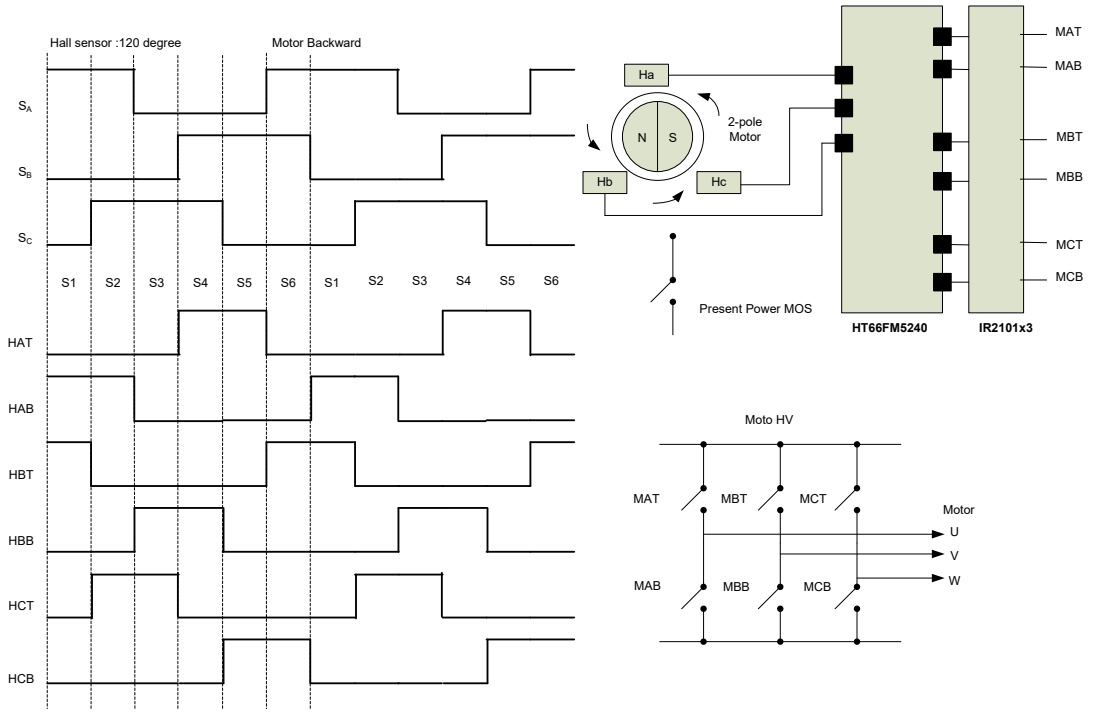
霍尔传感器出错 (HDCEN=X)	60 度			120 度			Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SA	SB	SC	SA	SB	SC	HAT	HAB	HBT	HBB	HCT	HCB
	1	0	1	1	1	1	0	0	0	0	0	0
	0	1	0	0	0	0	0	0	0	0	0	0

霍尔译码器出错功能真值表

真值表中的数据和它们怎么联系实际马达驱动信号如下面时序图所示。以典型 6 步骤通信为例，下图是马达向前转 / 马达向后转时序图。



马达驱动信号时序图 – 向前方向



马达驱动信号时序图 – 向后方向

霍尔传感器译码相关寄存器介绍

寄存器 HDCR 为霍尔传感器解码控制寄存器；寄存器 HDCD 为霍尔传感器译码输入数据寄存器；寄存器 HDCT0~HDCT11 为霍尔传感器译码表。寄存器 HCHK_NUM 为霍尔噪声滤波器检查号码寄存器；寄存器 HNF_MSEL 为霍尔噪声滤波器模式选择寄存器。

- **INTEG0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	HSEL	INTCS1	INTCS0	INTBS1	INTBS0	INTAS1	INTAS0
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　**HSEL**：HA/HB/HC 来源选择
0：H1/H2/H3
1：CMP1/CMP2/CMP3 输出

Bit 5~4　**INTCS1~INTCS0**：FHC 中断边沿控制 INTC
00：除能
01：上升沿
10：下降沿
11：双边沿

Bit 3~2　**INTBS1~INTBS0**：FHB 中断边沿控制 INTB
00：除能
01：上升沿
10：下降沿
11：双边沿

Bit 1~0 **INTAS1~INTAS0**：FHA 中断边沿控制 INTA
00：除能
01：上升沿
10：下降沿
11：双边沿

● HDCR 寄存器

Bit	7	6	5	4	3	2	1	0
Name	CTM_SEL1	CTM_SEL0	HDLY_MSEL	HALS	HDMS	BRKE	FRS	HDCEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	1	0	0	0	0

Bit 7~6 **CTM_SEL1~CTM_SEL0**：霍尔延迟电路 TM 选择
00：TM0(16-bit PTM))
01：TM1(16-bit PTM)
10：TM3(10-bit PTM)
11：未使用

Bit 5 **HDLY_MSEL**：霍尔延迟电路选择
0：选择原始路径
1：选择霍尔延迟电路

Bit 4 **HALS**：霍尔传感器译码模式选择
0：霍尔传感器 60 度配置
1：霍尔传感器 120 度配置

Bit 3 **HDMS**：霍尔传感器译码模式选择
0：S/W 模式
1：霍尔传感器模式

Bit 2 **BRKE**：马达刹车控制
0：AT/BT/CT/AB/BB/CB=V
1：AT/BT/CT=0，AB/BB/CB=1

Bit 1 **FRS**：马达向前转 / 向后转选择
0：向前转
1：向后转

Bit 0 **HDCEN**：霍尔传感器解码使能
0：除能
1：使能

● HDCD 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	SHC	SHB	SHA
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 未定义，读为“0”

Bit 2 **SHC**：S/W 霍尔 C

Bit 1 **SHB**：S/W 霍尔 B

Bit 0 **SHA**：S/W 霍尔 A

● **HDCTn 寄存器 n=0~11**

Bit	7	6	5	4	3	2	1	0
Name	—	—	HATDn	HABDn	HBTDn	HBBDn	HCTDn	HCBDn
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　**HATDn**：GAT 输出状态控制
0：输出 0
1：输出 1

Bit 4　**HABDn**：GAB 输出状态控制
0：输出 0
1：输出 1

Bit 3　**HBTDn**：GBT 输出状态控制
0：输出 0
1：输出 1

Bit 2　**HBBDn**：GBB 输出状态控制
0：输出 0
1：输出 1

Bit 1　**HCTDn**：GCT 输出状态控制
0：输出 0
1：输出 1

Bit 0　**HCBDn**：GCB 输出状态控制
0：输出 0
1：输出 1

● **HCHK_NUM 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	HCK_N4	HCK_N3	HCK_N2	HCK_N1	HCK_N0
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	0	0	0	0	0

Bit 7~5　未定义，读为“0”

Bit 4~0　**HCK_N4~HCK_N0**：霍尔噪声滤波器检查次数

● **HNF_MSEL 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	HNF_EN	HFR_SEL2	HFR_SEL1	HFR_SEL0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4　未定义，读为“0”

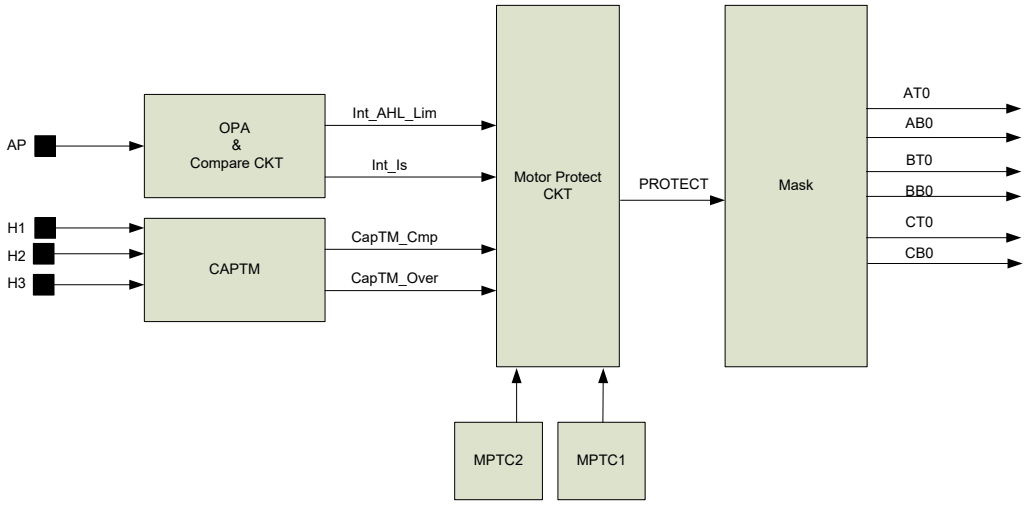
Bit 3　**HNF_EN**：霍尔噪声滤波器使能位
0：除能 (旁路)
1：使能

Bit 2~0　**HFR_SEL2~HFR_SEL0**：霍尔噪声滤波器时钟源选择
000：$f_{SYS}/2$
001：$f_{SYS}/4$
010：$f_{SYS}/8$
011：$f_{SYS}/16$
100：$f_{SYS}/32$

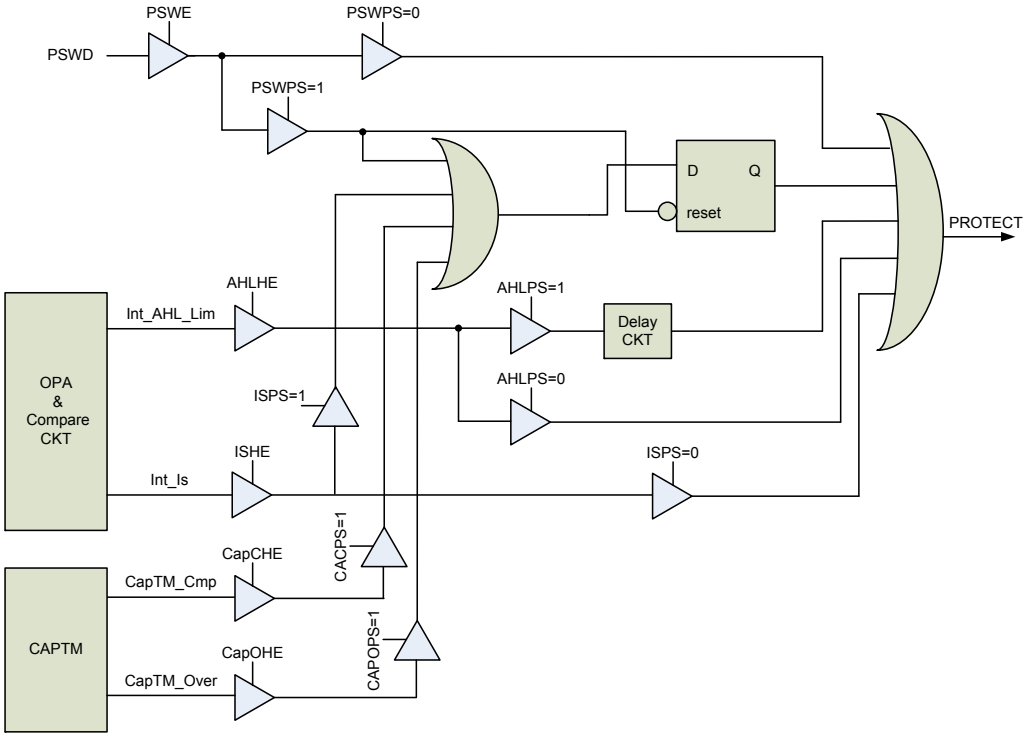
101：$f_{SYS}/64$
110：$f_{SYS}/128$
111：未使用

马达保护功能

马达运转一般需要较大电流，所以需要保护电路以防过电流和马达堵转等问题，减少马达破坏或安全原因。该单片机包括一些安全和保护特性。



保护电路方框图



保护功能控制

马达保护功能介绍

该单片机提供 3 种侦测马达堵转或过流保护电路，可即时关闭马达，以避免马达被烧毁或提供额外安全保护。

保护特性有：

- 堵转检查功能
- 过电流保护
- 软件关闭马达

当马达保护电路启动时，PROTECT=1，外部的门驱动晶体管对可选择刹车模式 (上臂关闭，下臂开启) 或是空转模式 (上臂关闭，下臂关闭)，由 MCF 寄存器的 FMOS 位决定。

马达保护电路有两种模式可以选择，一种是 Fault 模式，一种是 Pause 模式，由 MPTC2 寄存器决定。Fault 模式：保护功能启动由触发源状态启动决定；保护功能结束由触发源状态解除决定。Pause 模式：保护功能启动由触发源决定，保护功能结束都由软件决定。

电流保护功能

该单片机包含一个放大器，12 位 A/D 转换器、8 位 D/A 转换器以测量马达电流以及监控马达驱动过程中有无过电流的情形发生。如果检测到过流情况，应实时关闭马达外部驱动电路，以避免马达被烧毁。

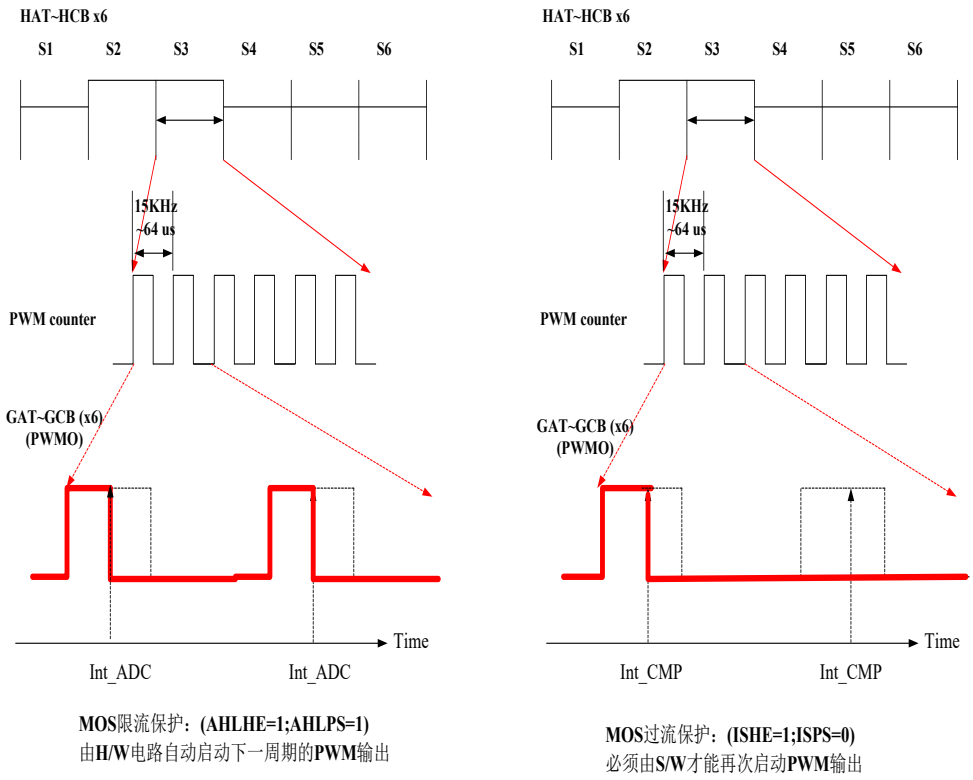
由于马达驱动过程 PCB 系统上噪声较大，加上使用内建 OPA 放大电路会将噪声放大，易导致误触发，故电流保护机制限制采用 fault mode 来做。

MOS 限电流保护机制 Int_AHL_Li：当 AHLHE=0，除能 H/W 模式；当 AHLHE=1，使能 H/W 模式。限电流电路为纯硬件电路，且 A/D 转换器通道选择必须选到 OPA 才有效。如果在系统启动期间发生过电流情况，必须除能限电流电路。

AHLPS=0：保护电路立即根据 Int_AHL_Lim 响应输出 PWM

AHLPS=1：保护电路会根据 Int_AHL_Lim 响应立即关闭 PWM 输出，但当保护电路解除保护时，PWM 输出响应会等待到下一个 PWM 周期到来才响应。

MOS 过电流保护机制 Int_Is：当 ISHE=0，除能 H/W 模式；当 ISHE=1，使能 H/W 模式。当 ISPS=0，选择 Fault 模式。当 ISPS=1，选择 Pause 模式。



过电流

马达堵转检测功能

针对霍尔传感器 3 相 BLDC 应用，16 位的 CAPTM 定时器用来监控 H1，H2 和 H3 输入以判断转子的移动速度。通过寄存器 CAPTMAH/CAPTMAL 选择过流信号，侦测霍尔传感器输入 H1，H2 和 H3，以监控转子速度。若发生过流现象，产生中断 CapTM_Cmp 或 CapTM_Over。可参考 CAPTM 章节。

CapTM_Cmp 堵转检查机制：当 CapCHE=0 时，除能 H/W 模式；当 CapCHE=1 时，使能 H/W 模式。

马达堵转检查机制限制采用 Pause mode 来做。当 CAPCPS=1 时，选择 Pause 模式。

CapTM_Over 堵转检查机制：当 CapOHE=0 时，除能 H/W 模式；当 CapOHE=1 时，使能 H/W 模式。当 CAPOPS=1 时，选择 Pause 模式。

马达保护电路相关寄存器介绍

寄存器 MPTC1 和 MPTC2 均为马达保护控制寄存器。

● MPTC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PSWD	PSWE	CapOHE	CapCHE	ISHE	AHLHE	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	—	—
POR	0	0	0	0	0	0	—	—

Bit 7 **PSWD**：保护 S/W 模式数据
0：PSWD=0
1：PSWD=1

Bit 6 **PSWE**：保护 S/W 模式使能位
0：除能
1：使能

Bit 5 **CapOHE**：CapTM_Over H/W 模式使能位
0：除能
1：使能

Bit 4 **CapCHE**：CapTM_Cmp H/W 模式使能位
0：除能
1：使能

Bit 3 **ISHE**：Int_Is H/W 模式使能位
0：除能
1：使能

Bit 2 **AHLHE**：Int_AHL_Lim H/W 模式使能位
0：除能
1：使能

Bit 1~0 未定义，读为“0”

● MPTC2 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	PSWPS	AHLPS	ISPS	CAPCPS	CAPOPS
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	1	0	0	1	1

Bit 7~5 未定义，读为“0”

Bit 4 **PSWPS**：Pause/Fault 模式选择
0：选择 Fault 模式
1：选择 Pause 模式

Bit 3 **AHLPS**：Int_AHL_Lim Pause/Fault 模式选择
0：保护电路立即根据 Int_AHL_Lim 响应输出 PWM
1：保护电路会根据 Int_AHL_Lim 响应立即关闭 PWM 输出，但当保护电路解除保护时，PWM 输出响应会等待到下一个 PWM 周期到来才响应。

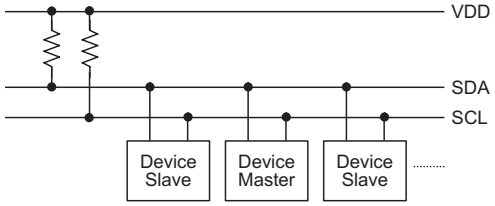
Bit 2 **ISPS**：Int_Is Pause/Fault 模式选择
0：选择 Fault 模式
1：选择 Pause 模式

Bit 1 **CAPCPS**：CapTM_Cmp Pause/Fault 模式选择
0：未定义，不被选择
1：选择 Pause 模式

Bit 0 **CAPOPS**：CapTM_Over Pause/Fault 模式选择
0：未定义，不被选择
1：选择 Pause 模式

I²C 接口

I²C 可以和传感器，EEPROM 内存等外部硬件接口进行通信。最初是由飞利浦公司研制，是适用于同步串行数据传输的双线式低速串行接口。I²C 接口具有两线通信，非常简单的通信协议和在同一总线上和多个设备进行通信的能力的优点，使之在很多的应用场合中大受欢迎。



I²C 主从总线连接图

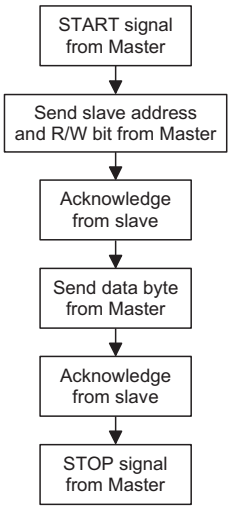
I²C 接口操作

I²C 串行接口是一个双线的接口，有一条串行数据线 SDA 和一条串行时钟线 SCL。由于可能有多个设备在同一条总线上相互连接，所以这些设备的输出都是开漏型输出。因此应在这些输出口上都应加上拉电阻。应注意的是：I²C 总线上的每个设备都没有选择线，但分别与唯一的地址一一对应，用于 I²C 通信。

如果有两个设备通过双向的 I²C 总线进行通信，那么就存在一个主机和一个从机。主机和从机都可以用于传输和接收数据，但只有主机才可以控制总线动作。那些处于从机模式的设备，要在 I²C 总线上传输数据只有两种方式，一是从机发送模式，二是从机接收模式。

在 I²C 通信过程中，建议用户不要通过应用程序让单片机进入 HALT。

如果引脚被配置为 I²C 接口中 SDA 或 SCL 功能，该引脚必须配置为开集输入 / 输出口，且通过相应的上拉电阻控制寄存器使能该引脚的上拉电阻功能。



I²C 寄存器

I²C 总线的四个控制寄存器是 IICC0，IICC1，IICA 和 I2CTOC 及一个数据寄存器 IICD。IICD 寄存器用于存储正在传输和接收的数据，当单片机将数据写入 I²C 总线之前，实际将被传输的数据存放在寄存器 IICD 中。从 I²C 总线接收到数据之后，单片机就可以从寄存器 IICD 中读取这个数据。I²C 总线上的所有传输或接收到的数据都必须通过寄存器 IICD。

寄存器名称	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IICC0	—	—	—	—	I2CDBNC1	I2CDBNC0	I2CEN	—
IICC1	IICHCF	IICHAAS	IICHBB	IICHTX	IICTXAK	IICSRW	IICRNIC	IICRXAK
IICD	IICDD7	IICDD6	IICDD5	IICDD4	IICDD3	IICDD2	IICDD1	IICDD0
IICA	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	—
I2CTOC	I2CTOEN	I2CTOF	I2CTOS5	I2CTOS4	I2CTOS3	I2CTOS2	I2CTOS1	I2CTOS0

I²C 寄存器列表

IICC0 寄存器

Bit	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	—	—	—	—	I2CDBNC1	I2CDBNC0	I2CEN	—
R/W	—	—	—	—	R/W	R/W	R/W	—
POR	—	—	—	—	0	0	0	—

Bit 7~4　未定义，读为“0”

Bit 3~2　**I2CDBNC1~I2CDBNC0**：I²C 抖动时间选择位
00：无抖动
01：2 个系统时钟的抖动
10：4 个系统时钟的抖动
11：4 个系统时钟的抖动

Bit 1　**I2CEN**：I²C 使能位
0：除能
1：使能

Bit 0　未定义，读为“0”

I²C 功能可以通过相应引脚共用功能控制位使能或除能，该引脚共用功能控制位用来选择与 I/O 共用引脚的 SDA 和 SCL 引脚功能。当与 SDA 和 SCL 共用引脚的 I/O 端口通过引脚共用控制位选择其他功能，则 I²C 功能除能且其工作电流将减少到最小值。相反，当 I/O 端口引脚通过引脚共用功能能选择 SDA 和 SCL 引脚时，I²C 功能使能。

IICC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	IICHCF	IICHAAS	IICHBB	IICHTX	IICTXAK	IICSRW	IICRNIC	IICRXAK
R/W	R	R	R	R/W	R/W	R	R/W	R
POR	1	0	0	0	0	0	0	1

Bit 7 **IICHCF**：I²C 总线数据传输完成标志位
0：数据正在被传输
1：8 位数据传输完成
IICHCF 是数据传输标志位。数据正在传输时该位为低。当 8 位数据传输完成时，此位为高并产生一个中断。

Bit 6 **IICHAAS**：I²C 总线地址匹配标志位
0：地址不匹配
1：地址匹配
此标志位用于决定从机地址是否与主机发送地址相同。若地址匹配此位为高，否则此位为低。

Bit 5 **IICHBB**：I²C 总线忙标志位
0：I²C 总线闲
1：I²C 总线忙
当检测到 START 信号时 I²C 忙，此位变为高电平。当检测到 STOP 信号时 I²C 总线停止，该位变为低电平。

Bit 4 **IICHTX**：从机处于发送或接收模式标志位
0：从机处于接收模式
1：从机处于发送模式

Bit 3 **IICTXAK**：I²C 总线发送确认标志位
0：从机发送确认标志
1：从机没有发送确认标志
单片机接收 8 位数据之后会将该位在第九个时钟传到总线上。如果单片机想要接收更多的数据，则应在接收数据之前将此位设置为“0”。

Bit 2 **IICSRW**：I²C 从机读 / 写标志位
0：从机应处于接收模式
1：从机应处于发送模式
IICSRW 位是从机读写标志位。决定主机是否希望传输或接收来自 I²C 总线的数据。当传输地址和从机的地址相同时，IICHAAS 位会被设置为高，主机将检测 IICSRW 位来决定进入发送模式还是接收模式。如果 IICSRW 位为高时，主机会请求从总线上读数据，此时设备处于传输模式。当 IICSRW 位为“0”时，主机往总线上写数据，设备处于接收模式以读取该数据。

Bit 1 **IICRNIC**：I²C 内部时钟使用选择位
0：I²C 使用内部时钟运行
1：I²C 无需使用内部时钟运行
在休眠、空闲 (慢速)、正常 (慢速) 模式下，I²C 模块无需使用内部时钟运行，如果 I²C 中断使能，将产生中断，此时 MCU 应设成正常模式下，并将 IICRCIN 位设为“0”。

Bit 0 **IICRXAK**：I²C 总线接收确认标志位
0：从机接收到确认标志
1：从机没有接收到确认标志
IICRXAK 位是接收确认标志位。如果 IICRXAK 位被重设为“0”即 8 位数据传输之后，设备在第九个时钟有接受到一个正确的确认位。如果单片机处于发送状态，发送方会检查 IICRXAK 位来判断接收方是否愿意继续接收下一个字节。因此直到 IICRXAK 为“1”时，传输方停止发送数据。这时，传输方将释放 SDA 线，主机发出停止信号。

IICD 寄存器用于存储发送和接收的数据。在单片机尚未将数据写入到 I²C 总线中时，要传输的数据应存在 IICD 中。I²C 总线接收到数据之后，单片机就可以从 IICD 寄存器中读取。

IICD 寄存器

Bit	7	6	5	4	3	2	1	0
Name	IICDD7	IICDD6	IICDD5	IICDD4	IICDD3	IICDD2	IICDD1	IICDD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

“x”：未知

Bit 7~0 **IICDD7~IICDD0**：I²C 数据缓冲区 bit 7~bit 0

IICA 寄存器

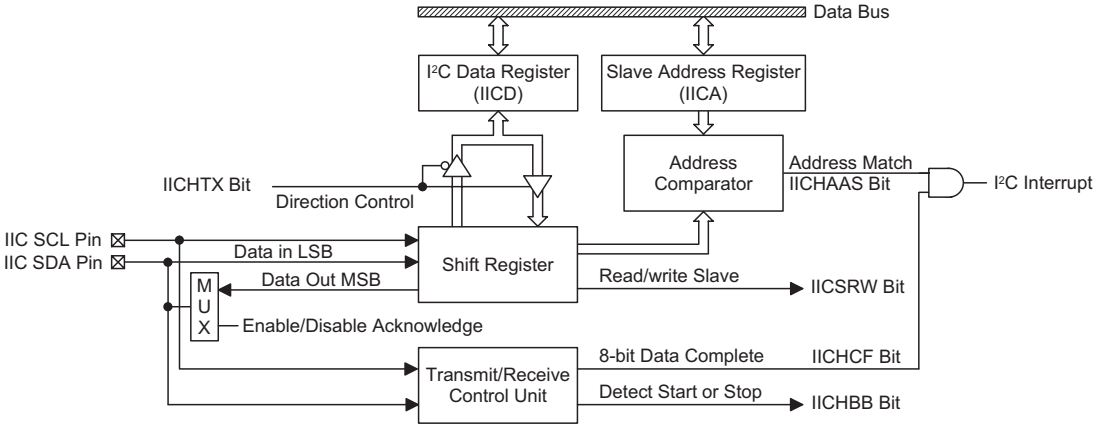
Bit	7	6	5	4	3	2	1	0
Name	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	—
POR	x	x	x	x	x	x	x	—

“x”：未知

Bit 7~1 **IICA6~IICA0**：I²C 从机地址位

IICA6~IICA0 是 I²C 从机地址位 bit 6~bit 0 位。IICA 寄存器用于存放 7 位从机地址，寄存器 IICA 中的第 7~1 位是单片机的从机地址，位 0 未定义。如果接至 I²C 的主机发送处的地址和寄存器 IICA 中存储的地址相符，那么就选中了这个从机。

Bit 0 未定义，读为“0”

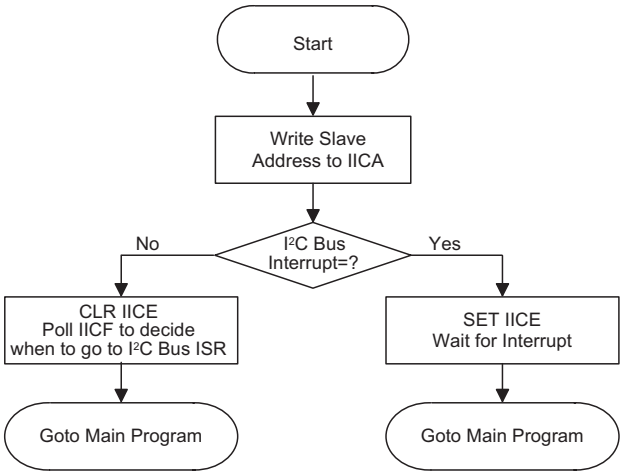


I²C 方框图

I²C 总线通信

I²C 总线上的通信需要四步完成，一个起始信号，一个从机地址发送，一个数据传输，还有一个停止信号。当起始信号被写入 I²C 总线时，总线上的所有从机都会接收到这个起始信号并且被通知总线上会即将有数据到达。数据的前 7 位是从机地址，高位在前，低位在后。如果发出的地址和从机地址匹配，IICC1 寄存器的 IICHAAS 位会被置位，同时产生 I²C 中断。进入中断服务程序后，系统要检测 IICHAAS 位，以判断 I²C 总线中断是来自从机地址匹配，还是来自 8 位数据传递完毕。在数据传递中，注意的是，在 7 位从机地址被发送后，接下来的一位，即第 8 位，是读 / 写控制位，该位的值会反映到 IICSRW 位中。从机通过检测 IICSRW 位以确定主控制器是要进入发送模式还是接收模式。在 I²C 总线开始传送数据前，需要先初始化 I²C 总线，初始化 I²C 总线步骤如下：

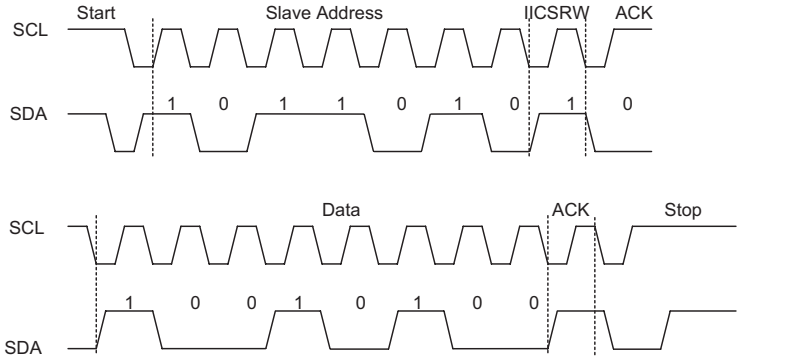
- 步骤 1
  设置引脚共用 I/O 口为 I²C 引脚功能。(SCL 和 SDA)
- 步骤 2
  设置 IICC0 寄存器中 I2CEN 位为“1”，以使能 I²C 总线
- 步骤 3
  向 I²C 总线地址寄存器 IICA 写入从机地址。
- 步骤 4
  设置 IICE 位，以使能 I²C 中断。



I²C 总线初始化流程图

I²C 总线起始信号

起始信号只能由连接 I²C 总线主机产生，而不是由只做从机设备产生。I²C 总线上的所有从机都可以侦测到起始信号。如果有从机侦测到起始信号，则表明 I²C 总线处于忙碌状态，并会置位 IICHBB。起始信号是指在 SCL 为高电平时，SDA 线上发生从高到低的电平变化。



S=Start (1 bit)
SA=Slave Address (7 bits)
SR=IICSRW bit (1 bit)
M=Slave device send acknowledge bit (1 bit)
D=Data (8 bits)
A=ACK (IICRXAK bit for transmitter, IICTXAK bit for receiver 1 bit)
P=Stop (1 bit)

S	SA	SR	M	D	A	D	A	.....	S	SA	SR	M	D	A	D	A	.....	P

注：* 当从机地址匹配时，单片机必须选择设置为发送模式还是接收模式。若设置为发送模式，需写数据至 IICD 寄存器；若设置为接收模式，需立即从 IICD 寄存器中虚读数据以释放 SCL 线。

I²C 通信时序图

从机地址

I²C 总线上的所有从机都会侦测由主机发出的起始信号。发送起始信号后，紧接着主机会发送从机地址以选择要进行数据传输的从机。所有在 I²C 总线上的从机接收到 7 位地址数据后，都会将其与各自内部的地址进行比较。如果从机从主机上接收到的地址与自身内部的地址相匹配，则会产生一个 I²C 总线中断信号。地址位接下来的一位为读 / 写状态位 (即第 8 位)，将被保存到 IICC1 寄存器的 IICSRW 位，随后发出一个低电平应答信号 (即第 9 位)。当单片机从机的地址匹配时，会将状态标志位 IICHAAS 置位。

I²C 总线有两个中断源，当程序运行至中断服务子程序时，通过检测 IICHAAS 位以确定 I²C 总线中断是来自从机地址匹配，还是来自 8 位数据传递完毕。当是从机地址匹配发生中断时，则从机或是用于发送模式并将数据写进 IICD 寄存器，或是用于接收模式并从 IICD 寄存器中读取空值以释放 SCL 线。

I²C 总线读 / 写信号

IICC1 寄存器的 IICSRW 位用来表示主机是要从 I²C 总线上读取数据还是要将数据写到 I²C 总线上。从机则通过检测该位以确定自己是作为发送方还是接收方。当 IICSRW 置 1，表示主机要从 I²C 总线上读取数据，从机则作为发送方，将数据写到 I²C 总线；当 IICSRW 清零，表示主机要写数据到 I²C 总线上，从机则做为接收方，从 I²C 总线上读取数据。

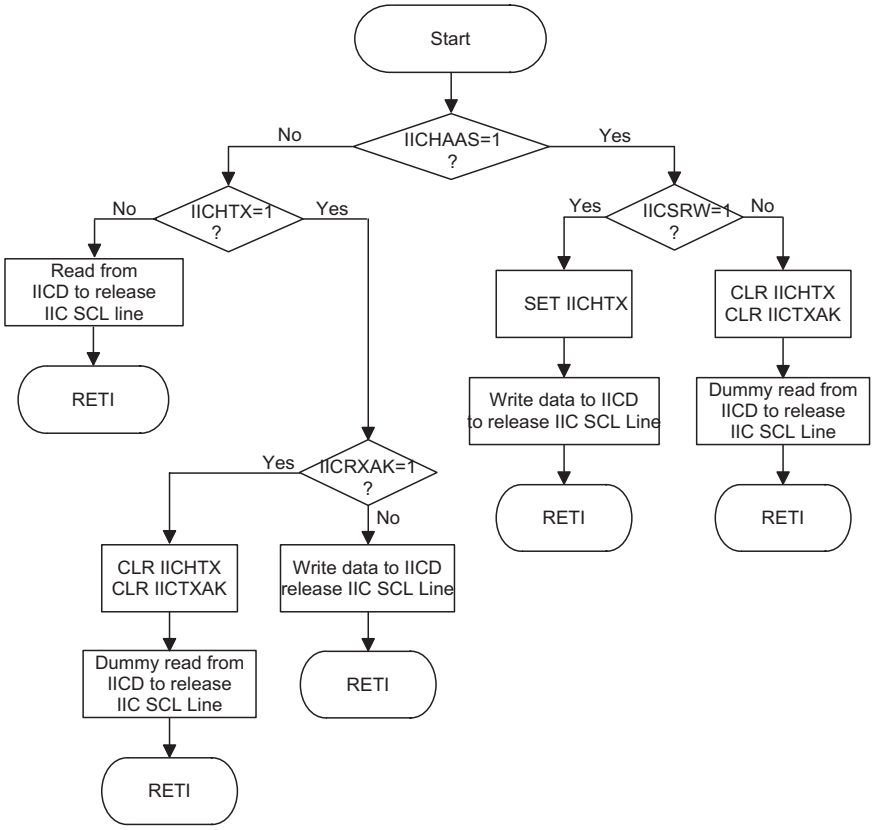
I²C 总线从机地址确认信号

主机发送呼叫地址后，当 I²C 总线上的任何从机内部地址与其匹配时，会发送一个应答信号。此应答信号会通知主机有从机已经接收到了呼叫地址。如果主机没有收到应答信号，则主机必须发送停止 (STOP) 信号以结束通信。当 IICHAAS 为高时，表示从机接收到的地址与自己内部地址匹配，则从机需检查 IICSRW 位，以确定自己是作为发送方还是作为接收方。如果 IICSRW 位为高，从机须设置成发送方，这样会置位 IICC1 寄存器的 IICHTX 位。如果 IICSRW 位为低，从机须设置成接收方，这样会清零 IICC1 寄存器的 IICHTX 位。

I²C 总线数据和确认信号

在从机确认接收到从地址后，会进行 8 位宽度的数据传输。这个数据传输顺序是的高位在前，低位在后。接收方在接收到 8 位数据后必须发出一个应答信号 (“0”) 以继续接收下一个数据。如果发送方没接收到应答信号，发送方将释放 SDA 线，同时，主机将发出 STOP 信号以释放 I²C 总线。所传送的数据存储在 IICD 寄存器中。如果设置成发送方，从机必须先将欲传输的数据写到 IICD 寄存器中；如果设置成接收方，从机必须从 IICD 寄存器读取数据。

当接收器想要继续接收下一个数据时，必须在第 9 个时钟发出应答信号 (IICTXAK)。被设为发送方的从机将检测寄存器 IICC1 中的 IICRXAK 位以判断是否传输下一个字节的数据，如果单片机不传输下一个字节，那么它将释放 SDA 线并等待接收主机的停止信号。

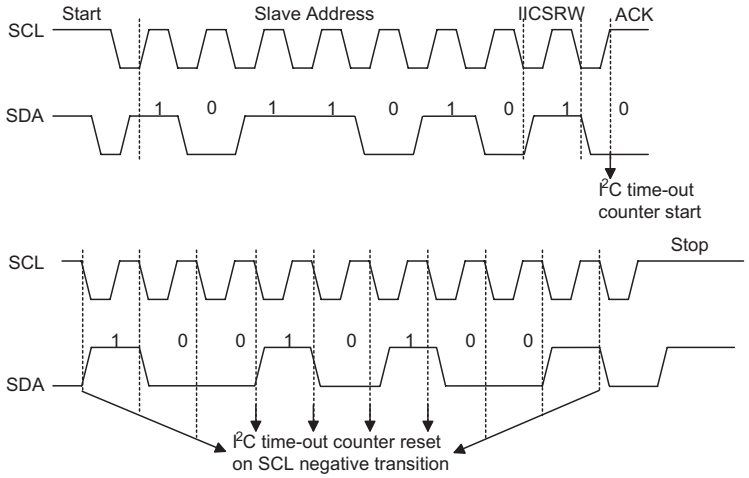


I²C 总线 ISR 流程图

I²C 超时控制

为了减少由于接收错误时钟源而产生的 I²C 锁定问题，系统提供了超时功能。在固定时间内如果 I²C 总线未接收到时钟源，则 I²C 电路和寄存器将会复位。

超时计数器在 I²C 总线接收到“START”信号和“地址匹配”条件时，超时计数器开始计数，并在 SCL 下降沿处清零。在下一个 SCL 下降沿来临之前，如果等待时间大于 I2CTOC 寄存器设定的超时时间，则会发生超时现象。当 I²C “STOP”条件发生时，超时计数器将停止计数。



I²C 超时时序图

当 I²C 超时计数器溢出时，计数器将停止计数，I2CTOEN 位被清零，且 I2CTOF 位被置高以表明超时计数器中断发生。超时计数器中断使用的也是 I²C 中断向量。当 I²C 超时发生时，I²C 内部电路会被复位，寄存器也将发生如下复位情况。

寄存器	I²C 超时发生后
IICD，IICA，IICC0	保持不变
IICC1	复位至 POR

超时发生后的 I²C 寄存器

I2CTOF 标志位可由应用程序清零。64 个超时时钟周期可由 I2CTOC 寄存器里的相应位来设置。超时时间的计算方法如下：

$$((1\sim64)\times32)/f_{SUB}$$

这里给出了一个 1ms~64ms 的范围。注意，LIRC 振荡器是一直处于使能状态。

I2CTOC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	I2CTOEN	I2CTOF	I2CTOS5	I2CTOS4	I2CTOS3	I2CTOS2	I2CTOS1	I2CTOS0
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 **I2CTOEN**：I²C 超时控制位
0：除能
1：使能

Bit 6 **I2CTOF**：I²C 超时标志位
0：没有超时
1：超时发生

Bit 5~0 **I2CTOS5~I2CTOS0**：I²C 超时时间定义位
I²C 超时时钟源是 $f_{SUB}/32$
I²C 超时时间计算方法：[I2CTOS5:I2CTOS0]+1) × ($32/f_{SUB}$)

UART 模块串行接口

UART 模块特性

- 全双工通用异步接收器 / 发送器
- 8 位或 9 位传输格式
- 奇校验、偶校验或无校验
- 1 位或 2 位停止位
- 8 位预分频的波特率发生器
- 奇偶、帧、噪声和溢出检测
- 支持地址匹配中断 (最后一位 =1)
- 独立的发送和接收使能
- 2-byte FIFO 接收缓冲器
- 发送和接收中断源：
  - ♦ 发送器为空
  - ♦ 发送器空闲
  - ♦ 接收完成
  - ♦ 接收器溢出
  - ♦ 地址匹配

UART 模块概述

该单片机具有一个全双工的异步串行通信接口——UART，可以很方便的与其它具有串行口的芯片通信。UART 具有许多功能特性，发送或接收串行数据时，将数据组成一个 8 位或 9 位的数据块，连同数据特征位一并传输。具有检测数据覆盖或帧错误等功能。UART 功能占用一个内部中断向量，当接收到数据或数据发送结束，触发 UART 中断。

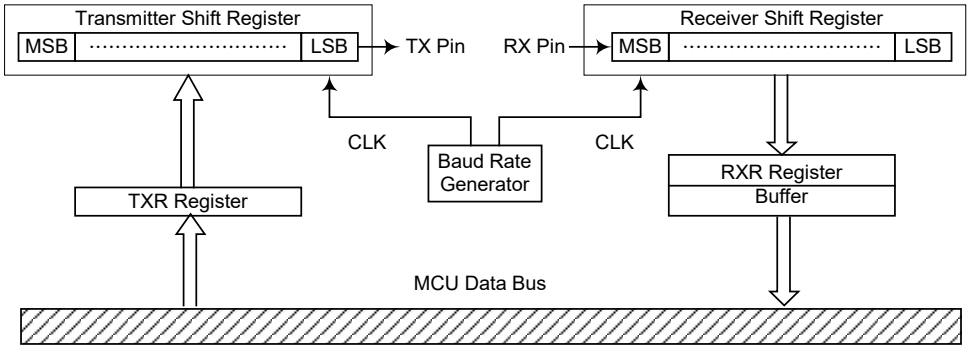
UART 外部引脚接口

内部 UART 有两个外部引脚 TX 和 RX，可与外部串行接口进行通信。TX 引脚为 UART 发送脚，当 UCR2 控制寄存器的 TXEN 位为 0 时，该引脚未配置为 UART 发送脚，此时可设置为 I/O 脚或其它与其共用引脚的其它引脚功能。RX 引脚为 UART 接收脚，当 UCR2 控制寄存器的 RXEN 位为 0 时，该引脚未配置为 UART 接收脚，此时可设置为 I/O 脚或其它与其共用引脚的其它引脚功能。若 UARTEN 位和 TXEN/RXEN 位均为“1”时，TX 脚为发送串行数据输出脚，RX 为接收串行数据输入脚。若 UARTEN 位或 TXEN/RXEN 位为“0”时，TX 和 RX 均用作普通 I/O 口或其它引脚共用功能。

UART 数据传输方案

下图显示了 UART 的整体结构。需要发送的数据首先写入 TXR 寄存器，接着此数据被传输到发送移位寄存器 TSR 中，然后在波特率发生器的控制下将 TSR 寄存器中数据一位位地移到 TX 引脚上，低位在前。TXR 寄存器被映射到单片机的数据存储器中，而发送移位寄存器没有实际地址，所以发送移位寄存器不可直接操作。

数据在波特率发生器的控制下，低位在前高位在后，从外部引脚 RX 进入接收移位寄存器 RSR。当数据接收完成，数据从接收移位寄存器移入可被用户程序操作的 RXR 寄存器中。RXR 寄存器被映射到单片机数据存储器中，而接收移位寄存器没有实际地址，所以接收移位寄存器不可直接操作。需要注意的是，上述发送寄存器 TXR 和接收寄存器 RXR，其实是共享一个地址的数据寄存器 TXR_RXR 寄存器。



UART 数据传输方案

UART 状态和控制寄存器

与 UART 功能相关的有五个寄存器——控制 UART 模块整体功能的 USR、UCR1 和 UCR2 寄存器，控制波特率的 BRG 寄存器，管理发送和接收数据的数据寄存器 TXR_RXT。

寄存器名称	位							
	7	6	5	4	3	2	1	0
USR	PERR	NF	FERR	OERR	RIDLE	RXIF	TIDLE	TXIF
UCR1	UARTEN	BNO	PREN	PRT	STOPS	TXBRK	RX8	TX8
UCR2	TXEN	RXEN	BRGH	ADDEN	WAKE	RIE	TIIE	TEIE
BRG	BRG7	BRG6	BRG5	BRG4	BRG3	BRG2	BRG1	BRG0
TXR_RXR	TXRXD7	TXRXD6	TXRXD5	TXRXD4	TXRXD3	TXRXD2	TXRXD1	TXRXD0

寄存器 USR 是 UART 的状态寄存器，可以通过程序读取。所有 USR 位是只读的。详细解释如下：

USR 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PERR	NF	FERR	OERR	RIDLE	RXIF	TIDLE	TXIF
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	1	0	1	1

Bit 7 **PERR**：奇偶校验出错标志位
0：奇偶校验正确
1：奇偶校验出错

PERR 是奇偶校验出错标志位。若 PERR=0，奇偶校验正确；若 PERR=1，接收到的数据奇偶校验出错。只有使能了奇偶校验此位才有效。可使用软件清除该标志位，即先读取 USR 寄存器再读 RXR 寄存器来清除此位。

Bit 6 **NF**：噪声干扰标志位
0：没有受到噪声干扰
1：受到噪声干扰

NF 是噪声干扰标志位。若 NF=0，没有受到噪声干扰；若 NF=1，UART 接收数据时受到噪声干扰。它与 RXIF 在同周期内置位，但不会与溢出标志位同时置位。可使用软件清除该标志位，即先读取 USR 寄存器再读 RXR 寄存器将清除此标志位。

Bit 5 **FERR**：帧错误标志位
0：无帧错误发生
1：有帧错误发生

FREE 是帧错误标志位。若 FREE=0，没有帧错误发生；若 FREE=1，当前的数据发生了帧错误。可使用软件清除该标志位，即先读取 USR 寄存器再读 RXR 寄存器来清除此位。

Bit 4 **OERR**：溢出错误标志位
0：无溢出错误发生
1：有溢出错误发生

OERR 是溢出错误标志位，表示接收缓冲器是否溢出。若 OERR=0，没有溢出错误；若 OERR=1，发生了溢出错误，它将影响下一组数据的接收。可通过软件清除该标志位，即先读取 USR 寄存器再读 RXR 寄存器将清除此标志位。

Bit 3 **RIDLE**：接收状态标志位
0：正在接收数据
1：接收器空闲

RIDLE 是接收状态标志位。若 RIDLE=0，正在接收数据；若 RIDLE=1，接收器空闲。在接收到停止位和下一个数据的起始位之间，RIDLE 被置位，表明 UART 空闲，RX 脚处于逻辑高状态。

Bit 2 **RXIF**：接收寄存器状态标志位
0：RXR 寄存器为空
1：RXR 寄存器含有有效数据

RXIF 是接收寄存器状态标志位。当 RXIF=0，RXR 寄存器为空；当 RXIF=1，RXR 寄存器接收到新数据。当数据从移位寄存器加载到 RXR 寄存器中，如果 UCR2 寄存器中的 RIE=1，则会触发中断。当接收数据时检测到一个或多个错误时，相应的标志位 NF、FERR 或 PERR 会在同一周期内置位。读取 USR 寄存器再读 RXR 寄存器，如果 RXR 寄存器中没有新的数据，那么将清除 RXIF 标志。

Bit 1 **TIDLE**：数据发送完成标志位
0：数据传输中
1：无数据传输

TIDLE 是数据发送完成标志位。若 TIDLE=0，数据传输中。当 TXIF=1 且数据发送完毕或者暂停字被发送时，TIDLE 置位。TIDLE=1，TX 引脚空闲且处于逻辑高状态。读取 USR 寄存器再写 TXR 寄存器将清除 TIDLE 位。数据字符或暂停字就绪时，不会产生该标志位。

Bit 0 **TXIF**：发送数据寄存器 TXR 状态位
0：数据还没有从缓冲器加载到移位寄存器中
1：数据已从缓冲器加载到移位寄存器中 (TXR 数据寄存器为空)
TXIF 是发送数据寄存器为空标志位。若 TXIF=0，数据还没有从缓冲器加载到移位寄存器中；若 TXIF=1，数据已从缓冲器中加载到移位寄存器中。读取 USR 存器再写 TXR 寄存器将清除 TXIF。当 TXEN 被置位，由于发送缓冲器未满，TXIF 也会被置位。

UCR1 寄存器

UCR1 和 UCR2 是 UART 的两个控制寄存器，用来定义各种 UART 功能，例如 UART 的使能与除能、奇偶校验控制和传输数据的长度等等。

详细解释如下：

Bit	7	6	5	4	3	2	1	0
Name	UARTEN	BNO	PREN	PRT	STOPS	TXBRK	RX8	TX8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	W
POR	0	0	0	0	0	0	x	0

“x”：未知

Bit 7 **UARTEN**：UART 功能使能位
0：UART 除能，TX 和 RX 脚用作普通 I/O 口或其它引脚共用功能
1：UART 使能，TX 和 RX 脚作为 UART 功能引脚
此位为 UART 的使能位。UARTEN=0，UART 除能，RX 和 TX 可用作普通的输入输出口或其它引脚共用功能；UARTEN=1，UART 使能，TX 和 RX 将分别由 TXEN 和 RXEN 控制。当 UART 被除能将清除缓冲器，所有缓冲器中的数据将被忽略，另外波特率计数器、错误和状态标志位被复位，TXEN、RXEN、TXBRK、RXIF、OERR、FERR、PERR 和 NF 清零而 TIDLE、TXIF 和 RIDLE 置位，UCR1、UCR2 和 BRG 寄存器中的其它位保持不变。若 UART 工作时 UARTEN 清零，所有发送和接收将停止，模块也将复位成上述状态。当 UART 再次使能时，它将在上次配置下重新工作。

Bit 6 **BNO**：发送数据位数选择位
0：8-bit 传输数据
1：9-bit 传输数据
BNO 是发送数据位数选择位。BNO=1，传输数据为 9 位；BNO=0，传输数据为 8 位。若选择了 9 位数据传输格式，RX8 和 TX8 将分别存储接收和发送数据的第 9 位。

Bit 5 **PREN**：奇偶校验使能位
0：奇偶校验除能
1：奇偶校验使能
此位为奇偶校验使能位。PREN=1，使能奇偶校验；PREN=0，除能奇偶校验。

Bit 4 **PRT**：奇偶校验选择位
0：偶校验
1：奇校验
奇偶校验选择位。PRT=1，奇校验；PRT=0，偶校验。

Bit 3 **STOPS**：停止位的长度选择位
0：有一位停止位
1：有两位停止位
此位用来设置停止位的长度。STOP=1，有两位停止位；STOP=0，只有一位停止位。

Bit 2 **TXBRK**：暂停字发送控制位
0：没有暂停字要发送
1：发送暂停字
TXBRK 是暂停字发送控制位。TXBRK=0，没有暂停字要发送，TX 引脚正常操作；TXBRK=1，将会发送暂停字，发送器将发送逻辑“0”。若 TXBRK 为高，缓冲

器中数据发送完毕后，发送器将至少保持 13 位宽的低电平直至 TXBRK 复位。

Bit 1 **RX8**：接收 9-bit 数据传输格式中的第 8 位 (只读)
此位只有在传输数据为 9 位的格式中有效，用来存储接收数据的第 9 位。BNO 是用来控制传输位数是 8 位还是 9 位。

Bit 0 **TX8**：发送 9-bit 数据传输格式中的第 8 位 (只写)
此位只有在传输数据为 9 位的格式中有效，用来存储发送数据的第 9 位。BNO 是用来控制传输位数是 8 位还是 9 位。

UCR2 寄存器

UCR2 是 UART 的第二个控制寄存器，它的主要功能是控制发送器、接收器以及各种 UART 中断源的使能或除能。它也可用来控制波特率，使能接收唤醒和地址侦测。

详细解释如下：

Bit	7	6	5	4	3	2	1	0
Name	TXEN	RXEN	BRGH	ADDEN	WAKE	RIE	TIIE	TEIE
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 **TXEN**：UART 发送使能位
0：UART 发送除能
1：UART 发送使能
此位为发送使能位。TXEN=0，发送将被除能，发送器立刻停止工作。另外缓冲器将被复位，此时 TX 引脚作为普通的输入输出 / 端口使用或引脚共用功能。若 TXEN=1 且 UARTEN=1，则发送将被使能，TX 引脚将由 UART 来控制。在数据传输时清除 TXEN 将中止数据发送且复位发送器，此时 TX 引脚作为普通的输入 / 输出端口使用。

Bit 6 **RXEN**：UART 接收使能位
0：UART 接收除能
1：UART 接收使能
此位为接收使能位。RXEN=0，接收将被除能，接收器立刻停止工作。另外缓冲器将被复位，此时 RX 引脚可作为普通输入 / 输出端口使用或其它引脚共用功能。若 RXEN=1 且 UARTEN=1，则接收将被使能，RX 引脚将由 UART 来控制。在数据传输时清除 RXEN 将中止数据接收且复位接收器，此时 RX 引脚可作为普通输入输出端口使用。

Bit 5 **BRGH**：波特率发生器高低速选择位
0：低速波特率
1：高速波特率
此位为波特率发生器高低速选择位，它和 BRG 寄存器一起控制 UART 的波特率。BRGH=1，为高速模式；BRGH=0，为低速模式。

Bit 4 **ADDEN**：地址检测使能位
0：地址检测除能
1：地址检测使能
此位为地址检测使能和除能位。ADDEN=1，地址检测使能，此时数据的第 8 位 (BON=0) 或第 9 位 (BON=1) 为高，那么接到的是地址而非数据。若相应的中断使能且接收到的值最高位为 1，那么中断请求标志将会被置位，若最高位为 0，那么将不会产生中断且收到的数据也会被忽略。

Bit 3 **WAKE**：RX 脚下降沿唤醒功能使能位
0：RX 脚下降沿唤醒功能除能
1：RX 脚下降沿唤醒功能使能
此位为接收唤醒功能的使能和除能位。若 WAKE=1 且在 IDLE 或 SLEEP 模式下，RX 引脚的下降沿将唤醒单片机 (请参考不同暂停模式下 RX 引脚的唤醒功能)。若 WAKE=0 且在 IDLE 或 SLEEP 模式下，RX 引脚的任何边沿都不能唤

醒单片机。

Bit 2 **RIE**：接收中断使能位
0：接收中断除能
1：接收中断使能
此位为接收中断使能或除能位。若 RIE=1，当 OERR 或 RXIF 置位时，UART 的中断请求标志置位；若 RIE=0，UART 中断请求标志不受 OERR 和 RXIF 影响。

Bit 1 **TIIE**：发送器空闲中断使能位
0：发送器空闲中断除能
1：发送器空闲中断使能
此位为发送器空闲中断的使能或除能位。若 TIIE=1，当 TIDLE 置位时，UART 的中断请求标志置位；若 TIIE=0，UART 中断请求标志不受 TIDLE 的影响。

Bit 0 **TEIE**：发送寄存器为空中断使能位
0：发送寄存器为空中断除能
1：发送寄存器为空中断使能
此位为发送寄存器为空中断的使能或除能位。若 TEIE=1，当 TXIF 置位时，UART 的中断请求标志置位；若 TEIE=0，UART 中断请求标志不受 TXIF 的影响。

TXR_RXR 寄存器

Bit	7	6	5	4	3	2	1	0
Name	TXRXD7	TXRXD6	TXRXD5	TXRXD4	TXRXD3	TXRXD2	TXRXD1	TXRXD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

“x”：未知

Bit 7~0 **TXRXD7~TXRDX0**：UART 发送 / 接收数据位 Bit7~Bit0

波特率发生器

UART 自身具有一个波特率发生器，通过它可以设定数据传输速率。波特率是由一个独立的内部 8 位计数器产生，它由 BRG 寄存器和 UCR2 寄存器的 BRGH 位来控制。BRGH 是决定波特率发生器处于高速模式还是低速模式，从而决定计算公式的选用。BRG 寄存器的值 N 可根据下表中的公式计算，N 的范围是 0 到 255。

UCR2 的 BRGH 位	0	1
波特率 (BR)	$\frac{f_{SYS}}{[64(N+1)]}$	$\frac{f_{SYS}}{[16(N+1)]}$

为得到相应的波特率，首先需要设置 BRGH 来选择相应的计算公式从而算出 BRG 的值。由于 BRG 的值不连续，所以实际波特率和理论值之间有一个偏差。

下面举例怎样计算 BRG 寄存器中的值 N 和误差。

波特率和误差的计算

系统选用 4M 时钟频率且 BRGH=0，若期望的波特率为 4800，计算它的 BRG 寄存器的值 N，实际波特率和误差。

根据上表，波特率 $BR=\frac{f_{SYS}}{[64(N+1)]}$

转换后的公式 $N=\frac{f_{SYS}}{(BR\times 64)}-1$

带入参数 $N=\frac{4000000}{(4800\times 64)}-1=12.0208$

取最接近的值，十进制 12 写入 BRG 寄存器，实际波特率如下

$$BR=\frac{4000000}{[64(12+1)]}=4808$$

因此，误差 $=\frac{4808-4800}{4800}=0.16\%$

下面两表给出 BRGH 取不同值时的实际波特率和误差。

波特率 K/BPS	BRGH=0								
	f_{SYS}=4MHz			f_{SYS}=3.579545MHz			f_{SYS}=7.159MHz		
	BRG	Kbaud	误差 (%)	BRGn	Kbaud	误差 (%)	BRGn	Kbaud	误差 (%)
0.3	207	0.300	0.16	185	0.300	0.00	—	—	—
1.2	51	1.202	0.16	46	1.190	-0.83	92	1.203	0.23
2.4	25	2.404	0.16	22	2.432	1.32	46	2.380	-0.83
4.8	12	4.808	0.16	11	4.661	-2.90	22	4.863	1.32
9.6	6	8.929	-6.99	5	9.321	-2.90	11	9.332	-2.90
19.2	2	20.833	8.51	2	18.643	-2.90	5	18.643	-2.90
38.4	—	—	—	—	—	—	2	32.286	-2.90
57.6	0	62.500	8.51	0	55.930	-2.90	1	55.930	-2.90
115.2	—	—	—	—	—	—	0	111.859	-2.90

BRGH=0 时的波特率和误差

波特率 K/BPS	BRGH=1								
	f_{SYS}=4MHz			f_{SYS}=3.579545MHz			f_{SYS}=7.159MHz		
	BRG	Kbaud	误差 (%)	BRGn	Kbaud	误差 (%)	BRGn	Kbaud	误差 (%)
0.3	—	—	—	—	—	—	—	—	—
1.2	207	1.202	0.16	185	1.203	0.23	—	—	—
2.4	103	2.404	0.16	92	2.406	0.23	185	2.406	0.23
4.8	51	4.808	0.16	46	4.76	-0.83	92	4.811	0.23
9.6	25	9.615	0.16	22	9.727	1.32	46	9.520	-0.83
19.2	12	19.231	0.16	11	18.643	-2.90	22	19.454	1.32
38.4	6	35.714	-6.99	5	37.286	-2.90	11	37.286	-2.90
57.6	3	62.5	8.51	3	55.930	-2.90	7	55.930	-2.90
115.2	1	125	8.51	1	111.86	-2.90	3	111.86	-2.90
250	0	250	0	—	—	—	—	—	—

BRGH=1 时的波特率和误差

BRG 寄存器

Bit	7	6	5	4	3	2	1	0
Name	BRG7	BRG6	BRG5	BRG4	BRG3	BRG2	BRG1	BRG0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

“x”：未知

Bit 7~0 **BRG7~BRG0：**波特率值

软件设置 BRGH 位 (设置波特率发生器的速度) 和 BRG 寄存器 (设置波特率的值)，一起控制 UART 的波特率。

注：若 BRGH=0，波特率 = f_{SYS}/[64×(N+1)]；
若 BRGH=1，波特率 = f_{SYS}/[16×(N+1)]。

UART 模块的设置与控制

UART 采用标准的不归零码传输数据，这种方法通常被称为 NRZ 法。它由 1 位起始位，8 位或 9 位数据位和 1 位或者两位停止位组成。奇偶校验是由硬件自动完成的，可设置成奇校验、偶校验和无校验三种格式。常用的数据传输格式由 8 位数据位，1 位停止位，无校验组成，用 8、N、1 表示，它是系统上电的默认格式。数据位数、停止位数和奇偶校验由 UCR1 寄存器的 BNO、PRT、PREN 和 STOPS 设定。用于数据发送和接收的波特率由一个内部的 8 位波特率发送器产生，数据传输时低位在前高位在后。尽管 UART 发送器和接收器在功能上相互独立，但它们使用相同的数据传输格式和波特率，在任何情况下，停止位是必须的。

UART 的使能和除能

UART 是由 UCR1 寄存器的 UARTEN 位来使能和除能的。若 UARTEN、TXEN 和 RXEN 都为高，则 TX 和 RX 分别为 UART 的发送端口和接收端口。若没有数据发送，TX 引脚默认状态为高电平。

UARTEN 清零将除能 TX 和 RX，其可用作普通 I/O 口或其它引脚共用功能。当 UART 被除能时将清空缓冲器，所有缓冲器中的数据将被忽略，另外错误和状态标志位被复位，TXEN、RXEN、TXBRK、RXIF、OERR、FERR、PERR 和 NF 清零而 TIDLE、TXIF 和 RIDLE 置位，UCR1、UCR2 和 BRG 寄存器中的其它位保持不变。若 UART 工作时 UARTEN 清零，所有发送和接收将停止，模块也将复位成上述状态。当 UART 再次使能时，它将在上次配置下重新工作。

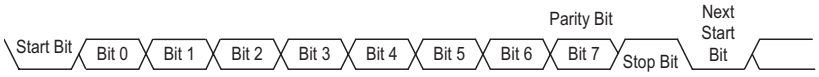
数据位、停止位位数以及奇偶校验的选择

数据传输格式由数据长度、是否校验、校验类型、地址位以及停止位长度组成。它们都是由 UCR1 寄存器的各个位控制的。BNO 决定数据传输是 8 位还是 9 位；PRT 决定校验类型；PRTEN 决定是否选择奇偶校验；而 STOPS 决定选用 1 位还是 2 位停止位。下表列出了各种数据传输格式。地址位用来确定此帧是否为地址。停止位的长度和数据位的长度无关。

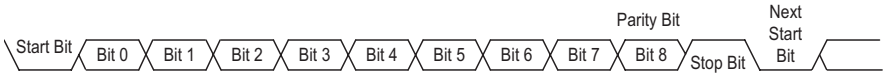
起始位	数据位	地址位	校验位	停止位
8 位数据位				
1	8	0	0	1
1	7	0	1	1
1	7	1	0	1
9 位数据位				
1	9	0	0	1
1	8	0	1	1
1	8	1	0	1

发送和接收数据格式

下图是传输 8 位和 9 位数据的波形。



8-Bit Data Format



9-Bit Data Format

UART 发送器

UCR1 寄存器的 BNO 位是控制数据传输的长度。BNO=1 其长度为 9 位，第 9 位 MSB 存储在 UCR1 寄存器的 TX8 中。发送器的核心是发送移位寄存器 TSR，它的数据由发送寄存器 TXR 提供，应用程序只须将发送数据写入 TXR 寄存器。上组数据的停止位发出前，TSR 寄存器禁止写入。如果还有新的数据要发送，一旦停止位发出，待发数据将会从 TXR 寄存器加载到 TSR 寄存器。TSR 不像其它寄存器一样映射到数据存储器，所以应用程序不能对其进行读写操作。TXEN=1，发送使能，但若 TXR 寄存器没有数据或者波特率没有设置，发送器将不会工作。先写 TXR 寄存器再置高 TXEN 也会触发发送。当发送器使能，若 TSR 寄存器为空，数据写入 TXR 寄存器将会直接加载到 TSR 寄存器中。发送器工作时，TXEN 清零，发送器将立刻停止工作并且复位，此时 TX 引脚用作普通 I/O 口或其它引脚共用功能。

发送数据

当 UART 发送数据时，数据从移位寄存器中移到 TX 引脚上，其低位在前高位在后。在发送模式中，TXR 寄存器在内部总线和发送移位寄存器间形成一个缓冲。如果选择 9 位数据传输格式，最高位 MSB 存储在 UCR1 寄存器的 TX8 中。

发送器初始化可由如下步骤完成：

- 正确地设置 BNO、PRT、PREN 和 STOPS 位以确定数据长度、校验类型和停止位长度。
- 设置 BRG 寄存器，选择期望的波特率。
- 置高 TXEN，使能 UART 发送器且使 TX 作为 UART 的发送端。
- 读取 USR 寄存器，然后将待发数据写入 TXR 寄存器。注意，此步骤会清除 TXIF 标志位。如果要发送多个数据只需重复上一步骤。

当 TXIF=0 时，数据将禁止写入 TXR 寄存器。可以通过以下步骤来清除 TXIF：

1. 读取 USR 寄存器
2. 写 TXR 寄存器

只读标志位 TXIF 由 UART 硬件置位。若 TXIF=1，TXR 寄存器为空，其它数据可以写入而不会覆盖以前的数据。若 TEIE=1，TXIF 标志位会影响中断。在数据传输时，写 TXR 指令会将待发数据暂存在 TXR 寄存器中，当前数据发送完毕后，待发数据被加载到发送移位寄存器中。当发送器空闲时，写 TXR 指令会将数据直接加载到 TSR 寄存器中，数据传输立刻开始且 TXIF 置位。当一帧数据发送完毕，TIDLE 将被置位。

可以通过以下步骤来清除 TIDLE：

1. 读取 USR 寄存器
2. 写 TXR 寄存器

清除 TXIF 和 TIDLE 软件执行次序相同。

发送暂停字

若 TXBRK=1，下一帧将会发送暂停字。它是由一个起始位、13×N (N=1, 2……) 位逻辑 0 组成。置位 TXBRK 将会发送暂停字，而清除 TXBRK 将产生停止位，传输暂停字不会产生中断。需要注意思的是，暂停字至少 13 位宽。若 TXBRK 持续为高，那么发送器会一直发送暂停字；当应用程序清除了 TXBRK，发送器将传输最后一帧暂停字再加上一位或者两位停止位。暂停字后的高电平保证下一帧数据起始位的检测。

UART 接收器

UART 接收器支持 8 位或者 9 位数据接收。若 BNO=1，数据长度为 9 位，而最高位 MSB 存放在 UCR1 寄存器的 RX8 中。接收器的核心是串行移位寄存器 RSR。RX 引脚上的数据送入数据恢复器中，它在 16 倍波特率的频率下工作，而串行移位器工作在正常波特率下。当在 RX 引脚上检测到停止位，数据从 RSR 寄存器中加载到 RXR 寄存器。RX 引脚上的每一位数据会被采样三次以判断其逻辑状态。RSR 不像其它寄存器一样映射在数据存储器，所以应用程序不能对其进行读写操作。

接收数据

当 UART 接收数据时，数据低位在前高位在后，连续地从 RX 引脚进入。RXR 寄存器在内部总线和接收移位寄存器间形成一个缓冲。RXR 寄存器是一个两层的 FIFO 缓冲器，它能保存两帧数据的同时接收第三帧数据，应用程序必须保证在接收完第三帧前读取 RXR 寄存器，否则忽略第三帧数据并且发生溢出错误。

接收器的初始化可由如下步骤完成：

- 正确地设置 BNO、PRT、PREN 和 STOPS 位以确定数据长度、校验类型和停止位长度。
- 设置 BRG 寄存器，选择期望的波特率。
- 置高 RXEN，使能 UART 发送器且使 RX 作为 UART 的接收端。

此时接收器被使能并检测起始位。

接收数据将会发生如下事件：

- 当 RXR 寄存器中有一帧以上的数据时，USR 寄存器中的 RXIF 位将会置位。
- 若 RIE=1，数据从 RSR 寄存器加载到 RXR 寄存器中将产生中断。
- 若接收器检测到帧错误、噪声干扰错误、奇偶出错或溢出错误，那么相应的错误标志位置位。

可以通过如下步骤来清除 RXIF：

1. 读取 USR 寄存器
2. 读取 RXR 寄存器

接收暂停字

UART 接收任何暂停字都会当作帧错误处理。接收器只根据 BNO 和 STOPS 位确定一帧数据的长度。若暂停字位数大于 BNO 和 STOPS 位指定的长度，接收器认为接收已完毕，RXIF 和 FERR 置位，RXR 寄存器清 0，若相应的中断允许且 RIDLE 为高将会产生中断。若暂停字较长，接收器收到起始位、数据位将会置位 FERR 标志，且在下一起始位前必须检测到有效的停止位。暂停字只会被认为包含信息 0 且会置位 FERR 标志。暂停字将会加载到缓冲器中，在接收

到停止位前不会再接收数据，没有检测到停止位也会置位只读标志位 RIDLE。

UART 接收到暂停字会产生以下事件：

- 帧错误标志位 FERR 置位。
- RXR 寄存器清零。
- OERR、NF、PERR、RIDLE 或 RXIF 可能会置位。

空闲状态

当 UART 接收数据时，即在起初位和停止位之间，USR 寄存器的接收标志位 RIDLE 清零。在停止位和下一帧数据的起始位之间，RIDLE 被置位，表示接收器空闲。

接收中断

USR 寄存器的只读标志位 RXIF 由接收器的边缘触发置位。若 RIE=1，数据从移位寄存器 RSR 加载到 RXR 寄存器时产生中断，同样地，溢出也会产生中断。

接收错误处理

UART 会产生几种接收错误，下面部分将描述各错误以及怎样处理。

溢出——OERR 标志

RXR 寄存器是一个两层的 FIFO 缓冲器，它能保存两帧数据的同时接收第三帧数据，应用程序必须保证在接收完第三帧前读取 RXR 寄存器，否则发生溢出错误。

产生溢出错误时将会发生以下事件：

- USR 寄存器中 OERR 被置位。
- RXR 寄存器中数据不会丢失。
- RSR 寄存器数据将会被覆盖。
- 若 RIE=1，将会产生中断。

先读取 USR 寄存器再读取 RXR 寄存器可将 OERR 清零。

噪声干扰——NF 标志

数据恢复时多次采样可以有效的鉴别出噪声干扰。当检测到数据受到噪声干扰时将会发生以下事件：

- 在 RXIF 上升沿，USR 寄存器中只读标志位 NF 置位。
- 数据从 RSR 寄存器加载到 RXR 寄存器中。
- 不产生中断，此位置位的同时由 RXIF 请求中断。

先读取 USR 寄存器再读取 RXR 寄存器可将 NF 清零。

帧错误——FERR 标志

若在停止位上检测到 0，USR 寄存器中只读标志 FERR 置位。若选择两位停止位，此两位都必须为高，否则将置位 FERR。它同数据一起存储在缓冲器中，可被任何复位清零。

奇偶校验错误——PERR 标志

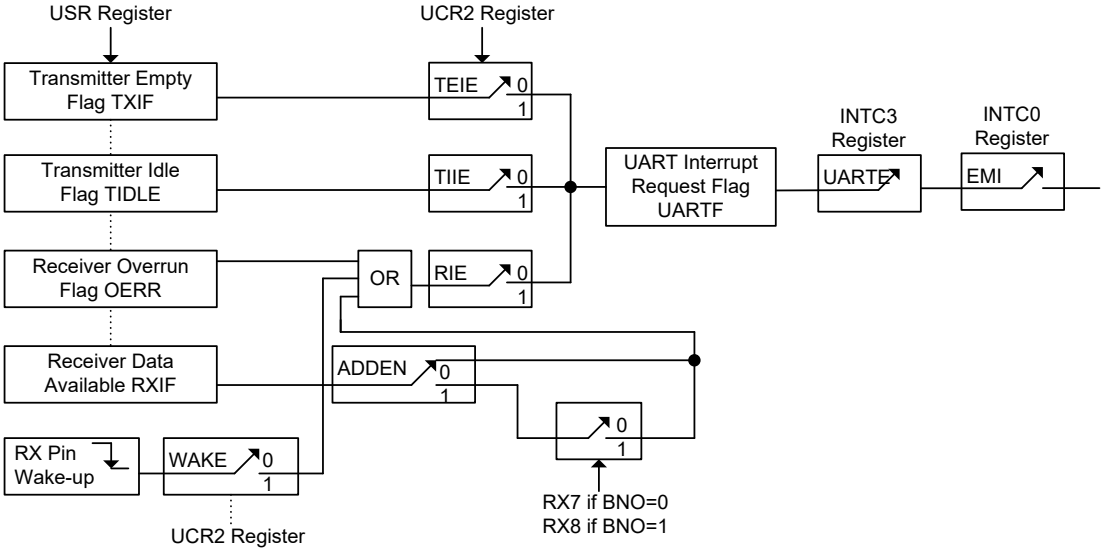
若接收到数据出现奇偶校验错误，USR 寄存器中只读标志 PERR 置位。只有使能了奇偶校验，选择了校验类型，此标志位才有效。它同数据一起存储在缓冲

器中，可被任何复位清除。注意，FERR 和 PERR 与相应的数据一起存储有缓冲器中，在读取数据之前必须先访问错误标志位。

UART 模块中断结构

UART 拥有一个单独的中断。发送寄存器为空、发送器空闲、接收器数据有效、接收器溢出、地址检测和 RX 引脚唤醒都会产生中断。当其中任何一种情况发生时，若其对应的中断控制位使能、 整体 UART 中断允许且堆栈未满，程序将会跳转到相应的中断向量执行中断服务程序，而后再返回主程序。这其中的四种情况在 USR 寄存器中有相关的标志位，若 UCR2 寄存器中相应中断允许位被置位，USR 寄存器中标志位将会产生中断。发送器有两个相应的中断允许位而接收器共用一个中断允许位。这些允许位可用于禁止个别的 UART 中断源。地址检测也是 UART 的中断源，它没有相应的标志位，若 UCR2 寄存器中 ADDEN=1，当检测到地址将会产生 UART 中断。RX 引脚唤醒也可以产生 UART 中断，它没有相应的标志位，当 UXR2 中的 WAKE 和 RIE 位被置位，RX 引脚上有下降沿可以唤醒单片机。应注意，RX 唤醒中断发生时，系统必须延时 tSST 才能正常工作。

注意，USR 寄存器标志位为只读状态，软件不能对其进行设置，在进入相应中断服务程序时也不能清除这些标志位，其它中断亦是如此。这些标志位仅在 UART 特定动作发生时才会自动被清除，详细解释见 UART 寄存器章节。整体 UART 中断的使能或除能可由中断控制寄存器中的相关中断使能控制位控制，其中断请求由 UART 模块决定。



UART 中断框图

地址检测模式

置位 UCR2 寄存器中的 ADDEN 将启动地址检测模式。若此位为“1”，可产生接收数据有效中断，其请求标志位为 RXIF。若 ADDEN 有效，只有在接收到数据最高位为 1 才会产生中断，中断允许位 UARE 和 EMI 也要使能才会产生中断。地址的最高位为第 9 位 (BNO=1) 或第 8 位 (BNO=0)，若此位为高，则接收到的是地址而非数据。只有接收的数据的最后一位为高才会产生中断。若 ADDEN 除能，每接收到一个有效数据便会置位 RXIF，而不用考虑数据的最后

一位。地址检测和奇偶校验在功能上相互排斥，若地址检测模式使能，必须保证操作的正确，同时必须将奇偶检验使能位清零，除能奇偶校验。

ADDEN	Bit 9 (BNO=1) Bit 8 (BNO=0)	产生 UART 中断
0	0	√
	1	√
1	0	×
	1	√

ADDEN 位功能

UART 模块暂停和唤醒

MCU 进入暂停模式后 UART 模块将暂停运行。当 UART 进入暂停模式后，UART 模块的时钟源将除能。当传送数据时 UART 进入暂停模式，发送将停止直到 UART 模块时钟再次使能。同样地，当接收数据时 UART 进入暂停模式，数据接收也会停止。当 UART 电路进入暂停模式，USR、UCR1、UCR2 、接收 / 发送寄存器以及 BRG 寄存器都不会受到影响。建议在 MCU 进入暂停模式前先确保数据发送或接收已完成。

UART 功能中包括了 RX 引脚的唤醒功能，由 UCR2 寄存器中 WAKE 位控制。进入暂停模式前，若该标志位与 UART 允许位 UARTEN、接收器允许位 RXEN 和接收器中断位 RIE 都被置位，则 RX 引脚的下降沿可唤醒单片机。唤醒后系统需延时 t_{SST} 才能正常工作，在此期间，RX 引脚上的任何数据将被忽略。若要唤醒并产生 UART 中断，除了唤醒使能控制位和接收中断使能控制位需置位外，全局中断允许位 EMI 和 UART 中断使能控制位 UARE 也必须置位；若这两控制位没有被置位，那么，单片机将可以被唤醒但不会产生中断。同样唤醒后系统需一定的延时才能正常工作，然后才会产生 UART 中断。

下面表格说明了在各暂停模式下 UART 模块 RX 引脚的唤醒功能。

如果主系统时钟可以来自高频 f_H 和低频 f_{SUB}。

工作模式	说明				RX 引脚唤醒功能
	CPU	f_{SYS}	f_H	f_{SUB}	
空闲模式 0	Off	Off	Off	On	当 UCR2.2(RIE)=1、UCR2.3(WAKE)=1 且 CPU 进入空闲模式 0，RX 引脚的下降沿将开启 f_{SYS} 并唤醒 CPU。
空闲模式 1	Off	On	On	On	当 UCR2.2(RIE)=1、UCR2.3(WAKE)=1 且 CPU 进入空闲模式 1： (1) 若 UART 没有在传输数据，RX 引脚的下降沿将开启 f_{SYS} 而 CPU 仍关闭。 (2) 若 UART 正在传输数据，CPU 将在数据传输完成后唤醒。 注：若 RIE=0、WAKE=1 且 UART 正在传输数据，CPU 在数据传输完成后也不唤醒。
休眠模式	Off	Off	Off	Off	当 UCR2.2(RIE)=1、UCR2.3(WAKE)=1 且 CPU 进入休眠模式， RX 引脚的下降沿将开启 f_{SYS} 并唤醒 CPU。

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。此单片机提供许多外部中断和内部中断功能，外部中断由 H1、H2、H3、INT1 和 NFIN 引脚动作产生，而内部中断由各种内部功能，如定时器模块、比较器、马达保护模组、线性霍尔传感器检测、PWM 模组、16 位 CAPTM 模块、时基、LVD、EEPROM、A/D 转换器、I^2C 和 UART 等产生。

中断寄存器

中断控制基本上是在一定单片机条件发生时设置请求标志位，应用程序中中断使能位的设置是通过位于专用数据存储器中的寄存器控制的。寄存器的数量由单片机决定，但总的分为三类。第一类是 INTC0~INTC3 寄存器，用于设置基本的中断；第二类是 MFI0~MFI6 寄存器，用于设置多功能中断。第三类是寄存器 INTEG，用来设置外部中断触发边沿类型。

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

功能	使能位	请求标志	注释
总中断	EMI	—	—
外部中断 0 (霍尔传感器 HA/HB/HC 中断)	HALLE	HALLF	MFI0
	HALAE	HALAF	霍尔噪声滤波器
	HALBE	HALBF	霍尔噪声滤波器
	HALCE	HALCF	霍尔噪声滤波器
外部中断 1	INT1E	INT1F	—
比较器 0	C0E	C0F	—
噪声滤波器中断	NFIE	NFIF	NFI 中断 噪声滤波器
A/D 转换器	AEOCE	AEOCF	—
	ALIME	ALIMF	—
CAPTM	CAPOE	CAPOF	—
	CAPCE	CAPCF	—
LVD	LVDE	LVDF	—
EEPROM 写中断	EPWE	EPWF	—
PWM	PWMDnE	PWMDnF	n=0~2
	PWMPE	PWMPF	—
TM	PTMnAE	PTMnAF	n=0~3
	PTMnPE	PTMnPF	n=0~3
I^2C	IICE	IICF	—
UART	UARTE	UARTF	—
多功能中断	MFnE	MFnF	n=0~6
时基中断	TBE	TBF	—

中断寄存器位命名模式

寄存器名称	位							
	7	6	5	4	3	2	1	0
INTEG0	—	HSEL	INTCS1	INTCS0	INTBS1	INTBS0	INTAS1	INTAS0
INTEG1	—	—	—	—	—	—	INT1S1	INT1S0
INTC0	—	C0F	INT1F	HALLF (MF0F)	C0E	INT1E	HALLE (MF0E)	EMI
INTC1	EPWF	LVDF	MF1F	NFIF	EPWE	LVDE	MF1E	NFIE
INTC2	MF5F	MF4F	MF3F	MF2F	MF5E	MF4E	MF3E	MF2E
INTC3	TBF	MF6F	UARTF	I2CF	TBE	MF6E	UARTE	I2CE
MFI0	—	HALCF	HALBF	HALAF	—	HALCE	HALBE	HALAE
MFI1	CAPCF	CAPOF	ALIMF	AEOCF	CAPCE	CAPOE	ALIME	AEOCE
MFI2	PWMPF	PWMD2F	PWMD1F	PWMD0F	PWMPE	PWMD2E	PWMD1E	PWMD0E
MFI3	—	—	PTM2AF	PTM2PF	—	—	PTM2AE	PTM2PE
MFI4	—	—	PTM0AF	PTM0PF	—	—	PTM0AE	PTM0PE
MFI5	—	—	PTM1AF	PTM1PF	—	—	PTM1AE	PTM1PE
MFI6	—	—	PTM3AF	PTM3PF	—	—	PTM3AE	PTM1PE

中断寄存器内容

INTEG0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	HSEL	INTCS1	INTCS0	INTBS1	INTBS0	INTAS1	INTAS0
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　**HSEL**：HA/HB/HC 来源选择
0：H1/H2/H3
1：CMP1/CMP2/CMP3 输出

Bit 5~4　**INTCS1~INTCS0**：FHC 中断边沿控制 INTC
00：除能
01：上升沿
10：下降沿
11：双边沿

Bit 3~2　**INTBS1~INTBS0**：FHB 中断边沿控制 INTB
00：除能
01：上升沿
10：下降沿
11：双边沿

Bit 1~0　**INTAS1~INTAS0**：FHA 中断边沿控制 INTA
00：除能
01：上升沿
10：下降沿
11：双边沿

INTEG1 寄存器

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

Bit 7~2　未定义，读为“0”

Bit 1~0　**INT1S1~INT1S0**：外部 INT1 中断边沿控制
00：除能
01：上升沿
10：下降沿
11：双边沿

INTC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	C0F	INT1F	HALLF (MF0F)	C0E	INT1E	HALLE (MF0E)	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　**C0F**：比较器 0 中断请求标志位
0：无请求
1：中断请求

Bit 5　**INT1F**：外部中断 1 中断请求标志位
0：无请求
1：中断请求

Bit 4　**HALLF**：霍尔传感器总中断请求标志位
0：无请求
1：中断请求

Bit 3　**C0E**：比较器 0 中断控制位
0：除能
1：使能

Bit 2　**INT1E**：外部中断 1 中断控制位
0：除能
1：使能

Bit 1　**HALLE**：霍尔传感器总中断控制位
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	EPWF	LVDF	MF1F	NFIF	EPWE	LVDE	MF1E	NFIE
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 **EPWF**：EEPROM 写中断请求标志位
0：无请求
1：中断请求

Bit 6 **LVDF**：LVD 中断请求标志位
0：无请求
1：中断请求

Bit 5 **MF1F**：多功能中断 1 请求标志位
0：无请求
1：中断请求

Bit 4 **NFIF**：Noise Filter 中断请求标志位
0：无请求
1：中断请求

Bit 3 **EPWE**：EEPROM 写中断控制位
0：除能
1：使能

Bit 2 **LVDE**：LVD 中断控制位
0：除能
1：使能

Bit 1 **MF1E**：多功能中断 1 控制位
0：除能
1：使能

Bit 0 **NFIE**：Noise Filter 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	MF5F	MF4F	MF3F	MF2F	MF5E	MF4E	MF3E	MF2E
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 **MF5F**：多功能中断 5 请求标志位
0：无请求
1：中断请求

Bit 6 **MF4F**：多功能中断 4 请求标志位
0：无请求
1：中断请求

Bit 5 **MF3F**：多功能中断 3 请求标志位
0：无请求
1：中断请求

Bit 4 **MF2F**：多功能中断 2 请求标志位
0：无请求
1：中断请求

Bit 3 **MF5E**：多功能中断 5 控制位
0：除能
1：使能

Bit 2 **MF4E**：多功能中断 4 控制位
0：除能
1：使能

Bit 1 **MF3E**：多功能中断 3 控制位
0：除能
1：使能

Bit 0 **MF2E**：多功能中断 2 控制位
0：除能
1：使能

INTC3 寄存器

Bit	7	6	5	4	3	2	1	0
Name	TBF	MF6F	UARTF	I2CF	TBE	MF6E	UARTE	I2CE
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 **TBF**：时基中断请求标志位
0：无请求
1：中断请求

Bit 6 **MF6F**：多功能中断 6 请求标志位
0：无请求
1：中断请求

Bit 5 **UARTF**：UART 中断请求标志位
0：无请求
1：中断请求

Bit 4 **I2CF**：I^2C 中断请求标志位
0：无请求
1：中断请求

Bit 3 **TBE**：时基中断控制位
0：除能
1：使能

Bit 2 **MF6E**：多功能中断 6 控制位
0：除能
1：使能

Bit 1 **UARTE**：UART 中断控制位
0：除能
1：使能

Bit 0 **I2CE**：I^2C 中断控制位
0：除能
1：使能

MFI0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	HALCF	HALBF	HALAF	—	HALCE	HALBE	HALAE
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 **HALCF**：霍尔传感器 C 中断请求标志位
0：无请求
1：中断请求

Bit 5 **HALBF**：霍尔传感器 B 中断请求标志位
0：无请求
1：中断请求

Bit 4 **HALAF**：霍尔传感器 A 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 **HALCE**：霍尔传感器 C 中断控制位
0：除能
1：使能

Bit 1 **HALBE**：霍尔传感器 B 中断控制位
0：除能
1：使能

Bit 0 **HALAE**：霍尔传感器 A 中断控制位
0：除能
1：使能

MFI1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	CAPCF	CAPOF	ALIMF	AEOCF	CAPCE	CAPOE	ALIME	AEOCE
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 **CAPCF**：捕捉定时器比较匹配中断请求标志位
0：无请求
1：中断请求

Bit 6 **CAPOF**：捕捉定时器捕捉溢出中断请求标志位
0：无请求
1：中断请求

Bit 5 **ALIMF**：A/D 转换器 EOC 后产生比较结果中断请求标志位
0：无请求
1：中断请求

Bit 4 **AEOCF**：A/D 转换器中断请求标志位
0：无请求
1：中断请求

Bit 3 **CAPCE**：捕捉定时器比较匹配中断控制位
0：除能
1：使能

Bit 2 **CAPOE**：捕捉定时器捕捉溢出中断控制位
0：除能
1：使能

Bit 1 **ALIME**：A/D 转换器 EOC 后产生比较结果中断控制位
0：除能
1：使能

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

MFI2 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PWMPF	PWMD2F	PWMD1F	PWMD0F	PWMPE	PWMD2E	PWMD1E	PWMD0E
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 **PWMPF**：PWM 周期匹配中断请求标志位
0：无请求
1：中断请求

Bit 6 **PWMD2F**：PWM2 占空比匹配中断请求标志位
0：无请求
1：中断请求

Bit 5 **PWMD1F**：PWM1 占空比匹配中断请求标志位
0：无请求
1：中断请求

Bit 4 **PWMD0F**：PWM0 占空比匹配中断请求标志位
0：无请求
1：中断请求

Bit 3 **PWMPE**：PWM 周期匹配中断控制位
0：除能
1：使能

Bit 2 **PWMD2E**：PWM2 占空比匹配中断控制位
0：除能
1：使能

Bit 1 **PWMD1E**：PWM1 占空比匹配中断控制位
0：除能
1：使能

Bit 0 **PWMD0E**：PWM0 占空比匹配中断控制位
0：除能
1：使能

MFI3 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	PTM2AF	PTM2PF	—	—	PTM2AE	PTM2PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 未定义，读为“0”

Bit 5 **PTM2AF**：TM2 比较器 A 匹配中断请求标志位
0：无请求
1：中断请求

Bit 4 **PTM2PF**：TM2 比较器 P 匹配中断请求标志位
0：无请求
1：中断请求

Bit 3~2 未定义，读为“0”

Bit 1 **PTM2AE**：TM2 比较器 A 匹配中断控制位
0：除能
1：使能

Bit 0 **PTM2PE**：TM2 比较器 P 匹配中断控制位
0：除能
1：使能

MFI4 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	PTM0AF	PTM0PF	—	—	PTM0AE	PTM0PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 未定义，读为“0”

Bit 5 **PTM0AF**：TM0 比较器 A 匹配中断请求标志位
0：无请求
1：中断请求

Bit 4 **PTM0PF**：TM0 比较器 P 匹配中断请求标志位
0：无请求
1：中断请求

Bit 3~2 未定义，读为“0”

Bit 1 **PTM0AE**：TM0 比较器 A 匹配中断控制位
0：除能
1：使能

Bit 0 **PTM0PE**：TM0 比较器 P 匹配中断控制位
0：除能
1：使能

MFI5 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	PTM1AF	PTM1PF	—	—	PTM1AE	PTM1PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 未定义，读为“0”

Bit 5 **PTM1AF**：TM1 比较器 A 匹配中断请求标志位
0：无请求
1：中断请求

Bit 4 **PTM1PF**：TM1 比较器 P 匹配中断请求标志位
0：无请求
1：中断请求

Bit 3~2 未定义，读为“0”

Bit 1 **PTM1AE**：TM1 比较器 A 匹配中断控制位
0：除能
1：使能

Bit 0 **PTM1PE**：TM1 比较器 P 匹配中断控制位
0：除能
1：使能

MFI6 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	PTM3AF	PTM3PF	—	—	PTM3AE	PTM3PE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6　未定义，读为“0”

Bit 5　**PTM3AF**：TM3 比较器 A 匹配中断请求标志位
0：无请求
1：中断请求

Bit 4　**PTM3PF**：TM3 比较器 P 匹配中断请求标志位
0：无请求
1：中断请求

Bit 3~2　未定义，读为“0”

Bit 1　**PTM3AE**：TM3 比较器 A 匹配中断控制位
0：除能
1：使能

Bit 0　**PTM3PE**：TM3 比较器 P 匹配中断控制位
0：除能
1：使能

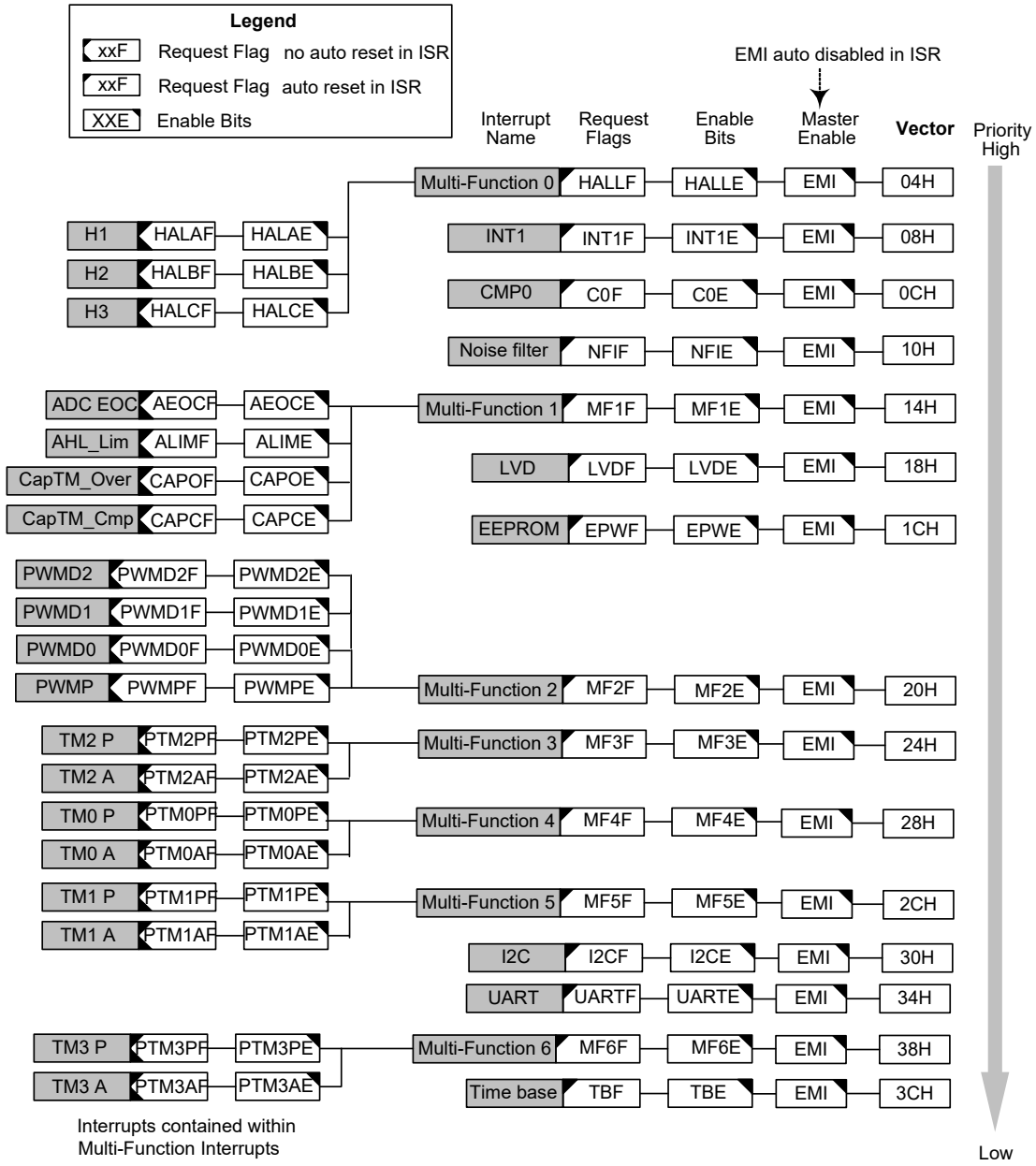
中断操作

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

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

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

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



中断结构

外部中断 0 (霍尔传感器中断)

外部中断 0，即霍尔传感器中断，属于多功能中断。每当 H1，H2 和 H3 引脚发生边沿变化时 (上升沿，下降沿和双边沿，通过 INTEG0 寄存器选择) 且经由 Hall Noise Filter 认可后，外部中断请求标志 HALAF、HALBF 和 HALCF 被置位，外部中断 (霍尔传感器中断) 请求产生。由于外部霍尔传感器引脚与其它 I/O 引脚共用，所以在使能霍尔传感器中断之前需先选择外部霍尔传感器引脚功能。霍尔传感器输入脚 H1，H2 和 H3 引脚上的信号变化可控制外部中断。若要跳转到相应中断向量地址，总中断控制位 EMI 和多功能中断使能位 HALLE 需先被置位。当中断使能，堆栈未满并且外部中断脚发生边沿改变，将调用外

部中断向量子程序。当响应外部中断服务子程序时，EMI 位会被清零以除能其它中断，多功能中断请求标志 HALLF 也可自动清除，但 HALAF，HALBF 和 HALCF 标志需在应用程序中手动清除。

外部中断 1

通过 INT1 引脚上的信号变化可控制外部中断 1。每当 INT1 引脚发生边沿变化 (触发边沿类型通过 INTEG1 寄存器选择)，INT1 中断请求标志 INT1F 被置位，外部中断请求产生。由于外部中断 1 引脚与其它 I/O 引脚共用，所以在使能外部中断 1 之前需先选择外部中断 1 引脚功能。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 INT1E 需先被置位。当中断使能，堆栈未满并且外部中断脚发生边沿改变，将调用外部中断向量子程序。当响应外部中断服务子程序时，中断请求标志位 INT1F 会自动复位且 EMI 位会被清零以除能其它中断。要注意的是，如果引脚被用作外部中断输入，任何外部中断引脚的上拉电阻选择仍然有效。

噪声滤波器引脚 NFIN 中断

通过 NFIN 引脚上的信号变化可控制噪声滤波器中断。每当 NFIN 引脚发生边沿变化 (触发边沿类型通过 NF_VIL 寄存器的 NFIS1 和 NFIS0 位决定)，NFIN 中断请求标志 NFIF 被置位，NFIN 中断请求产生。由于 NFIN 引脚与其它 I/O 引脚共用，所以在使能 NFIN 中断之前需先选择 NFIN 引脚功能。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 NFIE 需先被置位。当中断使能，堆栈未满并且外部中断脚发生边沿改变，将调用外部中断向量子程序。当响应外部中断服务子程序时，中断请求标志位 NFIF 会自动复位且 EMI 位会被清零以除能其它中断。要注意的是，如果引脚被用作外部中断输入，任何外部中断引脚的上拉电阻选择仍然有效。

比较器中断

比较器中断由内部比较器 0 控制。当比较器输出状态改变，比较器中断请求标志 C0F 被置位，比较器中断请求产生。若要跳转到相应中断向量地址，总中断控制位 EMI 和比较器中断使能位 C0E 需先被置位。当中断使能，堆栈未满并且比较器输入产生一个比较器输出变化时，将调用比较器中断向量子程序。当响应中断服务子程序时，外部中断请求标志位 C0F 会自动复位且 EMI 位会被清零以除能其它中断。

时基中断

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

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

TBC 寄存器

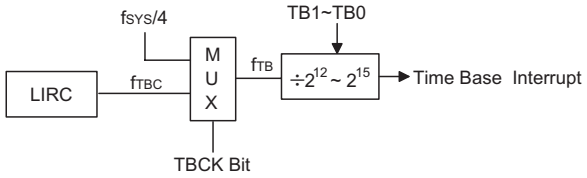
Bit	7	6	5	4	3	2	1	0
Name	TBON	TBCK	TB1	TB0	—	—	—	—
R/W	R/W	R/W	R/W	R/W	—	—	—	—
POR	0	0	1	1	—	—	—	—

Bit 7 **TBON**：时基控制位
0：除能
1：使能

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

Bit 5~4 **TB1~TB0**：选择时基溢出周期位
00：$2^{12}/f_{TB}$
01：$2^{13}/f_{TB}$
10：$2^{14}/f_{TB}$
11：$2^{15}/f_{TB}$

Bit 3~0 未定义，读为“0”



时基中断

多功能中断

此单片机中有多达 7 种多功能中断，与其它中断不同，它没有独立源，但由其它现有的中断源构成，分别为霍尔传感器中断，A/D 中断，PWM 模块中断，CAPTM 中断，TM 中断。

当多功能中断中任何一种中断请求标志 HALLF 和 MF1F~MF5F 被置位，多功能中断请求产生。当中断使能，堆栈未满，包括在多功能中断中的任意一个中断发生时，将调用多功能中断向量中的一个子程序。当响应中断服务子程序时，相关的多功能请求标志位会自动复位且 EMI 位会自动清零以除能其它中断。

但必须注意的是，在中断响应时，虽然多功能中断标志会自动复位，但多功能中断源的请求标志位，即霍尔传感器中断，A/D 中断，PWM 模块中断，CAPTM 中断，TM 中断的请求标志位不会自动复位，必须由应用程序清零。

A/D 转换器中断

A/D 转换器中断属于多功能中断，有两种中断方式。第一种是由 A/D 转换动作的结束来控制。当 A/D 转换器中断请求标志 ALIMF 被置位，即 A/D 转换过程完成时，中断请求发生。第二种是设定 ADCR1 寄存器的 ADCHVE/ADCLVE 位和 ADLVDH/ADLVDL 与 ADHVDH/ADHVDL 边界控制寄存器的值，在 EOC 后产生比较结果，自动产生中断。当总中断使能位 EMI 和 A/D 中断使能位 AEOCE 或 ALIME 被置位，允许程序跳转到各自的中断向量地址。当中断使能，堆栈未满且 A/D 转换动作结束或在 EOC 后产生比较结果时，将调用 A/D 转换器的中断向量子程序。当响应中断服务子程序时，相应的中断请求标志位 AEOCF/ALIMF 会自动清零。EMI 位也会被清零以除能其它中断。

PWM 模块中断

PWM 模块中断属于多功能中断，有四个中断，PWMDn 和 PWMP。当 PWM 占空比匹配或 PWM 周期匹配时，中断请求标志位 PWMDnF 或 PWMPF 被置位，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 和相应多功能中断使能位需先被置位。当中断使能，堆栈未满并且 PWM 占空比匹配或 PWM 周期匹配时，将调用 PWM 中断向量子程序。当响应 PWM 中断服务子程序时，EMI 位会被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 PWMDnF 或 PWMPF 标志需在应用程序中手动清除。

CAPTM 模块中断

CAPTM 模块中断属于多功能中断，有两个中断，CapTM_Over 和 CapTM_Cmp。当 CAPTM 捕捉溢出或比较匹配时，中断请求标志位 CAPOF 或 CAPCF 被置位，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 和相应多功能中断使能位需先被置位。当中断使能，堆栈未满并且捕捉溢出或比较匹配时，将调用 CAPTM 模块中断向量子程序。当响应 CAPTM 模块中断服务子程序时，EMI 位会被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 CAPOF 或 CAPCF 标志需在应用程序中手动清除。

TM 中断

周期型 TM 有两个中断。所有的 TM 中断属于多功能中断。周期型 TM 的两个中断请求标志位分别为 PTMnPF、PTMnAF 及两个使能位 PTMnPE、PTMnAE。当 TM 比较器 P 或比较器 A 匹配情况发生时，任意 TM 中断请求标志被置位，TM 中断请求产生。

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

EEPROM 中断

当写周期结束，EEPROM 中断请求标志 EPWF 被置位，EEPROM 中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI 和 EEPROM 中断使能位 EPWE 需先被置位。当中断使能，堆栈未满且 EEPROM 写周期结束时，可跳转至相关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，EPWF 中断请求标志也可自动清除。

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 LVDF 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，可跳转至相关中断向量子程序中执行。当低电压中断响应，EMI 将被自动清零以除能其它中断，LVDF 中断请求标志也可自动清除。

I²C 中断

当一个字节数据已由 I²C 接口接收或发送完，I²C 中断请求标志 IICF 被置位，I²C 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、I²C 中断使能位 IICE 需先被置位。当中断使能，堆栈未满且一个字节数据已被传送或接收完毕时，可跳转至相关 I²C 中断向量子程序中执行。当 I²C 中断响应，中断请求标志位 IICF 会自动复位且 EMI 位会被清零以除能其它中断。

UART 中断

UART 拥有一个单独的中断。发送寄存器为空、发送器空闲、接收器数据有效、接收器溢出、地址检测和 RX 引脚唤醒都会产生中断。当其中任何一种情况发生时，若其对应的中断控制位使能、 整体 UART 中断允许且堆栈未满，程序将会跳转到相应的中断向量执行中断服务程序，而后再返回主程序。当 UART 中断响应，中断请求标志位 UARTF 会自动复位且 EMI 位会被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 MF1F~MF6F 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。

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 位用于选择一个固定的电压参考点。LVDO 位被置位时低电压情况发生，若 LVDO 位为低表明 V_{DD} 电压工作在当前所设置低电压水平值之上。LVDEN 位用于控制低电压检测功能的开启 / 关闭，设置此位为高使能此功能，反之，关闭内部低电压检测电路。低电压检测会有一定的功耗，在不使用时可考虑关闭此功能，此举在功耗要求严格的电池供电应用中值得考虑。

LVDC 寄存器

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

Bit 7~6　未定义，读为“0”

Bit 5　**LVDO**：LVD 输出标志位
0：未检测到低电压
1：检测到低电压

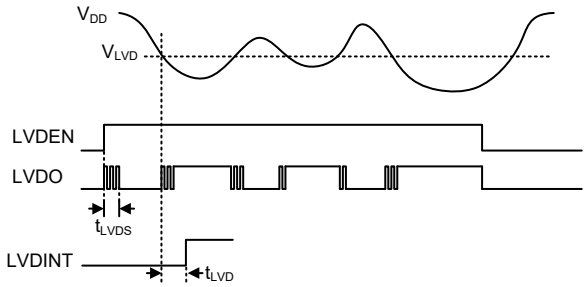
Bit 4　**LVDEN**：低电压检测控制位
0：除能
1：使能

Bit 3　未定义，读为“0”

Bit 2~0　**VLVD2~VLVD0**：选择 LVD 电压
000：3.6V
001：3.6V
010：3.6V
011：3.6V
100：3.6V
101：3.6V
110：3.6V
111：3.6V

LVD 操作

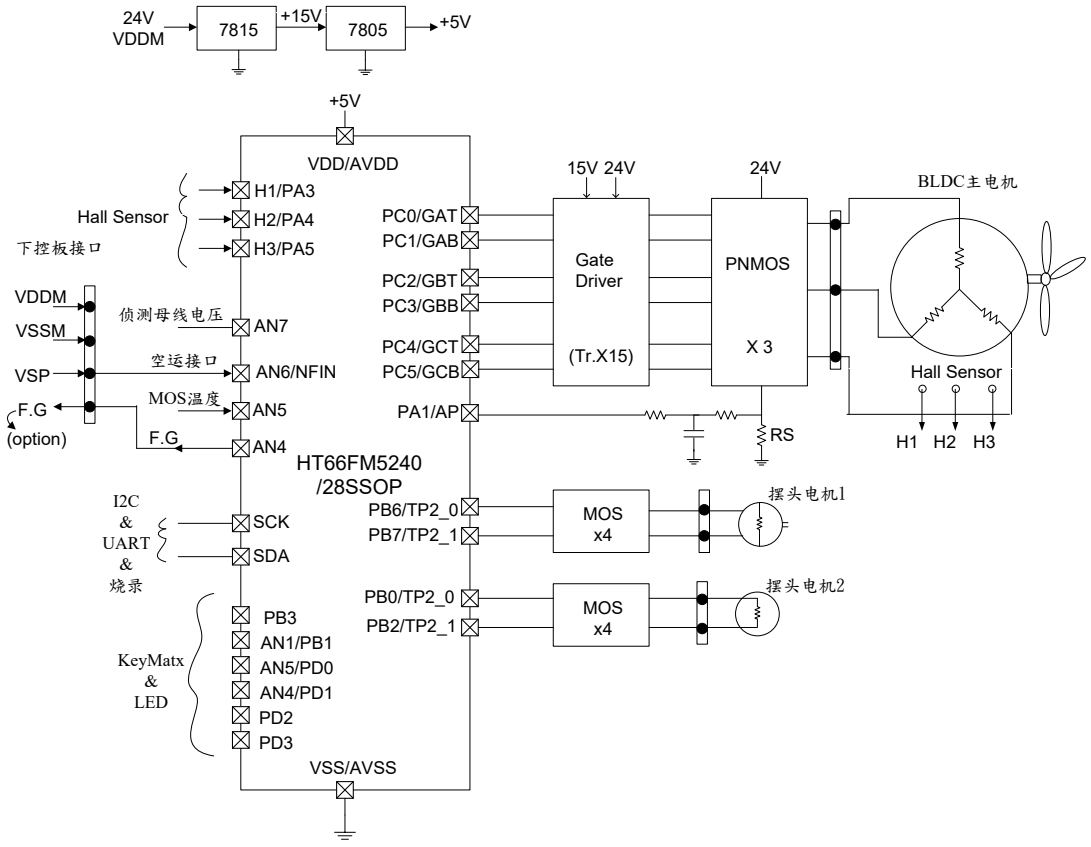
通过比较电源电压 V_{DD} 与存储在 LVDC 寄存器中的预置电压值的结果，低电压检测功能工作。其设置的值为 3.6V。当电源电压 V_{DD} 低于预置电压值时，LVDO 位被置为高，表明低电压产生。低电压检测功能由一个自动使能的参考电压提供。若 LVDEN 位为高，当单片机掉电时低电压检测器保持有效状态。低电压检测器使能后，读取 LVDO 位前，电路稳定需要一定的延时 t_{LVDS}。注意，V_{DD} 电压可能上升或下降比较缓慢，在 V_{LVD} 电压值附近时，LVDO 位可能有多种变化。



LVD 操作

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

应用电路



指令集

简介

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

为了更加容易理解各种各样的指令码，接下来按功能分组介绍它们。

指令周期

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

数据的传送

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

算术运算

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

逻辑和移位运算

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

分支和控制转换

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

位运算

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

查表运算

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

其它运算

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

指令集概要

下表中说明了按功能分类的指令集，用户可以将该表作为基本的指令参考。

惯例

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

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

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

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

指令定义

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

RR [m] Rotate Data Memory right
指令说明 将指定数据存储器的内容循环右移 1 位且第 0 位移到第 7 位。
功能表示 [m].i ← [m].(i+1) (i=0~6)
[m].7 ← [m].0
影响标志位 无

RRA [m] Rotate Data Memory right with result in ACC
指令说明 将指定数据存储器的内容循环右移 1 位，第 0 位移到第 7 位，移位结果存放到累加器，而指定数据存储器的内容保持不变。
功能表示 ACC.i ← [m].(i+1) (i=0~6)
ACC.7 ← [m].0
影响标志位 无

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

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

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

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

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

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

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

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

SIZ [m] Skip if increment Data Memory is 0

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

功能表示 [m] ←[m] + 1，如果 [m]=0 跳过下一条指令执行

影响标志位 无

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

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

功能表示 ACC ←[m] + 1，如果 ACC=0 跳过下一条指令执行

影响标志位 无

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

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

功能表示 如果 [m].i≠0，跳过下一条指令执行

影响标志位 无

SUB A, [m] Subtract Data Memory from ACC

指令说明 将累加器的内容减去指定的数据存储器的数据，把结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。

功能表示 ACC ← ACC – [m]

影响标志位 OV、Z、AC、C

SUBM A, [m] Subtract Data Memory from ACC with result in Data Memory

指令说明 将累加器的内容减去指定数据存储器的数据，结果存放到指定的数据存储器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。

功能表示 [m] ← ACC – [m]

影响标志位 OV、Z、AC、C

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

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

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

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

SZA [m] Skip if Data Memory is 0 with data movement to ACC
指令说明 将指定数据存储器内容复制到累加器，并判断指定数据存储器的内容是否为 0，若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示 ACC ← [m]，如果 [m]=0，跳过下一条指令执行
影响标志位 无

SZ [m].i Skip if bit i of Data Memory is 0
指令说明 判断指定数据存储器的第 i 位是否为 0，若为 0，则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示 如果 [m].i=0，跳过下一条指令执行
影响标志位 无

TABRD [m] Read table (specific page or current page) to TBLH and Data Memory
指令说明 将表格指针 (TBHP 和 TBLP，若无 TBHP 则仅 TBLP) 所指的程序代码低字节移至指定数据存储器且将高字节移至 TBLH。
功能表示 [m] ← 程序代码 (低字节)
TBLH ← 程序代码 (高字节)
影响标志位 无

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

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

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

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

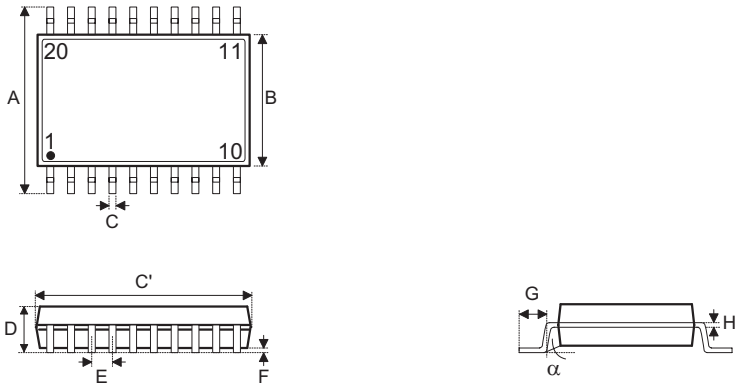
封装信息

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

封装信息的相关内容如下所示，点击可链接至 Holtek 网站相关信息页面。

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

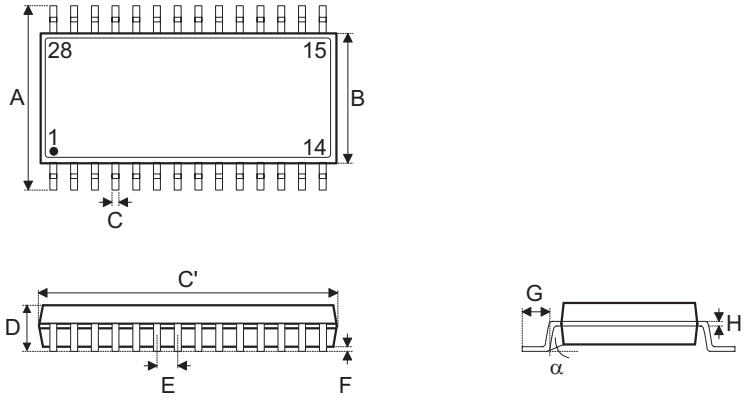
20-pin SSOP (150mil) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	典型值	最大值
A		0.236 BSC	
B		0.154 BSC	
C	0.008	—	0.012
C’		0.341 BSC	
D	—	—	0.069
E		0.025 BSC	
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

符号	尺寸 (单位：mm)		
	最小值	典型值	最大值
A		6.00 BSC	
B		3.90 BSC	
C	0.20	—	0.30
C’		8.66 BSC	
D	—	—	1.75
E		0.635 BSC	
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°

28-pin SSOP (150mil) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	典型值	最大值
A		0.236 BSC	
B		0.154 BSC	
C	0.008	—	0.012
C’		0.390 BSC	
D	—	—	0.069
E		0.025 BSC	
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

符号	尺寸 (单位：mm)		
	最小值	典型值	最大值
A		6.00 BSC	
B		3.90 BSC	
C	0.20	—	0.30
C’		9.90 BSC	
D	—	—	1.75
E		0.635 BSC	
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°