移动电源单片机

JXY-FC30LV/JXY-FC30HV

版本 : V1.00　日期 : 2018-02-24

目 录

特性

CPU 特性

- 工作电压：
  - ♦ f_{SYS}=8MHz：2.55V~5.50V
- V_{DD}=5V，系统时钟为 8MHz 时，指令周期为 0.5μs
- 提供暂停和唤醒功能，以降低功耗
- 振荡器类型：
  - ♦ 内部 RC – HIRC
  - ♦ 内部 32kHz RC – LIRC
- 多种工作模式：正常、低速、空闲和休眠
- 所有指令都可在 1~3 个指令周期内完成
- 查表指令
- 115 条指令
- 8 层堆栈
- 位操作指令

周边特性

- Flash 程序存储器：4K×16
- RAM 数据存储器：256×8
- True EEPROM 存储器：64×8
- 看门狗定时器
- 多达 30 个双向输入 / 输出口
- PB0~PB3 输出转换速率可控
- 串行接口模块，支持 I^2C 或 SPI 通信
- 软件控制的 4-SCOM 口 1/2 bias LCD 驱动器
- 3 个引脚与外部中断口共用
- 多个定时器模块用于时间测量、捕捉输入、比较匹配输出、PWM 输出及单脉冲输出
- 带死区时间控制的高精度 PWM 互补输出
- PWM 占空比可自动调节
- 2 个可产生中断的过电流保护功能
- 2 组可产生中断的过电压 / 欠电压保护功能
- USB 装置自动检测功能
- 双时基功能用以产生固定的中断信号
- 多通道 12-bit A/D 转换器
- 低电压复位功能
- 低电压检测功能
- Flash 程序存储器烧录可达 100,000 次
- Flash 程序存储器数据可保存 10 年以上
- True EEPROM 数据存储器烧录可达 1,000,000 次

- True EEPROM 数据存储器数据可保存 10 年以上
- 封装类型：
  - ♦ JXY-FC30LV：32-pin QFN
  - ♦ JXY-FC30HV：46-pin QFN

概述

该系列单片机是针对移动电源应用开发的 8 位高性能精简指令集的 Flash 型单片机，集成充电降压 PWM 控制，放电升压 PWM 控制，电量显示管理等移动电源所需的功能。具有一系列功能和特性，其 Flash 存储器可多次编程的特性给用户提供了极大的方便。存储器方面，还包含了一个 RAM 数据存储器和一个可用于存储序号、校准数据等非易失性数据的 True EEPROM 存储器。

在模拟特性方面，该系列单片机包含一个多通道 12 位 A/D 转换器、两个过电流保护模块，两组过电压 / 欠电压保护模块， PWM 占空比可自动调节的高精度 PWM 输出功能和 USB 设备自动检测功能。另外还带有多个使用灵活的定时器模块，可提供定时功能、脉冲产生功能及 PWM 产生功能。内部看门狗定时器、低电压复位和低电压检测等内部保护特性，外加优秀的抗干扰和 ESD 保护性能，确保单片机在恶劣的电磁干扰环境下可靠地运行。

该系列单片机提供了完全内建的完整的高速和低速振荡器，无需外接元件。其在不同工作模式之间动态切换的能力，为用户提供了一个优化单片机操作和减少功耗的手段。内建串行接口模块，支持 I^2C 和 SPI 传输，为设计者提供了一个易于外部硬件通信的接口。外加时基功能、I/O 使用灵活等其它特性，使这系列单片机可以很好应用于移动电源产品应用中。

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

选型表

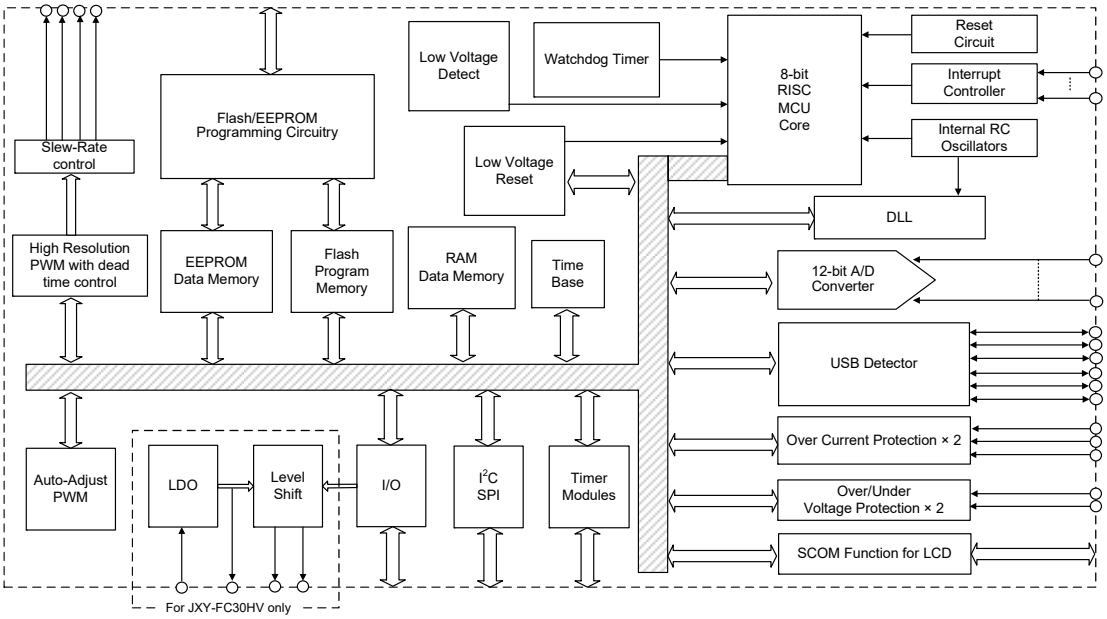
对此系列的单片机而言，大多数的特性参数都是一样的。主要差异在于 I/O 数量及是否支持 LDO 及电平转换功能。下表列出了各单片机的主要特性。

型号	V_{DD}	ROM	RAM	EEPROM	I/O	外部中断	TM 模块	H.R. PWM	A/D 转换器
JXY-FC30LV	2.55V~5.5V	4K×16	256×8	64×8	30	3	16-bit STM×1 10-bit PTM×1	√	12-bit×14
JXY-FC30HV					28	3			

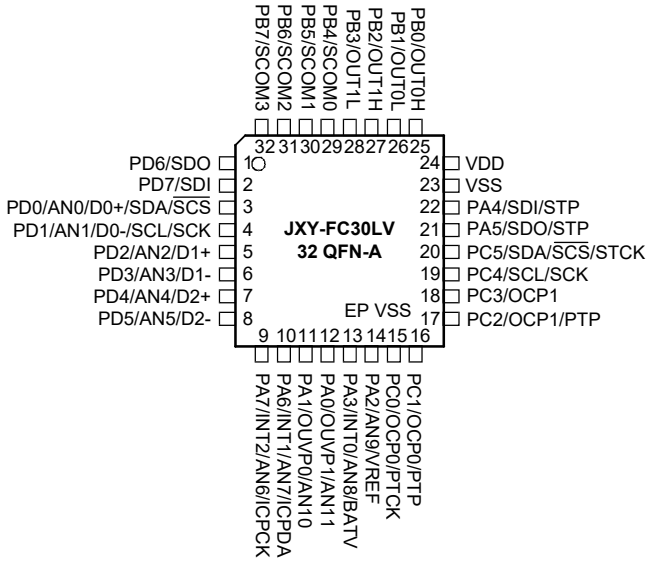
型号	自动调节 PWM 占空比	参考电压	OCP	OUVP	LDO	电平转换	PE+	Q.C 2.0	堆栈	封装形式
JXY-FC30LV	2	√	2	2	—	—	—	—	8	32QFN
JXY-FC30HV					5V	2	√	√		46QFN

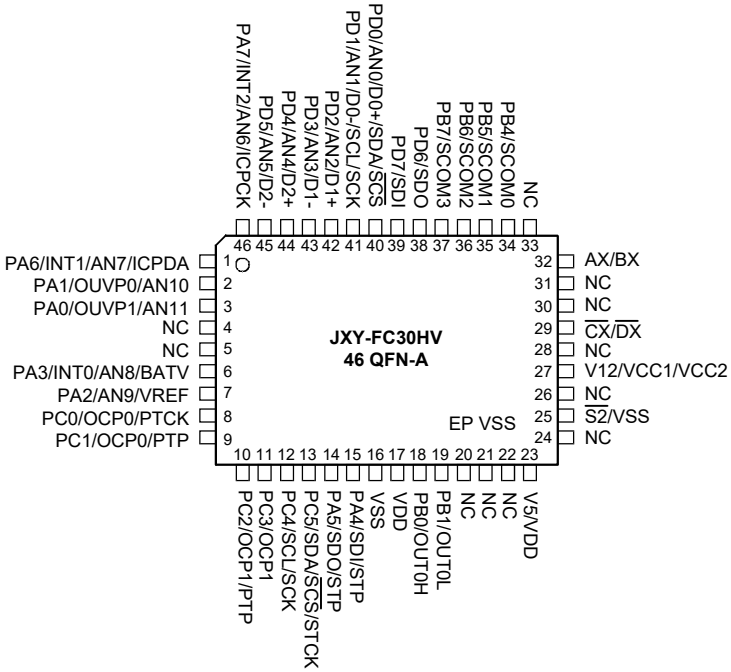
注：H.R. PWM：带死区时间插入控制的高精度互补 PWM 输出，若 HIRC 频率为 8MHz，占空比周期精度为 7.8ns。

方框图


Low Voltage Detect
Watchdog Timer
8-bit RISC MCU Core
Reset Circuit
Interrupt Controller
Internal RC Oscillators
Flash/EEPROM Programming Circuitry
Slew-Rate control
Low Voltage Reset
DLL
High Resolution PWM with dead time control
EEPROM Data Memory
Flash Program Memory
RAM Data Memory
Time Base
12-bit A/D Converter
USB Detector
Over Current Protection × 2
Over/Under Voltage Protection × 2
SCOM Function for LCD
Auto-Adjust PWM
LDO
Level Shift
I/O
I2C SPI
Timer Modules
For JXY-FC30HV only

引脚图


PB7/SCOM3
PB6/SCOM2
PB5/SCOM1
PB4/SCOM0
PB3/OUT1L
PB2/OUT1H
PB1/OUT0L
PB0/OUT0H
32 31 30 29 28 27 26 25
PD6/SDO 1
PD7/SDI 2
PD0/AN0/D0+/SDA/SCS 3
PD1/AN1/D0-/SCL/SCK 4
PD2/AN2/D1+ 5
PD3/AN3/D1- 6
PD4/AN4/D2+ 7
PD5/AN5/D2- 8
JXY-FC30LV
32 QFN-A
24 VDD
23 VSS
22 PA4/SDI/STP
21 PA5/SDO/STP
20 PC5/SDA/SCS/STCK
19 PC4/SCL/SCK
18 PC3/OCP1
17 PC2/OCP1/PTP
EP VSS
9 10 11 12 13 14 15 16
PA7/INT2/AN6/ICPCK
PA6/INT1/AN7/ICPDA
PA1/OUVP0/AN10
PA0/OUVP1/AN11
PA3/INT0/AN8/BATV
PA2/AN9/VREF
PC0/OCP0/PTCK
PC1/OCP0/PTP



注：1. 若共用脚同时有多种输出，所需的引脚功能可通过对应的引脚共用功能选择寄存器设置。
2. 引脚图中“EP”即“Exposed Pad”，连接到地。

引脚说明

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

引脚名称	功能	OPT	I/T	O/T	说明
JXY-FC30LV / JXY-FC30HV					
PA0/OUVP1/AN11	PA0	PAWU PAPU PAPS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	OUVP1	PAPS0	AN	—	OVP/UVP 1 输入
	AN11	PAPS0	AN	—	A/D 转换器外部信号输入通道
PA1/OUVP0/AN10	PA1	PAWU PAPU PAPS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	OUVP0	PAPS0	AN	—	OVP/UVP 0 输入
	AN10	PAPS0	AN	—	A/D 转换器外部信号输入通道
PA2/AN9/VREF	PA2	PAWU PAPU PAPS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	AN9	PAPS0	AN	—	A/D 转换器外部信号输入通道
	VREF	PAPS0	AN	—	ADC、OCPn/OUVPn、USB 电路 DAC 外部参考电压输入

引脚名称	功能	OPT	I/T	O/T	说明
PA3/INT0/AN8/BATV	PA3	PAWU PAPU PAPS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	AN8	PAPS0	AN	—	A/D 转换器外部信号输入通道
	BATV	PAPS0	AN	—	BATV 输入
	INT0	PAPS0 INTEG INTC0	ST	—	外部中断 0 输入
PA4/SDI/STP	PA4	PAWU PAPU PAPS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	SDI	PAPS1 PRM	ST	—	SPI 串行数据输入
	STP	PAPS1	ST	CMOS	STM 输出
PA5/SDO/STP	PA5	PAWU PAPU PAPS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	SDO	PAPS1 PRM	—	CMOS	SPI 串行数据输出
	STP	PAPS1	ST	CMOS	STM 输出
PA6/INT1/AN7/ICPDA	PA6	PAWU PAPU PAPS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	AN7	PAPS1	AN	—	A/D 转换器外部信号输入通道
	INT1	PAPS1 INTEG INTC0	ST	—	外部中断 1 输入
	ICPDA	—	ST	CMOS	ICP 数据 / 地址
PA7/INT2/AN6/ICPCK	PA7	PAWU PAPU PAPS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	INT2	PAPS1 INTEG INTC0	ST	—	外部中断 2 输入
	AN6	PAPS1	AN	—	A/D 转换器外部信号输入通道
	ICPCK	—	ST	—	ICP 时钟
PB0/OUT0H	PB0	PBPU PBPS	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	OUT0H	PBPS	—	CMOS	PWM0 输出
PB1/OUT0L	PB1	PBPU PBPS	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	OUT0L	PBPS	—	CMOS	PWM0 输出
PB2/OUT1H	PB2	PBPU PBPS	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	OUT1H	PBPS	—	CMOS	PWM1 输出

引脚名称	功能	OPT	I/T	O/T	说明
PB3/OUT1L	PB3	PBPU PBPS	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	OUT1L	PBPS	—	CMOS	PWM1 输出
PB4/SCOM0	PB4	PBPU PBPS	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCOM0	PBPS	—	SCOM	软件控制的 COM 输出
PB5/SCOM1	PB5	PBPU PBPS	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCOM1	PBPS	—	SCOM	软件控制的 COM 输出
PB6/SCOM2	PB6	PBPU PBPS	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCOM2	PBPS	—	SCOM	软件控制的 COM 输出
PB7/SCOM3	PB7	PBPU PBPS	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCOM3	PBPS	—	SCOM	软件控制的 COM 输出
PC0/OCP0/ PTCK	PC0	PCPU PCPS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTCK	PCPS0	ST	—	PTM 输入
	OCP0	PCPS0	AN	—	OCP0 输入
PC1/OCP0/ PTP	PC1	PCPU PCPS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTP	PCPS0	ST	CMOS	PTM 输出或捕捉功能输入
	OCP0	PCPS0	AN	—	OCP0 输入
PC2/OCP1/ PTP	PC2	PCPU PCPS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PTP	PCPS0	ST	CMOS	PTM 输出或捕捉功能输入
	OCP1	PCPS0	AN	—	OCP1 输入
PC3/OCP1	PC3	PCPU PCPS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	OCP1	PCPS0	AN	—	OCP1 输入
PC4/SCL/ SCK	PC4	PCPU PCPS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SCL	PCPS1 PRM	ST	CMOS	I^2C 时钟
	SCK	PCPS1 PRM	ST	CMOS	SPI 串行时钟
PC5/SDA/ $\overline{SCS}$/STCK	PC5	PCPU PCPS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDA	PCPS1	ST	NMOS	I^2C 数据输入 / 输出
	$\overline{SCS}$	PCPS1 PRM	ST	CMOS	SPI 从机选择
	STCK	PCPS1	ST	—	STM 输入

引脚名称	功能	OPT	I/T	O/T	说明
PD0/AN0/ D0+/SDA/ $\overline{SCS}$	PD0	PDPU PDPS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN0	PDPS0	AN	—	A/D 转换器外部信号输入通道
	D0+	PDPS0	—	AN	USB D0+ 0.6V 电压 V_{DP_SRC} 输出
	SDA	PDPS0 PRM	ST	NMOS	I^2C 数据输入 / 输出
	$\overline{SCS}$	PDPS0 PRM	ST	CMOS	SPI 从机选择
PD1/AN1/ D0-/ SCL/SCK	PD1	PDPU PDPS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN1	PDPS0	AN	—	A/D 转换器外部信号输入通道
	D0-	PDPS0	AN	—	USB 电源模式检测输入
	SCL	PDPS0 PRM	ST	NMOS	I^2C 时钟
	SCK	PDPS0 PRM	ST	CMOS	SPI 串行时钟
PD2/AN2/ D1+	PD2	PDPU PDPS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN2	PDPS0	AN	—	A/D 转换器外部信号输入通道
	D1+	PDPS0	—	AN	USB DAC0 输出
PD3/AN3/ D1-	PD3	PDPU PDPS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN3	PDPS0	AN	—	A/D 转换器外部信号输入通道
	D1-	PDPS0	—	AN	USB DAC1 输出
PD4/AN4/ D2+	PD4	PDPU PDPS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN4	PDPS1	AN	—	A/D 转换器外部信号输入通道
	D2+	PDPS1	—	AN	USB DAC2 输出
PD5/AN5/ D2-	PD5	PDPU PDPS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN5	PDPS1	AN	—	A/D 转换器外部信号输入通道
	D2-	PDPS1	—	AN	USB DAC3 输出
PD6/SDO	PD6	PDPU PDPS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDO	PDPS1 PRM	—	CMOS	SPI 串行数据输出
PD7/SDI	PD7	PDPU PDPS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	SDI	PDPS1 PRM	ST	—	SPI 串行数据输入
VDD	VDD	—	PWR	—	数字正电源
VSS	VSS	—	PWR	—	数字负电源 , 接地
仅 JXY-FC30HV					
V5	V5	—	—	PWR	5V LDO 输出

引脚名称	功能	OPT	I/T	O/T	说明
V12/VCC	V12/VCC	—	PWR	—	LDO 电源电压和电平转换器输出驱动电源
AX, BX	AX, BX	—	—	—	电平转换器输出，与 PB3/OUT1L 引脚内部连接
$\overline{CX}$, $\overline{DX}$	$\overline{CX}$, $\overline{DX}$	—	—	—	电平转换器输出，与 PB2/OUT1H 引脚内部连接

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

电平转换输入 / 输出关系及复位状态

电平转换输出	电平转换输入		复位状态
	A 输入 = 低	A 输入 = 高	
AX, BX	低	高	高

电平转换输出	电平转换输入		复位状态
	C 输入 = 低	C 输入 = 高	
$\overline{CX}$, $\overline{DX}$	高	低	低

极限参数

电源供应电压 ..... V_{SS}-0.3V ~ V_{SS}+6.0V
端口输入电压 ..... V_{SS}-0.3V ~ V_{DD}+0.3V
储存温度 ..... -50° C ~ 125° C
工作温度 ..... -40° C ~ 85° C
I_{OH}总电流 ..... -120mA
I_{OL}总电流 120mA
总功耗 600mW

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

直流电气特性

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 (HIRC)	—	$f_{SYS} = f_{HIRC}$ = 8MHz	V_{LVR}	—	5.5	V
			$f_{SYS} = f_{HIRC}$ /2 = 4MHz	V_{LVR}	—	5.5	V
			$f_{SYS} = f_{HIRC}$ /4 = 2MHz	V_{LVR}	—	5.5	V
			$f_{SYS} = f_{HIRC}$ /8 = 1MHz	V_{LVR}	—	5.5	V
	工作电压 (LIRC)	—	$f_{SYS} = f_{LIRC}$ = 32kHz	V_{LVR}	—	5.5	V
I_{DD}	工作电流 (HIRC)	3V	无负载，所有设备关闭 $f_{SYS} = f_{HIRC}$ /2 = 4MHz	—	0.4	0.6	mA
		5V		—	0.8	1.5	mA
		3V	无负载，所有设备关闭 $f_{SYS} = f_{HIRC}$ = 8MHz	—	0.8	1.2	mA
		5V		—	1.6	2.4	mA
	工作电流 (LIRC)	3V	无负载，所有设备关闭 $f_{SYS} = f_{LIRC}$ = 32kHz	—	10	20	μA
		5V		—	30	50	μA
I_{STB}	待机电流 (SLEEP 模式)	3V	无负载，所有设备关闭 WDT on	—	1.5	3	μA
		5V		—	3	5	μA
	待机电流 (IDLE0 模式)	3V	无负载，所有设备关闭 f_{SUB} on	—	3	5	μA
		5V		—	5	10	μA
	待机电流 (IDLE1 模式 , HIRC)	3V	无负载，所有设备关闭 f_{SUB} on, $f_{SYS} = f_{HIRC}$/2 = 4MHz	—	360	500	μA
		5V		—	700	900	μA
		3V	无负载，所有设备关闭 f_{SUB} on, $f_{SYS} = f_{HIRC}$ = 8MHz	—	360	500	μA
		5V		—	700	1000	μA
V_{IL}	I/O 口低电平输入电压	5V	—	0	—	1.5	V
		—	—	0	—	$0.2V_{DD}$	V
V_{IH}	I/O 口高电平输入电压	5V	—	3.5	—	5	V
		—	—	$0.8V_{DD}$	—	V_{DD}	V
I_{OL}	I/O 口灌电流 (除 PB0~PB3)	3V	$V_{OL} = 0.1V_{DD}$	22.6	45.2	—	mA
		5V		37.5	75	—	mA
I_{OH}	I/O 口源电流 (除 PB0~PB3)	3V	$V_{OH} = 0.9V_{DD}$	-5.12	-10.24	—	mA
		5V		-12.8	-25.6	—	mA
R_{PH}	I/O 口上拉电阻	3V	—	20	60	100	kΩ
		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 (HIRC)	V_{LVR}~5.5V	$f_{SYS} = f_{HIRC}$ = 8MHz	—	8	—	MHz
	系统时钟 (LIRC)	V_{LVR}~5.5V	$f_{SYS} = f_{LIRC}$ = 32kHz	—	32	—	kHz
f_{HIRC}	内部高速 RC 振荡器 (HIRC)	5V	Ta = 25°C	-2%	8	+2%	MHz
		5V ± 0.5V	Ta = 0°C ~ 70°C	-5%	8	+5%	MHz
		5V ± 0.5V	Ta = -40°C ~ 85°C	-7%	8	+7%	MHz
		V_{LVR}~5.5V	Ta = 0°C ~ 70°C	-7%	8	+7%	MHz
		V_{LVR}~5.5V	Ta = -40°C ~ 85°C	-10%	8	+10%	MHz
f_{LIRC}	内部低速 RC 振荡器 (LIRC)	5V	Ta = 25°C	-10%	32	+10%	kHz
		5V ± 0.5V	Ta = -40°C ~ 85°C	-40%	32	+40%	kHz
		V_{LVR}~5.5V	Ta = -40°C ~ 85°C	-50%	32	+60%	kHz
t_{TCK}	xTCK 引脚最小输入脉宽	—	—	0.3	—	—	μs
t_{INT}	外部中断最小输入脉宽	—	—	10	—	—	μs
t_{RSTD}	系统复位延迟时间 (上电复位，LVR 硬件复位，LVRC/WDTC 软件复位)	—	—	8.3	16.7	33.3	ms
	系统复位延迟时间 (WDT 溢出硬件复位)	—	—	8.3	16.7	33.3	ms
t_{SST}	系统启动时间 (从 f_{SYS} off 的 Pown Down 状态下唤醒)	—	$f_{SYS} = f_{HIRC}$ ~ f_{HIRC}/64	16	—	—	t_{HIRC}
		—	$f_{SYS} = f_{LIRC}$	2	—	—	t_{LIRC}
	系统启动时间 (正常模式↔低速模式)	—	f_{HIRC} off → on (HTO = 1)	16	—	—	t_{HIRC}
	系统启动时间 (从 f_{SYS} on 的 Pown Down 状态下唤醒)	—	$f_{SYS} = f_{HIRC}$ ~ f_{HIRC}/64	2	—	—	t_H
		—	$f_{SYS} = f_{LIRC}$	2	—	—	t_{LIRC}
	系统启动时间 (WDT 溢出硬件冷复位)	—	—	0	—	—	t_H
f_{I2C}	系统频率 (I^2C 标准模式 -100kHz)	—	无去抖	2	—	—	MHz
		—	2 个系统时钟去抖	4	—	—	MHz
		—	4 个系统时钟去抖	8	—	—	MHz
	系统频率 (I^2C 快速模式 -400kHz)	—	无去抖	5	—	—	MHz
		—	2 个系统时钟去抖	10	—	—	MHz
		—	4 个系统时钟去抖	20	—	—	MHz
t_{SRESET}	最小软件复位时间	—	—	45	90	120	μs

A/D 转换器电气特性

若无其它特别说明，Ta= -40°C~85°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DD}	A/D 转换器工作电压	—	—	2.55	—	5.5	V
V_{ADI}	A/D 转换器输入电压	—	—	0	—	V_{REF}	V
V_{REF}	A/D 转换器参考电压	—	—	2	—	V_{DD}	V
DNL	非线性微分误差	3V	V_{REF} = V_{DD}, t_{ADCK} = 0.5μs	-3	—	+3	LSB
		5V	V_{REF} = V_{DD}, t_{ADCK} = 0.5μs				
		3V	V_{REF} = V_{DD}, t_{ADCK} = 10μs				
		5V	V_{REF} = V_{DD}, t_{ADCK} = 10μs				
INL	非线性积分误差	3V	V_{REF} = V_{DD}, t_{ADCK} = 0.5μs	-4	—	+4	LSB
		5V	V_{REF} = V_{DD}, t_{ADCK} = 0.5μs				
		3V	V_{REF} = V_{DD}, t_{ADCK} = 10μs				
		5V	V_{REF} = V_{DD}, t_{ADCK} = 10μs				
I_{ADC}	A/D 转换器使能的额外电流	3V	无负载，t_{ADCK} = 0.5μs	—	1	2	mA
		5V		—	1.5	3	mA
t_{ADCK}	A/D 转换器时钟周期	—	—	0.5	—	10	μs
t_{ON2ST}	A/D 转换器 On-to-Start 时间	—	—	4	—	—	μs
t_{ADS}	A/D 采样时间	—	—	—	4	—	t_{ADCK}
t_{ADC}	A/D 转换时间 (包括采样和保持时间)	—	—	—	16	—	t_{ADCK}
GERR	增益误差	3V	V_{REF} = V_{DD}	-4	—	+4	LSB
		5V	V_{REF} = V_{DD}	-4	—	+4	LSB
OSRR	偏移误差	3V	V_{REF} = V_{DD}	-4	—	+4	LSB
		5V	V_{REF} = V_{DD}	-4	—	+4	LSB
V_R	OPA 输出电压	5V	Ta = 25°C	-1%	2.4	+1%	V
R_{PH}	VREF 引脚上拉电阻	3V	—	0.7	1	1.5	kΩ
		5V	—	0.7	1	1.3	kΩ
R_{BATV}	BATV_R1+ BATV_R2 电阻总和	3V	—	2	4	6	kΩ
		5V	—	2	4	6	kΩ
RR_{BATV}	BATV_R1/BATV_R2 电阻值比	3V	—	-1%	1:1	+1%	—
		5V	—	-1%	1:1	+1%	—
R_{OUVPn}	OUVPn_R1 + OUVPn_R0 电阻和	3V	—	1.5	3	4.5	kΩ
		5V	—	1.5	3	4.5	kΩ
RR_{OUVPn}	OUVPn_R1 / OUVPn_R0 电阻比值	3V	—	-2%	1:2	+2%	—
		5V	—	-2%	1:2	+2%	—

LVD & LVR 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_{LVR}	低电压复位电压	—	LVR 使能	-5%	2.55	+5%	V
V_{LVD}	低电压检测电压	—	LVD 使能，V_{LVD} = 2.7V	-5%	2.7	+5%	V
		—	LVD 使能，V_{LVD} = 3.0V		3.0		
		—	LVD 使能，V_{LVD} = 3.3V		3.3		
		—	LVD 使能，V_{LVD} = 3.6V		3.6		
		—	LVD 使能，V_{LVD} = 4.0V		4.0		
$I_{LVRLVDBG}$	工作电流	3V	LVD 使能，LVR 使能 VDPON = 0	—	45	60	μA
		5V		—	60	90	μA
		3V	LVD 使能，LVR 使能 VDPON = 1	—	200	350	μA
		5V		—	300	450	μA
t_{LVDS}	LVDO 稳定时间	—	LVR 使能，LVD off → on	—	—	15	μs
t_{LVR}	产生 LVR 复位的低电压最短保持时间	—	—	120	240	480	μs
t_{LVD}	产生 LVD 中断的低电压最短保持时间	—	—	60	120	240	μs
I_{LVR}	LVR 使能额外电流	—	LVD 除能	—	—	10	μA
I_{LVD}	LVD 使能额外电流	—	LVD 使能	—	60	90	μA

参考电压电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{BG}	Bandgap 参考电压	—	Trim @V_{DD} = 3.15V	-5%	1.25	+5%	V
t_{BGS}	V_{BG} 开启到稳定时间	—	无负载	—	—	150	μs

注：此 V_{BG} 电压使用于 USB 自动检测电路。

转换速率控制特性

若无其它特别说明，Ta= -40°C~85°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{OL}	PB0~PB3 口灌电流	3V	V_{OL} = 0.2V_{DD}	24	60	—	mA
		5V		60	150	—	mA
I_{OH}	PB0~PB3 口源电流	3V	V_{OH} = 0.8V_{DD}	-24	-60	—	mA
		5V		-60	-150	—	mA
SR_{RISE}	PB0~PB3 口输出上升沿转换速率	5V	SLEWCn[m+1, m] = 00B (n = 0, 1; m = 0 or 2) 0.5V → 4.5V, C_{LOAD} = 1000pF	200	—	—	V/μs
		5V	SLEWCn[m+1, m] = 01B (n = 0, 1; m = 0 or 2) 0.5V → 4.5V, C_{LOAD} = 1000pF	50	60	90	V/μs
		5V	SLEWCn[m+1, m] = 10B (n = 0, 1; m = 0 or 2) 0.5V → 4.5V, C_{LOAD} = 1000pF	25	30	55	V/μs
		5V	SLEWCn[m+1, m] = 11B (n = 0, 1; m = 0 or 2) 0.5V → 4.5V, C_{LOAD} = 1000pF	10	15	32	V/μs
SR_{FALL}	PB0~PB3 口输出下降沿转换速率	5V	SLEWCn[m+1, m] = 00B (n = 0, 1; m = 0 or 2) 4.5V → 0.5V, C_{LOAD} = 1000pF	200	—	—	V/μs
		5V	SLEWCn[m+1, m] = 01B (n = 0, 1; m = 0 or 2) 4.5V → 0.5V, C_{LOAD} = 1000pF	50	60	90	V/μs
		5V	SLEWCn[m+1, m] = 10B (n = 0, 1; m = 0 or 2) 4.5V → 0.5V, C_{LOAD} = 1000pF	25	30	55	V/μs
		5V	SLEWCn[m+1, m] = 11B (n = 0, 1; m = 0 or 2) 4.5V → 0.5V, C_{LOAD} = 1000pF	10	15	32	V/μs

过电流保护电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{OCP}	OCPn 工作电流	5V	OCPnEN[1:0] = 01B, DAC V_{REF} = 2.5V	—	730	1250	μA
V_{OS_CMP}	比较器输入失调电压	5V	未校准， (OCPnCOF[4:0] = 10000B)	-15	—	15	mV
			校准后	-4	—	4	mV
V_{HYS}	比较器迟滞电压	5V	—	20	40	60	mV
V_{CM_CMP}	比较器共模电压范围	5V	—	V_{SS}	—	V_{DD}-1.4	V
V_{OS_OPA}	OPA 输入失调电压	5V	未校准， (OCPnOOF[5:0] = 100000B)	-15	—	15	mV
			校准后	-4	—	4	mV
V_{CM_OPA}	OPA 共模电压范围	3V	—	V_{SS}	—	V_{DD}-1.4	V
		5V		V_{SS}	—	V_{DD}-1.4	V
V_{OR}	OPA 最大输出电压范围	3V	—	V_{SS}+0.1	—	V_{DD}-0.1	V
		5V		V_{SS}+0.1	—	V_{DD}-0.1	V
Ga	PGA 增益精度	5V	所有增益	-5	—	5	%
DNL	非线性微分误差	5V	DAC V_{REF} = V_{DD}	—	—	±1	LSB
INL	非线性积分误差	5V	DAC V_{REF} = V_{DD}	—	—	±1.5	LSB

过压 / 欠压保护电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{OUVP}	OUVPn 工作电流	5V	UVPnEN = 1, OVPnEN = 1, DAC V_{REF} = 2.5V	—	300	500	μA
V_{OS}	输入失调电压	5V	校准后	-4	—	4	mV
V_{HYS}	迟滞	5V	—	20	40	60	mV
V_{CM}	共模电压范围	5V	—	V_{SS}	—	V_{DD} - 1.4	V
DNL	非线性微分误差	5V	DAC V_{REF} = V_{DD}	—	—	±1	LSB
INL	非线性积分误差	5V	DAC V_{REF} = V_{DD}	—	—	±1.5	LSB

延迟锁相环电气特性

Ta=25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{DLL}	DLL 工作电流	3V	DLLEN = 1	—	0.9	1.2	mA
		5V		—	1.5	2	mA
f_{DLL}	DLL 工作频率	2.2V~5.5V	f_{HIRC} = 8MHz	-10%	8	+10%	MHz
t_{DLLS}	DLL 稳定时间	2.2V~5.5V	DLLEN = 0 → 1	—	20	30	μs

LDO 稳压器电气特性

C_{LOAD}=1μF, Ta = 25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{IN}	输入电压	—	—	6	—	28	V
V_{OUT}	LDO 输出电压	—	Ta = 25°C, I_{LOAD} = 1mA, V_{IN} = V_{OUT} + 1V	-2%	5	+2%	V
		—	-40°C ≤ Ta < 85°C, I_{LOAD} = 1mA, V_{IN} = V_{OUT} + 1V	-5%	5	+5%	V
ΔV_{LOAD}	负载调整率[注 1]	—	1mA ≤ I_{LOAD} ≤ 30mA V_{IN} = V_{OUT} + 1V	—	0.09	0.18	%/mA
V_{DROP}	损失电压[注 2]	—	ΔV_{OUT} = 2%, I_{LOAD} = 1mA V_{IN} = V_{OUT} + 2V	—	—	100	mV
I_Q	静态电流	—	无负载，V_{IN} = 12V	—	2	4	μA
ΔV_{LINE}	线性稳压率	—	6V ≤ V_{IN} ≤ 28V, I_{LOAD} = 1mA	—	—	0.2	%/V
TC	温度系数	—	-40°C ≤ Ta < 85°C, V_{IN} = V_{OUT} + 1V, I_{LOAD} = 10mA	—	±0.9	±2	mV/°C

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

2. 损失电压定义：输入 V_{IN}=V_{OUT}+2V 时测其输出电压，之后调整输入电压使输出电压跌落 2%，此时输入与输出电压之差

电平转换器电气特性

Ta = 25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{SOURCE}	AX, BX, $\overline{CX}$, $\overline{DX}$ 输出源电流	—	V_{CC} = 12V, V_{OH} = 10.4V	-60	-90	—	mA
I_{SINK}	AX, BX, $\overline{CX}$, $\overline{DX}$ 输出灌电流	—	V_{CC} = 12V, V_{OL} = 1.6V	60	90	—	mA

USB 自动检测电气特性

Ta = 25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DAC}	DAC 工作电压	—	—	2.2	—	5.5	V
V_{DP_SRC}	D0+ 输出电压	—	D0+ 输出源电流 = 250μA	0.5	0.6	0.7	V
I_{DAC}	DAC 工作电流	3V	无负载	—	0.6	0.9	mA
		5V		—	1.0	1.5	mA
I_{DACSD}	DAC 关断电流	—	无负载	—	—	0.1	μA
N_R	DAC 精度	—	—	—	8	—	bits
DNL	DAC 非线性微分误差	—	无负载， DAC 参考电压 = V_{DD}	—	—	±1	LSB
INL	DAC 非线性积分误差	—	无负载， DAC 参考电压 = V_{DD}	—	—	±2	LSB

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DACO}	DAC 输出电压范围	—	DAC Code = 00H	V_{SS}	—	V_{SS} + 0.2	V
		—	DAC Code = FFH	V_{REF} - 0.2	—	V_{REF}	V
V_{REF}	参考电压	—	—	2	—	V_{DD}	V
t_{ST}	稳定时间	3V	C_{LOAD} = 50pF	—	—	5	μs
		5V	C_{LOAD} = 50pF	—	—	5	μs
R_O	R2R 输出电阻	3V	—	—	3	—	kΩ
		5V	—	—	5	—	kΩ
OSRR	偏置误差	3V	V_{REF} = V_{DD} = 3V, Data word = 128	—	—	50	mV
		5V	V_{REF} = V_{DD} = 5V, Data word = 128	—	—	80	mV
GERR	增益误差	3V	V_{REF} = V_{DD} = 3V, Data word = 128	—	—	50	mV
		5V	V_{REF} = V_{DD} = 5V, Data word = 128	—	—	80	mV
I_{DACOL}	输出灌电流	3V	Data word = 00H, V_{DACO} = 0.1V_{REF}	20	—	—	mA
		5V		40	—	—	mA
I_{DACOH}	输出源电流	3V	Data word = FFH, V_{DACO} = 0.9V_{REF}	20	—	—	mA
		5V		40	—	—	mA
I_{SC}	输出短路电流	3V	Data word = FFH	0.25	—	—	mA
		5V		0.4	—	—	mA
R_{ON}	D1+ 和 D1-，D2+ 和 D2- 之间的模拟开关电阻	5V	—	—	20	35	Ω
R_{PL}	D0+, D0- 下拉电阻	5V	—	400	700	1400	kΩ
	D1+, D1-, D2+, D2- 下拉电阻			15	20	30	kΩ
ERR	D1+, D1-, D2+, D2- 输出电压误差	5V	DAC 参考电压 = V_{DD}, DAC 数字值 = 148, D1+, D1-, D2+ 或 D2- 连接一个 150kΩ 电阻到地	2.57	2.7	2.84	V
		5V	DAC 参考电压 = V_{DD}, DAC 数字值 = 110, D1+, D1-, D2+ 或 D2- 连接一个 150kΩ 电阻到地	1.9	2.0	2.1	V
t_{VDP_SRCS}	V_{DP_SRC} 开启稳定时间	—	V_{BG} 关闭	—	—	200	μs
		—	V_{BG} 开启	—	5	10	μs

LCD SCOM 电气特性

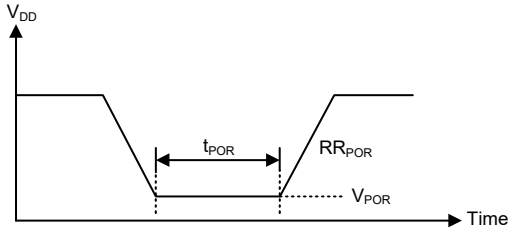
Ta = 25°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{BIAS}	LCD V_{DD}/2 偏压电流	5V	ISEL[1:0] = 00B	17.5	25	32.5	μA
			ISEL[1:0] = 01B	35	50	65	μA
			ISEL[1:0] = 10B	70	100	130	μA
			ISEL[1:0] = 11B	140	200	260	μA
V_{SCOM}	LCD COM 口 V_{DD}/2 电压	2.2V ~ 5.5V	无负载	0.475	0.5	0.525	V_{DD}

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_{POR}	上电复位电压速率	—	—	0.035	—	—	V/ms
t_{POR}	V_{DD} 保持为 V_{POR} 的最小时间	—	—	1	—	—	ms

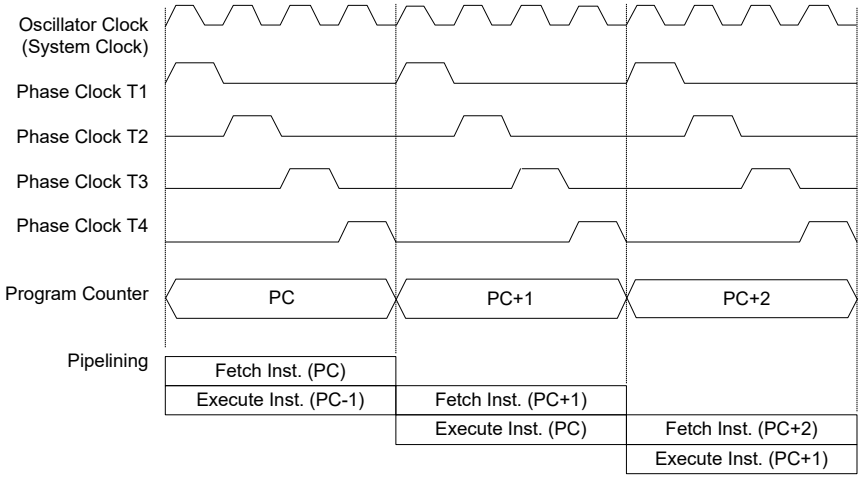


系统结构

内部系统结构是该系列单片机具有良好性能的主要因素。由于采用 RISC 结构，此系列单片机具有高运算速度和高性能的特点。通过流水线的方式，指令的取得和执行同时进行，此举使得除了跳转和调用指令需多一个指令周期外，其它大部分标准指令或扩展指令都能分别在一个或两个指令周期内完成。8 位 ALU 参与指令集中所有的运算，它可完成算术运算、逻辑运算、移位、递增、递减和分支等功能，而内部的数据路径则是以通过累加器和 ALU 的方式加以简化。有些寄存器在数据存储器中被实现，且可以直接或间接寻址。简单的寄存器寻址方式和结构特性，确保了在提供具有最大可靠度和灵活性的 I/O 和 A/D 控制系统时，仅需要少数的外部器件。使得这些单片机适用于低成本和大量生产的控制应用。

时序和流水线结构

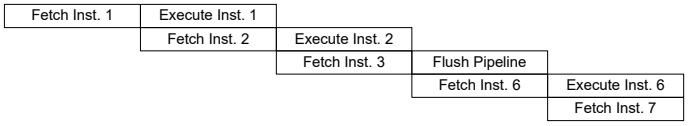
主系统时钟由 HIRC 或 LIRC 振荡器提供，它被细分为 T1~T4 四个内部产生的非重叠时序。在 T1 时间，程序计数器自动加一并抓取一条新的指令。剩下的时间 T2~T4 完成译码和执行功能，因此，一个 T1~T4 时钟周期构成一个指令周期。虽然指令的抓取和执行发生在连续的指令周期，但单片机流水线结构会保证指令在一个指令周期内被有效执行。除非程序计数器的内容被改变，如子程序的调用或跳转，在这种情况下指令将需要多一个指令周期的时间去执行。



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。此系列单片机的程序寄存器的宽度为 11 位。只有较低的 8 位，即所谓的程序计数器低字节寄存器 PCL，可以被用户直接读写。

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

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

程序计数器

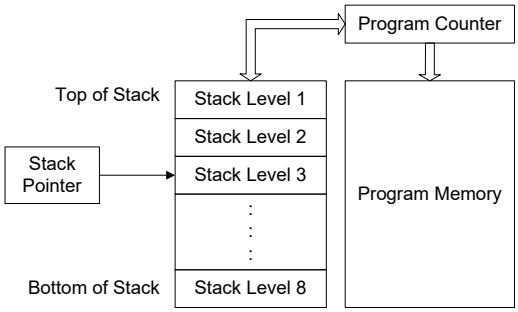
程序计数器的低字节，即程序计数器的低字节寄存器 PCL，可以通过程序控制，且它是可以读取和写入的寄存器。通过直接写入数据到这个寄存器，一个程序短跳转可直接执行，然而只有低字节的操作是有效的，跳转被限制在存储器的当前页中，即 256 个存储器地址范围内，当这样一个程序跳转要执行时，会插入一个空指令周期。程序计数器的低字节可由程序直接进行读取，PCL 的使用可能引起程序跳转，因此需要额外的指令周期。

堆栈

堆栈是一个特殊的存储空间，用来存储程序计数器中的内容。堆栈是不可读取或写入的。当前层由堆栈指针 (SP) 加以指示，同样也是不可读写的。在子程序调用或中断响应服务时，程序计数器的内容被压入到堆栈中。当子程序或中断响应结束时，返回指令 (RET 或 RETI) 使程序计数器从堆栈中重新得到它以前的值。当一个芯片复位后，堆栈指针将指向堆栈顶部。

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

若堆栈溢出，则首个存入堆栈的程序计数器数据将会丢失。



算术逻辑单元 – ALU

算术逻辑单元是单片机中很重要的部分，执行指令集中的算术和逻辑运算。ALU 连接到单片机的数据总线，在接收相关的指令码后执行需要的算术与逻辑操作，并将结果存储在指定的寄存器，当 ALU 计算或操作时，可能导致进位、借位或其它状态的改变，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：

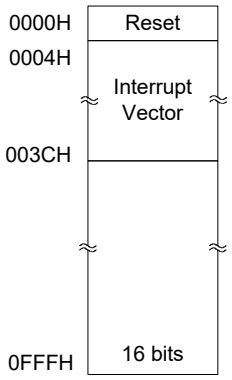
- 算术运算：
  ADD，ADDM，ADC，ADCM，SUB，SUBM，SBC，SBCM，DAA
  LADD，LADDM，LADC，LADCM，LSUB，LSUBM，LSBC，LSBCM，LDAA
- 逻辑运算：
  AND，OR，XOR，ANDM，ORM，XORM，CPL，CPLA
  LAND，LANDM，LOR，LORM，LXOR，LXORM，LCPL，LCPLA
- 移位运算：
  RRA，RR，RRCA，RRC，RLA，RL，RLCA，RLC
  LRRA，LRR，LRRCA，LRRC，LRLA，LRL，LRLCA，LRLC
- 递增和递减：
  INCA，INC，DECA，DEC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

Flash 程序存储器

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

结构

程序存储器的容量为 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。



程序存储器结构

特殊向量

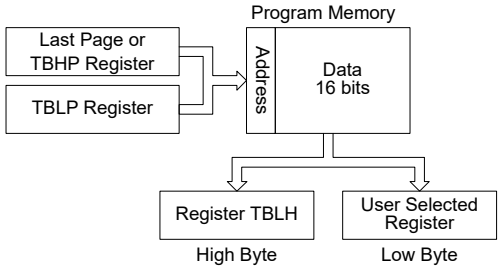
程序存储器内部某些地址保留用做诸如复位和中断入口等特殊用途。地址 0000H 是芯片复位后的程序起始地址。在芯片复位之后，程序将跳到这个地址并开始执行。

查表

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

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

下图是查表中寻址 / 数据流程：



查表范例

以下范例说明表格指针和表格数据如何被定义和执行。这个例子使用的表格数据用 ORG 伪指令储存在存储器中。ORG 指令的值“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 0F06H，即最后一页起始地址后的第六个地址。值得注意的是，假如“TABRD [m]”或“LTABRD [m]”指令被使用，则表格指针指向 TBHP 指定的页中 TBLP 指定的地址。在这个例子中，表格数据的高字节等于零，而当“TABRD [m]”或“LTABRD [m]”指令被执行时，此值将会自动的被传送到 TBLH 寄存器。

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

表格读取程序范例

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

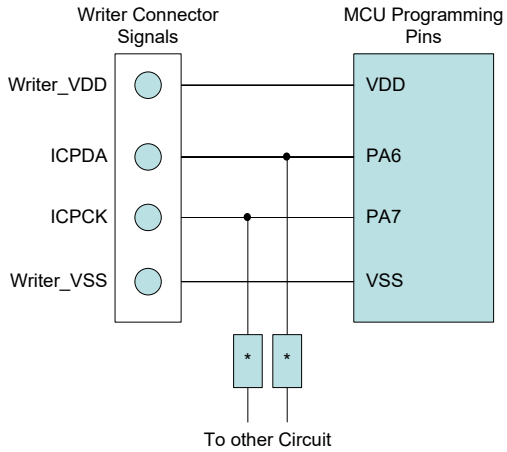
在线烧录

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

烧录器引脚名称	MCU 烧录引脚名称	功能
ICPDA	PA6	串行数据 / 地址烧录
ICPCK	PA7	时钟烧录
VDD	VDD	电源
VSS	VSS	地

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

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



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

数据存储器

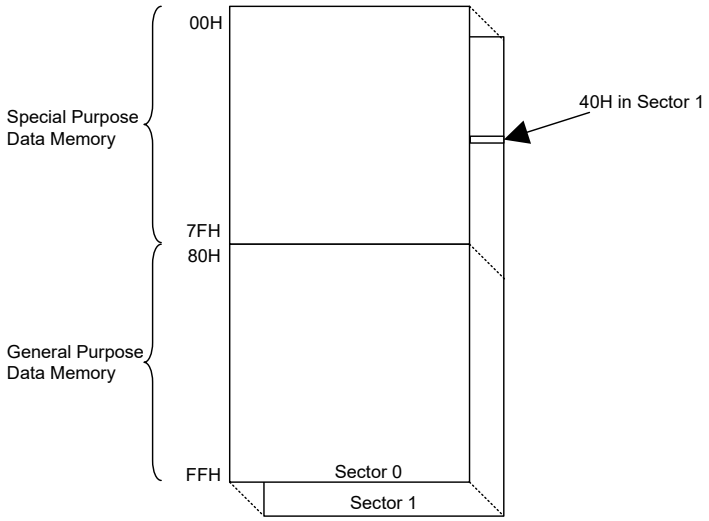
数据存储器是内容可更改的 8 位 RAM 内部存储器，用来储存临时数据。

结构

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

整个 8-bit 宽度的数据存储器被分为两个 sector。大部分特殊功能数据寄存器可在所有 sector 被访问，处于“40H”地址的 EEC 寄存器却只能在 sector 1 中被访问到。切换不同区域可通过设置存储器指针实现。此单片机的数据存储器的起始地址是“00H”。

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



数据存储器结构

数据存储器寻址

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

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

通用数据存储器

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

特殊功能数据存储器

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

	Bank 0	Bank 1
00H	IAR0	
01H	MP0	
02H	IAR1	
03H	MP1L	
04H	MP1H	
05H	ACC	
06H	PCL	
07H	TBLP	
08H	TBLH	
09H	TBHP	
0AH	STATUS	
0BH		
0CH	IAR2	
0DH	MP2L	
0EH	MP2H	
0FH	LVDC	
10H	SMOD	
11H	CTRL	
12H		
13H	LVRC	
14H	PA	
15H	PAC	
16H	PAPU	
17H	PAWU	
18H	SIMC0	
19H	SIMC1	
1AH	SIMD	
1BH	SIMA/SIMC2	
1CH	SIMTOC	
1DH	OUVP1PC	
1EH	EEA	
1FH	EED	
20H	SADOL	
21H	SADOH	
22H	SADC0	
23H	SADC1	
24H	ADJ0DT	STMC0
25H	ADJ0S	STMC1
26H	ADJ0C	STMDL
27H	ADJ0MAXH	STMDH
28H	ADJ0MAXL	STMAL
29H	ADJ0MINH	STMAH
2AH	ADJ0MINL	STMRP
2BH	ADJ0BH	PTMC0
2CH	ADJ0BL	PTMC1
2DH	ADJ1DT	PTMDL
2EH	ADJ1S	PTMDH
2FH	ADJ1C	PTMAL
30H	ADJ1MAXH	PTMAH
31H	ADJ1MAXL	PTMRPL
32H	ADJ1MINH	PTMRPH
33H	ADJ1MINL	
34H	ADJ1BH	
35H	ADJ1BL	
36H	SLEWC1	
37H	PC	
38H	PCC	
39H	PCPU	
3AH	PD	
3BH	PDC	
3CH	PDPU	
3DH	PB	
3EH	PBC	
3FH	PBPU	

	Bank 0	Bank 1
40H		EEC
41H	PWM0P	
42H	PWM0D	
43H	DLL0	
44H	PWM0C	
45H		
46H	PWM1P	
47H	PWM1D	
48H	DLL1	
49H	PWM1C	
4AH	OUVP0C3	
4BH	OVP0DA	
4CH	UVP0DA	
4DH	OUVP0C1	
4EH	OUVP0C2	
4FH	OUVP0C0	
50H	INTEG	
51H	INTC0	
52H	INTC1	
53H	INTC2	
54H	INTC3	
55H	MFI0	
56H	MFI1	
57H	DLLC	
58H	SLEWC0	
59H	OUVP0PC	
5AH	OUVP1C3	
5BH	SWS0	
5CH	SWS1	
5DH	OUTPC0	
5EH	WDTC	
5FH	TBC	
60H	OCP0C0	
61H	OCP0C1	
62H	OCP0DA	
63H	OCP0OCAL	
64H	OCP0CCAL	
65H	OCP1C0	
66H	OCP1C1	
67H	OCP1DA	
68H	OCP1OCAL	
69H	OCP1CCAL	
6AH	OCPPC	
6BH	OVP1DA	
6CH	UVP1DA	
6DH	OUVP1C0	
6EH	OUVP1C1	
6FH	OUVP1C2	
70H	SCOMC	
71H	PAPS0	
72H	PAPS1	
73H	PBPS	
74H	PCPS0	
75H	PCPS1	
76H	PDPS0	
77H	PDPS1	
78H	PRM	
79H	ADUDA0	
7AH	ADUDA1	
7BH	ADUDA2	
7CH	ADUDA3	
7DH	ADUC0	
7EH	ADUC1	
7FH	ADUC2	

: 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，IAR2

间接寻址寄存器 IAR0、IAR1 和 IAR2 的地址虽位于数据存储区，但其并没有实际的物理地址。间接寻址的方法准许使用间接寻址指针做数据操作，以取代定义实际存储器地址的直接存储器寻址方法。在间接寻址寄存器 IAR0、IAR1 和 IAR2 上的任何动作，将对间接寻址指针 MP0、MP1L/MP1H 或 MP2L/MP2H 所指定的存储器地址产生对应的读 / 写操作。它们总是成对出现，IAR0 和 MP0 可以访问 Sector 0，而 IAR1 和 MP1L/MP1H、IAR2 和 MP2L/MP2H 可以访问任何 Sector。因为这些间接寻址寄存器不是实际存在的，直接读取将返回“00H”的结果，而直接写入此寄存器则不做任何操作。

存储器指针 – MP0，MP1L/MP1H，MP2L/MP2H

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

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

间接寻址程序举例

- **Example 1**

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

Example 2

```
data .section 'data'
adres1 db ?
adres2 db ?
adres3 db ?
adres4 db ?
block  db ?
code .section at 0 'code'
org 00h
start:
mov a,04h                  ; setup size of block
mov block,a
mov a,01h                  ; setup the memory sector
mov mp1h,a
mov a,offset adres1        ; Accumulator loaded with first RAM address
mov mp1l,a                 ; setup memory pointer with first RAM address
loop:
clr IAR1                   ; clear the data at address defined by MP1L
inc mp1l                   ; increment memory pointer MP1L
sdz block                  ; check if last memory location has been cleared
jmp loop
continue:
     :
```

在上面的例子中有一点值得注意，即并没有确定 RAM 地址。

使用扩展指令直接寻址程序举例

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

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

累加器 – ACC

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

程序计数器低字节寄存器 – PCL

为了提供额外的程序控制功能，程序计数器低字节设置在数据存储器的特殊功能区域内，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 SC 标志位、CZ 标志位、零标志位 (Z)、进位标志位 (C)、辅助进位标志位 (AC)、溢出标志位 (OV)、暂停标志位 (PDF) 和看门狗定时器溢出标志位 (TO) 组成。这些算术 / 逻辑操作和系统运行标志位是用来记录单片机的运行状态。

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

SC、CZ、Z、OV、AC 和 C 标志位通常反映最近运算的状态。

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

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

STATUS 寄存器

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

“x”为未知

Bit 7 **SC**：当 OV 与当前指令操作结果 MSB 执行“XOR”所得结果

Bit 6 **CZ**：不同指令不同标志位的操作结果。
对于 SUB/SUBM/LSUB/LSUBM 指令，CZ 等于 Z 标志位。
对于 SBC/SBCM/LSBC/LSBCM 指令，CZ 等于上一个 CZ 标志位与当前零标志位执行“AND”所得结果。对于其它指令，CZ 标志位无影响。

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

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

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

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

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

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

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 数据存储器容量为 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。使用一个地址和一个数据寄存器以及一个仅位于 Sector 1 中的一个控制寄存器，可以实现对 EEPROM 的单字节读写操作。

EEPROM 寄存器

有三个寄存器控制内部 EEPROM 数据存储器总的操作，地址寄存器 EEA、数据寄存器 EED 及控制寄存器 EEC。EEA 和 EED 位于所有 Sector 中，它们能像其它特殊功能寄存器一样直接被访问。EEC 仅位于 Sector 1 中，可以通过 MP1L/MP1H 和 IAR1 或 MP2L/MP2H 和 IAR2 进行间接读取或写入。由于 EEC 控制寄存器位于 Sector 1 中的“40H”，在 EEC 寄存器上的任何操作被执行前，MP1L 或 MP2L 必须先设为“40H”，MP1H 或 MP2H 被设为“01H”。

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

EEPROM 寄存器列表

EEA 寄存器

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

Bit 7~6　未定义，读为“0”

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

EED 寄存器

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

Bit 7~0　**D7~D0**：数据 EEPROM 数据
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激活写周期。写周期结束后，硬件自动将此位清零。当 WREN 未先置高时，此位置高无效。

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

Bit 0　**RD**：EEPROM 读控制位
0：读周期结束
1：读周期有效
此位为数据 EEPROM 读控制位，由应用程序将此位置高将激活读周期。读周期结束后，硬件自动将此位清零。当 RDEN 未首先置高时，此位置高无效。

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

从 EEPROM 中读取数据

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

写数据到 EEPROM

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

写保护

防止误写入的写保护有以下几种。单片机上电后控制寄存器中的写使能位将被清除以杜绝任何写入操作。上电后存储区指针高字节寄存器 MP1H 和 MP2H 将重置为“0”，这意味着数据存储区 Sector 0 被选中。由于 EEPROM 控制寄存器位于 Sector 1 中，这增加了对写操作的保护措施。在正常程序操作中确保控制寄存器中的写使能位被清除将能防止不正确的写操作。

EEPROM 写中断

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

编程注意事项

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

写数据时，WREN 位置为“1”后，WR 须立即设置为高，以确保写周期正确地执行。写周期执行前总中断位 EMI 应先清零，写周期开始执行后再将此位重新使能。注意，EEPROM 读或写周期彻底完成前单片机不能进入 IDLE 或 SLEEP 模式，否则将导致 EEPROM 读或写操作失败。

程序举例

- **从 EEPROM 中读取数据 – 轮询法**

```
MOV  A, EEPROM_ADRES        ; user defined address
MOV  EEA, A
MOV  A, 040H                ; setup memory pointer low byte MP1L
MOV  MP1L, A                ; MP1L points to EEC register
MOV  A, 01H                 ; setup Memory Pointer high byte MP1H
MOV  MP1H, A
SET  IAR1.1                 ; set RDEN bit, enable read operations
SET  IAR1.0                 ; start Read Cycle - set RD bit
BACK:
SZ   IAR1.0                 ; check for read cycle end
JMP  BACK
CLR  IAR1                   ; disable EEPROM read/write
CLR  MP1H
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 low byte MP1L
MOV  MP1L, A                ; MP1L points to EEC register
MOV  A, 01H                 ; setup Memory Pointer high byte MP1H
MOV  MP1H, A
CLR  EMI
SET  IAR1.3                 ; set WREN bit, enable write operations
SET  IAR1.2                 ; start Write Cycle - set WR bit—executed
                            ; immediately after set WREN bit
SET  EMI
BACK:
SZ   IAR1.2                 ; check for write cycle end
JMP  BACK
CLR  IAR1                   ; disable EEPROM read/write
CLR  MP1H
```

振荡器

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

振荡器概述

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

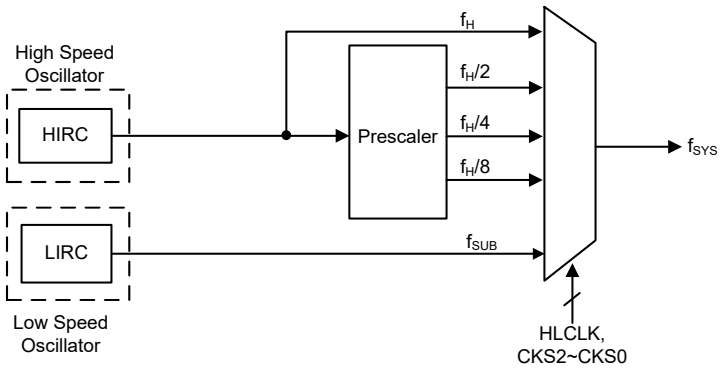
类型	名称	频率
内部高速 RC	HIRC	8MHz
内部低速 RC	LIRC	32kHz

振荡器类型

系统时钟配置

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

高速或低速振荡器的实际时钟源经由寄存器选择。低速或高速系统时钟频率由 SMOD 寄存器的 HLCLK 位及 CKS2~CKS0 位决定的。请注意，两个振荡器必须做出选择，即一个高速和一个低速振荡器。



系统时钟配置

内部 RC 振荡器 – HIRC

内部 RC 振荡器是一个集成的系统振荡器，不需其它外部器件。内部 RC 振荡器具有一固定频率：8MHz。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡频率因 V_{DD}、温度以及芯片制成工艺不同的影响减至最低程度。如果选择了该内部时钟，无需额外的引脚。

内部 32kHz 振荡器 – LIRC

内部 32kHz 系统振荡器也是一个低频振荡器。这种单片机有一个完全集成 RC 振荡器，它在 5V 电压下运行的典型频率值为 32kHz 且无需外部元件。芯片在制造时进行调整且内部含有频率补偿电路，使得振荡器因电源电压、温度及芯片制成工艺不同的影响减至最低。

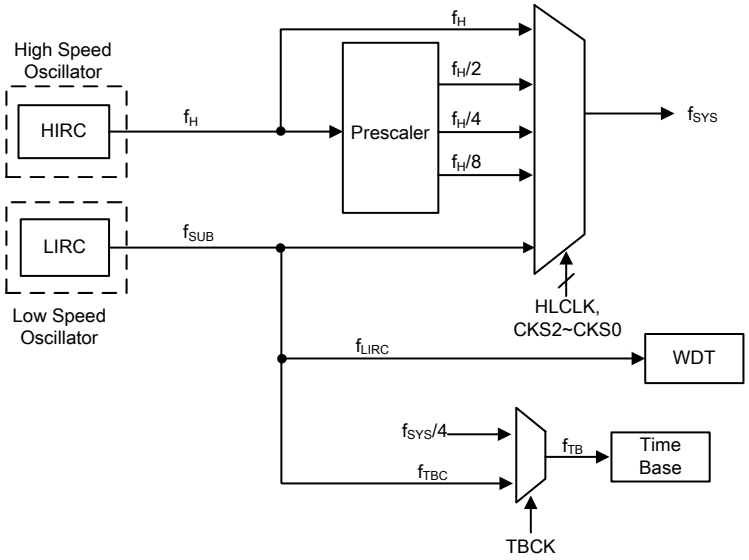
工作模式和系统时钟

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

系统时钟

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

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



设备时钟配置

注：当系统时钟源 f_{SYS} 由 f_H 到 f_{SUB} 转换时，高速振荡器将停止以节省耗电。因此，没有为外围电路提供 f_H~$f_H/8$ 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_{LIRC}	f_{TBC}
正常模式	On	f_H~f_H/8	On	On
低速模式	On	f_{SUB}	On	On
空闲模式 0	Off	Off	On	On
空闲模式 1	Off	On	On	On
休眠模式	Off	Off	On	Off

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

低速模式

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

休眠模式

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

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，系统振荡器停止，低频时钟 f_{SUB} 开启。

空闲模式 1

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

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

控制寄存器

寄存器 SMOD 和 CTRL 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_{SUB}
001：f_{SUB}
010：未定义
011：未定义
100：未定义
101：$f_H/8$
110：$f_H/4$
111：$f_H/2$

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

Bit 4 未定义，读为“0”

Bit 3 **LTO**：低速振荡器就绪标志位

0：未就绪
1：就绪

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

Bit 2 **HTO**：高速振荡器就绪标志位

0：未就绪
1：就绪

此位为高速系统振荡器就绪标志位，用于表明高速系统振荡器唤醒后何时稳定下来。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”。如果使用 HIRC 振荡器，唤醒后该位转换为高需 15~16 个时钟周期。

Bit 1 **IDLEN**：空闲模式控制位

0：除能
1：使能

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

Bit 0 **HLCLK**：系统时钟选择位

0：$f_H/2 \sim f_H/8$ 或 f_{SUB}
1：f_H

此位用于选择 f_H 或 $f_H/2 \sim f_H/8$ 还是 f_{SUB} 作为系统时钟。该位为高时选择 f_H 作为系统时钟，为低时则选择 $f_H/2 \sim f_H/8$ 或 f_{SUB} 作为系统时钟。当系统时钟由 f_H 时钟向 f_{SUB} 时钟转换时，f_H 将自动关闭以降低功耗。

CTRL 寄存器

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

“x”为未知

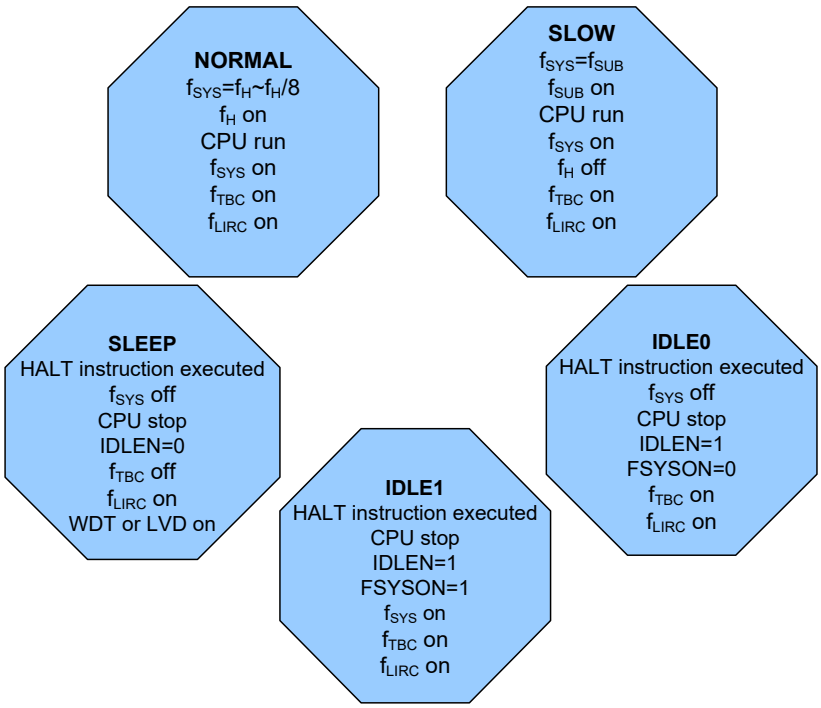
Bit 7 **FSYSON**：IDLE 模式时，f_{SYS} 控制位
0：除能
1：使能
Bit 6~3 未定义，读为“0”
Bit 2 **LVRF**：LVR 硬件复位标志位
具体描述见其它章节
Bit 1 **LRF**：LVRC 寄存器软件复位标志位
具体描述见其它章节
Bit 0 **WRF**：WDTC 寄存器软件复位标志位
具体描述见其它章节

工作模式切换

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

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

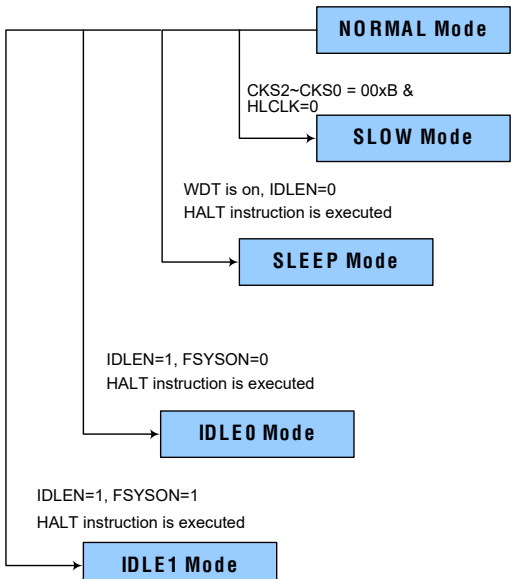
当 HLCLK 位变为低电平时，时钟源将由高速时钟源 f_H 转换成时钟源 $f_H/2$~$f_H/8$ 或 f_{SUB}。若时钟源来自 f_{SUB}，高速时钟源将停止运行以节省耗电。此时须注意，$f_H/16$ 和 $f_H/64$ 内部时钟源也将停止运行，由此会影响到内部功能的工作。所附流程图显示了单片机在不同工作模式间切换时的变化。



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。

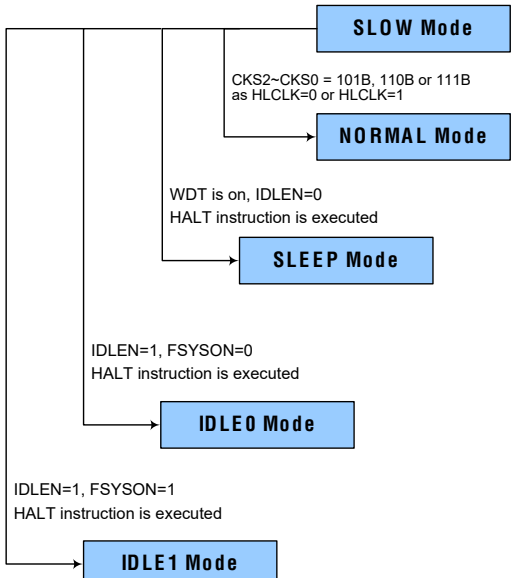
低速模式的时钟源来自 LIRC 振荡器，因此要求这些振荡器在所有模式切换动作发生前稳定下来。该动作由 SMOD 寄存器中 LTO 位控制。



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“101”、“110”或“111”。

高频时钟需要一定的稳定时间，通过检测 HTO 位的状态可进行判断。高速振荡器的稳定时间可参考交流电气特性表格。



进入休眠模式

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

- 系统时钟和时基时钟停止运行，应用程序停止在“HALT”指令处。WDT 或 LVD 继续运行，其时钟源来自 f_{LIRC}。
- 数据存储器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 的方法仅有一种即在应用程序执行“HALT”指令前设置寄存器 SMOD 中 IDLEN 位为“1”且 CTRL 寄存器中的 FSYSON 位为“0”。在上述条件下执行 HALT 指令后，将发生的情况如下：

- 系统时钟停止运行，应用程序停止在“HALT”指令处，时基时钟 f_{TBC} 和低频时钟 f_{LIRC} 将继续运行。
- 数据存储器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 的方法仅有一种即应用程序中执行“HALT”指令前需设置寄存器 SMOD 中 IDLEN 位为“1”且 CTRL 寄存器中的 FSYSON 位为“1”。在上述条件下执行该指令后，将发生的情况如下：

- 系统时钟、时基时钟 f_{TBC} 和低频时钟 f_{LIRC} 开启，应用程序停止在“HALT”指令处。
- 数据存储器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。

待机电流的注意事项

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

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

唤醒

单片机进入休眠或空闲模式之后，可以通过以下几种方式唤醒：

- PA 口下降沿
- 系统中断
- WDT 溢出

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

看门狗定时器控制寄存器

WDTC 寄存器用于控制 WDT 功能的使能 / 除能及选择溢出周期。

WDTC 寄存器

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

Bit 7~3 **WE4~WE0**：WDT 功能使能控制
10101 或 01010：使能
其它值：MCU 复位
如果由于不利的环境因素使这些位变为其它值，单片机将复位。复位动作发生在 t_{SRESET} 时间后，此复位完成后 CTRL 寄存器中的 WRF 标志位会被置位。

Bit 2~0 **WS2~WS0**：WDT 溢出周期选择
000：$2^8/f_{LIRC}$
001：$2^{10}/f_{LIRC}$
010：$2^{12}/f_{LIRC}$
011：$2^{14}/f_{LIRC}$
100：$2^{15}/f_{LIRC}$
101：$2^{16}/f_{LIRC}$
110：$2^{17}/f_{LIRC}$
111：$2^{18}/f_{LIRC}$
这三位控制 WDT 时钟源的分频比，从而实现对 WDT 溢出周期的控制。

CTRL 寄存器

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

“x”为未知

Bit 7 **FSYSON**：IDLE 模式时，f_{SYS} 控制位
具体描述见其它章节

Bit 6~3 未定义，读为“0”

Bit 2 **LVRF**：LVR 复位标志
具体描述见其它章节

Bit 1 **LRF**：LVRC 寄存器软件复位标志位
具体描述见其它章节

Bit 0 **WRF**：WDTC 寄存器软件复位标志位
0：未发生
1：发生
当发生 WDTC 控制寄存器软件复位时，此位被置为“1”。此位只能通过程序清零。

看门狗定时器操作

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

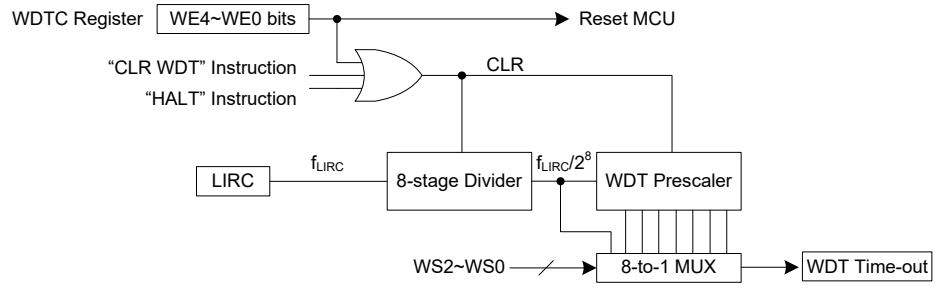
WE4 ~ WE0 位	WDT 功能
01010B 或 10101B	使能
其它值	MCU 复位

看门狗定时器功能控制

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 将被置位，仅程序计数器 PC 和堆栈指针 SP 将被复位。有三种方法可以用来清除 WDT 的内容。第一种是 WDT 复位，即将 WE4~WE0 位设置成除了 01010B 和 10101B 外的任意值；第二种是通过软件清除指令，而第三种是通过“HALT”指令。

该系列单片机只使用一条清看门狗指令“CLR WDT”。因此只要执行“CLR WDT”便清除 WDT。

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

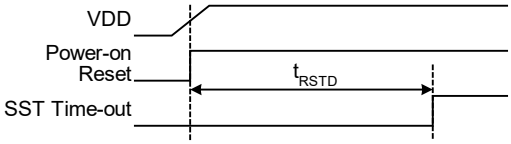
另一种复位为看门狗溢出单片机复位。不同方式的复位操作会对寄存器产生不同的影响。另一种复位为低电压复位即 LVR 复位，在电源供应电压低于 LVR 设定的值时，系统会产生 LVR 复位。

复位功能

此系列单片机提供多种内部事件触发复位：

上电复位

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



注：t_{RSTD} 为上电延迟时间，典型值为 16.7ms

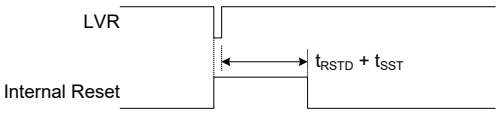
上电复位时序图

低电压复位 – LVR

单片机具有低电压复位电路，用来监测它的电源电压。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 LVD&LVR 电气特性中的 t_{LVR} 参数值。如果低电压存在时间没超过 t_{LVR} 参数的值，则 LVR 将会忽略它且不会执行复位功能。

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



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	R	R/W	R/W
POR	0	1	0	1	0	1	0	1

Bit 7 ~ 0 **LVS7 ~ LVS0**：LVR 电压选择
01010101：2.55V
00110011：2.55V
10011001：2.55V
10101010：2.55V
其它值：MCU 复位 (寄存器复位为 POR 值)
当相应的低电压出现后，且此低电压保持时间超过 t_{LVR} 值，将会产生 MCU 复位。MCU 复位后寄存器中的值与复位前保持不变。
除上述定义的值外，其它值都会产生 MCU 复位。复位操作将会在 t_{SRESET} 时间后执行。注意的是此处 MCU 复位后，寄存器的值将恢复到上电复位值。

- **CTRL 寄存器**

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

“x”为未知

Bit 7 **FSYSON**：IDLE 模式时，f_{SYS} 控制位
0：除能
1：使能

Bit 6~3 未定义，读为“0”

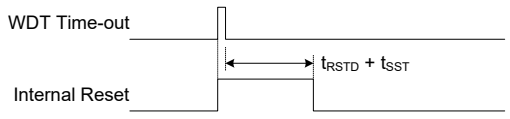
Bit 2 **LVRF**：LVR 复位标志
0：未发生
1：发生
当低电压复位情况发生时此位设置为“1”。此位只能通过程序清零。

Bit 1 **LRF**：LVRC 寄存器软件复位标志位
0：未发生
1：发生
当 LVRC 寄存器包含任何未定义的 LVR 电压值时此位设置为“1”，相当于软件复位功能。此位只能通过程序清零。

Bit 0 **WRF**：WDTC 寄存器软件复位标志位
具体描述见其它章节

正常运行时看门狗溢出复位

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

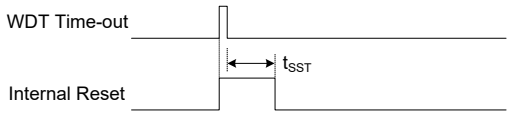


注：t_{RSTD} 为上电延迟时间，典型值为 16.7ms

正常运行时看门狗溢出复位时序图

休眠或空闲时看门狗溢出复位

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



休眠或空闲时看门狗溢出复位时序图

复位初始状态

不同的复位形式以不同的途径影响复位标志位。这些标志位，即 PDF 和 TO 位存放在状态寄存器中，由休眠或空闲模式功能或看门狗计数器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	所有定时器模块除能
输入 / 输出口	I/O 口设置为输入类型
堆栈指针	堆栈指针指向堆栈顶端

不同的复位形式对单片机内部寄存器的影响是不同的。为保证复位后程序能正常执行，了解寄存器在特定条件复位后的设置是非常重要的。下表即为不同方式复位后内部寄存器的状况。

寄存器	上电复位	WDT 溢出 (正常运行)	WDT 溢出 (空闲或休眠)
IAR0	0000 0000	0000 0000	uuuu uuuu
MP0	0000 0000	0000 0000	uuuu uuuu
IAR1	0000 0000	0000 0000	uuuu uuuu
MP1L	0000 0000	0000 0000	uuuu uuuu
MP1H	0000 0000	0000 0000	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu
TBHP	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	xx00 xxxx	xx1u uuuu	uu11 uuuu
IAR2	0000 0000	0000 0000	uuuu uuuu
MP2L	0000 0000	0000 0000	uuuu uuuu
MP2H	0000 0000	0000 0000	uuuu uuuu
LVDC	--00 -000	--00 -000	--uu -uuu
SMOD	000- 0011	000- 0011	uuu- uuuu
CTRL	0--- -x00	0--- -000	u--- -uuu
LVRC	0101 0101	0101 0101	uuuu uuuu
PA	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	uuuu uuuu
PAPU	0000 0000	0000 0000	uuuu uuuu
PAWU	0000 0000	0000 0000	uuuu uuuu
SIMC0	111- 0000	111- 0000	uuu- uuuu
SIMC1	1000 0001	1000 0001	uuuu uuuu
SIMD	xxxx xxxx	xxxx xxxx	uuuu uuuu
SIMA/SIMC2	0000 0000	0000 0000	uuuu uuuu
SIMTOC	0000 0000	0000 0000	uuuu uuuu
OUVP1PC	0000 0000	0000 0000	uuuu uuuu
EEA	--00 0000	--00 0000	--uu uuuu
EED	0000 0000	0000 0000	uuuu uuuu
SADOL(ADRFS=0)	xxxx ----	xxxx ----	uuuu ----
SADOL(ADRFS=1)	xxxx xxxx	xxxx xxxx	uuuu uuuu
SADOH(ADRFS=0)	xxxx xxxx	xxxx xxxx	uuuu uuuu
SADOH(ADRFS=1)	---- xxxx	---- xxxx	---- uuuu
SADC0	0000 0000	0000 0000	uuuu uuuu
SADC1	--00 0000	--00 0000	--uu uuuu
ADJ0DT	--00 0000	--00 0000	--uu uuuu
ADJ0S	0000 0000	0000 0000	uuuu uuuu

寄存器	上电复位	WDT 溢出 (正常运行)	WDT 溢出 (空闲或休眠)
ADJ0C	000- xx--	000- xx--	uuu- xx--
ADJ0MAXH	---- 0000	---- 0000	---- uuuu
ADJ0MAXL	0000 0000	0000 0000	uuuu uuuu
ADJ0MINH	---- 0000	---- 0000	---- uuuu
ADJ0MINL	0000 0000	0000 0000	uuuu uuuu
ADJ0BH	---- 0000	---- 0000	---- uuuu
ADJ0BL	0000 0000	0000 0000	uuuu uuuu
ADJ1DT	--00 0000	--00 0000	--uu uuuu
ADJ1S	0000 0000	0000 0000	uuuu uuuu
ADJ1C	000- xx--	000- xx--	uuu- xx--
ADJ1MAXH	---- 0000	---- 0000	---- uuuu
ADJ1MAXL	0000 0000	0000 0000	uuuu uuuu
ADJ1MINH	---- 0000	---- 0000	---- uuuu
ADJ1MINL	0000 0000	0000 0000	uuuu uuuu
ADJ1BH	---- 0000	---- 0000	---- uuuu
ADJ1BL	0000 0000	0000 0000	uuuu uuuu
SLEWC1	---- 0000	---- 0000	---- uuuu
PB	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	uuuu uuuu
PBPU	0000 0000	0000 0000	uuuu uuuu
PC	--11 1111	--11 1111	--uu uuuu
PCC	--11 1111	--11 1111	--uu uuuu
PCPU	--00 0000	--00 0000	--uu uuuu
PD	1111 1111	1111 1111	uuuu uuuu
PDC	1111 1111	1111 1111	uuuu uuuu
PDPU	0000 0000	0000 0000	uuuu uuuu
PWM0P	0000 0000	0000 0000	uuuu uuuu
PWM0D	0000 0000	0000 0000	uuuu uuuu
DLL0	0000 ----	0000 ----	uuuu ----
PWM0C	0000 0000	0000 0000	uuuu uuuu
PWM1P	0000 0000	0000 0000	uuuu uuuu
PWM1D	0000 0000	0000 0000	uuuu uuuu
DLL1	0000 ----	0000 ----	uuuu ----
PWM1C	0000 0000	0000 0000	uuuu uuuu
OUVP0C3	0001 0000	0001 0000	uuuu uuuu
OVP0DA	0000 0000	0000 0000	uuuu uuuu
UVP0DA	0000 0000	0000 0000	uuuu uuuu
OUVP0C1	--00 0000	--00 0000	--uu uuuu
OUVP0C2	0001 0000	0001 0000	uuuu uuuu
OUVP0C0	--00 0000	--00 0000	--uu uuuu
INTEG	--00 0000	--00 0000	--uu uuuu

寄存器	上电复位	WDT 溢出 (正常运行)	WDT 溢出 (空闲或休眠)
INTC0	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	uuuu uuuu
INTC2	0000 0000	0000 0000	uuuu uuuu
INTC3	0000 0000	0000 0000	uuuu uuuu
MFI0	-000 -000	-000 -000	-uuu -uuu
MFI1	-000 -000	-000 -000	-uuu -uuu
DLLC	10-- ---0	10-- ---0	uu-- ---u
SLEWC0	---- 0000	---- 0000	---- uuuu
OUVP0PC	0000 0000	0000 0000	uuuu uuuu
OUVP1C3	0001 0000	0001 0000	uuuu uuuu
SWS0	--00 0000	--00 0000	--uu uuuu
SWS1	---- -000	---- -000	---- -uuu
OUTPC0	0000 0000	0000 0000	uuuu uuuu
WDTC	0101 0011	0101 0011	uuuu uuuu
TBC	0011 -111	0011 -111	uuuu -uuu
OCP0C0	0000 0--0	0000 0--0	uuuu u--u
OCP0C1	--00 0000	--00 0000	--uu uuuu
OCP0DA	0000 0000	0000 0000	uuuu uuuu
OCP0OCAL	0010 0000	0010 0000	uuuu uuuu
OCP0CCAL	0001 0000	0001 0000	uuuu uuuu
OCP1C0	0000 0--0	0000 0--0	uuuu u--u
OCP1C1	--00 0000	--00 0000	--uu uuuu
OCP1DA	0000 0000	0000 0000	uuuu uuuu
OCP1OCAL	0010 0000	0010 0000	uuuu uuuu
OCP1CCAL	0001 0000	0001 0000	uuuu uuuu
OCPPC	0000 0000	0000 0000	uuuu uuuu
OVP1DA	0000 0000	0000 0000	uuuu uuuu
UVP1DA	0000 0000	0000 0000	uuuu uuuu
OUVP1C0	--00 0000	--00 0000	--uu uuuu
OUVP1C1	--00 0000	--00 0000	--uu uuuu
OUVP1C2	0001 0000	0001 0000	uuuu uuuu
SCOMC	-000 ----	-000 ----	-uuu ----
PAPS0	0000 0000	0000 0000	uuuu uuuu
PAPS1	0000 0000	0000 0000	uuuu uuuu
PBPS	0000 0000	0000 0000	uuuu uuuu
PCPS0	0000 0000	0000 0000	uuuu uuuu
PCPS1	---- 0000	---- 0000	---- uuuu
PDPS0	0000 0000	0000 0000	uuuu uuuu
PDPS1	0000 0000	0000 0000	uuuu uuuu
PRM	--00 -000	--00 -000	--uu -uuu
ADUDA0	0000 0000	0000 0000	uuuu uuuu

寄存器	上电复位	WDT 溢出 (正常运行)	WDT 溢出 (空闲或休眠)
ADUDA1	0000 0000	0000 0000	uuuu uuuu
ADUDA2	0000 0000	0000 0000	uuuu uuuu
ADUDA3	0000 0000	0000 0000	uuuu uuuu
ADUC0	0000 0000	0000 0000	uuuu uuuu
ADUC1	---- 0000	---- 0000	---- uuuu
ADUC2	--00 0000	--00 0000	--uu uuuu
STMC0	0000 0---	0000 0---	uuuu u---
STMC1	0000 0000	0000 0000	uuuu uuuu
STMDL	0000 0000	0000 0000	uuuu uuuu
STMDH	0000 0000	0000 0000	uuuu uuuu
STMAL	0000 0000	0000 0000	uuuu uuuu
STMAH	0000 0000	0000 0000	uuuu uuuu
STMRP	0000 0000	0000 0000	uuuu uuuu
PTMC0	0000 0---	0000 0---	uuuu u---
PTMC1	0000 0000	0000 0000	uuuu uuuu
PTMDL	0000 0000	0000 0000	uuuu uuuu
PTMDH	---- --00	---- --00	---- --uu
PTMAL	0000 0000	0000 0000	uuuu uuuu
PTMAH	---- --00	---- --00	---- --uu
PTMPRL	0000 0000	0000 0000	uuuu uuuu
PTMPRH	---- --00	---- --00	---- --uu
EEC	---- 0000	---- 0000	---- uuuu

注： “u”表示不改变
“x”表示未知
“-”表示未定义

输入 / 输出端口

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

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

寄存器名称	位							
	7	6	5	4	3	2	1	0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC5	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PBPU	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PCPU	—	—	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PC	—	—	PC5	PC4	PC3	PC2	PC1	PC0
PCC	—	—	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
PDPU	PDPU7	PDPU6	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
PD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
PDC	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0

I/O 逻辑功能寄存器列表

上拉电阻

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

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

PAPU 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **PAPU7~PAPU0**：PortA.7~ PortA.0 引脚上拉电阻控制位
0：除能
1：使能

PBPU 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
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 **PBPU7~PBPU0**：PortB.7~ PortA.0 引脚上拉电阻控制位
0：除能
1：使能

PCPU 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
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 **PCPU5~PCPU0**：PortC.5~ PortC.0 引脚上拉电阻控制位
0：除能
1：使能

PDPU 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PDPU7	PDPU6	PDPU5	PDPU4	PDPU3	PDPU2	PDPU1	PDPU0
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 **PDPU7~PDPU0**：PortD.7~ PortD.0 引脚上拉电阻控制位
0：除能
1：使能

PA 口唤醒

当使用暂停指令“HALT”迫使单片机进入休眠或空闲模式，单片机的系统时钟将会停止以降低功耗，此功能对于电池及低功耗应用很重要。唤醒单片机有很多种方法，其中之一就是使 PA 口的其中一个引脚从高电平转为低电平。这个功能特别适合于通过外部开关来唤醒的应用。PA 口的每个引脚可以通过设置 PAWU 寄存器来单独选择是否具有唤醒功能。需要注意的是，只有当引脚功能选为通用 I/O 功能且单片机处于休眠或空闲模式时，唤醒功能才会受 PAWU 控制开启，其它状态下此唤醒功能不可用。

PAWU 寄存器

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

Bit 7~0 **PAWU7~PAWU0**：PA 口唤醒功能控制位
0：除能
1：使能

输入 / 输出端口控制寄存器

每一个输入 / 输出口都具有各自的控制寄存器，即 PAC~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	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	1	1	1	1	1	1	1	1

Bit 7~0 **PAC7~PAC0**：PortA.7~ PortA.0 引脚输入 / 输出类型选择位
0：输出
1：输入

PBC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
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 **PBC7~PBC0**：PortB.7~ PortB.0 引脚输入 / 输出类型选择位
0：输出
1：输入

PCC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	PCC5	PCC4	PCC3	PCC2	PCC1	PCC0
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 **PCC5~PCC0**：PortC.5~ PortC.0 引脚输入 / 输出类型选择位
0：输出
1：输入

PDC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
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 **PDC7~PDC0**：PortD.7~ PortD.0 引脚输入 / 输出类型选择位
0：输出
1：输入

转换速率控制

通过 SLEWC0 和 SLEWC1 寄存器，可设置 PB0~PB3 端口的转换速率。具体可参考前面的转换速率控制特性表格。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SLEWC0	—	—	—	—	SLEWC03	SLEWC02	SLEWC01	SLEWC00
SLEWC1	—	—	—	—	SLEWC13	SLEWC12	SLEWC11	SLEWC10

转换速率控制寄存器列表

SLEWC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	SLEWC03	SLEWC02	SLEWC01	SLEWC00
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 3~2　**SLEWC03~SLEWC02**：PB2 输出转换速率控制
00：转换速率 = Level 0
01：转换速率 = Level 1
10：转换速率 = Level 2
11：转换速率 = Level 3

Bit 1~0　**SLEWC01~SLEWC00**：PB0 输出转换速率控制
00：转换速率 = Level 0
01：转换速率 = Level 1
10：转换速率 = Level 2
11：转换速率 = Level 3

SLEWC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	SLEWC13	SLEWC12	SLEWC11	SLEWC10
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 3~2　**SLEWC13~SLEWC12**：PB3 output slew rate selection
00：转换速率 = Level 0
01：转换速率 = Level 1
10：转换速率 = Level 2
11：转换速率 = Level 3

Bit 1~0　**SLEWC11~SLEWC10**：PB1 output slew rate selection
00：转换速率 = Level 0
01：转换速率 = Level 1
10：转换速率 = Level 2
11：转换速率 = Level 3

注：用户可参考转换速率控制特性章节针对不同应用设置所需的端口的输出转换速率。

引脚共用功能

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

引脚共用功能选择寄存器

封装中有限的引脚个数会对某些单片机功能造成影响。然而，引脚功能共用和引脚功能选择，使得小封装单片机具有更多不同的功能。单片机包含端口“x”输出功能选择寄存器 n 即 PxPSn 和部分功能输入源引脚或输出位置选择寄存器 PRM，这些寄存器可以用来选择多功能共用引脚上的特定功能。

当要使用引脚上的输入功能，相关的 PxPSn 及 PRM 寄存器需先进行合理设定。例如，SIM I^2C SDA 引脚功能被选择，需通过 PRM 寄存器选择 SDA 信号的输入引脚，且对应的引脚共用功能要通过寄存器 PxPSn 设置为 SDA 功能。但是，如果要使用外部中断功能，其对应的共用引脚功能需通过寄存器 PxPSn 选择为普通 I/O 功能。

特别需要注意的是，要确保所需的引脚共用功能被正确地选择和取消。对于大部分的引脚功能，要选择所需的引脚共用功能，首先应通过相应的引脚共用控制寄存器正确地选择该功能，然后再配置相应的外围功能设置以使能外围功能。然而对于一些数字输入引脚功能，如 INTn, xTCKn 等，当设置 PxPSn 寄存器相应位选择到此功能时，也可同时选中普通 I/O 功能。此时需特别注意，除了要进行上述的引脚共用功能寄存器设置及外围功能设置外，还应设置其对应的输入 / 输出端口控制寄存器，设置此引脚所在端口为输入状态。

要正确地取消引脚共用功能，首先应除能外围功能，然后再修改相应的引脚共用控制寄存器以选择其它的共用功能。

寄存器名称	位							
	7	6	5	4	3	2	1	0
PAPS0	PAS31	PAS30	PAS21	PAS20	PAS11	PAS10	PAS01	PAS00
PAPS1	PAS71	PAS70	PAS61	PAS60	PAS51	PAS50	PAS41	PAS40
PBPS	PBS7	PBS6	PBS5	PBS4	PBS3	PBS2	PBS1	PBS0
PCPS0	PCS31	PCS30	PCS20	PCS20	PCS11	PCS10	PCS01	PCS00
PCPS1	—	—	—	—	PCS51	PCS50	PCS41	PCS40
PDPS0	PDS31	PDS30	PDS21	PDS20	PDS11	PDS10	PDS01	PDS00
PDPS1	PDS71	PDS70	PDS61	PDS60	PDS51	PDS50	PDS41	PDS40
PRM	—	—	SDIPRM	SDOPRM	—	SCSBPRM	SCKSCLPRM	SDAPRM

引脚共用功能选择寄存器列表

- **PAPS0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PAS31	PAS30	PAS21	PAS20	PAS11	PAS10	PAS01	PAS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PAS3[1:0]**：PA3 引脚共用功能选择

00：PA3/INT0
01：AN8/BATV
10：PA3/INT0
11：PA3/INT0

Bit 5~4 **PAS2[1:0]**：PA2 引脚共用功能选择
00：PA2
01：AN9
10：VREF
11：PA2

Bit 3~2 **PAS1[1:0]**：PA1 引脚共用功能选择
00：PA1
01：OUVP0/AN10
10：PA1
11：PA1

Bit 1~0 **PAS0[1:0]**：PA0 引脚共用功能选择
00：PA0
01：OUVP1/AN11
10：PA0
11：PA0

- **PAPS1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PAS71	PAS70	PAS61	PAS60	PAS51	PAS50	PAS41	PAS40
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 **PAS7[1:0]**：PA7 引脚共用功能选择
00：PA7/INT2
01：AN6
10：PA7/INT2
11：PA7/INT2

Bit 5~4 **PAS6[1:0]**：PA6 引脚共用功能选择
00：PA6/INT1
01：AN7
10：PA6/INT1
11：PA6/INT1

Bit 3~2 **PAS5[1:0]**：PA5 引脚共用功能选择
00：PA5
01：STP
10：SDO
11：未定义

Bit 1~0 **PAS4[1:0]**：PA4 引脚共用功能选择
00：PA4
01：STP
10：SDI
11：未定义

- **PBPS 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PBS7	PBS6	PBS5	PBS4	PBS3	PBS2	PBS1	PBS0
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 **PBS7**：PB7 引脚共用功能选择
0：PB7
1：SCOM3

Bit 6 **PBS6**：PB6 引脚共用功能选择
0：PB6
1：SCOM2

Bit 5 **PBS5**：PB5 引脚共用功能选择
0：PB5
1：SCOM1

Bit 4 **PBS4**：PB4 引脚共用功能选择
0：PB4
1：SCOM0

Bit 3 **PBS3**：PB3 引脚共用功能选择
0：PB3
1：OUT1L

Bit 2 **PBS2**：PB2 引脚共用功能选择
0：PB2
1：OUT1H

Bit 1 **PBS1**：PB1 引脚共用功能选择
0：PB1
1：OUT0L

Bit 0 **PBS0**：PB0 引脚共用功能选择
0：PB0
1：OUT0H

- **PCPS0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PCS31	PCS30	PCS20	PCS20	PCS11	PCS10	PCS01	PCS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PCS3[1:0]**：PC3 引脚共用功能选择
00：PC3
01：OCP1
10：PC3
11：PC3
注：若设置 PCS3[1:0]=01 且 PCS2[1:0]=01，则这两个引脚可同时用作 OCP1 输入。

Bit 5~4 **PCS2[1:0]**：PC2 引脚共用功能选择
00：PC2
01：OCP1
10：PTP
11：PC2
注：若设置 PCS3[1:0]=01 且 PCS2[1:0]=01，则这两个引脚可同时用作 OCP1 输入。

Bit 3~2 **PCS1[1:0]**：PC1 引脚共用功能选择
00：PC1
01：OCP0
10：PTP
11：PC1
注：若设置 PCS1[1:0]=01 且 PCS0[1:0]=01，则这两个引脚可同时用作 OCP0 输入。

Bit 1~0 **PCS0[1:0]**：PC0 引脚共用功能选择
00：PC0/PTCK
01：OCP0
10：PC0/PTCK
11：PC0/PTCK
注：若设置 PCS1[1:0]=01 且 PCS0[1:0]=01，则这两个引脚可同时用作 OCP0 输入。

- **PCPS1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	PCS51	PCS50	PCS41	PCS40
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 **PCS5[1:0]**：PC5 引脚共用功能选择
00：PC5/STCK
01：SDA
10：SCS
11：PC5/STCK

Bit 1~0 **PCS4[1:0]**：PC4 引脚共用功能选择
00：PC4
01：SCL/SCK
10：PC4
11：PC4

- **PDPS0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PDS31	PDS30	PDS21	PDS20	PDS11	PDS10	PDS01	PDS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PDS3[1:0]**：PD3 引脚共用功能选择
00：PD3
01：AN3
10：D1-
11：PD3

Bit 5~4 **PDS2[1:0]**：PD2 引脚共用功能选择
00：PD2
01：AN2
10：D1+
11：PD2

Bit 3~2 **PDS1[1:0]**：PD1 引脚共用功能选择
00：PD1
01：AN1/D0-
10：SCL/SCK
11：PD1

Bit 1~0 **PDS0[1:0]**：PD0 引脚共用功能选择
00：PD0
01：AN0/D0+（USB D0+0.6V 输出引脚）
10：SDA
11：$\overline{\text{SCS}}$

- **PDPS1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	PDS71	PDS70	PDS61	PDS60	PDS51	PDS50	PDS41	PDS40
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 **PDS7[1:0]**：PD7 引脚共用功能选择
00：PD7
01：SDI
10：PD7
11：PD7

Bit 5~4 **PDS6[1:0]**：PD6 引脚共用功能选择
00：PD6
01：SDO
10：PD6
11：PD6

Bit 3~2 **PDS5[1:0]**：PD5 引脚共用功能选择
00：PD5
01：AN5
10：D2-
11：PD5

Bit 1~0 **PDS4[1:0]**：PD4 引脚共用功能选择
00：PD4
01：AN4
10：D2+
11：PD4

- **PRM 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	SDIPRM	SDOPRM	—	SCSBPRM	SCKSCLPRM	SDAPRM
R/W	—	—	R/W	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 **SDIPRM**：SDI 引脚位置选择
0：PA4
1：PD7

Bit 4 **SDOPRM**：SDO 引脚位置选择
0：PA5
1：PD6

Bit 3　　未定义，读为“0”

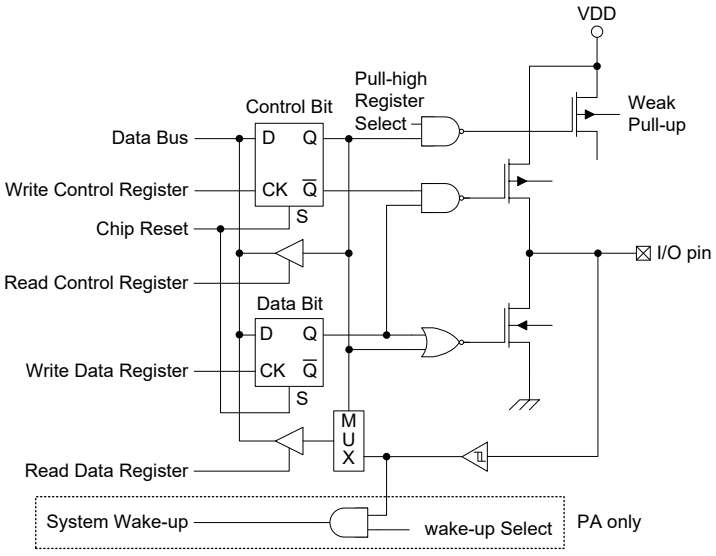
Bit 2　　**SCSBPRM**：$\overline{SCS}$ 引脚位置选择
0：PC5
1：PD0

Bit 1　　**SCKSCLPRM**：SCK/SCL 引脚位置选择
0：PC4
1：PD1

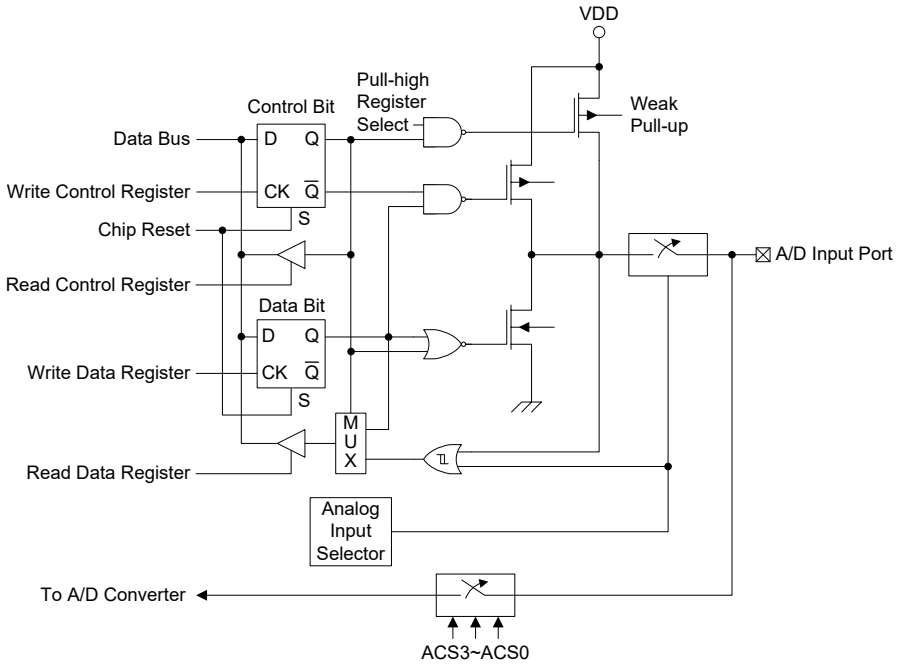
Bit 0　　**SDAPRM**：SDA 引脚位置选择
0：PC5
1：PD0

输入 / 输出引脚结构

下图为输入 / 输出引脚逻辑功能的内部结构图。具体输入 / 输出引脚的逻辑结构图可能与此图不同，这里只是为了方便对 I/O 引脚逻辑功能的理解提供一个参考。由于存在诸多的引脚共用结构，在此不方便提供所有类型引脚功能结构图。



通用输入 / 输出端口



A/D 输入 / 输出结构

编程注意事项

在编程中，最先要考虑的是端口的初始化。复位之后，所有的输入 / 输出数据及端口控制寄存器都将被设为逻辑高。所有输入 / 输出引脚默认为输入状态，而其电平则取决于其它相连接电路以及是否选择了上拉电阻。如果端口控制寄存器 PAC~PDC，某些引脚位被设定输出状态，这些输出引脚会有初始高电平输出，除非数据寄存器端口 PA~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。该系列单片机提供几个定时器模块 (简称 TM)，来实现和时间有关的功能。定时器模块是包括多种操作的定时单元，提供的操作有：定时 / 事件计数器，捕捉输入，比较匹配输出，单脉冲输出以及 PWM 输出等功能。每个定时器模块有两个独立中断。每个 TM 外加的输入输出引脚，扩大了定时器的灵活性，便于用户使用。

这里只介绍各种 TM 的共性，更多详细资料请参考相关定时器章节。

简介

该系列单片机包含 1 个 16-bit 标准型 TM 和 1 个 10-bit 周期型 TM，分别命名为 STM 和 PTM。虽然性质相似，但不同 TM 特性复杂度不同。本章介绍标准型和周期型 TM 的共性，更多详细资料分别见后面各章。两种类型 TM 的特性和区别见下表。

功能	STM	PTM
定时 / 计数器	√	√
捕捉输入	√	√
比较匹配输出	√	√
PWM 通道数	1	1
单脉冲输出	1	1
PWM 对齐方式	边沿对齐	边沿对齐
PWM 调节周期 & 占空比	占空比或周期	占空比或周期

TM 功能概要

TM 操作

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

TM 时钟源

驱动 TM 计数器的时钟源很多。通过设置 xTM 控制寄存器的 xTCK2~xTCK0 位 (其中的 x 可为 S 或 P，代表不同的 TM 类型)，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 xTCK 引脚时钟。xTCK 引脚时钟源用于允许外部信号作为 TM 时钟源或用于事件计数。

TM 中断

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

TM 外部引脚

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

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

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

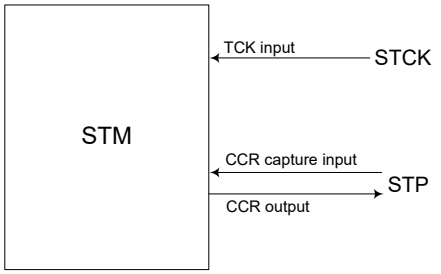
因 xTM 外部引脚 xTP 和 xTCK 与其它功能共用同一引脚，TM 功能在使用前，用户需要先通过引脚共用功能选择寄存器对相应位进行设置以选择 TM 引脚功能。特别需要注意的是，对于 xTCK 引脚，当通过引脚共用功能选择寄存器，选中 xTCK 引脚功能后，还需设置对应的输入 / 输出端口控制寄存器位，选择该引脚状态为输入类型。

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

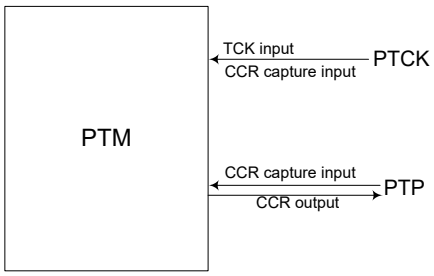
TM 外部引脚

TM 输入 / 输出引脚选择

通过设置相关引脚共用功能选择寄存器选择引脚功能为 TM 输入 / 输出功能或其它共用功能。每个 TM 输入 / 输出引脚都可通过对应的引脚共用功能选择位进行设置。正确设置选择位将相应的引脚用作 TM 输入 / 输出。更多引脚共用功能选择详见引脚共用功能章节。



STM 功能引脚控制方框图

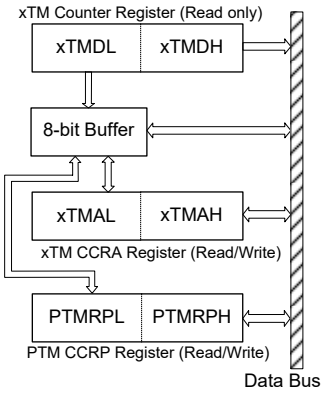


PTM 功能引脚控制方框图

编程注意事项

TM 计数寄存器和捕捉 / 比较寄存器 CCRA、CCRP 为 16-bit 或 10-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。值得注意的是 8-bit 缓存器的存取数据及相关低字节的读写操作仅在其相应的高字节读取操作执行时发生。

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

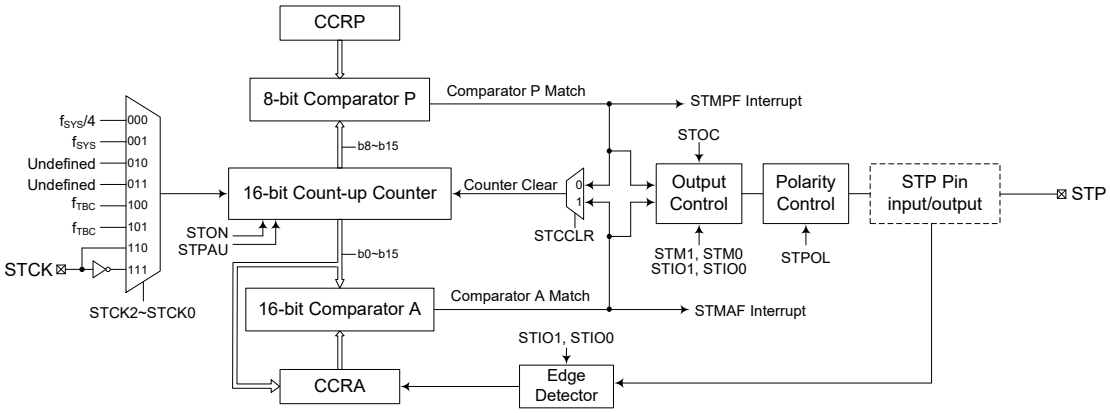


读写流程如下步骤所示：

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

标准型 TM – STM

标准型 TM 包括 5 种工作模式，即比较匹配输出、定时 / 事件计数器、捕捉输入、单脉冲输出和 PWM 输出模式。



标准型 TM 方框图

标准型 TM 操作

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

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

标准型 TM 寄存器介绍

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

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

16-bit 标准型 TM 寄存器列表

STMC0 寄存器

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

Bit 7 **STPAU**：STM 计数器暂停控制位

0：运行

1：暂停

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

Bit 6~4 **STCK2~STCK0**：选择 STM 计数时钟位

000：$f_{SYS}/4$

001：f_{SYS}

010：未定义

011：未定义

100：f_{TBC}

101：f_{TBC}

110：STCK 上升沿时钟

111：STCK 下降沿时钟

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

Bit 3 **STON**：STM 计数器 On/Off 控制位

0：Off

1：On

此位控制 STM 的总开关功能。设置此位为高则使能计数器使其运行，清零此位则除能 STM。清零此位将停止计数器并关闭 STM 减少耗电。当此位经由高到低转换时，内部计数器将复位清零；当此位经由高到低的转换时，内部计数器将保持其剩余值，直到此位再次改变为高电平。

若 STM 处于比较匹配输出模式、PWM 输出模式或单脉冲输出模式，当 STON 位经由低到高的转换时，STM 输出脚将复位至 STOC 位指定的初始值。

Bit 2~0 未定义，读为“0”

STMC1 寄存器

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

Bit 7~6 **STM1~STM0**：STM 工作模式选择位

00：比较匹配输出模式

01：捕捉输入模式

10：PWM 输出模式或单脉冲输出模式

11：定时 / 计数器模式

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

Bit 5~4 **STIO1~STIO0**：STM 外部引脚功能选择位

比较匹配输出模式

00：无变化

01：输出低

10：输出高

11：输出翻转

PWM 输出模式 / 单脉冲输出模式

00：强制无效状态

01：强制有效状态

10：PWM 输出

11：单脉冲输出

捕捉输入模式

00：在 STP 上升沿输入捕捉

01：在 STP 下降沿输入捕捉

10：在 STP 双沿输入捕捉

11：输入捕捉除能

定时 / 计数器模式

未使用

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

在比较匹配输出模式下，STIO1 和 STIO0 位决定当从比较器 A 比较匹配输出发生时 STM 输出脚 STP 如何改变状态。当从比较器 A 比较匹配输出发生时 STP 输出脚能设为切换高、切换低或翻转当前状态。若此两位同时为 0 时，这个输出将不会改变。STP 输出脚的初始值通过 STMC1 寄存器的 STOC 位设置取得。注意，由 STIO1 和 STIO0 位得到的输出电平必须与通过 STOC 位设置的初始值不同，否则当比较匹配发生时，STP 输出脚将不会发生变化。在 STP 输出脚改变状态后，通过 STON 位由低到高电平的转换复位至初始值。

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

Bit 3 **STOC**：STM 输出脚 STP 输出控制位

比较匹配输出模式

0：初始低

1：初始高

PWM 输出模式 / 单脉冲输出模式

0：低有效

1：高有效

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

Bit 2 **STPOL**：STP 输出脚输出极性控制位

0：同相

1：反相

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

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

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

STMDL 寄存器

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

Bit 7~0 STM 计数器低字节寄存器 bit7~bit0
STM 16-bit 计数器 bit7~bit0

STMDH 寄存器

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

Bit 7~0 STM 计数器高字节寄存器 bit7~bit0
STM 16-bit 计数器 bit15~bit8

STMAL 寄存器

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

Bit 7~0 STM CCRA 低字节寄存器 bit7~bit0
STM 16-bit CCRA bit7~bit0

STMAH 寄存器

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

Bit 7~0 STM CCRA 高字节寄存器 bit7~bit0
STM 16-bit CCRA bit15~bit8

STMRP 寄存器

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

Bit 7~0 STM CCRP 寄存器 bit7~bit0

STM CCRP 8-bit 寄存器，与 STM 计数器 bit15~bit8 比较。

比较器 P 匹配周期

0：65536 个 STM 时钟周期

1~255：(1~255) × 256 个 STM 时钟周期

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

标准型 TM 工作模式

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

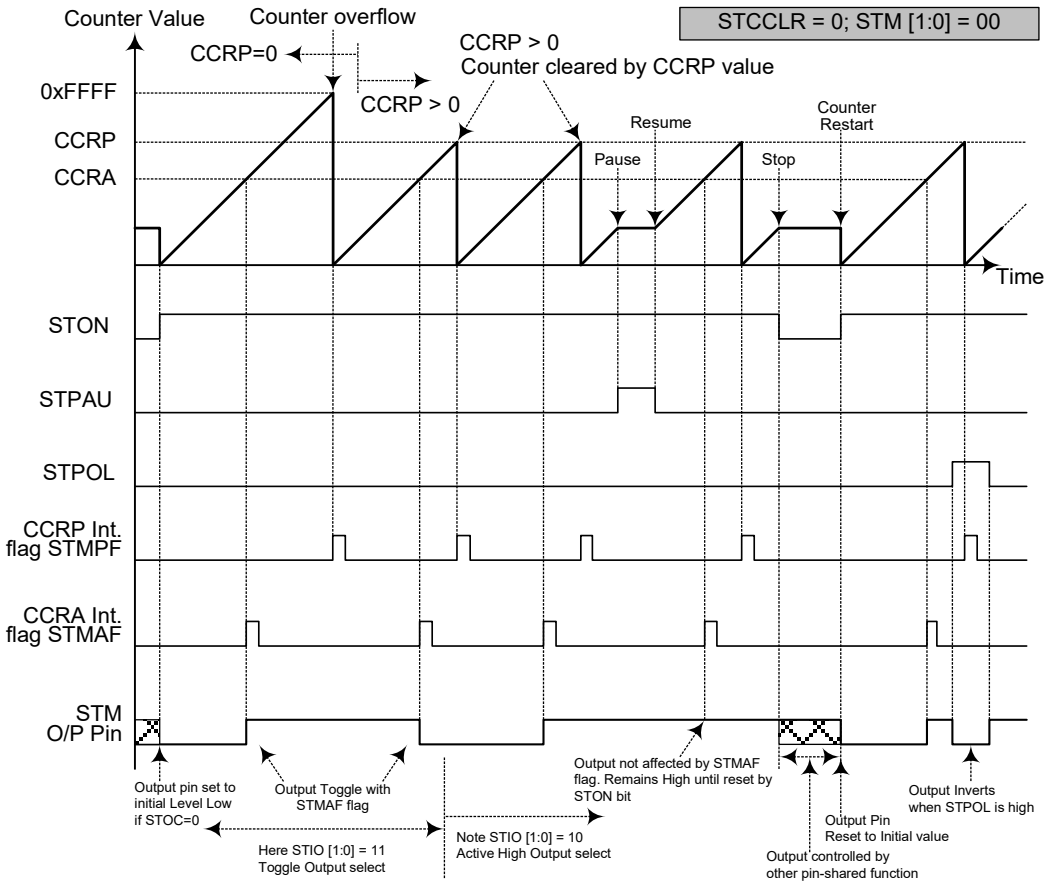
比较匹配输出模式

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

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

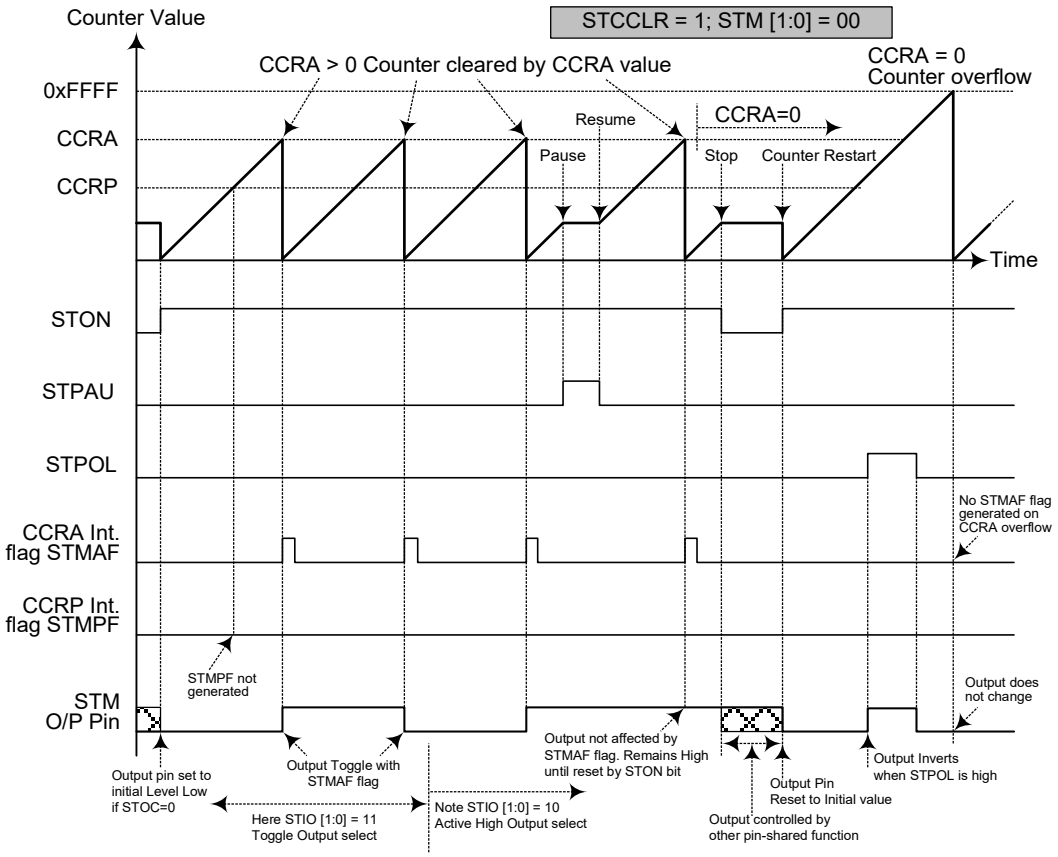
如果 CCRA 位都清除为零，当计数器的值达到 16 位最大值 FFFFH 时将溢出，但此时不会产生 STMAF 中断请求标志。

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



比较匹配输出模式 – **STCCLR=0**

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



比较匹配输出模式 – STCCLR=1

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

定时 / 计数器模式

为使 STM 工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“11”。定时 / 计数器模式与比较输出模式操作方式相同，并产生同样的中断请求标志。不同的是，在定时 / 计数器模式下 STM 输出脚未使用。因此，比较匹配输出模式中的描述和时序图可以适用于此功能。该模式中未使用 STM 外部引脚，这些引脚可通过引脚共用功能选择寄存器相应位设置用作普通 I/O 或其它功能。

PWM 输出模式

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

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

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

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

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

若 f_{SYS}=8MHz，STM 时钟源选择 f_{SYS}/4，CCRP=2，CCRA=128，

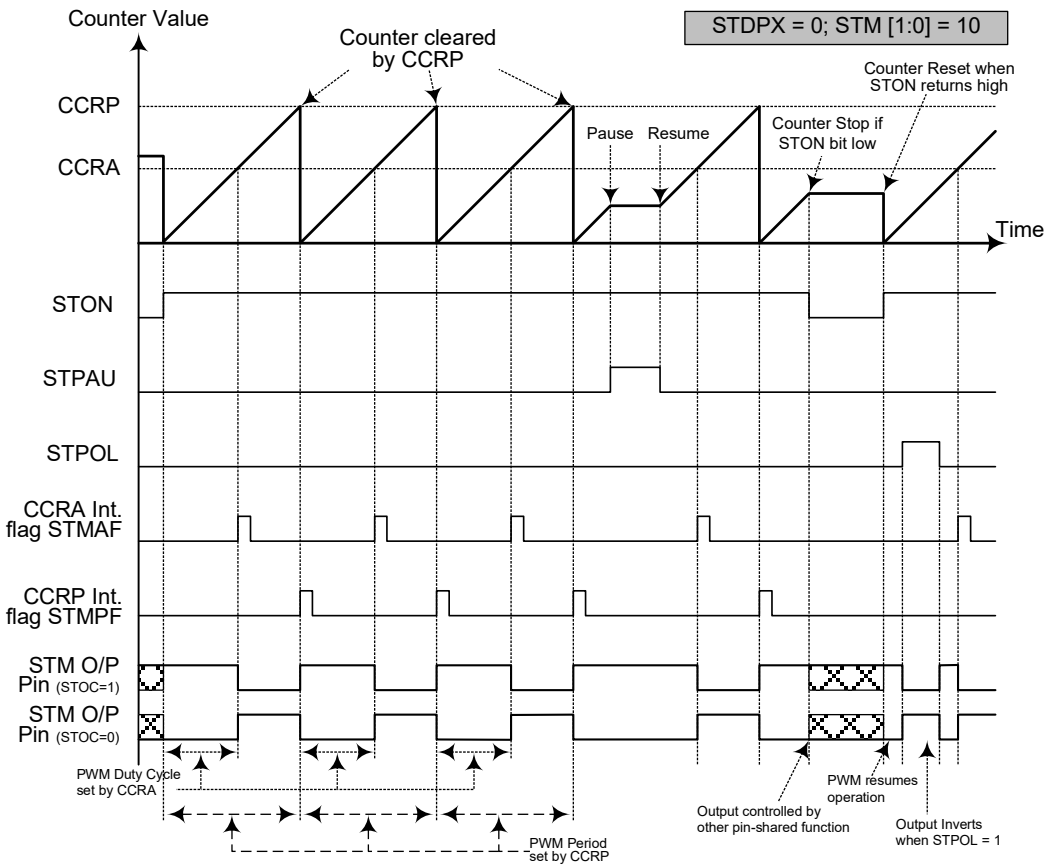
STM PWM 输出频率 = (f_{SYS}/4)/(2×256)= f_{SYS}/2048=4kHz，duty=128/(2×256)=25%

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

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

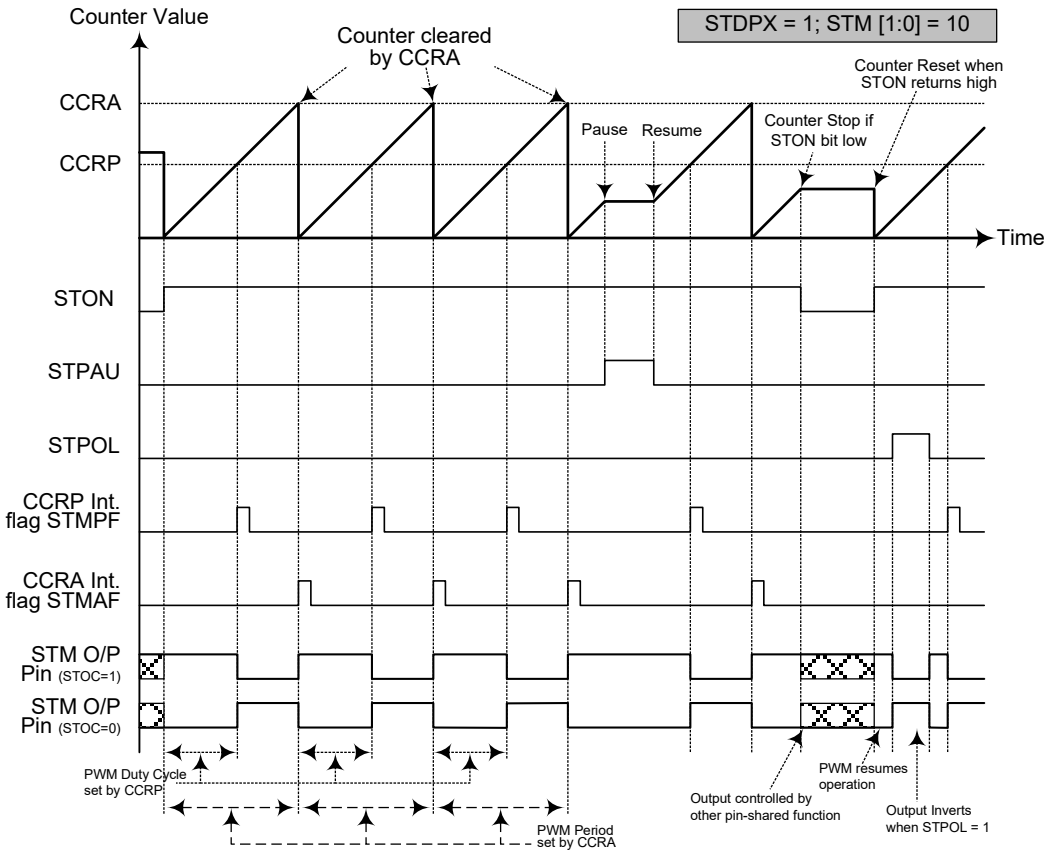
CCRP	1~255	0
Period	CCRA	
Duty	CCRP×256	65536

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



PWM 输出模式 – STDPX=0

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



PWM 输出模式 – STDPX=1

注：1. STDPX=1，计数器由 CCRA 清除

2. 计数器清零设置 PWM 周期

3. 即使当 STIO[1:0]=00 或 01，内部 PWM 功能不变

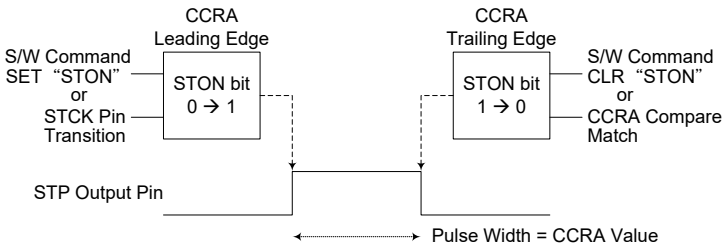
4. STCCLR 对 PWM 操作无影响

单脉冲输出模式

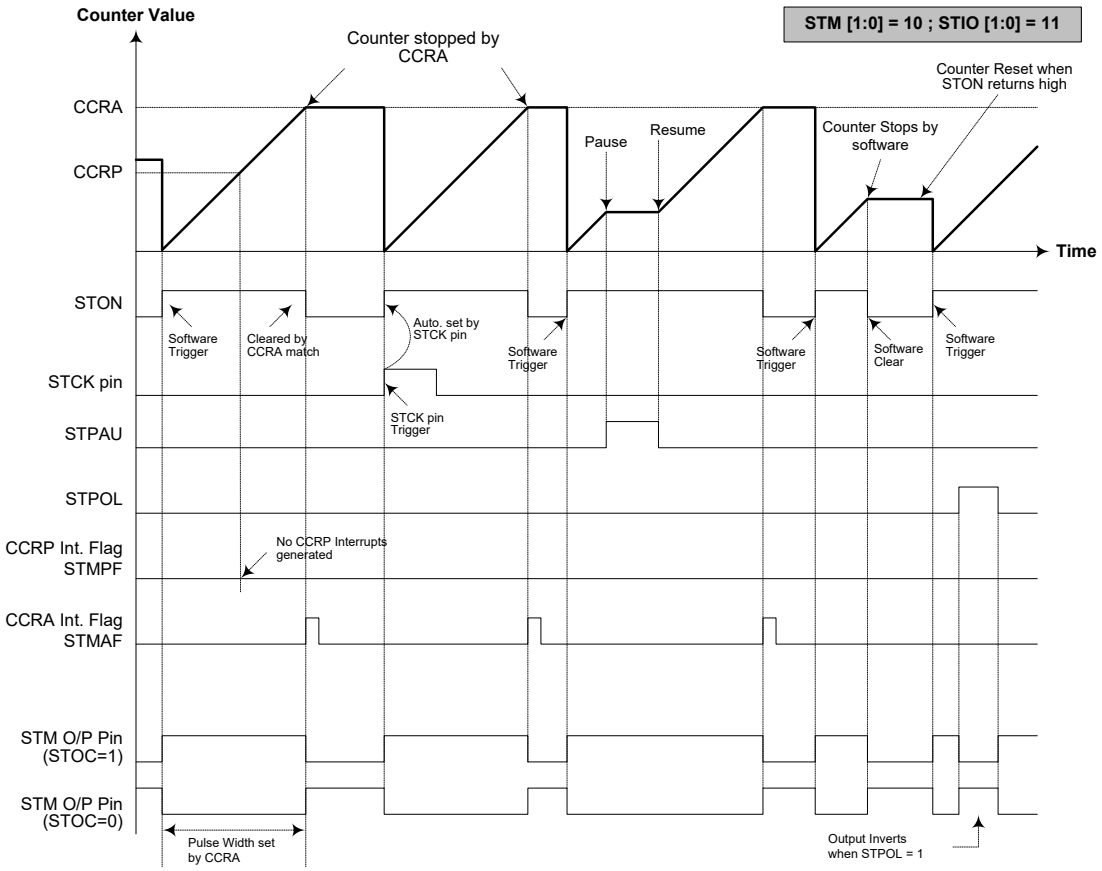
为使 STM 工作在此模式，STMC1 寄存器中的 STM1 和 STM0 位需要设置为“10”，同时 STIO1 和 STIO0 位需要设置为“11”。正如模式名所言，单脉冲输出模式，在 STM 输出脚将产生一个脉冲输出。

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

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



单脉冲产生示意图



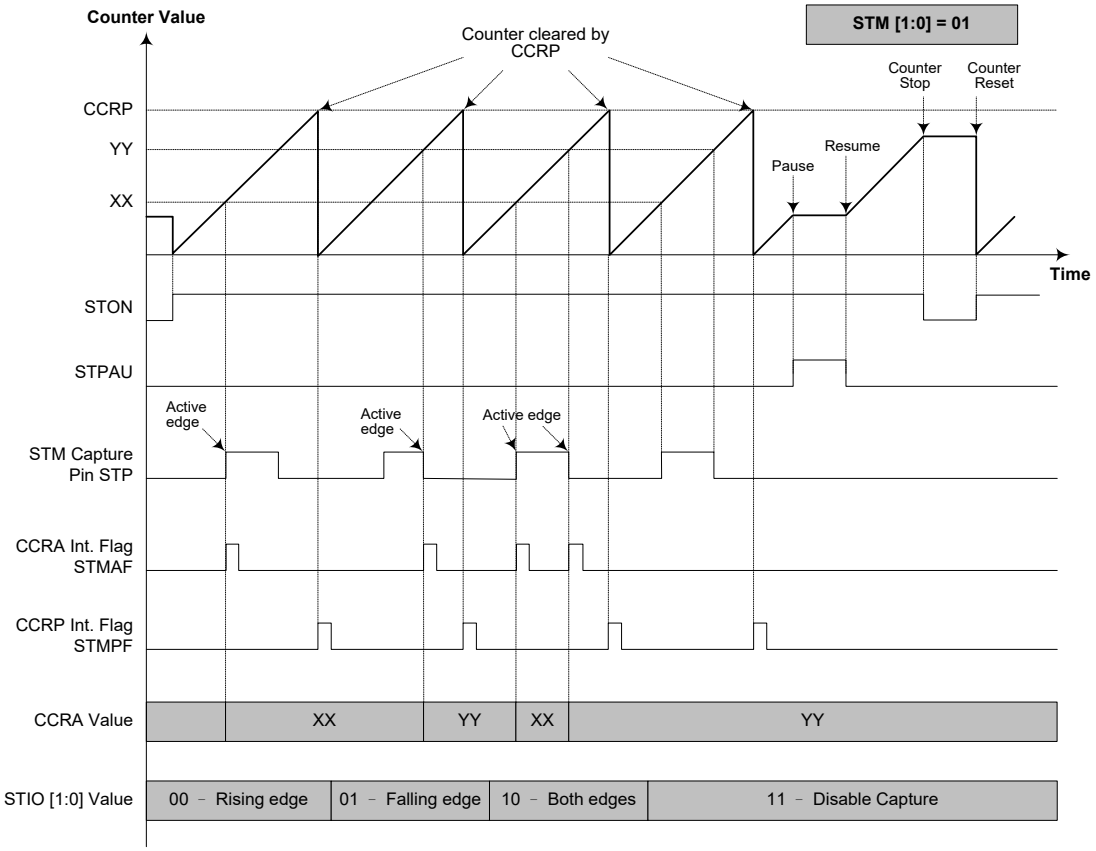
单脉冲输出模式

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

捕捉输入模式

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

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

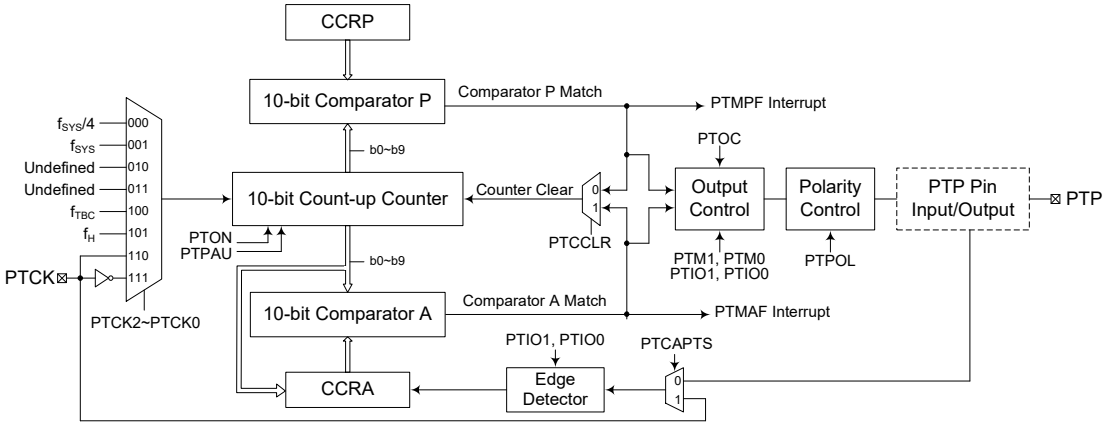


捕捉输入模式

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

周期型 TM – PTM

周期型 TM 包括 5 种工作模式，即比较匹配输出、定时 / 事件计数器、捕捉输入、单脉冲输出和 PWM 输出模式。



周期型 TM 方框图

周期型 TM 操作

周期型 TM 是 10 位宽度。周期型 TM 核心是一个由用户选择的内部或外部时钟源驱动的 10 位向上计数器，它还包括两个内部比较器即比较器 A 和比较器 P。这两个比较器将计数器的值与 CCRA 和 CCRP 寄存器中的值进行比较。CCRP 和 CCRA 是 10 位的，与计数器的所有位比较。

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

周期型 TM 寄存器介绍

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

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

10-bit 周期型 TM 寄存器列表

PTMC0 寄存器

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

Bit 7 **PTPAU**：PTM 计数器暂停控制位

0：运行

1：暂停

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

Bit 6~4 **PTCK2~PTCK0**：选择 PTM 计数时钟位

000：$f_{SYS}/4$

001：f_{SYS}

010：未定义

011：未定义

100：f_{TBC}

101：f_H

110：PTCK 上升沿

111：PTCK 下降沿

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

Bit 3 **PTON**：PTM 计数器 On/Off 控制位

0：Off

1：On

此位控制 PTM 的总开关功能。设置此位为高则使能计数器使其运行，清零此位则除能 PTM。清零此位将停止计数器并关闭 PTM 减少耗电。当此位经由低到高转变时，内部计数器将复位清零；当此位经由高到低转换时，内部计数器将保持其剩余值，直到此位再次改变为高电平。

若 PTM 处于比较匹配输出模式 , PWM 输出模式或单脉冲输出模式时，当 PTON 位经由低到高转换时，PTM 输出脚将复位至 PTOC 位指定的初始值。

Bit 2~0 未定义，读为“0”

PTMC1 寄存器

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

Bit 7~6 **PTM1~PTM0**：PTM 工作模式选择位

00：比较匹配输出模式

01：捕捉输入模式

10：PWM 输出模式或单脉冲输出模式

11：定时 / 计数器模式

这两位设置 PTM 需要的工作模式。为了确保操作可靠，PTM 应在 PTM1 和 PTM0 位有任何改变前先关掉。在定时 / 计数器模式，PTM 输出引脚功能被除能。

Bit 5~4 **PTIO1~PTIO0**：PTM 外部引脚功能选择位

比较匹配输出模式

00：无变化
01：输出低
10：输出高
11：输出翻转

PWM 输出模式 / 单脉冲输出模式

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

捕捉输入模式

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

定时 / 计数器模式

未使用

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

在比较匹配输出模式下，PTIO1 和 PTIO0 位决定当从比较器 A 比较匹配输出发生时 PTM 输出脚如何改变状态。当从比较器 A 比较匹配输出发生时 PTM 输出脚能设为切换高、切换低或翻转当前状态。若此两位同时为 0 时，这个输出将不会改变。PTM 输出脚的初始值通过 PTMC1 寄存器的 PTOC 位设置取得。注意，由 PTIO1 和 PTIO0 位得到的输出电平必须与通过 PTOC 位设置的初始值不同，否则当比较匹配发生时，PTM 输出脚将不会发生变化。在 PTM 输出脚改变状态后，通过 PTON 位由低到高电平的转换复位至初始值。

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

Bit 3 **PTOC**：PTM PTP 输出控制位

比较匹配输出模式

0：初始低
1：初始高

PWM 输出模式 / 单脉冲输出模式

0：低有效
1：高有效

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

Bit 2 **PTPOL**：PTM PTP 输出极性控制位

0：同相
1：反相

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

Bit 1 **PTCAPTS**：选择 PTM 捕捉触发源

0：来自 PTP 引脚
1：来自 PTCK 引脚

Bit 0 **PTCCLR**：选择 PTM 计数器清零条件位
0：PTM 比较器 P 匹配
1：PTM 比较器 A 匹配
此位用于选择清除计数器的方法。周期型 PTM 包括两个比较器 - 比较器 A 和比较器 P，两者都可以用作清除内部计数器。PTCCLR 位设为高，计数器在比较器 A 比较匹配发生时被清除；此位设为低，计数器在比较器 P 比较匹配发生或计数器溢出时被清除。计数器溢出清除的方法仅在 CCRP 被清除为 0 时才能生效。PTCCLR 位在 PWM 输出模式、单脉冲输出或输入捕捉模式时未使用。

PTMDL 寄存器

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

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

PTMDH 寄存器

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

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

PTMAL 寄存器

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

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

PTMAH 寄存器

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

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

PTMRPL 寄存器

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

Bit 7~0 PTM CCRP 低字节寄存器 bit 7 ~ bit 0
PTM 10-bit CCRP bit 7 ~ bit 0

PTMRPH 寄存器

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

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

周期型 TM 工作模式

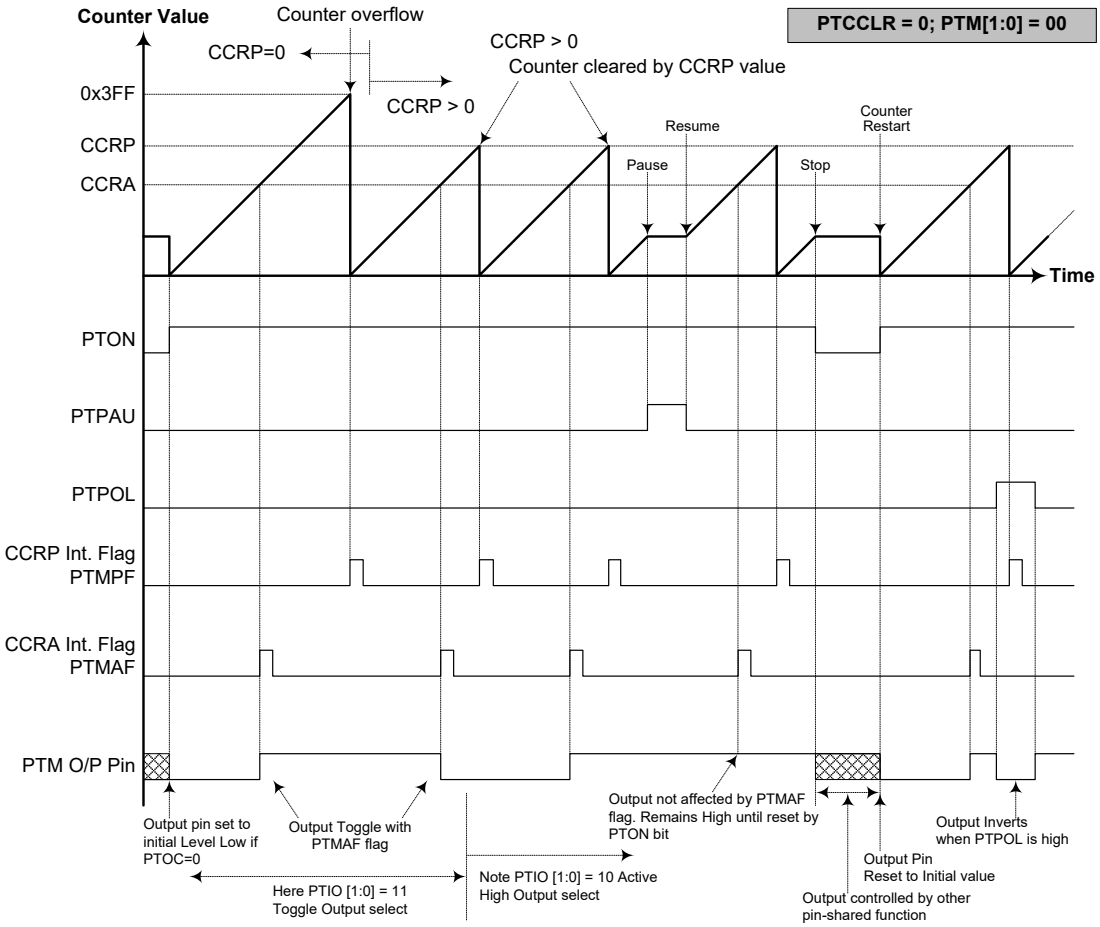
周期型 TM 有五种工作模式，即比较匹配输出模式、PWM 输出模式、单脉冲输出模式、捕捉输入模式或定时 / 计数器模式。通过设置 PTMC1 寄存器的 PTM1 和 PTM0 位选择任意模式。

比较匹配输出模式

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

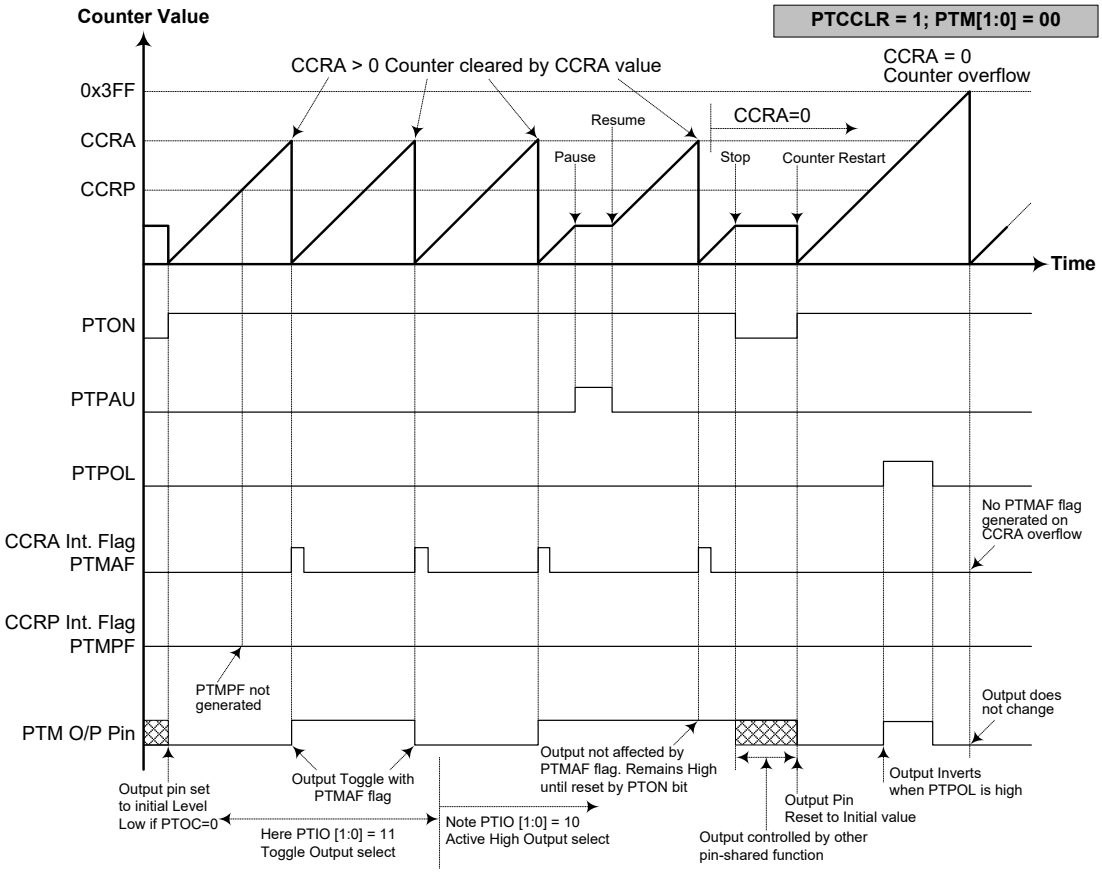
如果 PTMC1 寄存器的 PTCCLR 位设置为高，当比较器 A 比较匹配发生时计数器被清零。此时，即使 CCRP 寄存器的值小于 CCRA 寄存器的值，仅 PTMAF 中断请求标志产生。所以当 PTCCLR 为高时，不会产生 PTMPF 中断请求标志。在比较匹配输出模式中，CCRA 寄存器值不能设为“0”。

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



比较器匹配输出模式 – PTCCLR = 0

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



比较器匹配输出模式 – PTCCLR = 1

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

定时 / 计数器模式

为使 PTM 工作在此模式，PTMC1 寄存器的 PTM1 和 PTM0 位需要设置为“11”。定时 / 计数器模式与比较输出模式操作方式相同，并产生同样的中断请求标志。不同的是，在定时 / 计数器模式下 PTM 输出脚未使用。因此，比较匹配输出模式中的描述和时序图可以适用于此功能。该模式中未使用 PTM 外部引脚，这些引脚可通过引脚共用功能选择寄存器相应位设置用作普通 I/O 或其它功能。

PWM 输出模式

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

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

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

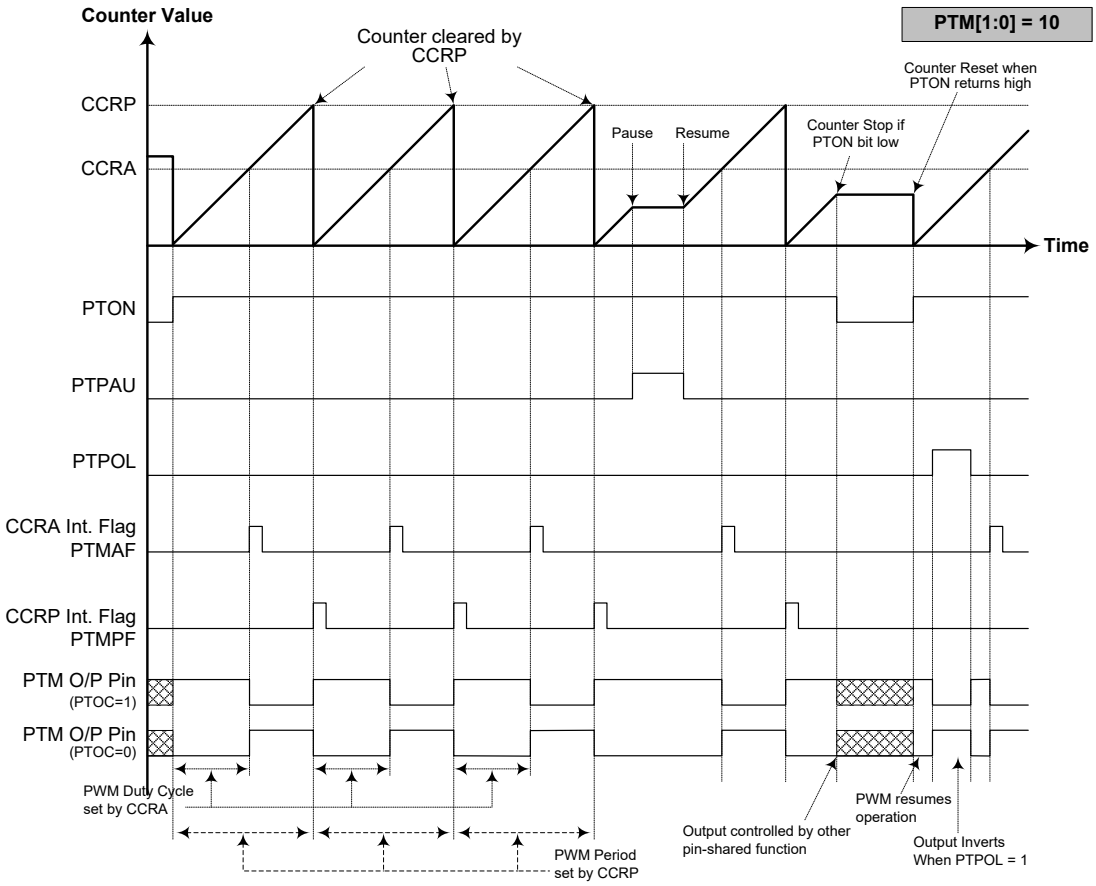
- **10-bit PTM，PWM 输出模式，边沿对齐**

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

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

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

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



PWM 输出模式

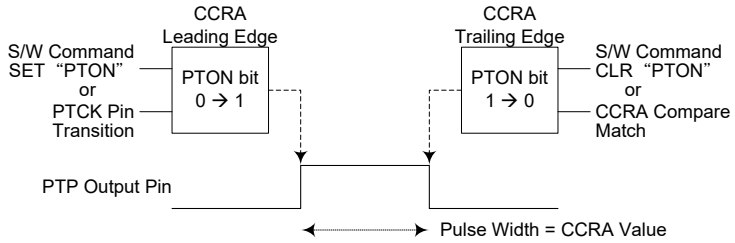
注：1. 计数器由 CCRP 清除
2. 计数器清零设置 PWM 周期
3. 当 PTIO[1:0]=00 或 01，PWM 功能不变
4. PTCCLR 位对 PWM 功能无影响

单脉冲输出模式

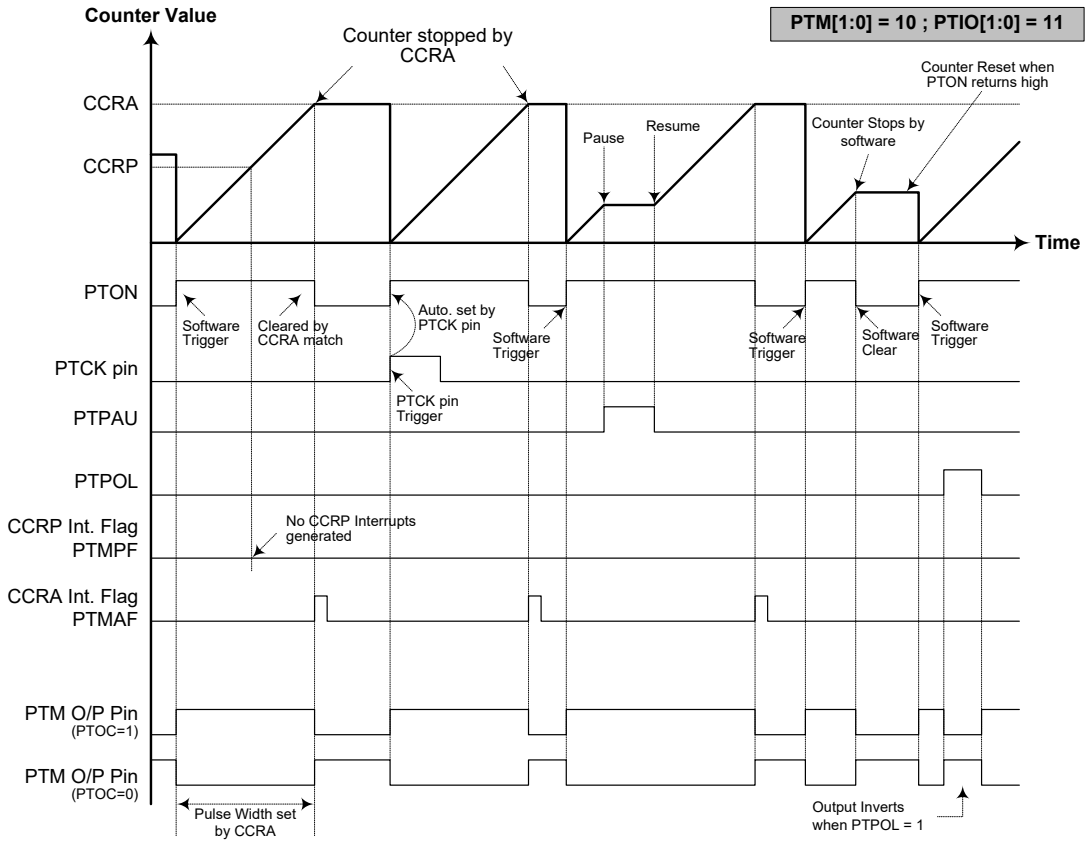
为使 PTM 工作在此模式，PTMC1 寄存器中的 PTM1 和 PTM0 位需要设置为“10”，并且相应的 PTIO1 和 PTIO0 需要设置为“11”。正如模式名所言，单脉冲输出模式，在 PTM 输出脚将产生一个脉冲输出。

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

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



单脉冲产生示意图



单脉冲模式

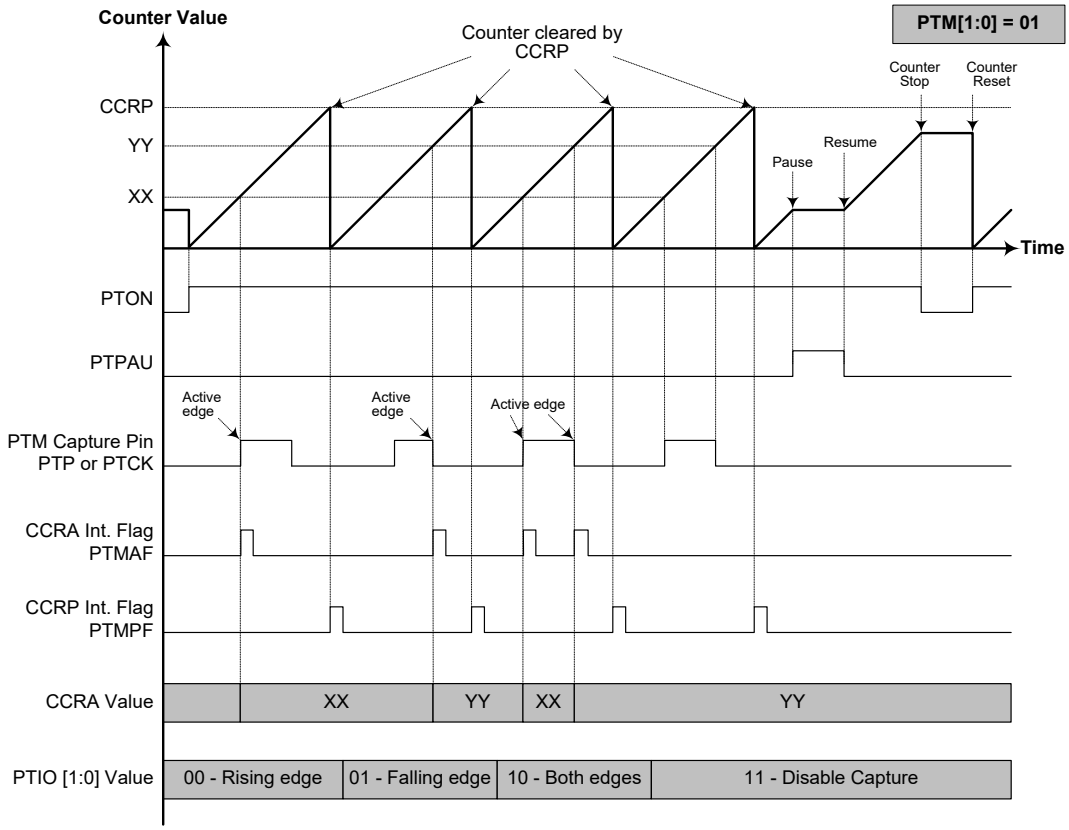
注：1. 通过 CCRA 匹配停止计数器
2. CCRP 未使用
3. 通过 PTCK 脚或设置 PTON 位为高来触发脉冲
4. PTCK 脚有效沿会自动置位 PTON
5. 单脉冲模式中，PTIO[1:0] 需置为“11”，且不能更改。

捕捉输入模式

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

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

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



捕捉输入模式

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

A/D 转换器

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

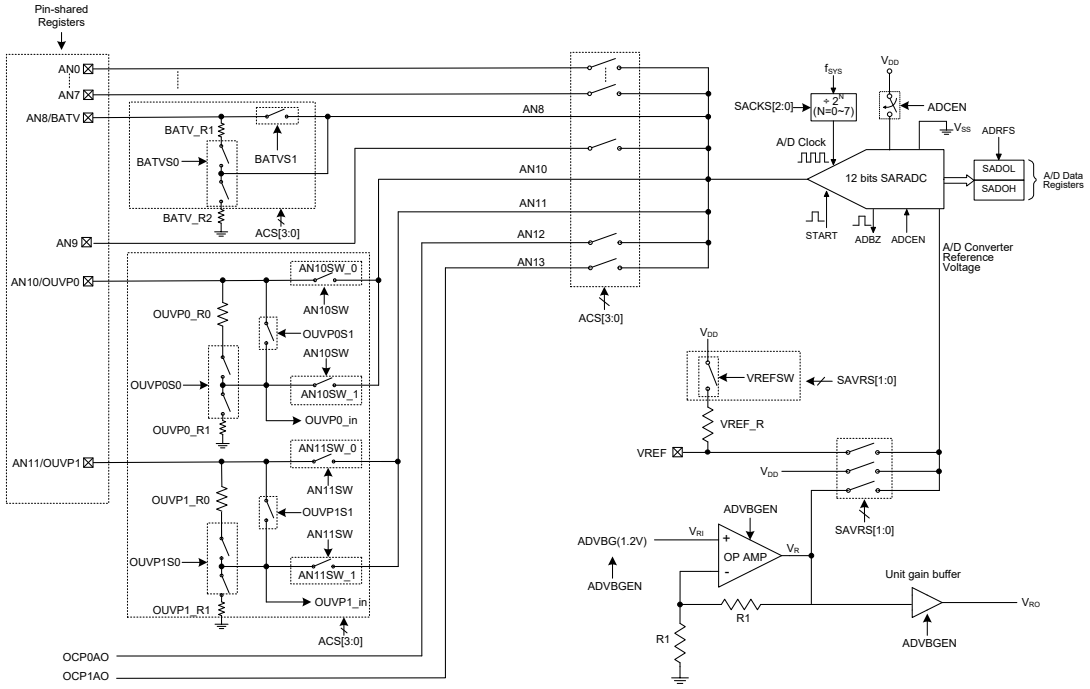
A/D 转换器简介

此系列单片机都包含一个多通道的 A/D 转换器，它们可以直接接入外部模拟信号（来自传感器或其它控制信号）或内部模拟信号（过电流保护电路 n OPA 输出信号）并直接将这些信号转换成 12 位的数字量。

选择转换外部或内部模拟信号直接由 ACS3~ACS0 位选择。若选择输入信号来自 AN8, AN10 或 AN11 通道，则除了设置 ACS3~ACS0 位选择此通道外，还应合理设置 SWS0 或 SWS1 寄存器进行相关配置，以得到想要的输入信号。关于 A/D 转换器输入信号的详细描述可参考“A/D 转换控制寄存器”和“A/D 转换器输入信号”章节进行了解。

外部信号输入通道	内部信号通道	信号通道选择位
AN0~AN11	AN12：OCP0 OPA 输出 AN13：OCP1 OPA 输出	ACS3~ACS0

下图显示了 A/D 转换器内部结构和相关的寄存器。



注：单位增益缓冲器输出的 V_{RO} 可作为内部 OCPn, OVPn 及 UVPn 功能 DAC 输入。

A/D 转换器结构

A/D 转换寄存器介绍

A/D 转换器的所有工作由一系列寄存器控制。一对只读寄存器来存放 12 位 A/D 转换数据的值。剩下几个控制寄存器设置 A/D 转换器的操作和控制功能。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SADOL (ADRFS=0)	D3	D2	D1	D0	—	—	—	—
SADOL (ADRFS=1)	D7	D6	D5	D4	D3	D2	D1	D0
SADOH (ADRFS=0)	D11	D10	D9	D8	D7	D6	D5	D4
SADOH (ADRFS=1)	—	—	—	—	D11	D10	D9	D8
SADC0	START	ADBZ	ADCEN	ADRFS	ACS3	ACS2	ACS1	ACS0
SADC1	—	—	ADVBGEN	SAVRS1	SAVRS0	SACKS2	SACKS1	SACKS0
SWS0	—	—	VREFSW	BATVS1	BATVS0	AN10SW	OUVP0S1	OUVP0S0
SWS1	—	—	—	—	—	AN11SW	OUVP1S1	OUVP1S0

A/D 转换寄存器列表

A/D 转换数据寄存器

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

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

A/D 转换数据寄存器

A/D 转换控制寄存器

两个 A/D 转换控制寄存器 SADC0 和 SADC1 及两个模拟开关控制寄存器 SWS0 和 SWS1，用来控制 A/D 转换相关的功能和操作。这些 8 位的寄存器定义包括选择连接至内部 A/D 转换器的模拟通道，数据转换格式，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。由于每个单片机只包含一个实际的模数转换电路，因此这些外部和内部模拟信号只能单个的被发送到转换器。SADC0 寄存器中的 ACS3~ACS0 位用于选择某个外部模拟输入信号还是内部模拟信号被连接到 A/D 转换器。

对于 AN8，AN10 及 AN11 输入通道，其输入侧内置分压电路或上拉电阻，可通过 SWS0 和 SWS1 寄存器的相应位进行单独设置。

引脚共用功能控制寄存器的相关位用来定义 I/O 端口中的哪些引脚为 A/D 转换器的模拟输入，哪些引脚不作为 A/D 转换输入。当引脚作为 A/D 输入时，其原来的通用 I/O 功能或其它引脚共用功能移除，此外，其内部上拉电阻也将自动断开。

- **SADC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	START	ADBZ	ADCEN	ADRFS	ACS3	ACS2	ACS1	ACS0
R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **START**：启动 A/D 转换

0 → 1 → 0：启动 A/D 转换

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。

Bit 6 **ADBZ**：A/D 转换忙碌标志位

0：A/D 转换结束或未开始转换

1：A/D 转换中

此只读位用于表明 A/D 转换过程是否完成。当 START 位由低变为高再变为低时，ADBZ 位为高，表明 A/D 转换已初始化。A/D 转换结束后，此位被清零。

Bit 5 **ADCEN**：A/D 转换器功能使能控制位

0：除能

1：使能

此位控制 A/D 内部功能。该位被置高将使能 A/D 转换器。如果该位设为低将关闭 A/D 转换器以降低功耗。当 A/D 转换除能时，A/D 数据寄存器 SADOH 和 SADOL 的内容不会改变。

Bit 4 **ADRFS**：A/D 转换数据格式选择位

0：A/D 转换数据格式 → SADOH=D[11:4]; SADOL=D[3:0]

1：A/D 转换数据格式 → SADOH=D[11:8]; SADOL=D[7:0]

此位控制存放在两个 A/D 数据寄存器中的 12 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**：A/D 转换器模输入通道选择位

0000：AN0
0001：AN1
0010：AN2
0011：AN3
0100：AN4
0101：AN5
0110：AN6
0111：AN7
1000：AN8
1001：AN9
1010：AN10
1011：AN11
1100：AN12 (来自 OCP0 OPA 输出 , OCP0AO)
1101：AN13 (来自 OCP1 OPA 输出 , OCP1AO)
1110：AN0
1111：AN0

- **SADC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	ADVBGEN	SAVRS1	SAVRS0	SACKS2	SACKS1	SACKS0
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　**ADVBGEN**：内部 1.2V Bandgap 及 OPA (增益 =2) 以及单位增益缓冲器控制位
0：除能
1：使能
ADVBGEN 位用于控制内部 1.2V Bandgap 电压及 OPA，单位增益缓冲器的使能。此内部 bandgap 电压可通过内部可编程增益放大器进行两倍放大，产生 2.4V 的输出电压 V_R。同时也可产生单位增益缓冲电压 V_{RO}，V_{RO} 电压可作为内部 OCPn, OVPn 及 UVPn 电路 DAC 输入。因此若有其它功能电路使用到 V_R 或 V_{RO} 参考电压，需提前设置此位为高，以使能内部 1.2V bandgap 电压及单位增益缓冲器电压输出。

Bit 4~3　**SAVRS1~SAVRS0**：A/D 转换器参考电压 ADC_V_{REF} 选择位
00：电源电压，V_{DD}
01：来自外部 VREF 引脚
10：内部 V_R 即 2.4V OPA 输出电压
11：电源电压，V_{DD}
此两位用于选择 A/D 转换器参考电压。注意若选择内部 2.4V V_R，则需先设置 ADVBGEN 位为高以使能内部 1.2V bandgap 电压及 OPA 功能。

Bit 2~0　**SACKS2~SACKS0**：A/D 转换器时钟源选择位
000：f_{SYS}
001：$f_{SYS}/2$
010：$f_{SYS}/4$
011：$f_{SYS}/8$
100：$f_{SYS}/16$
101：$f_{SYS}/32$
110：$f_{SYS}/64$
111：$f_{SYS}/128$

- **SWS0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	VREFSW	BATVS1	BATVS0	AN10SW	OUVP0S1	OUVP0S0
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　**VREFSW**：内建 1k 电阻开关 VREFSW 控制位
0：开关 off, 除能内建上拉电阻 VREF_R
1：开关 on, 使能内建上拉电阻 VREF_R
此位用于控制 VDD 与 VREF_R 电阻之间的模拟开关的 on/off。但只有当 SAVRS[1:0]=“01B” 即选择 A/D 转换器参考电压来自外部 VREF 引脚，设置此位为高，才能使能 VREF 引脚上的内建电阻 VREF_R，否则此开关始终为 OFF，内建电阻除能。

Bit 4 **BATVS1**：内部旁路开关 BATVS1 控制位

0：开关 off，内建分压电阻除能

1：开关 on，内建分压电阻使能

此位用于控制旁路开关 BATVS1 的 on/off。但只有当 ACS[3:0]=“1000B”即选择 AN8 为 A/D 转换信号通道，再设置此位为高，才能使 BATVS1 开关闭合为 ON，否则此开关始终为 OFF。

Bit 3 **BATVS0**：内部分压电阻开关 BATVS0 控制位

0：开关 off

1：开关 on

此位用于控制内部分压电阻开关 BATVS0 的 on/off。但只有当 ACS[3:0]=“1000B” 即选择 AN8 为 A/D 转换信号通道且设置 BATVS1=“0”，使 BATVS1 开关 OFF 时，设置此位为高，才能使 BATVS0 开关闭合为 ON, 否则此开关始终为 OFF。

Bit 2 **AN10SW**：AN10 输入选择

0：AN10SW_0 on, AN10SW_1 off

1：AN10SW_0 off, AN10SW_1 on

此位用于控制模拟开关 AN10SW_0 和 AN10SW_1 的 on/off。但只有当 ACS[3:0]=“1010B” 即选择 AN10 为 A/D 转换信号通道时，此位的设置才会有效。否则这两个开关始终为 OFF。

Bit 1 **OUVP0S1**：内部旁路开关 OUVP0S1 控制位

0：开关 off，内建分压电阻除能

1：开关 on，内建分压电阻使能

此位用于控制内部旁路开关 OUVP0S1 的 on/off。但只用当设置引脚共用功能选择寄存器位 PAS1[1:0]=“01B”即选择了 AN10 引脚功能，再设置此位为高，才能使 OUVP0S1 开关闭合为 ON，否则此开关始终为 OFF。

Bit 0 **OUVP0S0**：内部分压电阻开关 OUVP0S0 控制位

0：开关 off

1：开关 on

此位用于控制内部分压电阻开关 OUVP0S0 的 on/off。但只用当设置引脚共用功能选择寄存器位 PAS1[1:0]=“01B” 即选择了 AN10 引脚功能且设置 OUVP0S1=“0”，使 OUVP0S1 开关为 OFF 时，设置此位为高，才能使 OUVP0S0 开关闭合为 ON，否则此开关始终为 OFF。

- **SWS1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	AN11SW	OUVP1S1	OUVP1S0
R/W	—	—	—	—	—	R/W	R/W	R/W
POR	—	—	—	—	—	0	0	0

Bit 7~3 未定义，读为“0”

Bit 2 **AN11SW**：AN11 输入选择

0：AN11SW_0 on, AN11SW_1 off

1：AN11SW_0 off, AN11SW_1 on

此位用于控制模拟开关 AN11SW_0 和 AN11SW_1 的 on/off。但只有当 ACS[3:0]=“1011B” 即选择 AN11 为 A/D 转换信号通道时，此位的设置才会有效。否则这两个开关始终为 OFF。

Bit 1 **OUVP1S1**：内部旁路开关 OUVP1S1 控制位

0：开关 off，内建分压电阻除能

1：开关 on，内建分压电阻使能

此位用于控制内部旁路开关 OUVP1S1 的 on/off。但只用当设置引脚共用功能选择寄存器位 PAS0[1:0]=“01B”即选择了 AN11 引脚功能，设置此位为高，才能使 OUVP1S1 开关闭合为 ON，否则此开关始终为 OFF。

Bit 0 **OUVP1S0**：内部分压电阻开关 OUVP1S0 控制位

0：开关 off

1：开关 on

此位用于控制内部分压电阻开关 OUVP1S0 的 on/off。但只用当设置引脚共用功能选择寄存器位 PAS0[1:0]= “01B” 即选择了 AN11 引脚功能且设置 OUVP1S1= “0”，使 OUVP1S1 开关为 OFF 时，设置此位为高，才能使 OUVP1S0 开关闭合为 ON，否则此开关始终为 OFF。

A/D 转换器操作

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

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

A/D 转换器的时钟源为系统时钟 f_{SYS} 或其分频，而分频系数由 SADC1 寄存器中的 SACKS2~SACKS0 位决定。虽然 A/D 时钟源是由系统时钟 f_{SYS} 和 SACKS2~SACKS0 位决定，但可选择的最大 A/D 时钟源则有一些限制。由于允许的 A/D 时钟周期 t_{ADCK} 的范围为 0.5μs~10μs，所以选择系统时钟速度时就必须小心。如果系统时钟速度为 8MHz 时，SACKS2~SACKS0 位不能设为 “000”、“001” 或 “111”。必须保证设置的 A/D 转换时钟周期不小于时钟周期的最小值或大于时钟周期的最大值，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 转换时钟周期不在规定的范围内。

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

A/D 时钟周期范例

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

A/D 转换器参考电压

A/D 转换器参考电压可通过 SADC1 寄存器中 SAVRS1~SAVRS0 位选择来自电源电压 V_{DD}、外部参考源引脚 VREF 或内部 2.4V OPA 输出电压 V_R。若选择外部 VREF 引脚提供参考电压，需要另外先设置相关的引脚共用功能选择寄存器以选择引脚功能为 VREF。若选中 VREF 引脚为 A/D 转换器参考电压输入来源，此时可通过设置 SWS0 寄存器中的 VREFSW 位，选择 VREF 引脚是否连接内部上拉电阻。若选中内部 bandgap 电压通过 OPA 放大两倍后的输出电压 V_R 为 A/D 转换器参考电压，则需提前设置 ADVBGEN 位为高以使能内部 1.2V bandgap 和 OPA 功能。

注意，A/D 转换器输入的模拟信号值一定不能超过所选的参考电压值。

SAVRS[1:0]	参考电压源	说明
00, 11	V_{DD}	来自电源电压，V_{DD}
01	VREF 引脚	来自外部 VREF 引脚输入
10	V_R	来自内部 OPA 输出，V_R

A/D 转换器参考电压选择

A/D 转换器输入信号

所有的 A/D 模拟输入引脚都与 I/O 口及其它功能共用。使用引脚共用功能选择寄存器对应位，可以将它们设置为 A/D 转换器模拟输入脚或其它功能引脚。具体选择设置可参考“引脚共用功能选择寄存器”章节。如果对应的引脚被选择作为 A/D 转换器模拟输入功能，那么它原来的引脚功能将除能。通过这种方式，引脚的功能可由程序来控制，灵活地切换引脚功能。如果将引脚设为 A/D 输入，则通过寄存器设置的所有上拉电阻会自动断开。请注意，端口控制寄存器不需要为使能 A/D 输入而先设定为输入模式，当对应的引脚共用功能选择寄存器选择了 A/D 引脚功能时，端口控制寄存器的状态将自动为输入状态且无法修改。

A/D 转换器转换的模拟信号可以来自外部信号通道输入 AN0~AN11，也可为内部电路信号 AN12~AN13。内部信号可选择来自过电流保护电路 n 输出信号。通过设置 SADC0 寄存器的 ACS3~ACS0 位选择实际输入 A/D 转换器的内部或外部信号。

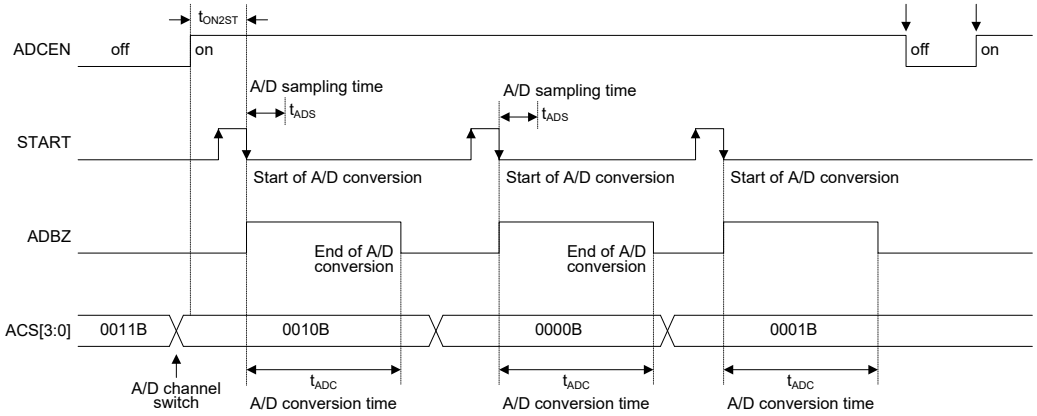
对于外部信号输入通道 AN8, AN10 及 AN11, 内置分压电路，可通过寄存器 SWS0~SWS1 的相应位对信号作进一步设置，以得到需要的输入信号。具体设置可参考相关寄存器描述详情。

A/D 转换率及时序图

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

最大 A/D 转换率 = A/D 时钟周期 ÷16

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



A/D 转换时序图

A/D 转换步骤

下面概述实现 A/D 转换过程的各个步骤。

- 步骤 1
  通过 SADC1 寄存器中的 SACKS2~SACKS0 位，选择所需的 A/D 转换时钟。
- 步骤 2
  将 SADC0 寄存器中的 ADCEN 位置高使能 A/D 转换器。
- 步骤 3
  通过 SADC0 寄存器中的 ACS3~ACS0 位，选择连接至内部 A/D 转换器的信号。
- 步骤 4
  通过 SADC1 寄存器中的 SAVRS1~SAVRS0 位选择参考电压。
- 步骤 5
  设置 SADC0 寄存器中的 ADRFS 位选择 A/D 转换器输出数据格式。
- 步骤 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 中断功能是激活的。因 A/D 转换器中断属于多功能中断，因此所在的多功能中断控制位 MFnE, 总中断控制位 EMI 以及 A/D 转换器中断位 ADE 都需要提前置位为“1”。
- 步骤 7
  现在可以通过设置 START 位从“0”到“1”再回到“0”，开始模数转换的过程。
- 步骤 8
  可以轮询 SADC0 寄存器中的 ADBZ 位，检查模数转换过程是否完成。当此位为逻辑高时，表示转换正在进行中。当此位为逻辑低时，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 SADOL 和 SADOH 获得转换后的值。若使能 A/D 中断且堆栈未满，可以在 A/D 中断子程序中读取 A/D 数据寄存器 SADOL 和 SADOH 以获得转换后的值。

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

编程注意事项

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

A/D 转换功能

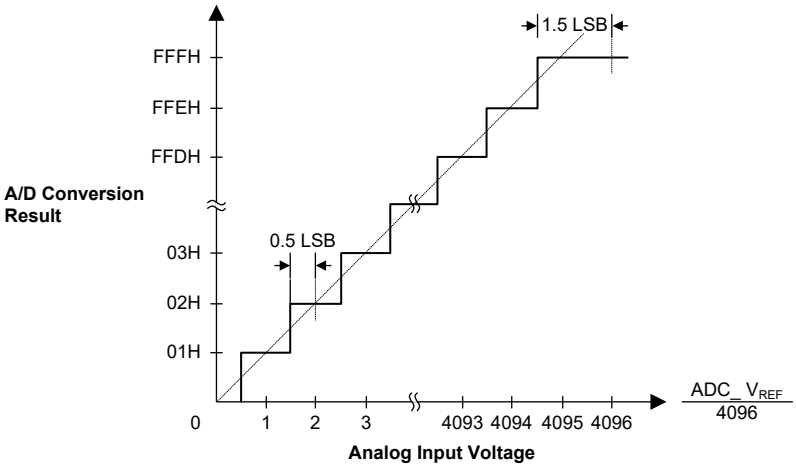
单片机含有一组 12 位的 A/D 转换器，它们转换的最大值可达 FFFH。由于模拟输入最大值等于实际输入的 A/D 转换器参考电压 ADC_V_{REF} 的电压值，因此每一位可表示 ADC_V_{REF}/4096 的模拟输入值。

$$1\ LSB = (ADC\_V_{REF}) \div 4096$$

通过下面的等式可估算 A/D 转换器输入电压值：

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

下图显示 A/D 转换器模拟输入值和数字输出值之间理想的转换功能。除了数字化数值 0，其后的数字化数值会在精确点之前的 0.5 LSB 处改变，而数字化数值的最大值将在 ADC_V_{REF} 之前的 1.5 LSB。注意此处的 ADC_V_{REF} 是指由 SAVRS1~SAVRS0 选择的实际输入 A/D 转换器参考电压。



理想的 A/D 转换功能

A/D 转换应用范例

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

范例：使用查询 ADBZ 的方式来检测转换结束

```
clr  ADE                    ; disable ADC interrupt
mov  a,03H
mov  SADC1,a                ; select fSYS/8 as A/D clock
set  ADCEN
mov  a,01h                  ; setup PDPS0 to configure pin AN0
mov  PDPS0,a
mov  a,20h
mov  SADC0,a                ; enable and connect AN0 channel to A/D converter
:
start_conversion:
clr  START                  ; high pulse on start bit to initiate conversion
set  START                  ; reset A/D
clr  START                  ; start A/D
polling_EOC:
sz   ADBZ                   ; poll the SADC0 register ADBZ bit to detect end
                            ; of A/D conversion
jmp  polling_EOC            ; continue polling
mov  a,SADOL                ; read low byte conversion result value
mov  SADOL_buffer,a         ; save result to user defined register
mov  a,SADOH                ; read high byte conversion result value
mov  SADOH_buffer,a         ; save result to user defined register
:
:
jmp  start_conversion       ; start next A/D conversion
```

范例：使用中断的方式来检测转换结束

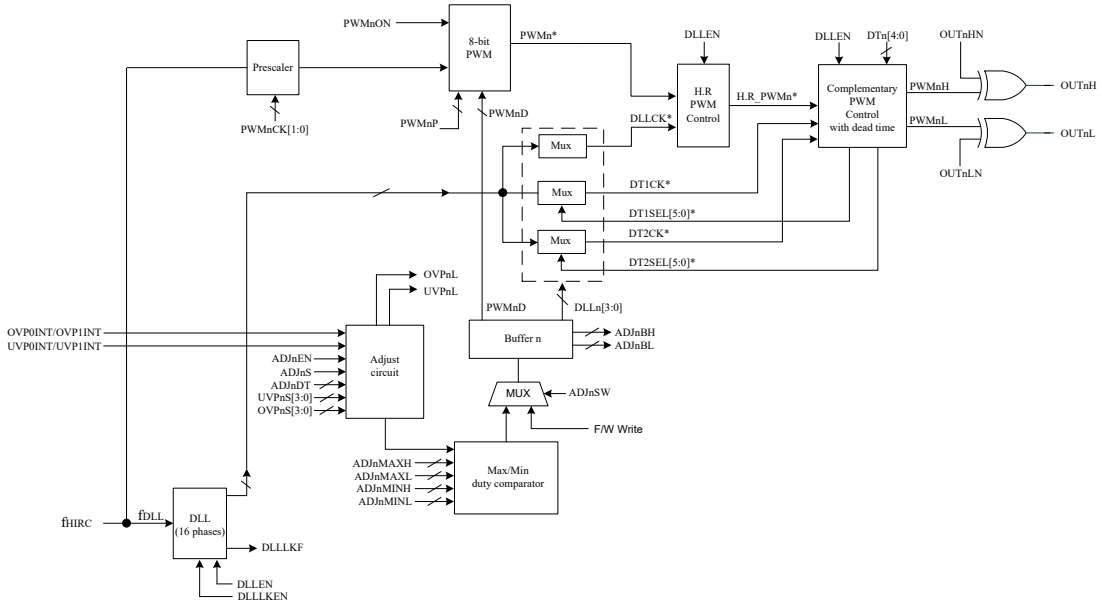
```
clr  ADE                    ; disable ADC interrupt
mov  a,03H
mov  SADC1,a                ; select fSYS/8 as A/D clock
set  ADCEN
mov  a,01h                  ; setup PDPS0 to configure pin AN0
mov  PDPS0,a
mov  a,20h
mov  SADC0,a                ; enable and connect AN0 channel to A/D converter
:
start_conversion:
clr  START                  ; high pulse on START bit to initiate conversion
set  START                  ; reset A/D
clr  START                  ; start A/D
clr  ADF                    ; clear ADC interrupt request flag
clr  MF1F                   ; clear Multi-function interrupt 1 request flag
set  ADE                    ; enable ADC interrupt
set  MF1E                   ; enable Multi-function 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,SADOL                ; read low byte conversion result value
mov  SADOL_buffer,a         ; save result to user defined register
mov  a,SADOH                ; read high byte conversion result value
mov  SADOH_buffer,a         ; save result to user defined register
:
:
EXIT_INT_ISR:
mov  a,status_stack
mov  STATUS,a               ; restore STATUS from user defined memory
mov  a,acc_stack            ; restore ACC from user defined memory
reti
```

可自动调节的高精度 PWM 发生器

该系列单片机包含一个功能高度集成的高精度 PWM 信号发生器，增强了应用的灵活性。

功能描述

该系列单片机内置的高精度 8-bit PWM 发生器电路由一个 PWM 产生电路，延迟锁相回路及一个带死区时间发生器控制的 PWM 互补输出电路组成。PWM 占空比可通过设置后进行自动调节。



Note：1. “*”为内部信号名称而非特殊功能寄存器位名称

DT1SEL[5:0] 和 DT2SEL[5:0] 基于 DTn[4:0] 自动进行计算与选择。

DT1CK 为 PWMnH 死区时间参考信号

DT2CK 为 PWMnL 死区时间参考信号

DLLCK 为 H.R_PWM 参考信号

2. H.R 为高精度（High Resolution）的缩写符号

H.R PWM 输出方框图（n=0 或 1）

H.R PWM 寄存器介绍

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

PWM 自动调节功能的控制也可通过一些寄存器进行设置。ADJnC 寄存器用于控制 PWM 自动调节功能的使能 / 除能，PWMn 占空比调整操作，存储 OUVPn（过压 / 欠压保护）比较器的输出状态。ADJnS 寄存器用于选择当过压或欠压状况发生时的调整步数。ADJnDT 寄存器用于设置自动调节功能触发延迟时间。两组寄存器对，ADJnMAXH & ADJnMAXL 和 ADJnMINH & ADJnMINL 分别用于设置可允许的占空比数据的最大和最小值。一个寄存器对 ADJnBH& ADJnBL 用于存储自动调节时 PWM 缓存的占空比数据。剩下的 OUTPC0 寄存器用于 PWMn 输出控制。

寄存器名称	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PWMnP	D7	D6	D5	D4	D3	D2	D1	D0
PWMnD	D7	D6	D5	D4	D3	D2	D1	D0
PWMnC	PWMnCK1	PWMnCK0	PWMnON	DTn4	DTn3	DTn2	DTn1	DTn0
DLLn	DLLn3	DLLn2	DLLn1	DLLn0	—	—	—	—
DLLC	DLLLKEN	DLLEN	—	—	—	—	—	DLLLKF
ADJnC	ADJnEN	ADJnV	ADJnSW	—	OVPnL	UVPnL	—	—
ADJnS	OVPnS3	OVPnS2	OVPnS1	OVPnS0	UVPnS3	UVPnS2	UVPnS1	UVPnS0
ADJnDT	—	—	D5	D4	D3	D2	D1	D0
ADJnMAXH	—	—	—	—	D11	D10	D9	D8
ADJnMAXL	D7	D6	D5	D4	D3	D2	D1	D0
ADJnMINH	—	—	—	—	D11	D10	D9	D8
ADJnMINL	D7	D6	D5	D4	D3	D2	D1	D0
ADJnBH	—	—	—	—	D11	D10	D9	D8
ADJnBL	D7	D6	D5	D4	D3	D2	D1	D0
OUTPC0	OUT1HS	OUT1LS	OUT0HS	OUT0LS	OUT1HN	OUT1LN	OUT0HN	OUT0LN

H.R PWM 发生器 & 自动调节控制寄存器列表（n= 0 或 1）

PWMnP 寄存器

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

Bit 7~0 **D7~D0**：8-bit PWMn 周期寄存器
PWMn 周期 = PWMnP[7:0] +1

PWMnD 寄存器

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

Bit 7~0 **D7~D0**：8-bit PWMn Duty 寄存器
PWMnP 和 PWMnD 两个寄存器用于 8-bit PWMn 周期与占空比控制。在设置过程中应注意下面几点：

1. 设置的 PWMnD 值应满足条件：1 ≤ PWMnD ≤ (PWMnP - 1)
2. PWMnD (最 小 值) = 1+ DLLn[3:0] - DTn[4:0]（ 其 中 DLLn[3:0] = 0000B, DTn[4:0] = 11111B）
3. PWMnD (最大值) = PWMnP-1 + DLLn[3:0] - DTn[4:0]（其中 DLLn [3:0] = 1111B, DTn[4:0] = 00000B）

PWMnC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	PWMnCK1	PWMnCK0	PWMnON	DTn4	DTn3	DTn2	DTn1	DTn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PWMnCK[1:0]**：PWM 计数时钟源选择

00：f_{HIRC}

01：$f_{HIRC}/2$

10：$f_{HIRC}/4$

11：$f_{HIRC}/8$

Bit 5 **PWMnON**：PWMn 功能使能控制

0：除能，PWM 计数值 = 0

1：使能

当设置 PWMnON 位为零，PWMn 功能关闭。OUTnH 和 OUTnL 的输出状态由 OUTPC0 寄存器相应的 OUTnHS 和 OUTnLS 位决定。

Bit 4~0 **DTn[4:0]**：PWMn 死区时间选择

00000：死区时间 = $t_{DLL}\times0 \sim t_{DLL}\times1$

00001：死区时间 = $t_{DLL}\times2 \sim t_{DLL}\times3$

00010：死区时间 = $t_{DLL}\times4 \sim t_{DLL}\times5$

00011：死区时间 = $t_{DLL}\times6 \sim t_{DLL}\times7$

00100：死区时间 = $t_{DLL}\times8 \sim t_{DLL}\times9$

...

...

11101：死区时间 = $t_{DLL}\times58 \sim t_{DLL}\times59$

11110：死区时间 = $t_{DLL}\times60 \sim t_{DLL}\times61$

11111：死区时间 = $t_{DLL}\times62 \sim t_{DLL}\times63$

注：$t_{DLL} = 1/(f_{HIRC}\times16)$

DLLn 寄存器

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

Bit 7~4 **DLLn[3:0]**：DLL 相位选择

0000：将 H.R_PWMn Duty 下降沿微调至 DLL phase#0 的输出上升沿

0001：将 H.R_PWMn Duty 下降沿微调至 DLL phase#1 的输出上升沿

0010：将 H.R_PWMn Duty 下降沿微调至 DLL phase#2 的输出上升沿

…

1110：将 H.R_PWMn Duty 下降沿微调至 DLL phase#14 的输出上升沿

1111：将 H.R_PWMn Duty 下降沿微调至 DLL phase#15 的输出上升沿

Bit 3~0 未定义，读为“0”

DLLC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	DLLLKEN	DLLEN	—	—	—	—	—	DLLLKF
R/W	R/W	R/W	—	—	—	—	—	R/W
POR	0	0	—	—	—	—	—	0

Bit 7 **DLLLKEN**：DLL 电路失锁保护功能使能控制

0：除能

1：使能

注：当 DLL 功能使能且 DLLLKEN=1，若有失锁现象发生，则 DLLLKF 位被置高同时 DLL 会自动解除失锁状况。当 DLL 功能使能但 DLLLKEN=0，即使失锁现象发生，DLLLKF 位也始终为零，且 DLL 不会自动解除失锁状况。

Bit 6 **DLLEN**：DLL 及死区时间功能控制

0：DLL 除能且不会插入死区时间

1：DLL 使能且插入死区时间（时间由 DTn[4:0] 决定）

若此位为零则 PWMnCK[1:0] 位可通过软件设置为“00~11”中任一值，且输出的 H.R_PWMn=PWMn，无死区时间插入。若此位为 1，则 PWMnCK[1:0] 位将被硬件强制设为“00”， PWMn 经由 DLL 微调输出带死区时间插入的高精度 PWMn。

Bit 5~1 未定义，读为“0”

Bit 0 **DLLLKF**：DLL 电路失锁标志位

0：没有发生失锁现象

1：发生失锁现象

此位可通过软件清零，但不能通过软件置 1。

注：当 DLL 功能使能且 DLLLKEN=1，若无失锁现象发生，则 DLLLKF 位为零，若有失锁现象发生，则 DLLLKF 位被置 1。但若 DLLLKEN=0 则无论有无失锁现象发生 DLLLKF 位始终为零。

ADJnDT 寄存器

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

Bit 7~6 未定义，读为“0”

Bit 5~0 **D5~D0**：自动调节 PWMn 延迟时间选择

000000：延迟时间 = PWMn 周期 × 2

000001：延迟时间 = PWMn 周期 × 4

000010：延迟时间 = PWMn 周期 × 6

…

111111：延迟时间 = PWMn 周期 × 128

即：延迟时间 = (ADJnDT[5:0]+1) × PWMn 周期 ×2

ADJnS 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OVPnS3	OVPnS2	OVPnS1	OVPnS0	UVPnS3	UVPnS2	UVPnS1	UVPnS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~4 **OVPnS[3:0]**：OVPn 自动调节 PWMn duty 步数选择
0000：0 step
0001：1 step
…
1111：15 step

Bit 3~0 **UVPnS[3:0]**：UVPn 自动调节 PWMn duty 步数选择
0000：0 step
0001：1 step
…
1111：15 step

ADJnC 寄存器

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

“x”为未知

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

Bit 6 **ADJnV**：PWMn Duty 自动调节动作控制
0：OVPnL 增加 duty, UVPnL 减少 duty
1：OVPnL 减少 duty, UVPnL 增加 duty

Bit 5 **ADJnSW**：PWMn Duty 自动调节系统控制
0：除能，F/W 通过 PWMnD+DLLn 寄存器写入 Buffer
1：使能，自动调节系统写入 Buffer
当 ADJnEN=0 时，自动调节功能关闭，则 ADJnSW 位始终为 0；当 ADJnEN=1 时，自动调节功能开启，当 OVPnL 或 UVPnL 为 1，ADJnSW 位自动切换为 1 触发自动调节系统以调整 duty。当自动调节完成，若要由 F/W 接手管理 PWMn duty，则需将 ADJnSW 位由 1 切换为 0。需注意的是，只有当 OVPnL 和 UVPnL 都为 0 时，才会成功完成 ADJnSW 位由 1 到 0 的切换。

Bit 4 未定义，读为“0”

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

Bit 2 **UVPnL**：UVPn 比较器输出状态位
0：输出低 (无欠压状况发生)
1：输出高 (欠压状况发生)

Bit 1~0 未定义，读为“0”

ADJnMAXH & ADJnMAXL 寄存器

Register	ADJnMAXH								ADJnMAXL							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	—	—	—	—	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

"—"：未定义，读为“0”

D11~D8：自动调节 PWMn 最大 duty 高字节

D7~D0：自动调节 PWMn 最大 duty 低字节
D11~D4 对应于寄存器 PWMnD [7:0] 位 , D3~D0 对应于寄存器 DLLn[3:0] 位

ADJnMINH & ADJnMINL 寄存器

Register	ADJnMINH								ADJnMINL							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	—	—	—	—	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	—	—	—	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

"—"：未定义，读为“0”

D11~D8：自动调节 PWMn 最小 duty 高字节

D7~D0：自动调节 PWMn 最小 duty 低字节
D11~D4 对应于寄存器 PWMnD [7:0] 位 , D3~D0 对应于寄存器 DLLn[3:0] 位

ADJnBH &ADJnBL 寄存器

Register	ADJnBH								ADJnBL							
Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Name	—	—	—	—	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	—	—	—	—	R	R	R	R	R	R	R	R	R	R	R	R
POR	—	—	—	—	0	0	0	0	0	0	0	0	0	0	0	0

"—"：未定义，读为“0”

D11~D8：自动调节 PWMn 缓存 duty 高字节

D7~D0：自动调节 PWMn 缓存 duty 低字节
D11~D4 对应于寄存器 PWMnD [7:0] 位 , D3~D0 对应于寄存器 DLLn[3:0] 位

OUTPC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OUT1HS	OUT1LS	OUT0HS	OUT0LS	OUT1HN	OUT1LN	OUT0HN	OUT0LN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **OUT1HS**：OCP/OUVP 发生时或 PWM 停止时 OUT1H 输出状态控制
0：输出 0
1：输出 1

Bit 6 **OUT1LS**：OCP/OUVP 发生时或 PWM 停止时 OUT1L 输出状态控制
0：输出 0
1：输出 1

Bit 5 **OUT0HS**：OCP/OUVP 发生时或 PWM 停止时 OUT0H 输出状态控制
0：输出 0
1：输出 1

Bit 4 **OUT0LS**：OCP/OUVP 发生时或 PWM 停止时 OUT0L 输出状态控制
0：输出 0
1：输出 1

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

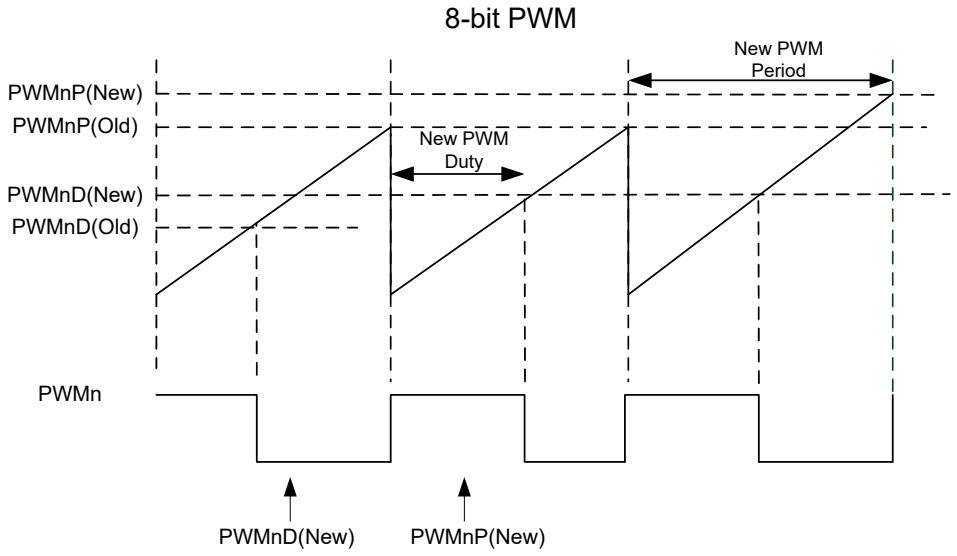
Bit 2 **OUT1LN**：OUT1L 信号反相控制
0：无反相
1：反相

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

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

PWM 发生器

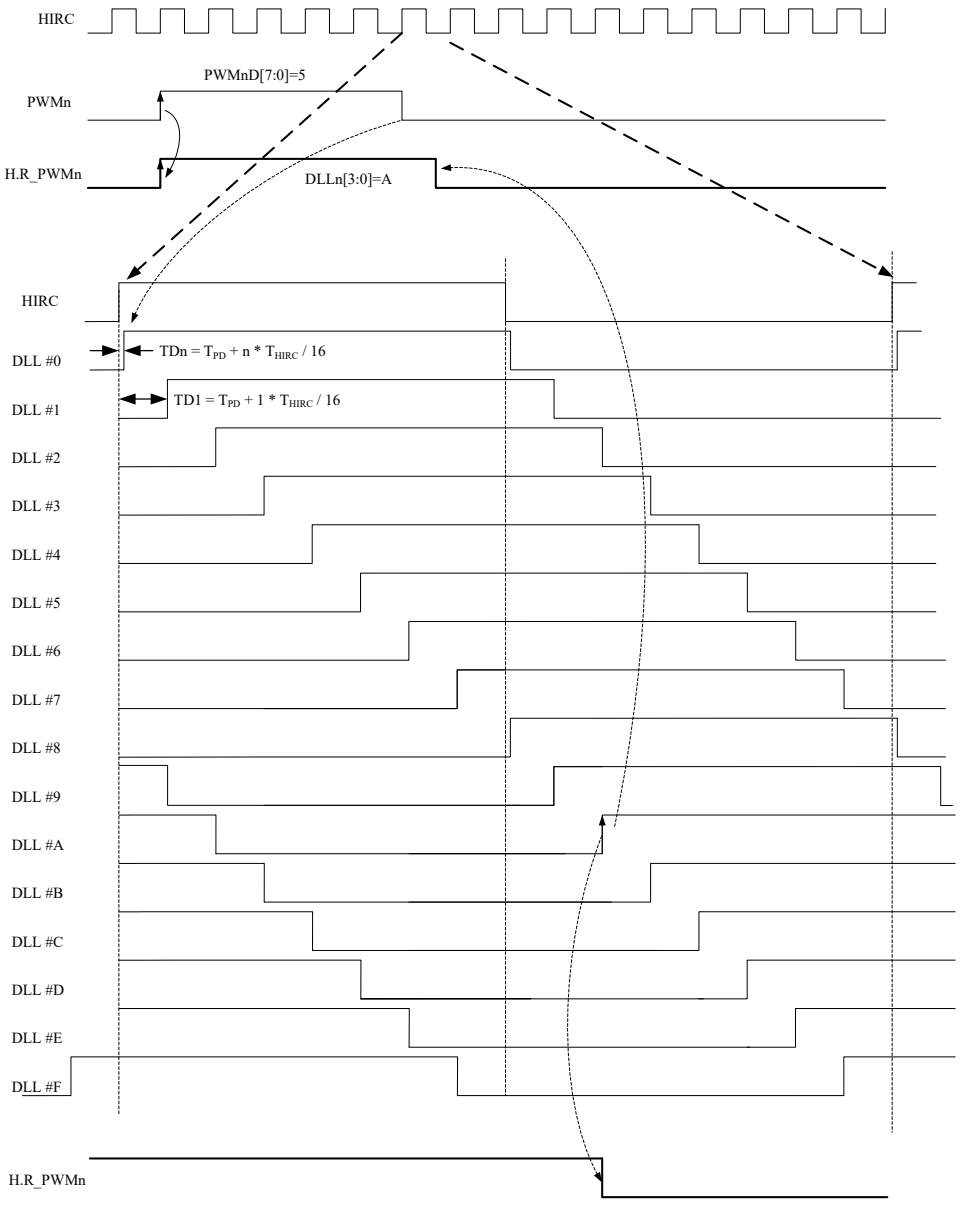
PWM 信号发生器由 HIRC 时钟驱动并产生一个 PWM 信号，通过配置 8-bit PWMnP 和 PWMnD 寄存器可调整生成的 PWMn 信号的占空比和周期。PWMn 信号周期与 PWMnC 寄存器的 PWMnCK[1:0] 位选择的 PWMn 计数器时钟频率相关，其值由 PWMnP 寄存器设置。PWMn 信号 duty 值由 PWMnD 寄存器设置。



S/W 写入值到 DLLn，DTn[4:0], PWMnD 和 PWMnP 寄存器后，F/W 会等待到 PWMn 计数器为零时，才会更新。

延迟锁相环

DLL 为延迟锁相环 (Delay Lock Loop) 的简写。DLL 可在一个 HIRC 时钟周期内产生 16 个相位输出（如下图所示）。此 16 个相位输出用于控制 PWMn 输出微调。若选择 PWMn 时钟 =f_{HIRC}，这表示 PWMn 输出占空比精度为 1/f_{HIRC}。PWMn 通过设置 DLLn 相位选择与 H.R_PWMn 控制 (由 DLL 寄存器中的 DLL[3:0] 控制) 电路后，对 PWMn 输出进行微调，当每次 PWMn 结束时，依 DLLn[3:0] 的设置延长如下图 PWMn 的输出。因此，藉由 DLL 可将 PWMn duty 精度提高 4 位。



DLL 失锁保护说明

该系列单片机提供失锁保护电路，通过设置 DLLLKEN 位为高可使能此保护电路。

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

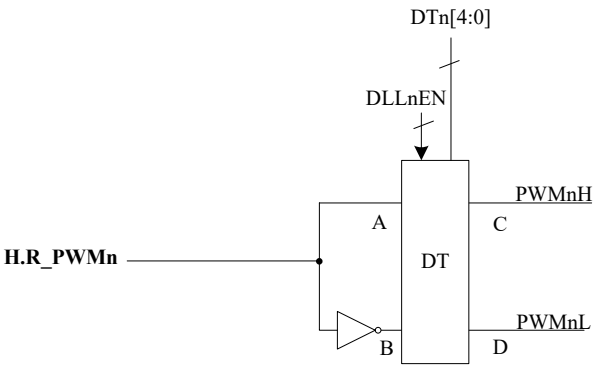
自动调节电路

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

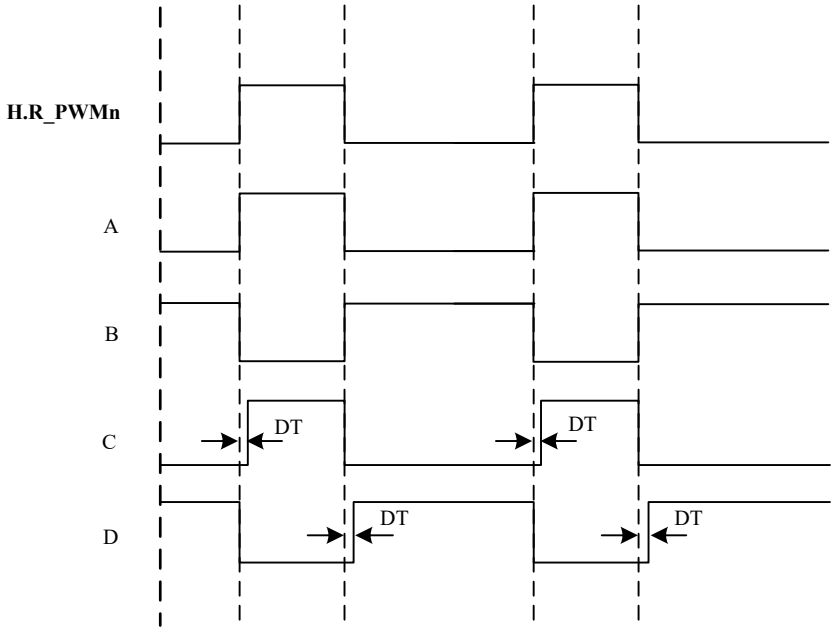
- 步骤 1. 设置 ADJnEN 为 0 关闭自动调节电路，并进行初始化设置：
  - 通过 12-bit 的 ADJnMAXH &ADJnMAXL / ADJnMINH &ADJnMINL 寄存器对设置最大 / 最小 duty。注：1<Min<(PWMnD+DLLn)<MAX
  - 配置 6-bit ADJnDT 寄存器位设置每次触发延迟时间。
  - 设置 ADJnC 寄存器中的 ADJnV 位，选择 OUVPn 发生时，duty 调整增减动作
  - 设置 ADJnS 寄存器选择 OUVPn 发生时，增减 duty 的步数（0~15）。
  - 过压及欠压电压值设定。注：UVPn < 目标电压 < OVPn
- 步骤 2. PWMn 启动：
  - PWMn 初始化，包括 PWMnP 和 (PWMnD+DLLn) 初始化。
  - 设置 PWMnON 为高以开启 PWMn 发生器。
  - 读取 OUVPn 输出后，调节 Duty，使输出电压 = 目标电压
- 步骤 3. 设置 ADJnEN 位为高以开启自动调节电路，OVPnL / UVPnL 电平可触发自动调节。
- 步骤 4. 若有 UVPn 中断发生，可通过 UVPn 和 OCP 比较器输出搭配 F/W 作时间延迟判断是负载瞬间加重导致的 UVP 还是外部设备短路造成的。
- 步骤 5. 若有 OVPn 中断发生，可通过 OVP 比较器输出搭配 F/W 作时间延迟判断是负载瞬间减少导致的 OVP 还是反馈电路异常造成的。

死区时间插入

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



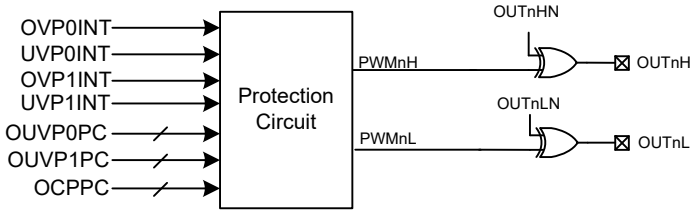
Complementary PWM output
(with dead time control)



注：上图中的 C 和 D 为 PWMn 互补信号插入死区时间后的输出信号

保护与反相控制

在 H.R_PWM 互补信号对中插入了一个死区时间可以预防过大的直流电流。但这两个信号也可能由于误操作或电气噪声等一些非预期的原因而同时处于无效状态。该系列单片机提供了一个保护功能，即在 PWMnH 或 PWMnL 信号为无效状态时强制使这两个信号输出反相信号。反相控制电路通过 OUTPC0 寄存器中的 OUTnHN 或 OUTnLN 位决定了是否将信号进行反相。



该系列单片机还提供了过电流保护与过压 / 欠压保护功能对 PWMn 的输出进行保护，具体操作可参考过电流保护章节中 OCPPC 寄存器与过压 / 欠压保护章节中 OUVPnPC 寄存器介绍。当有过电流，过电压或欠电压情况发生时或当 PWMn 除能时，输出的 OUT1H&OUT1L 及 OUT0H&OUT0L 的状态为高还是为低可通过 OUTPC0 寄存器的 OUT1HS, OUT1LS, OUT0HS 及 OUT0LS 位进行设置。当有 OCP, OVP, UVP 状况发生时会产生中断以通知 MCU 进行处理。一旦 OCP, OVP, UVP 状况解除，PWMn 将恢复输出。关于过电压保护及过压 / 欠压保护的更多内容，可参考相关章节。

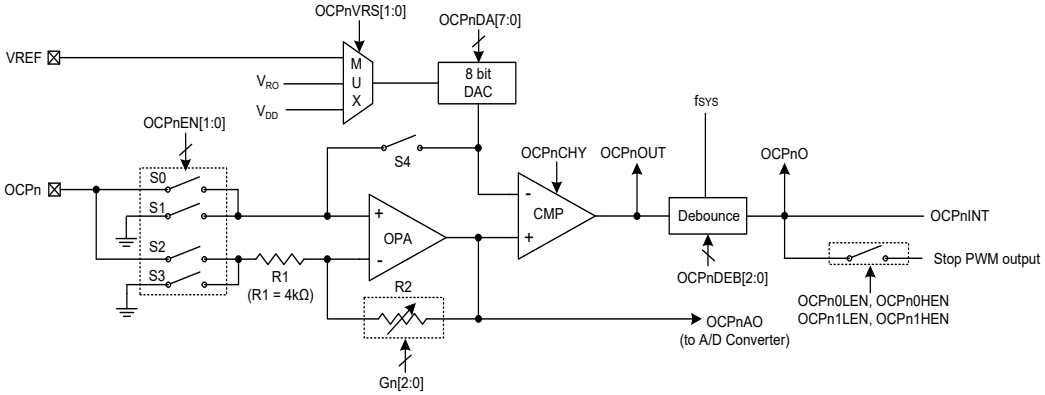
编程注意事项

读写流程如下步骤所示：

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

过电流保护 – OCP

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



注：1. V_{RO} 为 A/D 转换器单位增益缓冲器输出。
2. OCPnAO 信号可输入到 A/D 转换器进行量测。

OCP 电路方框图 (n=0 或 1)

OCP 操作

OCPn 电路可对输入的电流进行监测，防止输入的电流超过指定的值。从 OCPn 引脚输入的电流先被转换为一个电压信号并通过 OCPn 电路内置的可编程增益放大器 PGA 进行放大（1~50 倍增益，通过 OCPnC1 寄存器的 Gn2~Gn0 位设置）。放大后的电压信号输入到内置的比较器电路并与 D/A 转换器输出值进行比较。其中 D/A 转换器输出值由选择的 DAC 参考电压（可通过 OCPnC0 寄存器的 OCPnVRS[1:0] 位选择来自 V_{DD}, V_{RO} 或外部 V_{REF} 引脚输入）和 8-bit OCPnDA 寄存器值决定。比较器输出的 OCPnCOUT 先通过去抖电路处理（去抖周期由 OCPnC1 寄存器的 OCPnDEB[2:0] 位选择），之后输出 OCPnO 信号，用于指明是否有过电流状况发生。若无过电流情况，则 OCPnO 输出为 0。OCPnO 输出为 1，则表明有过电流情况，此时若相关中断使能，则会产生过电流保护中断。

此系列单片机针对高精度 PWM 发生器输出 OUT0H/OUT0L, OUT1H/OUT1L 信号提供了过电流保护控制，此功能通过 OCPPC 寄存器设置。若 PWMn 输出过电流保护功能使能，则当过电流情况发生时，这些信号将被强制输出低或高（由 OUTPC0 寄存器的 OUT0HS/OUT0LS, OUT1HS/OUT1LSW 位设置）。同时 OCPn 也会产生中断信号以告知 MCU。当过电流情况解除后，OUT0H/OUT0L, OUT1H/OUT1L 将会恢复正常输出。PWMn 输出 OUTnH/OUTnL 过电流保护功能设置可通过寄存器 OCPPC 及 OUTPC0 寄存器完成。

OCP 寄存器介绍

OCP 功能的所有操作由一系列的寄存器控制。一个寄存器用于为 OCP 电路提供参考电压。两个寄存器用于运算放大器和比较器输入失调校准。两个控制寄存器用于控制 OCP 功能，D/A 转换器参考电压选择，PGA 增益选择，比较器去抖动时间以及迟滞功能。剩下的 OCPPC 寄存器用于设置 PWMn 输出信号 OUT0H/OUT0L, OUT1H/OUT1L 的过电流保护控制。

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

OCPn 寄存器列表 (n=0 或 1)

OCPnC0 寄存器

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

Bit 7~6 **OCPnEN[1:0]**：OCPn 工作模式选择
00：OCPn 除能，S1 和 S3 on，S0 和 S2 off
01：同相模式，S0 和 S3 on，S1 和 S2 off
10：反相模式，S1 和 S2 on，S0 和 S3 off
11：校准模式，S1 和 S3 on，S0 和 S2 off

Bit 5~4 **OCPnVRS[1:0]**：OCPn DAC 参考电压选择
00：来自 V_{DD}
01：来自 VREF 引脚输入
10：来自内部 V_{RO}
11：来自 V_{DD}
注意：若设置 OCPnVRS[1:0] 位为 “10” 选择 OCPn DAC 参考电压来自内部 V_{RO}。需注意因 V_{RO} 为 A/D 转换器内部单位增益缓冲器输出，因此需提前设置 ADVBGEN 位为高以使能 Bandgap 及单位增益缓冲器功能。

Bit 3 **OCPnCHY**：OCPn 比较器迟滞功能控制位
0：除能
1：使能

Bit 2~1 未定义，读为“0”

Bit 0 **OCPnO**：OCPn 数字输出位
0：未有过电流情况发生
1：发生过电流情况

OCPnC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	Gn2	Gn1	Gn0	OCPnDEB2	OCPnDEB1	OCPnDEB0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6　未定义，读为“0”

Bit 5~3　**Gn[2:0]**：R2/R1 比率选择，从而定义 PGA 增益

000：单位增益缓冲器（同相模式）或 R2/R1 = 1（反相模式）

001：R2/R1=5

010：R2/R1=10

011：R2/R1=15

100：R2/R1=20

101：R2/R1=30

110：R2/R1=40

111：R2/R1=50

这三位用于选择 R2/R1 比率，从而定义 PGA 工作在同相或反相模式时的增益值。反相和同相模式中的 PGA 增益计算公式请参见“输入电压范围”章节。

Bit 2~0　**OCPnDEB[2:0]**：OCPn 输出滤波去抖动时间选择

000：旁路，无去抖动

001：(1~2) × t_{DEB}

010：(3~4) × t_{DEB}

011：(7~8) × t_{DEB}

100：(15~16) × t_{DEB}

101：(31~32) × t_{DEB}

110：(63~64) × t_{DEB}

111：(127~128) × t_{DEB}

注：$t_{DEB}=1/f_{SYS}$

OCPnDA 寄存器

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

Bit 7~0　**D7~D0**：OCPn DAC 输出电压控制位

OCPn DAC 输出 =（DAC 参考电压 /256) × OCPnDA[7:0]

OCPnOCAL 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OCPnOOFM	OCPnORSP	OCPnOOF5	OCPnOOF4	OCPnOOF3	OCPnOOF2	OCPnOOF1	OCPnOOF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	0	0	0	0	0

Bit 7 **OCPnOOFM**：OCPn 运算放大器正常工作或输入失调校准模式选择位

0：正常工作模式

1：输入失调校准模式

此位用来选择 OCPn 运算放大器输入校准功能。在设置此位之前，应先设置 OCPnEN[1:0] 位为“11” 然后才能设置 OCPnOOFM 为 1 以选择输入失调校准模式，之后需将 OCPnCOFM 位清零。运算放大器工作于输入失调校准模式。更具体操作参考“运算放大器输入失调校准”章节。

Bit 6 **OCPnORSP**：OCPn 运算放大器输入失调校准参考电压输入选择

0：选择负输入作为参考输入

1：选择正输入作为参考输入

Bit 5~0 **OCPnOOF[5:0]**：OCP 运算放大器输入失调电压校准值

这 6 位的寄存器位供运算放大器输入失调校准操作使用，并用于重新储存 OCPn 运算放大器输入失调校准值。更多详细内容可参考“运算放大器输入失调校准”章节。

OCPnCCAL 寄存器

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

Bit 7 **OCPnCOUT**：OCPn 比较器输出，正逻辑（只读）

0：输出 0（即正端输入电压 < 负端输入电压）

1：输出 1（即正端输入电压 > 负端输入电压）

此位用于当 OCPn 工作在输入失调校准模式时，显示正输入电压是否大于负输入电压。若 OCPnCPOUT 设为“1”，表示正输入电压大于负输入电压，否则正输入电压小于负输入电压。

Bit 6 **OCPnCOFM**：OCPn 比较器正常工作或输入失调校准模式选择位

0：正常工作模式

1：输入失调校准模式

此位用于选择 OCPn 比较器输入失调校准功能。在设置此位之前，应先设置 OCPnEN[1:0] 位为“11” 然后才能设置 OCPnCOFM 为 1 以选择输入失调校准模式，之后需将 OCPnOOFM 位清零。比较器输入失调校准模式使能。更具体操作参考“比较器输入失调校准”章节。

Bit 5 **OCPnCRSP**：OCPn 比较器输入失调校准参考电压输入选择

0：选择负输入作为参考输入

1：选择正输入作为参考输入

Bit 4~0 **OCPnCOF[4:0]**：OCPn 比较器输入失调校准值

这 6 位的寄存器位供比较器输入失调校准操作时使用，并用于重复储存校准值。更多详细内容可参考“比较器输入失调校准”章节。

OCPPC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	OCP11LEN	OCP11HEN	OCP10LEN	OCP10HEN	OCP01LEN	OCP01HEN	OCP00LEN	OCP00HEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **OCP11LEN**：OUT1L 过电流保护 1 使能控制

0：除能

1：使能

此位用于控制 OUT1L 信号的过电流保护 1 功能。若设置此位为 0，除能 OUT1L 的过电流保护 1 功能，则当有 OCP1 状况发生时，OUT1L 的输出不受影响。若设置此位为 1，使能 OUT1L 的过电流保护 1 功能，则当有 OCP1 状况发生时，OUT1L 的输出将由 OUTPC0 寄存器的 OUT1LS 位控制强制输出高或低电平。

Bit 6 **OCP11HEN**：OUT1H 过电流保护 1 使能控制

0：除能

1：使能

此位用于控制 OUT1H 信号的过电流保护 1 功能。若设置此位为 0，除能 OUT1H 的过电流保护 1 功能，则当有 OCP1 状况发生时，OUT1H 的输出不受影响。若设置此位为 1，使能 OUT1H 的过电流保护 1 功能，则当有 OCP1 状况发生时，OUT1H 的输出将由 OUTPC0 寄存器的 OUT1HS 位控制强制输出高或低电平。

Bit 5 **OCP10LEN**：OUT0L 过电流保护 1 使能控制

0：除能

1：使能

此位用于控制 OUT0L 信号的过电流保护 1 功能。若设置此位为 0，除能 OUT0L 的过电流保护 1 功能，则当有 OCP1 状况发生时，OUT0L 的输出不受影响。若设置此位为 1，使能 OUT0L 的过电流保护 1 功能，则当有 OCP1 状况发生时，OUT0L 的输出将由 OUTPC0 寄存器的 OUT0LS 位控制强制输出高或低电平。

Bit 4 **OCP10HEN**：OUT0H 过电流保护 1 使能控制

0：除能

1：使能

此位用于控制 OUT0H 信号的过电流保护 1 功能。若设置此位为 0，除能 OUT0H 的过电流保护 1 功能，则当有 OCP1 状况发生时，OUT0H 的输出不受影响。若设置此位为 1，使能 OUT0H 的过电流保护 1 功能，则当有 OCP1 状况发生时，OUT0H 的输出将由 OUTPC0 寄存器的 OUT0HS 位控制强制输出高或低电平。

Bit 3 **OCP01LEN**：OUT1L 过电流保护 0 使能控制

0：除能

1：使能

此位用于控制 OUT1L 信号的过电流保护 0 功能。若设置此位为 0，除能 OUT1L 的过电流保护 0 功能，则当有 OCP0 状况发生时，OUT1L 的输出不受影响。若设置此位为 1，使能 OUT1L 的过电流保护 0 功能，则当有 OCP0 状况发生时，OUT1L 的输出将由 OUTPC0 寄存器的 OUT1LS 位控制强制输出高或低电平。

Bit 2 **OCP01HEN**：OUT1H 过电流保护 0 使能控制

0：除能

1：使能

此位用于控制 OUT1H 信号的过电流保护 0 功能。若设置此位为 0，除能 OUT1H 的过电流保护 0 功能，则当有 OCP0 状况发生时，OUT1H 的输出不受影响。若设置此位为 1，使能 OUT1H 的过电流保护 0 功能，则当有 OCP0 状况发生时，OUT1H 的输出将由 OUTPC0 寄存器的 OUT1HS 位控制强制输出高或低电平。

Bit 1 **OCP00LEN**：OUT0L 过电流保护 0 使能控制

0：除能

1：使能

此位用于控制 OUT0L 信号的过电流保护 0 功能。若设置此位为 0，除能 OUT0L 的过电流保护 0 功能，则当有 OCP0 状况发生时，OUT0L 的输出不受影响。若设置此位为 1，使能 OUT0L 的过电流保护 0 功能，则当有 OCP0 状况发生时，OUT0L 的输出将由 OUTPC0 寄存器的 OUT0LS 位控制强制输出高或低电平。

Bit 0 **OCP00HEN**：OUT0H 过电流保护 0 使能控制

0：除能

1：使能

此位用于控制 OUT0H 信号的过电流保护 0 功能。若设置此位为 0，除能 OUT0H 的过电流保护 0 功能，则当有 OCP0 状况发生时，OUT0H 的输出不受影响。若设置此位为 1，使能 OUT0H 的过电流保护 0 功能，则当有 OCP0 状况发生时，OUT0H 的输出将由 OUTPC0 寄存器的 OUT0HS 位控制强制输出高或低电平。

输入电压范围

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

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

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

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

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

- 输入电压 0>V_{IN}>-0.2V 时，工作在反相模式下的 PGA 以及 PGA 输出可由以下公式获得。

$$V_{OUT} = -\frac{R_2}{R_1} \times V_{IN}$$

- 注意，若输入电压为负，其值不可小于 -0.2V，否则会产生漏电流。

OCPn 运算放大器和比较器失调校准

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

运算放大器输入失调校准

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

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

步骤 3. OCPnOOF[5:0] 的值加一，读 OCPnCOUT 位。
若 OCPnCOUT 位状态不变，则重复步骤 3 直到 OCPnCOUT 位状态改变。
若 OCPnCOUT 位状态改变，则记录下 OCPnOOF 值为 V_{OOS1}，前往步骤 4。

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

步骤 5. OCPnOOF[5:0] 的值减一，读 OCPnCOUT 位。
若 OCPnCOUT 位状态未改变，重复步骤 5 直到 OCPnCOUT 位状态改变。
若 OCPnCOUT 位状态改变，则记录下 OCPnOOF 值为 V_{OOS2}，前往步骤 6.

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

这里的 $V_{OOS} = \frac{V_{OOS1} + V_{OOS2}}{2}$

比较器输入失调校准

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

步骤 2. 设置 OCPnCOF[4:0]=00000，此时开始读 OCPnCOUT 位。

步骤 3. OCPnCOF[4:0] 的值加一，读 OCPnCOUT 位。
若 OCPnCOUT 位状态不变，则重复步骤 3 直到 OCPnCOUT 位状态改变。
若 OCPnCOUT 位状态改变，则记录下 OCPnCOF 值为 V_{COS1}，前往步骤 4。

步骤 4. 设置 OCPnCOF[4:0]=11111，此时开始读 OCPnCOUT 位。

步骤 5. OCPnCOF[4:0] 的值减一，读 OCPnCOUT 位。
若 OCPnCOUT 位状态未改变，重复步骤 5 直到 OCPnCOUT 位状态改变。
若 OCPnCOUT 位状态改变，则记录下 OCPnCOF 值为 V_{COS2}，前往步骤 6.

步骤 6. 将比较器输入失调校准值 V_{COS}，重新存入 OCPnCOF[4:0] 位段。此时失调校准步骤完成。

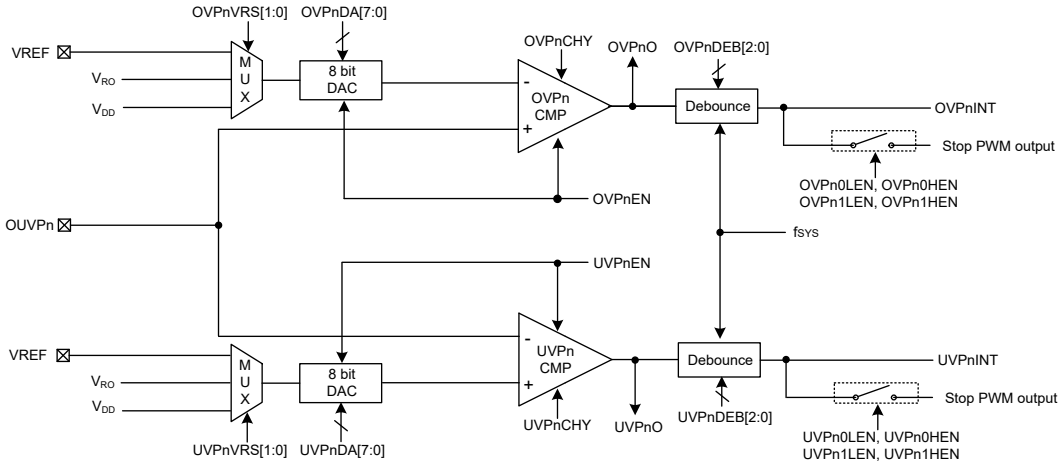
这里的 $V_{COS} = \frac{V_{COS1} + V_{COS2}}{2}$

过压 / 欠压保护 – OUVP

该系列单片机内置过压 / 欠压保护电路，用于电池充 / 放电应用的安全防护。

OUVP 操作

OUVP 电路提供过电压保护和欠电压保护功能。



注：V_{RO} 为 A/D 转换器单位增益缓冲器输出。

OUVP 电路方框图 (n=0 或 1)

过压保护功能

为防止输出电压超过指定的电压值，过压保护电路将 OUVPn 引脚输入的电压与参考电压进行比较。用于比较的参考电压由内部 DAC 电路输出，可通过 8-bit OVPnDA 寄存器控制输出电压的值。8-bit DAC 参考电压可通过 OVPnVRS[1:0] 位选择来自 V_{DD}, V_{RO} 或外部 VREF 引脚输入。一旦 OUVPn 引脚上输入的电压值大于设定的参考电压值，OVPnO 的值将从 0 变为 1。OVPnINT 为 OVPnO 经过去抖处理后的输出，用于指示 OUVPn 引脚上的输入电压是否超过了参考电压值，若为 1 且相关中断使能，会触发 OVPn 中断。通过设置 OVPnCHY 位可使能 OVPn 电路的比较器迟滞功能。

欠压保护功能

为防止输出电压低于指定的电压值，欠压保护电路将 OUVPn 引脚输入的电压与参考电压进行比较。用于比较的参考电压由内部 DAC 电路输出，可通过 8-bit UVPnDA 寄存器控制输出电压的值。8-bit DAC 参考电压可通过 UVPnVRS[1:0] 位选择来自 V_{DD}, V_{RO} 或外部 VREF 引脚输入。一旦 OUVPn 引脚上输入的电压值低于设定的参考电压值，UVPnO 的值将从 0 变为 1。UVPnINT 为 UVPnO 经过去抖处理后的输出，用于指示 OUVPn 引脚上的输入电压是否低于参考电压值，若为 1 且相关中断使能，会触发 UVPn 中断。通过设置 UVPnCHY 位可使能 UVPn 电路的比较器迟滞功能。

此系列单片机针对高精度 PWM 发生器输出 OUT0H/OUT0L, OUT1H/OUT1L 信号提供了过压 / 欠压保护控制，此功能通过 OUVPnPC 寄存器设置。若 PWMn 输出过压 / 欠压保护功能使能，则当有过压或欠压情况发生时，这些信号将被强制输出低或高（由 OUTPC0 寄存器的 OUT0HS/OUT0LS, OUT1HS/OUT1LSW 位设置）。OVPn/UVPn 同时也会产生中断信号以告知 MCU。当过压或欠压情况解除后，OUT0H/OUT0L, OUT1H/OUT1L 将会恢复正常输出。PWMn 输出保护具体设置可参考相关章节内容。

OUVP 寄存器介绍

过压和欠压保护的整个操作由一系列寄存器控制。

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

OUVP 寄存器列表（n=0 或 1）

OVPnDA 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 **D[7:0]**：OVPn DAC 输出电压控制位
OVPn DAC 输出 = (OVPn DAC 参考电压) × (OVPnDA[7:0])/256

UVPnDA 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 **D[7:0]**：UVPn DAC 输出电压控制位
UVPn DAC 输出 = (UVPn DAC 参考电压) × (UVPnDA[7:0])/256

OUVPnC0 寄存器

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

Bit 7~6 未定义，读为“0”

Bit 5 **OVPnEN**：OVPn 功能控制
0：除能
1：使能
若 OVPnEN=0，过电压保护 n 功能除能，则不会产生功耗，同时 OVPn 电路的比较器和 DAC 也会关闭。

Bit 4 **OVPnCHY**：OVPn 比较器迟滞功能控制
0：除能
1：使能

Bit 3~2 **OVPnVRS[1:0]**：OVPn DAC 参考电压选择

00：来自 V_{DD}

01：来自 VREF 引脚输入

10：来自内部 V_{RO}

11：来自 V_{DD}

注意：若设置 OVPnVRS[1:0] 位为 “10” 选择 OVPn DAC 参考电压来自内部 V_{RO}。需注意因 V_{RO} 为 A/D 转换器内部单位增益缓冲器输出，因此需提前设置 ADVBGEN 位为高以使能内部 Bandgap 及单位增益缓冲器功能。

Bit 1~0 **OVPnDEB[1:0]**：OVPn 比较器去抖时间选择

00：无去抖

01：(7~8) × $1/f_{SYS}$

10：(15~16) × $1/f_{SYS}$

11：(31~32) × $1/f_{SYS}$

OUVPnC1 寄存器

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

Bit 7~6 未定义，读为“0”

Bit 5 **UVPnEN**：UVPn 功能控制

0：除能

1：使能

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

Bit 4 **UVPnCHY**：UVPn 比较器迟滞功能控制

0：除能

1：使能

Bit 3~2 **UVPnVRS[1:0]**：UVPn DAC 参考电压选择

00：来自 V_{DD}

01：来自 VREF 引脚输入

10：来自内部 V_{RO}

11：来自 V_{DD}

注意：若设置 UVPnVRS[1:0] 位为 “10” 选择 UVPn DAC 参考电压来自内部 V_{RO}。需注意因 V_{RO} 为 A/D 转换器内部单位增益缓冲器输出，因此需提前设置 ADVBGEN 位为高以使能内部 Bandgap 及单位增益缓冲器功能。

Bit 1~0 **UVPnDEB[1:0]**：UVPn 比较器去抖时间选择

00：无去抖

01：(7~8) × $1/f_{SYS}$

10：(15~16) × $1/f_{SYS}$

11：(31~32) × $1/f_{SYS}$

OUVPnC2 寄存器

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

Bit 7 **OVPnO**：OVPn 比较器输出位
0：输出 0（即正端输入电压 < 负端输入电压）
1：输出 1（即正端输入电压 > 负端输入电压）

Bit 6 **OVPnCOFM**：OVPn 比较器正常工作或输入失调校准模式选择位
0：正常工作模式
1：输入失调校准模式

Bit 5 **OVPnCRS**：OVPn 比较器输入失调校准参考电压输入选择
0：选择负输入作为参考输入
1：选择正输入作为参考输入
OVPnCRS 位用于选择参考输入电压来自 OVPn D/A 转换器或是来自外部输入。注意仅在设置 OVPnCOFM 位为 1 选择 OVPn 比较器工作在输入失调校准模式时，此位的选择才有效。

Bit 4~0 **OVPnCOF[4:0]**：OVPn 比较器输入失调校准值

OUVPnC3 寄存器

Bit	7	6	5	4	3	2	1	0
Name	UVPnO	UVPnCOFM	UVPnCRS	UVPnCOF4	UVPnCOF3	UVPnCOF2	UVPnCOF1	UVPnCOF0
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	1	0	0	0	0

Bit 7 **UVPnO**：UVPn 比较器输出位
0：输出 0（即正端输入电压 < 负端输入电压）
1：输出 1（即正端输入电压 > 负端输入电压）

Bit 6 **UVPnCOFM**：UVPn 比较器正常工作或输入失调校准模式选择位
0：正常工作模式
1：输入失调校准模式

Bit 5 **UVPnCRS**：UVPn 比较器输入失调校准参考电压输入选择
0：选择负输入作为参考输入
1：选择正输入作为参考输入
UVPnCRS 位用于选择参考输入电压来自 UVPn D/A 转换器或是来自外部输入。注意仅在设置 UVPnCOFM 位为 1 选择 UVPn 比较器工作在输入失调校准模式时，此位的选择才有效。

Bit 4~0 **UVPnCOF[4:0]**：UVPn 比较器输入失调校准值

OUVPnPC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	UVPn1LEN	UVPn1HEN	UVPn0LEN	UVPn0HEN	OVPn1LEN	OVPn1HEN	OVPn0LEN	OVPn0HEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **UVPn1LEN**：OUT1L 欠压保护 n 使能控制

0：除能

1：使能

此位用于控制 OUT1L 信号的欠压保护 n 功能。若设置此位为 0，除能 OUT1L 的欠压保护 n 功能，则当有 UVPn 状况发生时，OUT1L 的输出不受影响。若设置此位为 1，使能 OUT1L 的欠压保护 n 功能，则当有 UVPn 状况发生时，OUT1L 的输出将由 OUTPC0 寄存器的 OUT1LS 位控制强制输出高或低电平。

Bit 6 **UVPn1HEN**：OUT1H 欠压保护 n 使能控制

0：除能

1：使能

此位用于控制 OUT1H 信号的欠压保护 n 功能。若设置此位为 0，除能 OUT1H 的欠压保护 n 功能，则当有 UVPn 状况发生时，OUT1H 的输出不受影响。若设置此位为 1，使能 OUT1H 的欠压保护 n 功能，则当有 UVPn 状况发生时，OUT1H 的输出将由 OUTPC0 寄存器的 OUT1HS 位控制强制输出高或低电平。

Bit 5 **UVPn0LEN**：OUT0L 欠压保护 n 使能控制

0：除能

1：使能

此位用于控制 OUT0L 信号的欠压保护 n 功能。若设置此位为 0，除能 OUT0L 的欠压保护 n 功能，则当有 UVPn 状况发生时，OUT0L 的输出不受影响。若设置此位为 1，使能 OUT0L 的欠压保护 n 功能，则当有 UVPn 状况发生时，OUT0L 的输出将由 OUTPC0 寄存器的 OUT0LS 位控制强制输出高或低电平。

Bit 4 **UVPn0HEN**：OUT0H 欠压保护 n 使能控制

0：除能

1：使能

此位用于控制 OUT0H 信号的欠压保护 n 功能。若设置此位为 0，除能 OUT0H 的欠压保护 n 功能，则当有 UVPn 状况发生时，OUT0H 的输出不受影响。若设置此位为 1，使能 OUT0H 的欠压保护 n 功能，则当有 UVPn 状况发生时，OUT0H 的输出将由 OUTPC0 寄存器的 OUT0HS 位控制强制输出高或低电平。

Bit 3 **OVPn1LEN**：OUT1L 过压保护 n 使能控制

0：除能

1：使能

此位用于控制 OUT1L 信号的过压保护 n 功能。若设置此位为 0，除能 OUT1L 的过压保护 n 功能，则当有 OVPn 状况发生时，OUT1L 的输出不受影响。若设置此位为 1，使能 OUT1L 的过压保护 n 功能，则当有 OVPn 状况发生时，OUT1L 的输出将由 OUTPC0 寄存器的 OUT1LS 位控制强制输出高或低电平。

Bit 2 **OVPn1HEN**：OUT1H 过压保护 n 使能控制

0：除能

1：使能

此位用于控制 OUT1H 信号的过压保护 n 功能。若设置此位为 0，除能 OUT1H 的过压保护 n 功能，则当有 OVPn 状况发生时，OUT1H 的输出不受影响。若设置此位为 1，使能 OUT1H 的过压保护 n 功能，则当有 OVPn 状况发生时，OUT1H 的输出将由 OUTPC0 寄存器的 OUT1HS 位控制强制输出高或低电平。

Bit 1 **OVPn0LEN**：OUT0L 过压保护 n 使能控制

0：除能

1：使能

此位用于控制 OUT0L 信号的过压保护 n 功能。若设置此位为 0，除能 OUT0L 的过压保护 n 功能，则当有 OVPn 状况发生时，OUT0L 的输出不受影响。若设置此位为 1，使能 OUT0L 的过压保护 n 功能，则当有 OVPn 状况发生时，OUT0L 的输出将由 OUTPC0 寄存器的 OUT0LS 位控制强制输出高或低电平。

Bit 0 **OVPn0HEN**：OUT0H 过压保护 n 使能控制

0：除能

1：使能

此位用于控制 OUT0H 信号的过压保护 n 功能。若设置此位为 0，除能 OUT0H 的过压保护 n 功能，则当有 OVPn 状况发生时，OUT0H 的输出不受影响。若设置此位为 1，使能 OUT0H 的过压保护 n 功能，则当有 OVPn 状况发生时，OUT0H 的输出将由 OUTPC0 寄存器的 OUT0HS 位控制强制输出高或低电平。

OVPn 和 UVPn 比较器失调校准

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

OVPn 比较器输入失调校准

步骤 1. 设置 OVPnCOFM=1, OVPnCRS=1 则 OVPn 比较器将工作在输入失调校准模式。为确保校准后，V_{OS} 值尽可能的小，失调校准时输入的参考电压应与比较器正常工作时输入的直流工作电压一致。

步骤 2. 设置 OVPnCOF[4:0]=00000，此时开始读 OVPnO 位。

步骤 3. OVPnCOF[4:0] 的值加一，读 OVPnO 位。
若 OVPnO 位状态不变，则重复步骤 3 直到 OVPnO 位状态改变。
若 OVPnO 位状态改变，则记录下 OVPnCOF 值为 V_{OS1}，前往步骤 4。

步骤 4. 设置 OVPnCOF[4:0]=11111，此时开始读 OVPnO 位。

步骤 5. OVPnCOF[4:0] 的值减一，读 OVPnO 位。
若 OVPnO 位状态未改变，重复步骤 5 直到 OVPnO 位状态改变。
若 OVPnO 位状态改变，则记录下 OCPnCOF 值为 V_{OS2}，前往步骤 6.

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

UVPn 比较器输入失调校准

步骤 1. 设置 UVPnCOFM=1, UVPnCRS=1 则 UVPn 比较器将工作在输入失调校准模式。为确保校准后，V_{OS} 值尽可能的小，失调校准时输入的参考电压应与比较器正常工作时输入的直流工作电压一致。

步骤 2. 设置 UVPnCOF[4:0]=00000，此时开始读 UVPnO 位。

步骤 3. UVPnCOF[4:0] 的值加一，读 UVPnO 位。
若 UVPnO 位状态不变，则重复步骤 3 直到 UVPnO 位状态改变。
若 UVPnO 位状态改变，则记录下 UVPnCOF 值为 V_{OS1}，前往步骤 4。

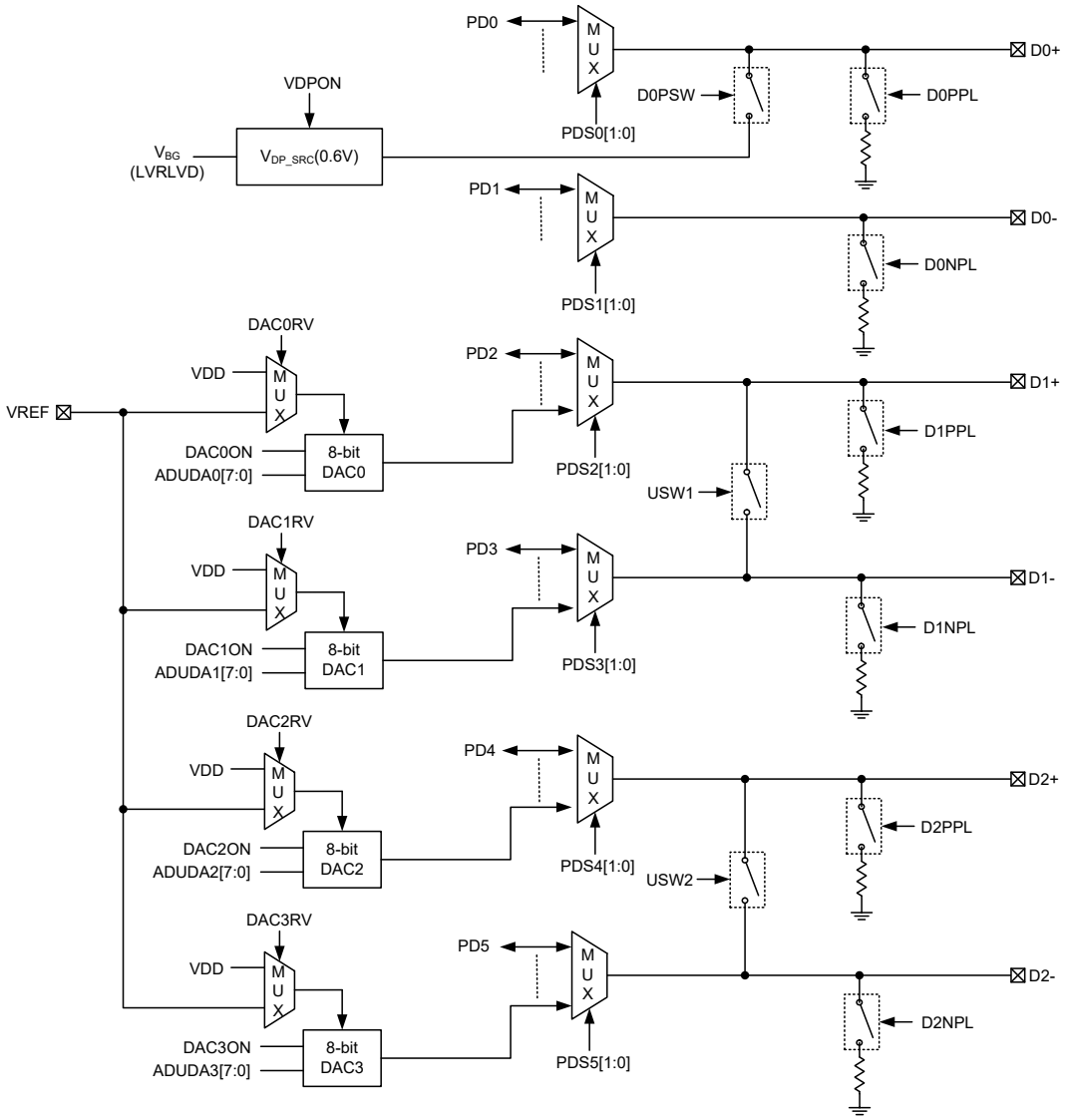
步骤 4. 设置 UVPnCOF[4:0]=11111，此时开始读 UVPnO 位。

步骤 5. UVPnCOF[4:0] 的值减一，读 UVPnO 位。
若 UVPnO 位状态未改变，重复步骤 5 直到 UVPnO 位状态改变。
若 UVPnO 位状态改变，则记录下 UCPnCOF 值为 V_{OS2}，前往步骤 6.

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

USB 自动检测

此系列单片机包含三对 USB 端口 D0+/D0-，D1+/D1- 以及 D2+/D2-，用于执行充 / 放电自动检测功能。用户可以通过监测 USB 接口端的电压与电流，识别连接到 USB 端口的是一个专用的充电器，便携设备，一般的 USB 接口或是带 USB 接口的充电装置。



USB 自动检测功能方框图

D0+/D0- 自动检测

通过正确设置 ADUC1 寄存器的 VDPON 位以及 D0PS 位使 D0PSW 开关 on，D0+ 端口可以输出 0.6V 电压，V_{DP_SRC}。但需注意的是，只有当 D0+ 端口通过相关的引脚共用功能选择寄存器设置为模拟或数字输入类型时，再设置 D0PS 位为高，才能使开关 D0PSW 闭合。当此端口连接了专用的充电器，在 D0+ 端口会输出 0.6V 电压。D0+ 和 D0- 引脚与其它通用 I/O 功能或 A/D 转换器功能共用引脚，可通过 PDPS0 寄存器的 PDS0[1:0] 及 PDS1[1:0] 位进行选择。D0+ 和 D0- 可以通过 ADUC2 寄存器的 D0PPL 和 D0NPL 位选择是否连接下拉电阻。

D1+/D1- 和 D2+/D2- 自动检测

USB 自动检测电路中，包含四个 8-bit D/A 转换器即 DAC0~DAC3。这四个 DACn 可通过 ADUC0 寄存器中的 DACnON 位进行独立的使能设置。DACn 输出的电压由 ADUC0 寄存器中的 DACnRV 位选择的 DACn 参考电压及相应的 8-bit 寄存器 ADUDAn 决定。在 D1+ 和 D1- 引脚间有一个由 USW1 位控制的模拟开关 USW1，同样的，在 D2+ 和 D2- 间也有一个由 USW2 位控制的模拟开关 USW2。但在设置这些开关时需注意只有当 D2+/D2- 其中一个引脚或 D1+/D1- 其中一个引脚通过引脚共用功能选择寄存器设置为模拟或数字输入类型时，设置 USW2 或 USW1 为高，才能使相应的开关为闭合。D1+ 和 D1- 或 D2+ 和 D2- 可通过设置 ADUC2 寄存器 D1PPL 和 D1NPL 或 D2PPL 和 D2NPL 位选择是否连接下拉电阻。

USB 自动检测寄存器

USB 自动检测功能是通过一系列寄存器进行控制的。

寄存器名称	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADUC0	DAC3RV	DAC2RV	DAC1RV	DAC0RV	DAC3ON	DAC2ON	DAC1ON	DAC0ON
ADUC1	—	—	—	—	USW2	USW1	VDPON	D0PS
ADUC2	—	—	D2NPL	D2PPL	D1NPL	D1PPL	D0NPL	D0PPL
ADUDA0	D7	D6	D5	D4	D3	D2	D1	D0
ADUDA1	D7	D6	D5	D4	D3	D2	D1	D0
ADUDA2	D7	D6	D5	D4	D3	D2	D1	D0
ADUDA3	D7	D6	D5	D4	D3	D2	D1	D0

USB 自动检测寄存器列表

ADUC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	DAC3RV	DAC2RV	DAC1RV	DAC0RV	DAC3ON	DAC2ON	DAC1ON	DAC0ON
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 **DAC3RV**：DAC3 参考电压选择
0：V_{DD}
1：VREF 引脚输入

Bit 6 **DAC2RV**：DAC2 参考电压选择
0：V_{DD}
1：VREF 引脚输入

Bit 5 **DAC1RV**：DAC1 参考电压选择
0：V_{DD}
1：VREF 引脚输入

Bit 4 **DAC0RV**：DAC0 参考电压选择
0：V_{DD}
1：VREF 引脚输入

Bit 3 **DAC3ON**：DAC3 使能控制
0：除能
1：使能

Bit 2 **DAC2ON**：DAC2 使能控制
0：除能
1：使能

Bit 1 **DAC1ON**：DAC1 使能控制
0：除能
1：使能

Bit 0 **DAC0ON**：DAC0 使能控制
0：除能
1：使能

ADUC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	USW2	USW1	VDPON	D0PS
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 **USW2**：USW2 开关 on/off 控制
0：Off
1：On
此位用于控制 USW2 开关的 on/off。若 D2+ 或 D2- 其中有一个引脚功能通过引脚共用功能选择寄存器设置为模拟或数字输入类型时，只有设置此位为高，才能使 USW2 开关 ON。

Bit 2 **USW1**：USW1 开关 on/off 控制
0：Off
1：On
此位用于控制 USW1 开关的 on/off。若 D1+ 或 D1- 其中有一个引脚功能通过引脚共用功能选择寄存器设置为模拟或数字输入类型时，只有设置此位为高，才能使 USW1 开关 ON。

Bit 1 **VDPON**：V_{DP_SRC} 电压使能控制
0：除能
1：使能

Bit 0 **D0PS**：D0PSW 开关 on/off 控制
0：Off
1：On
此位用于控制 D0PSW 开关的 on/off。若 D0+ 引脚通过引脚共用功能选择寄存器设置为模拟或数字输入类型时，只有此位为高，才能使 D0PSW 开关为 ON。

ADUC2 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	—	D2NPL	D2PPL	D1NPL	D1PPL	D0NPL	D0PPL
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 **D2NPL**：D2- 引脚下拉电阻控制位
0：除能
1：使能

Bit 4 **D2PPL**：D2+ 引脚下拉电阻控制位
0：除能
1：使能

Bit 3 **D1NPL**：D1- 引脚下拉电阻控制位
0：除能
1：使能

Bit 2 **D1PPL**：D1+ 引脚下拉电阻控制位
0：除能
1：使能

Bit 1 **D0NPL**：D0- 引脚下拉电阻控制位
0：除能
1：使能

Bit 0 **D0PPL**：D0+ 引脚下拉电阻控制位
0：除能
1：使能

ADUDA0 寄存器

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

Bit 7~0 **D7~D0**：8-bit DAC0 输出控制位
DAC0 输出 = (DAC0 参考电压) × (ADUDA0 [7:0]) / 256

ADUDA1 寄存器

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

Bit 7~0 **D7~D0**：8-bit DAC1 输出控制位
DAC1 输出 = (DAC1 参考电压) × (ADUDA1 [7:0]) / 256

ADUDA2 寄存器

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

Bit 7~0 **D7~D0**：8-bit DAC2 输出控制位
DAC2 输出 = (DAC2 参考电压) × (ADUDA2 [7:0]) / 256

ADUDA3 寄存器

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

Bit 7~0 **D7~D0**：8-bit DAC3 输出控制位
DAC3 输出 = (DAC3 参考电压) × (ADUDA3 [7:0]) / 256

串行接口模块 – SIM

此单片机内建一个串行接口模块，包括两种易与外部设备通信的串行接口：四线 SPI 或两线 I^2C 接口。这两种接口具有相当简单的通信协议，单片机可以通过这些接口与外部基于 SPI 或 I^2C 的传感器、闪存等硬件设备通信。因为 SIM 接口引脚是与其它 I/O 引脚共用，因此在使用 SIM 功能前，要先通过相应的引脚共用功能选择寄存器选定 SIM 引脚功能。因为 SPI 和 I^2C 这两种接口共用引脚和寄存器，所以要先通过 SIMC0 寄存器中的 SIM2~SIM0 位选择哪一种通信接口。若 SIM 功能使能且引脚用作 SIM 输入脚，可通过对应上拉电阻控制寄存器选择此脚的上拉电阻。

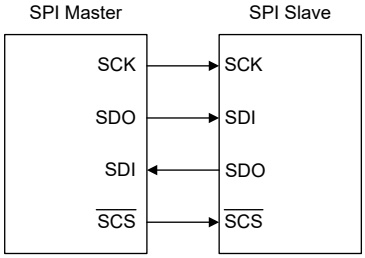
SPI 接口

SPI 接口常用于与外部设备如传感器、闪存或 EEPROM 内存等通信。四线 SPI 接口最初是由摩托罗拉公司研制，是一个有相当简单的通信协议的串行数据接口，这个协议可以简化与外部硬件的编程要求。

SPI 通信模式为全双工模式，且能以主 / 从模式的工作方式进行通信，单片机既可以做为主机，也可以做为从机。虽然 SPI 接口理论上允许一个主机控制多个从机，但此单片机 SPI 只提供了一个从机选择引脚 $\overline{SCS}$。若需要单个主机同时控制多个从机，可使用输入 / 输出引脚选择从机。

SPI 接口操作

SPI 接口是一个全双工串行数据传输器。SPI 接口的四线为：SDI、SDO、SCK 和 $\overline{SCS}$。SDI 和 SDO 是数据的输入和输出线。SCK 是串行时钟线，$\overline{SCS}$ 是从机的选择线。SPI 的接口引脚与普通 I/O 口和 I^2C 的功能脚共用。通过设定相关引脚共用选择位和 SIMC0/SIMC2 寄存器的对应位，来使能 SPI 接口。连接到 SPI 接口的单片机以从主 / 从模式进行通信，且主机完成所有的数据传输初始化，并控制时钟信号。由于单片机只有一个 $\overline{SCS}$ 引脚，所以只能拥有一个从机设备。可通过软件控制 $\overline{SCS}$ 引脚使能与除能，设置 CSEN 位为“1”使能 $\overline{SCS}$ 功能，设置 CSEN 位为“0”，$\overline{SCS}$ 引脚将处于浮空状态。

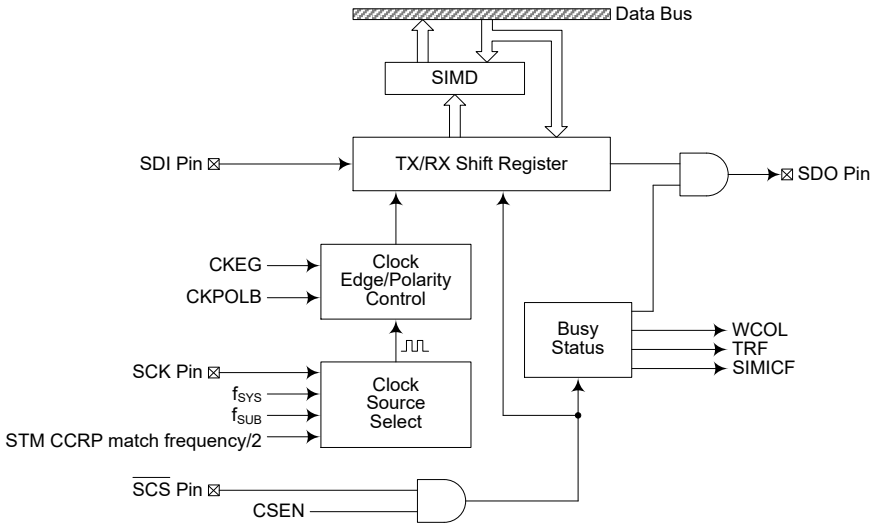


SPI 主 / 从机连接方式

该系列单片机的 SPI 功能具有以下特点：

- 全双工同步数据传输
- 主从模式
- 最低有效位先传或最高有效位先传的数据传输模式
- 传输完成标志位
- 时钟源上升沿或下降沿有效

SPI 接口状态受很多因素的影响，如单片机处于主机或从机的工作模式以及 CSEN、SIMEN 位的状态。



SPI 方框图

SPI 寄存器

有三个内部寄存器用于控制 SPI 接口的所有操作，其中有一个数据寄存器 SIMD、两个控制寄存器 SIMC0 和 SIMC2。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	SIMICF
SIMC2	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF
SIMD	D7	D6	D5	D4	D3	D2	D1	D0

SPI 寄存器列表

SPI 数据寄存器 – SIMD

SIMD 用于存储发送和接收的数据。这个寄存器由 SPI 和 I^2C 功能所共用。在单片机将数据写入到 SPI 总线之前，要传输的数据应先存在 SIMD 中。SPI 总线接收到数据之后，单片机就可以从 SIMD 数据寄存器中读取。所有通过 SPI 传输或接收的数据都必须通过 SIMD 实现。

- **SIMD 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	x	x	x	x	x	x	x	x

“x”为未知

Bit 7~0 **D7~D0**：SIM 数据寄存器位 bit 7 ~ bit 0

SPI 控制寄存器 – SIMC0/SIMC2

单片机中也有两个控制 SPI 接口功能的寄存器，SIMC0 和 SIMC2。应注意的是 SIMC2 与 I^2C 接口功能中的寄存器 SIMA 是同一个寄存器。SPI 功能不会用到寄存器 SIMC1，SIMC1 寄存器仅在工作于 I^2C 接口时才被使用。寄存器 SIMC0 用于控制使能 / 除能功能和设置数据传输的时钟频率。寄存器 SIMC2 用于其它的控制功能如 LSB/MSB 选择，写冲突标志位等。

- **SIMC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	SIMICF
R/W	R/W	R/W	R/W	—	R/W	R/W	R/W	R/W
POR	1	1	1	—	0	0	0	0

Bit 7~5 **SIM2~SIM0**：SIM 工作模式控制位

000：SPI 主机模式；SPI 时钟为 $f_{SYS}/4$

001：SPI 主机模式；SPI 时钟为 $f_{SYS}/16$

010：SPI 主机模式；SPI 时钟为 $f_{SYS}/64$

011：SPI 主机模式；SPI 时钟为 f_{SUB}

100：SPI 主机模式；SPI 时钟为 STM CCRP 匹配频率 /2

101：SPI 从机模式

110：I^2C 从机模式

111：未使用

这几位用于设置 SIM 功能的工作模式，除了选择 I^2C 或 SPI 功能，还可选择 SPI 的主从模式和 SPI 的主机时钟频率。SPI 时钟源可来自于系统时钟和 f_{SUB} 也可以选择来自 STM。若选择的是作为 SPI 从机，则其时钟源从外部主机而得。

Bit 4 未定义，读为“0”

Bit 3~2 **SIMDEB1~SIMDEB0**：I^2C 去抖时间选择位

这些位只有在 SIM 设置成 I^2C 接口模式时才有效。请参考 I^2C 寄存器部分。

Bit 1 **SIMEN**：SIM 控制位

0：除能

1：使能

此位为 SIM 接口的开 / 关控制位。此位为“0”时，SIM 接口除能，SDI、SDO、SCK 和 SCS 或 SDA 和 SCL 脚将失去 SPI 或 I^2C 功能，SIM 工作电流减小到最小值。此位为“1”时，SIM 接口使能。若 SIM 经由 SIM2~SIM0 位设置为工作在 SPI 接口，当 SIMEN 位由低到高转变时，SPI 控制寄存器中的设置不会发生变化，其首先应在应用程序中初始化。若 SIM 经由 SIM2~SIM0 位设置为工作在 I^2C 接口，当 SIMEN 位由低到高转变时，I^2C 控制寄存器中的设置，如 HTX 和 TXAK，将不会发生变化，其首先应在应用程序中初始化，此时相关 I^2C 标志，如 HCF、HAAS、HBB、SRW 和 RXAK，将被设置为其默认状态。

Bit 0 **SIMICF**：SIM SPI 传输未完成标志位

0：未发生

1：发生

此位仅当 SIM 配置在 SPI 从机模式时有效。如果 SPI 工作在从机模式且 SIMEN 和 CSEN 位都为“1”，但在 SPI 数据传输完全结束前 SCS 线被外部主机拉高，SIMICF 和 TRF 位都会被置高。在这种情况下，如果相应的中断功能使能将产生一个中断。然而，如果 SIMICF 位是由软件应用程序设为 1，那么 TRF 位将不会置高，也不会触发中断。

- **SIMC2 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	CKPOLB	CKEG	MLS	CSEN	WCOL	TRF
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 未定义位
用户可通过软件程序对这两位进行读写。

Bit 5 **CKPOLB**：SPI 时钟线的基础状态位
0：当时钟无效时，SCK 引脚为高电平
1：当时钟无效时，SCK 引脚为低电平
此位决定了时钟线的基础状态，若此位为高，当时钟无效时 SCK 为低电平，若此位为低，当时钟无效时 SCK 为高电平。

Bit 4 **CKEG**：SPI 的 SCK 有效时钟边沿类型位
CKPOLB=0
0：SCK 为高电平且在 SCK 上升沿抓取数据
1：SCK 为高电平且在 SCK 下降沿抓取数据
CKPOLB=1
0：SCK 为低电平且在 SCK 下降沿抓取数据
1：SCK 为低电平且在 SCK 上升沿抓取数据
CKEG 和 CKPOLB 位用于设置 SPI 总线上时钟信号输入和输出方式。这两位必须在执行数据传输前先被设置好，否则将产生错误的时钟边沿信号。CKPOLB 位决定时钟线的基本状态，若此位为高，则当时钟无效时，SCK 为低电平，若此位为低，则时钟无效时，SCK 为高电平。CKEG 位决定有效时钟边沿类型，取决于 CKPOLB 的设置。

Bit 3 **MLS**：SPI 数据移位顺序选择位
0：LSB 优先
1：MSB 优先
数据移位顺序选择位，用于选择数据传输时高位优先传输还是低位优先传输。此位设置为高时高位优先传输，为低时低位优先传输。

Bit 2 **CSEN**：SPI $\overline{\text{SCS}}$ 引脚控制位
0：除能
1：使能
CSEN 位用于 $\overline{\text{SCS}}$ 引脚的使能 / 除能控制。此位为低时，$\overline{\text{SCS}}$ 除能，则引脚处于浮空状态。此位为高时，SCS 作为从机选择引脚。

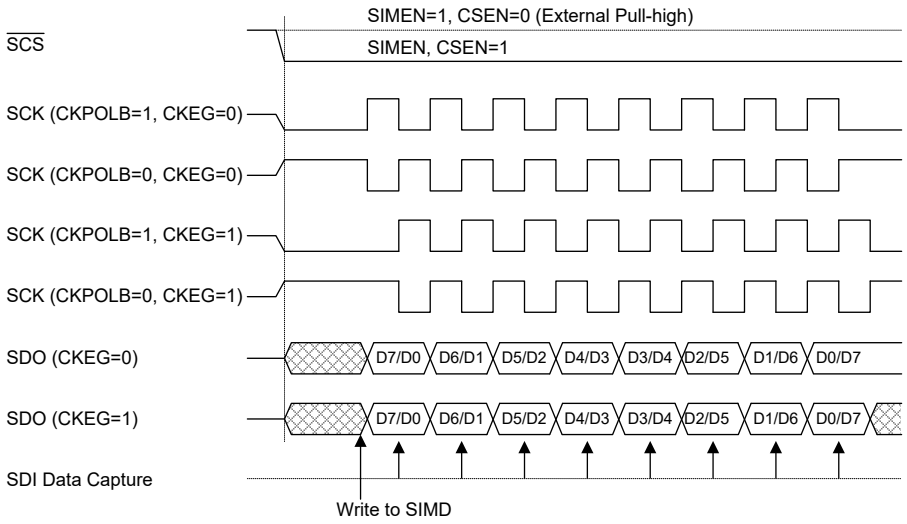
Bit 1 **WCOL**：SPI 写冲突标志位
0：无冲突
1：冲突
WCOL 标志位用于监测数据冲突的发生。此位为高时，表示在传输过程中有数据尝试被写入 SIMD 寄存器。若数据正在被传输时，此写操作无效。此位可通过应用程序清零。

Bit 0 **TRF**：SPI 发送 / 接收结束标志位
0：数据正在发送中
1：数据发送结束
TRF 位为发送 / 接收结束标志位，当 SPI 数据传输结束时，此位自动置为高，但须通过应用程序设置为“0”。此位也可用于产生中断。

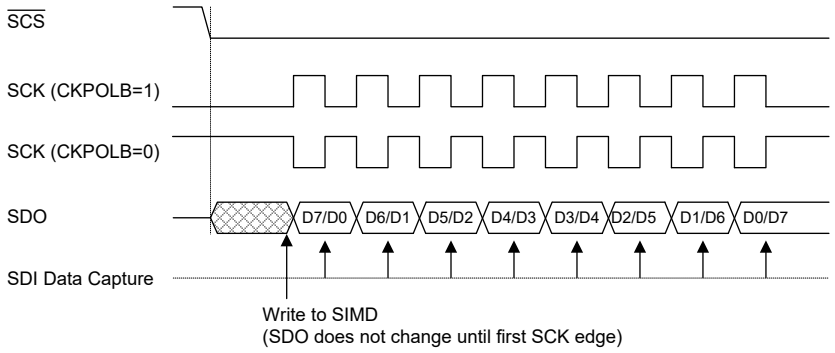
SPI 通信

将 SIMEN 设置为高，使能 SPI 功能之后，单片机处于主机模式，当数据写入到寄存器 SIMD 的同时传输 / 接收开始进行。数据传输完成时，TRF 位将自动被置位但清除只能通过应用程序完成。单片机处于从机模式时，收到主机发来的信号之后，会传输 SIMD 中的数据，而且在 SDI 引脚上的数据也会被移位到 SIMD 寄存器中。主机应在输出时钟信号之前先输出一个 $\overline{SCS}$ 信号以使能从机，从机的数据传输功能也应在与 SCK 信号相关的适当时候准备就绪，这由 CKPOLB 和 CKEG 位决定。所附时序图表明了在 CKPOLB 和 CKEG 位各种设置情况下从机数据与 SCK 信号的关系。

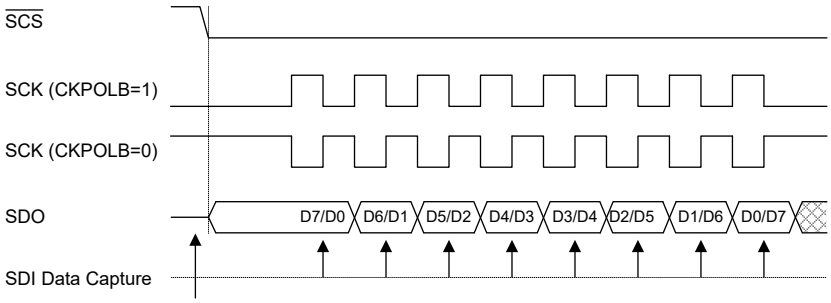
SPI 通信过程中，即使单片机进入空闲模式，若 SPI 时钟开启，SPI 功能仍将继续执行。



SPI 主机模式时序

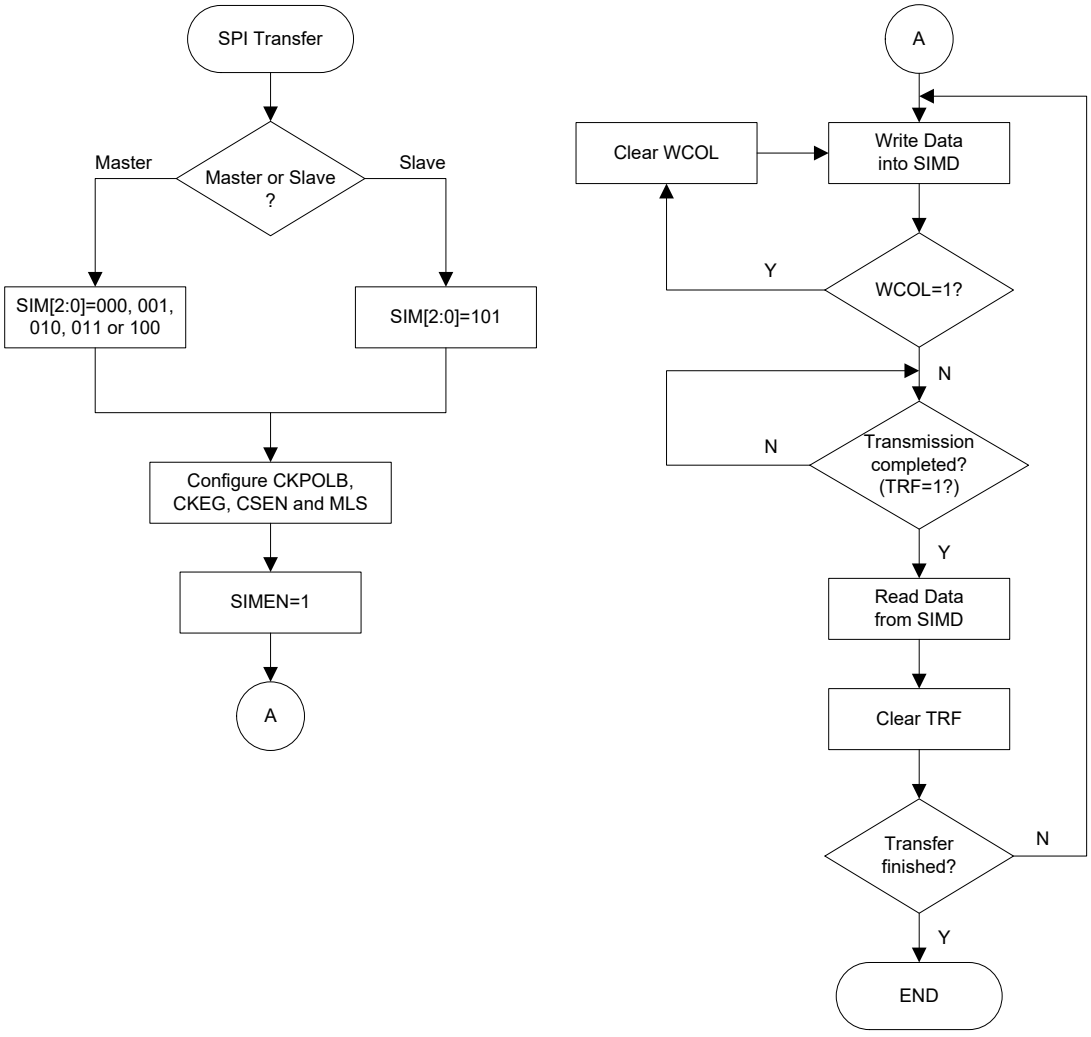


SPI 从机模式时序 – CKEG=0



Note: For SPI slave mode, if SIMEN=1 and CSEN=0, SPI is always enabled and ignores the SCS level.

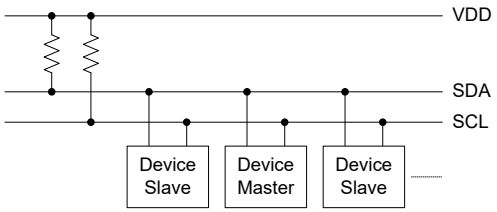
SPI 从机模式时序 – CKEG=1



SPI 传输控制流程图

I²C 接口

I²C 接口可以用于和传感器、EEPROM 存储器等外部硬件进行通信。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。

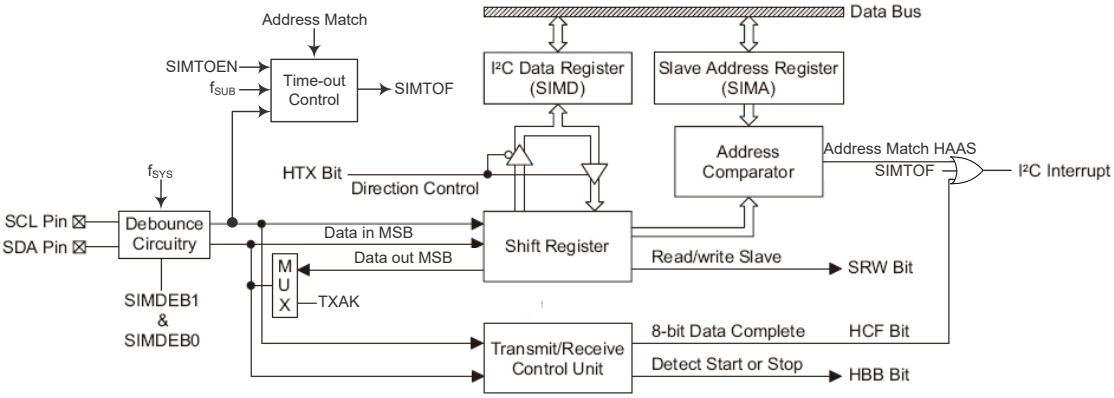


I²C 主从总线连接图

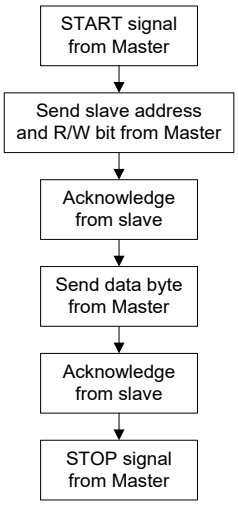
I²C 接口操作

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。当通过引脚共用功能选择寄存器选择 SCL/SDA 引脚功能，则此共用引脚对应的上拉电阻控制寄存器同样可用于使能或除能 SCL/SDA 引脚的上拉电阻连接。



I²C 方框图



SIMDEB1 和 SIMDEB0 位决定 I²C 接口的去抖时间。这个功能可以使用内部时钟在外部时钟上增加一个去抖间隔，减小时钟线上毛刺发生的可能性，以避免单片机发生误动作。如果选择了这个功能，去抖时间可以选择 2 个或 4 个系统时钟。为了达到需要的 I²C 数据传输速度，系统时钟 f_{SYS} 和 I²C 去抖时间之间存在一定的关系。I²C 标准模式或者快速模式下，用户需注意所选的系统时钟频率与标准匹配去抖时间的设置，其具体关系如下表所示。

I²C 去抖时间选择	I²C 标准模式（100kHz）	I²C 快速模式（400kHz）
无去抖时间	f_{SYS} > 2 MHz	f_{SYS} > 5 MHz
2 个系统时钟去抖时间	f_{SYS} > 4 MHz	f_{SYS} > 10 MHz
4 个系统时钟去抖时间	f_{SYS} > 8 MHz	f_{SYS} > 20 MHz

I²C 最小 f_{SYS} 频率要求

I²C 寄存器

I²C 总线有三个控制寄存器 SIMC0、SIMC1 和 SIMTOC，一个地址寄存器 SIMA 以及一个数据寄存器 SIMD。

寄存器名称	位							
	7	6	5	4	3	2	1	0
SIMC0	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	SIMICF
SIMC1	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
SIMA	SIMA6	SIMA5	SIMA4	SIMA3	SIMA2	SIMA1	SIMA0	D0
SIMD	D7	D6	D5	D4	D3	D2	D1	D0
SIMTOC	SIMTOEN	SIMTOF	SIMTOS5	SIMTOS4	SIMTOS3	SIMTOS2	SIMTOS1	SIMTOS0

I²C 寄存器列表

I²C 控制寄存器 – SIMC0, SIMC1, SIMTOC

单片机中有三个控制 I²C 接口功能的寄存器，SIMC0、SIMC1 和 SIMTOC。寄存器 SIMC0 用于控制使能 / 除能功能和设置数据传输的时钟频率。寄存器 SIMC1 包括多个用于指示 I²C 传输状态的相关标志位。SIMTOC 寄存器用于控制 I²C 超时功能，此寄存器在 I²C 超时控制一节介绍。

- **SIMC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	SIM2	SIM1	SIM0	—	SIMDEB1	SIMDEB0	SIMEN	SIMICF
R/W	R/W	R/W	R/W	—	R/W	R/W	R/W	R/W
POR	1	1	1	—	0	0	0	0

Bit 7~5 **SIM2~SIM0**：SIM 工作模式控制位
000：SPI 主机模式；SPI 时钟为 $f_{SYS}/4$
001：SPI 主机模式；SPI 时钟为 $f_{SYS}/16$
010：SPI 主机模式；SPI 时钟为 $f_{SYS}/64$
011：SPI 主机模式；SPI 时钟为 f_{SUB}
100：SPI 主机模式；SPI 时钟为 STM CCRP 匹配频率 /2
101：SPI 从机模式
110：I^2C 从机模式
111：未使用
这几位用于设置 SIM 功能的工作模式，用于选择 SPI 的主从模式和 SPI 的主机时钟频率及 I^2C 或 SPI 功能。SPI 时钟源可来自于系统时钟和 f_{SUB} 也可以选择来自 STM。若选择的是作为 SPI 从机，则其时钟源从外部主机而得。

Bit 4 未定义，读为“0”

Bit 3~2 **SIMDEB1~SIMDEB0**：I^2C 去抖时间选择位
00：无去抖时间
01：2 个系统时钟去抖时间
1x：4 个系统时钟去抖时间

Bit 1 **SIMEN**：SIM 接口功能控制位
0：除能
1：使能
此位为 SIM 接口的开 / 关控制位。此位为“0”时，SIM 接口除能，SDI、SDO、SCK 和 SCS 或 SDA 和 SCL 脚将失去 SPI 或 I^2C 功能，SIM 工作电流减小到最小值。此位为“1”时，SIM 接口使能。若 SIM 经由 SIM2~SIM0 位设置为工作在 SPI 接口，当 SIMEN 位由低到高转变时，SPI 控制寄存器中的设置不会发生变化，其首先应在应用程序中初始化。若 SIM 经由 SIM2~SIM0 位设置为工作在 I^2C 接口，当 SIMEN 位由低到高转变时，I^2C 控制寄存器中的设置，如 HTX 和 TXAK，将不会发生变化，其首先应在应用程序中初始化，此时相关 I^2C 标志，如 HCF、HAAS、HBB、SRW 和 RXAK，将被设置为其默认状态。

Bit 0 **SIMICF**：SIM SPI 传输未完成标志位
此位仅当 SIM 配置在 SPI 从机模式时有效。请参考 SPI 寄存器部分。

- **SIMC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	HCF	HAAS	HBB	HTX	TXAK	SRW	IAMWU	RXAK
R/W	R	R	R	R/W	R/W	R	R/W	R
POR	1	0	0	0	0	0	0	1

Bit 7 **HCF**：I^2C 总线数据传输结束标志位
0：数据正在传输中
1：8 位数据传输完成
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 **HAAS**：I^2C 地址匹配标志位
0：地址未匹配
1：地址匹配
此标志位用于决定从机地址是否与主机发送的地址相同。若地址匹配此位为高，否则此位为低。

Bit 5 **HBB**：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 总线空闲，该位变为低电平。

Bit 4 **HTX**：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 **TXAK**：I²C 总线发送应答标志位
0：从机发送应答标志
1：从机没有发送应答标志
从机接收完 8 位数据之后，该位将在第九个从机时钟时被传到总线上。如果从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 **SRW**：I²C 从机读 / 写位
0：从机应处于接收模式
1：从机应处于发送模式
SRW 位是从机读写位。决定主机是否希望传输数据或接收来自 I²C 总线的数据。当传输地址和从机的地址相同时，HAAS 位会被设置为高，从机将检测 SRW 位来决定进入发送模式还是接收模式。如果 SRW 位为高时，主机会请求从总线上读数据，此时从机处于发送模式。当 SRW 位为“0”时，主机往总线上写数据，从机处于接收模式以读取数据。

Bit 1 **IAMWU**：I²C 地址匹配唤醒控制位
0：除能
1：使能
此位设置为“1”则使能 I²C 地址匹配使系统从休眠或空闲模式中唤醒的功能。若进入休眠或空闲模式前 IAMWU 已经置高以使能 I²C 地址匹配唤醒功能，在系统唤醒后须软件清除此位以确保单片机正确地运行。

Bit 0 **RXAK**：I²C 总线接收应答标志位
0：从机接收到应答标志
1：从机未接收到应答标志
RXAK 位是接收应答标志位。如果 RXAK 位为“0”，即表示 8 位数据传输之后，从机在第九个时钟有接受到一个应答信号。如果从机处于发送状态，从机作为发送方会检查 RXAK 位来判断主机接收方是否愿意继续接收下一个字节。因此发送方会一直发送数据，直到 RXAK 为“1”时才停止发送数据。这时，发送方将释放 SDA 线，主机方可发出停止信号从而释放 I²C 总线。

I²C 数据寄存器 – SIMD

SIMD 用于存储发送和接收的数据。这个寄存器由 SPI 和 I²C 功能所共用。在单片机将数据写入到 I²C 总线之前，要传输的数据应先存在 SIMD 中。I²C 总线接收到数据之后，单片机就可以从 SIMD 数据寄存器中读取。所有通过 I²C 传输或接收的数据都必须通过 SIMD 实现。

- **SIMD 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	x	x	x	x	x	x	x	x

“x”为未知

Bit 7~0 **D7~D0**：SIM 数据寄存器位 bit 7 ~ bit 0

I²C 地址寄存器 – SIMA

SIMA 寄存器也在 SPI 接口功能中使用（用于 SPI 功能时其名称为 SIMC2）。SIMA 寄存器用于存放 7 位从机地址，寄存器 SIMA 中的 bit 7 ~ bit 1 是单片机的从机地址，bit 0 未定义。

如果接至 I²C 的主机发送出的地址和寄存器 SIMA 中存储的地址相符，那么就选中了这个从机。应注意的是寄存器 SIMA 和 SPI 接口使用的寄存器 SIMC2 位于同一个寄存器地址。

- **SIMA 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	SIMA6	SIMA5	SIMA4	SIMA3	SIMA2	SIMA1	SIMA0	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

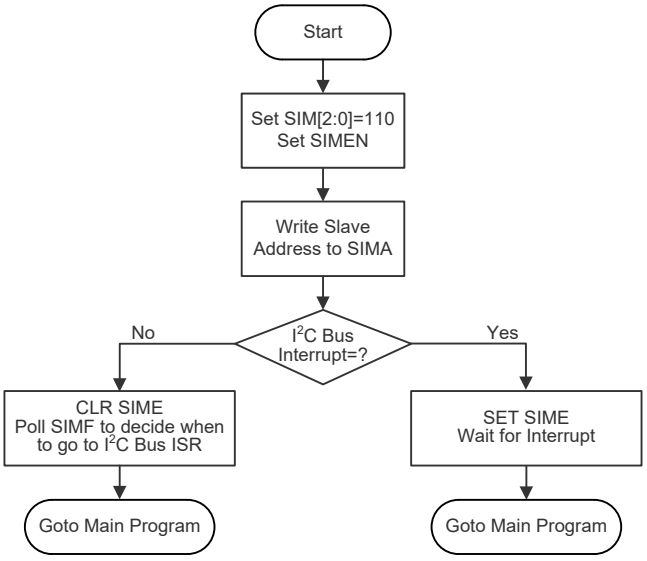
Bit 7~1 **SIMA6~SIMA0**：I²C 从机地址位
SIMA6~SIMA0 是从机地址 bit 6 ~ bit 0。

Bit 0 未定义位
此位可通过软件程序进行读写。

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，SIMC1 寄存器的 HAAS 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 HAAS 位和 SIMTOF 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，或是来自 I²C 超时。在数据传递中，要注意的是，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 SRW 位中。从机通过检测 SRW 位以确定自己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 SIMC0 寄存器中 SIM2~SIM0 位为“110”和 SIMEN 位为“1”，以使能 I²C 总线。
- 步骤 2
  向 I²C 总线地址寄存器 SIMA 写入从机地址。
- 步骤 3
  设置 SIME 位以使能 SIM 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 总线的主机产生，而不是由从机产生。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 HBB。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。

从机地址

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 SIMC1 寄存器的 SRW 位，从机随后发出一个低电平应答信号（即第 9 位）。当从机地址匹配时，从机会将状态标志位 HAAS 置位。

I²C 总线中断有三个中断源，当程序运行至中断服务子程序时，通过检测 HAAS 位和 SIMTOF 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，或是来自 I²C 超时。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 SIMD 寄存器，或是用于接收模式并从 SIMD 寄存器中读取空值以释放 SCL 线。

I²C 总线读 / 写信号

SIMC1 寄存器的 SRW 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。从机通过检测该位以确定自己是作为发送方还是接收方。当 SRW 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 SRW 清“0”，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。

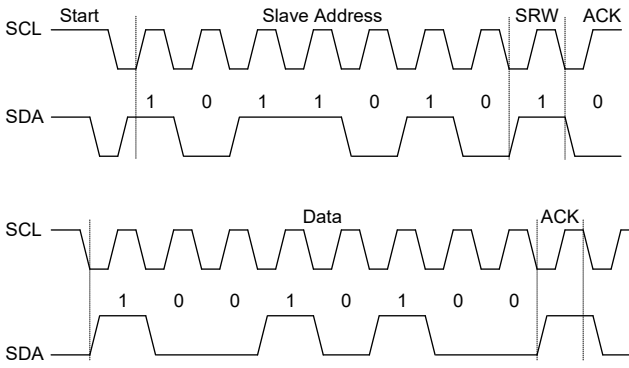
I²C 总线从机地址应答信号

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 HAAS 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 SRW 位，以确定自己是作为发送方还是作为接收方。如果 SRW 位为高，从机须设置成发送方，这样会置位 SIMC1 寄存器的 HTX 位。如果 SRW 位为低，从机须设置成接收方，这样会清零 SIMC1 寄存器的 HTX 位。

I²C 总线数据和应答信号

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。如果从机发送方没接收到来自主机接收方的应答信号，发送方将释放 SDA 线，此时主机方可发出 STOP 信号以释放 I²C 总线。所传送的数据存储在 SIMD 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 SIMD 寄存器中；如果设置成接收方，从机必须从 SIMD 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(TXAK)。被设为发送方的从机将检测寄存器 SIMC1 中的 RXAK 位以判断是否传输下一个字节的数据，如果从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。

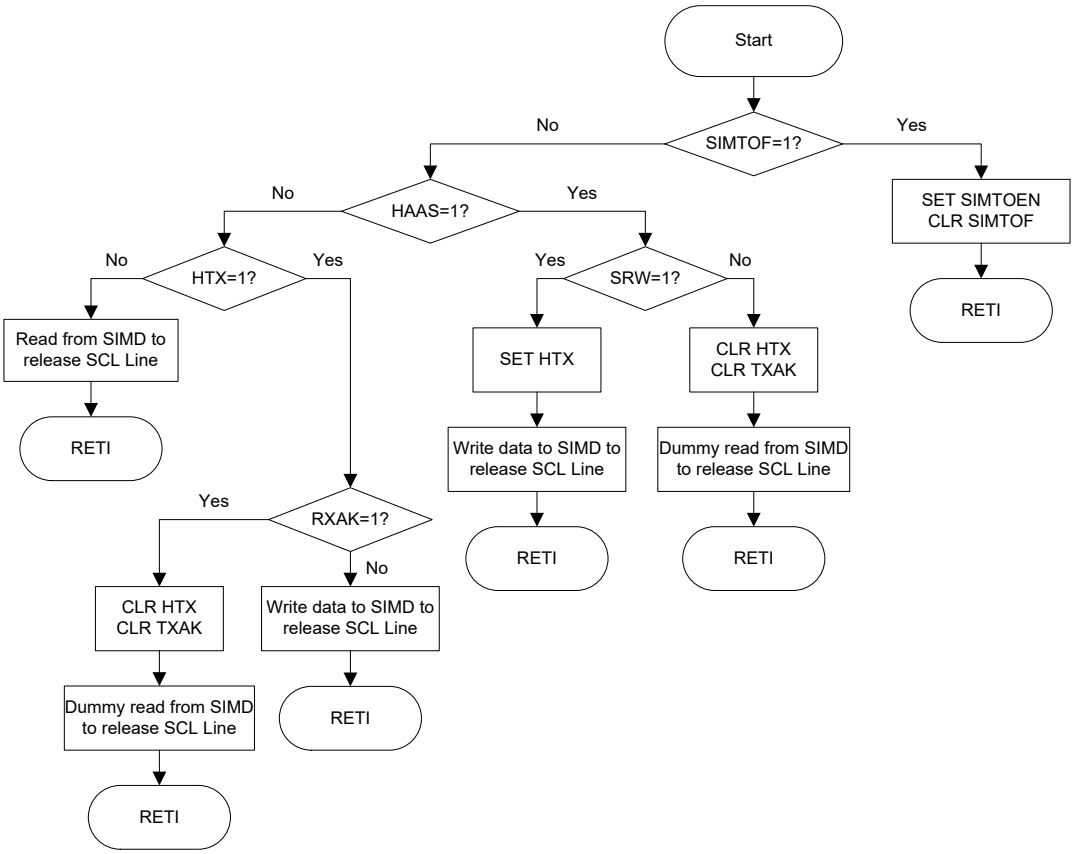


S=Start (1 bit)
SA=Slave Address (7 bits)
SR=SRW bit (1 bit)
M=Slave device send acknowledge bit (1 bit)
D=Data (8 bits)
A=ACK (RXAK bit for transmitter, TXAK bit for receiver, 1 bit)
P=Stop (1 bit)

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 SIMD 寄存器；若设置为接收模式，需立即从 SIMD 寄存器中虚读数据以释放 SCL 线。

I²C 通信时序图



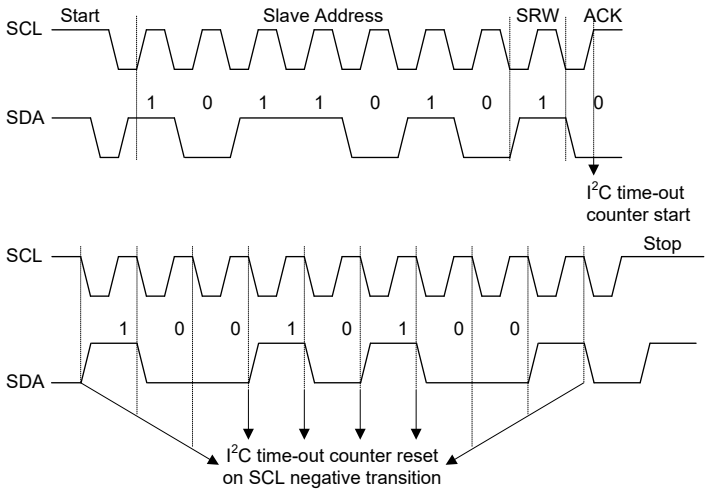
I²C 总线 ISR 流程图

I²C 超时

超时功能可减少 I²C 接收错误的时钟源而引起的锁死问题。如果连接到 I²C 总线的时钟源经过一段时间还未接收到，则在一定的超时周期后，I²C 电路和 SIMC1 寄存器将复位且 SIMTOC 寄存器中的 SIMTOF 位将被置位为“1”。I²C 电路的超时控制功能及溢出时间选择可通过 SIMTOC 寄存器进行开启与设置。

I²C 超时操作

超时计数器在 I²C 总线“START”和“地址匹配”条件下开始计数，且在 SCL 下降沿清零。在下一个 SCL 下降沿到来之前，如果超时时间大于 SIMTOC 寄存器指定的超时周期，则超时发生。I²C“STOP”条件发生时超时功能终止。



I²C 超时时序图

当 I²C 超时计数器溢出时，计数器将停止计数，SIMTOEN 位被清零，且 SIMTOF 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
SIMD，SIMA，SIMC0	保持不变
SIMC1	复位至 POR 状态

超时发生后的 I²C 寄存器

SIMTOF 标志位由应用程序清零。共有 64 个超时周期，可通过 SIMTOC 寄存器的 SIMTOSn 位进行选择。超时周期可通过公式计算：$((1\sim64) \times (32/f_{SUB}))$。由此可得超时周期范围为 1ms~64ms。

- **SIMTOC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	SIMTOEN	SIMTOF	SIMTOS5	SIMTOS4	SIMTOS3	SIMTOS2	SIMTOS1	SIMTOS0
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 **SIMTOEN**：I²C 超时控制位
0：除能
1：使能

Bit 6 **SIMTOF**：I²C 超时标志位
0：超时未发生
1：超时发生

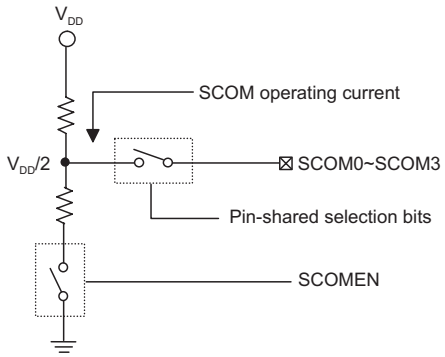
Bit 5~0 **SIMTOS5~SIMTOS0**：I²C 超时时间选择位
I²C 超时时钟源是 $f_{SUB}/32$
I²C 超时时间计算方法：$([SIMTOS[5:0]+1) \times (32/f_{SUB})$

SCOM LCD 驱动器

该系列单片机具有驱动外部 LCD 面板的能力。LCD 驱动的 COM 脚 SCOM0~SCOM3 与 I/O 口共用。LCD 的 COM 信号由软件编程实现。

LCD 操作

单片机通过设置输入 / 输出口作为 COM 引脚以驱动外部的液晶面板。LCD 驱动功能是由 SCOMC 寄存器来控制，该寄存器除了可设置 LCD 的开启和关闭外还可控制输出偏压值等功能，使得 COM 口输出 $V_{DD}/2$ 的电压，从而实现 1/2 bias LCD 的显示。



LCD COM 偏压

SCOMC 寄存器中的 SCOMEN 位是 LCD 驱动的总控制位，LCD 的 SCOMn 引脚可通过正确配置相关的引脚共用功能选择寄存器用于 LCD 驱动。需注意的是，端口控制寄存器不需要预先设置为输出来使能 LCD 驱动操作。

LCD 偏置电流控制

LCD 驱动器可以提供多种驱动电流选择以适应不同 LCD 面板的需求。通过设定 SCOMC 寄存器中 ISEL0 位和 ISEL1 位可以配置不同的偏压电阻以产生不同的驱动电流。

SCOMC 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	ISEL1	ISEL0	SCOMEN	—	—	—	—
R/W	—	R/W	R/W	R/W	—	—	—	—
POR	—	0	0	0	—	—	—	—

Bit 7　未定义，读为“0”

Bit 6~5　**ISEL1~ISEL0**：选择 R 型 LCD 的典型偏压电阻及偏置电流（V_{DD}=5V）
00：2×100kΩ (1/2 Bias), I_{BIAS} = 25μA
01：2×50kΩ (1/2 Bias), I_{BIAS} = 50μA
10：2×25kΩ (1/2 Bias), I_{BIAS} = 100μA
11：2×12.5kΩ (1/2 Bias), I_{BIAS} = 200μA

Bit 4　**SCOMEN**：LCD 模块控制位
0：除能
1：使能
当 SCOMEN 位为高时，所选择的电阻将被开启以产生 1/2 V_{DD} 的偏压。

Bit 3~0　未定义，读为“0”

中断

中断是单片机一个重要功能。当外部事件或内部功能如定时器模块或 A/D 转换器有效，并且产生中断时，系统会暂时中止当前的程序而转到执行相对应的中断服务程序。此系列单片机提供多个外部中断和内部中断功能，外部中断由 INT0~INT2 引脚动作产生，而内部中断由各种内部功能，如定时器模块、串行接口模块、欠压保护功能、过压保护功能、过流保护功能、时基、LVD、EEPROM 和 A/D 转换器等产生。

中断寄存器

中断控制基本上是在一定单片机条件发生时设置请求标志位，应用程序中中断使能位的设置是通过位于专用数据存储器中的一系列寄存器控制的。寄存器总的分为三类。第一类是 INTC0~INTC3 寄存器，用于设置基本的中断；第二类是 MFI0~MFI1 寄存器，用于设置多功能中断；最后一种有 INTEG 寄存器，用于设置外部中断边沿触发类型。

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

功能	使能位	请求标志	注释
总中断	EMI	—	—
INTn 引脚	INTnE	INTnF	n=0~2
过流保护	OCPnE	OCPnF	n=0 或 1
欠压保护	UVPnE	UVPnF	n=0 或 1
过压保护	OVPnE	OVPnF	n=0 或 1
多功能	MFnE	MFnF	n=0 或 1
A/D 转换器	ADE	ADF	—
时基	TBnE	TBnF	n=0 或 1
低电压检测	LVE	LVF	—
EEPROM	DEE	DEF	—
串行接口模块	SIME	SIMF	—
标准型 TM	STMPE	STMPF	—
	STMAE	STMAF	
周期型 TM	PTMPE	PTMPF	—
	PTMAE	PTMAF	

中断寄存器位命名模式

寄存器名称	位							
	7	6	5	4	3	2	1	0
INTEG	—	—	INT2S1	INT2S0	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	OVP0F	OCP1F	OCP0F	OVP0E	OCP1E	OCP0E	EMI
INTC1	INT0F	UVP1F	UVP0F	OVP1F	INT0E	UVP1E	UVP0E	OVP1E
INTC2	MF1F	MF0F	INT2F	INT1F	MF1E	MF0E	INT2E	INT1E
INTC3	SIMF	TB1F	TB0F	LVF	SIME	TB1E	TB0E	LVE
MFI0	—	DEF	STMAF	STMPF	—	DEE	STMAE	STMPE
MFI1	—	ADF	PTMAF	PTMPF	—	ADE	PTMAE	PTMPE

中断寄存器列表

INTEG 寄存器

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

Bit 7~6　未定义，读为“0”

Bit 5~4　**INT2S1~INT2S0**：INT2 脚中断边沿控制位
00：除能
01：上升沿
10：下降沿
11：双沿

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

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

INTC0 寄存器

Bit	7	6	5	4	3	2	1	0
Name	—	OVP0F	OCP1F	OCP0F	OVP0E	OCP1E	OCP0E	EMI
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7　未定义，读为“0”

Bit 6　**OVP0F**：过压保护 0 中断请求标志位
0：无请求
1：中断请求

Bit 5　**OCP1F**：过电流保护 1 中断请求标志位
0：无请求
1：中断请求

Bit 4　**OCP0F**：过电流保护 0 中断请求标志位
0：无请求
1：中断请求

Bit 3　**OVP0E**：过压保护 0 中断控制位
0：除能
1：使能

Bit 2　**OCP1E**：过电流保护 1 中断控制位
0：除能
1：使能

Bit 1　**OCP0E**：过电流保护 0 中断控制位
0：除能
1：使能

Bit 0　**EMI**：总中断控制位
0：除能
1：使能

INTC1 寄存器

Bit	7	6	5	4	3	2	1	0
Name	INT0F	UVP1F	UVP0F	OVP1F	INT0E	UVP1E	UVP0E	OVP1E
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **INT0F**：INT0 中断请求标志位
0：无请求
1：中断请求

Bit 6 **UVP1F**：欠压保护 1 中断请求标志位
0：无请求
1：中断请求

Bit 5 **UVP0F**：欠压保护 0 中断请求标志位
0：无请求
1：中断请求

Bit 4 **OVP1F**：过压保护 1 中断请求标志位
0：无请求
1：中断请求

Bit 3 **INT0E**：INT0 中断控制位
0：除能
1：使能

Bit 2 **UVP1E**：欠压保护 1 中断控制位
0：除能
1：使能

Bit 1 **UVP0E**：欠压保护 0 中断控制位
0：除能
1：使能

Bit 0 **OVP1E**：过压保护 1 中断控制位
0：除能
1：使能

INTC2 寄存器

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

Bit 7 **MF1F**：多功能中断 1 中断请求标志位
0：无请求
1：中断请求

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

Bit 5 **INT2F**：INT2 中断请求标志位
0：无请求
1：中断请求

Bit 4 **INT1F**：INT1 中断请求标志位
0：无请求
1：中断请求

Bit 3 **MF1E**：多功能中断 1 中断控制位
0：除能
1：使能

Bit 2 **MF0E**：多功能中断 0 中断控制位
0：除能
1：使能

Bit 1 **INT2E**：INT2 中断控制位
0：除能
1：使能

Bit 0 **INT1E**：INT1 中断控制位
0：除能
1：使能

INTC3 寄存器

Bit	7	6	5	4	3	2	1	0
Name	SIMF	TB1F	TB0F	LVF	SIME	TB1E	TB0E	LVE
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 **SIMF**：SIM 中断请求标志位
0：无请求
1：中断请求

Bit 6 **TB1F**：时基 1 中断请求标志位
0：无请求
1：中断请求

Bit 5 **TB0F**：时基 0 中断请求标志位
0：无请求
1：中断请求

Bit 4 **LVF**：LVD 中断请求标志位
0：无请求
1：中断请求

Bit 3 **SIME**：SIM 中断控制位
0：除能
1：使能

Bit 2 **TB1E**：时基 1 中断控制位
0：除能
1：使能

Bit 1 **TB0E**：时基 0 中断控制位
0：除能
1：使能

Bit 0 **LVE**：LVD 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	—	DEF	STMAF	STMPF	—	DEE	STMAE	STMPE
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

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

MFI1 寄存器

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

Bit 7　未定义，读为“0”
Bit 6　**ADF**：A/D 转换器中断请求标志位
　　0：无请求
　　1：中断请求
Bit 5　**PTMAF**：PTM 比较器 A 匹配中断请求标志位
　　0：无请求
　　1：中断请求
Bit 4　**PTMPF**：PTM 比较器 P 匹配中断请求标志位
　　0：无请求
　　1：中断请求
Bit 3　未定义，读为“0”
Bit 2　**ADE**：A/D 转换器中断控制位
　　0：除能
　　1：使能
Bit 1　**PTMAE**：PTM 比较器 A 匹配中断控制位
　　0：除能
　　1：使能
Bit 0　**PTMPE**：PTM 比较器 P 匹配中断控制位
　　0：除能
　　1：使能

中断操作

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

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

各个中断使能位以及相应的请求标志位，以优先级的次序显示在下图。一些中断源有自己的向量，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。

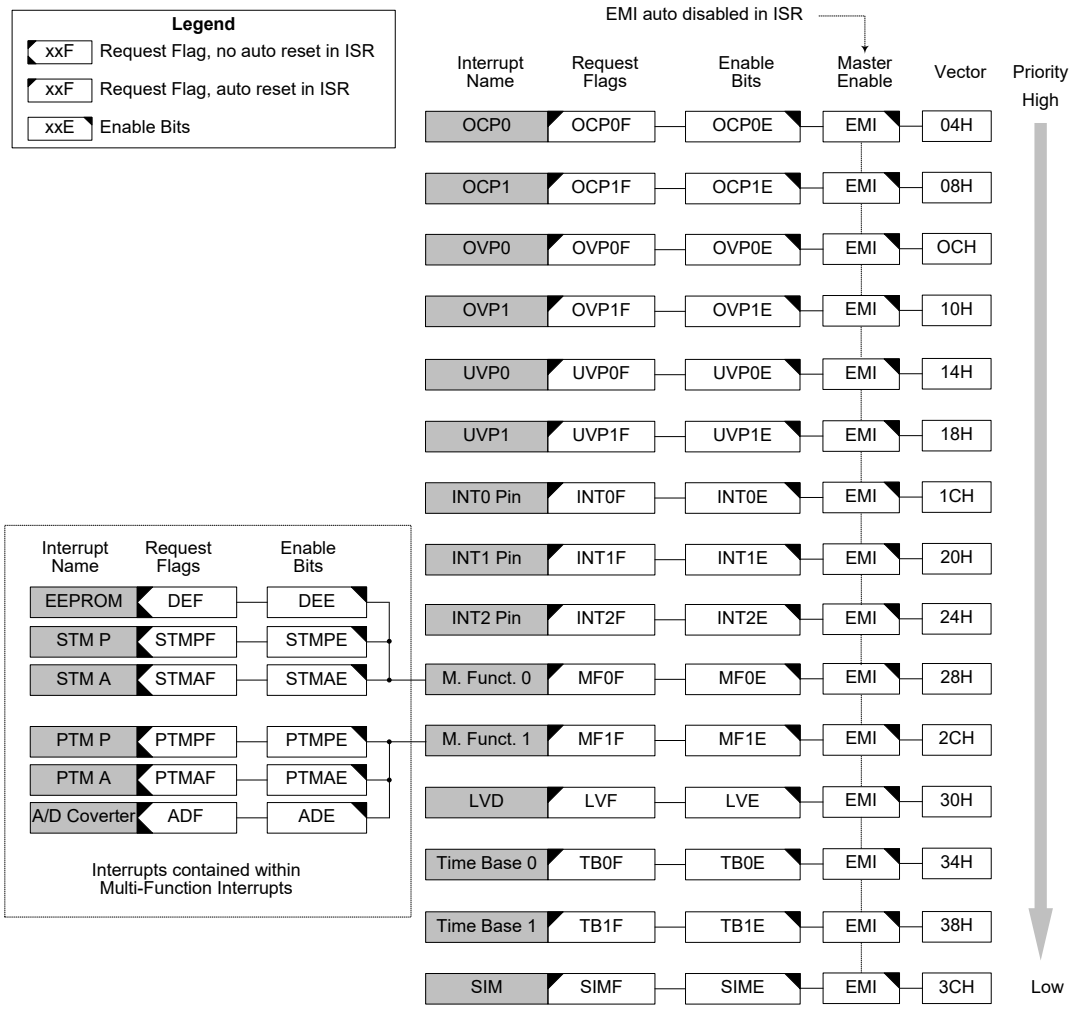
外部中断

通过 INT0~INT2 引脚上的信号变化可控制外部中断。当触发沿选择位设置好触发类型，INT0~INT2 引脚的状态发生变化，外部中断请求标志 INT0F~INT2F 被置位时外部中断请求产生。若要跳转到相应中断向量地址，总中断控制位 EMI 和相应中断使能位 INT0E~INT2E 需先被置位。此外，必须使用 INTEG 寄存器使能外部中断功能并选择触发沿类型。外部中断引脚和普通 I/O 口共用，如果通过相应的引脚共用功能选择寄存器选择外部中断引脚功能且相应寄存器中的中断使能位被置位，此引脚将被作为外部中断脚使用。需注意的是此时该引脚必须通过设置相应的 I/O 端口控制寄存器，将该引脚输入 / 输出类型设置为输入状态。

当中断使能，堆栈未满并且外部中断脚状态改变，将调用外部中断向量子程序。当响应外部中断服务子程序时，中断请求标志位 INT0F~INT2F 会自动复位且 EMI 位会被清零以除能其它中断。注意，即使此引脚被用作外部中断输入，其通过寄存器选择的上拉电阻仍保持有效。寄存器 INTEG 被用来选择有效的边沿类型，来触发外部中断。可以选择上升沿还是下降沿或双沿触发都产生外部中断。注意 INTEG 也可以用来除能外部中断功能。

过电流保护中断

当检测到过电流情况时，过电流保护中断请求标志位 OCP0F 或 OCP1F 被置位，OCP0 或 OCP1 中断请求发生。当总中断使能位 EMI 和 OCP0 或 OCP1 中断使能位 OCP0E 或 OCP1E 被置位，允许程序跳转到相应的中断向量地址。当中断使能，堆栈未满且过电流情况发生时，将调用 OCP0 或 OCP1 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 OCP0F 或 OCP1F 会自动清零。EMI 位也会被清零以除能其它中断。



中断结构

过压保护中断

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

欠压保护中断

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

多功能中断

此系列单片机中有两个多功能中断，与其它中断不同，它们没有独立的中断来源，而是由其它现有的中断源构成，即 TM 中断、EEPROM 中断和 A/D 转换器中断。

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

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

定时器模块中断

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

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

EEPROM 中断

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

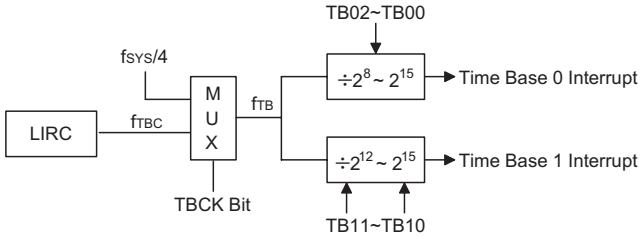
A/D 转换器中断

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

时基中断

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

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



时基中断

TBC 寄存器

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

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

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

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

Bit 3 未定义，读为“0”

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

LVD 中断

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

串行接口中断

串行接口模块中断，即 SIM 中断。当一个字节数据已由 SIM 接口接收或发送完，或 I^2C 从机地址匹配，或 I^2C 超时，中断请求标志 SIMF 被置位，SIM 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 和串行接口中断使能位 SIME 需先被置位。当中断使能，堆栈未满且以上任一种情况发生时，可跳转至相关多功能中断向量子程序中执行。当响应中断服务子程序时，串行接口中断标志位 SIMF 会自动复位且 EMI 将被自动清零以除能其它中断。

中断唤醒功能

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

编程注意事项

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

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

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

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

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

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

低电压检测 – LVD

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

LVD 寄存器

低电压检测功能由 LVDC 寄存器控制。VLVD2~VLVD0 位用于选择 5 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

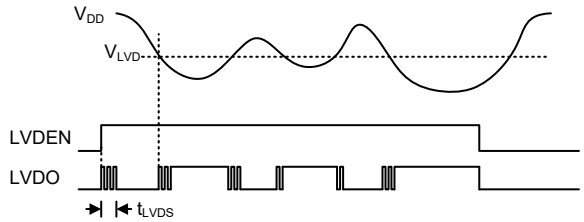
Bit 4　**LVDEN**：低电压检测功能控制位
0：除能
1：使能

Bit 3　未定义，读为“0”

Bit 2~0　**VLVD2~VLVD0**：LVD 电压选择
000：未定义
001：未定义
010：未定义
011：2.7V
100：3.0V
101：3.3V
110：3.6V
111：4.0V

LVD 操作

通过比较电源电压 V_{DD} 与通过 LVDC 寄存器选择的预置电压值的结果，低电压检测功能工作。其设置的范围为 2.7V~4.0V。当电源电压 V_{DD} 低于预置电压值时，LVDO 位被置为高，表明低电压产生。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 LVF 将被置位，中断产生，单片机将从休眠或空闲模式中被唤醒。若不要求低电压检测的唤醒功能使能，在单片机进入休眠或空闲模式前应将 LVF 标志置为高。

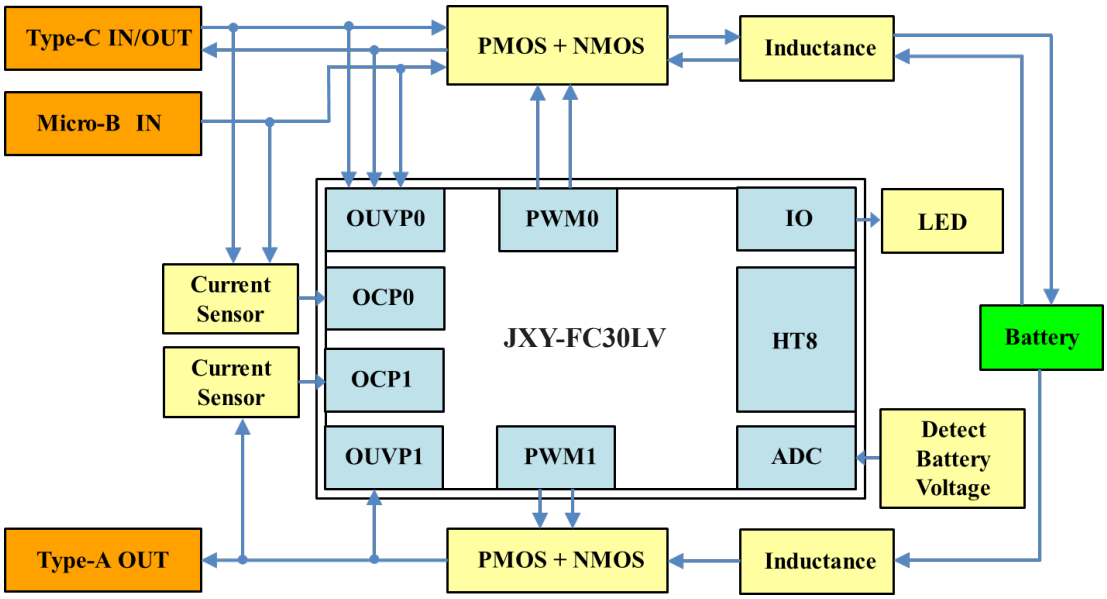
当 LVD 功能使能后，建议在使能 LVD 中断前先将 LVD 中断请求标志位清零以避免错误发生。

应用说明 – Type-C 移动电源

简介

移动电源产品主要分成四项功能，充电、放电、保护、电量显示，对移动电源内部电池充电，如 Micro-B、Type-C 端口；提供电源给外部装置，如手机 / 平板等设备；充 / 放电过程中的过电压、欠电压、过电流保护；LED 电量显示功能等。专用移动电源管理 MCU JXY-FC30LV 提供了上述所须功能的控制信号，现介绍如下。

硬件方框图



JXY-FC30LV Type-C 移动电源方框图

1. Type-C IN/OUT、Micro-B USB IN/OUT：系统皆为 5V 电压系统，而 Battery 端为 3.7V 电压系统，须由 MCU 输出互补式 PWM 控制 PMOS + NMOS 与 Inductance 达到充电降压及放电升压的电压转换电路管理。
2. 内建 OUVP 电路：在正常情况使用下 OUVP 引脚用来当 A/D Converter 检测输入电压进行充电降压控制，当异常情况判断到输入电压过高或过低时能实时停止 PWM 输出，保护电池。
3. 内建 OCP 电路：在正常情况使用下 OCP 引脚将 Current Sensor 点放大后通过 A/D Converter 读取电流值，当异常情况判断到输出电流过高时能实时停止 PWM 输出，保护电池。
4. 检测电池电压：在充电过程中，会去检测目前电池电压来决定选择涓流 / 恒流 / 恒压充电模式，在放电过程中检测电池电量，若过低则停止放电。
5. LED 和 Button：使用 Button 或充 / 放电过程中显示目前电量，使用四个 LED 显示 25/50/75/100% 电量。

功能说明

移动电源充电工作原理

USB 其输入电源为 5V，移动电源所采用的锂电池为 3.7V，外部输入电源对电池充电需要加上一个 Buck 降压转换器，通过 MCU 内部 PWM 功能控制 Buck 降压转换器对电池进行充电，MCU 内部设计的 OUVP、OCP 能针对充电过程进行过压 / 欠压、过电流保护，整个充电控制上主要分三种模式：涓流、恒流、恒压等充电模式，涓流、恒流、恒压的充电电压及电流根据用户的设计而有所不同。下列对各项模式进行说明。

涓流充电模式：对于已经完全放电的电池，电池电压小于 3V 时，第一个阶段会以恒定微小电流进行预充电，典型的涓流充电以 0.1C 电流对电池进行充电。

恒流充电模式：电池电压大于 3V 时，第二个阶段以恒定大电流进行充电，典型的恒流充电以 1C 电流对电池进行充电。

恒压充电模式：电池电压大于 4.1V，第三个阶段以恒定电压方式进行充电，随着恒压充电时间增加，充电电流会慢慢减少，典型的恒压充电判断电流低于 0.1C 时结束充电。

注：以一个 1000mAh 容量锂电池为例，0.1C 代表以 100mA 电流进行充电，1C 代表以 1000mA 进行充电。

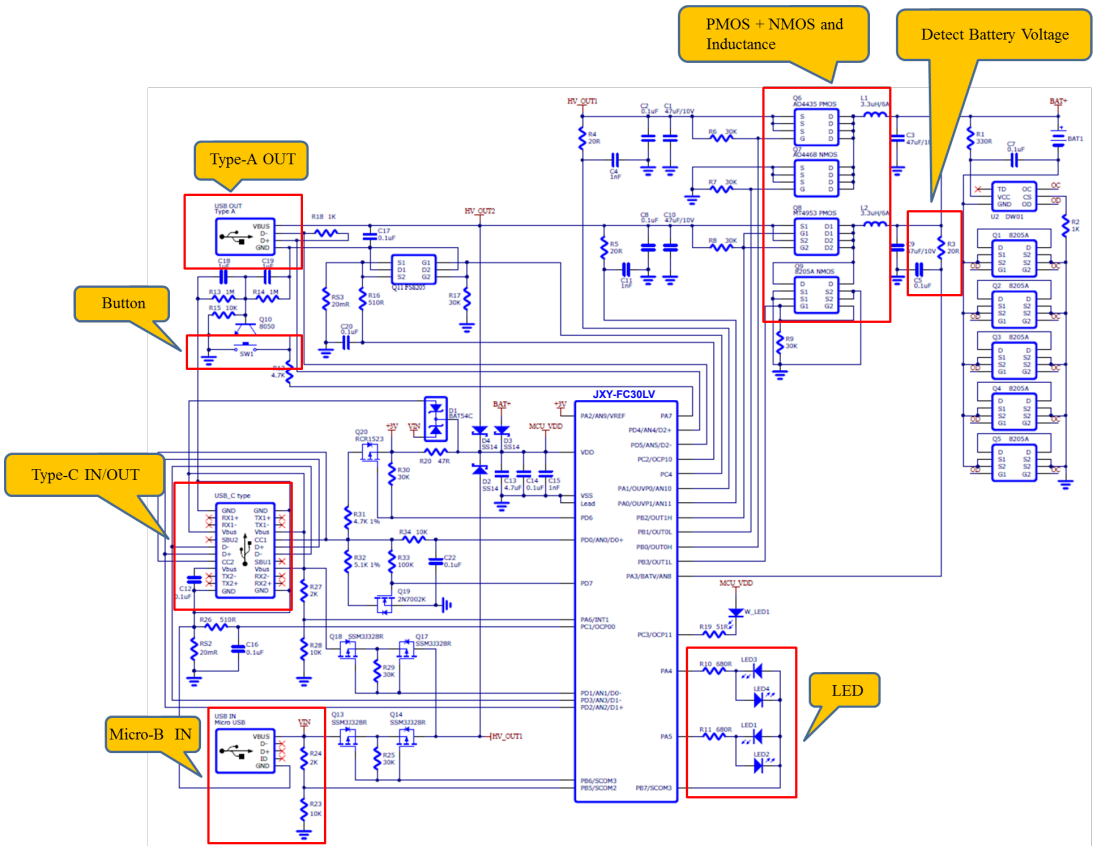
移动电源放电工作原理

本系统具备一个侦测电路能判断有无外部装置 (如手机) 接入，放电输出电源为 5V，锂电池为 3.7V，在电池放电方面，需要加上一个 Boost 升压转换器，通过 IC 内部 PWM 功能控制 Boost 升压转换器进行放电，IC 内部设计的 OCP 能针对放电过程进行过电流保护，整个放电控制分成两种模式，恒压、恒流模式。

恒压放电模式：通过 PWM 控制升压系统，使其维持 5V 输出，并做电流检测，判断目前输出电流的状态，输出电流过小则判断手机已经拔除，进行关机，输出电流过大进入到恒流模式，使输出不会发生过载的状况。

恒流放电模式：当输出电流过大时，会控制输出恒定电流，使输出不会发生过载的状况，并持续做电流检测，正常情况下，手机会判断此时电流已被固定，不会再加重负载，但如果手机处于异常状况，则可能会再继续加重负载，此时系统判断到输出电流过大，进行保护处理。

硬件电路图



JXY-FC30LV Type-C 移动电源电路

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。在该系列单片机中，提供了丰富且灵活的指令，共超过六十条，程序设计者可以事半功倍地实现他们的应用。

为了更加容易理解各种各样的指令码，接下来按功能分组介绍它们。

指令周期

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

数据的传送

单片机程序中数据传送是使用最为频繁的操作之一，使用几种 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，在该系列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 全都包含在该系列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该系列单片机的特性之一。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，该系列单片机允许在程序存储器中建立一个表格作为数据可直接存储的区域，只需要一组简易的指令即可对数据进行查表。

其它运算

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

指令集概要

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

惯例

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

助记符	说明	指令周期	影响标志位
算术运算			
ADD A,[m]	ACC 与数据存储器相加，结果放入 ACC	1	Z，C，AC，OV，SC
ADDM A,[m]	ACC 与数据存储器相加，结果放入数据存储器	1 注	Z，C，AC，OV，SC
ADD A，x	ACC 与立即数相加，结果放入 ACC	1	Z，C，AC，OV，SC
ADC A,[m]	ACC 与数据存储器、进位标志相加，结果放入 ACC	1	Z，C，AC，OV，SC
ADCM A,[m]	ACC 与数据存储器、进位标志相加，结果放入数据存储器	1 注	Z，C，AC，OV，SC
SUB A，x	ACC 与立即数相减，结果放入 ACC	1	Z，C，AC，OV，SC，CZ
SUB A,[m]	ACC 与数据存储器相减，结果放入 ACC	1	Z，C，AC，OV，SC，CZ
SUBM A,[m]	ACC 与数据存储器相减，结果放入数据存储器	1 注	Z，C，AC，OV，SC，CZ
SBC A,[m]	ACC 与数据存储器、进位标志的反相减，结果放入 ACC	1	Z，C，AC，OV，SC，CZ
SBCM A,[m]	ACC 与数据存储器、进位标志相减，结果放入数据存储器	1 注	Z，C，AC，OV，SC，CZ
SBC A，x	ACC 与立即数、进位标志相减，结果放入 ACC	1	Z，C，AC，OV，SC，CZ
DAA [m]	将加法运算中放入 ACC 的值调整为十进制数，并将结果放入数据存储器	1 注	C
逻辑运算			
AND A,[m]	ACC 与数据存储器做“与”运算，结果放入 ACC	1	Z
OR A,[m]	ACC 与数据存储器做“或”运算，结果放入 ACC	1	Z
XOR A,[m]	ACC 与数据存储器做“异或”运算，结果放入 ACC	1	Z
ANDM A,[m]	ACC 与数据存储器做“与”运算，结果放入数据存储器	1 注	Z
ORM A,[m]	ACC 与数据存储器做“或”运算，结果放入数据存储器	1 注	Z
XORM A,[m]	ACC 与数据存储器做“异或”运算，结果放入数据存储器	1 注	Z
AND A，x	ACC 与立即数做“与”运算，结果放入 ACC	1	Z
OR A，x	ACC 与立即数做“或”运算，结果放入 ACC	1	Z
XOR A，x	ACC 与立即数做“异或”运算，结果放入 ACC	1	Z
CPL [m]	对数据存储器取反，结果放入数据存储器	1 注	Z

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

助记符	说明	指令周期	影响标志位
ITABRDL [m]	读表指针 TBLP 自加，读取最后页的 ROM 内容，并送至数据存储器和 TBLH	2 注	无
其它指令			
NOP	空指令	1	无
CLR [m]	清除数据存储器	1 注	无
SET [m]	置位数据存储器	1 注	无
CLR WDT	清除看门狗定时器	1	TO，PDF
SWAP [m]	交换数据存储器的高低字节，结果放入数据存储器	1 注	无
SWAPA [m]	交换数据存储器的高低字节，结果放入 ACC	1	无
HALT	进入暂停模式	1	TO，PDF

注：1. 对跳转指令而言，如果比较的结果牵涉到跳转即需多达 3 个周期，如果没有发生跳转，则只需一个周期。

2. 任何指令若要改变 PCL 的内容将需要 2 个周期来执行。

3. 对于“CLR WDT”指令而言，TO 和 PDF 标志位也许会受执行结果影响，“CLR WDT”被执行后，TO 和 PDF 标志位会被清除，否则 TO 和 PDF 标志位保持不变。

扩展指令集

扩展指令用来提供更大范围的数据存储器寻址。当被存取的数据存储器位于 Sector 0 之外的任何数据存储器 Sector，扩展指令可直接存取数据存储器而无需使用间接寻址，此举也提高了 CPU 韧体性能。

助记符	说明	指令周期	影响标志位
算术运算			
LADD A,[m]	ACC 与数据存储器相加，结果放入 ACC	2	Z，C，AC，OV，SC
LADDM A,[m]	ACC 与数据存储器相加，结果放入数据存储器	2注	Z，C，AC，OV，SC
LADC A,[m]	ACC 与数据存储器、进位标志相加，结果放入 ACC	2	Z，C，AC，OV，SC
LADCM A,[m]	ACC 与数据存储器、进位标志相加，结果放入数据存储器	2注	Z，C，AC，OV，SC
LSUB A,[m]	ACC 与数据存储器相减，结果放入 ACC	2	Z，C，AC，OV，SC，CZ
LSUBM A,[m]	ACC 与数据存储器相减，结果放入数据存储器	2注	Z，C，AC，OV，SC，CZ
LSBC A,[m]	ACC 与数据存储器、进位标志的反相减，结果放入 ACC	2	Z，C，AC，OV，SC，CZ
LSBCM A,[m]	ACC 与数据存储器、进位标志相减，结果放入数据存储器	2注	Z，C，AC，OV，SC，CZ
LDAA [m]	将加法运算中放入 ACC 的值调整为十进制数，并将结果放入数据存储器	2注	C
逻辑运算			
LAND A,[m]	ACC 与数据存储器做“与”运算，结果放入 ACC	2	Z
LOR A,[m]	ACC 与数据存储器做“或”运算，结果放入 ACC	2	Z
LXOR A,[m]	ACC 与数据存储器做“异或”运算，结果放入 ACC	2	Z
LANDM A,[m]	ACC 与数据存储器做“与”运算，结果放入数据存储器	2注	Z
LORM A,[m]	ACC 与数据存储器做“或”运算，结果放入数据存储器	2注	Z
LXORM A,[m]	ACC 与数据存储器做“异或”运算，结果放入数据存储器	2注	Z
LCPL [m]	对数据存储器取反，结果放入数据存储器	2注	Z
LCPLA [m]	对数据存储器取反，结果放入 ACC	2	Z
递增和递减			
LINCA [m]	递增数据存储器，结果放入 ACC	2	Z
LINC [m]	递增数据存储器，结果放入数据存储器	2注	Z
LDECA [m]	递减数据存储器，结果放入 ACC	2	Z
LDEC [m]	递减数据存储器，结果放入数据存储器	2注	Z
移位			
LRRA [m]	数据存储器右移一位，结果放入 ACC	2	无
LRR [m]	数据存储器右移一位，结果放入数据存储器	2注	无
LRRCA [m]	带进位将数据存储器右移一位，结果放入 ACC	2	C
LRRC [m]	带进位将数据存储器右移一位，结果放入数据存储器	2注	C
LRLA [m]	数据存储器左移一位，结果放入 ACC	2	无
LRL [m]	数据存储器左移一位，结果放入数据存储器	2注	无
LRLCA [m]	带进位将数据存储器左移一位，结果放入 ACC	2	C

助记符	说明	指令周期	影响标志位
LRLC [m]	带进位将数据存储器左移一位，结果放入数据存储器	2注	C
数据传送			
LMOV A,[m]	将数据存储器送至 ACC	2	无
LMOV [m],A	将 ACC 送至数据存储器	2注	无
位运算			
LCLR [m].i	清除数据存储器的位	2注	无
LSET [m].i	置位数据存储器的位	2注	无
转移			
LSZ [m]	如果数据存储器为零，则跳过下一条指令	2注	无
LSZA [m]	数据存储器送至 ACC，如果内容为零，则跳过下一条指令	1注	无
LSNZ [m]	如果数据存储器不为零，则跳过下一条指令	2注	无
LSZ [m].i	如果数据存储器的第 i 位为零，则跳过下一条指令	2注	无
LSNZ [m].i	如果数据存储器的第 i 位不为零，则跳过下一条指令	2注	无
LSIZ [m]	递增数据存储器，如果结果为零，则跳过下一条指令	2注	无
LSDZ [m]	递减数据存储器，如果结果为零，则跳过下一条指令	2注	无
LSIZA [m]	递增数据存储器，将结果放入 ACC，如果结果为零，则跳过下一条指令	2注	无
LSDZA [m]	递减数据存储器，将结果放入 ACC，如果结果为零，则跳过下一条指令	2注	无
查表			
LTABRD [m]	读取当前页的 ROM 内容，并送至数据存储器和 TBLH	3注	无
LTABRDL [m]	读取最后页的 ROM 内容，并送至数据存储器和 TBLH	3注	无
LITABRD [m]	读表指针 TBLP 自加，读取当前页的 ROM 内容，并送至数据存储器和 TBLH	3注	无
LITABRDL [m]	读表指针 TBLP 自加，读取最后页的 ROM 内容，并送至数据存储器和 TBLH	3注	无
其它指令			
LCLR [m]	清除数据存储器	2注	无
LSET [m]	置位数据存储器	2注	无
LSWAP [m]	交换数据存储器的高低字节，结果放入数据存储器	2注	无
LSWAPA [m]	交换数据存储器的高低字节，结果放入 ACC	2	无

注：1. 对扩展跳转指令而言，如果比较的结果牵涉到跳转即需多达 4 个周期，如果没有发生跳转，则只需两个周期。

2. 任何扩展指令若要改变 PCL 的内容将需要 3 个周期来执行。

指令定义

ADC A，[m] Add Data Memory to ACC with Carry

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

ADCM A，[m] Add ACC to Data Memory with Carry

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

ADD A，[m] Add Data Memory to ACC

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

ADD A，x Add immediate data to ACC

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

ADDM A，[m] Add ACC to Data Memory

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

AND A，[m] Logical AND Data Memory to ACC

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

RLA [m]	Rotate Data Memory left with result in ACC
指令说明	将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位，结果送到累加器，而指定数据存储器的内容保持不变。
功能表示	ACC.(i+1) ← [m].i (i=0~6) ACC.0 ← [m].7
影响标志位	无

RLC [m]	Rotate Data Memory Left through Carry
指令说明	将指定数据存储器的内容连同进位标志左移 1 位， 第 7 位取代进位标志且原本的进位标志移到第 0 位。
功能表示	[m].(i+1) ← [m].i (i=0~6) [m].0 ← C C ← [m].7
影响标志位	C

RLC A [m]	Rotate Data Memory left through Carry with result in ACC
指令说明	将指定数据存储器的内容连同进位标志左移 1 位，第 7 位取代进位标志且原本的进位标志移到第 0 位 , 移位结果送回累加器，但是指定数据寄存器的内容保持不变。
功能表示	ACC.(i+1) ← [m].i (i=0~6) ACC.0 ← C C ← [m].7
影响标志位	C

RR [m] Rotate Data Memory right

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

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

影响标志位 无

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

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

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

影响标志位 无

RRC [m] Rotate Data Memory right through Carry

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

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

影响标志位 C

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

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

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

影响标志位 C

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

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

功能表示 ACC ← ACC — [m] — $\overline{C}$

影响标志位 OV、Z、AC、C、SC、CZ

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

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

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

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

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

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

SUB A，[m] Subtract Data Memory from ACC

指令说明 将累加器的内容减去指定的数据存储器的数据，把结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。

功能表示 ACC ← ACC — [m]

影响标志位 OV、Z、AC、C、SC、CZ

SUBM A，[m] Subtract Data Memory from ACC with result in Data Memory

指令说明 将累加器的内容减去指定数据存储器的数据，结果存放到指定的数据存储器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。

功能表示 [m] ← ACC — [m]

影响标志位 OV、Z、AC、C、SC、CZ

SUB A，x Subtract immediate Data from ACC

指令说明 将累加器的内容减去立即数，结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。

功能表示 ACC ← ACC — x

影响标志位 OV、Z、AC、C、SC、CZ

SWAP [m] Swap nibbles of Data Memory

指令说明 将指定数据存储器的低 4 位和高 4 位互相交换。

功能表示 [m].3~[m].0 ↔ [m].7~[m].4

影响标志位 无

SWAPA [m] Swap nibbles of Data Memory with result in ACC

指令说明 将指定数据存储器的低 4 位与高 4 位互相交换，再将结果存放到累加器且指定数据寄存器的数据保持不变。

功能表示 ACC.3~ACC.0 ← [m].7~[m].4

ACC.7~ACC.4 ← [m].3~[m].0

影响标志位 无

SZ [m] Skip if Data Memory is 0

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

功能表示 如果 [m]=0, 跳过下一条指令执行

影响标志位 无

SZA [m]	Skip if Data Memory is 0 with data movement to ACC
指令说明	将指定数据存储器内容复制到累加器，并判断指定数据存储器的内容是否为 0，若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	ACC ← [m]，如果 [m]=0，跳过下一条指令执行
影响标志位	无

SZ [m].i	Skip if bit i of Data Memory is 0
指令说明	判断指定数据存储器的第 i 位是否为 0，若为 0，则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	如果 [m].i=0，跳过下一条指令执行
影响标志位	无

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

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

ITABRD [m]	Increment table pointer low byte first and read table to TBLH and data memory
指令说明	将自加表格指针 TBLP 所指的程序代码低字节（当前页）移至指定的数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无

ITABRDL [m]	Increment table pointer low byte first and read table(last page) to TBLH and data memory
指令说明	将自加表格指针 TBLP 所指的程序代码低字节（最后一页）移至指定的数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无

XOR A，[m]	Logical XOR Data Memory to ACC
指令说明	将累加器的数据和指定的数据存储器内容逻辑异或，结果存放到累加器。
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

扩展指令定义

扩展指令被用来直接存取存储在任何数据存储器 Sector 中的数据。

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

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

LADD A，[m] Add Data Memory to ACC
指令说明 将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、SC

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

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

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

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

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

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

LCPLA [m] Complement Data Memory with result in ACC
指令说明 将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或 0 变 1，结果被存放回累加器且数据寄存器的内容保持不变。
功能表示 ACC ← $[\overline{m}]$
影响标志位 Z

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

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

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

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

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

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

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

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

LORM 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

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

LRLA [m]	Rotate Data Memory left with result in ACC
指令说明	将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位，结果送到累加器，而指定数据存储器的内容保持不变。
功能表示	ACC.(i+1) ← [m].i (i=0~6) ACC.0 ← [m].7
影响标志位	无

LRLC [m]	Rotate Data Memory Left through Carry
指令说明	将指定数据存储器的内容连同进位标志左移 1 位，第 7 位取代进位标志且原本的进位标志移到第 0 位。
功能表示	[m].(i+1) ← [m].i (i=0~6) [m].0 ← C C ← [m].7
影响标志位	C

LRLC 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

LRR [m] Rotate Data Memory right

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

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

影响标志位 无

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

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

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

影响标志位 无

LRRC [m] Rotate Data Memory right through Carry

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

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

影响标志位 C

LRRCA [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

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

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

功能表示 ACC ← ACC — [m] — $\overline{C}$

影响标志位 OV、Z、AC、C、SC、CZ

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

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

功能表示 [m] ← ACC — [m] — $\overline{C}$

影响标志位 OV、Z、AC、C、SC、CZ

LSDZ [m] Skip if Decrement Data Memory is 0

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

功能表示 [m] ← [m] -1，如果 [m]=0 跳过下一条指令执行

影响标志位 无

LSDZA [m] Skip if decrement Data Memory is zero with result in ACC

指令说明 将指定数据存储器内容减 1，判断是否为 0，如果为 0 则跳过下一条指令，此结果将存放到累加器，但指定数据存储器内容不变。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。

功能表示 ACC ← [m]-1，如果 ACC=0 跳过下一条指令执行

影响标志位 无

LSET [m] Set Data Memory

指令说明 将指定数据存储器的每一个位置位为 1。

功能表示 [m] ← FFH

影响标志位 无

LSET [m].i Set bit of Data Memory

指令说明 将指定数据存储器的第 i 位置位为 1。

功能表示 [m].i ← 1

影响标志位 无

LSIZ [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

LSIZA [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

LSNZ [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

LSNZ [m] Skip if Data Memory is not 0

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

功能表示 如果 [m]≠0，跳过下一条指令执行

影响标志位 无

LSUB A，[m] Subtract Data Memory from ACC

指令说明 将累加器的内容减去指定的数据存储器的数据，把结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。

功能表示 ACC ← ACC — [m]

影响标志位 OV、Z、AC、C、SC、CZ

LSUBM A，[m] Subtract Data Memory from ACC with result in Data Memory

指令说明 将累加器的内容减去指定数据存储器的数据，结果存放到指定的数据存储器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。

功能表示 [m] ← ACC — [m]

影响标志位 OV、Z、AC、C、SC、CZ

LSWAP [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

LSWAPA [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 位和高 4 位互相交换，再将结果存放到累加器且指定数据寄存器的数据保持不变。

功能表示 ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0

影响标志位 无

LSZ [m] Skip if Data Memory is 0

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

功能表示 如果 [m]=0□ 跳过下一条指令执行

影响标志位 无

LSZA [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

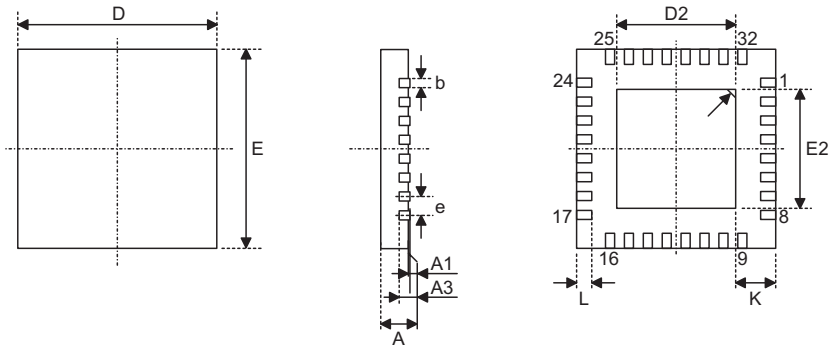
LSZ [m].i	Skip if bit i of Data Memory is 0
指令说明	判断指定数据存储器的第 i 位是否为 0，若为 0，则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	如果 [m].i=0，跳过下一条指令执行
影响标志位	无
LTABRD [m]	Move the ROM code to TBLH and data memory
指令说明	将表格指针 TBLP 所指的程序代码低字节（当前页）移至指定数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
LTABRDL [m]	Read table (last page) to TBLH and Data Memory
指令说明	将表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
LITABRD [m]	Increment table pointer low byte first and read table to TBLH and data memory
指令说明	将自加表格指针 TBHP 和 TBLP 所指的程序代码低字节移至指定的数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
LITABRDL [m]	Increment table pointer low byte first and read table(last page) to TBLH and data memory
指令说明	将自加表格指针 TBLP 所指的程序代码低字节（最后一页）移至指定的数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无

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

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

封装信息

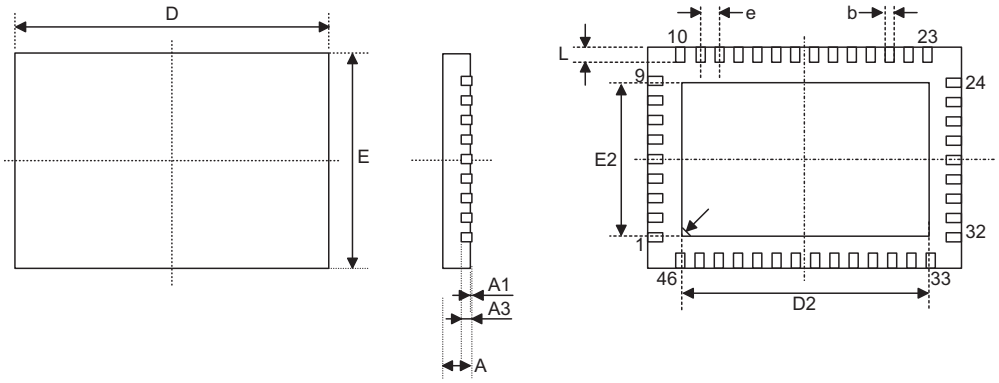
SAW Type 32-pin (5mm × 5mm) QFN 外形尺寸



符号	尺寸 (单位：inch)		
	最小	正常	最大
A	0.028	0.030	0.031
A1	0.000	0.001	0.002
A3	—	0.008 BSC	—
b	0.007	0.010	0.012
D	0.193	0.197	0.201
E	0.193	0.197	0.201
e	—	0.020 BSC	—
D2	0.122	0.126	0.130
E2	0.122	0.126	0.130
L	0.014	0.016	0.018
K	0.008	—	—

符号	尺寸 (单位：mm)		
	最小	正常	最大
A	0.700	0.750	0.800
A1	0.000	0.020	0.050
A3	—	0.203 BSC	—
b	0.180	0.250	0.300
D	4.900	5.000	5.100
E	4.900	5.000	5.100
e	—	0.50 BSC	—
D2	3.10	3.20	3.30
E2	3.10	3.20	3.30
L	0.35	0.40	0.45
K	0.20	—	—

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



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

符号	尺寸 (单位：mm)		
	最小	正常	最大
A	0.800	0.850	0.900
A1	0.000	0.020	0.040
A3	—	0.200 BSC	—
b	0.150	0.200	0.250
D	6.450	6.500	6.550
E	4.450	4.500	4.550
e	—	0.40 BSC	—
D2	5.00	5.10	5.20
E2	3.00	3.10	3.20
L	0.30	0.40	0.50