芯群

产品规格书

经济型 ADC OTP 单片机

BX66R004

版本：V1.41　　日期：2026-10-19

目录

1. 特性 5
1.1 CPU 特性 5
1.2 周边特性 5
2. 概述 6
3. 方框图 6
4. 引脚图 7
5. 引脚说明 7
6. 极限参数 10
7. 直流电气特性 11
7.1 工作电压特性 11
7.2 待机电流特性 11
7.3 工作电流特性 11
8. 交流电气特性 12
8.1 内部高速振荡器 – HIRC – 频率精准度 12
8.2 内部低速振荡器电气特性 – LIRC – 频率精准度 12
8.3 工作频率特性曲线 12
8.4 系统上电时间电气特性 13
9. 输入 / 输出口电气特性 13
10. 存储器电气特性 14
11. A/D 转换器电气特性 14
12. LVR 电气特性 15
13. LCD 电气特性 15
14. 上电复位特性 15
15. 系统结构 16
15.1 时序和流水线结构 16
15.2 程序计数器 17
15.3 堆栈 17
15.4 算术逻辑单元 – ALU 18
16. OTP 程序存储器 18
16.1 结构 18
16.2 特殊向量 18
16.3 查表 18
16.4 查表范例 19
16.5 在线烧录 – ICP 20
16.6 片上调试 – OCDS 20
16.7 OTP ROM 参数烧录功能 – ORPP 21
17. 数据存储器 24
17.1 结构 24
17.2 通用数据存储器 24
17.3 特殊功能数据存储器 24

18. 特殊功能寄存器 26
18.1 间接寻址寄存器 – IAR0, IAR1 26
18.2 存储器指针 – MP0, MP1 26
18.3 累加器 – ACC 26
18.4 程序计数器低字节寄存器 – PCL 27
18.5 表格寄存器 – TBLP, TBLH 27
18.6 状态寄存器 – STATUS 27
19. 振荡器 29
19.1 振荡器概述 29
19.2 系统时钟配置 29
19.3 内部 RC 振荡器 – HIRC 29
19.4 内部 32kHz 振荡器 – LIRC 29
20. 工作模式和系统时钟 30
20.1 系统时钟 30
20.2 系统工作模式 30
20.3 控制寄存器 32
20.4 工作模式切换 32
20.5 待机电流的注意事项 35
20.6 唤醒 35
21. 看门狗定时器 36
21.1 看门狗定时器时钟源 36
21.2 看门狗定时器控制寄存器 36
21.3 看门狗定时器操作 37
22. 复位和初始化 37
22.1 复位功能 37
22.2 复位初始状态 39
23. 输入 / 输出端口 42
23.1 上拉电阻 42
23.2 PA 口唤醒 43
23.3 输入 / 输出端口控制寄存器 43
23.4 引脚共用功能 43
23.5 输入 / 输出引脚结构 47
23.6 编程注意事项 47
24. 8-bit 定时 / 事件计数器 48
24.1 定时 / 事件计数器输入时钟源 48
24.2 定时 / 事件计数器寄存器 48
24.3 定时 / 事件计数器工作模式 51
25. 16-bit 定时 / 事件计数器 55
25.1 定时 / 事件计数器输入时钟源 55
25.2 定时 / 事件计数器寄存器 55
25.3 定时 / 事件计数器工作模式 57
25.4 编程注意事项 59
26. A/D 转换器 60
26.1 A/D 简介 60

26.2 A/D 转换寄存器介绍 60
26.3 A/D 转换器参考电压 62
26.4 A/D 转换器输入信号 62
26.5 A/D 转换器操作 63
26.6 A/D 转换率及时序图 63
26.7 A/D 转换步骤 64
26.8 编程注意事项 64
26.9 A/D 转换功能 65
26.10 A/D 转换应用范例 65
27. 软件控制的 LCD 驱动器 67
27.1 LCD 操作 67
27.2 LCD 偏压电流控制 67
28. 中断 68
28.1 中断寄存器 68
28.2 中断操作 70
28.3 外部中断 71
28.4 定时 / 事件计数器中断 71
28.5 A/D 转换器中断 71
28.6 时基中断 71
28.7 中断唤醒功能 72
28.8 编程注意事项 72
29. 配置选项 73
30. 应用电路 73
31. 指令集 74
31.1 简介 74
31.2 指令周期 74
31.3 数据的传送 74
31.4 算术运算 74
31.5 逻辑和移位运算 74
31.6 分支和控制转换 74
31.7 位运算 75
31.8 查表运算 75
31.9 其它运算 75
32. 指令集概要 76
32.1 惯例 76
33. 指令定义 78
34. 封装信息 89
34.1 16-pin NSOP (150mil) 外形尺寸 90
34.2 20-pin NSOP (150mil) 外形尺寸 91
34.3 20-pin SOP (300mil) 外形尺寸 92
34.4 20-pin SSOP (150mil) 外形尺寸 93

1. 特性

1.1 CPU 特性

- 工作电压：
  - ♦ f_{SYS}=8MHz：2.0V~5.5V
- V_{DD}=5V，系统时钟为 8MHz 时，指令周期为 0.5μs
- 暂停和唤醒功能，以降低功耗
- 振荡器类型：
  - ♦ 内部高速 8MHz RC – HIRC
  - ♦ 内部低速 32kHz RC – LIRC
- 多种工作模式：快速模式、低速模式、空闲模式和休眠模式
- 内部集成的振荡器，无需外接元件
- 所有指令都可在 1 或 2 个指令周期内完成
- 查表指令
- 61 条功能强大的指令系统
- 4 层硬件堆栈
- 位操作指令

1.2 周边特性

- OTP 程序存储器：(2K-16)×15
- 数据存储器：96×8
- OTP ROM 参数烧录功能 – ORPP
- 看门狗定时器功能
- 多达 18 个双向 I/O 口
- 两个与 I/O 口复用的外部中断输入
- 1 个 8-bit 可编程定时 / 事件计数器，具有 PWM 输出功能
  - ♦ 10 通道 PWM 功能，提供 3 种独立可调的占空比
- 1 个 16-bit 可编程定时 / 事件计数器
- 10 个外部通道 8-bit 分辨率的 A/D 转换器
- 1 个时基功能，用于产生固定时间的中断信号
- 低电压复位功能
- 软件控制的 7-SCOM 口 1/2 偏压的 LCD 驱动器
- 封装类型：16-pin NSOP，20-pin NSOP/SOP/SSOP

2. 概述

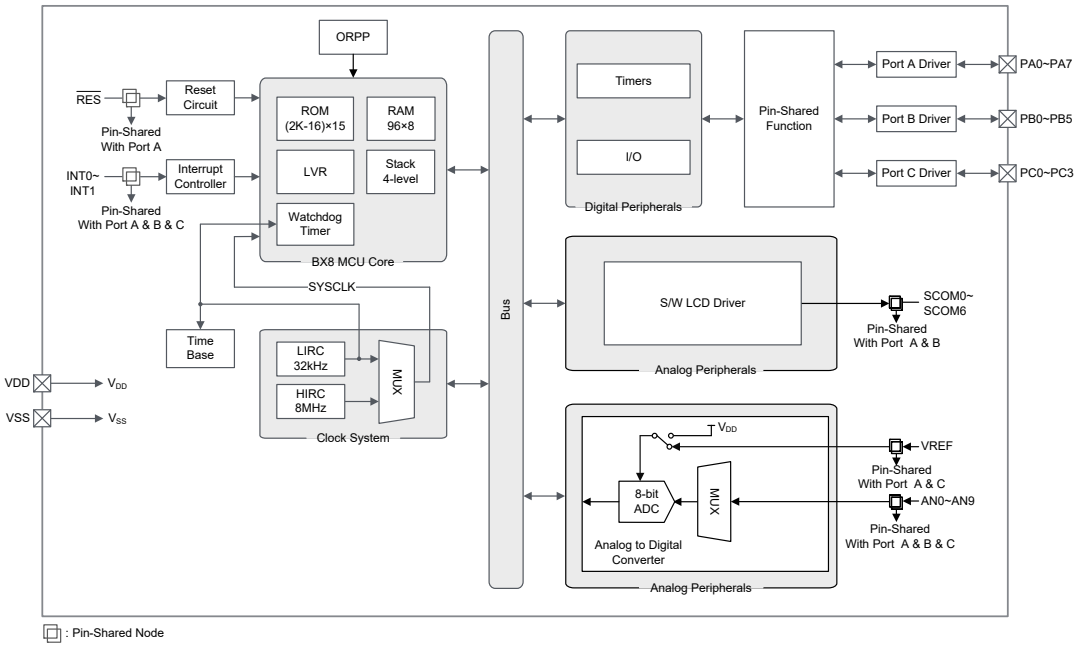
该单片机是一款具有 8 位高性能精简指令集的 OTP 单片机，专为经济型产品设计。

在模拟特性方面，该单片机包含一个多通道 8-bit A/D 转换器。两个使用灵活的定时 / 事件计数器可提供定时功能、事件计数功能及脉冲宽度测量功能。内部看门狗定时器和低电压复位等保护特性，外加优秀的抗干扰和 ESD 保护性能，确保单片机在恶劣的电磁干扰环境下可靠地运行。

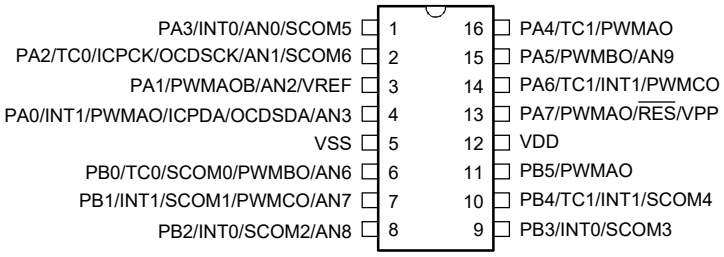
该单片机提供了内部高速和低速振荡器功能选项，这两个内建的系统振荡器无需外围元器件。其在不同工作模式之间动态切换的能力，为用户提供了一个优化单片机操作和减少功耗的手段。

外加 I/O 使用灵活和时基功能等其它特性，使该单片机可以广泛应用于电子测量、环境监控、手持式仪表、家用电器、电子控制工具、马达驱动等方面。

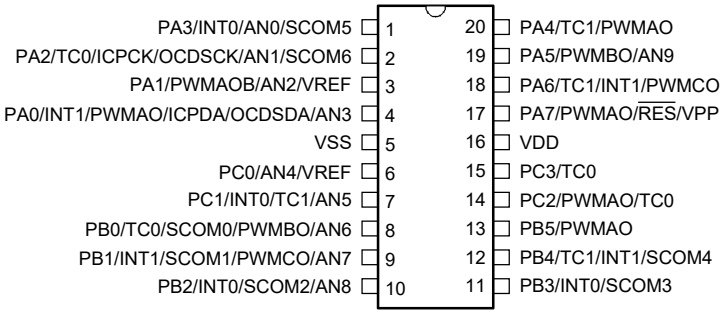
3. 方框图



4. 引脚图



BX66R004/BX66V004
16 NSOP-A



BX66R004/BX66V004
20 NSOP-A/SOP-A/SSOP-A

注：1. 若共用脚同时有多种输出，所需引脚共用功能通过引脚共用寄存器中相应的软件控制位决定。
2. BX66V004 是 BX66R004 的 EV 芯片。OCDSDA 和 OCDSCK 引脚为片上调试功能专用引脚，仅存在于 BX66R004 (Flash 型) 的 EV 芯片 BX66V004 (OTP 型) 中。
3. 在较小封装类型中可能含有未引出的引脚，需合理设置其状态以避免输入浮空造成额外耗电，详见“待机电流注意事项”和“输入 / 输出端口”章节。
4. VPP 引脚为 OTP 烧录高压输入，仅存在于 BX66R004 单片机。

5. 引脚说明

每个引脚的功能如下表所述，而引脚配置的详细内容见规格书其它章节。注意，对于存在不止一种封装的单片机，该表格反映的是较大封装类型的情况。

引脚名称	功能	OPT	I/T	O/T	说明
PA0/INT1/PWMAO/AN3/ICPDA/OCDSDA	PA0	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	INT1	INTEG INTC1 PAS0 IFS	ST	—	外部中断输入 1
	PWMAO	PAS0	—	CMOS	PWMA 信号输出
	AN3	PAS0	AN	—	A/D 转换器外部输入通道
	ICPDA	—	ST	CMOS	ICP 数据 / 地址引脚
	OCDSDA	—	ST	CMOS	OCDS 数据 / 地址，仅用于 EV 芯片

引脚名称	功能	OPT	I/T	O/T	说明
PA1/PWMAOB/AN2/VREF	PA1	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	PWMAOB	PAS0	—	CMOS	PWMA 信号反相输出
	AN2	PAS0	AN	—	A/D 转换器外部输入通道
	VREF	PAS0	AN	—	A/D 转换器外部参考电压输入
PA2/TC0/AN1/SCOM6/ICPCK/OCDSCK	PA2	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	TC0	PAS0 IFS	ST	—	定时 / 事件计数器 0 时钟输入
	AN1	PAS0	AN	—	A/D 转换器外部输入通道
	SCOM6	PAS0	—	AN	软件 LCD COM 输出
	ICPCK	—	ST	—	ICP 时钟引脚
	OCDSCK	—	ST	—	OCDS 时钟引脚，仅用于 EV 芯片
PA3/INT0/AN0/SCOM5	PA3	PAPU PAWU PAS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	INT0	INTEG INTC0 PAS0 IFS	ST	—	外部中断输入 0
	AN0	PAS0	AN	—	A/D 转换器外部输入通道
	SCOM5	PAS0	—	AN	软件 LCD COM 输出
PA4/TC1/PWMAO	PA4	PAPU PAWU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	TC1	PAS1 IFS	ST	—	定时 / 事件计数器 1 时钟输入
	PWMAO	PAS1	—	CMOS	PWMA 信号输出
PA5/PWMBO/AN9	PA5	PAPU PAWU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	PWMBO	PAS1	—	CMOS	PWMB 信号输出
	AN9	PAS1	AN	—	A/D 转换器外部输入通道
PA6/TC1/INT1/PWMCO	PA6	PAPU PAWU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能
	TC1	PAS1 IFS	ST	—	定时 / 事件计数器 1 时钟输入
	INT1	INTEG INTC1 PAS1 IFS	ST	—	外部中断输入 1
	PWMCO	PAS1	—	CMOS	PWMC 信号输出

引脚名称	功能	OPT	I/T	O/T	说明
PA7/PWMAO/$\overline{\text{RES}}$/VPP	PA7	PAWU PAS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻和唤醒功能；此 I/O 与 VPP 共用，若设置为输出高电平会有较大耗电
	PWMAO	PAS1	—	CMOS	PWMA 信号输出
	$\overline{\text{RES}}$	PAS1	ST	—	外部复位输入
	VPP	PAS1	PWR	—	OTP 烧录高压输入，不存在于 EV 芯片
PB0/TC0/SCOM0/PWMBO/AN6	PB0	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	TC0	PBS0 IFS	ST	—	定时 / 事件计数器 0 时钟输入
	SCOM0	PBS0	—	AN	软件 LCD COM 输出
	PWMBO	PBS0	—	CMOS	PWMB 信号输出
	AN6	PBS0	AN	—	A/D 转换器外部输入通道
PB1/INT1/SCOM1/PWMCO/AN7	PB1	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT1	INTEG INTC1 PBS0 IFS	ST	—	外部中断输入 1
	SCOM1	PBS0	—	AN	软件 LCD COM 输出
	PWMCO	PBS0	—	CMOS	PWMC 信号输出
	AN7	PBS0	AN	—	A/D 转换器外部输入通道
PB2/INT0/SCOM2/AN8	PB2	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT0	INTEG INTC0 PBS0 IFS	ST	—	外部中断输入 0
	SCOM2	PBS0	—	AN	软件 LCD COM 输出
	AN8	PBS0	AN	—	A/D 转换器外部输入通道
PB3/INT0/SCOM3	PB3	PBPU PBS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT0	INTEG INTC0 PBS0 IFS	ST	—	外部中断输入 0
	SCOM3	PBS0	—	AN	软件 LCD COM 输出
PB4/TC1/INT1/SCOM4	PB4	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	TC1	PBS1 IFS	ST	—	定时 / 事件计数器 1 时钟输入
	INT1	INTEG INTC1 PBS1 IFS	ST	—	外部中断输入 1
	SCOM4	PBS1	—	AN	软件 LCD COM 输出

引脚名称	功能	OPT	I/T	O/T	说明
PB5/PWMAO	PB5	PBPU PBS1	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	PWMAO	PBS1	—	CMOS	PWMA 信号输出
PC0/AN4/VREF	PC0	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	AN4	PCS0	AN	—	A/D 转换器外部输入通道
	VREF	PCS0	AN	—	A/D 转换器外部参考电压输入
PC1/INT0/TC1/AN5	PC1	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	INT0	INTEG INTC0 IFS	ST	—	外部中断输入 0
	TC1	IFS	ST	—	定时 / 事件计数器 1 时钟输入
	AN5	PCS0	AN	—	A/D 转换器外部输入通道
PC2/TC0/PWMAO	PC2	PCPU PCS0	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	TC0	PCS0 IFS	ST	—	定时 / 事件计数器 0 时钟输入
	PWMAO	PCS0	—	CMOS	PWMA 信号输出
PC3/TC0	PC3	PCPU	ST	CMOS	通用 I/O 口，可通过寄存器设置上拉电阻
	TC0	IFS	ST	—	定时 / 事件计数器 0 时钟输入
VDD	VDD	—	PWR	—	正电源电压
VSS	VSS	—	PWR	—	负电源电压，接地

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

6. 极限参数

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

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

7. 直流电气特性

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

7.1 工作电压特性

Ta=-40°C~85°C

符号	参数	测试条件	最小	典型	最大	单位
V_{DD}	工作电压 – HIRC	$f_{SYS}=f_{HIRC}$=8MHz	2.0	—	5.5	V
	工作电压 – LIRC	$f_{SYS}=f_{LIRC}$=32kHz	2.0	—	5.5	V

7.2 待机电流特性

Ta=25°C，除非另有说明。

符号	待机模式	测试条件		最小	典型	最大	最大 @85°C	单位
		V_{DD}	条件					
I_{STB}	休眠模式	2V	WDT off	—	0.45	0.80	7.00	μA
		3V		—	0.45	0.90	8.00	
		5V		—	0.5	2.0	10.0	
		2V	WDT on	—	1.5	3.0	5.5	μA
		3V		—	1.8	3.6	6.5	
		5V		—	3	5	10	
	空闲模式 0 – LIRC	2V	f_{SUB} on	—	2.4	4.0	8.0	μA
		3V		—	3	5	9	
		5V		—	5	10	11	
	空闲模式 1 – HIRC	2V	f_{SUB} on, f_{SYS}=8MHz	—	288	400	480	μA
		3V		—	360	500	600	
		5V		—	600	800	960	

注：当使用该表格电气特性数据时，以下几点需注意：
1. 任何数字输入都设置为非浮空的状态，与 VPP 共用的 I/O 不要设置为输出高电平的状态。
2. 所有测量都在无负载且所有外围功能关闭的条件下进行。
3. 无直流电流路径。
4. 所有待机电流数值都是在 HALT 指令执行后即停止执行所有指令后测得。

7.3 工作电流特性

Ta=-40°C~85°C

符号	工作模式	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{DD}	低速模式 – LIRC	2V	f_{SYS}=32kHz	—	12	24	μA
		3V		—	15	30	
		5V		—	30	50	
	快速模式 – HIRC	2V	f_{SYS}=8MHz	—	0.6	1.0	mA
		3V		—	0.8	1.2	
		5V		—	1.6	2.4	

注：当使用该表格电气特性数据时，以下几点需注意：
1. 任何数字输入都设置为非浮空的状态，与 VPP 共用的 I/O 不要设置为输出高电平的状态。
2. 所有测量都在无负载且所有外围功能关闭的条件下进行。

3. 无直流电流路径。
4. 所有工作电流数值通过一个连续的 NOP 指令循环程序测得。

8. 交流电气特性

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

8.1 内部高速振荡器 – HIRC – 频率精准度

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

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	温度				
f_{HIRC}	通过烧录器调整后的 8MHz HIRC 频率	3V/5V	25°C	-1.5%	8	+1.5%	MHz
			-40°C~85°C	-4%	8	+4%	
		2.0V~5.5V	25°C	-3%	8	+3%	
			-40°C~85°C	-5%	8	+5%	

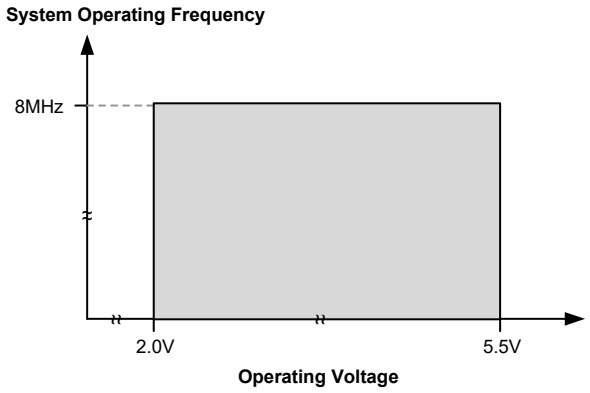
注：1. 烧录器可在 3V/5V 这两个可选的固定电压下对 HIRC 频率进行调整，在此提供 V_{DD}=3V/5V 时的参数值。
2. 3V/5V 表格列下面提供的是全压条件下的参数值。当应用电压范围是 2.0V~3.6V 时，建议烧录器电压固定在 3V；当应用电压范围是 3.3V~5.5V 时，建议烧录器电压固定在 5V。
3. 表格中提供的最小和最大误差值仅在对应的烧录器调整频率下有效。
4. 实际应用时，请注意 VDD 与 VSS 间需加 0.1μF 去耦电容，并尽可能接近 VDD 引脚，避免频率误差超出规格。

8.2 内部低速振荡器电气特性 – LIRC – 频率精准度

Ta=-40°C~85°C，除非另有说明。

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	温度				
f_{LIRC}	LIRC 频率	2.0V~5.5V	25°C	-20%	32	+20%	kHz
			-40°C~85°C	-50%	32	+60%	
t_{START}	LIRC 启动时间	—	—	—	—	500	μs

8.3 工作频率特性曲线



8.4 系统上电时间电气特性

Ta=-40°C~85°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
t_{SST}	系统启动时间 (从 f_{SYS} off 的状态下唤醒)	—	$f_{SYS}=f_H\sim f_H/64$，$f_H=f_{HIRC}$	—	128	—	t_{SYS}
		—	$f_{SYS}=f_{SUB}=f_{LIRC}$	—	2	—	t_{SUB}
	系统启动时间 (从 f_{SYS} on 的状态下唤醒)	—	$f_{SYS}=f_H\sim f_H/64$，$f_H=f_{HIRC}$	—	2	—	t_{SYS}
		—	$f_{SYS}=f_{SUB}=f_{LIRC}$	—	2	—	t_{SUB}
	系统速度切换时间 (快速模式 → 低速模式或 低速模式 → 快速模式)	—	f_{HIRC} off → on	—	128	—	t_{HIRC}
t_{RSTD}	系统复位延迟时间 (上电复位或 LVR 硬件复位)	—	RR_{POR}=5V/ms	5	16	80	ms
	系统复位延迟时间 (WDT 溢出或 $\overline{RES}$ 引脚复位)	—	—				

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

9. 输入 / 输出口电气特性

Ta=-40°C~85°C，除非另有说明。

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{IL}	I/O 口低电平输入电压	5V	—	0	—	1.5	V
		—	—	0	—	$0.2V_{DD}$	
	$\overline{RES}$ 引脚低电平输入电压	—	$V_{DD}\geq2.7$	0	—	$0.4V_{DD}$	V
		—	$2.0\leq V_{DD}<2.7$	0	—	$0.3V_{DD}$	V
V_{IH}	I/O 口高电平输入电压	5V	—	3.5	—	5	V
		—	—	$0.8V_{DD}$	—	V_{DD}	
	$\overline{RES}$ 引脚高电平输入电压	—	—	$0.9V_{DD}$	—	V_{DD}	V
I_{OL}	I/O 口灌电流	3V	$V_{OL}=0.1V_{DD}$	5	10	—	mA
		5V		10	20	—	
I_{OH}	I/O 口源电流	3V	$V_{OH}=0.9V_{DD}$	-2.5	-5.0	—	mA
		5V		-5.0	-10	—	
R_{PH}	I/O 口上拉电阻(注)	3V	—	20	60	100	kΩ
		5V	—	10	30	50	
I_{LEAK}	I/O 口输入漏电流	5V	$V_{IN}=V_{DD}$ 或 $V_{IN}=V_{SS}$	—	—	±1	μA
t_{TC}	TCn 时钟输入最小脉宽	—	—	20	—	—	ns
t_{INT}	中断引脚最小输入脉宽	—	—	0.3	—	—	μs

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
t_{RES}	外部复位最小低电压保持时间	—	—	0.3	—	—	μs

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

10. 存储器电气特性

Ta=-40°C~85°C，除非另有说明。

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{DD}	V_{DD} 读工作电压 – ORPP 存储器	—	—	2.0	—	5.5	V
	V_{DD} 写工作电压 – ORPP 存储器	—	—	4.5	—	5.5	V
OTP 程序存储器							
V_{PP}	V_{PP} 写工作电压 – ORPP 存储器	—	—	8.25	8.50	8.75	V
t_{WR}	写周期时间 – ORPP 存储器	—	—	—	300	450	μs
E_P	存储单元耐受性 – ORPP 存储器	—	—	1	—	—	W
t_{RETD}	ROM 数据保存时间	—	Ta=25°C	—	40	—	Year
Flash 程序存储器 – 仅用于 BX66V004							
t_{ACTV}	ROM 激活时间 – 从暂停模式唤醒	—	—	32	—	64	μs
RAM 数据存储器							
V_{DR}	RAM 数据保存电压	—	—	1.0	—	—	V

注：1. “W” 表示写次数。
2. 在计算从空闲 / 休眠模式唤醒的系统总启动时间时，还需加上 ROM 激活时间 t_{ACTV}。

11. A/D 转换器电气特性

Ta=-40°C~85°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{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
N_R	分辨率	—	—	—	—	8	Bit
DNL	A/D 非线性微分误差	—	$V_{REF}=V_{DD}$, $t_{ADCK}=0.5\mu s$	-1	—	1	LSB
INL	A/D 非线性积分误差	—	$V_{REF}=V_{DD}$, $t_{ADCK}=0.5\mu s$	-1	—	1	LSB
I_{ADC}	A/D 转换器使能的额外电流	2.2V	无负载，$t_{ADCK}=0.5\mu s$	—	150	225	μA
		3V		—	220	330	
		5V		—	400	600	
t_{ADCK}	A/D 转换器时钟周期	—	—	0.5	—	10.0	μs
t_{ON2ST}	A/D 转换器 On-to-Start 时间	—	—	4	—	—	μs
t_{ADC}	A/D 转换时间 (包括采样和保持时间)	—	—	—	12	—	t_{ADCK}

12. LVR 电气特性

Ta=-40°C~85°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
V_{LVR}	低电压复位电压	—	LVR 使能，电压选择 1.9V	-5%	1.9	+5%	V
			LVR 使能，电压选择 2.1V	-5%	2.1	+5%	
			LVR 使能，电压选择 3.15V	-5%	3.15	+5%	
			LVR 使能，电压选择 4.2V	-5%	4.2	+5%	
I_{LVR}	工作电流	3V	LVR 使能，V_{LVR}=1.9V	—	—	15	μA
		5V	LVR 使能，V_{LVR}=1.9V	—	15	30	
t_{LVR}	产生 LVR 复位的低电压最短保持时间	—	—	120	240	480	μs

13. LCD 电气特性

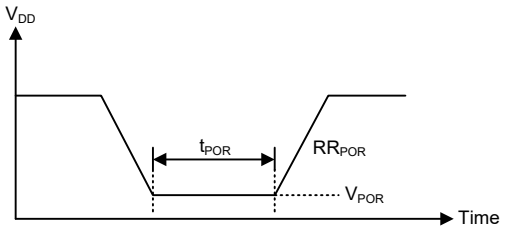
Ta=-40°C~85°C

符号	参数	测试条件		最小	典型	最大	单位
		V_{DD}	条件				
I_{BIAS}	LCD V_{DD}/2 偏压电流	3V	ISEL[2:0]=000B	10.5	15.0	22.5	μA
		5V		17.5	25.0	34.5	μA
		3V	ISEL[2:0]=001B	21	30	39	μA
		5V		35	50	65	μA
		3V	ISEL[2:0]=010B	42	60	78	μA
		5V		70	100	130	μA
		3V	ISEL[2:0]=011B	82.6	118.0	153.4	μA
		5V		140	200	260	μA
		3V	ISEL[2:0]=100B	1.5	3.0	4.5	μA
		5V		2.5	5.0	7.5	μA
		3V	ISEL[2:0]=101B~111B	5.2	7.5	9.8	μA
		5V		8.7	12.5	16.3	μA
V_{SCOM}	LCD COM 端口 V_{DD}/2 电压	2.2V~5.5V	无负载	0.475 V_{DD}	0.5V_{DD}	0.525 V_{DD}	V

14. 上电复位特性

Ta=-40°C~85°C

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



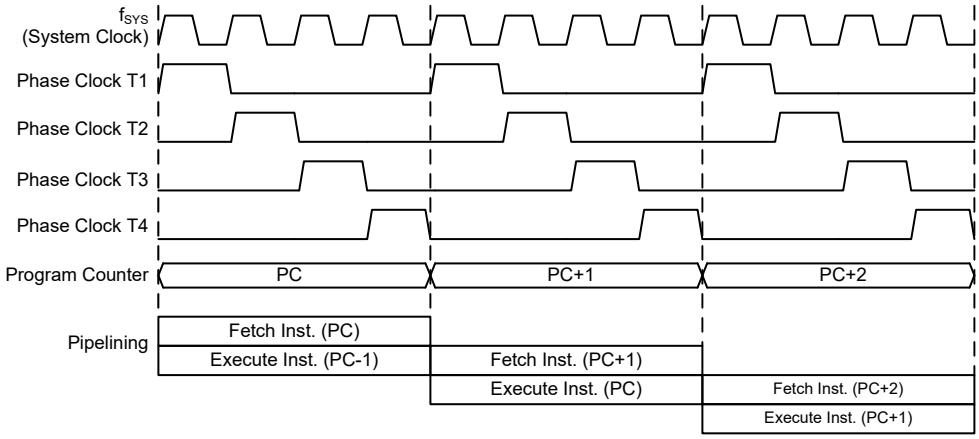
15. 系统结构

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

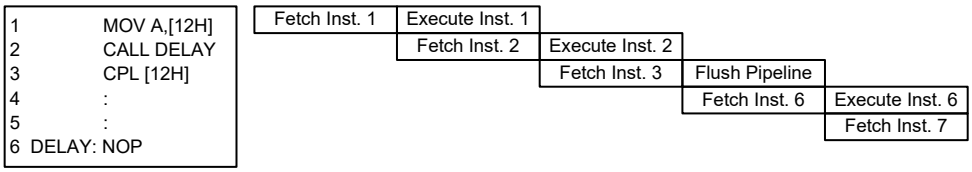
15.1 时序和流水线结构

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

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



系统时序和流水线



指令捕捉

15.2 程序计数器

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

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

程序计数器	
高字节	低字节 (PCL)
PC10~PC8	PCL7~PCL0

程序计数器

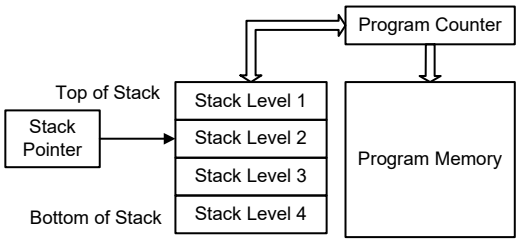
程序计数器的低字节，即程序计数器的低字节寄存器 PCL，可以通过程序控制，且它是可以读取和写入的寄存器。通过直接写入数据到这个寄存器，一个程序短跳转可直接执行，然而只有低字节的操作是有效的，跳转被限制在存储器的当前页中，即 256 个存储器地址范围内，当这样一个程序跳转要执行时，会插入一个空指令周期。程序计数器的低字节可由程序直接进行读取，PCL 的使用可能引起程序跳转，因此需要额外的指令周期。

15.3 堆栈

堆栈是一个特殊的存储空间，用来存储程序计数器中的内容。该单片机有四层堆栈，堆栈既不是数据部分也不是程序空间部分，而且它既不是可读取也不是可写入的。当前层由堆栈指针 (SP) 加以指示，同样也是不可读写的。在子程序调用或中断响应服务时，程序计数器的内容被压入到堆栈中。当子程序或中断响应结束时，返回指令 (RET 或 RETI) 使程序计数器从堆栈中重新得到它以前的值。当一个芯片复位后，堆栈指针将指向堆栈顶部。

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

若堆栈溢出，则首个存入堆栈的程序计数器数据将会丢失。



15.4 算术逻辑单元 – ALU

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

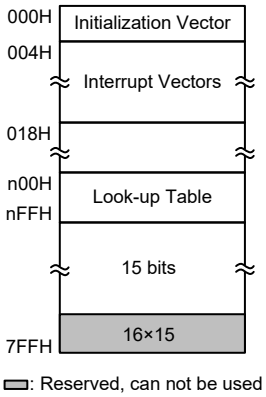
- 算术运算：ADD，ADDM，ADC，ADCM，SUB，SUBM，SBC，SBCM，DAA
- 逻辑运算：AND，OR，XOR，ANDM，ORM，XORM，CPL，CPLA
- 移位运算：RRA，RR，RRCA，RRC，RLA，RL，RLCA，RLC
- 递增和递减：INCA，INC，DECA，DEC
- 分支判断：JMP，SZ，SZA，SNZ，SIZ，SDZ，SIZA，SDZA，CALL，RET，RETI

16. OTP 程序存储器

程序存储器用来存放用户代码即储存程序。此单片机提供一次可编程的存储器 (OTP)，用户可将应用程序写入到芯片一次。

16.1 结构

程序存储器的容量为 (2K-16)×15 位。注意，减掉 16×15 位的区域需保留且不能被使用。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

16.2 特殊向量

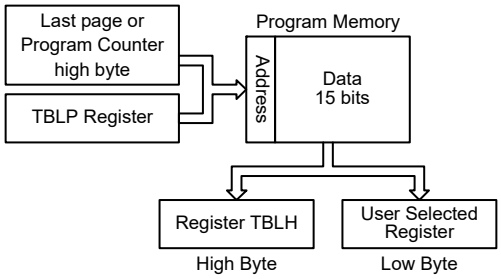
程序存储器内部某些地址保留用做诸如复位和中断入口等特殊用途。地址 000H 是芯片复位后的程序起始地址。在芯片复位之后，程序将跳到这个地址并开始执行。

16.3 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 中。这个寄存器定义表格总的地址。

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



16.4 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“0700H”指向的地址是 2K 程序存储器中最后一页的起始地址。表格指针低字节寄存器的初始值设为 06H，这可保证从数据表格读取的第一笔数据位于程序存储器地址 0706H，即最后一页起始地址后的第六个地址。值得注意的是，假如“TABRD [m]”指令被使用，则表格指针指向 TBLP 寄存器所指定的地址。在这个例子中，表格数据的高字节等于零，而当“TABRD [m]”指令被执行时，此值将会自动的被传送到 TBLH 寄存器。

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

16.4.1 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 present page
:
:
tabrd tempreg1           ; transfers value in table referenced by table pointer,
                         ; data at program memory address “0706H” 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 “0705H” transferred to
                         ; tempreg2 and TBLH
                         ; in this example the data “1AH” is transferred to
                         ; tempreg1 and data “0FH” to register tempreg2
                         ; the value “00H” will be transferred to the high byte
                         ; register TBLH
:
:
org 0700h                ; sets initial address of program memory
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
```

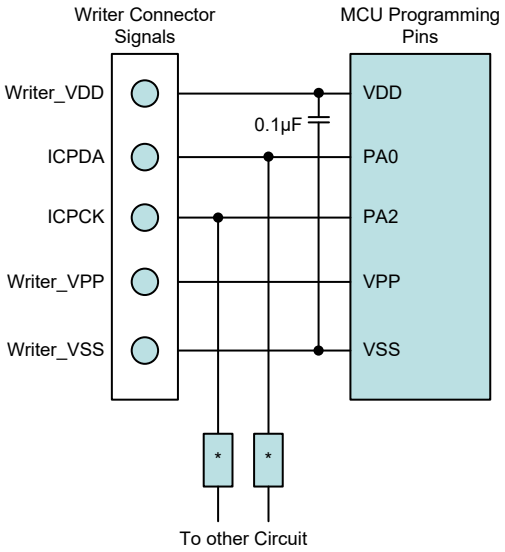
16.5 在线烧录 – ICP

提供 OTP 型程序存储器，用户可将应用程序写入到芯片一次。另外，芯群单片机提供 5 线接口的在线烧录方式。用户可将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烧录。

芯群烧录器引脚名称	MCU 在线烧录引脚名称	功能
ICPDA	PA0	串行数据 / 地址烧录
ICPCK	PA2	时钟烧录
VPP	VPP	OTP ROM 可烧录电源 (8.5V)
VDD	VDD	电源；烧录时需在 VDD 和 VSS 之间接一个 0.1μF 的电容
VSS	VSS	地

程序存储器可以通过 5 线的接口在线进行烧录。其中一条线用于数据串行下载或上传、一条线用于串行时钟、剩下三条用于提供电源。芯片在线烧写的详细使用说明超出此文档的描述范围，将由专门的参考文献提供。

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



注：1. ICP 烧录时，需在 VDD 和 VSS 之间接一个 0.1μF 的电容，并且尽可能靠近这两个引脚。
2. * 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。

16.6 片上调试 – OCDS

EV 芯片 BX66V004 用于单片机 BX66R004 仿真。EV 芯片提供片上调试功能 (OCDS) 用于开发过程中的单片机调试。除了片上调试功能外，单片机和 EV 芯片在功能上几乎是兼容的。用户可将 OCDSDA 和 OCDSCK 引脚连接至芯群 BX-IDE 开发工具，从而实现 EV 芯片对单片机的仿真。OCDSDA 引脚为 OCDS 数据 / 地址输入 / 输出脚，OCDSCK 引脚为 OCDS 时钟输入脚。当用户用 EV 芯片进行调试时，OCDSDA 和 OCDSCK 引脚上的其它共用功能在 EV 芯片无效。关于 OCDS 功能的详细描述，请参考“芯群 e-Link for 8-bit MCU OCDS 用户手册”文件。

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

16.7 OTP ROM 参数烧录功能 – ORPP

该单片机内建 ORPP 功能。单片机提供的 ORPP 功能使用户可以方便地对 OTP 程序存储器进行烧录。注意，写操作只可以对 OTP 程序存储器的最后一页写入数据，数据只能写入一次且无法擦除。

执行写操作之前需将 VPP 引脚接到 8.5V 电源，并且在写入操作完成后将高压电源从 VPP 引脚上移除。若 VPP 功能与 I/O 口共用引脚，则在使用 VPP 功能时，相应的 I/O 口不能设置为输出。

16.7.1 ORPP 寄存器

有三个寄存器可控制内部 ORPP 功能，即数据寄存器 ODL 和 ODH 以及控制寄存器 OCR。

寄存器名称	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OCR	—	—	—	—	WREN	WR	—	—
ODL	D7	D6	D5	D4	D3	D2	D1	D0
ODH	—	D14	D13	D12	D11	D10	D9	D8

ORPP 寄存器列表

- **ODL 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**：ORPP 程序存储器数据 bit 7~bit 0

- **ODH 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	D14	D13	D12	D11	D10	D9	D8
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 6~0　　**D14~D8**：ORPP 程序存储器数据 bit 14~bit 8

- **OCR 寄存器**

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

Bit 7~4　未定义，读为“0”

Bit 3　**WREN**：ORPP 写使能位
0：除能
1：使能
此位为 ORPP 写使能位，向 ORPP 写操作之前需将此位置高。写周期结束后，硬件自动将此位清零。将此位清零时，则禁止向 ORPP 写操作。

Bit 2　**WR**：ORPP 写控制位
0：写周期结束
1：开始写周期
此位为 ORPP 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 1~0　未定义，读为“0”

注：1. 在同一条指令中 WREN 和 WR 不能同时置为“1”。
2. 应注意，当写操作成功启动后，CPU 操作将停止。
3. 需确保 f_{SUB} 时钟运行稳定后才可执行写操作。
4. 需确保写操作已执行完毕后才可执行其它操作。

16.7.2 ORPP 写数据到 OTP 程序存储器

为了实现 ORPP 写操作，ORPP 中写入的数据需存入 ODH 和 ODL 寄存器中，要写入数据的地址需放入 TBLP 寄存器中。OCR 寄存器中的写使能位 WREN 先置高以使能写功能，然后 OCR 寄存器中的 WR 位需立即置高以开始写操作。这两条指令必须在两个指令周期内连续执行才可成功启动一个写操作。进行写操作之前应先将总中断使能位 EMI 清零，在一个有效的写启动步骤完成之后再将其使能。应注意，当写操作成功启动后，CPU 操作将停止。若写周期完成，CPU 将继续执行应用程序，WR 位将自动清除为“0”，通知用户数据已写入 OTP 程序存储器。

16.7.3 ORPP 从 OTP 程序存储器中读取数据

对于 ORPP 读操作，应首先将要读取数据的地址放入 TBLP 寄存器中。然后该数据可以使用“TABRDL [m]”指令从程序存储器中读取。当指令执行时，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。

16.7.4 编程注意事项

必须注意的是数据不会无意写入 OTP 程序存储器。在没有写动作时写使能位被正常清零可以增强保护功能。尽管没有必要，写一个简单的读回程序以检查新写入的数据是否正确还是应该考虑的。

WREN 位置位后，OCR 寄存器中的 WR 位需立即置位，以确保写周期正确地执行。写周期开始前总中断位 EMI 应先清零，在一个有效的写启动步骤完成之后再将此位重新使能。注意，单片机不应在 ORPP 写操作完全完成之前进入空闲或休眠模式，否则 ORPP 写操作将失败。

16.7.5 程序举例

ORPP 从 OTP 程序存储器中读取数据

```
Tempreg1 db?              ; temporary register
MOV A, 03H
MOV TBLP, A               ; set read address 03H
TABRDL Tempreg1           ; transfers value in table (last page) referenced
                          ; by table pointer, data at program memory address "0703H"
                          ; transferred to tempreg1 and TBLH
```

ORPP 写数据到 OTP 程序存储器

```
MOV A, ORPP_ADRES         ; user defined address
MOV TBLP, A
MOV A, ORPP_DATA_L        ; user defined data
MOV ODL, A
MOV A, ORPP_DATA_H
MOV ODH, A
MOV A, 00H
MOV OCR, A
CLR EMI
SET WREN                  ; set WREN bit, enable write operation
SET WR                    ; start Write Cycle - set WR bit - executed immediately
                          ; after setting WREN bit
SET EMI
BACK:
SZ  WR                    ; check for write cycle end
JMP BACK
NOP
```

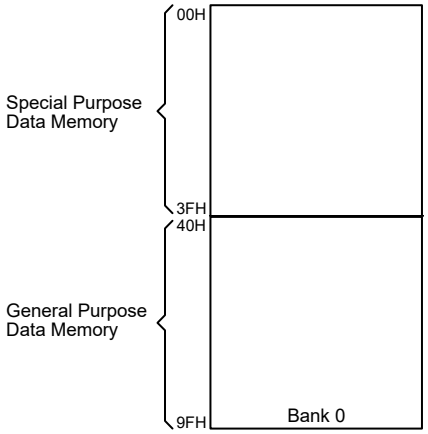
17. 数据存储器

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。

17.1 结构

数据存储器分为两种类型，第一种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。第二种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。

此单片机数据存储器的起始地址是“00H”。特殊功能数据存储器地址范围为 00H~3FH，而通用数据存储器地址范围为 40H~9FH。



数据存储器结构

17.2 通用数据存储器

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

17.3 特殊功能数据存储器

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

	Bank 0
00H	IAR0
01H	MP0
02H	IAR1
03H	MP1
04H	
05H	ACC
06H	PCL
07H	TBLP
08H	TBLH
09H	
0AH	STATUS
0BH	
0CH	INTEG
0DH	WDTC
0EH	TBC
0FH	
10H	SCC
11H	
12H	PAS0
13H	PAS1
14H	PA
15H	PAC
16H	PAPU
17H	PAWU
18H	
19H	
1AH	INTC0
1BH	INTC1
1CH	TMR1C
1DH	TMR1L
1EH	TMR1H
1FH	IFS
20H	SADC0
21H	SADC1
22H	SADO
23H	PB
24H	PBC
25H	PBPU
26H	PBS0
27H	PC
28H	PCC
29H	PCPU
2AH	PCS0
2BH	SCOMC
2CH	PBS1
2DH	OCR
2EH	ODL
2FH	ODH
30H	TMR0C/PWMC
31H	TMR0/PWMADATA
32H	PWMBDATA
33H	PWMCDATA
34H	
⋮	
3FH	

: Unused, read as 00H

特殊功能数据存储器结构

18. 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。

18.1 间接寻址寄存器 – IAR0, IAR1

间接寻址寄存器 IAR0 和 IAR1 的地址虽位于数据存储器寄存器区域，但不同于普通寄存器，它们没有实际的物理地址。与定义实际存储器地址的直接存储器寻址不同，间接寻址是使用间接寻址寄存器和存储器指针来执行存储器数据操作。在间接寻址寄存器 IAR0 和 IAR1 上的任何动作，将对存储器指针 MP0 或 MP1 所指定的存储器地址产生对应的读 / 写操作。它们总是成对出现，IAR0 和 MP0 或 IAR1 和 MP1 可以访问 Bank 0。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，而直接写入这些寄存器则不做任何操作。

18.2 存储器指针 – MP0, MP1

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

以下例子说明如何清空一个具有 4 个 RAM 地址的区块，它们已事先定义成地址 adres1 到 adres4。

18.2.1 间接寻址程序举例

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

在上面的例子中有一点值得注意，即并没有确定 RAM 地址。

18.3 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。

18.4 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，将程序计数器低字节规划在数据存储器的特殊功能区域内，通过对此寄存器进行操作，便可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。注意，当执行此操作时，会插入一个空指令周期。

18.5 表格寄存器 – TBLP, TBLH

特殊功能寄存器 TBLP 用于对存储在程序存储器中的表格进行操作。TBLP 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预先设定，由于它们的值可以被如“INC”或“DEC”的指令所改变，这就提供了一种简便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。

18.6 状态寄存器 – STATUS

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

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

Z、OV、AC 和 C 标志位通常反映最近运算的状态。

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

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

- **STATUS 寄存器**

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

“x”：未知

Bit 7~6　未定义，读为“0”

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

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

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

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

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

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未发生借位
C 也受循环移位指令的影响。

19. 振荡器

不同的振荡器选择可以让使用者在不同的应用需求中实现更大范围的功能。振荡器的灵活性使得在速度和功耗方面可以达到较优化。振荡器操作是通过相关的控制寄存器完成的。

19.1 振荡器概述

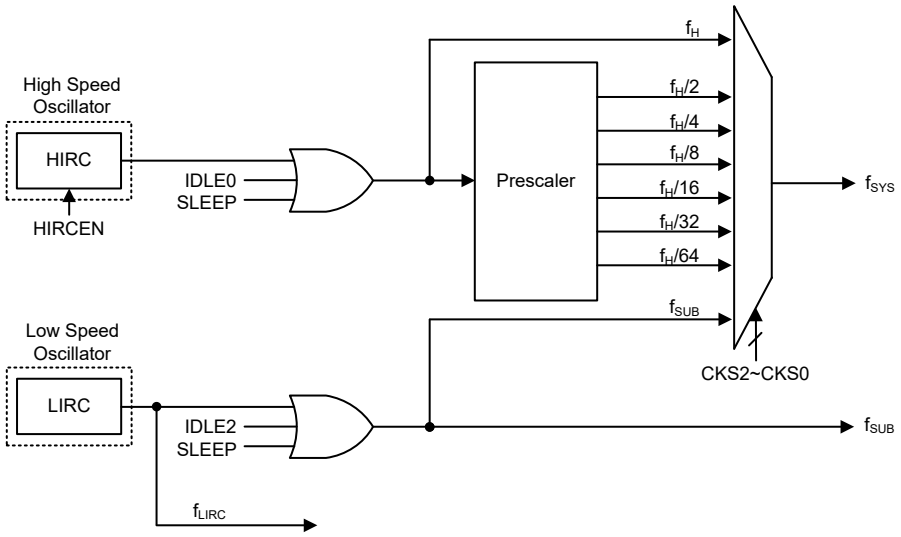
振荡器除了作为系统时钟源，还作为看门狗定时器和时基中断的时钟源。集成的内部振荡器不需要任何外围器件。它们提供的高速和低速系统振荡器具有较宽的频率范围。较高频率的振荡器提供更高的性能，但要求有更高的功率，反之亦然。动态切换快慢系统时钟的能力使单片机具有灵活而优化的性能 / 功耗比，此特性对功耗敏感的应用领域尤为重要。

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

振荡器类型

19.2 系统时钟配置

该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 高速振荡器 HIRC，低速振荡器为内部 32kHz 低速振荡器 LIRC。使用高速或低速振荡器作为系统时钟的选择是通过设置 SCC 寄存器中的 CKS2~CKS0 位决定的，系统时钟可动态选择。



系统时钟配置选项

19.3 内部 RC 振荡器 – HIRC

内部 RC 振荡器是一个集成的系统振荡器，无需其它外部器件。内部 RC 振荡器频率固定为 8MHz。芯片在制造时进行调整且内置频率补偿电路，使得振荡频率因 V_{DD}、温度以及芯片制成工艺不同的影响较大程度的降低。该内部时钟无需额外的引脚。

19.4 内部 32kHz 振荡器 – LIRC

内部 32kHz 系统振荡器是一个完全集成的低频 RC 振荡器，它的典型频率值为 32kHz 且无需外部元件。芯片在制造时进行调整且内置频率补偿电路，使得振荡器因电源电压、温度及芯片制成工艺不同的影响较大程度的降低。

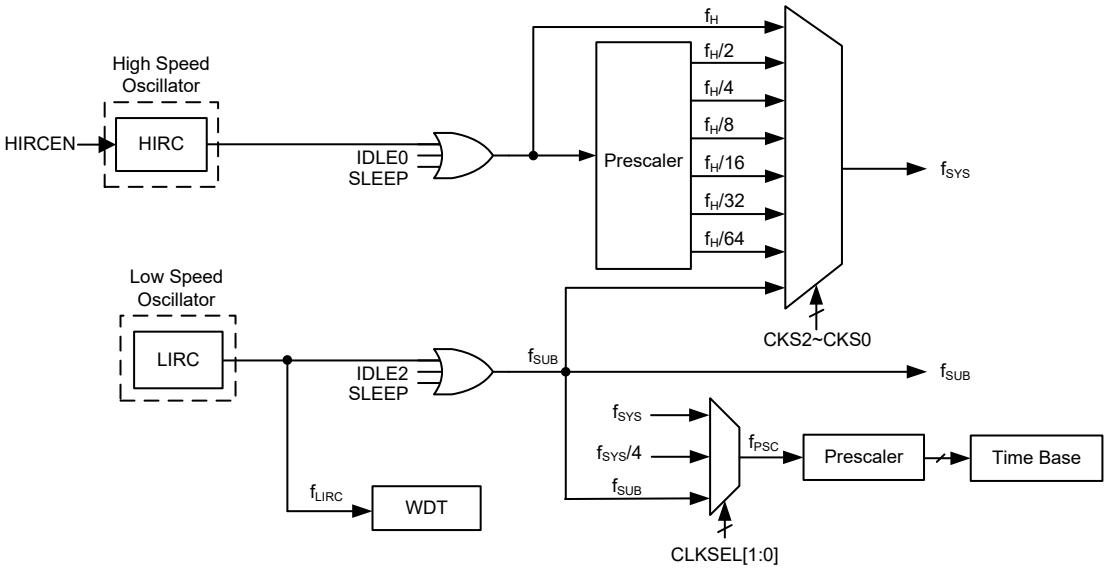
20. 工作模式和系统时钟

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

20.1 系统时钟

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

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



单片机时钟选项

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

20.2 系统工作模式

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

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

“x”：无关

注：1. 在低速模式中，f_H 开启或关闭由相应的振荡器使能位控制。
2. 在休眠模式中，f_{LIRC} 开启或关闭由 WDT 功能使能或除能控制。

快速模式

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

低速模式

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

休眠模式

执行 HALT 指令后且 SCC 寄存器中的 FHIDEN 和 FSIDEN 位都为低时，系统进入休眠模式。在休眠模式中，CPU 停止运行，f_{SUB} 停止为外围功能提供时钟。若看门狗定时器功能使能，f_{LIRC} 继续运行。

空闲模式 0

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

空闲模式 1

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

空闲模式 2

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

20.3 控制寄存器

寄存器 SCC 用于控制系统时钟和相应的振荡器配置。

- **SCC 寄存器**

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

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

Bit 4 未定义，读为“0”

Bit 3 **HIRCF**：HIRC 振荡器稳定标志位
0：HIRC 未稳定
1：HIRC 稳定
此位用于表明 HIRC 振荡器是否稳定。HIRCEN 位置高使能 HIRC 振荡器，HIRCF 位会先被清零，在 HIRC 稳定后会被置高。

Bit 2 **HIRCEN**：HIRC 振荡器使能控制位
0：除能
1：使能

Bit 1 **FHIDEN**：CPU 关闭时高频振荡器控制位
0：除能
1：使能
此位用来控制在执行 HALT 指令关闭 CPU 后高速振荡器是开启还是停止。

Bit 0 **FSIDEN**：CPU 关闭时低频振荡器控制位
0：除能
1：使能
此位用来控制在执行 HALT 指令关闭 CPU 后低速振荡器是开启还是停止。

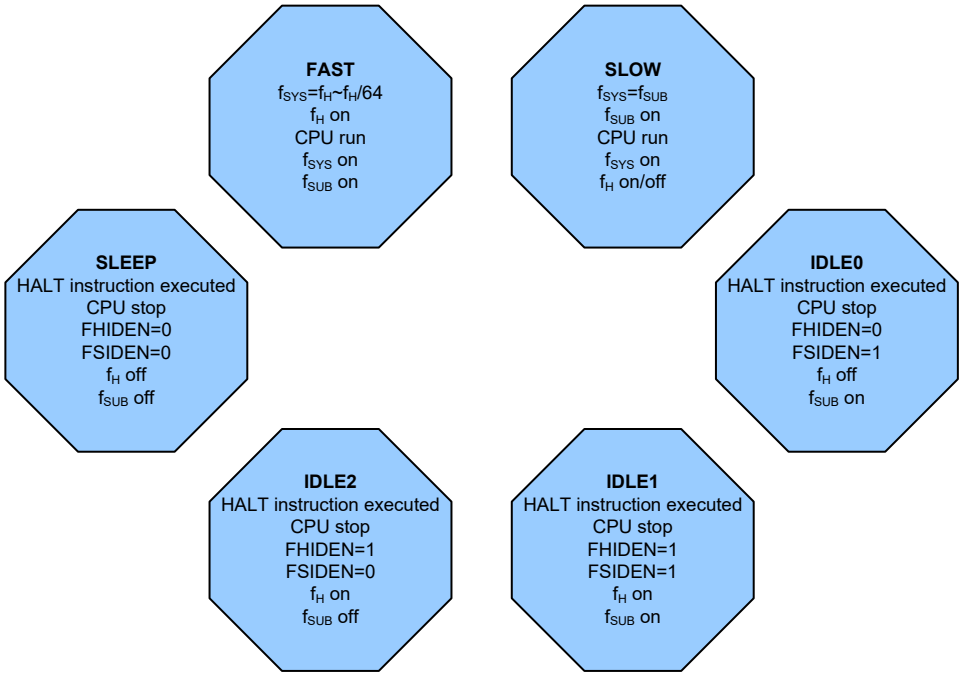
注：使用 CKS2~CKS0 位进行时钟切换设置之后，在相关时钟成功切换至目标时钟源之前需要一定的延时。因此，若接下来执行的操作需要目标时钟源立即响应，则在此之前必须规划适当的延迟时间。

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

20.4 工作模式切换

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

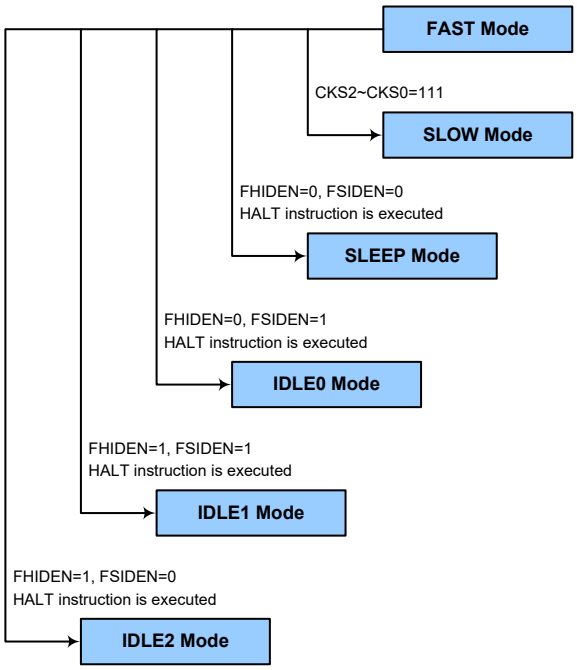
简单来说，快速模式和低速模式间的切换仅需设置 SCC 寄存器中的 CKS2~CKS0 位即可实现，而快速模式 / 低速模式与休眠模式 / 空闲模式间的切换经由 HALT 指令实现。当 HALT 指令执行后，单片机是否进入空闲模式或休眠模式由 SCC 寄存器中的 FHIDEN 和 FSIDEN 位决定的。



20.4.1 快速模式切换到低速模式

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

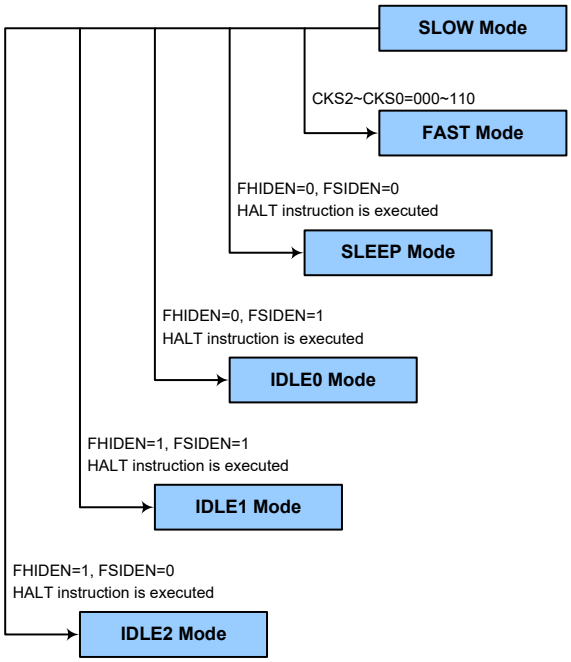
低速模式的系统时钟源来自 LIRC 振荡器，因此要求此振荡器在所有模式切换动作发生前稳定下来。



20.4.2 低速模式切换到快速模式

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

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



20.4.3 进入休眠模式

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

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

20.4.4 进入空闲模式 0

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

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

- 如果 WDT 功能使能，WDT 将被清零并重新开始计数。如果 WDT 功能除能，WDT 将被清零并停止计数。

20.4.5 进入空闲模式 1

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

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

20.4.6 进入空闲模式 2

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

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

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

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

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

20.6 唤醒

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

系统进入休眠或空闲模式之后，可以通过以下几种方式唤醒：

- PA 口下降沿

- 外部 $\overline{RES}$ 引脚复位
- 系统中断
- WDT 溢出

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

如果单片机由外部 $\overline{RES}$ 引脚唤醒，系统会完全复位；若系统由看门狗计数器溢出唤醒，将会启动看门狗复位并置位 TO 标志，这种复位只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。

21. 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。

21.1 看门狗定时器时钟源

WDT 定时器时钟源 f_{LIRC} 由内部低速振荡器 LIRC 提供。内部振荡器 LIRC 的频率大约为 32kHz，这个特殊的内部时钟周期会随 V_{DD}、温度和制成的不同而变化。看门狗定时器的时钟源可分频为 $[(2^8-2^0)\sim2^8]\sim[(2^{15}-2^7)\sim2^{15}]$ 以提供更大的溢出周期，分频比由 WDTC 寄存器中的 WS2~WS0 位来决定。

21.2 看门狗定时器控制寄存器

WDTC 寄存器用于选择溢出周期和控制 WDT 功能的使能 / 除能。

- **WDTC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	WDTEN	WS2	WS1	WS0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	1	1	1	1

Bit 7~4　未定义，读为“0”

Bit 3　**WDTEN**：WDT 功能使能控制
0：除能
1：使能

Bit 2~0　**WS2~WS0**：WDT 溢出周期选择位
000：$[(2^8-2^0)\sim2^8]/f_{LIRC}$
001：$[(2^9-2^1)\sim2^9]/f_{LIRC}$
010：$[(2^{10}-2^2)\sim2^{10}]/f_{LIRC}$
011：$[(2^{11}-2^3)\sim2^{11}]/f_{LIRC}$
100：$[(2^{12}-2^4)\sim2^{12}]/f_{LIRC}$
101：$[(2^{13}-2^5)\sim2^{13}]/f_{LIRC}$
110：$[(2^{14}-2^6)\sim2^{14}]/f_{LIRC}$
111：$[(2^{15}-2^7)\sim2^{15}]/f_{LIRC}$
这三位控制 WDT 时钟源的分频比，从而实现对 WDT 溢出周期的控制。

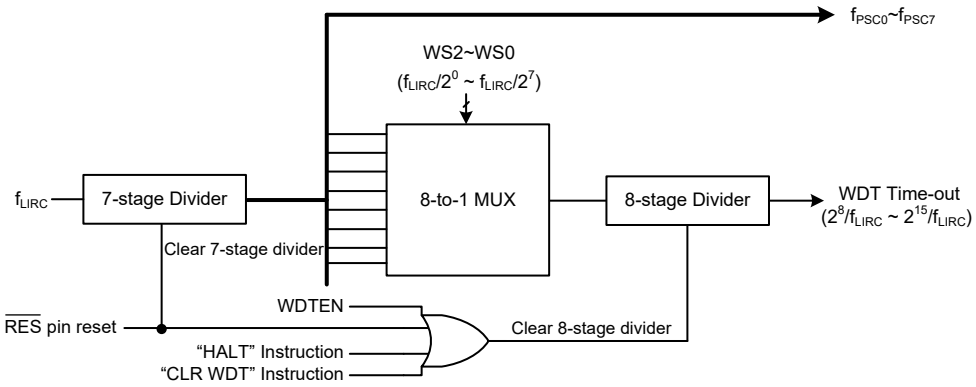
21.3 看门狗定时器操作

当 WDT 溢出时，它产生一个单片机复位的动作。这也就意味着正常工作期间，用户需在应用程序中看门狗溢出前有策略地清看门狗定时器以防止其产生复位，可使用清除看门狗指令实现。无论什么原因，程序失常跳转到一个未知的地址或进入一个死循环，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。另外，WDTC 寄存器中的 WDTEN 位可用来使能 / 除能看门狗定时器。

程序正常运行时，WDT 溢出将导致单片机复位，并置位状态标志位 TO。若系统处于休眠或空闲模式，当 WDT 发生溢出时，状态寄存器中的 TO 应置位，仅 PC 和堆栈指针复位。有四种方法可以用来清除 WDT 的内容。第一种是通过软件清除指令，第二种是通过“HALT”指令，第三种是外部硬件复位 (外部复位引脚低电平)。第四种是通过 WDTEN 位清零以除能看门狗定时器。

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

当设置分频比为 2^{15} 时，溢出周期最大。时钟源为 32kHz LIRC 振荡器，分频比为 2^{15} 时最大溢出周期约 1s，分频比为 2^{8} 时最小溢出周期约 8ms。



看门狗定时器

22. 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，它会被清为零，使得单片机从最低的程序存储器地址开始执行程序。

除了上电复位外，即使单片机处于正常工作状态，有些情况的发生也会迫使单片机复位。譬如当单片机上电后已经开始执行程序，$\overline{\text{RES}}$ 引脚被强制拉为低电平。这种复位为正常操作复位，单片机中只有一些寄存器受影响，而大部分寄存器不会改变，在复位引脚恢复至高电平后，单片机可以正常运行。

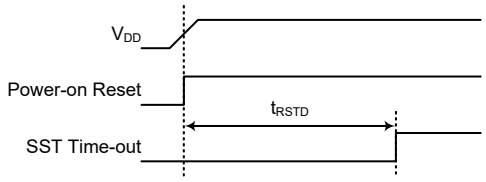
另一种复位为低电压复位即 LVR 复位，在电源供应电压低于 LVR 设置值时，系统会产生 LVR 复位，这种复位与 $\overline{\text{RES}}$ 脚拉低复位方式相似。另一种复位为看门狗溢出单片机复位，不同方式的复位操作会对寄存器产生不同的影响。不同方式的复位操作会对寄存器产生不同的影响。

22.1 复位功能

单片机的几种内部复位和外部复位方式将在此处做具体介绍。

22.1.1 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。



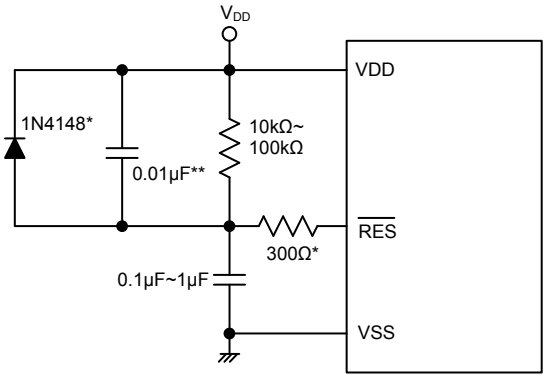
上电复位时序图

22.1.2 $\overline{\text{RES}}$ 引脚复位

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

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

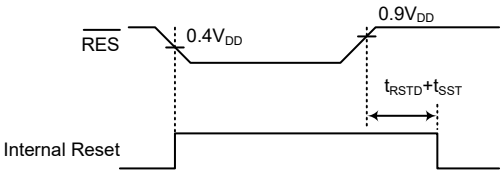
当系统在较强干扰的场合工作时，建议使用增强型的复位电路，如下图所示。



注：“*”表示建议加上此元件以加强静电保护。
“**”表示建议在电源有较强干扰场合加上此元件。

外部 $\overline{\text{RES}}$ 电路

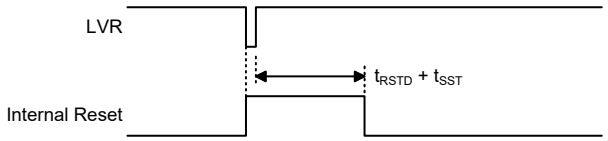
$\overline{\text{RES}}$ 引脚通过外部硬件强迫拉至低电平时，此种复位形式即会发生。这种复位方式和其它的复位方式一样，程序计数器会被清除为零且程序从头开始执行。



$\overline{\text{RES}}$ 复位时序图

22.1.3 低电压复位 – LVR

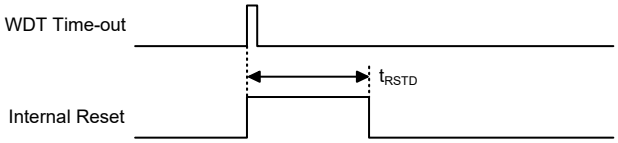
单片机具有低电压复位电路，用来监测它的电源电压，可通过配置选项进行选择。例如在更换电池的情况下，单片机供应的电压可能会在 0.9V~V_{LVR} 之间，这时 LVR 将会自动复位单片机。LVR 包含以下的规格：有效的 LVR 信号，即在 0.9V~V_{LVR} 的低电压状态的时间，必须超过 LVR 电气特性中 t_{LVR} 参数的值。如果低电压存在不超过 t_{LVR} 参数的值，则 LVR 将会忽略它且不会执行复位功能。V_{LVR} 参数值可通过配置选项进行设定。



低电压复位时序图

22.1.4 正常运行时看门狗溢出复位

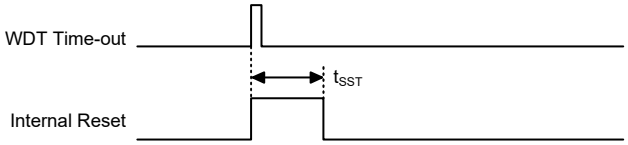
在快速模式或低速模式发生看门狗溢出复位时，看门狗溢出标志位 TO 将被设为“1”。



正常运行时看门狗溢出复位时序图

22.1.5 休眠或空闲时看门狗溢出复位

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



休眠或空闲时看门狗溢出复位时序图

22.2 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	快速模式或低速模式时的 $\overline{RES}$ 复位或 LVR 复位
1	u	快速模式或低速模式时的 WDT 溢出复位
1	1	空闲或休眠模式时的 WDT 溢出复位

“u”代表不改变

在单片机上电复位之后，各功能单元初始化的情形，列于下表。

项目	复位后情况
程序计数器	清除为零
中断	所有中断被除能
看门狗定时器，时基	都清为零，且 WDT 重新计数
定时 / 事件计数器	定时 / 事件计数器关闭
输入 / 输出口	I/O 口设为输入模式
堆栈指针	堆栈指针指向堆栈顶端

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

寄存器	上电复位	WDT 溢出 (正常操作)	$\overline{\text{RES}}$ 复位 (正常运行)	WDT 溢出 (空闲 / 休眠)
IAR0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
IAR1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	-xxx xxxx	-uuu uuuu	-uuu uuuu	-uuu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--11 uuuu
INTEG	---- 0000	---- 0000	---- 0000	---- uuuu
WDTC	---- 1111	---- 1111	---- 1111	---- uuuu
TBC	0-00 -000	0-00 -000	0-00 -000	u-uu -uuu
SCC	111- 0000	111- 0000	111- 0000	uuu- uuuu
PAS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAS1	-0-0 00-0	-0-0 00-0	-0-0 00-0	-u-u uu-u
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	-000 -000	-000 -000	-000 -000	-uuu- -uuu
TMR1C	00-0 1000	00-0 1000	00-0 1000	uu-u uuuu
TMR1L	0000 0000	0000 0000	0000 0000	uuuu uuuu
TMR1H	0000 0000	0000 0000	0000 0000	uuuu uuuu
IFS	0000 0000	0000 0000	0000 0000	uuuu uuuu
SADC0	000- 0000	000- 0000	000- 0000	uuu- uuuu
SADC1	---0 0000	---0 0000	---0 0000	---u uuuu
SADO	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
PB	--11 1111	--11 1111	--11 1111	--uu uuuu

寄存器	上电复位	WDT 溢出 (正常操作)	$\overline{\text{RES}}$ 复位 (正常运行)	WDT 溢出 (空闲 / 休眠)
PBC	--11 1111	--11 1111	--11 1111	--uu uuuu
PBPU	--00 0000	--00 0000	--00 0000	--uu uuuu
PBS0	-000 0000	-000 0000	-000 0000	-uuu uuuu
PC	---- 1111	---- 1111	---- 1111	---- uuuu
PCC	---- 1111	---- 1111	---- 1111	---- uuuu
PCPU	---- 0000	---- 0000	---- 0000	---- uuuu
PCS0	---- 0000	---- 0000	---- 0000	---- uuuu
SCOMC	0000 ----	0000 ----	0000 ----	uuuu ----
PBS1	---- -0-0	---- -0-0	---- -0-0	---- -u-u
OCR	---- 00--	---- 00--	---- 00--	---- uu--
ODL	0000 0000	0000 0000	0000 0000	uuuu uuuu
ODH	-000 0000	-000 0000	-000 0000	-uuu uuuu
TMR0C/PWMC	00-0 0000	00-0 0000	00-0 0000	uu-u uuuu
TMR0/ PWMADATA	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWMBDATA	0000 0000	0000 0000	0000 0000	uuuu uuuu
PWMCDATA	0000 0000	0000 0000	0000 0000	uuuu uuuu

注：“u”表示不改变；
“x”表示未知；
“-”表示未定义

23. 输入 / 输出端口

芯群单片机的输入 / 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。

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

寄存器名称	位							
	7	6	5	4	3	2	1	0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PAWU	PAWU7	PAWU6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PB	—	—	PB5	PB4	PB3	PB2	PB1	PB0
PBC	—	—	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PBPU	—	—	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PC	—	—	—	—	PC3	PC2	PC1	PC0
PCC	—	—	—	—	PCC3	PCC2	PCC1	PCC0
PCPU	—	—	—	—	PCPU3	PCPU2	PCPU1	PCPU0

“—”：未定义，读为“0”

I/O 逻辑功能寄存器列表

23.1 上拉电阻

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

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

- **PxPU 寄存器**

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

PxPUn：I/O Px 口上拉电阻控制位
0：除能
1：使能

PxPUn 位用于控制对应引脚的上拉电阻功能。这里的 x 可以是端口 A、B 或 C。但是，每个 I/O 端口实际有效位可能不同。

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

- **PAWU 寄存器**

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

PAWUn：PA 口引脚唤醒功能控制位
0：除能
1：使能

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

- **PxC 寄存器**

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

PxCn：I/O PA 口引脚类型选择位
0：输出
1：输入
PxCn 位用于控制对应引脚的状态类型。这里的 x 可以是端口 A、B 或 C。但是，每个 I/O 端口实际有效位可能不同。

23.4 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应用设计，而引脚的多功能将会解决很多此类问题。此外，这些引多功能引脚功能可以通过一系列寄存器进行设定。

23.4.1 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 A 输出功能选择寄存器，记为 PxSn，以及输入功能选择寄存器，记为 IFS，可以用来选择共用引脚的特定功能。

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

寄存器名称	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PAS0	PAS07	PAS06	PAS05	PAS04	PAS03	PAS02	PAS01	PAS00
PAS1	—	PAS16	—	PAS14	PAS13	PAS12	—	PAS10
PBS0	—	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
PBS1	—	—	—	—	—	PBS12	—	PBS10
PCS0	—	—	—	—	PCS03	PCS02	PCS01	PCS00
IFS	TC1PS1	TC1PS0	TC0PS1	TC0PS0	INT1PS1	INT1PS0	INT0PS1	INT0PS0

引脚共用功能选择寄存器列表

- **PAS0 寄存器**

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

Bit 7~6 **PAS07~PAS06**：PA3 引脚共用功能选择
00：PA3/INT0
01：SCOM5
10：AN0
11：PA3/INT0

Bit 5~4 **PAS05~PAS04**：PA2 引脚共用功能选择
00：PA2/TC0
01：SCOM6
10：AN1
11：PA2/TC0

Bit 3~2 **PAS03~PAS02**：PA1 引脚共用功能选择
00：PA1
01：PWMAOB
10：VREF
11：AN2

Bit 1~0 **PAS01~PAS00**：PA0 引脚共用功能选择
00：PA0/INT1
01：PWMAO
10：AN3
11：PA0/INT1

- **PAS1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	PAS16	—	PAS14	PAS13	PAS12	—	PAS10
R/W	—	R/W	—	R/W	R/W	R/W	—	R/W
POR	—	0	—	0	0	0	—	0

Bit 7　　未定义，读为“0”

Bit 6　　**PAS16**：PA7 引脚共用功能选择
0：PA7/$\overline{\text{RES}}$/VPP
1：PWMAO

Bit 5　　未定义，读为“0”

Bit 4　　**PAS14**：PA6 引脚共用功能选择
0：PA6/TC1/INT1
1：PWMCO

Bit 3~2　　**PAS13~PAS12**：PA5 引脚共用功能选择
00：PA5
01：PWMBO
10：AN9
11：PA5

Bit 1　　未定义，读为“0”

Bit 0　　**PAS10**：PA4 引脚共用功能选择
0：PA4/TC1
1：PWMAO

- **PBS0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	PBS06	PBS05	PBS04	PBS03	PBS02	PBS01	PBS00
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　　**PBS06**：PB3 引脚共用功能选择
0：PB3/INT0
1：SCOM3

Bit 5~4　　**PBS05~PBS04**：PB2 引脚共用功能选择
00：PB2/INT0
01：SCOM2
10：AN8
11：PB2/INT0

Bit 3~2　　**PBS03~PBS02**：PB1 引脚共用功能选择
00：PB1/INT1
01：SCOM1
10：AN7
11：PWMCO

Bit 1~0　　**PBS01~PBS00**：PB0 引脚共用功能选择
00：PB0/TC0
01：SCOM0
10：AN6
11：PWMBO

- **PBS1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	PBS12	—	PBS10
R/W	—	—	—	—	—	R/W	—	R/W
POR	—	—	—	—	—	0	—	0

Bit 7~3　未定义，读为“0”

Bit 2　**PBS12**：PB5 引脚共用功能选择
0：PB5
1：PWMAO

Bit 1　未定义，读为“0”

Bit 0　**PBS10**：PB4 引脚共用功能选择
0：PB4/TC1/INT1
1：SCOM4

- **PCS0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	PCS03	PCS02	PCS01	PCS00
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　**PCS03**：PC2 引脚共用功能选择
0：PC2/TC0
1：PWMAO

Bit 2　**PCS02**：PC1 引脚共用功能选择
0：PC1/INT0/TC1
1：AN5

Bit 1~0　**PCS01~PCS00**：PC0 引脚共用功能选择
00：PC0
01：AN4
10：VREF
11：PC0

- **IFS 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	TC1PS1	TC1PS0	TC0PS1	TC0PS0	INT1PS1	INT1PS0	INT0PS1	INT0PS0
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　**TC1PS1~TC1PS0**：TC1 输入源引脚选择
00：PA4
01：PA6
10：PC1
11：PB4

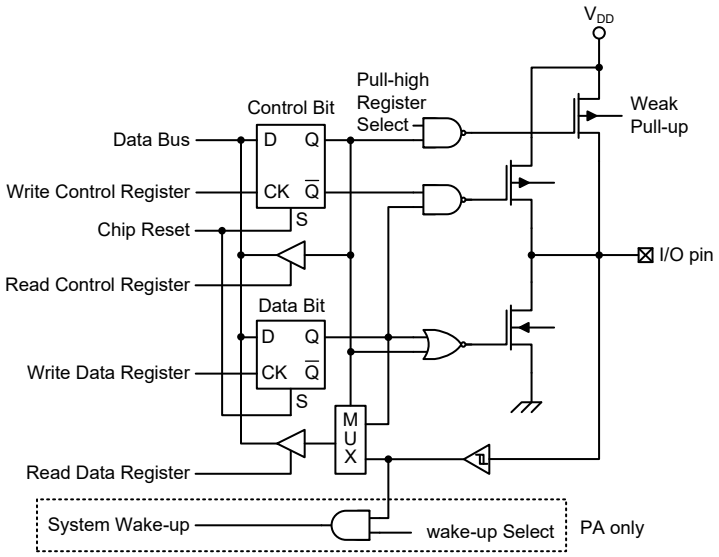
Bit 5~4　**TC0PS1~TC0PS0**：TC0 输入源引脚选择
00：PA2
01：PB0
10：PC2
11：PC3

Bit 3~2　　**INT1PS1~INT1PS0**：INT1 输入源引脚选择
00：PB1
01：PA0
10：PB4
11：PA6

Bit 1~0　　**INT0PS1~INT0PS0**：INT0 输入源引脚选择
00：PB2
01：PA3
10：PC1
11：PB3

23.5 输入 / 输出引脚结构

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



逻辑功能输入 / 输出端口结构

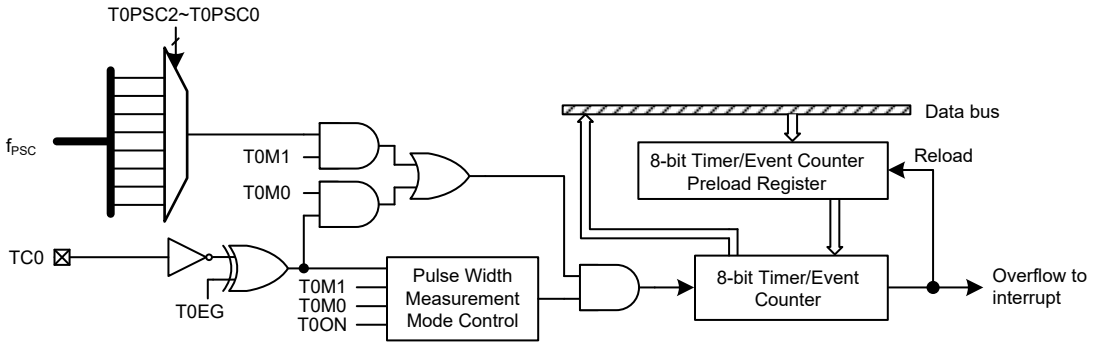
23.6 编程注意事项

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

24. 8-bit 定时 / 事件计数器

定时 / 事件计数器在任何单片机中都是一个很重要的部分，便于程序设计者实现和时间有关的功能。该单片机包含一个 8 位可编程向上计数的定时 / 事件计数器，其时钟源可由外部输入或内部提供。该计数器包括 4 种工作模式，可以当作一个普通定时器、外部事件计数器、脉冲宽度测量和 PWM 模式功能。



8-bit 定时 / 事件计数器结构

24.1 定时 / 事件计数器输入时钟源

定时 / 事件计数器的时钟源可有多种选择，可以是内部时钟，也可以是外部引脚。当定时 / 事件计数器工作在定时器模式、脉冲宽度测量模式和 PWM 模式时，使用内部时钟作为时钟源。定时 / 事件计数器的内部时钟可来自时基分频器，由定时 / 事件计数器控制寄存器 TMR0C 中的 T0PSC2~T0PSC0 位来确定。

当定时 / 事件计数器在事件计数器模式时，使用外部时钟源，时钟源由外部 TC0 引脚提供。每次外部引脚由高电平到低电平或由低电平到高电平 (由 T0EG 位决定) 进行转换时，计数器增加一。

24.2 定时 / 事件计数器寄存器

定时 / 事件计数器相关的寄存器有多种。第一种是 TMR0 寄存器，其包含了定时 / 事件计数器的实际值且可以预加载初始值。写入 TMR0 寄存器将把指定的数据传输到定时器 / 事件计数器。读取 TMR0 寄存器将读取定时 / 事件计数器的内容。第二种是 TMR0C 控制寄存器，用来定义定时 / 事件计数器的工作模式、控制使能或除能、选择有效边沿。另外四个寄存器用来控制脉宽调制通道的所有操作，三个数据寄存器 PWMADATA、PWMBDATA 和 PWMCDATA 以及一个控制寄存器 PWMC。应注意，PWMC 和 TMR0C 寄存器有相同的寄存器地址，PWMADATA 和 TMR0 寄存器也有相同的寄存器地址。

寄存器名称	位							
	7	6	5	4	3	2	1	0
TMR0	D7	D6	D5	D4	D3	D2	D1	D0
TMR0C	T0M1	T0M0	—	T0ON	T0EG	T0PSC2	T0PSC1	T0PSC0
PWMC	T0M1	T0M0	—	PWMDIV2	PWMDIV1	PWMDIV0	PWMSEL	PWMEN
PWMADATA	PWMAD7	PWMAD6	PWMAD5	PWMAD4	PWMAD3	PWMAD2	PWMAD1	PWMAD0
PWMBDATA	PWMBD7	PWMBD6	PWMBD5	PWMBD4	PWMBD3	PWMBD2	PWMBD1	PWMBD0
PWMCDATA	PWMCD7	PWMCD6	PWMCD5	PWMCD4	PWMCD3	PWMCD2	PWMCD1	PWMCD0

定时 / 事件计数器寄存器列表

24.2.1 定时器寄存器 – TMR0

定时器寄存器 TMR0 用于存放实际定时器值。在用作内部定时且收到一个内部计数时钟脉冲或用作外部计数且外部定时 / 事件计数器引脚发生状态跳变时，此寄存器对的值将会加一。定时器将从预载寄存器所载入的值开始计数，8-bit 定时 / 事件计数器计数到 FFH 时，定时器溢出且会产生一个内部中断信号。定时器随后重新载入预载寄存器的值并继续计数。

注意，为使定时器计数范围达到最大 FFH，预载寄存器需要先清为零。定时 / 事件计数器在关闭条件下，写数据到预载寄存器，会立即写入实际的计数器中。若定时 / 事件计数器使能且正在计数，此时写入预载寄存器的任何新数据将保留在预载寄存器中，直到溢出发生时才被写入实际计数器。

- **TMR0 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**：定时器预载寄存器字节

24.2.2 定时器控制寄存器 – TMR0C

芯群微控制器的定时器 / 事件计数器可工作于四种模式，由控制寄存器相关位决定。

定时器控制寄存器 TMR0C 配合相应的定时器寄存器可控制定时 / 事件计数器的全部操作。在使用定时器之前，需要先正确地设定定时器控制寄存器，以便保证定时器能正确操作，而这个过程通常在程序初始化期间完成。

要选择计时器工作于四种模式中的哪一种，即 PWM 模式、定时器模式、事件计数器模式或脉冲宽度测量模式，定时器控制寄存器中的 T0M1~T0M0 位必须设置为所需的逻辑值。定时器控制寄存器的 T0ON 位用于定时器开关控制，该位设定为逻辑高时，计数器开始计数，而清零时则停止计数。当使用内部时钟源 f_{PSC} 时，可以通过设置控制寄存器中的 T0PSC2~T0PSC0 位进行选择。如果使用外部时钟源，则内部时钟设定位的选择无效。如果定时 / 事件计数器工作在事件计数器模式或脉冲宽度测量模式，TMR0C 寄存器的 T0EG 位可用来选择有效触发边沿。

- **TMR0C 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	T0M1	T0M0	—	T0ON	T0EG	T0PSC2	T0PSC1	T0PSC0
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 7~6　　**T0M1~T0M0**：定时 / 事件计数器工作模式选择
00：PWM 模式
01：事件计数器模式
10：定时器模式
11：脉冲宽度测量模式

Bit 5　　未定义，读为“0”

Bit 4　　**T0ON**：定时 / 事件计数器计数使能控制
0：除能
1：使能
注意此位在 PWM 模式下无效。

Bit 3 **T0EG**：定时 / 事件计数器有效边沿选择
事件计数器模式
0：在上升沿计数
1：在下降沿计数
脉冲宽度测量模式
0：在下降沿启动计数，在上升沿停止计数
1：在上升沿启动计数，在下降沿停止计数
PWM 模式
未使用

Bit 2~0 **T0PSC2~T0PSC0**：定时 / 事件计数器内部时钟选择
000：$f_{PSC}/2^0$
001：$f_{PSC}/2^1$
010：$f_{PSC}/2^2$
011：$f_{PSC}/2^3$
100：$f_{PSC}/2^4$
101：$f_{PSC}/2^5$
110：$f_{PSC}/2^6$
111：$f_{PSC}/2^7$

24.2.3 定时器 PWM 模式寄存器 – PWMC, PWMADATA, PWMBDATA, PWMCDATA

脉宽调制通道的所有操作是通过四个寄存器来控制的，三个数据寄存器 PWMADATA、PWMBDATA 和 PWMCDATA 以及一个控制寄存器 PWMC。应注意的是，寄存器 PWMC 和寄存器 TMR0C 共用同一个寄存器地址，另外 PWMADATA 和 TMR0 寄存器也有相同的寄存器地址。

- **PWMC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	T0M1	T0M0	—	PWMDIV2	PWMDIV1	PWMDIV0	PWMSEL	PWMEN
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 7~6 **T0M1~T0M0**：定时 / 事件计数器工作模式选择
00：PWM 模式
01：事件计数器模式
10：定时器模式
11：脉冲宽度测量模式

Bit 5 未定义，读为“0”

Bit 4~2 **PWMDIV2~PWMDIV0**：f_{DIV} 频率选择
000：$f_{DIV}=f_{SYS}$
001：$f_{DIV}=f_{SYS}/2$
010：$f_{DIV}=f_{SYS}/3$
011：$f_{DIV}=f_{SYS}/4$
100：$f_{DIV}=f_{SYS}/8$
101：$f_{DIV}=f_{SYS}/16$
110：$f_{DIV}=f_{SYS}/32$
111：$f_{DIV}=f_{SYS}/64$

Bit 1 **PWMSEL**：PWM 模式选择
0：(6+2) 位模式
1：(7+1) 位模式

Bit 0 **PWMEN**：PWM 使能控制
0：除能
1：使能

注：1. 当 PWMEN 位清零除能 PWM 功能时，内部 PWMmO 信号将被强制拉低。当共用引脚选择作为 PWMmO 输出且 PWMEN 位清零时，外部 PWMmO 引脚为浮空状态 (m=A、B 或 C)。

2. 当 PWMEN 位设置为高时，第一个 PWM 调制周期的周期和占空比可能与预期不符合。在第一个 PWM 周期后，PWM 输出正常。
3. PWMAOB 为 PWMAO 的反相输出引脚，且可产生互补式输出。

- **PWMmDATA 寄存器 (m=A、B 或 C)**

Bit	7	6	5	4	3	2	1	0
Name	PWMmD7	PWMmD6	PWMmD5	PWMmD4	PWMmD3	PWMmD2	PWMmD1	PWMmD0
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 **PWMmD7~PWMmD0**：PWM 输出占空比 PWMmD bit 7 ~ bit 0

注意，修改 PWMmDATA 寄存器值会立即反映在 PWMmO 输出信号上。因此，在当前的 PWM 调制周期中，PWM 新旧数据的相接会导致占空比与设定值不一致的现象。此现象只会维持一个 PWM 周期，从下一个新的 PWM 周期开始将恢复与 PWMmD 设定值相同的 PWM 占空比。

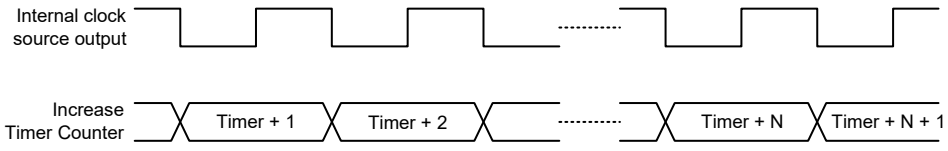
24.3 定时 / 事件计数器工作模式

定时 / 事件计数器可在四种工作模式下运行，即 PWM 模式、定时器模式、事件计数器模式或脉冲宽度测量模式。使用 TMR0C 寄存器中的 T0M1 和 T0M0 位选择工作模式。

24.3.1 定时器模式

为使定时 / 事件计数器工作在定时器模式，TMR0C 寄存器中的 T0M1 和 T0M0 位需要设置成“10”。在这个模式下，定时 / 事件计数器可以用来测量固定时间间隔，当定时 / 事件计数器发生溢出时，就会产生一个内部中断信号。

在定时器模式下，内部时钟 f_{PSC} 作为定时器时钟源。f_{PSC} 时钟源来自时基分频器，由定时器控制寄存器 TMR0C 中的 T0PSC2~T0PSC0 位选择。TMR0C 寄存器中的 T0ON 位需要置高以使能定时器。每次内部时钟发生由高到低的电平转换时，定时器值加一；当计数器的值达到 8 位最大值 FFH 时将溢出，会产生中断信号且定时器会重新载入预载寄存器的值，然后继续计数。在此模式下，即使单片机处于空闲 / 休眠模式，若所选内部时钟还有效且发生定时器溢出，将产生一个定时器中断请求，可作为一种唤醒源。



定时器模式时序图

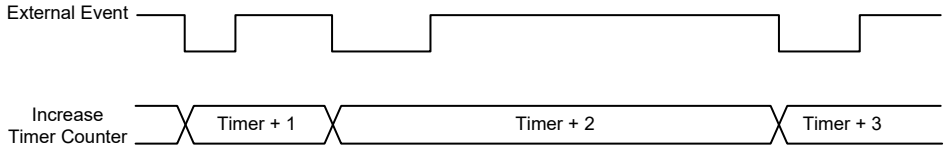
24.3.2 事件计数模式

为使定时 / 事件计数器工作在事件计数器模式，TMR0C 寄存器中的 T0M1 和 T0M0 位需要设置成“01”。在这个模式下可以通过定时 / 事件计数器来记录发生在 TC0 引脚上的的外部逻辑变化事件的次数。

在事件计数器模式下，外部定时器 TC0 引脚作为定时 / 事件计数器的时钟源。在设置完定时器控制寄存器其它位后，TMR0C 寄存器中的 T0ON 位需要置高以使能定时 / 事件计数器。若有效边沿选择位 T0EG 为低，则每次外部 TC0 引脚接收到由低到高的电平转换时，计数器值加一。若 T0EG 位为高，则每次 TC0 引脚接收到由高到低的电平转换时，计数器值加一。当计数器计满时将溢出，并产生中断请求信号且定时 / 事件计数器会重新载入预载寄存器的值，然后继续计数。

由于外部定时器引脚 TC0 与其它功能共用引脚，需预先设置相关引脚共用功能选择寄存器以选择 TC0 引脚功能。此外，该引脚还需通过 I/O 端口控制寄存器设置为输入引脚。

注意，在事件计数器模式下，即使单片机处于空闲 / 休眠模式，定时 / 事件计数器将继续对外部 TC0 引脚上发生的的逻辑事件变化进行计数。因此，当计数器溢出时，将产生一个定时器中断请求，可作为一种唤醒源。



事件计数器模式时序图 (T0EG=1)

24.3.3 脉冲宽度测量模式

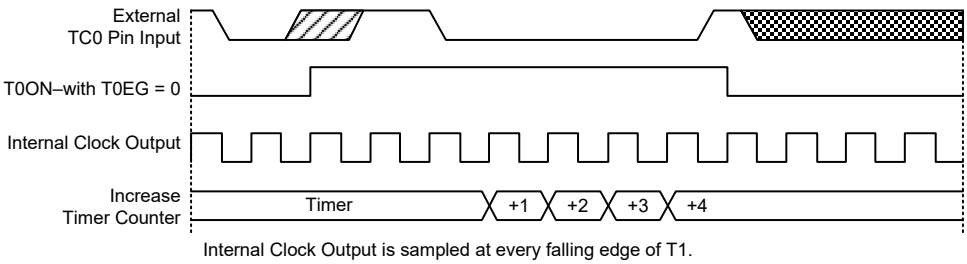
为使定时 / 事件计数器工作在脉冲宽度测量模式，TMR0C 寄存器中的 T0M1 和 T0M0 需要设置为“11”。在这个模式下，定时 / 事件计数器可用于测量外部定时器引脚上的外部脉冲宽度。

在脉冲宽度测量模式下，内部时钟 f_{PSC} 作为定时器时钟源。f_{PSC} 时钟源来自时基分频器，由定时器控制寄存器中的 T0PSC2~T0PSC0 位选择。在设置完定时器控制寄存器 TMR0C 中的其它位后，T0ON 位需要置高以使能定时 / 事件计数器。然而，只有在 TC0 引脚上接收到有效的逻辑转换边沿时，定时 / 事件计数器才真正开始计数。

当有效边沿选择位 T0EG 设置为低时，每次 TC0 引脚接收到由高到低的电平转换时，定时 / 事件计数器将在内部选定的时钟源下开始计数，直到 TC0 引脚回到它原来的高电平。此时使能位将自动清零以停止计数。而当 T0EG 位为高时，每次外部定时器引脚接收到由低到高的电平转换时定时 / 事件计数器将开始计数，直到 TC0 引脚回到它原来的低电平。同样使能位将自动清零以停止计数。注意，在脉冲宽度测量模式中，当 TC0 引脚上的外部控制信号回到它原来的电平时，使能位将自动清零。而在事件计数模式或定时模式，使能位只能在程序控制下清零。

可以通过程序读取定时 / 事件计数器当前值，从而获得 TC0 引脚上接收信号的脉冲宽度。由于使能位已被复位为零，任何出现在外部定时器引脚上的电平变化将被忽略。直到使能位被程序重新置高，定时器才可开始重新测量外部脉冲。通过这种方式可以很容易地实现单次脉冲宽度测量。注意，在这种模式下，定时 / 事件计数器是通过外部定时器引脚上的逻辑转换来控制，而非通过逻辑电平。当定时 / 事件计数器计满时将溢出，并产生中断请求信号且定时 / 事件计数器会重新载入预载寄存器的值，然后继续计数。

由于外部定时器引脚 TC0 与其它功能共用引脚，需预先设置相关引脚共用功能选择寄存器以选择 TC0 引脚功能。此外，该引脚还需通过 I/O 端口控制寄存器设置为输入引脚。注意，在脉冲宽度测量模式下，即使单片机处于空闲 / 休眠模式，定时 / 事件计数器将继续对外部 TC0 引脚上发生的的逻辑事件变化进行计数。因此，当计数器溢出时，将产生一个定时器中断请求，可作为一种唤醒源。



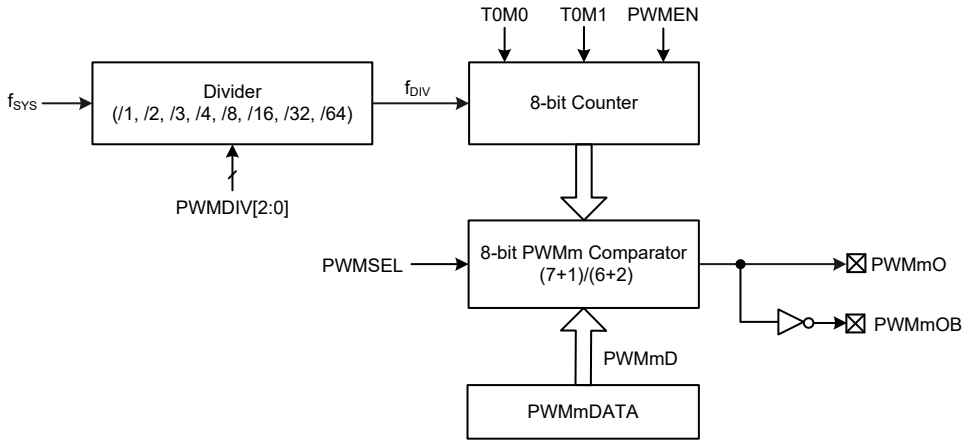
脉冲宽度测量模式时序图 (T0EG=0)

24.3.4 PWM 模式

为使定时 / 事件计数器工作在脉冲宽度调制模式，PWMC 寄存器中的 T0M1 和 T0M0 需要设置为“00”。在这个模式下，定时 / 事件计数器可用于时钟源来自内部选择的时钟源时产生 PWM 波形。

脉冲宽度调制通道配有 PWMC 和 PWMmDATA 寄存器。该 PWM 通道有一个数据寄存器，即 PWMmDATA，其内容是一个 8 位的数据，简称为 PWMmD。PWMmD 表示输出波形中每个调制周期的总的占空比。为了提高 PWM 调制频率，每个调制周期被调制成 2 个或 4 个独立的调制子区段，分别对应 (7+1) 位模式和 (6+2) 位模式。PWM 计数器的时钟频率为 f_{DIV}，来自系统时钟 f_{SYS} 或其分频。可以通过设置 PWMC 寄存器来选择 PWM 通道所需的模式、时钟源以及使能 / 除能控制。注意，当使用 PWM 时，只要将所需的值写入 PWMmDATA 寄存器并通过 PWMC 寄存器设置所需模式、时钟源和使能 / 除能控制，单片机内部电路即自动完成 PWM 各子调制周期的划分并输出 PWM 信号。注意，PWMAOB 为 PWMAO 的反相输出引脚，可产生互补式输出，并为连接的接口 (如蜂鸣器) 提供更多的电源。

将原始调制周期分成 2 个或 4 个子周期的方法，使产生更高的 PWM 频率成为可能，这样可以提供更广泛的应用。使用者需要理解 PWM 频率与 PWM 调制频率的不同之处。PWM 时钟为 f_{DIV}，当 PWM 值为 8-bit 时，整个 PWM 周期的频率为 $f_{DIV}/256$。(7+1) 位模式下的 PWM 调制频率为 $f_{DIV}/128$，而 (6+2) 位模式下的 PWM 调制频率为 $f_{DIV}/64$。



注：m=A、B 或 C，且仅有 PWMAO 有反相输出引脚 PWMAOB。

PWM 模式方框图

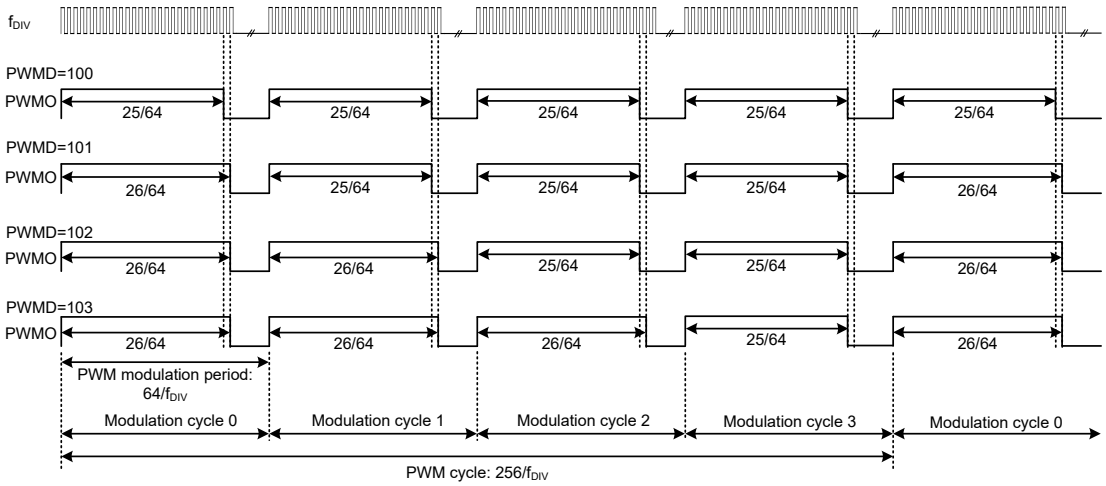
(6+2) 位 PWM 模式调制

在 (6+2) 位模式中，一个 PWM 周期又被分成 4 个调制周期，称为调制周期 0 ~ 调制周期 3。每个子周期包含 64 个 PWM 输入时钟周期。在这个模式下，PWMmD 被分成两个部分。第一部分包括 PWMmD 的 bit 7 ~ bit 2 位，表示 DC 值，第二部分为 PWMmD 的 bit 1 ~ bit 0 位，表示 AC 值。下表总结了在 (6+2) 位模式下 PWM 输出信号的调制频率、调制周期占空比、PWM 周期频率和 PWM 周期占空比的概况。

调制频率	调制周期 i	调制周期占空比		PWM 周期频率	PWM 周期占空比
$f_{DIV}/64$	i=0~3	i<AC	(DC+1)/64	$f_{DIV}/256$	PWMmD/256
		i ≥AC	DC/64		

(6+2) 位 PWM 模式概况

下图为 (6+2) 位模式 PWM 操作的波形图。重点是要了解如何将单个 PWM 周期分成 4 个的独立的调制周期 (序列号为 0~3)，并且了解 AC 值与 PWM 值的关系。PWM 输出波形如下图所示。



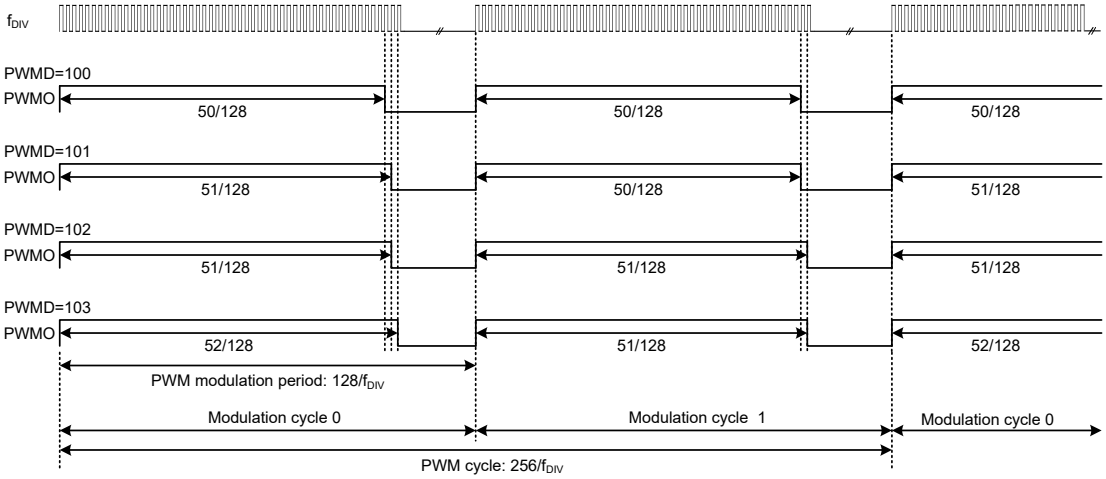
(6+2) 位 PWM 模式调制波形

(7+1) 位 PWM 模式调制

在 (7+1) 位模式中，一个 PWM 周期又被分成 2 个调制周期，称为调制周期 0 ~ 调制周期 1。每个子周期包含 128 个 PWM 输入时钟周期。在这个模式下，PWMmD 被分成两个部分。第一部分包括 PWMmD 的 bit 7 ~ bit 1 位，表示 DC 值，第二部分为 PWMmD 的 bit 0 位，表示 AC 值。下表总结了在 (7+1) 位模式下 PWM 输出信号的调制频率、调制周期占空比、PWM 周期频率和 PWM 周期占空比的概况。

调制频率	调制周期 i	调制周期占空比		PWM 周期频率	PWM 周期占空比
$f_{DIV}/128$	i=0~1	i<AC	(DC+1)/128	$f_{DIV}/256$	PWMmD/256
		i≥AC	DC/128		

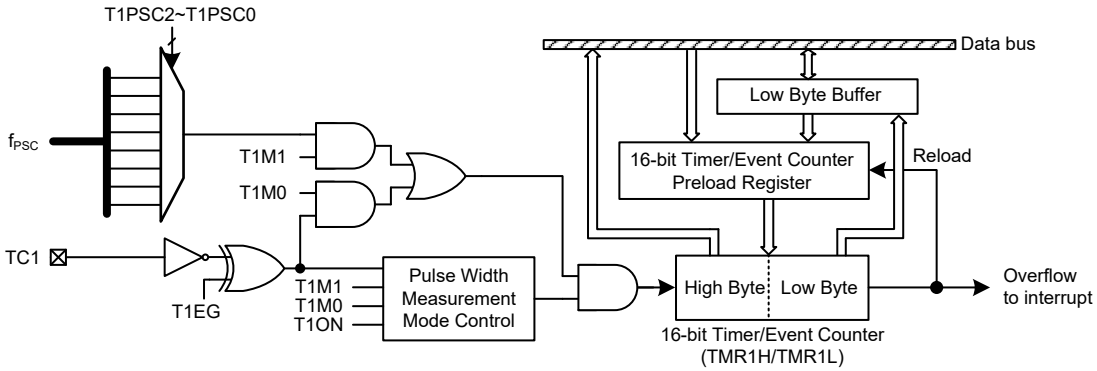
(7+1) 位 PWM 模式概况



(7+1) 位 PWM 模式调制波形

25. 16-bit 定时 / 事件计数器

定时 / 事件计数器在任何单片机中都是一个很重要的部分，便于程序设计者实现和时间有关的功能。该单片机包含一个 16 位可编程向上计数的定时 / 事件计数器，其时钟源可由外部输入或内部提供。该计数器包括 3 种工作模式，可以当作一个普通定时器、外部事件计数器或脉冲宽度测量使用。



16-bit 定时 / 事件计数器结构

25.1 定时 / 事件计数器输入时钟源

定时 / 事件计数器的时钟源可有多种选择，可以是内部时钟，也可以是外部引脚。当定时 / 事件计数器工作在定时器模式或脉冲宽度测量模式时，使用内部时钟作为时钟源。定时 / 事件计数器的内部时钟可来自时基分频器，由定时 / 事件计数器控制寄存器 TMR1C 中的 T1PSC2~T1PSC0 位来确定。

当定时 / 事件计数器在事件计数器模式时，使用外部时钟源，时钟源由外部 TC1 引脚提供。每次外部引脚由高电平到低电平或由低电平到高电平 (由 T1EG 位决定) 进行转换时，计数器增加一。

25.2 定时 / 事件计数器寄存器

定时 / 事件计数器相关的寄存器有两种。第一种是 TMR1L/TMR1H 寄存器对，其包含了定时 / 事件计数器的实际值且可以预加载初始值。写入 TMR1L/TMR1H 寄存器将把指定的数据传输到定时器 / 事件计数器。读取 TMR1L/TMR1H 寄存器将读取定时 / 事件计数器的内容。第二种是 TMR1C 控制寄存器，用来定义定时 / 事件计数器的工作模式、控制计数器的使能 / 除能以及选择有效边沿。

寄存器名称	位							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TMR1C	T1M1	T1M0	—	T1ON	T1EG	T1PSC2	T1PSC1	T1PSC0
TMR1L	D7	D6	D5	D4	D3	D2	D1	D0
TMR1H	D15	D14	D13	D12	D11	D10	D9	D8

定时 / 事件计数器寄存器列表

25.2.1 定时 / 事件计数器寄存器 – TMR1L, TMR1H

定时 / 事件计数器包含一个内部 16-bit 定时器，因此需要两个数据寄存器来存储定时器的值，即一个高字节寄存器 TMR1H 和一个低字节寄存器 TMR1L。在用作内部定时且收到一个内部计数时钟脉冲或用作外部计数且外部定时 / 事件计数器引脚发生状态跳变时，此寄存器对的值将会加一。定时 / 事件计数器将从预载寄存器所载入的值开始计数，16-bit

定时 / 事件计数器计数到 FFFFH 时，定时 / 事件计数器溢出且会产生一个内部中断信号。定时 / 事件计数器随后重新载入预载寄存器的值并继续计数。

注意，为了得到定时 / 事件计数器的最大计算范围 FFFFH，预载寄存器需要先清为零。定时 / 事件计数器在关闭条件下，写数据到预载寄存器，会立即写入实际的计数器中。若定时 / 事件计数器使能且正在计数，此时写入预载寄存器的任何新数据将保留在预载寄存器中，直到溢出发生时才被写入实际计数器。

写 TMR1L 寄存器时，数据仅被写入到一个内部低字节缓冲器，而当写 TMR1H 寄存器时，此高字节数据和低字节缓冲器中的值将分别被送入 TMR1H 和 TMR1L 寄存器中。因此定时 / 事件计数器预载寄存器将在每次写入 TMR1H 寄存器时被改变。读取 TMR1H 寄存器将把 TMR1H 和 TMR1L 寄存器中的值分别锁存到目标地址和低字节缓冲器中，读取 TMR1L 寄存器将读取到低字节缓冲器的值。

- **TMR1L 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：**定时 / 事件计数器预载寄存器低字节

- **TMR1H 寄存器**

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

Bit 7~0　　**D15~D8：**定时 / 事件计数器预载寄存器高字节

25.2.2 定时器控制寄存器 – TMR1C

定时器控制寄存器 TMR1C 配合相应的定时器寄存器可控制定时 / 事件计数器的全部操作。在使用定时器之前，需要先正确地设定定时器控制寄存器，以便保证定时器能正确操作，而这个过程通常在程序初始化期间完成。

要选择计时器工作于三种模式中的哪一种，即定时器模式、事件计数器模式或脉冲宽度测量模式，定时器控制寄存器中的 T1M1~T1M0 位必须设置为所需的逻辑值。定时器控制寄存器的 T1ON 位用于定时器开关控制，该位设定为逻辑高时，计数器开始计数，而清零时则停止计数。当使用内部时钟源 f_{PSC} 时，可以通过设置控制寄存器中的 T1PSC2~T1PSC0 位进行选择。如果使用外部时钟源，则内部时钟设定位的选择无效。如果定时 / 事件计数器工作在事件计数器模式或脉冲宽度测量模式，TMR1C 寄存器的 T1EG 位可用来选择有效触发边沿。

- **TMR1C 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	T1M1	T1M0	—	T1ON	T1EG	T1PSC2	T1PSC1	T1PSC0
R/W	R/W	R/W	—	R/W	R/W	R/W	R/W	R/W
POR	0	0	—	0	1	0	0	0

Bit 7~6 **T1M1~T1M0**：定时 / 事件计数器工作模式选择
00：未使用
01：事件计数器模式
10：定时器模式
11：脉冲宽度测量模式

Bit 5 未定义，读为“0”

Bit 4 **T1ON**：定时 / 事件计数器计数使能控制
0：除能
1：使能

Bit 3 **T1EG**：定时 / 事件计数器有效边沿选择
事件计数器模式
0：在上升沿计数
1：在下降沿计数
脉冲宽度测量模式
0：在下降沿启动计数，在上升沿停止计数
1：在上升沿启动计数，在下降沿停止计数

Bit 2~0 **T1PSC2~T1PSC0**：定时 / 事件计数器内部时钟选择
000：$f_{PSC}/2^0$
001：$f_{PSC}/2^1$
010：$f_{PSC}/2^2$
011：$f_{PSC}/2^3$
100：$f_{PSC}/2^4$
101：$f_{PSC}/2^5$
110：$f_{PSC}/2^6$
111：$f_{PSC}/2^7$

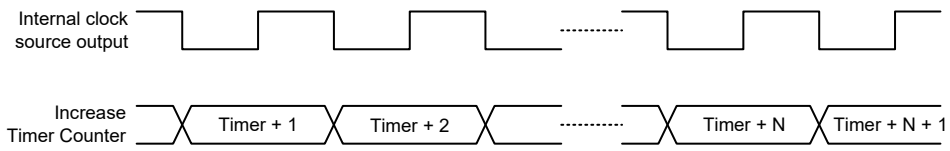
25.3 定时 / 事件计数器工作模式

定时 / 事件计数器可在三种工作模式下运行，即定时器模式、事件计数器模式或脉冲宽度测量模式。使用 TMR1C 寄存器中的 T1M1 和 T1M0 位选择工作模式。

25.3.1 定时器模式

为使定时 / 事件计数器工作在定时器模式，TMR1C 寄存器中的 T1M1 和 T1M0 位需要设置成“10”。在这个模式下，定时 / 事件计数器可以用来测量固定时间间隔，当定时 / 事件计数器发生溢出时，就会产生一个内部中断信号。

在定时器模式下，内部时钟 f_{PSC} 作为定时器时钟源。f_{PSC} 时钟源来自时基分频器，由定时器控制寄存器 TMR1C 中的 T1PSC2~T1PSC0 位选择。TMR1C 寄存器中的 T1ON 位需要置高以使能定时器。每次内部时钟发生由高到低的电平转换时，定时器值加一；应注意在该模式下，即使单片机进入空闲 / 休眠模式，若所选内部时钟仍开启，定时器将继续计数。当计数器的值达到 16 位最大值 FFFFH 时将溢出，会产生中断信号且定时器会重新载入预载寄存器的值，然后继续计数。在此模式下，即使单片机处于空闲 / 休眠模式，若所选内部时钟还有效且发生定时器溢出，将产生一个定时器中断请求，可作为一种唤醒源。



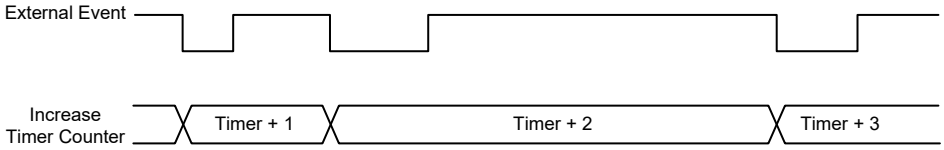
定时器模式时序图

25.3.2 事件计数模式

为使定时 / 事件计数器工作在事件计数器模式，TMR1C 寄存器中的 T1M1 和 T1M0 位需要设置成“01”。在这个模式下可以通过定时 / 事件计数器来记录发生在 TC1 引脚上的的外部逻辑变化事件的次数。

在事件计数器模式下，外部定时器 TC1 引脚作为定时 / 事件计数器的时钟源。在设置完定时器控制寄存器其它位后，TMR1C 寄存器中的 T1ON 位需要置高以使能定时 / 事件计数器。若有效边沿选择位 T1EG 为低，则每次外部 TC1 引脚接收到由低到高的电平转换时，计数器值加一。若 T1EG 位为高，则每次 TC1 引脚接收到由高到低的电平转换时，计数器值加一。当计数器计满时将溢出，并产生中断请求信号且定时 / 事件计数器会重新载入预载寄存器的值，然后继续计数。

由于外部定时器引脚 TC1 与其它功能共用引脚，需预先设置相关引脚共用功能选择寄存器以选择 TC1 引脚功能。此外，该引脚还需通过 I/O 端口控制寄存器设置为输入引脚。注意，在事件计数器模式下，即使单片机处于空闲 / 休眠模式，定时 / 事件计数器将继续对外部 TC1 引脚上发生的的逻辑事件变化进行计数。因此，当计数器溢出时，将产生一个定时器中断请求，可作为一种唤醒源。



事件计数器模式时序图 (T1EG=1)

25.3.3 脉冲宽度测量模式

为使定时 / 事件计数器工作在脉冲宽度测量模式，TMR1C 寄存器中的 T1M1 和 T1M0 需要设置为“11”。在这个模式下，定时 / 事件计数器可用于测量外部定时器引脚上的外部脉冲宽度。

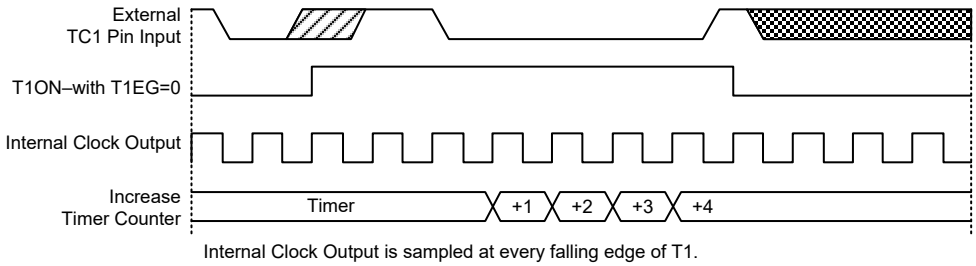
在脉冲宽度测量模式下，内部时钟 f_{PSC} 作为定时器时钟源。f_{PSC} 时钟源来自时基分频器，由定时器控制寄存器中的 T1PSC2~T1PSC0 位选择。在设置完定时器控制寄存器 TMR1C 中的其它位后，T1ON 位需要置高以使能定时 / 事件计数器。然而，只有在 TC1 引脚上接收到有效的逻辑转换边沿时，定时 / 事件计数器才真正开始计数。

当有效边沿选择位 T1EG 设置为低时，每次 TC1 引脚接收到由高到低的电平转换时定时 / 事件计数器将在内部选定的时钟源下开始计数，直到 TC1 引脚回到它原来的高电平。此时使能位将自动清零以停止计数。而当 T1EG 位为高时，每次外部定时器引脚接收到由低到高的电平转换时定时 / 事件计数器将开始计数，直到 TC1 引脚回到它原来的低电平。同样使能位将自动清零以停止计数。注意，在脉冲宽度测量模式中，当 TC1 引脚上的外部控制信号回到它原来的电平时，使能位将自动清零。而在其它两种模式，使能位只能在程序控制下清零。

可以通过程序读取定时 / 事件计数器当前值，从而获得 TC1 引脚上接收信号的脉冲宽度。由于使能位已被复位为零，任何出现在外部定时器引脚上的电平变化将被忽略。直到使能位被程序重新置高，定时器才可开始重新测量外部脉冲。通过这种方式可以很容易地

实现单次脉冲宽度测量。注意，在这种模式下，定时 / 事件计数器是通过外部定时器引脚上的逻辑转换来控制，而非通过逻辑电平。当定时 / 事件计数器计满时将溢出，并产生中断请求信号且定时 / 事件计数器会重新载入预载寄存器的值，然后继续计数。

由于外部定时器引脚 TC1 与其它功能共用引脚，需预先设置相关引脚共用功能选择寄存器以选择 TC1 引脚功能。此外，该引脚还需通过 I/O 端口控制寄存器设置为输入引脚。注意，在脉冲宽度测量模式下，即使单片机处于空闲 / 休眠模式，定时 / 事件计数器将继续对外部 TC1 引脚上发生的的逻辑事件变化进行计数。因此，当计数器溢出时，将产生一个定时器中断请求，可作为一种唤醒源。



脉冲宽度测量模式时序图 (T1EG=0)

25.4 编程注意事项

当定时 / 事件计数器工作在定时器模式时，若内部系统时钟作为定时器的时钟源，则定时器与单片机所有操作都能同步。在这个模式下，当定时器寄存器计满溢出时，单片机将产生一个内部中断信号，使程序进入相应的内部中断向量。当工作在脉冲宽度测量模式时，定时器时钟源同样使用内部系统时钟，但仅在外部定时器输入引脚上出现正确的逻辑条件时定时器才会运行。由于这个外部事件没有与内部定时器时钟同步，单片机在下一个定时器时钟到来时才能看到这个外部事件。因此在测量值上可能有小的差异，需要程序设计者在程序应用时加以注意。同样的情况发生在定时器设置为事件计数器模式时，该模式下的时钟来源是外部事件，与定时器内部系统时钟不同步。

当读取定时 / 事件计数器值或写数据到预载寄存器时，计数时钟会被禁止以避免发生错误，但这样做可能会导致计数错误，所以程序设计者应加以注意。在第一次使用定时 / 事件计数器之前，要仔细确认是否正确地设定初始值。中断控制寄存器中的定时器中断使能位需正确地设置，否则定时器相关内部中断不被响应。定时器控制寄存器中的有效边沿选择位、定时器工作模式选择位和时钟源控制位需要正确地设置以确保定时器能正确配置为所需的应用。在开启定时 / 事件计数器之前，需要确保先载入定时 / 事件计数器寄存器的初始值。定时 / 事件计数器配置初始化后，可以使用定时器控制寄存器中的使能位来开启或关闭定时器。

当定时 / 事件计数器产生溢出，其中断请求标志位被置位，中断请求产生。若定时 / 事件计数器中断使能，跳转至相关中断向量。不管中断是否使能，在空闲 / 休眠模式下，定时 / 事件计数器的溢出也会产生唤醒信号。若内部时钟源仍然处于有效状态或外部信号继续改变状态，上述情况就有可能发生。在这些情况下，定时 / 事件计数器继续计数，若溢出则将唤醒系统。为了防止这种唤醒，可以在执行“HALT”指令进入空闲 / 休眠模式之前将相应中断请求标志位置位。

26. A/D 转换器

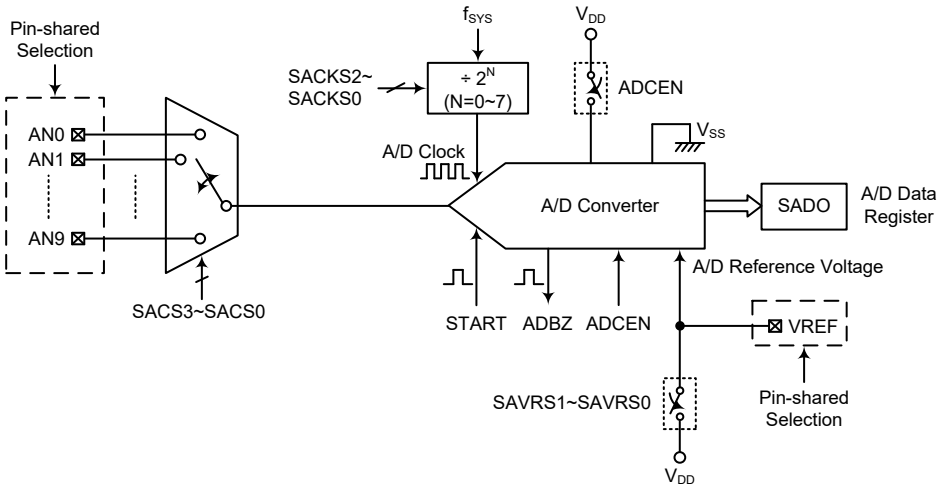
对于大多数电子系统而言，处理现实世界的模拟信号是共同的需求。为了完全由单片机来处理这些信号，首先需要通过 A/D 转换器将模拟信号转换成数字信号。将 A/D 转换器电路集成入单片机，可有效的减少外部器件，随之而来，具有降低成本和减少器件空间需求的优势。

26.1 A/D 简介

该单片机包含一个多通道的 A/D 转换器，它可以直接接入外部模拟信号 (例如传感器或其它控制信号) 并直接将这些信号转换成 8 位的数字量。选择转换外部模拟信号由 SACS3~SACS0 位控制。若要转换外部模拟信号，首先应正确设置好相应的共用引脚控制位，然后再通过 SACS3~SACS0 位选择所需的外部输入通道。关于 A/D 输入信号的详细描述请参考“A/D 转换器控制寄存器”和“A/D 转换器输入信号”两节内容。

外部输入通道	A/D 通道选择位
10: AN0~AN9	SACS3~SACS0

下图显示了 A/D 转换器内部结构和相关的寄存器。



A/D 转换器结构

26.2 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。一个只读寄存器来存放 8 位 A/D 转换数据的值。剩下两个控制寄存器 SADC0 和 SADC1 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
SADO	D7	D6	D5	D4	D3	D2	D1	D0
SADC0	START	ADBZ	ADCEN	—	SACS3	SACS2	SACS1	SACS0
SADC1	—	—	—	SAVRS1	SAVRS0	SACKS2	SACKS1	SACKS0

A/D 转换器寄存器列表

26.2.1 A/D 转换器数据寄存器 – SADO

对于具有 8 位 A/D 转换器的芯片，仅需一个数据寄存器存放转换结果。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。D0~D7 是 A/D 换转数据结果位。当 A/D 转换器除能时，数据寄存器的值将保持不变。

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

A/D 数据寄存器

26.2.2 A/D 转换器控制寄存器 – SADC0, SADC1

寄存器 SADC0 和 SADC1 用来控制 A/D 转换器的功能和操作。这些 8 位的寄存器定义包括选择连接至内部 A/D 转换器的模拟通道，数字化数据格式，A/D 时钟源，并控制和监视 A/D 转换器的忙碌状态。由于每个单片机只包含一个实际的模数转换电路，因此这些外部模拟信号中的每一个都需要分别被发送到转换器。SADC0 寄存器中的 SACS3~SACS0 位用于选择哪个外部模拟输入通道被连接到内部 A/D 转换器。

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

- **SADC0 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	START	ADBZ	ADCEN	—	SACS3	SACS2	SACS1	SACS0
R/W	R/W	R	R/W	—	R/W	R/W	R/W	R/W
POR	0	0	0	—	0	0	0	0

Bit 7 **START**：启动 A/D 转换位
0→1→0：启动
此位用于启动 A/D 转换过程。通常此位为低，但如果设为高再被清零，将启动 A/D 转换过程。

Bit 6 **ADBZ**：A/D 转换忙碌标志位
0：A/D 转换结束或未开始转换
1：A/D 转换中
此位用于表明 A/D 转换过程是否完成。当 START 位由低变为高再变为低时，ADBZ 位为高，表明 A/D 转换已启动。A/D 转换结束后，此位被清零。

Bit 5 **ADCEN**：A/D 转换器使能 / 除能控制位
0：除能
1：使能
此位控制 A/D 内部功能。该位被置高将使能 A/D 转换器。如果该位设为低将关闭 A/D 转换器以降低功耗。当 A/D 转换器除能时，A/D 数据寄存器 SADO 的内容将保持不变。

Bit 4 未定义，读为“0”

Bit 3~0 **SACS3~SACS0**：A/D 外部模拟通道输入选择位
0000：AN0
0001：AN1
0010：AN2
0011：AN3
0100：AN4
0101：AN5
0110：AN6
0111：AN7
1000：AN8
1001：AN9
1010~1111：无通道，输入浮空

- **SADC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	SAVRS1	SAVRS0	SACKS2	SACKS1	SACKS0
R/W	—	—	—	R/W	R/W	R/W	R/W	R/W
POR	—	—	—	0	0	0	0	0

Bit 7~5　未定义，读为“0”

Bit 4~3　**SAVRS1~SAVRS0**：A/D 转换器参考电压选择位
00：来自 VREF 引脚
01：来自内部 V_{DD}
1x：来自 VREF 引脚
这几位用于选择 A/D 转换器的参考电压。必须注意当 SAVRS1~SAVRS0 为“01”选择内部 A/D 转换器电源作为参考电压时，需正确的设置相应的共用引脚功能控制位，不能将 VREF 引脚设置为参考电压输入。否则，VREF 引脚的外部输入电压也会连接到内部 A/D 转换器电源。

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

26.3 A/D 转换器参考电压

A/D 转换器参考电压可以来自正电源电压 V_{DD} 或外部参考源引脚 VREF，可通过 SAVRS1~SAVRS0 位来选择。当 SAVRS1~SAVRS0 位为“01”时，A/D 转换器参考电压来自 V_{DD}。否则，当 SAVRS1~SAVRS0 位为“01”以外任何值时，A/D 转换器参考电压来自 VREF 引脚。由于 VREF 引脚与其它引脚功能共用 I/O 口，当 VREF 引脚被选作参考电压引脚时，相应的引脚共用功能选择位应被预先设置以除能其它引脚共用功能。然而，若选择 A/D 转换器电源作为 A/D 转换器的参考电压，VREF 引脚不能配置为参考电压输入功能，以避免 VREF 引脚与 A/D 转换器电源 V_{DD} 内部相连接。

26.4 A/D 转换器输入信号

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

SACS[3:0]	输入信号	描述
0000~1001	AN0~AN9	外部模拟通道输入 ANn
1010~1111	—	无通道，输入浮空

A/D 转换器输入信号选择

26.5 A/D 转换器操作

SADC0 寄存器中的 START 位，用于打开 A/D 转换器。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 位会自动置为“0”。此外，也会置位中断控制寄存器内相应的 A/D 中断请求标志位，如果中断使能，就会产生对应的内部中断信号。A/D 内部中断信号将引导程序跳转到相应的 A/D 内部中断地址。如果 A/D 内部中断被禁止，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 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，因为它们超出了 A/D 转换时钟周期规定的范围。

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

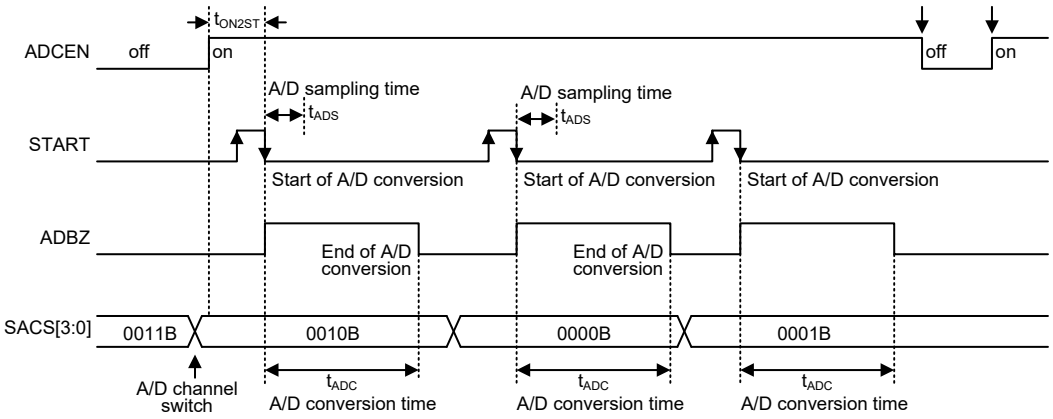
SADC0 寄存器中的 ADCEN 位用于控制 A/D 转换电路电源的开启和关闭。该位必须置高以开启 A/D 转换器电源。当设置 ADCEN 位为高开启 A/D 转换器内部电路时，在 A/D 转换成功开启前需一段延时。即使通过相关引脚共用控制位选择无引脚作为 A/D 输入，如果 ADCEN 设为“1”，那么仍然会产生功耗。因此在功耗敏感的应用中，当未使用 A/D 转换器功能时，建议设置 ADCEN 为低以减少功耗。

26.6 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 8 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 12 个 A/D 时钟周期。

$$最大\ A/D\ 转换率 = 1 \div (A/D\ 时钟周期 \times 12)$$

下列时序图表示外部通道输入信号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 $12t_{ADCK}$，t_{ADCK} 为 A/D 时钟周期。



A/D 转换时序图

26.7 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
  应设置相关的引脚共用控制位将该引脚规划为 A/D 输入引脚。通过设置 SACS3~SACS0 位选择哪个外部通道接至 A/D 转换器。
- 步骤 4
  通过 SADC1 寄存器中的 SAVRS1~SAVRS0 位选择参考电压。若选择参考电压 V_{DD}，应合理设置 PAS0 和 PCS0 寄存器以确保未选择到 VREF 引脚功能。
- 步骤 5
  如果要使用中断，则中断控制寄存器需要正确地设置，以确保 A/D 中断功能是激活的。总中断控制位 EMI 需要置位为“1”，以及 A/D 转换器中断位 ADE 也需要置位为“1”。
- 步骤 6
  现在可以通过设置 SADC0 寄存器中的 START 位从“0”到“1”再回到“0”，开始模数转换的过程。
- 步骤 7
  如果 A/D 转换正在进行中，ADBZ 位会被置为逻辑高。A/D 转换完成后，ADBZ 位会被置为逻辑低，并可从 SADO 寄存器中读取输出数据。

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

26.8 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。

26.9 A/D 转换功能

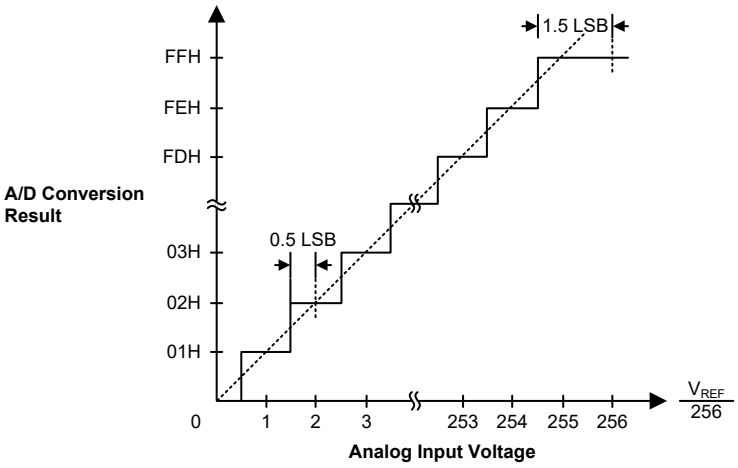
单片机含有一组 8 位的 A/D 转换器，它们转换的最大值可达 FFH。由于模拟输入最大值等于实际 A/D 转换器参考电压值，V_{REF}，因此每一位可表示 $V_{REF}/256$ 的模拟输入值。

$$1\ LSB = V_{REF} \div 256$$

通过下面的等式可估算 A/D 转换器输入电压值：

$$A/D\ 输入电压 = A/D\ 数字输出值 \times (V_{REF} \div 256)$$

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



理想的 A/D 转换功能

26.10 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 1：使用查询 ADBZ 的方式来检测转换结束

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

```
mov SADO_buffer,a          ; save result to user defined register
:
:
jmp start_conversion       ; start next A/D conversion
```

范例 2：使用中断的方式来检测转换结束

```
clr ADE                    ; disable ADC interrupt
mov a,03h                  ; select A/D input signal from external channel,
                           ; reference voltage from VDD,
mov SADC1,a                ; and fSYS/8 as A/D clock
mov a,03h
mov PAS0,a                 ; set PAS0 to configure pin AN0
mov a,20h
mov SADC0,a                ; enable the A/D converter and connect AN0 channel to A/D
                           ; converter
:
start_conversion:
clr START                  ; high pulse on START bit to initiate conversion
set START                  ; reset A/D
clr START                  ; start A/D
clr ADF                    ; clear ADC interrupt request flag
set ADE                    ; enable ADC interrupt
set EMI                    ; enable global interrupt
:
:
; ADC interrupt service routine
ADC_ISR:
mov acc_stack,a            ; save ACC to user defined memory
mov a,STATUS
mov status_stack,a         ; save STATUS to user defined memory
:
:
mov a,SADO                 ; read byte conversion result value
mov SADO_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
```

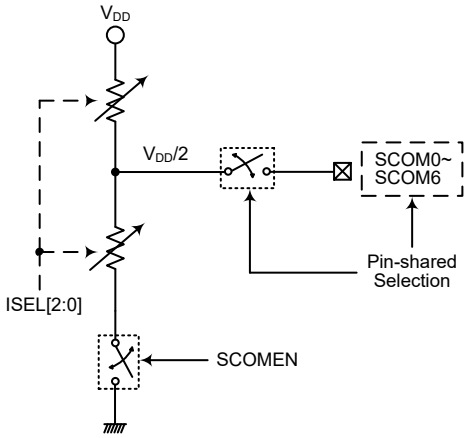
27. 软件控制的 LCD 驱动器

该单片机具有驱动外部 LCD 面板的能力。LCD 驱动的 COM 脚 SCOM0~SCOM6 与 I/O 口共用。LCD 的 COM 控制信号通过应用程序实现。

27.1 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 SCOMn 引脚 R 型偏压电流，使得 LCD COM 驱动产生电压 V_{SS}、$V_{DD}/2$ 和 V_{DD}，从而实现 1/2 偏压操作。

SCOMC 寄存器中的 SCOMEN 位是 LCD 驱动的主控制位，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先设置为输出以使能 LCD 驱动操作。



软件控制的 LCD 驱动结构

27.2 LCD 偏压电流控制

LCD COM 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。通过设置 SCOMC 寄存器中 ISEL2~ISEL0 位可以配置不同的偏压电流。所有的 COM 引脚与 I/O 口共用，通过相应的引脚共用功能选择位将这些引脚选作 SCOM 引脚。

- **SCOMC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	ISEL2	ISEL1	ISEL0	SCOMEN	—	—	—	—
R/W	R/W	R/W	R/W	R/W	—	—	—	—
POR	0	0	0	0	—	—	—	—

Bit 7~5 **ISEL2~ISEL0**：SCOM 典型偏压电流选择 (@V_{DD}=5V)
000：25μA
001：50μA
010：100μA
011：200μA
100：5μA
101~111：12.5μA

Bit 4 **SCOMEN**：软件控制 LCD 驱动使能控制位
0：除能
1：使能

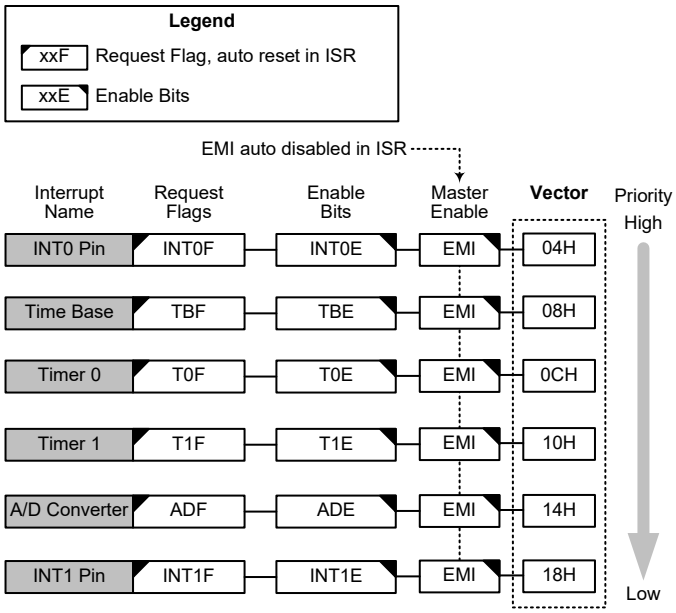
当 SCOMEN 位为“1”时，SCOMn 引脚可通过配置相应的引脚共用选择位使能。若 SCOMEN 为“0”，则 SCOMn 的输出固定在 V_{DD} 电平。应注意相应的引脚共用选择位应在 SCOMn 功能使能之前预先合理设置。

Bit 3~0　　未定义，读为“0”

28. 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溢出并且产生中断时，系统会暂时中止当前的程序而转到执行相对应的中断服务程序。此单片机提供多个外部中断和内部中断功能，外部中断由 INT0~INT1 引脚动作产生，而内部中断由各种内部功能产生，如定时 / 事件计数器和时基等。

各个中断使能位以及相应的请求标志位，以优先级的次序显示在下图。所有中断源都有各自的中断向量。



中断结构

28.1 中断寄存器

中断控制基本上是在单片机发生一些情况时设置请求标志位，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。寄存器总的分为两类。第一类是 INTC0~INTC1 寄存器，用于设置基本的中断；第二类是 INTEG 寄存器，用于设置外部中断边沿触发类型。

中断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=0~1
时基	TBE	TBF	—
定时 / 事件计数器	TnE	TnF	n=0~1
A/D 转换器	ADE	ADF	—

中断寄存器位命名模式

寄存器名称	位							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	T0F	TBF	INT0F	T0E	TBE	INT0E	EMI
INTC1	—	INT1F	ADF	T1F	—	INT1E	ADE	T1E

中断寄存器列表

- **INTEG 寄存器**

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

Bit 7~4　未定义，读为“0”

Bit 3~2　**INT1S1~INT1S0**：INT1 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 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	—	T0F	TBF	INT0F	T0E	TBE	INT0E	EMI
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7　未定义，读为“0”

Bit 6　**T0F**：定时器 / 事件计数器 0 中断请求标志位
0：无请求
1：中断请求

Bit 5　**TBF**：时基中断请求标志位
0：无请求
1：中断请求

Bit 4　**INT0F**：INT0 中断请求标志位
0：无请求
1：中断请求

Bit 3 **T0E**：定时器 / 事件计数器 0 中断控制位
0：除能
1：使能

Bit 2 **TBE**：时基中断控制位
0：除能
1：使能

Bit 1 **INT0E**：INT0 中断控制位
0：除能
1：使能

Bit 0 **EMI**：总中断控制位
0：除能
1：使能

- **INTC1 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	—	INT1F	ADF	T1F	—	INT1E	ADE	T1E
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 **INT1F**：INT1 中断请求标志位
0：无请求
1：中断请求

Bit 5 **ADF**：A/D 转换器中断请求标志位
0：无请求
1：中断请求

Bit 4 **T1F**：定时器 / 事件计数器 1 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 **INT1E**：INT1 中断控制位
0：除能
1：使能

Bit 1 **ADE**：A/D 转换器中断控制位
0：除能
1：使能

Bit 0 **T1E**：定时器 / 事件计数器 1 中断控制位
0：除能
1：使能

28.2 中断操作

若中断事件条件产生，如定时器溢出等等，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“JMP”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。

一旦中断子程序被响应，系统将自动清零 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，执行优先级如中断结构图所示。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。

28.3 外部中断

通过 INTn 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，INTn 引脚的状态发生变化，外部中断请求标志 INTnF 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 INTnE 需先被置位。此外，必须使用 INTEG 寄存器使能外部中断功能并选择触发沿类型。外部中断引脚和普通 I/O 口共用，如果相应寄存器中的中断使能位被置位，并且通过引脚共用寄存器选择外部中断脚，此引脚将被作为外部中断脚使用。此时该引脚必须通过设置控制寄存器，将该引脚设置为输入口。当中断使能，堆栈未满并且外部中断脚状态改变，将调用外部中断向量子程序。当响应外部中断服务子程序时，中断请求标志位 INTnF 会自动复位且 EMI 位会被清零以除能其它中断。注意，即使此引脚被用作外部中断输入，其上拉电阻仍保持有效。

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。

28.4 定时 / 事件计数器中断

当定时 / 事件计数器溢出，相应的中断请求标志位 TnF 被置位时定时 / 事件计数器中断请求产生。若要程序跳转到相应中断向量地址，总中断控制位 EMI 和相关定时器中断使能位 TnE 位需先被置位。当中断使能，堆栈未满且定时 / 事件计数器溢出时，将调用相关的定时器中断向量子程序。当定时器中断响应，相应的中断请求标志位 TnF 会自动复位且 EMI 位会被清零以除能其它中断。

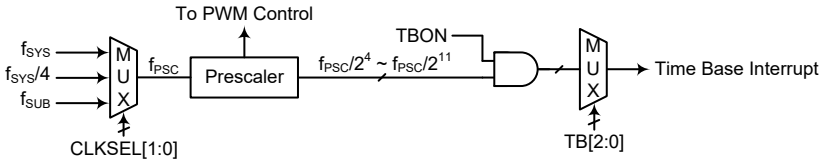
28.5 A/D 转换器中断

A/D 转换动作的结束控制 A/D 转换器中断。当 A/D 转换器中断请求标志 ADF 被置位，即 A/D 转换过程完成时，中断请求发生。若要跳转到相应中断向量地址，总中断控制位 EMI、A/D 转换器中断使能位 ADE 需先被置位。当中断使能，堆栈未满且 A/D 转换动作结束时，将调用 A/D 转换器中断向量子程序。当 A/D 转换器中断响应时，ADF 标志将自动清除，EMI 将被自动清零以除能其它中断。

28.6 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 TBF 被置位时，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 和时基使能位 TBE 需先被置位。当中断使能，堆栈未满且时基溢出时，将调用相关的中断向量子程序。当响应中断服务子程序时，中断请求标志位 TBF 会自动复位且 EMI 位会被清零以除能其它中断。

时基中断的目的是提供一个固定周期的中断信号。其时钟源 f_{PSC} 来自内部时钟源 f_{SYS}，$f_{SYS}/4$ 或 f_{SUB}。f_{PSC} 输入时钟首先经过分频器，分频率由程序设置 TBC 寄存器相关位获取合适的分频值以提供更长的时基中断周期。控制时基中断周期的时钟源通过 TBC 寄存器中的 CLKSEL1~CLKSEL0 位进行选择。编程时需注意，时基功能时钟源与定时 / 事件计数器内部时钟源相同。



时基中断

- **TBC 寄存器**

Bit	7	6	5	4	3	2	1	0
Name	TBON	—	CLKSEL1	CLKSEL0	—	TB2	TB1	TB0
R/W	R/W	—	R/W	R/W	—	R/W	R/W	R/W
POR	0	—	0	0	—	0	0	0

Bit 7　**TBON**：时基使能 / 除能控制位
0：除能
1：使能

Bit 6　未定义，读为“0”

Bit 5~4　**CLKSEL1~CLKSEL0**：分频器 f_{PSC} 时钟源选择
00：f_{SYS}
01：$f_{SYS}/4$
1x：f_{SUB}

Bit 3　未定义，读为“0”

Bit 2~0　**TB2~TB0**：选择时基溢出周期位
000：$2^4/f_{PSC}$
001：$2^5/f_{PSC}$
010：$2^6/f_{PSC}$
011：$2^7/f_{PSC}$
100：$2^8/f_{PSC}$
101：$2^9/f_{PSC}$
110：$2^{10}/f_{PSC}$
111：$2^{11}/f_{PSC}$

28.7 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情况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。

28.8 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零。

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。

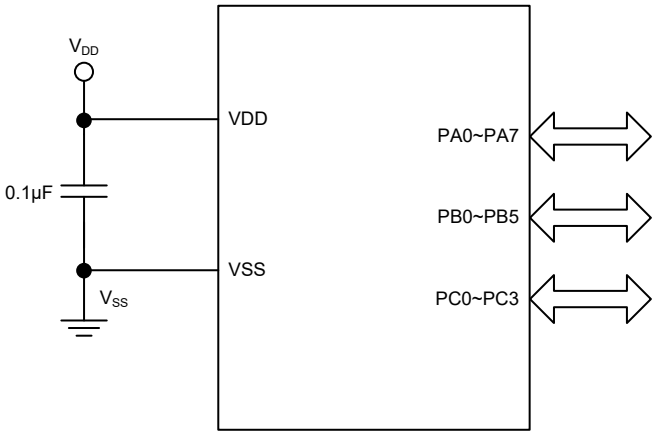
若从中断子程序中返回可执行 RET 或 RETI 指令。除了能返回至主程序外，RETI 指令还能自动设置 EMI 位为高，允许进一步中断。RET 指令只能返回至主程序，清零 EMI 位，除能进一步中断。

29. 配置选项

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

序号	选项
I/O 共用引脚选项	
1	$\overline{\text{RES}}$ 引脚复位功能选择： 0：I/O 口或其它功能 1：$\overline{\text{RES}}$ 引脚
LVR 选项	
2	LVR 功能选择： 0：使能 1：除能
3	LVR 电压选择： 00：1.9V 01：2.1V 10：3.15V 11：4.2V

30. 应用电路



31. 指令集

31.1 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。在芯群单片机中，提供了丰富且灵活的指令，共超过六十条，程序设计者可以事半功倍地实现它们的应用。

为了更加容易理解各种各样的指令码，接下来按功能分组介绍它们。

31.2 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。

31.3 数据的传送

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

31.4 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，在芯群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。

31.5 逻辑和移位运算

标准逻辑运算例如 AND、OR、XOR 和 CPL 全都包含在芯群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。

31.6 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。

31.7 位运算

提供数据存储器中单个位的运算指令是芯群单片机的特性之一。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。

31.8 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，芯群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。

31.9 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。

32. 指令集概要

下表中说明了按功能分类的指令集，用户可以将该表作为基本的指令参考。

32.1 惯例

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

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

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

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

33. 指令定义

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

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

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

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

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

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

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

CPL [m] Complement Data Memory

指令说明 将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或 0 变 1。

功能表示 $[m] \leftarrow [\overline{m}]$

影响标志位 Z

CPLA [m] Complement Data Memory with result in ACC

指令说明 将指定数据存储器中的每一位取逻辑反，相当于从 1 变 0 或 0 变 1，而结果被储存回累加器且数据存储器中的内容不变。

功能表示 $ACC \leftarrow [\overline{m}]$

影响标志位 Z

DAA [m] Decimal-Adjust ACC for addition with result in Data Memory

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

功能表示 $[m] \leftarrow ACC + 00H$ 或
$[m] \leftarrow ACC + 06H$ 或
$[m] \leftarrow ACC + 60H$ 或
$[m] \leftarrow ACC + 66H$

影响标志位 C

DEC [m] Decrement Data Memory

指令说明 将指定数据存储器内容减 1。

功能表示 $[m] \leftarrow [m] - 1$

影响标志位 Z

DECA [m] Decrement Data Memory with result in ACC

指令说明 将指定数据存储器的内容减 1，把结果存放回累加器并保持指定数据存储器的内容不变。

功能表示 $ACC \leftarrow [m] - 1$

影响标志位 Z

HALT	Enter power down mode
指令说明	此指令终止程序执行并关掉系统时钟，RAM 和寄存器的内容保持原状态，WDT 计数器和分频器被清“0”，暂停标志位 PDF 被置位 1，WDT 溢出标志位 TO 被清 0。
功能表示	TO ← 0 PDF ← 1
影响标志位	TO、PDF
INC [m]	Increment Data Memory
指令说明	将指定数据存储器的内容加 1。
功能表示	[m] ← [m] + 1
影响标志位	Z
INCA [m]	Increment Data Memory with result in ACC
指令说明	将指定数据存储器的内容加 1，结果存放回累加器并保持指定的数据存储器内容不变。
功能表示	ACC ← [m] + 1
影响标志位	Z
JMP addr	Jump unconditionally
指令说明	程序计数器的内容无条件地由被指定的地址取代，程序由新的地址继续执行。当新的地址被加载时，必须插入一个空指令周期，所以此指令为 2 个周期的指令。
功能表示	Program Counter ← addr
影响标志位	无
MOV A, [m]	Move Data Memory to ACC
指令说明	将指定数据存储器的内容复制到累加器。
功能表示	ACC← [m]
影响标志位	无
MOV A, x	Move immediate data to ACC
指令说明	将 8 位立即数载入累加器。
功能表示	ACC ← x
影响标志位	无
MOV [m], A	Move ACC to Data Memory
指令说明	将累加器的内容复制到指定的数据存储器。
功能表示	[m] ← ACC
影响标志位	无

NOP	No operation
指令说明	空操作，接下来顺序执行下一条指令。
功能表示	无操作
影响标志位	无

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

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

ORM A, [m]	Logical OR ACC to Data Memory
指令说明	将存在指定数据存储器中的数据和累加器逻辑或， 结果放到数据存储器。
功能表示	[m] ← ACC “OR” [m]
影响标志位	Z

RET	Return from subroutine
指令说明	将堆栈寄存器中的程序计数器值恢复， 程序由取回的地址继续执行。
功能表示	Program Counter←Stack
影响标志位	无

RET A, x	Return from subroutine and load immediate data to ACC
指令说明	将堆栈寄存器中的程序计数器值恢复且累加器载入指定的 立即数，程序由取回的地址继续执行。
功能表示	Program Counter ← Stack ACC←x
影响标志位	无

RETI	Return from interrupt
指令说明	将堆栈寄存器中的程序计数器值恢复且中断功能通过设置 EMI 位重新使能。EMI 是控制中断使能的主控制位。如果在执行 RETI 指令之前还有中断未被响应，则这个中断将在返回主程序之前被响应。
功能表示	Program Counter ←Stack EMI ← 1
影响标志位	无

RL [m]	Rotate Data Memory left
指令说明	将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位。
功能表示	[m].(i+1) ← [m].i (i=0~6) [m].0 ← [m].7
影响标志位	无

RLA [m]	Rotate Data Memory left with result in ACC
指令说明	将指定数据存储器的内容左移 1 位，且第 7 位移到第 0 位，结果送到累加器，而指定数据存储器的内容保持不变。
功能表示	ACC.(i+1) ← [m].i (i=0~6) ACC.0 ←[m].7
影响标志位	无

RLC [m]	Rotate Data Memory Left through Carry
指令说明	将指定数据存储器的内容连同进位标志左移 1 位，第 7 位取代进位标志且原本的进位标志移到第 0 位。
功能表示	[m].(i+1) ← [m].i (i=0~6) [m].0 ← C C ← [m].7
影响标志位	C

RLC A [m]	Rotate Data Memory left through Carry with result in ACC
指令说明	将指定数据存储器的内容连同进位标志左移 1 位，第 7 位取代进位标志且原本的进位标志移到第 0 位，移位结果送回累加器，但是指定数据寄存器的内容保持不变。
功能表示	ACC.(i+1) ← [m].i (i=0~6) ACC.0 ← C C ← [m].7
影响标志位	C

RR [m] Rotate Data Memory right

指令说明 将指定数据存储器的内容循环右移 1 位且第 0 位移到第 7 位。

功能表示 [m].i ← [m].(i+1) (i=0~6)
[m].7 ← [m].0

影响标志位 无

RRA [m] Rotate Data Memory right with result in ACC

指令说明 将指定数据存储器的内容循环右移 1 位，第 0 位移到第 7 位，移位结果存放到累加器，而指定数据存储器的内容保持不变。

功能表示 ACC.i ← [m].(i+1) (i=0~6)
ACC.7 ← [m].0

影响标志位 无

RRC [m] Rotate Data Memory right through Carry

指令说明 将指定数据存储器的内容连同进位标志右移 1 位，第 0 位取代进位标志且原本的进位标志移到第 7 位。

功能表示 [m].i ← [m].(i+1) (i=0~6)
[m].7← C
C ← [m].0

影响标志位 C

RRCA [m] Rotate Data Memory right through Carry with result in ACC

指令说明 将指定数据存储器的内容连同进位标志右移 1 位，第 0 位取代进位标志且原本的进位标志移到第 7 位，移位结果送回累加器，但是指定数据寄存器的内容保持不变。

功能表示 ACC.i ← [m].(i+1) (i=0~6)
ACC.7 ← C
C ← [m].0

影响标志位 C

SBC A, [m] Subtract Data Memory from ACC with Carry

指令说明 将累加器减去指定数据存储器的内容以及进位标志的反，结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。

功能表示 ACC ← ACC – [m] – $\overline{C}$

影响标志位 OV、Z、AC、C

SBCM A, [m]	Subtract Data Memory from ACC with Carry and result in Data Memory
指令说明	将累加器减去指定数据存储器的内容以及进位标志的反，结果存放到数据存储器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示	[m] ← ACC – [m] – $\overline{C}$
影响标志位	OV、Z、AC、C

SDZ [m]	Skip if Decrement Data Memory is 0
指令说明	将指定的数据存储器的内容减 1，判断是否为 0，若为 0 则跳过下一条指令，由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	[m] ← [m] – 1，如果 [m]=0 跳过下一条指令执行
影响标志位	无

SDZA [m]	Skip if decrement Data Memory is zero with result in ACC
指令说明	将指定数据存储器内容减 1，判断是否为 0，如果为 0 则跳过下一条指令，此结果将存放到累加器，但指定数据存储器内容不变。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	ACC ← [m] – 1，如果 ACC=0 跳过下一条指令执行
影响标志位	无

SET [m]	Set Data Memory
指令说明	将指定数据存储器的每一位设置为 1。
功能表示	[m] ← FFH
影响标志位	无

SET [m].i	Set bit of Data Memory
指令说明	将指定数据存储器的第 i 位置位为 1。
功能表示	[m].i ← 1
影响标志位	无

SIZ [m]	Skip if increment Data Memory is 0
指令说明	将指定的数据存储器的内容加 1，判断是否为 0，若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	[m] ←[m] + 1，如果 [m]=0 跳过下一条指令执行
影响标志位	无

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

SNZ [m].i	Skip if bit i of Data Memory is not 0
指令说明	判断指定数据存储器的第 i 位，若不为 0，则程序跳过下一条指令执行。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果为 0，则程序继续执行下一条指令。
功能表示	如果 [m].i≠0，跳过下一条指令执行
影响标志位	无

SUB A, [m]	Subtract Data Memory from ACC
指令说明	将累加器的内容减去指定的数据存储器的数据，把结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示	ACC ← ACC – [m]
影响标志位	OV、Z、AC、C

SUBM A, [m]	Subtract Data Memory from ACC with result in Data Memory
指令说明	将累加器的内容减去指定数据存储器的数据，结果存放到指定的数据存储器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示	[m] ← ACC – [m]
影响标志位	OV、Z、AC、C

SUB A, x	Subtract immediate Data from ACC
指令说明	将累加器的内容减去立即数，结果存放到累加器。如果结果为负，C 标志位清除为 0，反之结果为正或 0，C 标志位设置为 1。
功能表示	ACC ← ACC – x
影响标志位	OV、Z、AC、C

SWAP [m]	Swap nibbles of Data Memory
指令说明	将指定数据存储器的低 4 位和高 4 位互相交换。
功能表示	[m].3~[m].0 ↔ [m].7~[m].4
影响标志位	无

SWAPA [m]	Swap nibbles of Data Memory with result in ACC
指令说明	将指定数据存储器的低 4 位与高 4 位互相交换，再将结果存放到累加器且指定数据寄存器的数据保持不变。
功能表示	ACC.3~ACC.0 ← [m].7~[m].4 ACC.7~ACC.4 ← [m].3~[m].0
影响标志位	无

SZ [m]	Skip if Data Memory is 0
指令说明	指定数据存储器的内容会先被读出，后又被重新写入指定数据存储器内。判断指定数据存储器的内容是否为 0，若为 0，则程序跳过下一条指令执行。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	如果 [m]=0，跳过下一条指令执行
影响标志位	无

SZA [m]	Skip if Data Memory is 0 with data movement to ACC
指令说明	将指定数据存储器内容复制到累加器，并判断指定数据存储器的内容是否为 0，若为 0 则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	ACC ←[m]，如果 [m]=0，跳过下一条指令执行
影响标志位	无

SZ [m].i	Skip if bit i of Data Memory is 0
指令说明	判断指定数据存储器的第 i 位是否为 0，若为 0，则跳过下一条指令。由于取得下一个指令时会要求插入一个空指令周期，所以此指令为 2 个周期的指令。如果结果不为 0，则程序继续执行下一条指令。
功能表示	如果 [m].i=0，跳过下一条指令执行
影响标志位	无
TABRD [m]	Read table (specific page or current page) to TBLH and Data Memory
指令说明	将表格指针 (TBHP 和 TBLP，若无 TBHP 则仅 TBLP) 所指的程序代码低字节移至指定数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
TABRDL [m]	Read table (last page) to TBLH and Data Memory
指令说明	将表格指针 TBLP 所指的程序代码低字节 (最后一页) 移至指定数据存储器且将高字节移至 TBLH。
功能表示	[m] ← 程序代码 (低字节) TBLH ← 程序代码 (高字节)
影响标志位	无
XOR A, [m]	Logical XOR Data Memory to ACC
指令说明	将累加器的数据和指定的数据存储器内容逻辑异或，结果存放到累加器。
功能表示	ACC ← ACC“XOR”[m]
影响标志位	Z
XORM A, [m]	Logical XOR ACC to Data Memory
指令说明	将累加器的数据和指定的数据存储器内容逻辑异或，结果放到数据存储器。
功能表示	[m] ← ACC“XOR”[m]
影响标志位	Z
XOR A, x	Logical XOR immediate data to ACC
指令说明	将累加器的数据与立即数逻辑异或，结果存放到累加器。
功能表示	ACC ← ACC“XOR”x
影响标志位	Z

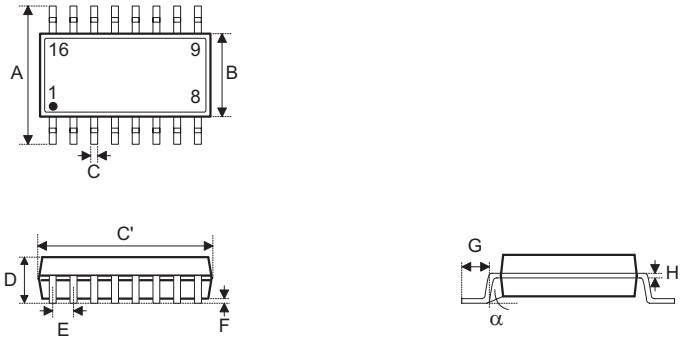
34. 封装信息

请注意，这里提供的封装信息仅作为参考。由于这个信息经常更新，提醒用户咨询芯群网站 以获取最新版本的封装信息。

封装信息的相关内容如下所示，点击可链接至芯群网站相关信息页面。

- 封装信息 (包括外形尺寸、包装带和卷轴规格)
- 封装材料信息
- 纸箱信息

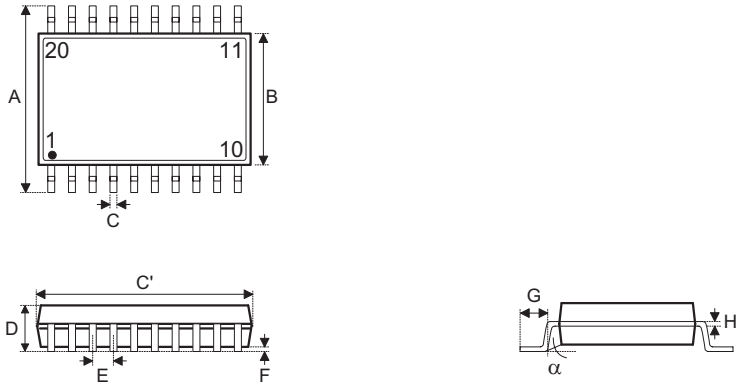
34.1 16-pin NSOP (150mil) 外形尺寸



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

符号	尺寸 (单位：mm)		
	最小值	典型值	最大值
A	6.00 BSC		
B	3.90 BSC		
C	0.31	—	0.51
C’	9.90 BSC		
D	—	—	1.75
E	1.27 BSC		
F	0.10	—	0.25
G	0.40	—	1.27
H	0.10	—	0.25
α	0°	—	8°

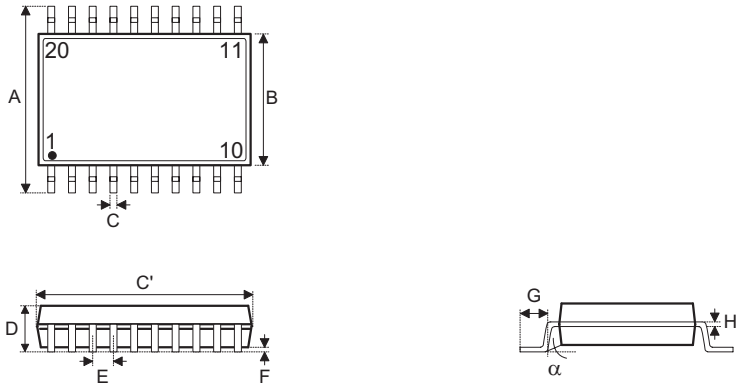
34.2 20-pin NSOP (150mil) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	典型值	最大值
A	0.228	0.236	0.244
B	0.146	0.154	0.161
C	0.009	—	0.012
C’	0.382	0.390	0.398
D	—	—	0.069
E	0.032 BSC		
F	0.002	—	0.009
G	0.020	—	0.031
H	0.008	—	0.010
α	0°	—	8°

符号	尺寸 (单位：mm)		
	最小值	典型值	最大值
A	5.80	6.00	6.20
B	3.70	3.90	4.10
C	0.23	—	0.30
C’	9.70	9.90	10.10
D	—	—	1.75
E	0.80 BSC		
F	0.05	—	0.23
G	0.50	—	0.80
H	0.21	—	0.25
α	0°	—	8°

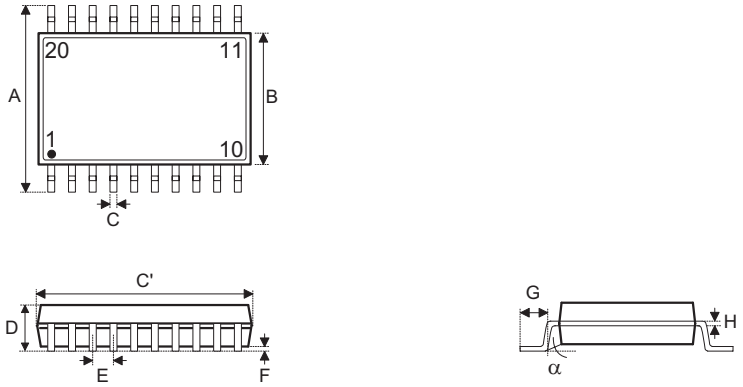
34.3 20-pin SOP (300mil) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	典型值	最大值
A		0.406 BSC	
B		0.295 BSC	
C	0.012	—	0.020
C’		0.504 BSC	
D	—	—	0.104
E		0.050 BSC	
F	0.004	—	0.012
G	0.016	—	0.050
H	0.008	—	0.013
α	0°	—	8°

符号	尺寸 (单位：mm)		
	最小值	典型值	最大值
A		10.30 BSC	
B		7.50 BSC	
C	0.31	—	0.51
C’		12.80 BSC	
D	—	—	2.65
E		1.27 BSC	
F	0.10	—	0.30
G	0.40	—	1.27
H	0.20	—	0.33
α	0°	—	8°

34.4 20-pin SSOP (150mil) 外形尺寸



符号	尺寸 (单位：inch)		
	最小值	典型值	最大值
A		0.236 BSC	
B		0.154 BSC	
C	0.008	—	0.012
C’		0.341 BSC	
D	—	—	0.069
E		0.025 BSC	
F	0.004	—	0.010
G	0.016	—	0.050
H	0.004	—	0.010
α	0°	—	8°

符号	尺寸 (单位：mm)		
	最小值	典型值	最大值
A		6.00 BSC	
B		3.90 BSC	
C	0.20	—	0.30
C’		8.66 BSC	
D	—	—	1.75
E		0.635 BSC	
F	0.10	—	0.25
G	0.41	—	1.27
H	0.10	—	0.25
α	0°	—	8°

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