东湾半导体
DONGCHENG SEMICONDUCTOR

DC8S3522

规格书

版本0.94

修改记录

版本	日期	描述
0.80	Jan, 2023	1. 更新 ADC 转换图 2. 更新 ADC 在 Fadc=1MHz 时的转换时间为 42us 3. 修改实例 asm 代码中 DECXSZ 和 INCXSZ 的说明 4.在"INSTRUCTION SET"示例 asm 代码的标签后插入冒号 5.在 ADC 电特性中添加转换时间注释 6. ICP 模式需要 PA7 引脚描述在 p.11
0.81	Jan, 2023	1. 更新 Fsys=20MHz(FXT)的最小工作电压 2. 修改引脚分配图 3.修改 DC8S3522 框图 4. 修改 LDOC 规范 5. 更新最小工作电压图 6. 修改 PWMCKS=FIRC*2 的描述信息
0.82	Feb, 2023	1. 在特征图中添加"LDOC 电压 vs VCC" 2. 删除第一页 3.更新 LDOC 规范
0.83	Feb, 2023	1. 从版本 0.1 到版本 0.79 删除修订历史中的条目 2. PA7 在 Feature、I/O Port、hink 和 DC 特性描述中没有高 sink 能力 3.在引脚汇总中增加 QFN-16 和 DFN-10 4. 更新 Ioh, Iol, Rpu 的典型值
0.84	Feb, 2023	1. 更新最小工作电压图 2. 更新 DC 特性中的 Icc
0.85	Feb, 2023	1. 添加引脚变化唤醒功能 2. 增加一段 Pin Change Wake Up 3.新增"edged VBG1.2V/2V/2.5V"功能 4. 增加 BG2P5TRIM 和 BG2TRIM 使用说明
0.90	Feb, 2023	1. 更新 DC 特性中的 Icc 2. 更新直流特性中的 LVD 迟滞窗口 3.更新直流特性中的 LVR 滞环窗口 4. 添加 LDOC 电流 vs.电压 5. 在 FEATURES 中删除 TBD 和相关修改 6. 添加 DC8S3521H
0.91	Feb, 2023	1.增加 DC8S3552/22 和 DC8S3522 的对比表 2. 将 CIN3 和 CPI4 更改为保留 3.添加"不要使用 ADVREFS=11 来选择 DAC 的 VREF" 4. 删除 BG2P5TRIM 5. 添加 PORSEL, ADVREFS=11, LDOCOUT, IRCFT, BG2TRIM 和 RDCTL 不能模拟 7. 18Eh.3 在仿真中，必须将 3 设置为 1 8. PA7 没有高灌电流能力，1/2 偏压和下拉 9. WDT 将 96~1536ms 改为 84~1344m 10. 将 WKT 的 12~96ms 改为 10.5~84ms
0.92	Feb, 2023	1. 在 PROM 描述中增加 ROM 寿命(第 14 页) 2. 修改 Ioh@3V 典型值为 5.3mA 3.将 p.42 示例代码中的 WDT Time out=192ms 修改为 168ms 4. 将 p.42 样例代码中的 WKT period=48ms 修改为 42ms 5. 修改"RESET Input Low width"典型值从 30us 到 11us 6.修改"CPU 启动时间"典型值为 21ms 7. 修正最大 FIRC 频率@-40℃~105℃Vcc=3~5V 为+2% 8.修改 LDOC 的描述:在第 8 页的功能部分中修改 1.2V LDO 调节器@Max 70mA 9. 修改 LDOC 的描述:在第 12 页引脚描述部分中，1.2V LDO 调节器@Max 70mA 输出 10. 修改 LDOC 的描述:第 91 页直流特性部分的 1.2V LDO 稳压器 11. 放大"LDOC 电流 vs 电压"图 12. 在第 8 页的功能部分中删除 2V ADC Vref 的加热线圈
0.93	Mar, 2023	1. 在最低工作电压曲线图中加入 RDCTL=8ns 的条件 2. 将存储器映射中的 SCIN=010 及 SCIP=011 改为保留(第 75 页)
0.94	Mar, 2023	1. 减少目录中的层次(hierarchy) 2. .将"RDCTL=3 或 11"变更为"RDCTL=8ns"(第 5, 8, 74 页) 3. 将最低工作电压曲线图中的"LVRE=0x02=2.3V 变更为"LVRE=2.3V"

目录

家族预览

	DC8S3552 (TK) DC8S3522 (IO)	DC8S3522
EV board	On chip debug	DC8S3552 (TK) DC8S3522 (IO)
RAM	336	256
EEPROM	128	X
CTK	V	X
SIRC	84 KHz@5V/25℃	95.6 KHz@5V/25℃
WDT	96ms, 192ms, 768ms,1536ms @5V	84ms, 168ms, 672ms, 1344 ms @5V
WKT	12ms,24ms,48ms,96ms @5V	10.5ms,21ms,42ms,84ms @5V
SFR.RDCTL	X	V (建议 RDCTL=8ns)
OPA	V	X
SFR.OPOF (CMPP to OPO)	OPOF=0 (POR, CMPP <= OPO) OPOF=1 (CMPP <= CIPx)	无 OPA, 在仿真时必须设置 OPOF =1 (CMPP connect to CIPx). CIN3 和 CIP4 保留
SFR.ADVREFS	VCC / 2.48V	VCC / 2 / 2.48V ADVREFS=2V, 不能仿真
SFR.BG2TRIM	X	读取 BG2TRIM 和写入 BGTRIM, 获得 ADVREFS=2.0V
SFR.SVRF (DAC VREF)	VCC / 1.2 / 2.48V	VCC / 1.2 / 2.48V
SFR.IRCFT	X	微调32级频率每个IRCF步骤，IRCFT 不能仿真
PAD.LDOC	X	LDOC, 不能仿真
PA7 High Sink	75mA@5V	48mA@5V No 1/2 bias 无下拉
IO	PA7~0 PB7~0 PD1~0	PA7~0 PB6~4, PB2~0
POR	1.95V 无 PORSEL	1.9V 有 PORSEL
Minimal Operating Voltage	1.9V @16MHz	2.3V @16MHz

特性

1. **ROM: 4K x 16 bits MTP**
2. **RAM: 256 x 8 位**
3. **堆栈: 8 级**
4. **系统时钟类型选择:**
   - 外部快时钟： 1~20 MHz 晶振 (FXT)
   - 内部 RC 快时钟: (FIRC, 16 MHz)
   - 外部慢时钟: 32768 Hz 晶振 (SXT)
   - 内部 RC 慢时钟: (SIRC, 95 KHz@V_{CC}=5V)
5. **系统时钟预分频器:**
   - 系统时钟可以 1/2/4/8 分频选项
6. **省电工作模式**
   - 快速模式: 慢时钟使能，CPU 在快时钟下保持运行
   - 慢速模式: 快时钟可以关闭或使能，CPU 在慢时钟下保持运行
   - 空闲模式: 快时钟和 CPU 停止。慢时钟，T2 和 WKT 保持运行
   - 停止模式: 所有时钟停止，T2 和 WKT 停止运行
7. **3 个独立定时器**
   - Timer0
     - 8 位定时器，除以 1~256 预分频选项，具有自动重载 / 计数器 / 中断 / 停止功能
   - Timer1
     - 8 位定时器，除以 1~256 预分频选项，具有自动重载 / 中断 / 停止功能
     - 溢出和翻转输出
   - T2
     - 15 位定时器，带有 4 个中断间隔时间选项
     - 空闲模式唤醒定时器或用作一个简单的 15 位时基
     - 时钟源: 慢时钟，Fsys/128 或 FIRC/512 (16 MHz/512)
8. **中断**
   - 三个外部中断引脚
     - 1 个引脚为下降沿唤醒触发并中断
     - 2 个引脚为上升或下降沿唤醒触发并中断

- Timer0 / Timer1 / T2 / WKT 中断
- ADC 中断
- 比较器中断
- PWM 中断
- LVD 中断

9. 唤醒定时器 (WKT)

- 由内置 RC 振荡器提供时钟，具有 4 个可调节的中断时间

  - 10.5ms / 21ms / 42ms / 84ms @Vcc=5V

10. 看门狗定时器 (WDT)

- 由内置 RC 振荡器提供时钟，具有 4 个可调的复位时间

- 84 ms / 168 ms / 672 ms / 1344ms @Vcc=5V

- 在停止模式下可以关闭 / 使能看门狗定时器

11. 6 个 16 位 PWMs

- 6 个单独的工作占空比可调，共享的周期可调
- PWM 时钟源: 系统时钟 (Fsys), FIRC (16 MHz), FIRC*2 (32 MHz)
- PWM0 支持互补输出 (PWM0P, PWM0N)
- PWM0 输出, 非重叠时间可调: (0~15)*(PWMCLK)
- PWM0N/0P/1/2/3/5 有两个输出(PWM4 只有一个输出)

12. 12 位 ADC， 13 个外部输入通道和 2 个内部电压通道

- 2 个内部电压通道: VBG, 1/4VCC
- ADC 参考电压: V_{CC}, V_{BG} (2.48V) and V_{BG} (2V)

13. 比较器

- 比较器 x 1

  - 带 7 位 DAC 输入

  - DAC 参考电压可选择 V_{CC} 或 V_{BG} (1.20V, 2.48V)

14. 复位源

- 上电复位
- 看门狗复位
- 低电压复位
- 外部引脚复位

15. 低电压复位 (LVR) / 低电压检测 (LVD)

- 16 级低电压复位: 2.05V ~ 4.15V, 可以选择关闭

- 15 级低电压检测: 2.20V ~ 4.15V, 可以选择关闭

16. 工作电压

- Fsys= 16 MHz, 2V~5.5V @LVR disable/25°C. 建议 LVR ≥ 2.30V 在 -40°C 至 +105°C

注意：上电电压必须超过 POR(1.9V)以及用户设定的 LVR 电压，参见“电气特性表”，从而避免进入 ROM 死区，导致芯片非正常工作

17. 工作温度范围: -40°C to + 105°C

18. 读表取指令：16 位 ROM 数据查表

19. 集成的 16 位循环冗余校验 (CRC) 功能

20. 指令集: 39 条指令

21. I/O 端口:

- 最多 18 个可编程 I/O 引脚
  - 开漏输出
  - CMOS 推挽输出
  - 施密特触发器输入，带上拉/下拉电阻选项
  - 所有 I/O 具备高灌电流
  - 1/2 V_{CC} (1/2 偏压) 输出
- 所有引脚变化唤醒(negedge 和 posedge 触发器)

22. LDOC

- 1.2V LDO 稳压器@70mA 输出到 PA3

23. IRCFT

- FIRC 频率 5 位微调，每个调整步长用于频率跟踪

24. 2V ADC Vref

25. 支持 5 线(ICP)或 8 线烧录

26. RDCTL: 程序 ROM 的读信号延迟控制

- 用户必须将该寄存器切换到“8ns”，以提高最低工作电压的性能

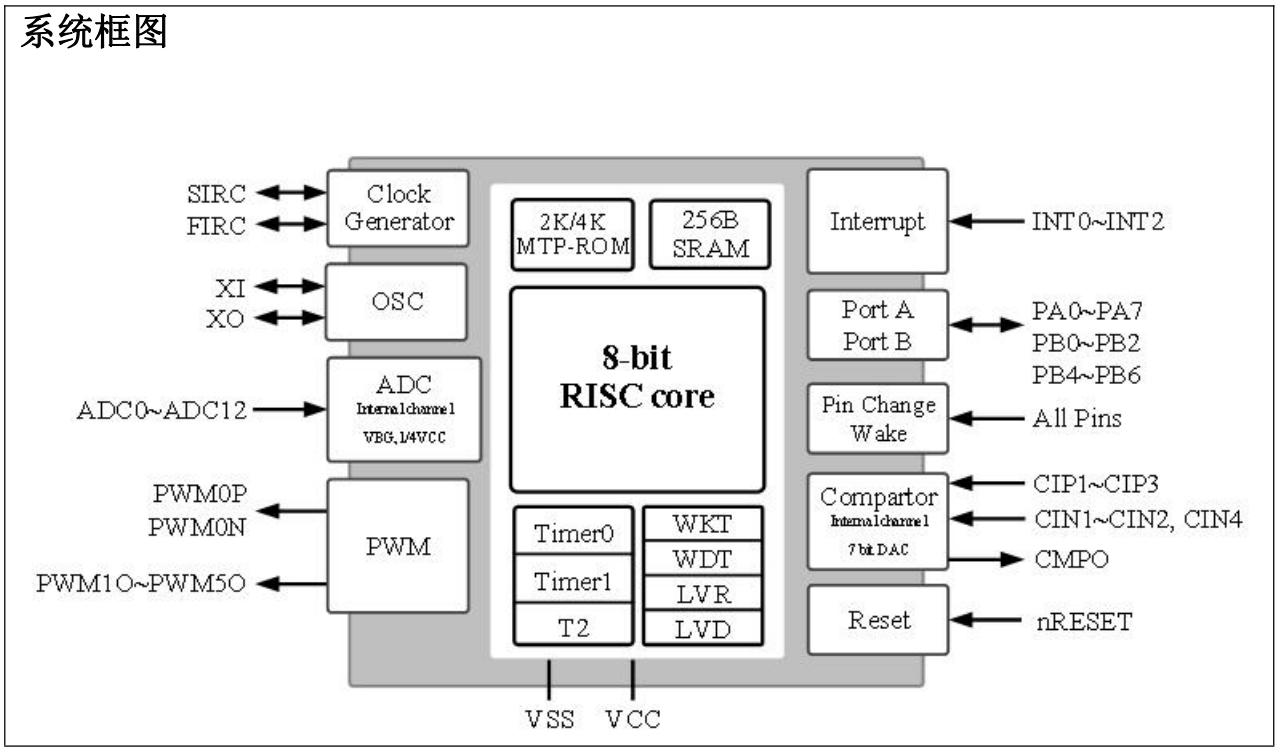
27. Trimmed VBG1.2V/2V

- 用户可以将 BG2TRIM 移动到 BGTRIM，获得精确的 2V VBG.

28. 封装类型:

- 16-pin SOP (150 mil)
- 10-pin MSOP (118 mil)
- 8-pin SOP (150 mil)
- 16-pin QFN (3*3*0.75 - 0.5mm)
- 10-pin DFN (3*3*0.75 - 0.5mm)

29. 支持 EV 板

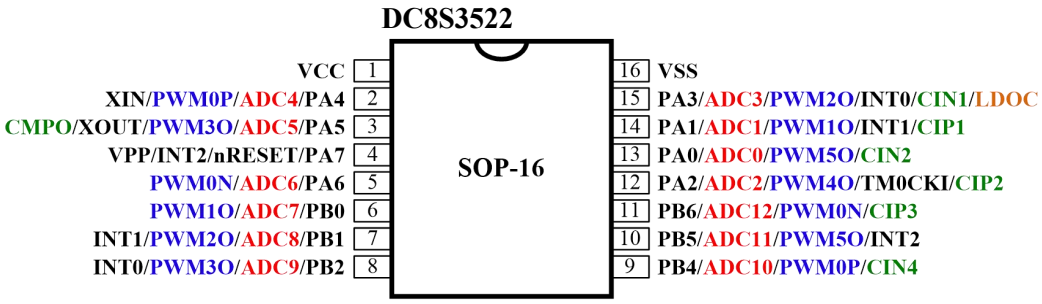


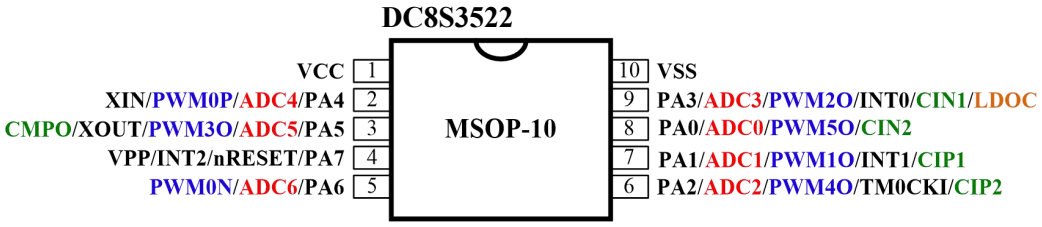
DC8S3552/3522

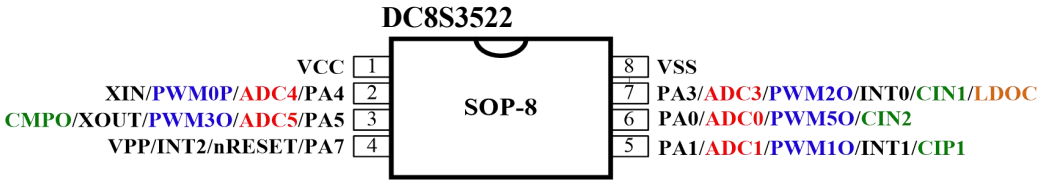
SYSTEM BLOCK DIAGRAM

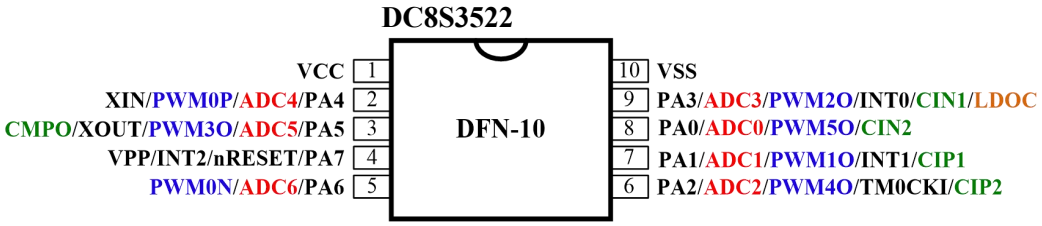
DC8S3522 Block Diagram

引脚分配图

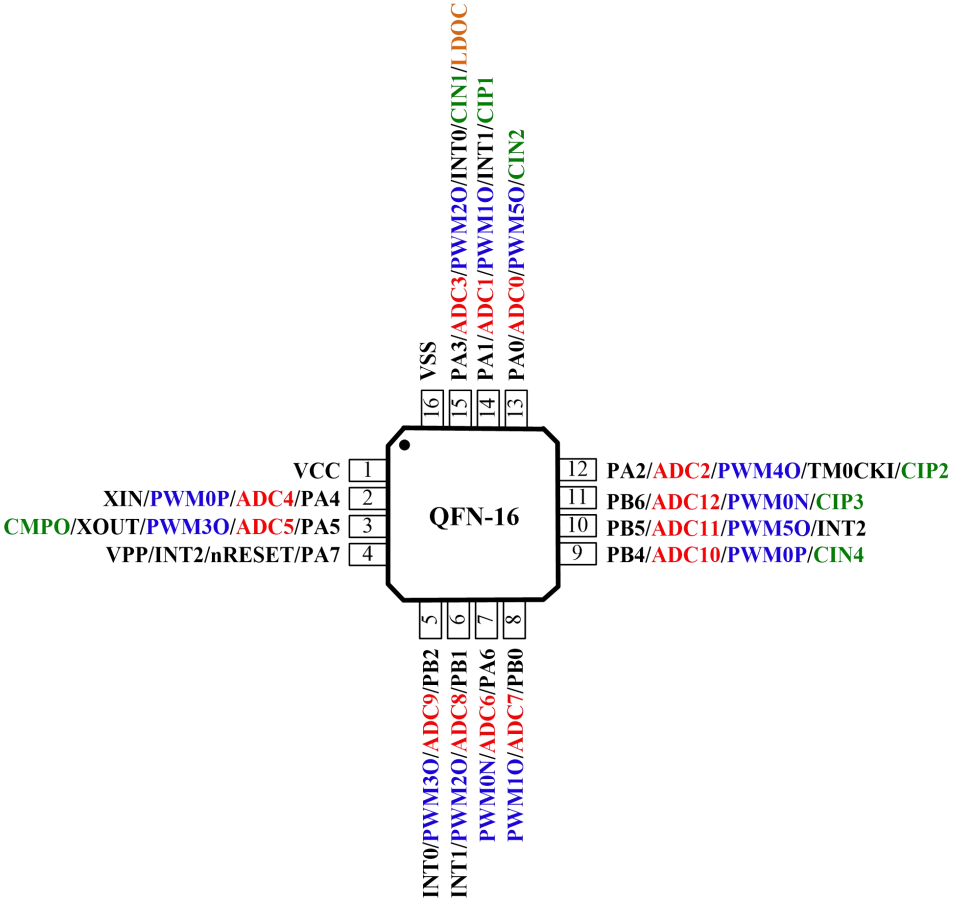

DC8S3522
SOP-16
VCC 1
XIN/PWM0P/ADC4/PA4 2
CMPO/XOUT/PWM3O/ADC5/PA5 3
VPP/INT2/nRESET/PA7 4
PWM0N/ADC6/PA6 5
PWM1O/ADC7/PB0 6
INT1/PWM2O/ADC8/PB1 7
INT0/PWM3O/ADC9/PB2 8
16 VSS
15 PA3/ADC3/PWM2O/INT0/CIN1/LDOC
14 PA1/ADC1/PWM1O/INT1/CIP1
13 PA0/ADC0/PWM5O/CIN2
12 PA2/ADC2/PWM4O/TM0CKI/CIP2
11 PB6/ADC12/PWM0N/CIP3
10 PB5/ADC11/PWM5O/INT2
9 PB4/ADC10/PWM0P/CIN4


DC8S3522
MSOP-10
VCC 1
XIN/PWM0P/ADC4/PA4 2
CMPO/XOUT/PWM3O/ADC5/PA5 3
VPP/INT2/nRESET/PA7 4
PWM0N/ADC6/PA6 5
10 VSS
9 PA3/ADC3/PWM2O/INT0/CIN1/LDOC
8 PA0/ADC0/PWM5O/CIN2
7 PA1/ADC1/PWM1O/INT1/CIP1
6 PA2/ADC2/PWM4O/TM0CKI/CIP2


DC8S3522
SOP-8
VCC 1
XIN/PWM0P/ADC4/PA4 2
CMPO/XOUT/PWM3O/ADC5/PA5 3
VPP/INT2/nRESET/PA7 4
8 VSS
7 PA3/ADC3/PWM2O/INT0/CIN1/LDOC
6 PA0/ADC0/PWM5O/CIN2
5 PA1/ADC1/PWM1O/INT1/CIP1


DC8S3522
DFN-10
VCC 1
XIN/PWM0P/ADC4/PA4 2
CMPO/XOUT/PWM3O/ADC5/PA5 3
VPP/INT2/nRESET/PA7 4
PWM0N/ADC6/PA6 5
10 VSS
9 PA3/ADC3/PWM2O/INT0/CIN1/LDOC
8 PA0/ADC0/PWM5O/CIN2
7 PA1/ADC1/PWM1O/INT1/CIP1
6 PA2/ADC2/PWM4O/TM0CKI/CIP2

DC8S3522


VSS
PA3/ADC3/PWM2O/INT0/CIN1/LDOC
PA1/ADC1/PWM1O/INT1/CIP1
PA0/ADC0/PWM5O/CIN2
16
15
14
13
VCC 1
XIN/PWM0P/ADC4/PA4 2
CMPO/XOUT/PWM3O/ADC5/PA5 3
VPP/INT2/nRESET/PA7 4
QFN-16
12 PA2/ADC2/PWM4O/TM0CKI/CIP2
11 PB6/ADC12/PWM0N/CIP3
10 PB5/ADC11/PWM5O/INT2
9 PB4/ADC10/PWM0P/CIN4
5
6
7
8
INT0/PWM3O/ADC9/PB2
INT1/PWM2O/ADC8/PB1
PWM0N/ADC6/PA6
PWM1O/ADC7/PB0

引脚描述

名称	输入/输出	引脚描述
PA0~PA7 PB0~PB2 PB4~PB6	I/O	可位编程 I/O 端口，用于施密特触发器输入，CMOS 推挽输出，或漏极开路输出或 1/2V_{CC}输出。上拉/下拉电阻可由软件分配
nRESET	I	外部低电平有效复位
VCC, VSS	P	电源输入引脚和地
XIN, XOUT	–	用于系统时钟（FXT 或 SXT）的晶体/谐振器振荡器连接
INT0~INT2	I	外部中断输入
TM0CKI	I	Timer0 在计数器模式下的输入
PWM0P	O	16 位 PWM0 正输出
PWM0N	O	16 位 PWM0 负输出
PWM1O~PWM5O	O	16 位 PWM1~PWM5 输出
CMPO	O	比较器状态输出
ADC0~ADC12	I	ADC 输入通道
CIN1, CIN2, CIN4	I	比较器负端口输入
CIP1~CIP3	I	比较器正端口输入
LDOC	O	1.2V LDO 稳压器 @Max 70mA output

编程引脚：

正常模式 (8-wire): VCC / VSS / PA0 / PA1 / PA3 / PA4 / PA5 / PA7。

ICP 模式(5-wire): VCC / VSS / PA0 / PA1 / PA7 。使用 ICP（在线编程）模式时，PCB 需要除去 PA0，PA1 的所有组件。

引脚摘要

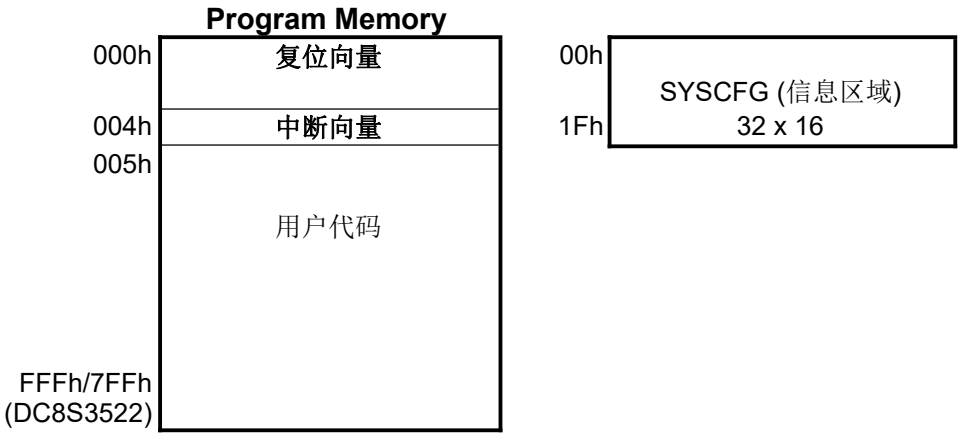
引脚编号				引脚名称	类型	GPIO							替代功能			
						输入				输出						
DC8S3522 (SOP-16)	DC8S3522 (QFN-16)	DC8S3522 (MSOP-10) (DFN-10)	DC8S3522 (SOP-8)			上拉控制	下拉控制	中断	唤醒	开漏	CMOS 推挽	1/2 V_{CC} (1/2 偏压)	PWM	ADC	比较	杂项
2	2	2	2	PA4/ADC4/PWM0P/XIN	I/O	●	●		●	●	●	●	●	●		XIN
3	3	3	3	PA5/ADC5/PWM3O/XOUT/CMPO	I/O	●	●		●	●	●	●	●	●	●	XOUT
4	4	4	4	PA7/nRESET/INT2/VPP	I/O	●		●	●	●	●					nRESET/VPP
5	7	5	–	PA6/ADC6/PWM0N	I/O	●	●		●	●	●	●	●	●		
6	8	–	–	PB0/ADC7/PWM1O	I/O	●	●		●	●	●	●	●	●		
7	6	–	–	PB1/ADC8/PWM2O/INT1	I/O	●	●	●	●	●	●	●	●	●		
8	5	–	–	PB2/ADC9/PWM3O/INT0	I/O	●	●	●	●	●	●	●	●	●		
9	9	–	–	PB4/ADC10/PWM0P/CIN4	I/O	●	●		●	●	●	●	●	●	●	
10	10	–	–	PB5/ADC11/PWM5O/INT2	I/O	●	●	●	●	●	●	●	●	●		
11	11	–	–	PB6/ADC12/PWM0N/CIP3	I/O	●	●		●	●	●	●	●	●	●	
12	12	6	–	PA2/ADC2/PWM4O/TM0CKI/CIP2	I/O	●	●		●	●	●	●	●	●	●	TM0CKI
13	13	8	6	PA0/ADC0/PWM5O/CIN2	I/O	●	●		●	●	●	●	●	●	●	
14	14	7	5	PA1/ADC1/PWM1O/INT1/CIP1	I/O	●	●	●	●	●	●	●	●	●	●	
15	15	9	7	PA3/ADC3/PWM2O/INT0/CIN1/LDOC	I/O	●	●	●	●	●	●	●	●	●	●	LDOC
16	16	10	8	VSS	P											
1	1	1	1	VCC	P											

功能描述

1 CPU 核心

1.1 程序 ROM (PROM)

该器件的 MTP ROM 为 4K Word，带有一个额外的 352 字信息区域来存储 SYSCFG 。ROM 可以多次写入，只要 SYSCFG 的 PROTECT (CFGWH.15) 位不设置就可以读取。
无论 PROTECT 置位还是清零都可以读取 SYSCFG，但只有在擦除用户 ROM 代码区域时才能清除 PROTECT 位。另一方面，如果设置了 PROTECT 位，则写入器不会读取用户 ROM 代码区域，并且在 PROTECT 位清零之前无法更新用户 ROM 代码。ROM 的擦写寿命为 1000 次@Vcc=5V/25℃。



复位向量 (000h)
复位后，系统将在地址 000h 处重新启动程序计数器 (PC)，所有寄存器都将恢复到默认值。

中断向量 (004h)
当中断发生时，程序计数器 (PC) 将被推到堆栈上并跳转到地址 004h。

1.2 System Configuration Register (SYSCFG)

系统配置寄存器 (SYSCFG) 位于 MTP 信息区域；它包含一个 16 位寄存器 (CFGWH)。 SYSCFG 决定了 CPU 初始状态的选项。它只能由 PROM Write 编写。

用户可以通过 SYSCFG 寄存器选择 LVR 工作模式和芯片工作模式。CFGWH 的第 15 位是代码保护选择位。如果该位为 1，则当用户读取 PROM 时，PROM 中的数据将受到保护。

位		**15~0**	
默认值		**0000_0000_0000_0000**	
位		描述	
CFGWH	15	**PROTECT**: 代码保护选择	
		0	关闭
		1	使能
	13-12	**WDTE**: WDT 复位使能	
		0X	关闭
		10	在快速/慢速模式下使能，在空闲/停止模式下关闭
		11	始终使能
	11-8	**LVR**: 低电压复位模式	
		0000	LV Reset 2.05V
		0001	LV Reset 2.20V
		0010	LV Reset 2.30V
		0011	LV Reset 2.45V
		0100	LV Reset 2.60V
		0101	LV Reset 2.75V
		0110	LV Reset 2.90V
		0111	LV Reset 3.00V
		1000	LV Reset 3.15V
		1001	LV Reset 3.30V
		1010	LV Reset 3.45V
		1011	LV Reset 3.60V
		1100	LV Reset 3.70V
		1101	LV Reset 3.85V
		1110	LV Reset 4.00V
		1111	LV Reset 4.15V
	7	**XRSTE**: 外部引脚 (PA7) 复位使能	
		0	关闭 (PA7 作为 I/O 引脚)
		1	使能
	5	**FIRCPSC**: FIRC 预分频	
		0	除以 1 (16 MHz)
		1	除以 2 (8 MHz)
	4	**PORSEL**: POR 占空比选择	
		0	POR 可实现 100%占空比
		1	POR 可在 1/16 占空比下工作(这个特性无法仿真)

	3-0	Dongcheng Semiconductor 保留

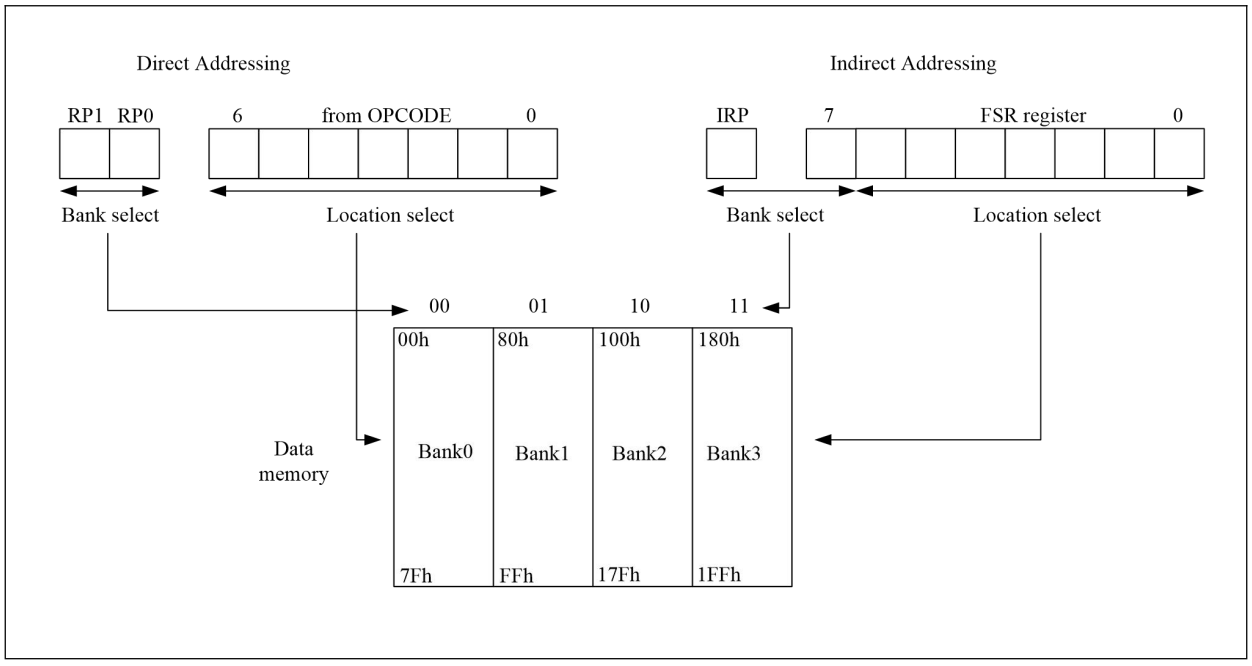
1.3 RAM 寻址模式

CPU 中有一个数据存储器分为四个部分 BANK。每个 BANK 最多可扩展到 7Fh (128 字节)。每个 BANK 的下部位置保留用于特殊功能寄存器 (SFR)。SFR 上方是通用寄存器，实现为静态 RAM。所有已实现的 BANK 都包含特殊功能寄存器。BANK 的一些常用特殊功能寄存器可能会在另一个 BANK 中进行镜像，以减少代码并加快访问速度。

RP1 和 RP0 位 (STATUS [6:5]) 是 BANK 选择位。

[RP1, RP0]	BANK
00	0
01	1
10	2
11	3

该页面可以直接或间接寻址。对 INDF 寄存器进行写值为间接寻址，INDF 寄存器不是实质寄存器，任何使用 INDF 寄存器的指令实际上都为访问文件选择寄存器 FSR 指向的寄存器。间接读取 INDF 寄存器本身 (FSR =“ 0”)将读为 00h，间接写入 INDF 寄存器将导致空操作（可能会影响状态位）。
通过将 8 位 FSR 寄存器和 IRP 位 (STATUS [7]) 连接起来，可以获得有效的 9 位地址。请参考下图。



直接 / 间接寻址

在 F/W 代码的开头保持 RP0=RP1=0 并使用新的指令集。
使用新指令的好处是用户可以忽略寄存器的 BANK 区位置，并且可以节省代码大小。新指令与旧指令几乎相同。通过将指令集中的 ”F” 替换为 ”X”，可以轻松使用新指令，而无需切换 BANK。

例如：

BC**F**	TM0IE	➔	BC**X**	TM0IE
DEC**F**	CNT, 1	➔	DEC**X**	CNT, 1
INC**F**SZ	RAM25, 0	➔	INC**X**SZ	RAM25, 0
MOVW**F**	PAMOD10	➔	MOVW**X**	PAMOD10
RL**F**	RAMA0, 0	➔	RL**X**	RAMA0, 0
SWAP**F**	ADCTL, 0	➔	SWAP**X**	ADCTL, 0

【BANK0】 000~07Fh		【BANK1】 080h~0FFh		【BANK2】 100h~17Fh		【BANK3】 180h~1FFh	
000h	**INDF**	080h	**INDF**	100h	**INDF**	180h	**INDF**
001h	TM0	081h	OPTION	101h	TM0	181h	OPTION
002h	**PCL**	082h	**PCL**	102h	**PCL**	182h	**PCL**
003h	**STATUS**	083h	**STATUS**	103h	**STATUS**	183h	**STATUS**
004h	**FSR**	084h	**FSR**	104h	**FSR**	184h	**FSR**
005h	PAD	085h	PAMOD10	105h	PINMOD	185h	DPL
006h	PBD	086h	PAMOD32	106h		186h	DPH
007h		087h	PAMOD54	107h		187h	CRCDL
008h		088h	PAMOD76	108h		188h	CRCDH
009h		089h	PWMCTL	109h	LVRPD	189h	CRCIN
00Ah	**PCLATH**	08Ah	**PCLATH**	10Ah	**PCLATH**	18Ah	**PCLATH**
00Bh	**INTIE**	08Bh	**INTIE**	10Bh	**INTIE**	18Bh	**INTIE**
00Ch	INTIF	08Ch	PBMOD10	10Ch	PCH	18Ch	TABR
00Dh	INTIE1	08Dh	PBMOD32	10Dh		18Dh	CMPCTL
00Eh	INTIF1	08Eh	PBMOD54	10Eh	BGTRIM	18Eh	CMPPNS
00Fh	CLKCTL	08Fh	PBMOD76	10Fh	IRCF	18Fh	DACTL
010h	TM0RLD	090h		110h	CFG0B	190h	
011h	TM0CTL	091h	OPTION2	111h	CFG02	191h	
012h	TM1	092h	PWMPRDH	112h	LDOCCTL	192h	
013h	TM1RLD	093h	PWMPRDL	113h	RDCTL	193h	
014h	TM1CTL	094h	PWM0DH	114h	IRCFT	194h	
015h	T2CTL	095h	PWM0DL	115h		195h	
016h	LVCTL	096h	PWM1DH	116h		196h	
017h	ADCDH	097h	PWM1DL	117h		197h	
018h	ADCTL	098h	PWM2DH	118h		198h	
019h	ADCTL2	099h	PWM2DL	119h		199h	
01Ah		09Ah	PWM3DH	11Ah		19Ah	
01Bh		09Bh	PWM3DL	11Bh		19Bh	
01Ch		09Ch	PWM4DH	11Ch		19Ch	
01Dh		09Dh	PWM4DL	11Dh		19Dh	
01Eh		09Eh	PWM5DH	11Eh		19Eh	
01Fh		09Fh	PWM5DL	11Fh		19Fh	
020h RAM Bank0 area (80 Bytes) 06Fh		0A0h RAM Bank1 area (80 Bytes) 0EFh		120h RAM Bank2 area (80 Bytes) 16Fh		1A0h Don't Use 1EFh	
070h **common area (16 Bytes)** 07Fh		0F0h **accesses 070h~07Fh** 0FFh		170h **accesses 070h~07Fh** 17Fh		1F0h **accesses 070h~07Fh** 1FFh	

◇ 范例: 通过使用直接寻址来读 / 写寄存器 **(RP0=RP1=0)**

```
CLKCTL      equ     00Fh        ; SFR in Bank0
TM1         equ     012h        ; SFR in Bank0
OPTION2     equ     091h        ; SFR in Bank1
LVRPD       equ     109h        ; SFR in Bank2
IRCF        equ     10Fh        ; SFR in Bank2
DPL         equ     185h        ; SFR in Bank3
RAM020      equ     020h        ; RAM in Bank0
RAM0A0      equ     0A0h        ; RAM in Bank1

MOVXW       TM1                 ; read TM1 (Bank0) to W
MOVXW       OPTION2             ; read OPTION2 (Bank1) to W
MOVXW       IRCF                ; read IRCF (Bank2) to W
MOVXW       DPL                 ; read DPL (Bank3) to W

MOVLW       16h                 ; W = 16h
MOVWX       RAM020              ; RAM[0x020] = W = 16h
MOVWX       RAM0A0              ; RAM[0x0A0] = W = 16h

MOVLW       37h                 ; W = 37h
MOVWX       LVRPD               ; LVRPD = W = 37h, force LVR/POR disable

MOVXW       CLKCTL              ; read SFR CLKCTL (00Fh) to W
MOVXW       IRCF                ; read SFR IRCF (10Fh) to W

MOVLW       0Bh                 ; W = 0Bh
MOVWX       CLKCTL              ; CLKCTL (00Fh) = W = 0Bh
MOVWX       IRCF                ; IRCF (10Fh) = W = 0Bh
```

◇ 范例: 通过使用间接寻址来读 / 写寄存器 **(RP0=RP1=0)**

```
BSX         IRP                 ; IRP = 1 => Bank2/3
MOVLW       0Fh                 ; W = 0Fh
MOVWX       FSR                 ; FSR = W = 0Fh
MOVXW       INDF                ; read SFR IRCF (10Fh) to W

BSX         IRP                 ; IRP = 1 => Bank2/3
MOVLW       0Fh                 ; W = 0Fh
MOVWX       FSR                 ; FSR = W = 0Fh
MOVLW       0Bh                 ; W = 0Bh
MOVWX       INDF                ; IRCF (10Fh) = W = 0Bh

BCX         IRP                 ; IRP = 0 => Bank0/1
MOVLW       0Fh                 ; W = 0Fh
MOVWX       FSR                 ; FSR = W = 0Fh
MOVXW       INDF                ; read SFR CLKCTL (00Fh) to W

BCX         IRP                 ; IRP = 0 => Bank0/1
MOVLW       0Fh                 ; W = 0Fh
MOVWX       FSR                 ; FSR = W = 0Fh
MOVLW       0Bh                 ; W = 0Bh
MOVWX       INDF                ; CLKCTL (00Fh) = W = 0Bh
```

1.4 程序计数器 (PC) 和堆栈

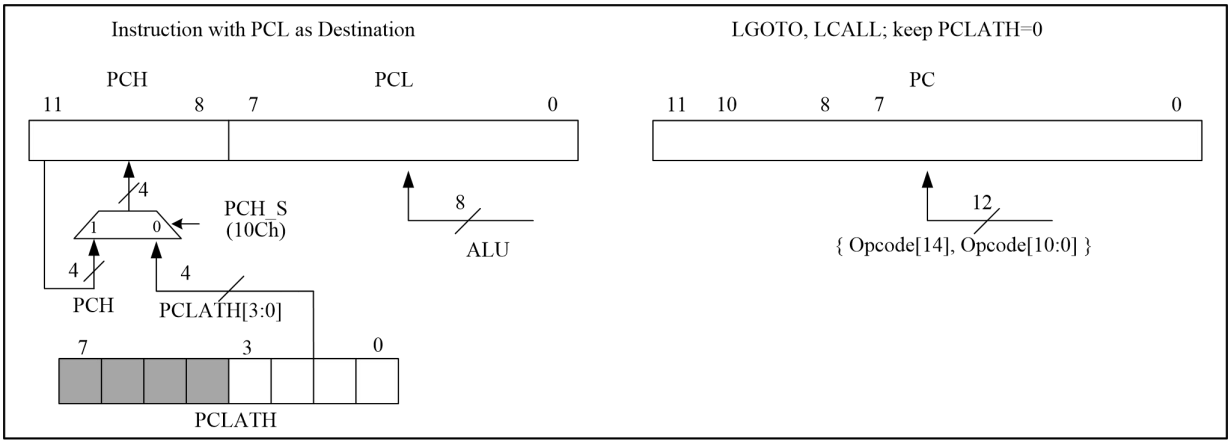
该程序计数器为 12 位的编程计数器，能够寻址一个 4Kx16 的 MTP ROM。当执行一个程序指令时，PC 将包含下一个要执行的程序指令的地址。除以下情况外，PC 值通常会增加 1。初始设置复位向量 (000h) 和中断向量 (004h) 用于 PC 初始化和中断。对于 CALL/GOTO 指令，PC 从指令字加载下 11 位地址，从 PCLATH[3] 加载高 1 位地址。对于 RET/RETI/RETLW 指令，PC 从顶级堆栈中检索其内容。

在执行 CALL/GOTO 指令之前，如果目标地址超过 2K，则必须设置 PCLATH[3]，否则必须清除 PCLATH[3]。与 RAM 寻址模式（参考 1.4 节）类似，芯片提供了新的指令集 LCALL/LGOTO 来替换 CALL/GOTO 指令集。当使用 LCALL/LGOTO 时，用户不必担心目的地地址，仅保持清除 PCLATH[3]。

编程计数器 (PC[7:0]) 的低字节数据可以通过 PCL 寄存器 (002h/082h/102h/182h) 进行读写。编程计数器 (PC[11:8]) 的高字节数据只能通过 PCH 寄存器 (10Ch) 读取。当执行以 PCL 寄存器为目标的任何指令时，内部标志 PCH_S 用于选择 PCH 的源。将 0x1C 写入 PCH 寄存器可以设置 PCH_S，将其他值写入 PCH 寄存器将清除 PCH_S。重置后，PCH_S 将被清除。

当 PCH_S 被清除为“0”时，同时执行任何以 PCL 寄存器为目标的指令，都会导致 PCH 被 PCLATH (00Ah/08Ah/10Ah/18Ah) 寄存器的内容所取代。这允许通过将所需的高字节写入 PCLATH 寄存器来更改程序计数器的整个内容。当低字节被写入 PCL 寄存器时，程序计数器的所有内容将变为 PCLATH 寄存器中包含的值和那些被写入 PCL 寄存器的值。

当 PCH_S 设置为 ‘1’ 时，执行任何以 PCL 寄存器为目标的指令，低字节将被写入 PCL 寄存器，并且不会改变 PCH。当使用任何以 PCL 寄存器为目标的指令时，建议将 PCH_S 设置为 ‘1’，但 C 语言不支持此函数。



002h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCL	PCL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

002h.7~0 **PCL:** 程序计数器数据位 7~0

00Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCLATH	GPR				PCLATH			
R/W	R/W				R/W			
Reset	0	0	0	0	0	0	0	0

00Ah.3~0 **PCLATH:** 当执行以 PCL 为目标的指令且 PCH_S 清零时编程计数器高字节数据
00Ah.3 **PCLATH:** 当执行 CALL/GOTO 指令时，编程计数器高 1 位
注意：当使用 LCALL/LGOTO 指令时，必须保持清除状态

10Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCH	PCH							
R/W	W				R/W			
Reset	0	0	0	0	0	0	0	0

10Ch.7~0 **PCH (W):** 当执行以 PCL 作为目标的指令时，编程计数器高字节源选择
写 0x1C 来设置 PCH_S = 1，PCH 保持原始值
写其他值来清除 PCH_S = 0，PCH 来自 PCLATH

10Ch.3~0 **PCH (R):** 编程计数器数据位 11~8

堆栈为 12 位宽度，8 阶深度。LCALL 指令和硬件中断将按顺序推入堆栈。当执行 RET/RETI/ RETLW 指令时按顺序弹堆栈。对于表查该器件提供了功能强大的表读取指令 TABRL 及 TABRH 通过设置 DPTR = {DPH, DPL} 寄存器将 16 位 ROM 数据返回到 W 寄存器。通过将 C 语言设置为 TABR (18Ch)，它还提供了另一种将 16 位 ROM 数据读入 W 寄存器的方法。

◇范例: 要查找位于“ TABLE1”和“ TABLE2”的 PROM 数据.

```
ORG         000h                        ; 复位向量
            LGOTO       START

START:
            MOVLW       00h
            MOVWX       RAM020          ; 设置查表地址
            MOVLW       1Ch             ; 写入 1Ch 到 PCH 以设置 PCH_S 标志
            MOVWX       PCH
LOOP:
            MOVXW       RAM020          ; 将索引值移至 W 寄存器
            LCALL       TABLE1          ; 查找数据
            …
            INCX        RAM020, 1       ; 增加下一个地址的索引地址
            …
            LGOTO       LOOP            ; 转到 LOOP 标签
            …
            MOVLW       (TABLE2 >>8) & 0xff
            MOVWX       DPH
            MOVLW       (TABLE2) & 0xff
            MOVWX       DPL             ; DPTR = {DPH, DPL} = TABLE2
; 通过指令 TABRL / TABRH 读表
            TABRL                       ; 将 PROM 低字节数据读到 W (W = 86h)
            TABRH                       ; 将 PROM 高字节数据读到 W (W = 19h)
            …
; 通过特殊功能寄存器 TABR 读表
            MOVLW       01h             ; TABR = 01h =  指令 TABRL
            MOVWX       TABR            ; 读取 PROM 低字节数据到 TABR (TABR = 86h)
            MOVXW       TABR            ; 读取 TABR 到 W (W = 86h)
            MOVLW       02h             ; TABR = 02h = 指令 TABRH
            MOVWX       TABR            ; 读取 PROM 高字节数据到 TABR (TABR = 19h)
            MOVXW       TABR            ; 读取 TABR 到 W (W = 19h)
            …
ORG         X00h
TABLE1:
            ADDWX       PCL, 1          ; Add the W with PCL, the result back in PCL.
            RETLW       55h             ; W=55h when return
            RETLW       56h             ; W=56h when return
            RETLW       58h             ; W=58h when return
            …
TABLE2:
            .DT         0x1986          ; 16-bit ROM data
            .DT         0x3719
```

…

注：芯片将 256 个 ROM 地址定义为一页，因此 ROM 有 16 页，000h~0FFh，100h~1FFh，…，F00h~FFFh。换句话说，PC 可以被定义为页面。查找表必须位于同一个页面上，以避免获取错误的数据。因此，例如，在 X00h (X = 1, 2, 3, ... , E, F) 启动一个查找表时，查找表最多有 255 个数据。如果查找表中的数据较少，则不需要将起始地址设置为 X00h，而只需要确认所有查找表数据都位于同一个页面上。

18Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TABR	TABR							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

18Ch.7~0 1. TABR 写入 01h = 指令 TABRL
2. TABR 写入 02h = 指令 TABRH
3. 在步骤 1 或步骤 2 之后，读取 TABR 以获取主 ROM 读表取值
在步骤 1 之后，读取 TABR 以获取 EEPROM 值（当 EEPEN = E2h 时）
ASM 的读表：使用 TABRL / TABRH 或 TABR
C 的读表：使用 TABR

ALU 和工作 (W) 寄存器

该 ALU 是 8 位宽的，能够进行加法、减法、移位和逻辑操作。在两个操作数指令中，通常有一个操作数是 W 寄存器，它是一个用于 ALU 操作的 8 位不可寻址寄存器。另一个操作数或者是文件寄存器，或者是直接常数。在单个操作数指令中，操作数是 W 寄存器或文件寄存器。根据所执行的指令，ALU 可能会影响状态寄存器中的携带(C)、数字携带(DC)和零(Z)标志的值。C 和 DC 标志分别作为减法借位和半借位。.

注意：借位状态和借位值相反。半借位状态和半借位值相反。

状态寄存器 (003h/083h/103h/183h)

此寄存器包含 ALU 的算术状态和重置状态。状态寄存器可以是任何指令的目的地，就像任何其他寄存器一样。如果状态寄存器是影响 Z、DC 或 C 位的指令的目的地，那么对这三个位的写入将被禁用。根据设备逻辑设置或清除这些位。因此，建议只使用 BCX、BSX 和 MOVWX 指令来更改状态寄存器，因为这些指令不会影响这些位。

STATUS	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W
Bit	描述							
7	**IRP**: 寄存器位于 Bank 的 选择位 (用于间接寻址) 0 = Bank 0,1 (000h - 0FFh) 1 = Bank 2,3 (100h - 1FFh)							
6:5	**RP1:RP0**: 寄存器位于 Bank 的选择位 (用于直接寻址) 00 = Bank 0 (000h - 07Fh) 01 = Bank 1 (080h - 0FFh) 10 = Bank 2 (100h - 17Fh) 11 = Bank 3 (180h - 1FFh) 每个 Bank 为 128 字节							

<table>
<tr><td>4</td><td>TO: 超时标志
0: 上电复位或 CLRWDT/SLEEP 指令后
1: WDT 超时</td></tr>
<tr><td>3</td><td>PD: 掉电标志
0: 上电复位或 CLRWDT 指令后
1: 执行 SLEEP 指令后</td></tr>
<tr><td>2</td><td>Z: 零标志
0: 逻辑运算的结果不为零
1: 逻辑运算的结果为零</td></tr>
<tr><td rowspan="3">1</td><td colspan="2">DC: Decimal Carry Flag or Decimal / Borrow Flag</td></tr>
<tr><td>ADD 指令</td><td>SUB 指令</td></tr>
<tr><td>0: 没进位
1: 低字节有进位</td><td>0: 低字节有借位
1: 没借位</td></tr>
<tr><td rowspan="3">0</td><td colspan="2">C: 进位标志或/借位标志</td></tr>
<tr><td>ADD 指令</td><td>SUB 指令</td></tr>
<tr><td>0: 没进位
1: MSB 有进位</td><td>0: MSB 有借位
1: 没借位</td></tr>
</table>

◇ 范例：将立即数据写入状态寄存器 STATUS 寄存器

```
MOVLW   00h
MOVWX   STATUS          ; 清除 STATUS 寄存器
```

◇ 范例：位寻址置位和清除 STATUS 寄存器

```
BSX     STATUS, 0       ; 设置 C=1
BCX     STATUS, 0       ; 清除 C=0
```

◇ 范例： 通过 BTXSS 指令确定 C 标志

```
BTXSS   STATUS, 0       ; 检查进位标志
GOTO    LABEL_1         ; 如果 C=0，跳转到 LABEL_1
GOTO    LABEL_2         ; 如果 C=1，跳转到 LABEL_2
```

2 复位

该器件有四种复位方式.

- 上电复位 (POR)
- 低电压复位 (LVR)
- 外部引脚复位 (XRST)
- 看门狗复位 (WDTR)

复位可以是由上电复位 (POR)、外部引脚复位 (XRST)、看门狗定时器复位 (WDTR) 或低电压复位 (LVR) 引起的。CFGWH 控制复位功能。复位后，SFR 返回其默认值，程序计数器 (PC) 被清除，并且系统从复位向量 000h 处开始运行。状态寄存器 (STATUS) 上的 TO 和 PD 标志指示系统复位状态。

2.1 上电复位 (POR)

在通电重置后，所有系统和外围控制寄存器将被设置为它们的默认硬件重置值。

2.2 低电压复位 (LVR)

低电压复位的功能是在电源电压低于阈值水平时进行静态复位。可以选择 16 个阈值级别。LVR 的操作模式由 CFGWH 寄存器定义。参见以下 LVR 选择表；用户还必须考虑工作频率的最低工作电压。

LVR 选择表：

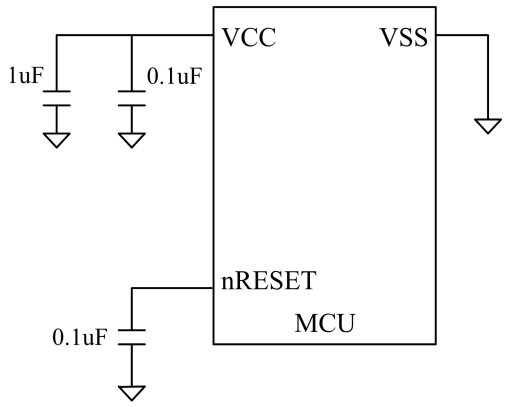
LVR level	Operating voltage
LVR2.05	5.5V > V_{CC} > 2.05V
LVR2.20	5.5V > V_{CC} > 2.20V
LVR2.30	5.5V > V_{CC} > 2.30V
LVR2.45	5.5V > V_{CC} > 2.45V
LVR2.60	5.5V > V_{CC} > 2.60V
LVR2.75	5.5V > V_{CC} > 2.75V
LVR2.90	5.5V > V_{CC} > 2.90V
LVR3.00	5.5V > V_{CC} > 3.00V
LVR3.15	5.5V > V_{CC} > 3.15V
LVR3.30	5.5V > V_{CC} > 3.30V
LVR3.45	5.5V > V_{CC} > 3.45V
LVR3.60	5.5V > V_{CC} > 3.60V
LVR3.70	5.5V > V_{CC} > 3.70V
LVR3.85	5.5V > V_{CC} > 3.85V
LVR4.00	5.5V > V_{CC} > 4.00V
LVR4.15	5.5V > V_{CC} > 4.15V

不同的 Fsys 有不同的最低工作电压，请参考直流特性的工作电压图表，如果所选择的 LVR 阈值级别低于 Fsys 最低工作电压，则系统可能进入死区并发生错误。

2.3 外部引脚复位 (XRST)

外部引脚复位可以通过 CFGWH 寄存器禁止或使能。它至少需要保持 2 个 SIRC 时钟周期才能被芯片所侦测到触发重置动作。XRST 将使所有控制寄存器恢复为其默认复位，而 TO/PD 标志不受复位的影响。

外部复位引脚为低电平有效，复位引脚为高电平时系统正在运行。复位引脚接收低电压系统复位。 外部复位可以在上电期间使系统复位，良好的外部复位电路可以保护系统以避免在异常电源条件下工作。



2.4 看门狗定时器复位 (WDTR)

WDT 溢出复位可通过 CFGWH 寄存器关闭或使能。它在快速/慢速模式运行，在空闲/停止模式运行或停止。WDT 溢出速度可以由 WDTPSC SFR 选择。WDT 通过器件复位或 CLRWDT SFR 位清除，WDT 溢出复位也将所有控制寄存器恢复为其默认复位值。TO/PD 标志不受复位的影响。

◇ 范例: 定义复位向量

```
        ORG     000h            ; 复位向量
        LGOTO   START           ; 跳转到用户程序地址

        ORG     010h
START:
        …                       ; 用户程序
        …
        LGOTO   START
```

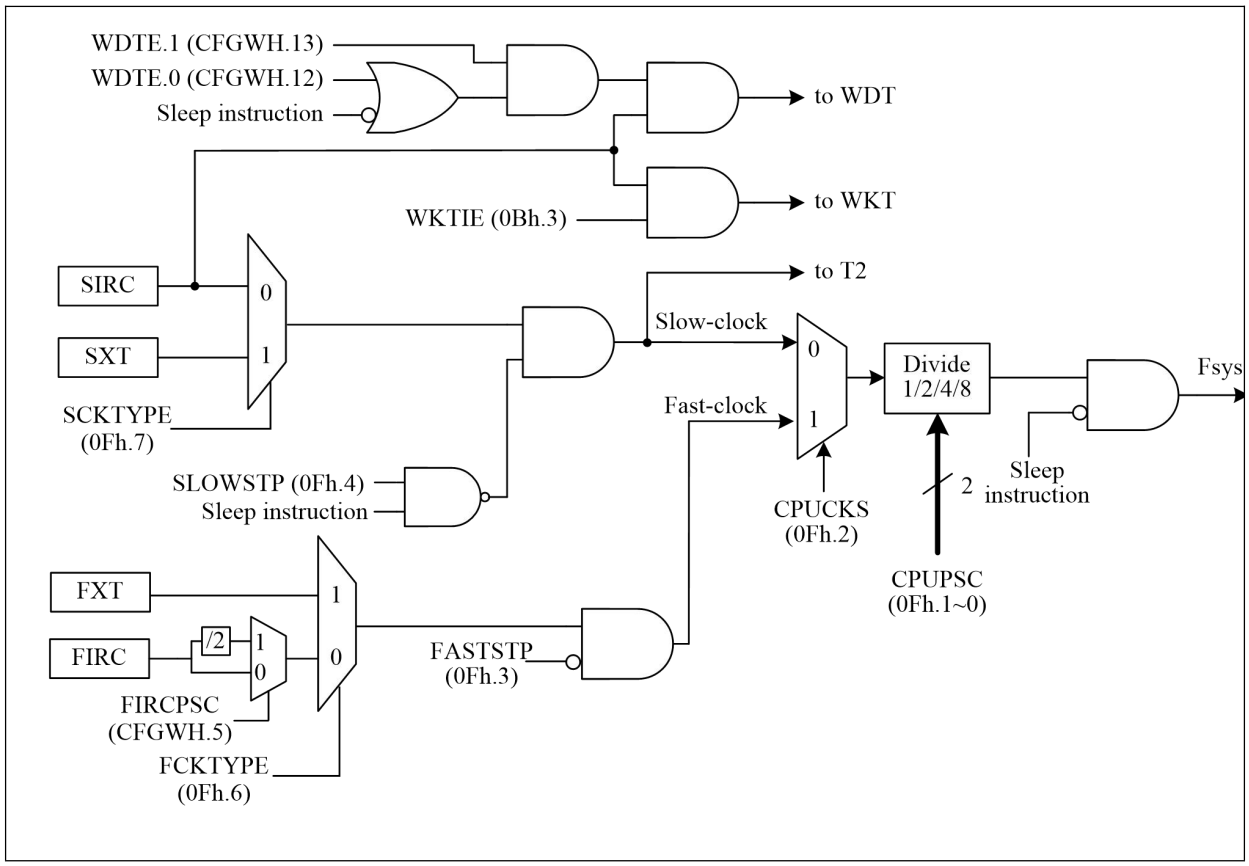
3 时钟电路和工作模式

3.1 系统时钟

该设备采用双时钟系统进行设计。时钟源有四种，FXT（外部快时钟）、SXT（外部慢时钟）、SIRC（内部RC慢时钟）和FIRC（内部RC快时钟）。每个时钟源都可以作为系统时钟应用于CPU内核。在空闲模式下，可以将慢时钟（SIRC 或 SXT）配置为保持振荡以为 T2 模块提供时钟源，或者 SIRC 为 WKT/WDT 模块提供时钟源。请参见下图。

重置后，设备以慢速模式85 KHz SIRC运行。为了芯片运行的安全，软件应选择合适的时钟速率。较高的V_{CC}允许芯片在更高的系统时钟频率下运行。在一个典型的情况下，一个16MHz的系统时钟速率需要V_{CC} > 1.9V。

CLKCTL (0Fh) SFR控制系统时钟的工作。硬件会自动阻止该寄存器的软件异常设置。不要同时写FASTSTP=1和CPUCKS=1。建议逐位写入此SFR。



时钟方案框图

FIRC（内部 RC 快时钟）的频率可以通过 IRCF (10Fh) 进行调整。当 IRCF=00h 时，频率最低。当 IRCF=7Fh 时，频率最高。因为每个 IC 可能具有不同的 IRCF 默认值，所以我们可以在上电后以确保上电复位后 FIRC=16MHz 的频率。

快速模式:

在这种模式下，将使用 FIRC 或 FXT 作为 CPU 时钟 (Fsys) 来执行程序。Timer0，Timer1 块由快时钟驱动。PWM0 块可以通过设置 PWMCKS (91h.5~4)选择 Fsys、FIRC (16 MHz) 或 FIRC*2 (32 MHz) 来驱动。T2 块通过设置 T2CKS (15h.3~2)，选择由慢时钟、Fsys/128 或 FIRC/512 (16 MHz/512) 来驱动。

慢速模式:

开机或复位后，设备进入慢速模式，默认的慢时钟为 SIRC。在此模式下，快时钟可以停止（通过 FASTSTP=1，相较省电）或运行（通过 FASTSTP=0）。所有外围块（Timer0、Timer1 等）时钟源都是慢时钟模式，除了 PWM 和 T2 块可以选择其他的时钟源。慢时钟可以选择 SIRC 或 SXT。

空闲模式:

执行 SLEEP 指令后，如果 SIRC 或 SXT 仍在振荡，则表示进入 IDLE 模式。IDLE 模式通过重置或启用中断唤醒而终止。有两种方法可以保持 SIRC 或 SXT 在空闲模式下振荡。

（1） 在执行 SLEEP 指令之前，清除 SLOWSTP=0, SIRC 或 SXT 仍然可以振荡。在这种情况下，慢时钟可以继续振荡，以保持 T2 块在空闲模式下持续运行；

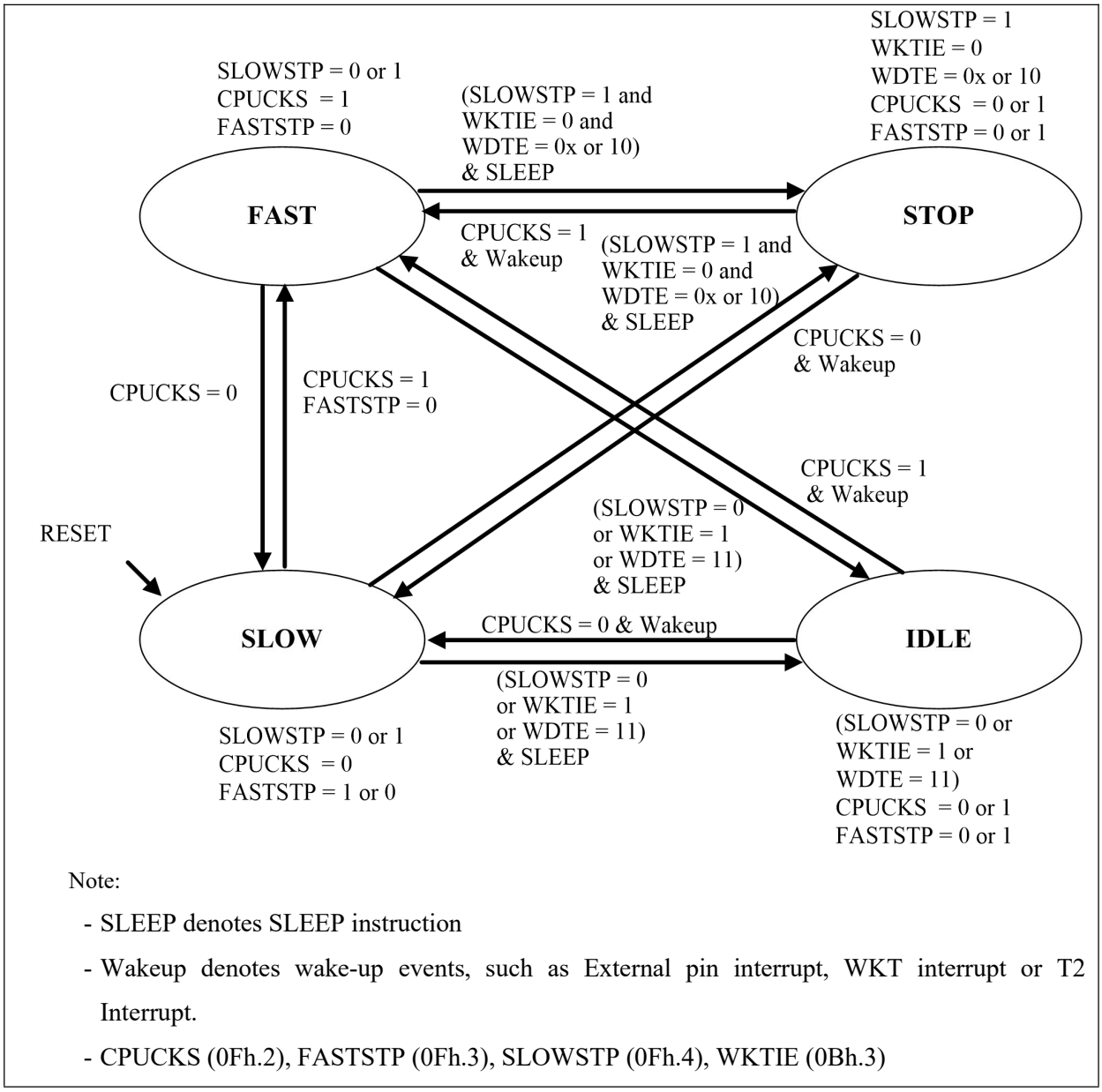
（2） 在执行 SLEEP 指令之前，设置 WKTIE=01b 或 WDTE=11b，SIRC 仍然可以振荡，以保持 WKT/WDT 在 IDLE 模式下运行。

停止模式:

在执行 SLEEP 指令之前将 SLOWSTP (0Fh.4) 置位，WKTIE (0Bh.3) 清零和 WDTE = 0xb 或 10b，则芯片将进入停止模式，所有模块都将停止。停止模式类似于空闲模式，不同之处在于，所有时钟振荡器（快时钟或慢时钟）均已停止，并且不产生任何时钟。

3.2 双系统时钟模式转换

该器件运行在以下四种模式之一：快速模式，慢速模式，空闲模式和停止模式。



Note:

- SLEEP denotes SLEEP instruction
- Wakeup denotes wake-up events, such as External pin interrupt, WKT interrupt or T2 Interrupt.
- CPUCKS (0Fh.2), FASTSTP (0Fh.3), SLOWSTP (0Fh.4), WKTIE (0Bh.3)

CPU 工作框图

CPU 模式和时钟菜单：

模式	Fsys	快时钟	慢时钟	TM0/TM1	T2	WKT	WDT	唤醒事件
快速	快时钟	运行	运行	运行	运行	运行	运行	X
慢速	慢时钟	Set by FASTSTP	运行	运行	运行	运行	运行	X
空闲	停止	停止	运行	停止	Set by T2CKS	Set by WKTIE	Set by WDTE	WKT/IO/T2
停止	停止	停止	停止	停止	停止	停止	停止	IO

- **快速模式切换至慢速模式**

当快速模式切换至慢速模式时，建议按顺序执行以下步骤：

(1) 切换至慢速模式 (CPUCKS=0)

(2) 停止快时钟 (FASTSTP=1)

◇ 范例：快速模式切换至慢速模式

```
BCX     CPUCKS      ; Fsys=慢时钟
BSX     FASTSTP     ; 停止快时钟
```

- **慢速模式切换至快速模式**

SLOW 模式可以通过 CLKCTL 寄存器中的 CPUCKS=0 来启用。当 SLOW 模式切换到 FAST 模式时，建议按顺序执行以下步骤：

(1) 启用快时钟 (FASTSTP=0)

(2) 切换到快速模式 (CPUCKS=1)

◇ 范例：慢速模式切换到快速模式

```
BCX     FASTSTP     ; 使能快时钟
NOP
BSX     CPUCKS      ; Fsys=快时钟
```

- **空闲模式设置**

IDLE 模式可以按顺序设置进行配置：

(1) 启用慢时钟 (SLOWSTP=0) or WKT (WKTIE=1) or WDT (WDTE=11b)

(2) 将 T2 时钟源切换为慢时钟 (T2CKS=0)

(3) 执行 SLEEP 指令

空闲模式可以通过外部中断、WKT 中断和 T2 中断来唤醒。

◇ 范例： 快速/慢速模式切换至空闲模式

```
BCX     SLOWSTP     ; 在执行 SLEEP 指令后慢时钟持续运行
MOVLW   0000__00__00b
MOVWX   T2CTL
SLEEP               ; 进入空闲模式
```

- **停止模式设置**

可以通过以下顺序设置停止模式：

(1) 停止慢时钟 (SLOWSTP=1)

(2) 停止 WKT (WKTIE=0)

(3) 执行 SLEEP 指令

停止模式只能被外部引脚中断唤醒。
备注： 当 WDTE=11b 时，CPU 无法进入停止模式。

◇ 范例：快速/慢速模式切换至停止模式

```
BSX     SLOWSTP      ; 在执行 SLEEP 指令后停止慢时钟.
MOVLW   0000 0000b   ; 关闭 WKT 计数
MOVWX   INTIE
SLEEP                ; 进入停止模式.
```

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.3 **WKTIE:** 唤醒定时器中断使能和唤醒定时器使能
0: 关闭
1: 使能

0Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CLKCTL	SCKTYPE	FCKTYPE	–	SLOWSTP	FASTSTP	CPUCKS	CPUPSC	
R/W	R/W	R/W	–	R/W	R/W	R/W	R/W	
Reset	0	0	–	0	1	0	1	1

0Fh.7 **SCKTYPE**: 慢时钟选择
0: 慢时钟为 SIRC
1: 慢时钟为 SXT

0Fh.6 **FCKTYPE**: 快时钟选择
0: 快时钟为 FIRC
1: 快时钟为 FXT

0Fh.4 **SLOWSTP**: 在 SLEEP 指令后停止慢时钟
0: 慢时钟在 SLEEP 指令后持续运行
1: 慢时钟在 SLEEP 指令后停止运行

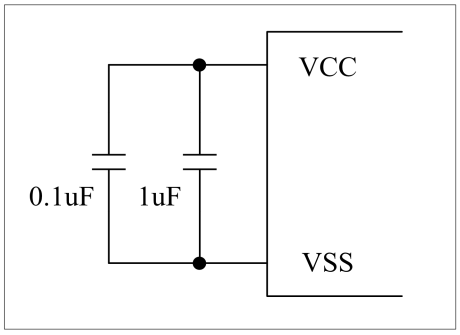
0Fh.3 **FASTSTP**: 停止快时钟
0: 快时钟运行
1: 快时钟停止

0Fh.2 **CPUCKS**: 系统时钟源选择
0: 慢时钟
1: 快时钟

0Fh.1~0 **CPUPSC**: 系统时钟源预分频器。系统时钟源
00: 除以 8　　01: 除以 4
10: 除以 2　　11: 除以 1

3.3 系统时钟振荡器

在内部快速 RC (FIRC) 模式下，片上振荡器产生 16 MHz 的系统时钟。由于电源噪声会降低内部时钟振荡器的性能，因此将电源旁路电容器 1 uF 和 0.1 uF 放置在靠近 VCC/VSS 引脚，可以提高时钟和整个系统的稳定性。



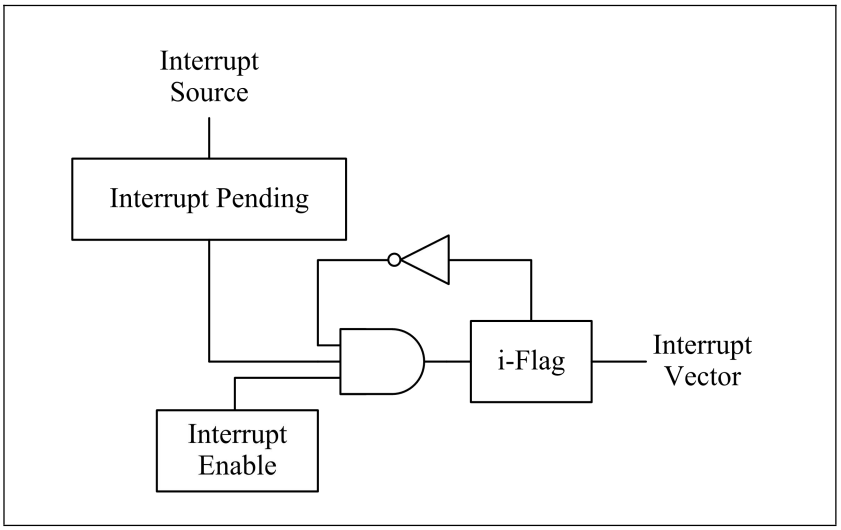
Internal RC Mode

4 中断

芯片有 1 级、1 个向量和 12 个中断源。每个中断源都有自己的使能控制位。无论其使能控制位是 0 还是 1，中断事件都将设置其各自的挂起标志。

如果相应的中断使能位 (INTIE[7:0], INTIE1[4], INTIE1[2:0]) 已置位，它将触发 CPU 来服务该中断。CPU 在当前执行的指令周期结束时接受中断。同时，向 CPU 插入 “LCALL 004” 指令，并设置 i-flag 以防止递归中断嵌套。

i-flag 在执行 “RETI” 指令之后被清除。也就是说，在中断服务未完成之前，主程序中至少有一条指令被执行。中断事件是电平触发的。F/W 在服务中断程序时必须清除中断事件寄存器。



◇ 范例：使用上升沿触发来设置 INT1 (PA1) 中断请求

```
            ORG     000h            ; 复位向量
            LGOTO   START           ; 跳转到用户程序地址

            ORG     004h            ; 所有中断的向量
            LGOTO   INT             ; 如果 INT1 (PA1) 输入出现上升沿

            ORG     005h
START:
            MOVLW   0000xxxxb
            MOVWX   PAMOD10         ; 选择模式0 作为 INT1 引脚模式
                                    ; 开漏输出低电平或上拉输入
            MOVLW   xxxxxx1xb
            MOVWX   PAD             ; 释放 INT1，变为施密特触发器
                                    ; 输入带上拉电阻
            MOVLW   xx1xxxxxb
            MOVWX   OPTION          ; 将 INT1 中断触发设置为上升沿
            MOVLW   111111**0**1b
            MOVWX   INTIF           ; 清除 INT1 中断请求标志
            MOVLW   0000001**0**b
            MOVWX   INTIE           ; 使能 INT1 中断
MAIN:
            …
            LGOTO   MAIN

INT:
            MOVWX   20h             ; 将 W 数据存储到 SRAM 20h
            MOVXW   STATUS          ; 获取 STATUS 数据
            MOVWX   21h             ; 将 STATUS 数据存储到 SRAM 21h

            BTXSC   INT1IF          ; 检测 INT1IF 位
            LCALL   INT1_SUB        ; INT1IF = 1, 跳转到 INT1 中断服务程序
            …                       ;
EXIT_INT:
            MOVXW   21h             ; 获取 SRAM 21h 数据
            MOVWX   STATUS          ; 恢复 STATUS 数据
            MOVXW   20h             ; 恢复 W 数据
            RETI                    ; 从中断返回

INT1_SUB:                           ; INT1 中断服务程序
            …
            MOVLW   111111**0**1b
            MOVWX   INTIF           ; 清除 INT1 中断请求标志
            RET
```

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.7 **ADCIE:** ADC 中断使能
0: 关闭
1: 使能

0Bh.6 **T2IE:** T2 中断使能
0: 关闭
1: 使能

0Bh.5 **TM1IE:** Timer1 中断使能
0: 关闭
1: 使能

0Bh.4 **TM0IE:** Timer0 中断使能
0: 关闭
1: 使能

0Bh.3 **WKTIE:** 唤醒定时器中断使能和唤醒定时器使能
0: 关闭
1: 使能

0Bh.2 **INT2IE:** INT2 中断使能
0: 关闭
1: 使能

0Bh.1 **INT1IE:** INT1 中断使能
0: 关闭
1: 使能

0Bh.0 **INT0IE:** INT0 中断使能
0: 关闭
1: 使能

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.7 **ADCIF:** ADC 中断事件挂起标志
当 ADC 转换结束后由 H/W 置位，对此位写 0 将清除该标志

0Ch.6 **T2IF:** T2 中断事件挂起标志
当 T2 溢出时由 H/W 置位，对此位写 0 将清除该标志

0Ch.5 **TM1IF:** Timer1 中断事件挂起标志
当 Timer1 溢出时由 H/W 置位，对此位写 0 将清除该标志

0Ch.4 **TM0IF:** Timer0 中断事件挂起标志
当 Timer0 溢出时由 H/W 置位，对此位写 0 将清除该标志

0Ch.3 **WKTIF:** 唤醒定时器中断事件挂起标志
当唤醒定时器超时时由 H/W 置位，对此位写 0 将清除该标志

0Ch.2 **INT2IF:** INT2 引脚下降沿中断挂起标志
当 INT2 引脚发生下降沿时由 H/W 置位，对此位写 0 将清除该标志

0Ch.1 **INT1IF:** INT1 引脚下降沿/上升沿中断挂起标志
当 INT1 引脚发生下降/上升沿时由 H/W 置位，对此位写 0 将清除该标志

0Ch.0 **INT0IF:** INT0 引脚下降沿/上升沿中断事件挂起标志
当 INT0 引脚发生下降/上升沿时由 H/W 置位，对此位写 0 将清除该标志

0Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	–	–	–	CMPIE	–	–	PWMIE	LVDIE
R/W	–	–	–	R/W	–	–	R/W	R/W
Reset	–	–	–	0	–	–	0	0

0Dh.4 **CMPIE:** 比较器中断使能
0: 关闭
1: 使能

0Dh.1 **PWMIE:** PWM 中断使能
0: 关闭
1: 使能

0Dh.0 **LVDIE:** LVD 中断使能
0: 关闭
1: 使能

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	–	–	–	CMPIF	–	–	PWMIF	LVDIF
R/W	–	–	–	R/W	–	–	R/W	R/W
Reset	–	–	–	0	–	–	0	0

0Eh.4 **CMPIF:** 比较器中断事件挂起标志
当 CMPO 匹配触发条件时由 H/W 置位，对此位写 0 将清除该标志

0Eh.1 **PWMIF:** PWM 中断事件等待标志
当 PWM 周期计数器翻转后由 H/W 置位，对此位写 0 将清除该标志

0Eh.0 **LVDIF:** LVD 中断事件等待标志
当 $V_{CC} < V_{LVD}$ 后由 H/W 置位，对此位写 0 将清除该标志

5 I/O 端口

5.1 PA0-PA7, PB0-PB2, PB4-PB6

每个 IO 都有 4 位作为模式设置。模式设置包括以下功能：开漏输出、CMOS 输出、上拉电阻、下拉电阻、引脚改变唤醒、PWMO 等等。所有 IO 都支持两个灌电流选项除了 PA7，由 HSINK (105h.2) 定义。**PA7 没有 1/2 偏压和高灌电流能力**。

这些引脚可以在不同的模式下操作，如下表所示：

PAxMOD PBxMOD	PADx PBDx	PA0~PA7, PB0~PB2, PB4~PB6 引脚功能	引脚状态	上拉电阻	数字输入	引脚改变 唤醒
0000b	0	开漏	低驱动	-	-	-
	1	输入	上拉	Y	Y	-
0001b	0	开漏	低驱动	-	-	-
	1	输入	高组态	-	Y	-
0010b	0	CMOS 输出	低驱动	-	-	-
	1		高驱动	-	-	-
0011b	X	模拟输入/输出 ADCx / CINx / CIPx / XT* / LDOC	高组态	-	-	-

*: XT 表示晶体振荡器

I/O 引脚菜单 1

PAxMOD PBxMOD	PADx PBDx	PA0~PA7, PB0~PB2, PB4~PB6 引脚功能	引脚状态	下拉电阻	数字输入	引脚改变 唤醒
0100b	0	开漏	低驱动	-	-	-
	1	输入	下拉	Y	Y	-
0101b	0	开漏	低驱动	-	-	-
	1	输入	高组态	-	Y	-
0110b	0	CMOS 输出	低驱动	-	-	-
	1		高驱动	-	-	-
0111b	X	功能 CMOS 输出 PWMx	-	-	-	-

I/O 引脚菜单 2

PAxMOD PBxMOD	PADx PBDx	PA0~PA7, PB0~PB2, PB4~PB6 引脚功能	引脚状态	上拉电阻	数字输入	引脚改变 唤醒
1000b	0	开漏	低驱动	-	-	-
	1	输入	上拉	Y	Y	Y
1001b	0	开漏	低驱动	-	-	-
	1	输入	高组态	-	Y	Y
1010b	0	CMOS 输出	低驱动	-	-	-
	1		高驱动	-	-	-
1011b		保留				

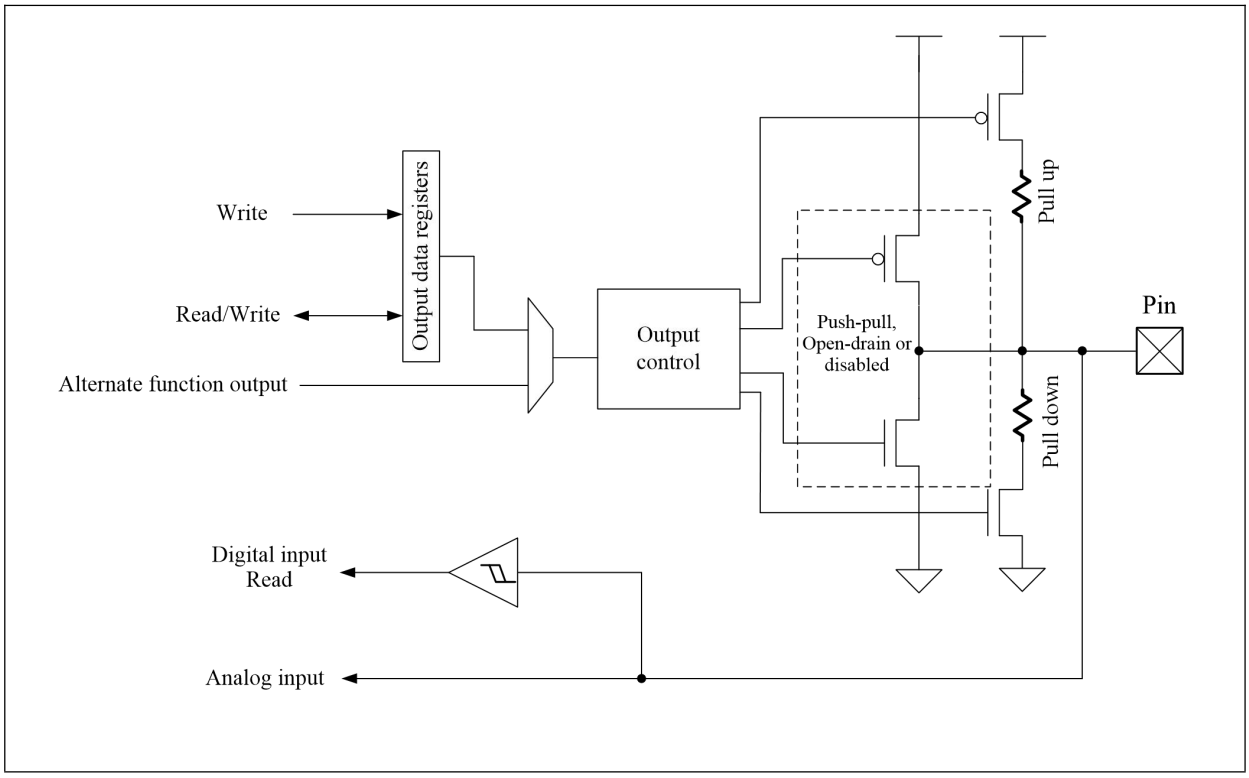
I/O 引脚菜单 3

PAxMOD PBxMOD	PADx PBDx	PA0~PA7, PB0~PB2, PB4~PB6 引脚功能	引脚状态	下拉电阻	数字输入	引脚改变 唤醒
1100b	0	开漏	低驱动	-	-	-
	1	输入	下拉	Y	Y	Y
1101b	0	开漏	低驱动	-	-	-
	1	输入	高组态	-	Y	Y
1110b	0	CMOS 输出	低驱动	-	-	-
	1		高驱动	-	-	-
1111b	X	模拟输出 1/2 V_{CC} (1/2 bias)(except PA7)	1/2 V_{CC}	-	-	-
		- (PA7)	上拉			

I/O 引脚菜单 4

引脚名称	PAxMOD / PBxMOD 设置		
	0011b (模拟输入/输出)	0111b (数字输出)	1111b (模拟输出)
PA0	ADC0 CIN2	PWM5O	1/2 bias
PA1	ADC1 CIP1	PWM1O	1/2 bias
PA2	ADC2 CIP2	PWM4O	1/2 bias
PA3	ADC3 CIN1 LDOC	PWM2O	1/2 bias
PA4	ADC4 XIN	PWM0P	1/2 bias
PA5	ADC5 XOUT	PWM3O	1/2 bias
PA6	ADC6	PWM0N	1/2 bias
PA7	-	-	Pull-up
PB0	ADC7	PWM1O	1/2 bias
PB1	ADC8	PWM2O	1/2 bias
PB2	ADC9	PWM3O	1/2 bias
PB4	ADC10 CIN4	PWM0P	1/2 bias
PB5	ADC11	PWM5O	1/2 bias
PB6	ADC12 CIP3	PWM0N	1/2 bias

PxxMOD 表的特殊功能



一般引脚结构

85h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMOD10	PA1MOD				PA0MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

86h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMOD32	PA3MOD				PA2MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

87h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMOD54	PA5MOD				PA4MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

88h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAMOD76	PA7MOD				PA6MOD			
R/W	R/W				R/W			
Reset	0	0	0	0	0	0	0	1

88h.7~4
88h.3~0
87h.7~4
87h.3~0
86h.7~4
86h.3~0
85h.7~4
85h.3~0

PA7MOD ~ PA0MOD: PA7~PA0 引脚模式控制
0000: 开漏或数字输入带上拉
0001: 开漏或数字输入
0010: CMOS 推挽输出
0011: 模拟输入/输出
0100: 开漏或数字输入带下拉(PA7没有下拉)
0101: 开漏或数字输入
0110: CMOS 推挽输出
0111: 外设功能输出

1000: 开漏或数字输入带上拉和引脚改变唤醒
1001: 开漏或数字输入和引脚改变唤醒
1010: CMOS 推挽输出
1011: 保留
1100: 开漏或数字输入带下拉和引脚改变唤醒(PA7没有下拉)
1101: 开漏或数字输入和引脚改变唤醒
1110: CMOS 推挽输出
1111: 1/2 V_{CC} (1/2 偏压) (除了 PA7) 或者上拉(PA7)

8Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMOD10	PB1MOD				PB0MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

8Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMOD32	-				PB2MOD			
R/W	-				R/W			
Reset	-				0	0	0	1

8Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMOD54	PB5MOD				PB4MOD			
R/W	R/W				R/W			
Reset	0	0	0	1	0	0	0	1

8Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBMOD76	-				PB6MOD			
R/W	-				R/W			
Reset	-				0	0	0	1

8Fh.3~0
8Eh.7~4
8Eh.3~0
8Dh.3~0
8Ch.7~4
8Ch.3~0

PB6MOD ~ PB4MOD, PB2MOD ~ PB0MOD: PB6~PB4 and PB2~PB0 引脚模式控制
0000: 开漏或数字输入带上拉
0001: 开漏或数字输入
0010: CMOS 推挽输出
0011: 模拟输入
0100: 开漏或数字输入带下拉
0101: 开漏或数字输入
0110: CMOS 推挽输出
0111: 外设功能输出
1000: 开漏或数字输入带上拉和引脚改变唤醒
1001: 开漏或数字输入和引脚改变唤醒
1010: CMOS 推挽输出
1011: 预留
1100: 开漏或数字输入带下拉和引脚改变唤醒
1101: 开漏或数字输入和引脚改变唤醒
1110: CMOS 推挽输出
1111: 1/2 V_{CC} (1/2偏压)

05h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PAD	PAD							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

05h.7~0 **PAD**: PA7~PA0 data

06h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBD	PBD							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

06h.7~0 **PBD**: PB7~PB0 数据

105h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PINMOD	–	–	Reserved	–	–	HSINK	Reserved	Reserved
R/W	–	–	R	–	–	R/W	R/W	R/W
Reset	–	–	0	–	–	1	0	0

105h.5 **Reserved**: 重置后读取为未知

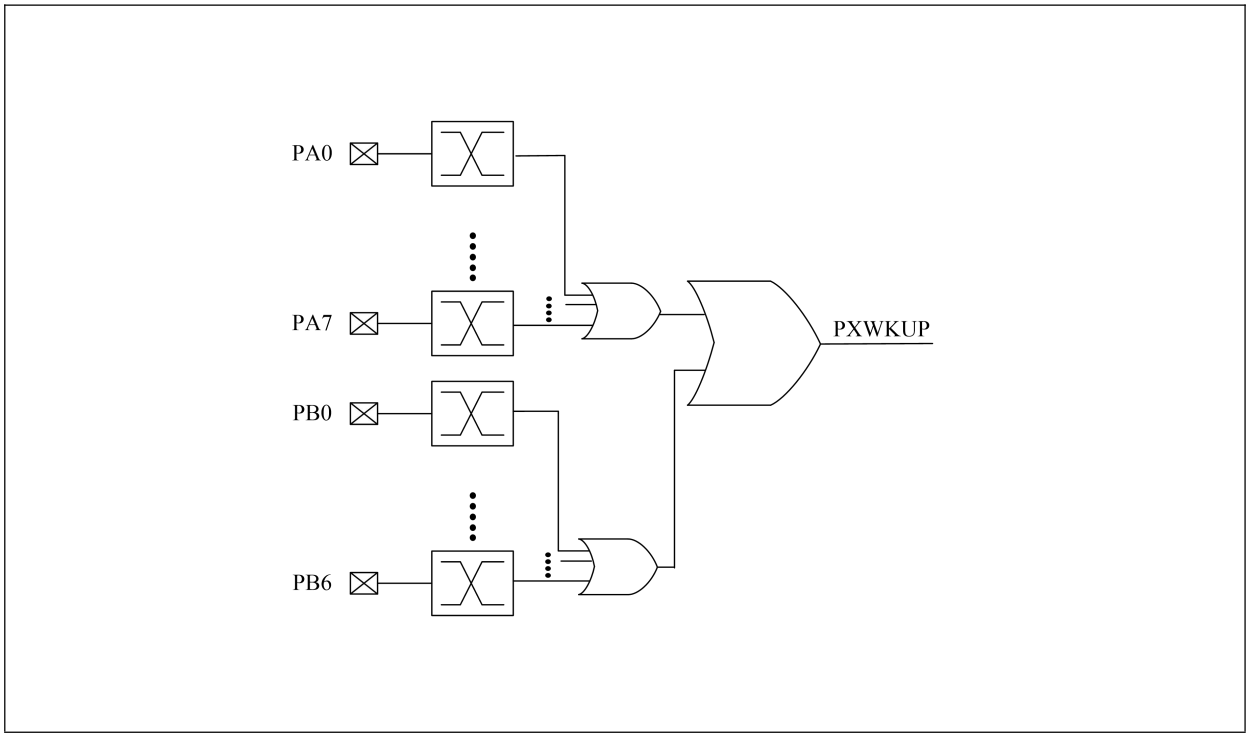
105h.2 **HSINK**: 所有 IO 端口高灌电流使能
0: 低灌电流
1: 高灌电流。（PA7无高灌电流能力）

105h.1 **Reserved**: 必须保持为0

105h.0 **Reserved**: 必须保持为0

5.2 引脚唤醒

所有的 IO 引脚都有引脚改变唤醒能力。

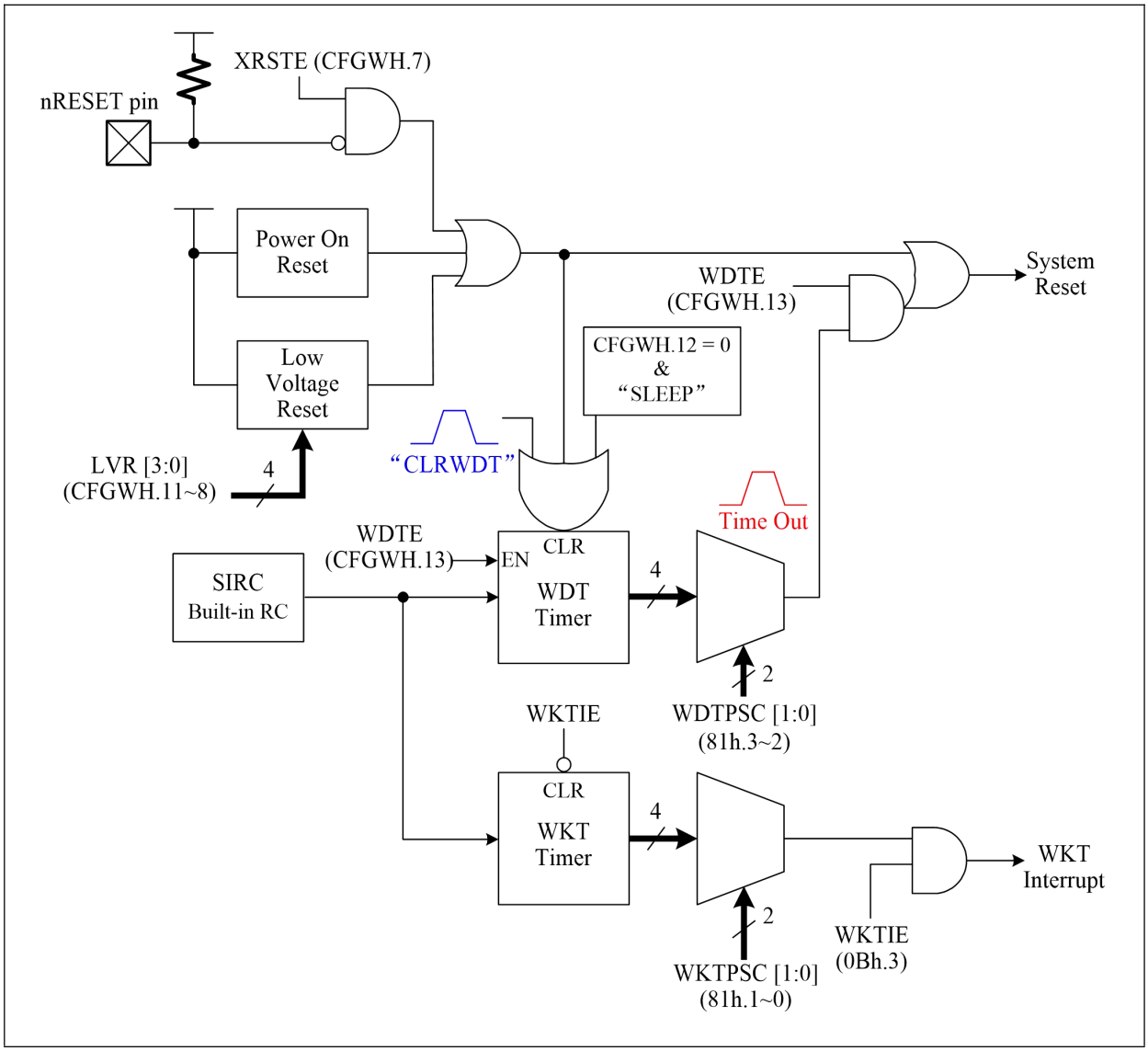


6 外围功能模块

6.1 看门狗 (WDT) /唤醒定时器 (WKT)

WDT 和 WKT 共享相同的内部 RC 振荡器，并有单独的计数器。WDT、WKT 的溢出周期可由个别预分频器 (WDTPSC[1:0], WKTPSC[1:0]) 进行选择。WDT 计时器由 CLRWDT 指令清除。如果启用了看门狗，则 WDT 将产生芯片复位信号。

WKT 计时器是一个间隔计时器，WKT 超时将生成 WKT 中断标志 (WKTIF)。WKT 定时器由 WKTIE=0 清除/停止。设置 WKTIE=1 时，CPU 无论在任何操作模式下，WKT 计时器都会一直计数。



WDT/WKT 框图

WDT 在不同模式下的行为如下表所示：

模式	CFGWH[13:12]		WDT
	WDTE[1]	WDTE[0]	
正常模式	0	0	停止
	0	1	停止
	1	0	运行
	1	1	运行
省电模式 (SLEEP)	0	0	停止
	0	1	停止
	1	0	停止
	1	1	运行

看门狗清除是由 CLRWDT 指令控制的。

◇ 范例： 通过 CLRWDT 指令清除看门狗计时器

```
MAIN:   …                       ; 执行程序
        CLRWDT                  ; 执行 CLRWDT 指令.
        …
        LGOTO   MAIN
```

◇ 范例：设置 WDT 时间

```
        MOVLW   0000**01**11b
        MOVWX   OPTION          ; 选择 WDT 超时=168 ms @5V
        …
```

◇ 范例：设置 WKT 周期和中断功能

```
        MOVLW   000001**10**b
        MOVWX   OPTION          ; 选择 WKT 周期=42 ms @5V
        MOVLW   1111**0**111b   ; 通过字节操作清除 WKT 中断请求标志
        MOVWX   INTIF           ; 不要使用位操作 “BCX WKTIF” 清除中断标志

        BSX     WKTIE           ; 使能 WKT 中断功能
```

03h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
STATUS	IRP	RP1	RP0	TO	PD	Z	DC	C
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

03h.4 **TO:** WDT 超时标志，只读
0: 上电复位或 CLRWDT / SLEEP 指令后
1: WDT 发生超时

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.3 **WKTIF:** 唤醒定时器中断事件挂起标志
当唤醒定时器超时时由 H/W 置位，对此位写 0 将清除该标志

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.3 **WKTIE:** 唤醒定时器中断使能和唤醒定时器使能
0: 关闭
1: 使能

81h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION	HWAUTO	INT0EDG	INT1EDG	–	WDTPSC		WKTPSC	
R/W	R/W	R/W	R/W	–	R/W		R/W	
Reset	0	0	0	–	1	1	1	1

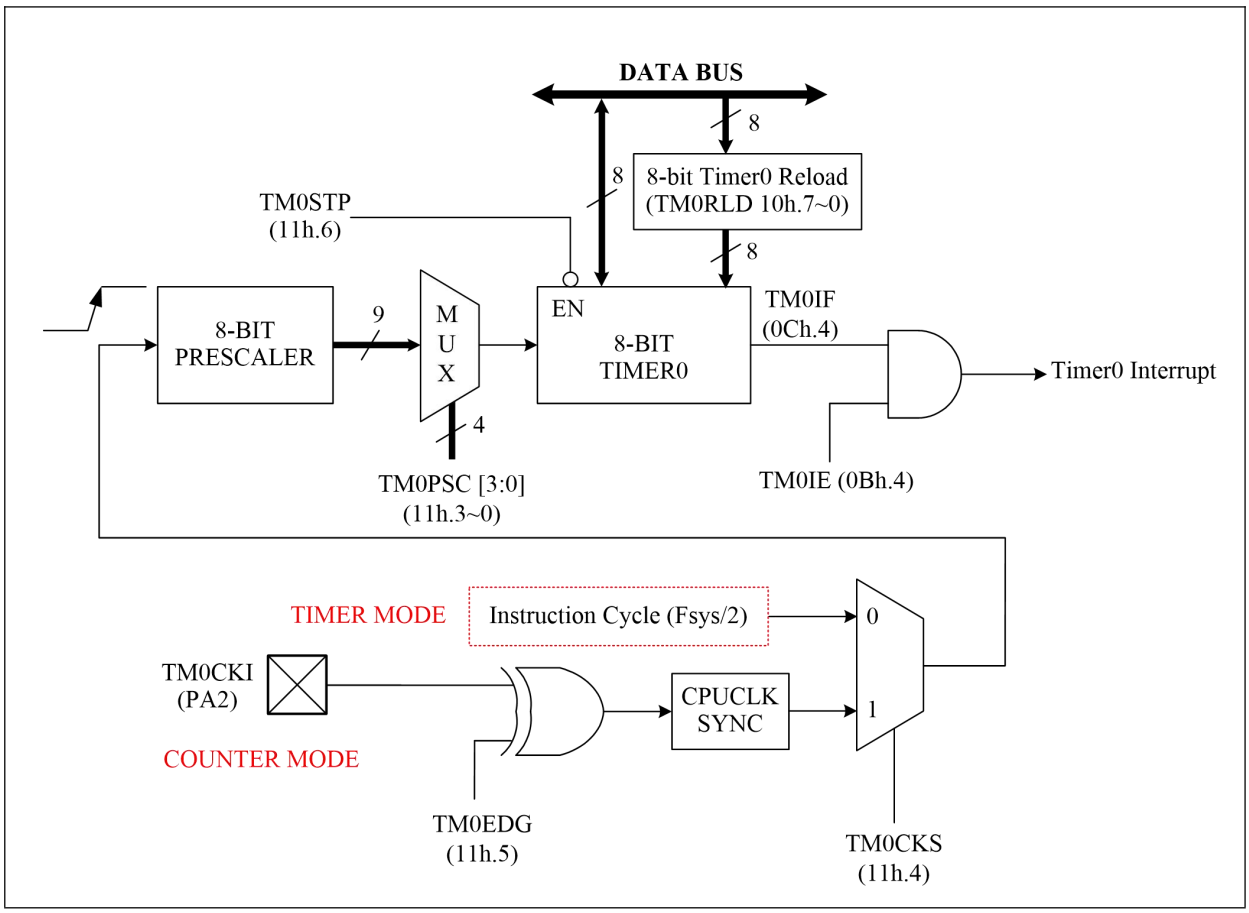
81h.3~2 **WDTPSC:** WDT 周期 (@V_{CC}=5V)
00: 84 ms
01: 168 ms
10: 672 ms
11: 1344 ms

81h.1~0 **WKTPSC:** WKT 周期 (@V_{CC}=5V)
00: 10.5 ms
01: 21 ms
10: 42 ms
11: 84 ms

6.2 Timer0

TM0 (01h.7~0) 是一个 8 位宽的寄存器。和其他任何寄存器一样可以读取或写入。此外，Timer0 会周期性地增加自身并自动翻转一个新的“偏移值”(TM0RLD)，同时它会根据预先调整的时钟源进行翻转，该时钟源可以是 Fsys/2 或 TM0CKI (PA2) 上升/下降输入。

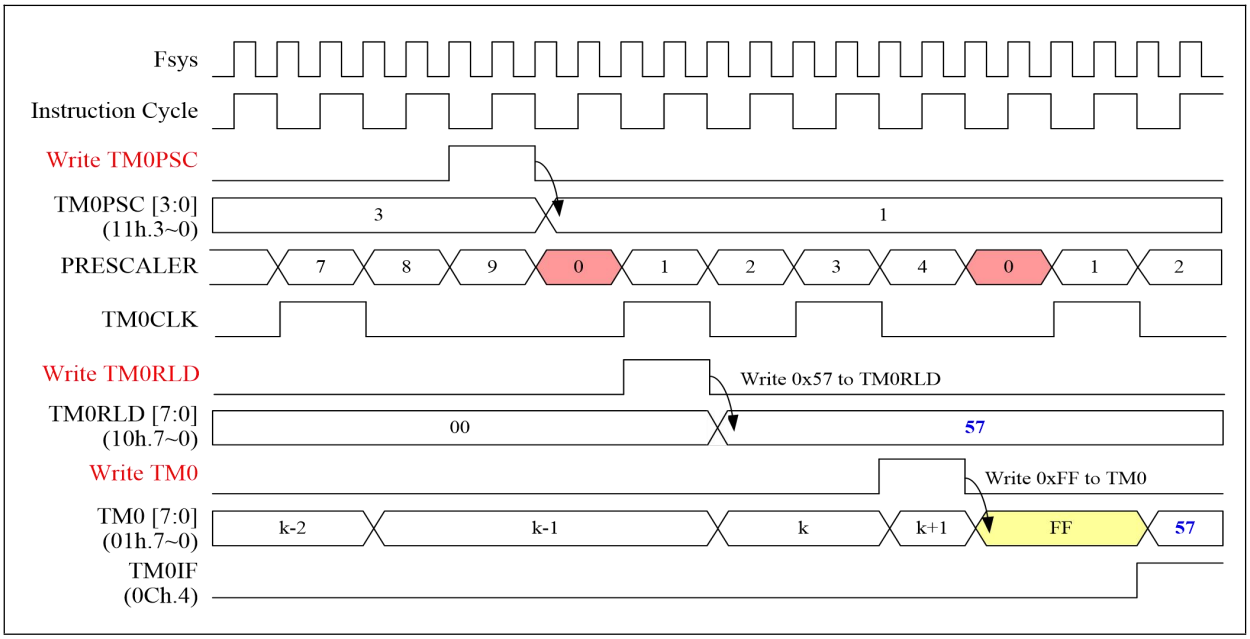
Timer0 的增加速率由“Timer0 预分频” (TM0PSC) 寄存器决定。当 Timer0 的计数翻转时，它总是产生 TM0IF (0Ch.4)。如果 TM0IE (0Bh.4) 置位，它会产生 Timer0 中断。如果 TM0STP (11h.6) 位置位，Timer0 可以停止计数。



Timer0 框图

下面的时序图描述了 Timer0 在纯定时器模式下工作。

当写入 Timer0 预除器 (TM0PSC) 时，内部的 8 位预分频器将被清除到 0，确保第一次 Timer0 计数时计数周期正确。TM0CLK 是导致 Timer0 在 TM0CLK 结束时增加 1 的内部信号。TM0WR 也是指示 Timer0 被指令直接写入的内部信号；同时，内部 8 位预分频器将被清零。当 Timer0 从 FFh 计数到 TM0RLD 时，如果 TM0IE（Timer0 中断使能）置位，TM0IF（Timer0 中断标志）将被设置为 1 并产生中断。



Timer0 在定时器模式下工作 (TM0CKS=0)

Timer0 中断时间的计算公式如下：

Timer0 中断间隔周期时间 = Fsys / 2 / TM0PSC / (256-TM0RLD)

◇ 范例: 如果 Fsys=8 MHz，设置 Timer0 在定时器模式下工作

```
; 设置 Timer0 时钟源和分频器
        MOVLW   00x00101b       ; TM0CKS=0，Timer0 时钟为指令周期
        MOVWX   TM0CTL          ; TM0PSC = 0101b, 除以 32

; 设置 Timer0 重新加载数据
        MOVLW   80H
        MOVWX   TM0RLD          ; 设置 Timer0 重载数据 = 128

; 设置 Timer0
        BSX     TM0STP          ; 停止 Timer0 计数
        CLRX    TM0             ; 清除 Timer0 内容

; 使能 Timer0 中断功能和启用 Timer0 计数
        MOVLW   1110 1111B
        MOVWX   INTIF           ; 清除 Timer0 中断请求标志
        BSX     TM0IE           ; 使能 Timer0 中断功能
        BCX     TM0STP          ; 使能 Timer0 计数
```

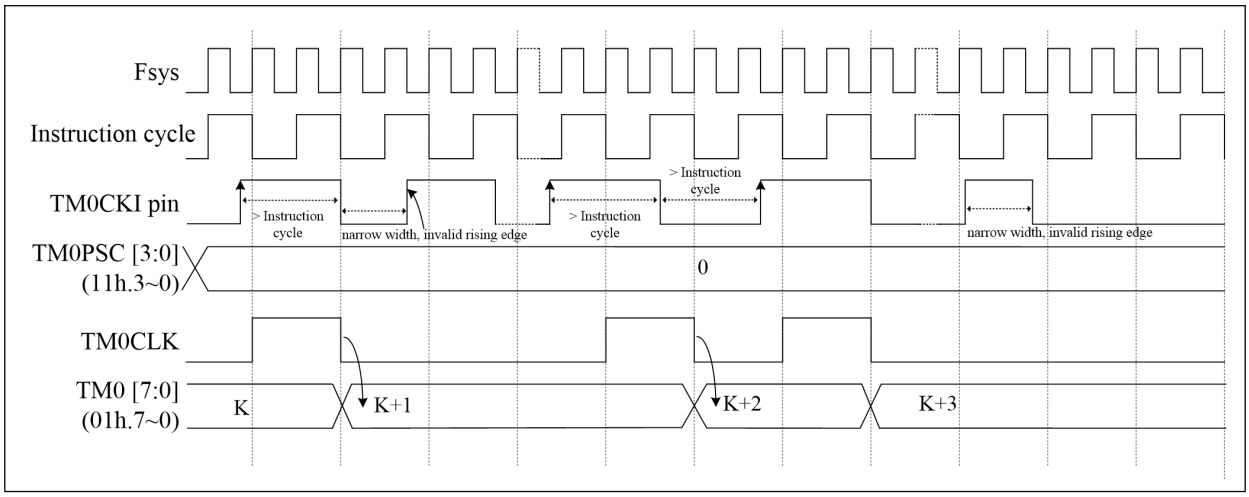
Timer0 中断频率计算公式 = Fsys / 2 / TM0PSC / (256-TM0RLD)

Fsys = 8MHz, TM0PSC = 除以 32, TM0RLD = 128

Timer0 中断频率 = 8 MHz / 2 / 32 / (256-128) = 0.976 KHz

下图描述了 Timer0 在计数器模式下的工作。

如果 TM0CKS=1，则 Timer0 计数器的源时钟来自 TM0CKI 引脚。TM0CKI 信号通过指令周期 (Fsys/2) 进行同步，这意味着 TM0CKI 的高/低持续时间必须长于一个指令周期时间 (Fsys/2)，以确保同步器正确检测到每个 TM0CKI 的变化。



Timer0 在 TM0CKI (TM0EDG=0)，TM0CKS=1 的计数器模式下工作

◇ 范例: 设置 Timer0 在计数器模式下工作，并且时钟源来自 TM0CKI 引脚 (PA2)

```
; 设置 Timer0 时钟源和分频器
        MOVLW   00110000B       ; TM0EDG = 1，为下降沿计数
        MOVWX   TM0CTL          ; TM0CKS = 1，Timer0 时钟为 TM0CKI
                                ; TM0PSC = 0000b, 除以1
; 设置 Timer0
        BSX     TM0STP          ; Timer0 停止计数
        CLRX    TM0             ; 清除 Timer0 内容

; 使能 Timer0 计数并读取 Timer0 计数器
        BCX     TM0STP          ; 使能 Timer0 计数
        …
        BSX     TM0STP          ; Timer0 停止计数
        MOVXW   TM0             ; 读取 Timer0 内容
```

01h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0	TM0							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

01h.7~0 **TM0:** Timer0 计数数据

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.4 **TM0IE:** Timer0 中断使能
0: 关闭
1: 使能

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.4 **TM0IF:** Timer0中断事件挂起标志
当 Timer0 溢出时由 H/W 置位，对此位写 0 将清除该标志

10h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0RLD	TM0RLD							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

10h.7~0 **TM0RLD:** Timer0 重载数据

11h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM0CTL	–	TM0STP	TM0EDG	TM0CKS	TM0PSC			
R/W	–	R/W	R/W	R/W	R/W			
Reset	–	0	0	0	0	0	0	0

11h.6 **TM0STP:** 停止 Timer0
0: Timer0 运行
1: Timer0 停止

11h.5 **TM0EDG:** TM0CKI 引脚的 Timer0 预分频器计数沿
0: 上升沿
1: 下降沿

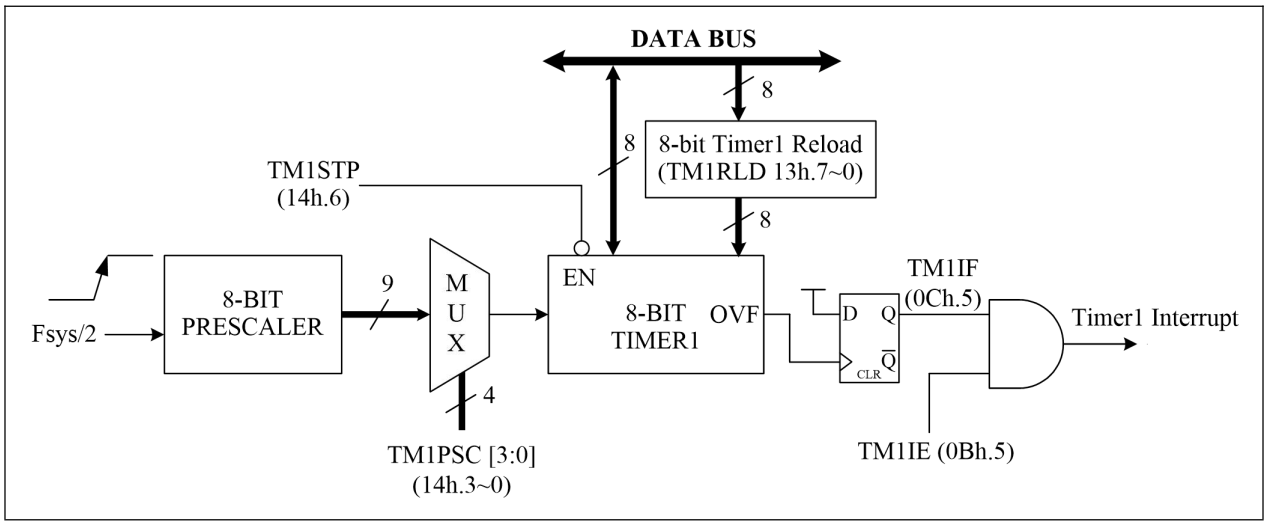
11h.4 **TM0CKS:** Timer0 预分频器时钟源
0: Fsys/2
1: TM0CKI 引脚 (PA2 引脚)

11h.3~0 **TM0PSC:** Timer0 预分频器。Timer0 预分频器时钟源除以

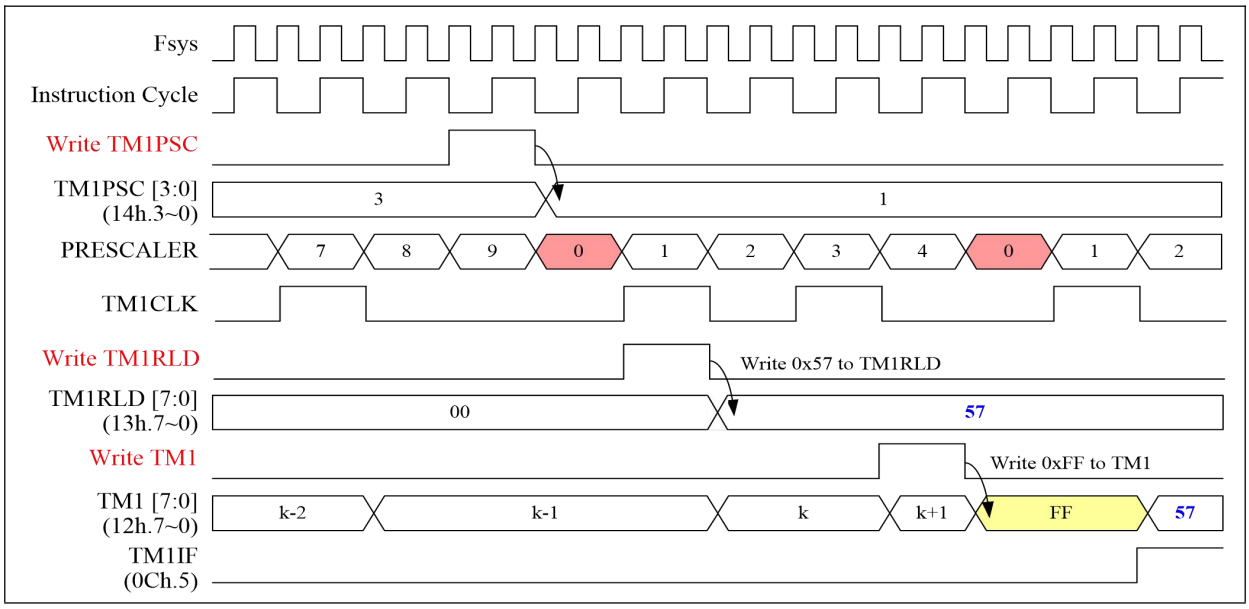
0000: 1	0001: 2	0010: 4	0011: 8
0100: 16	0101: 32	0110: 64	0111: 128
1xxx: 256			

6.3 Timer1

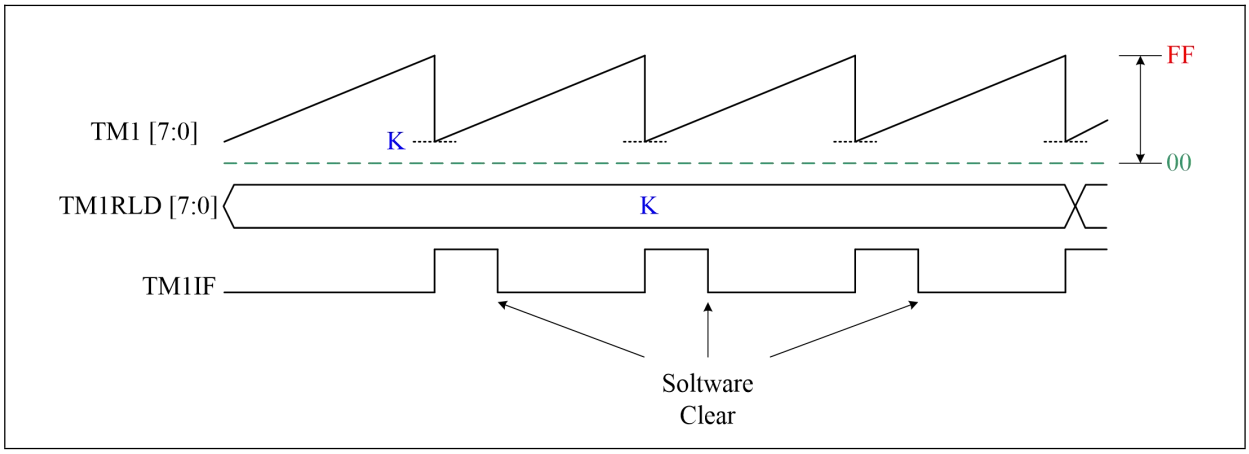
TM1 (12h.7~0) 是一个 8 位宽的寄存器。和任何其他寄存器一样可以读取或写入。此外，Timer1 会定期自行递增，并根据预分频的指令时钟 (Fsys/2) 在翻转时自动重载新的“偏移值” (TM1RLD）。Timer1 的增加速率由 TM1PSC 寄存器选择。如果 TM1IE 位置 1，它将产生 Timer1 中断。如果 TM1STP 位置 1，可以停止 Timer1 的计数。



Timer1 框图



Timer1 时序图



Timer1 重新加载图

◇ 范例：CPU 以慢时钟模式运行, Fsys = 慢时钟 / CPUPSC = 85 KHz / 2 = 42.5 KHz

```
; 设置 Timer1 时钟源和分频器
        MOVLW   0000[0011]b
        MOVWX   TM1CTL          ; TM1PSC = 0011b,  除以 8

; 设置 Timer1 重新加载数据
        MOVLW   FFh
        MOVWX   TM1RLD          ; 设置 Timer1 重载数据 = 255

; 设置 Timer1
        BSX     TM1STP          ; Timer1 停止计数
        CLRX    TM1             ; 清除 Timer1 内容

; 使能 Timer1 计数和中断功能
        MOVLW   11[0]11111b
        MOVWX   INTIF           ; 清除 Timer1 请求中断标志
        BSX     TM1IE           ; 使能 Timer1 中断功能
        BCX     TM1STP          ; 使能 Timer1 计数
```

Timer1 中断频率计算公式= Fsys / 2 / TM1PSC / (256-TM1RLD)
Fsys = 42.5 KHz, TM1PSC = div 8, TM1RLD = 255
Timer1 中断频率 = 42.5 KHz / 2 / 8 / (256-255) = 2.656 KHz

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.5 **TM1IE:** Timer1 中断使能
0: 关闭
1: 使能

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.5 **TM1IF:** Timer1 中断事件挂起标志
当 Timer1 溢出时由 H/W 置位，对此位写 0 将清除该标志

12h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1	TM1							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

12h.7~0 **TM1:** Timer1 计数数据

13h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1RLD	TM1RLD							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

13h.7~0 **TM1RLD:** Timer1 重载数据

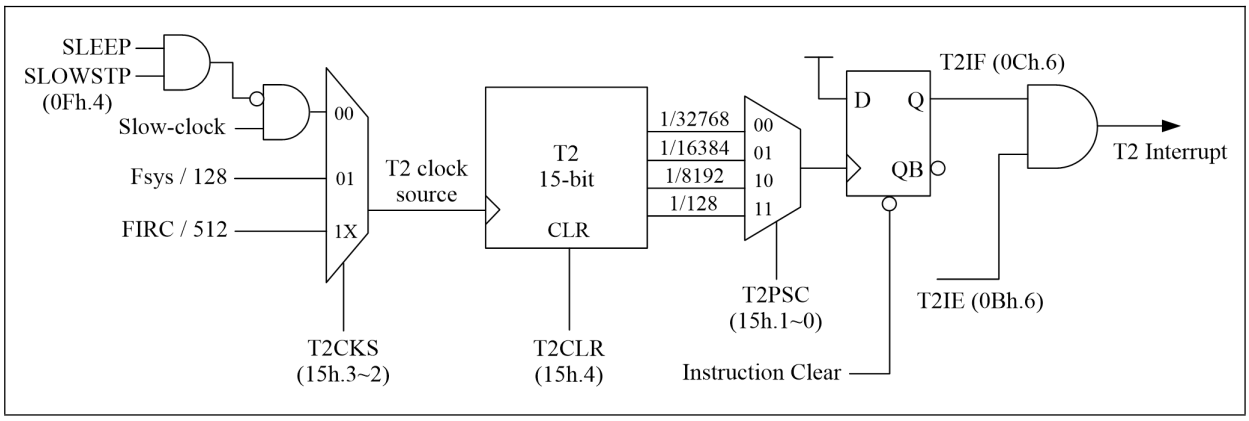
14h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TM1CTL	–	TM1STP	–	–	TM1PSC			
R/W	–	R/W	–	–	R/W			
Reset	–	0	–	–	0	0	0	0

14h.6 **TM1STP:**停止 Timer1
0: Timer1 运行
1: Timer1 停止

14h.3~0 **TM1PSC:** Timer1 预分频器。Timer1 预分频器时钟源除以
0000: 1 0001: 2 0010: 4 0011: 8
0100: 16 0101: 32 0110: 64 0111: 128
1xxx: 256

6.4 T2:15-bit Timer

T2 是一个 15 位计数器，时钟源来自慢时钟、Fsys/128 或 FIRC/512 (16 MHz/512)。它用于生成时基中断和 T2 计数器模块时钟。指令无法读取 T2 内容。T2 时钟频率取决于寄存器 T2PSC[1:0] (15h.1～0)位，可以是除以 32768/16384/8192/128 的其一，T2 计时溢出将产生中断标志 T2IF (0Ch.6)。T2 功能如下框图所描述。



T2 框图

◇范例: CPU 以快速模式运行，Fsys = 快时钟 / CPUPSC = FIRC / 2 = 8 MHz

```
; 设置 T2 时钟源和分频器
        MOVLW   0000**0101**b       ; T2CKS(15h.3~2) = 1, T2 时钟源为 Fsys/128
        MOVWX   T2CTL               ; T2PSC(15h.1~0) = 1, 除以 16384
        BSX     T2CLR               ; T2CLR = 1, 清除 T2 内容

; 使能 T2 中断功能
        MOVLW   1**0**111111b
        MOVWX   INTIF               ; 清除 T2 中断请求标志
        BSX     T2IE                ; 使能 T2 中断功能
        BCX     T2CLR               ; T2CLR = 0, 使能 T2 计数
```

T2 时钟源是 Fsys / 128 = 8 MHz / 128 = 62500 Hz, T2PSC = 16384

T2 频率 = 62500 Hz / 16384 = 3.815 Hz

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.6 **T2IE:** T2 中断使能
0: 关闭
1: 使能

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.6 **T2IF:** T2 中断事件挂起标志
当 T2 溢出时由 H/W 置位，对此位写 0 将清除该标志

0Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CLKCTL	SCKTYP	FCKTYPE	–	SLOWSTP	FASTSTP	CPUCKS	CPUPSC	
R/W	R/W	R/W	–	R/W	R/W	R/W	R/W	R/W
Reset	0	0	–	0	1	0	1	1

0Fh.4 **SLOWSTP:** 在执行 SLEEP 指令后停止慢时钟
0: 慢时钟在执行 SLEEP 指令后持续运行
1: 慢时钟在执行 SLEEP 指令后停止运行

15h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T2CTL	–	–	–	T2CLR	T2CKS		T2PSC	
R/W	–	–	–	R/W	R/W		R/W	
Reset	–	–	–	0	0	0	0	0

15h.4 **T2CLR:** 清除并停止 T2
0: T2 运行
1: T2 清除并停止

15h.3~2 **T2CKS:** T2 时钟源选择
00: 慢时钟
01: Fsys/128
1x: FIRC/512 (16 MHz/512)

15h.1~0 **T2PSC:** T2 预分频器。T2 时钟源 除以
00: 32768
01: 16384
10: 8192
11: 128

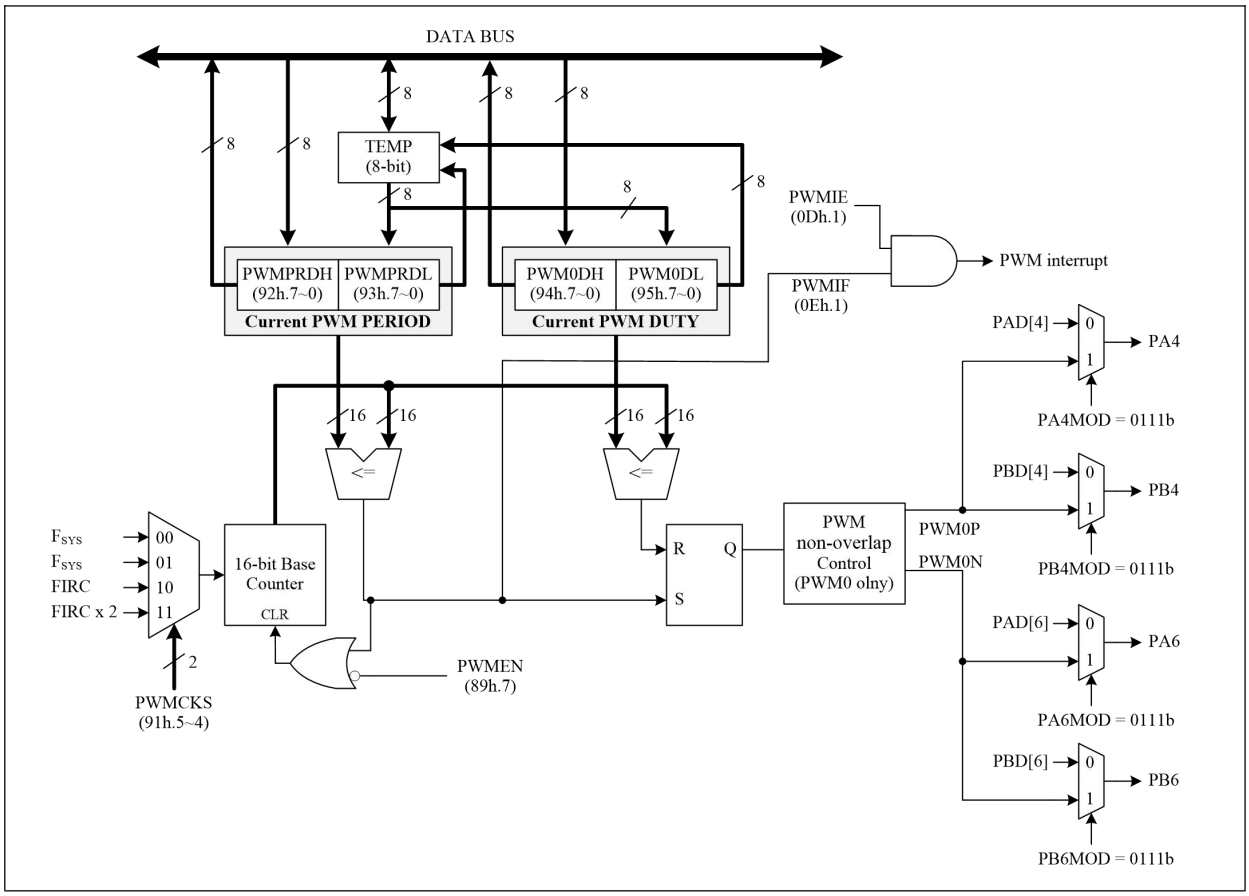
6.5 PWM: 16 bits PWM

这个芯片里有 6 个 PWM。PWM0~PWM5 具有独立的 16 位占空比控制寄存器，并共享一组 16 位周期寄存器。PWM 可以根据 PWM 时钟产生具有 65536 占空比分辨率的变化频率波形。PWM 时钟可选择 Fsys、FIRC (16 MHz) 或 FIRC*2 (32 MHz) 作为其时钟源。下面以 PWM0 为例进行说明。

16 位 PWMPRD、PWM0D 寄存器都具有低字节和高字节结构。高字节可以直接访问，但低字节只能通过内部的 8 位缓冲区访问，必须以特定的方式对这些寄存器对进行读取或写。需要注意的一点是，8 位缓冲区及其相关低字节的数据传输只有在执行相应高字节的写或读操作时才会发生。***简单地说，先写低字节，然后再写高字节；先读高字节，然后再读低字节。***

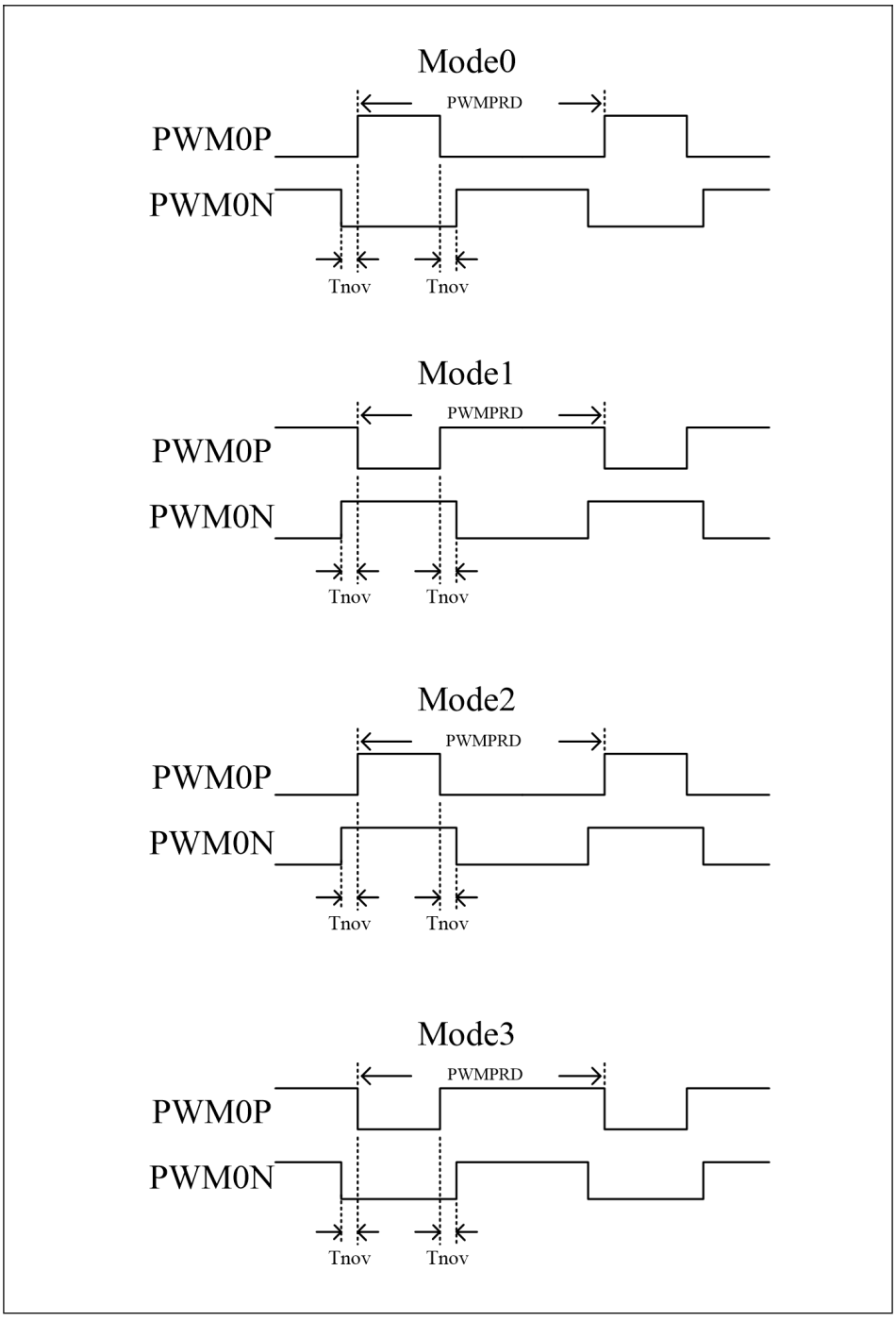
如果清除了 PWMEN，则将清除并停止 PWM0~5，否则 PWM0~5 将继续运行。PWM0 的结构如下所示。PWM0 的占空比可以通过写入 PWM0DH 和 PWM0DL 来更改。当 16 位基计数器与 16 位 PWM0 占空比寄存器 {PWM0DH，PWM0DL} 匹配时，PWM0 输出信号将重置到低电平。PWM0 周期可以通过将周期值写入 PWMPRDH 和 PWMPRDL 寄存器来设置。在写入 PWM0DH 或 PWMPRDH 寄存器后，H/W 将立即更新 PWM 周期。PWM0~5 共享一个中断标志，并在该期间结束时生成一个中断标志。

只有 PWM0 具有死区控制，可分为 PWM0P 和 PWM0N 输出，其余的 PWM1~PWM5 为简单 PWM 控制。PWM1~5 输出为 PWM1O~PWM5O。用户可以使用引脚模式设置将 PWMxO 输出到相应的 IO 引脚，有关引脚设置的更多信息，请参阅第 5 章。



PWM0 框图

只有 PWM0 可以通过 PWM0P 和 PWM0N 输出四种不同的模式。PWM 脉冲的边缘可以用 16 个不同的非重叠时钟间隔 (Tnov) 来分离。在 0~15 脉宽调制钟内，PWM0DZ (89h.3~0) 可以选择 Tnov 的宽度。默认的输出形式是 Mode0。四种输出模式的波形如下图所示。



PWM0 时序图

◇ 范例：

```
; 设置引脚模式
        MOVLW   xxxx0111b       ;
        MOVWX   PAMOD54         ; PA4 引脚作为 PWM0P

        MOVLW   xxxx0111b       ;
        MOVWX   PAMOD76         ; PA6 引脚作为 PWM0N

; 设置 PWM0 时钟源选项
        MOVLW   xx10xxxxb       ;
        MOVWX   OPTION2         ; FIRC 16 MHz 作为 PWM 时钟源

; 设置 PWM0 周期和占空比
        MOVLW   FFh
        MOVWX   PWMPRDL         ; 写入顺序: PWMPRDL 然后 PWMPRDH
        MOVLW   7Fh
        MOVWX   PWMPRDH         ; 设置 PWM 周期 = 7FFFh

        MOVLW   00h
        MOVWX   PWM0DL          ; 写入顺序: PWM0DL 然后 PWM0DH
        MOVLW   40h
        MOVWX   PWM0DH          ; 设置 PWM0 占空 = 4000h

; 设置 PWM0 使能和死区控制
        MOVLW   10000000b       ; 89h.7 = 1, PWM0 使能
        MOVWX   PWMCTL          ; 89h.5~4 = 0, PWM0 模式0 输出
                                ; 89h.3~0 = 0, PWM0 死区输出禁止
```

◇ 范例：

PWM0 时钟源 = FIRC 16 MHz, PWM 周期 = 7FFFh, PWM 占空比= 4000h

PWM0 输出频率 = 16 MHz / (周期+1) = 16 MHz / 32768 = 488 Hz

PWM0 输出占空比 = 占空比 / (周期+1) = 50%

0Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	–	–	–	CMPIE	–	–	PWMIE	LVDIE
R/W	–	–	–	R/W	–	–	R/W	R/W
Reset	–	–	–	0	–	–	0	0

0Dh.1 **PWMIE:** PWM 中断使能
0: 关闭
1: 使能

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	–	–	–	CMPIF	–	–	PWMIF	LVDIF
R/W	–	–	–	R/W	–	–	R/W	R/W
Reset	–	–	–	0	–	–	0	0

0Eh.1 **PWMIF:**PWM 中断事件等待标志
当 PWM 周期计数器翻转后由 H/W 设置，对此位写 0 将清除该标志

89h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMCTL	PWMEN	–	PWM0OM		PWM0DZ			
R/W	R/W	–	R/W		R/W			
Reset	0	–	0	0	0	0	0	0

89h.7 **PWMEN:** PWM0 ~PWM5 使能
0: 关闭
1: 使能

89h.5~4 **PWM0OM:** PWM0 输出模式选择
00: 模式 0
01: 模式 1
10: 模式 2
11: 模式3

89h.3~0 **PWM0DZ:** PWM0 非重叠控制
0000: 没有非重叠
0001: 非重叠宽度为 1 个 PWM 时钟周期
0010: 非重叠宽度为 2 个 PWM 时钟周期
...
1111: 非重叠宽度为 15 个 PWM 时钟周期

91h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OPTION2	–	–	PWMCKS		–	INT2SEL	INT1SEL	INT0SEL
R/W	–	–	R/W		–	R/W	R/W	R/W
Reset	–	–	0	0	–	0	0	0

91h.5~4 **PWMCKS:** PWM 时钟源选择
00: Fsys
01: Fsys
10: FIRC (16 MHz)
11: FIRC x 2 (32 MHz). Refer to the graph of minimal operating voltage for PWMCKS=FIRC x 2.

92h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMPRDH	PWMPRDH							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

92h.7~0 **PWMPRDH:** PWM0~5 周期高字节
写顺序: PWMPRDL 然后 PWMPRDH
读顺序: PWMPRDH 然后 PWMPRDL

93h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWMPRDL	PWMPRDL							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

93h.7~0 **PWMPRDL:** PWM0~5 周期低字节
写顺序: PWMPRDL 然后 PWMPRDH
读顺序: PWMPRDH 然后 PWMPRDL

94h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DH	PWM0DH							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

94h.7~0 **PWM0DH:** PWM0 占空高字节
写顺序: PWMxDL 然后 PWMxDH
读顺序: PWMxDH 然后 PWMxDL

95h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM0DL	PWM0DL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

95h.7~0 **PWM0DL:** PWM0 占空低字节
写顺序: PWMxDL 然后 PWMxDH
读顺序: PWMxDH 然后 PWMxDL

96h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1DH	PWM1DH							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

96h.7~0 **PWM1DH:** PWM1 占空高字节
写顺序: PWMxDL 然后 PWMxDH
读顺序: PWMxDH 然后 PWMxDL

97h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM1DL	PWM1DL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

97h.7~0 **PWM1DL:** PWM1 占空低字节
写顺序: PWMxDL 然后 PWMxDH
读顺序: PWMxDH 然后 PWMxDL

98h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM2DH	PWM2DH							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

98h.7~0 **PWM2DH**: PWM2 占空高字节
写顺序: PWMxDL 然后 PWMxDH
读顺序: PWMxDH 然后 PWMxDL

99h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM2DL	PWM2DL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

99h.7~0 **PWM2DL**: PWM2 占空低字节
写顺序: PWMxDL 然后 PWMxDH
读顺序: PWMxDH 然后 PWMxDL

9Ah	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM3DH	PWM3DH							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

9Ah.7~0 **PWM3DH**: PWM3 占空高字节
写顺序: PWMxDL 然后 PWMxDH
读顺序: PWMxDH 然后 PWMxDL

9Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM3DL	PWM3DL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

9Bh.7~0 **PWM3DL**: PWM3 占空低字节
写顺序: PWMxDL 然后 PWMxDH
读顺序: PWMxDH 然后 PWMxDL

9Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM4DH	PWM4DH							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

9Ch.7~0 **PWM4DH:** PWM4 占空高字节
写顺序: PWMxDL 然后 PWMxDH
读顺序: PWMxDH 然后 PWMxDL

9Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM4DL	PWM4DL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

9Dh.7~0 **PWM4DL:** PWM4 占空低字节
写顺序: PWMxDL 然后 PWMxDH
读顺序: PWMxDH 然后 PWMxDL

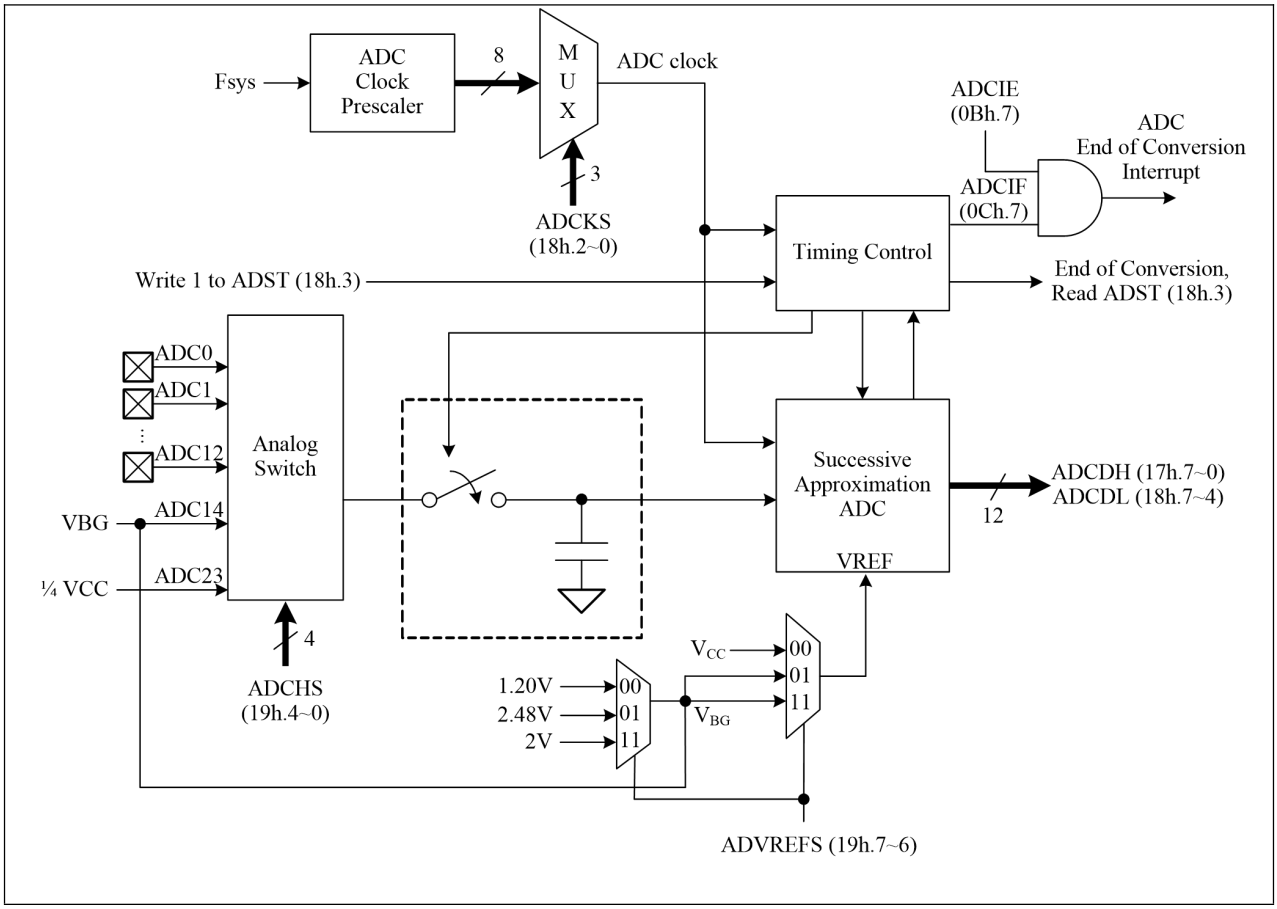
9Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM5DH	PWM5DH							
R/W	R/W							
Reset	1	0	0	0	0	0	0	0

9Eh.7~0 **PWM5DH:** PWM5 占空高字节
写顺序: PWMxDL 然后 PWMxDH
读顺序: PWMxDH 然后 PWMxDL

9Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PWM5DL	PWM5DL							
R/W	R/W							
Reset	0	0	0	0	0	0	0	0

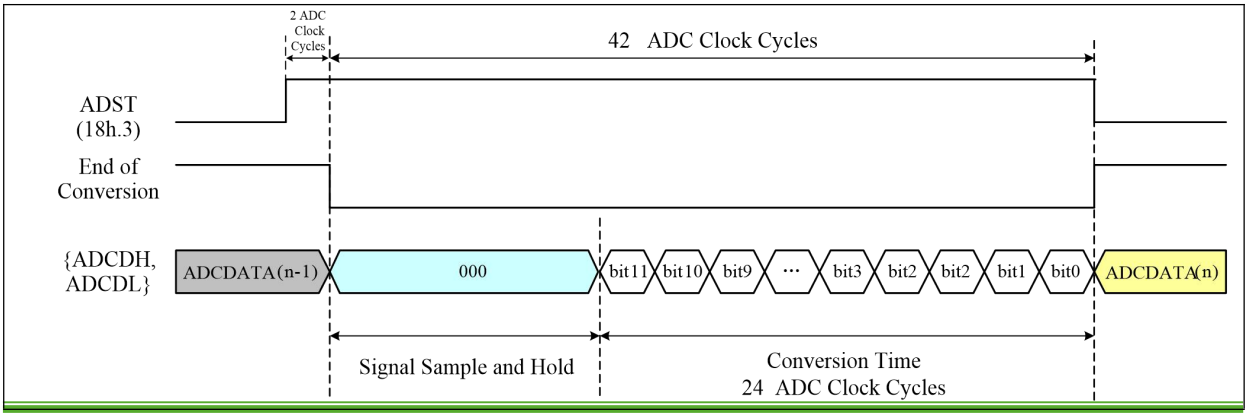
9Fh.7~0 **PWM5DL:** PWM5 占空低字节
写顺序: PWMxDL 然后 PWMxDH
读顺序: PWMxDH 然后 PWMxDL

6.6 模数转换器



该 12 位 ADC（模拟到数字转换器）由一个 17 通道的模拟输入多路复用器、控制寄存器、时钟发生器、12 位逐次逼近寄存器和输出数据寄存器组成。

要使用 ADC，用户需要设置 ADCKS (18h.2~0) 来选择一个合适的 ADC 时钟频率，该频率必须小于 1 MHz。然后用户通过设置 ADST (18h.3) 控制位来启动 ADC 转换。转换结束后，H/W 自动清除 ADST (18h.3) 位。用户可以通过轮询这一位以了解转换状态。当使用 IO 引脚作为 ADC 输入引脚时，相应的引脚模式应设置为 0011b。用户需要设置 ADCHS (19h.4~0)，以选择 ADC 的输入通道。此外，还可以选择一些参考输入通道，ADC14 为 VBG，ADC15 为 OPO，ADC23 为 1/4VCC。ADC 参考电压可以通过 ADVREFS (19h.7~6) 配置为 V_{CC} 或 V_{BG}，此外为 ADVREFS=01b 则需要 200uS 的预热时间。



◇范例：

[CPU 运行在快速模式 , Fsys = FIRC 16 MHz]
ADC 时钟频率 = 1 MHz, ADC 通道 = ADC2 (PA2)

◇ 范例：

```
        MOVLW   xxxx0011b       ; ADC2 (PA2) 为 ADC 输入引脚
        MOVWX   PAMOD32

        MOVLW   00000100b       ; ADCKS = Fsys/16, ADC clock = 1 MHz
        MOVWX   ADCTL

        MOVLW   00x00010b       ; ADC 参考电压选择 VCC
        MOVWX   ADCTL2          ; ADC 输入通道选择 ADC2

        BSX     ADST            ; 设置 18h.3 (ADST), 开始 ADC 转换

WAIT_ADC:
        BTXSC   ADST            ; 等待ADC转换完成.
        LGOTO   WAIT_ADC

        MOVXW   ADCDH           ; 读取 ADC 输出数据位 11~4
        MOVXW   ADCTL           ; 读取 ADC 输出数据位 3~0
        …
```

0Bh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE	ADCIE	T2IE	TM1IE	TM0IE	WKTIE	INT2IE	INT1IE	INT0IE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Bh.7 **ADCIE:** ADC 中断使能
0: 关闭
1: 使能

0Ch	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF	ADCIF	T2IF	TM1IF	TM0IF	WKTIF	INT2IF	INT1IF	INT0IF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

0Ch.7 **ADCIF:** ADC 中断事件挂起标志
当 ADC 转换结束后由 H/W 置位，对此位写 0 将清除该标志

17h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCDH	ADCDH							
R/W	R							
Reset	–	–	–	–	–	–	–	–

17h.7~0 **ADCDH:** ADC 输出数据位 11~4

18h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCTL	ADCDL				ADST	ADCKS		
R/W	R				R/W	R/W		
Reset	–	–	–	–	0	0	0	0

18h.7~4 **ADCDL:** ADC 输出数据位 3~0

18h.3 **ADST:** ADC 启动位
0: 转换结束后由 H/W 清除
1: ADC 开始转换

18h.2~0 **ADCKS:** ADC 时钟频率选择:
000: Fsys/256 100: Fsys/16
001: Fsys/128 101: Fsys/8
010: Fsys/64 110: Fsys/4
011: Fsys/32 111: Fsys/2

19h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCTL2	ADVREFS		–	ADCHS				
R/W	R/W		–	R/W				
Reset	0	0	–	1	1	1	1	1

19h.7~6 **ADVREFS:** ADC 参考电压选择和 V_{BG} 输出电压选择
00: ADC 参考电压为 V_{CC}，而 V_{BG} 为 1.20V
01: ADC 参考电压为 V_{BG}，而 V_{BG} 为 2.48V
1x: 保留
11: ADC 参考电压为 V_{BG}，而 V_{BG} 为 2.00V(这个特性无法仿真)(不要用于 DAC 的 VREF 的选择)

19h.3~0 **ADCHS:** ADC 通道选择
00000: ADC0 (PA0) 01000: ADC8 (PB1)
00001: ADC1 (PA1) 01001: ADC9 (PB2)
00010: ADC2 (PA2) 01010: ADC10 (PB4)
00011: ADC3 (PA3) 01011: ADC11 (PB5)
00100: ADC4 (PA4) 01100: ADC12 (PB6)
00101: ADC5 (PA5) 01110: VBG
00110: ADC6 (PA6) 10111: 1/4 VCC
00111: ADC7 (PB0) others: 保留

6.7 比较器

该器件中有一个比较器 (CMP)。

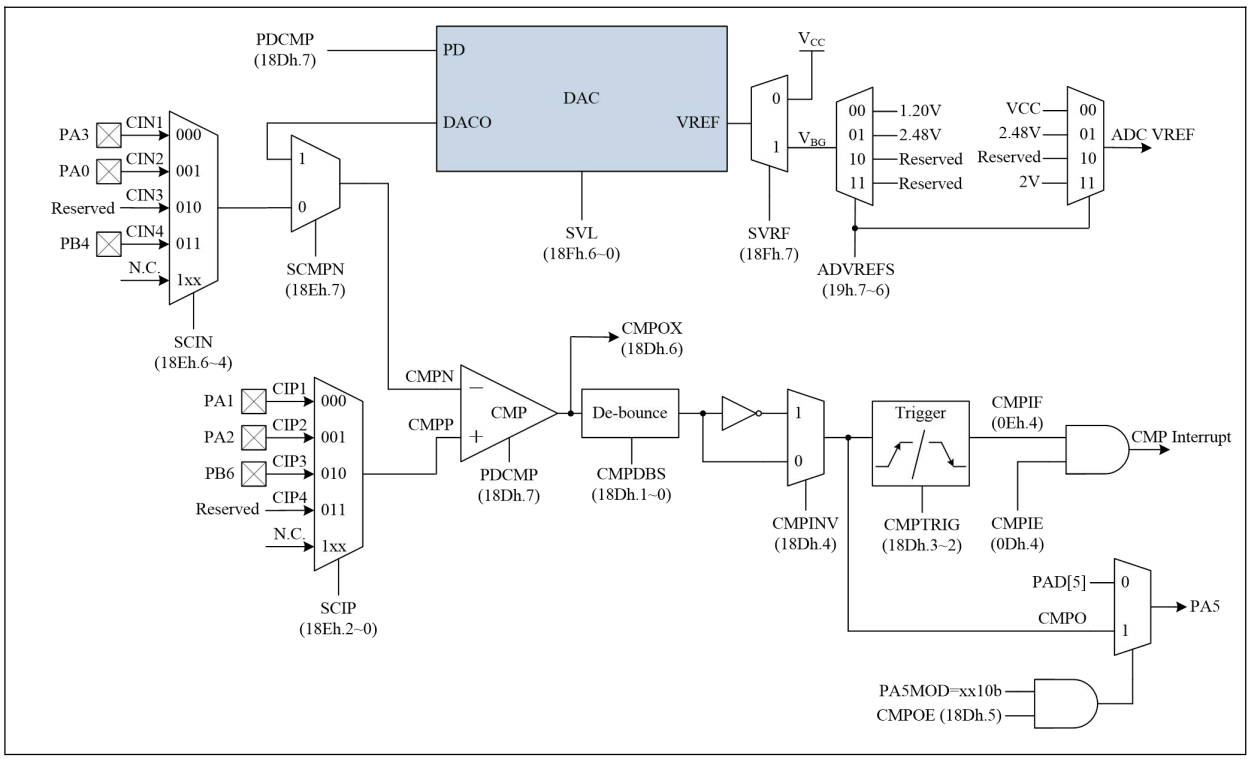
CMP 内置一个 7 位 DAC 模块，其输出可接入 CMP 的负输入端口。

DAC 的参考电压可以通过设置 SVRF (18Fh.7) 选择为 V_{CC} 或 V_{BG}。通过设置 ADVREFS (19h.7~6) 将 V_{BG} 配置为 1.20V 或 2.48V。通过设置 SVL (18Fh.6~0) 可以选择合适的电压电平以供用户应用正确操作，这将改变电阻以转换电压值。设置 PDCMP=1 (18Dh.7) 将使 DAC 和 CMP 进入断电模式。通过配置 SCMPN (18Eh.7)，负端口输入源将是外部引脚输入或 DAC 输出。通过定义 OPOF (18Eh.3)，正端口输入源可以是外部引脚输入或 OPA 输出 (OPO)。SCIN (18Eh.6~4) 和 SCIP (18Eh.2~0) 寄存器分别决定负端口和正端口外部输入源。

由于 CMP 的输入模块是由 PMOS 组成的，所以输入电压范围会受到 PMOS 的 Vth 的影响。因此，CMP 的最大输入电压为 (V_{CC}-0.5) V。同时，比较器的迟滞电压约为 30mV。

比较器原始输出 (CMPOX) 可以通过 CMPOX (18Dh.6) 位读取。芯片提供去抖模块对 CMPOX 信号进行去抖，用户可以通过 CMPDBS (18Dh.1~0) 选择去抖时间。去抖动输出信号可以通过 CMPINV (18Dh.4) 选择是否反相来产生 CMPO 信号。通过设置 CMPOE (18Dh.5) 可以将 CMPO 输出到引脚 (PA5)，并且 PA5MOD 应设置为 xx10b。

CMPO 也是中断触发模块产生中断标志 CMPIF (0Eh.4) 的触发源。触发模式由 CMPTRIG (18Dh.3~2) 选择。当比较器掉电时，仍会产生中断标志。因此，每次打开 CMP 模块后必须先清除中断标志，以防止使用虚拟标志。



0Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIE1	–	–	–	CMPIE	–	–	PWMIE	LVDIE
R/W	–	–	–	R/W	–	–	R/W	R/W

Reset	–	–	–	0	–	–	0	0

0Dh.4 **CMPIE:** 比较器中断启用
0: 关闭
1: 启用

0Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTIF1	–	–	–	CMPIF	–	–	PWMIF	LVDIF
R/W	–	–	–	R/W	–	–	R/W	R/W
Reset	–	–	–	0	–	–	0	0

0Eh.4 **CMPIF:**比较器中断事件挂起标志
当 CMPO 匹配触发条件时由 H/W 设置，对此位写 0 将清除该标志

19h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADCTL2	ADVREFS		–	ADCHS				
R/W	R/W		–	R/W				
Reset	0	0	–	1	1	1	1	1

19h.7~6 **ADVREFS:** ADC 参考电压选择和 V_{BG} 输出电压选择
00: ADC 参考电压为 V_{CC}，而 V_{BG} 为 1.20V
01: ADC 参考电压为 V_{BG}，而 V_{BG} 为 2.48V
10: 保留
11: ADC 参考电压为 VBG, VBG 为 2V(这个特性无法仿真)(不要用于 DAC 的 VREF 的选择)

18Dh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMPCTL	PDCMP	CMPOX	CMPOE	CMPINV	CMPTRIG		CMPDBS	
R/W	R/W	R	R/W	R/W	R/W		R/W	
Reset	1	1	0	0	0	0	0	0

18Dh.7 **PDCMP:** 比较器和 DAC 使能控制
0: 启用比较器和 DAC
1: 关闭比较器和 DAC

18Dh.6 **CMPOX:** 比较器原始输出 (CMPOX) 状态
0: $V_{CMPP} < V_{CMPN}$
1: $V_{CMPP} > V_{CMPN}$ or PDCMP =1

18Dh.5 **CMPOE:** 比较器输出 (CMPO) 信号输出到 PA5
0: 关闭
1: 使能, PA5MOD 应设置为 xx10b

18Dh.4 **CMPINV:** 比较器去抖输出反转选择
0: 无反转
1: 使反转

18Dh.3~2 **CMPTRIG:** 比较器中断触发模式
00: 上升沿
01: 下降沿
10: 双边沿
11: 高电平

18Dh.1~0 **CMPDBS:** 比较器原始输出 (CMPOX) 去抖时间
00: 无
01: 4 Fsys
10: 8 Fsys
11: 16 Fsys

18Eh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMPPNS	SCMPN	SCIN			–	SCIP		
R/W	R/W	R/W			–	R/W		
Reset	1	1	1	1	–	1	1	1

18Eh.7 **SCMPN:**比较器 CMPN 源选择
0: 比较器 CMPN 源为外部引脚 (CINx)
1: 比较器 CMPN 源为 DAC 输出

18Eh.6~4 **SCIN:** 比较器 CMPN 外部输入选择
000: 比较器 CMPN 外部输入为 CIN1 (PA3)
001: 比较器 CMPN 外部输入为 CIN2 (PA0)
010: 保留
011: 比较器 CMPN 外部输入为 CIN4 (PB4)
1xx: 无连接

18Eh.2~0 **SCIP:** 比较器 CMPP 输入选择
000: 比较器 CMPP 外部输入为 CIP1 (PA1)
001: 比较器 CMPP 外部输入为 CIP2 (PA2)
010: 比较器 CMPP 外部输入为 CIP3 (PB6)
011: 保留
1xx: 无连接

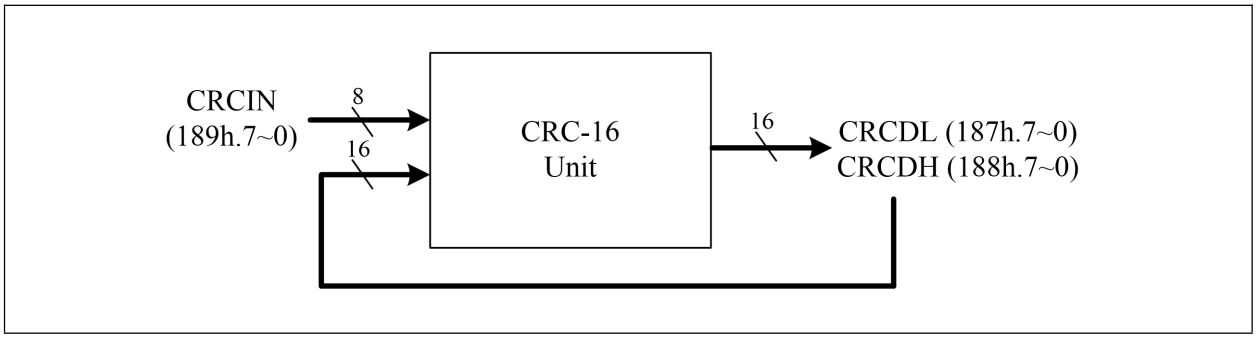
18Fh	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DACTL	SVRF	SVL						
R/W	R/W	R/W						
Reset	0	0	0	0	0	0	0	0

18Fh.7 **SVRF:** DAC 参考电压选择
0: V_{CC}
1: V_{BG} (电压等级由 ADVREFS 选择)

18Fh.6~0 **SVL:** DAC 输出电压选择 (参考源可以选择为 V_{CC} 或 V_{BG})
000_0000: 0/128 * 参考电压
000_0001: 1/128 * 参考电压
…
111_1110: 126/128 * 参考电压
111_1111: 127/128 * 参考电压

6.8 循环冗余检查 (CRC)

该芯片支持集成的 16 位循环冗余校验功能。循环冗余校验 (CRC) 计算单元是一种错误检测技术测试算法，用于验证数据传输或存储数据的正确性。CRC 计算将 8 位数据流或数据块作为输入，并生成 16 位输出余数。数据流由相同的生成多项式计算。



CRC16 Block Diagram

CRC 生成器基于 CRC-16-IBM 多项式提供 16 位 CRC 结果计算。在此 CRC 生成器中，只有一个多项式可用于数值计算。它不支持基于任何其他多项式的 16 位 CRC 计算。对 CRCIN 寄存器的每次写操作都会创建存储在 CRCDH 和 CRCDL 寄存器中的先前 CRC 值的组合。计算将需要一个 MCU 指令周期。

CRC-16-IBM (Modbus) 多项式表示: $X^{16} + X^{15} + X^2 + 1$

187h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CRCDL	CRCDL							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

187h.7~0 **CRCDL:** 16 位 CRC 校验数据位 7~0

188h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CRCDH	CRCDH							
R/W	R/W							
Reset	1	1	1	1	1	1	1	1

188h.7~0 **CRCDH:** 16 位 CRC 校验数据位 15~8

189h	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CRCIN	CRCIN							
W	W							
Reset	–	–	–	–	–	–	–	–

189h.7~0 **CRCIN:** CRC 数据输入，写入此寄存器以开始 CRC 计算

存储器映射

名称	地址	读/写	复位	描述
INDF (00h/80h/100h/180h)				**相关功能: RAM W/R**
INDF	00.7~0	R/W	-	不是实质寄存器，寻址 INDF 实际上指向其地址为 FSR 寄存器中的寄存值
TM0 (01h/101h)				**相关功能: Timer0**
TM0	01.7~0	R/W	00	Timer0 计数数据
PCL (02h/82h/102h/182h)				**相关功能: PROGRAM COUNT**
PCL	02.7~0	R/W	00	程序计数器数据位 7~0
STATUS (03h/83h/103h/183h)				**相关功能: STATUS**
IRP	03.7	R/W	0	寄存器 Bank 选择位 (用于间接寻址)
RP1	03.6	R/W	0	寄存器 Bank 选择位 1 用于直接寻址
RP0	03.5	R/W	0	寄存器 Bank 选择位 0 用于直接寻址
TO	03.4	R	0	WDT 超时标志，通过 PWRST，‘SLEEP’ 或 ‘CLRWDT’ 指令清除
PD	03.3	R	0	掉电标志，通过指令 ‘SLEEP’ 设置，通过指令 ‘CLRWDT’ 清除
Z	03.2	R/W	0	零标志
DC	03.1	R/W	0	十进制进位标志
C	03.0	R/W	0	进位标志
FSR (04h/84h/104h/184h)				**相关功能: RAM W/R**
FSR	04.7~0	R/W	-	文件选择寄存器，间接地址模式指针
PAD (05h)				**相关功能: 端口 A**
PAD	05.7~0	R	-	端口 A 引脚或 “数据寄存器” 状态
		W	FF	端口 A 输出数据寄存器
PBD (06h)				**相关功能: 端口 B**
PBD	06.6~4	R	-	端口 B 引脚或 “数据寄存器” 状态
		W	7	端口 B 输出数据寄存器
	06.2~0	R	-	端口 B 引脚或“数据寄存器”状态
		W	7	端口 B 输出数据寄存器
PCLATH (0Ah/8Ah/10Ah/18Ah)				**相关功能: 程序计数**
GPR	0A.7~4	R/W	0	通用寄存器
PCLATH	0A.3~0	R/W	0	程序计数器高字节的写缓冲区
INTIE (0Bh/8Bh/10Bh/18Bh)				**相关功能: 中断使能**
ADCIE	0B.7	R/W	0	ADC 中断使能 0: 关闭 1: 使能
T2IE	0B.6	R/W	0	T2 中断使能 0: 关闭 1: 使能
TM1IE	0B.5	R/W	0	Timer1 中断使能 0: 关闭 1: 使能
TM0IE	0B.4	R/W	0	Timer0 中断使能 0: 关闭 1: 使能
WKTIE	0B.3	R/W	0	唤醒定时器中断使能和唤醒定时器使能 0: 关闭 1: 使能
INT2IE	0B.2	R/W	0	INT2 引脚 (PA7 或 PB5) 中断使能 0: 关闭 1: 使能
INT1IE	0B.1	R/W	0	INT1 引脚 (PA1 或 PB1) 中断使能 0: 关闭 1: 使能
INT0IE	0B.0	R/W	0	INT0 引脚 (PA3 或 PB2) 中断使能 0: 关闭 1: 使能

名称	地址	读/写	复位	描述
INTIF (0Ch)				**相关功能: 中断标志**
ADCIF	0C.7	R	-	ADC 中断标志，ADC 转换完成后由 H/W 置位
		W	0	写 0: 清除该标志；写 1: 无动作
T2IF	0C.6	R	-	T2 中断事件挂起标志，当 T2 溢出时由 H/W 置位
		W	0	写 0: 清除该标志；写 1: 无动作
TM1IF	0C.5	R	-	Timer1 中断事件挂起标志，当 Timer1 溢出时由 H/W 置位
		W	0	写 0: 清除该标志；写 1: 无动作
TM0IF	0C.4	R	-	Timer0 中断事件挂起标志，当 Timer0 溢出时由 H/W 置位
		W	0	写 0: 清除该标志；写 1: 无动作
WKTIF	0C.3	R	-	WKT 中断事件挂起标志，当 WKT 超时时由 H/W 置位
		W	0	写 0: 清除该标志；写 1: 无动作
INT2IF	0C.2	R	-	INT2 (PA7 或 PB5) 中断事件挂起标志，当 INT2 引脚发生下降沿时由 H/W 置位
		W	0	写 0: 清除该标志；写 1: 无动作
INT1IF	0C.1	R	-	INT1 (PA1 或 PB1) 中断事件挂起标志，当 INT1 引脚发生下降沿/上升沿时由 H/W 置位
		W	0	写 0: 清除该标志；写 1: 无动作
INT0IF	0C.0	R	-	INT0 (PA3 或 PB2) 中断事件挂起标志，当 INT0 引脚发生下降沿/上升沿时由 H/W 置位
		W	0	写 0: 清除该标志；写 1: 无动作
INTIE1 (0Dh)				**相关功能: 中断使能**
CMPIE	0D.4	R/W	0	比较器中断使能 0: 关闭 1: 使能
PWMIE	0D.1	R/W	0	PWM 中断使能 0: 关闭 1: 使能
LVDIE	0D.0	R/W	0	LVD 中断使能 0: 关闭 1: 使能
INTIF1 (0Eh)				**相关功能: 中断标志**
CMPIF	0E.4	R	-	比较器中断标志，当 CMPO 匹配触发条件时由 H/W 置位
		W	0	写 0: 清除此标志；写 1: 无动作
PWMIF	0E.1	R	-	PWM 中断标志，PWM 周期计数器翻转后由 H/W 置位
		W	0	写 0: 清除此标志；写 1: 无动作
LVDIF	0E.0	R	-	LVD 中断标志，当 $V_{CC} < V_{LVD}$ 时由 H/W 置位
		W	0	写 0: 清除此标志；写 1: 无动作
CLKCTL (0Fh)				**相关功能: Fsys**
SCKTYPE	0F.7	R/W	0	慢时钟类型 0: SIRC 1: SXT
FCKTYPE	0F.6	R/W	0	高速时钟类型 0: FIRC 1: FXT
SLOWSTP	0F.4	R/W	0	在 SLEEP 指令后停止慢时钟 0: 慢时钟在 SLEEP 指令后持续运行 1: 慢时钟在 SLEEP 指令后停止运行
FASTSTP	0F.3	R/W	1	停止快时钟 0: 快时钟运行 1: 快时钟停止
CPUCKS	0F.2	R/W	0	系统时钟源选择

名称	地址	读/写	复位	描述
				0: 慢时钟　　1: 快时钟
CPUPSC	0F.1~0	R/W	11	系统时钟源预分频器，系统时钟源 00: 除 8　01: 除 4　10: 除 2　11: 除 1
TM0RLD (10h)				**相关功能: Timer0**
TM0RLD	10.7~0	R/W	00	Timer0 重载数据
TM0CTL (11h)				**相关功能: Timer0**
TM0STP	11.6	R/W	0	停止 Timer0 0: Timer0 运行 1: Timer0 停止
TM0EDG	11.5	R/W	0	TM0CKI (PA2) 边缘 0: 上升沿　1: 下降沿
TM0CKS	11.4	R/W	0	Timer0 预分频器时钟源 0: Fsys/2 1: TM0CKI 引脚 (PA2)
TM0PSC	11.3~0	R/W	0	Timer0 预分频器。Timer0 预分频器时钟源除以 0000: 1　0011: 8　0110: 64 0001: 2　0100: 16　0111: 128 0010: 4　0101: 32　1xxx: 256
TM1 (12h)				**相关功能: Timer1**
TM1	12.7~0	R/W	00	Timer1 计数数据
TM1RLD (13h)				**相关功能: Timer1**
TM1RLD	13.7~0	R/W	00	Timer1 重载数据
TM1CTL (14h)				**相关功能: Timer1**
TM1STP	14.6	R/W	0	停止 Timer1 0: Timer1 运行 1: Timer1 停止
TM1PSC	14.3~0	R/W	0	Timer1 预分频器。Timer1 预分频器时钟源除以 0000: 1　0011: 8　0110: 64 0001: 2　0100: 16　0111: 128 0010: 4　0101: 32　1xxx: 256
T2CTL (15h)				**相关功能: T2**
T2CLR	15.4	R/W	0	清除并停止 T2 0: T2 运行　1: T2 清除并停止
T2CKS	15.3~2	R/W	0	T2 时钟源选择 00: 慢时钟　11: Fsys/128　1x: FIRC/512 (16 MHz/512)
T2PSC	15.1~0	R/W	0	T2 预分频器。T2 时钟源除以 00: 32768　01: 16384　10: 8192　11: 128
LVCTL (16h)				**相关功能: LVD/LVR**
LVDF	16.7	R	0	低电压检测标志 0: $V_{CC} > V_{LVD}$ 1: $V_{CC} < V_{LVD}$
LVDHYS	16.6	R/W	0	LVD 迟滞 0: 关闭　1: 使能
LVRSAV	16.5	R/W	1	POR/LVR 在停止/空闲模式下自动关闭
LVDSAV	16.4	R/W	1	LVD 在停止/空闲模式下自动关闭
LVDS	16.3~0	R/W	0	LVD 电压 (V_{LVD}) 选择 0000: 关闭　0100 : 2.60V　1000: 3.15V　1100: 3.70V 0001: 2.20V　0101: 2.75V　1001: 3.30V　1101: 3.85V 0010: 2.30V　0110: 2.90V　1010: 3.45V　1110: 4.00V

名称	地址	读/写	复位	描述
				0011: 2.45V 0111: 3.00V 1011: 3.60V 1111: 4.15V
ADCDH (17h)				**相关功能: ADC**
ADCDH	17.7~0	R	-	ADC 输出数据位 11~4
ADCTL (18h)				**相关功能: ADC**
ADCDL	18.7~4	R	-	ADC 输出数据位 3~0
ADST	18.3	R/W	0	ADC 起始位 0: 转换结束后由 H/W 清除 1: ADC 开始转换
ADCKS	18.2~0	R/W	0	ADC 时钟频率选择. 1MHz (Typ.) 000: Fsys/256 010: Fsys/64 100: Fsys/16 110: Fsys/4 001: Fsys/128 011: Fsys/32 101: Fsys/8 111: Fsys/2
ADCTL2 (19h)				**相关功能: ADC**
ADVREFS	19.7~6	R/W	00	ADC 参考电压选择和 V_{BG} 输出电压选择 00: ADC 参考电压为 V_{CC}, V_{BG} 为 1.20V 01: ADC 参考电压为 V_{BG}, V_{BG} 为 2.48V 10: 保留 11: ADC 参考电压为 V_{BG}, V_{BG} 为 2.00V(这个特性无法仿真)(不要用于 DAC 的 VREF 的选择)
ADCHS	19.4~0	R/W	1F	ADC 通道选择 00000: ADC0 (PA0) 01000: ADC8 (PB1) 00001: ADC1 (PA1) 01001: ADC9 (PB2) 00010: ADC2 (PA2) 01010: ADC10 (PB4) 00011: ADC3 (PA3) 01011: ADC11 (PB5) 00100: ADC4 (PA4) 01100: ADC12 (PB6) 00101: ADC5 (PA5) 01110: VBG 00110: ADC6 (PA6) 10111: 1/4 VCC 00111: ADC7 (PB0) others: 保留
User Data Memory				
RAM	20~6F	R/W	-	RAM Bank0 区域(80 字节)
RAM	70~7F	R/W	-	RAM 公共区域(16 字节)
OPTION (81h/181h)				**相关功能: STATUS/INT0/INT1/WDT/WKT**
HWAUTO	81.7	R/W	0	进入/退出中断子程序、硬件自动保存/恢复 WREG、FSR、TABR、PCLATH、DPL、DPH 和 STATUS 不含 TO、PD 0: 关闭 1: 使能
INT0EDG	81.6	R/W	0	INT0 引脚边缘中断事件 0: 下降边缘触发 1: 上升边缘触发
INT1EDG	81.5	R/W	0	INT1 引脚边缘中断事件 0: 下降边缘触发 1: 上升边缘触发
WDTPSC	81.3~2	R/W	3	WDT 周期选择: 00: 84ms 01: 168ms 10: 672ms 11: 1344ms @5V
WKTPSC	81.1~0	R/W	3	WKT 周期选择: 00: 10.5ms 01: 21ms 10: 42ms 11: 84ms @5V
PAMOD10 (85h)				**相关功能: 端口 A**
PA1MOD	85.7~4	R/W	1	PA1 I/O 模式控制
PA0MOD	85.3~0	R/W	1	PA0 I/O 模式控制
PAMOD32 (86h)				**相关功能: 端口 A**
PA3MOD	86.7~4	R/W	1	PA3 I/O 模式控制

名称	地址	读/写	复位	描述
PA2MOD	86.3~0	R/W	1	PA2 I/O 模式控制
PAMOD54 (87h)				**相关功能: 端口 A**
PA5MOD	87.7~4	R/W	1	PA5 I/O 模式控制
PA4MOD	87.3~0	R/W	1	PA4 I/O 模式控制
PAMOD76 (88h)				**相关功能: 端口 A**
PA7MOD	88.7~4	R/W	0	PA7 I/O 模式控制
PA6MOD	88.3~0	R/W	1	PA6 I/O 模式控制
PWMCTL (89h)				**相关功能: PWM0**
PWMEN	89.7	R/W	0	PWM0 ~PWM5 使能 0: 关闭 1: 使能
PWM0OM	89.5~4	R/W	0	PWM0 输出模式 00: 模式 0 10: 模式 2 01: 模式 1 11: 模式 3
PWM0DZ	89.3~0	R/W	0	PWM0 非重叠控制 0000: 没有非重叠 0001: 非重叠宽度为 1 个 PWM 时钟周期 0010: 非重叠宽度为 2 个 PWM 时钟周期 … 1111: 非重叠宽度为 15 个 PWM 时钟周期
PBMOD10 (8Ch)				**相关功能: 端口 B**
PB1MOD	8C.7~4	R/W	1	PB1 I/O 模式控制
PB0MOD	8C.3~0	R/W	1	PB0 I/O 模式控制
PBMOD32 (8Dh)				**相关功能: 端口 B**
PB2MOD	8D.3~0	R/W	1	PB2 I/O 模式控制
PBMOD54 (8Eh)				**相关功能: 端口 B**
PB5MOD	8E.7~4	R/W	1	PB5 I/O 模式控制
PB4MOD	8E.3~0	R/W	1	PB4 I/O 模式控制
PBMOD76 (8Fh)				**相关功能: 端口 B**
PB6MOD	8F.3~0	R/W	1	PB6 I/O mode control
OPTION2 (91h)				**相关功能: PWM0/INT2/INT1/INT0**
PWMCKS	91.5~4	R/W	00	PWM 时钟源 0x: Fsys 10: FRC (16MHz) 11: FRC*2 (32MHz)参考PWMCKS=FIRC x 2的最小工作电压图
INT2SEL	91.2	R/W	0	INT2 引脚选择 0: PA7 1: PB5
INT1SEL	91.1	R/W	0	INT1 引脚选择 0: PA1 1: PB1
INT0SEL	91.0	R/W	0	INT0 引脚选择 0: PA3 1: PB2
PWMPRDH (92h)				**相关功能: PWM**
PWMPRDH	92.7~0	R/W	FF	PWM 周期位 15~8
PWMPRDL (93h)				**相关功能: PWM**
PWMPRDL	93.7~0	R/W	FF	PWM 周期位 7~0
PWM0DH (94h)				**相关功能: PWM0**
PWM0DH	94.7~0	R/W	80	PWM0 占空位 15~8
PWM0DL (95h)				**相关功能: PWM0**

名称	地址	读/写	复位	描述
PWM0DL	95.7~0	R/W	00	PWM0 占空位 7~0
PWM1DH (96h)				**相关功能: PWM1**
PWM1DH	96.7~0	R/W	80	PWM1 占空位 15~8
PWM1DL (97h)				**相关功能: PWM1**
PWM1DL	97.7~0	R/W	00	PWM1 占空位 7~0
PWM2DH (98h)				**相关功能: PWM2**
PWM2DH	98.7~0	R/W	80	PWM2 占空位 15~8
PWM2DL (99h)				**相关功能: PWM2**
PWM2DL	99.7~0	R/W	00	PWM2 占空位 t 7~0
PWM3DH (9Ah)				**相关功能: PWM3**
PWM3DH	9A.7~0	R/W	80	PWM3 占空位 15~8
PWM3DL (9Bh)				**相关功能: PWM3**
PWM3DL	9B.7~0	R/W	00	PWM3 占空位 7~0
PWM4DH (9Ch)				**相关功能: PWM4**
PWM4DH	9C.7~0	R/W	80	PWM4 占空位 15~8
PWM4DL (9Dh)				**相关功能: PWM4**
PWM4DL	9D.7~0	R/W	00	PWM4 占空位 7~0
PWM5DH (9Eh)				**相关功能: PWM5**
PWM5DH	9E.7~0	R/W	80	PWM5 占空位 t 15~8
PWM5DL (9Fh)				**相关功能: PWM5**
PWM5DL	9F.7~0	R/W	00	PWM5 占空位 7~0
User Data Memory				
RAM	A0~EF	R/W	-	RAM Bank1 区域(80 字节)
PINMOD (105h)				**相关功能: IO 端口**
Reserved	105.5	R	0	重置后读取为未知
HSINK	105.2	R/W	1	所有 IO 端口高灌电流使能 0: 低灌电流 1: 高灌电流.PA7 没有高灌电流能力
Reserved	105.1	R/W	0	必须保持在 0
Reserved	105.0	R/W	0	必须保持在 0
LVRPD (109h)				**相关功能: LVR/POR**
LVRPD	109.7~0	W	0	写入 37h 强制关闭 LVR 与 POR 写入 38h 强制关闭 LVR，POR 仍然启用 写入 39h 强制关闭 POR，LVR 仍然启用 写入其他启用 LVR 和 POR
PORPDF	109.1	R	0	POR 强制关闭标志 0: POR 是启用状态 1: POR 是强制关闭状态
LVRPDF	109.0	R	0	LVR 强制关闭标志 0: LVR 是启用状态 1: LVR 是强制关闭状态
PCH (10Ch)				**相关功能: PCH**
PCH	10C.7~0	W	00	当执行以 PCL 作为目标的指令时，编程计数器高字节源选择 写 0x1C 来设置 PCH_S = 1，PCH 保持原始值 写其他值来清除 PCH_S = 0，PCH 来自 PCLATH

名称	地址	读/写	复位	描述
PCH	10C.3~0	R	0	程序计数器数据位 11~8
BGTRIM (10Eh)				**相关功能: Bandgap**
BGTRIM	10E.4~0	R/W	CFG	VBG 1.2V trim value
IRCF (10Fh)				**相关功能:内部 RC**
IRCF	10F.6~0	R/W	CFG	FIRC trim value
BG2TRIM (111h)				**相关功能: Bandgap**
BG2TRIM	111.7~0	R	CFG	VBG 2V 配平值。用户可以将该寄存器移动到 BGTRIM 以获得确切的 2V VBG。这个特性无法仿真。
LDOCCTL (112h)				**相关功能: LDOC**
LDOCOUT	112.0	R/W	0	LDOC 输出控制 0: LDOC 不输出到 PA3 1: LDOC 输出到 PA3 (PA3MOD 设置为 0011b)(这个功能无法仿真)
RDCTL (113h)				**相关功能:Program ROM**
RDCTL	113.1~0	R/W	00	程序 ROM 的读信号延迟控制 00:程序 ROM 读取信号延时 20ns 01:程序 ROM 读信号延时 16ns 10:程序 ROM 读取信号延迟 12ns 11:程序 ROM 读取信号延迟 8ns 为了安全起见，请在慢时钟时更改寄存器。 用户必须将该寄存器切换到“8ns”，以提高最低工作电压的性能。 这个特性无法仿真。
IRCFT (114h)				**相关功能:Internal RC**
IRCFT	114.4~0	R/W	00	微调FIRC主频(这个功能无法仿真)
User Data Memory				
RAM	120~16F	R/W	-	RAM Bank1 区域(80 字节)
DPL (185h)				TBL 数据指针位 7~0
DPL	185.7~0	R/W	00	**相关功能: Table Read**
DPH (186h)				TBL 数据指针位 11~8
DPH	186.3~0	R/W	00	**相关功能: CRC16**
CRCDL (187h)				16 位 CRC 校验数据位 7~0
CRCDL	187.7~0	R/W	FF	**相关功能: CRC16**
CRCDH (188h)				16 位 CRC 校验数据位 15~8
CRCDH	188.7~0	R/W	FF	**相关功能: CRC16**
CRCIN (189h)				CRC 数据输入，写入此寄存器以开始 CRC 计算
CRCIN	189.7~0	W	0	TBL 数据指针位 7~0
TABR (18Ch)				**相关功能: Table Read**
TABR	18C.7~0	R/W	0	1. TABR 写 01h = 指令 TABRL 2. TABR 写 02h = 指令 TABRH 3. 在步骤 1 或步骤 2 之后，读取 TABR 以获取主 ROM 读表取值 在步骤 1 之后，读取 TABR 以获取 EEPROM 值 (当 EEPEN=E2h 时) *ASM 的读表：使用 TABRL / TABRH 或 TABR* *C 的读表：使用 TABR*
CMPCTL (18Dh)				**相关功能: 比较器**
PDCMP	18D.7	R/W	1	比较器和 DAC 使能控制 0: 启用比较器和 DAC

名称	地址	读/写	复位	描述
				1: 关闭比较器和 DAC
CMPOX	18D.6	R	1	比较器原始输出 (CMPOX) 状态 0: $V_{CMPP} < V_{CMPN}$ 1: $V_{CMPP} > V_{CMPN}$ or PDCMP =1
CMPOE	18D.5	R/W	0	比较器输出 (CMPO) 信号输出到 PA5 0: 关闭 1: 使能, PA5MOD 应设置为 xx10b
CMPINV	18D.4	R/W	0	比较器去抖输出反转选择 0: 无反转 1: 使反转
CMPTRIG	18D.3~2	R/W	0	比较器中断触发模式 00: 上升沿 01: 下降沿 10: 双边沿 11: 高电平
CMPDBS	18D.1~0	R/W	0	比较器原始输出 (CMPOX) 去抖时间 00: None 01: 4 Fsys 10: 8 Fsys 11: 16 Fsys
CMPPNS (18Eh)				**相关功能: 比较器/DAC/OPA**
SCMPN	18E.7	R/W	1	比较器 CMPN 源选择 0: 比较器 CMPN 源为外部引脚 (CINx) 1: 比较器 CMPN 源为 DAC 输出
SCIN	18E.6~4	R/W	7	比较器 CMPN 外部输入选择 000: 比较器 CMPN 外部输入为 CIN1 (PA3) 001: 比较器 CMPN 外部输入为 CIN2 (PA0) 010: 保留 011: 比较器 CMPN 外部输入为 CIN4 (PB4) 1xx: 无连接
-	18E.3	-	-	此位在仿真时必须设置为 1
SCIP	18E.2~0	R/W	7	比较器 CMPP 外部输入选择 000: 比较器 CMPP 外部输入为 CIP1 (PA1) 001: 比较器 CMPP 外部输入为 CIP2 (PA2) 010: 比较器 CMPP 外部输入为 CIP3 (PB6) 011: 保留 1xx: 无连接
DACTL (18Fh)				**相关功能: DAC/比较器**
SVRF	18F.7	R/W	0	DAC 参考电压选择 0: V_{CC} 1: V_{BG} (电压等级由 ADVREFS 选择)
SVL	18F.6~0	R/W	0	DAC 输出电压选择 (参考源可以选择为 V_{CC} 或 V_{BG}) 000_0000: 0/128 * 参考电压 000_0001: 1/128 * 参考电压 … 111_1110: 126/128 * 参考电压 111_1111: 127/128 * 参考电压

指令集

每条指令是一个 16 位字，分为一个操作码（用于指定指令类型）和一个或多个操作数（用于进一步指定该指令的操作）。下表中将指令分为面向字节，面向位和面向立即数的操作列表。

对于面向字节的指令，“ f”代表地址指示符，“ d”代表目标指示符。地址指示符用于指定指令将使用程序存储器中的哪个地址。目标指示符指定将操作结果放置在何处。如果“ d”为“ 0”，则结果存入 W 寄存器。如果“ d”为“ 1”，则结果放置在指令指定的地址中。

对于面向位的指令，“ b”代表位字段指示符，它选择受操作影响的位数，而“ f”代表地址指示符。对于立即数运算，“ k”代表立即数或常量值。

简记符号	描述
f	文件寄存器地址
b	位地址
k	立即数.常量数字或标签
d	目标选择字段，0：工作寄存器，1：文件寄存器
W	工作寄存器
Z	零标志
C	进位标志或/借位标志
DC	十进制进位标志或十进制/借位标志
PC	程序计数器
TOS	顶层堆栈
GIE	总中断使能标志 (i-Flag)
[]	选择字段
( )	内容
.	位域
B	之前
A	之后
←	分配方向

助记符		操作码	周期	影响标志	描述
面向字节的文件寄存器指令					
ADDWX	f, d	ff00 0111 dfff ffff	1	C, DC, Z	W 和 "f" 相加
ANDWX	f, d	ff00 0101 dfff ffff	1	Z	W 和 "f" 相与
CLRX	f	ff00 0001 1fff ffff	1	Z	清除"f"
CLRW		0000 0001 0100 0000	1	Z	清除 W
COMX	f, d	ff00 1001 dfff ffff	1	Z	"f"取反
DECX	f, d	ff00 0011 dfff ffff	1	Z	“ f”减 1
DECXSZ	f, d	ff00 1011 dfff ffff	1 or 2	-	"f"减 1，如果为零则跳过
INCX	f, d	ff00 1010 dfff ffff	1	Z	"f"加 1
INCXSZ	f, d	ff00 1111 dfff ffff	1 or 2	-	"f"加 1, 如果为零则跳过
IORWX	f, d	ff00 0100 dfff ffff	1	Z	W 和 “f” 相或
MOVX	f,d	ff00 1000 dfff ffff	1	Z	移 "f"
MOVXW	f	ff00 1000 0fff ffff	1	Z	将 "f"移至 W
MOVWX	f	ff00 0000 1fff ffff	1	-	将 W 移至 "f"
RLX	f, d	ff00 1101 dfff ffff	1	C	“f”带进位位左移
RRX	f, d	ff00 1100 dfff ffff	1	C	“f”带进位位右移
SUBWX	f, d	ff00 0010 dfff ffff	1	C, DC, Z	"f"减 W
SWAPX	f, d	ff00 1110 dfff ffff	1	-	"f"的高低半字节互换
TSTX	f	ff00 1000 1fff ffff	1	Z	检测"f" 是否为 0
XORWX	f, d	ff00 0110 dfff ffff	1	Z	W 和 "f"相异或
面向位的文件寄存器指令					面向位的文件寄存器指令
BCX	f, b	ff11 00bb bfff ffff	1	-	"f"的"b"位清零
BSX	f, b	ff11 01bb bfff ffff	1	-	"f"的"b"位置位
BTXSC	f, b	ff11 10bb bfff ffff	1 or 2	-	"f"的"b"位为 0 则跳过
BTXSS	f, b	ff11 11bb bfff ffff	1 or 2	-	"f"的"b"位为 1 则跳过
立即数和控制指令					立即数和控制指令
ADDLW	k	0001 1100 kkkk kkkk	1	C, DC, Z	立即数 "k" 和 W 相加
ANDLW	k	0001 1011 kkkk kkkk	1	Z	立即数 "k" 和 W 相与
LCALL	k	kk10 0kkk kkkk kkkk	2	-	调用子程序 "k"
CLRWDT		0001 1110 0000 0100	1	TO, PD	清除看门狗定时器
LGOTO	k	kk10 1kkk kkkk kkkk	2	-	跳转至分支"k"
IORLW	k	0001 1010 kkkk kkkk	1	Z	立即数"k" 和 W 相或
MOVLW	k	0001 1001 kkkk kkkK	1	-	将立即数 "k" 移至 W
NOP		0000 0000 0000 0000	1	-	空操作指令
RET		0000 0000 0100 0000	2	-	从子程序返回
RETI		0000 0000 0110 0000	2	-	从中断返回
RETLW	k	0001 1000 kkkk kkkk	2	-	带立即数返回，返回值在 W 中
SLEEP		0001 1110 0000 0011	1	TO, PD	进入睡眠模式，时钟振荡停止
SUBLW	k	0001 1111 kkkk kkkk	1	C, DC, Z	立即数"k"减去 W
TABRH		0000 0000 0101 1000	2	-	查找 ROM 高数据到 W
TABRL		0000 0000 0101 0000	2	-	查找 ROM 低数据到 W
XORLW	k	0001 1101 kkkk kkkk	1	Z	立即数"k" 和 W 相异或

ADDLW	W 和 立即数 "k" 相加	
语法	ADDLW k	
操作数	k : 00h ~ FFh	
运作方式	(W) ← (W) + k	
影响的状态位	C, DC, Z	
操作码	0001 1100 kkkk kkkk	
描述	将 W 寄存器的内容和 8 位立即数 'k' 相加，将结果放入W寄存器中。	
周期	1	
范例	ADDLW 0x15	B : W =0x10 A : W =0x25

ADDWX	W 和 "f" 相加	
语法	ADDWX f [,d]	
操作数	f : 000h ~ 1FFh, d : 0, 1	
运作方式	(目标) ← (W) + (f)	
影响的状态位	C, DC, Z	
操作码	ff00 0111 dfff ffff	
描述	将W寄存器的内容和寄存器 'f' 相加。 如果d为0，则结果放在W寄存器中。如果 'd' 为1，结果放回寄存器 'f'。	
周期	1	
范例	ADDWX FSR, 0	B : W =0x17, FSR =0xC2 A : W =0xD9, FSR =0xC2

ANDLW	W 和 立即数 "k" 逻辑与	
语法	ANDLW k	
操作数	k : 00h ~ FFh	
运作方式	(W) ← (W) AND k	
影响的状态位	Z	
操作码	0001 1011 kkkk kkkk	
描述	W 寄存器的内容与 8 位立即数 'k' 相与。 结果放在 W 寄存器中。	
周期	1	
范例	ANDLW 0x5F	B : W =0xA3 A : W =0x03

ANDWX	W 和 "f" 逻辑与	
语法	ANDWX f [,d]	
操作数	f : 000h ~ 1FFh, d : 0, 1	
运作方式	(目标) ← (W) AND (f)	
影响的状态位	Z	
操作码	ff00 0101 dfff ffff	
描述	W 寄存器的内容和寄存器 'f' 相与。 如果 'd' 为 0，则结果放在 W 寄存器中。 如果 'd' 为 1，结果放回寄存器 'f'。	
周期	1	
范例	ANDWX FSR, 1	B : W =0x17, FSR =0xC2 A : W =0x17, FSR =0x02

BCX	"f" 的 "b" 位清零	
语法	BCX f [,b]	
操作数	f : 000h ~ 1FFh, b : 0 ~ 7	
运作方式	(f.b) ← 0	
影响的状态位	-	
操作码	ff11 00bb bfff ffff	
描述	寄存器 'f' 中的 'b' 位被清零。	
周期	1	
范例	BCX FLAG_REG, 7	B : FLAG_REG =0xC7 A : FLAG_REG =0x47

BSX	"f" 的 "b" 位置位	
语法	BSX f [,b]	
操作数	f : 000h ~ 1FFh, b : 0 ~ 7	
运作方式	(f.b) ← 1	
影响的状态位	-	
操作码	ff11 01bb bfff ffff	
描述	寄存器 'f' 中的 'b' 位被置位。	
周期	1	
范例	BSX FLAG_REG, 7	B : FLAG_REG =0x0A A : FLAG_REG =0x8A

BTXSC	检测 "f" 的 "b" 位，为 0 则跳过	
语法	BTXSC f [,b]	
操作数	f : 000h ~ 1FFh, b : 0 ~ 7	
运作方式	如果（f.b）= 0 则跳过下一条指令	
影响的状态位	-	
操作码	ff11 10bb bfff ffff	
描述	如果寄存器 'f' 中的位 'b' 为 1，则执行下一条指令。 如果寄存器 'f' 中的位 'b' 为 0，则下一条指令放弃执行，而是执行一条 NOP，使其成为 2 周期指令。	
周期	1 or 2	
范例	LABEL1: BTXSC FLAG, 1 TRUE: GOTO SUB1 FALSE: ...	B : PC =LABEL1 A : 如果 FLAG.1 =0, PC =FALSE 如果 FLAG.1 =1, PC =TRUE

BTXSS	检测 "f" 的 "b" 位，为 1 则跳过	
语法	BTXSS f [,b]	
操作数	f : 000h ~ 1FFh, b : 0 ~ 7	
运作方式	如果（f.b）= 1 则跳过下一条指令	
影响的状态位	-	
操作码	ff11 11bb bfff ffff	
描述	如果寄存器 'f' 中的位 'b' 为 0，则执行下一条指令。 如果寄存器 'f' 中的位 'b' 为 1，则下一条指令放弃执行，而是执行一条 NOP 指令，使其成为 2 周期指令。	
周期	1 or 2	
范例	LABEL1: BTXSS FLAG, 1 TRUE: GOTO SUB1	B : PC =LABEL1 A : 如果 FLAG.1 =0, PC =TRUE

FALSE: ... 如果 FLAG.1 =1, PC =FALSE

CLRX	**清除 "f"**
语法	CLRX f
操作数	f : 000h ~ 1FFh
运作方式	(f) ← 00h, Z ← 1
影响的状态位	Z
操作码	ff00 0001 1fff ffff
描述	清除寄存器 'f' 的内容，并将 Z 位置 1。
周期	1
范例	CLRX FLAG_REG B : FLAG_REG =0x5A A : FLAG_REG =0x00, Z =1

CLRW	**清除 W**
语法	CLRW
操作数	-
运作方式	(W) ← 00h, Z ← 1
影响的状态位	Z
操作码	0000 0001 0100 0000
描述	清除 W 寄存器，并将 Z 位置 1
周期	1
范例	CLRW B : W =0x5A A : W =0x00, Z =1

CLRWDT	**清除看门狗定时器**
语法	CLRWDT
操作数	-
运作方式	WDT Timer ← 00h
影响的状态位	TO, PD
操作码	0001 1110 0000 0100
描述	CLRWDT 指令清除看门狗定时器
周期	1
范例	CLRWDT B : WDT 计数器 =? A : WDT 计数器 =0x00

COMX	**"f" 取反**
语法	COMX f [,d]
操作数	f : 000h ~ 1FFh, d : 0, 1
运作方式	(目标) ← ($\bar{f}$)
影响的状态位	Z
操作码	ff00 1001 dfff ffff
描述	寄存器 'f' 的内容被取反。 如果 'd' 为 0，结果放在 W 中。如果 'd' 为 1，结果放回寄存器 'f' 中。
周期	1
范例	COMX REG1, 0 B : REG1 =0x13 A : REG1 =0x13, W =0xEC

DECX	"f" 递减	
语法	DECX f [,d]	
操作数	f : 000h ~ 1FFh, d : 0, 1	
运作方式	(目标) ← (f) - 1	
影响的状态位	Z	
操作码	ff00 0011 dfff ffff	
描述	寄存器 'f' 的内容递减。 如果 'd' 为 0，则结果放在 W 寄存器中。 如果 'd' 为 1，结果放回寄存器 'f'。	
周期	1	
范例	DECX CNT, 1	B : CNT =0x01, Z =0 A : CNT =0x00, Z =1

DECXSZ	"f" 递减，如果为 0 则跳过	
语法	DECXSZ f [,d]	
操作数	f : 000h ~ 1FFh, d : 0, 1	
运作方式	(目标) ← (f) - 1, 如果结果为 0，则跳过下一条指令	
影响的状态位	-	
操作码	ff00 1011 dfff ffff	
描述	寄存器 'f' 的内容递减。 如果 'd' 为 0，结果放入 W 寄存器。 如果 'd' 为 1，结果放回寄存器 'f'。 如果结果为 1，则执行下一条指令。 如果结果为 0，则改为执行 NOP，使其成为 2 周期指令。	
周期	1 或 2	
范例	LABEL1: DECXSZ CNT, 1 LGOTO LOOP CONTINUE:	B : PC =LABEL1 A : CNT =CNT – 1 如果 CNT =0, “LGOTO LOOP”会被置换为 NOP 如果 CNT ≠0, “LGOTO LOOP”会被执行

INCX	"f" 递增	
语法	INCX f [,d]	
操作数	f : 000h ~ 1FFh	
运作方式	(目标) ← (f) + 1	
影响的状态位	Z	
操作码	ff00 1010 dfff ffff	
描述	寄存器 'f' 的内容递增。 如果 'd' 为 0，结果放入 W 寄存器。 如果 'd' 为 1，结果放回寄存器 'f'。	
周期	1	
范例	INCX CNT, 1	B : CNT =0xFF, Z =0 A : CNT =0x00, Z =1

INCXSZ	“f” 递增，如果为 0 则跳过
语法	INCXSZ f [,d]
操作数	f : 000h ~ 1FFh, d : 0, 1
运作方式	(目标) ← (f) + 1, 如果结果为 0，则跳过下一条指令
影响的状态位	-
操作码	ff00 1111 dfff ffff
描述	寄存器 'f' 的内容递增。如果 'd' 为 0，结果放入 W 寄存器。如果 'd' 为 1，结果放回寄存器 'f'。如果结果为 1，则执行下一条指令。 如果结果为 0，则改为执行 NOP，使其成为 2 周期指令。
周期	1 or 2
范例	LABEL1: INCXSZ CNT, 1 LGOTO LOOP CONTINUE: B : PC =LABEL1 A : CNT =CNT + 1 如果 CNT =0, “LGOTO LOOP”会被置换为 NOP 如果 CNT ≠0, “LGOTO LOOP”会被执行

IORLW	W 和立即数 “ k” 逻辑或
语法	IORLW k
操作数	k : 00h ~ FFh
运作方式	(W) ← (W) OR k
影响的状态位	Z
操作码	0001 1010 kkkk kkkk
描述	W 寄存器的内容与 8 位立即数 'k' 进行或运算。 结果放在 W 寄存器中。
周期	1
范例	IORLW 0x35 B : W =0x9A A : W =0xBF, Z =0

IORWX	W 和立即数 “ f” 逻辑或
语法	IORWF f [,d]
操作数	f : 000h ~ 1FFh, d : 0, 1
运作方式	(目标) ← (W) OR k
影响的状态位	Z
操作码	ff00 0100 dfff ffff
描述	W 寄存器与寄存器 'f' 进行或运算。 如果 'd' 为 0，结果放入 W 寄存器。 如果 'd' 为 1，结果放回寄存器 'f'。
周期	1
范例	IORWX RESULT, 0 B : RESULT =0x13, W =0x91 A : RESULT =0x13, W =0x93, Z =0

LCALL	调用子程序 “k”
语法	LCALL k
操作数	k : 0000h ~ 1FFFh
运作方式	Operation: TOS ← (PC) + 1, PC.12~0 ← k
影响的状态位	-
操作码	kk10 0kkk kkkk kkkk
描述	LCALL 子例程。首先，返回地址 (PC+1) 被推送到堆栈上。13 位直接地址加载到 PC 位 <12:0>。LCALL 是一个两周期指令
周期	2
范例	LABEL1: LCALL SUB1 B : PC =LABEL1 A : PC =SUB1, TOS =LABEL1 + 1

LGOTO	无条件转移
语法	LGOTO k
操作数	k : 0000h ~ 1FFFh
运作方式	PC.12~0 ← k
影响的状态位	-
操作码	kk10 1kkk kkkk kkkk
描述	LGOTO 是一个无条件的分支。13 位立即值加载到 PC 位 <12:0>。LGOTO 是一个两周期指令。
周期	2
范例	LABEL1: LGOTO SUB1 B : PC =LABEL1 A : PC =SUB1

MOVX	移 "f"
语法	MOVX f [,d]
操作数	f : 000h ~ 1FFh
运作方式	(目标) ← (f)
影响的状态位	Z
操作码	ff00 1000 dfff ffff
描述	寄存器 'f' 的内容根据 d 的状态移至目标。 如果 d=0，则目标为 W 寄存器。如果 d=1，则目标是文件寄存器 f 本身。 d=1 对测试文件寄存器很有用，因为状态标志 Z 受到影响。
周期	1
范例	MOVX FSR,0 B : FSR =0xC2, W =? A : FSR =0xC2, W =0xC2

MOVXW	将 "f" 移至 W
语法	MOVXW f
操作数	f : 000h ~ 1FFh
运作方式	(W) ← (f)
影响的状态位	Z
操作码	ff00 1000 0fff ffff
描述	寄存器 'f' 的内容移至 W 寄存器。
周期	1
范例	MOVXW FSR B : FSR =0xC2, W =? A : FSR =0xC2, W =0xC2

MOVLW	将立即数移至 W	
语法	MOVLW k	
操作数	k : 00h ~ FFh	
运作方式	(W) ← k	
影响的状态位	-	
操作码	0001 1001 kkkk kkkk	
描述	八位立即数 'k' 被加载到 W 寄存器中。无关位将为 0。	
周期	1	
范例	MOVLW 0x5A	B : W =? A : W =0x5A

MOVWX	将 W 移至 "f"	
语法	MOVWX f	
操作数	f : 000h ~ 1FFh	
运作方式	(f) ← (W)	
影响的状态位	-	
操作码	ff00 0000 1fff ffff	
描述	将数据从 W 寄存器移至寄存器 'f'。	
周期	1	
范例	MOVWX REG1	B : REG1 =0xFF, W =0x4F A : REG1 =0x4F, W =0x4F

NOP	空操作	
语法	NOP	
操作数	-	
运作方式	空操作	
影响的状态位	-	
操作码	0000 0000 0000 0000	
描述	空操作	
周期	1	
范例	NOP	-

RET	从子程序返回	
语法	RET	
操作数	-	
运作方式	PC ← TOS	
影响的状态位	-	
操作码	0000 0000 0100 0000	
描述	从子程序返回。堆栈被弹出，并且堆栈的顶部（TOS）被装入程序计数器。这是两个周期的指令。	
周期	2	
范例	RET	A : PC =TOS

RETI	**从中断返回**
语法	RETI
操作数	-
运作方式	PC ← TOS, GIE ← 1
影响的状态位	-
操作码	0000 0000 0110 0000
描述	从中断返回。弹出堆栈，并将堆栈顶层（TOS）加载到 PC 中。中断使能。这是两个周期的指令。
周期	2
范例	RETI　　A : PC =TOS, GIE =1

RETLW	**W 带立即数返回**
语法	RETLW k
操作数	k : 00h ~ FFh
运作方式	PC ← TOS, (W) ← k
影响的状态位	-
操作码	0001 1000 kkkk kkkk
描述	W 寄存器加载了八位立即数 'k'。程序计数器从堆栈的顶层（返回地址）加载。这是两个周期的指令。
周期	2
范例	LCALL TABLE　　B : W =0x07 :　　A : W =k8 的值 TABLE: ADDWX PCL, 1 RETLW k1 RETLW k2 : RETLW kn

RLX	**"f" 通过进位向左移**
语法	RLX f [,d]
操作数	f : 000h ~ 1FFh, d : 0, 1
运作方式	
影响的状态位	C
操作码	ff00 1101 dfff ffff
描述	寄存器 'f' 的内容通过进位标志向左移动一位。如果 'd' 为 0，结果放入 W 寄存器。如果 'd' 为 1，结果放回寄存器 'f'。
周期	1
范例	RLX REG1, 0　　B : REG1 =1110 0110, C =0 A : REG1 =1110 0110 W =1100 1100, C =1

RRX	"f" 通过进位向右移
语法	RRX f [,d]
操作数	f : 000h ~ 1FFh, d : 0, 1
运作方式	
影响的状态位	C
操作码	ff00 1100 dfff ffff
描述	寄存器 'f' 的内容通过进位标志向右移动一位。如果 'd' 为 0，结果放入 W 寄存器。如果 'd' 为 1，结果放回寄存器 'f'。
周期	1
范例	RRX REG1, 0 B : REG1 =1110 0110, C =0 A : REG1 =1110 0110 W =0111 0011, C =0

SLEEP	进入睡眠模式，时钟振荡停止
语法	SLEEP
操作数	-
运作方式	-
影响的状态位	TO, PD
操作码	001 1110 0000 0011
描述	进入睡眠模式，时钟振荡停止
周期	1
范例	SLEEP -

SUBLW	立即数减去W
语法	SUBLW k
操作数	k : 00h ~ FFh
运作方式	(W) ←k - (W)
影响的状态位	C, DC, Z
操作码	0001 1111 kkkk kkkk
描述	立即数 'k'减去W 寄存器（2的补码方法）。结果放在 W 寄存器中。
周期	1
范例	SUBLW 0x15 B : W =0x25 A : W =0xF0

SUBWX	"f" 减去 W
语法	SUBWX f [,d]
操作数	f : 000h ~ 1FFh, d : 0, 1
运作方式	(目标) ← (f) – (W)
影响的状态位	C, DC, Z
操作码	ff00 0010 dfff ffff
描述	从寄存器 'f' 中减去 W 寄存器（2 的补码方法）。如果 'd' 为 0，则结果放在 W 寄存器中。如果 'd' 为 1，结果放回寄存器 'f'。
周期	1
范例	SUBWX REG1, 1 B : REG1 =0x03, W =0x02, C =?, Z =? A : REG1 =0x01, W =0x02, C =1, Z =0 SUBWX REG1, 1 B : REG1 =0x02, W =0x02, C =?, Z =? A : REG1 =0x00, W =0x02, C =1, Z =1 SUBWX REG1, 1 B : REG1 =0x01, W =0x02, C =?, Z =? A : REG1 =0xFF, W =0x02, C =0, Z =0

SWAPX	"f" 互换半字节
语法	SWAPX f [,d]
操作数	f : 000h ~ 1FFh, d : 0, 1
运作方式	(目标,7~4) ← (f.3~0), (目标.3~0) ← (f.7~4)
影响的状态位	-
操作码	ff00 1110 dfff ffff
描述	寄存器 'f' 的高低半字节互换。如果 'd' 为 0，结果放入 W 寄存器。如果 'd' 为 1，结果放入寄存器 'f'。
周期	1
范例	SWAPX REG1, 0 B : REG1 =0xA5 A : REG1 =0xA5, W =0x5A

TABRH	将 DPTR 高字节返回给 W
语法	TABRH
操作数	-
运作方式	(W) ← ROM[DPTR] 高字节内容, 其中 DPTR = {DPH [max:8], DPL[7:0]}
影响的状态位	-
操作码	0000 0000 0101 1000
描述	W 寄存器加载 ROM[DPTR] 的高字节。这是两个周期的指令。
周期	2
范例	MOVLW (TAB1&0xFF) MOVWX DPL ;DPL 为寄存器 MOVLW (TBA1>>8)&0xFF MOVWX DPH ;DPH 为寄存器 TABRL ;W =0x89 TABRH ;W =0x37 ORG 0234H TAB1: DT 0x3789, 0x2277 ;16 位 ROM 数据

TABRL	将 DPTR 低字节返回给 W
语法	TABRL
操作数	-
运作方式	(W) ← ROM[DPTR] 低字节内容, 其中 DPTR = {DPH[max:8], DPL[7:0]}
影响的状态位	-
操作码	0000 0000 0101 0000
描述	W 寄存器加载 ROM[DPTR] 的低字节。这是两个周期的指令。
周期	2
范例	MOVLW (TAB1&0xFF) MOVWX DPL ;DPL 为寄存器 MOVLW (TBA1>>8)&0xFF MOVWX DPH ;DPH 为寄存器 TABRL ;W =0x89 TABRH ;W =0x37 ORG 0234H TAB1: DT 0x3789, 0x2277 ;16 位 ROM 数据

TSTX	检测 "f" 是否为 0
语法	TSTX f
操作数	f : 000h ~ 1FFh
运作方式	如果（f）为 0，则设置 Z 标志
影响的状态位	Z
操作码	ff00 1000 1fff ffff
描述	如果寄存器 'f' 的内容为 0，则零标志设置为 1。
周期	1
范例	TSTX REG1 B : REG1 =0, Z =? A : REG1 =0, Z =1

XORLW	W 和立即数异或
语法	XORLW k
操作数	k : 00h ~ FFh
运作方式	(W) ← (W) XOR k
影响的状态位	Z
操作码	0001 1101 kkkk kkkk
描述	W 寄存器的内容与 8 位立即数 'k' 进行异或。结果放在 W 寄存器中。
周期	1
范例	XORLW 0xAF B : W =0xB5 A : W =0x1A

XORWX	W 和 "f"异或
语法	XORWX f [,d]
操作数	f : 000h ~ 1FFh, d : 0, 1
运作方式	(目标) ← (W) XOR (f)
影响的状态位	Z
操作码	ff00 0110 dfff ffff
描述	W 寄存器与寄存器 'f' 的内容异或。如果 'd' 为 0，则结果放在 W 寄存器中。如果 'd' 为 1，结果放回寄存器 'f'。
周期	1
范例	XORWX REG1, 1　　B : REG1 =0xAF, W =0xB5 A : REG1 =0x1A, W =0xB5

电气特性

1 最大绝对额定值 (T_A = 25°C)

参数	范围	单位
电源电压	V_{SS} -0.3 to V_{SS} +5.5	V
输入电压	V_{SS} -0.3 to V_{CC} +0.3	
输出电压	V_{SS} -0.3 to V_{CC} +0.3	
每个引脚的高电位输出电流	-25	mA
所有引脚的高电位输出电流	-80	
每个引脚的低电位输出电流	+30	
所有引脚的低电位输出电流	+150	
最大工作电压	5.5	V
工作温度	-40 to +105	°C
储存温度	-65 to +150	

2 直流特性 (TA = 25°C, VCC = 5.0V，除非另有规定)

参数	符号	条件		最小	典型	最大	单位
工作电压	V_{cc}	Fsys = 20 MHz (FXT) (RDCTL=8ns)(PWMCKS=FIRC*1)		3.1	–	5.5	V
		Fsys = 16 MHz (FIRC) (RDCTL=8ns)(PWMCKS=FIRC*1)		2.3	–	5.5	V
		Fsys = 8 MHz (FIRC/2)		1.4	–	5.5	V
输入高电压	V_{IH}	所有输入	V_{CC} = 3.0~5.0V	0.6V_{CC}	–	V_{CC}	V
输入低电压	V_{IL}	所有输入	V_{CC} = 3.0~5.0V	V_{SS}	–	0.2V_{CC}	V
I/O 端口 拉电流	I_{OH}	所有引脚	V_{CC} = 5.0V, V_{OH} = 4.5V	6	12.7	–	mA
			V_{CC} = 3.0V, V_{OH} = 2.7V	2.5	5.3	–	
I/O 端口 灌电流	I_{OL}	所有引脚 (HSINK=1)	V_{CC} = 5.0V, V_{OL} = 0.5V	40	89	–	mA
			V_{CC} = 3.0V, V_{OL} = 0.3V	18	40	–	
		所有引脚 (HSINK=0)	V_{CC} = 5.0V, V_{OL} = 0.5V	25	48	–	mA
			V_{CC} = 3.0V, V_{OL} = 0.3V	10	21	–	
输入漏电流 (引脚为高)	I_{ILH}	所有输入	V_{IN} = V_{CC}	–	–	1	µA
输入漏电流 (引脚为低)	I_{ILL}	所有输入	V_{IN} = 0V	–	–	–1	µA

参数	符号	条件		最小	典型	最大	单位
工作电流 (空载)	I_{CC}	快速模式 FXT 20 MHz	V_{CC} = 5.0V	–	5.1	–	mA
			V_{CC} = 3.0V	–	2.7	–	
		快速模式 FIRC 16 MHz	V_{CC} = 5.0V	–	4.1	–	
			V_{CC} = 3.0V	–	2.5	–	
		快速模式 FIRC 8 MHz	V_{CC} = 5.0V	–	2.9	–	
			V_{CC} = 3.0V	–	1.7	–	
		快速模式 FIRC 4 MHz	V_{CC} = 5.0V	–	2.3	–	
			V_{CC} = 3.0V	–	1.4	–	
		快速模式 FIRC 2 MHz	V_{CC} = 5.0V	–	2.0	–	
			V_{CC} = 3.0V	–	1.2	–	
		慢速模式 SXT 32 KHz FIRC 停止	V_{CC} = 5.0V	–	0.67	–	
			V_{CC} = 3.0V	–	0.5	–	
		慢速模式 SIRC 除 1 FIRC 停止	V_{CC} = 5.0V	–	0.65	–	
			V_{CC} = 3.0V	–	0.47	–	
		慢速模式 SIRC 除 1 FIRC 停止 POR/LVR 关闭	V_{CC} = 5.0V	–	0.56	–	
			V_{CC} = 3.0V	–	0.41	–	
		空闲模式 SIRC 除 1 POR/LVR 关闭	V_{CC} = 5.0V	–	8.4	–	µA
			V_{CC} = 3.0V	–	2.7	–	
		停止模式 POR/LVR 关闭	V_{CC} = 5.0V	–	–	1	µA
			V_{CC} = 3.0V	–	–	1	
上拉电阻	R_{UP}	V_{IN} = 0 V Ports A, B	V_{CC} = 5.0V	–	34.5	–	KΩ
			V_{CC} = 3.0V	–	35	–	
1.2V LDO 稳压	LDOC	V_{CC} = 2.5 ~ 5.0V No Load		1.182	1.2	1.224	V
		V_{CC} = 2.5 ~ 5.0V Ioh=60mA, T_A = -20°C ~ 85°C		1.133	1.15	1.173	V

3 时钟时序

参数	条件		最小	典型	最大	单位
FIRC 频率(*)	T_A = -40°C ~ 105°C	V_{CC} = 3.0 ~ 5.0V	-5%	16	+2%	MHz
	T_A = -40°C ~ 105°C	V_{CC} = 4.0 V	-3%	16	+1.5%	
	T_A = 0°C ~ 70°C	V_{CC} = 4.0 V	-2%	16	+1.5%	
	T_A = 25°C	V_{CC} = 3.0 ~ 5.0 V	-1%	16	+1%	
	T_A = 25°C	V_{CC} = 4.0 V	-0.5%	16	+0.5%	

(*) FIRC 频率可除以 1/2/4/8.

4 复位时间特性 (TA = 25°C)

参数	条件	最小	典型	最大	单位
复位输入低脉宽	输入 V_{CC} = 5.0 V ±10 %	–	11	–	ms
WDT 时间	V_{CC} = 5.0 V, WDTPSC = 11b	–	1344	–	ms
WKT 时间	V_{CC} = 5.0 V, WKTPSC = 11b	–	84	–	ms
CPU 启动时间	V_{CC} = 5.0 V	–	21	–	ms

5 LVR 电路特性 (TA = 25°C)

参数	符号	条件	最小	典型	最大	单位
LVR 参考电压	LVR_{th}	T_A = 25°C	–	2.05	–	V
			–	2.20	–	
			–	2.30	–	
			–	2.45	–	
			–	2.60	–	
			–	2.75	–	
			–	2.90	–	
			–	3.00	–	
			–	3.15	–	
			–	3.30	–	
			–	3.45	–	
			–	3.60	–	
			–	3.70	–	
			–	3.85	–	
			–	4.00	–	
			–	4.15	–	
LVR 迟滞电压	V_{HYS_LVR}	T_A = 25°C	–	0	–	mV
低电压检测时间	T_{LVR}	T_A = 25°C	100	–	–	ms

6 LVD 电路特性 (TA = 25°C)

参数	符号	条件	最小	典型	最大	单位
LVD 参考电压	LVD_{th}	T_A = 25°C	–	2.20	–	V
			–	2.30	–	
			–	2.45	–	
			–	2.60	–	
			–	2.75	–	
			–	2.90	–	
			–	3.00	–	
			–	3.15	–	
			–	3.30	–	
			–	3.45	–	
			–	3.60	–	
			–	3.70	–	
			–	3.85	–	
			–	4.00	–	
			–	4.15	–	
LVD 迟滞电压	V_{HYS_LVD}	LVDHYS = 0	–	0	–	mV
		LVDHYS = 1	–	36	–	
低电压检测时间	T_{LVD}	T_A = 25°C	100	–	–	ms

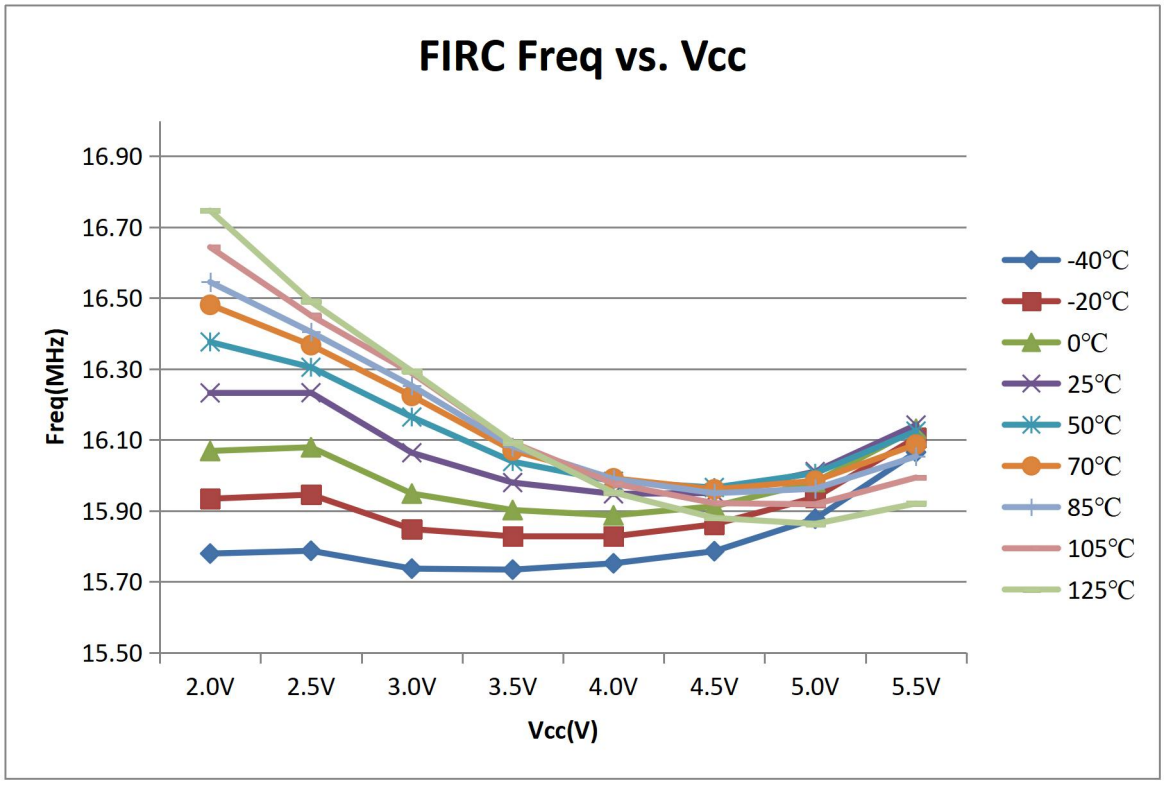
7 ADC 电气特性 (TA = 25°C, VCC = 3.0V to 5.5V, VSS = 0V)

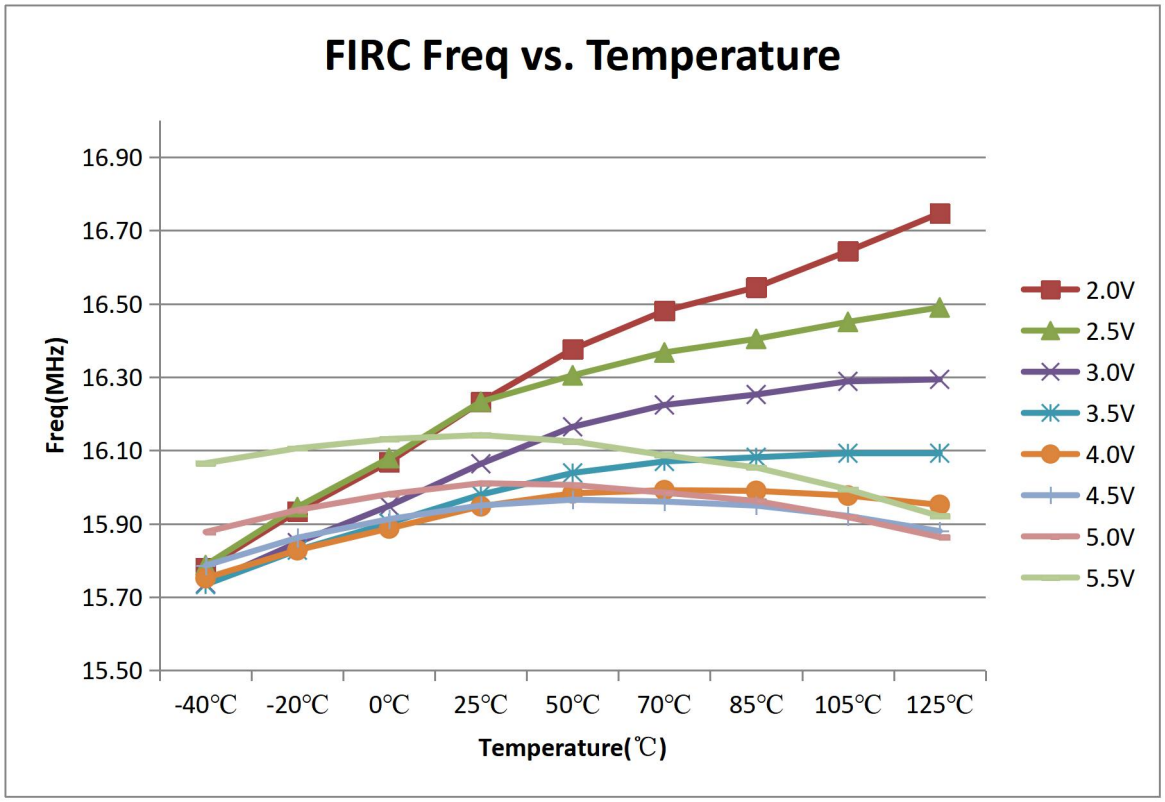
参数	条件	最小	典型	最大	单位
总精度	V_{CC} = 5.0V, V_{SS} = 0V, F_{ADC} = 1 MHz	–	±3	±13	LSB
积分非线性误差		–	±3.2	±15	
微分非线性误差		–	±1	±4	
最大输入时钟频率 (F_{ADC})	信号驱动源阻抗 (Rs<10K ohm)	–	–	2	MHz
	信号驱动源阻抗 (Rs<20K ohm)	–	–	1	
	信号驱动源阻抗 (Rs<50K ohm)	–	–	0.5	
	信号来源是 VBG (ADCHS=01110b)	–	–	2	
转换时间	F_{ADC} = 1 MHz (Include sample and hold time)	–	42	–	µs
带隙电压参考 (V_{BG})	25°C, V_{CC} = 3.0V~5.0V	-1%	1.20	+1%	V
	25°C~105°C, V_{CC} = 3.0V~5.0V	-1%	1.20	+1.5%	V
	-20°C~105°C, V_{CC} = 3.0V~5.0V	-2%	1.20	+1.5%	V
ADC 参考电压 (V_{REF}) (ADVREFS=01b)	25°C, V_{CC} = 3.0V~5.5V	-1.2%	2.48	+1.2%	V
	-20°C~105°C, V_{CC} = 3.0V~5.5V	-2.5%	2.48	+2%	V
V_{CC}/4 参考电压	25°C, V_{CC} = 3.0V~5.5V	-1%	0.25V_{CC}	+1%	V
输入电压	–	V_{SS}	–	V_{CC}	V

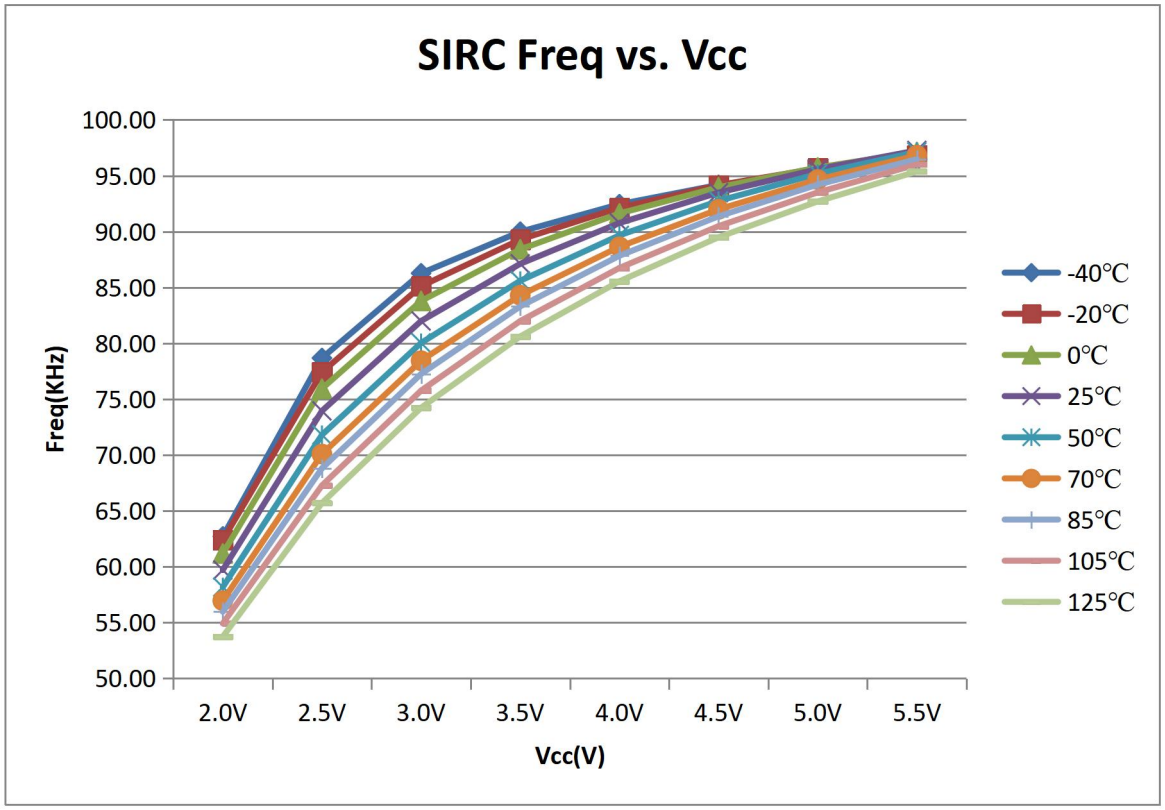
8 比较器特性 (TA = 25°C, VCC = 3.0V to 5.5V, VSS = 0V)

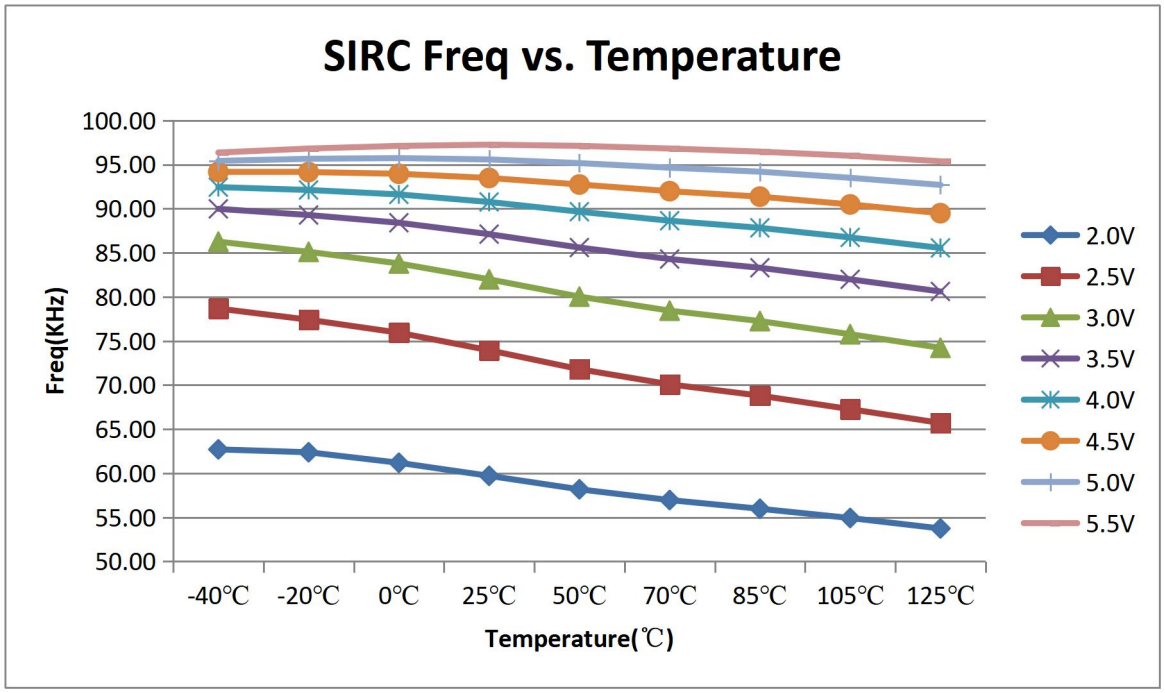
参数	条件	最小	典型	最大	单位
电源电压	–	2.2	–	5.5	V
静态电流	V_{CC} = 5.0V	–	100	–	µA
DAC Current	V_{CC} = 5.0V	60	–	220	µA
V_{OS_CMP}	V_{CC} = 5.0V	-15	–	15	mV
V_{CM_CMP}	V_{CC} = 5.0V	0	–	V_{CC}-0.5	V
V_{HYS_CMP}	V_{CC} = 5.0V	20	30	40	mV

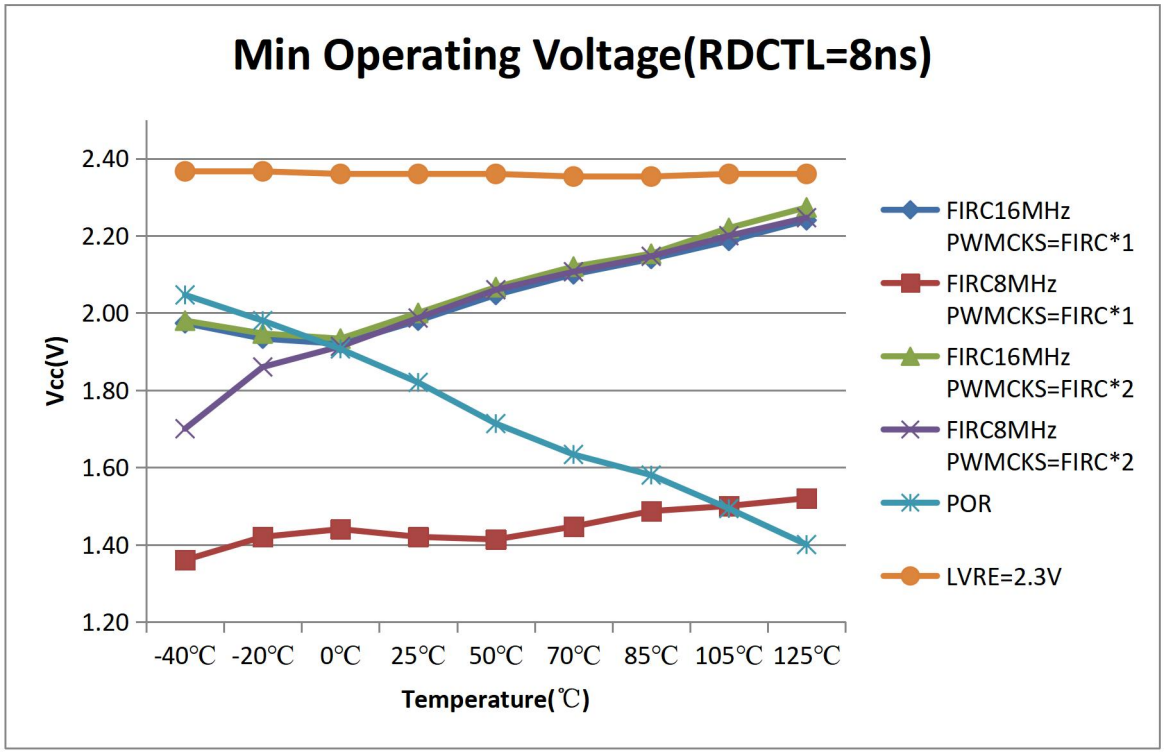
9 特性曲线图

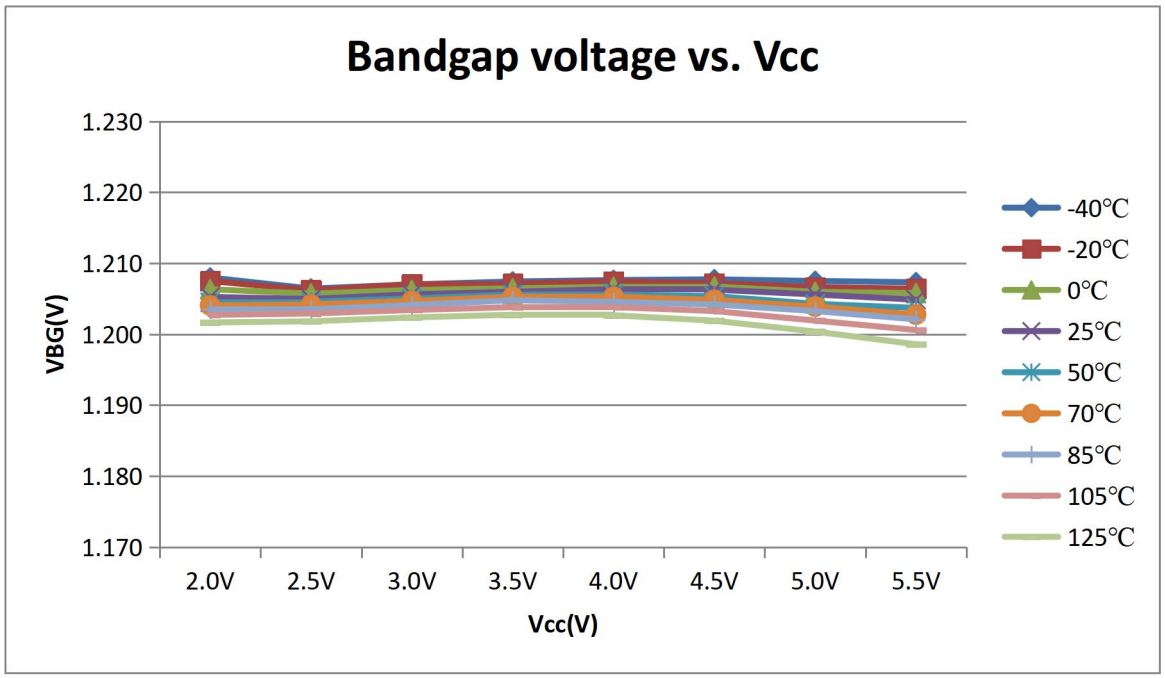

FIRC Freq vs. Vcc
16.90
16.70
16.50
16.30
16.10
15.90
15.70
15.50
Freq(MHz)
2.0V
2.5V
3.0V
3.5V
4.0V
4.5V
5.0V
5.5V
Vcc(V)
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃
125℃

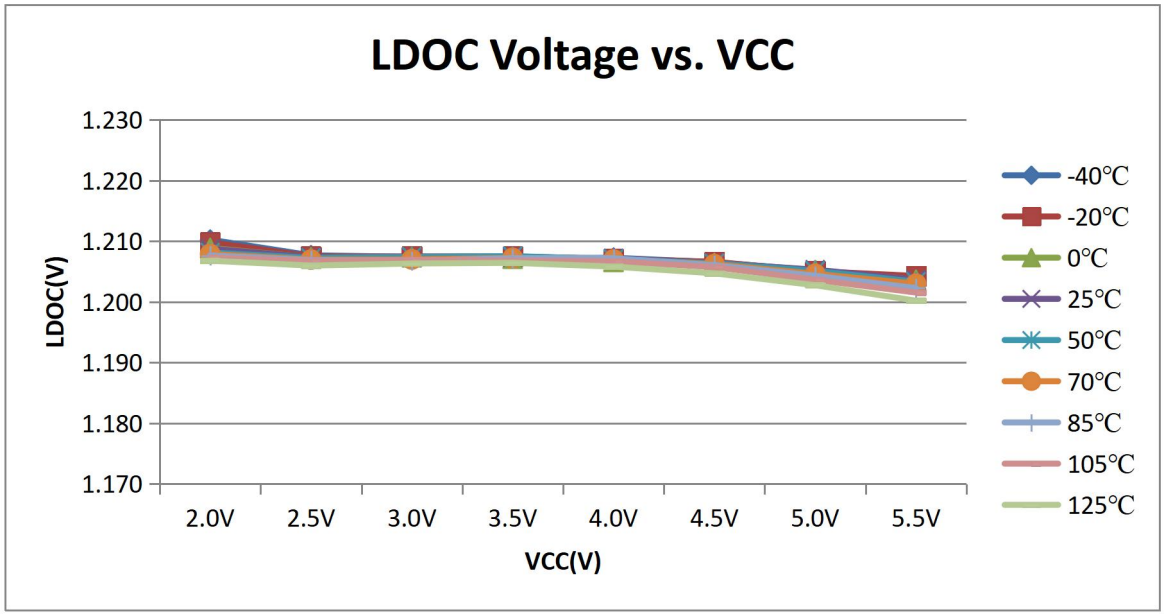

FIRC Freq vs. Temperature
16.90
16.70
16.50
16.30
16.10
15.90
15.70
15.50
Freq(MHz)
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃
125℃
Temperature(℃)
2.0V
2.5V
3.0V
3.5V
4.0V
4.5V
5.0V
5.5V

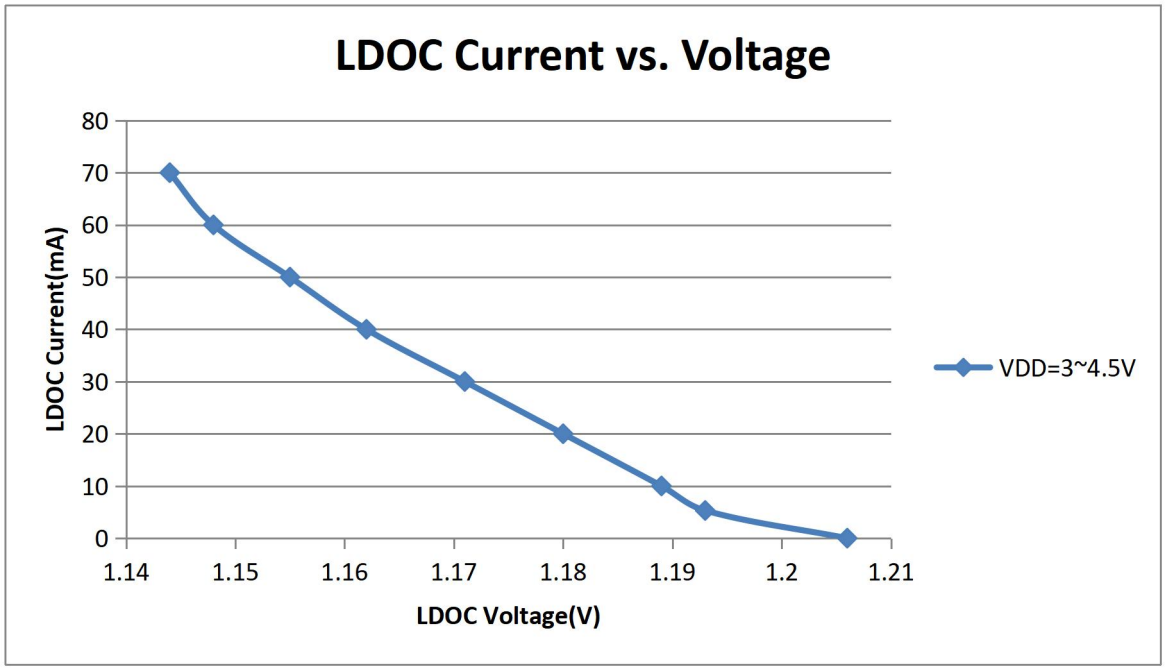

SIRC Freq vs. Vcc
100.00
95.00
90.00
85.00
80.00
75.00
70.00
65.00
60.00
55.00
50.00
Freq(KHz)
2.0V
2.5V
3.0V
3.5V
4.0V
4.5V
5.0V
5.5V
Vcc(V)
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃
125℃


SIRC Freq vs. Temperature
100.00
95.00
90.00
85.00
80.00
75.00
70.00
65.00
60.00
55.00
50.00
Freq(KHz)
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃
125℃
Temperature(℃)
2.0V
2.5V
3.0V
3.5V
4.0V
4.5V
5.0V
5.5V


Min Operating Voltage(RDCTL=8ns)
2.40
2.20
2.00
1.80
1.60
1.40
1.20
Vcc(V)
-40℃ -20℃ 0℃ 25℃ 50℃ 70℃ 85℃ 105℃ 125℃
Temperature(℃)
FIRC16MHz PWMCKS=FIRC*1
FIRC8MHz PWMCKS=FIRC*1
FIRC16MHz PWMCKS=FIRC*2
FIRC8MHz PWMCKS=FIRC*2
POR
LVRE=2.3V


Bandgap voltage vs. Vcc
1.230
1.220
1.210
1.200
1.190
1.180
1.170
VBG(V)
2.0V 2.5V 3.0V 3.5V 4.0V 4.5V 5.0V 5.5V
Vcc(V)
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃
125℃


LDOC Voltage vs. VCC
1.230
1.220
1.210
1.200
1.190
1.180
1.170
LDOC(V)
2.0V
2.5V
3.0V
3.5V
4.0V
4.5V
5.0V
5.5V
VCC(V)
-40℃
-20℃
0℃
25℃
50℃
70℃
85℃
105℃
125℃


LDOC Current vs. Voltage
80
70
60
50
40
30
20
10
0
LDOC Current(mA)
1.14
1.15
1.16
1.17
1.18
1.19
1.2
1.21
LDOC Voltage(V)
VDD=3~4.5V

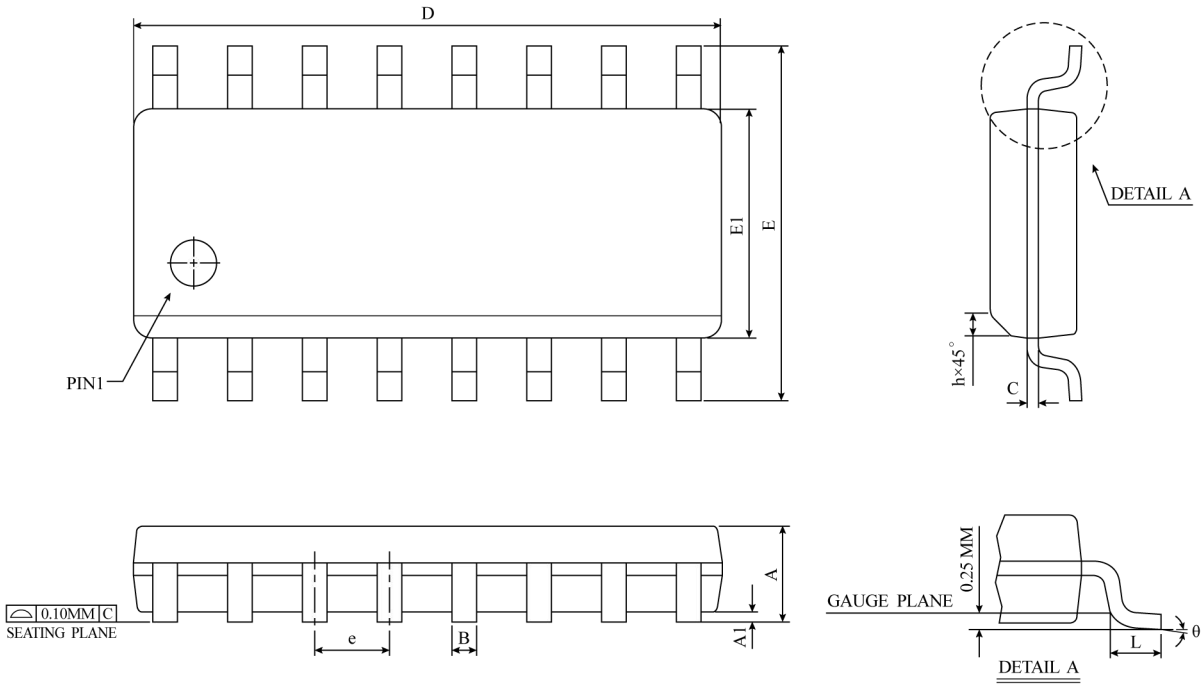
封装信息

请注意，此处提供的包装信息仅供参考。 由于此信息经常更新，因此用户可以联系销售人员以咨询最新的包装信息和库存。

订购信息：

Ordering number	Package
DC8S3522-MTP	Wafer / Dice 空片
DC8S3522-COD	Wafer / Dice 带代码
DC8S3522-MTP-16	SOP 16-pin (150 mil)
DC8S3522-MTP-53	MSOP 10-pin (118 mil)
DC8S3522-MTP-14	SOP 8-pin (150 mil)
DC8S3522-MTP-96	QFN 16-pin (3*3*0.75 - 0.5mm)
DC8S3522-MTP-B4	DFN 10-pin (3*3*0.75 - 0.5mm)

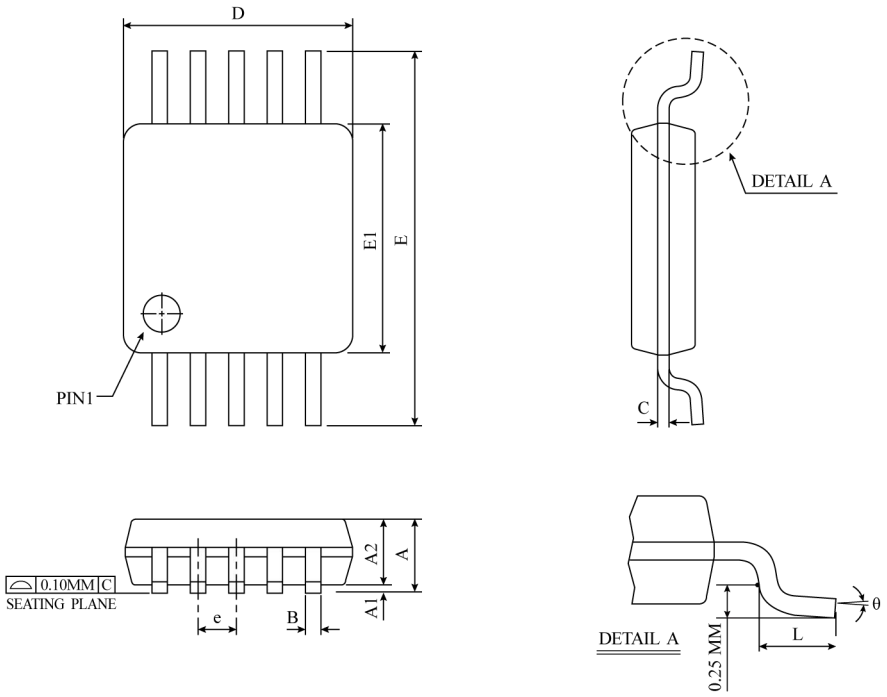
SOP-16 (150 mil) 封装尺寸



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	1.35	1.55	1.75	0.0532	0.0610	0.0688
A1	0.10	0.18	0.25	0.0040	0.0069	0.0098
B	0.33	0.42	0.51	0.0130	0.0165	0.0200
C	0.19	0.22	0.25	0.0075	0.0087	0.0098
D	9.80	9.90	10.00	0.3859	0.3898	0.3937
E	5.80	6.00	6.20	0.2284	0.2362	0.2440
E1	3.80	3.90	4.00	0.1497	0.1536	0.1574
e	1.27 BSC			0.050 BSC		
h	0.25	0.38	0.50	0.0099	0.0148	0.0196
L	0.40	0.84	1.27	0.0160	0.0330	0.0500
θ	0°	4°	8°	0°	4°	8°
JEDEC	MS-012 (AC)					

△ * NOTES：DIMENSION "D" DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS. MOLD FLASH, PROTRUSIONS AND GATE BURRS SHALL NOT EXCEED 0.15 MM (0.006 INCH) PER SIDE.

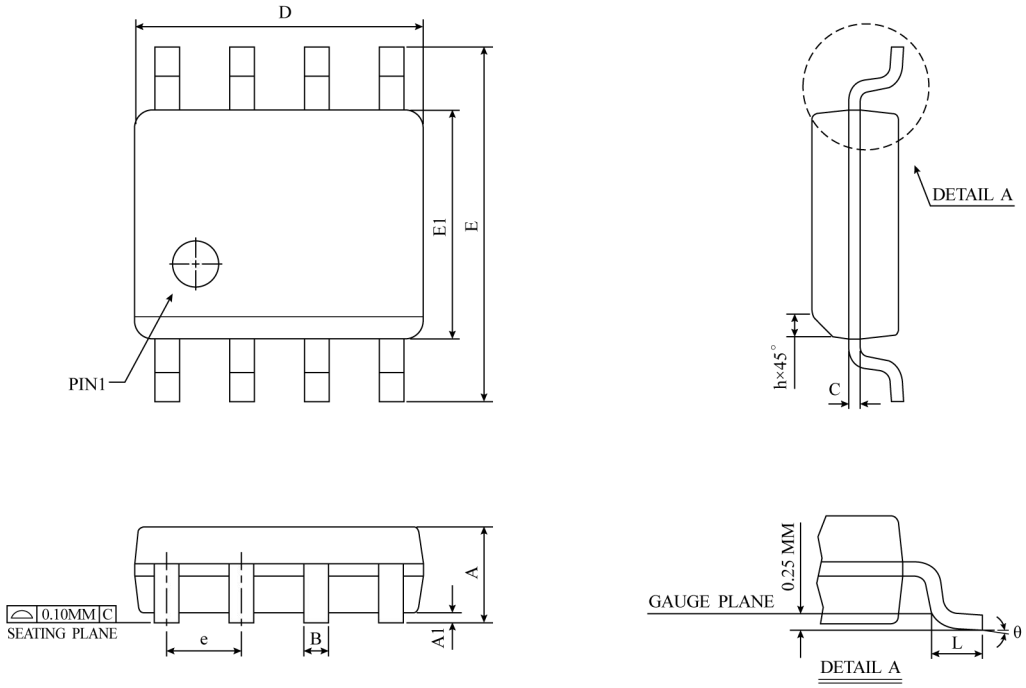
MSOP-10 (118 mil) 封装尺寸



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	0.81	0.96	1.10	0.032	0.038	0.043
A1	0.05	0.10	0.15	0.002	0.004	0.006
A2	0.75	0.85	0.95	0.030	0.034	0.037
B	0.17	0.22	0.27	0.007	0.009	0.011
C	0.13	0.18	0.23	0.005	0.007	0.009
D	2.90	3.00	3.10	0.114	0.118	0.122
E	4.75	4.90	5.05	0.187	0.193	0.199
E1	2.90	3.00	3.10	0.114	0.118	0.122
e	0.50 BSC			0.020 BSC		
L	0.40	0.55	0.70	0.016	0.022	0.028
θ	0°	3°	6°	0°	3°	6°
JEDEC						

⚠ * NOTES：DIMENSION〝D〞DOES NOT INCLUDE MOLD PROTRUSIONS OR GATE BURRS.
MOLD PROTRUSIONS AND GATE BURRS SHALL NOT EXCEED 0.12 MM (0.005 INCH) PER SIDE.
DIMENSION〝E1〞DOES NOT INCLUDE MOLD PROTRUSIONS
MOLD PROTRUSIONS SHALL NOT EXCEED 0.25 MM (0.010 INCH) PER SIDE.

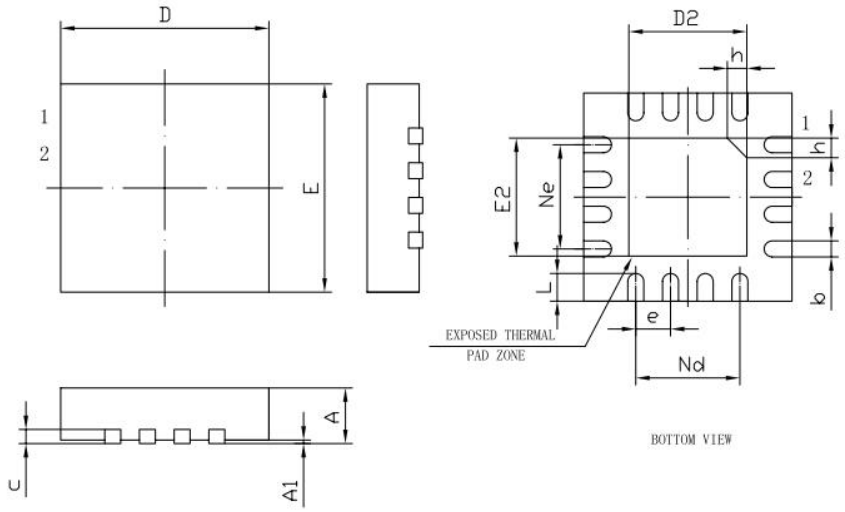
SOP-8 (150 mil) 封装尺寸



SYMBOL	DIMENSION IN MM			DIMENSION IN INCH		
	MIN	NOM	MAX	MIN	NOM	MAX
A	1.35	1.55	1.75	0.0532	0.0610	0.0688
A1	0.10	0.18	0.25	0.0040	0.0069	0.0098
B	0.33	0.42	0.51	0.0130	0.0165	0.0200
C	0.19	0.22	0.25	0.0075	0.0087	0.0098
D	4.80	4.90	5.00	0.1890	0.1939	0.1988
E	5.80	6.00	6.20	0.2284	0.2362	0.2440
E1	3.80	3.90	4.00	0.1497	0.1536	0.1574
e	1.27 BSC			0.050 BSC		
h	0.25	0.38	0.50	0.0099	0.0148	0.0196
L	0.40	0.84	1.27	0.0160	0.0330	0.0500
θ	0°	4°	8°	0°	4°	8°
JEDEC	MS-012 (AA)					

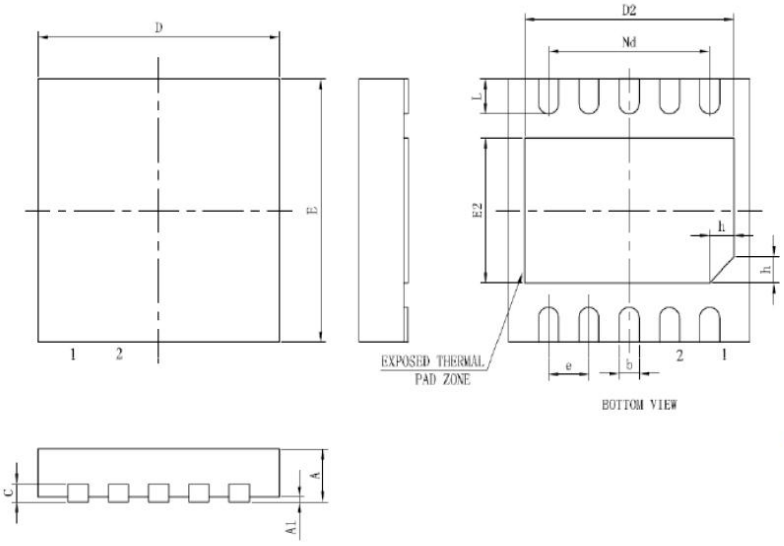
⚠ * NOTES：DIMENSION "D" DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS.
MOLD FLASH, PROTRUSIONS AND GATE BURRS SHALL
NOT EXCEED 0.15 MM (0.006 INCH) PER SIDE.

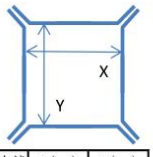
QFN-16 (3*3*0.75-0.5mm) 封装尺寸



SYMBOL	MILLIMETER		
	MIN	NOM	MAX
A	0.70	0.75	0.80
A1	—	0.02	0.05
b	0.18	0.25	0.30
c	0.18	0.20	0.25
D	2.90	3.00	3.10
D2	1.55	1.65	1.75
e	0.50BSC		
Ne	1.50BSC		
Nd	1.50BSC		
E	2.90	3.00	3.10
E2	1.55	1.65	1.75
L	0.35	0.40	0.45
h	0.20	0.25	0.30
L/F载体尺寸 (mil)	75x75		

DFN-10 (3*3*0.75-0.5mm) 封装尺寸





材质	Pad连线	X(mm)	Y(mm)
Cu	Y	2.7	1.9

ITEM	MILLIMETER		
	Min	Nom.	Max.
A	0.70	0.75	0.80
A1	-	0.02	0.05
b	0.18	0.25	0.30
c	0.18	0.20	0.25
D	2.90	3.00	3.10
D2	2.40	2.50	2.60
e	0.50BSC		
Nd	2.00BSC		
E	2.90	3.00	3.10
E2	1.45	1.55	1.65
L	0.30	0.40	0.50
h	0.20	0.25	0.30

Plating	Sn-Bi
Thickness	7~20um