M8P625

8BIT
AD 型
MTP MCU

Version 2.07

2024 年 09 月

磐 芯 电 子

http://www.masses-chip.com/

本公司不承担由本手册所涉及的产品或电路的运用和使用所引起的任何责任，本公司的产品不是专门设计来应用于外科植入、生命维持和任何本公司产品的故障会对个体造成伤害甚至死亡的领域。如果将本公司的产品应用于上述领域，即使这些是由本公司在产品设计和制造上的疏忽引起的，用户应赔偿所有费用、损失、合理的人身伤害或死亡所直接或间接产生的律师费用，并且用户保证本公司及其雇员、子公司、分支机构和销售商与上述事宜无关。

目录

1 产品简述

M8P625 是一颗采用高速低功耗 CMOS 工艺设计开发的 8 位高性能精简指令单片机，内部有 2K*16 位多次擦写编程存储器（MTP，擦写次数 1000 次），128*8 位 EEPROM（擦写次数 10000 次），128*8 位的数据存储器（RAM），18 个双向 I/O 口，2 个 8/16 位 Timer 定时器/计数器，1 路 UART，1 路 IIC SLAVE，1 路 8 位 PWM，1 路 8+4 位分辨率的互补 PWM，12+3 路 12 位 AD 转换器，支持多种系统工作模式和多个中断源。

1.1 特性

- CPU 特性
  - 高性能精简指令
  - 2K*16位的MTP程序存储器
  - 128*8位的EEPROM
  - 128*8位的数据存储器
  - 8级堆栈缓存器
  - 支持查表指令
- I/O 口
  - 最多18个双向I/O口（IOB2开漏输出）
  - 所有端口可设置弱上拉
  - IOB端口可设置弱下拉
  - IOB口变化中断
  - 两路外部中断
- 2 个定时器/计数器
  - TC0/TC1：具有自动装载功能的定时/计数器
- 2 路 PWM
  - PWM0：8位分辨率PWM
  - PWM1：8+4位分辨率PWM，带死区控制及互补输出
- 系统时钟
  - 内部高速RC振荡器： 16MHz
  - 内部低速RC振荡器： 64KHz/500KHz
  - 外部高速晶体振荡器：500KHz-20MHz
  - 外部低速晶体振荡器：0-500KHz
- 系统工作模式
  - 普通模式
  - 绿色模式
  - 休眠模式
- 12+3 路 12 位 ADC
  - 内嵌参考电压2V、3V、4V、VDD
  - 12路外部输入
  - 1路内部电源电压检测VDD/4
  - 1路内部GND电压检测
  - 1路内部参考电压检测
- 多路中断/唤醒源
  - 定时器中断：TC0/TC1
  - IOB口变化中断
  - ADC转换中断
  - 外部中断
- 看门狗定时器
- 特殊功能
  - 可编程代码保护
  - ISP功能
- 封装形式
  - MSOP10
  - SOP16
  - SOP20

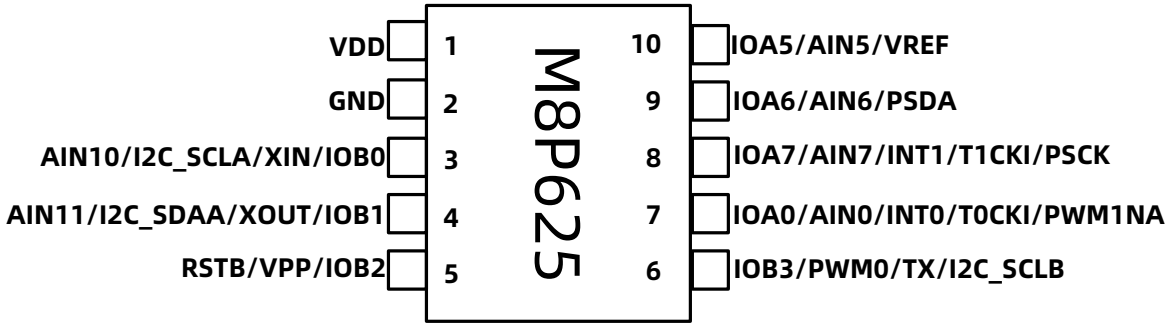
1.2 应用注意事项

1、使用外部中断 INT0、INT1 要注意 ，当中断触发和进休眠的操作 （即 STOP 从 0 到 1）同时发生时，会导致外部中断 INT0、INT1 无效且一直无法唤醒休眠，如需使用外部中断，尽量使用 IO 变化中断，如果必须使用外部中断 INT0、INT1，必须要开启 WDT 作为唤醒源，WDT 唤醒后会立马进中断程序。
2、使用 UART 时，注意内部高频振荡器的电压特性曲线，建议使用和实际应用电压相同或接近的烧录电压进行烧录。
3、使用 ADC 要注意，ADC 输入通道选择 VDD/4 在休眠下也会产生电流。
4、使用 VPP 口时注意，VPP 口只有输出低有驱动能力，在烧录时可达到 11V 的高压。
5、使用 ADC 时需注意，ADC 使用时最好去除最大和最小，取中间平均值，偶尔可能会有错误的值出来，在 ADC 采样转换之前、切换通道和参考电压都要注意延时一下 16us。
6、EFT 选 8T 抗干扰性更好一些。

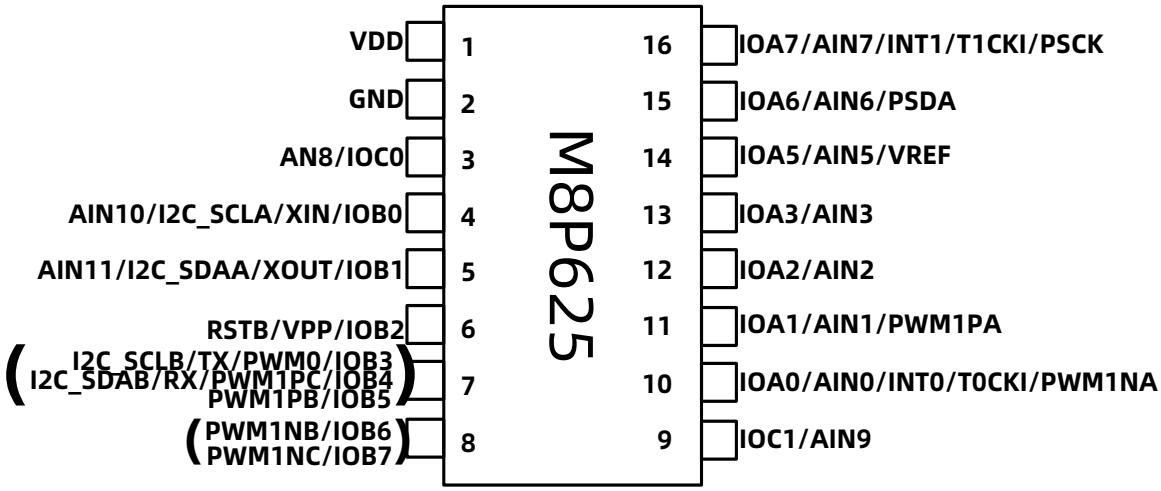
1.3 引脚图

注：芯片仿真烧录口分别是VDD、PSDA、PSCK、VPP、GND。

1.3.3 MSOP10

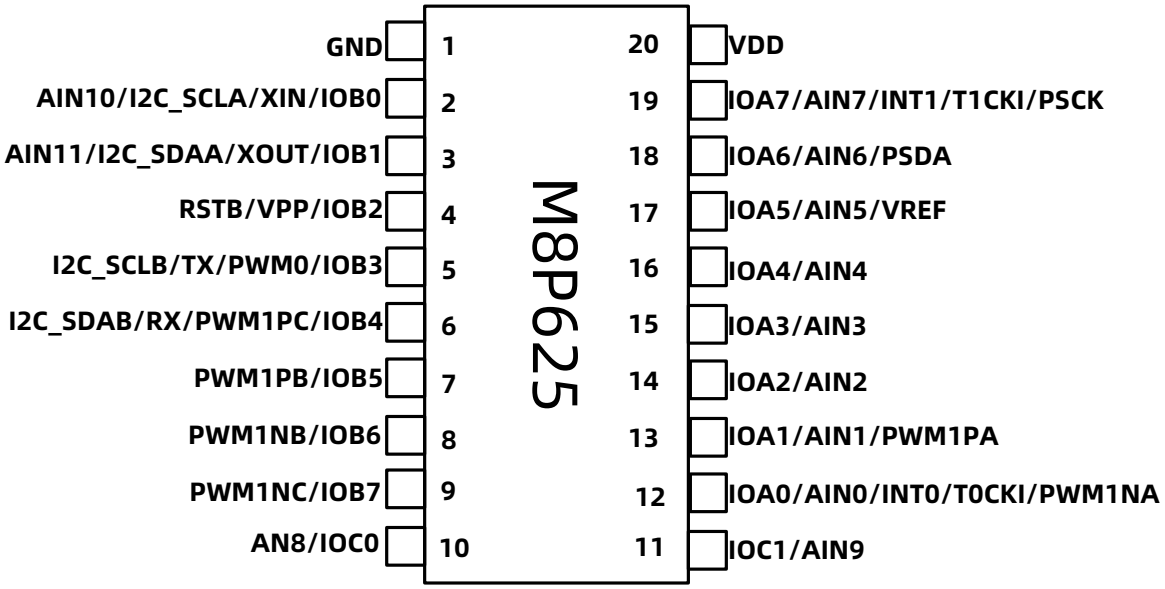


1.3.2 SOP16



注：引脚 7 是 IOB3、IOB4、IOB5 内部是短接在一起的，引脚 8 是 IOB6、IOB7 内部是短接在一起的，用到其中一个 IO 时，另外的 IO 用做输入。

1.3.1 SOP20


GND
1
AIN10/I2C_SCLA/XIN/IOB0
2
AIN11/I2C_SDAA/XOUT/IOB1
3
RSTB/VPP/IOB2
4
I2C_SCLB/TX/PWM0/IOB3
5
I2C_SDAB/RX/PWM1PC/IOB4
6
PWM1PB/IOB5
7
PWM1NB/IOB6
8
PWM1NC/IOB7
9
AN8/IOC0
10
M8P625
20
VDD
19
IOA7/AIN7/INT1/T1CKI/PSCK
18
IOA6/AIN6/PSDA
17
IOA5/AIN5/VREF
16
IOA4/AIN4
15
IOA3/AIN3
14
IOA2/AIN2
13
IOA1/AIN1/PWM1PA
12
IOA0/AIN0/INT0/T0CKI/PWM1NA
11
IOC1/AIN9

1.4 引脚描述

名称	类型	说明
VDD,GND	P	电源输入端
IOA0	I/O	输入/输出 IO，SMT，上拉电阻
AIN0	A	AD 通道 0
T0CKI	I	TC0 外部时钟输入
INT0	I	外部中断
PWM1NA	O	PWM1 反相输出
IOA1	I/O	输入/输出 IO，SMT，上拉电阻
AIN1	A	AD 通道 1
PWM1PA	O	PWM1 正相输出
IOA2	I/O	输入/输出 IO，SMT，上拉电阻
AIN2	A	AD 通道 2
IOA3	I/O	输入/输出 IO，SMT，上拉电阻
AIN3	A	AD 通道 3
IOA4	I/O	输入/输出 IO，SMT，上拉电阻
AIN4	A	AD 通道 4
IOA5	I/O	输入/输出 IO，SMT，上拉电阻
AIN5	A	AD 通道 5
VREF	A	AD 外部参考电压输入
IOA6	I/O	输入/输出 IO，SMT，上拉电阻
AIN6	A	AD 通道 6
PSDA	I/O	编程用
IOA7	I/O	输入/输出 IO，SMT，上拉电阻
AIN7	A	AD 通道 7
INT1	I	外部中断
T1CKI	I	TC1 外部时钟输入
PSCK	I/O	编程用
IOB0	I/O	输入/输出 IO，SMT，上拉电阻、下拉电阻、变化中断
XIN	A	外部晶体振荡器接口
AIN10	A	AD 通道 10
I2C_SCLA	I/O	I2C 通讯时钟口
IOB1	I/O	输入/输出 IO，SMT，上拉电阻、下拉电阻、变化中断
XOUT	A	外部晶体振荡器接口
AIN11	A	AD 通道 11
I2C_SDAA	I/O	I2C 通讯数据口
IOB2	I/O	输入/输出 IO，SMT，上拉电阻，变化中断
RSTB	I	外部复位输入，上拉电阻
VPP	P	编程高压电源

名称	类型	说明
IOB4 PWM1PC RX I2C_SDAB	I/O O I I/O	输入/输出 IO，SMT，上拉电阻、下拉电阻、变化中断 PWM1 正相输出 通用异步串行通讯接收口 I2C 通讯数据口
IOB5 PWM1PB	I/O O	输入/输出 IO，SMT，上拉电阻、下拉电阻、变化中断 PWM1 正相输出
IOB6 PWM1NB	I/O O	输入/输出 IO，SMT，上拉电阻、下拉电阻、变化中断 PWM1 反相输出
IOB7 PWM1NC	I/O O	输入/输出 IO，SMT，上拉电阻、下拉电阻、变化中断 PWM1 反相输出
IOC0 AIN8	I/O A	输入/输出 IO，SMT，上拉电阻、下拉电阻、变化中断 AD 通道 8
IOC1 AIN9	I/O A	输入/输出 IO，SMT，上拉电阻、下拉电阻、变化中断 AD 通道 9

注：I = 输入　O = 输出　I/O = 输入/输出　P = 电源　A = 模拟端口

2 中央处理器（CPU）

2.1 程序存储器

地址	说明
0x0000	复位向量
0x0001 ~ 0x0007	用户区
0x0008	中断向量
0x0009 ~ 0x07EF	用户区
0x07F0 ~ 0x07FF	厂商保留区
0x0800 ~ 0x087F	EEPROM

2.1.1 复位向量（0000H）

M8P625有以下四种复位方式

- 上电复位
- 看门狗复位
- 外部复位
- 欠压复位

发生上述任一种复位后，程序将从0000H处重新开始执行，系统寄存器也将都恢复为初始默认值。

例：定义复位向量

```
        ORG         0000H
        GOTO        Main_Program        ;//跳转至用户程序开始
        ...
Main_Program:                           ;//用户程序开始
        ...
Main:
        ...
        GOTO        Main                ;//用户主程序循环
```

2.1.2 中断向量（0008H）

M8P625中断向量地址为0008H。 一旦有中断响应，程序计数器PC的当前值就会存入堆栈缓存器并跳转到0008H处开始执行中断服务程序。

例：中断服务程序

```
        ORG     0000H
        GOTO    Main_Program        ;//跳转到程序开始
        ORG     0008H
        GOTO    Interrupt           ;//发生中断后,跳转到中断子程序
Main_Program:
        ...
Main:
        ...
        GOTO    Main                ;//主程序循环

Interrupt:
        PUSH                        ;//压栈、保存A、STATUS
        ...
        POP                         ;//出栈、恢复 A、STATUS
        RETIE

        END
```

2.1.3 查表

使用RDT指令可以读取程序区数据，其中读到的16位数据高位放在HBUF中，低位放在A寄存器中。FSR1和FSR0组成12位程序区数据寻址指针。

例 1：查找 ROM 地址为“DTAB”的值

```
        MOVIA   0               ;//要查的数据在表中的位置
        ADDIA   LOW(DTAB)       ;//获取数据表地址低位
        MOVAR   FSR0            ;//设置数据表低位指针
        MOVIA   0               ;//要查的数据在表中的位置
        ADCIA   HIGH(DTAB)      ;//获取数据表地址高位
        MOVAR   FSR1            ;//设置数据表高位指针
                                ;//若需读取表的其它数据,修改指针
        RDT                     ;//读取表的第一个数据0x0102
        MOVAR   TABDL           ;//将低位数据0x02放在TABDL
        MOVR    HBUF,A          ;//高位数据读入累加器A
        MOVAR   TABDH           ;//将高位数据0x01放在TABDH
        ...
DTAB:
        DW      0x0102
        DW      0x1112
```

使用加 PCL 地址来跳转，通过 GOTO 指令可以跳转不同的程序标号。

例 2:+PCL　GOTO 表

```
        MOVR        ADDRESS,A          ;//获取表格地址
        ADDAR       PCL,R
        GOTO        TAB1               ;//PCL +0 处理程序
        GOTO        TAB2               ;//PCL +1 处理程序
        GOTO        TAB3               ;//PCL +2 处理程序
TAB1:
            处理程序
TAB2:
            处理程序
TAB3:
            处理程序
```

使用加PCL地址来跳转，通过RETIA指令可以读取数据表。

例 3:+PCL　RETIA 表

```
        MOVR        ADDRESS,A          ;//获取地址
        ADDAR       PCL,R              ;//地址指针加 1
        RETIA       0                  ;//PCL +0
        RETIA       1                  ;//PCL +1
        RETIA       2                  ;//PCL +2
        ...
```

2.2 EEPROM

“Electrically Erasable Programmable Read Only Memory”为可擦可编程只读存储器，由于其非易失的存储结构，即使在电源掉电的情况下存储器内的数据仍然保存完好。EEPROM 的写入电压为2.7V~5.5V，可以读写 10000 次。

2.2.1 EECON1 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EECON1	EEMOD	-	-	-	-	-	WERR	EEWEN
读/写	R/W	-	-	-	-	-	R	R/W
复位后	0	-	-	-	-	-	0	0

Bit 7 **EEMOD**： EEPROM写入模式使能
0 = 退出EEPROM写入模式
1 = 使能EEPROM写入模式

Bit 1 **WERR**： EEPROM写错误标志
0 = 未发生EEPROM写入失败
1 = EEPROM写入失败

Bit 0 **EEWEN**： EEPROM写入使能
0 = 退出EEPROM写入
1 = 使能EEPROM写入

注：（1）所有非EECON1和EECON2的寄存器写入操作，将使EEWEN清零。
（2）在写EEPROM时要关闭所有中断。

2.2.2 EECON2 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EECON2	-	-	-	-	EELOCK3	EELOCK2	EELOCK1	EEWR
读/写	-	-	-	-	W	W	W	W
复位后	-	-	-	-	0	0	0	0

Bit 3 EELOCK3： 解锁流程1
在EECON1[0]=1，EECON2=00H时，使用置位指令置位EELOCK3，完成EE或MTP写入解锁流程1，完成该操作后，EECON2的值为08H

Bit 2 EELOCK2： 解锁流程2
在EECON1[0]=1，EECON2=08H时，使用置位指令置位EELOCK2，完成EE或MTP写入解锁流程2，完成该操作后，EECON2的值为04H

Bit 1 EELOCK1： 解锁流程3
在EECON1[0]=1，EECON2=04H时，使用置位指令置位EELOCK2，完成EE或MTP写入解锁流程2，完成该操作后，EECON2的值为02H

Bit 0 EEWR： 启动写入操作
在EECON1[0]=1，EECON2=02H时，使用置位指令置位EEWR，启动EE写入操作，写入操作完成后，EECON2=00H

例：写 EE

```
            MOVIA       HIGH(DTAB)      ;//获取EE地址高位
            MOVAR       FSR1            ;//设置EE地址高位指针
            MOVIA       LOW(DTAB)       ;//获取EE地址低位
            MOVAR       FSR0            ;//设置EE地址低位指针
            MOVIA       0x55            ;//设置EE写入数据,即将0x55送入ACC为写入
                                        ;//的EE数据

            BSET        bEEMOD          ;//读写EE/MTP时需置位
            BSET        bEEWEN          ;//使能EE/MTP写入
            BSET        bEELOCK3        ;//EE/MTP写入解锁流程1
            BSET        bEELOCK2        ;//EE/MTP写入解锁流程2
            BSET        bEELOCK1        ;//EE/MTP写入解锁流程3
            BSET        bEEWR           ;//写入
                                        ;//CPU暂停,等待EE写入结束
            NOP
            BCLR        bEEMOD          ;//写完EE,清除该位
            JBTS0       bWERR
            GOTO        WeeErrProcess   ;//写入失败,跳转到错误处理程序
            ...
;//     EEPROM地址区间为0x0800-0x087F
```

注：EEPORM或MTP写入时，请关闭所有唤醒源，不然唤醒源可能中断写入操作，造成写入失败，写入电压 VDD：2.7V~5.5V

例：读 EE

```
        MOVIA     HIGH(DTAB)     ;//获取数据表地址高位
        MOVAR     FSR1           ;//设置数据表高位指针
        MOVIA     LOW(DTAB)      ;//获取数据表地址低位
        MOVAR     FSR0           ;//设置数据表低位指针
                                 ;//若需读取表的其它数据,修改指针
        BSET      bEEMOD         ;//读EE,需置位
        RDT                      ;//读取EE
        BCLR      bEEMOD         ;//清除该位
        MOVAR     TABDL          ;//将EE数据放在TABDL

;//    EEPROM地址区间为0x0800-0x087F
```

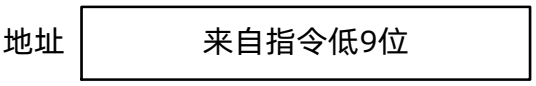
2.3 数据存储器

2.3.1 数据存储器结构

地址	间接寻址 INDF0	间接寻址 INDF1	间接寻址 INDF2	直接寻址
0x100 ~ 0x1FF	NO	YES	YES	YES
0x000 ~ 0x0FF	YES	NO		

2.3.2 数据存储器寻址模式

☆ 直接寻址模式



如：MOVAR 0x001；A 中的值传送给地址为 0x001 的 RAM 中

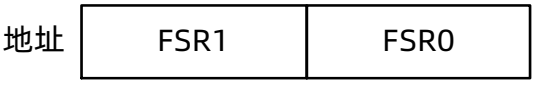
☆ 间接寻址模式 0

如：MOVAR INDF0；A 中的值传送给 FSR0 指向的 RAM 中

☆ 间接寻址模式 1

如：MOVAR INDF1；A 中的值传送给 FSR1 指向的 RAM 中

☆ 间接寻址模式 2



如：MOVAR INDF2； A 中的值传送给{FSR1:FSR0}指向的 RAM 中

2.3.3 系统寄存器定义

数据寄存器映射表								
	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
0x000 ~ 0x070	GPR							
0x080 ~ 0x0A0	RESERVE							
0x0A8	-	-	-	-	-	-	-	-
0x0B0	INDF0	FSR0	-	-	-	-	-	-
0x0B8	INDF1	FSR1	PCL	STATUS	OPTION	OSCM	WDTC	IOBICR
0x0C0	INDF2	HBUF	PDB	-	INTCR0	INTF0	INTCR1	INTF1
0x0C8	IOA	OEA	PUA	ANSA	IOB	OEB	PUB	ANSB
0x0D0	IOC	OEC	PUC	ANSC	IOIDS	I2CADR	I2CCON	I2CBUF
0x0D8	PWM0CR	PWM0D	TXCR	TXREG	RXCR	RXREG	BRGDH	BRGDL
0x0E0	ADCON0	ADCON1	ADL	ADH	ADCON2	T1CR	TC1CL	TC1CH
0x0E8	T0CR	TC0CL	TC0CH	PWM1DEAD	PWM1S	PWM1CR	PWM1DL	PWM1DH
0x0F0	EECON1	EECON2	-	-	-	-	-	-
0x0F8	-	-	-	-	-	-	-	-
0x1F8	-	-	VREFCAL	-	IRCCAL	-	-	-

2.3.4 INDF0 间接寻址寄存器 0

访问INDF0寄存器时，实现间接寻址模式0，访问到的是FSR0寄存器所指向的寄存器内容，间接寻址模式0仅可寻址通用寄存器区0x0000~0x00FF空间。

2.3.5 INDF1 间接寻址寄存器 1

访问INDF1寄存器时，实现间接寻址模式1，访问到的是FSR1寄存器所指向的寄存器内容，间接寻址模式1仅可寻址通用寄存器区0x0100~0x01FF空间。

2.3.6 FSR0 间接寻址指针 0

使用间接寻址模式0访问通用寄存器时，FSR0为地址指针；当以间接寻址模式2访问通用寄存器时，FSR0作为地址指针的低位。

2.3.7 FSR1 间接寻址指针 1

使用间接寻址模式1访问通用寄存器时，FSR1为地址指针；当以间接寻址模式2访问通用寄存器时，FSR1作为地址指针的高位。

2.3.8 HBUF 查表数据高 8 位

使用RDT指令读取程序区数据时，读到的16位数据高8位放在HBUF中。

2.3.9 PCL 程序计数器指针低位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PCL	PCL7	PCL6	PCL5	PCL4	PCL3	PCL2	PCL1	PCL0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

Bit [7:0] **PCL[7:0]**： 程序计数器指针低位

用户将该PCL作为目的操作数做加法运算时（ADDAR PCL、ADCAR PCL），13位PC值参与运算，运算结果写入PC，实现程序的相对跳转；加法运算外的其它运算时，仅PCL参与运算，PCH保持不变。PCH不可寻址。

2.3.10 STATUS 状态寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
STATUS	-	-	-	-	-	Z	DC	C
读/写	-	-	-	-	-	R/W	R/W	R/W
复位后	-	-	-	-	-	X	X	X

Bit 2 **Z**： 零标志

0 = 算术/逻辑运算的结果非零

1 = 算术/逻辑运算的结果为零

Bit 1 **DC**： 辅助进位标志

0 = 加法运算时低四位没有进位，或减法运算后有向高四位借位

1 = 加法运算时低四位有进位，或减法运算后没有向高四位借位

Bit 0 **C**： 进位标志

0 = 加法运算后没有进位、减法运算有借位发生或移位后移出逻辑“0”

1 = 加法运算后有进位、减法运算没有借位发生或移位后移出逻辑“1”

3 复位

3.1 复位方式

- 上电复位（POR）
- 外部复位（MCLRB Reset）
- 欠压复位（BOR）
- 看门狗定时器复位（WDT Reset）

M8P625 有以上 4 种复位方式，任何一种复位都会使 PC 程序计数器清零，让程序从 0000H 处开始运行，并且使系统寄存器值复位。

4 系统时钟

4.1 概述

M8P625支持双时钟系统：高速时钟和低速时钟。高速时钟由外部晶体振荡器或内置的16MHz RC振荡电路（HIRC 16MHz）提供，低速时钟由外部低速晶体振荡器（32768Hz）或内置的低速RC振荡电路（LIRC 64KHz/500KHz）提供。两种时钟都可作为系统时钟源Fosc，系统工作在低速模式时，Fosc 2分频后为一个指令周期。低频系统时钟源和高频系统时钟源可根据芯片配置字进行配置。

注：工作时勿在进行高低频切换同时 STOP CPU 操作，可能会造成系统紊乱。

4.2 OSCM 寄存器

工作模式控制寄存器 OSCM

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
OSCM	STBH	STBL	HSPDX2	STOP	CLKM	STPH	LPSPD	STPL
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	X	X	0	0	X	1	0	1

Bit 7 **STBH**： 高频振荡器稳定标志

Bit 6 **STBL**： 低频振荡器稳定标志

Bit 5 **HSPDX2:** 高频振荡器倍频（仅用于定时器选用高频作为时钟）

0 = 不使能倍频

1 = 倍频使能

Bit 4 **STOP**： CPU工作状态标志位

0 = CPU正常工作

1 = CPU停止工作，所有中断或看门狗溢出唤醒

Bit 3 **CLKM**： 系统时钟状态标志位

0 = CPU运行于高频时钟

1 = CPU运行于低频时钟

Bit 2 **STPH**： 高频振荡器控制

0 = 休眠状态或低速模式下高速振荡器仍然工作

1 = 休眠状态或低速模式下关闭高频振荡器

Bit 1 **LPSPD**： 低频振荡器频率选择

0 = 内部低频振荡器频率64KHz

1 = 内部低频振荡器频率500KHz

Bit 0 **STPL**： 低频振荡器控制

0 = 休眠状态下低频振荡器仍然工作

1 = 休眠状态下低频振荡器停止工作

注：CLKM 的初始状态由配置字决定。

4.3 系统时钟的工作模式

普通模式：普通模式有两种分别是：1.高频时钟工作，低频时钟工作，不进 STOP
（电流特性参考电性参数表 I_{DD1}）
2.高频时钟停止，低频时钟工作，不进 STOP

绿色模式：绿色模式有两种分别是：1.高频时钟工作，低频时钟工作，进 STOP
（电流特性参考电性参数表 I_{SP1}）
2.高频时钟停止，低频时钟工作，进 STOP
（电流特性参考电性参数表 I_{SP2}）
绿色模式可以由所有中断或 WDT 唤醒。

休眠模式：休眠模式只有一种是：高频时钟停止，低频时钟停止，进 STOP
（电流特性参考电性参数表 I_{SP3}）
休眠模式可以由外部中断、IO 变化中断或 WDT 唤醒。

注：（1）省电建议，程序运行时跑高频，快速跑完程序然后进休眠，此时休眠下需设置高频时钟停止工作。
（2）各工作模式的工作电流参考电性参数表。
（3）绿色和休眠模式下，如果总中断不开启，所有中断唤醒能唤醒芯片但是不会进中断。

4.4 IRCCAL 寄存器

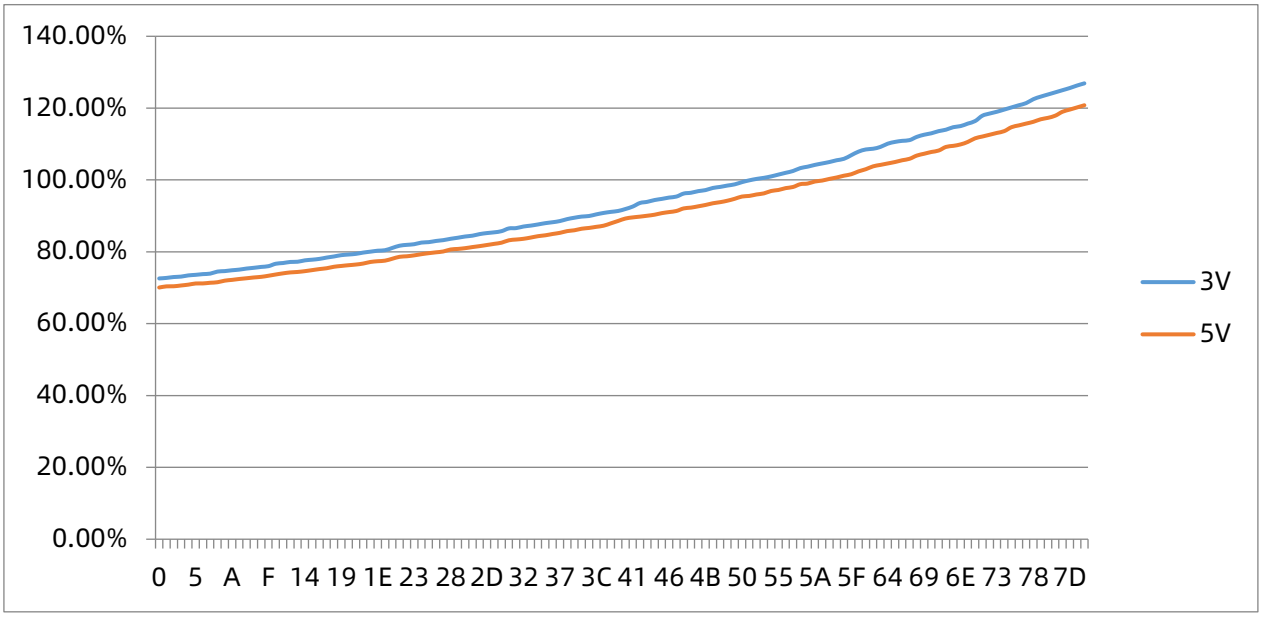
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IRCCAL	-	IRCAL6	IRCCAL5	IRCCAL4	IRCCAL3	IRCCAL2	IRCCAL1	IRCCAL0
读/写	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	-	X	X	X	X	X	X	X

内置的高频 RC 振荡电路在芯片上电后频率为校准过的 16MHz，但程序中可以通过特殊的流程来调整此频率以满足特定应用需求。

例：调整 IRC 频率

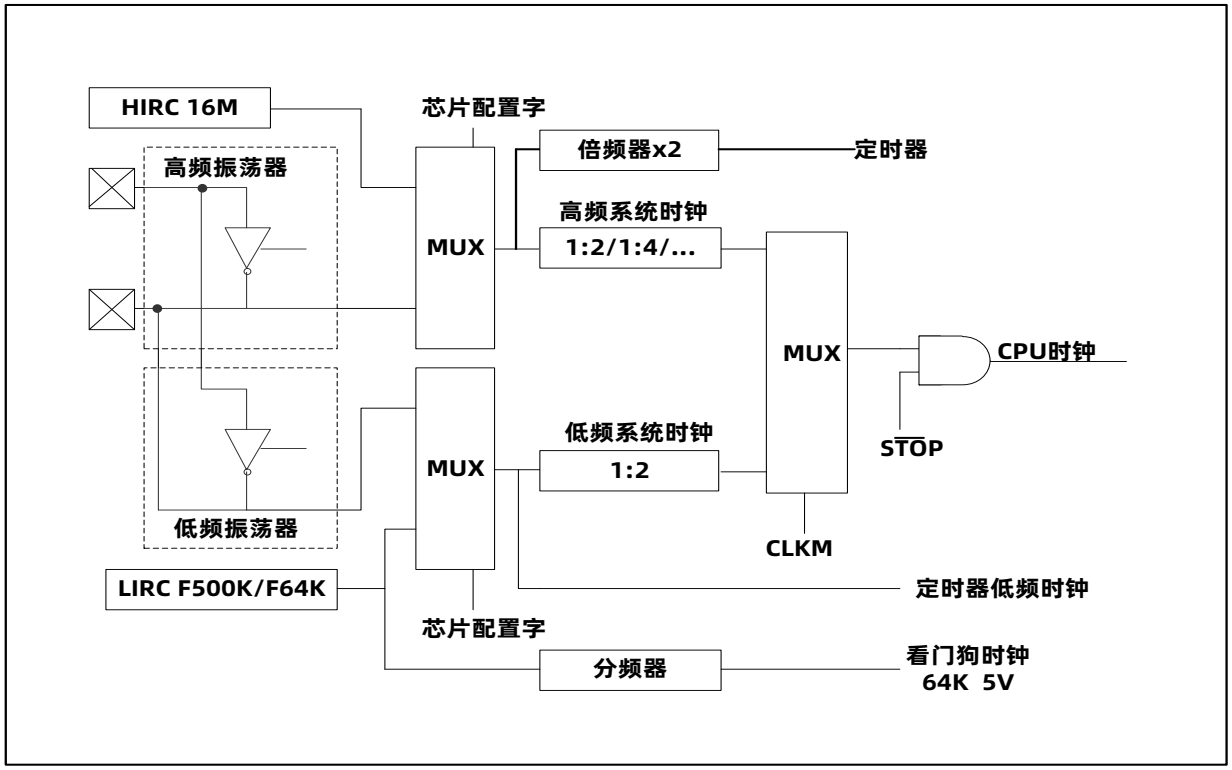
```
TASK_IRCCAL:
        MOVIA     0x55
        MOVAR     0x1F9          ;//1F9H地址写入055H
        MOVIA     0xAA
        MOVAR     0x1F9          ;//1F9H地址写入0AAH
        MOVIA     VALUE
        MOVAR     IRCCAL         ;//写入IRCCAL
        ...
;//若需继续在IRCCAL寄存器内写入其他值需要重复以上所有步骤
```

IRC 调整频率图



注：具体值不做设计保证。

4.5 系统时钟结构框图



	高速运行模式 (CLKM=0)	低速运行模式 (CLKM=1)	休眠模式 (STOP=1)
高频振荡器	运行	由 STPH 决定	由 STPH 决定
低频振荡器	运行	运行	由 STPL 决定
WDT	配置字决定	由配置字决定	由配置字决定
TC0/TC1	TCxEN	若选择高速时钟， 需 STPH=0	高速时钟源&STPH=0 低速时钟源&STPL=0

4.6 系统时钟高低频切换

高频振荡器稳定计数器：64 Clocks（内部 IRC 模式）/1024 Clocks（外部高频振荡器模式）
低频振荡器稳定计数器：16 Clocks（内部 RC 模式）/1024 Clocks（外部低频振荡器模式）

高低频切换时间：

高频切低频：1 个低频时钟周期+1 个高频时钟周期
低频切高频&STBH=0：1 个低频时钟周期+高频振荡器起振时间+高频振荡器稳定时间
低频切高频&STBH=1：1 个低频时钟周期+1 个高频时钟周期

唤醒时间：

CLKM=0&STBH=0：高频振荡器起振时间+高频振荡器稳定时间
CLKM=0&STBH=1：64 Clocks
CLKM=1&STBL=0：低频振荡器起振时间+低频振荡器稳定时间
CLKM=1&STBL=1：16 Clocks

5 中断

5.1 概述

M8P625有多路中断源：TC0/TC1，IO口变化中断，UART发送/接收中断，I2C中断，ADC中断，INT0、INT1中断。中断可以将系统从睡眠模式中唤醒，在唤醒前，中断请求被锁定。一旦程序进入中断，寄存器OPTION的位GIE被硬件自动清零以避免响应其它中断。系统退出中断后，硬件自动将GIE置“1”，以响应下一个中断。

设置 GIE 和中断控制寄存器 INTCR0/INTCR1 来使能中断，查询 INTF0/INTF1 中断标志寄存器判断中断是否发生。

注：使用外部中断 INT0、INT1 要注意，当中断触发和进休眠的操作（即 STOP 从 0 到 1）同时发生时，会导致外部中断 INT0、INT1 无效且无法唤醒休眠；
解决方法：如需使用外部中断尽量使用 IO 变化中断，如果必须使用外部中断 INT0、INT1，必须要开启 WDT 作为唤醒源。

5.2 OPTION 配置寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
OPTION	GIE	-	TO	PD	MINT11	MINT10	MINT01	MINT00
读/写	R/W	-	R	R	R/W	R/W	R/W	R/W
复位后	0	-	1	1	0	0	0	0

Bit 7 **GIE**： 全局中断控制位
0 = 屏蔽所有中断（响应中断后自动清零）
1 = 总中断使能（RETIE指令会将该位置1）

Bit [3:2] **MINT1[1:0]**： INT1中断模式选择

MINT1[1:0]	INT1 中断模式选择
00	上升沿中断
01	下降沿中断
1X	变化中断

Bit [1:0] **MINT0[1:0]**： INT0中断模式选择

MINT0[1:0]	INT0 中断模式选择
00	上升沿中断
01	下降沿中断
1X	变化中断

5.3 IO 变化中断使能寄存器

IOB 变化中断使能寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IOBICR	IOBICR7	IOBICR6	IOBICR5	IOBICR4	IOBICR3	IOBICR2	IOBICR1	IOBICR0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

Bit [7:0] **IOBICRx:** IO端口变化中断使能（x=0-7）

0 = 屏蔽IOB口电平变化中断

1 = 使能IOB口电平变化中断

5.4 INTCR0 中断控制寄存器 0

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTCR0	TKIE	ADIE	RXIE	TXIE	I2CIE	-	TC1IE	TC0IE
读/写	R/W	R/W	R/W	R/W	R/W	-	R/W	R/W
复位后	0	0	0	0	0	-	0	0

Bit 7 **TKIE**:

0 = 屏蔽触摸按键中断1

1 = 使能触摸按键中断1

Bit 6 **ADIE**:

0 = 屏蔽ADC转换中断

1 = 使能ADC转换中断

Bit 5 **RXIE**:

0 = 屏蔽串行通讯接收中断

1 = 使能串行通讯接收中断

Bit 4 **TXIE:**

0 = 屏蔽串行通讯发送中断

1 = 使能串行通讯发送中断

Bit 3 **I2CIE**:

0 = 屏蔽I2C中断

1 = 使能I2C中断

Bit 1 **TC1IE**:

0 = 屏蔽TC1溢出中断

1 = 使能TC1溢出中断

Bit 0 **TC0IE**:

0 = 屏蔽TC0溢出中断

1 = 使能TC0溢出中断

5.5 INTF0 中断标志寄存器 0

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTF0	TKIF	ADIF	RXIF	TXIF	I2CIF	-	TC1IF	TC0IF
读/写	R/W	R/W	R	R	R/W	-	R/W	R/W
复位后	0	0	0	0	0	-	0	0

注：除 USART 之外的所有中断标志位需软件清零。

Bit 7 **TKIF**:
0 = 未产生触摸按键中断1
1 = 产生触摸按键中断1

Bit 6 **ADIF**:
0 = 未产生ADC转换中断
1 = 产生ADC转换中断

Bit 5 **RXIF:**
0 = 未产生串行通讯接收中断
1 = 产生串行通讯接收中断

Bit 4 **TXIF:**
0 = 未产生串行通讯发送中断
1 = 产生串行通讯发送中断

Bit 3 **I2CIF:** (I2C Master写数据)
0 = 未收到数据或数据已读
1 = 缓冲区内接收到数据（读I2CBUF清零）

Bit 1 **TC1IF**:
0 = 未产生TC1溢出中断
1 = 产生TC1溢出中断

Bit 0 **TC0IF**:
0 = 未产生TC0溢出中断
1 = 产生TC0溢出中断

5.6 INTCR1 中断控制寄存器 1

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTCR1	-	-	-	-	INT1IE	INT0IE	-	IOBCHIE
读/写	-	-	-	-	R/W	R/W	-	R/W
复位后	-	-	-	-	0	0	-	0

Bit 3 **INT1IE**:
0 = 屏蔽外部端口中断1
1 = 使能外部端口中断1

Bit 2 **INT0IE**:
0 = 屏蔽外部端口中断0
1 = 使能外部端口中断0

Bit 0 **IOBCHIE**:
0 = 屏蔽端口B变化中断
1 = 使能端口B变化中断

5.7 INTF1 中断标志寄存器 1

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTF1	-	-	-	-	INT1IF	INT0IF	-	IOBCHIF
读/写	-	-	-	-	R/W	R/W	-	R/W
复位后	-	-	-	-	0	0	-	0

注：所有中断标志位需软件清零。

Bit 3 **INTF1IF**:
0 = 未产生外部中断INT1
1 = 产生外部中断INT1

Bit 2 **INTF0IF**:
0 = 未产生外部中断INT0
1 = 产生外部中断INT0

Bit 0 **IOBCHIF**:
0 = 对应输入端口状态未发生变化
1 = 对应输入端口状态发生变化

5.8 中断范例

例:IO 变化中断（以 IOB0 口为例）

```
;//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            ORG         0000H
            GOTO        Main_Program            ;//跳转到程序开始
            ORG         0008H
            GOTO        Interrupt               ;//发生中断后,跳转到中断子程序
;//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Main_Program:                                   ;//程序开始
IO_INTERRUPT_INIT:                              ;//IO 中断初始化
;//1、端口设置
            BSET        PUB,0                   ;//IOB0 端口设置为上拉
            BCLR         OEB,0                  ;//IOB0 端口设置为输入
;//2、允许 B 口变化中断设置
            BSET        IOBICR,0                ;//IOB 口变化中断使能
;//3、中断使能
            BSET        INTCR1, IOBCHIE
            BCLR        INTF1, IOBCHIF
;//4、总中断使能
            BSET        OPTION,GIE              ;//总中断使能
Main:                                           ;//程序主循环
            ...
            GOTO        Main
;//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Interrupt:                                      ;//中断子程序
;//中断进来
            PUSH                                ;//压栈、保存 A、STATUS
;//中断处理程序
            JBTS0       INTF1,IOBCHIF           ;//检测 IO 中断标志位
            GOTO        Interrupt_IO
Interrupt_End:
;//中断结束
            POP                                 ;//出栈、恢复 A、STATUS
            RETIE
Interrupt_IO:
            BCLR        INTF1, IOBCHIF
;//IO 中断处理程序
            ...
            GOTO        Interrupt_End

            END
```

例: INT0 中断

```
;//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            ORG         0000H
            GOTO        Main_Program            ;//跳转到程序开始
            ORG         0008H
            GOTO        Interrupt               ;//发生中断后,跳转到中断子程序
;//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Main_Program:                                   ;//程序开始
INT0_Init:                                      ;//INT0 初始化
;//1、端口设置
            BSET        PUA,0                   ;//INT0端口设置为上拉
            BCLR        OEA,0                   ;//INT0端口设置为输入
;//2、中断模式选择
            MOVIA       0x31                    ;//INT0下降沿中断
            MOVAR       OPTION
;//3、中断使能
            BSET        INTCR1, INT0IE          ;//INT0中断使能
            BCLR        INTF1, INT0IF
;//4、总中断使能
            BSET        OPTION,GIE              ;//总中断使能
Main:                                           ;//程序主循环
            ...
            GOTO        Main
;//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Interrupt:                                      ;//中断子程序
;//中断进来
            PUSH                                ;//压栈、保存 A、STATUS
;//中断处理程序
            JBTS0       INTF1,INT0IF            ;//检测 INT0 标志位
            GOTO        Interrupt_INT0
Interrupt_End:
;//中断结束
            POP                                 ;//出栈、恢复 A、STATUS
            RETIE
Interrupt_INT0:
            BCLR        INTF1,INT0IF            ;//清  INT0 标志位
;//INT0 中断处理程序
            ...
            GOTO        Interrupt_End

            END
```

6 端口

6.1 IOA

IOA 数据寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IOA	IOA7	IOA6	IOA5	IOA4	IOA3	IOA2	IOA1	IOA0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

IOA 方向寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
OEA	OEA7	OEA6	OEA5	OEA4	OEA3	OEA2	OEA1	OEA0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

Bit [7:0] **OEAx**: A口输出使能（x=0-7）

0 = 输入

1 = 输出

IOA 上拉使能寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PUA	PUA7	PUA6	PUA5	PUA4	PUA3	PUA2	PUA1	PUA0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

Bit [7:0] **PUAx**: A口上拉使能（x=0-7）

0 = 上拉关闭

1 = 上拉使能

IOA 端口模式控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ANSA	ANSA7	ANSA6	ANSA5	ANSA4	ANSA3	ANSA2	ANSA1	ANSA0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

Bit [7:0] **ANSAx**: A口模式控制（x=0-7）

0 = 作为数字IO口

1 = 作为模拟端口（IO输入功能屏蔽）

6.2 IOB

IOB 数据寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IOB	IOB7	IOB6	IOB5	IOB4	IOB3	IOB2	IOB1	IOB0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

IOB 方向寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
OEB	OEB7	OEB6	OEB5	OEB4	OEB3	OEB2	OEB1	OEB0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

Bit [7:0] **OEBx**： B口输出使能（x=0-7）

0 = 输入

1 = 输出

注：IOB[2]作为输出口的注意事项

（1）需将 PUB[2]置 1 才能输出高电平。

（2）IOB[2]输出的高电平是由上拉电阻提供的，所以驱动能力弱。

（3）IOB[2]输出的低电平驱动能力比其他端口弱，输出低电平时内部电路会关闭上拉电阻。

（4）IOB[2]只有输出低有驱动能力，在烧录时可达到 11V 的高压。

IOB 上拉使能寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PUB	PUB7	PUB6	PUB5	PUB4	PUB3	PUB2	PUB1	PUB0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

Bit [7:0] **PUBx**： B口上拉使能（x=0-7）

0 = 上拉关闭

1 = 上拉使能

IOB 下拉使能寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PDB	PDB7	PDB6	PDB5	PDB4	PDB3	-	PDB1	PDB0
读/写	R/W	R/W	R/W	R/W	R/W	-	R/W	R/W
复位后	0	0	0	0	0	-	0	0

Bit [7:0] **PDBx**： B口下拉使能（x=0,1,3-7）

0 = 下拉关闭

1 = 下拉使能

IOB 端口模式控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ANSB	-	-	-	-	-	-	ANSB1	ANSB0
读/写	-	-	-	-	-	-	R/W	R/W
复位后	-	-	-	-	-	-	0	0

Bit [1:0] **ANSBx**: B口模式控制（x=0,1）

0 = 作为数字IO口

1 = 作为模拟端口（IO输入功能屏蔽）

注：同一 IO 口上下拉电阻同时打开时，IO 口将自动屏蔽输入功能（读该端口状态为 0），此时端口电平接近于 VDD/2。

IOB 变化中断使能寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IOBICR	IOBICR7	IOBICR6	IOBICR5	IOBICR4	IOBICR3	IOBICR2	IOBICR1	IOBICR0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

Bit [7:0] **IOBICRx:** IO端口变化中断使能（x=0-7）

0 = 屏蔽IOB口电平变化中断

1 = 使能IOB口电平变化中断

6.3 IOC

IOC 数据寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IOC	-	-	-	-	-	-	IOC1	IOC0
读/写	-	-	-	-	-	-	R/W	R/W
复位后	-	-	-	-	-	-	0	0

IOC 方向寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
OEC	-	-	-	-	-	-	OEC1	OEC0
读/写	-	-	-	-	-	-	R/W	R/W
复位后	-	-	-	-	-	-	0	0

Bit [1:0] **OECx**: C口输出使能（x=0,1）

0 = 输入

1 = 输出

IOC 上拉使能寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PUC	-	-	-	-	-	-	PUC1	PUC0
读/写	-	-	-	-	-	-	R/W	R/W
复位后	-	-	-	-	-	-	0	0

Bit [1:0] **PUCx**: C口上拉使能（x=0,1）

0 = 上拉关闭

1 = 上拉使能

IOC 端口模式控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ANSC	-	-	-	-	-	-	ANSC1	ANSC0
读/写	-	-	-	-	-	-	R/W	R/W
复位后	-	-	-	-	-	-	0	0

Bit [1:0] **ANSCx**: C口模式控制（x=0,1）

0 = 作为数字IO口

1 = 作为模拟端口（IO输入功能屏蔽）

6.4 IOIDS 端口驱动控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IOIDS	IOBHIS	IOBLIS	IOAHIS	IOALIS	IOBHDS	IOBLDS	IOAHDS	IOALDS
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

Bit [7] **IOBHIS**: IOB7-4翻转电平选择

IOBHIS	IOB[7:4]电平选择
0	SMT
1	低反转点

Bit [6] **IOBLIS**: IOB3-0翻转电平选择

IOBLIS	IOB[3:0]电平选择
0	SMT
1	低反转点

Bit [5] **IOAHIS**: IOA7-4翻转电平选择

IOAHIS	IOA[7:4]电平选择
0	SMT
1	低反转点

Bit [4] **IOALIS**: IOA3-0翻转电平选择

IOAHIS	IOA[3:0]电平选择
0	SMT
1	低反转点

Bit [3] **IOBHDS**: IOB7-4输出驱动电流选择

IOBHDS	IOB[7:4]输出驱动电流选择
0	大驱动(I_{OL2}\ I_{OH2})
1	小驱动(I_{OL1}\ I_{OH1})

Bit [2] **IOBLDS**: IOB3-0输出驱动电流选择

IOBLDS	IOB[3:0]输出驱动电流选择
0	大驱动(I_{OL2}\ I_{OH2})
1	小驱动(I_{OL1}\ I_{OH1})

Bit [1] **IOAHDS**: IOA7-4输出驱动电流选择

IOAHDS	IOA[7:4]输出驱动电流选择
0	大驱动(I_{OL2}\ I_{OH2})
1	小驱动(I_{OL1}\ I_{OH1})

Bit [0] **IOALDS**: IOA3-0输出驱动电流选择

IOALDS	IOA[3:0]输出驱动电流选择
0	大驱动(I_{OL2}\ I_{OH2})
1	小驱动(I_{OL1}\ I_{OH1})

注：（1）IOB2 口驱动能力参考 I_{OL5}。
（2）IO 口的各个驱动能力参考电性参数表。

7 定时器 0/1(TC0/1)

7.1 概述

M8P625 TC0/TC1 为带有可设置 1:128 预分频器及周期寄存器的 8 位/16 位定时计数器，具有休眠状态下唤醒功能。

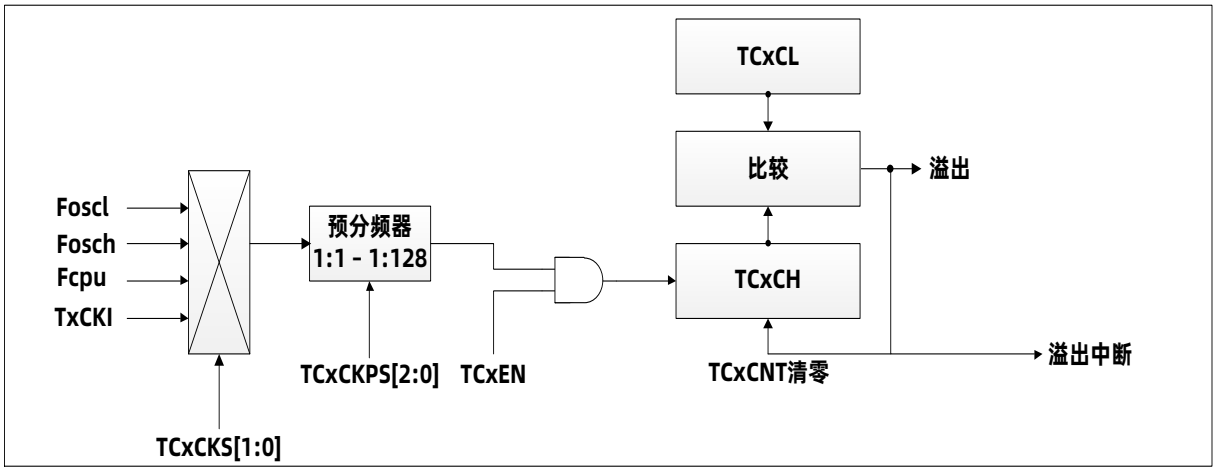
在 8 位模式下，TCxCL 作为 TCx 的周期寄存器器，TCx 使能后，TCxCH 递加，当 TCxCH 与 TCxCL 数值相等，TCx 溢出，将 TCxCH 清零重新开始计数，同时将中断标志位 TCxIF 置 1。

在 16 位模式下，[TCxCH,TCxCL]作为 16 位的计数器，TC0 使能后，16 位计数器递加，当计数值等于 0xFFFF 时，16 位计数器将清零重新开始计数，同时将中断标志位 TCxIF 置 1。

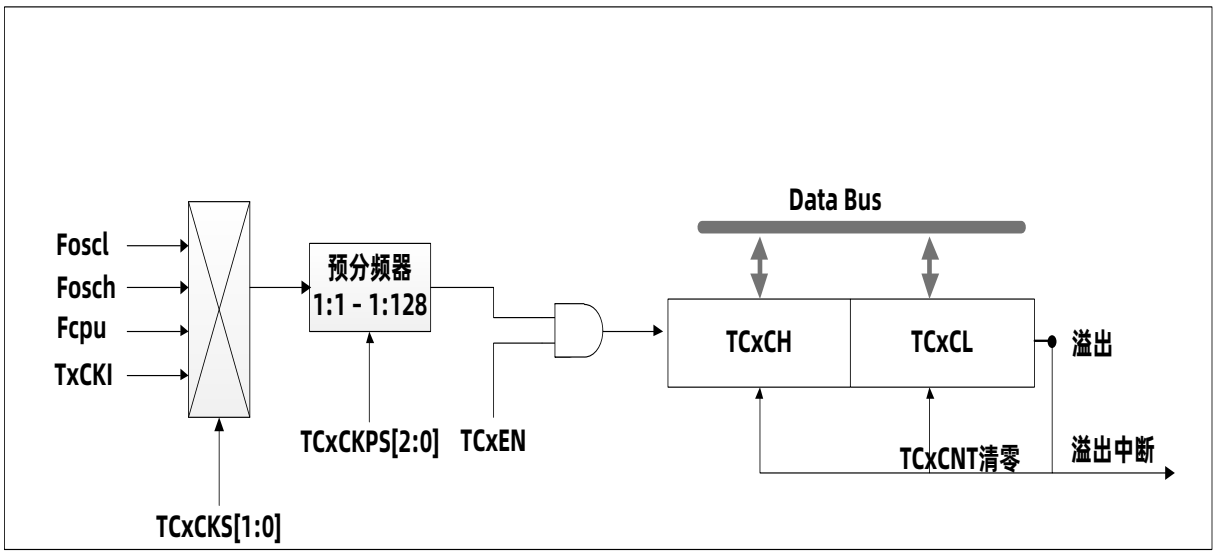
- 可选择时钟源：高频系统时钟 Fosch、低频系统时钟 Foscl 和指令时钟 Fcpu 、外部时钟 TxCKI
- 可选择 8 位模式和 16 位模式
  - 8 位模式下，通过设置周期寄存器，可任意设置 TC0 的周期
- 预分频比多级可选，最大可选择 1:128
- 溢出中断功能
- 溢出中断唤醒功能（当输入频率选择 Foscl 或 Fosch 时，若所选择的时钟源振荡器一直工作，此时 TC0 在休眠状态下依然工作，溢出中断可唤醒 CPU）

TC0/TC1 两种模式框图

8 位模式



16 位模式



7.2 TxCR 控制寄存器(x=0,1)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TxCR	TCxEN	TCxMOD	-	TCxCKS1	TCxCKS0	TCxCKPS2	TCxCKPS1	TCxCKPS0
读/写	R/W	R/W	-	R/W	R/W	R/W	R/W	R/W
复位后	0	0	-	0	0	0	0	0

Bit 7 **TCxEN**: TCx模块使能位

0 = 关闭TCx

1 = 使能TCx

Bit 6 **TCxMOD**: TCx模式选择位

0 = 8位模式

1 = 16位模式

Bit [4:3] **TCxCKS[1:0]**: TCx时钟源选择

TCxCKS[1:0]	TCx 时钟源选择
00	Foscl(低频系统时钟)
01	Fosch(高频系统时钟)
10	Fcpu
11	T0CKI（TC0）/T1CKI（TC1）

Bit [2:0] **TCxCKPS[2:0]**: TCx预分频比选择

TCxCKPS[2:0]	TCx 预分频比
000	1:1
001	1:2
010	1:4
011	1:8
100	1:16
101	1:32
110	1:64
111	1:128

7.3 TCxCL TCx 计数器低 8 位/周期寄存器(x=0,1)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TCxCL	TCxCL7	TCxCL6	TCxCL5	TCxCL4	TCxCL3	TCxCL2	TCxCL1	TCxCL0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	X	X	X	X	X	X	X	X

7.4 TCxCH TCx 计数器高位(x=0,1)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TCxCH	TCxCH7	TCxCH6	TCxCH5	TCxCH4	TCxCH3	TCxCH2	TCxCH1	TCxCH0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	X	X	X	X	X	X	X	X

7.5 定时器范例

例：以 TC0 为例，内置的 16MHz RC 震荡电路提供振荡频率，定时 125us 程序

```
;//+++++++++++++++++++++++++++++++++++++++++++++++++++++++
            ORG         0000H
            GOTO        Main_Program        ;//跳转到程序开始
            ORG         0008H
            GOTO        Interrupt           ;//发生中断后,跳转到中断子程序
;//+++++++++++++++++++++++++++++++++++++++++++++++++++++++
Main_Program:                               ;//程序开始
TC0_Init:                                   ;//TC0初始化
;//1、确保内置高频16M时钟在工作状态
            MOVIA       b'00000000'
            MOVAR       OSCM
;//2、配置T0CR控制寄存器
            MOVIA       b'00001100'         ;//8位模式,高频系统时钟源,1:16分频
            MOVAR       T0CR                ;//将模式位配置字写入T0CR
;//3、清零计数寄存器
            CLRR        TC0CH               ;//将TC0CH清零
;//4、配置自动加载计数器
            MOVIA       124                 ;//将124写入A
            MOVAR       TC0CL               ;//0.0625*16*(124+1)=125us
;//5、使能TC0,开启TC0中断
            BCLR        INTF0,TC0IF         ;//TC0 中断标志清零
            BSET        T0CR,7              ;//使能 TC0
            BSET        INTCR0, TC0IE       ;//使能 TC0 溢出中断
            BSET        OPTION, GIE         ;//开启总中断
Main:                                       ;//程序主循环
            ...
            GOTO        Main
```

```
;//+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Interrupt:                                  ;//中断子程序
            PUSH                            ;//压栈,保存 A,STATUS
;//中断处理程序
            JBTS0       INTF0, TC0IF        ;//检测 TC0 标志位
            GOTO        Interrupt_ TC0
Interrupt_End:
            POP                             ;//出栈,恢复 A,STATUS
            RETIE
Interrupt_ TC0:
            BCLR        INTF0,TC0IF         ;//清零 TC0 标志位
;//TC0 中断处理程序
            ...
            GOTO        Interrupt_End

            END
```

8 脉宽调制模块 PWM0

8.1 概述

M8P625 有一个 8 位分辨率 PWM0，时基使用 TC0。

8.2 PWM0CR 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM0CR	PWM0EN	PWM0OE	-	-	-	-	-	-
读/写	R/W	R/W	-	-	-	-	-	-
复位后	0	0	-	-	-	-	-	-

Bit 7 **PWM0EN**： PWM0模块使能位

0 = 关闭PWM0

1 = 使能PWM0

Bit 6 **PWM0OE**： PWM0输出控制

0 = PWM0信号不从管脚输出，管脚用做IO

1 = PWM0信号从管脚输出

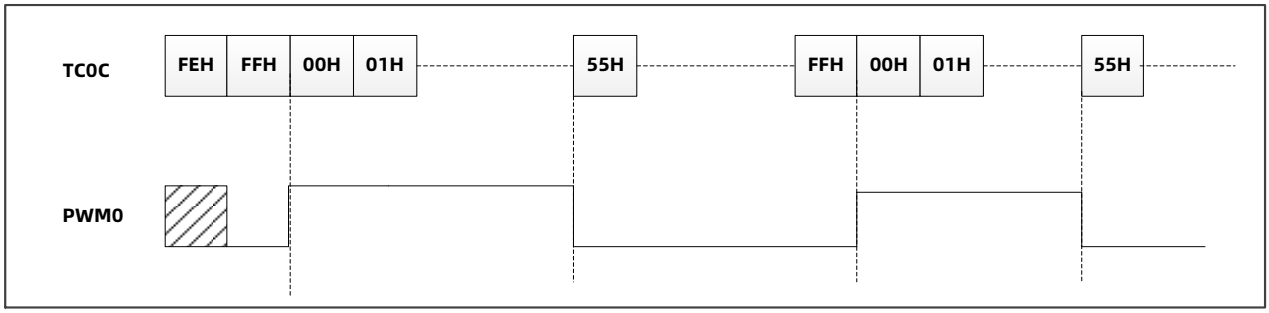
注：当不使用PWM模块时请保持bit6为0，不然会影响到对应IO端口的输出。

8.3 PWM0D 数据高位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM0D	PWM0D 7	PWM0D 6	PWM0D 5	PWM0D4	PWM0D 3	PWM0D 2	PWM0D 1	PWM0D 0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

8.4 PWM0 输出波形示例

例： PWM0CR=11000000B，PWM0D=55H，TC0CL=FFH



8.5 PWM0 范例

例: PWM0 以 TC0 为时基，内置的 16MHz RC 震荡电路提供振荡频率，设置 50%占空比，在 IOB3 口输出 1kHz 波形

```
;//+++++++++++++++++++++++++++++++++++++++++++++++++++++++
            ORG         0000H
            GOTO        Main_Program        ;//跳转到程序开始
            ORG         0008H
            GOTO        Interrupt           ;//发生中断后,跳转到中断子程序
;//+++++++++++++++++++++++++++++++++++++++++++++++++++++++
Main_Program:                               ;//程序开始
PWM0_Init:                                  ;//PWM0初始化
;//1、IO端口设置:
            BSET        OEB,3               ;//设置 IOB3 为波形输出端口
;//2、时钟源设置:
            MOVIA       b'00001110'         ;// 8位模式,高频系统时钟源,1:64分频
            MOVAR       T0CR                ;//将模式位配置字写入
            CLRR        TC0CH               ;//将TC0CH清零
            MOVIA       249                 ;//将249写入A
            MOVAR       TC0CL               ;//16MHz/ (249+1)/64=1KHz
;//3、PWM0设置:
            MOVIA       b'01000000'         ;//设置PWM0信号从管脚输出
            MOVAR       PWM0CR              ;//端口输出波形
            MOVIA       (249+1)/2           ;//产生占空比为50%的PWM波形
            MOVAR       PWM0D               ;//设置数据高位
;//4、使能PWM0和TC0:
            BSET        T0CR,7              ;//使能定时器T0
            BSET        PWM0CR,7            ;//使能PWM0
Main:                                       ;//程序主循环
            ...
            GOTO        Main
```

9 脉宽调制模块 PWM1

9.1 概述

M8P625 有 1 路带有死区控制的 PWM，时基使用 TC1，可独立进行设置，8+4 位分辨率。

- 正常 PWM 模式（定时器 1 需设置为 8 位模式）
- 单脉冲模式（定时器 1 需设置为 16 位模式）
- 互补输出
- 死区控制
- 12 位分辨率模式
- 8+4 分辨率模式
- 多个管脚输出

9.2 PWM1CR 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM1CR	PWM1EN	PWM1POE	PWM1NOE	PWM1PAS	PWM1NAS	-	PWM1M	PWM1T
读/写	R/W	R/W	R/W	R/W	R/W	-	R/W	R/W
复位后	0	0	0	0	0	-	0	0

Bit 7 **PWM1EN**: PWM1模块使能位
0 = 关闭PWM1
1 = 使能PWM1

Bit 6 **PWM1POE**: PWM1正相波形输出使能位
0 = 端口用作IO
1 = 端口输出PWM1P波形

Bit 5 **PWM1NOE**: PWM1反相波形输出使能位
0 = 端口用作IO
1 = 端口输出PWM1N波形

Bit 4 **PWMxPAS**: PWM1P波形有效电平选择
0 = 端口输出PWM1P波形高电平有效
1 = 端口输出PWM1P波形低电平有效

Bit 3 **PWM1NAS**: PWM1N波形有效电平选择
0 = 端口输出PWM1N波形低电平有效
1 = 端口输出PWM1N波形高电平有效

Bit 1 **PWM1M**: PWM1模式选择
0 = 正常PWM模式（需将定时器1设置成8为模式）
1 = 单脉冲模式（需将定时器1设置成16位模式）

Bit 0 **PWM1T**: PWM1触发事件设置
0 = T1CKI上升沿
1 = T1CKI上升沿/下降沿

注：当不使用PWM模块时请保持bit5，6为0，不然会影响到对应IO端口的输出。

9.3 PWM1S 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM1S	-	-	-	-	-	PWM1C	PWM1B	PWM1A
读/写	-	-	-	-	-	R/W	R/W	R/W
复位后	-	-	-	-	-	0	1	0

Bit 2 **PWM1C**: PWM1C 输出管脚设置
0 = PWM1PC/PWM1NC 用作IO口
1 = PWM1PC/PWM1NC 用作PWM输出

Bit 1 **PWM1B**: PWM1B 输出管脚设置
0 = PWM1PB/PWM1NB 用作IO口
1 = PWM1PB/PWM1NB 用作PWM输出

Bit 0 **PWM1A**: PWM1A 输出管脚设置
0 = PWM1PA/PWM1NA 用作IO口
1 = PWM1PA/PWM1NA 用作PWM输出

9.4 PWM1DH 数据高位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM1DH	PWM1DH7	PWM1DH6	PWM1DH5	PWM1DH4	PWM1DH3	PWM1DH2	PWM1DH1	PWM1DH0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

9.5 PWM1DL 数据低位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM1DL	PWM1DL7	PWM1DL6	PWM1DL5	PWM1DL4	-	-	-	-
读/写	R/W	R/W	R/W	R/W	-	-	-	-
复位后	0	0	0	0	-	-	-	-

9.6 PWMDEADT PWM 死区控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWMDEADT	DEADTF3	DEADTF2	DEADTF1	DEADTF0	DEADTR3	DEADTR2	DEADTR1	DEADTR0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

Bit [7:4] **DEADTF[3:0]**: 前死区宽度设置

DEADTF[3:0]	前死区时间设定
0000	无前死区
0001	前死区时间为 1*(时基时钟周期/2)
0011	前死区时间为 2*(时基时钟周期/2)
0010	前死区时间为 3*(时基时钟周期/2)
1010	前死区时间为 4*(时基时钟周期/2)
1011	前死区时间为 5*(时基时钟周期/2)
1001	前死区时间为 6*(时基时钟周期/2)
1101	前死区时间为 7*(时基时钟周期/2)
0101	前死区时间为 8*(时基时钟周期/2)
0100	前死区时间为 9*(时基时钟周期/2)
0110	前死区时间为 10*(时基时钟周期/2)
0111	前死区时间为 11*(时基时钟周期/2)
1111	前死区时间为 12*(时基时钟周期/2)
1110	前死区时间为 13*(时基时钟周期/2)
1100	前死区时间为 14*(时基时钟周期/2)
1000	前死区时间为 15*(时基时钟周期/2)

Bit [3:0] **DEADTR[3:0]**: 后死区宽度设置

DEADTR[3:0]	后死区时间设定
0000	无后死区
0001	后死区时间为 1*(时基时钟周期/2)
0011	后死区时间为 2*(时基时钟周期/2)
0010	后死区时间为 3*(时基时钟周期/2)
1010	后死区时间为 4*(时基时钟周期/2)
1011	后死区时间为 5*(时基时钟周期/2)
1001	后死区时间为 6*(时基时钟周期/2)
1101	后死区时间为 7*(时基时钟周期/2)
0101	后死区时间为 8*(时基时钟周期/2)
0100	后死区时间为 9*(时基时钟周期/2)
0110	后死区时间为 10*(时基时钟周期/2)
0111	后死区时间为 11*(时基时钟周期/2)
1111	后死区时间为 12*(时基时钟周期/2)
1110	后死区时间为 13*(时基时钟周期/2)
1100	后死区时间为 14*(时基时钟周期/2)
1000	后死区时间为 15*(时基时钟周期/2)

死区时间设置：

Tdeadr = DEADTR * 时基时钟周期

Tdeadf = DEADTF * 时基时钟周期

注：（1）时基时钟周期即各 PWM 所选择的时钟源经预分频之后的时钟周期。
（2）3 路 PWM 共用同一档前/后死区宽度设置寄存器，但当每路 PWM 选择不同的时基时，死区宽度计算是对应不同的时基时钟周期。

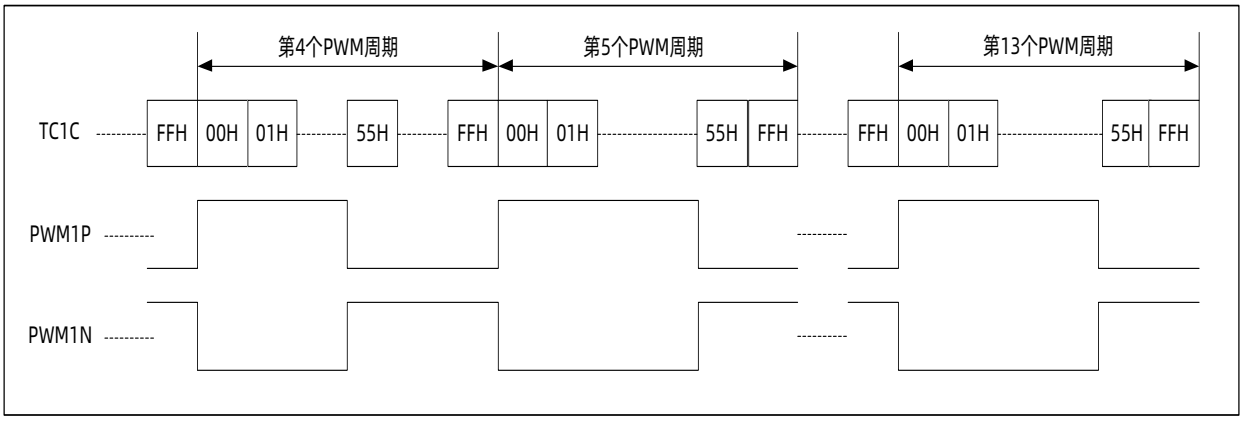
9.7 8+4 位分辨率模式

PWM1D[3:0]为 4 位扩展位，PWMD[11:4]决定 PWM 脉冲基础宽度。在每 16 个 PWM 周期循环中，扩展位中的有效位对应的 PWM 周期，输出的 PWM 脉冲宽度为（PWMD[11:4]+1），而其余的 PWM 周期，输出的 PWM 脉冲宽度为（PWMD[11:4]），这样得到的 PWM 输出是等效的 12 位 PWM 分辨率效果。

PWM1D[3:0]对应的扩展周期序号：

PWM1D[3:0]	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
PWM1D3		●		●		●		●		●		●		●		●
PWM1D2			●				●				●				●	
PWM1D1					●								●			
PWM1D0									●							

例： PWM1CR=11100000B，PWM1DH=55H，PWM1DL=02H，TC0CL=FFH



9.8 单脉冲模式

单脉冲模式下，当 T1CKI 输入 n 个事件（上升沿或下降沿），PWM1DH、PWM1DL 作为 PWM 脉冲宽度。

脉冲宽度设置：

PulsWidth = 62.5ns*定时器 1 的预分频比*PWM1D

若定时器 1 的分频比为 1:16，PWM1DH=50，PWM1DL=0，将 PWM1D 数据高低位转换为二进制，则 PWM1D=1100100000，转化为十进制 PWM1D=800，由脉宽计算公式可得：PulsWidth = 62.5ns*16*800=800us；若设置分频比为 1:16，PWM1D(max)=4096，可设置脉宽的最大宽度为 4096us(约为 4ms)。

滤波时间设置：

滤波时间=62.5ns*定时器 1 的预分频比*PWMDEADT[2:0]

PWMDEADT[2:0]作为触发信号的滤波时间设置，触发信号的采样时钟为定时器的计数时钟；若定时器 1 的分频比为 1:16，PWMDEADT[7:0]设置为”00000010”，滤波时间=62.5ns*16*2=2us；若设置分频比为 1:16，PWMDEADT[7:0]设置为”00000111”，设置的最大滤波时间=62.5ns *16*8=8us。

PWMDEADT[7:3]作为触发事件设置，高可选 16 个触发事件中产生的脉冲个数。

PWMDEADT[7:3]	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
PWMDEADT[7]	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
PWMDEADT[6]		●		●		●		●		●		●		●		●
PWMDEADT[5]			●				●				●				●	
PWMDEADT[4]					●								●			
PWMDEADT[3]									●							

范例：单脉冲模式下，以 PWMDEADT3 为例，TC1 时基

```
;// ++++++++++++++++++++++++++++++++++++++++++++++++++++++
            ORG         0000H
            GOTO        Main_Program        ;//跳转到程序开始
            ORG         0008H
            GOTO        Interrupt           ;//发生中断后,跳转到中断子程序
;//++++++++++++++++++++++++++++++++++++++++++++++++++++++
Main_Program:                               ;//程序开始
PWM1_Init:                                  ;//PWM1初始化
;//1、确保内置高频16M时钟在工作状态
            MOVIA   b'00000000'             ;//设置系统时钟为高频16M时钟
            MOVAR       OSCM
;//2、IO 端口设置
            MOVIA   0x7F                    ;//T1CKI 口作外接信号输入
            MOVAR       OEA
            MOVIA       0x03                ;//PWM1NA/PWM1PA 口输出 PWM1 波形
            MOVAR       OEA
            ......
```

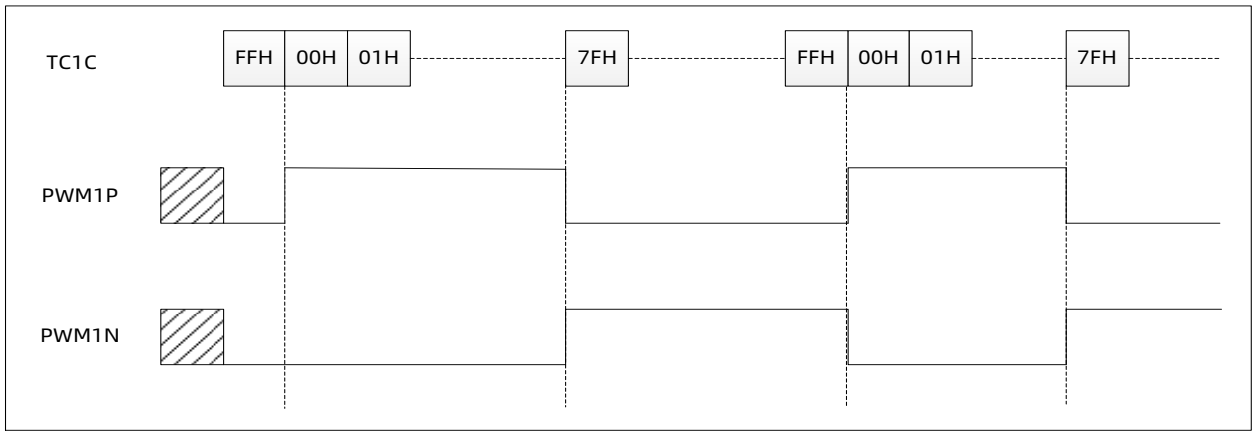
```
//3、时钟源设置(以TC1为例,使其产生1KHz的波形)
            MOVIA       0xFF
            MOVAR       TC1CH
            MOVIA       0xFF
            MOVAR       TC1CL
            MOVIA       B'11001100'          ;//定时器使能,16 位模式,高频系统时钟源,1:16 分频
            MOVAR       T1CR                 ;//定时器 T1
;//4、PWM设置
            MOVIA       b'11000011'          ;//PWM 时基选择定时器 T1
            MOVAR       PWM1CR               ;//PWM1 使能,端口输出波形
            MOVIA       b'00001111'          ;//一个脉冲;滤波时间8us
            MOVAR       PWM1DEADT
            MOVIA       b'00000001'          ;// PWM1PA/PWM1NA  用作PWM输出
            MOVAR       PWM1S
            MOVIA       50
            MOVAR       PWM1DH
            MOVIA       00
            MOVAR       PWM1DL
Main:                                        ;//程序主循环
            ...
            GOTO         Main
;//++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Interrupt:                                   ;//中断子程序
            PUSH                             ;//压栈,保存 A,STATUS
;//中断处理程序
            NOP
Interrupt_End:
            POP                              ;//出栈,恢复 A,STATUS
            RETIE

            END
```

注：单脉冲模式下，设置 TC1 定时器时，时钟源只能选用 Fosch，若选用 Foscl、Fcpu 或 T1CKI 作为时钟源，则无法产生相对应的现象。

9.9 PWM 输出波形示例

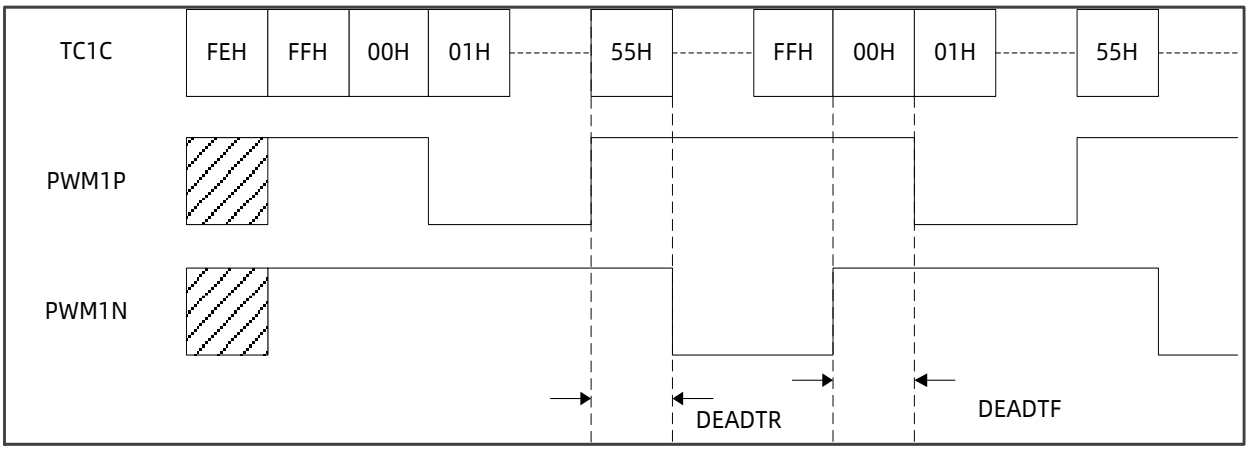
9.9.1 互补 PWM 输出

例： PWM1CR=11100000B，PWM1DH=00H，PWM1DL=7FH，TC1CL=FFH



9.9.2 带死区的互补 PWM 输出

例：PWM1CR=11111100B，PWM1DH=00H，PWM1DL=55H，TC1CL=FFH，PWM1DEADT=00010001B



9.10 PWM1 范例

例：以 TC1 时基，内置的 16MHz RC 震荡电路提供振荡频率，设置 50%占空比，在 IOA0 口输出 1kHz 波形

```
;// ++++++++++++++++++++++++++++++++++++++++++++++++++++++
            ORG         0000H
            GOTO        Main_Program    ;//跳转到程序开始
            ORG         0008H
            GOTO        Interrupt       ;//发生中断后,跳转到中断子程序
;//++++++++++++++++++++++++++++++++++++++++++++++++++++++
Main_Program:                           ;//程序开始
PWM1_Init:                              ;//PWM1初始化
;//1、IO 端口设置:
            BSET        OEA,0           ;//设置 IOA0 为波形输出口
;//2、时钟源设置:
            MOVIA       b'00001110'     ;//8位模式,高频系统时钟,1:64分频
            MOVAR       T1CR            ;//将模式位配置字写入
            CLRR        TC1CH           ;//将 TC0CH 清零
            MOVIA       249             ;//将249写入A
            MOVAR       TC1CL           ;//16MHz/64/(249+1)=1KHz
;//3、PWM1设置:
            MOVIA       b'01111000'
            MOVAR       PWM1CR          ;//端口输出PWM1P波形、PWM1N波形,高低电平有效
            MOVIA       b'000000001'    ;//PWM1PA/PWM1NA 用作PWM输出
            MOVAR       PWM1S
            CLRR        PWM1DL          ;//低位清零
            MOVIA       (249+1)/2       ;//产生占空比为50%的PWM波形
            MOVAR       PWM1DH
;//4、使能PWM1、TC1:
            BSET        T1CR,7          ;//使能定时器T0
            BSET        PWM1CR,7        ;//使能PWM0
Main:                                   ;//程序主循环
            ...
            GOTO        Main
```

10 通用串行通讯口（USART）

10.1 概述

M8P625 支持异步全双工模式和同步半双工模式。

注：使用UART时，注意内部高频振荡器的电压特性曲线，建议使用和实际应用电压相同或接近的烧录电压进行烧录。

10.2 TXCR 发送控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TXCR	TXEN	TMCLR	SYNC	TX9	SLAVE	SPD1	SPD0	TXD9
读/写	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	1	0	0	0	0	0	0

Bit 7 **TXEN**： 使能发送
0 = 屏蔽USART发送功能
1 = 使能USART发送功能

Bit 6 **TMCLR**： 发送寄存器空标志
0 = 正在发送数据，移位寄存器不空
1 = 数据已发送，移位寄存器空

Bit 5 **SYNC**： 同步模式
0 = 选择异步模式
1 = 选择同步模式

Bit 4 **TX9**： 数据长度选择
0 = 8位数据
1 = 9位数据

Bit 3 **SLAVE**： 同步发送/接收模式
0 = Master
1 = SLAVE

Bit [2:1] **SPD[1:0]**： 发送接收速度选择

SPD[1:0]	波特率分频比(n)
00	4
01	16
10	64
11	256

Bit 0 **TXD9**： 发送数据第9位数据

10.3 TXREG 发送数据寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TXREG	TX0D7	TX0D6	TX0D5	TX0D4	TX0D3	TX0D2	TX0D1	TX0D0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	X	X	X	X	X	X	X	X

10.4 RXCR 接收控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
RXCR	RXEN	CKPS	-	RX9	SREN	RXOVF	FRER	RXD9
读/写	R/W	R/W	-	R/W	R/W	R	R	R
复位后	0	0	-	0	0	0	0	0

Bit 7 **RXEN**： 使能接收
0 = 屏蔽USART接收功能
1 = 使能USART接收功能

Bit 6 **CKPS**：同步模式时钟模式选择
0 = 上升沿发送数据
1 = 下降沿发送数据

Bit 4 **RX9**： 数据长度选择
0 = 8位数据
1 = 9位数据

Bit 3 **SREN**：同步接收开始
0 = 停止同步接收
1 = 开始同步接收，单字节接收模式下接收完一个字节自动清零

Bit 2 **RXOVF**： 接受缓冲区溢出标志
0 = 接收缓冲区未发生溢出
1 = 接收缓冲区溢出，读缓冲区自动清零

Bit 1 **FRER**： 接收数据格式错
0 = 当前接收数据未发生格式错
1 = 当前接收数据格式错（未收到停止位）

Bit 0 **RXD9**： 接收数据第9位数据

10.5 RXREG 接收数据寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
RXREG	RX0D7	RX0D6	RX0D5	RX0D4	RX0D3	RX0D2	RX0D1	RX0D0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	X	X	X	X	X	X	X	X

10.6 BRGDH 波特率寄存器高位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BRGDH	SBYTE	-	-	-	-	-	BRGD9	BRGD8
读/写	R/W	-	-	-	-	-	R/W	R/W
复位后	0	-	-	-	-	-	0	0

Bit 7 **SBYTE**: 同步接收模式选择

0 = 多字节接收

1 = 单字节接收，接收完一个字节后自动清除SREN

10.7 BRGDL 波特率寄存器低位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BRGDL	BRGD7	BRGD6	BRGD5	BRGD4	BRGD3	BRGD2	BRGD1	BRGD0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

10.8 USART 使用说明

10.8.1 波特率设置

通过设置 BRGD 和 SPD 来获得所需的波特率。

波特率计算公式：目标波特率 = Fosc/((BRGD+1)*n)。

常用波特率设置（Fosc=16MHz）

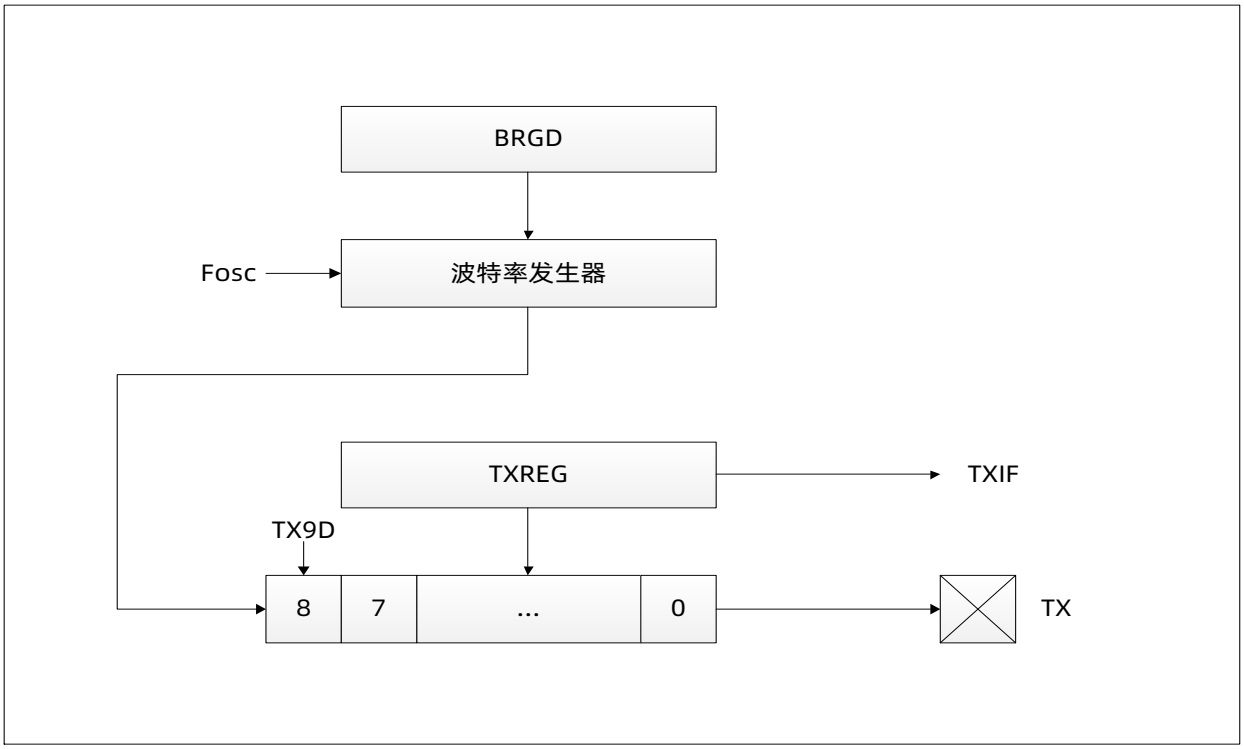
常用波特率	N（波特率分频比）	BRGD	偏差
300	256	0x0CF	0.16%
600	256	0x067	0.16%
1200	64	0x0CF	0.16%
2400	64	0x067	0.16%
4800	16	0x0CF	0.16%
9600	16	0x067	0.16%
19200	4	0x0CF	0.16%
38400	4	0x067	0.16%
57600	4	0x044	0.64%
115200	4	0x022	-0.79%

10.8.2 异步发送

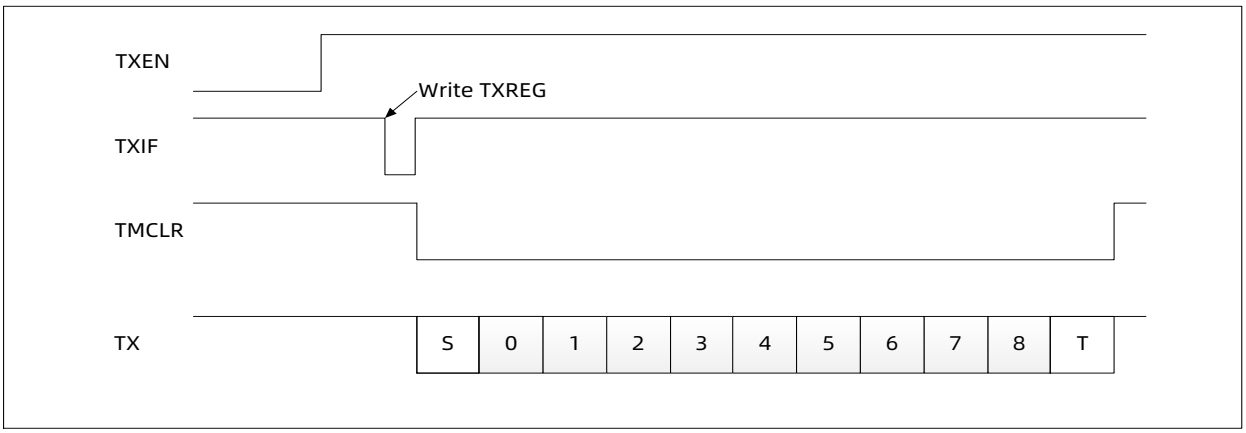
当 TXEN 使能时，TXIF 中断标志为 1 说明 TXREG 发送寄存器为空，TMCLR=1 说明发送移位寄存器为空，发送器处于空闲状态。

空闲状态时写入 TXREG，写入数据将立即装载到发送移位寄存器中，此时，TXIF 为 0，TMCLR=0，发送器进入发送状态。此时再次写入 TXREG，TXIF 将清零，说明 TXREG 有未发送数据，发送移位寄存器发送完毕后，TXREG 数据将自动载入发送移位寄存器继续发送，且 TXIF 为空。

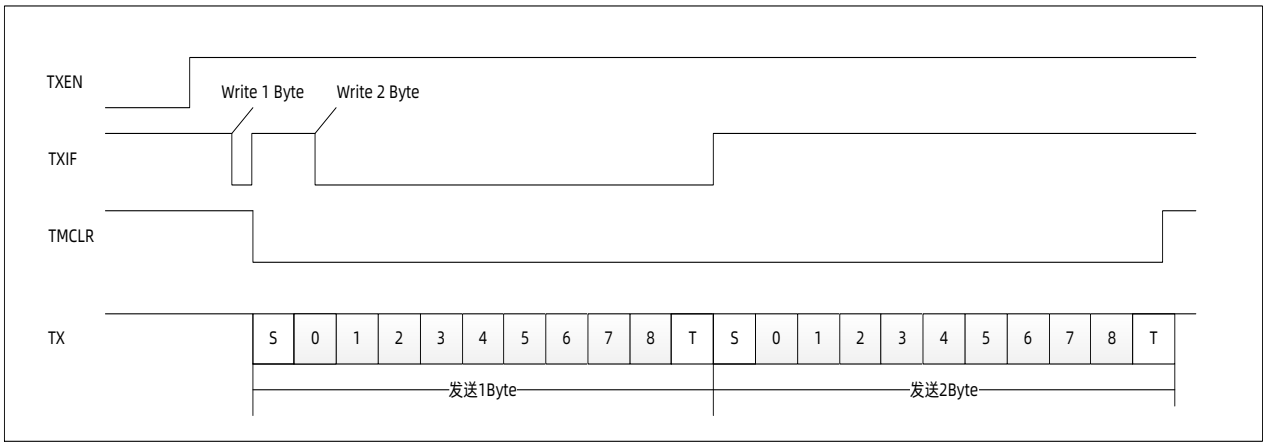
当 TXIF 为 0 时写入 TX0REG，将覆盖上次写入数据。



单字节发送：



多字节发送：



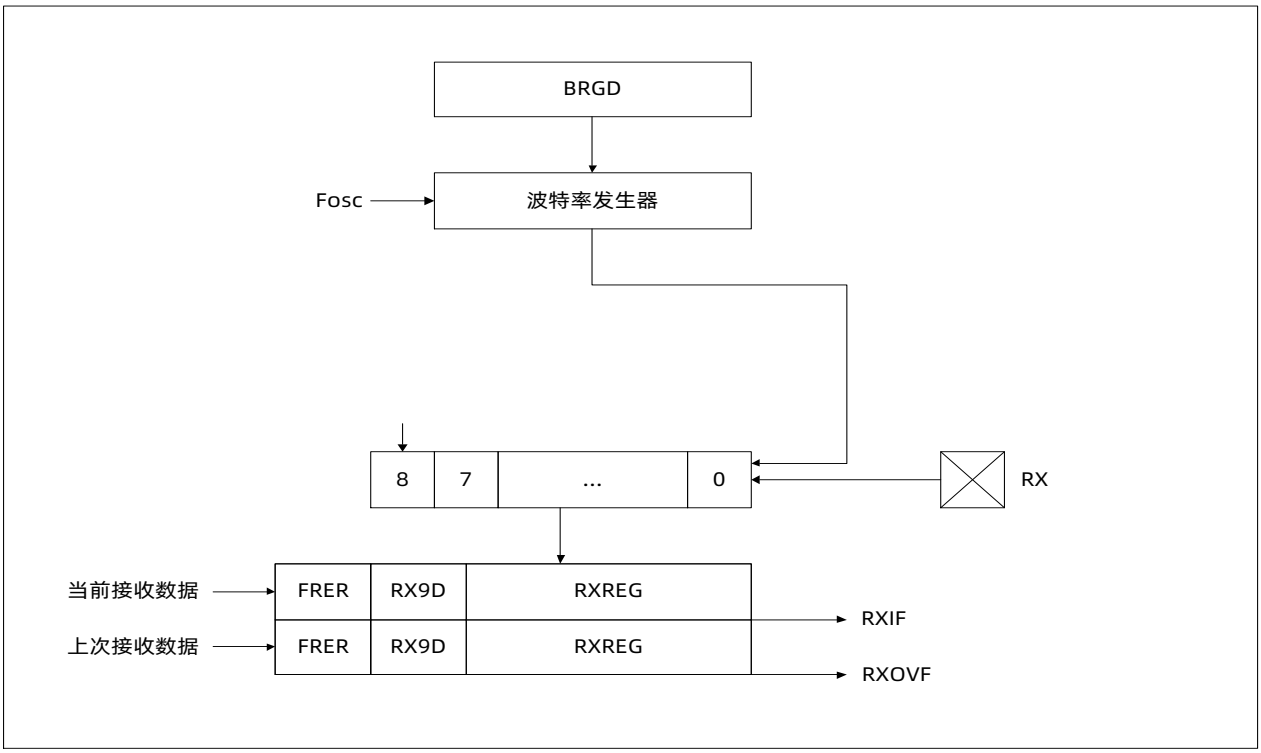
STEP1：设置波特率 SSPD=X，BRGD=X，SYNC=0
STEP2：设置 TXEN=1，设置数据模式 TX9=X
STEP3：写入数据高位 TXD9
STEP4：写入 TXREG，启动发送
STEP5：当 TMCLR =1 时，写入 TXREG 发送下一个字节
STEP6：重复 STEP5，直到该帧数据发送完成

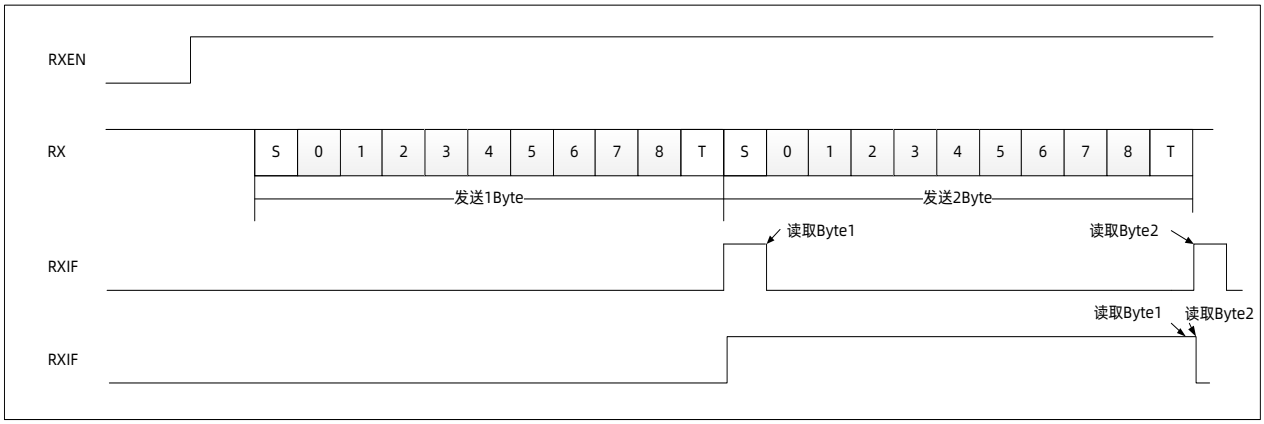
例: USART 异步发送

```
;// ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
            ORG         0000H
            GOTO        Main_Program    ;//跳转到程序开始
            ORG         0008H
            GOTO        Interrupt       ;//发生中断后,跳转到中断子程序
;//++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Main_Program:                           ;//程序开始
USART_Init:                             ;//USART初始化
;//1、端口设置
            BSET        OEB,6           ;//0:输入 1:输出
;//2、TXCR 设置:
            MOVIA       b'10000000'
            MOVAR       TXCR            ;//使能USART发送功能
;//3、波特率设置:                       ;//目标波特率 = Fosc/((BRGD+1)*n)
            MOVIA       0x00
            MOVAR       BRGDH           ;//波特率寄存器高位清零
            MOVIA       0x67
            MOVAR       BRGDL           ;//波特率 38400
;//4、数据发送
            MOVIA       0x55            ;//以发送数据55为例
            MOVAR       TXREG
            JBTS1       TXCR, 6
            GOTO        $-1
Main:                                   ;//程序主循环
            ...
            GOTO        Main
;//++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Interrupt:                              ;//中断子程序
            PUSH                        ;//压栈,保存 A,STATUS
;//中断处理程序
            NOP
Interrupt_End:
            POP                         ;//出栈,恢复 A,STATUS
            RETIE

            END
```

10.8.3 异步接收

设置异步模式，使能 RXEN，开始启动异步接收。RX 管脚处于高电平时，接收器处于空闲状态，当检测到 RX 变为低电平，接收器检测该低电平是否有效起始位，若为有效起始位，则启动数据时钟恢复电路和数据恢复电路进行接收。1 个数据接收完成后，RXIF 置 1，当接收 3 个数据未读取，RXOVF 置 1，同时舍弃第三个接收数据。完全读取 RXREG 后 RXIF 自动清零。





STEP1：设置波特率 SSPD=X，BRGD=X，SYNC=0
STEP2：设置 RXEN=1
STEP3：等待接收完成 RXIF=1
STEP4：判断 FRER=0，若为 1，帧格式错误，舍弃数据
STEP5：读取 RX9D
STEP6：读取 RXREG，重复 3-6

例: USART 异步接收

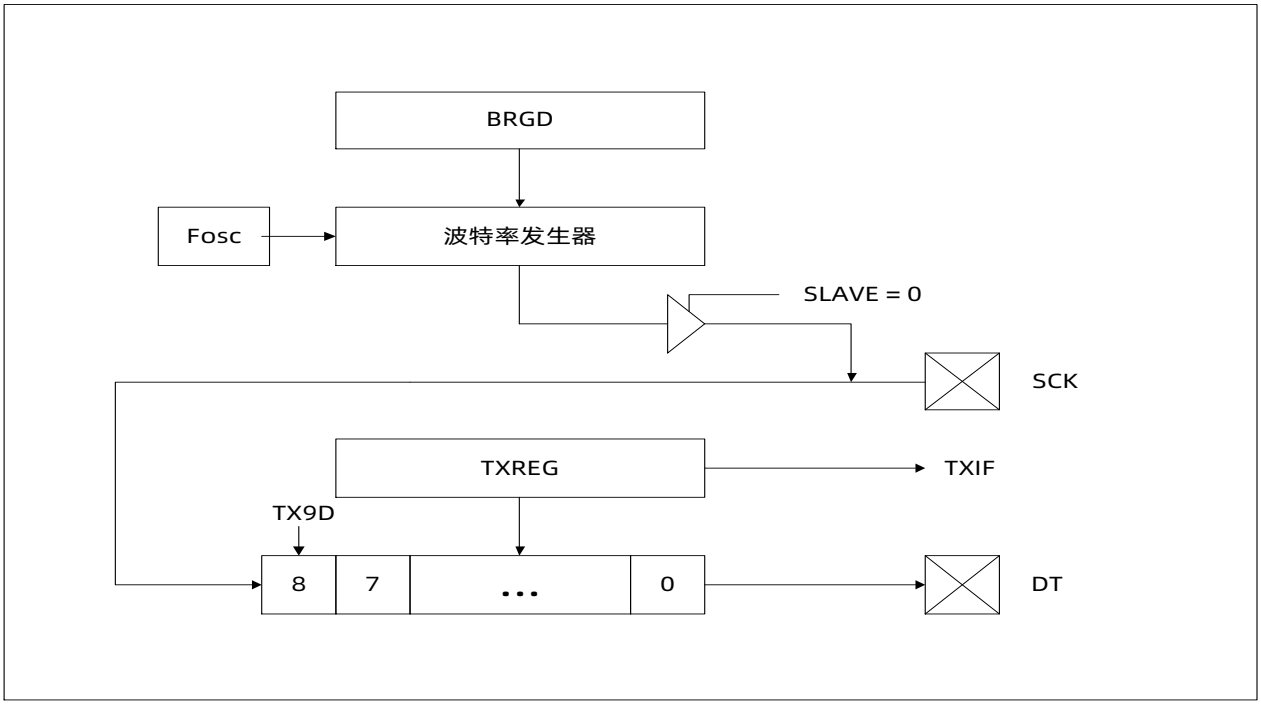
```
USART_DATA      EQU    ‘00’              ;//特殊寄存器定义声明
;// +++++++++++++++++++++++++++++++++++++++++++++++++++++++
                ORG          0000H
                GOTO         Main_Program   ;//跳转到程序开始
                ORG          0008H
                GOTO         Interrupt      ;//发生中断后,跳转到中断子程序
;//+++++++++++++++++++++++++++++++++++++++++++++++++++++++
Main_Program:                               ;//程序开始
USART_Init:                                 ;//USART初始化
;//1、端口设置
                BCLR         OEB,7          ;//0:输入  1:输出
;//2、TXCR 设置:
                MOVIA        b'00000000'
                MOVAR        TXCR           ;//屏蔽USART发送功能
;//3、RXCR 设置:
                MOVIA        b'10000000'
                MOVAR        RXCR           ;//使能USART接收功能
;//4、波特率设置:                            ;//目标波特率 = Fosc/((BRGD+1)*n)
                MOVIA        0x00
                MOVAR        BRGDH          ;//波特率寄存器高位清零
                MOVIA        0x67
                MOVAR        BRGDL          ;//波特率  38400
;//5、中断设置:
                BCLR         INTF0,5
                BSET         INTCR0,5       ;//开启 USRAT 接收中断
                BSET         OPTION,7       ;//开启总中断
Main:                                       ;//程序主循环
                ...
                GOTO          Main
;//+++++++++++++++++++++++++++++++++++++++++++++++++++++++
Interrupt:                                  ;//中断子程序
                PUSH                        ;//压栈,保存 A,STATUS
;//中断处理程序
                JBTS1        INTF0,5
                GOTO         Interrupt_End:
                MOVR         RXREG,A
                MOVAR        USART_DATA     ;//读取 RXREG
Interrupt_End:
                POP                         ;//出栈,恢复 A,STATUS
                RETIE

                END
```

10.8.4 同步发送

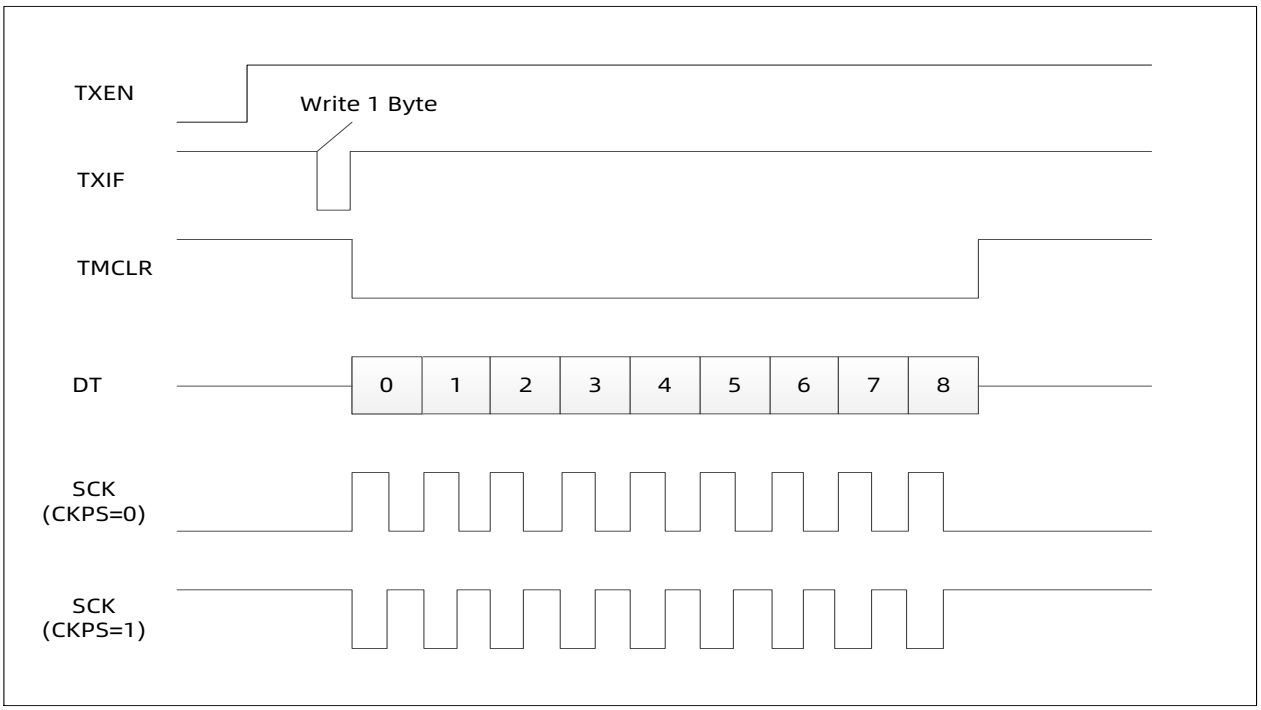
当 TXEN=1，SYNC=1 时，使能同步发送功能。 CKPS 选择发送时钟极性， TXIF 中断标志为 1 说明 TXREG 发送寄存器为空，TMCLR=1 说明发送移位寄存器为空，发送器处于空闲状态。

空闲状态写入 TXREG，写入数据将立即装载到发送移位寄存器中，此时，TXIF 为 1，TMCLR=0，发送器进入发送状态。此时再次写入 TXREG，TXIF 将清零，说明 TXREG 有未发送数据，发送移位寄存器发送完毕后，TXREG 数据将自动载入发送移位寄存器继续发送，且 TXIF 为空。

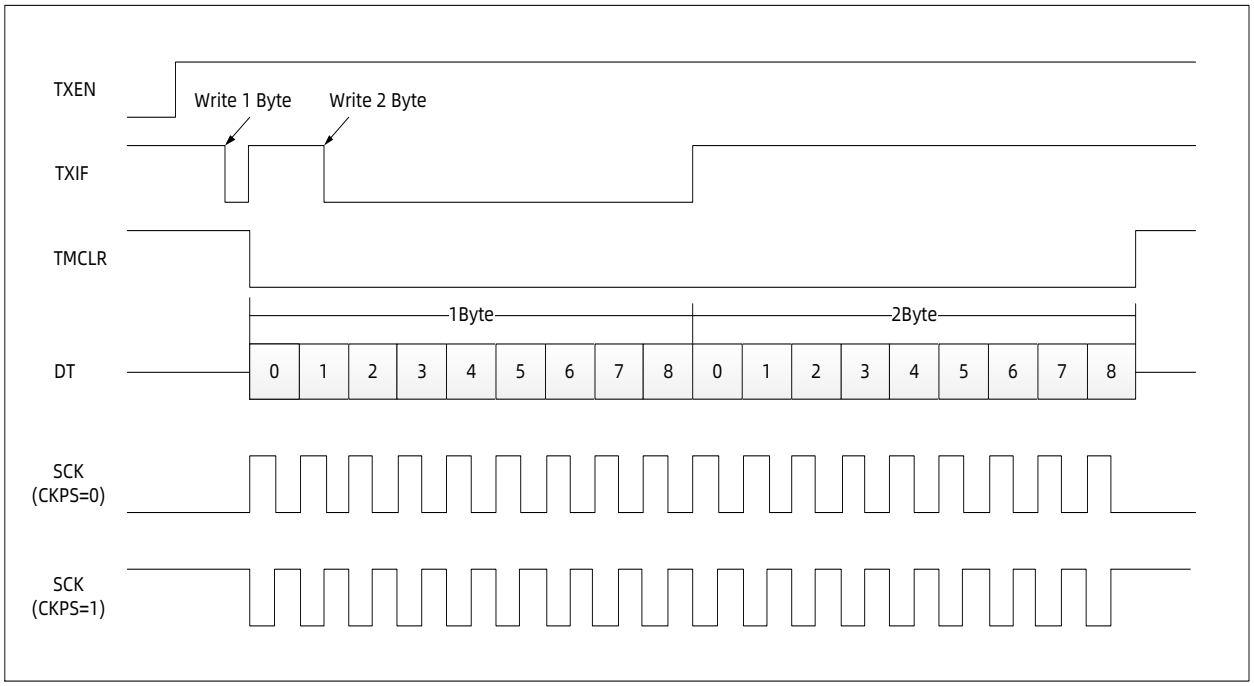
当 TXIF 为 0 时写入 TX0REG，将覆盖上次写入数据。



单字节发送：



多字节发送：



参考操作步骤 SLAVE=0：

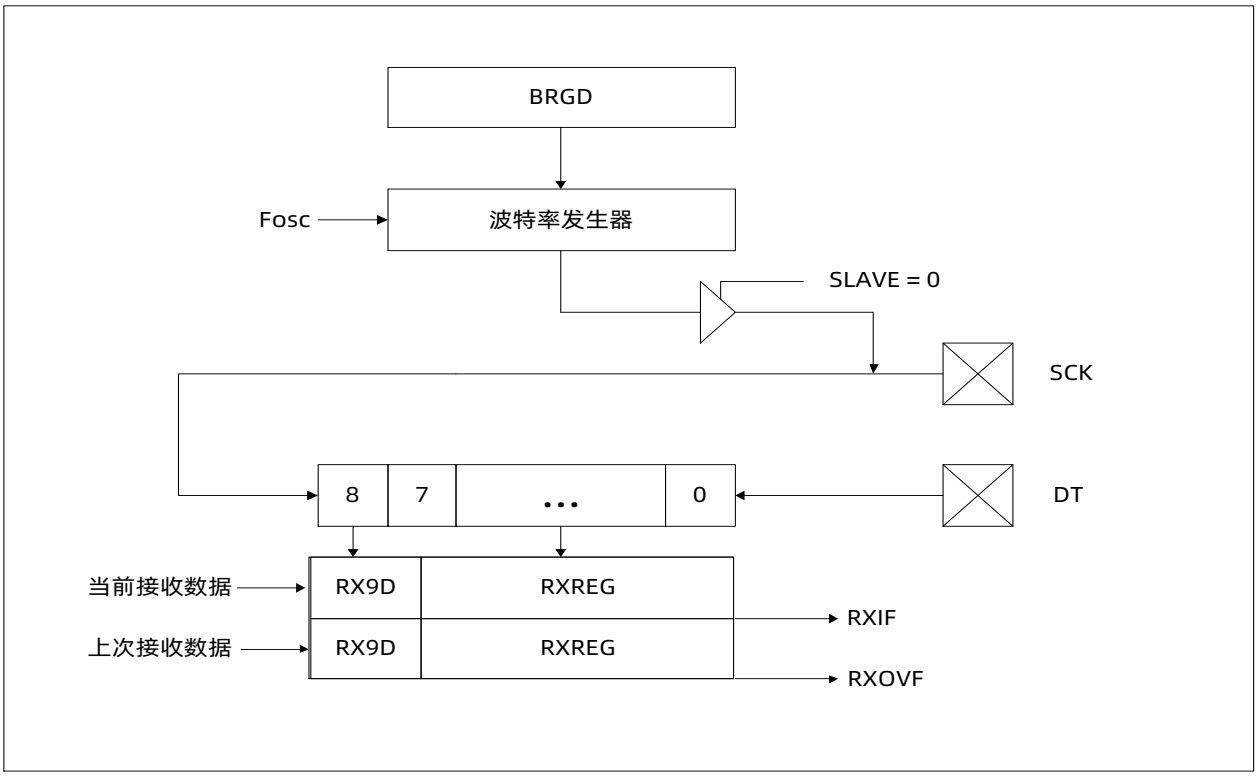
STEP1：设置波特率 SSPD=X，BRGD=X，SYNC=1
STEP2：设置 TXEN=1，设置数据模式 TX9=X
STEP3：写入数据高位 TXD9
STEP4：写入 TXREG，启动发送
STEP5：当 TXIF=1 时，写入 TXREG 发送下一个字节
STEP6：重复 STEP5，直到该帧数据发送完成

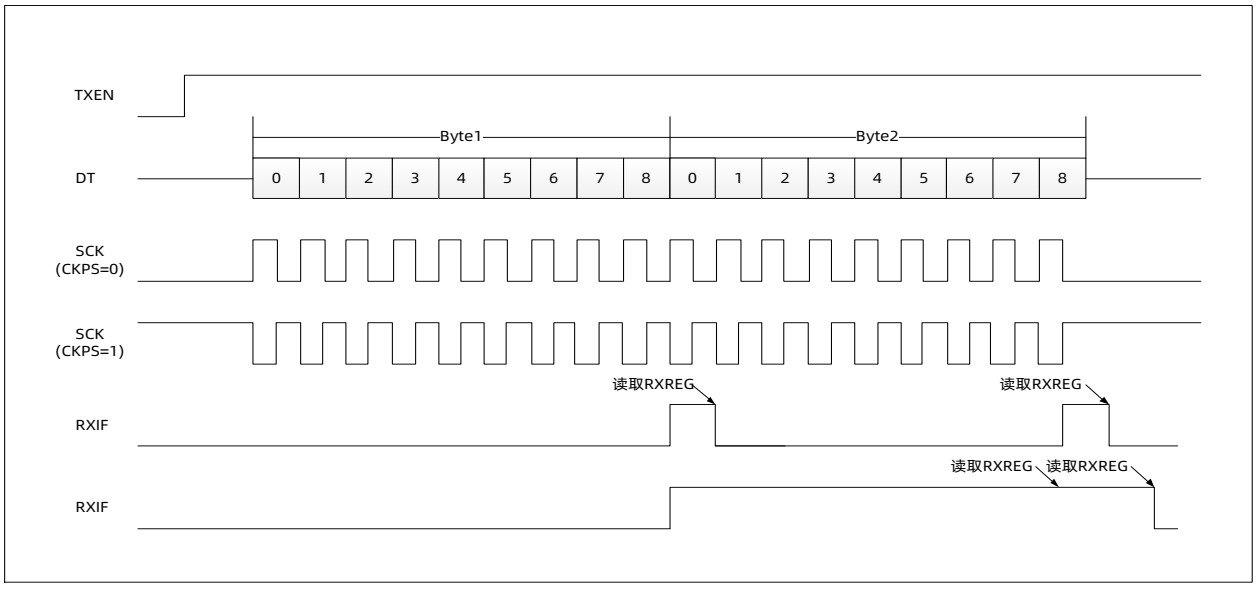
参考操作步骤 SLAVE=1：

STEP1：设置波特率 SSPD=X，BRGD=X，SYNC=1
STEP2：设置 TXEN=1，设置数据模式 TX9=X
STEP3：当 TXIF=1 时，写入数据高位 TXD9
STEP：写入 TXREG 等待发送下一个字节
STEP5：重复 STEP3-4，直到该帧数据发送完成

10.8.5 同步接收

设置同步 SYNC=1 模式，使能 RXEN，开始启动异步接收。RX 管脚处于高电平时，接收器处于空闲状态，当检测到 RX 变为低电平，接收器检测该低电平是否有效起始位，若为有效起始位，则启动数据时钟恢复电路和数据恢复电路进行接收。1 个数据接收完成后，RXIF 置 1，当接收 3 个数据未读取，RXOVF 置 1，同时舍弃第三个接收数据。完全读取 RXREG 后 RXIF 自动清零。





参考操作步骤 SLAVE=0：

STEP1：设置波特率 SSPD=X，BRGD=X，SYNC=0
STEP2：设置 RXEN=1
STEP3：写 SREN 启动接收
STEP4：等待接收完成 RXIF=1
STEP5：读取 RX9D
STEP6：读取 RXREG，单字节接收（SBYTE=1）重复 3-6；多字节接收（SBYTE=0）重复 4-6

参考操作步骤 SLAVE=1：

STEP1：设置波特率 SSPD=X，BRGD=X，SYNC=0
STEP2：设置 RXEN=1
STEP3：写 SREN 启动接收
STEP4：等待接收完成 RXIF=1
STEP5：读取 RX9D
STEP6：读取 RXREG，单字节接收（SBYTE=1）重复 3-6；多字节接收（SBYTE=0）重复 4-6

10.8.6 唤醒及休眠模式下通讯

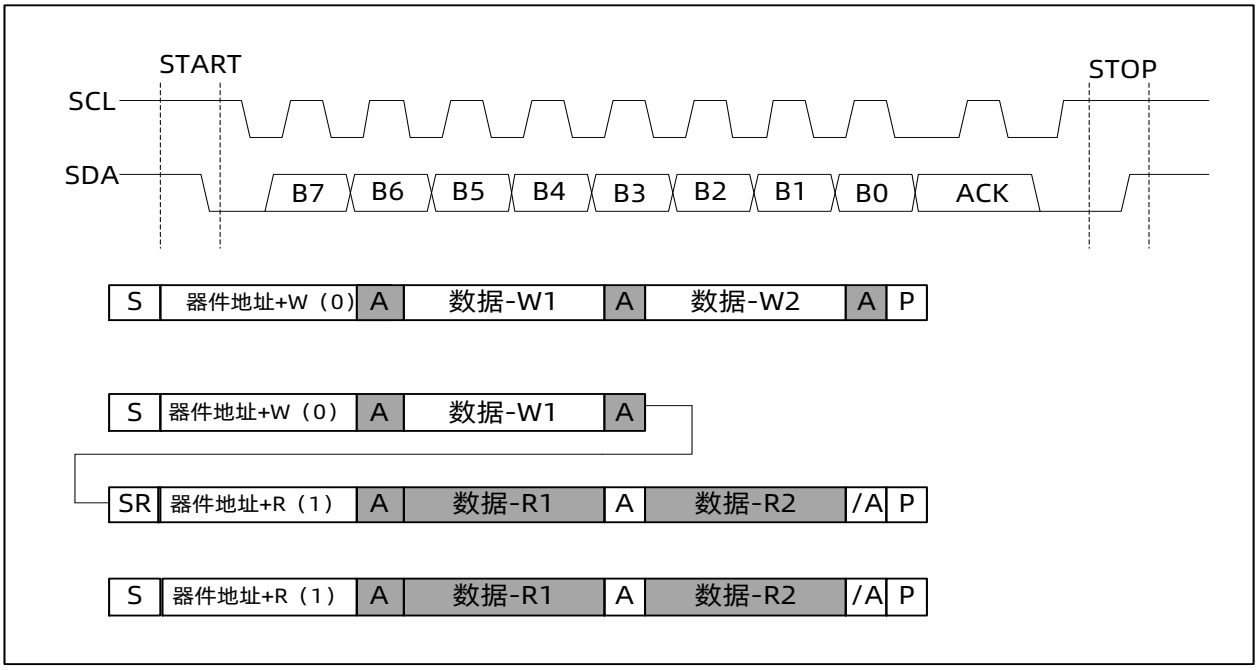
TXIE 置 1 时，TXIF 中断标志唤醒 CPU
RXIE 置 1 时，RXIF 中断标志唤醒 CPU
异步接收时，检测到 START 位将自动使能高频振荡器，接收完成后唤醒 CPU
同步接收时，若作为主机，则休眠状态下部工作；作为从机，则接收 1 个字节完成后唤醒 CPU

11 串行通讯口（I2C）

11.1 概述

M8P625 支持高速 I2C（400K）Slave。

11.2 通讯波形示意



11.3 I2CCON I2C 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CCON	I2CEN	R_W	D_A	BF	NACK	-	IICM	IICPS
读/写	R/W	R	R	R	R	-	R/W	R/W
复位后	0	0	0	0	0	-	0	0

Bit 7 **I2CEN**： 使能发送
0 = 屏蔽串行通讯功能
1 = 使能串行通讯功能

Bit 6 **R_W**： 发送寄存器空标志
0 = Master写数据（复位后、START、STOP后）
1 = Master读数据（主机发接收数据命令）

Bit 5 **D_A**： 数据地址标志
0 = 主机发送的是地址
1 = 主机发送的是数据(Start后发过的第一个数据)

Bit 4 **BF**： 数据缓冲区满
R_W = 0
0 = 已读或未接受到数据
1 = 从机接收到数据，未读
R_W = 1
0 = 数据已发送或正在发送
1 = 有数据待发送

Bit 3 **NACK**： Master读数据ACK
0 = Master继续接收下一个数据
1 = Master停止接收数据

Bit 1 **IICM**： IIC 总线模式
0 = 从机在未完成接收不 HOLD 总线(高速 Master 可能会造成通讯紊乱)
1 = 从机在未完成接收(未读 I2CBUF)时 HOLD 总线(拉低 I2C_SCL)

Bit 0 **IICPS**： I2C 总线脚位选择
0 = 选择A组总选(I2C_SCLA，I2C_SDAA)
1 = 选择B组总线(I2C_SCLB，I2C_SDAB)

11.4 I2CADR I2C 地址寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CADR	I2CA7	I2CA6	I2CA5	I2CA4	I2CA3	I2CA2	I2CA1	I2CA0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
复位后	0	0	0	0	0	0	0	0

Bit[7:0] I2CADR[**7:0**]： 从机地址
地址(I2C 写入)：I2CADR
地址(I2C 读取)：I2CADR+1

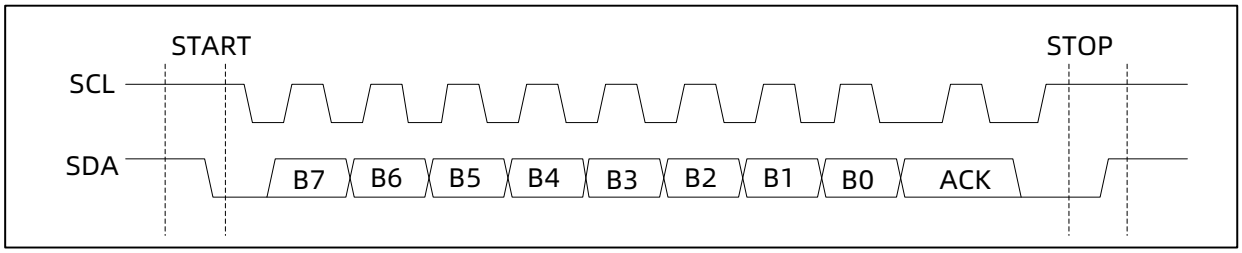
11.5 I2CBUF 数据寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CBUF	I2CBUF7	I2CBUF6	I2CBUF5	I2CBUF4	I2CBUF3	I2CBUF2	I2CBUF1	I2CBUF0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	X	X	X	X	X	X	X	X

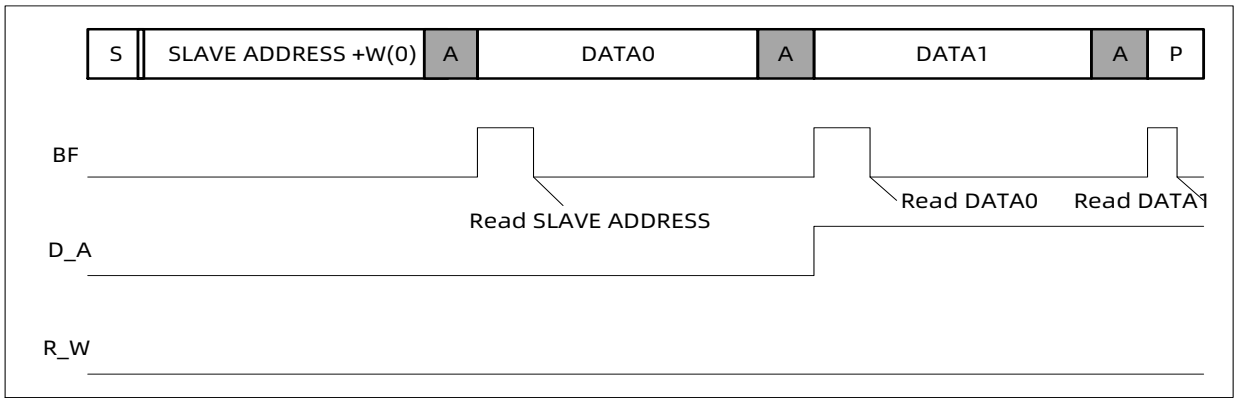
11.6 唤醒及休眠模式下通讯

当 I2CCON.7 为 1 时，SCL、SDA 线的低电平会唤醒 CPU，并开始通讯，通讯期间 CPU 无法进入休眠模式。

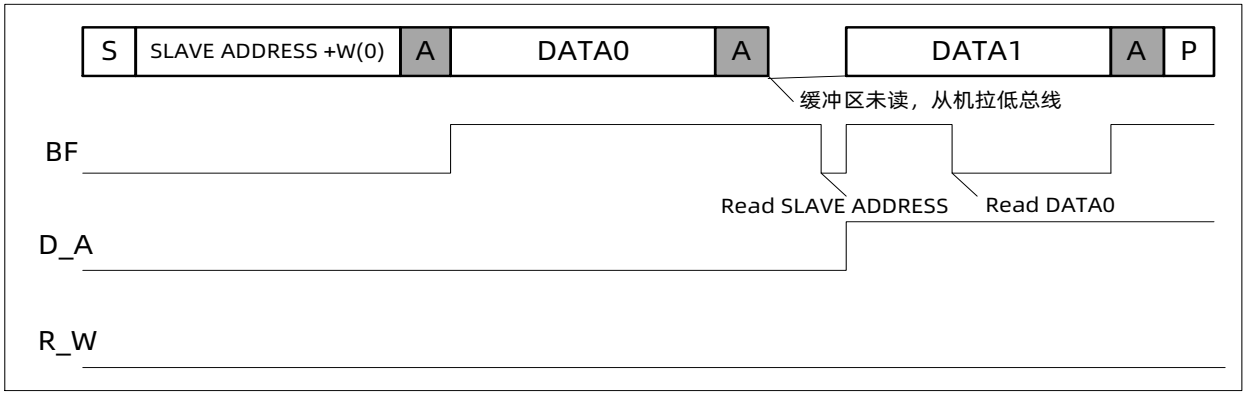
11.7 通讯波形图



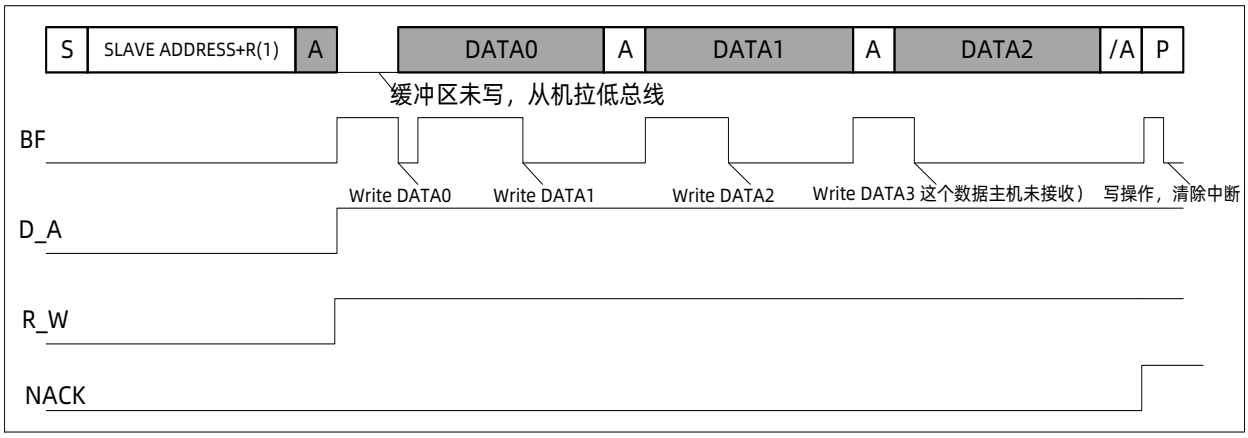
地址匹配后，主机向从机写入两个数据



两个数据没有读取，从机拉低 SCL，等待数据读取



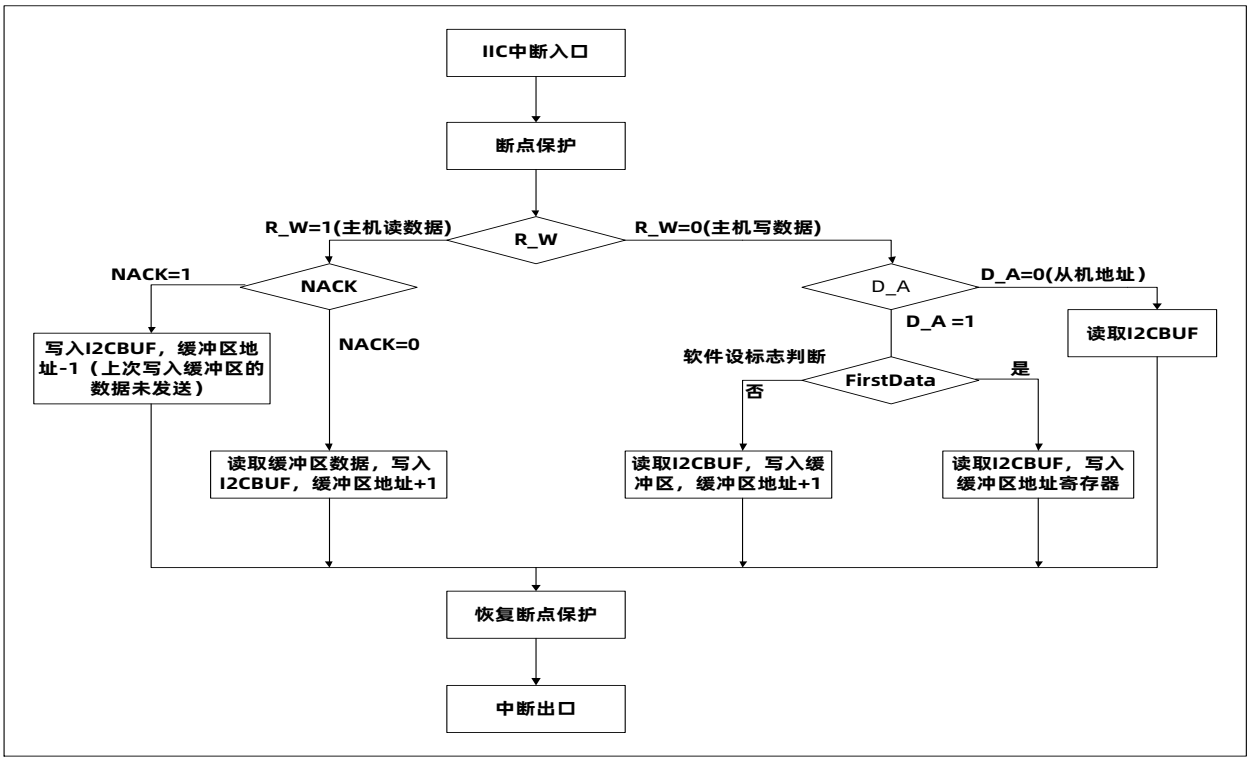
I2C Master 读数据此操作



11.8 应用示例

M8P625 作为从机，允许主机通过 I2C 接口对其 RAM 进行读写操作

11.8.1 从机软件流程图



11.8.2 例程

```
;//-------------------------------------------------------
LIST P= M8P625
#include     M8P625.inc
;//-------------------------------------------------------
#DEFINE          bFirstByte    0x0F,0        ;//设置软件标志
SlaveBuffAddr         EQU     0EH           ;//缓冲区地址
StartAdr              EQU     10H
;//-------------------------------------------------------
              ORG          0000H
              GOTO         MainProgram
;//-------------------------------------------------------
;//中断入口
              ORG          0008H
              GOTO         Interrupt
;//-------------------------------------------------------
```

```
MainProgram:
IIC_Init:                                   ;//IIC 初始化
            BSET        PUB,0               ;//上拉使能
            BSET        PUB,1
            BCLR        OEB,0
            BCLR        OEB,1
            MOVIA       b'10000010'
            MOVAR       I2CCON
            MOVIA       0xA2                ;//IICSLAVE 为 0xA2(写)/0xA3（读）
            MOVAR       I2CADR
            BSET        bI2CIE              ;//使能 IIC 中断
            BSET        OPTION,GIE
Main:
            CLRWDT
            GOTO        Main
;//++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;//中断处理子程序
Interrupt:
            PUSH
            MOVR        FSR0,A
            MOVAR       sFSR0               ;//中断中用到 FSR0,保护
            JBTS0       intf0,3
            GOTO        Interrupt _pint
Interrupt _End:
            BCLR        FLAG
            MOVR        sFSR0,A
            MOVAR       FSR0                ;//中断中用到 FSR0,保护
            POP
            RETIE
Interrupt _pint:
            BSET        FLAG
            JBTS1       I2CCON,R_W          ;//R_W
            GOTO        iicMasterWrite
iicMasterRead:
            JBTS0       I2CCON,NACK
            GOTO        iicMasterReadNoAck
            MOVR        SlaveBuffAddr,A
            MOVAR       FSR0
            MOVR        INDF0,A
            MOVAR       I2CBUF
            INCR        SlaveBuffAddr,R
            GOTO        Interrupt _End
iicMasterReadNoAck:                         ;//Master 读数据完毕
            DECR        SlaveBuffAddr,R     ;//上次写入数据未接收,地址减 1
```

```
                MOVAR       I2CBUF                  ;//清除中断
                GOTO        Interrupt _End
iicMasterWrite:
                JBTS0       I2CCON,D_A
                GOTO        iicGetBufAddr
iicAddr:
                MOVR        I2CBUF,A                ;//读数据,清空缓冲区
                BSET        bFirstByte
                GOTO        Interrupt _End
iicGetBufAddr:
                JBTS1       bFirstByte
                GOTO        iicGetData
                MOVR        I2CBUF,A
                MOVAR       SlaveBuffAddr
                BCLR        bFirstByte
                GOTO        Interrupt _End
iicGetData:
                MOVR        SlaveBuffAddr,A
                MOVAR       FSR0
                MOVR        I2CBUF,A
                MOVAR       INDF0
                INCR        SlaveBuffAddr,R
                GOTO        Interrupt _End
;//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

12 模数转换器（ADC）

12.1 概述

M8P625有一个12路外部通道（AIN0~AIN11）和3路内部通道（VDD/4，VREF和GND）12位分辨率的A/D 转换器，可以将模拟信号转换成12位数字信号。进行AD 转换时，首先要选择输入通道，然后启动AD转换。转换结束后，系统自动将EOC设置为“1”，并将转换结果存入寄存器ADH和寄存器ADL中。

注：ADC使用时最好去除最大和最小，取中间平均值，偶尔可能会有错误的值出来，在ADC采样转换之前、切换通道和参考电压都要注意延时一下16us。

12.2 ADCON0 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADCON0	ADON	ADS	ADEOC	ADFM	CHS3	CHS2	CHS1	CHS0
读/写	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	1	0	1	0

Bit 7 **ADON**： ADC使能控制位
0 = 关闭ADC
1 = 使能ADC

Bit 6 **ADS**： ADC 启动位
0 = 停止，转换完成自动清零
1 = 开始（每次写入1将重新启动ADC）

Bit 5 **ADEOC**： ADC 状态控制位
0 = 转换过程中
1 = 转换结束，ADS 复位

Bit 4 **ADFM**： 数据格式选择位
0 = ADRES = {ADH[7:0]，ADL[7:4]}；ADL[3:0] = 0
1 = ADRES = {ADH[3:0]，ADL[7:0]}；ADH[7:4] = 0

Bit [3:0] **CHS[3:0]**： ADC 输入通道选择位
[0000] ~ [1011] = AIN0 ~ AIN11
[1100] = VDD/4
[1101] = 内建 VREF 基准电平
[1111] = GND

注：（1）若 ADENB = 1，用户应设置 IOA.n/AINn 为无上拉电阻的输入模式，系统不会自动设置。若已经设置了 ANSEL.n，IOA.n/AINn 的数字 I/O 功能都是隔离开来的。
（2）ADC 输入通道选择 VDD/4 在休眠下也会产生电流。

12.3 ADCON1 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADCON1	AGEN	ADCKS2	ADCKS1	ADCKS0	VREMS1	VREMS0	VHS1	VHS0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

Bit 7 **AGEN**： 增益控制位

0 = 关闭增益控制

1 = 开启增益控制(VREMS=00时,此位无效)

Bit [6:4] **ADCKS[2:0]**： ADC 时钟源选择位

ADCKS[2:0]	ADC 时钟源选择
000	Fcpu
001	Fcpu/2
010	Fcpu/4
011	Fcpu/8
100	Fcpu/16
101	Fcpu/32
110	Fcpu/64
111	

Bit [3:2] **VREMS[1:0]**： ADC 参考电压模式选择位

VREMS[1:0]	ADC 参考电压模式
00	VDD
01	内部参考电压
10	外部参考电压
11	内部参考与外部参考连接

Bit [1:0] **VHS[1:0]**:

AGEN=0 时为 ADC 内建基准电平选择位

VHS[1:0]	内建 VREF 基准电平
00	关闭内部参考
01	2.0V
10	3.0V
11	4.0V

AGEN=1 时为增益选择位,将通道信号放大后进行 ADC 转换

VHS[1:0]	放大倍数
00	关闭内部参考
01	10.83 倍
10	10.82 倍
11	10.78 倍

注：（1）若由 VHS[1:0]控制选择的内部 VREF 电平高于 VDD，内部 VREF 为 VDD。
例：VHS[1:0] = 11（内部 VREF = 4.0V），VDD = 3.0V，则实际内部 VREF = 3.0V。
（2）12 位 AD 转换时间 = 16 个 AD 时钟。

12.4 ADCON2 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADCON2	-	-	-	-	ADVOS3	ADVOS2	ADVOS1	ADVOS0
读/写	-	-	-	-	R/W	R/W	R/W	R/W
复位后	-	-	-	-	0	0	0	0

Bit [3:0] **ADVOS[3:0]**: ADC失调补偿寄存器

注：小信号采集时需要注意校准

ADC通道选择内部GND通道，先设置ADCON2为0x00，若ADC的GND通道转换值为0，就把ADCON2加1，直到ADC的GND通道转换值不为0时，ADCON2的值就是调校好的值，ADCON2值最大等于15。

```
Main_Program:                                   ;//程序开始
ADC_Init:
;//1、ADC控制寄存器  设置
            MOVIA       b'10011111'             ;//选择内部GND通道
            MOVAR       ADCON0
            MOVIA       b'00000000'
            MOVAR       ADCON1
            MOVIA       b'00000000'
            MOVAR       ADCON2
;//2、开始校准
ADC_ CHANGE:
            BSET        ADCON0,6
            JBTS0       ADCON0,6
            GOTO        $-1
            MOVIA       0x00
            JCMPAR      ADH
            GOTO        ADC_ CHANGE_PRO
            JCMPAR      ADL
            GOTO        ADC_ CHANGE_PRO         ;//此时 ADC 转化值不为 0
            MOVIA       0x0F
            ANDAR       ADCON2,A
            JNCMPAI     0x0F                    ;//ADCON2 的值最大为 15
            GOTO        ADC_ CHANGE_END
            INCR        ADCON2,R
            GOTO        ADC_CHANGE
ADC_ CHANGE_PRO:
            MOVIA       0x0F
            ANDAR       ADCON2,A
            JCMPAI      0x00                    ;// ADCON2 的值最小为 0
            DECR        ADCON2,R                ;//调较后，ADCON2 的值减一
ADC_ CHANGE_END:                                ;//程序主循环
            …
```

12.5 ADH ADC 数据高字节

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADH	-	-	-	-	-	-	-	-
读/写	R	R	R	R	R	R	R	R
复位后	X	X	X	X	X	X	X	X

12.6 ADL ADC 数据低字节

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADL	-	-	-	-	-	-	-	-
读/写	R	R	R	R	R	R	R	R
复位后	X	X	X	X	X	X	X	X

注：ADH/ADL 的数据格式与 ADM 相关，当 ADFM=1 时，ADH[7:4] =0,ADH[3:0]存放高四位数据 ADL[7:0]存放低 8 位数据；当 ADFM=0 时，ADH[7:0]存放高 8 位数据，ADL[7:4]存放低 4 位数据，AD:[3:0] = 0。

12.7 ADC 范例

例: ADC 模数转换器

将高位的值存在寄存器 ADC_DATAH，低位存在寄存器 ADC_DATAL。

```
ADC_DATAH       EQU  00H          ;//特殊寄存器定义声明
ADC_DATAL           EQU  01H
;// ++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                ORG          0000H
                GOTO         Main_Program    ;//跳转到程序开始
                ORG          0008H
                GOTO         Interrupt       ;//发生中断后,跳转到中断子程序
;//++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Main_Program:                                ;//程序开始
ADC_Init:                                    ;//ADC初始化
//1、端口设置
                BCLR         OEA,1
                BSET         ANSA,1          ;//AIN1设置为模拟输入
;//2、ADCON0 设置
                MOVIA        b'10010001'     ;//使能ADC,输入通道选择AIN1
                MOVAR        ADCON0
;//3、ADCON1 设置
                MOVIA        b'00000000'
                MOVAR        ADCON1
;//4、ADCON2 设置
                MOVIA        b'00000000'
                MOVAR        ADCON2
                NOP                          ;//延时一段时间再采集
                ...
                NOP
;//5、开启转换
                BSET         ADCON0,6
                JBTS0        ADCON0,6
                GOTO         $-1
                MOVR         ADH,A
                MOVAR        ADC_DATAH
                MOVR         ADL,A
                MOVAR        ADC_DATAL
```

```
Main:                                   ;//程序主循环
            ...
            GOTO        Main
;//++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Interrupt:                              ;//中断子程序
            PUSH                        ;//压栈,保存 A,STATUS
;//中断处理程序
            NOP
Interrupt_End:
            POP                         ;//出栈,恢复 A,STATUS
            RETIE

            END
```

13 看门狗（WDT）

13.1 概述

看门狗定时器的时钟为内部独立 RC 时钟。
配置字 WDTEN 设置看门狗定时器的三种工作状态：
（1）始终开启 WDT 功能,即在 STOP 模式下仍然工作，溢出可唤醒 STOP
（2）使能：绿色或休眠模式下关闭，即 STOP 下关闭
（3）屏蔽 WDT 功能，即始终关闭
配置字 TWDTEN 设置看门狗的四种溢出时间： 4.5ms、18ms、72ms 或 288ms。

注：看门狗正常溢出后，程序复位到 0000H，但是在休眠模式下看门狗溢出程序是继续往下运行。

13.2 OPTION 配置寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
OPTION	GIE	-	TO	PD	-	-	-	-
读/写	R/W	-	R	R	-	-	-	-
复位后	0	-	1	1	-	-	-	-

Bit 5 **TO**： 超时位
0 = WDT发生溢出
1 = 上电复位或清除WDT

Bit 4 **PD**： 掉电位
0 = 进入休眠模式
1 = 上电复位或清除WDT

13.3 WDTC 看门狗控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
WDTC	WDTC7	WDTC6	WDTC5	WDTC4	WDTC3	WDTC2	WDTC1	WDTC0
读/写	W	W	W	W	W	W	W	W
复位后	-	-	-	-	-	-	-	-

注：（1）WDTC 写入 0x5A 将清除 WDT 定时器，写入其他值无效。
（2）CLRWDT 指令也可清除 WDT 定时器。

14 芯片配置字（OPTION BIT）

烧录选项	内容	说明
程序区写入保护	允许改写程序区	
	不允许改写程序区	
CPU 运行速度选择	4T（LVR>2.1V）	高频模式下 CPU 速度选择;
	8T（LVR>1.7V）	
	16T	
	32T	
	64T	
	128T	
	256T	
晶振驱动电流	驱动 0（普驱）	默认驱动 1
	驱动 1（强驱）	
晶振反馈电阻	电阻 0（小电阻）	默认电阻 1
	电阻 1（大电阻）	
振荡器模式	HIRC+LIRC	双系统时钟
	HXT+LIRC	
	HIRC+LXT	
外部复位端口	作为外部复位端口	
	作为 IO 口	
启动模式选择	高速启动	
	低速启动	
输出端口读取	从端口读取	
	从输出寄存器读取	

烧录选项	内容	说明
复位电压选择	LVR=3.8V	系统高速运行时，请选择相应较高的 LVR 电压，以保证系统的可靠性
	LVR=3.7V	
	LVR=3.6V	
	LVR=3.5V	
	LVR=2.5V	
	LVR=2.4V	
	LVR=2.3V	
	LVR=2.2V	
	LVR=2.1V（FCPU<4T）	
	LVR=2.0V（FCPU<4T）	
	LVR=1.9V（FCPU<4T）	
	LVR=1.8V（FCPU<4T）	
	LVR=1.7V（FCPU<8T）	
	LVR=1.6V（FCPU<8T）	
	LVR=1.5V（FCPU<8T）	
	LVR=1.4V（FCPU<8T）	
低频晶振供电	芯片 VDD	
	BIAS 供电	
EEPROM 写入保护	不允许 EE 写入	
	允许 EE 写入	
WDT 使能选择	屏蔽 WDT 功能	
	使能，绿色或休眠模式下关闭	
	始终开启 WDT 功能	
WDT 溢出时间	WDT 溢出时间=4.5ms	VDD=5V 典型值
	WDT 溢出时间=18ms	
	WDT 溢出时间=72ms	
	WDT 溢出时间=288ms	
仿真电压选择	VDD 5.0V（<200mA）	
	VDD 3.3V（<300mA）	
	外供电源	
EEPROM 选择	下载时擦除 EEPROM	
	下载时不擦除 EEPROM	

15 电性参数

15.1 极限参数

储存温度.....-50℃~125℃
工作温度.....-40℃~85℃
电源供应电压.....0V~5.5V
端口输入电压.....GND-0.3V~VDD+0.3V

注：如果器件工作条件超出上述极限参数，将造成器件永久性破坏。如果在极限参数最大值上长时间工作，器件稳定性会受到影响。为保障器件稳定运行请在规定范围内工作。

15.2 直流特性

符号	参数	测试条件		最小值	典型值	最大值	单位
		VDD	条件（常温 25℃）				
VDD	工作电压	—	Fosc = 16MHz，16T	1.8	-	5.5	V
IDD1	动态电流 1	3V	高频运行（HIRC=16M） 低频运行（LIRC=64K） FCPU=HIRC/16T 全速工作	-	0.8	-	mA
		5V		-	1.0	-	
ISP1	静态电流 1	3V	高频运行（HIRC=16M）， 低频运行（LIRC=64K）， STOP =1 无唤醒源	-	130	-	uA
		5V		-	200	-	
ISP2	静态电流 2	3V	高频停止 低频运行（LIRC=64K） STOP =1 无唤醒源	-	1.5	-	uA
		5V		-	5.5	-	
ISP3	静态电流 3	3V	高频停止 低频停止 STOP =1 无唤醒源	-	0.3	-	uA
		5V		-	0.5	-	
ISP4	静态电流 4	3V	高频停止（HIRC=16M） 低频运行（LIRC=64K）， FCPU=HIRC/16T，STOP =1， WDT 唤醒（72ms）	-	5.6	-	uA
		5V		-	1.6	-	
ISP5	静态电流 5	3V	高频停止 外部低频运行(LXT=32768HZ) STOP=1 无唤醒源	-	3	-	μA
		5V		-	10	-	

符号	参数	测试条件		最小值	典型值	最大值	单位
		VDD	条件（常温 25℃）				
V_{IL1}	输入低电平	3V	SMT	0	-	0.3VDD	V
V_{IH1}	输入高电平	3V		0.7VDD	-	VDD	
V_{IL2}	输入低电平	5V		0	-	0.3VDD	
V_{IH2}	输入高电平	5V		0.6VDD	-	VDD	
V_{IL3}	输入低电平	3V	低翻转	0	-	0.2VDD	
V_{IH3}	输入高电平	3V		0.3VDD	-	VDD	
V_{IL4}	输入低电平	5V		0	-	0.1VDD	
V_{IH4}	输入高电平	5V		0.2VDD	-	VDD	
R_{PH}	上拉电阻	5V	VIN = GND	-	20	-	kΩ
		3V	VIN = GND	-	40	-	
R_{PL}	下拉电阻	5V	VIN = VDD	-	20	-	
		3V	VIN = VDD	-	40	-	
I_{OL1}	输出灌电流	5V	输出口，Vout=GND+0.6V	-	6	-	mA
		3V		-	4	-	
I_{OH1}	输出拉电流	5V	输出口，Vout=VDD-0.6V	-	4	-	
		3V		-	3	-	
I_{OL2}	输出灌电流	5V	输出口，Vout =GND+0.6V	-	55	-	
		3V		-	35	-	
I_{OH2}	输出拉电流	5V	输出口，Vout=VDD-0.6V	-	25	-	
		3V		-	15	-	
I_{OL3}	输出灌电流	5V	输出口，Vout =GND+0.6V	-	10	-	
		3V		-	8	-	

注：具体值不做设计保证。

15.3 交流特性

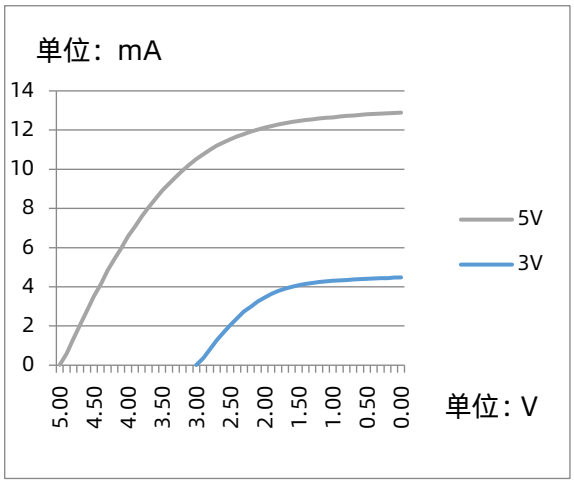
常温(25℃)下，测试了 5 种不同等效串联电阻（ESR）类型的 32.768KHZ 晶振，在以下推荐的两种晶振配置字下所对应的不同负载电容的起振时间、过秒和电流数据：

晶振配置字参数	测试条件			起振时间 (mS)	过秒 (M/S)	电流 (uA)
	等效串联电阻 ESR 类型(KΩ)	负载电容（pF）	VDD			
驱动电流:驱动 1(强驱) 反馈电阻:电阻 1(大电阻) 供电选择:芯片 VDD 供电	30	18	5V	240	186	13
			3V	310	3	2.5
	40	18	5V	270	203	13.8
			3V	340	20	2.7
	50	18	5V	340	143	15.3
			3V	460	4	2.9
	60	24	5V	270	131	14
			3V	580	25	3.2
	70	24	5V	250	123	13.8
			3V	650	19	3.1
驱动电流:驱动 1(强驱) 反馈电阻:电阻 1(大电阻) 供电选择:BIAS 供电 (此配置为默认配置字)	30	18	5V	370	-5	13
			3V	790	-17	9.5
	40	18	5V	440	4	12.9
			3V	920	-4	9.4
	50	18	5V	690	-18	14
			3V	1400	-23	107
	60	24	5V	1300	5	13.9
			3V	2400	-1	10.8
	70	24	5V	1100	-11	12.3
			3V	2200	-17	9.6

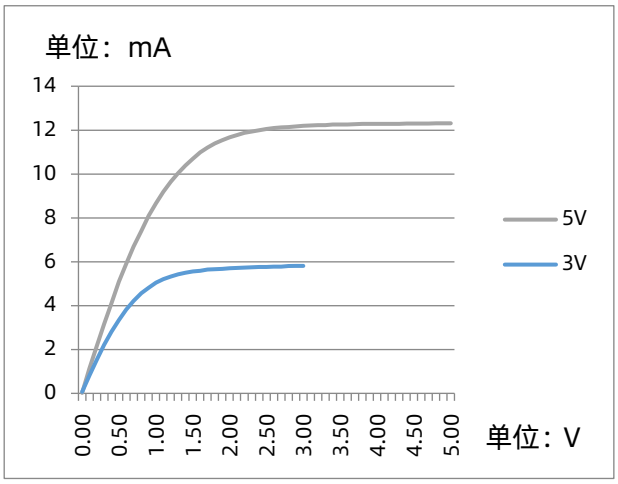
注：（1）晶振供电选择为 VDD 供电时起振时间会偏小，但是过秒会随电压变化而变化。
晶振供电选择为 BIAS 供电时起振时间会偏大，但是过秒不会随电压变化而变化。
（2）强驱：可以快速起振，但消耗更多电流。
普驱：功耗低，但起振时间较长。
大电阻：与晶振的谐振电阻兼容性较好，推荐使用。
小电阻：对晶振的谐振电阻比较敏感，如果谐振电阻过大可能会导致晶振起振困难。
（3）数据仅供参考，具体值不做设计保证。

15.4 IO 口拉灌电流特性

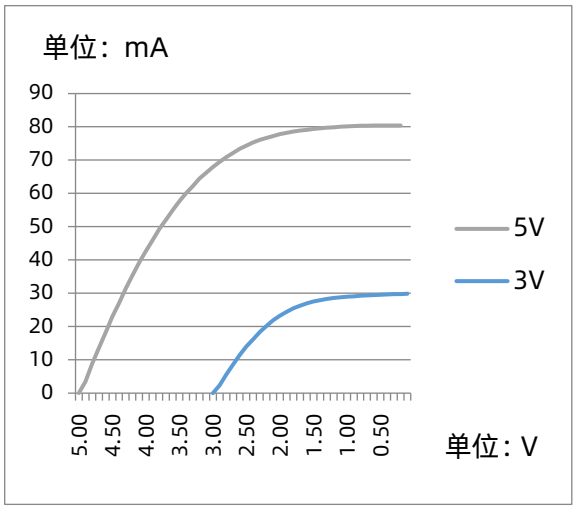
I_{OH1}



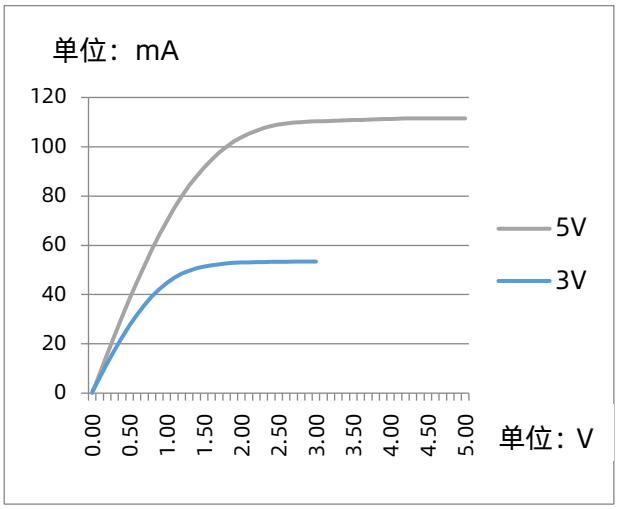
I_{OL1}



I_{OH2}

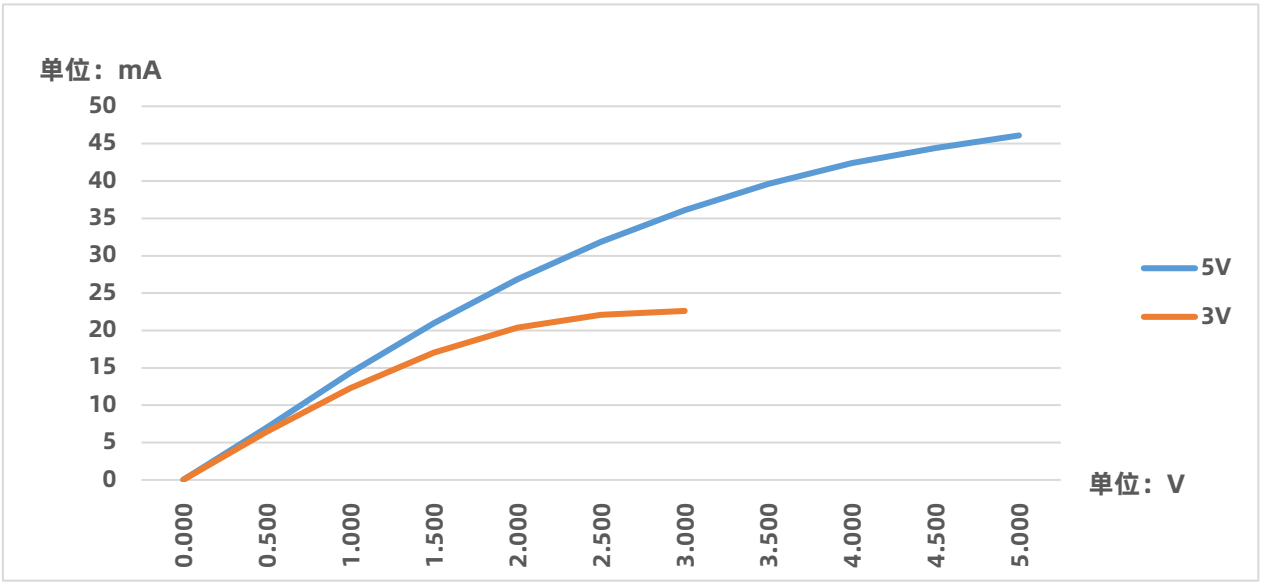


I_{OL2}



注：具体值不做设计保证。

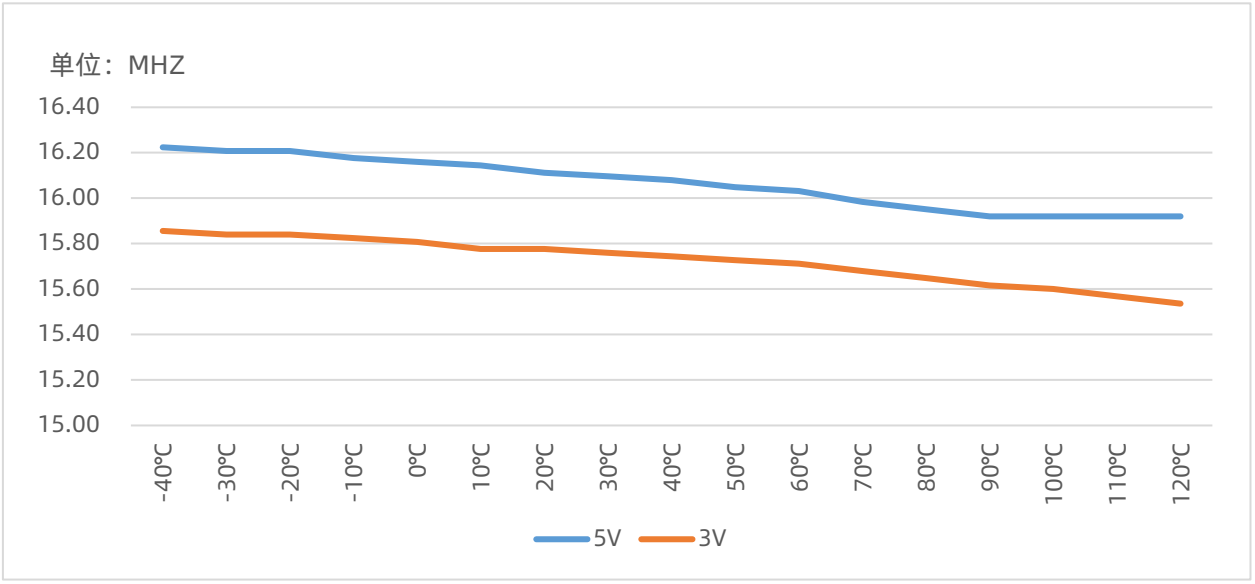
IOB2（VPP 口）灌电流 I_{OL3}



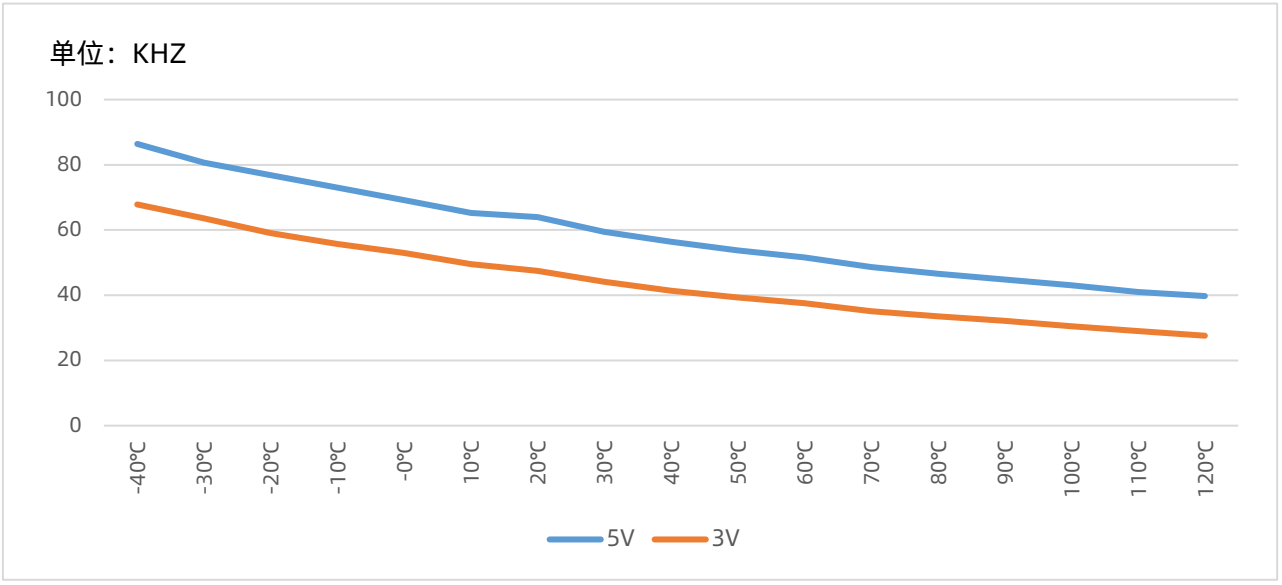
注：具体值不做设计保证。

15.5 系统时钟特性

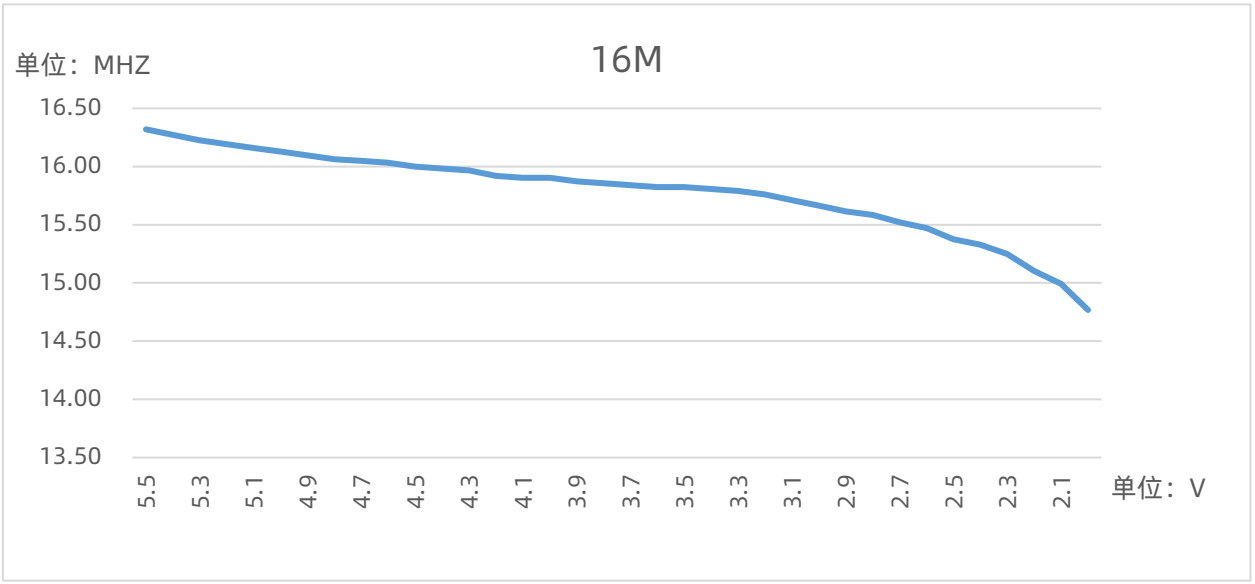
高频时钟（16MHZ）随温度变化曲线



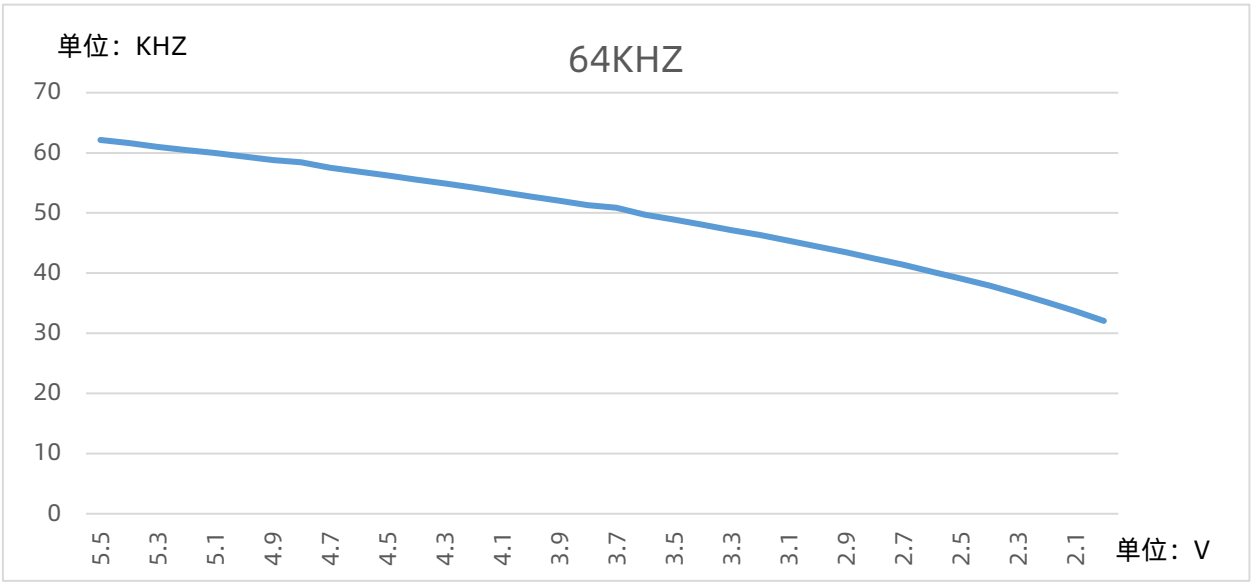
低频时钟（64KHZ） 随温度变化曲线



常温下（25℃），高频时钟（16MHZ）随电压变化曲线



常温下（25℃），低频时钟（64KHZ）随电压变化曲线



注：具体值不做设计保证。

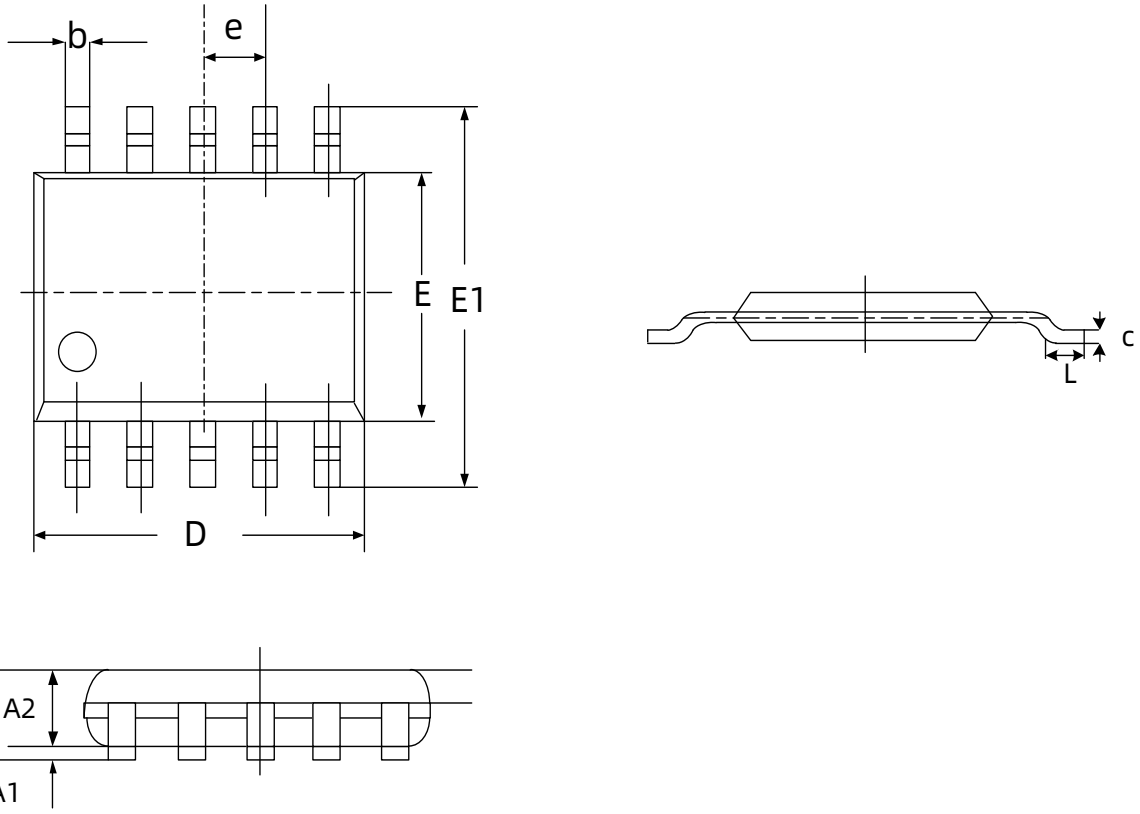
15.6 ADC 电气特性

符号	参数	测试条件 条件（常温 25℃）	最小值	典型值	最大值	单位
V_{ADC}	ADC 工作电压	-	2.5	-	5.5	V
I_{ADC}	ADC 工作电流	VDD=5V	-	200	-	μA
V_{AIN}	ADC 输入电压	-	GND	-	V_{REF}	V
V_{iREF1}	内部参考电压 1	内部 2V 参考电压,VDD>=2V+0.5V	-2%	2	+2%	V
V_{iREF2}	内部参考电压 2	内部 3V 参考电压,VDD>=2V+0.5V	-2%	3	+2%	V
V_{iREF3}	内部参考电压 3	内部 4V 参考电压,VDD>=2V+0.5V	-2%	4	+2%	V
T_{VREF}	参考稳定时间	VDD=5V，参考电压选择和切换后	-	50	-	μs
F_{ADC}	ADC 时钟	V_{REF}= VDD =5V	-	-	2	MHZ
T_{CON}	ADC 转换时间	-	-	16	-	T_{ADC}
DNL	微分非线性误差	V_{REF}= VDD =5V，F_{ADC}=1MHZ	-2	-	+2	LSB
INL	积分非线性误差	V_{REF}= VDD =5V，F_{ADC}=1MHZ	-3	-	+3	LSB
E_Z	偏移误差	V_{REF}= VDD =5V，F_{ADC}=1MHZ	-5	-	+5	LSB
E_F	满刻度误差	V_{REF}= VDD =5V，F_{ADC}=1MHZ	-10	-	+10	LSB

注：具体值不做设计保证。

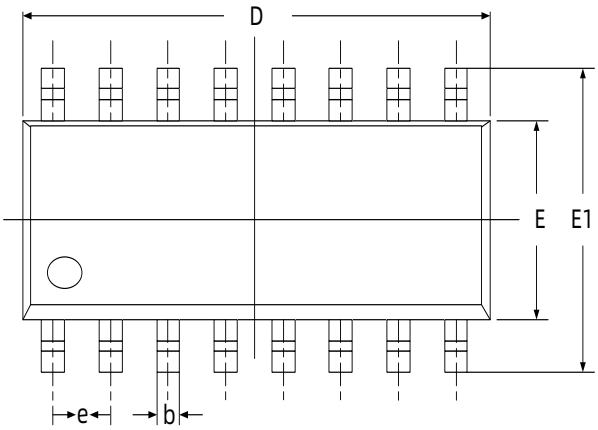
16 封装信息

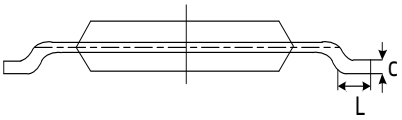
16.1 MSOP10

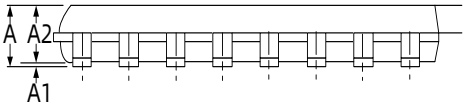


符号	单位（mm）		
	最小	正常	最大
A	0.820	---	1.100
A1	0.020	---	0.150
A2	0.750	0.850	0.950
b	0.180	---	0.280
D	2.900	3.000	3.100
e	0.500BSC		
E	2.900	3.000	3.100
E1	4.750	---	5.050
L	0.400	---	0.800
c	0.090	---	0.230

16.2 SOP16

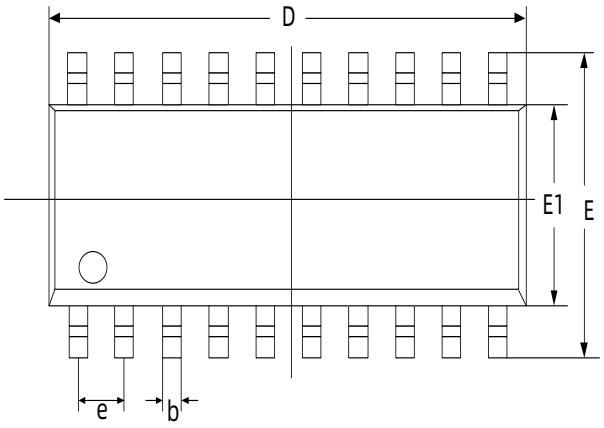


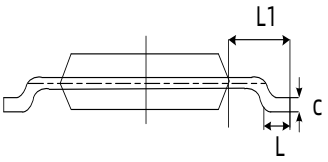


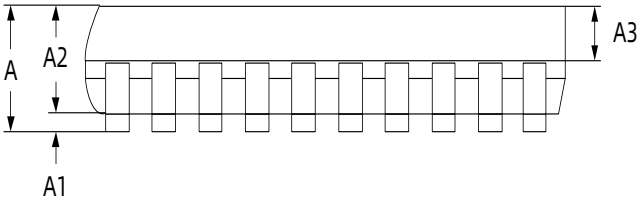


符号	单位（mm）		
	最小	正常	最大
A	1.520	1.600	1.680
A1	0.100	0.150	0.200
A2	1.420	1.450	1.480
b	---	0.406	---
D	9.750	9.800	9.850
e	1.270BSC		
E	3.750	3.800	3.850
E1	6.050	6.150	6.250
L	0.350	0.500	0.650
c	---	0.203	---

16.3 SOP20







符号	单位（mm）		
	最小	正常	最大
A	-	-	2.650
A1	0.100	0.150	0.200
A2	2.250	2.300	2.350
A3	0.970	1.020	1.070
b	0.350	-	0.430
D	12.700	12.800	12.900
e	1.270BSC		
E	10.250	10.350	10.450
E1	7.400	7.500	7.600
L	0.550	0.700	0.850
L1	1.400REF		
c	0.250	---	0.290

17 指令集简述

17.1 概述

M8Pxxx系列指令集是一种精简指令集（RISC），指令宽度为16位，由操作码和0～2个操作数组成。指令按照功能可分为5类，即字节操作指令、位操作指令、立即数指令、分支指令、特殊控制指令。

一个指令周期由 1 个系统时钟周期组成,除非条件测试结果为真或指令执行改变了程序计数器的值，否则执行所有的指令都只需要一个指令周期。对于上诉两种特征情况，指令执行需要两个指令周期。

任何一条指定文件寄存器作为指令一部分的指令都进行读-修改-写操作。读寄存器、修改数据并根据指令或目标标识符“d”存储结果。即使是写寄存器的指令也将先对改寄存器进行读操作。

17.2 符号说明

符号	范围	说明	符号	范围	说明
R/r	0-0x1ff	寄存器地址	C	-	进位标志
A	-	ACC 寄存器	DC	-	半进位标志
B/b	0-7	位地址	Z	-	零标志
I/i	0-0xff	立即数	d	0-1	目的操作数定义
K/k	0-0x1fff	标号	GIE	-	总中断使能位
TOS	-	栈顶	stkp	-	堆栈指针
PC	-	PC 指针			

17.3 M8Pxxx 指令集表

指令集表中，d=1，目的操作数为 R；d=0，目的操作数为 A

指令类型	助记符	指令说明	周期数	影响标志位	备注
寄存器操作指令	ADDAR R,d	R+A → d	1	Z,DC,C	
	ADCAR R,d	R+A+C → d	1	Z,DC,C	
	SUBAR R,d	A-R → d	1	Z,DC,C	
	SBCAR R,d	A-R- C → d	1	Z,DC,C	
	SUBRA R,d	R-A → d	1	Z,DC,C	
	SBCRA R,d	R-A- C → d	1	Z,DC,C	
	ANDAR R,d	R&A→ d	1	Z	
	ORAR R,d	R|A → d	1	Z	
	XORAR R,d	R^A → d	1	Z	
	COMR R,d	R → d	1	Z	
	MOVR R,d	R → d	1	Z	
	MOVAR R	A → R	1	-	
	CLRR R	0 → R	1	Z	
	SWAPR R,d	R 半字节交换 → d	1	-	
	RLR R,d	R[7] → C, {R[6:0],C} → d	1	C	
	RLRNC R,d	{R[6:0],0} → d	1	-	
	RRR R,d	R[0] → C, {C,R[7:1]} → d	1	C	
	RRRNC R,d	{0,R[7:1]} → d	1	-	
	DECR R,d	R-1 → d	1	Z	
	DJZR R,d	R-1 → d, SKIP if 0	1(2)	-	
	INCR R,d	R+1 → d	1	Z	
	JZR R,d	R+1 → d, SKIP if 0	1(2)	-	
	JNZR R,d	R+1 → d, SKIP if !0	1(2)	-	
	DJNZR R,d	R-1 → d, SKIP if !0	1(2)	-	
	JCMPAR R	SKIP if A=R	1(2)	Z,C	
	JNCMPAR R	SKIP if A≠R	1(2)	Z,C	
	JGAR R	SKIP if A≥R	1(2)	Z,C	
	JLAR R	SKIP if A<R	1(2)	Z,C	
	XCHAR R	A←→R	1	-	
位操作指令	JBTS0 R,b	SKIP if R[b]=0	1(2)	-	
	JBTS1 R,b	SKIP if R[b]=1	1(2)	-	
	BCLR R,b	0 → R[b]	1	-	
	BSET R,b	1 → R[b]	1	-	

指令类型	助记符	指令说明	周期数	影响标志位	备注
立即数操作指令	ADDIA I	I+A → A	1	Z,DC,C	
	ADCIA I	I+A+C → A	1	Z,DC,C	
	SUBIA I	I-A → A	1	Z,DC,C	
	SBCIA I	I-A-C → A	1	Z,DC,C	
	SUBAI I	A-I → A	1	Z,DC,C	
	SBCAI I	A-I-C → A	1	Z,DC,C	
	ANDIA I	A&I → A	1	Z	
	ORIA I	A|I → A	1	Z	
	XORIA I	A^I → A	1	Z	
	MOVIA I	I → A	1	-	
	RETIA I	Stack → PC, I → A	2	-	
	JCMPAI I	SKIP if A=I	1(2)	Z,C	
	JNCMPAII	SKIP if A≠I	1(2)	Z,C	
特殊操作指令	RLA	A[7] → C, {A[6:0],C}→ A	1	C	
	RLANC	{A[6:0],0} → A	1	-	
	RRA	A[0] → C, {C,A[7:1]} → A	1	C	
	RRANC	{0,A[7:1]} → A	1	-	
	DECA	A-1 → A	1	Z	
	DJZA	A-1 → A, SKIP if 0	1(2)	-	
	INCA	A+1 → A	1	-	
	JZA	A+1 → A, SKIP if 0	1(2)	-	
	RETIE	Stack → PC, 1 → GIE	2	-	
	RETURN	Stack → PC	2	-	
	NOP	None Operation	1	-	
	RDT	ROM[{fsr1,fsr0}] →{HBUF, A}	3	-	
	DAA	加法后十进制调整	1	DC，C	
	DSA	减法后十进制调整	1	DC，C	
	PUSH	A，STATUS 压栈	1	-	
	POP	A，STATUS 出栈	1	Z，DC，C	
	CLRWDT	清除 WDT 寄存器	1	PD，TO	
分支指令	CALL I	I → PC，PC → Stack	2	-	
	GOTO I	I → PC	2	-	

17.4 M8Pxxx 指令说明

指令集详细说明请到官网下载：
M8Pxxx 指令说明

18 修正记录

版本	日期	描述
V1.00	2018-11-11	初版
…	…	…
V2.07	2024-08-14	勘误