

# M9F680

**8BIT  
TK+AD 型  
FLASH MCU**

Version 1.10

2024 年 01 月



**磐 芯 电 子**

本公司保留对产品可靠性、功能和设计方面的改进作进一步说明的权利  
数据手册的更改,恕不另行通知

<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

<b>1 产品简述</b> .....	<b>8</b>
1.1 特性 .....	8
1.2 应用注意事项.....	10
1.3 引脚图.....	11
1.3.1 SOP16.....	11
1.3.2 TSSOP20.....	12
1.3.3 SOP20 .....	13
1.3.4 SOP28 .....	14
1.3.5 SSOP28 .....	15
1.4 引脚描述 .....	16
<b>2 中央处理器 (CPU)</b> .....	<b>19</b>
2.1 程序存储器.....	19
2.1.1 复位向量 (0000h) .....	19
2.1.2 中断向量 (0008h,0018h) .....	20
2.1.3 程序计数器.....	20
2.1.4 堆栈.....	21
2.1.5 SPUSH 和 SPOP 指令.....	22
2.1.6 快速寄存器堆栈 .....	23
2.1.7 查表.....	23
2.2 数据存储器.....	24
2.2.1 数据存储器结构.....	24
2.2.2 BSR 页面选择寄存器 .....	24
2.2.3 快速操作寄存器 .....	25
2.2.4 数据存储器寻址模式 .....	26
2.2.5 间接寻址 0/1/2 .....	27
2.2.6 8x8 硬件乘法器 .....	28
2.2.7 STATUS 状态寄存器.....	28
2.2.8 系统寄存器定义 .....	29
2.3 FLASH 自编程 (IAP) .....	31
2.3.1 EECON1 寄存器 .....	31
2.3.2 EECON2 寄存器.....	31
2.3.3 Flash 写入缓存器 .....	32
2.3.4 Flash 操作示例.....	32
<b>3 复位</b> .....	<b>33</b>
3.1 复位方式.....	33
3.2 PCON 复位状态寄存器 .....	34
<b>4 系统时钟</b> .....	<b>35</b>
4.1 概述 .....	35
4.2 OSCM 寄存器 .....	35
4.3 系统时钟的工作模式.....	36
4.3.1 普通模式.....	36

4.3.2 绿色模式 .....	36
4.3.3 休眠模式 .....	36
4.4 IRCCAL 寄存器.....	37
4.5 IRCCAH 寄存器.....	37
4.6 系统时钟结构框图.....	39
4.7 系统时钟高低频切换.....	40
<b>5 中断 .....</b>	<b>41</b>
5.1 概述 .....	41
5.2 中断向量分配 .....	41
5.3 OPTION 配置寄存器.....	42
5.4 IO 端口变化中断使能寄存器 .....	43
5.5 INTCR0 中断控制寄存器 0 .....	44
5.6 INTF0 中断标志寄存器 0 .....	45
5.7 INTP0 中断优先级寄存器 0.....	46
5.8 INTCR1 中断控制寄存器 1 .....	47
5.9 INTF1 中断标志寄存器 1 .....	48
5.10 INTP1 中断优先级寄存器 1 .....	49
5.11 INTCR2 中断控制寄存器 2 .....	50
5.12 INTF2 中断标志寄存器 2 .....	50
5.13 INTP2 中断优先级寄存器 2.....	50
<b>6 端口 .....</b>	<b>51</b>
6.1 数据寄存器 IOx (x = A/B/C/D) .....	51
6.2 输出锁存寄存器 IOxOR (x = A/B/C/D) .....	51
6.3 输出方向寄存器 OEx (x = A/B/C/D) .....	52
6.4 上拉控制寄存器 PUX (x = A/B/C/D) .....	52
6.5 下拉控制寄存器 PDX (x = A/B/C/D) .....	52
6.6 模拟端口设置寄存器 ANSx (x = A/B/C/D) .....	53
6.7 驱动电流选择寄存器 IOxODS1/ IOxODS0 (x = A/B/C/D) .....	53
6.8 翻转电平设置寄存器 IOxIPS/FLIPCR (x = A/B/C/D) .....	54
6.9 数字功能可选端口控制模块.....	55
6.9.1 端口复用控制寄存器表 .....	55
6.9.2 全端口映射控制 .....	55
6.9.3 PWM0 复用控制.....	56
6.9.4 LED COM 端口复用控制寄存器 .....	56
<b>7 定时器 0(TC0/PWM0) .....</b>	<b>57</b>
7.1 概述 .....	57
7.2 T0CR 控制寄存器.....	58
7.3 T0CR2 TC0 控制寄存器 2.....	59
7.4 TC0CL TC0 计数器低位.....	59
7.5 TC0CH TC0 计数器高位 .....	59
7.6 PWM0xCR PWM 控制寄存器 (x = 0,1,2,3) .....	60
7.7 PWM0xD PWM 占空比寄存器 (x = 0,1,2,3) .....	60
7.8 PWM0x 波形实例 (x = 0,1,2,3) .....	61

<b>8 定时器 1 (TC1/PWM1)</b> .....	<b>62</b>
8.1 概述 .....	62
8.2 T1CR 控制寄存器 .....	63
8.3 INTPTC1 TC1 中断控制寄存器 .....	64
8.4 TC1CL TC1 计数器低位 .....	64
8.5 TC1CH TC1 计数器高位 .....	64
8.6 TC1PRL TC1 周期寄存器低位 .....	65
8.7 TC1PRH TC1 周期寄存器高位 .....	65
8.8 PWM1CR 控制寄存器 .....	66
8.9 PWM1xD 数据寄存器 (x = 0,1) .....	67
8.10 PWM 死区控制寄存器 .....	67
8.11 PWM1 波形示例 .....	68
8.11.1 互补 PWM1 输出 .....	68
8.11.2 带死区的互补 PWM1 输出 .....	68
8.11.3 PWM 波形图 .....	69
<b>9 定时器 2 (TC2)</b> .....	<b>70</b>
9.1 概述 .....	70
9.2 T2CR 控制寄存器 .....	71
9.3 TC2CL 计数器低位 .....	71
9.4 TC2CH 计数器高位 .....	71
9.5 TC2PRL 周期寄存器低位 .....	72
9.6 TC2PRH 周期寄存器高位 .....	72
9.7 TC2GCR 门控控制寄存器 .....	73
9.7.1 门控-TC0 溢出周期 .....	74
9.7.2 门控-上升沿到下降沿模式 .....	74
9.7.3 门控-下降沿到上升沿模式 .....	74
9.7.4 门控-上升沿到上升沿模式 .....	75
9.7.5 门控-下降沿到下降沿模式 .....	75
<b>10 通用串行通讯口 (USART)</b> .....	<b>76</b>
10.1 概述 .....	76
10.2 TX0CR 发送控制寄存器 .....	76
10.3 TX0REG 发送数据寄存器 .....	77
10.4 RX0CR 接收控制寄存器 .....	77
10.5 RX0REG 接收数据寄存器 .....	78
10.6 BRGD0H 波特率寄存器高位 .....	78
10.7 BRGD0L 波特率寄存器低位 .....	78
10.8 USART 使用说明 .....	79
10.8.1 波特率设置 .....	79
10.8.2 异步发送 .....	80
10.8.3 异步接收 .....	82
10.8.4 同步发送 .....	83
10.8.5 同步接收 .....	85
10.8.6 唤醒及休眠模式下通讯 .....	86

<b>11 I2C .....</b>	<b>87</b>
11.1 概述.....	87
11.2 功能描述.....	88
11.2.1 起始 START 和停止 STOP 信号 .....	89
11.2.2 7 位地址数据格式.....	89
11.2.3 应答 .....	89
11.2.4 仲裁 .....	90
11.3 工作模式.....	91
11.3.1 主机发送模式.....	91
11.3.2 主机接收模式 .....	92
11.3.3 从机接收模式 .....	93
11.3.4 从机发送模式 .....	94
11.3.5 广播呼叫.....	95
11.3.6 其它状态.....	96
11.4 I2C 控制寄存器 .....	97
11.4.1 I2C 控制寄存器 .....	97
11.4.2 I2C 状态寄存器 .....	98
11.4.3 I2C 数据寄存器 .....	98
11.4.4 I2C 从机地址寄存器 .....	99
<b>12 SPI .....</b>	<b>100</b>
12.1 概述.....	100
12.2 SPI 框图.....	101
12.3 功能描述 .....	102
12.4 SPI 寄存器 .....	104
12.4.1 SPI 控制寄存器.....	104
12.4.2 SPI 状态寄存器 .....	105
12.4.3 SPI 数据寄存器 .....	105
12.5 工作模式 .....	106
12.5.1 主机模式.....	106
12.5.2 从机模式 .....	106
12.6 时钟格式和数据传输.....	107
12.7 模式故障侦测 .....	107
12.8 写冲突错误.....	108
<b>13 触摸按键 (CDC) .....</b>	<b>109</b>
<b>14 模数转换器(ADC) .....</b>	<b>110</b>
14.1 概述.....	110
14.2 ADCON0 寄存器 .....	110
14.3 ADCON1 寄存器 .....	111
14.4 ADCON2 寄存器 .....	112
14.5 ADCON3 寄存器 .....	113
14.6 ADH/ADL AD 结果寄存器 .....	114
14.6.1 左对齐, ADFM = 0 .....	114
14.6.2 右对齐, ADFM = 1 .....	114

---

14.7 AD 转换时间 .....	114
<b>15 比较器 (CMP) .....</b>	<b>115</b>
15.1 概述.....	115
15.2 比较器框图.....	115
15.3 CMPC0 比较器控制寄存器 0.....	116
15.4 CMPC1 比较器控制寄存器 1.....	117
15.5 CMPC2 比较器控制寄存器 2.....	118
<b>16 LED .....</b>	<b>119</b>
16.1 概述.....	119
16.2 LED 控制寄存器.....	119
16.3 LED COM 使能寄存器.....	120
16.4 LED SEG 端口选择寄存器 .....	121
16.5 LED 显示缓存寄存器 .....	122
<b>17 看门狗 (WDT) .....</b>	<b>123</b>
17.1 概述.....	123
<b>18 芯片配置字 (OPTION BIT) .....</b>	<b>124</b>
<b>19 电性参数.....</b>	<b>126</b>
19.1 极限参数 .....	126
19.2 直流特性 .....	127
19.3 交流特性 .....	129
19.4 IO 口拉灌电流特性.....	130
19.5 系统时钟特性 .....	132
19.6 FLASH 特性 .....	135
19.7 ADC 电气特性 .....	136
<b>20 封装信息.....</b>	<b>137</b>
20.1 SOP16 .....	137
20.2 TSSOP20.....	138
20.3 SOP20.....	139
20.4 SOP28.....	140
20.5 SSOP28.....	141
<b>21 指令集简述 .....</b>	<b>142</b>
21.1 概述.....	142
21.2 符号说明 .....	142
21.3 M9 指令集表 .....	143
<b>22 修正记录.....</b>	<b>146</b>

# 1 产品简述

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，16K\*8 位 Flash 程序存储器，(503+30)\*8 位 RAM，最多 30 个双向 I/O 口，1 个 8 位 Timer 定时器/计数器，2 个 16 位 Timer 定时器/计数器，1 路 USART，1 路 SPI，1 路 IIC，4 路 8 位 PWM，2 路 16 位 PWM，28+3 路 12 位 AD 转换器，30 路触摸，LED 模块，比较器模块，支持 ISP 烧录，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
  - 高速 CPU
  - 15 级堆栈缓存器
  - 支持查表指令
- ROM
  - 16K\*8 位的 Flash 程序存储器
  - 可编程代码保护(可分区设置)
  - IAP/ISP 功能
- RAM
  - (512+30)\*8 位的数据存储器
- I/O 口
  - 最多 30 个双向 I/O 口
  - 所有端口可设置弱上拉下拉
  - 4 级驱动
  - 所有端口电平变化中断
- 多路中断
  - 2 个中断向量与 2 级优先级可选
  - 3 个定时器中断
  - 2 路可选电平外部中断
  - 支持所有端口电平变化中断
  - 其它外设中断
- 定时器 TC0
  - 8 位计数/16 位计数
  - PWM0: 4 路 8 位分辨率
  - PWM0: 可整组选择端口
  - 1-8 倍后分频
- 定时器 TC1
  - 16 位计数
  - 向上, 向下, 中心对齐计数模式
  - PWM1: 2 路 16 位分辨率
  - PWM1: 独立互补模式
  - PWM1: 可映射到任意端口
- 定时器 TC2
  - 16 位计数
  - 门控功能
- 看门狗定时器
- USART
  - 宽范围波特率
  - 支持半双工同步模式
  - 可映射到任意端口
- I2C
  - 主从模式
  - 7 位地址
  - 支持标准 (100kbps)
  - 可映射到任意端口
- SPI
  - 全双工模式
  - 主从模式
  - 多种波特率
- 系统时钟
  - 内部高速 RC 振荡器: 16 (±3%)
  - 内部低速 RC 振荡器: 64KHz/500KHz
  - 外部高速晶体振荡器: 4-24MHz
  - 外部低速晶体振荡器: 32.768KHz
- 系统工作模式
  - 普通模式: 高低速时钟同时工作
  - 低速模式: 仅低速时钟工作
  - 休眠模式: 高低速时钟都停止工作
- 28+3 路 12 位 ADC
  - 26 路外部输入
  - 1 路内部电源电压检测 VDD/4
  - 1 路内部参考
  - 1 路 GND



■ LED

- 驱动LED段码
- 最多8COM
- 所有的端口都可作为SEG口
- 亮度可调

■ 比较器

- 4个正相输入源
- 4个反相输入源
- 1路内部电源电压检测VDD/2
- 1路内部GND电压检测
- 1路可设内部参考电压检测

■ ISP烧录

■ CDC 触摸按键模块

- 最多 30 路通道
- 无需外接电容

■ 封装形式

- SOP16
- TSSOP20
- SOP20
- SOP28
- SSOP28

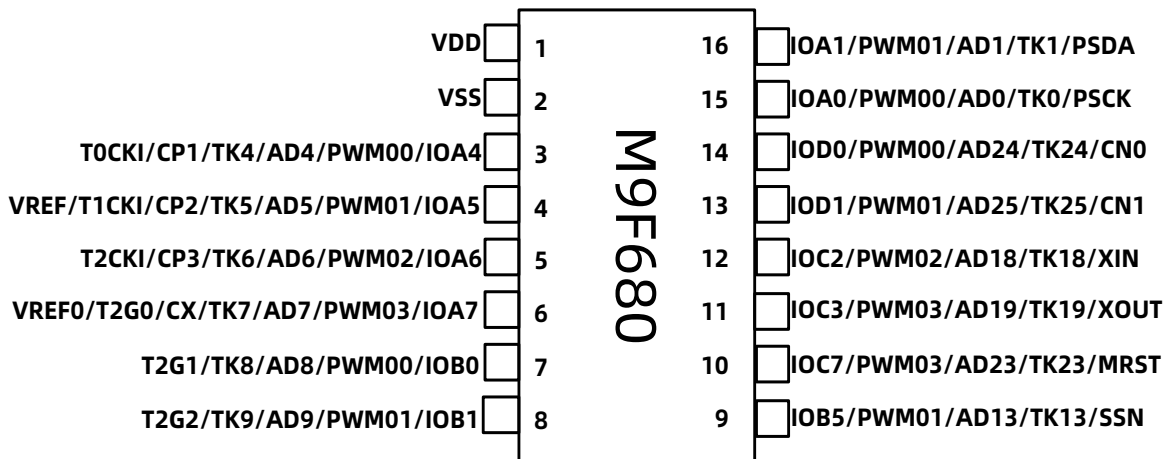
## 1.2 应用注意事项

- 1、CMP 的 CN 复用端口 CN0、CN1、CN2、CN3 (IOD0、IOD1、IOD2、IOD3) 有高于 VDD 和低于 GND 的电压输入时 (例: 测过零信号、蜂鸣器应用) 会干扰 CN 复用脚之间的 CMP 功能和触摸功能。
- 2、CMP 的 CP 复用端口 CP0、CP1、CP2、CP3 (IOA3、IOA4、IOA5、IOA6) 有高于 VDD 和低于 GND 的电压输入时 (例: 测过零信号、蜂鸣器应用) CMP 整个模块功能和触摸整个模块功能就会出现异常。
- 3、TK18 (IOC2) 和 TK19 (IOC3) 需要一起都做触摸功能, 或者都不做触摸功能, 不要一个脚使用触摸功能, 另外一个脚不使用触摸功能。
- 4、使用 UART 时要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 (烧录电压在 ZC\_Writer 软件 “烧录参数设置” 里面配置)。
- 5、使用 TC1 时, 注意跟其他 TC 中断有区别, TC1 中断应用时关总中断时记得先关低的总中断 (GIEL), 再关高的总中断 (GIEH)。
- 6、C 语言开发要注意中断程序里面需要调用一个空函数, 建议用我们的 DEMO 去开发功能。
- 7、EEPROM 的应用建议使用我们的 DEMO 来开发。
- 8、使用触摸时建议使用我们的触摸调试工具来配置阈值。
- 9、使用触摸TK19通道时注意, ADC默认通道为AD19, 如果ADC使能, 默认通道没修改会造成TK19通道异常。

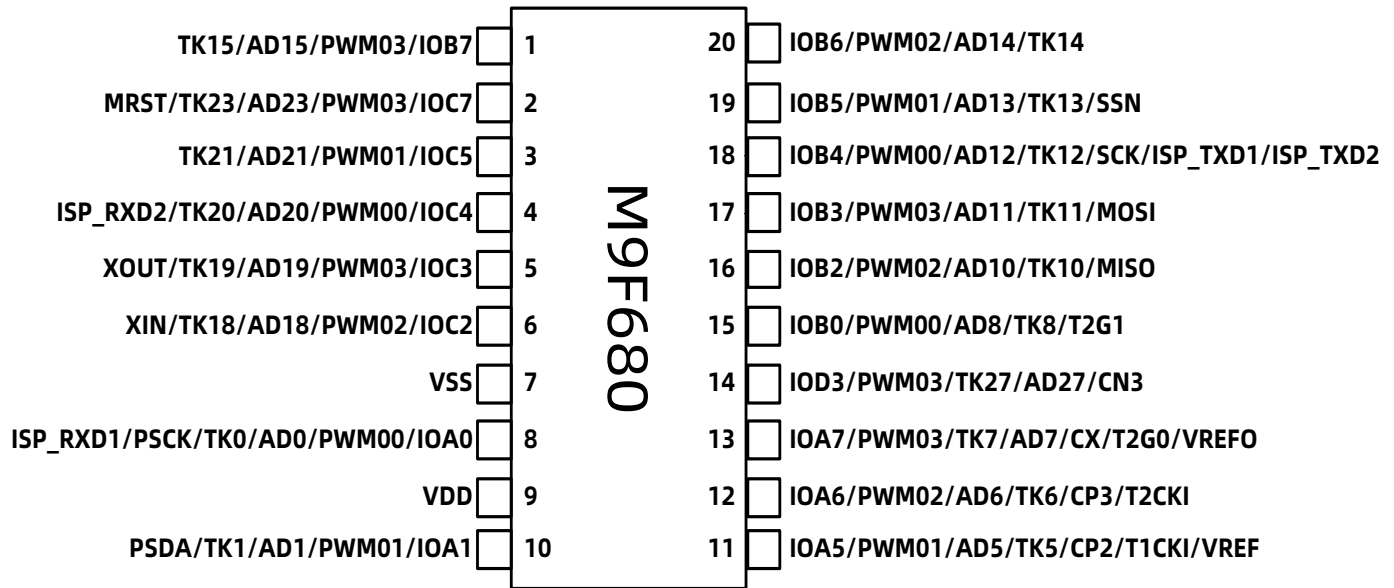
## 1.3 引脚图

注：芯片仿真烧录口分别是VDD、PSDA、PSCK、GND。

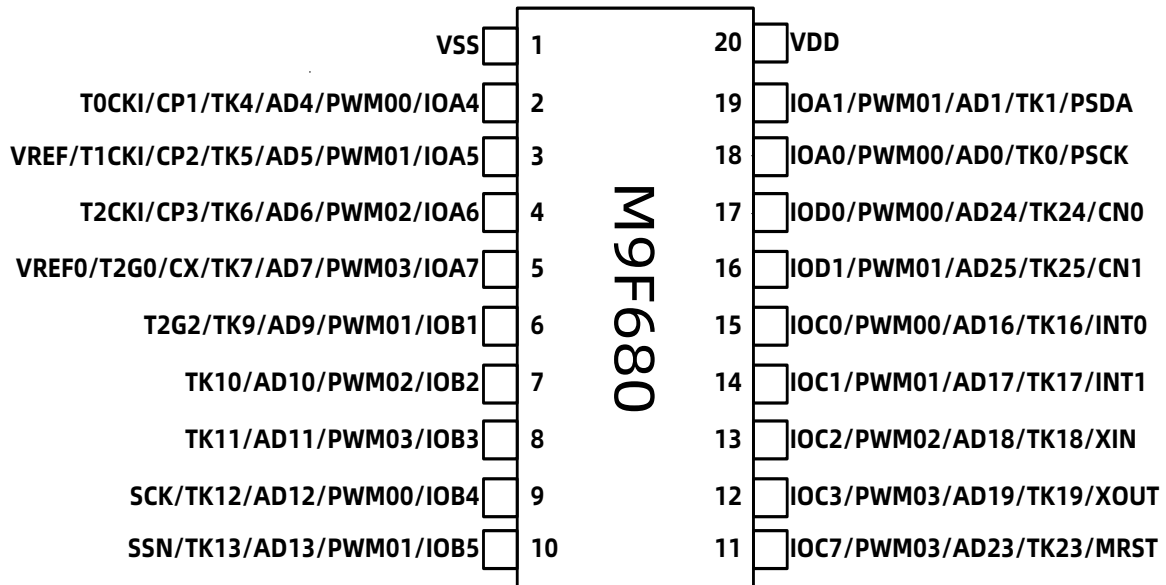
### 1.3.1 SOP16



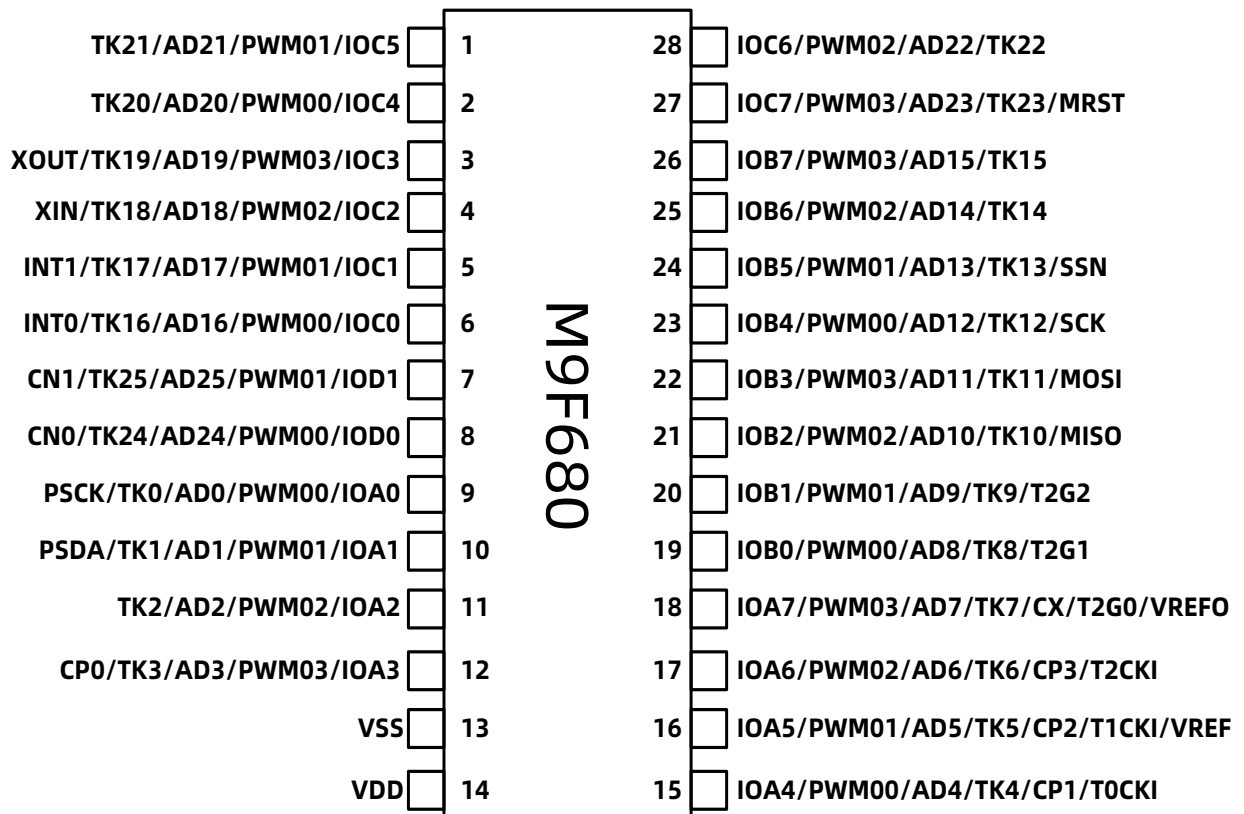
### 1.3.2 TSSOP20



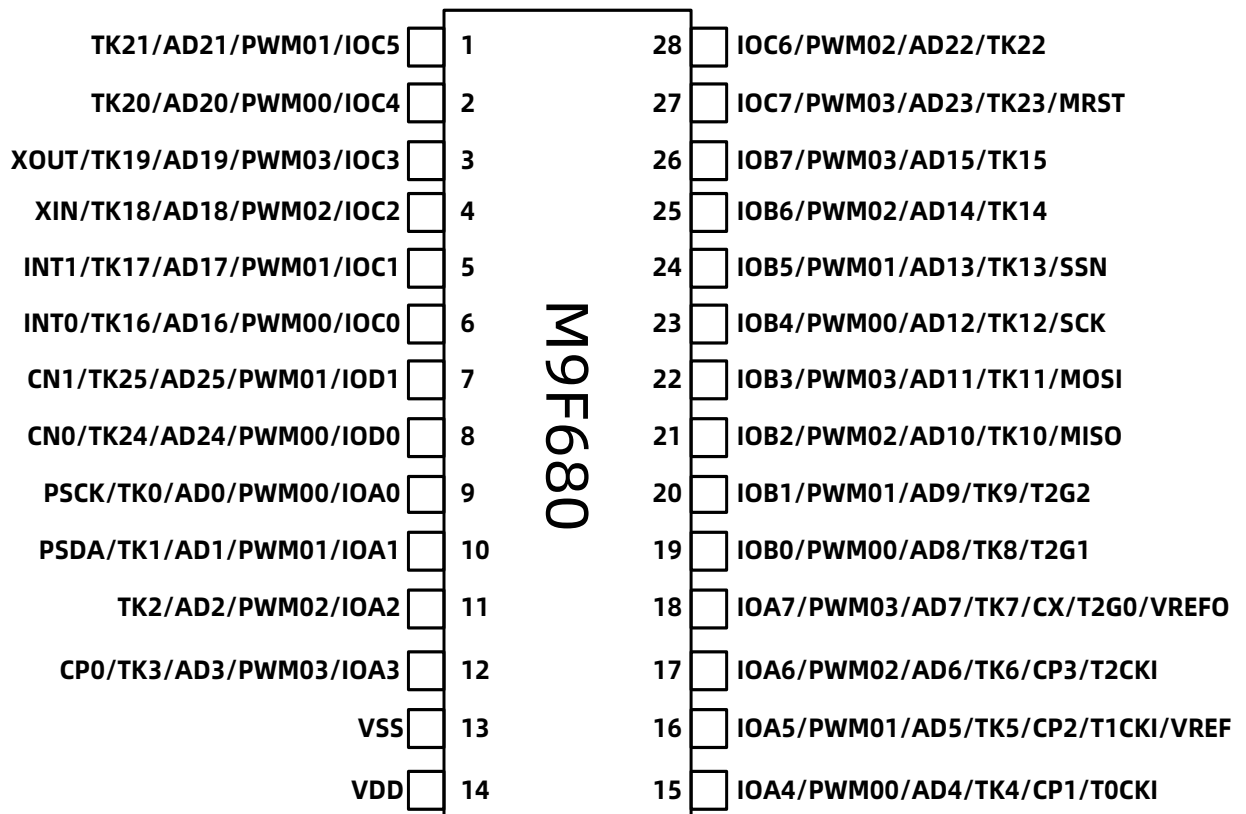
### 1.3.3 SOP20



### 1.3.4 SOP28



### 1.3.5 SSOP28



**注：**

- (1) 引脚图中未注明的 UASRT、I2C 与 PWM1 相关功能口可映射到任意端口上。
- (2) 引脚图中未注明的 LED COM0-7 可整体选择在某一组端口上。
- (3) PWM00/PWM01/PWM02/PWM03 只能整体在选择在端口高位或低位上。
- (4) TK18 和 TK19 通道需要一起都做触摸功能，或者都不做触摸功能，如果一个脚做触摸另一个脚不做触摸，会导致做触摸引脚被影响。
- (5) 以上可选端口功能控制介绍，详见端口 6.9 章节。

## 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/下拉电阻, 变化中断
PSCK	I	编程/仿真串行时钟输入
AD0	A	AD 通道 0
TK0	A	触摸按键通道 0
ISP_RXD1	I	ISP 下载 RXD1 口
IOA1	I/O	输入/输出 IO、SMT、上拉/下拉电阻
PSDA	I/O	编程/仿真串行数据
AD1	A	AD 通道 1
TK1	A	触摸按键通道 1
IOA2	I/O	输入/输出 IO、SMT、上拉/下拉电阻
AD2	A	AD 通道 2
TK2	A	触摸按键通道 2
IOA3	I/O	输入/输出 IO、SMT、上拉/下拉电阻
AD3	A	AD 通道 3
TK3	A	触摸按键通道 3
CP0	A	比较器 CMP 正相输入端口 0
IOA4	I/O	输入/输出 IO、SMT、上拉/下拉电阻
AD4	A	AD 通道 4
TK4	A	触摸按键通道 4
CP1	A	比较器 CMP 正相输入端口 1
T0CKI	I	TC0 外部时钟输入
IOA5	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD5	A	AD 通道 5
TK5	A	触摸按键通道 5
CP2	A	比较器 CMP 正相输入端口 2
T1CKI	I	TC1 外部时钟输入
VREF	A	AD 外部参考电压输入
IOA6	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD6	A	AD 通道 6
TK6	A	触摸按键通道 6
CP3	A	比较器 CMP 正相输入端口 3
T2CKI	I	TC2 外部时钟输入
IOA7	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD7	A	AD 通道 7
TK7	A	触摸按键通道 7
CX	O	比较器 CMP 输出
T2G0	I	TC2 门控信号输入
VREFO	O	VREF 电压输出(微弱驱动)



名称	类型	说明
IOB0	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD8	A	AD 通道 8
TK8	A	触摸按键通道 8
T2G1	I	TC2 门控信号输入
IOB1	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD9	A	AD 通道 9
TK9	A	触摸按键通道 9
T2G2	I	TC2 门控信号输入
IOB2	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD10	A	AD 通道 10
TK10	A	触摸按键通道 10
MISO	I/O	SPI 通讯数据口
IOB3	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD11	A	AD 通道 11
TK11	A	触摸按键通道 11
MOSI	I/O	SPI 通讯数据口
IOB4	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD12	A	AD 通道 12
TK12	A	触摸按键通道 12
SCK	I/O	SPI 通讯时钟口
ISP_TXD1	I	ISP 下载 TXD1 口
ISP_TXD2	I	ISP 下载 TXD2 口
IOB5	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD13	A	AD 通道 13
TK13	A	触摸按键通道 13
SSN	I	SPI 通讯片选口
IOB6	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD14	A	AD 通道 14
TK14	A	触摸按键通道 14
IOB7	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD15	A	AD 通道 15
TK15	A	触摸按键通道 15
IOC0	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD16	A	AD 通道 16
TK16	A	触摸按键通道 16
INT0	I	外部中断
IOC1	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD17	A	AD 通道 17
TK17	A	触摸按键通道 17
INT1	I	外部中断

名称	类型	说明
IOC2	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD18	A	AD 通道 18
TK18	A	触摸按键通道 18
XIN	A	外部晶体振荡器接口
IOC3	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD19	A	AD 通道 19
TK19	A	触摸按键通道 19
XOUT	A	外部晶体振荡器接口
IOC4	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD20	A	AD 通道 20
TK20	A	触摸按键通道 20
ISP_RXD2	I	ISP 下载 RXD2 口
IOC5	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD21	A	AD 通道 21
TK21	A	触摸按键通道 21
IOC6	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD22	A	AD 通道 22
TK22	A	触摸按键通道 22
IOC7	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD23	A	AD 通道 23
TK23	A	触摸按键通道 23
MRST	I	外部复位口(低电平有效, 内建上拉到 VDD)
IOD0	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD24	A	AD 通道 24
TK24	A	触摸按键通道 24
CN0	A	比较器 CMP 反相输入端口 0
IOD1	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD25	A	AD 通道 25
TK25	A	触摸按键通道 25
CN1	A	比较器 CMP 反相输入端口 1
IOD2	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD26	A	AD 通道 26
TK26	A	触摸按键通道 26
CN2	A	比较器 CMP 反相输入端口 2
IOD3	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
AD27	A	AD 通道 27
TK27	A	触摸按键通道 27
CN3	A	比较器 CMP 反相输入端口 3
IOD4	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
TK28	A	触摸按键通道 28
IOD5	I/O	输入/输出 IO、SMT、上拉电阻、下拉电阻
TK29	A	触摸按键通道 29

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
0x0002 ~ 0x0007	用户区
0x0008 ~ 0x0018	中断向量
0x0019 ~ 0x3FFF	用户区

#### 2.1.1 复位向量 (0000h)

M9F680 有以下五种复位方式：

- 上电复位
- 看门狗复位
- 外部复位
- 欠压复位
- 软件复位(软复位指令：SRESET)

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,0018h)

M9F680 有高低两种中断向量地址。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相应的中断向量地址开始执行中断服务程序。

使用触摸库时，低优先级中断被触摸暂用，其它中断不再能使用低向量地址。

### 例：中断服务程序

```

        ORG      0000h
        GOTO    Main_Program      ;//跳转到程序开始
        ORG      0008h
        GOTO    Interrupt_High    ;//发生高优先级中断后，跳转到中断子程序
        ORG      0018h
        GOTO    Interrupt_Low     ;//发生低优先级中断后，跳转到中断子程序
Main_Program:
        ...
Main:
        ...
        GOTO    Main              ;//主程序循环

Interrupt_High:
        ...
        RETIE

Interrupt_Low:
        ...
        RETIE

        END
    
```

## 2.1.3 程序计数器

### PCL 寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PCL	PCL[7:0]							
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

Bit [7:0] **PCL[7:0]**: 程序计数器指针低字节

用户将该 PCL 作为目的操作数做加法运算时 (ADDRA PCL、ADCRA PCL)，14 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。

**PCLATH 寄存器**

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PCLATH	-	-	PCLATH[5:0]					
读/写	-	-	R/W	R/W	R/W	R/W	R/W	R/W
复位后	-	-	0	0	0	0	0	0

Bit [5:0] **PCLATH[5:0]:** 程序计数器指针高字节锁存PC[13:8]

程序计数器 (Program Counter, PC) 指定欲取出执行的指令的地址。PC 为 14 位宽, 保存在两个不同的 8 位寄存器中。存储低字节的寄存器称为 PCL 寄存器, 该寄存器可读写。存储高字节的寄存器, 即 PCH 寄存器, 存储 PC[13:8]位; 该寄存器不可直接读写。更新 PCH 寄存器的操作是通过 PCLATH 寄存器实现的。PCLATH 的内容通过执行写 PCL 的任何操作被传送到程序计数器。PC 是按字节寻址程序存储器的。为了防止 PC 不能正确获取字指令, 需要将 PCL 的最低有效位固定取值为 0。PC 每次加 2 来寻址程序存储器中的顺序指令。CALL、RCALL、GOTO 和程序跳转指令直接写入程序计数器。对于这些指令, PCLATH 的内容不会传送到程序计数器。

## 2.1.4 堆栈

**STKPTR 寄存器**

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
STKPTR	STKUOV	STKDOV	-	-	STKPTR[3:0]			
读/写	R/W	R/W	-	-	R/W	R/W	R/W	R/W
复位后	0	0	-	-	0	0	0	0

Bit 7 **STKUOV:** 堆栈向上溢出

0 = 堆栈未向上溢出 (软件清零或复位清零)

1 = 堆栈上溢出

Bit 6 **STKDOV:** 堆栈向下溢出

0 = 堆栈未向下溢出 (软件清零或复位清零)

1 = 堆栈下溢出

Bit [3:0] **STKPTR[3:0]:** 堆栈指针

**TOSL 寄存器**

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TOSL	TOS[7:0]							
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

Bit [7:0] **TOSL[7:0]:** 栈顶寄存器低八位

**TOSH 寄存器**

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TOSH	-	-	-	TOS[12:8]				
读/写	-	-	-	R/W	R/W	R/W	R/W	R/W
复位后	-	-	-	0	0	0	0	0

Bit [4:0] **TOSH[4:0]**: 栈顶寄存器高五位

返回地址堆栈允许最多 15 个程序调用和中断的任意组合。当执行 CALL、RCALL 指令或响应中断时，PC 值会被压入该堆栈。而在执行 RETURN、RETIA 或 RETIE 指令时，PC 值会从堆栈弹出。PCLATH 不受 RETURN 或 CALL 指令的影响。通过 16 位的 RAM 和一个 4 位的堆栈指针 STKPTR 来实现 15 字的堆栈操作。堆栈既不占用程序存储空间，也不占用数据存储空间。堆栈指针是可读写的，并且通过栈顶（TOS）特殊文件寄存器可以读写栈顶地址。也可以使用这些寄存器将数据压入堆栈或者从堆栈弹出。

执行 CALL 类型指令进行压栈操作，首先堆栈指针加 1，并且将 PC（PC 已经指向 CALL 之后的下一条指令）的内容写入堆栈指针所指向的地址单元。执行 RETURN 类型指令进行出栈操作，STKPTR 寄存器所指向的地址单元的内容会被传送给 PC，然后堆栈指针减 1。所有复位后，堆栈指针均会初始化为 0000。堆栈指针值 0000 不指向任何 RAM 单元，它仅仅是一个复位值。状态位指示堆栈是已满、上溢还是下溢。

有两个寄存器{TOSH:TOSL}用于保存由 STKPTR 寄存器所指向的堆栈单元的内容。这可以让用户在必要时实现软件堆栈。在 CALL、RCALL 或中断后，软件可以通过读取{TOSH:TOSL}寄存器来读取压入堆栈的值。这些值可以被存放到由用户定义的软件堆栈。返回时，软件将这些值存回{TOSH:TOSL}并执行返回。为防止意外的堆栈操作，访问堆栈时用户必须禁止全局中断允许（GIE）位。

STKPTR 寄存器包含堆栈指针值、STKUOV（堆栈满）状态位和 STKDOV（堆栈下溢）状态位。堆栈指针值可为 0 至 15 范围内的任意值。向堆栈压入值前，堆栈指针加 1；而从堆栈弹出值后，堆栈指针减 1。复位时，堆栈指针值为零。用户可以读写堆栈指针的值。向堆栈压入 PC 值 16 次（且没有值从堆栈弹出）后，STKUOV 位置 1。通过软件或 POR 将 STKUOV 位清零。压栈操作会覆盖第 16 次压栈的值，并且 STKPTR 将保持为 15。当堆栈弹出次数足够卸空堆栈时，下一次出栈会向 PC 返回一个零值，并将 STKUOV 位置 1，而堆栈指针则保持为零。STKUOV 位将保持置 1，直到由软件清零或发生 POR 为止。

## 2.1.5 SPUSH 和 SPOP 指令

由于栈顶是可以读写的，因此将值压入堆栈或从堆栈弹出而不影响程序的正常执行是非常理想的。指令集包含两条指令 SPUSH 和 SPOP，使用这两条指令可在软件控制下对堆栈顶 TOS 执行操作。然后就可以修改 TOSH 和 TOSL，将数据或返回地址压入堆栈。SPUSH 指令将当前的 PC 值压入堆栈。执行该指令会使堆栈指针加 1 并将当前的 PC 值装入堆栈。SPOP 指令通过将堆栈指针减 1 来放弃当前的 TOS 值，然后前一个入栈的值就成为了 TOS 值。

## 2.1.6 快速寄存器堆栈

M9F680 为 STATUS、AREG 和 BSR 寄存器提供的快速寄存器堆栈具有从中断“快速返回”的功能。每个寄存器的堆栈只有一级且不可读写。当处理器转入中断向量处执行时，它装入对应寄存器的当前值。所有中断源都会将值压入堆栈寄存器。如果使用 RETIE, FAST 指令从中断返回，这些寄存器中的值就会被装回相关的寄存器。如果同时允许低优先级中断和高优先级中断，从低优先级中断返回时，无法可靠地使用堆栈寄存器。如果在为低优先级中断提供服务时，发生了高优先级中断，则低优先级中断存储在堆栈寄存器中的值将被覆盖。在为低优先级中断提供服务时，用户必须用软件保存关键寄存器的值。如果未使用中断优先级，所有中断都可以使用快速寄存器堆栈从中断返回。如果没有使用中断，快速寄存器堆栈可以用于在子程序调用结束后恢复 STATUS、AREG 和 BSR 寄存器。要在子程序调用中使用快速寄存器堆栈，必须执行 CALL label, FAST 指令将 STATUS、AREG 和 BSR 寄存器的内容存入快速寄存器堆栈。然后执行 RETURN, FAST 指令，从快速寄存器堆栈恢复这些寄存器。

## 2.1.7 查表

利用 RDT 指令可以读取程序区数据，TBLPTRH 和 TBLPTRL 组成 16 位表地址，读取数据到 TABLAT 寄存器。

### 例 1: 查找 ROM 地址为“DTAB”的值

MOVIA	HIGH(DTAB)	;获取数据表地址高位
MOVAR	TBLPTRH	;设置数据表高位指针
MOVIA	LOW(DTAB)	;获取数据表地址低位
MOVAR	TBLPTRL	;设置数据表低位指针
		;若需读取表的其它数据，修改指针
RDT*+		;读取表的第一个数据0x01，读取后指针+1
MOVR	TABLAT,0	;将数据0x01放在A
...		
DTAB:		
DB	0x01,0x02,0x03,0x04,0x05,0x06,0x07.....	

## 2.2 数据存储器

### 2.2.1 数据存储器结构

数据存储器是用静态 RAM 实现的。在数据存储器中，每个寄存器都有 12 位地址，允许数据存储器实现为最大 4096 个字节。存储空间最多被分为 16 个存储区，每个存储区包含 256 个字节。数据存储器由特殊功能寄存器 (Special Function Register, SFR) 和通用寄存器 (General Purpose Register, GPR) 组成。SFR 用于单片机和外设功能模块的控制和状态指示，GPR 则用于用户应用程序的数据存储和中间结果暂存。任何未实现的存储单元均读为 0。指令集和架构支持跨所有存储区的操作。可以通过直接、间接寻址模式访问整个数据存储器。确保能在一个周期中访问常用寄存器 (SFR 和某些 GPR)，本器件实现了一个快速操作存储区。该存储区是一个 256 字节的存储空间，它可实现对 SFR 和 GPR Bank 0 的低地址单元的快速访问，而无需使用存储区选择寄存器 (Bank Select Register, BSR)。

### 2.2.2 BSR 页面选择寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BSR	-	-	-	-	BSR[3:0]			
读/写	-	-	-	-	R/W	R/W	R/W	R/W
复位后	-	-	-	-	0	0	0	0

Bit [3:0] **BSR[3:0]:** 直接寻址的页面选择

容量较大的数据存储器需要高效的寻址机制，以便对所有地址进行快速访问。理想状况下，这意味着不必为每次读写操作提供完整地址。本器件是使用 RAM 存储区分区机制实现快速访问的。这种机制将存储空间分成连续的 16 个 256 字节的存储区。根据不同的指令，可以通过完整的 12 位地址，或通过 8 位的低字节地址和 4 位存储区指针直接寻址每个存储单元。指令集中的大部分指令都使用存储区指针，也就是存储区选择寄存器 (BSR)。BSR 保存单元地址的高 4 位，而指令本身则包括单元地址的低 8 位。只使用 BSR 的低 4 位 (BSR[3:0])，不使用高 4 位；高 4 位总是读为 0 且不能被写入。可以通过使用 BANKBSR 指令直接装入 BSR。BSR 的值代表数据存储器中的存储区；指令中的 8 位指向存储区中的存储单元，可以将它看作距离存储区下边界的偏移量。由于最多可有 16 个寄存器共享同一个低位地址，用户必须非常小心以确保在执行数据读或写之前选择了正确的存储区。

当选择存储区时，只有已实现的存储区才可以读写。对未实现的存储区进行的写操作将被忽略，而读这些存储区会返回 0。虽然是这样，这些操作仍然会对 STATUS 寄存器起作用，就好像操作成功了一样。指令中只有 MOVRR 指令指定源寄存器和目标寄存器的完整 12 位地址。该指令在执行时完全忽略 BSR。所有其他指令仅包含作为操作数的低位地址，而且必须使用 BSR 或快速操作存储区来寻址目标寄存器。



### 2.2.3 快速操作寄存器

使用 BSR 和嵌入的 8 位地址，用户可以寻址数据存储器的整个空间，但这同时也意味着用户必须始终确保选择了正确的存储区。否则可能会从错误的单元读取数据或将数据写入错误的单元。如果本来是向 GPR 进行写操作，却将结果写入了 SFR，后果是非常严重的。但是在每次对数据存储器进行读或写操作时验证或更改 BSR 会严重影响工作效率。为了提高访问大多数常用数据存储单元的效率，现为数据存储器配置了快速操作存储区，这样可以允许用户访问被映射的存储区而无需指定 BSR。快速操作存储区由 Bank 0 的前 96 个字节 (00h-5Fh) 和 Bank 15 的后 160 个字节 (60h-FFh) 组成。地址较低的部分被称为“快速操作 RAM”，由 GPR 组成。地址较高的部分则被映射为器件的 SFR。这两个区域被连续地映射到快速操作存储区并且可以用一个 8 位地址进行线性寻址。包括快速操作 RAM 位（指令中的“a”参数）的指令使用快速操作存储区。当“a”等于 1 时，指令使用 BSR 和包含在操作码中的 8 位地址对数据存储器寻址。当“a”为 0 时，强制指令使用快速操作存储区地址映射，此时完全忽略 BSR 的当前值。此“强制”寻址模式可使指令在一个周期内对数据地址进行操作，而不需要首先更新 BSR。这意味着用户可以更高效地对 8 位地址为 60h 或以上的 SFR 进行取值和操作。地址为 60h 以下的快速操作 RAM 非常适合于存储那些用户可能需要快速访问的数据值，如直接计算结果或常用程序变量。快速操作 RAM 也可实现更加快速和高效的现场保护和变量切换代码。

## 2.2.4 数据存储寻址模式

本器件主要有三种寻址模式：立即数寻址，直接寻址，间接寻址。

立即数寻址：主要是 MOVRR 指令时的寻址。

直接寻址：给出八位地址，根据 BSR 或者快速操作寄存器确定地址。

间接寻址：有三组 12 位数据指针，FSR0，FSR1，FSR2。

寻址模式	地址组成
立即数寻址	INST[11:0]
直接寻址	BSR0[4:0], INST[7:0]
间接寻址 0	FSR0H, FSR0L
间接寻址 1	FSR1H, FSR1L
间接寻址 2	FSR2H, FSR2L

页	地址	用途	说明
BANK0	0x000-0x001	SFR	快速操作寄存器
	0x002-0x008	Access RAM	未实现，禁止使用
			未实现，禁止使用
	0x009-0x05F	Access RAM	快速操作寄存器
BANK1	0x060-0x0FF	GPR	-
	0x100-0x1BF	GPR	-
BANK2	0x1C0-0x1FF	GPR <sup>(1)</sup>	Flash RAM
	0x200-0x21D	GPR <sup>(2)</sup>	LED RAM
	0x21E-0x2FF	-	未实现，读为 0
BANK3	0x300-0x3FF	-	未实现，读为 0
BANK4	0x400-0x4FF	-	未实现，读为 0
BANK5	0x500-0x5FF	-	未实现，读为 0
BANK6	0x600-0x6FF	-	未实现，读为 0
BANK7	0x700-0x7FF	-	未实现，读为 0
BANK8	0x800-0x8FF	-	未实现，读为 0
BANK9	0x900-0x9FF	-	未实现，读为 0
BANK10	0xA00-0xAFF	-	未实现，读为 0
BANK11	0xB00-0xBFF	-	未实现，读为 0
BANK12	0xC00-0xCFF	-	未实现，读为 0
BANK13	0xD00-0xDFF	-	未实现，读为 0
BANK14	0xE00-0xEFF	-	未实现，读为 0
BANK15	0xF00-0xF5F	-	未实现，读为 0
	0xF60-0xFFFF	SFR	快速操作寄存器

注：

(1) 表示此 GPR 地址被 Flash 写入缓存器占用。

(2) 表示此 GPR 地址被 LED 显示缓存器占用。

## 2.2.5 间接寻址 0/1/2

**FSRxL 寄存器**

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
FSRxL	FSRxL[7:0]							
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

Bit [7:0] **FSRxL[7:0]**: FSRx 寄存器低七位地址 (x = 0/1/2)

**FSRxH 寄存器**

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
FSRxH	-	-	-	-	FSRxH[3:0]			
读/写	-	-	-	-	R/W	R/W	R/W	R/W
复位后	-	-	-	-	0	0	0	0

Bit [3:0] **FSRxH[3:0]**: FSR0 寄存器高四位地址 (x = 0/1/2)

**注: INDFx, POSTINCx, POSTDECx, PREINCx, PLUSWx 不是实际存在的寄存器, 对这些寄存器的操作实际上是对以 FSRx 为地址的寄存器的操作。(x = 0/1/2, 下同)**

访问 INDFx 寄存器时, 实现间接寻址 x, 访问到的是{FSRxH:FSRxL}寄存器值作为数据指针所指向的寄存器内容, 间接寻址模式 x 可寻址所有页面空间。

访问 POSTINCx 寄存器时, 实现间接寻址模式 x, 访问到的是{FSRxH:FSRxL}寄存器值作为数据指针所指向的寄存器内容, 访问后, FSRx 的值自加一, 间接寻址模式 x 可寻址所有页面空间。

访问 POSTDECx 寄存器时, 实现间接寻址模式 x, 访问到的是{FSRxH:FSRxL}寄存器值作为数据指针所指向的寄存器内容, 访问后, FSRx 的值自减一, 间接寻址模式 x 可寻址所有页面空间。

访问 PREINCx 寄存器时, 实现间接寻址模式 x, 访问前, FSRx 的值自加一, 访问到的是{FSRxH:FSRxL}寄存器值作为数据指针所指向的寄存器内容, 间接寻址模式 x 可寻址所有页面空间。

访问 PLUSWx 寄存器时, 实现间接寻址模式 x, 访问到的是{FSRxH:FSRxL}+AREG 寄存器值作为数据指针所指向的寄存器内容, 间接寻址模式可寻址所有页面空间。

## 2.2.6 8x8 硬件乘法器

本器件包含一个8\*8 硬件乘法器（是ALU的一部分）。该乘法器可在一个指令周期内执行无符号运算并产生一个16位运算结果，该结果存储在—对乘积寄存器[PRODH:PRODL]中。该乘法器执行的运算不会影响STATUS 寄存器中的任何标志。

### PRODL 寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PRODL	PRODL[7:0]							
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

Bit [7:0] **PRODL[7:0]:** 乘法结果寄存器低八位

### PRODH 寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PRODH	PRODH[7:0]							
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

Bit [7:0] **PRODH[7:0]:** 乘法结果寄存器高八位

## 2.2.7 STATUS 状态寄存器

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

Bit 4 **N:** 负标志，此位用于有符号的算术运算（以二进制补码方式进行），它可以表示结果是否为负（ALU MSB = 1）

0 = 运算结果为正

1 = 运算结果为负

Bit 3 **OV:** 溢出标志，此位用于有符号的算术运算（以二进制补码方式进行），它表明运算结果溢出了7位二进制数的，范围溢出导致符号位（bit 7）发生改变。

0 = 运算未溢出

1 = 运算结果溢出

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，减法时有向高四位借位

1 = 加法运算时低四位有进位，减法时没有向高四位借位

Bit 0 **C:** 进位标志

0 = 加法运算后没有进位，减法时有借位

1 = 加法运算后有进位，减法时没有借位

## 2.2.8 系统寄存器定义

数据寄存器映射表 (F60h - FFFh)									
地址	寄存器	地址	寄存器	地址	寄存器	地址	寄存器	地址	寄存器
FFFh	-	FDfH	INDF2	FBfH	ANSD	F9fH	-	F7fH	I2CCR
FFEh	TOSH	FDEh	POSTINC2	FBEh	ANSC	F9Eh	-	F7Eh	I2CSTA
FFDh	TOSL	FDDh	POSTDEC2	FBDh	ANSB	F9Dh	T0CR2	F7Dh	I2CDATA
FFCh	STKPTR	FDCh	PREINC2	FBCh	ANSA	F9Ch	T0CR	F7Ch	I2CADR
FFBh	-	FDBh	PLUSW2	FBBh	PUD	F9Bh	TC0CH	F7Bh	SPICR
FFAh	PCLATH	FDAh	FSR2H	FBAh	PUC	F9Ah	TC0CL	F7Ah	SPISTA
FF9h	PCL	FD9h	FSR2L	FB9h	PUB	F99h	PWM00CR	F79h	SPIDATA
FF8h	-	FD8h	STATUS	FB8h	PUA	F98h	PWM00D	F78h	-
FF7h	TBLPTRH	FD7h	INTCR1	FB7h	PDD	F97h	PWM01CR	F77h	TKCTR5
FF6h	TBLPTRL	FD6h	INTCR0	FB6h	PDC	F96h	PWM01D	F76h	TKCTR4
FF5h	TABLAT	FD5h	INTF1	FB5h	PDB	F95h	PWM02CR	F75h	TKCTR3
FF4h	PRODH	FD4h	INTF0	FB4h	PDA	F94h	PWM02D	F74h	TKCTR2
FF3h	PRODL	FD3h	INTP1	FB3h	IODICR	F93h	PWM03CR	F73h	TKCTR1
FF2h	OPTION	FD2h	INTP0	FB2h	IOCICR	F92h	PWM03D	F72h	TKCTR0
FF1h	EECON1	FD1h	TX0CR	FB1h	IOBICR	F91h	T1CR	F71h	TKCHS3
FF0h	EECON2	FD0h	RX0CR	FB0h	IOAICR	F90h	TC1PRH	F70h	TKCHS2
FEFh	INDF0	FCfH	BRGD0H	FAfH	IODODS1	F8fH	TC1PRL	F6fH	TKCHS1
FEEh	POSTINC0	FCEh	BRGD0L	FAEh	IODODS0	F8Eh	TC1CH	F6Eh	TKCHS0
FEDh	POSTDEC0	FCDh	TX0REG	FADh	IOCODS1	F8Dh	TC1CL	F6Dh	TKCHTH
FECh	PREINC0	FCCh	RX0REG	FACH	IOCODS0	F8Ch	PWM1CR	F6Ch	TKCHTL
FEBh	PLUSW0	FCBh	IOD	FABh	IOBODS1	F8Bh	PWM10DH	F6Bh	TKRCTR
FEAh	FSR0H	FCAh	IOC	FAAh	IOBODS0	F8Ah	PWM10DL	F6Ah	TKCCTR
FE9h	FSR0L	FC9h	IOB	FA9h	IOAODS1	F89h	PWM11DH	F69h	T2CR
FE8h	AREG	FC8h	IOA	FA8h	IOAODS0	F88h	PWM11DL	F68h	TC2GCR
FE7h	INDF1	FC7h	OED	FA7h	ADCON3	F87h	CMPC2	F67h	TC2PRH
FE6h	POSTINC1	FC6h	OEC	FA6h	ADCON2	F86h	CMPC1	F66h	TC2PRL
FE5h	POSTDEC1	FC5h	OEB	FA5h	ADCON1	F85h	CMPC0	F65h	TC2CH
FE4h	PREINC1	FC4h	OEA	FA4h	ADCON0	F84h	IODIPS	F64h	TC2CL
FE3h	PLUSW1	FC3h	IODOR	FA3h	ADH	F83h	IOCIPS	F63h	FLIPCR
FE2h	FSR1H	FC2h	IOCOR	FA2h	ADL	F82h	IOBIPS	F62h	INTCR2
FE1h	FSR1L	FC1h	IOBOR	FA1h	-	F81h	IOAIPS	F61h	INTF2
FE0h	BSR	FC0h	IOAOR	FA0h	OSCM	F80h	PCON	F60h	INTP2

注：空白处为未实现地址，读为 0，无法写。

数据寄存器映射表 (E0h - F5Fh)									
地址	寄存器	地址	寄存器	地址	寄存器	地址	寄存器	地址	寄存器
F5Fh	MPTX	F3Fh	-	F1Fh	LEDCR1	EFFh	IRCCAL	EDFh	-
F5Eh	MPRX	F3Eh	-	F1Eh	LEDCR0	EFEh	IRCCAH	EDEh	-
F5Dh	MPSDA	F3Dh	-	F1Dh	COMIOS	EFDh	-	EDDh	-
F5Ch	MPSCL	F3Ch	-	F1Ch	MPCOM	EFCh	-	EDCh	-
F5Bh	MPPWM0	F3Bh	-	F1Bh	SEGIOSD	EFBh	-	EDBh	-
F5Ah	MPPWM10	F3Ah	-	F1Ah	SEGIOSC	EFAh	-	EDAh	-
F59h	MPPWM11	F39h	-	F19h	SEGIOSB	EF9h	CALLOCK	ED9h	-
F58h	-	F38h	-	F18h	SEGIOSA	EF8h	-	ED8h	-
F57h	-	F37h	-	F17h	-	EF7h	-	ED7h	-

数据寄存器映射表 (200h - 21Dh)									
地址	寄存器	地址	寄存器	地址	寄存器	地址	寄存器	地址	寄存器
200h	LEDDS00	206h	LEDDS06	20Ch	LEDDS12	212h	LEDDS18	218h	LEDDS24
201h	LEDDS01	207h	LEDDS07	20Dh	LEDDS13	213h	LEDDS19	219h	LEDDS25
202h	LEDDS02	208h	LEDDS08	20Eh	LEDDS14	214h	LEDDS20	21Ah	LEDDS26
203h	LEDDS03	209h	LEDDS09	20Fh	LEDDS15	215h	LEDDS21	21Bh	LEDDS27
204h	LEDDS04	20Ah	LEDDS10	210h	LEDDS16	216h	LEDDS22	21Ch	LEDDS28
205h	LEDDS05	20Bh	LEDDS11	211h	LEDDS17	217h	LEDDS23	21Dh	LEDDS29

数据寄存器映射表 (000h/0001h)									
地址	寄存器	地址	寄存器	地址	寄存器	地址	寄存器	地址	寄存器
-	-	-	-	-	-	001h	FLASHLOCK	000h	INTPTC1

注：空白处为未实现地址，读为 0，无法写。

## 2.3 Flash 自编程 (IAP)

Flash 编程只能通过页写操作，即 Flash 写操作每次都是对一个页面的地址进行写操作，每一页的大小为 64 个字节。读为自由地址读取，可直接指针操作。

### 2.3.1 EECON1 寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
EECON1	PRGT4	PRGT3	PRGT2	PRGT1	PRGT0	CLRPL	ERASE	WRITE
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	1	0	0	0	0	0	0

Bit [7:3] **PRGT[4:0]**: 必须保持为01000b

Bit 2 **CLRPL**: IAP清除Flash内部缓冲区操作，IAP写入操作必须置1

0 = 不清除Flash内部缓冲区

1 = 清除Flash内部缓冲区

Bit 1 **ERASE**: IAP擦除操作

0 = 不允许IAP擦除操作

1 = 允许IAP擦除操作

Bit 0 **WRITE**: IAP写入操作

0 = 不允许IAP写入操作

1 = 允许IAP写入操作

### 2.3.2 EECON2 寄存器

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

Bit 7 **WERR**: Flash写操作出错标志，写EECON1时该位自动置0。

Bit 3 **EELOCK3**: 解锁流程1

Bit 2 **EELOCK2**: 解锁流程2

Bit 1 **EELOCK1**: 解锁流程3

### 2.3.3 Flash 写入缓存器

数据存储器 BANK1 中的 0x1C0 - 0x1FF 为 Flash 写入缓存器，操作方式与普通 RAM 方式相同。Flash 写入缓存器暂用通用寄存器的 RAM。

### 2.3.4 Flash 操作示例

#### 例：擦 Flash 操作

BCLR	OPTION,GIE	;操作前要关中断
BSET	EECON1,1	;擦除Flash使能
MOVR	TBLPTRHBUF,A	;写高位地址
MOVAR	TBLPTRH	
MOVR	TBLPTRLBUF,A	;写低位地址
MOVAR	TBLPTRL	
BSET	EECON2,3	;Flash解锁流程1
BSET	EECON2,2	;Flash解锁流程2
BSET	EECON2,1	;Flash解锁流程3
WDT*		;写入，CPU暂停，擦除结束后再运行下一条指令
NOP		
...		

#### 例：Flash页写操作

BCLR	OPTION,GIE	;操作前要关中断
MOVIA	DATA0	
MOVAR	FLASHBUF+0	
...		
MOVIA	DATA63	
MOVAR	FLASHBUF+63	;写Flash页面缓冲区64个字节
BSET	EECON1,2	;清缓存区
BSET	EECON1,0	;写Flash使能
MOVIA	HIGH(DTAB)	;写Flash页高位地址
MOVAR	TBLPTRH	
MOVIA	LOW(DTAB)	;写Flash页低位地址
MOVAR	TBLPTRL	
BSET	EECON2,3	;Flash解锁流程1
BSET	EECON2,2	;Flash解锁流程2
BSET	EECON2,1	;Flash解锁流程3
WDT*		;写入，CPU暂停，写入结束后再运行下一条指令
NOP		
...		;必要时软件校验是否写入成功

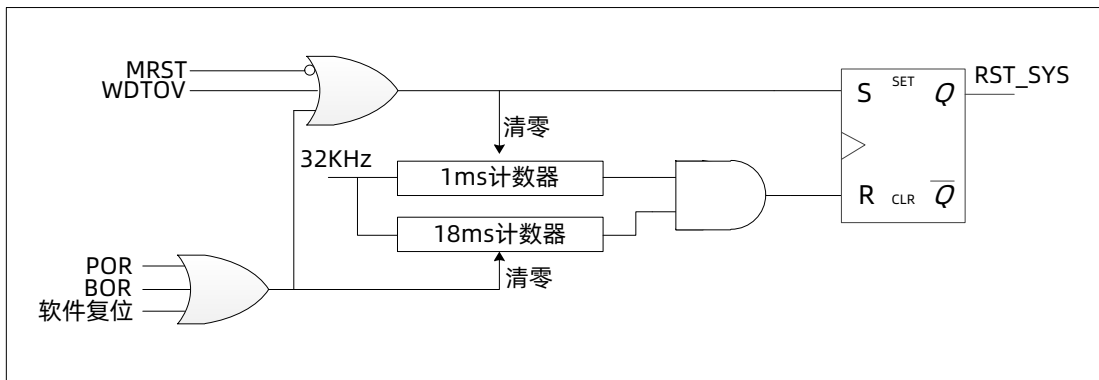


## 3 复位

### 3.1 复位方式

- 上电复位 (POR) (复位时间 5V 供电下为 20ms 左右, 3V 供电下 35ms 左右)
- 外部复位 (MCLR Reset)
- 欠压复位 (BOR)
- 看门狗定时器复位 (WDT Reset)
- 软件复位(软复位指令: SRESET)

M9F680有以上5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或3FF8h (由配置字决定) 处开始运行, 并且使系统寄存器值复位, 默认从0000h开始运行。



### 3.2 PCON 复位状态寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PCON	-	SOFTRST	TO	PD	-	-	POR	BOR
读/写	-	R/W	R	R	-	-	R/W	R/W
复位后	-	x	1	1	-	-	0	1

- Bit 6      **SOFTRST:** 软复位  
0 = 上电复位、掉电复位, 软件清零  
1 = 发生软复位
- 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
- Bit 1      **POR:** 上电复位位  
0 = 发生上电复位  
1 = 未发生上电复位, 软件置1
- Bit 0      **BOR:** 欠压复位位  
0 = 发生欠压复位  
1 = 未发生欠压复位, 软件置1

# 4 系统时钟

## 4.1 概述

M9F680支持双时钟系统：高速时钟和低速时钟。高速时钟由外部高速晶体振荡器（HXT）或内置高速RC振荡器（HIRC，16MHz）提供；低速时钟由外部低速晶体振荡器（LXT，如：32768Hz）或内置的低速RC振荡器（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速系统源可根据芯片配置字进行配置。

**注：**

- (1) 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。
- (2) 使用触摸功能时，高速时钟只能选择内部 HIRC 16MHz 时钟。

## 4.2 OSCM 寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
OSCM	STBH	STBL	-	STOP	CLKM	STPH	LIRC500K	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      **Reserved**: 保留，请保持为0
- Bit 4      **STOP**: CPU工作状态标志位  
             0 = CPU正常工作，所有复位唤醒  
             1 = CPU停止工作
- 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      **LIRC500K**: 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

### 4.3.1 普通模式

普通模式有两种分别是：

- (1) 系统时钟选择高频时钟源，STOP = 0。(电流特性参考电性参数表 I<sub>DD1</sub>、I<sub>DD2</sub>)
- (2) 系统时钟选择低频时钟源，STOP = 0。

### 4.3.2 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，STOP = 1。(电流特性参考电性参数表 I<sub>SP1</sub>)
- (2) 高频时钟停止，低频时钟工作，STOP = 1。(电流特性参考电性参数表 I<sub>SP2</sub>)

绿色模式可以由所有中断或 WDT 唤醒，此模式下只停止 CPU，外设依旧可以运行（外设所选时钟需开启）。

### 4.3.3 休眠模式

休眠模式只有一种是：

- (1) 高频时钟停止，低频时钟停止，STOP = 1。(电流特性参考电性参数表 I<sub>SP3</sub>)  
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	-	IRCCAL[6:0]						
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

24MHz 高频内部 RC 振荡频率校准寄存器。复位后，初值为出厂校准值。

## 4.5 IRCCAHA 寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IRCCAHA	-	IRCCAHA[6:0]						
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

16MHz 高频内部 RC 振荡频率校准寄存器。复位后，初值为出厂校准值。

内置的双高频 RC 振荡器在芯片出厂后，频率将校准到 16MHz 和 24MHz，IRCCAHA 和 IRCCAL 复位初值即为出厂校准值，但程序中可以通过特殊的写入流程来微调此频率以满足特定应用需求。

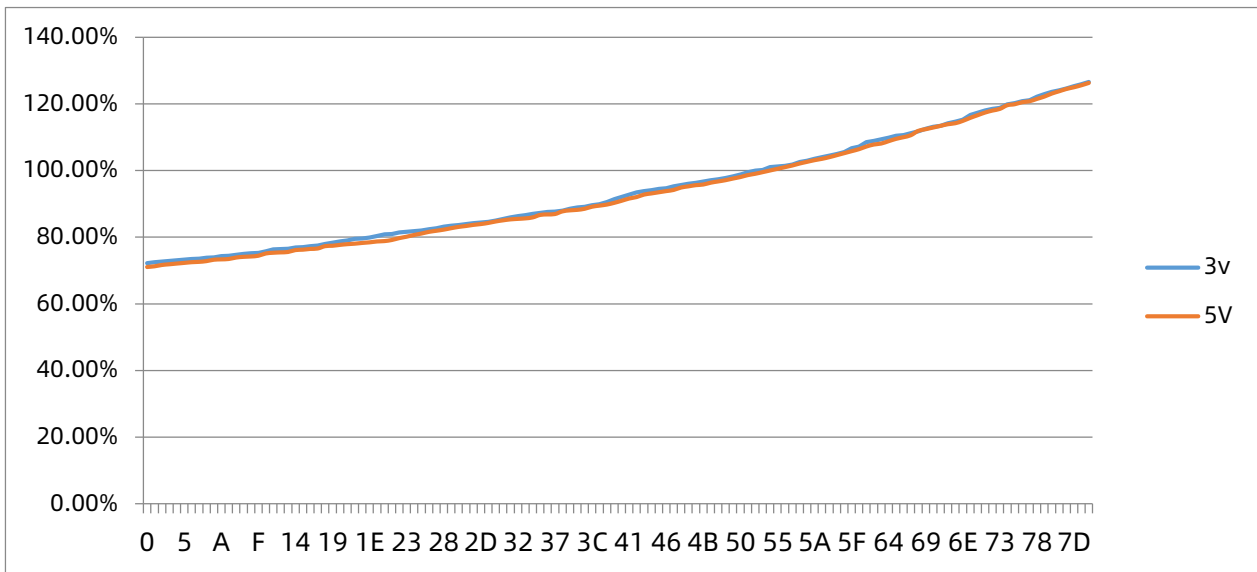
### 例：调整 16MHz 高频内部 IRC 频率

```

TASK_IRCCAHA:
    BANKBSR    CAL_BANKSEL
    MOVIA     0x55
    MOVAR     CALLOCK           ;// CALLOCK写入0x55
    MOVIA     0xAA
    MOVAR     CALLOCK           ;// CALLOCK写入0xAA
    MOVIA     VALUE
    MOVAR     IRCCAHA           ;//写入IRCCAHA
    ...
    
```

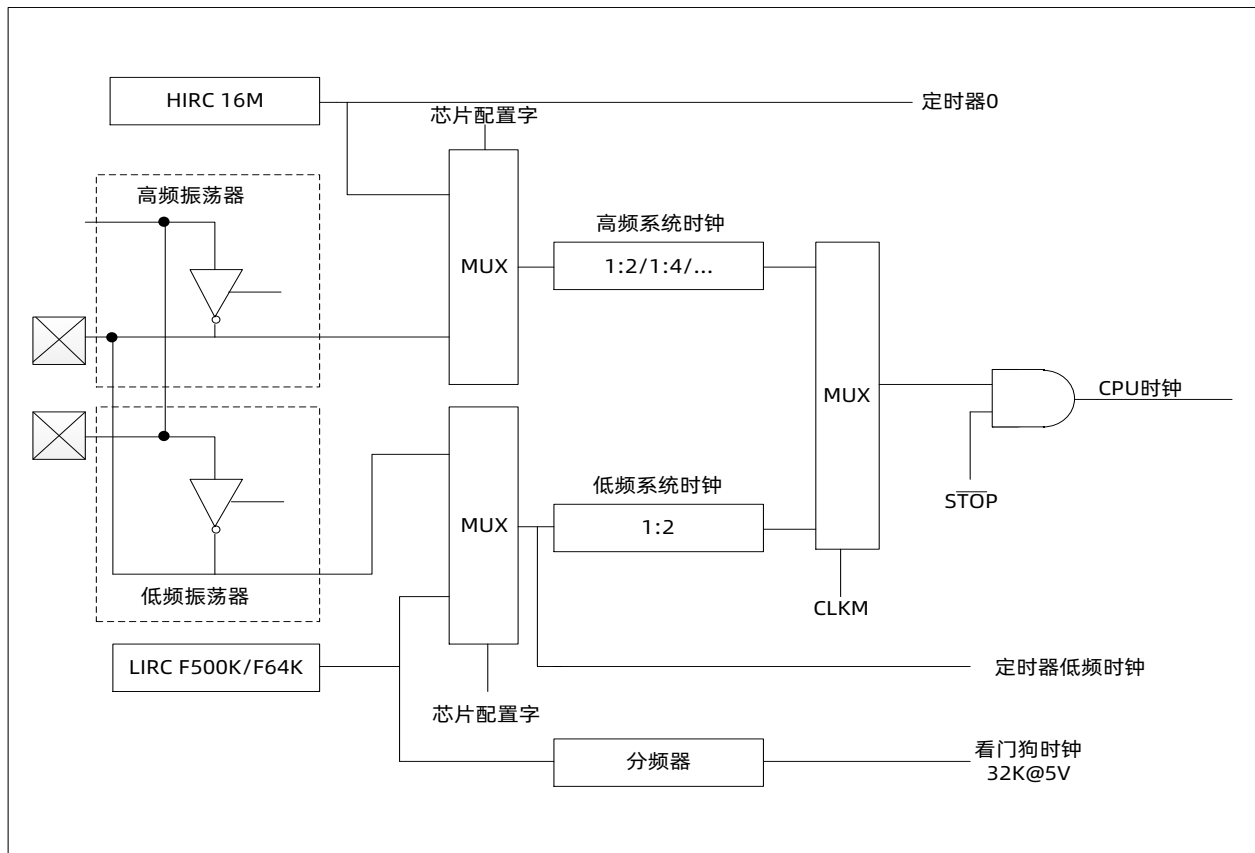
;//若需继续在IRCCAHA寄存器内写入其他值需要重复以上所有步骤

HIRC 调整频率趋势图



注：具体值不做设计保证。

## 4.6 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	TXEN	若选择高速时钟, 需 STPH=0	高速时钟源&STPH=0 低速时钟源&STPL=0

## 4.7 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，唤醒时间 = 高频振荡器起振时间+高频振荡器稳定时间

CLKM = 0&STBH = 1，唤醒时间 = 高频振荡器稳定时间

CLKM = 1&STBL = 0，唤醒时间 = 低频振荡器起振时间+低频振荡器稳定时间

CLKM = 1&STBL = 1，唤醒时间 = 低频振荡器稳定时间



# 5 中断

## 5.1 概述

M9F680可通过配置字选择位选择单优先级还是多优先级，单优先级是高优先级，所有中断默认都是高优先级。多优先级有两种优先级，高优先级和低优先级，通过INTPn寄存器进行控制。同级及高级优先级中断响应时不允许低级优先级中断嵌套，低级优先级中断允许高级优先级中断嵌套响应。

响应低优先级中断时，GIEL 会自动清零，同时将 GIEH/GIEL 状态压栈，执行中断时由于 GIEL 已被清零不允许响应同级中断，若 GIEH 的值是 1，则允许高优先级的中断嵌套；响应高优先级时，GIEH 和 GIEL 会被自动清零，同时将 GIEH/GIEL 状态压栈。

所有中断信号都可以唤醒 CPU（在不使能全局中断的情况下也可以唤醒）。

## 5.2 中断向量分配

高优先级中断：0008h

低优先级中断：0018h

### 5.3 OPTION 配置寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
OPTION	GIEH	GIEL	-	-	MINT1[1:0]		MINT0[1:0]	
读/写	R/W	R/W	-	-	R/W	R/W	R/W	R/W
复位后	0	0	-	-	0	0	0	0

Bit 7 **GIEH**: 全局中断高优先级中断控制位  
 0 = 屏蔽高优先级中断  
 1 = 使能高优先级中断, 中断响应后自动清零

Bit 6 **GIEL**: 全局中断低优先级中断控制位  
 0 = 屏蔽低优先级中断  
 1 = 使能低优先级中断, 中断响应后自动清零

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]**: INTO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

**注: 同时使能 GIEL 和 GIEH, 高优先中断可嵌套低优先级中断, 但同级中断不可嵌套。**

## 5.4 IO 端口变化中断使能寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IOAICR	IOAICR7	IOAICR6	IOAICR5	IOAICR4	IOAICR3	IOAICR2	IOAICR1	IOAICR0
IOBICR	IOBICR7	IOBICR6	IOBICR5	IOBICR4	IOBICR3	IOBICR2	IOBICR1	IOBICR0
IOCICR	IOCICR7	IOCICR6	IOCICR5	IOCICR4	IOCICR3	IOCICR2	IOCICR1	IOCICR0
IODICR	-	-	IODICR5	IODICR4	IODICR3	IODICR2	IODICR1	IODICR0
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] **IOxICRy**: IOA端口变化中断使能 (x = A/B/C/D, y = 0-7)

0 = 屏蔽IOxy口电平变化中断

1 = 使能IOxy口电平变化中断

## 5.5 INTCR0 中断控制寄存器 0

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTCR0	RX0IE	TX0IE	I2CIE	CMP0IE	TC2GIE	TC2IE	TC1IE	TC0IE
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      **RX0IE:** USART0接收中断使能位  
           0 = 屏蔽USART0接收中断  
           1 = 使能USART0接收中断
- Bit 6      **TX0IE:** USART0发送中断使能位  
           0 = 屏蔽USART0发送中断  
           1 = 使能USART0发送中断
- Bit 5      **I2CIE:** I2C中断使能位  
           0 = 屏蔽I2C中断  
           1 = 使能I2C中断
- Bit 4      **CMPOIE:** 比较器中断使能位  
           0 = 屏蔽CMP中断  
           1 = 使能CMP中断
- Bit 3      **TC2GIE:** TC2门控中断使能位  
           0 = 屏蔽TC2门控中断  
           1 = 使能TC2门控中断
- Bit 2      **TC2IE:** TC2溢出中断使能位  
           0 = 屏蔽TC2溢出中断  
           1 = 使能TC2溢出中断
- Bit 1      **TC1IE:** TC1溢出中断使能位  
           0 = 屏蔽TC1溢出中断  
           1 = 使能TC1溢出中断
- Bit 0      **TC0IE:** TC0溢出中断使能位  
           0 = 屏蔽TC0溢出中断  
           1 = 使能TC0溢出中断

## 5.6 INTF0 中断标志寄存器 0

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTF0	RX0IF	TX0IF	-	CMP0IF	TC2GIF	TC2IF	TC1IF	TC0IF
读/写	R	R	-	R/W	R/W	R/W	R/W	R/W
复位后	0	1	-	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      **RX0IF**: USART0接收中断标志  
           0 = 未产生USART0接收中断  
           1 = 产生USART0接收中断（对RX0REG进行读操作后自动清零）
- Bit 6      **TX0IF**: USART0发送中断标志  
           0 = 未产生USART0发送中断（数据不为空，还在发送中）  
           1 = 产生USART0发送中断（数据为空，可发送下一个数据）
- Bit 4      **CMP0IF**: CMP中断标志  
           0 = 未产生CMP中断  
           1 = 产生CMP中断
- Bit 3      **TC2GIF**: TC2门控中断标志  
           0 = 未产生TC2门控中断  
           1 = 产生TC2门控中断
- Bit 2      **TC2IF**: TC2溢出中断标志  
           0 = 未产生TC2溢出中断  
           1 = 产生TC2溢出中断
- Bit 1      **TC1IF**: TC1溢出中断标志  
           0 = 未产生TC1溢出中断  
           1 = 产生TC1溢出中断
- Bit 0      **TC0IF**: TC0溢出中断标志  
           0 = 未产生TC0溢出中断  
           1 = 产生TC0溢出中断

## 5.7 INTPO 中断优先级寄存器 0

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTPO	RX0IP	TX0IP	I2CIP	CMP0IP	TC2GIP	TC2IP	-	TC0IP
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      **RX0IP:** UART0接收中断优先级控制位

0 = 高优先级

1 = 低优先级

Bit 6      **TX0IP:** UART0发送中断优先级控制位

0 = 高优先级

1 = 低优先级

Bit 5      **I2CIP:** I2C中断优先级控制位

0 = 高优先级

1 = 低优先级

Bit 4      **CMP0IP:** CMP中断优先级控制位

0 = 高优先级

1 = 低优先级

Bit 3      **TC2GIP:** TC2门控中断优先级控制位

0 = 高优先级

1 = 低优先级

Bit 2      **TC2IP:** TC2溢出中断优先级控制位

0 = 高优先级

1 = 低优先级

Bit 0      **TC0IP:** TC0溢出中断优先级控制位

0 = 高优先级

1 = 低优先级

## 5.8 INTCR1 中断控制寄存器 1

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTCR1	INT1IE	INT0IE	TKIE	ADIE	IODCHIE	IOCCHIE	IOBCHIE	IOACHIE
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      **INT1IE:** 外部端口中断1使能位  
           0 = 屏蔽外部端口中断1  
           1 = 使能外部端口中断1
- Bit 6      **INT0IE:** 外部端口中断0使能位  
           0 = 屏蔽外部端口中断0  
           1 = 使能外部端口中断0
- Bit 5      **TKIE:** 触摸中断使能位  
           0 = 屏蔽触摸中断  
           1 = 使能触摸中断
- Bit 4      **ADIE:** AD转换中断使能位  
           0 = 屏蔽AD转换中断  
           1 = 使能AD转换中断
- Bit 3      **IODCHIE:** 端口D变化中断使能位  
           0 = 屏蔽端口D变化中断  
           1 = 使能端口D变化中断
- Bit 2      **IOCCHIE:** 端口C变化中断使能位  
           0 = 屏蔽端口C变化中断  
           1 = 使能端口C变化中断
- Bit 1      **IOBCHIE:** 端口B变化中断使能位  
           0 = 屏蔽端口B变化中断  
           1 = 使能端口B变化中断
- Bit 0      **IOACHIE:** 端口A变化中断使能位  
           0 = 屏蔽端口A变化中断  
           1 = 使能端口A变化中断

## 5.9 INTF1 中断标志寄存器 1

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTF1	INT1IF	INT0IF	TKIF	ADIF	IODCHIF	IOCCHIF	IOBCHIF	IOACHIF
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

**注：所有中断标志位需软件清零。**

- Bit 7      **INT1IF**: 外部端口中断1标志  
           0 = 未产生外部端口中断1  
           1 = 产生外部端口中断1
- Bit 6      **INT0IF**: 外部端口中断0标志  
           0 = 未产生外部端口中断0  
           1 = 产生外部端口中断0
- Bit 5      **TKIF**: TK中断标志  
           0 = 未产生触摸中断  
           1 = 产生触摸中断
- Bit 4      **ADIF**: AD中断标志  
           0 = 未产生AD转换中断  
           1 = 产生AD转换中断
- Bit 3      **IODCHIF**: 端口D变化中断标志  
           0 = 未产生端口D变化中断  
           1 = 产生端口D变化中断
- Bit 2      **IOCCHIF**: 端口C变化中断标志  
           0 = 未产生端口C变化中断  
           1 = 产生端口C变化中断
- Bit 1      **IOBCHIF**: 端口B变化中断标志  
           0 = 未产生端口B变化中断  
           1 = 产生端口B变化中断
- Bit 0      **IOACHIF**: 端口A变化中断标志  
           0 = 未产生端口A变化中断  
           1 = 产生端口A变化中断



## 5.10 INTP1 中断优先级寄存器 1

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTP1	INT1IP	INT0IP	TKIP	ADIP	IODCHIP	IOCCHIP	IOBCHIP	IOACHIP
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      **INT1IP:** 外部端口中断1优先级控制位  
             0 = 高优先级  
             1 = 低优先级
- Bit 6      **INT0IP:** 外部端口中断0优先级控制位  
             0 = 高优先级  
             1 = 低优先级
- Bit 5      **TKIP:** 触摸中断优先级控制位  
             0 = 高优先级  
             1 = 低优先级
- Bit 4      **ADIP:** AD中断优先级控制位  
             0 = 高优先级  
             1 = 低优先级
- Bit 3      **IODCHIP:** 端口D变化中断优先级控制位  
             0 = 高优先级  
             1 = 低优先级
- Bit 2      **IOCCHIP:** 端口C变化中断优先级控制位  
             0 = 高优先级  
             1 = 低优先级
- Bit 1      **IOBCHIP:** 端口B变化中断优先级控制位  
             0 = 高优先级  
             1 = 低优先级
- Bit 0      **IOACHIP:** 端口A变化中断优先级控制位  
             0 = 高优先级  
             1 = 低优先级

## 5.11 INTCR2 中断控制寄存器 2

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTCR2	-	-	-	-	-	-	-	SPIIE
读/写	-	-	-	-	-	-	-	R/W
复位后	-	-	-	-	-	-	-	0

Bit 0      **SPIIE:** SPI中断使能位  
 0 = 屏蔽SPI中断  
 1 = 使能SPI中断

## 5.12 INTF2 中断标志寄存器 2

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTF2	-	-	-	-	-	-	-	SPIF2
读/写	-	-	-	-	-	-	-	R
复位后	-	-	-	-	-	-	-	0

**注：所有中断标志位需软件清零。**

Bit 0      **SPIF2:** SPI中断标志  
 0 = 清除对应SPIF或MODF后自动清0  
 1 = 产生SPI中断

**注：SPIF2 为 SPIF&MODF 或逻辑产生，只读，故清除此标志需清除对应被置 1 的 SPIF 或 MODF。**

## 5.13 INTP2 中断优先级寄存器 2

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTP2								SPIP
读/写								R/W
复位后								0

Bit 0      **SPIP:** SPI中断优先级控制位  
 0 = 高优先级  
 1 = 低优先级

## 6 端口

### 6.1 数据寄存器 IOx (x = A/B/C/D)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IOA	IOA7	IOA6	IOA5	IOA4	IOA3	IOA2	IOA1	IOA0
IOB	IOB7	IOB6	IOB5	IOB4	IOB3	IOB2	IOB1	IOB0
IOC	IOC7	IOC6	IOC5	IOC4	IOC3	IOC2	IOC1	IOC0
IOD	-	-	IOD5	IOD4	IOD3	IOD2	IOD1	IOD0
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

注：如读 IOA 的值读取的是引脚电平（模拟端口设置寄存器必需关闭，默认端口是模拟使能），向 IOA 写值是写入输出寄存器。

### 6.2 输出锁存寄存器 IOxOR (x = A/B/C/D)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IOAOR	IOAOR7	IOAOR6	IOAOR5	IOAOR4	IOAOR3	IOAOR2	IOAOR1	IOAOR0
IOBOR	IOBOR7	IOBOR6	IOBOR5	IOBOR4	IOBOR3	IOBOR2	IOBOR1	IOBOR0
IOCOR	IOCOR7	IOCOR6	IOCOR5	IOCOR4	IOCOR3	IOCOR2	IOCOR1	IOCOR0
IODOR	-	-	IODOR5	IODOR4	IODOR3	IODOR2	IODOR1	IODOR0
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

Bit [7:0] **IOxORy**: 输出锁存 (x = A/B/C/D, y = 0-7)

0 = 输出为0

1 = 输出为1

注：如对 IOAOR 寄存器的读操作是对 IOA 输出的读，对 IOAOR 寄存器的写操作是对 IOA 输出的写。

### 6.3 输出方向寄存器 OEx (x = A/B/C/D)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
OEA	OEA7	OEA6	OEA5	OEA4	OEA3	OEA2	OEA1	OEA0
OEB	OEB7	OEB6	OEB5	OEB4	OEB3	OEB2	OEB1	OEB0
OEC	OEC7	OEC6	OEC5	OEC4	OEC3	OEC2	OEC1	OEC0
OED	-	-	OED5	OED4	OED3	OED2	OED1	OED0
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] **OExy**: 输出/输出使能 (x = A/B/C/D, y = 0-7)

0 = 输入模式

1 = 输出模式

### 6.4 上拉控制寄存器 PUX (x = A/B/C/D)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PUA	PUA7	PUA6	PUA5	PUA4	PUA3	PUA2	PUA1	PUA0
PUB	PUB7	PUB6	PUB5	PUB4	PUB3	PUB2	PUB1	PUB0
PUC	PUC7	PUC6	PUC5	PUC4	PUC3	PUC2	PUC1	PUC0
PUD	-	-	PUD5	PUD4	PUD3	PUD2	PUD1	PUD0
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] **PUxy**: 上拉使能 (x = A/B/C/D, y = 0-7)

0 = 上拉关闭

1 = 上拉使能

### 6.5 下拉控制寄存器 PDx (x = A/B/C/D)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PDA	PDA7	PDA6	PDA5	PDA4	PDA3	PDA2	PDA1	PDA0
PDB	PDB7	PDB6	PDB5	PDB4	PDB3	PDB2	PDB1	PDB0
PDC	PDC7	PDC6	PDC5	PDC4	PDC3	PDC2	PDC1	PDC0
PDD	-	-	PDD5	PDD4	PDD3	PDD2	PDD1	PDD0
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] **PDxy**: 下拉使能 (x = A/B/C/D, y = 0-7)

0 = 下拉关闭

1 = 下拉使能

**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。**

## 6.6 模拟端口设置寄存器 ANSx (x = A/B/C/D)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ANSA	ANSA7	ANSA6	ANSA5	ANSA4	ANSA3	ANSA2	ANSA1	ANSA0
ANSB	ANSB7	ANSB6	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0
ANSC	ANSC7	ANSC6	ANSC5	ANSC4	ANSC3	ANSC2	ANSC1	ANSC0
ANSD	-	-	ANSD5	ANSD4	ANSD3	ANSD2	ANSD1	ANSD0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	1	1	1	1	1	1	1	1

Bit [7:0] **ANSxy**: 端口类型设置 (x = A/B/C/D, y = 0-7)

0 = 数字端口

1 = 模拟端口功能 (数字输入功能被屏蔽, 即端口状态MCU无法读取)

**注: 模拟端口模式, 仅数字输入功能被屏蔽, 但可输出。**

## 6.7 驱动电流选择寄存器 IOxODS1/ IOxODS0 (x = A/B/C/D)

### 驱动电流选择寄存器 1

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IOAODS1	IOAODS17	IOAODS16	IOAODS15	IOAODS14	IOAODS13	IOAODS12	IOAODS11	IOAODS10
IOBODS1	IOBODS17	IOBODS16	IOBODS15	IOBODS14	IOBODS13	IOBODS12	IOBODS11	IOBODS10
IOCODS1	IOCODS17	IOCODS16	IOCODS15	IOCODS14	IOCODS13	IOCODS12	IOCODS11	IOCODS10
IODODS1	-	-	IODODS15	IODODS14	IODODS13	IODODS12	IODODS11	IODODS10
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

### 驱动电流选择寄存器 0

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IOAODS0	IOAODS07	IOAODS06	IOAODS05	IOAODS04	IOAODS03	IOAODS02	IOAODS01	IOAODS00
IOBODS0	IOBODS07	IOBODS06	IOBODS05	IOBODS04	IOBODS03	IOBODS02	IOBODS01	IOBODS00
IOCODS0	IOCODS07	IOCODS06	IOCODS05	IOCODS04	IOCODS03	IOCODS02	IOCODS01	IOCODS00
IODODS0	-	-	IODODS05	IODODS04	IODODS03	IODODS02	IODODS01	IODODS00
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] **IOAx**: 驱动能力选择 (x = A/B/C/D, y = 0-7)

IOxODS1y, IOxODS0y	IOxy 驱动电流选择
00	最大(IoL1\ IoH1)
01	次大(IoL2\ IoH2)
10	次小(IoL3\ IoH3)
11	最小(IoL4\ IoH4)

## 6.8 翻转电平设置寄存器 IOxIPS/FLIPCR (x = A/B/C/D)

### 翻转控制寄存器 IOxIPS

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
IOAIPS	IOAIPS7	IOAIPS6	IOAIPS5	IOAIPS4	IOAIPS3	IOAIPS2	IOAIPS1	IOAIPS0
IOBIPS	IOBIPS7	IOBIPS6	IOBIPS5	IOBIPS4	IOBIPS3	IOBIPS2	IOBIPS1	IOBIPS0
IOCIPS	IOCIPS7	IOCIPS6	IOCIPS5	IOCIPS4	IOCIPS3	IOCIPS2	IOCIPS1	IOCIPS0
IODIPS	-	-	IODIPS5	IODIPS4	IODIPS3	IODIPS2	IODIPS1	IODIPS0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	1	1	1	1	1	1	1	1

Bit [7:0] **IOxIPSy**: 翻转电平选择 (x = A/B/C/D, y = 0-7)

0 = SMT

1 = 1.5 V 或 1/2VDD (由FLIPCR寄存器控制)

**注: IODIPS[7:6] 复位值为 00b。**

### 翻转控制寄存器 FLIPCR

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
FLIPCR	-	-	-	-	-	-	-	FLIPCRO
读/写	-	-	-	-	-	-	-	R/W
复位后	-	-	-	-	-	-	-	0

Bit 0 **FLIPCRO**: IO 口翻转控制, 在 IOxIPSy 为 1 时有效 (x = A/B/C/D, y = 0-7)

0 = 翻转电平为 1/2VDD

1 = 翻转电平为 1.5V

## 6.9 数字功能可选端口控制模块

引脚图中未标明的数字功能中，USART、I2C 和 PWM1 可以映射到任意端口，PWM0 与 LED COM 只能整体选择端口。

### 6.9.1 端口复用控制寄存器表

寄存器名称	复用功能	复用说明
MPTX	USART TX	全端口映射
MPRX	USART RX	全端口映射
MPSDA	I2C SDA	全端口映射
MPSCL	I2C SCL	全端口映射
MPPWM10	PWM10	全端口映射
MPPWM11	PWM11	全端口映射
MPPWM0	MPPWM0x (x = 0-3)	按端口整体选择
MPCOM	LED COM0 - COM7	按端口整体选择

### 6.9.2 全端口映射控制

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	-	-	PORTSEL[1:0]		-	PINSEL[2:0]		
读/写	-	-	R/W	R/W	-	R/W	R/W	R/W
复位后	-	-	0	0	-	0	0	0

Bit [5:4] **PORTSEL[1:0]**: 映射端口选择位

PORTSEL[1:0]	IO 端口组选择
00	IOA
01	IOB
10	IOC
11	IOD

Bit [2:0] **PINSEL[2:0]**: 映射管脚号选择位(x = A、B、C、D)

PINSEL[2:0]	IO 端口位选择
000	IOx0
001	IOx1
010	IOx2
011	IOx3
100	IOx4
101	IOx5
110	IOx6
111	IOx7

如需将 USART TX 端口映射到 IOB3，则可如下操作：

MPTX = 0x13; //TX 将映射到 IOB3 口所在的管脚。

### 6.9.3 PWM0 复用控制

PWM0 复用以四路为一组，下面的引脚为四路 PWM00 的引脚，其他三路按顺序向下排。比如 MPPWM0 设置为 00h，PWM00 从 IOA0 输出 PWM01 从 IOA1 输出 PWM02 从 IOA2 输出 PWM03 从 IOA3 输出，当 MPPWM0 设置为 34h 时，由于 IOD 口只到 IOC5 因此 PWM02、PWM03 不输出。

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
MPPWM0	-	-	PORTSEL[1:0]		-	PINSEL2	-	-
读/写	-	-	R/W	R/W	-	R/W	-	-
复位后	-	-	0	0	-	0	-	-

Bit [5:4] **PORTSEL[1:0]:** PWM0 输出端口 IOx 选择(x = A、B、C、D)

MPPWM [5:4]	IO 口选择
00	IOA
01	IOB
10	IOC
11	IOD

Bit 2 **PINSEL2:** PWM0y 输出端口高低位选择 (y = 0-3)  
 0 = PWM00/01/02/03 输出端口依次为 IOx0/1/2/3  
 1 = PWM00/01/02/03 输出端口依次为 IOx4/5/6/7

**注：由于 IOD 端口无 IOD6&IOD7 口，故当 PWM0 选择 IOD 口时，PWM02&PWM03 将无法输出。**

### 6.9.4 LED COM 端口复用控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
MPCOM	-	-	MPCOM[5:4]		-	-	-	-
读/写	-	-	R/W	R/W	-	-	-	-
复位后	-	-	0	0	-	-	-	-

Bit [5:4] **MPCOM [5:4]:** COM 复用

MPCOM[5:4]	IO 口选择
00	COM0-COM7 分别对应 IOA0-IOA7
01	COM0-COM7 分别对应 IOB0-IOB7
10	COM0-COM7 分别对应 IOC0-IOC7
11	COM0-COM5 分别对应 IOD0-IOD5

**注：由于 IOD 端口无 IOD6&IOD7 口，故当 COM 选择 IOD 口时，COM6&COM7 将无法输出。**



# 7 定时器 0(TCO/PWM0)

## 7.1 概述

TCO 为带有可设置 1-128 预分频器及 1- 8 后分频器和周期寄存器的 8 位/16 位定时计数器，具有休眠状态下唤醒功能。

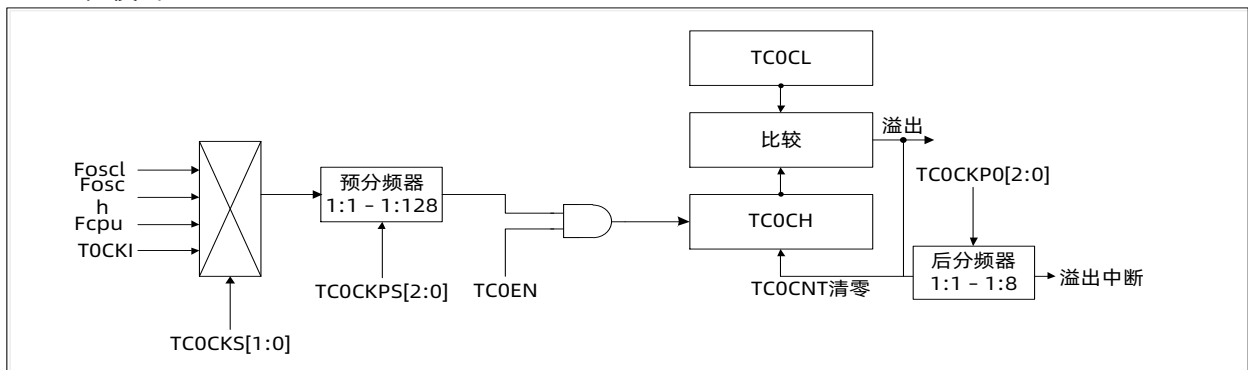
在 8 位模式下，TCOCL 作为 TCO 的周期寄存器，TCO 使能后，TCOCH 递加，当 TCOCH 与 TCOCL 数值相等，TCO 溢出，将 TCOCH 清零重新开始计数，同时将中断标志位 TCOIF 置 1。

在 16 位模式下，{TCOCH:TCOCL}作为 16 位的计数器，TCO 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 TCOIF 置 1。

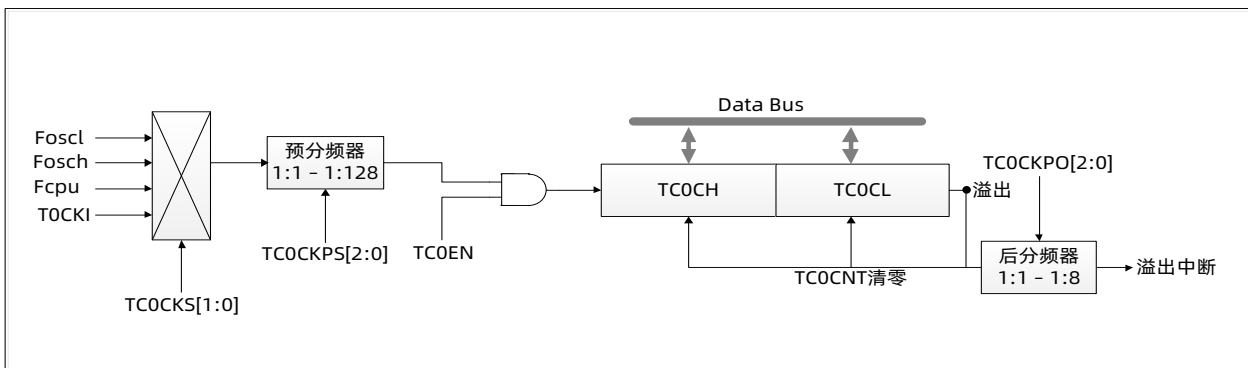
- 可选择时钟源：高频系统时钟  $F_{OSCH}$ 、低频系统时钟  $F_{OSCL}$ 、指令时钟  $F_{CPU}$  或外部时钟  $TOCKI$
- 可选择 8 位模式和 16 位模式，使能 PWM0 时，只能在 8 位模式
- 预分频比多级可选，最大可选择 1:128
- 后分频比多级可选，最大可选择 1:8
- 溢出中断功能
- 溢出中断唤醒功能（当输入频率选择  $F_{OSCH}$  或  $F_{OSCL}$  时，若所选择的时钟源振荡器一直工作，此时 TCO 在休眠状态下依然工作，溢出中断可唤醒 CPU）

TCO 框图

8 位模式



16 位模式



## 7.2 TOCR 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TOCR	TCOEN	TCOMOD	TCOOSCS	TCOCKS[1:0]		TCOCKPS[2:0]		
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

Bit 7

**TCOEN**: TCO模块使能位

0 = 关闭TCO

1 = 使能TCO

Bit 6

**TCOMOD**: TCO模式选择位

0 = 8位模式

1 = 16位模式

Bit 5

**TCOOSCS**: TCO内部高频时钟选择（系统时钟选择外部晶振时，此控制位无效）

0 = 24MHz（使用触摸功能，禁止选择）

1 = 16MHz

Bit [4:3]

**TCOCKS[1:0]**: TCO时钟源选择

TCOCKS[1:0]	TCO 时钟源选择
00	F <sub>OSCL</sub> (低频系统时钟)
01	F <sub>OSCH</sub> (高频系统时钟)
10	F <sub>CPU</sub>
11	TOCKI

Bit [2:0]

**TCOCKPS[2:0]**: TCO预分频比选择

TCOCKPS[2:0]	TCO 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 TOCR2 TCO 控制寄存器 2

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TOCR2	-	-	-	-	-	TCOCKPO[2:0]		
读/写	-	-	-	-	-	R/W	R/W	R/W
复位后	-	-	-	-	-	0	0	0

Bit [2:0] **TCOCKPO[2:0]**: TCO 后分频, 在原来预分频的基础上再分频

TCOCKPO[2:0]	TCO 后分频比
000	1:1
001	1:2
010	1:3
011	1:4
100	1:5
101	1:6
110	1:7
111	1:8

## 7.4 TCOCL TCO 计数器低位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TCOCL	TCOCL[7:0]							
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 TCOCH TCO 计数器高位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TCOCH	TCOCH[7:0]							
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

## 7.6 PWM0xCR PWM 控制寄存器 (x = 0,1,2,3)

PWM0x 可以复用到很多 IO 口，其复用以四路为一组，具体见 6.9 小节。

**PWM0xCR PWM0x 控制寄存器**

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM00CR	PWM00EN	PWM00OE	PWM00M	-	-	-	-	-
PWM01CR	PWM01EN	PWM01OE	PWM01M	-	-	-	-	-
PWM02CR	PWM02EN	PWM02OE	PWM02M	-	-	-	-	-
PWM03CR	PWM03EN	PWM03OE	PWM03M	-	-	-	-	-
读/写	R/W	R/W	R/W	-	-	-	-	-
复位后	0	0	0	-	-	-	-	-

- Bit 7      **PWM0xEN**: PWM0x使能位  
 0 = 关闭PWM  
 1 = 使能PWM (TC0必须选择8位模式)
- Bit 6      **PWM0xOE**: PWM0x输出控制  
 0 = PWM信号不从管脚输出，管脚用做IO  
 1 = PWM信号从管脚输出
- Bit 5      **PWM0xM**: PWM0x输出极性控制  
 0 = PWM高电平有效  
 1 = PWM低电平有效

## 7.7 PWM0xD PWM 占空比寄存器 (x = 0,1,2,3)

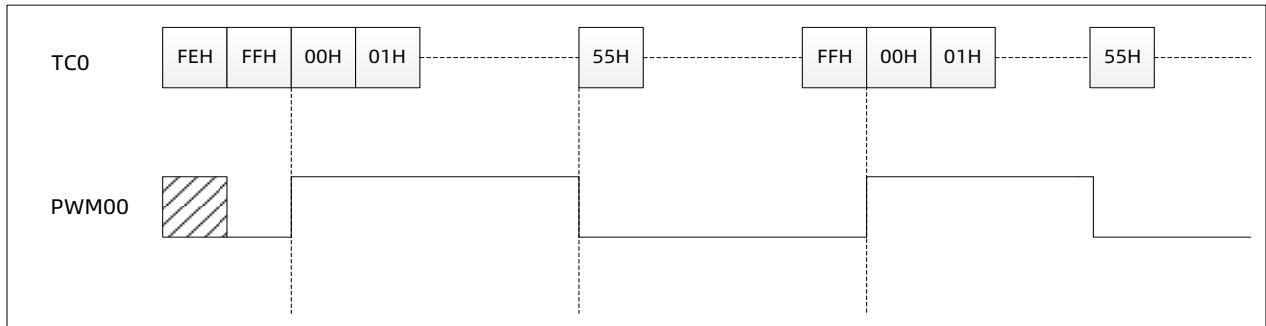
**PWM0xD 占空比寄存器**

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM00D	PWM00D[7:0]							
PWM01D	PWM01D[7:0]							
PWM02D	PWM02D[7:0]							
PWM03D	PWM03D[7:0]							
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

Bit [7:0]      **PWM0xD**: PWM0x占空比控制寄存器

## 7.8 PWM0x 波形实例 (x = 0,1,2,3)

例: PWM00CR = 11100000B, PWM00D = 55h, TC0CL = FFh。



# 8 定时器 1 (TC1/PWM1)

## 8.1 概述

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

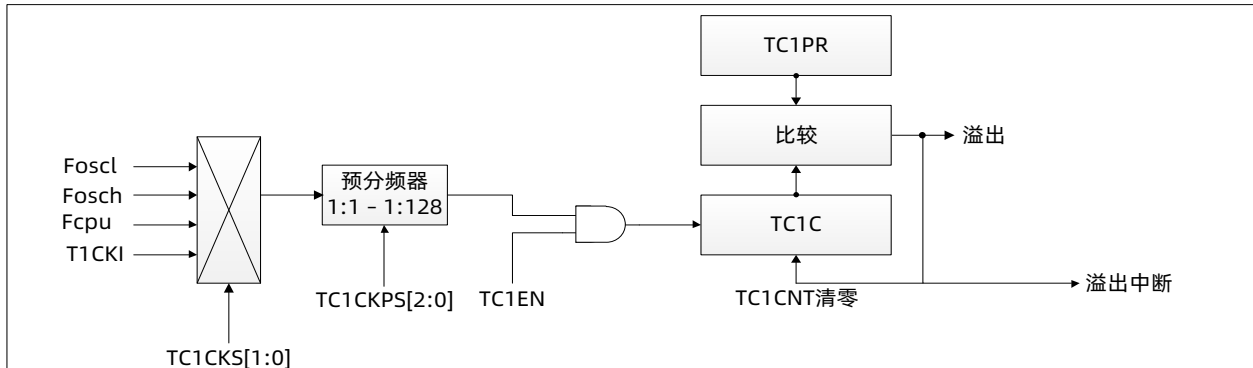
向上计数模式，TC1 使能后，TC1C 递增，当 TC1C 计数值与 TC1PR 相等时，TC1 溢出，将 TC1CH 清零重新开始计数，同时将中断标志位 T1IF 置 1。

向下计数模式，TC1 使能后，TC1C 递减，当 TC1C 计数值为 0 时，TC1 溢出，TC1C 将重新载入 TC1PR 的值开始递减计数，同时将中断标志位 T1IF 置 1。

中间对齐方式，TC1 使能后，TC1C 递增计数，当 TC1C 递增到 TC1PR 时，产生溢出中断，然后 TC1C 开始递减，递减到 0 时，再次产生溢出中断，同时开始递增。

- 可选择时钟源：高频系统时钟  $F_{OSCH}$ 、低频系统时钟  $F_{OSCL}$ 、指令时钟  $F_{CPU}$  或外部时钟 T1CKI
- 16 位周期寄存器
- 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  $F_{OSCH}$  或  $F_{OSCL}$  时，若所选择的时钟源振荡器一直工作，此时 TC1 在休眠状态下依然工作，溢出中断可唤醒 CPU。）

TC1 框图



## 8.2 T1CR 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
T1CR	TC1EN	TC1MOD[1:0]		TC1CKS[1:0]		TC1CKPS[2:0]		
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

Bit 7 **TC1EN**: TC1模块使能位  
 0 = 关闭TC1  
 1 = 使能TC1

Bit [6:5] **TC1MOD[1:0]**: TC1模式选择位

TC1MOD[1:0]	TC1 模式选择
00	递增模式
01	递减模式
1x	中心对齐模式

Bit [4:3] **TC1CKS[1:0]**: TC1时钟源选择

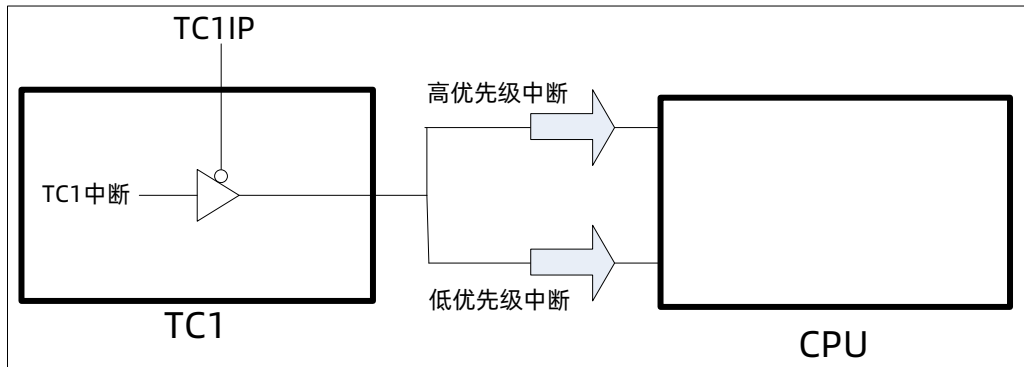
TC1CKS[1:0]	TC1 时钟源选择
00	F <sub>OSCL</sub> (低频系统时钟)
01	F <sub>OSCH</sub> (高频系统时钟)
10	F <sub>CPU</sub>
11	T1CKI

Bit [2:0] **TC1CKPS[2:0]**: TC1预分频比选择

TC1CKPS[2:0]	TC1 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

### 8.3 INTPTC1 TC1 中断控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTPTC1	INTPTC1 [7:0]							
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



#### 注：TC1 中断使用注意事项

- (1) TC1 中断是高、低优先级同时触发，当程序同时使能高、低优先级时，只会进高优先级中断。
- (2) 使用 TC1 中断且程序同时使能高、低优先级时，低优先级中断处理程序中，须查询 TC1 中断标志位，如有置起，清 0。
- (3) 使用 TC1 中断时必须要保持该寄存器全为 0。

### 8.4 TC1CL TC1 计数器低位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TC1CL	TC1CL[7:0]							
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

### 8.5 TC1CH TC1 计数器高位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TC1CH	TC1CH[7:0]							
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



## 8.6 TC1PRL TC1 周期寄存器低位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TC1PRL	TC1PRL[7:0]							
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

## 8.7 TC1PRH TC1 周期寄存器高位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TC1PRH	TC1PRH[7:0]							
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

## 8.8 PWM1CR 控制寄存器

PWM1 输出端口可以复用到任意 IO 端口，详见章节 6.9。

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM1CR	PWM10EN	PWM10OE	PWM10M	PWM11EN	PWM11OE	PWM11M	PWM1S	-
读/写	R/W	R/W	R/W	R/W	R/W	R/W	RW	-
复位后	0	0	0	0	0	0	0	-

Bit 7      **PWM10EN**: PWM10模块使能位

0 = 关闭PWM10

1 = 使能PWM10

Bit 6      **PWM10OE**: PWM10波形输出使能位

0 = 端口用作IO

1 = 端口输出PWM10波形

Bit 5      **PWM10M**: PWM10输出模式

0 = 高电平有效

1 = 低电平有效

Bit 4      **PWM11EN**: PWM11模块使能位

0 = 关闭PWM11

1 = 使能PWM11

Bit 3      **PWM11OE**: PWM11波形输出使能位

0 = 端口用作IO

1 = 端口输出PWM11波形

Bit 2      **PWM11M**: PWM11输出模式

0 = 低电平有效

1 = 高电平有效

Bit 1      **PWM1S**: PWM1模式

0 = 独立输出

1 = 互补输出

## 8.9 PWM1xD 数据寄存器 (x = 0,1)

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM1xDH	PWM1xDH[7:0]							
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

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM1xDL	PWM1xDL[7:0]							
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

## 8.10 PWM 死区控制寄存器

当 PWM1CR 设置为互补模式时, PWM10D 作为数据寄存器, PWM11D 作为死区寄存器, PWM11DH 前死区, PWM11DL 后死区。

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM11DH	PWM11DH[7:0]							
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

Bit [7:0] **PWM11DH[7:0]**: 前死区宽度设置

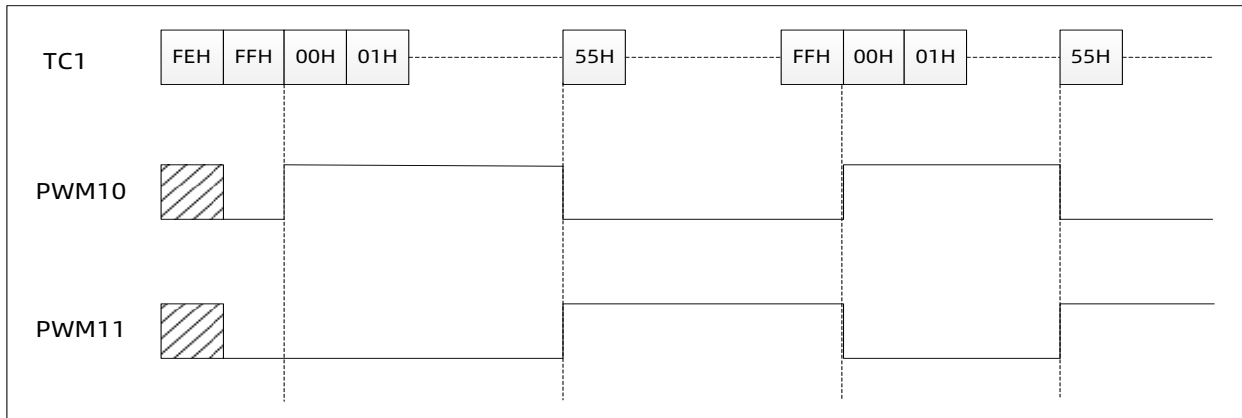
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM11DL	PWM11DL[7:0]							
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	0	0	0	0	0

Bit [7:0] **PWM11DL[7:0]**: 后死区宽度设置

## 8.11 PWM1 波形示例

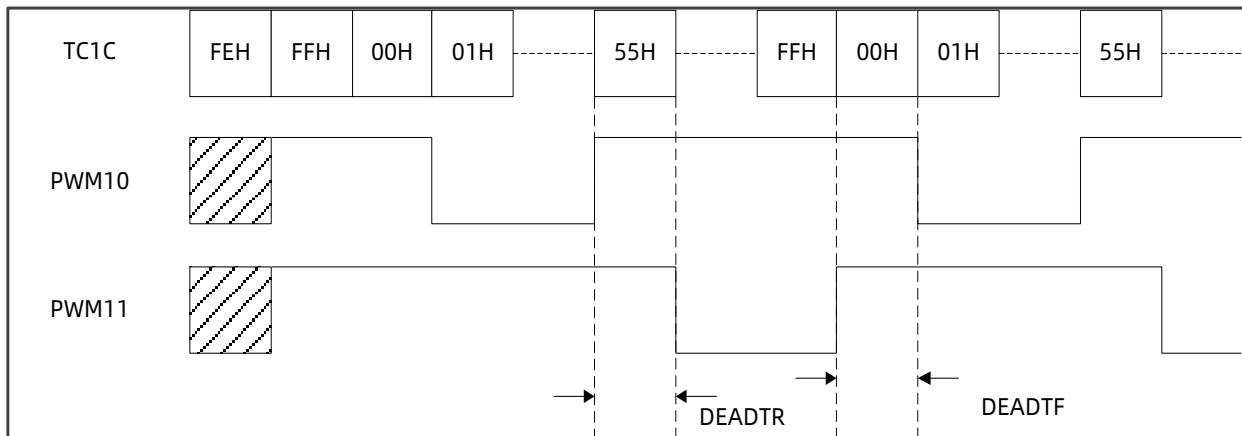
### 8.11.1 互补 PWM1 输出

例：PWM1CR=11011010B, PWM10DH=00h, PWM10DL=55h, TC1PRL=FFh, TC1PRH=00h  
 PWM11DH=00h, PWM11DL=00h。



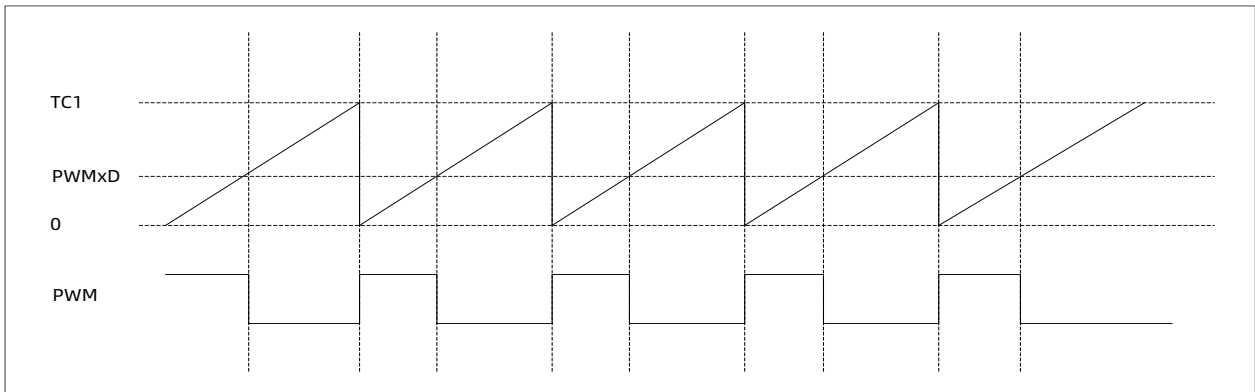
### 8.11.2 带死区的互补 PWM1 输出

例：PWM1CR=1111110B, PWM10DH=00h, PWM10DL=55h, TC1PRL=FFh, TC1PRH=00h,  
 PWM11DH=01h, PWM11DL=01h。

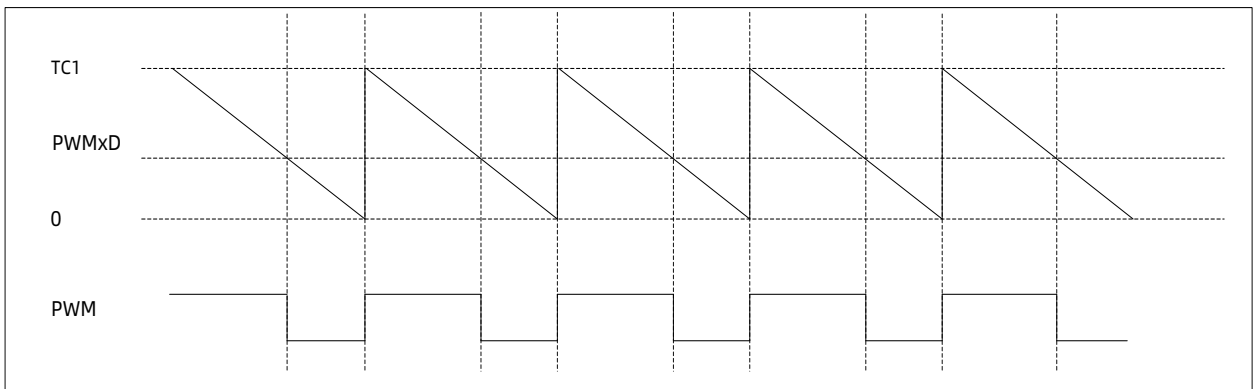


### 8.11.3 PWM 波形图

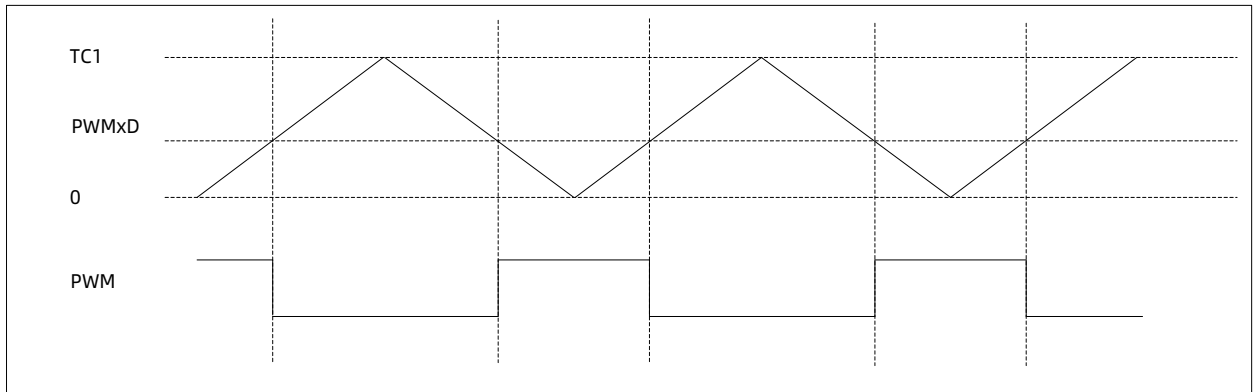
TC1 递增模式



TC1 递减模式



TC1 中心对齐模式



**注：请不要在 PWM 模块使能时修改 PWM1，因为修改会立刻生效，导致当前这个周期出错。**

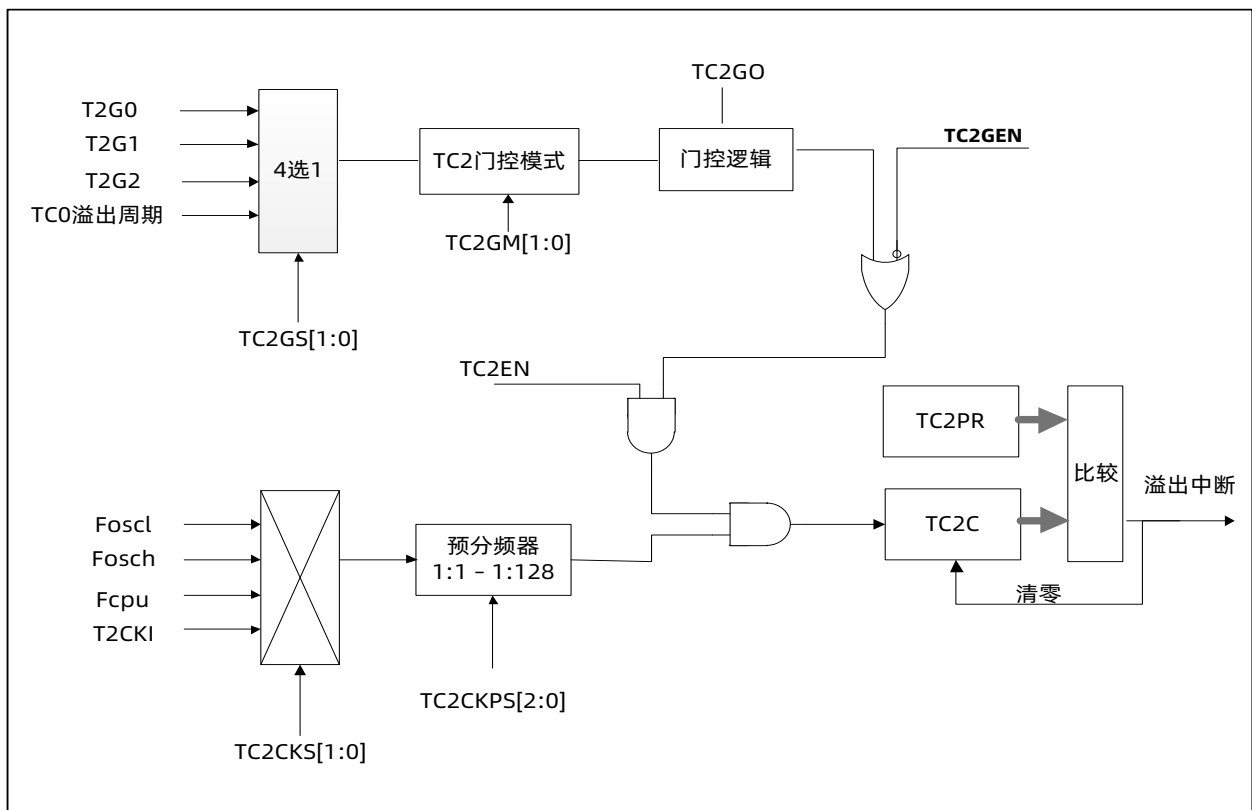
# 9 定时器 2 (TC2)

## 9.1 概述

TC2 为带门控功能的可设置 1-128 预分频器及周期寄存器的 16 位定时计数器，具有休眠状态下唤醒功能。

- 可选择时钟源：高频系统时钟  $F_{OSCH}$ 、低频系统时钟  $F_{OSCL}$ 、指令时钟  $F_{CPU}$  或  $T2CKI$
- 可选择预分频比，最大 1:128
- 门控模式可选择门控源：外部门控信号  $T2G0$ 、 $T2G1$ 、 $T2G2$  或  $TC0$  溢出信号
- 门控和溢出中断功能
- 溢出中断唤醒功能（当输入频率选择  $F_{OSCH}$  或  $F_{OSCL}$  输出时，若所选择的时钟源振荡器一直工作，此时 TC2 在休眠状态下依然工作，溢出中断可唤醒 CPU。）

TC2 框图



## 9.2 T2CR 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
T2CR	TC2EN	-	-	TC2CKS[1:0]		TC2CKPS[2:0]		
读/写	R/W	-	-	R/W	R/W	R/W	R/W	R/W
复位后	0	-	-	0	0	0	0	0

Bit 7 **TC2EN**: TC2模块使能位  
 0 = 关闭TC2  
 1 = 使能TC2

Bit [4:3] **TC2CKS[1:0]**: TC2时钟源选择位

TC2CKS[1:0]	TC2 计数时钟源选择
00	F <sub>OSCL</sub> (低频系统时钟)
01	F <sub>OSCH</sub> (高频系统时钟)
10	F <sub>CPU</sub>
11	T2CKI

Bit [2:0] **TC2CKPS[2:0]**: TC2预分频比选择

TCxCKPS[2:0]	TC2 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

## 9.3 TC2CL 计数器低位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TC2CL	TC2CL[7:0]							
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

## 9.4 TC2CH 计数器高位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TC2CH	TC2CH[7:0]							
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

## 9.5 TC2PRL 周期寄存器低位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TC2PRL	TC2PRL[7:0]							
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

## 9.6 TC2PRH 周期寄存器高位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TC2PRH	TC2PRH[7:0]							
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



## 9.7 TC2GCR 门控制控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TC2GCR	TC2GEN	TC2GO	-	-	TC2GS[1:0]		TC2GM[1:0]	
读/写	R/W	R/W	-	-	R/W	R/W	R/W	R/W
复位后	0	0	-	-	0	0	0	0

Bit 7 **TC2GEN**: TC2门控模式使能位  
 0 = 关闭门控功能, TC2是否计数仅受TC2GEN控制  
 1 = 使能门控功能

Bit 6 **TC2GO**: 启动门控控制位  
 0 = 完成门控计数, 自动清零  
 1 = 启动门控

Bit [3:2] **TC2GS[1:0]**: TC2门控源选择位

TC2GS[1:0]	TC2 门控选择
00	T2G0
01	T2G1
10	T2G2
11	TC0 溢出周期 (只支持上升沿到上升沿模式)

Bit [1:0] **TC2GM[1:0]**: TC2门控模式选择位 (只有T2G0/T2G1/T2G2下可选择)

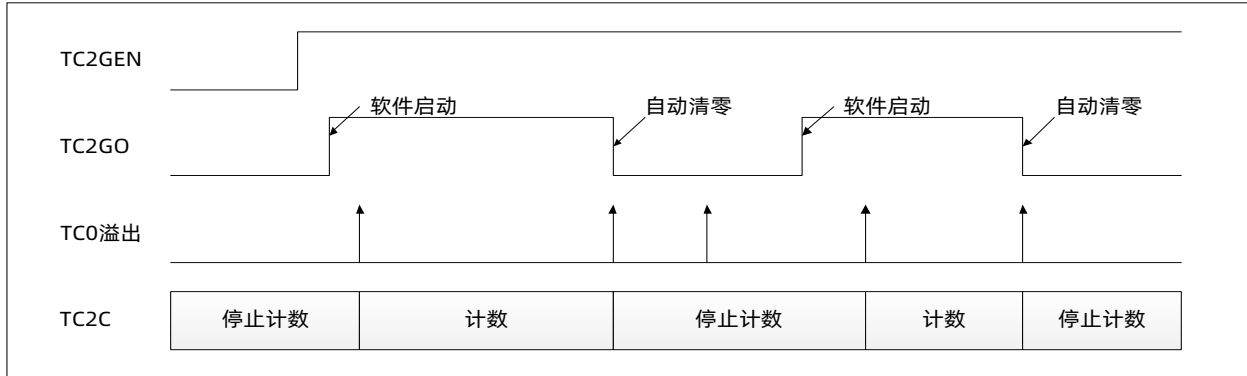
TC2GM[1:0]	TC2 门控模式选择
00	上升沿到下降沿
01	下降沿到上升沿
10	上升沿到上升沿
11	下降沿到下降沿

**注:**

- (1) 启动门控前需先将门控使能, 不可同时置 1。
- (2) 选择 TC0 溢出信号作为门控源时, 若 TC0 是 8 位模式, 则 TC0CL 不能为 0。

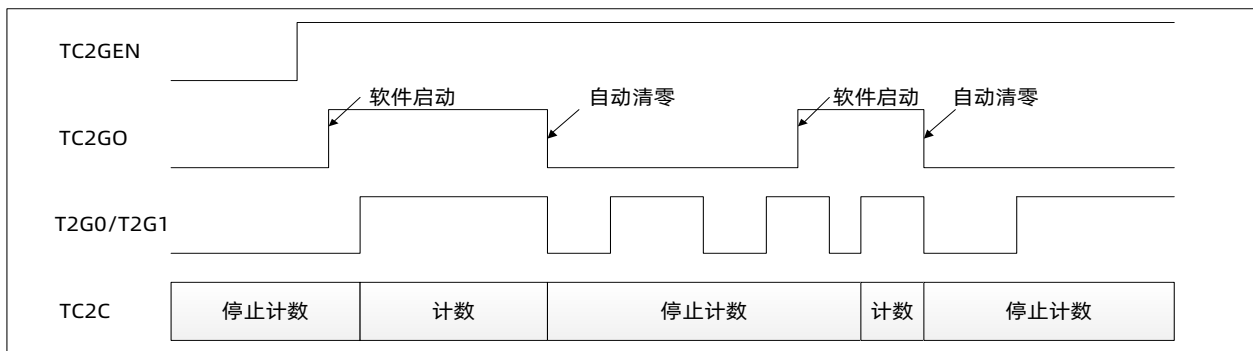
### 9.7.1 门控-TC0 溢出周期

上升沿到上升沿模式：启动门控计数后，门控逻辑从第一次 TC0 溢出开始计数，第二次 TC0 溢出停止计数，如下图。



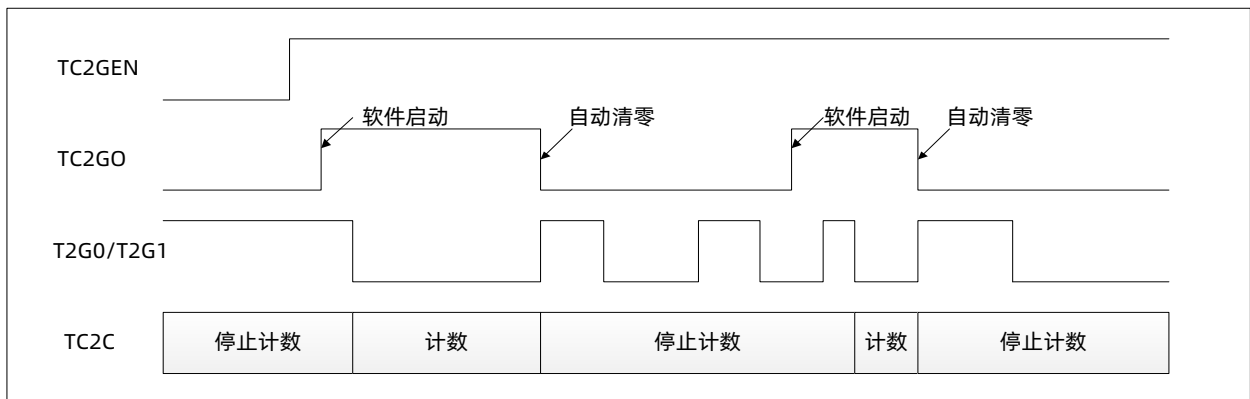
### 9.7.2 门控-上升沿到下降沿模式

上升沿到下降沿模式：启动门控计数后，门控逻辑从捕获到的第一个上升沿开始计数，然后捕获到下降沿停止计数，如下图。



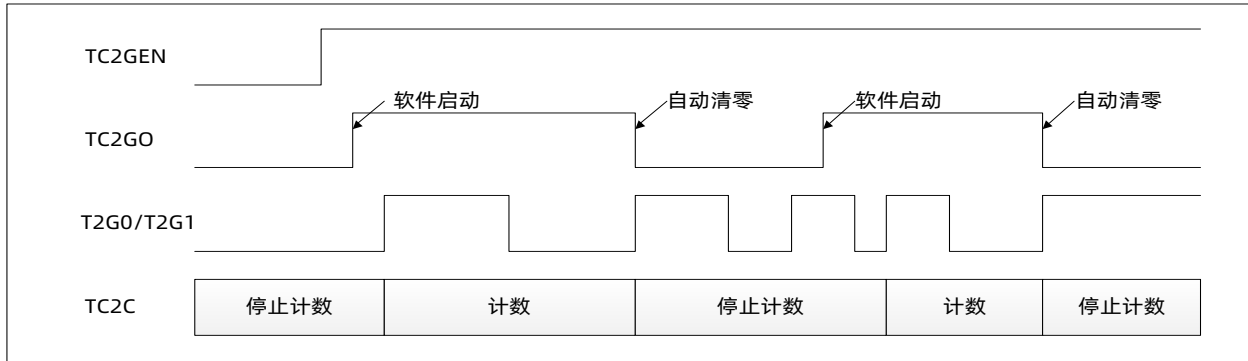
### 9.7.3 门控-下降沿到上升沿模式

下降沿到上升沿模式：启动门控计数后，门控逻辑从捕获到的第一个下降沿开始计数，然后捕获到上升沿停止计数，如下图。



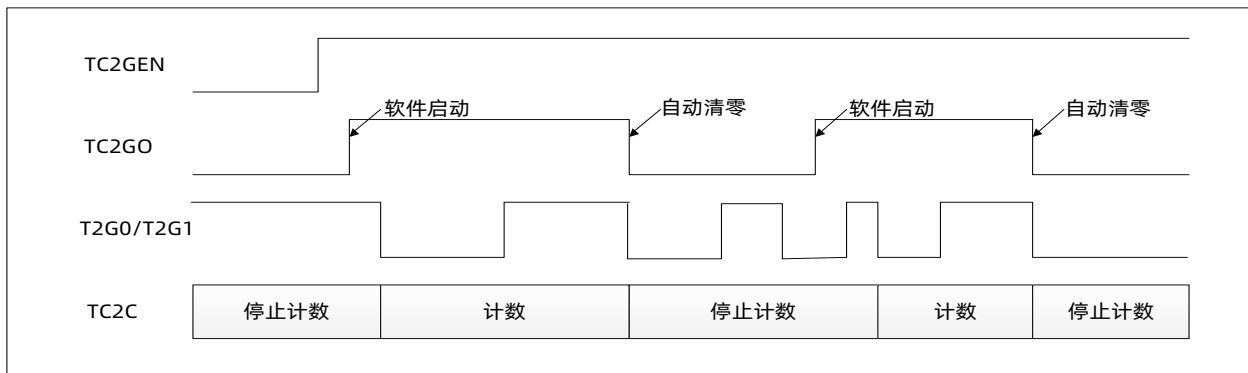
### 9.7.4 门控-上升沿到上升沿模式

上升沿到上升沿模式：启动门控计数后，门控逻辑从捕获到的第一个上升沿开始计数，然后捕获到第二个上升沿停止计数，如下图。



### 9.7.5 门控-下降沿到下降沿模式

下降沿到下降沿模式：启动门控计数后，门控逻辑从捕获到的第一个下降沿开始计数，然后捕获到第二个下降沿停止计数，如下图。



# 10 通用串行通讯口 (USART)

## 10.1 概述

支持异步全双工模式和同步半双工模式。

USART 的 TX、RX 口可以复用到任意 IO 口，详见章节 6.9。

## 10.2 TX0CR 发送控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TX0CR	TXEN	TMCLR	SYNC	TX9	SLAVE	SPD[1:0]		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]**: 波特率时钟源分频比

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 TX0REG 发送数据寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TX0REG	TX0REG[7:0]							
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 RX0CR 接收控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
RX0CR	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	x

- 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 RXOREG 接收数据寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
RXOREG	RXOREG[7:0]							
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 BRGD0H 波特率寄存器高位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BRGD0H	SBYTE	ODEN	-	-	-	-	BRGD0H[9:8]	
读/写	R/W	W	-	-	-	-	R/W	R/W
复位后	0	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

Bit 6      **ODEN**: 开漏输出使能  
 0 = 不使能  
 1 = 使能

**注: 开漏使能时, 并非真正意义上的开漏, 即管脚允许接入的高电平依旧需要在芯片允许范围内。只是芯片发送数据位为 1 (高电平) 时, 不是内部推挽输出的 1 (高电平), 此时输出 1 (高电平) 需外接电阻上拉置需要的 1 (高电平)。**

Bit [1:0]      **BRGD[9:8]**: 10位波特率寄存器高2位

## 10.7 BRGD0L 波特率寄存器低位

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BRGD0L	BRGD0L[7:0]							
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

Bit [7:0]      **BRGD[7:0]**: 10位波特率寄存器低8位

## 10.8 USART 使用说明

### 10.8.1 波特率设置

通过设置 BRGD 和 SPD (分频比 n) 来获得所需的波特率。

波特率计算公式： 目标波特率 =  $F_{OSC}/((BRGD+1)\times n)$

常用波特率设置 ( $F_{OSC} = 16\text{MHz}$ )

目标波特率	波特率时钟源分频比	BRGD	偏差
300	256	0xCF	0.16%
600	256	0x67	0.16%
1200	64	0xCF	0.16%
2400	64	0x67	0.16%
4800	16	0xCF	0.16%
9600	16	0x67	0.16%
19200	4	0xCF	0.16%
38400	4	0x67	0.16%
57600	4	0x44	0.64%
115200	4	0x22	-0.79%

常用波特率设置 ( $F_{OSC} = 24\text{MHz}$ )

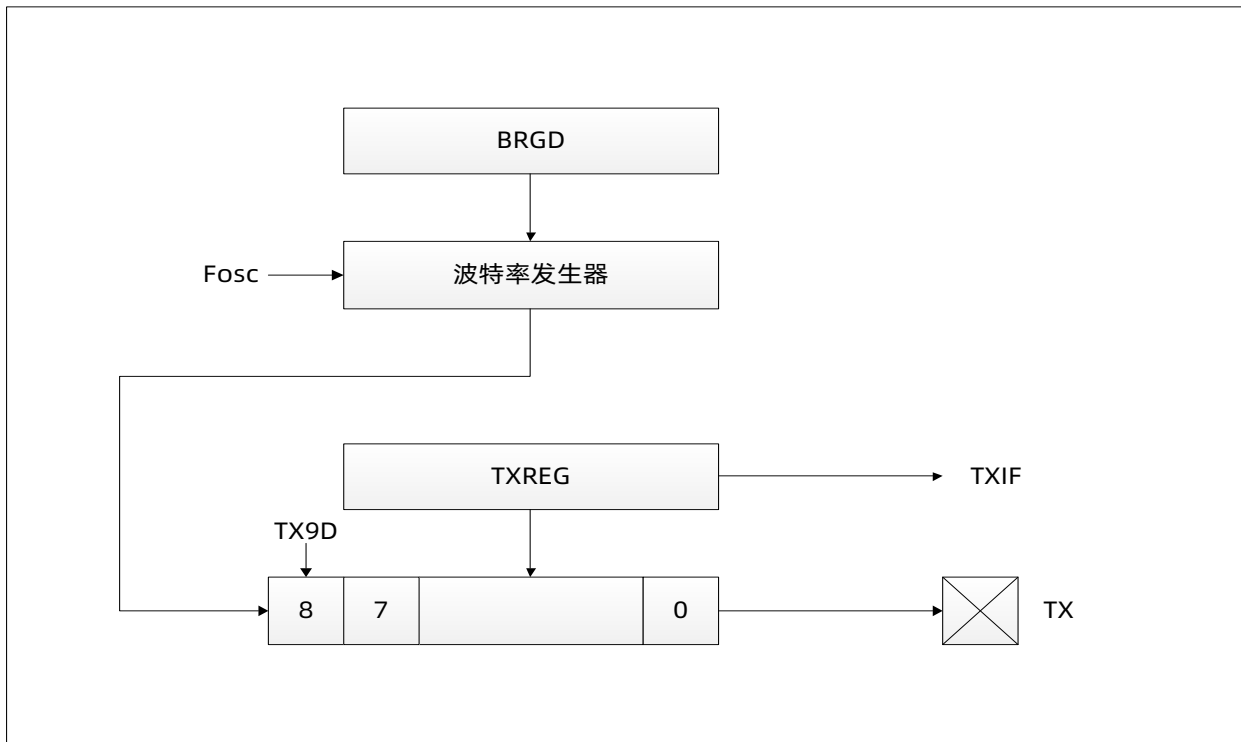
目标波特率	波特率时钟源分频比	BRGD	偏差
300	256	0x137	0.16%
600	256	0x9B	0.16%
1200	64	0x137	0.16%
2400	64	0x9B	0.16%
4800	16	0x137	0.16%
9600	16	0x9B	0.16%
19200	4	0x137	0.16%
38400	4	0x9B	0.16%
57600	4	0x67	0.16%
115200	4	0x33	0.16%

### 10.8.2 异步发送

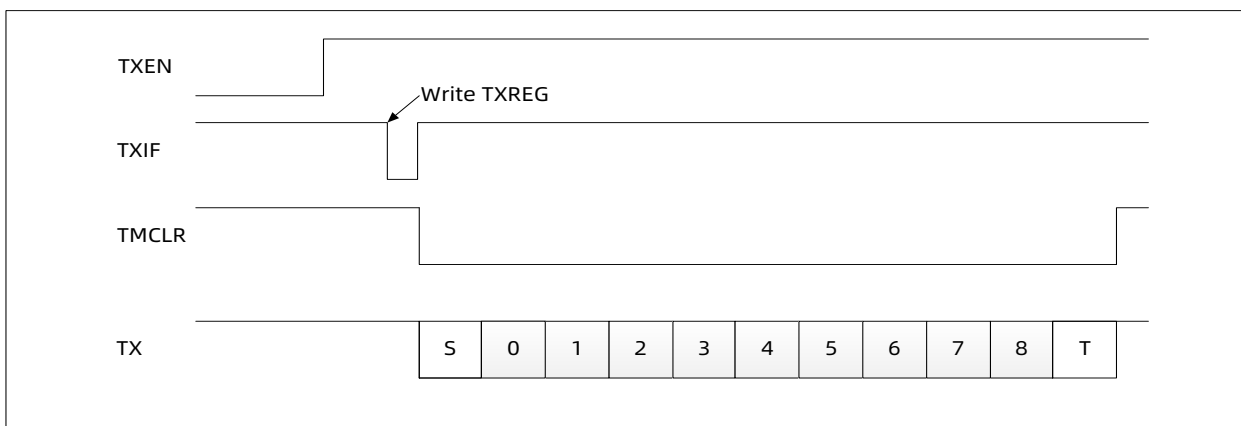
当 TXEN 使能时，TXIF 中断标志为 1 说明 TX0REG 发送寄存器为空，TMCLR = 1 说明发送移位寄存器为空，发送器处于空闲状态。

空闲状态时写入 TX0REG，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 TX0REG，TXIF 将清零，说明 TX0REG 有未发送数据，发送移位寄存器发送完毕后，TX0REG 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。

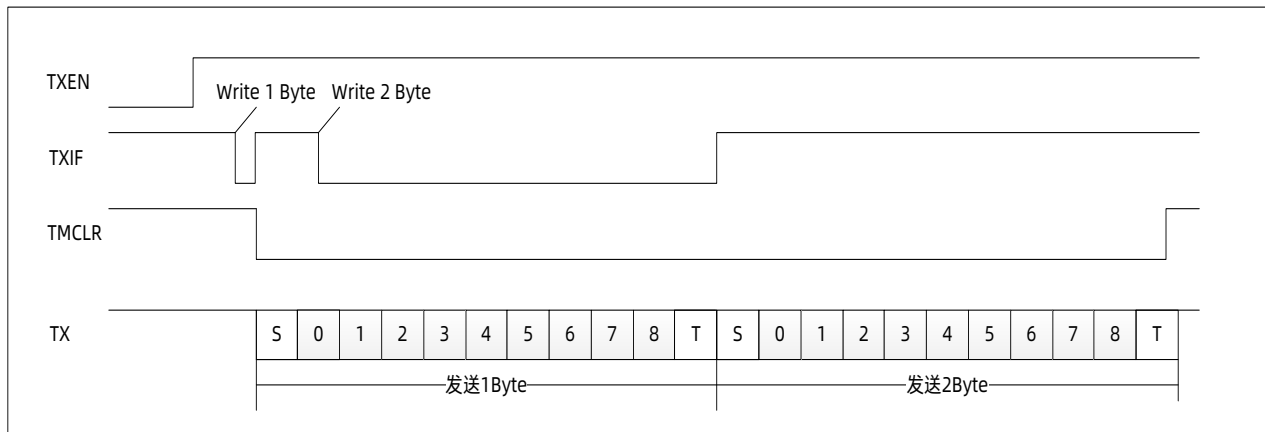


单字节发送：





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

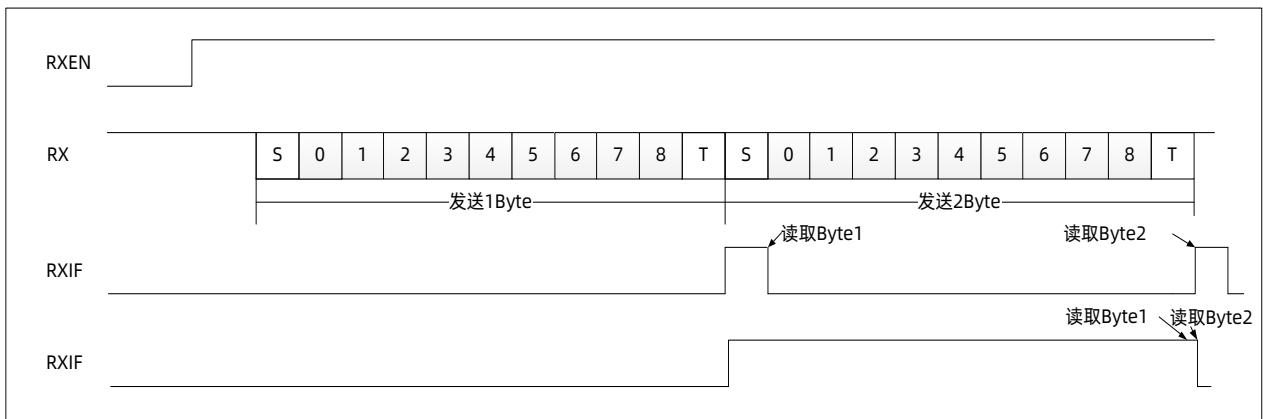
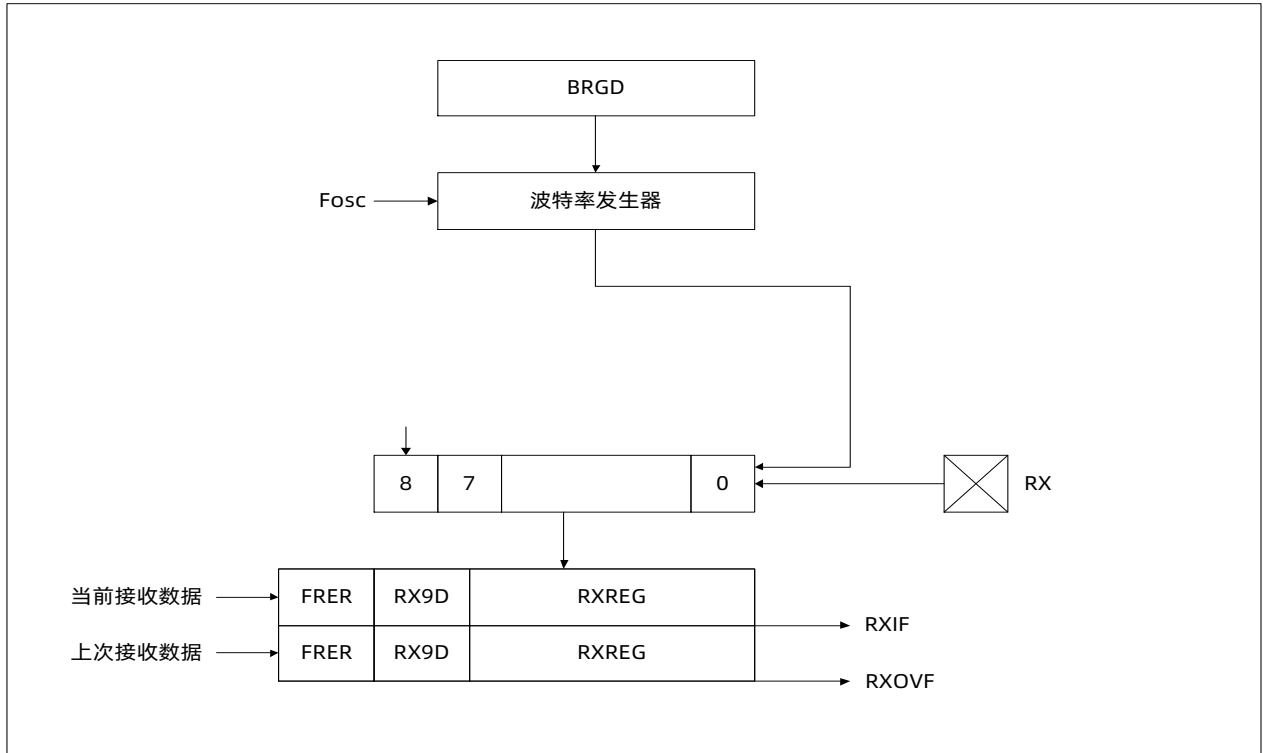
STEP4: 写入 TX0REG, 启动发送

STEP5: 当 TXIF = 1 时, 写入 TX0REG 发送下一个字节

STEP6: 重复 STEP5, 直到该帧数据发送完成

### 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 RXOREG 后 RXIF 自动清零。



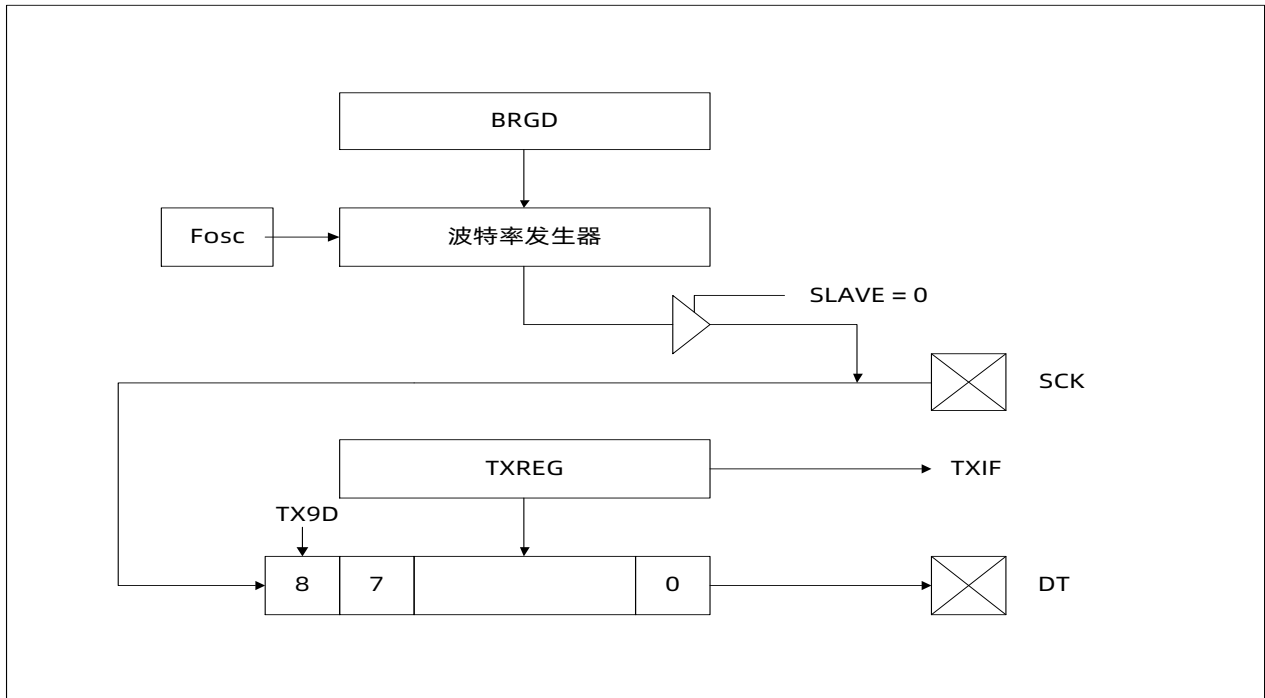
- STEP1: 设置波特率  $SSPD = X$ ,  $BRGD = X$ ,  $SYNC = 0$
- STEP2: 设置  $RXEN = 1$
- STEP3: 等待接收完成  $RXIF = 1$
- STEP4: 判断  $FRER = 0$ , 若为 1, 帧格式错误, 舍弃数据
- STEP5: 读取  $RX9D$
- STEP6: 读取  $RXOREG$ , 重复 3-6

### 10.8.4 同步发送

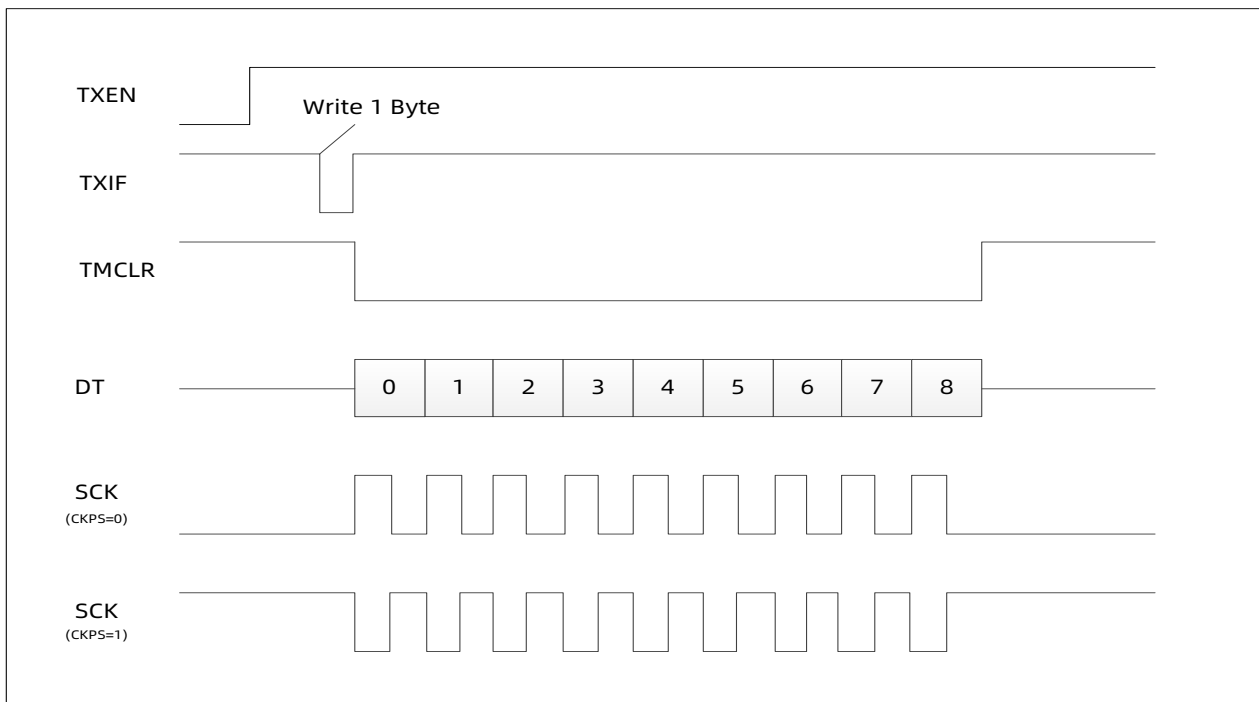
当 TXEN=1, SYNC=1 时, 使能同步发送功能。CKPS 选择发送时钟极性, TXIF 中断标志为 1 说明 TXOREG 发送寄存器为空, TMCLR=1 说明发送移位寄存器为空, 发送器处于空闲状态。

空闲状态写入 TXOREG, 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 TXOREG, TXIF 将清零, 说明 TXOREG 有未发送数据, 发送移位寄存器发送完毕后, TXOREG 数据将自动载入发送移位寄存器继续发送, 且 TXIF 为空。

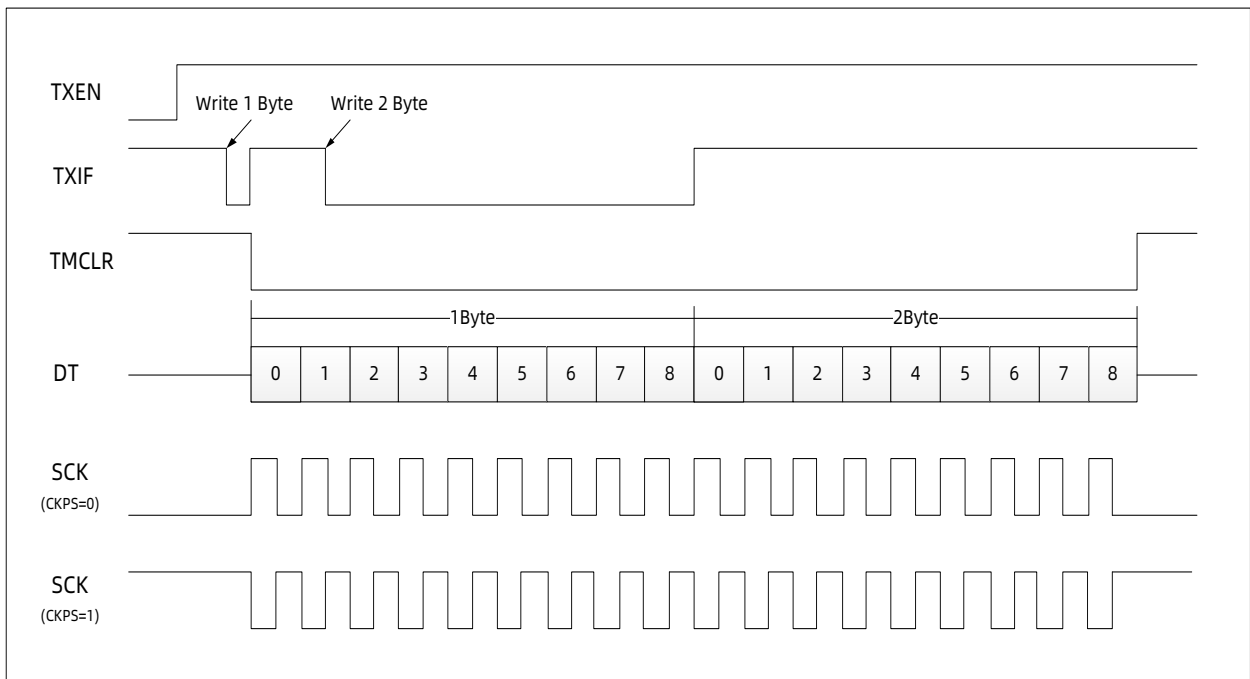
当 TXIF 为 0 时写入 TXOREG, 将覆盖上次写入数据。



单字节发送:



多字节发送:



参考操作步骤 SLAVE = 0:

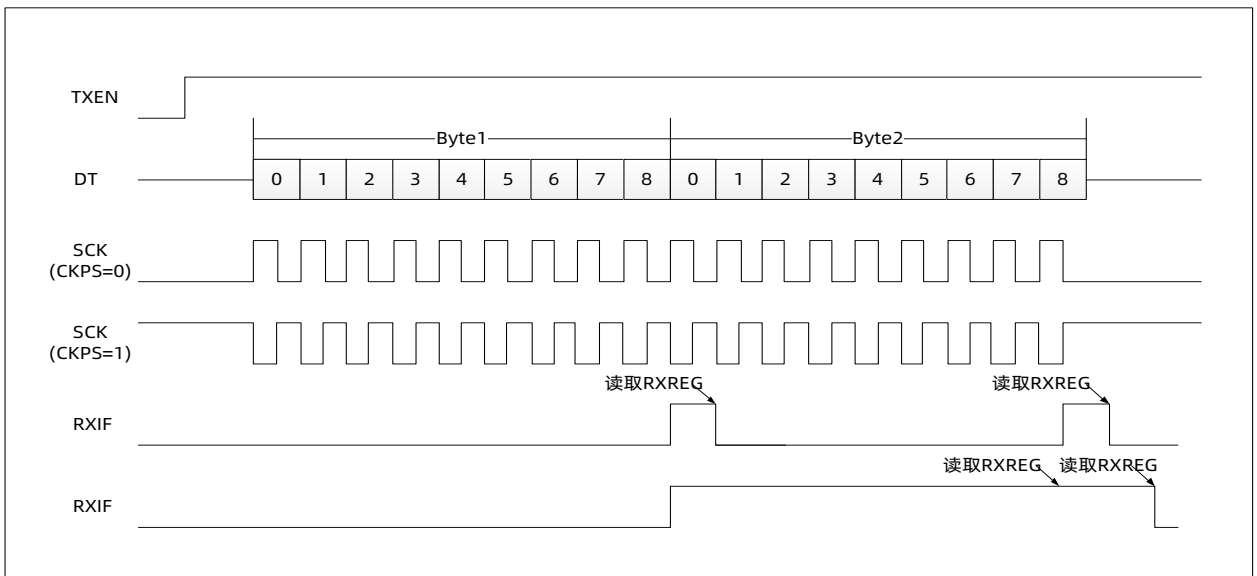
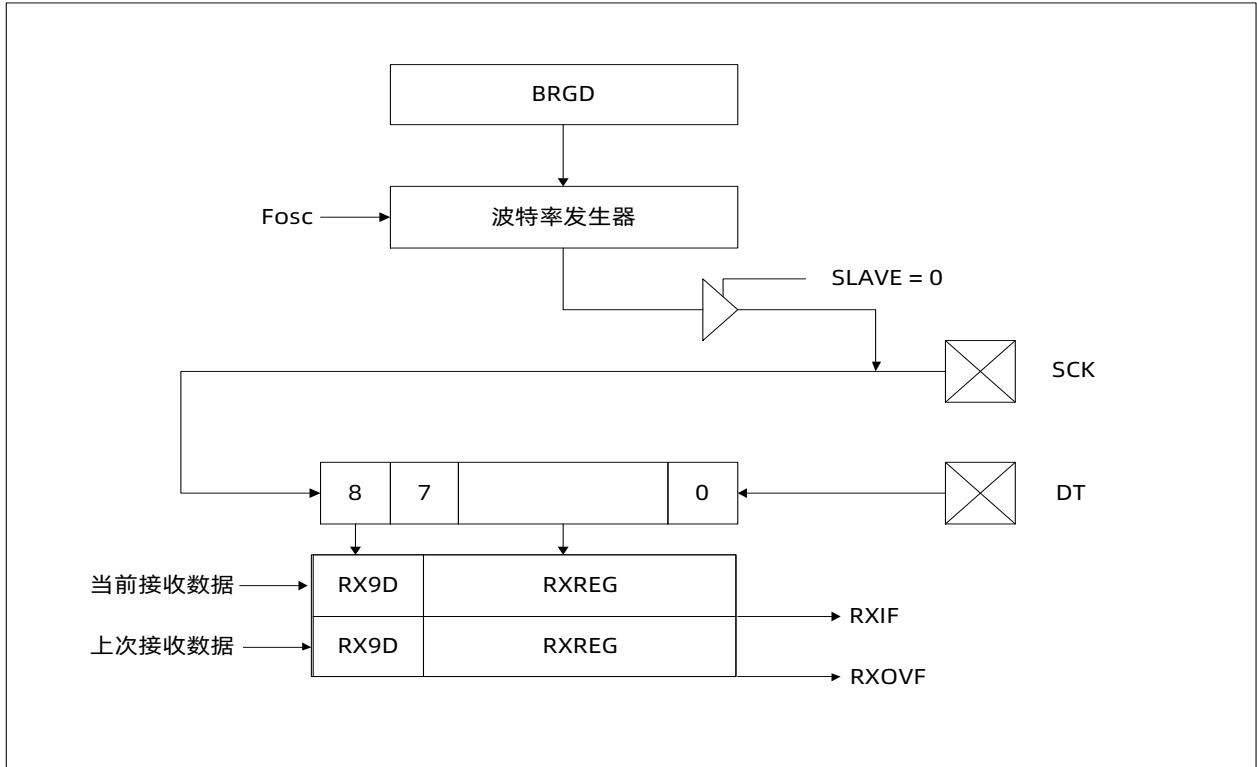
- STEP1: 设置波特率 SSPD = X, BRGD = X, SYNC = 1
- STEP2: 设置 TXEN = 1, 设置数据模式 TX9 = X
- STEP3: 写入数据高位 TXD9
- STEP4: 写入 TX0REG, 启动发送
- STEP5: 当 TXIF = 1 时, 写入 TX0REG 发送下一个字节
- STEP6: 重复 STEP5, 直到该帧数据发送完成

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
- STEP4: 写入 TX0REG 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，开始启动同步接收。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 RXOREG 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 RXOREG, 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 RXOREG, 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 中断标志可唤醒 CPU。

RXIE 置 1 时, RXIF 中断标志可唤醒 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 I2C

## 11.1 概述

I2C 总线作为微控制器与 I2C 设备之间的串行接口。连接在 I2C 总线上的多个设备之间可以相互交换信息。

I2C 总线使主机和从机之间数据可以双向传输。没有中心主机，通过仲裁同时允许多主机系统。同步串行时钟允许器件之间通过一条串行总线以不同位率传输。I2C 总线支持 4 种传输模式，包括主机发送模式，主机接收模式，从机发送模式和从机接收模式。I2C 接口仅支持 7 位寻址模式和广播呼叫。

I2C 的 SDA、SCL 口可以复用到任意 IO 口，详见章节 6.9。

## 11.2 功能描述

对于双向传输操作，SDA 和 SCL 引脚必须开漏模式。该接口最基本的操作是执行线与功能。当一个或多个 I2C 器件输出 0 时，I2C 总线上为低电平。当所有 I2C 器件输出 1 时，产生高电平，允许上拉电阻将总线拉高。

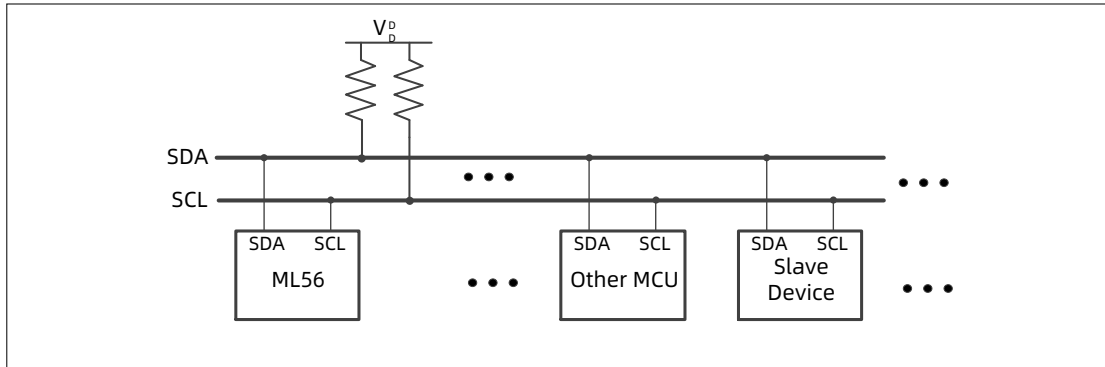


图 11.2-1 I2C 总线连接图

当两线都为高时，I2C 空闲。同时，任何器件可作为主机占用总线并在产生起始信号后传输，在发送停止信号结束传输之前，总线视为忙。主机产生所有串行时钟脉冲和起始与停止信号。然而如果总线上没有起始信号，所有器件均作为没被寻址的从机。硬件寻找自己的从机地址或广播呼叫地址。（广播呼叫地址检测由 GC 使能或禁止。）如果接收到的地址匹配时，请求中断。

I2C 总线上的每个传输为 9 位长度，由 8 位数据（先 MSB）和一个应答位组成。每次传输的字节数（一次有效的 START 与停止信号之间）不受限制，但每个字节后都跟随着一个应答位。主机产生 8 个时钟脉冲发送 8 位数据，在 SCL 总线上的第 8 个下降沿，器件将 SDA 改变输出为输入，并在第 9 个时钟脉冲读应答值。第 9 个时钟脉冲之后，如果下一次接收还没有准备好，数据接收器件将 SCL 总线拉低，从而迫使下一字节的传输暂停。当接收器释放 SCL 总线，数据传输继续。

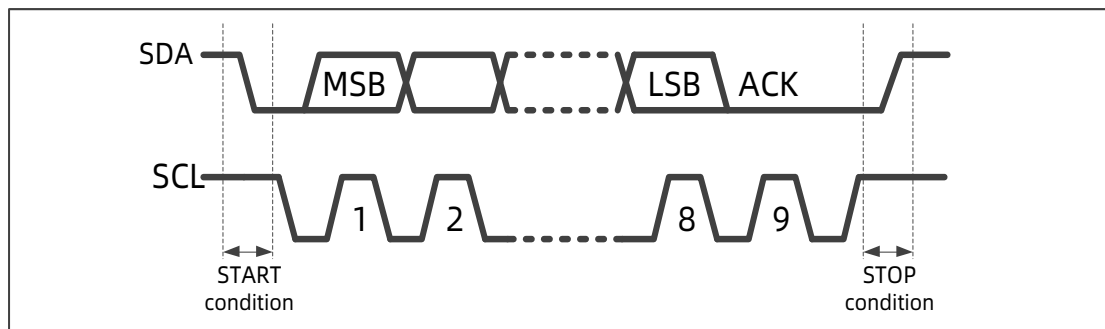


图 11.2-2 I2C 总线数据协议图



## 11.2.1 起始 START 和停止 STOP 信号

I2C 总线协议定义两个状态开始和结束传输，起始(S) 和停止(P) 信号。起始信号：当 SCL 为高时，在 SDA 总线上有从高到低的电平变化。停止信号：当 SCL 为高时，在 SDA 上有从低到高的电平变化。起始或停止信号常由主机产生，在起始信号产生之后 I2C 总线视为忙，在停止信号之后，I2C 总线视为空闲。停止信号出现后，主机设备将释放控制权并返回为无寻址从机。因此，原来寻址的从机将变成未被寻址的从机，I2C 总线空闲并等待下一个起始信号。

通常由主机产生停止信号中止数据传输，然而，如果主机仍希望在总线上通信，就会重复产生起始信号(重复 START) 和地址。传输中可能存在着各种结合的读/写格式。

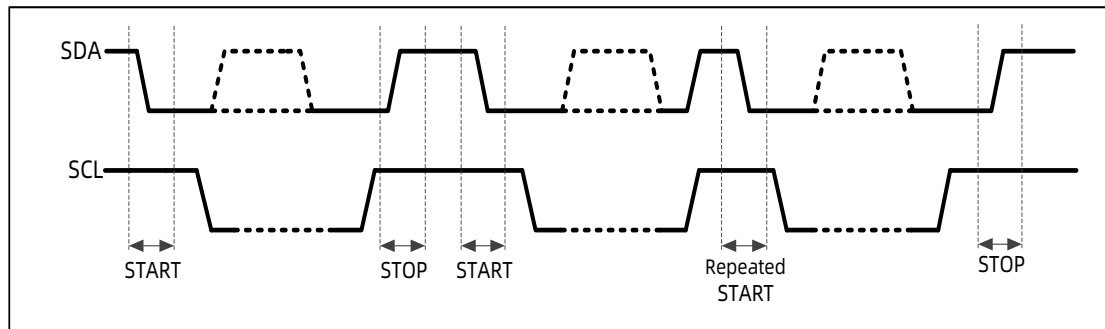


图 11.2-3 起始 START、重复起始 Repeated START 和停止 STOP 信号

## 11.2.2 7 位地址数据格式

产生起始信号之后，应该由主机传输一个字节的特殊数据，包括在第 8 位数据方向位(R/W)之后的 7 位从机地址(SLA)，寻址目标从机并决定数据流的方向。如果 R/W 位为 0，表示主机向所选从机写信息，如果该位为 1，表示主机从从机读取信息。一个地址包由从机地址和读(R)或写(W) 位组成，分别称之为 SLA+R 或 SLA+W。一次传输基本上由一个起始信号，一个 SLA+R/W，一个或多个数据包和一个停止信号组成。在 SLA+R/W 指定从机地址后，第二个和之后的 8 位数据字节由主机或从机地址根据 R/W 位配置。

“广播呼叫”是个例外，它可以通过将第一个字节的数据全部赋值为 0 来寻址所有器件。广播呼叫用于当主机希望发送相同信息到几个从机时情况。当该地址在使用时，其他器件根据软件配置可能响应应答或忽略。如果器件响应广播呼叫，其操作就像从机接收器模式。

在数据传输过程中，SDA 总线上的数据必须在时钟为高的期间内保持稳定，数据总线仅在 SCL 为低时改变。

## 11.2.3 应答

任何传输字节的第 9 个 SCL 脉冲被视作应答信号(ACK)。通过将 SDA 拉低，允许接收器件(可以为主机或从机)对发送器件的响应(可以为主机或从机)。应答由主机产生相关时钟脉冲，发送器件必须在应答时钟脉冲期间释放 SDA 总线控制，ACK 为低电平有效信号，在时钟脉冲为高时将 SDA 总线拉低，通知发送器接收器已经接收到数据。通常，被寻址的接收器在接收到一个字节之后要求产生一个 ACK，当从机接收器没有应答(NACK)从机地址，SDA 线必须由从机拉离高电平，以让主机产生停止或重复起始信号。

如果从机接收器应答了从机地址，将切换到不寻址从机模式，不再接收任何数据，该从机保持 SDA 总线为高，主机应该产生停止信号或复位起始信号。

如果主机接收器传输时，由于主机在传输时控制字节数目，就必须向从机发送器标记数据的结束，而不是在最后一个字节产生应答信号。从机发送器切换到不寻址模式，并释放 SDA 总线，允许主机产生停止信号或复位起始信号。

## 11.2.4 仲裁

主机仅在总线空闲时开始传输，可能是两个或更多主机产生起始信号。在这种情况下，当 SCL 为高时，就要在 SDA 总线上有仲裁。仲裁期间，相互竞争的第一个主机设备在 SDA 上置 1(高)，其他主机发送 0(低)，由于其电平与自身电平不匹配而关闭其数据输出，仲裁失败的主机立即切换成不被寻址从机，并检测自身的从机地址以判断是否被仲裁胜出的主机寻址，它也释放 SDA 为高电平，以防影响胜出主机开始数据传输。但是，仲裁失败的主机继续在 SCL 上产生时钟脉冲，直到它失去仲裁的最后字节输入。如果检测的地址与仲裁失败的主机的自身从机地址相匹配，它就会切换到被寻址的从机模式。

输出数据后，所有主机进行仲裁以持续侦测 SDA 总线。如果从 SDA 总线上读取的值与主机输出的值不匹配时，就失去仲裁。注，当一个主机在其他主机输出低时，自身输出一个高的 SDA 值时就可能失去仲裁，只要有一个主机保存，仲裁就将继续，这可能会占用很多位。如果几个主机试图寻址相同从机时，仲裁将继续输入数据包。

仲裁可以超过几位，第一阶段是比较地址位，如果主机试图寻址相同器件，仲裁继续比较数据位或应答位。

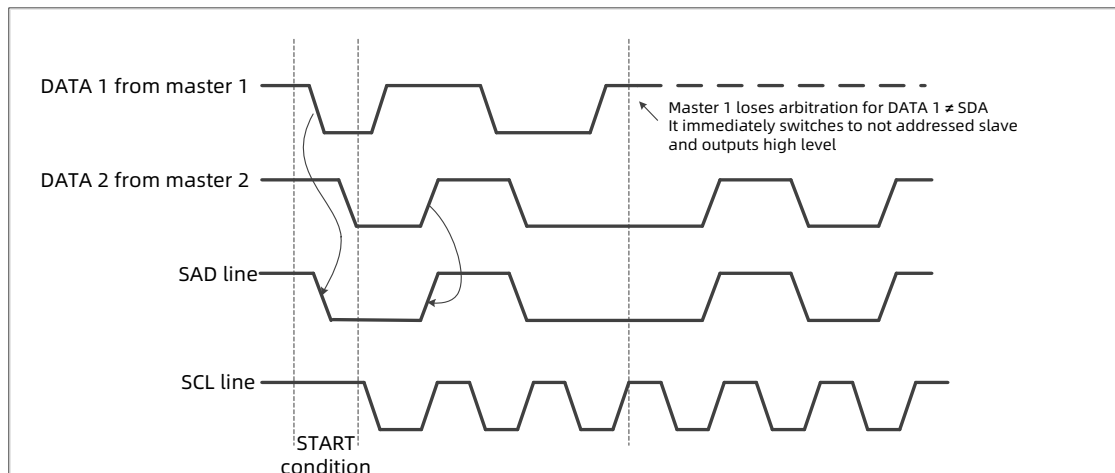


图 11.2-4 两台主机仲裁过程

I2C 总线的这种仲裁机制，让总线上的设备可以有多个主机，而且没有优先等级。从机不介入仲裁。

## 11.3 工作模式

I2C 协议定义了四种模式：主机发送，主机接收，从机发送和从机接收。还有一种特殊模式广播呼叫模式，其操作方式与从机接收模式类似。

### 11.3.1 主机发送模式

在主机发送模式下，向从机接收器发送数据。主机通过 CR[2:0]设置期望时钟速率并向 I2CEN 写 1 使能 I2C 总线，设置 ST 为 1 进入主机发送模式，只要总线空闲，硬件将测试总线并产生起始信号，成功产生起始信号后，SI 标志将置 1 且 I2CSTA 的状态码为 08h，之后就是给 I2CDATA 载入目标从机地址和数据方向位“写” (SLA+W)，SLA+W 开始传输时 SI 位必须清零。

在 SLA+W 字节被发送且由被寻址的从机器件返回应答(ACK)后，SI 标志再次置 1 且 I2CSTA 读值为 18h，依据用户定义的通信协议持续发送数据，全部数据发送完成后，主机可以通过设置 STO 发送停止信号并清 SI 中止传输，在没有发送停止信号下重复起始信号（重复 START）将立即开始新一轮传输。

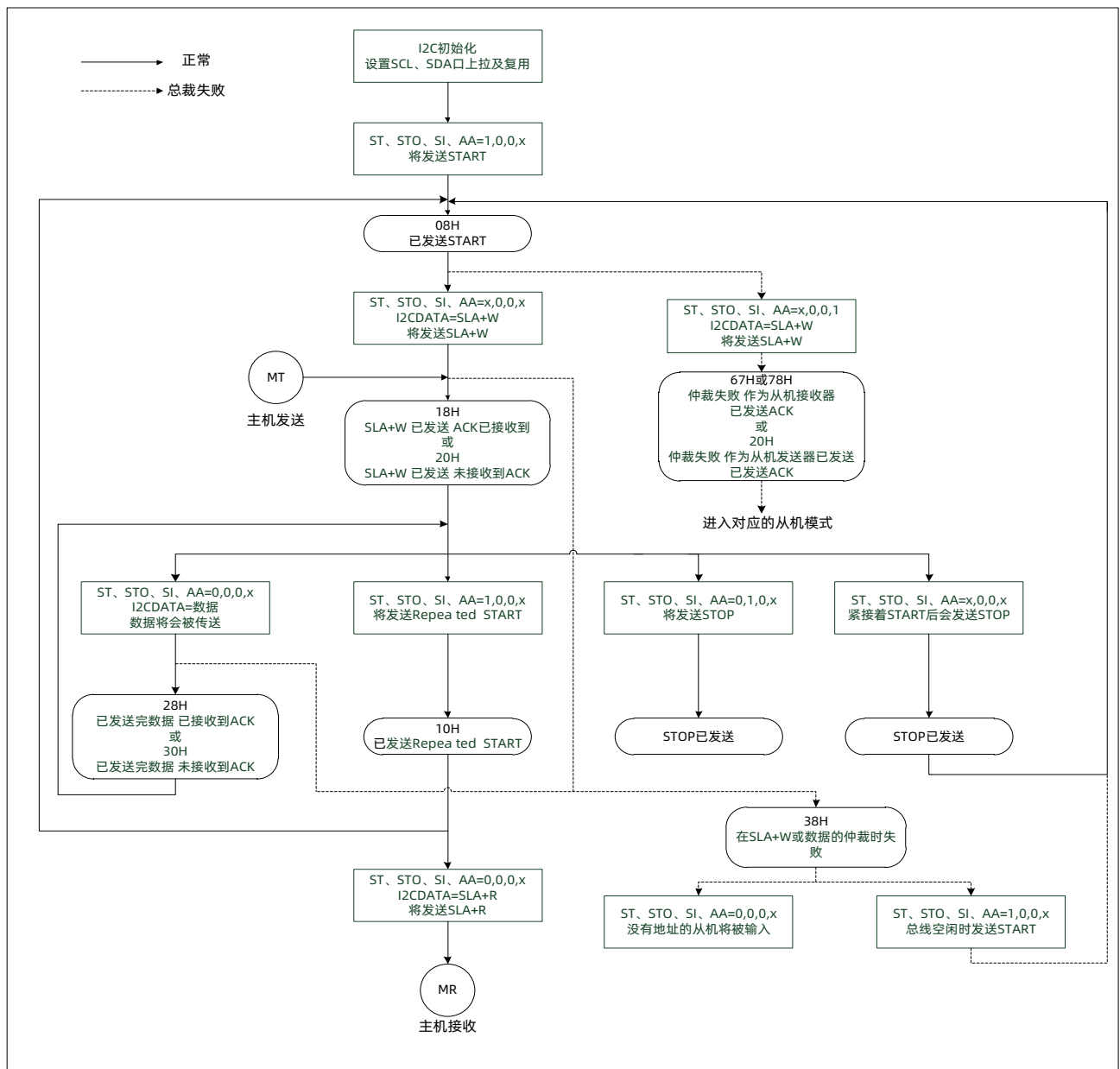


图 11.3-1 主机发送模式流程与状态图

### 11.3.2 主机接收模式

在主机接收模式下,从从机发送器接收数据。传输开始与主机发送模式相似,在起始信号之后,I2CDATA应该加载目标从机地址和数据方向位“读”(SLA+R),SLA+R 字节发送后,且返回应答位 AA,重新置位 SI 标志且 I2CSTA 读数为 40h,SI 标志应该被清零以便开始接收从机发送过来的数据,如果 AA 置 1,主机接收器接收到数据后将应答从机发送器,如果清零 AA,主机接收器接收到数据后将不会应答从机,并释放从机发送器为不被寻址的从机,然后,主机产生停止信号中止传输或重复起始信号开始新一轮传输。

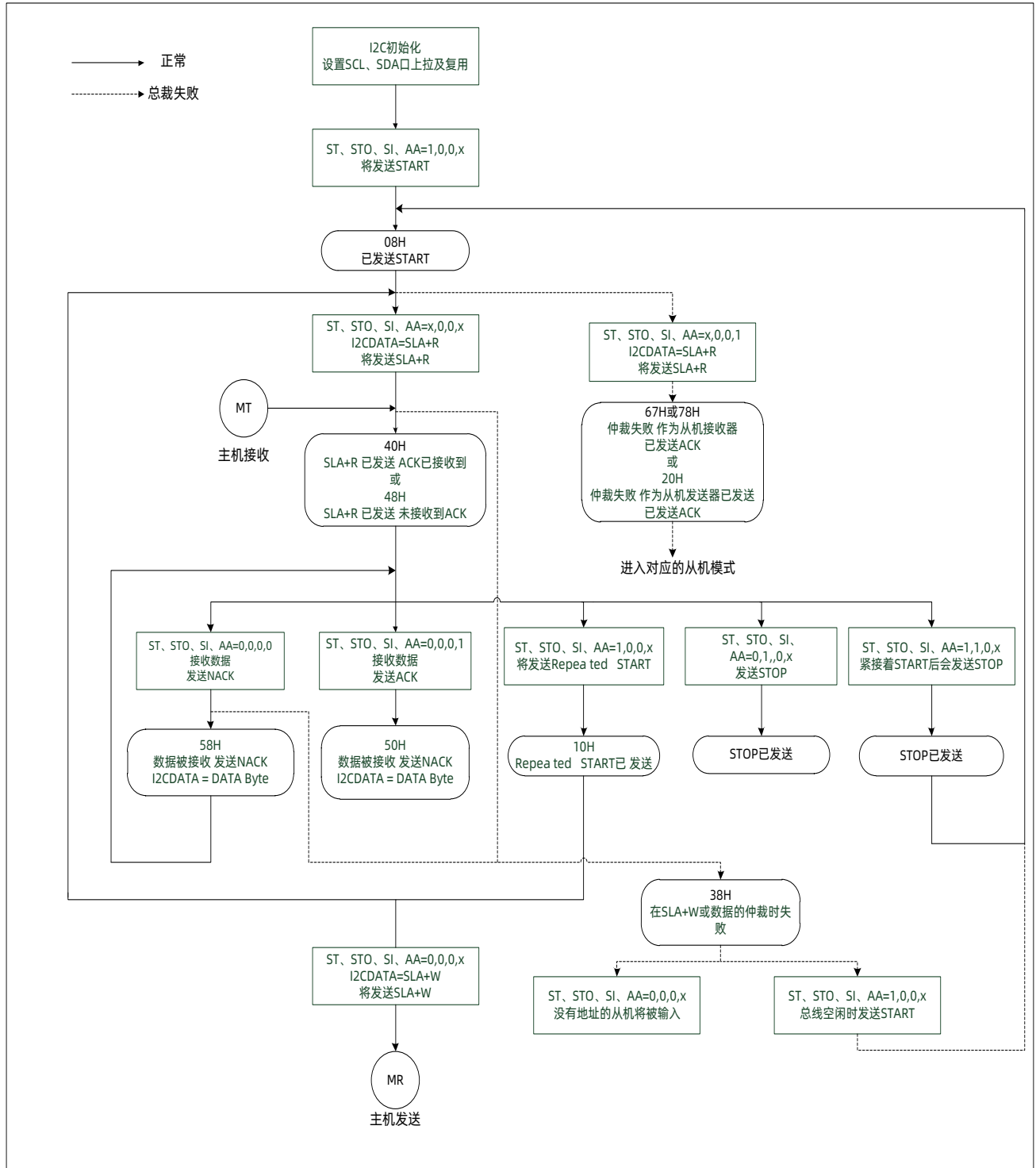


图 11.3-2 主机接收模式流程与状态图

### 11.3.3 从机接收模式

在从机接收模式下，从主机发送器接收数据。发送开始之前，I2CADDR 必须装载响应器件的地址，以让主机寻址，从机模式下 CR[2:0]无效，AA 位必须设置使能应答自身从机地址或广播呼叫，完成以上初始过程后，I2C 等待自身地址被寻址与数据方向位“写” (SLA+W)或被广播呼叫寻址。如果在仲裁失败时，也可以进入从机接收模式。

在从机被 SLA+W 寻址后，需清 SI 标志以便接收主机发送过来的数据，传输期间，如果 AA 位为 0，从机将在下一次接收到的数据字节之后返回无应答 NACK，从机也不被寻址并与主机分离，不能接收 I2CDATA 的任何字节，而保持当前接收到的数据。

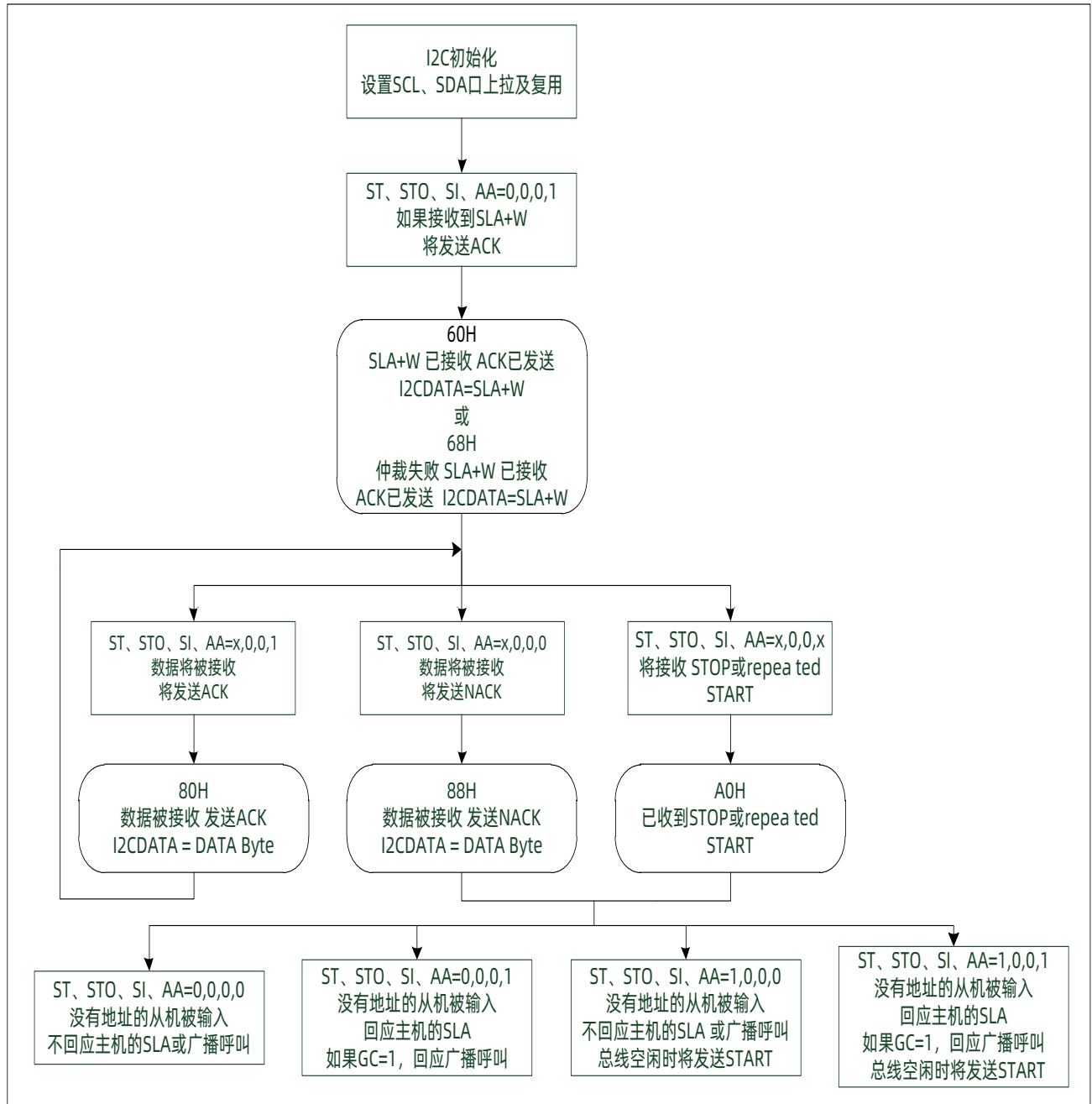


图 11.3-3 从机接收模式流程与状态图

### 11.3.4 从机发送模式

在从机发送模式下，发送几个字节数据到主机接收器。确定 I2CADR 和 I2CCR 的值之后，I2C 等待自己的地址被寻址“读”(SLA+R)。如果仲裁失败后，也可以进入从机发送模式。

在从机被 SLA+R 寻址后，应该清 SI 标志以便传输数据到主机发送器，通常主机接收器将在从机发送每个字节数据之后返回应答，如果没有接收到应答，如果继续传输将发送全 1，并成为不被寻址的从机，如果在传输中清了 AA 标志，从机发送最后一个字节数据，下一次传输数据全为 1，从机变为未定址从机。

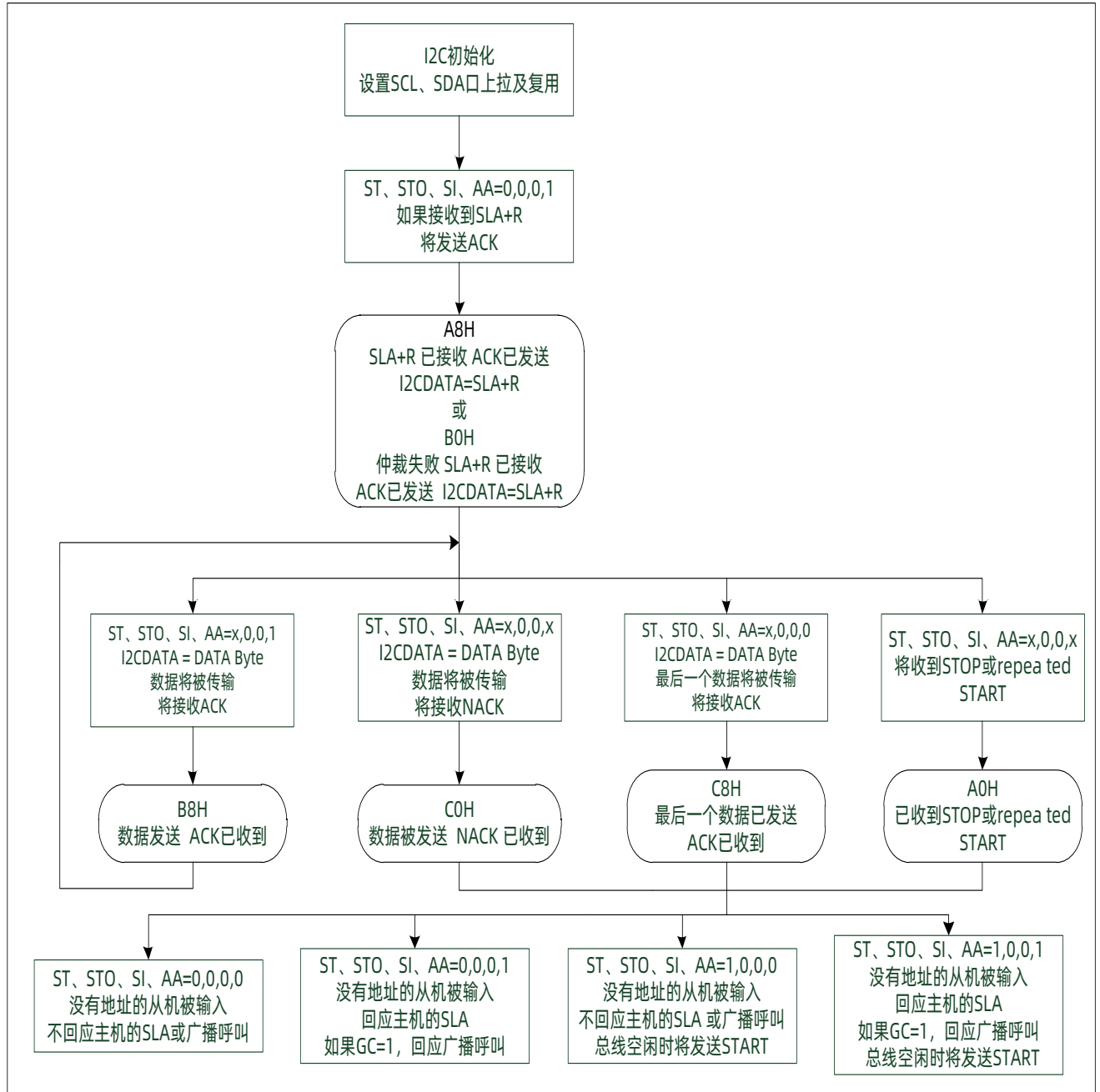


图 11.3-4 从机发送模式流程与状态图

### 11.3.5 广播呼叫

广播呼叫是从机接收模式的一种特殊情况，即从机地址和数据方向位全为 0，且 GC 及 AA 都置 1，使能接收广播呼叫模式。被广播呼叫寻址的从机在正常从机接收模式的 I2CSTA 里有不同状态码，如果仲裁失败也可以进入广播呼叫模式。

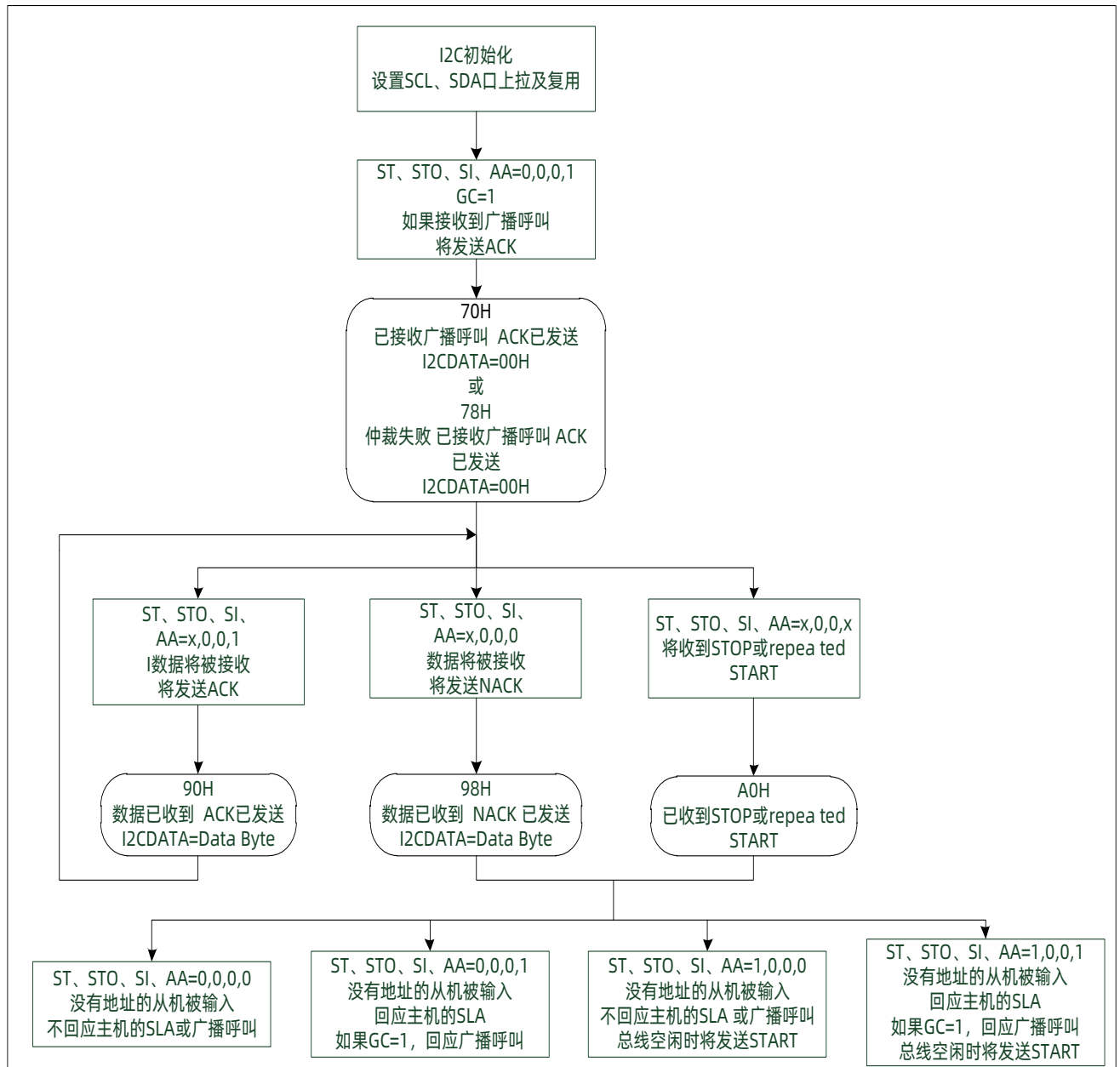


图 11.3-5 广播呼叫模式流程与状态图

### 11.3.6 其它状态

有两个 I2CSTA 状态码与 25 个定义状态不一致，即前面提到的 F8h 和 00h 状态。

第一个状态码 F8h 表示在每次传输期间没有得到相关信息，同时，SI 标志为 0 且没有 I2C 中断请求。

另一个标志码 00h 意味在传输过程中发生错误，总线错误是由 START 或停止信号暂时出现在一个非法的位置，如地址字节里第 2 位换到第 8 位，或数据字节包括应答位，当出现总线错误时，SI 标志立即置 1，当在 I2C 总线上检测到总线错误，工作器件立即切换到不被寻址从机模式，释放 SDA 和 SCL 总，置位 SI 标志，将 00H 载入 I2CSTA。要从总线错误恢复，STO 位必须设置为逻辑 1 且 SI 必须清零，然后，STO 由硬件清零且在没有停止信号就释放 I2C 总线。

特例：如果没有成功产生 START 或重复起始信号，I2C 总线被 SDA 的低电平阻挡，如一个从机器件没有位同步，可以通过在 SCL 总线上发送额外时钟脉冲解决这个问题。当 ST 位置 1 时，I2C 硬件发送额外时钟脉冲，但是由于 SDA 被拉低，不能产生起始信号，当 SDA 总线最终被释放，发送一个普通的 START 条件，进入状态 08h，继续进行串行传输。当 SDA 为低，如果发送重复起始信号，I2C 硬件也执行以上相同的动作。此情况下，在成功发送起始信号后，进入状态 08h，而不是进入 10h。注软件不能解决这类总线问题。



## 11.4 I2C 控制寄存器

### 11.4.1 I2C 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CCR	CR2	I2CEN	ST	STO	SI	AA	CR1	CR0
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, Bit [1:0] **CR[2:0]: 时钟速度控制位**

CR[2:0]	时钟速度控制	CR[2:0]	时钟速度控制
000	$7.625 * F_{CPU} * 10^{-3} \text{bit/s}$	100	$2.225 * F_{CPU} * 10^{-3} \text{bit/s}$
001	$10.4 * F_{CPU} * 10^{-3} \text{bit/s}$	101	$17.85 * F_{CPU} * 10^{-3} \text{bit/s}$
010	$15.625 * F_{CPU} * 10^{-3} \text{bit/s}$	110	$37.025 * F_{CPU} * 10^{-3} \text{bit/s}$
011	$31.25 * F_{CPU} * 10^{-3} \text{bit/s}$	111	禁止使用

**注：速度不能超过 100kbps。**

Bit 6 **I2CEN: 模块使能位**  
 0 = 关闭 I2C  
 1 = 使能 I2C

Bit 5 **ST: I2C 产生起始信号**  
 0 = 不产生起信号  
 1 = 总线空闲, 将产生起始信号, 总线忙, 等待总线停止信号后, 再产生起始信号。主机模式下, I2C 准备好, 在发送或接收一个或多个字节时, 将在总线上产生一个重复起始信号。

**注：ST 可在任何时候置 1, 但在从机模式下, ST 不能在检测到总线起始信号或重复起始信号后自动由硬件清零, 必需由软件清零。**

Bit 4 **STO: I2C 产生停止信号**  
 0 = 不产生停止信号  
 1 = 主机模式时产生停止信号, 当检测到总线上出现停止信号。I2C 硬件清除 STO 标志。STO 标示的设置也用于将 I2C 设备从错误状态 (I2CSTA 为 00h) 恢复, 此条件下, 没有停止信号发送 I2C 总线上。若 ST 和 STO 都置 1, 且在主机模式下设备为原始的, I2C 总线将产生停止信号并立即伴随着起始信号。如果设备为从机模式, 置 STO 恢复到非寻址从机, STO 将会硬件清零。

Bit 3 **SI: I2C 中断标志位**  
 0 = 未产生中断  
 1 = 产生中断 (状态码 F8h 不会产生此标志)

**注：I2C 所有 26 种状态 (释放总线 F8H 除外) 出现一种, 硬件置 1, 用户可根据 ST 数值来判断 I2C 执行到哪一步。**

Bit 2 **AA: 应答位**  
 0 = 应答 NACK  
 1 = 应答 ACK

## 11.4.2 I2C 状态寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CSTA	I2CSTA7	I2CSTA6	I2CSTA5	I2CSTA4	I2CSTA3	-	-	-
读/写	R/W	R/W	R/W	R/W	R/W	-	-	-
复位后	1	1	1	1	1	-	-	-

Bit [7:3] **I2CSTA[7:3]:**

主机模式		从机模式	
状态	描述	状态	描述
0x08	开始信号	0xA0	从机发送重复开始或停止信号
0x10	主机重复开始信号	0xA8	从机发送地址应答
0x18	主机发送地址应答	0xB0	从机发送仲裁失败
0x20	主机发送地址无应答	0xB8	从机发送数据应答
0x28	主机发送数据应答	0xC0	从机发送数据无应答
0x30	主机发送数据无应答	0xC8	从机发送最后数据应答
0x38	主机仲裁失败	0x60	从机接收地址应答
0x40	主机接收地址应答	0x68	从机接收仲裁失败
0x48	主机接收地址无应答	0x80	从机接收数据应答
0x50	主机接收数据应答	0x88	从机接收地址无应答
0x58	主机接收数据无应答	0x70	广播呼叫模式地址应答
0x00	总线错误	0x78	广播呼叫模式仲裁失败
-	-	0x90	广播呼叫模式数应答
-	-	0x98	广播呼叫模式数据无应答
0xF8		释放总线	

## 11.4.3 I2C 数据寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CDATA	I2CDATA[7:0]							
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

### 11.4.4 I2C 从机地址寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
I2CADR	I2CADR[7:1]							GC
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 [7:1]     **I2CADR[7:1]**: I2C从机地址

    主机模式: 无效

    从机模式: 存放7位从机地址

Bit 0

**GC**: 广播呼叫位

    主机模式: 无效

    从机模式: 0 = 广播呼叫模式忽略, 不响应

    1 = 如果AA置1, 参与广播呼叫模式, 若AA清零, 忽略广播呼叫

# 12 SPI

## 12.1 概述

SPI 模块允许在 MCU 和外设(包括其他 MCU)之间进行全双工、同步、串行通信。该模块可以被编程为作为主设备或从设备工作。

- 全双工模式
- 三线同步传输
- 主从模式
- 多个 SPI 波特率
- 可编程极性和相位的串行时钟
- 具有 MCU 中断能力和主模式故障错误标志
- 写入冲突标志保护
- 8 位数据首先传输最高位 (MSB)，最后传输最低位 (LSB)
- 从机模式通过控制端口来控制从机设备

## 12.2 SPI 框图

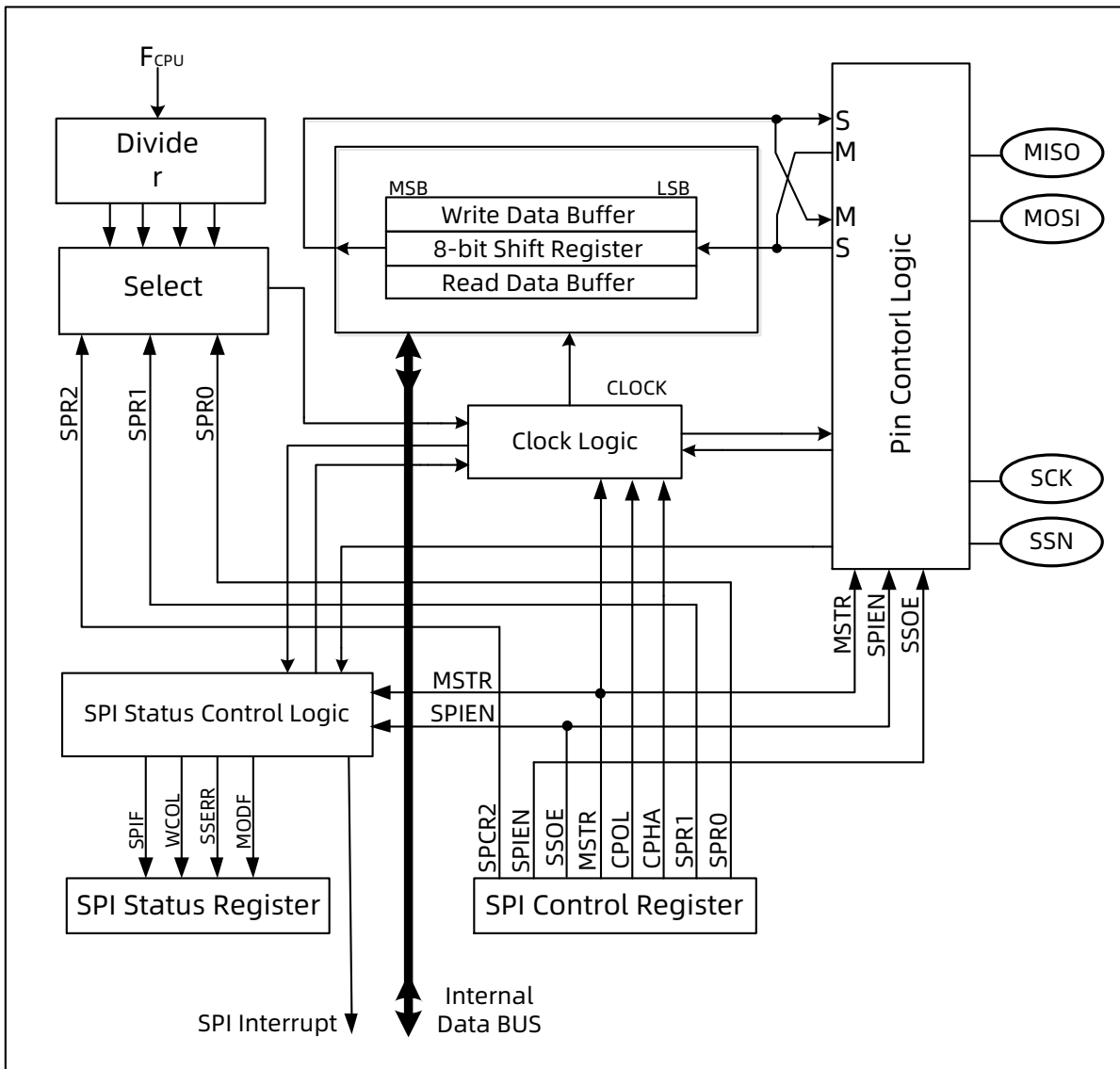


图 12.1 SPI 结构图

## 12.3 功能描述

SPI 结构图展示了 SPI 的体系结构。SPI 寄存器板块是 SPI 模块的主要组成部分，包括逻辑控制，波特率控制和管脚逻辑控制，SPI 包括移位寄存器和读出数据缓冲器，传送数据是单缓冲器，接收数据是双缓冲器。在传送完成之前传送的数据不能写入移位装置。

SPI 需要四个管脚主进/从出(MISO), 主出/从进(MOSI), 移位时钟(SCK), 和从机选择(SSN)。MOSI 脚用于传输主机到从机的 8 位数据, MOSI 是一个主机设备的输出引脚, 从机设备的输入引脚。相应的, MISO 用于接收从机到主机的串行数据。

SCK 引脚为主机模式下的时钟输出, 从机模式的时钟输入。移位时钟用于 MOSI 和 MISO 脚之间数据传输的时钟同步。位移时钟由主机输出, 一组 SPI 传输系统上只能有一个主机以避免设备冲突。建议该管脚设置为史密特触发输入模式。

每路从机外设通过设定从机选择脚(SSN)使能。当需要读取任何从机时, 该信号脚必须保持低。当 SSN 为高, 从机读写将被禁止。若为多从机模式, 在同一时刻必须只有一个从机保持此低电位, 对于主机 SSN 脚不做任何用途, 可配置为普通端口另做他用。SSN 可用于多主机模式下错误侦测功能。

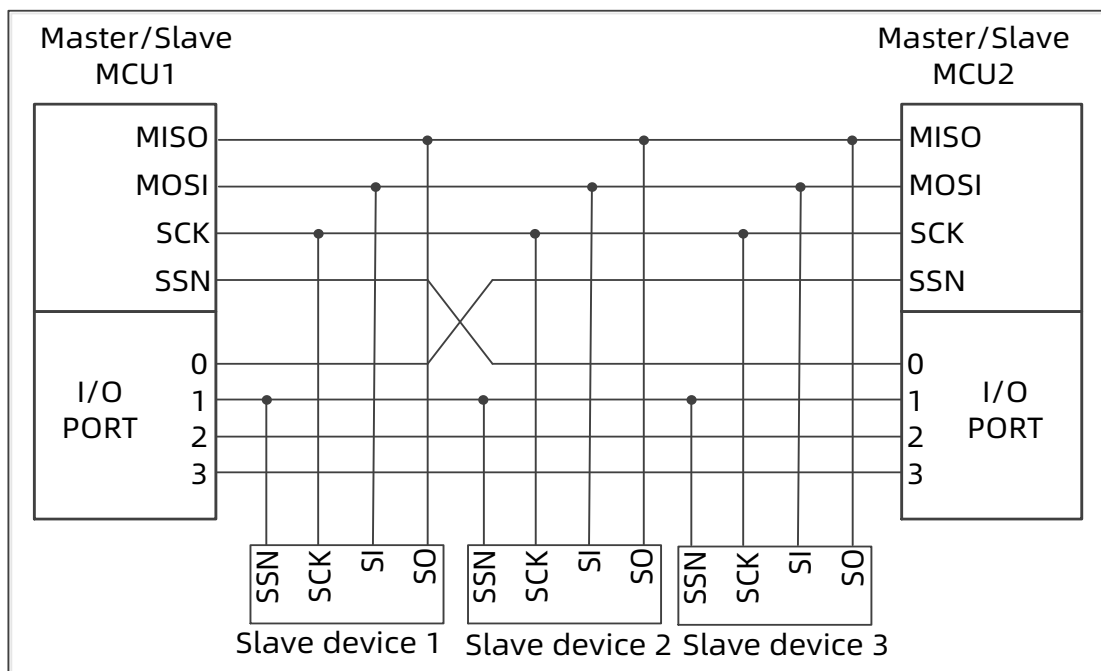


图 12.2-1 SPI 多主机多从机连接图

典型的 SPI 设备通信总线通常为 3 信号线相连, MOSI ~ MOSI, MISO ~ MISO 和 SCK ~ SCK. 主机通过四线并行连接的方式, 每根 SSN 线分别控制每个从机。MCU1 和 MCU2 可以任意定义为主/从机模式. 需配置为主机模式侦测功能避免多主机冲突。

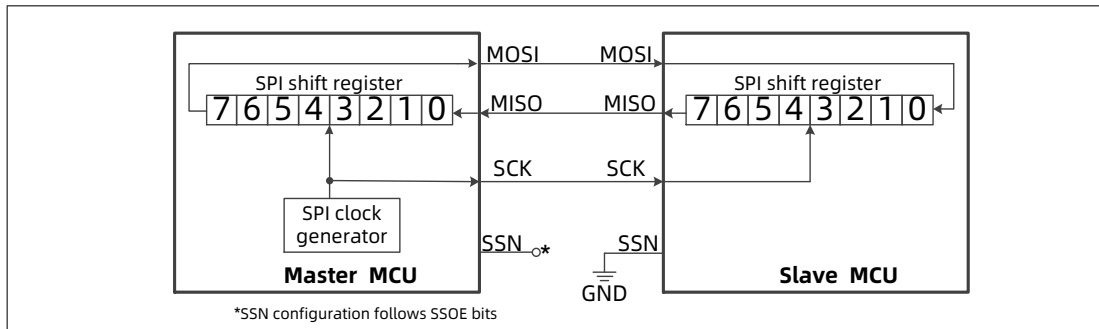


图 12.2-2 SPI 单主机单从机连接图

在传输时, 主机通过 MOSI 线向从机发送数据。同时, 主机也通过 MISO 线由从机接收数据。此时主机和从机的两个数据寄存器可被视为一个 16 位的循环位移寄存器。因此, 当主机向从机某地址送数据时, 从机内该地址内的数据同时也由从机推向主机。传输进行了交换的动作。

## 12.4 SPI 寄存器

### 12.4.1 SPI 控制寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SPICR	SPCR2	SPIEN	SSOE	MSTR	CPOL	CPHA	SPCR1	SPCR0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
复位后	0	0	0	1	0	1	0	0

Bit 7, Bit [1:0] **SPCR[2:0]:** 时钟速度控制位

000 =  $F_{CPU}/2$

001 =  $F_{CPU}/4$

010 =  $F_{CPU}/8$

011 =  $F_{CPU}/16$

100 =  $F_{CPU}/32$

101 =  $F_{CPU}/64$

110 =  $F_{CPU}/128$

111 = 主模式下不产生时钟, 当 CPOL 为 1 时, SCKO 输出为高阻态, 否则为低电平。

Bit 6 **SPIEN:** SPI 使能位

0 = 不使能

1 = 使能

Bit 5 **SSOE:** SSN 控制

0 = SSN 有效

1 = SSN 无效

**注: 从机模式下, 当 CPHA 为 0 时, 此位没用。当此位置一, MODF 中断请求就不会产生。**

Bit 4 **MSTR:** 主从控制

0 = 作为从机

1 = 作为主机

Bit 3 **CPOL:** 时钟极性选择

0 = 空闲状态 SCK 为低

1 = 空闲状态 SCK 为高

Bit 2 **CPHA:** 时钟相位选择

0 = 第一个时钟沿采样

1 = 第二个时钟沿采样



### 12.4.2 SPI 状态寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SPISTA	SPIF	WCOL	SSERR	MODF	-	-	-	-
读/写	R	R	R	R	-	-	-	-
复位后	0	0	0	0	-	-	-	-

Bit 7 **SPIF**: 数据传输完成标志

0 = 数据传输未完或置 1 后读取 SPIDATA 清 0

1 = 数据传输完成

Bit 6 **WCOL**: 写冲突位

0 = 未发生写冲突事件或置 1 后读取 SPISTA 清 0

1 = 发生写冲突事件

Bit 5 **SSERR**: 从机接收错误标志

0 = 未发生模式错误或 SPI 重新使能

1 = 发生从机接收错误, 在从机接收过程中 SSN 被取消, 硬件置 1, 清除 SPEN, 此位才能清除 0

Bit 4 **MODF**: 模式错误标志

0 = 未发生模式错误或置 1 后读取 SPISTA 清 0

1 = 发生模式错误, SSN 电平与 SPI 模式不一致, 硬件置 1 且立即切回从机模式

### 12.4.3 SPI 数据寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SPIDATA	SPIDATA[7:0]							
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

## 12.5 工作模式

### 12.5.1 主机模式

对 MSTR 位置 1，芯片作为主机模式开始 SPI 传输模块开始工作。整个 SPI 系统中只允许一个主机启动传输。每次传输总是由主机发起，对主机 SPIDAT 寄存器的写开始传送。在 SCK 控制下在 MOSI 管脚传送数据。8 位数据传输完毕，SPIF 由硬件自动置位以示完成一个字节数据传输，同时由从机接收到的数据传送到 SPIDAT。从 SPIDAT 读出数据后，用户才可以清除 SPIF。

### 12.5.2 从机模式

设定 MSTR 为 0，SPI 将工作在从机模式。当作为从机模式时，SCK 管脚变为输入脚，它将被另外一个主机的 SPI 设备控制，SSN 管脚需要设置为输入，同样，在数据传输完成前保持低电平状态。如果 SSN 变为高电平，SPI 将被迫进入闲置状态。如果 SSN 管脚在传输的过程被置高，那么传输将被取消，同时接受数据的缓存区也将进入闲置状态。

在从机模式下，数据在 MOSI 管脚从主机向从机流动，在 MISO 管脚从从机向主机流动。根据 SCK 的时钟控制数据由主机位移传入，每次一个字节传输完成 SPIF 置 1，此时读取 SPIDATA 寄存器即为该字节内容。对 SPIDATA 的读实际上就是对缓冲器的读。为了防止缓冲器溢出和由于溢出导致的数据丢失，SPIF 必须在数据第二次从移位寄存器向读缓冲器传送前清零。

## 12.6 时钟格式和数据传输

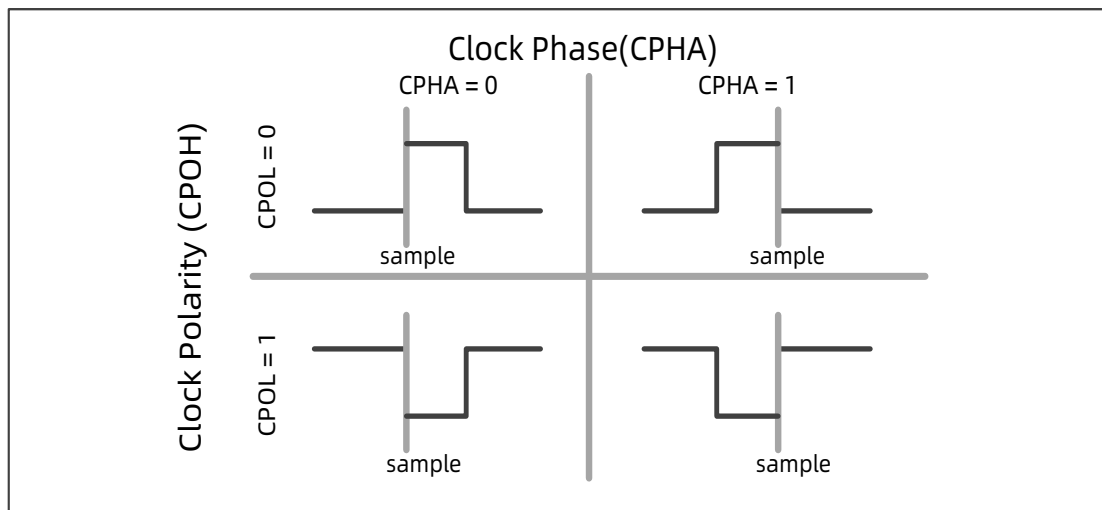


图 12.3 SPI 时钟格式图

为了适应各种各样的同步串行外设，SPI 提供时钟极性位 CPOL 和时钟相位位 CPHA 用以控制。如图 SPI 时钟格式所示，CPOL 和 CPHA 组合出四种不同的时钟格式。CPOL 位表示空闲状态时 SCK 脚电位。CPHA 位表示是由 MOSI 或由 MISO 上那条线的边缘采样。在同一系统上的主从设备中，CPOL 和 CPHA 的应该是相同的，传输不同的数据格式，将产生随机错误结果。

在 SPI 传输中，总是由主机启动传输。如果 SPI 被选定作为主模式（MSTR = 1）并且打开传输（SPIEN = 1），对主机的 SPI 数据寄存器（SPIDATA）写入内容将启动 SPI 时钟和数据传输。传出一个字节的同时会接受一个字节的內容，此后 SPI 时钟停止，主机和从机的 SPIF 同时被置 1。如果 SPI 中断使能位设置为 1，全局中断使能，芯片将执行中断服务程序。

关于从机模式下，SSN 信号需要注意。如时钟格式图所示，CPHA=0 时，第一个 SCK 边沿为 MSB 的采样点。因此，从机必须在 SCK 第一个采样边沿出现之前先把 MSB 传出。SSN 的下降沿可用于准备 MISO 的 MSB。因此，每次成功串行传输一个字节后，该引脚必须切换先高然后低，每个成功逐次串行字节之间。此外，如果从机将数据写入 SPI 数据寄存器（SPIDATA）时，如果 SSN 为低电位，则会发生写冲突错误。

当 CPHA = 1，采样边沿位于 SPCLK 时钟的第二个边沿。从机使用的第一个 SCK 时钟转移的 MSB，而不是 SSN 的下降沿。因此，在每次成功传输时 SSN 可以始终保持低电位保持低之间的转移。此格式更适合单主机单从机的结构使用。从机的 SSN 可以不连接在 SPI 系统中，直接接地。

## 12.7 模式故障侦测

在一个 SPI 网络中，当不止一个设备有可能成为主机时，为减少数据传输错误，模式故障侦测功能是非常有用的。模式故障侦测发现 SSN 由其它设备拉低，说明系统上有一个从机试图寻找主机地址并把注主机认为成从机。此时，硬件自动将 SPICR 的 MSTR 和 SPIEN 清除，从而 SPI 功能关闭，同时使错误侦测标志 MODF 置 1，如果之前已打开中断，则会进入中断向量。

## 12.8 写冲突错误

写冲突检测显示当正在进行一次传送时，设备正在试图写数据到 SPIDATA。SPIDATA 在传送方不是双缓冲器，对 SPIDATA 的写被直接写进 SPI 移位寄存器，如果这种写在转移过程中被误用，将发生一个写冲突错误(WCOL 将被置位)。如果转移连续稳定没有受到干扰，那么导致错误的写数据是没有写进移位装置。一次写冲突通常是一个从机错误，原因是当主机开始一次传送时主机知道传送正在进行，所以主机没有理由产生写冲突错误，尽管 SPI 逻辑可以在主机和从机之间进行写冲突检测。WCOL 标志用软件清除。

SPI 是信号对于接收数据，是双向缓冲的。在前一笔数据传输完成之前不能对传输缓冲写入新的数据。否则对 SPIDATA 写入内容会引起写冲突错误。对于发送端 SPIDATA 不是双缓冲的，此时对 SPIDATA 写入数据被直接写入到 SPI 移位寄存器。一旦产生写冲突错误，WCOL 将通过硬件设置为 1 表示写冲突。在这种情况下，当前的数据传输继续其传送，而新的数据将会丢失。虽然 SPI 逻辑可以检测写在主机模式和从机模式的冲突，写冲突通常是从机错误引起的，因为当主机启动传输时，从机并没有指示一个从机没有指示标志。在从机接收模式，对 SPIDATA 写入内容，也会引起写冲突错误。

## 13 触摸按键（CDC）

30 路触摸按键通道，内置电容无需外接，可替代机械式触摸按键，实现防水防尘，简单易用的操作接口。

**注：**

**（1）TK18 和 TK19 通道需要一起都做触摸功能，或者都不做触摸功能，如果一个脚做触摸另一个脚不做触摸，会导致做触摸引脚被影响。**

**（2）CDC 功能请使用官网对应的库。**

# 14 模数转换器(ADC)

## 14.1 概述

M9F680有28路外部通道(AIN0~AIN27)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,系统自动将转换结果存入寄存器ADH和ADL中。

**注: MCU电源口VDD&GND口并联104电容, 104电容位置应紧靠IC, 电源走线也应先进入104电容再进入MCU。**

## 14.2 ADCON0 寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADCON0	ADEN	ADS	ADFM	CHS[4:0]				
读/写	R/W	R/W	R/W	R/W	RW	R/W	R/W	R/W
复位后	0	0	0	1	0	0	1	1

- Bit 7      **ADEN:** 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      **ADFM:** 数据格式选择位  
           0 = 左对齐, 即ADRES = {ADH[7:0], ADL[7:4]}; ADL[3:0] = 0  
           1 = 右对齐, 即ADRES = {ADH[3:0], ADL[7:0]}; ADH[7:4] = 0
- Bit [4:0]    **CHS[4:0]:** ADC 输入通道选择位  
           00000 ~ 11011 = AIN0 ~ AIN27  
           11100 = VDD/4  
           11101 = VREF  
           11110 = 预留  
           11111 = GND

**注: 需要采集的外部通道, 用户应设置对应的端口为输入模式并且将端口设为模拟端口。**

## 14.3 ADCON1 寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADCON1	AGEN	ADCKS[2:0]			VREMS[1:0]		[VHS1:0]	
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

Bit [7] **AGEN**: 增益控制位

0 = 关闭增益控制

1 = 开启增益控制 (VREM=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 [7:1:0] **VHS[1:0]**: 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

注: 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。

## 14.4 ADCON2 寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADCON2	-	TRGEN	VREFO	-	ADVOS[3:0]			
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 6      **TRGEN:** 触发事件触发AD转换

0 = 不允许

1 = 允许

Bit 5      **VREFO:** 内部参考输出

0 = 关闭内部参考输出

1 = 参考电压从端口输出(VREFO端口)

Bit 4      **Reserved:** 保留, 请保持为0

Bit [3:0]    **ADVOS[3:0]:** ADC失调补偿寄存器



## 14.5 ADCON3 寄存器

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADCON3	AGS[3:0]				-	-	-	-
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

Bit [7:4] **AGS[3:0]**: 增益选择位, 将通道信号放大后进行ADC转换

AGS[3:0]	放大倍数
0000	19.80 倍
0001	10.20 倍
0010	6.94 倍
0011	5.27 倍
0100	4.25 倍
0101	3.33 倍
0110	2.85 倍
0111	2.50 倍
1000	2.22 倍
1001	2.00 倍

Bit [3:0] **Reserved**: 保留, 请保持为0

注: (1) AGEN 建议在 2V 参考下使用。  
 (2) 关于 AGS 应用请参考《ADC 的使用说明及范例》。

## 14.6 ADH/ADL AD 结果寄存器

### 14.6.1 左对齐, ADFM = 0

#### ADH 结果寄存器高字节

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADH	ADREG[11:4]							
读/写	R	R	R	R	R	R	R	R
复位后	x	x	x	x	x	x	x	x

Bit [7:0] **ADREG[11:4]:** 12 位 ADC 结果寄存器高 8 位

#### ADL 结果寄存器低字节

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADL	ADREG[3:0]				-	-	-	-
读/写	R	R	R	R	R	R	R	R
复位后	x	x	x	x	x	x	x	x

Bit [7:4] **ADREG[3:0]:** 12 位 ADC 结果寄存器低 4 位

### 14.6.2 右对齐, ADFM = 1

#### ADH 结果寄存器高字节

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADH	-	-	-	-	ADREG[11:8]			
读/写	R	R	R	R	R	R	R	R
复位后	0	0	0	0	0	0	0	0

Bit [3:0] **ADREG[11:8]:** 12 位 ADC 结果寄存器高 4 位

#### ADL 结果寄存器低字节

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ADL	ADREG[7:0]							
读/写	R	R	R	R	R	R	R	R
复位后	0	0	0	0	0	0	0	0

Bit [7:0] **ADREG[7:0]:** 12 位 ADC 结果寄存器低 8 位

## 14.7 AD 转换时间

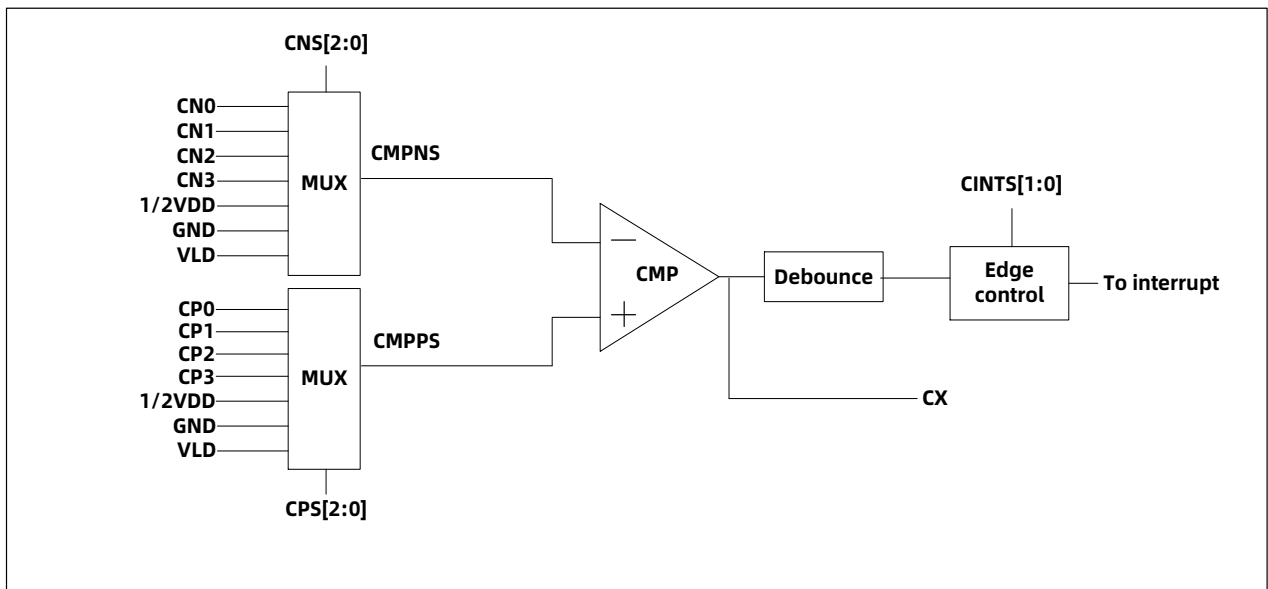
注：12 位 AD 转换时间 = 1/(ADC clock)\*16 sec

# 15 比较器 (CMP)

## 15.1 概述

比较器具有多种输入源、多种参考电压、输出极性可选择、多种输出中断触发和输出信号可唤醒等功能，增强了使用的灵活性，适应各种广泛的应用。

## 15.2 比较器框图



## 15.3 CMPC0 比较器控制寄存器 0

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
CMPC0	CMPEN	CMPOUT	CMPNS[2:0]			CMPPS[2:0]		
R/W	R/W	R	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 **CMPEN**: 比较器使能控制位

0 = 关闭比较器

1 = 使能比较器

Bit 6 **CMPOUT**: 比较器输出位

0 = CP脚输入电压小于CN脚

1 = CP脚输入电压大于CN脚

Bit [5:3] **CMPNS[2:0]**: 比较器反相输入信号选择位

CMPNS[2:0]	输入信号选择
000	CN0
001	CN1
010	CN2
011	CN3
100	1/2VDD
101	GND
110	VLD
111	未定义

Bit [2:0] **CMPPS[2:0]**: 比较器正相输入信号选择位

CMPPS[2:0]	输入信号选择
000	CP0
001	CP1
010	CP2
011	CP3
100	1/2VDD
101	GND
110	VLD
111	未定义

**注:**

(1) 如果端口输入电压超过芯片工作电压范围, 会导致端口高压或负压对 CMP 产生干扰。

(2) 设计者必须在使能比较器中断之前将比较器使能, 以避免未知的中断发生。

## 15.4 CMPC1 比较器控制寄存器 1

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
CMPC1	CMPOEN	COPVRC	-	CMPVLD[4:0]				
R/W	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 **CMPOEN**: 比较器输出使能

0 = 关闭比较器信号输出

1 = 比较器信号从端口输出

Bit 6 **COPVRC**: P端信号输入控制

0 = 无影响

1 = 不论CMPPS为何值, CP0通道导通

Bit 5 **Reserved**: 必需保持为0

Bit [4:0] **CMPVLD[4:0]**: VLD电压选择位

CMPVLD [4:0]	电压 (V)	CMPVLD [4:0]	电压 (V)	CMPVLD [4:0]	电压 (V)	CMPVLD [4:0]	电压 (V)
00000	1.25	01000	1.65	10000	2.05	11000	2.45
00001	1.30	01001	1.70	10001	2.10	11001	2.50
00010	1.35	01010	1.75	10010	2.15	11010	2.55
00011	1.40	01011	1.80	10011	2.20	11011	2.60
00100	1.45	01100	1.85	10100	2.25	11100	2.65
00101	1.50	01101	1.90	10101	2.30	11101	2.70
00110	1.55	01110	1.95	10110	2.35	11110	2.75
00111	1.60	01111	2.00	10111	2.40	11111	2.80

## 15.5 CMPC2 比较器控制寄存器 2

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
CMPC2	CINTS[1:0]		-	-	-	DEB[2:0]		
R/W	R/W	R/W	-	-	-	R/W	R/W	R/W
复位后	0	0	-	-	-	0	0	0

Bit [7:6] **CINTS[1:0]:** 比较器中断触发类型选择

CINTS[1:0]	触发类型
00	下降沿触发
01	上升沿触发
1X	双边沿触发

Bit [2:0] **DEB[2:0]:** 比较器逻辑输出滤波设置

DEB[2:0]	滤波设置
000	关闭
001	4
010	8
011	16
100	32
101	64
110	128
111	256

注:

(1) 滤波时间为  $T_{deb} = T_{CPU} * DEB * 3/4$ 。

(2) 端口的 Cx 输出不经过滤波。

(3) 滤波算法简介:

系统时钟对比较器的输出进行采样，为 1 时滤波计数器加 1。为零时滤波计数器减 1，滤波器初值为设置值的一半，当滤波器计数器  $> DEB * 3/4$  时。滤波结果为 1；滤波器计数器  $> DEB * 1/4$  时，滤波结果为 0。

# 16 LED

## 16.1 概述

该模块由 LED 控制寄存器以及 LED 数据寄存器组成，可以驱动 LED 段码。SEG 口可以复用到所有 IO 口，COM 可以 COM0-COM7 为一组复用到 IOA、IOB、IOC 或 IOD 端口，如 COM0-COM7 可以分别复用到 IOA0-IOA7，具体见 6.9 节 IO 复用。

- 可选择 1-8 个 COM
- 任意 IO 端口都可作为 SEG 口的使用
- 支持 LED 亮度可调节

## 16.2 LED 控制寄存器

控制寄存器 LEDCR0

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
LEDCR0	-	LEDEN	LEDPOL	-	-	LEDFR[2:0]		
读/写	-	R/W	R/W	R/W	-	R/W	R/W	R/W
复位后	-	0	0	0	-	0	0	0

Bit 6      **LEDEN:** LED 功能使能位

0 = 屏蔽 LED 功能

1 = 使能 LED 功能

Bit 5      **LEDPOL:** LED 极性选择

0 = COM 使能为 1，即共阳扫描，对应的 LED RAM 置 0，点亮对应段位

1 = COM 使能为 0，即共阴扫描，对应的 LED RAM 置 1，点亮对应段位

Bit 4      **Reserved:** 保留，请保持为 0

Bit [2:0]    **LEDFR[2:0]:** COM 脉冲宽度

扫描脉冲宽度  $\approx 32 * 512 * T_{OSC} * (LEDFR + 1)$

**控制寄存器 LEDCR1**

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
LEDCR1	-	-	-	BRIGHT[4:0]				
读/写	-	-	-	RW	RW	RW	RW	RW
复位后	-	-	-	1	1	1	1	1

 Bit [4:0] **BRIGHT[4:0]: 亮度控制**

BRIGHT[4:0]	说明
00000	关显示
00001	占空比 1/16
00010	占空比 2/16
00011	占空比 3/16
00100	占空比 4/16
00101	占空比 5/16
00110	占空比 6/16
00111	占空比 7/16
01000	占空比 8/16
01001	占空比 9/16
01010	占空比 10/16
01011	占空比 11/16
01100	占空比 12/16
01101	占空比 13/16
01110	占空比 14/16
01111	占空比 15/16
1xxxx	占空比 16/16

**注：BRIGHT 值通过控制 COM 占空比实现改变显示亮度。**

## 16.3 LED COM 使能寄存器

**控制寄存器 COMIOS**

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
COMIOS	COMIOS[7:0]							
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

 Bit [7:0] **COMIOSx: COM 端口使能位 (x = 0-7)**

0 = 用作 IO 口

1 = 用作 LED COM 输出

**注：COM0-7 口可选择在任意一组 IO 上，详见章节 6.9。**



## 16.4 LED SEG 端口选择寄存器

**SEGIOSx 寄存器 (x = A/B/C/D)**

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
SEGIOSA	SEGIOSA7	SEGIOSA6	SEGIOSA5	SEGIOSA4	SEGIOSA3	SEGIOSA2	SEGIOSA1	SEGIOSA0
SEGIOSB	SEGIOSB7	SEGIOSB6	SEGIOSB5	SEGIOSB4	SEGIOSB3	SEGIOSB2	SEGIOSB1	SEGIOSB0
SEGIOSC	SEGIOSC7	SEGIOSC6	SEGIOSC5	SEGIOSC4	SEGIOSC3	SEGIOSC2	SEGIOSC1	SEGIOSC0
SEGIOSD	-	-	SEGIOSD5	SEGIOSD4	SEGIOSD3	SEGIOSD2	SEGIOSD1	SEGIOSD0
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] **SEGIOSxy**: 端口 IOxy 的 SEG 功能使能位 (x = A/B/C/D, y = 0-7)

0 = 用作 IO 口

1 = 用作 LED SEG 输出

**注: SEGIOSA/SEGIOSB/SEGIOSC/SEGIOSD 分别对应 IOA/IOB/IOC/IOD。**

## 16.5 LED 显示缓存寄存器

数据存储器 BANK2 中的 0x200 - 0x21D 为 LED 显示缓存器, 操作方式与普通 RAM 方式相同。不适用 LED 模块时, LED RAM 可放普通 RAM 使用。

地址	SEG 对应 IO	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
		COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0
LEDDS00	IOA0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0	SEG0
LEDDS01	IOA1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1	SEG1
LEDDS02	IOA2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2	SEG2
LEDDS03	IOA3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3	SEG3
LEDDS04	IOA4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4	SEG4
LEDDS05	IOA5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5	SEG5
LEDDS06	IOA6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6	SEG6
LEDDS07	IOA7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7	SEG7
LEDDS08	IOB0	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8	SEG8
LEDDS09	IOB1	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9	SEG9
LEDDS10	IOB2	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10	SEG10
LEDDS11	IOB3	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11	SEG11
LEDDS12	IOB4	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12	SEG12
LEDDS13	IOB5	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13	SEG13
LEDDS14	IOB6	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14	SEG14
LEDDS15	IOB7	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15	SEG15
LEDDS16	IOC0	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16	SEG16
LEDDS17	IOC1	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17	SEG17
LEDDS18	IOC2	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18	SEG18
LEDDS19	IOC3	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19	SEG19
LEDDS20	IOC4	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20	SEG20
LEDDS21	IOC5	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21	SEG21
LEDDS22	IOC6	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22	SEG22
LEDDS23	IOC7	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23	SEG23
LEDDS24	IOD0	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24	SEG24
LEDDS25	IOD1	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25	SEG25
LEDDS26	IOD2	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26	SEG26
LEDDS27	IOD3	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27	SEG27
LEDDS28	IOD4	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28	SEG28
LEDDS29	IOD5	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29	SEG29

# 17 看门狗 (WDT)

## 17.1 概述

看门狗定时器的时钟为内部独立的低速 RC 时钟源。

芯片配置字中可选择看门狗定时器的的工作模式，共三种模式可选择：

- (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

芯片配置字中可选择看门狗的溢出时间，可选择时间分别为 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。**

# 18 芯片配置字 (OPTION BIT)

烧录选项	内容	说明
振荡器模式	HIRC(16M)+LIRC	双系统时钟 (使用触摸时只能选择 HIRC16M)
	HIRC(16M)+LXT	
	HXT(24M)+LIRC	
	HXT(32M)+LIRC	
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	
晶振反馈电阻	电阻 0(小电阻)	默认电阻 1
	电阻 1(大电阻)	
外部复位端口	作为外部复位端口	低电平有效, 内置上拉
	作为通用 IO 端口	
启动模式选择	低速启动	
	高速启动	
复位电压选择	LVR=1.4V (FCPU<8T)	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=1.5V (FCPU<8T)	
	LVR=1.6V (FCPU<8T)	
	LVR=1.7V (FCPU<8T)	
	LVR=1.8V (FCPU<4T)	
	LVR=1.9V (FCPU<4T)	
	LVR=2.0V (FCPU<4T)	
	LVR=2.1V (FCPU<4T)	
	LVR=2.2V	
	LVR=2.3V	
	LVR=2.4V	
	LVR=2.5V	
	LVR=3.5V	
WDT 工作模式选择	屏蔽 WDT 功能	
	使能 WDT, 绿色或休眠模式下关闭	
	始终开启 WDT 功能	

烧录选项	内容	说明
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	
3000-37FF 写保护	不允许擦写	
	允许擦写	
3800-3FFF 写保护	不允许擦写	
	允许擦写	
仿真电压 选择	VDD5.0(<200mA)	
	VDD3.3(<300mA)	
	外供电源	

# 19 电性参数

## 19.1 极限参数

储存温度.....	-50°C ~ 125°C
工作温度.....	-40°C ~ 85°C
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。**

## 19.2 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°C)				
VDD	工作电压	—	Fosc = 16MHz, 32T	1.4	-	5.5	V
IDD1	工作电流 1	3V	高频运行 (HIRC=16M) 低频运行 (LIRC=64K) F <sub>CPU</sub> =HIRC/16T 全速工作	-	0.4	-	mA
		5V		-	0.7	-	
IDD2	工作电流 2	3V	高频运行 (HIRC=24M) 低频运行 (LIRC=64K) F <sub>CPU</sub> =HIRC/16T 全速工作	-	0.5	-	mA
		5V		-	0.8	-	
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	-	0.3	-	mA
		5V		-	0.5	-	
ISP2	静态电流 2	3V	高频停止 低频运行 (LIRC=64K) STOP =1 无唤醒源	-	2	-	μA
		5V		-	6	-	
ISP3	静态电流 3	3V	高频停止 低频停止 STOP =1 无唤醒源	-	0.8	-	μA
		5V		-	0.9	-	
ISP4	静态电流 4	3V	高频停止 外部低频运行(LXT=32768HZ) STOP=1 无唤醒源	-	3.5	-	μA
		5V		-	15	-	

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°C)				
V <sub>IL1</sub>	输入低电平	3V	SMT	0	-	0.3VDD	V
V <sub>IH1</sub>	输入高电平	3V		0.7VDD	-	VDD	
V <sub>IL2</sub>	输入低电平	5V		0	-	0.3VDD	
V <sub>IH2</sub>	输入高电平	5V		0.7VDD	-	VDD	
V <sub>IL3</sub>	输入低电平	3V	1/2VDD	0	-	0.5VDD	
V <sub>IH3</sub>	输入高电平	3V		0.5VDD	-	VDD	
V <sub>IL4</sub>	输入低电平	5V		0	-	0.5VDD	
V <sub>IH4</sub>	输入高电平	5V		0.5VDD	-	VDD	
V <sub>IL5</sub>	输入低电平	3V	1.5V	0	-	1.3	
V <sub>IH5</sub>	输入高电平	3V		1.6	-	VDD	
V <sub>IL6</sub>	输入低电平	5V		0	-	2.3	
V <sub>IH6</sub>	输入高电平	5V		2.7	-	VDD	
I <sub>PH</sub>	上拉电阻	5V	V <sub>IN</sub> = GND	-	15	-	kΩ
		3V	V <sub>IN</sub> = GND	-	28	-	
I <sub>PL</sub>	下拉电阻	5V	V <sub>IN</sub> = VDD	-	40	-	
		3V	V <sub>IN</sub> = VDD	-	70	-	
I <sub>OL1</sub>	输出灌电流	5V	输出口, V <sub>OUT</sub> = VSS + 0.6V	-	25	-	mA
		3V		-	19	-	
I <sub>OH1</sub>	输出拉电流	5V	输出口, V <sub>OUT</sub> = VDD - 0.6V	-	21	-	
		3V		-	13	-	
I <sub>OL2</sub>	输出灌电流	5V	输出口, V <sub>OUT</sub> = VSS + 0.6V	-	15	-	
		3V		-	12.5	-	
I <sub>OH2</sub>	输出拉电流	5V	输出口, V <sub>OUT</sub> = VDD - 0.6V	-	19	-	
		3V		-	11	-	
I <sub>OL3</sub>	输出灌电流	5V	输出口, V <sub>OUT</sub> = VSS + 0.6V	-	11	-	
		3V		-	7.5	-	
I <sub>OH3</sub>	输出拉电流	5V	输出口, V <sub>OUT</sub> = VDD - 0.6V	-	9	-	
		3V		-	6	-	
I <sub>OL4</sub>	输出灌电流	5V	输出口, V <sub>OUT</sub> = VSS + 0.6V	-	4	-	
		3V		-	2.5	-	
I <sub>OH4</sub>	输出拉电流	5V	输出口, V <sub>OUT</sub> = VDD - 0.6V	-	5	-	
		3V		-	3	-	

注：具体值不做设计保证。



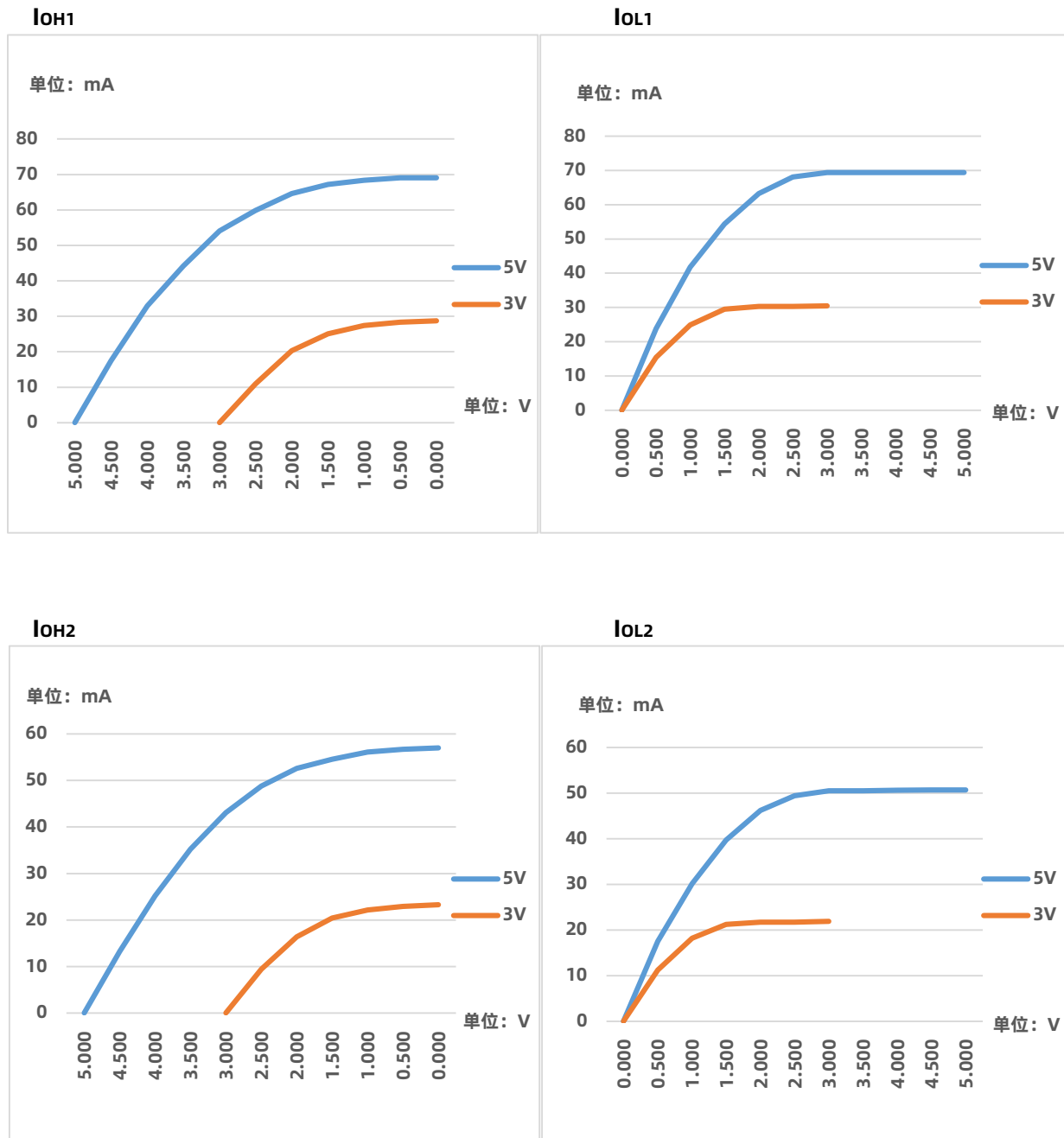
## 19.3 交流特性

常温(25°C)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			
反馈电阻:电阻 1(大电阻) (此配置为默认配置字)	30	18	5V	350	128	10.8
			3V	580	-2	3
	40	18	5V	380	129	10.6
			3V	620	9	3
	50	18	5V	550	78	10.6
			3V	830	-11	3
	60	24	5V	560	79	11.8
			3V	1300	4	3.7
	70	24	5V	500	76	11.9
			3V	990	-2	3.6

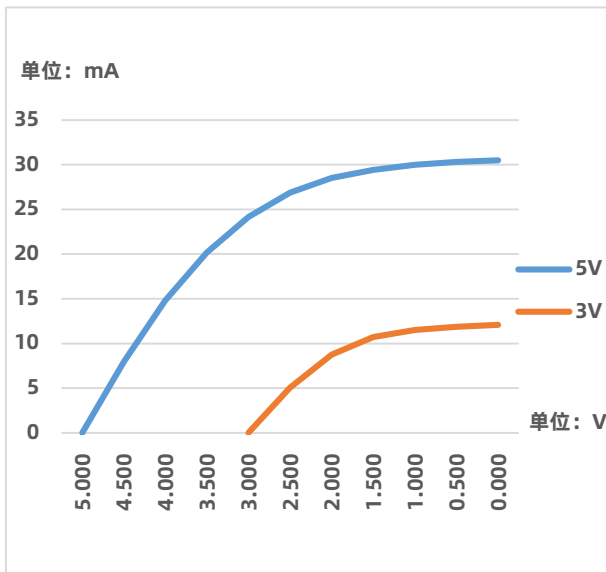
注：数据仅供参考，具体值不做设计保证。

## 19.4 IO 口拉灌电流特性

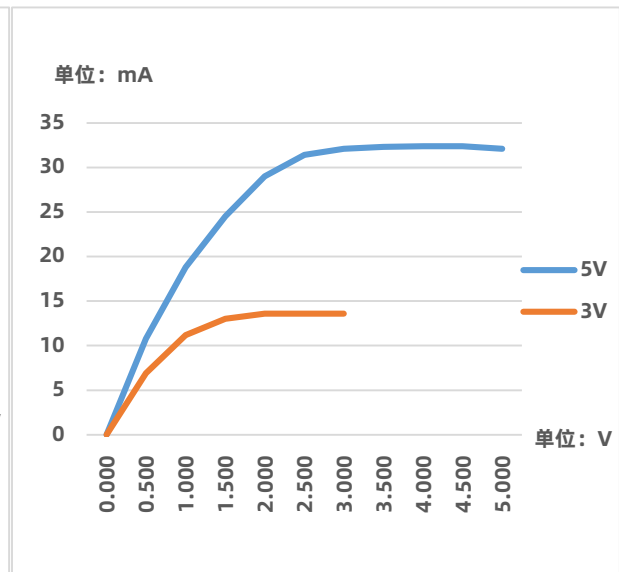


注：具体值不做设计保证。

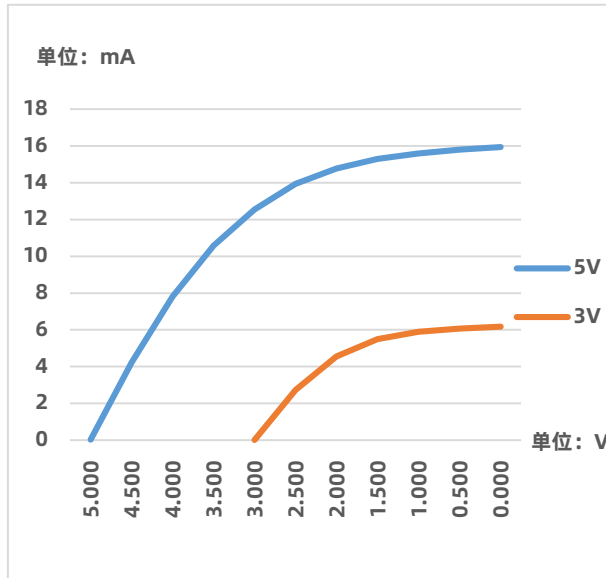
**IOH3**



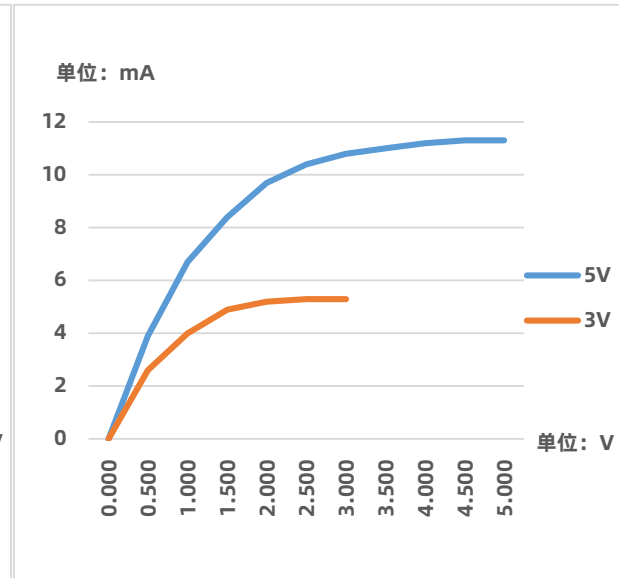
**IOI3**



**IOH4**



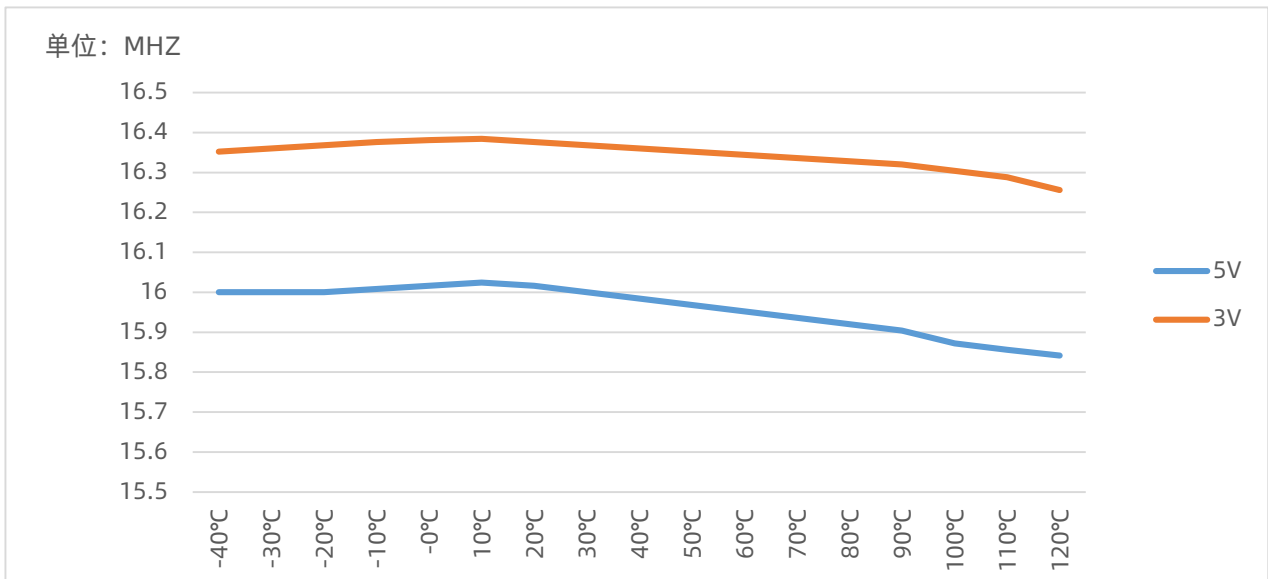
**IOI4**



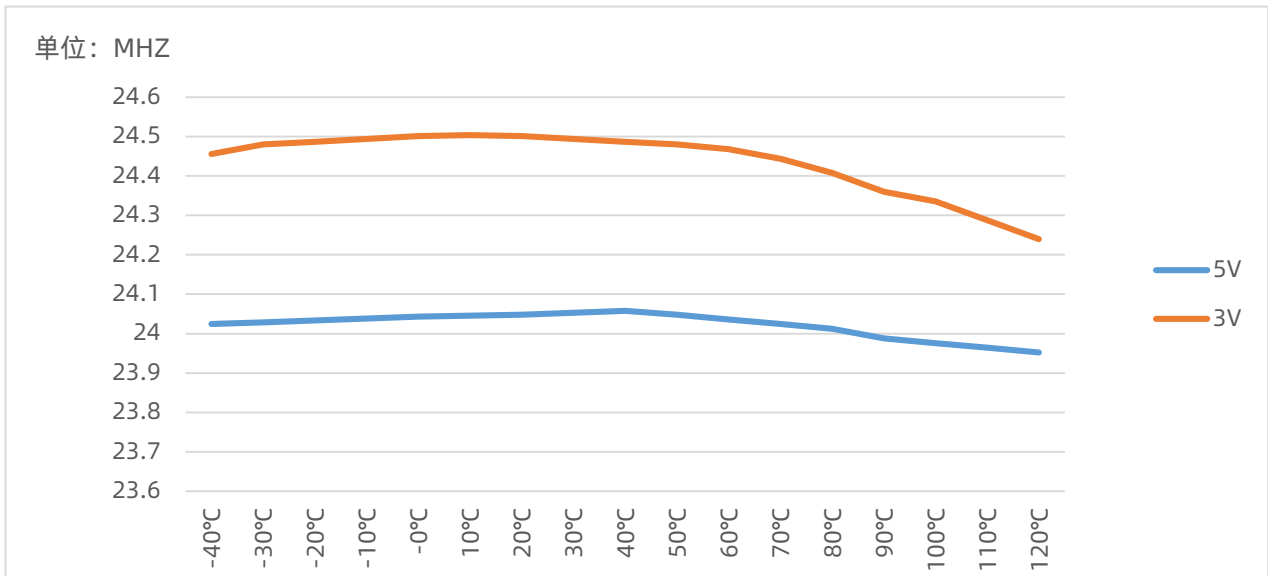
注: 具体值不做设计保证。

## 19.5 系统时钟特性

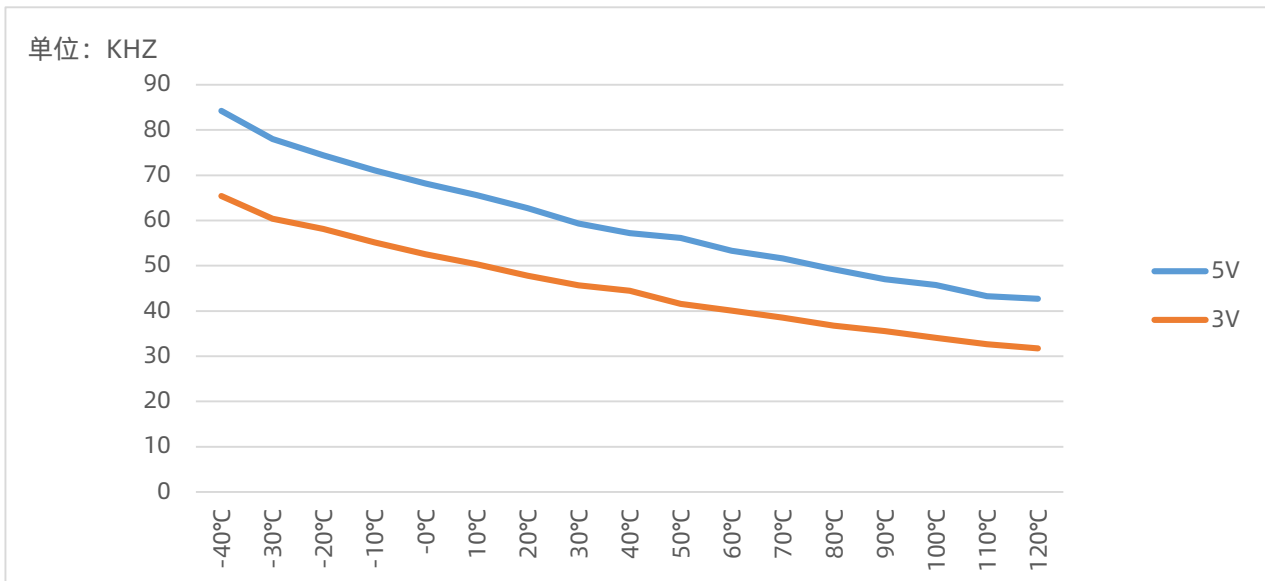
高频时钟（16MHZ）随温度变化曲线



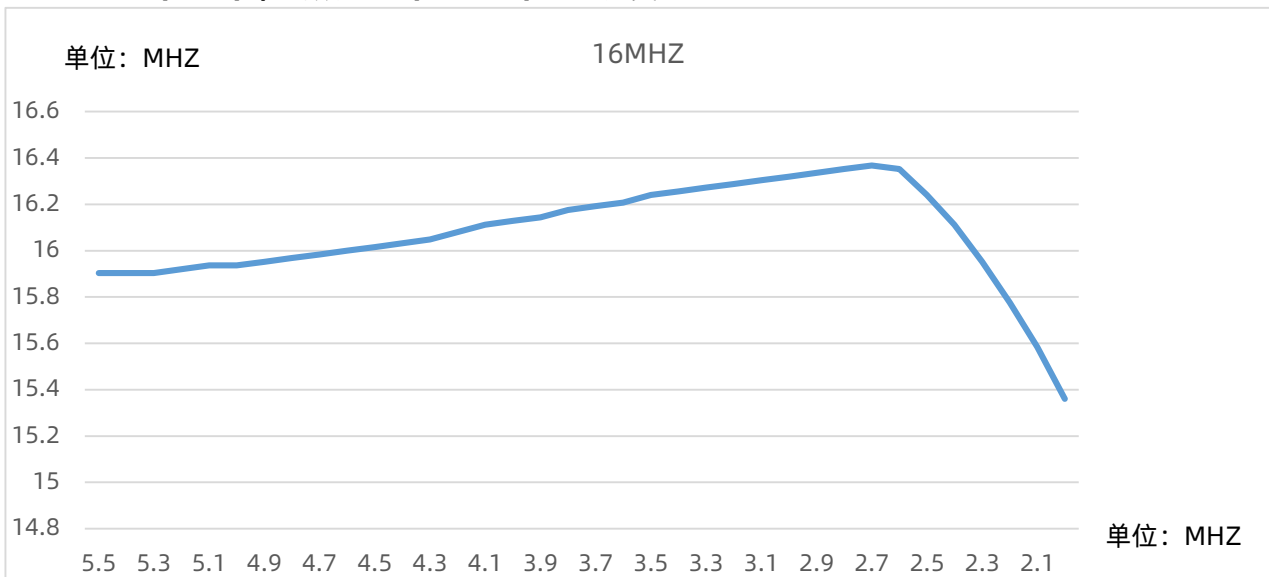
高频时钟（24MHZ）随温度变化曲线



低频时钟 (64KHZ) 随温度变化曲线

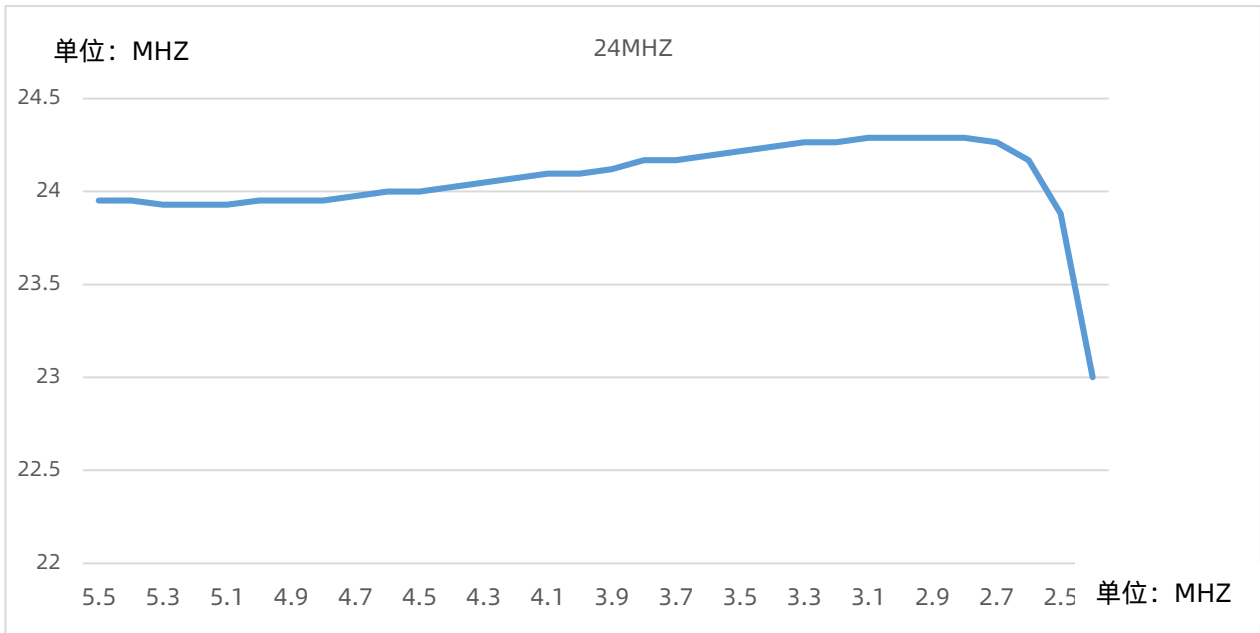


常温下 (20°C) , 高频时钟 (16MHZ) 随电压变化曲线

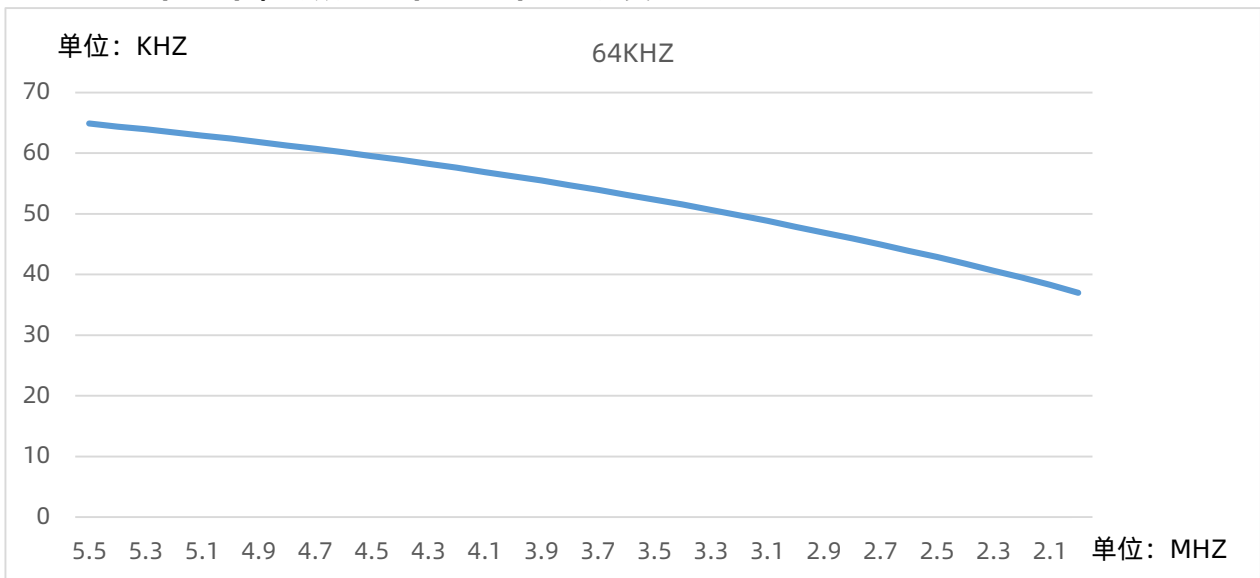


注: 具体值不做设计保证。

常温下 (20°C) , 高频时钟 (24MHZ) 随电压变化曲线



常温下 (20°C) , 低频时钟 (64KHZ) 随电压变化曲线



## 19.6 Flash 特性

参数	符号	温度 (°C)	最小值	典型值	最大值	单位
编程电压	$V_F^{(1)}$	$T_A = 25$	-	2.4	-	V
编程擦写次数	$N_{ENDUR}$	$T_J = -40 \sim 85$	100,000	-	-	Cycle
数据保存时间	$T_{RET}$	$T_A = 25$	10	-	-	Year
页擦除时间	$T_{PERASE}$	$T_A = 25$	-	3	-	ms
页编程时间	$T_{PROG}$	$T_A = 25$	-	2	-	ms

注：除此之外，必需保证在最低工作电压之上。

## 19.7 ADC 电气特性

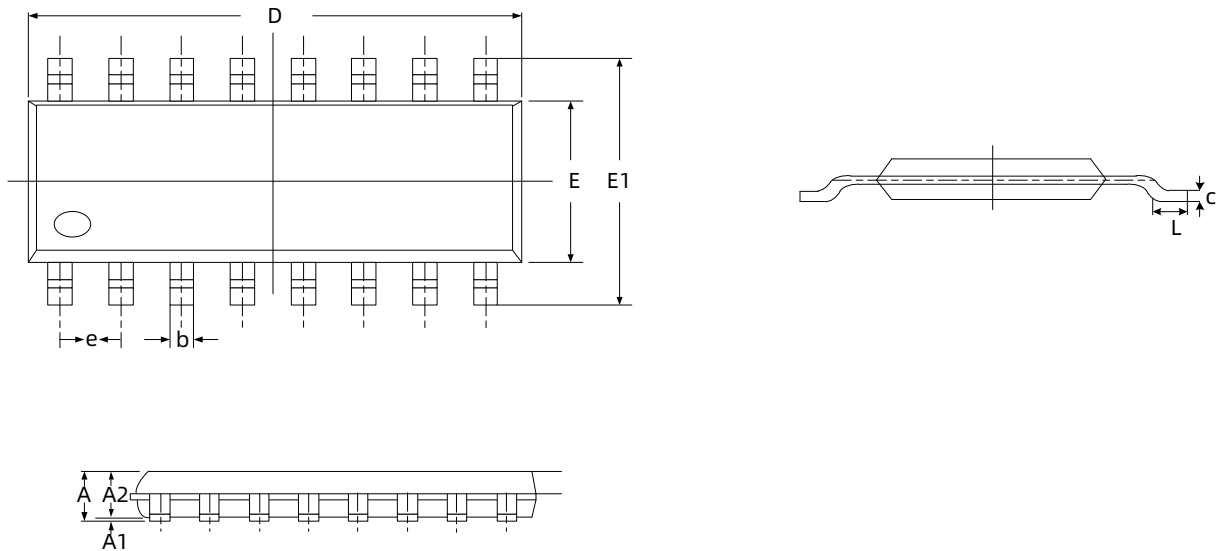
符号	参数	测试条件	最小值	典型值	最大值	单位
		条件 (常温 25°C)				
V <sub>ADC</sub>	ADC 工作电压	-	2.5	-	5.5	V
I <sub>ADC</sub>	ADC 工作电流	VDD=5V	-	200	-	μA
V <sub>AIN</sub>	ADC 输入电压	-	GND	-	V <sub>REF</sub>	V
V <sub>IREF1</sub>	内部参考电压 1	内部 2V 参考电压, VDD ≥ 2V + 0.5V	-2%	2	+2%	V
V <sub>IREF2</sub>	内部参考电压 2	内部 3V 参考电压, VDD ≥ 2V + 0.5V	-2%	3	+2%	V
V <sub>IREF3</sub>	内部参考电压 3	内部 4V 参考电压, VDD ≥ 2V + 0.5V	-2%	4	+2%	V
T <sub>VREF</sub>	参考稳定时间	VDD=5V, 参考电压选择和切换后	-	50	-	μs
F <sub>ADC</sub>	ADC 时钟	V <sub>REF</sub> = VDD =5V	-	-	2	MHZ
T <sub>CON</sub>	ADC 转换时间	-	-	16	-	T <sub>ADC</sub>
DNL	微分非线性误差	V <sub>REF</sub> = VDD =5V, F <sub>ADC</sub> =1MHZ	-2	-	+2	LSB
INL	积分非线性误差	V <sub>REF</sub> = VDD =5V, F <sub>ADC</sub> =1MHZ	-3	-	+3	LSB
E <sub>Z</sub>	偏移误差	V <sub>REF</sub> = VDD =5V, F <sub>ADC</sub> =1MHZ	-5	-	+5	LSB
E <sub>F</sub>	满刻度误差	V <sub>REF</sub> = VDD =5V, F <sub>ADC</sub> =1MHZ	-10	-	+10	LSB

注：具体值不做设计保证。



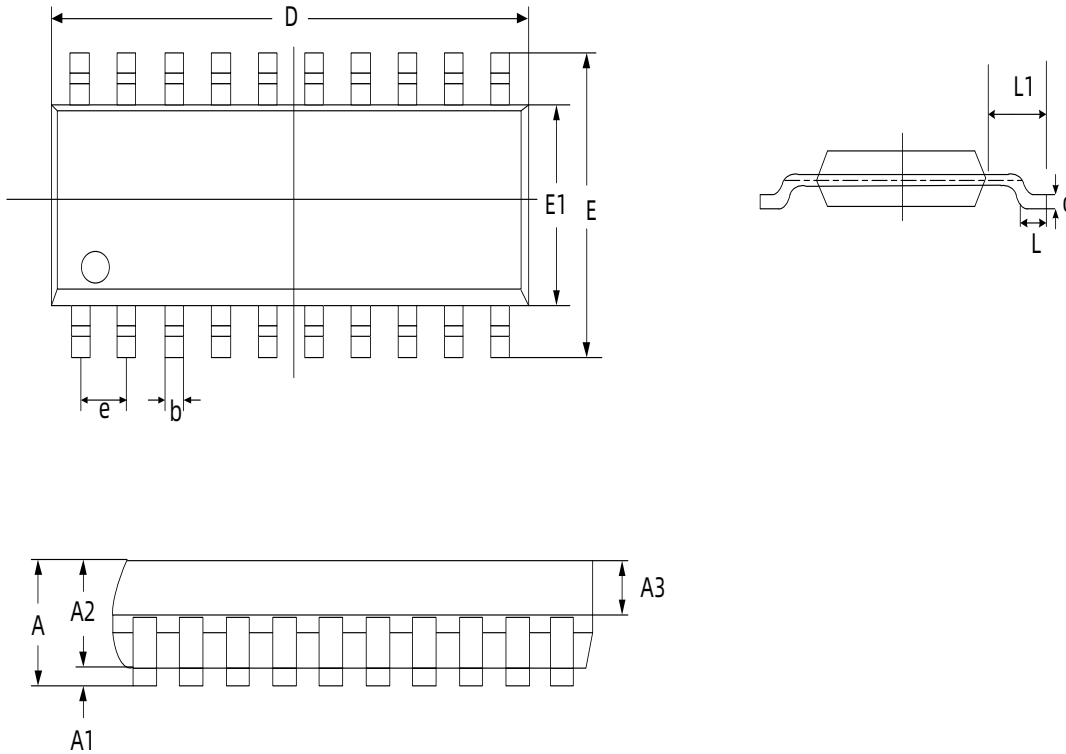
# 20 封装信息

## 20.1 SOP16



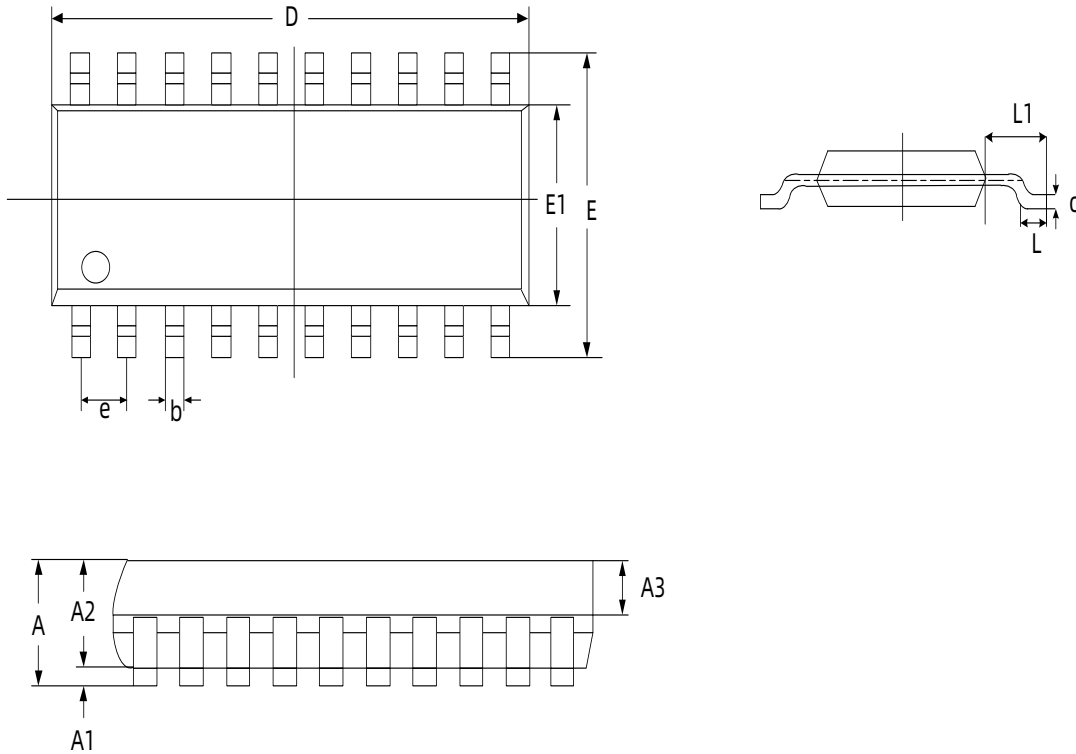
符号	单位 (mm)		
	最小	正常	最大
A	1.520	1.600	1.680
A1	0.100	0.150	0.200
A2	1.420	1.450	1.480
b	---	0.406	---
D	9.750	9.800	9.850
e	1.270BSC		
E	3.750	3.800	3.850
E1	6.050	6.150	6.250
L	0.350	0.500	0.650
c	---	0.203	---

## 20.2 TSSOP20



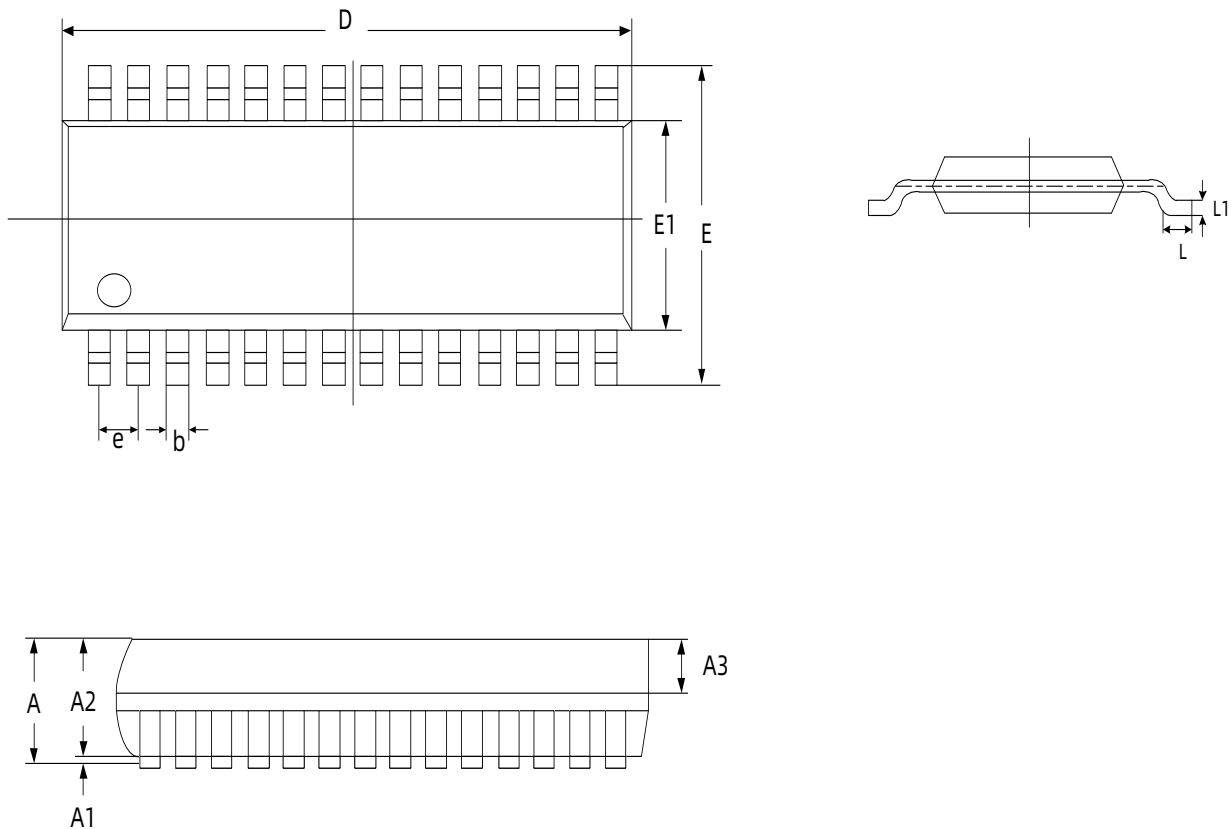
符号	单位 (mm)		
	最小	正常	最大
A	1.000	-	1.100
A1	0.050	-	0.150
A2	0.900	0.950	1.000
A3	0.390	-	0.400
b	0.200	0.220	0.240
D	6.400	6.450	6.500
e	0.550	0.650	0.750
E	6.250	6.400	6.550
E1	---	4.350	4.400
L	0.500	0.600	0.700
L1	1.000REF		
c	0.100	-	0.190

## 20.3 SOP20



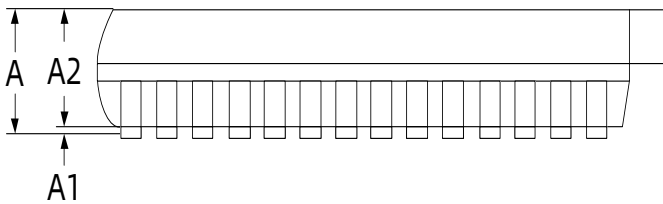
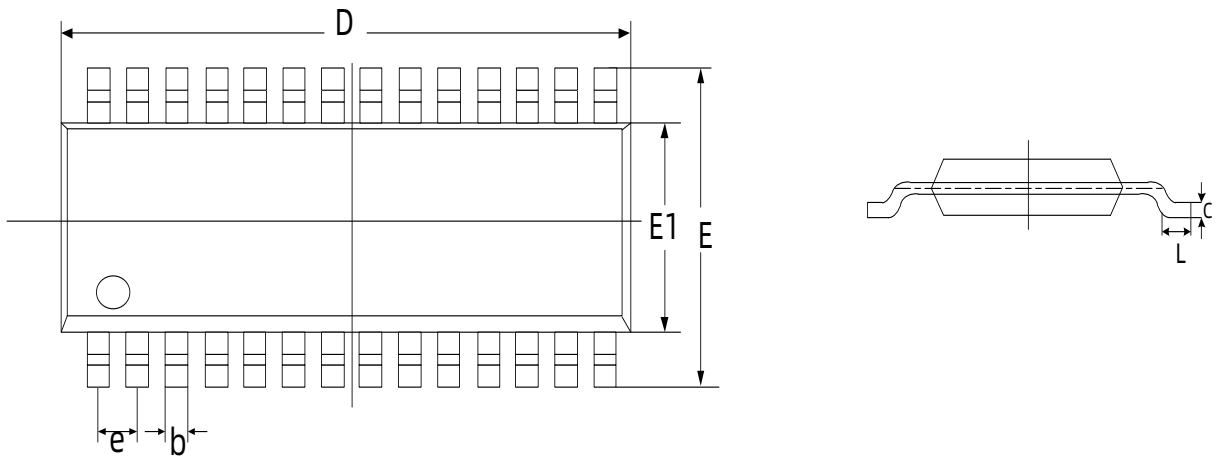
符号	单位 (mm)		
	最小	正常	最大
A	-	-	2.650
A1	0.100	0.150	0.200
A2	2.250	2.300	2.350
A3	0.970	1.020	1.070
b	0.350	-	0.430
D	12.700	12.800	12.900
e	1.270BSC		
E	10.250	10.350	10.450
E1	7.400	7.500	7.600
L	0.550	0.700	0.850
L1	1.400REF		
c	0.250	---	0.290

## 20.4 SOP28



符号	单位 (mm)		
	最小	正常	最大
A	2.485	2.490	2.495
A1	---	0.150	----
A2	2.250	2.340	2.450
A3	---	1.120	---
b	---	0.460	---
D	17.900	18.00	18.100
e	1.270BSC		
E	10.250	10.350	10.450
E1	7.510	7.520	7.530
L	0.600	0.700	0.800
L1	0.100REF		

## 20.5 SSOP28



符号	单位 (mm)		
	最小	正常	最大
A	1.450	1.500	1.550
A1	0.020	0.050	0.080
A2	1.400	1.450	1.500
b	---	0.254	---
D	9.300	9.800	10.300
e	0.635BSC		
E	6.250	6.350	6.450
E1	3.750	3.800	3.850
L	0.350	0.500	0.650
c	---	0.203	---

# 21 指令集简述

## 21.1 概述

M9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：字节操作类指令、位操作类指令、控制操作类指令、立即数操作类指令和存储器操作类指令。

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。对于上述两种特征情况，指令执行需要两个指令周期。

任何一条指定文件寄存器，作为指令的一部分都会进行读-修改-写操作。读寄存器、修改数据并根据指令或目标寄存器标识符“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。

## 21.2 符号说明

符号	范围	说明	符号	范围	说明
R/r	000h-fffh	8位文件寄存器地址	PC	-	PC指针
Rs	000h-fffh	12位文件寄存器源地址	C	-	进位标志
Rd	000h-fffh	12位文件寄存器目标地址	DC	-	半进位标志
A	-	AREG寄存器	DC	-	半进位标志
a		快速操作RAM位	Z	-	零标志
BSR		存储选择寄存器	OV	-	溢出标志
b	0-7	位地址	N	-	负标志
K/k		立即数、常数或标号	d	0-1	目的操作数定义
TOS	-	栈顶	TO	-	超时位
Label	-	标号名称	PD	-	掉电位
s	-	快速调用选择位	GIE	-	总中断使能位

## 21.3 M9 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
字节操作类指令	ADDAR R, d, a	A 与 R 相加	1	C, DC, Z, OV, N	1, 2
	ADCAR R, d, a	A 带进位与 R 相加	1	C, DC, Z, OV, N	1, 2
	ANDAR R, d, a	A 和 R 作逻辑与运算	1	Z, N	1, 2
	CLRR R, a	将 R 清零	1	Z	2
	COMR R, d, a	对 R 取反	1	Z, N	1, 2
	JERA R, a	R 与 A 比较, 相等跳过	1 (2 或 3)	-	3
	JGRA R, a	R 与 A 比较, 大于跳过	1 (2 或 3)	-	3
	JLRA R, a	R 与 A 比较, 小于跳过	1 (2 或 3)	-	1
	DECR R, d, a	R 递减 1	1	C, DC, Z, OV, N	1, 2, 3
	DJZR R, d, a	R 递减 1, 为 0 跳过	1 (2 或 3)	-	1, 2, 3
	DJNZR R, d, a	R 递减 1, 非 0 跳过	1 (2 或 3)	-	1
	INCR R, d, a	R 递增 1	1	C, DC, Z, OV, N	1, 2, 3
	JZR R, d, a	R 递增 1, 为 0 跳过	1 (2 或 3)	-	3
	JNZR R, d, a	R 递增 1, 非 0 跳过	1 (2 或 3)	-	1
	ORAR R, d, a	A 与 R 作逻辑或运算	1	Z, N	1
	MOVR R, d, a	传送 R	1	Z, N	1
	MOVRR Rs, Rd	从源 Rs 送到目标 Rd	2	-	
	MOVAR R, a	将 A 中的内容送入 R	1	-	
	MULAR R, a	A 与 R 相乘	1	-	1
	NEGR R, a	对 R 取补	1	C, DC, Z, OV, N	
	RLR R, d, a	R 带进位循环左移	1	C, Z, N	1
	RLNCR R, d, a	R 循环左移 (无进位)	1	Z, N	
	RRR R, d, a	R 带进位循环右移	1	C, Z, N	
	RRNCR R, d, a	R 循环右移 (无进位)	1	Z, N	
	SETR R, a	将 R 全置为 1	1	-	1
	SBCAR R, d, a	A 减去 R (带借位)	1	C, DC, Z, OV, N	
	SUBRA R, d, a	R 减去 A	1	C, DC, Z, OV, N	1
	SBCRA R, d, a	R 减去 A (带借位)	1	C, DC, Z, OV, N	
	SWAPR R, d, a	对 R 进行半字节交换	1	-	3
	JREZ R, a	测试 R, 为 0 则跳过	1 (2 或 3)	-	1
	XORAR R, d, a	A 和 R 作逻辑异或运算	1	Z, N	
	位操作指令	BCLR R, b, a	将 R 中的指定位清零	1	-
BSET R, b, a		将 R 中的指定位置 1	1	-	1
JBTS0 R, b, a		测试 R 的位, 为 0 跳过	1 (2 或 3)	-	2, 3
JBTS1 R, b, a		测试 R 的位, 为 1 跳过	1 (2 或 3)	-	2, 3
BNEG R, b, a		将 R 中的指定位取反	1	-	1

指令类型	助记符	指令说明	周期数	影响标志位	备注
控制操作类指令	RJBC Label	如果有进位则跳转	1 (2)	-	
	RJBN Label	如果为负则跳转	1 (2)	-	
	RJBNC Label	如果没有进位则跳转	1 (2)	-	
	RJBNN Label	如果不为负则跳转	1 (2)	-	
	RJBNOV Label	如果未溢出则跳转	1 (2)	-	
	RJBNZ Label	如果不为零则跳转	1 (2)	-	
	RJBOV Label	如果溢出则跳转	1 (2)	-	
	RGOTO Label	无条件跳转	2	-	
	RJBZ Label	如果为零则跳转	1 (2)	-	
	CALL k, s	调用子程序	2	-	
	DAA	对 A 进行十进制调整	1	C	
	GOTO k	跳转到地址	2	-	
	NOP	无操作	1	-	
	NOP	无操作	1	-	3
	SPOP	弹出堆栈栈顶 TOS	1	-	
	SPUSH	压入堆栈栈顶 TOS	1	-	
	RCALL Label	相对调用	2	-	
	SRESET	软件器件复位	1	所有	
	RETIE s	中断返回并允许中断	2	GIE/GIEH	
	RETIA k	返回并将 k 送入 A	2	-	
RETURN s	从子程序返回	2	-		
立即数操作指令	ADDIA k	A 与立即数 k 相加	1	C, DC, Z, OV, N	
	ANDIA k	立即数 k 和 A 作逻辑与运算	1	Z, N	
	ORIA k	立即数 k 和 A 作逻辑或运算	1	Z, N	
	LDFSR R,k	传送立即数 k 到 FSR	2	-	
	BANKBSR k	将立即数 k 送入 BSR	1	-	
	MOVIA k	将立即数 k 送入 A	1	-	
	MULIA k	立即数 k 与 A 相乘	1	-	
	SUBIA k	立即数 k 减去 A	1	C, DC, Z, OV, N	
	XORIA k	立即数 k 与 A 作逻辑异或运算	1	Z, N	
存储器操作指令	RDT*	表读	2	-	
	RDT*+	表读, 后递增		-	
	RDT*-	表读, 后递减		-	
	RDT+*	表读, 预递增		-	
	WDT*	表写	2	-	



**注：**

- (1) 当端口寄存器修改自身时，修改时使用的值是引脚上的当前值。例如，如果将一引脚配置为输入，其对应数据锁存器中的值将为 1，但此时若有外部器件将该引脚驱动为低电平，则被写回数据锁存器的数据值将是 0。
- (2) 如果程序计数器 (PC) 被修改或者条件测试为真，则该指令需要两个周期。第二个周期执行一条 NOP 指令。
- (3) 某些指令是双字指令。除非指令的第一个字获取这 16 位中包含的信息，否则第二个字将作为 NOP 指令执行。这将确保所有程序存储单元内存储的都是合法的指令。

## 22 修正记录

版本	日期	描述
V1.00	2021-04-23	初版
---	---	---
V1.09	2024-01-04	添加 SOP16 封装
V1.10	2024-01-22	勘误