产品特性

- 高性能、低功耗的 8 位 AVR® 微处理器
- RISC 结构
  - 118 条指令 – 大多数指令执行时间为单个时钟周期
  - 32 个 8 位通用工作寄存器
  - 全静态工作
  - 工作于 16 MHz 时性能高达 16 MIPS
- 非易失性程序和数据存储器
  - 2K 字节的系统内可编程 Flash
    擦写寿命 : 10,000 次
  - 128 字节的系统内可编程 EEPROM
    擦写寿命 : 10,000 次
  - 128 字节的片内 SRAM
  - 可以对锁定位进行编程以及实现 EEPROM 数据的加密
- 外设特点
  - 具有独立预分频器的 8 位定时器 / 计数器
  - 具有独立预分频器的高速 8 位定时器
    2 个具有独立输出比较寄存器的高频率 PWM
    非重叠的反相 PWM 输出引脚
  - 具有开始状态检测器的通用串行接口
  - 10 位 ADC
    11 个单端通道
    8 个差分 ADC 通道
    7 对具有可编程增益（1x, 20x）的差分通道
  - 片内模拟比较器
  - 外部中断
  - 11 个引脚电平变化可以触发中断
  - 具有独立片内振荡器的可编程看门狗定时器
- 特殊的处理器特点
  - 低功耗空闲模式、噪声抑制模式、省电模式
  - 上电复位以及可编程的掉电检测
  - 片内 / 片外中断源
  - 通过 SPI 端口在系统内可编程
  - 经过标定的片内 RC 振荡器
- I/O 和封装
  - 20 引脚 PDIP/SOIC: 16 个可编程 I/O 线
  - 32 引脚 MLF: 16 个可编程 I/O 线
- 工作电压
  - ATtiny26L：2.7V - 5.5V
  - ATtiny26：4.5V - 5.5V
- 速度等级
  - ATtiny26L：0 - 8 MHz
  - ATtiny26：0 - 16 MHz
- ATtiny26L 的功耗
  - 16 MHz, 5V, 25°C:15 mA
  - 1 MHz, 3V, 25°C: 0.70 mA
  - 1 MHz, 3V, 25°C, 空闲模式 : 0.18 mA
  - 掉电模式 : < 1 µA

具有 2KB 系统内可编程 Flash 的 8 位 AVR® 微控制器

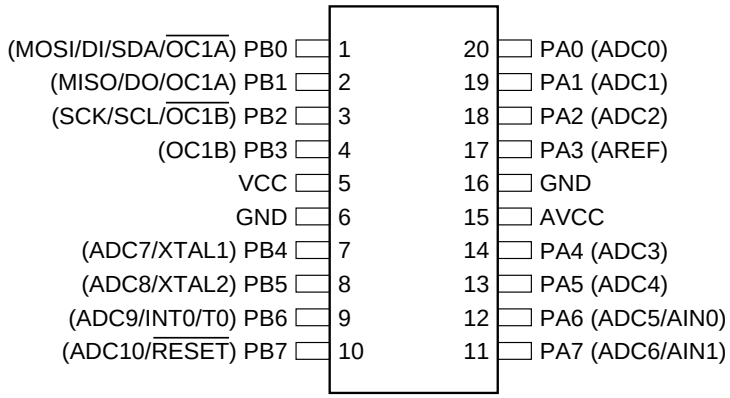
ATtiny26
ATtiny26L

本文是英文数据手册的中文翻译，其目的是方便中国用户的阅读。它无法自动跟随原稿的更新，同时也可能存在翻译上的错误。读者应该以英文原稿为参考以获得更准确的信息。

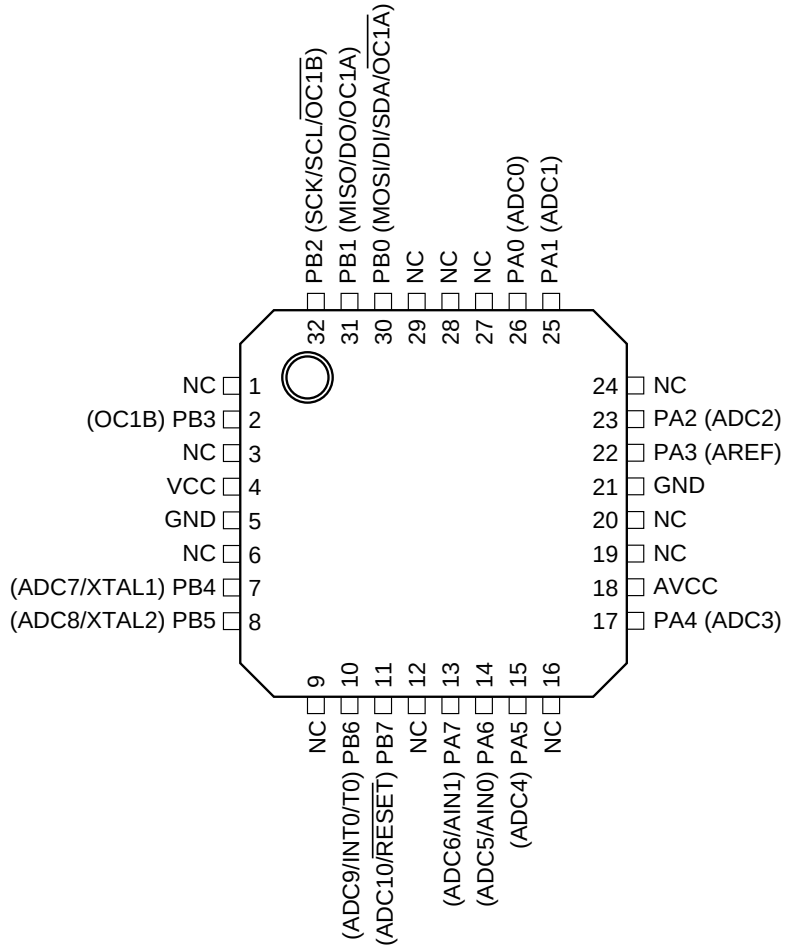
Rev. 1477E–AVR–12/03

引脚配置

PDIP/SOIC



MLF Top View



综述

ATtiny26(L)是基于增强的AVR RISC结构的低功耗8位CMOS微控制器。由于其先进的指令集以及单时钟周期指令执行时间， ATtiny26(L) 的数据吞吐率高达 1 MIPS/MHz，从而可以缓减系统在功耗和处理速度之间的矛盾。

AVR 内核具有丰富的指令集和 32 个通用工作寄存器。所有的寄存器都直接与算逻单元(ALU) 相连接，使得一条指令可以在一个时钟周期内同时访问两个独立的寄存器。这种结构大大提高了代码效率，并且具有比普通的 CISC 微控制器最高至 10 倍的数据吞吐率。ATtiny26(L) 有包含11个单端通道与8个差分通道的高精度ADC。七个具有可选增益为20x的差分通道。其中四个具有可选增益的差分通道可同时使用。 ATtiny26(L) 还有含两个独立输出的高频 8 位 PWM 模块。两个 PWM 输出有用来同步整流的反向非重叠输出引脚。ATtiny26(L)的通用串行接口允许通过TWI或SM总线接口的 软件的执行。根据其特点，它适用于高集成度的电池充电器、镇流器、低端调温器、火警探测器等。

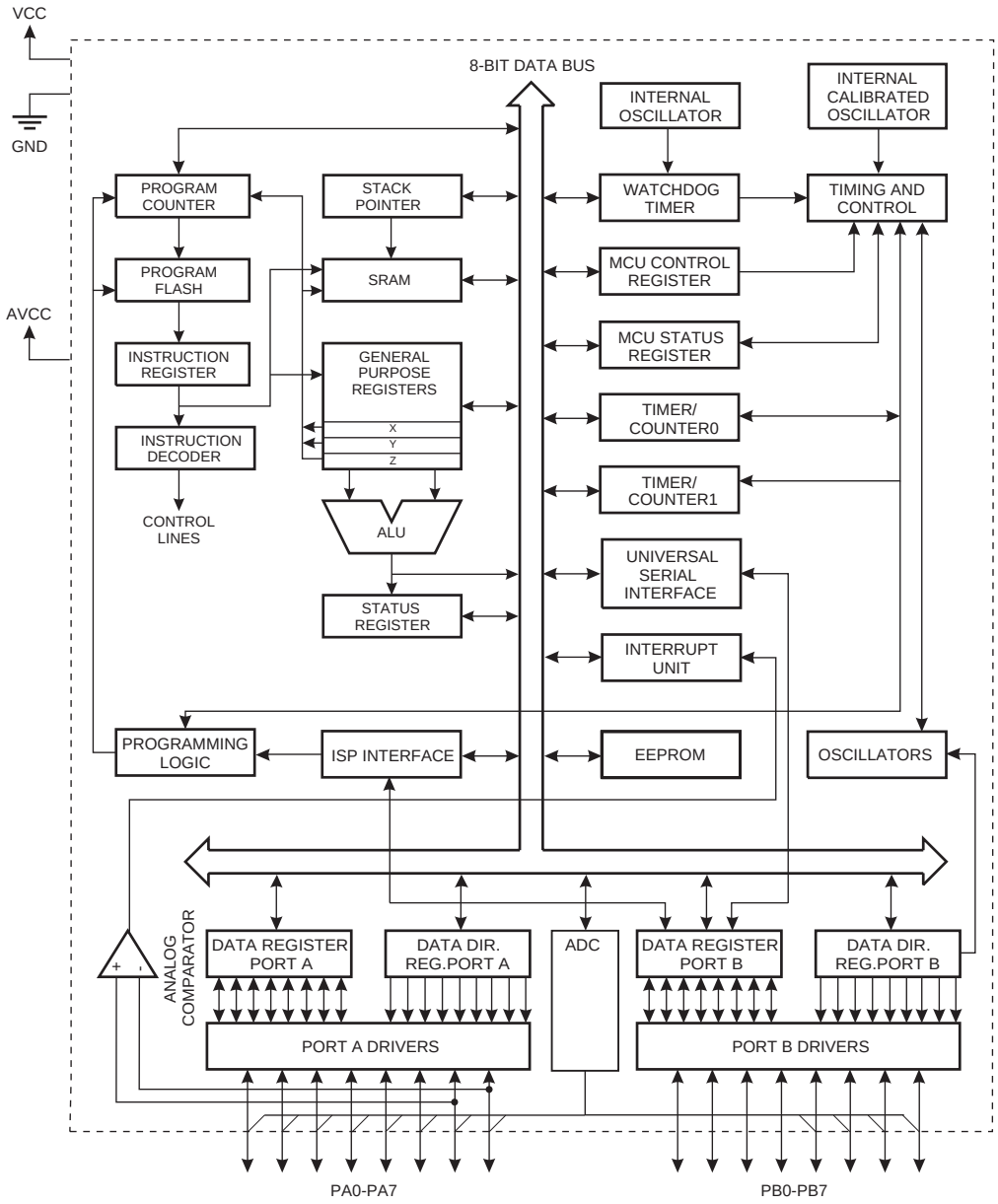
ATtiny26(L) 有2K字节Flash，128字节EEPROM，128字节SRAM，16个通用 I/O 口线，32个通用工作寄存器，两个8定时器/计数器，其中一个有PWM 输出，片内/外振荡器，片内 / 外中断，可编程看门狗定时器，含两个差分输入电压增益级的 11 通道 10 位模数转换器，以及四 个可以通过软件进行选择的省电模式。工作于空闲模式时 CPU 停止工作，而T/C以及中断系统继续工作； ATtiny26(L)还有专门的ADC噪声抑制模式降低ADC转换噪声，ADC 噪声抑制模式时终止 ADC 以外所有 I/O 模块的工作；掉电模式时晶体振荡器停止振荡，所有功能除了中断和硬件复位之外都停止工作； Standby模式下只有外部晶体振荡器运行，其余与掉电模式相同。 ATtiny26(L) 中引脚触发唤醒与中断使能的特性，使得器件只消耗极少的电流，同时具有快速启动能力。

本芯片是以 Atmel 高密度非易失性存储器技术生产的。通过将 8 位 RISC CPU 与系统内可编程的 Flash 集成在一个芯片内， ATtiny26(L) 成为一个功能强大的单片机，为许多嵌入式控制应用提供了灵活而低成本的解决方案。

ATtiny26(L)具有一整套的编程与系统开发工具，包括:宏汇编、程序调试器/软件仿真器、仿真器及评估板。

方框图

Figure 1. ATtiny26(L) 结构框图



引脚说明

VCC 数字电路的电源

GND 地

AVCC AVCC 是端口 A 与 A/D 转换器的电源。不使用 ADC 时该引脚应直接与 V_{CC} 连接。使用 ADC 时应通过一个低通滤波器与 V_{CC} 连接，详见 P75。

端口 A(PA7..PA0) 端口 A 为 8 位通用 I/O 口，每个管脚都具有独立可控制的内部上拉电阻。端口 A 还做为 ADC 与模拟比较器的模拟输入端及引脚触发中断端口，详见 P92“ 端口的第二功能 ”。

端口 B(PB7..PB0) 端口 B 为 8 位通用 I/O 口，PB6..0 具有独立可控制的内部上拉电阻。没有作为复位引脚使用时，PB7 可以用作 I/O 口。编程 RSTDISBL 熔丝位 (“0”) 后，PB7 作为 I/O 口使用，取代其作为 RESET 引脚的功能。端口 B 在 ADC、时钟、 T/C、 USI、 SPI 编程与引脚触发中断中的功能描述，详见 P92“ 端口的第二功能 ”。

外部复位通过在 PB7/$\overline{\text{RESET}}$ 引脚上施加低电平实现。即使时钟不运行，只要复位脉冲大于 50 ns 时就会产生复位，更短的脉冲不保证产生复位。

XTAL1 反向振荡放大器与片内时钟操作电路的输入端。

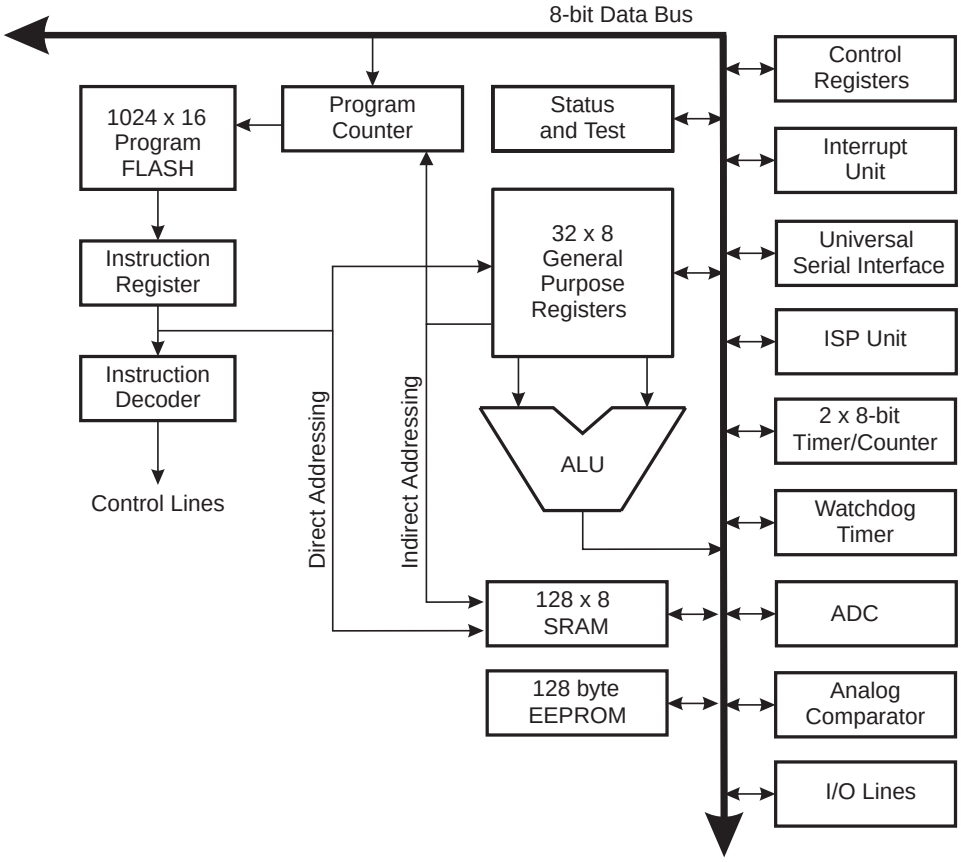
XTAL2 反向振荡放大器的输出端。

结构综述

快速访问寄存器文件包括 32 个 8 位通用工作寄存器，访问时间为一个时钟周期。从而实现了单时钟周期的 ALU 操作。在典型的 ALU 操作中，两个位于寄存器文件中的操作数同时被访问，然后执行运算，结果再被送回到寄存器文件。整个过程仅需一个时钟周期。

寄存器文件里有 6 个寄存器可以用作 3 个 16 位的间接寻址寄存器指针以寻址数据空间，实现高效的地址运算。其中一个指针还可以作为程序存储器查询表的地址指针。这些附加的功能寄存器即为 16 位的 X、 Y、 Z 寄存器。

Figure 2. ATtiny26(L) AVR 增强 RISC 结构



ALU支持寄存器之间以及寄存器和常数之间的算术和逻辑运算。ALU也可以执行单寄存器操作。Figure 2 给出 ATtiny26(L) AVR 增强 RISC 微控制器结构。除寄存器操作模式外，当寄存器文件位于数据空间的最低的 32 个地址 ($00 - $1F)，可使用常规内存寻址方式，就象对普通存储器地址一样访问它们。

I/O存储器空间包含64个可以直接寻址的地址，作为CPU外设的控制寄存器、T/C、ADC，以及其他 I/O 功能。映射到数据空间即为寄存器文件之后的地址 $20 - $5F。

为了获得最高的性能以及并行性， AVR 采用了 Harvard 结构，具有独立的数据和程序总线。程序存储器里的指令通过一级流水线运行。 CPU 在执行一条指令的同时读取下一条指令（在本文称为预取）。这个概念实现了指令的单时钟周期运行。程序存储器是可以在线编程的 Flash。

程序流程通过跳转指令和调用指令来控制，从而直接寻址整个地址空间。所有指令长度为 16 位，亦即每个程序存储器地址都包含一条 16 位的指令。

在中断和调用子程序时返回地址的程序计数器 (PC) 保存于堆栈之中。堆栈位于通用数据 SRAM，因此其深度仅受限于 SRAM 的大小。在复位例程里用户首先要初始化堆栈指针

SP。这个指针位于 I/O 空间，可以进行读写访问。对于用 C 语言编写的程序，在链接程序中必须给出堆栈深度，详见 C 程序用户指南。

128 字节数据 SRAM 可以通过 5 种不同的寻址模式进行访问。

AVR 存储器空间为线性的平面结构。

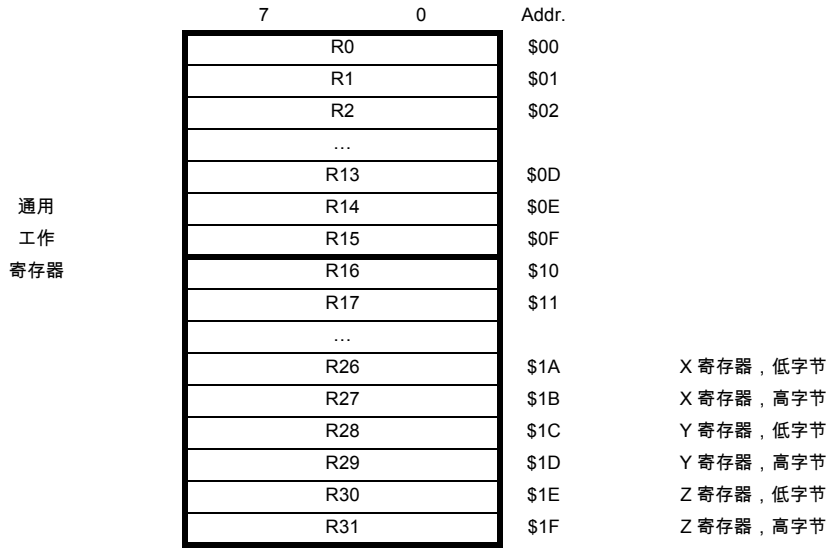
I/O 存储器空间包含 64 个可以直接寻址的地址，作为 CPU 外设的控制寄存器、T/C，以及其他 I/O 功能。 AVR 存储器空间为线性的平面结构。

AVR有一个灵活的中断模块。控制寄存器位于I/O空间。状态寄存器里有全局中断使能位。每个中断在中断向量表里都有独立的中断向量。各个中断的优先级与其在中断向量表的位置有关，中断向量地址越低，优先级越高。

通用寄存器文件

Figure3 为 CPU 32 个通用工作寄存器的结构。

Figure 3. AVR CPU 通用工作寄存器



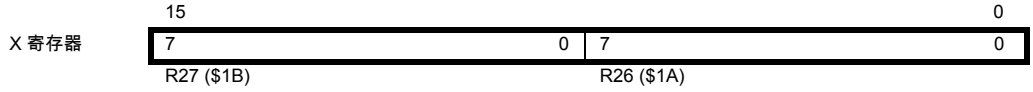
大多数操作寄存器文件的指令都可以直接访问所有的寄存器，而且多数这样的指令的执行时间为单个时钟周期。只有常数计算与寄存器指令 SBCI、SUBI、CPI、ANDI 与 ORI 及载入立即数指令 LDI 例外。这些指令用于寄存器文件的后半部分 – R16..R31。而通用的 SBC、 SUB、 CP、AND 和 OR 及其他两寄存器之间或单寄存器中的指令用于整个寄存器文件。

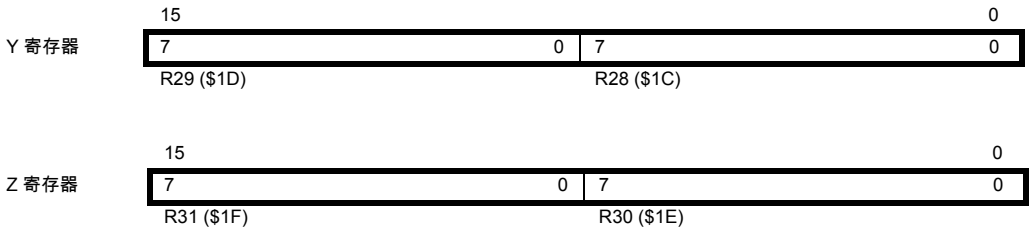
如 Figure3 所示，每个寄存器都有一个数据内存地址，将他们直接映射到用户数据空间的头 32 个地址。虽然寄存器文件的物理实现不是 SRAM，这种内存组织方式在访问寄存器方面具有极大的灵活性，因为 X、 Y、 Z 寄存器可以设置为指向任意寄存器的指针。

X、Y、Z 寄存器

寄存器 R26..R31 除了用作通用寄存器外，还可以作为数据间接寻址用的地址指针。这三个间接寻址寄存器定义如下：

Figure 4. X、 Y、 Z 寄存器





在不同的寻址模式中，这些地址寄存器可以实现固定偏移量，自动加一和自动减一功能。具体细节请参见指令集。

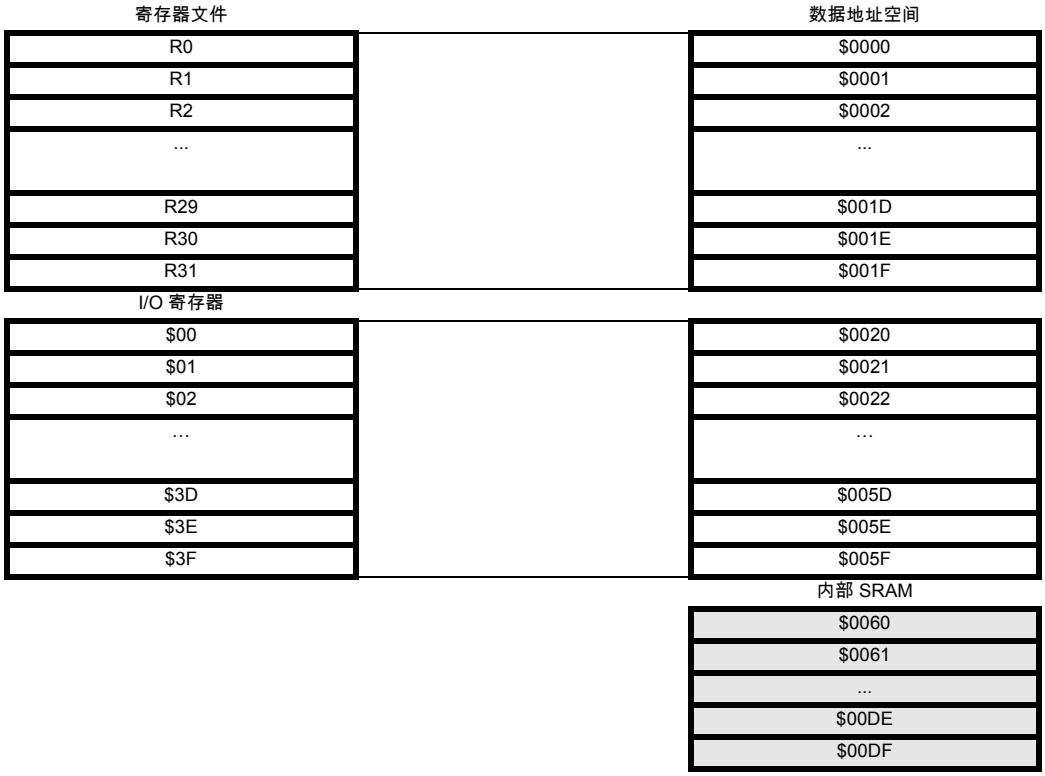
ALU - 算术逻辑单元

AVR ALU 与 32 个通用工作寄存器直接相连。寄存器与寄存器之间、寄存器与立即数之间的 ALU 运算只需要一个时钟周期。 ALU 操作分为 3 类：算术、逻辑和位操作。

系统内可编程的 Flash 程序存储器

ATtiny26(L)具有2K字节的在线编程Flash ,用于存放程序指令代码。因为所有的AVR指令为 16 位或 32 位 ,故而 Flash 组织成 1K x 16 位的形式。Flash 存储器至少可以擦写 10,000 次。ATtiny26(L) 的程序计数器 (PC) 为 10 位 ,因此可以寻址 1024 字的程序存储器空间。P103“存储器编程” 详述了Flash数据下载的操作。P9“程序与数据寻址模式” 给出了程序存储器的寻址模式。

Figure 5. SRAM 组织结构



SRAM 数据存储器

Figure5 给出了 ATtiny26(L) SRAM 空间的组织结构。

前 224 个数据存储器包括了寄存器文件、 I/O 存储器及内部数据 SRAM。起始的 96 个地址为寄存器文件与 I/O 存储器，接着是 128 字节的内部数据 SRAM。

数据存储器的寻址方式分为 5 种 : 直接寻址、带偏移量的间接寻址、间接寻址、带预减量的间接寻址和带后增量的间接寻址。寄存器文件中的寄存器 R26 到 R31 为间接寻址的指针寄存器。

直接寻址范围可达整个数据区。带偏移量的间接寻址模式能够寻址到由寄存器 Y 和 Z 给定的基址附近的 63 个地址。

在自动预减和后加的间接寻址模式中，寄存器 X、 Y 和 Z 自动增加或减少。

ATtiny26(L)的全部32个通用寄存器、64个I/O寄存器及128个字节的内部数据SRAM可以通过所有上述的寻址模式进行访问。

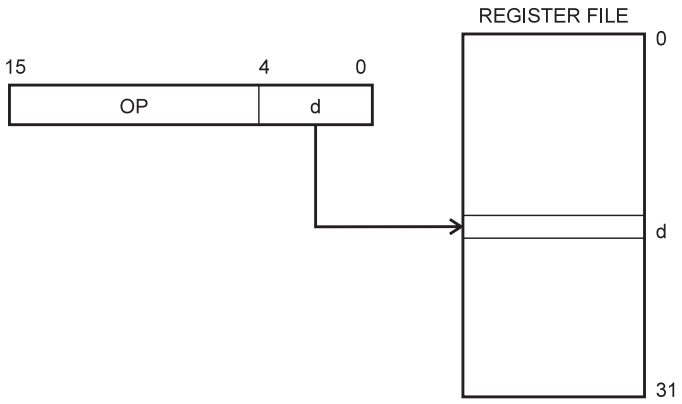
下节中将对不同的寻址模式进行详细说明。

程序与数据寻址模式

ATtiny26(L) AVR 增强型 RISC 微控制器提供强大、有效的 Flash 程序存储器、SRAM、寄存器文件及 I/O 数据存储器的寻址模式。本节给出 AVR 结构支持的不同寻址模式。在图中， OP 表示指令字中的操作代码部分。为简单起见，在图中没有给出确切的寻址位。

直接寄存器，单寄存器 Rd

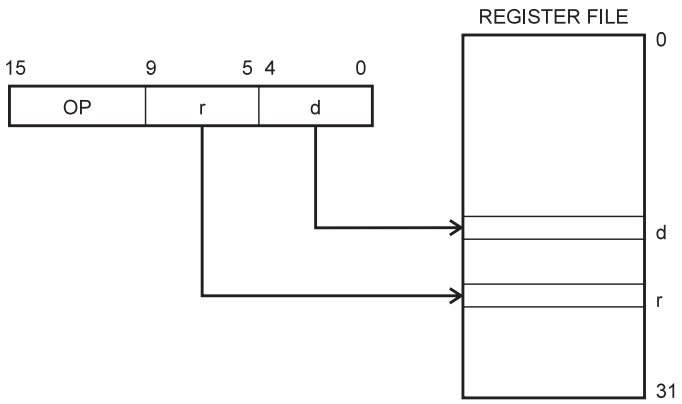
Figure 6. 直接单寄存器寻址



寄存器 d (Rd) 中含有操作数。

直接寄存器，双寄存器 Rd 与 Rr

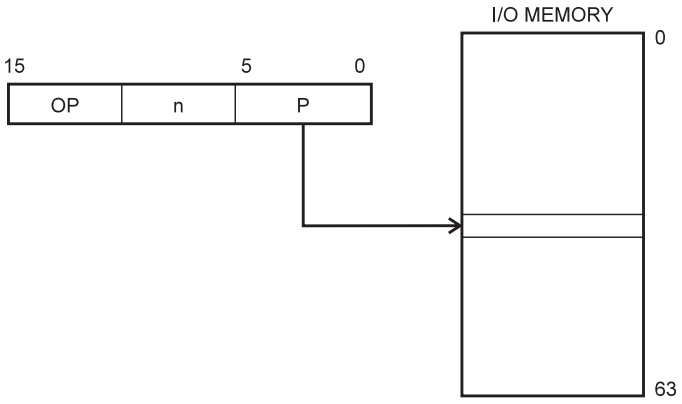
Figure 7. 直接双寄存器寻址



寄存器 r (Rr) 与 d (Rd) 中含有操作数。结果存于寄存器 d (Rd)。

I/O 直接

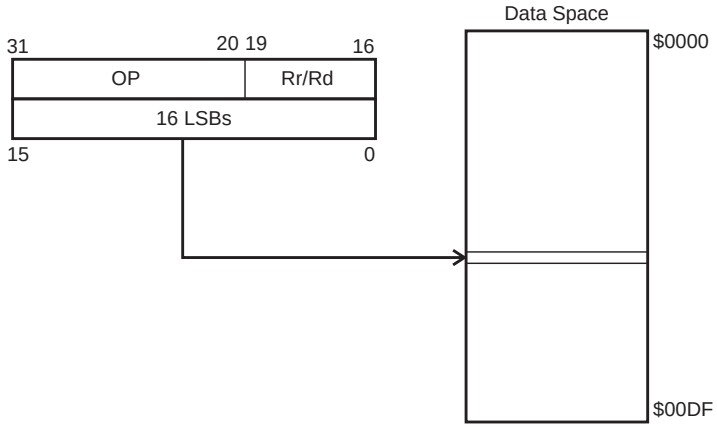
Figure 8. I/O 直接寻址



操作数地址含在指令字的 6 位中。 n 表示目的或源寄存器地址。

数据直接

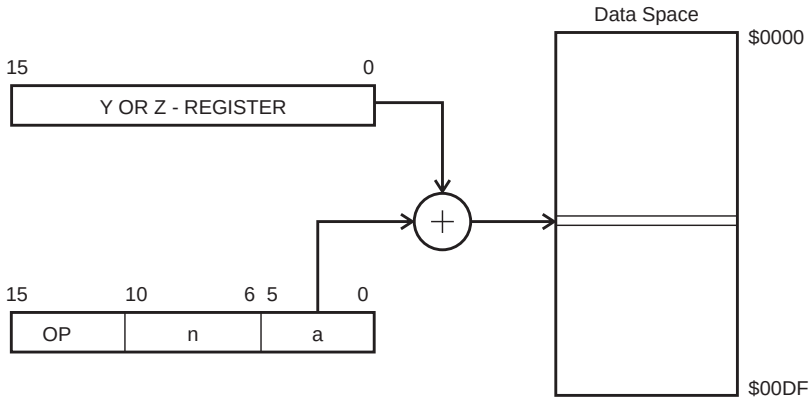
Figure 9. 直接数据寻址



16 位数据地址包含在两字指令的 16 LSB 中。 Rd/Rr 列出目的或源寄存器。

带偏移量的数据间接

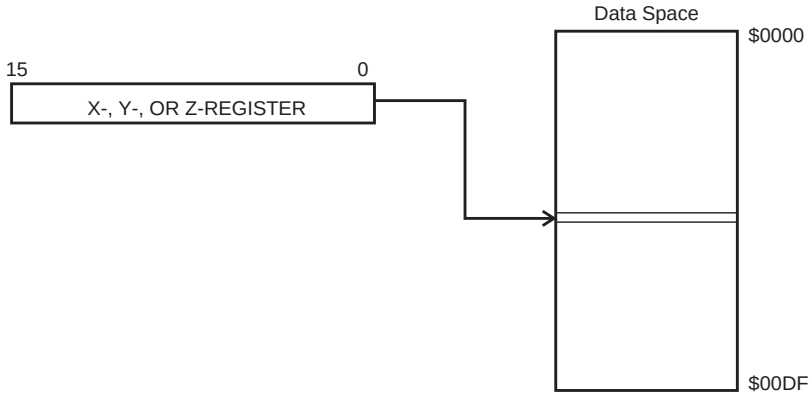
Figure 10. 带偏移量的数据间接



操作数地址是 Y 或 Z 寄存器内容与指令字中含有地址的 6 位相加的结果。

数据间接

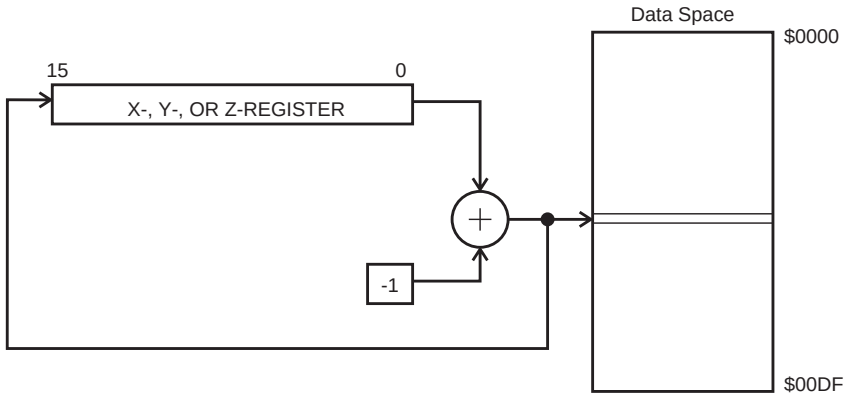
Figure 11. 数据间接寻址



操作数地址为 X、 Y 或 Z 寄存器中的内容。

带偏移量的数据间接

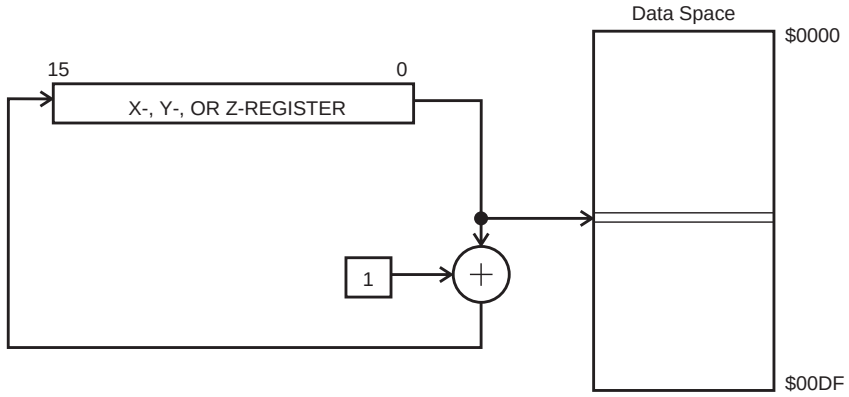
Figure 12. 带偏移量的数据间接寻址



在操作前，X、Y 或 Z 寄存器自减，操作数地址为 X、Y 或 Z 寄存器自减后的内容。

带后增量的间接数据

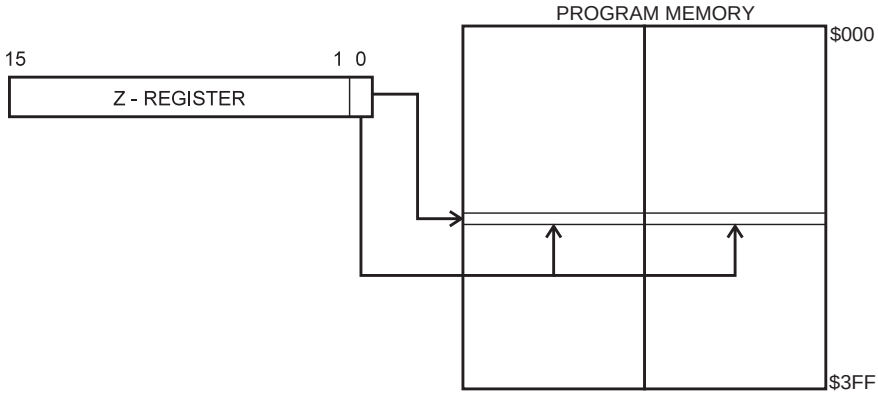
Figure 13. 带后增量的间接数据寻址



X、Y 或 Z 寄存器在操作结束后自加。操作数地址为 X、Y 或 Z 寄存器在自加前的内容。

使用 LPM 指令的常数寻址

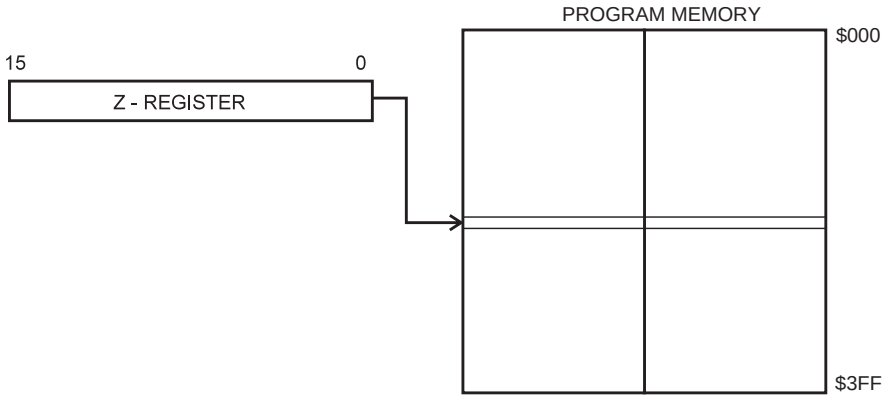
Figure 14. 代码存储器常数寻址



常数字节地址由 Z 寄存器内容给出。15 MSB 选择字地址 (0 - 1K)，LSB = 0 时选择低字节，LSB = 1 时选择高字节。

间接程序寻址，IJMP 与 ICALL

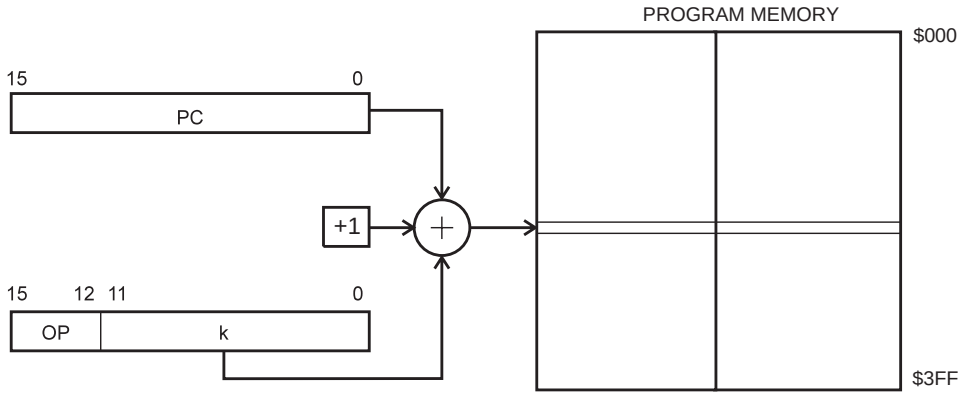
Figure 15. 间接程序存储器寻址



程序继续在 Z 寄存器给出的地址上执行 (即，PC 从 Z 寄存器下载内容)。

相关程序寻址，RJMP 与 RCALL

Figure 16. 相关程序存储器寻址



程序在地址 PC + k + 1 处执行。相关地址 k 从 -2048 到 2047。

EEPROM 数据存储器

ATtiny26(L) 包含128字节的 EEPROM数据存储器。它是作为一个独立的数据空间而存在的，可以按字节读写。 EEPROM 的寿命至少为 100,000 次擦除周期。 EEPROM 的访问由地址寄存器、数据寄存器和控制寄存器决定，详见 P58“EEPROM 读 / 写 访问 ”。

对 EEPROM 的编程请参见 P103“ 存储器编程 ”

存储器访问时间与指令执行时序

本节说明指令执行与内部存储器访问的通用访问时序概念。

AVR CPU 由系统时钟驱动，由外部时钟晶体产生。芯片内部不对此时钟进行分频。

Figure17 说明了由 Harvard 结构决定的并行取指和指令执行，以及可以进行快速访问的寄存器文件的概念。这是一个基本的流水线概念，性能高达 1 MIPS/MHz，具有优良的性价比、功能 / 时钟比、功能 / 功耗比。

Figure 17. 并行取指和指令执行

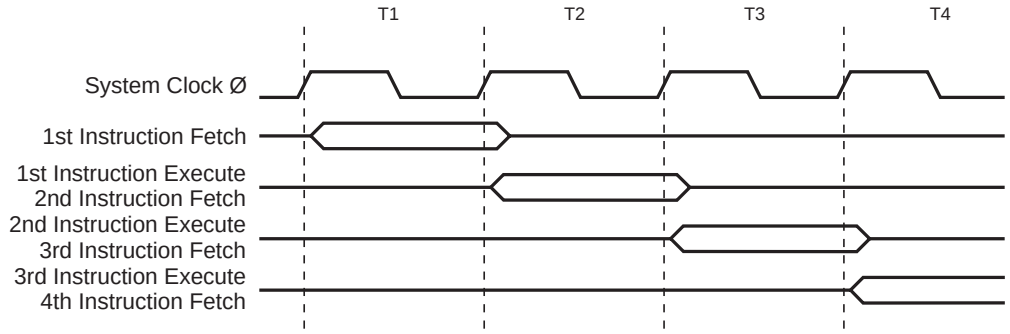
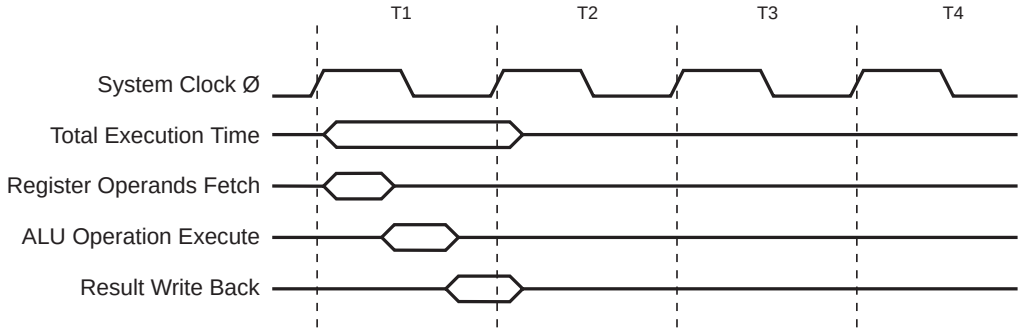


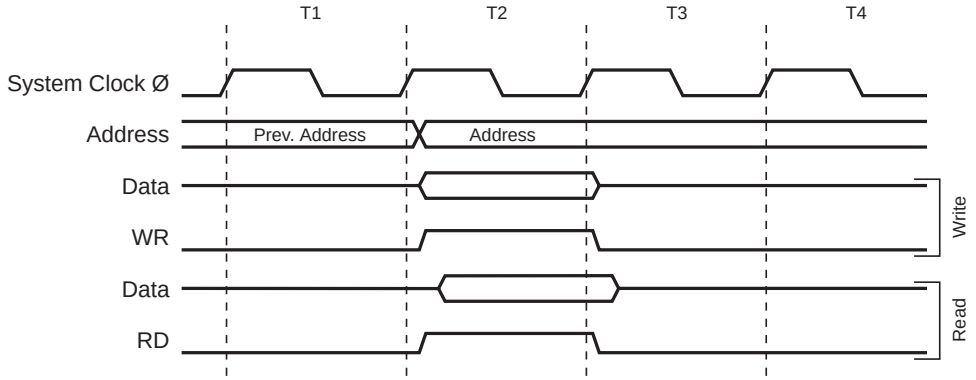
Figure18 演示的是寄存器文件内部访问时序。在一个时钟周期里， ALU 可以同时对两个寄存器操作数进行操作，同时将结果保存到目的寄存器中去。

Figure 18. 单时钟周期 ALU 操作



如 Figure19 所示，内部数据 SRAM 访问在两个系统时钟中完成。

Figure 19. 片内数据 SRAM 访问周期



I/O 存储器

ATtiny26(L) 的 I/O 空间定义见 Table 1

Table 1. ATtiny26(L) I/O 空间 (1)

地址	名称	功能
$3F ($5F)	SREG	状态寄存器
$3D ($5D)	SP	堆栈指针
$3B ($5B)	GIMSK	通用中断屏蔽寄存器
$3A ($5A)	GIFR	通用中断标志寄存器
$39 ($59)	TIMSK	T/C 中断屏蔽寄存器
$38 ($58)	TIFR	T/C 中断标志寄存器
$35 ($55)	MCUCR	MCU 控制寄存器
$34 ($54)	MCUSR	MCU 状态寄存器
$33 ($53)	TCCR0	T/C0 控制寄存器
$32 ($52)	TCNT0	T/C0 (8 位)
$31 ($51)	OSCCAL	振荡器标定寄存器
$30 ($50)	TCCR1A	T/C1 控制寄存器 A
$2F ($4F)	TCCR1B	T/C1 控制寄存器 B
$2E ($4E)	TCNT1	T/C1 (8 位)
$2D ($4D)	OCR1A	T/C1 输出比较寄存器 A
$2C ($4C)	OCR1B	T/C1 输出比较寄存器 B
$2B ($4B)	OCR1C	T/C1 输出比较寄存器 C
$29 ($29)	PLLCSR	PLL 控制与状态寄存器
$21 ($41)	WDTCR	看门狗定时器控制寄存器
$1E ($3E)	EEAR	EEPROM 地址寄存器
$1D ($3D)	EEDR	EEPROM 数据寄存器
$1C ($3C)	EECR	EEPROM 控制寄存器
$1B ($3B)	PORTA	数据寄存器，端口 A
$1A ($3A)	DDRA	数据方向寄存器，端口 A
$19 ($39)	PINA	输入引脚，端口 A
$18 ($38)	PORTB	数据寄存器，端口 B
$17 ($37)	DDRB	数据方向寄存器，端口 B
$16 ($36)	PINB	输入引脚，端口 B
$0F ($2F)	USIDR	通用串行接口数据寄存器
$0E ($2E)	USISR	通用串行接口状态寄存器
$0D ($2D)	USICR	通用串行接口控制寄存器
$08 ($28)	ACSR	模拟比较器控制与状态寄存器
$07 ($27)	ADMUX	ADC 多路复用选择寄存器

Table 1. ATtiny26(L) I/O 空间 (1) (Continued)

地址	名称	功能
$06($26)	ADCSR	ADC 控制与状态寄存器
$05($25)	ADCH	ADC 数据寄存器高
$04($24)	ADCL	ADC 数据寄存器低

Note: 1. 保留与未用地址在表中未列出。

ATtiny26(L) 所有的I/O及外设都被放置于I/O空间。所有的I/O位置都可以通过IN 与OUT指令来访问 , 在 32 个通用工作寄存器和 I/O 之间传输数据。地址为 $00 - $1F 的 I/O 寄存器还可用 SBI 和 CBI 指令直接进行位寻址 , 而 SBIS 和 SBIC 则用来检查某一位的值。更多内容请参见指令集。为了与后续产品兼容 , 保留位在执行写操作时应写 "0" , 而保留的 I/O 寄存器则不应进行写操作。

I/O 和外设控制寄存器在后续其他章节进行介绍。

状态寄存器 - SREG

AVR 状态寄存器 – SREG – I/O 地址为 $3F

Bit	7	6	5	4	3	2	1	0	
$3F ($5F)	I	T	H	S	V	N	Z	C	SREG
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

- **Bit 7 – I: 全局中断使能**

I 置位时使能全局中断。单独的中断使能由其他独立的控制寄存器控制。如果 I 清零 , 则不论单独中断标志置位与否，都不会产生中断。任意一个中断发生后 I 清零，而执行 RETI 指令后 I 恢复置位以使能中断。 I 也可以通过 SEI 和 CLI 指令来置位和清零。

- **Bit 6 – T: 位拷贝存储**

位拷贝指令 BLD 和 BST 利用 T 作为目的或源地址。BST 把寄存器的某一位拷贝到 T , 而 BLD 把 T 拷贝到寄存器的某一位。

- **Bit 5 – H: 半进位标志**

半进位标志 H 表示算术操作发生了半进位。此标志对于 BCD 运算非常有用。详见指令集的说明。

- **Bit 4 – S: 符号位 , S = N ⊕ V**

S 为负数标志 N 与 2 的补码溢出标志 V 的异或。详见指令集的说明。

- **Bit 3 – V: 2 的补码溢出标志**

支持 2 的补码运算。详见指令集的说明。

- **Bit 2 – N: 负数标志**

表明算术或逻辑操作结果为负。详见指令集的说明。

- **Bit 1 – Z: 零标志**

表明算术或逻辑操作结果为零。详见指令集的说明。

- **Bit 0 – C: 进位标志**

表明算术或逻辑操作发生了进位。详见指令集的说明。

堆栈指针 - SP

ATtiny26(L) 的堆栈指针由I/O空间中的8位寄存器实现。由于ATtiny26(L)数据存储器只有224 个位置，因此使用 8 位就足够了。

Bit	7	6	5	4	3	2	1	0	
$3D ($5D)	**SP7**	**SP6**	**SP5**	**SP4**	**SP3**	**SP2**	**SP1**	**SP0**	**SP**
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

堆栈指针指向数据 SRAM 堆栈区。在此聚集了子程序堆栈和中断堆栈。调用子程序和使能中断之前必须定义堆栈空间，且堆栈指针必须指向高于 $60 的地址空间。使用 PUSH 指令将数据推入堆栈时指针减一；而子程序或中断返回地址推入堆栈时指针将减二。使用 POP 指令将数据弹出堆栈时，堆栈指针加一；而用 RET 或 RETI 指令从子程序或中断返回时堆栈指针加二。

复位与中断处理

ATtiny26(L) 有十一个中断源。每个中断和复位在程序空间都有独立的中断向量。所有的中断事件都有自己的使能位。当使能位置位，且状态寄存器的全局中断使能位I也置位时，中断可以发生。

程序存储区的最低地址缺省为复位向量和中断向量。完整的向量列表请参见 Table 2。列表也决定了不同中断的优先级。向量所在的地址越低，优先级越高。RESET 具有最高的优先级，第二个为 INT0 – 外部中断请求 0。

Table 2. 复位与中断向量

向量号	程序地址	源	中断定义
1	$000	RESET	硬件引脚与看门狗复位
2	$001	INT0	外部中断请求 0
3	$002	I/O Pins	引脚触发中断
4	$003	TIMER1, CMPA	T/C1 比较匹配 1A
5	$004	TIMER1, CMPB	T/C1 比较匹配 1B
6	$005	TIMER1, OVF1	T/C1 溢出
7	$006	TIMER0, OVF0	T/C0 溢出
8	$007	USI_STRT	USI 启动
9	$008	USI_OVF	USI 溢出
A	$009	EE_RDY	EEPROM 准备好
B	$00A	ANA_COMP	模拟比较器
C	$00B	ADC	ADC 转换完

程序启动最典型与通用的复位与中断向量地址为：

```
地址      标签 代码                          注释
$000              rjmp    RESET           ; 复位句柄
$001              rjmp    EXT_INT0        ; IRQ0 句柄
$002              rjmp    PIN_CHANGE      ; 引脚变化句柄
$003              rjmp    TIM1_CMP1A      ; 定时器 1 比较匹配 1A
$004              rjmp    TIM1_CMP1B      ; 定时器 1 比较匹配 1B
$005              rjmp    TIM1_OVF        ; 定时器 1 溢出句柄
$006              rjmp    TIM0_OVF        ; 定时器 0 溢出句柄
$007              rjmp    USI_STRT        ; USI 启动句柄
$008              rjmp    USI_OVF         ; USI 溢出句柄
$009              rjmp    EE_RDY          ; EEPROM 准备好句柄
$00A              rjmp    ANA_COMP        ; 模拟比较器句柄
$00B              rjmp    ADC             ; ADC 转换句柄
;
$009    RESET:    ldi     r16, RAMEND     ; 主程序启动
$00A              out     SP, r16
$00B              sei
  …        …         …      …
```

复位源

ATtiny26(L) 提供四种复位源：

- 上电复位。当提供的电压低于掉电复位阈值 V_{POT}，MCU 复位。
- 外部复位。使用 PB7/$\overline{\text{RESET}}$ 引脚作为外部复位，覆盖 I/O 功能，RSTDISBL 熔丝位未编程 (“1”)，当 $\overline{\text{RESET}}$ 引脚上低电压时间大于 50 ns 时， MCU 复位。
- 看门狗复位。当看门狗定时器周期完且看门狗使能， MCU 复位。
- 掉电检测复位。当提供电压 V_{CC} 低于掉电检测复位阈值 V_{BOT}，MCU 复位。

复位时，所有的 I/O 寄存器恢复初始值，程序从地址 $000 开始执行。地址 $000 的指令必须为一条相对跳转指令 RJMP 来复位程序。若程序不使能中断源，中断向量将不被使用，从而普通程序代码可存储于这些位置。 Figure20 给出了 ATtiny26(L) 的复位逻辑。Table 3 则给出了 ATtiny26(L) 复位电路的时序与参数。

Figure 20. ATtiny26(L) 的复位逻辑

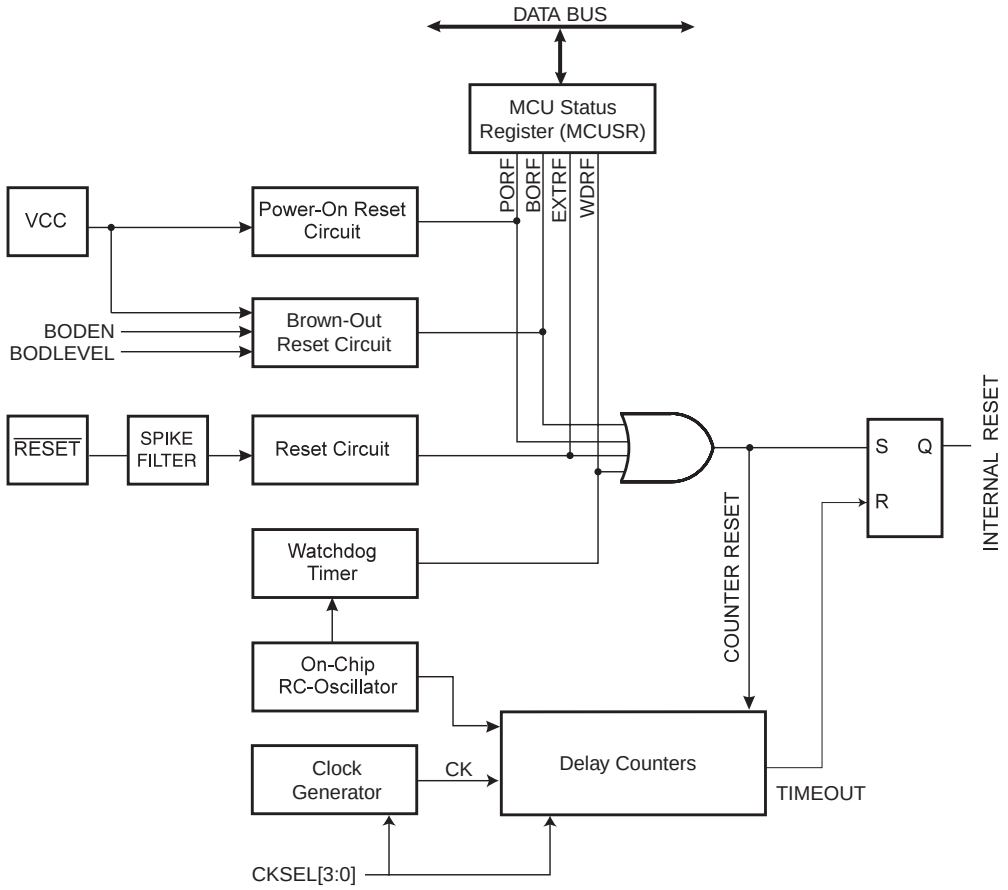


Table 3. 复位特性

符号	参数	状态	最小	典型	最大	单位
V_{POT}	上电复位阈值电压 (上升沿)			1.4	2.3	V
	上电复位阈值电压 (下降沿)(1)			1.3	2.3	V
V_{RST}	$\overline{\text{RESET}}$ 引脚阈值电压		0.2		0.9	V_{CC}
t_{RST}	$\overline{\text{RESET}}$ 引脚最小脉宽			750		ns

Table 3. 复位特性

符号	参数	状态	最小	典型	最大	单位
V_{BOT}	掉电检测复位阈值电压 (2)	BODLEVEL = 1	2.5	2.7	3.2	V
		BODLEVEL = 0	3.7	4.0	4.2	
t_{BOD}	掉电检测最小低电压周期	BODLEVEL = 1		2		µs
		BODLEVEL = 0		2		µs
V_{HYST}	掉电检测器迟滞			130		mV

Notes: 1. 只有当电压低于 V_{POT} (下降沿) , 才启动上电复位。
2. V_{BOT} 可能低于芯片的标称工作电压。若存在这种情况，应在芯片生产时检测 V_{CC} = V_{BOT} 时的情况，以保证在 V_{CC} 低于正常工作电压前出现掉电检测复位。测试时，对于 ATtiny26L，要求BODLEVEL=1 ；而对ATtiny26，则BODLEVEL=0。BODLEVEL=1 对 ATtiny26 无效。

复位启动时间请参见 P24“ 系统时钟及时钟选项 ”。当 CPU 从掉电模式被唤醒时，只利用了启动时间的时钟记数部分。看门狗振荡器用来为启动时间的实时部分定时。

上电复位

上电复位 (POR) 脉冲由片上检测电路产生。 Table 3 中定义了检测电压。当 V_{CC} 低于检测电压后 POR 被激活。 POR 电路在检测到错误电压时触发启动复位。

上电复位 (POR) 电路保证芯片从上电复位。达到上电复位阈值电压时会激活确定延迟时间的延迟计数器，使器件在 V_{CC} 上升后还保持 RESET 状态。延迟计数器的延迟时间由 CKSEL熔丝位决定。延迟周期的不同选择见P24“系统时钟及时钟选项”。当V_{CC} 低于检测电压后，RESET 信号被立即激活。

Figure 21. 通过 VCC， MCU 启动 $\overline{\text{RESET}}$

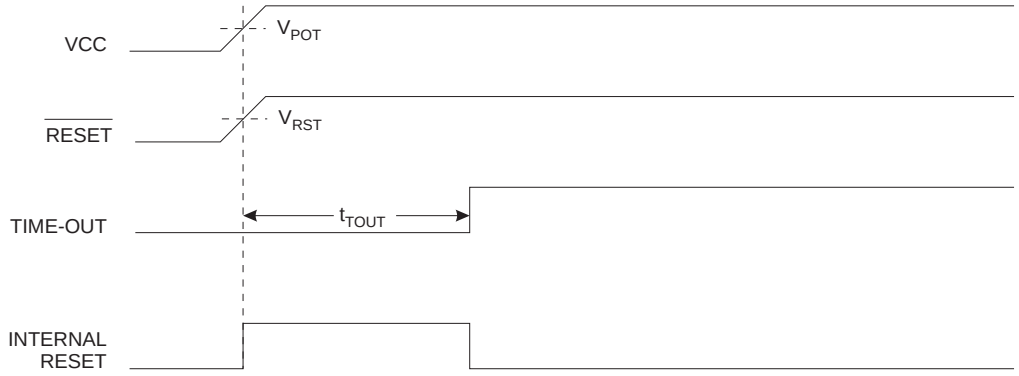
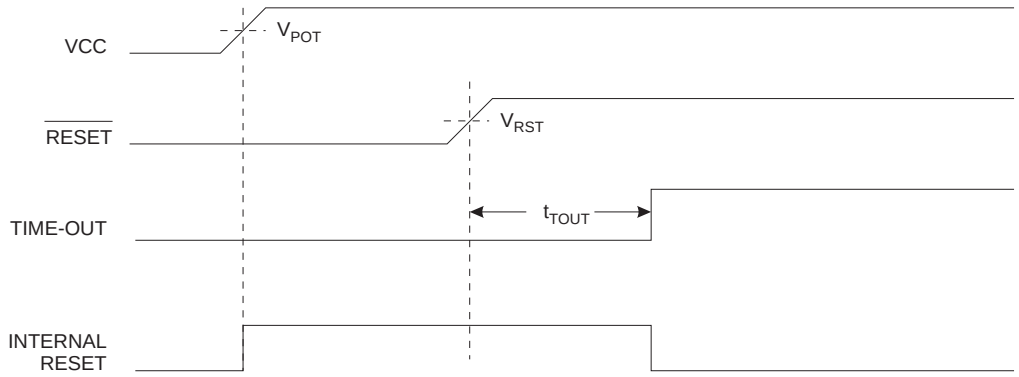


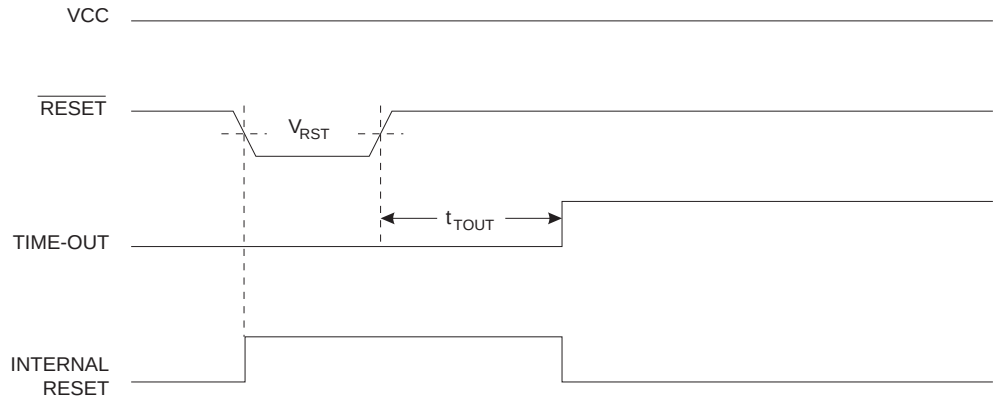
Figure 22. 外部控制 MCU 启动 $\overline{\text{RESET}}$



外部复位

在 $\overline{\text{RESET}}$ 引脚上加低电压产生外部中断。即使时钟不运行，只要复位脉冲超过 500 ns 就产生复位。过短的脉冲不保证产生复位。当信号的上跳沿达到复位阈值电压 V_{RST} 后，经过溢出时间 t_{TOUT}，延迟定时器启动 MCU。

Figure 23. 运行中外部复位

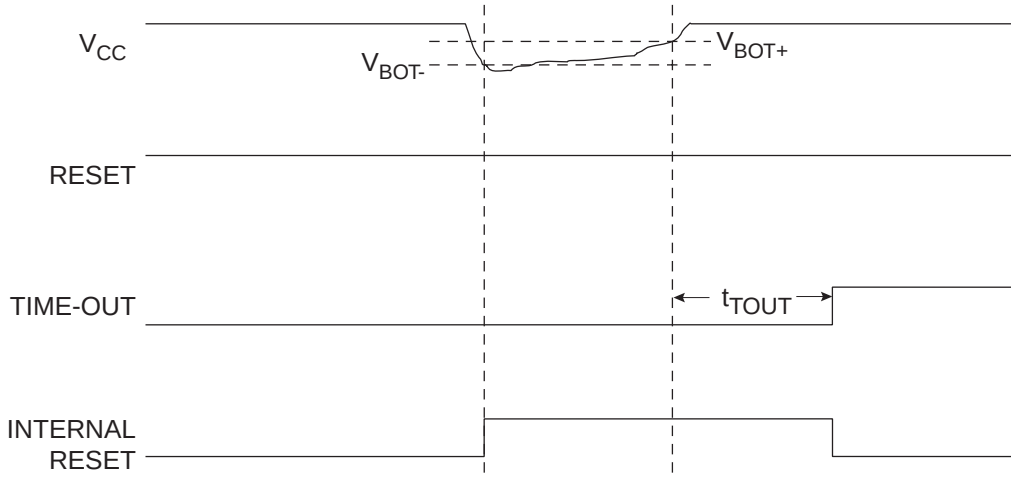


掉电检测复位

ATtiny26(L) 有片内 BOD 电路，用于在系统运行时对系统电压 V_{CC} 的检测。BOD 电路通过 BODEN 熔丝位决定其使能与否。当 BOD 使能且 V_{CC} 低于触发电压，掉电检测复位被立即触发。当 V_{CC} 大于触发电压，掉电检测复位在延迟后停用。延迟的定义与 Table 2 中 POR 信号延迟相同。BOD 检测阈值可通过 BODLEVEL 熔丝位设定为 2.7V (BODLEVEL 未编程)，或 4.0V (BODLEVEL 已编程)。阈值电压有 50 mV 的迟滞效应，以避免系统电源的毛刺误触发 BOD。

若电压低于触发电压的时间长于 Table 3 中给出的 t_{BOD} ，BOD 电路将只检测 V_{CC} 的下降过程。

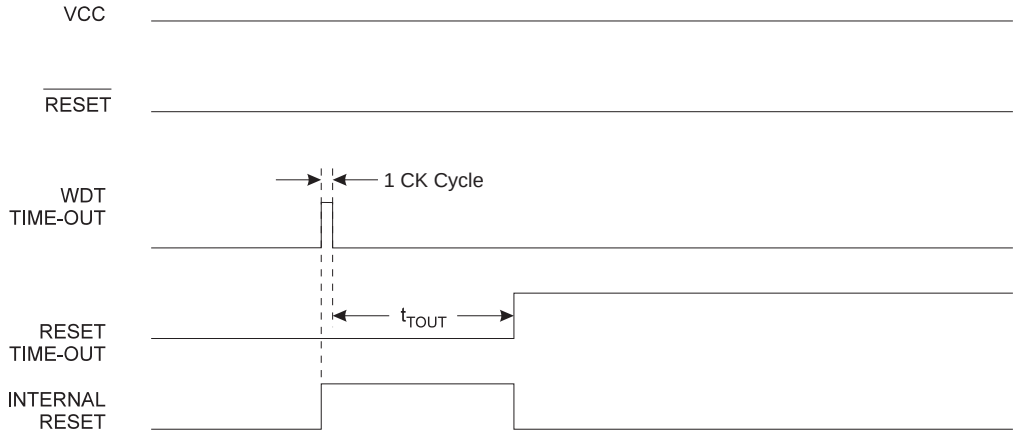
Figure 24. 运行中 BOD 复位



看门狗复位

当看门狗定时器溢出时，它将触发产生一个时钟周期宽度的复位脉冲，从脉冲的下降开始，经过设定的启动延时时间， MCU 启动运行，看门狗工作详见 P56。

Figure 25. 看门狗溢出

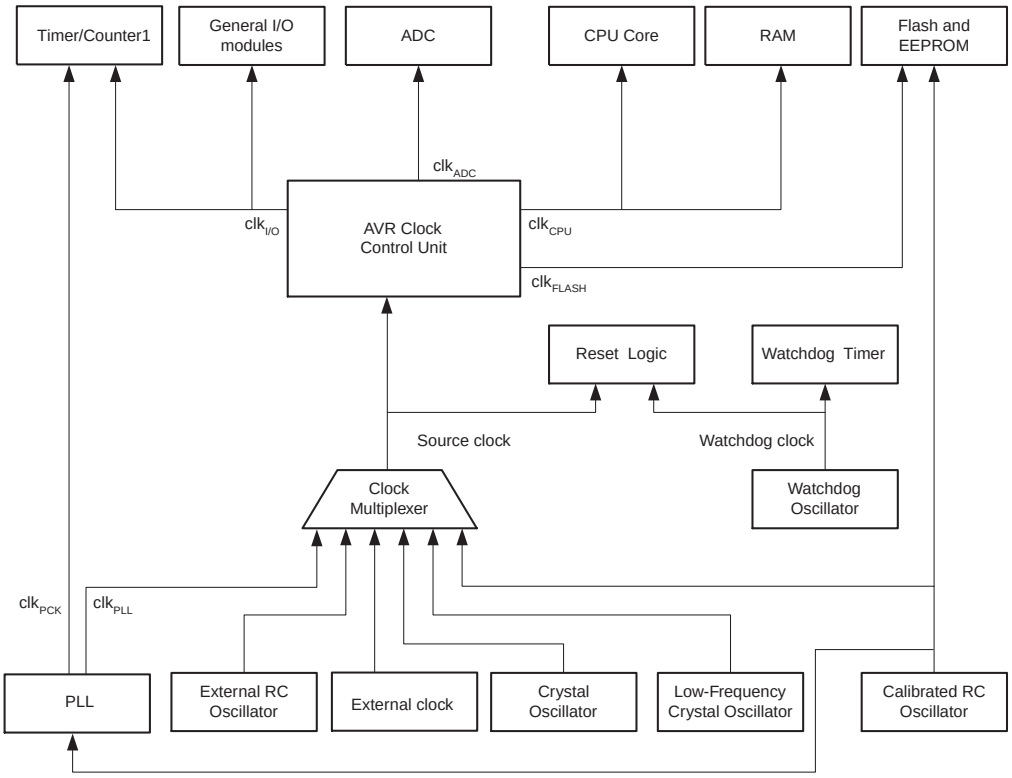


系统时钟及时钟选项

时钟系统及其分布

Figure26 为AVR的主要时钟系统及其分布。这些时钟并不需要同时工作。为了降低功耗，可以通过使用不同的睡眠模式来禁止无需工作的模块的时钟，详见 P38“ 电源管理及睡眠模式 ”。时钟系统详见下图。

Figure 26. 时钟分布



CPU 时钟 - clk_{CPU}

CPU时钟与操作AVR内核的子系统相连，如通用寄存器文件、状态寄存器及保存堆栈指针的数据存储器。终止 CPU 时钟将使内核停止工作和计算。

I/O 时钟 - $clk_{I/O}$

I/O时钟用于主要的I/O模块，如定时器/计数器和USI。I/O时钟还用于外部中断模块。要注意的是有些外部中断由异步逻辑检测，因此即使 I/O 时钟停止了这些中断仍然可以得到监控。

Flash 时钟 - clk_{FLASH}

Flash 时钟控制 Flash 接口的操作。此时钟通常与 CPU 时钟同时挂起或激活。

ADC 时钟 - clk_{ADC}

ADC具有专门的时钟。这样可以在ADC工作的时候停止CPU和I/O时钟以降低数字电路产生的噪声，从而提高 ADC 转换精度。

为外设产生快速时钟的片内 PLL - clk_{PCK}

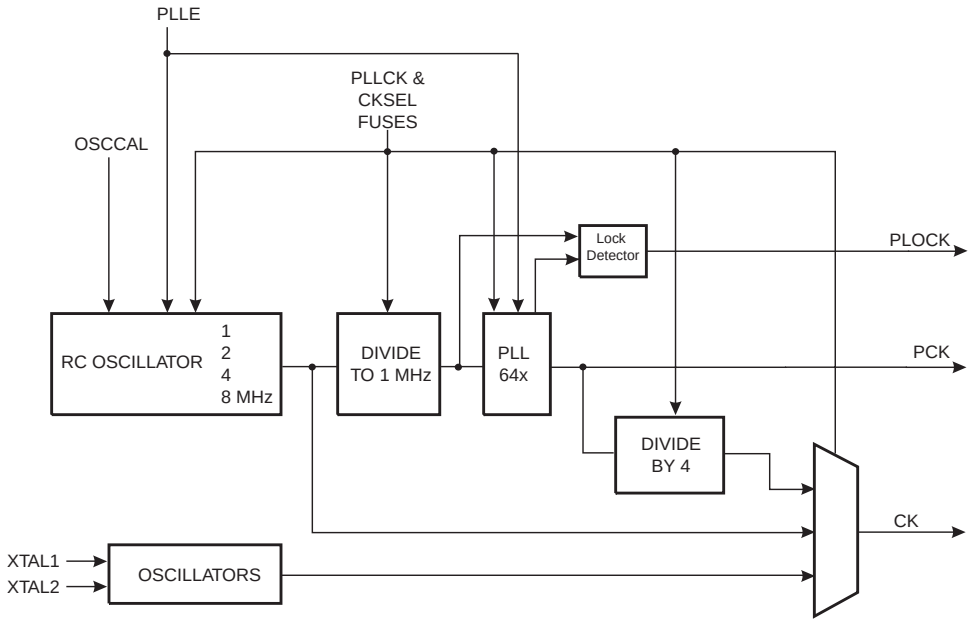
ATtiny26(L) 片上PLL产生的时钟频率为(标称为)1 MHz的输入时钟的64倍。1 MHz的输入时钟源是内部 RC 振荡器分频后得到的，祥见 P24Figure27 。当 PLL 参考频率为 1 MHz 时，快速外设时钟为 64 MHz。快速外设时钟或此时钟的分频信号可作为 T/C1 的时钟源。

PLL锁定于RC振荡器。因此通过OSCCAL寄存器调整RC振荡器时同时也调整了快速外设时钟。但即使 RC 振荡器频率超过了 1 MHz，快速外设频率最大值将饱和于 70 MHz 并保持在此最大频率上。要注意，此时 PLL 并未锁定 RC 振荡器时钟。

因此不推荐使用 OSCCAL 调整 RC 振荡器频率高于 1 MHz，以保证 PLL 的正常工作范围。只有当寄存器 PLLCSR 的 PLLE 位置位或 PLLCK 熔丝位编程 (“0”)，内部 PLL 才使能。PLL 锁定时， PLLCSR 寄存器的 PLOCK 位置位。

在掉电与 Standby 模式下，内部 1 MHz RC 振荡器与 PLL 均关闭。

Figure 27. PCK 时钟系统



时钟源

芯片有如下几种通过 Flash 熔丝位进行选择的时钟源，见 Table 4。时钟输入到 AVR 时钟发生器，再分配到相应的模块。引脚 PB5 (XTAL2) 与 PB4 (XTAL1) 作为 I/O 引脚受限于时钟设置，参见 Table 5。

Table 4. 时钟源选择

器件时钟选项	PLLCK	CKSEL3..0
外部晶体 / 陶瓷振荡器	1	1111 - 1010
外部低频晶振	1	1001
外部 RC 振荡器	1	1000 - 0101
标定的内部 RC 振荡器	1	0100 - 0001
外部时钟	1	0000
PLL 时钟	0	0001

Table 5. PB5 与 PB4 功能与器件时钟选项的关系 (1)

器件时钟选项	PLLCK	CKSEL [3:0]	PB4	PB5
外部时钟	1	0000	XTAL1	I/O
内部 RC 振荡器	1	0001	I/O	I/O
内部 RC 振荡器	1	0010	I/O	I/O
内部 RC 振荡器	1	0011	I/O	I/O
内部 RC 振荡器	1	0100	I/O	I/O
外部 RC 振荡器	1	0101	XTAL1	I/O
外部 RC 振荡器	1	0110	XTAL1	I/O
外部 RC 振荡器	1	0111	XTAL1	I/O
外部 RC 振荡器	1	1000	XTAL1	I/O
外部低频振荡器	1	1001	XTAL1	XTAL2
外部晶体 / 陶瓷振荡器	1	1010	XTAL1	XTAL2
外部晶体 / 陶瓷振荡器	1	1011	XTAL1	XTAL2
外部晶体 / 陶瓷振荡器	1	1100	XTAL1	XTAL2
外部晶体 / 陶瓷振荡器	1	1101	XTAL1	XTAL2
外部晶体 / 陶瓷振荡器	1	1110	XTAL1	XTAL2
外部晶体 / 陶瓷振荡器	1	1111	XTAL1	XTAL2
PLL	0	0001	I/O	I/O

Note: 1. 对于所有的熔丝位，“1” 表示未编程，“0” 代表已编程。

不同的时钟选项将在后续部分进行介绍。当 CPU 自掉电模式或省电模式唤醒之后，被选择的时钟源用来为启动过程定时，保证振荡器在开始执行指令之前进入稳定状态。当CPU 从复位开始工作时，还有额外的延迟时间以保证在 MCU 开始正常工作之前电源达到稳定

电平。这个启动时间的定时由看门狗振荡器完成。看门狗溢出时间所对应的 WDT 振荡器周期数列于 Table 6。看门狗振荡器的频率由工作电压决定。

Table 6. 看门狗振荡器周期数

典型的溢出时间 (V_{CC} = 5.0V)	典型的溢出时间 (V_{CC} = 3.0V)	时钟周期数
4.1 ms	4.3 ms	4K (4,096)
65 ms	69 ms	64K (65,536)

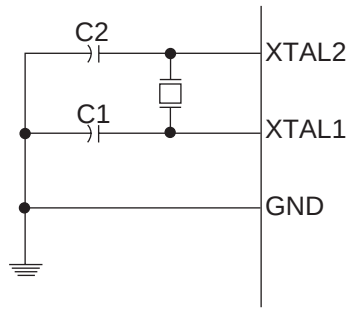
缺省时钟源

器件出厂时 CKSEL = “0001”， SUT = “10” 且 PLLCK 未编程。这个缺省设置的时钟源是 1 MHz的内部RC振荡器，启动时间为最长。这种设置保证用户在使用ISP或并行编程器时可以得到所需的时钟源。

晶体振荡器

.XTAL1 与 XTAL2 分别为用作片内振荡器的反向放大器的输入和输出，如 Figure28 所示。这个振荡器可以使用石英晶体，也可以使用陶瓷谐振器。振荡器的最大频率为 12 MHz。当使用该时钟选项，当保持 CKOPT 为未编程状态。 C1 和 C2 的数值要一样。最佳的数值与使用的晶体或谐振器有关，还与杂散电容和环境的电磁噪声有关。Table 7 给出了针对晶体选择电容的一些指南。对于陶瓷谐振器，应该使用厂商提供的数值。

Figure 28. 晶体振荡器连接图



振荡器可以工作于三种不同的模式，每一种都有一个优化的频率范围。工作模式通过熔丝位 CKSEL3..1 来选择，如 Table 7 所示。

Table 7. 晶体振荡器工作模式

CKSEL3..1	频率范围 (MHz)	使用晶体时电容 C1 和 C2 的推荐范围 (pF)
101[1]	0.4 - 0.9	–
110	0.9 - 3.0	12 - 22
111	3.0 - 16	12 - 22
	16 -	12 - 15

Note: 1. 此选项不适用于晶体，只能用于陶瓷谐振器。

如 Table 8 所示，熔丝位 CKSEL0 以及 SUT1..0 用于选择启动时间。

Table 8. 晶体振荡器时钟选项对应的启动时间

CKSEL0	SUT1..0	掉电与节电模式下的启动时间	复位时额外的延迟时间 (V_{CC} = 5.0V)	推荐用法
0	00	258 CK(1)	4.1 ms	陶瓷谐振器，电源快速上升
0	01	258 CK(1)	65 ms	陶瓷谐振器，电源缓慢上升
0	10	1K CK(2)	–	陶瓷谐振器， BOD 使能
0	11	1K CK(2)	4.1 ms	陶瓷谐振器，电源快速上升
1	00	1K CK(2)	65 ms	陶瓷谐振器，电源缓慢上升
1	01	16K CK	–	石英振荡器，BOD 使能
1	10	16K CK	4.1 ms	石英振荡器，电源快速上升
1	11	16K CK	65 ms	石英振荡器，电源慢速上升

Notes: 1. 这些选项只能用于工作频率不太接近于最大频率，而且启动时的频率稳定性对于应用而言不重要的情况。
2. 这些选项是为陶瓷谐振器设计的，可以保证启动时频率足够稳定。若工作频率不太接近于最大频率，而且启动时的频率稳定性对于应用而言不重要时也适用于晶体。

低频晶体振荡器

为了使用 32.768 kHz 钟表晶体作为器件的时钟源，必须将 PLLCK 设置为 “1”，熔丝位 CKSEL设置为“1001”以选择低频晶体振荡器。晶体的连接方式如Figure28所示。通过对熔丝位 CKOPT 的编程，用户可以使能 XTAL1 和 XTAL2 的内部电容，从而去除外部电容。内部电容的标称数值为 36 pF。

选择了这个振荡器之后，启动时间由熔丝位 SUT 确定，如 Table 9 所示。

Table 9. 低频晶体振荡器的启动时间

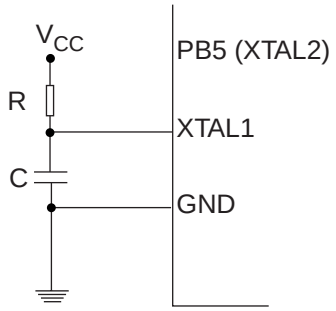
SUT1..0	掉电模式和省电模式的启动时间	复位时的额外延迟时间 (V_{CC} = 5.0V)	推荐用法
00	1K CK(1)	4.1 ms	电源快速上升，或是 BOD 使能
01	1K CK(1)	65 ms	电源缓慢上升
10	32K CK	65 ms	启动时频率已经稳定
11	保留		

Note: 1. 这些选项只能用于启动时的频率稳定性对应用而言不重要的情况。

外部 RC 振荡器

对于时间不敏感的应用可以使用 Figure29 的外部 RC 振荡器。频率可以通过方程 f = 1/(3RC) 进行粗略地鼓估计。电容 C 至少要 22 pF。通过编程熔丝位 CKOPT ,用户可以使能 XTAL1 和 GND 之间的片内 36 pF 电容，从而无需外部电容。

Figure 29. 外部 RC 配置



振荡器可以工作于四个不同的模式，每个模式有自己的优化频率范围。工作模式通过熔丝位 CKSEL3..0 选取，如 Table 10 所示。

Table 10. 外部 RC 振荡器工作模式

CKSEL3..0	频率范围 (MHz)
0101	- 0.9
0110	0.9 - 3.0
0111	3.0 - 8.0
1000	8.0 - 12.0

选择了这个振荡器之后，启动时间由熔丝位 SUT 确定，如 Table 11 所示。

Table 11. 外部 RC 振荡器的启动时间

SUT1..0	掉电模式的启动时间	复位时的额外延迟时间 (V_{CC} = 5.0V)	推荐用法
00	18 CK	–	BOD 使能
01	18 CK	4.1 ms	电源快速上升
10	18 CK	65 ms	电源缓慢上升
11	6 CK(1)	4.1 ms	电源快速上升，或是 BOD 使能

Notes: 1. 这些选项只能用于工作频率不太接近于最大频率时的情况。

标定的片内 RC 振荡器

标定的片内 RC 振荡器提供了固定的 1.0、 2.0、 4.0 或 8.0 MHz 的时钟。这些频率都是 5V、 25°C 下的标称数值。这个时钟也可以作为系统时钟，只要按照 Table 12 对熔丝位 CKSEL进行编程即可。选择这个时钟(此时不能对CKOPT进行编程)之后就无需外部器件了。复位时硬件将标定字节加载到 OSCCAL 寄存器，自动完成对 RC 振荡器的标定。在 5V ,25°C 和频率为 1.0 MHz 时 ,这种标定可以提供标称频率 ± 1% 的精度。当使用这个振荡器作为系统时钟时，看门狗仍然使用自己的看门狗定时器作为溢出复位的依据。更多的有关标定数据的信息请参见 P105“ 校准字节 ”。

Table 12. 片内标定的 RC 振荡器工作模式

CKSEL3..0	标称频率 (MHz)
0001(1)	1.0
0010	2.0
0011	4.0
0100	8.0

Note: 1. 出厂时的设置

选择了这个振荡器之后，启动时间由熔丝位 SUT 确定，如 Table 13 所示。 XTAL1 和 XTAL2 为通用 I/O 端口。

Table 13. 内部标定 RC 振荡器的启动时间

SUT1..0	掉电模式的启动时间	复位时的额外延迟时间 (V_{CC} = 5.0V)	推荐用法
00	6 CK	–	BOD 使能
01	6 CK	4.1 ms	电源快速上升
10(1)	6 CK	65 ms	电源缓慢上升
11	保留		

Note: 1. 出厂时的设置。

振荡器标定寄存器 - OSCCAL

Bit	7	6	5	4	3	2	1	0	
$31 ($51)	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	OSCCAL
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	标定数据								

- **Bits 7..0 – CAL7..0: 振荡器标定数据**

将标定数据写入这个地址可以对内部振荡器进行调节以消除由于生产工艺所带来的振荡器频率偏差。复位时 1 MHz 的标定数据 (标识数据的高字节，地址为 0x00) 自动加载到 OSCCAL 寄存器。如果需要内部 RC 振荡器工作于其他频率 ,标定数据必须人工加载 : 首先通过编程器读取标识数据，然后将标定数据保存到 Flash 或 EEPROM 之中。这些数据可以通过软件读取，然后加载到 OSCCAL 寄存器。当 OSCCAL 为零时振荡器以最低频率工作。当对其写入不为零的数据时内部振荡器的频率将增长。写入 $FF 即得到最高频率。标定的振荡器用来为访问 EEPROM 和 Flash 定时。有写 EEPROM 和 Flash 的操作

时不要将频率标定到超过标称频率的 10%，否则写操作有可能失败。要注意振荡器只对 1.0、2.0、4.0 和 8.0 MHz 这四种频率进行了标定，其他频率则无法保证，如 Table 14 所示

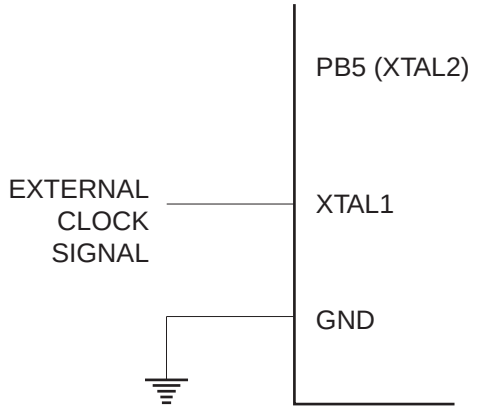
Table 14. 内部 RC 振荡器频率范围

OSCCAL 数值	最小频率，标称频率的百分比 (%)	最大频率，标称频率的百分比 (%)
$00	50%	100%
$7F	75%	150%
$FF	100%	200%

外部时钟

为了从外部时钟源驱动芯片， XTAL1 必须如 Figure30 所示的进行连接。同时，熔丝位 CKSEL必须编程为“0000”，且PLLCK为“1”。若熔丝位CKOPT也被编程，用户就可以使用内部的 XTAL1 和 GND 之间的 36 pF 电容。

Figure 30. 外部时钟配置图



选择了这个振荡器之后，启动时间由熔丝位 SUT 确定，如 Table 15 所示。

Table 15. 外部时钟的启动时间

SUT1..0	掉电模式的启动时间	复位时的额外延迟时间 (V_{CC} = 5.0V)	推荐用法
00	6 CK	–	BOD 使能
01	6 CK	4.1 ms	电源快速上升
10	6 CK	65 ms	电源缓慢上升
11	保留		

为了保证 MCU 能够稳定工作，不能突然改变外部时钟源的振荡频率。工作频率突变超过 2% 将会产生异常现象。应该在 MCU 保持复位状态时改变外部时钟的振荡频率。

高频 PLL 时钟 - PLL_{CLK}

芯片内有一个锁定于 RC 振荡器的 PLL，它输出标称为 64 MHz 的时钟，可以为 T/C1 与系统时钟所使用。作为系统时钟源时，要编程熔丝位 PLLCK(“0”)，使其被四分频。使用该选项时， CKSEL3..0 要设置为 “0001”。此时工作电压应当为 4.5 - 5.5V 以保证器件安全工作。此时，系统时钟频率为 16 MHz (64 MHz/4)。使用这个时钟选项时，启动时间由 Table 16 给出的 SUT 熔丝位决定，参见 P24“PCK 时钟系统 ” 。

Table 16. PLLCK 的启动时间

SUT1..0	掉电模式的启动时间	复位时的额外延迟时间 (V_{CC} = 5.0V)	推荐用法
00	1K CK	–	BOD 使能
01	1K CK	4.1 ms	电源快速上升
10	1K CK	65 ms	电源缓慢上升
11	16K CK	–	电源缓慢上升

MCU 状态寄存器 - MCUSR

Bit	7	6	5	4	3	2	1	0	
$34 ($54)	–	–	–	–	WDRF	BORF	EXTRF	PORF	MCUSR
读 / 写	R	R	R	R	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	参见各个位的说明				

- **Bit 7..4 – Res: 保留位**

ATtiny26(L) 中这些位为保留位，读返回值为 ”0”。

- **Bit 3 – WDRF: 看门狗复位标志**

看门狗复位发生时置位。上电复位将使其清零，也可以通过写 ”0” 来清除。

- **Bit 2 – BORF: 掉电检测复位标志**

掉电检测复位发生时置位。上电复位将使其清零，也可以通过写 ”0” 来清除。

- **Bit 1 – EXTRF: 外部复位标志**

外部复位发生时置位。上电复位将使其清零，也可以通过写 ”0” 来清除。

- **Bit 0 – PORF: 上电复位标志**

上电复位发生时置位。只能通过写 ”0” 来清除。

使用这些复位标志来识别复位条件时，用户应该尽早读取此寄存器的数据，然后将其复位。如果在其他复位发生之前将此寄存器复位，则后续复位源可以通过检查复位标志来了解。

中断处理

ATtiny26(L) 有两个 8 位中断屏蔽控制寄存器: GIMSK – 通用中断屏蔽寄存器和 TIMSK – T/C 中断屏蔽寄存器。

任一中断发生时全局中断使能位 I 被清零，从而禁止了所有其他的中断。用户软件可以在中断程序里置位 I 来实现中断嵌套。此时所有的中断都可以中断当前的中断服务程序。执行 RETI 指令后 I 自动置位。

程序计数器跳转到实际的中断向量以执行中断处理程序，同时硬件将清除相应的中断标志。中断标志也可以通过对其写 ”1” 的方式来清除。

当中断发生后，如果相应的中断使能位为 "0"，则中断标志位置位，并一直保持到中断执行，或者被软件清除。

如果全局中断标志被清零，则所有已发生的中断都不会被执行，直到 I 置位。然后挂起的各个中断按中断优先级依次执行。

要注意外部级中断不需要中断标志。若中断条件在中断使能之前就消失了，中断不会被触发。

要注意的是，进入中断服务程序时状态寄存器不会自动保存，中断返回时也不会自动恢复。这些工作必须由用户通过软件来完成。

中断响应时间

AVR 中断响应时间最少为 4 个时钟周期。4 个时钟周期后，程序跳转到实际的中断处理例程。在这 4 个时钟期期间 PC 自动入栈。在通常情况下，中断向量为一个跳转指令，此跳转需要 2 个时钟周期。如果中断在一个多时钟周期指令执行期间发生，则在此多周期指令执行完毕后 MCU 才会执行中断程序。

中断返回需要 4 个时钟。在此期间 PC(10 位) 将被弹出栈， AVR 退出中断后总是回到主程序并至少执行一条指令才可以去执行其他被挂起的中断。要注意的是，进入中断服务程序时状态寄存器不会自动保存，中断返回时也不会自动恢复。这些工作必须由用户通过软件来完成。

通用中断屏蔽寄存器 - GIMSK

Bit	7	6	5	4	3	2	1	0	
$3B ($5B)	–	INT0	PCIE1	PCIE0	–	–	–	–	GIMSK
读 / 写	R	R/W	R/W	R/W	R	R	R	R	
初始值	0	0	0	0	0	0	0	0	

- **Bit 7 – Res: 保留位**

在 ATtiny26(L) 中该位为保留位，读返回值为 ”0” 。

- **Bit 6 – INT0: 外部中断请求 0 使能**

当 INT0 为 '1’，而且状态寄存器 SREG 的 I 标志置位，相应的外部引脚中断就使能了。MCU通用控制寄存器– MCUCR的中断敏感电平控制0位 1/0 (ISC01与ISC00)决定中断是由上升沿、下降沿，还是 INT0 电平触发的。只要使能，即使 INT0 引脚被配置为输出，只要引脚电平发生了相应的变化，中断可将产生。外部中断请求 0 相应的中断从程序存储器地址 $001 开始执行，详见 P37“ 外部中断 ” 。

- **Bit 5 – PCIE1: 引脚电平变化中断使能 1**

当 PCIE1 位和状态寄存器的全局中断使能位 I 都为 ”1” 时，引脚中断变化在模拟引脚 PB[7:4]、PA[7:6] 与 PA[3] 处使能。除非屏蔽该中断，否则在上述引脚中任何电压变化都会引起中断。从程序存储器地址 $002，执行相应的引脚电平变化中断请求，请参见 P37“ 引脚变化中断 ” 。

- **Bit 4– PCIE0: 引脚电平变化中断使能 0**

当PCIE0位和状态寄存器的全局中断使能位I都为”1”时，引脚中断变化在数字引脚PB[3:0]处使能。除非屏蔽该中断，否则在上述引脚中任何电压变化都会引起中断。从程序存储器地址 $002，执行相应的引脚电平变化中断请求，请参见 P37“ 引脚变化中断 ” 。

- **Bits 3..0 – Res: 保留位**

在 ATtiny26(L) 中这几位为保留位，读返回值为 ”0” 。

通用中断标志寄存器 - GIFR

Bit	7	6	5	4	3	2	1	0	
$3A ($5A)	–	INTF0	PCIF	–	–	–	–	–	GIFR
读 / 写	R	R/W	R/W	R	R	R	R	R	
初始值	0	0	0	0	0	0	0	0	

- **Bit 7 – Res: 保留位**

在 ATtiny26(L) 中该位为保留位，读返回值为 ”0” 。

- **Bit 6 – INTF0: 外部中断标志 0**

INT0引脚电平发生跳变时触发中断请求，并置位相应的中断标志INTF0。如果SREG的位I以及GICR寄存器相应的中断使能位INT0为”1”，MCU即跳转到相应的中断向量。进入中断服务程序之后该标志自动清零。此外，标志位也可以通过写入 ”1” 来清零。

- **Bit 5 – PCIF: 引脚电平变化中断标志**

当引脚 PB[7:0]、PA[7:6] 或 PA[3] 的信号触发中断请求， PCIF 置 ”1”，PCIE1 从模拟引脚 PB[7:4]、 PA[7:6] 与 PA[3] 使能中断。 PCIE0 从数字引脚 PB[3:0] 使能中断。注意，若中断使能位 PCIE1 与 PCIE0 未置位，也会屏蔽标志。例如，若 PCIE0 清零，引脚 PB[3:0] 的变化不会置位 PCIF。若使能引脚第二功能，PCIF 位屏蔽。若 SREG 寄存器的 I 位与 GIMSK 寄存器的 PCIE 位置 ”1”， MCU 将跳转到中断向量地址 $002。当中断程序执行时，该位清零。该位也可通过写逻辑 ”1” 来清除，请参见 P37“ 引脚变化中断 ” 。

- **Bits 4..0 – Res: 保留位**

在 ATtiny26(L) 中这几位为保留位，读返回值为 ”0” 。

T/C 中断屏蔽寄存器 - TIMSK

Bit	7	6	5	4	3	2	1	0	
$39 ($59)	–	OCIE1A	OCIE1B	–	–	TOIE1	TOIE0	–	TIMSK
读 / 写	R	R/W	R/W	R	R	R/W	R/W	R	
初始值	0	0	0	0	0	0	0	0	

- **Bit 7 – Res: 保留位**

在 ATtiny26(L) 中该位为保留位，读返回值为 ”0” 。

- **Bit 6 – OCIE1A:T/C1 输出比较匹配中断使能**

当 OCIE1A 位和状态寄存器的全局中断使能位 I 都为 ”1” 时， T/C1 的输出比较匹配 A 中断使能。当 T/C1 的比较匹配 A 发生时，相应的位于 $003 的中断服务程序得以执行。在 T/C 中断标志寄存器中的 T/C1 的比较标志置 ”1”。

- **Bit 5 – OCIE1B: T/C1 输出比较匹配中断使能**

当 OCIE1B 位和状态寄存器的全局中断使能位 I 都为 ”1” 时， T/C1 的输出比较匹配 B 中断使能。当 T/C1 的比较匹配 B 发生时，相应的位于 $004 的中断服务程序得以执行。在 T/C 中断标志寄存器中的 T/C1 的比较标志置 ”1”。

- **Bit 4..3 – Res: 保留位**

在 ATtiny26(L) 中这几位为保留位，读返回值为 ”0” 。

- **Bit 2 – TOIE1: T/C1 溢出中断使能**

当 TOIE1 和状态寄存器的全局中断使能位 I 都为 ”1” 时，T/C1 的溢出中断使能。当 T/C1 发生溢出，即 TIFR 中的溢出标志位置位时，位于 $005 的中断服务程序得以执行。

- **Bit 1 – TOIE0: T/C0 溢出中断使能**

当TOIE0和状态寄存器的全局中断使能位I都为”1”时，T/C0的溢出中断使能。当T/C0发生溢出，即 TIFR 中的溢出标志位置位时，位于 $006 的中断服务程序得以执行。

- **Bit 0 – Res: 保留位**

在 ATtiny26(L) 中该位为保留位，读返回值为 ”0” 。

T/C 中断标志寄存器 - TIFR

Bit	7	6	5	4	3	2	1	0	
$38 ($58)	–	OCF1A	OCF1B	–	–	TOV1	TOV0	–	TIFR
读 / 写	R	R/W	R/W	R	R	R/W	R/W	R	
初始值	0	0	0	0	0	0	0	0	

- **Bit 7 – Res: 保留位**

在 ATtiny26(L) 中该位为保留位，读返回值为 ”0” 。

- **Bit 6 – OCF1A: 输出比较标志 1A**

当 T/C1 与 OCR1A(输出比较寄存器 1A) 的值匹配时， OCF1A 置位。此位在中断服务程序里硬件清零，也可以对其写 1 来清零。当 SREG 中的位 I、OCIE1A 和 OCF1A 都置位时，中断服务程序得到执行。

- **Bit 5 – OCF1B: 输出比较标志 1B**

当 T/C1 与 OCR1B(输出比较寄存器 1B) 的值匹配时， OCF1B 置位。此位在中断服务程序里硬件清零，也可以对其写 1 来清零。当 SREG 中的位 I、OCIE1B 和 OCF1B 都置位时，中断服务程序得到执行。

- **Bits 4..3 – Res: 保留位**

在 ATtiny26(L) 中这几位为保留位，读返回值为 ”0” 。

- **Bit 2 – TOV1:T/C1 溢出标志**

当 T/C1 溢出时， TOV1 置位。执行相应的中断服务程序时此位硬件清零。此外， TOV1 也可以通过写 1 来清零。当 SREG 中的位 I、 TOIE1(T/C1 溢出中断使能) 和 TOV1 都置位时，中断服务程序得到执行。

- **Bit 1 – TOV0: T/C0 溢出标志**

当 T/C0 溢出时，TOV0 置位。执行相应的中断服务程序时此位硬件清零。此外，TOV0 也可以通过写 1 来清零。当 SREG 中的位 I、TOIE0(T/C0 溢出中断使能) 和 TOV0 都置位时，中断服务程序得到执行。

- **Bit 0 – Res: 保留位**

在 ATtiny26(L) 中该位为保留位，读返回值为 ”0” 。

外部中断

外部中断通过引脚 INT0 触发。如果使能了中断，即使引脚 INT0 配置为输出，只要电平发生了合适的变化，中断也会触发。这个特点可以用来产生软件中断。通过设置 MCU 控制寄存器 MCUCR，中断可以由下降沿、上升沿、引脚变化或者是低电平触发。当外部中断使能并且配置为电平触发，只要引脚电平为低，中断就会产生。

引脚电平变化中断

若中断使能且引脚第二功能没有被中断屏蔽，引脚电平变化中断是通过端口 B 与引脚 PA3、PA6 及 PA7 的 I/O 引脚变化触发的。寄存器 GIMSK 的 PCIE1 位使能引脚 PB[7:4]、PA[7:6] 与 PA[3] 中断。；PCIE0 位使能 PB[3:0] 中断。

引脚中断变化与其他中断有两点不同。第一，若中断标志位没有设置，则引脚电平变化中断使能位 PCIE1 与 PCIE0 会屏蔽标志位。而在大多数中断操作中，标志位始终激活，只有在执行中断服务时由中断使能将其屏蔽。

第二，请注意当引脚使用其第二功能时，引脚电平变化中断禁用。例如，当引脚为 AREF、AIN0 或 AIN1、OC1A、$\overline{OC1A}$、OC1B、$\overline{OC1B}$、XTAL1 或 XTAL2，Timer0 定时或 $\overline{RESET}$ 功能时，不会产生引脚电平变化中断。第二功能屏蔽那些引脚电平变化中断及如何使能，请参见 Table 17。例如，在 USI 双线模式、USI 三线模式或 T/C1 反向输出比较使能时 PB0 的引脚变化使能被禁用。

如果中断使能，即使变化引脚设为输出，该中断也会被触发。这个特点可以用来产生软件中断。同样，即使引脚触发其他中断，例如外部中断，也可同时触发引脚电平变化中断。即一个外部事件可能引起多个中断。

熔丝位编程，其值为 “0”；未编程，其值为 “1”。每栏中各功能为 “ 或” 的关系。

Table 17. 第二功能

Pin	第二功能	设置第二功能的控制寄存器 [位名称] (1)	位或熔丝位值 (2)
PA3	AREF	ADMUX[REFS0]	1
PA6	模拟比较器	ACSR[ACD]	0
PA7	模拟比较器	ACSR[ACD]	0
PB0	USI 两线模式 USI 三线模式 TC1 比较 /PWM	USICR[USIWM1] USICR[USIWM1,USIWM0] TCCR1A[COM1A1,COM1A0,PWM1A]	1 01 011
PB1	USI 三线模式 TC1 比较 /PWM	USICR[USIWM1,USIWM0] TCCR1A[COM1A1] TCCR1A[COM1A0]	01 1 1
PB2	USI 两线模式 USI 三线模式 TC1 比较 /PWM	USICR[USIWM1] USICR[USIWM1,USIWM0] TCCR1A[COM1B1,COM1B0,PWM1B]	1 01 011
PB3	TC1 比较 /PWM	TCCR1A[COM1B1] TCCR1A[COM1B0]	1 1
PB4	XTAL1，时钟源	FUSE[PLLCK,CKSEL] FUSE[PLLCK,CKSEL]	10000 10101-11111
PB5	XTAL2，时钟源	FUSE[PLLCK,CKSEL]	11001-11111
PB6	外部中断 TC0 时钟	GIMSK[INT0],MCUCR[ISC01,ISC01] TCCR0[CS02,CS01]	100 11
PB7	RESET	RSTDISBL FUSE	1

Notes: 1. 每行表示一位或熔丝位使能该功能。
2. 熔丝位值为 “0” 表示已编程；为 “1” 表示未编程。

MCU 控制寄存器 - MCUCR

MCU 控制寄存器包含通用 MCU 功能控制位。

Bit	7	6	5	4	3	2	1	0	
$35 ($55)	–	PUD	SE	SM1	SM0	–	ISC01	ISC00	MCUCR
读 / 写	R	R/W	R/W	R/W	R/W	R	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

- **Bits 7 – Res: 保留位**

在 ATtiny26(L) 中该位为保留位，读返回值为 ”0” 。

- **Bit 6 – PUD: 上拉禁用**

当该位置位，即使 DDxn 与 PORTxn 设置为使能上拉电阻 ({DDxn, PORTxn} = 0b01)，I/O 端口的上拉也会被禁用，详见 P88“ 配置引脚 ” 。

- **Bit 5 – SE: 休眠使能**

当 SLEEP 指令执行时， SE 位必须置位，以使 MCU 进入休眠模式。为避免误操作，建议在执行 SLEEP 指令前再设置 SE 位。

- **Bits 4,3 – SM1/SM0: 休眠模式选择位 1 和 0**

如下表所示，这两位在四种休眠模式中做出选择。

Table 18. 休眠模式

SM1	SM0	休眠模式
0	0	空闲模式
0	1	ADC 噪声抑制模式
1	0	掉电模式
1	1	Standby 模式

详见 “ 休眠模式 ” 部分。

- **Bit 2 – Res: 保留位**

在 ATtiny26(L) 中该位为保留位，读返回值为 ”0” 。

- **Bits 1, 0 – ISC01, ISC00: 中断触发方式控制 0 Bit 1 and Bit 0**

外部中断 0 由引脚 INT0 激发，如果 SREG 寄存器的 I 标志位和相应的中断屏蔽位置位的话。触发方式如下表所示。

Table 19. 中断 0 触发方式控制 (1)

ISC01	ISC00	说明
0	0	INT0 为低电平时产生中断请求 .
0	1	INT0 引脚上任意的逻辑电平变化都将引发中断
1	0	INT0 的下降沿产生异步中断请求
1	1	INT0 的上升沿产生异步中断请求

Note: 1. 改变 ISC10/ISC00 时，必须先在寄存器 GIMSK 中清除 INT0 的中断使能位来将其禁用。否则当该位变化时会出现中断。

电源管理及睡眠模式

睡眠模式可以使应用程序关闭 MCU 中没有使用的模块，从而降低功耗。 AVR 具有不同的睡眠模式，允许用户根据自己的应用要求实施剪裁。

进入睡眠模式的条件是置位寄存器 MCUCR 的 SE，然后执行 SLEEP 指令。具体哪一种模式 (空闲模式、ADC 噪声抑制模式、掉电模式、省电模式、Standby 模式和扩展 Standby 模式) 由 MCUCR 的 SM1 和 SM0 决定，如 Table 18 所示。使能的中断可以将进入睡眠模式的 MCU 唤醒。经过启动时间，外加 4 个时钟周期后，MCU 就可以运行中断例程了。然后返回到 SLEEP 的下一条指令。唤醒时不会改变寄存器文件和 SRAM 的内容。如果在睡眠过程中发生了复位，则 MCU 唤醒后从中断向量开始执行。

P39Table 20 介绍了ATtiny26不同的时钟系统及其分布。此图在选择合适的睡眠模式时非常有用。

空闲模式

当 SM1..0 为 “00” 时，SLEEP 指令将使 MCU 进入空闲模式。在此模式下，CPU 停止运行，而模拟比较器、 ADC、 USI、 T/C、看门狗和中断系统继续工作。这个休眠模式只停止了 clk_{CPU} 和 clk_{FLASH}，其他时钟则继续工作。

象定时器溢出与 USI 启动与溢出等内外部中断都可以唤醒 MCU。如果不需要从模拟比较器中断唤醒 MCU，为了减少功耗，可以切断比较器的电源。方法是置位模拟比较器控制和状态寄存器 ACSR 的 ACD。如果 ADC 使能，进入此模式后将自动启动一次转换。

ADC 噪声抑制模式

当 SM1..0 为 “01” 时， SLEEP 指令将使 MCU 进入 ADC 噪声抑制模式。在此模式下，CPU 停止运行，而 ADC、外部中断、USI 启动状态检测和看门狗继续工作。这个睡眠模式只停止了 $clk_{I/O}$、 clk_{CPU} 和 clk_{FLASH}，其他时钟则继续工作。

此模式提高了 ADC 的噪声环境，使得转换精度更高。 ADC 使能的时候，进入此模式将自动启动一次 AD 转换。 ADC 转换结束中断、外部复位、看门狗复位、 BOD 复位、 USI 启动状态中断、 EEPROM 准备好中断、外部中断 INT0 ，或引脚电平变化中断可以将 MCU 从 ADC 噪声抑制模式唤醒。

掉电模式

当 SM1..0 为 “10” 时， SLEEP 指令将使 MCU 进入掉电模式。在此模式下，外部晶体停振，而外部中断、 USI 启动状态检测及看门狗 (如果使能的话) 继续工作。只有外部复位、看门狗复位、 BOD 复位、 USI 启动状态中断、外部电平中断 INT0 ，或引脚电平变化中断可以使 MCU 脱离掉电模式。这个睡眠模式停止了所有的时钟，只有异步模块可以继续工作。

从施加掉电唤醒条件到真正唤醒有一个延迟时间，此时间用于时钟重新启动并稳定下来。唤醒周期与由熔丝位 CKSEL 定义的复位周期是一样的，如 P26“ 时钟源 ” 所示。

注意，当使用外部电平中断方式将 MCU 从掉电模式唤醒时，必须保持外部电平一定的时间，以降低 MCU 对噪声的敏感性。看门狗振荡器时钟对变化电平进行两次采样，若达到要求电平， MCU 被唤醒。在 3.0V 与 25°C 时看门狗振荡器周期为 1.0 µs (标称值)。看门狗振荡器频率请参见电气特性部分。

若在 MCU 唤醒及启动执行前唤醒条件结束，即 INT0 低电平保持时间不够，中断引发的唤醒将不会执行。

Standby 模式

当 SM1..0 为 “11” 且选择外部晶振或谐振器时钟时 ,SLEEP 指令将使 MCU 进入 Standby 模式。这一模式与掉电模式唯一的不同之处在于振荡器继续工作。其唤醒时间只需要 6 个时钟周期。

Table 20. 在不同睡眠模式下活动的时钟以及唤醒源

	工作的时钟					振荡器		唤醒源		
睡眠模式	clk_{CPU}	clk_{FLASH}	clk_{IO}	clk_{ADC}	使能的主时钟	INT0，与引脚变化	USI 启动状态	EEPROM 准备好	ADC	其他 I/O
空闲模式			X	X	X	X	X	X	X	X
ADC 噪声抑制模式				X	X	X(2)	X	X	X	
掉电模式						X(2)	X			
Standby(1)					X	X(2)	X			

Notes: 1. 时钟源为外部晶体或谐振器
2. 电平中断 INT0

最小化功耗

试图降低 AVR 控制系统的功耗时需要考虑几个问题。一般来说，要尽可能利用睡眠模式，并且使尽可能少的模块继续工作。不需要的功能必须禁止。下面的模块需要特殊考虑以达到尽可能低的功耗。

模数转换器

使能时， ADC 在睡眠模式下继续工作。为了降低功耗，在进入睡眠模式之前需要禁止 ADC。重新启动后的第一次转换为扩展的转换。具体请参照 P75“ 模数转换器 ”。

模拟比较器

在空闲模式时，如果没有使用模拟比较器，可以将其关闭。在 ADC 噪声抑制模式下也是如此。在其他睡眠模式模拟比较器是自动关闭的。如果模拟比较器使用了内部电压基准源，则不论在什么睡眠模式下都需要关闭它。否则内部电压基准源将一直使能。请参见 P72“ 模拟比较器 ” 以了解如何配置模拟比较器。

掉电检测 BOD

如果系统没有利用掉电检测器 BOD，这个模块也可以关闭。如果熔丝位 BODEN 被编程，从而使能了 BOD 功能，它将在各种休眠模式下继续工作。在深层次的休眠模式下，这个电流将占总电流的很大比重。请参看 P23“ 掉电检测复位 ” 以了解如何配置 BOD。

片内基准电压

使用 BOD、模拟比较器和 ADC 时可能需要内部电压基准源。若这些模块都禁止了，则基准源也可以禁止。重新使能后用户必须等待基准源稳定之后才可以使用它。如果基准源在休眠过程中是使能的，其输出立即可以使用。

看门狗定时器

如果系统无需利用看门狗，这个模块也可以关闭。若使能，则在任何休眠模式下都持续工作，从而消耗电流。在深层次的睡眠模式下，这个电流将占总电流的很大比重。请参看 P56“ 看门狗定时器 ” 以了解如何配置看门狗定时器。

端口引脚

进入休眠模式时，所有的端口引脚都应该配置为只消耗最小的功耗。最重要的是避免驱动电阻性负载。在休眠模式下 I/O 时钟 $clk_{I/O}$ 和 ADC 时钟 clk_{ADC} 都被停止了，输入缓冲器也禁止了，从而保证输入电路不会消耗电流。在某些情况下输入逻辑是使能的，用来检测唤醒条件。用于此功能的具体引脚请参见 P91“ 数字输入使能和休眠模式 ”。如果输入缓冲器是使能的，此时输入不能悬空，信号电平也不应该接近 $V_{CC}/2$，否则输入缓冲器会消耗额外的电流。

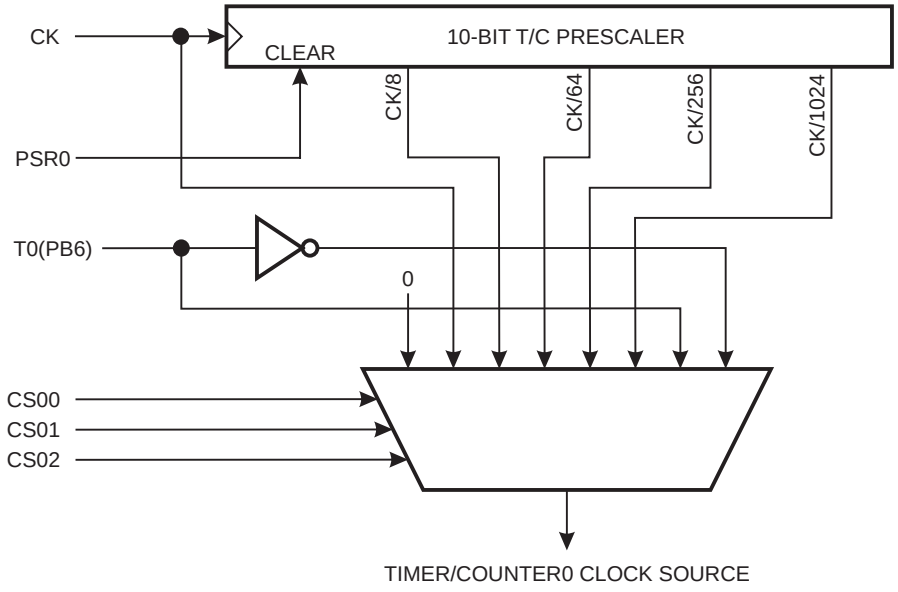
定时器 / 计时器

ATtiny26(L) 有两个通用功能8位T/C。这些T/C有独立的预分频器。T/C0时钟(CK)作为时基。 T/C1 有两个时钟模式，同步模式与异步模式。同步模式使用 CK 作为时基而异步模式使用快速外设时钟 (PCK) 作为时基。

T/C0 预分频器

Figure31 给出了 T/C 预分频器。

Figure 31. T/C0 预分频器

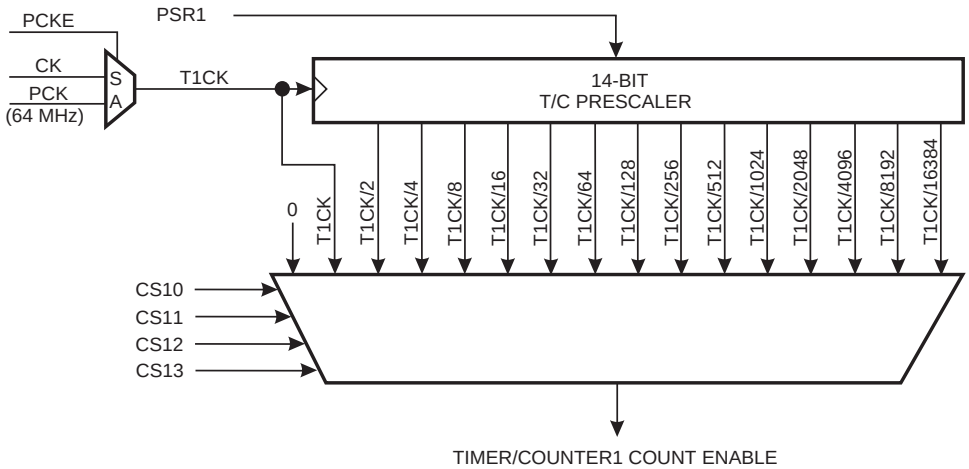


四个预分频部分为：CK/8、 CK/64、 CK/256 与 CK/1024， CK 为振荡器时钟。 CK、外部源与停止可选作时钟源。

T/C1 预分频器

Figure32 给出了T/C1预分频器。对T/C1 ,异步模式下 ,时钟在PCK到PCK/16384间选择，而同步模式下，时钟在 CK 到 CK/16384 间选择。时钟选项及 T/C1 控制寄存器 TCCR1B 在 P47Table 24 中给出。设置 TCCR1B 寄存器的 PSR1 位复位预分频器。PLLCSR 寄存器的 PCKE 位使能异步模式。

Figure 32. T/C1 预分频器



8 位 T/C0

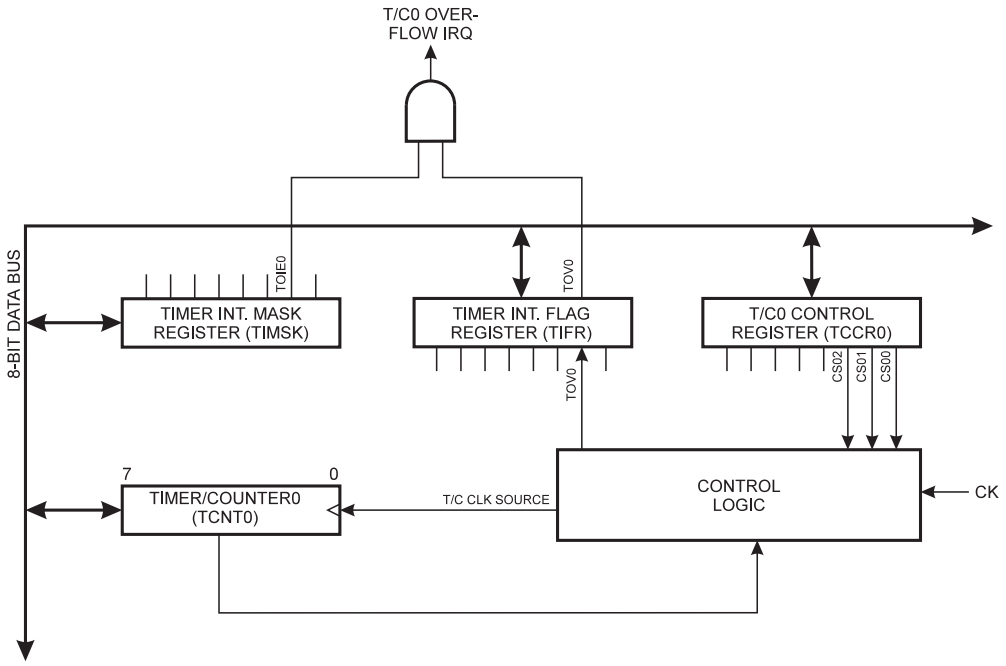
Figure33 给出 T/C0 方框图。

8 位 T/C0 可从 CK、预分频 CK 及外部引脚选择时钟源。另外，它还可如 T/C0 控制寄存器 TCCR0 中说明的那样终止。溢出状态标志在 T/C 中断标志寄存器 TIFR 中。控制信号在 T/C0控制寄存器TCCR0中。T/C0的中断使能/禁用设置在T/C中断屏蔽寄存器TIMSK中。

当 T/C0 是外部时钟定时，外部信号同 CPU 振荡频率同步。为保证对外部时钟正确的采样，两个外部时钟转换的最小时间间隔不小于一个内部 CPU 时钟周期。外部时钟信号在内部 CPU 时钟的上升沿采样。

8 位 T/C0 的特性是使用低预分频因子得到高分辨率与高精度。类似的，高预分频因子使 T/C0 适用于低速功能或精确时序操作。

Figure 33. T/C0 方框图



T/C0 控制寄存器 - TCCR0

Bit	7	6	5	4	3	2	1	0	
$33 ($53)	–	–	–	–	PSR0	CS02	CS01	CS00	TCCR0
读 / 写	R	R	R	R	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

- **Bits 7..4 – Res: 保留位**

在 ATtiny26(L) 中这几位为保留位，读返回值为 ”0” 。

- **Bit 3 – PSR0: T/C0 预分频器复位**

置位时 T/C0 的预分频器复位。操作完成后这一位由硬件自动清零。写入零时不会引发任何动作。该位总是读为 0。

- **Bits 2, 1, 0 – CS02, CS01, CS00: 时钟选择 0, Bit 2, 1 与 0**

用于选择 T0 预分频源。

Table 21. 时钟 0 选择位定义

CS02	CS01	CS00	说明
0	0	0	Stop, 无时钟， T/C0 不工作
0	0	1	CK
0	1	0	CK/8
0	1	1	CK/64
1	0	0	CK/256
1	0	1	CK/1024
1	1	0	时钟由 T0 引脚输入，下降沿触发
1	1	1	时钟由 T0 引脚输入，上升沿触发

Stop 状态提供了定时器使能 / 禁用功能。如果使用外部引脚模式，必须对相应的实际数据方向控制寄存器进行设置 (清零以设置为输入)。

T/C0 - TCNT0

Bit	7	6	5	4	3	2	1	0	
$32 ($52)	MSB							LSB	TCNT0
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

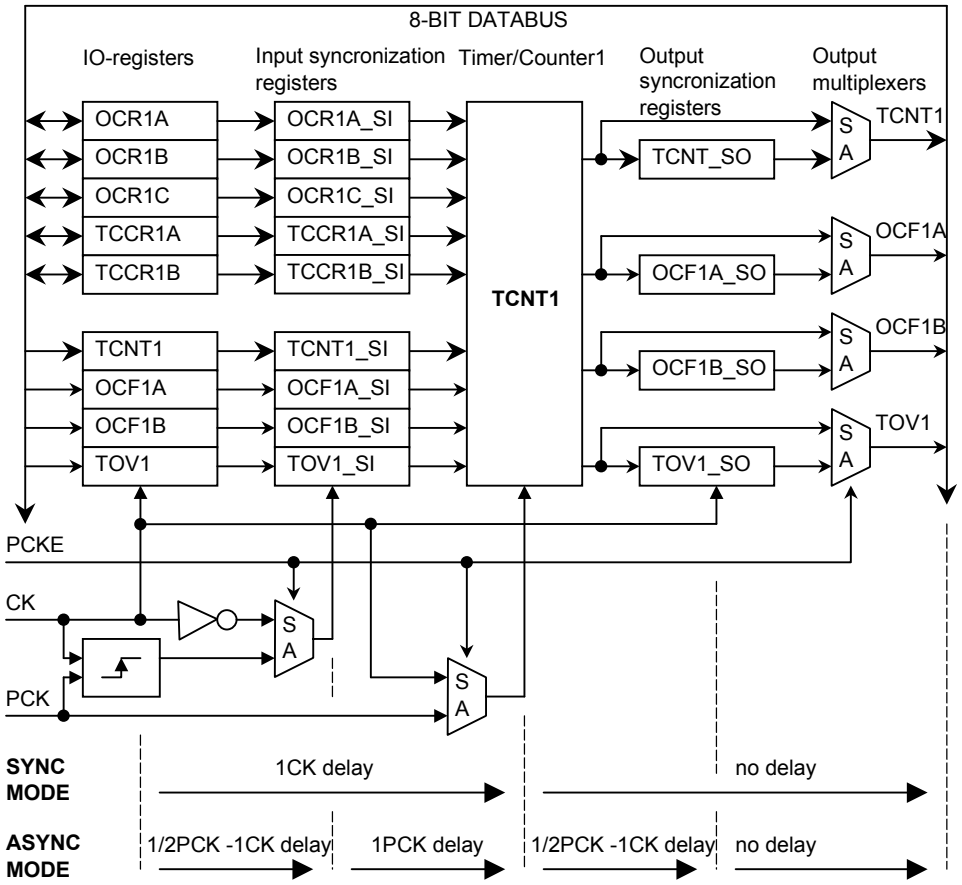
T/C0 为可读写访问的向上计数寄存器。若时钟源存在，则在写 T/C0 操作结束后，T/C0 继续进行计数。

8 位 T/C1

T/C1 有两个时钟模式 : 同步模式与异步模式。同步模式使用系统时钟 (CK) 作为时基，而异步模式使用快速外设时钟 (PCK) 作为时基。当 PLLCSR 寄存器的 PCKE 为 “1” 时，异步模式使能。 T/C1 通用操作在异步模式下加以说明，至于同步模式只有在与异步模式出现不同情况时才加以说明。Figure34 给出 T/C1 同步寄存器框图及寄存器间的同步延迟。注意，图中未给出详细的时钟选通控制。 T/C1 寄存器值在影响计数器操作前会通过内部同步寄存器，而这会造成输入同步延迟。对寄存器 TCCR1A、 TCCR1B、 OCR1A、 OCR1B 及 OCR1C 完成写操作后可立即进行读操作。 T/C1 (TCNT1) 寄存器及标志 (OCF1A、 OCF1B 与 TOV1) 的读出值由于输入输出同步而延迟。

该模块可由低预分频因子得到高分辨率及高精度。 T/C1 支持两种精度，高速 8 位脉宽调制使用的时钟速度可以高达 64 MHz。该模式下， T/C1 与输出比较寄存器作为无溢出、无重叠的不反向 / 反向输出的双单路 PWM 使用。详见 P52 的说明。类似的，高预分频因子使 T/C1 适用于低速功能或精确时序操作。

Figure 34. T/C1 同步寄存器方框图

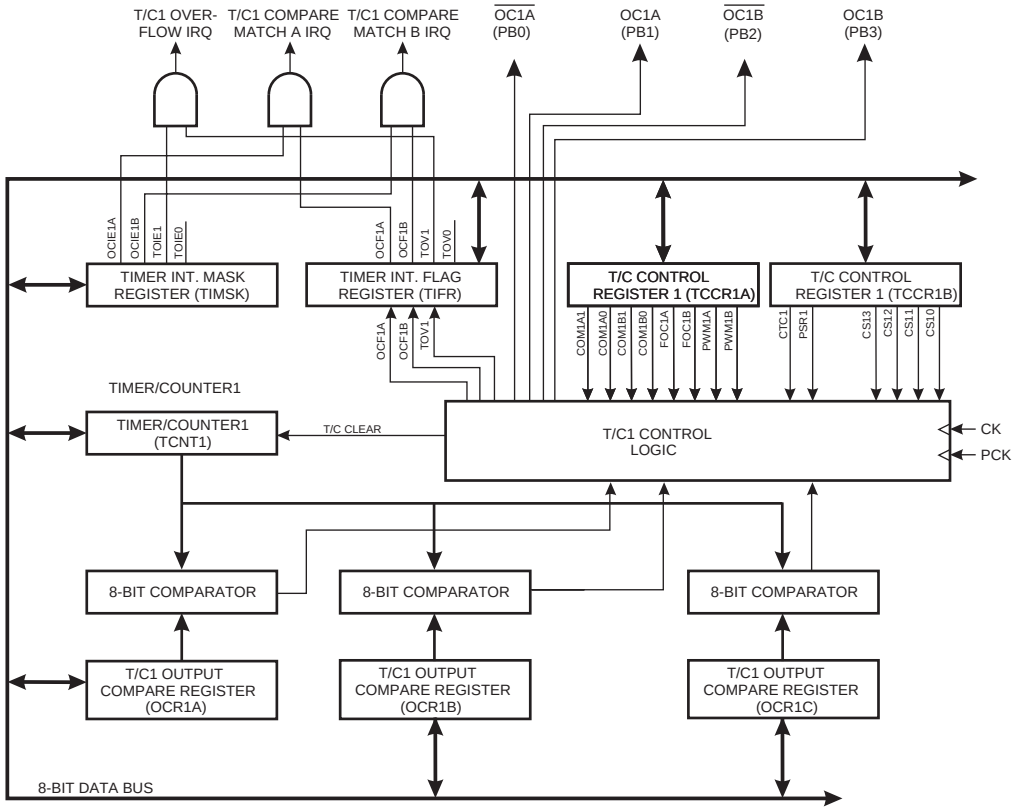


当预分频器工作在异步模式下的快速 64 MHz PCK 时钟时，T/C1 与预分频器可使用所有的时钟源。

注意，系统时钟频率必须低于 PCK 频率的一半。只有当系统时钟由 PCK 二分频产生时，PCK与系统时钟的比值才刚好为2。当系统时钟为高时，异步T/C1的同步机制至少需要两个 PCK 边沿。如果系统时钟频率太高，可能会丢失数据或控制值。

Figure35 给出 T/C1 方框图。

Figure 35. T/C1 方框图



T/C中断标志寄存器TIFR中有三个状态标志 (溢出与比较匹配)。T/C控制寄存器TCCR1A与 TCCR1B 中有控制信号。T/C 中断屏蔽寄存器 TIMSK 中有中断使能 / 禁用设置。

T/C1包含三个输出比较寄存器OCR1A、OCR1B与OCR1C ,作为数据源与T/C1中内容进行比较。在普通模式下，输出比较功能操作使用三个输出比较寄存器。OCR1A 确定在OC1A引脚(PB1)操作 ,且其可在普通模式与 PWM模式下产生定时器1 OC1A中断。类似的， OCR1B 确定在 OC1B 引脚 (PB3) 操作，且其可在普通模式与 PWM 模式下产生定时器 1 OC1B 中断。在普通或 PWM 模式下，OCR1C 保持 T/C 最大值 ,即 ,清除比较匹配值。当T/C1从$FF计数到$00或从OCR1C计数到$00就会产生一个溢出中断(TOV1)。在普通模式下，反向 PWM 输出 $\overline{\text{OC1A}}$ 与 $\overline{\text{OC1B}}$ 未连接。

PWM 模式下 ,OCR1A 与 OCR1B 提供与 T/C 的比较值。比较匹配发生时产生 PWM 输出 (OC1A、$\overline{\text{OC1A}}$、OC1B、$\overline{\text{OC1B}}$)。PWM 模式下，T/C 计数达到 OCR1C 给出的值后 ,再重新从 $00 开始计数。利用该特性，可使计数器“ 满” 值低于 $FF。PWM 频率选择与其他预分频选项一起给出。 Table 27 列出获得从 20 kHz 到 250 kHz，每级 10 kHz 及从 250 kHz 到500 kHz ，每级 50 kHzPWM频率时的时钟选择与OCR1C值。更高的PWM频率可通过降低分辨率获得。

T/C1 控制寄存器 A - TCCR1A

Bit	7	6	5	4	3	2	1	0	
$30 ($50)	COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	PWM1A	PWM1B	TCCR1A
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

- **Bits 7, 6 – COM1A1, COM1A0: 比较 A 输出模式 , Bits 1 与 0**

COM1A1 与 COM1A0 位决定了比较匹配发生时输出引脚 OC1A 的电平。同时其方向控制位要设置为 1 以使能输出引脚。注意再普通模式下 $\overline{OC1A}$ 未连接。

Table 22. 比较器 A 模式选择

COM1A1	COM1A0	说明
0	0	T/C 比较 A 不与 OC1A 相连接
0	1	比较匹配发生时 OC1A 取反
1	0	比较匹配发生时 OC1A 清零
1	1	比较匹配发生时 OC1A 置位

PWM 模式下，这些位有不同的功能，参见 P50Table 25。

- **Bits 5, 4 – COM1B1, COM1B0: 比较 B 输出模式 , Bits 1 与 0**

COM1B1 与 COM1B0 位决定了比较匹配发生时输出引脚 OC1B 的电平。同时其方向控制位要设置为 1 以使能输出引脚。注意再普通模式下 $\overline{OC1B}$ 未连接。

Table 23. 比较器 B 模式选择

COM1B1	COM1B0	说明
0	0	T/C 比较 B 不与 OC1B 相连接
0	1	比较匹配发生时 OC1B 取反
1	0	比较匹配发生时 OC1B 清零
1	1	比较匹配发生时 OC1B 置位

PWM 模式下，这些位有不同的功能，参见 P50Table 25。

- **Bit 3 – FOC1A: 强制输出比较匹配 1A**

对其写 1 后，波形发生器将立即进行比较操作。比较匹配输出引脚 OC1A 将按照 COM1A1:0 的设置输出相应的电平。如果 COM1A1、COM1A0 与 FOC1A 在一个周期中写入，新设置可立即使用。无论定时器值是多少，强制输出比较位可用来改变输出引脚值。如果比较匹配出现， COM1A1、 COM1A0 将自动执行程序，但不会产生中断。 FOC1A 位总是为零，若 PWM1A 位不设置， FOC1A 无效。

- **Bit 2 – FOC1B: 强制输出比较匹配 1B**

对其写 1 后，波形发生器将立即进行比较操作。比较匹配输出引脚 OC1B 将按照 COM1B1:0 的设置输出相应的电平。如果 COM1B1、COM1B0 与 FOC1B 在一个周期中写入，新设置可立即使用。无论定时器值是多少，强制输出比较位可用来改变输出引脚值。如果比较匹配出现， COM1B1、 COM1B0 将自动执行程序，但不会产生中断。 FOC1B 位总是为零，若 PWM1B 位不设置， FOC1B 无效。

- **Bit 1 – PWM1A: 脉宽调制 A 使能**

当该位置位，使能 PWM 模式。此时 T/C1 的比较器 OCR1A 与计数器值在与 OCR1C 寄存器值比较匹配后复位为 $00。

- **Bit 0 – PWM1B: 脉宽调制 B 使能**

当该位置位，使能 PWM 模式。此时 T/C1 的比较器 OCR1B 与计数器值在与 OCR1C 寄存器值比较匹配后复位为 $00。

T/C1 控制寄存器 B - TCCR1B

Bit	7	6	5	4	3	2	1	0	
$2F ($4F)	CTC1	PSR1	–	–	CS13	CS12	CS11	CS10	TCCR1B
读 / 写	R/W	R/W	R	R	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

- **Bit 7 – CTC1: 比较匹配时清除 T/C**

当 CTC1 控制位置位，在与 OCR1C 寄存器值比较匹配后 T/C1 复位到 $00。如果控制位被清零， T/C1 继续计数，且不受比较匹配影响。

- **Bit 6 – PSR1: 预分频复位 T/C1**

置位时 T/C 的预分频器复位。操作完成后这一位由硬件自动清零。写入零时不会引发任何动作。该位总是读为 0。

- **Bit 5..4 – Res: 保留位**

在 ATtiny26(L) 中这几位为保留位，读返回值为 ”0” 。

- **Bits 3..0 – CS13, CS12, CS11, CS10: 时钟选择位 3, 2, 1 与 0**

时钟选择位 3、2、 1 和 0 定义 T/C1 的预分频源。

Table 24. T/C1 预分频选择

CS13	CS12	CS11	CS10	说明异步模式	说明同步模式
0	0	0	0	T/C1 停止	T/C1 停止
0	0	0	1	PCK	CK
0	0	1	0	PCK/2	CK/2
0	0	1	1	PCK/4	CK/4
0	1	0	0	PCK/8	CK/8
0	1	0	1	PCK/16	CK/16
0	1	1	0	PCK/32	CK/32
0	1	1	1	PCK/64	CK/64
1	0	0	0	PCK/128	CK/128
1	0	0	1	PCK/256	CK/256
1	0	1	0	PCK/512	CK/512
1	0	1	1	PCK/1024	CK/1024
1	1	0	0	PCK/2048	CK/2048
1	1	0	1	PCK/4096	CK/4096
1	1	1	0	PCK/8192	CK/8192
1	1	1	1	PCK/16384	CK/16384

停止状态提供定时器使能 / 禁止功能。

T/C1 - TCNT1

Bit	7	6	5	4	3	2	1	0	
$2E ($4E)	MSB							LSB	TCNT1
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

初始值	0	0	0	0	0	0	0	0

这 8 位寄存器包含 T/C1 的值。

T/C1为可读写访问的向上计数器。由于与CPU同步，在同步模式下T/C1数据写入T/C1延迟一个 CPU 时钟周期；在异步模式下，最多延迟两个 CPU 时钟周期。

T/C1 输出比较寄存器 - OCR1A

Bit	7	6	5	4	3	2	1	0	
$2D ($4D)	MSB							LSB	OCR1A
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

输出比较寄存器 A 为 8 位读 / 写寄存器。

T/C 输出比较寄存器A中的数据不断与T/C1比较。比较匹配操作在TCCR1A中给出说明。只有在 T/C1 计数到 OCR1A 值时才会出现比较匹配。用软件对 TCNT1 与 OCR1A 写入相同值不好产生比较匹配。

在比较事件同步延迟后，比较匹配会设置比较中断标志 OCF1A。

T/C1 输出比较寄存器 B - OCR1B

Bit	7	6	5	4	3	2	1	0	
$2C ($4C)	MSB							LSB	OCR1B
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

输出比较寄存器 B 为 8 位读 / 写寄存器。

T/C 输出比较寄存器B中的数据不断与T/C1比较。比较匹配操作在TCCR1A中给出说明。只有在 T/C1 计数到 OCR1B 值时才会出现比较匹配。用软件对 TCNT1 与 OCR1B 写入相同值不好产生比较匹配。

在比较事件同步延迟后，比较匹配会设置比较中断标志 OCF1B。

T/C1 输出比较寄存器 C - OCR1C

Bit	7	6	5	4	3	2	1	0	
$2B ($4B)	MSB							LSB	OCR1C
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

输出比较寄存器 C 为 8 位读 / 写寄存器。

T/C输出比较寄存器C中的数据不断与T/C1比较。只有在T/C1计数到OCR1C值时才会出现比较匹配。用软件对 TCNT1 与 OCR1C 写入相同值不好产生比较匹配。

如果设置寄存器 TCCR1B 的 CTC1 位，比较匹配将清除 TCNT1 且设置溢出中断标志 (TOV1)。在比较事件同步延迟后，比较匹配会设置比较中断标志

该寄存器在普通模式和 PWM 模式下功能相同。

PLL 控制与状态寄存器 - PLLCSR

Bit	7	6	5	4	3	2	1	0	
$29 ($29)	–	–	–	–	–	PCKE	PLLE	PLOCK	PLLCSR
读 / 写	R	R	R	R	R	R/W	R/W	R	
初始值	0	0	0	0	0	0	0/1	0	

- **Bit 7..3 – Res: 保留位**

在 ATtiny26(L) 中这几位为保留位，读返回值为 ”0” 。

- **Bit 2 – PCKE: PCK 使能**

PCKE位改变T/C1时钟源。该位置位时异步时钟模式使能且使用快速64 MHz PCK时钟作为 T/C1 时钟源。如果该位清零，同步时钟模式使能，使用系统时钟 CK 作为 T/C1 时钟源。只有在 PLLE 位设置后该位才能设置。只有当 PLL 锁定，即 PLOCK 位为 1，才能安全设置该位。

- **Bit 1 – PLLE: PLL 使能**

当 PLLE 置位， PLL 启动，且如果需要内部 RC 振荡器作为 PLL 参考时钟启动。若 PLL 选作系统时钟源，则该位值始终为 1。

- **Bit 0 – PLOCK: PLL 锁定探测器**

PLOCK置位表示PLL已经锁定参考时钟，可以安全地使能T/C1的 PCK。PLL使能后，PLL的锁定时间大致为 64 µs/100 µs (典型值 / 最差值)。

异步模式下的 T/C1 初始化

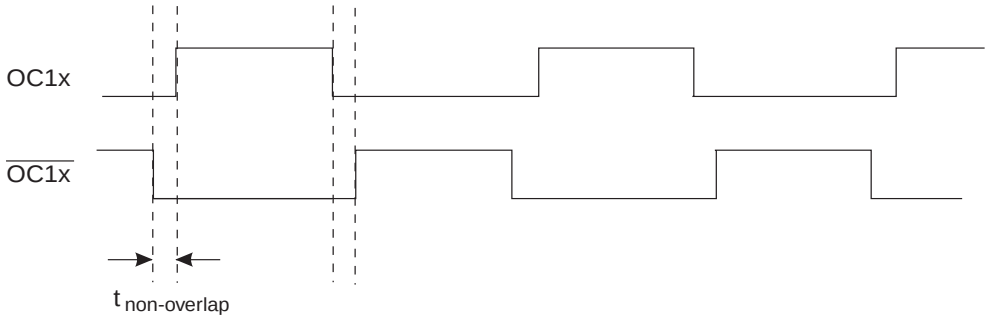
改变 T/C1为异步模式，首先使能PLL，然后查询PLOCK直至其置位，接着设置PCKE位。

PWM 模式下的 T/C1

选择 PWM 模式时， T/C1 与输出比较寄存器 C – OCR1C 组成双 8 位，自由振荡无毛刺 PWM 产生器， PB1(OC1A) 与 PB3(OC1B) 作为输出引脚。相反无重叠输出引脚为 PB0($\overline{OC1A}$)与PB2($\overline{OC1B}$)。无重叠输出对(OC1A - $\overline{OC1A}$与OC1B - $\overline{OC1B}$)从来不同时设置。这就允许直接驱动功率开关。无重叠时间为一个预分频时钟周期，且高电平时间比低电平时间短一个周期。

无重叠时间通过延迟上升沿得到，即在异步模式下上升沿为一个预分频周期及一个 PCK 周期延迟而下降沿为一个 PCK 周期延迟；在同步模式下上升沿为一个预分频周期及一个CK周期延迟而下降沿为一个CK周期延迟。在两种模式下高电平时间比低电平时间短一个预分频周期。

Figure 36. 无重叠输出对



x = A or B

当计数器值达到 OCR1A 与 OCR1B 时，根据 T/C1 控制寄存器 A – TCCR1A 中 COM1A1/COM1A0或COM1B1/COM1B0位的设置，对OC1A与OC1B输出设置或清零，参见 Table 25。

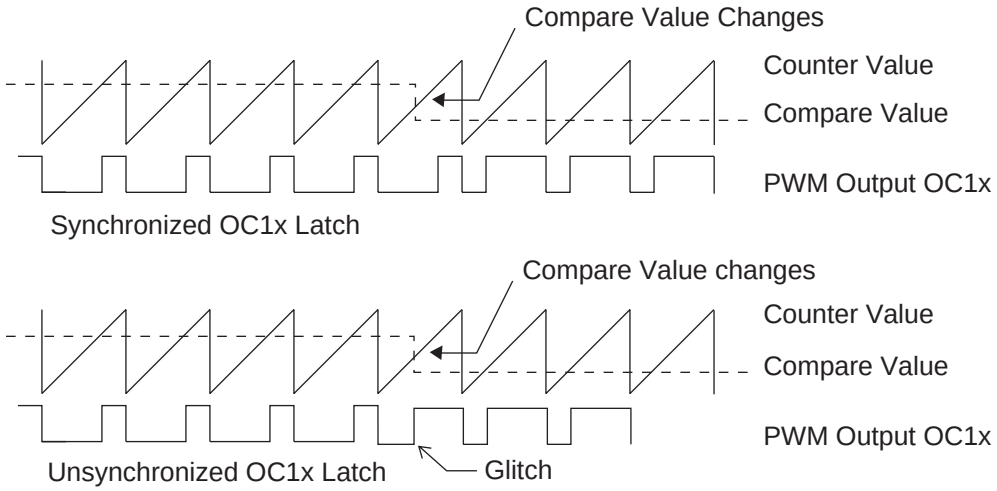
T/C1 作为向上计数器，从 $00 开始计数到 OCR1C 中的值，再回到 $00 重新开始。在与 OC1C 比较匹配完的同步延迟后，设置溢出中断标志 TOV1。

Table 25. PWM 模式下的比较模式选择

COM1x1	COM1x0	输出比较引脚的影响
0	0	OC1x 未连接 $\overline{OC1x}$ 未连接
0	1	比较匹配时 OC1x 清零。在 TCNT1 = $01 后设置一个预分频周期。 比较匹配后 $\overline{OC1x}$ 设置一个预分频周期。当 TCNT1 = $00 时清零。
1	0	比较匹配时 OC1x 清零，TCNT1 = $01 置位 $\overline{OC1x}$ 未连接
1	1	比较匹配后 OC1x 设置一个预分频周期。当 TCNT = $00 时清零。 $\overline{OC1x}$ 未连接

注意，在 PWM 模式下，对输出比较寄存器 OCR1A 或 OCR1B 写入时，数据值先传送到临时地址。当 T/C 达到 OCR1C 时，该值存入 OCR1A 或 OCR1B。这防止在非同步 OCR1A 或 OCR1B 事件中出现奇数时钟长度的 PWM 脉冲（毛刺），例子可参见 Figure 37。

Figure 37. 非同步 OCR 锁存结果



在写入与存储操作之间的时间里，对 OCR1A 或 OCR1B 的读操作将直接读取临时地址的内容。这意味着从 OCR1A 或 OCR1B 读出的总是最新写入的值。

当 OCR1A 或 OCR1B 为 $00 或最大值时，如在 OCR1C 寄存器中的说明，根据 COM1A1/COM1A0 的设置，输出 PB1(OC1A) 或 PB3(OC1B) 保持低或高，见 Table 26。

Table 26. PWM 输出 OCR1x = $00 或 OCR1C, x = A 或 B

COM1x1	COM1x0	OCR1x	输出 OC1x	输出 $\overline{OC1x}$
0	1	$00	L	H
0	1	OCR1C	H	L
1	0	$00	L	未连接
1	0	OCR1C	H	未连接
1	1	$00	H	未连接
1	1	OCR1C	L	未连接

PWM 模式 ,定时器溢出标志 – TOV1 如在普通 T/C 模式下一样设置。溢出中断操作与普通 T/C 模式下完全一样。即，当定时器溢出中断与全局中断使能，执行该操作。这同样适用于定时器输出比较标志与中断。

PWM 频率为定时器时钟 1 频率除以 (OCR1C 值 + 1)。公式如下：

$$f_{PWM} = \frac{f_{TCK1}}{(OCR1C + 1)}$$

分辨率给出表示 OCR1C 寄存器中的值所需的位数。公式如下：

$$Resolution_{PWM} = \log_2(OCR1C + 1)$$

Table 27. 异步模式下的 T/C1 时钟预分频选择

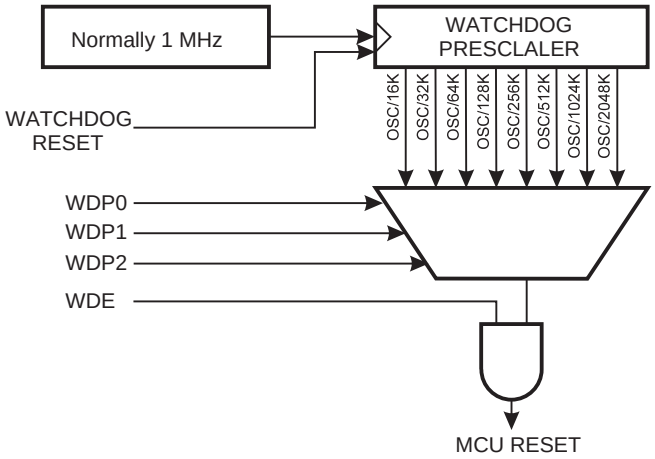
PWM 频率 (kHz)	时钟选择	CS13..CS10	OCR1C	分辨率 (位)
20	PCK/16	0101	199	7.6
30	PCK/16	0101	132	7.1
40	PCK/8	0100	199	7.6
50	PCK/8	0100	159	7.3
60	PCK/8	0100	132	7.1
70	PCK/4	0011	228	7.8
80	PCK/4	0011	199	7.6
90	PCK/4	0011	177	7.5
100	PCK/4	0011	159	7.3
110	PCK/4	0011	144	7.2
120	PCK/4	0011	132	7.1
130	PCK/2	0010	245	7.9
140	PCK/2	0010	228	7.8
150	PCK/2	0010	212	7.7
160	PCK/2	0010	199	7.6
170	PCK/2	0010	187	7.6
180	PCK/2	0010	177	7.5
190	PCK/2	0010	167	7.4
200	PCK/2	0010	159	7.3
250	PCK	0001	255	8.0
300	PCK	0001	212	7.7
350	PCK	0001	182	7.5
400	PCK	0001	159	7.3
450	PCK	0001	141	7.1
500	PCK	0001	127	7.0

看门狗定时器

看门狗定时器由独立的 1 Mhz 片内振荡器驱动。这是 V_{CC} = 5V 时的典型值。请参见特性数据以了解其他 V_{CC} 电平下的典型值。通过设置看门狗定时器的预分频器可以调节看门狗复位的时间间隔，从 16 到 2048 ms。看门狗复位指令 WDR 用来复位看门狗定时器。复位时间有 8 个选项。如果没有及时复位定时器，一旦时间超过复位周期，ATtiny26(L) 就复位，并执行复位向量指向的程序。具体的看门狗复位时序在 P23 有说明。

为了防止无意之间禁止看门狗定时器，当禁用看门狗时必须依据特别的关闭序列，详见看门狗定时器控制寄存器。

Figure 38. 看门狗定时器



看门狗定时器控制寄存器 - WDTCR

Bit	7	6	5	4	3	2	1	0	
$21 ($41)	–	–	–	WDCE	WDE	WDP2	WDP1	WDP0	WDTCR
读 / 写	R	R	R	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

- **Bits 7..5 – Res: 保留位**

在 ATtiny26(L) 中这几位为保留位，读返回值为 ”0” 。

- **Bit 4 – WDCE: 看门狗修改使能**

清零 WDE 时必须置位 WDCE，否则不能禁止看门狗。一旦置位，硬件将在紧接的 4 个时钟周期之后将其清零。请参考有关 WDE 的说明来禁止看门狗。在安全等级 1 与 2，修改预分频器也必须置位 WDCE。

- **Bit 3 – WDE: 看门狗使能**

WDE为"1“时，看门狗使能，否则看门狗将被禁止。只有在WDCE为"1“时WDE才能清零。以下为关闭看门狗的步骤：

1. 在同一个指令内对 WDCE 和 WDE 写 "1“，即使 WDE 已经为 "1“
2. 在紧接的 4 个时钟周期之内对 WDE 写 "0”，这会禁用看门狗。

- **Bits 2..0 – WDP2, WDP1, WDP0: 看门狗定时器预分频器 2, 1 和 0**

WDP2、 WDP1 和 WDP0 决定看门狗定时器的预分频器，如 Table 28 所示。

Table 28. 看门狗定时器预分频器选项 (1)

WDP2	WDP1	WDP0	看门狗振荡器周期	V_{CC} = 3.0V 时典型的溢出周期	V_{CC} = 5.0V 时典型的溢出周期
0	0	0	16K (16,384)	17.1 ms	16.3 ms
0	0	1	32K (32,768)	34.3 ms	32.5 ms
0	1	0	64K (65,536)	68.5 ms	65 ms
0	1	1	128K (131,072)	0.14 s	0.13 s
1	0	0	256K (262,144)	0.27 s	0.26 s
1	0	1	512K (524,288)	0.55 s	0.52 s
1	1	0	1,024K (1,048,576)	1.1 s	1.0 s
1	1	1	2,048K (2,097,152)	2.2 s	2.1 s

Note: 1. 看门狗振荡器频率与电压有关。WDR – 看门狗复位 – 指令应该在看门狗定时器使能前执行。这确保复位周期决定于看门狗定时器预分频设置。若看门狗使能前未复位，定时器可能不从 0 开始。

EEPROM 读 / 写 访问

EEPROM 的访问寄存器位于 I/O 空间。

EEPROM 的写访问典型时间为 8.3 ms。自定时功能可以让用户监测何时开始写下一字节。当 EEPROM 准备好接收新数据时，特定的 EEPROM 准备好中断设置为触发。

即使复位状态出现，进行中的 EEPROM 写操作也会执行完。

为了防止无意识的 EEPROM 写操作，在写 EEPROM 时需要执行一个特定的写时序。具体参看 EEPROM 控制寄存器的内容。

当执行 EEPROM 写操作时， CPU 会停止工作 2 个周期，然后再执行后续指令。

当执行 EEPROM 读操作时， CPU 会停止工作 4 个周期，然后再执行后续指令。

EEPROM 地址寄存器 - EEAR

Bit	7	6	5	4	3	2	1	0	
$1E ($3E)	–	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	EEAR
读 / 写	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	X	X	X	X	X	X	X	

- **Bit 7 – RES: 保留位**

在 ATtiny26(L) 中该位为保留位，读返回值为 ”0” 。

- **Bit 6..0 – EEAR6..0: EEPROM 地址**

EEAR 指定了 128 字节的 EEPROM 空间。EEPROM 地址是线性的，从 0 到 127。EEAR 的初始值没有定义。在访问 EEPROM 之前必须为其赋予正确的数据。

EEPROM 数据寄存器 - EEDR

Bit	7	6	5	4	3	2	1	0	
$1D ($3D)	MSB							LSB	EEDR
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

- **Bit 7..0 – EEDR7..0: EEPROM 数据**

对于 EEPROM 写操作， EEDR 是需要写到 EEAR 单元的数据；对于读操作， EEDR 是从地址 EEAR 读取的数据。

EEPROM 控制寄存器 - EECR

Bit	7	6	5	4	3	2	1	0	
$1C ($3C)	–	–	–	–	EERIE	EEMWE	EEWE	EERE	EECR
读 / 写	R	R	R	R	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

- **Bit 7..4 – RES: 保留位**

在 ATtiny26(L) 中这几位为保留位，读返回值为 ”0” 。

- **Bit 3 – EERIE: EEPROM 准备好使能**

若 SREG 的 I 为 "1"，则置位 EERIE 使能 EEPROM 准备好中断。清零 EERIE 则禁止此中断。当 EEWE 清零时 EEPROM 准备好中断即可发生。

- **Bit 2 – EEMWE: EEPROM 写使能**

EEMWE决定设置EEWE为"1“是否可以启动EEPROM写操作。当EEMWE为"1“时，在4个时钟周期内置位EEWE将把数据写入EEPROM的指定地址；若EEMWE为"0“，则EEWE

不起作用。 EEMWE 置位后 4 个周期，硬件对其清零。见 EEPROM 写过程对 EEWE 位的描述。

- **Bit 1 – EEWE: EEPROM 写使能**

写使能信号 EEWE 是 EEPROM 的写入选通信号。当 EEPROM 数据和地址设置好之后，需置位 EEWE 以便将数据写入 EEPROM。此时 EEMWE 必须置位，否则 EEPROM 写操作将不会发生。写时序如下 (第 2 和第 3 步的次序可更改) :

1. 等待 EEWE 为 ”0”
2. 将新的 EEPROM 地址写入 EEAR (可选)
3. 将新的 EEPROM 数据写入 EEDR (可选)
4. 对 EECR 寄存器的 EEMPE 写 "1“
5. 在置位 EEMWE 之后的 4 个周期内置位 EEWE

注意 : 如有中断发生于步骤 4 和 5 之间将导致写操作失败。因为此时 EEPROM 写使能操作将超时。如果一个操作 EEPROM 的中断打断了另一个 EEPROM 操作，EEAR 或 EEDR 寄存器可能被修改，引起 EEPROM 操作失败。建议此时关闭全局中断标志 I。

经过写访问时间 (典型值 8.3 ms) 之后，EEPE 硬件清零。用户可以凭此位判断写时序是否已经完成。 EEWE 置位后， CPU 要停止两个时钟周期才会运行下一条指令。

- **Bit 0 – EERE: EEPROM 读使能**

读使能信号 EERE 是 EEPROM 的写入选通信号。当 EEPROM 地址设置好之后，需置位 EERE 以便将数据读入 EEAR。EEPROM 数据的读取只需要一条指令。读取 EEPROM 时 CPU 要停止 4 个时钟周期然后才能执行下一条指令。

用户在读取 EEPROM 时应该检测 EEWE。如果一个写操作正在进行，就无法读取 EEPROM，也无法改变寄存器 EEAR。

Table 29. EEPROM 编程时间

符号	标定的 RC 振荡器周期数 (1)	典型编程时间
CPU 发起的 EEPROM 写操作	8448	8.5 ms

Note: 1. 使用 1 MHz 时钟，不依赖于 CKSEL 熔丝位的设置。

掉电模式下的 EEPROM 写操作

当进入掉电休眠模式而 EEPROM 写操作被激活， EEPROM 写操作将继续，并在写访问时间结束前完成。但当操作完成后，晶振进行运行，芯片也没有完全进入掉电模式。因此建议在进入掉电模式前确认完成写操作。

防止 EEPROM 数据丢失

如果电源电压过低，CPU 和 EEPROM 有可能工作不正常，造成 EEPROM 数据的毁坏 (丢失)。这种情况在使用独立的 EEPROM 器件时也会遇到。

由于电压过低造成 EEPROM 数据损坏有两种可能：一是电压低于 EEPROM 写操作所需要的最低电压；二是 CPU 本身已经无法正常工作。

EEPROM 数据损坏的问题可以通过以下方法来避免：

1. 当电压过低时保持 AVR RESET 信号为低。这可以通过使能芯片的掉电检测电路 BOD 来实现。如果 BOD 电平无法满足要求，则可以使用外部复位电路。
2. 当电压过低时保持 AVR 内核在掉电休眠模式时。这将防止 CPU 尝试去解码与执行指令，有效保护 EEPROM 寄存器出现无意识写操作。
3. 若不需要提供软件改变存储器内容，将常数存于 Flash 存储器。 CPU 不能修改 Flash 存储器，则数据不会受到破坏。

通用串行接口 - USI

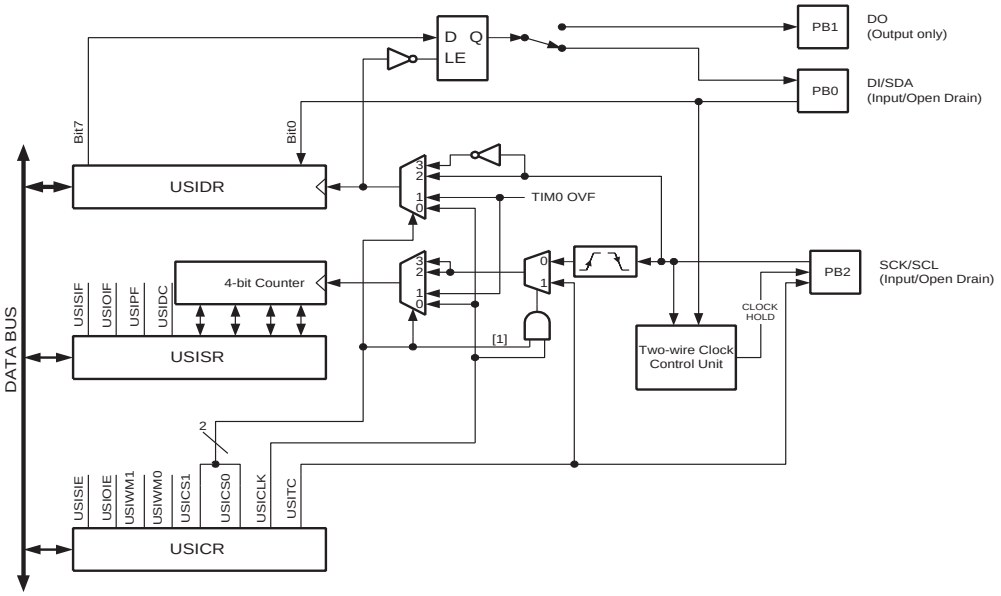
通用串行接口，或 USI，提供了进行串行通信所需的最基本的硬件资源。和最小化的控制软件结合之后，USI 可以提供比使用纯软件高得多的传输速率并使用更少的代码空间。可以使用中断来最小化处理器的工作量。 USI 的主要特性是：

- **两线同步数据传输 (主机或从机 , $f_{SCLmax} = f_{CK}/16$)**
- **三线同步数据传输 (主机 , $f_{SCKmax} = f_{CK}/2$, Slave $f_{SCKmax} = f_{CK}/4$)**
- **数据接收中断**
- **可以从空闲模式唤醒**
- **两线模式下可从所有的睡眠模式唤醒，包括掉电模式**
- **两线启动条件检测器有中断能力**

概述

Figure 39. 给出了一个简化的 USI 框图。

Figure 39. 通用串行接口框图



通过数据总线可直接访问 8 位的移位寄存器，这个寄存器包含正在发送和接收的数据。由于这个寄存器没有缓冲，因此要尽快读出其中的数据以保证数据不丢失。依据不同的工作模式，最高位与两个输出引脚之一相连。在串行寄存器输出和输出引脚之间有一个透明的锁存器，它的作用是将数据输出延迟到和用于数据输入采样的时钟沿相反的时钟沿。串行输入总是经过数据输入引脚 (DI) 进行采样，而与配置无关。

通过数据总线还可以访问 4 比特的计数器，并产生溢出中断。串行寄存器和计数器由相同的时钟驱动，从而定时器可以对接收或发送的比特数进行计数，并在传输结束时产生中断。当选择了外部时钟时，计数器将对时钟的上下两个沿进行计数。此时计数器统计沿的数目而不是比特数。有三种可选的时钟源：SCK 引脚、定时器 0 溢出或通过软件产生。

检测到起始条件后，两线时钟控制单元可以产生中断。它也可以在检测到起始条件或计数器溢出之后通过拉低时钟引脚来产生等待状态。

寄存器描述

USI 数据寄存器 - USIDR

Bit	7	6	5	4	3	2	1	0	
$0F ($2F)	**MSB**							**LSB**	USIDR
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

USI 串行寄存器没有缓冲，寻址数据寄存器 USIDR 时，实际操作的是串行寄存器。若在写寄存器的同一个周期产生了一个串行时钟，寄存器将被更新，但不会进行移位操作。移位操作依赖于 USICS1..0 的设置，可由外部时钟沿、定时器 / 计数器 0 溢出或直接通过软件使用 USICLK 来控制。即使没有选择任何连接模式 (USIWM1..0 = 0)，移位寄存器仍然可以使用外部数据输入 (DI/SDA) 及外部时钟输入 (SCK/SCL)。

输出引脚 (DO 或 SDA，由连接模式确定) 通过输出锁存器与数据寄存器的最高位 (位 7) 相连。选择了外部时钟源时 (USICS1 = 1)，输出锁存器在串行时钟的前半个周期打开 (透明)，如果使用的是内部时钟源 (USICS1 = 0)，它将一直打开。锁存器打开时写入新的 MSB 会立即在输出引脚反映出来。锁存器保证输入数据的采样时间与输出数据的改变发生在相反的时钟沿。

为了输出移位寄存器的数据，必须利用方向寄存器将对应的引脚设置为输出。

USI 状态寄存器 - USISR

Bit	7	6	5	4	3	2	1	0	
$0E ($2E)	USISIF	USIOIF	USIPF	USIDC	USICNT3	USICNT2	USICNT1	USICNT0	USISR
读 / 写	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

状态寄存器包括中断标志、线状态标志及计数器值。

注意在 USISR 寄存器进行读 - 修改 - 写操作，即使用 SBI 或 CBI 指令，将清除挂起的中断标志。建议用 OUT 指令改变寄存器内容。

- **Bit 7 – USISIF: 起始状态中断标志**

在两线模式下，检测到起始条件将置位 USISIF 标志。当使用输出禁止模式或三线模式，并且 (USICSx = 0b11 & USICLK = 0) 或 (USICS = 0b10 & USICLK = 0) 时， SCK 引脚的任意电平变化都会使此标志置位。

USISIF 置位时，若 USICR 寄存器的 USISIE，以及全局中断使能标志也置位，则会产生中断。 USISIF 标志位清零的唯一方式是对其写入逻辑 1。在两线模式下清除这一标志会释放由于检测到起始条件而保持的 SCL。

起始条件中断可把处理器从所有睡眠状态唤醒。

- **Bit 6 – USIOIF: 计数器溢出中断标志**

4 位计数器溢出 (即从 15 跳变到 0) 时此标志置位。若 USICR 寄存器的 USIOIE，以及全局中断使能标志也置位，则会产生中断。 USIOIF 标志位清零的唯一方式是对其写入逻辑 1。在两线模式下清这一标志会释放由于计数器溢出而保持的 SCL。

溢出中断可以将处理器从所有睡眠模式唤醒。

- **Bit 5 – USIPF: 停止状态标志**

在两线模式下，如果检测到停止状态， USIPF 标志置位。 USIOIF 标志位清零的方法是对其写入逻辑 1。注意这不是一个中断标志位。在进行总线主机仲裁时，可使用这个标志。

- **Bit 4 – USIDC: 数据输出冲突**

如果移位寄存器中的位 7 与物理引脚所对应的值不同， USIDC 置位。此标志仅在两线模式下有效。在进行总线主机仲裁时，可使用这个标志。

- **Bits 3..0 – USICNT3..0: 计数器值**

反映的是 4 位计数器的当前值。 CPU 可以直接读写这几位。

使计数器计数的时钟源有外部时钟边沿检测器、定时器 / 计数器 0 溢出及通过软件使用 USICLK 或 USITC 产生的时钟源。时钟源的确定由 USICS1..0 设置。外部时钟操作另有一个特性，即可以通过写 USITC 来产生时钟。方法是设置外部时钟源 (USICS1 = 1) 时将 USICLK 置 1。

即使没有选择任何连接模式 (USIWM1..0 = 0)，计数器仍然可以使用外部时钟输入 (SCK/SCL)。

USI 控制寄存器 - USICR

Bit	7	6	5	4	3	2	1	0	
$0D ($2D)	**USISIE**	**USIOIE**	**USIWM1**	**USIWM0**	**USICS1**	**USICS0**	**USICLK**	**USITC**	**USICR**
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	W	W	
初始值	0	0	0	0	0	0	0	0	

控制寄存器包括中断使能控制、连接模式设置、时钟选择设置及时钟选择信号。

- **Bit 7 – USISIE: 起始条件中断使能**

将此位置 1 可使能起始条件检测中断。如果 USISIE 及全局中断使能标志位置位时产生了一个这样的中断，那么中断将立即得到处理。更多的细节请参见 P59“Bit 7 – USISIF: 起始状态中断标志 ” 的描述。

- **Bit 6 – USIOIE: 计数器溢出中断使能**

此标志为 1 可产生计数器溢出中断。如果 USIOIE 及全局中断使能标志位置位时产生了一个这样的中断，那么中断将立即得到处理。更多的细节请参见 P59“Bit 6 – USIOIF: 计数器溢出中断标志 ” 的描述。

- **Bit 5..4 – USIWM1..0: 连接模式**

USIWM1..0用来设置连接模式。基本上只有输出功能受这几位的影响。数据及时钟输入不受所选模式的影响，总是保持同样的功能。因此，即使输出被禁止，计数器和移位寄存器照样可由外部提供时钟，也可以进行数据输入采样。 USIWM1..0 与 USI 操作的关系在 Table 30 中有简要介绍。

Table 30. USIWM1..0 与 USI 操作之间的关系

USIWM1	USIWM0	说明
0	0	输出，时钟保持，起始检测器禁止。引脚以普通端口方式工作。
0	1	三线模式。使用 DO、 DI 及 SCK 引脚 在这种模式下*数据输出* (DO) 功能取代了普通端口 IO 功能，但对应的 DDR 仍然控制这数据方向。端口引脚设置为输入时，引脚的上拉电阻由 PORT 位来控制。 *数据输入* (DI) 及*串行时钟* (SCK) 功能不影响正常的端口功能。作为主机工作时，软件通过操作 PORT 寄存器来产生时钟脉冲，同时数据方向设为输出。 USICR 寄存器中的 USITC 位可用作这一目的。
1	0	两线模式。使用 SDA (DI) 及 SCL (SCK) 引脚 (1)。 *串行数据* (SDA) 及*串行时钟* (SCL) 引脚是双向的，且使用集电极开路输出驱动器。通过设置 DDR 寄存器中相应的位来启动输出驱动器。 SDA 引脚的驱动器使能时，如果移位寄存器的输出或 PORT 寄存器对应的位为 0，那么输出驱动器会把 SDA 线强制拉低。否则 SDA 线将不被驱动 (即将它释放)。 SCL 引脚输出驱动器使能时，如果 PORT 寄存器中的对应位为 0，或者由于起始检测器的作用， SCL 线被强制置低。否则 SCL 线将不被驱动。 当起始检测器检测到起始条件且输出允许时， SCL 被拉低。清起始条件标志 (USISIF) 将释放此口线。启动这一模式不会影响 SDA 及 SCL 引脚的输入。在两线模式下 SDA 及 SCL 引脚的上拉无效。
1	1	两线模式。使用 SDA 及 SCL 引脚 两线模式下的一些操作在上面已有所描述，除了以下这点：当发生计数器溢出时 SCL 线保持为低，并一直保持到计数器溢出标志位 (USIOIF) 被清零。

Note: 1. 为了避免混淆， DI 及 SCK 引脚分别被重命名为*串行数据* (SDA) 及*串行时钟* (SCL)。

- **Bit 3..2 – USICS1..0: 时钟源选择**

通过这几位可以设置移位寄存器及计数器的时钟源。数据输出锁存器保证在使用外部时钟 (SCK/SCL) 时，输出数据的改变与输入数据 (DI/SDA) 的采样发生在相反的时钟沿。如果选择了软件方式或定时器 0 溢出时钟，输出锁存器即成为是透明的，输出可以立即改变。 USICS1..0 为 0 时软件方式使能。此时，向 USICLK 写 1 就可以同时给移位寄存器和计数器提供时钟。对于外部时钟 (USICS1 = 1)， USICLK 位不再用作选通信号，而是通过 USITC 在外部时钟及软件时钟之间进行选择。

Table 31 给出了 USICS1..0 及 USICLK 的设置与移位寄存器及 4 位计数器所使用的时钟之间的关系。

Table 31. USICS1..0 及 USICLK 的设置

USICS1	USICS0	USICLK	移位寄存器时钟源	4 位计数器时钟源
0	0	0	无时钟	无时钟
0	0	1	软件时钟 (USICLK)	软件时钟 (USICLK)
0	1	X	定时器 / 计数器 0 比较匹配	定时器 / 计数器 0 比较匹配
1	0	0	外部时钟，上升沿	外部时钟，上升及下降沿
1	1	0	外部时钟，下降沿	外部时钟，上升及下降沿
1	0	1	外部时钟，上升沿	软件时钟 (USITC)
1	1	1	外部时钟，下降沿	软件时钟 (USITC)

- **Bit 1 – USICLK: 时钟选通**

若 USICS1..0 为 0，置位 USICLK 将使移位寄存器进行一次移位，计数器累加一。此即为软件时钟。在同一指令周期里输出立即得到更新。移入移位寄存器的值在上一个指令周期得以采样。这一位的读出值为 0。

使用外部时钟时 (USICS1 = 1)， USICLK 的功能由时钟选通变为时钟选择寄存器。在这种情况下设置 USICLK 将选择 USITC 作为软件时钟源来驱动 4 位计数器 (见 Table 31)。

- **Bit 0 – USITC: 交替变换时钟端口引脚**

USITC 置位将使 SCK/SCL 出现 0、1 的交替变换。方向寄存器的设置不影响触发，但需要置位 DDRB2 使相应的端口成为输出。这个特性为主机实现提供了一个产生时钟的简单办法。这一位的读返回值为 0。

选用外部时钟 (USICS1 = 1) 并且 USICLK 置 1 时，对 USITC 进行写入将直接驱动 4 位计数器。作为主机工作时，这样可以更早地知道传输何时结束。

功能描述

三线模式

USI的三线工作模式与串行外设接口(SPI)模式0和模式1兼容，但没有从机选择(SS)功能。不过这可以通过软件来实现。这种模式下的引脚名称为 DI、 DO 和 SCK。

Figure 40. 三线模式简化框图

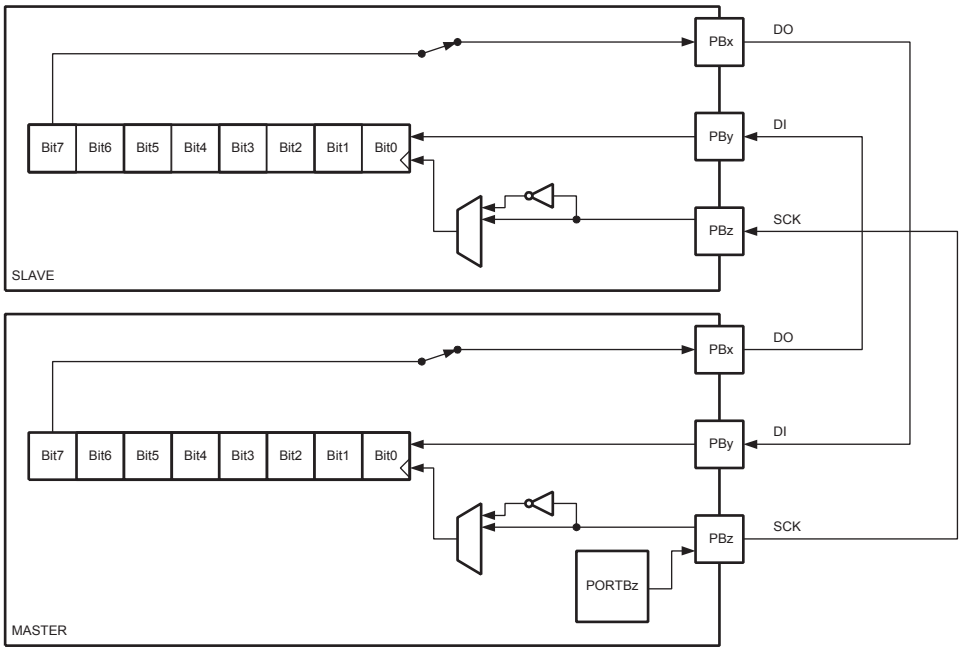


Figure40 给出了两个工作于三线模式的 USI 单元，一个为主机，另一个为从机。两个移位寄存器的连接方式使得 8 个 SCK 时钟之后，两个寄存器中的数据相互交换。同样的时钟还驱动 USI 的 4 位计数器。因此计数器溢出 (中断) 标志 USIOIF 可用来判断传输何时完成。这个时钟可以由两种方式产生：一是由主机软件通过操作端口寄存器 PORTB 来操作 PB2 引脚，二是置位 USICR 寄存器的 USITC。

Figure 41. 三线模式时序图

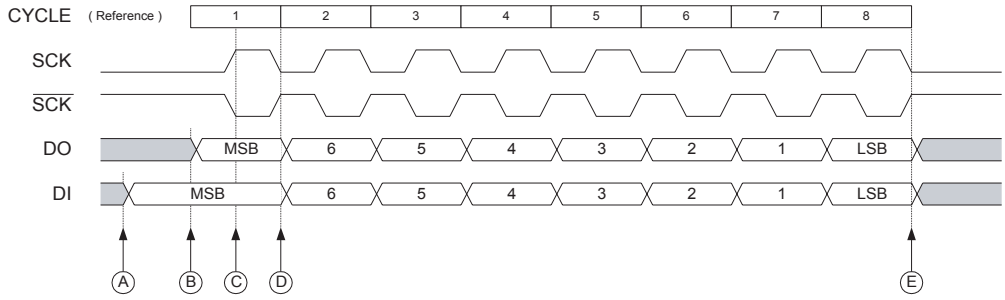


Figure 41. 给出了三线模式的时序。图的顶端是 SCK 的参考周期。在每一个这样的周期里都有一个比特的数据转移到 USI 移位寄存器 (USIDR) 中。SCK 时序展示了两种外部时钟模式。工作于外部时钟模式 0(USICS0 = 0) 时， DI 在时钟的上升沿采样，输出 DO 在下降沿改变 (数据寄存器移动一位)，外部时钟模式 1(USICS0 = 1) 使用与模式 0 相反的时钟沿，即在在下降沿进行数据采样，在上升沿改变输出。 USI 时钟模式对应于 SPI 数据模式 0 和模式 1。

由 Figure 41. 时序图可以看出，总线传输包括以下步骤：

1. 通过向串行数据寄存器中写入需要发送的数据来准备数据输出。通过设置数据方向寄存器 (DDRB2) 中的对应位来启动数据输出。注意， A 与 B 点之间并没有什么特殊的顺序，但是它们都必须早于数据采样点 C 点之前至少半个 USCK 周期。这点必须得到保障，以保证数据准备所需条件得到满足。 4 位计数器被重置为 0。
2. 主机通过软件改变 SCK 两次 (C 与 D) 来产生一个时钟脉冲。主从设备的输入引脚 (DI) 上的值由 USI 在第一个沿 (C) 进行采样；数据输出则在其相对的沿 (D) 改变。4 位计数器将对两个沿进行计数。
3. 在一个完整的寄存器 (字节) 传输过程中，第 2 步将重复 8 次。
4. 8 个时钟脉冲 (16 个时钟沿) 之后 , 计数器溢出 , 表明传输完成。传输的数据必须在下一次传输开始之前得到处理。如果处理器处于空闲模式，那么溢出中断会将它唤醒。依据通讯协议，从机现在可以将它的输出置为高阻状态。

SPI 主机工作例子

接下来的代码说明了如何将 USI 模块当作 SPI 主机来使用：

```
SPITransfer:
    out     USIDR,r16
    ldi     r16,(1<<USIOIF)
    out     USISR,r16
    ldi     r16,(1<<USIWM0)+(1<<USICS1)+(1<<USICLK)+(1<<USITC)
SPITransfer_loop:
    out     USICR,r16
    sbis    USISR,USIOIF
    rjmp    SPITransfer_loop
    in      r16,USIDR
    ret
```

这段代码仅使用了 8 条指令 (+ ret)，非常优化。示例代码假定 DO 及 SCK 引脚已经通过设置 DDRB 寄存器成为输出引脚。调用函数之前，r16 寄存器包含了要送到从机的数据，传输结束之后， r16 寄存器包含了从从机接收回来的数据。

第二和第三条指令是清 USI 计数器溢出标志及 USI 计数器。第四第五条指令设置三线模式、上升沿移位寄存器时钟、在 USITC 选通时进行计数以及触发 SCK。循环将重复运行 16 次。

下面的代码则演示了在最大速率 (fsck = fck/2) 下如何将 USI 模块作为 SPI 主机来使用：

```
SPITransfer_Fast:

    out     USIDR,r16
    ldi     r16,(1<<USIWM0)+(0<<USICS0)+(1<<USITC)
    ldi     r17,(1<<USIWM0)+(0<<USICS0)+(1<<USITC)+(1<<USICLK)

    out     USICR,r16 ; MSB
    out     USICR,r17
    out     USICR,r16
    out     USICR,r17
    out     USICR,r16
    out     USICR,r17
    out     USICR,r16
    out     USICR,r17
    out     USICR,r16
    out     USICR,r17
    out     USICR,r16
    out     USICR,r17
    out     USICR,r16
    out     USICR,r17
    out     USICR,r16 ; LSB
    out     USICR,r17

    in      r16,USIDR
ret
```

SPI 从机工作例子

接下来的代码展示了如何将 USI 模块作为 SPI 从机来使用：

```
init:
    ldi     r16,(1<<USIWM0)+(1<<USICS1)
    out     USICR,r16
...
SlaveSPITransfer:
    out     USIDR,r16
    ldi     r16,(1<<USIOIF)
    out     USISR,r16
SlaveSPITransfer_loop:
    sbis    USISR,USIOIF
    rjmp    SlaveSPITransfer_loop
    in      r16,USIDR
    ret
```

这段代码仅使用了 8 条指令 (+ ret)，非常优化。示例代码假定 DO 及 SCK 引脚已经通过设置 DDRB 寄存器分别成为输出和输入引脚。调用函数之前， r16 寄存器包含了要送到主机的数据，传输结束之后， r16 寄存器包含了从主机接收回来的数据。

开头的两条指令为初始化指令，仅需执行一次。这些指令用来设置三线模式及上升沿沿移位寄存器时钟。循环一直重复到 USI 计数器溢出标志位置位。

两线模式

USI 两线模式兼容 IC 间 (TWI) 总线协议，但没有输出转换速率限制及输入噪声滤波器。这种模式下的引脚名为 SCL 和 SDA。

Figure 42. 两线模式框图

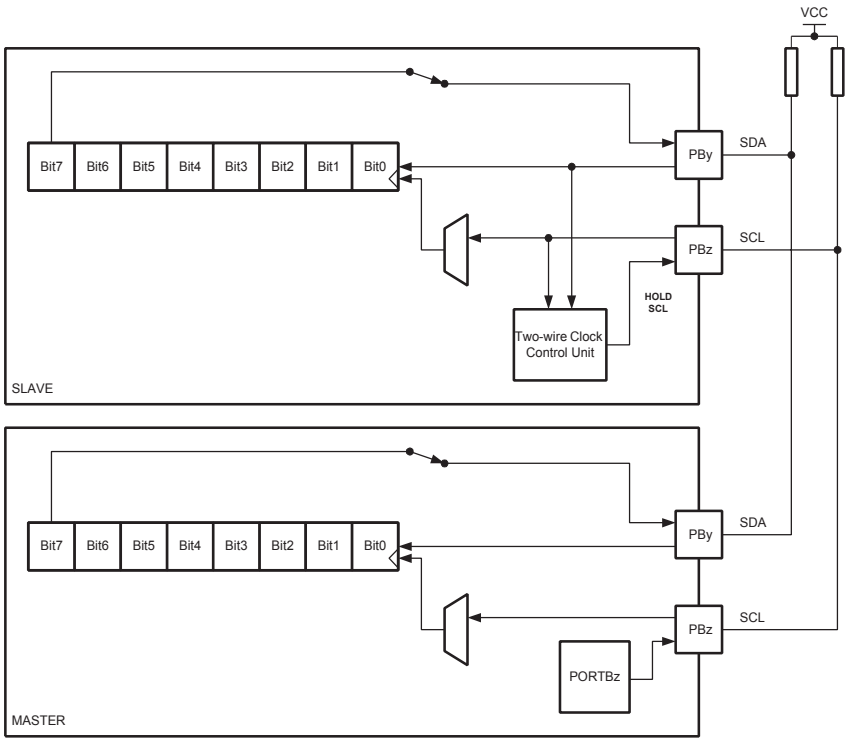
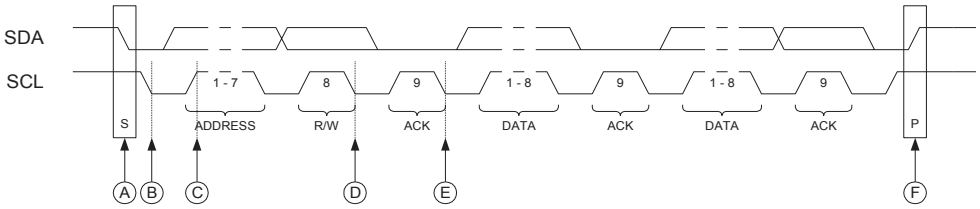


Figure42 展示了两个工作在两线模式下的 USI 单元，一个为主机，另一个为从机。由于系统的工作在很大程度上取决于所使用的通信方案，这里仅仅给出了物理层。在这一层中，主机和从机的主要区别是：串行时钟由主机产生，只有从机使用时钟控制单元。时钟的产生必须由软件实现，但移位操作是自动完成的。数据传输过程中只有下降沿触发数据移位。通过强制 SCL 时钟为低，从机可以在传输启动或结束时插入等待状态。这说明，产生上升沿之后，主机必须检查 SCL 线是否已经释放。

由于时钟同时驱动计数器计数，计数器溢出可用来判断传输是否结束。主机通过 PORTB 寄存器使 PB2 引脚产生时钟。

物理层不确定数据的方向。数据流控制由协议决定，如 TWI 总线协议。

Figure 43. 两线模式典型时序图



根据此时序图 (Figure 43.)，总线传输包括以下步骤：

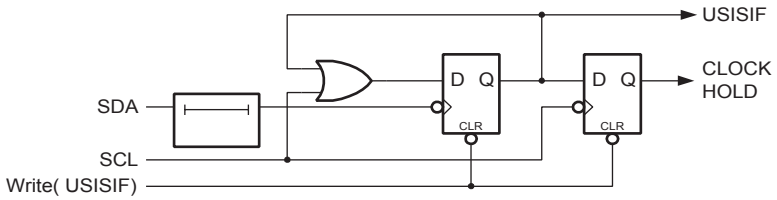
1. 当SCL为高时(A)，主机可通过将SDA强制拉低来产生一个起始条件。有两种方法可使 SDA 强制为低：一种是对移位寄存器的第 7 位写 0，另一种是将 PORTB0 寄存器中的对应位置 0。在此之前还需要通过数据方向寄存器将相关引脚设置为

输出。从机起始条件监测逻辑 (Figure 44.) 检测到起始条件后置位 USISIF 标志。如果有必要，可通过此标志产生中断。

2. 此外，当主机强制在 SCL 线产生下降沿 (B) 后，起始条件检测器将保持 SCL 为低。这可将从机从睡眠状态唤醒，或在设置移位寄存器接收地址之前完成其他任务。这个操作通过清起始条件标志及复位计数器来实现。
3. 主机设置第一个需要传输的位，并释放 SCL 线 (C)。从机在 SCL 时钟上升沿对数据进行采样并将数据转移到串行寄存器中。
4. 当包含从机地址及数据方向 (读或写) 的 8 位数据全都传输完完毕之后，从机计数器溢出， SCL 被强制置为低 (D)。如果这个从机不是主机所寻址的设备，它将释放 SCL 线并等待下一个起始条件。
5. 如果从机被寻址，那么在 SCL 被再次拉低之前 (在释放 SCL(D) 之前计数器要达到 14)，在应答期间它将保持 SDA 线为低。 R/W 位决定是主机还是从机输出数据。若 R/W 为 1，主机执行读操作 (即从设备驱动 SDA 线)。在应答 (E) 之后从机可以保持 SCL 线为低。
6. 现在可以在同一方向传输多个字节的数据了，直到主机发出停止条件 (F)，或者产生一个新的起始条件。

如果从机无法接收更多的数据，那么它不要应答最后接收到的数据。主机进行读操作时，接收到最后一个字节后，它必须强制应答位为低来结束读操作。

Figure 44. 起始条件监测器逻辑电路图



起始条件监测器

Figure 44. 为起始条件监测器。SDA 被延迟 (50 到 300 ns)，以保证 SCL 的有效采样。起始条件监测器仅在两线模式下使能。

起始条件监测器工作在异步模式，因此可以将处理器从掉电睡眠模式唤醒。但是，通讯协议可能对 SCL 的保持时间有限制。在这种情况下就必须考虑由 CKSEL 熔丝位确定的晶振启动时间 (见 P24“ 时钟系统及其分布 ”)。更多细节请参见 P59“Bit 7 – USISIF: 起始状态中断标志 ” 的描述。

其他的 USI 用法

如果 USI 不用于串行通讯，则由于其设计上的灵活性，可以用来完成其他工作。

半双工异步数据传输

在三线模式下使用移位寄存器可以实现比软件方案更紧凑、更高效的 UART 功能。

4 位计数器

这个 4 比特计数器可作为独立的计数器来使用，并可产生溢出中断。要注意的是，如果计数器由外部时钟源提供时钟，那么两个时钟边沿都会使其计数。

12 位定时器 / 计数器

将 USI 的 4 比特计数器与定时器 / 计数器 0 结合起来使用可得到一个 12 位的计数器。

边沿触发的外部中断

把计数器的值设到最大 (F)，可实现一个额外的外部中断。溢出标志位及中断使能位都为此外部中断服务。可通过 USICS1 来选择这一特性。

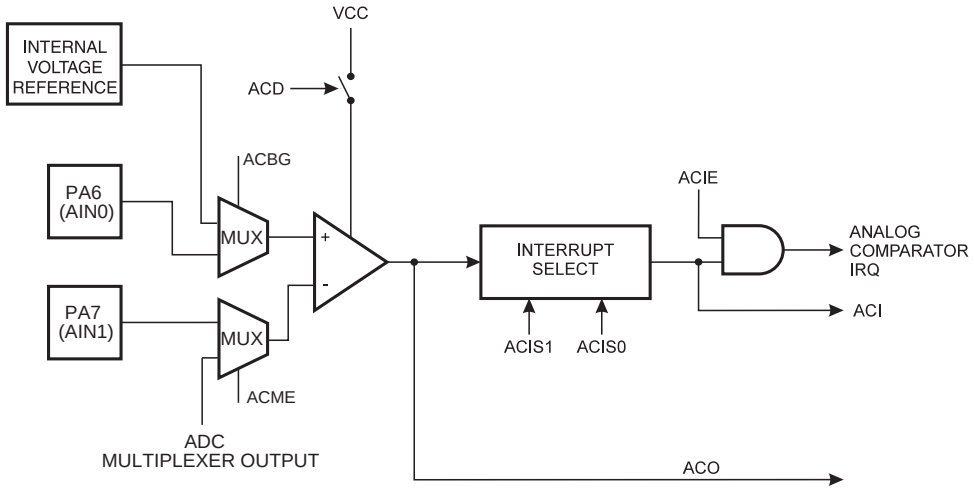
软件中断

计数器溢出中断还可用作软件中断。

模拟比较器

模拟比较器对正极 AIN0 的值与负极 AIN1 的值进行比较。当 AIN0 上的电压比负极 AIN1 上的电压要高时，模拟比较器的输出 ACO 即置位。比较器的输出可用来触发定时器 / 计数器 1 的输入捕捉功能。此外，比较器还可触发自己专有的、独立的中断。用户可以选择比较器是以上升沿、下降沿还是交替变化的边沿来触发中断。 Figure45 为比较器及其外围逻辑电路的框图。

Figure 45. 模拟比较器框图



模拟比较器控制及状态寄存器 - ACSR

Bit	7	6	5	4	3	2	1	0	
$08 ($28)	**ACD**	**ACBG**	**ACO**	**ACI**	**ACIE**	**ACME**	**ACIS1**	**ACIS0**	**ACSR**
读 / 写	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	X	0	0	0	0	0	

- **Bit 7 – ACD: 模拟比较器禁用**

ACD 置位时，模拟比较器的电源被切断。可以在任何时候设置此位来关掉模拟比较器。这可以减少器件工作模式及空闲模式下的功耗。改变 ACD 位时，必须清零 ACSR 寄存器的 ACIE 位来禁止模拟比较器中断。否则 ACD 改变时可能会产生中断。

- **Bit 6 – ACBG: 选择模拟比较器的能隙基准源**

ACBG 置位后，模拟比较器的正极输入由固定能隙基准源所取代。

- **Bit 5 – ACO: 模拟比较器输出**

模拟比较器的输出直接连到 ACO。

- **Bit 4 – ACI: 模拟比较器中断标志**

当比较器的输出事件触发了由 ACI1 及 ACI0 定义的中断模式时， ACI 置位。如果 ACIE 和 SREG 寄存器的全局中断标志 I 也置位，那么模拟比较器中断服务程序即得以执行，同时 ACI 被硬件清零。 ACI 也可以通过写 1 来清零。

- **Bit 3 – ACIE: 模拟比较器中断使能**

当 ACIE 位被置 1 且状态寄存器中的全局中断标志 I 也被置位时，模拟比较器中断被激活。否则中断被禁止。

- **Bit 2 – ACME: 模拟比较器复用使能**

当 ACME 置位且 ADC 关闭 (ADCSR 寄存器中的 ADEN 位为 0)，ADMUX 的 MUX3...0 选择输入引脚代替模拟比较器的负输入，如 P70Table 33 所示。若 ACME 清零或 ADEN 置位， PA7(AIN1) 作为模拟比较器的负输入。

- **Bits 1, 0 – ACIS1, ACIS0: 模拟比较器中断模式选择**

这两位确定哪个事件可以触发模拟比较器中断。 Table 32 给出了不同的设置。

Table 32. ACIS1/ACIS0 设置 (1)

ACIS1	ACIS0	中断模式
0	0	比较器输出变化即可触发中断
0	1	保留
1	0	比较器输出的下降沿产生中断
1	1	比较器输出的上升沿产生中断

Note: 1. 当改变ACIS1/ACIS0位时，必须通过清除ACSR寄存器中断使能位禁用模拟比较中断。否则当位变化时可能会出现中断。

Table 33. 模拟比较输入选择 (1)

ACME	ADEN	MUX3...0(3)	模拟比较负输入
0	X	XXXX	AIN1
1	1	XXXX	AIN1
1	0	0000	ADC0
1	0	0001	ADC1
1	0	0010	ADC2
1	0	0011	ADC3
1	0	0100	ADC4
1	0	0101	ADC5
1	0	0110	ADC6(2)
1	0	0111	ADC7(2)
1	0	1000	ADC8
1	0	1001	ADC9
1	0	1010	ADC10
1	0	1011	未定义
1	0	1100	未定义
1	0	1101	未定义
1	0	1110	未定义
1	0	1111	未定义

Notes: 1. MUX4 不影响模拟比较器输入选项
2. 若模拟比较器使能 ,则PA6 与PA7的引脚电平变化中断禁用 ,无论AIN1 或 AIN0是否作为模拟比较器的输入。
3. 在一个时钟周期延迟后， MUX3...0 选项有效。

模数转换器

特点

- **10 位精度**
- **±2 LSB 的绝对精度**
- **0.5 LSB 的非线性度**
- **可选补偿注销**
- **65 - 260 µs 的转换时间**
- **11 复用单端输入通道**
- **8 路差分输入通道**
- **含可选增益为 20x 的 7 路差分输入通道**
- **可选的向左调整 ADC 读数**
- **0 - AVCC 的 ADC 输入电压范围**
- **可选的 ADC 参考电压**
- **连续转换或单次转换模式**
- **ADC 转换结束中断**
- **基于睡眠模式的噪声抑制器**

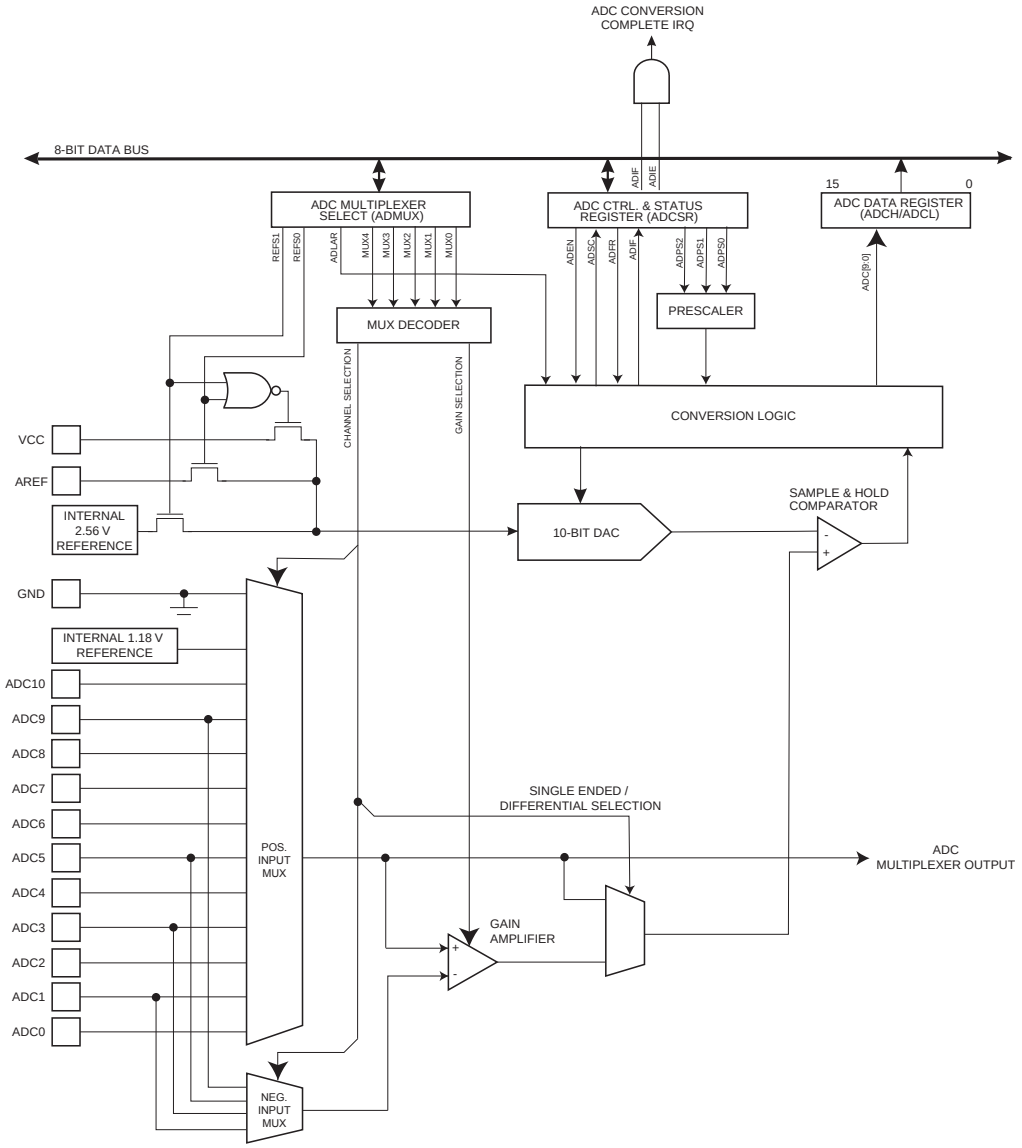
ATtiny26(L)有一个10位的逐次逼近型ADC。ADC与一个11通道的模拟多路复用器连接，能对 8 路差分电压输入或来自端口 A 的七路与来自端口 B 的四路组成的 11 路单端输入电压进行采样。其中七路有可编程增益级，可在 A/D 转换前提供 dB (1x) 与 26 dB (20x) 的电压增益。共有四组三路差分模拟输入通道选项。每组中所有的通道共负极，而其他 ADC 输入作为正输入端。单端电压输入以 0V (GND) 作为基准。

ADC 包括一个采样保持电路，以确保在转换过程中输入到 ADC 的电压保持恒定。ADC 的框图如 Figure46 所示。

ADC 由 AVCC 引脚单独提供电源。 AVCC 与 V_{CC} 之间的偏差不能超过 ± 0.3V。请参考 P86“ADC 噪声抑制技术 ” 来了解如何连接这个引脚。

标称值为 2.56V 的基准电压位于器件之内。基准电压可以通过在 AREF 引脚上加一个电容进行解耦，以更好地抑制噪声。

Figure 46. 模数转换器方框图



操作

ADC 通过逐次逼近的方法将输入的模拟电压转换成一个 10 位的数字量。最小值代表 GND ,最大值代表AREF引脚上的电压再减去1 LSB。通过写ADMUX寄存器的REFS位可以把 AVCC 或内部 2.56V 的参考电压连接到 AREF 引脚。在 AREF 上外加电容可以对片内参考电压进行解耦以提高噪声抑制性能。

模拟输入通道可以通过写ADMUX 寄存器的 MUX 位来选择。任何 ADC 输入引脚 ,像 GND 及固定能隙参考电压，都可以作为 ADC 的单端输入。选中的 ADC 输入引脚可选作差分增益放大的正输入或负输入。

如果选择差分通道 ,差分增益级由差分增益因子的选择决定。注意正端的输入电压必须大于负端的输入电压，否则增益达到 0V (GND) 将趋于饱和。放大后的电压值将作为 ADC 的模拟输入电压。若使用单端输入，则将不使用增益放大器。

ADC 有两种工作模式 – 单独转换与连续工作模式。使用单独转换模式 ,每次转换由用户启动。使用连续转换模式， ADC 不断采样与更新 ADC 数据寄存器。 ADCSR 寄存器的 ADFR 位选择使用哪种模式。

ADC 由 ADCSR 寄存器的 ADEN 位使能。在 ADEN 设置前，参考电压与输入通道无效。当 ADEN 清零时， ADC 没有功耗，因此建议在进入省电模式前关闭 ADC。

通过在 ADSC 位写入逻辑 1 来启动转换。该位在转换过程中始终为 1，当转换完成后该位置 0。若在转换过程中选择差分数据通道， ADC 将在执行通道变换前结束转换。

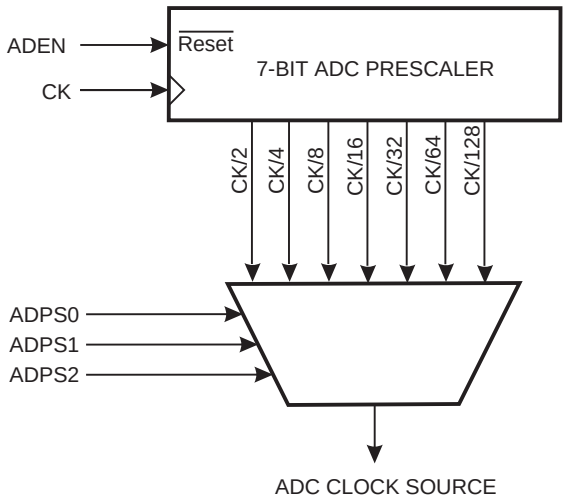
ADC转换结果为10位，存放于ADC数据寄存器ADCH及ADCL中。默认情况下转换结果为右对齐，但可通过设置 ADMUX 寄存器的 ADLAR 变为左对齐。

如果要求转换结果左对齐，且最高只需 8 位的转换精度，那么只要读取 ADCH 就足够了。否则要先读 ADCL，再读 ADCH，以保证数据寄存器中的内容是同一次转换的结果。一旦读出 ADCL， ADC 对数据寄存器的寻址就被阻止了。也就是说，读取 ADCL 之后，即使在读 ADCH 之前又有一次 ADC 转换结束，数据寄存器的数据也不会更新，从而保证了转换结果不丢失。 ADCH 被读出后， ADC 即可再次访问 ADCH 及 ADCL 寄存器。

ADC转换结束可以触发中断。即使由于转换发生在读取ADCH与ADCL之间而造成ADC无法访问数据寄存器，并因此丢失了转换数据，中断仍将触发。

预分频与转换时序

Figure 47. ADC 预分频器



逐次逼近电路需要一个从 50 kHz 到 200 kHz 的输入时钟以获得最大精度。 ADC 模块包括一个预分频器，它可以产生可接受的 ADC 时钟。

ADCSR 寄存器的 ADPS 位用于从片内产生一个超过 100 kHz 的适当的 ADC 时钟输入信号。预分频器从 ADCSR 寄存器的 ADEN 位置位启动 ADC 起开始计数。ADEN 置位时预分频器保持运转，当 ADEN 为低时预分频器复位。

转换在 ADCSR 的 ADSC 位设置后的上升沿开始。如果使用差分通道，转换在 ADEN 设置后的第二个上升沿启动。

正常转换需要 13 个 ADC 时钟周期。在这种情况下， ADC 需要更多的时钟周期来初始化及最小化补偿错误。ADC 使能 (ADCSRA 寄存器的 ADEN 置位) 后的第一次转换需要 25 个 ADC 时钟周期。

当使用差分通道时需特别注意。一旦使用差分通道，需要 125 µs 来稳定增益级值。因此在头 125 µs 时不能启动转换。或在此时得到的转换增益应该舍弃。在改变 ADC 参考电压 (通过改变 ADMUX 寄存器的 REFS1:0 位) 后的第一次差分转换前，也应观察 125 µs。

在普通的 ADC 转换过程中，采样保持在转换启动之后的 1.5 个 ADC 时钟开始；而第一次 ADC 转换的采样保持则发生在转换启动之后的 13.5 个 ADC 时钟。转换结束后，ADC 结果被送入 ADC 数据寄存器，且 ADIF 标志置位。ADSC 同时清零 (单次转换模式)。之后软

件可以再次置位 ADSC 标志，从而在 ADC 的第一个上升沿启动一次新的转换。在连续转换模式下，当 ADSC 为 1 时，只要转换一结束，下一次转换马上开始。使用连续转换模式且 ADC 时钟频率为 200 kHz 时最低转换时间为 65 µs，等价于 15 kSPS。转换时间请见 Table 34。

Figure 48. ADC 时序图，第一次转换 (单次转换模式)

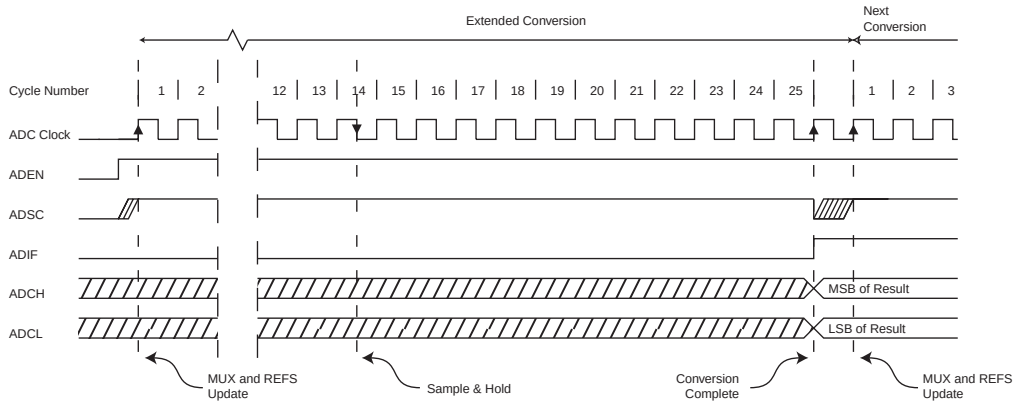


Figure 49. ADC 时序图，单次转换

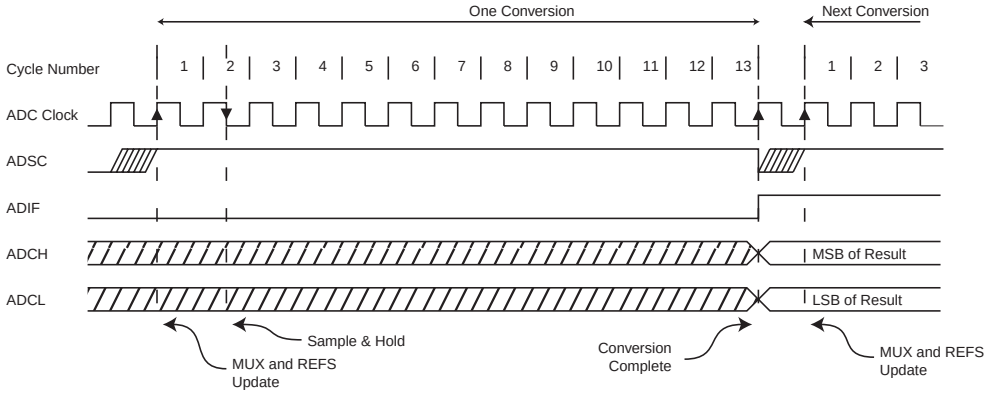


Figure 50. ADC 时序图，连续转换

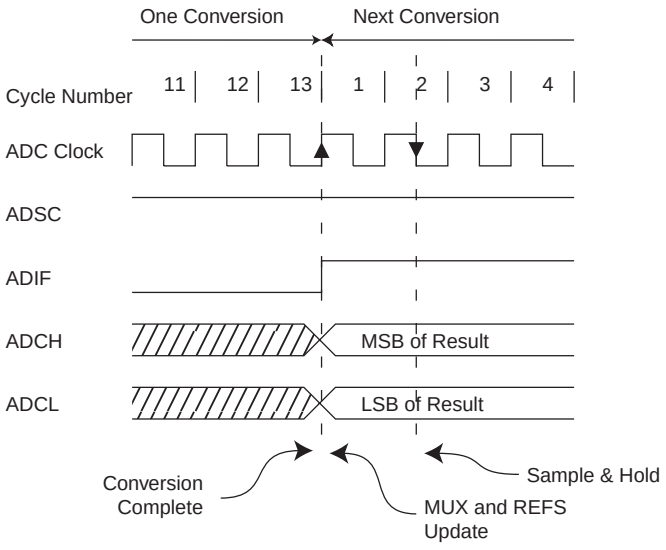


Table 34. ADC 转换时间

条件	采样 & 保持 (启动转换后的时钟周期数)	转换时间 (周期)	转换时间 (µs)
第一次转换	13.5	25	125 - 500
正常转换	1.5	13	65 - 260

ADC 噪声抑制器

ADC的噪声抑制器使其可以在睡眠模式下进行转换，从而降低由于CPU及外围I/O设备噪声引入的影响。噪声抑制器可在 ADC 降噪模式及空闲模式下使用。为了使用这一特性，应采用如下步骤：

1. 确定 ADC 已经使能，且没有处于转换状态。工作模式应该为单次转换，并且 ADC 转换结束中断使能。
   ADEN = 1
   ADSC = 0
   ADFR = 0
   ADIE = 1
2. 进入 ADC 降噪模式 (或空闲模式)。一旦 CPU 被挂起， ADC 便开始转换。
3. 如果在 ADC 转换结束之前没有其他中断产生，那么 ADC 中断将唤醒 CPU 并执行 ADC 转换结束中断服务程序。

ADC 转换结果

转换结束后 (ADIF 为高)，转换结果被存入 ADC 结果寄存器 (ADCL, ADCH)。

单次转换的结果如下：

$$ADC = \frac{V_{IN} \cdot 1024}{V_{REF}}$$

式中，V_{IN} 为被选中引脚的输入电压，V_{REF} 为参考电压 (参见 P78Table 36 及 P79Table 37)。 0x000 代表模拟地电平， 0x3FF 代表所选参考电压的数值减去 1LSB。

使用差分通道的结果为：

$$ADC = \frac{(V_{POS} - V_{NEG}) \cdot GAIN \cdot 1024}{V_{REF}}$$

式中， V_{POS} 为输入正电压， V_{NEG} 为输入负电压，GAIN 为所选增益因子， V_{REF} 为所选参考电压。注意，V_{POS} 必须始终大于 V_{NEG} ，否则 ADC 值最大为 0x000。Figure51 给出差分输入范围 的解析。

Table 35 给出当差分输入通道对(ADCn - ADCm)所选增益为GAIN，参考电压为V_{REF}时输出结果。

Figure 51. 差分测量范围

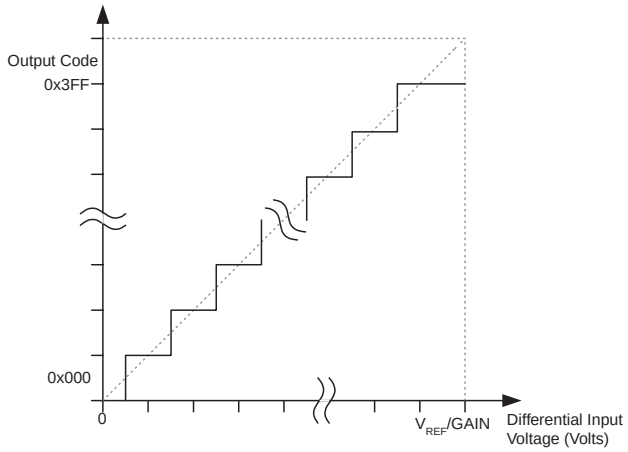


Table 35. 输入电压及输出码之间的关联关系

V_{ADCn}	读出的代码	对应的十进制数值
$V_{ADCm} + V_{REF}/GAIN$	0x3FF	1023
$V_{ADCm} + (1023/1024)\ V_{REF}/GAIN$	0x3FF	1023
$V_{ADCm} + (1022/1024)\ V_{REF}/GAIN$	0x3FE	1022
...	...	...
$V_{ADCm} + (1/1024)\ V_{REF}/GAIN$	0x001	1
V_{ADCm}	0x000	0

例：

ADMUX = 0xEB (ADC0 - ADC1, ， 20x 增益，参考电压 2.56V，左对齐)

.ADC0 上的电压为 400 mV ， ADC1 上的电压为 300 mV。

ADCR = 1024 * 20 * (400 - 300) / 2560 = 800 = 0x320

ADCL 的内容为 0x00 ,ADCH 的内容为 0xC8。向 ADLAR 写 0 ,对结果进行右对齐之后得到 ADCL = 0x20 ， ADCH = 0x03。

ADC 多路复用选择寄存器 - ADMUX

Bit	7	6	5	4	3	2	1	0	
$07 ($27)	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

- **Bit 7, 6 – REFS1, REFS0: 参考电压选择**

如 Table 36 所示 ,通过这几位可以选择参考电压。如果在转换过程中改变了它们的设置，只有等到当前转换结束 (ADCSR 寄存器的 ADIF 置位) 之后改变才会起作用。为得到最大精度，改变这两位后第一次的转换结果应该舍弃。若使用差分通道，建议 AVCC 或外部 AREF不超过(AVCC - 0.2V) ,因为这样会影响ADC精度。如果在AREF引脚上施加了外部参考电压，内部参考电压就不能被选用了。

Table 36. ADC 参考电压选择

REFS1	REFS0	参考电压选择
0	0	AVCC
0	1	AREF (PA3)，内部 Vref 关闭
1	0	2.56 V 的片内基准电压源，AREF 引脚 (PA3) 未连接
1	1	2.56 V 的片内基准电压源， AREF 引脚外加滤波电容

- **Bit 5 – ADLAR: ADC 转换结果左对齐**

ADLAR影响ADC转换结果在ADC数据寄存器中的存放形式。ADLAR置位时转换结果为左对齐，否则为右对齐。 ADLAR 的改变将立即影响 ADC 数据寄存器的内容，不论是否有转换在进行。关于这一位的完整描述请见 P85“ADC 数据寄存器 – ADCL 及 ADCH” 。

- **Bits 4..0 – MUX4..MUX0: 模拟通道与增益选择位**

通过这几位的设置，可以对连接到 ADC 的模拟输入及差分通道的增益进行选择。细节见 Table 37。如果在转换过程中改变这几位的值，那么只有到转换结束 (ADCSR 寄存器的 ADIF 置位) 后新的设置才有效。

Table 37. 输入通道与增益选择

MUX4..0	单端输入	差分输入正极	差分输入负极	增益
00000	ADC0	N/A		
00001	ADC1			
00010	ADC2			
00011	ADC3			
00100	ADC4			
00101	ADC5			
00110	ADC6			
00111	ADC7			
01000	ADC8			
01001	ADC9			
01010	ADC10			
01011	N/A	ADC0	ADC1	20x
01100		ADC0	ADC1	1x
01101(1)		ADC1	ADC1	20x
01110		ADC2	ADC1	20x
01111		ADC2	ADC1	1x
10000	N/A	ADC2	ADC3	1x
10001(1)		ADC3	ADC3	20x
10010		ADC4	ADC3	20x
10011(1)		ADC4	ADC3	1x
10100	N/A	ADC4	ADC5	20x
10101		ADC4	ADC5	1x
10110(1)		ADC5	ADC5	20x
10111		ADC6	ADC5	20x
11000		ADC6	ADC5	1x
11001	N/A	ADC8	ADC9	20x
11010		ADC8	ADC9	1x
11011(1)		ADC9	ADC9	20x
11100		ADC10	ADC9	20x
11101		ADC10	ADC9	1x
11110	1.18V (V_{BG})	N/A		
11111	0V (GND)			

Note: 1. 只作为补偿测量。见 P86“ 偏移补偿方案 ” 。

ADC 控制及状态寄存器 - ADCSR

Bit	7	6	5	4	3	2	1	0	
$06 ($26)	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSR
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

- **Bit 7 – ADEN: ADC 使能**

ADEN置位即启动ADC ,否则ADC功能关闭。在转换过程中关闭ADC将立即中止正在进行的转换。

- **Bit 6 – ADSC: ADC 开始转换**

在单次转换模式下，ADSC 置位将启动一次 ADC 转换。在连续转换模式下，ADSC 置位将启动首次转换。第一次转换在 ADC 启动之后置位 ADSC，或者在使能 ADC 的同时置位 ADSC。第一次转换执行 ADC 初始化的工作。

在转换进行过程中读取 ADSC 的返回值为 1 ,直到转换结束。当虚转换在实转换前出现，ADSC 在完成实转换前一直为高。 ADSC 清零不产生任何动作。

- **Bit 5 – ADFR: ADC 连续工作选择**

当该位置位， ADC 工作在连续工作模式。在该模式下， ADC 不断对数据寄存器采样与更新。该位清零将终止连续工作模式。

- **Bit 4 – ADIF: ADC 中断标志**

在 ADC 转换结束，且数据寄存器被更新后， ADIF 置位。如果 ADIE 及 SREG 中的全局中断使能位 I 也置位，ADC 转换结束中断服务程序即得以执行，同时 ADIF 硬件清零。此外，还可以通过向此标志写 1 来清 ADIF。要注意的是，如果对 ADCSR 进行读 - 修改 - 写操作，那么待处理的中断会被禁止。这也适用于 SBI 及 CBI 指令。

- **Bit 3 – ADIE: ADC 中断使能**

若 ADIE 及 SREG 的位 I 置位， ADC 转换结束中断即被使能。

- **Bits 2..0 – ADPS2..0: ADC 预分频器选择位**

由这几位来确定 CK 频率与 ADC 输入时钟之间的分频因子。

Table 38. ADC 预分频选择

ADPS2	ADPS1	ADPS0	分频因子
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

ADC 数据寄存器 - ADCL 及 ADCH

ADLAR = 0

Bit	15	14	13	12	11	10	9	8	
$05 ($25)	–	–	–	–	–	–	ADC9	ADC8	ADCH
$04 ($24)	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	
读 / 写	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
初始值	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

ADLAR = 1

Bit	15	14	13	12	11	10	9	8	
$05 ($25)	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADCH
$04 ($24)	ADC1	ADC0	–	–	–	–	–	–	ADCL
	7	6	5	4	3	2	1	0	
读 / 写	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
初始值	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

ADC 转换结束后，转换结果存于这两个寄存器之中。ADMUX 寄存器中的 ADLAR 位影响从寄存器中读取结果的方式。若 ADLAR 置位，则结果左对齐，否则右对齐 (默认值)。如果转换结果为左对齐，且要求的精度不高于 8 比特，那么仅需读取 ADCH 就足够了。否则必须先读出 ADCL 再读 ADCH。

- **ADC9..0: ADC 转换结果**

这几位表示 ADC 转换的结果。对差分通道，它是增益调整后的绝对值，如 P79Table 37 所示。对单端通道， $000 表示模拟地， $3FF 表示所选参考电压减 1LSB。

扫描多路通道

由于模拟通道的转换总是延迟到转换结束，连续运行模式不必中断转换就能扫描多路通道。典型的 ADC 转换结束中断用来执行通道移位。无论如何，用户应将如下因素考虑在。

一旦结果就绪，中断就会触发。在连续工作模式下，当中断触发，下一次转换立即开始。若在中断触发后 ADMUX 改变，下一次转换已经开始，则将使用旧的设置。

ADC 噪声抑制技术

ATtiny26(L) 内部及外部的数字电路都会产生电磁干扰 (EMI)，从而影响模拟测量的精度。如果转换精度要求较高，那么可以通过以下方法来减少噪声：

1. ATtiny26(L) 模拟部分与模拟器件应该有独立的模拟地。模拟地与数字地通过 PCB 的单独点连接在一起。
2. 模拟通路越短越好。保证模拟信号线位于模拟地之上，并使它们与高速转换切换的数字信号线分开。
3. 如 Figure52 所示， AVCC 应通过一个 LC 网络与数字电压源 V_{CC} 连接。
4. 使用 ADC 噪声抑制器来降低来自 CPU 的干扰噪声。
5. 如果有 ADC 端口被用作数字输出，那么必须保证在转换进行过程中它们不会有电平的切换。

偏移补偿方案

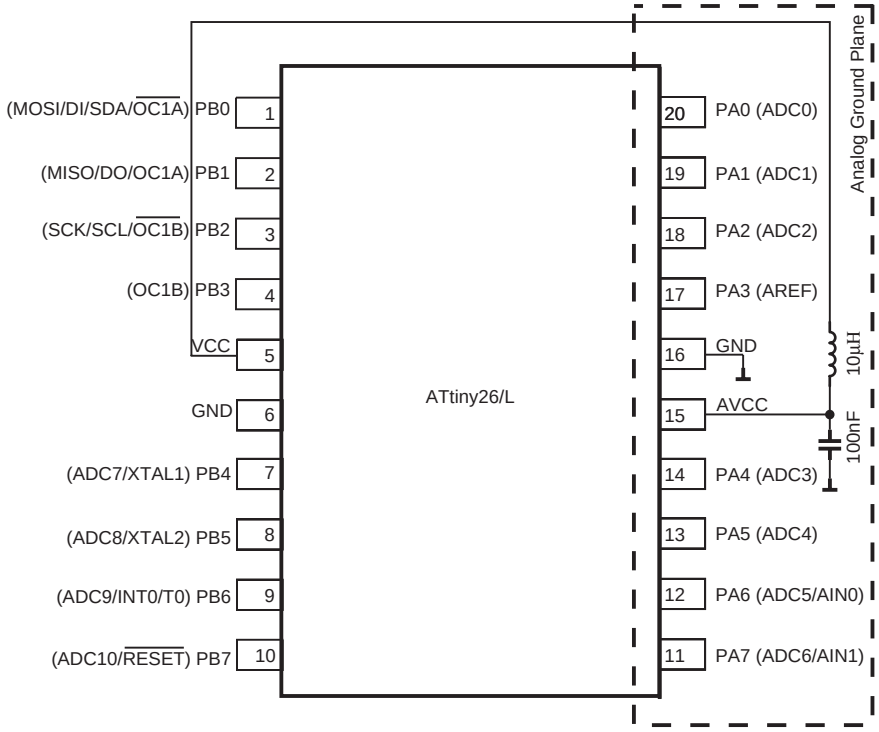
ADC 中所有的有效增益级与差分到单端级前都有偏移补偿电路，以尽量实现零偏移。

在统一差分测量的情况下，差分到单端级最差的偏移为 5 mV 偏移 (典型值 3 mV) 或 2LSBs。

在 20x 增益差分测量情况下，ADC 转换结果最差偏移量为 10 mV。如果差分通道转换使用内部参考电压 (2.56V) 则 10 位 ADC 的 1LSB 为 2.56 mV。即 ,最大错误大约为 4LSBs。该偏差在短期内保持稳定，由于温度作为导致偏移的主要因素变化比较慢。一般情况下，在温度范围内偏移变化为 5 mV，即大约 2LSBs。

如果想得到更好的偏移补偿，可以使用一条通道作为差分输入参考与完全模拟链路的实际偏移测量通道。这样使用软件可从测量结果中减去残余偏移量。使用这种偏移修正软件，通道的偏移量将低于 1LSB。

Figure 52. ADC 电源连接图

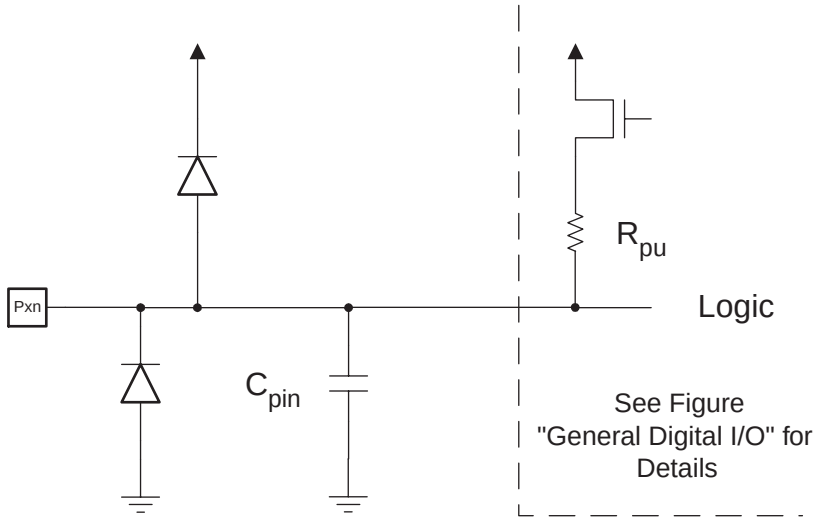


I/O 端口

介绍

作为通用数字 I/O 使用时，所有 AVR I/O 端口都具有真正的读 - 修改 - 写功能。这意味着用 SBI 或 CBI 指令改变某些管脚的方向 (或者是端口电平、禁止 / 使能上拉电阻) 时不会无意地改变其他管脚的方向 (或者是端口电平、禁止 / 使能上拉电阻)。输出缓冲器具有对称的驱动能力，可以输出或吸收大电流，直接驱动 LED。所有的端口引脚都具有与电压无关的上拉电阻。并有保护二极管与 V_{CC} 和地相连，如 Figure53 所示。

Figure 53. I/O 引脚等效原理图



本节所有的寄存器和位以通用格式表示：小写的 “x” 表示端口的序号，而小写的 “n” 代表位的序号。但是在程序里要写完整。例如， PORTB3 表示端口 B 的第 3 位，而本节的通用格式为 PORTxn。物理 I/O 寄存器和位定义列于 P102“I/O 端口寄存器的说明 ” 。

每个端口都有三个 I/O 存储器地址 : 数据寄存器 – PORTx、数据方向寄存器 – DDRx 和端口输入引脚 – PINx。数据寄存器和数据方向寄存器为读 / 写寄存器，而端口输入引脚为只读寄存器。但是需要特别注意的是，对 PINx 寄存器某一位写入逻辑 "1“ 将造成数据寄存器相应位的数据发生 "0“ 与 “1“ 的交替变化。当寄存器 MCUCR 的上拉禁止位 PUD 置位时所有端口引脚的上拉电阻都被禁止。

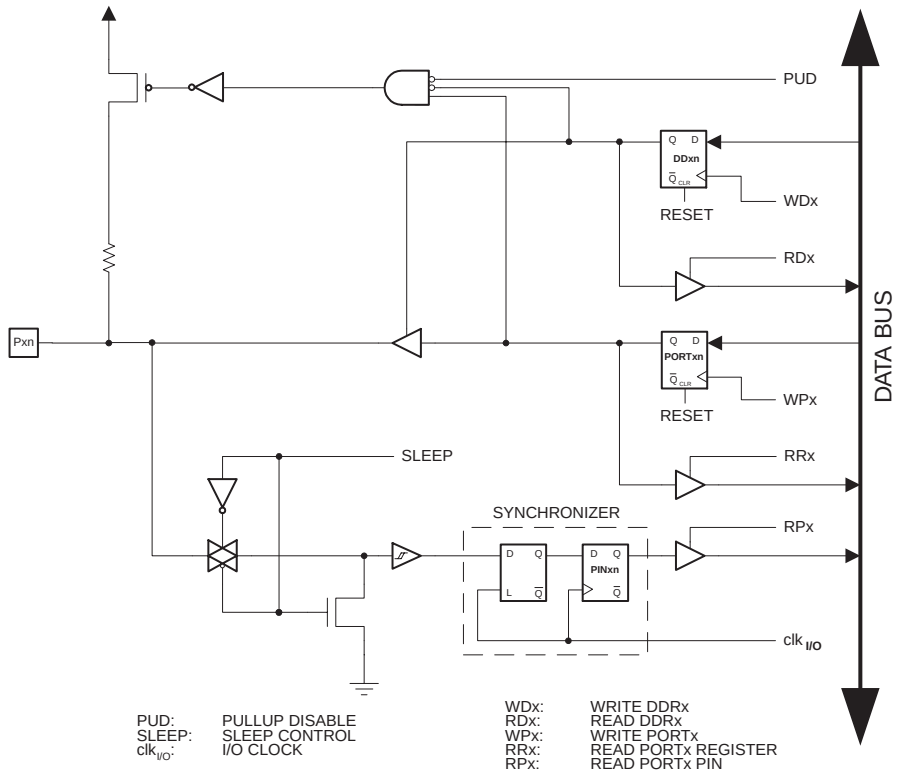
作为通用数字 I/O 时的端口请参见 P87“ 作为通用数字 I/O 的端口 ” 。多数端口引脚是与第二功能复用的，如 P92“ 端口的第二功能 ” 所示。请参见各个模块的具体说明以了解引脚的第二功能。

使能某些引脚的第二功能不会影响其他属于同一端口的引脚用于通用数字 I/O 目的。

作为通用数字 I/O 的端口

端口为具有可选上拉电阻的双向 I/O 端口。Figure54 为一个 I/O 端口引脚的说明。

Figure 54. 通用数字 I/O(1)



Note: 1. WRx, WPx, WDx, RRx, RPx 和 RDx 对于同一端口的所有引脚都是一样的。 $clk_{I/O}$, SLEEP 和 PUD 则对所有的端口都是一样的。

配置引脚

每个端口引脚都具有三个寄存器位 : DDxn、PORTxn 和 PINxn , 如 P102“I/O 端口寄存器的说明 ” 所示。 DDxn 位于 DDRx 寄存器， PORTxn 位于 PORTx 寄存器， PINxn 位于 PINx 寄存器。

DDxn 用来选择引脚的方向。 DDxn 为 "1“ 时， Pxn 配置为输出，否则配置为输入。

引脚配置为输入时，若 PORTxn 为 "1“，上拉电阻将使能。如果需要关闭这个上拉电阻，可以将 PORTxn 清零，或者将这个引脚配置为输出。复位时各引脚为高阻态，即使此时并没有时钟在运行。

引脚配置为输出时，若 PORTxn 为 "1“，引脚输出高电平 ("1“)，否则输出低电平 (“0“)。

在高阻态 (三态)({DDxn, PORTxn} = 0b00) 和输出高电平 ({DDxn, PORTxn} = 0b11) 两种状态之间进行切换时，上拉电阻使能 ({DDxn, PORTxn} = 0b01) 或输出低电平 ({DDxn, PORTxn} = 0b10) 这两种模式必然会有一个发生。通常，上拉电阻使能是完全可以接受的，因为高阻环境并不不区分强高电平输出和上拉输出。如果实际应用环境不允许这样，则可以通过置位 SFIOR 寄存器的 PUD 来禁止所有端口的上拉电阻。

在上拉输入和输出低电平之间切换也有同样的问题。用户必须选择高阻态 ({DDxn, PORTxn} = 0b00) 或输出高电平 ({DDxn, PORTxn} = 0b11) 作为中间步骤。

Table 39 总结了引脚的控制信号。

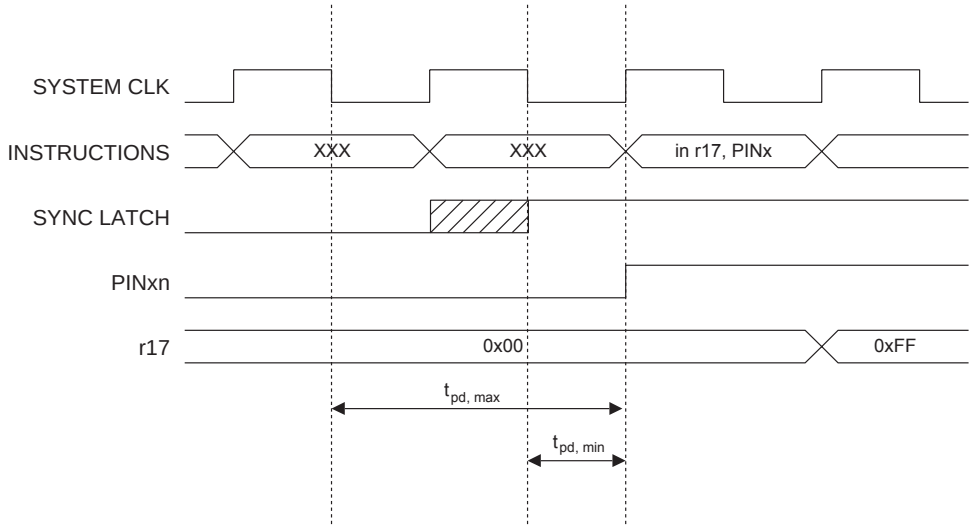
Table 39. 端口引脚配置

DDxn	PORTxn	PUD(位于 MCUCR)	I/O	上拉电阻	说明
0	0	X	输入	No	高阻态 (Hi-Z)
0	1	0	输入	Yes	被外部电路拉低时输出电流
0	1	1	输入	No	高阻态 (Hi-Z)
1	0	X	输出	No	输出低电平 (吸收电流)
1	1	X	输出	No	输出高电平 (源电流)

读取引脚上的数据

不论 DDxn 如何配置，引脚电平的信息都可以通过 PINxn 寄存器来获得。如 Figure54 所示， PINxn 寄存器的各个位与其前面的锁存器组成了一个同步器。这样就可以避免由于引脚电平在内部时钟边沿发生变化而造成的亚稳定。其缺点是引入了延迟。 Figure55 为读取引脚电平时同步器的时序图。最大和最小传输延迟分别为 $t_{pd,max}$ 和 $t_{pd,min}$。

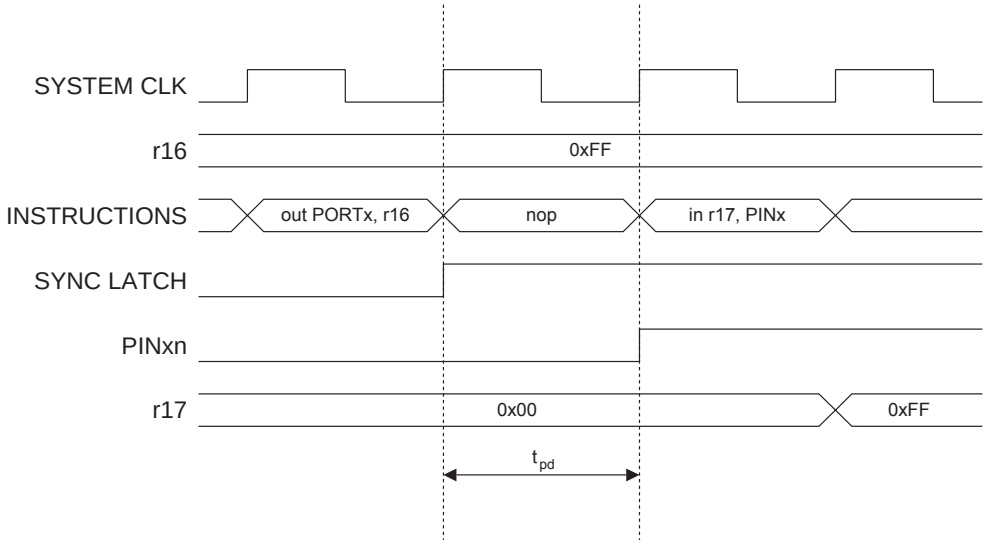
Figure 55. 读取引脚数据时的同步



考虑第一个系统时钟下降沿之后起始的时钟周期。当时钟信号为低时锁存器是关闭的，而时钟信号为高时信号可以自由通过，如图中 SYNC LATCH 信号的阴影区所示。时钟为低时信号即被锁存，然后在紧接着的系统时钟上升沿锁存到PINxn寄存器。如$t_{pd,max}$ 和$t_{pd,min}$ 所示，根据信号施加时间的不同，引脚上信号转换的延迟时间界于 ½ 到 1½ 个系统时钟。

如 Figure56. 所示，读取软件赋予的引脚电平时需要在赋值指令 *out* 和读取指令 *in* 之间至少有一个时钟周期的间隔，如 *nop* 指令。*out* 指令在时钟的上升沿置位 SYNC LATCH 信号。此时同步器的延迟时间 t_{pd} 为一个系统时钟周期。

Figure 56. 读取软件赋予的引脚电平的同步情况



下面的例程演示了如何置位端口 B 的引脚 0 和 1，并使引脚 2 和 3 输出低电平，以及将引脚 4 到 7 设置为输入，并且为引脚 6 和 7 设置上拉电阻。然后程序将各个引脚的数据读回来。如前面讨论的那样，我们在输出和输入语句之间插入了一个 *nop* 指令。

汇编代码例程 (1)

```
...
; 定义上拉电阻和设置高电平输出
; 为端口引脚定义方向
ldi   r16,(1<<PB7)|(1<<PB6)|(1<<PB1)|(1<<PB0)
ldi   r17,(1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0)
out   PORTB,r16
out   DDRB,r17
; 为了同步插入 nop 指令
nop
; 读取端口引脚
in    r16,PINB
...
```

C 代码例程

```
unsigned char i;
...
/* 定义上拉电阻和设置高电平输出 */
/* 为端口引脚定义方向 */
PORTB = (1<<PB7)|(1<<PB6)|(1<<PB1)|(1<<PB0);
DDRB = (1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0);
/* 为了同步插入 nop 指令 */
_NOP();
/* 读取端口引脚 */
i = PINB;
...
```

Note: 1. 在汇编程序里使用了两个暂存器。其目的是为了使整个操作过程的时间最短。

数字输入使能和休眠模式

如 Figure54 所示，数字输入信号 (施密特触发器的输入) 可以钳位到地。图中的 SLEEP 信号由 MCU 休眠控制器在各种掉电模式、省电模式以及 Standby 模式下设置，以防止在输入悬空或模拟输入电平接近 $V_{CC}/2$ 时消耗太多的电流。

引脚作为外部中断输入时 SLEEP 信号无效。但若外部中断没有使能， SLEEP 信号仍然有效。引脚的第二功能使能时 SLEEP 也让位于第二功能，如 P92“ 端口的第二功能 ” 里描述的那样。

如果逻辑高电平 (“1”) 出现在一个被设置为 " 上升沿、下降沿或任何逻辑电平变化都引起中断 " 的外部异步中断引脚上，即使该外部中断未被使能，但从上述休眠模式唤醒时，相应的外部中断标志位仍会被置 "1”。这是因为引脚电平在休眠模式下被钳位到 "0” 电平。唤醒过程造成了引脚电平从 "0” 到 ”1” 的变化。

未连接引脚的处理

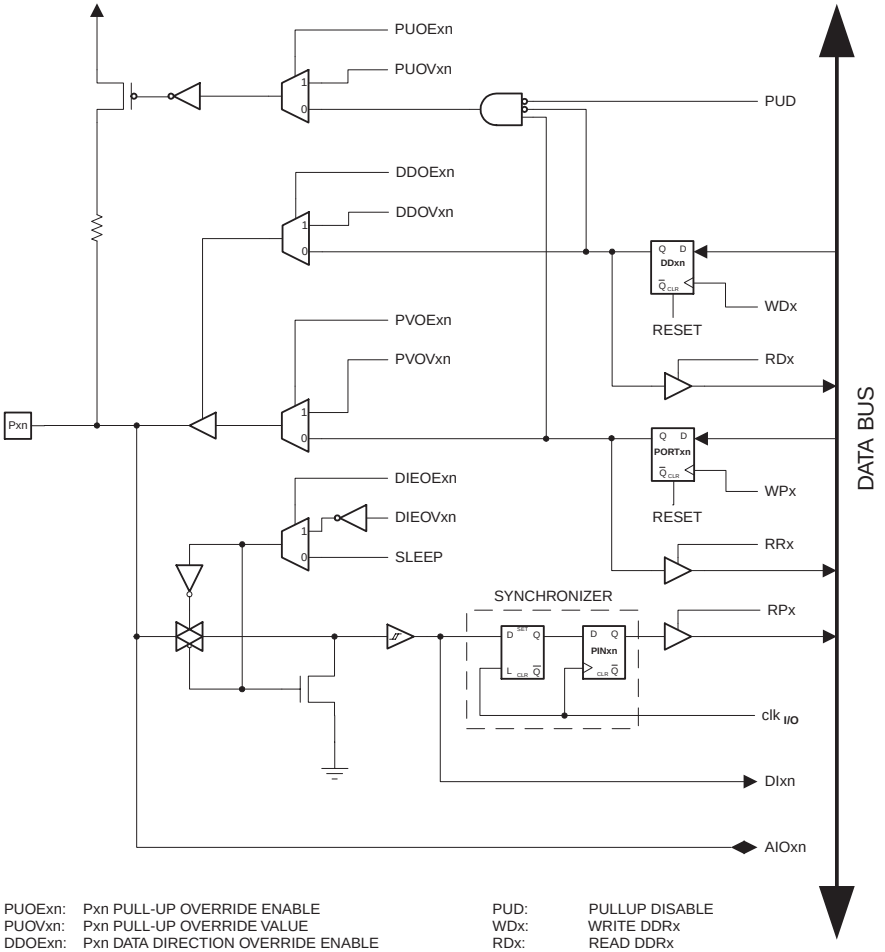
如果有引脚未被使用，建议给这些引脚赋予一个确定电平。虽然如上文所述，在深层休眠模式下大多数数字输入被禁用，但还是需要避免因引脚没有确定的电平而造成悬空引脚在其它数字输入使能模式 (复位、工作模式、空闲模式) 消耗电流。

最简单的保证未用引脚具有确定电平的方法是使能内部上拉电阻。但要注意的是复位时上拉电阻将被禁用。如果复位时的功耗也有严格要求则建议使用外部上拉或下拉电阻。不推荐直接将未用引脚与 V_{CC} 或 GND 连接，因为这样可能会在引脚偶然作为输出时出现冲击电流。

端口的第二功能

除了通用数字 I/O 功能之外，大多数端口引脚都具有第二功能。Figure57 说明了由 Figure54 简化得出的端口引脚控制信号是如何被第二功能取代的。这些被重载的信号不会出现在所有的端口引脚，但本图可以看作是适合于 AVR 系列处理器所有端口引脚的一般说明。

Figure 57. 端口的第二功能 (1)



PUOExn:	Pxn PULL-UP OVERRIDE ENABLE	PUD:	PULLUP DISABLE
PUOVxn:	Pxn PULL-UP OVERRIDE VALUE	WDx:	WRITE DDRx
DDOExn:	Pxn DATA DIRECTION OVERRIDE ENABLE	RDx:	READ DDRx
DDOVxn:	Pxn DATA DIRECTION OVERRIDE VALUE	RRx:	READ PORTx REGISTER
PVOExn:	Pxn PORT VALUE OVERRIDE ENABLE	WPx:	WRITE PORTx
PVOVxn:	Pxn PORT VALUE OVERRIDE VALUE	RPx:	READ PORTx PIN
DIEOExn:	Pxn DIGITAL INPUT-ENABLE OVERRIDE ENABLE	$clk_{I/O}$:	I/O CLOCK
DIEOVxn:	Pxn DIGITAL INPUT-ENABLE OVERRIDE VALUE	DIxn:	DIGITAL INPUT PIN n ON PORTx
SLEEP:	SLEEP CONTROL	AIOxn:	ANALOG INPUT/OUTPUT PIN n ON PORTx

Note: 1. WPx, WDx, RLx, RPx和RDx对于同一个端口的所有引脚都是一样的。$clk_{I/O}$, SLEEP和PUD 则对所有的端口都是一样的。其他信号只对某一个引脚有效。

Table 40 为重载信号的简介。表中没有给出 Figure57 的引脚和端口索引。这些重载信号是由第二功能模块产生的。

Table 40. 第二功能重载信号的一般说明

信号名称	全 称	说 明
PUOE	上拉电阻 重载使能	若此信号置位，上拉电阻使能将受控于 PUOV ；若此信号清零，则 {DDxn, PORTxn, PUD} = 0b010 时上拉电阻使能。
PUOV	上拉电阻 重载值	若 PUOE 置位，则不论 DDxn、PORTxn 和 PUD 寄存器各个位如何配置， PUOV 置位 / 清零时上拉电阻使能 / 禁止
DDOE	数据方向 重载使能	如果此信号置位，则输出驱动使能由 DDOV 控制；若此信号清零，输出驱动使能由 DDxn 寄存器控制。
DDOV	数据方向 重载值	若 DDOE 置位，则 DDOV 置位 / 清零时输出驱动使能 / 禁止，而不管 DDxn 寄存器的设置如何。
PVOE	端口数据 重载使能	如果这个信号置位，且输出驱动使能，端口数据由 PVOV 控制；若 PVOE 清零，且输出驱动使能，端口数据由寄存器 PORTxn 控制。
PVOV	端口数据 重载值	若 PVOE 置位，端口值设置为 PVOV，而不管寄存器 PORTxn 如何设置。
DIEOE	端口信号切换 重载使能	若 PTOE 置位，则端口寄存器位取反。
DIEOV	数字输入 使能覆盖使能	如果这个信号置位，数字输入使能由 DIEOV 控制；若 DIEOE 清零，数字输入使能由 MCU 的状态确定 (正常模式，睡眠模式)。
DI	数字输入	此信号为第二功能的数字输入。在图中，这个信号与施密特触发相连，并且在同步器之前。除非数字输入用作时钟源，否则第二功能模块将使用自己的同步器。
AIO	模拟信号 输入 / 输出	模拟输入 / 输出。信号直接与引脚接点相连，而且可以用作双向端口。

下面的几小节将简单地说明每个端口的第二功能以及相关的信号。具体请参考有关第二功能的说明。

MCU 控制寄存器 - MCUCR

MCU 控制寄存器位包括：

Bit	7	6	5	4	3	2	1	0	
$35 ($55)	–	PUD	SE	SM1	SM0	–	ISC01	ISC00	MCUCR
读 / 写	R	R/W	R/W	R/W	R/W	R	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

- **Bit 6 – PUD: 禁用上拉电阻**

置位时，即使将寄存器 DDxn 和 PORTxn 配置为使能上拉电阻 ({DDxn, PORTxn} = 0b01)，I/O 端口的上拉电阻也被禁止。请参见 P88“ 配置引脚 ”。

端口 A 的第二功能

端口A的第二功能为ADC的模拟输入与模拟比较器，其引脚电平变化中断示于Table 41。若要将端口 A 的部分引脚配置为输出，则当转换进行时，实质上引脚仍作为输出使用。这可能会损坏转换结果。有关 ADC 的内容在 P75“ 模数转换器 ” 中给出。模拟比较器的内容在 P72“ 模拟比较器 ” 中给出。当中断使能且并未被第二功能所屏蔽，则即使引脚 PA7、PA6 及 PA3 置为输出，引脚电平变化中断也可在其上触发，详见 P37“ 引脚变化中断 ”。

Table 41. 端口 A 的第二功能

端口引脚	第二功能
PA7	ADC6 (ADC 输入通道 6) AIN1 (模拟比较器负输入端) PCINT1 (引脚电平变化中断 1)
PA6	ADC5 (ADC 输入通道 5) AIN0 (模拟比较器正输入端) PCINT1 (引脚电平变化中断 1)
PA5	ADC4 (ADC 输入通道 4)
PA4	ADC3 (ADC 输入通道 3)
PA3	AREF (ADC 外部参考) PCINT1 (引脚电平变化中断 1)
PA2	ADC2 (ADC 输入通道 2)
PA1	ADC1 (ADC 输入通道 1)
PA0	ADC0 (ADC 输入通道 0)

Table 42与Table 43 给出了端口A第二功能与P88Figure57 重载信号的对应关系。其中有引脚 PA7、 PA6 与 PA3 数字输入的变化。 PA3 输出及上拉驱动器被覆盖。

- **ADC6/AIN1 端口 – A, Bit 7**

AIN1 : 模拟比较器负输入端及 ADC6 : ADC 输入通道 6。设置该引脚为输入时应关闭内部上拉电阻，以避免数字端功能被模拟比较器或模数转换器的功能妨碍。

PCINT1 : 引脚电平变化中断 1 引脚。当全局中断、引脚电平变化中断使能且第二功能未屏蔽中断，则引脚电平变化中断在该引脚使能。屏蔽的第二功能为模拟比较器。若引脚电平变化中断使能且未被第二功能屏蔽，则在 SLEEP 模式下引脚 PA7 的数字输入使能。

- **ADC5/AIN0 端口 – A, Bit 6**

AIN0 : 模拟比较器正输入端及 ADC5 : ADC 输入通道 5。设置该引脚为输入时应关闭内部上拉电阻，以避免数字端功能被模拟比较器或模数转换器的功能妨碍。

PCINT1 : 引脚电平变化中断 1 引脚。当全局中断、引脚电平变化中断使能且第二功能未屏蔽中断，则引脚电平变化中断在该引脚使能。屏蔽的第二功能为模拟比较器。若引脚电平变化中断使能且未被第二功能屏蔽，则在 SLEEP 模式下引脚 PA7 的数字输入使能。

- **ADC4, ADC3 端口 – A, Bit 5, 4**

ADC4/ADC3 : ADC 输入通道 4、 3。设置该引脚为输入时应关闭内部上拉电阻，以避免数字端功能被模数转换器的功能妨碍。

- **AREF/PCINT1 端口 – A, Bit 3**

AREF：ADC 外部参考电压。当 PA3 引脚作为外部基准或使用在 AREF 引脚连接外部电容 (通过设置 ADMUX 寄存器的 REFS0 为 1) 的内部电压 (2.56V) 时，引脚的上拉电阻与输出驱动器禁用。

PCINT1：引脚电平变化中断 1 引脚。当全局中断、引脚电平变化中断使能且第二功能未屏蔽中断，则引脚电平变化中断在该引脚使能。屏蔽的第二功能为 ADC 的模拟参考电压。若引脚电平变化中断使能且未被第二功能屏蔽，则在 SLEEP 模式下引脚 PA3 的数字输入使能。

Table 42. PA7..PA4 的第二功能重载信号

信号名称	PA7/ADC6/ AIN1/PCINT1	PA6/ADC5/ AIN0/PCINT1	PA5/ADC4	PA4/ADC3
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	0	0	0	0
PVOV	0	0	0	0
DIEOE	PCINT1_ENABLE(1) • ACSR[ACD]	PCINT1_ENABLE(1)• ACSR[ACD]	0	0
DIEOV	1	1	0	0
DI	PCINT1	PCINT1	–	–
AIO	ADC6 输入，AIN1	ADC5 输入，AIN0	ADC4 输入	ADC3 输入

Table 43. PA3..PA0 的第二功能重载信号

信号名称	PA3/AREF/PCINT1	PA2/ADC2	PA1/ADC1	PA0/ADC0
PUOE	ADMUX[REFS0]	0	0	0
PUOV	0	0	0	0
DDOE	ADMUX[REFS0]	0	0	0
DDOV	0	0	0	0
PVOE	0	0	0	0
PVOV	0	0	0	0
DIEOE	PCINT1_ENABLE(1) • ~(2)ADMUX[REFS0]	0	0	0
DIEOV	1	0	0	0
DI	PCINT1	–	–	–
AIO	模拟参考输入	ADC2 输入	ADC1 输入	ADC0 输入

Notes: 1. 注意PCINT1中断只有在全局中断标志使能、GIMSK寄存器中PCIE1标志设置及引脚第二功能禁用时使能，如 P37“ 引脚变化中断 ” 所述。
2. 符号 “~” 表示无操作。

端口 B 的第二功能

端口 B 的第二功能为 ADC、计时、T/C、USI、SPI 编程及引脚电平变化中断。关于 ADC 的说明见 P75“ 模数转换器 ” ；计时见 P6“ 结构综述 ” ；定时器见 P42“ 定时器 / 计时器 ” ； USI 的内容见 P61“ 通用串行接口 – USI” 。当中断使能且并未被第二功能所屏蔽，则即使引脚 PB7 - PB0 设置为输出，引脚电平变化中断也可在其上触发，详见 P37“ 引脚变化中断 ” 。在编程模式下的引脚功能见 P103“ 存储器编程 ” 的说明。端口 B 的第二功能列于 Table 44。

Table 44. 端口 B 的第二功能

端口引脚	第二功能
PB7	ADC10 (ADC 输入通道 10) RESET (外部复位输入) PCINT1 (引脚电平变化中断 1)
PB6	ADC9 (ADC 输入通道 9) INT0 (外部中断 0 输入) T0 (T/C0 外部计数器时钟输入) PCINT1 (引脚电平变化中断 1)
PB5	ADC8 (ADC 输入通道 l 8) XTAL2 (晶振输出) PCINT1 (引脚电平变化中断 1)
PB4	ADC7 (ADC 输入通道 7) XTAL1 (晶振输入) PCINT1 (引脚电平变化中断 1)
PB3	OC1B (T/C1 PWM 输出 B， T/C1 输出比较 B 匹配输出) PCINT0 (引脚电平变化中断 0)
PB2	SCK (USI 时钟输入 / 输出) SCL (USI 外部开集串行时钟) OC1B (反向 T/C1 PWM 输出 B) PCINT0 (引脚电平变化中断 0)
PB1	DO (USI 数据输出) OC1A (T/C1 PWM 输出 A, T/C1 输出比较 A 匹配输出) PCINT0 (引脚电平变化中断 0)
PB0	DI (USI 数据输入) SDA (USI 串行数据) OC1A (反向 T/C1 PWM 输出 A) PCINT0 (引脚电平变化中断 0)

引脚配置如下：

- **ADC10/RESET/PCINT1 – 端口 B, Bit 7**

ADC10 : ADC 输入通道 10。设置该引脚为输入时应关闭内部上拉电阻，以避免数字端功能被模数转换器的功能妨碍。

RESET : 外部复位输入低电平使能，通过对 RSTDISBL 熔丝位写 “1” 实现。当该引脚作为 RESET 引脚时，上拉电阻激活，输出驱动器与数字输入无效。

PCINT1 : 引脚电平变化中断 1 引脚。当全局中断、引脚电平变化中断使能且第二功能未屏蔽中断，则引脚电平变化中断在该引脚使能。屏蔽的第二功能为 RESET。若引脚电平变化中断使能且未被第二功能屏蔽，则在 SLEEP 模式下引脚 PB7 的数字输入使能。

- **ADC9/INT0/T0/PCINT1 – 端口 B, Bit 6**

ADC9：ADC 输入通道 9。设置该引脚为输入时应关闭内部上拉电阻，以避免数字端功能被模数转换器的功能妨碍。

INT0：外部中断源 0：PB6 引脚可通过对全局输入屏蔽寄存器 GIMSK 的 INT0 位置 1 成为外部中断源。

T0：T/C0 外部计数器时钟输入的使能是通过对 T/C0 控制寄存器 TCCR0 的 CS02 与 CS01 位置 1 实现的。

PCINT1：引脚电平变化中断 1 引脚。当全局中断、引脚电平变化中断使能且第二功能未屏蔽中断，则引脚电平变化中断在该引脚使能。屏蔽的第二功能为 INT0 及 T0。若引脚电平变化中断使能且未被第二功能屏蔽，则在 SLEEP 模式下引脚 PB6 的数字输入使能。

- **ADC8/XTAL2/PCINT1 – 端口 B, Bit 5**

ADC8：ADC 输入通道 8。设置该引脚为输入时应关闭内部上拉电阻，以避免数字端功能被模数转换器的功能妨碍。

XTAL2：芯片时钟振荡器引脚 2。作为除内部标定 RC 振荡器、外部时钟及 PLL 时钟外的芯片时钟源的时钟引脚。当作为时钟引脚时，不能作为 I/O 引脚使用。当使用内部标定 RC 振荡器、外部时钟及 PLL 时钟作为芯片时钟源时， PB5 为普通 I/O 引脚。

PCINT1：引脚电平变化中断 1 引脚。当全局中断、引脚电平变化中断使能且第二功能未屏蔽中断，则引脚电平变化中断在该引脚使能。屏蔽的第二功能为 XTAL2 输出。若引脚电平变化中断使能且未被第二功能屏蔽，则在 SLEEP 模式下引脚 PB5 的数字输入使能。

- **ADC7/XTAL1/PCINT1 – 端口 B, Bit 4**

ADC7：ADC 输入通道 7。设置该引脚为输入时应关闭内部上拉电阻，以避免数字端功能被模数转换器的功能妨碍。

XTAL1：芯片时钟振荡器引脚 1。作为除内部标定 RC 振荡器及 PLL 时钟外的芯片时钟源的时钟引脚。当作为时钟引脚时，不能作为 I/O 引脚使用。当使用内部标定 RC 振荡器及 PLL 时钟作为芯片时钟源时， PB4 为普通 I/O 引脚。

PCINT1：引脚电平变化中断 1 引脚。当全局中断、引脚电平变化中断使能且第二功能未屏蔽中断，则引脚电平变化中断在该引脚使能。屏蔽的第二功能为 XTAL1 输入。若引脚电平变化中断使能且未被第二功能屏蔽，则在 SLEEP 模式下引脚 PB4 的数字输入使能。

- **OC1B/PCINT0 – 端口 B, Bit 3**

OC1B：输出比较匹配输出：PB3 可以作为 T/C1 输出比较模块 B 的输出。此时引脚必须配置为输出 (DDB3 设置为 “1”)。 OC1B 引脚也是 PWM 模式的输出引脚。

PCINT0：引脚电平变化中断 0 引脚。当全局中断、引脚电平变化中断使能且第二功能未屏蔽中断，则引脚电平变化中断在该引脚使能。屏蔽的第二功能为 OC1B。若引脚电平变化中断使能且未被第二功能屏蔽，则在 SLEEP 模式下引脚 PB3 的数字输入使能。

- **SCK/SCL/$\overline{\text{OC1B}}$/PCINT0 – 端口 B, Bit 2**

SCK：USI 三线模式下的时钟输入 / 输出。当 SPI 使能该引脚作为输入。在 USI 三线模式下 DDRB2 控制数据方向，主机模式时为输出；从机模式时为输入。

SCL：USI 两线模式下的外部开集串行时钟。当 PORTB2 清零或检测 USI 启动状态时 SCL 引脚拉低； DDRB2 置 1。在 USI 两线模式下上拉电阻禁用。

$\overline{\text{OC1B}}$：反向 T/C1 PWM 输出 B：若 USI 未使能，PB2 引脚可作为 T/C1 PWM 模式下的反向输出。这时要将 DDB2 置 1 。

PCINT1：引脚电平变化中断 0 引脚。当全局中断、引脚电平变化中断使能且第二功能未屏蔽中断，则引脚电平变化中断在该引脚使能。屏蔽的第二功能为 OC1B 与 USI 时钟 SCK/SCL。若引脚电平变化中断使能且未被第二功能屏蔽，则在 SLEEP 模式下引脚 PB2 的数字输入使能。

- **DO/OC1A/PCINT0 – 端口 B, Bit 1**

DO：USI 三线模式下的数据输出。当数据方向位 DDB1 置 1 时，DO 覆盖 PORTB1 值 且被端口驱动。但 PORTB1 位仍然控制上拉电阻，若数据方向为输入且 PORTB1 置 1，使能上拉电阻。

OC1A：：输出比较匹配输出：PB1 可以作为 T/C1 输出比较模块 A 的输出。此时引脚必须配置为输出 (DDB1 设置为 “1”)。 OC1B 引脚也是 PWM 模式定时器功能的输出引脚。

PCINT0：引脚电平变化中断 0 引脚。当全局中断、引脚电平变化中断使能且第二功能未屏蔽中断，则引脚电平变化中断在该引脚使能。屏蔽的第二功能为 OC1A 与 USI 三线模式下的 DO。若引脚电平变化中断使能且未被第二功能屏蔽，则在 SLEEP 模式下引脚 PB1 的数字输入使能。

- **DI/SDA/$\overline{\text{OC1A}}$/PCINT0 – 端口 B, Bit 0**

DI：：USI 三线模式下的数据输入。 USI 三线模式不会覆盖普通端口功能，因此引脚必须设置为输入。

SDA：USI 两线模式下的串行数据。串行数据为双向且使用开集输出。将该引脚设为输出，则使能 SDA。当 PORTB0 或 USI 移位寄存器为 0(DDB0 为 1) 时该引脚被拉低。在 USI 两线模式下上拉电阻禁用。

$\overline{\text{OC1A}}$：反向 T/C1 PWM 输出 A：若未编程或在 USI 时，PB0 引脚可作为 T/C1 PWM 模式下的反向输出。这时要将 DDB0 置 1 。

PCINT0：引脚电平变化中断 0 引脚。当全局中断、引脚电平变化中断使能且第二功能未屏蔽中断，则引脚电平变化中断在该引脚使能。屏蔽的第二功能为 $\overline{\text{OC1A}}$ 与 USI 数据 DI 或 SDA。若引脚电平变化中断使能且未被第二功能屏蔽，则在SLEEP模式下引脚PB0的数字

输入使能。 Table 45 与 Table 46 给出了端口 B 第二功能与 P92“ 端口的第二功能 ” 重载信号的对应关系。

Table 45. PB7..PB4 的第二功能重载信号

信号名称	PB7/ADC10/$\overline{\text{RESET}}$/PCINT1	PB6/ADC9/INT0/TO/PCINT1	PB5/ADC8/XTAL2/PCINT1	PB4/ADC7/XTAL1
PUOE	RSTDSBL(1)	0	~(5)PB5IOENABLE(3)	~PB4IOENABLE(3)
PUOV	1	0	0	0
DDOE	RSTDSBL(1)	0	~PB5IOENABLE(3)	~PB4IOENABLE(3)
DDOV	0	0	0	0
PVOE	0	0	0	0
PVOV	0	0	0	0
DIEOE	PCINT1_ENABLE(2) | RSTDSBL(1)	~T0_EXT_CLOCK (6) • PCINT1_ENABLE(2) | INT0_ENABLE(4)	PCINT1_ENABLE(2) | ~PB5IOENABLE(3)	PCINT_ ENABLE (2) | ~PB4IOENABLE(3) | EXT_CLOCK_ENABLE(7)
DIEOV	PCINT1_ENABLE(2) • ~(5)RSTDSBL(1)	1	PCINT1_ENABLE(2) • PB5IOENABLE(3)	PCINT1_ENABLE(2)• PB4IOENABLE(3) | EXT_CLOCK_ENABLE
DI	PCINT1	INT0, T0, PCINT1	PCINT1	外部时钟 , PCINT1
AIO	ADC10，复位输入	ADC9	ADC8, XTAL2	XTAL1

Notes:
1. RSTDISBL 熔丝位 (低电平激活) 的说明见 P20“ 复位源 ” 。
2. 注意 ,只有当全局中断标志使能 ,GIMSK寄存器中的PCIE1标志位置位且引脚第二功能如P37“引脚变化中断” 中所述禁用时，PCINT1 中断才使能。
3. PB5IOENABLE 和 PB4IOENABLE 由熔丝位 PLLCK 与 CKSEL 给出，如 P26“ 时钟源 ” 中所述。
4. 当全局中断标志使能且 GIMSK 中的 INT0 标志如 P37“ 外部中断 ” 所述设置，将使能外部低电平中断。
5. 符号 “~” 表示无操作。
6. 外部时钟禁用的 T/C0 操作见 P43“8 位 T/C0” 。
7. 外部时钟选择位 PLLCK 与 CKSEL 熔丝位的说明见 P26“ 时钟源 ” 。

Table 46. PB3..PB0 的第二功能重载信号

信号名称	PB3/OC1B/PCINT0	PB2/SCK/SCL/$\overline{OC1B}$/PCINT0	PB1/DO/OC1A/PCINT0	PB0/DI/SDA/$\overline{OC1A}$
PUOE	0	USI_TWO-WIRE(3)	0	USI_TWO-WIRE(3)
PUOV	0	0	0	0
DDOE	0	USI_TWO-WIRE(3)	0	USI_TWO-WIRE(3)
DDOV	0	(USI_SCL_HOLD(4) | ~(8)PORTB2) • DDB2	0	(~SDA | ~PORTB0) • DDB0
PVOE	OC1B_ENABLE(1)	USI_TWO-WIRE(3) • DDB2 | OC1B_ENABLE(1)	USI_THREE-WIRE(3) | OC1A_ENABLE(1)	USI_TWO-WIRE(3)• DDB0 | $\overline{OC1A}$_ENABLE(1)
PVOV	OC1B	~(USI_TWO-WIRE • DDB2) • $\overline{OC1B}$	USI_THREE-WIRE(3) • DO(6) | ~USI_THREE-WIRE • OC1A_ENABLE(1) • OC1A	~(USI_TWO-WIRE• $\overline{DDB0}$) • $\overline{OC1A}$_ENABLE(1) • $\overline{OC1A}$
DIEOE	PCINT0_ENABLE(2) • ~OC1B_ENABLE(1)	~(USI_TWO-WIRE | $\overline{USI\_THREE}$-WIRE | $\overline{OC1B}$_ENABLE) • PCINT0_ENABLE(2) | USI_START_I.ENABLE(5)	~(USI_THREE-WIRE | OC1A_ENABLE) • PCINT0_ENABLE(2)	~(USI_TWO-WIRE(3) | $\overline{USI\_THREE}$-WIRE(3) | $\overline{OC1A}$_ENABLE(1)) • PCINT0_ENABLE(2) | USI_START_I.ENABLE(5)
DIEOV	1	1	1	1
DI	PCINT0	PCINT0, SCL, SCK	PCINT0	PCINT0, SDA
AIO	–	–	–	–

Notes:
1. T/C1 比较匹配输出、 T/C1 PWM 输出 OC1A/OC1B 及 $\overline{OC1A}$/$\overline{OC1B}$ 的使能见 P45“8 位 T/C1” 中的相关说明。
2. 注意 ,只有当全局中断标志使能 ,GIMSK寄存器中的PCIE0标志位置位且引脚第二功能如P37“引脚变化中断” 中所述禁用时，PCINT0 中断才使能。
3. 两线与三线模式的说明见 P61“ 通用串行接口 – USI” 。
4. SCL 说明见 P61“ 通用串行接口 – USI” 。
5. 当全局中断标志使能且 USICR 寄存器中的 USISIE 标志如 P61“ 通用串行接口 – USI” 所述设置， USI 启动中断才会使能。
6. 在 USI 三线模式下 DO 有效，操作说明见 P61“ 通用串行接口 – USI” 。
7. USI 两线模式下的 数据引脚 SDA 与三线模式下的 DI 操作见 P61“ 通用串行接口 – USI” 。
8. 符号 “~” 表示无操作。

I/O 端口寄存器的说明

端口 A 数据寄存器 - PORTA

Bit	7	6	5	4	3	2	1	0	
$1B ($3B)	**PORTA7**	**PORTA6**	**PORTA5**	**PORTA4**	**PORTA3**	**PORTA2**	**PORTA1**	**PORTA0**	**PORTA**
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

端口 A 数据方向寄存器 - DDRA

Bit	7	6	5	4	3	2	1	0	
$1A ($3A)	**DDA7**	**DDA6**	**DDA5**	**DDA4**	**DDA3**	**DDA2**	**DDA1**	**DDA0**	**DDRA**
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

端口 A 输入引脚地址 - PINA

Bit	7	6	5	4	3	2	1	0	
$19 ($39)	**PINA7**	**PINA6**	**PINA5**	**PINA4**	**PINA3**	**PINA2**	**PINA1**	**PINA0**	**PINA**
读 / 写	R	R	R	R	R	R	R	R	
初始值	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

端口 B 数据寄存器 - PORTB

Bit	7	6	5	4	3	2	1	0	
$18 ($38)	**PORTB7**	**PORTB6**	**PORTB5**	**PORTB4**	**PORTB3**	**PORTB2**	**PORTB1**	**PORTB0**	**PORTB**
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

端口 B 数据方向寄存器 - DDRB

Bit	7	6	5	4	3	2	1	0	
$17 ($37)	**DDB7**	**DDB6**	**DDB5**	**DDB4**	**DDB3**	**DDB2**	**DDB1**	**DDB0**	**DDRB**
读 / 写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
初始值	0	0	0	0	0	0	0	0	

端口 B 输入引脚地址 - PINB

Bit	7	6	5	4	3	2	1	0	
$16 ($36)	**PINB7**	**PINB6**	**PINB5**	**PINB4**	**PINB3**	**PINB2**	**PINB1**	**PINB0**	**PINB**
读 / 写	R	R	R	R	R	R	R	R	
初始值	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

存储器编程

程序存储器和数据存储器锁定位

ATtiny26 提供了 2 个锁定位，根据其被编程 (“0”) 还是没有被编程 (“1”) 的情况可以获得 Table 48 中列出的附加性能。锁定位只能通过芯片擦除命令擦写为 “1”。

Table 47. 锁定位字节 (1)

锁定位字节	位号	描述	默认值
	7	–	1 (未编程)
	6	–	1 (未编程)
	5	–	1 (未编程)
	4	–	1 (未编程)
	3	–	1 (未编程)
	2	–	1 (未编程)
LB2	1	锁定位	1 (未编程)
LB1	0	锁定位	1 (未编程)

Note: 1. “1” 表示未编程，“0” 表示被编程。

Table 48. 锁定位保护模式

存储器锁定位			
LB 模式	LB2(2)	LB1(2)	保护类型
1	1	1	没有使能存储器保护特性
2	1	0	在并行和串行编程模式中 Flash 和 EEPROM 的进一步编程被禁止，熔丝位被锁定。(1)
3	0	0	在并行和串行编程模式中 Flash 和 EEPROM 的进一步编程及验证被禁止，锁定位和熔丝位被锁定 (1)

Notes: 1. 在编程熔丝位前先对锁定位编程。
2. “1” 表示未被编程，“0” 表示被编程。

熔丝位

ATtiny26中有两个熔丝位字节。Table 49与Table 50简单地描述了所有熔丝位的功能以及他们是如何映射到熔丝字节的。如果熔丝位被编程则读返回值为 0。

Table 49. 熔丝位高位字节

熔丝位高位字节	位号	描述	默认值
	7	–	1 (未编程)
	6	–	1 (未编程)
	5	–	1 (未编程)
RSTDISBL(2)	4	选择 PB7 为 I/O 引脚还是 RESET 引脚	1 (未编程，PB7 为 RESET 引脚)
SPIEN(1)	3	使能串行编程与数据下载	0 (未编程，SPI 编程使能)
EESAVE	2	执行芯片擦除时 EEPROM 的内容不擦除	1 (未编程， EEPROM 内容不保留)
BODLEVEL	1	掉电检测触发电平	1 (未编程)
BODEN	0	掉电检测器使能	1 (未编程，BOD 禁用)

Notes: 1. 在串行编程模式下 SPIEN 熔丝位不可访问。
2. 当对 RSTDISBL 熔丝位编程，使用并行编程来改变熔丝位或执行下一层程序。

Table 50. 熔丝位低位字节

熔丝位低位字节	位号	描述	默认值
PLLCK	7	内部时钟使用 PLL	1 (未编程)
CKOPT(3)	6	振荡器选项	1 (未编程)
SUT1	5	选择起动时间	1 (未编程)(1)
SUT0	4	选择起动时间	0 (已编程)(1)
CKSEL3	3	选择时钟源	0 (已编程)(2)
CKSEL2	2	选择时钟源	0 (已编程)(2)
CKSEL1	1	选择时钟源	0 (已编程)(2)
CKSEL0	0	选择时钟源	1 (已编程)(2)

Notes: 1. 对于默认时钟源，SUT1..0 的默认值导致最大的起动时间。详细内容见 P29Table 13。
2. CKSEL3..0的默认设置导致了片内RC振荡器运行于1MHz。详细内容见P25Table 4 。
3. CKOPT 熔丝位的功能依靠 CKSEL 位的设置，见 P24“ 系统时钟及时钟选项 ” 。

熔丝位的状态不受芯片擦除命令的影响。如果锁定位 1(LB1) 被编程则熔丝位被锁定。在编程锁定位前先编程熔丝位。

熔丝位的锁存

器件进入编程模式时熔丝位的值被锁存。其间熔丝位的改变不会生效，直到器件退出编程模式。不过这不适用于 EESAVE 熔丝位。它一旦被编程立即起作用。在正常模式中器件上电时熔丝位也被锁存。

标识字节

所有的 Atmel 微控制器都具有一个三字节的标识代码用来区分器件型号。这个代码可以通过串行和并行模式读取，也可以在芯片被锁定时读取。这三个字节分别存储于三个独立的地址空间。

对于 ATtiny26 这三个标志字节是：

1. $000: $1E (表示由 Atmel 公司生产)。
2. $001: $91 (表示芯片包含 2KB Flash)。
3. $002: $09 (当 $001 字节的内容为 $91 时表示这是 ATtiny26)。

校准字节

ATtiny26为内部RC振荡器保存四个不同的校准字节。这些字节位于标识地址空间0x000、0x0001、0x0002 和 0x0003 的高位，分别代表 1、2、4 和 8 MHz。在复位期间，1 MHz 的值被自动写入 OSCCAL 寄存器，若使用其他频率则需手动载入标定值，详见 P30“ 振荡器标定寄存器 – OSCCAL” 。

并行编程参数，引脚映射及命令

这部分描述了如何对 ATtiny26 的 Flash 程序存储器， EEPROM 数据存储器，存储锁定位及熔丝位进行并行编程和校验。除非另有说明，脉冲宽度至少为 250 ns。

信号名称

在这一节 ATtiny26 的相关引脚以并行编程信号的名称进行引用，如 Figure58 和 Table 51 所示。表中没有描述的引脚沿用原来的称谓。

XA1/XA0 决定了给 XTAL1 引脚一个正脉冲时所执行的操作。具体编码请见 Table 53。

给 $\overline{WR}$ 或 $\overline{OE}$ 输入脉冲时所加载的命令决定了要执行的操作。具体命令请参见 Table 54。

Figure 58. 并行编程

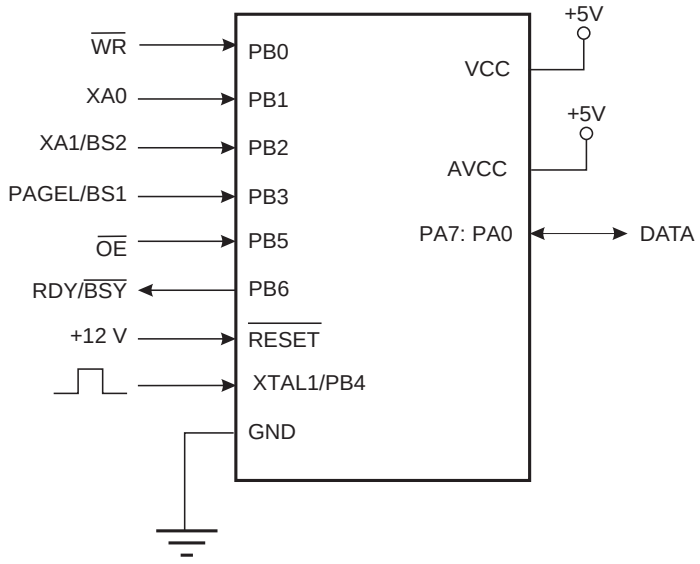


Table 51. 引脚名称映射

编程模式信号的名称	引脚名称	I/O	功能
$\overline{WR}$	PB0	I	写脉冲 (低电平有效)
XA0	PB1	I	XTAL 动作位 0
XA1/BS2(1)	PB2	I	XTAL动作位1 字节选择 2 (“0” 选择低位字节 , “1” 选择第二高位字节)
PAGEL/BS1(1)	PB3	I	加载程序存储器和 EEPROM 数据页字节选择 1 (“0” 选择低位字节 , “1” 选择高位字节)
$\overline{OE}$	PB5	I	输出使能 (低电平有效)
RDY/$\overline{BSY}$	PB6	O	0: 设备忙于编程 , 1: 设备等待新的命令。
DATA	PA7:0	I/O	双向数据总线 ($\overline{OE}$ 为低时输出)

Note: 1. 该引脚在两个不同的控制信号中使用。在休眠的说明中 , 一般只提一个信号例如 , “ 给 BS1 一个正脉冲 ” 实际是 “ 给 PAGEL/BS1 一个正脉冲 ”。 .

Table 52. 进入编程模式所需要的引脚数据

引脚	符号	数值
PAGEL/BS1	Prog_enable[3]	0
XA1/BS2	Prog_enable[2]	0
XA0	Prog_enable[1]	0
$\overline{WR}$	Prog_enable[0]	0

Table 53. XA1 和 XA0 的编码 (1)

XA1	XA0	给 XTAL1 施加脉冲激发的动作
0	0	加载 Flash 或 EEPROM 地址 (通过 BS1 确定是高位还是低位字节)
0	1	加载数据 (通过 BS1 决定是高位还是低位 Flash 数据字节)
1	0	加载命令
1	1	无操作，空闲

Note: 1. [XA1, XA0] = 0b11 为 “ 无操作，空闲 ”。XTAL1 无脉冲时，命令、地址及数据寄存器保持不变。也就是说，尽管 BS2 与 XA1 复用，但由如下所示的方法使用 BS2 没有任何问题。只有当读熔丝位 ($\overline{OE}$ 为低) 且 XTAL1 无脉冲时， BS2 不确定。

Table 54. 命令字节编码

Command Byte	Command Executed
1000 0000	芯片擦除
0100 0000	写熔丝位
0010 0000	写锁定位
0001 0000	写 Flash
0001 0001	写 EEPROM
0000 1000	读标识字节和校准字节
0000 0100	读熔丝位和锁定位
0000 0010	读 Flash
0000 0011	读 EEPROM

Table 55. 一页包含的字和 Flash 中的页数

Flash 大小	页大小	PCWORD	页号	PCPAGE	PCMSB
1K 字 (2K 字节)	16 字	PC[3:0]	64	PC[9:4]	9

Table 56. 一页包含的字和 EEPROM 中的页数

EEPROM 大小	页大小	PCWORD	页数	PCPAGE	EEAMSB
128 字节	4 字节	EEA[1:0]	32	EEA[7:0]	7

并行编程

进入编程模式

通过下面的算法进入并行编程模式：

1. 在 V_{CC} 及 GND 之间提供 4.5 - 5.5V 的电压，且至少等待 100 µs。
2. 将 $\overline{\text{RESET}}$ 拉低，并至少改变 XTAL1 电平 6 次
3. 将 P102Table 52 中列出的的 Prog_enable 引脚置为 "0000"，并等待至少 100 ns
4. 给 $\overline{\text{RESET}}$ 提供 11.5 - 12.5V 的电压。在向 $\overline{\text{RESET}}$ 提供 +12V 电压后的 100 ns 内，Prog_enable 引脚的任何行为都会导致芯片无法进入编程模式

注意，如果 $\overline{\text{RESET}}$ 引脚由 RSTDISBL 熔丝位编程禁用，它可能将无法按上面所提到的算法进行。同样，当使用外部晶振或外部 RC 结构时，也无法按上面的算法进行，因为无法提供合格的 XTAL1 脉冲。在这种情况下，按下面给出的算法进行计算：

1. 将 P102Table 52 中列出的的 Prog_enable 引脚置为 "0000"
2. 在 V_{CC} 及 GND 之间提供 4.5 - 5.5V 的电压，同时给 $\overline{\text{RESET}}$ 提供 11.5 - 12.5V 的电压。
3. 等待 100 ns
4. 重新对熔丝位编程以保证选择外部时钟作为时钟源 (CKSEL3:0 = 0b0000) 并且 $\overline{\text{RESET}}$ 引脚被激活 (RSTDISBL 未编程)。若锁定位被编程，则在改变熔丝位前必须执行芯片擦除命令。
5. 通过对芯片掉电或将 $\overline{\text{RESET}}$ 引脚改为 0b0 来退出编程模式。
6. 用上面给出的算法进入编程模式。

高效编程的几点考虑

在编程过程中，加载的命令及地址保持不变。为了实现高效的编程应考虑以下因素：

- 对多个存储单元进行读或写操作时，命令仅需加载一次
- 当需要写入的数据为 $FF 时可以跳过，因为这就是执行全片擦除命令后 Flash 及 EEPROM(除非 EESAVE 熔丝位被编程) 的内容。
- 只有在编程或读取 Flash 及 EEPROM 中新的 256 字时才需要用到地址高位字节。在读标识字节时也需考虑这一点。

芯片擦除

芯片擦除操作会擦除 Flash 及 EEPROM(1) 存储器以及锁定位。程序存储器没有擦除结束之前锁定位不会被擦除。全片擦除不影响熔丝位。芯片擦除命令必须在编程 Flash 与 / 或 EEPROM 之前完成。

Note: 1. 如果 EESAVE 熔丝位被编程，那么在芯片擦除时 EEPRPOM 不受影响。

加载 " 芯片擦除 " 命令的过程：

1. 将 XA1、 XA0 置为 10 以启动命令加载
2. 将 BS1 置为 0
3. DATA 赋值为 “1000 0000”。这是芯片擦除命令
4. 给 XTAL1 提供一个正脉冲，进行命令加载
5. 给 $\overline{WR}$ 提供一个负脉冲，启动芯片擦除。 RDY/$\overline{BSY}$ 变低
6. 等待 RDY/$\overline{BSY}$ 变高，然后才能加载新的命令

对 Flash 进行编程

Flash是以页的形式组织起来的，如P103Table 55所示。编程Flash时，程序数据被锁存到页缓冲区中。这样一整页的程序数据可以同时得到编程。下面的步骤描述了如何对 Flash 进行编程：

A、加载 " 写 Flash" 命令

1. 将 XA1、 XA0 置为 "10"，启动命令加载
2. 将 BS1 置 0
3. DATA 赋值为 “0001 0000”，这是写 Flash 命令
4. 给 XTAL1 提供一个正脉冲以加载命令

B、加载地址低位字节

1. 将 XA1、 XA0 置为 "00"，启动地址加载
2. 将 BS1 置 0，选择低位地址
3. DATA 赋值为地址低位字节 ($00 - $FF)
4. 给 XTAL1 提供一个正脉冲，加载地址低位字节

C、加载数据低位字节

1. 将 XA1、 XA0 置为 “01“，启动数据加载
2. DATA 赋值为数据低位字节 ($00 - $FF)
3. 给 XTAL1 提供一个正脉冲，加载数据字节

D、加载数据高位字节

1. 将 BS1 置为 “1“，选择数据高位字节
2. 将 XA1、 XA0 置为 “01“，启动数据加载
3. DATA 赋值为数据高位字节 ($00 - $FF)
4. 给 XTAL1 提供一个正脉冲，进行数据字节加载

E. 重复 B 到 D 操作，直到整个缓冲区填满或此页中所有的数据都已加载

地址信息中的低位用于页内寻址，高位用于 FLASH 页的寻址，详见 P106Figure59 。如果页内寻址少于 8 位 (页地址 < 256) ，那么进行页写操作时地址低字节中的高位用于页寻址。

F、加载地址高位字节

1. 将 XA1、 XA0 置为 “00“，启动地址加载操作
2. 将 BS1 置为 “1“，选择高位地址
3. DATA 赋值为地址高位字节 ($00 - $FF)
4. 给 XTAL1 提供一个正脉冲，加载地址高位字节

G. 编程一页数据

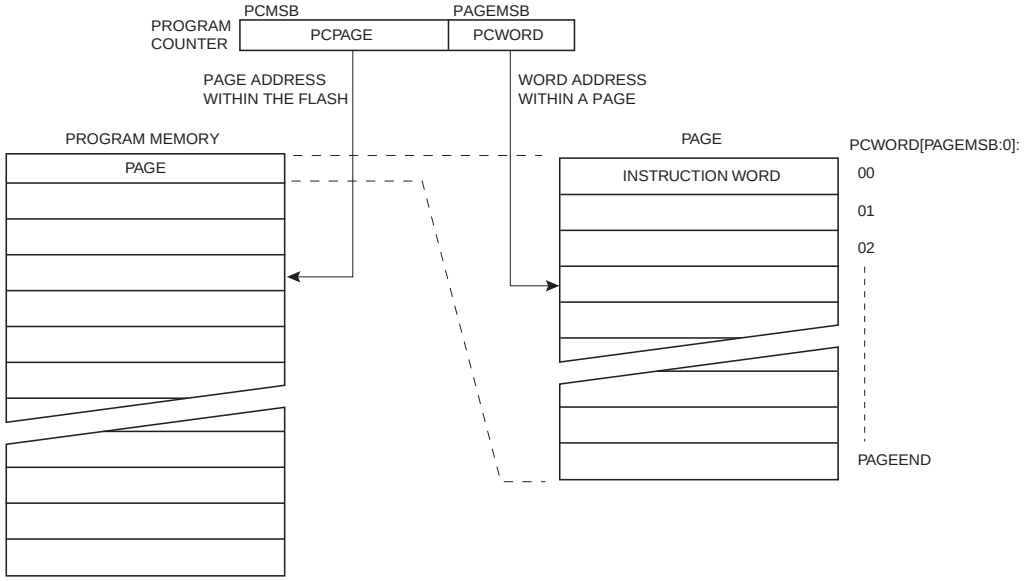
1. 设置 BS1 为 “0”。
2. 给 $\overline{WR}$ 提供一个负脉冲，对整页数据进行编程， RDY/$\overline{BSY}$ 变低
3. 等待 RDY/$\overline{BSY}$ 变高 (见 Figure60 的信号波形)

H、重复 B 到 G 的操作，直到整个 Flash 编程结束或者所有的数据都被编程

I、结束页编程

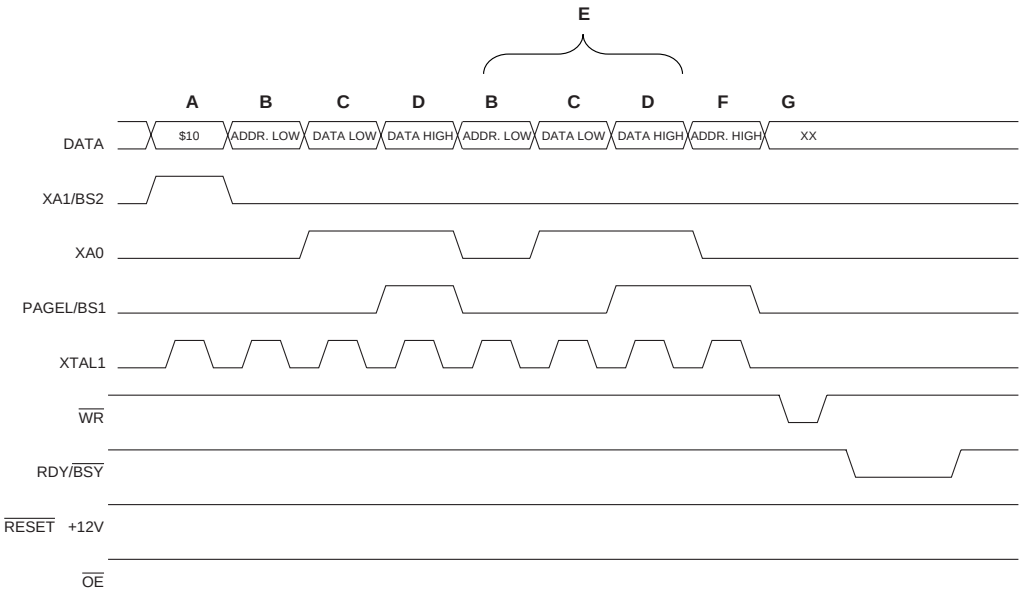
1. 将 XA1、 XA0 置为 “10“，启动命令加载操作
2. DATA 赋值为 “0000 0000“，这是不操作指令
3. 给 XTAL1 提供一个正脉冲，加载命令，内部写信号复位

Figure 59. 对以页为组织单位的 Flash 进行寻址 (1)



Note: 1. PCPAGE 及 PCWORD 列于 P103Table 55。

Figure 60. Flash 编程波形 (1)



Note: 1. 不用考虑“ XX”，各个大写字母对应于前面描述的 Flash 编程阶段。

对 EEPROM 进行编程

如 P103Table 56 所示， EEPROM 也以页为单位。编程 EEPROM 时，编程数据锁存于页缓冲区中。这样可以同时对一页数据进行编程。 EEPROM 数据存储器编程算法如下 (命令、地址及数据加载的细节请参见 P109“ 对 Flash 进行编程 ”) :

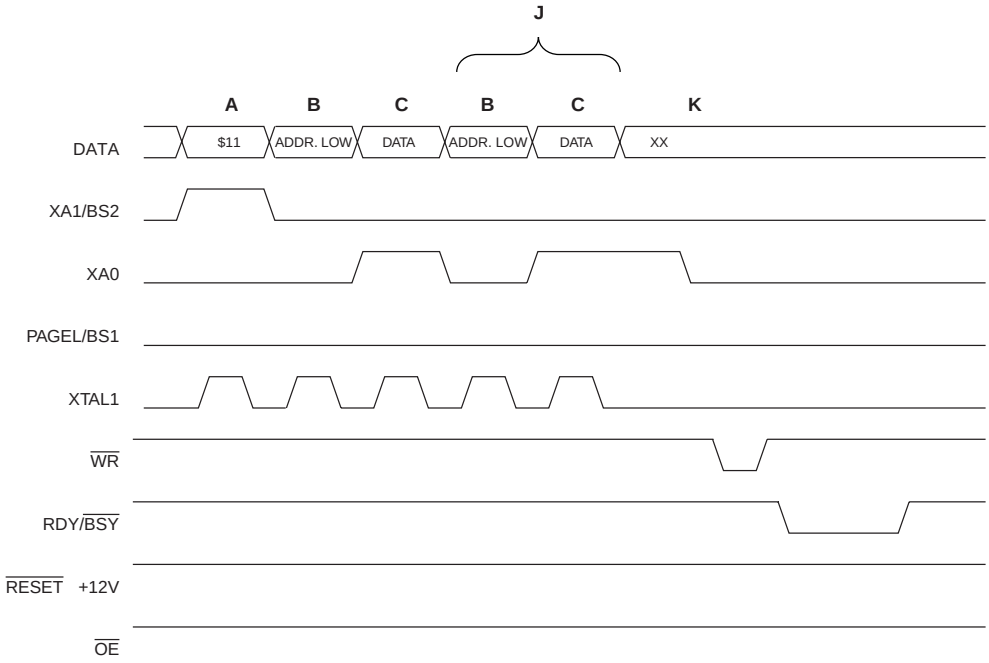
1. A：加载命令 “0001 0001”
2. B：加载地址低位字节 ($00 - $FF)
3. C：加载数据 ($00 - $FF)

J：重复步骤 2 到 3，直到整个缓冲区填满

K：对 EEPROM 页进行编程

1. 将 BS 置 “0“
2. 给 $\overline{WR}$ 提供一个负脉冲，开始对 EEPROM 页进行编程， RDY/$\overline{BSY}$ 变低
3. 等到 RDY/$\overline{BSY}$ 变高再对下一页进行编程 (信号波形见 Figure61)

Figure 61. EEPROM 编程波形



读取 Flash 的内容

读 Flash 存储器的过程如下 (命令及地址加载细节见 P109“ 对 Flash 进行编程 ”) :

1. A：加载命令 “0000 0010”
2. F：加载地址高位字节 ($00 - $03)
3. B：加载地址低位字节 ($00 - $FF)
4. 将 $\overline{OE}$ 置 “0“， BS1 置 “0“，然后从 DATA 读出 Flash 字的低位字节
5. 将 BS 置 “1“，然后从 DATA 读出 Flash 字的高位字节
6. 将 $\overline{OE}$ 置 “1“

读取 EEPROM 的内容

读 EEPROM 存储器的步骤如下 (命令及地址加载细节见 P109“ 对 Flash 进行编程 ”) :

1. A：加载命令 “0000 0011”。
2. B：加载地址低位字节 ($00 - $FF)
3. 将 $\overline{OE}$ 置 “0“， BS1 置 “0“，然后从 DATA 读出 EEPROM 数据字节
4. 将 $\overline{OE}$ 置 “1“

对熔丝低位进行编程

对熔丝低位的编程步骤如下 (命令及数据装在细节见 P109“ 对 Flash 进行编程 ”) :

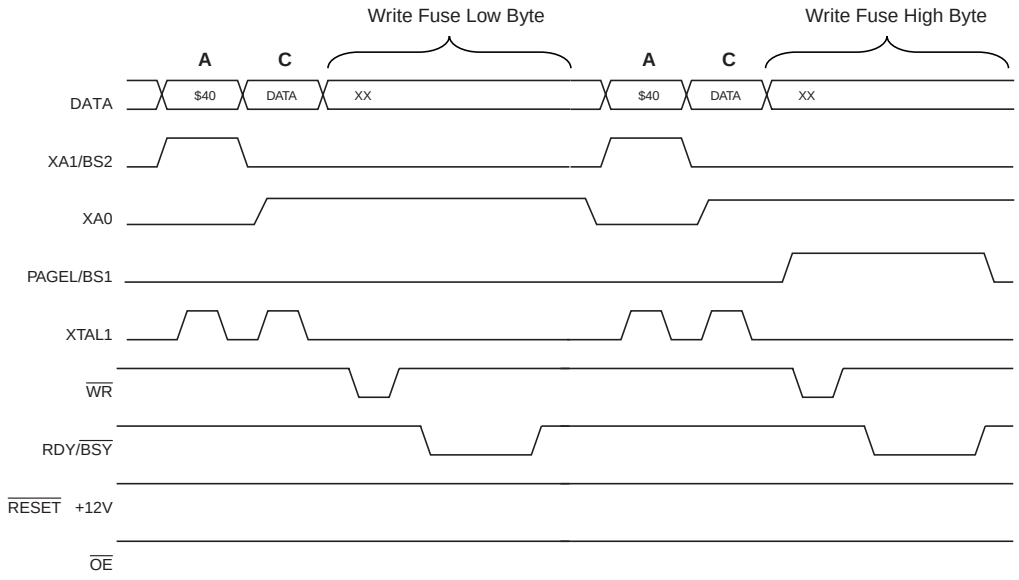
1. A：加载命令 “0100 0000”
2. C：加载数据低字节，若某一位为 0 表示需要进行编程，否则需要擦除
3. Set BS1 and BS2 to “0”.
4. 给 $\overline{WR}$ 提供一个负脉冲，并等待 RDY/$\overline{BSY}$ 变高

对熔丝高位进行编程

对熔丝高位的编程步骤如下 (命令及数据装在细节见 P109“ 对 Flash 进行编程 ”) :

1. A：加载命令 “0100 0000”
2. C：加载数据高字节，若某一位为 0 表示需要进行编程，否则需要擦除
3. 将 BS1 置 “1“、 BS2 置 “0“，选择高位数据字节
4. 给 $\overline{WR}$ 提供一个负脉冲并等待 RDY/$\overline{BSY}$ 变高
5. 将 BS1 置 “0“，选择低位字节

Figure 62. 熔丝位编程波形



锁定位编程

锁定位编程步骤如下 (命令及数据装在细节见 P109“ 对 Flash 进行编程 ”) :

1. A：加载命令 “0010 0000”
2. C.：加载数据低字节，位 n 为 0 表示对此锁定位编程。
3. 给 $\overline{WR}$ 提供一个负脉冲并等待 RDY/$\overline{BSY}$ 变高

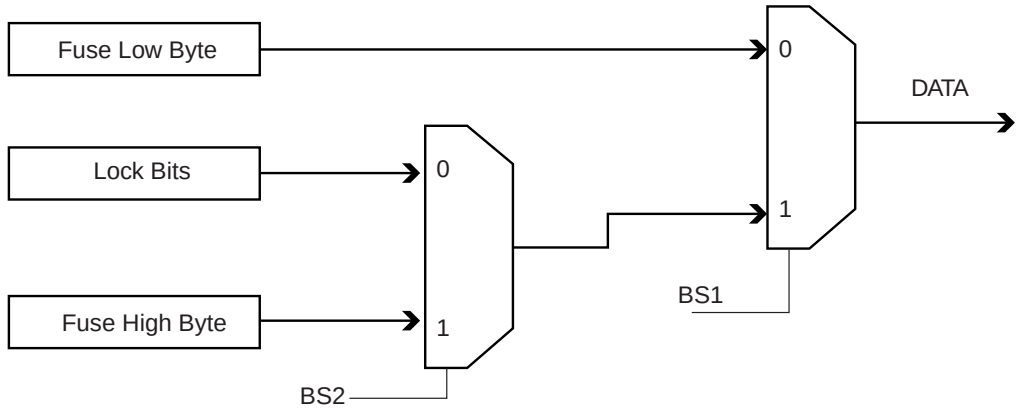
锁定位只能通过芯片擦除命令来清除。

读取熔丝位及锁定位

读取熔丝位及锁定位的步骤如下 (命令加载细节见 P109“ 对 Flash 进行编程 ”) :

1. A：加载命令 “0000 0100”
2. 将 $\overline{OE}$、BS2 和 BS1 置 “0“，然后从 DATA 读取熔丝低位的状态 (“0” 表示已编程)
3. 将 $\overline{OE}$ 置 “0” ,BS2 和 BS1 置 “1” ,然后从 DATA 读取熔丝高位的状态 (“0” 表示已编程)
4. 将 $\overline{OE}$ 和 BS2 置 “0” ,BS1 置 “1” ,然后从 DATA 读取锁定位的状态 (“0” 表示已编程)
5. 将 $\overline{OE}$ 置 “1”

Figure 63. 读操作过程中 BS1、 BS2 与熔丝位及锁定位的对应关系



读取标识字节

读取标识字节的算法如下 (命令及地址加载参考 P109“ 对 Flash 进行编程 ”) :

1. A：加载命令 “0000 1000”
2. B：加载地址低字节 $00 - $02
3. 将 $\overline{OE}$ 与 BS1 置 “0”，然后从 DATA 读取标识字节
4. 将 $\overline{OE}$ 置 “1”

读取校准字节

读取校准字节的算法如下 (命令及地址加载参考 P109“ 对 Flash 进行编程 ”) :

1. A：加载命令 “0000 1000”
2. B：加载地址低字节
3. 将 $\overline{OE}$ 置 “0”，BS1 置 “1”，然后从 DATA 读取校准字节
4. 将 $\overline{OE}$ 置 “1”

并行编程特性

Figure 64. 并行编程时序，包括一些常规的时序要求

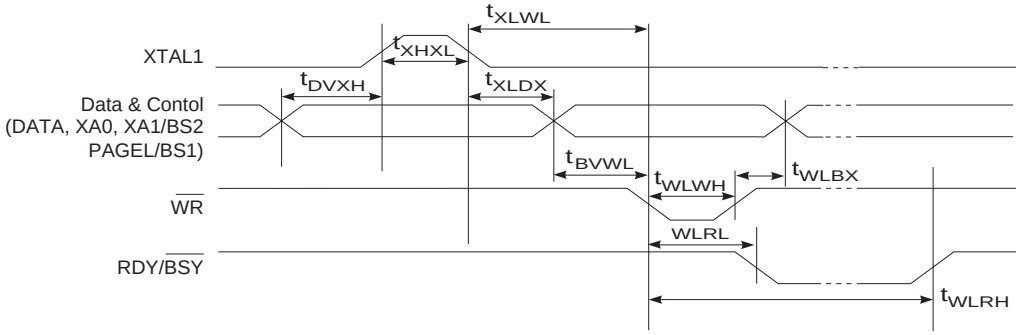
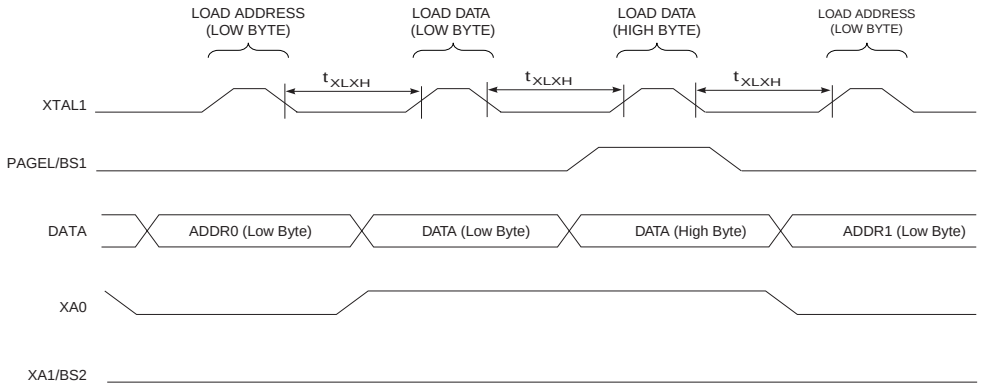
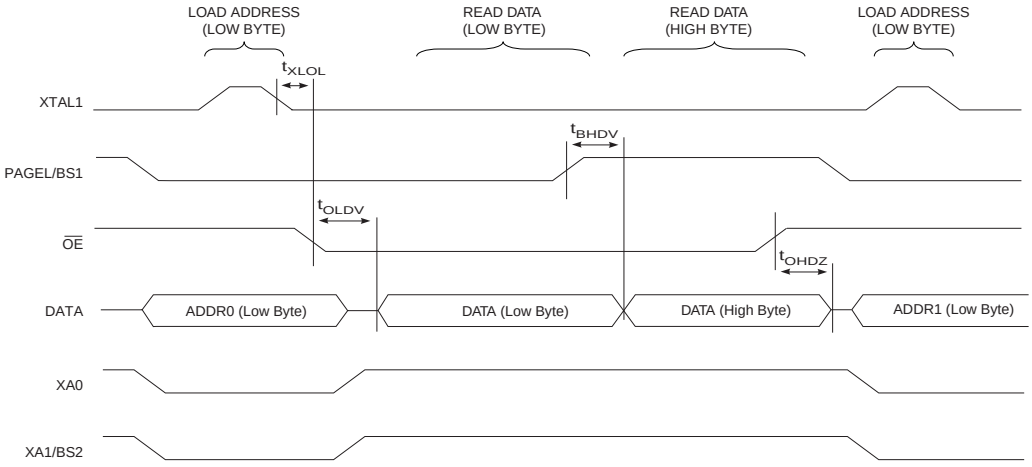


Figure 65. 并行编程时序，有时序要求的加载序列 (1)



Note: 1. Figure64 给出的时序要求 (t_{DVXH}、 t_{XHXL} 及 t_{XLDX}) 也适用于加载操作。

Figure 66. 并行编程时序，有时序要求的读序列 (同一页)(1)



Note: 1. Figure64 给出的时序要求 (即 t_{DVXH}、 t_{XHXL} 及 t_{XLDX}) 也适用于读操作。

Table 57. 并行编程参数 V_{CC} = 5V ± 10%

符号	参数	最小值	典型值	最大值	单位
V_{PP}	编程使能电压	11.5		12.5	V
I_{PP}	编程使能电流			250	µA
t_{DVXH}	在 XTAL1 为高之前数据及控制有效	67			ns
t_{XLXH}	从 XTAL1 低到 XTAL1 高	200			ns
t_{XHXL}	XTAL1 为高时的脉宽	150			ns
t_{XLDX}	XTAL1 为低之后数据及控制保持	67			ns
t_{XLWL}	从 XTAL1 低到 $\overline{WR}$ 低	0			ns
t_{WLBX}	$\overline{WR}$ 为低之后 BS2/1 保持	67			ns
t_{BVWL}	$\overline{WR}$ 为低 BS1 有效	67			ns
t_{WLWH}	$\overline{WR}$ 为低时的脉宽	150			ns
t_{WLRL}	从 $\overline{WR}$ 低到 RDY/$\overline{BSY}$ 为低	0		1	µs
t_{WLRH}	从 $\overline{WR}$ 低到 RDY/$\overline{BSY}$ 为高 (1)	3.7		4.5	ms
t_{WLRH_CE}	从 $\overline{WR}$ 低到 RDY/$\overline{BSY}$ 为高，芯片擦除操作 (2)	7.5		9	ms
t_{XLOL}	从 XTAL1 低到 $\overline{OE}$ 为低	0			ns
t_{BVDV}	BS1 有效至 DATA 有效	0		250	ns
t_{OLDV}	从 $\overline{OE}$ 低到 DATA 有效			250	ns
t_{OHDZ}	从 $\overline{OE}$ 高到 DATA 为三态			250	ns

Notes: 1. 在进行 Flash、 EEPROM、熔丝位及锁定位写操作时 t_{WLRH} 有效。
2. 在执行芯片擦除操作时 t_{WLRH_CE} 有效。

串行下载

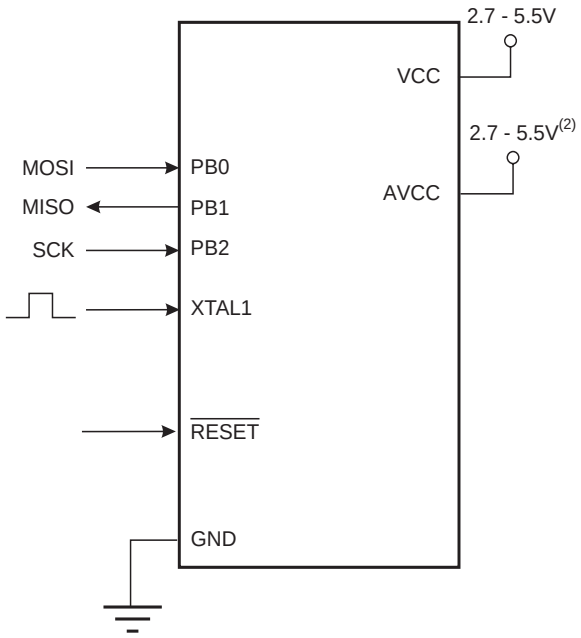
当 $\overline{\text{RESET}}$ 为低电平时，可以通过串行 SPI 总线对 Flash 及 EEPROM 进行编程。串行接口包括 SCK、 MOSI(输入) 及 MISO(输出)。 $\overline{\text{RESET}}$ 为低之后，应在执行编程 / 擦除操作之前执行编程允许指令。P113Table 58 列出了 SPI 编程所需引脚的映射。不是所有的器件都使用 SPI 引脚专用于内部 SPI 接口。注意，在说明串行下载时，MOSI 与 MISO 分别用来说明串行数据的输入与输出。

串行编程引脚映射

Table 58. 串行编程引脚映射

符号	引脚	I/O	说明
MOSI	PB0	I	串行数据输入
MISO	PB1	O	串行数据输出
SCK	PB2	I	串行时钟

Figure 67. 串行编程及校验 (1)



Notes: 1. 如果芯片由片内振荡器提供时钟，那么就不用在 XTAL1 引脚上连接时钟源。
2. V_{CC} -0.3V < AVCC < V_{CC} +0.3V，但是 AVCC 必须在 2.7 - 5.5V 范围内。

编程 EEPROM 时， MCU 在自定时的编程操作中会插入一个自动擦除周期，从而无需执行芯片擦除命令。芯片擦除操作将程序存储器及 EEPROM 的内容都擦除为 $FF。

时钟通过 CKSEL 熔丝位确定。串行时钟 (SCK) 的最小低电平时间和最小高电平时间要满足如下要求：

低 :> f_{ck} < 12 MHz 时为 2 个 CPU 时钟周期，f_{ck} >= 12 MHz 时为 3 个 CPU 时钟周期

高 :> f_{ck} < 12 MHz 时为 2 个 CPU 时钟周期，f_{ck} >= 12 MHz 时为 3 个 CPU 时钟周期

SPI 串行编程算法

向 ATtiny26 串行写入数据时，数据在 SCK 的上升沿得以锁定。

从 ATtiny26 读取数据时，数据在 SCK 的下降沿输出。时序细节见 Figure68、Figure69 和 Table 69。

在串行编程模式下对 ATtiny26 进行编程及校验时，应遵循以下的步骤 (见 Table 60 中的 4 字节指令格式) :

1. 上电顺序 :
   在 RESET 及 SCK 为 “0” 时，向 V_{CC} 及 GND 供电。在一些系统中，编程器不能保证在上电时 SCK 保持为低。在这种情况下， SCK 拉低之后应在 RESET 加一正脉冲，而且这个脉冲至少要维持 2 个 CPU 时钟周期。
2. 上电之后等待至少 20 ms ,然后向 MOSI 引脚输入串行编程使能指令以使能串行编程。
3. 通信不同步将造成串行编程指令不工作。同步之后，在发送编程使能指令的第三个字节时，第二个字节的内容 ($53) 将被反馈回来。不论反馈的内容正确与否，指令的 4 个字节必须全部传输。如果 $53 未被反馈，则需要向 RESET 提供一个正脉冲以开始新的编程使能指令。
4. Flash 的编程以一次一页的方式进行。页的大小见 P103Table 55。在执行加载程序存储页指令时，通过 4 LSB 的地址信息，数据以字节为单位加载到存储页。为保证加载的正确性，应先向给定地址传送数据低字节，之后是高字节。程序存储页通过地址的高 8 位以及写程序存储器页指令获得数据。如果不使用查询的方式，那么在操作下一页数据之前应等待至少 t_{WD_FLASH} 的时间 (见 Table 59)。在 Flash 写操作完成之前访问串行编程接口会导致编程错误。
5. 提供了地址及数据信息之后，适合的写指令将以字节为单位对 EEPROM 编程。EEPROM 存储单元总是在写入新数据之前自动擦除。如果不使用查询的方式，那么在操作下一页数据之前应等待至少 t_{WD_EEPROM} 的时间 (见 Table 59)。对于全片擦除之后的芯片，数据为 $FF 的不需要编程。
6. 可通过读指令来校验任何一个存储单元的内容。数据从串行输出口 MISO 输出。
7. 编程结束后可以将 RESET 拉高开始正常操作。
8. 下电序列 (如果需要) :
   将 RESET 置 “1”。
   切断 V_{CC}。

Flash 数据查询

当 Flash 正处于某一页的编程状态时，读取此页中的内容将得到 $FF。编程结束后，被编程的数据即可以正确读出。通过这种方法可以确定何时可以写下一页。由于整个页是同时编程的，这一页中的任何一个地址都可以用来查询。 Flash 数据查询不适用于数据 $FF。因此，在编程 $FF 时，用户至少要等待 t_{WD_FLASH} 才能进行下一页的编程。由于全片擦除将所有的单元擦为 $FF，所以编程数据为 0xFF 时可以跳过这个操作。 t_{WD_FLASH} 的值见 Table 59。

EEPROM 数据查询

当 EEPROM 正在处理一个字节的编程操作时，读取此地址将返回 $FF。编程结束后，被编程的数据即可以正确读出。这一方法可用来判断何时可以写下一个字节。数据查询对数据 $FF 无效。但用户应该考虑到，全片擦除将所有的单元擦为 $FF，所以编程数据为 $FF 时可以跳过这个操作。不过这不适用于全片擦除时 EEPROM 内容被保留的情况。用户若在此时编程 $FF，在进行下一字节编程之前至少等待 t_{WD_EEPROM} 的时间。 t_{WD_EEPROM} 的值见 Table 59。

Table 59. 写下一个 Flash 或 EEPROM 单元之前的最小等待时间

符号	最小等待时间
t_{WD_FLASH}	4.5 ms
t_{WD_EEPROM}	9.0 ms
t_{WD_ERASE}	9.0 ms
t_{WD_FUSE}	4.5 ms

Figure 68. 串行编程波形图

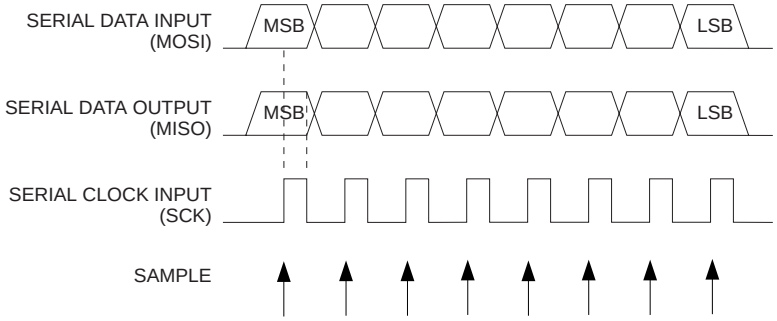


Table 60. 串行编程指令集

指令	指令格式				操作
	字节 1	字节 2	字节 3	字节 4	
编程使能	1010 1100	0101 0011	xxxx xxxx	xxxx xxxx	$\overline{\text{RESET}}$ 拉低后使能串行编程
全片擦除	1010 1100	100x xxxx	xxxx xxxx	xxxx xxxx	擦除 EEPROM 及 Flash
读程序存储器	0010 **H**000	xxxx xx**aa**	**bbbb bbbb**	**oooo oooo**	从字地址为 **a**:**b** 的程序存储器读取 **H**(高或低字节) 数据的 **o**
加载程序存储器页	0100 **H**000	xxxx xxxx	xxxx **bbbb**	**iiii iiii**	向字地址为 **b** 的程序存储页 **H**(高或低字节) 写入数据 **i**。应先写低字节再写高字节
写程序存储器页	0100 1100	xxxx xx**aa**	**bbbb** xxxx	xxxx xxxx	在地址 **a**:**b** 加载程序存储页
读 EEPROM 存储器	1010 0000	xxxx xxxx	x**bbb bbbb**	**oooo oooo**	从 EEPROM 的地址 **b** 处读出数据 **o**
写 EEPROM 存储器	1100 0000	xxxx xxxx	x**bbb bbbb**	**iiii iiii**	向 EEPROM 地址 **b** 处中写入数据 **i**
读锁定位	0101 1000	0000 0000	xxxx xxxx	xxxx **xxoo**	读锁定位。“0” 为已编程，“1” 为未编程。细节见 P99Table 47。
写锁定位	1010 1100	111x xxxx	xxxx xxxx	1111 11**ii**	写锁定位。写 “0” 表示编程锁定位。细节见 P99Table 47。
读标识字节	0011 0000	xxxx xxxx	xxxx xx**bb**	**oooo oooo**	从地址 **b** 读取标识字节 **o**
写熔丝位	1010 1100	1010 0000	xxxx xxxx	**iiii iiii**	“0” 表示已编程，“1” 表示未编程。见 P100Table 50。
写高熔丝位	1010 1100	1010 1000	xxxx xxxx	xxx**i iiii**	“0” 表示已编程，“1” 表示未编程。见 P100Table 49。
读熔丝位	0101 0000	0000 0000	xxxx xxxx	**oooo oooo**	读熔丝位。“0” 表示已编程，“1” 表示未编程。细节见 P100Table 50。
读高熔丝位	0101 1000	0000 1000	xxxx xxxx	xxx**o oooo**	读熔丝高位。“0” 表示已编程，“1” 表示未编程。细节见 P100Table 49。
读校准字节	0011 1000	xxxx xxxx	0000 00**bb**	**oooo oooo**	读校准字节 **o**.

Note: **a** = 地址高位，**b** = 地址低位，**H** = 0 - 低字节，1 - 高字节 ，**o** = 数据输出，**i** = 数据输入，x = 无用途

串行编程特性参数

Figure 69. 串行编程时序

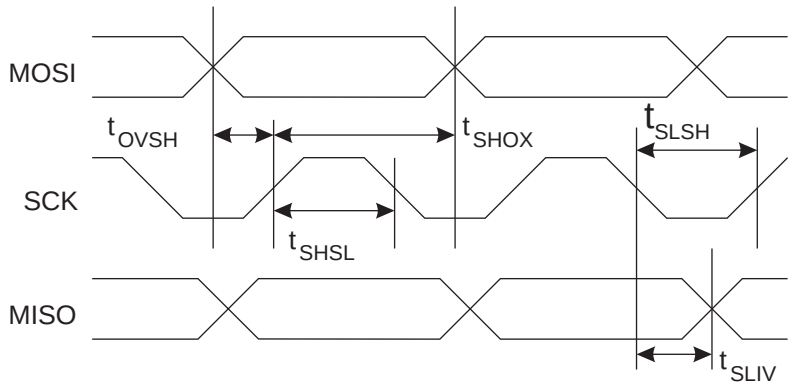


Table 61. 串行编程特性，T_A = -40°C 到 85°C， V_{CC} = 2.7V - 5.5V (除非另有说明)(1)

符号	参数	最小值	典型值	最大值	单位
$1/t_{CLCL}$	振荡器频率 (V_{CC} = 2.7 - 5.5 V)	0		8	MHz
t_{CLCL}	振荡器周期 (V_{CC} = 2.7 - 5.5 V)	125			ns
$1/t_{CLCL}$	振荡器频率 (V_{CC} = 4.5 - 5.5 V)	0		16	MHz
t_{CLCL}	振荡器周期 (V_{CC} = 4.5 - 5.5 V)	62.5			ns
t_{SHSL}	SCK 脉宽高	2 t_{CLCL} (1)			ns
t_{SLSH}	SCK 脉宽低	2 t_{CLCL} (1)			ns
t_{OVSH}	当 SCK 为高 MOSI 建立	t_{CLCL}			ns
t_{SHOX}	SCK 为高后 MOSI 保持	2 t_{CLCL}			ns
t_{SLIV}	MISO 有效 SCK 为低		20		ns

Note: 1. f_{ck} < 12 MHz 为 2 t_{CLCL}，f_{ck} >= 12 MHz 为 3 t_{CLCL}

电气特性

绝对极限值 *

工作温度	-55°C 到 +125°C
存储温度	-65°C 到 +150°C
各个引脚对地的电压，除了 $\overline{\text{RESET}}$	-1.0V 到 V_{CC} + 0.5V
$\overline{\text{RESET}}$ 引脚对地的电压	-0.5V 到 +13.0V
最大工作电压	6.0V
每个 I/O 引脚上的直流电流	40.0 mA
V_{CC} 与 GND 引脚上的直流电流	200.0 mA

*NOTICE: 如果强制芯片在超出“ 绝对极限值” 表中所列的条件之下工作可能造成器件的永久损坏。这仅是工作应力的极限。并不表示器件可以工作于表中所列条件之下，或是那些超越工作范围明确规定的其他条件之下。长时间工作于绝对极限值可能会影响器件的寿命。

直流特性 T_A = -40°C - 85°C， V_{CC} = 2.7V-5.5V (除非另外说明)

符号	参数	条件	最小值	典型值	最大值	单位
V_{IL}	输入低电压	除 XTAL1 引脚	-0.5		$0.2V_{CC}$	V
V_{IL1}	输入低电压	XTAL1 引脚，选择外部时钟	-0.5		$0.1V_{CC}$	V
V_{IH}	输入高电压	除了 XTAL1 及 RESET 引脚	$0.6V_{CC}$(3)		V_{CC} +0.5	V
V_{IH1}	输入高电压	XTAL1 引脚，选择外部时钟	$0.8V_{CC}$(3)		V_{CC} +0.5	V
V_{IH2}	输入高电压	$\overline{\text{RESET}}$ 引脚	$0.9V_{CC}$(3)		V_{CC} +0.5	V
V_{OL}	输出低电压 (4) (端口 A、 B)	I_{OL} = 20 mA，V_{CC} = 5V I_{OL} = 10 mA，V_{CC} = 3V			0.7 0.5	V V
V_{OH}	输出高电压 (5) (端口 A、 B)	I_{OH} = -20 mA，V_{CC} = 5V I_{OH} = -10 mA， V_{CC} = 3V	4.2 2.3			V V
I_{IL}	输入漏电流， I/O 引脚	Vcc = 5.5V，引脚为低 (绝对值)			1	µA
I_{IH}	输入漏电流， I/O 引脚	Vcc = 5.5V，引脚为低 (绝对值)			1	µA
R_{RST}	Reset 引脚上拉电阻		20		100	kΩ
R_{pu}	I/O 引脚上拉电阻		20		100	kΩ

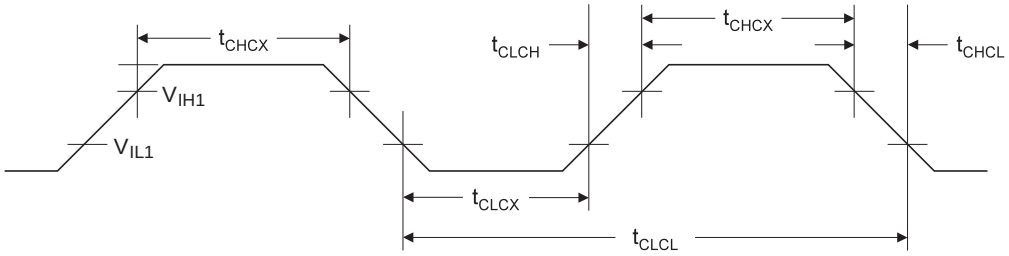
直流特性 T_A = -40°C - 85°C， V_{CC} = 2.7V-5.5V (除非另外说明) (Continued)

符号	参数	条件	最小值	典型值	最大值	单位
I_{CC}	电源电流	工作于 1 MHz,V_{CC} = 3V (ATtiny26L)		0.70(1)		mA
		工作于 4 MHz, V_{CC} = 3V (ATtiny26L)		2.5(1)	6	mA
		工作于 8 MHz, V_{CC} = 5V (ATtiny26)		8(1)	15	mA
		空闲 1 MHz, V_{CC} = 3V (ATtiny26L)		0.18(1)		mA
		空闲 4 MHz, V_{CC} = 3V (ATtiny26L)		0.75(1)	2	mA
		空闲 8 MHz, V_{CC} = 5V (ATtiny26)		3.5(1)	7	mA
	掉电模式 (6)	WDT 使能，V_{CC} = 3V		7.5(1)	15	µA
		WDT 禁用，V_{CC} = 3V		0.3(1)	3	µA
V_{ACIO}	模拟比较器输入偏置电压	V_{CC} = 5V V_{in} = V_{CC}/2		<10	40	mV
I_{ACLK}	模拟比较器输入漏电流	V_{CC} = 5V V_{in} = V_{CC}/2	-50		50	nA
t_{ACID}	模拟比较器传输延时	V_{CC} = 2.7V V_{CC} = 4.0V		750 500		ns

Notes:
1. 典型值为在 25°C 时的值。
2. " 最大值 " 表示保证引脚读取数值为低时的最高值
3. " 最小值 " 表示保证引脚读取数值为高时的最低值
4. 虽然在稳定状态条件(非瞬态)下每个I/O端口都可以吸收比测试条件下更多的电流(20 mA ,V_{CC} = 5V；10 mA，V_{CC} = 3V) ，但是仍需要遵循以下要求：
   1] 所有端口的 IOL 总和不能超过 400 mA。
   2] 端口 A0 - A7 的 IOL 总和不能超过 300 mA。
   3] 端口 B0 - B7 的 IOL 总和不能超过 300 mA。
   如果 IOL 超过了测试条件， VOL 可能超过相关指标。不保证引脚可以吸收比列于此处的测试条件更大的电流。
5. 虽然在稳定状态条件 (非瞬态) 下每个 I/O 端口都可以输出比测试条件下更多的电流 (20 mA ,V_{CC} = 5V ； 10 mA，V_{CC} = 3V) ，但是需要遵循以下要求：
   1] 所有端口的 IOH 总和不能超过 400 mA。
   2] 端口 A0 - A7 的 IOH 总和不能超过 300 mA。
   3] 端口 B0 - B7 的 IOH 总和不能超过 300 mA。
   如果 IOH 超过了测试条件， VOH 可能超过相关指标。不保证引脚可以输出比列于此处的测试条件更大的电流。
6. 掉电状态时最小 V_{CC} 为 2.5V。

外部时钟驱动波形

Figure 70. 外部时钟驱动波形



外部时钟驱动

Table 62. 外部时钟驱动

符号	参数	V_{CC} = 2.7 - 5.5V		V_{CC} = 4.5 - 5.5V		单位
		最小值	最大值	最小值	最大值	
$1/t_{CLCL}$	振荡器频率	0	8	0	16	MHz
t_{CLCL}	时钟周期	125		62.5		ns
t_{CHCX}	高电平时间	50		25		ns
t_{CLCX}	低电平时间	50		25		ns
t_{CLCH}	上升时间		1.6		0.5	µs
t_{CHCL}	下降时间		1.6		0.5	µs
Δt_{CLCL}	时钟周期的变化		2		2	

ADC 特性参数

Table 63. 单端通道下 ADC 特性参数， T_A = -40°C 到 85°C

符号	参数	条件	最小值	典型值	最大值	单位
	分辨率	单端转换		10		Bits
	绝对精度 (包括 INL、 DNL、量化误差、增益及偏置误差)	单端转换 V_{REF} = 4V，V_{CC} = 4V ADC 时钟 = 200 kHz		1		LSB
		单端转换 V_{REF} = 4V，V_{CC} = 4V ADC 时钟 = 1 MHz		2		LSB
		单端转换 V_{REF} = 4V，V_{CC} = 4V ADC 时钟 = 200 kHz 噪声抑制模式		1		LSB
		单端转换 V_{REF} = 4V，V_{CC} = 4V ADC 时钟 = 1 MHz 噪声抑制模式		2		LSB
	整体非线性 (INL)	单端转换 V_{REF} = 4V，V_{CC} = 4V ADC 时钟 = 200 kHz		0.5		LSB
	差分非线性 (DNL)	单端转换 V_{REF} = 4V，V_{CC} = 4V ADC 时钟 = 200 kHz		0.5		LSB
	增益误差	单端转换 V_{REF} = 4V，V_{CC} = 4V ADC 时钟 = 200 kHz		0.75		LSB
	偏移误差	单端转换 V_{REF} = 4V，V_{CC} = 4V ADC 时钟 = 200 kHz		0.5		LSB
	时钟频率		50		1000	kHz
	转换时间		13		260	µs
AVCC	模拟电压		V_{CC} - 0.3(1)		V_{CC} + 0.3(2)	V
V_{REF}	参考电压		2.0		AVCC	V
V_{IN}	输入电压		GND		V_{REF}	V
	ADC 转换输出		0		1023	LSB
	输入带宽			38.5		kHz
V_{INT}	片内参考电压		2.3	2.56	2.7	V
R_{REF}	参考电压输入阻抗			32		kΩ
R_{AIN}	模拟输入阻抗			100		MΩ

Note: 1. AVCC 最小值为 2.7V
2. AVCC 最大值为 5.5V

Table 64. 差分通道下 ADC 特性参数， T_A = -40°C 到 85°C

符号	参数	条件	最小值	典型值	最大值	单位
	分辨率	Gain = 1x			10	Bits
		Gain = 20x			10	Bits
	绝对精度	Gain = 1x V_{REF} = 4V， V_{CC} = 5V ADC 时钟 = 50 - 200 kHz		24		LSB
		Gain = 20x V_{REF} = 4V， V_{CC} = 5V ADC 时钟 = 50 - 200 kHz		27		LSB
	整体非线性 (INL) (偏移标定与增益误差后的精度)	Gain = 1x V_{REF} = 4V， V_{CC} = 5V ADC 时钟 = 50 - 200 kHz		1.5		LSB
		Gain = 20x V_{REF} = 4V， V_{CC} = 5V ADC 时钟 = 50 - 200 kHz		2		LSB
	增益误差	Gain = 1x		2		%
		Gain = 20x		2.5		%
	偏移误差	Gain = 1x V_{REF} = 4V, V_{CC} = 5V ADC 时钟 = 50 - 200 kHz		4		LSB
		Gain = 20x V_{REF} = 4V, V_{CC} = 5V ADC 时钟 = 50 - 200 kHz		6		LSB
	时钟频率		50		200	kHz
	转换时间		26		65	µs
AVCC	模拟电压		V_{CC} - 0.3(1)		V_{CC} + 0.3(2)	V
V_{REF}	参考电压		2.0		AVCC - 0.5	V
V_{IN}	输入电压		GND		V_{CC}	V
V_{DIFF}	输入差分电压		0		V_{REF}/Gain	V
	ADC 转换输出		0		1023	LSB
	输入带宽			4		kHz
V_{INT}	片内参考电压		2.3	2.56	2.7	V
R_{REF}	参考电压输入阻抗			32		kΩ
R_{AIN}	模拟输入阻抗			100		MΩ

Notes: 1. AVCC 最小值为 2.7V。
2. AVCC 最大值为 5.5V。

ATtiny26 典型特征参数

下面的图表给出了器件的典型性能。在产生过程中并不测试这些数据。全部电流测量数据都是在所有的 I/O 引脚配置为输入且内部上拉电阻使能的条件下测得的。时钟源为外部正弦波发生器产生的满幅值正弦波。

掉电模式下的功耗与选择的时钟无关。

耗电流与多个因素有关：工作电压、工作频率、 I/O 的负载、 I/O 引脚开关速率、执行的代码及环境温度。最主要的因素是工作电压和工作频率。

容性负载 I/O 的输出电流可通过公式 C_L*V_{CC}*f 进行估算， C_L = 负载电容， V_{CC} = 工作电压， f = I/O 引脚平均开关频率。

器件的特性化是在比测试极限频率更高的频率进行的。但是不保证器件能够正常在高于订货信息表给出的工作频率。

看门狗使能的掉电模式和看门狗禁止的掉电模式之间的电流差值即为开关看门狗定时器所需的电流。

工作电流

Figure 71. 工作电流和工作频率 (0.1 - 1.0 MHz) 的关系

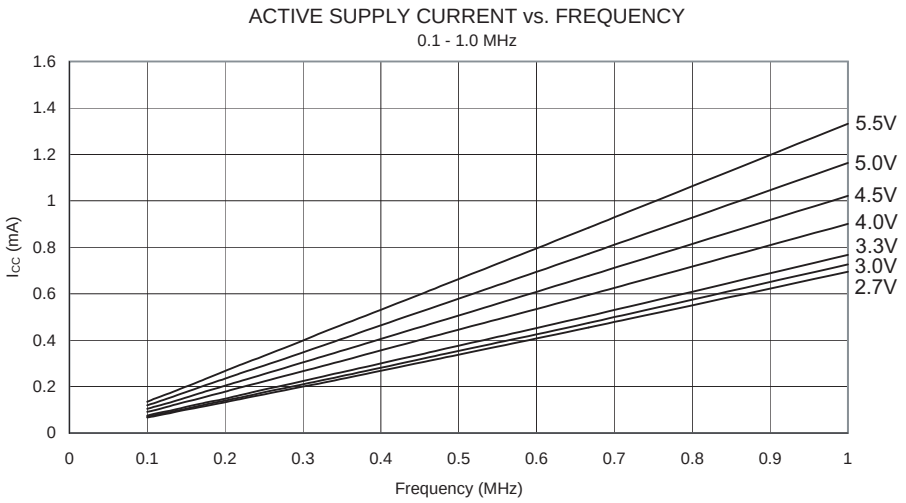


Figure 72. 工作电流和工作频率 (1 - 20 MHz) 的关系

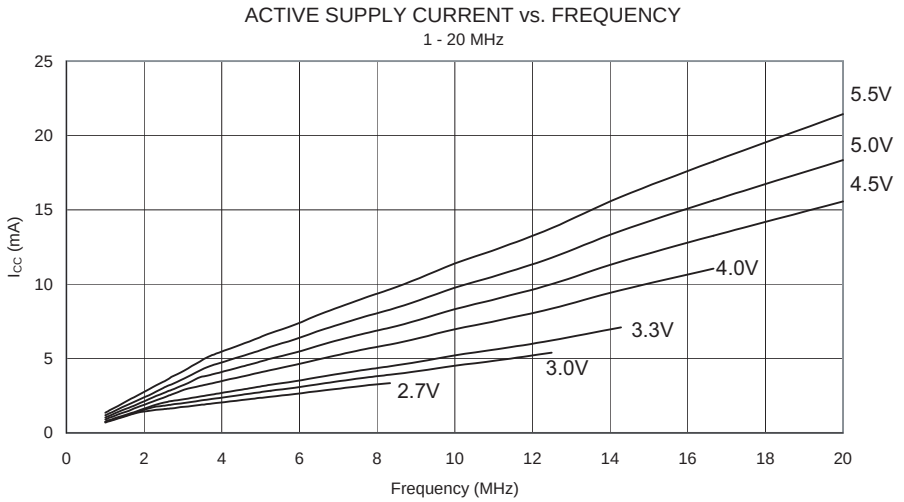


Figure 73. 工作电流和 V_{CC} 的关系 (内部 RC 振荡器，8 MHz)

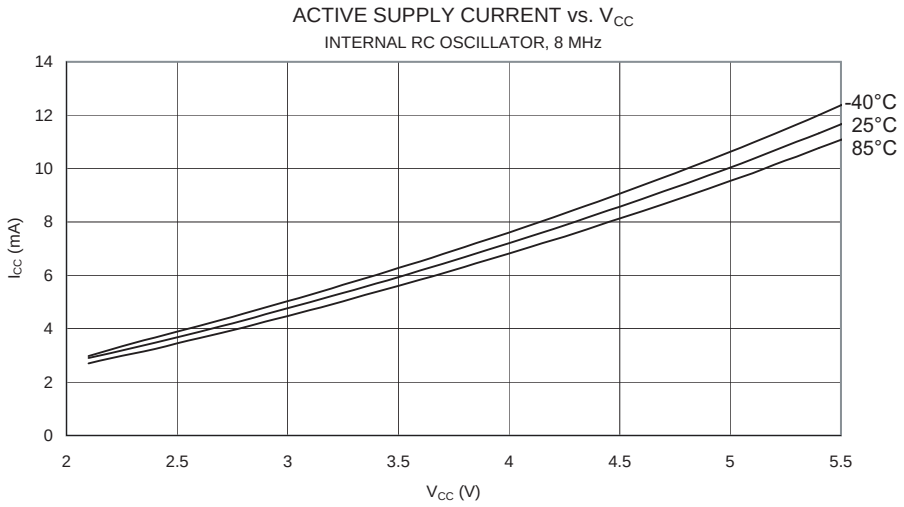


Figure 74. 工作电流和 V_{CC} 的关系 (内部 RC 振荡器， 4 MHz)

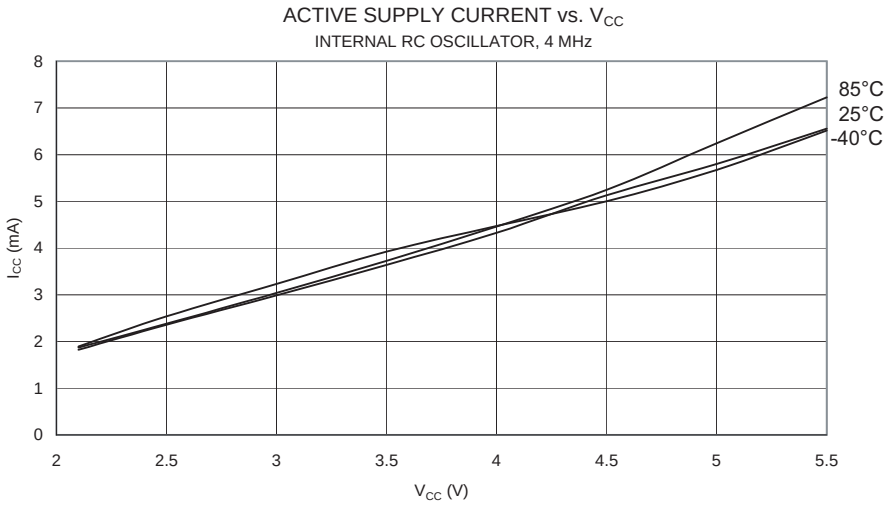


Figure 75. 工作电流和 V_{CC} 的关系 (内部 RC 振荡器， 2MHz)

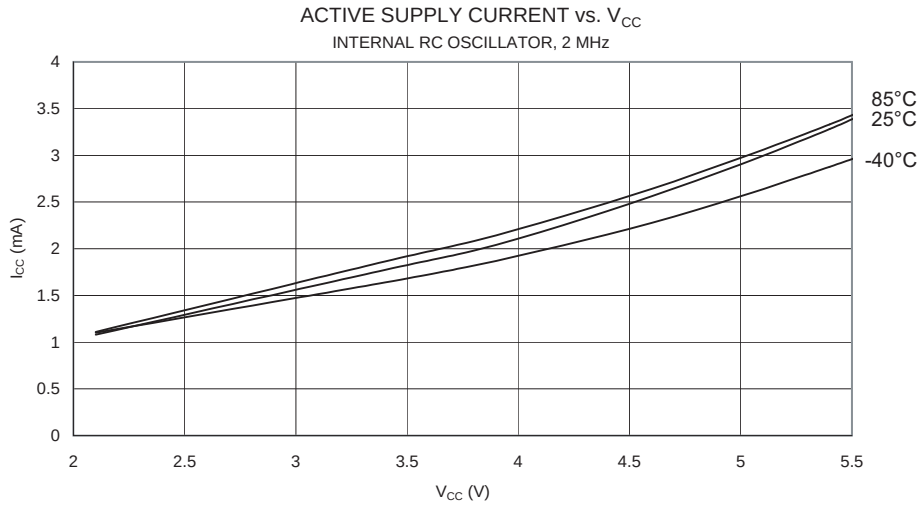


Figure 76. 工作电流和 V_{CC} 的关系 (内部 RC 振荡器， 1 MHz)

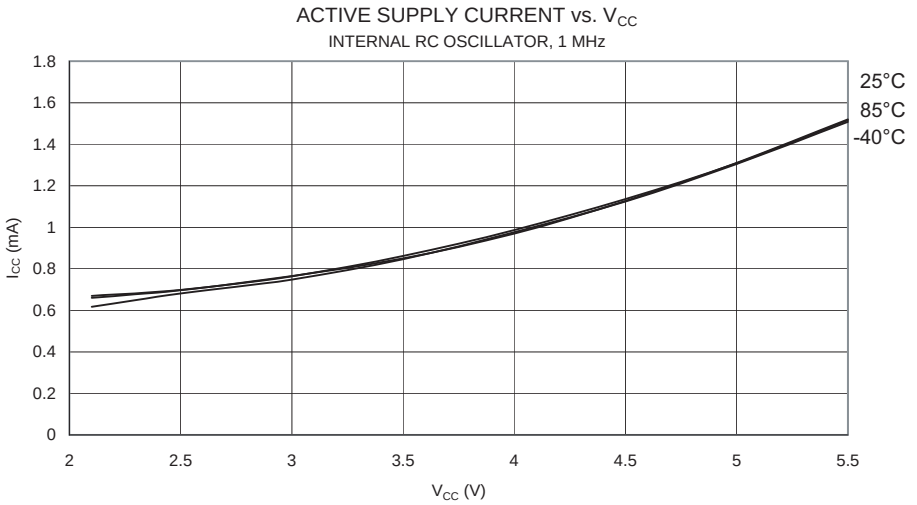


Figure 77. 工作电流和 V_{CC} 的关系 (PLL 振荡器)

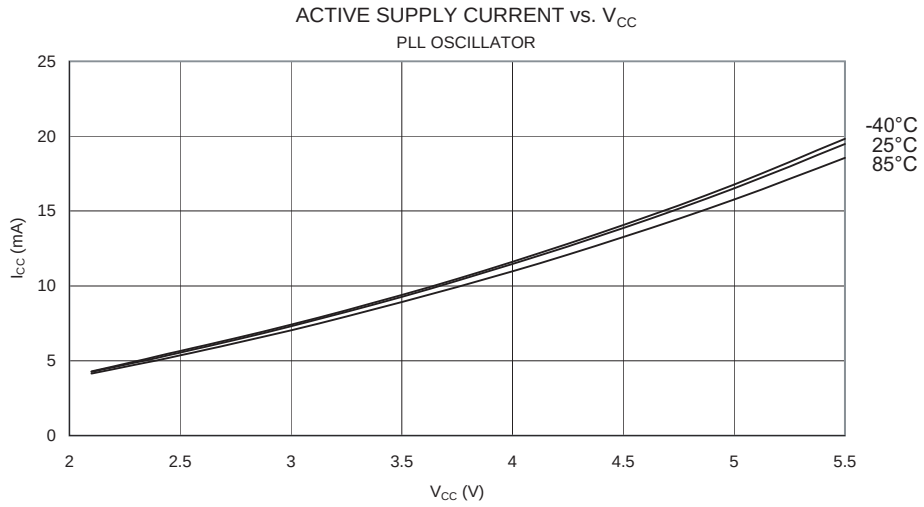
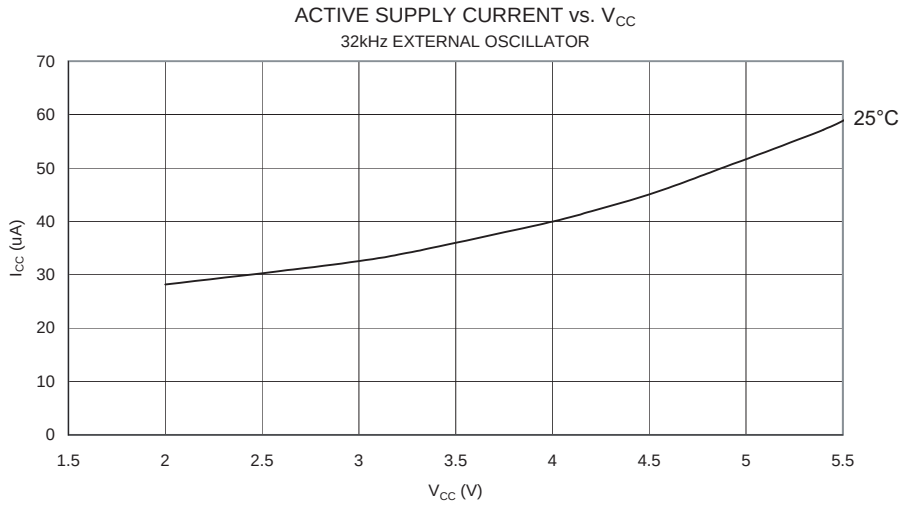


Figure 78. 工作电流和 V_{CC} 的关系 (32 kHz 外部振荡器)



空闲模式电流

Figure 79. 空闲模式电流和工作频率 (0.1 - 1.0 MHz) 的关系

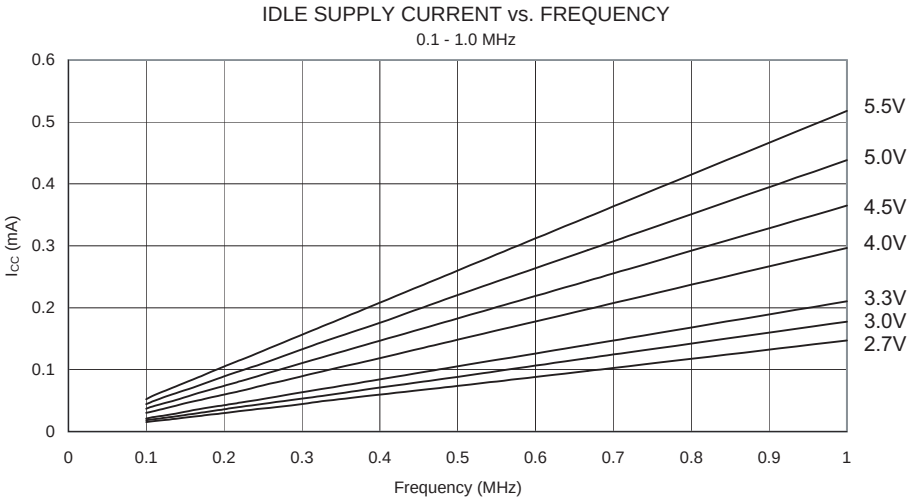


Figure 80. 空闲模式电流和工作频率 (1 - 20 MHz) 的关系

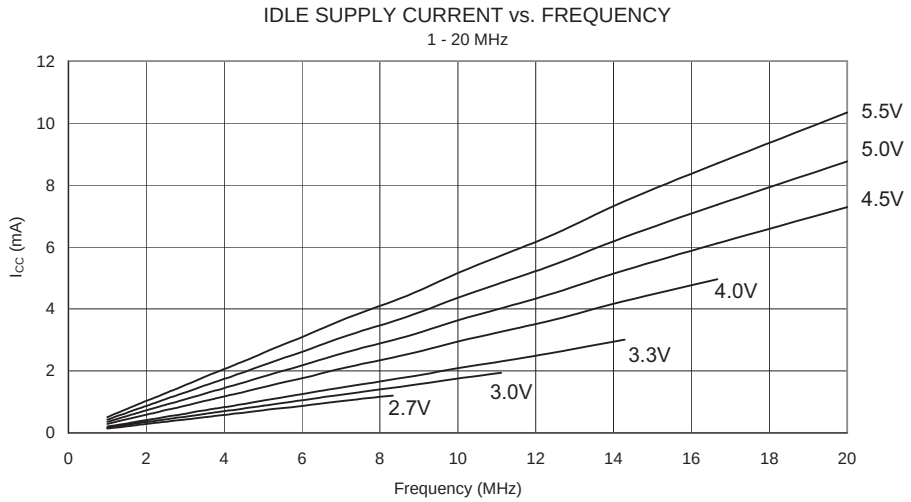


Figure 81. 空闲模式电流和 V_{CC} 的关系 (内部 RC 振荡器 ， 8 MHz)

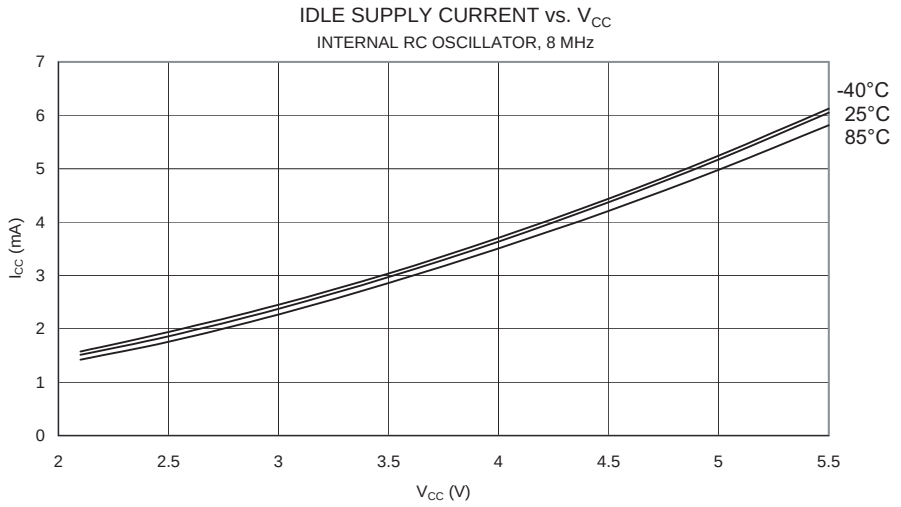


Figure 82. 空闲模式电流和 V_{CC} 的关系 (内部 RC 振荡器 ，4 MHz)

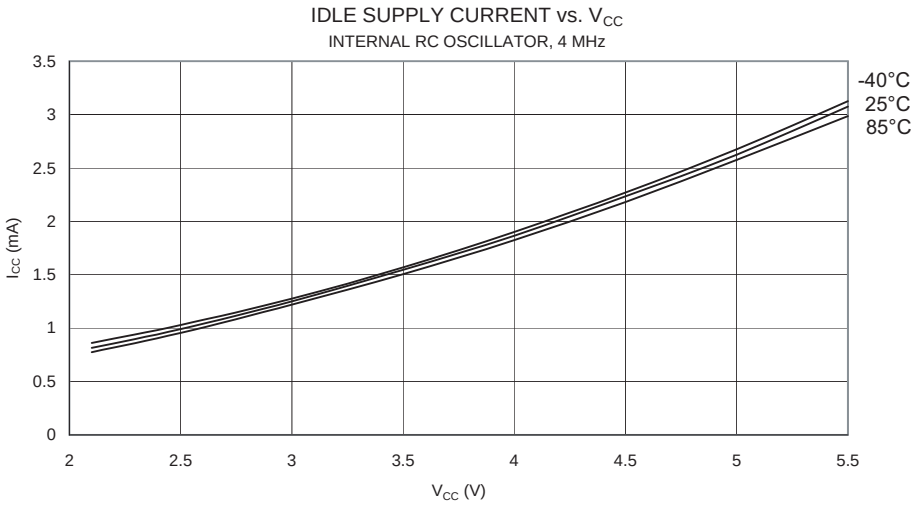


Figure 83. 空闲模式电流和 V_{CC} 的关系 (内部 RC 振荡器 ， 2 MHz)

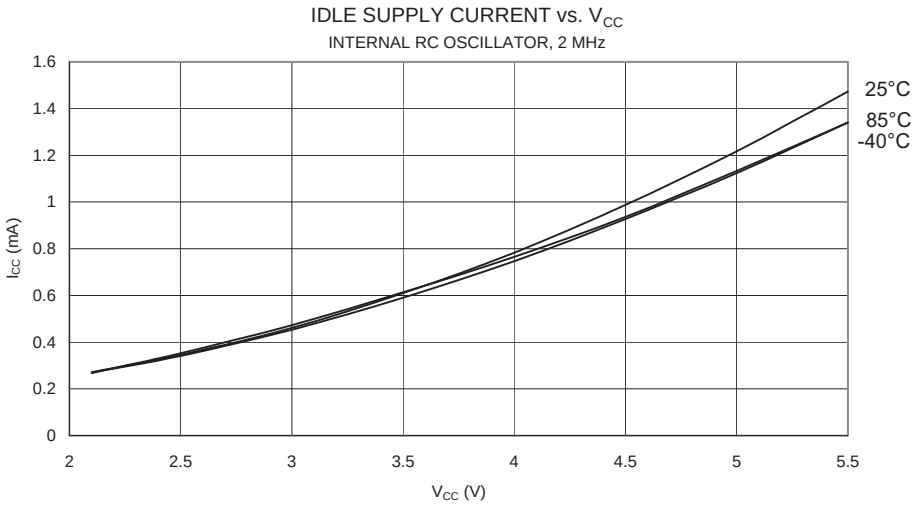


Figure 84. 空闲模式电流和 V_{CC} 的关系 (内部 RC 振荡器 ， 1 MHz)

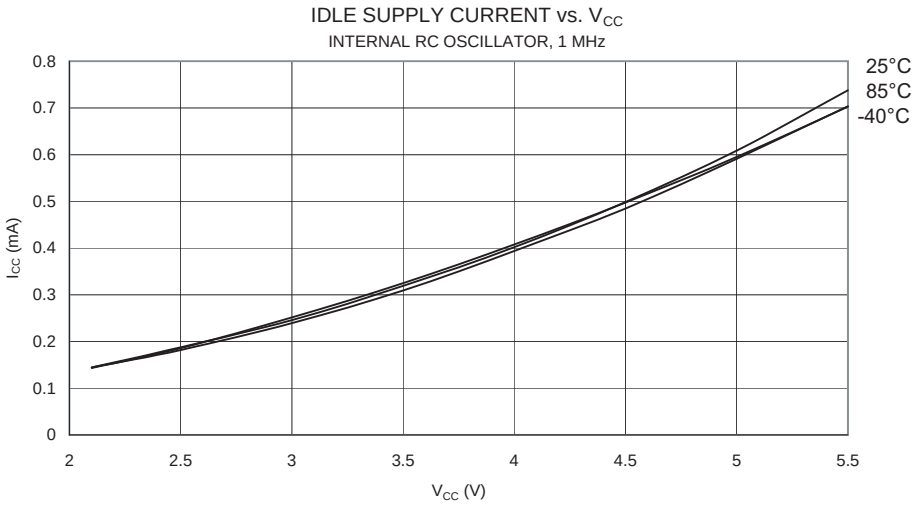


Figure 85. 空闲模式电流和 V_{CC} 的关系 (PLL 振荡器)

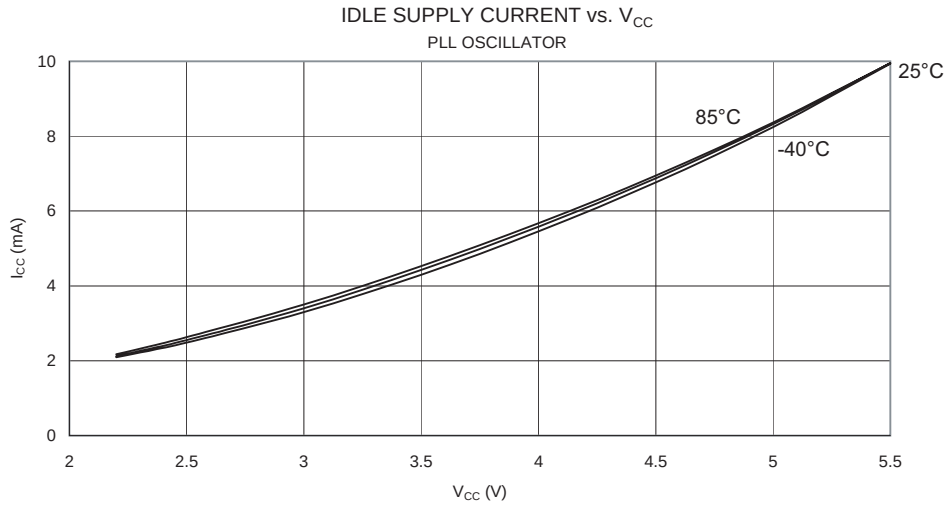
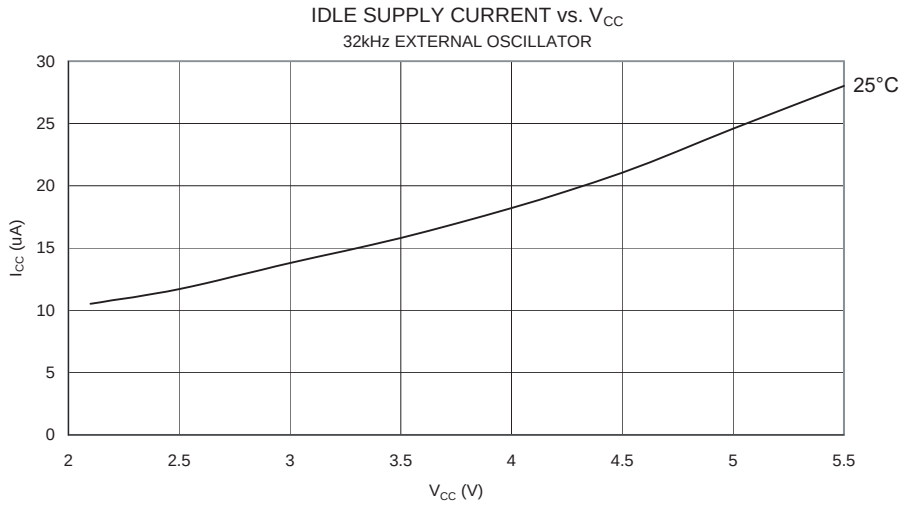


Figure 86. 空闲模式电流和 V_{CC} 的关系 (32 kHz 外部振荡器)



掉电模式电流

Figure 87. 掉电模式电流和 V_{CC} 的关系 (看门狗定时器禁用)

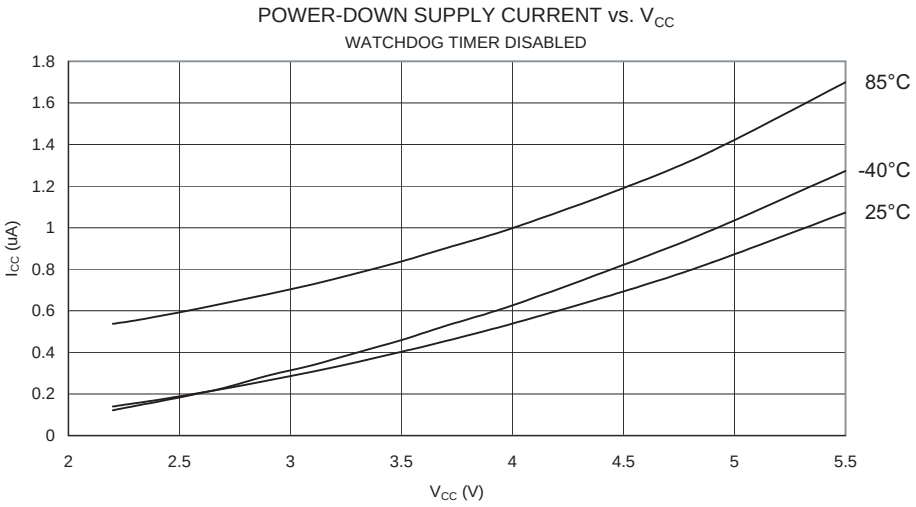
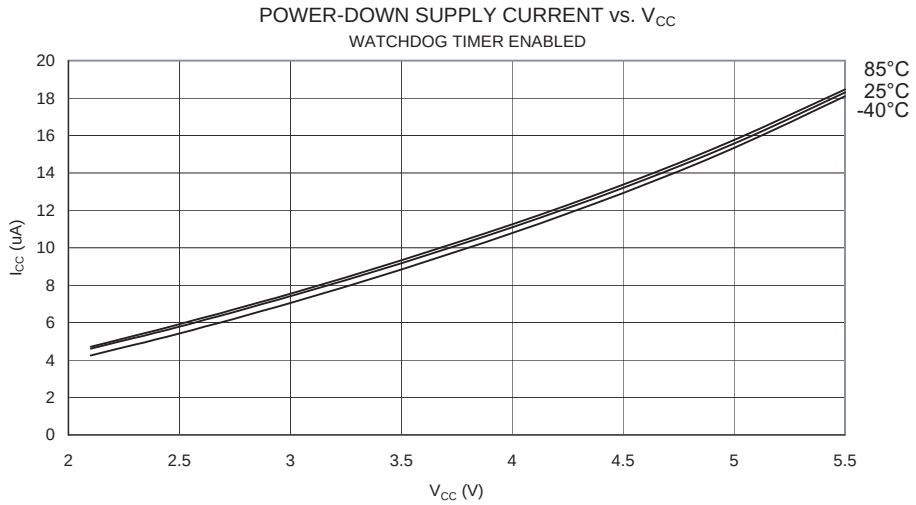


Figure 88. 掉电模式电流和 V_{CC} 的关系 (看门狗定时器使能)



Standby 模式电流

Figure 89. Standby 模式电流和 V_{CC} 的关系 (455 kHz 谐振器，看门狗定时器禁用)

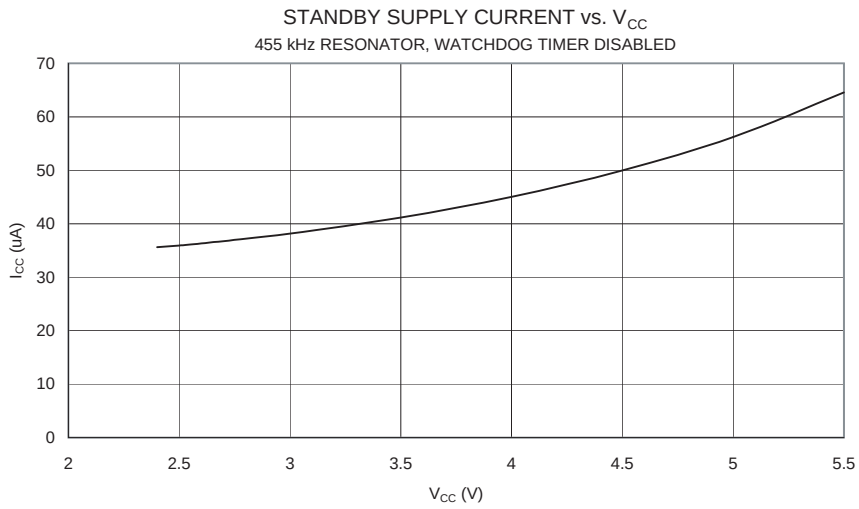


Figure 90. Standby 模式电流和 V_{CC} 的关系 (1 MHz 谐振器，看门狗定时器禁用)

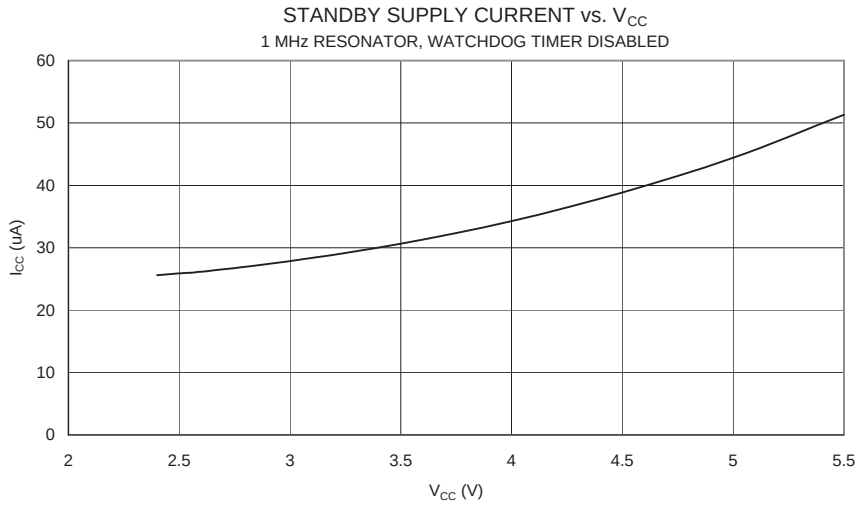


Figure 91. Standby 模式电流和 V_{CC} 的关系 (2 MHz 谐振器，看门狗定时器禁用)

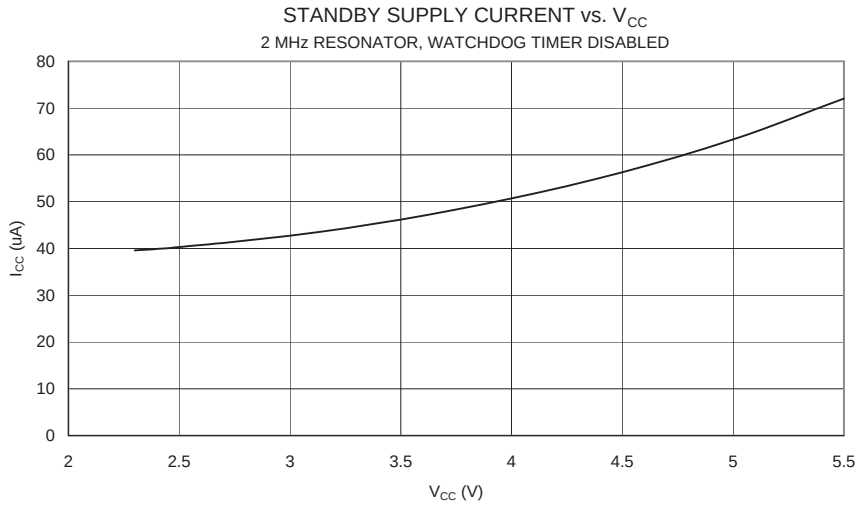


Figure 92. Standby 模式电流和 V_{CC} 的关系 (2 MHz XTAL，看门狗定时器禁用)

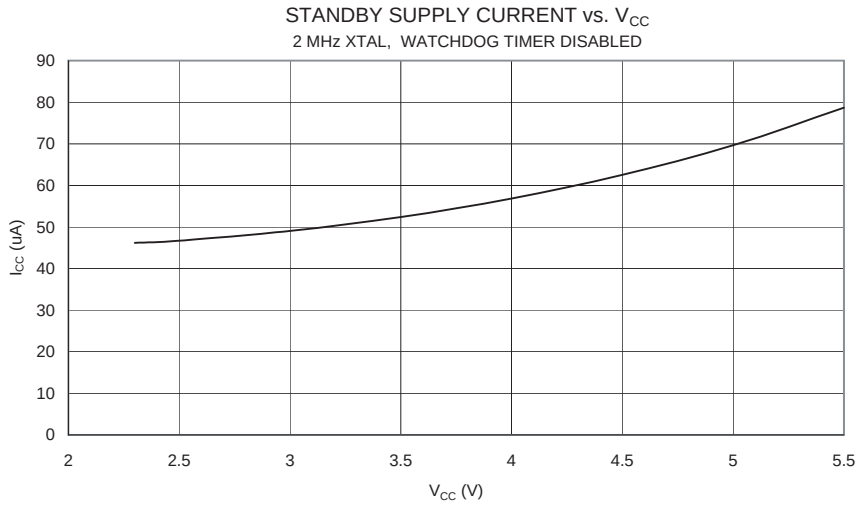


Figure 93. Standby 模式电流和 V_{CC} 的关系 (4 MHz 谐振器，看门狗定时器禁用)

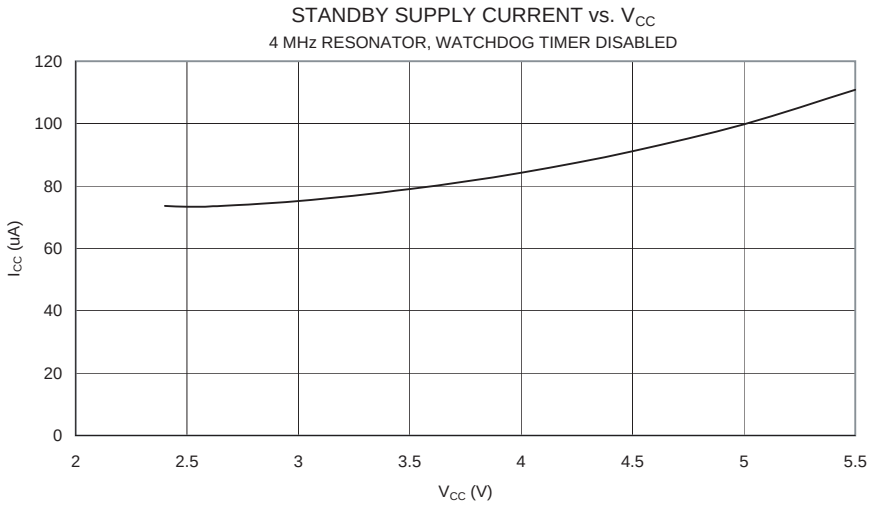


Figure 94. Standby 模式电流和 V_{CC} 的关系 (4 MHz XTAL，看门狗定时器禁用)

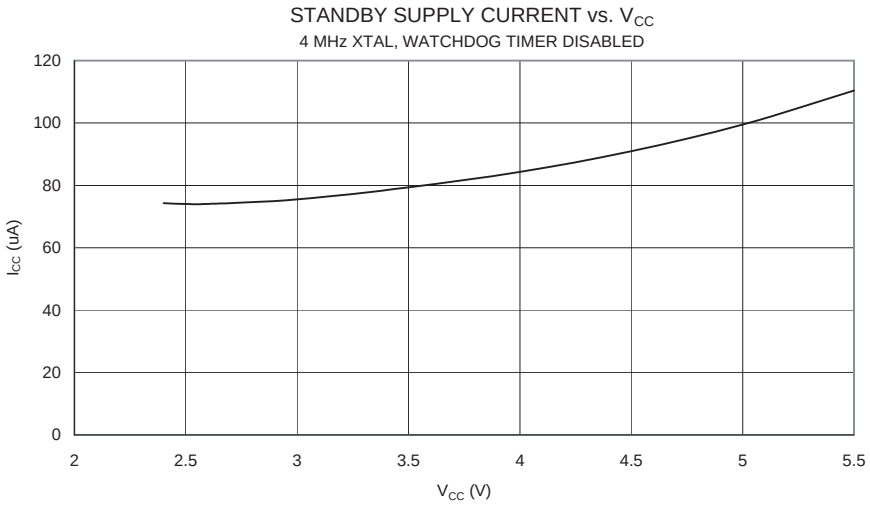


Figure 95. Standby 模式电流和 V_{CC} 的关系 (6 MHz 谐振器，看门狗定时器禁用)

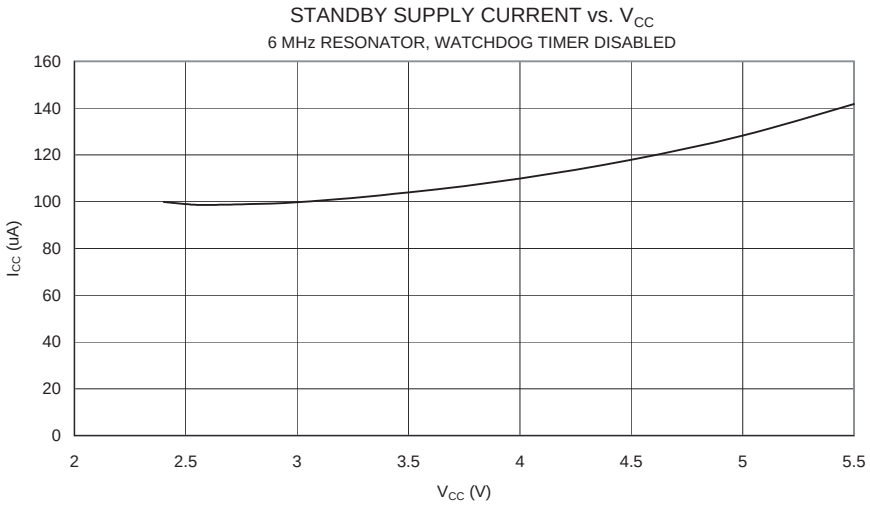
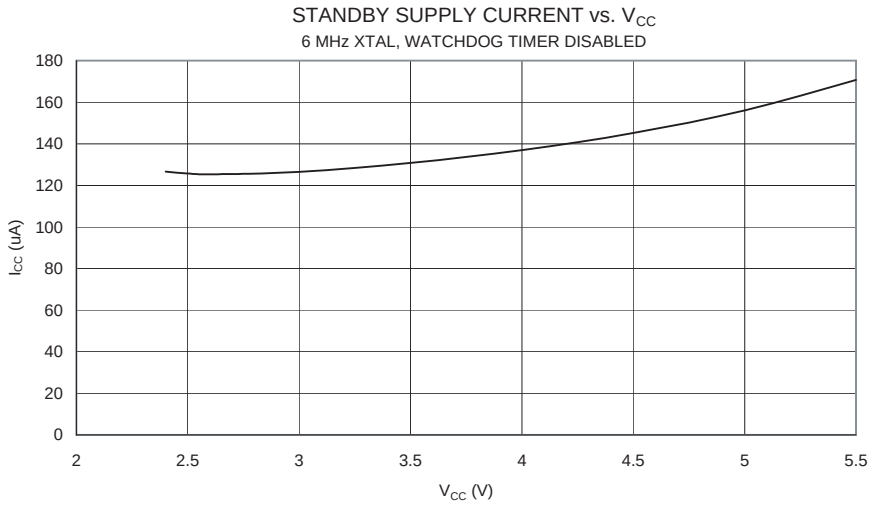


Figure 96. Standby 模式电流和 V_{CC} 的关系 (6 MHzXTAL，看门狗定时器禁用)



引脚上拉

Figure 97. I/O 引脚上拉电阻电流和输入电压的关系 (V_{CC} = 5V)

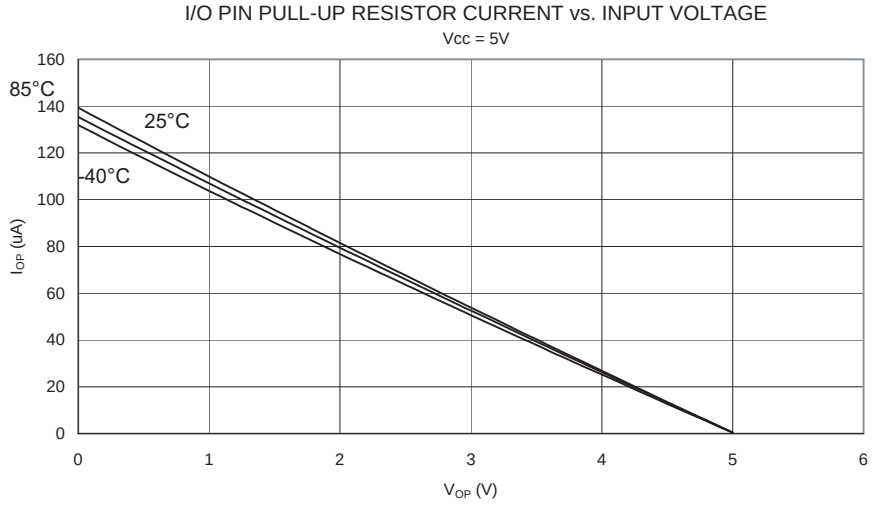


Figure 98. I/O 引脚上拉电阻电流和输入电压的关系 (V_{CC} = 2.7V)

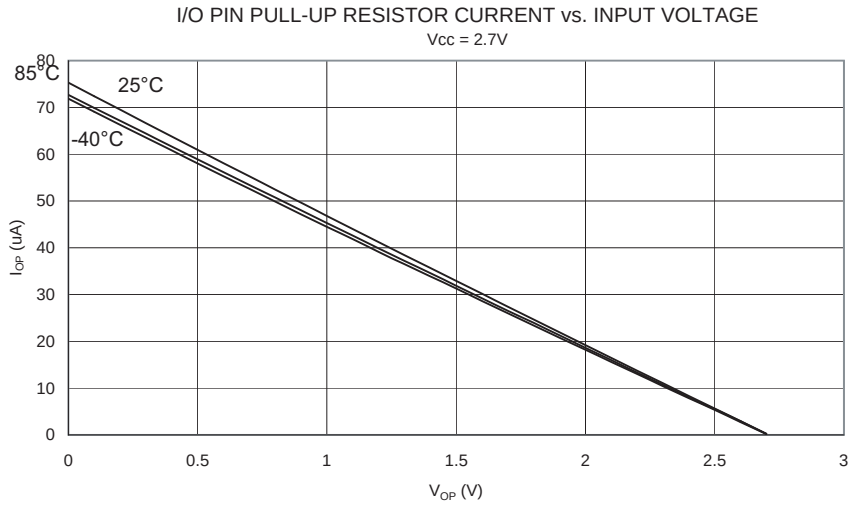


Figure 99. 复位 (Reset) 引脚上拉电阻电流和 Reset 引脚电压的关系 (V_{CC} = 5V)

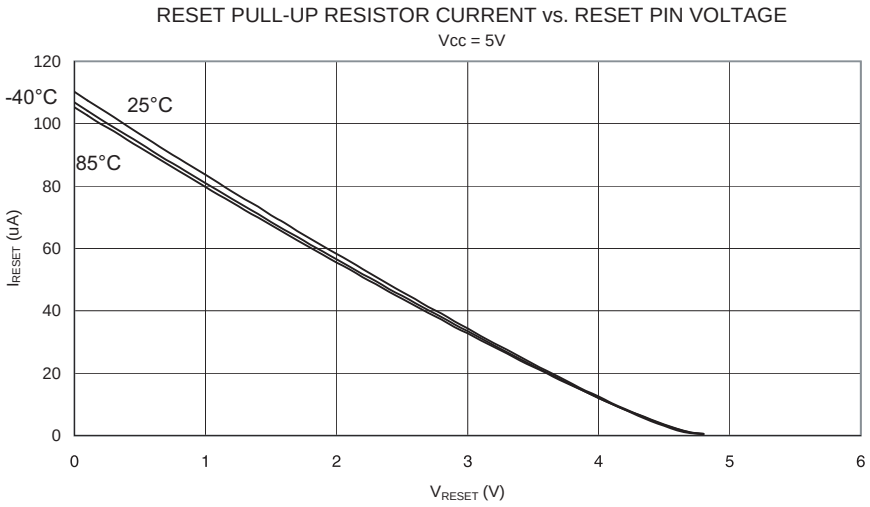
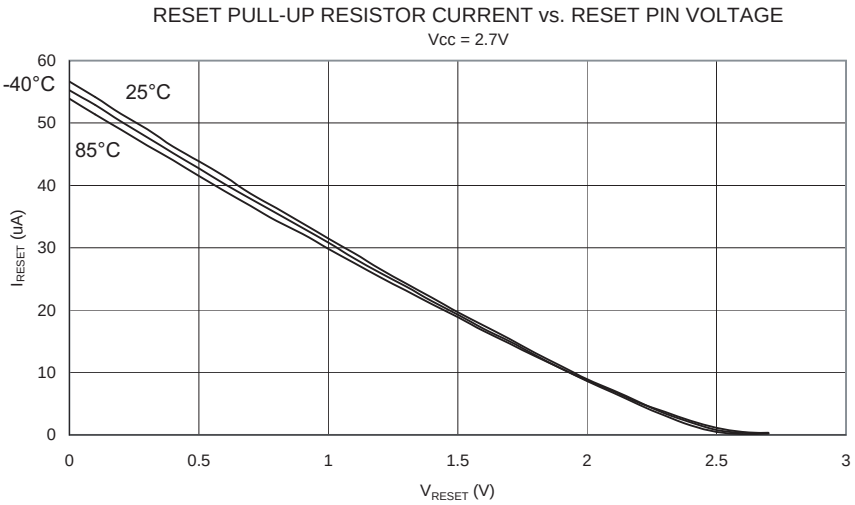


Figure 100. 复位 (Reset) 引脚上拉电阻电流和 Reset 引脚电压的关系 (V_{CC} = 2.7V)



驱动能力

Figure 101. I/O 引脚源电流和输出电压的关系 (V_{CC} = 5V)

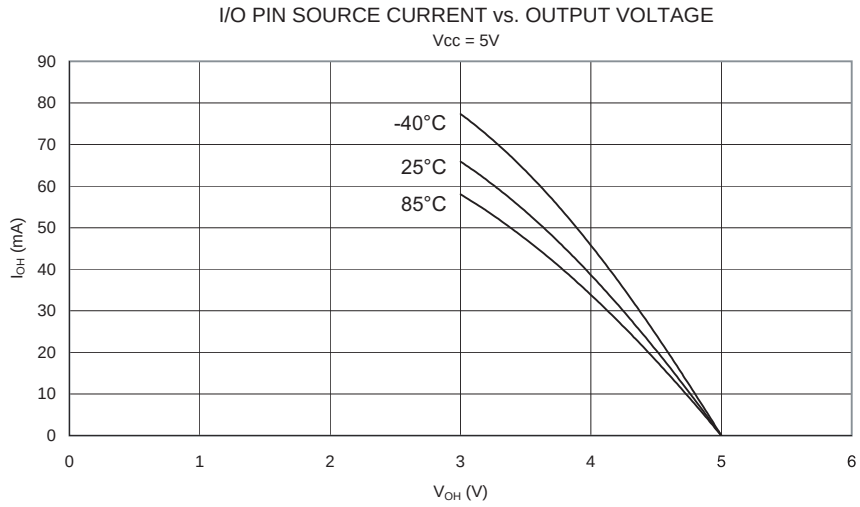


Figure 102. I/O 引脚源电流和输出电压的关系 (V_{CC} = 2.7V)

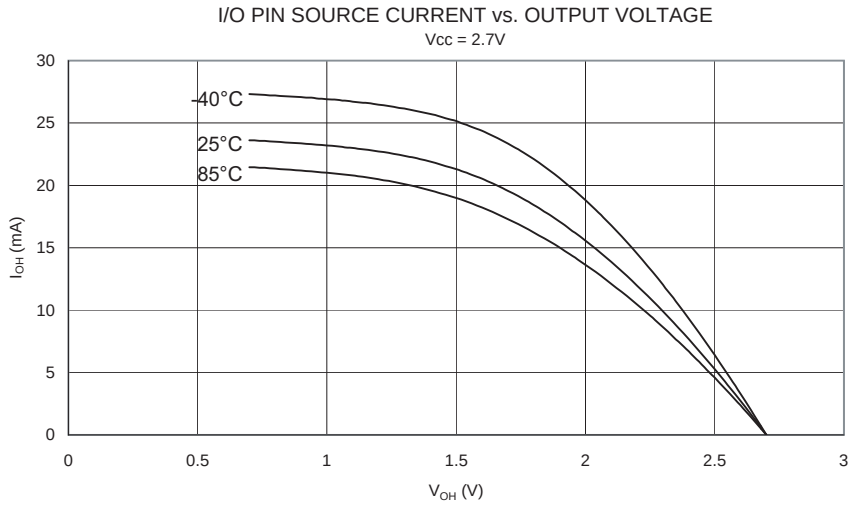


Figure 103. I/O 引脚吸收电流和输出电压的关系 (V_{CC} = 5V)

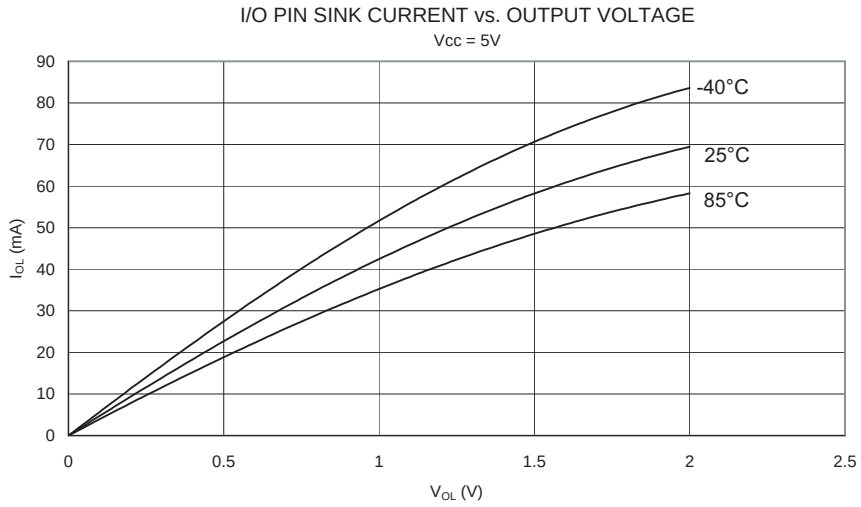


Figure 104. I/O 引脚吸收电流和输出电压的关系 (V_{CC} = 2.7V)

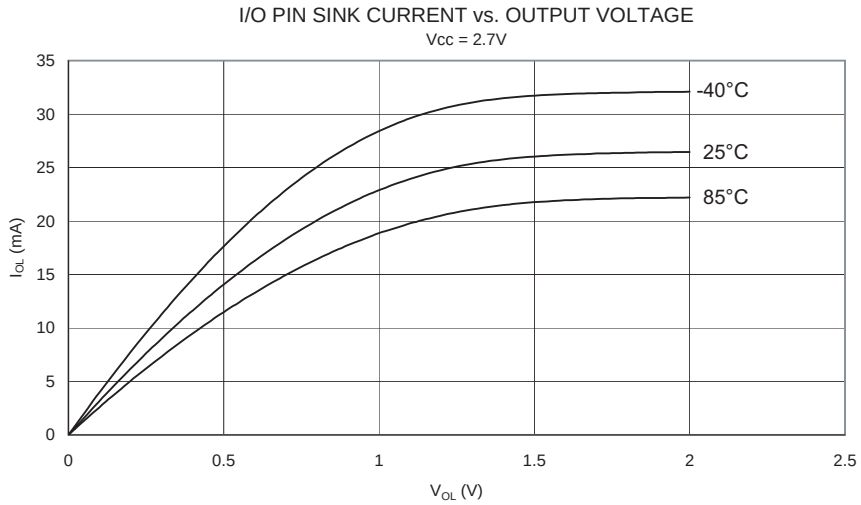


Figure 105. Reset 引脚作为 I/O – 源电流和输出电压的关系 (V_{CC} = 5V)

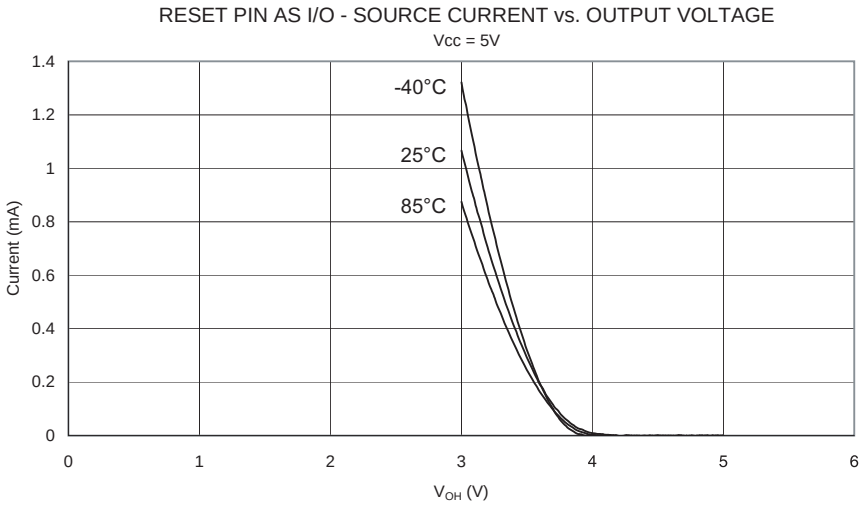


Figure 106. Reset 引脚作为 I/O – 源电流和输出电压的关系 (V_{CC} = 2.7V)

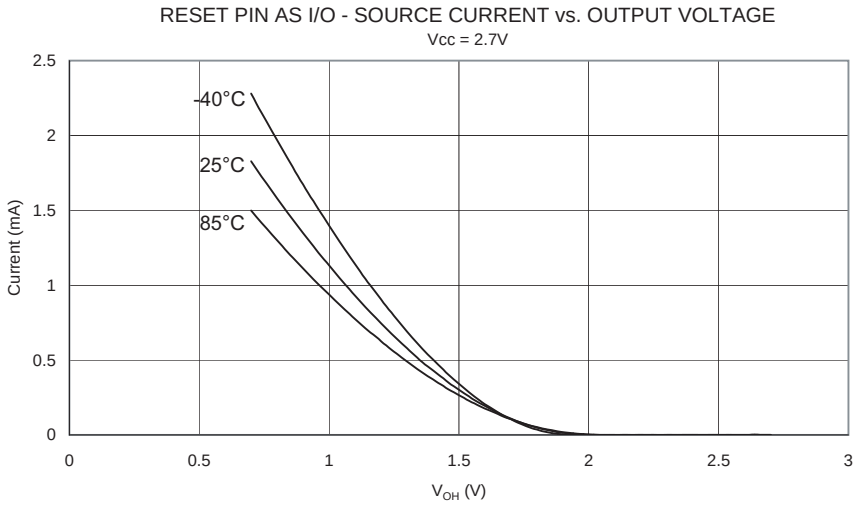


Figure 107. Reset 引脚作为 I/O – 吸收电流和输出电压的关系 (V_{CC} = 5V)

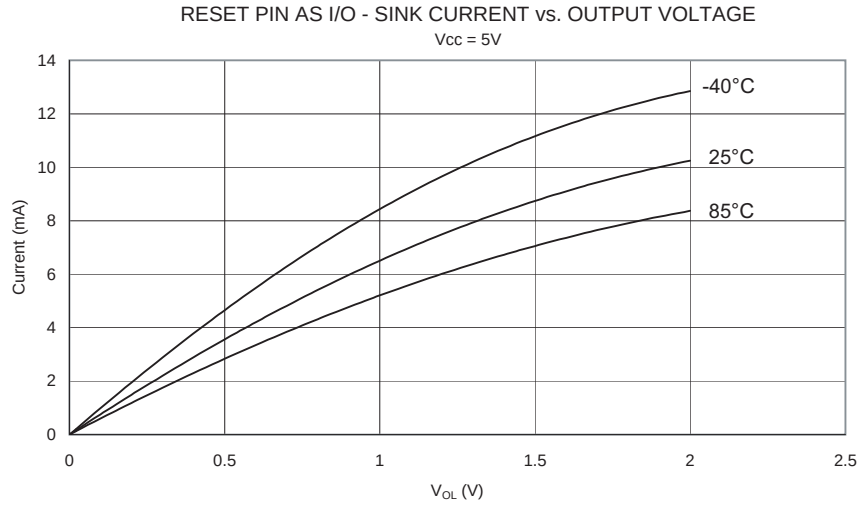
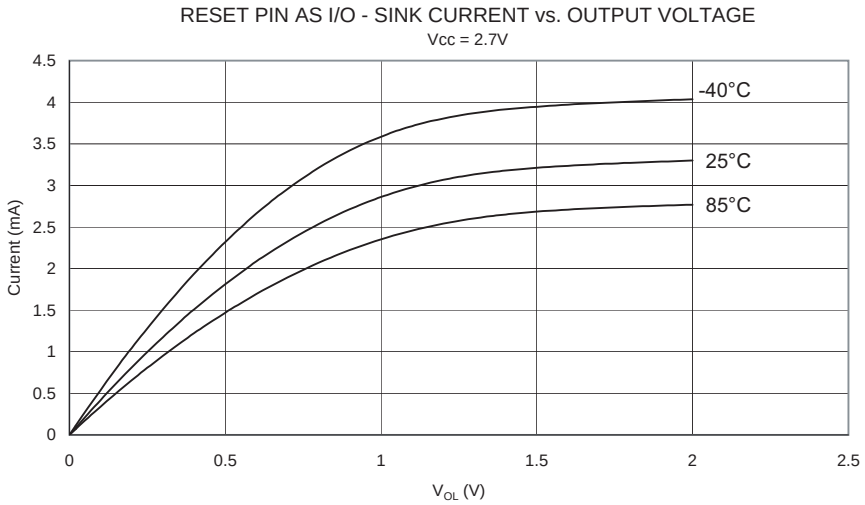


Figure 108. Reset 引脚作为 I/O – 吸收电流和输出电压的关系 (V_{CC} = 2.7V)



引脚门限及滞后

Figure 109. I/O 引脚输入门限电压和 V_{CC} 的关系 (V_{IH}, I/O 引脚读出值为 “1”)

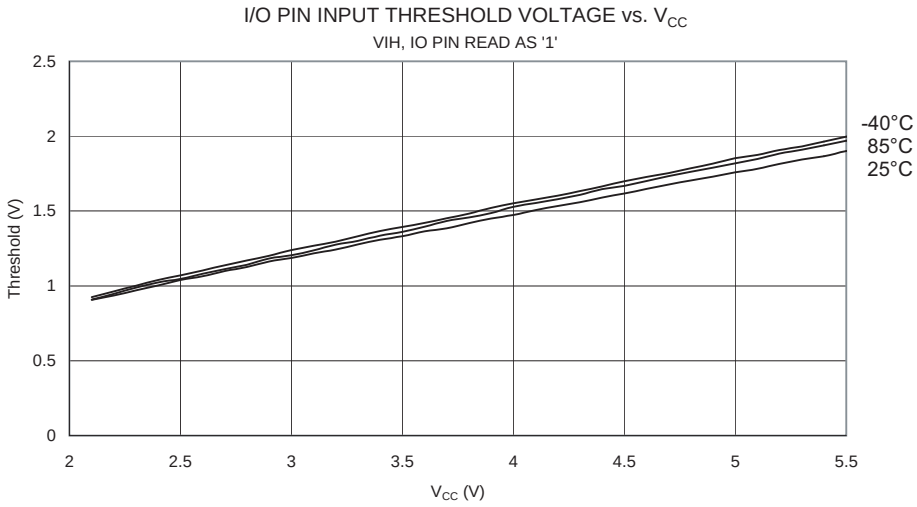


Figure 110. I/O 引脚输入门限电压和 V_{CC} 的关系 (V_{IL}, I/O 引脚读出值为 “0”)

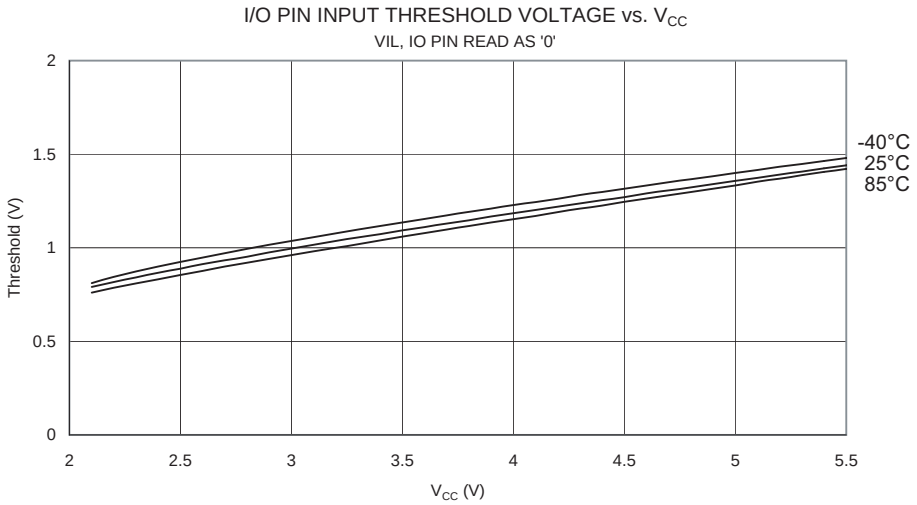


Figure 111. I/O 引脚输入迟滞和 V_{CC} 的关系

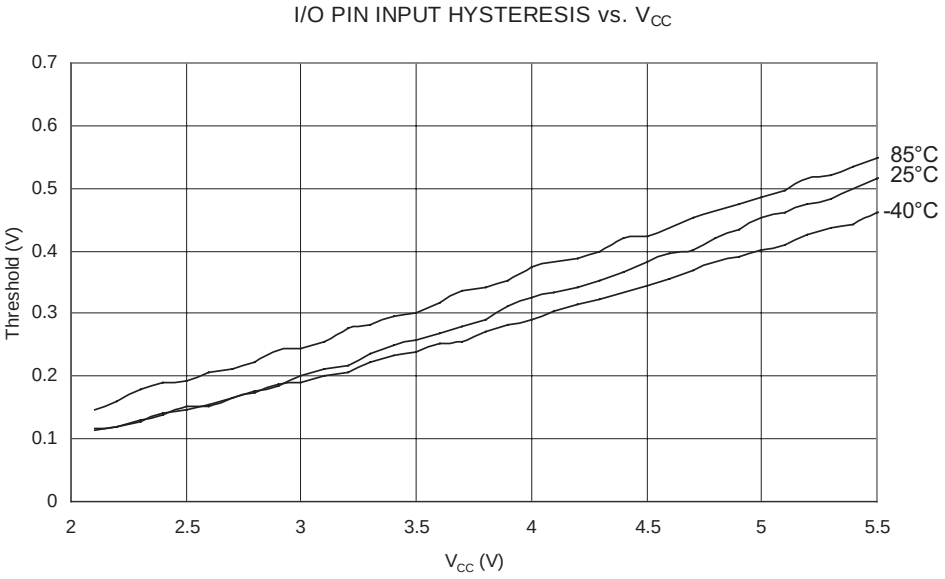


Figure 112. Reset 作为普通 I/O 输入门限电压和 V_{CC} 的关系 (V_{IH},Reset 引脚读出值为 “1”)

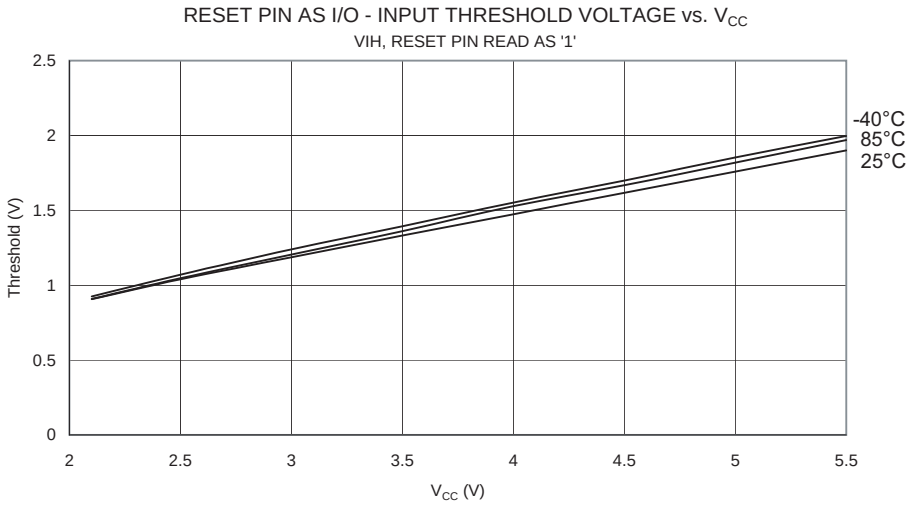


Figure 113. Reset 作为普通 I/O 输入门限电压和 V_{CC} 的关系 (V_{IL},Reset 引脚读出值为 “0”)

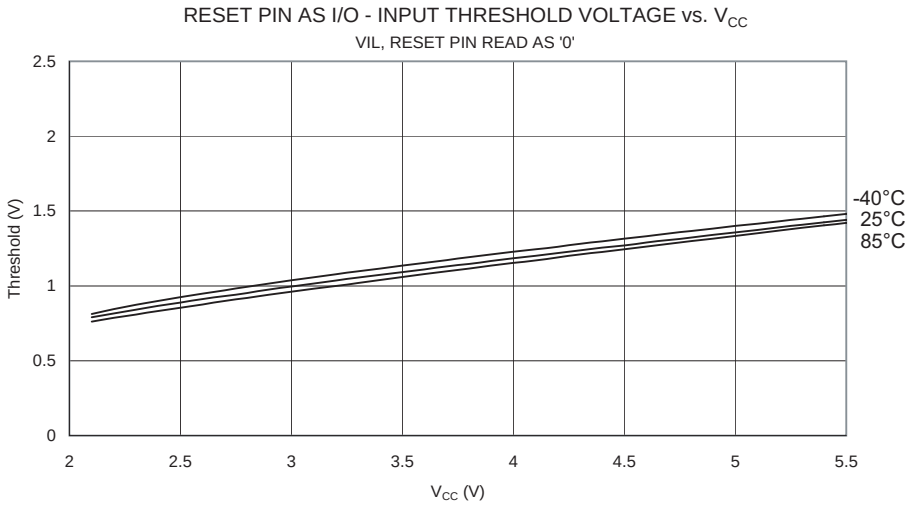


Figure 114. Reset 输入迟滞和 V_{CC} 的关系

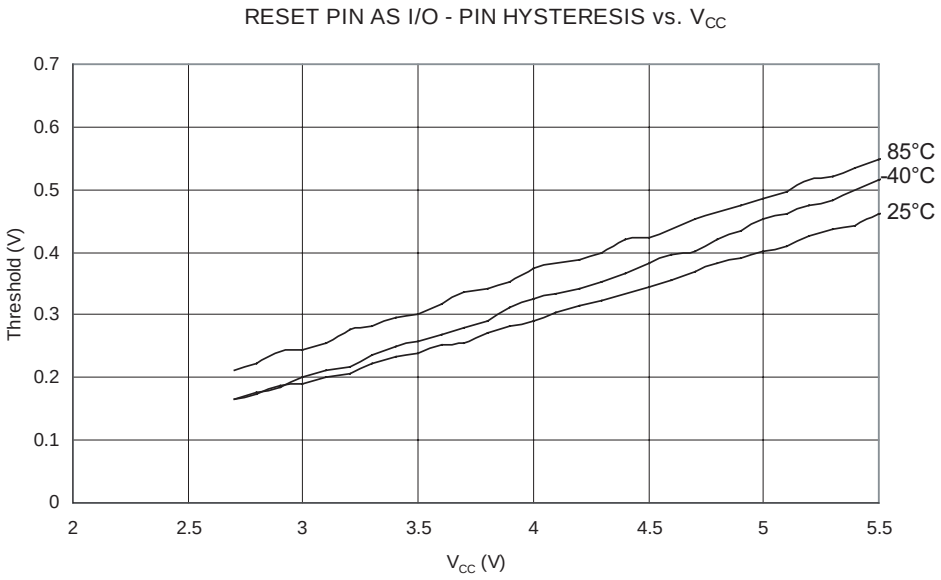


Figure 115. Reset 输入门限电压和 V_{CC} 的关系 (V_{IH},Reset 引脚读出值为 “1”)

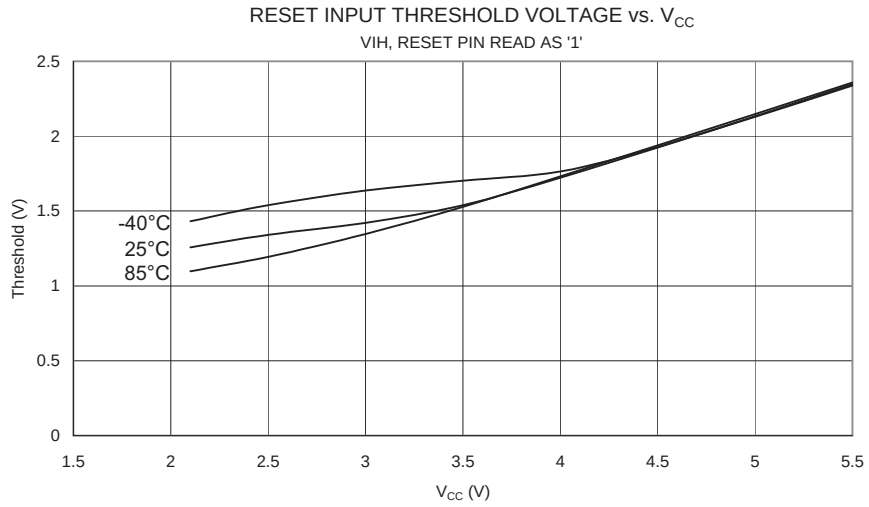


Figure 116. Reset 输入门限电压和 V_{CC} 的关系 (V_{IL},Reset 引脚读出值为 “0”)

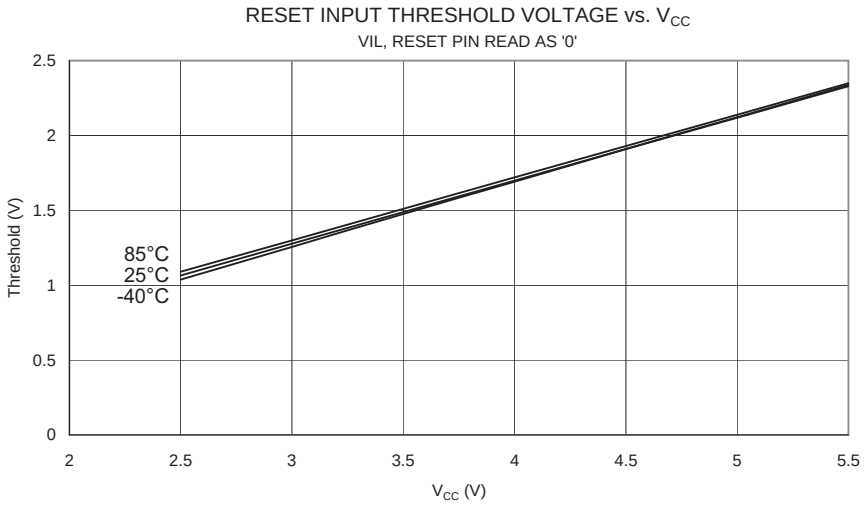
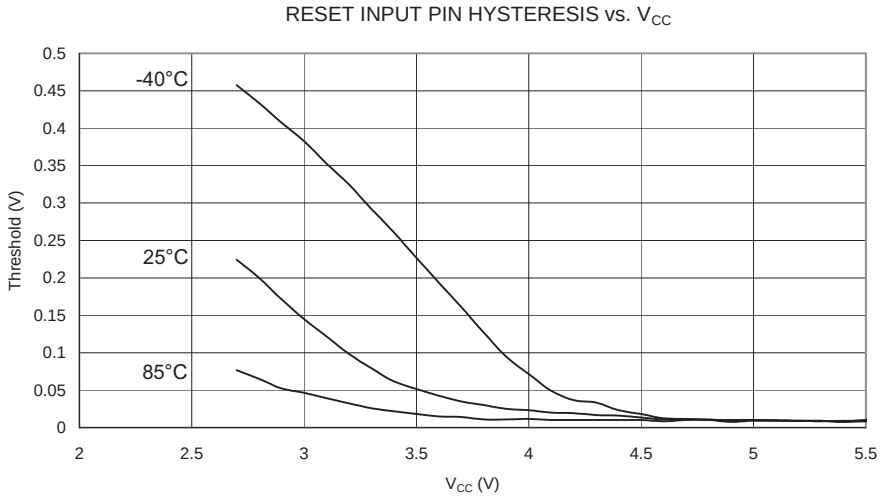


Figure 117. Reset 输入迟滞和 V_{CC} 的关系



BOD 门限值与模拟比较器偏移量

Figure 118. BOD 门限值和温度的关系 (BOD 电平为 4.0V)

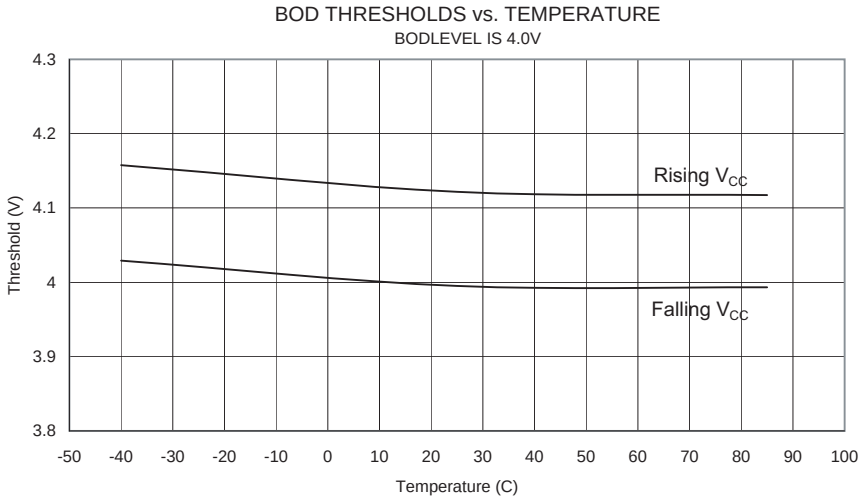


Figure 119. BOD 门限值和温度的关系 (BOD 电平为 2.7V)

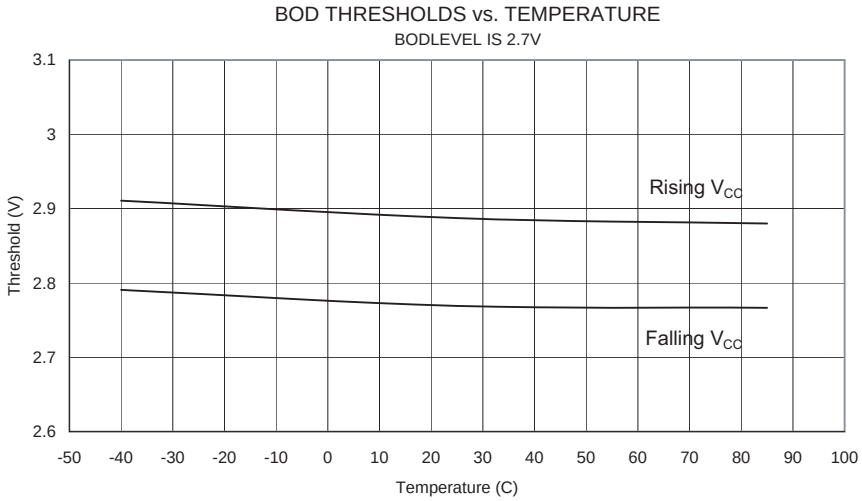


Figure 120. 能隙电压和 V_{CC} 的关系

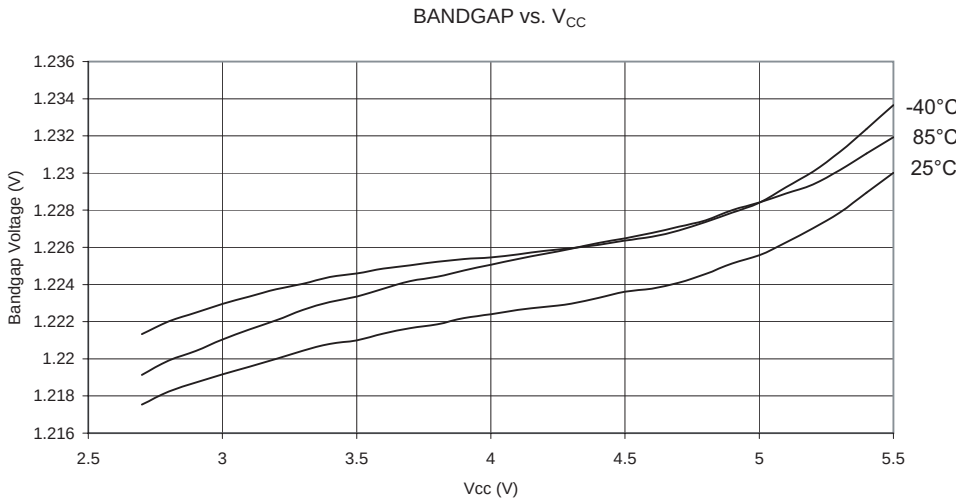


Figure 121. 模拟比较器偏置电压和共模电压的关系 (V_{CC}= 5.0V)

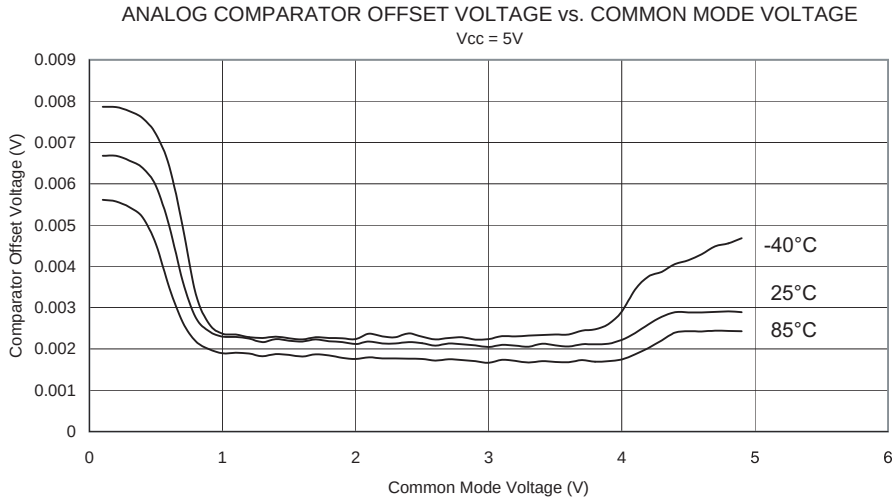
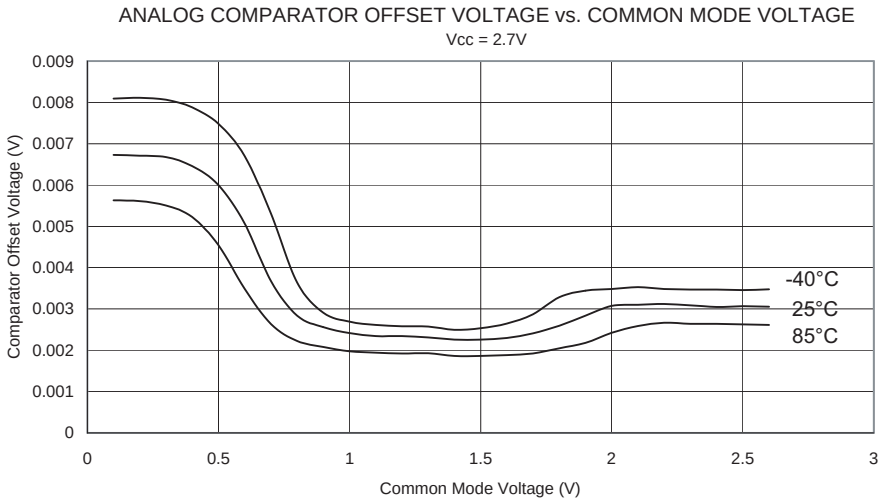


Figure 122. 模拟比较器偏置电压和共模电压的关系 (V_{CC}= 2.7V)



内部振荡器速度

Figure 123. 看门狗振荡器电流和 V_{CC} 的关系

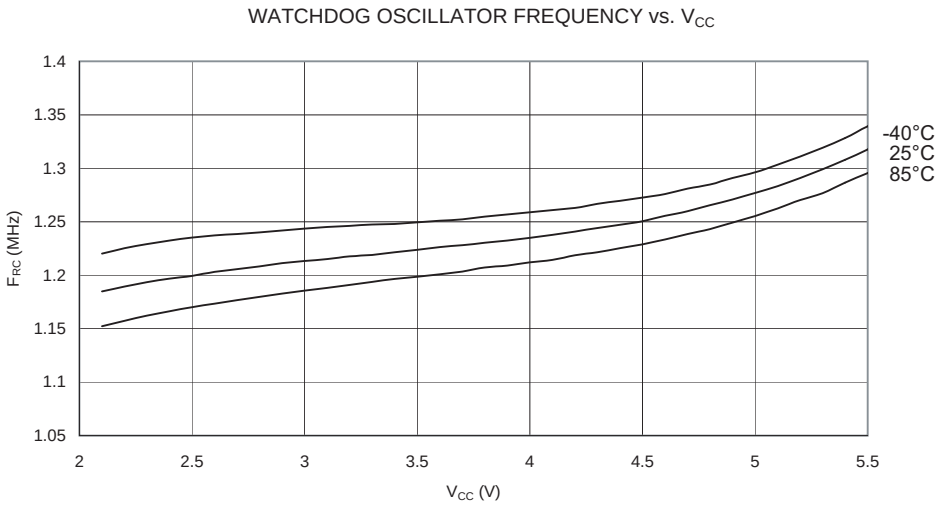


Figure 124. 标定 8 MHz RC 振荡器频率与温度的关系

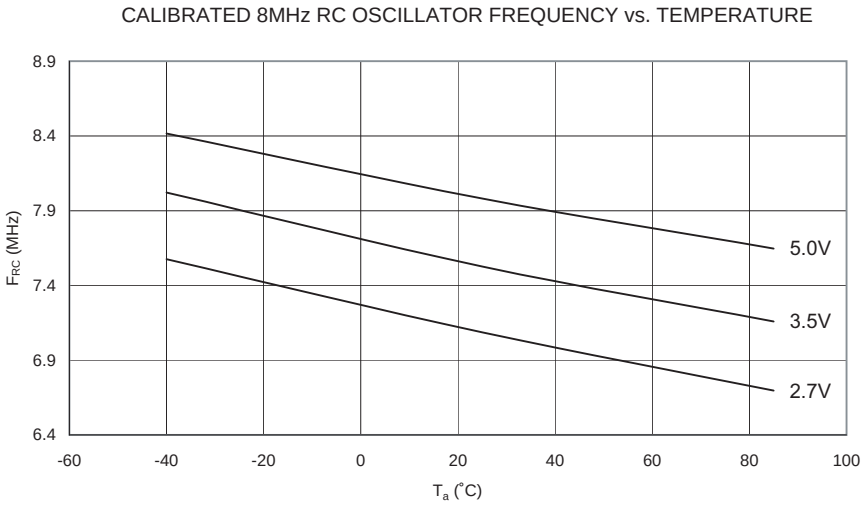


Figure 125. 标定 8 MHz RC 振荡器频率与 V_{CC} 的关系

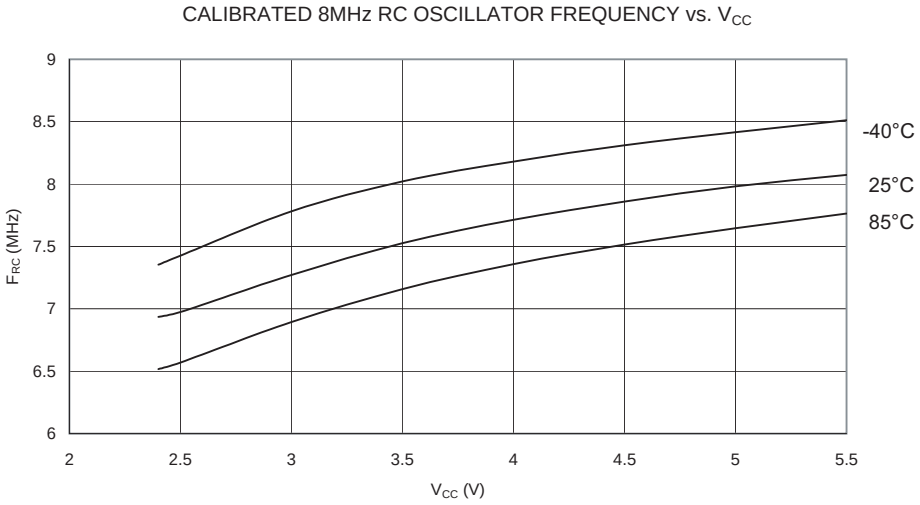


Figure 126. 标定 8 MHz RC 振荡器频率与 Osccal 值的关系

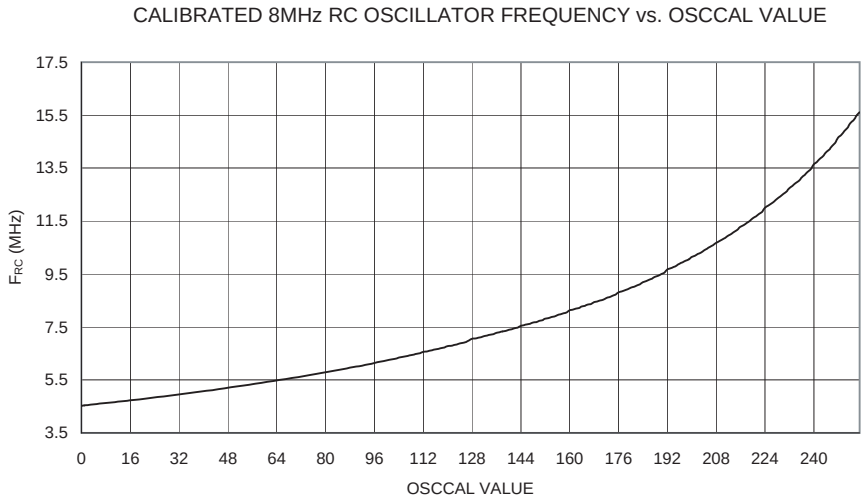


Figure 127. 标定 4 MHz RC 振荡器频率与温度的关系

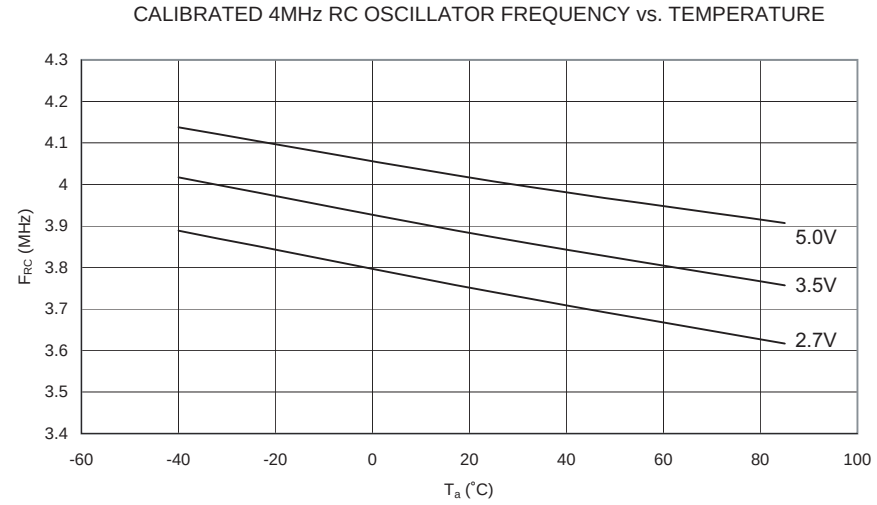


Figure 128. 标定 4 MHz RC 振荡器频率与 V_{CC} 的关系

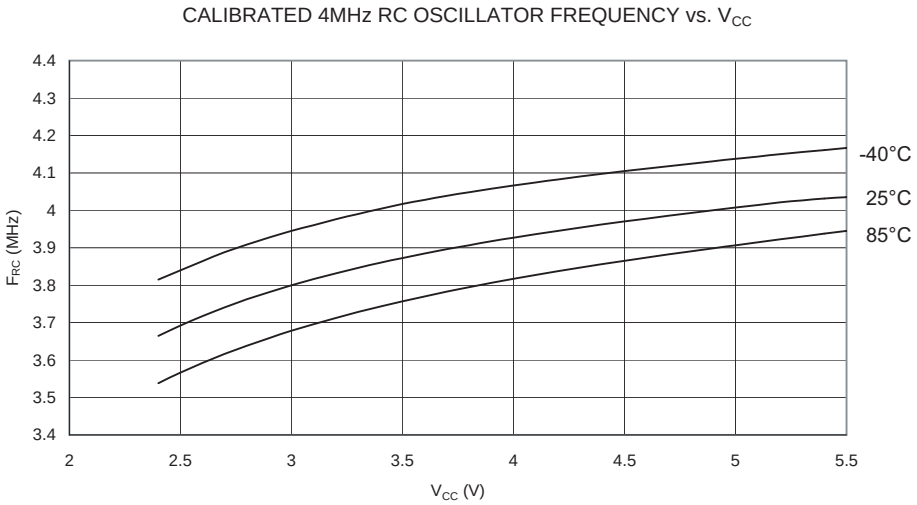


Figure 129. 标定 4 MHz RC 振荡器频率与 Osccal 值的关系

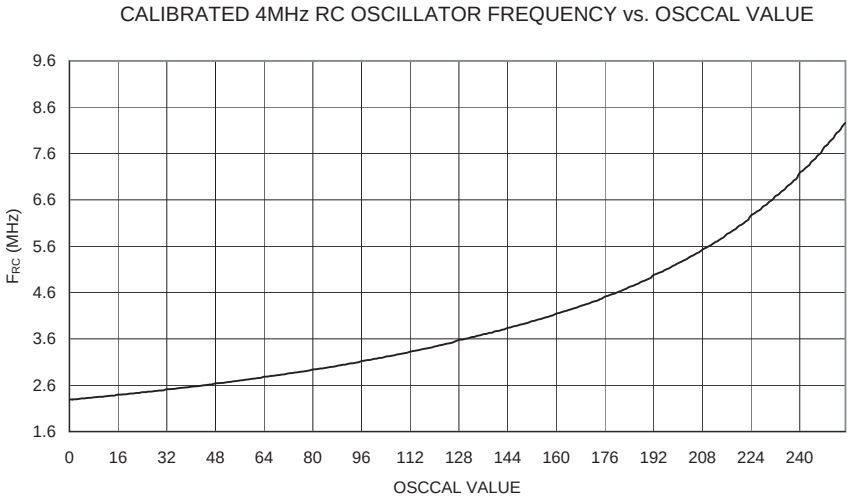


Figure 130. 标定 2 MHz RC 振荡器频率与温度的关系

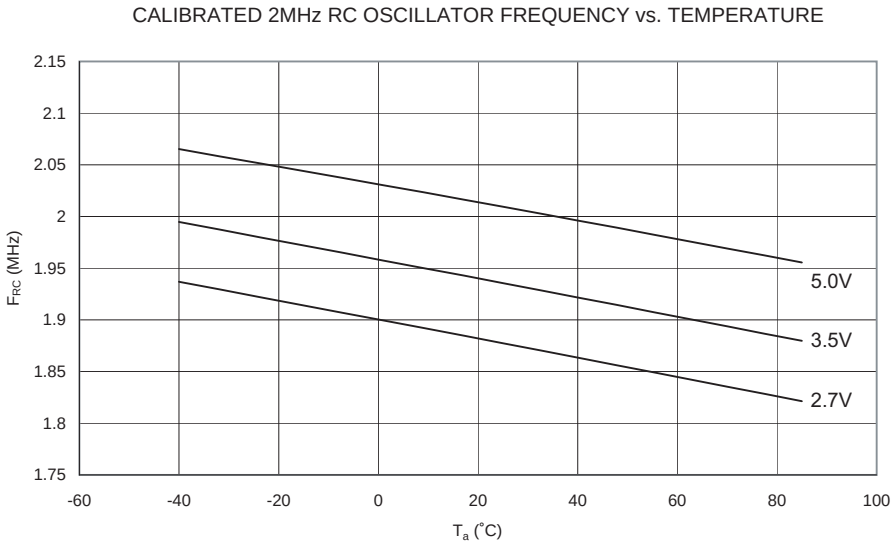


Figure 131. 标定 2 MHz RC 振荡器频率与 V_{CC} 的关系

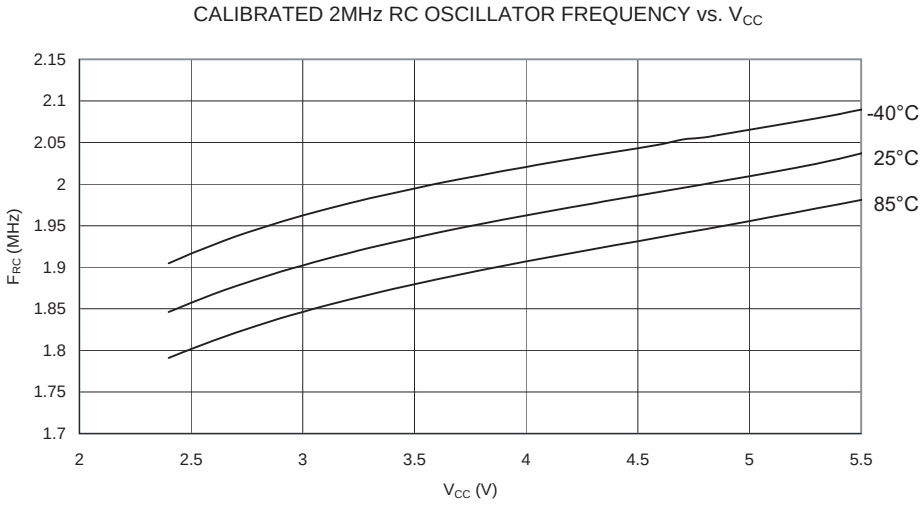


Figure 132. 标定 2 MHz RC 振荡器频率与 Osccal 值的关系

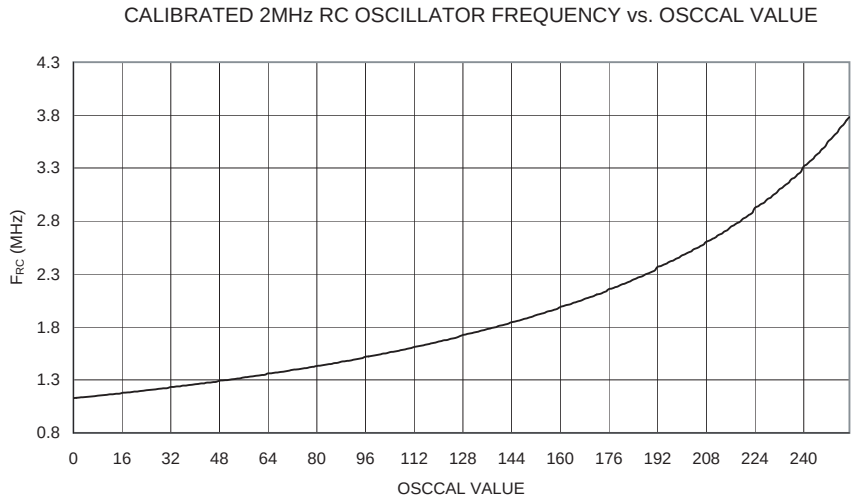


Figure 133. 标定 1 MHz RC 振荡器频率与温度的关系

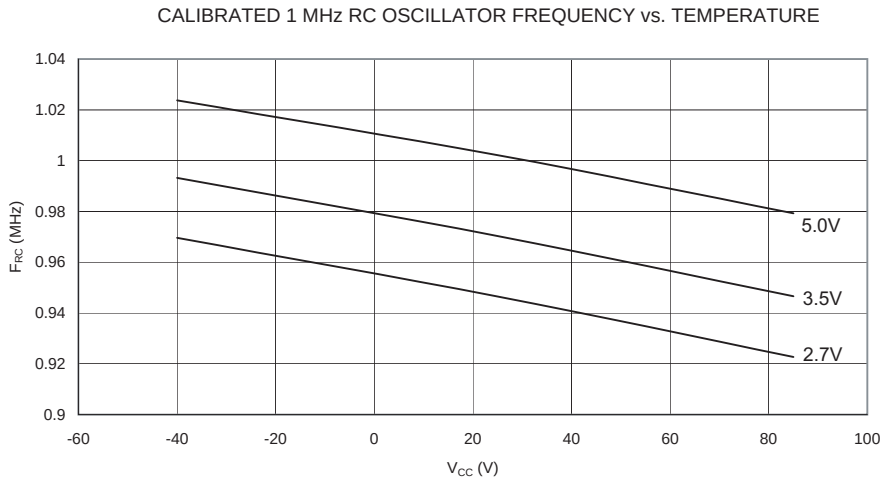


Figure 134. 标定 1 MHz RC 振荡器频率与 V_{CC} 的关系

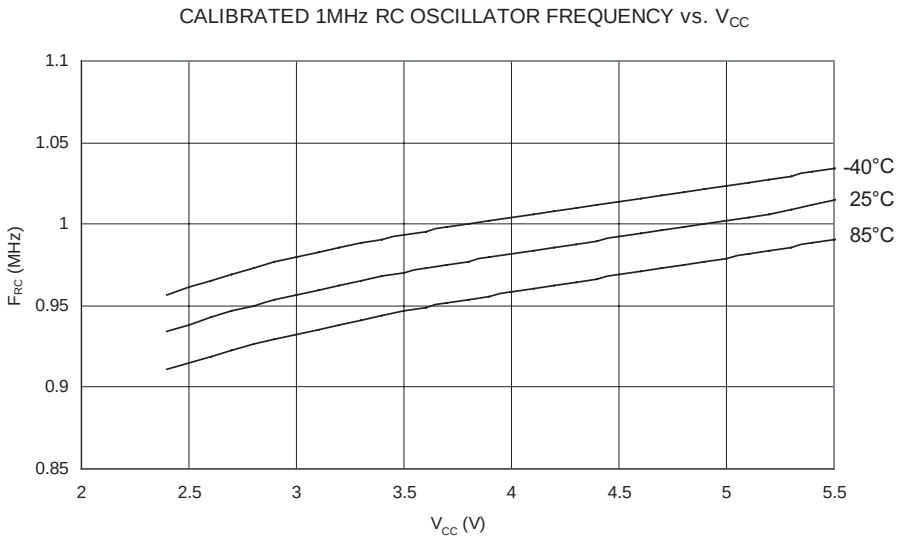


Figure 135. 标定 1 MHz RC 振荡器频率与 Osccal 值的关系

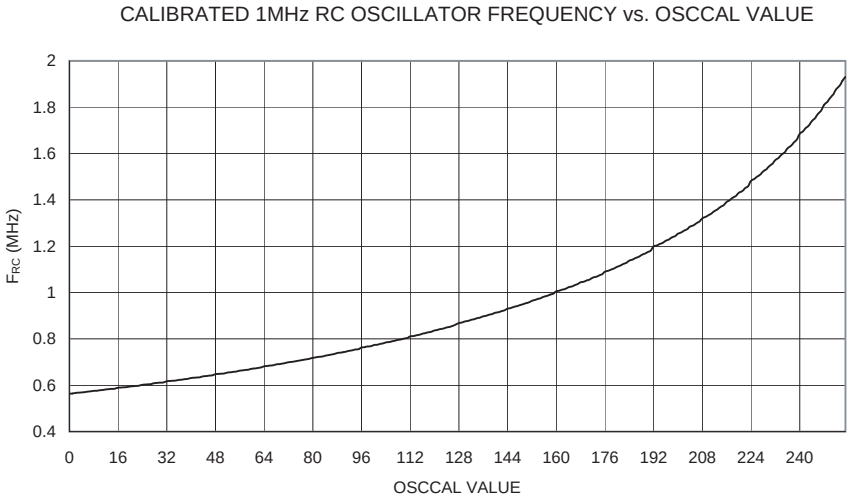


Figure 136. BOD 电流和 V_{CC} 的关系

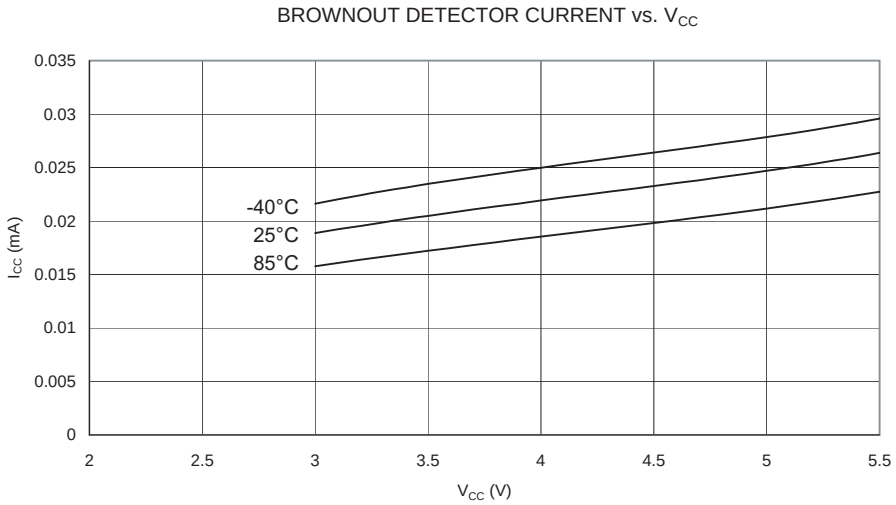


Figure 137. ADC 电流和 V_{CC} 的关系 (AREF = AV_{CC})

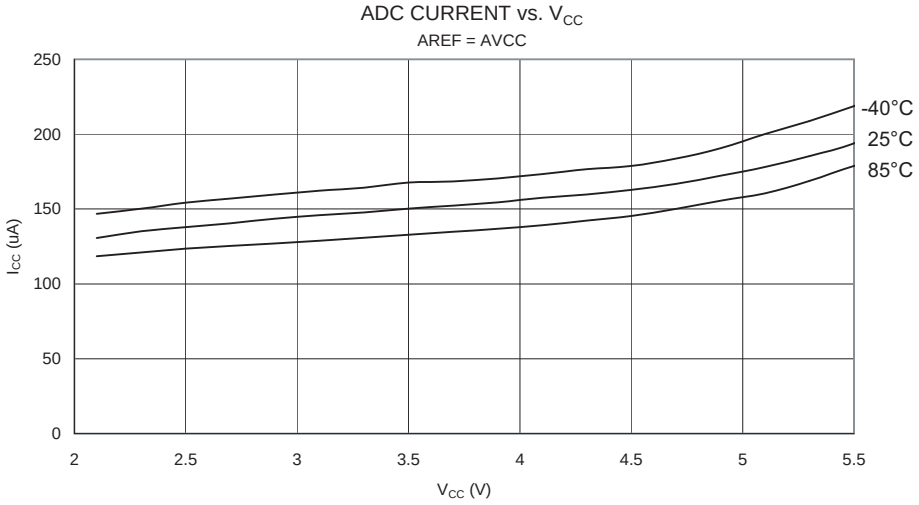


Figure 138. AREF 外部参考电流和 V_{CC} 的关系

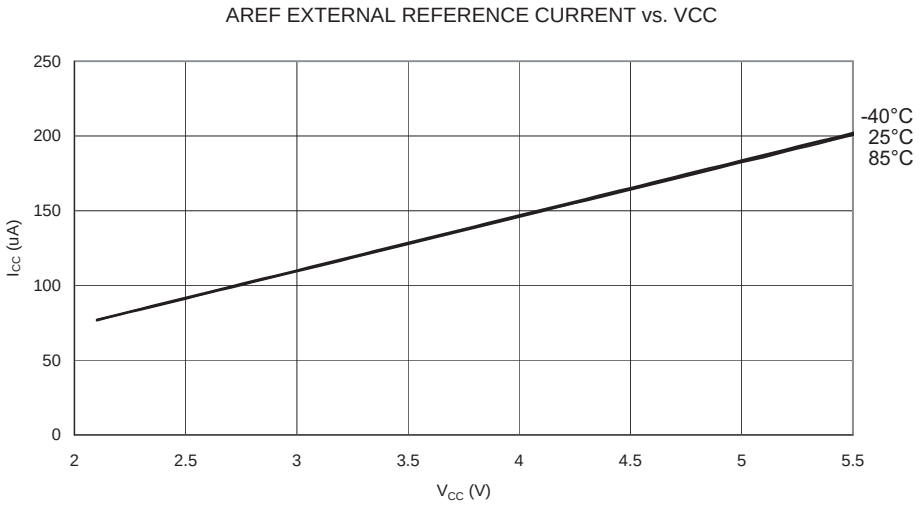


Figure 139. 模拟比较器电流和 V_{CC} 的关系

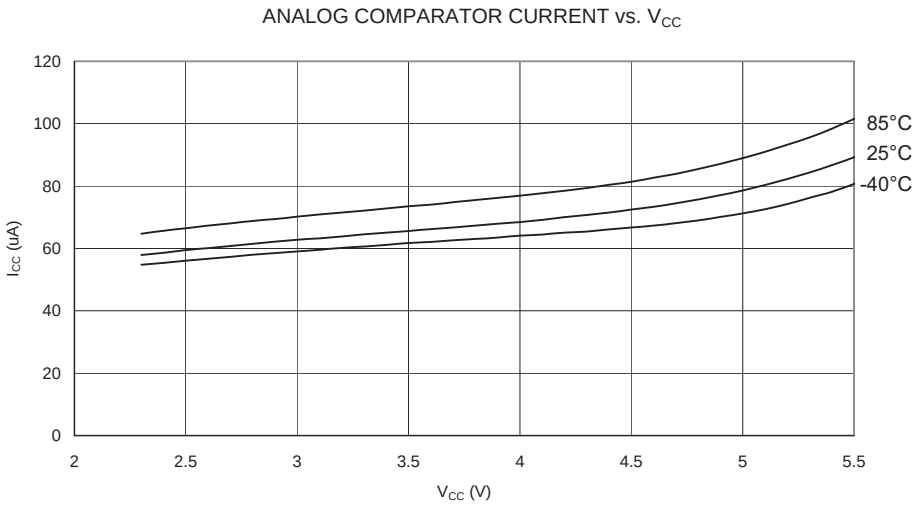
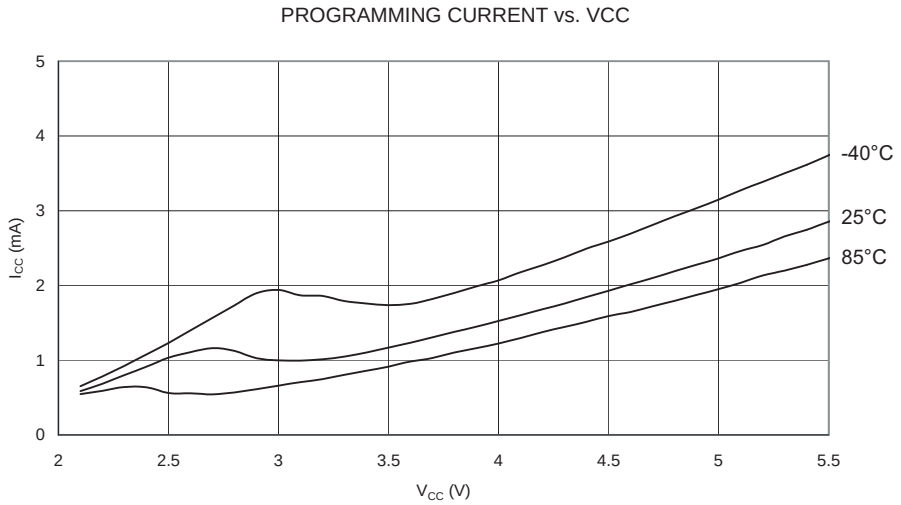


Figure 140. 编程电流与 V_{CC} 的关系



复位电流消耗及复位脉宽

Figure 141. 复位电流和 V_{CC} 的关系 (0.1 - 1.0 MHz，包括流经复位上拉电阻的电流)

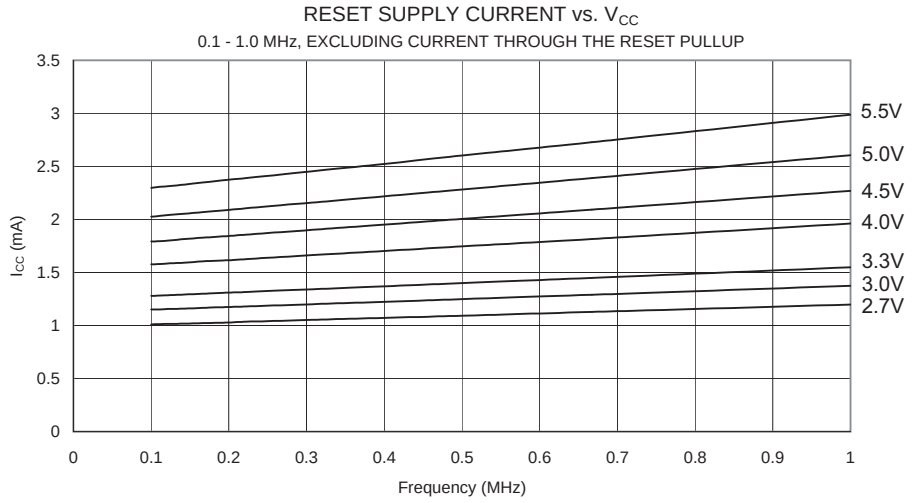


Figure 142. 复位电流和 V_{CC} 的关系 (1 - 20 MHz，包括流经复位上拉电阻的电流)

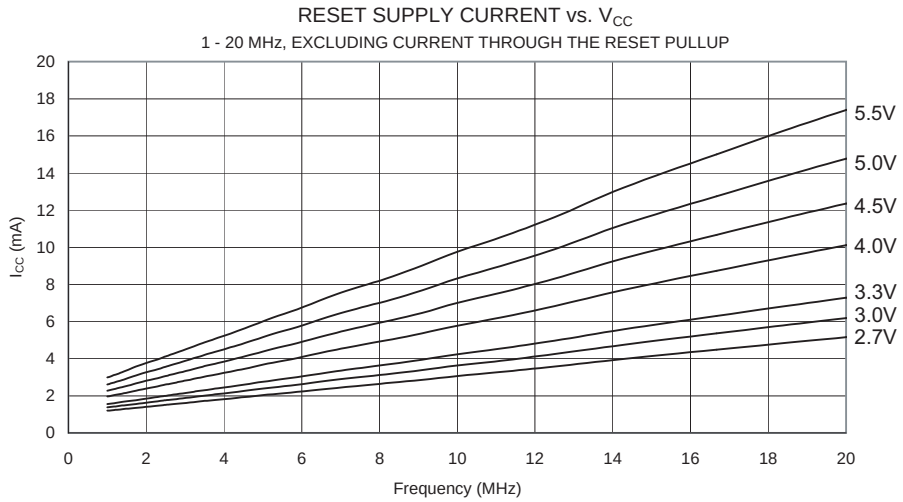
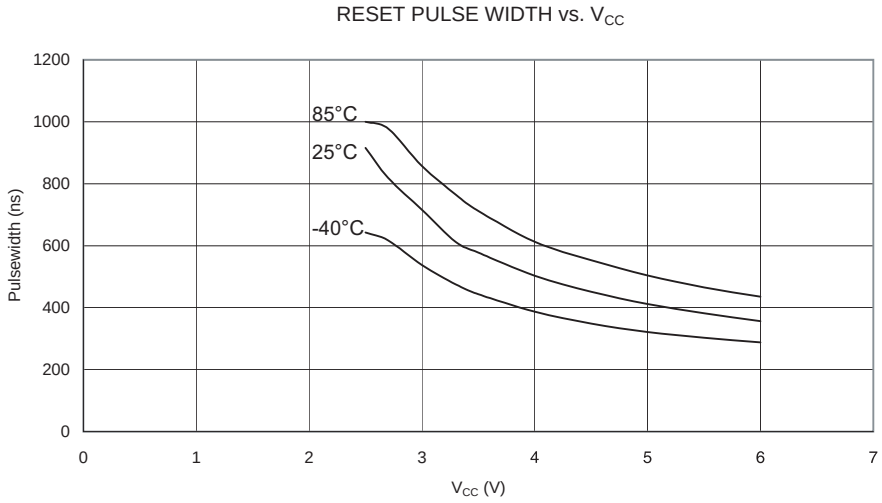


Figure 143. 复位脉宽和 V_{CC} 的关系



寄存器概述

地址	名称	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	页码
$3F ($5F)	SREG	I	T	H	S	V	N	Z	C	17
$3E ($5E)	保留									
$3D ($5D)	SP	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	18
$3C ($5C)	保留									
$3B ($5B)	GIMSK	-	INT0	PCIE1	PCIE0	-	-	-	-	33
$3A ($5A)	GIFR	-	INTF0	PCIF	-	-	-	-	-	34
$39 ($59)	TIMSK	-	OCIE1A	OCIE1B	-	-	TOIE1	TOIE0	-	34
$38 ($58)	TIFR	-	OCF1A	OCF1B	-	-	TOV1	TOV0	-	36
$37 ($57)	保留									
$36 ($56)	保留									
$35 ($55)	MCUCR	-	PUD	SE	SM1	SM0	-	ISC01	ISC00	38
$34 ($54)	MCUSR	-	-	-	-	WDRF	BORF	EXTRF	PORF	32
$33 ($53)	TCCR0	-	-	-	-	PSR0	CS02	CS01	CS00	44
$32 ($52)	TCNT0	T/C0 (8 位)								45
$31 ($51)	OSCCAL	振荡器标定寄存器								30
$30 ($50)	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	PWM1A	PWM1B	47
$2F ($4F)	TCCR1B	CTC1	PSR1	-	-	CS13	CS12	CS11	CS10	49
$2E ($4E)	TCNT1	T/C1 (8 位)								50
$2D ($4D)	OCR1A	T/C1 输出比较寄存器 A (8 位)								50
$2C ($4C)	OCR1B	T/C1 输出比较寄存器 B (8 位)								51
$2B ($4B)	OCR1C	T/C1 输出比较寄存器 C (8 位)								51
$2A ($4A)	保留									
$29 ($49)	PLLCSR	-	-	-	-	-	PCKE	PLLE	PLOCK	
$28 ($48)	保留									
$27 ($47)	保留									
$26 ($46)	保留									
$25 ($45)	保留									
$24 ($44)	保留									
$23 ($43)	保留									
$22 ($42)	保留									
$21 ($41)	WDTCR	-	-	-	WDCE	WDE	WDP2	WDP1	WDP0	56
$20 ($40)	保留									
$1F ($3F)	保留									
$1E ($3E)	EEAR	-	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	58
$1D ($3D)	EEDR	EEPROM 数据寄存器 (8 位)								58
$1C ($3C)	EECR	-	-	-	-	EERIE	EEMWE	EEWE	EERE	58
$1B ($3B)	PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	
$1A ($3A)	DDRA	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	
$19 ($39)	PINA	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	
$18 ($38)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	
$17 ($37)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	
$16 ($36)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	
$15 ($35)	保留									
$14 ($34)	保留									
$13 ($33)	保留									
$12 ($32)	保留									
$11 ($31)	保留									
$10 ($30)	保留									
$0F ($2F)	USIDR	通用串行接口数据寄存器 (8 位)								61
$0E ($2E)	USISR	USISIF	USIOIF	USIPF	USIDC	USICNT3	USICNT2	USICNT1	USICNT0	62
$0D ($2D)	USICR	USISIE	USIOIE	USIWM1	USIWM0	USICS1	USICS0	USICLK	USITC	63
$0C ($2C)	保留									
$0B ($2)B	保留									
$0A ($2A)	保留									
$09 ($29)	保留									
$08 ($28)	ACSR	ACD	ACBG	ACO	ACI	ACIE	ACME	ACIS1	ACIS0	72
$07 ($27)	ADMUX	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	82
$06 ($26)	ADCSR	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0	84
$05 ($25)	ADCH	ADC 数据寄存器高字节								85
$04 ($24)	ADCL	ADC 数据寄存器低字节								85
…	保留									
$00 ($20)	保留									

指令集概述

指令	操作数	说明	操作	标志	# 时钟数
		算术和逻辑指令			
ADD	Rd, Rr	无进位加法	Rd ← Rd + Rr	Z,C,N,V,H	1
ADC	Rd, Rr	带进位加法	Rd ← Rd + Rr + C	Z,C,N,V,H	1
ADIW	Rdl, K	立即数与字相加	Rdh:Rdl ← Rdh:Rdl + K	Z,C,N,V,S	2
SUB	Rd, Rr	无进位减法	Rd ← Rd - Rr	Z,C,N,V,H	1
SUBI	Rd, K	减立即数	Rd ← Rd - K	Z,C,N,V,H	1
SBC	Rd, Rr	带进位减法	Rd ← Rd - Rr - C	Z,C,N,V,H	1
SBCI	Rd, K	带进位减立即数	Rd ← Rd - K - C	Z,C,N,V,H	1
SBIW	Rdl, K	从字中减立即数	Rdh:Rdl ← Rdh:Rdl - K	Z,C,N,V,S	2
AND	Rd, Rr	逻辑与	Rd ← Rd • Rr	Z,N,V	1
ANDI	Rd, K	与立即数的逻辑与操作	Rd ← Rd • K	Z,N,V	1
OR	Rd, Rr	逻辑或	Rd ← Rd v Rr	Z,N,V	1
ORI	Rd, K	与立即数的逻辑或操作	Rd ← Rd v K	Z,N,V	1
EOR	Rd, Rr	异或	Rd ← Rd ⊕ Rr	Z,N,V	1
COM	Rd	1 的补码	Rd ← $FF - Rd	Z,C,N,V	1
NEG	Rd	2 的补码	Rd ← $00 - Rd	Z,C,N,V,H	1
SBR	Rd, K	设置寄存器的位	Rd ← Rd v K	Z,N,V	1
CBR	Rd, K	寄存器位清零	Rd ← Rd • ($FF - K)	Z,N,V	1
INC	Rd	加一操作	Rd ← Rd + 1	Z,N,V	1
DEC	Rd	减一操作	Rd ← Rd - 1	Z,N,V	1
TST	Rd	测试是否为零或负	Rd ← Rd • Rd	Z,N,V	1
CLR	Rd	寄存器清零	Rd ← Rd ⊕ Rd	Z,N,V	1
SER	Rd	寄存器置位	Rd ← $FF	None	1
		跳转指令			
RJMP	k	相对跳转	PC ← PC + k + 1	None	2
IJMP		间接跳转到 (Z)	PC ← Z	None	2
RCALL	k	相对子程序调用	PC ← PC + k + 1	None	3
ICALL		间接调用 (Z)	PC ← Z	None	3
RET		子程序返回	PC ← STACK	None	4
RETI		中断返回	PC ← STACK	I	4
CPSE	Rd, Rr	比较，相等则跳过下一条指令	if (Rd = Rr) PC ← PC + 2 or 3	None	1/2/3
CP	Rd, Rr	比较	Rd - Rr	Z,N,V,C,H	1
CPC	Rd, Rr	带进位比较	Rd - Rr - C	Z,N,V,C,H	1
CPI	Rd, K	与立即数比较	Rd - K	Z,N,V,C,H	1
SBRC	Rr, b	寄存器位为 "0" 则跳过下一条指令	if (Rr(b) = 0) PC ← PC + 2 or 3	None	1/2/3
SBRS	Rr, b	寄存器位为 "1" 则跳过下一条指令	if (Rr(b) = 1) PC ← PC + 2 or 3	None	1/2/3
SBIC	P, b	I/O 寄存器位为 "0" 则跳过下一条指令	if (P(b) = 0) PC ← PC + 2 or 3	None	1/2/3
SBIS	P, b	I/O 寄存器位为 "1" 则跳过下一条指令	if (P(b) = 1) PC ← PC + 2 or 3	None	1/2/3
BRBS	s, k	状态寄存器位为 "1" 则跳过下一条指令	if (SREG(s) = 1) then PC ← PC + k + 1	None	1/2
BRBC	s, k	状态寄存器位为 "0" 则跳过下一条指令	if (SREG(s) = 0) then PC ← PC + k + 1	None	1/2
BREQ	k	相等则跳转	if (Z = 1) then PC ← PC + k + 1	None	1/2
BRNE	k	不相等则跳转	if (Z = 0) then PC ← PC + k + 1	None	1/2
BRCS	k	进位位为 "1" 则跳转	if (C = 1) then PC ← PC + k + 1	None	1/2
BRCC	k	进位位为 "0" 则跳转	if (C = 0) then PC ← PC + k + 1	None	1/2
BRSH	k	大于或等于则跳转	if (C = 0) then PC ← PC + k + 1	None	1/2
BRLO	k	小于则跳转	if (C = 1) then PC ← PC + k + 1	None	1/2
BRMI	k	负则跳转	if (N = 1) then PC ← PC + k + 1	None	1/2
BRPL	k	正则跳转	if (N = 0) then PC ← PC + k + 1	None	1/2
BRGE	k	有符号数大于或等于则跳转	if (N ⊕ V = 0) then PC ← PC + k + 1	None	1/2
BRLT	k	有符号数负则跳转	if (N ⊕ V = 1) then PC ← PC + k + 1	None	1/2
BRHS	k	半进位位为 "1" 则跳转	if (H = 1) then PC ← PC + k + 1	None	1/2
BRHC	k	半进位位为 "0" 则跳转	if (H = 0) then PC ← PC + k + 1	None	1/2
BRTS	k	T 为 "1" 则跳转	if (T = 1) then PC ← PC + k + 1	None	1/2
BRTC	k	T 为 "0" 则跳转	if (T = 0) then PC ← PC + k + 1	None	1/2
BRVS	k	溢出标志为 "1" 则跳转	if (V = 1) then PC ← PC + k + 1	None	1/2
BRVC	k	溢出标志为 "0" 则跳转	if (V = 0) then PC ← PC + k + 1	None	1/2
BRIE	k	中断使能则跳转	if (I = 1) then PC ← PC + k + 1	None	1/2
BRID	k	中断禁止则跳转	if (I = 0) then PC ← PC + k + 1	None	1/2
		数据传送指令			
MOV	Rd, Rr	寄存器间复制	Rd ← Rr	None	1
LDI	Rd, K	加载立即数	Rd ← K	None	1
LD	Rd, X	加载间接寻址数据	Rd ← (X)	None	2
LD	Rd, X+	加载间接寻址数据，然后地址加一	Rd ← (X), X ← X + 1	None	2
LD	Rd, -X	地址减一后加载间接寻址数据	X ← X - 1, Rd ← (X)	None	2

指令集概述 (Continued)

指令	操作数	说明	操作	标志	# 时钟数
LD	Rd, Y	加载间接寻址数据	Rd ← (Y)	None	2
LD	Rd, Y+	加载间接寻址数据，然后地址加一	Rd ← (Y), Y ← Y + 1	None	2
LD	Rd, -Y	地址减一后加载间接寻址数据	Y ← Y - 1, Rd ← (Y)	None	2
LDD	Rd,Y+q	加载带偏移量的间接寻址数据	Rd ← (Y + q)	None	2
LD	Rd, Z	加载间接寻址数据	Rd ← (Z)	None	2
LD	Rd, Z+	加载间接寻址数据，然后地址加一	Rd ← (Z), Z ← Z + 1	None	2
LD	Rd, -Z	地址减一后加载间接寻址数据	Z ← Z - 1, Rd ← (Z)	None	2
LDD	Rd, Z+q	加载带偏移量的间接寻址数据	Rd ← (Z + q)	None	2
LDS	Rd, k	从 SRAM 加载数据	Rd ← (k)	None	2
ST	X, Rr	以间接寻址方式存储数据	(X) ← Rr	None	2
ST	X+, Rr	以间接寻址方式存储数据，然后地址加一	(X) ← Rr, X ← X + 1	None	2
ST	-X, Rr	地址减一后以间接寻址方式存储数据	X ← X - 1, (X) ← Rr	None	2
ST	Y, Rr	加载间接寻址数据	(Y) ← Rr	None	2
ST	Y+, Rr	加载间接寻址数据，然后地址加一	(Y) ← Rr, Y ← Y + 1	None	2
ST	-Y, Rr	地址减一后加载间接寻址数据	Y ← Y - 1, (Y) ← Rr	None	2
STD	Y+q, Rr	加载带偏移量的间接寻址数据	(Y + q) ← Rr	None	2
ST	Z, Rr	加载间接寻址数据	(Z) ← Rr	None	2
ST	Z+, Rr	加载间接寻址数据，然后地址加一	(Z) ← Rr, Z ← Z + 1	None	2
ST	-Z, Rr	地址减一后加载间接寻址数据	Z ← Z - 1, (Z) ← Rr	None	2
STD	Z+q, Rr	加载带偏移量的间接寻址数据	(Z + q) ← Rr	None	2
STS	k, Rr	从 SRAM 加载数据	(k) ← Rr	None	2
LPM		加载程序空间的数据	R0 ← (Z)	None	3
LPM	Rd, Z	加载程序空间的数据	Rd ← (Z)	None	3
IN	Rd, P	从 I/O 端口读数据	Rd ← P	None	1
OUT	P, Rr	想 I/O 端口输出数据	P ← Rr	None	1
PUSH	Rr	将寄存器推入堆栈	STACK ← Rr	None	2
POP	Rd	将寄存器弹出堆栈	Rd ← STACK	None	2
		位和位测试指令			
SBI	P, b	I/O 寄存器位置位	I/O(P,b) ← 1	None	2
CBI	P, b	I/O 寄存器位清零	I/O(P,b) ← 0	None	2
LSL	Rd	逻辑左移	Rd(n+1) ← Rd(n), Rd(0) ← 0	Z,C,N,V	1
LSR	Rd	逻辑右移	Rd(n) ← Rd(n+1), Rd(7) ← 0	Z,C,N,V	1
ROL	Rd	带进位循环左移	Rd(0) ← C, Rd(n+1) ← Rd(n), C ← Rd(7)	Z,C,N,V	1
ROR	Rd	带进位循环右移	Rd(7) ← C, Rd(n) ← Rd(n+1), C ← Rd(0)	Z,C,N,V	1
ASR	Rd	算术右移	Rd(n) ← Rd(n+1), n = 0..6	Z,C,N,V	1
SWAP	Rd	高低半字节交换	Rd(3..0) ← Rd(7..4), Rd(7..4) ← Rd(3..0)	None	1
BSET	s	标志置位	SREG(s) ← 1	SREG(s)	1
BCLR	s	标志清零	SREG(s) ← 0	SREG(s)	1
BST	Rr, b	从寄存器将位赋给 T	T ← Rr(b)	T	1
BLD	Rd, b	将 T 赋给寄存器位	Rd(b) ← T	None	1
SEC		进位位置位	C ← 1	C	1
CLC		进位位清零	C ← 0	C	1
SEN		负标志位置位	N ← 1	N	1
CLN		负标志位清零	N ← 0	N	1
SEZ		零标志位置位	Z ← 1	Z	1
CLZ		零标志位清零	Z ← 0	Z	1
SEI		全局中断使能	I ← 1	I	1
CLI		全局中断禁用	I ← 0	I	1
SES		符号测试标志位置位	S ← 1	S	1
CLS		符号测试标志位清零	S ← 0	S	1
SEV		2 的补码溢出标志置位	V ← 1	V	1
CLV		2 的补码溢出标志清零	V ← 0	V	1
SET		SREG 的 T 置位	T ← 1	T	1
CLT		SREG 的 T 清零	T ← 0	T	1
SEH		SREG 的半进位标志置位	H ← 1	H	1
CLH		SREG 的半进位标志清零	H ← 0	H	1
NOP		休眠		无	1
SLEEP		复位看门狗	(见 休眠函数的说明)	无	1
WDR		终止	(见 WDR/ 定时器的说明)	无	1

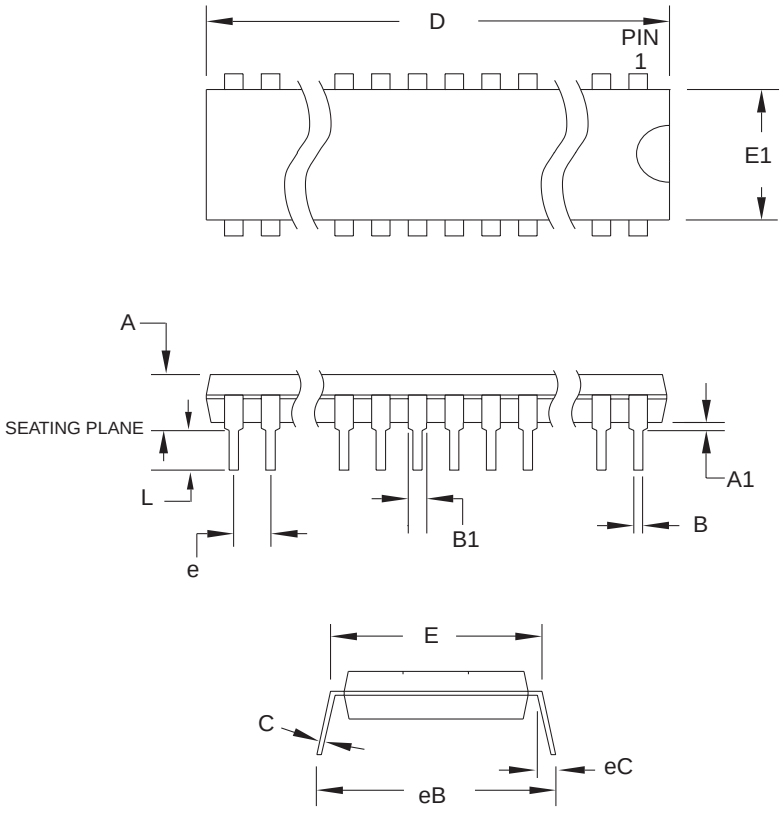
产品信息 (1)

速度 (MHz)	所需电源	产品号	封装	工作范围
8	2.7 - 5.5V	ATtiny26L-8PC ATtiny26L-8SC ATtiny26L-8MC	20P3 20S 32M1-A	商业级 (0°C - 70°C)
		ATtiny26L-8PI ATtiny26L-8SI ATtiny26L-8MI	20P3 20S 32M1-A	工业级 (-40°C - 85°C)
16	4.5 - 5.5V	ATtiny26-16PC ATtiny26-16SC ATtiny26-16MC	20P3 20S 32M1-A	商业级 (0°C - 70°C)
		ATtiny26-16PI ATtiny26-16SI ATtiny26-16MI	20P3 20S 32M1-A	工业级 (-40°C - 85°C)

Note: 1. 产品也可以 wafer 的形式提供，订货信息细节以及最小定货量请与 Atmel 当地机构联系。

封装类型	
20P3	20 引脚， 0.300" 宽， PDIP 封装
20S	20 引脚， 0.300" 宽， SOIC 封装
32M1-A	32 焊盘， 5 x 5 x 1.0， MLF 封装

20P3



COMMON DIMENSIONS
(Unit of Measure = mm)

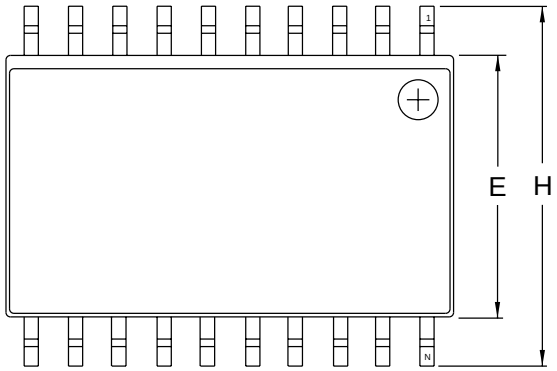
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	5.334	
A1	0.381	–	–	
D	25.493	–	25.984	Note 2
E	7.620	–	8.255	
E1	6.096	–	7.112	Note 2
B	0.356	–	0.559	
B1	1.270	–	1.551	
L	2.921	–	3.810	
C	0.203	–	0.356	
eB	–	–	10.922	
eC	0.000	–	1.524	
e	2.540 TYP			

Notes:
1. This package conforms to JEDEC reference MS-001, Variation AD.
2. Dimensions D and E1 do not include mold Flash or Protrusion. Mold Flash or Protrusion shall not exceed 0.25 mm (0.010").

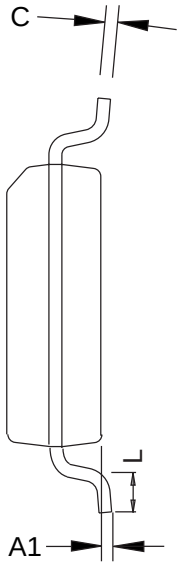
1/12/04

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**20P3**, 20-lead (0.300"/7.62 mm Wide) Plastic Dual Inline Package (PDIP)	20P3	C

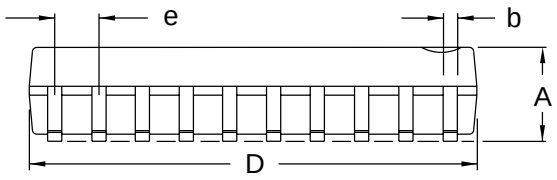
20S



Top View



End View



Side View

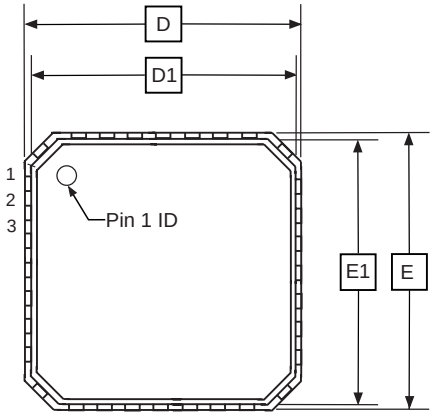
COMMON DIMENSIONS
(Unit of Measure = inches)

SYMBOL	MIN	NOM	MAX	NOTE
A	0.0926		0.1043	
A1	0.0040		0.0118	
b	0.0130		0.0200	4
C	0.0091		0.0125	
D	0.4961		0.5118	1
E	0.2914		0.2992	2
H	0.3940		0.4190	
L	0.0160		0.050	3
e	0.050 BSC			

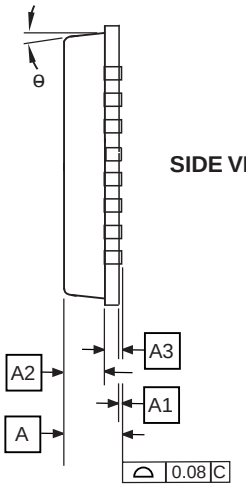
Notes:
1. This drawing is for general information only; refer to JEDEC Drawing MS-013, Variation AC for additional information.
2. Dimension "D" does not include mold Flash, protrusions or gate burrs. Mold Flash, protrusions and gate burrs shall not exceed 0.15 mm (0.006") per side.
3. Dimension "E" does not include inter-lead Flash or protrusion. Inter-lead Flash and protrusions shall not exceed 0.25 mm (0.010") per side.
4. "L" is the length of the terminal for soldering to a substrate.
5. The lead width "b", as measured 0.36 mm (0.014") or greater above the seating plane, shall not exceed a maximum value of 0.61 mm (0.024") per side.

1/9/02

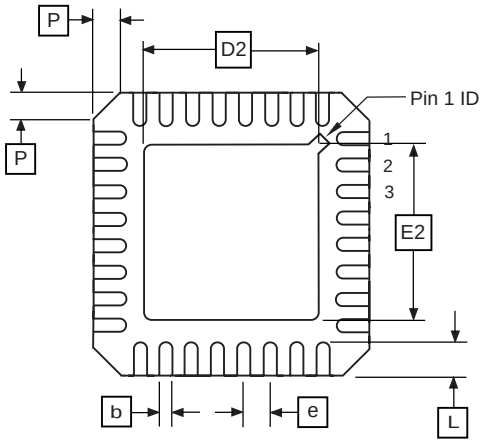
	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**20S2**, 20-lead, 0.300" Wide Body, Plastic Gull Wing Small Outline Package (SOIC)	20S2	A



TOP VIEW



SIDE VIEW



BOTTOM VIEW

COMMON DIMENSIONS
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	0.80	0.90	1.00	
A1	–	0.02	0.05	
A2	–	0.65	1.00	
A3		0.20 REF		
b	0.18	0.23	0.30	
D		5.00 BSC		
D1		4.75 BSC		
D2	2.95	3.10	3.25	
E		5.00 BSC		
E1		4.75BSC		
E2	2.95	3.10	3.25	
e		0.50 BSC		
L	0.30	0.40	0.50	
P	–	–	0.60	
θ	–	–	12^{o}	

Notes: 1. JEDEC Standard MO-220, Fig. 2 (Anvil Singulation), VHHD-2.

01/15/03

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**32M1-A**, 32-pad, 5 x 5 x 1.0 mm Body, Lead Pitch 0.50 mm Micro Lead Frame Package (MLF)	32M1-A	C

ATtiny26 数据手册改变日志

本节所指的页号为本文的相关页码，修订号则为文档的修订号。

从版本 Rev. 1477D-05/03 到版本 Rev. 1477E-10/03 的改动

1. 从数据手册中删除“ 预报” 部分
2. 更新 “Features” on page 1 部分。
3. 从 P2“ 引脚配置 ” 中删除 SSOP 封装参考。
4. 在 P19Table 3 中更新 V_{RST} 与 t_{RST}。
5. 更新 P30“ 标定的片内 RC 振荡器 ” 。
6. 更新 P122“ 电气特性 ” 中有关 V_{OL}, I_{IL}, I_{IH}, I_{CC} 掉电与 V_{ACIO} 的直流特性。
7. 更新P125“ADC 特性参数” 与P122中V_{INT}, INL 及增益误差。确定P122“绝对精度” 中排版错误。
8. 在 P142“驱动能力” 中增加Figure106 ； 在P151“BOD门限值与模拟比较器偏移量” 中增加 Figure120, Figure121 与 Figure122。更新 Figure117 与 Figure118。
9. 从 P161“ 指令集概述 ” 中删除 LPM Rd, Z+ 指令。ATtiny26 不支持该指令。

从版本 Rev. 1477C-09/02 到版本 Rev. 1477D-05/03 的改变

1. 更新 P168“ 封装信息 ” 。
2. 从 P125“ADC 特性参数 ” 中删除 ADHSM。
3. 添加 P60“ 掉电模式下的 EEPROM 写操作 ” 部分。
4. 添加 P27“ 缺省时钟源 ” 部分。
5. 校正 P49“Bit 0 – PLOCK: PLL 锁定探测器 ” 中 PLL 锁定值。
6. 在 P73 添加选择差分通道时的转换时间。
7. 修正 P84 中 {DDxn, PORTxn} 值。
8. 添加 P91“ 未连接引脚的处理 ” 部分。
9. P100Table 49 中添加有关 RSTDISBL 熔丝位的注意事项。
10. 修正 P108Figure61 DATA 值。
11. 在 P115Table 59 中添加 WD_FUSE 周期。
12. 更新 P125“ADC 特性参数” ；添加P 122Table 64, “ 差分通道下 ADC 特性参数 , T_A = -40°C 到 85°C,” 。
13. 更新 P127“ATtiny26 典型特征参数 ” 。
14. P161“ 指令集概述 ” 中添加 LPM Rd, Z 与 LPM Rd, Z+ 指令。

从版本 Rev. 1477B-04/02 到版本 Rev. 1477C-09/02 的改变

1. 改变 Flash 的擦写寿命为 10,000 次。

从版本 Rev. 1477A-03/02 到版本 Rev. 1477B-04/02 的改变

1. 在 P24“ 系统时钟及时钟选项 ” 中删除所有省电模式部分的内容。
2. 更新 P75“ 模数转换器 ” 部分，添加在差分与单端转换中如何读取转换结果。
3. 更新 P163“ 产品信息 (1)” ，添加 MLF 封装信息。

Atmel Corporation

2325 Orchard Parkway
San Jose, CA 95131
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

Regional Headquarters

Europe

Atmel Sarl
Route des Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
Tel: (41) 26-426-5555
Fax: (41) 26-426-5500

Asia

Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

Japan

9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Atmel Operations

Memory

2325 Orchard Parkway
San Jose, CA 95131
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

Microcontrollers

2325 Orchard Parkway
San Jose, CA 95131
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
Tel: (33) 2-40-18-18-18
Fax: (33) 2-40-18-19-60

ASIC/ASSP/Smart Cards

Zone Industrielle
13106 Rousset Cedex, France
Tel: (33) 4-42-53-60-00
Fax: (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
Tel: (44) 1355-803-000
Fax: (44) 1355-242-743

RF/Automotive

Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
Tel: (49) 71-31-67-0
Fax: (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Biometrics/Imaging/Hi-Rel MPU/ High Speed Converters/RF Data-com

Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80

e-mail

literature@atmel.com

Web Site

http://www.atmel.com

1477E–AVR–12/03 0M