AT90S2313

特点

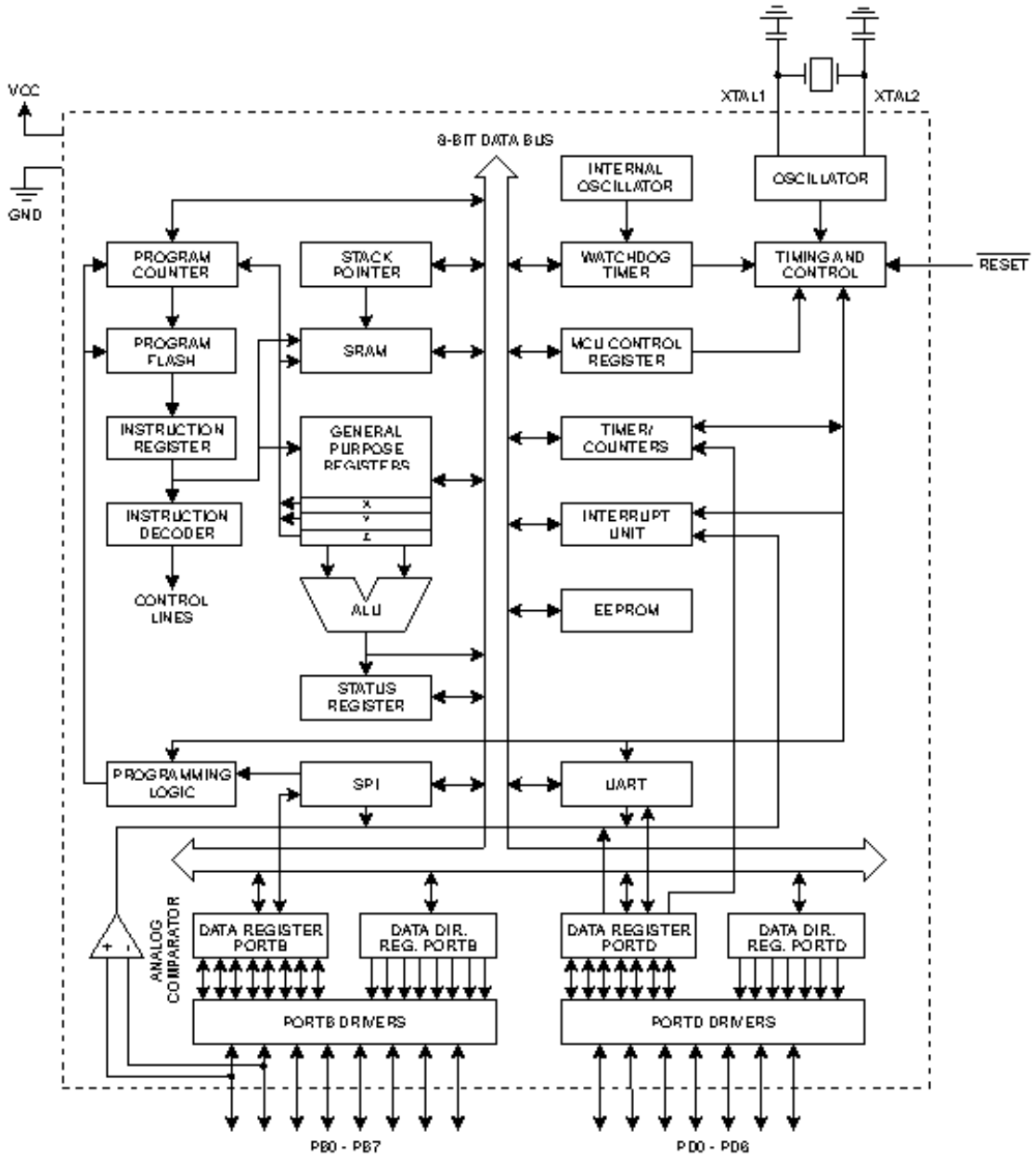
1. AVR RISC 结构
2. AVR—高性能、低功耗 RISC 结构
   —118 条指令——大多数为单指令周期
   —32 个 8 位通用（工作）寄存器
   —工作在 10MHz 时具有 10MIPS 的性能
3. 数据和非易失性程序内存
   —2K 字节的在线可编程 FLASH（擦除次数：1000 次）
   —128 字节 SRAM
   —128 字节在线可编程 EEPROM（寿命：100000 次）
   —程序加密位
4. 外围（Peripheral）特点
   ——一个可预分频（Prescale）的 8 位定时器/计数器
   ——一个可预分频、具有比较、捕捉和 8-，9-，10 位 PWM 功能的 16 位定时器/计数器
   —片内模拟比较器
   —可编程的看门狗定时器（由片内振荡器生成）
   —用于下载程序的 SPI 口
   —全双工 UART
5. 特别的 MCU 特点
   —低功耗空闲和掉电模式
   —内外部中断源
6. 规范（Specification）
   —低功耗、高速 CMOS 工艺
   —全静态工作
7. 4MHz、3V、25℃条件下的功耗：
   —工作模式：2.8mA
   —空闲模式：0.8mA
   —掉电模式：<1μA
8. I/O 和封装
   —15 个可编程的 I/O 脚
   —20 脚 PDIP 和 SOIC 封装
9. 工作电压
   —2.7V-6.0V（AT90S2313-4）
   —4.0V-6.0V（AT90S2313-10）
10. 速度
    —0-4MHz（AT90S2313-4）
    —0-10MHz（AT90S2313-10）

描述

AT90S2313 是一款基于 AVR RISC 的低功耗 CMOS 的 8 位单片机。通过在一个时钟周期内执行一条指令，AT90S2313 可以取得接近 1MIPS/MHz 的性能，从而使得设计人员可以在功耗和执行速度之间取得平衡。

AVR 核将 32 个工作寄存器和丰富的指令集联结在一起。所有的工作寄存器都与 ALU（算逻单元）直接相连，允许在一个时钟周期内执行的单条指令同时访问两个独立的寄存器。这种结构提高了代码效率，使 AVR 得到了比普通 CISC 单片机高将近 10 倍的性能。

图 1　AT90S2313 结构方框图



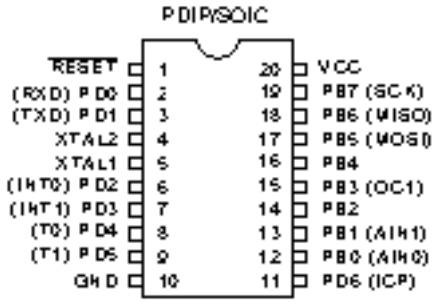
AT90S2313 具有以下特点：2K 字节 FLASH，128 字节 EEPROM，128 字节 SRAM，15 个通用 I/0 口，32 个通用工作寄存器，具有比较模式的灵活的定时器/计数器，内外中断源，可编程的 UART，可编程的看门狗定时器，下载程序用的 SPI 口以及两种可通过软件选择的省电模式。工作于空闲模式时，CPU 将停止运行，而寄存器、定时器/计数器、看门狗和中

断系统继续工作；掉电模式时振荡器停止工作，所有功能都被禁止，而寄存器内容得到保留。只有外部中断或硬件复位才可以退出此状态。

器件是以 ATMEL 的高密度非易失性内存技术生产的。片内 FLASH 可以通过 ISP 接口或通用编程器多次编程。通过将增强的 RISC 8 位 CPU 与 FLASH 集成在一个芯片内，2313 为许多嵌入式控制应用提供了灵活而低成本的方案。

AT90S2313 具有一整套的编程和系统开发工具：宏汇编、调试/仿真器、在线仿真器和评估板。

管脚配置



管脚定义

VCC、GND：电源

B 口（PB7..PB0）：

B 口是一个 8 位双向 I/O 口，每一个管脚都有内部上拉电阻（可单独选择）。PB0 和 PB1 还可作为片内模拟比较器的正（AIN0）负（AIN1）输入端。B 口的输出缓冲器能够吸收 20mA 的电流，可直接驱动 LED。当作为输入时，如果外部被拉低，由于上拉电阻的存在，管脚将输出电流。在复位过程中，B 口为三态，即使此时时钟还未起振。

B 口作为特殊功能口的使用方方法见以后章节。

D 口（PD6..PD0）：

D 口是一个带内部上拉电阻的 7 位双向 I/O 口。输出缓冲器能够吸收 20mA 的电流。当作为输入时，如果外部被拉低，由于上拉电阻的存在，管脚将输出电流。在复位过程中，D 口为三态，即使此时时钟还未起振。

D 口作为特殊功能口的使用方方法见以后章节。

/RESET： 复位输入。超过 50ns 的低电平将引起系统复位。低于 50ns 的脉冲不能保证可靠复位。

XTAL1： 振荡器放大器的输入端。

XTAL2： 振荡器放大器的输出端。

晶体振荡器：

XTAL1 和 XTAL2 分别是片内振荡器的输入、输出端，可使用晶体振荡器或是陶瓷振荡器。当使用外部时钟时，XTAL2 应悬空。

图 2　振荡器连接

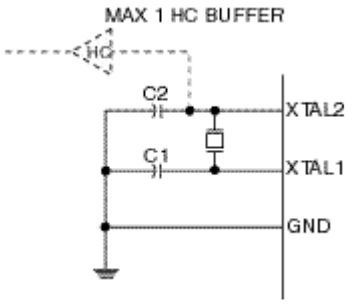
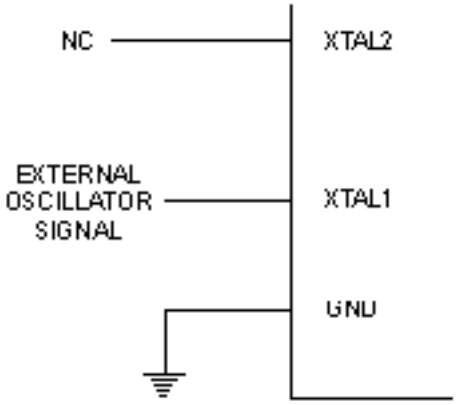
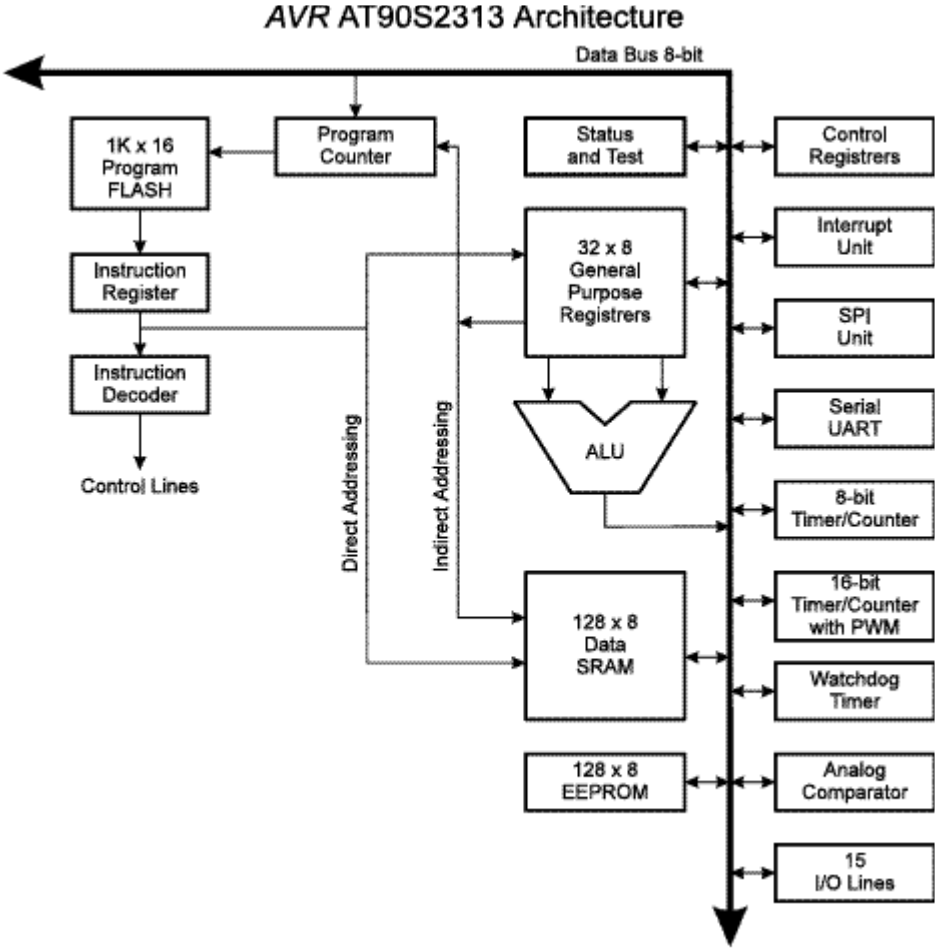


图 3　外部时钟驱动配置



结构纵览

图 4 AT90S2313 AVR RISC **结构**



快速访问寄存器文件包含 32 **个** 8 **位可单周期访问的通用寄存器。这意味着在一个时钟周期内，** ALU **可以完成一次如下操作：读取寄存器文件中的两个操作数，执行操作，将结果存回到寄存器文件。**

寄存器文件中的 6 **个可以组成** 3 **个** 16 **位用于数据寻址的间接寻址寄存器指针，以提高地址运算能力。其中** Z **指针还用于查表功能。**

ALU **支持两个寄存器之间、寄存器和常数之间的算术和逻辑操作，以及单寄存器的操作。除了寄存器操作模式，通常的内存访问模式也适用于寄存器文件。这是因为** AT90S2313 **为寄存器文件分配了** 32 **个最低的数据空间地址（**$00 - $1F**），允许其象普通内存地址一样访问。**

I/O **内存空间包括** 64 **个地址作为** CPU **外设的控制寄存器，** T/C**，以及其他** I/O **功能。**I/O **内存可以直接访问，也可以作为数据地址（**$20 - $5F**）来访问。**

AVR **采用了** HARVARD **结构：程序和数据总线分离。程序内存通过两段式的管道（**Pipeline**）进行访问：当** CPU **在执行一条指令的同时，就去取下一条指令。这种预取指的概念使得指令可以在一个时钟完成。**

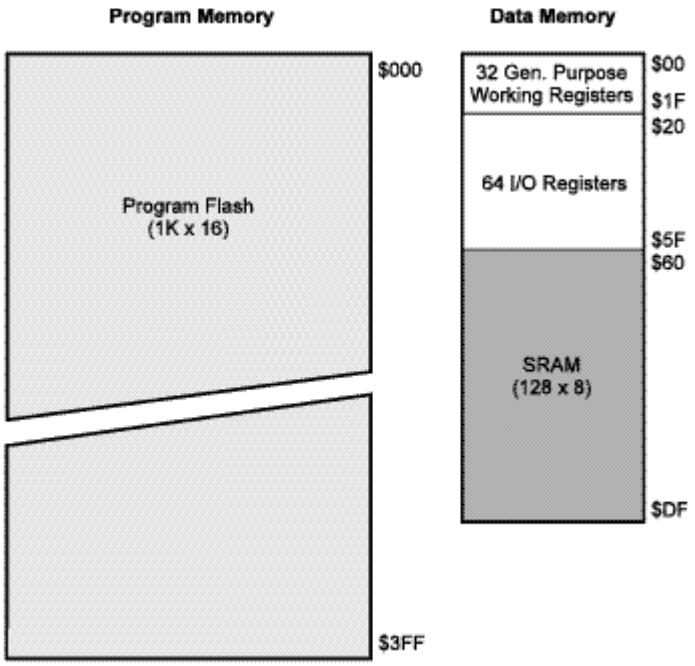
相对跳转和相对调用指令可以直接访问 1K **地址空间。所有的** AVR **指令都为** 16 **位长。每个程序内存地址都包含一条** 16 **位或** 32 **位的指令。**

当执行中断和子程序调用时，返回地址存储于堆栈中。堆栈分布于通用数据 SRAM **之中，堆栈大小只受** SRAM **数量的限制。用户应该在复位例程里就初始化** SP**。**SP **为可读写的** 8

位堆栈指针。

AVR 结构的内存空间是线性的。

图 5　内存映像



中断模块由 I/O 空间中的控制寄存器和状态寄存器中的全局中断使能位组成。每个中断都具有一个中断向量，由中断向量组成的中断向量表位于程序存储区的最前面。中断向量地址低的中断具有高的优先级。

通用工作寄存器文件

图 6　通用工作寄存器

	7 … 0	地址	
通用工作寄存器	R0	$00	
	R1	$01	
	R2	$02	
	…		
	R13	$0D	
	R14	$0E	
	R15	$0F	
	R16	$10	
	R17	$11	
	…		
	R26	$1A	X 寄存器低字节
	R27	$1B	X 寄存器高字节
	R28	$1C	Y 寄存器低字节
	R29	$1D	Y 寄存器高字节
	R30	$1E	Z 寄存器低字节
	R31	$1F	Z 寄存器高字节

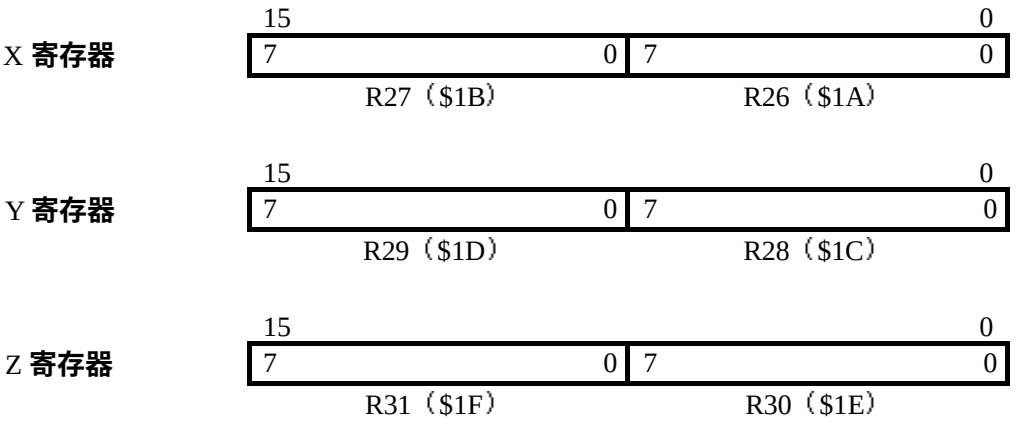
所有的寄存器操作指令都可以单指令的形式直接访问所有的寄存器。例外情况为 5 条涉及常数操作的指令：SBCI、SUBI、CPI、ANDI 和 ORI。这些指令只能访问通用寄存器文件的后半部分：R16 到 R31。

如图 6 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。虽然寄存器文件的实现与 SRAM 不同，这种内存组织方式在访问寄存器方面具有极大的灵活性。

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。

图 7 X、Y、Z 寄存器



ALU

AVR ALU 与 32 个通用工作寄存器直接相连。ALU 操作分为 3 类：算术、逻辑和位操作。

在线可编程 FLASH

AT90S2313 具有 2K 字节的 FLASH。因为所有的指令为 16 位宽，故尔 FLASH 结构为 1K×16。FLASH 的擦除次数至少为 1000 次。

AT90S2313 的程序计数器（PC）为 10 位宽，可以寻址到 1024 个字的 FLASH 程序区。

EEPROM

AT90S2313 包含 128 字节的 EEPROM。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 100000 次（擦除）。EEPROM 的访问由地址寄存器，数据寄存器和控制寄存器决定。

SRAM

图 8 表明了 AT90S2313 的数据组织方式。

图 8 SRAM 分布

寄存器文件		数据地址空间
R0		$00
R1		$01
R2		$02
…		…
R29		$1D
R30		$1E
R31		$1F
I/O 寄存器		数据地址空间

$00		$20
$01		$21
$02		$22
…		…
$3D		$5D
$3E		$5E
$3F		$5F
		片内 SRAM
		$60
		$61
		$62
		…
		$DD
		$DE
		$DF

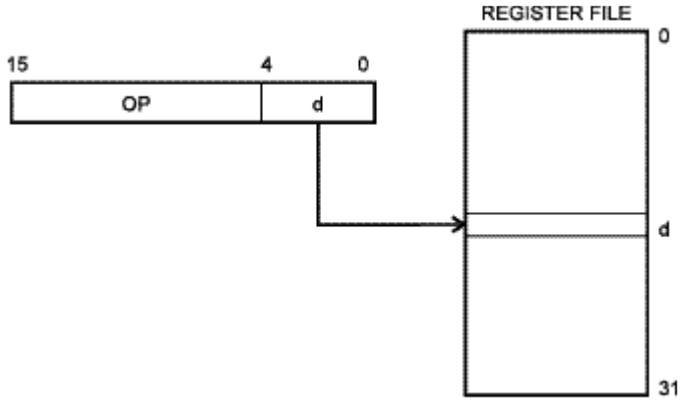
224 个数据地址用于寻址寄存器文件，I/O 和 SRAM。起始的 96 个地址为寄存器文件+I/O，其后的 128 个地址用于寻址 SRAM。
数据寻址模式分为 5 种：直接，带偏移量的间接，间接，预减的间接，后加的间接。寄存器 R26 到 R31 为间接寻址的指针寄存器。
直接寻址范围可达整个数据空间。
带偏移量的间接寻址模式寻址到 Y、Z 指针给定地址附近的 63 个地址。
带预减和后加的间接寻址模式要用到 X、Y、Z 指针。
32 个通用寄存器，64 个 I/O 寄存器和 128 字节的 SRAM 可以被所有的寻址模式访问。

程序和数据寻址模式

AT90S2313 支持强大而有效的寻址模式。本节将要介绍各种寻址模式。

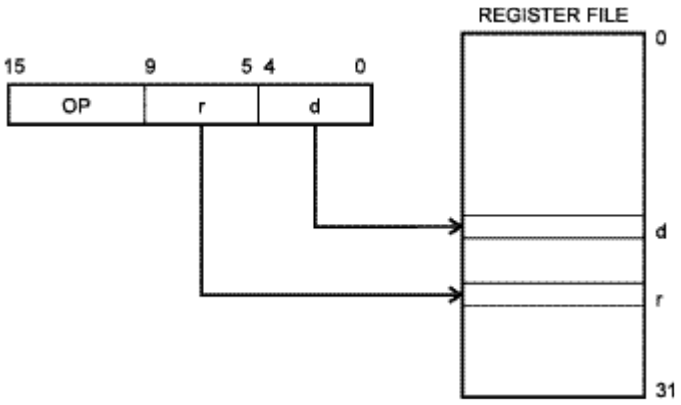
单寄存器直接寻址

图 9



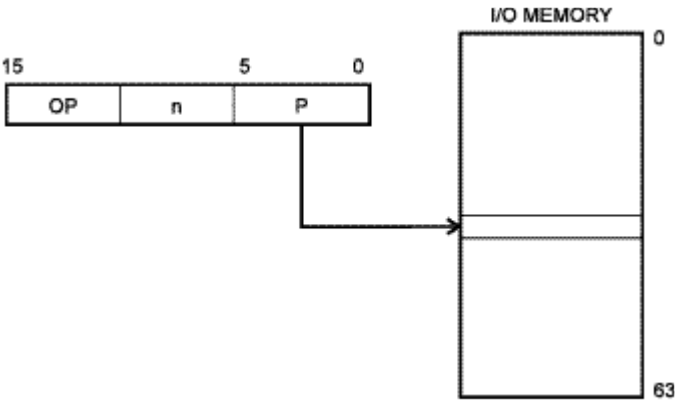
双寄存器直接寻址

图 10



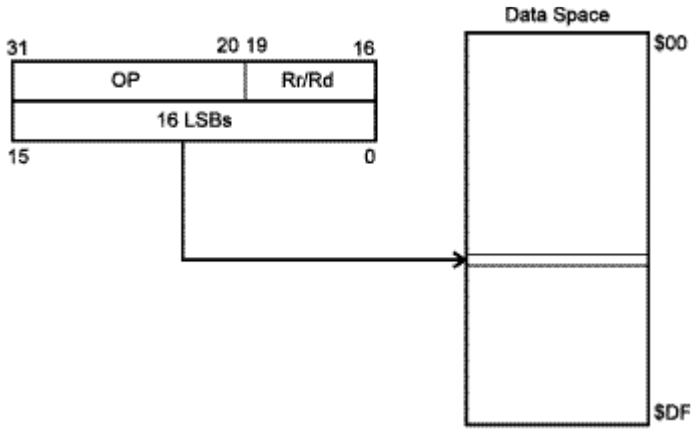
I/O 直接寻址

图 11



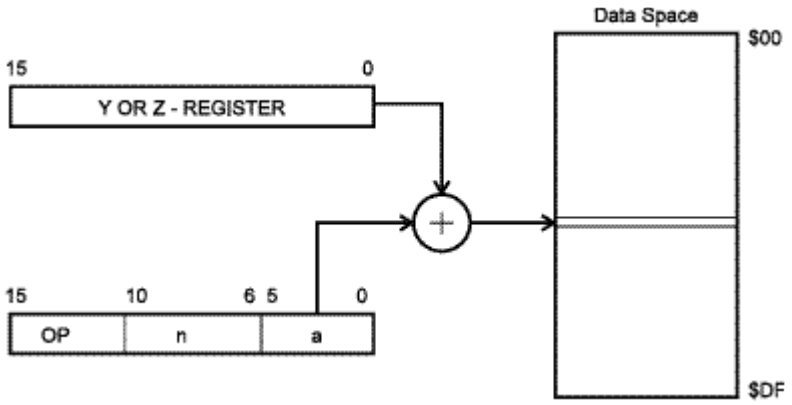
数据直接寻址

图 12



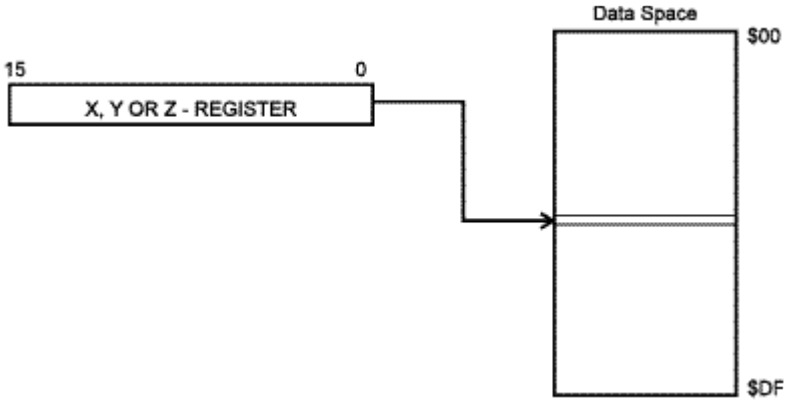
带偏移的数据间接寻址

图 13



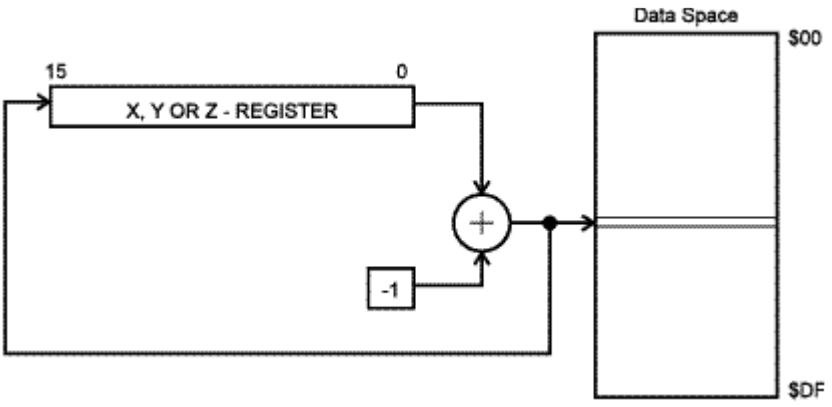
数据间接寻址：

图 14



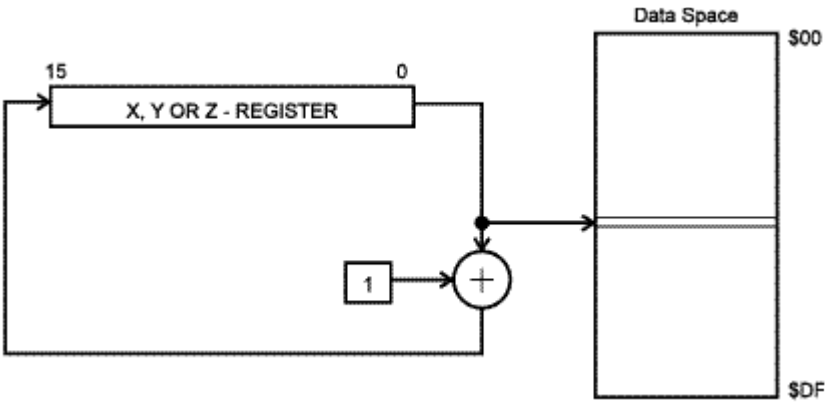
带预减的数据间接寻址

图 15



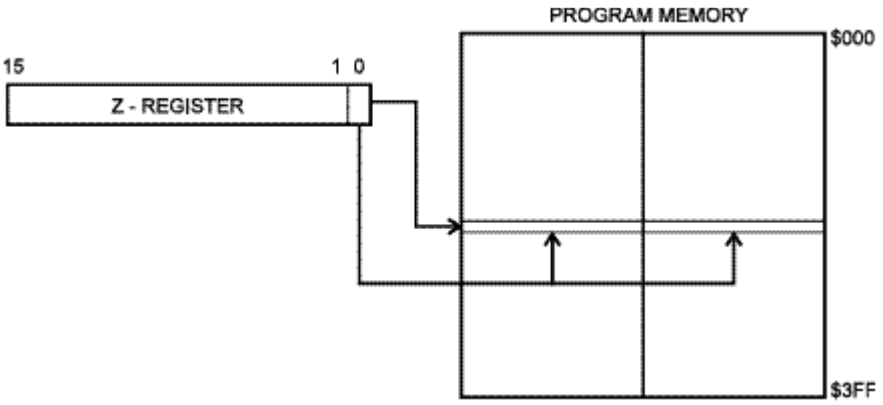
带后加的数据间接寻址

图 16



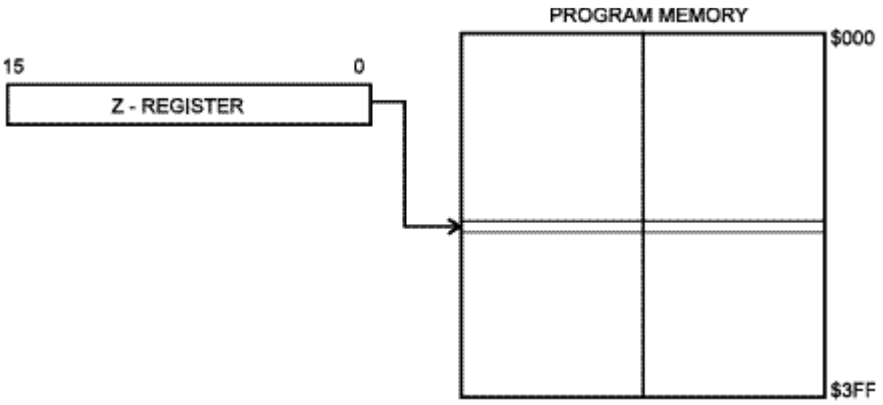
使用 LPM指令寻址常数

图 17



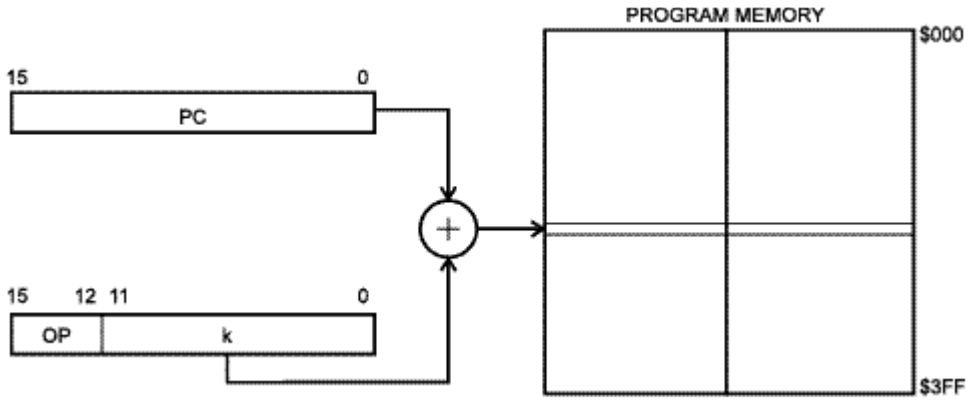
间接程序寻址，IJMP 和 ICALL

图 18



相对程序寻址，RJMP 和 RCALL

图 19



内存访问和指令执行时序

这一节介绍指令执行和内存访问时序。

AVR CPU **由系统时钟Φ驱动。此时钟由外部晶体直接产生。**

图 20 **说明了由** HARVARD **结构决定了的并行取指和执行，以及快速访问寄存器文件的概念。这是一个基本的，达到** 1MIPS/MHz**，优良的性价比，功能/时钟比，功能/功耗比的流水线概念。**

图 20 并行取指与指令执行

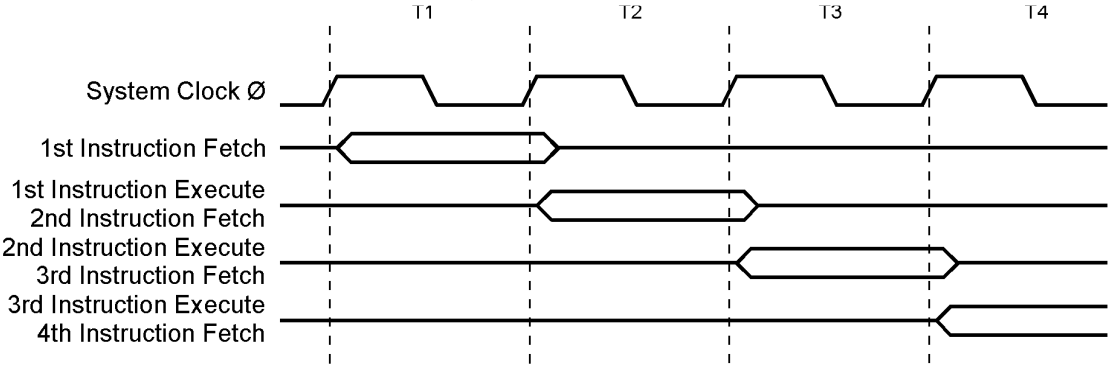


图 21 **演示的是寄存器文件内部时序。在一个时钟周期里，**ALU **可以同时对两个寄存器操作数进行操作，同时将结果存回到其中的一个寄存器中去。**

图 21 单时钟 ALU **操作**

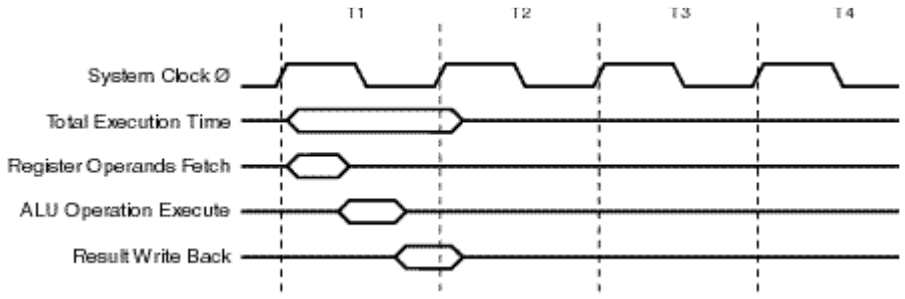
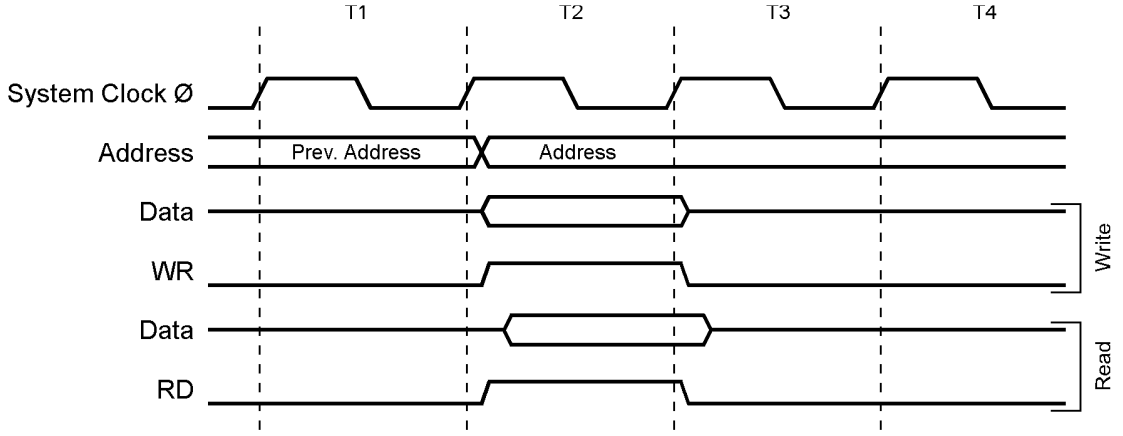


图 22 片内 SRAM 访问周期



I/O 内存

表 1 AT90S2313 的 I/O 空间

地址（16 进制）	名称	功能
$3F($5F)	SREG	状态寄存器
$3D($5D)	SPL	堆栈指针低字节
$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 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
$33($53)	TCCR0	T/C0 控制寄存器
$32($52)	TCNT0	T/C0（8 位）
$2F($4F)	TCCR1A	T/C1 控制寄存器 A
$2E($4E)	TCCR1B	T/C1 控制寄存器 B
$2D($4D)	TCNT1H	T/C1 高字节
$2C($4C)	TCNT1L	T/C1 低字节
$2B($4B)	OCR1AH	输出比较寄存器高字节
$2A($4A)	OCR1AL	输出比较寄存器低字节
$25($45)	ICR1H	T/C1 输入捕捉寄存器高字节
$24($44)	ICR1L	T/C1 输入捕捉寄存器低字节
$21($41)	WDTCR	看门狗控制寄存器
$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
$18($38)	PORTB	B 口数据寄存器
$17($37)	DDRB	B 口数据方向寄存器
$16($36)	PINB	B 口输入引脚
$12($32)	PORTD	D 口数据寄存器
$11($31)	DDRD	D 口数据方向寄存器
$10($30)	PIND	D 口输入引脚
$0C($2C)	UDR	UART 数据寄存器
$0B($2B)	USR	UART 状态寄存器
$0A($2A)	UCR	UART 控制寄存器

$09($29)	UBRR	UART 波特率寄存器
$08($28)	ACSR	模拟比较器控制及状态寄存器

AVR2313 的所有 I/O 和外围都被放置在 I/O 空间。IN 和 OUT 指令用来访问不同的 I/O 地址，以及在 32 个通用寄存器之间传输数据。地址为$00-$1F 的 I/O 寄存器还可用 SBI 和 CBI 指令进行位寻址，而 SIBC 和 SIBS 则用来检查单个位置位与否。当使用 IN 和 OUT 指令时地址必须在$00-$3F 之间。如果要象 SRAM 一样访问 I/O 寄存器，则相应地址要加上$20。在本文档里所有 I/O 寄存器的 SRAM 地址写在括号中。
为了与后续产品兼容，保留未用的未应写“0”，而保留的 I/O 寄存器则不应写。
一些状态标志位的清除是通过写“1”来实现的。CBI 和 SBI 指令读取已置位的标志位时，会回写“1”，因此会清除这些标志位。CBI 和 SBI 指令只对$00-$1F 有效。
I/O 寄存器和外围控制寄存器在后续章节介绍。

状态寄存器 SREG（Status Register）

BIT	7	6	5	4	3	2	1	0
$3F（$5F）	**I**	**T**	**H**	**S**	**V**	**N**	**Z**	**C**
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

I：全局中断使能
置位时使能全局中断。单独的中断使能由个独立控制寄存器控制。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。I 在复位时清零，RETI 指令执行后置位。
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。
H：半加标志位
S：符号位
总是 N 与 V 的异或。
V：二进制补码溢出标志位
N：负数标志位
Z：零标志位
C：进位标志位
状态寄存器在进入中断和退出中断时并不自动进行存储和恢复。这项工作由软件完成。

堆栈指针 SP

BIT	7	6	5	4	3	2	1	0
$3D（$5D）	**SP7**	**SP6**	**SP5**	**SP4**	**SP3**	**SP2**	**SP1**	**SP0**
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

堆栈指针指向位于 SRAM 的函数及中断堆栈。堆栈空间必须在调用函数或中断使能之前定义。指针必须指向高于$60 的地址。用 PUSH 指令推数据入栈时，堆栈指针将减一，而当调用函数或中断时，指针将减二。使用 POP 指令时，堆栈指针将加一，而用 RET 或 RETI 返回时，指针将加二。

复位和中断处理

AT90S2313 有 10 个中断源。每个中断源在程序空间都有一个独立的中断向量。所有的中断

事件都有自己的使能位。当使能位置位，且 I 也置位的情况下，中断可以发生。
器件复位后，程序空间的最低位置自动定义为复位及中断向量。完整的中断表见图 2。在中断向量表中处于低地址的中断具有高的优先级。所以，RESET 具有最高的优先级。

表 2　复位与中断向量

向量号	程序地址	来源	定义
1	$000	RESET	硬件管脚，上电复位和看门狗复位
2	$001	INT0	外部中断 0
3	$002	INT1	外部中断 1
4	$003	TIMER1 CAPT1	T/C1 捕捉事件
5	$003	TIMER1 COMP1	T/C1 比较匹配
6	$004	TIMER1 OVF1	T/C1 溢出
7	$005	TIMER0 OVF0	T/C0 溢出
8	$006	UART，RX	UART 接收结束
9	$007	UART，UDRE	UART 数据寄存器空
10	$008	UART，TX	UART 发送结束
11	$00A	ANA_COMP	模拟比较器

设置中断向量地址最典型的方法如下：

```
地址    标号     代码                              注释
$000             RJMP    RESET                   ; 复位
$001             RJMP    EXT_INT0                ; IRQ0
$002             RJMP    EXT_INT1                ; IRQ1
$003             RJMP    TIM_CAPT1               ; T1 捕捉
$004             RJMP    TIM_COMP1               ; T1 比较匹配
$005             RJMP    TIM_OVF1                ; T1 溢出
$006             RJMP    TIM_OVF0                ; T0 溢出
$007             RJMP    UART_RXC                ; UART 接收结束
$008             RJMP    UART_DRE                ; UART 数据空
$009             RJMP    UART_TXC                ; UART 发送结束
$00a             RJMP    ANA_COMP                ; 模拟比较器
;
$00b    MAIN:    LDI     R16, LOW(REMEND)        ; 主程序开始
$00c             OUT     SPL, R16
$00d             <指令>  XXX
—       —        —       —
```

复位源

AT90S2313 有 3 个复位源：

- 上电复位。当电源电压低于上电门限 V_{POT} 时 MCU 复位。
- 外部复位。当/RESET 引脚上的低电平超过 50ns 时 MCU 复位。
- 看门狗复位。看门狗定时器超时后 MCU 复位。

在复位期间，所有的 I/O 寄存器被设置为初始值，程序从地址$000 开始执行。$000 地址中放置的指令必须为 RJMP—相对跳转指令—跳转到复位处理例程。若程序永远不需中断，则中断向量就可放置通常的程序代码。图 23 为复位电路的逻辑图。表 3 定义了复位电路的时序和电参数。

图 23　复位逻辑

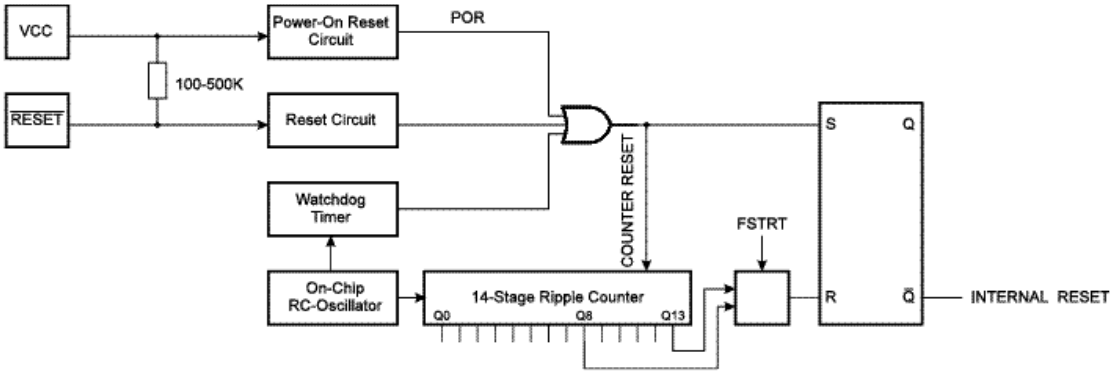


表 3　复位电参数（V_{CC} = 5.0V）

符号	参数	最小值	典型值	最大值	单位
V_{POT}(1)	上电复位电压门限（上升）	1.0	1.4	1.8	V
	上电复位电压门限（下降）	0.4	0.6	0.8	V
V_{RST}	复位引脚门限电压	-	-	0.85V_{CC}	V
t_{TOUT}	复位延迟周期，FSTRT 未编程	11	16	21	ms
t_{TOUT}	复位延迟周期，FSTRT 已编程	1.0	1.1	1.2	ms

注：1.除非电源电压低于 V_{POT}，否则上电复位不会发生。

用户可以按照典型振荡器起振特性来选择启动时间。用于时间溢出的 WDT 振荡周期数示于表 4。看门狗振荡器的频率与工作电压有关，具体参见后续章节的典型特性。

表 4 看门狗振荡器周期数

FSTRT	溢出时间（V_{CC}=5V）	WDT 周期数
编程	1.1ms	1K
未编程	16.0ms	16K

上电复位

上电复位（POR）保证器件在上电时正确复位。如图 23 所示，看门狗定时器驱动一个内部定时器，此定时器保证 MCU 只有在 V_{CC} 达到 V_{POT} 且过了一定时间之后才启动。位于 FLASH 内的 FSTRT 熔丝位编程后可以使 MCU 以较短的时间起动，如果用户使用了陶瓷振荡器或是其他快速起动的振荡器。

如果内置于片内的启动时间足够的话，/RESET 可以与 V_{CC} 直接相连，或是外接上拉电阻。如果在加上 V_{CC} 的同时保持/RESET 为低，则可以延长复位周期。例子可参看图 25。

图 24　MCU 起动，/RESET 与 V_{CC} 相连

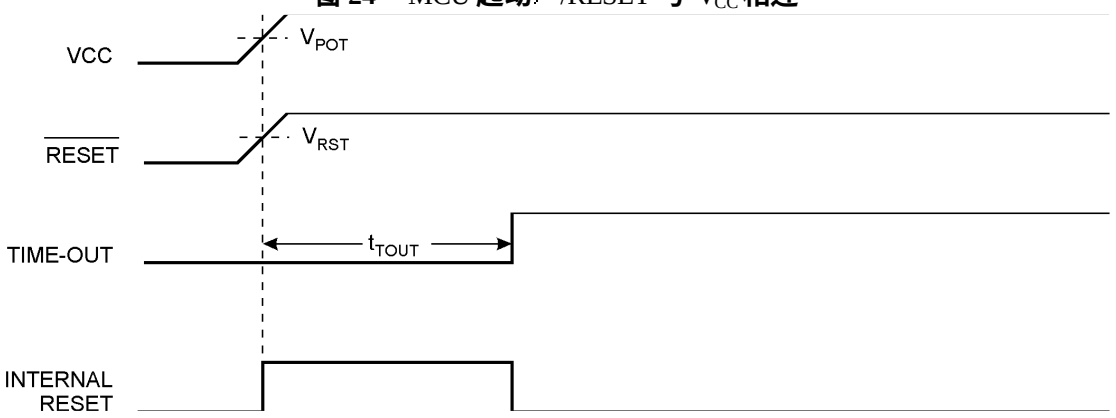
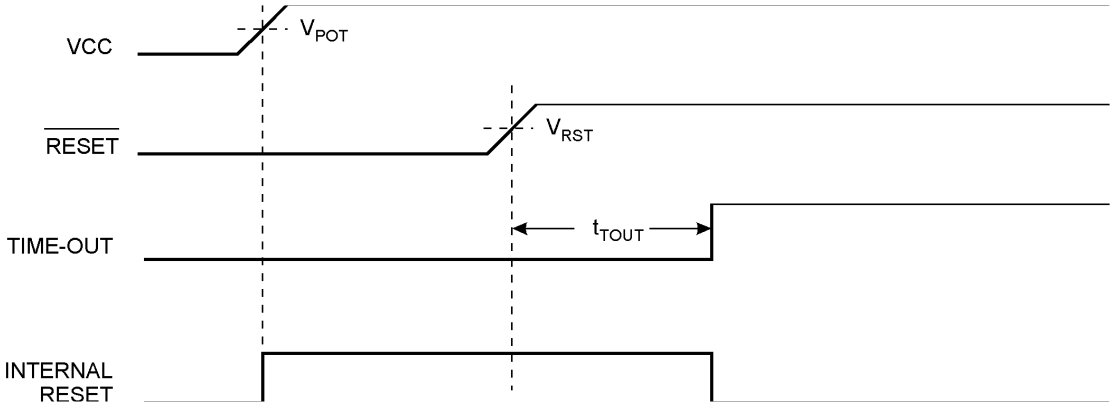


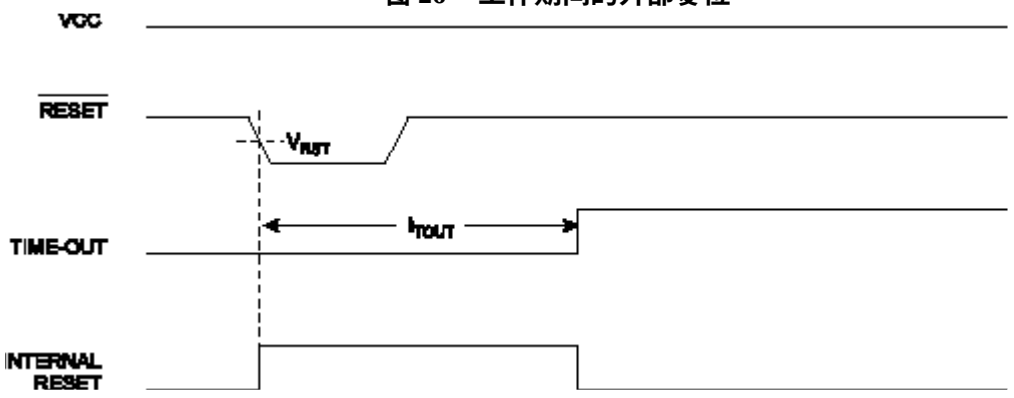
图 25　MCU 启动，/RESET 由外电路控制



外部复位

外部复位由外加于/RESET 引脚的低电平产生。大于 50ns 的复位脉冲将造成芯片复位。施加短脉冲不能保证可靠复位。当外加信号达到复位门限电压 V_{RST}（上升沿）时，t_{TOUT} 延时周期开始。然后，MCU 启动。

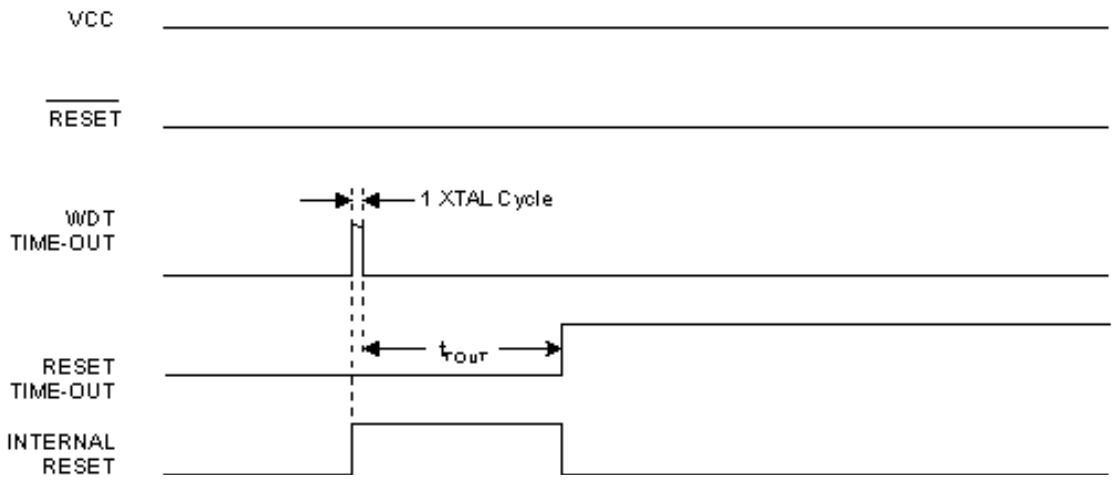
图 26　工作期间的外部复位



看门狗复位

当看门狗定时器溢出时，将产生 1 个 XTAL 周期的复位脉冲。在脉冲的下降沿，延时定时器开始对 t_{TOUT} 记数。

图 27　工作期间的看门狗复位



中断处理

AT90S2313 有 2 个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。

一个中断产生后，全局中断使能位 I 将被清零，后续中断被屏蔽。用户可以在中断例程里对 I 置位，从而开放中断。执行 RETI 后 I 重新置位。

对于那些由可以保持为静态的事件（如输出比较寄存器 1 与 T/C1 值相匹配）驱动的中断，事件发生后中断标志将置位。如果中断标志被清除而中断条件仍然存在，则标志只有在新事件发生后才会置位。

当程序计数器指向实际中断向量开始执行相应的中断例程时，硬件清除对应的中断标志。一些中断标志位也可以通过软件写“1”来清除。

当一个符合条件的中断发生后，如果相应的中断使能位为“0”，则中断标志位挂起，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，则所有的中断都不会被执行，直到 I 置位。然后被挂起的各个中断按中断优先级依次中断。

注意：外部电平中断没有中断标志位，因此当电平变为非中断电平后，中断条件即终止。

进入中断和退出中断时 MCU 不会自动保存或恢复状态寄存器，故尔需由软件处理。

通用中断屏蔽寄存器—GIMSK

BIT	7	6	5	4	3	2	1	0
$3B($5B)	**INT1**	**INT0**	-	-	-	-	-	-
读/写	R/W	R/W	R	R	R	R	R	R
初始值	0	0	0	0	0	0	0	0

位 5..0：保留

INT1：外部中断 1 请求使能

当 INT0 和 I 都为“1”时，外部引脚中断使能。MCU 通用控制寄存器（MCUCR）中的中断检测控制位 1/0（ISC11 和 ISC10）定义中断 1 是上升沿中断还是下降沿中断，或者是低电平中断。即使管脚被定义为输出，中断仍可产生。

INT0：外部中断 0 请求使能

当 INT0 和 I 都为“1”时，外部引脚中断使能。MCU 通用控制寄存器（MCUCR）中的中断检测控制位 1/0（ISC01 和 ISC00）定义中断 0 是上升沿中断还是下降沿中断，或者是低电平中断。即使管脚被定义为输出，中断仍可产生。

通用中断标志寄存器—GIFR

BIT	7	6	5	4	3	2	1	0
$3A($5A)	**INTF1**	**INTF0**	-	-	-	-	-	-
读/写	R/W	R/W	R	R	R	R	R	R
初始值	0	0	0	0	0	0	0	0

位 5..0：保留

INTF1：外部中断标志 1

当 INT1 管脚有事件触发中断请求时，INTF1 置位（“1”）。如果 SREG 中的 I 及 GIMSK 中的 INT1 都为“1”，则 MCU 将跳转到中断地址$002。中断例程执行后，此标志被清除。另外，标志也可以通过对其写“1”来清除。

INTF0：外部中断标志 0

当 INT0 管脚有事件触发中断请求时，INTF0 置位（“1”）。如果 SREG 中的 I 及 GIMSK 中

的 INT0 都为“1”，则 MCU 将跳转到中断地址$001。中断例程执行后，此标志被清除。另外，标志也可以通过对其写“1”来清除。

T/C 中断屏蔽寄存器—TIMSK

BIT	7	6	5	4	3	2	1	0
$39($59)	TOIE1	OCIE1A	-	-	TICIE1	-	TOIE0	-
读/写	R/W	R/W	R	R	R/W	R	R/W	R
初始值	0	0	0	0	0	0	0	0

位 5、4、2、0：保留

TOIE1：T/C1 溢出中断使能

当 TOIE1 和 I 都为“1”时，T/C1 溢出中断使能。当 T/C1 溢出，或 TIFR 中的 TOV1 位置位时，中断例程（$005）得到执行。

OCIE1A：T/C1 输出比较匹配中断使能

当 TOIE1 和 I 都为“1”时，输出比较匹配中断使能。当 T/C1 的比较匹配发生，或 TIFR 中的 OCIE1A 置位，中断例程（$004）将执行。

TICIE1：T/C1 输入捕捉中断使能

当 TICIE1 和 I 都为“1”时，输入捕捉中断使能。当 T/C1 的输入捕捉事件发生（PD6），或 TIFR 中的 ICF1 置位，中断例程（$003）将执行。

TOIE0：T/C0 溢出中断使能

当 TOIE0 和 I 都为“1”时，T/C0 溢出中断使能。当 T/C0 溢出，或 TIFR 中的 TOV0 位置位时，中断例程（$006）得到执行。

T/C 中断标志寄存器—TIFR

BIT	7	6	5	4	3	2	1	0
$38($58)	TOV1	OCF1A	-	-	ICF1	-	TOV0	-
读/写	R/W	R/W	R	R	R/W	R	R/W	R
初始值	0	0	0	0	0	0	0	0

位 5、4、2、0：保留

TOV1：T/C1 溢出中断标志位

当 T/C1 溢出时，TOV1 置位。执行相应的中断例程后此位硬件清零。此外，TOV1 也可以通过写“1”来清零。当 SREG 中的位 I、TOIE1 和 TOV1 一同置位时，中断例程得到执行。在 PWM 模式中，当 T/C1 在$0000 改变记数方向时，TOV1 置位。

OCF1A：输出比较标志 1A

当 T/C1 与 OCR1A 的值匹配时，OCF1A 置位。此位在中断例程里硬件清零，或者通过对其写“1”来清零。当 SREG 中的位 I、OCIE1A 和 OCF1A 一同置位时，中断例程得到执行。

ICF1：输入捕捉标志位

当输入捕捉事件发生时，ICF1 置位，表明 T/C1 的值已经送到输入捕捉寄存器 ICR1。此位在中断例程里硬件清零，或者通过对其写“1”来清零。当 SREG 中的位 I、TICIE1A 和 ICF1 一同置位时，中断例程得到执行。

TOV0：T/C0 溢出中断标志位

当 T/C0 溢出时，TOV0 置位。执行相应的中断例程后此位硬件清零。此外，TOV0 也可以通过写“1”来清零。当 SREG 中的位 I、TOIE0 和 TOV0 一同置位时，中断例程得到执行。

外部中断

外部中断由 INT0 和 INT1 引脚触发。应当注意，如果中断使能，则即使 INT0/INT1 配置为

输出，中断照样会被触发。此特点提供了一个产生软件中断的方法。触发方式可以为上升沿，下降沿或低电平。这些设置由 MCU 控制寄存器 MCUCR 决定。当设置为低电平触发时，只要电平为低，中断就一直触发。

中断响应时间

AVR 中断响应时间最少为 4 个时钟周期。在这 4 个时钟期间，PC（2 个字节）自动入栈，而 SP 减 2。在通常情况下，中断向量为一个相对跳转指令，此跳转要花 2 个时钟周期。如果中断在一个多周期指令执行期间发生，则在此多周期指令执行完后 MCU 才会执行中断程序。

中断返回亦需 4 个时钟。在此期间，PC 将被弹出栈，SREG 的位 I 被置位。如果在中断期间发生了其他中断，则 AVR 在退出中断程序后，要执行一条主程序指令之后才能再响应被挂起的中断。

MCU控制寄存器—MCUCR

BIT	7	6	5	4	3	2	1	0
$35($55)	-	-	**SE**	**SM**	**ISC11**	**ISC10**	**ISC01**	**ISC00-**
读/写	R	R	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位 7、6：保留

SE：休眠使能

执行 SLEEP 指令时，SE 必须置位才能使 MCU 进入休眠模式。为了防止无意间使 MCU 进入休眠，建议与 SLEEP 指令相连使用。

SM：休眠模式

此位用于选择休眠模式。SM 为“0”时为闲置模式；SM 为“1”时为掉电模式。

ISC11，ISC10：中断检测控制 1 位 1 和位 0

选择 INT1 中断的边沿或电平，如下表所示：

表 5　中断 1 检测控制

ISC11	ISC10	描述
0	0	低电平中断
0	1	保留
1	0	下降沿中断
1	1	上升沿中断

注：改变 ISC11/ISC10 时，首先要禁止 INT1（清除 GIMSK 的 INT1 位），否则可能引发不必要的中断。

ISC01，ISC00：中断检测控制 0 位 1 和位 0

选择 INT0 中断的边沿或电平，如下表所示：

表 6　中断 0 检测控制

ISC01	ISC00	描述
0	0	低电平中断
0	1	保留
1	0	下降沿中断
1	1	上升沿中断

注：改变 ISC01/ISC00 时，首先要禁止 INT0（清除 GIMSK 的 INT0 位），否则可能引发不必要的中断。

INTn 引脚的电平在检测边沿之前采样。如果边沿中断使能，则大于一个 MCU 时钟的脉冲将触发中断。如果选择了低电平触发，则此电平必须保持到当前执行的指令结束。

休眠模式

进入休眠模式的条件是 SE 为“1”，然后执行 SLEEP 指令。使能的中断将唤醒 MCU。完成中断例程后，MCU 执行 SLEEP 以后的指令。在休眠期间，寄存器文件及 I/O 内存的内容不会丢失。如果在休眠模式下复位，则 MCU 从 RESET 向量（$000）处开始运行。

闲置模式：

当 SM 为“0”时，SLEEP 指令将使 MCU 进入闲置模式。在此模式下，CPU 停止运行，而定时器/计数器、看门狗和中断系统继续工作。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 ACSR 的 ACD。如果 MCU 从闲置模式唤醒，CPU 将立即执行指令。

掉电模式：

当 SM 为“1”时，SLEEP 指令将使 MCU 进入掉电模式。在此模式下，外部晶振停振，而外部中断及看门狗（在使能的前提下）继续工作。只有外部复位、看门狗复位和外部电平中断（INT0 和 INT1）可以使 MCU 脱离掉电模式。

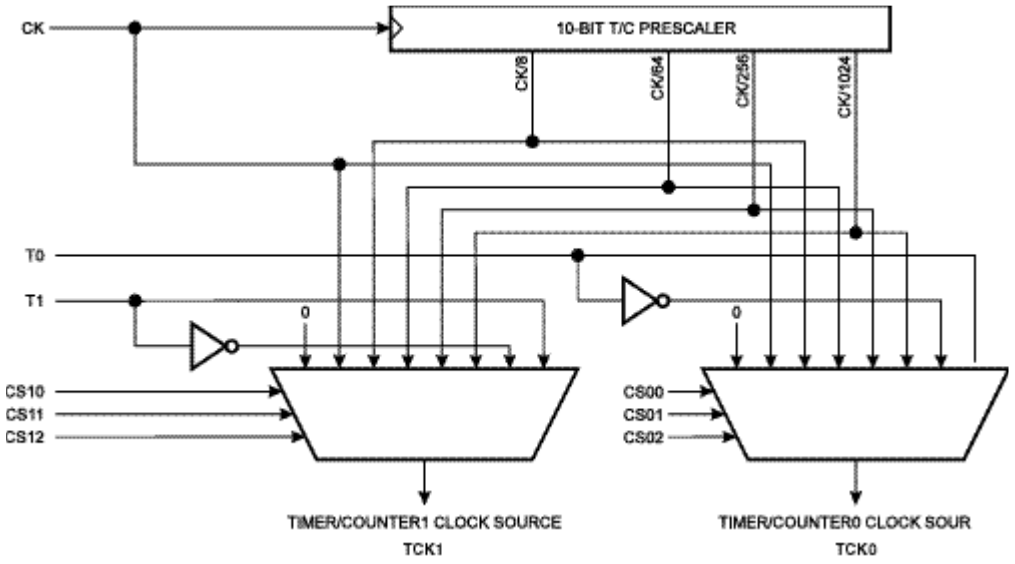
使用外部电平中断唤醒 MCU 时要注意保持低电平大于 T_{TOUT} 的时间，否则 MCU 继续保持掉电模式。

定时器/计数器

AT90S2313 内部有两个通用定时器/计数器：一个 8 位 T/C 和一个 16 位 T/C。T/C 从同一个 10 位的预分频定时器取得预分频的时钟。T/C 既可用作使用片内时钟的定时器，也可用作对外部触发信号记数的计数器。

T/C 的预分频器

图 28　T/C 的预分频器

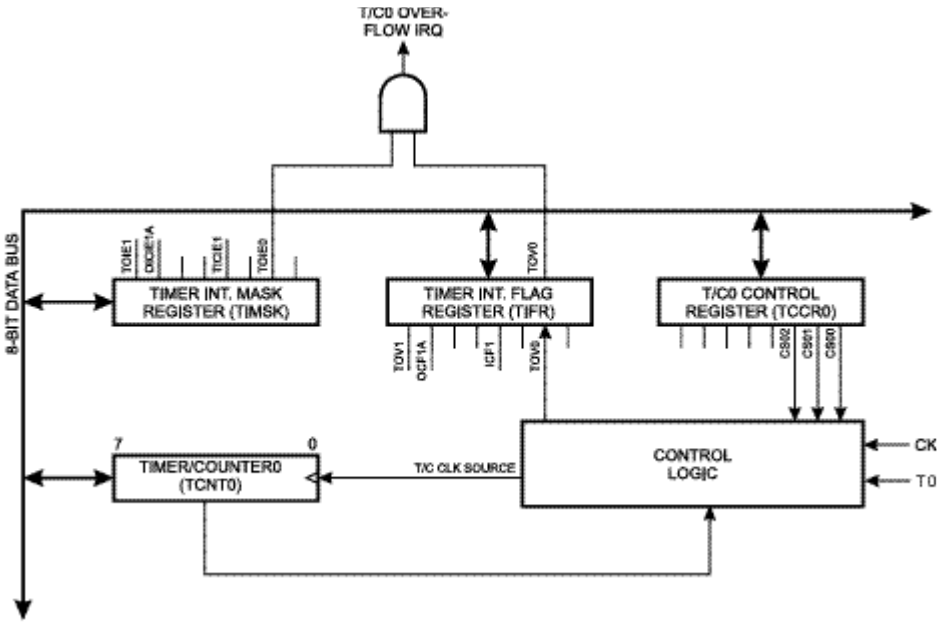


4 种可选的预分频时钟为：CK/8、CK/64、CK/256 和 CK/1024。CK 为振荡器时钟。还可以选择 CK、外部时钟，以及停止工作。

8 位 T/C0

图 29 为 T/C0 的框图。

图 29　T/C0 工作框图



T/C0 **的时钟可以选择** CK、**预分频的** CK **或外部引脚输入。另外还可以由** T/C0 **控制寄存器** TCCR0 **来停止它。**TIFR **为状态标志寄存器，** TCCR0 **为控制寄存器，而** TIMSK **控制** T/C0 **的中断屏蔽。**

当 T/C0 **由外部时钟信号驱动时，为了保证** CPU **对信号的正确采样，要保证外部信号的转换时间至少为一个** CPU **时钟周期。**MCU **在内部** CPU **时钟的上升沿对外部信号进行采样。**

在低预分频条件下，T/C0 **具有高分辨率和高精度的特点；而在高预分频条件下，**T/C0 **非常适用于低速功能，如计时。**

T/C0 控制寄存器—TCCR0

BIT	7	6	5	4	3	2	1	0
$33($53)	-	-	-	-	-	**CS02**	**CS01**	**CS00**
读/写	R	R	R	R	R	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位 7..3：**保留**

CS02、CS01、CS00：时钟选择

表 7 T/C0 **预分频选择**

CS02	CS01	CS00	**描述**
0	0	0	**停止**
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	**外部引脚** T0，**下降沿**
1	1	1	**外部引脚** T0，**上升沿**

当 T/C0 **由外部引脚** T0 **驱动时，即使** PD4（T0）**配置为输出，管脚上的信号变化照样可以使计数器发生相应的变化。这就为用户提供了一个软件控制的方法。**

T/C0—TCNT0

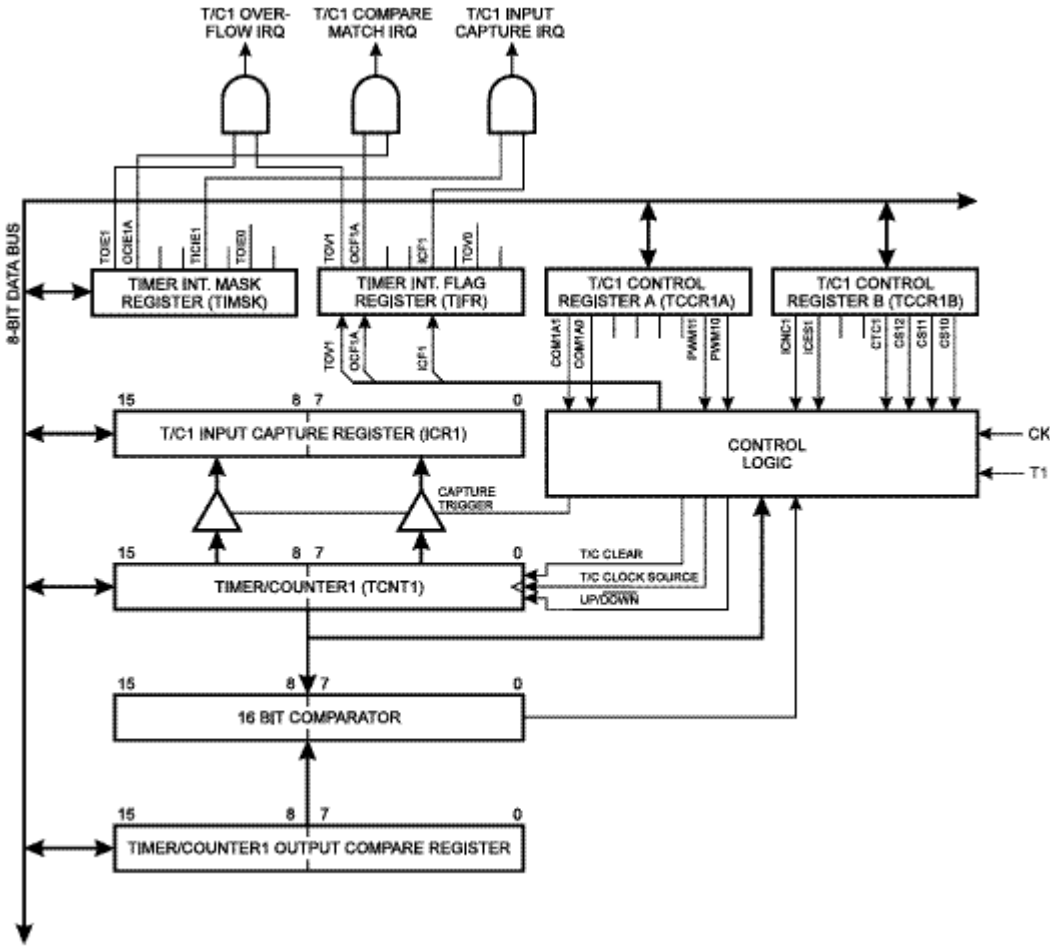
BIT	7	6	5	4	3	2	1	0
$32($52)	**MSB**							**LSB**
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

T/C0 **是可以进行读/写访问的向上计数器。只要有时钟输入，**T/C0 **就会在写入的值基础上向上记数。**

16 位 T/C1

图 30 **为** T/C1 **的框图。**

图 30 T/C1 **工作框图**



T/C1 **的时钟可以选择** CK、**预分频的** CK **或外部引脚输入。另外还可以由** T/C1 **控制寄存器** TCCR1B **来停止它。**TIFR **为状态标志寄存器，而** TIMSK **控制** T/C1 **的中断屏蔽。**

当 T/C0 **由外部时钟信号驱动时，为了保证** CPU **对信号的正确采样，要保证外部信号的转换时间至少为一个** CPU **时钟周期。**MCU **在内部** CPU **时钟的上升沿对外部信号进行采样。**

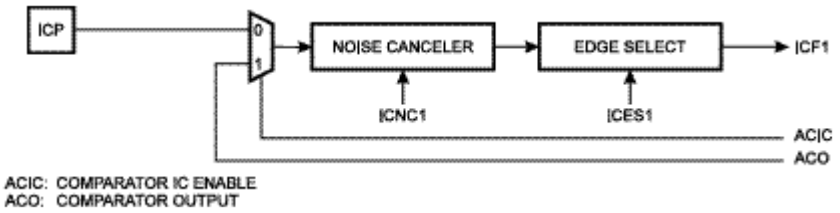
在低预分频条件下，T/C1 **具有高分辨率和高精度的特点；而在高预分频条件下，**T/C1 **非常适用于低速功能，如计时。**

利用输出比较寄存器 OCR1A **作为数据源，**T/C1 **还可以实现输出比较的功能。此功能包括比较匹配时清除计数器和输出比较管脚** 1 **的动作（几种动作可任选一种）。**

T/C1 **还可以用作** 8，9 **或** 10 **位的** PWM **调制器。在此模式下，计数器和** OCR1 **寄存器用作无尖峰干扰的中心对称的** PWM。

当输入捕捉引脚 ICP 发生相应事件时，T/C1 的值将被传到输入捕捉寄存器 ICR1。捕捉事件的设置由 TCCR1B 控制。此外，模拟比较器也可以设置为触发输入捕捉。ICP 引脚逻辑见图 31。

图 31　　ICP 引脚原理图



如果噪声抑制功能使能，则触发信号要进行 4 次采样。只有当 4 个采样值都相等时，才会触发捕捉标志。

T/C1 控制寄存器 A—TCCR1A

BIT	7	6	5	4	3	2	1	0
$2F($4F)	**COM1A1**	**COM1A0**	-	-	-	-	**PWM11**	**PWM10**
读/写	R/W	R/W	R	R	R	R	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位 5.2：保留

COM1A1，COM1A0：比较输出模式，位 1 和 0

COM1A1 和 COM1A0 决定 T/C1 的比较匹配发生时输出引脚的动作。这些动作都将影响管脚 OC1-输出比较引脚 1（PB3）。这是 I/O 口的第二功能，相应的方向控制位要设置为“1”以便将其配置为输出。具体配置见表 8。

表 8　比较 1 模式选择

COM1A1	COM1A0	描述
0	0	T/C1 与输出引脚 OC1 断开
0	1	OC1 输出变换
1	0	清除 OC1
1	1	置位 OC1

PWM11，PWM10：PWM 选择

表 9　PWM 模式选择

PWM11	PWM10	描述
0	0	T/C1 的 PWM 操作无效
0	1	T/C1 为 8 位 PWM
1	0	T/C1 为 9 位 PWM
1	1	T/C1 为 10 位 PWM

T/C1 控制寄存器 B—TCCR1B

BIT	7	6	5	4	3	2	1	0
$2E($4E)	**ICNC1**	**ICES1**	-	-	**CTC1**	**CTC2**	**CS11**	**CS10**
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

位 5、4：保留

ICNC1：输入捕捉抑制器（4 个时钟）

ICNC1 高有效。输入捕捉在 ICP-输入捕捉引脚的第一个上升/下降边沿触发。当 ICNC1 为“1”时，ICP 信号要进行 4 次连续采样，只有 4 个采样值都有效时输入捕捉标志才置位。采样频

率为 XTAL 时钟。

ICES1：输入捕捉 1 边沿选择

当 ICES1 位为“0”时，T/C1 的值在 ICP 引脚电平的下降沿被传送到输入捕捉寄存器 ICR1。若 ICES1 位为“1”，则 T/C1 的值在 ICP 引脚电平的上升沿被传送到 ICR1。

CTC1：比较匹配时清除 T/C1

CTC1 为“1”时，比较 A 匹配事件发生后，T/C1 将复位为 0。若 CTC1 为“0”，则 T/C1 将连续记数而不受比较匹配的影响。由于比较匹配事件的检测发生在匹配发生之后的一个 CPU 时钟，故而定时器的预分频比率的不同将引起此功能有不同的表现。当预分频为 1，A 比较匹配寄存器的值设置为 C 时，定时器的记数方式为：

.|.C-2 | C-1 | C | 0 | 1 |…

而当预分频为 8 时，定时器的记数方式则为：

..| C-2, C-2, C-2, C-2, C-2, C-2, C-2, C-2 | C-1, C-1, C-1, C-1, C-1, C-1, C-1, C-1 | C, 0, 0, 0, 0, 0, 0, 0 |…

在 PWM 模式下，这几位没有作用。

CS12、CS11、CS10：时钟选择

表 10 T/C1 预分频选择

CS12	CS11	CS10	描述
0	0	0	停止
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	外部引脚 T1，下降沿
1	1	1	外部引脚 T1，上升沿

当 T/C1 由外部引脚 T1 驱动时，即使 PD5（T1）配置为输出，管脚上的信号变化照样可以使计数器发生相应的变化。这就为用户提供了一个软件控制的方法。

T/C1—TCNT1H 和 TCNT1L

BIT	15	14	13	12	11	10	9	8
$2D($4D)	MSB							
$2C($4C)								LSB
	7	6	5	4	3	2	1	0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	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
	0	0	0	0	0	0	0	0

此 16 位寄存器包含了 T/C1 的值。当 CPU 访问这两个寄存器时，为了保证高字节和低字节能够同时读写，要用到一个 8 位的临时寄存器 TEMP。此寄存器在访问 OCR1A 和 ICR1 的时候也要用到。如果主程序和中断程序在访问寄存器时都要用到 TEMP，那么在适当的时候需要关闭中断使能防止出错。

- 写 TCNT1：
  当 CPU 写 TCNT1H 时，数据将被放置在 TEMP 寄存器。当 CPU 写低字节 TCNT1L 时，此数据及 TEMP 中的数据一并写入 TCNT1。因此，在写 TCNT1（16 位）时，首先要写 TCNT1H。
- 读 TCNT1：
  当 CPU 读取 TCNT1L 时，TCNT1L 的数据将送入 CPU，同时，TCNT1H 将送入 TEMP 寄存器。等到 CPU 读取 TCNT1H 时，TEMP 中的数据送入 CPU。因此，在读 16 位的

TCNT1 时，首先要读 TCNT1L。

T/C1 是向上记数器或上/下记数器（在 PWM 模式下）。若 T/C1 被置数，则 T/C1 将在预置数的基础上记数。

T/C1 输出比较寄存器 A—OCR1AH 和 OCR1AL

BIT	15	14	13	12	11	10	9	8
$2B($4B)	MSB							
$2A($4A)								LSB
	7	6	5	4	3	2	1	0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	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
	0	0	0	0	0	0	0	0

T/C1 输出比较寄存器包含与 T/C1 值连续比较的数据。

由于 OCR1A 为 16 位寄存器，所以在访问时要用到 TEMP 寄存器以保证两个字节的同步更新。其读写过程与读写 TCNT1 相同。

T/C1 输入捕捉寄存器—ICR1H 和 ICR1L

BIT	15	14	13	12	11	10	9	8
$25($45)	MSB							
$24($44)								LSB
	7	6	5	4	3	2	1	0
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	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
	0	0	0	0	0	0	0	0

按照 ICES1 的设定，输入捕捉引脚 ICP 发生上跳变或下跳变时，TCNT1 被送入 ICR1。同时，ICF1 置位。

由于 ICR1 为 16 位寄存器，所以在访问时要用到 TEMP 寄存器以保证同时读取两个字节。读写过程与读写 TCNT1 相同。

PWM 模式下的 T/C1：

选择 PWM 模式后，T/C1 和输出比较寄存器 1-OCR1 共同组成一个 8，9 或 10 位的无尖峰的自由运行的 PWM，其输出引脚为 PB3（OC1）。T/C1 作为上/下计数器，从 0 记数到 TOP，然后反向记数回到 0。当计数器中的数值和 OCR1A 的数值（低 8，9 或 10 位）一致时，PB3（OC1）引脚按照 COM1A0 和 COM1A1 的设置动作。

表 11　TOP 值及 PWM 频率

PWM 分辨率	TOP 的值	频率
8 位	$00FF（255）	F_{TC1}/510
9 位	$01FF（511）	F_{TC1}/1022
10 位	$03FF（1023）	F_{TC1}/2046

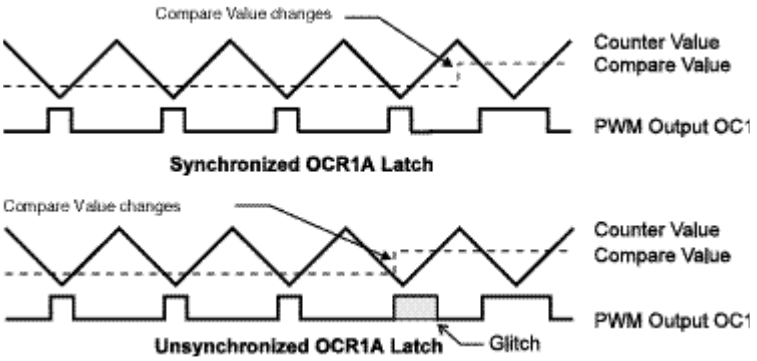
表 12　PWM 模式下的比较 1 模式选择

COM1A1	COM1A0	OC1
0	0	不用作 PWM 功能
0	1	不用作 PWM 功能
1	0	向上记数时的匹配清除 OC1；而向下记数时的匹配置位 OC1（正向 PWM）
1	1	向下记数时的匹配清除 OC1；而向上记数时的匹配置位 OC1（反向 PWM）

注： 在 PWM 模式下，OCR1A 的低 10 位首先存储在一个临时的位置，等到 T/C1 达到 TOP 时才真正存入

OCR1A。这样可以防止在写 OCR1A 时由于失步而出现奇数长度的 PWM 脉冲。

图 32　失步的 OCR1 锁存



如果在执行写和锁存操作的时候读取 OCR1A，读到的是临时位置的数据。

OCR1 的值为$0000 或 TOP 时 OC1 的输出见表 13。

表 13　OCR=$0000 或 TOP 时的 PWM 输出

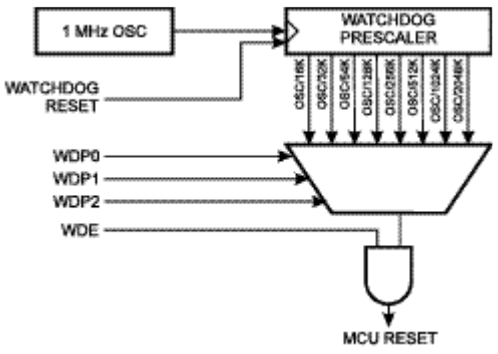
COM1A1	COM1A0	OCR1A	OC1
1	0	$0000	L
1	0	TOP	H
1	1	$0000	H
1	1	TOP	L

在 PWM 模式下，当计数器达到$0000 时将置位 TOV1。此时发生的中断与正常情况下的中断是完全一样的。

看门狗定时器

看门狗定时器由片内独立的振荡器驱动。在 V_{CC}=5V 的条件下，典型振荡频率为 1MHz。通过调整定时器的预分频因数（8 种），可以改变看门狗复位时间间隔。看门狗复位指令是 WDT。如果定时时间已经到，而且没有执行 WDT 指令，则看门狗将复位 MCU。2313 从复位地址重新开始执行。为了防止不小心关闭看门狗，需要有一个特定的关闭程序。

图 33　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)	-	-	-	WDTOE	WDE	WDP2	WDP1	WDP0
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

位 7..5：保留

WDTOE：看门狗关闭使能

当 WDE 清零时此位必须为“1”才能关闭看门狗。在置位的 4 个时钟后，硬件对其清零。

WDE：看门狗使能

WDE 为“1”时，看门狗使能。只有在 WDTOE 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 WDTOE 和 WDE 写逻辑 1，即使 WDE 已经为“1”。

2．在 4 个时钟之内，对 WDE 写逻辑 0。

WDP2..0：预分频器

表 14　　看门狗定时器预分频选择

WDP2	WDP1	WDP0	振荡周期	典型溢出时间 V_{CC}=3V	典型溢出时间 V_{CC}=5V
0	0	0	16K	47ms	15ms
0	0	1	32K	94ms	30ms
0	1	0	64K	0.19s	60ms
0	1	1	128K	0.38s	0.12s
1	0	0	256K	0.75s	0.24s
1	0	1	512K	1.5s	0.49s
1	1	0	1024K	3.0s	0.97s
1	1	1	2048K	6.0s	1.9s

注：看门狗的振荡频率于电压有关。

WDT 应该在看门狗使能之前执行一次。如果看门狗在复位之前使能，则看门狗定时器有可能不是从 0 开始记数。

EEPROM 读/写

EEPROM 访问寄存器位于 I/O 空间。

写 EEP 的时间与电压有关，大概在 2.5~4ms 之间。自定时功能可以让用户监测何时开始写下一字节。如果用户要操作 EEPROM，应当注意如下问题：在电源滤波时间常数比较大的电路中，上电/下电时 V_{CC} 上升/下降会比较慢。此时 MCU 将工作于低于晶振所要求的电源电压。在这种情况下，程序指针有可能跑飞，并执行 EEP 写指令。为了保证 EEP 的数据完整性，建议使用电压复位电路。

为了防止无意识的 EEPROM 写操作，需要执行一个特定的写时序。具体参看后续内容。

当执行 EEPROM 读/写操作时，CPU 会停止工作 2 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
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

位 7：保留

EEAR6..EEAR0：

EEPROM 的地址是线性的。

EEPROM 数据寄存器—EEDR

BIT	7	6	5	4	3	2	1	0
$1D($3D)	MSB							LSB
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

EEDR7..EEDR0：EEPROM 数据

对于 EEPROM 写操作，EEDR 是需要写到 DDAR 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)	-	-	-	-	-	**EEMWE**	**EEWE**	**EERE**
读/写	R	R	R	R	R	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位 7..3：保留

EEMWE：EEPROM主写使能

EEMWE 决定是否设置 EEWE 为“1”以写 EEPROM。当 EEMWE 为“1”时，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。

EEWE：EEPROM写使能

当 EEP 数据和地址设置好之后，需置位 EEWE 以便将数据写入 EEPROM。写时序如下（第 2 和第 3 步不是必须的）：

1. 等待 EEWE 为 0；
2. 将 EEP 的新地址写入 EEAR；
3. 将新数据写入 EEDR；
4. 置位 EEMWE；
5. 在置位 EEMWE 的 4 个周期内，对 EEWE 写逻辑 1。

经过写访问时间（V_{CC}=2.7V 时为 4ms 左右，V_{CC}=5V 时为 2.5ms 左右）之后，EEWE 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 2 个周期。

注意：发生在步骤 4 和 5 之间的中断将导致写操作失败。如果一个操作 EEP 的中断打断了 EEP 操作，RRAR 或 EEDR 寄存器可能被修改，引起 EEP 操作失败。建议此时关闭全局中断标志 I。

EERE：EEPROM读使能

当 EEP 地址设置好之后，需置位 EERE 以便将数据读入 EEDR。EERE 清零表示 EEPROM 的数据已经读入 EEDR。EEPROM 数据的读取只需要一条指令，且无需等待。EERE 置位后，CPU 要停止 2 个周期。

用户在读取 EEP 时应该检测 EEWE。如果一个写操作正在进行，写 EEAR 和 EEDR 将中断 EEP 的写入，使得结果无法预测。

防止 EEPROM数据毁坏：

由于电源电压过低，CPU 和 EEPROM 有可能工作不正常，造成 EEPROM 数据的毁坏。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：一是电压低至 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 3 种方法解决：

1、 当电压过低时保持/RESET 信号为低。这可以通过外加复位电路（BOD—Brown-out Detection）来完成。有些 AVR 产品本身就内含 BOD 电路。详情请看有关数据手册。
2、 当 V_{CC} 过低时使 AVR 内核处于掉电休眠状态。这可以防止 CPU 对程序解码和执行代码，有效防止对 EEPROM 的误操作。
3、 将那些不需修改的常数存储于 FLASH 之中。

UART

AT90S2313 具有全双工通用异步收发器。其主要特点为：

- 波特率发生器可以产生大量的波特率（bps）
- 在低时钟下仍然可以得到高的波特率
- 8 或 9 位数据
- 噪声滤波
- 过速检测
- 帧错误检测
- 错误起始位检测
- 3 个独立的中断：发送结束，发送数据寄存器空，接收结束

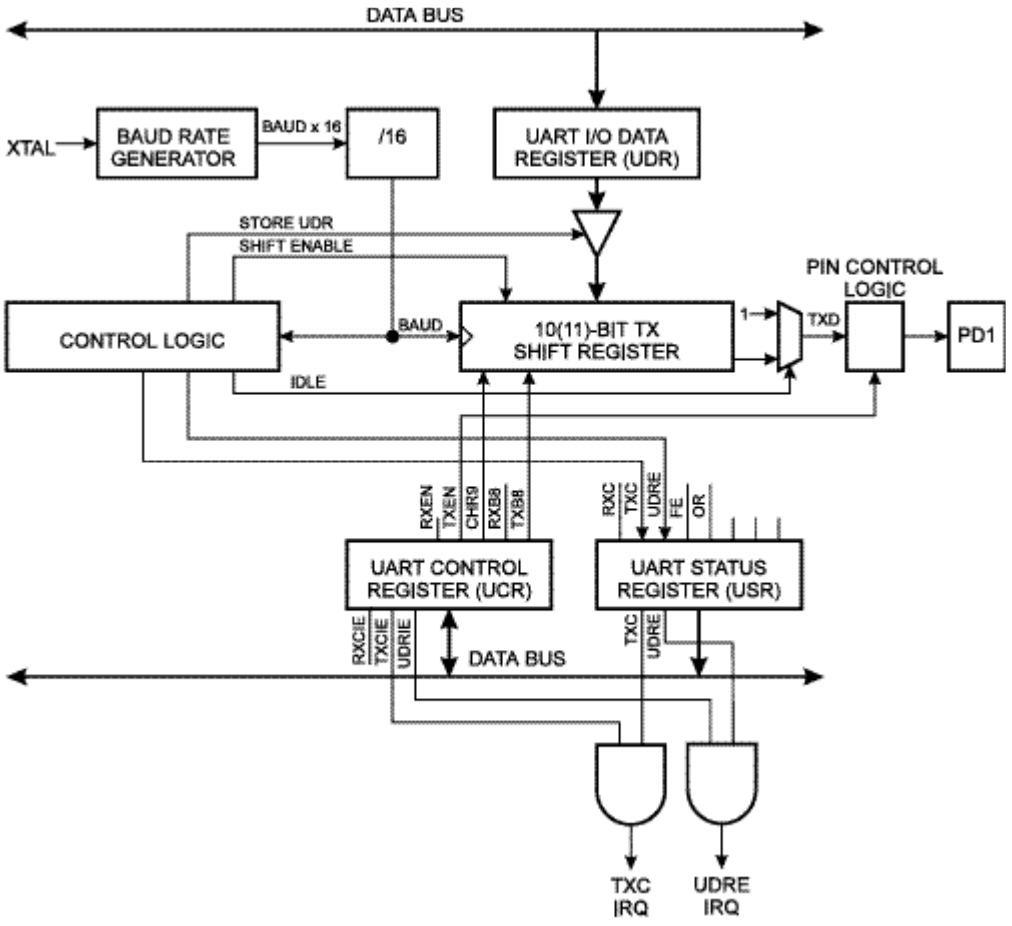
数据发送

图 34 为 UART 发送器的原理图。

把待发送的数据写入 UART 数据寄存器 UDR 将起动数据发送。在如下情况下 UDR 的数据进入发送移位寄存器：

- 若前一个字符的停止位已经移出移位寄存器，则 UDR 的数据立即送入移位寄存器。
- 若前一个字符的停止位还没有移出移位寄存器，则要等到停止位移出后，UDR 的数据才送入移位寄存器。

图 34 UART 发送器



如果 10（11）位收发器移位寄存器为空，UDR 中的数据将传送到移位寄存器。此时 UDRE

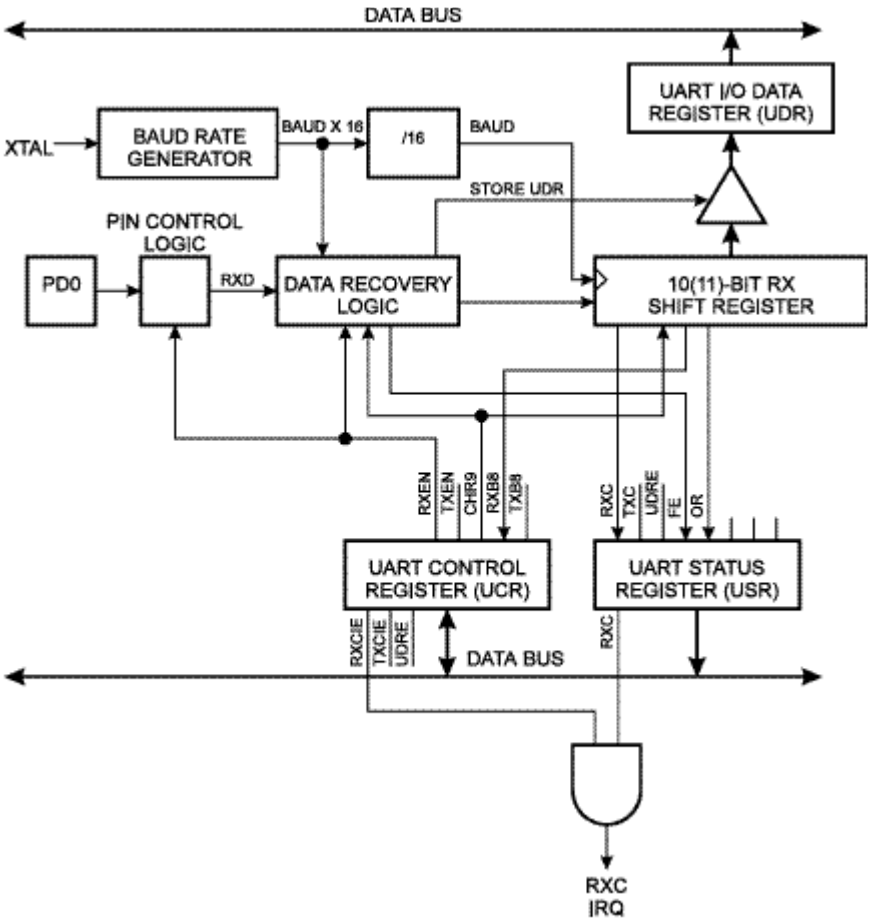
置位，表明 UART 可以接收下一个数据。当数据送入移位寄存器的时候，移位寄存器的 0 位（起始位）自动清零，而位 9 和 10（停止位）置位。如果选择了 9 位数据格式（UART 的 CHR9 置位），则 UCR 中的 TXB8 将送到移位寄存器的位 9。

UART 首先在 TXD 引脚送出起始位，然后是数据，低位在前。如果 UDR 里有新数据，则 UART 会在停止位发送完毕只有自动加载数据。在加载数据的同时，UDRE 置位，并一直保持到有新数据写入 UDR。如果 UDR 没有新数据，而且停止位也已经输出一个 bit 的长度，则发送结束标志 TXC 置位。

UCR 的 TXEN 使能 UART 发送器。当 TXEN 为“0”时，PD1 可用作普通 I/O 口。当 TXEN 为“1”时，UART 输出自动连接到 PD1，强迫其为输出，而不管方向寄存器的设置。

数据接收

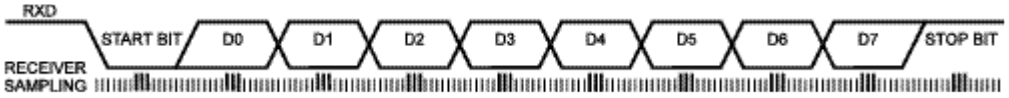
图 35　UART 接收器



接收器的前端以 16 倍于波特率的频率对 RXD 引脚进行采样。如果在此管脚处于空闲状态的时候检测到低电平，则认为这是起始位的下降沿，起始位检测序列开始。假定采样 1 为第一次检测到低电平的时刻。CPU 会在采样 8，9 和 10 对 RXD 进行 3 次连续采样。如果有两个或全部采样值为高，则认为当前信号是一个虚假的起始位，要丢弃。MCU 将开始等待新一次的 1 到 0 的转换。

如果检测到了一个有效的起始位，MCU 就会开始采样数据。数据位的检测同样是在采样 8，9 和 10。两个或 3 个相同的值被认为是当前位的值。图 36 说明了采样过程。

图 36　采样接收到的数据



当停止位进入接收器时，3 个采样值当中必须要有两个以上为高。否则，USR 中的帧错误标志位 FE 置位。在读 UDR 之前，用户应该检查 FE。

不管停止位有效与否，接收到的数据都将被送入 UDR，USR 的 RXC 置位。UDR 实际上是两个物理上分离的寄存器，一个用于发送，一个用于接收。读取 UDR 时，访问的是接收 UDR，而在写 UDR 时，访问的是发送 UDR。如果数据格式为 9 位，则 UCR 的 RXB8 在数据传送到 UDR 时加载到发送移位寄存器的位 9。

如果在读取 UDR 之前，UART 又接收到一个数据，则 USR 的 OR 置位。这表明数据无法转移到 UDR 而丢失了。OR 一直保持到 UDR 被读取。因此，如果波特率比较高，或者 CPU 负载比较重，用户应该在读 UDR 时首先检测 OR 标志。

如果 RXEN 为“0”，则接收器不工作。PD0 可以用作普通 I/O 口。而若 RXEN 置位，则 UART 接收器连接到 PD0，强迫其作为输入而不管方向寄存器的设置。

当 UCR 的 CHR9 为“1”时，收发的数据格式为 9 位。要发送的第 9 位是 UCR 的 TXB8（要在写 UDR 之前设置），而接收到的第 9 位是 RXB8。

UART 控制

UART数据寄存器—UDR

BIT	7	6	5	4	3	2	1	0
$0C($2C)	**MSB**							**LSB**
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

UDR 实际上是两个物理上分离的寄存器，一个用于发送，一个用于接收。读取 UDR 时，访问的是接收 UDR，而在写 UDR 时，访问的是发送 UDR。

UART状态寄存器—USR

BIT	7	6	5	4	3	2	1	0
$0B($2B)	**RXC**	**TXC**	**UDRE**	**FE**	**OR**	-	-	-
读/写	R	R	R	R	R	R	R	R
初始值	0	0	1	0	0	0	0	0

位 2..0：保留

RXC：UART接收结束

RXC 置位表示接收到的数据已经从接收移位寄存器传送到 UDR，但是不管数据是否有误。如果 RXCIE 为“1”，则 RXC 置位时将引起接收结束中断。RXC 在读 UDR 时自动被清除。如果采用中断方式，则中断例程必须读一次 UDR 以清除 RXC，否则中断结束后又会引发中断。

TXC：UART发送结束

TXC 置位表示数据已经从发送移位寄存器发送出去，且 UDR 中没有新的要发送的数据。在半双工通信应用当中，由于发送器在发送完数据之后要立即转换到接收模式，所以这个标志位特别有用。

如果 TXCIE 已置位，则 TXC 置位将引发发送结束中断。进入中断例程后 TXC 自动清零，或者用户可以对其写“1”以达到清零的目的。

UDRE：UART数据寄存器空

当数据从 UDR 传送到发送移位寄存器后，UDRE 置位，表明发送器已经准备好接收新的要发送的数据。

当 UDRIE 置位，则只要 UDRE 为“1”，UART 发送结束中断就可以执行。写 UDR 将复位 UDRE。如果利用中断方式发送数据，则在 UART 数据寄存器空中断例程里必须写 UDR 以清除 UDRE，否则中断将连续发生。

复位后 UDRE 的初始值为“1”，表明发送器就绪。

FE：帧错误

MCU 检测到帧错误（如：检测到停止位为“0”）时 FE 置位。

当检测到数据停止位为“1”时 FE 复位。

OR：过速

如果 UDR 未读，而新的数据又已进入移位寄存器，则 OR 置位。

UDR 数据移入后 OR 清零。

UART控制寄存器—UCR

BIT	7	6	5	4	3	2	1	0
$0A($2A)	RXCIE	TXCIE	UDRIE	RXEN	TXEN	CHR9	RXB8	TXB8
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R	W
初始值	0	0	0	0	0	0	1	0

RXCIE：接收结束中断使能

TXCIE：发送结束中断使能

UDRIE：UART数据寄存器空中断使能

RXEN：接收使能

使能 UART 接收。接收禁止将导致 TXC，OR 和 FE 无法置位，同时也不能复位已经置位的标志位。

TXEN：发送使能

TXEN 清零不能立即关闭发送器，如果发送器当前正在发送的话。只有等到发送完毕后，发送器才真正关闭。

CHR9：9位字符

置位后 MCU 将接收和发送 9 位字符。第 9 位字符的读/写分别由 RXB8/TXB8 控制。第 9 位字符可用作停止位或奇偶校验。

RXB8：接收的第9位

接收到的字符的第 9 位

TXB8：发送的第9位

发送字符的第 9 位

波特率产生器

波特率计算公式为：

$$BAUD = \frac{f_{CK}}{16(UBRR+1)}$$

式中，BAUD = 波特率，

f_{CK} = 晶振时钟频率

UBRR = UART **波特率寄存器的内容**（0-255）

表 15 **给出了在某些晶振下波特率对应的** UBRR **的值。**

表 15　不同频率下 UBRR **的设定值**

波特率	**1MHz**	**%误差**	**1.8432 MHz**	**%误差**	**2MHz**	**%误差**	**2.4576 MHz**	**%误差**
2400	UBRR = 25	0.2	47	0.0	51	0.2	63	0.0
4800	12	0.2	23	0.0	25	0.2	31	0.0
9600	6	7.5	11	0.0	12	0.2	15	0.0
14400	3	7.8	7	0.0	8	3.7	10	3.1
19200	2	7.8	5	0.0	6	7.5	7	0.0
28800	1	7.8	3	0.0	3	7.8	4	6.3
38400	1	22.9	2	0.0	2	7.8	3	0.0
57600	0	7.8	1	0.0	1	7.8	2	12.5
76800	0	22.9	1	33.3	1	22.9	1	0.0
115200	0	84.3	0	0.0	0	7.8	0	25.0

波特率	**3.2768 MHz**	**%误差**	**3.6864 MHz**	**%误差**	**4MHz**	**%误差**	**4.608M Hz**	**%误差**
2400	UBRR = 84	0.4	95	0.0	103	0.2	119	0.0
4800	42	0.8	47	0.0	51	0.2	59	0.0
9600	20	1.6	23	0.0	25	0.2	29	0.0
14400	13	1.6	15	0.0	16	2.1	19	0.0
19200	10	3.1	11	0.0	12	0.2	14	0.0
28800	6	1.6	7	0.0	8	3.7	9	0.0
38400	4	6.3	5	0.0	6	7.5	7	6.7
57600	3	12.5	3	0.0	3	7.8	4	0.0
76800	2	12.5	2	0.0	2	7.8	3	6.7
115200	1	12.5	1	0.0	1	7.8	2	20.0

波特率	**7.3728 MHz**	**%误差**	**8MHz**	**%误差**	**9.216M Hz**	**%误差**	**11.059 MHz**	**%误差**
2400	UBRR = 191	0.0	207	0.2	299	0.0	287	-
4800	95	0.0	103	0.2	119	0.0	143	0.0
9600	47	0.0	51	0.2	59	0.0	71	0.0
14400	31	0.0	34	0.8	39	0.0	47	0.0
19200	23	0.0	25	0.2	29	0.0	35	0.0
28800	15	0.0	16	2.1	19	0.0	23	0.0
38400	11	0.0	12	0.2	14	0.0	17	6.7
57600	7	0.0	8	3.7	9	0.0	11	0.0
76800	5	0.0	6	7.5	7	6.7	8	6.7
115200	3	0.0	3	7.8	4	0.0	5	20.0

UART波特率寄存器—UBRR

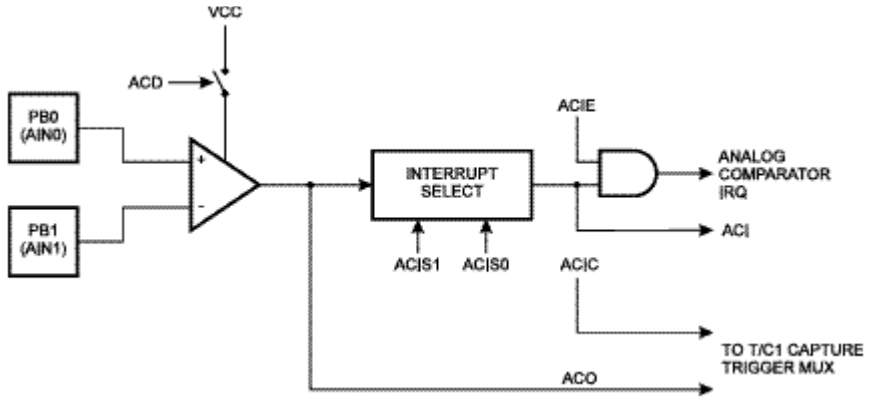
BIT	7	6	5	4	3	2	1	0
$09($29)	**MSB**							**LSB**
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

模拟比较器

模拟比较器比较正输入端 PB0（AIN0）**和负输入端** PB1（AIN1）**的值。如果** PB0（AIN0）

的电压高于 PB1（AIN1）的值，比较器的输出 ACO 将置位。此输出可用来触发模拟比较器中断（上升沿、下降沿或电平变换），也可以触发 T/C1 的输入捕捉功能。其框图如图 37 所示。

图 37 模拟比较器框图



模拟比较器控制和状态寄存器—ACSR

BIT	7	6	5	4	3	2	1	0
$08($28)	ACD	-	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0
读/写	R/W	R	R	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位 6：保留

ACD：模拟比较器禁止

当 ACD 为“1”时模拟比较器的电源将切断。可以在任何时候对其置位以关闭模拟比较器。这样可以减少器件的功耗。改变 ACD 时要注意禁止模拟比较器的中断，否则有可能引发不必要的中断。

ACO：模拟比较器输出

ACO 与比较器的输出直接相连。

ACI：模拟比较器中断标志位

当比较器输出触发中断时 ACI 将置位。中断方式由 ACIS1 和 ACIS0 决定。如果 ACI 和 I 都为“1”，则 CPU 执行比较器中断例程。进入中断例程后，ACI 被硬件清零。此外，ACI 也可以通过对此位写“1”来达到清零的目的。要注意的是，如果 ACSR 的另一些位被 SBI 或 CBI 指令修改时，ACI 亦被清零。

ACIE：模拟比较器中断使能

ACIE 为“1”时，比较器中断使能。

ACIC：模拟比较器输入捕捉使能

ACIC 为“1”时，T/C1 的输入捕捉功能由比较器中断触发。此时，比较器的输出与 T/C1 的输入捕捉前端直接相连，T/C1 的输入捕捉噪声抑制和边沿选择仍然适用。如果 ACIC 为“0”，则模拟比较器与 T/C1 没有关联。为了使能比较器驱动的 T/C1 输入捕捉中断，TICIE1 必须置位。

ACIS1，ACIS0：模拟比较器中断模式选择

表 16 ACIS1/ACIS0 设置

ACIS1	ACIS0	中断模式
0	0	电平变换引发中断
0	1	保留
1	0	（ACO）下降沿中断

1	1	（ACO）上升沿中断

注：改变 ACIS1/ACIS0 时要注意禁止模拟比较器的中断，否则有可能引发不必要的中断。

I/O 口

所有的 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（值、禁止/使能、上拉）时不会无意地改变其他管脚的方向（值、禁止/使能、上拉）。

B 口

B 口是 8 位双向 I/O 口。

B 口有 3 个 I/O 地址：数据寄存器—PORTB，$18（$38），数据方向寄存器—DDRB，$17（$37）和输入引脚—PINB，$16（$36）。PORTB 和 DDRB 可读可写，PINB 只可读。

所有的管脚都可以单独选择上拉电阻。引脚缓冲器可以吸收 20mA 的电流，能够直接驱动 LED。当管脚被拉低时，如果上拉电阻已经激活，则引脚会输出电流。

B 口的第二功能如下表所示：

表 17　B 口第二功能

管脚	第二功能
PB0	AIN0（模拟比较器正输入端）
PB1	AIN1（模拟比较器负输入端）
PB3	OC1（T/C1 比较匹配输出）
PB5	MOSI（程序下载时的数据输入线）
PB6	MISO（程序下载时的数据输出线）
PB7	SCK（串行时钟）

当使用 B 口的第二功能时，DDRB 和 PORTB 要设置成对应的值。

B 口数据寄存器—PORTB

BIT	7	6	5	4	3	2	1	0
$18($38)	**PORTB7**							**PORTB0**
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

B 口数据方向寄存器—DDRB

BIT	7	6	5	4	3	2	1	0
$17($37)	**DDB7**							**DDB0**
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

B 口输入引脚地址—PINB

BIT	7	6	5	4	3	2	1	0
$16($36)	**PINB7**							**PINB0**
读/写	R	R	R	R	R	R	R	R
初始值	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z

PINB 不是一个寄存器，这个地址用来访问 B 口的物理值。读取 PORTB 时，读到的是 B 口锁存的数据；而读取 PINB 时，读到的是施加于引脚上的逻辑数值。

B 口用作通用数字 I/O

作为通用数字 I/O 时，B 口的 8 个管脚具有相同的功能。

PBn，通用 I/O 引脚：DDRB 中的 DDBn 选择引脚的方向。如果 DDBn 为“1”，则 PBn 为输出脚；如果 DDBn 为“0”，则 PBn 为输入脚。在复位期间，B 口为三态口。

表 18　B 口的配置

DDBn	PORTBn	I/O	上拉	注释
0	0	输入	N	三态（高阻）
0	1	输入	Y	外部拉低时会输出电流
1	0	输出	N	推挽 0 输出
1	1	输出	N	推挽 1 输出

n：7，6..0，引脚号

B 口的第二功能

- SCK—PB7
  下载程序时的时钟
- MISO—PB6
  程序上载时的输出数据
- MOSI—PB5
  下载程序时的数据
- OC1—PB3
  输出比较匹配
- AIN1—PB1
  当配置为输入（DDB1=0），无上拉电阻（PB1=0）时，为模拟比较器的负输入端
- AIN0—PB0
  当配置为输入（DDB0=0），无上拉电阻（PB0=0）时，为模拟比较器的正输入端

B 口示意图

图 38　B 口示意图（PB0 **和** PB1）

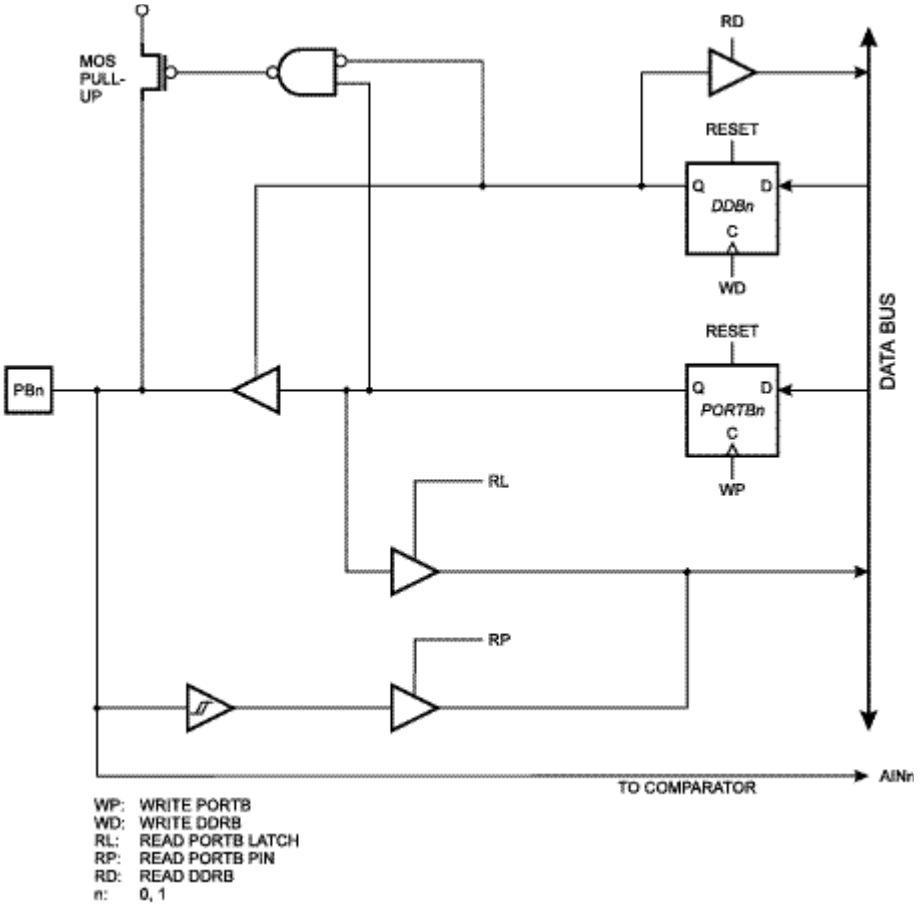


图 39　B 口示意图（PB3）

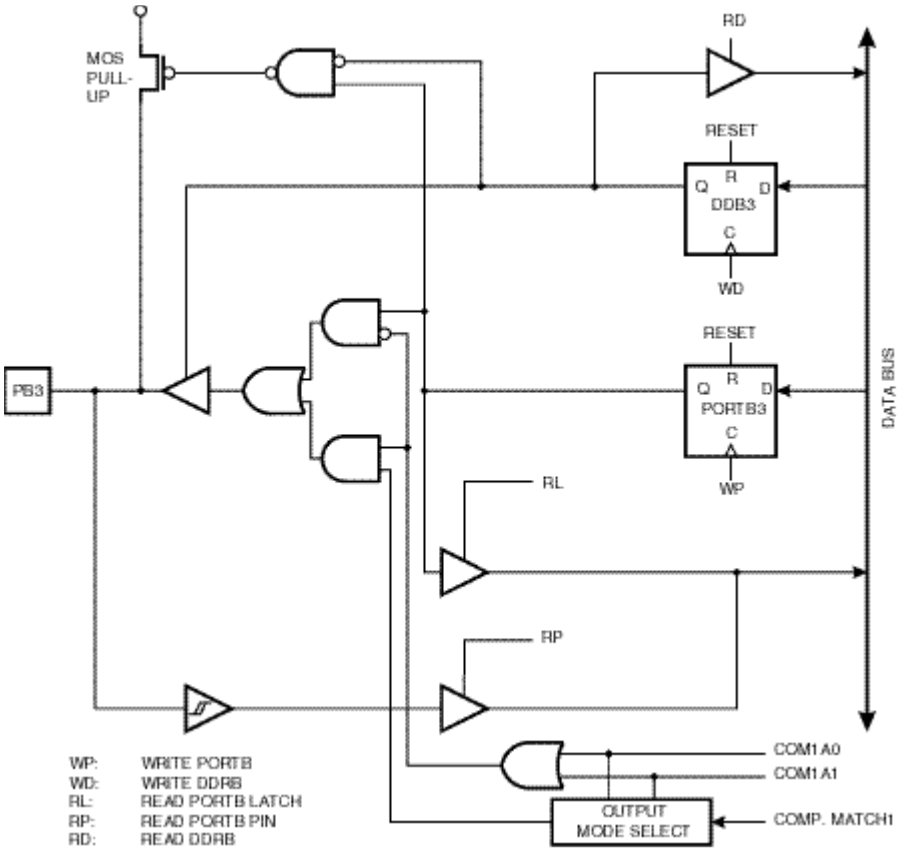


图 40　B 口示意图（PB2 **和** PB4）

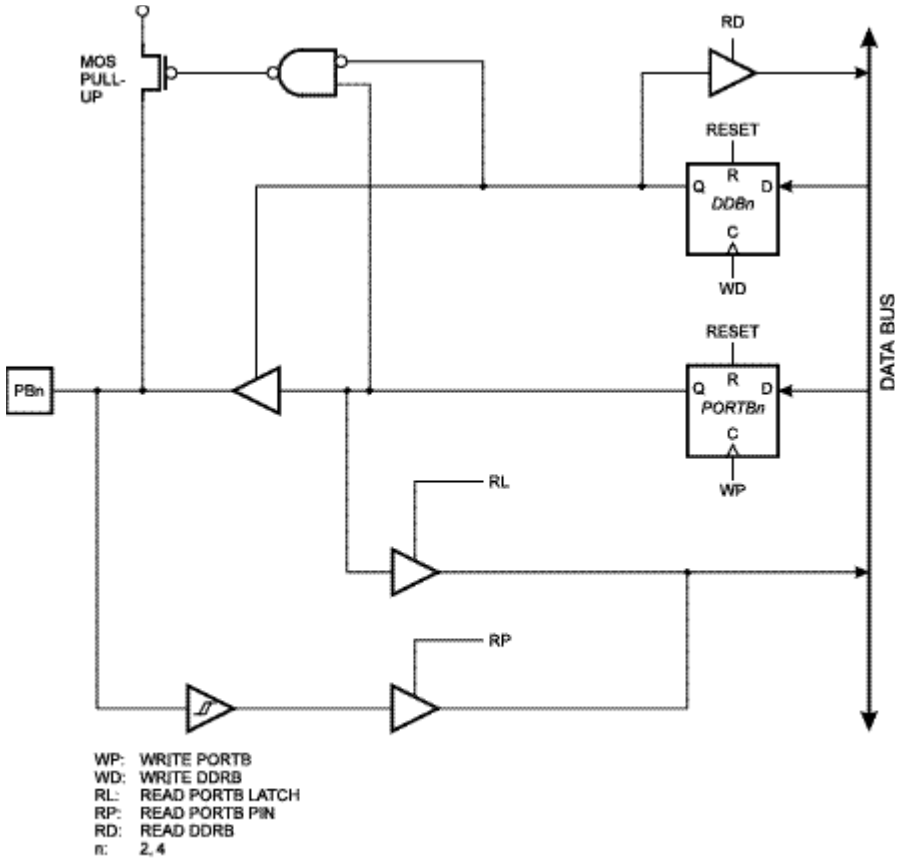


图 41　B 口示意图（PB5）

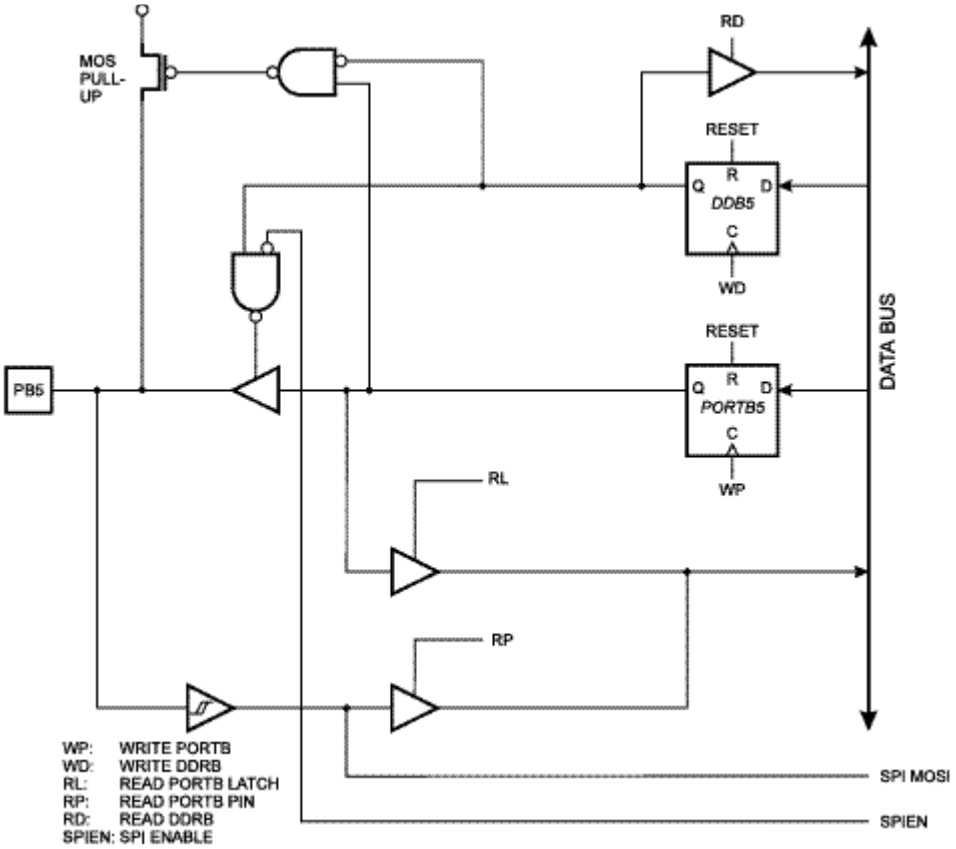


图 42　B 口示意图（PB6）

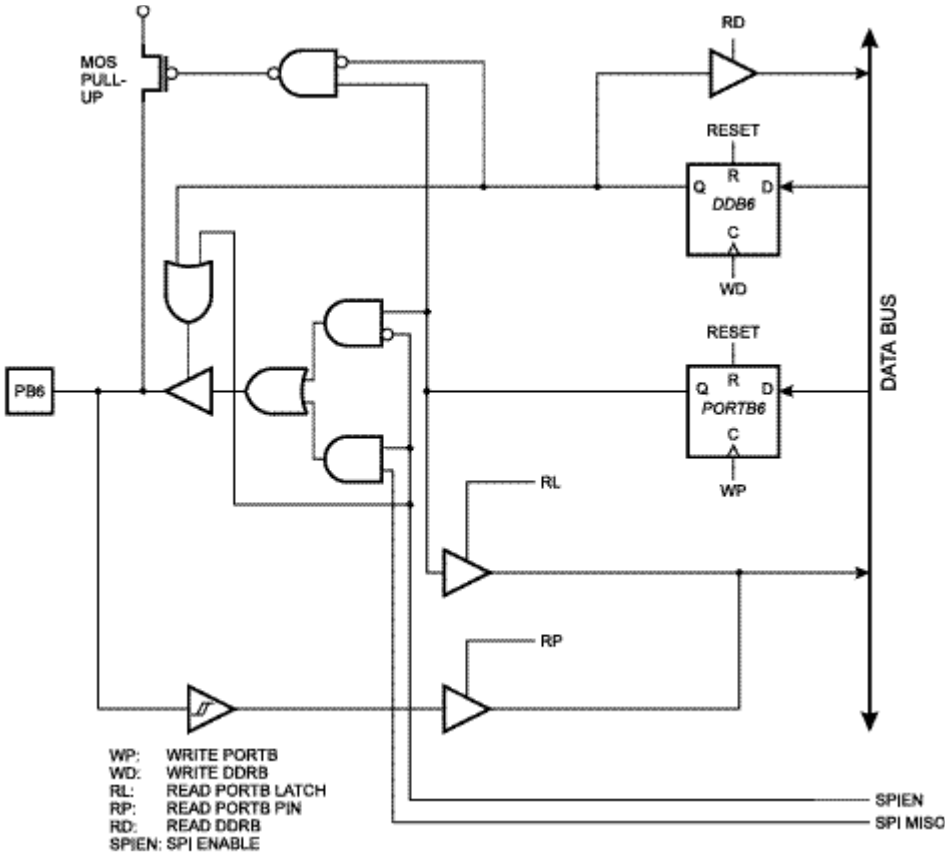
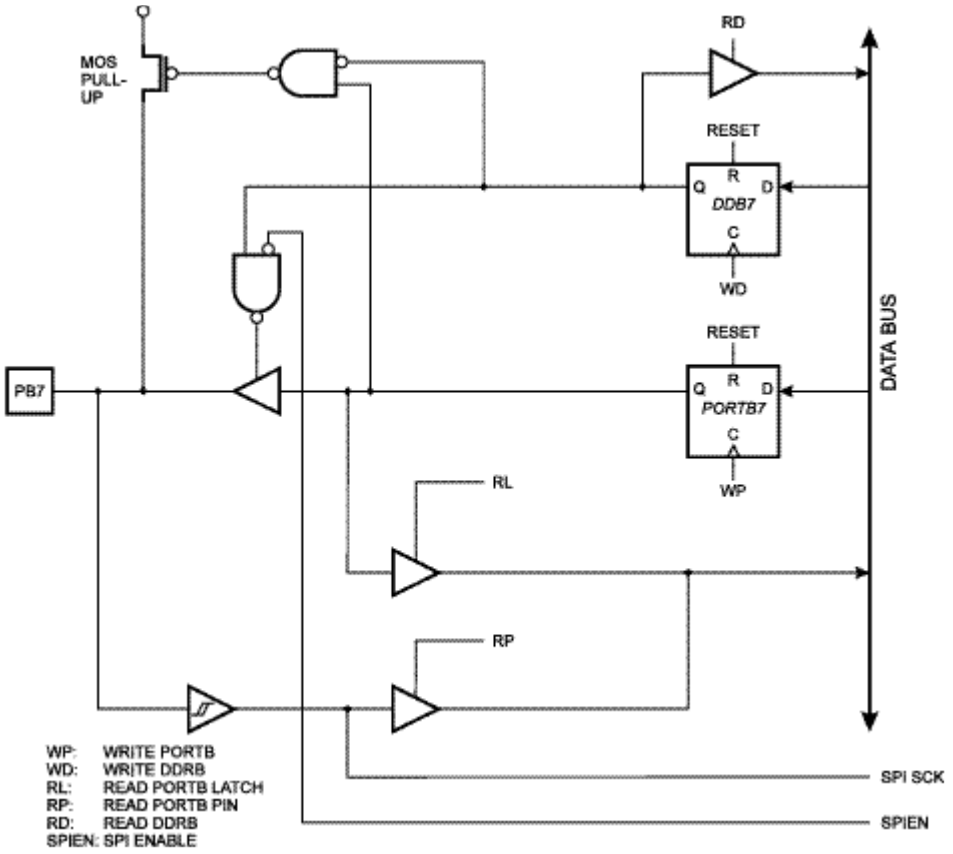


图 43　B 口示意图（PB7）



D 口

D 口有 3 个 I/O 地址：数据寄存器—PORTD，$12（$32），数据方向寄存器—DDRD，$11（$31）和输入引脚—PIND，$10（$30）。PORTD 和 DDRD 可读可写，PIND 只可读。

D 口有 7 个带上拉电阻的双向 I/O 管脚，PD6~PD0。引脚缓冲器可以吸收 20mA 的电流，能够直接驱动 LED。当管脚被拉低时，如果上拉电阻已经激活，则引脚会输出电流。

D 口的第二功能如下表所示：

表 19　D 口第二功能

管脚	第二功能
PD0	RXD（UART 接收引脚）
PD1	TXD（UART 发送引脚）
PD2	INT0（外部中断 0 输入）
PD3	INT1（外部中断 1 输入）
PD4	T0（T/C0 的外部输入）
PD5	T1（T/C1 的外部输入）
PD6	ICP（T/C1 的输入捕捉引脚）

D 口数据寄存器—PORTD

BIT	7	6	5	4	3	2	1	0
$12($32)	-	PORTD6						PORTD0
读/写	R	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

D 口数据方向寄存器—DDRD

BIT	7	6	5	4	3	2	1	0
$11($31)	-	DDD6						DDB0
读/写	R	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

D 口输入引脚地址—PIND

BIT	7	6	5	4	3	2	1	0
$10($30)	-	PIND6						PINB0
读/写	R	R	R	R	R	R	R	R
初始值	0	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z

PIND 不是一个寄存器，这个地址用来访问 D 口的物理值。读取 PORTD 时，读到的是 D 口锁存的数据；而读取 PIND 时，读到的是施加于引脚上的逻辑数值。

D 口用作通用数字 I/O

PDn，通用 I/O 引脚：DDRD 中的 DDDn 选择引脚的方向。如果 DDDn 为“1”，则 PDn 为输出脚；如果 DDDn 为“0”，则 PDn 为输入脚。在复位期间，D 口为三态口。

表 20　D 口的配置

DDDn	PORTDn	I/O	上拉	注释
0	0	输入	N	三态（高阻）
0	1	输入	Y	外部拉低时会输出电流
1	0	输出	N	推挽 0 输出
1	1	输出	N	推挽 1 输出

n：6.0，引脚号

D 口的第二功能

- ICP—PD6
  T/C1 的输入捕捉引脚
- T1—PD5
  定时器 1 的外部时钟源
- T0—PD4
  定时器 1 的外部时钟源
- INT1—PD3
  外部中断源 1
- INT0—PB2
  外部中断源 0
- TXD—PD1
  发送数据引脚。发送器使能后，引脚自动配置为输出而不管 DDR1。
- RXD—PD0
  接收数据引脚。接收器使能后，引脚自动配置为输入而不管 DDR0。若此时 PORTD0 为“1”，则上拉有效。

D 口示意图

图 44　D 口示意图（PD0）

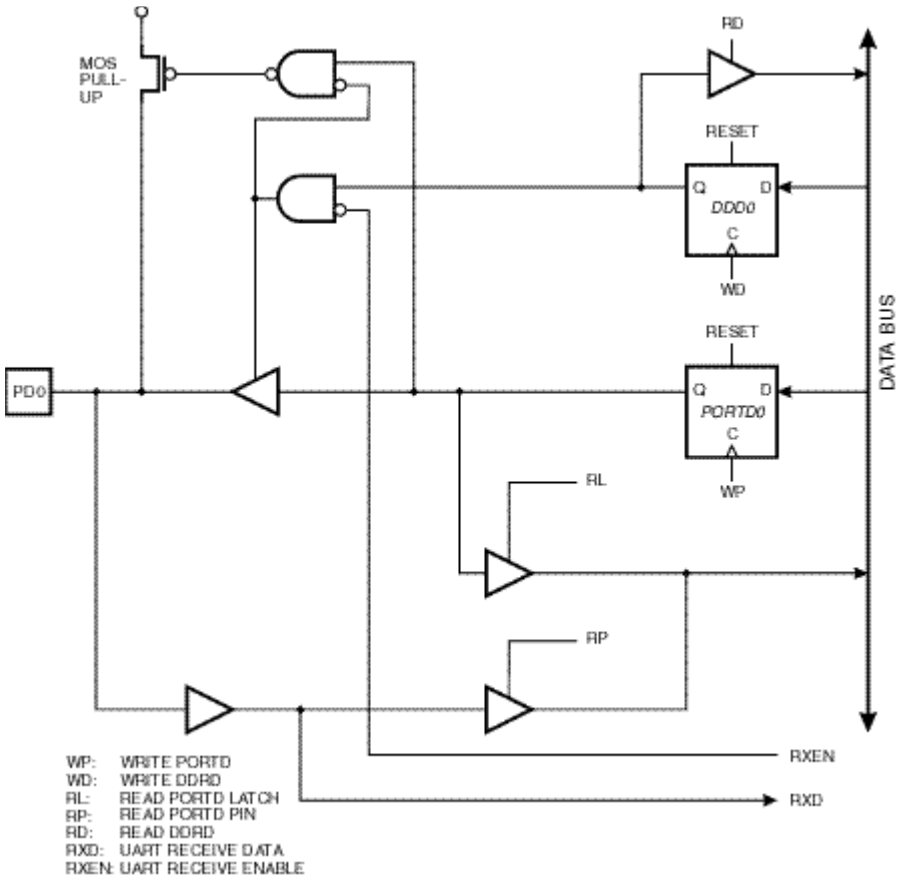


图 45　D 口示意图（PD1）

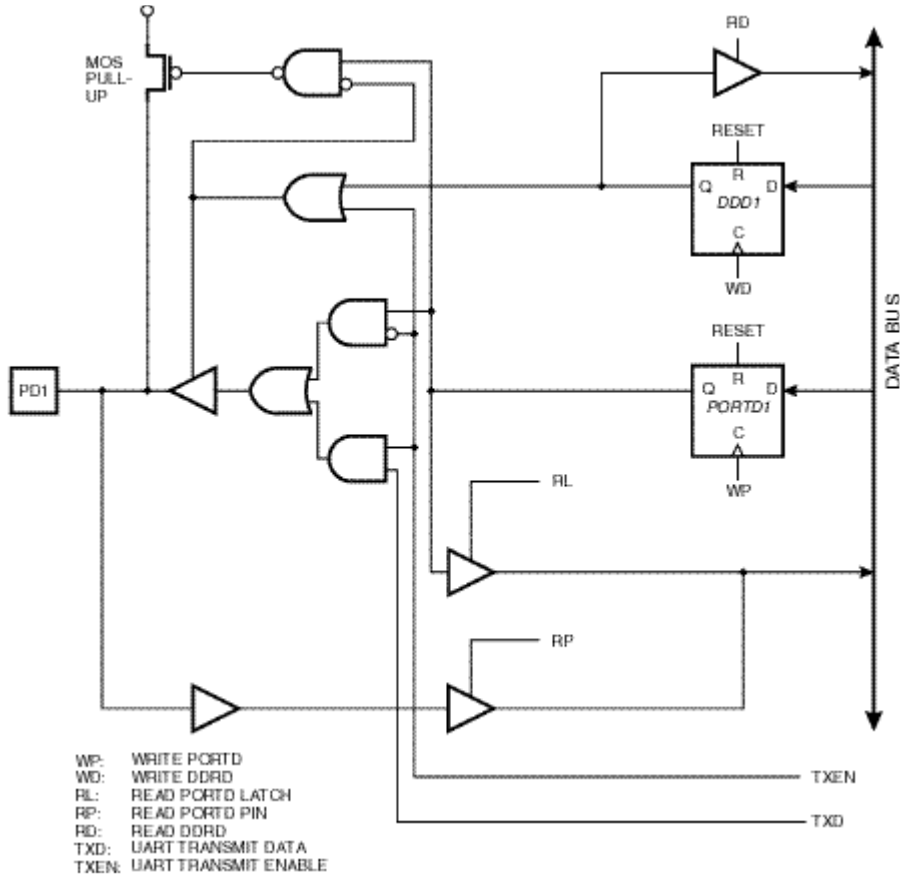


图 46　D 口示意图（PD2 **和** PD3）

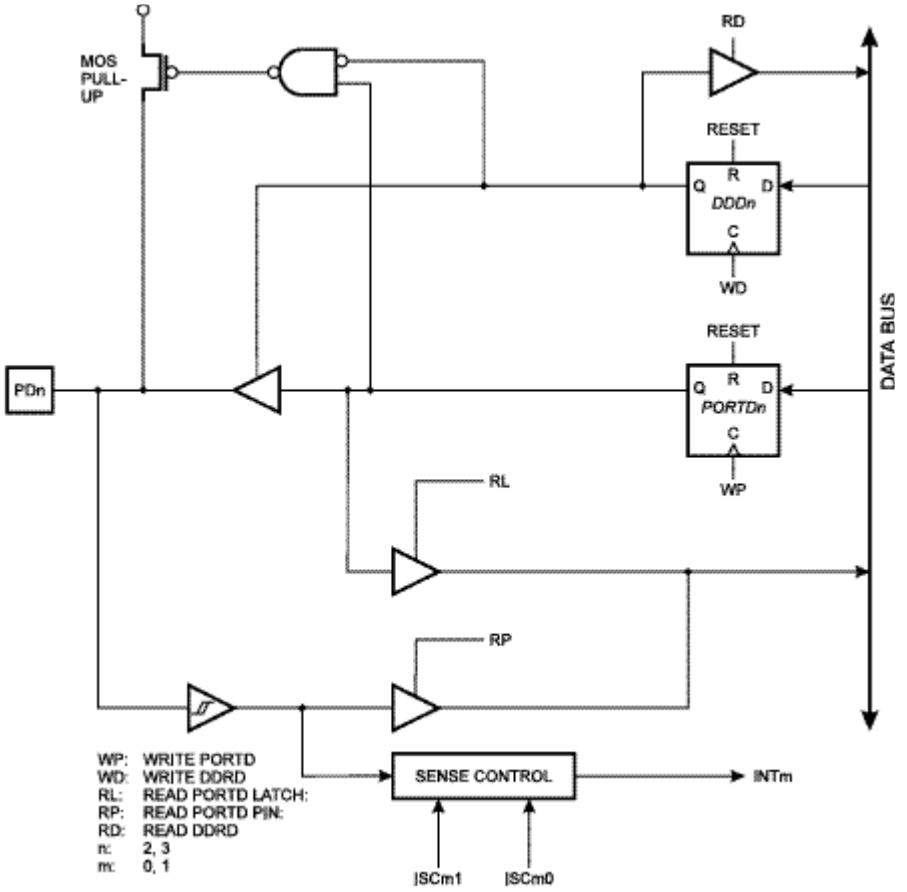


图 47　D 口示意图（PD4 **和** PD5）

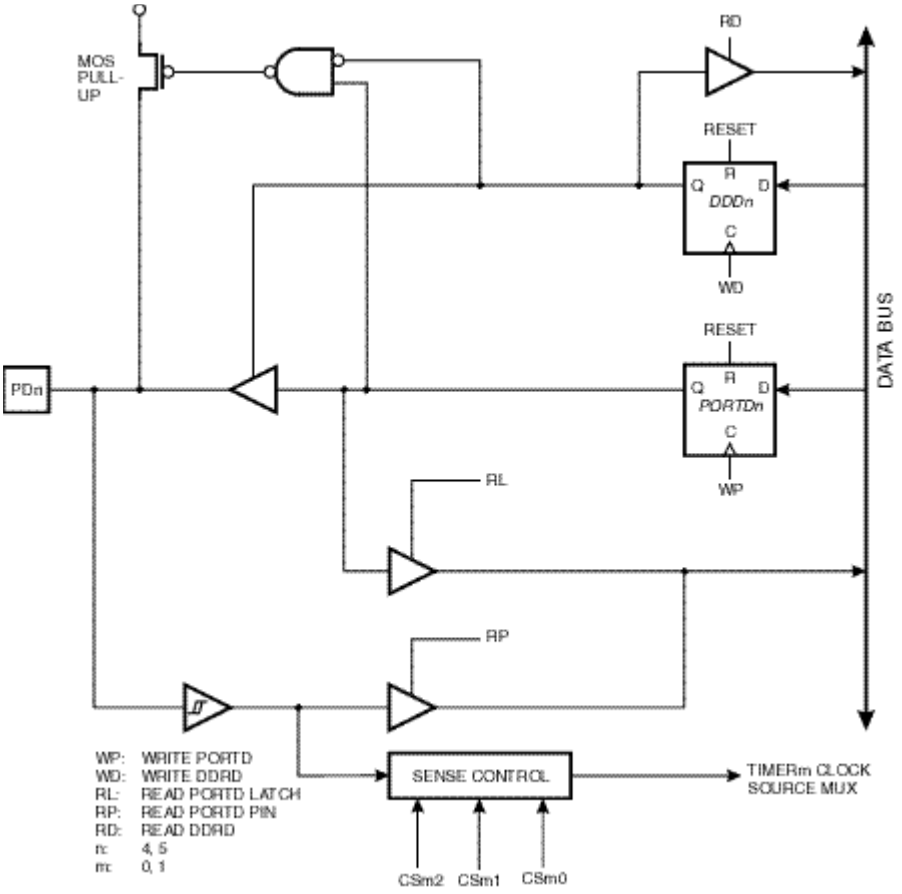
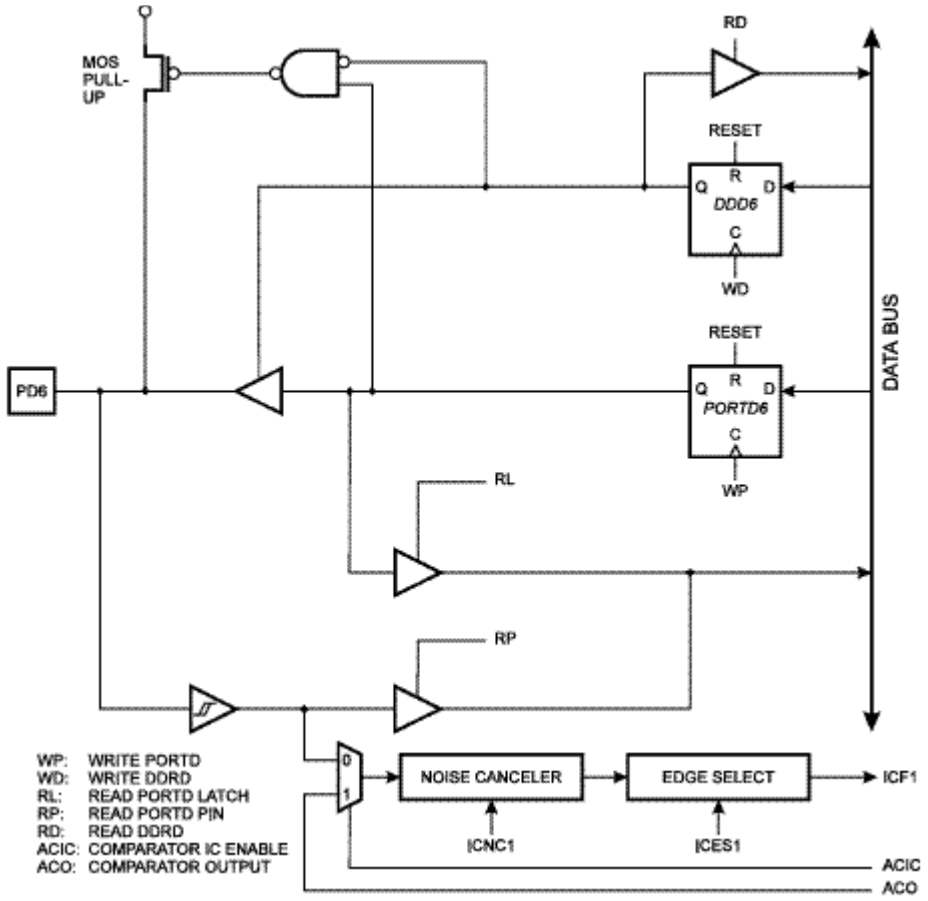


图 48　D 口示意图（PD6）



程序编程

程序和数据锁定位

AT90S2313 具有两个锁定位，如表 21 所示。锁定位只能通过片擦除命令擦除。

表 21　锁定保护模式

程序锁定位			保护类型
模式	LB1	LB2	
1	1	1	无锁定功能
2	0	1	禁止编程 (1)
3	0	0	禁止校验

注意：1、在并行编程模式下，熔断位编程也被禁止。要先编程熔断位，然后编程锁定位。

熔断位

AT90S2313 有两个熔断位：SPIEN 和 FSTRT。

- SPIEN 编程（为“0”）后，串行下载程序使能。缺省值为“0”。
- FSTRT 编程（为“0”）后，MCU 选择短起动时间。缺省值为“1”。量大时用户也可以订购缺省值为“0”的产品。

串行下载程序时不能访问熔断位，只能在并行下载程序时访问。芯片擦除命令不影响熔断位。

厂标

所有的 Atmel 微处理器都有 3 字节的厂标，用以识别器件。此代码在串行或并行模式下都可以访问。其位置为：

1、$000：$1E（表明是 Atmel 生产的）

2、$001：$91（2K 字节的 FLASH）

3、$002：$01（当$01 地址为$91 时，器件为 2313）

注意：在锁定保护模式 3 有效时，厂标不能以串行模式读出。其返回值将为$00，$01 和$02。

编程 FLASH 和 EEPROM

AT90S2313 具有 2K 字节的片内可编程 FLASH 和 128 字节的 EEPROM，在出厂时已经被擦除为“1”。器件支持+12V 高压并行编程和低压串行编程。+12V 只用来使能高压编程，不会有明显的电流流过。

在两种编程模式下，FLASH 是以字节的形式写入的。而对于 EEPROM，片内集成了自擦和自定时除功能。在编程时，要注意电源电压要满足要求。

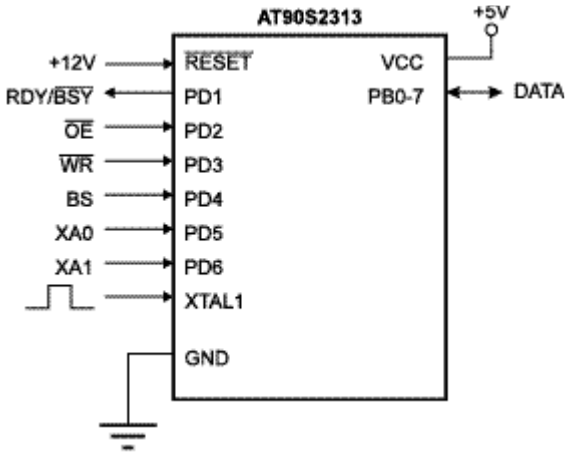
表 22 编程电源电压

型号	串行编程	并行编程
AT90S2313	2.7V – 6.0V	4.5V – 5.5V

并行编程

本节介绍如何在并行模式下对 FLASH、EEPROM、熔断位和锁定位进行编程。

图 49 并行编程



信号命名：

AT90S2313 的某些管脚在本节将以并行编程的信号来命名。XA1/XA0 决定了当 XTAL1 引脚上出现正脉冲时要进行的动作。

当驱动/WE 或/OE 时，执行加载的命令。

表 23 管脚命名

编程时的信号名称	管脚	I/O	功能
RDY/BSY	PD1	O	0：器件忙 1：可以接受新命令
/OE	PD2	I	输出使能（低电平）
/WR	PD3	I	写脉冲（低电平）

BS	PD4	I	字节选择（“0”：低字节 “1”：高字节）
XA0	PD5	I	见表 24
XA1	PD6	I	见表 24
DATA	PB0-7	I/O	双向数据总线（/OE 为低时输出）

表 24 XA1 和 XA0 编码

XA1	XA0	XTAL1 给正脉冲后的操作
0	0	加载 FLASH 或 EEPROM 的地址（BS 决定是高位地址还是低位地址）
0	1	加载数据（BS 决定是高位地址还是低位地址）
1	0	加载命令
1	1	无操作

表 25 命令字节编码

命令字节	执行的命令
1000 0000	芯片擦除
0100 0000	写熔断位
0010 0000	写锁定位
0001 0000	写 FLASH
0001 0001	写 EEPROM
0000 1000	读厂标
0000 0100	读熔断位和锁定位
0000 0010	读 FLASH
0000 0011	读 EEPROM

进入编程模式：

以下步骤使器件进入并行编程模式：

1、按表 22 在 V_{CC} 和 GND 之间加上电源电压

2、拉低/RESET 和 BS，等待至少 100ns

3、给/RESET 引脚加上 11.5~12.5V 的电压。如果 BS 在/RESET 加上+12V 之后 100ns 之内发生动作，将导致器件无法进入编程模式。

芯片擦除：

此命令擦除所有的 FLASH 和 EEPROM，以及锁定位。在 FLASH 和 EEPROM 完全擦除之前，锁定位不会擦除。擦除过程不影响熔断位。擦除命令必须在对 FLASH 和 EEPROM 重新编程之前执行。

加载“擦除”命令：

1、设置 XA1，XA0 为“10”—使能命令加载

2、设置 BS 为“0”。

3、设置 DATA 为“1000 0000”—擦除命令

4、给 XTAL1 一个正脉冲—加载命令

5、给/WE 施加一个 t_{WLWH_CE} 的负脉冲。擦除命令在 RDY/BSY 引脚没有反馈。

FLASH 编程：

A：加载命令

1、设置 XA1，XA0 为“10”—使能命令加载

2、设置 BS 为“0”。

3、设置 DATA 为“0001 0000”一写 FLASH 命令
4、给 XTAL1 一个正脉冲一加载命令
B：加载高位地址
1、设置 XA1，XA0 为“00”一使能地址加载
2、设置 BS 为“1”一 选择地址的高位字节
3、设置 DATA = 地址的高位字节（$00 - $03）
4、给 XTAL1 一个正脉冲一加载地址的高位字节
C：加载低位地址
1、设置 XA1，XA0 为“00”一使能地址加载
2、设置 BS 为“0”一 选择地址的低位字节
3、设置 DATA = 地址的低位字节（$00~$FF）
4、给 XTAL1 一个正脉冲一加载地址的低位字节
D：加载数据的低位字节
1、设置 XA1，XA0 为“01”一使能数据加载
3、设置 DATA = 数据的低位字节（$00~$FF）
5、给 XTAL1 一个正脉冲一加载数据的低位字节
E：写数据的低位字节
1、设置 BS 为“0”一 选择低位数据
2、给/WR 一个负脉冲一开始编程数据，同时 RDY/BSY 变低
3、等待 RDY/BSY 变高，然后开始编程下一字节
（波形见图 50）
F：加载数据的高位字节
1、设置 XA1，XA0 为“01”一使能数据加载
2、设置 DATA = 数据的高位字节（$00~$FF）
3、给 XTAL1 一个正脉冲一加载数据的低位字节
G：写数据的高位字节
1、 设置 BS 为“1”一 选择高位数据
2、 给/WR 一个负脉冲一开始编程数据，同时 RDY/BSY 变低
3、 等待 RDY/BSY 变高，然后开始编程下一字节
（波形见图 51）
器件在编程时保存加载的命令和地址。为了有效地进行编程，请注意以下几点：

- 当写（读）多个内存地址时，命令只需加载一次。
- 仅在编程新的页（256 字节）时才需要加载高位地址字节。
- 因为芯片擦除之后所有的 FLASH 和 EEPROM 的内容都为“1”，故数据为$FF 时可以跳过编程。

以上几点适用于 EEPROM 的编程，以及 FLASH、EEPROM 和厂标的读取。

图 50 编程波形一

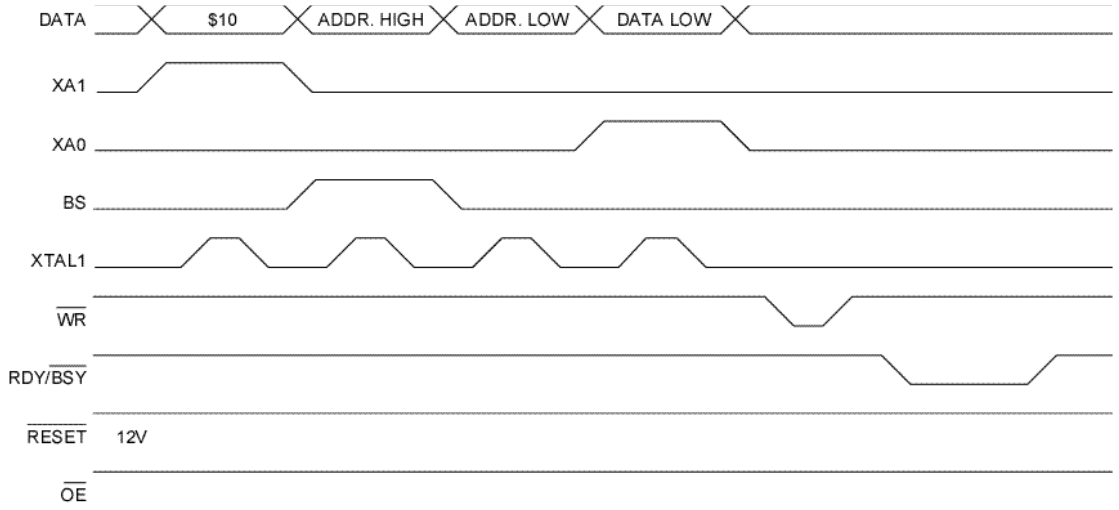
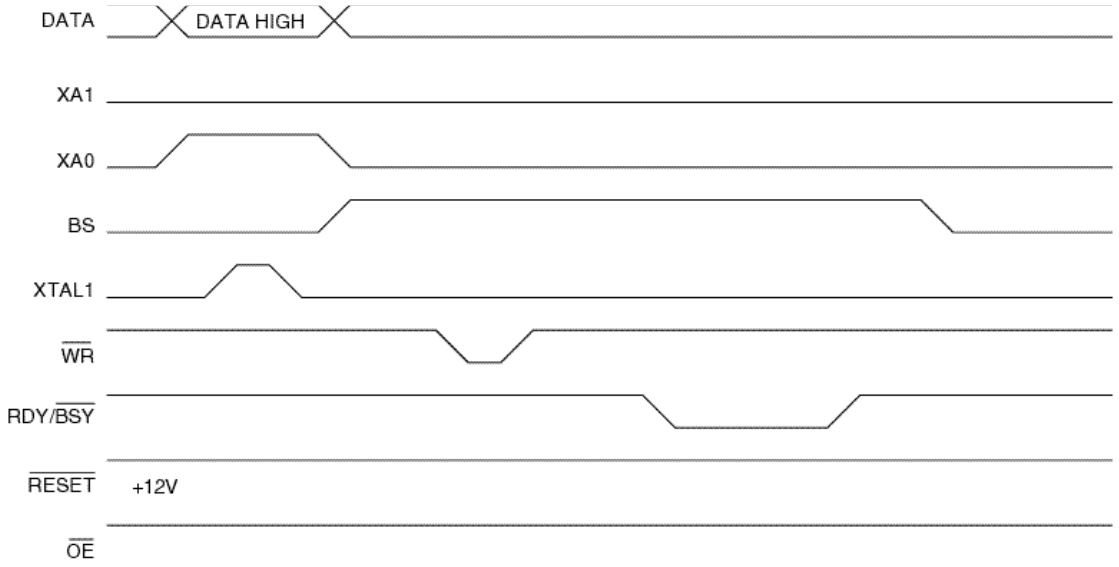


图 51 编程波形二



读 FLASH:

1、A：**加载命令**“0000 0010”

2、B：**加载地址的高位字节**（$00~$01）

3、C：**加载地址的低位字节**（$00~$FF）

4、**设置**/OE **和** BS **为**“0”。**此时可以从** DATA **总线读** FLASH **数据的低位字节。**

5、**设置** BS **为**“1”。 **此时可以读** FLASH **数据的高位字节。**

6、**设置**/OE **为**“1”。

编程 EEPROM:

1、A：**加载命令**“0001 0001”

2、B：**加载地址的低位字节**（$00~$3F）

3、D：**加载数据的低位字节**（$00~$FF）

4、E：**写数据的低位字节**

读 EEPROM:

1、 A：**加载命令**“0000 0011”

2、 B：**加载地址的低位字节**（$00~$3F）

3、 **设置**/OE **和** BS **为**“0”。**此时可以从** DATA **总线读取数据的低位字节。**

4、 **设置**/OE **为**“1”。

编程熔断位：

1、A：**加载命令**“0100 0000”

2、D：**加载数据的低位字节。**Bit n =“0”**代表要编程，** “1”**代表要擦除。**

Bit 5 = SPIEN

Bit 0 = FSTRT

Bit 7- 6, 4 - 1 = “1”。**这些位是保留位。**

3、**给**/WR **施加一个** t_{WLWH_PFB} **的负脉冲。此命令在** RDY/BSY **引脚没有反馈。**

编程锁定位：

1、A：**加载命令**“0010 0000”

2、D：**加载数据的低位字节。**Bit n =“0”**代表要编程。**

Bit 2 = Lock Bit2

Bit 1 = Lock Bit1

Bit 7- 3, 0 = “1”。**这些位是保留位。**

4、 E：**写数据的低位字节**

锁定位只能在芯片擦除上时清除。

读熔断位和锁定位：

1、 A：**加载命令**“0000 0100”

2、 **设置**/OE **为**“0”，BS **为**“1”。 **此时可以从** DATA **总线读取数据。**

Bit 7 = Lock Bit1

Bit 6 = Lock Bit2

Bit 5 = SPIEN

Bit 0 = FSTRT

3、 **设置**/OE **为**“1”。

读厂标：

1、 A：**加载命令**“0000 1000”

C：**加载数据的低位字节**（$00~$02）。

设置/OE **为**“0”，BS **为**“0”。

2、 **设置**/OE **为**“1”。

并行编程特性

图 52　并行编程时序

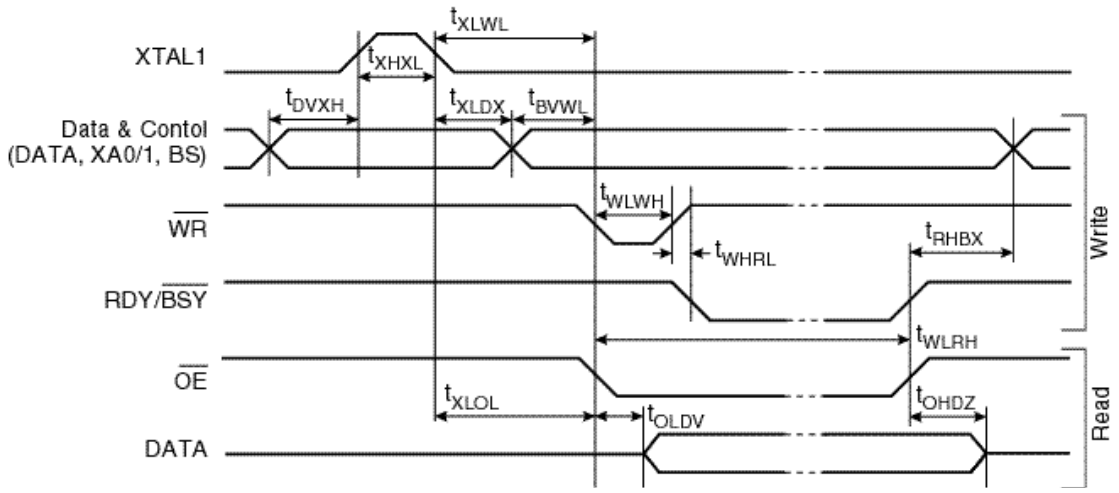


表 26　并行编程特性　T_A = 25℃±10%，V_{CC} = 5V±10%

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}	Data & Control Setup before XTAL1 High	67			ns
t_{XHXL}	XTAL2 脉宽	67			ns
t_{XLDX}	Data & Control Hold after XTAL1 Low	67			ns
t_{XLWL}	XTAL1 Low to /WR Low	67			ns
t_{BVWL}	BS Valid to /WR Low	67			ns
t_{RHBX}	BS Hold after RDY/BSY High	67			ns
t_{WLWH}	/WR Pluse Width Low(1)	67			ns
t_{WHRL}	/WR High to RDY/BSY Low(2)		20		ns
t_{WLRH}	/WR Low to RDY/BSY Gigh(2)	0.5	0.7	0.9	ms
t_{XLOL}	XTAL1 Low to /OE Low	67			ns
t_{OLDV}	/OE Low to DATA Valid		20		ns
t_{OHDZ}	/OE High to DATA Tri-stated			20	ns
t_{WLWH_CE}	/WR Pulse Width Low for Chip Erase	5	10	15	ms
t_{WLWH_PFB}	/WR Pulse Width Low for Programming the Fuse Bits	1.0	1.5	1.8	ms

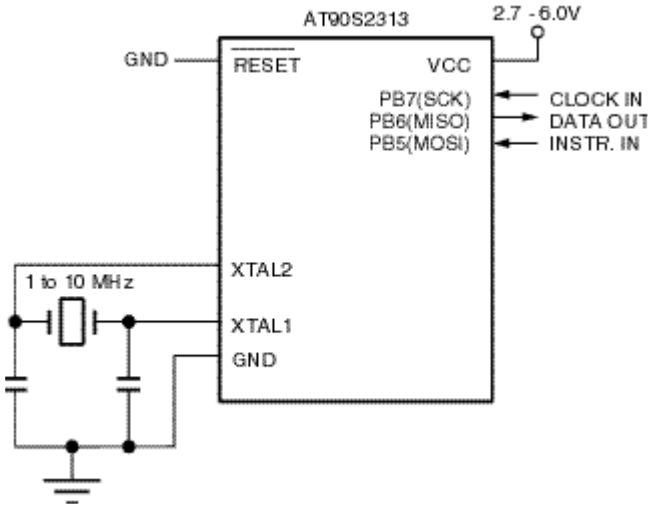
注：1、**在芯片擦除时使用** t_{WLWH_CE}，**而在编程锁定位是使用** t_{WLWH_PFB}。

2、**如果** t_{WLWH} **保持地比** t_{WLRH} **长，则看不见** RDY/BSY **脉冲。**

串行下载

当/RESET 拉到地时，FLASH 和 EEPROM 可以利用 SPI 总线进行串行下载。串行接口包括 SCK，MOSI 和 MISO。/RESET 拉低后，在进行编程/擦除之前首先要执行编程使能指令。

图 53　串行编程和校验



对于 EEPROM，由于其本身有自动擦除功能和自定时功能，因此更新时无需擦除。擦除指令将使 FLASH 和 EEPROM 的内容全部变为$FF。

FLASH 和 EEPROM 的地址是分离的。FLASH 的范围是$0000~$03FF，EEPROM 的范围是$000~$03F。

时钟可以从 XTAL1 引脚输入，或是将晶振接到 XTAL1 和 XTAL2。SCK 脉冲的最小高低电平时间为：

低：> 2 个 XTAL1 时钟

高：> 2 个 XTAL1 时钟

串行编程算法

进行串行编程时，数据在 SCK 的上升沿输入 AT90S2313，在 SCK 的下降沿输出。

编程算法如下：

1、上电过程：
   在/RESET 和 SCK 拉低的同时在 V_{CC} 和 GND 之间加上电源电压。
2、至少等待 20ms。 然后在 MOSI（PB5）串行输入编程使能指令。
3、如果通讯失步则串行编程将失败。如果同步，则在写编程使能命令第 3 个字节的时候器件会响应第二个字节（$53）。不论响应正确与否，4 字节指令必须发完。如果响应不是$53，则要产生一个 SCK 正脉冲并发送编程使能指令。如果发送编程使能指令 32 次都没有正确的响应就说明外部器件不存在或器件已坏。
4、如果此时执行了擦除指令，则须等待 t_{WD_ERASE}，然后在/RESET 上施加正脉冲，回到第二步。
5、FLASH 和 EEPROM 是一个字节一个字节编程的。发送完写指令后要等待 t_{WD_PROG} 的时间。对于擦除过的器件，数据$FF 就用不着再写了。
6、任意一个内存地址都可以用读指令在 MISO（PB6）读出。
7、编程结束后，可以把/RESET 拉高，进入正常工作模式。

8、下电过程（如果需要的话）：

将 XTAL1 拉低（如果没有用外部晶振，或者用的是内部 RC 振荡器）。

把/RESET 拉高。

关掉电源。

EEPROM 数据检测

写 EEPROM 时，如果内部的自擦除过程没有结束，读正在写的地址会得到 P1；自擦除过程结束后，器件则返回 P2（P1 和 P2 的定义见表 27）。当写过程结束后，读取的数据则为写入的数据。用这种方法可以确定何时可以写入新数据。但是对于特定的数据 P1 和 P2，就不可以用这种方法了。此时应当在编程新数据之前至少等待 t_{WD_PROG} 的时间。如果芯片在编程 EEPROM 之前已经进行过芯片擦除，则数据$FF 就可以不用再编程了。

表 27　EEPROM 数据检测返回值

型号	P1	P2
AT90S2313	$80	$7F

FLASH 数据检测：

写 FLASH 时，如果内部写过程没有结束，则读取正在写的地址时会得到返回值$7F，否则，读取结果为写入的数据。但是对于数据$7F，应当在编程新数据之前至少等待 t_{WD_PROG} 的时间。如果芯片在编程 FLASH 之前已经进行过芯片擦除，则数据$FF 就可以不用再编程了。

图 54　串行编程波形

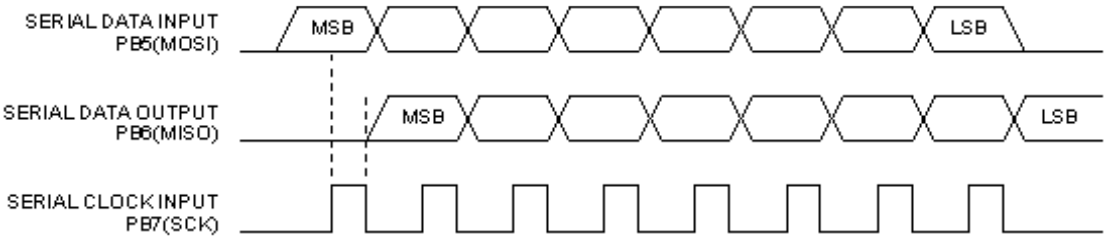


表 28　AT90S2313 的串行编程指令

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	/RESET 为低时使能串行编程
芯片擦除	1010 1100	100x xxxx	xxxx xxxx	xxxx xxxx	擦除 FLASH 和 EE
读 FLASH	0010 **H**000	0000 00**aa**	**bbbb bbbb**	**oooo oooo**	从字地址 a:b 读取 H（高或低）字节 o
写 FLASH	0100 **H**000	0000 00**aa**	**bbbb bbbb**	**iiii iiii**	写 H（高或低）字节 i 到字地址 a:b
读 EEPROM	1010 0000	0000 0000	0**bbb bbbb**	**oooo oooo**	从地址 b 读取数据 o
写 EEPROM	1100 0000	0000 0000	00**bb bbbb**	**iiii iiii**	写数据 i 到地址 b
写锁定位	1010 1100	111x x***21***x	xxxx xxxx	xxxx xxxx	写锁定位
读厂标	0011 0000	xxxx xxxx	xxxx xx**bb**	**oooo oooo**	从地址 b 读厂标 o(1)

注：　a = 地址高 Bit

b = 地址低 Bit

H = 0：低地址；1：高地址

o = 输出数据

i = 输入数据

x = 任意

1 = Lock Bit1

2 = Lock Bit2

注意：厂标不能在锁定模式 3 下读出。

串行编程电特性

图 55 串行编程时序

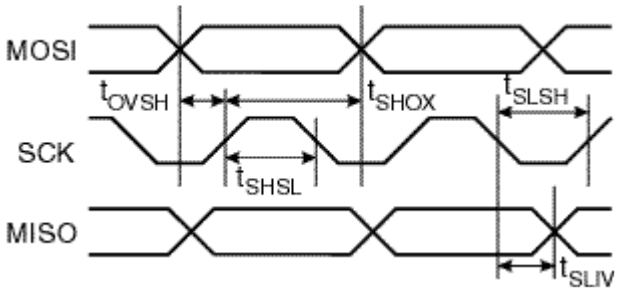


表 30 串行编程电特性，T_A = -40℃到 85℃，V_{CC} = 2.7V – 6.0V

符号	参数	最小值	典型值	最大值	单位
1/ t_{CLCL}	振荡频率（V_{CC} = 2.7V – 4.0V）	0		4	MHz
t_{CLCL}	振荡周期（V_{CC} = 2.7V – 4.0V）	250			ns
1/ t_{CLCL}	振荡频率（V_{CC} = 4.0V – 6.0V）	0		8	MHz
t_{CLCL}	振荡周期（V_{CC} = 4.0V – 6.0V）	125			ns
t_{SHSL}	SCK 高	2 t_{CLCL}			ns
t_{SLSH}	SCK 低	2t_{CLCL}			ns
t_{OVSH}	MOSI Setup to SCK High	t_{CLCL}			ns
t_{SHOX}	MOSI Hold after SCK High	2t_{CLCL}			ns
t_{SLIV}	SCK Low to MISO Valid	10	16	32	ns

表 21 擦除指令之后的最小等待时间

符号	3.2V	3.6V	4.0V	5.0V
t_{WD_ERASE}	18ms	14ms	12ms	8ms

表 22 写指令之后的最小延迟时间

符号	3.2V	3.6V	4.0V	5.0V
t_{WD_PROG}	9ms	7ms	6ms	4ms

直流特性

T_A = -40℃到 85℃，V_{CC} = 2.7V – 6.0V

符号	参数	条件	最小值	典型值	最大值	单位
V_{IL}	输入低电压	除了 XTAL1	-0.5		0.3 V_{CC} (1)	V
V_{IL1}	输入低电压	XTAL1	-0.5		0.1 (1)	V
V_{IH}	输入高电压	除了 XTAL1 和 /RESET	0.6 V_{CC} (2)		V_{CC} +0.5	V
V_{IH1}	输入高电压	XTAL1	0.7 V_{CC} (2)		V_{CC} +0.5	V
V_{IH2}	输入高电压	/RESET	0.85 V_{CC} (2)		V_{CC} +0.5	V
V_{OL}	输出低电压 (3) B、D 口	I_{OL} =20mA，V_{CC} = 5V			0.6 0.5	V

	[3] B、D 口	I_{OL} =10Ma，V_{CC} = 3V				
V_{OH}	输出高电压 [4] B、D 口	I_{OH} =20mA，V_{CC} = 5V I_{OH} =10mA，V_{CC} = 3V	4.3 2.3			V
I_{IL}	输入泄露电流 I/O 脚	V_{CC} = 6V，pin low			1.5	μA
I_{IH}	输入泄露电流 I/O 脚	V_{CC} = 6V，pin low			980	nA
RRST	复位上拉电阻		100		500	kΩ
$R_{I/O}$	I/O 口的上拉电阻		35		120	kΩ
I_{CC}	电源电流	工作状态，V_{CC} = 3V，4MHz			3.0	mA
		闲置状态，V_{CC} = 3V，4MHz			1.0	mA
I_{CC}	掉电模式 [5]	WDT 使能，V_{CC} = 3V		9	15.0	μA
		WDT 禁止，V_{CC} = 3V		<1	2.0	μA
V_{ACIO}	模拟比较器输入偏置电流	V_{CC} = 5V			40	mV
I_{ACLK}	模拟比较器输入泄露电流	V_{CC}=5V V_{IN} =V_{CC}/2	-50		50	nA
t_{ACPD}	模拟比较器传输延迟	V_{CC} = 2.7V V_{CC} = 4.0V		750 500		ns

注：

1、“最大值”代表保证可以“0”读取时的最高电压

2、“最小值”代表保证可以“1”读取时的最低电压

3、所有管脚的 I_{OL} 之和不能超过 200mA

D0 – D5 和 ZTAL2 的 I_{OL} 之和不能超过 100mA

B0 – B7 和 D6 的 I_{OL} 之和不能超过 100mA

3、所有管脚的 I_{OH} 之和不能超过 200mA

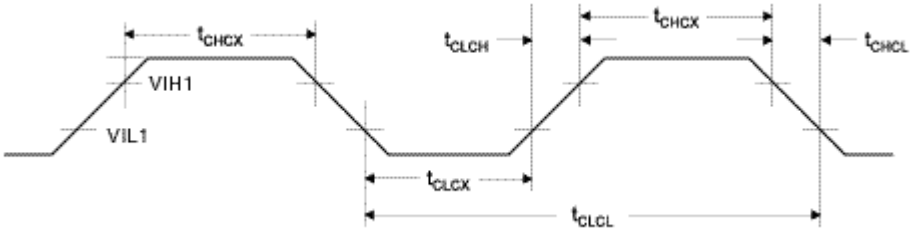
D0 – D5 和 ZTAL2 的 I_{OH} 之和不能超过 100mA

B0 – B7 和 D6 的 I_{OH} 之和不能超过 100mA

4、掉电时的最小 V_{CC} 为 2V

外部时钟驱动波形

图 56　外部时钟



外部时钟

符号	参数	V_{CC}=2.7V~4.0V		V_{CC}=4.0V~6.0V		单位
		最小值	最大值	最小值	最大值	
1/ t_{CLCL}	振荡频率	0	4	0	10	MHz
t_{CLCL}	时钟周期	250		100		ns
t_{CHCX}	高电平时间	100		40		ns
t_{CLCX}	低电平时间	100		40		ns
t_{CLCH}	上升时间		1.6		0.5	μs
t_{CHCL}	下降时间		1.6		0.5	μs

典型特性

后续图表表明了器件的典型特点。这些数据并没有进行 100%**的测试。功耗测量的条件为所有** I/O **引脚配置为输入（有上拉）。正弦波发生器作为时钟。**

掉电模式的功耗与时钟的选择无关。

器件功耗受以下因素影响：工作电压，工作频率，I/O **口的加载，**I/O **口变换频率，执行的代码以及工作温度。主要因素是工作电压和频率。**

容性负载的功耗可由公式 C_L*V_{CC}*f **进行计算。式中，**C_L **为负载电容，**V_{CC} =**工作电压，**f=I/O **引脚的平均开关频率。**

器件曲线标度高于测试的极限。使用时一定要按照订购器件的指标来使用。

工作于掉电模式时，看门狗使能及禁止两条曲线之差即表示了看门狗的功耗。

图 57　工作电流与频率的关系

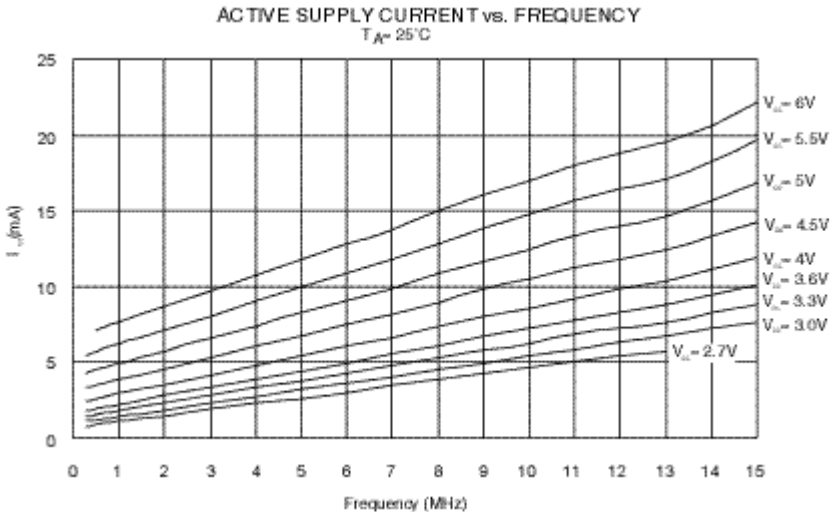


图 58　工作电流与电压的关系

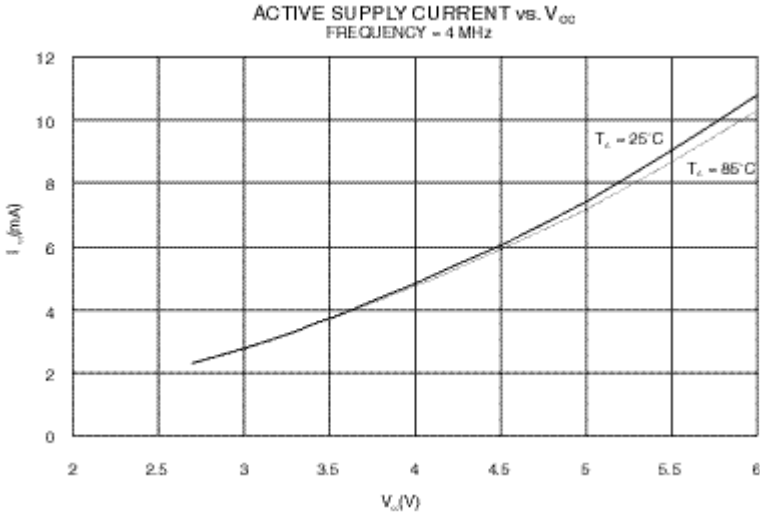


图 59　工作电流与频率的关系，闲置模式

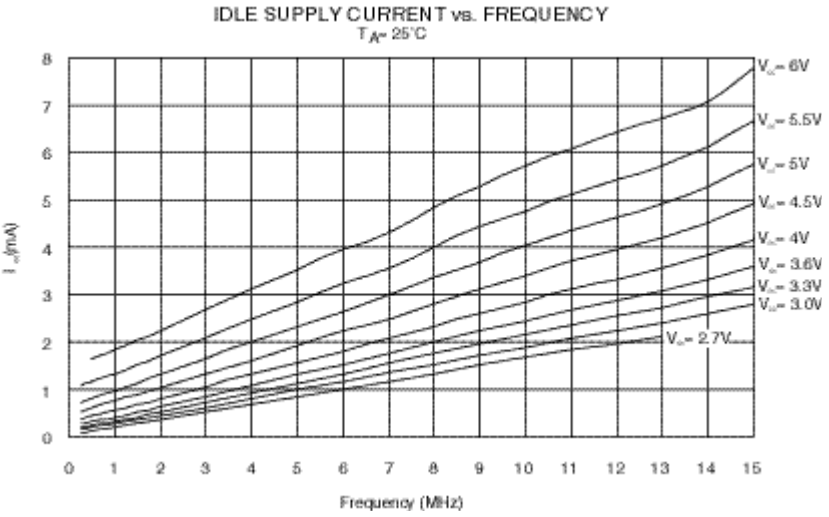


图 60　工作电流与电压的关系，闲置模式

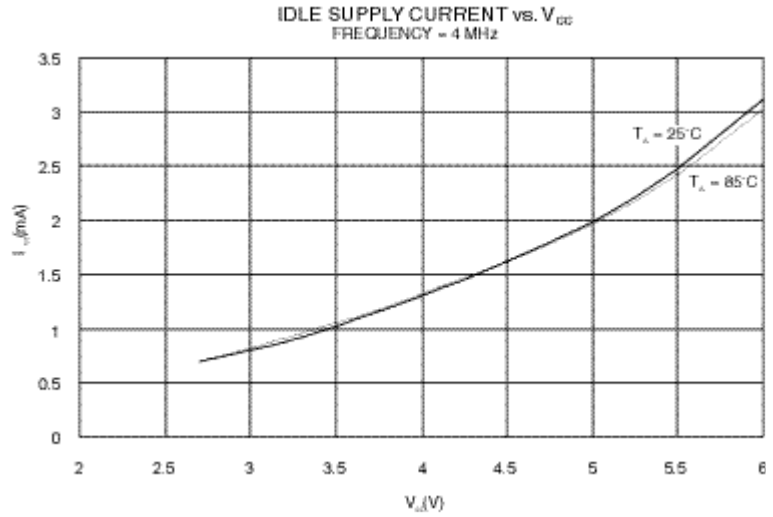


图 61　工作电流与电压的关系，掉电模式，看门狗禁止

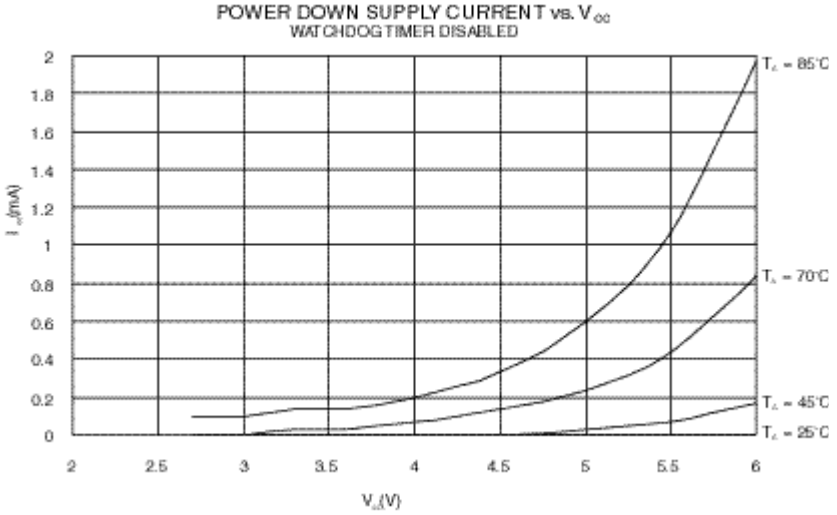


图 62　工作电流与电压的关系，掉电模式，看门狗使能

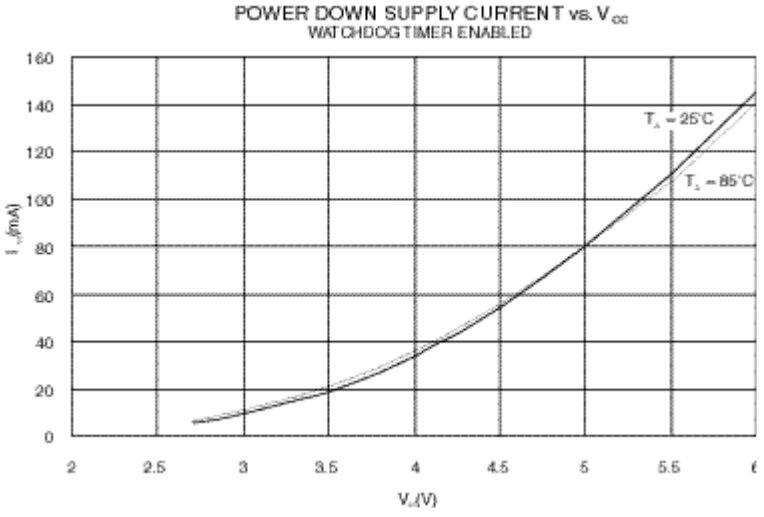


图 63　模拟比较器电流与电压的关系

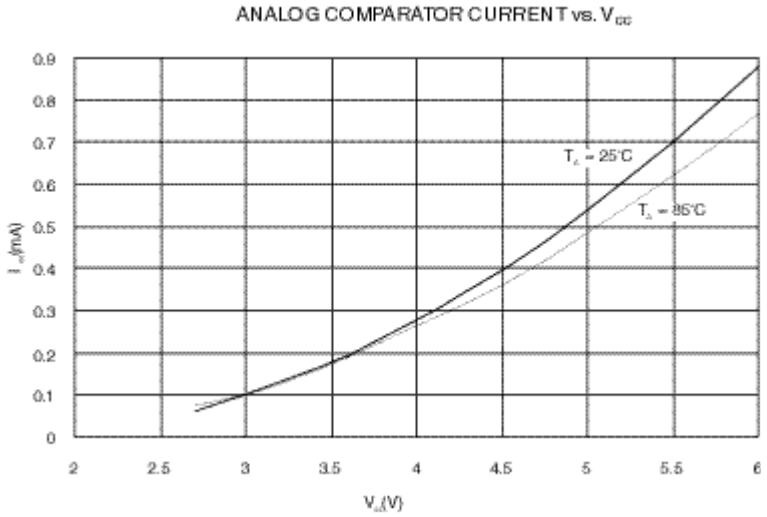


图 64　模拟比较器偏置电压与共模电压的关系

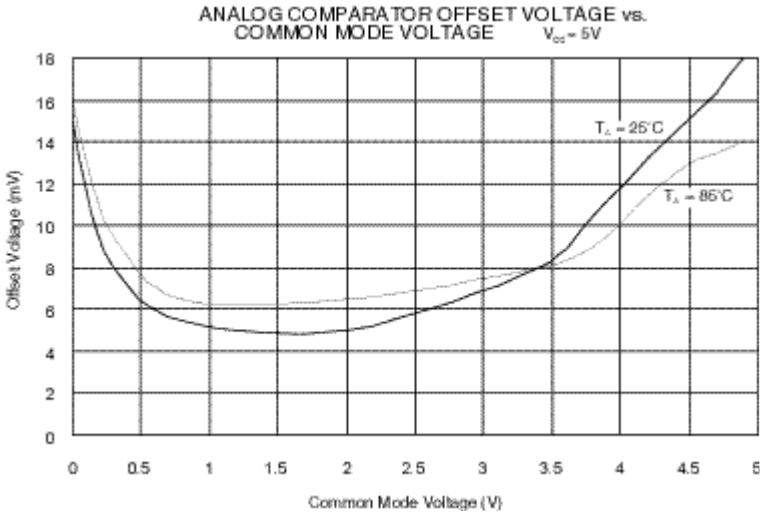


图 65　模拟比较器偏置电压与共模电压的关系

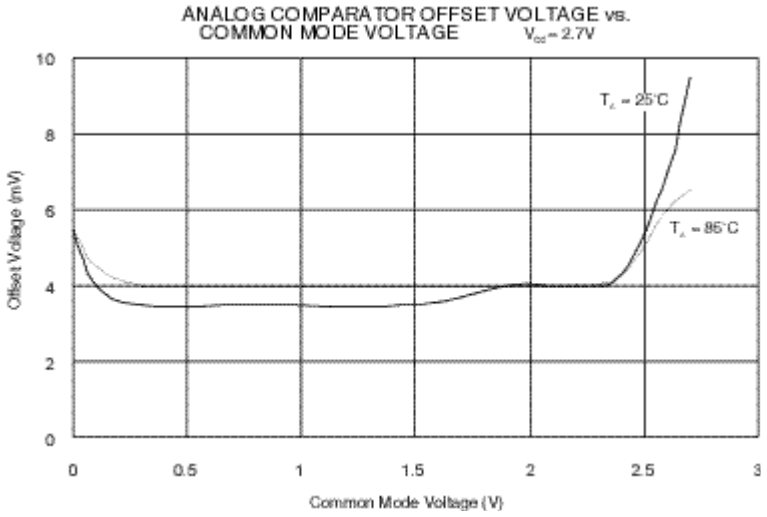


图 66　模拟比较器输入泄漏电流

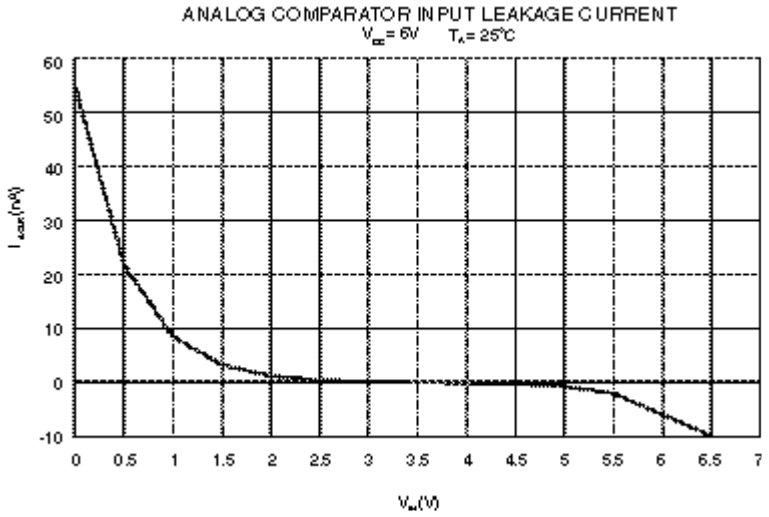


图 67　看门狗振荡频率与电压的关系

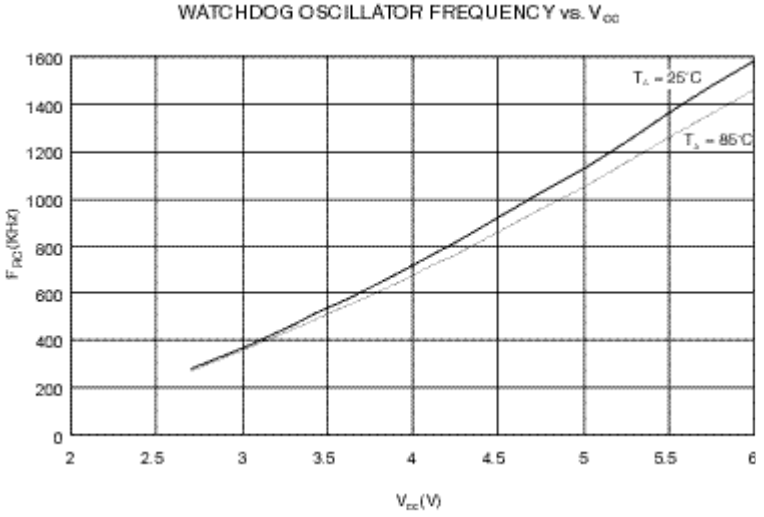


图 68　上拉电阻电流与输入电压的关系

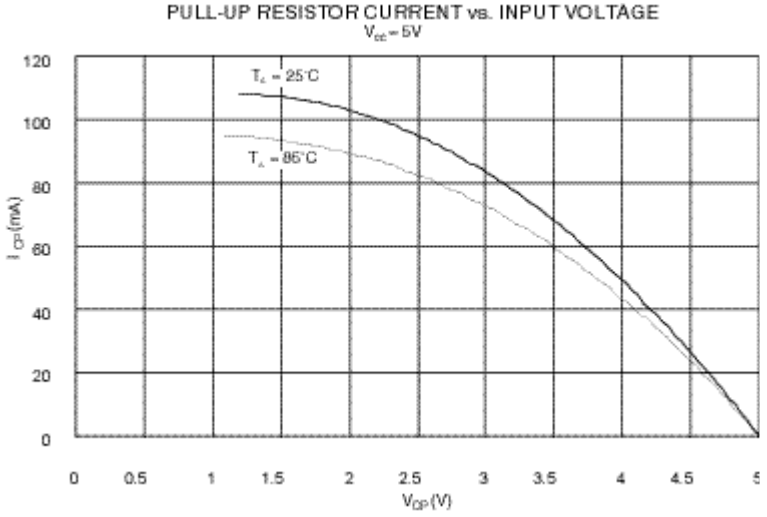


图 69　上拉电阻电流与输入电压的关系

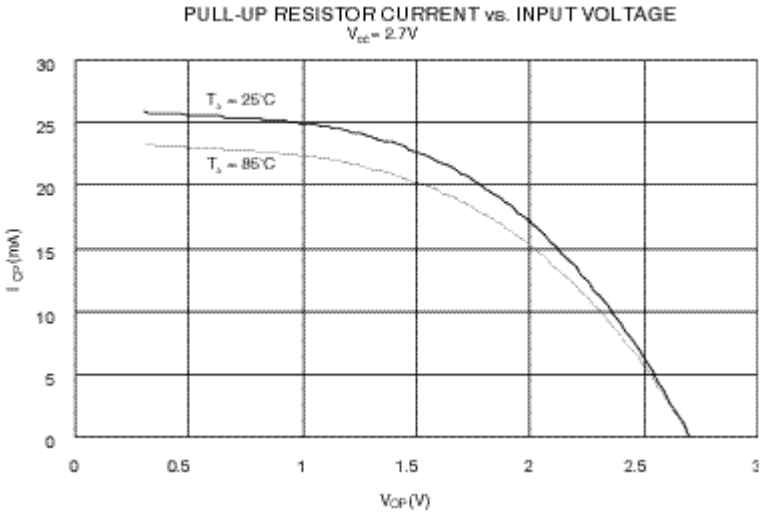


图 70　I/O 引脚吸入电流与输出电压的关系

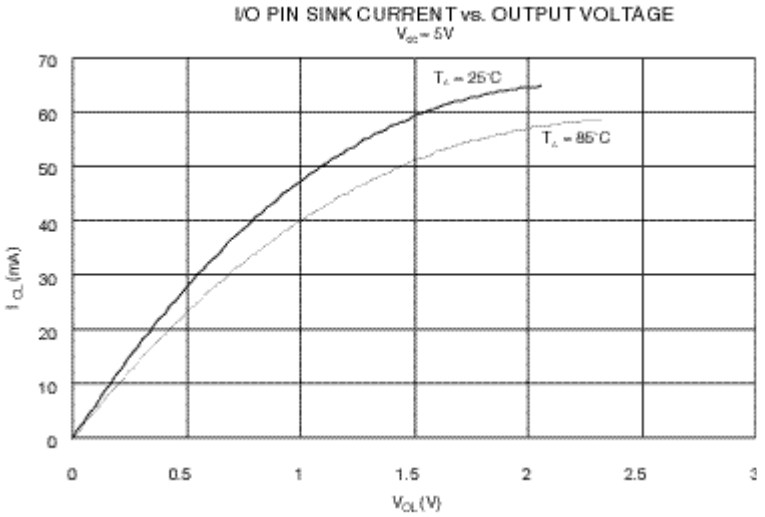


图 71　I/O 引脚输出电流与输出电压的关系

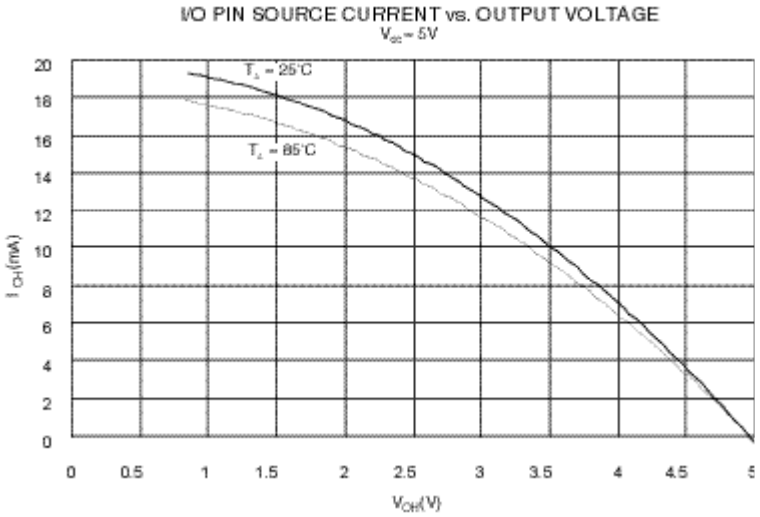


图 72　I/O 引脚吸入电流与输出电压的关系

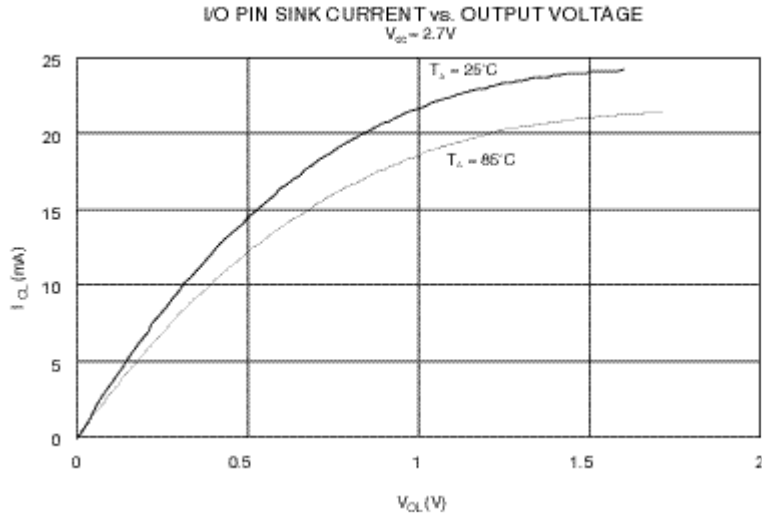


图 73　I/O 引脚输出电流与输出电压的关系

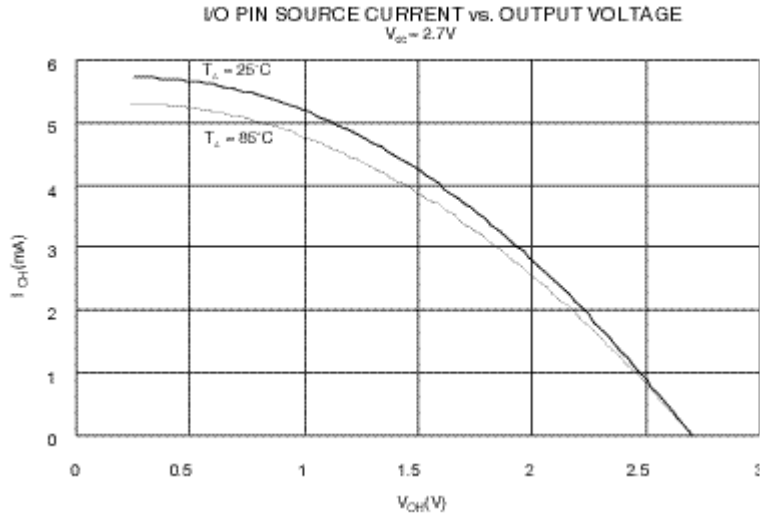


图 74　I/O 引脚输出门限电压与电压的关系

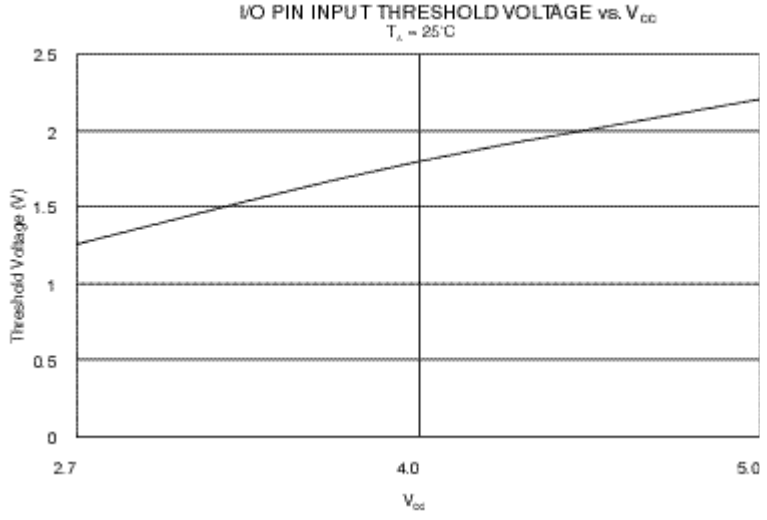
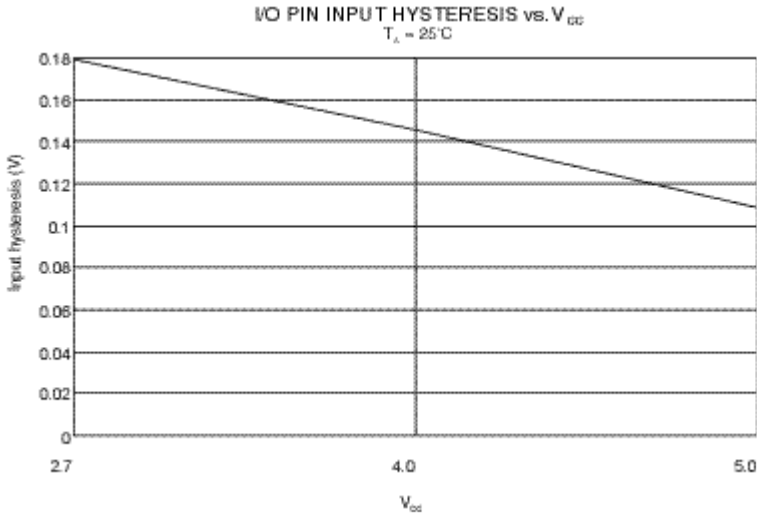


图 75　I/O 引脚输入容限与电压的关系



指　令

Rd——目的寄存器　　Rr——源寄存器

K——常数　　k——常数表示的地址

b——寄存器文件或 I/O 寄存器的位　　s——状态寄存器中的位

Z、Y、X——R31:R26

A——I/O 地址　　q——偏移量（6 位）

助记符	操作数	描述	操作	受影响的标志	时钟数
算术及逻辑操作					
ADD	Rd，Rr	不带进位位加	Rd ← Rd + Rr	Z,C,N,V,S,H	1
ADC	Rd，Rr	带进位位加	Rd ← Rd + Rr + C	Z,C,N,V,S,H	1
ADIW	Rd，K	给字加立即数	Rd+1:Rd← Rd+1:Rd+K	Z,C,N,V,S	2
SUB	Rd，Rr	不带进位位减	Rd ← Rd - Rr	Z,C,N,V,S,H	1
SUBI	Rd，K	减立即数	Rd ← Rd - K	Z,C,N,V,S,H	1
SBC	Rd，Rr	带进位位减	Rd ← Rd – Rr - C	Z,C,N,V,S,H	1
SBC I	Rd，K	带进位减立即数	Rd ← Rd – K - C	Z,C,N,V,S,H	1
SBIW	Rd，K	给字减立即数	Rd+1:Rd← Rd+1:Rd-K	Z,C,N,V,S	2
AND	Rd，Rr	逻辑与	Rd ← Rd ▪ Rr	Z, N,V,S	1
ANDI	Rd，K	与立即数	Rd ← Rd ▪ K	Z, N,V,S	1
OR	Rd，Rr	逻辑或	Rd ← Rd ∨ Rr	Z, N,V,S	1
ORI	Rd，K	或立即数	Rd ← Rd ∨ K	Z, N,V,S	1
EOR	Rd，Rr	异或	Rd ← Rd O± Rr	Z, N,V,S	1
COM	Rd	1	Rd ← $FF - Rd	Z,C,N,V,S	1
NEG	Rd	2 的补码	Rd ← $00 - Rd	Z,C,N,V,S,H	1
SBR	Rd，K	寄存器的位置位	Rd ← Rd ∨ K	Z, N,V,S	1
CBR	Rd，K	寄存器的位清零	Rd ← Rd ▪ （$FF - K）	Z, N,V,S	1
INC	Rd	加一	Rd ← Rd + 1	Z, N,V,S	1
DEC	Rd	减一	Rd ← Rd - 1	Z, N,V,S	1
TST	Rd	零和负测试	Rd ← Rd ▪ Rd	Z, N,V,S	1
CLR	Rd	清除寄存器	Rd ← Rd O± Rd	Z, N,V,S	1
SER	Rd	置位寄存器	Rd ← $FF	-	1

跳转指令					
RJMP	k	相对跳转	PC ← PC + k + 1	-	2
IJMP		间接跳转（Z）	PC(15:0)←Z, PC(21:16)←0	-	2
JMP	k	绝对跳转	PC ← k	-	3
RCALL	k	相对调用	PC ← PC + k + 1	-	3/4
ICALL		间接调用（Z）	PC(15:0)←Z, PC(21:16)←0	-	3/4
RET		调用返回	PC ← 堆栈	-	4/5
RETI		中断返回	PC ← 堆栈	I	4/5
CPSE	Rd，Rr	比较，相等则跳	若（Rd=Rr）PC←PC+2 或 3	-	1/2/3
CP	Rd，Rr	比较	Rd - Rr	Z,C,N,V,S,H	1
CPC	Rd，Rr	带进位位比较	Rd – Rr - C	Z,C,N,V,S,H	1
CPI	Rd，K	与立即数比较	Rd – K	Z,C,N,V,S,H	1
SBRC	Rd，b	寄存器位清零即跳	若（Rd（b） = 0）PC←PC+2 或 3	-	1/2/3
SBRS	Rd，b	寄存器位置位即跳	若（I/O（A，b） = 1）PC←PC+2 或 3	-	1/2/3
SBIC	A，b	I/O 寄存器位清零即跳	若（I/O（A，b） = 0）PC←PC+2 或 3	-	1/2/3
SBIS	A，b	I/O 寄存器位置位即跳	若（Rd（b） = 1）PC←PC+2 或 3	-	1/2/3
BRBS	s，k	状态标志置位跳	若（SREG（s）=1）PC←PC+k+1	-	1/2
BRBC	s，k	状态标志清零跳	若（SREG（s）=0）PC←PC+k+1	-	1/2
BREQ	k	相等即跳	若（Z=1）PC←PC+k+1	-	1/2
BRNE	k	不等即跳	若（Z=0）PC←PC+k+1	-	1/2
BRCS	k	进位即跳	若（C=1）PC←PC+k+1	-	1/2
BRCC	k	没有进位即跳	若（C=0）PC←PC+k+1	-	1/2
BRSH	k	大于等于即跳（无符号）	若（C=0）PC←PC+k+1	-	1/2
BRLO	k	小于即跳（无符号）	若（C=1）PC←PC+k+1	-	1/2
BRMI	k	负即跳	若（N=1）PC←PC+k+1	-	1/2
BRPL	k	正即跳	若（N=0）PC←PC+k+1	-	1/2
BRGE	k	大于等于即跳（有符号）	若 （ NO+V=1 ）PC←PC+k+1	-	1/2
BRLT	k	小于即跳（有符号）	若 （ NO+V=0 ）PC←PC+k+1	-	1/2
BRHS	k	H 位置位即跳	若（H=1）PC←PC+k+1	-	1/2
BRHC	k	H 位清零即跳	若（H=0）PC←PC+k+1	-	1/2
BRTS	k	T 置位即跳	若（T=1）PC←PC+k+1	-	1/2
BRTC	k	T 清零即跳	若（T=0）PC←PC+k+1	-	1/2
BRVS	k	V 置位即跳	若（V=1）PC←PC+k+1	-	1/2
BRVC	k	V 清零即跳	若（V=0）PC←PC+k+1	-	1/2
BRIE	k	中断使能即跳	若（I=1）PC←PC+k+1	-	1/2
BRID	k	中断禁止即跳	若（I=0）PC←PC+k+1	-	1/2
数据传输指令					

MOV	Rd，Rr	**拷贝寄存器**	Rd ← Rr	-	1
LDI	Rd，**K**	**取立即数**	Rd ← **K**	-	1
LDS	Rd，k	**从数据区取数**	Rd ←（k）	-	2
LD	Rd，X	**间接取数**	Rd ←（X）	-	2
LD	Rd，X+	**间接取数，后加**	Rd ←（X），X←X+1	-	2
LD	Rd，-X	**间接取数，先减**	X←X-1，Rd ←（X）	-	2
LD	Rd，Y	**间接取数**	Rd ←（Y）	-	2
LD	Rd，Y+	**间接取数，后加**	Rd ←（Y），Y←Y+1	-	2
LD	Rd，-Y	**间接取数，先减**	Y←Y-1，Rd ←（Y）	-	2
LDD	Rd,Y+q	**带偏移间接取数**	Rd ←（Y + q）	-	2
LD	Rd，Z	**间接取数**	Rd ←（Z）	-	2
LD	Rd，Z+	**间接取数，后加**	Rd ←（Z），Z←Z+1	-	2
LD	Rd，-Z	**间接取数，先减**	Z←Z-1，Rd ←（Z）	-	2
LDD	Rd,Z+q	**带偏移间接取数**	Rd ←（Z + q）	-	2
STS	k ，Rr	**存数于数据区**	（k）← Rr	-	2
ST	X，Rr	**间接存数**	（X）← Rr	-	2
ST	X+，Rr	**间接存数，后加**	（X）← Rr，X←X+1	-	2
ST	X-，Rr	**间接存数，先减**	X←X-1，（X）← Rr	-	2
ST	Y，Rr	**间接存数**	（Y）← Rr	-	2
ST	Y+，Rr	**间接存数，后加**	（Y）← Rr，Y←Y+1	-	2
ST	Y-，Rr	**间接存数，先减**	Y←Y-1，（Y）← Rr	-	2
STD	Y+q,Rr	**带偏移间接存数**	（Y + q）← Rr	-	2
ST	Z，Rr	**间接存数**	（Z）← Rr	-	2
ST	Z+，Rr	**间接存数，后加**	（Z）← Rr，Z←Z+1	-	2
ST	Z-，Rr	**间接存数，先减**	Z←Z-1，（Z）← Rr	-	2
STD	Z+q,Rr	**带偏移间接存数**	（Z + q）← Rr	-	2
LPM		**在程序区取数**	R0 ←（Z）	-	3
IN	Rd，A	**从** I/O **取数**	Rd ← I/O（A）	-	1
OUT	A，Rr	**存数于** I/O	I/O（A）← Rr	-	1
PUSH	Rr	**入栈**	STACK ← Rr	-	2
POP	Rd	**出栈**	Rd ← STACK	-	2
位及位测试指令					
LSL	Rd	**逻辑左移**	Rd(n + 1) ← Rd(n)，Rd(n)←0，C←Rd(7)	Z,C,N,V,H	1
LSR	Rd	**逻辑右移**	Rd(n) ← Rd(n + 1)，Rd(7)←0，C←Rd(0)	Z,C,N,V	1
ROL	Rd	**带进位位左移**	Rd(0)←C，Rd(n + 1) ← Rd(n)，C←Rd(7)	Z,C,N,V,H	1
ROR	Rd	**带进位位右移**	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=6..0	Z,C,N,V	1
SWAP	Rd	**高低位交换**	Rd(3..0) ←→ Rd(7..4)	-	1
BSET	s	**设置标志**	SREG（s）← 1	SREG（s）	1
BCLR	s	**清除标志**	SREG（s）← 0	SREG（s）	1
SBI	A，b	**设置** I/O **寄存器的位**	I/O（A，b）← 1	-	2
CBI	A，b	**清除** I/O **寄存器的位**	I/O（A，b）← 0	-	2

BST	Rr，b	Rr 的 b 位到 T	T ← Rr（b）	T	1
BLD	Rd，b	T 到 Rr 的 b 位	Rr（b）← T	-	1
SEC		置位 C	C ← 1	C	1
CLC		清零 C	C ← 0	C	1
STN		置位 N	N ← 1	N	1
CLN		清零 N	N ← 0	N	1
SEZ		置位 Z	Z ← 1	Z	1
CLZ		清零 Z	Z ← 0	Z	1
SEI		置位 I	I ← 1	I	1
CLI		清零 I	I ← 0	I	1
SES		置位 S	S ← 1	S	1
CLS		清零 S	S ← 0	S	1
SEV		置位 V	V ← 1	V	1
CLV		清零 V	V ← 0	V	1
SET		置位 T	T ← 1	T	1
CLT		清零 T	T ← 0	T	1
SHE		置位 H	H ← 1	H	1
CLH		清零 H	H ← 0	H	1
NOP		空操作		-	1
SLEEP		休眠指令		-	3
WDR		看门狗复位		-	1

定货信息

速度（MHz）	电源（V）	定货号	封装	工作范围
4	2.7 – 6.0	AT90S2313 – 4PC AT90S2313 – 4SC	20P3 20S	商用 （0℃ - 70℃）
		AT90S2313 – 4PI AT90S2313 – 4SI	20P3 20S	工业 （-40℃ - 85℃）
10	4.0 – 6.0	AT90S2313 – 10PC AT90S2313 – 10SC	20P3 20S	商用 （0℃ - 70℃）
		AT90S2313 – 10PI AT90S2313 – 10SI	20P3 20S	工业 （-40℃ - 85℃）

封装类型	
20P3	20 脚，0.300’’，塑料双列直插（PDIP）
20S	20 脚，0.300’’，塑料 Gull-Wing 小尺寸（SOIC）

开发过程及所需工具

汇编软件

AVR ASM（免费软件，可从www.atmel.com或WWW.SL.COM.CN下载）

仿真器

AVR ICE（STDPOD 或 ADCPOD），或 ICE 200。调试软件为 AVR STUDIO（免费软件，可从www.atmel.com或WWW.SL.COM.CN下载）

下载器

START KIT 200 或第三方厂商（如：广州天河双龙电子有限公司 SL-AVR,LT-48.）