AT90S2323/2343

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
   —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
5. 特别的 MCU 特点
   —低功耗空闲和掉电模式
   —内外部中断源
   —上电复位电路
   —可选择的片内 RC 振荡器
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：2.4mA
   —空闲模式：0.5mA
   —掉电模式：<1μA
8. I/O 和封装
   —3 个可编程的 I/O 脚（AT90S/LS2323）
   —5 个可编程的 I/O 脚（AT90S/LS2343）
   —8 脚 PDIP 和 SOIC 封装
9. 工作电压
   —2.7V-6.0V（AT90LS2323/AT90LS2343）
   —4.0V-6.0V（AT90S2323/AT90S2343）
10. 速度
    —0-4MHz（AT90LS2323/2343/AT90LS2343）
    —0-10MHz（AT90S2323/2343/AT90S2343）

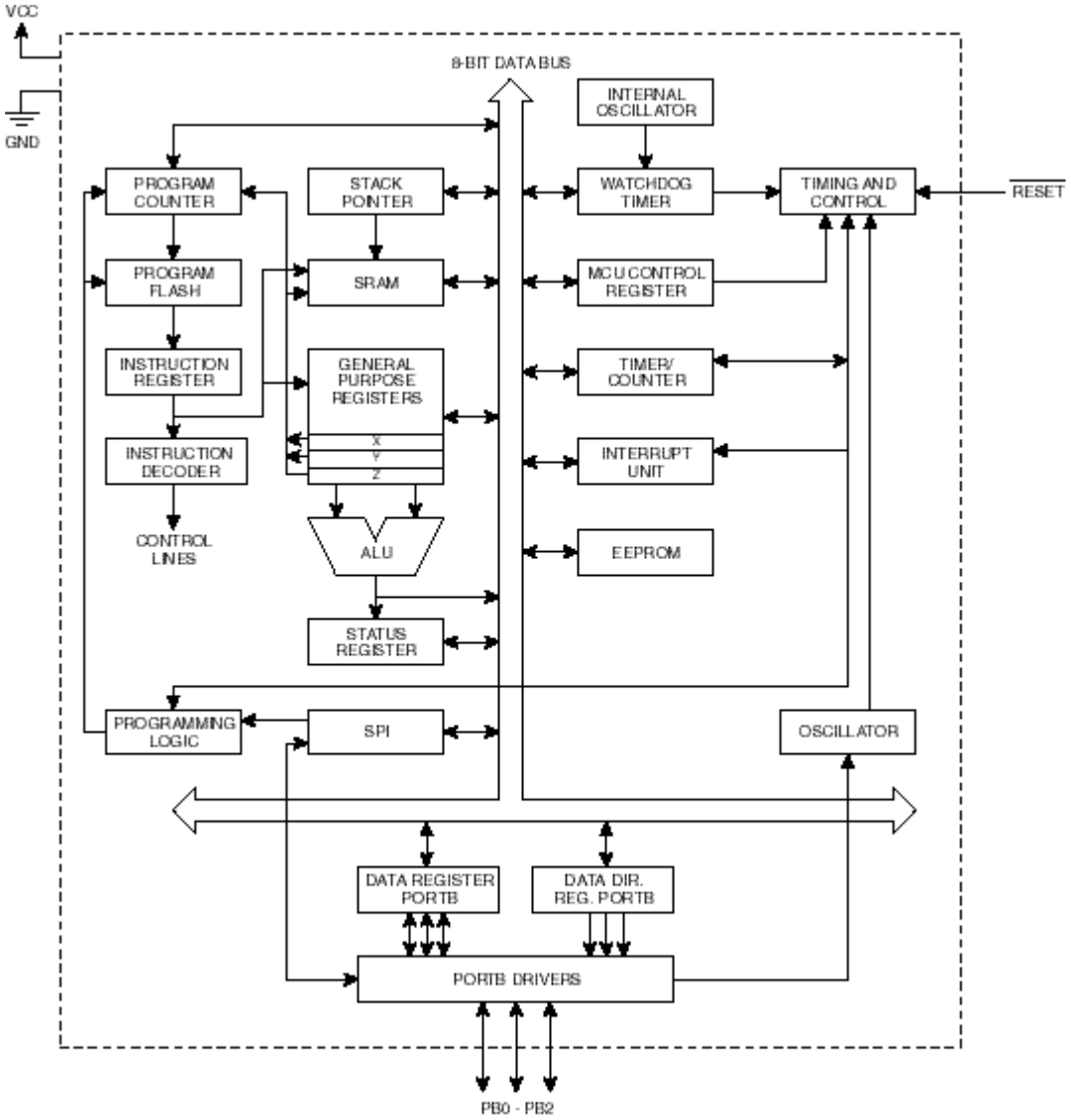
描述

AT90S2323/2343 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，AT90S2323/2343 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　AT90S/LS2343 结构方框图

图 2　AT90S/LS2323 结构方框图



AT90S2323/2343 具有以下特点：2K 字节 FLASH，128 字节 EEPROM，128 字节 SRAM，3（AT90S/LS2323）/5（AT90S/LS2343）个通用 I/0 口，32 个通用工作寄存器，8 位定时器/计数器，内外中断源，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，2323/2343 为许多嵌入式控制应用提供了灵活而低成本的方案。

AT90S2323/2343 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。

AT90S/LS2323 和 AT90S/LS2343 的比较

AT90S/LS2323 适用于外接晶振或陶瓷振荡器。起动时间可选 1ms（晶振）或 16ms（陶瓷振荡器）。有 3 个 I/O 口。

AT90S/LS2343 适用于外接振荡源或选择片内振荡器作为时钟。有 5 个 I/O 口。

表 1　差异表

型号	AT90S/LS2323	AT90S/LS2343
片内振荡器放大器	有	无
内部 RC 时钟	无	有
PB3 可作为 I/O	不行	内部振荡模式时
PB4 可作为 I/O	不行	是
起动时间	1ms/16ms	固定的 16μs

管脚配置

AT90S/LS2323 的管脚定义

VCC、GND：电源

B 口（PB2..PB0）：

B 口是一个 3 位双向 I/O 口，每一个管脚都有内部上拉电阻（可单独选择）。在复位过程中，B 口为三态，即使此时时钟还未起振。

/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。

AT90S/LS2343 的管脚定义

VCC、GND：电源

B 口（PB4..PB0）：

B 口是一个 5 位双向 I/O 口，每一个管脚都有内部上拉电阻（可单独选择）。当选用外部时钟源时，PB3 作为时钟输入端。在复位过程中，B 口为三态，即使此时时钟还未起振。

/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。

CLOCK： 外部时钟模式时的时钟信号输入。

时钟选择

晶振

AT90S/LS2323 具有一个反向放大器，可用作为片内振荡器，如图 3 所示。如果不用外部时钟源的话，推荐使用 AT90S/LS2343，以得到更多的 I/O 脚。

图 3　振荡器连接

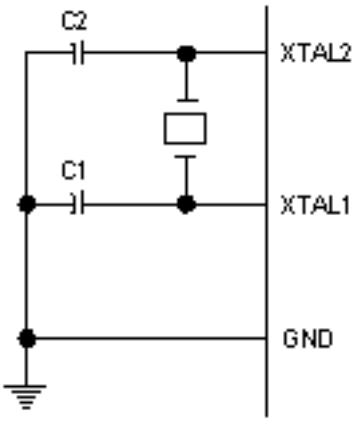
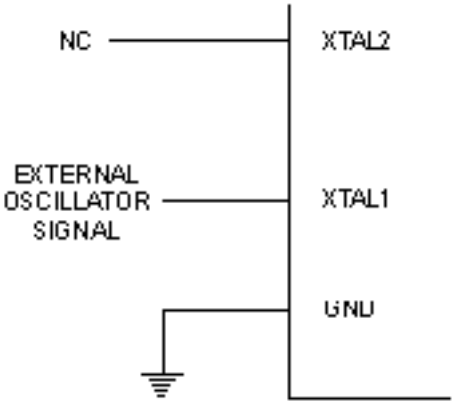
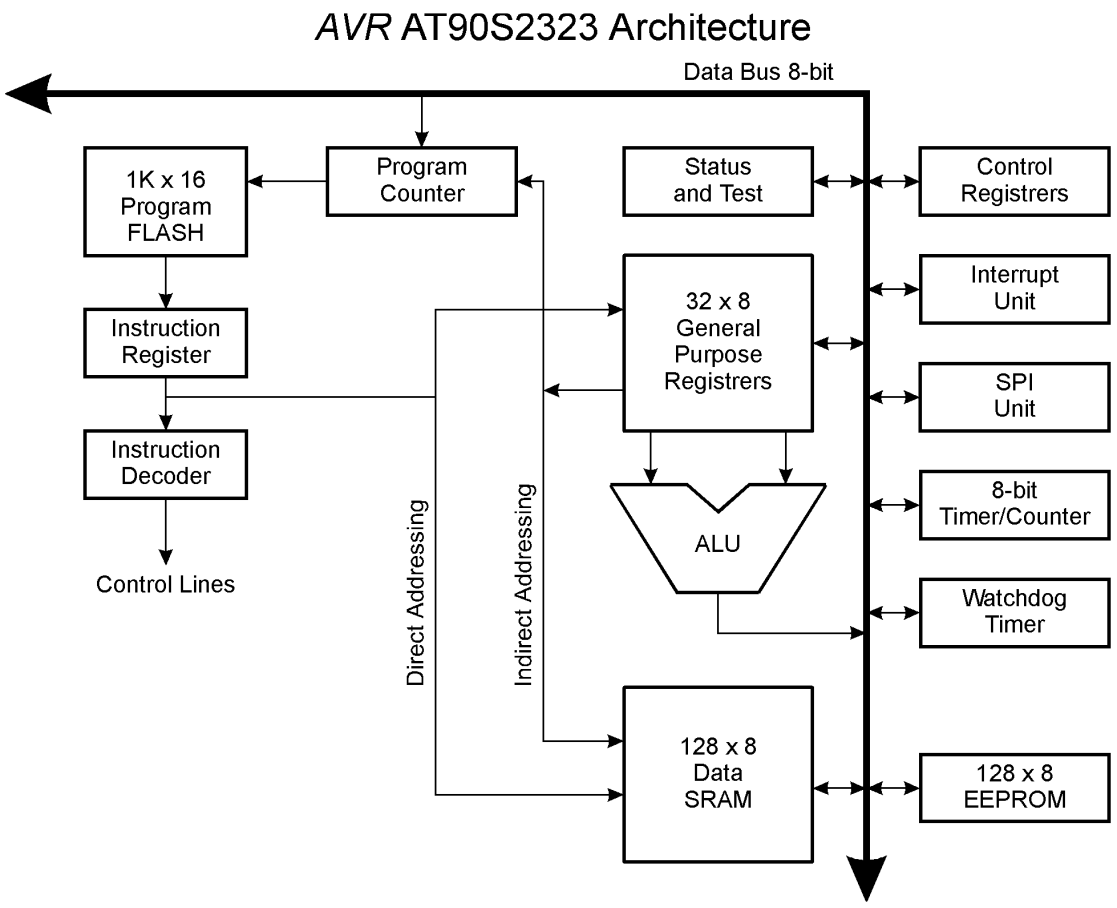


图 4　外部时钟驱动配置



结构纵览

图 5　AT90S2323/2343 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 AT90S2323/2343 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。

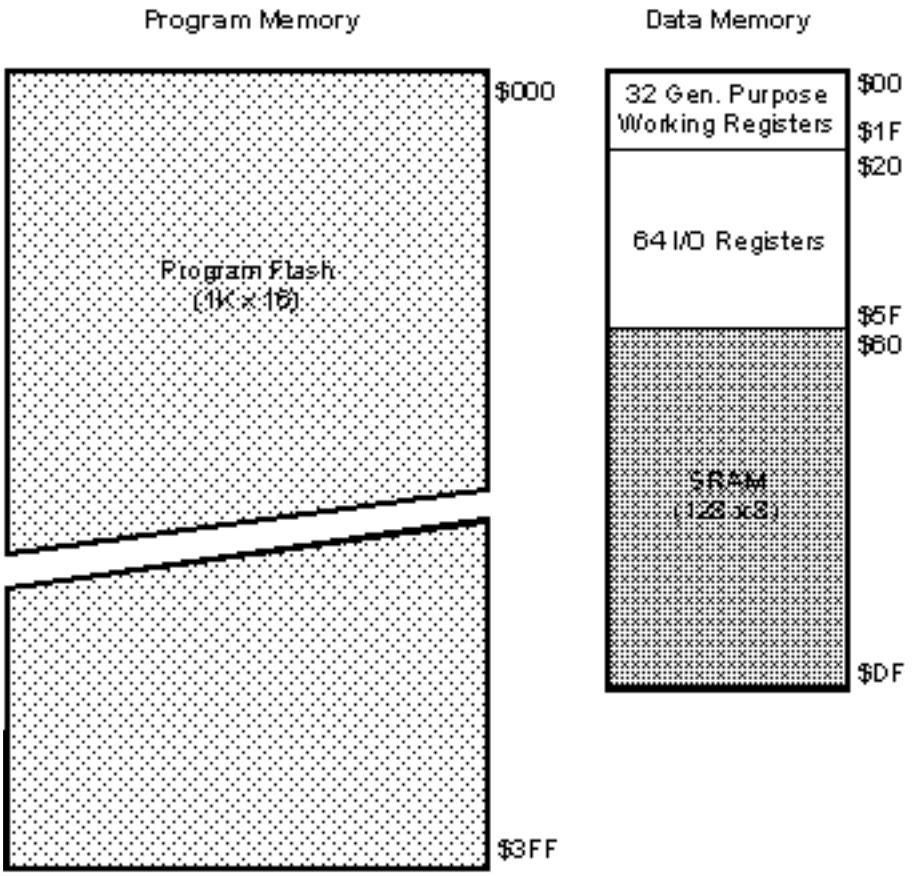
AVR 采用了 HARVARD 结构：程序和数据总线分离。程序内存通过两段式的管道（Pipeline）进行访问：当 CPU 在执行一条指令的同时，就去取下一条指令。这种预取指的概念使得指令可以在一个时钟完成。

相对跳转和相对调用指令可以直接访问 1K 地址空间。大多数的 AVR 指令为 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 6　内存映像



EEPROM数据存储区

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 7　通用工作寄存器

	7	0	地址	
	R0		$00	
	R1		$01	
	R2		$02	
	…			
	R13		$0D	
	R14		$0E	
通	R15		$0F	
用	R16		$10	
工	R17		$11	
作	…			
寄	R26		$1A	X 寄存器低字节
存	R27		$1B	X 寄存器高字节

器			
	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

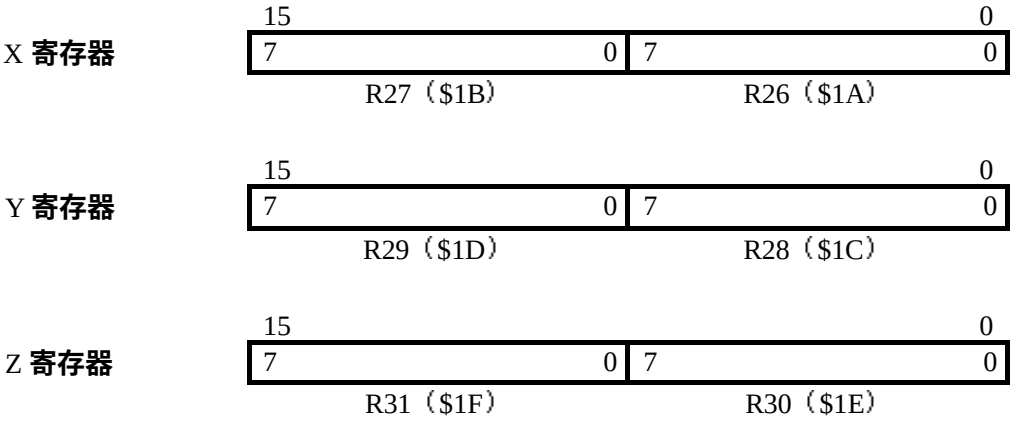
所有的寄存器操作指令都可以单指令的形式直接访问所有的寄存器。例外情况为 5 条涉及常数操作的指令：SBCI、SUBI、CPI、ANDI 和 ORI。这些指令只能访问通用寄存器文件的后半部分：R16 到 R31。

如图 7 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 8 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

AT90S2323/2343 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。

AT90S2323/2343 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

AT90S2323/2343 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 AT90S2323/2343 的数据组织方式。

图 9 SRAM 分布

寄存器文件		数据地址空间
R0		$00
R1		$01

R2
…
R29
R30
R31

I/O 寄存器

$00
$01
$02
…
$3D
$3E
$3F

$02
…
$1D
$1E
$1F

数据地址空间

$20
$21
$22
…
$5D
$5E
$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。

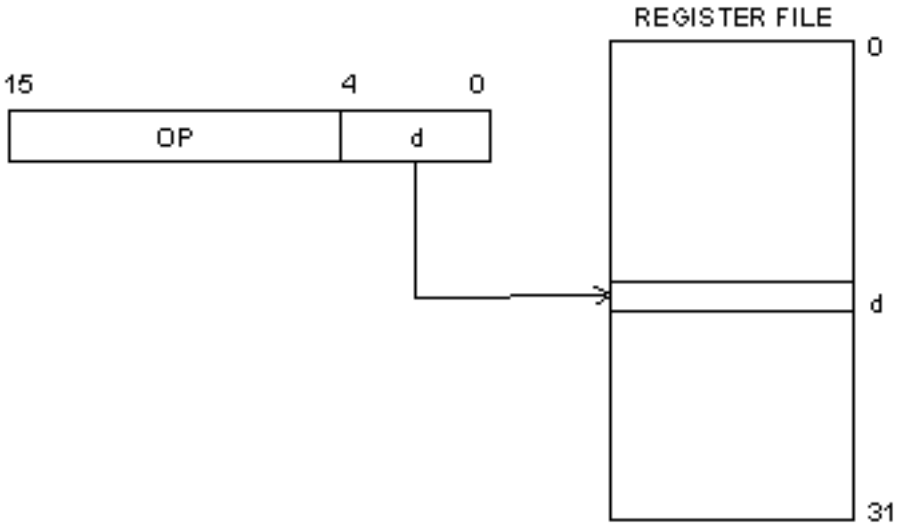
32 个通用寄存器，64 个 I/O 寄存器和 128 字节的 SRAM 可以被所有的寻址模式访问。

程序和数据寻址模式

AT90S2323/2343 支持强大而有效的寻址模式。本节将要介绍各种寻址模式。

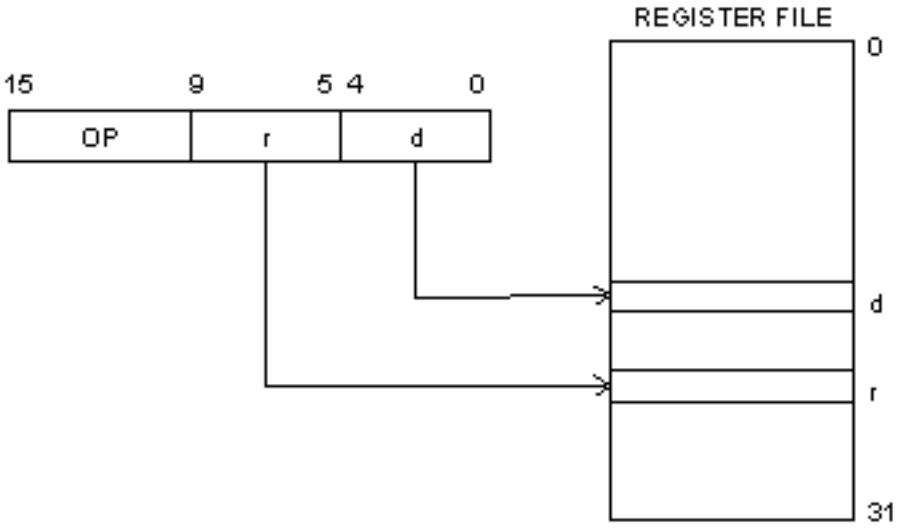
单寄存器直接寻址：

图 10



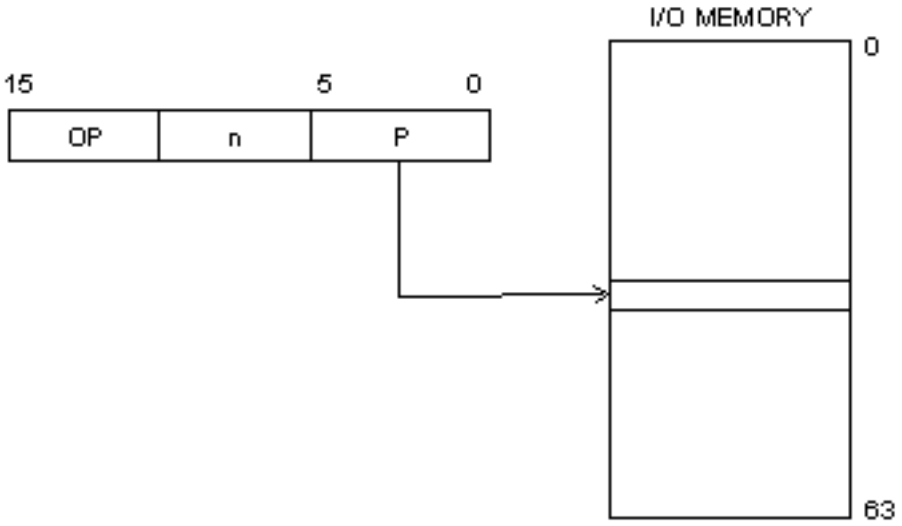
双寄存器直接寻址：

图 11



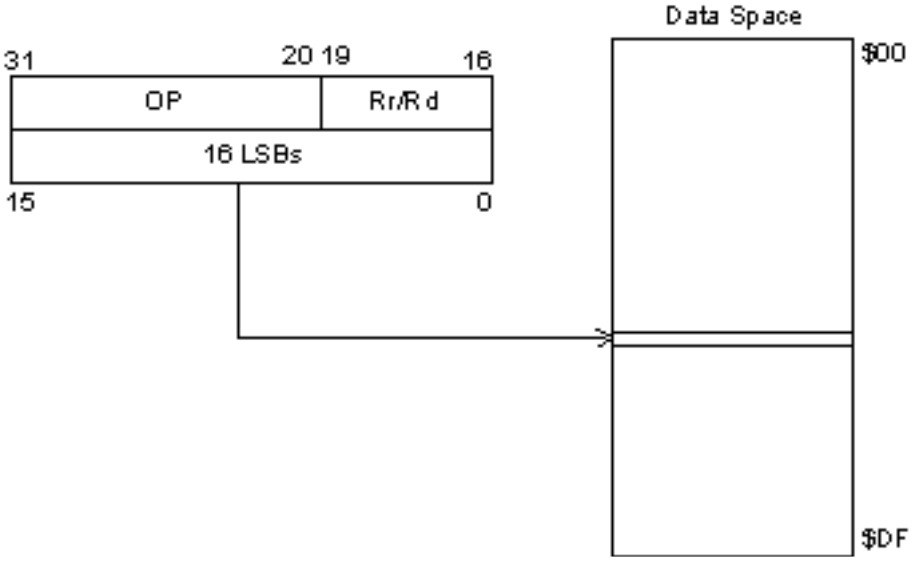
I/O 直接寻址：

图 12



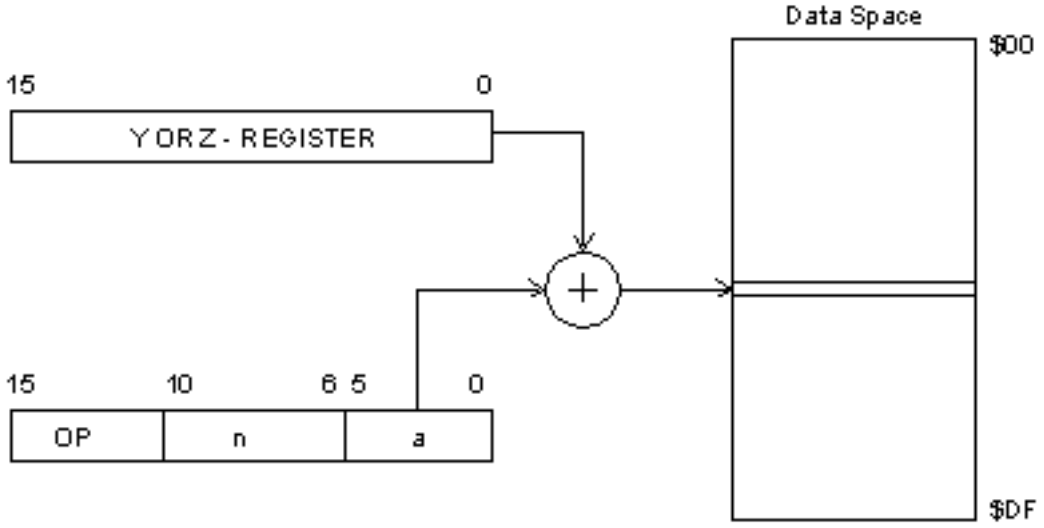
数据直接寻址：

图 13



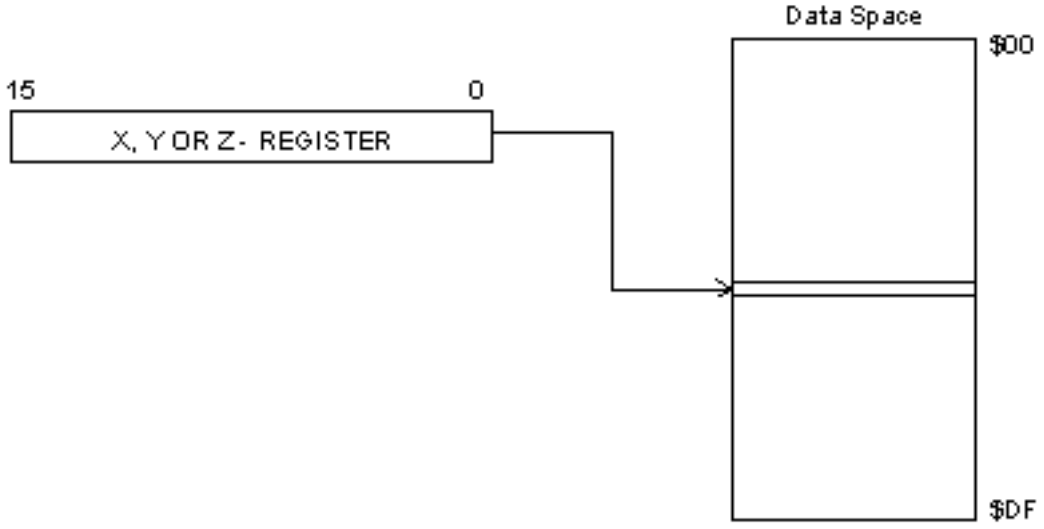
带偏移的数据间接寻址：

图 14



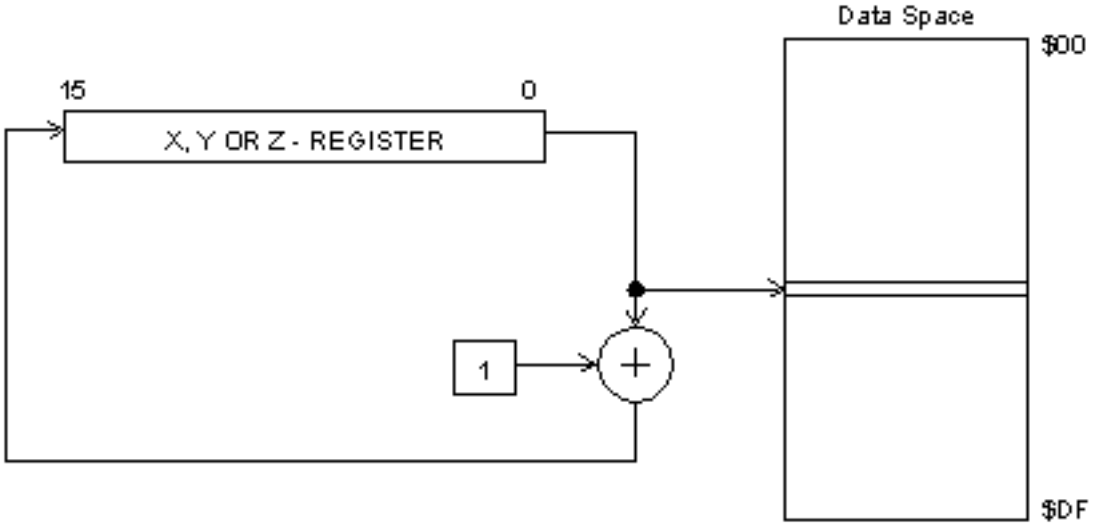
数据间接寻址：

图 15



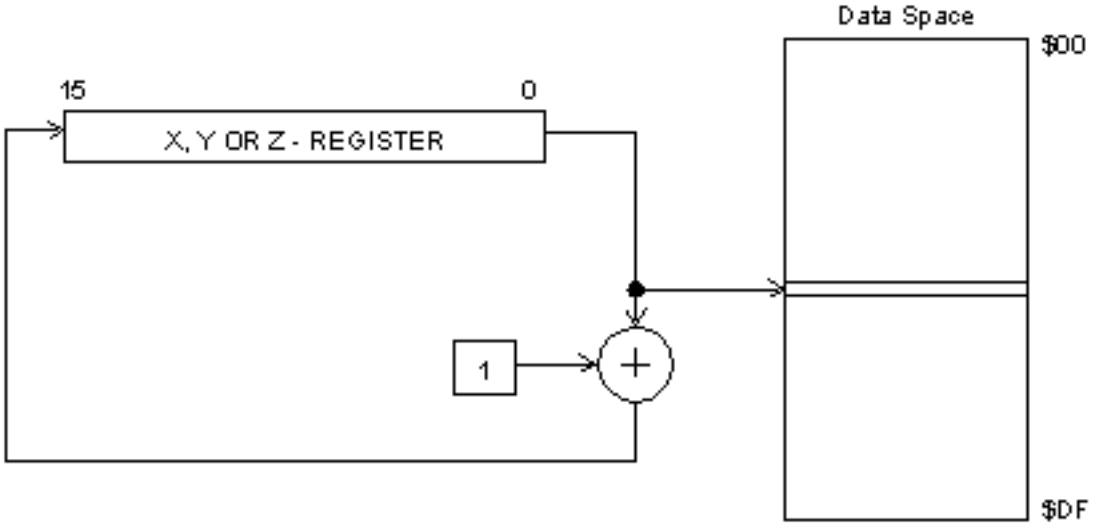
带预减的数据间接寻址：

图 16



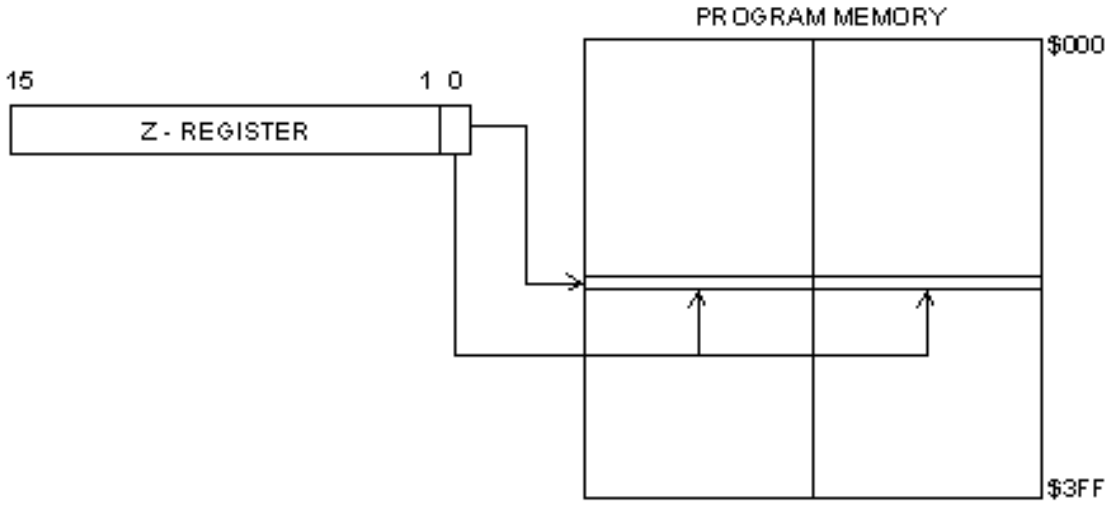
带后加的数据间接寻址：

图 17



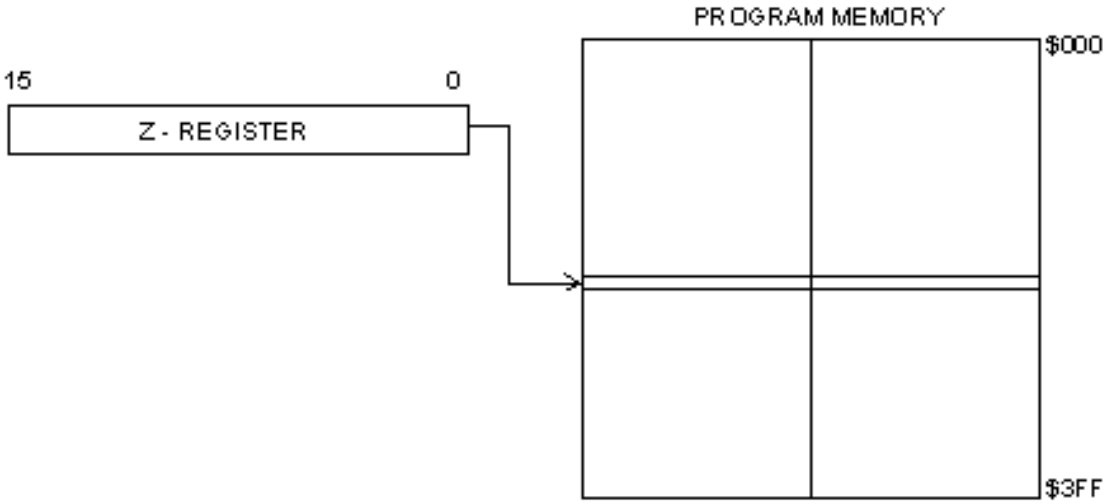
使用 LPM指令寻址常数：

图 18



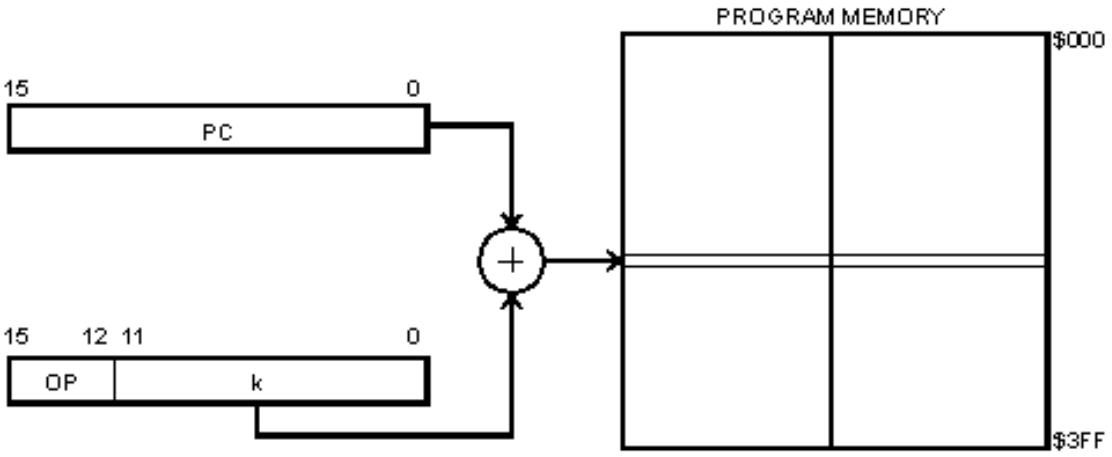
间接程序寻址，IJMP 和 ICALL：

图 19



相对程序寻址，RJMP 和 RCALL：

图 20



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 21 并行取指与指令执行

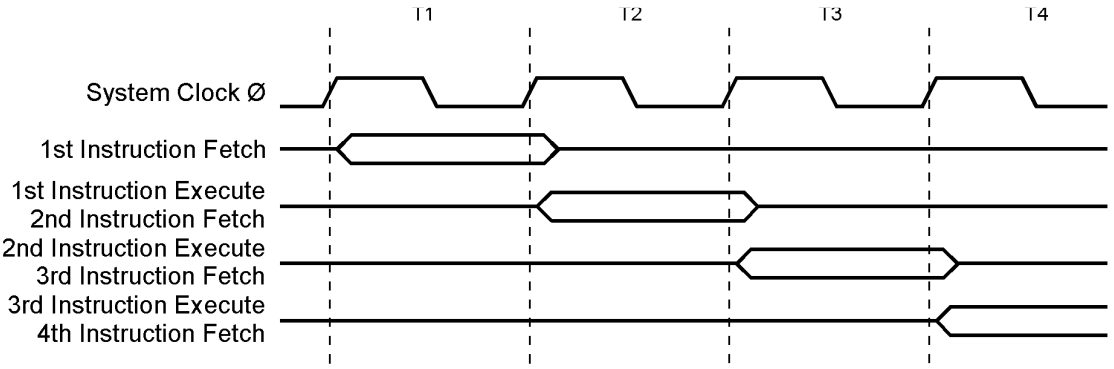


图 21 **演示的是寄存器文件内部时序。在一个时钟周期里，**ALU **可以同时对两个寄存器操作**

数进行操作，同时将结果存回到其中的一个寄存器中去。

图 22 单时钟 ALU 操作

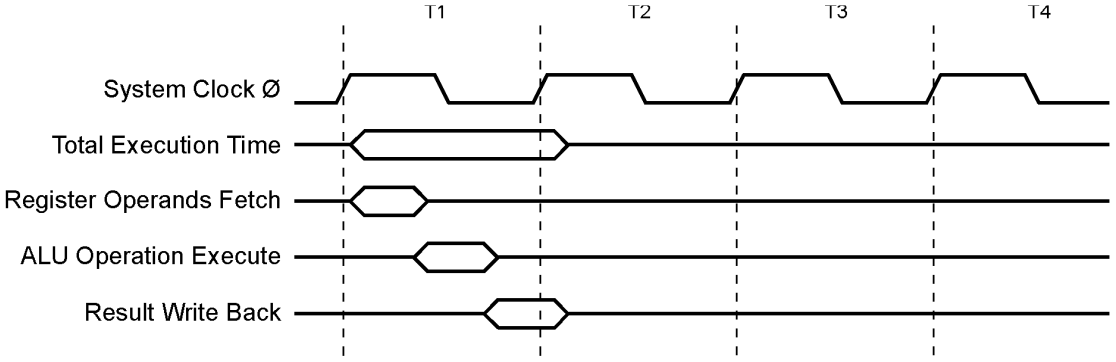
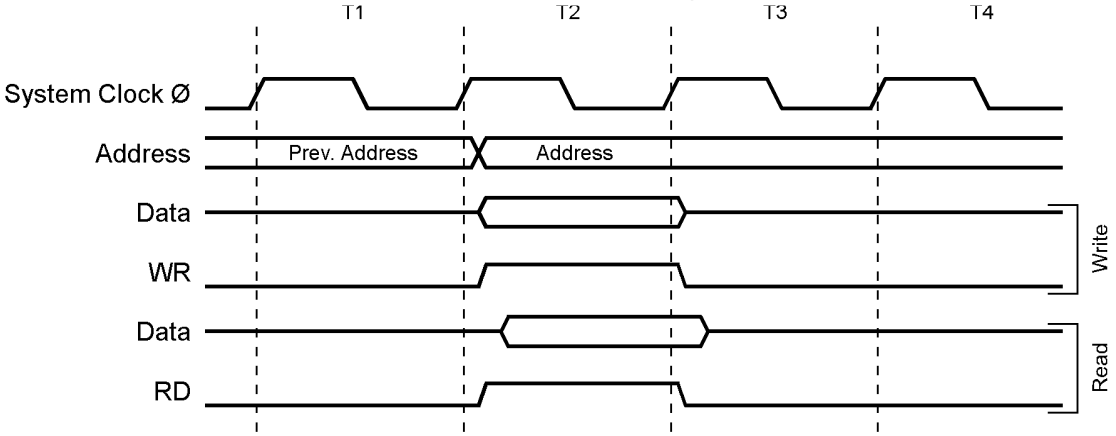


图 23 片内 SRAM 访问周期



I/O 内存

表 2 AT90S2323/2343 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
$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）
$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

AVR2323/2343 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 SPL

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

AT90S2323/2343 有 2 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 3。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
7	$002	TIMER0，OVF0	T/C0 溢出

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    TIM_OVF0    ；T0 溢出
$003      MAIN:   LDI     R16, LOW(REMEND)  ；主程序开始
$00c              OUT     SPL, R16
$00d              <指令>   XXX
—         —       —       —
```

复位源

AT90S2323/2343 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一相对跳转指令一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 24 为复位电路的逻辑图。表 4 定义了复位电路的时序和电参数。

图 24　复位逻辑

AT90S/LS2323 具有可编程的上电起动时间。FSTRT 编程时选择的是最短的起动时间。出厂时这一位没有编程。

AT90S/LS2343 具有固定的起动时间。

表 4　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	-	$0.6V_{CC}$		V
t_{TOUT}	复位延迟周期，FSTRT 未编程（2323）	11	16	21	ms
t_{TOUT}	复位延迟周期，FSTRT 已编程（2323）	1.0	1.1	1.2	ms
t_{TOUT}	复位延迟周期（2343）	11	16	21	μs

注：1.除非电源电压低于 V_{POT}，否则上电复位不会发生。

表 5　复位电参数（V_{CC} = 3.0V）

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	-	$0.6V_{CC}$		V
t_{TOUT}	复位延迟周期，FSTRT 未编程（2323）	22	32	42	ms

t_{TOUT}	复位延迟周期，FSTRT 已编程（2323）	2.0	2.2	2.4	ms
t_{TOUT}	复位延迟周期（2343）	22	32	42	μs

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 5。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。

上电复位：

AT90S2323/2343 设计为可以由 RC 振荡（AT90S/SL2343），片内振荡器（AT90S/LS2323），或外部时钟源驱动。V_{CC} 达到 V_{POT} 后，MCU 要等待 t_{TOUT} 的时间才起动。如果时钟由外部时钟源提供，则在 V_{CC} 达到给定频率所要求的电压之前，不要加载时钟信号。对于 AT90S2323，用户可以选择不同的起动时间。振荡周期数见表 6。而 AT90S2343 的起动时间仅为 1 个看门狗周期。

表 6 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

图 25 MCU 起动，/RESET 与 V_{CC} 相连

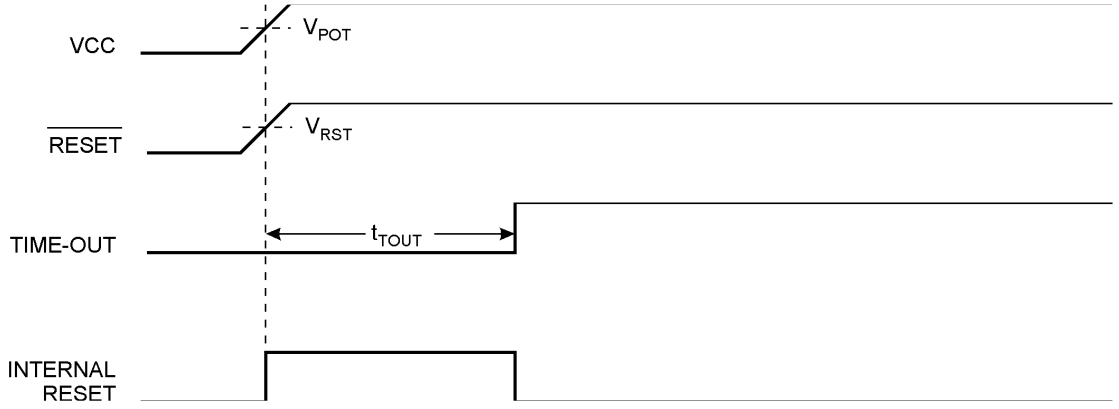
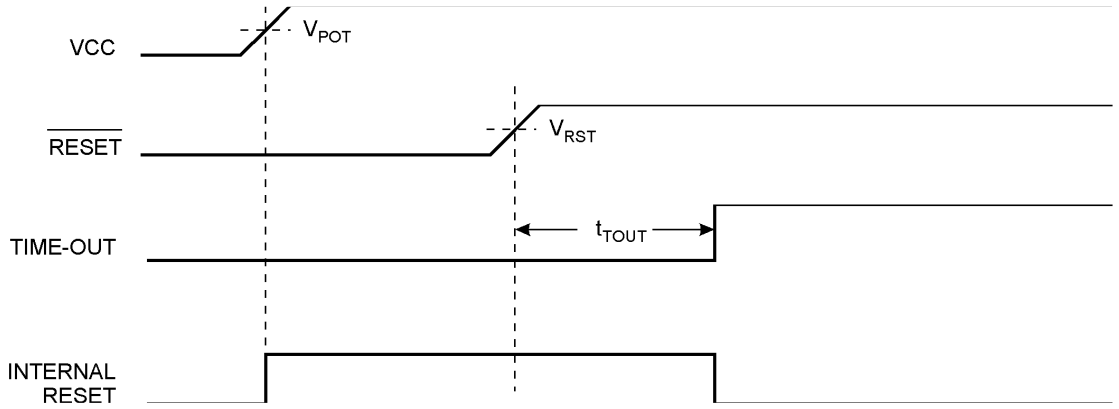


图 26 MCU 启动，/RESET 由外电路控制

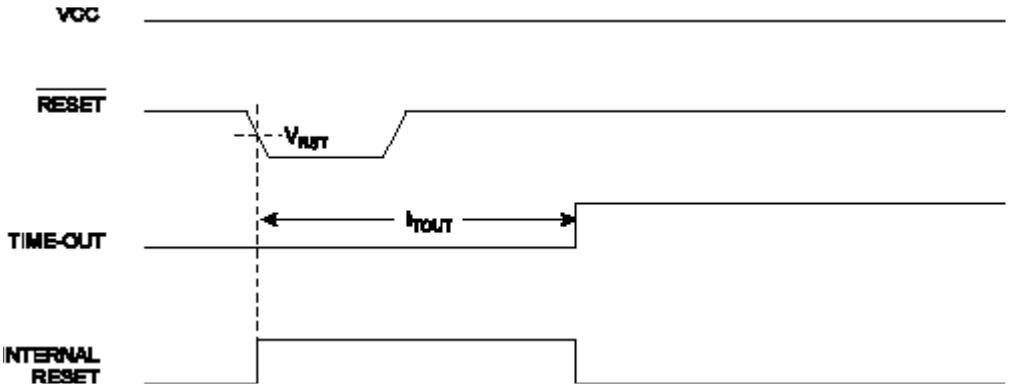


外部复位：

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。

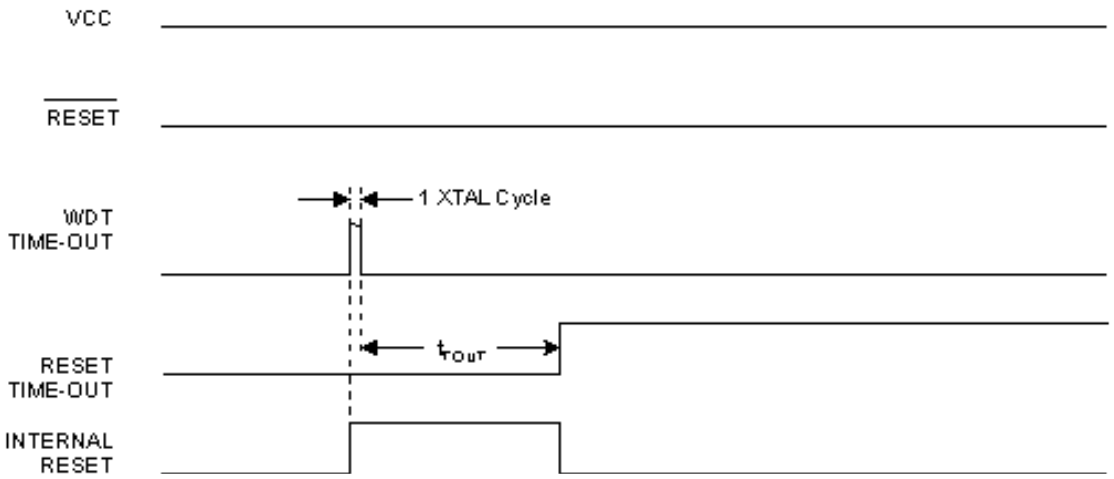
图 27　工作期间的外部复位



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 28　工作期间的看门狗复位



MCU 状态寄存器—MCUSR

BIT	7	6	5	4	3	2	1	0
$34($54)	-	-	-	-	-	-	**EXTRF**	**PORF**
读/写	R	R	R	R	R	R	R/W	R/W
初始值	0	0	0	0	0	0		

位 7..2：保留

EXTRF：外部复位标志

上电复位时这一位没有定义（X）。外部复位时置位。看门狗复位对其没有影响。

PORF：上电复位标志

由上电复位置位。看门狗复位或外部复位对其没有影响。

表 7　复位后的 PORF 和 EXTFR

复位源	PORF	EXTRF
上电复位	1	没有定义
外部复位	不变化	1
看门狗复位	不变化	不变化

如果要利用 PORF 和 EXTRF 来识别复位条件，用户软件要尽早对其清零。检查 PORF 和 EXTRF 的语句在对其清零之前执行。如果某一位在外部复位或看门狗复位之前清零，则复位可以通过如下真值表找出来：

表 8　复位源鉴别

PORF	EXTRF	复位源
0	0	看门狗复位
0	1	外部复位
1	0	上电复位
1	1	上电复位

中断处理：

AT90S2323/2343 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。

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)	-	**INT0**	-	-	-	-	-	-
读/写	R	R/W	R	R	R	R	R	R
初始值	0	0	0	0	0	0	0	0

位 7、5..0：保留

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)	-	**INTF0**	-	-	-	-	-	-
读/写	R	R/W	R	R	R	R	R	R
初始值	0	0	0	0	0	0	0	0

位 7、5..0：保留

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)	-	-	-	-	-	-	**TOIE0**	-
读/写	R	R	R	R	R	R	R/W	R
初始值	0	0	0	0	0	0	0	0

位 7.2、0：保留

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（$002）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)	-	-	-	-	-	-	**TOV0**	-
读/写	R	R	R	R	R	R	R/W	R
初始值	0	0	0	0	0	0	0	0

位 7.2、0：保留

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 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**	-	-	**ISC01**	**ISC00-**
读/写	R	R	R/W	R/W	R	R	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位 7、6、3、2：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。

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 9　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

注意：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。**

休眠模式

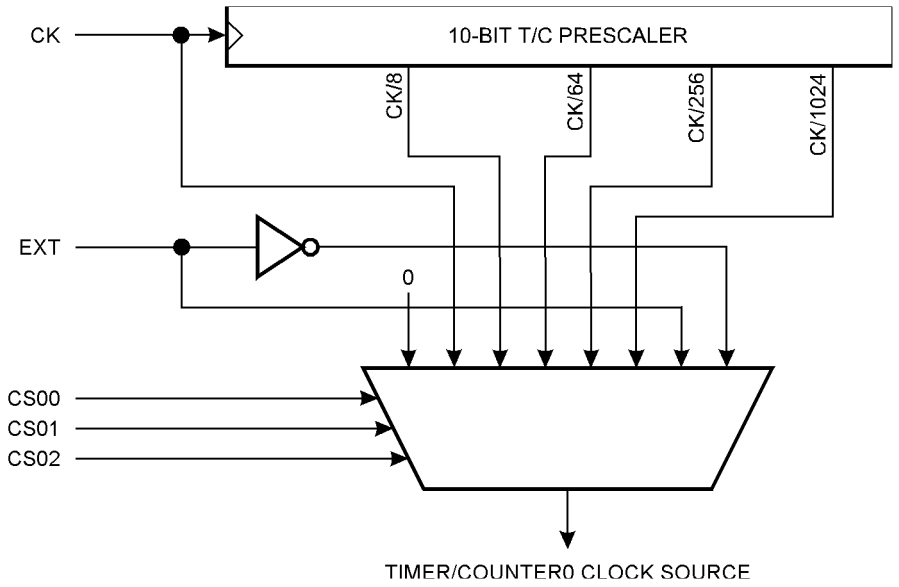
进入休眠模式的条件是 SE **为“**1**”，然后执行** SLEEP **指令。使能的中断将唤醒** MCU**。完成中断例程后，**MCU **执行** SLEEP **以后的指令。在休眠期间，寄存器文件及** I/O **内存的内容不会丢失。如果在休眠模式下复位，则** MCU **从** RESET **向量（**$000**）处开始运行。**
闲置模式：
当 SM **为“**0**”时，**SLEEP **指令将使** MCU **进入闲置模式。在此模式下，**CPU **停止运行，而定时器/计数器、看门狗和中断系统继续工作。内外部中断都可以唤醒** MCU**。如果不需要从模拟比较器中断唤醒** 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

AT90S2323/2343 **内部有一个** 8 **位通用定时器/计数器** T/C0**。**T/C0 **从一个** 10 **位的预分频定时器取得预分频的时钟。**T/C0 **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 29　T/C0 **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 30　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。

看门狗定时器

看门狗定时器由片内独立的振荡器驱动。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。2323/2343 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 31　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 11　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

注意：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。

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。

I/O 口—B 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）。

对于 AT90S/LS2323，B 口是 3 位双向 I/O 口；对于 AT90S/LS2343，B 口是 5 位双向 I/O 口。请注意：PORTB、DDRB 和 PINB 的位 3 和 4 对 AT90S/LS2323 不适用。

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 12　B 口第二功能

管脚	第二功能
PB0	MOSI（程序下载时的数据输入线）
PB1	MISO（程序下载时的数据输出线） INT0（外部中断 0 输入）
PB3	SCK（程序下载时的串行时钟） T0（T/C0 的记数时钟输入）
PB7	CLOCK（时钟输入，仅适用于 AT90S/LS2343）

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)	-	-	-	**PORTB4**				**PORTB0**
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

B 口数据方向寄存器—DDRB

BIT	7	6	5	4	3	2	1	0
$17($37)	-	-	-	**DDB4**				**DDB0**
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

B 口输入引脚地址—PINB

BIT	7	6	5	4	3	2	1	0
$16($36)	-	-	-	**PINB4**				**PINB0**
读/写	R	R	R	R	R	R	R	R
初始值	0	0	0	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 **口的各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

- CLOCKK—PB3
  AT90S/LS2343 **的时钟输入。当** RCEN **为“**0**”时，器件以内部** RC **振荡器为时钟，此引脚作为通用** I/O **口；如果** RCEN **没有编程，则需要外接时钟源。**
- SCK/T0—PB2
  下载程序时的时钟。
  在正常操作过程中，此引脚可作为外部记数时钟输入。
- MISO—PB1
  程序上载时的输出数据。
  在正常操作过程中，此引脚可作为 INT0**。**
- MOSI—PB0
  下载程序时的数据

程序编程

程序和数据锁定位

AT90S2323/2343 具有两个锁定位，如表 14 所示。锁定位只能通过片擦除命令擦除。

表 14　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、在高压串行编程模式下，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

AT90S/LS2323 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”。 串行下载程序时不能访问此熔断位。
- FSTRT 编程（为“0”）后，MCU 选择短起动时间。缺省值为“1”。FSTRT 的改变只能在下一次上电复位后生效。

芯片擦除命令不影响熔断位。

AT90S/LS2343 有两个熔断位：SPIEN 和 RCEN。

- SPIEN 编程（为“0”）后，串行下载程序使能。缺省值为“0”。 串行下载程序时不能访问此熔断位。
- RCEN 编程（为“0”）后，MCU 选择内部 RC 振荡器作为时钟。缺省值为“0”。 RCEN 的改变只能在下一次上电复位后生效。

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。

对于 AT90S/LS2323，其位置为：

1、$000：$1E（表明是 Atmel 生产的）

2、$001：$91（2K 字节的 FLASH）

3、$002：$02（当$01 地址为$91 时，器件为 2323）

对于 AT90S/LS2343，其位置为：

1、$000：$1E（表明是 Atmel 生产的）

2、$001：$91（2K 字节的 FLASH）

3、$002：$03（当$01 地址为$91 时，器件为 2343）

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

AT90S2323/2343 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 高压串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 15　编程电源电压

型号	低压串行编程	高压串行编程
AT90S2323	4.0V – 6.0V	4.5V – 5.5V
AT90LS2323	2.7V – 6.0V	4.5V – 5.5V
AT90S2343	4.0V – 6.0V	4.5V – 5.5V
AT90LS2343	2.7V – 6.0V	4.5V – 5.5V

高压串行编程

图 49　高压串行编程

编程算法：

以下步骤使器件进入高压串行编程模式：

1、上电序列：在 V_{CC} 和 GND 之间加上 4.5 – 5.5V 电压。拉低 PB5 和 PB0，并保持至少 100ns。改变 PB3 的电平至少 4 次，脉宽至少为 100ns。将 PB3 拉低，等待至少 100ns。给 PB5（/RESET）加上 12V 的电压，并在 PB0 变化之前保持至少 100ns。在继续后续指令前等待至少 8μs。

2、FLASH 的编程以字节为单位。首先加地址，然后是数据。PB2（RDY/BSY）变高说明写入过程结束。

3、EEPROM 的编程以字节为单位。首先加地址，然后是数据。PB2（RDY/BSY）变高说明写入过程结束。

4、任何地址的内容可通过读指令由 PB2 读出。

5、下电序列：拉低 PB3；置位 PB5（“1”）；关电源。

数据在时钟的上升沿输入/输出。

图 33　高压串行编程波形

表 16　高压串行编程指令集

指令		指令格式				注释
		指令 1	指令 2	指令 3	指令 4	
片擦除	PB0 PB1 PB2	0_1000_0000_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0100_1100_00 x_xxxx_xxxx_xx	执行完指令 3 后要等待 t_{WLWH_CE}
写 FLASH 高低地址	PB0 PB1 PB2	0_0001_0000_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_00**aa**_00 0_0001_1100_00 x_xxxx_xxxx_xx	0_**bbbb_bbbb**_00 0_0000_1100_00 x_xxxx_xxxx_xx		指令 2 每 256 字节执行一次，指令 3 则每个地址执行一次。
写 FLASH 低字节	PB0 PB1 PB2	0_**iiii_iiii**_00 0_0010_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 0_0000_0000_00		指令 3 后要等待 PB2 变高。
写 FLASH 高字节	PB0 PB1 PB2	0_**iiii_iiii**_00 0_0010_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0111_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0111_1100_00 0_0000_0000_00		指令 3 后要等待 PB2 变高。
读 FLASH 高低地址	PB0 PB1 PB2	0_0000_0010_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_00**aa**_00 0_0001_1100_00 x_xxxx_xxxx_xx	0_**bbbb_bbbb**_00 0_0000_1100_00 x_xxxx_xxxx_xx		每个地址都需执行指令 2 和 3
读 FLASH	PB0	0_0000_0000_00	0_0000_0000_00			每个地址都

低字节	PB1 PB2	0_0110_1000_00 x_xxxx_xxxx_xx	0_0110_1100_00 **o_oooo_oooo**_xx			需执行指令 1 和 2
读 FLASH 高字节	PB0 PB1P PB2	0_0000_0000_00 0_0111_1000_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0111_1100_00 **o_oooo_oooo**_xx			每个地址都需执行指令 1 和 2
写 EEP 低地址	PB0 PB1 PB2	0_0001_0001_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0**bbb_bbbb**_00 0_0000_1100_00 x_xxxx_xxxx_xx			每个地址都需执行指令 1 和 2
写 EEP 字节	PB0 PB1 PB2	0_**iiii_iiii**_00 0_0010_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 0_0000_0000_00		指令 3 后要等待 PB2 变高。
读 EEP 低地址	PB0 PB1 PB2	0_0000_0011_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0**bbb_bbbb**_00 0_0000_1100_00 x_xxxx_xxxx_xx			每个地址都需执行指令 1 和 2
读 EEP 字节	PB0 PB1 PB2	0_0000_0000_00 0_0110_1000_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 **o_oooo_oooo**_xx			每个地址都需执行指令 1 和 2
写熔丝位（2323）	PB0 PB1 PB2	0_0100_0000_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_11**S**1_111**F**_00 0_0010_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 x_xxxx_xxxx_xx	执行完指令 3 后要等待 t_{WLWH_PFB}，**S**、**F** 为 “0” 代表编程
写熔丝位（2343）	PB0 PB1 PB2	0_0100_0000_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_11**S**1_111**R**_00 0_0010_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 x_xxxx_xxxx_xx	执行完指令 3 后要等待 t_{WLWH_PFB}，**S**、**R** 为 “0” 代表编程
写锁定位	PB0 PB1 PB2	0_0010_0000_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_1111_1**21**1_00 0_0010_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_0100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 x_xxxx_xxxx_xx	指令 4 后要等待 PB2 变高。**2**，**1**= “0” 表示编程
读熔丝位/锁定位（2323）	PB0 PB1 PB2	0_0000_0100_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1000_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 **1_2S**xx_xx**F**x_xx		若 **1**，**2**，**S**，**F** 为 “0” 表示已编程
读熔丝位/锁定位（2343）	PB0 PB1 PB2	0_0000_0100_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1000_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 **1_2S**xx_xx**R**x_xx		若 **1**，**2**，**S**，**R** 为 “0” 表示已编程
读厂标	PB0 PB1 PB2	0_0000_1000_00 0_0100_1100_00 x_xxxx_xxxx_xx	0_0000_00**bb**_00 0_0000_1100_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1000_00 x_xxxx_xxxx_xx	0_0000_0000_00 0_0110_1100_00 **o_oooo_ooo**x_xx	重复指令 2-4 以读取其他字节

注意： **a** = 地址的高比特位

b = 地址的低比特位

i = 输入的数据

o = 输出的数据

x = 不用管

1 = 锁定位 1

2 = 锁定位 2

F = FSTRT

R = RCEN

S = SPIEN

高压串行编程特性

图 34 高压串行编程时序

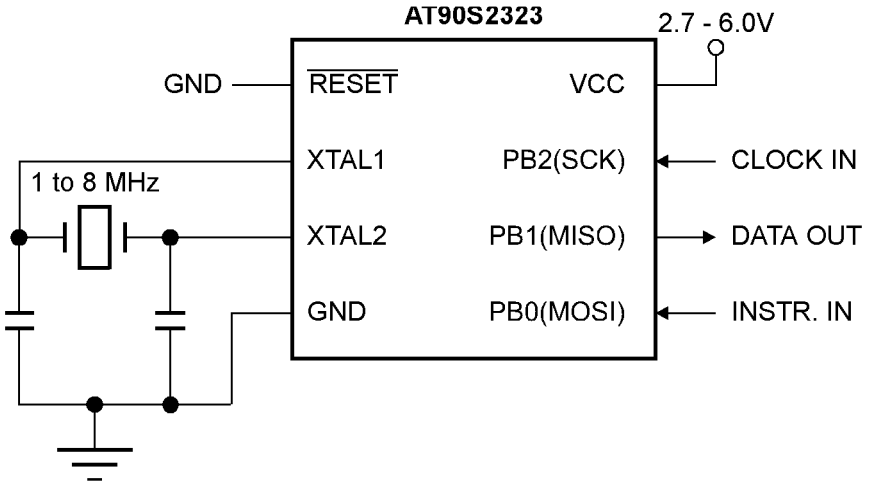
表 17 高压串行编程特性 T_A = 25℃±10%，V_{CC} = 5V±10%

符号	参数	最小值	典型值	最大值	单位
t_{SHSL}	脉冲（高）宽度	100			ns
t_{SLSH}	SCI（XTAL1/PB3）脉冲（低）宽度	100			ns
t_{IVSH}	SDI（PB0），SII（PB1）有效到 SCI 高	50			ns
t_{SHIX}	SCI 高后 SDI（PB0），SII（PB1）保持	50			ns
t_{SHOV}	SCI（XTAL1/PB3）高到 SDO（PB2）有效	10	16	32	ns
t_{WLWH_CE}	片擦除时指令 3 等待时间	5	10	15	ms
t_{WLWH_PFB}	写熔丝位时指令 3 等待时间	1.0	1.5	1.8	ms

低压串行下载

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 35　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~$07F。

时钟可以从 XTAL1/PB3 引脚输入，或者使用内部 RC 振荡器（仅对 2343 有效）。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 AT90S2323/2343，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（PB0）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（PB1）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
AT90S2323/2343	$00	$FF

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$FF，否则，读取结果为写入的数据。但是对于数据$FF，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

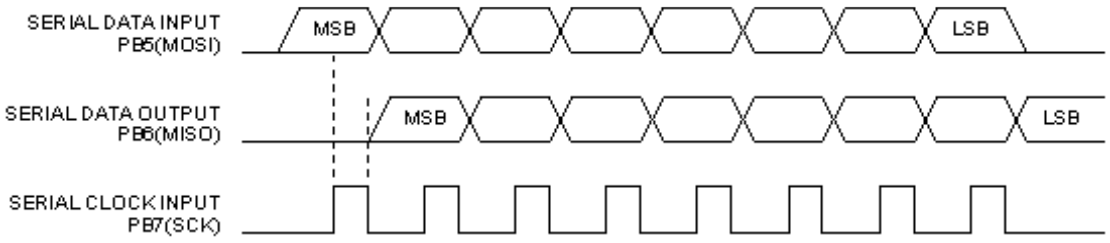


表 28　AT90S2323/2343 的串行编程指令

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	x**bbb bbbb**	**oooo oooo**	**从地址 b 读取数据 o**
写 EEPROM	1100 0000	0000 0000	x**bbb bbbb**	**iiii iiii**	**写数据 i 到地址 b**
读锁定/熔丝位（2323）	0101 1000	xxxx xxxx	xxxx xxxx	**12S**x xxx**F**	**“0”代表编程**
读锁定/熔丝位（2343）	0101 1000	xxxx xxxx	xxxx xxxx	**12S**x xxx**R**	**“0”代表编程**
写锁定位	1010 1100	111x x***21***x	xxxx xxxx	xxxx xxxx	**写锁定位**
写 FSTRT（2323）	1010 1100	1011 111**F**	xxxx xxxx	xxxx xxxx	**“0”代表编程** [1,2]
写 RCEN（2343）	1010 1100	1011 111**R**	xxxx xxxx	xxxx xxxx	**“0”代表编程** [1,2]
读厂标	0011 0000	xxxx xxxx	xxxx xx**bb**	**oooo oooo**	**从地址 b 读厂标 o**[3]

注意：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

F = FSTRT

R = RCEN

S = SPIEN

2、RCEN **和** FSTRT **改变后要重新上电。**

3、**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 I_{OL} =10Ma，V_{CC} = 3V			0.5 0.4	V
V_{OH}	输出高电压 (4) B、D 口	I_{OH} =20mA，V_{CC} = 5V I_{OH} =10mA，V_{CC} = 3V	4.2 2.4			V
I_{IL}	输入泄露电流 I/O 脚	V_{CC} = 6V， pin low			8.0	μA
I_{IH}	输入泄露电流 I/O 脚	V_{CC} = 6V， pin low			800	nA
RRST	复位上拉电阻		100		500	kΩ
$R_{I/O}$	I/O 口的上拉电阻		30		150	kΩ
I_{CC}	AT90S2343	工作，4MHz，V_{CC} =3V			3.0	mA
		空闲，4MHz，V_{CC} =3V			1.1	mA
		掉电， 4MHz (3)， V_{CC} =3V，看门狗使能			25.0	μA
		掉电， 4MHz (3)， V_{CC} =3V，看门狗关闭			20.0	μA
	AT90S2323	工作，4MHz，V_{CC} =3V			4.0	mA
		空闲，4MHz，V_{CC} =3V		1.0	1.2	mA
		掉电， 4MHz (3)， V_{CC} =3V，看门狗使能		9.0	15.0	μA
		掉电， 4MHz (3)， V_{CC} =3V，看门狗关闭		< 1	2.0	μA

注意:

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、 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。

定货信息

速度（MHz）	电源（V）	定货号	封装	工作范围
4	2.7 – 6.0	AT90S2323 – 4PC AT90S2323 – 4SC	8P3 8S2	**商用** （0℃ - 70℃）
		AT90S2323 – 4PI AT90S2323 – 4SI	8P3 8S2	**工业** （-40℃ - 85℃）
10	4.0 – 6.0	AT90S2323 – 10PC AT90S2323 – 10SC	8P3 8S2	**商用** （0℃ - 70℃）
		AT90S2323 – 10PI AT90S2323 – 10SI	8P3 8S2	**工业** （-40℃ - 85℃）
4	2.7 – 6.0	AT90S2343 – 4PC AT90S2343 – 4SC	8P3 8S2	**商用** （0℃ - 70℃）
		AT90S2343 – 4PI AT90S2343 – 4SI	8P3 8S2	**工业** （-40℃ - 85℃）
10	4.0 – 6.0	AT90S2343 – 10PC AT90S2343 – 10SC	8P3 8S2	**商用** （0℃ - 70℃）
		AT90S2343 – 10PI AT90S2343 – 10SI	8P3 8S2	**工业** （-40℃ - 85℃）

封装类型	
8P3	8 **脚，**0.300’’ **宽，塑料双列直插**（PDIP）
8S2	8 **脚，**0.200’’ **宽，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.）