AT90S1200

特点

1. AVR RISC 结构
2. AVR—高性能、低功耗 RISC 结构
   —89 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 12MHz 时具有 12MIPS 的性能
3. 数据和非易失性程序内存
   —1K 字节的在线可编程 FLASH（擦除次数：1000 次）
   —64 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
   —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
5. 特别的 MCU 特点
   —低功耗空闲和掉电模式
   —内外部中断源
   —可选的片内 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：2.0mA
   —空闲模式：0.4mA
   —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（AT90S1200-4）
   —4.0V-6.0V（AT90S1200-12）
10. 速度
    —0-4MHz（AT90S1200-4）
    —0-8MHz（AT90S1200-12）

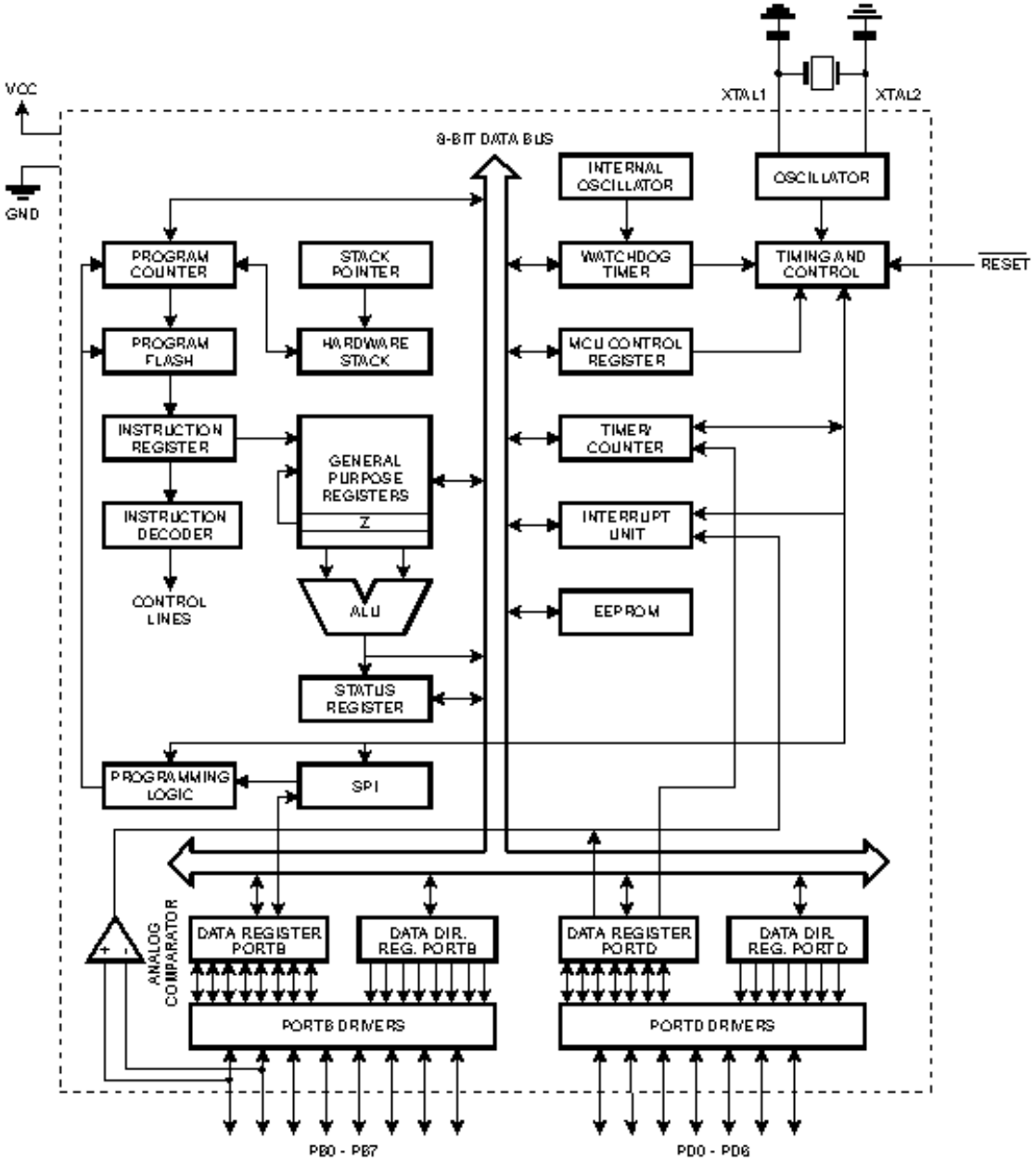
描述

AT90S1200 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，AT90S1200 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　AT90S1200 结构方框图



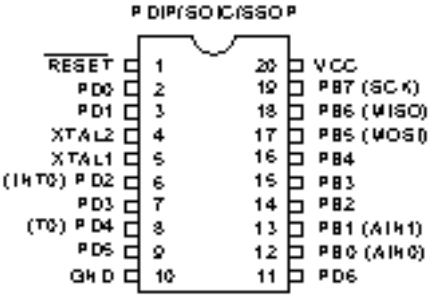
这种结构可以有效地支持高级语言编程，同时保持代码密度紧凑。AT90S1200 具有以下特点：1K 字节 FLASH，64 字节 EEPROM，15 个通用 I/0 口，32 个通用工作寄存器，内外中断源，可编程的看门狗定时器，下载程序用的 SPI 口以及两种可通过软件选择的省电模式。工作于空闲模式时，CPU 将停止运行，而寄存器、定时器/计数器、看门狗和中断系统继续工作；掉电模式时振荡器停止工作，所有功能都被禁止，而寄存器内容得到保留。只有外部中断或硬件复位才可以退出此状态。

器件是以 ATMEL 的高密度非易失性内存技术生产的。片内 FLASH 允许多次编程。通过将增强的 RISC 8 位 CPU 与 FLASH 集成在一个芯片内，1200 为许多嵌入式控制应用提供了灵活而

低成本的方案。

AT90S1200 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。

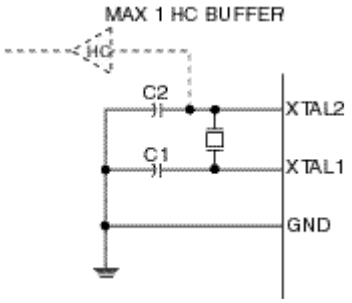
XTAL1： 振荡器放大器的输入端。

XTAL2： 振荡器放大器的输出端。

晶体振荡器：

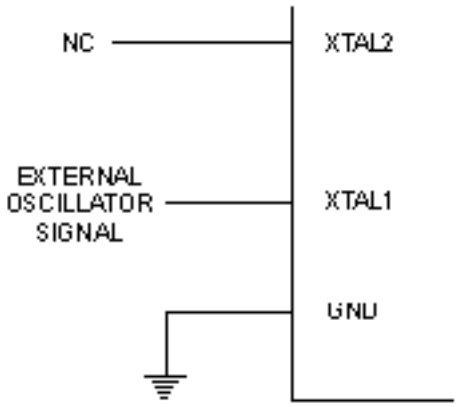
XTAL1 和 XTAL2 分别是片内振荡器的输入、输出端，可使用晶体振荡器或是陶瓷振荡器。当使用外部时钟时，XTAL2 应悬空。

图 2　振荡器连接



注：若要利用 MCU 的振荡器作为外围器件的时钟，应该如上图一样连接一个 HC 缓冲器。

图 3　外部时钟驱动配置

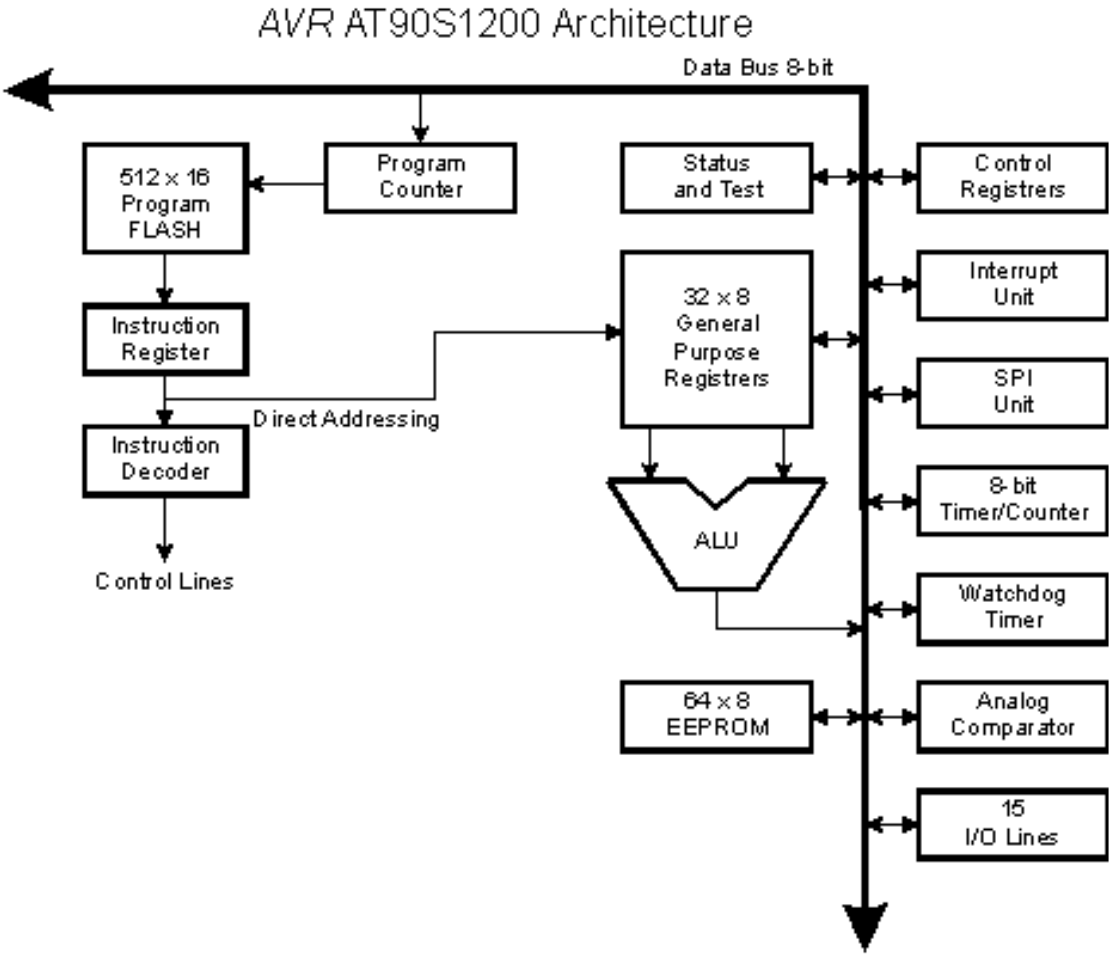


片内 RC 振荡器

可以选择片内的 RC **振荡器（频率为** 1M**）作为** MCU **的时钟。从而** AT90S1200 **可以在“零外围”的情况下工作。当** RCEN **位编程为“**0**”时** RC **振荡器即成为** MCU **时钟。**RCEN **位只能在并行编程模式下改变。所以如果要利用** RC **振荡器来进行串行下载的话，首先要并行改变** RCEN**。为了客户方便起见，产品** AT90S1200A **在出厂时已经将** RCEN **编程。**

结构纵览

图 4 AT90S1200 AVR RISC **结构**



快速访问寄存器堆包含 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：读取寄存器堆中的两个操作数，执行操作，将结果存回到寄存器堆。** 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。** 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 512 **个地址空间。所有的** AVR **指令都为** 16 **位长，也就是说，每一个程序内存地址都包含一条** 16 **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。 1200 **的堆栈为** 3 **级硬件堆栈。** I/O **内存空间包含** 64 **个** CPU **外围的地址，如控制寄存器，** T/C**，** A/D **等。** AVR **结构的内存空间是线性的。**

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。**

通用工作寄存器堆

图 5　AVR CPU 通用工作寄存器

通用寄存器	R0
	R1
	R2
	…
	…
	R28
	R29
	R30(Z)
	R31

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堆的后半部分：R16 到 R31。

寄存器 R30 也用为寄存器间接寻址的 8 位指针。

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

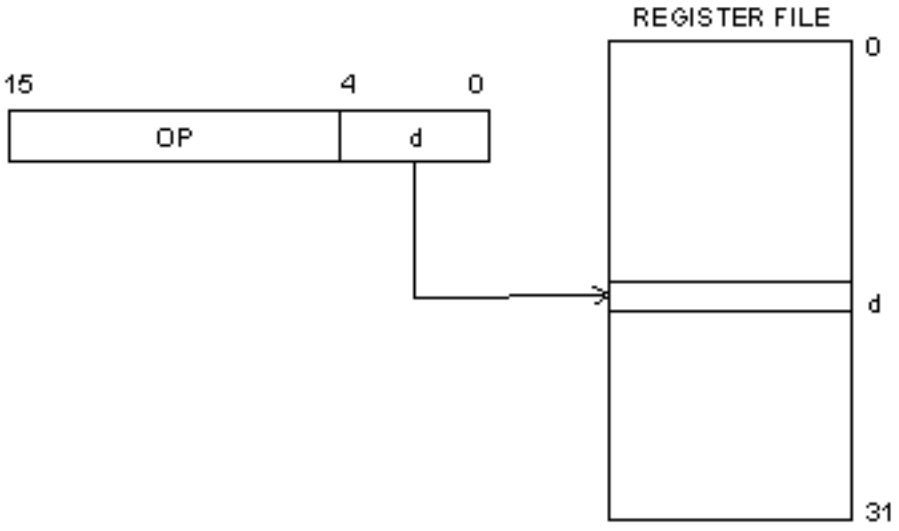
AT90S1200 具有 1K 字节的 FLASH。因为所有的指令为 16 位宽，故尔 FLASH 结构为 512×16。FLASH 的擦除次数至少为 1000 次。

AT90S1200 的程序计数器（PC）为 9 位宽，可以寻址到 512 个字的 FLASH 程序区。

程序和数据寻址模式

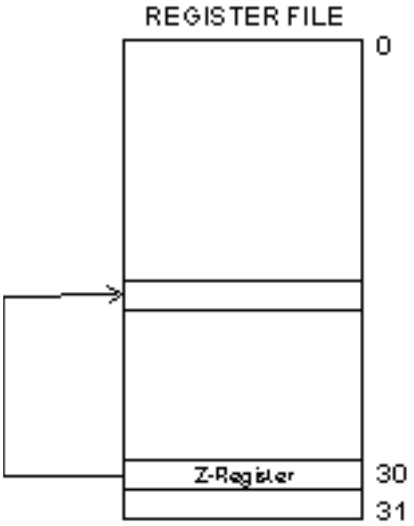
1、单寄存器直接寻址

图 6



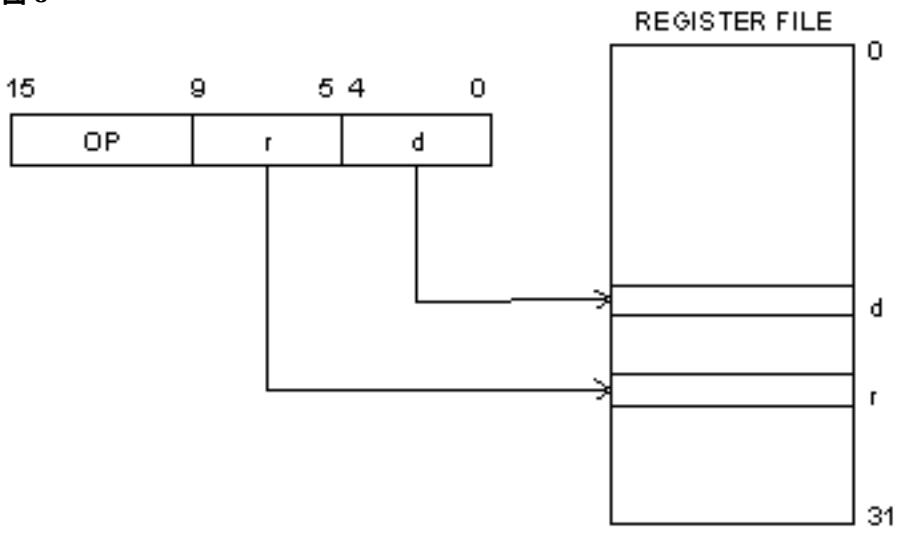
2、单寄存器间接寻址

图 7



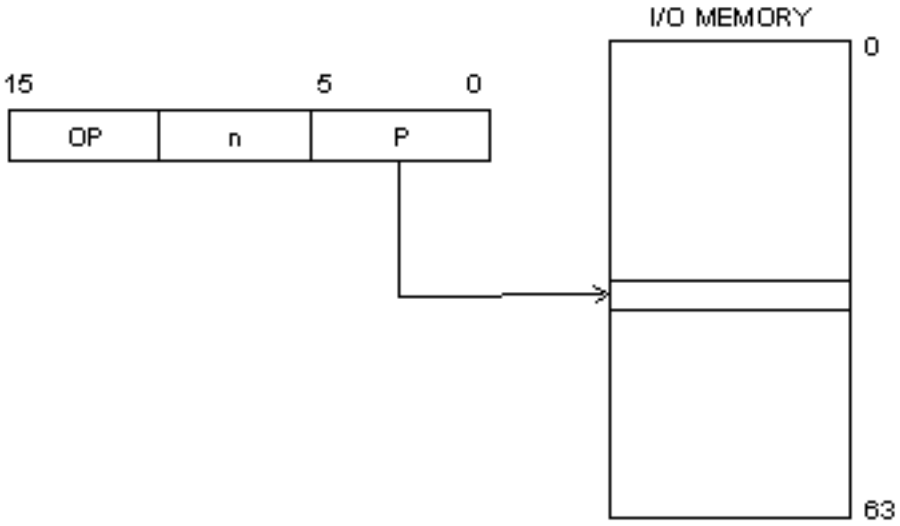
3、双寄存器直接寻址

图 8



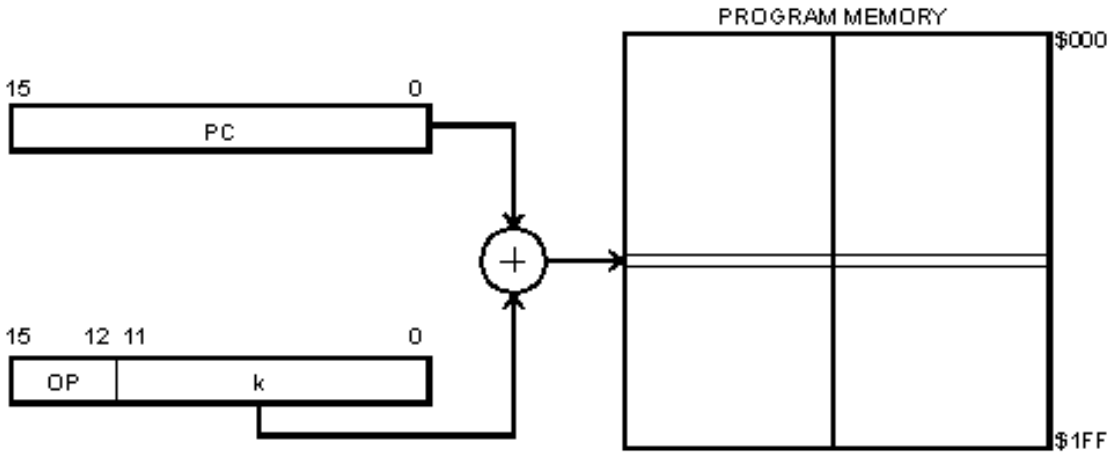
4、I/O 直接寻址

图 9



5、程序相对寻址

图 10



子程序和中断硬件堆栈

AT90S1200 使用 3 级硬件堆栈，宽度为 9 位。
RCALL 指令和中断将 PC 推入堆栈 0，而原有的堆栈 0 和 1 的内容进入深一级的位置。执行 RET 或 RETI 后堆栈 0 的 PC 将出栈，而堆栈 1 和 2 的内容上升一级。
如果有多于 3 个的子程序或中断连续（嵌套）发生，则第一个进栈的内容将丢失。

EEPROM

AT90S1200 包含 64 字节的 EEPROM，其写寿命为 100000 次。具体操作见后续章节。

指令操作时序

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 11 取指与指令执行同时进行

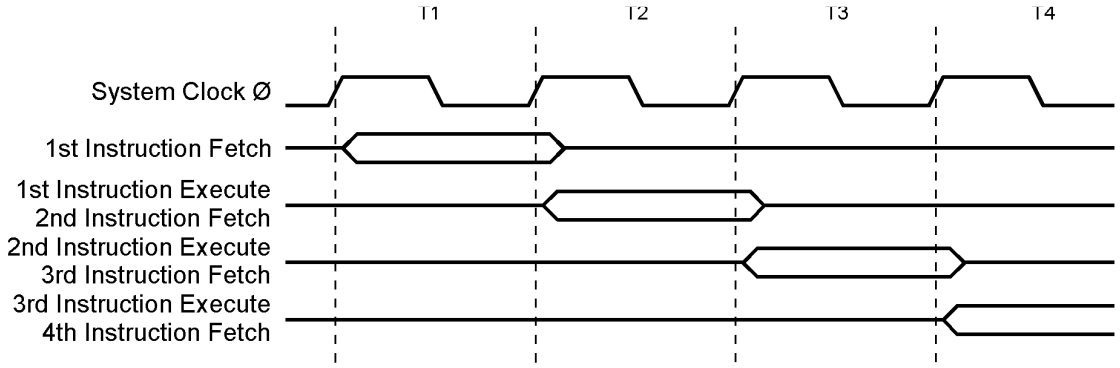
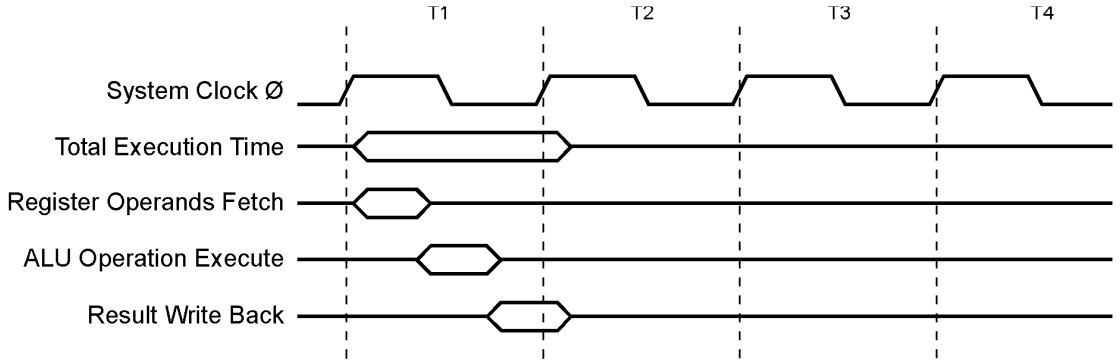


图 12 单时钟 ALU 操作



I/O 内存

表 1 AT90S1200 的 I/O 空间

地址（16 进制）	名称	功　　能
$3F	SREG	状态寄存器
$3B	GIMSK	通用中断屏蔽寄存器
$39	TIMSK	T/C 屏蔽寄存器
$38	TIFR	T/C 中断标志寄存器
$35	MCUCR	MCU 控制寄存器
$33	TCCR0	T/C0 控制寄存器
$32	TCNT0	T/C0（8 位）
$21	WDTCR	看门狗控制寄存器
$1E	EEAR	EEPROM 地址寄存器
$1D	EEDR	EEPROM 数据寄存器
$1C	EECR	EEPROM 控制寄存器
$18	PORTB	B 口数据寄存器
$17	DDRB	B 口数据方向寄存器
$16	PINB	B 口输入引脚
$12	PORTD	D 口数据寄存器
$11	DDRD	D 口数据方向寄存器
$10	OIND	D 口输入引脚
$08	ACSR	模拟比较器控制及状态寄存器

AVR1200 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。

为了与后续产品兼容，保留未用的位应写“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。

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	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。

复位和中断处理

AT90S1200 有 3 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。这 3 个中断事件有自己的使能位。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	中断定义
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	TIMER0，OVF0	T/C0 溢出
4	$003	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    TIM0_OVF        ；T0 溢出
$003                    RJMP    ANA_COMP        ；模拟比较器
；
$004        MAIN:       <指令>  XXX             ；主程序开始
—           —           —       —
```

复位源

AT90S1200 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 13 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 13　复位逻辑

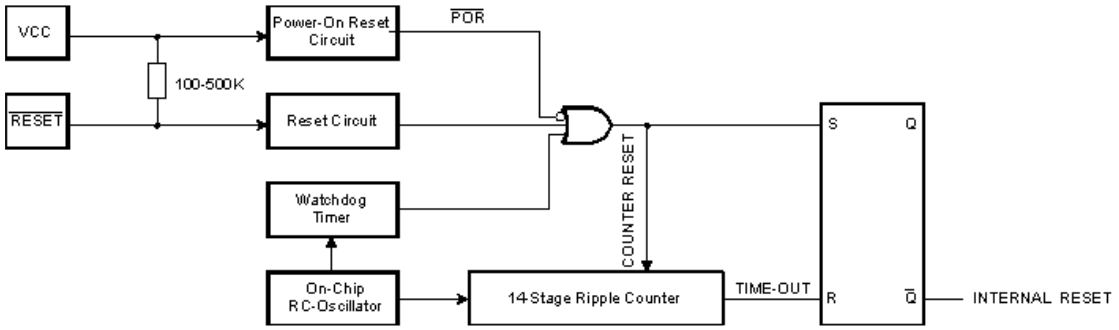


表 3　复位电参数（V_{CC} = 5.0V）

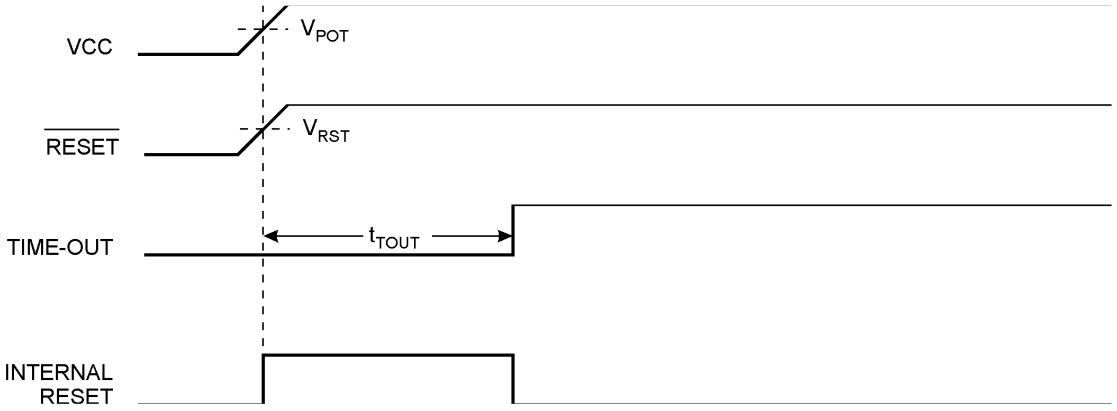
符号	参数	最小值	典型值	最大值	单位
V_{POT}(1)	上电复位电压门限（上升）	0.8	1.2	1.6	V
	上电复位电压门限（下降）	0.2	0.4	0.6	V
V_{RST}	管脚门限电压	-	-	$0.85V_{CC}$	V
t_{POR}	上电复位周期	2	3	4	Ms
t_{TOUT}	复位延迟周期（等于 16K 个 WDT 时钟）	11	16	21	Ms

注：1.**除非电源电压低于** V_{POT}，**否则上电复位不会发生。**

上电复位：

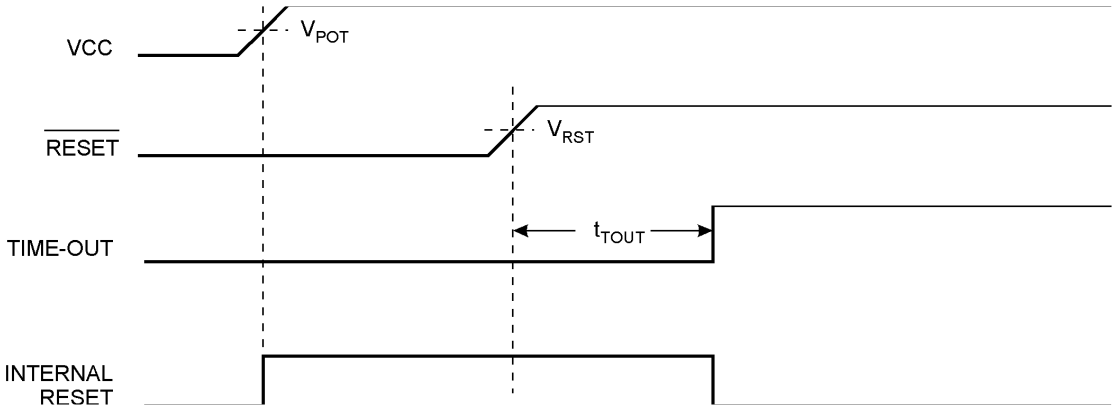
上电复位（POR**）保证器件在上电时正确复位。如图** 13 **所示，在电源电压达到** V_{POT}**后，片内定时器将启动延时。**

图 14　MCU **启动，**/RESET **与** V_{CC} **相连**



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** 15。

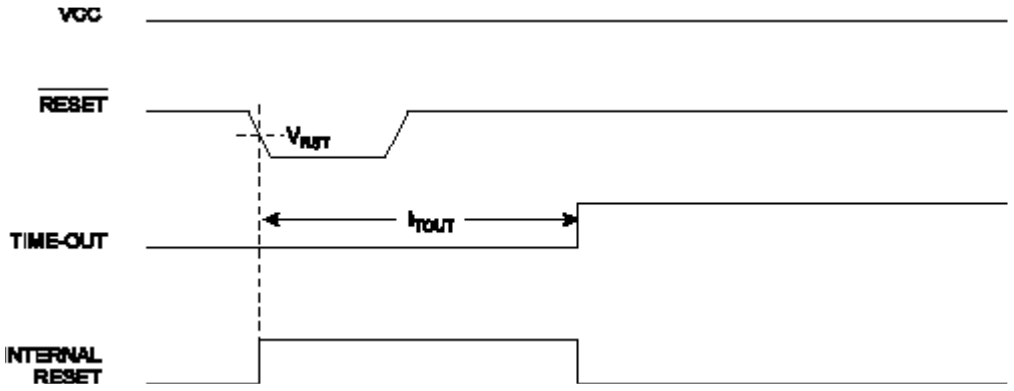
图 15　MCU 启动，/RESET 由外电路控制



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。**

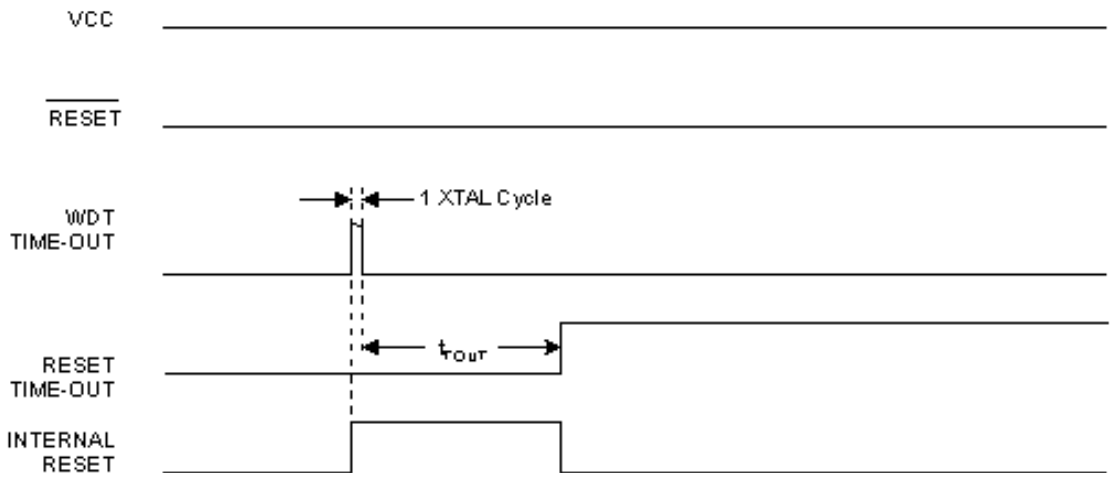
图 16　工作期间的外部复位



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 17　工作期间的看门狗复位



中断处理：

AT90S1200 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”，则中断标志位置位，并一直保持到中断执行，或者被软件清除。

如果全局中断标志被清零，则所有的中断都不会被执行，直到 I 置位。

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	-	**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。

T/C 中断屏蔽寄存器—TIMSK

BIT	7	6	5	4	3	2	1	0
$39	-	-	-	-	-	-	**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	-	-	-	-	-	-	**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。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 INT0 设置为低电平触发时，只要电平为低，中断就一直挂起。即使 INT0 配置为输出中断也会发生。这种特性可以用来实现软件中断。

用户不能直接访问中断标志位。如果怀疑有一个边沿触发中断被挂起，则标志位可以通过如下步骤清除：

1、清除 GIMSK 的 INT0 位以禁止外部中断。

2、选择电平触发中断方式。

3、选择需要的中断沿。

4、置位 INT0，重新使能外部中断。

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 自动入栈。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。

要注意 AT90S1200 只有一个 3 级硬件堆栈。如果 3 个以上例程嵌套发生，则只有最后的 3 个返回地址得到保留，其他的将丢失。

MCU控制寄存器—MCUCR

BIT	7	6	5	4	3	2	1	0
$35	-	-	**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：中断检测控制位

选择 INT0 中断的边沿或电平，如下表所示：

表 4　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。

INT0 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，寄存器堆及 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。

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）可以使 MCU 脱离掉电模式。

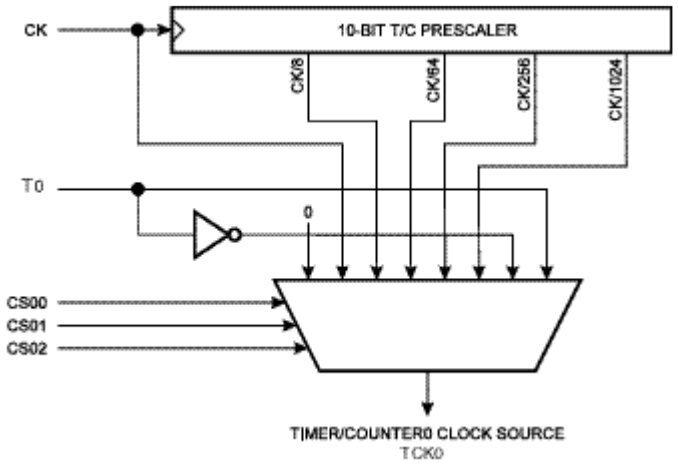
使用外部电平中断唤醒 MCU 时要注意保持低电平大于 T_{TOUT} 的时间，否则 MCU 继续保持掉电模式。

定时器/计数器 0（T/C0）

AT90S1200 内部有一个 8 位的通用定时器/计数器。T/C0 从 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。

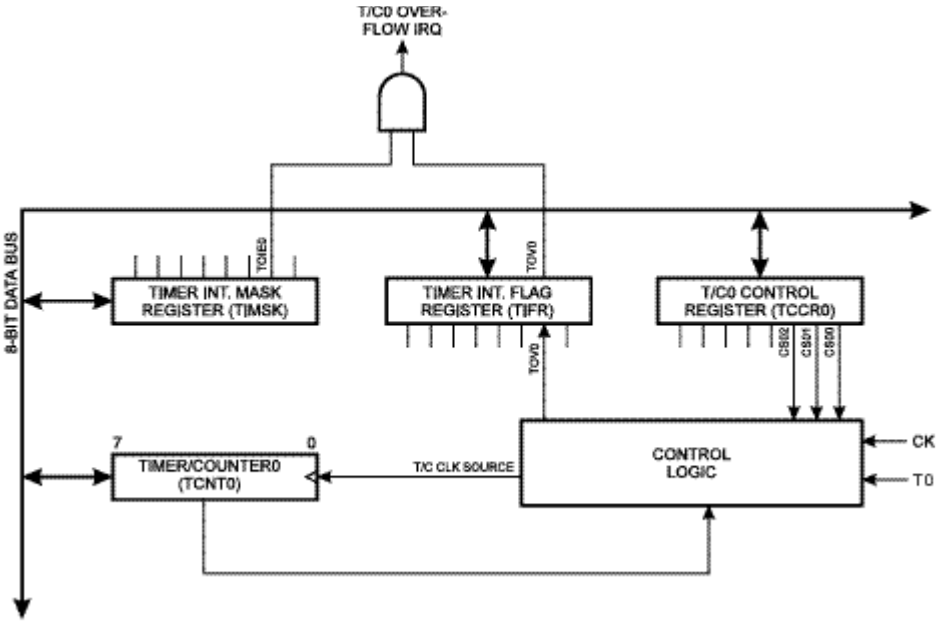
T/C0 的预分频器

图 18　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。对于 T/C0 还可以选择 CK、外部时钟，以及停止工作。图 19 显示了 T/C0 的工作框图。

图 19　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。

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	-	-	-	-	-	**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 5 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

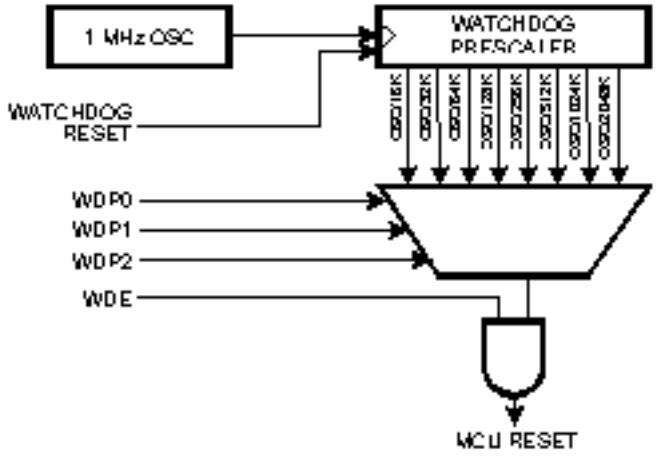
BIT	7	6	5	4	3	2	1	0
$32	**MSB**							**LSB**
读/写	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

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。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**。**MCU **从复位地址重新开始执行。**

图 20　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	-	-	-	-	**WDE**	**WDP2**	**WDP1**	**WDP0**
读/写	R	R	R	R	R/W	R/W	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位 7..4**：保留**

WDE：看门狗使能

WDE 为“1”时，看门狗使能

WDP2..0：预分频器

表 6　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，建议使用低电压复位电路。

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	-	-	**EEAR5**	**EEAR4**	**EEAR3**	**EEAR2**	**EEAR1**	**EEAR0**
读/写	R	R	R/W	R/W	R/W	R/W	R/W	R/W
初始值	0	0	X	X	X	X	X	X

位 7..6：保留

EEAR5..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	**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
$1E	-	-	-	-	-	-	**EEWE**	**EERE**
读/写	R	R	R	R	R	R	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位 7..2：保留

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。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。

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。

注意：当一个访问 EEPROM 的中断例程被另一个访问 EEPROM 的中断例程中断时，EEAR 和 EEDR 会被改变，造成被中断的例程无法正确访问 EEROM。因此，要确保全局中断使能位 I 在整个中断例程中都为“0”。

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。

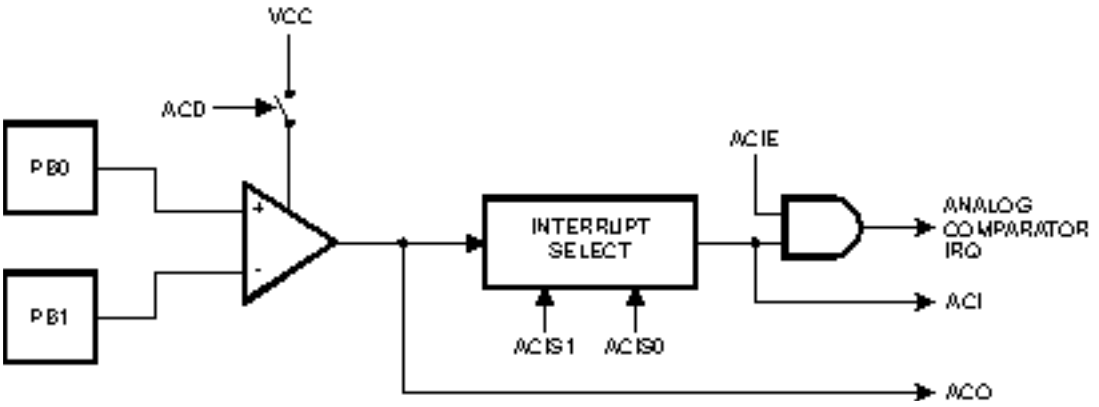
2、当 V_{CC} 过低时使 AVR 内核处于掉电休眠状态。这可以防止 CPU 对程序解码和执行代码，有效防止对 EEPROM 的误操作。

3、将那些不需修改的常数存储于 FLASH 之中。

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）。其框图如图 21 所示。

图 21 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	**ACD**	-	**ACO**	**ACI**	**ACIE**	-	**ACIS1**	**ACIS0**
读/写	R/W	R	R	R/W	R/W	R	R/W	R/W
初始值	0	0	0	0	0	0	0	0

位 6、2：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。

ACIS1，ACIS0：模拟比较器中断模式选择

表 7　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

注意：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），数据方向寄存器—DDRB（$17）和输入引脚—PINB（$16）。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 8　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）
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	**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	**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	**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 9　**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
- 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 22　B 口示意图（PB0 **和** PB1）

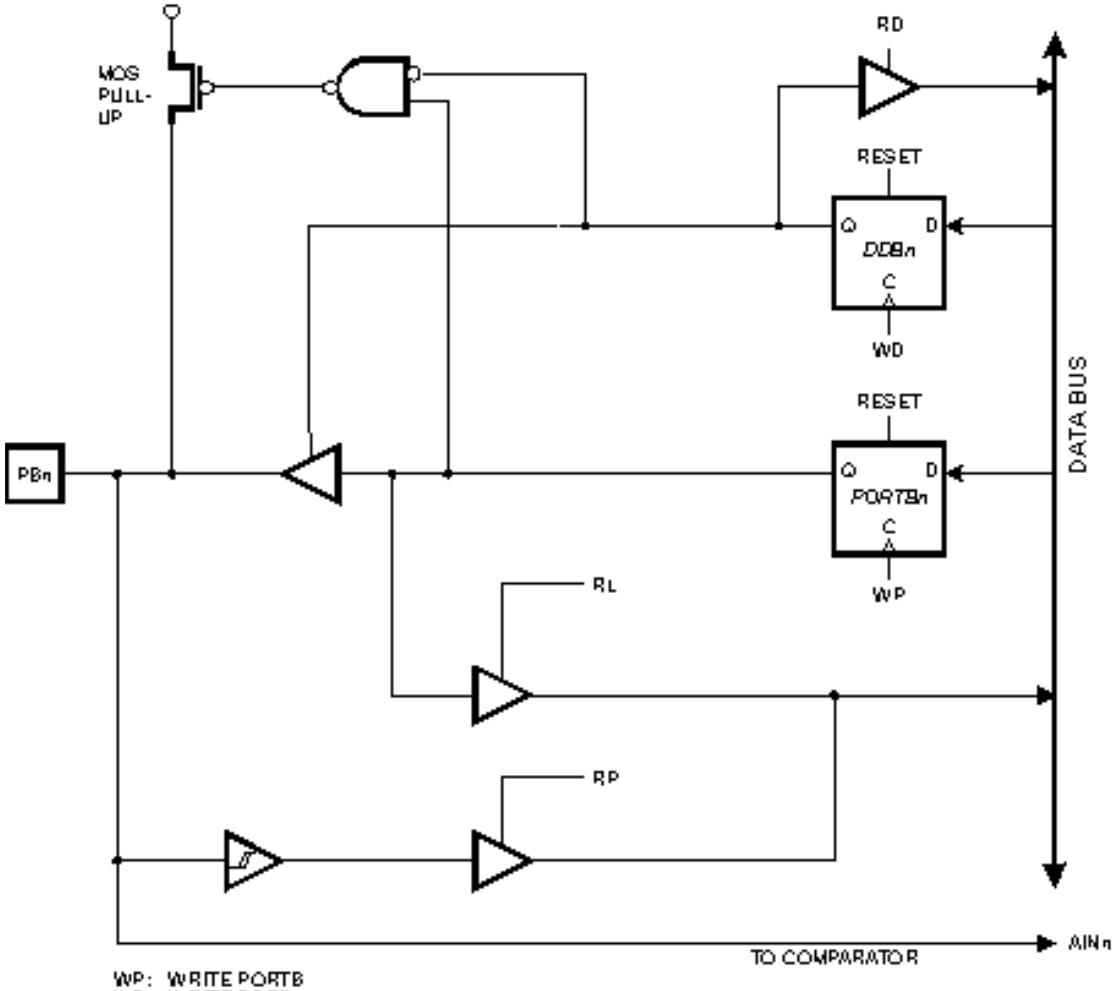


图 23　B 口示意图（PB2、PB3 **和** PB4）

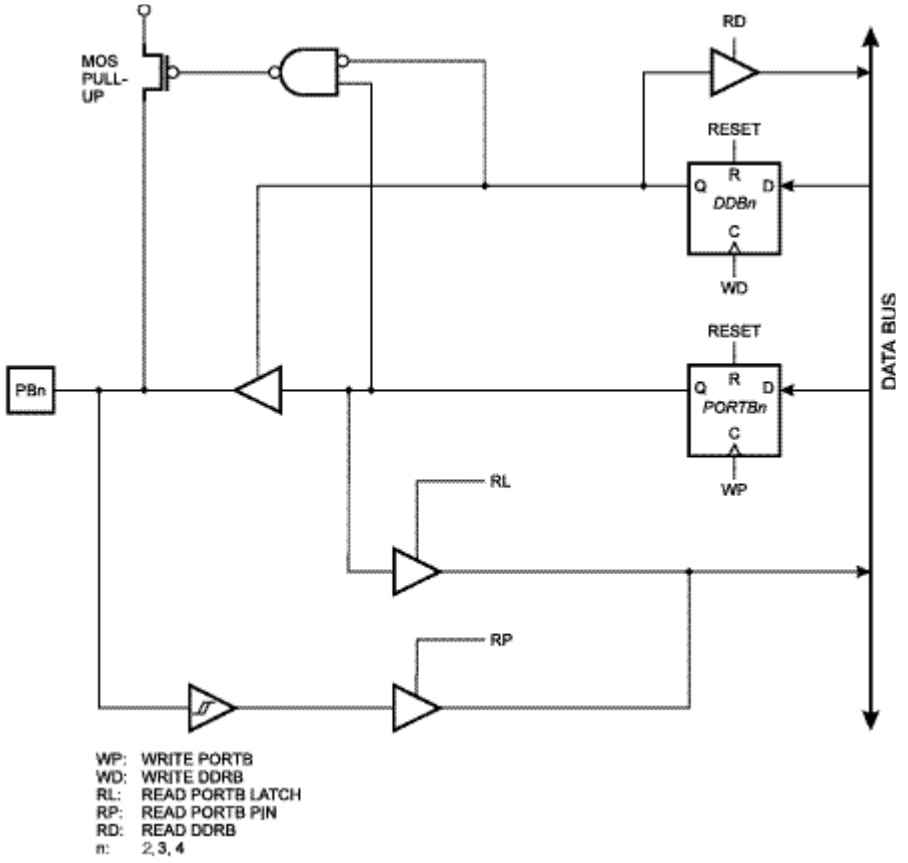


图 24　B 口示意图（PB5）

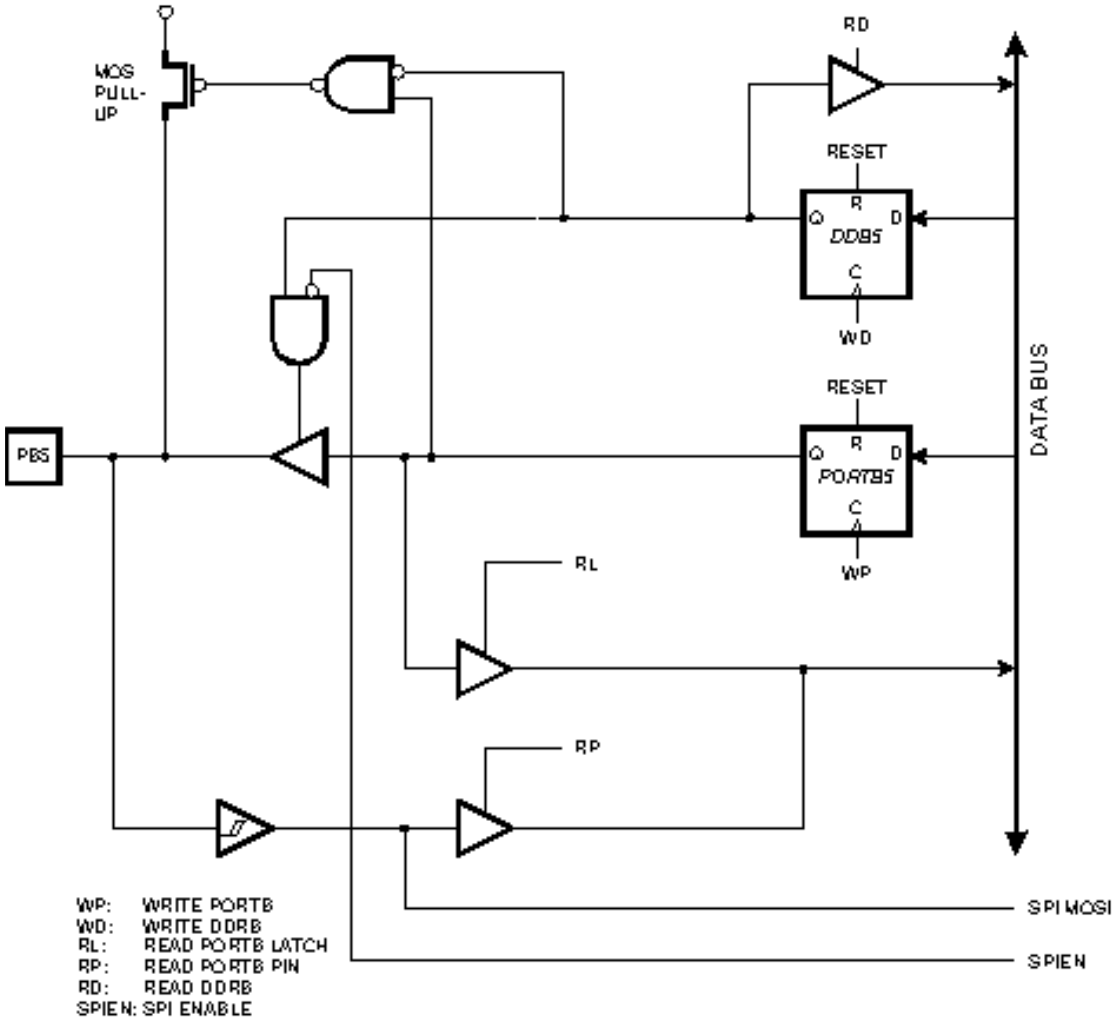


图 25　B 口示意图（PB6）

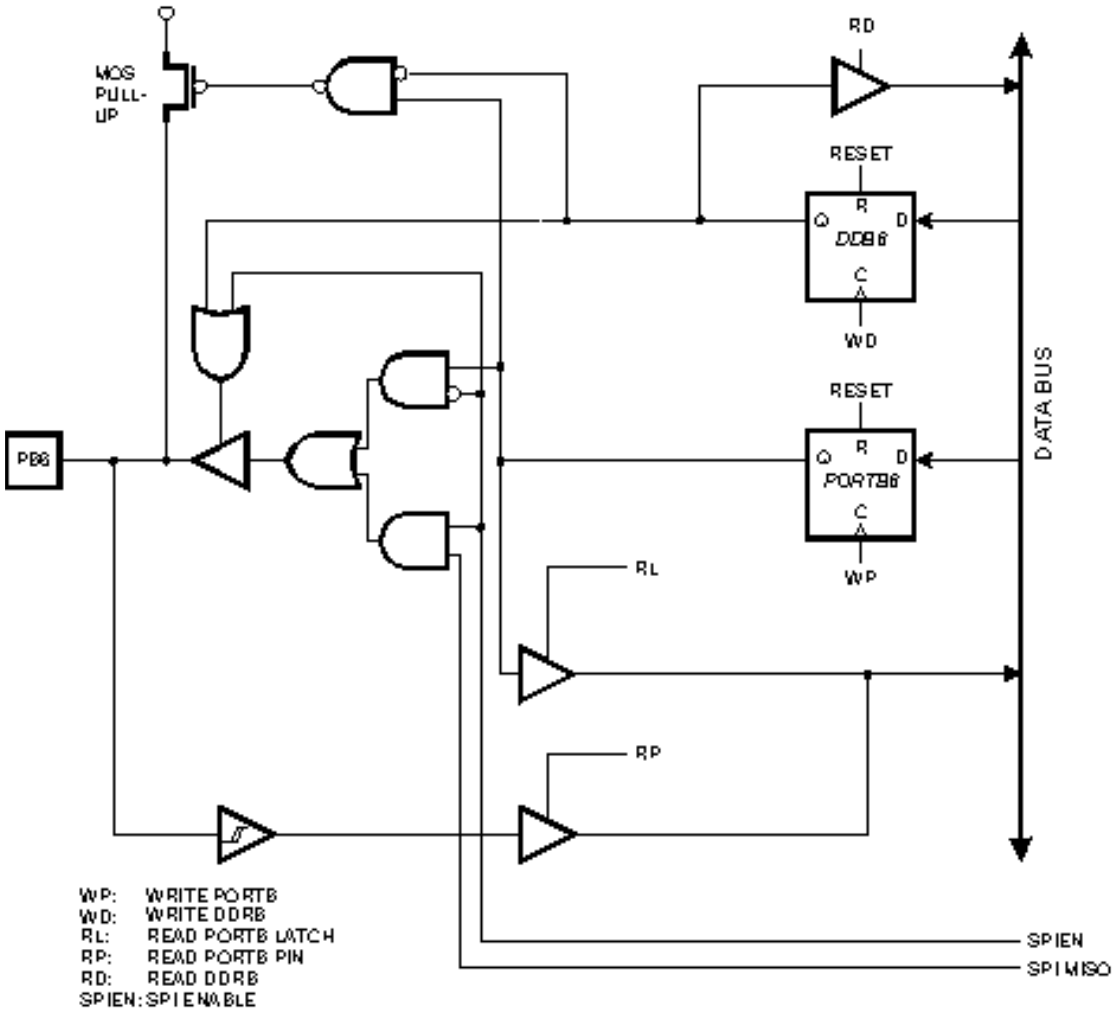
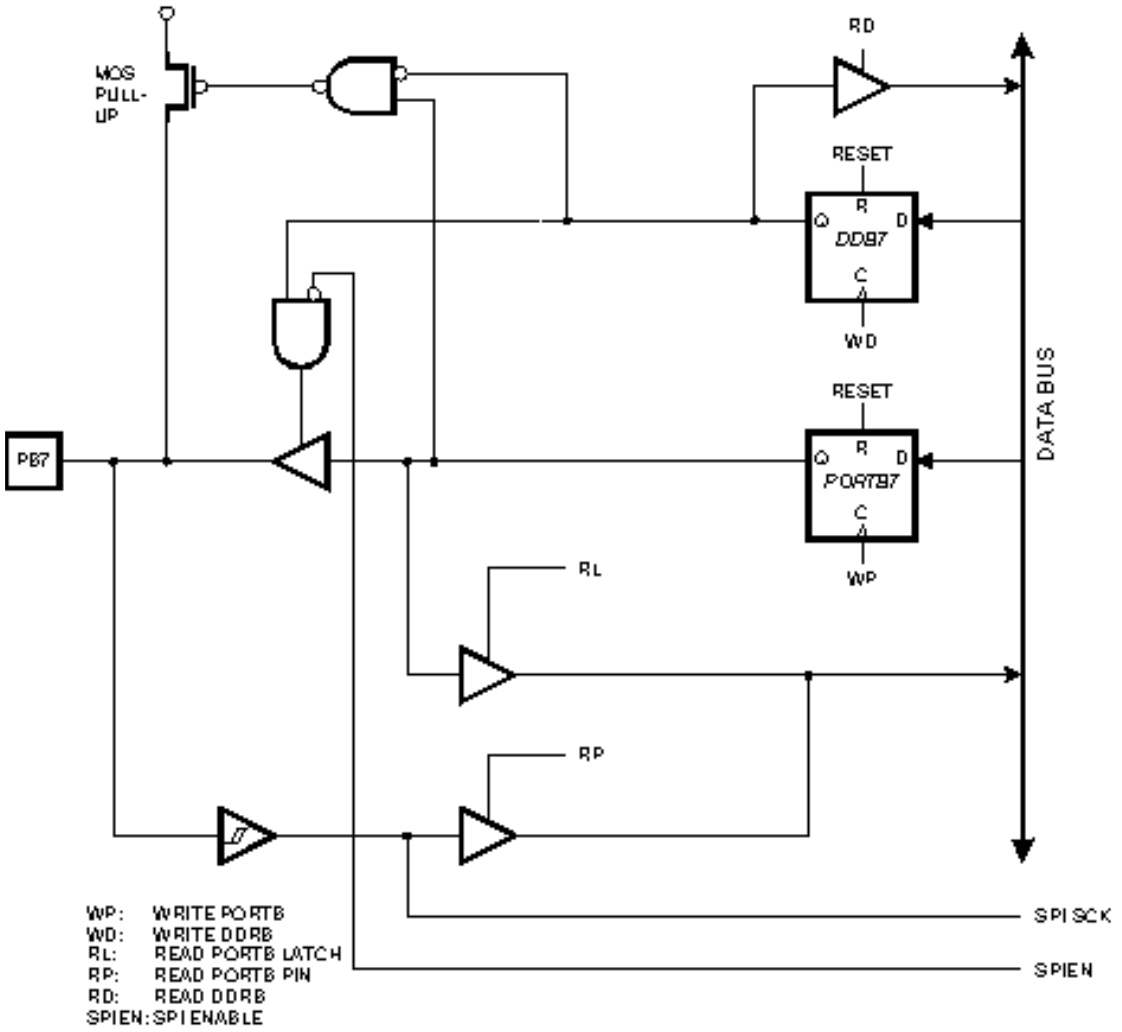


图 26　B 口示意图（PB7）



D 口

D 口有 3 个 I/O 地址：数据寄存器—PORTD（$12），数据方向寄存器—DDRD（$11）和输入引脚—PIND（$10）。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 10　D 口第二功能

管脚	第二功能
PD2	INT0（外部中断 0 输入）
PD4	T0（T/C0 的外部输入）

D 口数据寄存器—PORTD

BIT	7	6	5	4	3	2	1	0
$12	-	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	-	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	-	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 9　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

- T0—PD4
  T/C0 **的时钟源**
- INT0—PD2
  外部中断源

D 口示意图

图 27　D **口示意图**（PD0、PD1、PD3、PD5 **和** PD6）

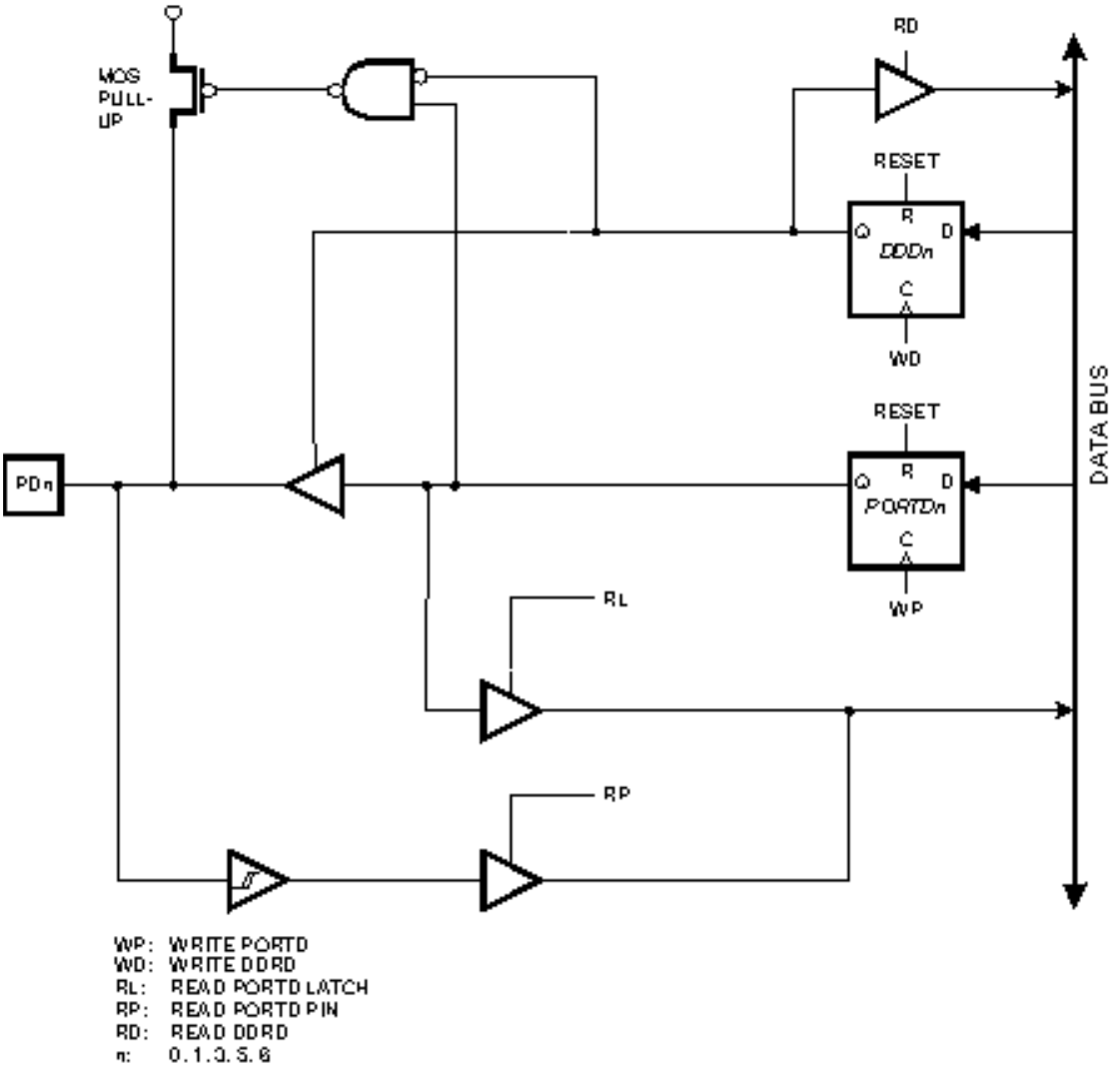


图 28　D 口示意图（PD2）

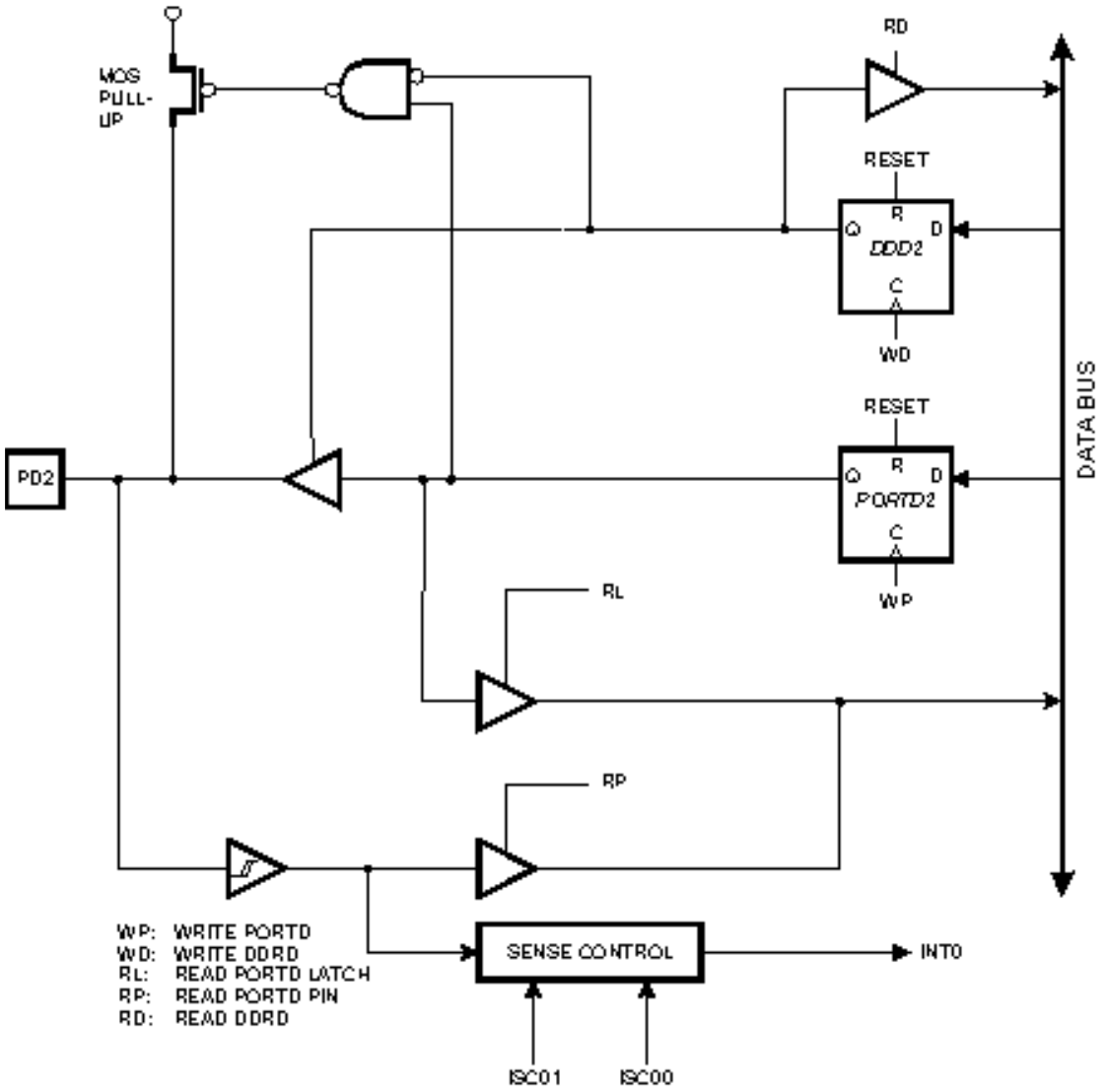
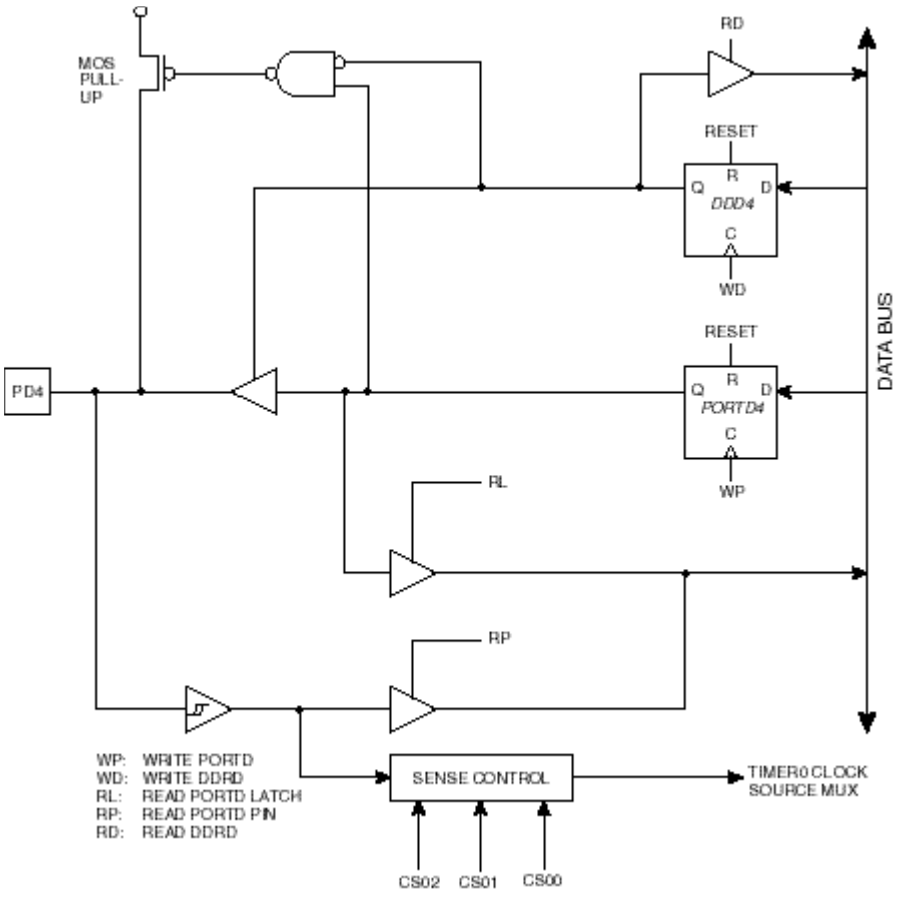


图 29　D 口示意图（PD4）



程序编程

程序和数据锁定位

AT90S1200 具有两个锁定位，如表 12 所示。锁定位只能通过片擦除命令擦除。

表 12　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：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

AT90S1200 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”。
- RCEN 编程（为“0”）后，MCU 以内部 RC 振荡器为时钟。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：$90（1K 字节的 FLASH）

3、$002：$01（当$01 地址为$90 时，器件为 1200）

注：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

AT90S1200 具有 1K 字节的片内可编程 FLASH 和 64 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。

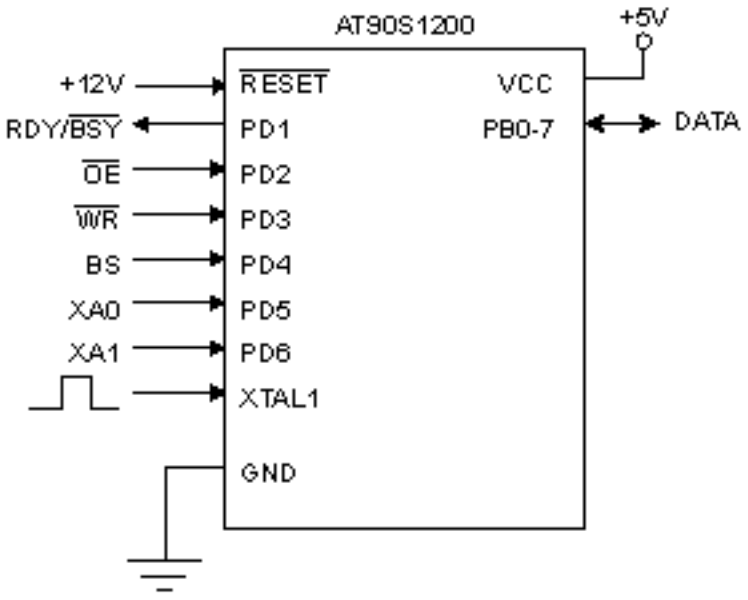
在两种编程模式下，FLASH 是以字节的形式写入的。而对于 EEPROM，片内集成了自擦和自定时除功能。在编程时，要注意电源电压要满足要求。

表 13 编程电源电压

型号	串行编程	并行编程
AT90S1200	2.7V – 6.0V	4.5V – 5.5V

并行编程

图 30 并行编程



信号命名：

AT90S1200 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 14　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 15
XA1	PD6	I	见表 15
DATA	PB0-7	I/O	双向数据总线（/OE 为低时输出）

表 15　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 16　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 13 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”**一使能命令加载**

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（由于只有** 1K **字节，故为**$00 **或**$01）

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 31）

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 32）

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 31 编程波形一

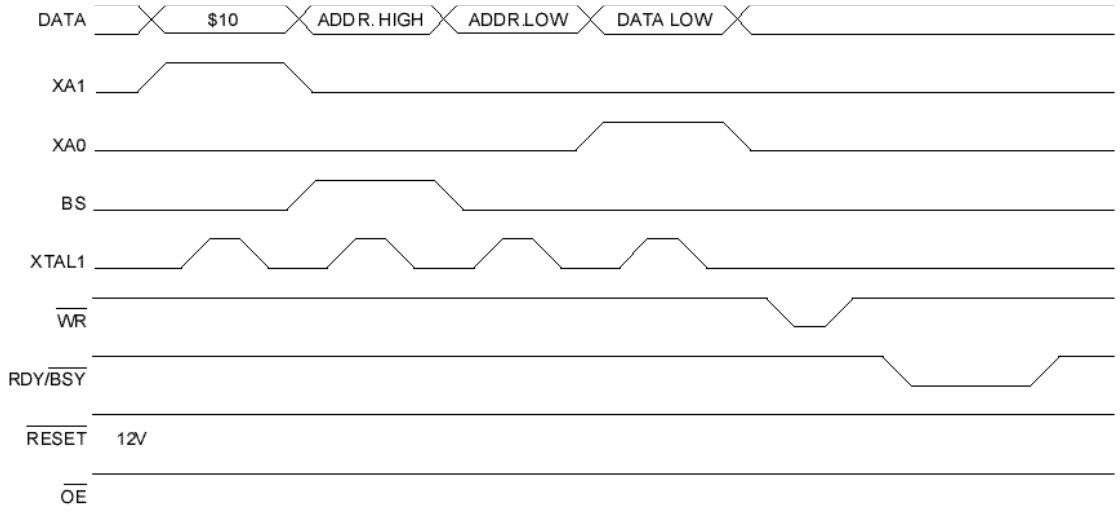
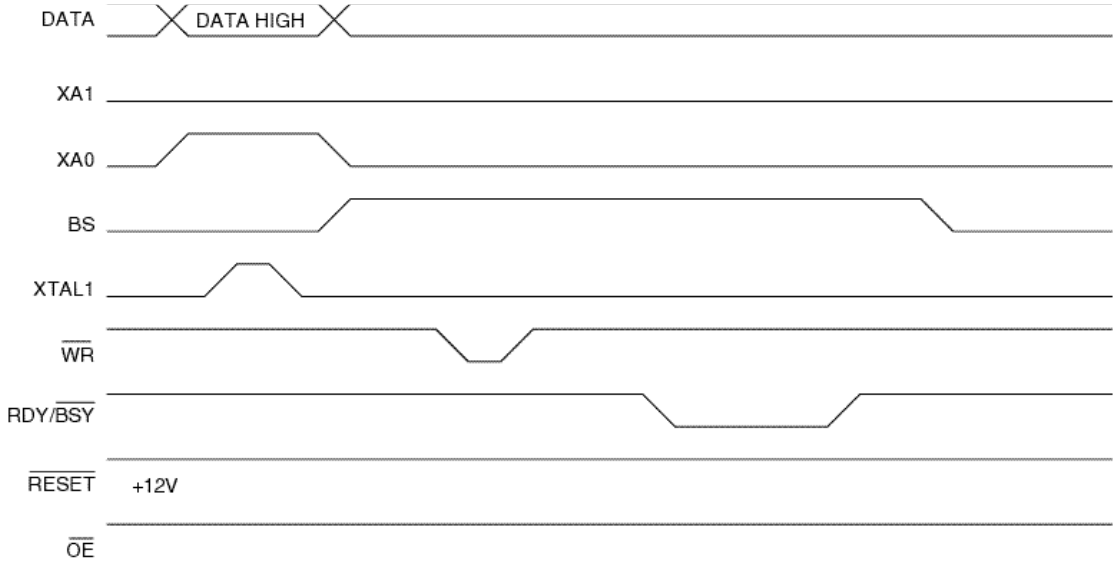


图 32 编程波形二



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 = RCEN

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 = RCEN

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 33　并行编程时序

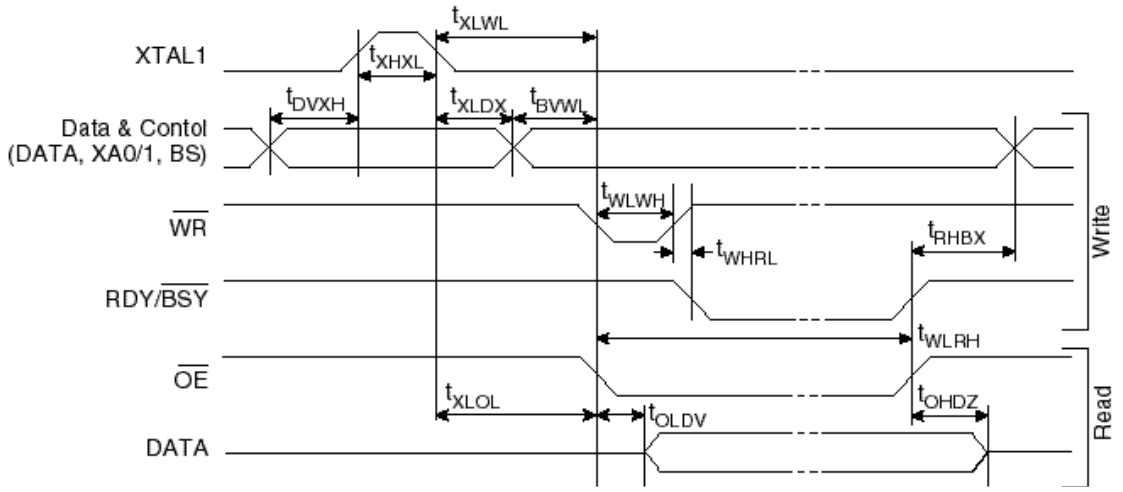


表 17　并行编程特性　T_A = 25℃±10%，V_{CC} = 5V±10%

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_{DVHX}	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

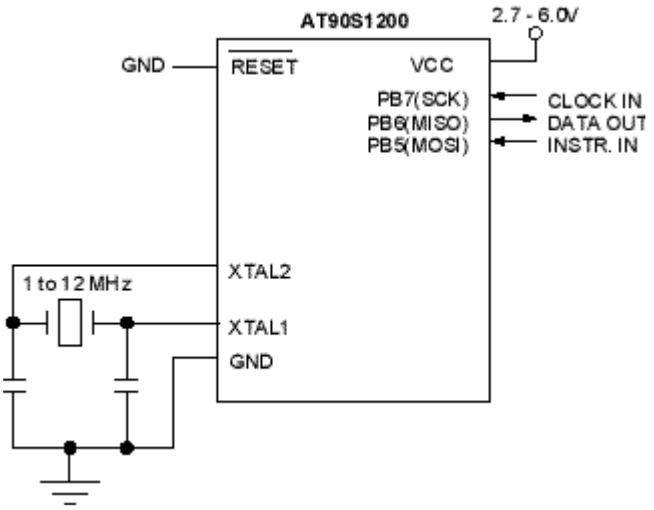
注：1、**在芯片擦除时使用** t_{WLWH_CE}，**而在编程锁定位是使用** t_{WLWH_PFB}。

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 34　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~$01FF，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：> 1 个 XTAL1 时钟

高：> 4 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 AT90S1200，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。

4、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。

5、任意一个内存地址都可以用读指令在 MISO（PB6）读出。

编程结束后，可以把/RESET 拉高，进入正常工作模式。

6、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 18）。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 18　EEPROM 数据检测返回值

型号	P1	P2
AT90S1200	$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 35　串行编程波形

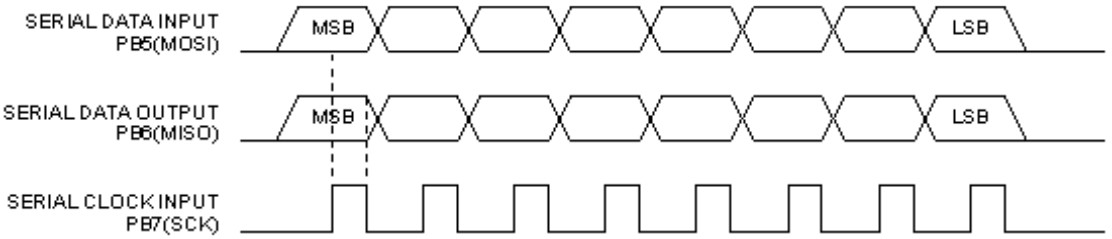


表 19　AT90S1200 的串行编程指令

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 000**a**	**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 000**a**	**bbbb bbbb**	**iiii iiii**	写 H（高或低）字节 i 到字地址 a:b
读 EEPROM	1010 0000	0000 0000	00**bb 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	1111 1***2***1	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

注意：厂标不能在锁定模式 3 下读出。

串行编程电特性

图 36 串行编程时序

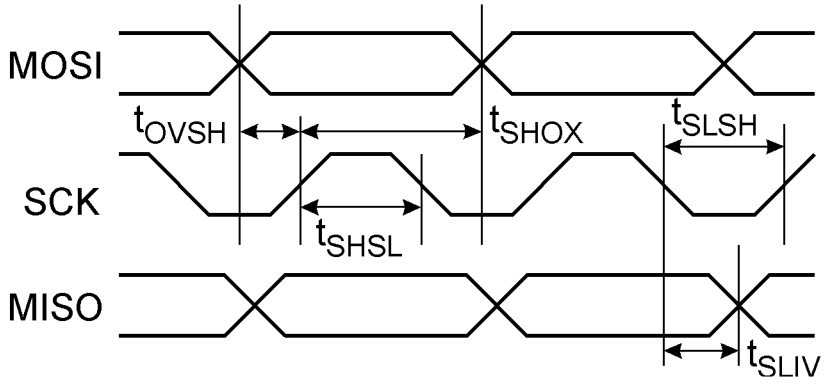


表 20 串行编程电特性，T_A = -40℃到 85℃，V_{CC} = 2.7V – 6.0V

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		12	MHz
t_{CLCL}	振荡周期（V_{CC} = 4.0V – 6.0V）	83.3			ns
t_{SHSL}	SCK 高	4 t_{CLCL}			ns
t_{SLSH}	SCK 低	t_{CLCL}			ns
t_{OVSH}	MOSI Setup to SCK High	1.25 t_{CLCL}			ns
t_{SHOX}	MOSI Hold after SCK High	2.5 t_{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			0.6 0.5	V
V_{OH}	输出高电压 (4) B、D 口	I_{OH} =20mA，V_{CC} = 5V I_{OH} =10mA，V_{CC} = 3V	4.3 2.3			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	V_{CC} = 6V，pin low			980	nA

	流 I/O 脚					
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

4、 所有管脚的 I_{OH} 之和不能超过 200mA

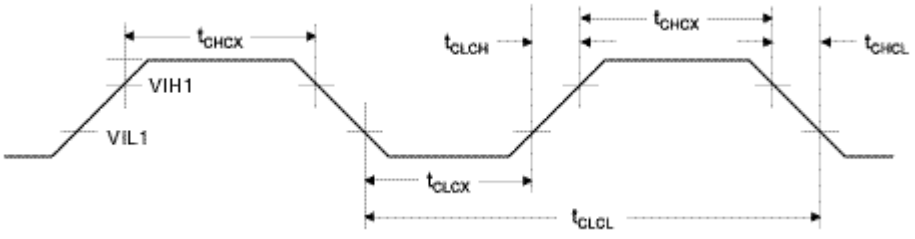
D0 – D5 和 ZTAL2 的 I_{OH} 之和不能超过 100mA

B0 – B7 和 D6 的 I_{OH} 之和不能超过 100mA

5、 掉电时的最小 V_{CC} 为 2V

外部时钟驱动波形

图 37　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	12	MHz
t_{CLCL}	时钟周期	250		83.3		ns
t_{CHCX}	高电平时间	100		83.3		ns
t_{CLCX}	低电平时间	100		83.3		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 38　工作电流与频率的关系

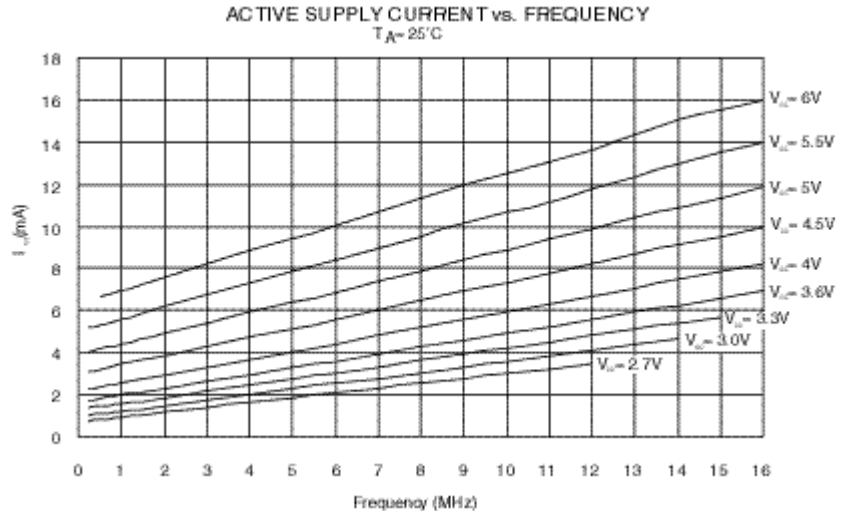


图 39　工作电流与电压的关系

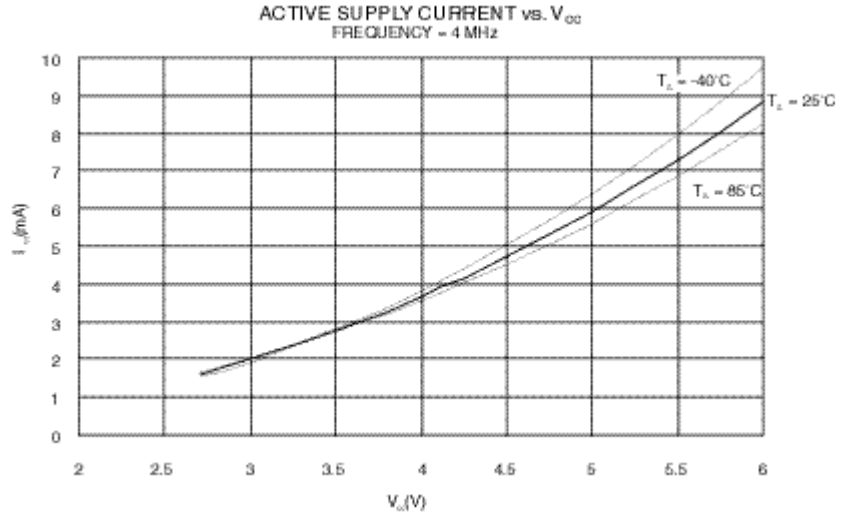


图 40　工作电流与电压的关系，内部振荡器

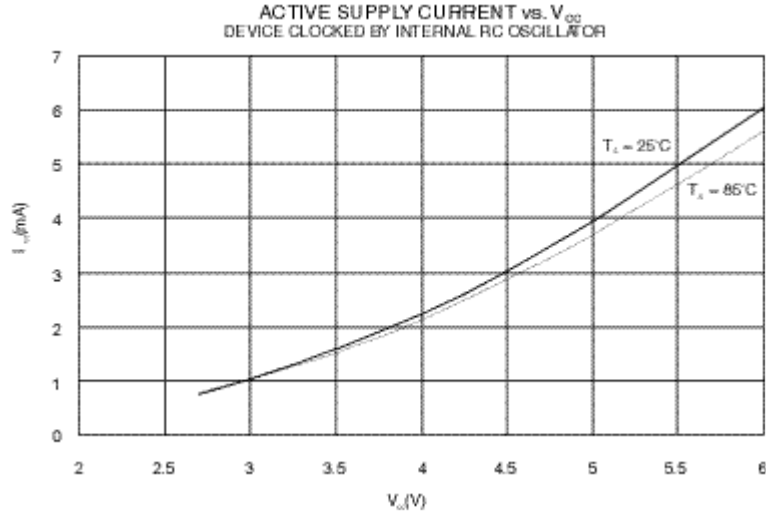


图 41　工作电流与频率的关系，闲置模式

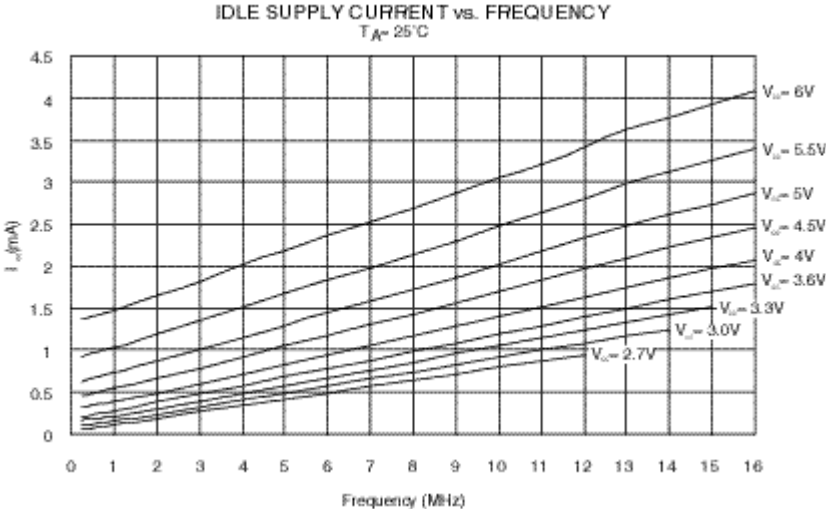


图 42　工作电流与电压的关系，闲置模式

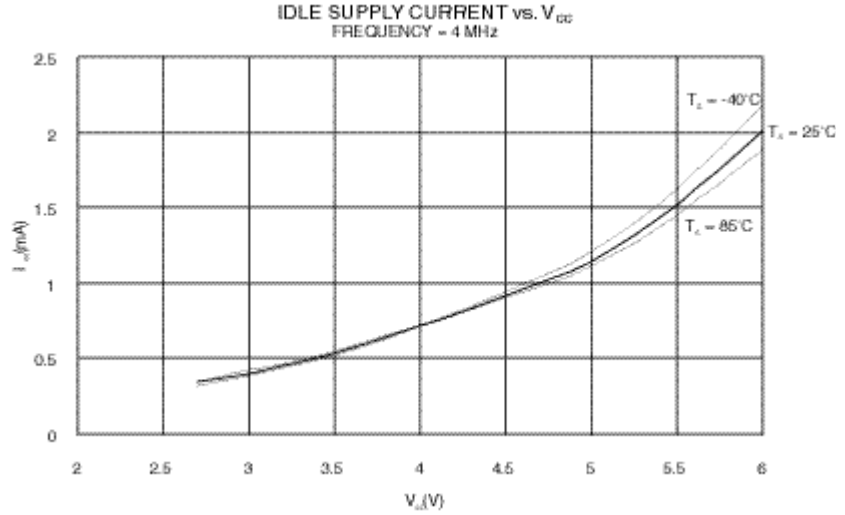


图 43　工作电流与电压的关系，闲置模式，内部振荡器

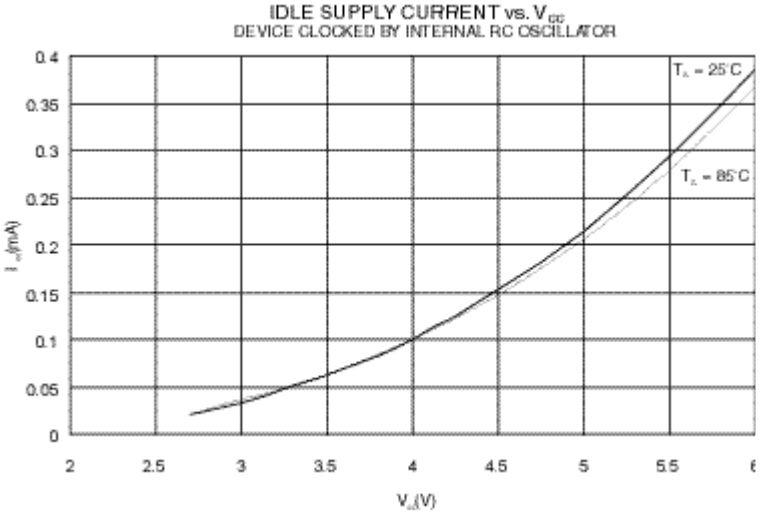


图 44　工作电流与电压的关系，掉电模式，看门狗禁止

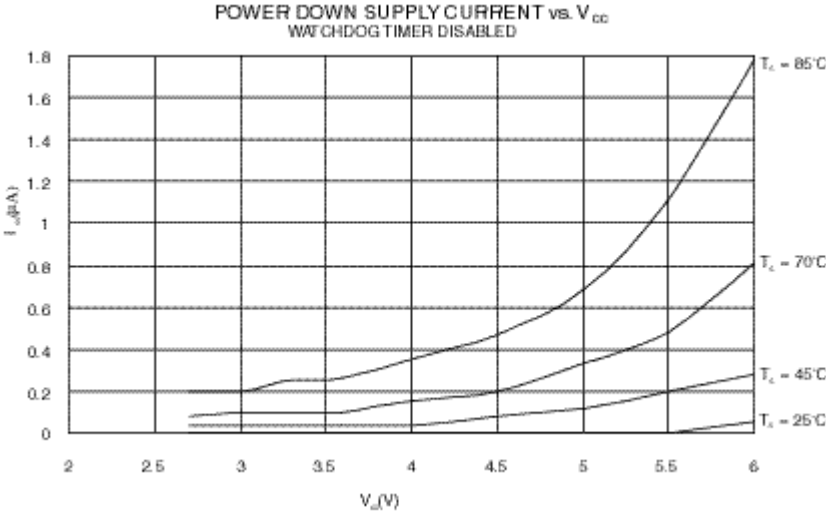


图 45　工作电流与电压的关系，掉电模式，看门狗使能

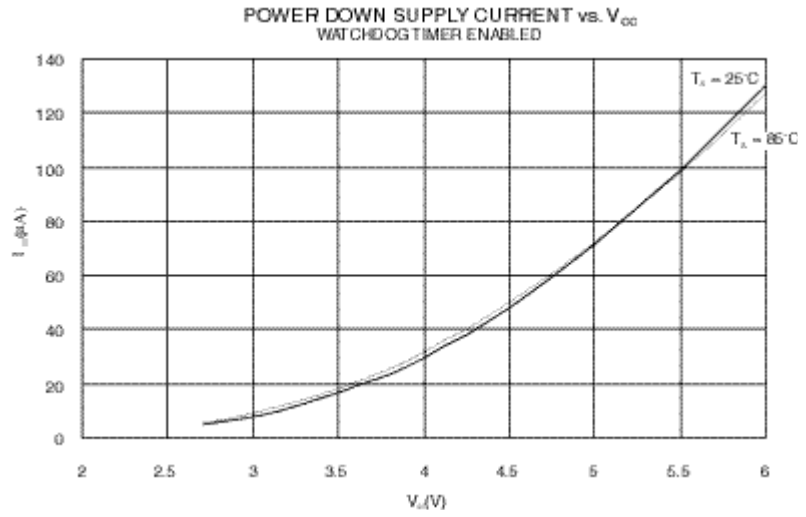


图 46　片内 RC 振荡器频率与电压的关系

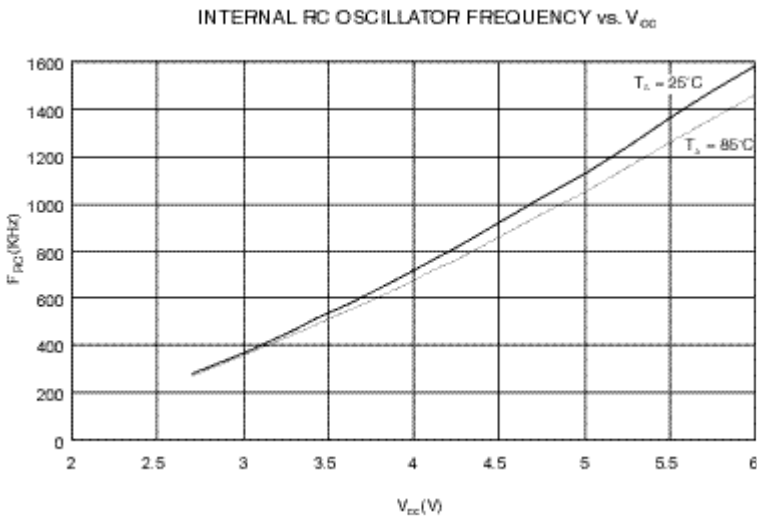


图 47　模拟比较器电流与电压的关系

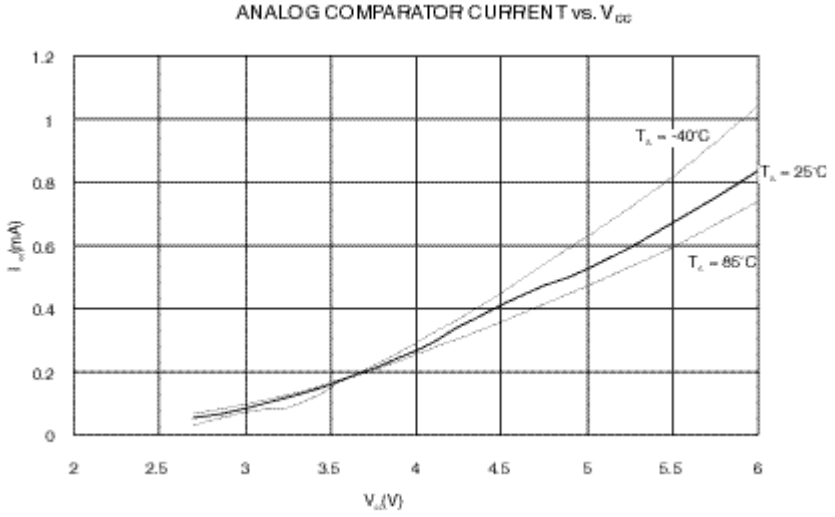


图 48　模拟比较器偏置电压与共模电压的关系， V_{CC}=5V

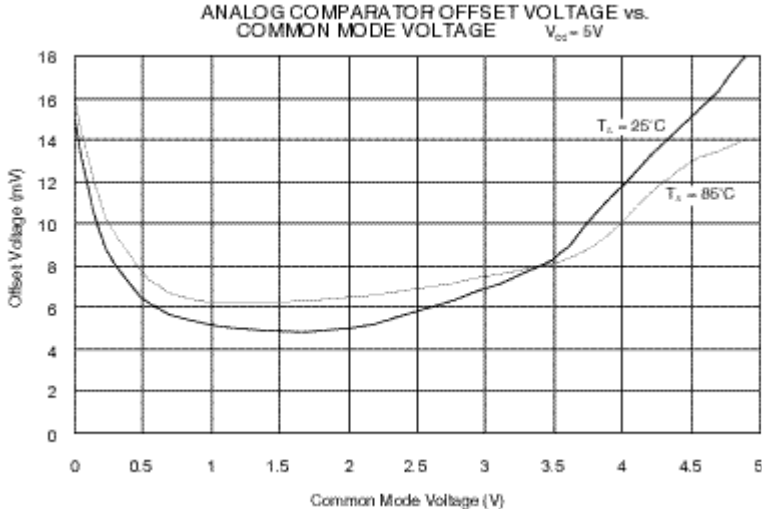


图 49　　模拟比较器偏置电压与共模电压的关系

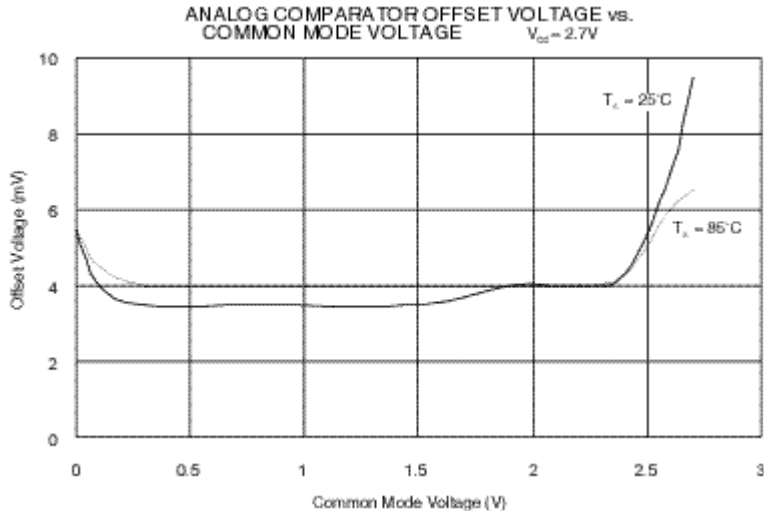


图 50 模拟比较器输入泄漏电流

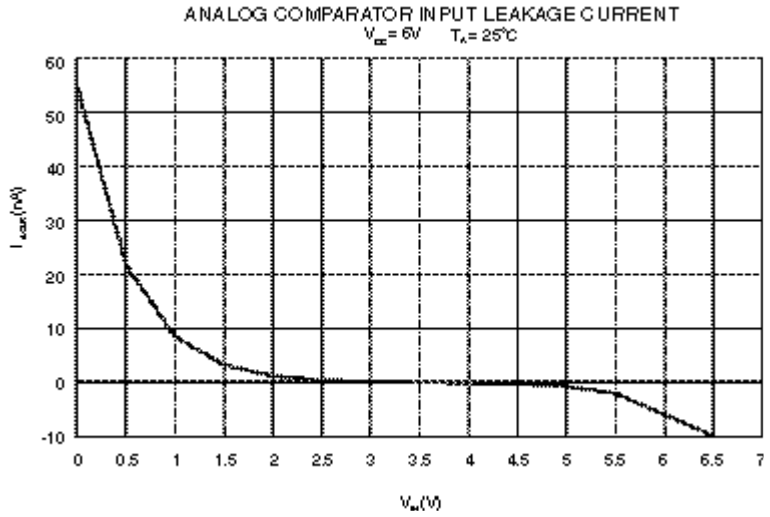


图 51　上拉电阻电流与输入电压的关系

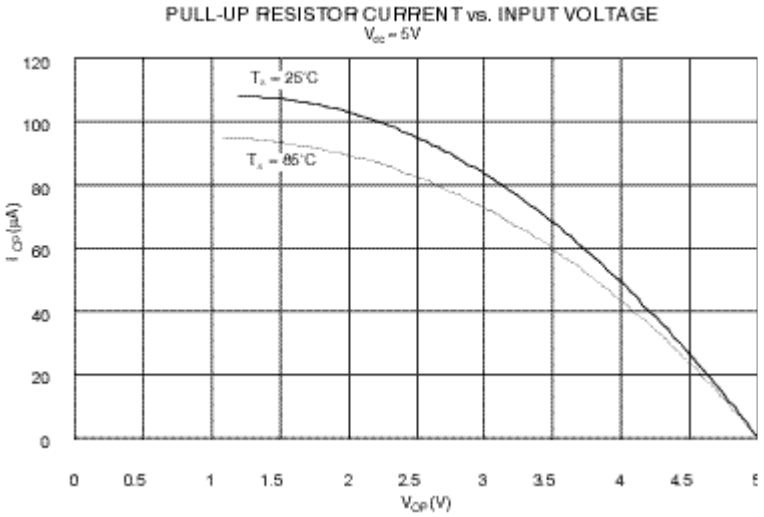


图 52　上拉电阻电流与输入电压的关系

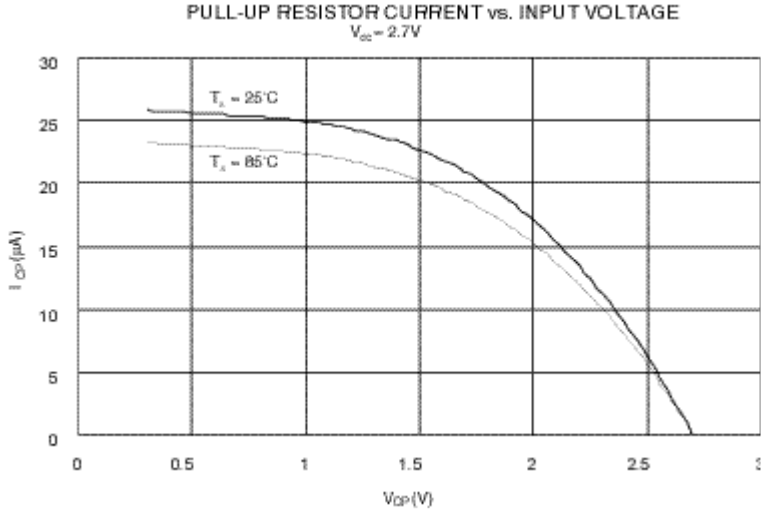


图 53　I/O 引脚吸入电流与输出电压的关系

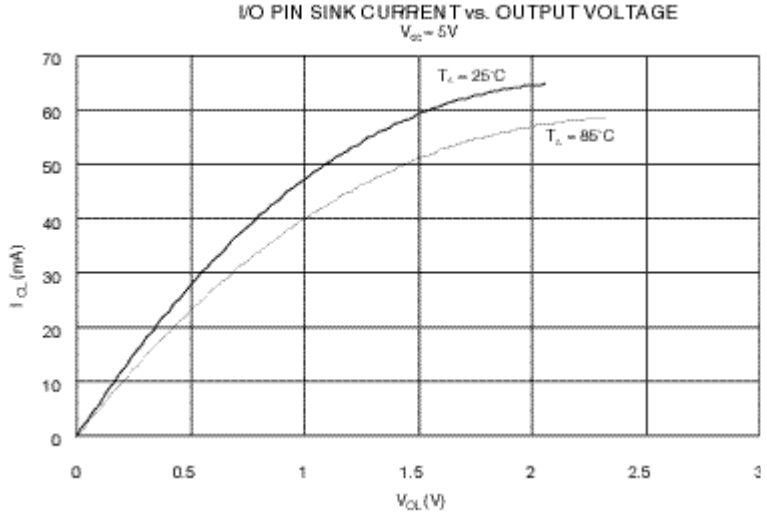


图 54　I/O 引脚输出电流与输出电压的关系

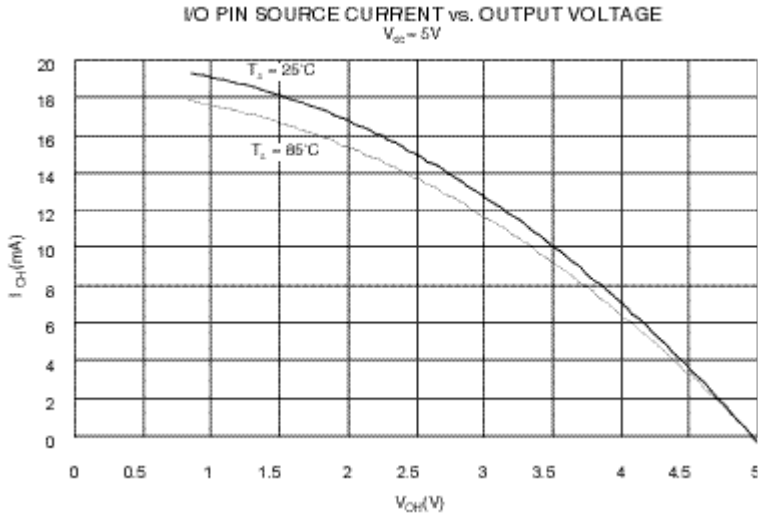


图 55　I/O 引脚吸入电流与输出电压的关系

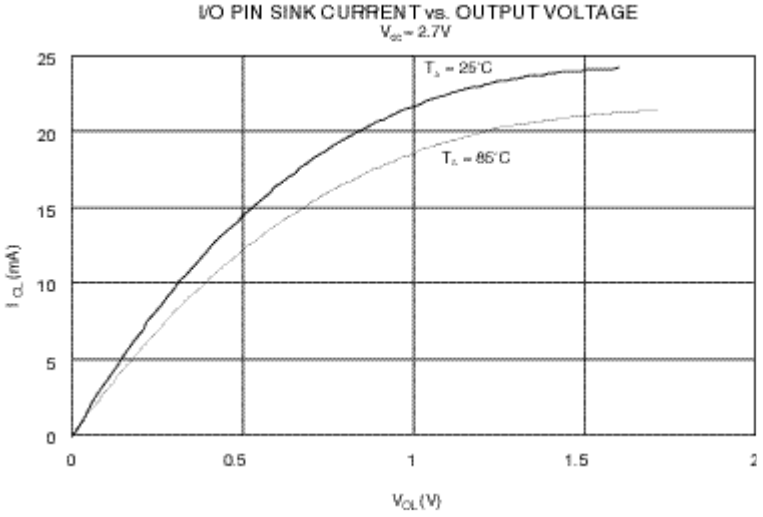


图 56　I/O 引脚输出电流与输出电压的关系

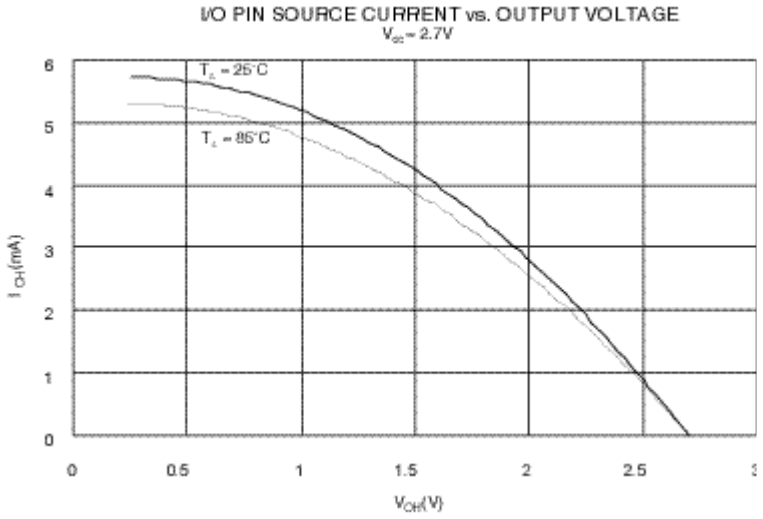


图 57　I/O 引脚输出门限电压与电压的关系

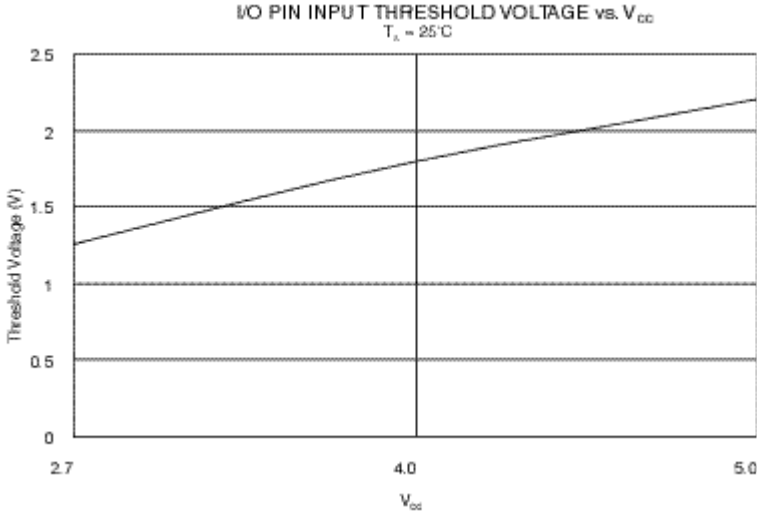
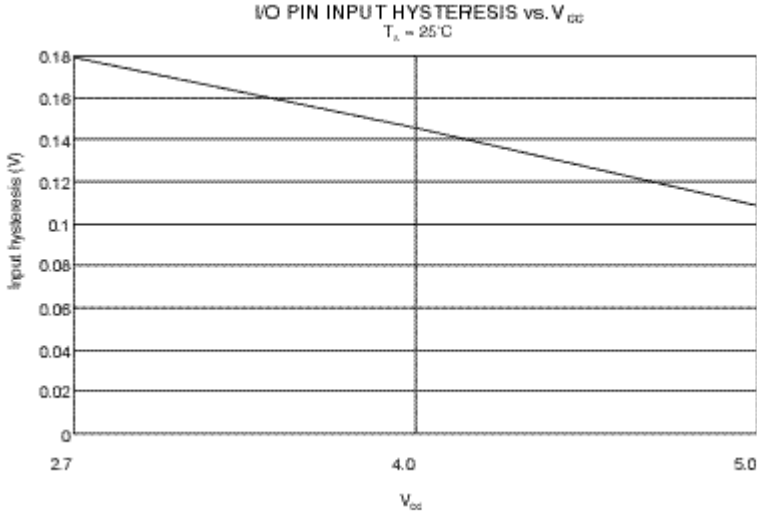


图 58　I/O 引脚输入容限与电压的关系



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——寄存器堆或 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
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
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	取反	Rd ← $FF - Rd	Z,C,N,V,S	1
NEG	Rd	取补码	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 1	Rd ← Rd + 1	Z, N,V,S	1
DEC	Rd	减 1	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
RCALL	k	相对调用	PC ← PC + k + 1	-	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					
LD	Rd，Z	间接取数	Rd ← （Z）	-	2
ST	Z，Rd	间接存数	（Z）← Rd	-	2
MOV	Rd，Rr	拷贝寄存器	Rd ← Rr	-	1
LDI	Rd，**K**	取立即数	Rd ← **K**	-	1
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
位及位测试指令					
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
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	高低半字节交换	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
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	AT90S1200 – 4PC AT90S1200 – 4SC AT90S1200 – 4YC	20P3 20S 20Y	商用 （0℃ - 70℃）
		AT90S1200 – 4PI AT90S1200 – 4SI	20P3 20S	工业

		AT90S1200 – 4YI	20Y	（-40℃ - 85℃）
12	4.0 – 6.0	AT90S1200 – 12PC AT90S1200 – 12SC AT90S1200 – 12YC	20P3 20S 20Y	**商用** （0℃ - 70℃）
		AT90S1200 – 12PI AT90S1200 – 12SI AT90S1200 – 12YI	20P3 20S 20Y	**工业** （-40℃ - 85℃）

注意： AT90S1200A – XXX **为** RCEN **已编程器件的定货号**

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）
20Y	20 **脚，** 5.3 **毫米，塑料缩小小尺寸**（SSOP）

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.）