

AVR 单片机特点

每种 MCU 都有自身的优点与缺点，与其它 8-bit MCU 相比，AVR 8-bit MCU 最大的特点是：

- 哈佛结构，具备 1MIPS / MHz 的高速运行处理能力；
- 超功能精简指令集（RISC），具有 32 个通用工作寄存器，克服了如 8051 MCU 采用单一 ACC 进行处理造成的瓶颈现象；
- 快速的存取寄存器组、单周期指令系统，大大优化了目标代码的大小、执行效率，部分型号 FLASH 非常大，特别适用于使用高级语言进行开发；
- 作输出时与 PIC 的 HI/LOW 相同，可输出 40mA（单一输出），作输入时可设置为三态高阻抗输入或带上拉电阻输入，具备 10mA-20mA 灌电流的能力；
- 片内集成多种频率的 RC 振荡器、上电自动复位、看门狗、启动延时等功能，外围电路更加简单，系统更加稳定可靠；
- 大部分 AVR 片上资源丰富：带 E2PROM, PWM, RTC, SPI, UART, TWI, ISP, AD, Analog Comparator, WDT 等；
- 大部分 AVR 除了有 ISP 功能外，还有 IAP 功能，方便升级或销毁应用程序。
- 性价比高。

开发 AVR 单片机，需要哪些编译器、调试器？

软件名称	类型	简介	官方网址
AVR Studio	IDE、汇编编译器	ATMEL AVR Studio 集成开发环境(IDE)，可使用汇编语言进行开发（使用其它语言需第三方软件协助），集软硬件仿真、调试、下载编程于一体。ATMEL 官方及市面上通用的 AVR 开发工具都支持 AVRStudio。	www.atmel.com
GCCAVR (WinAVR)	C 编译器	GCC 是 Linux 的唯一开发语言。GCC 的编译器优化程度可以说是目前世界上民用软件中做的最好的，另外，它有一个非常大优点是，免费！在国外，使用它的人几乎是最多的。但，相对而言，它的缺点是，使用操作较为麻烦。	sourceforge.net
ICC AVR	C 编译器（集烧写程序功能）	市面上(大陆)的教科书使用它作为例程的较多，集成代码生成向导，虽然它的各方面性能均不是特别突出，但使用较为方便。虽然 ICCAVR 软件不是免费的，但它有 Demo 版本，在 45 天内是完全版。	www.imagecraft.com
CodeVision AVR	C 编译器（集烧写程序功能）	与 Keil C51 的代码风格最为相似，集成较多常用外围器件的操作函数，集成代码生成向导，有软件模块，不是免费软件，Demo 版为限 2KB 版。	www.hpinfofotech.ro
ATman AVR	C 编译器	支持多个模块调试(AVRStudio 不支持多个模块调试)。	www.atmanecl.com
IAR AVR	C 编译器	IAR 实际上在国外比较多人使用，但它的价格较为昂贵，所以，中国大陆内，使用它的开发人员较少，只有习惯用 IAR 的工程师才会去使用它。	www.iar.com

AVR 的仿真方式

一般来说，AVR 有三种仿真方式：

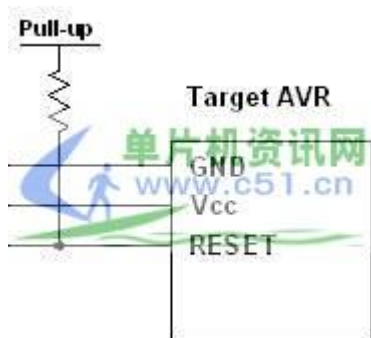
(1) JTAG 仿真方式，适用于具备 JTAG 仿真接口的 AVR。如：Atmega16/32，Atmega64/128 等。

JTAG 是 IEEE 的标准规范，通过这个标准，可对具有 JTAG 接口的芯片的硬件电路进行边界扫描和故障检测。部分 AVR 型号带 JTAG 仿真调试接口，可使用 JTAG 仿真方式。

(2) debugWIRE 仿真方式，适用于具备 debugWIRE 仿真接口的 AVR。如：Attiny13/24/2313，Atmega48/88/168 等。

debugWIRE 是用以降低成本和调试引脚的开销，ATMEL 在 AVR 器件上使用的新的调试接口：debugWIRE，与 JTAG 相比其主要区别在于仅使用一根信号线

(RESET)，即可完成调试信息的交互，达到控制程序流向，执行指令以及编程熔丝位的功能。它的总的连接图如下：



这里的 RESET 信号被用于传递调试信息。

(3) 采用仿真头替代 AVR MCU 仿真方式，适用于不带仿真接口的 AVR。如 Attiny26，Atmega8，Atmega8515 等

AVR 单片机基本硬件电路设计包括：AVR 复位电路和下载电路的设计，另外 AVR 晶振电路可以不加。

- AVR 复位电路的设计

与传统的 51 单片机相比，AVR 单片机内置复位电路，并且在熔丝位里，可以控制复位时间，所以，AVR 单片机可以不设外部上电复位电路，依然可以正常复位，稳定工作。

若是系统需要设置按键复位电路，那么注意，AVR 单片机是低电平复位，如下图，设计按键复位电路：



- AVR 下载电路的设计

一般来说, AVR 的编程方式有:

- (1) 串行编程, ISP 编程
- (2) 高压/并行编程
- (3) JTAG 编程
- (4) IAP 编程

一般情况, 系统板都需要设计下载线路, 对 AVR 进行编程。目前的 AVR 芯片基本上都具备 ISP 接口, 可通过 ISP 接口进行编程。所以, 最常见的是, 在系统板上留 ISP 接口。

那么什么是 ISP 呢?

ISP 是 In System Program 的缩写, 意思是在系统编程, 亦即是在线编程。它一共使用了两条电源线: VCC、GND, 三条信号线: SCK、MOSI、MISO, 以及复位线: RESET。由于仅仅使用了几条数据线, 所以我们亦常将其称为串行编程。

值得注意的是: 大部分 AVR 的 ISP 数据端口亦为 SCK、MOSI、MISO 引脚(如 tiny13/24/2313, mega48/88/168/8, mega16/32/162 等), 如下:

[调试器] [目标 MCU]

VCC ----- VCC

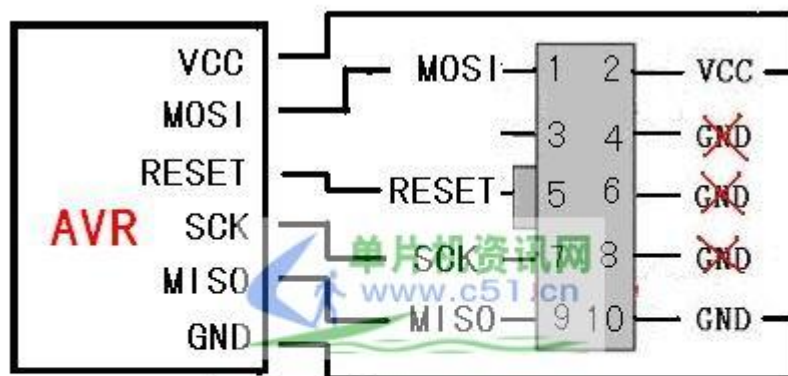
GND ----- GND

RESET ----- RESET

SCK ----- SCK

MOSI ----- MOSI

MISO ----- MISO



ATmega48/88/168/8,

ATmega16/32/162,

ATtiny13/24/26/2313.

• AVR 晶振电路的设计

与传统的 51 单片机相比, AVR 单片机内置 RC 振荡电路。出厂时, 未进行时钟源设置的 AVR, 其时钟源使用的是内部 RC 振荡, 一般情况使用的是 1M 频率。

通过对熔丝位的设置, 可以设置 MCU 的内部 RC 振荡频率。例如: 4M、8M 等。

不过, 内置 RC 振荡, 在一致性方面存在差异, 它因生产的批次有所差异, 亦与温度等因素有较大的相关性。所以, 在一些对时钟要求较高的场合, 如: 精确定时, RS232 通信等, 这些场合, 建议使用外部的晶振线路。

