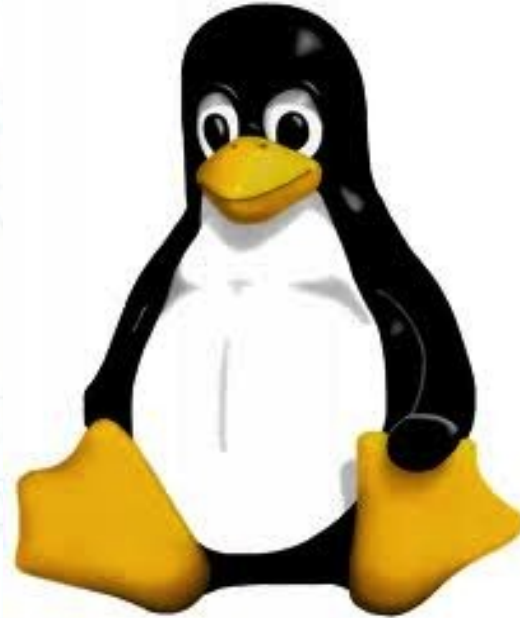


AVR Microcontrollers for the Linux User



Hardware Details

--pick the chip for your application--

8-bit *(Not counting the AVR32 series)*

Up to ~100 pins

Harvard architecture

32 General purpose registers



Internal oscillators (up to 20MHz)

1 to 256kb of code

1.8 to 5.5v operation (down to 650nA)

What You Will Need

Chips: Mouser, Digikey... (~ \$0.90 and up)

 Enlarge	556-ATTINY45-20PU	ATTiny45-20PU	Atmel	Microcontrollers (MCU) 4kB Flash 0.256kB EEPROM 6 I/O Pins	Page 191 Data Sheet	1,341 In Stock	1: \$1.87 10: \$1.44 25: \$1.33 100: \$1.32 More: Get Quote	Buy Min.: 1 Mult.: 1	 Details
--	-------------------	---------------	-------	--	------------------------	-------------------	--	--	--

Programmer: Sparkfun, adafruit, ebay... (~ \$20 and up)



▲ PGM-08702 **RoHS** ✓
STK500 Compatible USB Programmer

\$47.95

1

[Add to Cart](#)

[Add to Wish List](#)

Programming Choices

Arduino

- Chip needs bootloader

- Language is similar to C/C++

Discrete AVR

- Needs separate programmer or ISP

- Uses C or Assembly

 - `avr-gcc`, `avr-binutils`, `avr-libc`

C Example: Blink a LED

```
#include <avr/io.h>
#include <util/delay.h>

int main (void) {
    DDRB = 0xFF; // set PORTB for
Output
    while (1) { // loop forever...
        PORTB = 0x20; // set PORTB 6 high
        _delay_ms(1000); // wait
        PORTB = 0x04; // set PORTB 6
Low
        _delay_ms(1000); // wait
    }
    return 1; }
```

ASM Example: Blink a LED

```
#include <avr/io.h>
delay:
    * * * delay loop, or call to util/delay.h goes
here ***
    ret

.global main
main:
    sbi _SFR_IO_ADDR(DDRB), 0xFF
loop:
    sbi _SFR_IO_ADDR(PORTB), 0x20
    rcall delay
    cbi _SFR_IO_ADDR(PORTB), 0x20
    rcall delay
    rjmp loop
.end
```

That Delay Loop Thing...

```
/* delay 393,219 cycles (0.393 sec @ 1  
MHz) see: bit.ly/gbhxWo */
```

```
delay:
```

```
clr r1
```

```
clr r2
```

```
ldi r16, 2
```

```
dsh:
```

```
dec r1
```

```
brne dsh
```

```
dec r2
```

```
brne dsh
```

```
dec r16
```

```
brne dsh
```

```
ret
```

Build & Burn

Compile

```
avr-gcc -g -Os -mcall-prologues -Wall -mmcu=attiny45 my-file.c
```

Convert

```
avr-objcopy -R .eeprom -O ihex my-file.obj my-file.hex
```

Burn

assuming you are happy with the default fuse settings...

```
avrdude -p t45 -b 115200 -c stk500v2 -P /dev/ttyX -U flash:w:myfile.hex
```

Projects

