

Optiboot - ATmega328P at custom frequency

Pieter P

This is a guide to use the ATmega328P (the microcontroller of an Arduino UNO) with a custom crystal, and run it at a different frequency than the standard 16 MHz.

Downloading Optiboot

The Arduino has a bootloader in order to be able to upload sketches over the UART (serial) connection. When running at a different clock speed, you have to recompile the bootloader. To do this, you need Optiboot.

First, install git (if you haven't already):

```
$ sudo apt-get install git
```

Then clone Optiboot:

```
$ git clone https://github.com/Optiboot/optiboot.git
```

Next, go to the 'bootloaders' folder of your Arduino IDE installation, for example:

```
$ cd ~/opt/arduino-1.8.1/hardware/arduino/avr/bootloaders
```

Rename the old 'optiboot' folder:

```
$ mv optiboot optiboot.old
```

And replace it by the new version of Optiboot you just cloned:

```
$ cp -R ~/optiboot/optiboot/bootloaders/optiboot/ optiboot && cd optiboot
```

Compiling the Optiboot bootloader

Open the Makefile to add a new option with a custom frequency:

```
$ gedit Makefile
```

Then duplicate the atmega328 and atmega328_isp options, and rename them, for example, atmega328_12 and atmega328_12_isp. Change the CPU speeds, and save the file. The result could be something like this:

```
#Atmega328p @12MHz
atmega328_12: TARGET = atmega328
atmega328_12: MCU_TARGET = atmega328p
atmega328_12: CFLAGS += $(COMMON_OPTIONS)
atmega328_12: AVR_FREQ = 12000000L
atmega328_12: LDSECTIONS = -Wl,--section-start=.text=0x7e00 -Wl,--section-
start=.version=0x7ffe
atmega328_12: $(PROGRAM)_atmega328_12.hex
atmega328_12: $(PROGRAM)_atmega328_12.lst

atmega328_12_isp: atmega328
atmega328_12_isp: TARGET = atmega328
atmega328_12_isp: MCU_TARGET = atmega328p
# 512 byte boot, SPIEN
atmega328_12_isp: HFUSE = DE
# Low power xtal (12MHz) 16KCK/14CK+65ms
atmega328_12_isp: LFUSE = FF
```

```
# 2.7V brownout
atmega328_12_isp: EFUSE = 05
atmega328_12_isp: isp
```

Now tell Optiboot to use the avr-gcc compiler that came with the Arduino IDE:

```
$ export ENV=arduino
```

And compile the firmware:

```
$ make atmega328_12
```

The output should look something like this:

```
avr-gcc (GCC) 4.9.2
Copyright (C) 2014 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

BAUD RATE CHECK: Desired: 115200, Real: 115200, UBRR1 = 15, Error=0.0%
../../../../tools/avr/bin/avr-gcc -g -Wall -Os -fno-split-wide-types -mrelax -mmcu=atmega328p -
DF_CPU=12000000L -DBAUD_RATE=115200 -DLED_START_FLASHES=3 -c -o optiboot.o optiboot.c
../../../../tools/avr/bin/avr-gcc -g -Wall -Os -fno-split-wide-types -mrelax -mmcu=atmega328p -
DF_CPU=12000000L -DBAUD_RATE=115200 -DLED_START_FLASHES=3 -Wl,--section-
start=.text=0x7e00 -Wl,--section-start=.version=0x7ffe -Wl,--relax -nostartfiles -nostdlib -o
optiboot_atmega328_12.elf optiboot.o -lc
../../../../tools/avr/bin/avr-size optiboot_atmega328_12.elf
   text  data  bss   dec   hex filename
   464     0     0   464   1d0 optiboot_atmega328_12.elf
../../../../tools/avr/bin/avr-objcopy -j .text -j .data -j .version --set-section-flags
.version=alloc,load -O ihex optiboot_atmega328_12.elf optiboot_atmega328_12.hex
../../../../tools/avr/bin/avr-objdump -h -S optiboot_atmega328_12.elf >
optiboot_atmega328_12.lst
```

Adding the option in the Arduino IDE

The next step is to add a new option in the *Tools* menu in the Arduino IDE.

Open the 'boards.txt' file for editing, for example:

```
$ gedit ~/opt/arduino-1.8.1/hardware/arduino/avr/boards.txt
```

You have to add a menu entry to select the speed, change the f_cpu value, and point to the right bootloader. Edit the 'uno' entry, so it looks like this:

```
uno.name=Arduino/Genuino Uno

uno.vid.0=0x2341
uno.pid.0=0x0043
uno.vid.1=0x2341
uno.pid.1=0x0001
uno.vid.2=0x2A03
uno.pid.2=0x0043
uno.vid.3=0x2341
uno.pid.3=0x0243

uno.upload.tool=avrdude
uno.upload.protocol=arduino
uno.upload.maximum_size=32256
uno.upload.maximum_data_size=2048
uno.upload.speed=115200

uno.bootloader.tool=avrdude
uno.bootloader.low_fuses=0xFF
uno.bootloader.high_fuses=0xDE
uno.bootloader.extended_fuses=0x05
uno.bootloader.unlock_bits=0x3F
uno.bootloader.lock_bits=0x0F
#uno.bootloader.file=optiboot/optiboot_atmega328.hex
```

```

uno.build.mcu=atmega328p
#uno.build.f_cpu=16000000L
uno.build.board=AVR_UNO
uno.build.core=arduino
uno.build.variant=standard

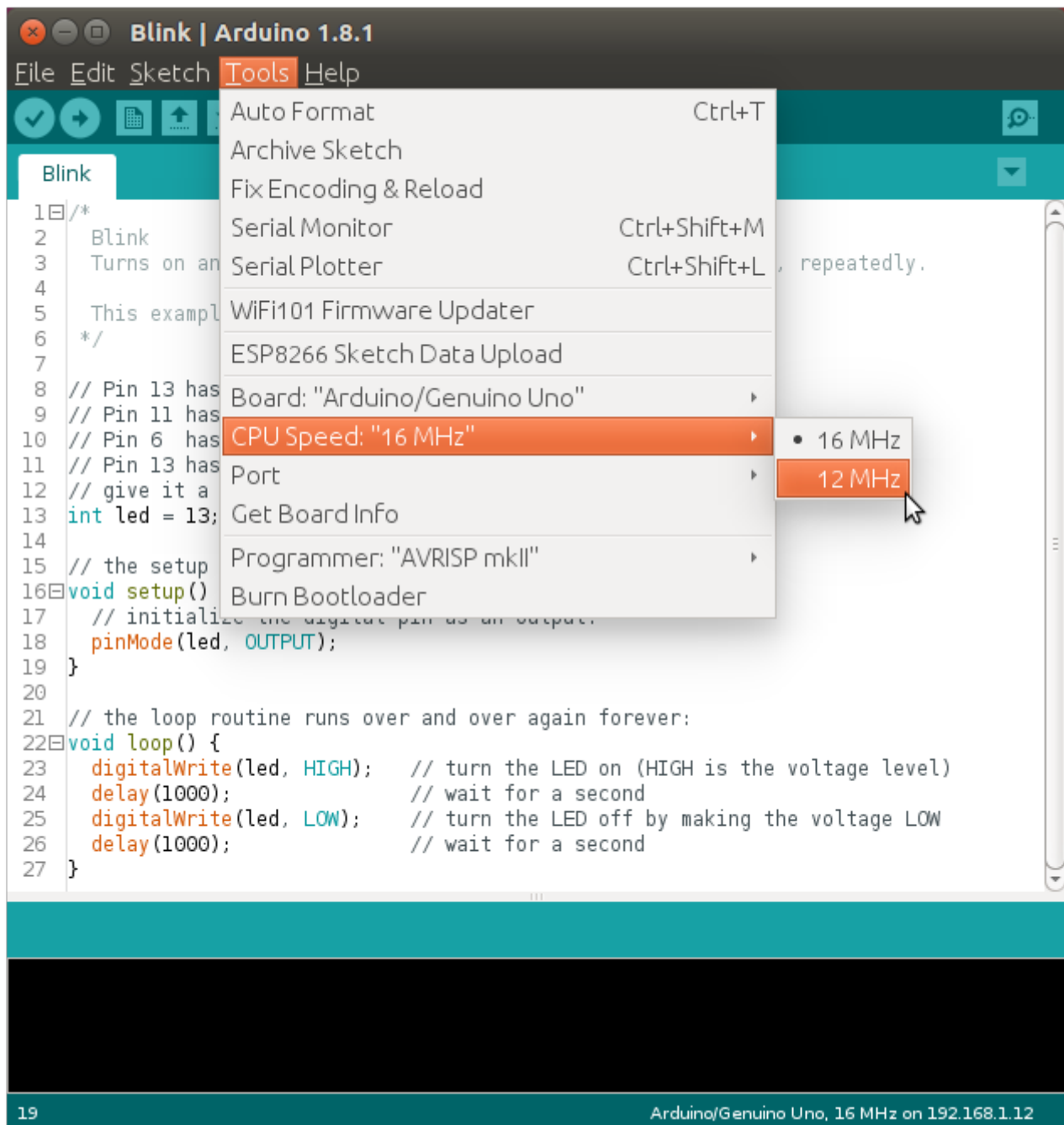
menu.speed=CPU Speed

uno.menu.speed.16=16 MHz
uno.menu.speed.12=12 MHz
uno.menu.speed.16.bootloader.file=optiboot/optiboot_atmega328.hex
uno.menu.speed.12.bootloader.file=optiboot/optiboot_atmega328_12.hex
uno.menu.speed.16.build.f_cpu=16000000L
uno.menu.speed.12.build.f_cpu=12000000L

```

Note that the original `f_cpu` and `bootloader.file` lines are commented out. Also make sure that the speeds and bootloader filenames are the same as the ones in the Optiboot Makefile.

Restart the Arduino IDE for the changes to take effect. You should see the new options when opening the 'Tools' menu.

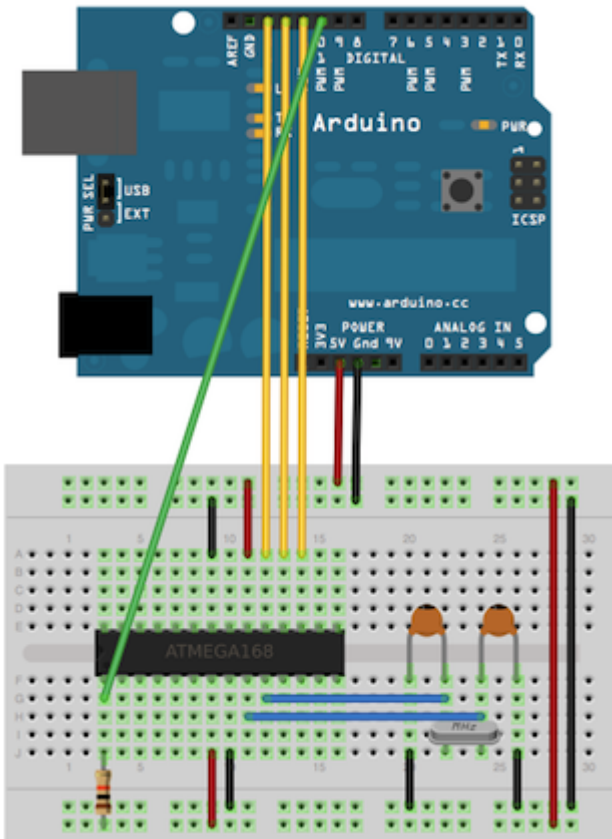


Flashing the bootloader to the ATmega328P

Uploading the ArduinoISP firmware

Connecting to the Arduino

Check the result with the following image:



Burning the bootloader

Testing

The LED on D13 should start blinking.