Back to tutorial index

FabISP: Programming

1 To program the FabISP, you first need to install the necessary software for your operating system and download the firmware.
2 Then you edit the Makefile
3 Set the fuses / program the board
4 Verify that the board is working properly
5 Then you need to open up the jumpers to make it a programmer.

Install Necessary Software for AVR Programming:

For the electronics units in the Fab Academy, you will need:
1 Avrdude (for programming AVR microcontrollers)
2 GCC (to compile C code)

Jump to the instructions for your Operating System:
• Ubuntu
• Mac OS
• Windows

Ubuntu Software Install

Get and install avrdude / GCC software and dependencies:

Open Terminal and type:

```
sudo apt-get install flex byacc bison gcc libusb-dev avrdude
```

Then type:

```
sudo apt-get install gcc-avr
```

- type "y" when asked to do so by your system

Then type:

```
sudo apt-get install avr-libc
```

Then type (may already be installed):

```
sudo apt-get install libc6-dev
```

Download and Unzip the Firmware:

Move to the desktop

```
 cd ~/Desktop
```

Download the firmware from the Fab Academy Electronics Production page.

```
wget
http://academy.cba.mit.edu/classes/embedded_programming/firmware.zip
```

Unzip the firmware

```
unzip firmware.zip
```

Mac OS Software Install

Get and install avrdude / GCC software and dependencies:

1. **Download and Install Crosspack AVR** - Has an installer.
2.
3. **Get Make (via XCode):**
   - If you are running Lion or higher - you can download XCode from the Apple App store.
   - If you are running a pre-Lion OSX - Get the install disks that came with your mac and install the developer tools.

Download the firmware (right click on the link below and save it to your desktop):

FabISP Firmware for MacOS 10.8.2 (Mountain Lion, possibly for Lion too?)

FabISP Firmware for earlier versions of MacOS

Open terminal navigate to the desktop:

```
cd ~/Desktop/
```

Unzip the firmware.zip directory (the directory will be "firmware.zip" if you downloaded the earlier version):

```
unzip fabISP_mac.0.8.2_firmware.zip
```

Move into the newly created firmware directory on your desktop

```
cd ~/Desktop/firmware
```

Windows Software / Drivers Install

Get and install avrdude / GCC software and dependencies and drivers:

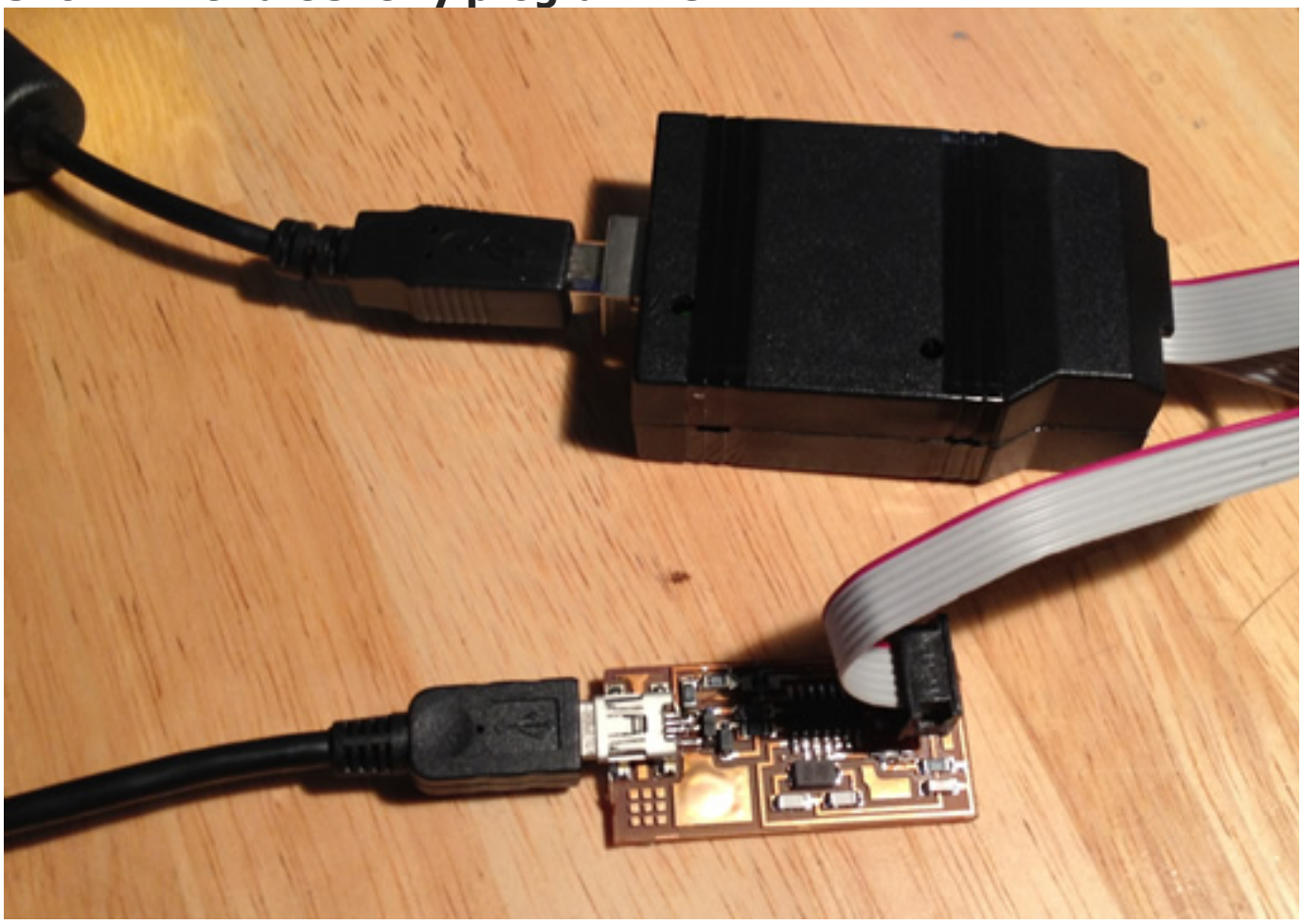
1. **Warning, WinAVR is abandoned. Installing it can destroy your systems path variable!** You can use the installer, but before you start, take note of your current system path.
2. **Download and Install WinAVR** - Has a (broken) installer.
3. - Here is a step-by-step set of instructions
4. After installing check your systems path variable, if it only contains the path to the winavr installation:
    - copy those values
    - restore your old path
    - add the windavr path back to it
    - close any commandprompt window you may have open
5. Download the drivers for your version of Windows
6. Download the FabISP Firmware
7. Plug in another FabISP or USBtiny programmer.
8. Install the drivers:
9. Go to the Start menu -> Control Panel -> Device Manager (or run "mmc devmgmt.msc")
    - Locate "FabISP" under "Other Devices"
    - Right click on the "FabISP"
    - Select "Update Driver Software.."
    - Select "Browse my computer"
    - Navigate to the drivers folder you downloaded at step 4 and click on the folder.
    - Hit "ok" to install the drivers for the USBtiny / FabISP

Power the FabISP Board

The board needs power:

- Make sure that the mini USB connector for the FabISP you are trying to program is plugged into your computer.
- **AND** that a separate pogramer is plugged in to the 6-pin programming header. (this could be another working FabISP or the ATAVRISP2, a USBtiny, an AVR Dragon, etc.)

Shown with a USBtiny programmer



If you are using the ATAVRISP2 programmer, you can skip step 7, you do not need to edit the Makefile, it is already set up to work with the ATAVRISP2 If you are using another

programmer you will need to edit the Makefile.

Helpful ATAVRISP2 Programmer Light Indicator Messages

If you are using the ATAVRISP2 programmer only. If you connect the programmer to the 6-pin programming header on your FabISP board and you get:

- **Green Light:** means that the header is soldered correctly, the board is getting power.
- **Yellow Light:** means that the board is getting power, but most likely the 6-pin programming header is not soldered correctly (re-flow your solder joints / check for cold joints, check for shorts).
- **Red Light:** means that the board is not getting power - check for shorts.

Edit the Makefile

The Makefile is in the firmware directory that you downloaded. The Makefile is set up to work with the AVRISP2 by default. If you are using another programmer, you will need to edit the Makefile.

Ubuntu:

```
nano Makefile
```

Mac:

Open the Makefile with TextEdit.

Windows:

Open the Makefile with Notepad++.

Make Changes - All OS:

A window will open containing the Makefile. Go to the line that says:

#AVRDUDE = avrdude -c usbtiny -p $(DEVICE) # edit this

line for your programmer
AVRDUDE = avrdude -c avrisp2 -P usb -p $(DEVICE) # edit this line for your programmer

- If using the USBtiny programmer or another FabISP
- Remove the "#" in front of the line with "usbtiny" in it
- Add a "#" to beginning the line with the "avrisp2" in it to comment it out.
- save the Makefile

Program the FabISP (All OS):

Navigate to the directory where you saved the FabISP firmware. If you followed the instructions above, this will be the desktop.

Open your terminal / command line interface and move to the firmware directory.

Ubuntu / Windows type:

```
cd Desktop/firmware
```

For Mac users who downloaded the modified firmware:

```
cd Desktop/fabISP_mac.0.8.2_firmware
```

Next you need to compile the firmware.

Type:

```
make clean
```

If you are successful - you will see this response from the system:

```
akaziuna@Titan:~/Desktop/firmware$ make clean
rm -f main.hex main.lst main.obj main.cof main.list
main.map main.eep.hex
main.elf *.o usbdrv/*.o main.s usbdrv/oddebug.s
usbdrv/usbdrv.s
```

Type:

```
make hex
```

If you are successful - you will see this response from the system:

```
akaziuna@Titan:~/Desktop/firmware$ make hex
avr-gcc -Wall -Os -DF_CPU=20000000 -Iusbdrv -I. -
DDEBUG_LEVEL=0
-mmcu=attiny44 -c usbdrv/usbdrv.c -o
usbdrv/usbdrv.o
avr-gcc -Wall -Os -DF_CPU=20000000 -Iusbdrv -I. -
DDEBUG_LEVEL=0
-mmcu=attiny44 -x assembler-with-cpp -c
usbdrv/usbdrvasm.S -o usbdrv/usbdrvasm.o
avr-gcc -Wall -Os -DF_CPU=20000000 -Iusbdrv -I. -
DDEBUG_LEVEL=0
-mmcu=attiny44 -c usbdrv/oddebug.c -o
usbdrv/oddebug.o
avr-gcc -Wall -Os -DF_CPU=20000000 -Iusbdrv -I. -
DDEBUG_LEVEL=0
-mmcu=attiny44 -c main.c -o main.o
avr-gcc -Wall -Os -DF_CPU=20000000 -Iusbdrv -I. -
DDEBUG_LEVEL=0
-mmcu=attiny44 -o main.elf usbdrv/usbdrv.o
usbdrv/usbdrvasm.o usbdrv/oddebug.o
main.o
rm -f main.hex main.eep.hex
avr-objcopy -j .text -j .data -O ihex main.elf
main.hex
avr-size main.hex
   text     data      bss      dec      hex  filename
      0     2020        0     2020      7e4  main.hex
```

Next, you need to set the fuses so your board will use the external clock (crystal)

Type:

```
make fuse
```

If you are successful - you will see the following response from the system:

```
akaziuna@Titan:~/Desktop/firmware$ sudo make fuse
avrdude -c usbtiny -p attiny44  -U hfuse:w:0xDF:m -
U lfuse:w:0xFF:m

avrdude: AVR device initialized and ready to accept
```

```
instructions

Reading |
################################################## 
| 100% 0.01s

avrdude: Device signature = 0x1e9207
avrdude: reading input file "0xDF"
avrdude: writing hfuse (1 bytes):

Writing |
################################################## 
| 100% 0.00s

avrdude: 1 bytes of hfuse written
avrdude: verifying hfuse memory against 0xDF:
avrdude: load data hfuse data from input file 0xDF:
avrdude: input file 0xDF contains 1 bytes
avrdude: reading on-chip hfuse data:

Reading |
################################################## 
| 100% 0.00s

avrdude: verifying ...
avrdude: 1 bytes of hfuse verified
avrdude: reading input file "0xFF"
avrdude: writing lfuse (1 bytes):

Writing |
################################################## 
| 100% 0.01s

avrdude: 1 bytes of lfuse written
avrdude: verifying lfuse memory against 0xFF:
avrdude: load data lfuse data from input file 0xFF:
avrdude: input file 0xFF contains 1 bytes
avrdude: reading on-chip lfuse data:

Reading |
```

```
##################################################
| 100% 0.00s

avrdude: verifying ...
avrdude: 1 bytes of lfuse verified

avrdude: safemode: Fuses OK

avrdude done.  Thank you.
```

Next you want to program the board to be an ISP.

Then type:

```
make program
```

If you are successful - you will see the following response from the system.

```
akaziuna@Titan:~/Desktop/firmware$ sudo make
program
[sudo] password for akaziuna:
avrdude -c usbtiny -p attiny44  -U
flash:w:main.hex:i

avrdude: AVR device initialized and ready to accept
instructions

Reading |
##################################################
| 100% 0.01s

avrdude: Device signature = 0x1e9207
avrdude: NOTE: FLASH memory has been specified, an
erase cycle will be performed
         To disable this feature, specify the -D
option.
avrdude: erasing chip
avrdude: reading input file "main.hex"
avrdude: writing flash (2020 bytes):

Writing |
##################################################
| 100% 5.68s
```

```
avrdude: 2020 bytes of flash written
avrdude: verifying flash memory against main.hex:
avrdude: load data flash data from input file
main.hex:
avrdude: input file main.hex contains 2020 bytes
avrdude: reading on-chip flash data:

Reading |
################################################
| 100% 3.36s



avrdude: verifying ...
avrdude: 2020 bytes of flash verified

avrdude: safemode: Fuses OK

avrdude done.  Thank you.

avrdude -c usbtiny -p attiny44  -U hfuse:w:0xDF:m -
U lfuse:w:0xFF:m

avrdude: AVR device initialized and ready to accept
instructions

Reading |
################################################
| 100% 0.01s

avrdude: Device signature = 0x1e9207
avrdude: reading input file "0xDF"
avrdude: writing hfuse (1 bytes):

Writing |
################################################
| 100% 0.00s
```

```
avrdude: 1 bytes of hfuse written
avrdude: verifying hfuse memory against 0xDF:
avrdude: load data hfuse data from input file 0xDF:
avrdude: input file 0xDF contains 1 bytes
avrdude: reading on-chip hfuse data:

Reading |
################################################## 
| 100% 0.00s

avrdude: verifying ...
avrdude: 1 bytes of hfuse verified
avrdude: reading input file "0xFF"
avrdude: writing lfuse (1 bytes):

Writing |
################################################## 
| 100% 0.00s

avrdude: 1 bytes of lfuse written
avrdude: verifying lfuse memory against 0xFF:
avrdude: load data lfuse data from input file 0xFF:
avrdude: input file 0xFF contains 1 bytes
avrdude: reading on-chip lfuse data:

Reading |
################################################## 
| 100% 0.00s

avrdude: verifying ...
avrdude: 1 bytes of lfuse verified

avrdude: safemode: Fuses OK

avrdude done.  Thank you.
```

Wooo! Success!

Errors

- If you get errors - read the errors and follow instructions.
- If avrdude cannot connect to your board's microcontroller -

you should follow the "Troubleshooting Short Circuits" instructions and ask your instructor for help.

To Verify That Your ISP is working correctly:

Ubuntu 11.10:

Type:

```
lsusb
```

If your FabISP has been successfully programmed, you should see a list of the USB devices plugged into your computer. The FabISP will be listed in a line like the following:

```
Bus 002 Device 004: ID 1781:0c9f Multiple Vendors
USBtiny
```

Note: Ubuntu 10.10 lists the device as something like:

```
Bus 002 Device 004: ID 1781:0c9f Multiple Vendors
```

Mac:

Go to the System Profiler > Hardware > USB > Hub: **Step - By - Step:**

1 Click the "apple" menu in your main toolbar
2 Select "about this mac"
3 Select "more info"
4 Under the "Contents" menu in the left hand navigation
5 - Click "Hardware" to expand the hardware menu (if not already expanded)
6 - Click "USB"
7 - Under the "USB Device Tree"
8 - Click "Hub" to expand the hub menu (if not already expanded)
9 - "FabISP" should be listed in the hub menu
10 Your FabISP device has been successfully programmed and is recognized by your computer.

Windows:

Go to the Start Menu > Hardware and Sound. The FabISP

should be listed.

After You Have Programmed the Board:

Remove the 0 ohm resistor and solder bridge as shown in the picture below. Now you can use it as a programmer to program other boards.