

## Additional Reading Material

This material contains

1. Precautions that should be taken before programming.
2. Ways to troubleshoot when the Mainboard is not programmed properly.

### 1. Precautions:

- Make sure that you disconnect all the external connections from the Mainboard before programming it to avoid electrical shorts.
- Make sure that the USBasp is connected properly from both sides. That is, LED marked 'G' on USBasp should be glowing.
- Make sure that the microcontroller is powered up properly. This you can check from the appropriate Vcc and GND pins of the microcontroller.
- Make sure that in the terminal you're in the same directory as the one in which you have the saved HEX file to be programmed by the controller.

### 2. Troubleshoot:

#### **2.1) Software checks:**

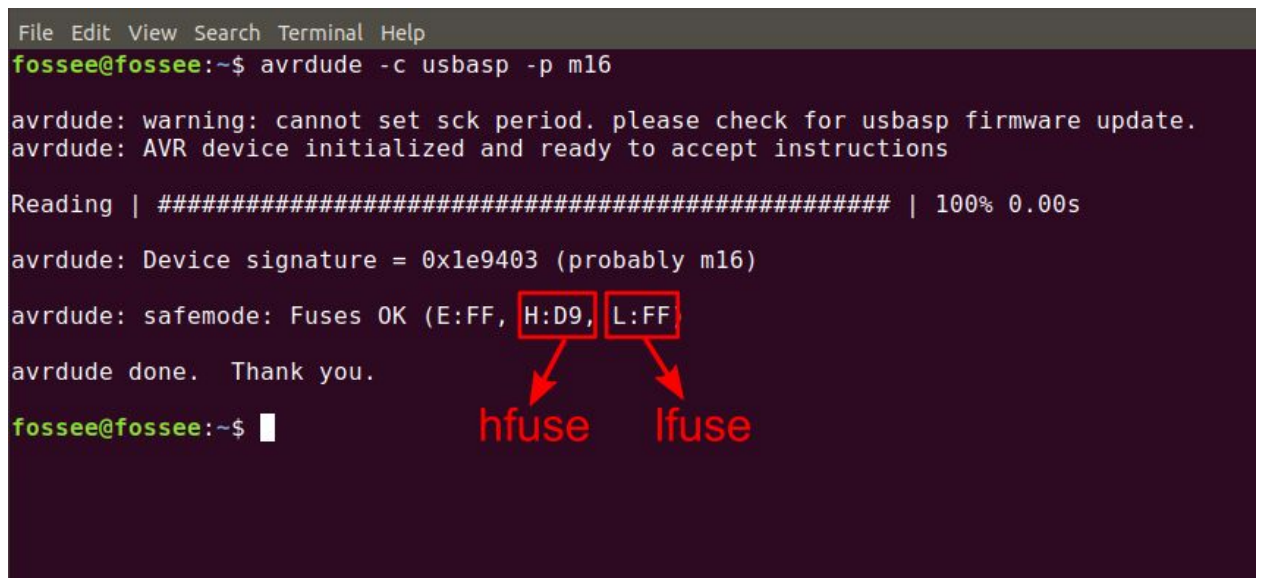
If your controller is not programmed properly, first check whether

- AVRDUDE is working properly,

For that type the command **avrdude -c usbasp -p m16** in the terminal and press ENTER.

You should get the response as shown in the image.

Fig 1.1



```
File Edit View Search Terminal Help
fossee@fossee:~$ avrdude -c usbasp -p m16

avrdude: warning: cannot set sck period. please check for usbasp firmware update.
avrdude: AVR device initialized and ready to accept instructions

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

avrdude: Device signature = 0x1e9403 (probably m16)
avrdude: safemode: Fuses OK (E:FF, H:D9, L:FF)
avrdude done. Thank you.

fossee@fossee:~$
```

The terminal output shows the successful execution of the `avrdude -c usbasp -p m16` command. The device signature is identified as `0x1e9403 (probably m16)`. The fuse status is reported as `Fuses OK (E:FF, H:D9, L:FF)`. Red boxes highlight the `H:D9` and `L:FF` values, with red arrows pointing to the labels `hfuse` and `lfuse` respectively.

If AVRDUDE is not installed properly, “**avrdude: command not found**” will be displayed.

## 2.2) Check the Hardware Connections:

- Check that USBasp is connected properly.
- Check that the FRC cable of the USBasp is not stranded in the midway.
- Check that the microcontroller firmly sits on the base and is powered up properly.
- Make sure there are no external modules attached while programming.

## 2.3) Check the parameters of the avrdude command:

- Check whether you have given all the essential flags in the below command  
**avrdude -c usbasp -p m16 -U flash:w:sample.hex**

## 2.4) How to read fuse bits of a microcontroller:

- First, connect OpenPLC v1 Mainboard to your computer using USBasp programmer.
- In the terminal, type **avrdude -c usbasp -p m16 -U lfuse:r::-h -U hfuse:r::-h**
  - 'r' indicates that a read operation is being performed.
  - 'h' indicates that value will be in raw hex format.
  - The factory default settings for Atmega16 Fuse bits are: **hfuse** set to **0x99** hex and **lfuse** set to **0xE1** hex value.
  - Since there is a 16 MHz crystal on board, we have set the fuse bits values accordingly before creating this tutorial.
  - So the expected values for the fuse bits are: **hfuse** set to **0xD9** hex and **lfuse** set to **0xFF** hex value as shown in the image.( Fig 1.1)

If you get different values you can change them as explained in the section 2.5.  
Otherwise, you can skip section 2.5.

## 2.5) How to set up fuse bits for a microcontroller:

**Disclaimer:** Please follow the instructions to change the fuse bits as described below, only

1. If the fuse bits read do not match as explained in the section 2.4.
2. If the timers are behaving way out of sync.
3. If the serial communication is misbehaving with a huge error rate.
4. If you understand the risk involved in changing fuse bits.
5. If you have replaced the microcontroller with a new one.

Change the fuse bits values only if they're **not** matching with the following hex values

**lfuse : 0xFF and hfuse : 0xD9**

**A fused crystal oscillator or incorrect settings of the fuse bits may cause irreversible damage to the microcontroller.**

- Reverify twice before changing the fuse bits because sometimes the fuse bits values may be read wrong due to communication failure.
- Verify that your programmer is proper and functional.

So to change the fuse bits, type the following command in the terminal.

```
sudo avrdude -p m16 -c usbasp -U lfuse:w:0xFF:m -U hfuse:w:0xD9:m -B10
```

```
File Edit View Search Terminal Help
fossee@fossee:~$ sudo avrdude -p m16 -c usbasp -U lfuse:w:0xFF:m -U hfuse:w:0xD9:m -B10
```

This will set the **hfuse** to **0xFF** and **lfuse** to **0xD9**. You will get the response as shown below.

```
File Edit View Search Terminal Help
fossee@fossee:~$ sudo avrdude -p m16 -c usbasp -U lfuse:w:0xFF:m -U hfuse:w:0xD9:m -B10
[sudo] password for fossee:

avrdude: set SCK frequency to 93750 Hz
avrdude: warning: cannot set sck period. please check for usbasp firmware update.
avrdude: AVR device initialized and ready to accept instructions

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

avrdude: Device signature = 0x1e9403 (probably m16)
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: reading input file "0xD9"
avrdude: writing hfuse (1 bytes):

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

avrdude: 1 bytes of hfuse written
avrdude: verifying hfuse memory against 0xD9:
avrdude: load data hfuse data from input file 0xD9:
avrdude: input file 0xD9 contains 1 bytes
avrdude: reading on-chip hfuse data:

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

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

avrdude: safemode: Fuses OK (E:FF, H:D9, L:FF)

avrdude done. Thank you.

fossee@fossee:~$
```