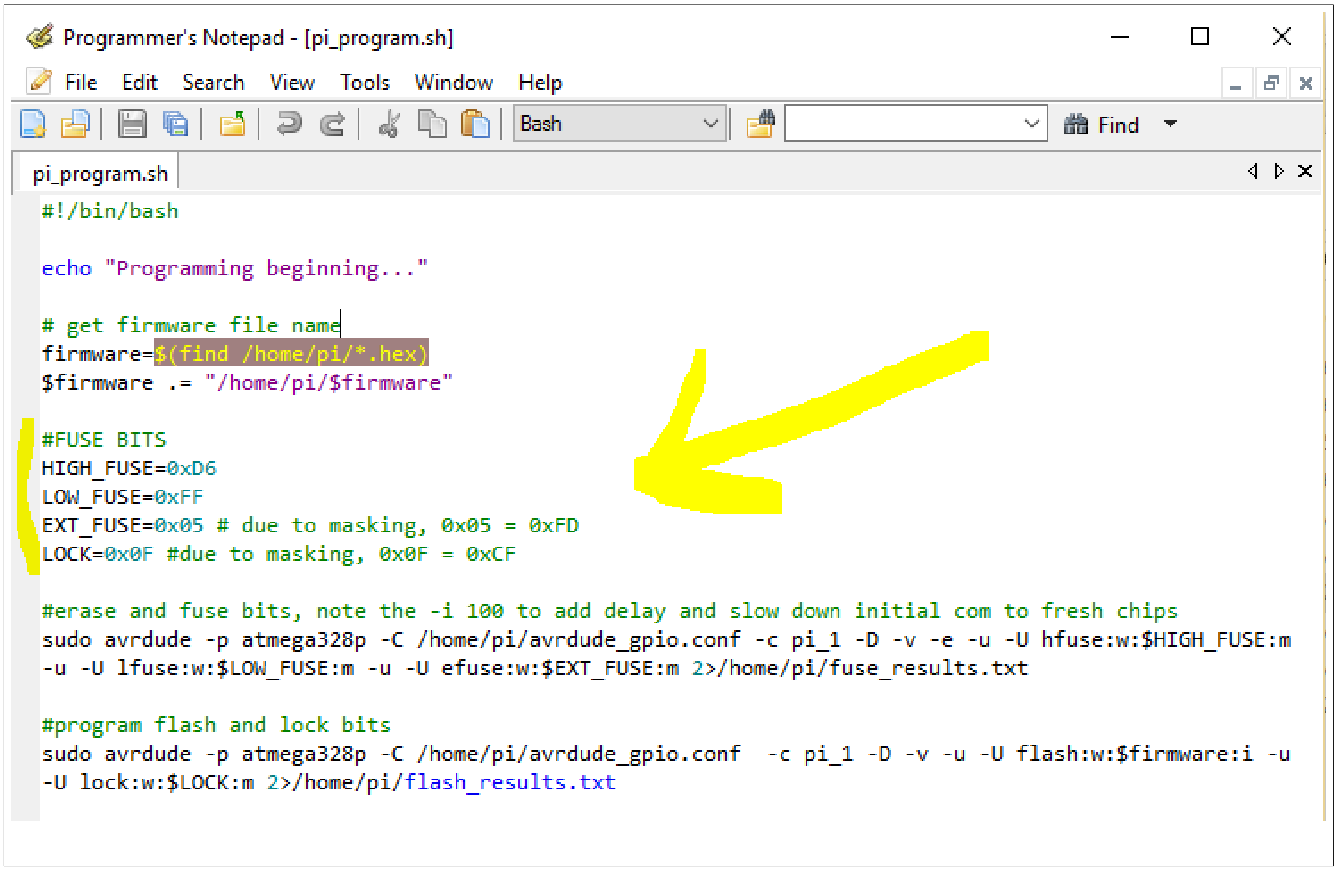
Pi_Grammer Setup Instructions
PL 8/26/2016

This guide will show how to setup a Pi_Grammer for production.

1. Grab a Raspi 2, a micro SD card (with generic Pi_grammer image pre-programmed) Pi_grammer Shield, 1x6 cable and adapter.

2. Create a bash file with proper fuse/ext/lock bits.
   - Note, this can be done on your "usual" computer to avoid having to setup the Pi with a screen/mouse/keyboard each setup.
   - Please start with the example "pi_program.sh" file that lives in the repo.
     - Most of the time, you should only need to adjust the fuse/ext/lock bits. These are variables at the top of the bash file, for ease of editing.


Programmer's Notepad - [pi_program.sh]
File Edit Search View Tools Window Help
Bash
Find
pi_program.sh

```
#!/bin/bash

echo "Programming beginning..."

# get firmware file name
firmware=$(find /home/pi/*.hex)
$firmware .= "/home/pi/$firmware"

#FUSE BITS
HIGH_FUSE=0xD6
LOW_FUSE=0xFF
EXT_FUSE=0x05 # due to masking, 0x05 = 0xFD
LOCK=0x0F #due to masking, 0x0F = 0xCF

#erase and fuse bits, note the -i 100 to add delay and slow down initial com to fresh chips
sudo avrdude -p atmega328p -C /home/pi/avrdude_gpio.conf -c pi_1 -D -v -e -u -U hfuse:w:$HIGH_FUSE:m
-u -U lfuse:w:$LOW_FUSE:m -u -U efuse:w:$EXT_FUSE:m 2>/home/pi/fuse_results.txt

#program flash and lock bits
sudo avrdude -p atmega328p -C /home/pi/avrdude_gpio.conf  -c pi_1 -D -v -u -U flash:w:$firmware:i -u
-U lock:w:$LOCK:m 2>/home/pi/flash_results.txt
```

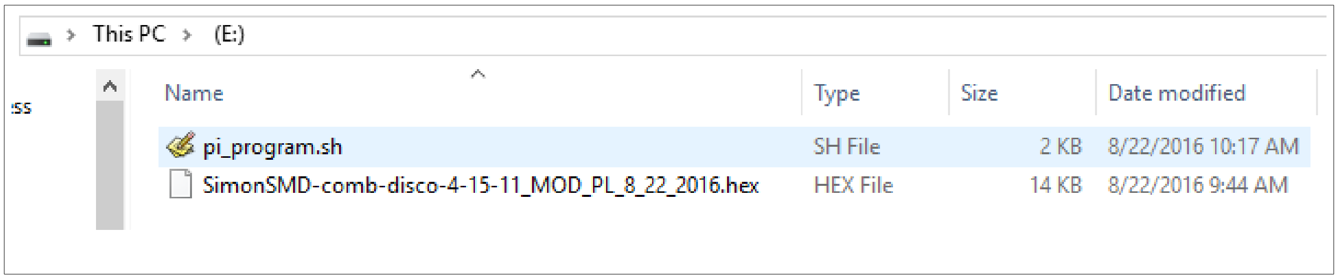
   - Note, AVRDUDE will not let you write to "unused" bits. This effects how we write to the EXT and LOCK bits:
     - If you want to write the value "0xFD" to the EXT bit (which is a common value we write on a lot of 328 based arduino products), then use the value "0x05" in AVRDUDE.

        - **The 3 bits on the right are the only ones we care about.**
        - “0xFD” = 1111 1**101**
        - “0x05” = 0000 0**101**
    - If you want to write the usual “0xCF” to LOCK, then write 0x0F” in AVRDUDE.
        - **The 4 bits on the right are the only ones we care about.**
        - “0xCF” = 1100 **1111**
        - “0x0F” = 0000 **1111**

3. Go get the hex file from the product folder.

4. Copy the hex file and bash file onto your thumb drive.

   Note, the hex file can be named anything (and so should be left the same as it was – for traceability). The bash file must be named exactly “pi_program.sh”. These should be the only two files on the thumb drive.



This PC > (E:)

Name	Type	Size	Date modified
pi_program.sh	SH File	2 KB	8/22/2016 10:17 AM
SimonSMD-comb-disco-4-15-11_MOD_PL_8_22_2016.hex	HEX File	14 KB	8/22/2016 9:44 AM

5. Plug in the uSD card (with generic image) into your Pi.

6. Power up your Pi with a microB USB cable (either a computer hub or wall wort).

7. Wait for blinking stat LED. This indicates that bootup is complete and the python script is running.

8. Take your prepared thumb drive (with hex and bash files) from your windows machine, and plug in your thumb drive into USB on the Pi.

9. Verify the circular LED sequence happens twice.
    - One for hex file.
    - Second for bash file.

- This verifies that the python script recognized the “media” device, found a “.hex” and a “pi_program.sh” file on it., and copied it to the local folder successfully.

10. Remove thumb drive.
11. Shutdown the Pi.
    - Press and hold the “SHUTDOWN” button.
    - All LEDs begin blinking.
    - When blinking stops, that means it has fully shut down.
    - Unplug power.