Bootloader Flash Guide For ATMEGA32A MCU For Both Sesame & Basketweave Or Any Other Keebs That Use This MCU

By
Protieus

Important Note:

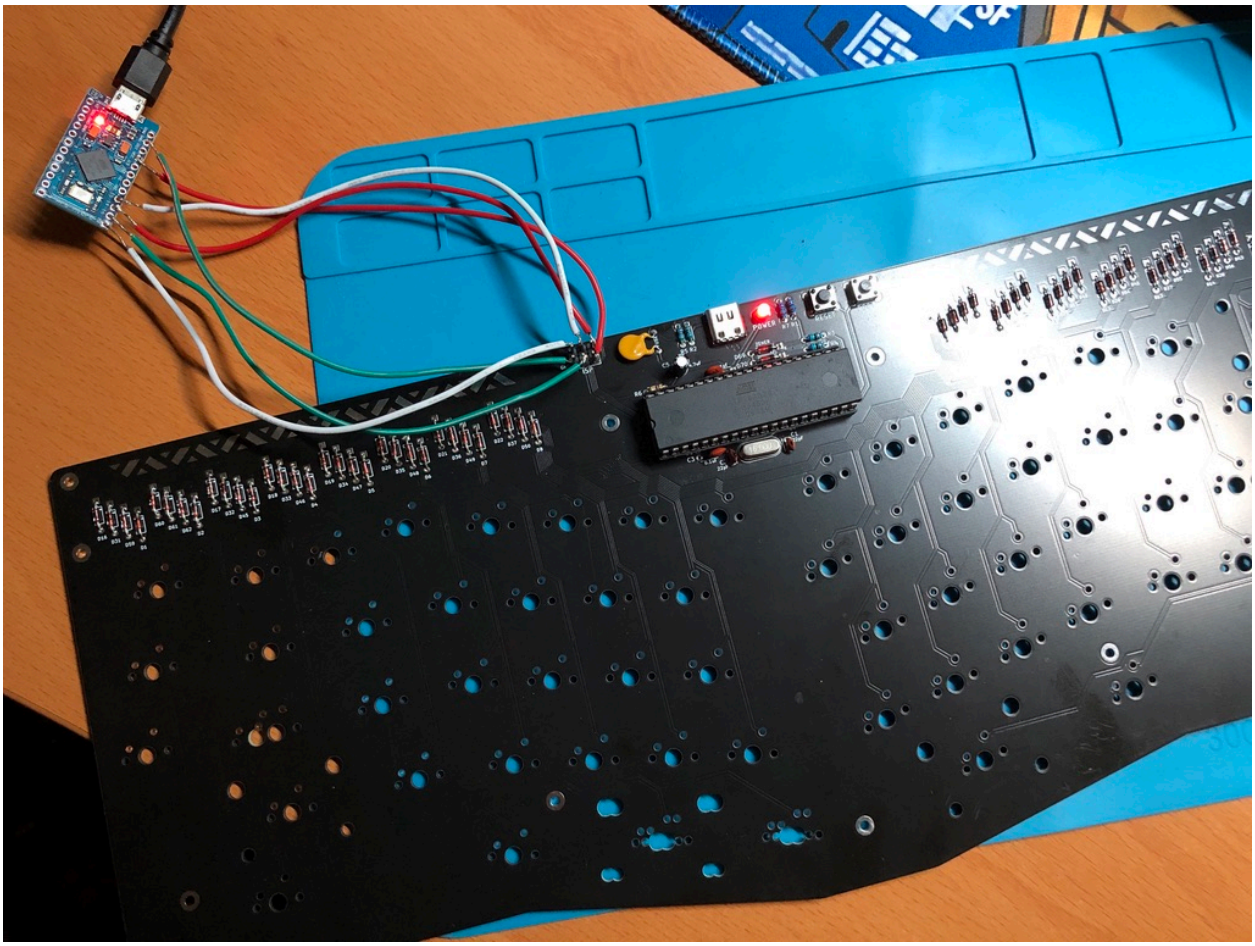
All ATMEGA32a chips that come from the factory do not have bootloader installed and therefore you need to flash a bootloader before it can be recognized by you PC so that you can flash a firmware to it. Most of my guide is taken from Red Herring keyboard github from the firmware section. https://github.com/dcpedit/redherring/tree/main/firmware Shout out to dcpedit. Without his wonderful detailed firmware setup guide, I won't be able to get my keyboard working. I also refer to this guide for the wire pins setup for the AVR programmer https://hackaday.io/project/159973-z80-mbc2-a-4-ics-homebrew-z80-computer/log/150087-how-use-the-icsp-port-with-the-usbasp-programmer-under-linux-to-burn-the-bootloader
I also refer to this guide for the Pro Micron to ISP pcb wire pin out
https://github.com/qmk/qmk_firmware/blob/master/docs/isp_flashing_guide.md

The guide assumes that you already built an QMK Msys environment for your keyboard hex building and can be used to flash your bootloader into the mcu.

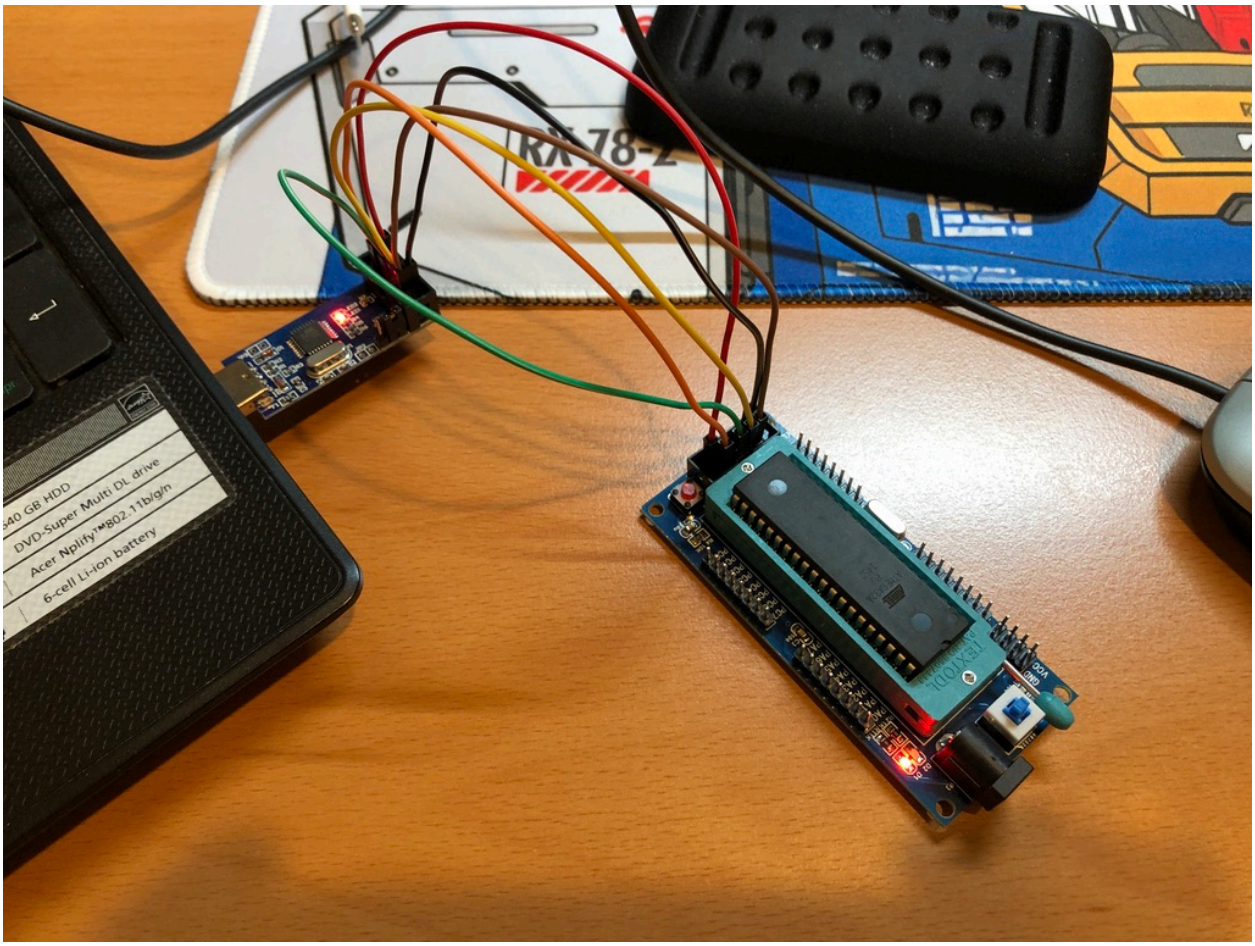
ALWAYS REMEMBER: Don't rage quit and give up. Take deep breaths and calm down. Do things with patience. Read the guide 2x before you actually perform the actions. It took me 2 months of research to get a clear understanding of the method and issues.

There are 2 methods that you can flash a bootloader into the mcu:

1. Using a Pro Micron to act as a programmer wired to the keyboard pcb ISP pins. (Easiest since PM is widely accessible and you would need some light soldering).

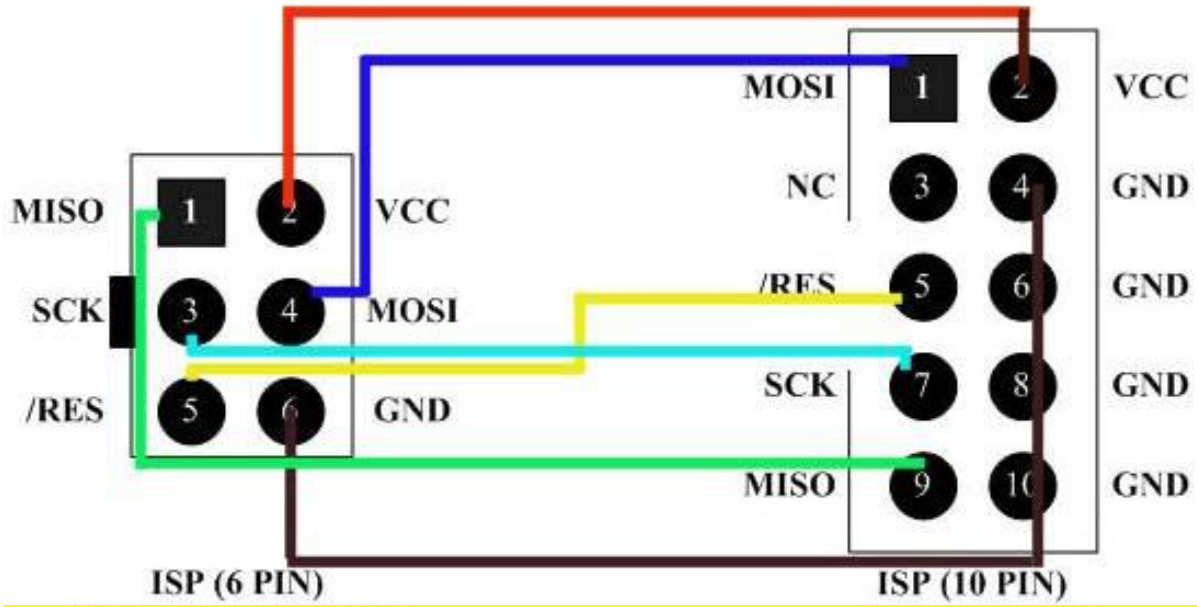


2. Using a clone or any AVR programmer with a mcu chip module to flash without the use of pcb ISP pins (Would need to buy the AVR programmer from either Aliexpress or Amazon, etc).



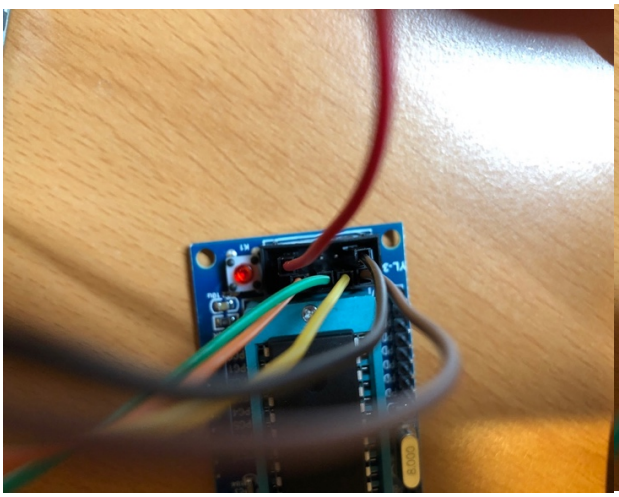
METHOD 1: AVR Programmer with MCU Module.

1. Connect the jumper wires to the AVR programmer.

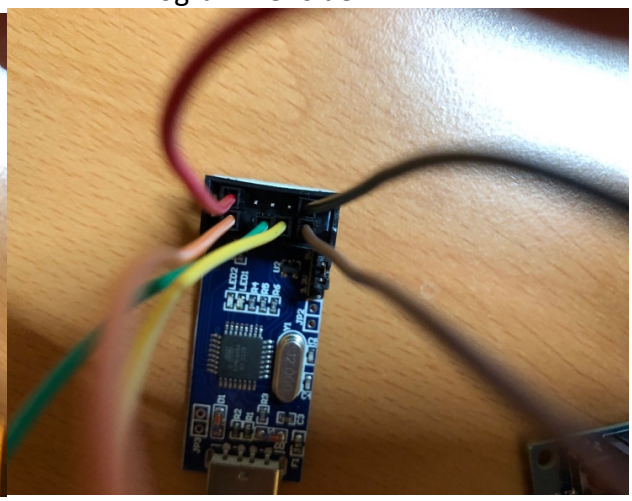


For the ground wire, connect only the pin 10 on the AVR programmer side to the ISP ground side. Ignore the rest of the GND pins (4, 6, 8) from the AVR programmer side. Then connect the rest of the corresponding pins VCC, MISO, SCK, MOSI, RES as per diagram above. 6 pins are on the ISP pcb side. 10 pins are on the AVR programmer side. Notice the "notch" on the AVR 10 pin side? Use that as reference on the orientation of where the pins are. FYI: VCC provides power.

MCU module side

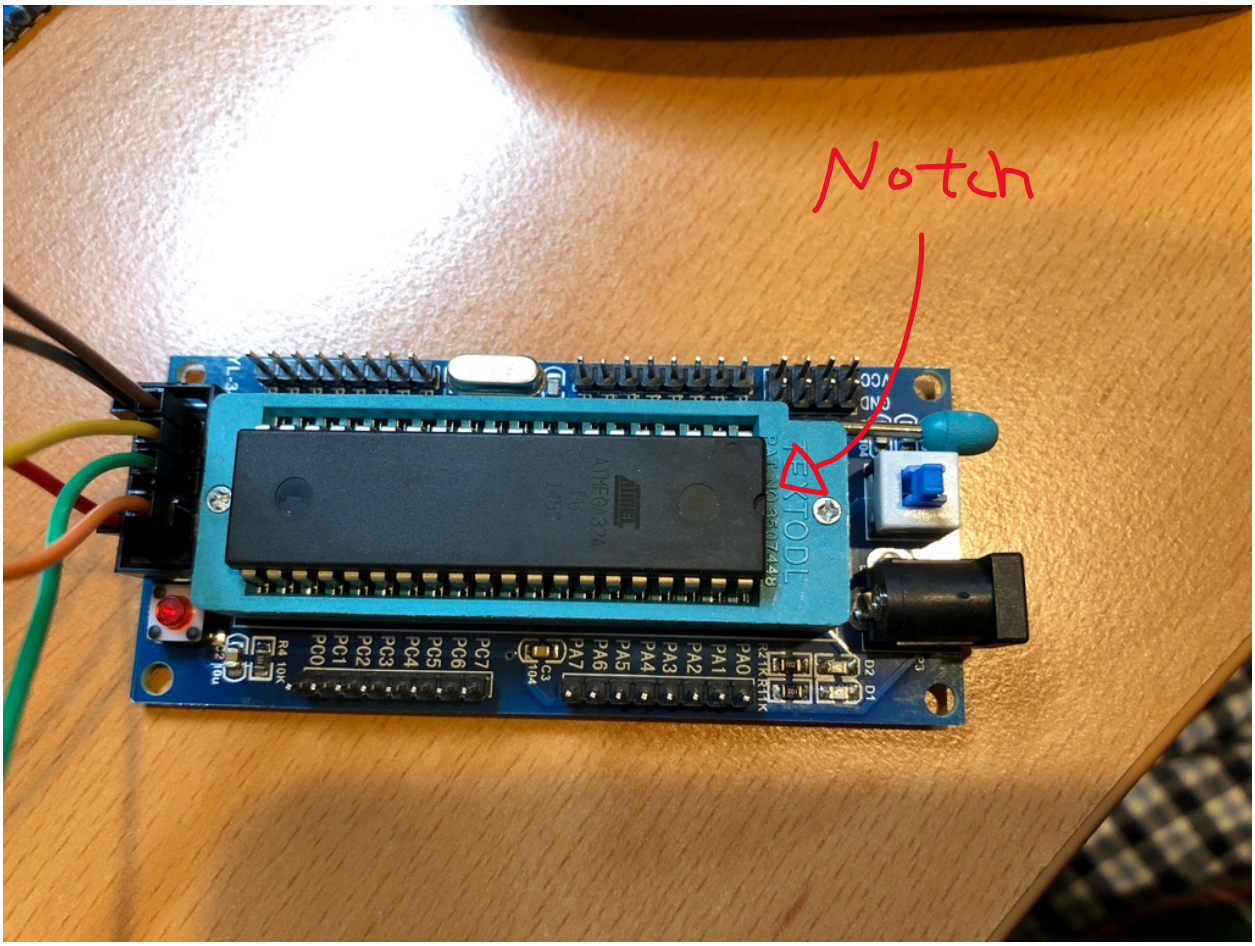


AVR Programmer side



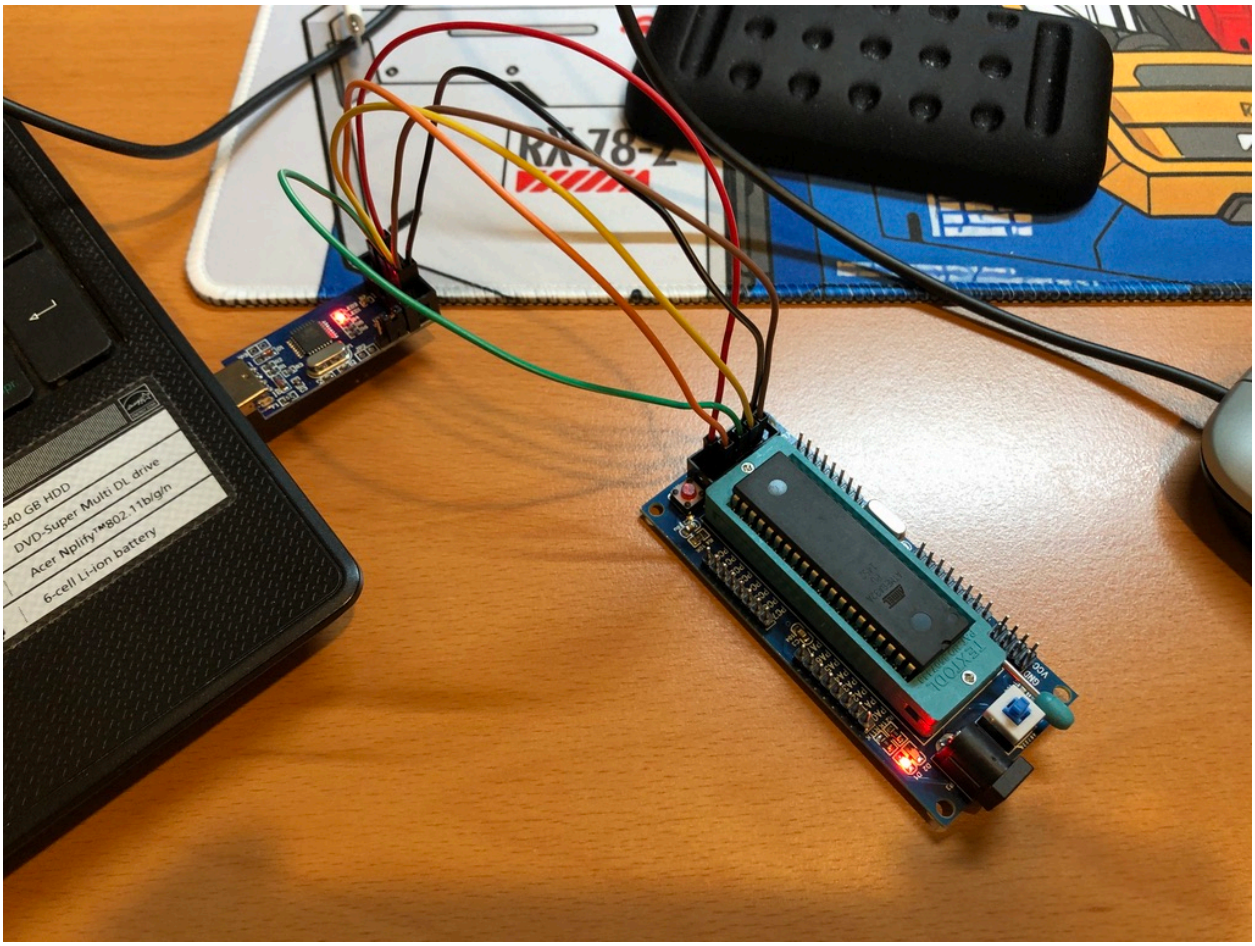
Sorry my pic is a bit hard to see. Use this as a reference to the diagram above. You connect the same exact way as the AVR Programmer side on the MCU module side. Using jumper wires, you don't need to deal with messy soldering.

2. Insert the ATMEGA32a mcu, the notch on the chip facing towards the blue button as per picture below:



3. Once all your jumper wires are connected to the correct pins, plug your AVR programmer into your PC usb A port. You will red LED light up.

4. Next press the blue button to turn on the MCU module side and a red LED will light up like below picture.

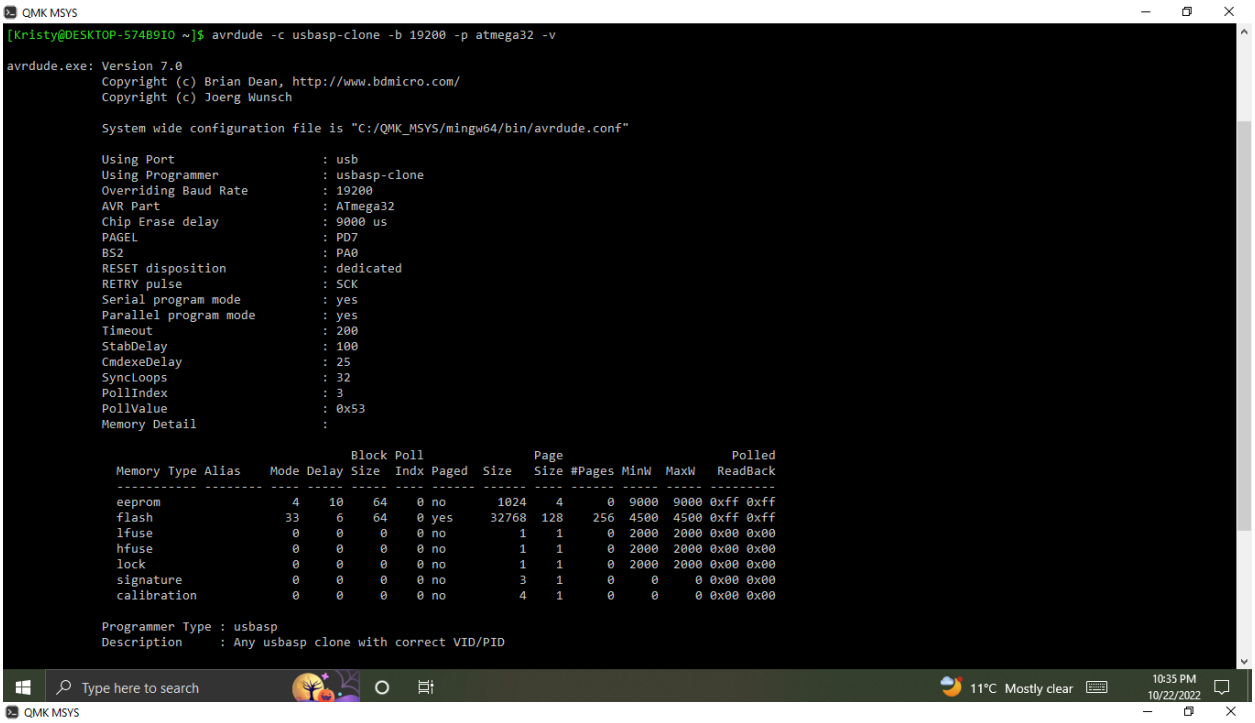


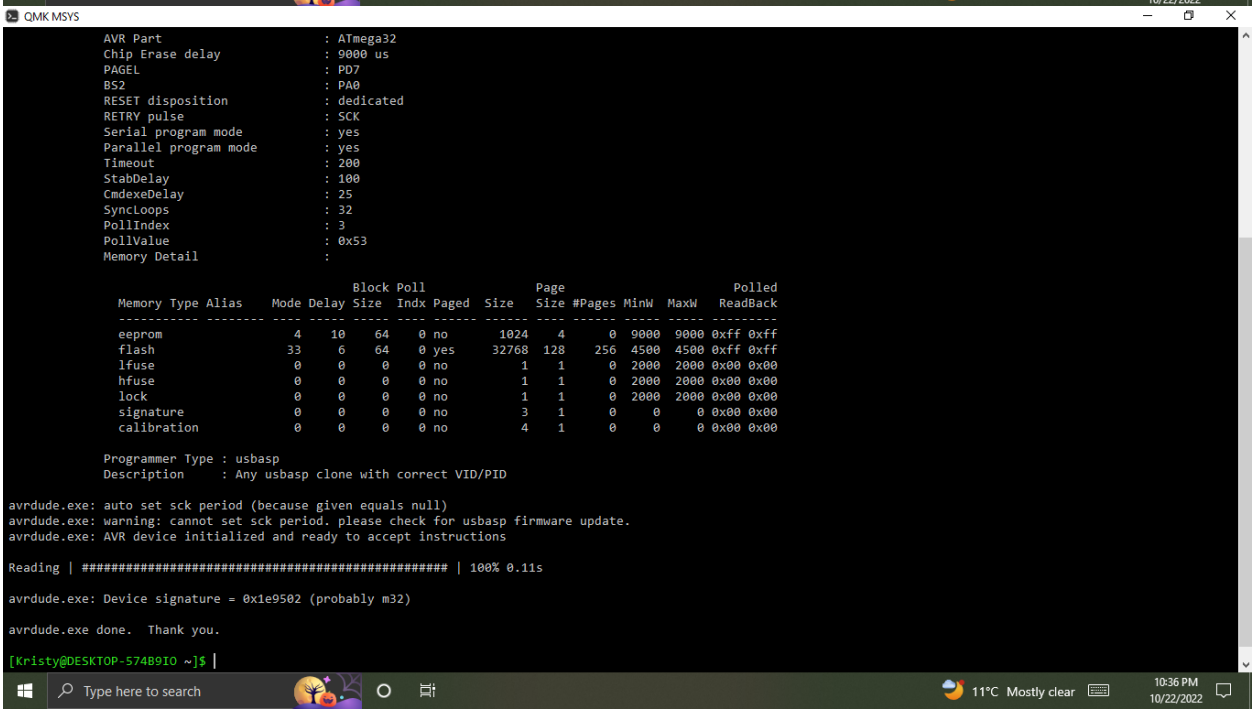
Programming The MCU With Commands (the annoying part):

1. Making sure your mcu chip exist by getting the info on it by typing the following commands:

avrdude -c usbasp-clone -b 19200 -p atmega32 -v

You should see the following screen.

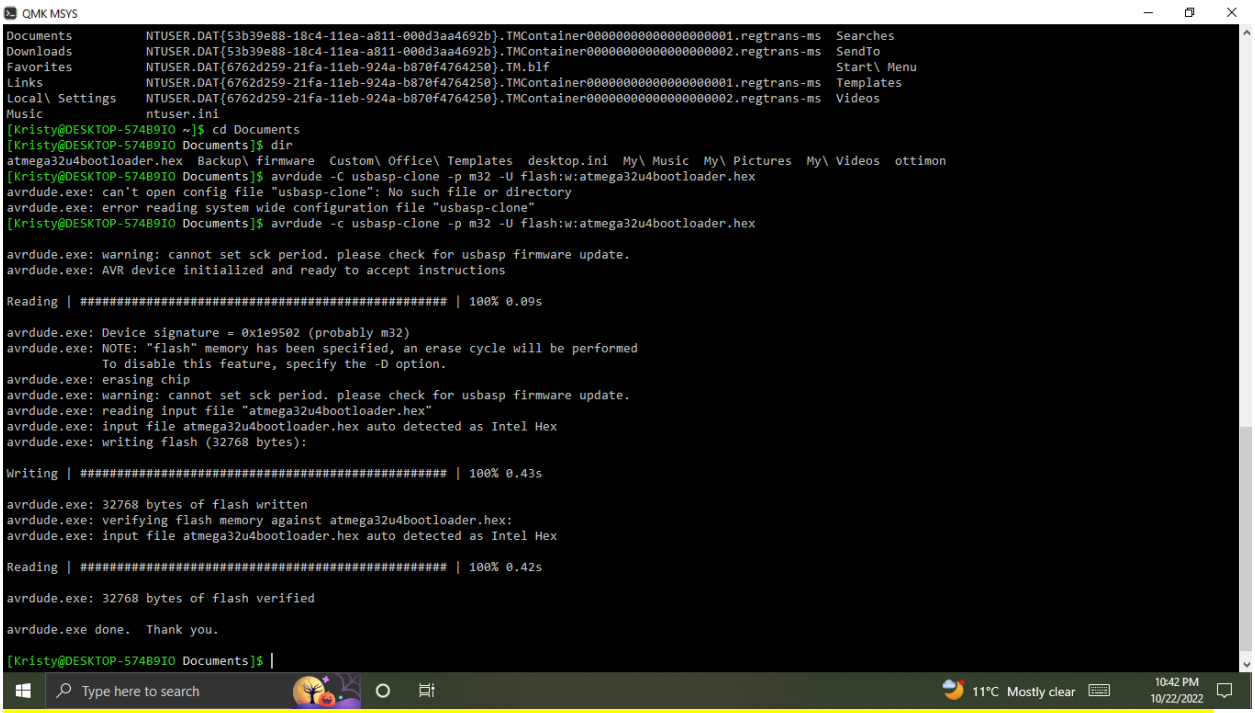




When you look closely, you will see there is a warning on the set SCK period and check for usbasp firmware update. Ignore that message since all clone AVR programmer will have this issue but it will not affect the bootloader flashing ability. So now the info that it display is correct and that the mcu is detected and is a ATMEGA32a.

2. Next flash the bootloader for ATMEGA32a. Make sure you pick the correct bootloader HEX file from Red Herring github page. Type the following command to flash the mcu.

avrdude -c usbasp-clone -p m32 -U flash:w:bootloader.hex:i

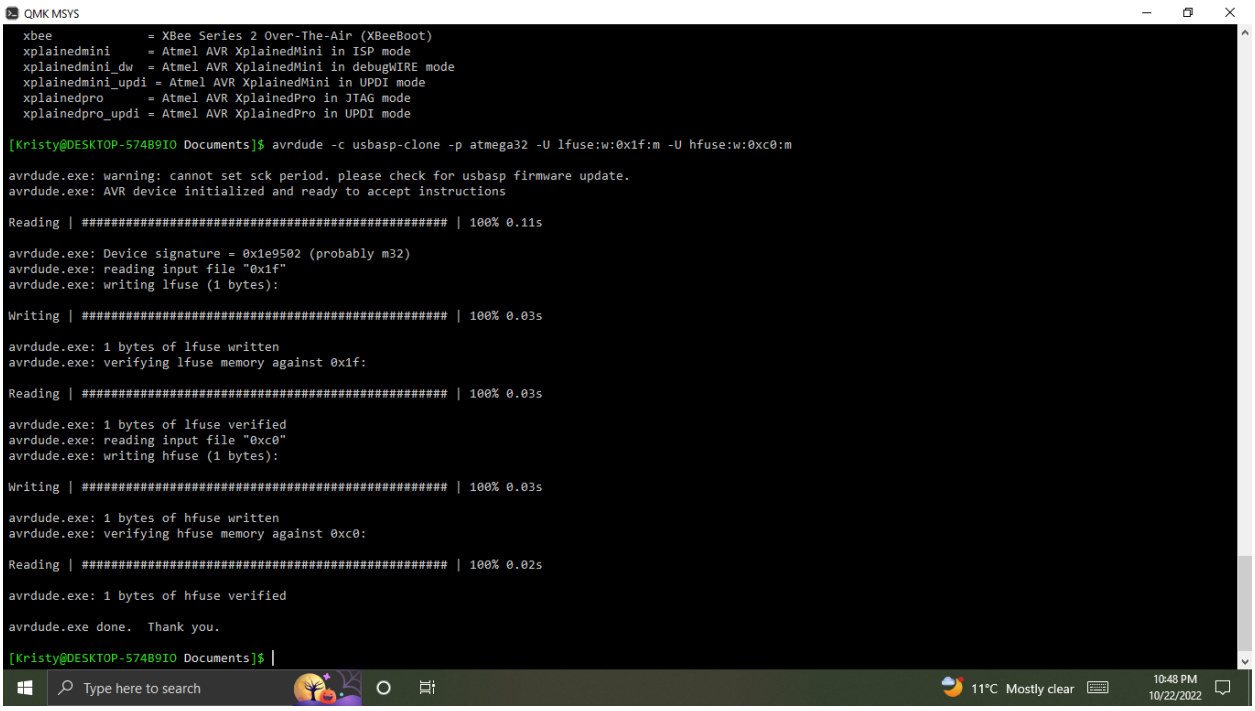


Ignore the file name atmega32u4bootloader.hex since I flashed the wrong bootloader before but the overall output screen should be the same. You just want to make sure at the end of the output, it says “avrdude.exe: XXXX bytes of flash verified”. Again ignore the set SCK period warning as per discussed above in this guide.

3. Next set the fuses for the mcu. Warning, make sure you use the correct fuse settings. Once set, they are set and cannot be change by using this command method. Changing set fuses require some kind of different programming method. You only set the fuses once, and once they are set, the fuse settings will stick to the mcu. Type the command to set the fuses as follows:

avrdude -c usbasp-clone -p atmega32 -U lfuse:w0x1f:m -U hfuse:w:0xc0:m

Output screen should be like below:

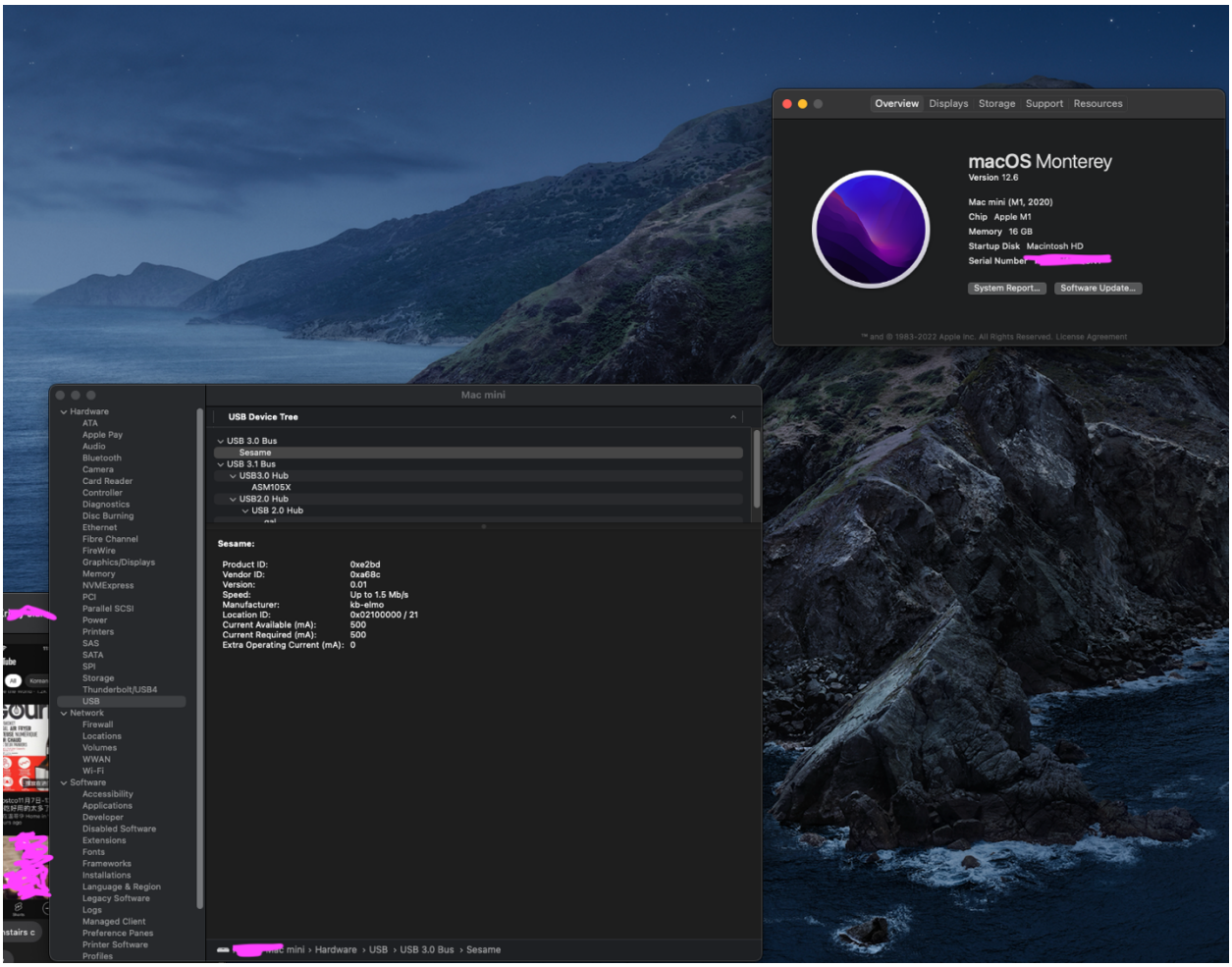


There should be 2 sets of fuses that are set and verified. Also ignore the set SCK period warning as per discussed before.

4. Congratulations! You have completed the hard and annoying part of the process.

5. Next stop is to flash the keyboard firmware hex to the mcu chip (This guide assumes you know how to use the QMK Toolbox to flash your keyboard firmware) once you installed the mcu chip onto the bracket socket on your keyboard pcb and plug in the usb c to usb a to your pc.

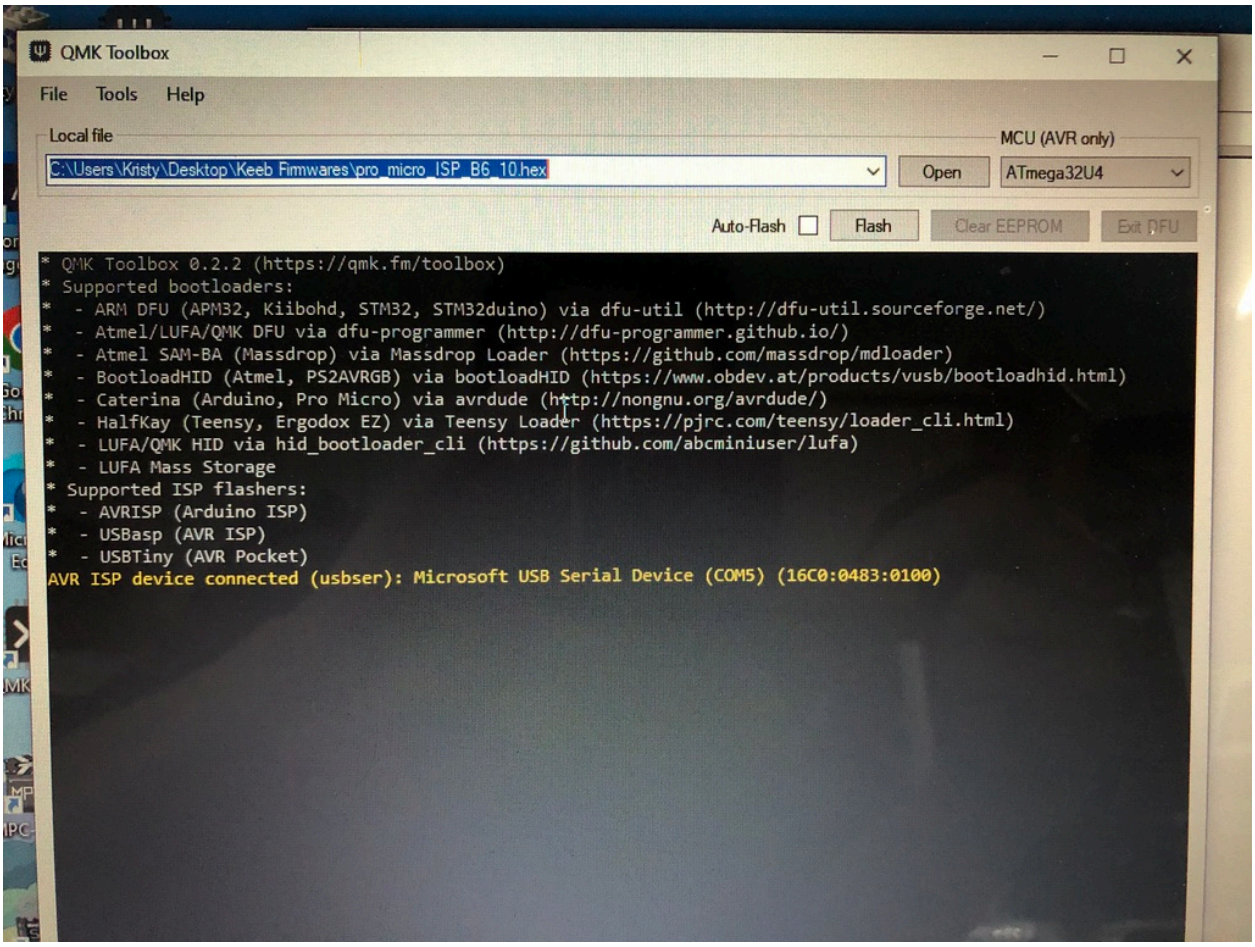
6. Once you flashed the firmware successfully and plugged into your pc, you should see the keyboard getting recognized and shows up in the USB device section. See screenshot below and my system is a MacOS but windows will have the device display in the device manager.

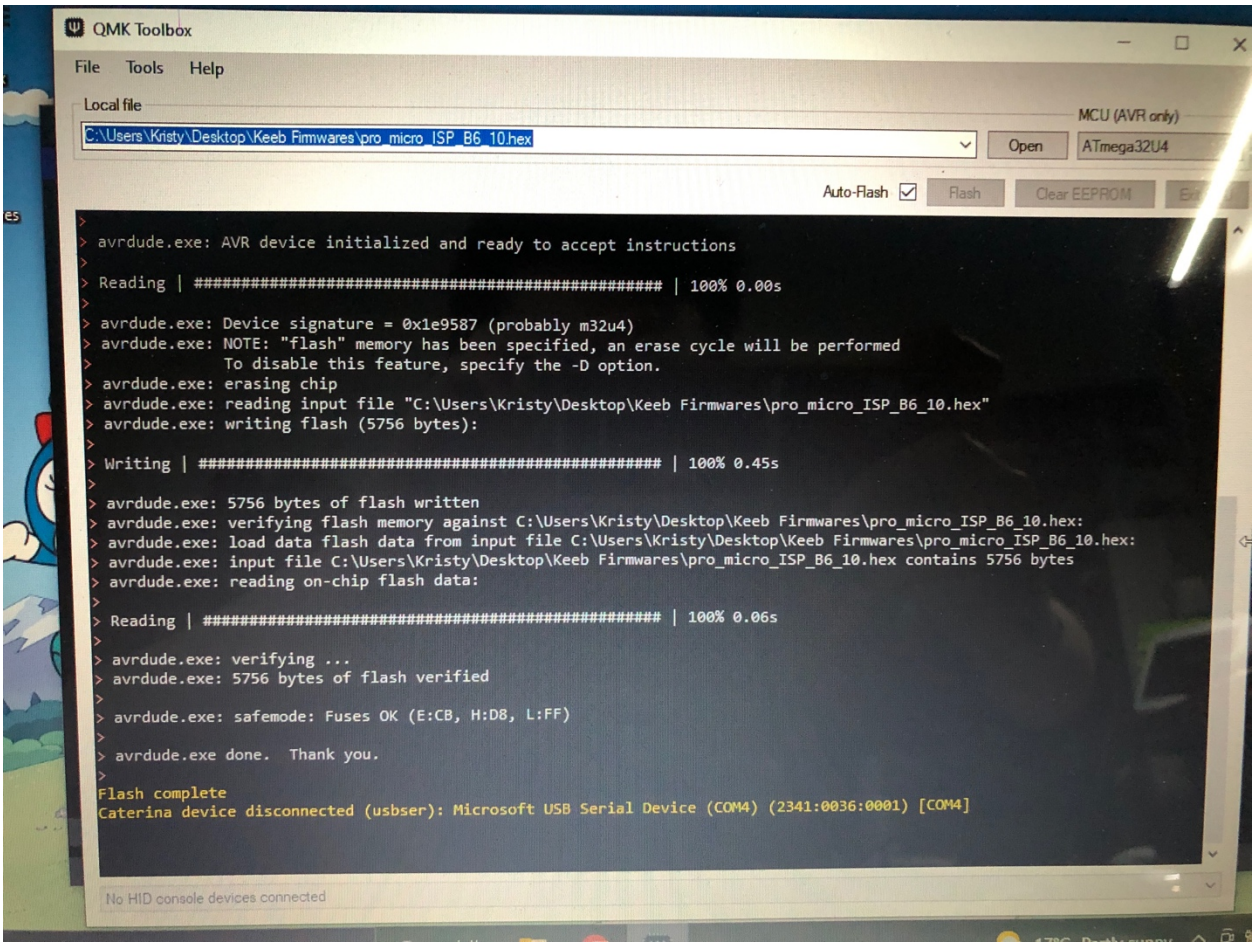


Method 2: Using Pro Micron as a AVR programmer.

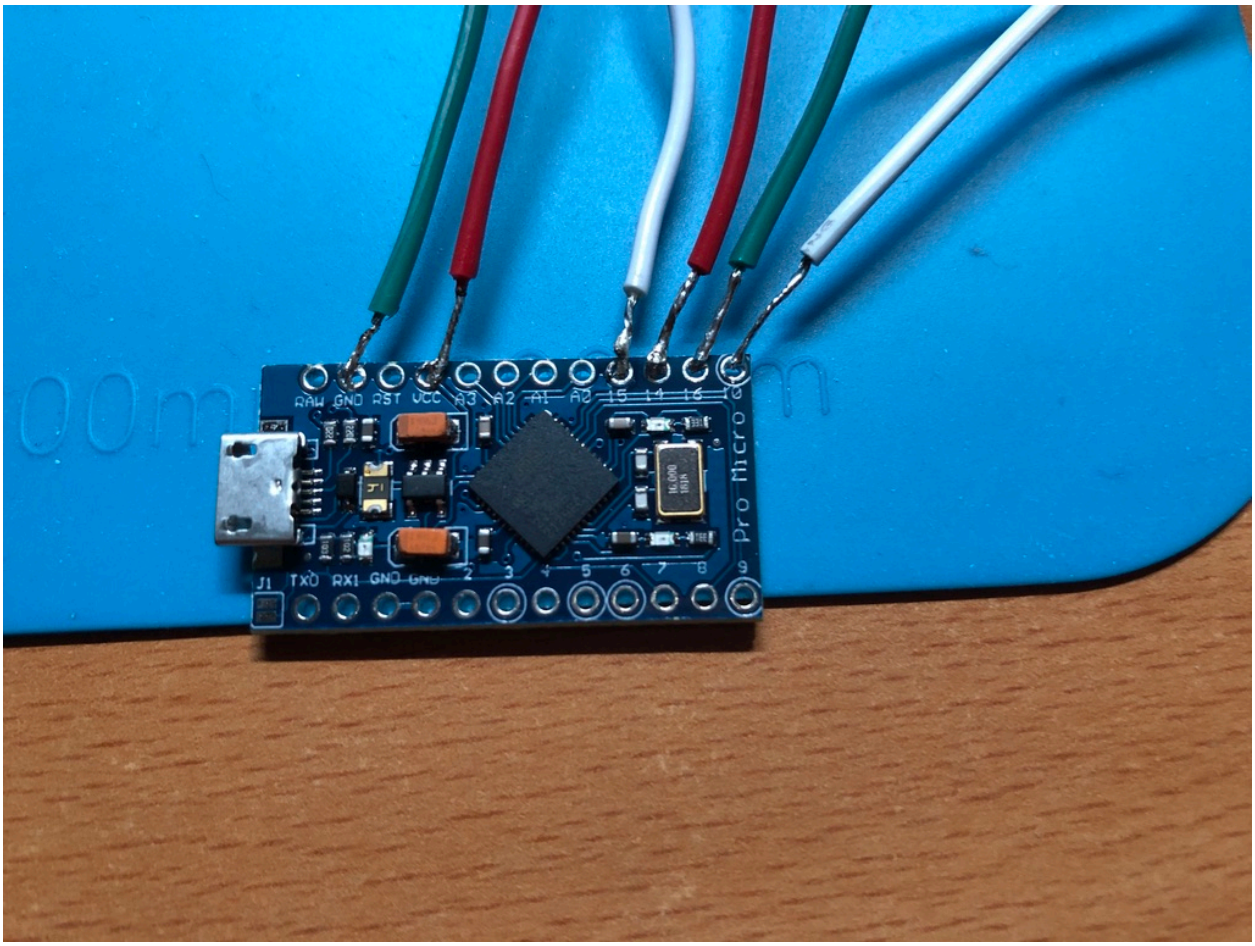
1. Before soldering the wires onto the Pro Micron, we need to flash the appropriate firmware onto it so that it can mimic a virtual USB port for the ATMEGA32a.

2. Using QMK Toolbox to flash the firmware called pro_micro_ISP_B6_10.hex taken from Red Herring's page to the Pro Micron. The result will be like the screenshot below. I am assuming you already know how to use the QMK Toolbox to flash the firmware by now so I will not cover it in this guide.


QMK Toolbox
File
Tools
Help
Local file
C:\Users\Kristy\Desktop\Keeb Fimwares\pro_micro_ISP_B6_10.hex
Open
MCU (AVR only)
ATmega32U4
Auto-Flash
Flash
Clear EEPROM
Exit DFU
* QMK Toolbox 0.2.2 (https://qmk.fm/toolbox)
* Supported bootloaders:
* - ARM DFU (APM32, Kiibohd, STM32, STM32duino) via dfu-util (http://dfu-util.sourceforge.net/)
* - Atmel/LUFA/QMK DFU via dfu-programmer (http://dfu-programmer.github.io/)
* - Atmel SAM-BA (Massdrop) via Massdrop Loader (https://github.com/massdrop/mdloader)
* - BootloadHID (Atmel, PS2AVRGB) via bootloadHID (https://www.obdev.at/products/vusb/bootloadhid.html)
* - Caterina (Arduino, Pro Micro) via avrdude (http://nongnu.org/avrdude/)
* - HalfKay (Teensy, Ergodox EZ) via Teensy Loader (https://pjrc.com/teensy/loader_cli.html)
* - LUFA/QMK HID via hid_bootloader_cli (https://github.com/abcminiuser/lufa)
* - LUFA Mass Storage
* Supported ISP flashers:
* - AVRISP (Arduino ISP)
* - USBasp (AVR ISP)
* - USBTiny (AVR Pocket)
AVR ISP device connected (usbser): Microsoft USB Serial Device (COM5) (16C0:0483:0100)



3. Lightly tack solder the wires or use some method to secure the wires to the selected pin holes on the Pro Micron as per picture below:

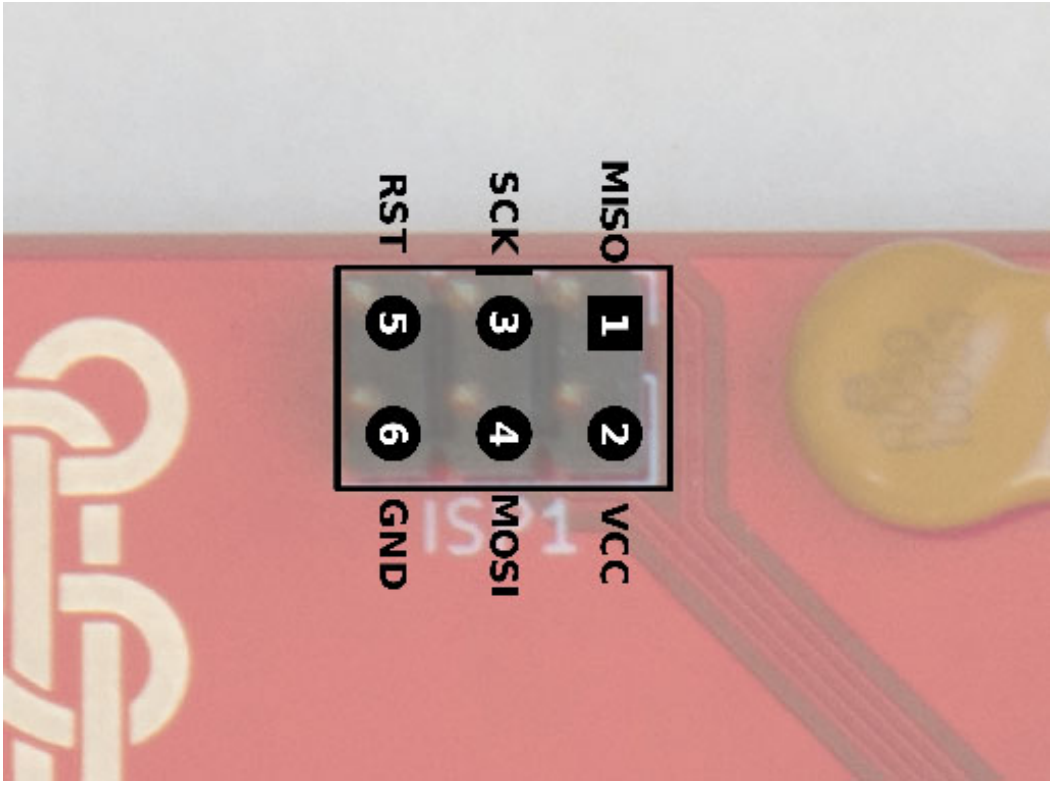


As per QMK ISP flash guide, you would need the following pin holes for use as shown below:

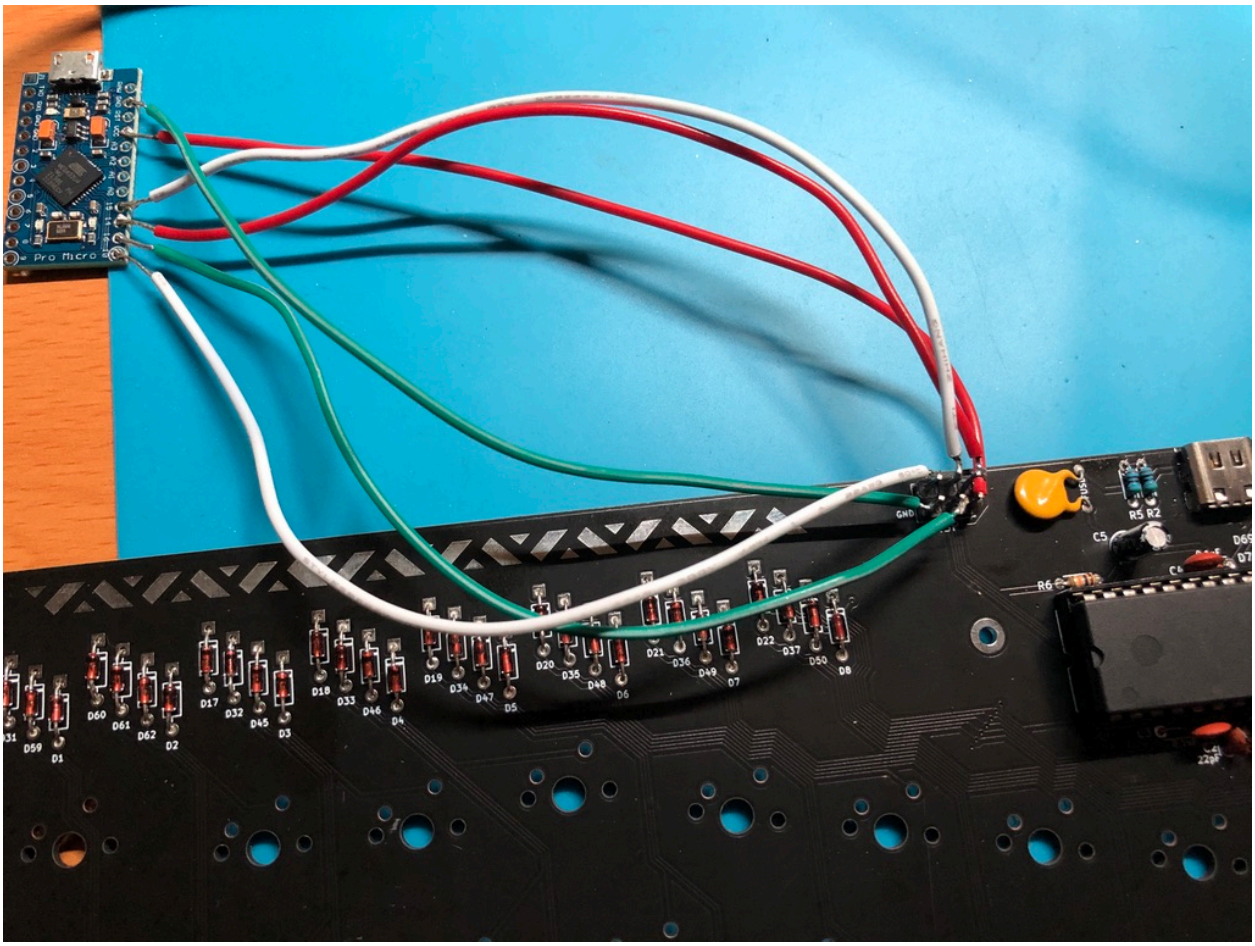
Wiring

Pro Micro	Keyboard
VCC	VCC
GND	GND
10 (B6)	RESET
15 (B1)	SCLK
16 (B2)	MOSI
14 (B3)	MISO

Note: The () ones are for Elite-C mcu unit since they have different pin holes label. So each pin holes correspond to the corresponding ISP pins on the keyboard pcb side. Usually the keyboard side will have the pins labeled. Basketweave will have the same pin layout as the Red Herring pcb as shown below (Image is taken from dcpedit's github Red Herring firmware page):



4. Use the wiring table above from the Pro Micron side to connect to the ISP pins on the keyboard side. Your final connections should look like this.

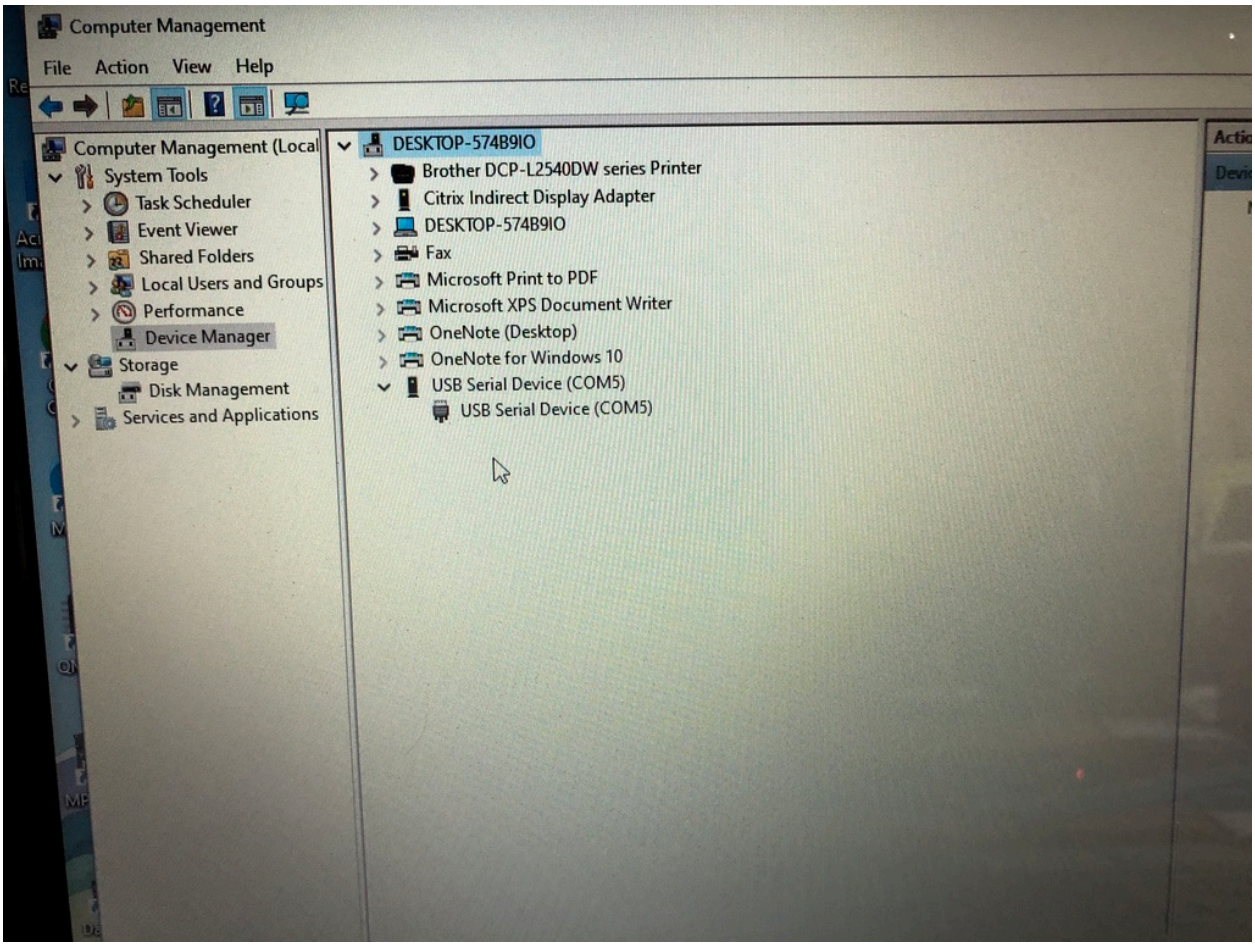


I didn't have the jumper wires, so I had to lightly solder every wire connections.

5. Once you securely connected all the wires to their corresponding targets, and plug in the PM to the PC side and you should see LED lights for both units. Some keyboards like the Sesame does not have a LED on their pcb.

6. Before you can start using the commands to flash a bootloader, you first need to find the COM port that the PM is using via Device Manager on the Windows machine. I don't about the Mac side since I did it on a Windows machine and is easier.

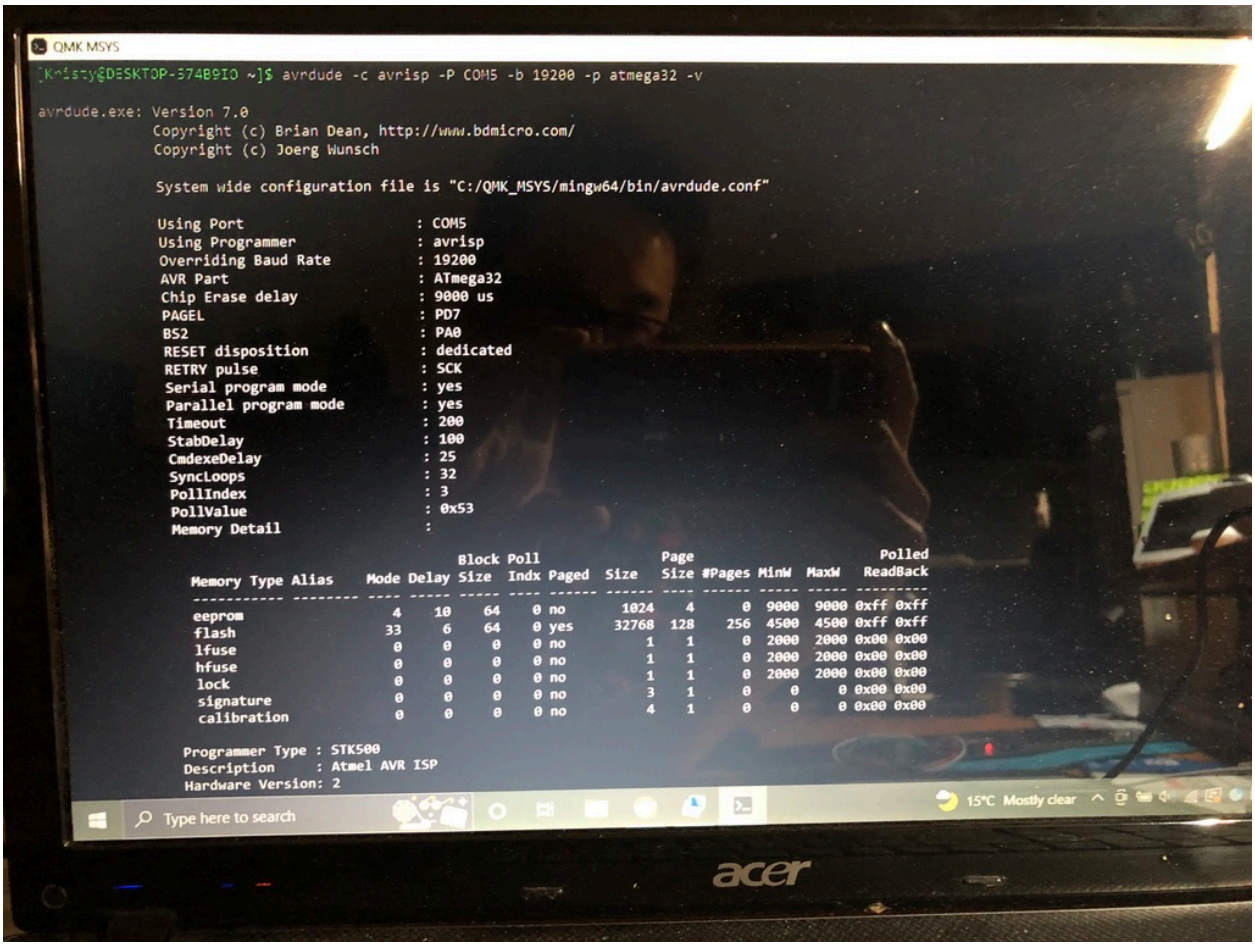
7. Right click your windows PC icon and select Device or Computer management. Then click on View and select group by container and you will see your USB Serial Device with a COM number beside it. This is your PM.

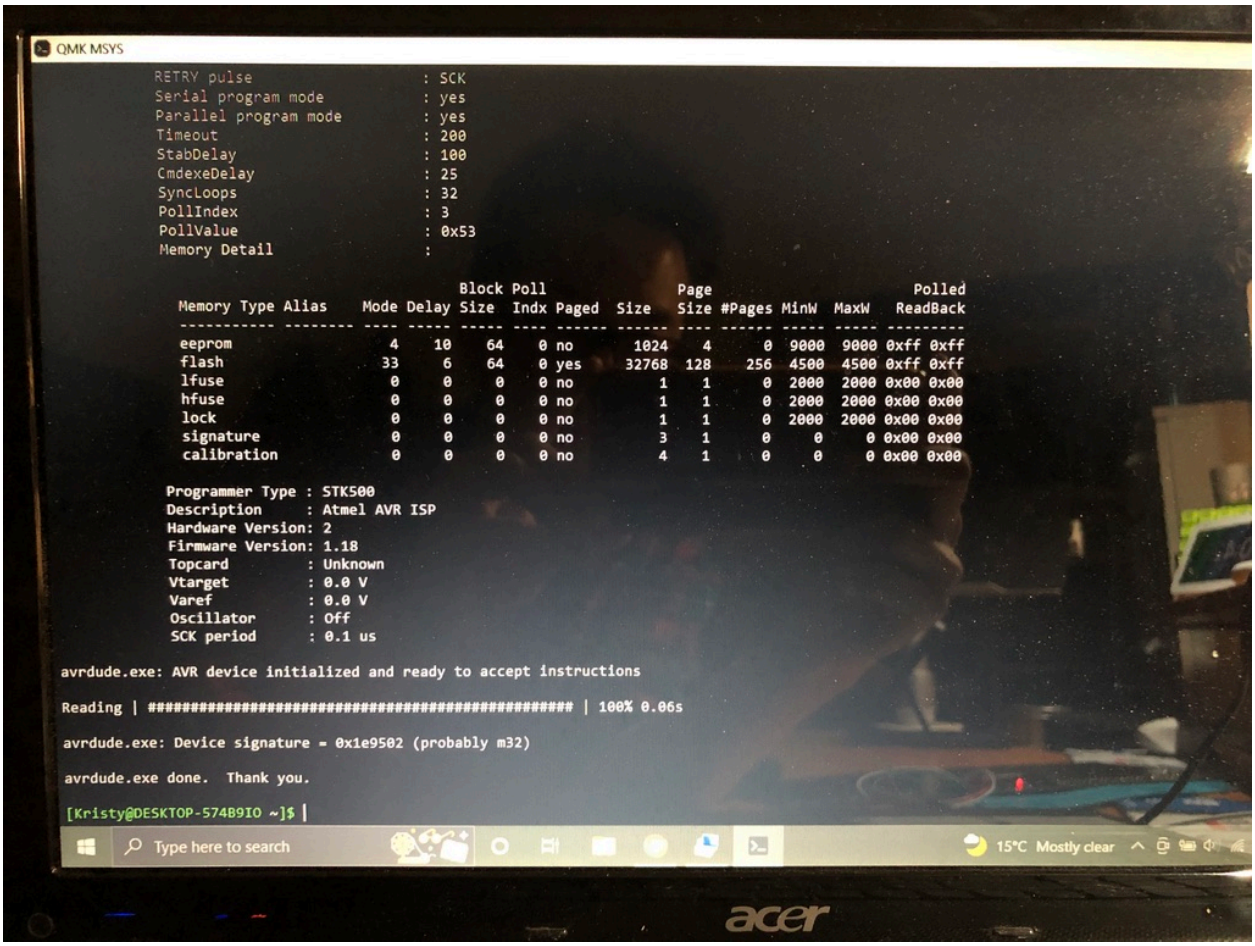


8. Then type the follow command to see if the PC detects the ATMEGA32a on the pcb or not:

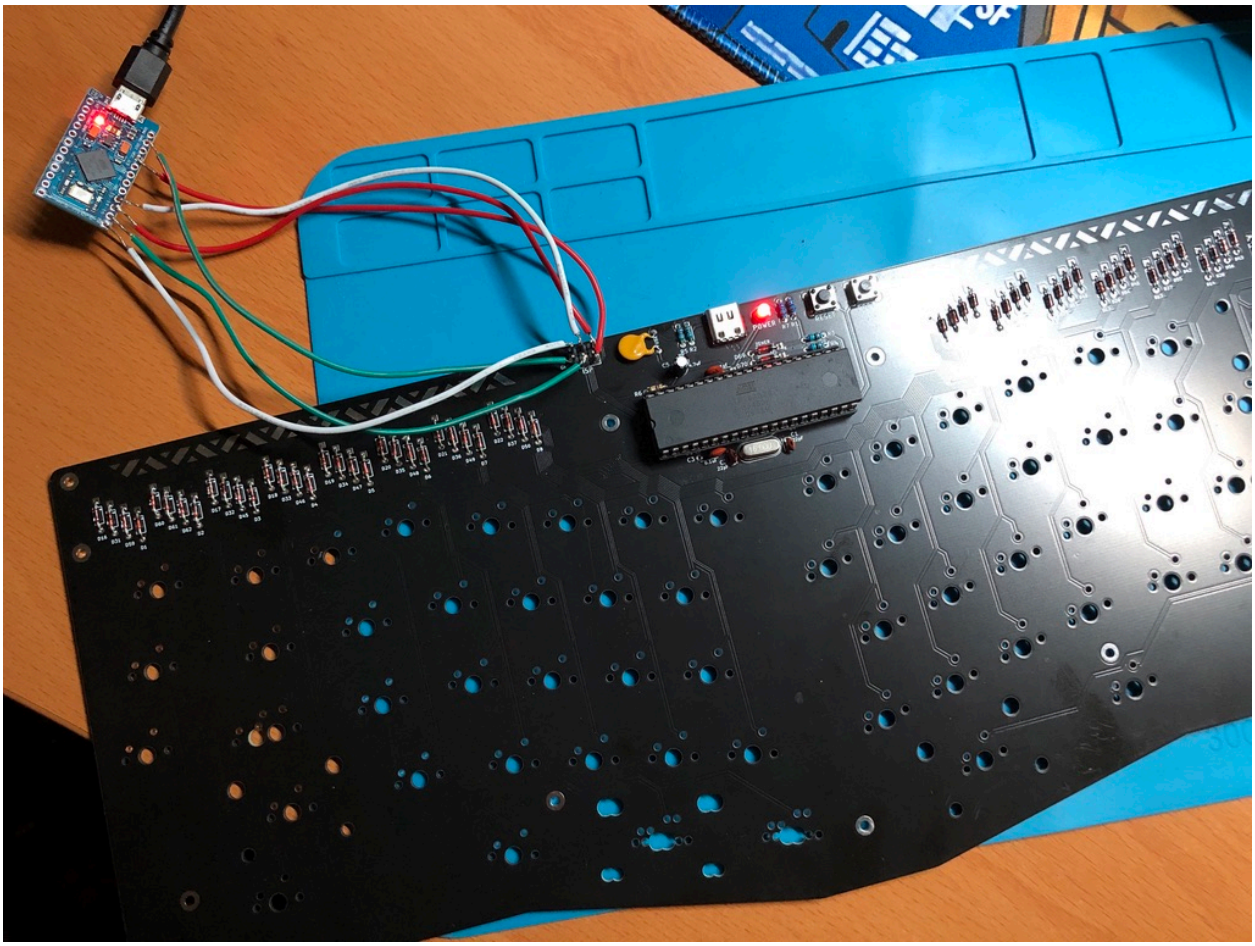
avrdude -c avrisp -P COM5 -b 19200 -p atmega32 -v

You should similar info screenshot as below:


QMK MSYS
[K^isty@DESKTOP-374B9IO ~]$ avrdude -c avrisp -P COM5 -b 19200 -p atmega32 -v
avrdude.exe: Version 7.0
Copyright (c) Brian Dean, http://www.bdmicro.com/
Copyright (c) Joerg Wunsch
System wide configuration file is "C:/QMK_MSYS/mingw64/bin/avrdude.conf"
Using Port : COM5
Using Programmer : avrisp
Overriding Baud Rate : 19200
AVR Part : ATmega32
Chip Erase delay : 9000 us
PAGEL : PD7
BS2 : PA0
RESET disposition : dedicated
RETRY pulse : SCK
Serial program mode : yes
Parallel program mode : yes
Timeout : 200
StabDelay : 100
CmdexeDelay : 25
SyncLoops : 32
PollIndex : 3
PollValue : 0x53
Memory Detail :
Memory Type Alias Mode Delay Block Size Poll Indx Paged Size Page Size #Pages MinW MaxW Polled ReadBack
eeprom 4 10 64 0 no 1024 4 0 9000 9000 0xff 0xff
flash 33 6 64 0 yes 32768 128 256 4500 4500 0xff 0xff
lfuse 0 0 0 0 no 1 1 0 2000 2000 0x00 0x00
hfuse 0 0 0 0 no 1 1 0 2000 2000 0x00 0x00
lock 0 0 0 0 no 1 1 0 2000 2000 0x00 0x00
signature 0 0 0 0 no 3 1 0 0 0 0x00 0x00
calibration 0 0 0 0 no 4 1 0 0 0 0x00 0x00
Programmer Type : STK500
Description : Atmel AVR ISP
Hardware Version: 2
Type here to search
15°C Mostly clear
acer

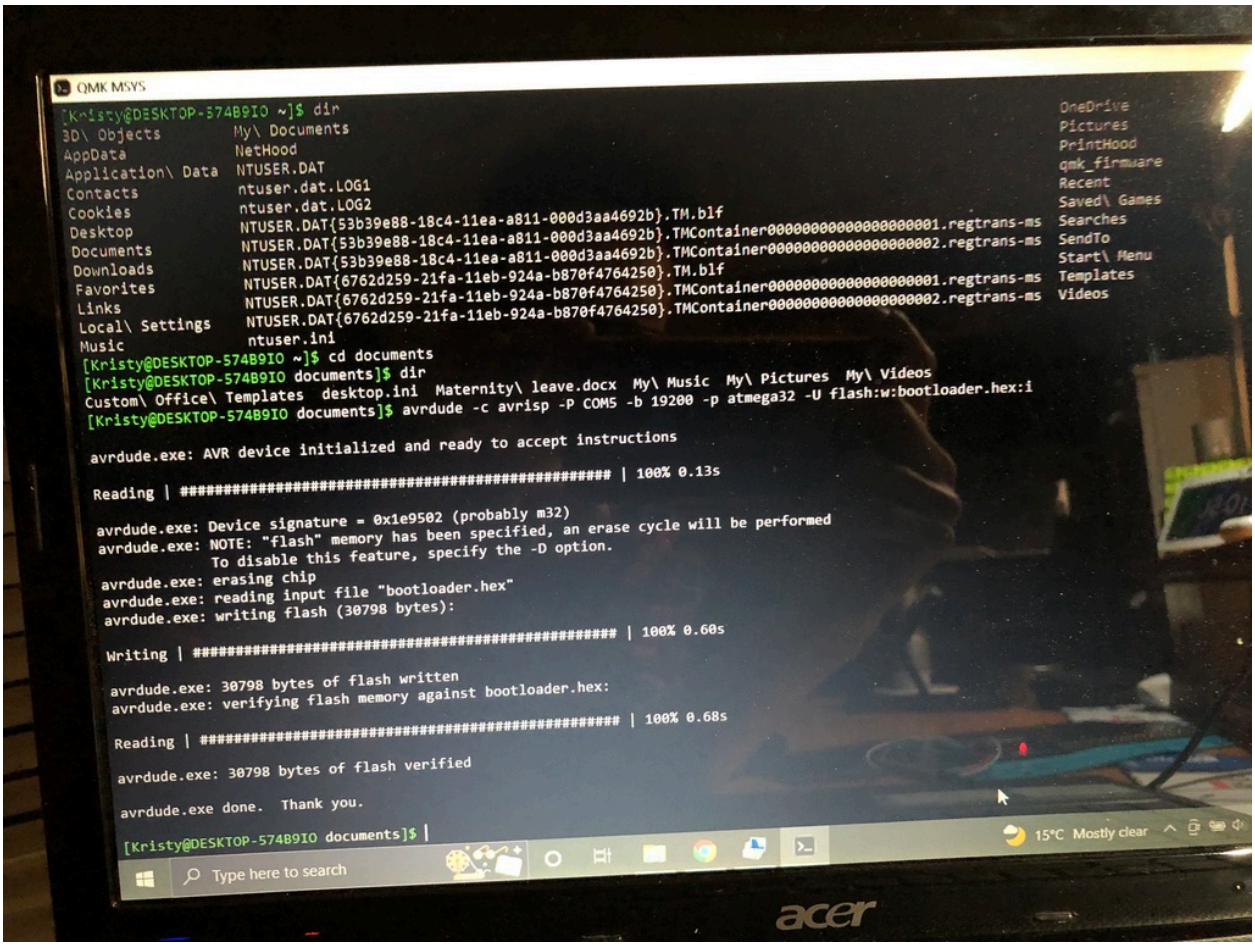


Note: No more set SCK period warning for this method and sorry about the screenshot since I was using my wife's laptop and too lazy to screen it.



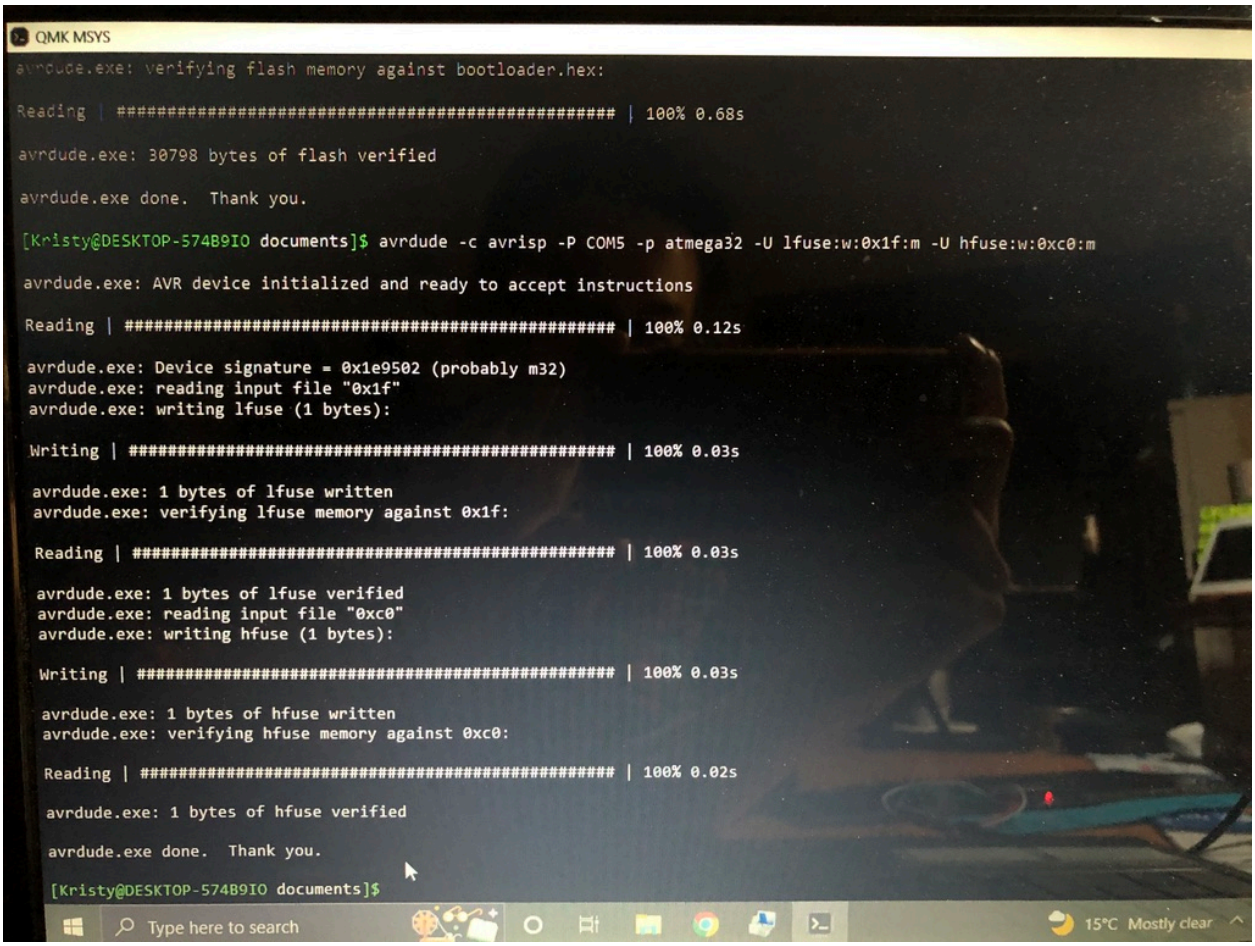
9. Flash the correct bootloader on the mcu by using the following command:

avrdude -c avrisp -P COM5 -b 19200 -p atmega32 -U flash:w:bootloader.hex:i

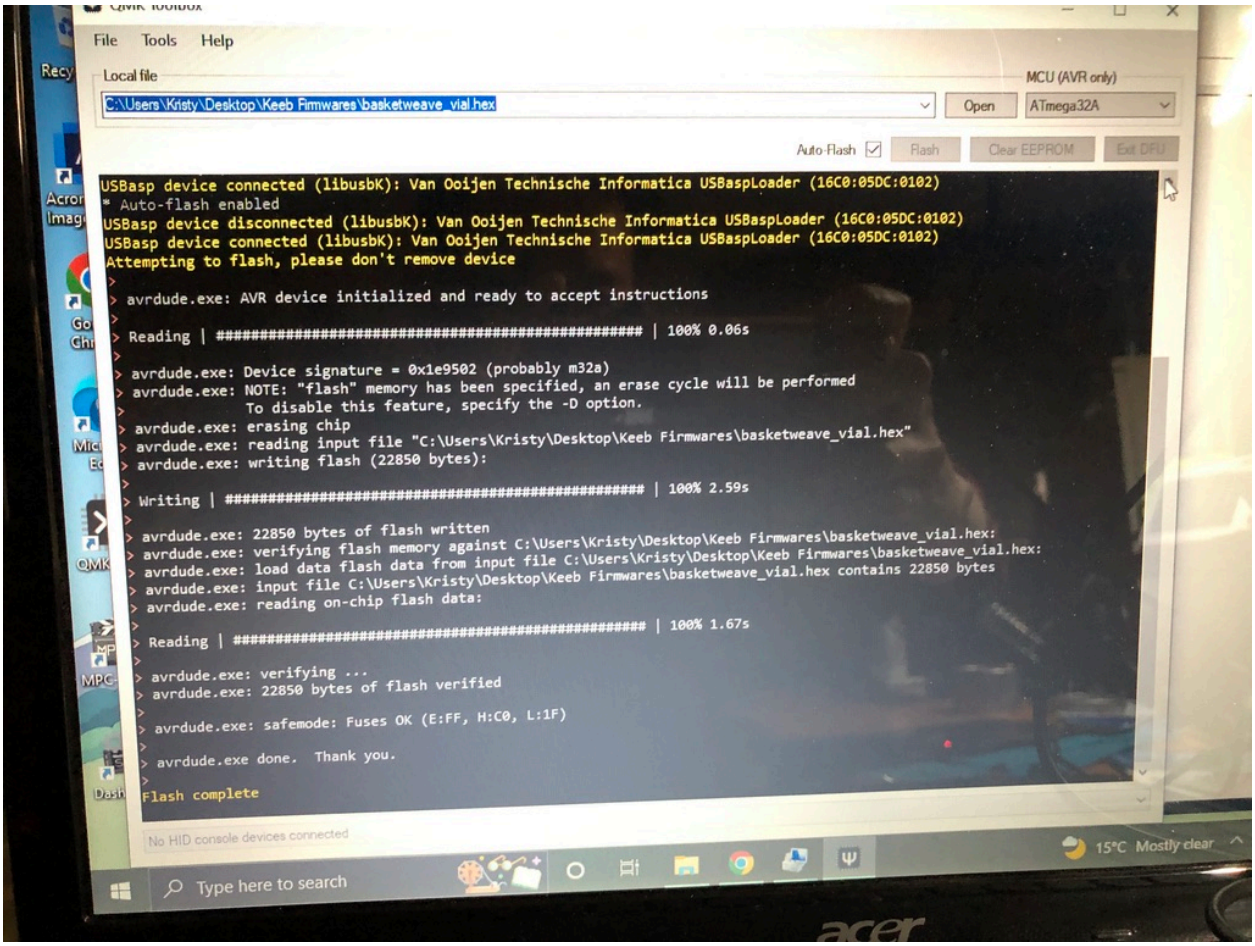


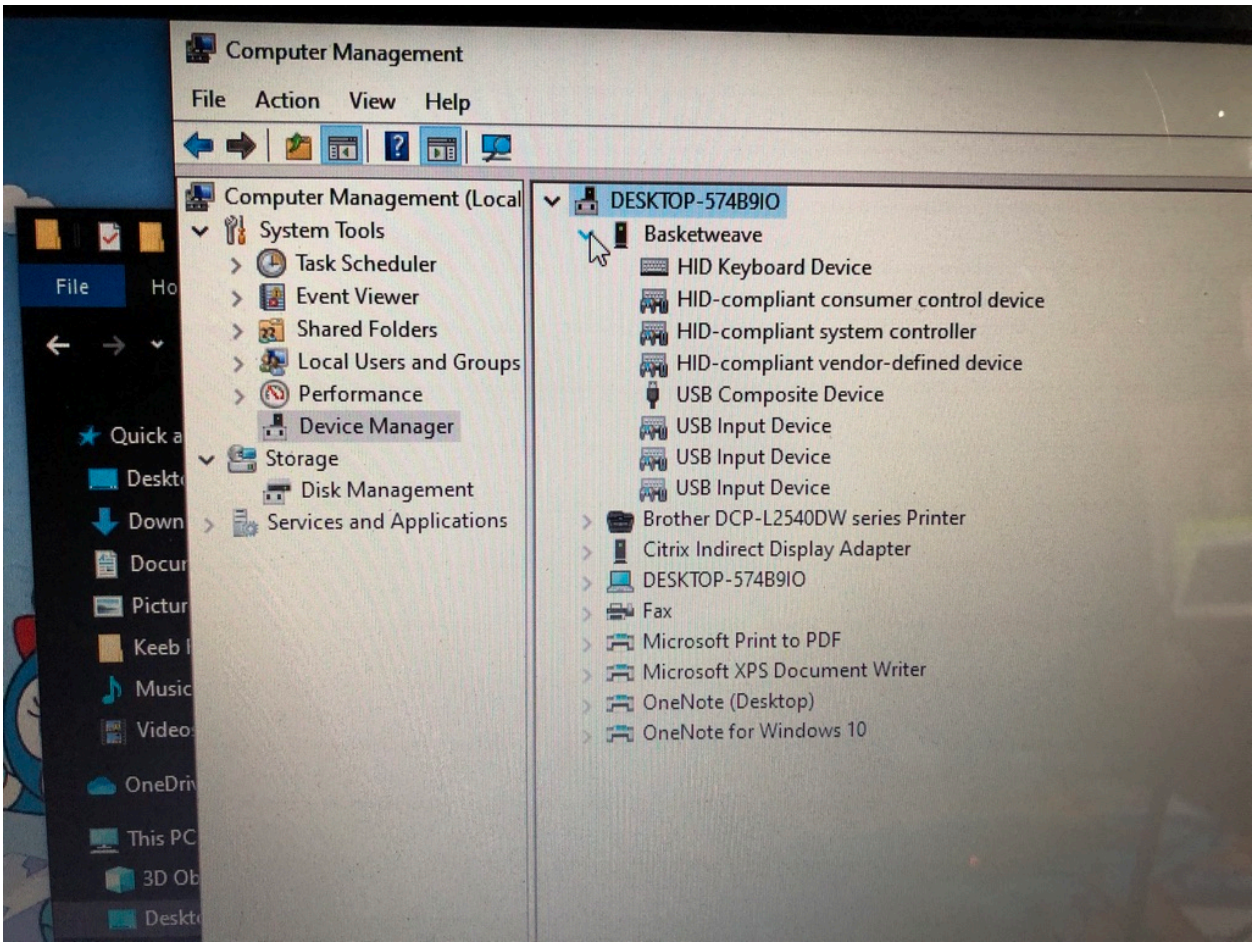
10. Set the fuses as per below command:

avrdude -c avrisp -P COM5 -p atmega32 -U lfuse:w:0x1f:m -U hfuse:w:0xc0:m



11. Now with both the bootloader and the fuses flashed, flash your keyboard firmware onto it by disconnecting all your setup. Make sure no wires on your ISP are attached to anything. De-solder the wires if you have to. Then connect your USB C to USB A to your PC and the moment of truth comes. Go to your device management on your PC and see. For MACs, a keyboard setup program will pop up.


File Tools Help
Local file
C:\Users\Kristy\Desktop\Keeb Firmwares\basketweave_vial.hex
Open
MCU (AVR only)
ATmega32A
Auto-Flash
Flash
Clear EEPROM
Exit DFU
USBasp device connected (libusbK): Van Ooijen Technische Informatica USBaspLoader (16C0:05DC:0102)
* Auto-flash enabled
USBasp device disconnected (libusbK): Van Ooijen Technische Informatica USBaspLoader (16C0:05DC:0102)
USBasp device connected (libusbK): Van Ooijen Technische Informatica USBaspLoader (16C0:05DC:0102)
Attempting to flash, please don't remove device
> avrdude.exe: AVR device initialized and ready to accept instructions
> Reading | ## | 100% 0.06s
> avrdude.exe: Device signature = 0x1e9502 (probably m32a)
> avrdude.exe: NOTE: "flash" memory has been specified, an erase cycle will be performed
> To disable this feature, specify the -D option.
> avrdude.exe: erasing chip
> avrdude.exe: reading input file "C:\Users\Kristy\Desktop\Keeb Firmwares\basketweave_vial.hex"
> avrdude.exe: writing flash (22850 bytes):
> Writing | ## | 100% 2.59s
> avrdude.exe: 22850 bytes of flash written
> avrdude.exe: verifying flash memory against C:\Users\Kristy\Desktop\Keeb Firmwares\basketweave_vial.hex:
> avrdude.exe: load data flash data from input file C:\Users\Kristy\Desktop\Keeb Firmwares\basketweave_vial.hex:
> avrdude.exe: input file C:\Users\Kristy\Desktop\Keeb Firmwares\basketweave_vial.hex contains 22850 bytes
> avrdude.exe: reading on-chip flash data:
> Reading | ## | 100% 1.67s
> avrdude.exe: verifying ...
> avrdude.exe: 22850 bytes of flash verified
> avrdude.exe: safemode: Fuses OK (E:FF, H:C0, L:1F)
> avrdude.exe done. Thank you.
Flash complete
No HID console devices connected
Type here to search
15°C Mostly clear
acer



CONGRATS! You have survived the flashing process and make your keeb a functional working device instead of an expensive paper weight.