Terminator Hunter Killer Robot

By  Alex Eastabrook

Hi All at Imagination Tech,

My Idea for this competition is to use the Ci20 to drive this Terminator Robot. It is modelled after the T-1 Battlefield Robot. My idea is to use the Ci20 to connect to various I/O peripherals and provide the main hub for controlling them.

I have specified various interesting peripherals in my initial sketch such as 4 PIR sensors to detect heated movement. This would not allow anyone on foot to approach the robot undetected. It would also be pretty fun to see the T-1 turn and face you if you try and sneak up on it.

I have chosen to put basic Ultrasonic Rangefinders on the front and back of the unit to allow it to sense how far away it is from any walls with relative ease. This will make my job easier if I try and provide it some pathfinding algorithms.

The Ci20 Has Wi-Fi so initially I would be looking at just using this in a remote control context with some additional programmed logic.

I would be 3D Printing the Terminator Head with a slot for a Webcam (Ci20 Webcam Interface) to enable the Robot to serve as a IP Camera through the Ci20 as well.

The Nerf Dart Launcher on the shoulder will just be a ‘Buy a USB Dart Launcher’ from Amazon and fit it to work.

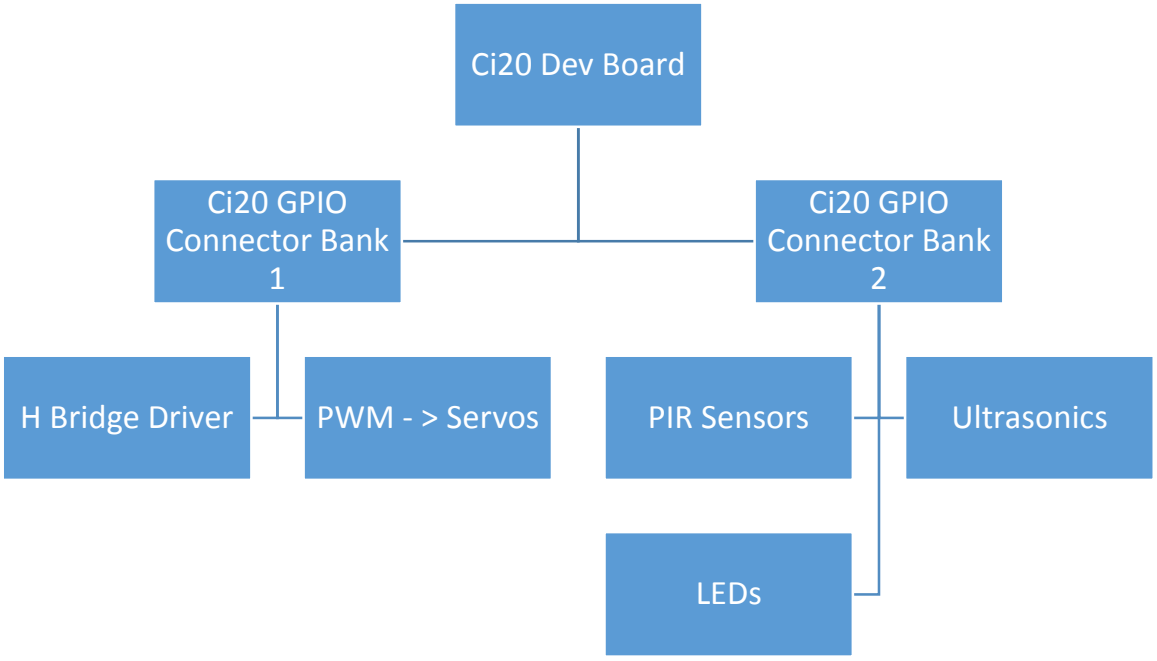
I have homebuilt CNC mill and 3D printers in my workshop so rapid development and fabrication of parts would not prove to be a problem. I work as an embedded software engineer so the interfacing and control should not be a problem either.

First off.

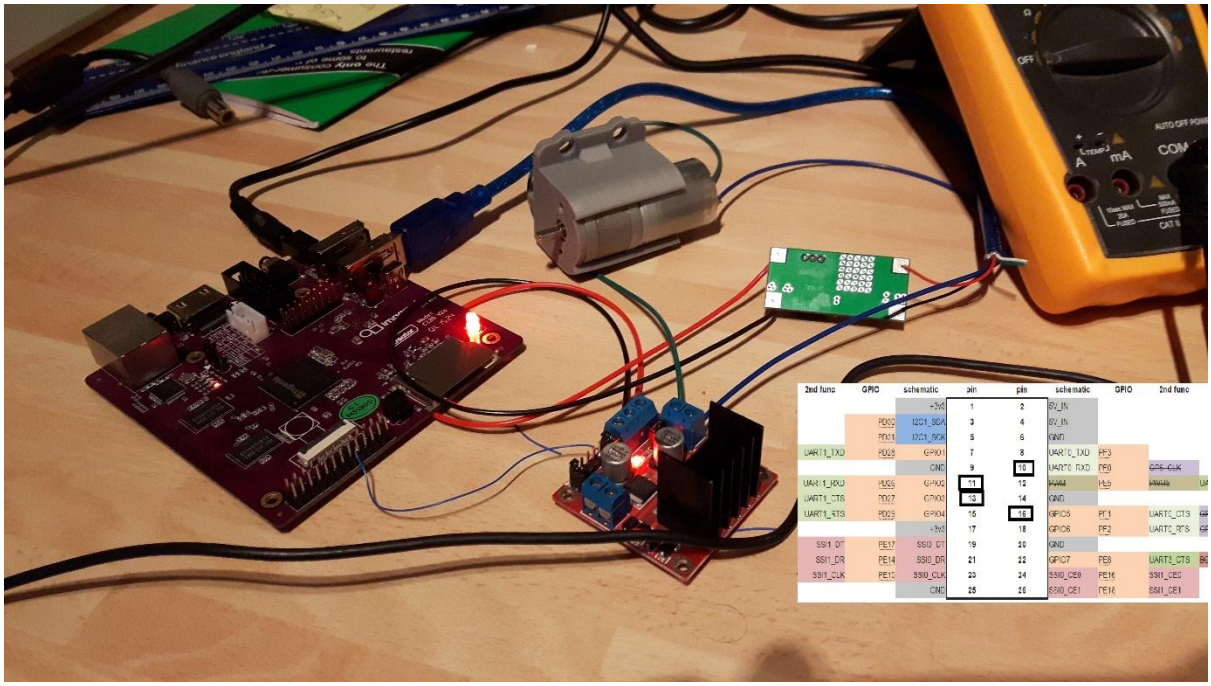
I had been tinkering around with the Ci20 for a few days before starting work, In this I updated to the latest version of Debian by SD card flash. This was needed as I couldn't get any of the LEDs on board to blink / show cpu activity.

With that done, I started thinking about how I would fill out my initial project spec.

The Initial plan was to architect the system like this.



I set about doing this, wiring up the Ci20 to my H Bridge driver – With a motor attached.



I went to http://elinux.org/CI20_GPIO_LED_Blink_Tutorial and got the hello world code.

```
cd /sys/class/gpio/
echo 124 > export #enable access to the GPIO
cd gpio124/ #enter new directory
cat direction #check urrent direction
echo out > direction #Set GPIO as output
cat direction #Check new GPIO direction
cat value #Check current value
echo 1 > value #Set value as high as a test, LED should lit-up
cat value #Check that value changed
#Endless loop, 1 sec on plus 1 sec off, press CTRL+C to end
while ( true ); do echo 1 > value; cat value; sleep 1; echo 0 > value;
cat value; sleep 1; done;
```

I changed the value of 124 to be the appropriate pins for my application.

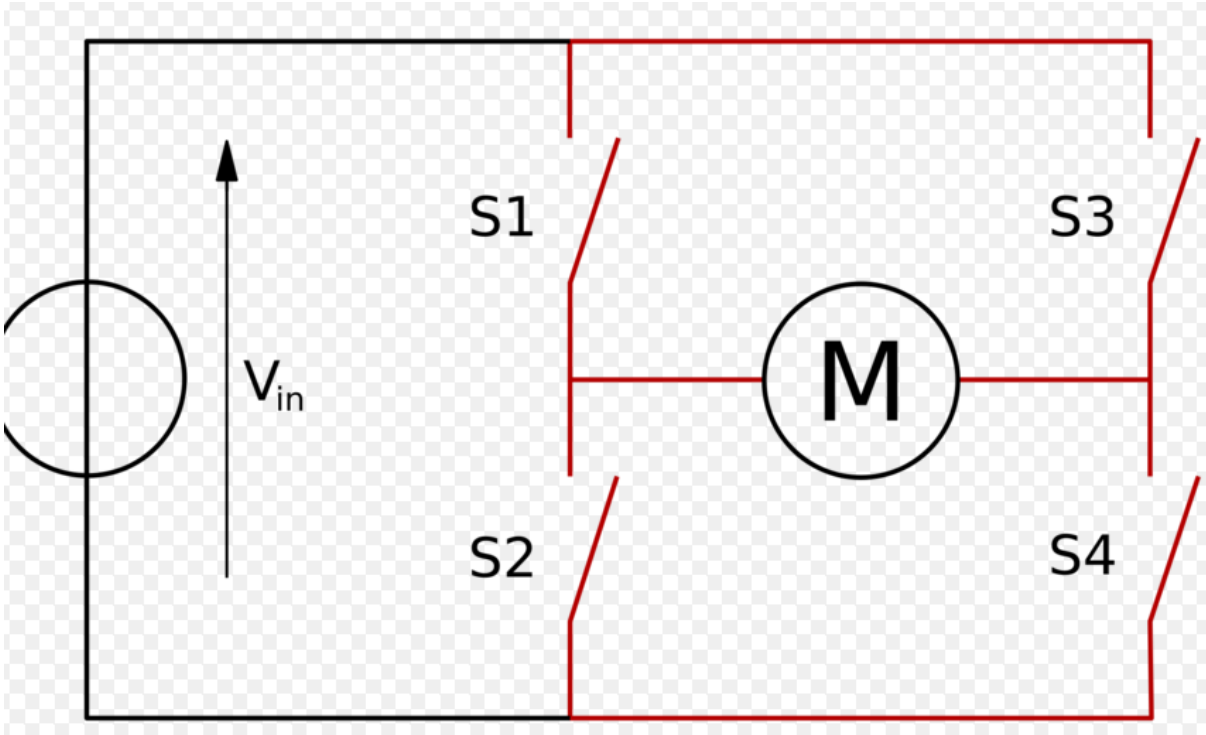
2nd func	GPIO	schematic	pin	pin	schematic	GPIO	2nd func	3rd func	4th func
		+3v3	1	2	5V_IN				
	PD30	I2C1_SDA	3	4	5V_IN				
	PD31	I2C1_SCK	5	6	GND				
UART1_TXD	PD28	GPIO1	7	8	UART0_TXD	PF3			
		GND	9	10	UART0_RXD	PF0	~~GPS_CLK~~		
UART1_RXD	PD26	GPIO2	11	12	~~PWM~~	PE5	~~PWM5~~	UART3_TXD	~~SCLK_RSTN~~
UART1_CTS	PD27	GPIO3	13	14	GND				
UART1_RTS	PD29	GPIO4	15	16	GPIO5	PF1	UART0_CTS	~~GPS_MAG~~	
		+3v3	17	18	GPIO6	PF2	UART0_RTS	~~GPS_SIG~~	
SSI1_DT	PE17	SSI0_DT	19	20	GND				
SSI1_DR	PE14	SSI0_DR	21	22	GPIO7	PE8	UART3_CTS	~~BCLK_AD~~	
SSI1_CLK	PE15	SSI0_CLK	23	24	SSI0_CE0	PE16	SSI1_CE0		
		GND	25	26	SSI0_CE1	PE18	SSI1_CE1		

I.e.

122, 123, 160, and 161. This allowed all my motors to rotate when the GPIO provided a signal to the driver. However, I thought.. “I wonder what state these pins are in when the system power is cycled.”

I used the reset button (With only one leg connected of the H Bridge) and it stated turning.

I took a good look around to see if there was any way I could put anything in the linux bootup scripts, but realised that this would probably take too long before the GPIO Level was lowered, and my H Bridge would have been fried by then.



It would have turned S1, S2, S3, and S4 on all at once. VDD - > GND. Unhappy driver and unhappy Alex result.

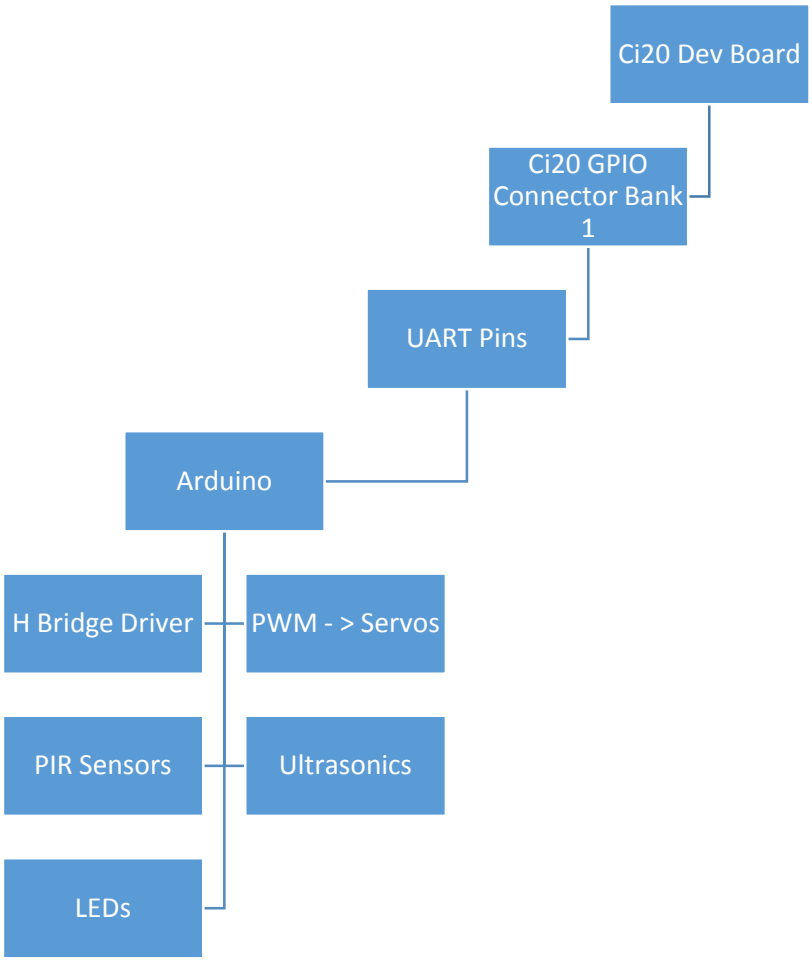
So I started thinking around the problem a bit. I could use a pin – and tie that to the enable line on the driver. But being unhappy at using another GPIO for the cause, elected to move on.

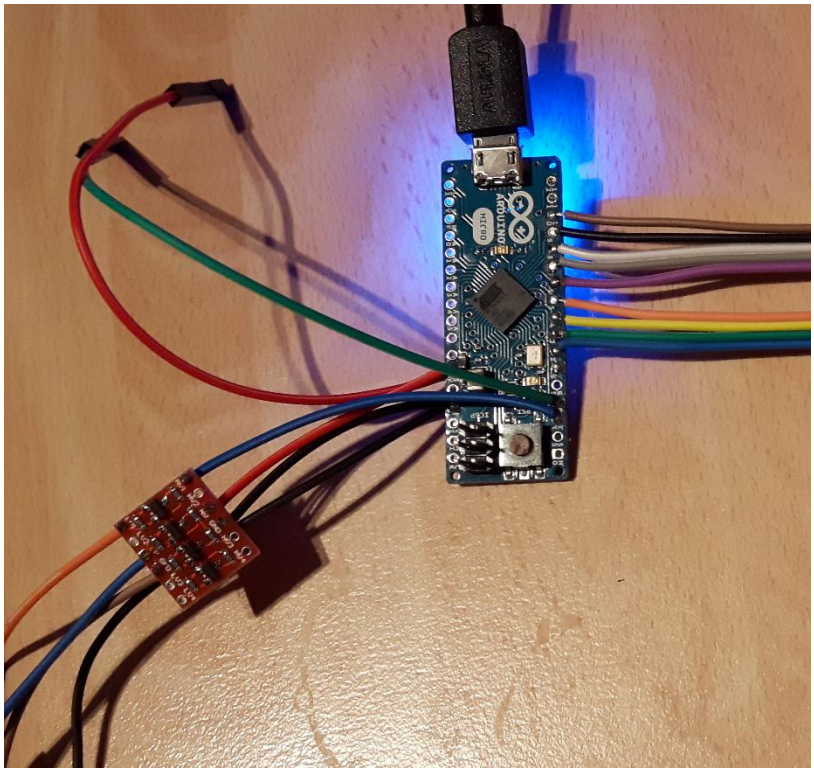
At this point, I start to think.. I can pass commands from the Ci20 to an Arduino off the board. I can even use the Ci20 UART GPIO pins to do this !

2nd func	GPIO	schematic	pin	pin	schematic	GPIO	2nd func	3rd func	4th func
		+3v3	1	2	5V_IN				
	PD30	I2C1_SDA	3	4	5V_IN				
	PD31	I2C1_SCK	5	6	GND				
UART1_TXD	PD28	GPIO1	7	8	UART0_TXD	PF3			
		GND	9	10	UART0_RXD	PF0	~~GPS_CLK~~		
UART1_RXD	PD26	GPIO2	11	12	~~PWM~~	PE5	~~PWM5~~	UART3_TXD	~~SCLK_RSTN~~
UART1_CTS	PD27	GPIO3	13	14	GND				
UART1_RTS	PD29	GPIO4	15	16	GPIO5	PF1	UART0_CTS	~~GPS_MAG~~	
		+3v3	17	18	GPIO6	PF2	UART0_RTS	~~GPS_SIG~~	
SSI1_DT	PE17	SSI0_DT	19	20	GND				
SSI1_DR	PE14	SSI0_DR	21	22	GPIO7	PE8	UART3_CTS	~~BCLK_AD~~	
SSI1_CLK	PE15	SSI0_CLK	23	24	SSI0_CE0	PE16	SSI1_CE0		
		GND	25	26	SSI0_CE1	PE18	SSI1_CE1		

So I wire up an Arduino Micro to the Ci20. Ofcourse using a Logic Level shifter – to avoid putting 5v onto the GPIO (Some of the Ci20 pins are 5v Tolerant, I do not believe the UART pins are).

So now the system looks like this.


Ci20 Dev Board
Ci20 GPIO Connector Bank 1
UART Pins
Arduino
H Bridge Driver
PWM - > Servos
PIR Sensors
Ultrasonics
LEDs



So at this point I need to find out how to use the UART on the Ci20

I flashed the Arduino with a program to say hello world at 9600 baud.

I did all of the easy stuff – eg, sudo cat /dev/ttyS0 but this showed no traces.

At this point I was thinking – Did I wire the level shifter up incorrectly, but I took my little pocket scope and checked, nice clean, square traces.

So – I check the wiki. Not a whole lot of information there for UART setup. A few articles on how to connect a USB to Serial to the Ci20 for Headless setup, but not a lot on how to use Ci20 - > Client device.

So at this point, I start to look at Rasperry Pi Guides for Serial comms as both platforms use Linux / Debian.

http://blog.oscarliang.net/raspberry-pi-and-arduino-connected-serial-gpio/

In order to use the Raspberry Pi's serial port, we need to disable getty (the program that displays login screen) by find this line in file */etc/inittab*

```
T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
```

(I removed the line that used the Ci20s ttySx unit)

To prevents the Raspberry Pi from sending out data to the serial ports when it boots, go to file */boot/cmdline.txt* and find the line and remove it

```
console=ttyAMA0,115200 kgdboc=ttyAMA0,115200
```

The Ci20 does not have this, but it does have a boot config file. I cat'd this with a grep to find what line this was on

cat /boot/config-3.18.3-ci20-1 -n | grep console

```
ci20@ci20:~$ cat /boot/config-3.18.3-ci20-1 -n  | grep console
  3803  #CONFIG_CMDLINE="console=ttyS4,115200 clk_ignore_unused"
ci20@ci20:~$
```

I then navigated to line 3803 – and removed that.

At this point I rebooted.

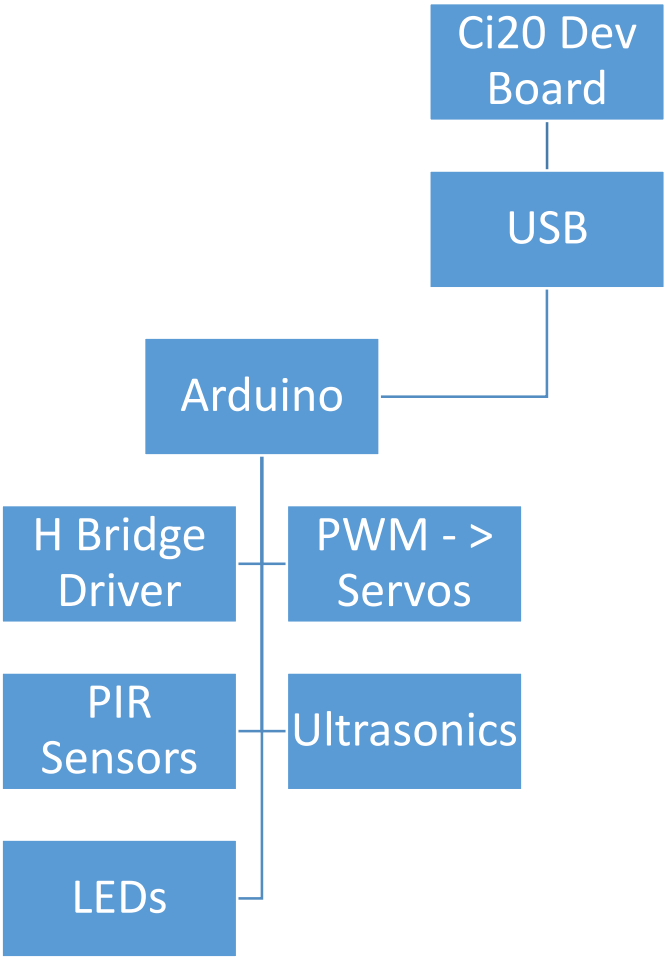
Exeucting the command sudo cat /dev/ttyS0 .. ttys04 on the CI20 still yielded no joy on any of the UARTs even when I moved Arduino off UART0, to 1,2,3 etc. (Trying all the uart ports)

I then start thinking, maybe its just cat, so I tried, Screen, Cu, and Miniterm.

Stil the same problem.

At this point, I was pretty frustrated.

So I just plugged the Arduino into the USB port on the Ci20.



I plug the Arduino in by USB

```
ci20@ci20: ~
ci20@ci20:~$ cat /dev/ttyS
ttyS0  ttyS1  ttyS2  ttyS3  ttyS4
ci20@ci20:~$ cat /dev/ttyS0
cat: /dev/ttyS0: Permission denied
ci20@ci20:~$ sudo cat /dev/ttyS0
[sudo] password for ci20:
^Cci20@ci20:~$ ^C
ci20@ci20:~$ sudo cat /dev/ttyS0
^Cci20@ci20:~$ sudo cat /dev/ttyACM0
Hello World - Ci20 -> Arduino Demo

Hello World - Ci20 -> Arduino Demo

Hello World - Ci20 -> Arduino Demo

Hello World - Ci20 -> Arduino Demo

Hello World - Ci20 -> Arduino Demo

Hello World - Ci20 -> Arduino Demo

Hello World - Ci20 -> Arduino Demo
```

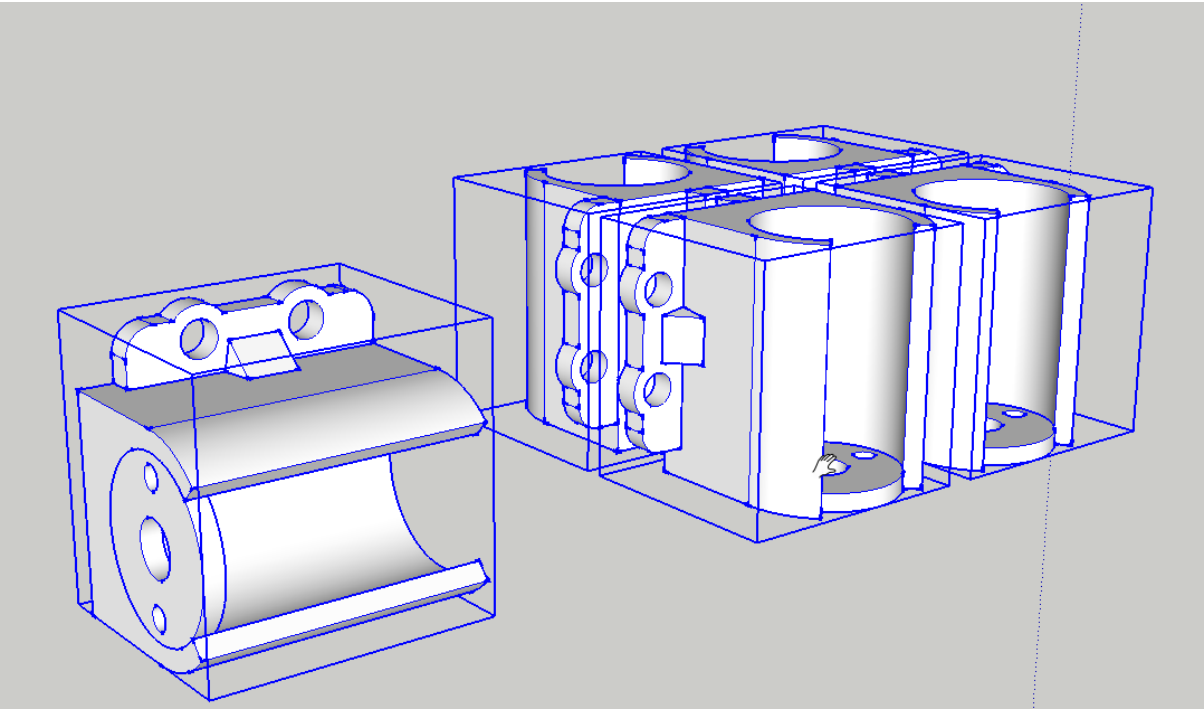
Data from Arduino is on the Ci20

Hardware Stream :

Now is probably a good point to talk about the hardware I used on this project. I am a big fan of Aluminium extrusion and square nuts as a construction method. And because of this – I have a lot of it lying around the workshop. It was very easy to decide, Aluminium Extrusion + 3D Printed parts = Easy robot construction.

Motor Mounts :

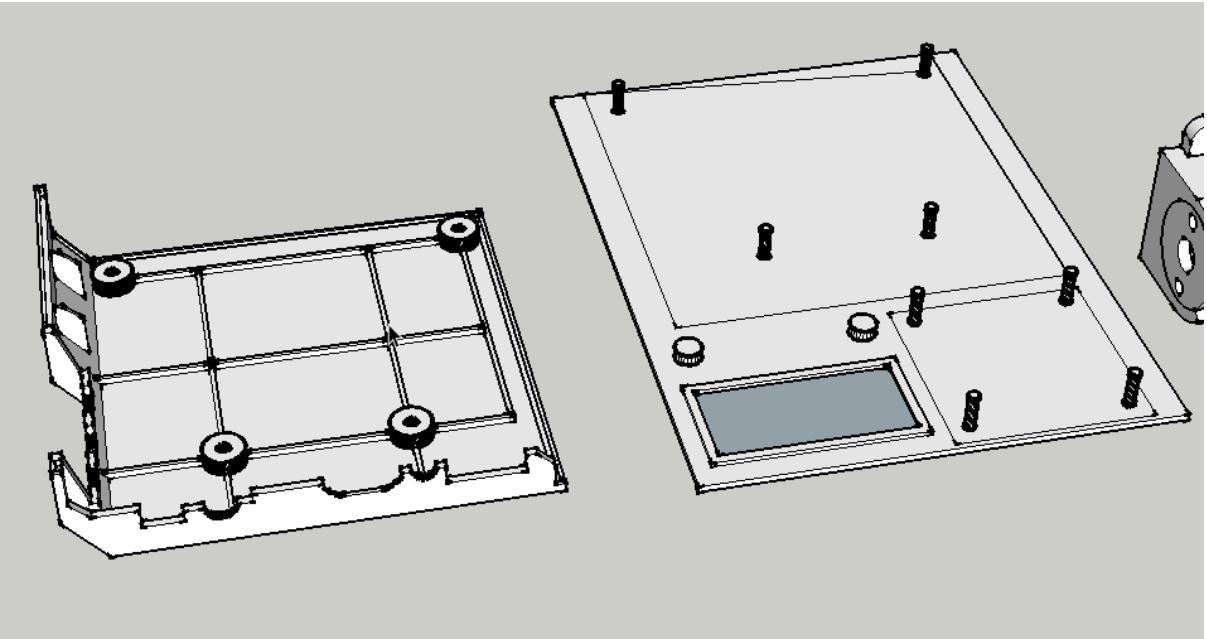
Mounting Motors can be a difficult point for most projects. Especially if the motors have enough torque to free themselves from hot glue etc. That is why I used Sketchup to design a custom mount that was suitable for the robot.



Ci20 Mount.

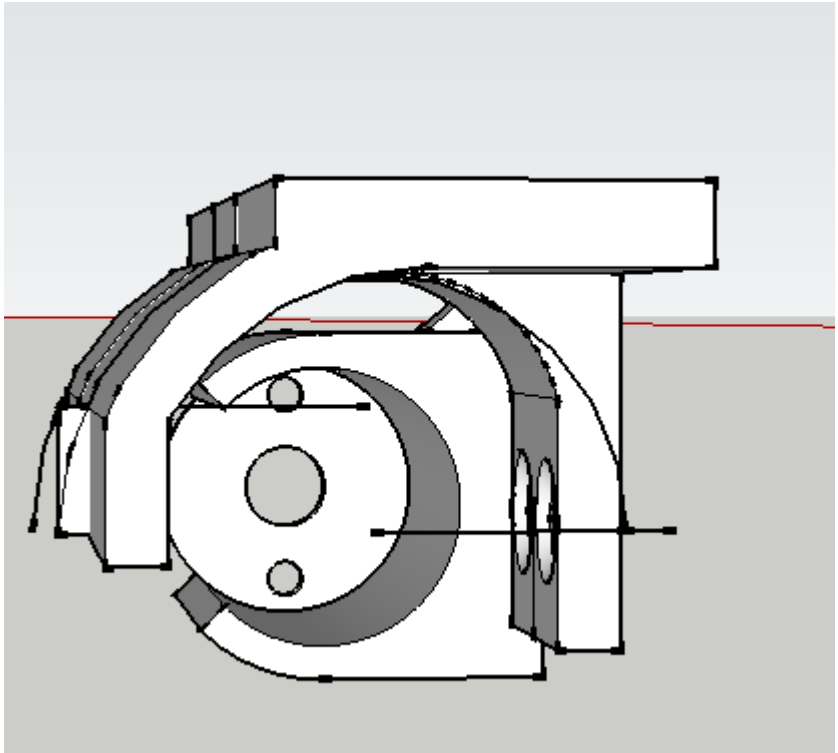
This was a nice giveaway from imagination tech – I imported their 3d Printable case from

http://blog.imgtec.com/powervr/updated-creator-ci20-improved-board-layout-flowcloud-support



And used the location of the holes to make a 3d printed baseplate for my metal robot shell to act as an insulator from the electrical base of the ci20 shorting against anything. Also – Its nice to hold things in place without using hot glue.

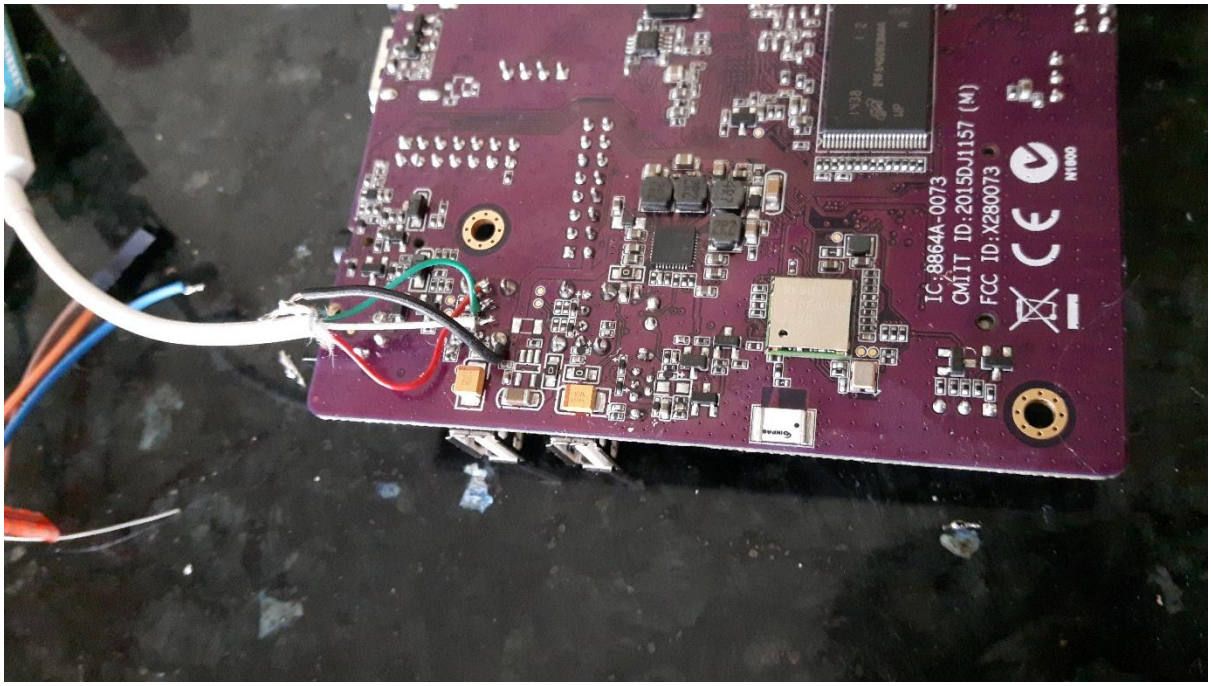
Front Bumper



Also made a ‘ Front Bumper ‘ in sketchup to hold some LEDs

Attaching the Ci20.

I made the case as small as can be required, as a factor of this – I was expecting to use the GPIO expansion port. With this not available, and having to use USB, this made the Ci20 not fit anymore.



I cut open my micro usb cable and hard soldered it to the ci20.

Developing on the Ci20.

With the hard soldered Arduino, I didn't want to have to constantly cycle the connector in fears that it would fatigue or strain the solder joints on the base. So – let's see how much we can do on the Ci20.

I currently have SSH Access to the Ci20, But I am trying to run Arduino software. To do this, I will need a GUI.

Thankfully Debian has something for that.

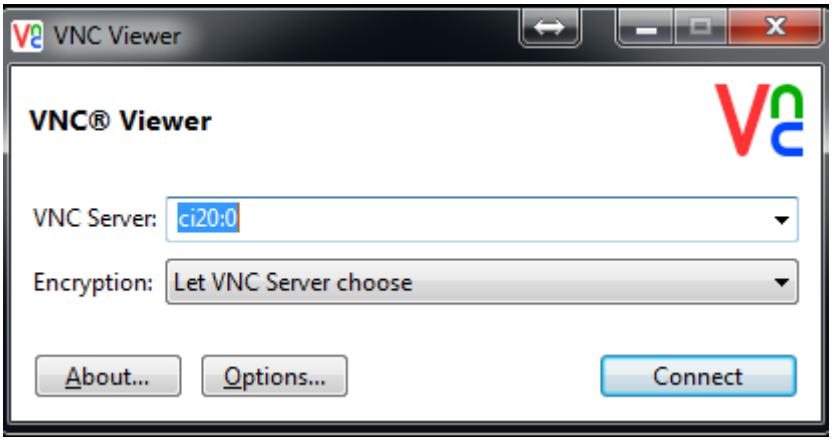
Type in on the SSH command prompt x11vnc.

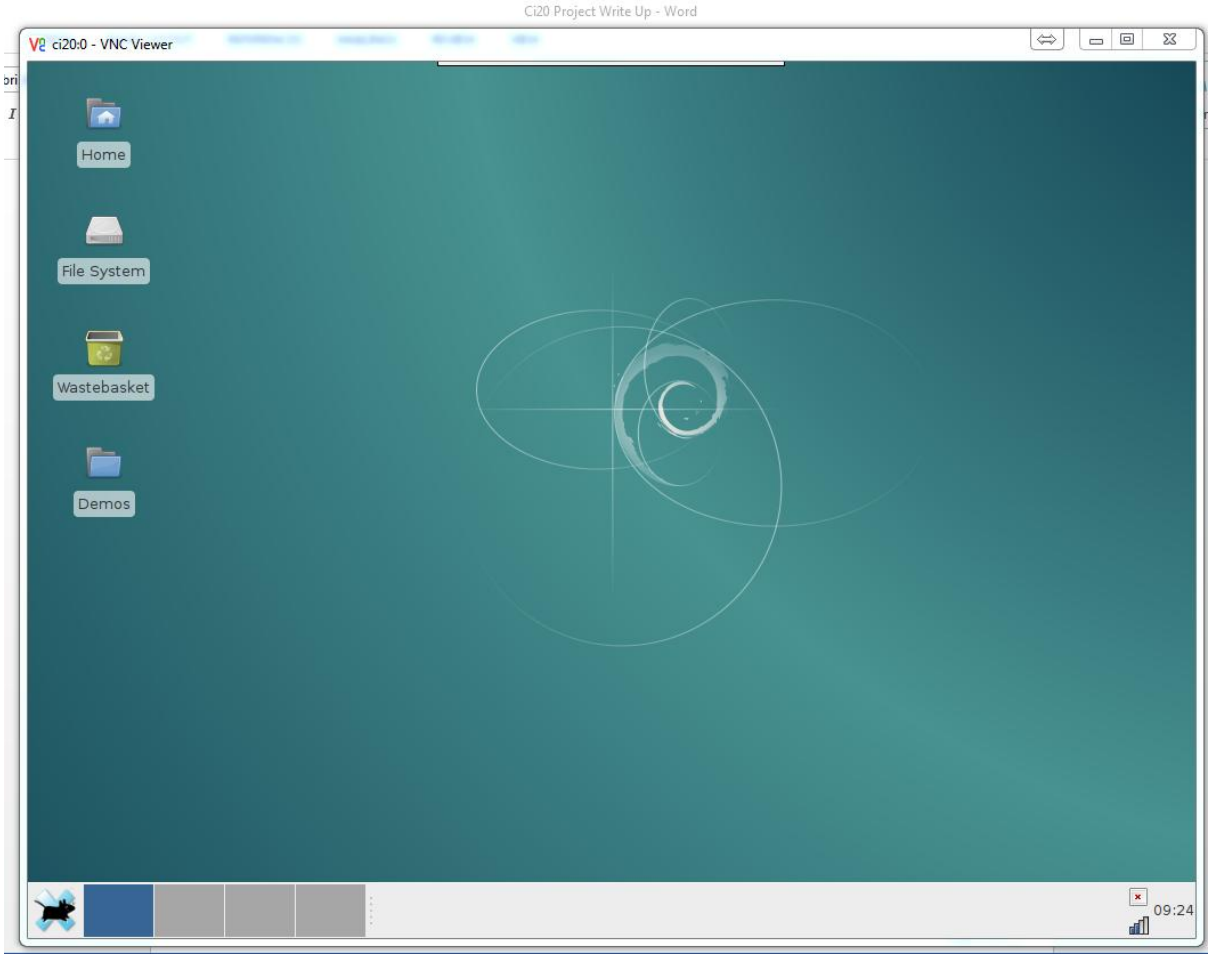
This will host a vnc server on the ci20 to connect to

```
ci20@ci20: ~
x11vnc
##################################################################
#@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@#
#@                                                              @#
#@  **  WARNING  **  WARNING  **  WARNING  **  WARNING  **      @#
#@                                                              @#
#@        YOU ARE RUNNING X11VNC WITHOUT A PASSWORD!!           @#
#@                                                              @#
#@  This means anyone with network access to this computer     @#
#@  may be able to view and control your desktop.               @#
#@                                                              @#
#@ >>> If you did not mean to do this Press CTRL-C now!! <<< @#
#@                                                              @#
#@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@#
#@                                                              @#
#@  You can create an x11vnc password file by running:         @#
#@                                                              @#
#@       x11vnc -storepasswd password /path/to/passfile         @#
#@  or   x11vnc -storepasswd /path/to/passfile                  @#
#@  or   x11vnc -storepasswd                                    @#
#@                                                              @#
#@  (the last one will use ~/.vnc/passwd)                       @#
#@                                                              @#
#@  and then starting x11vnc via:                               @#
#@                                                              @#
#@      x11vnc -rfbauth /path/to/passfile                       @#
#@                                                              @#
#@  an existing ~/.vnc/passwd file from another VNC             @#
#@  application will work fine too.                             @#
#@                                                              @#
#@  You can also use the -passwdfile or -passwd options.        @#
#@  (note -passwd is unsafe if local users are not trusted)     @#
#@                                                              @#
#@  Make sure any -rfbauth and -passwdfile password files       @#
#@  cannot be read by untrusted users.                           @#
#@                                                              @#
#@  Use x11vnc -usepw to automatically use your                 @#
#@  ~/.vnc/passwd or ~/.vnc/passwdfile password files.          @#
#@  (and prompt you to create ~/.vnc/passwd if neither          @#
#@  file exists.)  Under -usepw, x11vnc will exit if it         @#
#@  cannot find a password to use.                              @#
#@                                                              @#
#@                                                              @#
#@  Even with a password, the subsequent VNC traffic is         @#
#@  sent in the clear.  Consider tunnelling via ssh(1):         @#
#@                                                              @#
#@    http://www.karlrunge.com/x11vnc/#tunnelling               @#
```

Lots of warnings about passwords. Im on a local, protected lan. This does not bother me.

I am using vnc viewer


VNC Viewer
VNC® Viewer
VNC Server:
ci20:0
Encryption:
Let VNC Server choose
About...
Options...
Connect


Ci20 Project Write Up - Word
ci20:0 - VNC Viewer
Home
File System
Wastebasket
Demos
09:24

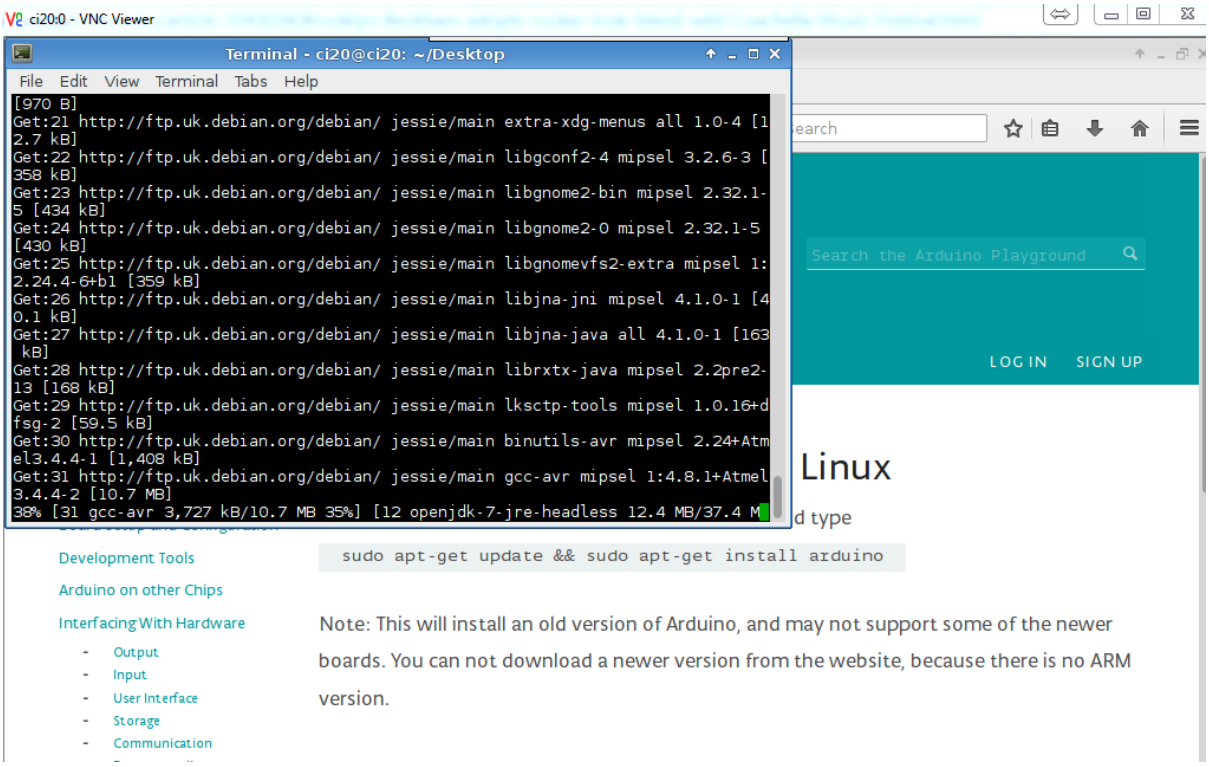
Yay !

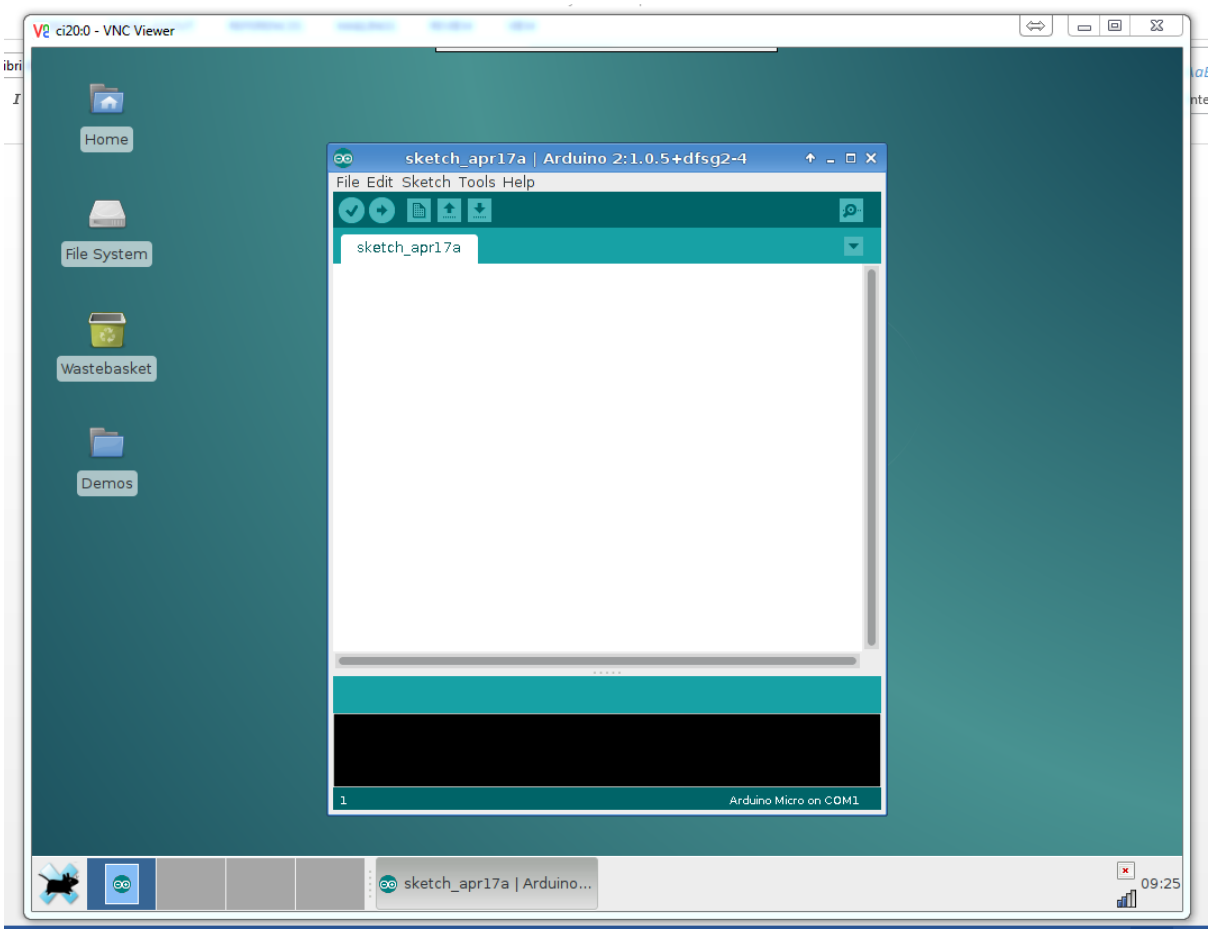
Install Arduino on Raspbian Linux

To install Arduino on Raspbian, just open Terminal and type

```
sudo apt-get update && sudo apt-get install arduino
```

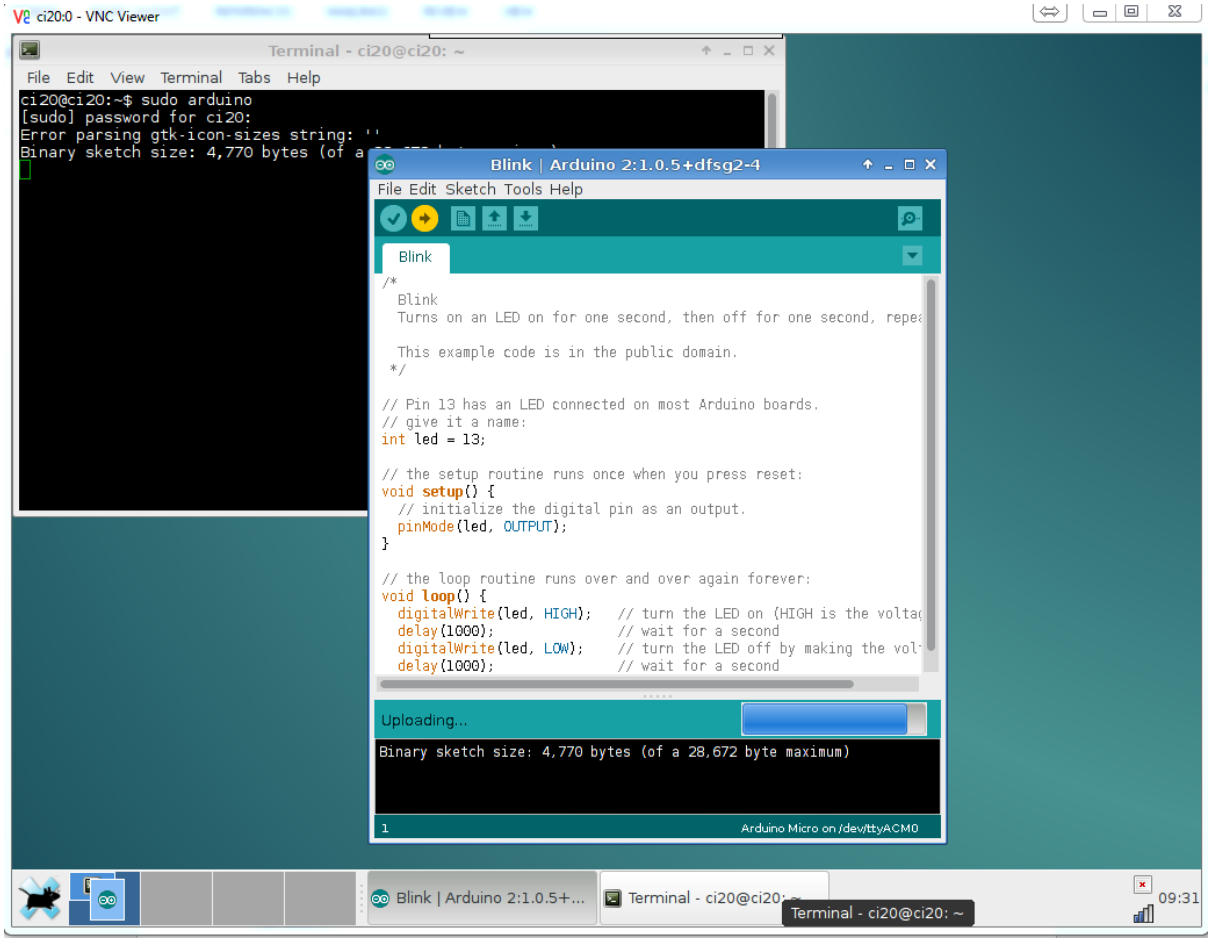
Note: This will install an old version of Arduino, and may not support some of the newer boards. You can not download a newer version from the website, because there is no ARM version.



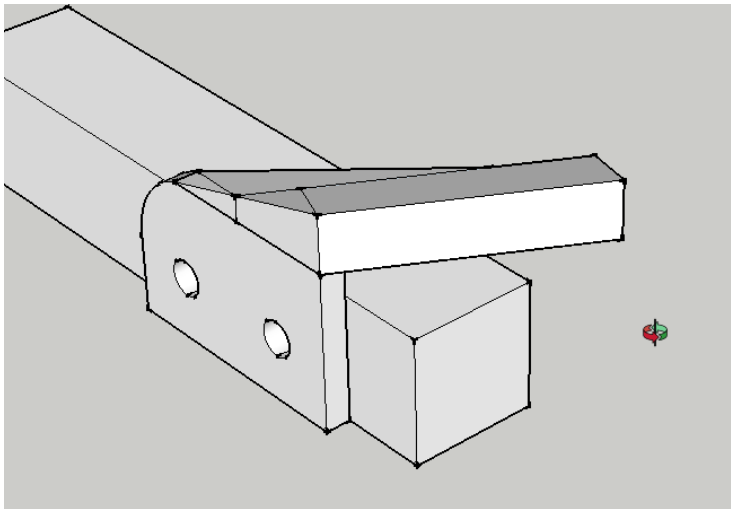


Now this looks good. But when it comes to programming time. No joy. Despite on loading, asking to be added to dialout, it will not show /ttyACM0. Time to launch it with root.

That does It !

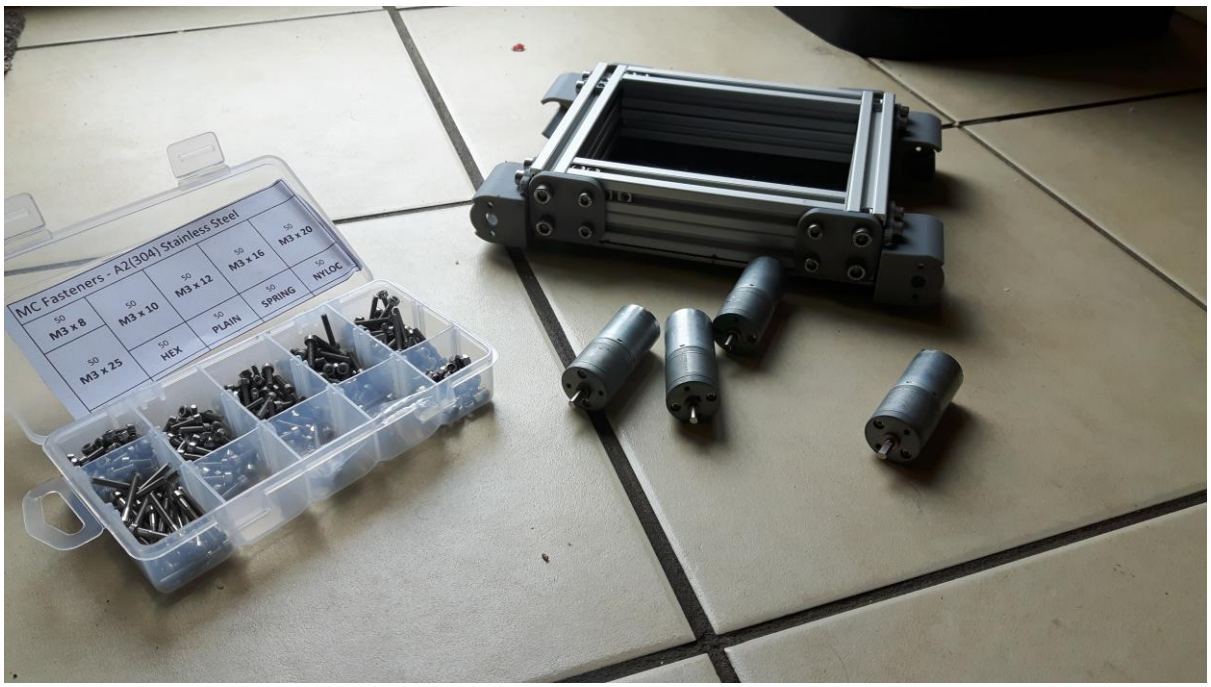


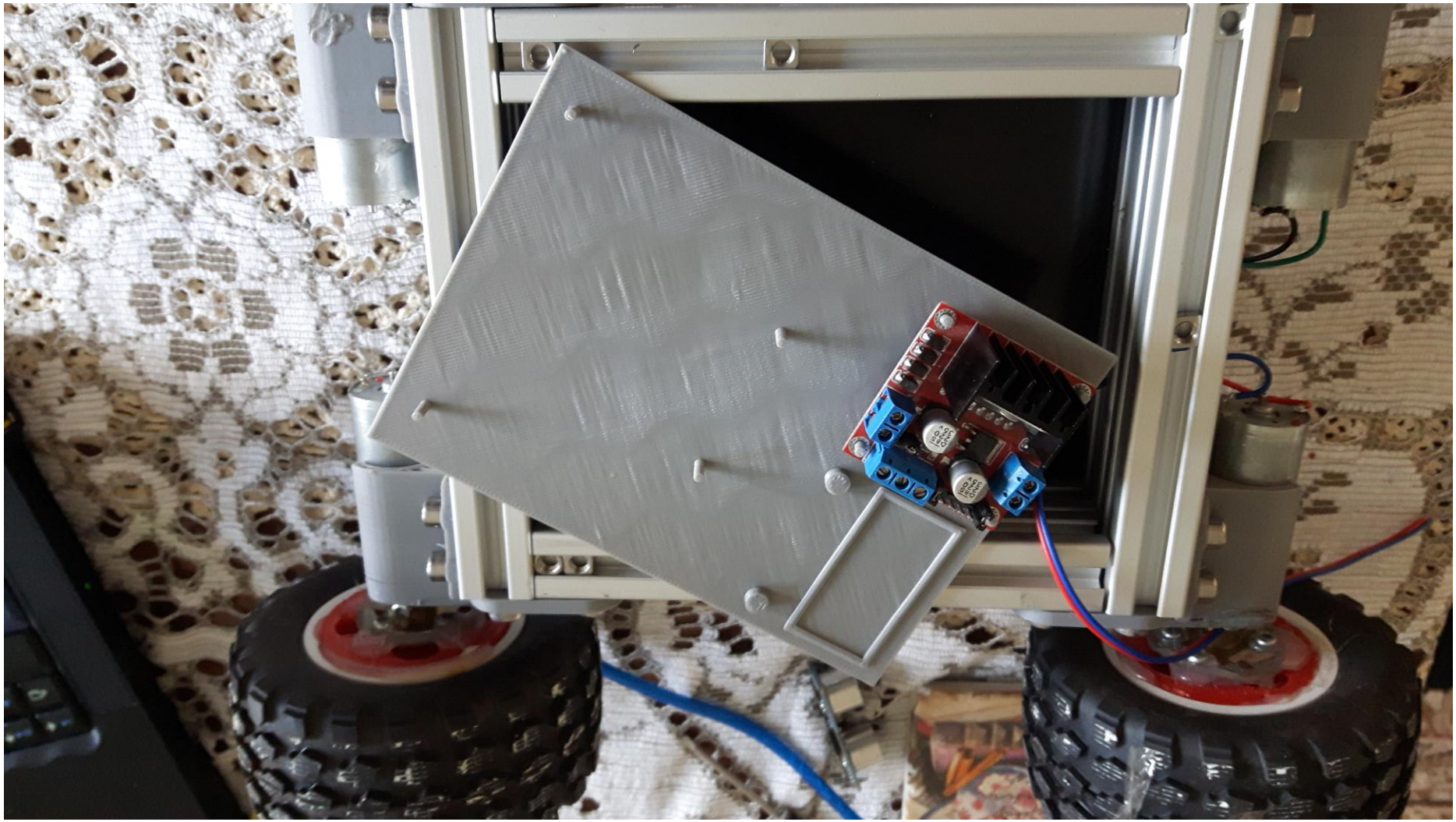
It should be noted, when you want to import libs into the arduino compilation process, because you are running as root, you will need to point the lib directory to somewhere that exists.

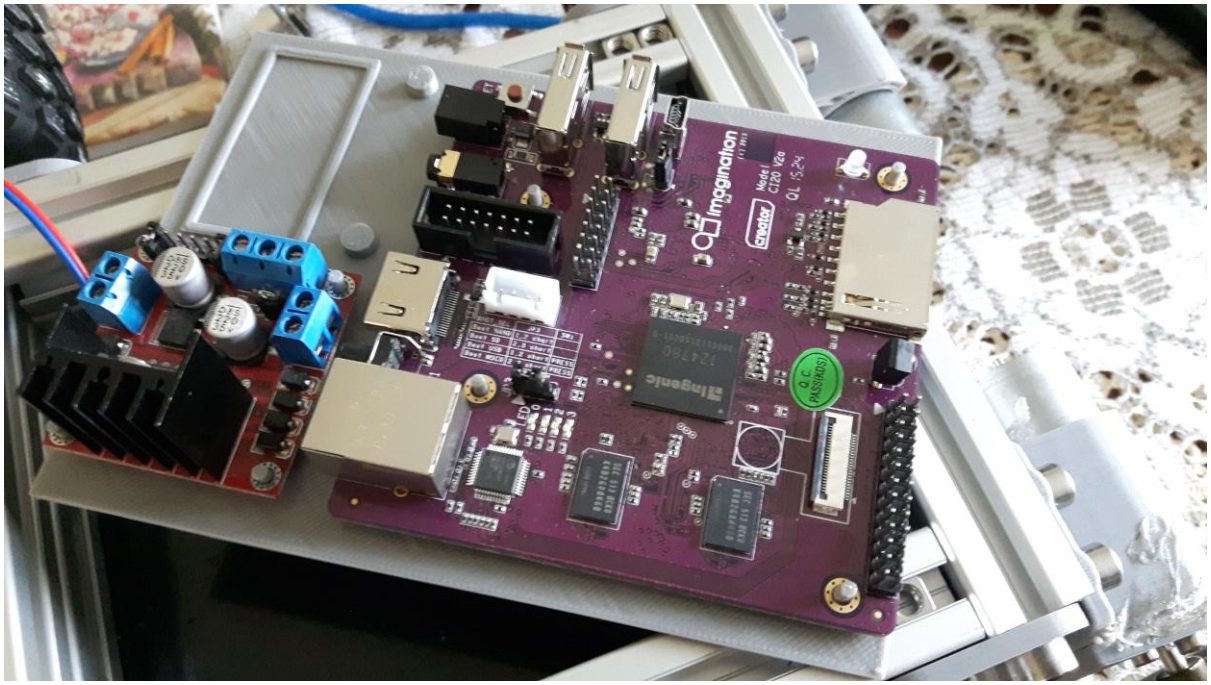


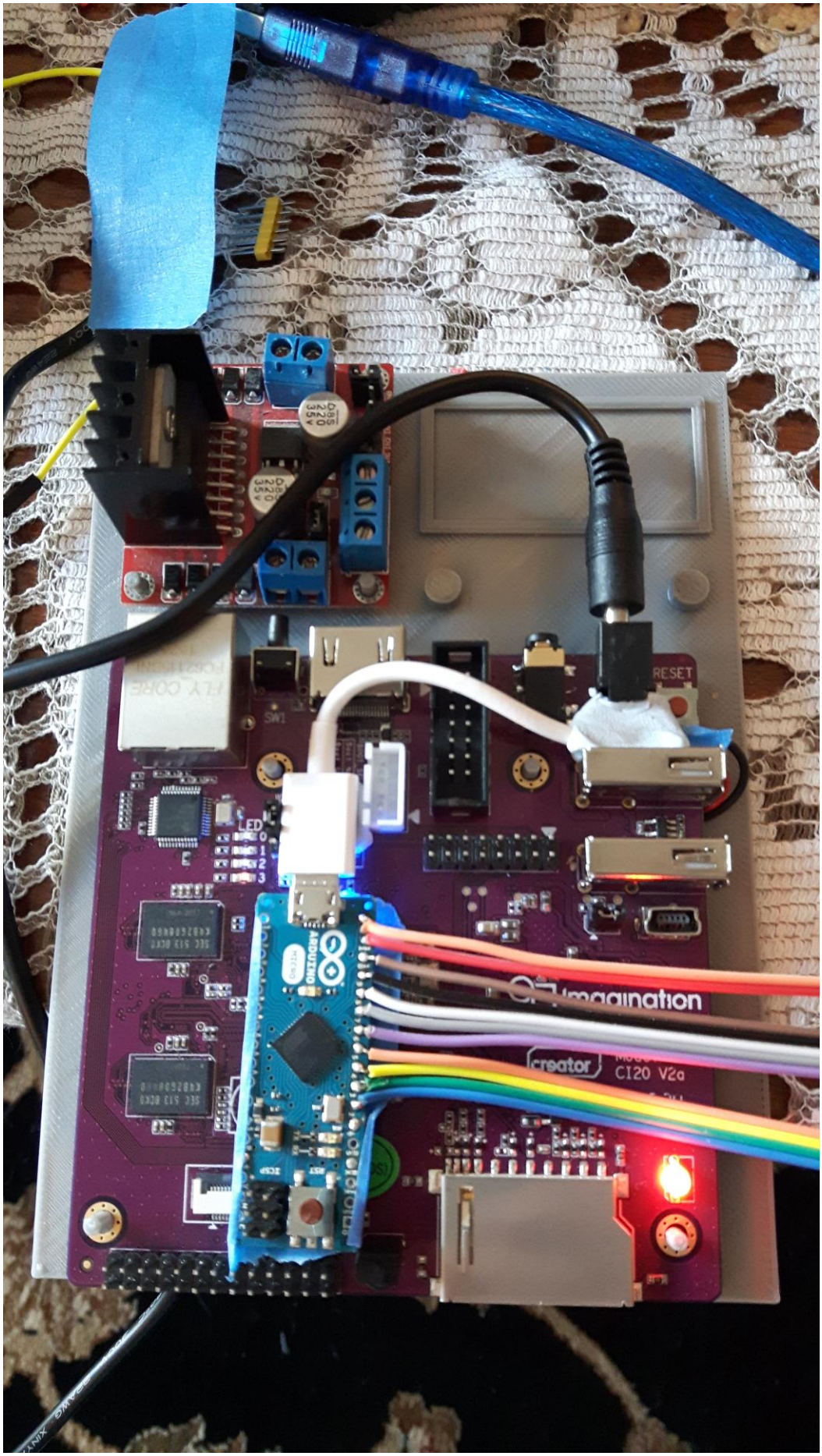
I wanted to put in an angle mount for the PIR sensors. Upon getting them I found the HC-SR501 performance to be a bit lacking. Sensitivity was good, but I found that there was a 'Minimum' time that the logic would be high for of about 5 seconds once triggered. I could have spent some time to re-work the PCB and put in a capacitor with a lower value.

I didn’t bother with this. A detection event will be enough for this Ci20 Demonstrator to look impressive enough.






imagination
Model
C120 V2a
Q1 15.24
Q.C.
PASS(RGB)


RESET
LED
imagination
creator
C120 V2a