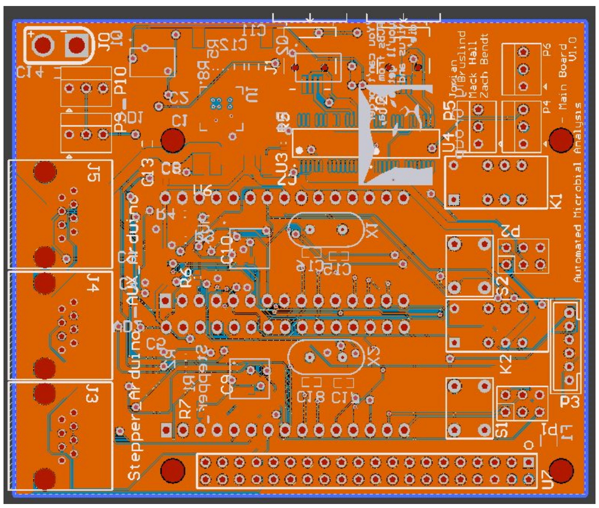
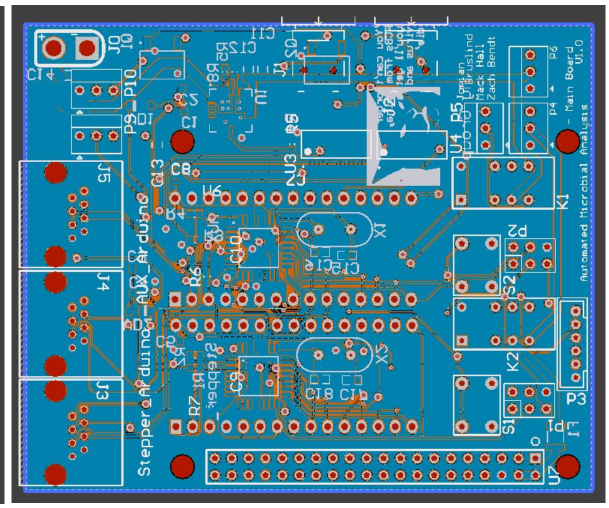
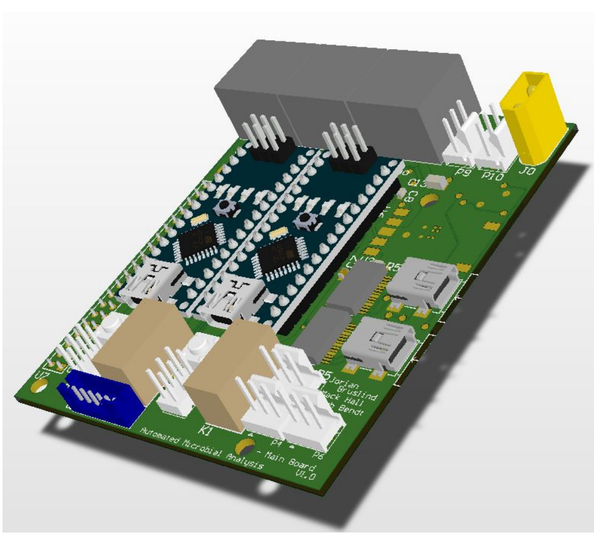
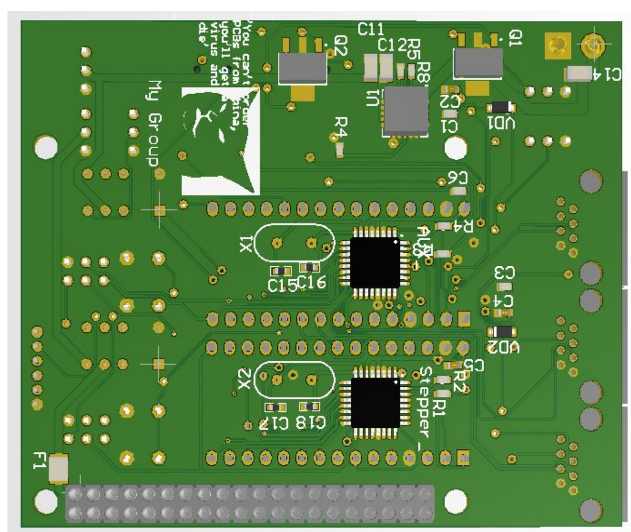
Primary Control PCB

Block Owner: Mack Hall
Date: Mar-17-2020

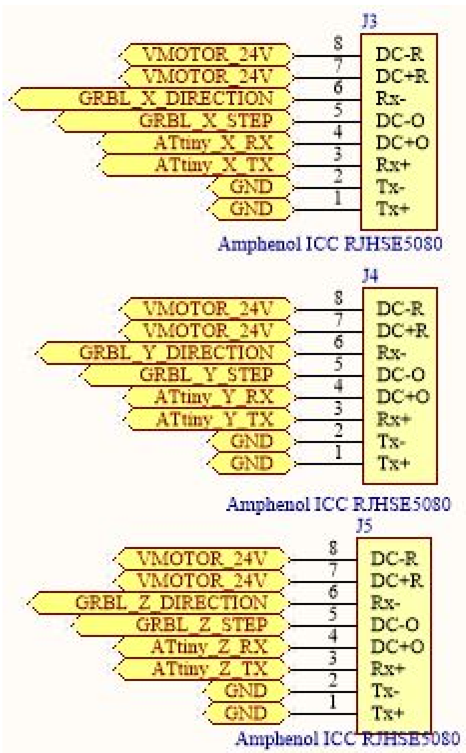
Design Overview

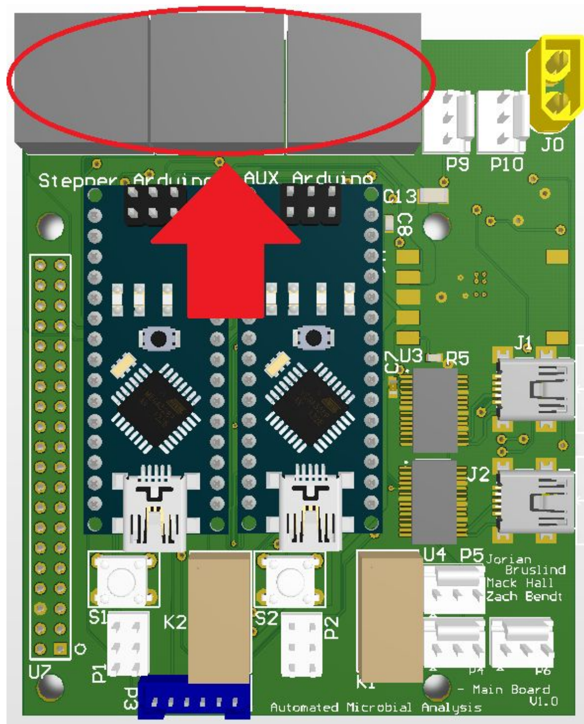





Design Validation Overview

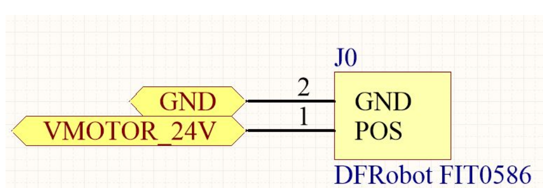
This PCB handles everything related to driving the motors from a software point of view. It interprets the motor coordinates sent from the Raspberry Pi, sends signals and power to the Auxiliary PCBs, and processes encoder data from the motors.

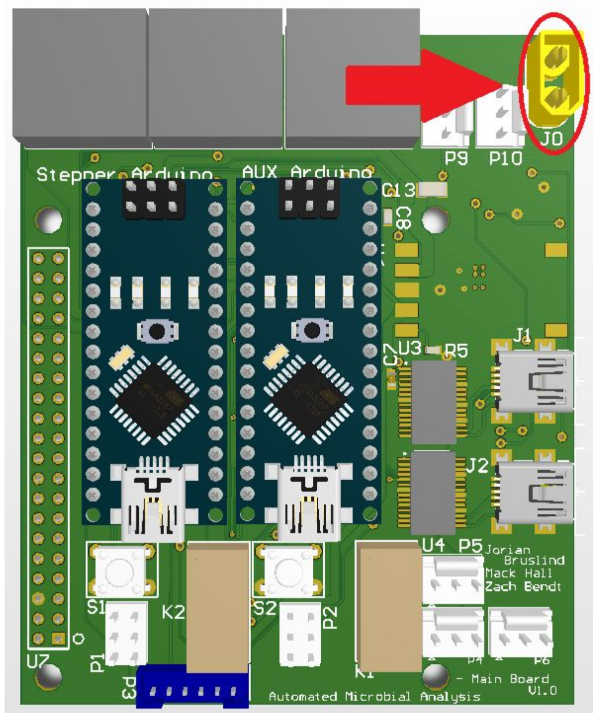
As mentioned previously, the Auxiliary Driver PCB accepts its input from the Central Processing PCBs in the form of an ethernet cable. The ethernet cable carries 24VDC, a direction signal, a step signal, and RX and TX lines for serial communication. The ethernet cable was chosen due to its ubiquity and low cost. Four pins in total are dedicated to power delivery, while the other four pins are dedicated to other signals. Since the Central Processing PCB has to control three separate Auxiliary Diver PCBs, the Central board has three ethernet ports (J3, J4, & J5), one for each Auxiliary PCB.



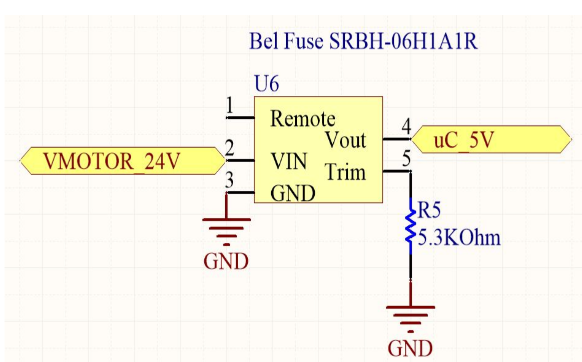


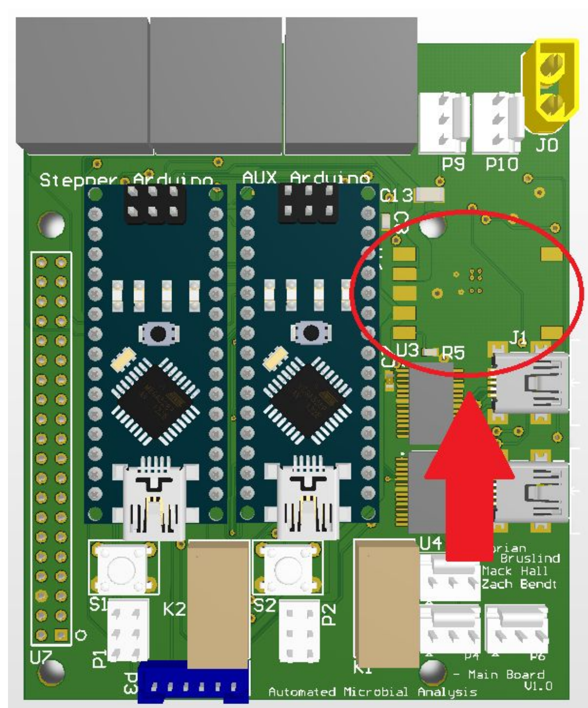
The Central Processing PCB also distributes power to the rest of the system from the external power supply. The board takes in power from J0, a male XT30 connector, which is rated up to 30A as the name implies. The power supply itself that we are using is rated for only 20A, so this is a non-issue.



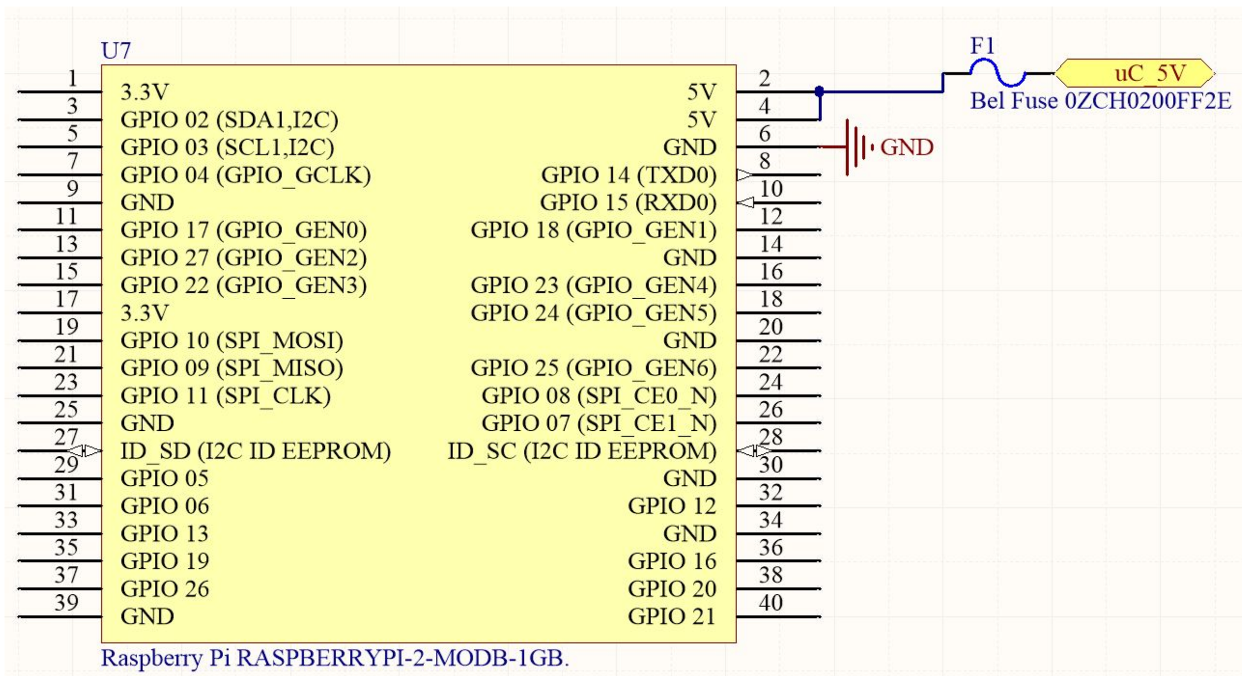


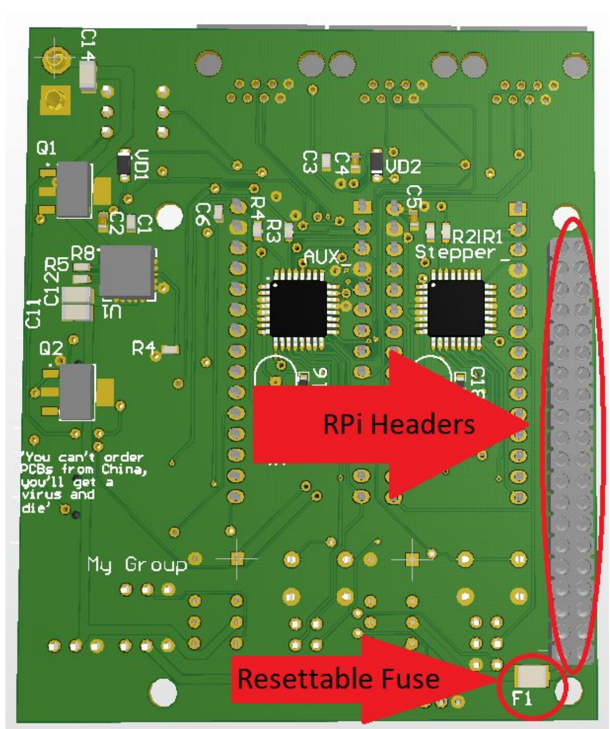
The Central Processing PCB also requires a fairly robust 5VDC converter, since it is responsible for powering the Raspberry Pi as well as two ATMEGA328 microcontrollers. As such, the PCB utilizes the Bel Fuse SRBH-06H1A1R (U3 on the PCB [missing a 3D model]), which is capable of stepping down a 24VDC input into a 5VDC output up to 6A. This is overkill for our system, as each of the listed components that draw from the SRBH-06H1A1R only take about 1A each. As such, we have a safety factor of 100%.



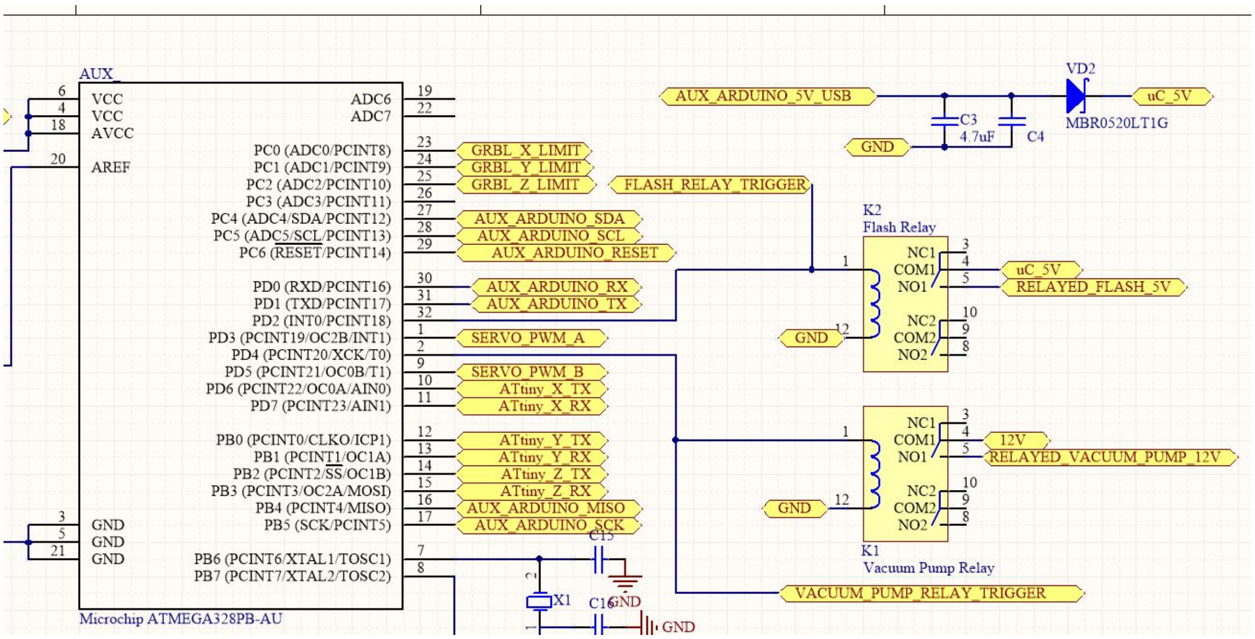


As mentioned above, the Central PCB provides 5VDC to the Raspberry Pi via its GPIO pins. This is not recommended by the makers of the Raspberry Pi, as the GPIO pins bypass the internal fusing and regulation done in order to ensure a safe and regulated 5V line for the system. Before going to the upside-down headers that allow the Central PCB to be worn as a Raspberry Pi hat, the 5V line on the Central PCB must go through a resettable thermal fuse (F1 on the PCB) in order to ensure that the Raspberry Pi cannot draw enough current to harm itself. The fuse we are using is the Bel Fuse 0ZCH0200FF2E, which is rated for 2A before tripping. Once tripped, the fuse will reset after being returned to room temperature.

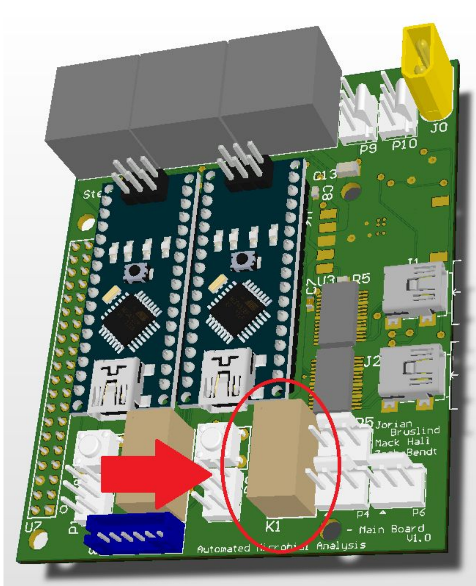


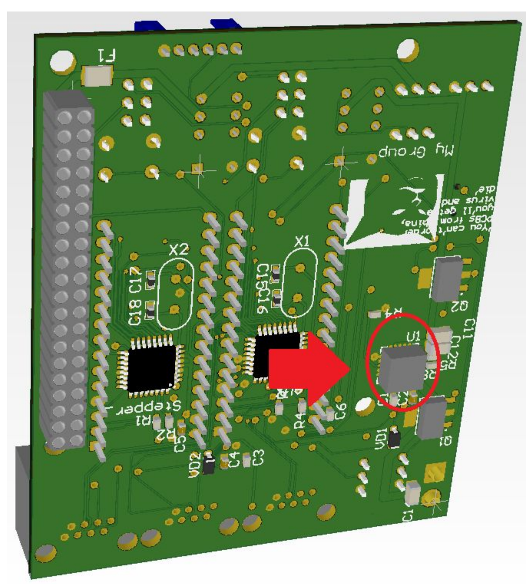


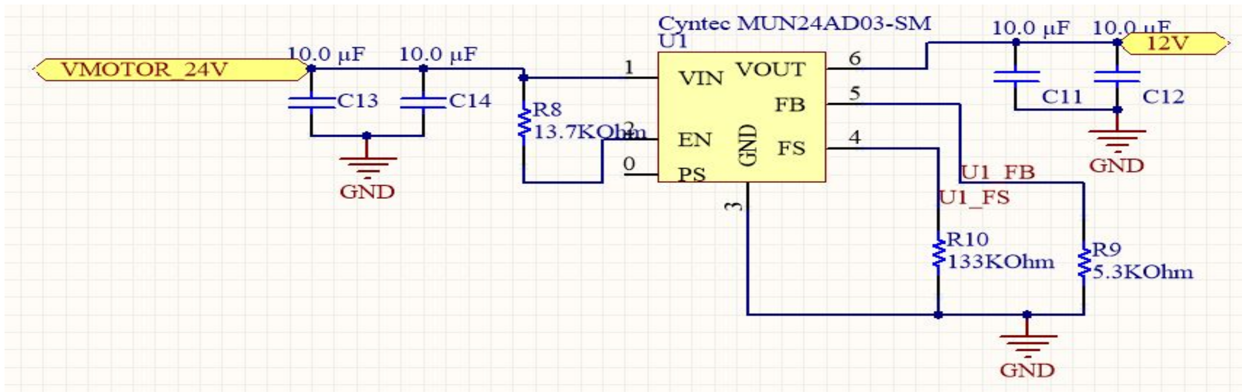
Furthermore, the Central PCB is also responsible for providing 12VDC to drive the vacuum pump. The vacuum pump simply turns on when a voltage is applied to the terminals. Therefore, the system will utilize a relay to turn the pump on and off. The schematic is as shown:



On the Auxiliary ATMEGA328 found on the Central PCB, the enable pin of the K1 relay is tied to PD4 (physical pin 2), which is capable of sending a 5VDC signal to open or close the relay. When the K1 relay is open, the RELAYED_VACUUM_PUMP_12V net, which is connected to the positive terminal of the vacuum pump, is left floating, meaning the vacuum pump is getting no voltage and is therefore off. When the K1 relay is closed, the RELAYED_VACUUM_PUMP_12V net receives 12VDC and turns the pump on.

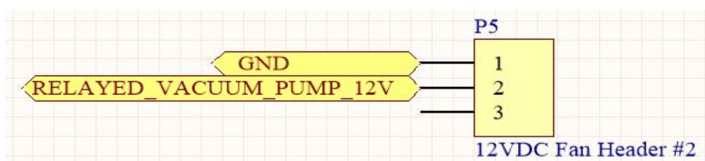


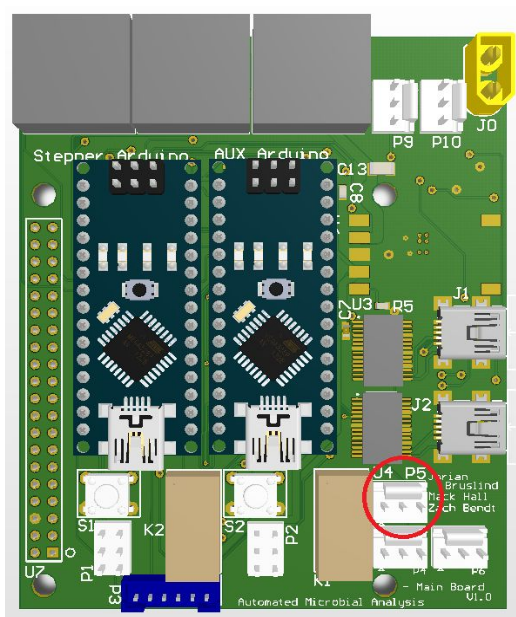




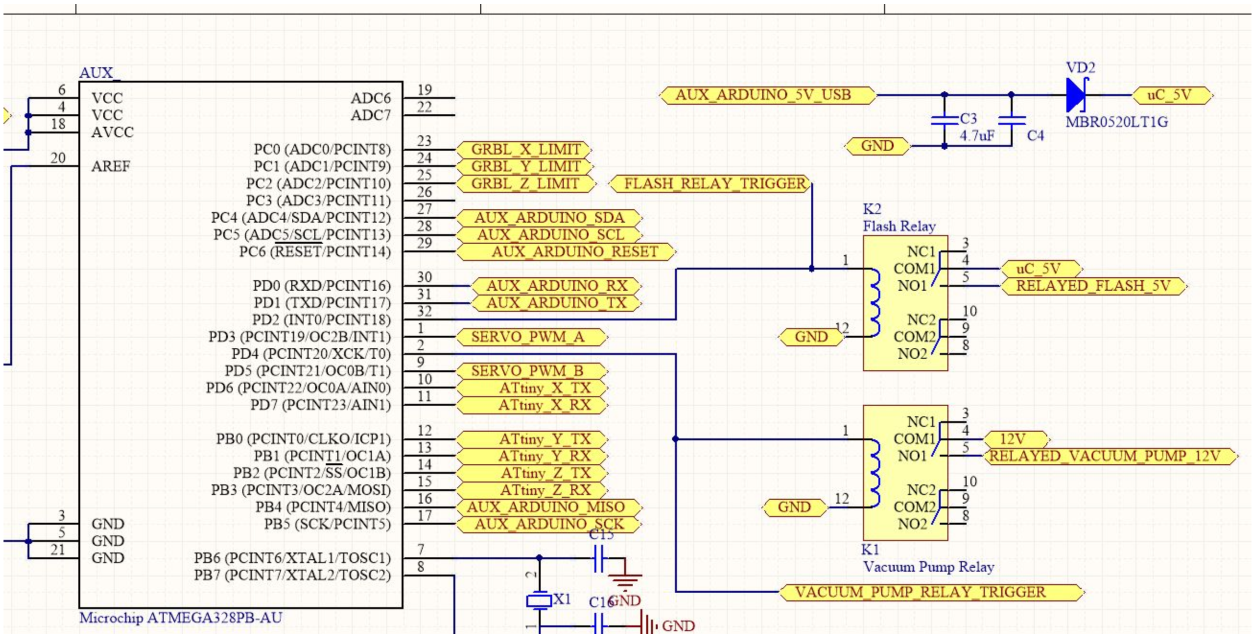
The 12VDC line is supplied by U1, a Cyntec MUN24AD03-SM, which is a simple buck converter that steps down a 24VDC signal to a 12VDC signal. The 24VDC signal is the “common” DC-bus throughout the system, which means it is supplied from an external load-regulating power supply that rectifies wall voltage. The Cyntec MUN24AD03-SM is capable of handling up to 3A, and is therefore more than plenty for our vacuum pump, which only draws 1A at maximum.

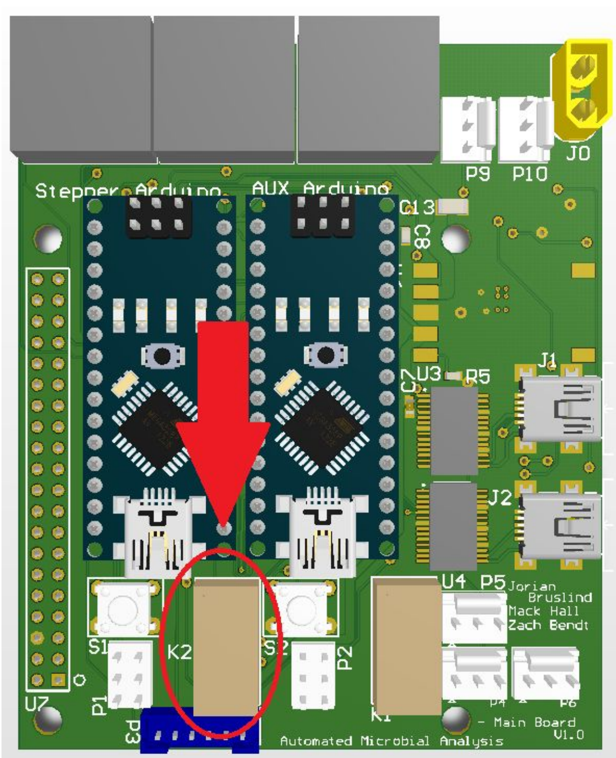
The D2028 vacuum pump is connected to the Primary Control PCB via the P5 header.



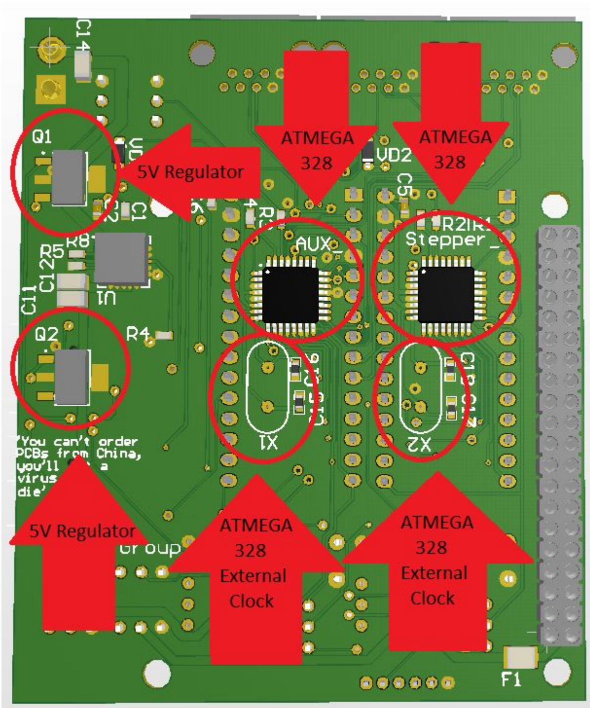


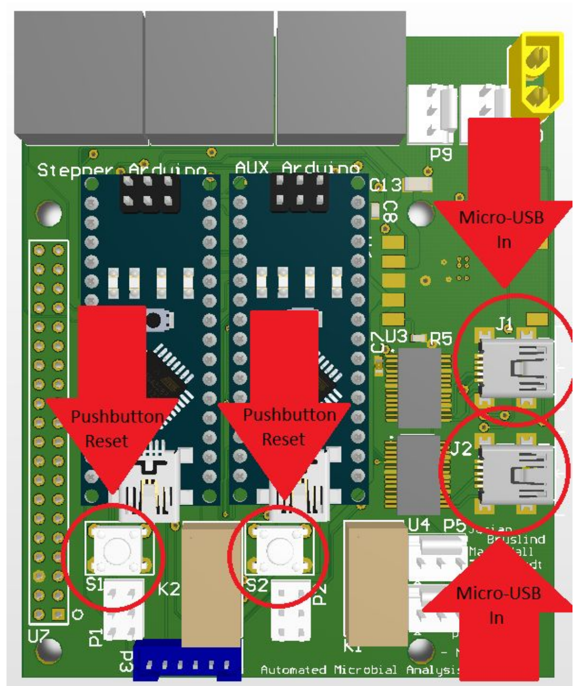
The other relay on the board, K2, is responsible for enabling the LEDs below the camera, in order to act as a flash. These LEDs take a 5V input, and are therefore tied to the 5V regulated line on the board. This setup is very similar to the vacuum pump, however it is only 5VDC instead of 12VDC.

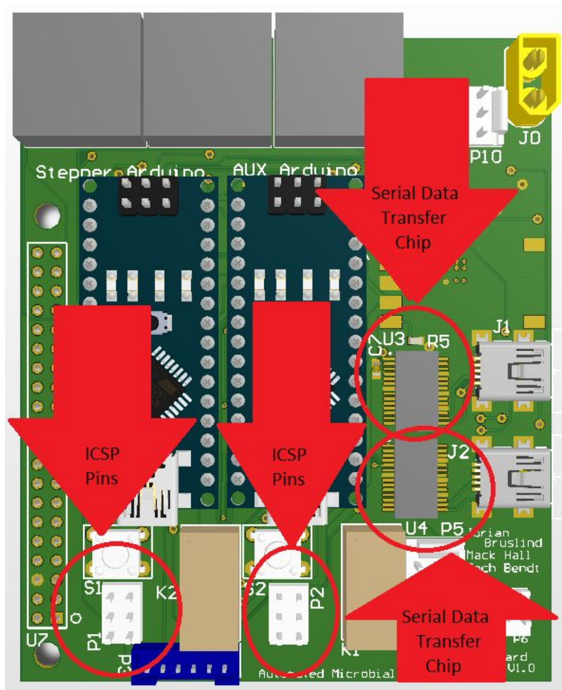




The last thing this Central PCB needs to do is to connect all of the prior connections to two ATMEGA328 microsontrollers, which will actually run the software we need to control the system. The PCB includes sockets to plop in two Arduino Nanos for testing and debugging purposes, but this will not be used in the final system. The back of the PCB hides the solder pads for two SMD ATMEGA 328s (AUX_ and Stepper_ on the PCB), as well as two external 16MHz clocks (X1 and X2), and two 5V regulators (Q1 and Q2).





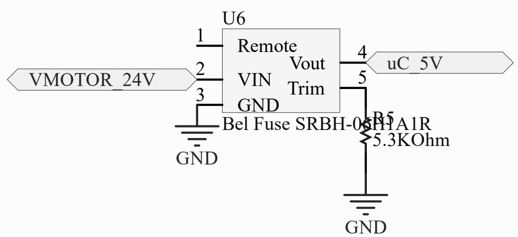


Essentially, the PCB has the equivalent hardware of two integrated Arduino Nanos on board. They have the equivalent connections lifted straight from the official Nano schematic itself found here.

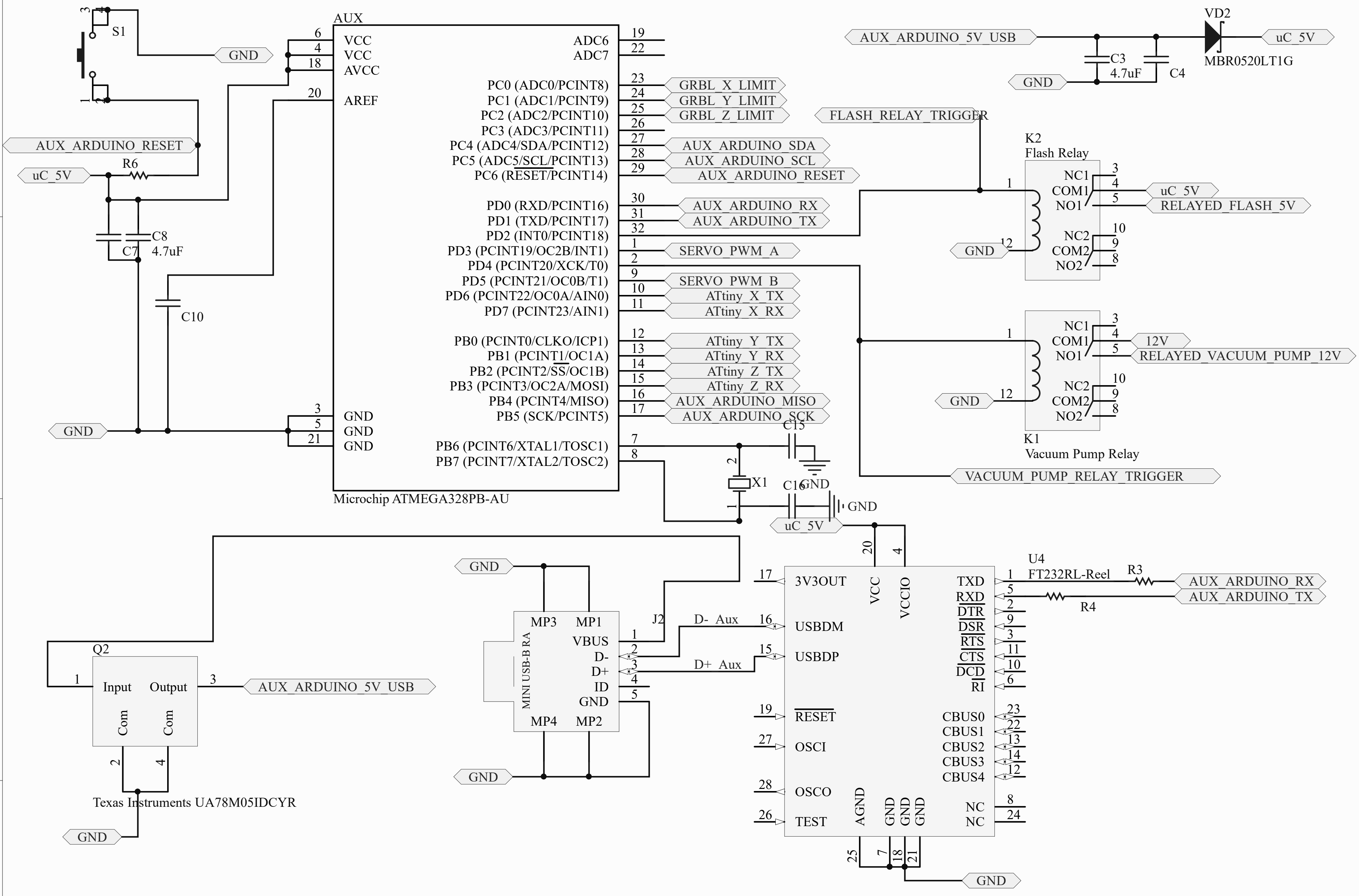
The Stepper_ ATMEGA 328 is responsible for running a program called ‘grbl’. grbl is a well-established software that is used for driving stepper motors. The user inputs G-Code, and grbl will generate the equivalent signals to be sent to the stepper motor driver that will move the motor to the coordinates defined in said G-Code. Essentially, the stepper driver is a median in which grbl can move the stepper motor directly. Along with other signals, the necessary signals generated from grbl are sent through the ethernet cable, where they are eventually interpreted by the TCM2209 on the Auxiliary Diver PCB. In the final system, the Raspberry Pi will be running a script which will send G-Code serially to the Stepper_ ATMEGA, which will control the position of the stepper motors indirectly via the stepper motor divers.

The AUX_ ATMEGA 328 is responsible for reading back the messages sent by the Auxiliary Stepper PCBs, as well as turning on the relays for the vacuum pump and the camera flash. An additional ATMEGA 328 was used in conjunction with the Stepper_ ATMEGA 328 due to computational bottlenecks. grbl is a very demanding program, and attempting to run additional code could impact the timing of the wave generation, meaning that the end stepper motor motion would not be as accurate or as smooth.

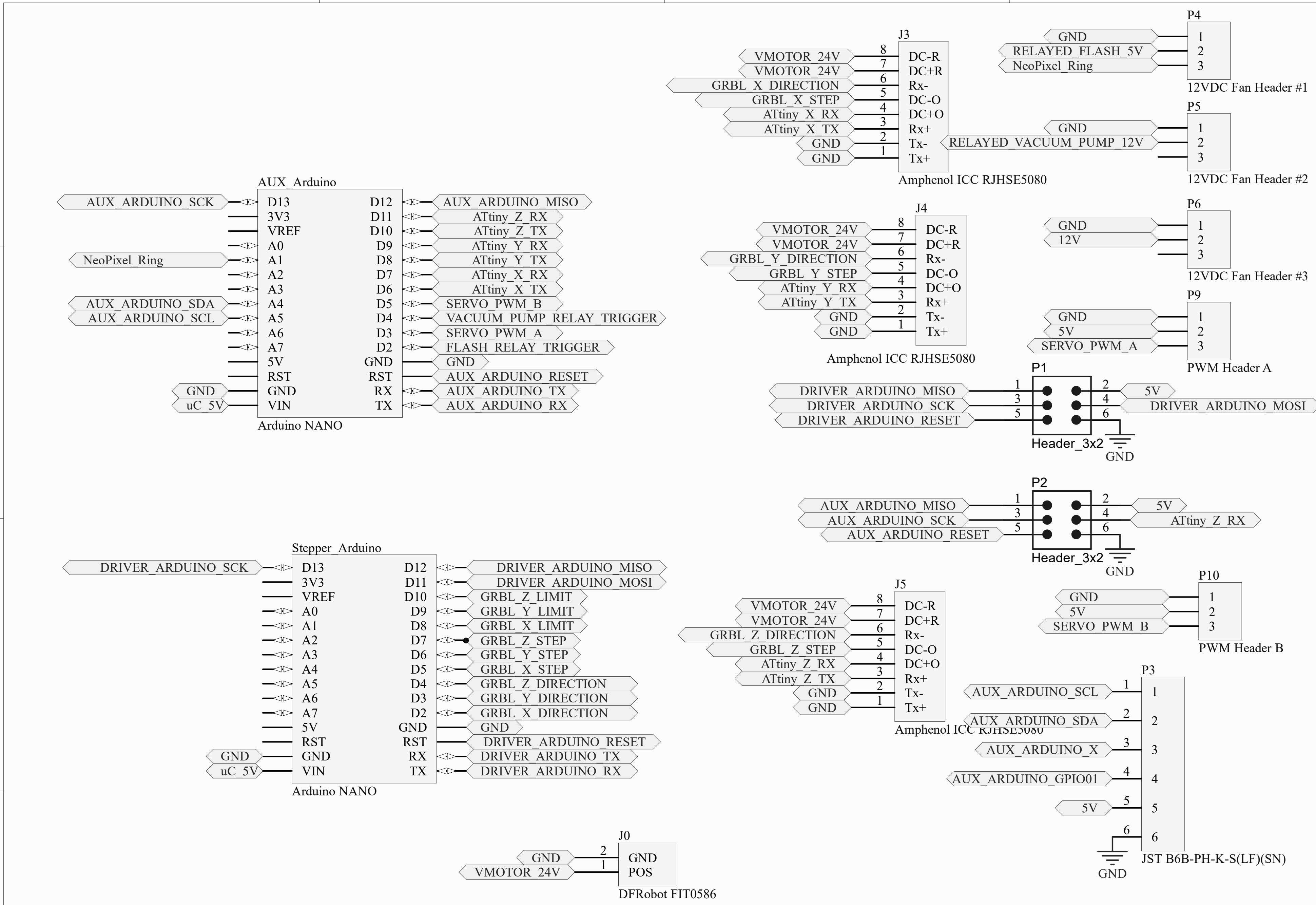
A detailed list, and link to all of the datasheets for the components on the Central Processing PCB are found here: Central PCB Materials



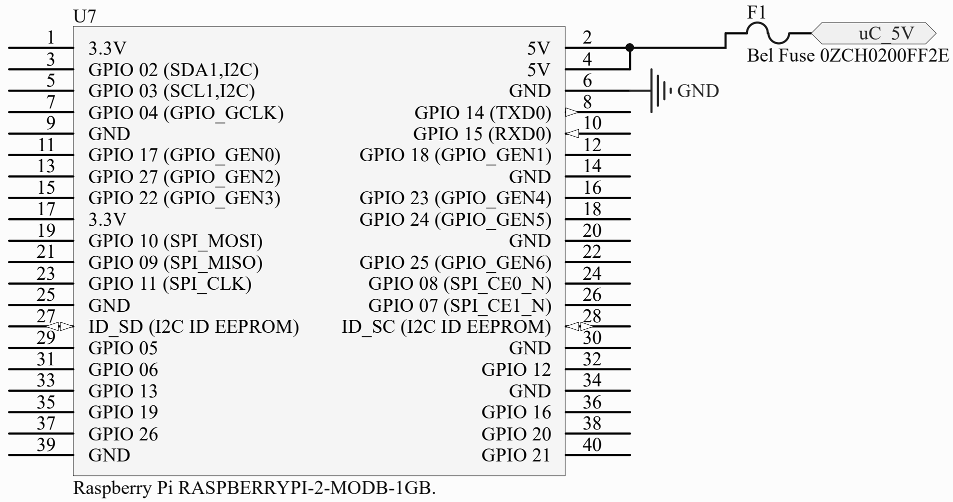
Title			
Size A	Number		Revision
Date:	2/9/2020	Sheet of	
File:	5V_Regulator.SchDoc	Drawn By:	



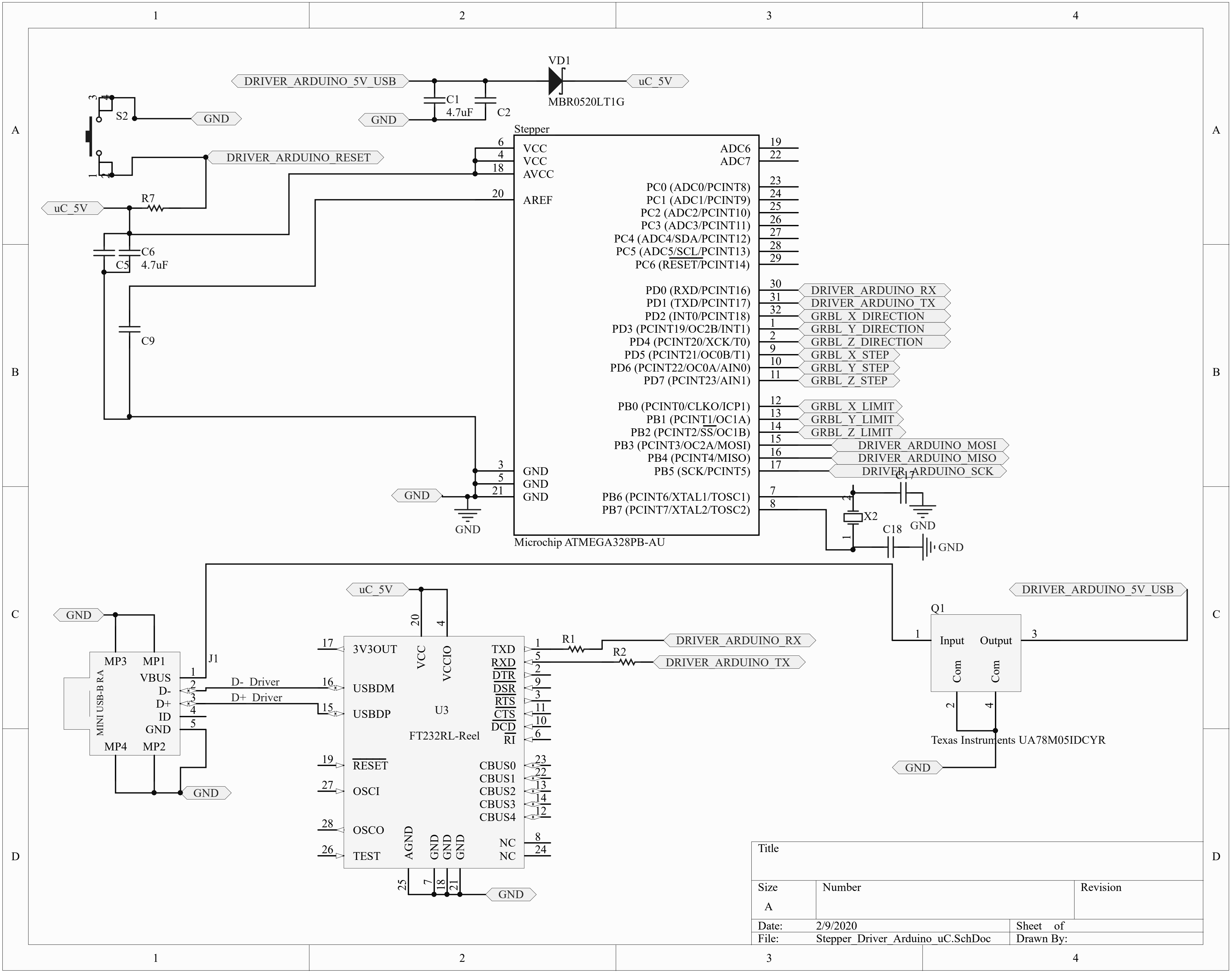
Title		
Size: A	Number	Revision
Date: 2/9/2020		Sheet of
File: Auxillary_Arduino_uC.SchDoc		Drawn By:



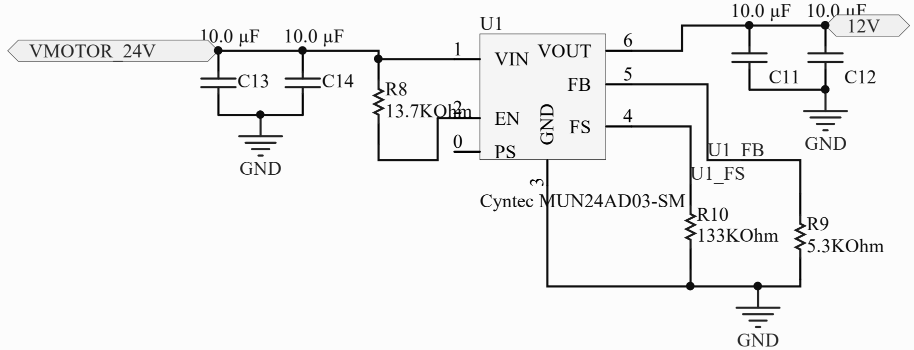
Title		
Size: A	Number	Revision
Date: 2/9/2020		Sheet of
File: Connectors.SchDoc		Drawn By:



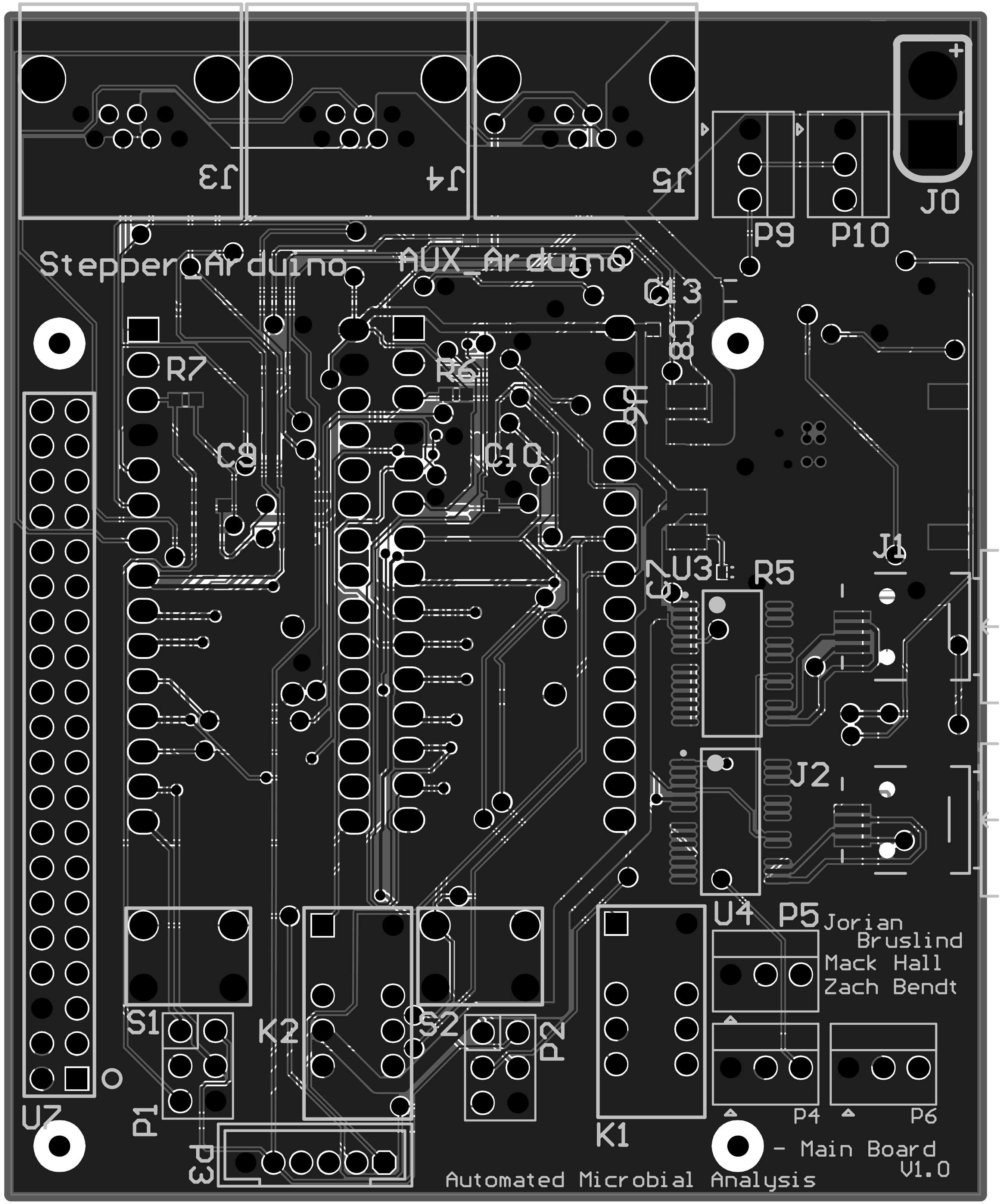
Title		
Size A	Number	Revision
Date: 2/9/2020		Sheet of
File: RPI_Hat.SchDoc		Drawn By:



1
2
3
4
A
B
C
D
DRIVER_ARDUINO_5V_USB
VD1
MBR0520LT1G
uC_5V
C1
4.7uF
C2
GND
S2
DRIVER_ARDUINO_RESET
R7
C5
C6
4.7uF
C9
Stepper
VCC
VCC
AVCC
AREF
GND
GND
GND
ADC6
ADC7
PC0 (ADC0/PCINT8)
PC1 (ADC1/PCINT9)
PC2 (ADC2/PCINT10)
PC3 (ADC3/PCINT11)
PC4 (ADC4/SDA/PCINT12)
PC5 (ADC5/SCL/PCINT13)
PC6 (RESET/PCINT14)
PD0 (RXD/PCINT16)
PD1 (TXD/PCINT17)
PD2 (INT0/PCINT18)
PD3 (PCINT19/OC2B/INT1)
PD4 (PCINT20/XCK/T0)
PD5 (PCINT21/OC0B/T1)
PD6 (PCINT22/OC0A/AIN0)
PD7 (PCINT23/AIN1)
PB0 (PCINT0/CLKO/ICP1)
PB1 (PCINT1/OC1A)
PB2 (PCINT2/SS/OC1B)
PB3 (PCINT3/OC2A/MOSI)
PB4 (PCINT4/MISO)
PB5 (SCK/PCINT5)
PB6 (PCINT6/XTAL1/TOSC1)
PB7 (PCINT7/XTAL2/TOSC2)
Microchip ATMEGA328PB-AU
DRIVER_ARDUINO_RX
DRIVER_ARDUINO_TX
GRBL_X_DIRECTION
GRBL_Y_DIRECTION
GRBL_Z_DIRECTION
GRBL_X_STEP
GRBL_Y_STEP
GRBL_Z_STEP
GRBL_X_LIMIT
GRBL_Y_LIMIT
GRBL_Z_LIMIT
DRIVER_ARDUINO_MOSI
DRIVER_ARDUINO_MISO
DRIVER_ARDUINO_SCK
C17
X2
C18
J1
MP3
MP1
VBUS
D-
D+
ID
GND
MP4
MP2
MINI USB-B RA
D- Driver
D+ Driver
U3
FT232RL-Reel
3V3OUT
USBDM
USBDP
RESET
OSCI
OSCO
TEST
VCC
VCCIO
AGND
GND
GND
GND
TXD
RXD
DTR
DSR
RTS
CTS
DCD
RI
CBUS0
CBUS1
CBUS2
CBUS3
CBUS4
NC
NC
R1
R2
Q1
Input
Output
Com
Com
Texas Instruments UA78M05IDCYR
Title
Size
A
Number
Revision
Date: 2/9/2020
Sheet of
File: Stepper_Driver_Arduino_uC.SchDoc
Drawn By:



Title			
Size A	Number		Revision
Date:	2/9/2020	Sheet of	
File:	Voltage_Regulator.SchDoc	Drawn By:	


Stepper_Arduino
AUX_Arduino
P9
P10
J0
C13
C8
R7
R6
C9
C10
U6
U3
R5
J1
J2
U4
P5
Jorian
Bruslind
Mack Hall
Zach Bendt
S1
K2
S2
P2
K1
P1
P3
U7
P4
P6
- Main Board
V1.0
Automated Microbial Analysis

Design Validation Interface Table

120vc-24vdc_cnvrtr_prmry_cntrl_pcb_dcpwr

Inominal: 7-9A	For the Raspberry Pi: ● From the official datasheet of the Raspberry Pi, the board draws ~1A of nominal current. https://bit.ly/2Sr4IJA For the TMC2209: ● According to the datasheet for the TMC2209, the stepper driver we are using, there is a current limiting function. We will utilize this function to ensure that our stepper motors never exceed 2A for reduced heat output and increased component longevity. 2A across 3 motors = 6A nominal. https://bit.ly/37sqxN0 For the Arduino Nanos: ● From the Arduino official website, the Arduino Nanos draw ~20 mA nominally. https://bit.ly/2SrcumT
Ipeak: 12A	For the Raspberry Pi: ● From the official website of the Raspberry Pi, the board should have at least a 2.5A power supply. https://bit.ly/2vxPcCy For the TMC2209: ● According to the datasheet for the TMC2209, the stepper driver we are using, there is a current limiting function. We will utilize this function to ensure that our stepper motors never exceed 2A for reduced heat output and increased component longevity. 2A across 3 motors = 6A nominal. https://bit.ly/37sqxN0 For the Arduino Nanos: ● From the Arduino official website, the Arduino Nanos draw ~20 mA nominally. https://bit.ly/2SrcumT

Vmin: 21VDC	According to the datasheet of our power supply, it will output 24VDC with 0.5% load regulation, meaning it will drop to a minimum of 22.8V at maximum rated load. https://bit.ly/2UWRi9I
Vnominal: 22-24VDC	According to the datasheet of our power supply, it will output 24VDC with 0.5% load regulation, meaning it will drop to a minimum of 22.8V at maximum rated load. Furthermore, it has a maximum output voltage of 28.8V, meaning we can get a theoretical minimum voltage of 27.37V at maximum rated load. https://bit.ly/2UWRi9I
Vmax: 26VDC	According to the datasheet of our power supply, it will output 24VDC with 0.5% load regulation, meaning it will drop to a minimum of 22.8V at maximum rated load. Furthermore, it has a maximum output voltage of 28.8V, meaning we can get a theoretical minimum voltage of 27.37V at maximum rated load. https://bit.ly/2UWRi9I

rspbrry_p_prmry_cntrl_pcb_dsig

Logic-Level: 3.3V	The source of this signal is the Arduino Nano, which according to the official user manual, utilizes an Atmega328p, which is capable of a 5VDC digital signal. https://bit.ly/2uyfkgN
Other: 115200 Baud	The source of this signal is an Arduino Nano running grbl, which translates G-Code from the Raspberry Pi to stepper motor signals. According to the official website for grbl, the Nano will output a step signal for every step it wants the stepper motor to step. This design is universal across all stepper motors, and is inherent by design. https://github.com/grbl/grbl
Other: Active-High	The source of this signal is an Arduino Nano running grbl, which translates G-Code from the Raspberry Pi to stepper motor signals. According to the official website for grbl, the Nano will output a direction signal, high for

	clockwise, and low for counter-clockwise. https://github.com/grbl/grbl
Other: UART Protocol	The source of this signal is an Arduino Nano running grbl, which translates G-Code from the Raspberry Pi to stepper motor signals. According to the official website for grbl, the Nano will output a step signal for every step it wants the stepper motor to step. This design is universal across all stepper motors, and is inherent by design. https://github.com/grbl/grbl

prmry_cntrl_pcb_vcm_pmp_dcpwr

Inominal: 0.5-0.8A	For the D2028B: • 12V and 12W maximum operation, makes for 1A peak current http://bit.ly/2tOJsEd • Allow nominal current to be 20%-50% below peak current • Adjusted through experimental observation
Ipeak: 1A	For the D2028B: • 12V and 12W maximum operation, makes for 1A peak current http://bit.ly/2tOJsEd • Adjusted through experimental observation
Vmin: 12VDC	For the D2028B: • Motor is rated for 12VDC operation http://bit.ly/2tOJsEd
Vmax: 15VDC	For the D2028B: • Motor is rated for 12VDC operation http://bit.ly/2tOJsEd • Motor can be ran at higher voltages (and therefore, higher currents) for short periods of time, at the cost of longevity to the motor.

	• Adjusted through experimental observation

prmry_cntrl_pcb_axlry_drvr_pcb_dsig

Logic-Level: 5V	The source of this signal is the Arduino Nano, which according to the official user manual, utilizes an Atmega328p, which is capable of a 5VDC digital signal. https://bit.ly/2uyfkgN
Other: Sends a step pulse signal	The source of this signal is an Arduino Nano running grbl, which translates G-Code from the Raspberry Pi to stepper motor signals. According to the official website for grbl, the Nano will output a step signal for every step it wants the stepper motor to step. This design is universal across all stepper motors, and is inherent by design. https://github.com/grbl/grbl
Other: Sends a direction signal	The source of this signal is an Arduino Nano running grbl, which translates G-Code from the Raspberry Pi to stepper motor signals. According to the official website for grbl, the Nano will output a direction signal, high for clockwise, and low for counter-clockwise. https://github.com/grbl/grbl

prmry_cntrl_pcb_axlry_drvr_pcb_dcpwr

Inominal: 1.8-2.2A	According to the datasheet for the TMC2209, the stepper driver we are using, there is a current limiting function. We will utilize this function to ensure that our stepper motors never exceed 2A for reduced heat output and increased component longevity. https://bit.ly/37sqxN0
Ipeak: 2.5A	According to the datasheet for the TMC2209, the stepper driver we are using, there is a current limiting function. We will utilize this function to ensure that our stepper motors never exceed 2A for reduced heat output and increased component longevity. https://bit.ly/37sqxN0

Vmax: 25VDC	According to the datasheet for the TMC2209, the stepper driver we are using, the IC is capable of handling up to a 29VDC input. We will be feeding this chip from a 24VDC power supply, so this will be a non-issue. https://bit.ly/37sqxN0
Vmin: 22VDC	According to the datasheet for the TMC2209, the stepper driver we are using, the IC is capable of handling as low as a 4.75VDC input. We will be feeding this chip from a 24VDC power supply with 0.5% load regulation, so this will be a non-issue. https://bit.ly/3bFQ1Ku https://bit.ly/37sqxN0

axlry_drvr_pcb_prmry_cntrl_pcb_dsig

Logic-Level: 5VDC	The source of this signal is an ATtiny85, which according to the official datasheet, has a 5VDC digital signal. https://bit.ly/2SLR82e
Other: USI Protocol	The source of this signal is an ATtiny85, which according to the official datasheet, utilizes the Universal Serial Interface. https://bit.ly/2SLR82e
Other: 9600 Baud	The source of this signal is an ATtiny85, which according to the official datasheet, is capable of up to 10 MHz, which is well over 9600 baud or 9.6 kHz. https://bit.ly/2SLR82e