Preliminary Design Document for Just A Really Excessive Defense System

Version 1.0

May 14, 2021

Table of Contents

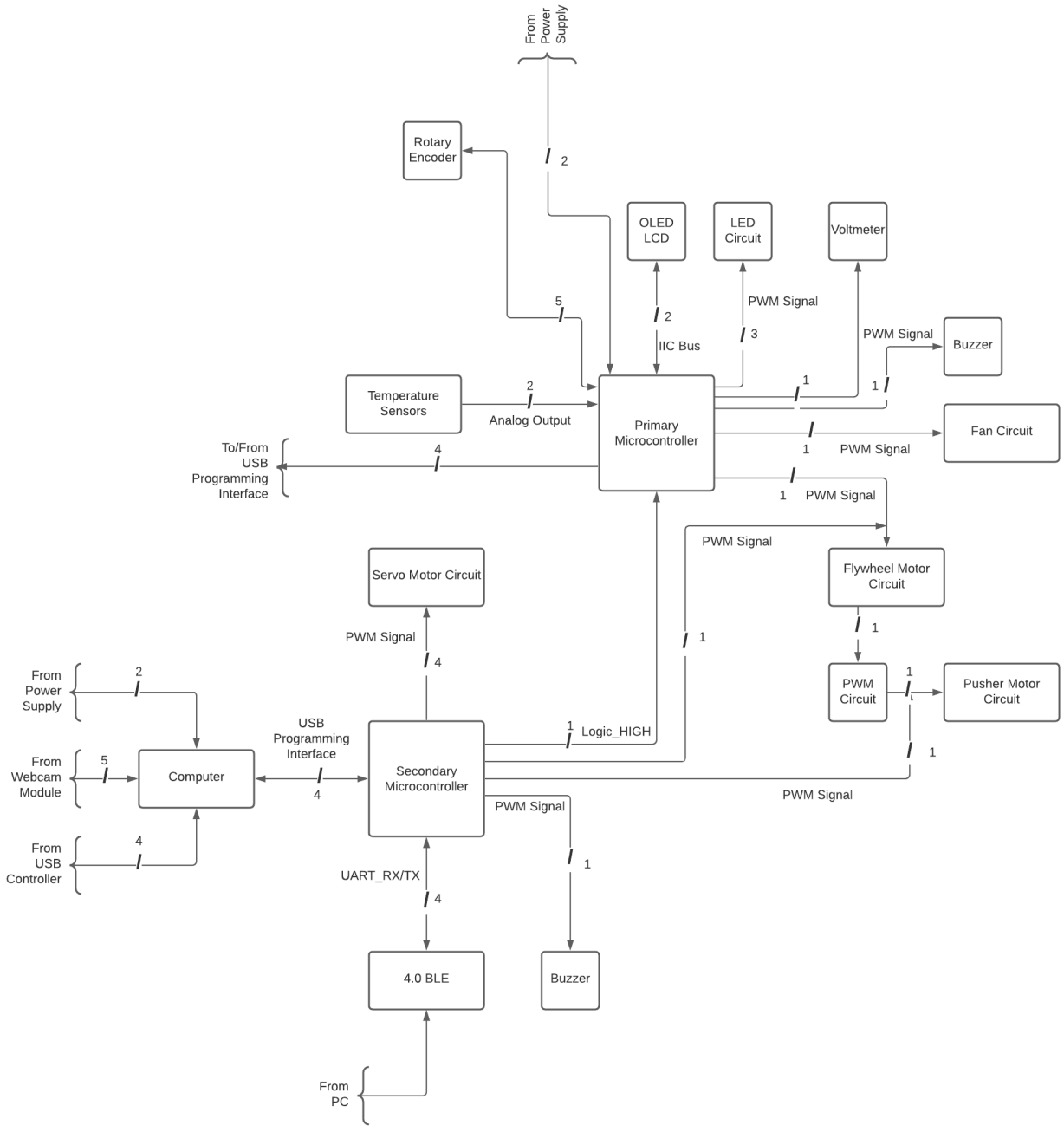
1. Introduction

J.A.R.E.D.S, or *Just A Very Excessive Defense System*, is a fully autonomous, dismountable smart sentry gun defense system that can automatically track and shoot targets, using a heavily modified Nerf gun. J.A.R.E.D.S is designed to be a versatile system that can be used to satisfy many different tasks. Namely, it can be used as a complete defense system to add security to locations, or as an alarm system that can alert the operator of unwanted activity. The product is targeted towards consumers who are looking for an effective, user-friendly security system, such as home defense/security enthusiasts.

This project is being designed for the TEJ4M course at the Earl of March Secondary School. The design and implementation processes must be completed in a timeframe of two months, for the duration of the entire course. The project manager will be Mr. Roller, a computer engineering and technology teacher at Earl of March.

2. Product Design

2.1 Block Diagram - Figure 2.1-1

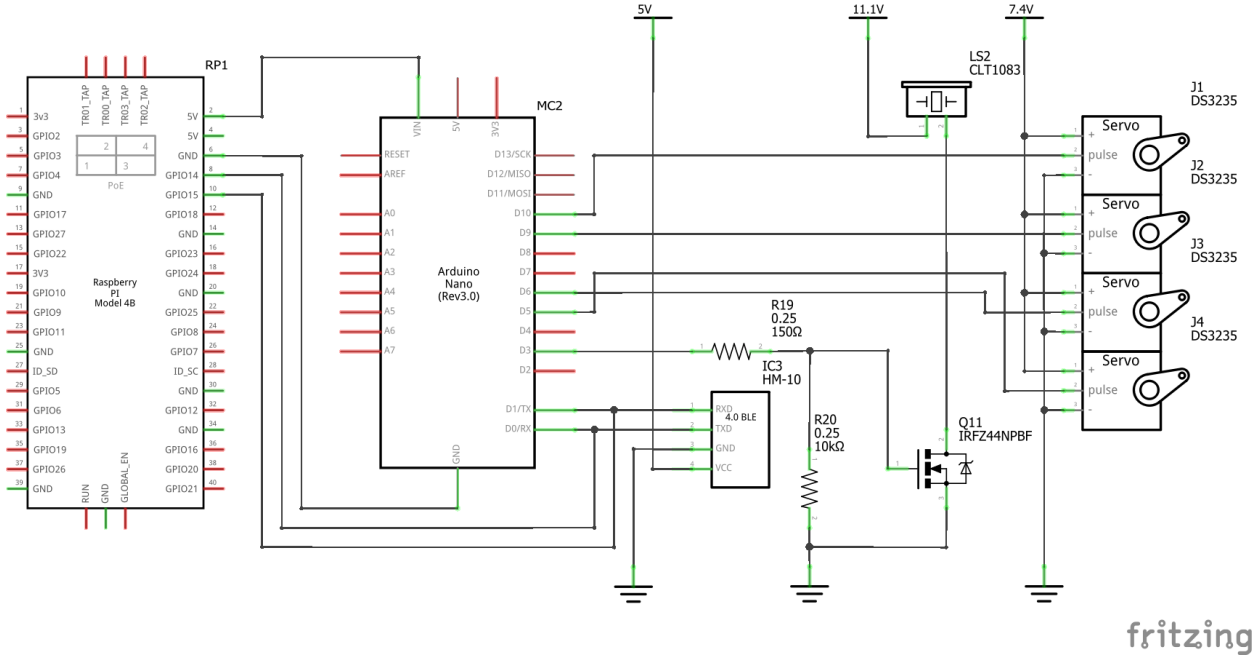


2.2 Sub-Circuit Descriptions

2.2.1 Sub-Circuit #1 - Frame Circuitry

This circuit is responsible for controlling the sentry gun frame, using servo motors to adjust the position of the sentry gun, as well as processing all of the incoming data from video streams and other user interfaces. This circuit contains 4xServo motors (J1, J2, J3, J4), 1x4.0 BLE Module (IC3), 1xBuzzer (LS2), an Arduino Nano (MC2) and a Raspberry Pi 4 Model B (RP1). The buzzer will be used connected to digital PWM pin 8 on the Arduino, which will serve as an alarm when a target is located, or for just basic, short audio cues to notify the user for a certain message, such as low battery levels, processing errors, hardware malfunctions, etc.

Figure 2.2.1-1



This circuit is the heart of the sentry gun and is the most integral part of the entire circuit. This sub-circuit will provide functionality to the rest of the sentry gun, and will allow the gun to operate autonomously without any user intervention, or controlled remotely via a USB controller or a bluetooth application. The Raspberry Pi will contain the software for auto-targeting and processing controller input, and the appropriate data will be transmitted to the Arduino, which will process the incoming data and determine how to position the sentry gun and servos in such a way to lock onto targets, and will be able to fire the gun when the target is locked on and the Raspberry Pi sends a signal to indicate that the gun is positioned correctly and ready to be fired.

The Raspberry Pi will be connected to the Arduino using a USB 2.0 port on the Pi and the Mini USB Type-B serial port on the Arduino. This interface will not only be used for uploading code and accessing the serial debug interface, but also plays an important role for operation. Using the PySerial library, a serial communication interface will be established between the two devices, allowing for data transmission between both the Pi and Arduino. The Arduino will be used to control most of the hardware of the system using the digital I/O pins, however the Raspberry Pi will be responsible for processing the webcam stream and running an OpenCV python script to track targets, and processing controller input for modes of manual operation. For more information on the USB interface see section **2.3.1.3**.

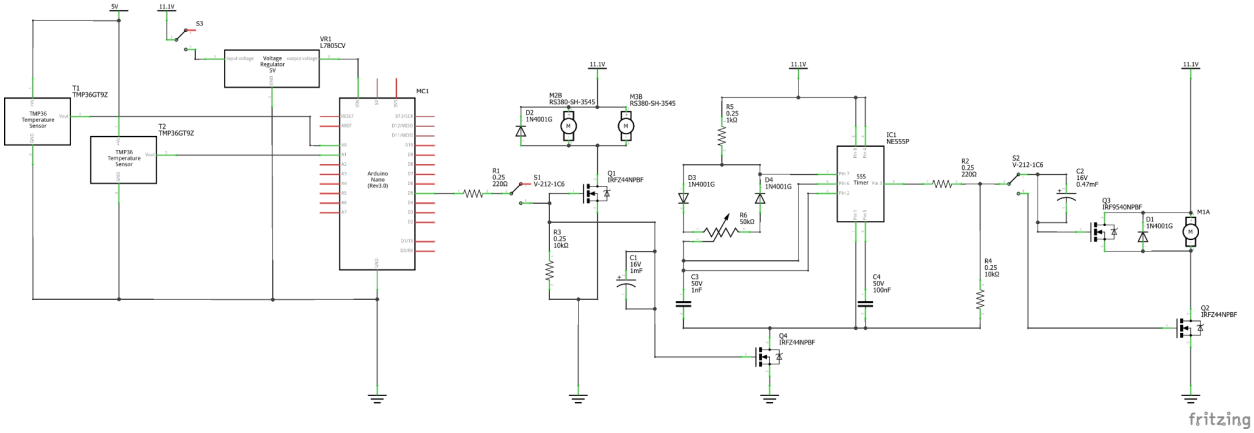
The servo motors are rated for 5-7.4v, and will be operated at 7.4v, which is the nominal voltage of a 2S LiPo battery pack, which in turn will also increase the maximum torque of the servo motors. Each signal pin will be connected to a digital PWM pin on the Arduino (Pins 5, 6, 9, 10). One servo will control the rotation of the sentry, the remaining 3 will be used to control the arm positioning. By generating a PWM signal, the internal circuitry of the servo motor will position the arm of the servo at an angle between 0° and 270°.

A 4.0 BLE module (IC3) will be connected to the Arduino Nano +5v, Gnd, and their respective TX/RX pins. This module will receive signals from an external bluetooth device, which can be used to control the sentry gun remotely. When a piconet is established between the module and another control device, the Arduino Nano will override manual controls, and will pause receiving data from the Raspberry Pi until the user disconnects their bluetooth device. Once controls are overridden, the Arduino will process all of the information independent of the Raspberry Pi. For more information, see section **2.3.1.2**.

2.2.2 Sub-Circuit #2 - DC Motor Circuitry/Main Firing & Temperature Sensors

This circuit controls the firing of the gun, both manual and autonomous. The main components in this circuit are the 3xDC motors (M1A, M2B, M3B), 2xTMP36 temperature sensors (T1, T2), a PWM module (**Sub-Circuit #7**), 2xSPDT switches (S1, S2), and an Arduino Nano (MC1) for processing. There are 3 DC motors in this circuit, and can be further classified into two different groups: 1xPusher motor, and 2xFlywheel motors. The gun uses a simple hopper-fed system to feed rubber balls (Ammunition) into a conveyor mechanism. This conveyor mechanism will feed the balls into a set of flywheels, which will force the balls out of the barrel, at approximately 130-150FPS. On each set of motors, there is a flyback diode to protect a MOSFET from voltage spikes and current surges, due to the motor inductance when the motors are switched off. Also, it should be noted that all MOSFETs in the circuit have 150Ω resistors to protect the I/O pins from large currents when the MOSFETs are initially switched on, and also have a 10kΩ pull-down resistor to keep the gate logic low and avoid a state of high impedance (HIZ), for defined logic inputs and reliable switching.

Figure 2.2.2-1



When the Arduino is powered, a pulse width-modulated signal will be generated constantly at pin D5. The duty cycle of this signal can be configured by the user (20% - 100%). This pin will lead to the first SPDT switch (S1). When this switch is pressed, an N-Channel MOSFET will be switched on (5v pin output; IRFZ44NPBF gate threshold: 2-4v) and ground the motor circuit, which will begin revving the motors up. The normally open of this switch will lead to the gate of another MOSFET, which will switch on/off the PWM circuit. A 1000µF filtering capacitor is added to the gate of the transistor to filter the PWM signal to allow for the PWM MOSFET to be switched on constantly when the switch is pressed. This will allow for a constant PWM output from the PWM circuit as long as the flywheel motors are revved up. It is wired this way so that the pusher motor can only be powered once the flywheel motors have power, thus preventing the gun from jamming when trying to force balls into still flywheels.

The output of the PWM circuit leads to the second firing switch (S2), with a 220Ω resistor to protect the IC pin from high currents, and a 10kΩ pull-down resistor to force the line logic low when not driven high by the IC. The normally open pin is connected to the gate of a MOSFET which will switch the pusher motor on and begin running the conveyor system. On the normally closed pin, there is a separate wire leading to another MOSFET, with another gate capacitor for filtering the PWM output. This MOSFET is responsible for switching on the motor braking circuit, which is essential in stopping the pusher motor immediately once the switch is released. When this MOSFET is switched on, it will short the two terminals of the motor. Due to the inductive nature of a DC motor, the generated current from the already rotating shaft will cause the shaft to spin in the opposite direction, which will slow the speed of the motor much faster than if not wired for motor braking, as the inertia of the shaft will cause it to continue to rotate even after the trigger is released.

Two TMP36 (T1, T2) temperature sensors will be used to monitor the temperature of the DC motors. The DC motors in this circuit are rated for 6v, however they will be overvolted at 12v in this circuit for extra performance, so it is crucial to be able to be aware of the temperature of the motors, and when to stop firing to let them cool. One sensor will be located on the pusher motor, and the other sensor will be located on either of the two flywheel motors. The connections for the temperature sensors are simple, and only require one wire leading to an analog pin on the

Arduino. For this design they will be connected to analog pins A0 and A1. The analog input from the sensors will be processed and displayed on the GUI.

2.2.3 Sub-Circuit #3 - J.A.R.E.D.S Interactive Interface GUI Hardware Components

This circuit is responsible for processing user input via incremental rotary encoder and displaying the GUI on an OLED LCD. This circuit is composed of an OLED LCD (LED1) to display the graphics, a rotary encoder (A1) to collect user input, and an Arduino Nano (MC1) for processing. The GUI will allow the user to configure different weapon settings, such as cooling fan speed, RGB LED levels, motor speed, and to monitor important information such as temperature readings from sensors.

Figure 2.2.3-1

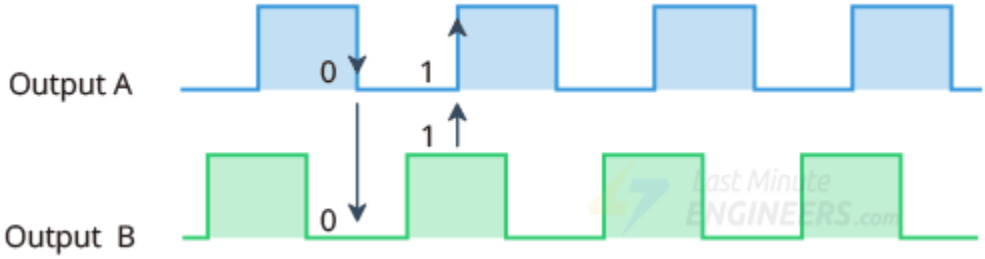


Image sourced from:
https://lastminuteengineers.com/wp-content/uploads/arduino/rotary-encoder-output-pulses-in-anticlockwise-rotation.png

The rotary encoder is connected to digital pins 2, 4, and 7 on the Arduino. The encoder uses two pulsed outputs: Output A, and Output B. When the user rotates the rotary encoder, two pulses are generated from the two output pins. When the top set of pulses proceeds the bottom set of pulses, rotation is clockwise and vice versa. The rotary encoder also has a push switch, which will be used to select options and data values in the GUI. The user will be able to rotate the encoder to navigate different menu items, and to change data values such as PWM duty cycle, RGB LED levels, sentry gun rate of fire, etc.

The OLED LCD will be connected to the Arduino's I2C bus. SCL and SDA pins will be connected to their respective lines on the bus. Pull-up resistors will be added to each line on the bus in order to restore the logic high after being driven low. For more information on this interface, see section **2.3.1.1**.

2.2.4 Sub-Circuit #4 - Microcontroller Interface

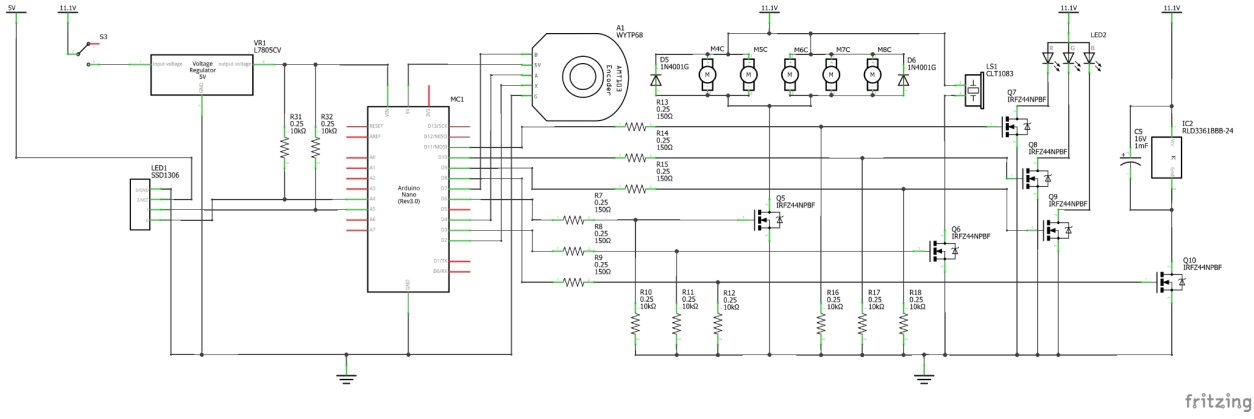
This simple circuit disables the primary microcontroller (MC1) when the secondary microcontroller (MC2) is connected to the system. This is to ensure there are no clashes between user input and autonomous operation, and the secondary microcontroller will gain full control over the system, controlling all of the components (See **Sub-Circuit #6**). When connecting the 10-pin female header to the 10-pin male header on the main gun circuit board,

one of the pins will connect from an Arduino digital output pin (pin 2). This wire will lead to the gate of a MOSFET which connects MC1 to ground, with a 150Ω gate resistor. The default state of this MOSFET will be high, due to the 10kΩ pull-up resistor connected to the MOSFET gate and positive power supply. However, when the 10-pin connector is connected to the circuit, the low output of MC2 pin 2 will switch the MOSFET off, breaking the circuit, thus disabling MC1.

2.2.5 Sub-Circuit #5 - Primary Microcontroller Hardware Components

This circuit contains all of the additional hardware components which add additional features to the sentry gun. Features include: Cooling fans for motors, RGB LEDs, a voltmeter to monitor the voltage of the power supply, and a buzzer. Components used in this circuit are 2x25mm cooling fans (M4C, M5C), 3x40mm cooling fans (M6C, M7C, M8C), a buzzer (LS1), an RGB LED strip (LED2), a digital voltmeter for voltage readings (IC2), and an Arduino (MC1). A simple SPST mini boat rocker switch is located on the side of the gun, which will be the main power switch to allocate power to the system. When this switch is switched on, a linear voltage regulator will convert the DC 12v to DC 5v, which will be used to power the Arduino. All components will be switched on/off using an N-Channel enhancement type MOSFET, with 150Ω gate resistors to protect the Arduino I/O pins from high currents, as well as a pull-down resistor at the gates of each MOSFET to keep the gate logic low when the output pins are not driven to a high state.

Figure 2.2.5-1



As the DC motors are being overvolted to 12v, some form of cooling will be required to prevent the motors from overheating, and to also extend the lifespan of the motors. For this reason cooling fans have been implemented into the circuit, and will be set up in a push/pull configuration to allow for optimal airflow inside of the gun. There will be 3 main sets of fans: 2x25mm fans for the flywheel motors, 2x40mm fans for the pusher motor and other internal components, and 1x40mm exhaust fan at the rear of the gun. All fan sets will be connected to the same MOSFET, due to limited PWM output pins on the Arduino.

RGB LED strips will be implemented into the circuit to illuminate the gun hopper, and can be customized to any colour by the user via the *J.A.R.E.D.S Interactive Interface* (See section

2.3.1.1 or **Sub-Circuit #3**). The LED strip is common cathode, so each RGB lead must be connected to ground. Each lead connects to ground through a separate MOSFET. The gates of these MOSFETs are switched on/off using a PWM output pin to simulate an analog signal (0-255), in order to have access to the full colour palette. A buzzer is connected to the circuit, which will play a short audio cue when power is switched on. The ground wire is connected to a MOSFET, with the gate being driven by a PWM output pin on the Arduino. A voltmeter is also connected to the circuit in a similar fashion as the buzzer, however the MOSFET gate is driven by just a standard digital I/O pin, to power on/off the voltmeter for power saving settings. A 1000µF electrolytic capacitor (C5) is connected in parallel with the voltmeter to smooth out the line for any power supply voltage drops to generate a consistent reading.

The table below shows the corresponding pins for each of the MOSFETs in the circuit.

Figure 2.2.5-2

I/O Pin	Part	PWM
3	Buzzer	Yes
6	Cooling fans	Yes
8	Voltmeter	No
9	LED_Blue	Yes
10	LED_Green	Yes
11	LED_Red	Yes

2.2.6 Sub-Circuit #6 - Secondary Microcontroller Hardware Components

This is the circuit which connects the secondary microcontroller to the rest of the system, and all of the hardware components in the gun. This will allow controls to be overridden by the Arduino and sentry gun firmware, so the gun can shoot autonomously. The main components in this circuit are an Arduino (MC2), and all of the hardware listed in **Sub-Circuit #5**. As with Sub-Circuit #5, gate resistors and pull-down resistors of the same values will be used.

When mounting the sentry gun to the frame, there will be a 10-pin male header on the side of the gun which will be used to connect the sentry gun hardware to the secondary microcontroller, which will do a lot of the processing and drive the circuit. Below is a list of all the connections for the circuit header.

Figure 2.2.6-1

Pin No.	From	To	Type	Connection Description
1	MC2-2	Q12_Gate	Low	Disable MC1
2	MC2-11	Q1_Gate	Low/High	Flywheel motor control
3	MC2-12	Q3_Gate	Low/High	Pusher motor braking circuit
4	MC2-13	Q2_Gate	PWM	Pusher motor control and speed adjustment
5	MC2-4	Q5_Gate	HIGH	Cooling fan system Vcc
6	MC2-7	Q10_Gate	Low/High	Voltmeter control
7	MC2-A0	Q7_Gate	Low/High	RGB_red control
8	MC2-A1	Q8_Gate	Low/High	RGB_green control
9	MC2-A2	Q9_Gate	Low/High	RGB_blue control
10	NC			

The circuit essentially works identically to **Sub-Circuit #5**, however the system logic is being controlled by MC2 as opposed to MC1. All of the connections are made to the gate of each MOSFET. However, due to lack of PWM output pins on the Arduino, most components will not have PWM capabilities, and will only be in an on/off state, with the exception of the pusher motor (Q2_Gate). As MC1 is disabled when the 10-pin connector is connected to the 10-pin male header on the circuit board, there will be no conflicting logic and the system will operate smoothly and efficiently without problems.

2.2.7 Sub-Circuit #7 - PWM Circuit

This simple PWM circuit will use a 555 timer IC chip from Texas Instruments to generate a pulse width-modulated signal, which will be used to control the pusher motor RPM, and thus sentry gun rate of fire. A 50kΩ rotary potentiometer will be used to modify the PWM duty cycle, to vary the speed of the motor. The components used in this circuit are the 555 Timer from Texas Instruments (IC1), a 50kΩ rotary potentiometer (R6), two 1N4001G rectifier diodes (D3, D4), A 1kΩ charging resistor (R5), and two capacitors: 1nF and 100nF (C3, C4).

Figure 2.2.7-1

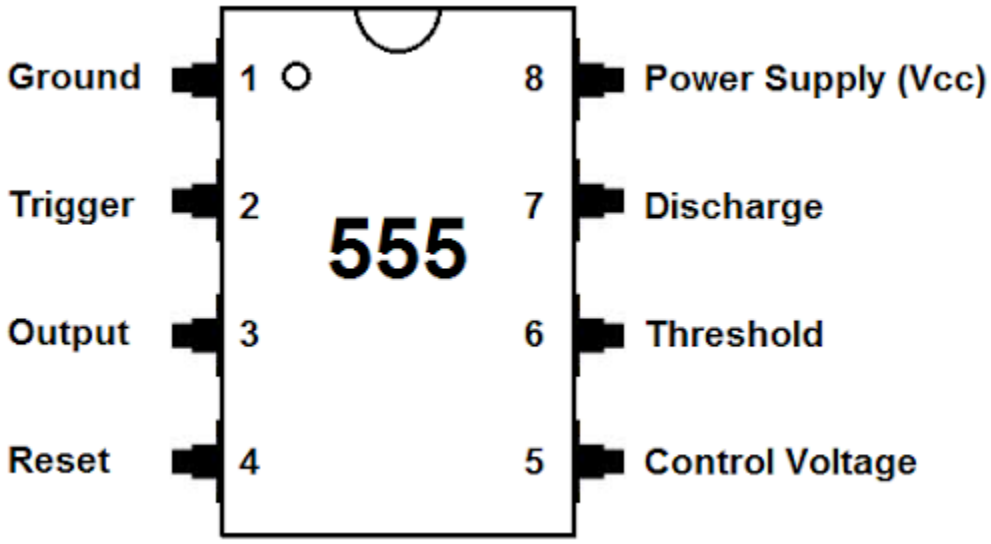


Image sourced from:
https://lh3.googleusercontent.com/proxy/o-I-cxAF0mxhcP4Oi6_SuKp6slsuoaoy10daJ0mAZnAw-NNEY6Cto1CLFcgE5Ctsn4Yt2VaX5FAdvBvbFXA7uUo252h-GhvdjEatudFwOuRAfRnHMCMVFyU8Gw

Pin 7 on the 555 timer is the discharge pin. A capacitor (C3) is connected to this pin, as well as through the positive voltage supply through the charging resistor (R5), and to ground. The capacitor will charge through the resistor. When the capacitor reaches a threshold value, internal components will activate the discharge through pin 7. This process will produce a pulse signal at the output pin 3. By varying the potentiometer resistance, the charge:discharge time ratio will change as a result, effectively changing the duty cycle and the width of the pulses at the output.

2.3 Product Interfaces

2.3.1 Internal Interfaces

2.3.1.1 MC1 - LED1 I2C Bus for GUI

From	To	Signal Name
MC1-A4	LED1-0	IIC_SCL
MC1-A5	LED1-1	IIC_SDA

This communication interface will be used to display the graphics for the GUI on the OLED LCD. The interface uses the IIC, or Inter-Integrated Circuit bus, which is a synchronous serial communication protocol created in 1982 by Philips Semiconductor. The IIC bus is a two wire

bus, one wire for the clock signal and another wire for the data transmission line. A master device (Arduino) will be connected to the bus, along with a slave device (OLED). Devices will be wired in parallel with the bus, and up to 127 devices can be connected. Communication will be initiated between the master and slave devices when the master generates a clock signal on the IIC_SCL (Serial clock) line. The slave will receive this clock signal and wait to be addressed by the master. Once the clock signal is generated, the master will send a start bit, followed by a 7-bit peripheral address, followed by a read/write bit, on the IIC_SDA (Serial data) line. If the slave receives the data, it will respond with an ACK (acknowledge) bit to indicate that the data has been received, and will follow with an internal registry address, and communication will continue, with an ACK bit after every byte, to ensure that every byte transmitted is received correctly. The I2C bus is also open drain, so each line will have a pull-up resistor to restore the signal to high when no device is setting it low.

2.3.1.2 MC1 - IC3 BLE 4.0 Module Communication

From	To	Signal Name
MC2-0	IC3-2	UART_RX
MC2-1	IC3-1	UART_TX

This communication interface will be used to communicate between the Arduino and 4.0 BLE receiver module for bluetooth signals. This communication interface uses UART, or Universal Asynchronous Receiver Transmitter. The two pins that will be used are the RX/TX pins for both devices. As UART transmits data asynchronously, there is no common clock signal between the two devices, so the data transmission rate must be set for both devices (For the purposes of this project, the clock speed will be set to 9600bauds/s). Both RX/TX pins will be used to transmit data between the devices (See **Figure 1.1**). Each byte of data will be indicated by a start bit, which in this case is a low pulse. The end of the byte will be indicated by an end bit, which in this case will be a high pulse.

2.3.1.3 RP1 - MC2 USB Interface for Programming/Communication

From	To	Signal Name
RP1-8	MC2-0	USB_Data+
RP1-10	MC2-1	USB_Data-

This communication/programming interface will be used for two main purposes: To upload code to the Arduino, and to communicate between the host device (Raspberry Pi) and peripheral device (Arduino). The interface uses USB, or Universal Serial Bus, which is a synchronous

serial communication interface. The USB Type-A connector has 4 wires: Vcc, Gnd, Data+ Data-. The Vcc and Gnd pins will power most peripheral devices (An Aduino for this project) from the host device without the need of an external power supply. The other two wires, Data+ and Data-, are used for data transmission between devices. Each packet is sent in frames. Two data wires are used to reduce the electromagnetic noise of the lines, using half-duplex differential signalling which essentially doubles the amplitude of the signal. For more information on USB, see USB 101: An Introduction to Universal Serial Bus 2.0.

2.3.2 External Interfaces

2.3.2.1 Arduino Mini Type-B USB 2.0 (Serial Debugging/Programming Interface)

The Mini USB Type-B port on the Arduino Nano is a serial debugging/programming interface that will be used for uploading code and accessing the serial monitor in the Arduino IDE, for debugging and other purposes. The Arduino will be connected to a host computer that will be used to upload code. Any standard USB Type-A to USB Type-B connector can be used to connect this device to a host computer; therefore no special or custom connectors are required for this external interface. For more information on USB and a brief description of the protocol, see section **2.3.1.3**.

2.3.2.2 Gaming Controller USB 2.0

The controller will use a USB Type-A connector to connect to the host device. Most controllers use a Micro USB port, however older models can have cables fixed to the controller or connected with a proprietary connector. Although, most gaming controllers come with a supplied USB connector for charging/connecting to the gaming console. For more information on USB and a brief description of the protocol, see section **2.3.1.3**.

2.3.2.3 Webcam Module USB 2.0

The Raspberry Pi is connected to an external webcam via USB, and the video stream serves as the input for the target-tracking software. The webcam module is connected to the host device using a USB Type-A connector, however the module connector itself uses a 5 pin JST-SH (1mm pitch) connector, with the following pinouts:

Pin	Signal
1	VCC
2	Data+
3	Data-
4	GND

5	NC

For more information on USB and a brief description of the protocol, see section **2.3.1.3**.

2.3.2.4 Bluetooth 4.0 Receiver/Transmitter

A 4.0 BLE (Bluetooth low energy) module will receive data from an external bluetooth device running an application to remotely control the sentry. In order for the device to transmit/receive signals, an antenna supplied chip is required to encode/decode/transmit data via antenna. When connecting, the module pings nearby bluetooth devices, and a piconet is established once both devices are connected, which will allow for communication using short radio waves, with the signals operating at a frequency of 2.4GHz to 2.48GHz.

3. Scheduling

3.1 Tasks

3.1.1 Design Phase/Open

Tasks which can be completed at any time throughout the duration of the project

1. Order all parts
2. Design 3D models for prototype pieces using Onshape
    - Rear housing for pusher motor to keep in place
    - New battery compartment to house 3S LiPo battery pack
    - System for mounting/dismounting Nerf gun to frame
    - Any other small pieces for either functional/aesthetic purposes
3. Export finalized models to .STL file format and 3D print using PLA

3.1.2 Hardware

1. Nerf gun modifications/Schematic implementation
    a. Record firing demo to compare performance
    b. Disassemble Nerf gun, remove all screws and remove all internals
    c. Modify shell of the blaster to account for and house new components (e.g. Slots for cooling fans, external wires, OLED LCD, etc.)
    d. Apply primer and new finish with clear coat with new colour scheme (Black/White/Red)
    e. Solder PWM circuit
    f. Solder main components from Figure 1.2a and Figure 1.2b onto perfboard
    g. Solder/connect all components to main circuit board

   h. Mount all components to modified Nerf gun
   i. Reassemble modified Nerf gun and proceed with coding Arduino to test circuit and new components
2. Frame Construction
   a. Design frame using Onshape, including dimensions in inches
   b. Saw wood into dimensions and sand until smooth
   c. Drill holes and sockets for dowel/servo motors
   d. Assemble frame and servo motors
   e. Solder components in Figure 1.1 to perfboard
   f. Solder/connect all servos and other components to the Arduino
   g. Test frame for structural stability, diagnose any potential problems and make any necessary adjustments
   h. Proceed with coding Raspberry Pi and Arduino

3.1.3 Firmware

Most if not all of section 3.1.2 must be completed before proceeding with firmware tasks

1. Upon completion of the Nerf gun, program basic features to Arduino to supply basic functionality and operation to the system
   - Basic GUI implementation using rotary encoder and OLED LCD to allow user to configure system settings
     - PWM control for flywheel and pusher motors
     - Fan speed control
     - LED RGB levels
     - Basic GUI navigational features and graphics for aesthetic purposes
2. Upon completion of the frame, program Arduino and servo motors to test for functionality and stability
   - Program servo motors for basic rotation along axes, for swivel and tilt rotation for a full range of motion
3. Continue/finish programming Nerf gun GUI
4. Begin programming Raspberry Pi and setup serial communication between Pi and Arduino
5. Program main sentry gun functionality and alternate operating modes
   - Develop bluetooth application for PC to control sentry gun remotely
   - Program autonomous operating mode in Raspberry Pi using OpenCV to track faces and motion
   - Program manual operating mode using USB gaming controller
   - Program bluetooth operating mode using separate bluetooth application

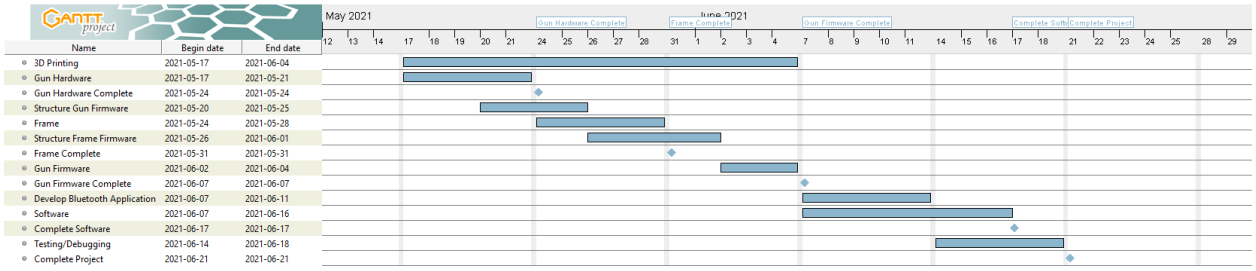
3.1.4 Finalize Project/Debugging

Final project testing/Debugging/Implement any uncompleted and/or last-minute features

1. Testing/Debugging
    a. Extensively test both system hardware and firmware of the finished product for reliability and stability and/or any potential design flaws
    b. Make any necessary improvements to the design to ensure a reliable product
2. Implement new features
    a. Implement any small, last-minute desired features/components not listed in the design documentation
    b. Debug any and all new features added, and re-evaluate the reliability and stability of the product again by running extensive tests

3.2 Gantt Chart Schedule

Figure 3.2-1



APPENDIX

Appendix A. Schematic

Figure 1.0 - *J.A.R.E.D.S_schem*

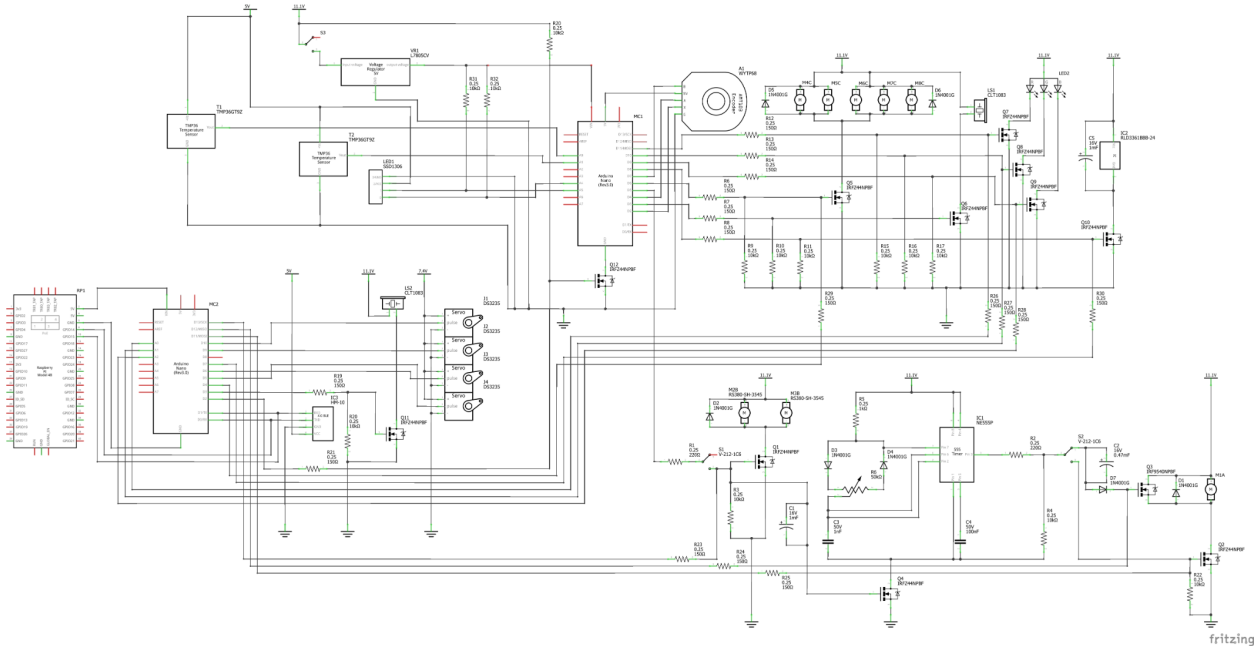


Figure 1.1 - *J.A.R.E.D.S_Frame-Hardware_schem*

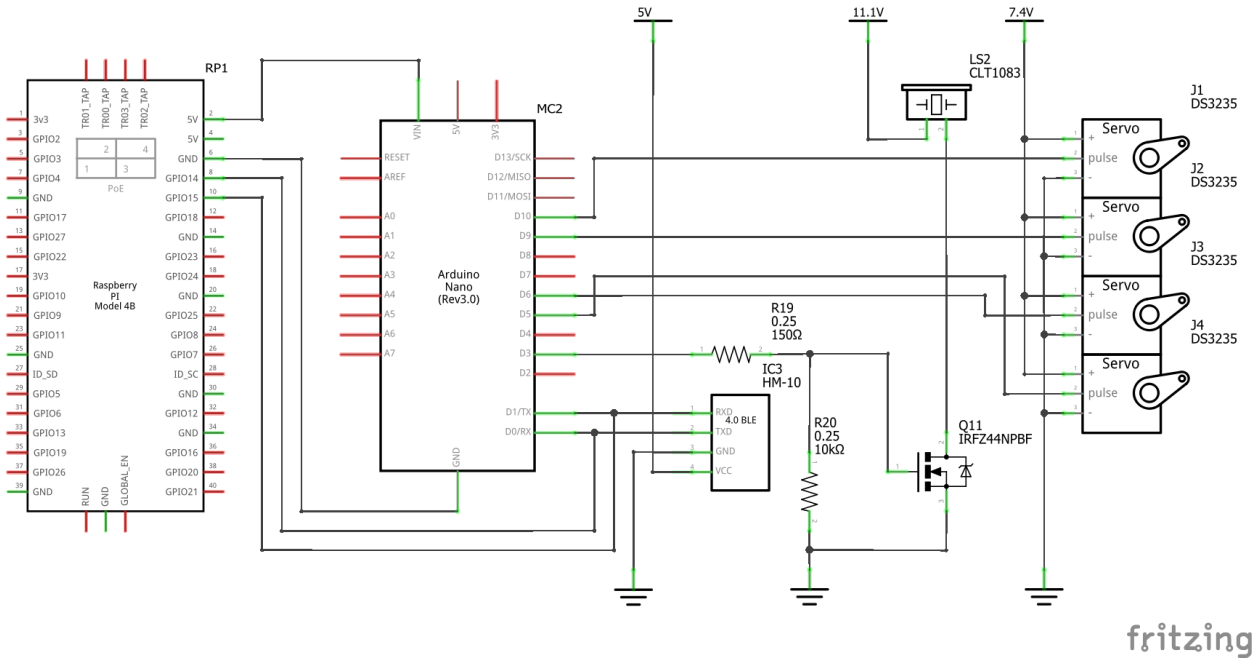


Figure 1.2 - *J.A.R.E.D.S_Gun_schem*

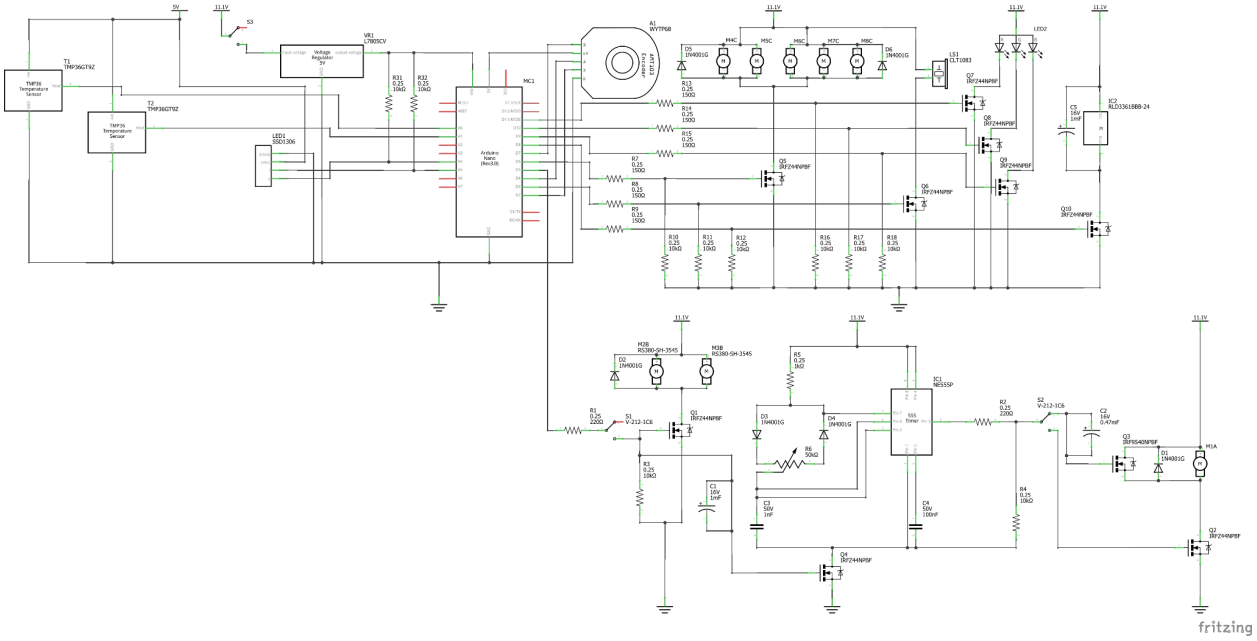


Figure 1.2a - *J.A.R.E.D.S_Main-Firing-Circuitv1_schem*

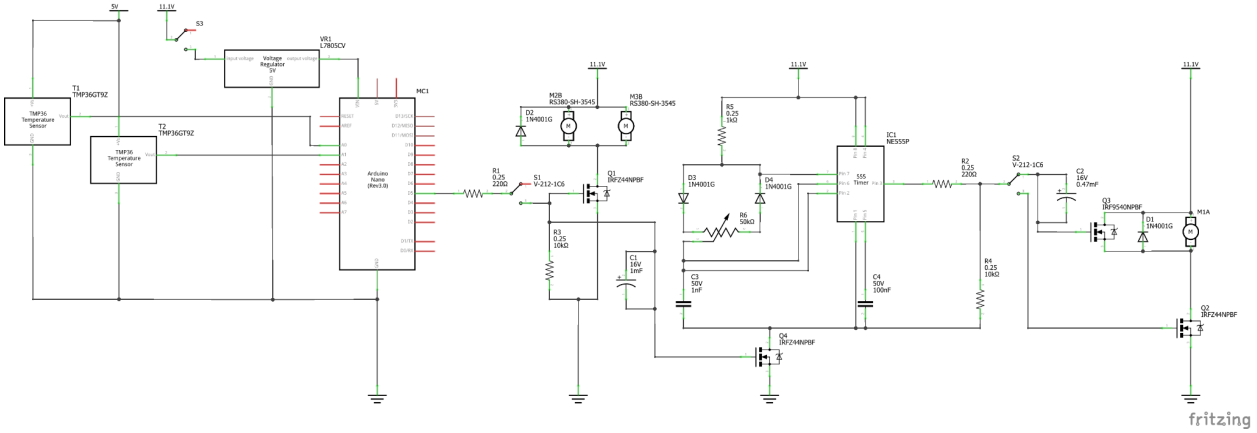
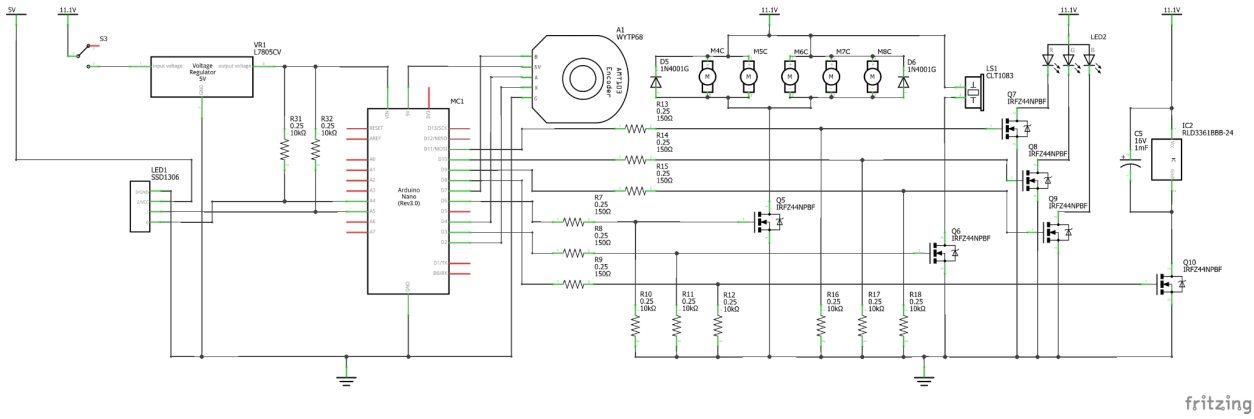


Figure 1.2b - *J.A.R.E.D.S_Secondary-Circuit_schem*



Appendix B. Parts List

Reference Designator	Quantity	Part Descriptor	Part Number	Link
MC	2	Arduino Nano		Arduino Nano
RP	1	Raspberry Pi 4B		Raspberry Pi 4 Model B
M1A	1	180 Brushed DC Motor		MTB Wolverine 180
M2B M3B	2	380 Brushed DC Motor	RS-380SH-3545	380 Rival Motor
M2C M3C	2	25mm DC 12V Brushless Cooling Fan	GDA2510B12N1 CP20215-CA	25mm Cooling Fan
M6C M7C M8C	3	40mm DC 12V Brushless Cooling Fan	AN-YDM4010B12	40mm Cooling Fan
Q3	1	P-Channel Enhancement Mode MOSFET	IRF9540NPBF	IRF9540NPBF
Q* ~~Q3~~	11	N-Channel Enhancement Mode MOSFET	IRFZ44NPBF	IRFZ44NPBF
C1 C2 C5	3	Electrolytic Capacitor		
C3 C4	2	Ceramic Capacitor		
LED1	1	0.96” I2C 128x64 OLED LCD Display	SSD1306	0.96" I2C 12864 OLED LCD Display
LED2	1	Common Cathode RGB LED Strip		
VR	1	5V Linear Voltage Regulator	L7805CV	5V Linear Voltage Regulator

LS	2	DC 3-24V Electronic Buzzer	CLT1083	Electronic Buzzer
T1 T2		Temperature Sensor	TMP36GT9Z	Temperature Sensor
S1 S2	2	21A SPDT Genuine Omron Microswitch	V-212-1C6	21A Genuine Omron Microswitch
S3	1	Mini SPST Boat Rocker Switch		SPST Boat Rocker Switch
J	4	35kg High Torque Coreless Digital Servo Motor	DS3235	35kg Digital Servo Motor
A	1	360° Quadrature Rotary Encoder w/ Push Button	WYTP68	Rotary Encoder w/ Push Button
D*	7	50V Rectifier Diode	1N4001G	50V Rectifier Diode
R6	1	50kΩ Rotary Potentiometer		
R* ~~R6~~	31	0.25W Carbon Film Resistor		
IC1	1	555 Timer	NE555P	555 Timer
IC2	1	0.36” Digital Voltage Meter	RLD3361BBB-24	Digital Voltmeter
IC3	1	4.0 BLE Module	HM-10	4.0 BLE Module