EEL 4924 Senior Design

Project: Glorious Freedom

Team: Aperture Laboratories

Submitted by:

James Bocinsky (CE) and Mason Rawson (CE)

Abstract:

James Bocinsky and Mason Rawson, both CE majors, will be pairing up as a team to create an autonomous turret. The primary subsystems of the project will include a battery power supply, a camera, a central processing controller, and pan-tilt mounted nerf guns. The brain will use opencv with face detection to track an object's motion. Software will calculate where and when to shoot nerf guns based on current object movement. We also have a manual override remote controller using wifi communication that allows a user to aim and shoot the turrets.

Features:

The overall goal of the glorious freedom sentry is to autonomously track an object and fire nerf guns at it. The project is inspired by the autonomous sentry turrets from the game "portal", and the team will stick to this theme throughout the project. There will be one sentry (the overall shooting system) which has two turrets. Each turret has an independently controlled and aimed nerf gun.

The sentry will have two modes of operation, an autonomous mode and a manual mode. During autonomous mode, the sentry will be controlled by image processing via OpenCV. The image processing system, a raspberry pi 3 (pi), will use video input from our camera to the OpenCV library to pick out an object from a still background. The pi will then use SPI to send the objects pixel location to the main board. The main board will then translate the current object's location to motor positions for each turret. Once the position translation has been calculated, each turret will aim at the object and fire it's nerf gun if the object is no longer moving.

During manual mode, the turret will be able to be controlled by a human using a separate controller. The controller will have input from buttons and a joystick. The joystick will control where the nerf guns point. Buttons will control commands and state. The button commands are shoot left nerf gun, shoot right nerf gun, shoot both nerf guns, reset aiming position, and toggle the mode of operation. The commands and joystick information will be sent to the main board on the sentry via a wifi module. Once the sentry has received the information it will operate as a user should expect; moving the sentries according to the joystick and firing the nerf guns according to a corresponding button press.

Technology Selection:

Our goal was to minimize mechanical work and custom electrical subsystems by using off-the-shelf parts. This would allow us to pursue advanced embedded computing with plenty of room to grow within the embedded computing.

To minimize mechanical and custom electrical work we took various steps. First we bought pan tilt mounts instead of 3D printed custom mounts. We will also use a USB camera to simplify this interface. The pi was chosen for video input as it was OpenCV friendly and powerful enough to handle the library. This allowed us to focus on the controls aspect of the project on the main board over configuring and setting up opencv on a raw barebones microcontroller. We took apart and used DC motors from an electric nerf gun instead of creating one from scratch. Our main board, using an msp432 and supporting an ESP8266, had very minimal analog components to allow us to focus on the controls and software aspects of the project. The controller board had an msp432 as well to keep things consistent across the project. It supported an ESP8266 as well for wifi communication, a joystick and buttons for human interactivity.

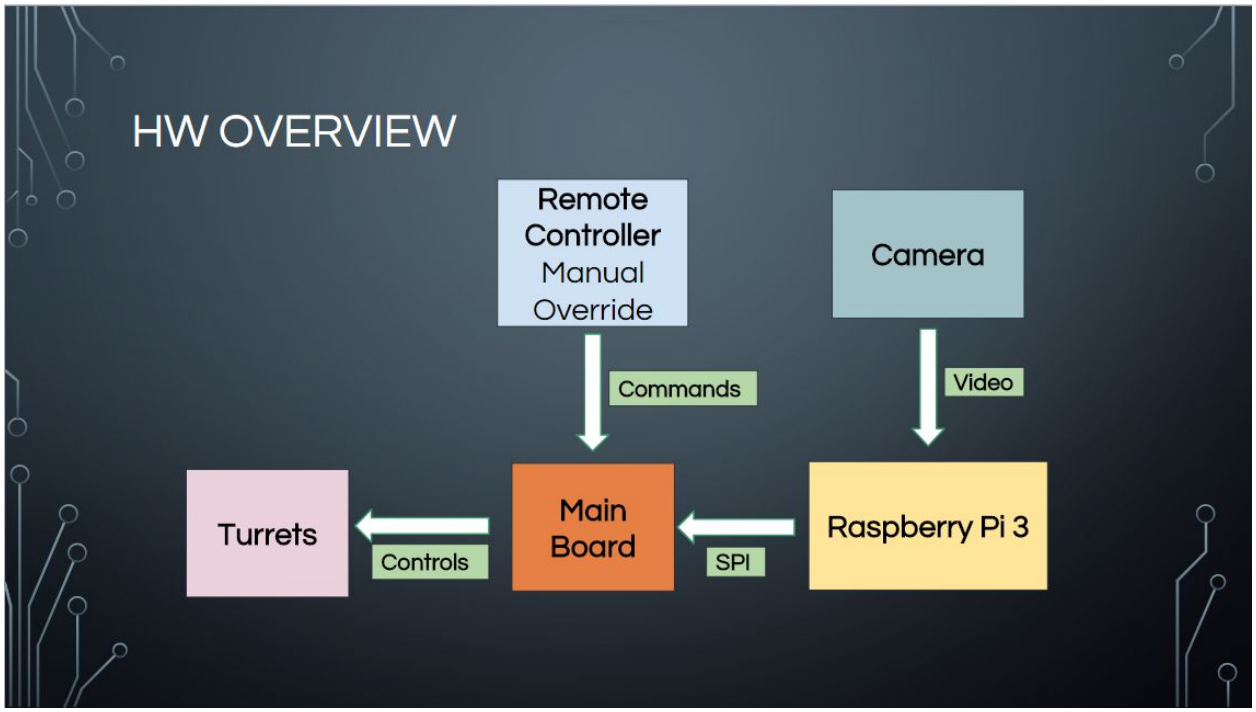
For software we utilized python for OpenCV, lua scripts for the ESP8266, C and C++ for our on board processing and controls. We used the openCV library, the most commonly used free computer vision software to assist in our face detection.

Technology:	Quantity:
Steppers and Drivers	2
Metal Pan Tilt Mounts	2
Nerf Guns	2
Camera	2
Raspberry Pi 3	1
Joystick	1
Buttons	6
ESP8266 NodeMCU Wifi Module	2
MSP432	2

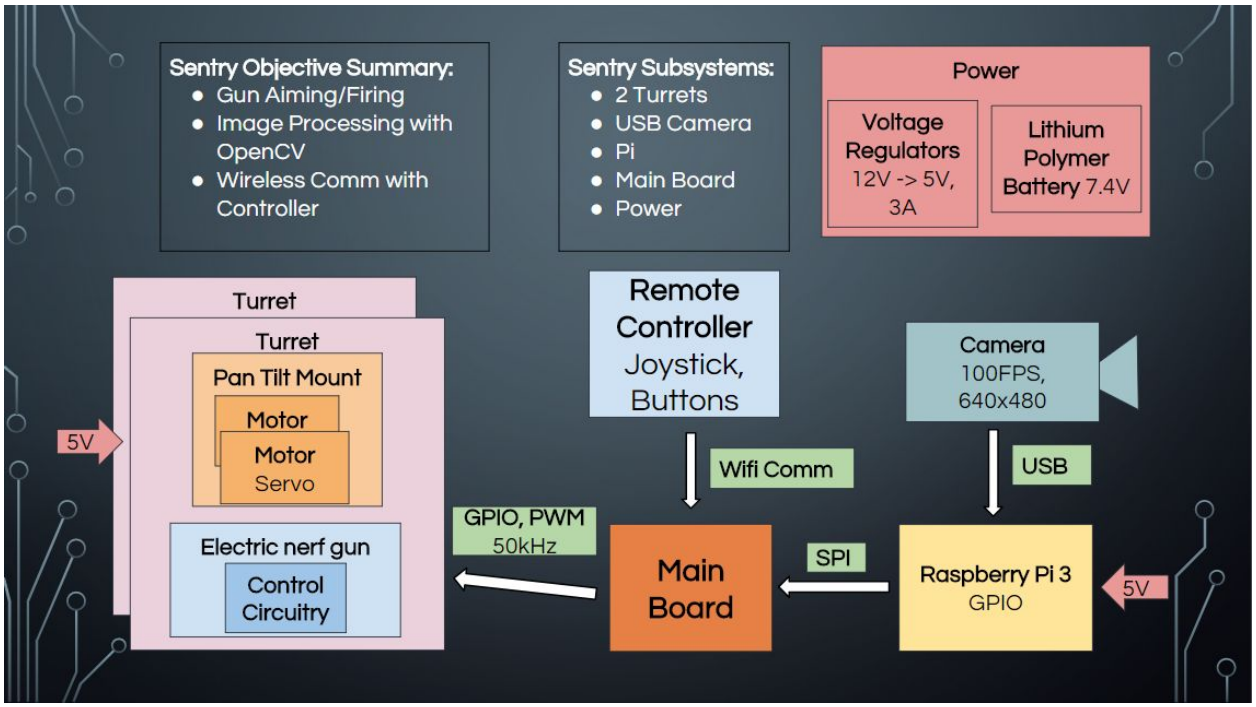
4.5V Battery Pack	1
Lithium Polymer Battery 7.4V	1
Portable USB Battery Charger	1
6 Bullet Ammo Clip	2
Voltage Regulator	4
Buck Converter	1
Various Resistors (see schematic for values)	x
Various Capacitors (see schematic for values)	x

Hardware Diagrams:

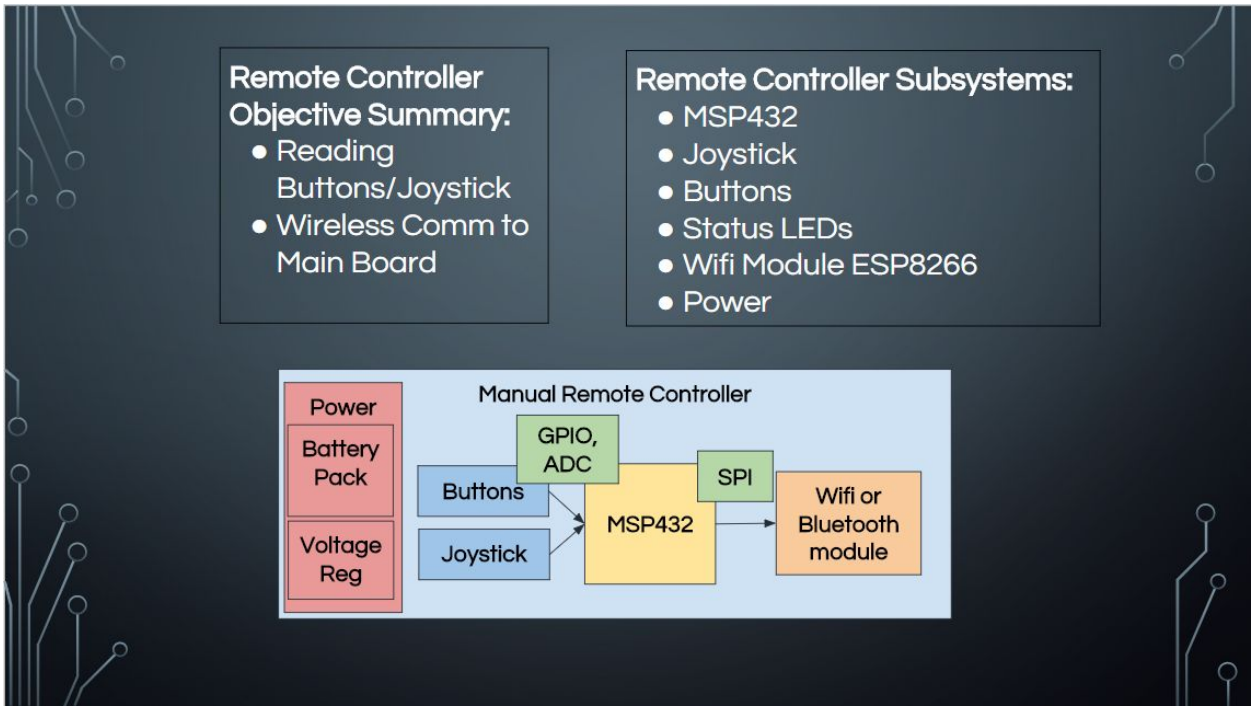
System Overview:



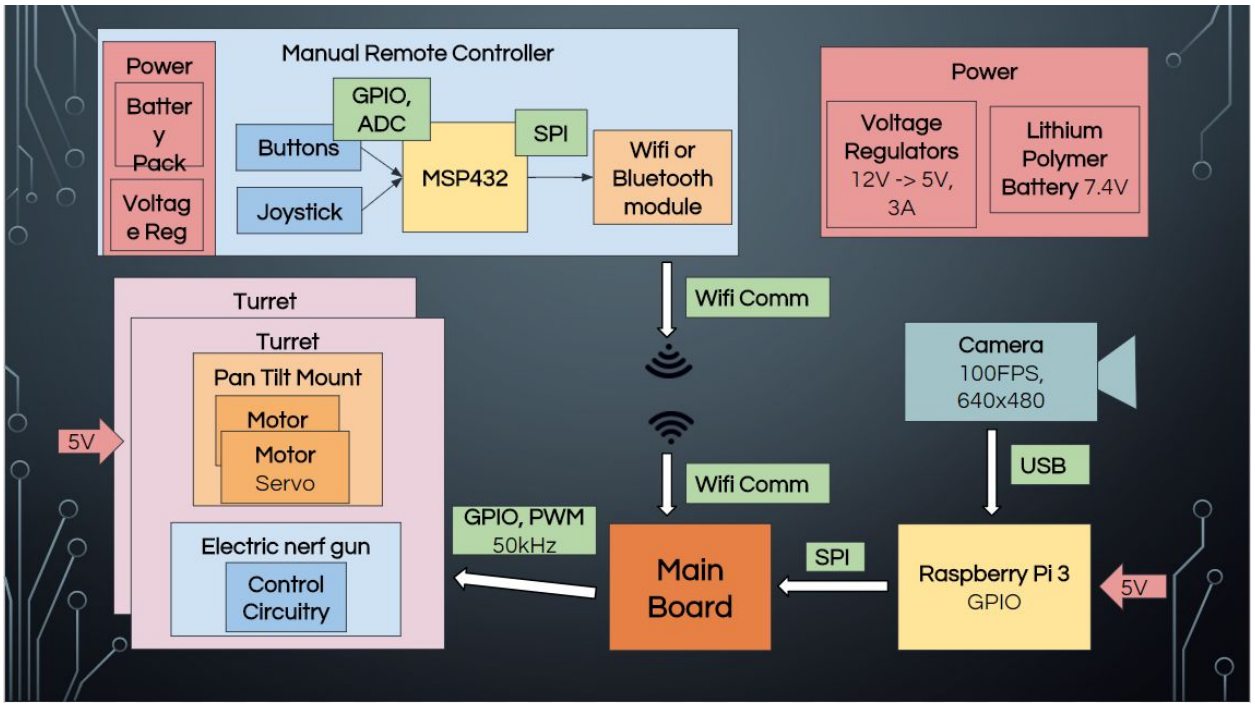
Sentry Overview:



Controller Overview:

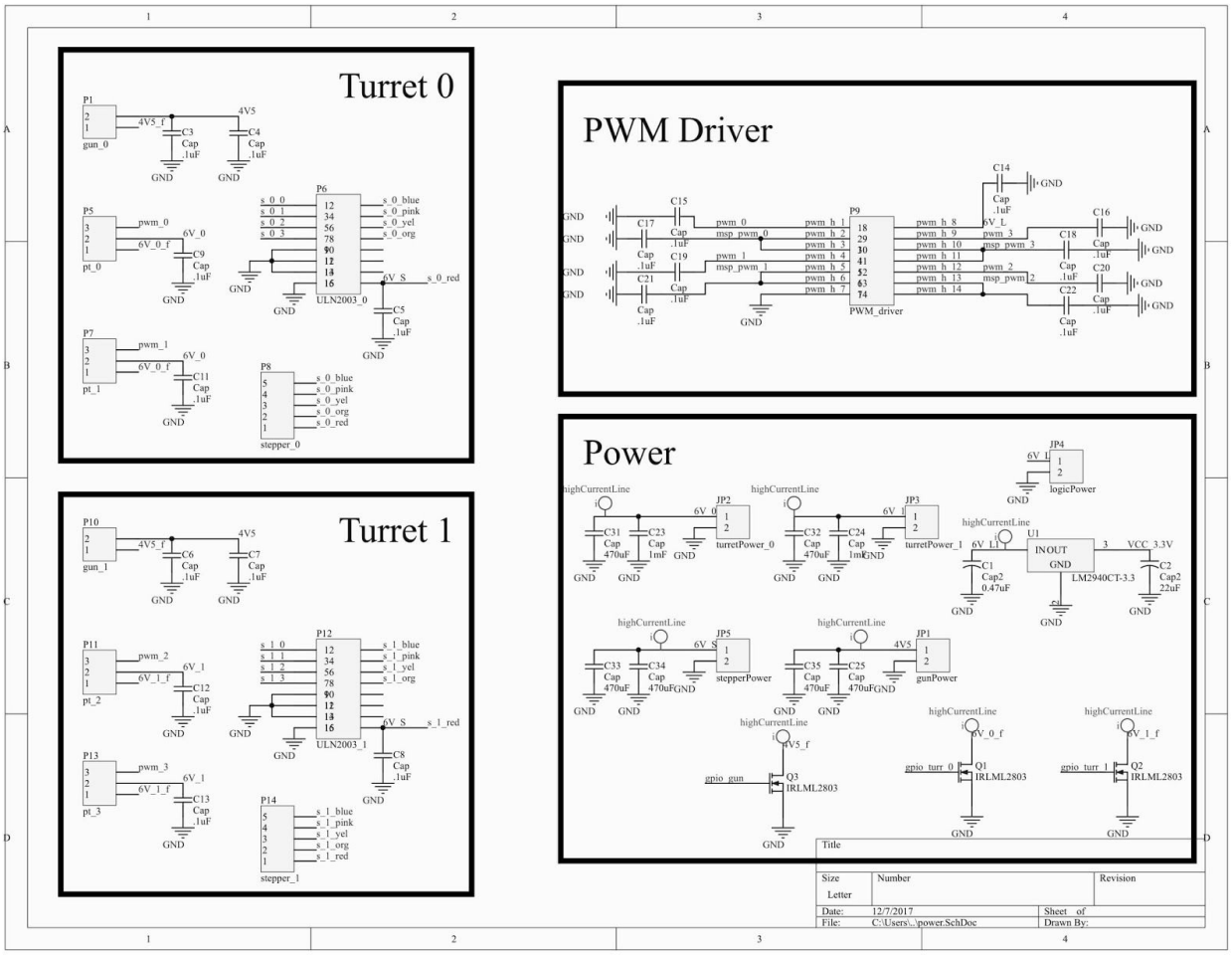


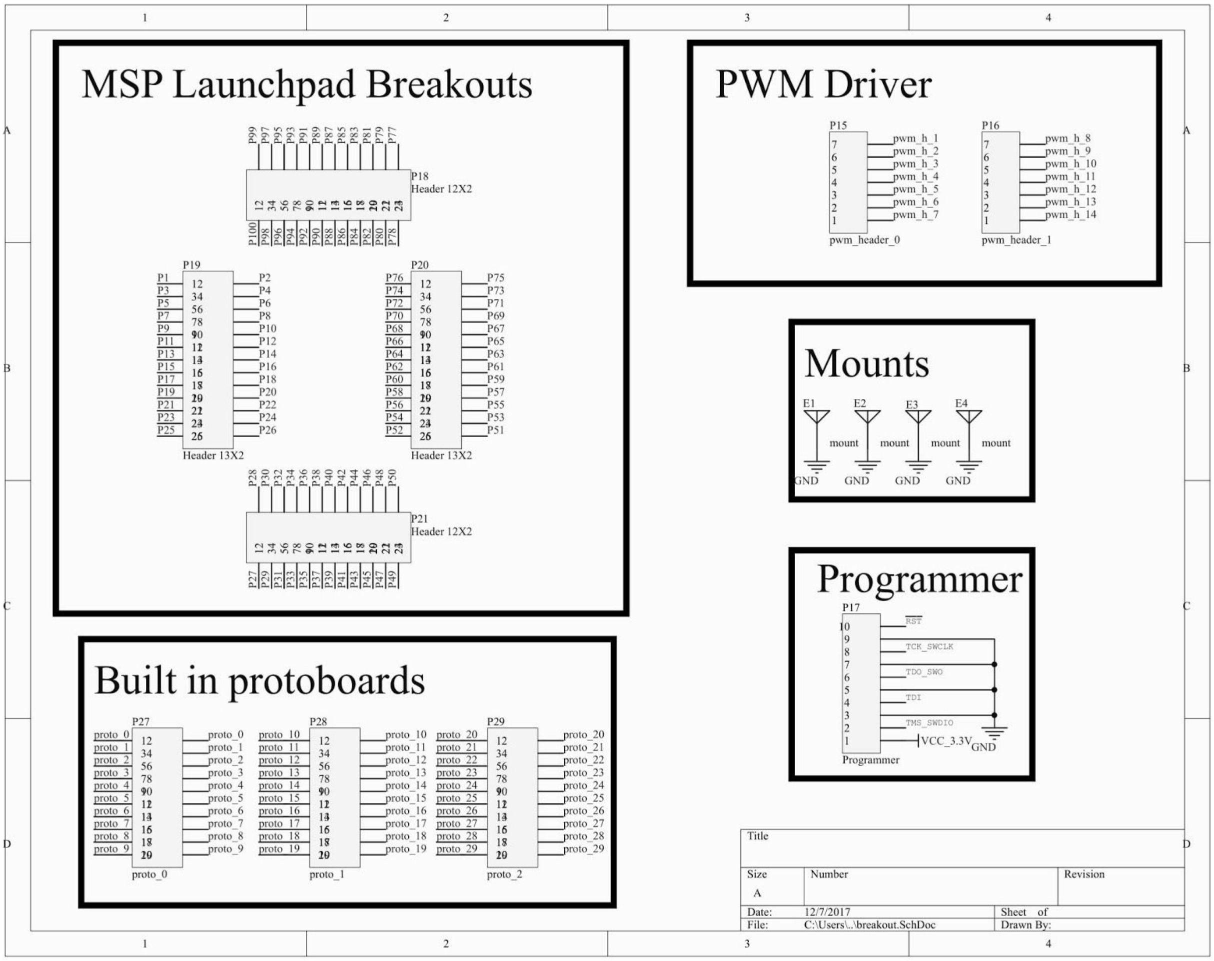
Full System Overview:



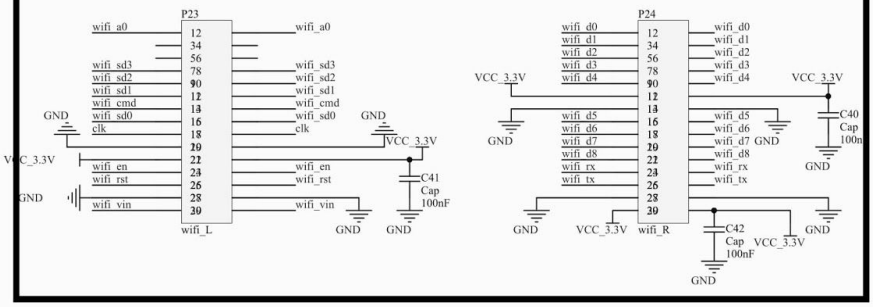
Schematics:

Main Board:

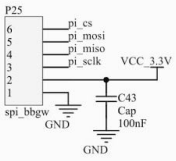



MSP Launchpad Breakouts
P18
Header 12X2
P19
Header 13X2
P20
Header 13X2
P21
Header 12X2
PWM Driver
P15
pwm_header_0
P16
pwm_header_1
pwm_h_1
pwm_h_2
pwm_h_3
pwm_h_4
pwm_h_5
pwm_h_6
pwm_h_7
pwm_h_8
pwm_h_9
pwm_h_10
pwm_h_11
pwm_h_12
pwm_h_13
pwm_h_14
Mounts
E1
E2
E3
E4
mount
GND
Programmer
P17
RST
TCK_SWCLK
TDO_SWO
TDI
TMS_SWDIO
VCC_3.3V
GND
Built in protoboards
P27
proto_0
P28
proto_1
P29
proto_2
Title
Size
A
Number
Revision
Date:
12/7/2017
Sheet of
File:
C:\Users\..\breakout.SchDoc
Drawn By:

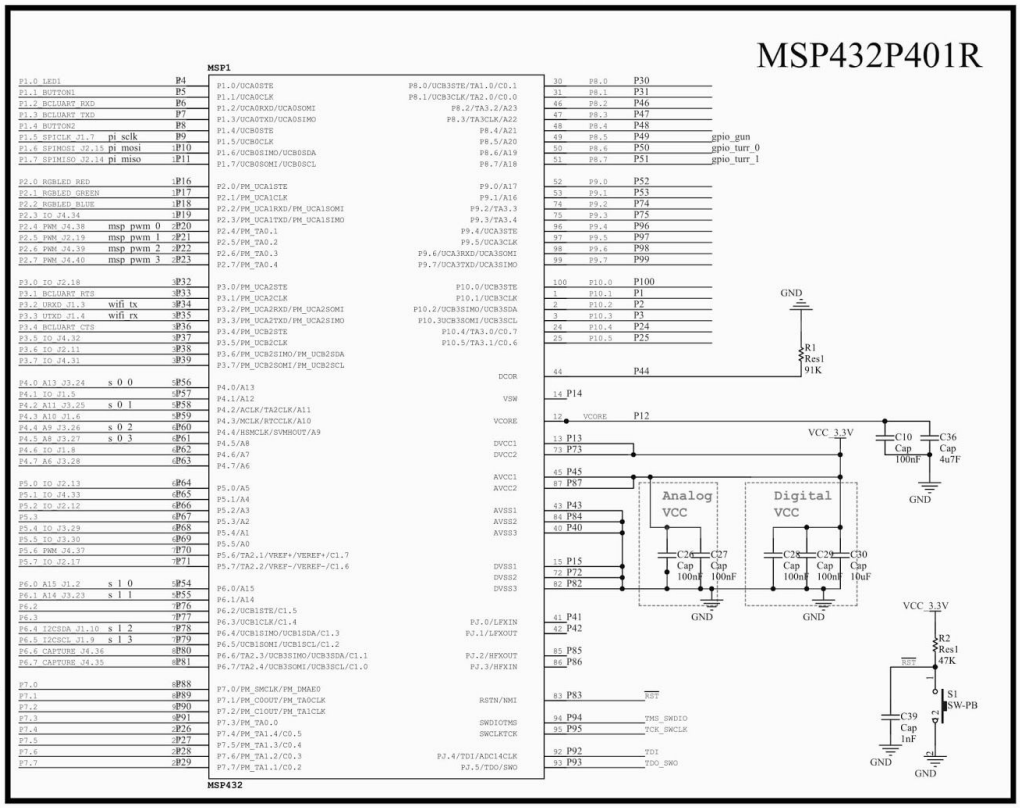
ESP8266



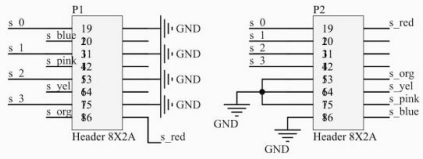
Raspberry Pi



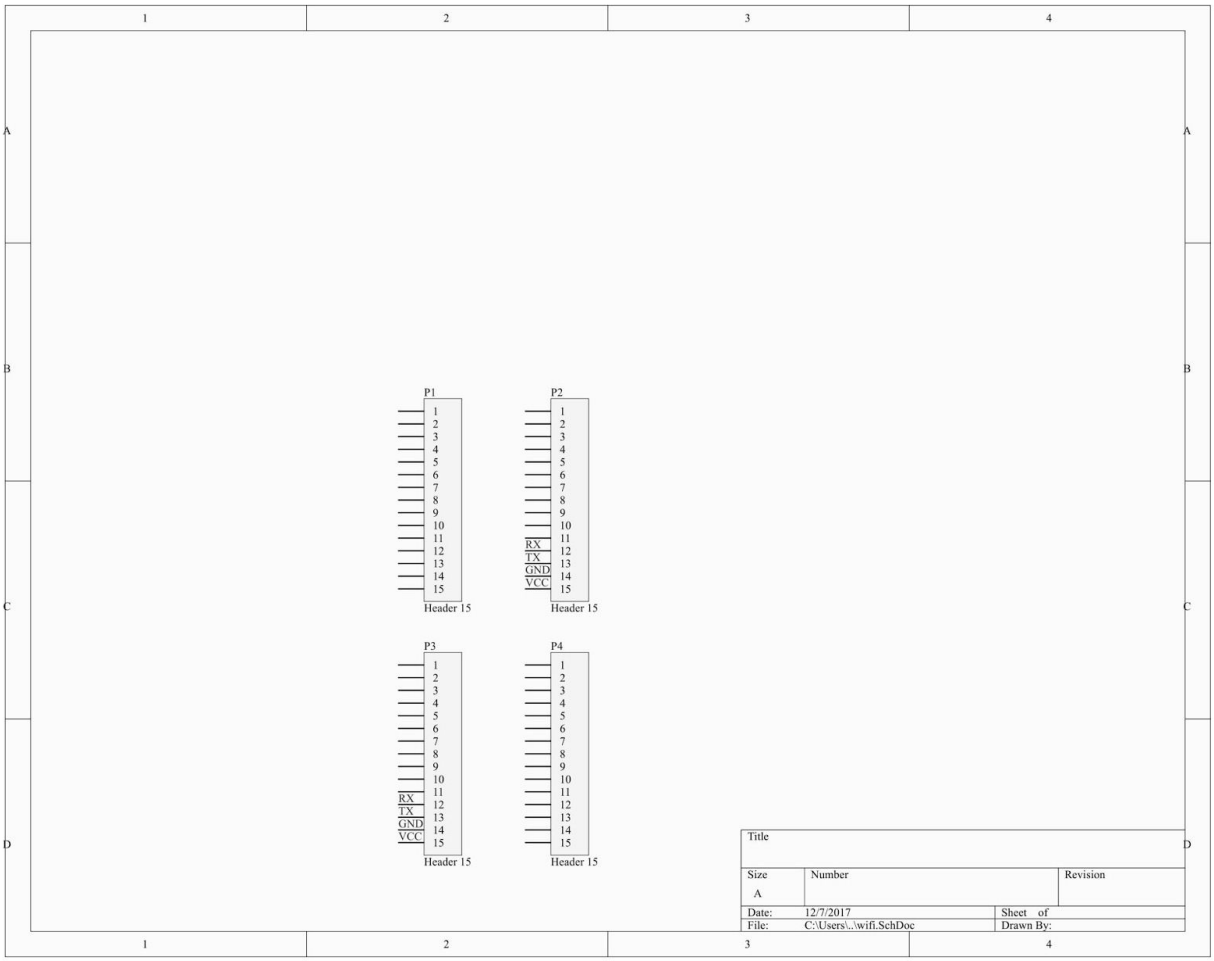
Title		
Size: A	Number	Revision
Date: 12/7/2017		Sheet of
File: C:\Users\..\comm.SchDoc		Drawn By:

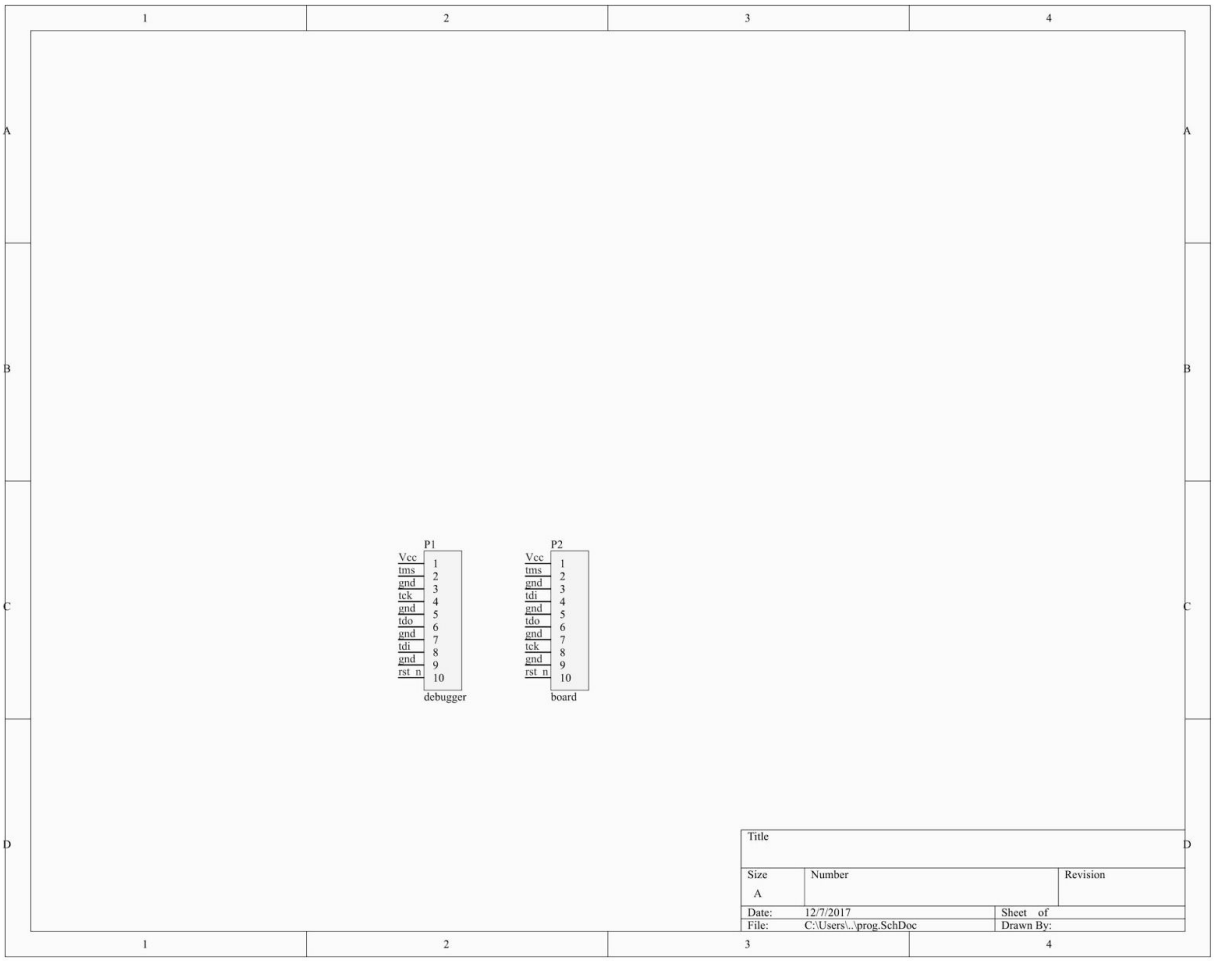


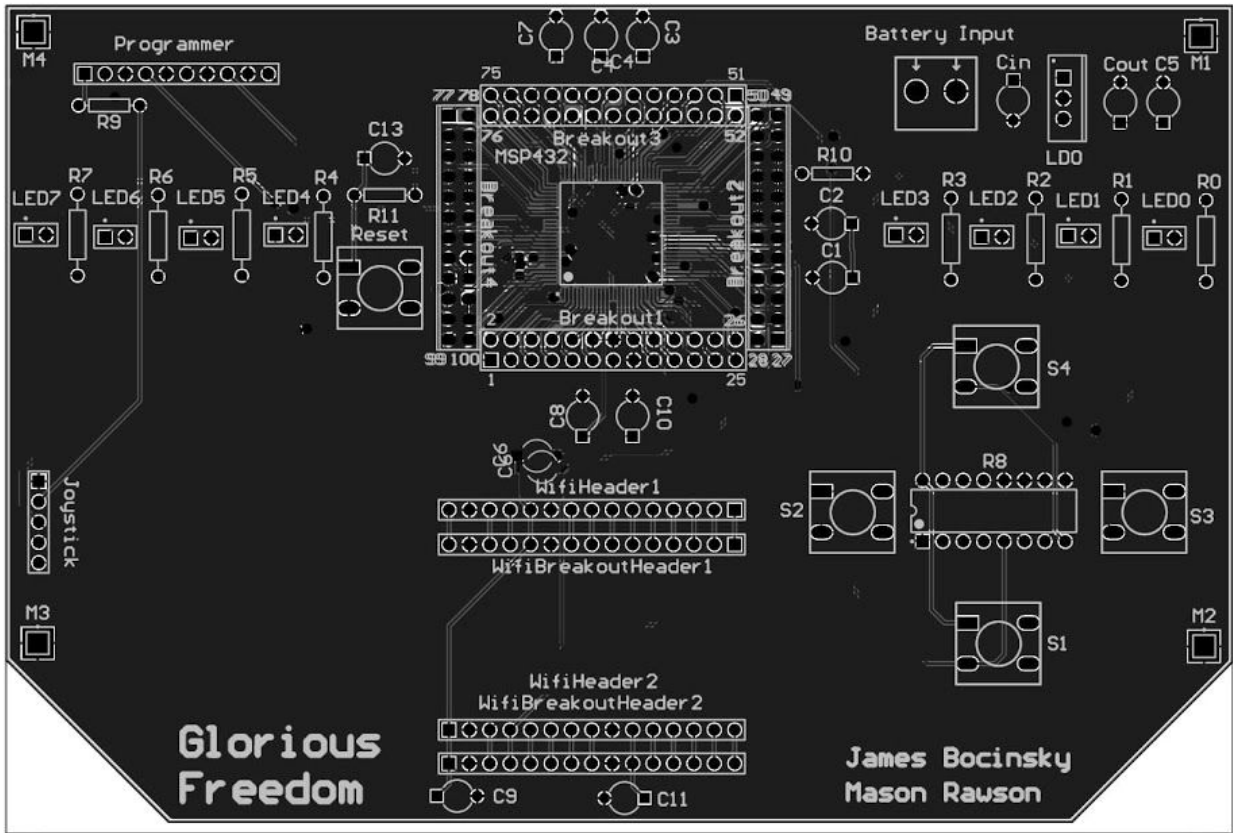
Title		
Size: Letter	Number	Revision
Date: 12/7/2017		Sheet of
File: C:\Users\..\logic.SchDoc		Drawn By:

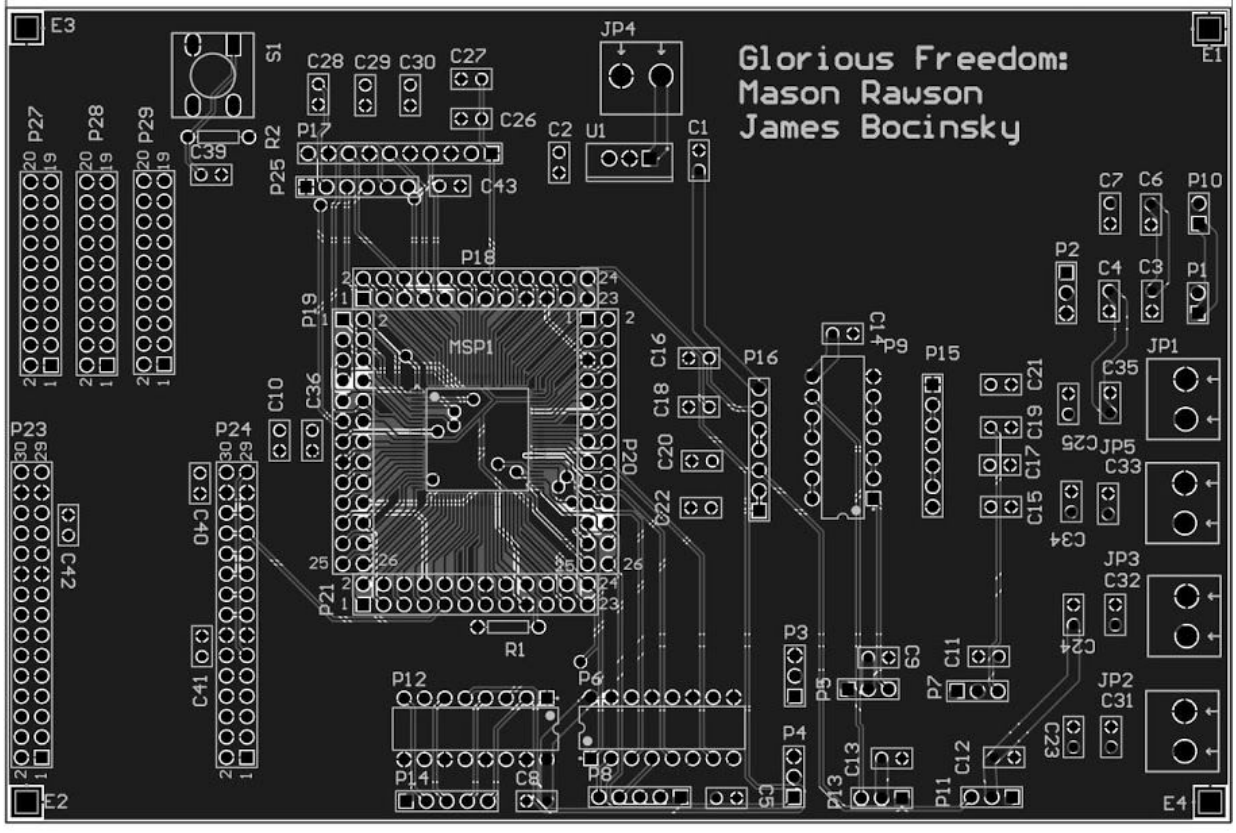


Title			
Size A	Number		Revision
Date:	12/7/2017	Sheet of	
File:	C:\Users\..\step.SchDoc	Drawn By:	

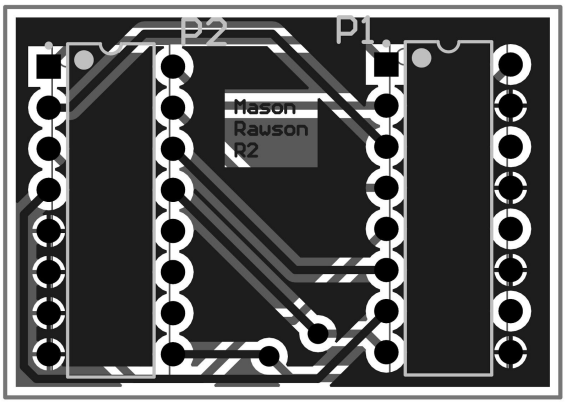

1
2
3
4
A
B
C
D
P1
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Header 15
P2
1
2
3
4
5
6
7
8
9
10
11
RX
12
TX
13
GND
14
VCC
15
Header 15
P3
1
2
3
4
5
6
7
8
9
10
11
RX
12
TX
13
GND
14
VCC
15
Header 15
P4
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Header 15
Title
Size
A
Number
Revision
Date:
12/7/2017
Sheet of
File:
C:\Users\..\wifi.SchDoc
Drawn By:

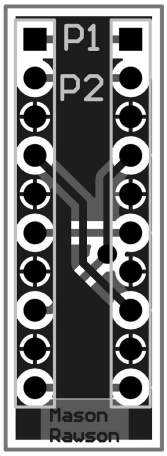

1
2
3
4
A
B
C
D
P1
Vcc 1
tms 2
gnd 3
tck 4
gnd 5
tdo 6
gnd 7
tdi 8
gnd 9
rst_n 10
debugger
P2
Vcc 1
tms 2
gnd 3
tdi 4
gnd 5
tdo 6
gnd 7
tck 8
gnd 9
rst_n 10
board
Title
Size
A
Number
Revision
Date: 12/7/2017
Sheet of
File: C:\Users\..\prog.SchDoc
Drawn By:


Programmer
Battery Input
Cin
Cout
LDO
Breakout3
MSP432
Breakout4
Breakout2
Breakout1
Reset
LED7
LED6
LED5
LED4
LED3
LED2
LED1
LED0
Joystick
WifiHeader1
WifiBreakoutHeader1
WifiHeader2
WifiBreakoutHeader2
Glorious
Freedom
James Bocinsky
Mason Rawson

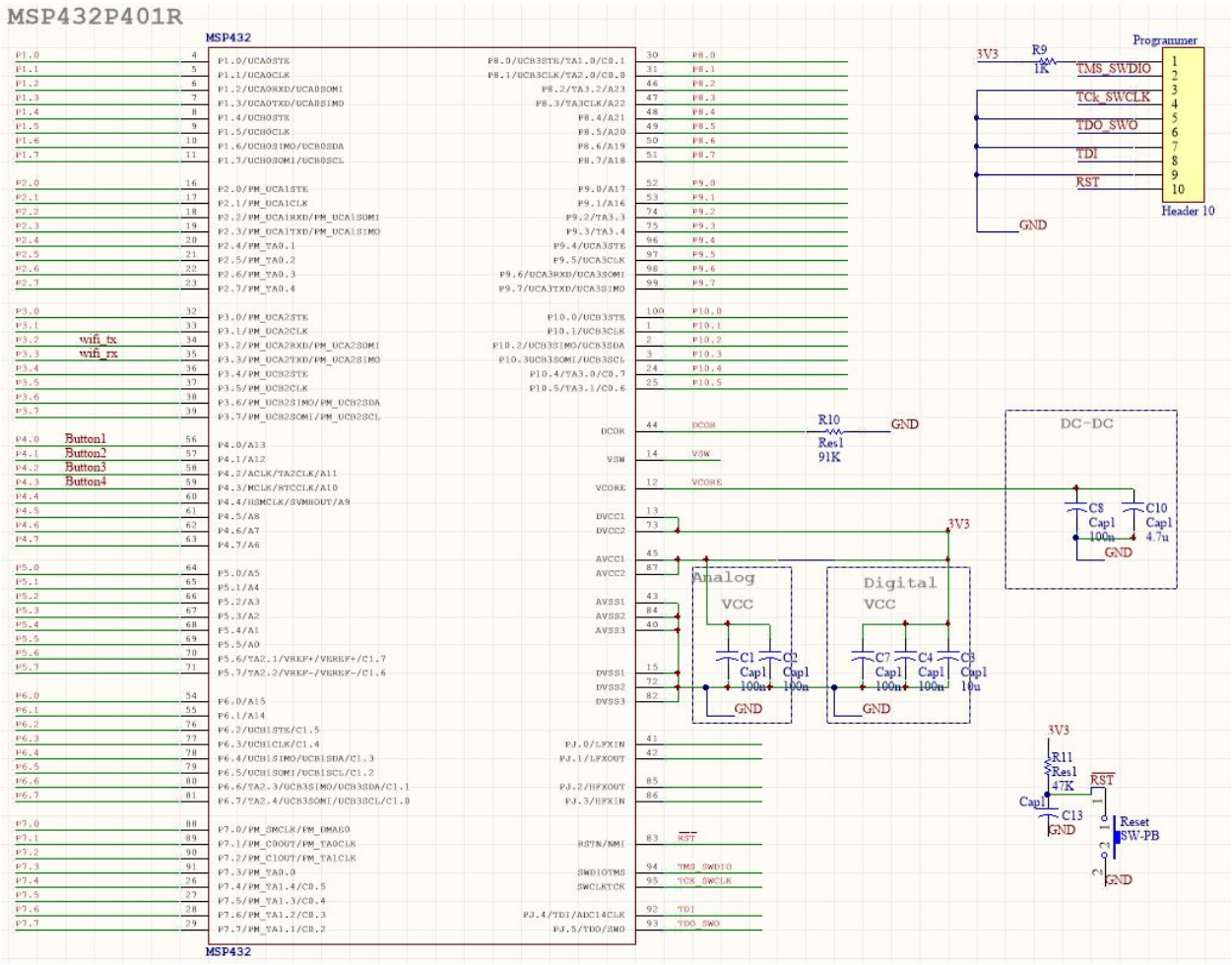

Glorious Freedom:
Mason Rawson
James Bocinsky
MSP1

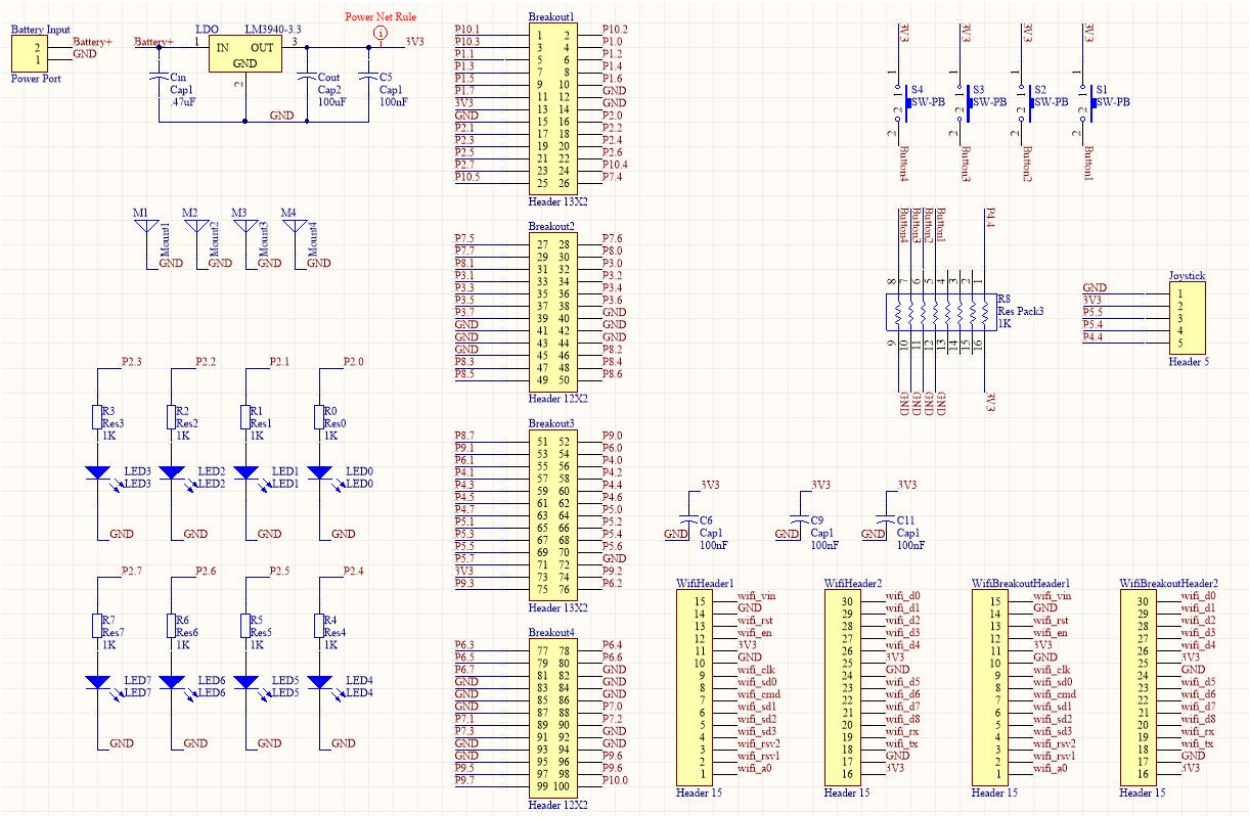

P3
P1
Glorious Freedom
Wifi Board
Mason Rawson
Rev 1
P4
P2


P2
P1
Mason
Rawson
R2


P1
P2
Mason
Rawson

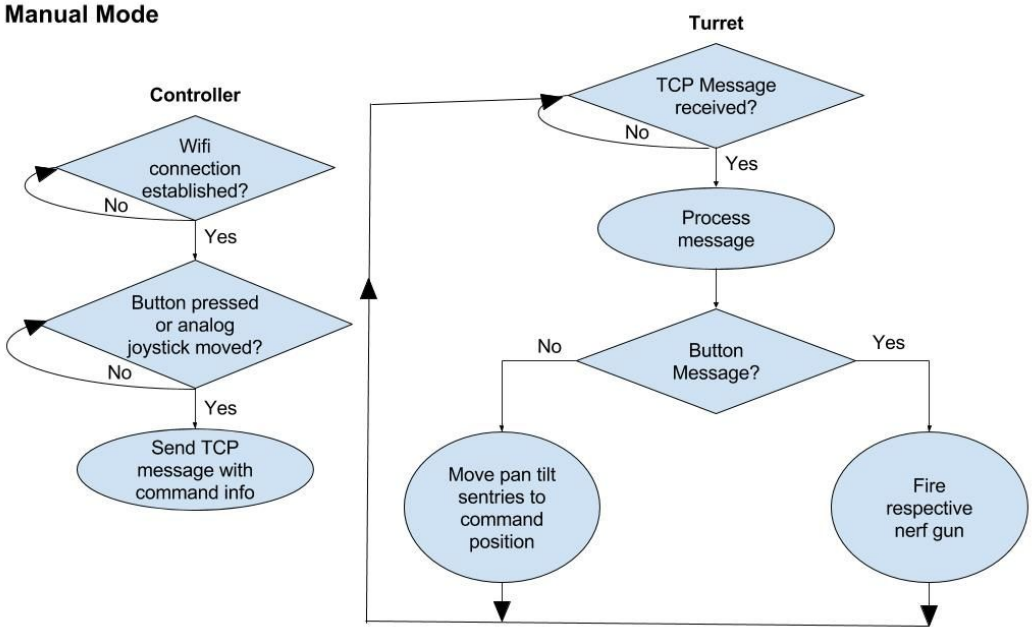
Controller:


MSP432P401R
MSP432
Programmer
Header 10
3V3
R9
1K
TMS_SWDIO
TCk_SWCLK
TDO_SWO
TDI
RST
GND
R10
Res1
91K
DC-DC
C8
C10
Cap1
100n
4.7u
Analog
VCC
Digital
VCC
C1
C2
C7
C4
C3
10u
R11
47K
C13
Reset
SW-PB
Button1
Button2
Button3
Button4
wifi_tx
wifi_rx

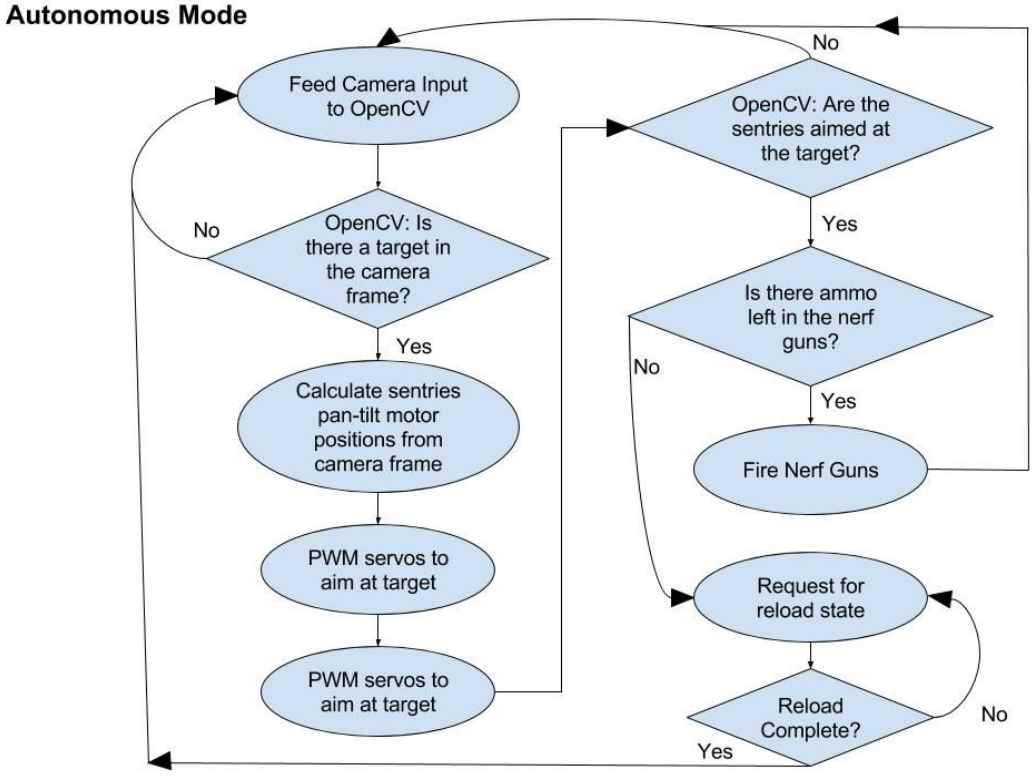

Battery Input
Battery+
GND
Power Port
LDO
LM3940-3.3
IN
OUT
GND
Power Net Rule
3V3
Cin
Cap1
47uF
Cout
Cap2
100nF
C5
Cap1
100nF
Breakout1
Header 13X2
Breakout2
Header 12X2
Breakout3
Header 13X2
Breakout4
Header 12X2
S4
S3
S2
S1
SW-PB
Button4
Button3
Button2
Button1
M1
M2
M3
M4
Mount1
Mount2
Mount3
Mount4
P2.3
P2.2
P2.1
P2.0
R3
R2
R1
R0
Res3
Res2
Res1
Res0
1K
LED3
LED2
LED1
LED0
P2.7
P2.6
P2.5
P2.4
R7
R6
R5
R4
Res7
Res6
Res5
Res4
LED7
LED6
LED5
LED4
P4.4
R8
Res Pack3
Joystick
Header 5
C6
C9
C11
Cap1
100nF
WifiHeader1
WifiHeader2
WifiBreakoutHeader1
WifiBreakoutHeader2
Header 15
wifi_vin
GND
wifi_rst
wifi_en
3V3
wifi_clk
wifi_sd0
wifi_cmd
wifi_sd1
wifi_sd2
wifi_sd3
wifi_rsv2
wifi_rsv1
wifi_a0
wifi_d0
wifi_d1
wifi_d2
wifi_d3
wifi_d4
wifi_d5
wifi_d6
wifi_d7
wifi_d8
wifi_rx
wifi_tx

Software FlowCharts:

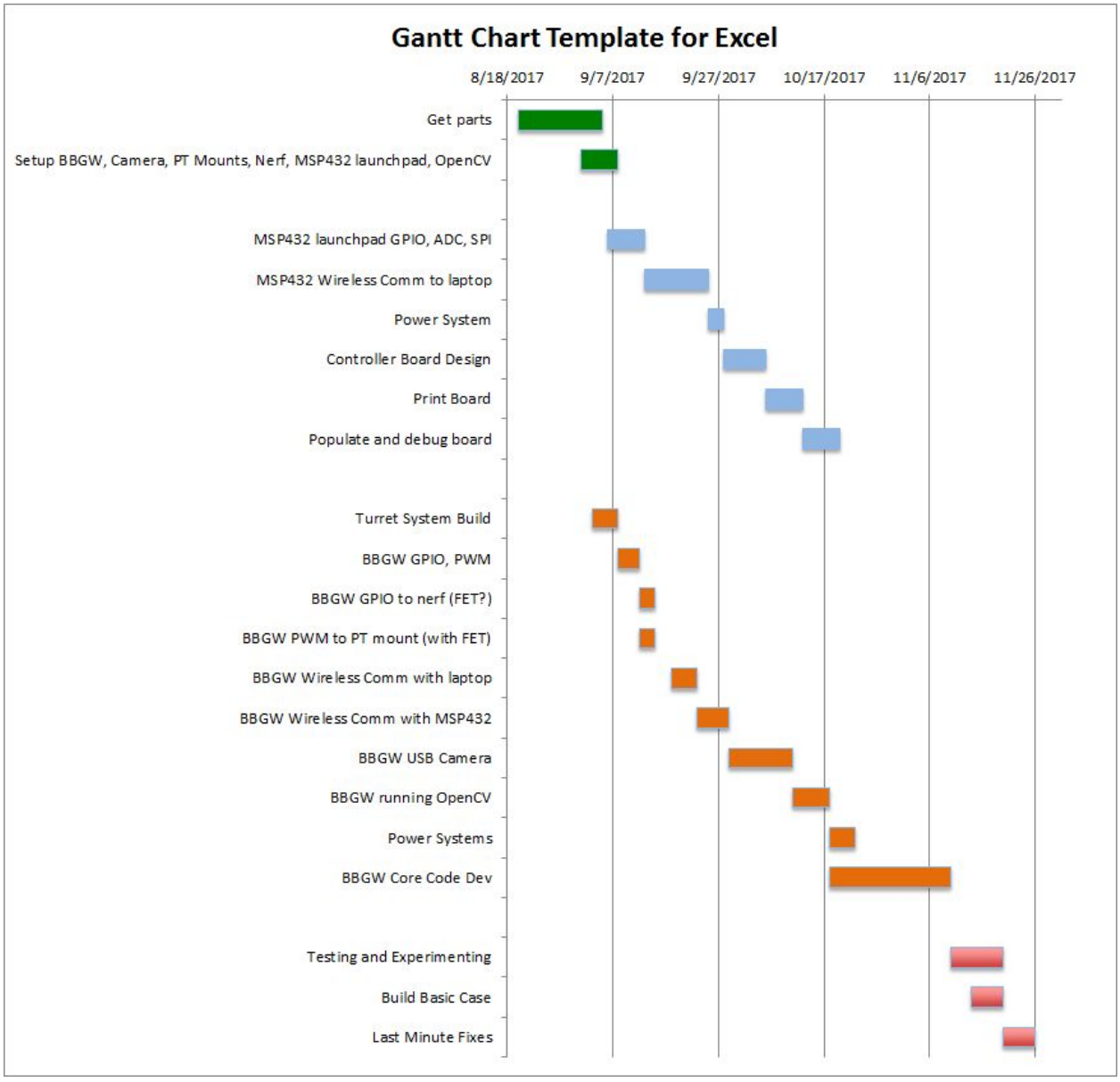
Manual Mode



Autonomous Mode



Gantt Chart:


Gantt Chart Template for Excel
8/18/2017
9/7/2017
9/27/2017
10/17/2017
11/6/2017
11/26/2017
Get parts
Setup BBGW, Camera, PT Mounts, Nerf, MSP432 launchpad, OpenCV
MSP432 launchpad GPIO, ADC, SPI
MSP432 Wireless Comm to laptop
Power System
Controller Board Design
Print Board
Populate and debug board
Turret System Build
BBGW GPIO, PWM
BBGW GPIO to nerf (FET?)
BBGW PWM to PT mount (with FET)
BBGW Wireless Comm with laptop
BBGW Wireless Comm with MSP432
BBGW USB Camera
BBGW running OpenCV
Power Systems
BBGW Core Code Dev
Testing and Experimenting
Build Basic Case
Last Minute Fixes

Individual Responsibilities & Gantt Chart Breakdown:

Assignments: J = James, M = Mason, S = Shared

Task Name	Start	End	Duration (days)	Assignment
Get parts	8/20/2017	9/5/2017	16	S
Setup BBGW, Camera, PT Mounts, Nerf, MSP432 launchpad, OpenCV	9/1/2017	9/8/2017	7	S
MSP432 launchpad GPIO, ADC, SPI	9/6/2017	9/13/2017	7	J
MSP432 Wireless Comm to laptop	9/13/2017	9/25/2017	12	J
Power System	9/25/2017	9/28/2017	3	J
Controller Board Design	9/28/2017	10/6/2017	8	S
Print Board	10/6/2017	10/13/2017	7	J
Populate and debug board	10/13/2017	10/20/2017	7	S
Turret System Build	9/3/2017	9/8/2017	5	S
BBGW GPIO, PWM	9/8/2017	9/12/2017	4	M
BBGW GPIO to nerf (FET?)	9/12/2017	9/15/2017	3	J
BBGW PWM to PT mount (with FET)	9/12/2017	9/15/2017	3	M
BBGW Wireless Comm with laptop	9/18/2017	9/23/2017	5	M
BBGW Wireless Comm with MSP432	9/23/2017	9/29/2017	6	S
BBGW USB Camera	9/29/2017	10/11/2017	12	M
BBGW running OpenCV	10/11/2017	10/18/2017	7	S
Power Systems	10/18/2017	10/23/2017	5	M
BBGW Core Code Dev	10/18/2017	11/10/2017	23	S
Testing and Experimenting	11/10/2017	11/20/2017	10	S
Build Basic Case	11/14/2017	11/20/2017	6	S
Last Minute Fixes	11/20/2017	11/26/2017	6	S

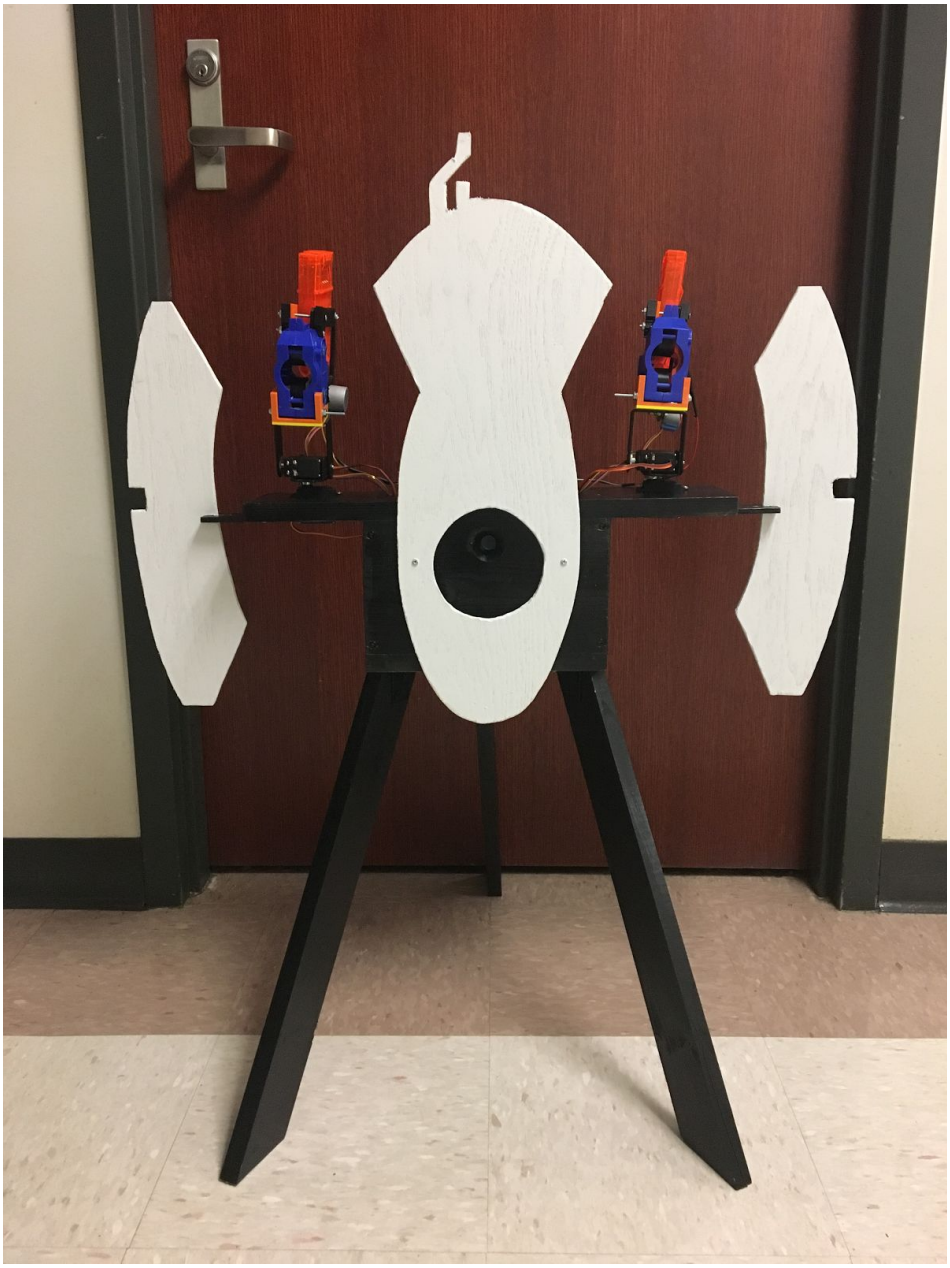
Engineering Solutions:

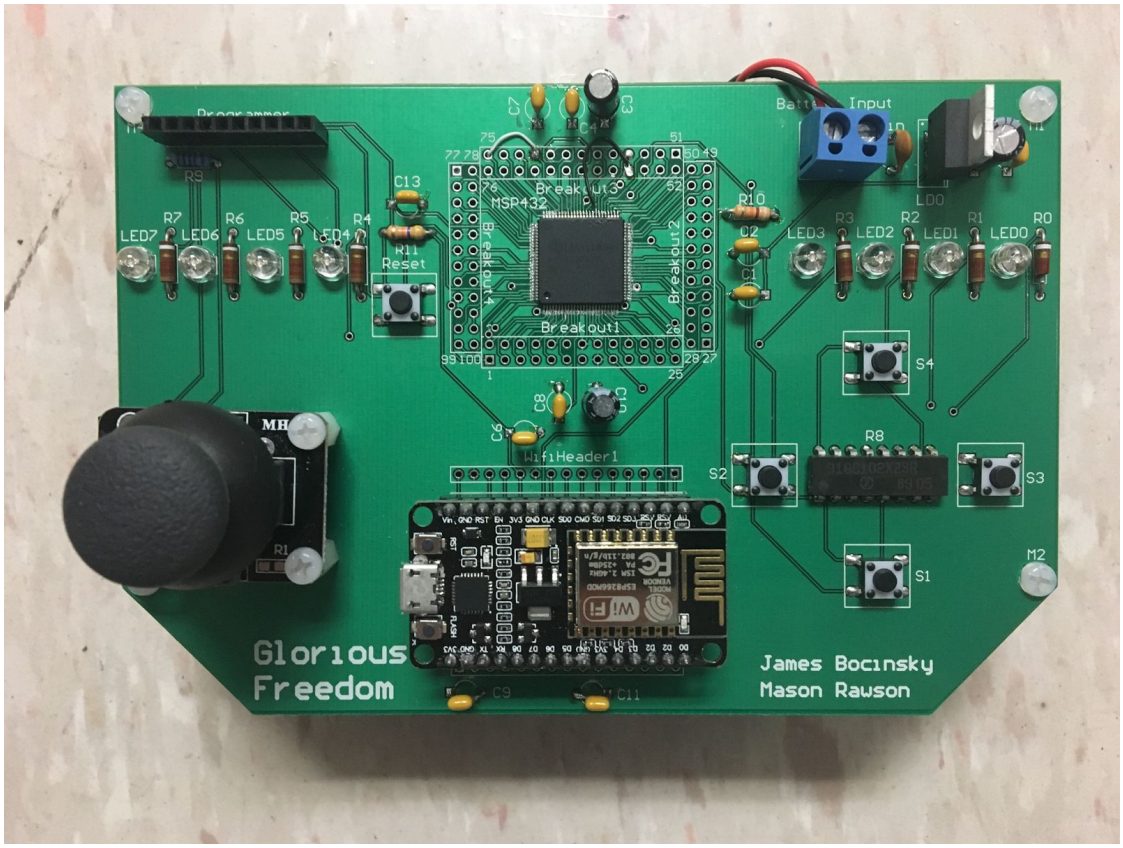
We needed to overcome many, many issues to be able to achieve our goal. I will highlight some of the most important ones here. The absolute worst error occurred at the end when we put everything together. The SPI communication from the Pi to the MSP would work for a random amount of time, and then turn to garbage. When it would become garbage was impossible to predict and seemingly impossible to recover from without a hardware reset. We later found that we could recover by shutting down the SPI peripheral and giving enough time delay before trying to re-initialize the system. We had never seen this before placing all parts inside our actual turret mount. We theorize that it is noise from the high power lines which run to the turret motors.

Another major problem was getting OpenCV to work on the Beaglebone. Unfortunately, the binaries which OpenCV relies on are incompatible with the Beaglebone. These error likely came from errors in cross compiling and simply went unnoticed in the Debian distribution since not many people run OpenCV on the beaglebone. Rather than investing significant effort in making it work on the beaglebone, we switched to a modern Raspberry Pi where OpenCV was fully supported. The major problems came with breadboard prototyping. High current spikes are very common in our design. They can happen so fast that when using a power supply, it is not obvious that a current limit was reached. This problem would lead to many many errors such as turning off the PWM drivers which would cause the servos to go haywire in addition to making the stepper motors simply not work. Additionally there was large amounts noise from the stepper wires running close to each other on the breadboard which had to be carefully bypassed out with capacitors.

Lastly we had various implementation details which lead to problems. For instance, the stepper motors didn't have enough torque to push a bullet in the default bullet clip, so we removed the spring. The buck converter we chose for the turret motors would hit its current limit and magically act like a capacitor which was hard to identify. We now use a much larger buck converter. We melted traces, once on our power distribution board and twice on our stepper motor breakout boards. Two servo motors simply stopped working last minute. And finally, the main board had either routing errors or layout errors on most subsystems. The wifi chip pinouts were flipped. The wrong layout for the stepper motor drivers was used. We switched from using a chip to using fets for the PWM driver since multiple traces didn't route. The programmer had two pins flipped because I followed outdated documentation. These and other issues plagued the project, but with a lot of hard work we were able to solve all of these problems.

Pictures:




Glorious
Freedom
James Bocinsky
Mason Rawson
MSP432
Breakout1
Breakout2
Breakout3
WifiHeader1
Input
LED0
LED1
LED2
LED3
LED4
LED5
LED6
LED7
Reset
S1
S2
S3
S4
M1
M2


Glorious
Freedom
OFF ON