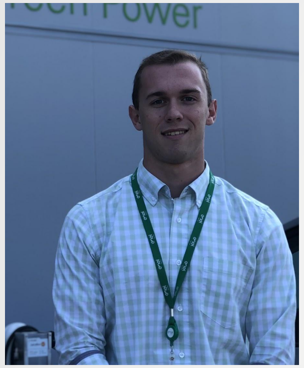
GyroGlove
Midway Design Review

Senior Design Project ‘21
Team #21

Meet the team



Bradley Spillert
CSE

Jacob Moynihan
EE

Son Pham
EE

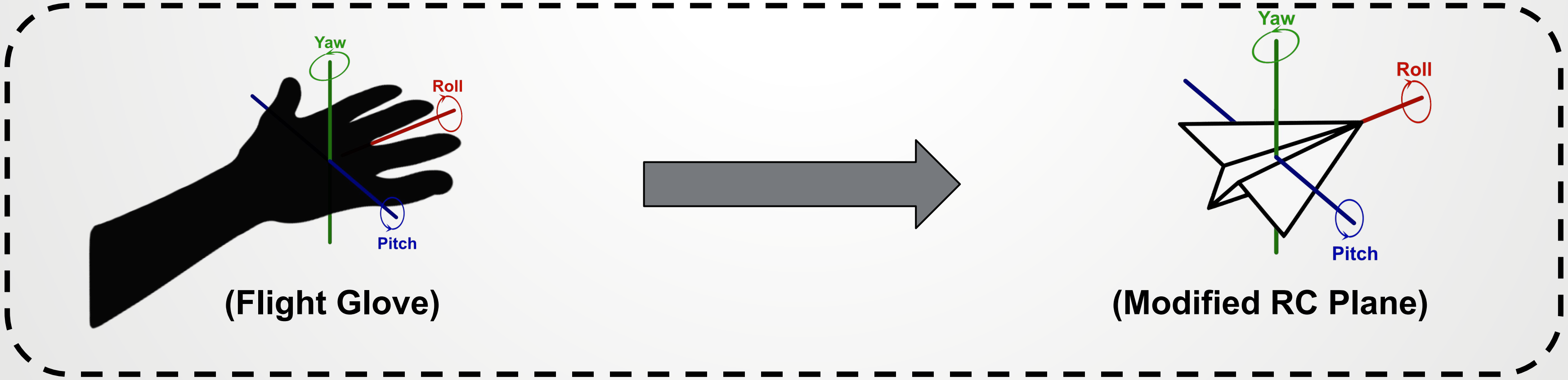
Professor Do-Hoon Kwon
Team Advisor

Problem Statement

Having a reliable flight control system is paramount for the success of any manned aircraft. While the current convention of using control mechanisms akin to analog sticks and steering wheels has proven to be timelessly effective, such mechanisms can often lack one valuable prospect: The immersion factor.

Problem Statement (cont'd)

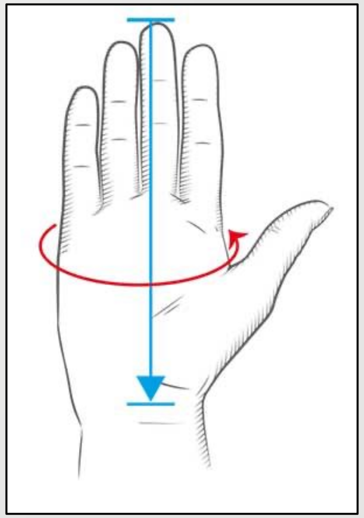
This is where GyroGlove comes into play. GyroGlove is a modern alternative to the classical stick-lever-wheel approach to flight control, ultimately allowing the pilot to control the aircraft simply by rotating their hand (orientation control) and curling their fingers (thrust control). These systems work elegantly in tandem to provide an immersive sensation of being "one with the aircraft", so to speak.



GyroGlove System

GyroGlove - System Specifications

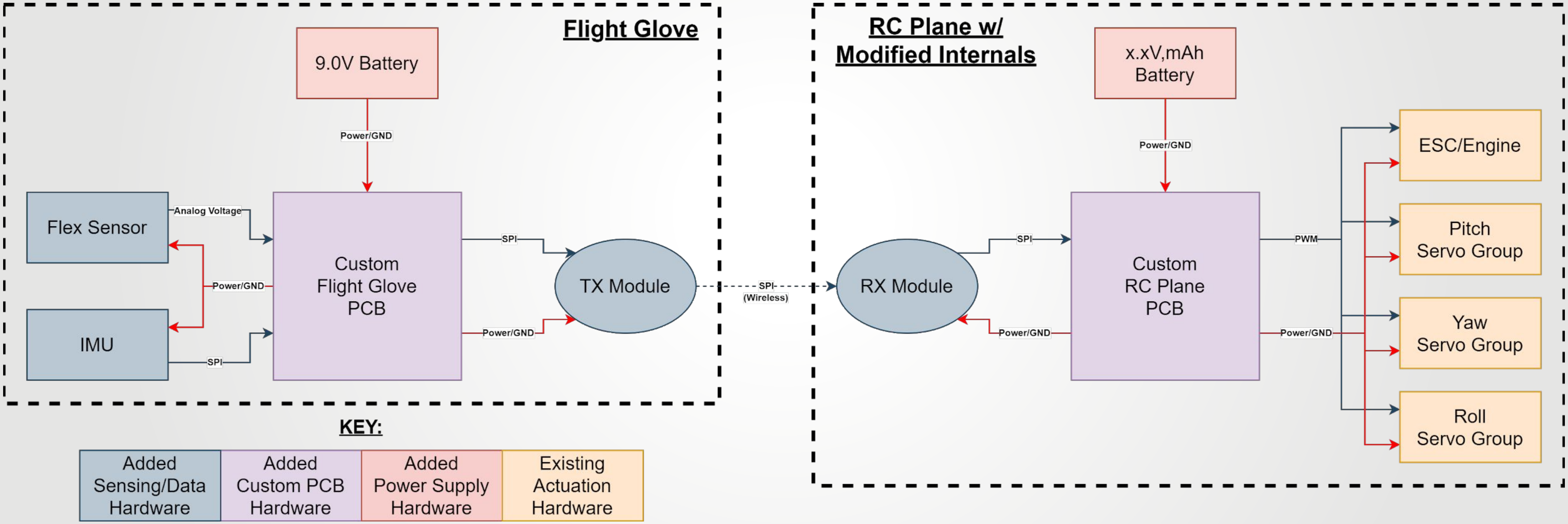
- Flight glove battery life exceeds RC plane battery life by at least 100%
- Aircraft rotation stays within ±5.0° of IMU rotation (±7.5° for yaw axis)
- IMU capable of detecting rotation between ±85.0° along each axis
- Flex sensor capable of generating dynamic output over 90.0° flex range
- Functional range of at least 100 meters
- Control latency of 20 milliseconds or less within the functional range
- Operating frequency of 2.4GHz
- Compliance with all FCC standards and regulations



Subsystem Overview

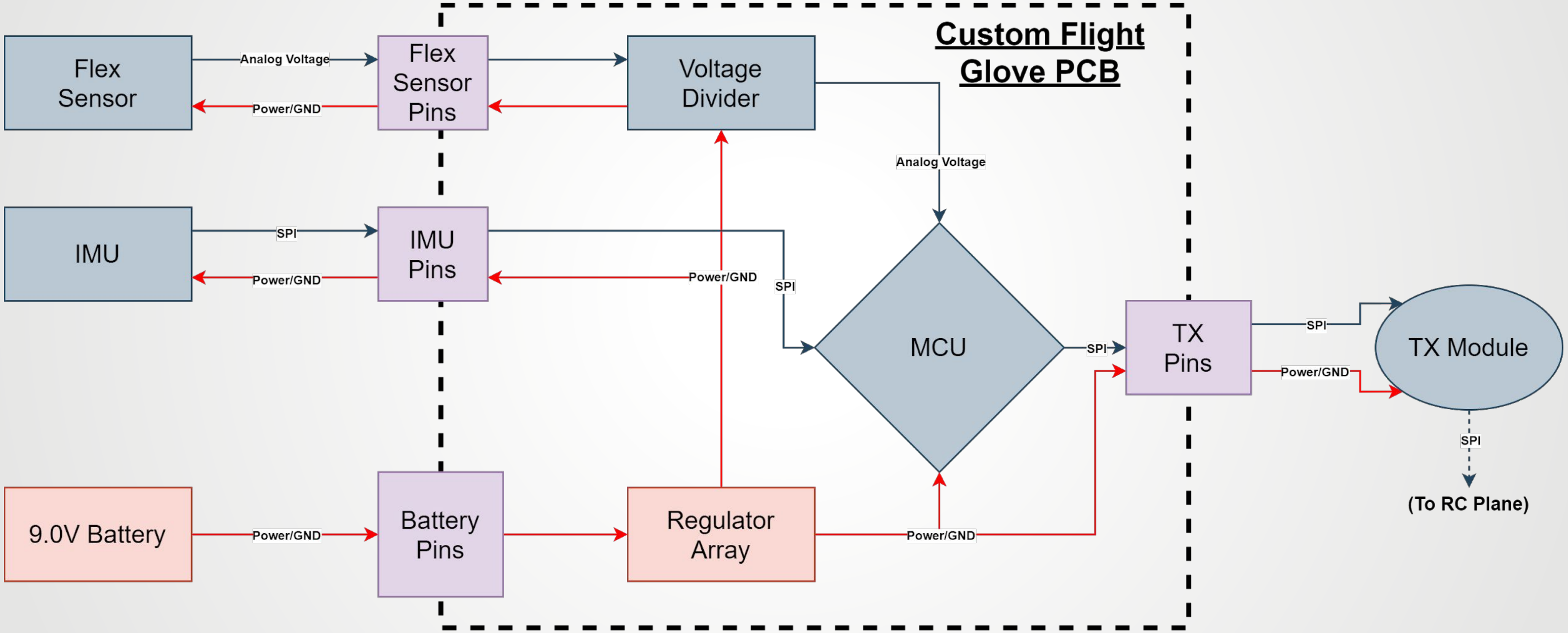
1. **Flight Glove: Captures/interprets movement of the pilot's hand and encodes data for TX to RC plane**
   1.1. Rotational movement capturing: Using hand-mounted IMU
   1.2. Thrust control capturing: Using 3" flex sensor mounted along middle finger
2. **RC Plane: Receives encoded actuation data from Flight Glove and animates existing RC plane actuation hardware**
   2.1. Pitch/yaw/roll servo groups: Control the RC plane's rotational orientation, as defined by glove IMU movement
   2.2. Engine/ESC: Controls the RC plane thrust, as defined by glove flex sensor tension

Hardware Block Diagram (Overview)



*For MDR, Arduino Nanos are being used in place of both custom PCBs.

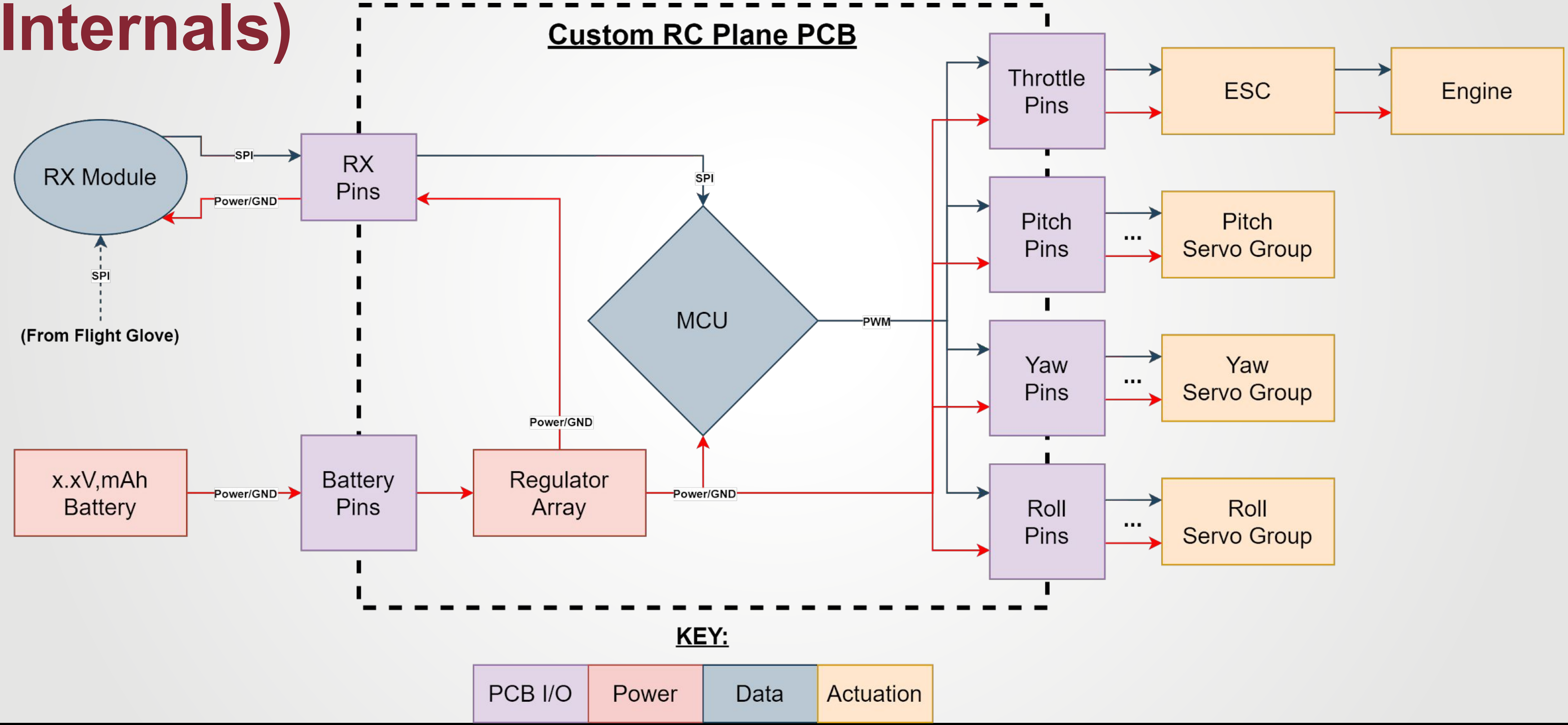
Hardware Block Diagram (Flight Glove)



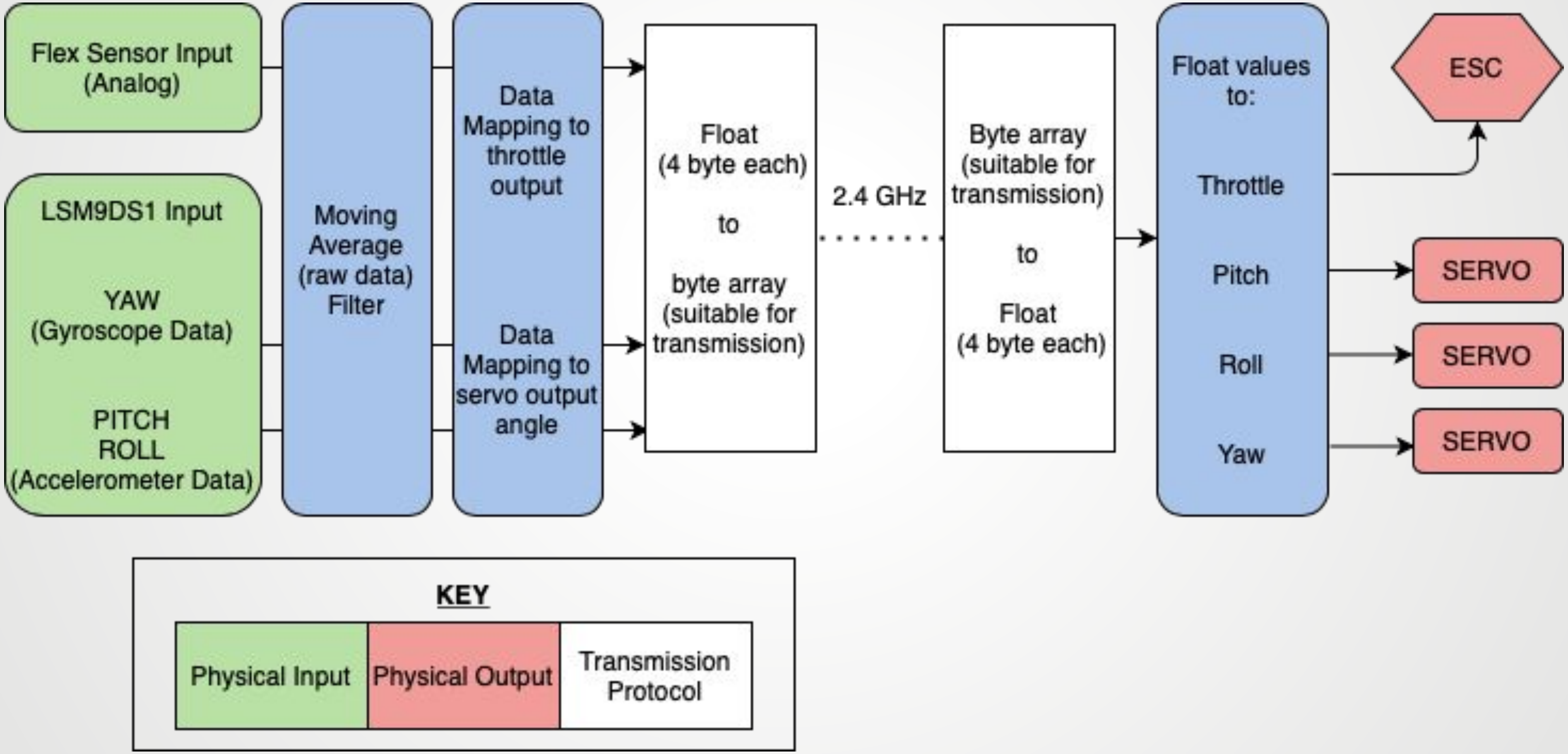
KEY:

PCB I/O	Power	Data

Hardware Block Diagram (RC Plane w/ Modified Internals)

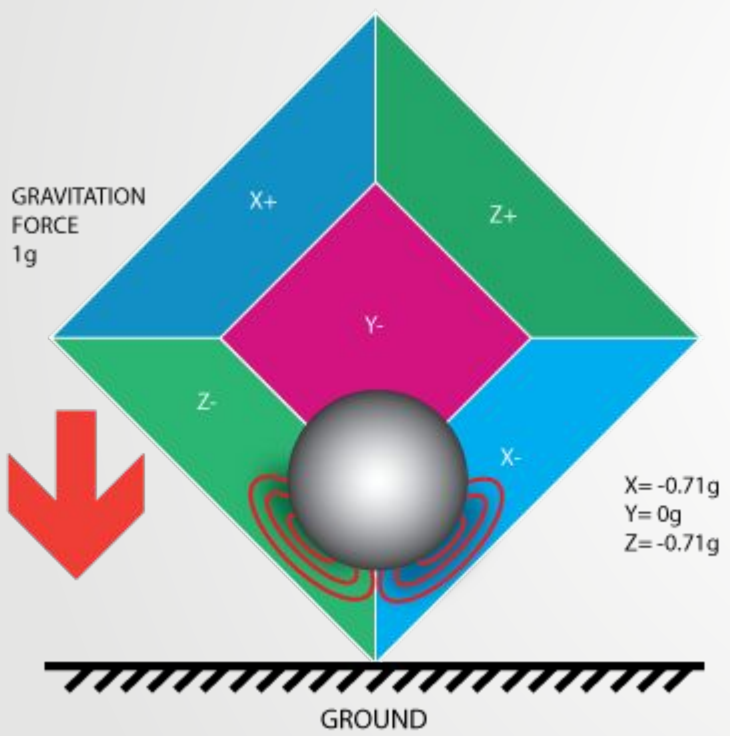


Software Block Diagram

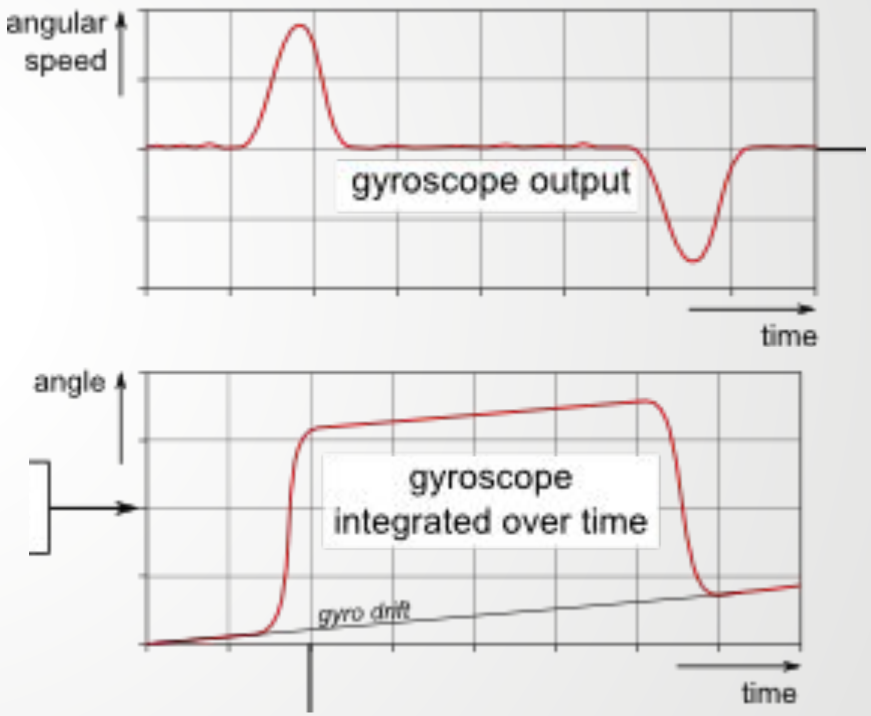


Accelerometer vs. Gyroscope

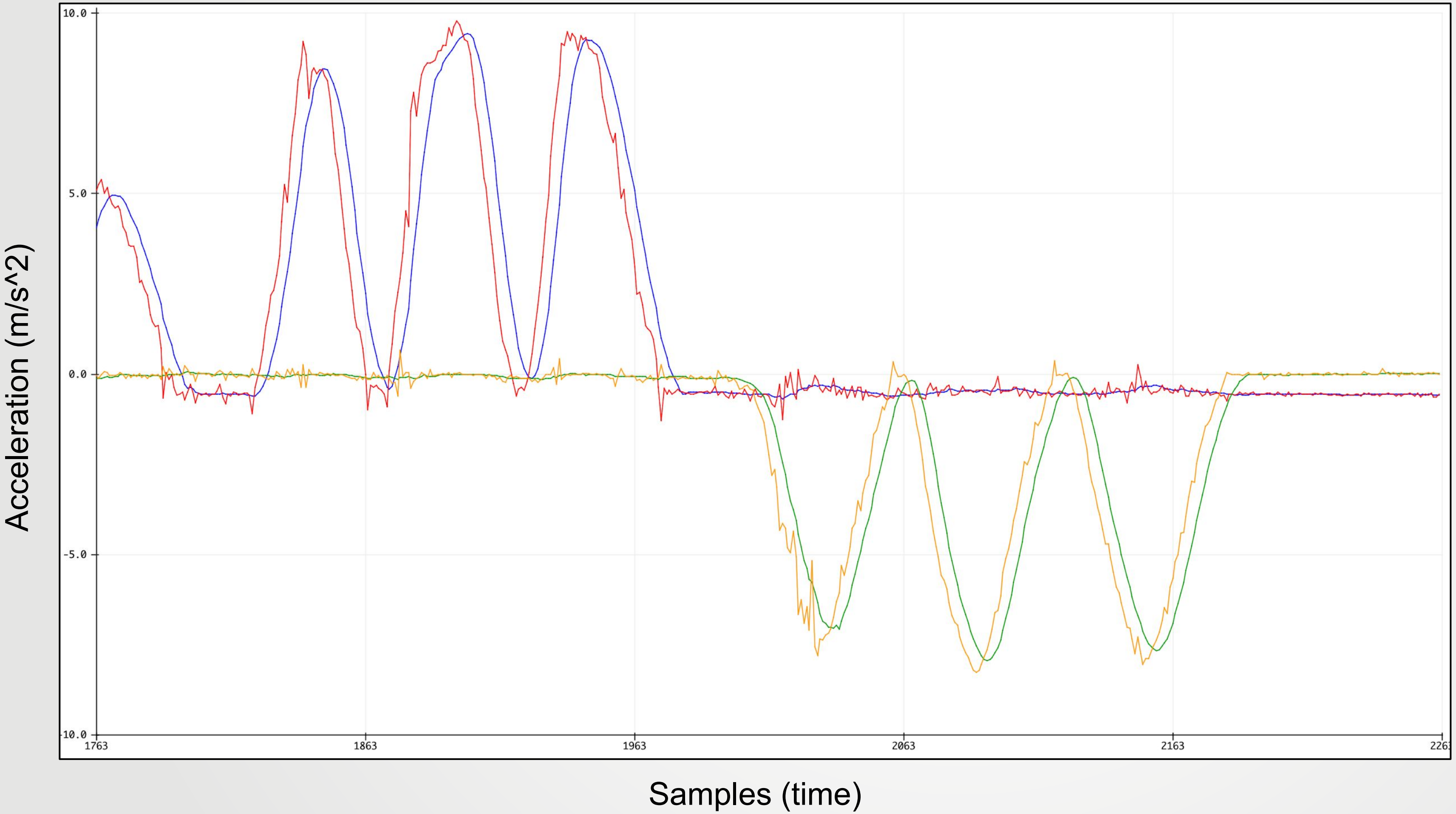
Accelerometer (X,Y)



Gyro (Integration) (Z)



Rolling Average Filter

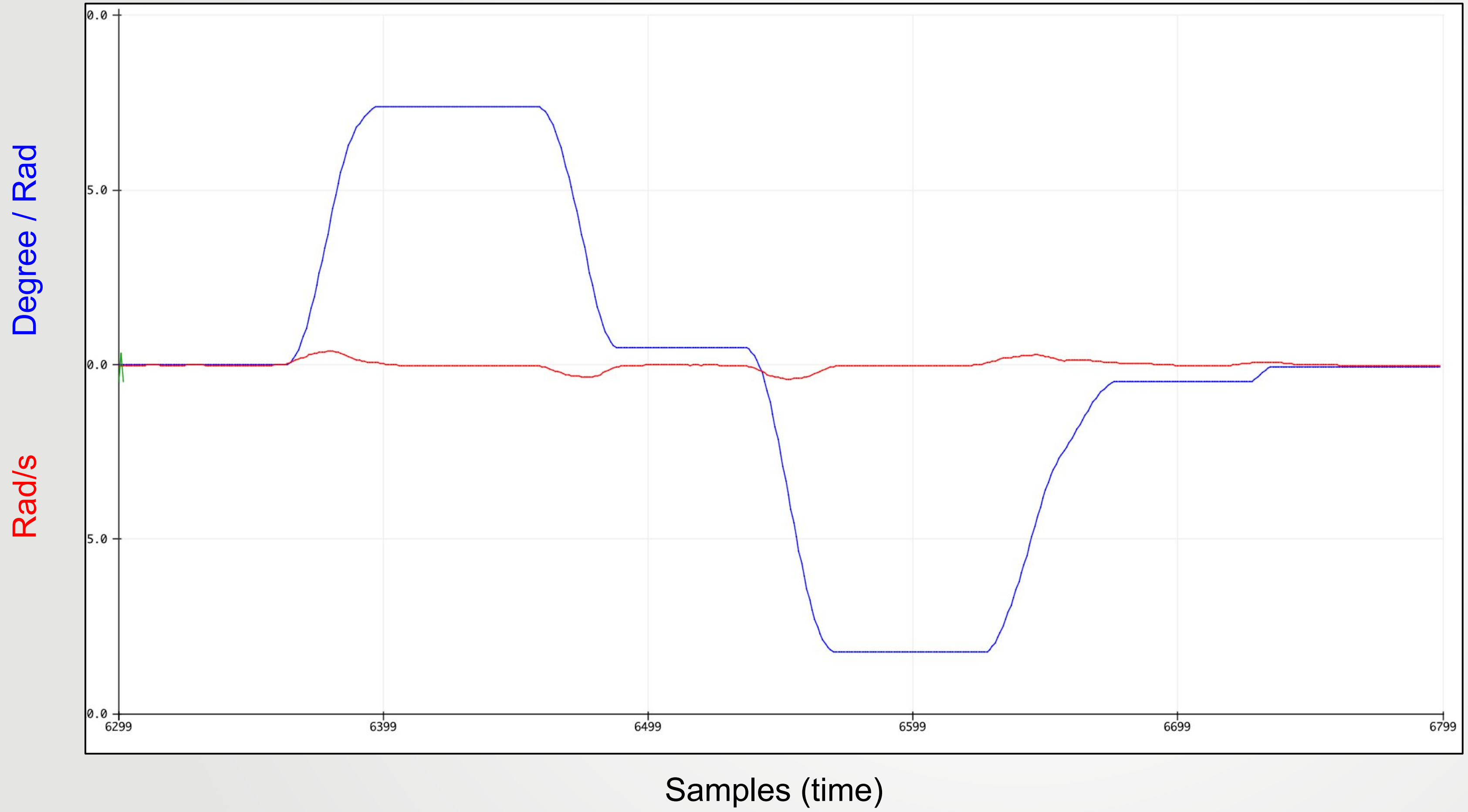


KEY
PITCH Before filter
PITCH After filter

ROLL Before filter
ROLL After filter

* Deliberate pitch change followed by deliberate roll change *

Yaw Integration



KEY

GYRO Reading

YAW After integration

* Deliberate YAW change *
Gyro data has rolling average filter applied

Kalman Filtering and Calibration Method

```
// Calculate Pitch & Roll from accelerometer (deg)
accPitch = -(atan2(acc.XAxis, sqrt(acc.YAxis*acc.YAxis + acc.ZAxis*acc.ZAxis))*180.0)/M_PI;
accRoll  = (atan2(acc.YAxis, acc.ZAxis)*180.0)/M_PI;

// Kalman filter
kalPitch = kalmanY.update(accPitch, gyr.YAxis);
kalRoll = kalmanX.update(accRoll, gyr.XAxis);

Serial.print(accPitch);
Serial.print(":");
Serial.print(accRoll);
Serial.print(":");
Serial.print(kalPitch);
Serial.print(":");
Serial.print(kalRoll);
Serial.print(":");
Serial.print(acc.XAxis);
Serial.print(":");
Serial.print(acc.YAxis);
Serial.print(":");
Serial.print(acc.ZAxis);
Serial.print(":");
Serial.print(gyr.XAxis);
Serial.print(":");
Serial.print(gyr.YAxis);
Serial.print(":");
Serial.print(gyr.ZAxis);

Serial.println();
```

```
///////////////////////////////////   LOOP   ///////////////////////////////////
void loop() {
  if (state==0){
    Serial.println("\nReading sensors for first time...");
    meansensors();
    state++;
    delay(1000);
  }

  if (state==1) {
    Serial.println("\nCalculating offsets...");
    calibration();
    state++;
    delay(1000);
  }

  if (state==2) {
    meansensors();
```

List of Hardware/Software (To Date)

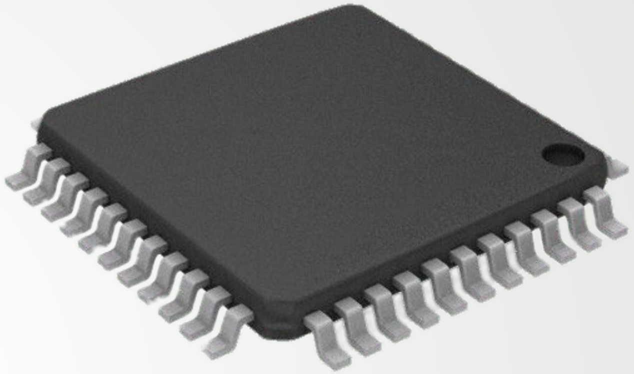
Hardware

- Arduino Nano (x2)
- Arduino Uno [retired]
- NRF24L01+PA+LNA RF Module (x2)
- LSM9DS1 IMU Board
- GY-521 IMU Board [retired]
- SG-90 Servo Motor (x3)
- 3” Flex Sensor
- 10kohm resistor
- LED (placeholder for engine)
- 200 ohm resistor (for LED)
- Nitrile ESD Glove
- Breadboard Power Supply Module
- Breadboard, Jumper Wires

Software

- Arduino Uno : 1st iteration of main code [retired]
- Arduino Nano : rehauled main code, receiver & transmitter integration

Hardware Plan for FPR



Custom PCB MCUs: ATmega328

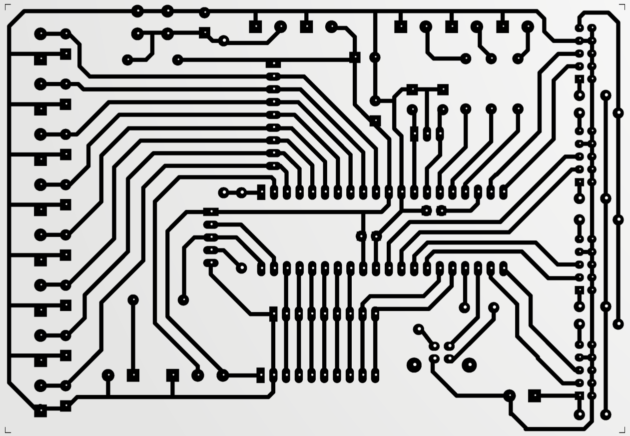
IMU Sensor: LSM9DS1 Board

Flex Sensor: 3” (~25kΩ)

TX/RX: NRF24L01 Transceivers

Power: 9.0V Battery (Flight Glove) and Unknown Battery* (RC Plane)

Actuators: RC plane’s existing servos/ESC/engine



*We may have to swap out the RC plane’s stock battery in order to power our custom PCB *and* the plane’s motors/engine/etc.

Rough Cost Estimate (To Date)

- Arduino Nano (2pcs) - $10.98
- Nitrile Work Glove FN330 - $12.49
- GY-521 Module - $4.99
- NRF24L01+PA+LNA RF Module (2pcs) - $9.99
- LSM9DS1 IMU Board - $15.95
- Jumper wires and resistors pack - $ 11.79

Total purchased for MDR : $66.19

Rough Cost Estimate (Projected for CDR / FPR)

- (PCBs, MCUs, push buttons, demo aircraft)
- Demo Aircraft (with servos and engine) : roughly $89.99 - $100
- MCUs ATMEGA 328 (4pcs) : $12
- Push Buttons (pack of 25 pcs) : 7.98$
- PCBs (customized) : ~ $ 25

- Total for MDR + CDR : ~ $221.17

CDR Deliverables

1. Implementation of Kalman Filter & calibration mechanism (Bradley)
2. Custom PCB schematic design via Altium/KiCad (Jacob)*
3. Custom PCB construction via Altium (Son)*
4. Realization of Arduino Nano code on PCB-mounted MCUs (Bradley)
5. Implementation of finalized glove layout/assembly (Jacob & Son)
6. Implementation of finalized RC plane modifications/assembly (Jacob & Son)

*Son will deal with high-level PCB realization & fab-house coordination, Jacob will deal with low-level schematic design & component layout.

Gantt Chart

GANTT CHART PROJECT TEAM 21

Team plan for Spring 2021

RC Plane

Task ID	Task Name	Start Date	End Date	2/1/2021	2/8/2021	2/15/2021	2/22/2021	3/1/2021	3/8/2021	3/15/2021
1	Implement Filtering and Calibration method (Bradley)	2/1/2021	2/14/2021							
2	Testing and Debugging the main code (All)	2/1/2021	2/20/2021							
3	Design and Implement Custom PCP (Son and Jake)	2/1/2021	2/24/2021							
4	Finalize the glove (Son and Jake)	2/14/2021	3/2/2021							
5	Finalize the RC plane modification (Son and Jake)	2/13/2021	3/6/2021							
6	Testing and Debugging for CDR (All)	3/1/2021	3/6/2021							
7	CDR preparation (All)	3/2/2021	3/12/2021							
8	CDR presentation (All)	3/4/2021	3/12/2021							

University of Massachusetts Amherst BE REVOLUTIONARY

MDR Accomplishments (Individual - Son)

- **Filtering Design (Kalman Filter): may try using FIR and IIR**

  - Principle : Dealing with gravitational acceleration error. Using mathematical model, the filter predicts based on previous system state and the variance of noise

- **Calibration alternative method (dealing with offset error)**
- **Flex Sensor Code**
- **Debugged and checked main code**

  - Conversion from Arduino Uno to Nano (MPU)

  - Checking all methods that are operative

MDR Accomplishments (Individual - Jacob)

- Developed/tested preliminary Arduino Uno code responsible for:
    - receiving input data from GY-521 IMU module & 3” flex sensor
    - interpreting & mapping this data to particular hand movements
    - generating dynamic PWM actuation signals for pitch/yaw/roll servos as well as LED (placeholder for main engine)
- Updated system specifications & problem statement
- Updated hardware block diagrams
- Determined block-level topology for proposed custom PCB requirement
- Proposed CDR deliverables and plan to discretize workload
- Proposed FPR hardware plan

MDR Accomplishments (Individual - Bradley)

- **Updated software block diagram**
- **Created Arduino Nano Demo**
  - 2x NRF24L01 RF Transmitter + LSM9DS1 9-DOF
  - Programmed rolling/running average filter, transmission protocol (on arduino nano for NRF), debugged and tested on Arduino Nano
  - Servo control and mapping via 2.4GHz RF
  - Took prototype from the GY-521 IMU (Gyro) to the LSM9DS1 (Accelerometer and Gyro)
- **Software block diagram(s), helped with MDR slides, etc.**

Prototype Demo


Questions?
University of Massachusetts Amherst
BE REVOLUTIONARY™