**MEMORANDUM**  A yellow and green logo

Description automatically generated

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**To:** Prof. Charlie Refvem - crefvem@calpoly.edu

Prof. Eric Espinoza-Wade - erwade@calpoly.edu

Department of Mechanical Engineering, Cal Poly SLO

**From:** Jonathan Lam – [jlam94@calpoly.edu](mailto:jlam94@calpoly.edu)

Vinh T. Vo – [vvo11@calpoly.edu](mailto:vvo11@calpoly.edu)

Department of Mechanical Engineering, Cal Poly SLO

**Course:** ME 507 - Mechanical Control System Design

**Group:** MECHA08

**Date:** 05/10/2024

**RE:** Term Project Memo – Basketball Laucher

A diagram of a machine

Description automatically generatedA green machine with a blue container

Description automatically generated

# ABSTRACT

This report provides a detailed summary of the Basketball Launcher project from the first ideation stage to completion. After the project proposal, the team began by creating the mechanical design using SOLIDWORKS and successfully built the physical model within two weeks. Following this, the team focused on designing custom PCBs for both the launcher and its controller. Simultaneously, while the PCB design was underway, the team also developed the necessary software for both the controller and the launcher. This approach allowed for the software to be directly implemented onto the new PCB as soon as it arrived.

The purpose of this report is to document the entire project’s process, including the mechanical design, electronic design, and programming efforts. It also highlights some of the challenges encountered during the project and how they were addressed. Additionally, the report includes most of the materials and documents related to the project, which are attached as appendices for further reference. This comprehensive overview aims to provide a clear understanding of the project’s development

Table of Contents

[ABSTRACT 1](#_Toc167404559)

[OBJECTIVE 3](#_Toc167404560)

[OVERVIEW 3](#_Toc167404561)

[IDEATION 4](#_Toc167404562)

[MECHANICAL DESIGN 5](#_Toc167404563)

[CONTROLLER 6](#_Toc167404564)

[LAUNCHER 7](#_Toc167404565)

[CYCLOIDAL GEARBOX 7](#_Toc167404566)

[LAUNCHER BODY 8](#_Toc167404567)

[PCBA DESIGN 10](#_Toc167404568)

[CONTROLLER 11](#_Toc167404569)

[LAUNCHER 12](#_Toc167404570)

[SOFTWARE IMPLEMENTATION 14](#_Toc167404571)

[CONTROLLER 14](#_Toc167404572)

[LAUNCHER 15](#_Toc167404573)

[APPENDIX A – CONTROLLER PCB 16](#_Toc167404574)

[APPENDIX B1 – LAUNCHER PCB – PAGE 1 17](#_Toc167404575)

[APPENDIX B2 – LAUNCHER PCB – PAGE 2 18](#_Toc167404576)

[APPENDIX B3 – LAUNCHER PCB – PAGE 3 19](#_Toc167404577)

[APPENDIX B4 – LAUNCHER PCB – PAGE 4 20](#_Toc167404578)

[APPENDIX C – CONTROLLER CODE IMPLEMENTATION 21](#_Toc167404579)

[APPENDIX D – LAUNCHER CODE IMPLEMENTATION 22](#_Toc167404580)

[APPENDIX E – MECHANICAL BILL OF MATERIAL 23](#_Toc167404581)

[APPENDIX F – ELECTONIC BILL OF MATERIAL 25](#_Toc167404582)

# OBJECTIVE

## OVERVIEW

The Term Project in ME 507 requires students to design, build, program, test, and document an intelligent machine following specific guidelines. The project must include a custom PCB with an STM32F411 MCU (or similar), programmed in C, C++, or Rust. It should have two or more actuators, unique sensors, a closed-loop control algorithm, and a wireless controller for hands-free command or as a wireless emergency stop (e-stop).

The project also needs to fill up high construction quality and adheres to ME 507 lab safety rules, ensuring safety for all. Allowed manufacturing techniques include 3D printing, PCB fabrication, laser-cutting, and water-jetting for flat parts. Safety requirements include communicating with instructors before using or charging batteries and incorporating an emergency stop feature that activates communication with the wireless controller is lost.

## IDEATION

The design is inspired by the Peashooter from the Plants vs. Zombies mobile game. The overall design of the two main assemblies in the project is sketched and listed below for reference. The sketch represents a rough idea of how all components are connected

**A diagram of a machine

Description automatically generated**

**Figure 1**. Ideation Sketch

The project is divided into two main assemblies:

* The controller will use an STM32 Black Pill microcontroller with an MPU6050 imu sensor attached to the user's hand. The controller will send signals via Bluetooth to the launcher, allowing the launcher to move according to the imu sensor and launch the ball.
* The launcher will be the main assembly of the project, it receives data from the controller and moves in 2 DOF (yaw & pitch). It will launch the basketball when the user pushes the button from the controller. A custom PCB centered around the STM32F401CEU6 chip is also a main requirement for this project and acts as the centerpiece of all mechanical components.

# MECHANICAL DESIGN

The CAD modeling phase of the Basketball Launcher project was completed ahead of schedule, allowing the primary focus to shift to electronics and programming. Early completion of the CAD design ensured sufficient time to optimize the PCBA

The final CAD model, shown below, features the detailed design of the launcher with the controller conveniently attached to the side. Completing the CAD modeling early also helped identify and address potential mechanical issues, providing a solid implementation of electronic and programming tasks, thereby ensuring the project proceeded smoothly.

A diagram of a machine

Description automatically generated

**Figure 2**. CAD Design

## CONTROLLER

A person holding a blue device

Description automatically generated **Figure 3**. Controller

The controller housing includes two 3D-printed brackets connected by a hinge, allowing easy access to the PCBA inside. The controller features two buttons: a green button that puts the launcher into a moving mode, which follows the controller's motion via IMU data, and a black button that speeds up the BLDC motor, launching the ball. Note that the implemented code allows only one button to be pressed at a time; pressing both buttons simultaneously will result in no action unless user wants to trigger the launcher when it’s in the Emergency Stop stage.

Additionally, there are three indicator LEDs for a user-friendly interface. One LED indicates the IMU connection status, while the other two LEDs indicate when either of the buttons is pressed. In terms of electronics and programming, the MPU6050 IMU uses the I2C communication protocol to send angular speed data to the STM32 BlackPill microcontroller. The microcontroller combines this data with the status of the two buttons and sends it to the HC-05 Bluetooth module in SLAVE mode via UART. The HC-05 then transmits this data to another HC-05 module in MASTER mode, which is connected to the launcher's custom PCBA.

More details about the code implementation will be discussed in the SOFTWARE IMPLEMENTATION section and Appendix C.

## LAUNCHER

### CYCLOIDAL GEARBOX

A drawing of a circular object

Description automatically generated**Figure 4**. Cycloidal Gearbox

The concept of the cycloidal gearbox is inspired by the YouTube channel "How To Mechatronics." This gearbox aims to design a 3D-printed mechanism robust enough to support spinning the entire launcher’s body, providing a 1:10 reduction gear ratio, and driven by a stepper motor.

The initial design was developed using a MATLAB script capable of calculating various bearing dimension scenarios. This aids in the sorting and purchasing of materials while minimizing costs.

The gearbox is constructed using M6 bolts that act as shafts for the 6mm inner diameter ball bearings of the cycloidal gearbox. Additionally, the assembly includes ball bearings with inner diameters of 25mm, 35mm, and 55mm, which are positioned on the eccentric center shaft.

The bottom housing is designed to integrate a NEMA 17 stepper motor, which inputs rotational motion. The output achieves a ten time increase in torque, which is then transmitted via five M6 bolts to the launcher's U-brackets.

### LAUNCHER BODY

A green and red object with a ball on a table

Description automatically generated

**Figure 5**. Launcher Body (Front View)

The launcher's body features a 3D-printed U-shaped bracket mounted on top of the cycloidal gearbox using five M6 bolts, which supports the launcher's 1st axis. Above the gearbox, a shaft and bearings subassembly provide support for the 2nd axis, which is also driven by a NEMA 17 stepper motor with an integrated 1:19 planetary gear reduction gearbox. This built-in gearbox ensures that the launcher maintains its static position when no motion is required, preventing it from tilting upwards due to the offset center of mass of the launcher in case the motor is disabled.

The launcher barrel is designed to launch two 5” small-scale basketballs, addressing both safety and focus requirements of the course. The basketballs are launched using two BLDC motors with a 650 Kv rating, each equipped with a 3” diameter friction wheel with O-rings on the outside. These wheels rotate at 7215 rpm at 11.1VDC from a LiPo battery, theoretically providing a launch speed of approximately 95 ft/s under ideal, non-slip conditions. However, due to the safety concern, the BLDC motors’ speed are limited to 30% which correspond to 28.5 ft/s in ideal, non-slip condition.

However, several factors can reduce the actual speed. The deformation of the ball absorbs some kinetic energy, friction along the barrel reduces speed, and aerodynamic drag affects the ball once it exits the barrel. The quality of the ESC driving the BLDC motors can also cause speed inconsistencies between 2 BLDC motors, further influencing the ball's trajectory due to drag and Magnus effect.

A green machine with a blue container

Description automatically generated

**Figure 6**. Launcher Body (Back View)

Below the barrel, there are three limit switches that restrict the motion of the launcher in both vertical and horizontal directions. These limit switches come with a built-in pull-up resistor and a B3B-XH connector, making it convenient to connect to the main PCBA. The electronic box houses the PCBA, an 11.1V LiPo battery encased in an explosion-proof LiPo battery safe bag, and all wire connections between components. The box is topped with a lid featuring a rocket switch to power the project. This electronic box not only shields the electronic components from external factors such as dirt and water but also protects the surroundings in case of a battery explosion.

# PCBA DESIGN

A blue electronic device with many small components

Description automatically generated with medium confidence

**Figure 7**. Controller PCB Assembly

A close-up of a circuit board

Description automatically generated

**Figure 8**. Launcher PCB Assembly

## CONTROLLER

A close-up of a circuit board

Description automatically generated

**Figure 8**. Controller PCB

In the controller’s PCB, the connections are relatively simple compared to the launcher PCB. The controller’s PCB primarily serves as an extended platform for mounting the STM32 BlackPill and the MPU6050, with internal wire trace connections for I2C communication to the BlackPill's I2C1 bus. The PCB also includes internal trace connections for the HC-05 Bluetooth module, connected to the BlackPill's UART1. Additionally, there are three surface-mounted indicator LEDs with resistors to display the IMU status and a button for user interaction.

The board is powered by a 5V USB-C cable, which can draw power from an external source such as a laptop, a 5V adapter, or a 5V battery. More details regarding the critical wire trace connections between the MCU and are described in the table below.

**Table 1**. Controller PCB Connectivity

|  |  |  |
| --- | --- | --- |
| MCU | Description | Connection |
| PA9 | UART1 – TX | HC-05 – RX |
| PA10 | UART2 – RX | HC-05 – TX |
| PB3 | GPIO-B3 | R1 – LED1 – GND |
| PB4 | GPIO-B4 | R2 – LED2 – GND |
| PB5 | GPIO-B5 | R3 – LED3 – GND |
| PB6 | I2C1 – SCL | MPU-6050 - SCL |
| PB7 | I2C1 – SDA | MPU-6050 - SDA |

**Note:** For detail PCB schematic, refer to Appendix A, which attached at the end of this report**.**

## LAUNCHER

A close-up of a circuit board

Description automatically generated

**Figure 9**. Launcher PCB

The launcher PCB is the project's centerpiece, consuming the most effort and time for optimization and revisions. Due to time constraints, students could only purchase the fabricated PCB once. Overall, the PCB including 4 separate area that is describe in Appendix B1 à B4 as follow:

* The 1st area (Appendix B1): located at the bottom right corner of the PCB is the power management section, which includes all components related to the power source. This area features a fuse and a PMOSFET for circuit protection in case the polarity of the 12 VDC power source is reversed. It also includes a switch regulator to step down from 12VDC to 5VDC, followed by a linear regulator to further reduce the voltage from 5VDC to 3.3VDC. Additionally, this section contains the necessary resistors, capacitors, and other components required for the operation of these regulators.
* The 2nd area (Appendix B2): located in the top left corner of the PCB is the section dedicated to the STM32F411CEU6 MCU chip. This area includes all necessary bypass capacitors for filtering the input power to the MCU, a typical reset circuit, and a 25 MHz crystal to support the MCU’s functionality. Additionally, this section contains three indicator LEDs for user interface and required resistors, capacitors for the chip
* The 3rd area (Appendix B3): Located in the bottom left area of the PCB is the section focused on the two DRV8825 stepper motor driver chips, which control and run the two NEMA 17 stepper motors mentioned in the launcher description. These drivers are paired with potentiometers that can provide a voltage signal from 0-0.45VDC to the VREF pin, thereby limiting the current through the stepper motors from 0-0.9A. Additionally, this section includes all the necessary resistors and capacitors required for the operation of these two drivers.
* The 4th area (Appendix B4): this section also includes additional components such as connectors, test pads, and unused pins distributed across the PCB for potential use.

The board is powered by a 12 VDC power source, which can draw power from a 12 VDC adapter or an 11.1 V Lipo battery. More details regarding the critical wire trace connections between the MCU and are described in the table below.

**Note:** For detail PCB schematic, refer to Appendix B1 à B4, which attached at the end of this report**.**

**Table 2**. Launcher PCB Connectivity

|  |  |  |
| --- | --- | --- |
| MCU | Description | Connection |
| PA1 | PWM TIM2\_CH2 | ESC-1 |
| PA2 | PWM TIM2\_CH2 | ESC-2 |
| PA3 | GPIO OUTPUT | EN\_1 (DRV8825) |
| PA4 | GPIO OUTPUT | EN\_2 (DRV8825) |
| PA5 | GPIO OUTPUT | DIR\_1 (DRV8825) |
| PA6 | GPIO OUTPUT | DIR\_2 (DRV8825) |
| PA7 | GPIO OUTPUT | STEP\_1 (DRV8825) |
| PA8 | GPIO OUTPUT | STEP\_2 (DRV8825) |
| PA9 | USART1\_TX | HC-05 – RX |
| PA10 | USART1\_RX | HC-05 – TX |
| PA11 | USART6\_TX | RX (DEBUG) |
| PA12 | USART6\_RX | TX (DEBUG) |
| PA13 | SWDIO | SWDIO (ST-LINK) |
| PA14 | SWCLK | SWCLK (ST-LINK) |
| PH0 | OSC-IN | OSC-IN (CRYSTAL) |
| PH1 | OSC-OUT | OSC-OUT (CRYSTAL) |
| PB0 | GPIO OUTPUT | Switch 1 |
| PB1 | GPIO OUTPUT | Switch 2 |
| PB2 | GPIO OUTPUT | Switch 3 |

# 

# SOFTWARE IMPLEMENTATION

## CONTROLLER

**Table 3**. Controller Finite State Machine

|  |  |  |  |
| --- | --- | --- | --- |
| State | Description | Actions | Transitions |
| STATE0 | INIT | - Attempt to connect  - Turn on LED1 if successful | - If unsuccessful, go à STATE1  - If successful, go à STATE2 |
| STATE1 | ERROR | - Toggle LED1 if failed to connect to the IMU | - Always go à STATE0 |
| STATE2 | IMU | - Continuously read the data from the IMU | - Always go à STATE3 |
| STATE3 | BUTTON\_LED | - Read the buttons + update the LEDs | - Always go à STATE4 |
| STATE4 | TRANSFER | - Send data to UART1 & UART6 (debug) | - Always go à STATE2 |

A diagram of a state error

Description automatically generated

**Figure 10**. Controller Finite State Machine

## LAUNCHER

**Table 4**. Launcher Finite State Machine

|  |  |  |  |
| --- | --- | --- | --- |
| State | Description | Actions | Transitions |
| STATE0 | INIT | Init all components | Always go à STATE0 |
| STATE1 | DECISION HUB | Check condition and branch to different states | If S1 = 1 || S2 = 1 || S3 = 1à STATE3  ESTOP = 1 à STATE2  MOVE = 1 & SHOT = 0 à STATE3  MOVE = 0 & SHOT = 1 à STATE4 |
| STATE2 | ESTOP | Stop all components | If dT < 5000 🡪 STATE2  dT > 5000 🡪 STATE1 |
| STATE3 | STEPPER RUN | Step 2 steppers in different speed | Always go à STATE1 |
| STATE4 | BLDC RUN | Spin 2 BLDC Motor at 30% speed | Always go à STATE1 |

A diagram of a state

Description automatically generated

**Figure 11**. Controller Finite State Machine

# APPENDIX A – CONTROLLER PCB

A diagram of a computer

Description automatically generated

# APPENDIX B1 – LAUNCHER PCB – PAGE 1

A diagram of a machine

Description automatically generated

# APPENDIX B2 – LAUNCHER PCB – PAGE 2

A diagram of a circuit board

Description automatically generated

# APPENDIX B3 – LAUNCHER PCB – PAGE 3

A diagram of a computer

Description automatically generated

# APPENDIX B4 – LAUNCHER PCB – PAGE 4

A white paper with orange and black text

Description automatically generated with medium confidence

# APPENDIX C – CONTROLLER CODE IMPLEMENTATION

while (1) {

    switch (STATE) {

            case 0: // STATE\_INIT

                if (stat) {

                  stat = mpu6050\_init(&imu, &hi2c1);

                  HAL\_Delay(500);

                  mpu6050\_calibrate(&imu);

                  HAL\_Delay(500);

                  HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_3, stat);

                  STATE = STATE\_2\_IMU;

                } else {

                    STATE = STATE\_1\_ERROR;

                }

                break;

            case 1:

                HAL\_GPIO\_TogglePin(GPIOB, GPIO\_PIN\_3);

                HAL\_Delay(200);

                STATE = STATE\_0\_INIT;

                break;

            case 2:

                mpu6050\_update(&imu);

                gX = mpu6050\_get\_gX(&imu);

                gY = mpu6050\_get\_gY(&imu);

                gZ = mpu6050\_get\_gZ(&imu);

                STATE = STATE\_3\_BUTTON\_LED;

                break;

            case 3:

                shot = HAL\_GPIO\_ReadPin(GPIOA, GPIO\_PIN\_4);

                move = HAL\_GPIO\_ReadPin(GPIOA, GPIO\_PIN\_5);

                HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_4, shot); // LED2

                HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_5, move); // LED3

                STATE = STATE\_4\_TRANSFER;

                break;

            case 4:

                snprintf(buffer, sizeof(buffer), "%d\t%d\t%d\t%d\r\n", (int)move, (int)shot, gZ, gY);

                HAL\_UART\_Transmit(&huart1, (uint8\_t \*)buffer, strlen(buffer), 100);

                HAL\_UART\_Transmit(&huart2, (uint8\_t \*)buffer, strlen(buffer), 100);

                HAL\_Delay(10);

                STATE = STATE\_2\_IMU;

                break;

        }

}

# APPENDIX D – LAUNCHER CODE IMPLEMENTATION

**while** (1)

{

// RUN FSM

// STATE 0: INIT ALL OBJECTS

**if** (STATE == STATE\_0\_INIT) {

// INIT RADIO TIMER

**HAL\_TIM\_Base\_Start**(&htim4);

**HAL\_TIM\_IC\_Start\_IT**(&htim4, TIM\_CHANNEL\_1);

**HAL\_TIM\_IC\_Start\_IT**(&htim4, TIM\_CHANNEL\_2);

// INIT STEPPER MOTOR

**Stepper\_init**(&STEPPER1, GPIOA, GPIO\_PIN\_3, GPIO\_PIN\_5, GPIO\_PIN\_7);

**Stepper\_init**(&STEPPER2, GPIOA, GPIO\_PIN\_4, GPIO\_PIN\_6, GPIO\_PIN\_8);

// INIT D4215 BLDC MOTORS

**D4215\_init** (&ESC1,&htim2,TIM\_CHANNEL\_2);

**D4215\_init** (&ESC2,&htim2,TIM\_CHANNEL\_3);

// INIT 3 LIMIT SWITCHES

**Switch\_init** (&S1, GPIOB, GPIO\_PIN\_0);

**Switch\_init** (&S2, GPIOB, GPIO\_PIN\_1);

**Switch\_init** (&S3, GPIOB, GPIO\_PIN\_2);

// INIT 2 EXTRA LEDS

**LED\_init** (&LED1, GPIOB, GPIO\_PIN\_14);

**LED\_init** (&LED2, GPIOB, GPIO\_PIN\_15);

// MOVE --> STATE 1: DECISION HUB

STATE = STATE\_1\_HUB;

}

// STATE 1: DECISION HUB TO BRANCH TO DIFFERENT HUB

**else** **if** (STATE == STATE\_1\_HUB) {

// ENABLE 2 STEPPERS

**Stepper\_enable**(&STEPPER1);

**Stepper\_enable**(&STEPPER2);

// READ 3 LIMIT SWITCHES

SW1 = **Switch\_getStatus**(&S1);

SW2 = **Switch\_getStatus**(&S2);

SW3 = **Switch\_getStatus**(&S3);

// CHECK ESTOP

**if** (ESTOP == 1) {

ti = **HAL\_GetTick**();

STATE = STATE\_2\_ESTOP;

}

// CHECK LIMIT SWITCHES OR "MOVE" FLAG FROM CONTROLLER

**else** **if** ((MOVE == 1 && SHOT == 0) || SW1 == 0 || SW2 == 0 || SW3 == 0) {

**if** (SW1 == 0) {

SW = 1;

angleTarget = 10;

yawDirectionSW = 0;

} **else** **if** (SW2 == 0) {

SW = 2;

angleTarget = 10;

yawDirectionSW = 1;

} **else** **if** (SW3 == 0) {

SW = 3;

angleTarget = 10;

pitchDirectionSW = 0;

} **else** {

angleTarget = 10;

SW = 0;

}

STATE = STATE\_3\_STEPPER;

}

// CHECK "SHOT" FLAG FROM CONTROLLER

**else** **if** (MOVE == 0 && SHOT == 1) {

STATE = STATE\_4\_BLDC;

}

// STOP IF

**else** {

**D4215\_set**(&ESC1, 0);

**D4215\_set**(&ESC2, 0);

STATE = STATE\_1\_HUB;

}

}

// STATE 2: EMERGENCY STOP STATE

**else** **if** (STATE == STATE\_2\_ESTOP) {

// DISABLE STEPPER MOTORS

**Stepper\_disable**(&STEPPER1);

**Stepper\_disable**(&STEPPER2);

// STOP BLDC MOTORS

**D4215\_set**(&ESC1, 0);

**D4215\_set**(&ESC2, 0);

// CHECK FOR RESET BY HOLDING BUTTONS FOR 5 SECOND

**if** (MOVE == 1 && SHOT == 1) {

tf = **HAL\_GetTick**();

dt += (tf-ti);

ti = tf;

}

**else** {

dt = 0;

}

STATE = (dt > 5000) ? STATE\_1\_HUB : STATE\_2\_ESTOP;

}

// STATE 3: SPIN STEPPER MOTORS

**else** **if** (STATE == STATE\_3\_STEPPER) {

// LEFT SWITCH HIT --> MOVE TO THE RIGHT

**if** (SW == 1) {

**for** (**int** i = 0; i <= angleTarget/2; i++) {

**Stepper\_setspeed** (&STEPPER1, 10, yawDirectionSW);

}

SW = 0;

}

// RIGHT SWITCH HIT --> MOVE TO THE LEFT

**else** **if** (SW == 2) {

**for** (**int** i = 0; i <= angleTarget/2; i ++) {

**Stepper\_setspeed** (&STEPPER1, 10, yawDirectionSW);

}

SW = 0;

}

// MID SWITCH HIT --> MOVE UP

**else** **if** (SW == 3) {

**for** (**int** i = 0; i <= 10 \* angleTarget; i ++) {

**Stepper\_setspeed** (&STEPPER2, 10, pitchDirectionSW);

}

SW = 0;

}

// IF NOT SWITCH, MOVE ACCODING TO THE IMU DATA

**else** {

**uint16\_t** yaw\_map = **map**(yaw, 20, 250, 5, 30);

**if** (yaw > 20) {

**for** (**int** i = 0; i <= yaw\_map; i ++) {

**Stepper\_setspeed** (&STEPPER1,1,yawDirection);

}

}

**uint16\_t** pitch\_map = **map**(pitch, 20, 250, 10, 50);

**if** (pitch > 20) {

**for** (**int** i = 0; i <= pitch\_map; i ++) {

**Stepper\_setspeed**(&STEPPER2,1,pitchDirection);

}

}

}

STATE = STATE\_1\_HUB;

}

// STATE 4: BLDC MOTOR

**else** **if** (STATE == STATE\_4\_BLDC) {

**D4215\_set**(&ESC1, 30);

**D4215\_set**(&ESC2, 30);

STATE = STATE\_1\_HUB;

}

// STATE 1: DECISION HUB IN CASE SOME ERROR

**else** {

STATE = STATE\_1\_HUB;

}

/\* USER CODE END WHILE \*/

/\* USER CODE BEGIN 3 \*/

}

/\* USER CODE END 3 \*/

}

# APPENDIX E – MECHANICAL BILL OF MATERIAL

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Picture | Parts | Unit Price | Qty | Total Amount |
| 1 | A black and silver mechanical device  Description automatically generated | NEMA17 | $ 15.00 | 1 | $ 15.00 |
| 2 | A small black and silver device with blue and red wires  Description automatically generated | NEMA17 + GEAR BOX | $ 40.00 | 1 | $ 40.00 |
| 3 | A small electric motor with wires and wires  Description automatically generated | D421 BLDC | $ 27.00 | 2 | $ 54.00 |
| 4 | A blue and red electrical device  Description automatically generated | UBEC ESC | $ 22.00 | 2 | $ 44.00 |
| 5 | A black round object on a white background  Description automatically generated | O RING | $ 2.00 | 2 | $ 4.00 |
| 6 | A black and red electrical wire  Description automatically generated | SWITCH | $ 10.00 | 1 | $ 10.00 |
| 7 | A close up of a basketball  Description automatically generated | BASKETBALL | $ 6.00 | 1 | $ 6.00 |
| 8 | A close up of a circuit board  Description automatically generated | HC-05 | $ 10.00 | 2 | $ 20.00 |
| 9 | A black and silver device  Description automatically generated | LIMIT SWITCH | $ 1.00 | 3 | $ 3.00 |
| 10 |  | CONNECTOR | $ 10.00 | 1 | $ 10.00 |
| 11 | A couple of electrical wires  Description automatically generated with medium confidence | CONNECTOR | $ 3.00 | 2 | $ 6.00 |
| 12 | A couple of batteries with red and green wires  Description automatically generated | BATTERY | $ 18.00 | 2 | $ 36.00 |
| 13 | A white swirly object  Description automatically generated | FILLAMENT | $ 20.00 | 4 | $ 80.00 |
| 14 | A group of silver rings  Description automatically generated | BEARINGS | $ 10.00 | 2 | $ 20.00 |
| 15 | A group of silver circles  Description automatically generated | BEARINGS | $ 10.00 | 3 | $ 30.00 |
| 16 | A black and silver ring  Description automatically generated | BEARINGS | $ 6.00 | 2 | $ 12.00 |
| 17 | A pair of silver rings  Description automatically generated | BEARINGS | $ 10.00 | 1 | $ 10.00 |
| 18 | A couple of metal objects with blue rings  Description automatically generated with medium confidence | NUTS | $ 6.00 | 1 | $ 6.00 |
| 19 | A close up of a nut  Description automatically generated | NUTS | $ 10.00 | 1 | $ 10.00 |
| 20 | A group of screws on a white background  Description automatically generated | BOLTS | $ 10.00 | 2 | $ 20.00 |

# APPENDIX F – ELECTONIC BILL OF MATERIAL

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Picture | Number Part | | Buy | Cost |
| CONNECTOR | | | | |
| A black object with a round hole  Description automatically generated | | PJ-102AH | 1 | $ 0.70 |
| A white electrical plug  Description automatically generated | | B3B-XH-A | 4 | $ 0.84 |
| A white electrical connector with metal pins  Description automatically generated with medium confidence | | B4B-XH-A | 3 | $ 0.69 |
| CAPACITORS | | | | |
| A close up of a box  Description automatically generated | | C0805C103J1GEC7800 | 4 | $ 0.58 |
| A black and white box  Description automatically generated | | CC0603KRX7R8BB104 | 20 | $ 2.00 |
| A grey and brown box  Description automatically generated | | C2012X5R1E475K125AB | 5 | $ 0.95 |
| A black and grey cube  Description automatically generated | | CC0805KRX5R8BB106 | 3 | $ 1.14 |
| A red and white capacitor  Description automatically generated | | 8.6002E+11 | 2 | $ 0.26 |
| A black and grey cube  Description automatically generated | | CC0603KRX5R8BB105 | 10 | $ 1.00 |
| A black and white box  Description automatically generated | | CC0805MKX5R8BB226 | 3 | $ 1.23 |
| A close up of a box  Description automatically generated | | C0805C300M3GAC7800 | 5 | $ 1.20 |
| A grey and black rectangular object  Description automatically generated | | CGA4J2X7R1E474K125AA | 5 | $ 1.20 |
| RESISTORS | | | | |
| A blue square object with a hole in the middle  Description automatically generated | | 3386P-1-502LF | 1 | $ 1.77 |
| A white rectangular object with a black square on it  Description automatically generated | | MCU08050D5002BP100 | 2 | $ 1.70 |
| A white rectangular object with a black square on it  Description automatically generated | | RC0805FR-0710KL | 12 | $ 1.20 |
| A black rectangular object with a white border  Description automatically generated | | RC1206FR-071K5L | 5 | $ 0.50 |
| A white rectangular object with a black square on it  Description automatically generated | | ERJ-3EKF1004V | 5 | $ 0.50 |
| A white and black rectangular object  Description automatically generated | | ERJ-U6SFR10V | 8 | $ 5.28 |
| A white box with a black square  Description automatically generated | | RC0805FR-07200RL | 10 | $ 1.00 |
| OTHERS | | | | |
| A white box with black trim  Description automatically generated | | MFU0603FF04000P500 | 2 | $ 0.48 |
| A black and silver capacitor  Description automatically generated | | RLB0914-330KL | 2 | $ 0.84 |
|  | | TL1105BF160Q | 2 | $ 0.40 |
| A close up of a transistor  Description automatically generated | | 631-FC4STCBMF25.0-ND | 2 | $ 0.68 |
| A blue and silver object with two metal rods  Description automatically generated with medium confidence | | 36-5127-ND | 10 | $ 3.80 |
| A blurry image of a box  Description automatically generated | | VAOL-S8SB4 | 6 | $ 4.32 |
| PCB + MCU + REGULATORS + MOTOR DRIVER | | | | |
| A close-up of a black electronic device with Fallingwater in the background  Description automatically generated | | TLV1117LV33DCYR | 2 | $ 0.70 |
| A black and white square object  Description automatically generated | | PMPB15XP,115 | 2 | $ 0.94 |
| A black square object with five metal rods  Description automatically generated | | L6981N50DR | 1 | $ 2.87 |
| A close-up of a computer chip  Description automatically generated | | STM32F411CEU6 | 1 | $ 7.40 |
|  | | DRV8825 | 2 | $ 11.30 |
| A blue circuit board with white text  Description automatically generated | | LAUNCHER PCB | 1 | $ 72.28 |
| A blue circuit board with white text  Description automatically generated | | CONTROL PCB | 1 | $ 26.42 |