





**Department of Mechanical and Industrial Engineering  
Program: Mechanical Engineering**

Course Number	MEC830
Course Title	Mechatronics Systems Design
Semester/Year	Fall/2024
Instructor	Dr. Sajad Saeedi
TA	Ishan Mehta

<b>Project Report NO.</b>	<b>2</b>
---------------------------	----------

Report Title	Inverted Pendulum Design
--------------	--------------------------

Section No.	02
Group No.	01
Submission Date/ Due Date	November 20 <sup>th</sup> , 2024

Name	Student ID	Signature*
Adam Di Benedetto	XXXX20099	
Adam Moore	XXXX30313	
Waleed Idrees	xxxx7336	
Jacky Su	xxxx84295	Jacky Su
Hamayoon Ashraf	xxx03313	

(Note: remove the first 4 digits from your student ID)

\*By signing above you attest that you have contributed to this submission and confirm that all work you have contributed to this submission is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a "0" on the work, an "F" in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at: <http://www.rverson.ca/senate/policies/pol60.pdf>.

## Table of Contents

1. Abstract.....	5
2. Introduction.....	6
3. Theory.....	7
4. Design - MBSE.....	12
5. Failure Mode and Effect Analysis, FMEA.....	17
Top-Down Functional Approach.....	19
Bottom-Up Structural Approach.....	21
Severity Ranking (SR).....	23
Occurrence Rating (OR).....	30
Detection Ranking (DR).....	35
Risk Priority Number, RPN.....	40
Ishikawa (Fishbone) Diagram.....	47
FMEA Conclusion.....	48
6. Concurrency.....	49
Organization.....	49
Communication.....	49
Design Problem.....	50
Product Development.....	51
Responsibility Table.....	52
7. Modelling and Design.....	53
Mechanical Design.....	53
Electrical Design.....	58

8. HMI - Control Panel.....	59
Safety Considerations.....	59
Maintainability.....	61
Design of Controls.....	64
Bottom-Up Reasoning.....	68
Top-Down Reasoning.....	69
Labelling and Displays.....	70
9. Software Implementation.....	72
PID Control.....	74
As shown in this code, the proportional, integral, and derivative errors are all computed and added together.....	75
Pole-Placement Control.....	75
Noise and Filtering.....	75
10. Software Version Control.....	76
11. Performance and Responses.....	76
12. Bill of Materials / Cost Breakdown.....	79
13. References.....	80
14. Appendices.....	81
Appendix A - Panel Stand Part Drawing.....	81
Appendix B - Stand Wedge Part Drawing.....	82
Appendix C - Stepper Motor Mount Part Drawing.....	82
Appendix D - Encoder Mount Part Drawing.....	83
Appendix E - Pendulum Arm Part Drawing.....	83

Appendix F - Arm Support Part Drawing.....	84
Appendix G - Control Front Panel Part Drawing.....	84
Appendix H - Control Panel Supports Part Drawing.....	85
Appendix I - MBSE Drawings.....	85
Appendix J - Code.....	87



## **1. Abstract**

The report discusses in detail the design, development, and testing of a self-balancing pendulum system. The main goal of this project was to get practical experience while solving problems of dynamic control. In the design and execution, certain requirements were followed, such as keeping the length of the pendulum between 40-50 cm and using a mass of 50-100 grams. The project focuses on the integration of sensors, actuators, and control algorithms that will be able to provide real-time accuracy and responsiveness. Optimization with PID and pole placement controllers was used to successfully counter disturbances to maintain the pendulum upright. Performance testing showed that the system without the controller behaved as expected, with the pendulum oscillating and eventually settling at a downward position of 180 degree due to gravity. When using the PID controller, the system was quick to react to disturbances by making rapid motor adjustments to correct the pendulum angle. The pole placement controller also responded well to counter oscillations and keep the pendulum upright. These results demonstrate the effectiveness of the applied control strategies for achieving stability along with reliable performance.

## 2. Introduction

Inverted pendulum systems have been employed for years in studying feedback and control in dynamic systems. In fact, they are an excellent way to understand how stability works in real-time applications. This report discusses the design and implementation of an inverted pendulum system. The objective of this project is to obtain practical experience in mechatronics by building a self balancing inverted pendulum. It is designed using sensors, actuators, and an arduino based control system. Constant adjustments have to be made to maintain the pendulum upright to avoid tipping over, which highlights the core challenges of feedback and control. The pendulum would require precise real-time control so that stability can be maintained by not tilting and responding to external disturbances.. Systems of this nature are applied in automation and robotics, where stability and quick responses are essential. The project will be utilizing components from the ELEGOO kit, along with additional parts. The design specifications include a pendulum length of 40-50 cm and a mass of 50-100 grams. Also the system includes safety features and reliability by including an emergency stop, LCD interface, and modular construction for easy maintenance and part replacement. Additionally, Failure Mode and Effects Analysis (FMEA) was applied to identify potential risks and ensure the system performs consistently in various conditions. Furthermore, advanced control strategies such as PID and pole placement controllers were utilized to optimally stabilize the pendulum. These controllers were fine tuned to enhance system performance, ensuring errors and smooth operation. Through this project, critical concepts in dynamics system design and control were explored, offering valuable insights into real world engineering applications.

### 3. Theory

The theory for this project is very similar to that of its predecessor. The difference is that since the model will be physically constructed, theories like the equations of motion will not be directly used in the experiment. The equations of motion will nevertheless be laid out as a complete description of the experimental setup, and to define the variables of interest.

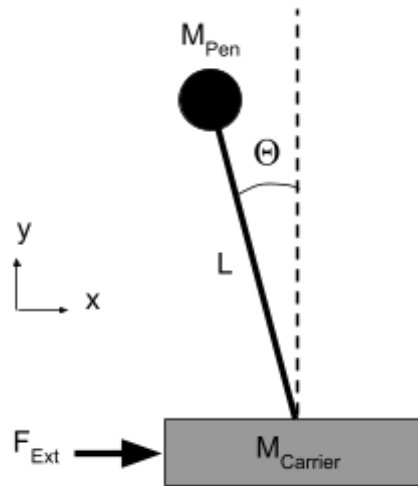


Figure 3.1: Free Body Diagram of the Inverted Pendulum System.

Figure 3.1 shows a general representation of the pendulum on a moving carrier with external force applied to the carrier to balance the pendulum. Even though the implementation we chose is rotary, the force applied to the carrier is tangential, so the 2-dimensional model above is still equivalent to our constrained 3-dimensional setup.

From this free body diagram, we can construct the following equations of motion the same as was done in project 1:

Beginning with:

$$T = \frac{1}{2} m_c \dot{x}^2 + \frac{1}{2} m_p V_p^2 \rightarrow \text{we need } V_p \text{ in usable terms!}$$

From this diagram, we can see that:

$$\begin{aligned} x_p &= x_c - L \sin \theta \\ y_p &= L \cos \theta \end{aligned} \quad \therefore V_p = \frac{d}{dt} [(x_c - L \sin \theta) + (L \cos \theta)] = [\dot{x}_c - L \dot{\theta} \cos \theta - (-L \dot{\theta} \sin \theta)]$$

$$V_p^2 = \dot{x}_c^2 - 2 \dot{x}_c \dot{\theta} \cos \theta + L^2 \dot{\theta}^2 \quad (=1)$$

Subbing the new  $V_p$  expression into  $T$  expression! (Take  $x, \dot{x}$  as  $x_c$  and  $\dot{x}_c$  from now on)

$$\begin{aligned} T &= \frac{1}{2} m_c \dot{x}^2 + \frac{1}{2} (\dot{x}^2 - 2 \dot{x} \dot{\theta} L \cos \theta + L^2 \dot{\theta}^2) \cdot m_p \\ &= \frac{1}{2} (m_c + m_p) \dot{x}^2 - m_p \dot{\theta} \dot{x} L \cos \theta + \frac{1}{2} m_p L^2 \dot{\theta}^2 \end{aligned}$$

Now, we don't actually need a potential energy expression if we take 2 equations for the 2 Virtual forces:

torque =  $m_p g L \sin \theta$

$$\begin{cases} \frac{d}{dt} \frac{\partial T}{\partial \dot{x}} - \frac{\partial T}{\partial x} = F & \text{①} \\ \frac{d}{dt} \frac{\partial T}{\partial \dot{\theta}} - \frac{\partial T}{\partial \theta} = m_p g L \sin \theta & \text{②} \end{cases}$$

Solving ①:

$$F = \frac{d}{dt} [(m_c + m_p) \dot{x} - m_p \dot{\theta} L \cos \theta] - 0 = (m_c + m_p) \ddot{x} - m_p L \ddot{\theta} \cos \theta + m_p L \dot{\theta}^2 \sin \theta \quad \text{new eqn. ①}$$

Solving ②:

$$m_p g L \sin \theta = \frac{d}{dt} [-m_p \dot{x} L \cos \theta + m_p L^2 \dot{\theta}] - m_p \dot{\theta} \dot{x} L \sin \theta = -m_p L \ddot{x} \cos \theta + m_p L \dot{x} \sin \theta + m_p L^2 \ddot{\theta} - m_p \dot{\theta} \dot{x} L \sin \theta$$

Simplify  $\rightarrow$

$$m_p g L \sin \theta = -m_p L \ddot{x} \cos \theta + m_p L \dot{x} \sin \theta + m_p L^2 \ddot{\theta} - m_p \dot{\theta} \dot{x} L \sin \theta$$

$$L \ddot{\theta} = \ddot{x} \cos \theta + g \sin \theta \quad \text{new eqn. ②}$$

If we want to incorporate air resistance and damping, we simply replace  $F$  with  $F - c_{\text{cart}} \dot{x}$  and replace  $m_p g L \sin \theta$  with  $m_p g L \sin \theta - c_{\text{pen}} \dot{\theta} - d_{\text{pivot}} \dot{\theta}$  and rearrange to get the following EOM's:

$$\begin{aligned} \text{①} \quad \ddot{x} &= \frac{F + m_p L \ddot{\theta} \cos \theta - m_p L \dot{\theta}^2 \sin \theta - c_{\text{cart}} \dot{x}}{m_c + m_p} \\ \text{②} \quad \ddot{\theta} &= \frac{g \sin \theta + \ddot{x} \cos \theta - c_{\text{pen}} \dot{\theta} - d_{\text{pivot}} \dot{\theta}}{L} \end{aligned}$$

Figure 3.2: Equation of Motion Derivation

To stabilize the pendulum, we will use 2 different controller types PID and full-state feedback (which uses pole-placement for tuning). The controllers will both achieve stabilization by controlling the applied force at the carrier with their output.

PID:

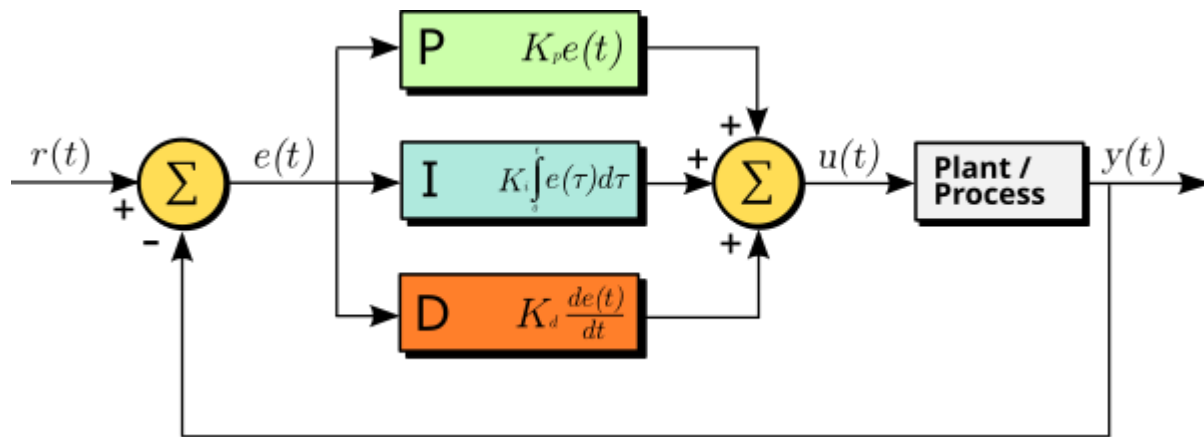


Figure 3.3: Block diagram of a PID controller.

Credit: Arturo Urquiza, Wikipedia.org (Public Domain).

The PID controller is the simplest of the two because it's a SISO (Single Input, Single Output) controller. Like any controller, it requires a feedback loop and in this case, feedback is subtracted from the setpoint to get the error at a given time,  $e(t)$ . This is used to calculate 3 different values which will later be combined into the final output.

The first value is the error multiplied by the proportional gain, which results in a response directly proportional to the current deviation from the setpoint. The second value is the integral of  $e(t)$  multiplied by the integral gain, which results in a response proportional to the accumulated error over time. The third value is the derivative of  $e(t)$  with respect to time, multiplied by the differential error, which results in a response proportional to the instantaneous rate of change of error.

The 3 gain values ( $K$ 's) can be tuned for an ideal response. Generally, you want the system to achieve the setpoint as quickly as possible with minimal overshoot and minimal steady-state error.

### Full-State Feedback (Pole Placement):

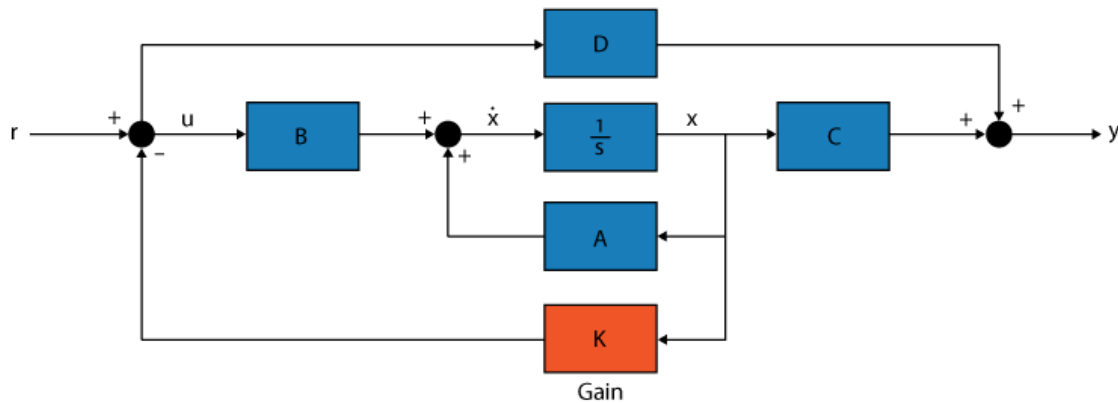


Figure 3.4 Block Diagram for Pole Placement (Full-state Feedback)

Credit: Mathworks.org (Public Domain)

A full-state feedback controller uses a MISO design (Multi-Input, Single-Output), because it takes all the variables representing the system's state and performs operations on them to produce a single output response. Each input has a gain ( $K$ ), so there can be any number of gain values, unlike a PID which always has 3. The gains are stored in a matrix to be multiplied into the input matrix.

The pole-placement approach for a full-state feedback controller is based on shifting the transfer function poles to a desirable location to achieve a certain response. For example, suppose the transfer function of a controller is  $G(s) = 1/[(s - 1)(s + 4)]$ . In that case, the poles are located at 1 and -4, as these values make the denominator equal zero, causing the result to equal infinity, i.e. a pole. Understanding where the poles need to be located is an important feat when using pole placement as well. Knowing the poles for stability allows for the

designer to choose strategic poles, negative imaginary, which allows the designer to choose which stability fits their system best, overdamped, underdamped or critically damped.

When applied in a software context, full state feedback can simply be implemented by measuring the 4 variables representing the state according to the equations of motion. These are  $x$ ,  $\dot{x}$ ,  $\theta$ , and  $\dot{\theta}$ . There is then a gain corresponding to each state variable.

The gains can be multiplied into their variables and then everything is summed up to compute the final response. You will see this implementation later in the report.

#### 4. Design - MBSE

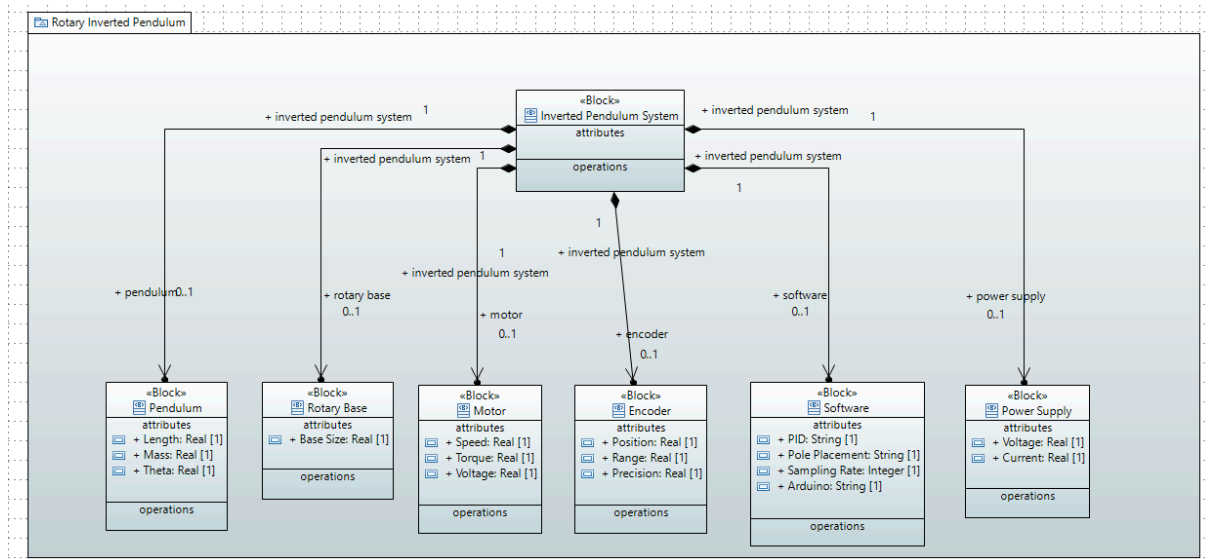


Figure 4.1 Block Definition Diagram of the whole system

The block definition diagram in figure 4.1 shows the combination of all the subsystems and their attributes in respect to the main system. Showing the subsystems of the whole system allowed the group to understand the different components that were needed to develop the project.

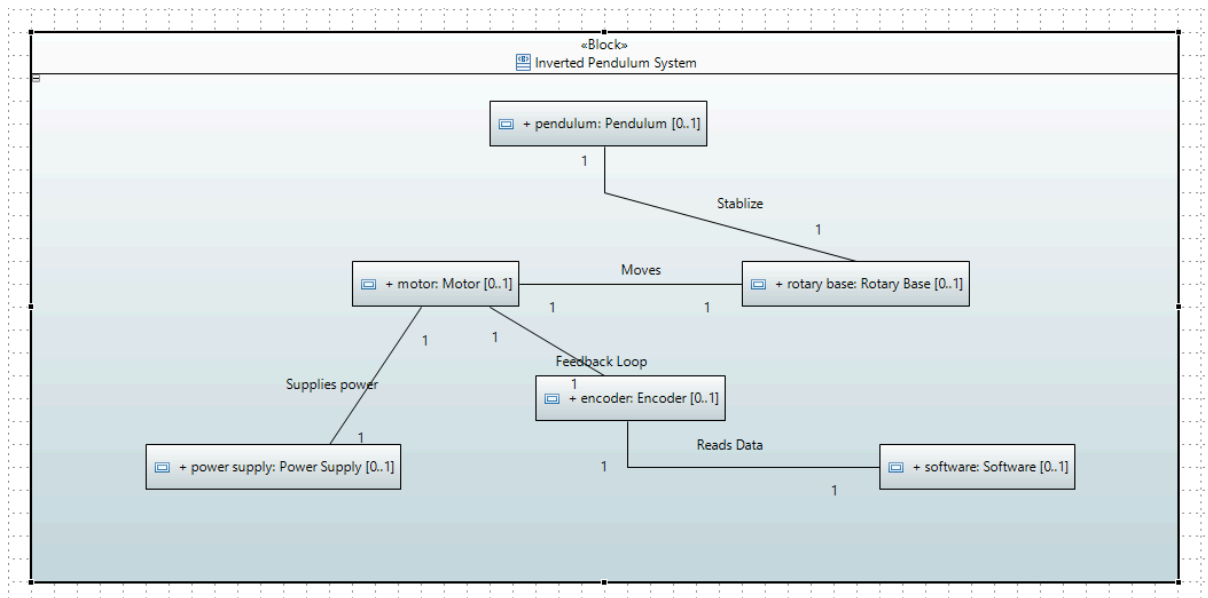


Figure 4.2 Internal Block Diagram of System



The internal block diagram is a more indepth view of the system. From the subsystems in the BDD it can model the system in a more in depth fashion. Showing the connections within the system and between the subsystems allowed the team to grasp a better understanding of the model in question, and gain more insight into the relationship between all the subsystems.

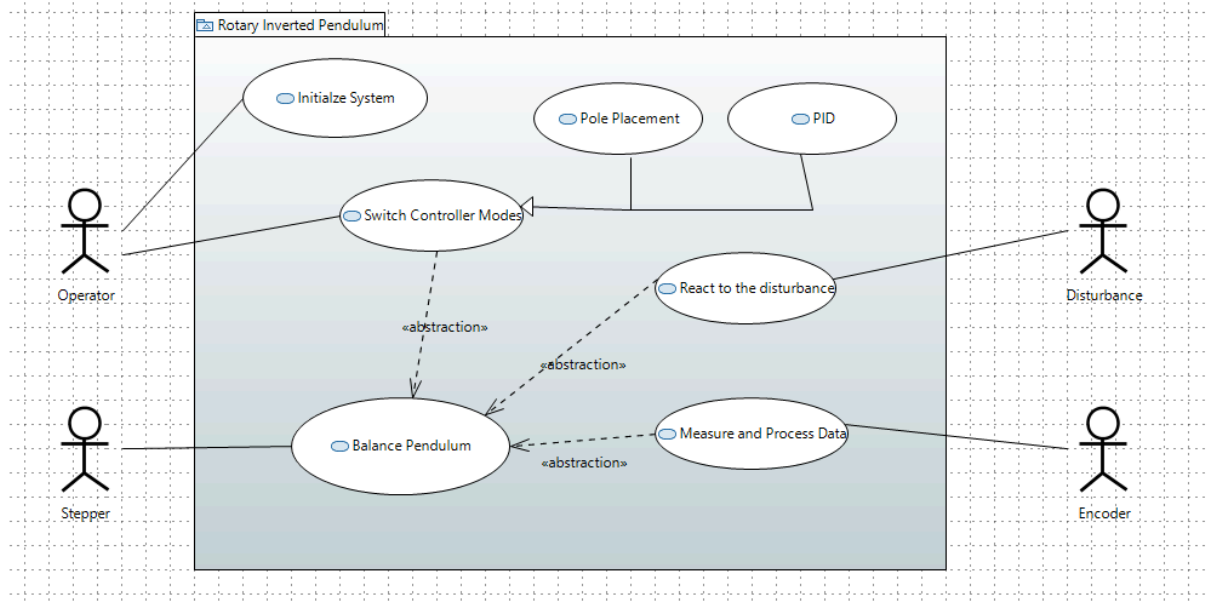


Figure 4.3 Use Case Diagram for the Rotary Inverted Pendulum

The importance of a Use Case Diagram (UCD) seen in figure 4.3 is understanding the actors when using the system. This UCD is modelled after the whole use of the system, by implementing the actors in this scenario it shows a path and simplistic version of how the system will interact with the different actors in the system. By including generalizations and “<<include>>” relationships it allowed us to better understand the system.

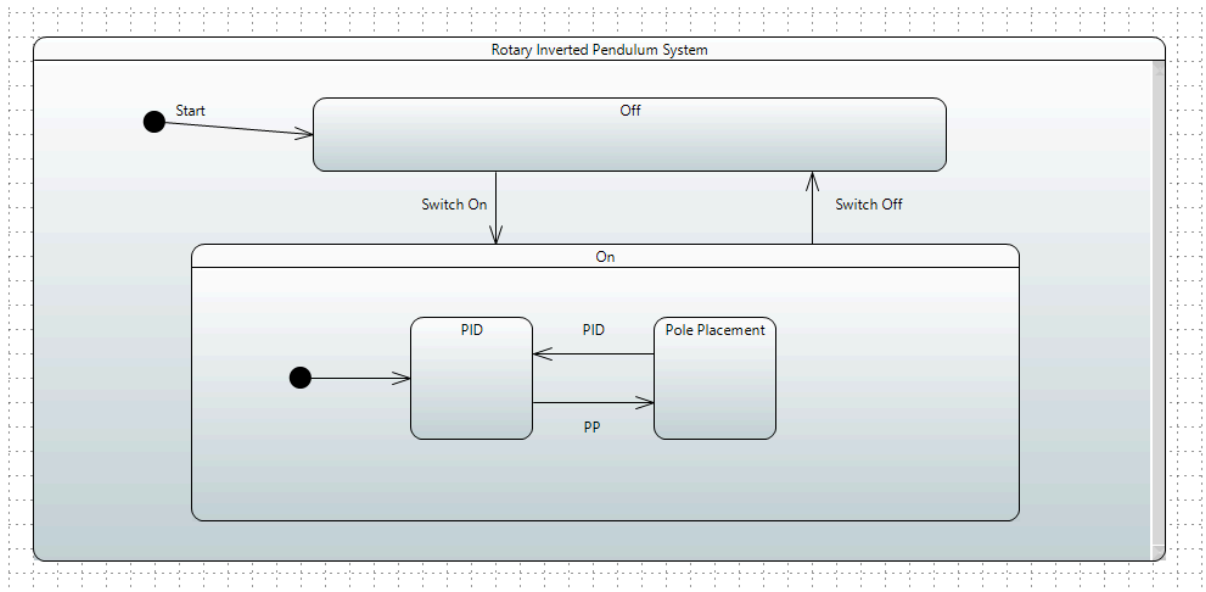


Figure 4.4 State Machine Diagram for Control Panel

The choice to use a State Machine Diagram (SMD) to model the control panel for the system seemed like an obvious choice when working with MBSE. The SMD allows to show the different flow states the model can be present in. By modelling the SMD it allowed for a better design of the control panel and how it is possible to flow seamlessly between the different controls and power states.

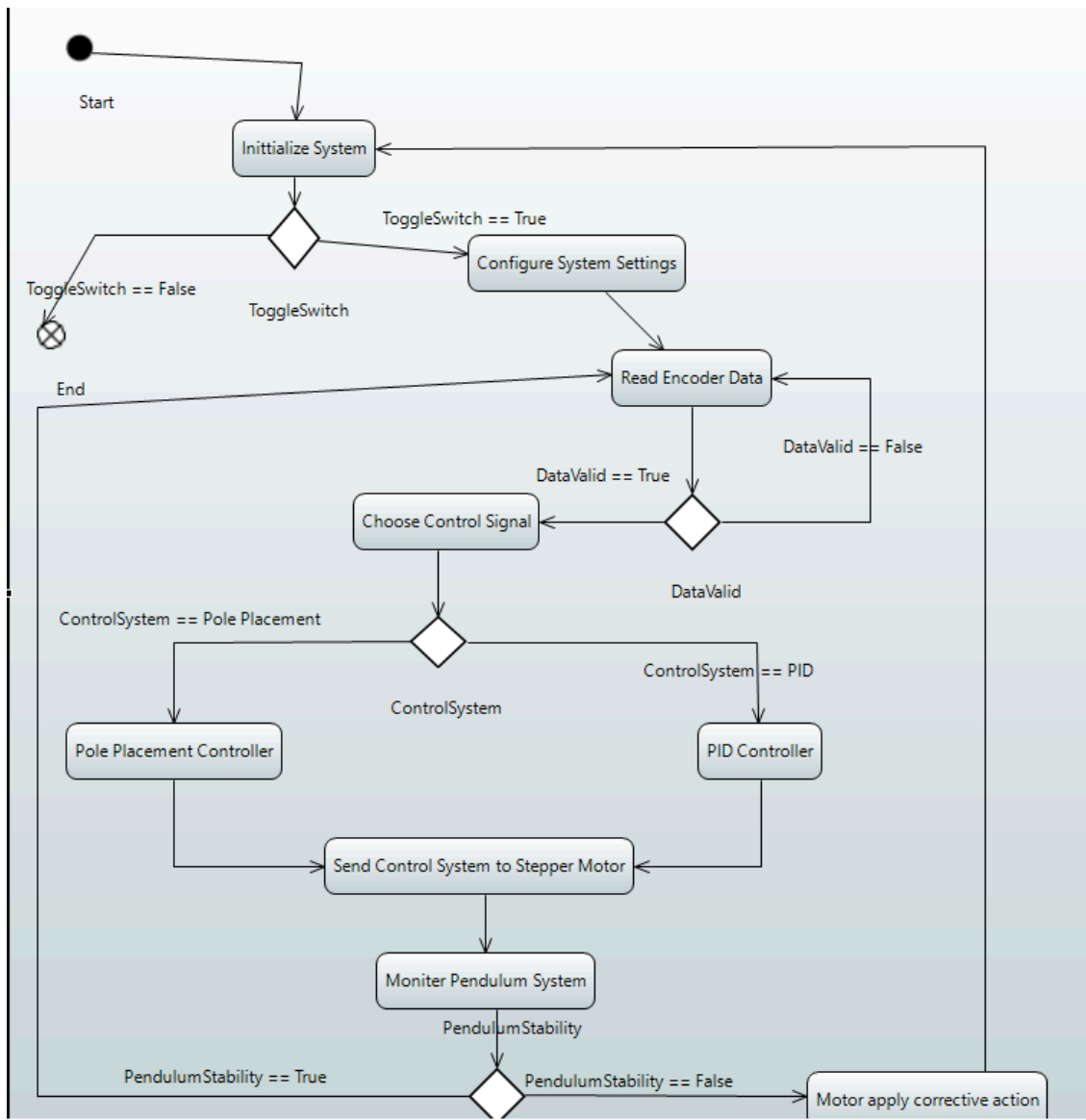


Figure 4.5 Activity Diagram of the System

The Activity Diagram, Figure 4.5, was modelled as a flow of the system, showing different decisions and use cases that may have come up in possible code and other scenarios. By showing the seamless flow of the system and allowing the designers to see how the systems activity in its optimized state.

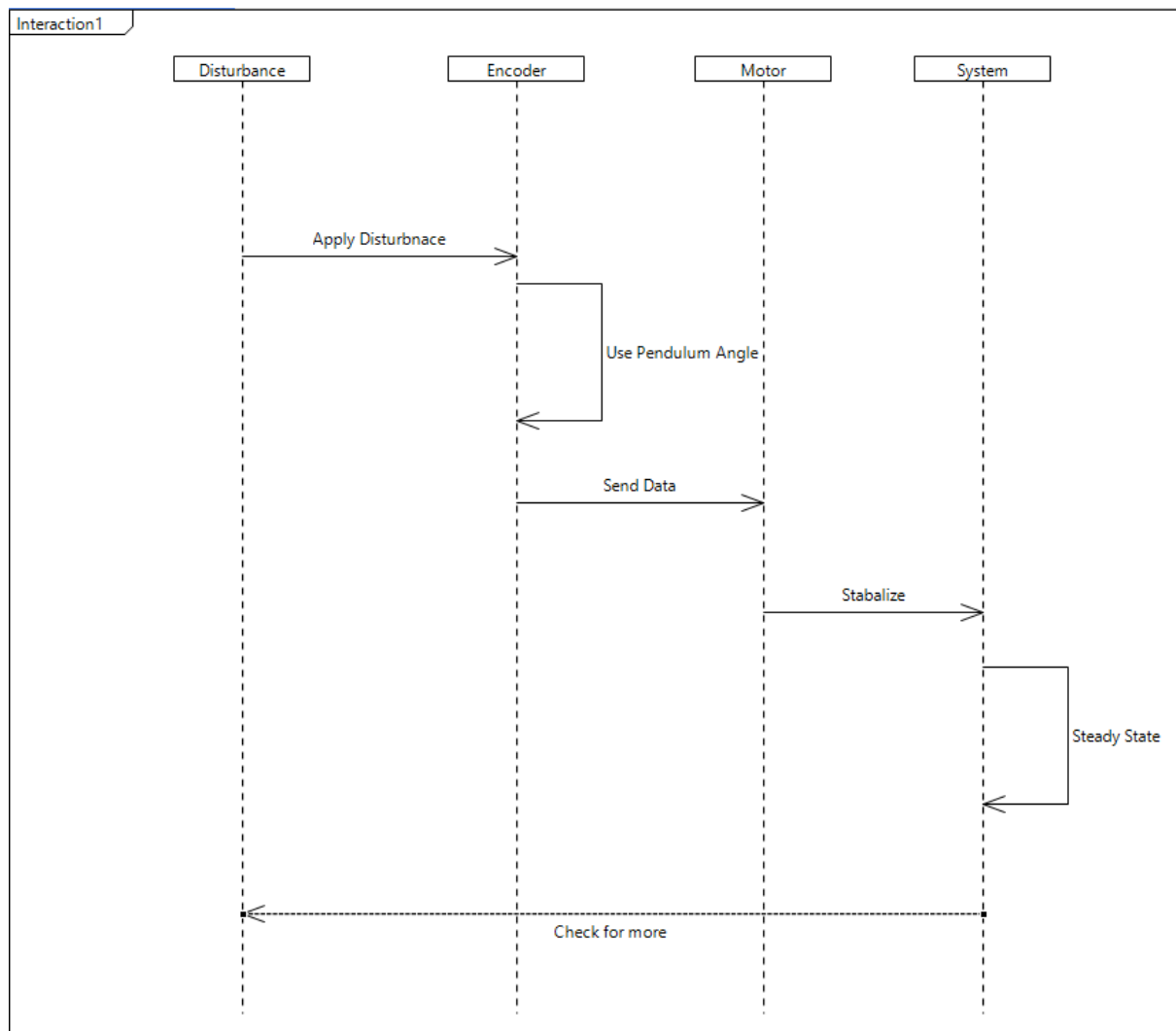


Figure 4.6 Sequence Diagram

**\*\*REFER TO APPENDIX I FOR ALL MBSE DIAGRAM\*\***

## **5. Failure Mode and Effect Analysis, FMEA**

Failure Mode and Effect Analysis (FMEA) is a structured iterative analysis method. The goal of FMEA is to assess products and systems for their reliability and safety. FMEA is not meant to make the perfect design, but a process to try to create the best possible design at early parts of the design to avoid major costly failures.

Some benefits of FMEA include:

- Enhancing design and manufacturing steps
- Reducing changes within the later stages of the design process
- Reducing customer complaints
- Reducing safety failures
- Reducing maintenance and warranty costs

Failures can be grouped into three categories of failures:

### **Physical Flaws**

- Overloading or Fatigue
- Corrosion
- Electrical Hazards

### **Process Errors**

- Errors in Design
- Analysis
- Manufacturing
- Maintenance
- Calculation Errors
- Wrong Assumptions

### Errors in Perspective or Attitude

- Overconfidence
- Indifference
- Arrogance
- Selfishness
- Judgement Errors
- Lack of Experience

### Steps

1. Identify **Failure Modes**, how a product can fail
2. Identify **Root Cause Analysis**, the cause of failure
3. The Effects of Failure
4. Recommended Actions, to prevent the failures

## Top-Down Functional Approach

The Top-Down Approach is used in the early parts of design, before parts have been identified. The goal of the Top-Down Functional Approach is to look for logic errors in the expected function and operation of a product. Failure mode for the whole system will be analyzed, then these failure modes will be traced down into the smaller subsystems.

The overall function of the Inverted Pendulum System is to keep the inverted pendulum upright. The following table lists the major subsystems needed for the Inverted Pendulum System. The table will also include the function of each subsystem, and why it is needed within the system.

Subsystem Name	Subsystem Function
Base	This subsystem will support and hold all the other subsystems together.
Motor	This subsystem will provide torque and rotational motion for the other subsystems.
Linkage	The subsystem that is connected from the motor and the load (the inverted pendulum)
Sensor	This subsystem will measure pendulum position parameters (angle and/or distance)

Table 5.1: Subsystem and Subsystem Functions

Some possible ways for the system to fail include:

- The inverted pendulum cannot be kept upright
- Start/Stop Button fails due to damage
- Motor burns out
- Mechanical parts fail due to wear and tear
- Sensor is not sensitive enough
- Sensor detects too much noise
- Power source is not strong enough to power the overall system



### Bottom-Up Structural Approach

The Bottom-Up Approach is used when specific parts or major assemblies have been designed. The goal of the Bottom-Up Structural Approach is to look for physical errors in the design/manufacturing process. Failure mode will start at each of the parts, and then the failure effects will be followed “up” to see how it will impact the whole product.

The following table includes potential failure modes of each individual subsystem, and the potential failure effects on the whole system.

Process Step/Input	Potential Failure Mode	Potential Failure Effects
Base Subsystem	Base is not sturdy enough	The whole system shifts around
	Base is not sturdy enough	The whole system tips over
	Base is not sturdy enough	The base cracks/collapses due to system weight
Arm	The Arm is not sturdy enough	Arm snaps, and load drops
	Arm is not attached on properly to the pendulum	Inverted Pendulum slides up/down on the system while still attached to the system
	Arm is not attached on properly to the pendulum (while System is not running)	Inverted Pendulum falls off the system
	Arm is not attached on properly to the pendulum (while System is	Inverted Pendulum detaches off the system while system is running

	running)	
Sensor Subsystem	Sensor is not sensitive enough	Measurement of the pendulum is not accurate, and the PID/Pole Placement will provide inaccurate corrective adjustments
	Sensor is too sensitive	May pick up signal noise and provide inaccurate corrective adjustments
Rotating Subsystem	Motor spins too fast	Motor burns out
	Motor spins too fast	Arm rotates too fast causing the pendulum to overshoot
	Motor spins too fast	Connecting wires get tangled up
	Motor does not provide enough torque	System cannot correct large errors due to large disturbances

Table 5.2: Potential Failure Modes and Effects Analysis for Subsystems

## Severity Ranking (SR)

The following Table from Engineering Robust Designs will be used to rank the severity of the failure mode effects. The scale will be from Ranks 1 to 10, where Rank 10 Severities are categorized as “Hazardous, without warning”, and Rank 1 Severities would have no effect on the system.

**Table 5-1 A Ranking System for the Severity of Effects for a Design FMEA**

<b>Effect</b>	<b>Description of Severity</b>	<b>Rank</b>
Hazardous, without warning	Failure affects safe product operation or involves noncompliance with government regulation without warning	10
Hazardous, with warning	Failure affects safe product operation or involves noncompliance with government regulation with warning	9
Very high	Product is inoperable with loss of primary function	8
High	Product is operable, but at a reduced level of performance	7
Moderate	Product is operable, but comfort or convenience items are inoperable	6
Low	Product is operable, but comfort or convenience items operate at a reduced level of performance	5
Very low	Fit and finish or squeak and rattle—items do not conform; most customers notice defect	4
Minor	Fit and finish or squeak and rattle—items do not conform; average customers notice defect	3
Very minor	Fit and finish or squeak and rattle—items do not conform; discriminating customers notice defect	2
None	No effect	1

Figure 5.1: Ranking System for the Severity of Effects for a Design FMEA [3]

The following Table includes the potential failure effects of the inverted pendulum system with a description of severity. These potential failures will also be ranked according to the Ranking System for the Severity of Effects for a Design FMEA (Figure 5.1).

Potential Failure Effects	Description of Severity	Rank
The whole system shifts around	<b>Hazardous, with Warning. Impact Hazard.</b> The whole system shifting could move and impact people or objects. This affects safe product operations or involves noncompliance with government regulations with warning	9
The whole system tips over	<b>Hazardous, with Warning. Impact Hazard.</b> The whole system tipping over could move and impact people or objects. This affects safe product operations or involves noncompliance with government regulations with warning	9
The base cracks/collapses due to system weight	<b>High. Entrapment Hazard.</b> If the base cracks/collapses, a person's body could get trapped or pinched. The whole system will remain operable, but at a reduced level of performance	7
Arm snaps, and load drops	<b>Very High. Entrapment Hazard.</b> If the arm breaks, the inverted pendulum will fall off of the system, and a person could get crushed. The whole system will become inoperable, as there is nothing to balance. This will prevent the system from performing the primary function.	8
Inverted Pendulum slides up/down on the	<b>High. Impact Hazard.</b> If the inverted pendulum slides up/down, the pendulum could strike a person or an object. The	7

system while still attached to the system	whole system will remain operable, but at a reduced level of performance	
Inverted Pendulum falls off the system	<b>Very High. Entrapment Hazard.</b> If the inverted pendulum falls off of the system, a person's body could get crushed or trapped. The whole system will become inoperable, as there is nothing to balance. This will prevent the system from performing the primary function.	8
Inverted Pendulum detaches off the system while system is running	<b>Hazardous, without warning. Ejection Hazard.</b> If the inverted pendulum detaches from the system while it is running, the inverted pendulum may fly outwards at a high velocity, at a random direction. This could harm nearby operators or the nearby setting. This failure will affect safe product operations or involve noncompliance with government regulations without warning.	10
Measurement of the pendulum is not accurate, and the PID/Pole Placement will provide inaccurate corrective adjustments	<b>High.</b> System is operable, but with inaccurate corrective adjustments, the system will perform at a reduced level of performance.	7
May pick up signal noise and provide inaccurate corrective	<b>High.</b> If signal noise is picked up, inaccurate values will be sent, and inaccurate corrective adjustments will be made. The system is operable, but at a redacted level.	7

adjustments		
Motor burns out	<b>Very High. Contact Hazard.</b> If the motor burns out, it could result in a hot surface. The whole system will become inoperable, as the system cannot perform the primary function.	8
Arm rotates too fast causing the pendulum to overshoot	<b>High. Impact Hazard.</b> System is operable, but with the arm is overshooting, the system could strike a person or an object. The system will perform at a reduced level of performance.	7
Connecting Wires gets Tangled up	<b>Hazardous, with Warning. Entrapment/Entanglement Hazard.</b> While rotating, the connecting wires can get tangled up and get caught within an object or person. This affects safe product operations or involves noncompliance with government regulations with warning.	9
System cannot correct large errors due to large disturbances	<b>High.</b> System is operable, but the system not being able to correct large errors would be a reduced level of performance.	7

Table 5.3: Unsorted Table of Failure Effects with Description of Severity with Ranks

The Table of Failure Effects with Description of Severity with Ranks (Table 5.3) will be sorted from highest rank to lowest rank, as the most serious effects take precedence.

Potential Failure Effects	Description of Severity	Rank
Inverted Pendulum detaches off the system while system is running	<b>Hazardous, without warning. Ejection Hazard.</b> If the inverted pendulum detaches from the system while it is running, the inverted pendulum may fly outwards at a high velocity, at a random direction. This could harm nearby operators or the nearby setting. This failure will affect safe product operations or involve noncompliance with government regulations without warning.	10
The whole system shifts around	<b>Hazardous, with Warning. Impact Hazard.</b> The whole system shifting could move and impact people or objects. This affects safe product operations or involves noncompliance with government regulations with warning	9
The whole system tips over	<b>Hazardous, with Warning. Impact Hazard.</b> The whole system tipping over could move and impact people or objects. This affects safe product operations or involves noncompliance with government regulations with warning	9
Connecting Wires gets Tangled up	<b>Hazardous, with Warning. Entrapment/Entanglement Hazard.</b> While rotating, the connecting wires can get tangled up and get caught within an object or person. This affects safe product operations or involves noncompliance with government regulations with warning.	9

Arm snaps, and load drops	<b>Very High. Entrapment Hazard.</b> If the arm breaks, the inverted pendulum will fall off of the system, and a person could get crushed. The whole system will become inoperable, as there is nothing to balance. This will prevent the system from performing the primary function.	8
Inverted Pendulum falls off the system	<b>Very High. Entrapment Hazard.</b> If the inverted pendulum falls off of the system, a person's body could get crushed or trapped. The whole system will become inoperable, as there is nothing to balance. This will prevent the system from performing the primary function.	8
Motor burns out	<b>Very High. Contact Hazard.</b> If the motor burns out, it could result in a hot surface. The whole system will become inoperable, as the system cannot perform the primary function.	8
The base cracks/collapses due to system weight	<b>High. Entrapment Hazard.</b> If the base cracks/collapses, a person's body could get trapped or pinched. The whole system will remain operable, but at a reduced level of performance	7
Inverted Pendulum slides up/down on the system while still attached to the system	<b>High. Impact Hazard.</b> If the inverted pendulum slides up/down, the pendulum could strike a person or an object. The whole system will remain operable, but at a reduced level of performance	7
Measurement of the	<b>High.</b> System is operable, but with inaccurate corrective	7



pendulum is not accurate, and the PID/Pole Placement will provide inaccurate corrective adjustments	adjustments, the system will perform at a reduced level of performance.	
May pick up signal noise and provide inaccurate corrective adjustments	<b>High.</b> If signal noise is picked up, inaccurate values will be sent, and inaccurate corrective adjustments will be made. The system is operable, but at a reduced level.	7
Arm rotates too fast causing the pendulum to overshoot	<b>High. Impact Hazard.</b> System is operable, but with the arm is overshooting, the system could strike a person or an object. The system will perform at a reduced level of performance.	7
System cannot correct large errors due to large disturbances	<b>High.</b> System is operable, but the system not being able to correct large errors would be a reduced level of performance.	7

Table 5.4: Table of Failure Effects with Description of Severity with Ranks, Sorted by Highest Rankings to Lowest Ranking

## Occurrence Rating (OR)

Occurrence Rating is the likelihood of the failure occurring. The following Table “A Ranking System for the Occurrence of Failure in a Design FMEA” will be used to rank the Occurrence Rating for the potential failures. Rank 10 suggests that the probability of failure is “almost inevitable”, and Rank 1 suggests that probability of failure is “unlikely”.

**Table 5-2 A Ranking System for the Occurrence of Failure in a Design FMEA**

<b>Probability</b>	<b>Rate</b>	<b>Rank</b>
Very high: Failure almost inevitable	> 1 in 2	10
Hazardous, with warning	1 in 3	9
High: Repeated failures	1 in 8	8
	1 in 20	7
Moderate: Occasional failures	1 in 80	6
Low to very low	1 in 400	5
	1 in 2000	4
Low: Relatively few failures	1 in 15,000	3
Very minor failures	1 in 150,000	2
Remote: Failure unlikely	< 1 in 1,500,000	1

Figure 5.2: A Ranking System for the Occurrence of Failure in a Design FMEA [3]

The following Table includes the Potential Failures Effects and their potential causes. The Occurrence Rating will be determined using the Ranking System for the Occurrence of Failure in a Design FMEA (Figure 5.2).

Potential Failure Effects	Potential Causes	Occurrence Rating
Inverted Pendulum detaches off the system while system is running	Connection between pendulum load and arm is not tighten properly	<b>2, Very Minor Failures.</b> The Chance Inverted Pendulum detaching off the system while the system is running is very minor (about 1 in 150,000).
The whole system shifts around	Friction between the base and the ground is not high enough  Force exerted by system correction is too large	<b>10, Very High: Failure is almost inevitable.</b> The whole system is currently shifting around every time it is turned on.
The whole system tips over	Friction between the base and the ground is not high enough  Excessive Force is applied onto system	<b>1, Remote: Failure Unlikely.</b> The chance of the whole system tipping over is very unlikely. There are supports on the leg that prevents the system from completely tipping over when running.
Connecting Wires gets	System rotates around too many	<b>7, High: Repeated Failures.</b>

Tangled up	times  Connecting wires are not clamped down onto the	The rate of where the connecting wires get tangled up is about 1 in 20, while the system ran.
Arm snaps, and load drops	Repeated stress cycles  Excessive Force applied onto system	<b>2, Very Minor Failures.</b> The Chance of the Arm snapping off is very minor, and will take many operation cycles (about 1 in 150,000).
Inverted Pendulum falls off the system (static)	Screw is not tighten properly  Screw gets loose after repeated stress cycles	<b>1, Remote: Failure Unlikely.</b>  The chance of the inverted pendulum falling off the system while the system remains static is very unlikely.
Motor burns out	Too much voltage applied  Overload, too much load  Motor is turned on/off repeatedly	<b>1, Remote: Failure Unlikely.</b>  The motor, Nema 23, was specifically selected to support the designated pendulum load with, and additional 50-100g mass at the end of the pendulum. The main causes of increased failures would be if the motor was not operated

		properly.
The base cracks/collapses due to system weight	System is too heavy Base material quality is poor Repeated stress cycles	<b>1, Remote: Failure Unlikely.</b> The base is built very sturdy enough, and will support the system. Chance of the system being too heavy for the base is very low. However the base may crack over many repeated stress cycles. (<1 in 1,500,000).
Inverted Pendulum slides up/down on the system while still attached to the system	Connecting Screw between pendulum and arm linkage is not tighten properly Connecting Screw between pendulum and arm linkage gets loose	<b>2, Very Minor Failures.</b> The Chance Inverted Pendulum sliding up/down the system, while still attached to the system is very minor (about 1 in 150,000).
Measurement of the pendulum is not accurate, and the PID/Pole Placement will provide inaccurate corrective adjustments	Sensor is not working properly (not sensitive enough)	<b>2, Very Minor Failures.</b> The rate of where the pendulum measurement is not accurate due to the sensor not working properly is about 1 in 150,000.
May pick up signal noise and provide inaccurate	Setting where the system is contains too much signal noise	<b>6, Moderate: Occasional Failures.</b> A signal filter will be

corrective adjustments	System Noise Filter is not working properly	applied to filter out the signals, but occasional loud noise spikes may be picked up and cause inaccurate corrective adjustments.
Arm does not rotate fast enough to keep the pendulum upright	Motor does not provide enough torque	<b>1, Remote: Failure Unlikely.</b> The motor, Nema 23, was specifically selected to provide enough torque to move the designated pendulum load with, and additional 50-100g mass at the end of the pendulum.
System cannot correct large errors due to large disturbances	Motor is providing too much torque PID/Pole placement is not calibrated properly	<b>5, Low Failures.</b> The rate of where the system cannot correct large errors is about 1 in 400.

Table 5.5: Table of Potential Failure Effects and Potential Causes

## Detection Ranking (DR)

The Detection Ranking measures the likelihood of the control system detecting the failure before it occurs. This can be a probabilistic or qualitative measurement. The Ranking will be determined using the “Ranking System for the Detection of a Cause of Failure or Failure Mode in a Design FMEA”. Rank 10 will be for systems that are “absolutely uncertain” when trying to detect the failure, while rank 1 will mean a system would be able to detect a failure almost certainly.

**Table 5-3** A Ranking System for the Detection of a Cause of Failure or Failure Mode in a Design FMEA

Detection	Description of Likelihood	Rank
Absolutely uncertain	Uncertain that a design control will detect a potential cause of failure or subsequent failure mode; or there is no design control	10
Very remote	Very remote chance that a design control will detect a potential cause of failure or subsequent failure mode	9
Remote	Remote chance that a design control will detect a potential cause of failure or subsequent failure mode	8
Very low	Very low chance that a design control will detect a potential cause of failure or subsequent failure mode	7
Low	Low chance that a design control will detect a potential cause of failure or subsequent failure mode	6
Moderate	Moderate chance that a design control will detect a potential cause of failure or subsequent failure mode	5
Moderately high	Moderately high chance that a design control will detect a potential cause of failure or subsequent failure mode	4
High	High chance that a design control will detect a potential cause of failure or subsequent failure mode	3
Very high	Very high chance that a design control will detect a potential cause of failure or subsequent failure mode	2
Almost certain	Almost certain that a design control will detect a potential cause of failure or subsequent failure mode	1

Figure 5.3: A Ranking System for the Detection of a Cause of Failure or Failure Mode in a Design FMEA [3]

The following table includes the potential failure modes, the potential causes, and existing controls and procedures (Inspection and Test) to prevent the Failure Modes and/or the Causes.

Potential Failure Effects	Potential Causes	Current Controls/Procedures to Prevent Failure	Detection
Inverted Pendulum detaches off the system while system is running	Connection Screw between pendulum load and arm is not tighten properly	There is a Moderately High chance that a Visual Inspection/Physical Test will be able to successfully detect a loose screw between the pendulum load and the arm system.	Moderately High, 4
The whole system shifts around	Friction between the base and the ground is not high enough Force exerted by system correction is too large	There is currently no design controls to detect system shifting around when it is running	Absolutely Uncertain, 10
The whole system tips over	Friction between the base and the ground is not high enough Excessive Force is applied onto system	There is a very moderate chance where a visual check will be able to prevent failure. If the system is shifting too much, the system	Moderate, 5



		can be shut down before the whole system tips over.	
Connecting Wires gets Tangled up	System rotates around too many times  Connecting wires are not clamped down onto the	There is a low chance, where the wire cable clamps will prevent the wires from tangling up. The wire cable clamps are used to keep the wiring as neat as possible, and they act as a way to prevent the wires from tangling up.	Low, 6
Arm snaps, and load drops	Repeated stress cycles  Excessive Force applied onto system	There is a remote chance that the arm snapping will be detected by visual inspection	Remote, 8
Inverted Pendulum falls off the system (static)	Screw is not tighten properly  Screw gets loose after repeated stress cycles	There is a Moderately High chance that a Visual Inspection/Physical Test will be able to successfully detect a loose screw.	Moderately High, 4
Motor burns out	Too much voltage applied  Overload, too much load  Motor is turned on/off repeatedly	There is a remote chance that the motor burn out will be detected by visual inspection	Remote, 8

The base cracks/collapses due to system weight	System is too heavy Base material quality is poor Repeated stress cycles	There is a moderate chance that a visual inspection will be able to detect a small crack, and prevent larger failures as in collapses.	Moderate, 5
Inverted Pendulum slides up/down on the system while still attached to the system	Connecting Screw between pendulum and arm linkage is not tighten properly Connecting Screw between pendulum and arm linkage gets loose	There is a Moderately High chance that a Visual Inspection/Physical Test will be able to successfully detect a loose screw.	Moderately High, 4
Measurement of the pendulum is not accurate, and the PID/Pole Placement will provide inaccurate corrective adjustments	Sensor is not working properly (not sensitive enough)	There is a moderately high chance to detect if the sensor is not working properly, using visual inspection.	Moderately high, 4
May pick up signal noise and provide inaccurate corrective adjustments	Setting where the system is contains too much signal noise System Noise Filter is not working properly	There is a very low chance that noise will be detected using visual inspection.	Very Low, 7
Arm does not rotate fast enough to keep the	Motor does not provide enough torque	There is a moderately high chance to detect if the sensor	Moderately high, 4

pendulum upright		is not working properly, using visual inspection.	
System cannot correct large errors due to large disturbances	Motor is providing too much torque  PID/Pole placement is not calibrated properly	There is a moderately high chance to detect if the sensor is not working properly, using visual inspection.	Moderately high, 4

Table 5.6: Table of Potential Failure Effects, Potential Causes, Current Procedures/Systems to Prevent Failures, and Detection Ranking

**Risk Priority Number, RPN**

Sample Calculation of RPN, for Failure Mode, the whole system shifts around

Severity Ranking, SR

Occurrence Ranking, OR

Detection Ranking, DR

Risk Priority Number, RPN

$$RPN = SO * OR * DR$$

$$RPN = 9 * 10 * 10$$

$$RPN = 900$$

The Risk Priority Number for when the whole system shifts around is 900.

The following table includes the calculated RPN Values with the original Severity Ranking, Occurrence Rating, and Detection Ranking.

Old				
Potential Failure Effects	Severity	Occurrence	Detection	RPN
Inverted Pendulum detaches off the system while system is running	10	2	4	80
The whole system shifts around	9	10	10	900
The whole system tips over	9	1	5	45
Connecting Wires gets Tangled up	9	7	6	378
Arm snaps, and load drops	8	2	8	128
Inverted Pendulum falls off the system	8	1	4	32
Motor burns out	8	1	8	64
The base cracks/collapses due to system weight	7	1	5	35

Inverted Pendulum slides up/down on the system while still attached to the system	7	2	4	56
Measurement of the pendulum is not accurate, and the PID/Pole Placement will provide inaccurate corrective adjustments	7	2	4	56
May pick up signal noise and provide inaccurate corrective adjustments	7	6	7	294
Arm rotates too fast causing the pendulum to overshoot	7	1	4	28

Table 5.7: Calculated Risk Priority Number, RPN, for all Potential Failure Effects of the  
Inverted Pendulum System

The following table, Table 5.8, includes the Recommended Actions, and the team responsible for the Recommended Actions. It also includes the new Occurrence Ratings and Detection Rankings as a result of implementing the Recommended Actions. The new RPN value would also be calculated due to the change for the new Occurrence Ratings and Detection Rankings.

New					
<b>Actions Recommended</b>	<b>Resp.</b>	<b>Severity</b>	<b>Occurrence</b>	<b>Detection</b>	<b>RPN</b>
Increase regular visual maintenance inspection quantity, to improve detection rating	Maintenance Team	10	2	3	60
Introduce Rubber Padding on Base Support to reduce the Occurrence of the whole system shifting around	Mechanical Design Team	9	5	10	450
Introduce Rubber Padding on Base Support to reduce the Occurrence of the tipping over due to excess force	Mechanical Design Team	9	1	5	45

Introduce more wire cable clamps, so that the wires will have less places to move. This will lower the occurrence rating of this failure of happening	Mechanical Design Team	9	4	6	216
Increase regular visual maintenance inspection quantity, to improve detection rating	Mechanical Design Team	8	2	5	80
Increase regular visual maintenance inspection quantity, to look for loose screws. This will improve detection rating	Mechanical Design Team	8	1	3	24
Increase regular visual maintenance inspection quantity, to look at motor health. This will improve detection rating	Mechanical Design Team	8	1	5	40



<p>Increase regular visual maintenance inspection quantity, to look at motor health. This will improve detection rating</p>	<p>Mechanical Design Team</p>	7	1	4	28
<p>Increase regular visual maintenance inspection quantity, to look for loose screws. This will improve detection rating</p>	<p>Mechanical Design Team</p>	7	2	3	42
<p>Increase regular visual maintenance inspection quantity, to make sure the sensor is working properly. This will improve detection rating</p>	<p>Mechanical Design Team</p>	7	2	2	28
<p>Introduce a signal filter to filter out signal noise. This will improve the occurrence rating.</p>	<p>Coding Team</p>	7	3	7	147

Increase regular visual maintenance inspection quantity, to make sure the sensor is working properly. This will improve detection rating	Mechanical Design Team	7	1	3	21
Make sure to calibrate the PID/Pole Placement Parameters to allow for large disturbance corrections	Coding Team	7	2	4	56

Table 5.8: Recommended Actions and New Calculated Risk Priority Number for all Potential Failure Effects

## Ishikawa (Fishbone) Diagram

The Ishikawa Diagram is a tool used to help lay out the components of the Inverted Pendulum System. The Spine is the primary System, the Inverted Pendulum. The Ribs include the subsystems of the Inverted Pendulum System. The Secondary Ribs consists of the potential failures that may happen for each of the subsystems.

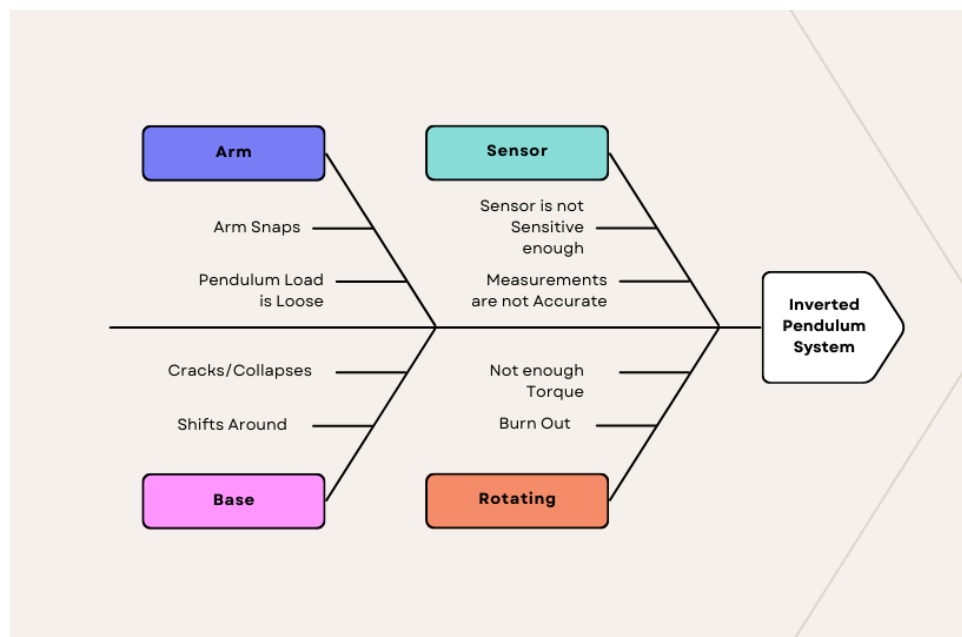


Figure 5.4: Ishikawa (Fishbone) Diagram for Inverted Pendulum System

## **FMEA Conclusion**

The goal of FMEA is not to create a perfect design on the first iteration, but to work towards designing the best possible design. Through the FMEA process, the team will try to identify as many failure modes as early as possible to redesign the system and avoid failure modes at a lower expense. If a failure mode is not identified within the FMEA process, then this will result in the failure mode not being able to be addressed. The highest Risk Failure Mode was when the whole system shifted when the system ran with a RPN of 900. By introducing rubber paddings on the base supports, the Occurrence Rating dropped a significant amount. In addition, introducing Preventative Maintenance to increase the amount of visual inspections was a critical way to improve the Detection Ranking value of many of the Failure Modes. As a result, this reduces the Risk Priority Number.

## **6. Concurrency**

Concurrent Design is when different teams are working together at the same time. There are many advantages of Concurrent Design. Some examples include:

- Reduced design times
- Faster time-to-market
- Reduced design problems
- Increased reliability
- Increased quality

## **Organization**

Understanding organization was important to concurrent design, it was important to the design process team members stayed organized throughout the process. By organizing the work split throughout the group, members were able to work where they felt more comfortable, creating a smooth process. Having accountability and version control helped the team members stay organised in their process and work. This highlights the importance of keeping this Pillar a pivotal part of the design process.

## **Communication**

The team mainly used Discord and during the in person labs to host check up meetings, and always as a way to get a hold of each other. These check up meetings were used as a way to make sure everyone understood what had to be done for the next meeting. Any questions that one would have could be answered by any of the other members would be done within this period as well. In addition, these meetings were very helpful, to understand what the other members were working on as well. The report was done on Google Docs. This allowed

multiple members to work on the report at the same time. A Github was created as well for the team to have access to the most updated code for the Inverted Pendulum System.

### Design Problem

The design problem is broken down into 4 separate areas, Product Characteristics, Functional Requirements, Constraints, and Performance Metrics.

#### Product Characteristics

The system must be safe
The system must be durable
The system must be able to be transported
The system must be reliable

Table 7.1: Table of Product Characteristics of Inverted Pendulum System

#### Functional Requirement

System must be able to stabilize the pendulum when a small disturbance is applied to mass
System must be able to stabilize the pendulum when a large disturbance is applied to mass
System must use a control method to stabilize the pendulum

Table 7.2: Table of Functional Requirements of Inverted Pendulum System

### Constraints

Pendulum Length must be between 40-50cm
Pendulum mass must be 50-100 g
Pendulum must use PID and Pole Placement controller methods

Table 7.3: Table of Constraints of Inverted Pendulum System

### Performance Metrics

System must be able to stabilize itself within 5 seconds	$\frac{(t-5)}{5} * 100\%$
Safe	$P = \frac{5-stars}{5}$

Table 7.4: Table of Performance Metrics of the Inverted Pendulum System

## Product Development

The Inverted Pendulum System has had many iterations since the first iteration. After analysing our system with FMEA, many failure modes have been identified, and changes were implemented to prevent known problems. The first iteration of the Inverted Pendulum System was made, and then brought to an in person meeting with the team. The system has a test run, and multiple features were suggested to improve the future iterations. One thing the team noticed was that the system shifted around a lot when it was running. A solution to this potential problem was to include rubber padding on the base supports to prevent the system from shifting around when it ran.

## Responsibility Table

Names	List of Responsibilities
Adam Di Benedetto	Physical System Design + Hardware Build CAD modelling/drawings/renders HMI + Control Panel
Adam Moore	Arduino code Performance testing (graphs) Theory section Code description
Waleed Idrees	MBSE (All Diagrams) Report
Jacky Su	FMEA <ul style="list-style-type: none"> <li>• Severity Ranking</li> <li>• Occurrence Rating</li> <li>• Detection Ranking</li> <li>• Risk Priority Number, RPN</li> </ul> Concurrency Pillars
Hamayoon Ashraf	Report Intro + Abstract

Table 7.1: Team Responsibility Table



## 7. Modelling and Design

### Mechanical Design

The final render of the inverted pendulum system can be seen below through realistic modelling in Fusion360. This final render/design is composed of 7 different unique components to which had to be designed and customised with CAD drawings. The final design incorporates 3 panel stands, 5 wedges, 1 pendulum arm, 1 arm support, 1 custom encoder mount, 1 custom stepper motor mount and 1 wire guide. The encoder and stepper motor objects within the render and assembly drawings DO NOT REPRESENT OUR WORK and were selected from grabCAD as a default dimensioned product matching our desired component. It should also be noted that any dimensions listed in drawings are metric in millimetres.



Figure 7.1: Fusion 360 Drawing of Inverted Pendulum Assembly Render

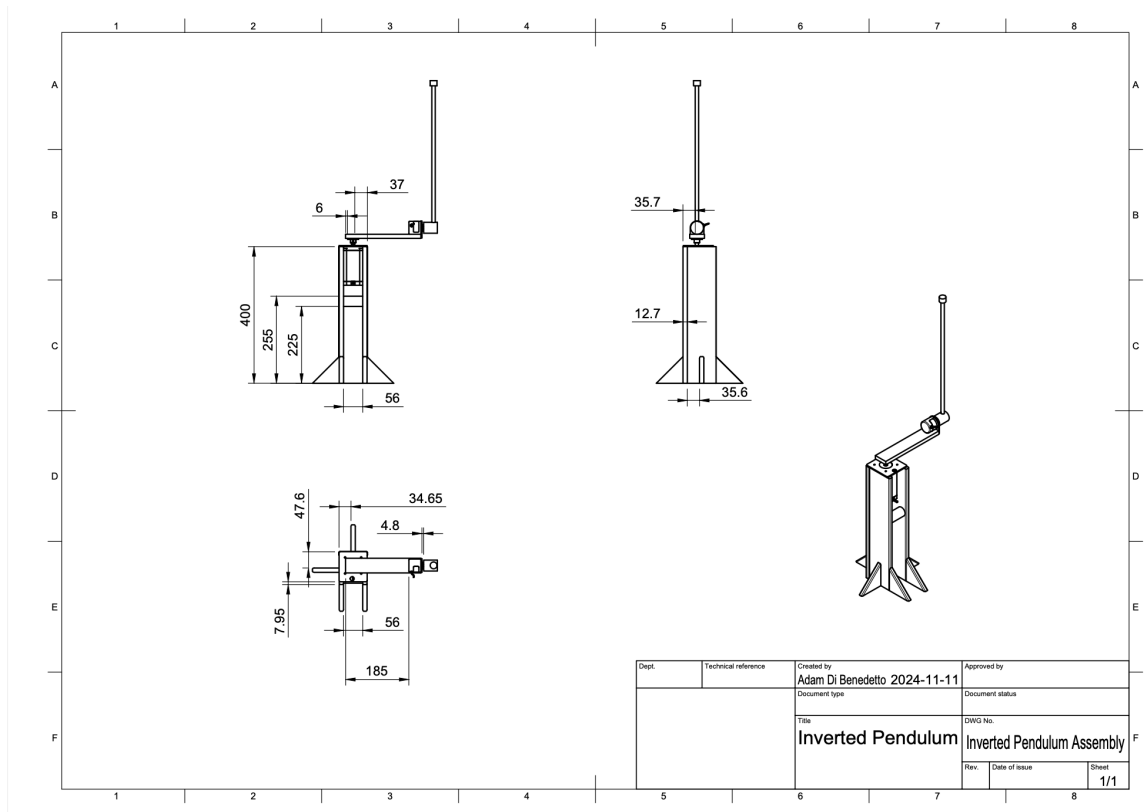


Figure 7.2: Fusion 360 Drawing of Inverted Pendulum Assembly Drawing

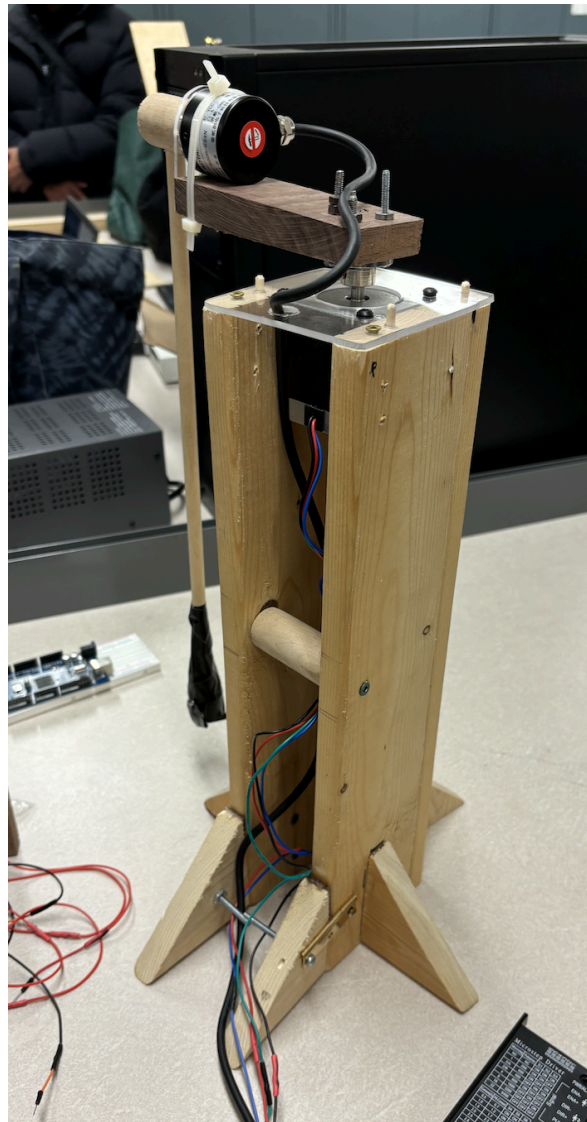


Figure 7.3: Fusion 360 Drawing of Inverted Pendulum Assembly Drawing

The part drawings for these individual components can be seen in the **Appendices**. Moving onto the control panel, the control panel consists of 2 panel supports, 1 custom control panel sheet, 1 toggle switch, 2 push buttons and 1 LCD screen. The pushbuttons, toggle switch and LCD screen objects within the render and assembly drawings of the control panel seen below DO NOT REPRESENT OUR WORK and were selected from grabCAD as a default dimensioned product matching our desired component. It should also be noted that any dimensions listed in drawings are metric in millimetres.

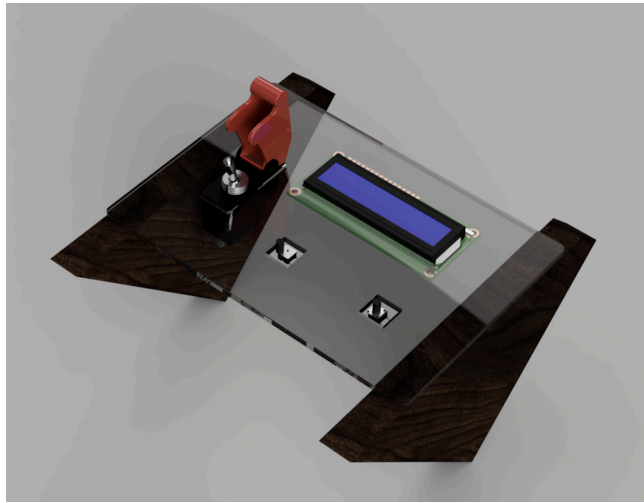


Figure 7.4: Fusion 360 Drawing of Inverted Pendulum Control Panel Assembly Render

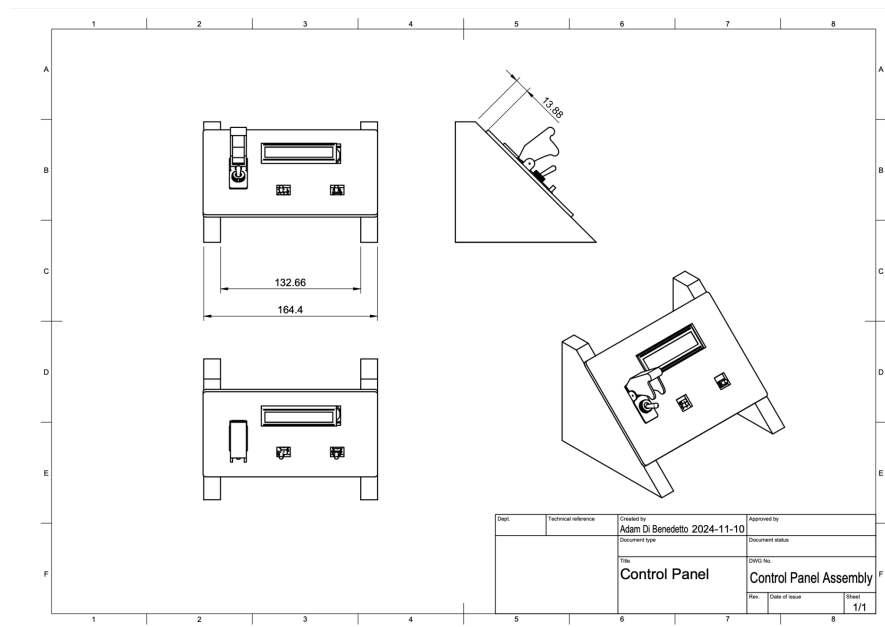


Figure 7.5: Fusion 360 Drawing of Inverted Pendulum Control Panel Assembly Drawing

Every custom component seen in the above assembly of the control panel can be found in their individual part drawings seen in the **Appendices**. In continuation, the explanation and concept behind the human machine interface is described in a later section, **HMI** where a justification is made for each component, placement, dimension and function.

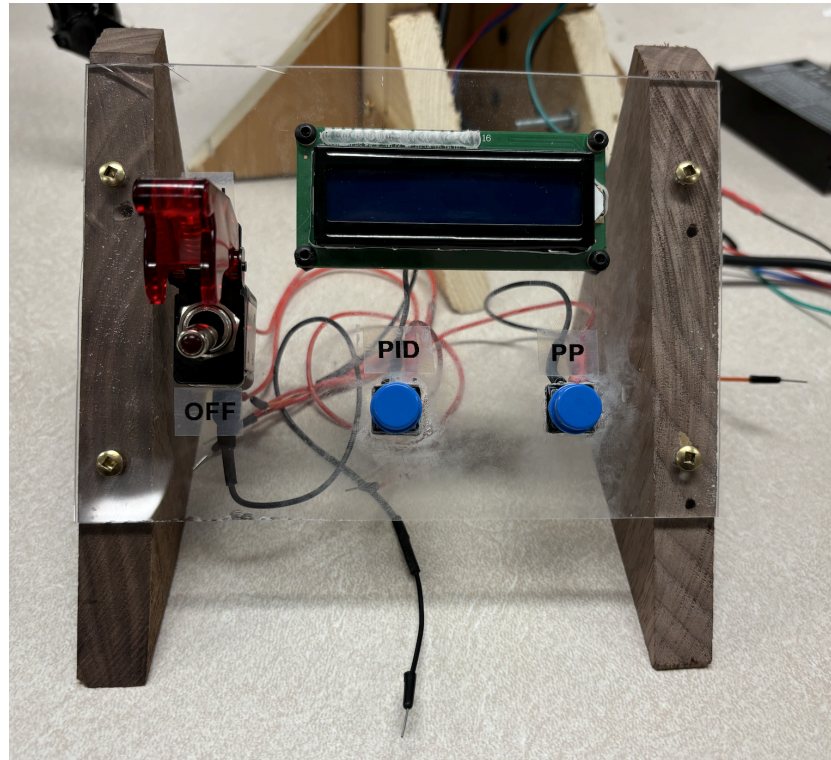


Figure 7.6: Fusion 360 Drawing of Inverted Pendulum Assembly Drawing

If a simulation of an inverted pendulum system is desired, one can refer to project 1 where a modelling and simulation are showcased within a python/pygame script executed.

## Electrical Design

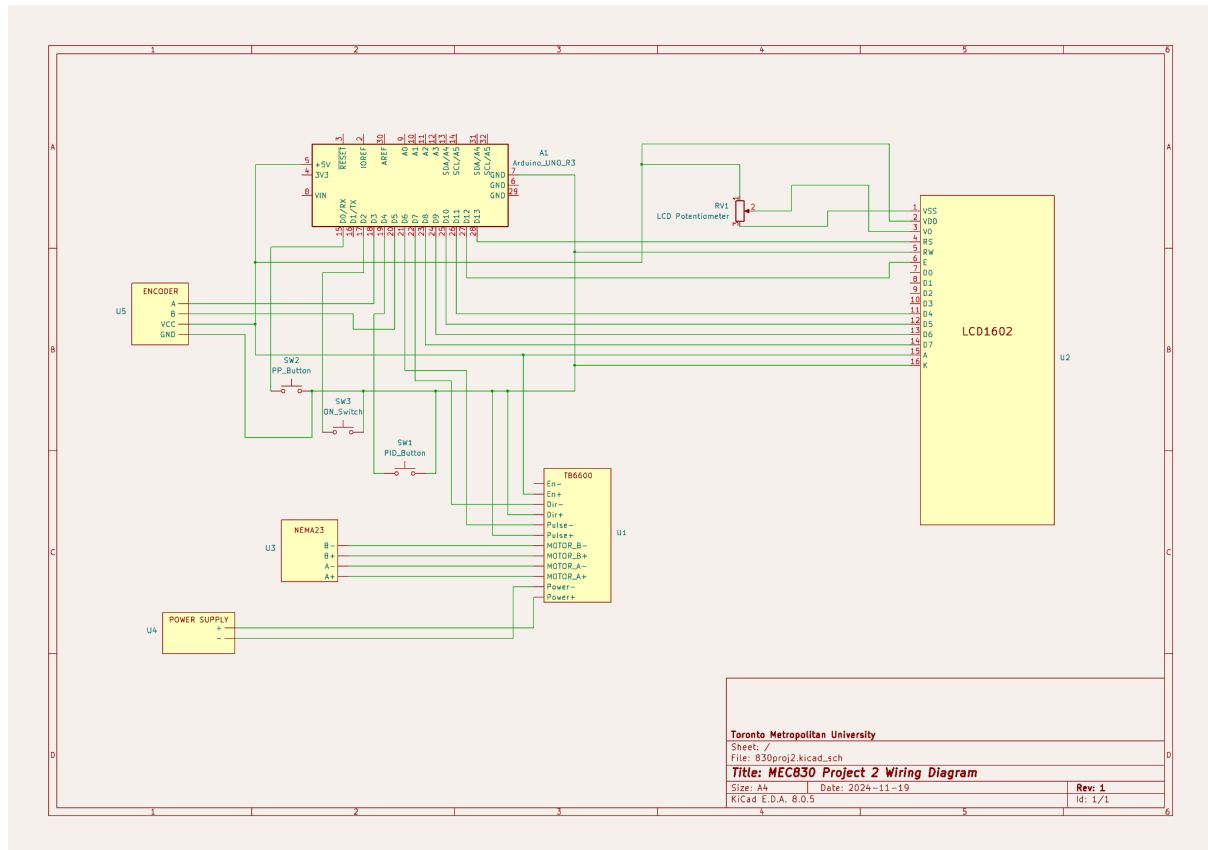


Figure 7.7: KiCAD Wiring Diagram of Inverted Pendulum System

## 8. HMI - Control Panel

To begin with, the primary objective of designing and constructing a control panel for this mechanism was to follow the 3 basic golden rules. The control panel design should place users in control, reduce the user's memory load and make the interface consistent. One of the obvious concepts to implement is safety in which our design must maintain adequate safety considerations to protect the user and any surrounding viewers from harm. Connecting both, the speed of the machine needs to be studied in accordance with the reaction to inputs as if the system needs to shut down immediately, how can we implement an input to allow this to happen with as little delay as possible?

### Safety Considerations

Beginning with accidental activation of the system, which can prevent any further damage to the system or danger to an unexpected user, we can implement a simple cover or physical safeguard over the main control to mitigate any accidental activations. In other words, if the system was turned on, the user's intention was to turn on the mechanism and there was no occurrence of an accident. The simplest example of this is a cover for a toggle switch as seen in the below figure.



Figure 8.1: Toggle Switch with safety cover [1]

Furthermore, we can implement some visual or audio cues for emergency situations. This can be in the form of an active or passive/active buzzer which can emit a loud repeated sound that can imply an urgency in the situation. A great example of implementing this in our inverted

pendulum system could be to emit the buzzer sound or a loud sound when the system is unstable or unbalanced, then the buzzer would turn off or stop emitting sound when the system is stable or the pendulum arm is within a certain range of angle that isn't of any danger. This allows the user to have a heads up as any minor disturbance in the system would alert the user to, let's say, push the classic big red button as an emergency stop to shut off the mechanism. Considering that the average conscious reaction which would incorporate cognitive processing takes 0.15 seconds, this concept should be sufficient, though a more accurate analysis would be done when the mechanism is built.

Leading to the inclusion of an emergency stop to physically cut power, we can implement an emergency stop that is located in series with the power supply unit or any source for the driver/stepper motor as if those are stopped, there should be no physical danger.



Figure 8.2: Emergency Stop button [5]

Another major safety topic is that the design should be accessible to all users of all potential dimensions. A quick way to overcome this issue is to maintain a height or position of controls near the ground which would mean height and wingspan/reach would not be a factor from a vertical perspective.



## Maintainability

To begin with, our design/system must be reliable. It must be reliable to accurately depict a free-moving pendulum that can be simulated physically without any parts breaking down. Additionally, in case a failure were to occur in some section of the design, the system was built in a way where core sections could be removed.

The control panel had no soldered wires to and from the Arduino so if any failure were to occur between the controls and the Arduino board/driver, the wires could easily be swapped out. Additionally, the controls on the control panel are not glued in beside the push buttons. The LCD screen, toggle switch and emergency stop are all non-permanent joints secured by fasteners as they can easily be swapped with any spares if any failure were to occur.

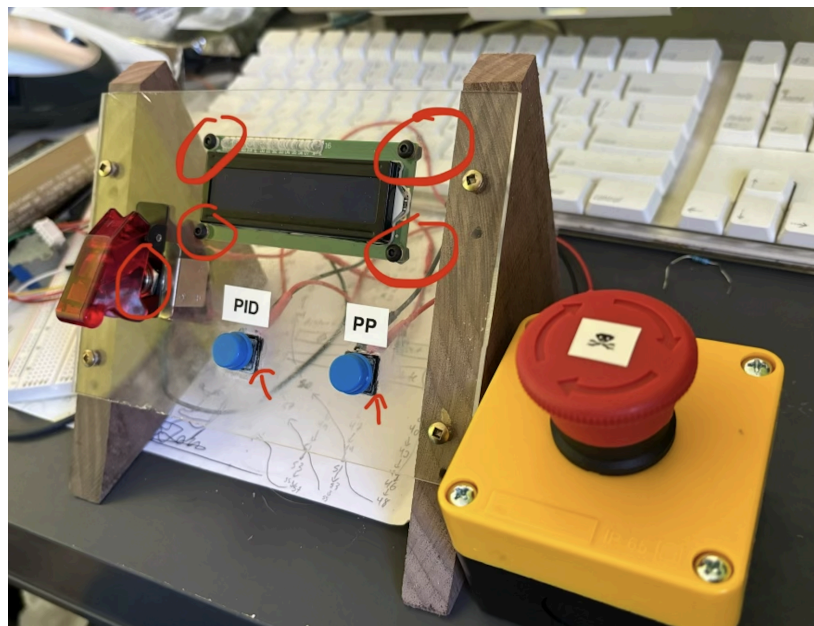


Figure 8.3: Control Panel Components Fastening Techniques

As seen from the mechanical design, the pendulum arm can be detached from the encoder, better yet, the encoder and pendulum arm combination can be safely detached from the stepper motor shaft by loosening the flange shaft coupler which takes a few seconds seen

below. Additionally, the encoder can be removed individually through loosening the 3 screws connecting the encoder to the custom acrylic encoder mount.

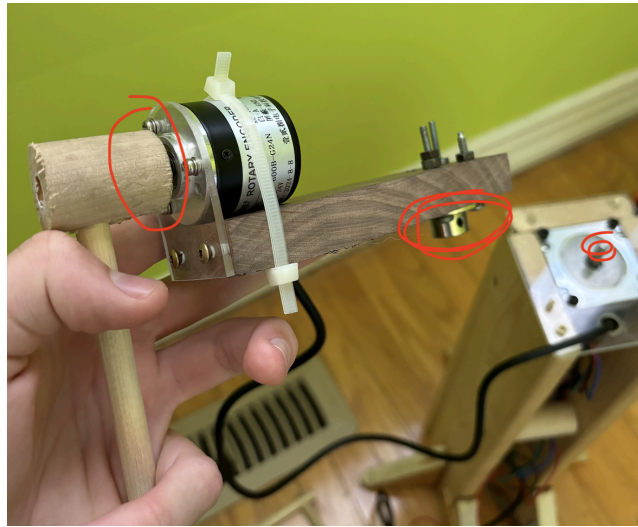


Figure 8.4: Pendulum Arm Fastening Techniques

If there is a problem with the stepper motor, the 2 screws on the acrylic stepper plate can be loosened and removed so that the stepper can safely be lifted upwards over the guide pegs. Be sure to detach any of the necessary wires from the driver/Arduino board if physically disconnecting the pendulum arm (encoder) and/or stepper motor.

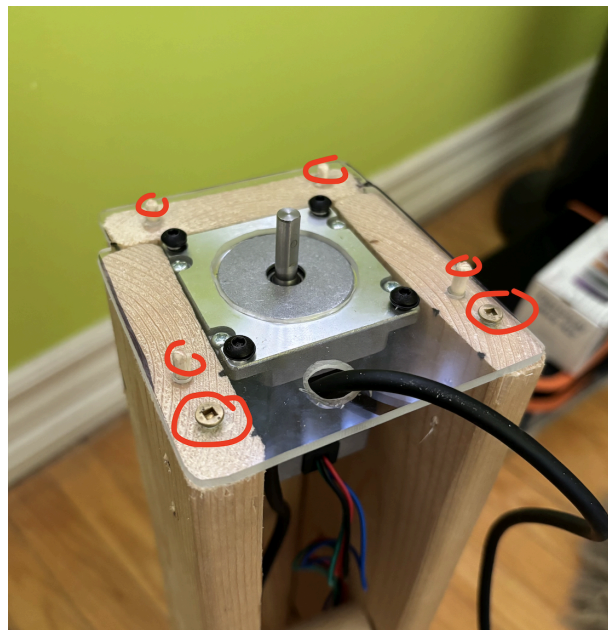


Figure 8.5: Stepper Motor Fastening Techniques

The idea of maintainability is also why both the encoder and stepper motor wires are guided through the base and not permanently secured as this allows for replaceability of the part if required and for basic disassemble/reassemble maintenance. In terms of reassembly, all cables and wires to and from the components on the inverted pendulum system have labels corresponding to their place in assembly. An example of this is the cables from the stepper motor labelled with A+, B+, A- and B- corresponding to where they should be connected on the TB6600 driver. The encoder cables have this same behaviour with its channel A and channel B wires labelled along with power and ground. The control panel wires do not have labels as they are simply positive and negative connections, colour coordinated with red and black. This can be seen below with visual examples of this implementation.

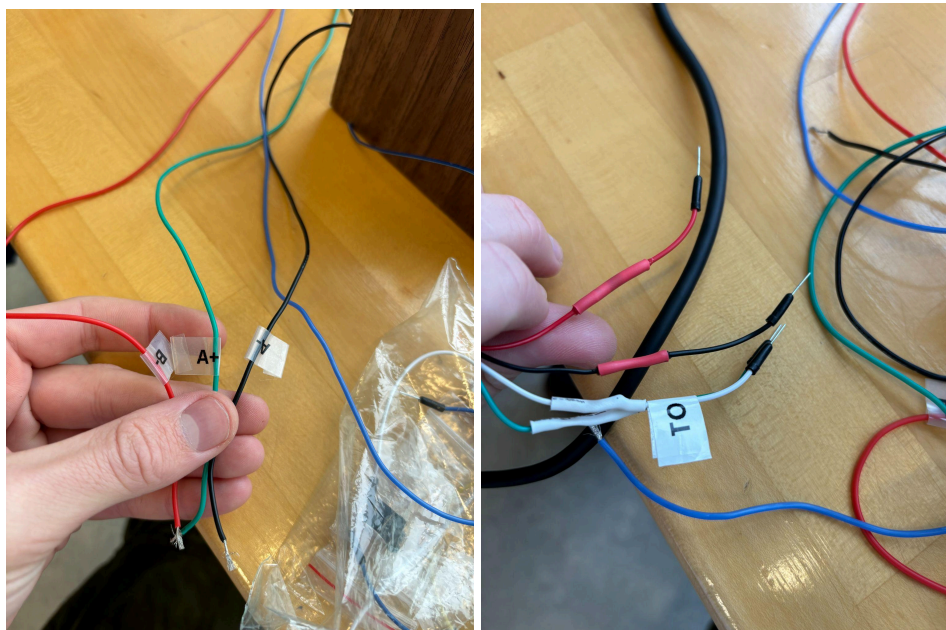


Figure 8.6: Sample of Labeled Wires for Components

## Design of Controls

The choice of buttons is an important topic as to compare the suitable speed, accuracy, ease of operation and required space. All referenced features are from the below figure:

**TABLE 4.8**  
Characteristics of Common Controls (adapted from Damon, Stoudt, and McFarland 1963; Murrell 1965; Chapanis and Kinkade 1972)

Control	Suitability where speed of operation is required	Suitability where accuracy of operation is required	Space required to mount control	Ease of operation in array of like controls	Ease of check reading in array of like controls
Toggle switch (on-off)	Good	Good	Small	Good	Good
Rocker switch	Good	Good	Small	Good	Fair <sup>1</sup>
Push button	Good	Unsuitable	Small	Good	Poor <sup>1</sup>
Legend switch	Good	Good	Small	Good	Good
Rotary selector switch (discrete steps)	Good	Good	Medium	Poor	Good
Knob	Unsuitable	Fair	Small-Medium	Poor	Good
Crank	Fair	Poor	Medium-Large	Poor	Poor <sup>2</sup>
Handwheel	Poor	Good	Large	Good	Good
Lever	Good	Poor (H) Fair (V)	Medium-Large	Good	Good
Foot pedal	Good	Poor	Large	Poor	Poor

Figure 8.7: Control Type Characteristics [1]

The one/off or start/stop switch is selected to be a toggle switch with the addition of a safety cover. This switch is suitable as this control button is responsible for engaging the system as a whole which requires significant speed. Additionally, this decision aligns with all 3 golden rules as it reduces the memory load of the user and makes the interface consistent as the toggle arm up will always remain a certain state and the toggle arm down will remain a certain state (on vs off) and will never change throughout the system. It was decided that 2 other control buttons should be implemented to dynamically change/select the different controller types that are actively balancing the pendulum, these being the PID and Pole-Placement controller. As seen in the table above, knobs, cranks and handwheels are all either unsuitable, fair or poor for applications where speed and ease of use are important. A rotary selector switch would be an efficient choice as it would only theoretically require 3 discrete steps: no controller, PID, and PP but this control is much more difficult to operate in an array of controls and requires more space on the control panel. Additionally, if one were to

desire the PP controller, they would have to turn the selector switch passing through the enable of the other controller which would disrupt the performance metric or system response.

A push button is a suitable option with the only downside of poor ease of reading as there is no built-in visual or audio indicator. The benefit of this downside is that there is no dedicated off or on position and the motion is exactly the same to enable/disable which reduces the memory load of the user and follows one of the golden rules.

Another way to combat this poor ease of reading is the incorporation of an LCD display which could digitally express the current reading or state of the buttons. The simple logic could be if one push button is pressed, one controller is selected when the other push button is pressed, the other controller is selected. If the same controller's button is pressed, it will toggle the action (on  $\rightarrow$  off, off  $\rightarrow$  on). The benefit of this control over an actual toggle switch is that it does not make the interface consistent which counteracts golden rule #3. A toggle switch cannot automatically flip to its set state unless some exterior actuator moves it, thus changing the working state of the buttons. When determining spacing, the table below is referenced.

**TABLE 4.7**  
Recommended Separations for Various Types of Controls (adapted from  
Bradley 1954)


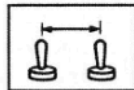
Control	Type of Use	Measurement of Separation	Recommended Separation			
			Minimum		Desirable	
			mm	in.	mm	in.
Push Button	One Finger (Randomly)		12	½	51	2
	One Finger (Sequentially)		6	¼	25	1
	Different Fingers (Randomly or Sequentially)		12	½	12	½
Toggle Switch	One Finger (Randomly)		20	¾	51	2
	One Finger (Sequentially)		12	½	25	1
	Different Fingers (Randomly or Sequentially)		16	⅝	20	¾

Figure 8.8: Control Separation Standards [1]

The control of the toggle switch start/stop or on/off is a single control. The 2 decided push buttons can be classified as different fingers to be safe as well as a random selection. The user isn't obligated to select the push buttons in any order, more of a random decision to change controller type. The desirable spacing is 12mm or 0.5 inches which will be implemented as a minimum distance in the design of the control panel. The toggle switch shall remain on the left side (to start) and as a safe measure, should be at least another 20 mm or ¾ inches from the other push buttons (as a minimum) as this is once again a random selection as the user can start/stop the system any time and there is not a sequence to follow with the one toggle switch.

Controls for initial disturbance angle are not quite necessary for this design as the user can physically apply force to the pendulum arm to destabilize it which will trigger whatever response the system is set to.

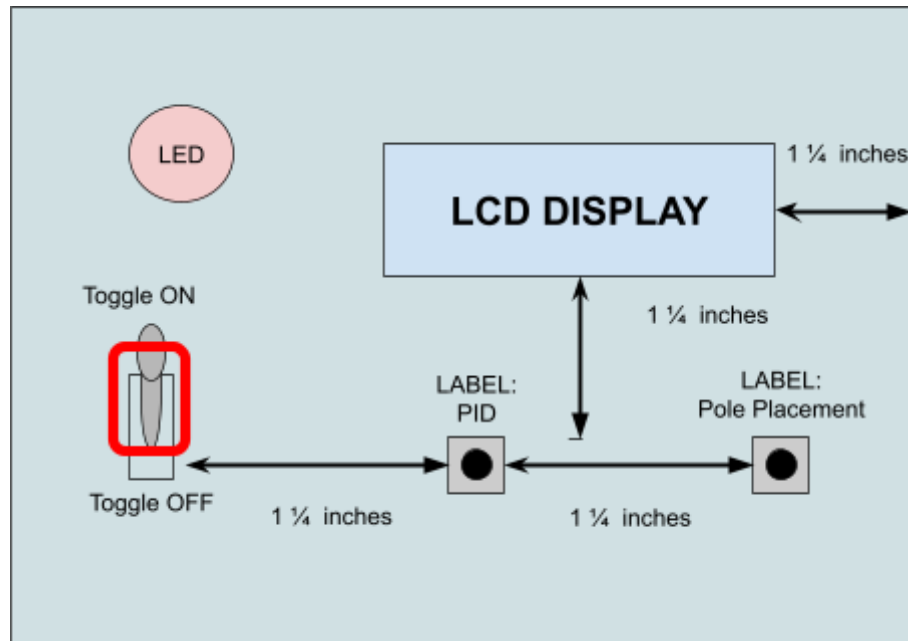


Figure 8.9: Sketch of Initial Control Panel Layout

If the toggle switch selected and purchased already has a built-in LED in the tip, the external LED can be neglected to mitigate any redundancies. Converting this concept to a basic Fusion 360 model with individual components obtained from GrabCAD. To decide on control mechanics, a simple HMI control expectations table can be referenced to conclude that toggle switches up is equal to on, down is off (guided by the cover), and press is engaged (to select the buttons) as seen below.

Function	Direction of Control
On/start/engage	Up, right, forward, press
Off/start/dis-engage	Down, left, rearward, pull

Figure 8.10: System Function with Direction of Control



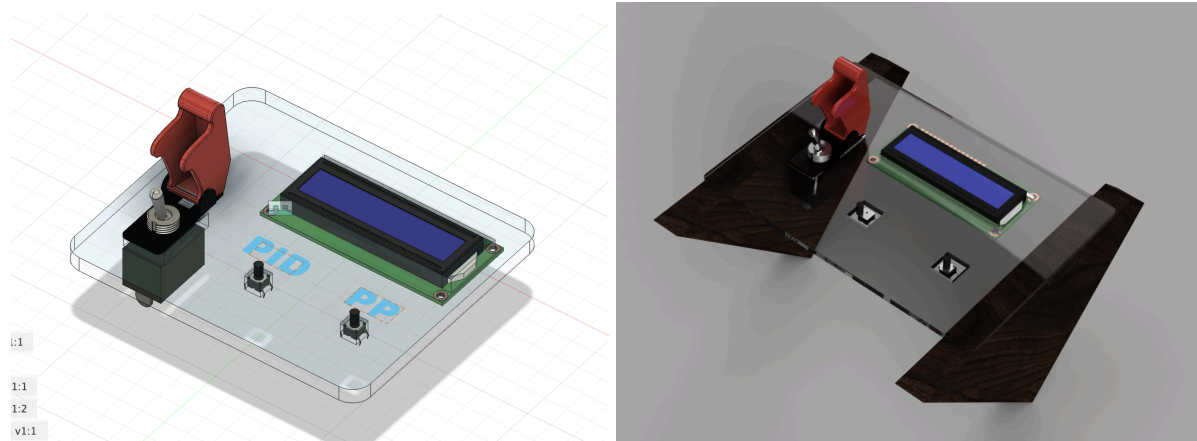


Figure 8.11: High-level Fusion 360 Model and Sketch of Control Panel

### Bottom-Up Reasoning

This method of reasoning implemented a solution built from the individual components and how they function separately to eventually work all together.

#### Gather Information:

The user must rely on information displayed on the LCD and toggle switch LED or audio cues from the buzzer to understand feedback like controller selection and the system's stability. An example of this is at any point, the user must know which controller is actively selected, if the buzzer makes a noise, the system is unstable, etc.

#### Mental Picture:

Users are building an understanding of the system's current state through the various unique elements. As an example, the toggle switch position is always an indication of the state of the system's power, and push buttons are mapped based on the selection of the possible controllers.



### Time Consumption:

In terms of the use of the LCD, the user must confirm and align with the display of the LCD.

The user relies on the component feedback which adds more time steps towards actions. The user visually analyzing and confirming the controller state through the LCD adds another step to process time.

### **Top-Down Reasoning**

This method of reasoning begins with the goals and objectives of the system, which are broken down to smaller items.

### Concept of Machine:

The design of the HMI control panel follows the 3 golden rules. The users are controlled by the toggle switch covers, emergency stop mechanisms to cut operations immediately, etc. The panel reduced the memory load for the user as the state identifiers minimized mental efforts to understand the current system's settings. The push buttons also toggle controller selections dynamically so the user doesn't need to remember the previous position selected.

### Quick Response:

The design implements rapid action tools like the emergency stop to immediately cut power with theoretically minimal delay. As detailed earlier, the human reaction time is 0.15 seconds which is incorporated within the audible and visual cues (LEDs, LCDs and buzzer).

Additionally, the components can be easily detached for maintenance, or emergency actions.

### Error Risk:

An error may occur when the user assumes or interprets something that differs from reality.

An example is the user misreading data displayed on the LCD which would guide the user into selecting the wrong controller or accidentally unselecting/deactivating the stability.

### Labelling and Displays

Moving onto the displays, the information displayed should be dynamic, need to know and clear and kept to a minimum. As the LCD is selected as an annunciator light to display information on the control panel, the main piece of information required is the controller type selected which can be either “NONE”, “PID SELECTED” or “PP SELECTED” as an example. This LCD with 1 line can display the necessary information to the user about how the system will respond as it effectively describes if the system is unstable (no controller) or if the system will stabilize and what specific controller will stabilize it. For this reason, no other displays such as digital readouts or moving pointers are required.

A main display to showcase the system being on or engaged is the built-in red LED in the toggle switch; For this reason, another LED was not implemented in the control panel design as to avoid any redundancies and thus, allowing the main LED to convey the most important message to the user.

To reduce the mental load on the user, controls were incorporated to reduce any scale markings or fine increments which can increase the mental processing time before the user makes a decision to physically activate a control.

Another active display could be a graph which would be seen or viewed by the user on their computer screen from the execution of their Arduino script or potentially using serial communication to control a python script that displays active graphs.

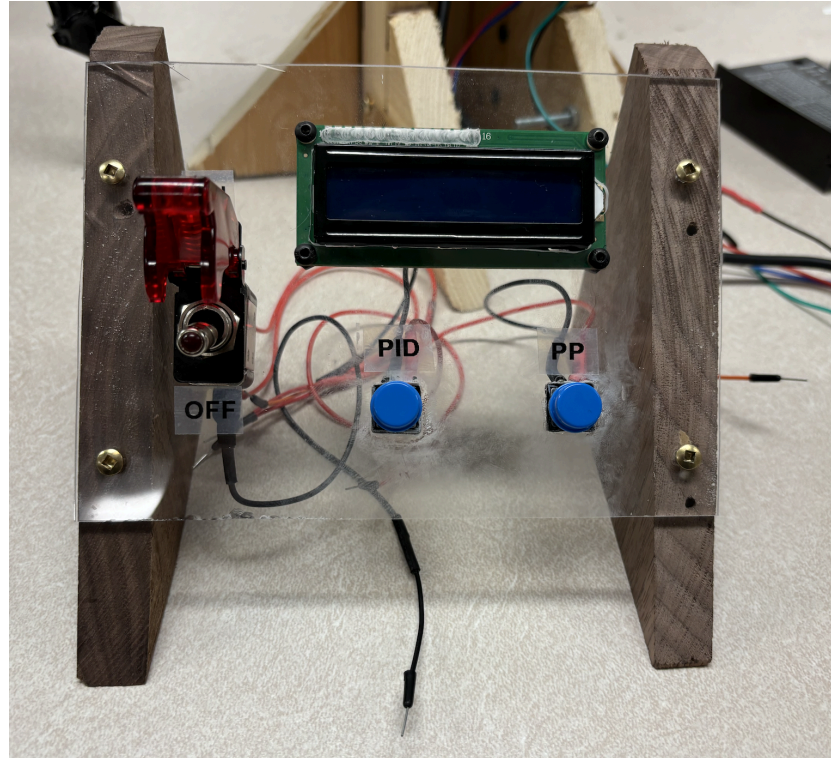


Figure 8.12: Front User View of Control Panel

## 9. Software Implementation

Explanation of code:

The Arduino code is shown in Appendix J.

The code begins by including the 2 necessary libraries, AccelStepper.h for controlling the stepper more easily, and LiquidCrystal.h for controlling the LCD display more easily. It then defines the 2 pins, the 2-channel encoder, the pulse and direction pins for the stepper motor. It also declares the 3 pins for the PID and Pole Placement controller toggles and the ON/OFF switch. It also declares an lcd object from the LiquidCrystal library with the 6 necessary pins. Next, the code declares integers to enable/disable the different controllers, and a “failed” variable that triggers when the pendulum reaches a critical angle. The code also declares doubles for the 3 PID gains, the previous, accumulated and derivative errors, the PID response, and the final PID gain. In the next block are the declarations for the 4 gains of the pole placement controller and a double to store the pole placement response.

After this are some variables used for the encoder to check its direction and store the magnitude of displacement. There is also a variable to set a deadzone of 2 units.

Next, variables are declared to store the stepper speed, acceleration, the goto angle, and step position, and the last step position. These are used with the AccelStepper library to control it later.

An integer to store the beginning time and the current time are declared. Also, a double is declared to store the time interval for differentiation. Also, a serial counter is declared. This is used to increment and slow down the output on the serial port which makes the graphs on the serial plotter easier to look at.

Finally, a stepper object is declared from the AccelStepper library with the required arguments to define the control pins.

In the `setup()` method, the serial is started and the pinmodes for all the pins are set. The stepper object also has its speed and acceleration set with the variables declared earlier. An interrupt is also defined for when channel B detects a change on the encoder. The LCD begins with 16 character width and 2 rows, then it's cleared and "NO CONTROLLER" is printed to it.

The `read_encB()` method is the interrupt declared earlier. It triggers on both rising and falling edges, and checks the state of the A pin to determine the direction of rotation. It increments the `enc_counter` variable to record the change in state.

Inside the loop, it begins by recording the beginning time, and incrementing the serial counter (which will be used later). It checks the state of the buttons and enables/disables the PID and Pole Placement control as needed. If the ON/OFF switch is not activated, both the PID and Pole Placement controllers will be disabled.

The LCD panel is also updated to display which type of controller is current enabled, or it is set to "NO CONTROLLER" if the ON/OFF switch isn't activated

It checks if Pole Placement is enabled, it gets the state of  $x$ ,  $\dot{x}$ ,  $\theta$ , and  $\dot{\theta}$  as these represent the full state of the system. It then computes the response by multiplying in each of the 4 gains and summing the results.

Next, it calculates the error for the PID response. Important to note here is that error is squared while preserving the sign (which is important). Making error proportional to the square of the deviation means greater deviations are more heavily weighted to provide better response. The accumulated error is updated and the derivative error is calculated using the previous error and the time interval. The PID response is then computed by multiplying in the  $K_p$ ,  $K_i$ , and  $K_d$  and summing everything up.

After this point, there is a quick check to see whether the encoder angle is past a critical point, and if so, the failed variable is set to 1 so the motor will be disabled for safety.

The PID response sign is checked to determine the correct direction of motor movement, and the direction pin is sent a HIGH or LOW signal accordingly. The PID response is then multiplied by the gain and constrained.

Finally, it is checked whether the PID is enabled, if the response is not 0, if the encoder angle is outside the deadzone, and also that the failed variable isn't set to 0. If the conditions are met, the stepper position is updated using the PID response and the motor is told to move to that position.

The other option is that if pole placement is enabled, the same checks are done and if the conditions are met, the step position is updated and the motor is told to move to that position.

The previous error value is updated for use in the next iteration of the loop, and the serial counter is checked to see if it has reached 1000. If it has, the encoder and stepper positions are printed to the serial monitor and the serial counter is reset. This slows down the flow of data to the serial plotter graphs, making them easier to look at.

## PID Control

```
//Generate PID response

int error = enc_counter;

error = error * abs(error); //Increases sensitivity to higher values

acc_error += error*time_interval;

der_error = (error - prev_error)/time_interval;

PID_Response = Kp*error + Ki*acc_error + Kd*der_error;
```

As shown in this code, the proportional, integral, and derivative errors are all computed and added together.

### Pole-Placement Control

```
//Generate PP response  
  
float theta = enc_counter;  
  
float theta_d = (enc_counter - prev_error)/time_interval;  
  
float x = step_pos;  
  
float x_d = (step_pos - last_step_pos)/time_interval;  
  
  
PP_Response = K1*x + K2*x_d + K3*theta + K4*theta_d;
```

As shown in this code, the full state of the system is computed and then multiplied by the corresponding gains to compute the controller response.

### Noise and Filtering

Simulated system and watch response with no controller to see if there's noise in the system  
(show graph with potential noise if any)

If there is, we can implement a simple moving average filter technique and average the last 4 data points (to maintain accurate current reading and reduce filter noise)

Simulate system and watch response with no controller

## 10. Software Version Control

By incorporating a version control through github and google drive, our group was able to impressively increase collaboration, accountability and overall quality of the project.

Allowing group members to work on the codes concurrently, version control prevented overwriting and other conflicts that could have arised. This also allowed for members to track the work done and seamlessly move from different versions to troubleshoot any errors during testing.

## 11. Performance and Responses

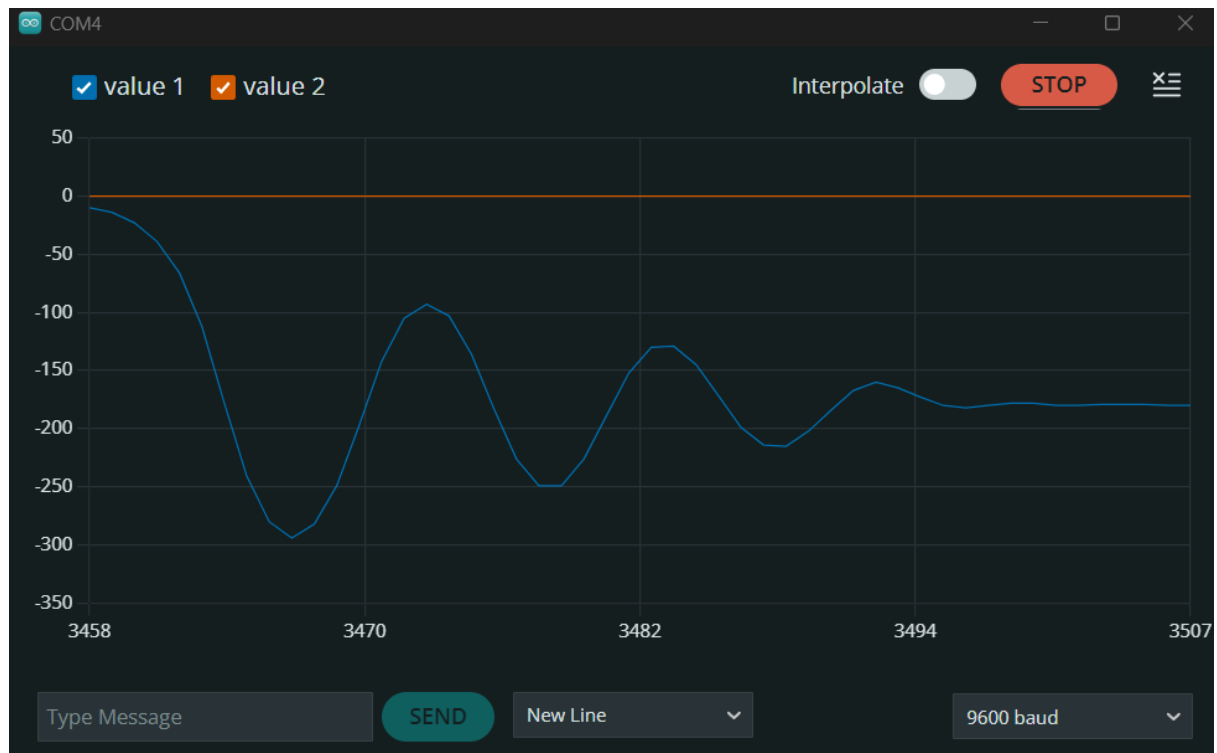


Figure 11.1 Performance without controller (Blue is angle, Red is stepper response)

It can be seen in figure 11.1 that the system behaves as expected when no controller is used. The pendulum drops and oscillates before settling at 180 degrees which is the downward pointing position.





Figure 11.2 Performance with PID controller (Blue is angle, Red is stepper response)

It can be seen in figure 11.2 that the system with the PID controller responds to a disturbance to correct the pendulum angle. It does this with a rapid change in motor position. It then stays still when the pendulum has reached 0 degrees again. The settling time is quite short for a small disturbance like this, which is good for keeping the pendulum stable over long periods of time.

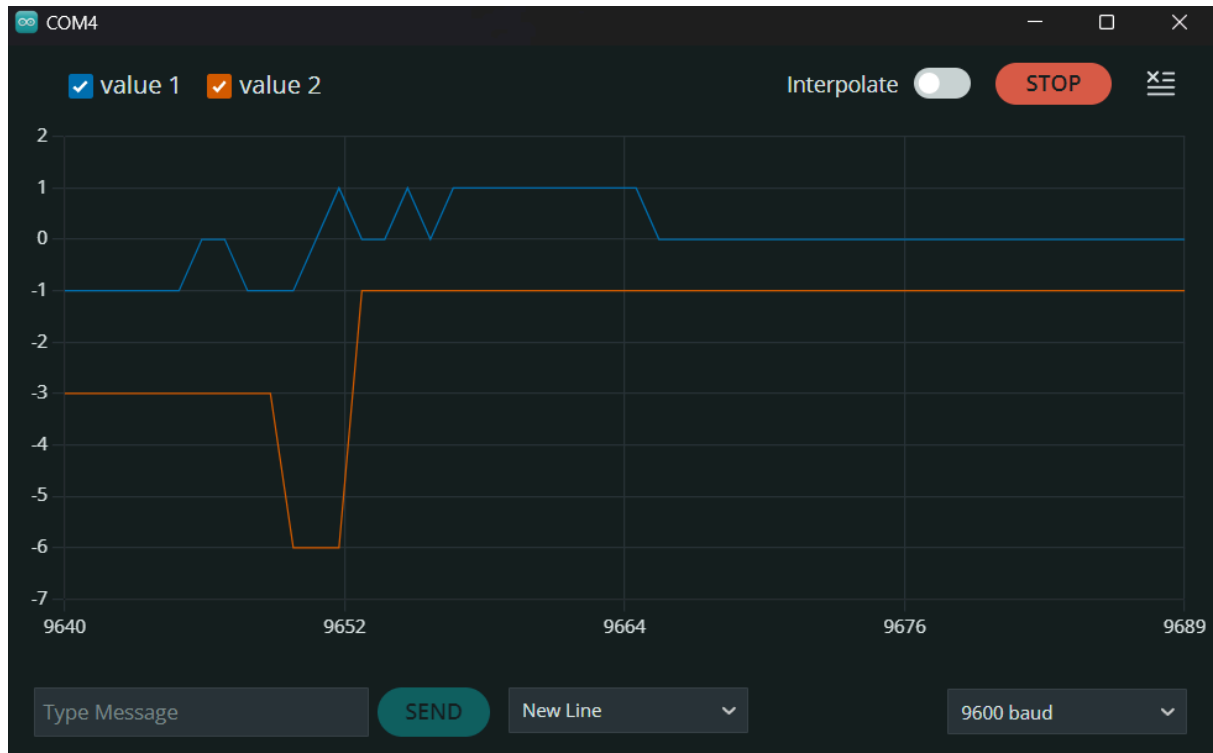


Figure 11.3 Performance with PP controller (Blue is angle, Red is stepper response)

It can be seen in figure 11.3 that the system with the PP controller also responds to a disturbance to correct the pendulum angle. It does this with a rapid change in motor position. We can see it also moves in the opposite direction to counteract the oscillation in the pendulum angle and eventually settles when the pendulum is again at 0 degrees.

From these graphs, it's clear the performance of both the PID and Pole Placement controllers is adequate to balance the pendulum at steady state and handle small disturbances with precise, rapid responses.

## 12. Bill of Materials / Cost Breakdown

1 - Nema 23 - \$28

1 - Tb6600 driver - \$18

1 - Encoder - \$20

1 - Flange shaft coupler mounts - \$4

2 - Push buttons- \$3

1 - Toggle switch - \$5

≈ 15 - Screw set - \$1

1 - Emergency button \$15

≈ 46in<sup>2</sup> of 1/8" Acrylic sheet

≈ 230in<sup>2</sup> of 1/2" Pine Wood

≈ 30 - Jumper cables (male → female)

1 - Passive Buzzer

1 - 2x16 LCD display

1 - Elegoo R3 Uno board

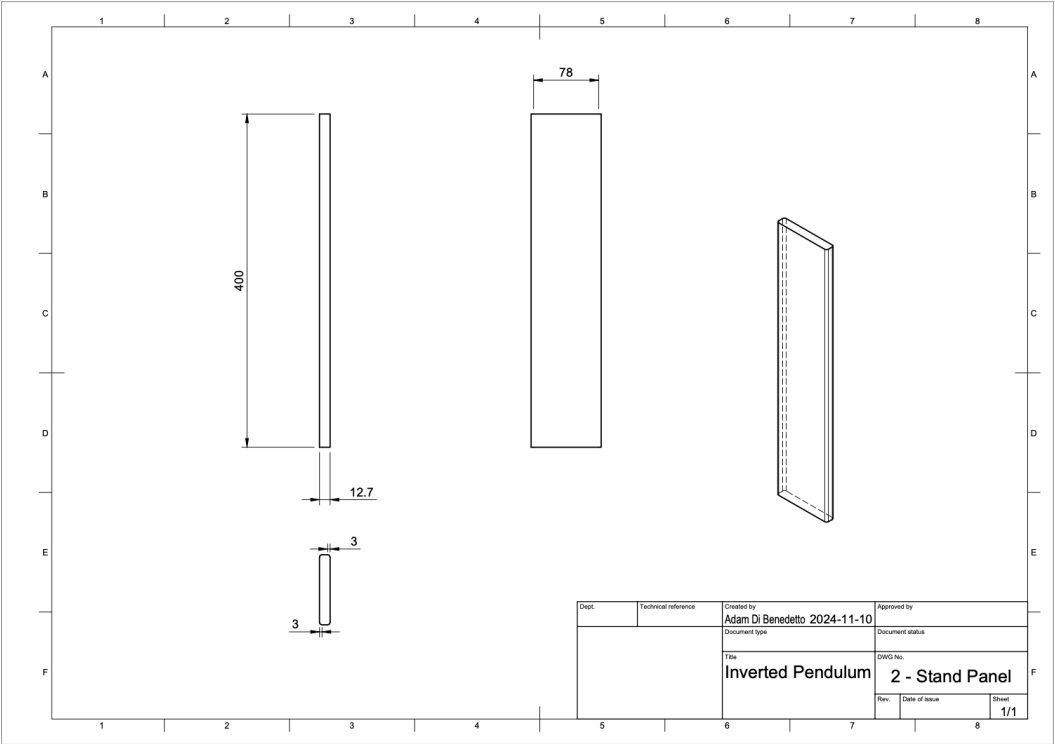
Miscellaneous: extension soldered wires, super glue, hot glue

### 13. References

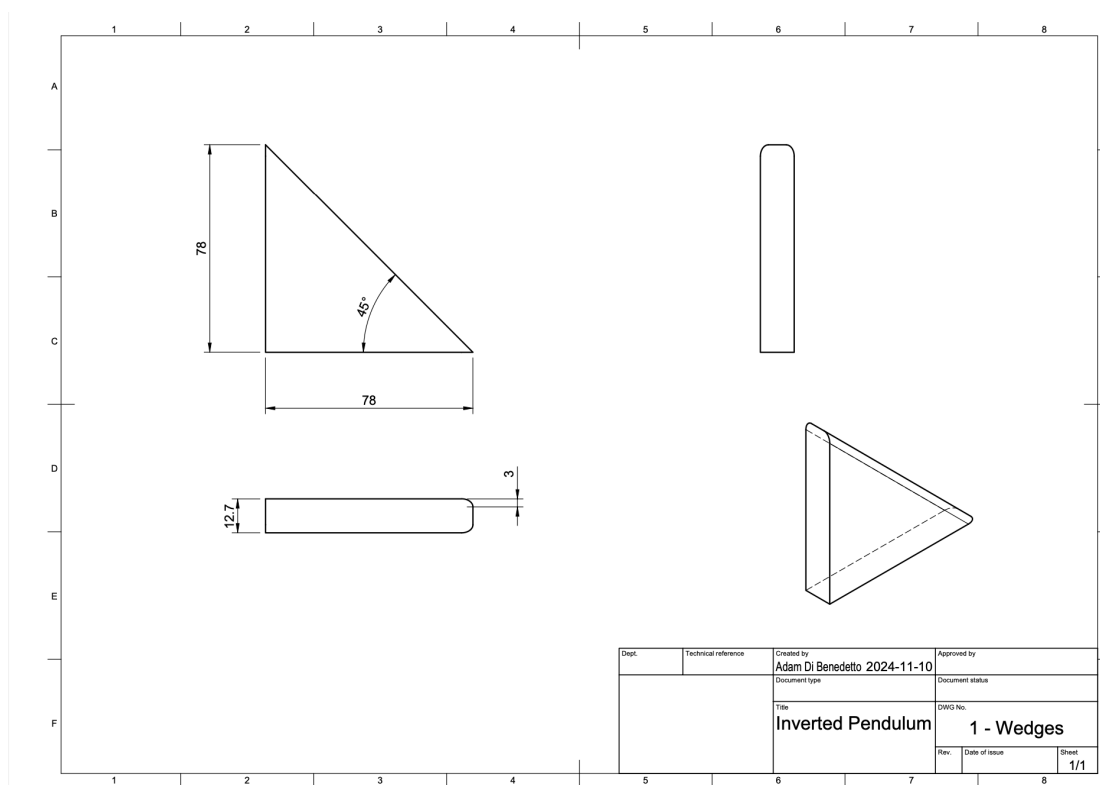
- [1] Kodack's Ergonomic Design for People at Work, 2ed., John Wiley & Sons, New York, 2004
  
- [2] Toggle Switch Security Cover Guard – ESG Asia Pacific,  
<https://www.eccosafetygroup.com.au/product/toggle-switch-cover-guard/> (accessed Nov. 7, 2024).
  
- [3] J. X. Wang, Engineering Robust Designs with Six Sigma. Prentice Hall, 2005.
  
- [4] Mechatronics Design (MEC830), 2019, V. Chan
  
- [5] “Baomain Red Sign Emergency Stop Switch push button weatherproof push button switch with box 10a 660V 1NO 1NC : Amazon.ca: Tools & Home Improvement,” Baomain Red Sign Emergency Stop Switch Push Button, [Amazon Link](#) (accessed Nov. 6, 2024).
  
- [6] MoThunderz. “How to Use a Rotary Encoder with an Arduino - CODE EXPLAINED!” YouTube, YouTube, 11 Jan. 2023, [www.youtube.com/watch?v=fgOfSHTYeio](http://www.youtube.com/watch?v=fgOfSHTYeio).
  
- [7] How to Mechatronics. “Stepper Motors and Arduino - The Ultimate Guide.” YouTube, YouTube, 15 May 2022, [www.youtube.com/watch?v=7spK\\_BkMJys](http://www.youtube.com/watch?v=7spK_BkMJys).
  
- [8] Sergio A. Castaño Giraldo. “State Feedback Controller Design on Arduino | Applied Control System Course.” YouTube, YouTube, 11 Apr. 2024, [www.youtube.com/watch?v=v1MZQ2kHa0I](http://www.youtube.com/watch?v=v1MZQ2kHa0I).

14. Appendices

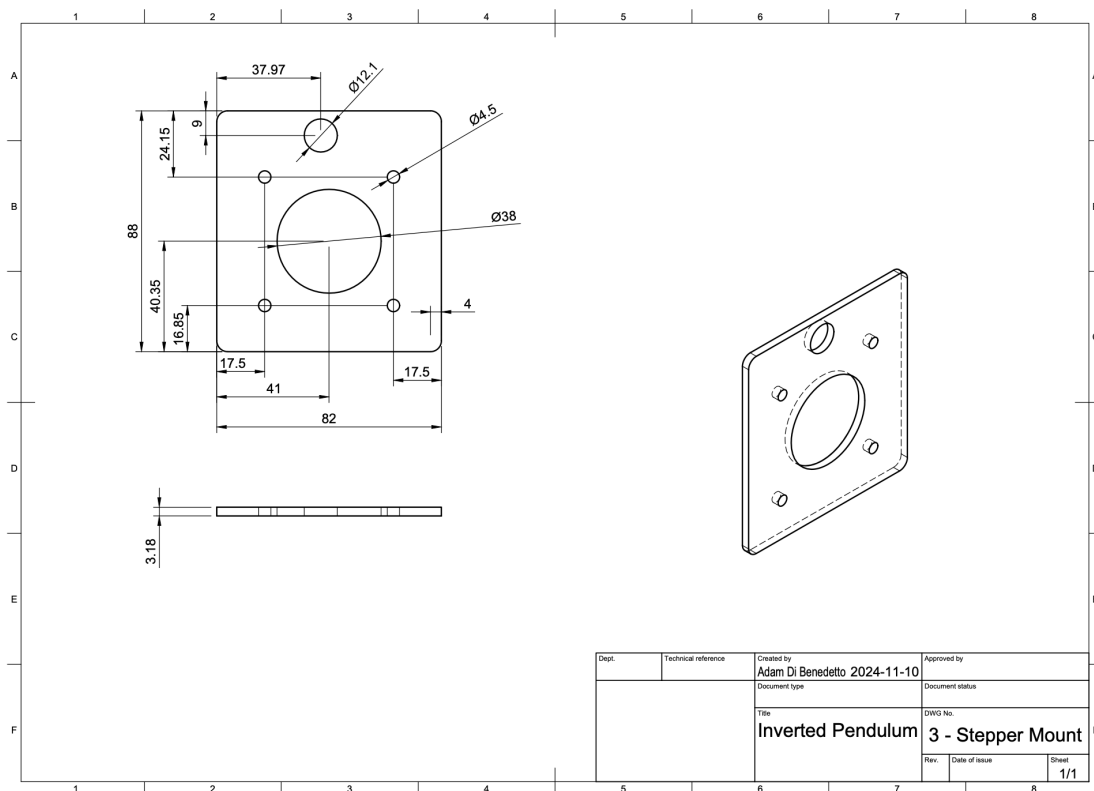
Appendix A - Panel Stand Part Drawing



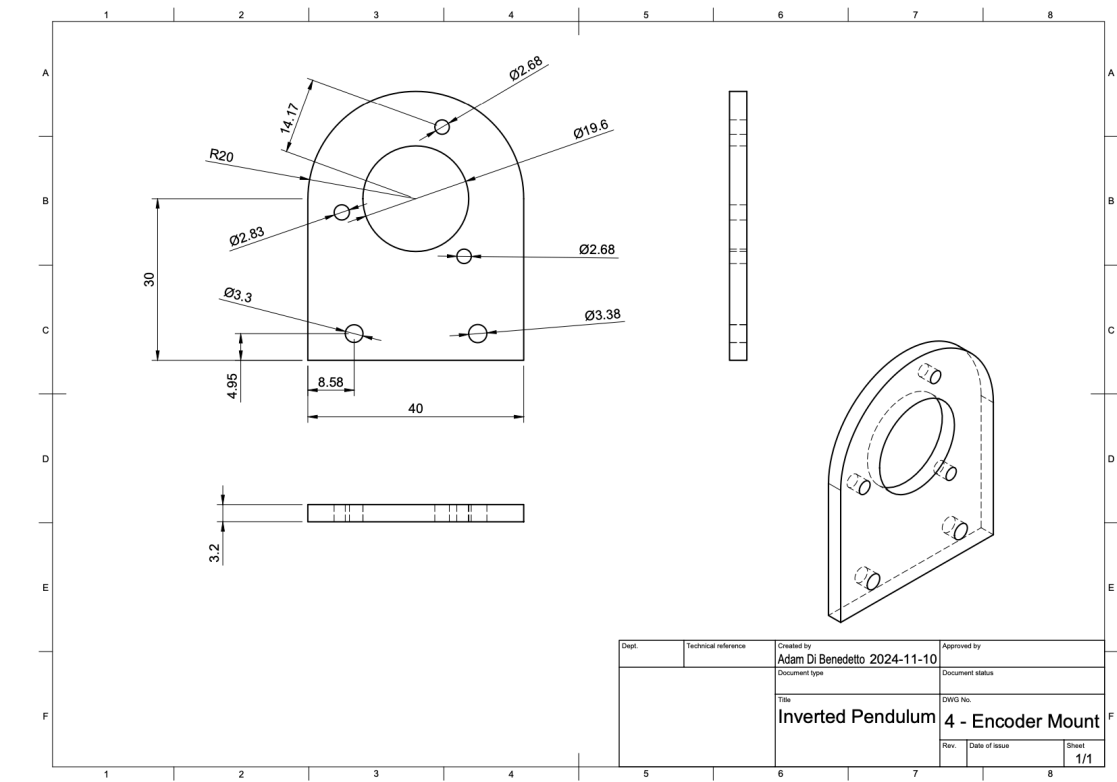
## Appendix B - Stand Wedge Part Drawing



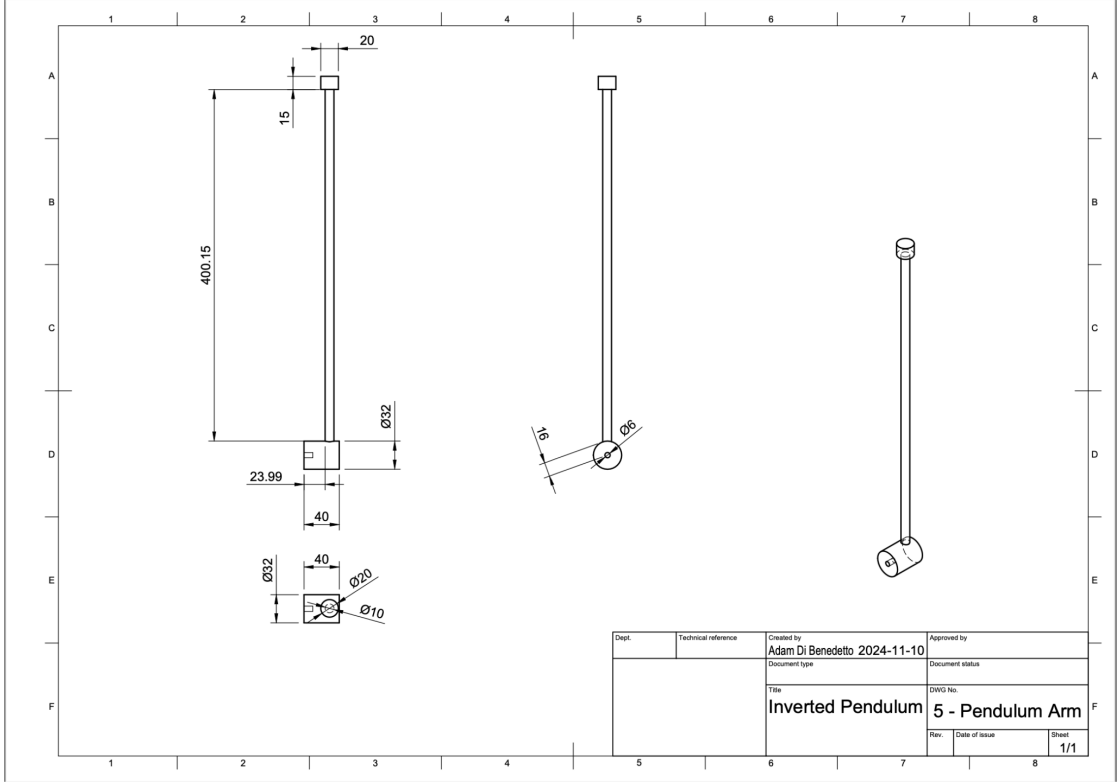
## Appendix C - Stepper Motor Mount Part Drawing



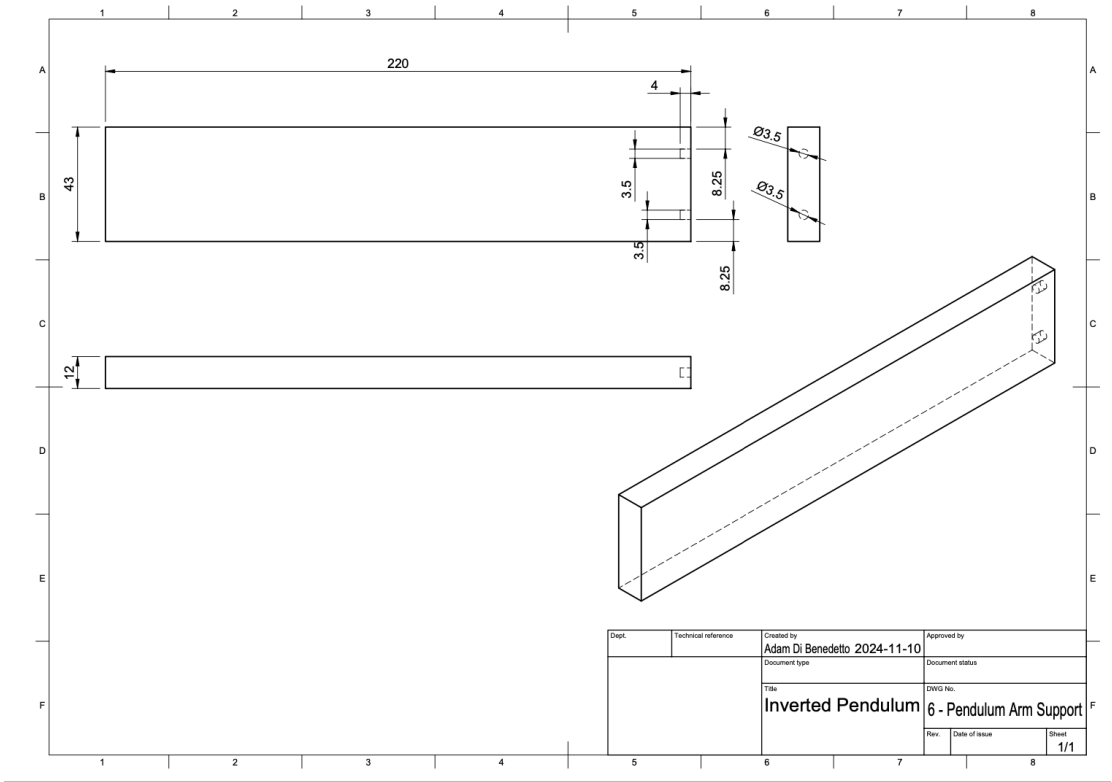
Appendix D - Encoder Mount Part Drawing



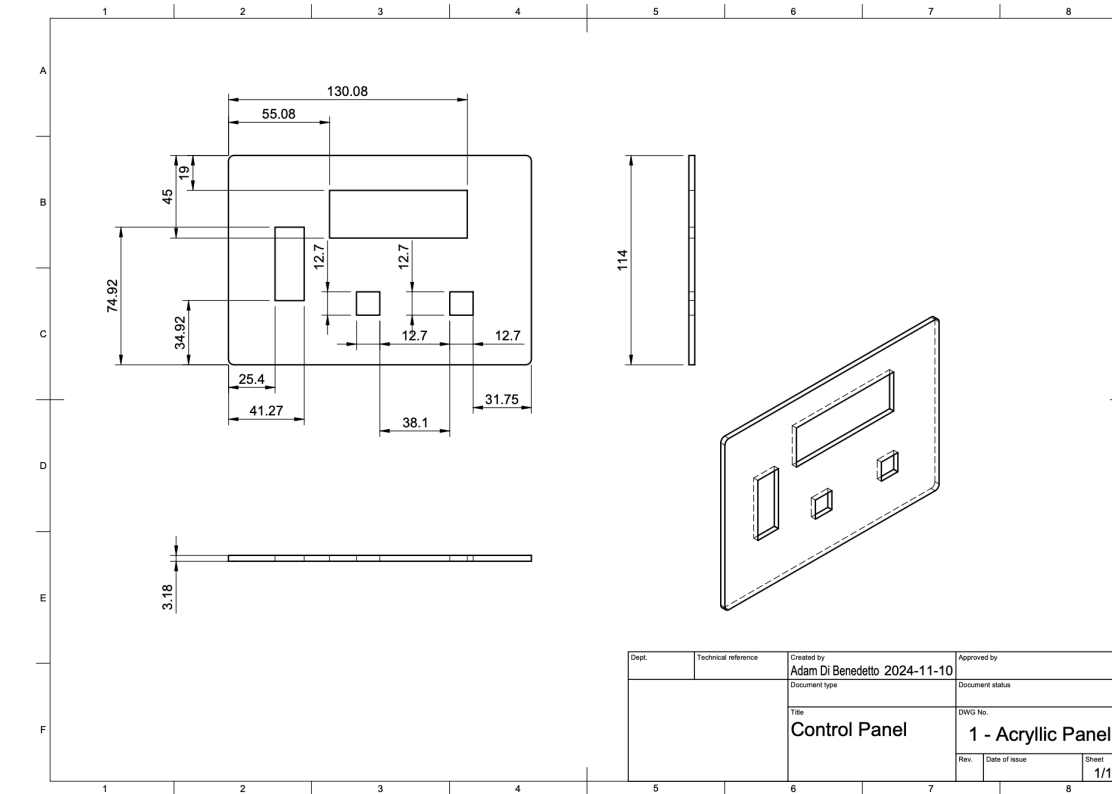
Appendix E - Pendulum Arm Part Drawing



Appendix F - Arm Support Part Drawing

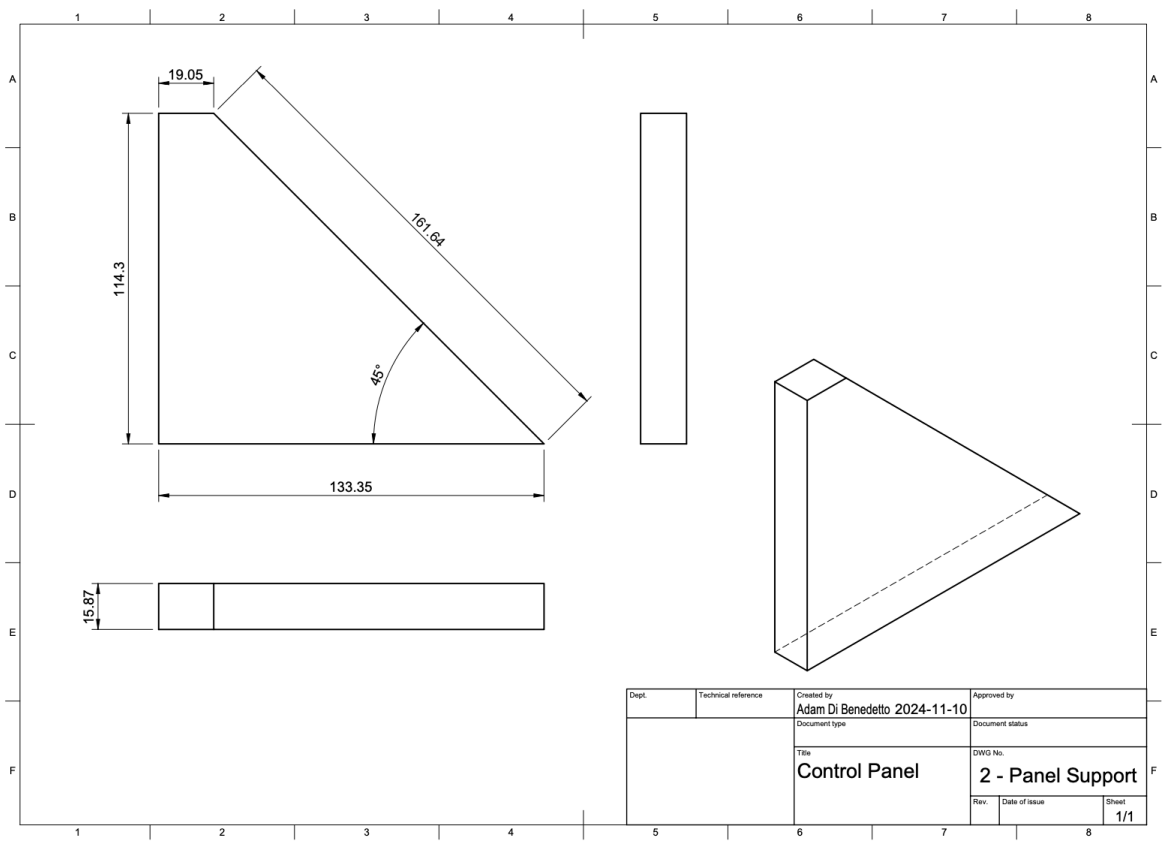


Appendix G - Control Front Panel Part Drawing

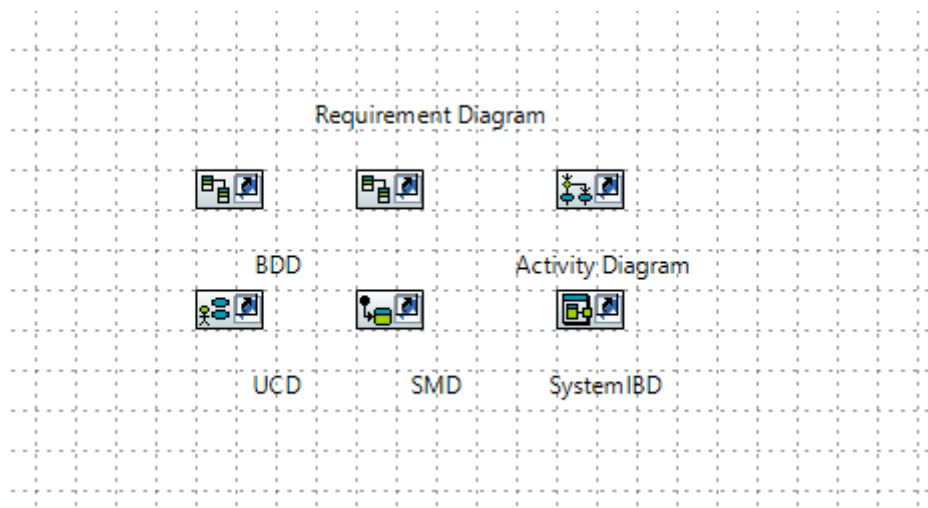




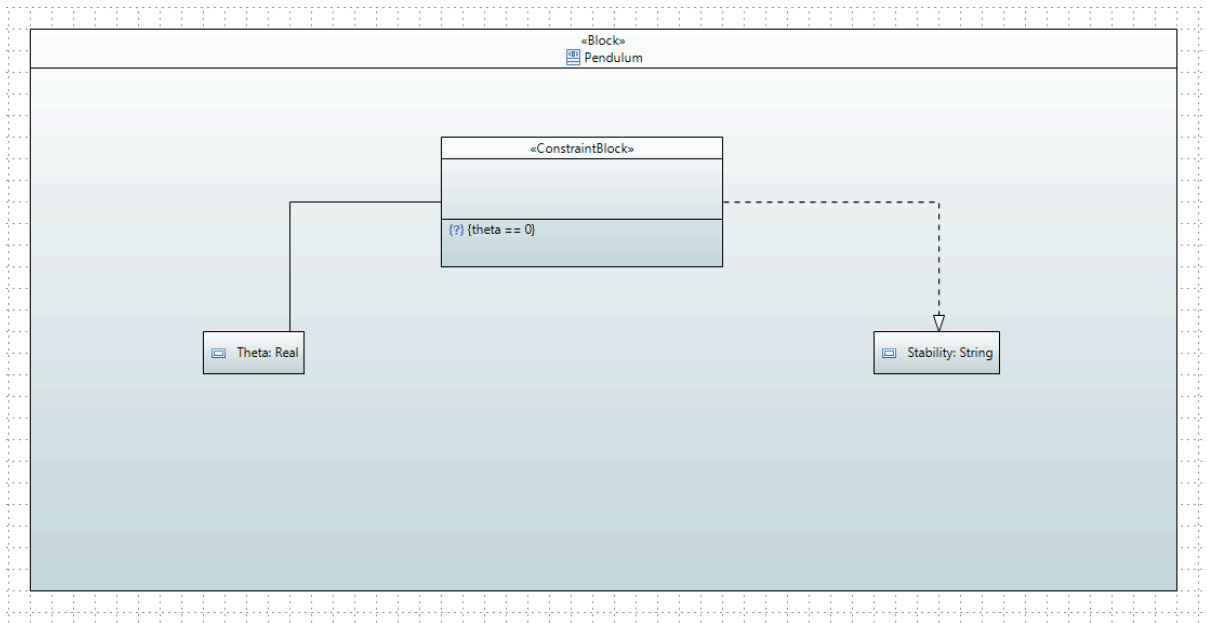
## Appendix H - Control Panel Supports Part Drawing



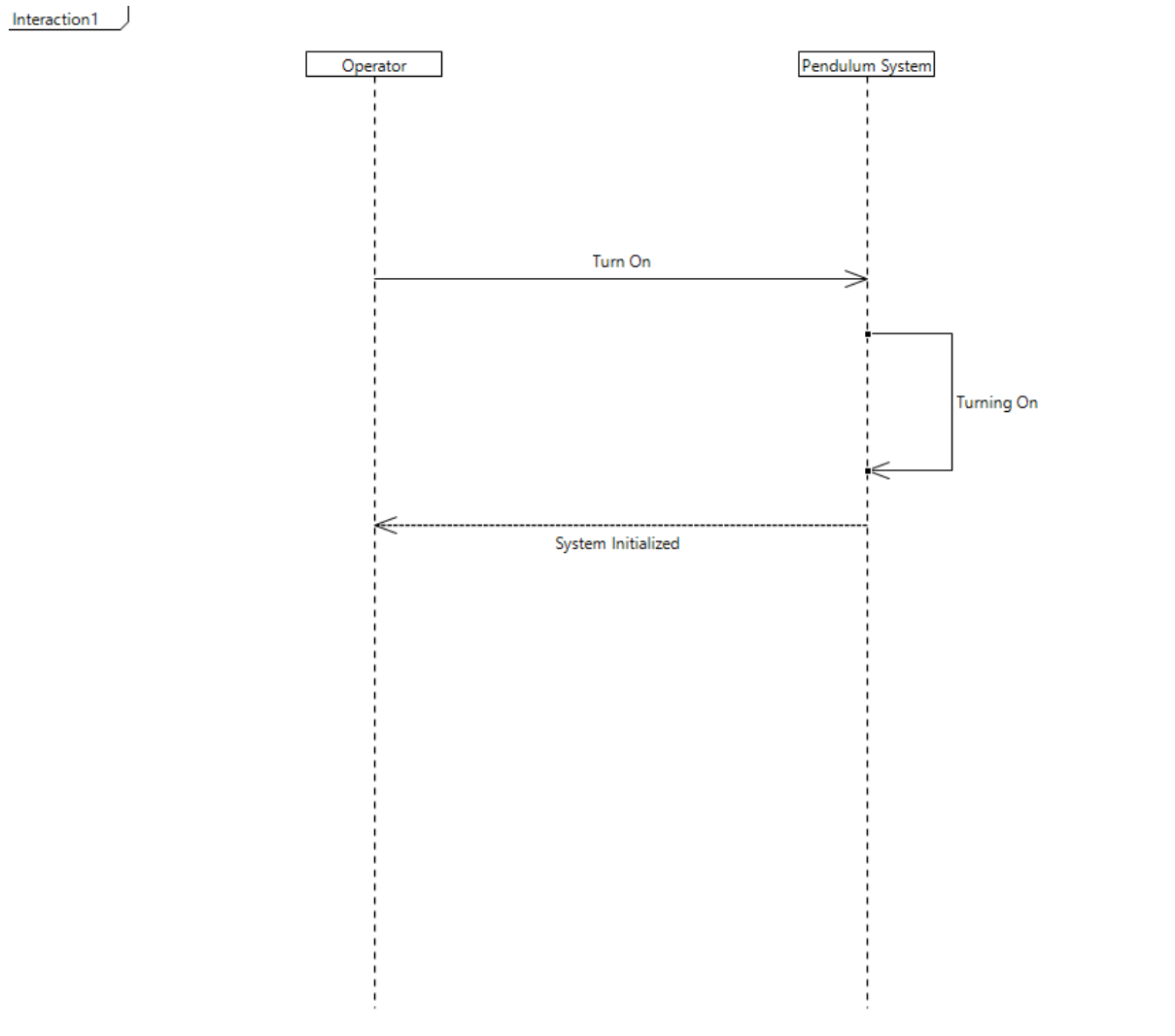
## Appendix I - MBSE Drawings



Package Diagram



Parametric Diagram for Pendulum Arm



Sequence Diagram for Initializing the System

## Appendix J - Code

```
//MEC830 Project 2 code

#include <AccelStepper.h>
#include <LiquidCrystal.h>

#define ENC_A 5
#define ENC_B 3
```

```
#define ST_PULSE 6

#define ST_DIRECTION 7


#define PID_TOGGLE 2

#define PP_TOGGLE 0

#define ON_SWITCH 4


LiquidCrystal lcd(13, 12, 11, 10, 9, 8);


int System_Enabled;

int PID_Enabled = 0;

int PP_Enabled = 0;

int failed = 0;


double Kp = 8;

double Ki = 0.012;

double Kd = 0.008;

double prev_error = 0;

double acc_error = 0;

double der_error = 0;

double PID_Response = 0;

double PID_gain = 1;


double K1 = 0;

double K2 = 0;
```

```
double K3 = 1;

double K4 = 0;

double PP_Response = 0;


int prev_A = 0;

int prev_B = 0;

int checker = 0;

int enc_counter = 0;

int enc_dz = 2;


//int pulse_length = 0;

//int prev_pulse_time = 0;

float step_speed = 8000;

float step_accel = 20000;

float step_goto = 500;

int step_pos = 0;

int last_step_pos = 0;


int beg_time = 0;

int time_counter = 0;

double time_interval = 0.01;


int serial_counter = 0;


AccelStepper stepper1(1, 6, 7);
```

```
void setup() {  
    Serial.begin(9600);  
  
    pinMode(ENC_A, INPUT_PULLUP);  
    pinMode(ENC_B, INPUT_PULLUP);  
  
    stepper1.setMaxSpeed(step_speed);  
    stepper1.setAcceleration(step_accel);  
  
    pinMode(PID_TOGGLE, INPUT_PULLUP);  
    pinMode(PP_TOGGLE, INPUT_PULLUP);  
    pinMode(ON_SWITCH, INPUT_PULLUP);  
  
    //Declares an interrupt that will occur whenever encoder pin B changes  
    attachInterrupt(digitalPinToInterrupt(ENC_B), read_encB, CHANGE);  
  
    lcd.begin(16, 2);  
    lcd.clear();  
    lcd.print("NO CONTROLLER");  
}  
  
void read_encB(){  
    //Determine direction by checking state of encoder pin A
```

```

if (digitalRead(ENC_B) == HIGH && digitalRead(ENC_A) == HIGH){
    enc_counter++;
}

else if (digitalRead(ENC_B) == HIGH && digitalRead(ENC_A) == LOW){
    enc_counter--;
}
}

void loop() {
    beg_time = millis(); //Must occur at the beginning
    serial_counter++;

    //Check buttons
    if (digitalRead(PID_TOGGLE) == LOW){
        PID_Enabled = 1;
        PP_Enabled = 0;
        lcd.clear();
        lcd.print("PID CONTROL");
    }

    if (digitalRead(PP_TOGGLE) == LOW){
        PID_Enabled = 0;
        PP_Enabled = 1;
        lcd.clear();
        lcd.print("PP CONTROL");
    }
}

```

```

if (digitalRead(ON_SWITCH) == HIGH){

    PID_Enabled = 0;

    PP_Enabled = 0;

    lcd.clear();

    lcd.print("NO CONTROLLER");

}

if (PP_Enabled){

    //Generate PP response

    float theta = enc_counter;

    float theta_d = (enc_counter - prev_error)/time_interval;

    float x = step_pos;

    float x_d = (step_pos - last_step_pos)/time_interval;

    PP_Response = K1*x + K2*x_d + K3*theta + K4*theta_d;

}

//Generate PID response

int error = enc_counter;

error = error * abs(error); //Increases sensitivity to higher values

acc_error += error*time_interval;

der_error = (error - prev_error)/time_interval;

PID_Response = Kp*error + Ki*acc_error + Kd*der_error;

if (enc_counter > 70){

```



```

    failed = 1; //Disables system if the pendulum has fallen too far
}

//Set direction for stepper motor
if (PID_Response > 0){
    digitalWrite(7, LOW);
}
else{
    digitalWrite(7, HIGH);
}

//Transform PID response to use for stepper motor
PID_Response = constrain(PID_Response * PID_gain, -2000, 2000);

if (PID_Enabled == 1 && PID_Response != 0 && abs(enc_counter) > enc_dz && !failed){
    last_step_pos = step_pos;
    step_pos += PID_Response;
    stepper1.moveTo(step_pos);
    stepper1.runToPosition();
}
else if (PP_Enabled == 1 && abs(enc_counter) > enc_dz && !failed){
    last_step_pos = step_pos;
    step_pos += PP_Response;
    stepper1.moveTo(step_pos);
    stepper1.runToPosition();
}

```

```
}

prev_error = enc_counter;

//time_interval = (millis() - beg_time) / 1000; //Must occur at the end

if (serial_counter > 1000){
    Serial.print(map(enc_counter, 0, 300, 0, 180));
    Serial.print(",");
    Serial.println(step_pos / 100);
    serial_counter = 0;
}
}
```