Game Development with Custom Controller: Design Report

Ian Burns, Andy Li, Ian Rogers
Software Design Lead, Hardware Design Lead, Team Leader

ECE 401 - Electrical and Computer Engineering Senior Design
Department of Electrical Engineering and Computer Science
University of Tennessee
Knoxville, TN 37996

ylx739@vols.utk.edu
ali20@vols.utk.edu
irogers1@vols.utk.edu

Project Mentor: James Osborne

Version 1.01
03/06/24

Executive Summary

Our senior design project's goal is to redefine the gaming experience of Super Smash Bros by developing a custom version of the game. Our main idea to make it more custom, and different is the characters will be college mascots, each with its own special ability, different amount of health, and more! Along with that, we are going to make a custom game controller tailored for this game. By doing both of these things for our Senior design project the project will combine software development, and hardware engineering since our majors are Computer Engineering we wanted a project that had a mix of software and hardware. Our goals are to create a better user experience for the game, and a special controller made specifically for this game.

Our main objectives are to create the modified game with unique characters, gameplay, and mechanics / powerups. Along with that, the custom controller will be optimized for this game by incorporating special features such as using a potentiometer for an analog input to the arduino which will return a value that will be used for the game, certain buttons for activating an ability and more.

Throughout this report you will find explanations of problems resulting in why we decided on this project, the specifications for the project, our initial design decisions and concepts, a basic timeline for when we want things to be completed and turned in as deliverables, and our budget for the project including what money we spent and on what. By the end of this report our goal is that you will understand the reasoning for this project, our solution on how to improve the game, and controller, and how we plan to actually implement the solutions we came up with.

Table of Contents

List of Figures

List of Tables

Problem Definition & Background

The main problem we are trying to solve through our project is there are certain aspects of Super Smash Bros, and certain aspects of the controllers such as Xbox and Playstation controllers that we don't particularly like, along with some features we would like to see added. Our idea was to create a game from scratch that kind of follows in its footsteps, and create a custom controller with all the similar functionality, but adding some features we want specifically that aren't really on current game controllers. Not only are we having these problems, but when asking fellow classmates, gamers, and friends a lot of people agreed that there were some things just missing in the current game / controllers used for games that we want to address.

The main aspect of Super Smash Bros is you control a character and try to use abilities to knock the other person off / get rid of all their health, and most of the characters are Mario, or other characters from Mario. However, our idea was to make the characters college mascots, and give each one an ability based on their mascot. For example, Alabama's mascot is an Elephant so it would be slower than others but potentially do more damage while Tennessee's mascot is Smokey the dog which would be faster but less damage, etc. This project's game will be used by customers by downloading the software we made, and then hooking up the laptop or whatever is running the software to a monitor. Then the customer would be able to use our custom controller to play the game via bluetooth and usb.

The underlying background that needs to be known to solve this problem is actually quite a lot. We need to first understand how to actually implement this software and program the actual game which will require extensive research into different software packages, languages, etc. While the controller will take research into the Arduino board we are using, how we can work UART with it, and actually hook up buttons, joysticks, and a battery all together which could potentially need a PCB board which would require PCB knowledge as well.

Requirement Specification

Throughout this project we as a team have determined a couple requirements for this project to be counted as a successful project. One of the main requirements for this project to be a success is the fact we need a complete, working game that can be played with our custom controller. The reason that the complete game is required is due to the amount of time we as a group have to work on this project which also means there should be very little bugs, if any, in the actual software of the game due to the fact that most games that are put out for people to play have very little to zero bugs. As for the hardware, which is the custom controller, the reason having a fully functioning controller is a requirement for this project is because without the controller the game / software itself would practically be unplayable. Since we are going to program the software game to respond to that of the controllers, without the controller the game itself wouldn't

function which also means a non-successful project. We developed these requirements in discussion with our technical advisor in our initial project meeting. Our technical advisor agreed with us that a successful project would have each component, the software / game, and the hardware both functioning by themselves, but also when combining them to make one full project. As for prioritization, the main priority is the controller due to the fact that through our experiences as Computer Engineers the hardware is where things can get messed up easily, especially since there is a chance our controller is made on a PCB board. We have learned that whenever you are dealing with PCB board design you have to account for it not working when you receive the board, and there are lots of V&V tests that have to be done to verify the functionality of the board. This is why the controller is priority number one for us. The software / game is priority number two just due to the fact that there are less things to account for except for verifying the gameplay is functioning properly which is why we decided it's priority two.

Technical Approach

Our full design for the custom controller and game is to create a working system where the controller will interface with the game. Our controller will run in a configuration so it will take inputs from an assortment of buttons, joysticks, and a slider potentiometer into an arduino board. The Arduino board's job is to convert the signals from the inputs to an output signal that corresponds to an ASCII key. For example, if we use a button as a jump key, the board will read the signal input from a button press and convert to an ASCII value and send it via USB or Bluetooth to the game. The game will have control inputs for movement such as W, A, S and D. The block diagram below in figure 1. Demonstrates a general layout of our custom controller design.

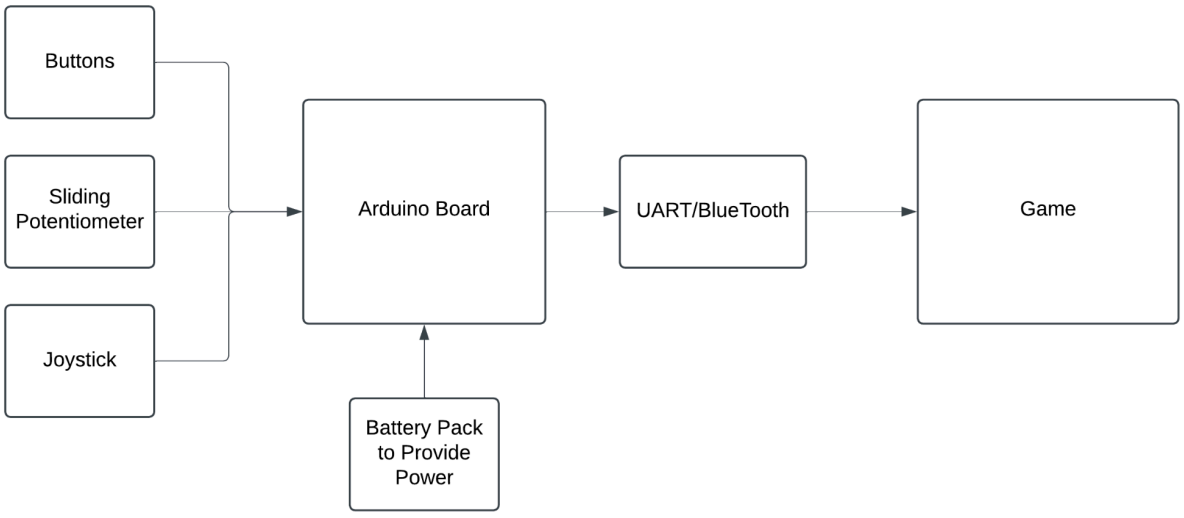


Figure 1. Block Diagram of Controller Layout.

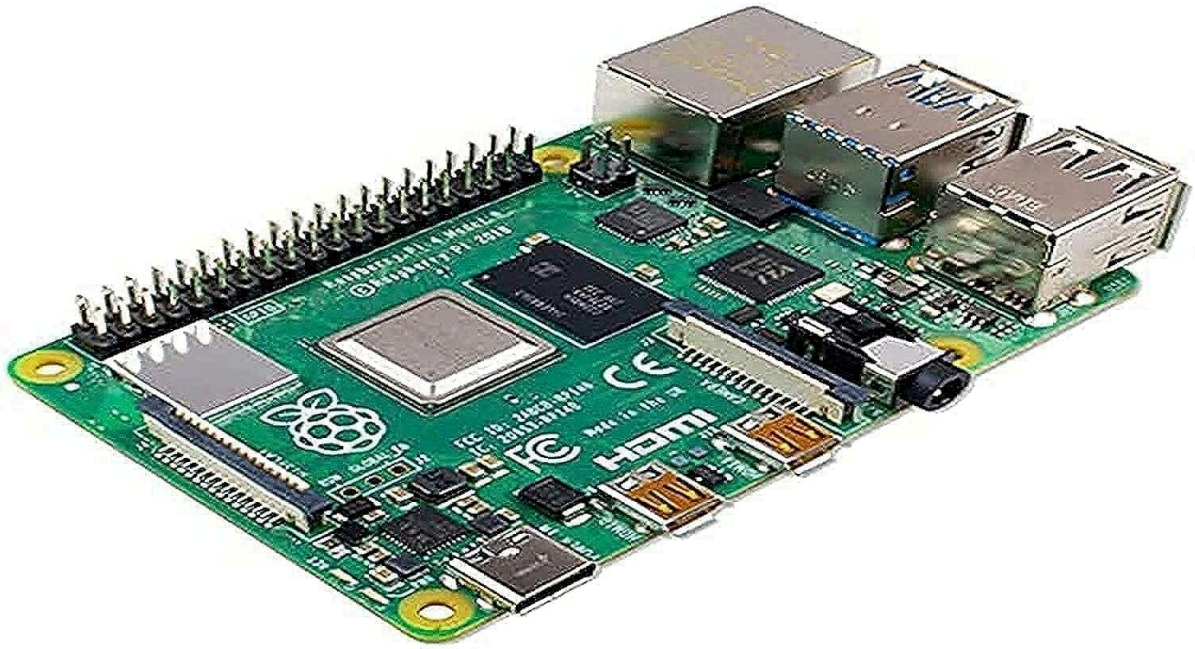
As it can be seen from the block diagram, we will power the Arduino board using a battery pack when using the bluetooth option for the controller. Originally, the team wanted to use WiFi, but after talking with our advisor we found that WiFi would have too much latency to implement a real-time, latency free experience for consumers. One important factor is that we would also need to run the board at a frequency that is not too high that it will consume the Battery pack's power quickly but also a fast enough frequency to process and output the data quickly. One of the key factors of our design is the sliding potentiometer. The potentiometer works as a variable resistor so given that a it is given a VDD of 5 volts, the slider will act as a resistor so that when it is full turn on, there is full current passing through, or when it is closed, no voltage will flow giving an analog value of 0. We will use an analog to digital converter or ADC to convert this signal in the Arduino and output it as an integer.

One of the last things to be implemented into our controller design is the transfer of information from the microcontroller to the game through bluetooth. This will be implemented after the controller is complete and can work with the game using USB.

For the game design aspect, we plan to use the GoDot game engine to develop the game. GoDot's code using GDscript is similar in syntax to python. The game will implement many feature movement, platform, unique character, and a variable jump feature that will input from the sliding potentiometer. Overall, we plan to integrate the custom game controller and the game to create a cohesive game experience.

Design Concepts

After deciding on our design approach at a high level, it is then necessary to zoom in on the individual design choices that will help to accomplish the goals of this project. To do so, we consider first, the options for our hardware design, and second, the options for our software design. Beginning with hardware design, the first consideration we came across was the choice of microcontroller. Microcontrollers used in previous courses, such as Texas Instruments' Tiva C series, were considered for their familiarity and excellent low-power performance. Arduino microcontrollers such as the Micro or Uno product lines were considered at the same time as Raspberry Pi microcontrollers, with both companies having extensive amounts of documentation for different applications. The ability to communicate wirelessly was also examined on each of these microcontrollers.



[1]

Figure 2: Raspberry Pi 4

Controller customization was another important consideration for our design. Options for controller input were considered, with commonplace methods such as joysticks, buttons and triggers considered to make up the core of the controller. For our unique spin on the recognizable controller design, new outputs to the controller such as an OLED display or a 7-segment display were discussed for their ability to present unique information to the user. Unique controller input options were similarly evaluated, from toggle switches to continuous sliders or dials. For housing the controller, we discussed using CAD software to develop a 3D printed housing. The actual design of the controller housing has specific considerations such as the button and joystick layout, the ease of use, or the comfort of holding the controller.

[2]

Figure 3: Example of 3D-Printed Controller

On the software side, the primary consideration was which game development software to utilize for implementing the game. Four game engines/libraries were considered: Unity, UnrealEngine, GoDot, and Pygame. Unity is the most widely used game engine, and thus has the the most resources, but uses C#, which would be new to the team. UnrealEngine might have the best implementation for 3D models and uses C++, which the team is familiar with, but 3D models might be too much depth for the time constraints. GoDot is completely free and has options for C++, and is more lightweight compared to the first two. Pygame uses Python, which the team is familiar with, but is restricted to 2D implementation and lacks the resources the other options have.

Concept Evaluation

To evaluate our options, we discussed the pros and cons of each of our options in the context of a project that must be completed in six months, if not quicker. Our last step to justify our decisions was to create decision matrices for the most important concept decisions.

The choice of microcontroller was focused around the following priorities: wireless connectivity, power efficiency, ease of use, and level of documentation. Microcontrollers that were previously used by team members, such as the Tiva C series, were highly efficient and familiar from previous coursework but lacked the wireless connection options that are necessary for our design goals. The Arduino Micro provides consistent performance in a much smaller package than other

options, but requires additional hardware for wireless communication. The Raspberry Pi line has built-in wireless capabilities and has plenty of documentation, but uses Python, which is less familiar to our hardware lead. The Arduino Uno has built-in wireless capabilities, and uses a variant of C++, which our hardware lead is proficient in.

Developing the design for our controller revolves around balancing innovation, practicality and comfort. Common inputs such as joysticks and buttons provide practical usability with a comfortable design, but lack the innovative edge to have a unique product. The idea of a display on the controller opens the door for innovative ideas, but aren't practical for most applications. New input options also serve to bring an innovative aspect to the controller, but binary inputs like toggle switches don't present a great deal of opportunity. Continuous input options such as a dial or slider present innovative options, but have concerns over practicality. As far as housing, options for shaping the controller revolve around the comfortable ergonomic shape that has been implemented by many popular game controllers.

The selection of game development was evaluated based on the metrics of compatibility with team expertise, suitability for intended gameplay, amount of resources available, and ease of use. Unity stands out for its amount of resources, but its reliance on C# presents a learning curve for the team. Unreal Engine has a similar number of resources with a C++ base, but could still be too heavy for the purpose of our intended gameplay. GoDot has a C++ base, but still presents a usability learning curve. Pygame has a comfortable base language of Python, is the easiest to use, but lacks in resources.

Concept Selection

In order to accurately weigh the options for our design choices, we created three decision matrices for three of our most important decisions. Figure 2 shows the decision matrix for the first concept decision, which was our choice of microcontroller

	(A) Decision to evaluate:	Choice of Microcontroller			
	Score (auto-calculated):	**16**	**19**	**16**	**21**
	(D) Options:	Arduino Micro	Raspberry Pi 4	Tiva C Series	Arduino Uno
(B) Factors of this decision	**(C) Rank of Factors (0-3)**	(E) Meets need? (0 - 3)	(E) Meets need? (0 - 3)	(E) Meets need? (0 - 3)	(E) Meets need? (0 - 3)
Wireless Connectivity	3	1	3	1	3
Power Efficiency	1	3	2	3	2
Ease of Use	2	2	1	3	2
Level of Documentation	2	3	3	2	3

Provided by Zapier.com, adapted from a tool by Idea Sandbox (idea-sandbox.com) [3]

Figure 4. Decision Matrix Used to Determine Choice of Microcontroller.

For this design decision, we considered our options for the microcontroller to be used to power the custom game controller. After considering some of the pros and cons of each, we decided to weigh these options using the metrics of wireless connectivity, power efficiency, ease of use, and level of documentation. We decided that the most important factor should be the microcontroller's ability to connect wirelessly, and gave it the highest weight accordingly. The microcontroller's ease of use metric considers the programming language relative to the team's expertise, and the programming software that accompanies it, while the level of documentation metric considers the amount of previous work the microcontroller has for similar tasks. These were weighed evenly, as we felt a successful design is equally reliant on both of these metrics. The final metric we considered was power efficiency, which we determined was important to consider, but ultimately lower on the priority list. Evaluating our options, we found the Arduino Micro has extensive amounts of documentation, and its smaller size means it has excellent power efficiency. However, the Micro does not have built-in wireless capabilities and would require external hardware to make that possible. The team also has not extensively used Arduino's language, so there is a potential learning curve there, but it's unlikely to be an issue because Arduino's language is based heavily on C++. The Raspberry Pi 4 has a similar amount of documentation, and does have built-in wireless capabilities for both Bluetooth and WiFi, but the accompanying setup and software would be completely new to the team. Its larger size also means it will have more power consumption than the Micro. The TIVA C series has been used by the team before, and has excellent power efficiency, but it has less documentation than the other brands and does not have built-in wireless capabilities. The Arduino Uno has great documentation and has built-in wireless communication. Same as the Arduino Micro, there is a potential learning curve, and its larger size means it consumes more power, but those aspects aren't enough to reduce its weighted value below being the best choice for our design.

The second design concept up for selection was the elements to include in our controller layout. Figure 3 below shows the decision matrix for this decision.

	(A) Decision to evaluate:	Controller Elements			
	Score (auto-calculated):	**17**	**13**	**16**	**18**
	(D) Options:	Joysticks/Buttons	Display (OLED/7Seg)	Toggle Switches	Continuous Inputs (Slider/Dial)
(B) Factors of this decision	**(C) Rank of Factors (0-3)**	(E) Meets need? (0 - 3)	(E) Meets need? (0 - 3)	(E) Meets need? (0 - 3)	(E) Meets need? (0 - 3)
Practicality	3	3	1	2	2
Comfort	2	3	2	3	3
Innovation	2	1	3	2	3

Provided by Zapier.com, adapted from a tool by Idea Sandbox (idea-sandbox.com) [3]

Figure 5. Decision Matrix Used to Determine Controller Layout.

For this design decision, we considered the aspects to include in the layout of our custom controller. Considering the designs of popular game controllers, we chose to weigh our options on the metrics of practicality, comfort, and innovation. The most important factor is practicality, as our priority is for the controller to be able to perform comparatively with other popular controllers. Comfort and innovation are equally important, as we want users to be familiar with how to operate the controller, but also to introduce unique elements that present a new challenge for intrigued users. The first group of controller elements we explored was the mainstay elements of console controllers: joysticks and buttons. For this controller to perform well for our game, these controller elements are necessary to provide a practical setup. The next idea we considered was a display on the controller itself, which provides great opportunity for innovation, but ultimately fails the test of practicality as users will not be able to look at the controller during intense gameplay. After this we considered toggle switches, which are not a common input on game controllers, but ultimately don't provide as much opportunity for innovation. The final consideration was a continuous value input, such as a dial or slider. This presents a learning curve for users, as such an input is not common in game controllers, but we don't feel this learning curve will be too difficult to overcome. This also provides much more room for innovation than toggle switches. Ultimately, our controller will include the two highest scoring elements groups: familiar concepts such as joysticks and buttons, and a continuous slider or dial input.

The final concept up for selection was the choice of game engine/library to use for housing the game design. Figure 4 below shows the decision matrix used to help with this decision.

	(A) Decision to evaluate:	Choice of Game Engine			
	Score (auto-calculated):	**18**	**12**	**20**	**19**
	(D) Options:	UnrealEngine	Unity	Pygame	GoDot
(B) Factors of this decision	**(C) Rank of Factors (0-3)**	(E) Meets need? (0 - 3)	(E) Meets need? (0 - 3)	(E) Meets need? (0 - 3)	(E) Meets need? (0 - 3)
Team Expertise	3	3	1	3	3
Suitability for Gameplay	1	1	1	3	2
Ease of Use	2	1	1	3	2
Amount of Resources	2	3	3	1	2

Provided by Zapier.com, adapted from a tool by Idea Sandbox (idea-sandbox.com) [2]

Figure 6. Decision Matrix Used to Determine Controller Layout.

For our final decision matrix, we considered our choice of game engine to build our game with. We chose to weigh this option on the metrics of team expertise with the coding language, suitability for our intended gameplay, ease of use, and amount of resources available. We chose for the most important metric to be the team's expertise with the coding language used, with ease of use and amount of resources holding moderate importance, and suitability for our intended gameplay as least important, given the simplicity of our game concept. UnrealEngine uses a familiar language, and has plenty of resources, but is likely too heavyweight for our purposes. Unity suffers from the same issue of being too heavy weight, while also using a less familiar coding language. Pygame is the easiest to use, and uses Python, which our team is comfortable with, but does lack in resources. Godot has slightly more resources than Pygame, but has a greater amount of setup required. Ultimately, we decided to use the highest scoring Pygame to begin development, with the idea of expanding with GoDot after setting up most of the game flow.

Expected Deliverables

The first primary deliverable is a custom-designed controller built using the Arduino Uno R4 microcontroller. This controller will feature three main inputs in the forms of a joystick, three buttons and a variable slider for analog inputs. The variable slider is the key innovation of our custom controller design and will enable precise control within the game. The controller will also support both USB and Bluetooth interfacing for versatile connectivity. It will also include a rechargeable battery for portability. All of the components for the controller will be housed in a custom 3D printed controller that is designed for ergonomics and durability.

The second deliverable is a fully functional game developed using Godot to showcase the features and capabilities of the custom controller. The game will be designed to communicate with the controller to receive real-time inputs and convert the inputs to in-game actions. Our unique feature from the custom controller will be incorporated into the game as a variable jump. This variable jump will be more interactive and provide more variations to gameplay. The game will draw inspirations from **Super Smash Bros**.

The third deliverable is a set of user manuals to guide customers through the custom controller such as setup, connectivity, and operations. Along with the manual, we will provide game instructions that will outline the mechanics for our game and how they integrate with the custom controller.

Current Deliverables

March 6, 2025

As per the project management sections, our team is on track with expected deliverables as of March 6. We have almost completed the game development and any future adjustments will trend towards visual aspects of the game and error testing. As of this update, our game has a fully functional character select screen that could use some improvements visually. As far as the game functionality, we have two

characters that have smooth movement with three capabilities: melee attack, ranged attack, and shield. The melee attack was implemented early in development, but as of this update we have incorporated an attack cooldown, as well as some hitbox adjustments. The ranged attack shoots a small laser-like projectile that will cause knockback when it hits another player, but less knockback than the melee attack. The shield action creates a force field around a character, during which the character can not be attacked, but the shielding character is unable to move while shielding. These three actions are set to individual keys for each player, but will be implemented into one button on the controller that is adjusted by the sliding potentiometer. Additional game updates include newly added friction/air resistance that will slightly slow a player that is attacked, as well as the ability for a player to move against knockback to avoid getting pushed off the edge.

As for the controller, we have the controller connected with one joystick, one button, and a sliding potentiometer. With our current implementation, the sliding potentiometer determines which button will be pressed given where the slider is at. For example, when the slider is to the left, it will output a "1" key to the computer, when it is in the middle, it will output a "2" and so on. There will only be an output when the button is pressed. Each of the numbers matches up with an action from the game such as the attack, block, and shoot actions. We tested the controller's logic via breadboard and moved all the connections to a perfboard after verifying that the controller is working. We currently have the internal workings for the controller and have begun the process onto 3D printing a housing for the controller.

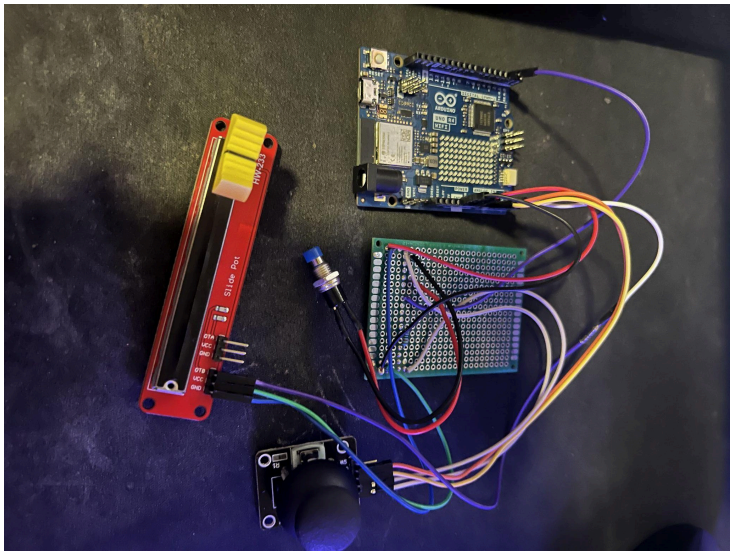


Figure 7. Current Progress of Controller, March 6th.

May 2, 2025

As of May 2nd, the day before the showcase, our design has been finalized, and no more changes will be made. Since March, the game design has had a number of updates. A title screen with a logo has been created at the beginning of the game loop, and both the character select and map select screens have been greatly improved with clear titles and better visuals. The graphics for

the attacks and the shield have been updated, as the melee attack now shows a clear effect, the projectile has been changed to a football, and the shield graphic is now larger to encapsulate the entirety of the character. The mechanics for the projectile and melee attacks have been tweaked, with the cooldowns being adjusted to provide balance. The code also went through some small tweaks to maximize compatibility with the controllers. As of our final team session, the controllers perform perfectly with the game design.

We have also finalized the design for two controllers that can be simultaneously playing the game. The 3D printed casing has been updated since its last iteration to greatly reduce its width and height, and also was slightly tweaked to improve the fit of our components. An opening was added to the casing to fit the wire after we had to move on from Bluetooth. The Arduino code also had to be updated to program two separate boards for our controllers. We have also reduced the size of the perfboard to help accommodate the reduced size of our casing. As of our final team session, both controllers are fully operational and are working with little to no delay. Due to technical difficulties, bluetooth was scrapped and removed from our final demonstration because we picked the Arduino R4 UNO WIFI with the assumption that the built in ESP32 module would be used as a bluetooth connection to the game. However, after further research and development, we found that the ESP32 was used to assist with the board's main processor's functions. One of these functions were the HID functions where we would emulate keyboard inputs to the game. We could have programmed the ESP32 module separately but this would cause us to lose access to the HID function which is an integral part to our controller.

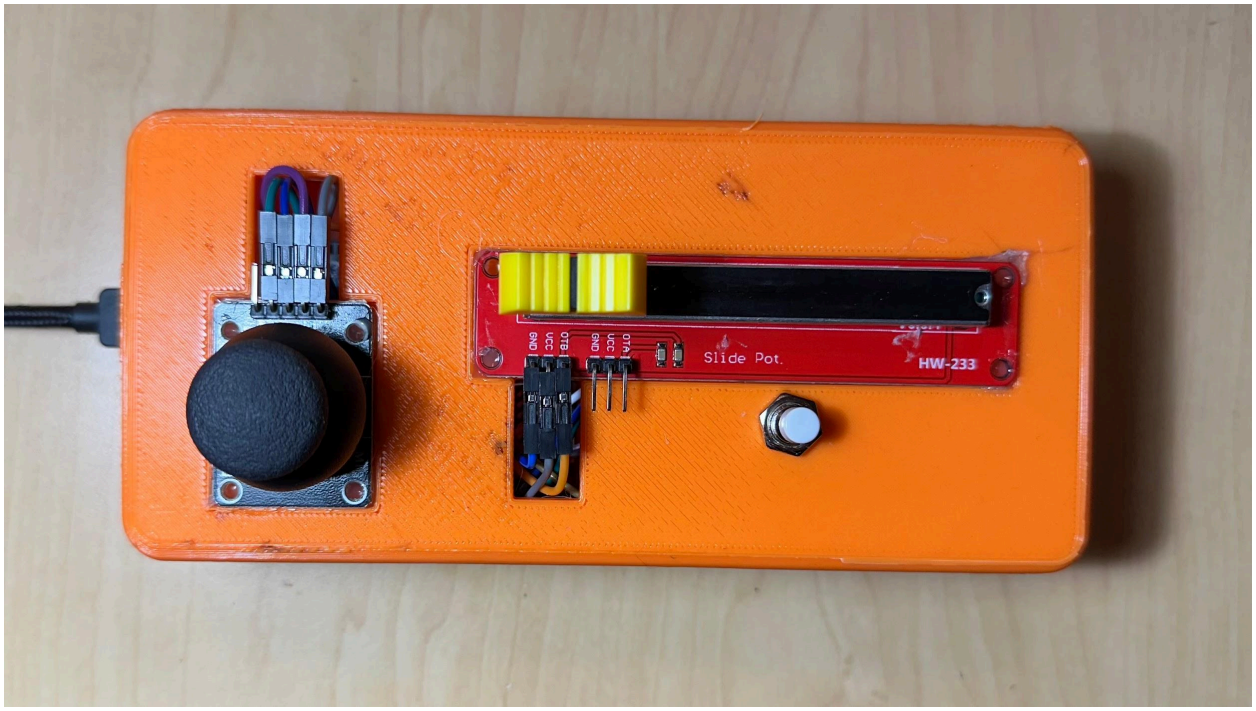


Figure 8. Current Progress of Controller, May 2nd.

Project Management

The table below is a timeline that the team will use to maintain a steady progress to advance the project. We will use the table to ensure all milestones are met before project deadlines. The main deadline that is to be met by the end of Fall 2024 semester is to ensure that research and parts are ordered on time for prototyping in the spring. All of the parts should be ordered by the start of december, which will allow us time to begin prototyping.

By the mid February, a prototype of the custom controller should be created without the 3-D printed housing. The custom controller should be interfacing via USB by this time and bluetooth will be implemented by another deadline. As well a prototype of the game with all of the main components/ features implemented to test with the custom controller. This deadline should align the team with the first project presentation and report deadlines around Feb. 12-14.

By the early march, the team should have the custom controller interfacing with bluetooth and the battery pack. The game development should be completed at this point with finishing touches. This will set the team up to complete the second progress report and presentation deadline by March 5-7.

Near the beginning of April, the custom controller should be completed as well as a prototype of the 3-D housing. Continuous verification between the game and the custom controller to ensure that our key concepts and ideas are implemented with the project's scope. Game should be completely done with minor updates and adjustments. This will allow the team to create our third presentation and report by April 2-4.

By Mid-April, the project should be wrapped up with the last iteration of the custom 3-D printed housing controller and a user manual should be created along with game instruction.

Table 1. Fall 2024 and Spring 2025 Project Deadlines

Deadline Date	Deadline Details
10/1/24	Brainstorming key features in project and game design
10/3/24	Begin researching components for custom controller, game engines.
11/1/24	Begin project design
11/8/24	Project design presentation completed
11/15/24	Completed parts list, all parts ordered
12/10/24	All parts received

1/20/25	Begin Work on building custom controller prototype and game development
2/9/25	Completed custom controller with interfacing via USB
2/10/25	Begin creating project presentation and report
2/12/25	Project Presentation 1 – Design/Early Prototypes (2-min video)
2/13/25	Project Report 1
2/14/25	Project Meeting
3/2/25	Controller and game MVP completed and begin working on project 2 presentation and report
3/5/25	Project Presentation 2
3/6/25	Project Report 2
3/7/25	Project Meeting
3/9/25	Begin integrating bluetooth and battery pack
3/10/25	Begin 3-D modeling design and finishing touches to the game.
3/29/25	Completed final 3-D model and begin project presentation and report 3
4/2/25	Project Presentation 3
4/3/25	Project Report
4/4/25	Project Meeting
4/10/25	Begin writing user manual, game instruction, poster, and product pitch.
4/23/25	Final Poster and Product Pitch
4/30/25	Project Presentation – Product Release
5/1/25	Project Report - Final
5/2/25	Project Archive

Budget

The majority of our budget will go toward the purchase of four Arduino Uno microcontrollers. The aim is to have two functional game controllers, and ordering two extra microcontroller

boards gives us freedom to simultaneously try multiple controller layouts and programs. We also have purchased a set of slide potentiometers and a set of joysticks to be able to begin our work on the controller, with the option of adding greater detail later. Additionally, we purchased two rechargeable batteries to be used on the final two controllers.

Monetary Budget

Item	Quantity	Unit Price ($)	Total Cost ($)	Purpose
Slide Potentiometer Set	1	12.99	12.99	Controller input for jump slider
Rechargeable Battery	2	30.0	60.0	Power supply for controllers
Game Joystick Set	1	12.99	12.99	Analog input for controller movement
Arduino UNO R4 WiFi Board	4	25.0	100.0	Microcontroller for controller systems
PCB Board (Estimated)	1	200.0	200.0	Custom circuit for improved integration
Total	-	-	385.98	-

Figure 9: Monetary Budget

Additionally, we calculate a rough estimate for the number of human-hours that this project will take to complete. For software design, which can begin immediately, we estimated 7 hours a week from the beginning of January to the end of March, as we hope to have the vast majority of the game design completed before the hardware. For hardware assembly, we estimated a higher 10 hours a week in total, starting at the end of January and going to the end of April. For documentation and meetings, we estimated 2 hours a week for the majority of the semester. The total estimate for human hours for this project can be found below.

Human-Hours Estimate

Task	Start Date	End Date	Average Hours per Week	Duration (Weeks)	Total Hours
Software Design	January 1	March 31	7	13	91
Hardware Assembly	Late January	April 30	10	14	140
Documentation & Meetings	January 1	April 30	2	17	34
Total	-	-	-	-	265

Figure 10: Human-Hours Estimate

Standards and Ethics

One big thing to think about as an Engineer is Ethics. While most people probably don't think Engineers think about it, it's one of the most important aspects of any project and when you get to the workforce. When thinking about how our project is free of ethical concerns the biggest concern we discovered was copyright. Since we are using school mascots that raises an ethical concern of copyright infringement. However, after doing some research and studying on the matter we learned that if we do not sell our product and we only use it ourselves then we are not breaking any copyrights, and we are still holding firm to our ethical beliefs within our project. We will be working on discovering other ethical aspects of our project that we can look into throughout the rest of this semester where our final project will have more than one example. However, at this time that's all we have regarding ethics. As for standards, standards are used to help provide a framework for teams to design products that are safe and consistent. A list of standards that apply to our controller design includes the following:

- IEEE 802.15 - Wireless Network Standards
- IEC/IEEE 82079-1 - Instruction of Use Standards
- ISO 9241 - Ergonomic Requirements Input Devices Standards

One of the important things that will need to be followed for our design is the IEEE 802.15 standard which is standard for wireless networks, which includes Bluetooth. Bluetooth is considered a wireless personal-area network or WPAN. IEEE 802.15 is the standard that network developers use to develop networks such as Bluetooth (WPAN), Internet of Things, mesh networks, and other communication devices. For the design of our controller, we are using Bluetooth to communicate between our controller and the game. There is nothing that needs to be done for our design to stay compliant with the standards as the Bluetooth protocol is already developed while complying with the standards set in IEEE 802.15.

Another important standard that is relevant to our project is IEC/IEEE 82079-1:2019, which describes the principles and general requirements for the preparation of information for use. As relative to our project, this pertains to creating a set of instructions for how to use our custom controller. This standard ensures that documentation is clear, accessible, and effective in guiding users through the safe and efficient use of a product. For our custom game controller, compliance with IEEE 82079-1 will help create an instruction manual that is comprehensive, but also easy to understand for the average user. The manual will include step-by-step instructions for any setup that might be required, instructions for regular use, and some general information for troubleshooting. The standard also discusses the importance of safety-related information, requiring that warnings be prominently displayed. For a custom controller that has not been reviewed by a professional team, we will need to provide sufficient warnings and safety information in our manual, as to ensure the safety of users. Additionally, the manual will be

presented to willing participants, after which they will be asked to perform simple actions on the controller, such as ‘move right’ or ‘perform a ranged attack’. They will be asked to say what they are thinking out loud as they try to perform the requested actions. From this testing, we will gain information about which tasks are found more difficult to new users, and thus which inputs should be more thoroughly explained in the user manual. By following these guidelines, we will provide users with an informative manual that ensures a straight-forward process of using the controller while following industry standards for safety warnings and information.

The newest standard that we tested with our controller is the ISO 9241, which is a standard for the ergonomics for non-keyboard inputs such as e-pens, joysticks, and controllers. To test the ergonomics and general feel for the controller, we ask participants to use our controller to play a game where the user would move around and collect a series of balls that would randomly teleport to a different place after pick up. After the participant has collected 10 balls, we would ask them a series of questions about the ergonomics, responsiveness, and tactile feedback, such as:

- How does the weight of the controller feel—too heavy, too light, or just right?
  Light 1 - 10 Heavy
- Are the grip surfaces easy to hold on to?
  Hard 1 - 10 Easy
- Would you be comfortable holding onto the controller for extended periods?
  Shorter 1 - 10 Longer
- Are you able to interact with the joystick, button, and slider without needing to readjust your hands? If so, are you able to reach the joystick, button, and slider comfortably, or does it feel like a reach?

These questions will help us further develop the controllers’ ergonomics and functionality. We hope to obtain quantitative data within a few weeks to improve the controller. The test will be done on the week of April 14th with a minimum of 5 participants.

Feedback from Users

Throughout this semester we asked our friends, peers, and technical advisor what their thoughts on the controller and game were and how exactly we can improve through iterations / if this project continued after this year. The main adjustment that kept getting changed on the controller side of things was the size of the controller. Like previously mentioned in the report the controller started off very thick and long in length causing it to be slightly uncomfortable which was one of the main issues mentioned by multiple people we talked to. Their experience was that if we wanted to pitch this controller as a business the thickness would have to be addressed. Throughout the design phase we iterated through multiple controller designs, which eventually led to the final design that not only shrunk the depth of the controller by around an inch and a half, but also the length by around an inch was really a huge improvement.

Another thing that users we let try the product prior to the design showcase mentioned it being hard to understand how many lives your player had left in the game. The reasoning was we didn't have the hearts from the final design in the code yet. This user feedback was very easy to address by adding 3 hearts on each corner of the screen representing both players' lives left. Another user added that instead of the game exiting whenever someone wins it should take you back to the homescreen to allow the option of changing characters or playing again, etc. Following that feedback we now have the entire game in a loop and it will redirect back to the initial home screen following a game. Using all this feedback gathered through users of the game/controller allowed us to make a better, more put together product for the design showcase.

References

[1] Raspberry Pi. "Raspberry Pi 4 Model B." [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/ . [Accessed: 03-Dec-2024].

[2] B. Houghton, "Alpakka 3D printed controller review," HackSpace Magazine, May 2020. [Online]. Available: https://hackspace.raspberrypi.com/articles/alpakka-3d-printed-controller-review. [Accessed: Dec. 3, 2024].

[3] M. Pinola, "The Decision Matrix: Make Better Decisions With This Spreadsheet Template," Zapier, Feb. 18, 2019. [Online]. Available: https://zapier.com/blog/decision-matrix-template/. [Accessed: Dec. 3, 2024].