

SCHOOL *of* BUSINESS AND TECHNOLOGY

Department of Engineering and Aviation Sciences

**Design of a VR Game-Based Learning Application**

**Zachary Allen**

Advisor: Dr. Zhang

Started: March 3, 2019

Design of a VR Game-Based Learning Application

By

Zachary Allen

Submitted to the Department of Engineering and Aviation Sciences in partial fulfillment of the requirements for the degree of Bachelor of Science in Engineering at the

UNIVERSITY OF MARYLAND EASTERN SHORE

Date

The author hereby grants the University of Maryland Eastern Shore permission to reproduce and distribute publicly, paper and or electronic copies of this document in whole or in parts.

Authors Zachary Allen

Signature

Date

Department of Engineering and Aviation Sciences

Certified by

List of Contents

[List of Contents 3](#_Toc24562138)

[List of Figures 5](#_Toc24562139)

[List of Tables 6](#_Toc24562140)

[Abstract 7](#_Toc24562141)

[1. Introduction 8](#_Toc24562142)

[1.1 Background/Motivation 8](#_Toc24562143)

[1.2 Objective 10](#_Toc24562144)

[1.3 Design Requirements 10](#_Toc24562145)

[1.4 Design Constraints 10](#_Toc24562146)

[1.5 Design Approach 10](#_Toc24562147)

[1.6 Standards 11](#_Toc24562148)

[2. Project Description 12](#_Toc24562149)

[2.1 System Description 12](#_Toc24562150)

[2.2 System Diagram 12](#_Toc24562151)

[2.3 System Functions 12](#_Toc24562152)

[3. Implementation Plan 14](#_Toc24562153)

[3.1 Tasks 14](#_Toc24562154)

[3.2 Team Organization 15](#_Toc24562155)

[3.3 Timeline/Milestones/Delivery Plan 15](#_Toc24562156)

[4. Implementation 16](#_Toc24562157)

[4.1 Implementation of Task 1. 16](#_Toc24562158)

[4.1.1. Implementation of Subtask 1A 16](#_Toc24562159)

[4.1.2. Implementation of Subtask 1B 16](#_Toc24562160)

[4.1.3. Implementation of Subtask 1C 17](#_Toc24562161)

[4.2 Implementation of Task 2. 17](#_Toc24562162)

[4.2.1. Implementation of Subtask 2A 17](#_Toc24562163)

[4.2.2. Implementation of Subtask 2B 18](#_Toc24562164)

[4.2.3. Implementation of Subtask 2C 19](#_Toc24562165)

[4.3 Implementation of Task 3 21](#_Toc24562166)

[4.3.1. Implementation of Subtask 3A 21](#_Toc24562167)

[4.3.2. Implementation of Subtask 3B 21](#_Toc24562168)

[4.3.3. Implementation of Subtask 3C 22](#_Toc24562169)

[4.3.4. Implementation of Subtask 3D 22](#_Toc24562170)

[4.3.5. Implementation of Subtask 3E 22](#_Toc24562171)

[4.4 Implementation of Task 4 22](#_Toc24562172)

[4.4.1. Implementation of Subtask 4A 22](#_Toc24562173)

[4.4.2. Implementation of Subtask 4B 23](#_Toc24562174)

[4.4.3. Implementation of Subtask 4C 23](#_Toc24562175)

[4.4.4. Implementation of Subtask 4D 23](#_Toc24562176)

[4.5 Implementation of Task 5 23](#_Toc24562177)

[4.5.1. Implementation of Subtask 5A 23](#_Toc24562178)

[4.5.2. Implementation of Subtask 5B 24](#_Toc24562179)

[4.5.3. Implementation of Subtask 5C 24](#_Toc24562180)

[4.6 Implementation of Task 6 24](#_Toc24562181)

[4.6.1. Implementation of Subtask 6A 25](#_Toc24562182)

[4.6.2. Implementation of Subtask 6B 26](#_Toc24562183)

[4.7 Implementation of Task 7 26](#_Toc24562184)

[5. Conclusion (Discussion and Future Plans) 27](#_Toc24562185)

[Acknowledgment 28](#_Toc24562186)

[Appendix 29](#_Toc24562187)

[A. Source Code. 29](#_Toc24562188)

[1. Basic Enemy.cs 29](#_Toc24562189)

[2. Basic Enemy Health.cs 29](#_Toc24562190)

[3. PlayerHealth.cs 30](#_Toc24562191)

[4. Boss Enemy.cs 30](#_Toc24562192)

[5. Boss Health.cs 31](#_Toc24562193)

[6. Aggroed.cs 32](#_Toc24562194)

[REFERENCES 33](#_Toc24562195)

List of Figures

[[Figure 1 - People Playing a Game Together 8](#_Toc23335121)](#_Toc24562196)

[[Figure 2 - An Example of a VR Headset 9](#_Toc23335121)](#_Toc24562197)

[[Figure 3 - Design Approach 11](#_Toc23335121)](#_Toc24562198)

[[Figure 4 - The Game's System Diagram 12](#_Toc23335121)](#_Toc24562199)

[[Figure 5 - The Game's Original Structure 16](#_Toc23335121)](#_Toc24562200)

[[Figure 6 - The First Room 17](#_Toc23335121)](#_Toc24562201)

[[Figure 7 - First Hallway 18](#_Toc23335121)](#_Toc24562202)

[[Figure 8 - Second Hallway 18](#_Toc23335121)](#_Toc24562203)

[[Figure 9 - Third Hallway 19](#_Toc23335121)](#_Toc24562204)

[[Figure 10 - Broken 4th Hallway 19](#_Toc23335121)](#_Toc24562205)

[[Figure 11 - Eyes of the Eldritch 20](#_Toc23335121)](#_Toc24562206)

[[Figure 12 - Overhead View of Last Hallway 20](#_Toc23335121)](#_Toc24562207)

[[Figure 13 - The Decimal to Binary Puzzle 21](#_Toc23335121)](#_Toc24562208)

[[Figure 14 - T-Posing Model of the Security Enemy 24](#_Toc23335121)](#_Toc24562209)

[[Figure 15 - Two Workers Slacking 25](#_Toc23335121)](#_Toc24562210)

[[Figure 16 - Overhead View of the Working NPC 25](#_Toc23335121)](#_Toc24562211)

[[Figure 17 - The Plane to Space 26](#_Toc23335121)](#_Toc24562212)

List of Tables

[Table I - Planned Project Timeline 15](#_Toc23335282)

Abstract

By the end of the project, summarize the project into short text and put here.

1. Introduction

This project will utilize virtual reality and game-based learning to create an ominous puzzle game with shooter elements akin to Call of Duty. The end goal of this project is to provide a game that both allows the players to learn elements taught in a digital-logic circuits classroom and give them time to unwind battling braindead enemies hindering their progress.

## Background/Motivation

For some, video games are a hobby in their everyday lives. Whether they are playing games to relieve themselves from stress after a long day; learn something new, or to entertain an audience on the internet, the video game market can provide enjoyment for just about anyone. Ultimately, the purpose of video games is to provide fun, enjoyment, and replayability for the consumers. Video games can also serve as a means to bring various people together like shown in Figure 1. People from various backgrounds can come together through a single hobby they enjoy.



Figure - People Playing a Game Together

For others, video games are long, strenuous projects that must be built as a development team for consumers to buy. In an ideal world, developers would work throughout the week with little overtime and no development crunch to release the right game for the right audience. Of course, even a released video game does not get produced perfectly. Issues may pop up after the video game reaches the shelf.

Today, video games are constantly being patched. Patches in video games serve two major purposes: fix existing bugs and/or add more content. Bug fixes typically come as a result of consumer feedback. The consumer can provide bug reports to the developers to tell them what is not working properly. These reports serve as a basis for a “bug fix patch.” However, this process remains throughout the game’s shelf life. Often times, fixing one bug could expose another bug and so on and so forth.

Game patches can also add content to the game itself. This practice has existed since the era of MMORPGs (Massively Multiplayer Online Role-Playing Games) where patches would add content for players of all levels and even expand upon the base game with expansions. Today, add-on patches end up serving as an excuse to release games unfinished, effectively time-gating the content to artificially increase the shelf life of a game. The best developers to follow utilize patches to fix bugs as they get reported and provide additional content that is not mandatory for the base game but enhances the game for better enjoyability.

While some video game developers are major corporations creating high-budget games designed to bring in a lot more money then what went into the game, others are small teams of people creating a passion project to share with the world. No matter who the developers are, most video games are crafted with care, attention, and intent to produce an impact on their fans. With the introduction of more efficient video game crafting tools, anyone can join the video game development with the right amount of practice and dedication.

For instance, one tool people could use to create video games is Unity. The unity website provides a free version for anyone to get started with and more advanced versions should they decide to make a career out of producing games. Unity also provides an asset store for people to use to help craft their game to suit their big design. They also have an active forum and classes to help aspiring developers learn the system and produce a game in no time.

Suppose someone wanted to create an edutainment game on Unity. Some people may think to themselves, “Why bother? Are these games only meant for young kids?” On the contrary, edutainment games do not have to be restricted to just the young. There is a potential market to bring such video games to high school and even college. A game can be made to teach more advanced concepts while providing moments to turn off their brains after thinking on a puzzle for so long.

Perhaps an immersive video gaming experience could help breathe new life into edutainment video games. With virtual reality technology breathing new life and brimming with more potential than ever before, edutainment games could seek a new home in VR gaming. All they would need is a VR headset like the one shown in the figure below. This surging technology could provide more means to teach students more educational subjects than ever before while including breaks from learning.



Figure - An Example of a VR Headset

This idea of combining video gaming with education is called game-based learning, also known as GBL. An example idea of a game-based learning application is to use a puzzle game similar to the 3D *Legend of Zelda* games combined with a shooter game such as *Call of Duty*. This combination allows the player to solve puzzles teaching various concepts taught in class such as a class on circuitry while also letting them unwind between puzzles by mindlessly shooting down enemies. Ideally, the GBL application would have an equal mix between shooting and learning so the player is learning without being burnt out from just learning without a moment to relax.

## Objective

The overarching goal of the project is to create a game for engineering students learning digital logic circuits that both teach them elements of the class and engages them in the game world.

## Design Requirements

The game will take about half an hour to complete. There will be characters and objects to interact with. These interactions are either chatting with friendly NPCs, knocking out enemies, or using objects available to complete a given puzzle. The game itself will have 4 puzzles to solve with permanent rewards for each puzzle to assist the player. The game will also utilize teleportation to work over the limited play area virtual reality provides. The game will end with a 5-question quiz that will serve as the endgame. Essentially, this design will result in a puzzle + shooter action game with unnerving elements throughout.

## Design Constraints

The virtual reality technology provided by the institution provides the biggest restraint in this project. The labs utilize the HTC Vive virtual reality headset; and therefore, the game must run with this headset in mind. The headset itself is wired, so the game must be built to provide functions to work over the limited play area as in the SteamVR’s teleportation feature.

Because of the nature of virtual reality gaming, prolonged exposure can result in headaches. This will severely limit the duration of the game itself. Considering this restriction, the game’s duration ended up being about 30 minutes.

## Design Approach

The design for this game is split into three intertwined parts: the story, the world, and the puzzles. Without a story, there is no reason for the world to exist. Without the puzzles, there is no reason to create this game utilizing GBL. Without the world, there is no gam. In order to create the game, the story must be established.

In this case, the story involves the player winding up deep within an engineering facility. They must get a feel of their situation and try to escape the facility. Along the way, the player will encounter two types of enemies. The first are the previous people who were rendered braindead from struggling to solve the puzzles. The second is the security guards sent by the facility’s boss as a result of fear for the facility’s secrets being exposed. After being confronted by the boss, the player will be challenged to one last quiz and then sent to an unknown facility somewhere in space.

The world is developed based on the story. The world takes place within a hidden testing chamber of an engineering facility. The facility was purposed to teach their trainees digital logic circuits. However, years have passed, and no one has solved all four of its puzzles. These trainees have become braindead and with attack anyone who gets in their line of sight. Over time, the player will learn than the facility is holding a secret the boss does not want to have leaked. The boss will send security to stop the player the closer they are to the end.

From there, the basis for what the player will see can be made. There will be a visible health bar with the health having a max value that increases twice throughout the puzzles. The puzzles will also award players with two weapon upgrades just in time for the elite enemies towards the end. Because of the limited movement space, the player will also be allowed to teleport to bypass this limitation. Since each room and hallway will be its own scene in Unity, each time a new scene is loaded, that scene will become the respawn point should the player lose their health points. The approach will be highlighted in the figure below taking all the points from this section of the report into consideration.

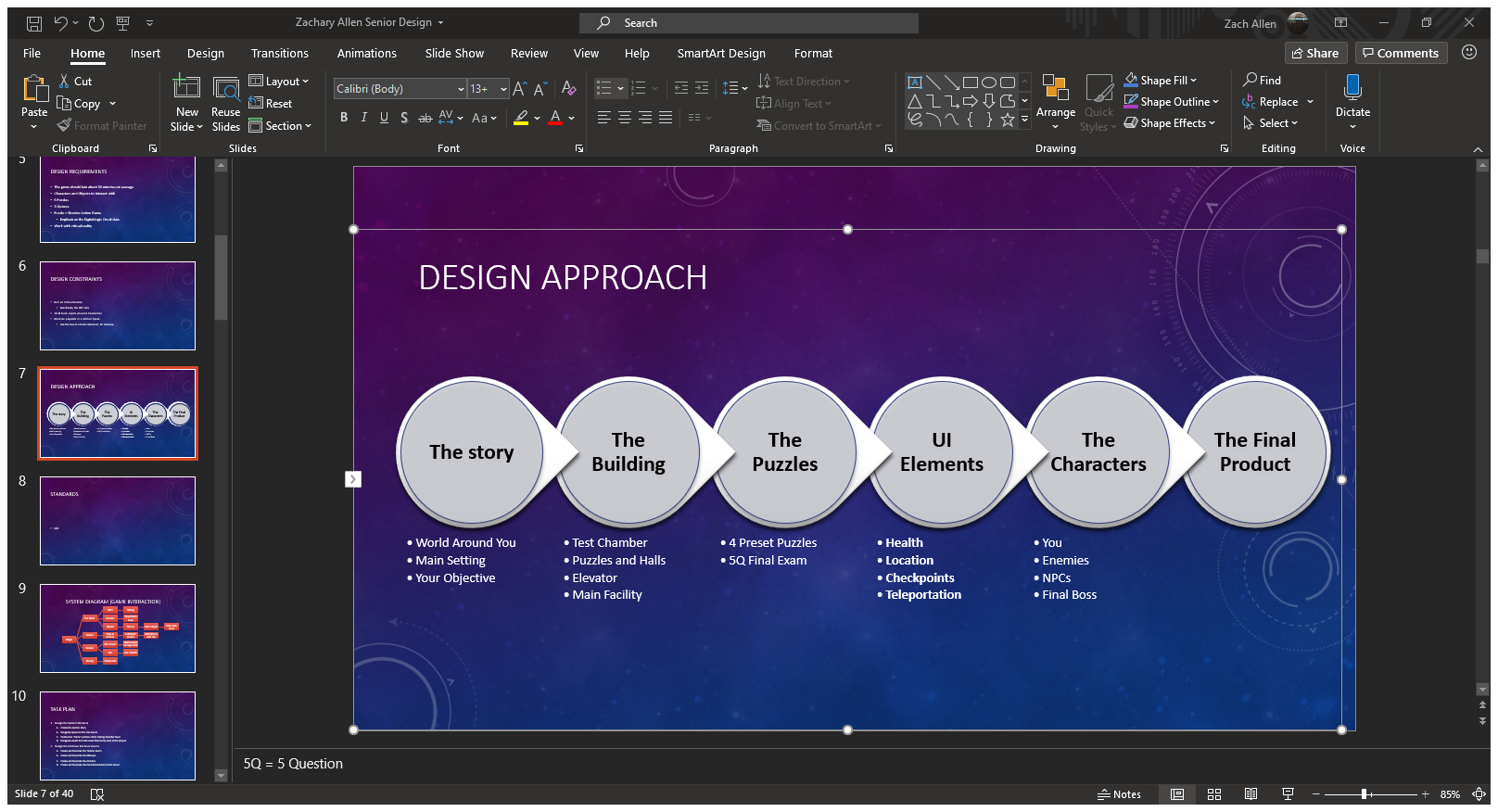


Figure - Design Approach

## Standards

There are no standards the project must comply with.

1. Project Description

## System Description

This game will utilize Unity and virtual reality. The player’s movements combined with their controller inputs dictate how the game is controlled. There will be button inputs for helping with puzzles, beating down enemies, and teleporting around rooms larger than the VR play area.

## System Diagram

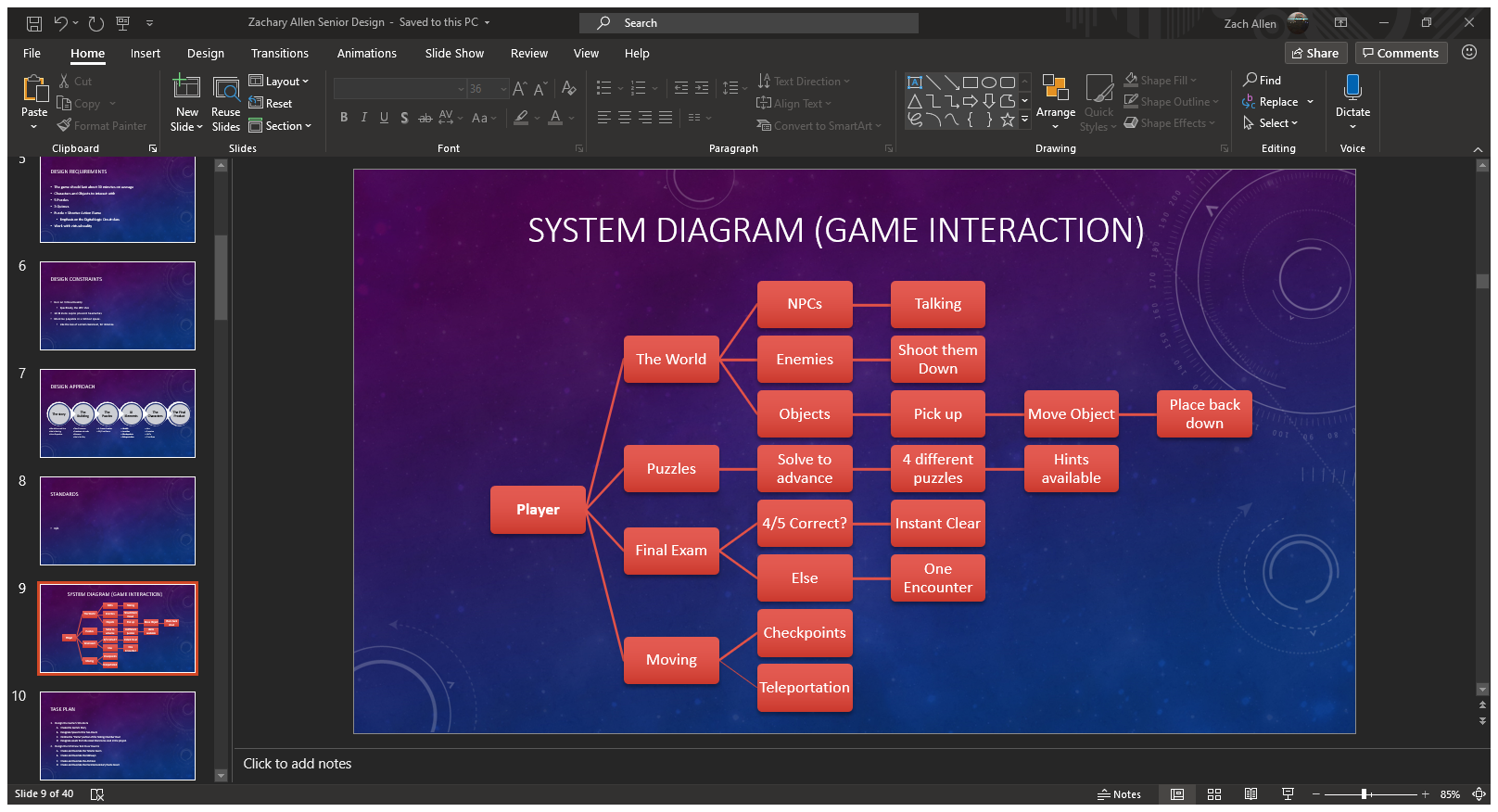


Figure - The Game's System Diagram

## System Functions

The game will feature four different interactions that are intertwined throughout the game. For interactions with the world, the player can either talk with friendly NPCs, take out enemies before they are taken out themselves, or interact with specific objects to move them around the play area. Other objects are dropped by enemies to help recover any damage taken throughout the adventure.

Puzzle interactions are simple. The player must clear each of the four puzzles to continue. A hint will be provided for each puzzle to guide the player in the right direction. After clearing the puzzle, the player will be rewarded with upgrades whose potency is based on completion time with faster puzzle-solving leading to better rewards.

After solving the puzzles, the player will be placed under a Final Exam by the antagonist. There will be 5 questions that change be changed in the main menu by the professor. The player must answer 4 of the 5 correctly to clear the game. Otherwise, they must face 3 elite enemies and defeat them to clear the game.

Movement is a critical aspect of this game. While the player can move around freely, the world will be larger than the play space virtual reality allows. To compensate, the player will be allowed to teleport throughout the world. There will also be a system to load the player back to the beginning of whatever room they are in to prevent having to start from the first room after each death.

1. Implementation Plan

## Tasks

* Task 1: Design the Game’s Foundation
  + Subtask A: Develop the Game’s Plot
  + Subtask B: Establish the Game’s Structure
  + Subtask C: Designate Assets to Create the Project
* Task 2: Design the Hallways of the Game
  + Subtask A: The Starting Room
  + Subtask B: Basic Hallways
  + Subtask C: Advanced Hallways
* Task 3: Design the Puzzle Rooms
  + Subtask A: Decimal to Binary
  + Subtask B: Minterms and Maxterms
  + Subtask C: Boolean Algebra
  + Subtask D: K-Map
  + Subtask E: D Flip Flop
* Task 4: Create the Game’s UI Elements
  + Subtask A: Enable Teleportation
  + Subtask B: Create an HP System
  + Subtask C: Establish Interactivity with the Objects
  + Subtask D: Make the Checkpoint System
* Task 5: Design the Game’s Enemies
  + Subtask A: Braindead Trainees
  + Subtask B: Regular Security Patrol
  + Subtask C: The Boss’ Elite Guard
* Task 6: Create the Main Lobby
  + Subtask A: A Live Office
  + Subtask B: The Final Exam
* Task 7: Verify the Program
  + Subtask A: Ensure Every Event Works Properly
  + Subtask B: Ensure Each Scene is Loaded One After Another
  + Subtask C: Debug the Game

## Team Organization

This project will be completed by one person. As such, Zachary Allen will be solely responsible for the completion of each task shown in the timeline. The timeline below shows what is to be expected after each week until it is time for the final presentation

## Timeline/Milestones/Delivery Plan

Table - Planned Project Timeline

|  |  |  |
| --- | --- | --- |
| Week | Tasks | Details |
| 1-2 | 1A, 1B, 1C | Create the Foundation for the Game |
| 3-4 | 2A, 2B, | Design the Hallways between Puzzles |
| 5 | 2C | Create the final hallways |
| 6-7 | 3A, 4A, 4C | Start the Puzzle Room and UI Elements |
| 8-11 | 5A | Create the First Enemy |
| 12-14 | 4B, 4D | The Rest of the UI is made |
| 16-18 | 3B, 3C, 3D | The Other Puzzles are developed |
| 19 | 3E, 5B, 5C | The Harder Enemies are made |
| 20 | 6A, 6B | Create the endgame |
| 21-22 | 7A, 7B, 7C | Verify that the entire game works. |

1. Implementation

This section of the report will go through each task and mention how they have been implemented throughout the project.

## Implementation of Task 1.

This task is meant to develop the cornerstone for this entire game. This means the story, structure, and assets used are determined in this task. By establishing a foundation for the game, the rest of the tasks can follow suit as the game is being developed.

### Implementation of Subtask 1A

This task established the plotline for this game. For this adventure, the player is a trainee who got trapped within the facility’s training chambers. The player wakes up and must try to leave the chamber. They will solve the facility’s digital logic circuit puzzles and take down other trainees who became braindead throughout the testing process. As the game goes on, the player will learn that the antagonist did not mean for anyone to clear the chambers and will start to send his security personnel to stop the player. After confronting the boss and clearing his final exam, the player will be sent away so the boss’s secrets can remain hidden from the public eye. Where the player ends up is the basis for another GBL Senior Design Project involving puzzles in circuits.

### Implementation of Subtask 1B

Now that the game’s story is set, the task to develop the game’s structure is afoot. Originally, the game would take place in one large chamber with winding hallways connecting to each puzzle room as shown in the figure below. However, this idea was eventually scrapped after realizing the player could not realistically navigate a large, winding facility.

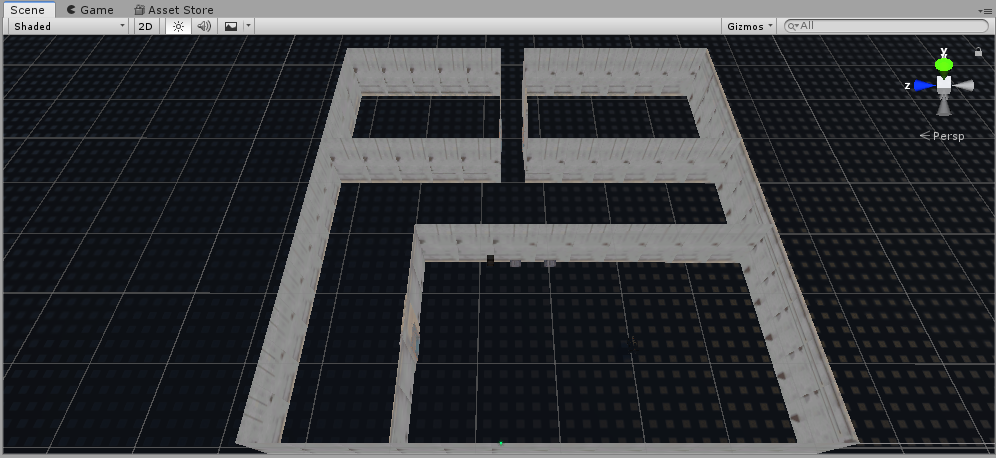


Figure - The Game's Original Structure

Therefore, the game took on a new structure. This structure takes full advantage of loading scenes in and out while compensating for the limited virtual reality movements. Each room, therefore, became its own scene in the final project with a script loading between each room as the player leaves the room before. This also includes the hallways connecting each room. Therefore, teleportation must be used as explained later on in the Implementation section.

### Implementation of Subtask 1C

With the story and structure of the game established, the time came to establish the assets the project will use. All of the assets will be mentioned in the Acknowledgements section with credit to their creators. One important asset to mention, however, is the SteamVR asset. This asset allows this project to run through its full potential by enabling this game to run in virtual reality. This asset almost makes movement through larger rooms possible with teleportation.

## Implementation of Task 2.

This task begins building the rooms the facility will go through. To start things off, the player will begin in the first room of the facility and will then go through a dark hallway to get to the first puzzle. With the structure being room then hallway and then room and so on, the first step would be to create the first room and the hallways to navigate.

### Implementation of Subtask 2A

The game begins in a surprisingly lit room deep inside the testing chambers. In the lore, these rooms are where trainees begin to complete their run through the facility to clear each puzzle. Such a room is implemented as shown in the figure below. The player will use this room to get familiar with the controls and proceed to the first hallway once acclimated with the controls.

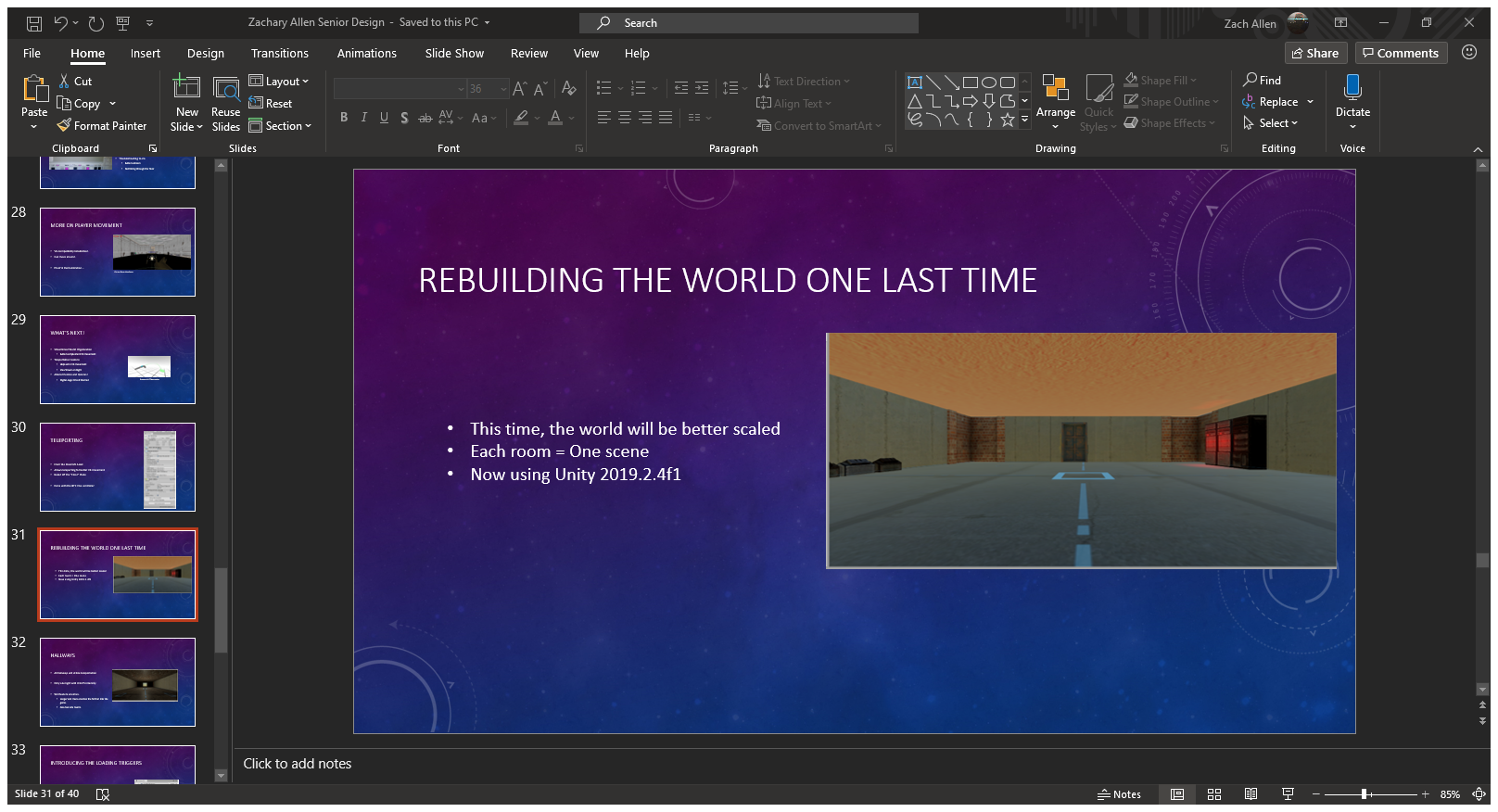


Figure - The First Room

### Implementation of Subtask 2B

Once the first room was built, the next objective was to create the 5 total hallways the game will have. There will be 3 simple-to-navigate hallways and 2 more advanced ones as the game progresses. This subtask focuses on the first 3 hallways. The first hallway is shown in the caption below. This room is very dark compared to the starting room and introduces the player to the first enemy of the game. However, this hallway is short, and the player can easily circumvent the enemies. Clearing this hallway grants the player access to the decimal-to-binary puzzle.

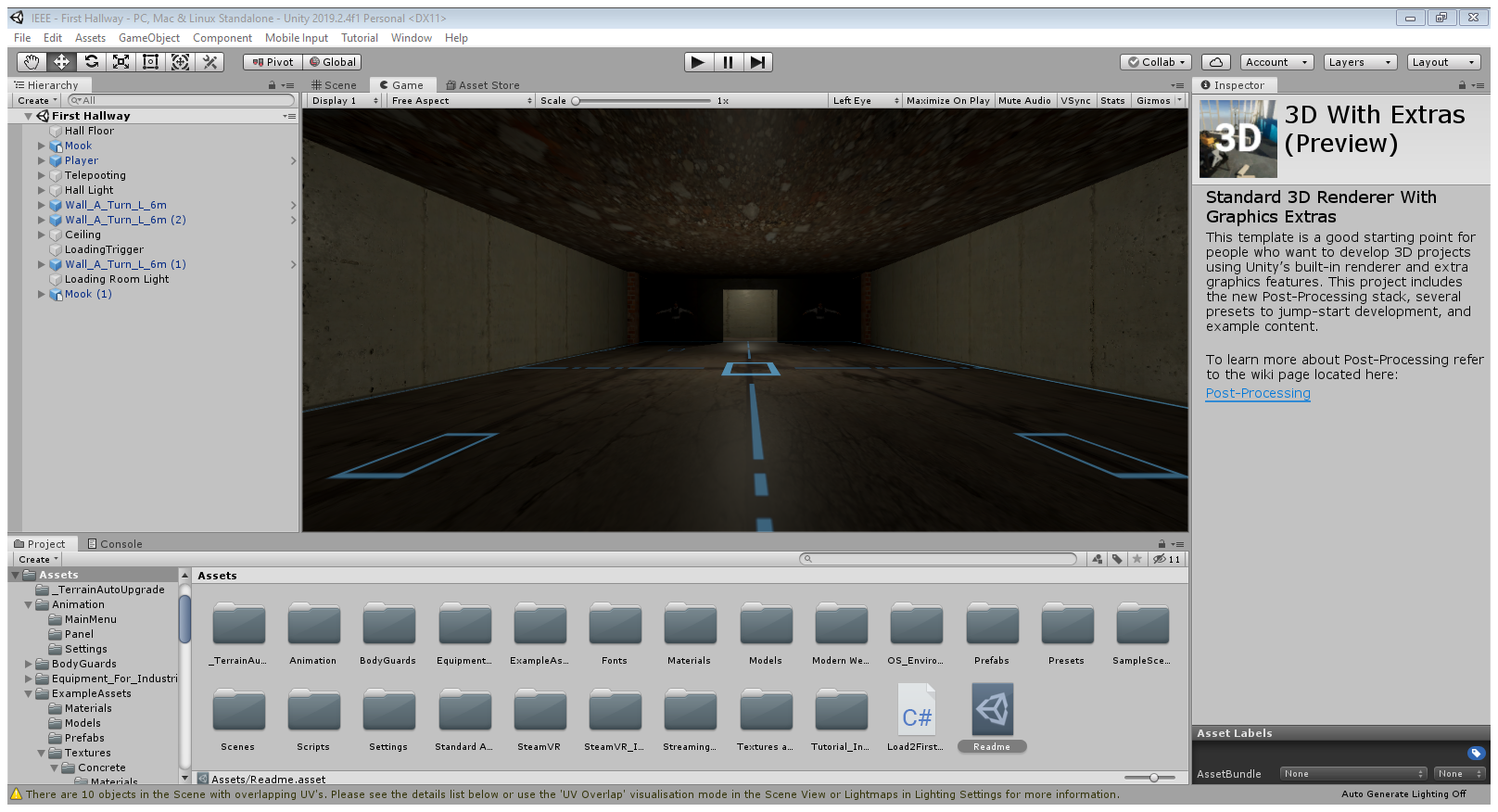


Figure - First Hallway

The second hallway is shown in the next figure below. This hallway is longer than the first one and even has a brief outdoor view. Like the first hallway, this one is a straight shot from point A to point B to get to the next puzzle that will involve minterms and maxterms.



Figure - Second Hallway

The last simple hallway is shown below. This one bends at a 90-degree angle. This hallway will introduce the second enemy type of the game where the story escalated to the boss starting to actively keep the player in the testing facilities. These enemies will have slightly higher health than the first enemy type and are more ruthless. Clearing this hallway sends players to the Boolean algebra puzzle.

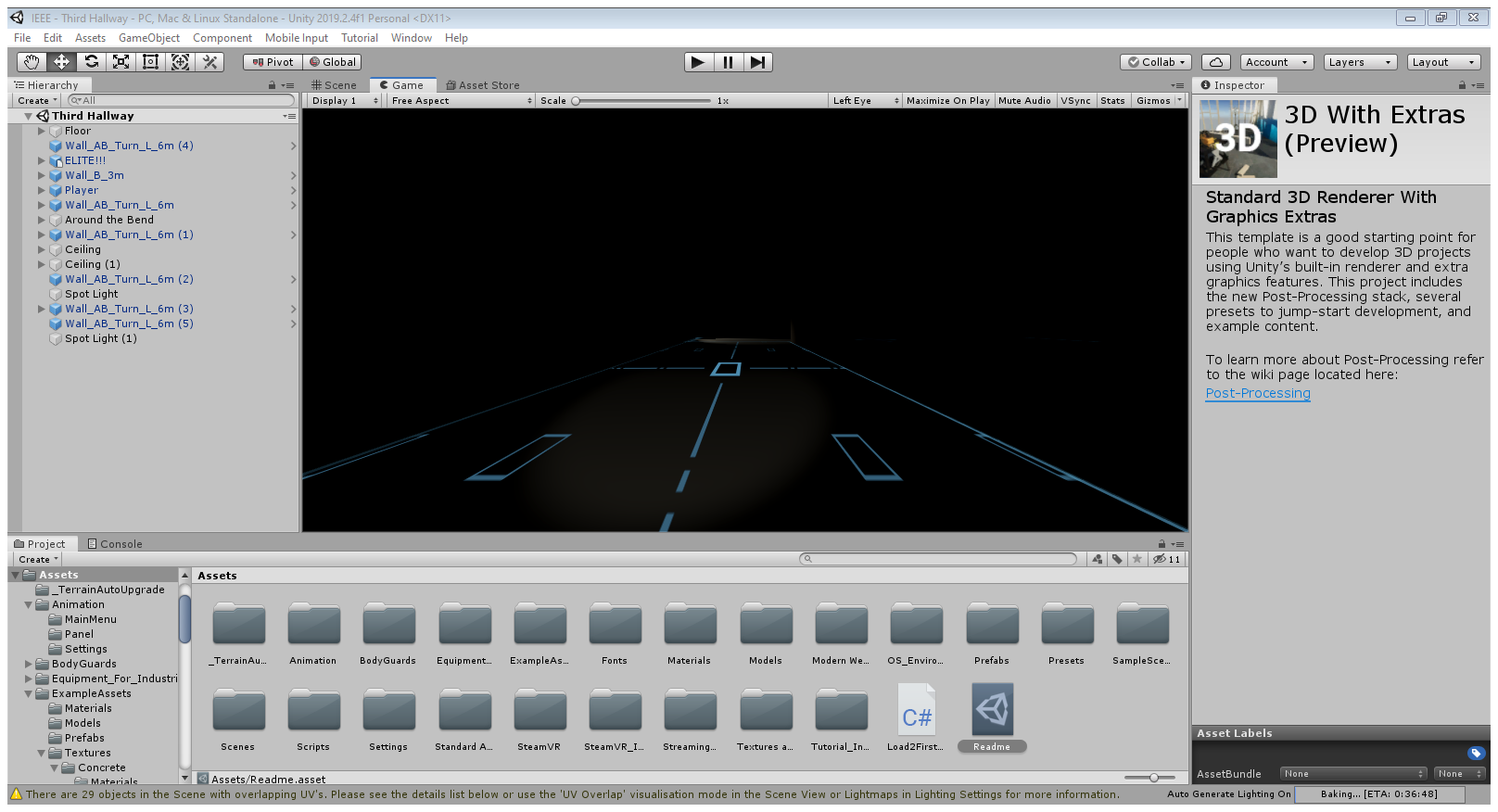


Figure - Third Hallway

### Implementation of Subtask 2C

As mentioned in the above section, this game has 5 hallways. This subtask will implement the final two hallways the player will encounter. The first one is shown below. Here, the player must teleport between chopped off sections of flooring and avoid falling to meet the past victims of this accursed hallway. Once they clear this platforming section, they are granted access to the K-Map puzzle.

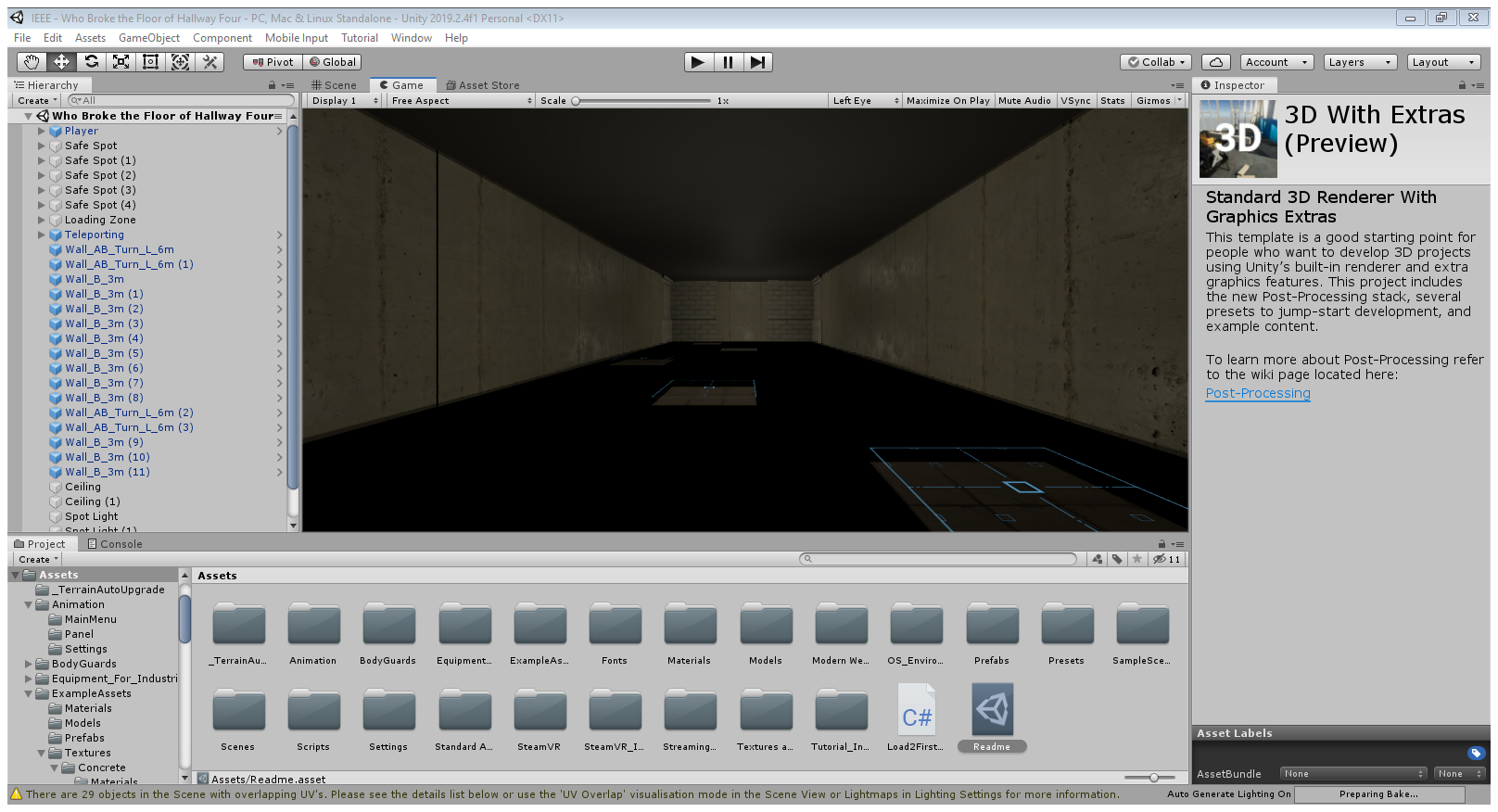


Figure - Broken 4th Hallway

One more thing to note about the fourth hallway is the creature showcased in the following figure. This is the “Eyes of the Eldritch.” It is an unknown entity with unknown origins and no natural explanation as to why it is under this hallway and why it broke the hallway. Many unfortunate souls have fallen to the eldritch in their journey of the chambers. One has to wonder if the player will be next….

A picture containing floor, indoor, sitting, room

Description automatically generated

Figure - Eyes of the Eldritch

The last hallway is represented by the figure below. This hallway features both elevation and broken floors. The end of the hallway has a boss battle preventing the player from accessing the final puzzle and the main lobby. Details about the boss is detailed in its respective implementation section.

A picture containing indoor, sitting, table, small

Description automatically generated

Figure - Overhead View of Last Hallway

## Implementation of Task 3

Once the hallways are built and established, the next task of creating the puzzles can be implemented. The purpose of these puzzles is to test the player’s knowledge of digital logic circuits. The first four puzzles also come with a reward system that will help the player in their journey. Completing the final puzzle will grant the player access to the main lobby and the end game.

### Implementation of Subtask 3A

The first puzzle has the player doing a simple conversion from decimal to binary. This puzzle has 2 parts: the first conversion and then taking the 2’s complement of the first number. The player will need to know numbers from -512 to +1023 to be able to do the conversion since the puzzle uses a 10-bit system. For this puzzle, the decimal number that will need to be converted to binary and its 2’s complement is 500 in reference to song lyrics posted on the wall.

The puzzle is shown in the figure below. The puzzle consists of 10 lights with respective buttons below. Red lights represent a “0” and green lights represent a “1.” The puzzle defaults to all values being 0. Once the correct combination of 1’s and 0’s is made, the door opens to the 2’s complement puzzle. Completing that puzzle allows the player to enter the next hallway and grants them an extra 25 damage to their attacks. This goes hand in hand with the user interface shown in a later implementation.

A sign on a brick building

Description automatically generated

Figure - The Decimal to Binary Puzzle

### Implementation of Subtask 3B

The second puzzle the player will encounter is a simple minterm and maxterm puzzle. The player will be presented with a truth table, 8 sums of the truth table itself as movable boxes, and two baskets labelled “Minterm” and “Maxterm.” By placing each box in the right basket, the player will be able to advance to the 90-degree hallway and continue the game. The player will also be rewarded with 25% reduced damage that will stack with another puzzle’s reward later on.

### Implementation of Subtask 3C

The third puzzle players will encounter involves utilizing Boolean Algebra to simplify two functions presented. The room will feature two monitors for each of these equations and the player must input the simplified equation into the monitor. Each wrong guess will cause a flat 10 damage penalty, so the player must plan accordingly before throwing random guesses into the air. Completing this puzzle will reward the second damage boost and allows them to visit the Eyes of the Eldritch.

### Implementation of Subtask 3D

The next puzzle uses another truth table with 8 lines. However, the room will have 8 squares on the floor. Below this room is the Eyes of the Eldritch. Any incorrect input will drop the player to its eternal deadly stare. The player must step only on the squares that outputs a 1 from the truth table. Therefore, the player is using their feet to create a K-map of the provided truth table. This puzzle presents the final reward of the second 25% damage reduction, just in time for the last hallway and its impending boss battle.

### Implementation of Subtask 3E

## Implementation of Task 4

Once the world is established as well as its puzzles, the user interface elements must be created. This includes the way the player moves around the world, how the world can be interacted with, and how to prevent the player from have to restart the *entire* game after each game over. Since the UI elements of this game is done through C# code, these subtasks will refer to the Appendix for each of the subsections’ respective script file.

### Implementation of Subtask 4A

Because of the limited mobility of the HTC Vive and how large each of the scenes in the game are, basic moving will not be enough to navigate the game. The solution to this problem is to apply teleportation to the game. This is established by using the SteamVR asset in Unity. For the sake of this project, teleportation is achieved by duplicating the floors of each room and establishing a teleportation layer for each floor that has collision. The teleportation itself is performed by using the trackpad of the Vive controllers. This action of teleportation will also be utilized to take players between each room and hallway of the game.

### Implementation of Subtask 4B

A game is no fun if there is no risk involved with how one plays it. That is why games usually come with a health system to encourage players to think carefully and provide a boost of adrenaline should they make it out of a rough encounter by the skin of their teeth.

This game will also come implemented with a health system. The player will have a base health of 100 with damage modifications that reduces their damage from enemies after solving the second and fourth puzzles. Enemies will also drop healing caplets to heal the player from any damage taken. What happens when the player reaches 0 health will be explained in subtask 4E.

### Implementation of Subtask 4C

In order for the puzzles to work, there must be an ability to interact with the objects in the world. For instance, this subtask implements the ability to push the buttons in the decimal-to-binary puzzle, picking up boxes for the minterm and maxterm puzzle, and interacting with non-playable characters in the game. This task takes advantage of using box colliders and triggers to ensure each puzzle and interaction works as intended.

### Implementation of Subtask 4D

The final UI element in the game is the checkpoint system that the game will run by. The game will run in a way that the respawn scene should the player die will be the same scene that they are currently on. So, if the player dies falling into the Eyes of the Eldritch, they would not have to keep doing the first three puzzles and hallways just to try again. Of course, the player will respawn with all of their health and any progress made against enemies in the room will be saved.

## Implementation of Task 5

A game is boring if the only obstacles are brick wall puzzles that interrupt the pacing of the game. For this reason, this task implements 3 types of enemies into the game. The enemy type the player faces changes with how far they are within the game itself. The scripts that the enemies run on can be seen in the Appendix below.

### Implementation of Subtask 5A

The first enemy players will face are the braindead engineers who have failed the trials the player is currently going through. These enemies are relatively slow, have poor detection, and go down in 3 shots[[1]](#footnote-1). These enemies will have 25 health points and do base damage of 10 to the player. These enemies will be encountered in the first two hallways of the game.

### Implementation of Subtask 5B

Once the player solves the second puzzle, they earn the attention of the game’s antagonist. He is afraid that if the player solves all of the chamber’s puzzles, the company’s dark secrets will be exposed, and the company will go under as a result. Therefore, he will start sending security officials into the facility to stop the player.

These enemies are shown in the figure below. They will have 75 health and will deal 20 base damage to the player. They also move faster than the braindead enemies and can easily overwhelm the player if they are not careful.



Figure - T-Posing Model of the Security Enemy

### Implementation of Subtask 5C

Before the player can enter the final puzzle, the antagonist is rendered on red alert. He will send his strongest security official to deal with the player: the Senturion. This hulking bodyguard will have 255 health and deal 40 base damage with each attack. The Senturion serves as the miniboss of the game. Once he reaches half of his current health, he will split into two enemies that have the same health as before the split.

Of course, the boss does not have just one Senturion at his disposal. He has two more at his disposal should the player fail the Final Exam at the end of the game. These enemies should not be taken lightly in any way, shape, or form.

## Implementation of Task 6

The penultimate task of the project involves creating the final room the player will explore in the game: the main lobby. Here, the player will explore a normal facility that completely contrasts what the player was navigating up until now. There will be NPCs to interact with as they go about their daily lives in the facility and the last encounter with the antagonist himself and an ending event that will send players over to Tyler’s virtual reality game-based learning application.

### Implementation of Subtask 6A

The following two figures showcases the main lobby. The first figure shows two NPCs standing about. They are slacking off in the corridor behind the lobby, but they mean no harm to the player. The second figure shows one NPC tinkering with the facility’s equipment. He is going to ignore the player and focus on his job at hand.

A picture containing water, man, riding, large

Description automatically generated

Figure - Two Workers Slacking

A picture containing indoor, computer, small, sitting

Description automatically generated

Figure - Overhead View of the Working NPC

The next figure showcases what is outside the front lobby of the facility. Here is a plane that is meant for employees when they are sent to other facilities around the world. Unfortunately for the player, this plane is ready to take them to an alien space craft somewhere in space. Regardless of the outcome of the Final Exam, the player is forced to enter the plane and be sentenced to this space craft. Of course, they can try to escape the space craft; but the premise for that escape is showcased by a similar project done in the past.

A close up of a door

Description automatically generated

Figure - The Plane to Space

### Implementation of Subtask 6B

At the front of the main lobby, the player will encounter the main antagonist of the game. He is not happy with the player escaping the facility and potentially leaking his company’s secrets to the nation. This subtask implements his last stand to the player: The Final Exam. Here, the player is forced to answer 5 questions about digital logic circuits using the information learned from the puzzles as a reference. The fate of the player is determined by how many questions they get right.

Should the player get at least 4 right, they immediately clear the game, but not without being placed on the plane and out into space since the boss does not want to take any chances. On the other hand, he will summon 2 Senturions after the player should they miss more than 1 question on the Final Exam. Of course, defeating the two allows the game to end as normal.

## Implementation of Task 7

1. Conclusion (Discussion and Future Plans)

By the end of the project, conclude the project and your learning experience.

Acknowledgment

If you get help or support from someone else (besides the team member and the advisor) and want to show your appreciation, put here (**do not include the advisor**).

Appendix

1. Source Code.
2. Basic Enemy.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.AI;

public class BasicEnemy : MonoBehaviour

{

private NavMeshAgent \_nav;

private Transform \_player;

private BasicEnemyHealth \_enemyHealth;

// Start is called before the first frame update

void Start()

{

\_nav = GetComponent<NavMeshAgent>();

\_player = GameObject.FindGameObjectWithTag("Player").transform; //Searches for Player Location.

\_enemyHealth = GetComponent<BasicEnemyHealth>();

}

// Update is called once per frame

void Update()

{

\_nav.SetDestination(\_player.position); //Gives Chase to Player

/\* if (\_enemyHealth.HP > 0)

{

\_nav.SetDestination(\_player.position); //Gives Chase to Player

}

else

{

\_nav.enabled = false; // Can't have dead enemies aggroing, can we?

} \*/

}

}

1. Basic Enemy Health.cs

using System;

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class BasicEnemyHealth : MonoBehaviour

{

public float HP = 25; //Basic Mooks will take 3 hits

//At Base Damage 10

private Animator \_animator;

void Start()

{

\_animator = GetComponent<Animator>();

}

public void TakeDamage (float damage)

{

if (HP <= 0)

{

return;

}

HP -= damage; // Count the Health down.

if (HP <= 0) // Enemy Defeated

{

Death();

}

}

private void Death()

{

\_animator.SetTrigger("Death");

}

}

1. PlayerHealth.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class PlayerHealth : MonoBehaviour

{

public Slider HealthBar; // Player Health Display

public float maxHealth = 100;

public float damageMod = 1; // 0.75 after 1st puzzle. 0.5 after 3rd puzzle.

private float \_CurrentHealth;

void Start()

{

\_CurrentHealth = maxHealth; //Set HP value

}

// Update is called once per frame

public void Damaged(float damage) // When Hit

{

\_CurrentHealth -= damage \* damageMod; // Enemy Damage \* Damage Mod

HealthBar.value = \_CurrentHealth; // = New Current Health

}

}

1. Boss Enemy.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.AI;

public class BossEnemy : MonoBehaviour

{

private NavMeshAgent \_nav;

private Transform \_player;

private BossHealth \_enemyHealth;

// Start is called before the first frame update

void Start()

{

\_nav = GetComponent<NavMeshAgent>();

\_player = GameObject.FindGameObjectWithTag("Player").transform; //Searches for Player Location.

\_enemyHealth = GetComponent<BossHealth>();

}

// Update is called once per frame

void Update()

{

if (\_enemyHealth.HP > 0)

{

\_nav.SetDestination(\_player.position); //Gives Chase to Player

}

else

{

\_nav.enabled = false; // Can't have dead enemies aggroing, can we?

}

}

}

1. Boss Health.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class BossHealth : MonoBehaviour

{

public float HP = 255; //Boss must be chufty :P

private Animator \_animator;

void Start()

{

\_animator = GetComponent<Animator>();

}

public void TakeDamage(float damage)

{

if (HP != 0)

{

return;

}

HP -= damage; // Count the Health down.

if (HP <= 0) // Enemy Defeated

{

Death();

}

}

private void Death()

{

\_animator.SetTrigger("Death");

}

}

1. Aggroed.cs

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Aggroed : MonoBehaviour

{

Animator \_animator;

GameObject \_player;

private bool \_collidedWithPlayer;

private void Awake()

{

\_player = GameObject.FindGameObjectWithTag("Player"); // Search Player

\_animator = GetComponent<Animator>();

}

private void OnTriggerEnter(Collider other)

{

if (other.gameObject == \_player) // Mook meets player

{

\_animator.SetBool("IsNearPlayer", true);

}

}

private void OnTriggerExit(Collider other)

{

if (other.gameObject == \_player) // Player escapes mook

{

\_animator.SetBool("IsNearPlayer", false);

}

}

void Attack() // In Combat

{

if (\_collidedWithPlayer) // Mook fights player

{

\_player.GetComponent<PlayerHealth>().Damaged(10); // 10 for basic mook. 20 for elite mooks. 40 for the boss.

}

}

}

REFERENCES

1. 3 Shots: This is from base player weapon stat where their damage for each shot is 10. It will increase to 30 by endgame. [↑](#footnote-ref-1)