ECE 385: Digital Systems Laboratory

Fall 2022

Final Project

Final Project Proposal
Tower of the Sorcerer

Jialiang Xu (jx17)
Jinrui Hu (jinruih2)
TA: Neo Yuan
November 7, 2022

1 Idea and Overview

We propose to implement the classic game, *Tower of the Sorcerer*, on the DE-10 lite FPGA board. A screenshot of the gameplay is shown in Figure 1. Despite lacking the source code, information about the v1.2 game can be found online[1,2].

When finished, the full hardware implementation should include a VGA controller that shows content on a compatible monitor, a NIOS II CPU to interface a USB keyboard, an on-chip memory to store the texture of the game and character attributes (*e.g.*, attack strength, defence strength, a health bar, and gold) and other peripherals such as the System Bus.

Note that the original game contains 50 tower floors, 35+ monsters, and 32+ items, which, given the constraints of our working time and FPGA resources, essentially renders the remaking of the whole game unreasonable. Our plan is to implement a simplfied version of the game, with the first few floors of the tower and first fraction of the game storyline instead of complete features of the game.

Figure 1: A Screenshot of the Gameplay.

2 Block Diagram

In Figure 2, the block diagram of the project is shown. There are a few main components in this block diagram:

[1] https://lutris.net/games/tower-of-the-sorcerer/
[2] https://tig.fandom.com/wiki/Tower_of_the_Sorcerer

- The **User** controls the main character and the game UI via the USB keyboard.
- The **USB keyboard** interfaces with the NIOS II CPU.
- The **NIOS II CPU** interacts with the **On-chip memory** where the texture files are stored via the **Avalon Bus**. We will use tiles to store the visual design of the game elements.
- The **VGA controller** is responsible for displaying the current game state to the user so the user can decide next action to take.
- The **Game Logic Software** in the middle controls the gameplay and the interaction logic.

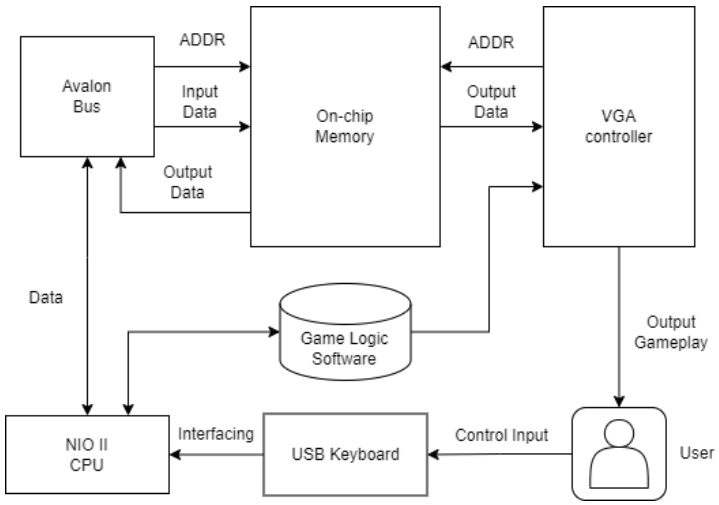

Figure 2: The Block Diagram of the Implementation

3 List of Features

Given the complexity of the game logic, we aim to implement a subset of the following expected features:

- Drawing the map with colored walls, floors, stairs, monsters, items, and the main character.
- The main character should interact with the environment correctly. *E.g.*, no glitches when walking, stoping at the wall, entering next floor when on the stairs blocks.

- The main character should be able to pick up and use items (*e.g.*, keys, lotions), the effect should be reflected on the display immediately (*e.g.*, keys opening doors, lostions increasing health bar)
- The main character should be able to combat with the monsters and the results should be intuitive (*e.g.*, the health bar descreasing, monsters vanishing with an increase in gold)
- The main character should be able to interact with the NPC (Non-player characters) by dialoging and trading items.
- There should be some mandatory plots where the user is watching a fixed sequence of actions.

4 Expected Difficulty

The Difficulty mainly comes from the following aspects:

- **Game Logic Implementation**. Being an RPG game, *Tower of the Sorcerer* has multiple systems (*e.g.*, the combat system, the character attribute system, the environment interaction system). We need to figure out the best way to make these systems interacting with each other properly.
- **Visual Design and Manipulation**. The game has rich visual elements. Statically, each of the numerous game elements (*e.g.*, the main character, the monsters, the items, and environments such as floors, walls, doors) has multiple colors. We need to find a way to correctly store and fetch the needed color, without violating the memory constraints. Dynamically, when the player is interacting with the environment (*e.g.*, switching back and forth the tower floors), the data for different floors should be displayed on the screen properly. The map state also needs to be tracked, (*e.g.*, when the player goes from the second floor to the first floor, only the items left in the first floor should be displayed, not a completely new first floor).
- **Synchronization and Hardware Design**. The state of the character and the state of the game (*e.g.* environment condition, item usage, monster attributes, character attributes) should be changed in a synchronous manner, *i.e.* nothing should be lagging others. And this can be a potential issue when hardware delays and wait cycles are involved.

5 Proposed Timeline

For the four-week final project period, we plan to utilize the first week collecting game elements and building the minimal hardware implemnetation. During the second week, we aim to build a Minimal Viable Product with basic working functionalities of the game. The third and fourth weeks will be spent on implementing more features and debugging.