ECE 385

Spring 2024

Final Project

Final Project Proposal

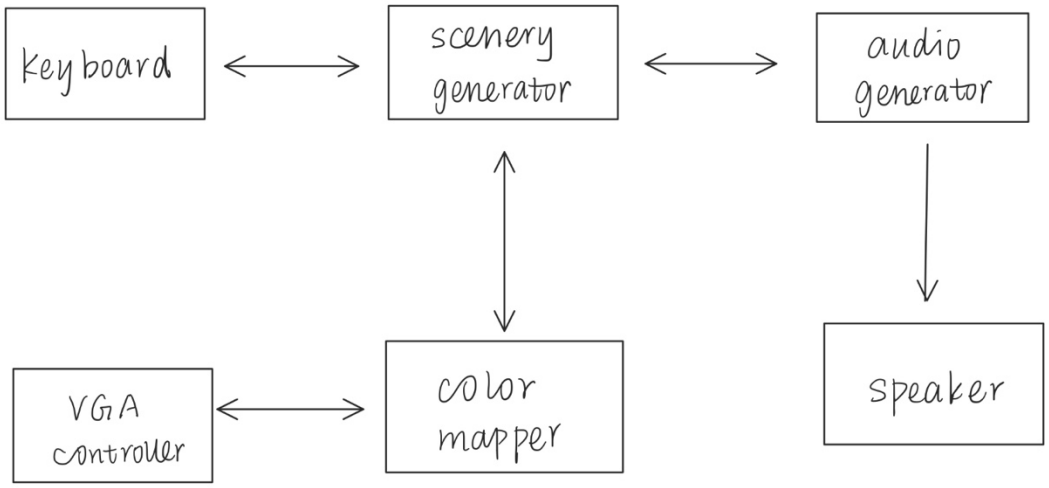
By Wendi Wang (wendiw2)

1. **Idea and Overview**

We propose to design and implement a modified version of *Temple Run* on the Urbana Board as a System-on-chip. Using SystemVerilog, we plan to implement a scenery generator, color mapper, and VGA controller. Besides, we also plan to implement some essential components like the SRAM, SDRAM (storing audio), and Keyboard. Our goal is to demonstrate the SoC using the USB keyboard and VGA monitor to run our copy of *Temple Run.*

Some explanations to "scenery generator" mentioned above: Since *Temple Run* is an "infinite" game, this game never ends. Additionally, the game scenes of *Temple Run* will always be changing and updating before the user loses the game. All the game scenes are generated randomly from several possible choices (cliffs, rivers, fires, etc.). To accommodate to this property, we plan to write a scenery generator to randomly generate the next game scene to draw periodically.

2. **Block Diagram**



3. **List of Features**

i. Baseline set of features for the project to be considered working

The start screen:

The start screen will contain a button with a line of instruction: Press "Enter" to start the game. Additionally, after pressing the "Enter" key, there will be a background music throughout the process of game.

The game screen:

- ✧ The game screen consists of two parts: the background and a "running" person. If the user presses nothing on the USB keyboard, the running person will directly be running upward (in 2D version) or forward (in 3D version).
- ✧ Simultaneously, the initial background will be moving upward or forward, extending to new obstacles such as cliffs, rivers, fires, etc. Each obstacle can

be climbed over by pressing W / A / S / D keys on the USB keyboard. If wrong keys are pressed or the user doesn't press any keys, the game is over with a new screen displaying the user's scores and choices of re-starting the game or exiting.

- ✧ Each time the user presses some keys to get over an obstacle, there will be corresponding background sounds according to the type of obstacles. Apart from getting over the obstacles, along the path of the running person, there will be some randomly distributed coins to collect by pressing some specific keys on the USB keyboard. The number of coins collected will be the scores demonstrated on the ending screen.
- ✧ The speed of the running person will be gradually increasing to a specific value and then keeps that value (relatively fast).

The ending screen:

When the game is over, a new screen will appear to display the user's score (the number of coins collected) and choices of re-starting the game or exiting the game.

Note: All the screens will be colorful using the color mapper module.

ii. Additional features that may be implemented for extra difficulty

A simple version of *Temple Run* is a 2D game, but *Temple Run* itself is a 3D game from a third-party perspective. The 2D game can be extended to 3D version by displaying 3D objects on 2D planes. The process of drawing 3D graphics might require the implementation of a 3D graphics renderer.

4. Expected Difficulty

i. Baseline set of features for the project to be considered working

Difficulty: 7/10

Explanation: This is primarily due to the complexity involved in implementing the game mechanics, such as character movement, obstacle generation, collision detection, score tracking, and user interface elements. Additionally, integrating background music, sound effects, and colorful visuals using a color mapper module adds another layer of complexity. Furthermore, changing the speed of the running person may also be challenging since it might involve adjusting clock signals.

ii. Additional features that may be implemented for extra difficulty

Difficulty: 2/10

Explanation: Implementing a 3D graphics renderer can be challenging, especially in terms of managing perspective transformations and rendering techniques.

iii. Total Estimated Difficulty: 9/10

5. Proposed Timeline

Week 1 (Apr. 8th – Apr. 12th)	Modify Lab 6.2 and Lab 7 to construct a start screen and finish the functionality of pressing “Enter” on the USB keyboard to enter the game screen. Construct the game screen and scenery generator to build a “never-end” game.
Week 2 (Apr. 13th – Apr. 19th)	Complete the 2D version of the game except for the background sounds and music. Users can control the running person to get over obstacles by pressing correct keys on the USB keyboard.
Week 3 (Apr. 20th – Apr. 26th)	Add sounds and music to the 2D version game. Change the running person’s speed gradually to a specific value.
Week 4 (Apr. 27th – May 3rd)	(Complete additional features for extra difficulty) Design a 3D graphics renderer to extend the 2D version to a 3D version *Temple Run.*