CSEE 4840 Spring 2023

Project Design Document

Rhythm Game

Lu Zhang (lz2879), Qiao Zhang (qz2486), Jinke Zhao (jz3581),

Xinyuan Fu (xf2247), Zerui Li (zl3194)

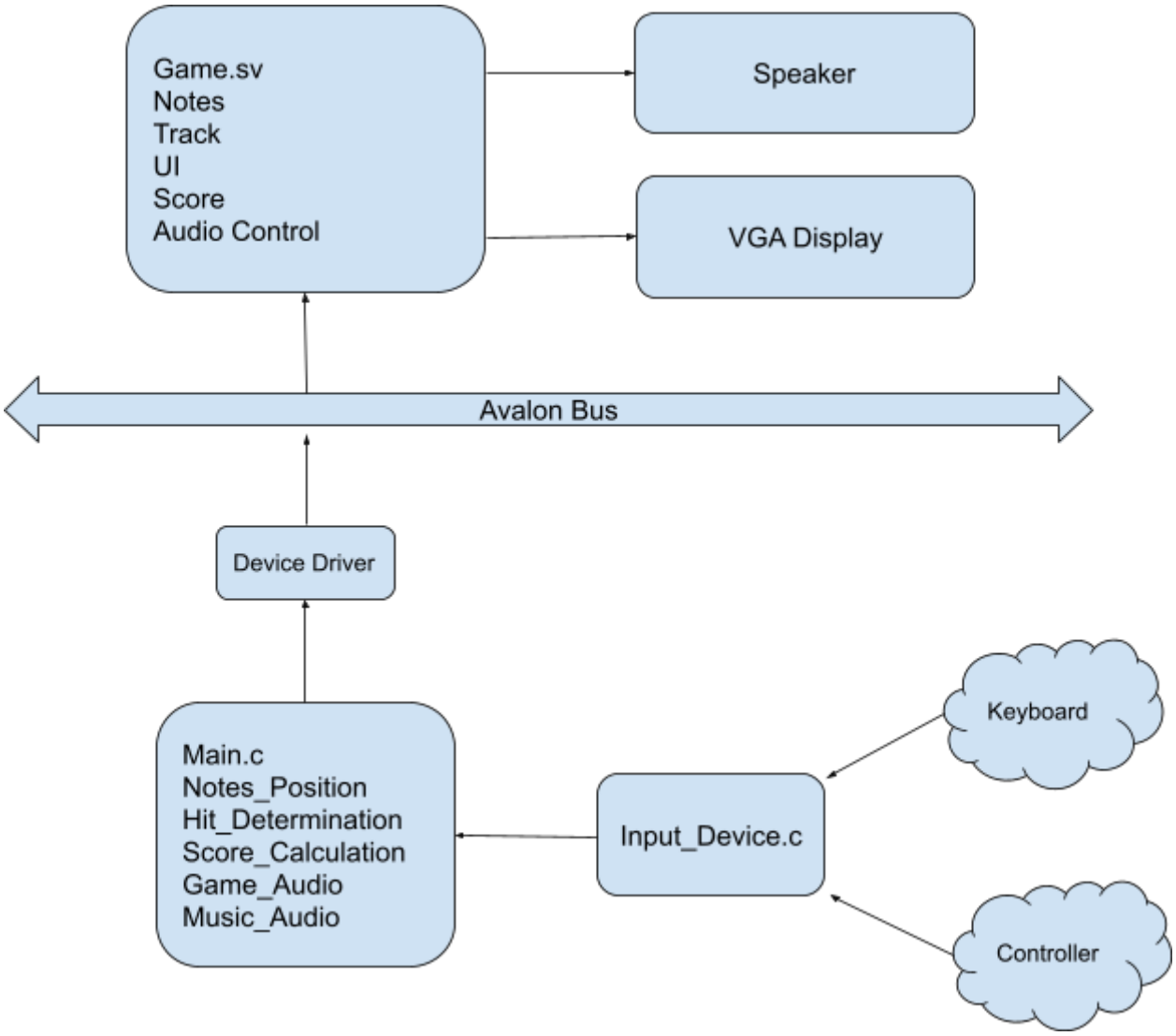
1. Introduction

Our design project is a 2D rhythm arcade game that requires a VGA display and speaker to provide an immersive gaming experience. Players will use a keyboard or controller to interact with the game, following the rhythm of the music to hit notes or turn the nod as they reach the baseline. The more players hit the notes or turn the nod accurately, the higher their score will be.

Our hardware will implement display, audio, and user control input. The main device is the DE1-SoC Board. We will receive input from the user and send it to the software part to analyze; and receive display or audio output signal from software, like note coordinates, sound rhythm.

Our software can manage everything for the game, it can acquire the input from the user, control the game logic, calculate the game score, and send the information to the hardware. In the beginning, it will translate the input device from the USB port, like what we do in lab 2. There are several functions to perform game logic, such as determining if a note is missed or hit, calculating the score based on how many notes are hit, notes position on the screen, play the song and the hit audio.

2. System Block Diagram



3. Hardware Implementation

There are three essential input or output devices, including a display, speaker, and keyboard.

Display: The display is used to show the main GUI at first. Once the game start, the display will show notes dropping from the top of the display. If the user hit a note correctly according to the software logic, then an indicator will show up and let the user know if that input is a hit. In the opposite way, When the user misses a note, a miss indicator will also pop up.

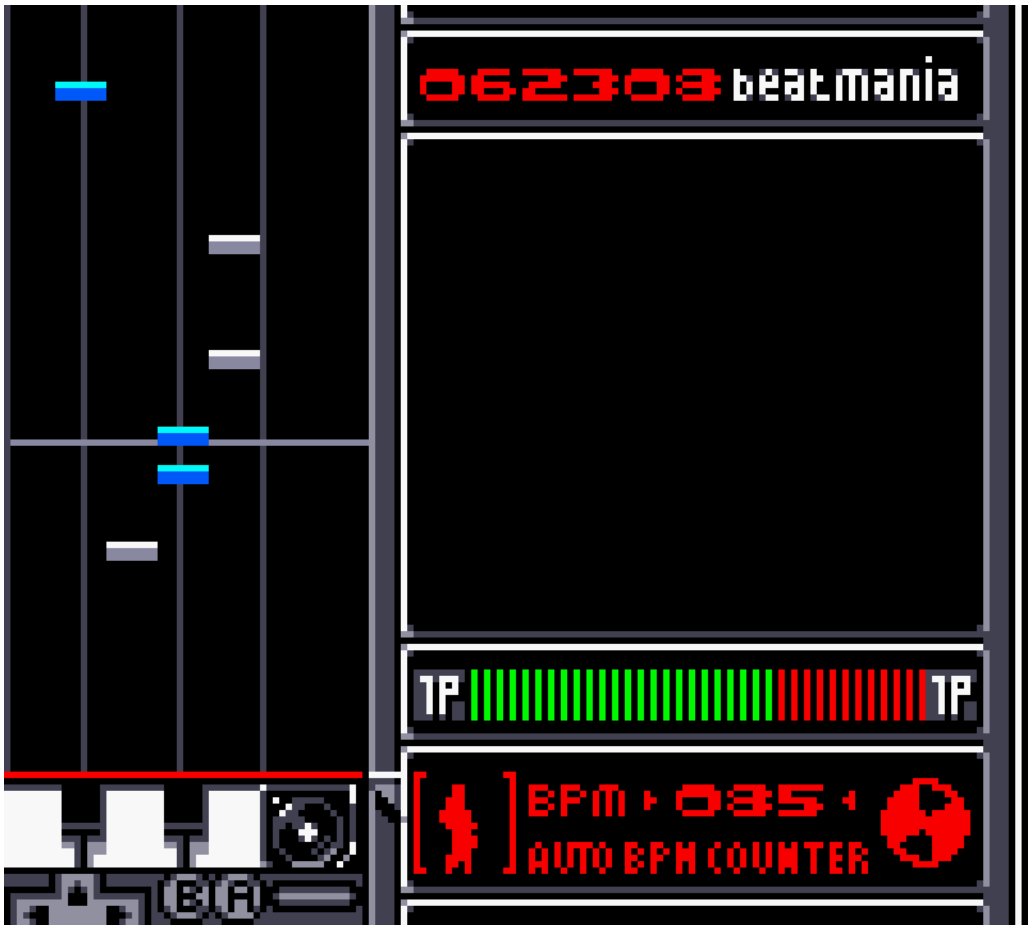


Figure 2. Sample Screenshot of the game interface

Speaker: The speaker is mainly used to play the music, the rhythm of music will help the user to hit the note. The speaker will also play some specific sound to indicate user input. Application of playing music on DE1-SOC mainly relies on the HPS Linux deployment and onboard audio chip WM8231 to decode the music file read in the SD card. The detailed block diagram of this speaker driver system shows in figure 3.

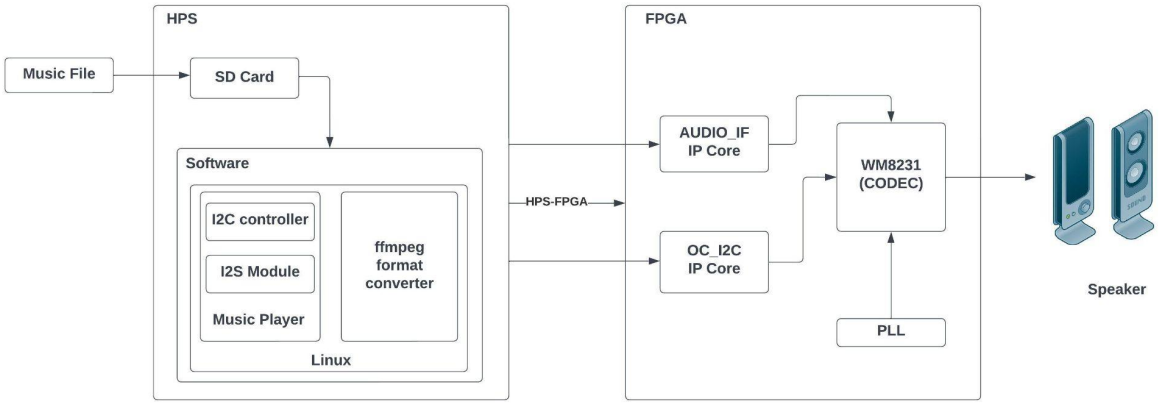


Figure 3. Speaker Driving Block Diagram

Input Device(Controller/Keyboard): The input device Fig4. will supply an operating platform for user's input, it mainly consists of a couple of buttons, including the "hit" for each track and other operations like "confirm", and "end". Keyboard will be the alternative input device, in case of the custom controller does not have a proper interface for FPGA.

Figure 4. Custom Controller

4. Software Implementation

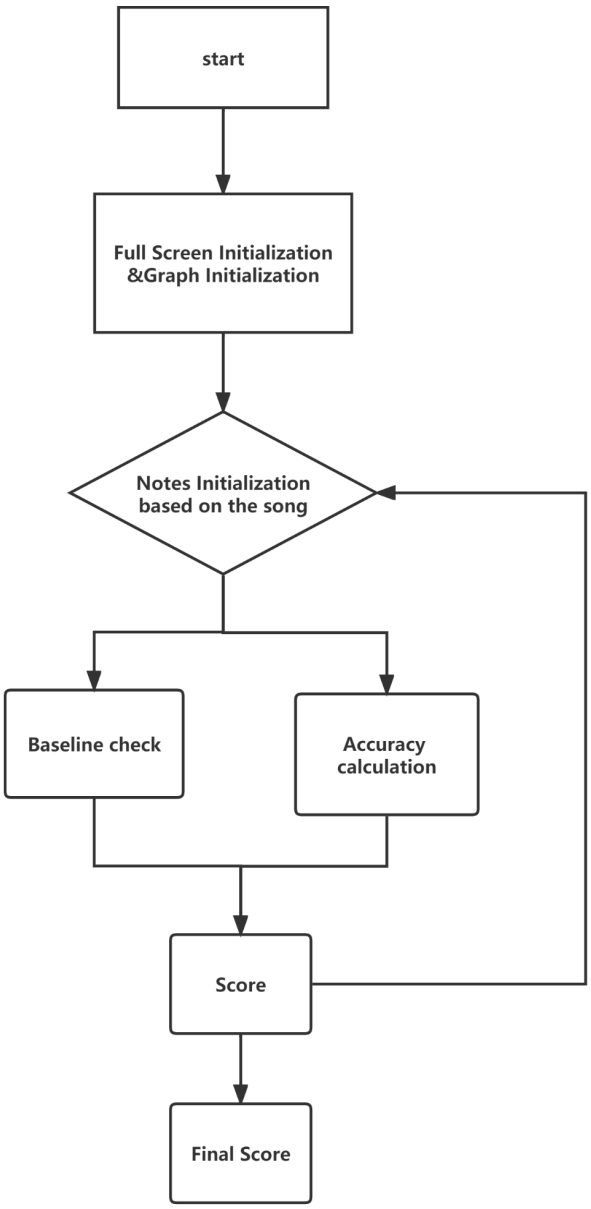


Figure 5. Software Algorithm

Step 1: Initiate the song and the display

In this step, initiate the graph and sound, including UI and main menu.

Step 2: Preload necessary information for the notes.

Based on the user's choice in the main menu, preload all information about the note placement, such as when a note should be displayed on which track and the range of hit timestamp for each note.

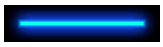
Step 3: Generate notes

In this step, notes will be generated based on the preload information.

Step 4: Input detection

In this step, detect user input, and compare the timestamp with the preload notes timestamp. Determine hit or miss based on the comparison result. If a hit occurs, calculate the score according to the accuracy.

5. Graphic Memory Budget

Classification	**Graphics**	**Pixels**	**Size(bits)**
note_blue		78*18	4072
note_red		53*19	4592
background		191*323	60712
hit_mark		64*88	29952

hit_indicater		77*35	12800
miss_indicater		47*35	9176

6. Milestones Plan

a. Construct the functional game machine and config its driver.

b. Design the driver for the speaker and keyboard.

c. Testing the hardware driver on FPGA.

d. Once the hardware is successfully set up, implement the basic game logic software.

e. Test and refine. Improving the user experience, perfecting UI, and maybe add more features to enhance playability (if time permit)