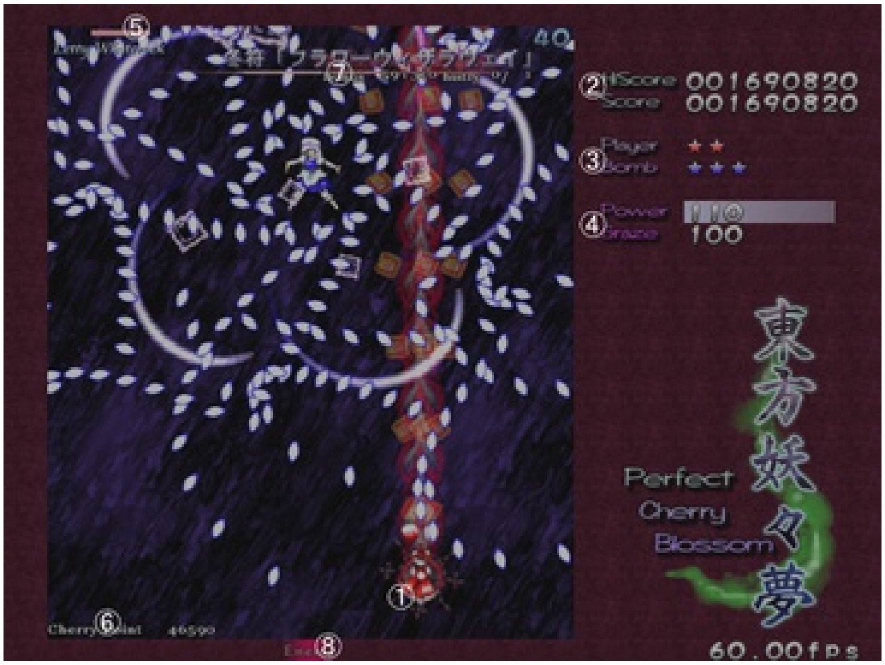
2D Bullet Dodging Game

1 Members

Po-Cheng Liu (pl2812), Xinye Jiang (xj2253)

2 Introduction

Touhou is a very popular and classical Japanese series of 2D shoot'em up games made independently by Zun alone for the most part. Essentially, the player controls the main character who shoots bullets while dodging the enemies' bullets, also known as vertically-scrolling danmaku. It is well-known for its delicate design of danmaku, tough challenges even at the lowest difficulty level, music also written by Zun himself, and the character design. In this project, we want to create a similar 2D bullet dodging game, empowered by our FPGA board, LCD screen, a speaker, and a USB keyboard.



A screenshot from a boss fight in one *Touhou* game (wikipedia.org)

The code that generates the bullets and other game mechanics would be written in Verilog and C, used on our FPGA board, and the LCD screen would display the virtual effect, with simple keyboard control to move around the playable character.

In *Touhou*, the bullets shot by the playable character are usually a straight line or locked on the enemies with very little damage compared to the bullets the enemies shoot out. In this way, dodging the attack becomes the priority of the gameplay instead of shooting the enemy, making Touhou very different from other 2D shooting games. We would like to focus on the dodging side of this kind of game too, and explore this possibility of automatic tracing bullets.

On the other hand, in *Touhou*, bullets generated by the enemies, whether they are street-enemies or bosses, are not random based. Different patterns of danmaku are used for each enemy. We would like to do the same and instead of all 6 to 7 chapters in each touhou game, we would make it shorter and more compact for our hardware to run.

All in all, software controls the character's generation, movement, action, and score calculation, and hardware handles display and all bullet movement by its momentum, and also reports to software if a bullet hits an enemy or player. In addition, we want to use some soundtracks and sound effects accompanying the gameplay, so we would also need to drive a speaker.

3 Peripherals

Video, audio, USB keyboard.

4 Hardware Needed

We already have an FPGA board, a LCD screen, and a USB keyboard provided in the labs, so we only need **a speaker** for the audio effect.