Street Fighter Game Design Proposal

Jiyang Yin (jy3557) Yang Li (yl6070) Zhewen Guo (zg2567)

March 2026

1 Introduction

This project proposes the implementation of a **2D fighting game on an FPGA-based embedded system**, inspired by classic arcade games such as *Street Fighter*. The game will support two players controlling characters that can move, jump, crouch, and perform attack actions while competing to reduce the opponent's health.

The objective of this project is to demonstrate how an FPGA platform can support **real-time interactive gaming**, combining graphics rendering, input processing, collision detection, and state-based gameplay logic. Implementing the game directly on FPGA hardware allows the system to leverage parallel computation and deterministic timing to achieve responsive control and real-time graphics output.

The system will output video through **VGA**, while player inputs will be received from hardware interfaces such as buttons, keyboard, or joystick controllers. Core gameplay elements such as character movement, attack interaction, and health management will be implemented using hardware modules and finite state machines.

This project aims to demonstrate **hardware/software co-design, real-time system integration, and multimedia processing** within the constraints of FPGA resources.

Figure 1: Example gameplay scene inspired by Street Fighter.

2 Game Features

The FPGA-based fighting game will include the following core gameplay features:

Character Movement

Players can control characters to move left or right, jump, and crouch. Character positions will update continuously according to player inputs.

Attack Actions

Each character can perform basic attacks such as punch or kick. Attack actions will activate temporary hitboxes used for collision detection.

Collision Detection

Bounding box collision detection will determine whether an attack successfully hits the opponent.

Health Bar System

Each player has a health bar that decreases when hit by the opponent.

Game States

The game will transition between three main states:

- Start Screen
- Playing State
- Game Over State

Graphics Output

Characters, background elements, and UI components such as health bars will be rendered and displayed on a VGA monitor.

3 Game Logic

The game will operate using a **frame-based update mechanism** synchronized with the display refresh rate.

At each frame update, the system will:

1. Read player input
2. Update each character's state using a finite state machine
3. Update character positions and actions
4. Detect collisions between attack hitboxes and opponent hurtboxes
5. Update player health values
6. Render the next frame to the display

State transitions depend on current input signals and character conditions.

4 Hardware–Software Interface

1. Game Logic on ARM
2. FPGA for VGA
3. USB Controller Input Handling
4. Audio Output
5. Memory and Data Storage

5 Challenges and Considerations

Several technical challenges are expected during implementation.

Real-Time Graphics Rendering

Rendering moving sprites and UI elements while maintaining stable frame timing requires efficient graphics pipelines.

Input Responsiveness

Player controls must be processed with minimal latency to ensure responsive gameplay.

System Integration

Integrating input processing, character logic, rendering, and optional audio into a real-time system requires careful synchronization.