# ECE 385

Spring 2024

Final Project

# FPGA-based Tetris Game

# Name:Jie Wang, Shitian Yang

Student ID: 3200112404, 3200112415

Prof. Chushan Li,

Prof. Zuofu Cheng

ZJU-UIUC Institute

May 24th , 2024, Friday D-225

TA: Jiebang Xia

**Demo Point: TODO/100**

## Introduction

### 1.1 Project Overview

The objective of this project is to develop a real-time Tetris game on an FPGA using SystemVerilog and C. The game features a VGA display for graphical output and a PS/2 keyboard for user input. The project involves implementing the game logic, rendering graphics, handling user inputs, and managing game states.

### 1.2 Motivation and Goals

This project was chosen for its educational value in combining hardware and software design skills. The goals include gaining practical experience in digital system design, enhancing proficiency in SystemVerilog and C, and understanding the integration of hardware and software components in embedded systems.

## System Design

### 2.1 Block Diagram

The block diagram of the system includes the following modules:

* Keyboard Handler
* Game Logic Controller
* Audio Handler
* VGA Display Handler
* Memory Module
* Timer Module
* Score and Level System
* Bonus Functionality

### 2.2 Description of Modules

* **Keyboard Handler**: Captures user inputs from the PS/2 keyboard to control the game.
* **Game Logic Controller**: Manages the game state, including piece generation, movement, rotation, and collision detection.
* **Audio Handler**: Manages background music and sound effects.
* **VGA Display Handler**: Renders the game graphics on the VGA display.
* **Memory Module**: Stores the game state, including the current position of pieces and cleared lines.
* **Timer Module**: Controls the game speed and progression by generating timing signals.
* **Score and Level System**: Tracks the player’s score and adjusts the game difficulty level.
* **Bonus Functionality**: Adds additional features such as multiplayer mode and enhanced graphics.

### 2.3 Hardware and Software Components

The hardware component is the FPGA, which runs the SystemVerilog modules for game logic and display rendering. The software component is a C program running on an embedded processor within the FPGA, handling game states, score tracking, and user input processing.

## 3. Features and Functionality

### 3.1 Baseline Features

* **Piece Generation, Movement, and Rotation**: Random generation of tetrominoes, with movement and rotation controlled by the player.
* **Line Clearing and Game Progression**: Automatic clearing of filled lines and progression to subsequent levels.
* **Basic Scoring System**: Tracks and displays the player’s score.
* **Simple Color Graphics**: Different colors for different tetrominoes.
* **Audio Background Music (BGM)**: Plays background music during the game.

### 3.2 Additional Features

* **Increasing Levels of Difficulty**: Game speed increases with each level.
* **High Score Table**: Stores and displays high scores using non-volatile memory.
* **Enhanced Graphical Effects**: Includes animations for line clears and other events.
* **Game Loading from a CD**: Inspired by console game loading mechanisms.
* **Multiplayer Mode**: Allows multiple players to play simultaneously.

## 4. Implementation

### 4.1 SystemVerilog Implementation

The game logic, including piece movement, rotation, and collision detection, is implemented in SystemVerilog. The VGA display handler module generates VGA signals based on the game state and renders the game graphics.

### 4.2 C Code Implementation

The C program handles the overall game state, user input processing, and score tracking. It communicates with the SystemVerilog modules via memory-mapped I/O.

### 4.3 Integration

The hardware and software components are integrated by passing game state information from the C program to the SystemVerilog modules. This integration ensures real-time updates and rendering of the game graphics.

## 5. Testing and Debugging

### 5.1 Testing Strategy

The system was tested using unit tests for individual modules, integration tests for combined modules, and system tests for the entire game. Both simulation and on-hardware testing were performed.

### 5.2 Debugging Process

Debugging tools included simulation software for SystemVerilog and printf debugging for C. The process involved identifying and fixing issues related to timing, synchronization, and data transfer between modules.

### 5.3 Test Results

The test results confirmed the correct functionality of the game logic, input handling, and VGA display. Issues such as timing mismatches and synchronization errors were resolved during testing.

## 6. Challenges and Solutions

### 6.1 Technical Challenges

* **Timing and Synchronization**: Ensuring proper timing and synchronization between hardware and software components.
* **Data Transfer**: Efficiently transferring large amounts of data between the C program and SystemVerilog modules.

### 6.2 Project Management Challenges

* **Time Constraints**: Managing time effectively to meet project deadlines.
* **Workload Management**: Balancing the workload between team members and ensuring consistent progress.

## 7. Future Work

### 7.1 Planned Enhancements

* **Improved Graphics**: Adding more detailed and visually appealing graphics.
* **Network Multiplayer**: Implementing a networked multiplayer mode.

### 7.2 Potential Improvements

* **Performance Optimization**: Enhancing the performance of the game by optimizing the hardware and software design.
* **User Interface**: Improving the user interface for a better gaming experience.

## 8. Conclusion

### 8.1 Summary of Achievements

The project successfully implemented a real-time Tetris game on an FPGA, including features such as piece generation, movement, rotation, line clearing, and a scoring system.

### 8.2 Lessons Learned

Key lessons include the importance of timing and synchronization in FPGA design, effective debugging techniques, and the value of teamwork in managing complex projects.

### 8.3 Final Thoughts

The project provided a comprehensive learning experience in digital system design, combining hardware and software development. It has potential for further enhancements and serves as a solid foundation for future projects.

## 9. References

* FPGA documentation and reference guides.
* SystemVerilog and C programming resources.
* Online tutorials and articles on VGA signal generation and game development.

## 10. Appendices

* **Block Diagrams**: Detailed diagrams of the system architecture.
* **Code Snippets**: Key sections of the SystemVerilog and C code.
* **Test Logs**: Detailed logs of the testing process and results.

窗体顶端