数字逻辑设计 2019/2020：Final Project

王跃明老师

Game Design: Finger Dancer

Anna Tang 3180300155

Joshua Malmberg 3180300189

Chen Yi Hui 3180300684

Justin Choi 3180300160

1. Abstract

The following report presents the final project for the course “Digital Logic Design”, taken in the fall and winter semester of 2019-2020. We chose to create a finger dancing game, where the player needs to recreate the pattern generated by the circuit in a given time frame. The game is implemented on the SWORD board, and only requires the four component seven-segment display, switches, and the LED lights below the display. To reflect on what we have learned from the course, we have used a variety of different modules such as registers, counters, and frequency dividers. Proper memory and register access have also been applied here. Modules from previous labs such as the multiplexers, and full adder have been recycled into this project. We have also constructed our own input, display and compare modules from scratch. This report will summarize the game behavior, the design process, simulations and debugging process.

1. Introduction
2. Background

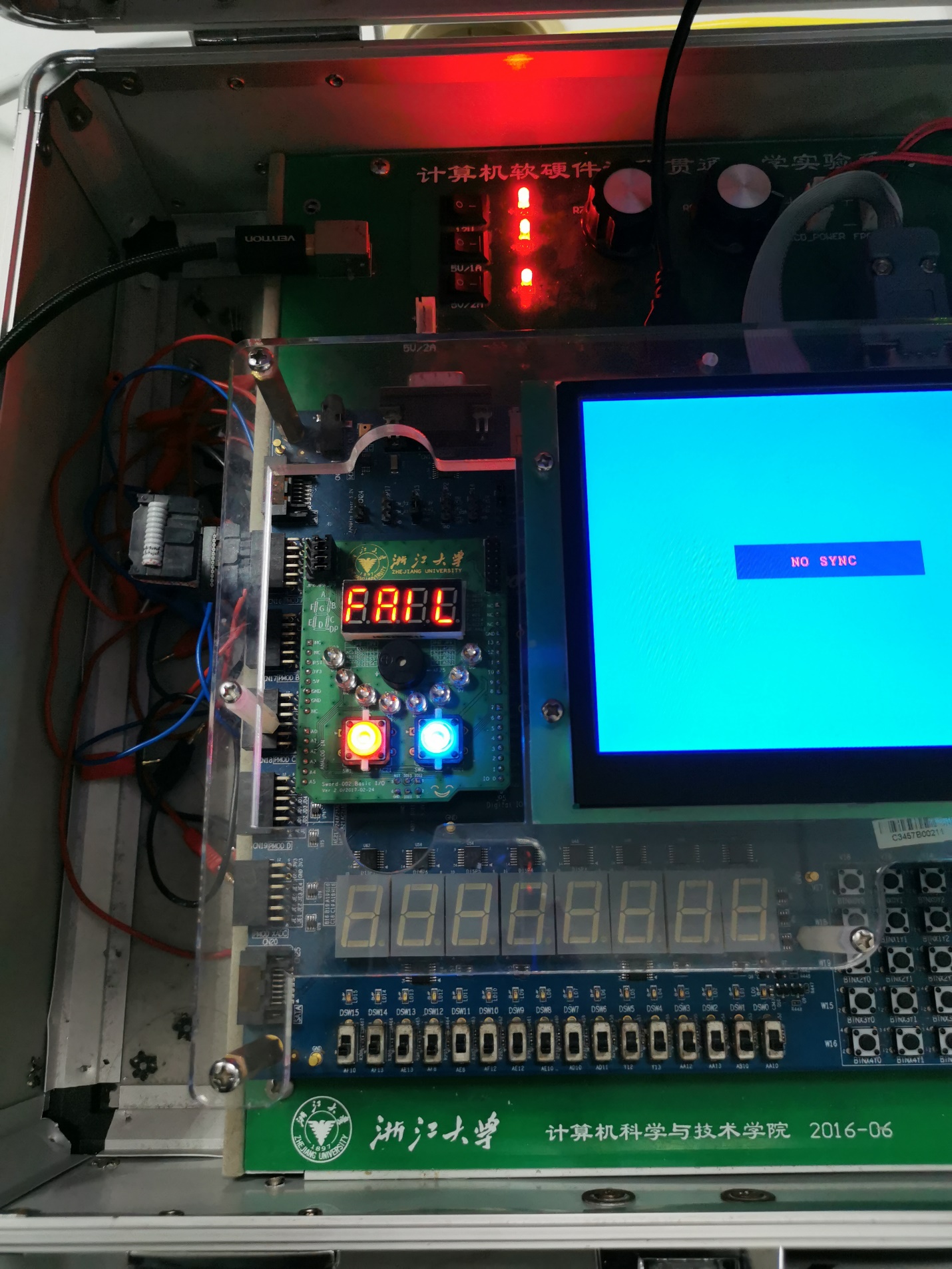
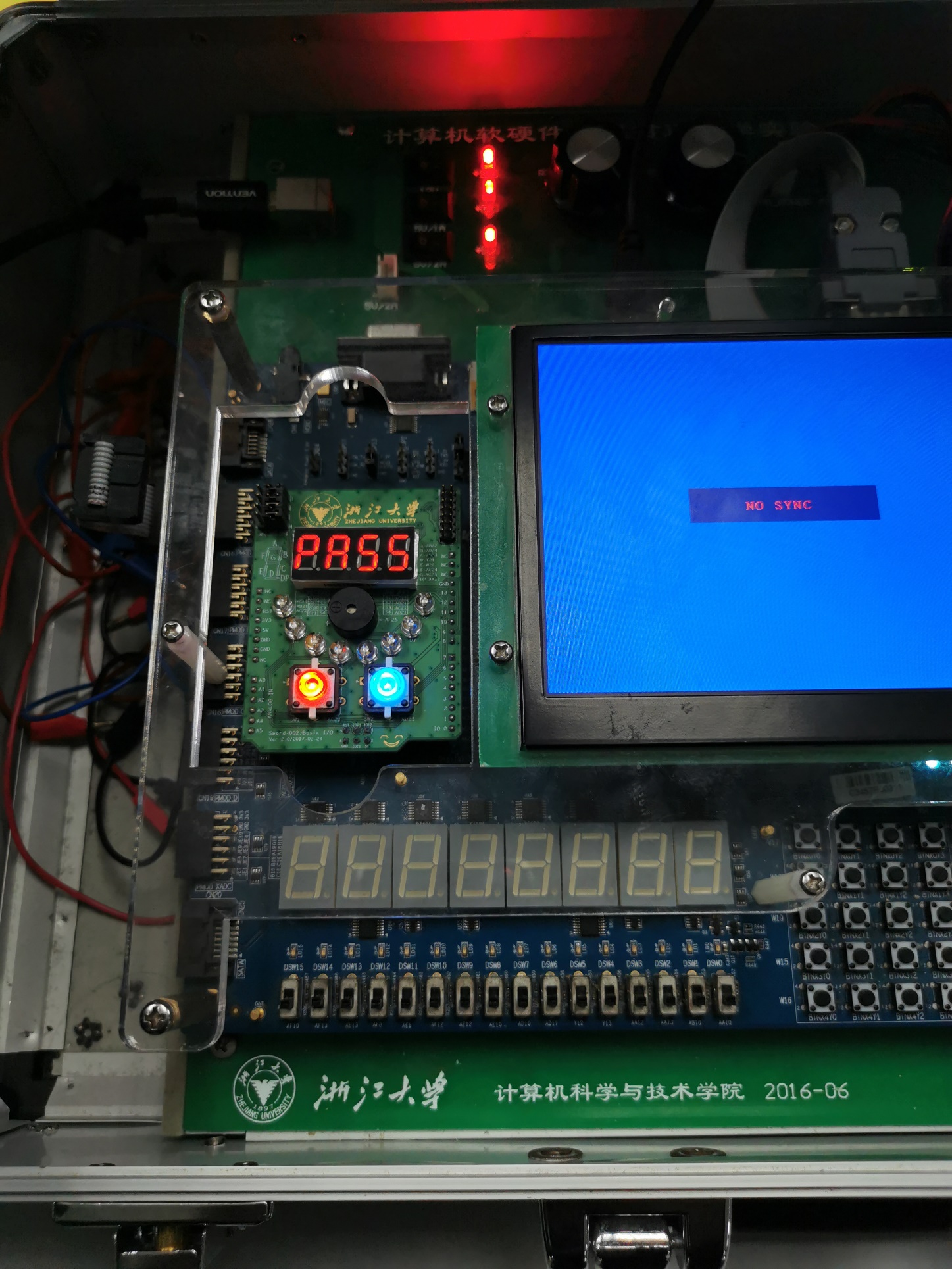
“Finger Dancer” is a simple game where the player is given a pattern of lights and must match it using the corresponding switches before the time runs out. If the player successfully completes the round, “PASS” will be displayed on the seven-segment display and the score increases. In the case that the player does match the pattern, “FAIL” will be displayed and automatically ends the game. As the player progresses, the time for each round will decrease.

1. Purpose

The purpose of this final assignment is to make use of all the various skills and tools learned throughout this course and apply it to construct this game. This allows our team to practice working on the FPGA SWORD board and get a sense of what it is like to design practical circuits.

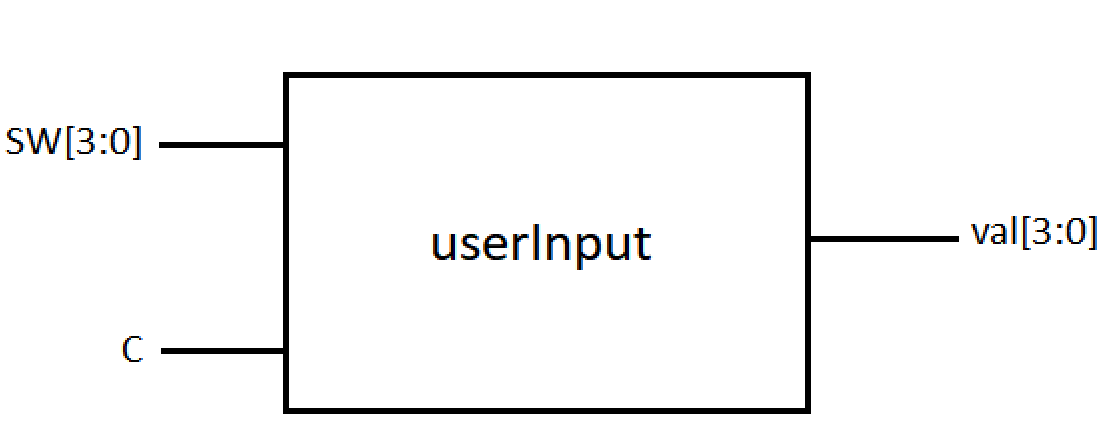
1. Instruments and Materials
2. PC with Xilinx ISE 14.7
3. SWORD/FPGA Board
4. User Manual: How to Play Finger Dancer

The basic idea of “Finger Dancer” is to match the pattern indicated on the LEDs below the seven-segment display, using the switches. Each time the player successfully does so, “PASS” is indicated on the seven-segment display, and their score is increased. The game continues with the duration of each round progressively decreasing. If the player fails to match the switches with the LEDs, the game ends and “FAIL” is displayed. The score is reset, and the player may start the game again.

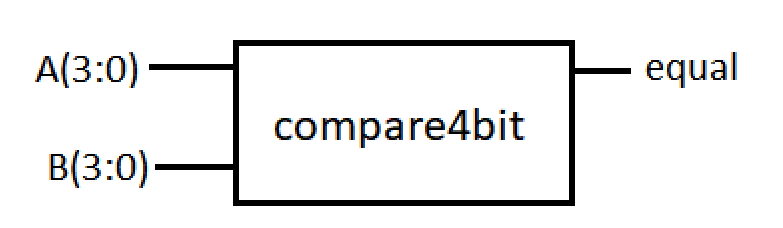
1. Implementation and Design
2. Circuit Design
3. Taking in User Input – Input

The input module takes in the logical on/off values from the switches (SW[3:0]) and returns it to the output val[3:0] for further processing in the compare4bit module. Input C polls for input at every clock cycle.

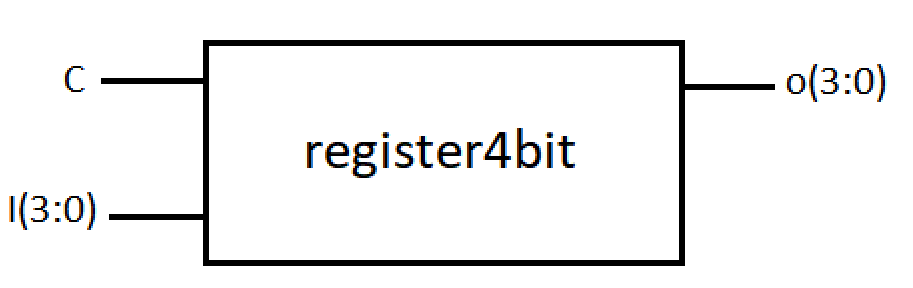


1. Comparing Input – compare4bit

The compare4bit module takes in two 1-bit binary numbers as inputs; one from the input module and one from the state module. To check if these numbers are equal, they are compared by using a complemented XOR gate. If the two inputs are equal, the output “equal” is set to return 1. Otherwise, difference inputs set the output to 0 . This output is used as an enable signal for the other modules.

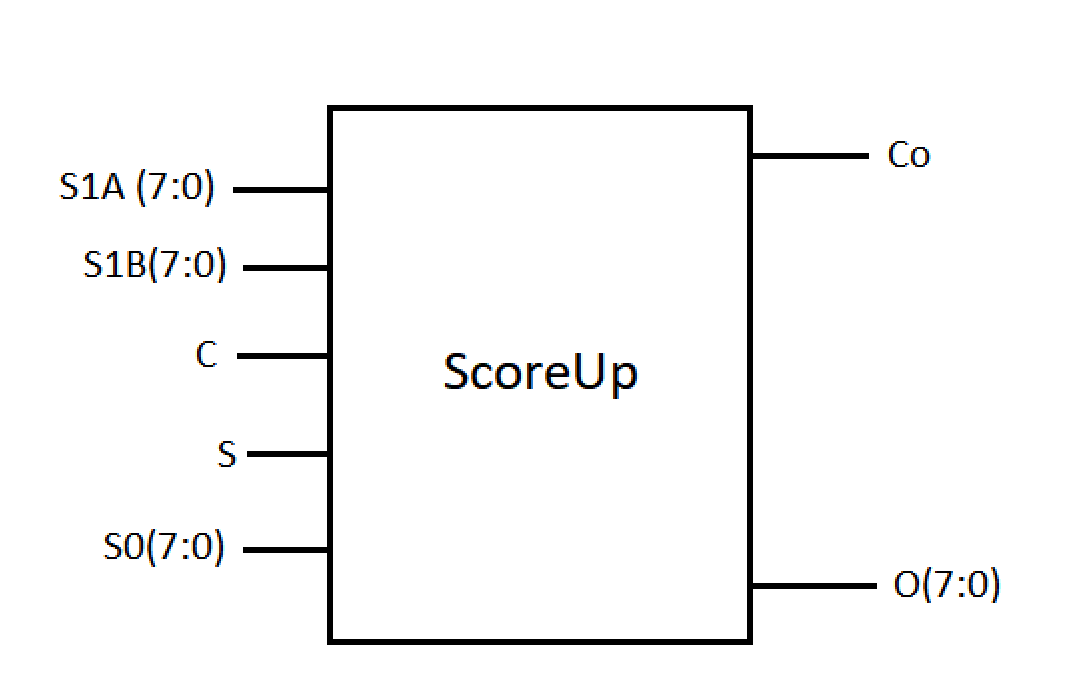


1. Registers -register4bit



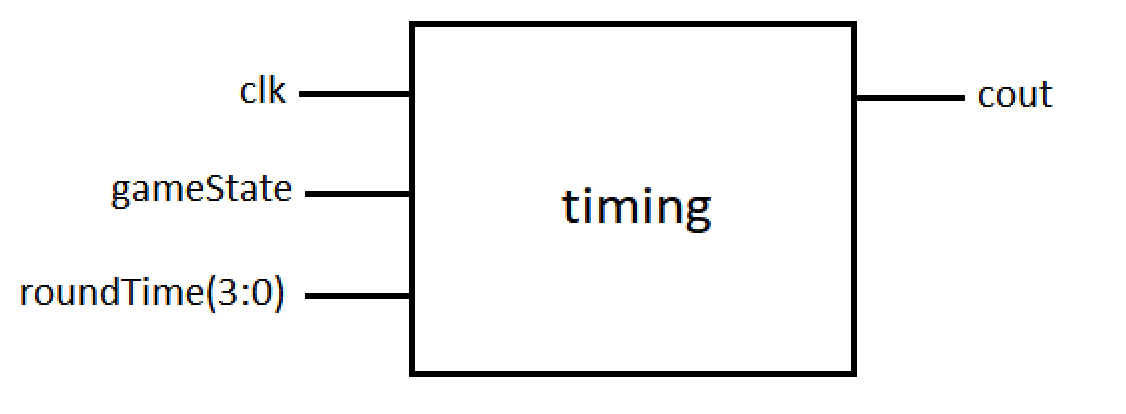
1. Updating the Score – ScoreUp

The ScoreUp module consists of



1. Changing State – UpdateState
2. Changing the Duration – UpdateTime
3. Initializing the Modules – Timing

The timing module consists of a D flip flop, and two frequency dividers. Its purpose is to prepare and initialize the modules in the circuit to update the score, duration, and state of each round in the game. It takes in a clock cycle through input clk, and the two frequency dividers are triggered when clk is at a positive-edge. gameState acts as the enabling input signal for this module, and roundTime(3:0) provides the frequency for processing, which is then returned in output cout.

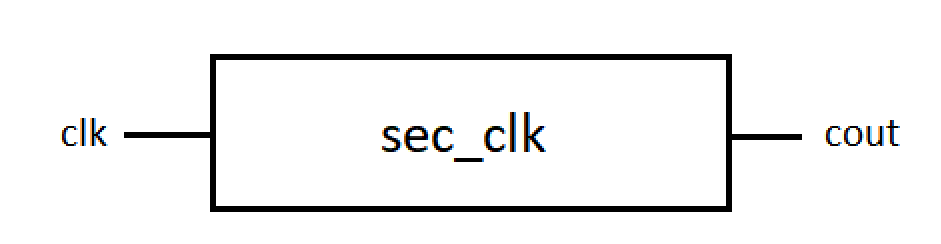


1. Display

The display module consists of the scoreboard module (from a previous lab experiment), a binary-to-BCD decoder and a module that specifically displays “PASS” or “FAIL”.

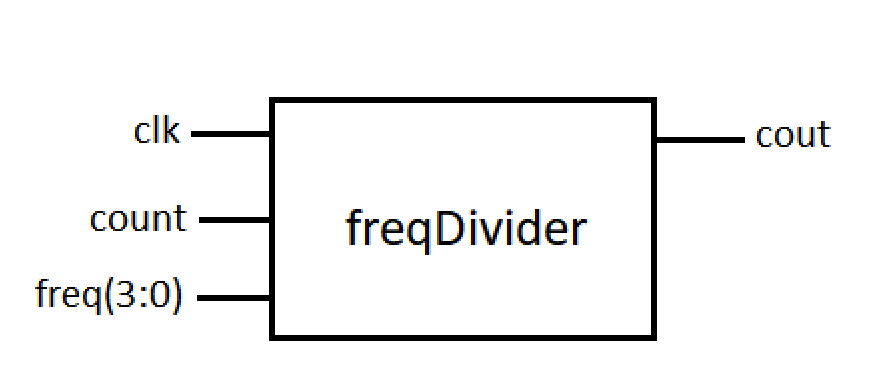
1. One Second Clock – sec\_clock

This module takes in the internal clock pulse of the circuit into input clk, and produces clock pulses of one-second interval for return in cout.

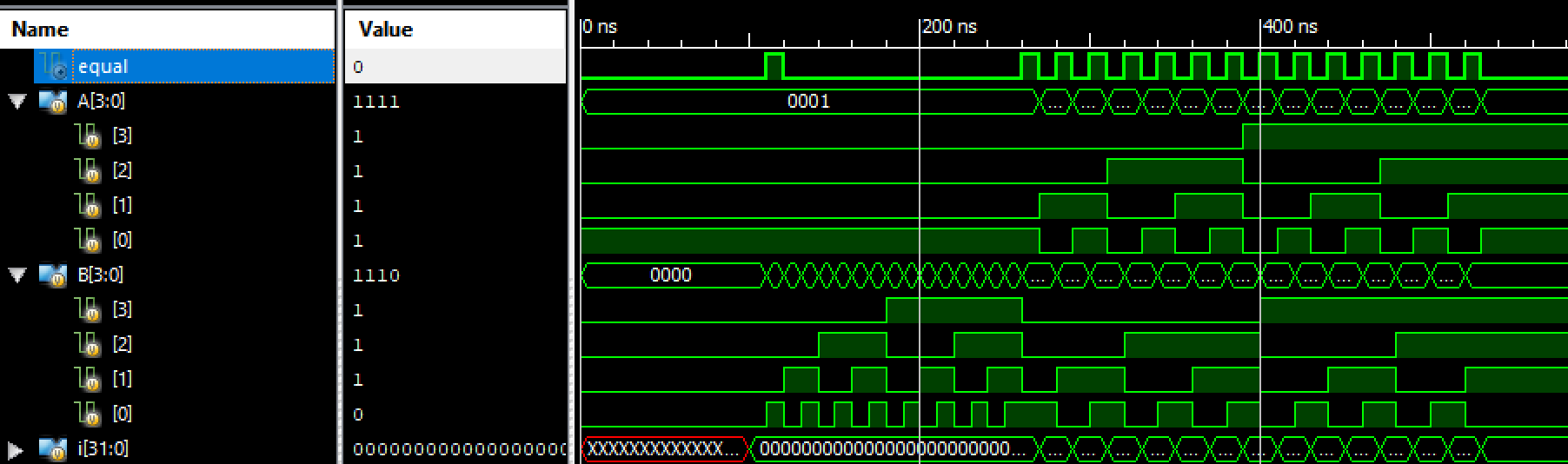


1. Frequency Divider – freqDiv

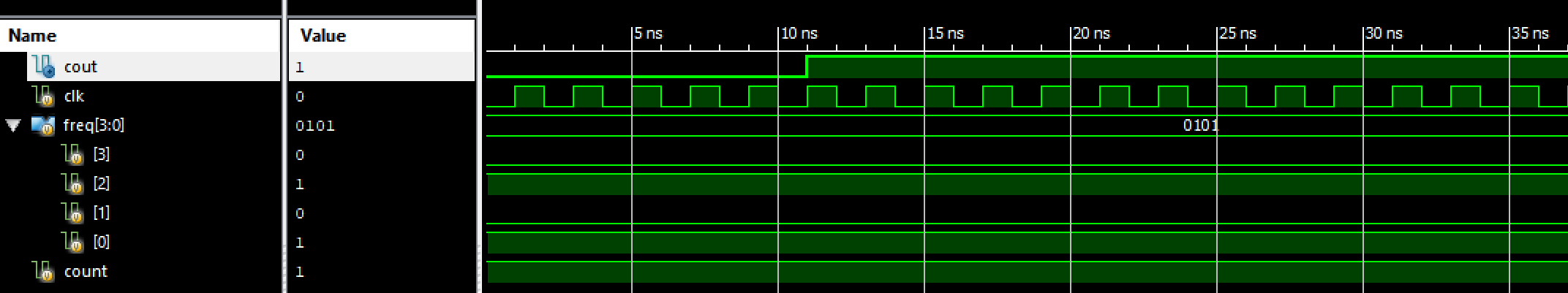
The frequency divider module takes in a input signal of a frequency through freq[3:0] and generates a output signal of cout, when triggered by a positive-edge input of a clock cycle (clk). The count input acts as an enabling signal for the module. The purpose of this module is to provide a frequency for other modules in the entire circuit.



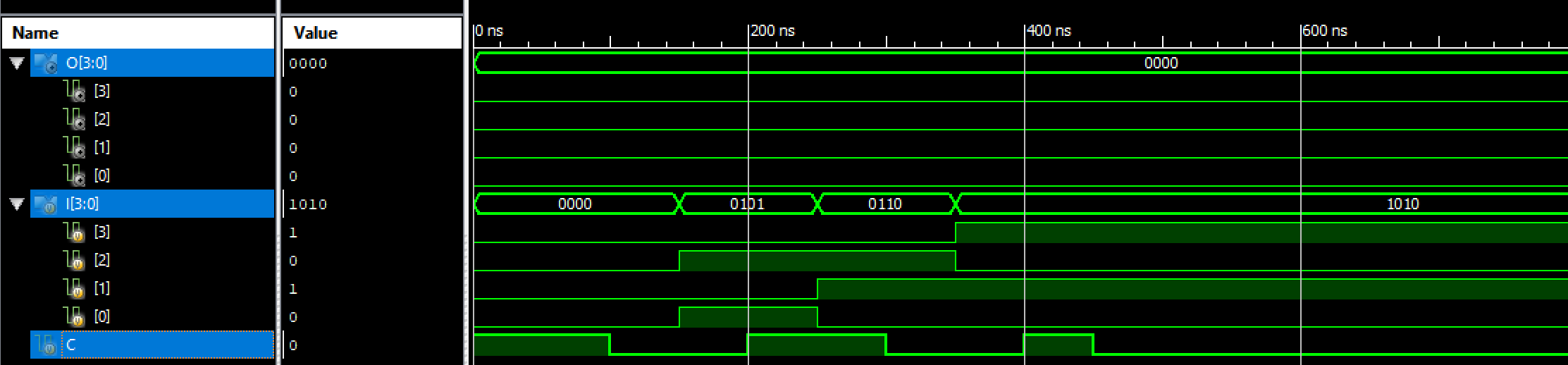
1. Flowchart
2. Simulations
3. Top (Entire Combined Circuit)
4. userInput
5. compare4bit



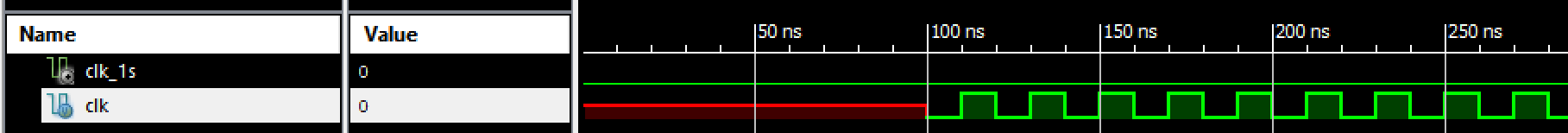
1. freqDivider



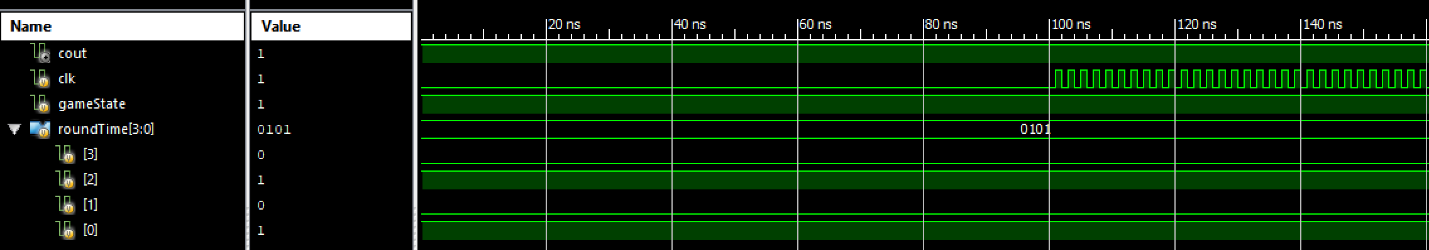
1. register4bit



1. sec\_clk



1. timing



1. scoreUp
2. stateUp
3. timeUp
4. Debugging
5. Final Comments
6. Source Code and Pinouts
7. Work Distribution

Anna Tang (Group Leader): 30%

Joshua Malmberg: 30%

Chen Yi Hui: 20%

Justin Choi: 20%

10- References