UNIVERSITY OF NEVADA LAS VEGAS. DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING LABORATORIES.

Class:	CPE200L - 1002	Semester:	Summer 2022

Points		Document author:	Daniela Nikoloska
		Author's email:	nikolosk@unlv.nevada.edu
		Document topic:	Final Project Report

Instructor's comments:

1. Introduction / Theory of Operation

For my final project, I designed an altered version of Keno using the Altera DE2-115 board and concepts learned in class. Keno is a popular casino game where players must choose 1 to 10 numbers from a set of numbers between 1 and 80. Then a random number generator (RNG) selects 20 random numbers from the same set. The more numbers match, the bigger the payout. In this altered version of Keno, the player starts off with 10 credits. Instead of a set of numbers between 1 and 80, the altered version will have a set of numbers between 1 and 15. The rules of the game are as follows: The RNG will select 4 random numbers out of the set of 15. If the player guesses 1 out of the 4 selected numbers, they will receive one point. If no numbers are correctly guessed, they lose one point. Once the player loses all the points from incorrectly guessing, they lose the game.

The project consists of the following modules: LFSR-based RNG, selector, user input, comparator, and result.

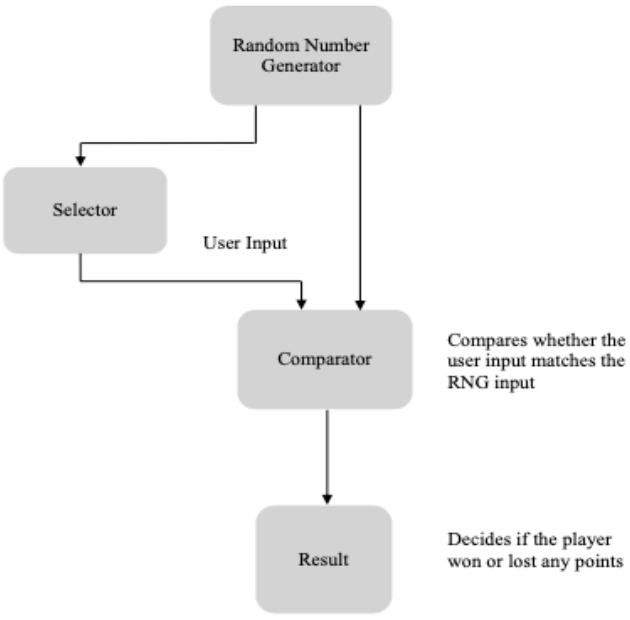

Figure 1: Submodules Block Diagram

The LFSR-based module is like the one discussed in Lab 4. For this project, the 4-bit LFSR module is used to determine a sequence of 15 pseudo-random values. A 4-bit number will be input into the RNG to begin the process. Different "taps" are then selected from the output and are processed by the XOR gate.

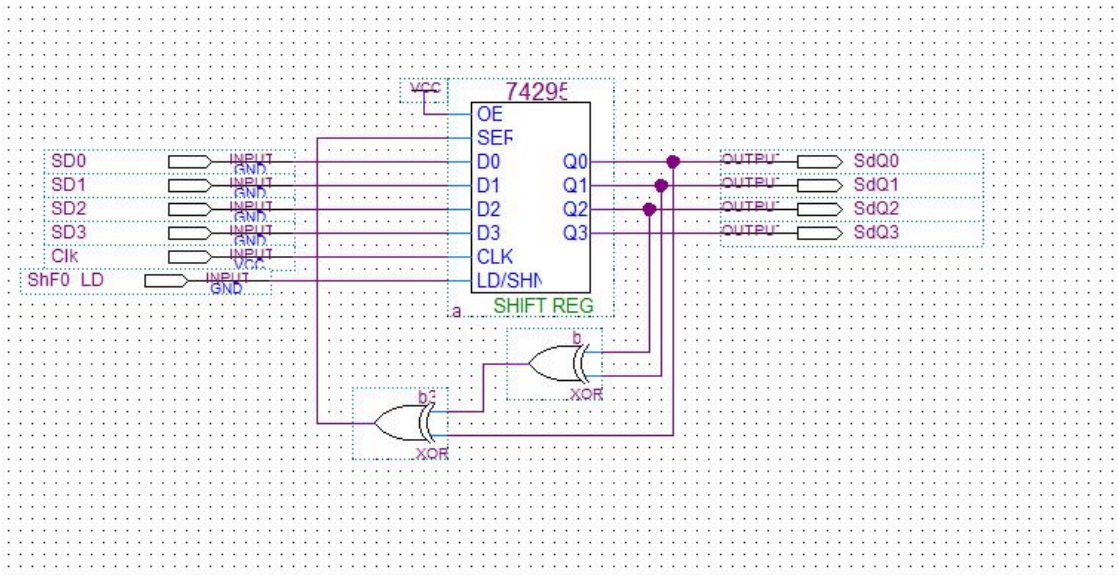

Figure 2: Random Number Generator

After the LFSR chooses a sequence of four random numbers then the player will input their selection of one number using the DE2-115 board. The user input data and RNG input will flow into the comparator module which will determine whether the user's selected number matches any of the four. Then, the data flows into the result module. If the user guessed correctly one point will be added to the 10 credits and if guessed incorrectly one point will be subtracted. Using the knowledge learned from lab 3 I decided to create a full adder for adding and subtracting multiple bits.

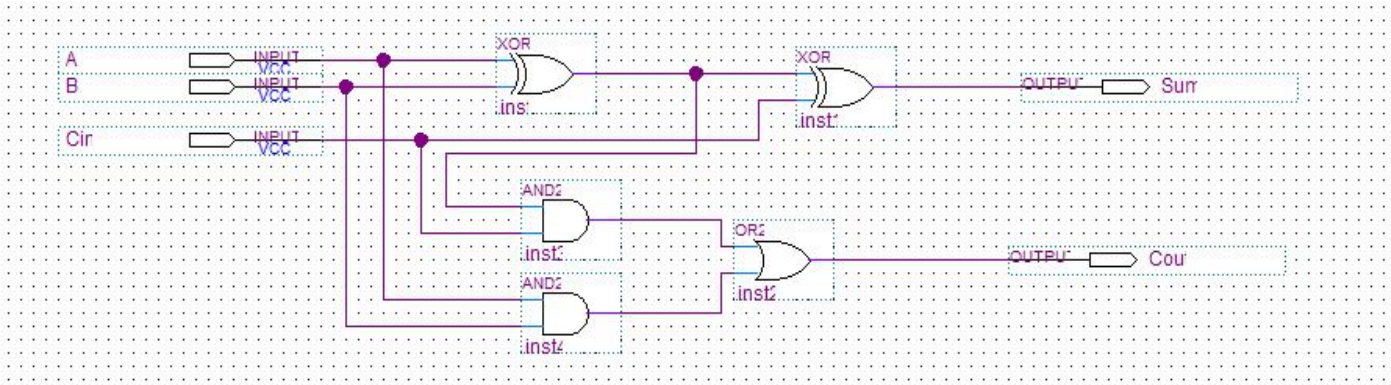

Figure 3: Full Adder

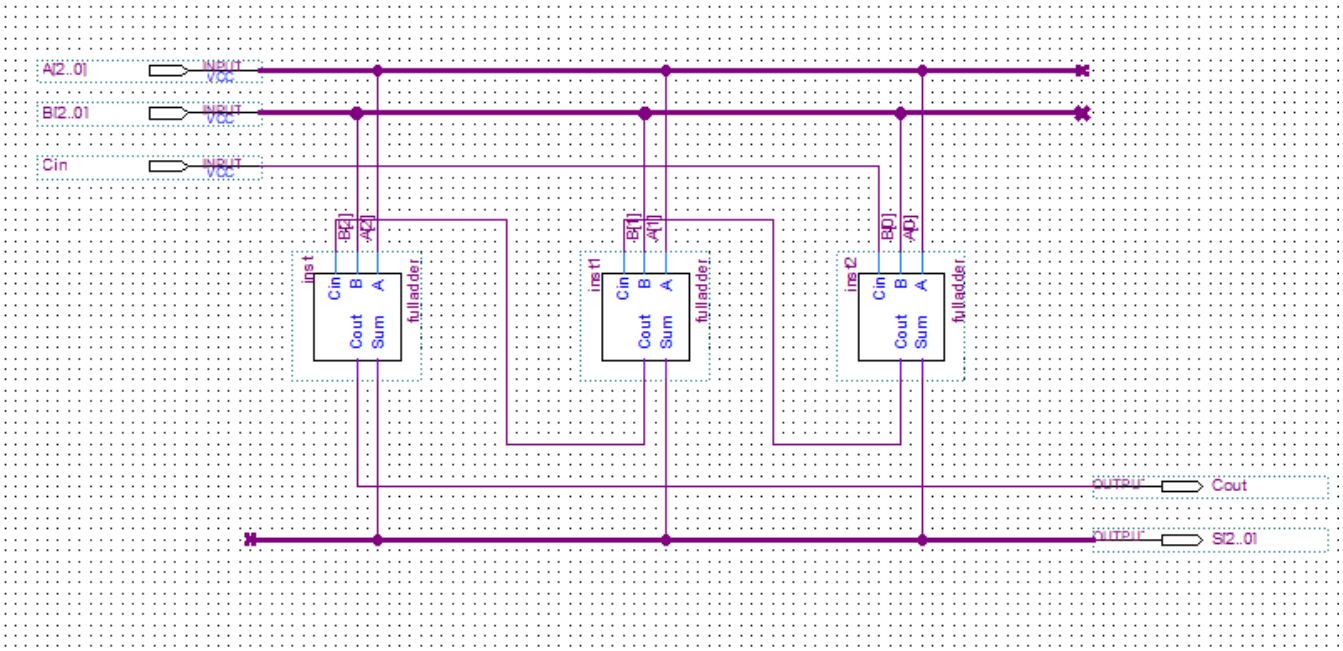

Figure 4: 3-bit Ripple Carry Adder

As shown in Figure 2 and Figure 3, the full adder was converted into a symbol file and used to create a 3-bit Ripple Carry Adder. The purpose of the 3-bit Ripple Carry Adder is to increment the player's score by 1 point.

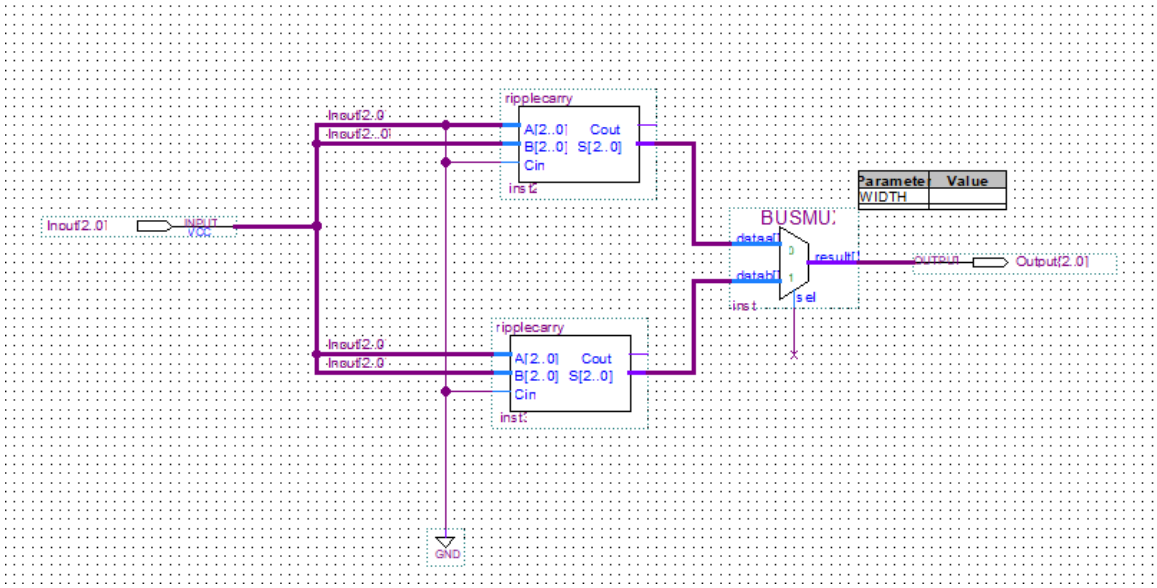

Figure 5: ALU

In figure 5, it is shown that the 3-bit ripple carry adder was used to implement an ALU that can increase or decrease the player's score by 1. The multiplexer added is meant to increment (0) or decrement (1) for the input.

2. Conclusions & Summary

Overall, I had a good understanding of what my project needed and how each individual submodule worked. With that being said, I had difficulty putting the big picture together and figuring out how each submodule should be connected. I ran into issues with the LFSR-based RNG ring I designed. As well as with the ALU which hindered me from continuing. The labs that I understood and completed without difficulty helped me create accurate submodules. I struggled to complete Lab 5 and as a result, I could not figure out the ALU submodule needed for my project.