BCD to Binary Converter

Conversion of a Decimal Input to a Binary Output

Shrutee Rakshit, Preethi Venkatesan, Sahijnoor Mahal, and Mahdee Rahman
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
e-mails: shruteerakshit@oakland.edu, pvenkatesan@oakland.edu, smahal@oakland.edu, and mahdeerahman@oakland.edu

***Abstract* - This project will convert a decimal number input via a PS/2 keyboard into a binary output that is shown on the LEDs of the Nexys A7-100T FPGA board. The purpose of this project is to convert human input decimal numbers into a form understood by computers.**

I. Introduction

This report will cover the process of designing, simulating, and implementing a BCD to binary converter. A keyboard will be used to enter the input. The outputs will be displayed on the LEDs of the Nexys A7-100T board. The input will be a 4 digit decimal value, in other words, 16 bits in BCD. The output will be 14 bits in binary.

The motivation for this project was to provide a method of translation between human input numbers and computer language. While humans have a better comprehension of the decimal system, computers only function using binary numbers. Our project provides a method for human input decimal numbers to be converted into a binary output that a computer can understand and use. This project also saves people the time and hassle of converting decimal numbers to binary numbers manually.

Several topics learned in class were applied in this project. One such topic was conversion between BCD and binary. Binary multiplication and addition were also topics that were important to this project.

Another topic utilized was finite state machines (FSMs). In this project, a FSM was used to determine when a user inputs a number by pressing a button on the keyboard, as well as keeping track of the number of decimal numbers inputted.

Additionally, everything learned in class about VHDL was crucial to this project, notably structural description.

Despite all that was learned in class, there were still some topics which required supplemental research. The process of converting BCD to binary was taught in class. However, a circuit needed to be designed to convert BCD to binary for this project. Therefore, several methods of converting BCD to binary were researched, such as the double dabble method, before choosing the one that was used for the actual project.

Another topic that was researched was how to use a 4x4 matrix keypad with the Nexys A7-100T board. However, in the end, a PS/2 keyboard was chosen for the user input instead of the matrix keypad.

This project has many applications, including keypad input from humans to machine readable numbers. Also, this project can aid engineers and computer scientists in computer programming and digital logic.

II. Methodology

A. Top Design and FSM

In this project decimal numbers were inputted through a keyboard as the human input. The output of the keyboard was transferred to the four 8-bit registers using a finite state machine which had four enable signals. Using this the keyboard output was added to the register depending on which enable is equal to 1 as the finite state machine had en0, en1, en2, and en3 for four registers. The input in those registers then went through 4 decoders. Each register had its own decoder. Those decoders decoded the 8 bit keyboard scan key to its 4 bit BCD equivalent.

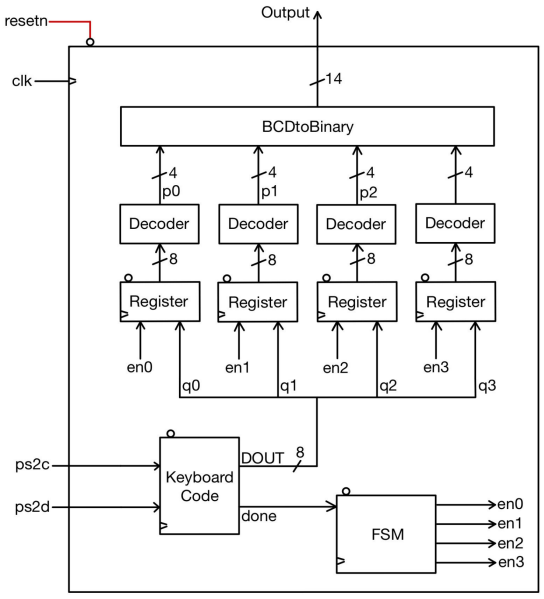

Figure 1: Top Block Diagram

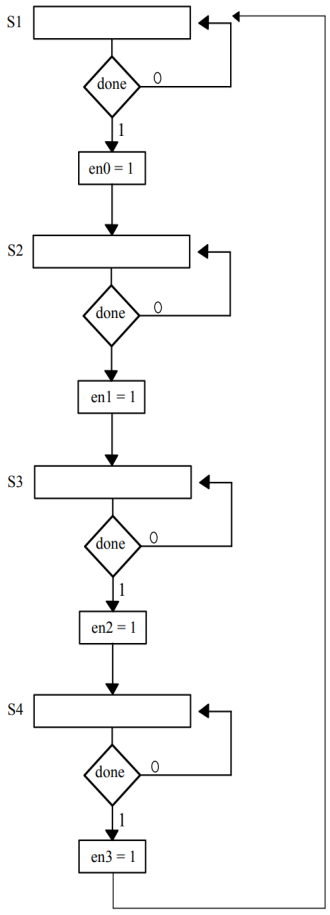

Figure 2: FSM

B. BCD to Binary Converter

The output of each decoder was sent to four 4-bit registers. The 4 bit BCD numbers from each of the four registers were further multiplied using 10-bit multipliers. As 10-bit multipliers were used, the 4 bit BCD inputs were sign extended to become 10 bits. The numbers in the first register were multiplied by 1000 and the numbers in the second register were multiplied by 100. Similarly the 3rd register's numbers were multiplied by 10 and the 4th register's numbers were multiplied by 1. The outputs of all four multipliers were 14 bits. Next, the multiplier output of the first and second register were added together using a 14-bit adder and similarly, the 3rd multiplier output was added with the output of the 4th multiplier using a 14-bit adder. The outputs of both the 14-bit adders were added together using another 14-bit adder. The output of this adder gave the 14-bit binary output which was the result of the BCD input. This BCD to binary converter only took inputs from 0000 to 9999.

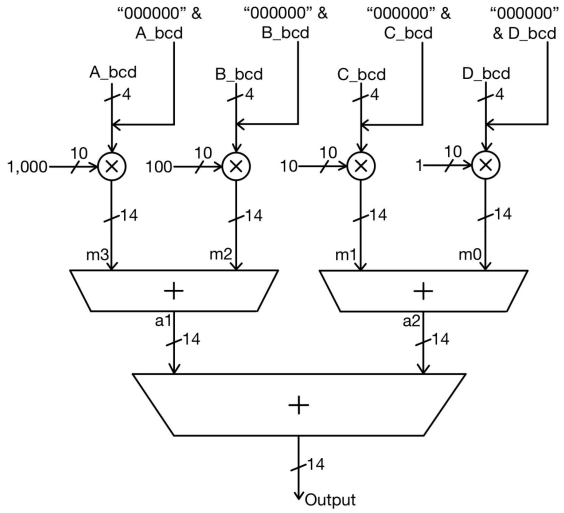

Figure 3: Conversion Circuit Diagram

III. Experimental Setup

The functionality of the project was tested using Vivado's simulation software. The individual components were tested separately to ensure individual accuracy. The results from 14-bit multipliers were simulated with a variety of 10 bit inputs, each of which were verified by their decimal equivalent. The same was done for the 14-bit adder.

The next step is to test the functionality of all the components (the combination of multipliers and adders) together in the BCD to Binary converter. The expected result would be the binary equivalent (14 bits) of a specified BCD number input (16 bits). A test bench was created to test the circuit designed for BCD to binary conversion (see Figure 4).

A PS/2 keyboard was used as the external input. Connected via usb to the FPGA board, this keyboard transfers two pieces of information: the scan code of a specific key, as well as a done signal, meaning the user has released a key.

Reading all four decimal inputs correctly via the keyboard needed to be verified via another testbench. For this, a separate Vivado project was created with the keyboard component and all its subcomponents removed. This allowed the scan code and done signal to be manually inputted as keyboard inputs. These were then provided to the FSM and following components to ensure that following keyboard inputs, the FSM can select the correct inputs. The results of this testbench are shown in Figure 5.

IV. Results

The results of the full functional project with the external PS/2 keyboard interface with the Nexys A7-100T FPGA board can be seen in the below linked video.

https://www.youtube.com/watch?v=La4y-cSSTHk

One issue that was encountered dealt with receiving the four keyboard inputs. As only four decimal numbers can be inputted for conversion, a FSM needed to be used to capture only the scan codes for the numbers that were pressed. An error in the early FSM code allowed numbers to be inputted, even if the user had not released the key. The done signal is outputted by the keyboard component when the user releases the selected key, only then can that specific key's scan code be captured.

This logic error was remedied by editing the FSM program to check the done signal after every state, and only then switching on the respective enables.

Conclusions

The mathematical converter contains four multipliers, with one for each BCD input. The outputs from the four multipliers are inputted into two adders (with each adder for two of the multiplier outputs). A final adder was utilized to add the outputs of the two adders, creating the final 14 bit binary output.

Initially, the binary output was incorrect for BCD inputs where the numbers were different from each other. After determining that this, indeed, was a logic error, it was confirmed the problem was in the FSM. After the addition of a few simple statements in the existing FSM program checking the done signal after every state, this problem was remedied.

In the end, this project successfully converted a 16 bit BCD input into its 14 bit binary representation. When given a 4 digit input, the FPGA board displayed the corresponding binary output as expected. Figure 6 shows the output on the LEDs of the Nexys board when an input of "1245" is given on the keyboard. The LEDs on the board show the correct binary output, "10011011101." Several other decimal inputs were tested to make sure that the board consistently showed the correct output.

Designing, simulating, and implementing our project led to several major takeaways. One of which is the importance of having a concrete understanding of the circuit before starting the VHDL code. Not being completely sure of the circuit before starting the code led to significant errors in the code. Once, the top design was completely changed, and that led to having to redo all of the VHDL code for the top file.

Another important conclusion gathered was closely checking the FSM logic when running into any logic errors. There was a point when the mathematical conversion between four BCD inputs was correct conditionally; to check those other conditions was the key in determining the error in the FSM program.

References

[1] D.Llamocca, *VHDL Coding for FPGAs* [Online]. Available: http://www.secs.oakland.edu/~llamocca/VHDLforFPGAs.html

[2] [Accessed 20 November 2022].

[3] Lecture Notes - Unit 6

[4] Lecture Notes - Unit 7

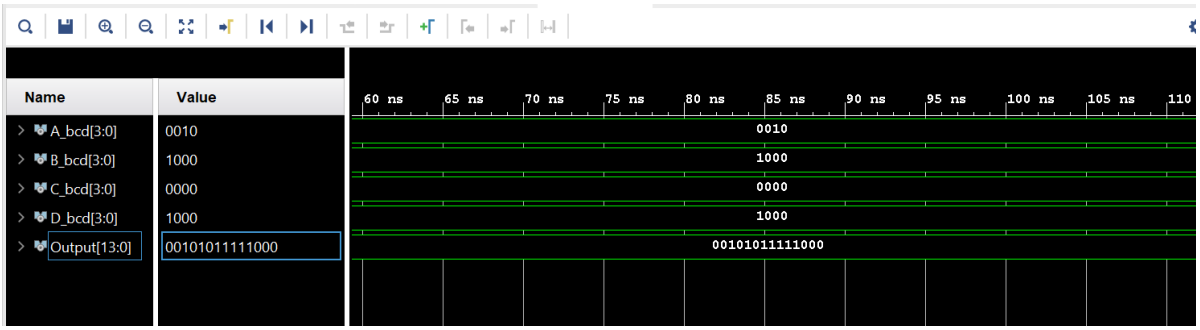

Figure 4: Testbench for BCD to Binary Converter Block

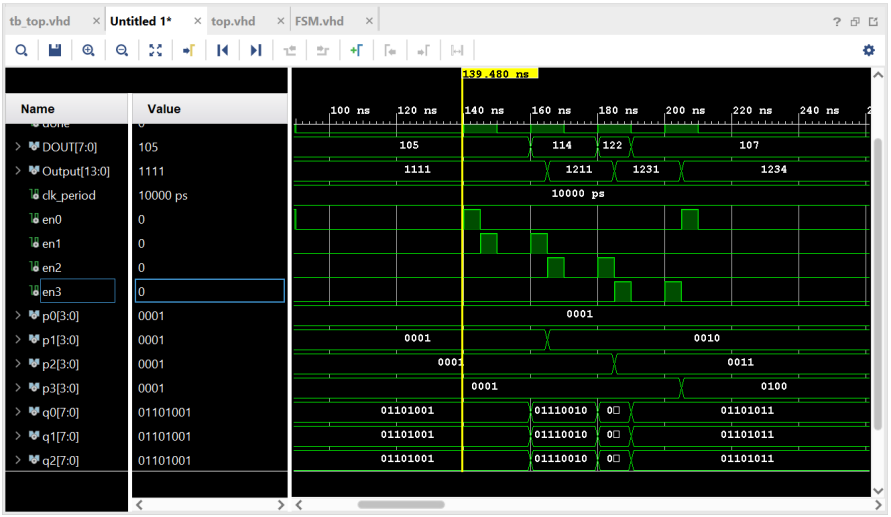

Figure 5: Testbench for Simulating Keyboard Inputs

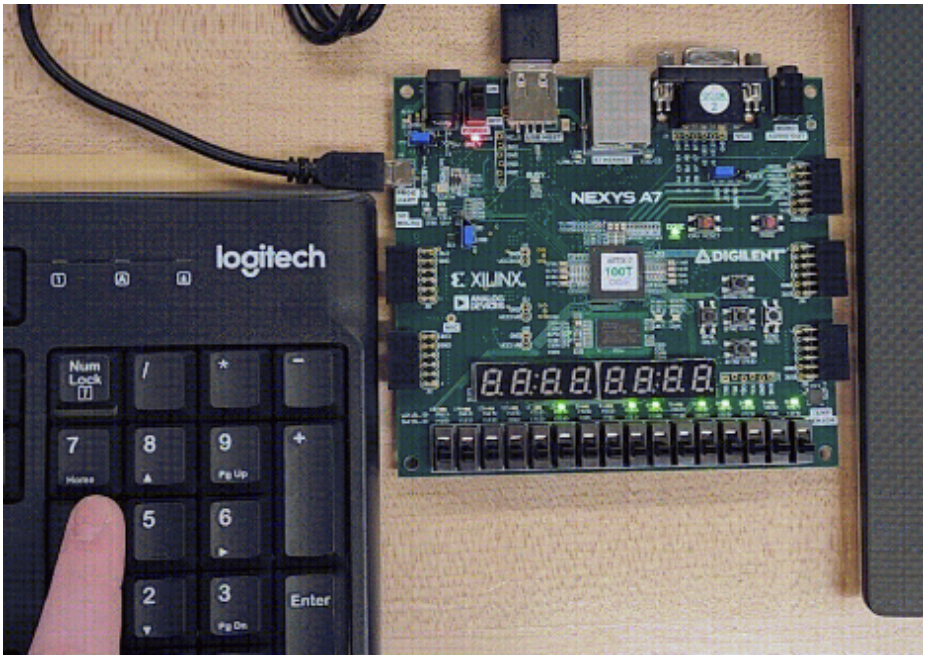

Figure 6: Results on FPGA board with Input “1245” on Keyboard