Decimal-to-Binary Converter

Design Project

List of Authors (Cody Wilson, Suha Eshaq, Caleb Bacon, Wissam Isaac)
Electrical and Computer Engineering Department
School of Engineering and Computer Science
Oakland University, Rochester, MI
E-mails: cwilson3@oakland.edu, suhaeshaq@oakland.edu, cbacon@oakland.edu, weshaq@oakland.edu

***Abstract*—**This project covered the design and implementation of a decimal to binary converter. This was done to the help students who are interested in electrical and computer engineering understand binary numbers by providing an easy to use method of converting a decimal number to its equivalent binary number. This was accomplished using Basys 3 board, Vivado software and VHSIC Hardware Description Language (VHDL) as the description language. Using concurrent description, behavioral description, structural description, sequential circuits, and Finite State Machines (FSMs) were necessary to build the project. It was found that using FSM that outputs only a ten nanosecond pulse along with the debouncer was necessary when using push buttons on Basys 3 board. It was also found that a registers was needed in order to go through the four seven-segment displays using right and left buttons.

I. Introduction

This report will demonstrate the process of creating a Decimal to Binary Converter. It covers the design, architecture, testing, and operation of this converter.

The motivation behind designing this converter was to save calculation time and give a simple representation of how a decimal to binary conversion works through the Basys 3 board. In addition it is helpful to students who are interested in electrical and computer engineering understand binary numbers by providing an easy to use method of converting a decimal number to its equivalent binary number.

Topics that were covered in the design of the converter consist of FSMs, Hex to Seven-Segments display conversion, and other mathematical operator components. A new component that was learned was a button debouncer. The debouncer component cancels out button bouncing in the mechanical action of the button. This debouncer was applied to the up, down, left, and right buttons.

This design has many applications such as checking the solution of an engineer's written homework or coding project. Another application of this converter design is to use it as a teaching element to show students how decimal to binary conversion works.

II. Methodology

The decimal to binary converter was designed to show a decimal number onto the seven-segments display and then convert it to its binary representation and have it shown with fourteen Light Emitting Diodes (LEDs). The numbers on the seven-segments display were controlled by four buttons. The left and right buttons would switch between which digit on the display and the up and down buttons would increase and decrease the value stored in the selected digit.

The circuit utilized an architectural design and was divided up into 4 primary components, the primary registers, decimal-to-binary converter, seven-segments display, and buttons. Each of these components contain a variety of decoders, multiplexores, adders, subtractors, registers, and FSMs. A drawing showing all of these components and how they are connected is shown in Figure 1. These components were designed in the Vivado software using VHDL and then implemented onto a Basys 3 board.

A. Primary Registers

The primary registers can be found in the top center part of the drawing in Figure 1. This component is responsible for storing the values that will be converted to binary. It was the first that needed to be completed because it is connected to all of the other components. It is made up of 16 D Flip-Flops in parallel which allow for each decimal digit to be stored as it's Binary Converted Decimal (BCD) equivalent. The input to the registers is from the up and down buttons and the enable for the registers is controlled by the left and right buttons.

B. Decimal-to-Binary Converter

The decimal-to-binary converter can be found in the bottom right corner of the drawing in Figure 1. It is

comprised of three multipliers and three adders which are responsible for converting the BCD values that are stored in the primary registers to binary and displaying the values on fourteen LEDs. This was done by taking each digit from the registers and multiplying them by either one, ten, one-hundred, or one-thousand in binary depending on their digit place. These multiplied values were then added together and output to the LEDs.

C. Seven-Segments Display

The seven-segments display can be found in the top right corner of Figure 1. This component allows the board to display the decimal value that is being selected by the user. Its selector is being controlled by a FSM. This FSM has four states to cycle through. Each of these states represents one of the seven-segment displays. The FSM's enable is controlled by a counter that is being operated at two-hundred and fifty hertz. This frequency allowed the user to read the seven-segment displays without seeing the board cycle through each display. The selector coming from the FSM runs through a decoder which converts the signals to a higher bit count allowing the signal (an) to be processed. This signal was used to enable the seven-segment displays.

D. Left and Right Buttons

The buttons can be broken down into two separate components due to their complexity.

The left and right buttons can be found in the bottom left corner of the drawing in Figure 1. This component is responsible for switching which digit is being altered in the registers. This is done by having a two bit signal that will either increase or decrease depending on the button pressed. When the button is pressed, it is debounced and put through a second FSM that outputs a ten nanosecond pulse. This pulse then gets put through an adder with the two bit signal. The adder output then is stored in a register so that on the two bit signal is increased or decreased by one.

A multiplexer was necessary to determine which button was pressed and whether it would add or subtract. The select line from this was attached to the output of the adder side ten nanosecond FSM with a register in between for the ten nanosecond delay it would add. This was done so when adding, the multiplexer would switch to the adder side for the one pulse and then back to the subtractor. When the buttons are not being pressed, nothing is being subtracted so the signal will remain as it was even though it going through the subtractor every ten nanoseconds.

The two bit signal for this component also used with a decoder that switches the enable for the primary registers and works as the select line for the multiplexer that goes to the up and down buttons.

E. Up and Down Buttons

The up and down buttons can be found in the top right corner of the drawing in Figure 1. This component is responsible for changing the value of the digit that is currently selected by the left and right buttons. This was done with a setup similar to the left and right buttons. The big difference here was the signal being four bits and being output to the primary registers. The input signal comes from a multiplexer with inputs from the primary registers that is controlled by the left and right buttons. This input is then added together with the button input which is debounced and sent through the FSM that outputs a ten nanosecond pulse.

The output of this adder then had to be limited to the values zero through nine because the signal is four bits and can go from zero to fifteen. This was addressed by having a decoder that when input with fifteen or ten, would output zero or nine respectively.

A multiplexer was necessary to determine which button was pressed and whether it would add or subtract. The select line from this was attached to the output of the adder side ten nanosecond FSM. This was done so when adding, the multiplexer would switch to the adder side for the one pulse and then back to the subtractor. When the buttons are not being pressed, nothing is being subtracted so the signal will remain as it was even though it going through the subtractor every ten nanoseconds.

F. Top Level Design

The top level design is where the four primary components were brought together, cation had to be taken to ensure that the least significant digit register was port mapped to the least significant slot on the multiplexers and multipliers. This was critical because an error here may have been difficult to debug.

In addition to the port mapping of all the components, the constraints file needed to be created specifically for the Basys 3 board. This file is what enabled the use of the four seven-segments display, fourteen LEDs, up, down, left, and right buttons, reset switch, and one-hundred megahertz clock..

III. Experimental Setup

Viviado was not only used as a software to develop the project but to test the functionality of the project. That was done simply by creating a test bench for the various components inside the project as well as to test the project

as a whole. Each of the primary components had their own test bench with the buttons having two for the left right and up down features.

For the decimal-to-binary converter component a test bench was created to test if the multiplication and adding processes were functioning properly. That was done by multiplying a ones place value by one, tens place value by ten, hundreds place value by one-hundred and finally thousands place value by one-thousand. Those values were chosen randomly for test purposes. The output of the multipliers then went to three adders which determined the sum of the numbers so they could be displayed in binary. Adding a delay of one-hundred nanoseconds between each set of values was also necessary for process of testing to ensure the values were fully evaluated.

The second component that was simulated was the right and left buttons. These were tested separately from the up and down buttons due to complexity of the buttons.

In order to test this component, a clock process was made so that the clock is high for five nanoseconds and then low for the next five nanoseconds. Then a reset that was low for one-hundred nanoseconds and high for the remainder of the simulation. Finally, each one of the buttons were given either a high or a low value with a delay in after each to check whether the output would change based on the value given to each button. These had to be ran for at least forty milliseconds because of the debouncer requiring time to properly debounce the input signal.

The third component tested was the up and down buttons which were supposed to increment the number displayed in each 7-segment display. This component was tested using almost same strategy as the one used for right and left buttons. The difference was that the up and down buttons needed to be tested to ensure that if the up button was pressed after reaching nine, the register would remain at nine. It also needed to ensure that if the down button was pressed after reaching zero, the register would remain at zero.

The fourth component tested was the seven-segments display. This component was tested by simulating random values for the four inputs on the multiplexer and watching the seven-segments display output. The output was designed to switch between each of the four seven-segment displays and show the proper value from the multiplexer. It was also meant to be switching at a frequency of two-hundred fifty hertz so that they display could be readable. To achieve this, A clock process made so that the clock is high for five nanoseconds and then low for the next five nanoseconds.

Finally, a test bench was created for the top file to test the project as a whole. A clock process made so that the clock is high for five nanoseconds and then low for the next five nanoseconds. A reset that was low for one-hundred nanoseconds and high for the remainder of the simulation was also added. Then, the buttons were given values of high or low to observe what is being output in the four seven-segments displays and LEDs.

IV. Results

The results of this project were obtained first by running several simulations using Vivado software. Later on, after completing the testing primary components and the project as a whole, those results were implemented to the Basys 3 board by creating a constraint file to enable the LEDs, 7-segment displays, push buttons and the switch on the board.

It was found that FSMs were necessary when using the buttons on the board. By pushing the button and let it go, the FSM takes that input through two states. In state one, when the button is high (while it's being pushed) it outputs a high signal then it goes to second state. In the second state, it checks if it's still high. If yes, it keeps it in state two. If input is low (not pushed/ let go) it goes back to state one.

Without having this FSM, it was not possible for the button to work properly for the purpose it was used for. It was also explainable that before using the clock reducer FSM, the project was not working properly as it was not possible to have the decimal number displayed on the four seven-segment displays.

Conclusions

The main take-away point of this project is learning the importance of each component and how it works along with other components to create the whole project. The architecture design was completed based on results were obtained from testing it. Throughout the progress of developing the project, it was noticed that few components were missing in order to complete the project. As example for that, a clock reducer FSM, an FSM that outputs a ten nanosecond pulse to the right-left buttons, and registers to delay the output of right-left buttons as shown in Figure 1. Further development can be made to allow the user to convert a number in binary to its decimal equivalent. The user can use the switches to input the binary number and see the results of conversion in the four 7-segment displays.

References

[1] Prof. D. Llamoca, "Generic Array Multiplier (NxN bits, unsigned)," Unit 4, VHDL Projects. http://www.secs.oakland.edu/~llamocca/VHDLforFPGAs.html

[2] Prof. D. Llamoca, "(Debouncing Circuit)," Unit 7, VHDL Projects. http://www.secs.oakland.edu/~llamocca/VHDLforFPGAs.htm

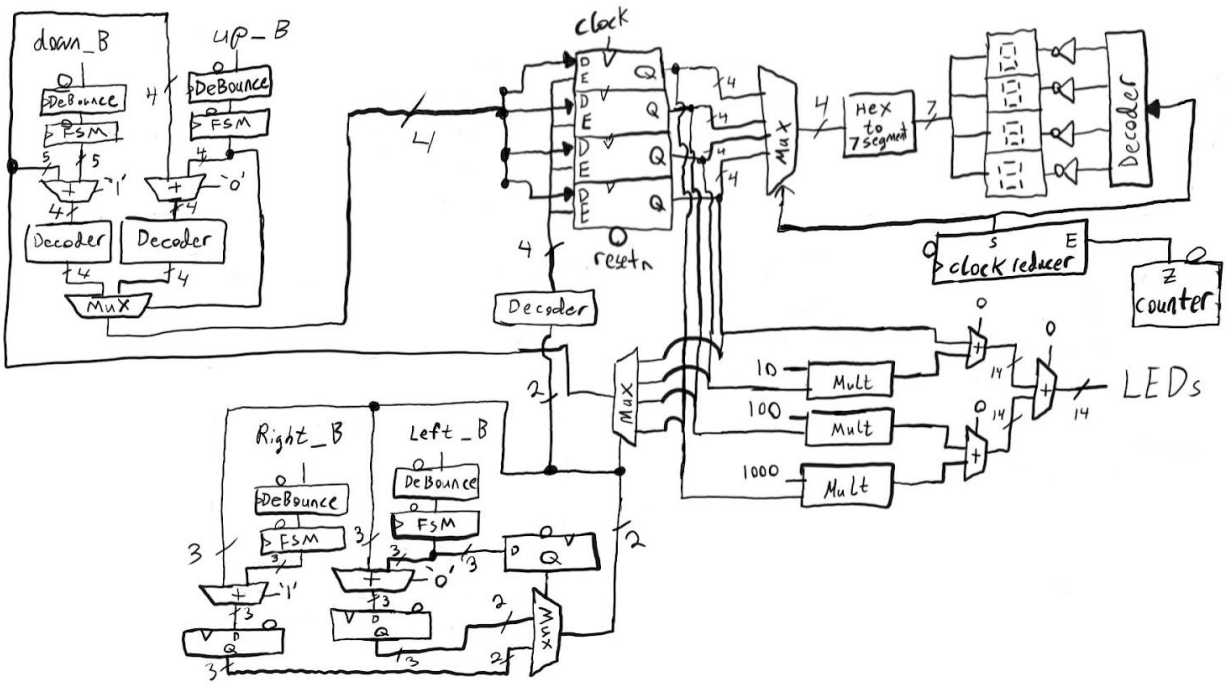

Figure 1. Hand Drawing of the Decimal-to-Binary Architecture.