ECEN 429: Introduction to Digital Systems Design Laboratory

North Carolina Agricultural and Technical State University

Department of Electrical and Computer Engineering

Freddie Boadu (Reporter)

February 6, 2020

Pre Lab #2

**Introduction**

Seven Segment LED displays are used to many digital systems. They are used in shot clocks at basketball games, and clocks in the house on appliances. Encoders and Full adders are always very important in digital logic. Encoders allow translation in digital systems and adder allows carry in and carry outs to be used. The purpose of this prelab is to introduce us to Seven segment LED displays, encoders, and full adders. This will be done by answering 3 questions that are part of the prelab. First question is dealing with assignment of the pins to the Seven-Segment LED display, second is dealing with encoders and the code surrounding the figure provided, and the third question is centered around a full adder. It is important that we know how to do all of these in VHDL because they are a foundation for most of the digital systems we encounter today.

**Background, Design Solution and Results**

This prelab provides us some same VHDL code that displays how a Seven-Segment Display should be done. We are also responsible for coding and making a truth table for the encoder provided in the prelab document. We are also supposed to make a code for a full adder with 3 inputs.

Problem 1:

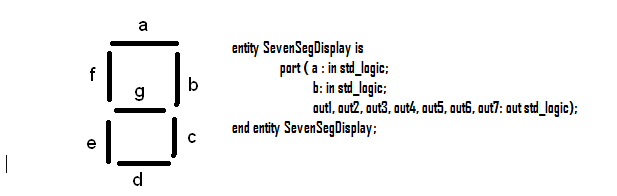
Problem 1 has given us a Seven-Segment display. We need to figure out the FPGA pins that would be used for each output that will be used. The segment has 7 outputs that need to be accounted for. We can figure out which pins need to be assigned based on the reference manual for the Basys-3 that is provided to us. The prelab provides us with a diagram to assist us further in labeling everything: 

Figure 1.1 The Seven-Segment display code provided by the prelab. It includes two inputs and seven outputs to make the device work.

In the document, they reference to A as CA and B as CB, and so forth. The pin assignment will be according this. The figure provided below shows what the Basys-3 reference manual has provided us with:

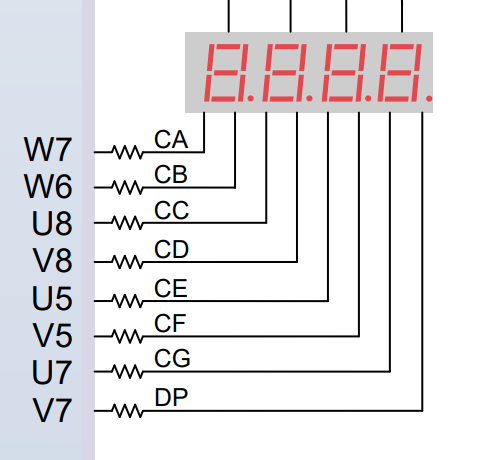


Figure 1.2 pin assignment according the Basys-3 reference manual.

The pin assignment is as follows:

|  |  |
| --- | --- |
| Pin | Output |
| W7 | Out1(CA-A) |
| W6 | Out2(CB-B) |
| U8 | Out3(CC-C) |
| V8 | Out4(CD-D) |
| U5 | Out5(CE-E) |
| V5 | Out6(CF-F) |
| U7 | Out7(CG-G) |

Table 1.1 This table is the pin assignment for the Seven-segment display according to the code provided to us in the prelab for problem one. Each pin is assigned to corresponding part of the display and it will light up accordingly based on what number is being asked for and designed on the board.

Problem 2:

The second problem of the lab deals with an encoder. The encoder is a 4:2. We are to draw a truth table for the display and then describe how we would do the code in VHDL. The image of the encoder is drawn accordingly:

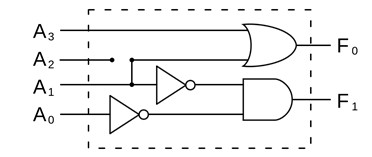


Figure 1.3 this is the encoder this provided for problem 2. The figure has 4 inputs(A0-A3) and 2 outputs(F0-F1).

The truth table for this figure is:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A3 | A2 | A1 | A0 | F0 | F1 |
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | x | x |
| 0 | 1 | 0 | 1 | x | x |
| 0 | 1 | 1 | 0 | x | x |
| 0 | 1 | 1 | 1 | x | x |
| 1 | 0 | 0 | 0 | x | x |
| 1 | 0 | 0 | 1 | x | x |
| 1 | 0 | 1 | 0 | x | x |
| 1 | 0 | 1 | 1 | x | x |
| 1 | 1 | 0 | 0 | x | x |
| 1 | 1 | 0 | 1 | x | x |
| 1 | 1 | 1 | 0 | x | x |
| 1 | 1 | 1 | 1 | x | x |

Table 1.2 This would be a truth table to for a 4:2 encoder. The encoder would go up to the 0-3. The rest of the results would be don’t cares.

For the second part of the problem 2, we were asked to do some pseudo code for the encoder. The code would be close to:

Entity fourbytwoencoder is

Port( A: in std\_logic\_vector(3 downto 0);--Represents A0-A3

F: out std\_bit\_vector (1 downto 0);--represents F0 and F1

End;

Architecture bev of fourbytwoencoder is

Begin

Process (A)—represents the A0-A3

Begin

Case A is

When “0000”=> F<=”00”;

When “0001”=> F<=”01”;

When “0010”=> F<=”10”;

When “0011”=> F<=”11”;

when others =>F<=”xx”;

end case;

end process;

end;

This code will simulate the encoder. It is roughly VHDL.

Problem 3

For problem 3 we are dealing with a full adder. We are asked to find the expression for SUM. We are given the expression for Carry out as CO = CI **and** (A **or** B) **or** (A **and** B) and the VHDL code for it is CO <= CI **and** (A **or** B) **or** (A **and** B). For sum, the expression for it would be SUM=(A xor B) xor CI and for VHDL, SUM= (A xor B) xor CI.

The full truth table for the full adder is:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CI | B | A | Sum | C-out |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

Table 1.3 this is a table to represents a full adder with the inputs of A,B,CI(carry in) and the outputs of Sum, and CO(carryout).

The VHDL code for this full adder would look like:

Entity fulladder is

Port( CI,B,A: in std\_logic;

Sum,Cout: out std\_logic);

End;

Architecture bev of fulladder is

Begin

Sum<= A xor B xor CI;

CO<= (A AND B) OR (A OR B) AND CI;

End;

**Conclusion**

After doing the prelab, I understand how a Seven-Segment LED display, encoders, and full adders work in VHDL. Preforming the prelab is going to help me understand how to better program these three problems listed in the prelab. The challenge of this prelab was trying to understand how significant each one of the features were to digital design. Segments are used in clocks, encoders are used a lot of conversions of decimal to binary and etc, and the adders are used for math operations that wouldn’t be possible with the adders.