**ELEC 204 Digital Design Lab Report**

Lab 1

Name: Metehan Gelgi

64178

Date: 10.12.2019

1. **Introduction and objectives**

In this lab, our aim was designing a 3-bit comparator module that can compare two 3-bit binary inputs and indicate if these inputs are equal, smaller or greater than each other. Inputs are unsigned binary values. I compared 3-bit values by using VHDL code. In our code I implemented if-else statement to compare values. First, I have compared most significant bits of values. Then I compared next bits of values by assuming most significant bits are same (ie. A=101 B=110 to compare second bits: AG = (A2B2+22)A11, BG=(A2B2+22)B11 ).For the most significant bits are not same, still this process will work correctly, because algorithm will ensure that this situation is correct which I explained in methods part.

1. **Methods**

This VHDL code has 6 different with in 2 bus (by 3 bits) inputs.

=>A0, A1, A2 (in form of A2A1A0 for 3-bit variable)

=>B0, B1, B2 (in form of B2B1B0 for 3-bit variable)

And 3 signals independent of the objective of lab to get rid of writing too many codes. (defining)

=>x0= A0B0+00

=>x1= A1B1+11

=>x2= A2B2+22

Has 3 different outputs

=>AG= A22 + X2A11 + X2 X1A00

=>BG= B22 + X2B11 + X2X1B00

=>EQ= X0X1X2

This VHDL code has to compare 2 different 3 bits inputs then return “0” or “1” for AG, BG, EQ. Process is based on kind of simple if else statement. Because first check whether most significant bits are same or one of them larger. Then passed to the second bits for these inputs. For second bits I give one more and gates to this step which is X2.This X2 transfers the checksum of the first bits to the second bits, and seconds bits checked in the same manner with first bits. For last step (third bits of statement which is A0B0) I add two and gates to third bits comparing that is X2 and X1 (X2 transfers 1st bits knowledge X2. transfers2nd bits knowledge to the 3rd bits)

.ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Figure 1: Code of VHDL

Truth table also supports VHDL code. By this truth table we can see that when A2 is greater than B2 other bits of A and B are not important for comparing.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Truth Table for 3-bit comparator | | | | | |
| A2B2 | A1B1 | A0B0 | A<B | A=B | A>B |
| A2 > B2 | X | X | 0 | 0 | 1 |
| A2 = B2 | A1 > B1 | X | 0 | 0 | 1 |
| A2 = B2 | A1 = B1 | A0 > B0 | 0 | 0 | 1 |
| A2 = B2 | A1 = B1 | A0 = B0 | 0 | 1 | 0 |
| A2 < B2 | X | X | 1 | 0 | 0 |
| A2 = B2 | A1 < B1 | X | 1 | 0 | 0 |
| A2 = B2 | A1 = B1 | A0 < B0 | 1 | 0 | 0 |
| X= Does not matter | | | | | |

Table 1: Truth table for 3-bit comparator

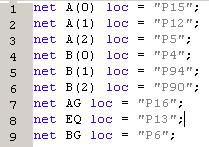
metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Figure 2: RLT Schematics of VHDL code

**Implementation on Board**

Then Implementation Constraint file was opened an to implement VHDL code to FPGA board as inputs and outputs. 3 serial locations were given for A and B and 3 LEDs were given for outputs. There are six bits in inputs. (A(0), A(1), A(2), B(0), B(1), B(2)). There are three bits in outputs. (AG (A Greater), BG (B Greater), EQ (Equal))

**Figure 3: FPGA Board locations**

elektronik eşyalar, devre içeren bir resim

Açıklama otomatik olarak oluşturuldu

Figure 4: FPGA board EQ Example elektronik eşyalar, devre içeren bir resim

Açıklama otomatik olarak oluşturuldu

Figure 5: FPGA board AG Example

**EQ => A=000, B=000**

**AG => A=001, B=000**

elektronik eşyalar, devre içeren bir resim

Açıklama otomatik olarak oluşturulduFigure 6: FPGA board BG Example

**BG => A=001, B=011**

1. **Problems encountered, errors and warnings resolved**

I encountered a couple of problems while doing this lab. First of all, 3-bit comparator is not like 2-bit comparator. It has different process from 2-bits or 1-bits comparison. That’s why I had long way to do. First problem that I encountered is comparing every bit separately. On this step I used kind of if else statement. By this method I could check step by step.

For second problem is our code was so complicated, so I had to use some methods to get rid of this complexity which restrain of legibility of code. I have overcome this problem using signal variables (X2 , X1 and X0). (can be seen in Figure 1)

I had a couple of syntax error. Because it is a new programming language for me. Then I could handle this problem.

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Figure 7: Syntax Error

Another error is, I used different names for AG, BG, EQ in main file and ucf file by mistake. That’s why I got error. Then I changed names of outputs to handle with this problem.

ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Figure 8: Variable Error

1. **Conclusion**

In this Lab I prepared schematic to implement on VHDL code for 3-bit comparator. First, I have created k-map for 3-bit inputs and outputs to compare them. I have taken the simplest forms of outputs by using these tables. These simplest forms are used for VHDL code to implement on board. I used these simplest forms because by this way I can get most efficient program. In this code, I used signal variables to get rid of complex code that needs to be read by the user. I got this idea by watching some videos on youtube(1). I also have learned so much things which will be useful for digital design on computer. Because it is a first lab, I learned how to use Xilinx to implement our truth tables and k-maps on the board. I got a couple of syntax errors then I fixed them. Overall this lab is starting point of digital design and Regardless of the language, our focus is digital design which will be so significant for thinking on computer systems.

References

1. <https://www.youtube.com/watch?v=8nKnq_9jomQ&list=PLEKaxav-vKEbupmPe676FvOM3v3AzdorM&index=15>
2. Marro, Ciletti (2018). Digital Design. New York, NY; Pearson
3. <https://ku.blackboard.com/webapps/blackboard/execute/displayLearningUnit?course_id=_28361_1&content_id=_222385_1>
4. <https://www.semanticscholar.org/paper/Area-and-power-efficient-4-bit-comparator-design-by-Sharma-Sharma/2146d2435eb2e510faabbb096bb8bf98643779c9/figure/0>