ECSE 222 Fall 2022

VDHL Assignment 5 Report: Sequential Statements and a 4-Bit Comparator

Part 1 – Participants

Karim Elgammal 260920556

Lutming Wang 261006348

Part 2 – Executive Summary

Throughout this lab we have been able to explore implementations of a 4-bit comparator, storage elements, and sequential circuits. Furthermore, with these implementations we were introduced to implementations of these circuits using sequential assignment statements in VHDL, types of statements that allow for sequential circuits to be implemented. Up to this point we really have only been using concurrent assignment statements to implement our circuits and so this lab served as a familiarization of the sequential type of assignments. Most of the implementations within this lab are behavioral implementations that take advantage of this introduction to sequential assignment statements. Some have used a structural style of implementation.

Part 3, 4 & 5 : Implementation and Results

4-Bit Comparator

The 4-Bit Comparator circuit was implemented using a behavioral architecture style; the logic of the implementation consists of using the IEEE.std_logic_unsigned.ALL package to compare the magnitude of each of the inputs. This allows us to directly compare the numbers using VDHL's unsigned package. As seen below, we used four blocks of if statements to catch all the cases that could happen according to the specifications of the VDHL 5 assignment instructions. These four cases are evaluated by the condition, and then depending on the case, the signals of each of the signifiers are assigned to the appropriate value. Moreover, the order in which these conditions are declared are important, as we know that the overflow case must be caught before any comparing is done, as only the overflow signal is high in that case.

```
architecture beh of karim_elgammal_comparator is
begin
    process (A, B)
        begin
            overflow <= '0';
    if B = 15 then
    overflow <= '1';
    AgtBplusOne <= '0';
    AgteBplusOne <= '0';
    AltBplusOne <= '0';
    AlteBplusOne <= '0';
    AeqBplusOne <= '0';

    elsif A = (B + 1) then
    overflow <= '0';
    AgtBplusOne <= '0';
    AgteBplusOne <= '1';
    AltBplusOne <= '0';
    AlteBplusOne <= '1';
    AeqBplusOne <= '1';

    elsif A > (B + 1) then
    AgtBplusOne <= '1';
    AgteBplusOne <= '1';
    overflow <= '0';
    AltBplusOne <= '0';
    AlteBplusOne <= '0';
    AeqBplusOne <= '0';
```

```
    elsif A < (B + 1) then
    AgtBplusOne <= '0';
    AgteBplusOne <= '0';
    overflow <= '0';
    AltBplusOne <= '1';
    AlteBplusOne <= '1';
    AeqBplusOne <= '0';
    end if;

    end process;
end beh;
```

Figure 1 - Behavioral Implementation of 4 Bit Comparator

Results:

With A = $5_{(10)}$

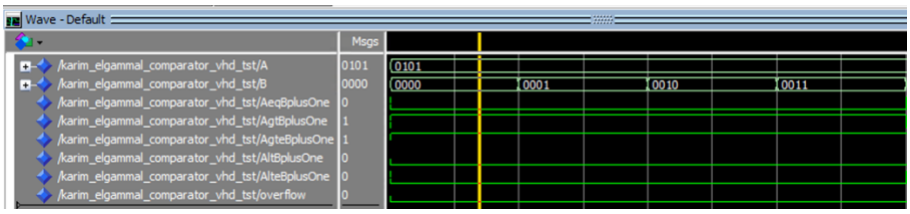

Figure 2A - As observed above, the correct signals become high as long as A is greater than B + 1. This is true until our next case below when A = B + 1.

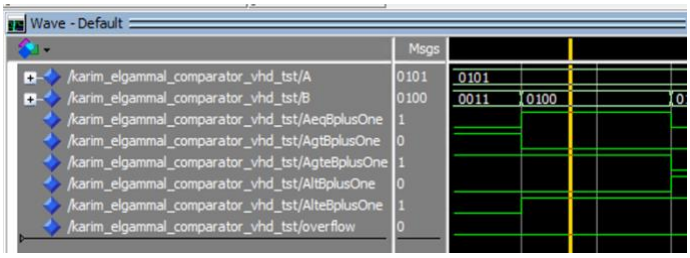

Figure 2B - This is the case of when A is equal to B + 1. We can observe when B is 4, that the signals which indicate an equality become all high.

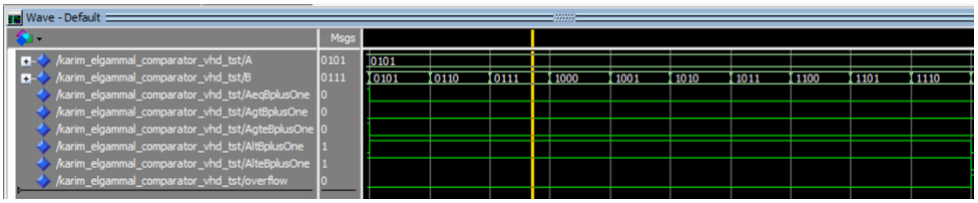

Figure 2C - *This case showcases when A is less than B + 1, and it can be observed from the signals which are high that the comparator behaves accordingly to the logic presented within the instructions.*

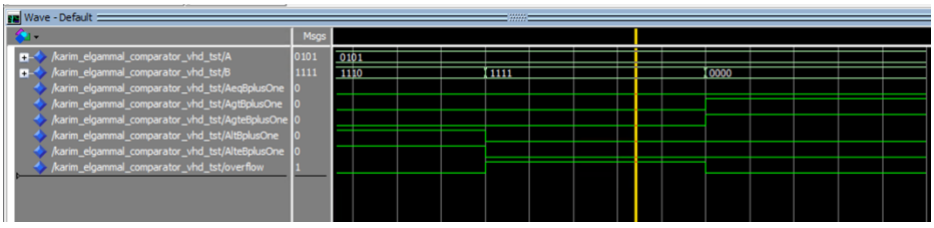

Figure 2D - Finally, this is the case when B + 1 cannot be expressed in 4 bits, meaning that an overflow has occurred. We can see here that the circuit designed signifies the overflow case and stops any other signal from being high if there is an overflow present.

Timing Analysis:

Using the Quartus timing analysis tool, we were able to set restraints to the timings of each of the inputs and outputs within an SDC file. The constraints are inputted as follows. Where each of the inputs are compared against each of the outputs in each line. We set a time constraint of 5ns to catch the paths which are in violation.

```
set_max_delay -from [get_ports A[*]] -to [get_ports AgtBplusOne] 5
set_max_delay -from [get_ports A[*]] -to [get_ports AgteBplusOne] 5
set_max_delay -from [get_ports A[*]] -to [get_ports AltBplusOne] 5
set_max_delay -from [get_ports A[*]] -to [get_ports AlteBplusOne] 5
set_max_delay -from [get_ports A[*]] -to [get_ports AeqBplusOne] 5
set_max_delay -from [get_ports A[*]] -to [get_ports overflow] 5

set_max_delay -from [get_ports B[*]] -to [get_ports AgtBplusOne] 5
set_max_delay -from [get_ports B[*]] -to [get_ports AgteBplusOne] 5
set_max_delay -from [get_ports B[*]] -to [get_ports AltBplusOne] 5
set_max_delay -from [get_ports B[*]] -to [get_ports AlteBplusOne] 5
set_max_delay -from [get_ports B[*]] -to [get_ports AeqBplusOne] 5
set_max_delay -from [get_ports B[*]] -to [get_ports overflow] 5
```

Figure 3A – SDC File Limitations

Through this method we came to the paths in violation of this constraint by way of the Timing Closure Recommendations of the TimeQuest timing analyzer. This gave us a list of violating paths shown in figure 3B. From this analysis we can see that the longest delay happened only within the B inputs. This is quite plausible as the B input goes through the most operations (inevitably gates) to reach the output. Operations such as adding 1 to the input and comparing to the A input. As we see the highest slack is between B[0] and the overflow signal, with a slack of -4.673. This tells us that this is indeed the critical path of the 4-Bit comparator. Furthermore, we can calculate the delay of the critical path by adding the positive slack to the limitation of 5ns. So, $Delay = 5ns + 4.673ns = 9.673ns$.

Timing Closure Recommendations

Summary [hide details]

This design contains failing setup paths with a worst-case slack of -4.673 ns. Run Report Timing Closure Recommendations for recommendations on how to close setup timing. For recommendations for any particular path, click the appropriate link in the table below.

Top Failing Paths [hide details]

	Slack	From	To	Recommendations
1	-4.673	B[0]	overflow	Report recommendations for this path
2	-4.661	B[1]	overflow	Report recommendations for this path
3	-4.623	B[3]	overflow	Report recommendations for this path
4	-4.585	B[2]	overflow	Report recommendations for this path

Figure 3B – Violating Paths of 4 Bit Comparator Circuit

Top-level Entity Name	karim_elgammal_comparator
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	15 / 32,070 (< 1 %)
Total registers	0
Total pins	14 / 457 (3 %)

Figure 3C – Reporting of Logic Utilization and Total Pins to Implement Design

Implementation on the DE1-SoC Board:

After performing the timing analysis of the 4-bit comparator, we used the pin planner to control the inputs A and B using slider switches SW3 – SW0 (for A) and SW9- SW6 (for B), where SW0 and SW6 is the LSB of A and B respectively. Further, we then connected the output signals; overflow, AeqBplusOne AgtBplusOne, AgteBplusOne, AltBplusOne and AlteBplusOne to LED0 – LED5 respectively. This allowed us a fully functional comparator on the FPGA board. Figure 4 showcases each case of our working program.

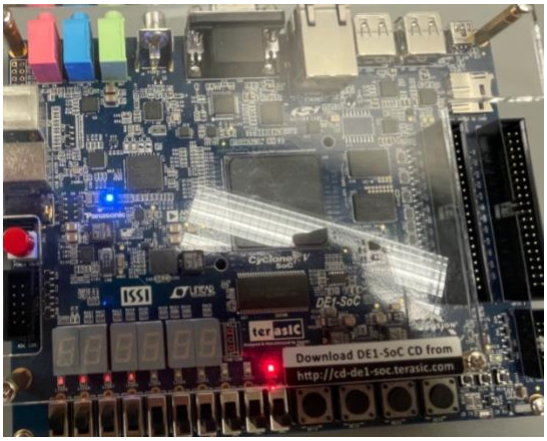

Figure 4A: Overflow case (when B = 15)

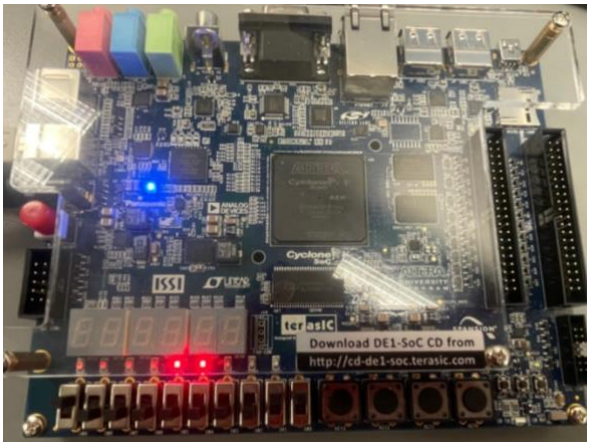

Figure 4B: A<(B+1), A ≤ (B+1) case (A= 3, B= 8)

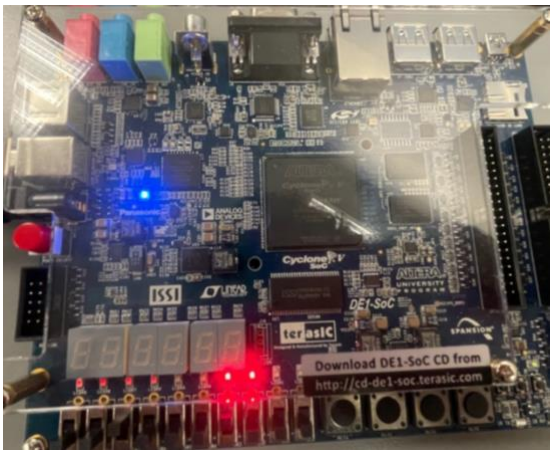

Figure 4C: A>(B+1), A ≥(B+1) case (A=8, B=2)

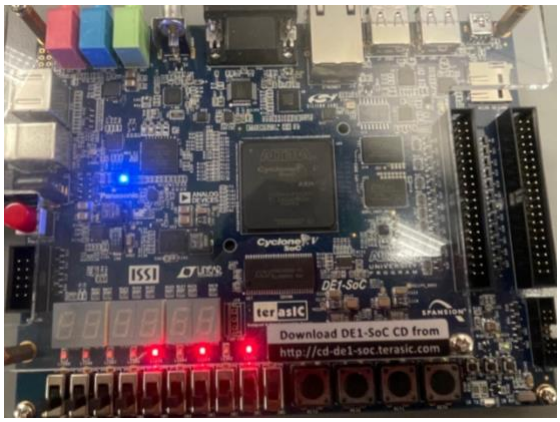

Figure 4D: A=(B+1), A ≤ (B+1), A ≥(B+1) case (A=1, B=0)

Part 6: Conclusions

Throughout this investigation, we have been able to successfully implement a 4-bit comparator and run the implementation on the DE1-SoC board. Our designs utilize a behavioural implementation of code, which allowed for an efficient design process that gave way for easy debugging. Further, we have incorporated the requirements of the design successfully. Further, we have been able to perform timing analysis on our design to arrive at the critical path of the design.