**A picture containing logo

Description automatically generated**

**King Fahd University for Petroleum & Minerals**

**Mini Project Report:**

**Descent Timer**

Date:4/12/2020

**Prepared by:**

ID:201855360, Ammar Alfaifi, Sec:54

ID:201840840, Sami Alfaifi, Sec:53

**For:**

**COE203 - 201**

**1.Objectives:**

In this project we aim to design a descent timer based on Verilog and programmed on an FPGA board. Also, we will simulate the project based on computer-generated clock signals

**2.Proceduer:**

Task1: Designing mod10:

This module is used to count from 9-0 for the first digit of the seconds.

1. Create new projects of HDL type.
2. Create a counter that counts from 9 to 0 and reset itself automatically when it becomes 0 than name it “mod10”.
3. Make the inputs: clk, reset, CE, LED.
4. Make the outputs: reg [3:0] counter, CEO
5. Assign (counter == 0) && CE to CEO
6. Start always block with posedge of the clk
7. Start condition statement with if LED is True then assign 0 to counter.
8. If not so check whether reset or CEO is True then assign 9 to counter.
9. If not so check if CE is True then decrement counter by 1.
10. End the module

Task2: Designing mod06:

This module is used to count from 5-0 for the second digit of the seconds.

1. Create new Verilog file named mod06.
2. Create another counter that counts from 5 to 0 and reset itself automatically when it becomes 5.
3. Make the inputs: clk, reset, CE, LED.
4. Make the outputs: reg [3:0] counter, CEO
5. Assign (counter == 0) && CE to CEO
6. Start always block with posedge of the clk
7. Start condition statement with if LED is True then assign 0 to counter.
8. If not so check whether reset or CEO is True then assign 5 to counter.
9. If not so, check if CE is True then decrement counter by 1.
10. End the module

Task3: Designing DownCounter:

This module is used to count from 9-0 for the two minute digits as well as loading the starting time inputs. If the inputs larger than 99 an LED error will turn on and set all digits zeros.

1. Create new Verilog file named BCD\_3D.
2. Make the inputs: clk, reset, load, CE, [3:0] data, LED.
3. Make the outputs: CEO, reg error, reg [3:0]count.
4. Assign (counter == 0) && CE to CEO.
5. Start always block with posedge of the clk
6. Start condition statement with if LED is True then assign 0 into count and error.
7. If not so, check if the input data exceeding 9 is True assign 1 to error and zero into count.
8. If not so, check whether reset or CEO is True then assign 9 to counter.
9. If not so, check if CE is True then decrement counter by 1.
10. End the module

Task4: Designing the Main Module Timer:

This module is used to gather the previous modules and instantiate them.

1. Create new Verilog file named timer.
2. Make the inputs: clk, reset, load, CE, [3:0] I1, I0.
3. Make the outputs: LED, ErrorLED, [7:0]seg, [3:0]an.
4. Assign S2 = (&(~I0))? ( |I1? 4'b1001 : 0) : (I0-1)
5. Assign S3 = (&(~I0))? (I1-1) : I1.
6. Assign ErrorLED = Error1 | Error2.
7. Assign STOP = LED | ErrorLED.
8. Instantiate the mod10 as s0 and pass in "clk, reset, CE, STOP, D0, CEO1".
9. Instantiate the mod06 as s1 and pass in " clk, reset, CE, STOP, D0, CEO2".
10. Instantiate the DownCounter as m0 and pass in "clk, reset, load, CEO2, S2, STOP, CEO2, Error1, D2"
11. Instantiate the DownCounter as m1 and pass in "clk, reset, load, CEO3, S3, STOP, CEO3, Error2, D3"
12. Instantiate DISP7SEG as ssd and pass in "clk, D0, D1, D2, D3, text\_mode, slow, med, fast, error, wrong, seg, an".
13. For uploading to a FPGA board, instantiate onehz as mHz and pass in "clk, reset, CE, counter, clk\_1hz"
14. End module.

**3.Problem Faced:**

No problem faced

**4.Work Distribution:**

|  |  |  |
| --- | --- | --- |
|  | Sami | Ammar |
| Task1 | 0% | 100% |
| Task2 | 20% | 80% |
| Task3 | 100% | 0% |
| Report | 45% | 55% |
| Questions | 40% | 60% |

**6.Conclusion:**

We managed to design a response circuit that uses the previous circuit to measure a person response time. We learnt about the saturated counters; that when they reached the maximus do not recount again. Also, we learnt to combine to Xilinx projects and extend one project with the other.