**CSEN 605 :Digital System Design **

**Assignment Report**

**Team Name : Supercalifragilisticexpialidocious**

|  |  |  |
| --- | --- | --- |
| **Member Name** | **Member ID** | **Tutorial #** |
| Sara Nagui | 34-6144 | T17 |
| Karim Kozman | 34-2267 | T17 |
| Nadeen Ghali | 34-1259 | T17 |
| Bassam Emad | 34-11232 | T17 |
| Mariam Rizk | 34-2226 | T20 |
| Mariam Sameh | 34-564 | T20 |

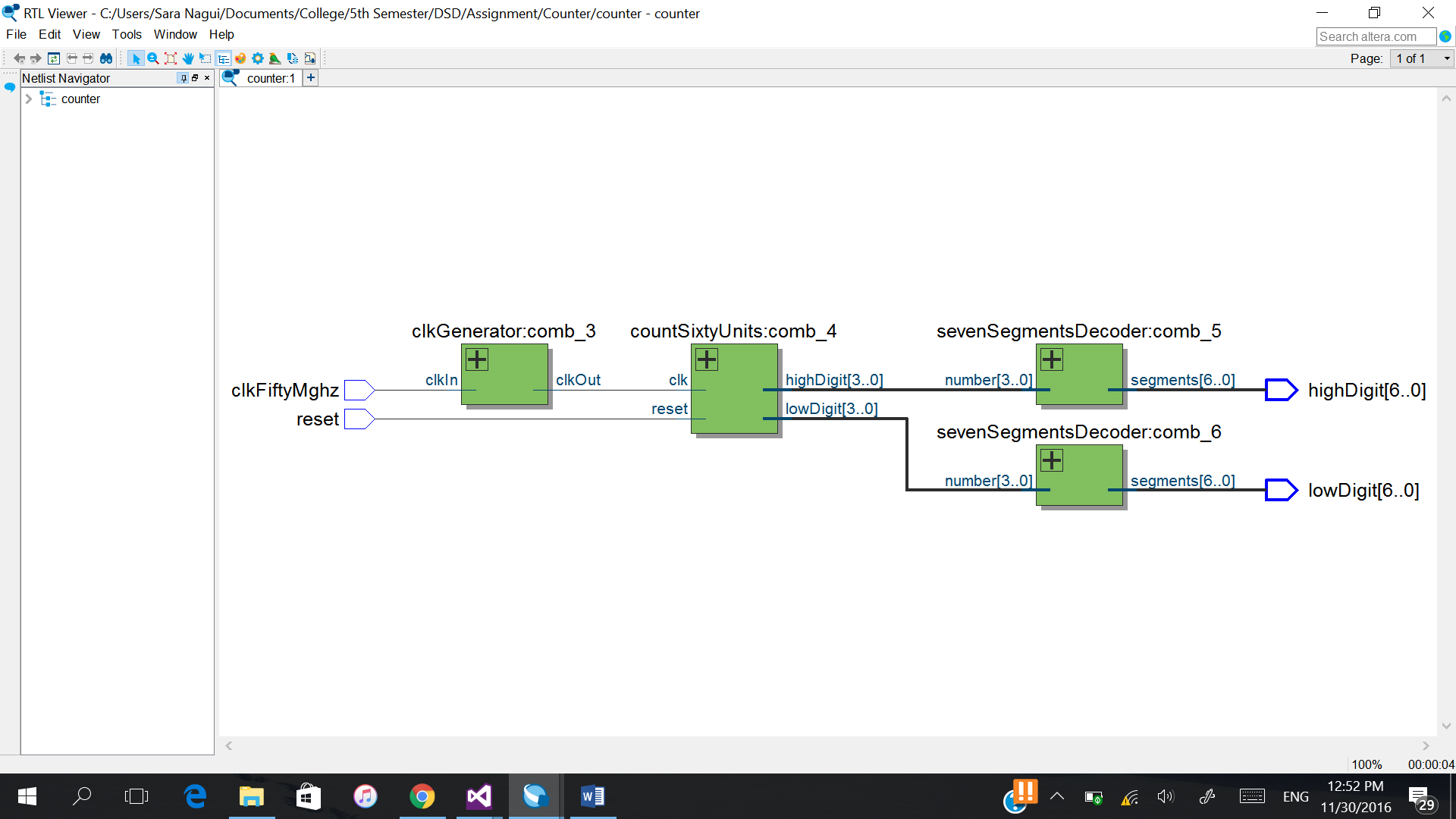
**Project Title : Counter**

**Submission Date :**

**11/30/2016**

**Design Methodology :**

**RTL Design :**



**Implementation:**

**Module Description:**

The main file which connects everything (**counter.v**) connects the other modules together. It takes the clock from the board which is 50MHz and generates its own clock using (clkGenerator comb\_3) then it takes this generated clock along with the reset signal from the board and connects it to (countSixtyUnits comb\_4) which counts from 0 to 59 then 0 and repeats and sets the output to 0 if the reset button is clicked. The two output digits are in binary so to connect them to the board we need to decode them into segments namely 7 segments as explained in the Altera Flow Example. That’s why the outputs highDigit and lowDigit are both connected to (sevenSegmentsDecoder comb\_5) and (sevenSegmentsDecoder comb\_6) respectively. The inputs of the whole circuit are the reset signal and the board clock and the outputs of the whole circuit are both digits the low and high digits each represented in 7 segments and connected to the pins according to the table.

**clkGenerator.v** takes 50MHz as input and switches the generated clock every 25MHz (using the counter that starts from 24 999999 and decrements each positive edge of the original clock till 0 then repeats) so that one whole cycle takes 50MHz.

**countSixtyUnits.v** is a counter that either resets or counts from 0 to 59 then 0 and repeats and represents the number in two digits each 4 bits (because the highest number in the lower digit is 9).

**sevenSegmentsDecoder.v** translates a number from 0-9 (so one digit) to seven segments using the help of the table of the altera board to display the actual number in decimal.

**Code Listing**

module counter(clkFiftyMghz,reset,highDigit,lowDigit);  
input wire clkFiftyMghz, reset;  
output wire [6:0] highDigit,lowDigit;  
wire clkHz;   
wire [3:0] cHigh, cLow;   
clkGenerator (clkFiftyMghz,clkHz);  
countSixtyUnits (clkHz,reset,cLow,cHigh);  
sevenSegmentsDecoder (cHigh,highDigit);  
sevenSegmentsDecoder (cLow,lowDigit);  
endmodule

module clkGenerator(clkIn,clkOut);  
input clkIn;  
output reg clkOut= 0;  
reg [24:0] counter = 0;  
always @(posedge clkIn)   
begin  
 if (counter == 0) begin  
 counter <= 24999999;  
 clkOut <= ~clkOut;  
 end else begin  
 counter <= counter -1;  
 end  
end  
endmodule

module countSixtyUnits(clk,reset,lowDigit,highDigit);  
input clk , reset;  
output reg [3:0] lowDigit = 0;  
output reg [3:0] highDigit = 0;  
always@(posedge clk , negedge reset)  
begin  
 if(~reset)  
 begin  
 lowDigit <= 0;  
 highDigit <= 0;  
 end  
 else  
 begin  
 if(lowDigit == 9)   
 begin  
 lowDigit <= 0;  
 if (highDigit == 5) highDigit <= 0;  
 else highDigit <= highDigit + 1;  
 end  
 else  
 begin  
 lowDigit <= lowDigit + 1;  
 end  
 end  
end  
endmodule

module sevenSegmentsDecoder(number,segments);   
input[3:0]number;  
output[6:0]segments;  
reg [6:0] segments;  
always@(number)   
begin  
 case (number)  
 0: segments<= ~7'b0111111;   
 1: segments<= ~7'b0000110;   
 2: segments<= ~7'b1011011;   
 3: segments <= ~7'b1001111;   
 4: segments<= ~7'b1100110;   
 5: segments <= ~7'b1101101;   
 6: segments<= ~7'b1111101;   
 7: segments <= ~7'b0000111;   
 8: segments<= ~7'b1111111;   
 9: segments <= ~7'b1101111;   
 default: segments <= 7'bx;  
 endcase

end  
endmodule