LAB 1

Tyler Korz

CMPEN 311 Sec: 002

Test Bench Code:

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 05/18/2020 12:27:00 PM

// Design Name:

// Module Name: testbench

// Project Name: 7 segment display

// Target Devices:

// Tool Versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

// Module : Testbench

// Description : Using the test sequence given generate the output of the 7 segment display by calling dff3 and counter modules

module testbench();

reg clrn\_tb;

reg clk\_tb;

reg u\_tb;

wire [2:0] q\_tb;

wire [2:0] ns\_tb;

wire a,b,c,d,e,f,g;

dff3 dff3\_tb(ns\_tb, clrn\_tb, clk\_tb, q\_tb);

counter counter\_tb(q\_tb, u\_tb, ns\_tb, a, b, c, d, e, f, g);

initial begin

clrn\_tb = 0;

clk\_tb = 1;

u\_tb = 1;

#1 clrn\_tb = 1;

#16 u\_tb = 0;

end

always begin

#1;

clk\_tb = ~clk\_tb;

end

endmodule

Design Code:

`timescale 1ns / 1ps

//////////////////////////////////////////////////////////////////////////////////

// Company:

// Engineer:

//

// Create Date: 05/18/2020 12:08:45 PM

// Design Name:

// Module(s) Name: dff3 and counter

// Project Name: 7 segment display

// Target Devices:

// Tool Versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

//////////////////////////////////////////////////////////////////////////////////

// Module : D flip-flop

// Description : At the positive edge of the clock, set the current state based on the clear

// Input(s) : 3-bit Next State, Clear and Clock

// Output(s) : 3-bit Current State

module dff3(input [2:0] ns, input clrn, input clk, output reg [2:0] q);

always @ (posedge clk)

begin

if (clrn == 1) begin

q <= ns;

end

else begin

q <= 3'b000;

end

end

endmodule

// Module : Up/Down Counter

// Description : Combinational circuit that generates the next state and LED control signals based on input u

// Input(s) : 3-bit Current State, 1-bit u

// Output(s) : 3-bit Next State, 7 1-bit segments of the LED

module counter(input [2:0] q, input u, output reg [2:0] ns,

output reg a, output reg b, output reg c, output reg d, output reg e, output reg f, output reg g );

always @(\*)

begin

if (u == 1) begin

if (q == 3'b101) begin

ns <= 3'b000;

end

else begin

ns <= q + 1;

end

end

else begin

if (q == 3'b000) begin

ns <= 3'b101;

end

else begin

ns <= q - 1;

end

end

case(q)

3'b000: begin

g=1'b1; f=1'b0; e=1'b0; d=1'b0; c=1'b0; b=1'b0; a=1'b0;

end

3'b001: begin

g=1'b1; f=1'b1; e=1'b1; d=1'b1; c=1'b0; b=1'b0; a=1'b1;

end

3'b010: begin

g=1'b0; f=1'b1; e=1'b0; d=1'b0; c=1'b1 ;b=1'b0; a=1'b0;

end

3'b011: begin

g=1'b0; f=1'b1; e=1'b1; d=1'b0; c=1'b0; b=1'b0; a=1'b0;

end

3'b100: begin

g=1'b0; f=1'b0; e=1'b1; d=1'b1; c=1'b0; b=1'b0; a=1'b1;

end

3'b101: begin

g=1'b0; f=1'b0; e=1'b1; d=1'b0; c=1'b0; b=1'b1; a=1'b0;

end

endcase

end

endmodule

Simulation: A screenshot of a computer

Description automatically generated

Schematic:

A computer screen shot of a computer

Description automatically generated

I/O Planning:

A screenshot of a computer game

Description automatically generated

Floorplanning:

A screenshot of a computer

Description automatically generated

Implemented Schematic:

A computer screen shot of a computer

Description automatically generated