`timescale 1ns / 1ps

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

// Company:

// Engineer:

//

// Create Date: 12:01:04 05/10/2015

// Design Name:

// Module Name: Main

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

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

module Main(

//inputs

input clk,

input time\_set,

input inc\_hr,

input inc\_min,

input rst,

input enable,

//outputs

output [6:0] outsegh1,outsegh2,

output [6:0] outsegm1,outsegm2,

output [6:0] outsegs1,outsegs2,

output [6:0] Q,

output [3:0] Four\_Wheels

);

//wires

wire [32:0]cnt = 0;

wire clk\_1hz, clk\_100hz;

wire [5:0] outh,outm,outs;

wire [3:0] hr1,hr2,min1,min2,sec1,sec2;

wire [6:0] d1,d2,d3,d4;

wire [1:0] select;

//clock Div

ClockDiviper CD(.clk(clk),.clk\_1hz(clk\_1hz));

//clock TheAbaza

ClockTheAbaza AP(.clk(clk),.clk\_100hz(clk\_100hz));

//clock/Adjust

ClockAdjust CA(.rst(rst), .clk\_1hz(clk\_1hz), .outh(outh),.outm(outm),.outs(outs),.time\_set(time\_set),.inc\_hr(inc\_hr),.inc\_min(inc\_min));

//Hours output

BCD bcd1 (.bin(outh),.bcd1(hr2),.bcd0(hr1));

bcd27seg bcd27seg1 (.bcd(hr1),.seg7(d3));

bcd27seg bcd27seg2 (.bcd(hr2),.seg7(d4));

//Minute Output

BCD bcd2 (.bin(outm),.bcd1(min2),.bcd0(min1));

bcd27seg bcd27seg3 (.bcd(min1),.seg7(d1));

bcd27seg bcd27seg4 (.bcd(min2),.seg7(d2));

//Second Output

BCD bcd3 (.bin(outs),.bcd1(sec2),.bcd0(sec1));

bcd27seg bcd27seg5 (.bcd(sec1),.seg7(outsegs1));

bcd27seg bcd27seg6 (.bcd(sec2),.seg7(outsegs2));

//MUX

Mux4 M(.select(select),.d1(d1),.d2(d2),.d3(d3),.d4(d4),.q(Q));

//Binary Counter

twoBit Binco(.clk(clk\_100hz),.rst(rst),.O(select));

//Binar Decoder

Dec D( .binary\_in(select),.decoder\_out(Four\_Wheels) ,.enable(enable));

Endmodule

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

`timescale 1ns / 1ps

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

// Company:

// Engineer:

//

// Create Date: 12:02:00 05/10/2015

// Design Name:

// Module Name: ClockDiviper

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

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

module ClockDiviper(clk,clk\_1hz);

input clk;

output reg clk\_1hz;

reg [32:0]cnt;

always@(posedge clk)

begin

if(cnt==24999999)

begin

clk\_1hz<=~clk\_1hz;

cnt<=0;

end

else

cnt<=cnt+1;

end

endmodule

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

`timescale 1ns / 1ps

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

// Company:

// Engineer:

//

// Create Date: 17:22:43 05/10/2015

// Design Name:

// Module Name: ClockTheAbaza

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

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

module ClockTheAbaza(clk,clk\_100hz);

input clk;

output reg clk\_100hz;

reg [32:0]cnt;

always@(posedge clk)

begin

if(cnt==4999)

begin

clk\_100hz<=~clk\_100hz;

cnt<=0;

end

else

cnt<=cnt+1;

end

endmodule

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

`timescale 1ns / 1ps

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

// Company:

// Engineer:

//

// Create Date: 12:06:33 05/10/2015

// Design Name:

// Module Name: ClockAdjust

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

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

module ClockAdjust(rst, clk\_1hz, outh,outm,outs,time\_set,inc\_hr,inc\_min);

input rst, clk\_1hz, time\_set,inc\_hr,inc\_min;

output reg [5:0] outs,outm,outh;

always@(posedge clk\_1hz,posedge rst)

begin

if(rst==1)

begin

outh<= 0;

outm<= 0;

outs<= 0;

end

else

begin

if(time\_set == 0)// Run Clock

begin

if(outs!=6'd59)

begin

outs<=outs+1;

end

if(outs==6'd59)

begin

outs<= 0;

outm<=outm+1;

end

if(outm==6'd59)

begin

outm<= 0;

outh<=outh+1;

end

if(outh==6'd23)

outh<= 0;

end

else if(time\_set==1)// Edit Clock

begin

if(inc\_hr==1)

begin

if(outh==6'd23)

outh<= 0;

else

outh<=outh+1;

end

if(inc\_min==1)

begin

if(outm==6'd59)

outm<= 0;

else

outm<=outm+1;

end

end

end

end

endmodule

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

`timescale 1ns / 1ps

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

// Company:

// Engineer:

//

// Create Date: 12:15:28 05/10/2015

// Design Name:

// Module Name: BCD

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

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

module BCD(bin, bcd1,bcd0);

input [5:0] bin;

output reg [3:0] bcd1,bcd0;

always@\*

case (bin)

6'd0 : begin bcd1 <= 4'b0000; bcd0 <= 4'b0000; end

6'd1 : begin bcd1 <= 4'b0000; bcd0 <= 4'b0001; end

6'd2 : begin bcd1 <= 4'b0000; bcd0 <= 4'b0010; end

6'd3 : begin bcd1 <= 4'b0000; bcd0 <= 4'b0011; end

6'd4 : begin bcd1 <= 4'b0000; bcd0 <= 4'b0100; end

6'd5 : begin bcd1 <= 4'b0000; bcd0 <= 4'b0101; end

6'd6 : begin bcd1 <= 4'b0000; bcd0 <= 4'b0110; end

6'd7 : begin bcd1 <= 4'b0000; bcd0 <= 4'b0111; end

6'd8 : begin bcd1 <= 4'b0000; bcd0 <= 4'b1000; end

6'd9 : begin bcd1 <= 4'b0000; bcd0 <= 4'b1001; end

6'd10 : begin bcd1 <= 4'b0001; bcd0 <= 4'b0000; end

6'd11 : begin bcd1 <= 4'b0001; bcd0 <= 4'b0001; end

6'd12 : begin bcd1 <= 4'b0001; bcd0 <= 4'b0010; end

6'd13 : begin bcd1 <= 4'b0001; bcd0 <= 4'b0011; end

6'd14 : begin bcd1 <= 4'b0001; bcd0 <= 4'b0100; end

6'd15 : begin bcd1 <= 4'b0001; bcd0 <= 4'b0101; end

6'd16 : begin bcd1 <= 4'b0001; bcd0 <= 4'b0110; end

6'd17 : begin bcd1 <= 4'b0001; bcd0 <= 4'b0111; end

6'd18 : begin bcd1 <= 4'b0001; bcd0 <= 4'b1000; end

6'd19 : begin bcd1 <= 4'b0001; bcd0 <= 4'b1001; end

6'd20 : begin bcd1 <= 4'b0010; bcd0 <= 4'b0000; end

6'd21 : begin bcd1 <= 4'b0010; bcd0 <= 4'b0001; end

6'd22 : begin bcd1 <= 4'b0010; bcd0 <= 4'b0010; end

6'd23 : begin bcd1 <= 4'b0010; bcd0 <= 4'b0011; end

6'd24 : begin bcd1 <= 4'b0010; bcd0 <= 4'b0100; end

6'd25 : begin bcd1 <= 4'b0010; bcd0 <= 4'b0101; end

6'd26 : begin bcd1 <= 4'b0010; bcd0 <= 4'b0110; end

6'd27 : begin bcd1 <= 4'b0010; bcd0 <= 4'b0111; end

6'd28 : begin bcd1 <= 4'b0010; bcd0 <= 4'b1000; end

6'd29 : begin bcd1 <= 4'b0010; bcd0 <= 4'b1001; end

6'd30 : begin bcd1 <= 4'b0011; bcd0 <= 4'b0000; end

6'd31 : begin bcd1 <= 4'b0011; bcd0 <= 4'b0001; end

6'd32 : begin bcd1 <= 4'b0011; bcd0 <= 4'b0010; end

6'd33 : begin bcd1 <= 4'b0011; bcd0 <= 4'b0011; end

6'd34 : begin bcd1 <= 4'b0011; bcd0 <= 4'b0100; end

6'd35 : begin bcd1 <= 4'b0011; bcd0 <= 4'b0101; end

6'd36 : begin bcd1 <= 4'b0011; bcd0 <= 4'b0110; end

6'd37 : begin bcd1 <= 4'b0011; bcd0 <= 4'b0111; end

6'd38 : begin bcd1 <= 4'b0011; bcd0 <= 4'b1000; end

6'd39 : begin bcd1 <= 4'b0011; bcd0 <= 4'b1001; end

6'd40 : begin bcd1 <= 4'b0100; bcd0 <= 4'b0000; end

6'd41 : begin bcd1 <= 4'b0100; bcd0 <= 4'b0001; end

6'd42 : begin bcd1 <= 4'b0100; bcd0 <= 4'b0010; end

6'd43 : begin bcd1 <= 4'b0100; bcd0 <= 4'b0011; end

6'd44 : begin bcd1 <= 4'b0100; bcd0 <= 4'b0100; end

6'd45 : begin bcd1 <= 4'b0100; bcd0 <= 4'b0101; end

6'd46 : begin bcd1 <= 4'b0100; bcd0 <= 4'b0110; end

6'd47 : begin bcd1 <= 4'b0100; bcd0 <= 4'b0111; end

6'd48 : begin bcd1 <= 4'b0100; bcd0 <= 4'b1000; end

6'd49 : begin bcd1 <= 4'b0100; bcd0 <= 4'b1001; end

6'd50 : begin bcd1 <= 4'b0101; bcd0 <= 4'b0000; end

6'd51 : begin bcd1 <= 4'b0101; bcd0 <= 4'b0001; end

6'd52 : begin bcd1 <= 4'b0101; bcd0 <= 4'b0010; end

6'd53 : begin bcd1 <= 4'b0101; bcd0 <= 4'b0011; end

6'd54 : begin bcd1 <= 4'b0101; bcd0 <= 4'b0100; end

6'd55 : begin bcd1 <= 4'b0101; bcd0 <= 4'b0101; end

6'd56 : begin bcd1 <= 4'b0101; bcd0 <= 4'b0110; end

6'd57 : begin bcd1 <= 4'b0101; bcd0 <= 4'b0111; end

6'd58 : begin bcd1 <= 4'b0101; bcd0 <= 4'b1000; end

6'd59 : begin bcd1 <= 4'b0101; bcd0 <= 4'b1001; end

6'd60 : begin bcd1 <= 4'b0110; bcd0 <= 4'b0000; end

endcase

endmodule

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

`timescale 1ns / 1ps

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

// Company:

// Engineer:

//

// Create Date: 12:17:24 05/10/2015

// Design Name:

// Module Name: bcd27seg

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

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

module bcd27seg(bcd,seg7);

input [3:0] bcd;

output reg [6:0] seg7;

always@\*

case(bcd)

5'h00: seg7 <= 7'b1000000;

5'h01: seg7 <= 7'b1111001;

5'h02: seg7 <= 7'b0100100;

5'h03: seg7 <= 7'b0110000;

5'h04: seg7 <= 7'b0011001;

5'h05: seg7 <= 7'b0010010;

5'h06: seg7 <= 7'b0000010;

5'h07: seg7 <= 7'b1111000;

5'h08: seg7 <= 7'b0000000;

5'h09: seg7 <= 7'b0010000;

endcase

endmodule

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

`timescale 1ns / 1ps

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

// Company:

// Engineer:

//

// Create Date: 12:57:17 04/16/2015

// Design Name:

// Module Name: Mux4

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

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

module Mux4(

select,

d1,

d2,

d3,

d4,

q

);

input[1:0] select;

input[6:0] d1,d2,d3,d4;

output reg [6:0] q;

always @(select)

begin

case (select)

2'b00 : q = d1;

2'b01 : q = d2;

2'b10 : q = d3;

2'b11 : q = d4;

endcase

end

endmodule

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

`timescale 1ns / 1ps

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

// Company:

// Engineer:

//

// Create Date: 12:54:00 04/16/2015

// Design Name:

// Module Name: twoBit

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

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

module twoBit(

clk,

rst,

O

);

input clk,rst;

output reg[1:0] O;

always @(posedge clk)

begin

if(rst == 0)

begin

O=O+1;

end

if(rst == 1)

O = 2'b00;

end

endmodule

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

`timescale 1ns / 1ps

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

// Company:

// Engineer:

//

// Create Date: 12:14:36 04/16/2015

// Design Name:

// Module Name: Dec

// Project Name:

// Target Devices:

// Tool versions:

// Description:

//

// Dependencies:

//

// Revision:

// Revision 0.01 - File Created

// Additional Comments:

//

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

module Dec (

binary\_in , // 2 bit binary input

decoder\_out , // 4-bit out

enable // Enable for the decoder

);

input [1:0] binary\_in ;

input enable ;

output [3:0] decoder\_out ;

reg [3:0] decoder\_out ;

always @ (enable or binary\_in)

begin

decoder\_out = 0;

if (enable) begin

case (binary\_in)

2'b00 : decoder\_out = 4'b1110;

2'b01 : decoder\_out = 4'b1101;

2'b10 : decoder\_out = 4'b1011;

2'b11 : decoder\_out = 4'b0111;

endcase

end

end

endmodule

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

NET "Q[0]" LOC = L14 ;

NET "Q[1]" LOC = H12 ;

NET "Q[2]" LOC = N14 ;

NET "Q[3]" LOC = N11 ;

NET "Q[4]" LOC = P12 ;

NET "Q[5]" LOC = L13 ;

NET "Q[6]" LOC = M12 ;

NET "clk" LOC = B8 ;

NET "rst" LOC = P11;

NET "time\_set" LOC = L3 ;

NET "inc\_hr" LOC = K3 ;

NET "inc\_min" LOC = B4 ;

NET "enable" LOC = G3 ;

NET "Four\_Wheels[0]" LOC = F12;

NET "Four\_Wheels[1]" LOC = J12;

NET "Four\_Wheels[2]" LOC = M13;

NET "Four\_Wheels[3]" LOC = K14;