**PARTE 2**

module parte2 (SW, LEDG, LEDR);

input [8:0] SW;

output [8:0] LEDR;

output [8] LEDG;

assign LEDR[8:0] = SW[8:0];

wire c1, c2, c3;

somador\_completo A0 (SW[0], SW[4], SW[8], LEDG[0], c1);

somador\_completo A1 (SW[1], SW[5], c1, LEDG[1], c2);

somador\_completo A2 (SW[2], SW[6], c2, LEDG[2], c3);

somador\_completo A3 (SW[3], SW[7], c3, LEDG[3], LEDG[4]);

endmodule

module somador\_completo (a, b, ci, s, co);

input a, b, ci;

output co, s;

wire d;

assign d = a ^ b;

assign s = d ^ ci;

assign co = (b & ~d) | (d & ci);

endmodule

**PARTE 3**

module parte3 (SW, HEX0, HEX1);

input [3:0] SW;

output [0:6] HEX0, HEX1;

wire z;

wire [3:0] M, A;

assign A[3] = 0;

compara C0 (SW[3:0], z);

circuitoA AA (SW[2:0], A[2:0]);

mux\_4bit\_2to1 MU (z, SW[3:0], A, M);

circuitoB BB (z, HEX1);

char\_7seg S0 (M, HEX0);

endmodule

module char\_7seg (C, DISPLAY);

input [3:0] C;

output [0:6] DISPLAY;

assign DISPLAY[0] = ((~X[3] & ~X[2] & ~X[1] & X[0]) | (~X[3] & X[2] & ~X[1] & ~X[0]));

assign DISPLAY[1] = ((~X[3] & X[2] & ~X[1] & X[0]) | (~X[3] & X[2] & X[1] & ~X[0]));

assign DISPLAY[2] = (~X[3] & ~X[2] & X[1] & ~X[0]);

assign DISPLAY[3] = ((~X[3] & ~X[2] & ~X[1] & X[0]) | (~X[3] & X[2] & ~X[1] & ~X[0]) | (~X[3] & X[2] & X[1] & X[0]) | (X[3] & ~X[2] & ~X[1] & X[0]));

assign DISPLAY[4] = ~((~X[2] & ~X[0]) | (X[1] & ~X[0]));

assign DISPLAY[5] = ((~X[3] & ~X[2] & ~X[1] & X[0]) | (~X[3] & ~X[2] & X[1] & ~X[0]) | (~X[3] & ~X[2] & X[1] & X[0]) | (~X[3] & X[2] & X[1] & X[0]));

assign DISPLAY[6] = ((~X[3] & ~X[2] & ~X[1] & X[0]) | (~X[3] & ~X[2] & ~X[1] & ~X[0]) | (~X[3] & X[2] & X[1] & X[0]));

endmodule

module compara (VV, z);

input [3:0] VV;

output z;

assign z = ((VV[2] | VV[1]) & VV[3]);

endmodule

module circuitoA (V, A);

input [2:0] V;

output [2:0] A;

assign A[0] = V[0];

assign A[1] = ~V[1];

assign A[2] = (V[2] & V[1]);

endmodule

module circuitoB (z, DISPLAY);

input z;

output [0:6] DISPLAY;

assign DISPLAY[0] = z;

assign DISPLAY[1:2] = 2'b00;

assign DISPLAY[3:5] = {3{z}};

assign DISPLAY[6] = 1;

endmodule

module mux\_4bit\_2to1 (s, U, V, M);

input s;

input [3:0] U, V;

output [3:0] M;

assign M = ({4{~s}} & U) | ({4{s}} & V);

endmodule

**PARTE 4**

module parte4(SW, LEDR, LEDG, HEX0, HEX1);

input [7:0] SW;

output [7:0] LEDR;

output [8] LEDG;

output [6:0] HEX1, HEX0;

wire cmp1, cmp2;

wire c1, c2, c3;

wire [3:0] S;

compara1 CMP0 (SW[3:0], cmp1)

compara1 CMP01 (SW[7:4], cmp2)

assign LEDR[7:0] = SW[7:0]

assign LEDG[8] = (cmp1 | cmp2)

somador\_completo A0 (SW[0], SW[4], SW[8], S[0], c1);

somador\_completo A1 (SW[1], SW[5], c1, S[1], c2);

somador\_completo A2 (SW[2], SW[6], c2, S[2], c3);

somador\_completo A3 (SW[3], SW[7], c3, S[3], S[4]);

comparao2 CMP002 (S, z);

circuitoA AA (S[3:0], A);

circuitoB BB (z, HEX1);

mux\_4bit\_2to1 MU (z, S[3:0], A, M);

char\_7seg(M, HEX0);

module mux\_4bit\_2to1 (s, U, V, M);

input s;

input [3:0] U, V;

output [3:0] M;

assign M = ({4{~s}} & U) | ({4{s}} & V);

endmodule

module circuitoA (V, A);

input [2:0] V;

output [2:0] A;

assign A[0] = V[0];

assign A[1] = ~V[1];

assign A[2] = (V[2] & V[1]);

endmodule

module circuitoB (z, DISPLAY);

input z;

output [0:6] DISPLAY;

assign DISPLAY[0] = z;

assign DISPLAY[1:2] = 2'b00;

assign DISPLAY[3:5] = {3{z}};

assign DISPLAY[6] = 1;

endmodule

module compara1 (V, z);

input [3:0] V;

output z;

assign z = (V[3] & (V[2] | V[1]));

endmodule

module char\_7seg (C, DISPLAY);

input [3:0] C;

output [0:6] DISPLAY;

assign DISPLAY[0] = ((~X[3] & ~X[2] & ~X[1] & X[0]) | (~X[3] & X[2] & ~X[1] & ~X[0]));

assign DISPLAY[1] = ((~X[3] & X[2] & ~X[1] & X[0]) | (~X[3] & X[2] & X[1] & ~X[0]));

assign DISPLAY[2] = (~X[3] & ~X[2] & X[1] & ~X[0]);

assign DISPLAY[3] = ((~X[3] & ~X[2] & ~X[1] & X[0]) | (~X[3] & X[2] & ~X[1] & ~X[0]) | (~X[3] & X[2] & X[1] & X[0]) | (X[3] & ~X[2] & ~X[1] & X[0]));

assign DISPLAY[4] = ~((~X[2] & ~X[0]) | (X[1] & ~X[0]));

assign DISPLAY[5] = ((~X[3] & ~X[2] & ~X[1] & X[0]) | (~X[3] & ~X[2] & X[1] & ~X[0]) | (~X[3] & ~X[2] & X[1] & X[0]) | (~X[3] & X[2] & X[1] & X[0]));

assign DISPLAY[6] = ((~X[3] & ~X[2] & ~X[1] & X[0]) | (~X[3] & ~X[2] & ~X[1] & ~X[0]) | (~X[3] & X[2] & X[1] & X[0]));

endmodule

module compara2 (V, z);

input [4:0] V;

output z;

assign z = V[4] | ((V[3] & V[2]) | (V[3] & V[1]));

endmodule

module somador\_completo(a, b, ci, s, co);

input a, b, ci;

output co, s;

wire d;

assign d = a ^ b;

assign s = d ^ ci;

assign co = (b & ~d) | (d & ci);

endmodule

**PARTE 5**

parte5 (SW, LEDR, HEX0, HEX1, HEX2);

input [15:0]SW;

output [15:0] LEDR;

output [6:0] HEX2, HEX1, HEX0;

assign LEDR[15:0] = SW[15:0]

wire [4:0] T1, T0;

wire [3:0] Z1, Z0, S2, S1, S0;

always begin

T0 = SW[3:0] + SW[11:8];

if (T0 > 9) begin

Z0 = 10;

c1 = 1;

end else begin

Z0 = 0;

c1 = 0;

end

S0 = T0 - Z0;

T1 = SW[7:4] + SW[15:12] + c1;

if (T1 > 9) begin

Z1 = 10;

c2 = 1;

end else begin

Z1 = 0;

c2 = 0;

end

S1 = T1 - Z1;

S2 = c2;

end

endmodule