**//主模块, 顶层模块**

module lcd\_driver(

input CLOCK\_50, // 50 MHz clock

input [17:0] SW, // Toggle Switch[17:0]

input K0\_clk, //Control Display

output LCD\_ON, // LCD Power ON/OFF

output LCD\_BLON, // LCD Back Light ON/OFF

output LCD\_RW, // LCD Read/Write Select, 0 = Write, 1 = Read

output LCD\_EN, // LCD Enable

output LCD\_RS, // LCD Command/Data Select, 0 = Command, 1 = Data

inout [7:0] LCD\_DATA // LCD Data bus 8 bits

);

// reset delay gives some time for peripherals to initialize

wire DLY\_RST;

Reset\_Delay r0( .iCLK(CLOCK\_50),.oRESET(DLY\_RST) );

// turn LCD ON

assign LCD\_ON = 1'b1;

assign LCD\_BLON = 1'b1;

//Store SW

wire [3:0] hex3,hex2,hex1,hex0;

assign hex3 = SW[17:14];

assign hex2 = SW[13:10];

assign hex1 = SW[7:4];

assign hex0 = SW[3:0];

LCD\_Display u1(

.iCLK\_50MHZ(CLOCK\_50),

.iRST\_N(DLY\_RST),

.K0\_clk(K0\_clk),

.hex0(hex0),

.hex1(hex1),

.hex2(hex2),

.hex3(hex3),

.DATA\_BUS(LCD\_DATA),

.LCD\_RW(LCD\_RW),

.LCD\_E(LCD\_EN),

.LCD\_RS(LCD\_RS)

);

Endmodule

**//扩展功能以及打印值转换模块**

module LCD\_display\_string(index,out,K0\_clk, hex3,hex2,hex0,hex1);

input K0\_clk;

input [4:0] index;

input [3:0] hex3,hex2,hex1,hex0;

output [7:0] out;

reg [7:0] out;

reg [7:0] dec;

reg [11:0] b;

reg [3:0] count=4'd0;

reg a;

reg [7:0] regs[15:0];

initial

begin

regs[15]= 8'h20;

regs[14]= 8'h20;

regs[13]= 8'h20;

regs[12]= 8'h20;

regs[11]= 8'h20;

regs[10]= 8'h20;

regs[9]= 8'h20;

regs[8]= 8'h20;

regs[7]= 8'h20;

regs[6]= 8'h20;

regs[5]= 8'h20;

regs[4]= 8'h20;

regs[3]= 8'h20;

regs[2]= 8'h20;

regs[1]= 8'h20;

regs[0]= 8'h20;

end

always @(negedge K0\_clk)

if(count == 16)

count <= 1'b0;

else if(!K0\_clk)

begin

regs[count] <= {hex3,hex2};

count<=count+1'b1;

end

// Line 1

always

case (index)

5'h00: out <= 8'h41;

5'h01: out <= 8'h53;

5'h02: out <= 8'h43;

5'h03: out <= 8'h49;

5'h04: out <= 8'h49;

5'h05: out <= 8'h3a;

5'h06: out <= 8'h30;

5'h07: out <= 8'h78;

5'h08: out <= {4'h0,hex1};

5'h09: out <= {4'h0,hex0};

5'h0b: out <= {hex1,hex0};

5'h0d:

begin

dec<={hex1,hex0};

b[11:8]=(dec/100)%10;

out <= {4'h0,b[11:8]};

end

5'h0e:

begin

dec<={hex1,hex0};

b[7:4]=(dec/10)%10;

out <= {4'h0,b[7:4]};

end

5'h0f:

begin

dec<={hex1,hex0};

b[3:0]=dec%10;

out <= {4'h0,b[3:0]};

end

// Line 2

5'h10: out <= regs[0];

5'h11: out <= regs[1];

5'h12: out <= regs[2];

5'h13: out <= regs[3];

5'h14: out <= regs[4];

5'h15: out <= regs[5];

5'h16: out <= regs[6];

5'h17: out <= regs[7];

5'h18: out <= regs[8];

5'h19: out <= regs[9];

5'h1a: out <= regs[10];

5'h1b: out <= regs[11];

5'h1c: out <= regs[12];

5'h1d: out <= regs[13];

5'h1e: out <= regs[14];

5'h1f: out <= regs[15];

default: out <= 8'h20;

endcase

endmodule

**//延时模块**

module Reset\_Delay(iCLK,oRESET);

input iCLK;

output reg oRESET;

reg [19:0] Cont;

always@(posedge iCLK)

begin

if(Cont!=20'hFFFFF)

begin

Cont <= Cont+1'b1;

oRESET <= 1'b0;

end

else

oRESET <= 1'b1;

end

endmodule