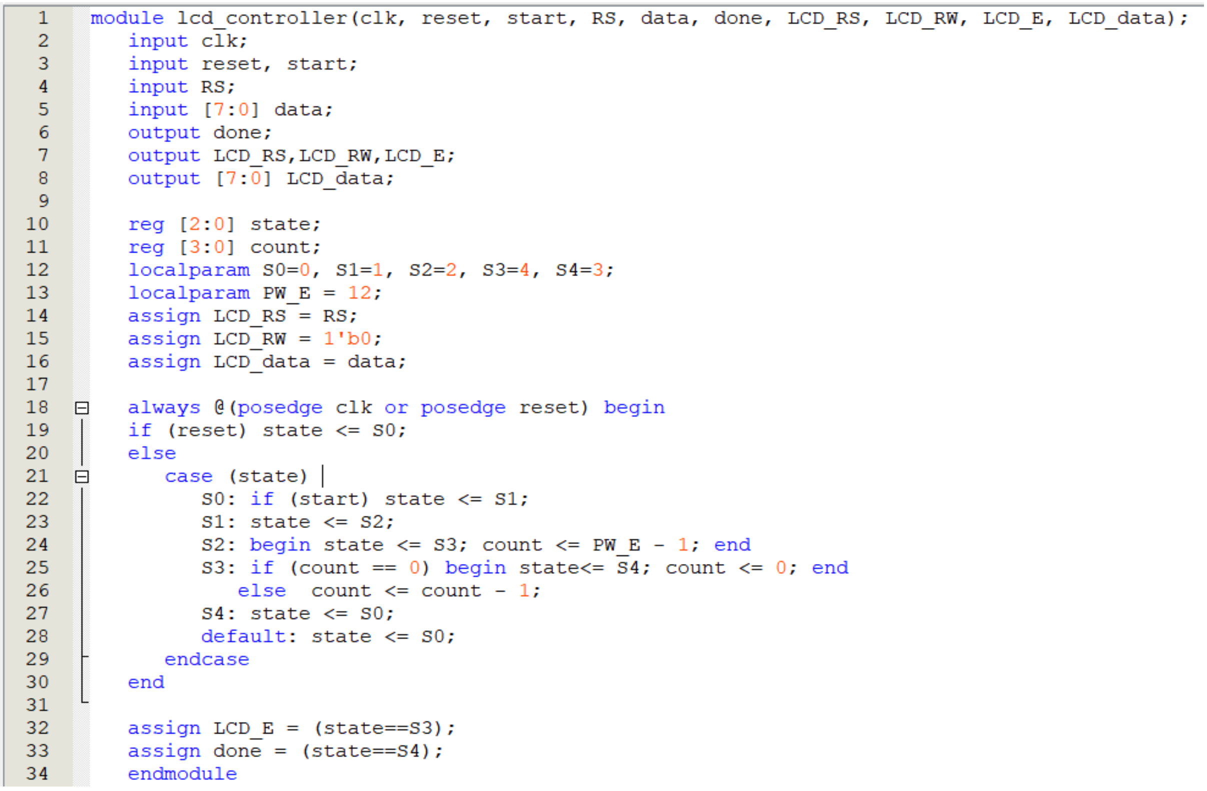
# < 실습과제 9 >

2017253041 홍성우

1.

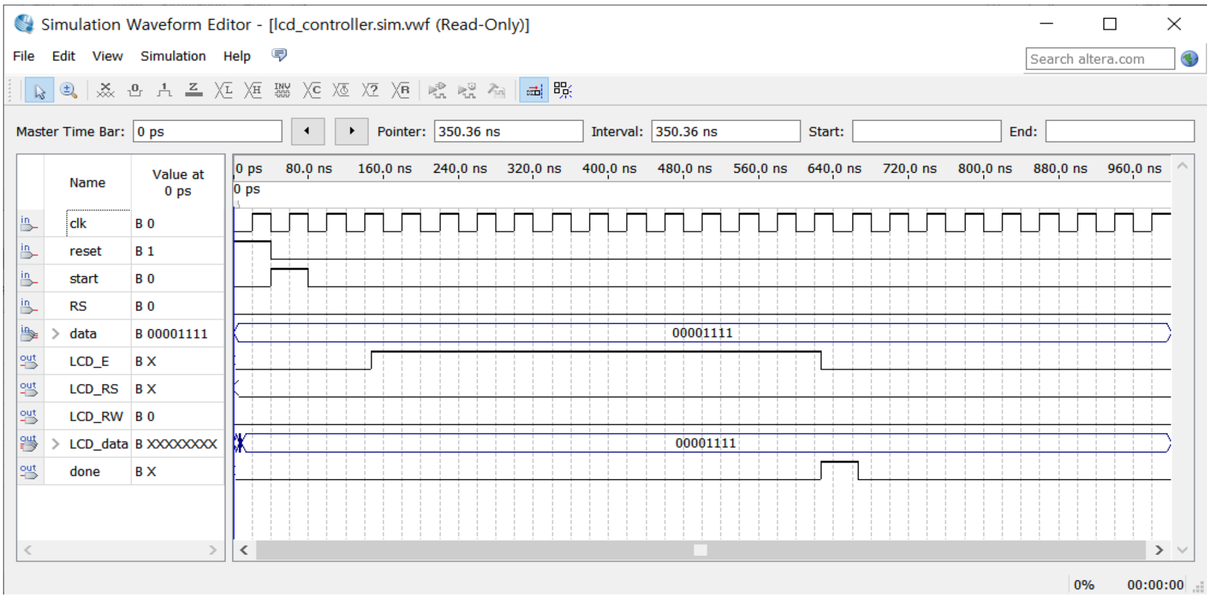
(1)



```
module lcd_controller(clk, reset, start, RS, data, done, LCD_RS, LCD_RW, LCD_E, LCD_data);
   input clk;
   input reset, start;
   input RS;
   input [7:0] data;
   output done;
   output LCD_RS,LCD_RW,LCD_E;
   output [7:0] LCD_data;

   reg [2:0] state;
   reg [3:0] count;
   localparam S0=0, S1=1, S2=2, S3=4, S4=3;
   localparam PW_E = 12;
   assign LCD_RS = RS;
   assign LCD_RW = 1'b0;
   assign LCD_data = data;

   always @(posedge clk or posedge reset) begin
   if (reset) state <= S0;
   else
      case (state)
         S0: if (start) state <= S1;
         S1: state <= S2;
         S2: begin state <= S3; count <= PW_E - 1; end
         S3: if (count == 0) begin state<= S4; count <= 0; end
            else  count <= count - 1;
         S4: state <= S0;
         default: state <= S0;
      endcase
   end

   assign LCD_E = (state==S3);
   assign done = (state==S4);
   endmodule
```

-FPGA는 CPU가 없으므로 LCD controller 모듈이 제어신호로 하드웨어처럼 작동한다.

(2)



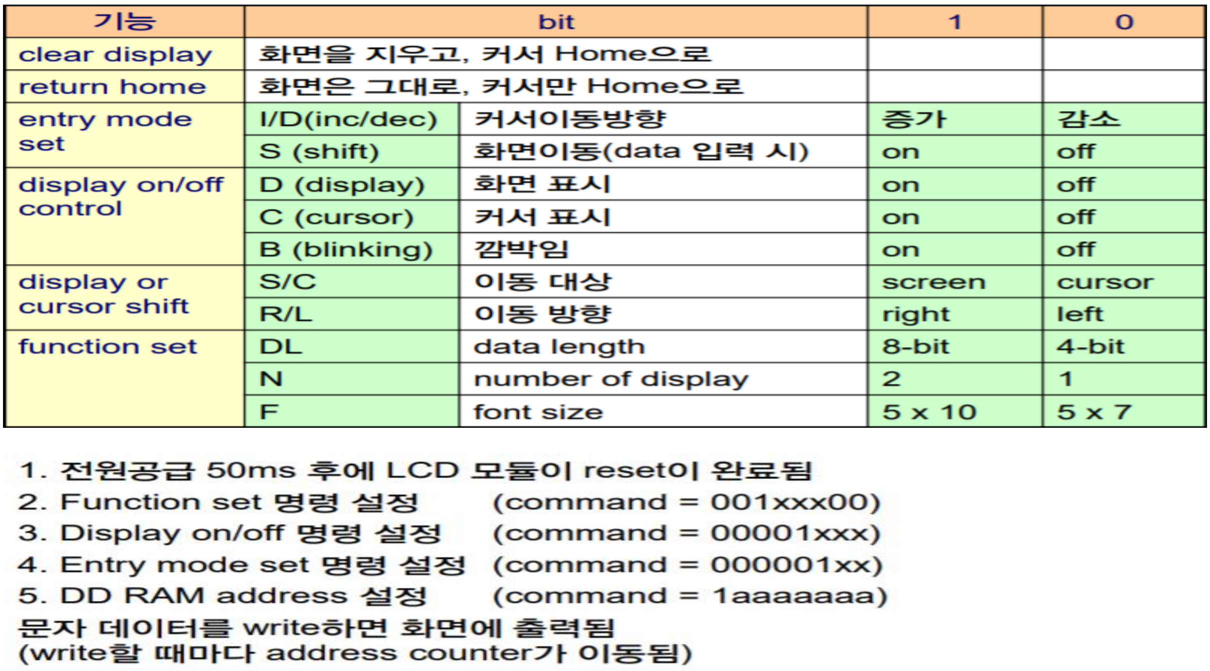
Simulation Waveform Editor - [lcd_controller.sim.vwf (Read-Only)]

Master Time Bar: 0 ps  Pointer: 350.36 ns  Interval: 350.36 ns  Start:  End:

| Name | Value at 0 ps |
|---|---|
| clk | B 0 |
| reset | B 1 |
| start | B 0 |
| RS | B 0 |
| data | B 00001111 |
| LCD_E | B X |
| LCD_RS | B X |
| LCD_RW | B 0 |
| LCD_data | B XXXXXXXX |
| done | B X |

2.

(1)

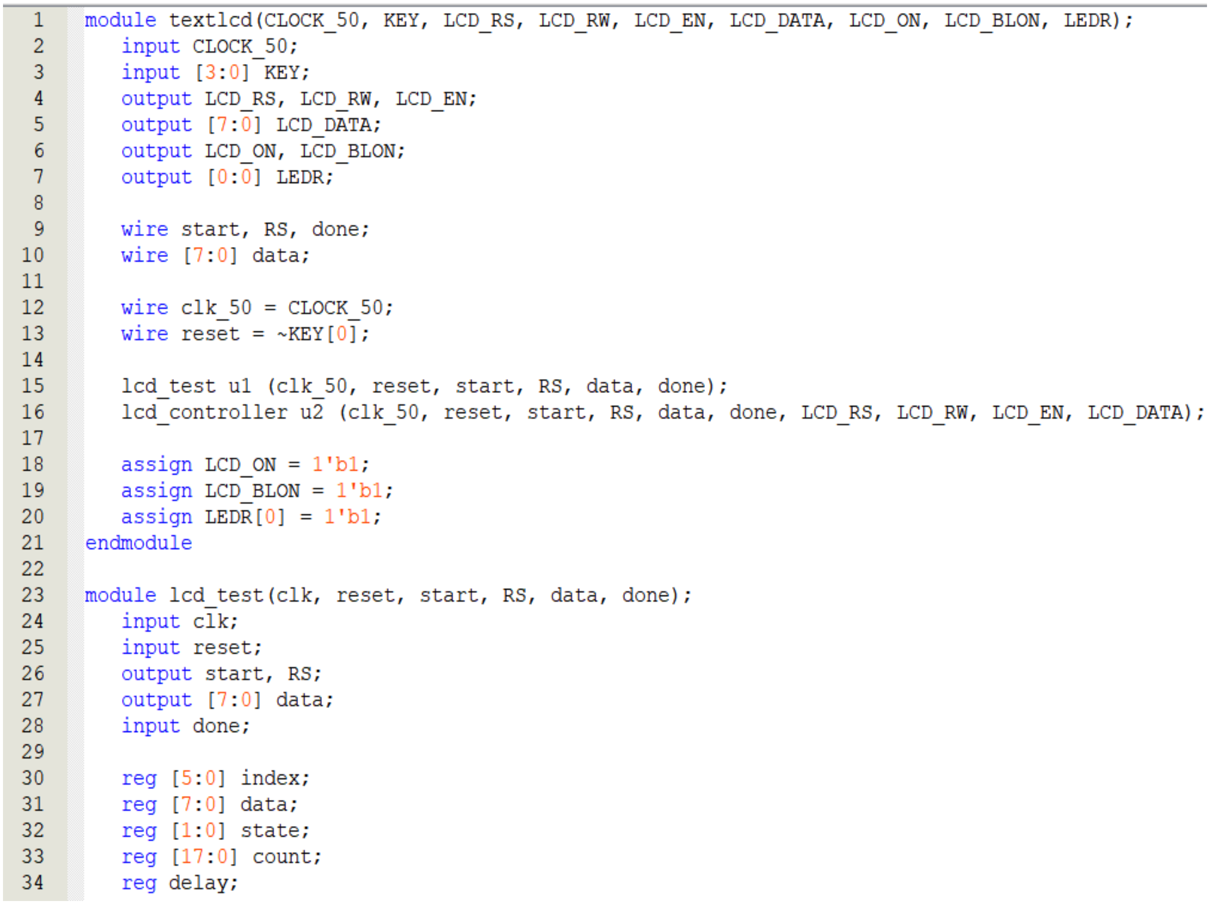


| 기능 | bit | | 1 | 0 |
|---|---|---|---|---|
| clear display | 화면을 지우고, 커서 Home으로 | | | |
| return home | 화면은 그대로, 커서만 Home으로 | | | |
| entry mode set | I/D(inc/dec) | 커서이동방향 | 증가 | 감소 |
| | S (shift) | 화면이동(data 입력 시) | on | off |
| display on/off control | D (display) | 화면 표시 | on | off |
| | C (cursor) | 커서 표시 | on | off |
| | B (blinking) | 깜박임 | on | off |
| display or cursor shift | S/C | 이동 대상 | screen | cursor |
| | R/L | 이동 방향 | right | left |
| function set | DL | data length | 8-bit | 4-bit |
| | N | number of display | 2 | 1 |
| | F | font size | 5 x 10 | 5 x 7 |

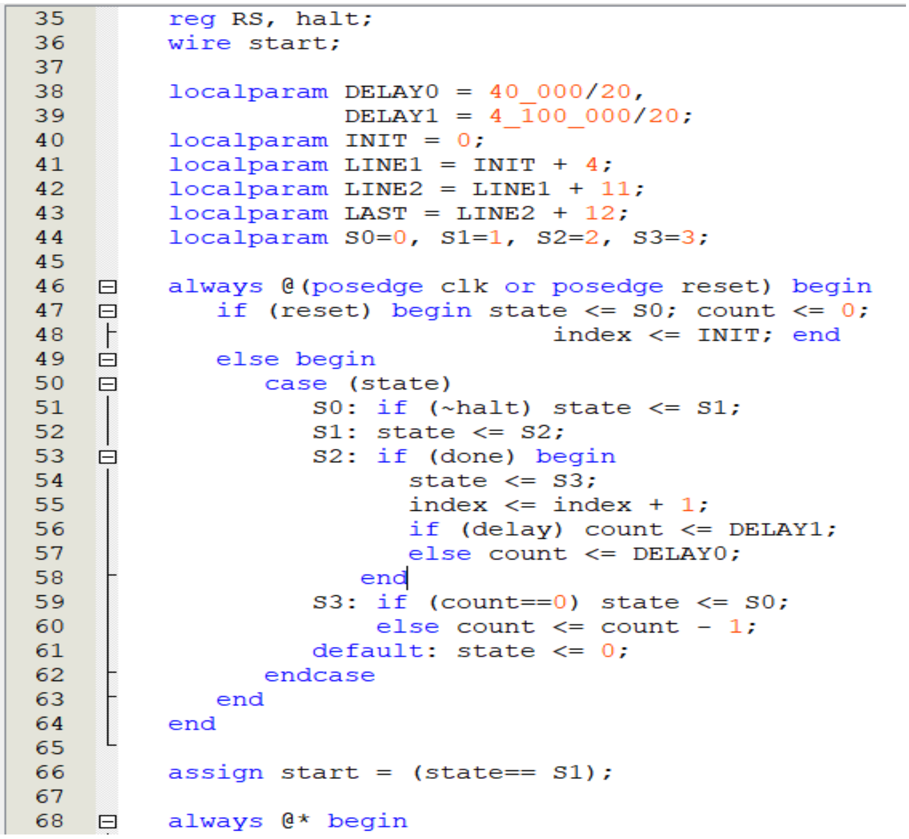
1. 전원공급 50ms 후에 LCD 모듈이 reset이 완료됨
2. Function set 명령 설정 (command = 001xxx00)
3. Display on/off 명령 설정 (command = 00001xxx)
4. Entry mode set 명령 설정 (command = 000001xx)
5. DD RAM address 설정 (command = 1aaaaaaa)

문자 데이터를 write하면 화면에 출력됨
(write할 때마다 address counter가 이동됨)

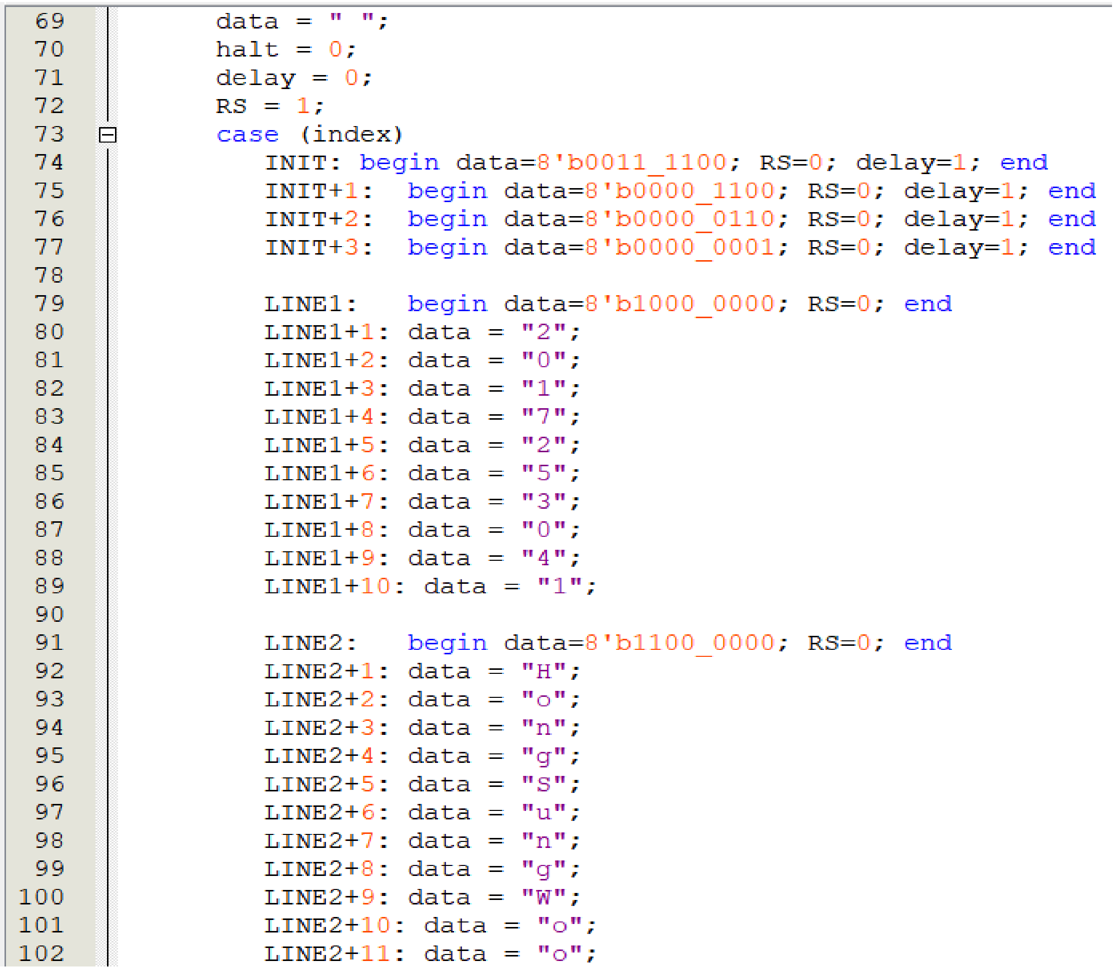
(2)



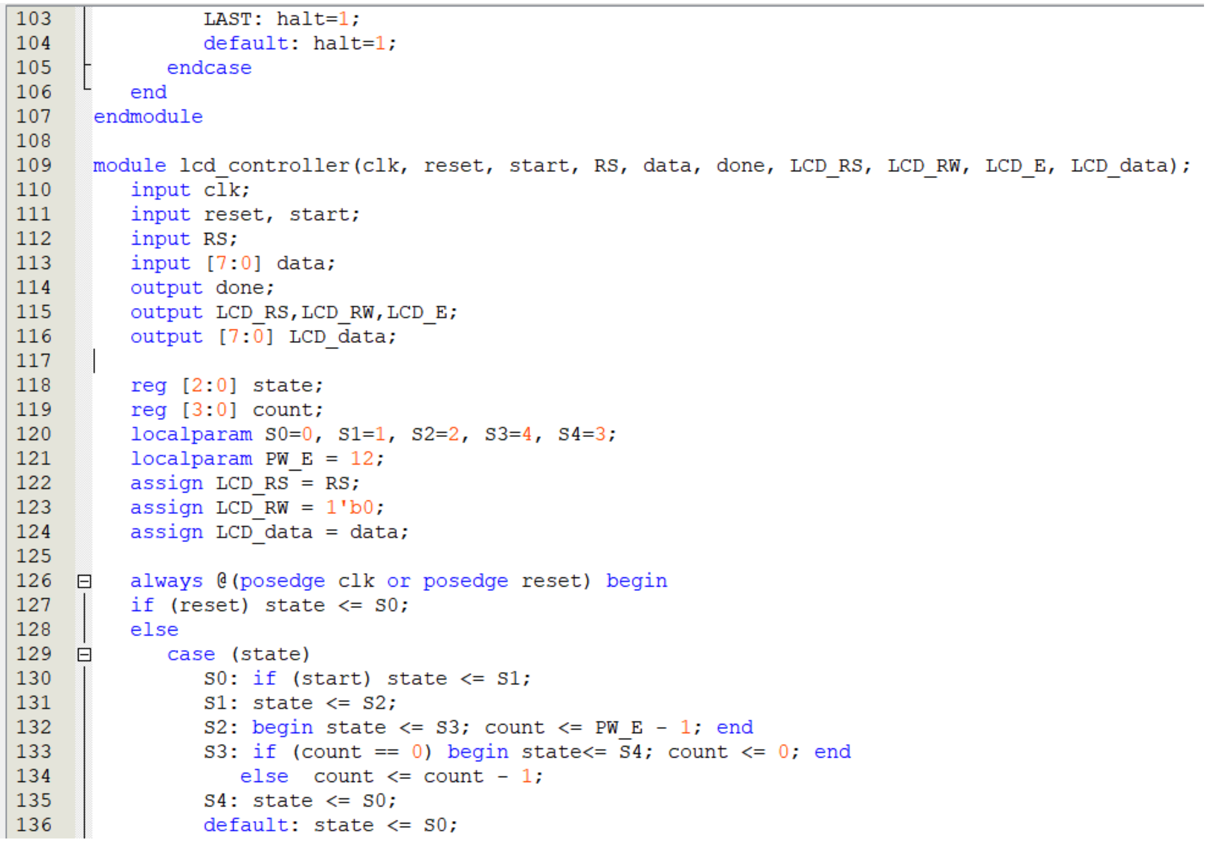
```
module textlcd(CLOCK_50, KEY, LCD_RS, LCD_RW, LCD_EN, LCD_DATA, LCD_ON, LCD_BLON, LEDR);
   input CLOCK_50;
   input [3:0] KEY;
   output LCD_RS, LCD_RW, LCD_EN;
   output [7:0] LCD_DATA;
   output LCD_ON, LCD_BLON;
   output [0:0] LEDR;

   wire start, RS, done;
   wire [7:0] data;

   wire clk_50 = CLOCK_50;
   wire reset = ~KEY[0];

   lcd_test u1 (clk_50, reset, start, RS, data, done);
   lcd_controller u2 (clk_50, reset, start, RS, data, done, LCD_RS, LCD_RW, LCD_EN, LCD_DATA);

   assign LCD_ON = 1'b1;
   assign LCD_BLON = 1'b1;
   assign LEDR[0] = 1'b1;
endmodule

module lcd_test(clk, reset, start, RS, data, done);
   input clk;
   input reset;
   output start, RS;
   output [7:0] data;
   input done;

   reg [5:0] index;
   reg [7:0] data;
   reg [1:0] state;
   reg [17:0] count;
   reg delay;
```



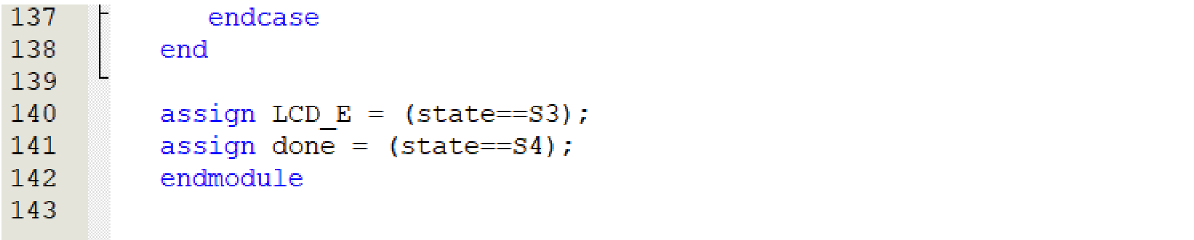
```
reg RS, halt;
wire start;

localparam DELAY0 = 40_000/20,
           DELAY1 = 4_100_000/20;
localparam INIT = 0;
localparam LINE1 = INIT + 4;
localparam LINE2 = LINE1 + 11;
localparam LAST = LINE2 + 12;
localparam S0=0, S1=1, S2=2, S3=3;

always @(posedge clk or posedge reset) begin
   if (reset) begin state <= S0; count <= 0;
                        index <= INIT; end
   else begin
      case (state)
         S0: if (~halt) state <= S1;
         S1: state <= S2;
         S2: if (done) begin
               state <= S3;
               index <= index + 1;
               if (delay) count <= DELAY1;
               else count <= DELAY0;
            end
         S3: if (count==0) state <= S0;
             else count <= count - 1;
         default: state <= 0;
      endcase
   end
end

assign start = (state== S1);

always @* begin
```



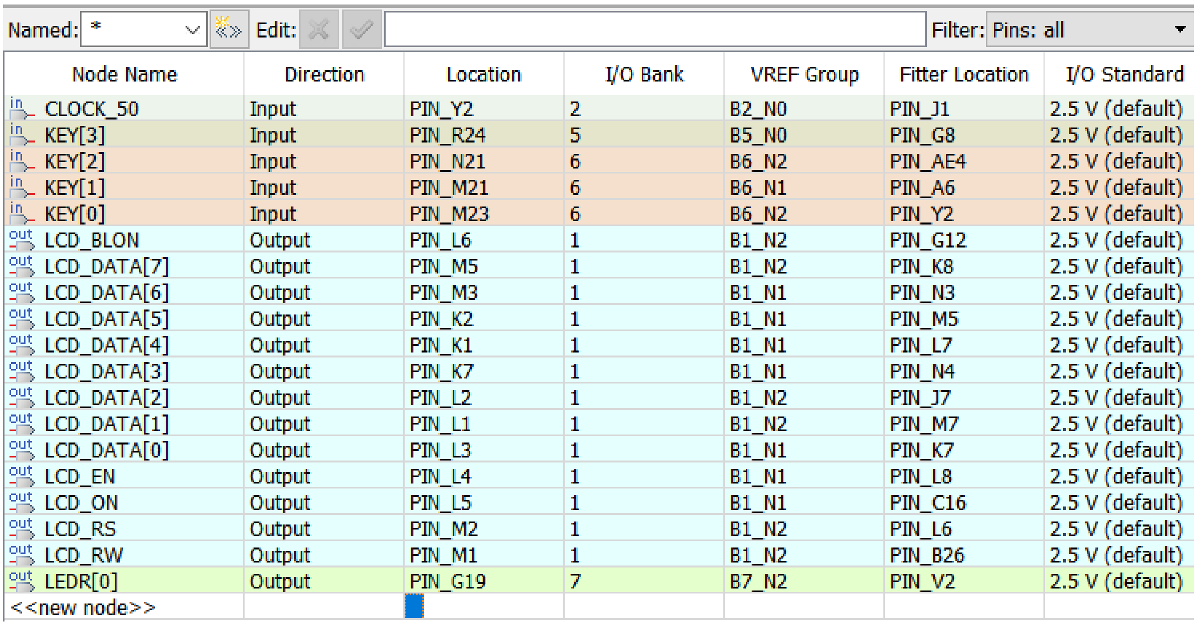
```
   data = " ";
   halt = 0;
   delay = 0;
   RS = 1;
   case (index)
      INIT: begin data=8'b0011_1100; RS=0; delay=1; end
      INIT+1:  begin data=8'b0000_1100; RS=0; delay=1; end
      INIT+2:  begin data=8'b0000_0110; RS=0; delay=1; end
      INIT+3:  begin data=8'b0000_0001; RS=0; delay=1; end

      LINE1:   begin data=8'b1000_0000; RS=0; end
      LINE1+1: data = "2";
      LINE1+2: data = "0";
      LINE1+3: data = "1";
      LINE1+4: data = "7";
      LINE1+5: data = "2";
      LINE1+6: data = "5";
      LINE1+7: data = "3";
      LINE1+8: data = "0";
      LINE1+9: data = "4";
      LINE1+10: data = "1";

      LINE2:   begin data=8'b1100_0000; RS=0; end
      LINE2+1: data = "H";
      LINE2+2: data = "o";
      LINE2+3: data = "n";
      LINE2+4: data = "g";
      LINE2+5: data = "S";
      LINE2+6: data = "u";
      LINE2+7: data = "n";
      LINE2+8: data = "g";
      LINE2+9: data = "W";
      LINE2+10: data = "o";
      LINE2+11: data = "o";
```



```
         LAST: halt=1;
         default: halt=1;
      endcase
   end
endmodule

module lcd_controller(clk, reset, start, RS, data, done, LCD_RS, LCD_RW, LCD_E, LCD_data);
   input clk;
   input reset, start;
   input RS;
   input [7:0] data;
   output done;
   output LCD_RS,LCD_RW,LCD_E;
   output [7:0] LCD_data;

   reg [2:0] state;
   reg [3:0] count;
   localparam S0=0, S1=1, S2=2, S3=4, S4=3;
   localparam PW_E = 12;
   assign LCD_RS = RS;
   assign LCD_RW = 1'b0;
   assign LCD_data = data;

   always @(posedge clk or posedge reset) begin
   if (reset) state <= S0;
   else
      case (state)
         S0: if (start) state <= S1;
         S1: state <= S2;
         S2: begin state <= S3; count <= PW_E - 1; end
         S3: if (count == 0) begin state<= S4; count <= 0; end
            else  count <= count - 1;
         S4: state <= S0;
         default: state <= S0;
```

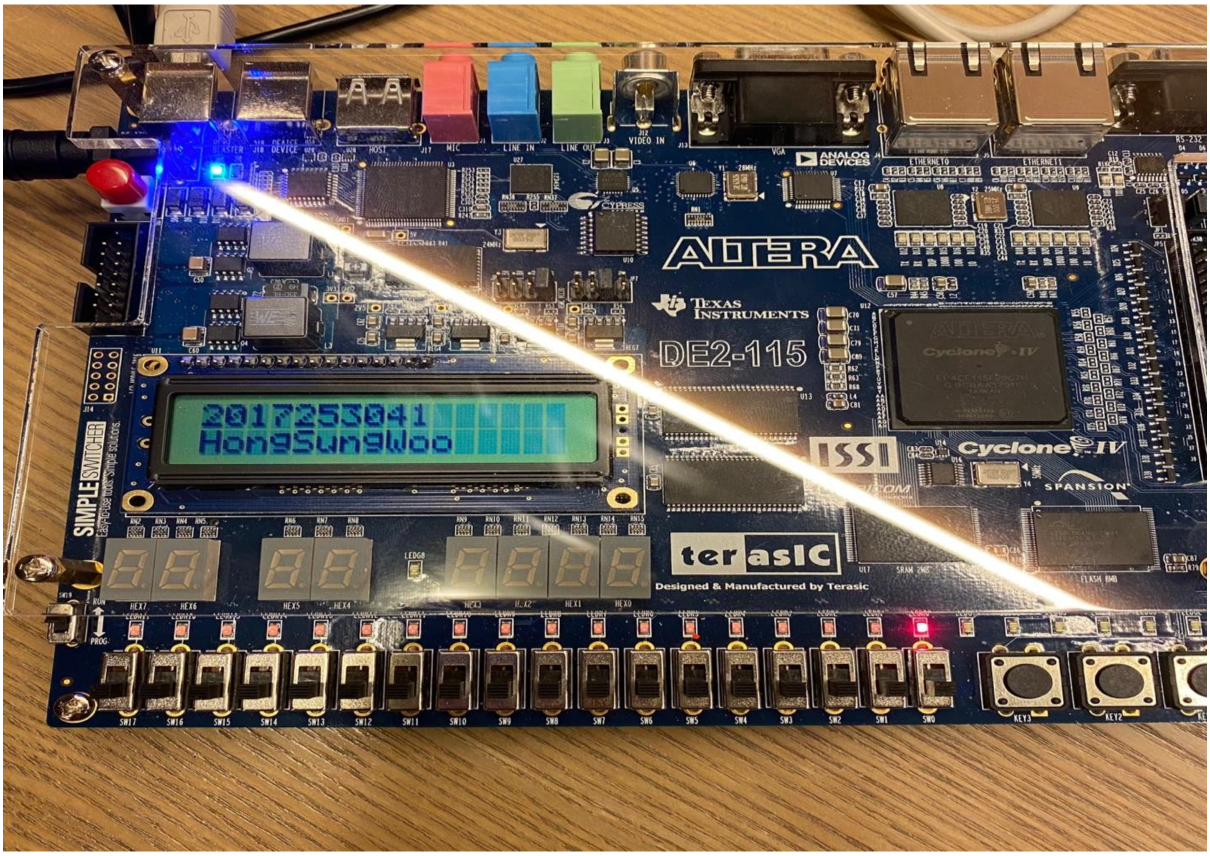


```
      endcase
   end

   assign LCD_E = (state==S3);
   assign done = (state==S4);
   endmodule
```

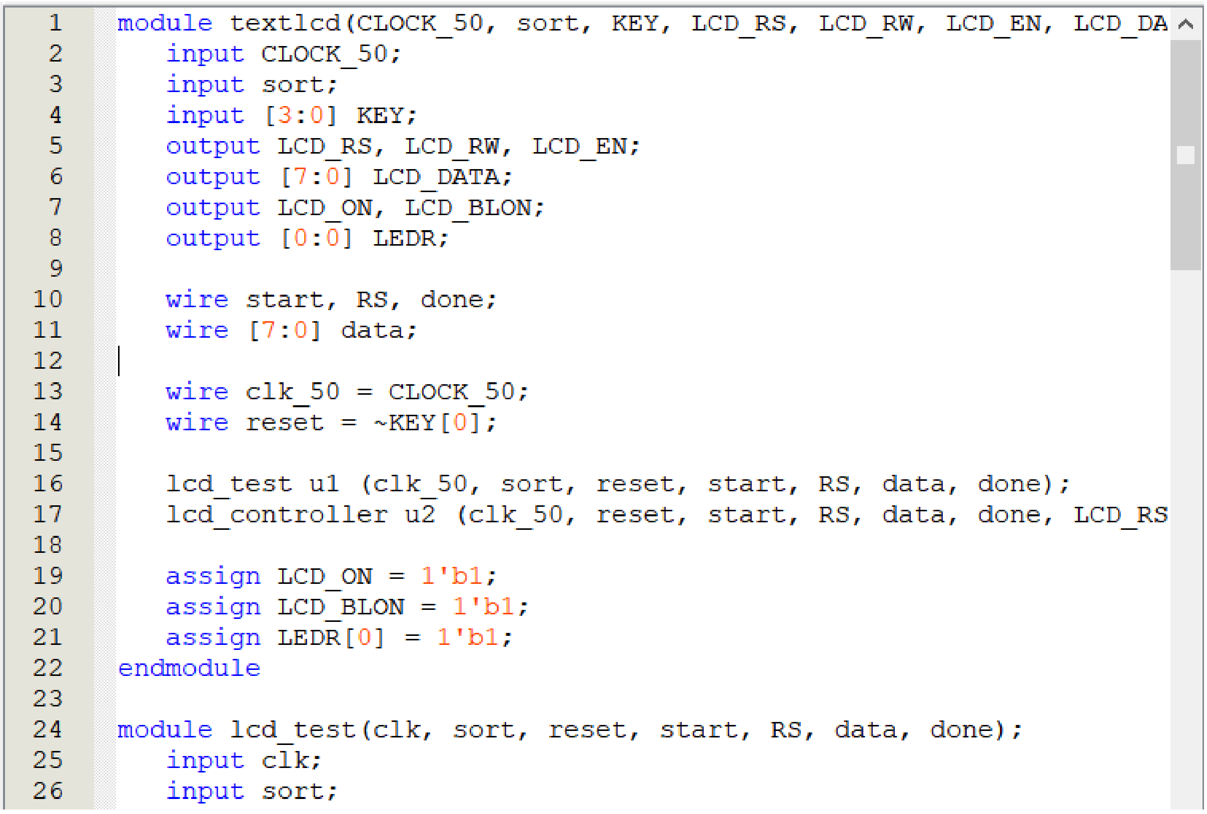


Named: * Edit: Filter: Pins: all

| Node Name | Direction | Location | I/O Bank | VREF Group | Fitter Location | I/O Standard |
|---|---|---|---|---|---|---|
| CLOCK_50 | Input | PIN_Y2 | 2 | B2_N0 | PIN_J1 | 2.5 V (default) |
| KEY[3] | Input | PIN_R24 | 5 | B5_N0 | PIN_G8 | 2.5 V (default) |
| KEY[2] | Input | PIN_N21 | 6 | B6_N2 | PIN_AE4 | 2.5 V (default) |
| KEY[1] | Input | PIN_M21 | 6 | B6_N1 | PIN_A6 | 2.5 V (default) |
| KEY[0] | Input | PIN_M23 | 6 | B6_N2 | PIN_Y2 | 2.5 V (default) |
| LCD_BLON | Output | PIN_L6 | 1 | B1_N2 | PIN_G12 | 2.5 V (default) |
| LCD_DATA[7] | Output | PIN_M5 | 1 | B1_N2 | PIN_K8 | 2.5 V (default) |
| LCD_DATA[6] | Output | PIN_M3 | 1 | B1_N1 | PIN_N3 | 2.5 V (default) |
| LCD_DATA[5] | Output | PIN_K2 | 1 | B1_N1 | PIN_M5 | 2.5 V (default) |
| LCD_DATA[4] | Output | PIN_K1 | 1 | B1_N1 | PIN_L7 | 2.5 V (default) |
| LCD_DATA[3] | Output | PIN_K7 | 1 | B1_N1 | PIN_N4 | 2.5 V (default) |
| LCD_DATA[2] | Output | PIN_L2 | 1 | B1_N2 | PIN_J7 | 2.5 V (default) |
| LCD_DATA[1] | Output | PIN_L1 | 1 | B1_N2 | PIN_M7 | 2.5 V (default) |
| LCD_DATA[0] | Output | PIN_L3 | 1 | B1_N1 | PIN_K7 | 2.5 V (default) |
| LCD_EN | Output | PIN_L4 | 1 | B1_N1 | PIN_L8 | 2.5 V (default) |
| LCD_ON | Output | PIN_L5 | 1 | B1_N1 | PIN_C16 | 2.5 V (default) |
| LCD_RS | Output | PIN_M2 | 1 | B1_N2 | PIN_L6 | 2.5 V (default) |
| LCD_RW | Output | PIN_M1 | 1 | B1_N2 | PIN_B26 | 2.5 V (default) |
| LEDR[0] | Output | PIN_G19 | 7 | B7_N2 | PIN_V2 | 2.5 V (default) |
| <<new node>> | | | | | | |

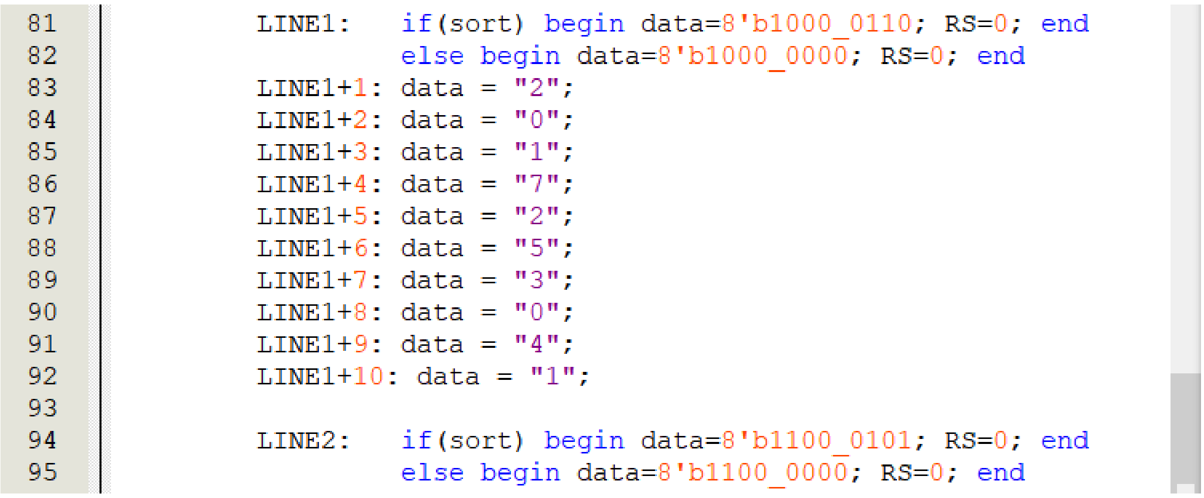


(3)



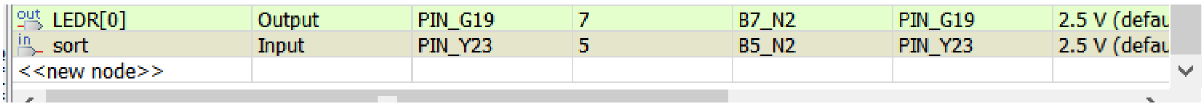
```
module textlcd(CLOCK_50, sort, KEY, LCD_RS, LCD_RW, LCD_EN, LCD_DA
   input CLOCK_50;
   input sort;
   input [3:0] KEY;
   output LCD_RS, LCD_RW, LCD_EN;
   output [7:0] LCD_DATA;
   output LCD_ON, LCD_BLON;
   output [0:0] LEDR;

   wire start, RS, done;
   wire [7:0] data;

   wire clk_50 = CLOCK_50;
   wire reset = ~KEY[0];

   lcd_test u1 (clk_50, sort, reset, start, RS, data, done);
   lcd_controller u2 (clk_50, reset, start, RS, data, done, LCD_RS

   assign LCD_ON = 1'b1;
   assign LCD_BLON = 1'b1;
   assign LEDR[0] = 1'b1;
endmodule

module lcd_test(clk, sort, reset, start, RS, data, done);
   input clk;
   input sort;
```

-> (2)번 소스코드에서 sort만 부분부분 추가해줬다.



```
        LINE1:   if(sort) begin data=8'b1000_0110; RS=0; end
                 else begin data=8'b1000_0000; RS=0; end
        LINE1+1: data = "2";
        LINE1+2: data = "0";
        LINE1+3: data = "1";
        LINE1+4: data = "7";
        LINE1+5: data = "2";
        LINE1+6: data = "5";
        LINE1+7: data = "3";
        LINE1+8: data = "0";
        LINE1+9: data = "4";
        LINE1+10: data = "1";

        LINE2:   if(sort) begin data=8'b1100_0101; RS=0; end
                 else begin data=8'b1100_0000; RS=0; end
```

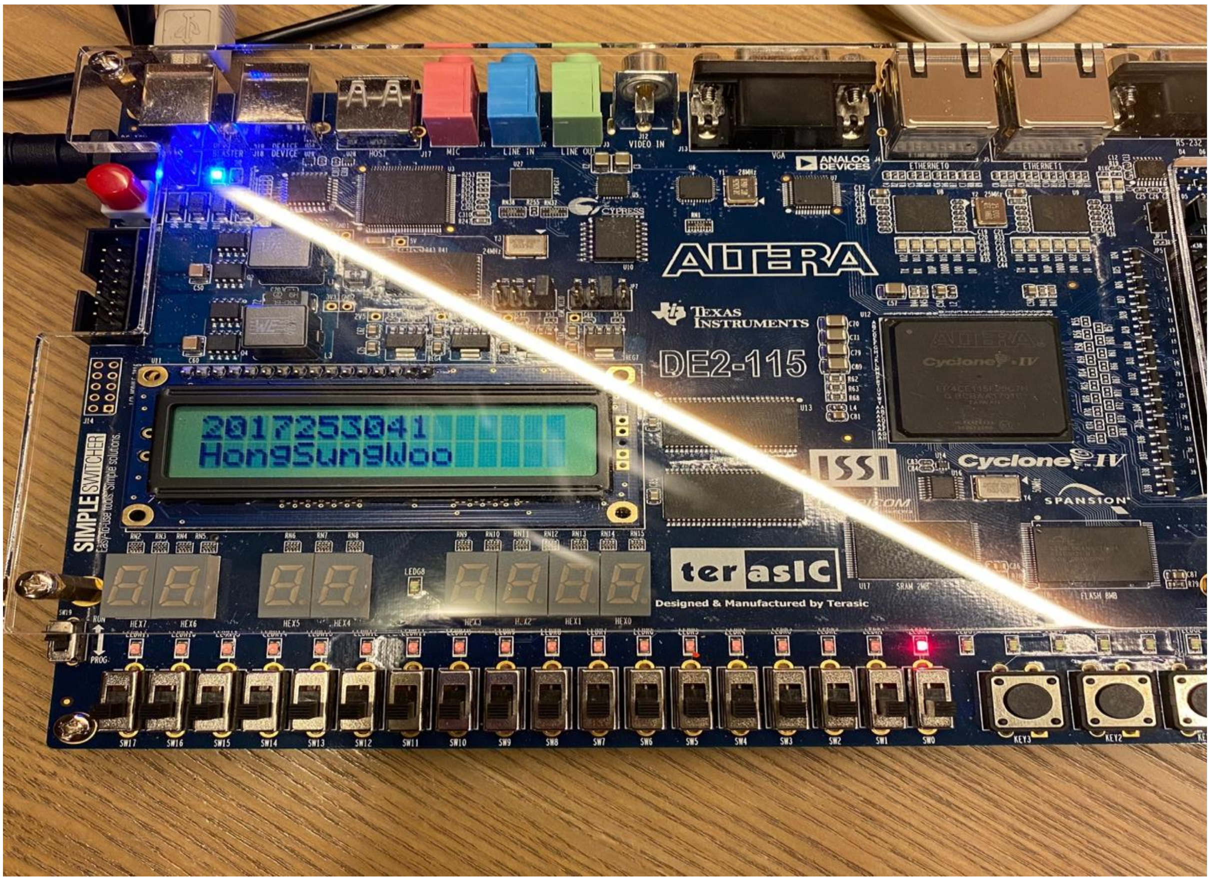
-> (2)번 소스코드에서 sort만 부분부분 추가해줬고, LINE2에선 이름이 11자리여서 뒤에 7bit를 45에서 시작할 수 있도록 설정해줬다.



| | | | | | | |
|---|---|---|---|---|---|---|
| out LEDR[0] | Output | PIN_G19 | 7 | B7_N2 | PIN_G19 | 2.5 V (defau |
| in sort | Input | PIN_Y23 | 5 | B5_N2 | PIN_Y23 | 2.5 V (defau |
| <<new node>> | | | | | | |

-> sort를 SW[17]로 설정해줬다. (교수님이 과제에선 SW[0]으로 설정하라고 하셨지만 SW[17]으로 설정하는 것이 더 보기 편했습니다. 죄송합니다☹)

< SW[17]을 내렸을 때 >



< SW[17]을 올렸을 때 >