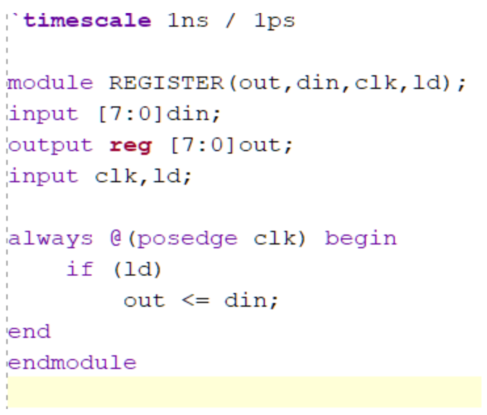
FIR FILTER USING MAC

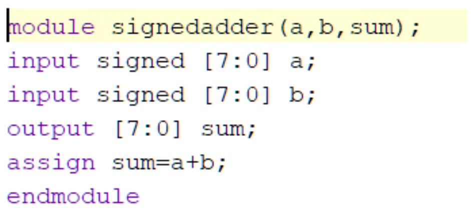
- VERILOG CODE

8 BIT REGISTER



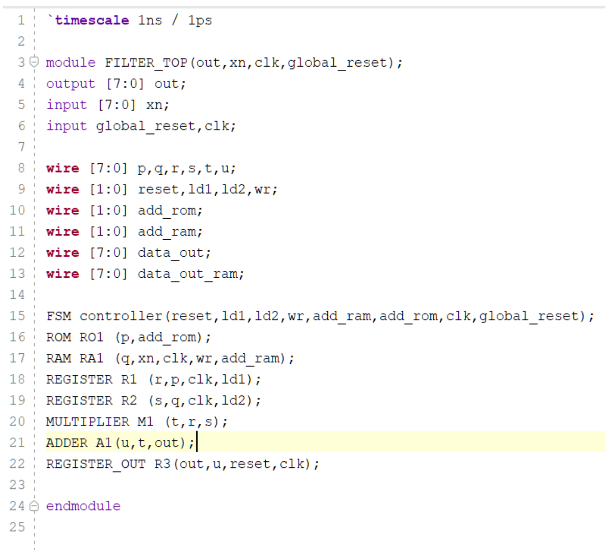
```verilog
`timescale 1ns / 1ps

module REGISTER(out,din,clk,ld);
input [7:0]din;
output reg [7:0]out;
input clk,ld;

always @(posedge clk) begin
    if (ld)
        out <= din;
end
endmodule
```

8 BIT SIGNED ADDER



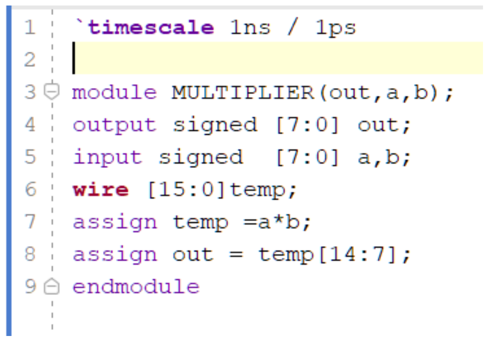
```verilog
module signedadder(a,b,sum);
input signed [7:0] a;
input signed [7:0] b;
output [7:0] sum;
assign sum=a+b;
endmodule
```

TOP MODULE



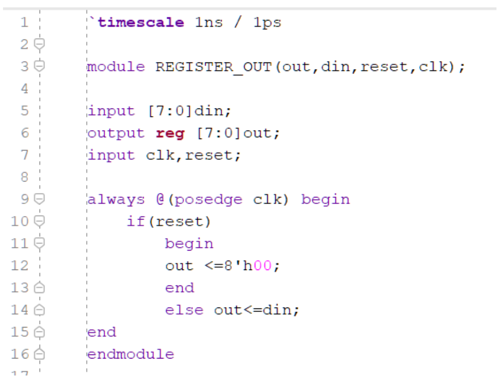
```verilog
`timescale 1ns / 1ps

module FILTER_TOP(out,xn,clk,global_reset);
output [7:0] out;
input [7:0] xn;
input global_reset,clk;

wire [7:0] p,q,r,s,t,u;
wire [1:0] reset,ld1,ld2,wr;
wire [1:0] add_rom;
wire [1:0] add_ram;
wire [7:0] data_out;
wire [7:0] data_out_ram;

FSM controller(reset,ld1,ld2,wr,add_ram,add_rom,clk,global_reset);
ROM RO1 (p,add_rom);
RAM RA1 (q,xn,clk,wr,add_ram);
REGISTER R1 (r,p,clk,ld1);
REGISTER R2 (s,q,clk,ld2);
MULTIPLIER M1 (t,r,s);
ADDER A1(u,t,out);
REGISTER_OUT R3(out,u,reset,clk);

endmodule
```

8 BIT SIGNED MULTIPLIER



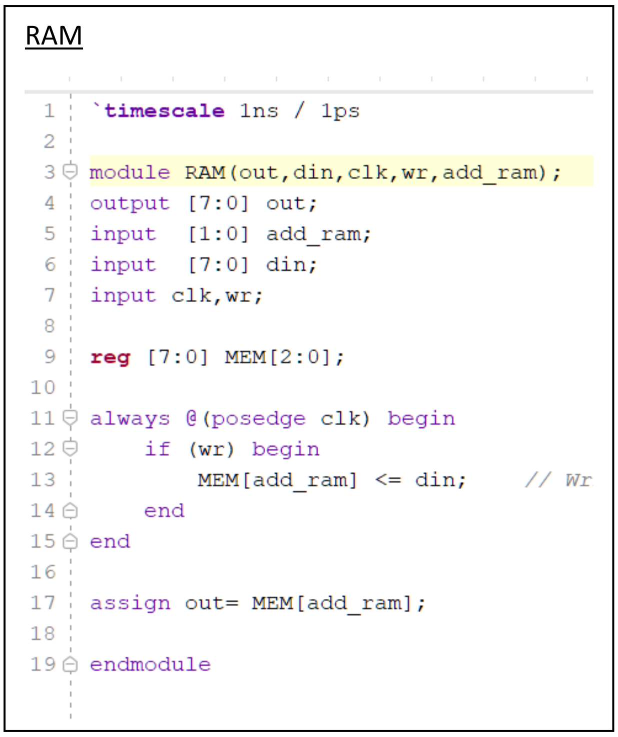
```verilog
`timescale 1ns / 1ps

module MULTIPLIER(out,a,b);
output signed [7:0] out;
input signed  [7:0] a,b;
wire [15:0]temp;
assign temp =a*b;
assign out = temp[14:7];
endmodule
```

8 BIT OUTPUT REGISTER



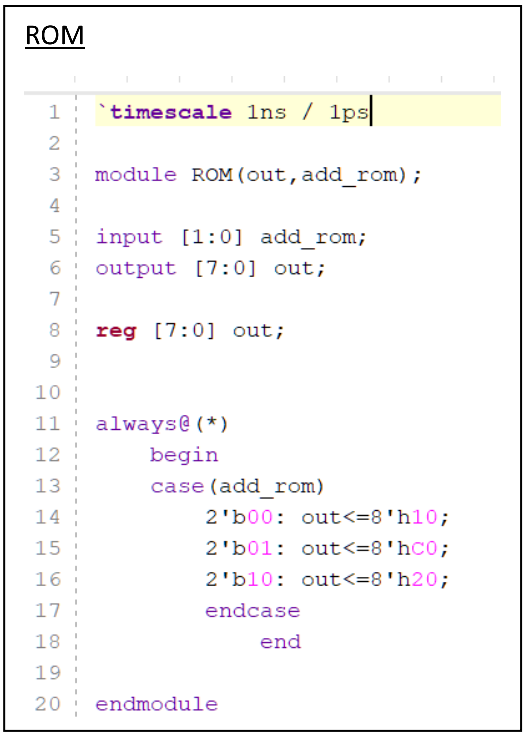
```verilog
`timescale 1ns / 1ps

module REGISTER_OUT(out,din,reset,clk);

input [7:0]din;
output reg [7:0]out;
input clk,reset;

always @(posedge clk) begin
    if(reset)
        begin
        out <=8'h00;
        end
        else out<=din;
end
endmodule
```

RAM



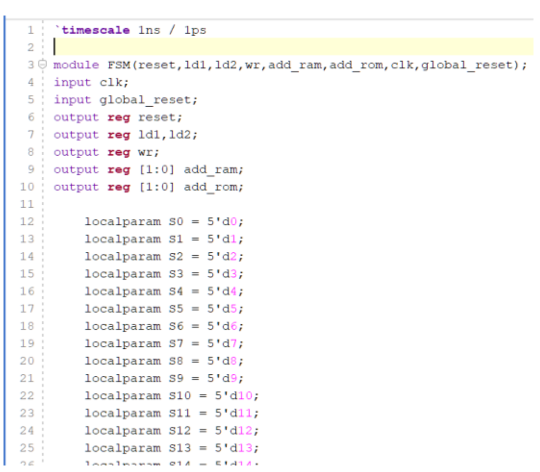
```verilog
`timescale 1ns / 1ps

module RAM(out,din,clk,wr,add_ram);
output [7:0] out;
input  [1:0] add_ram;
input  [7:0] din;
input clk,wr;

reg [7:0] MEM[2:0];

always @(posedge clk) begin
    if (wr) begin
        MEM[add_ram] <= din;     // Wr
    end
end

assign out= MEM[add_ram];

endmodule
```

ROM

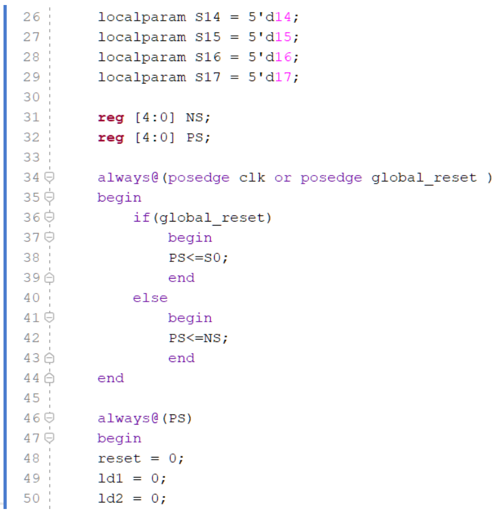


```verilog
`timescale 1ns / 1ps

module ROM(out,add_rom);

input [1:0] add_rom;
output [7:0] out;

reg [7:0] out;


always@(*)
    begin
    case(add_rom)
        2'b00: out<=8'h10;
        2'b01: out<=8'hC0;
        2'b10: out<=8'h20;
        endcase
            end

endmodule
```

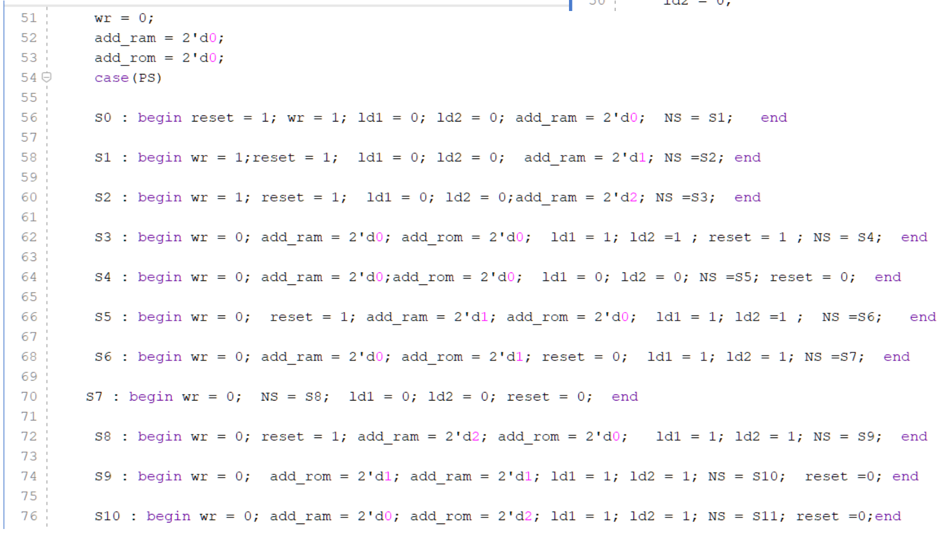
FSM



```verilog
`timescale 1ns / 1ps

module FSM(reset,ld1,ld2,wr,add_ram,add_rom,clk,global_reset);
input clk;
input global_reset;
output reg reset;
output reg ld1,ld2;
output reg wr;
output reg [1:0] add_ram;
output reg [1:0] add_rom;

    localparam S0 = 5'd0;
    localparam S1 = 5'd1;
    localparam S2 = 5'd2;
    localparam S3 = 5'd3;
    localparam S4 = 5'd4;
    localparam S5 = 5'd5;
    localparam S6 = 5'd6;
    localparam S7 = 5'd7;
    localparam S8 = 5'd8;
    localparam S9 = 5'd9;
    localparam S10 = 5'd10;
    localparam S11 = 5'd11;
    localparam S12 = 5'd12;
    localparam S13 = 5'd13;
```



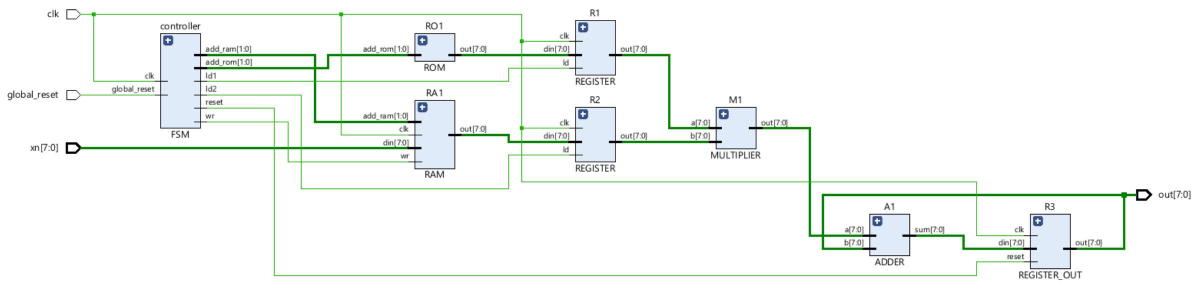
```verilog
    localparam S14 = 5'd14;
    localparam S15 = 5'd15;
    localparam S16 = 5'd16;
    localparam S17 = 5'd17;

    reg [4:0] NS;
    reg [4:0] PS;

    always@(posedge clk or posedge global_reset )
    begin
        if(global_reset)
            begin
            PS<=S0;
            end
        else
            begin
            PS<=NS;
            end
    end

    always@(PS)
    begin
    reset = 0;
    ld1 = 0;
    ld2 = 0;
```



```verilog
    wr = 0;
    add_ram = 2'd0;
    add_rom = 2'd0;
    case(PS)

    S0 : begin reset = 1; wr = 1; ld1 = 0; ld2 = 0; add_ram = 2'd0;  NS = S1;   end

    S1 : begin wr = 1;reset = 1;  ld1 = 0; ld2 = 0;  add_ram = 2'd1; NS =S2; end

    S2 : begin wr = 1; reset = 1;  ld1 = 0; ld2 = 0;add_ram = 2'd2; NS =S3;  end

    S3 : begin wr = 0; add_ram = 2'd0; add_rom = 2'd0;  ld1 = 1; ld2 =1 ; reset = 1 ; NS = S4;  end

    S4 : begin wr = 0; add_ram = 2'd0;add_rom = 2'd0;  ld1 = 0; ld2 = 0; NS =S5; reset = 0;  end

    S5 : begin wr = 0;  reset = 1; add_ram = 2'd1; add_rom = 2'd0;  ld1 = 1; ld2 =1 ;  NS =S6;   end

    S6 : begin wr = 0; add_ram = 2'd0; add_rom = 2'd1; reset = 0;  ld1 = 1; ld2 = 1; NS =S7;  end

   S7 : begin wr = 0;  NS = S8;  ld1 = 0; ld2 = 0; reset = 0;  end

    S8 : begin wr = 0; reset = 1; add_ram = 2'd2; add_rom = 2'd0;   ld1 = 1; ld2 = 1; NS = S9;  end

    S9 : begin wr = 0;  add_rom = 2'd1; add_ram = 2'd1; ld1 = 1; ld2 = 1; NS = S10;  reset =0; end

    S10 : begin wr = 0; add_ram = 2'd0; add_rom = 2'd2; ld1 = 1; ld2 = 1; NS = S11; reset =0;end
```

```verilog

 S10 : begin wr = 0; add_ram = 2'd0; add_rom = 2'd2; ld1 = 1; ld2 = 1; NS = S11; reset =0;end

 S11 : begin wr = 0; add_ram = 2'd0;  ld1 = 0; ld2 = 0; NS = S12;reset =0; end

 S12 : begin wr = 0; reset = 1; add_rom =2'd1; add_ram = 2'd2;  ld1 = 1; ld2 = 1 ;  NS =S13;  end

 S13 : begin wr = 0; reset =0;  add_rom = 2'd2; add_ram = 2'd1;  ld1 = 1; ld2 = 1; NS =S14;  end

S14 : begin wr = 0; add_ram = 2'd0;  ld1 = 0; ld2 = 0; NS =S15; reset =0; end

 S15 : begin wr = 0; reset =1; add_rom = 2'd2; add_ram = 2'd2; ld1 = 1; ld2 = 1;  NS =S16; end

 S16 : begin wr = 0; reset = 0; add_ram = 2'd0; ld1 = 0; ld2 = 0; NS =S0; end


 endcase
 end

endmodule
```

- <u>RTL SCHEMATIC</u>

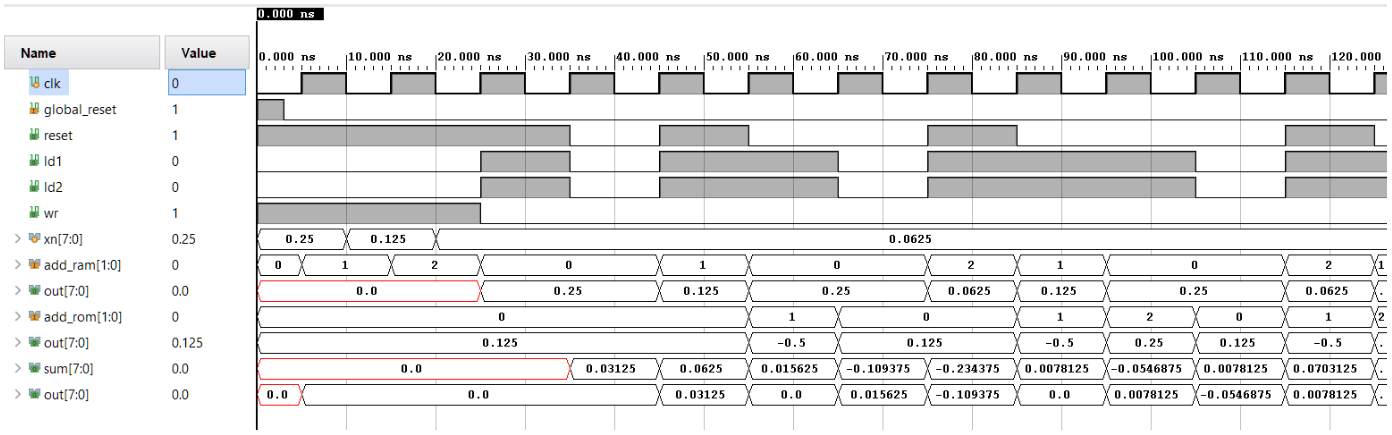


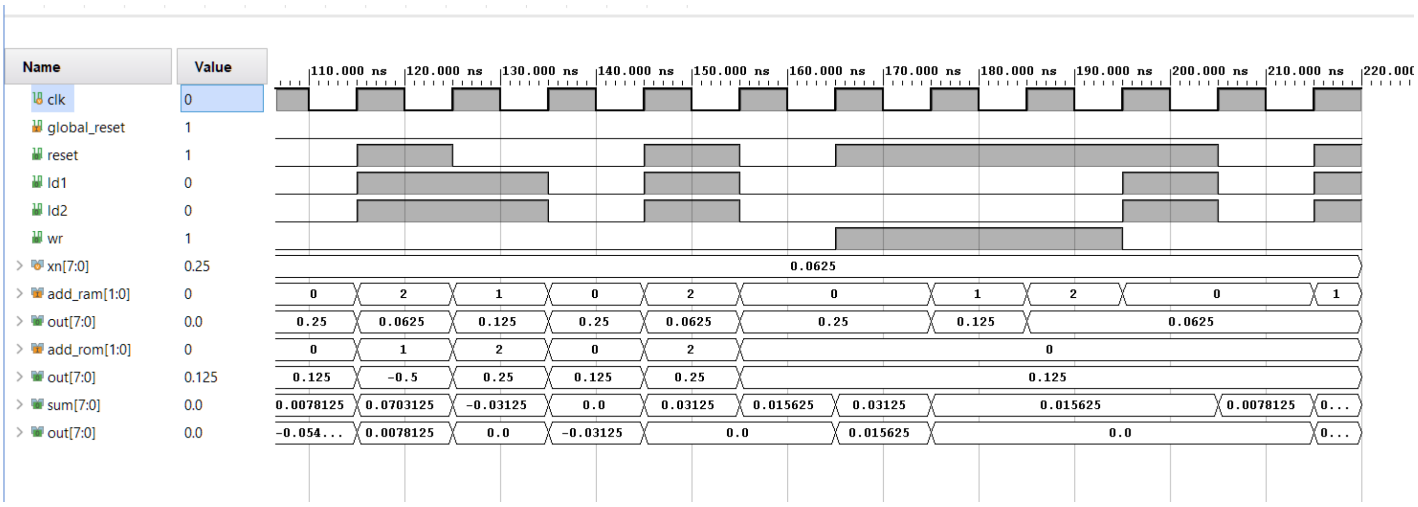
- <u>TESTBENCH</u>

```verilog
`timescale 1ns / 1ps

module FILTER_TEST_BENCH();

reg [7:0] xn;

reg clk, global_reset;

wire [7:0] out;

FILTER_TOP dut(out,xn,clk,global_reset);

initial begin
    clk=1'b0;
 end
always #5 clk=~clk;

initial begin
global_reset = 1'b1;
xn=8'h20;
#3 global_reset = 1'b0;
#7 xn=8'h10;
#10 xn=8'h08;
#200 $finish;
end
```

- SIMULATION RESULTS

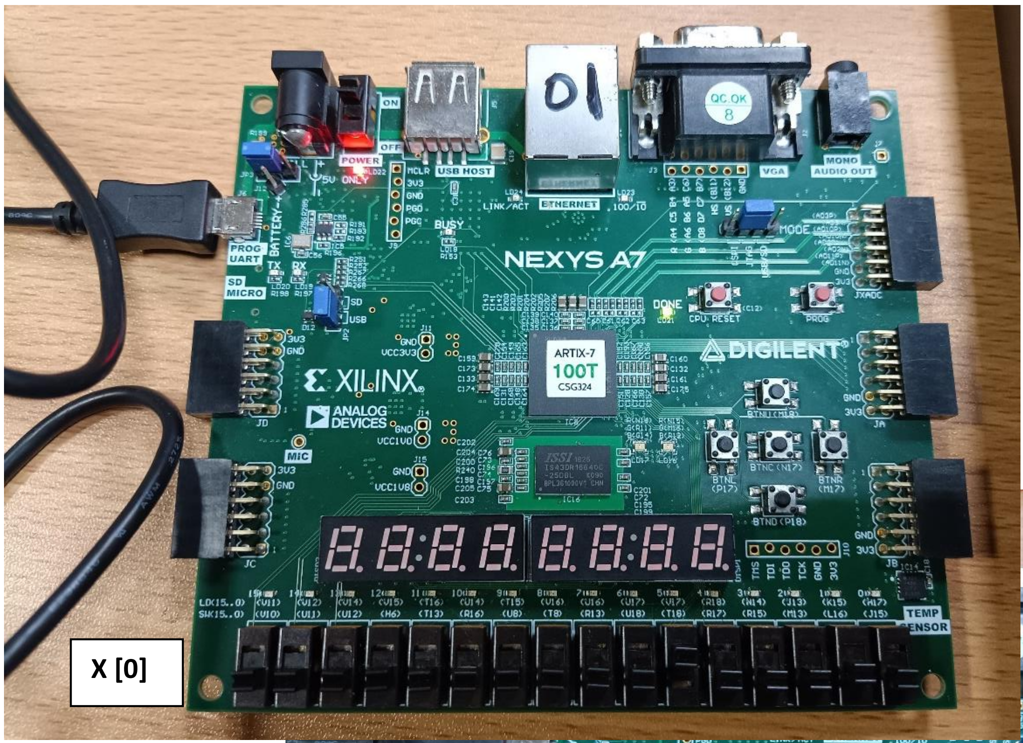
	xn	hn
1	00100000 //0.25	00010000 //0.125
2	00010000 //0.125	11000000 //-0.5
3	00001000 //0.0625	00100000 //0.25



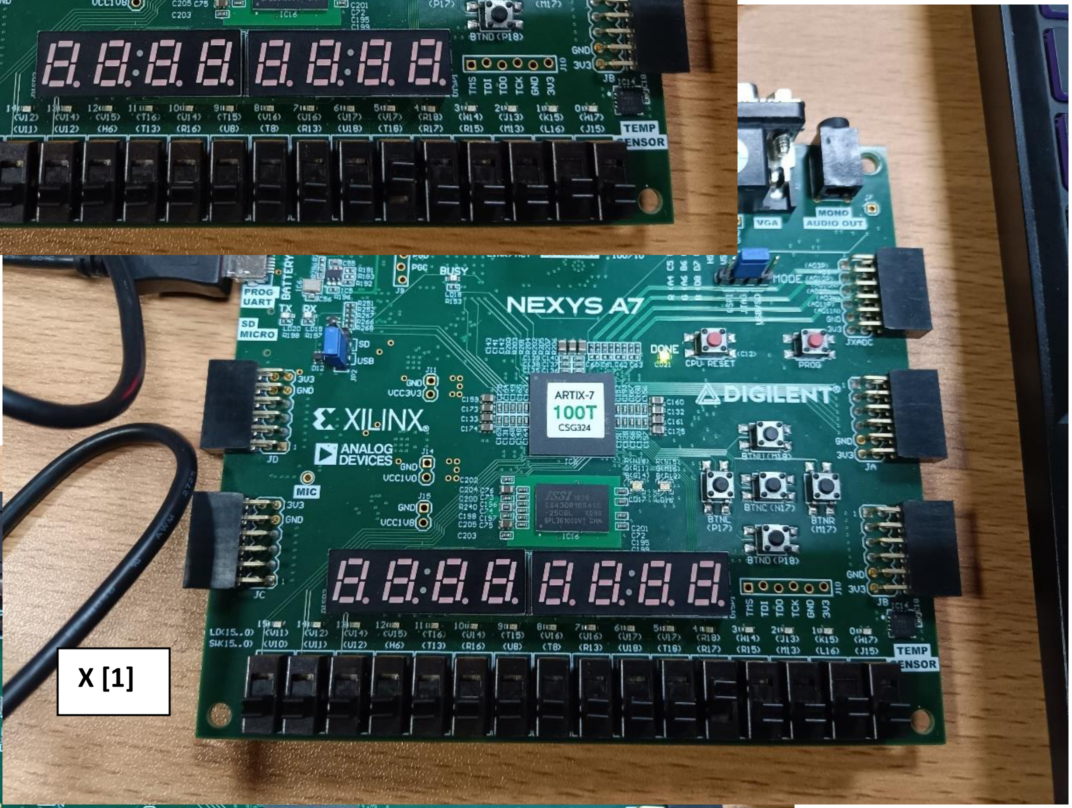


OUTPUT	
Y [0]	0.03125 (00000100)
Y [1]	-0.109375 (11110010)
Y [2]	0.0078125 (00000001)
Y [3]	0 (00000000)
Y [4]	0.015625 (00000010)

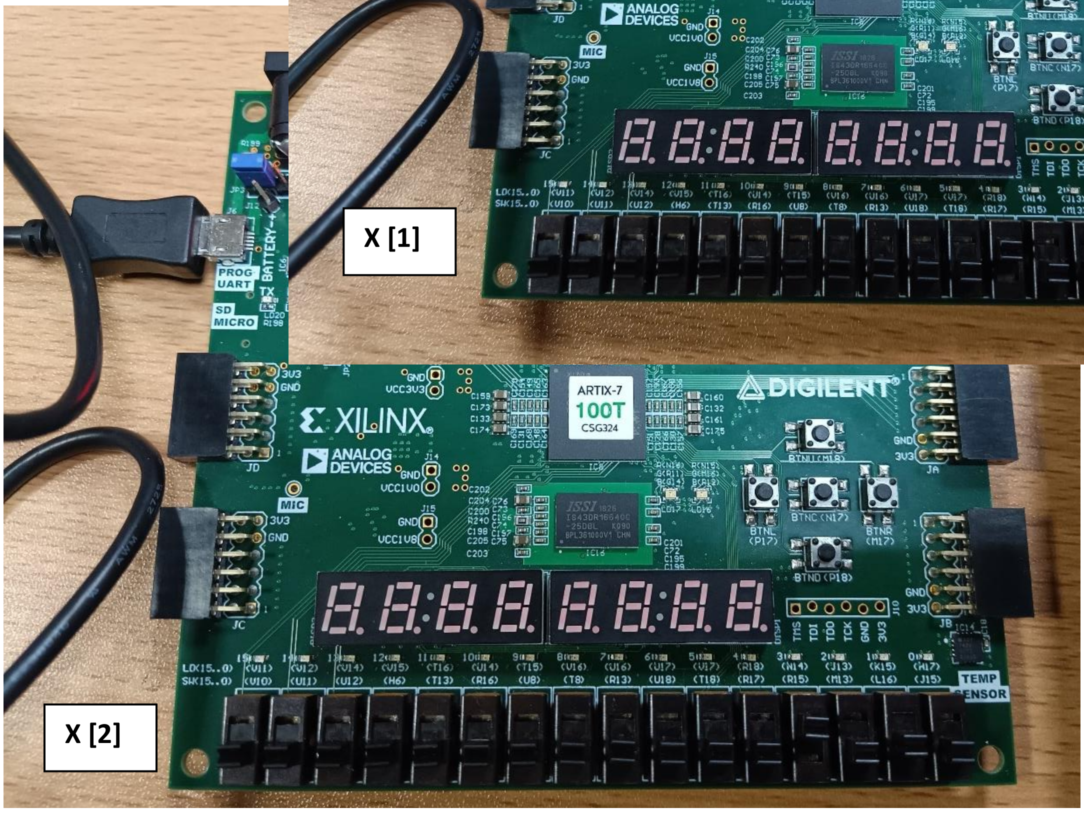
- FPGA IMPLEMENTATION RESULTS



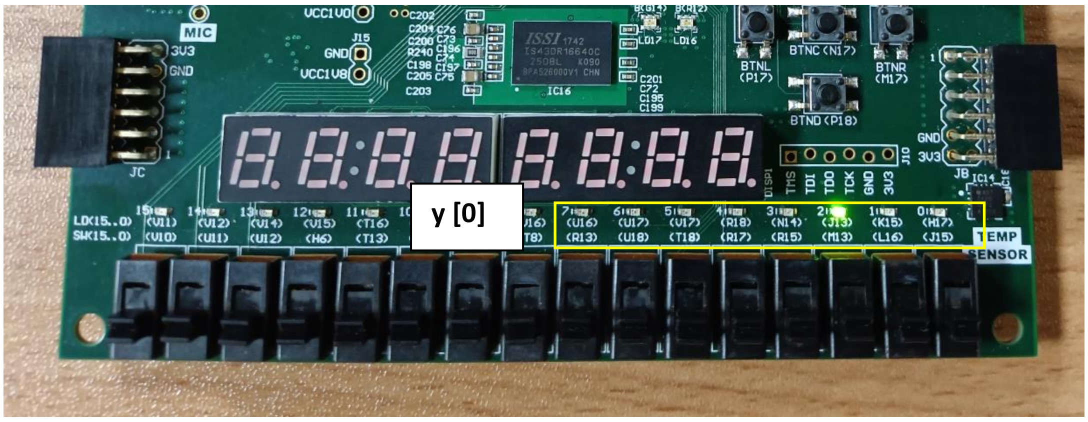
X [0]

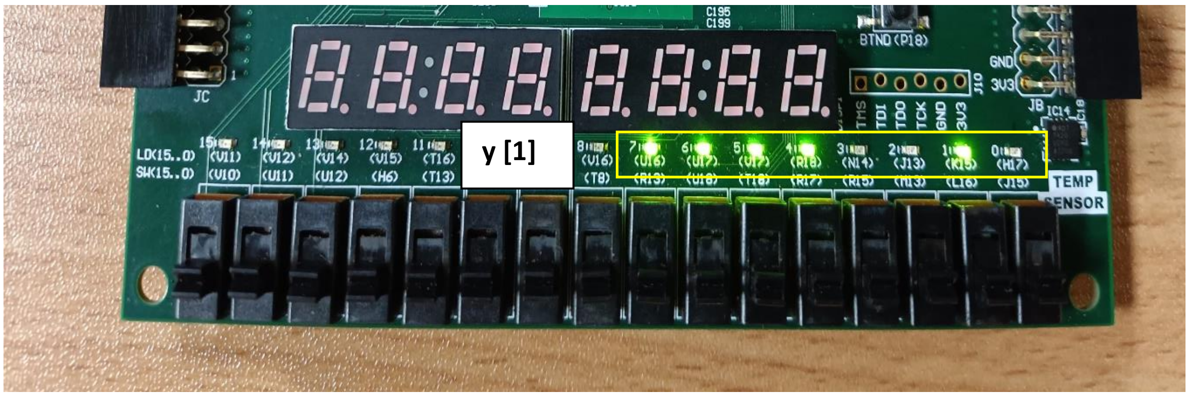


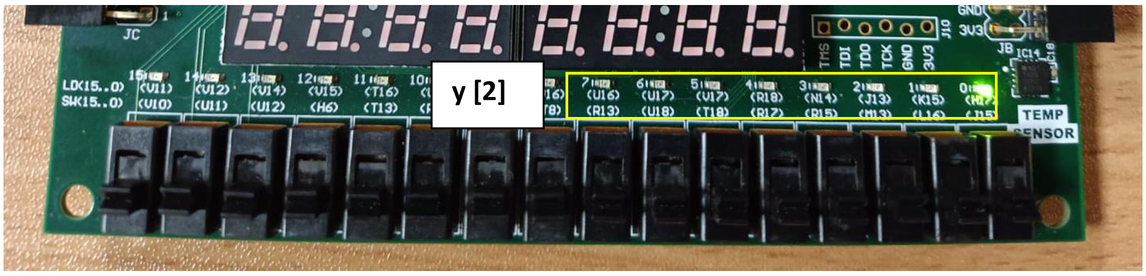
X [1]

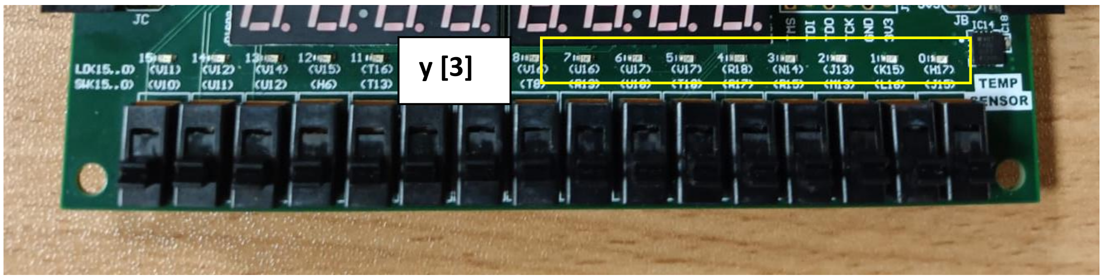


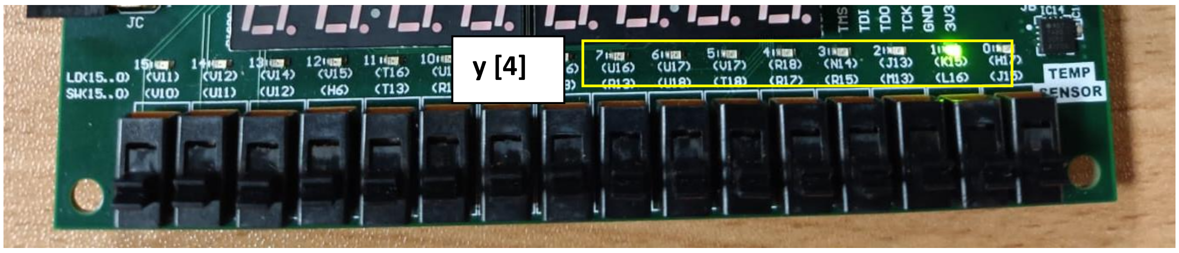
X [2]


y [0]


y [1]


y [2]


y [3]


y [4]