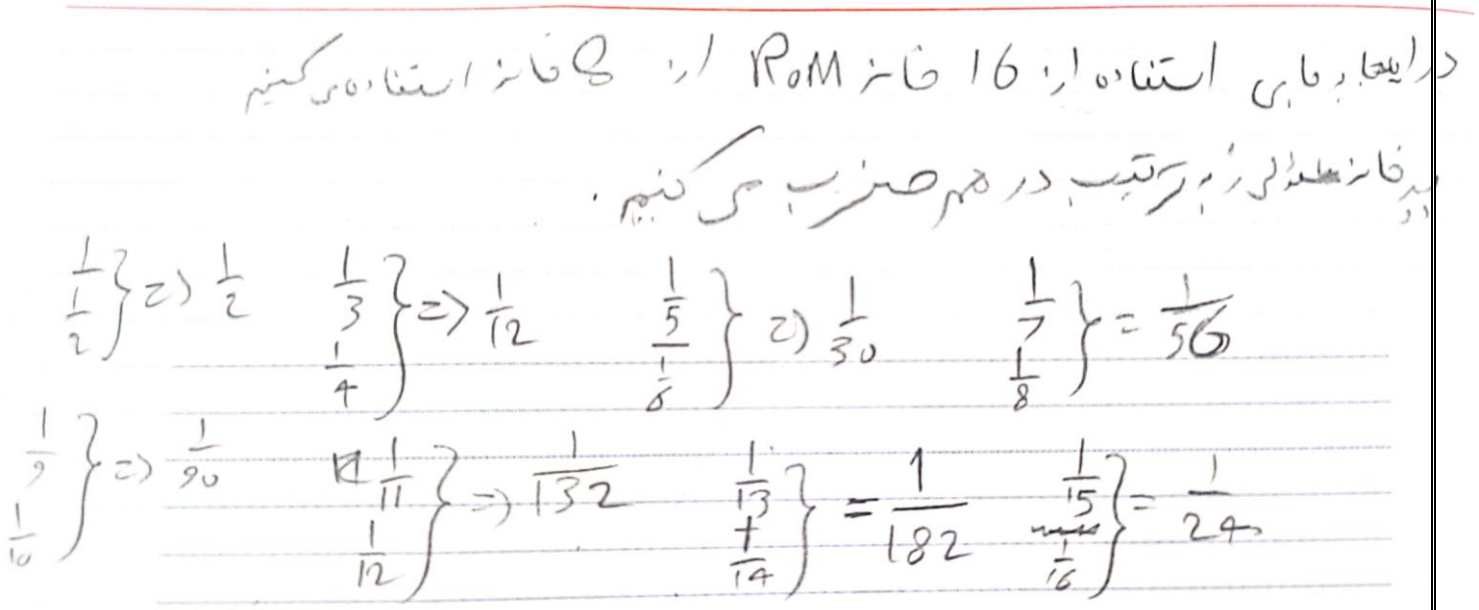
In The Name of God

Mohammad Mashreghi – SID =810199492 - LN = 89

1)

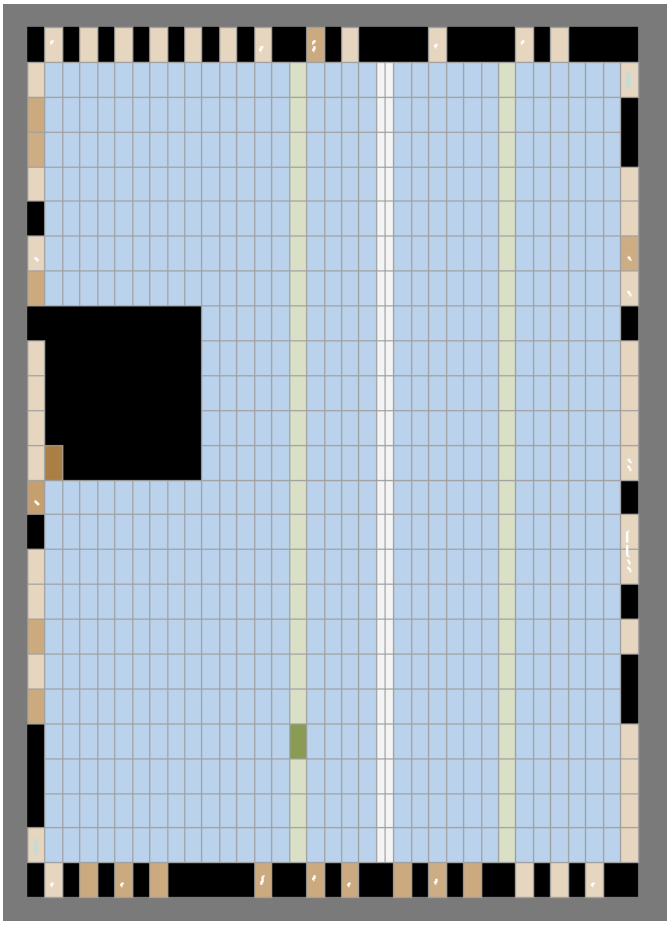
A)

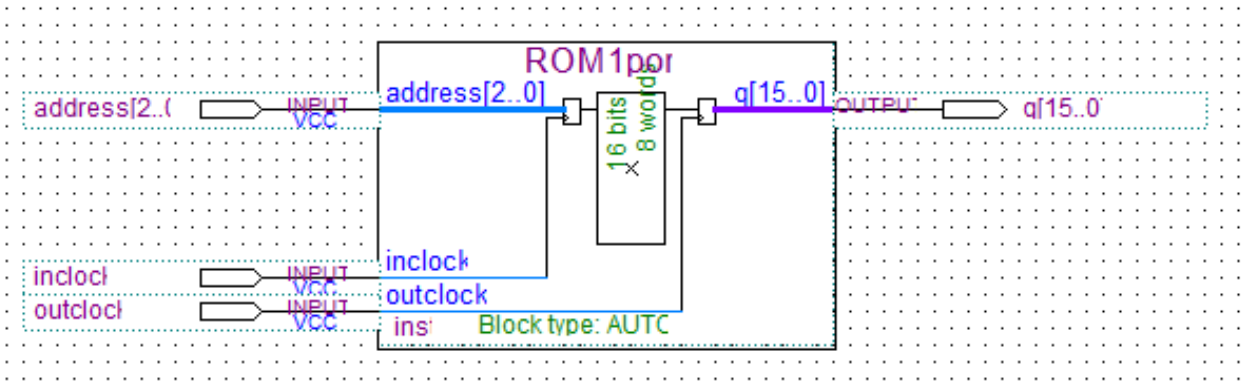
```
e = 1;
a = 1;
for (k = 1; k < 16; k=k+2) {
    a = a * x;
    a = a * x;
    a = a * (1/k)
    a = a * (1/(k+1))
    e = e + a;
}
```



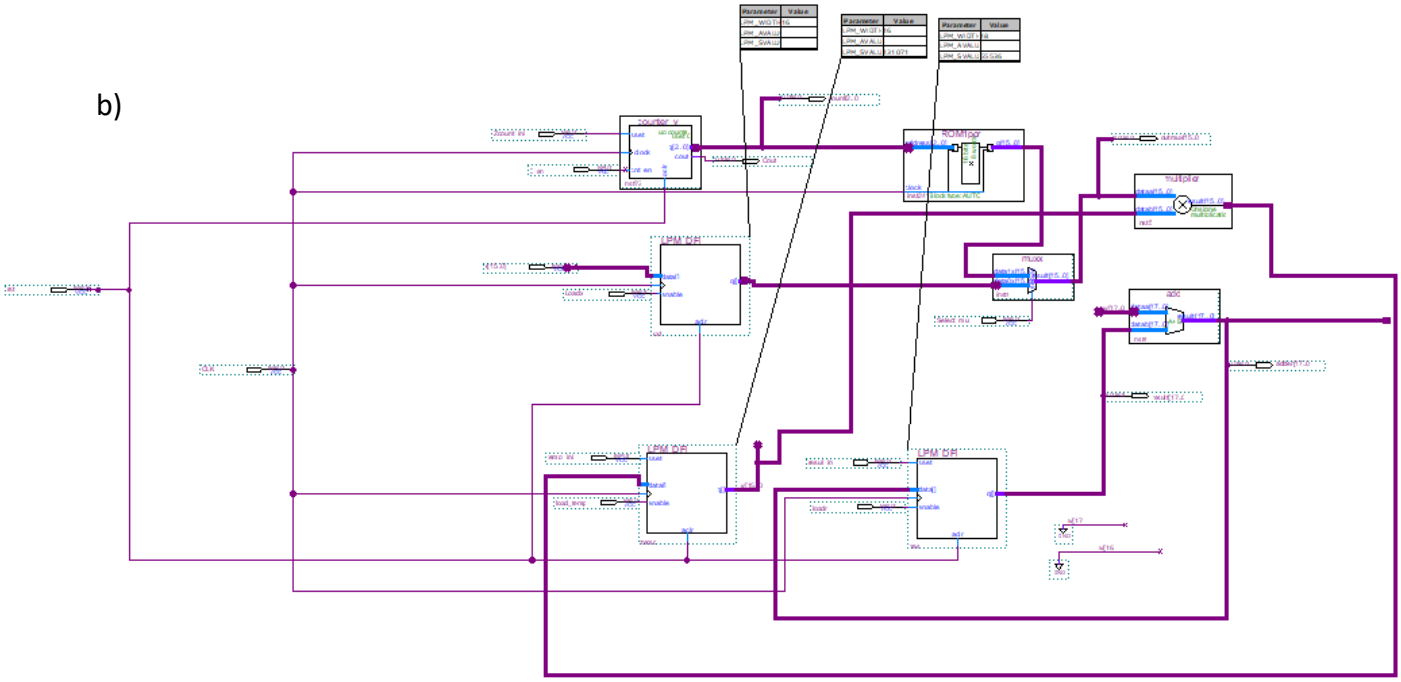
0.5	0.1000000000000000	0
0.08333333	0.0001010101010101	1
0.03333333	0.0000100010001000	2
0.0178571428	0.0000010010010010	3
0.01111111111	0.0000001011011000	4
0.00757575757575	0.0000000111110000	5
0.005494505494	0.0000000101101000	6
0.004166666666	0.0000000100010001	7

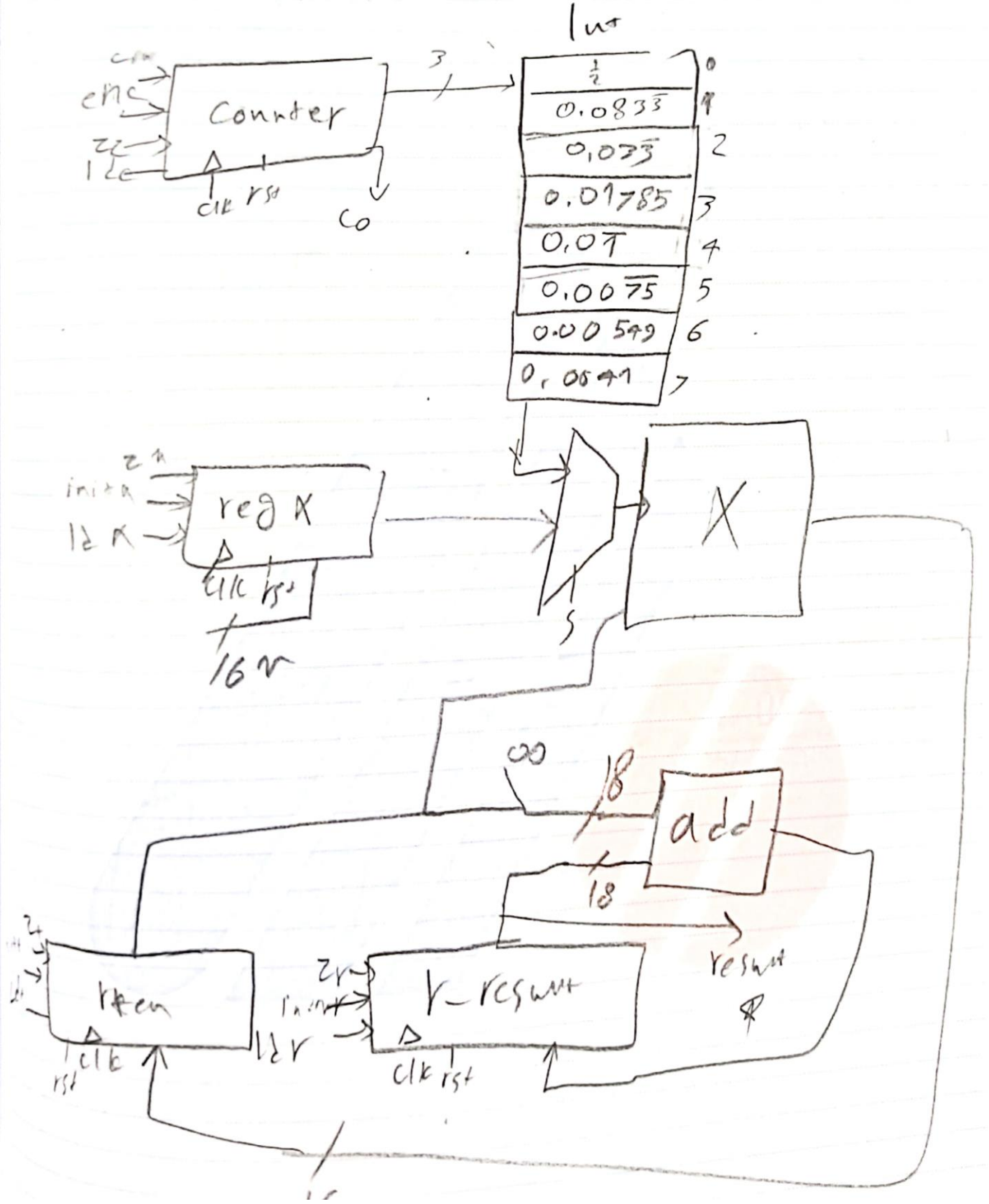
Addr	+000	+001	+010	+011	+100	+101	+110	+111	ASCII
0000	1000000000000000	0001010101010101	0000100010001000	0000010010010010	0000001011011000	0000000111110000	0000000101101000	0000000100010001	.....

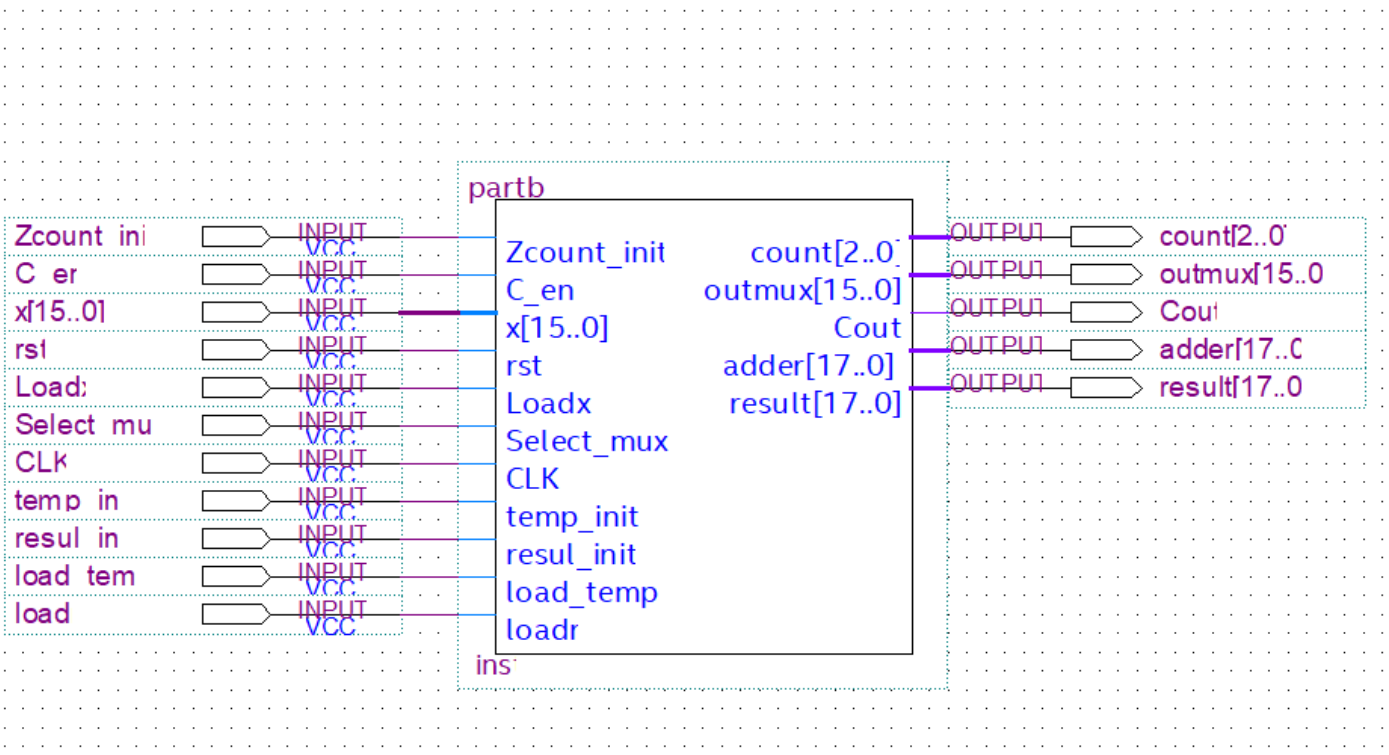


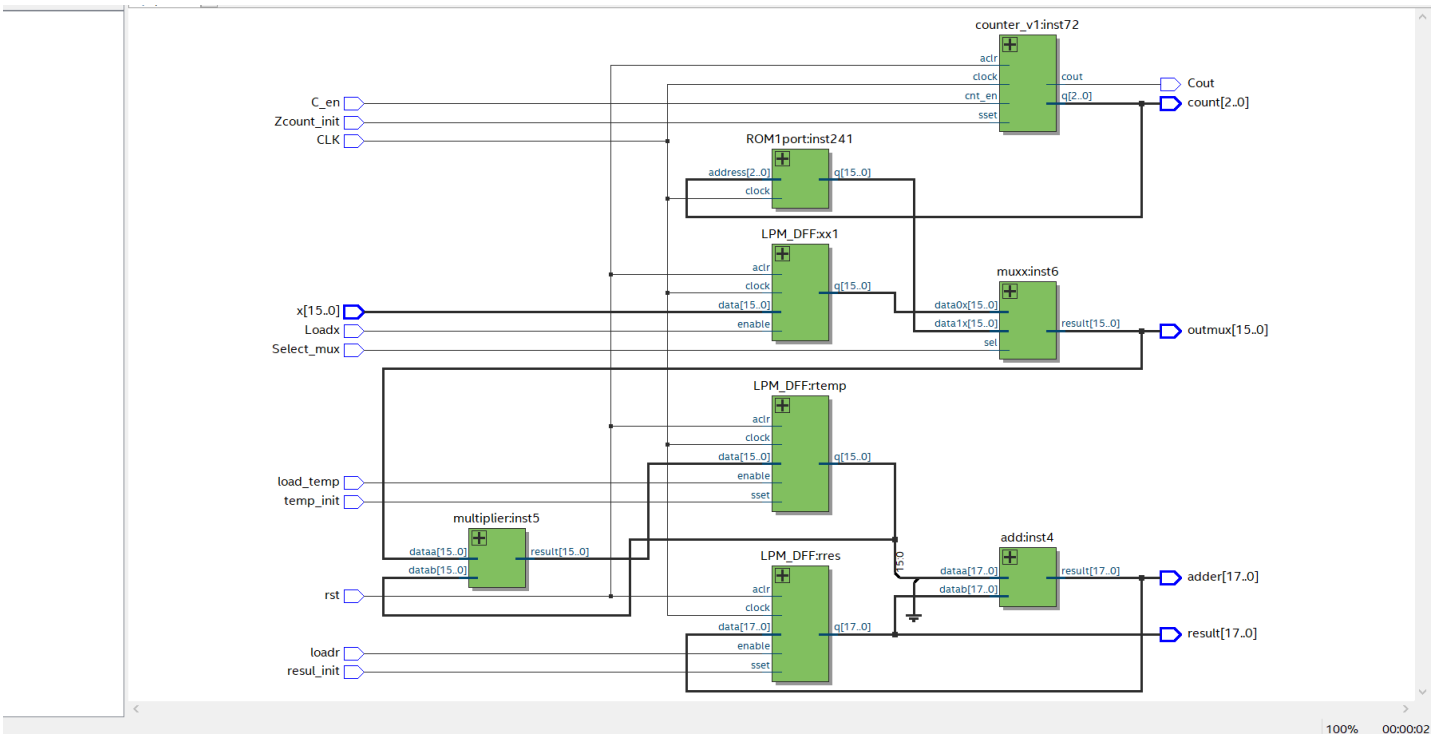

ROM1por
address[2..0]
q[15..0]
16 bits
8 words
address[2..(
INPUT
VCC
inclock
outclock
Block type: AUTC

b)



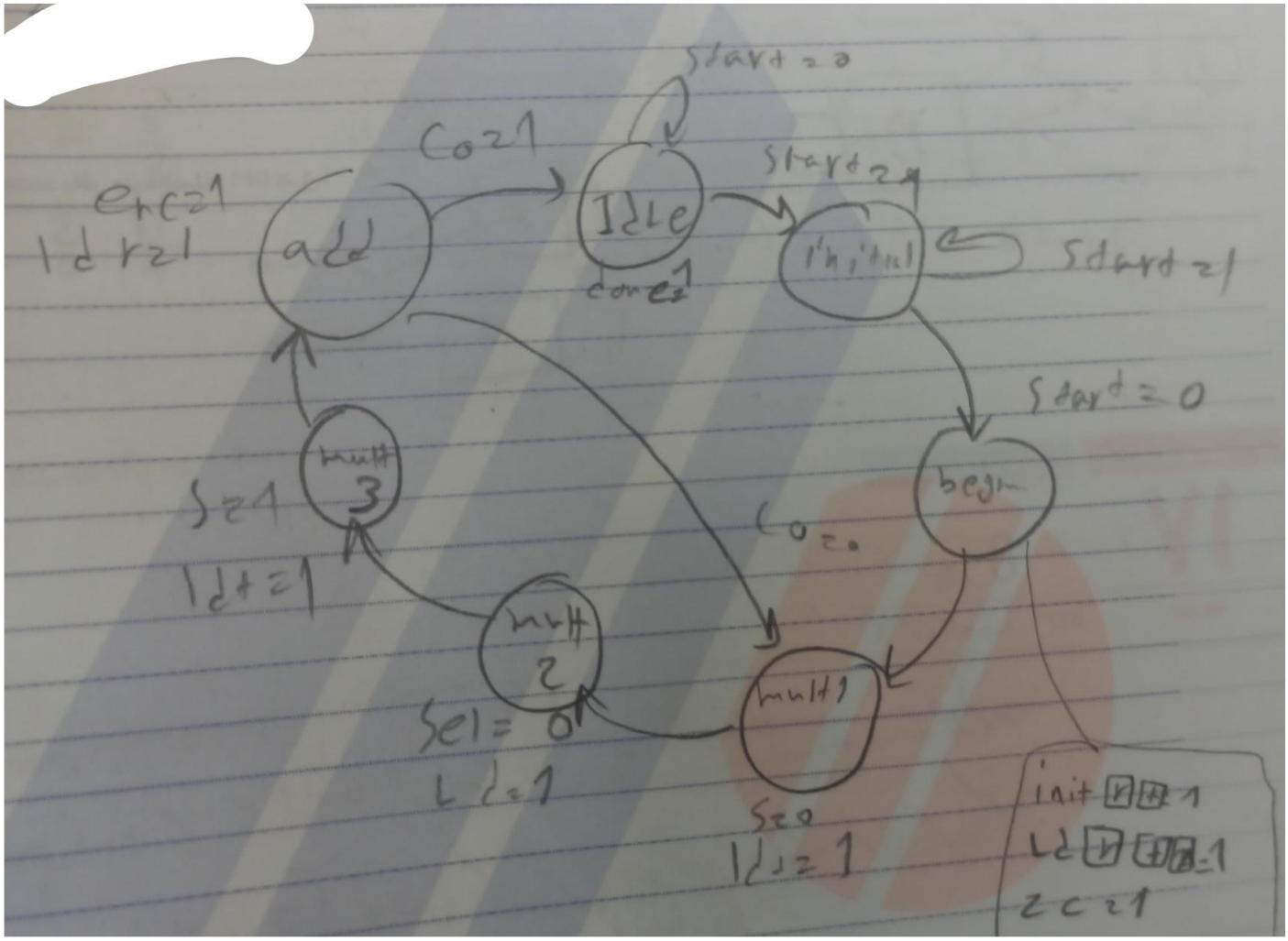

Counter
3
clk
rst
Co
0
1/2
0.0833
2
0.03571
3
0.01785
4
0.01
5
0.00575
6
0.000549
7
0.05471
reg X
16
X
5
∞
18
add
18
result
clk
rst


partb
Zcount_init
C_en
x[15..0]
rst
Loadx
Select_mux
CLK
temp_init
resul_init
load_temp
loadr
count[2..0
outmux[15..0]
Cout
adder[17..0]
result[17..0]
INPUT
VCC
OUTPUT
count[2..0
outmux[15..0
Cou
adder[17..0
result[17..0


counter_v1:inst72
aclr
clock
cnt_en
sset
cout
q[2..0]
Cout
count[2..0]
C_en
Zcount_init
CLK
ROM1port:inst241
address[2..0]
clock
q[15..0]
LPM_DFF:xx1
aclr
clock
data[15..0]
enable
q[15..0]
muxx:inst6
data0x[15..0]
data1x[15..0]
sel
result[15..0]
outmux[15..0]
x[15..0]
Loadx
Select_mux
LPM_DFF:rtemp
load_temp
temp_init
sset
multiplier:inst5
dataa[15..0]
datab[15..0]
rst
add:inst4
dataa[17..0]
datab[17..0]
result[17..0]
adder[17..0]
LPM_DFF:rres
data[17..0]
q[17..0]
loadr
resul_init
100%
00:00:02

c)

state machine diagram:



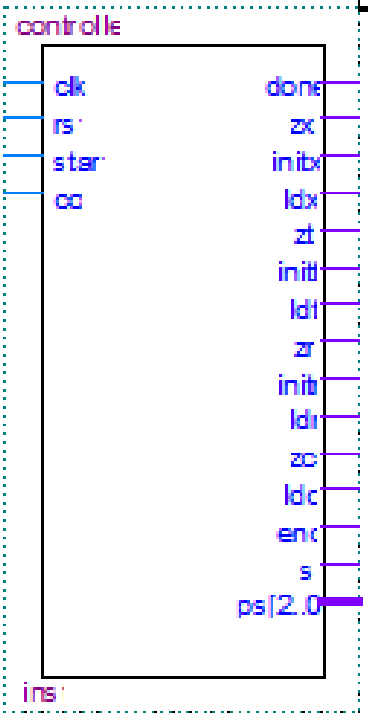
b_cosh.sv controller.sv

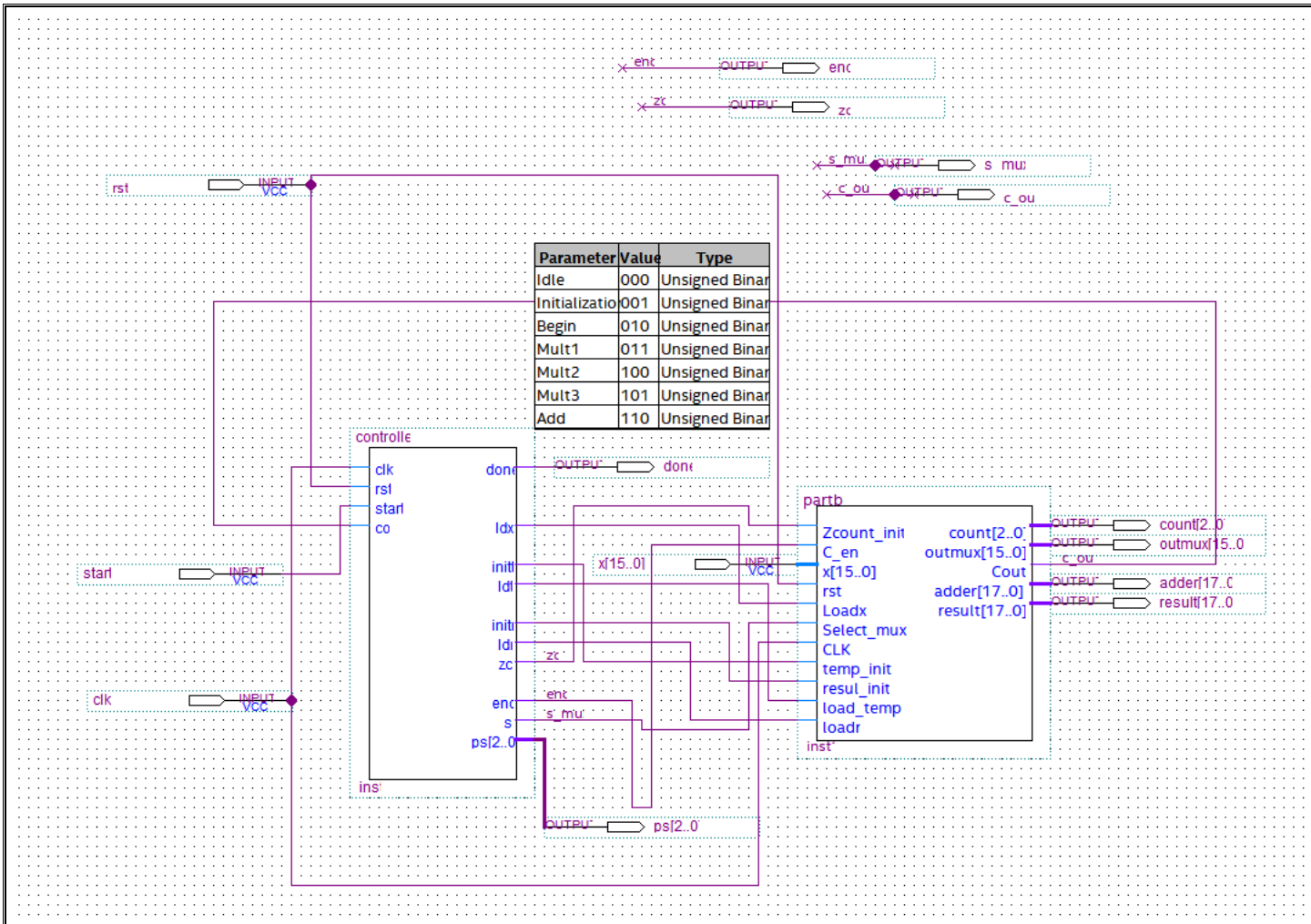
Users > M.M > Desktop > CA6 > controller.sv

```
module controller(input clk,rst,start,co, output reg done,zx,initx,ldx,zt,initt,ldt,zr
  reg [2:0] ns;
  parameter [2:0] Idle = 3'd0, Initialization = 3'd1, Begin = 3'd2, Mult1 = 3'd3, Mult
  always@(ps,co,start) begin
    ns = Idle;
    case(ps)
      Idle: ns = (start)? Initialization:Idle;
      Initialization: ns = (start)? Initialization:Begin;
      Begin: ns = Mult1;
      Mult1: ns = Mult2;
      Mult2: ns = Mult3;
      Mult3: ns = Add;
      Add: ns = (co)? Idle:Mult1;
    endcase
  end

  always@(ps,co,start) begin
  {done,zx,initx,ldx,zt,initt,ldt,zr,initr,ldr,zc,ldc,enc,s}=14'b00000000000000000;
    case(ps)
      Idle: done = 1'b1;
      Begin: begin
        initr = 1'b1;
        ldr = 1'b1;
        initt = 1'b1;
        ldt = 1'b1;
        zc = 1'b1;
        ldx = 1'b1;
      end
      Mult1: begin
        s = 1'b0;
        ldt = 1'b1;
      end
      Mult2: begin
        s = 1'b0;
        ldt = 1'b1;
      end
      Mult3: begin
        s = 1'b1;
```

```
            zc = 1'b1;
            ldx = 1'b1;
          end
          Mult1: begin
            s = 1'b0;
            ldt = 1'b1;
          end
          Mult2: begin
            s = 1'b0;
            ldt = 1'b1;
          end
          Mult3: begin
            s = 1'b1;
            ldt = 1'b1;
          end
          Add: begin
            enc = 1'b1;
            ldr = 1'b1;
          end
        endcase
      end

      always@(posedge clk, posedge rst) begin
        if (rst) ps <= Idle;
        else ps <= ns;
      end
    endmodule
```

Parameter	Value	Type
Idle	000	Unsigned Binar
Initializatio	001	Unsigned Binar
Begin	010	Unsigned Binar
Mult1	011	Unsigned Binar
Mult2	100	Unsigned Binar
Mult3	101	Unsigned Binar
Add	110	Unsigned Binar





Parameter	Value	Type
Idle	000	Unsigned Binar
Initializatio	001	Unsigned Binar
Begin	010	Unsigned Binar
Mult1	011	Unsigned Binar
Mult2	100	Unsigned Binar
Mult3	101	Unsigned Binar
Add	110	Unsigned Binar

Testbench:

```
`timescale 1ns/1ns
module tb_coshx();
  reg clk = 1'b0;
  reg rst = 1'b0;
  reg start = 1'b0;
  reg [15:0]x = 16'b1000000000000000;
  wire done;
  wire [17:0]result,ADDER;
  wire [15:0]sel_output;
  wire [2:0] counting;
  wire co_coun,sel_num,enc,zc;
  wire [2:0] ps;
  partddd12d cut(done,clk,co_coun,zc,enc,rst,sel_num,x,start,ADDER,counting,sel_output,ps,result);
  /*
output  done;
input   clk;
output  c_out;
output  zc;
output  enc;
input   rst;
output  s_mux;
input   [15:0] x;
input   start;
output  [17:0] adder;
output  [2:0] count;
output  [15:0] sel_output;
output  [2:0] ps;
output  [17:0] result;
  */


  always #100 clk=~clk;
  initial begin
    #100 rst = 1'b1;
```

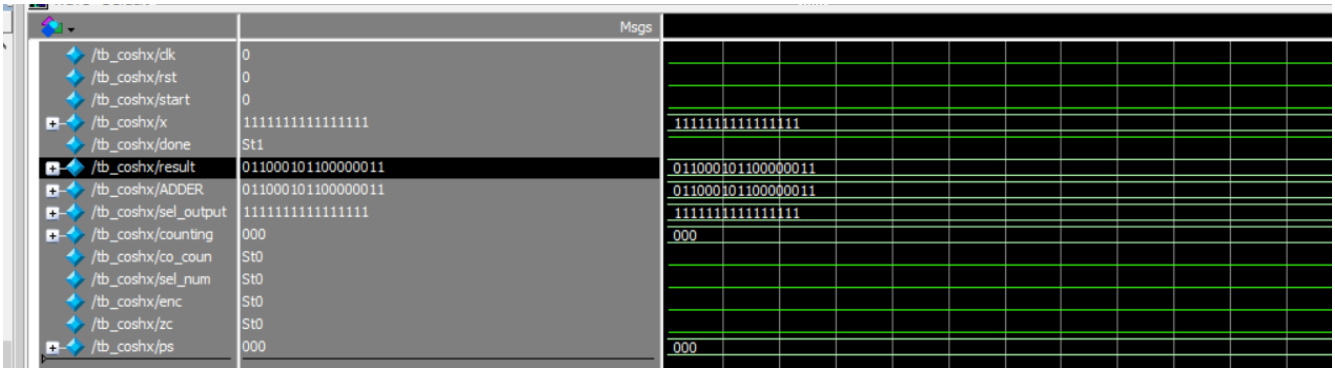
```
    always #100 clk=~clk;
    initial begin
      #100 rst = 1'b1;
      #100
      rst = 1'b0;
      x = 16'b1000000000000000;
      #520
      start = 1'b1;
      #370
      start = 1'b0;
      x = 16'b1000000000000000;
      #8000
      //x = 16'b1011101110011001;
      start = 1'b1;
      #100
       start = 1'b0;
      //#562
      //start = 1'b1;
      #8000
      //start = 1'b0;
      //x = 16'b0000000000000000;
      #10
      #100 rst = 1'b1;
      #100
      rst = 1'b0;
      x = 16'b1111111111111111;
      #520
      start = 1'b1;
      #370
      start = 1'b0;
      #8000
      $stop;
    end
endmodule
```

test 1:

X=1

Cosh(1)=1.54308063482

Result of devise : 1.5430145263671875

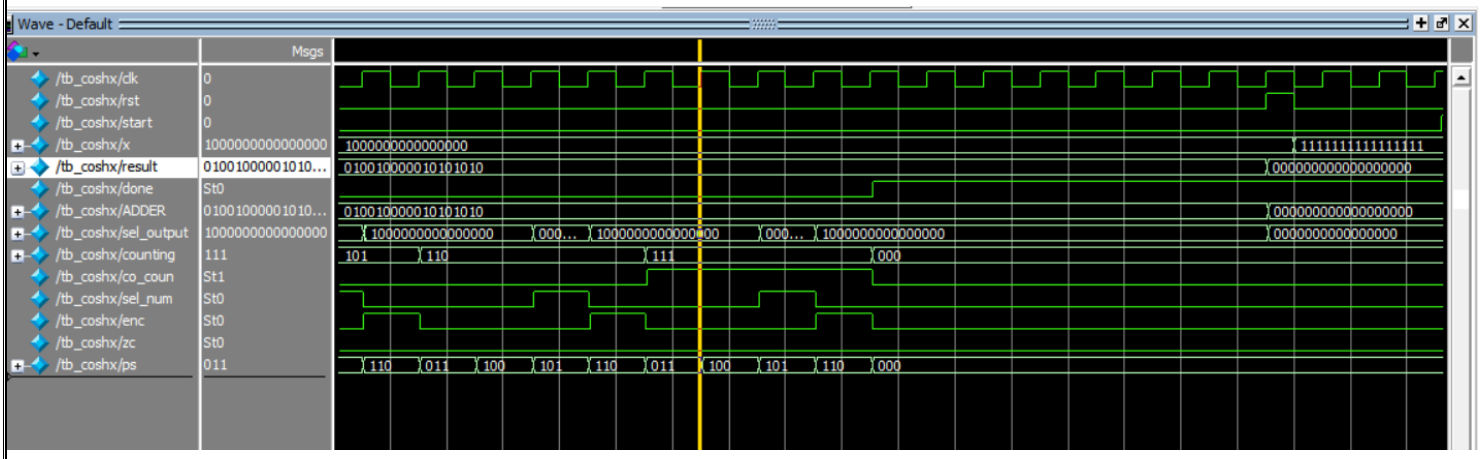


test 2:

X=0.5

Cosh(0.5)=1.1276259

Result= 1.12744140625



test 3:

X=0.8

Cosh(0.8)= 1.3374349463

Result= 1.33740234375