# Digital Systems I

## Computer Assignment #3

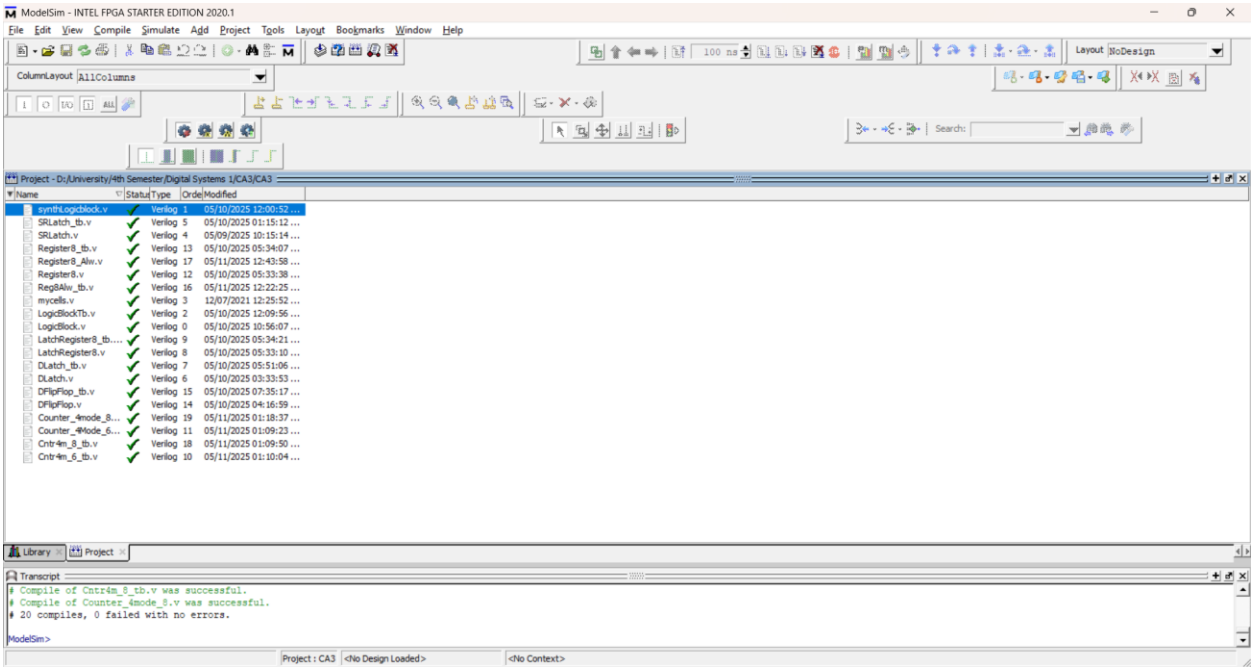
***Logic block synthesis, Latches and flip flops, Clocked feedback***

Amirali Dehghani

SID: 810102443

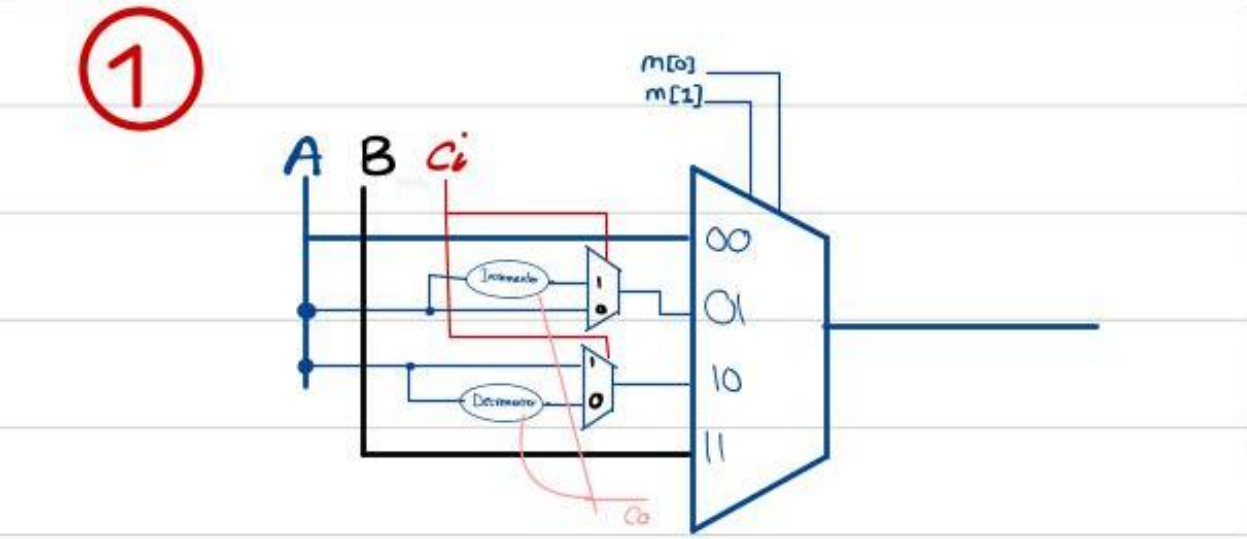
# Introduction

This is an image of the project that I have created in ModelSim for the simulation of my circuit.



# Question 1

## Drawing circuit on paper

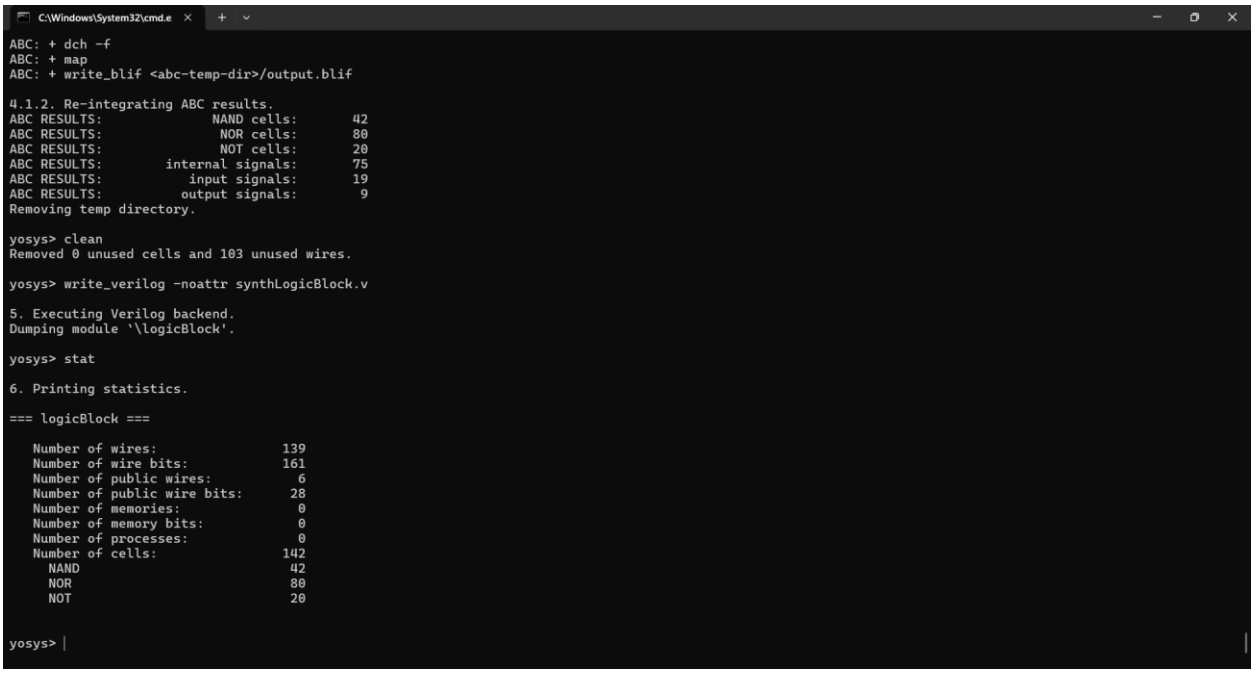



## Code

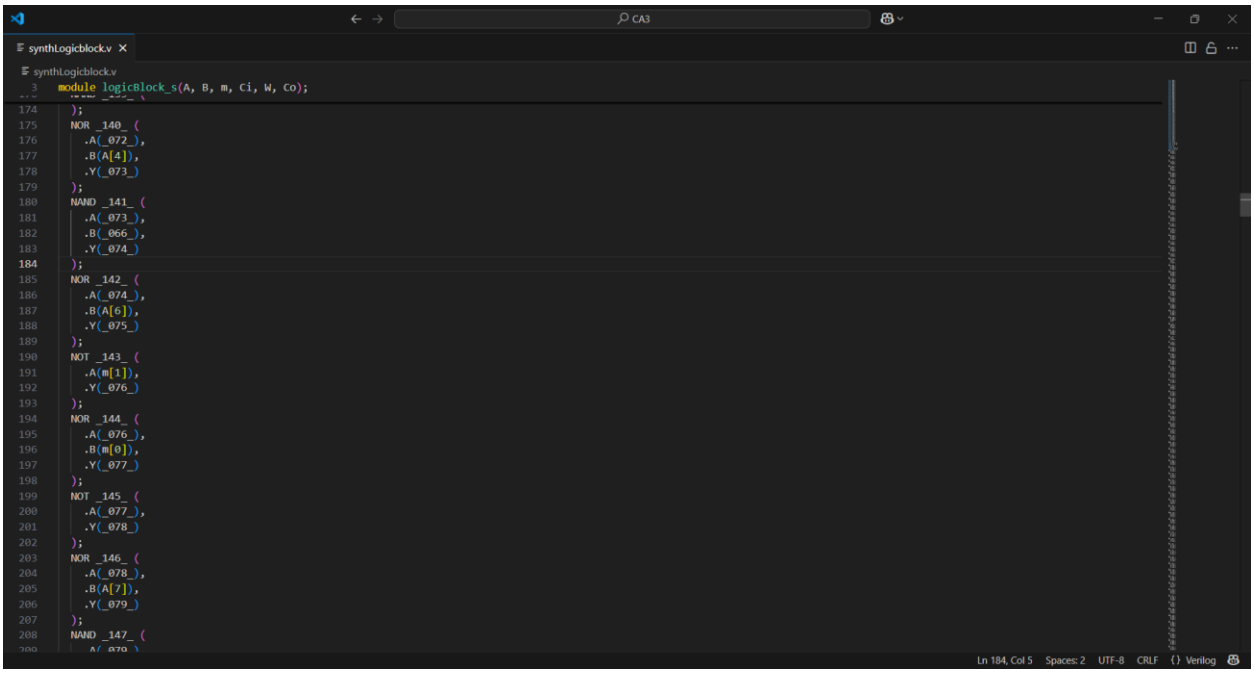
```
`timescale 1ns/1ns
module DLatch(input D, clk, rst, output wire Q, Qb);
wire Db, i, j, Q1, Q2, rst_bar;
not #(6) not1(Db, D), not2 (rst_bar, rst), not3(Q, Q2);
nand #(8) n1(i, D, clk), n2(j, Db, clk), n3(Q1, i, Qb), n4(Qb, j, Q1),
n5(Q2, Q1, rst_bar);
endmodule
```

# Question 2

## Yosys




*Yosys command line environment*

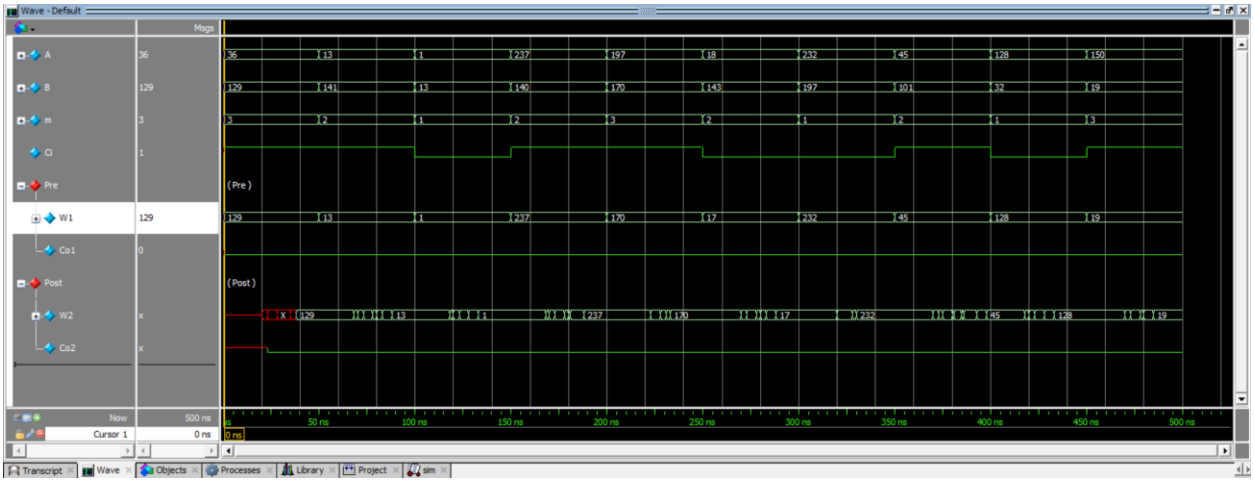



*Part of the synthesised output from Yosys*

## Code

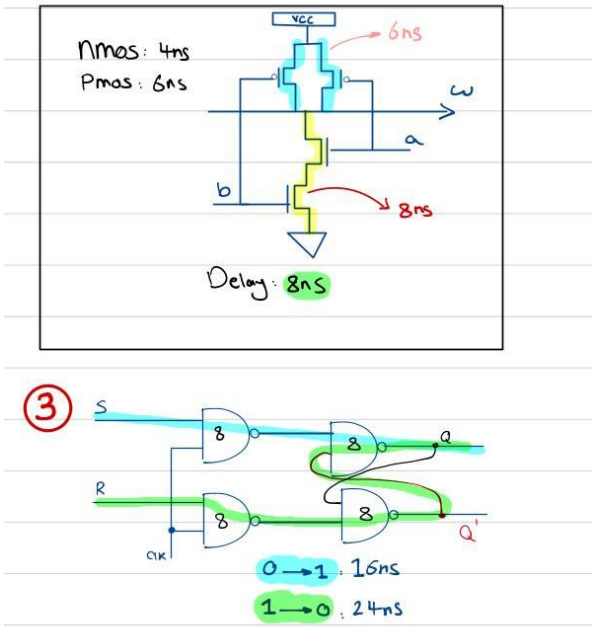
```
`timescale 1ns/1ns
module logicBlockTb();
    reg [7:0] A,B;
    reg [1:0] m;
    reg Ci;
    wire [7:0] W1,W2;
    wire Co1,Co2;

    logicBlock uut1(.A(A),.B(B),.W(W1),.Ci(Ci),.Co(Co1),.m(m));
    logicBlock_s uut2(.A(A),.B(B),.W(W2),.Ci(Ci),.Co(Co2),.m(m));

    initial begin
        Ci=0;A=0;B=0;m=0;
        repeat(10) begin
            A = $random % (2**8);
            B = $random % (2**8);
            Ci = $random % 2;
            m = $random % 4;
            #50;
        end
        $stop;
    end
endmodule
```

## Output



# Question 3

## Drawing circuit and calculating its delay




## Code

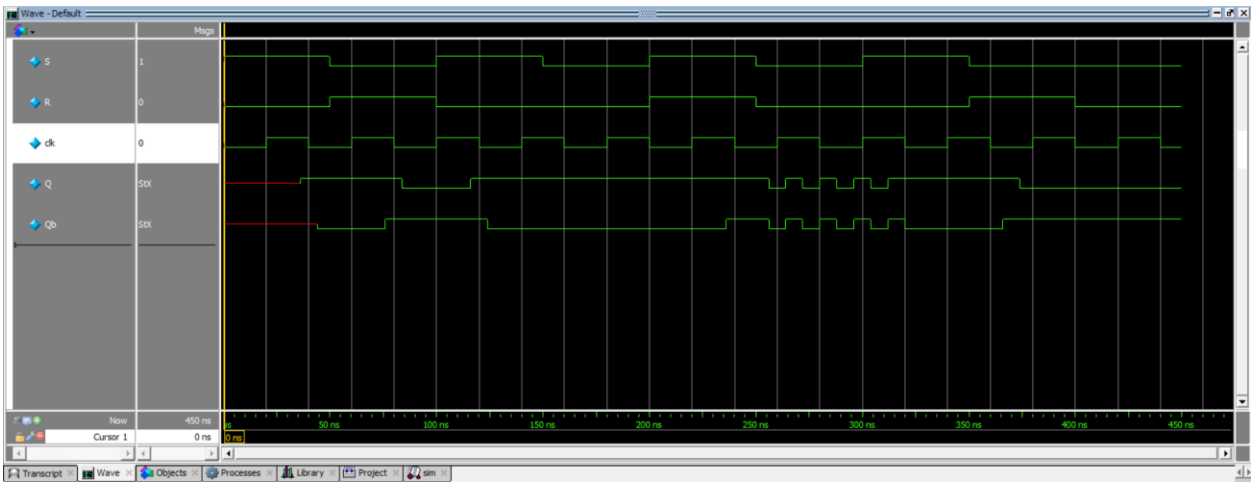
```
`timescale 1ns/1ns
module SRLatch(input S, R, clk, output wire Q, Qb);
wire S1, R1;
nand #(8) n1(S1, S, clk), n2(R1, R, clk), n3(Q, S1, Qb), n4(Qb, R1, Q);
endmodule
```

## Testbench

```
`timescale 1ns/1ns
module TB_SRLatch();
reg S, R, clk;
wire Q, Qb;
SRLatch uut(.S(S), .R(R), .clk(clk), .Q(Q), .Qb(Qb));
initial begin
    clk = 0;
    forever #20 clk = ~clk;
end
initial begin
    S = 1; R = 0; #50;
    S = 0; R = 1; #50;
    S = 1; R = 0; #50;
    S = 0; R = 0; #50;
    S = 1; R = 1; #50;
    S = 0; R = 0; #50;
    S = 1; R = 0; #50;
```

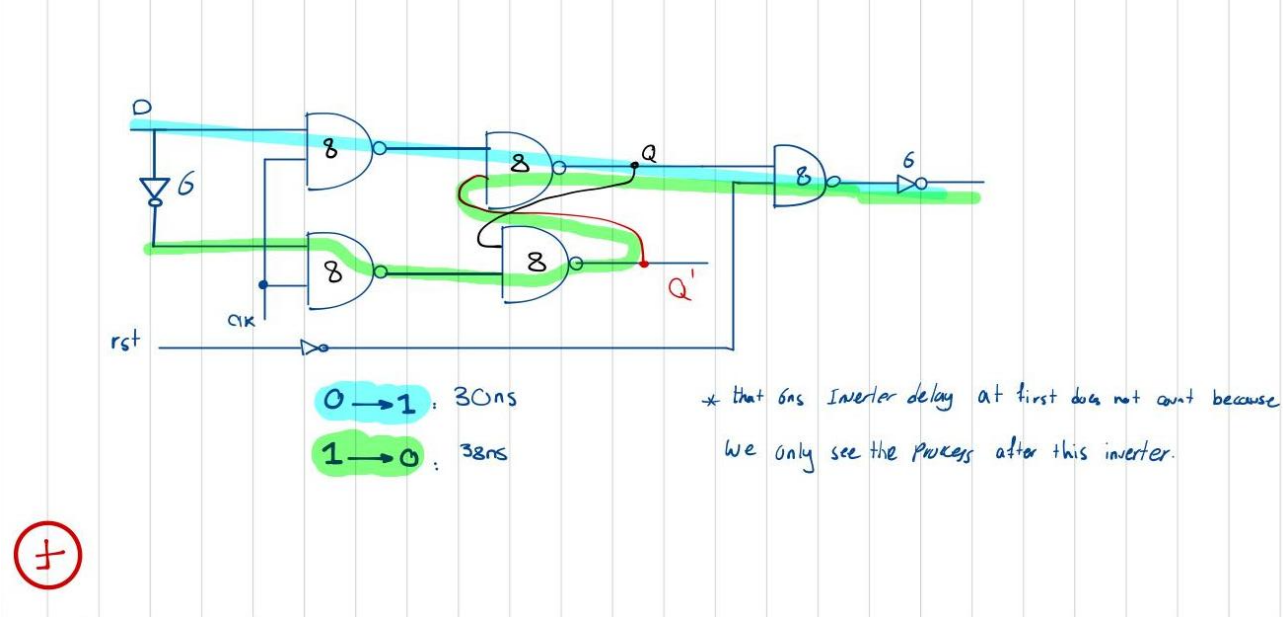
```
    S = 0; R = 1; #50;
    S = 0; R = 0; #50;
    $stop;
end
endmodule
```

## Output



# Question 4

## Drawing circuit and calculating its delay




## Code

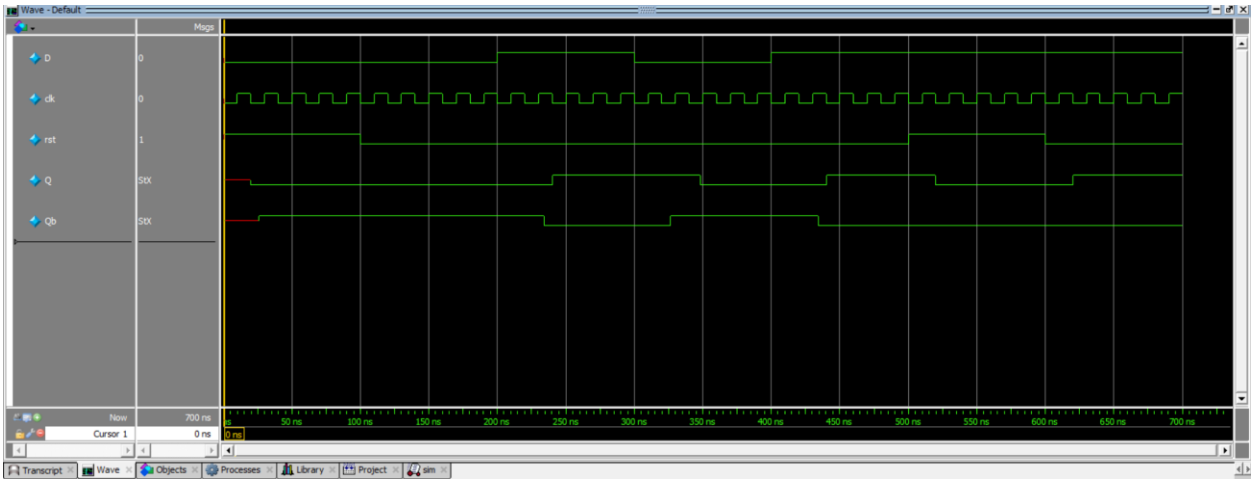
```
`timescale 1ns/1ns
module DLatch(input D, clk, rst, output wire Q, Qb);
wire Db, i, j, Q1, Q2, rst_bar;
not #(6) not1(Db, D), not2 (rst_bar, rst), not3(Q, Q2);
nand #(8) n1(i, D, clk), n2(j, Db, clk), n3(Q1, i, Qb), n4(Qb, j, Q1),
n5(Q2, Q1, rst_bar);
endmodule
```

## Testbench

```
`timescale 1ns/1ns

module TB_DLatch();
    reg D, clk, rst;
    wire Q, Qb;
    DLatch uut(.D(D), .clk(clk), .rst(rst), .Q(Q), .Qb(Qb));
    initial begin
        clk = 0;
        forever #10 clk = ~clk;
    end
    initial begin
        rst = 1;
        D = 0;
        #100;
        rst = 0;
```

```
        #100;
        D = 1; #100;
        D = 0; #100;
        D = 1; #100;
        rst = 1; #100;
        rst = 0; #100;
        $stop;
    end
endmodule
```

## Output



# Question 5

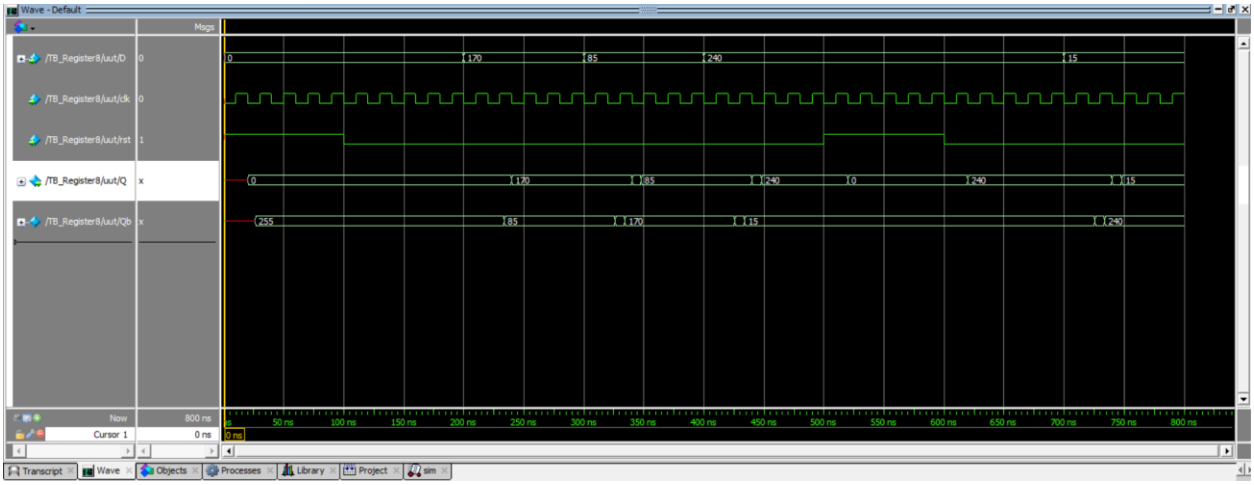
## Code

```
`timescale 1ns/1ns
module LatchRegister_8 (input [7:0] D,input clk,input rst,output wire [7:0]
Q);
    wire [7:0] Qb;
    DLatch d0(.D(D[0]), .clk(clk), .rst(rst), .Q(Q[0]), .Qb(Qb[0]));
    DLatch d1(.D(D[1]), .clk(clk), .rst(rst), .Q(Q[1]), .Qb(Qb[1]));
    DLatch d2(.D(D[2]), .clk(clk), .rst(rst), .Q(Q[2]), .Qb(Qb[2]));
    DLatch d3(.D(D[3]), .clk(clk), .rst(rst), .Q(Q[3]), .Qb(Qb[3]));
    DLatch d4(.D(D[4]), .clk(clk), .rst(rst), .Q(Q[4]), .Qb(Qb[4]));
    DLatch d5(.D(D[5]), .clk(clk), .rst(rst), .Q(Q[5]), .Qb(Qb[5]));
    DLatch d6(.D(D[6]), .clk(clk), .rst(rst), .Q(Q[6]), .Qb(Qb[6]));
    DLatch d7(.D(D[7]), .clk(clk), .rst(rst), .Q(Q[7]), .Qb(Qb[7]));
endmodule
```

## Testbench

```
`timescale 1ns/1ns
module TB_Register8();
    reg [7:0] D;
    reg clk, rst;
    wire [7:0] Q;
    LatchRegister_8 uut(.D(D), .clk(clk), .rst(rst), .Q(Q));
    initial begin
        clk = 0;
        forever #10 clk = ~clk;
    end
    initial begin
        rst = 1; D = 8'b00000000; #100;
        rst = 0; #100;
        D = 8'b10101010; #100;
        D = 8'b01010101; #100;
        D = 8'b11110000; #100;
        rst = 1; #100;
        rst = 0; #100;
        D = 8'b00001111; #100;
        $stop;
    end
endmodule
```

# Output


Wave - Default
Msgs
/TB_Register8/uut/D
/TB_Register8/uut/clk
/TB_Register8/uut/rst
/TB_Register8/uut/Q
/TB_Register8/uut/Qb
Now
Cursor 1
Transcript
Wave
Objects
Processes
Library
Project
sim

# Question 6

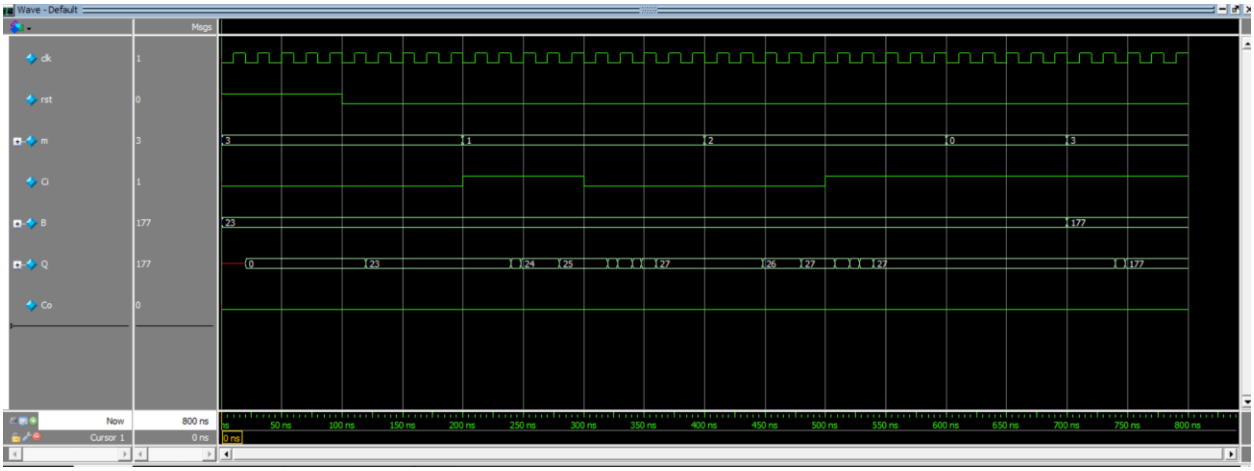
## Code

```
module Counter8bit_4m_1 (input clk, input rst, input [7:0] B, input [1:0] m,
input Ci, output [7:0] Q, output Co);
    wire [7:0] A;
    wire [7:0] W;
    wire [7:0] Qb;
    assign A = Q;
    logicBlock logic (.A(A), .B(B), .m(m), .Ci(Ci), .W(W), .Co(Co));
    LatchRegister_8 reg8 (.D(W), .clk(clk), .rst(rst), .Q(Q));
endmodule
```

## Testbench

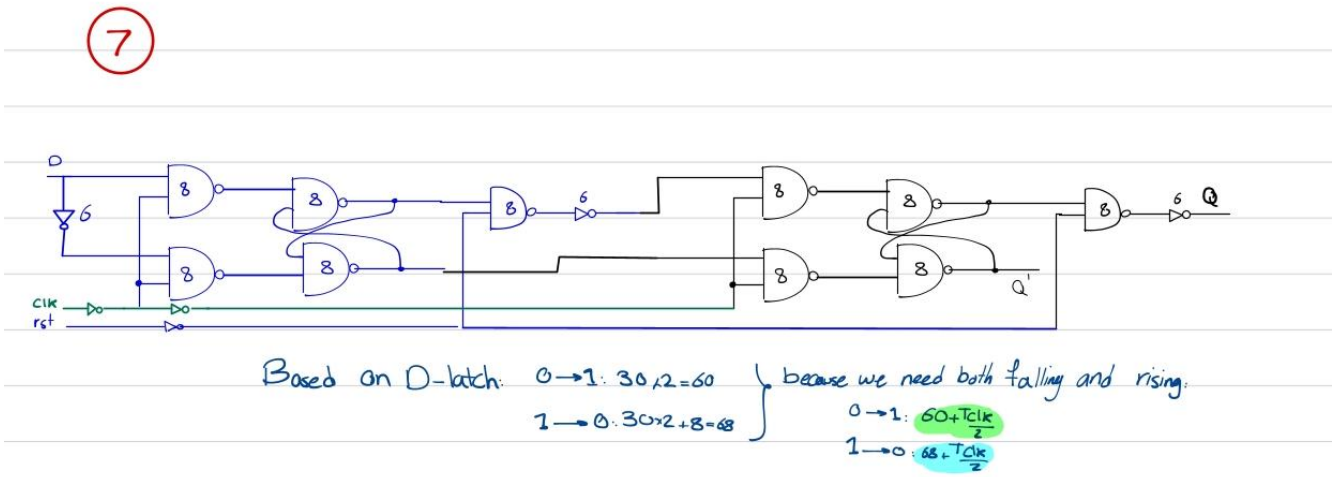
```
`timescale 1ns/1ns
module TB_Counter8bit_4m_1();
    reg clk, rst;
    reg [1:0] m;
    reg Ci;
    reg [7:0] B;
    wire [7:0] Q;

    Counter8bit_4m_1 uut(.clk(clk), .rst(rst), .B(B), .m(m), .Ci(Ci),
.Co(Co), .Q(Q));

    initial begin
        clk = 0;
        forever #10 clk = ~clk;
    end

    initial begin
        rst = 1; B = 8'b0010111; m = 2'b11; Ci = 0; #100;
        rst = 0; #100;
        m = 2'b01; Ci = 1; #100;
        m = 2'b01; Ci = 0; #100;
        m = 2'b10; Ci = 0; #100;
        m = 2'b10; Ci = 1; #100;
        m = 2'b00; #100;
        m = 2'b11; B = 8'b10110001; #100;
        $stop;
    end
endmodule
```

# Output



# Question 7

## Calculating Delay on paper




## Code

```
`timescale 1ns/1ns
module DFlipFlop(input D, clk, rst, output Q, Qb);
    wire clk_n, qm, qmb;
    not #(6) not_clk(clk_n, clk);
    DLatch master(.D(D), .clk(clk_n), .rst(1'b0), .Q(qm), .Qb(qmb));
    DLatch slave (.D(qm), .clk(clk), .rst(rst), .Q(Q), .Qb(Qb));
endmodule
```

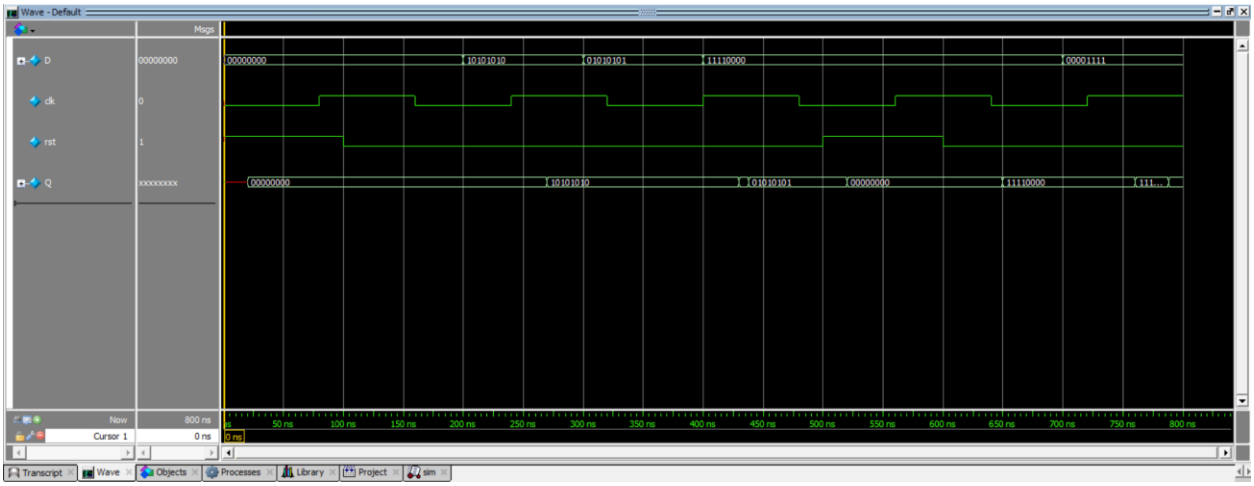
```
module Reg8FlipFlop(input [7:0] D, input clk, input rst, output [7:0] Q);
    wire [7:0] Qb;
    genvar i;
    generate
        for (i = 0; i < 8; i = i + 1) begin
            DFlipFlop dff(.D(D[i]), .clk(clk), .rst(rst), .Q(Q[i]),
.Qb(Qb[i]));
        end
    endgenerate
endmodule
```

```
`timescale 1ns/1ns
module Reg8Always(input [7:0] D, input clk, input rst, output reg [7:0] Q);
reg [7:0] temp;
    always @(posedge clk, posedge rst) begin
        if (rst) temp = 8'b0;
        else
            temp = D;
```

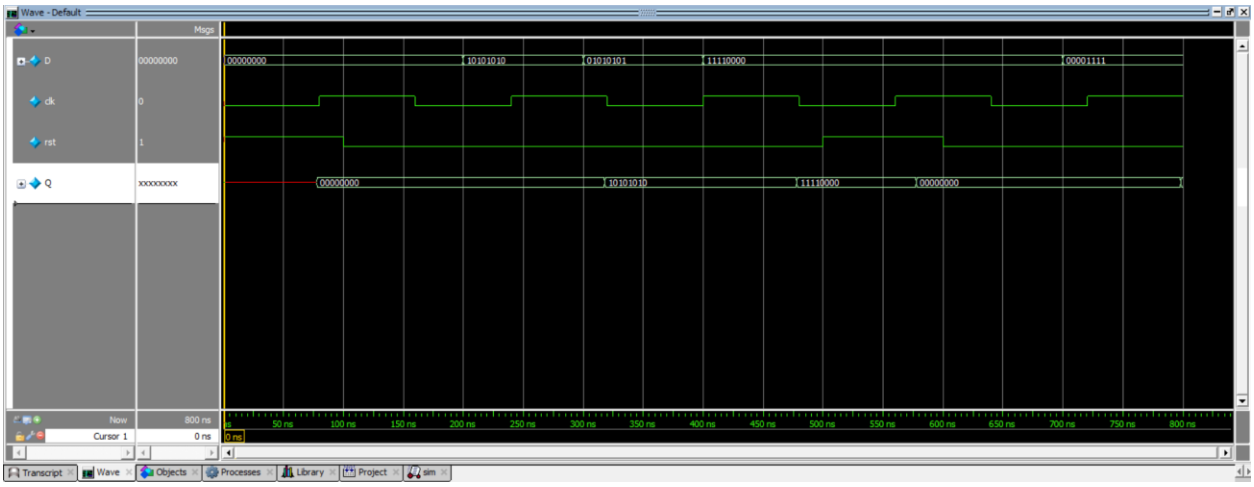
```
        Q <= #78 temp;
    end
endmodule
```

## Testbench

## Output




*FlipFlop*




*Always statement*

# Question 8

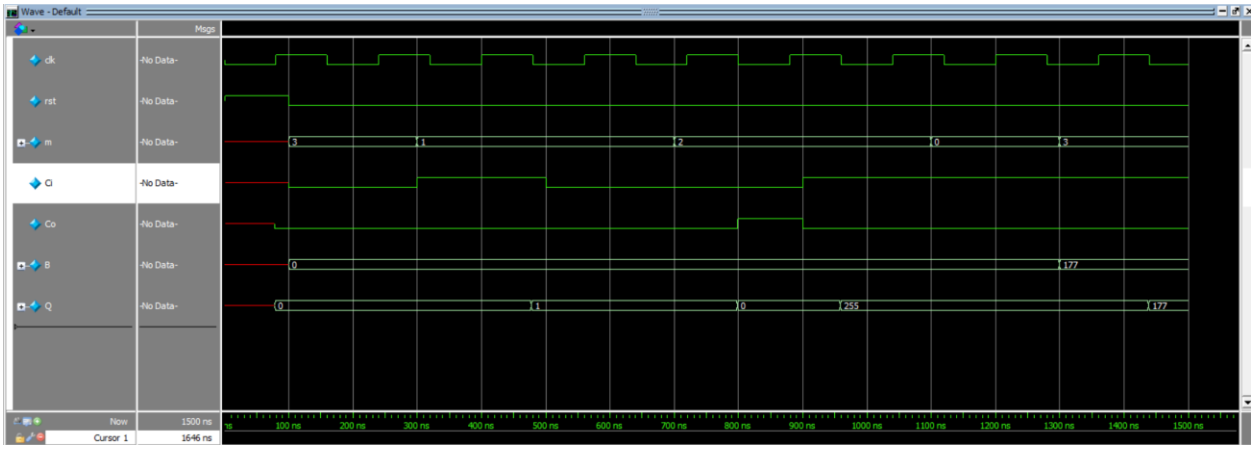
## Code

```
module Counter8bit_4m_2 (input clk, input rst, input [7:0] B, input [1:0] m,
input Ci, output [7:0] Q, output Co);
    wire [7:0] A;
    wire [7:0] W;
    wire [7:0] Qb;
    assign A = Q;
    logicBlock logic (.A(A), .B(B), .m(m), .Ci(Ci), .W(W), .Co(Co));
    Reg8Always reg8 (.D(W), .clk(clk), .rst(rst), .Q(Q));
endmodule
```

## Testbench

```
`timescale 1ns/1ns
module TB_Counter8bit_4m_2();
    reg clk, rst;
    reg [1:0] m;
    reg Ci;
    wire Co;
    reg [7:0] B;
    wire [7:0] Q;

    Counter8bit_4m_2 uut(.clk(clk), .rst(rst), .B(B), .m(m), .Ci(Ci),
.Co(Co), .Q(Q));

    initial begin
        clk = 0;
        forever #80 clk = ~clk;
    end

    initial begin
        rst = 1; #100
        rst = 0;
        B = 8'b0000000; m = 2'b11; Ci = 0; #200;
        m = 2'b01; Ci = 1; #200;
        m = 2'b01; Ci = 0; #200;
        m = 2'b10; Ci = 0; #200;
        m = 2'b10; Ci = 1; #200;
        m = 2'b00; #200;
        m = 2'b11; B = 8'b10110001; #200;
        $stop;
    end
endmodule
```

## Output




# Question 9

## Code

```
module Counter16bit (input clk, input rst, input [15:0] B, input [1:0] m,
input Ci, output [15:0] Q, output Co);
    wire [7:0] Qhi, Qlo;
    wire CoLo;
    Counter8bit_4m_2 low (.clk(clk), .rst(rst), .B(B[7:0]), .m(m), .Ci(Ci),
.Co(CoLo), .Q(Qlo));
    Counter8bit_4m_2 high (.clk(clk), .rst(rst), .B(B[15:8]), .m(m),
.Ci(CoLo), .Co(Co), .Q(Qhi));
    assign Q = {Qhi, Qlo};
endmodule
```