UNIVERSITY OF TEHRAN

College of Engineering

Electrical and Computer Engineering Department

Digital Systems I, ECE 894

Computer Assignment 5

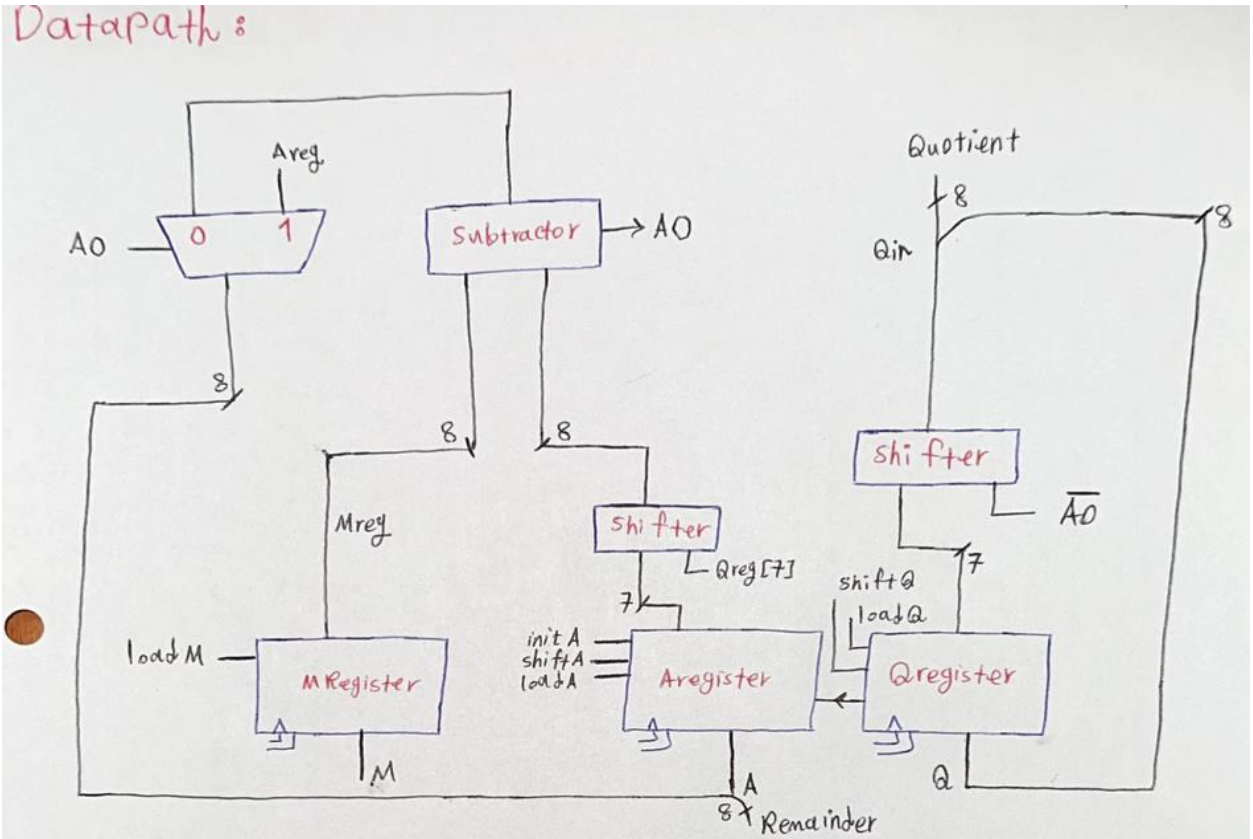
Shahaboddin Sheybani

810101454

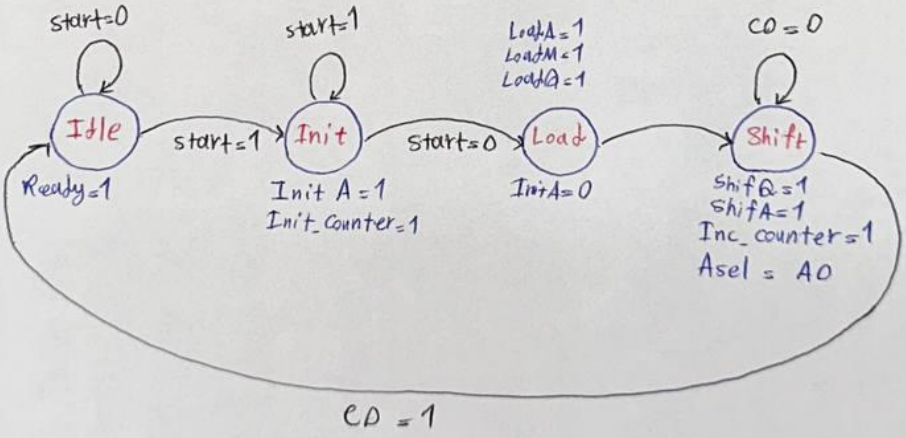
Spring 1402-03

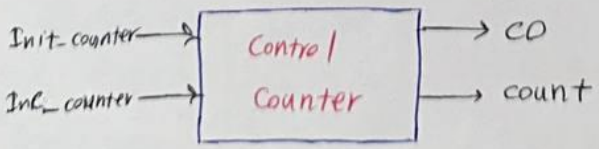
Design:

Datapath:



Controller:



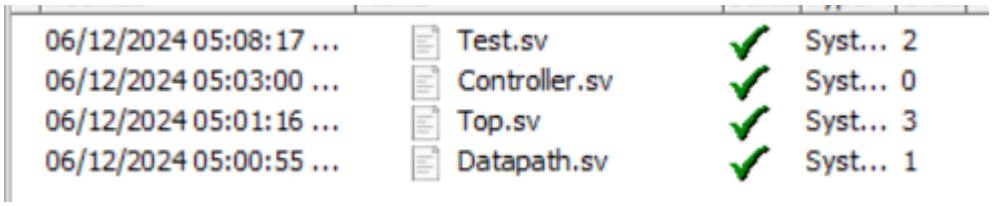


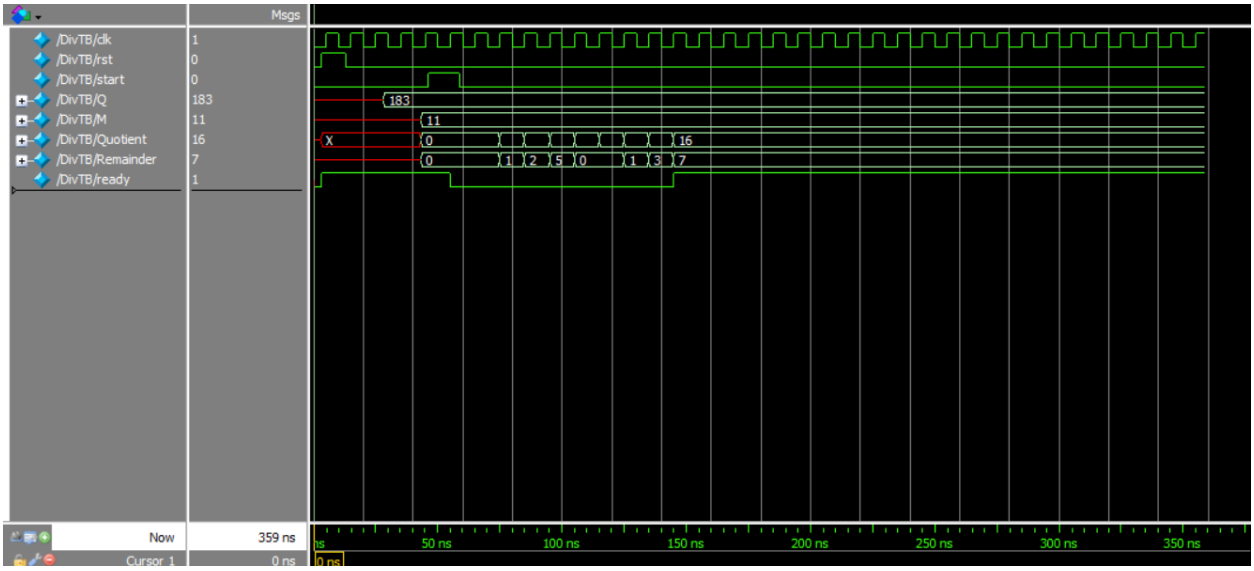
Pre-Synthesis

```
`timescale 1ns/1ns
module DivDp (input [7:0]Q, M, input loadM, loadA, loadQ, InitA, Asel, shiftA, shiftQ, clk, rst, output logic [7:0]Quotient , output logic [7:0] Remainder, output logic A0);

logic [7:0] Qreg, Mreg, Qin, A,Ain, Areg, Ares, SubResult, MUXResult;

always@ (posedge clk, posedge rst) begin

    if(rst) begin
       Areg <= 8'b0;
end
else if(InitA) begin
           A <= 8'b0;
             end
    else if(loadA) begin
           Areg <= A;
                       end
    else if(shiftA) begin
              Areg <=  Ares;
                       end
end

always @ (posedge clk, posedge rst) begin
    if(rst)
       Qreg <= 8'b0;
    else begin
        if(loadQ)
           Qreg <= Q;
        else
           if(shiftQ)
               Qreg <= Qin;
    end
end

always @(posedge clk, posedge rst) begin
    if(rst)
       Mreg <= 8'b0;
    else begin
         if(loadM)
           Mreg <= M;
    end
```

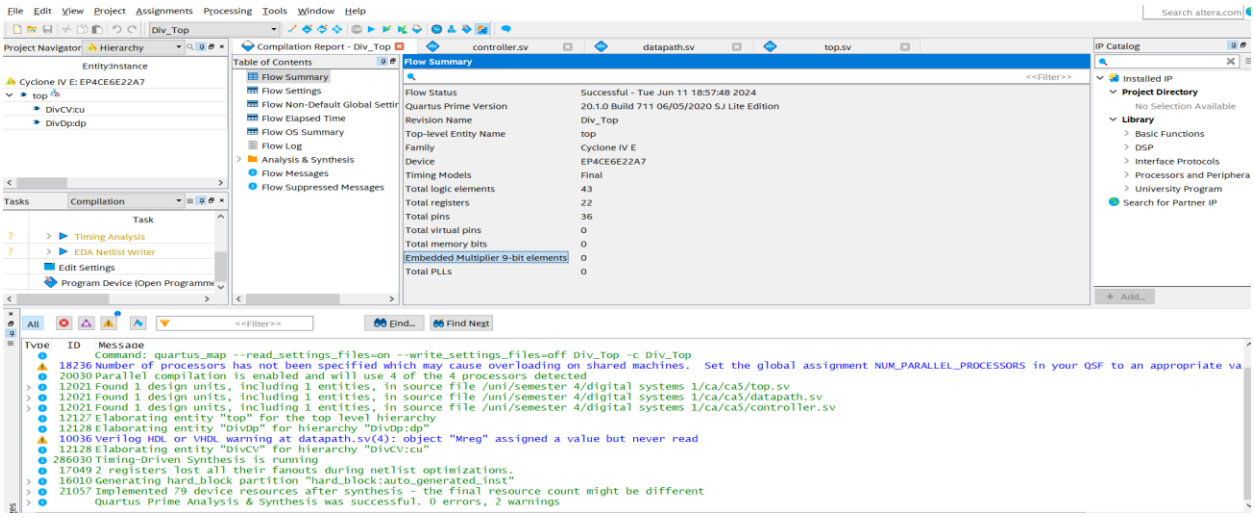
```
`timescale 1ns/1ns

module DivCV (input clk, rst, start, A0, output logic ready, loadA, loadM, loadQ, shiftA, shiftQ, InitA, Asel);
logic Co;
logic Init_Counter, Inc_Counter;
logic [1:0]pstate, nstate;
logic [2:0] Count;

parameter [1:0]
Idle = 0, Init =1, Load =2, Shift=3;
always @ (pstate , start, A0, Co) begin
    nstate =0;
    {loadA, loadM, loadQ, shiftA, shiftQ, InitA, Asel,ready} = 8'b0;
    {Init_Counter, Inc_Counter} = 2'b0;
    case (pstate)
    Idle: begin nstate = start ? Init : Idle; ready = 1'b1; end
    Init: begin nstate = start ? Init:Load; Init_Counter= 1'b1; InitA= 1'b1; end
    Load: begin nstate = Shift; InitA= 1'b0;loadM=1'b1; loadQ = 1'b1; loadA = 1'b1; end
    Shift:begin nstate = Co ?Idle :Shift; shiftQ= 1'b1; shiftA= 1'b1; Inc_Counter= 1'b1; Asel= A0; end
    default: nstate=Idle;
endcase
end

always @(posedge clk, posedge rst) begin
    if(rst)
       pstate <= Idle;
    else
       pstate <= nstate;
end

always @(posedge clk, posedge rst) begin
    if(rst) Count<=3'b0;
    else
    if(Init_Counter) Count <= 3'b1;
    else
    if(Inc_Counter) Count <= Count+1'b1;
end

assign Co = &Count;
```

```
`timescale 1ns/1ns
module top (input clk, rst, start, input[7:0] Q, M, output[7:0] Quotient, Remainder, output ready);
logic loadA, loadM, loadQ, shiftA, shiftQ, InitA, Asel, A0;
DivDp dp (Q, M, loadM, loadA, loadQ, InitA, Asel, shiftA, shiftQ, clk, rst, Quotient, Remainder, A0);
DivCV cu (clk, rst, start, A0, ready, loadA, loadM, loadQ, shiftA, shiftQ, InitA, Asel);
endmodule
```


06/12/2024 05:08:17 ... Test.sv Syst... 2
06/12/2024 05:03:00 ... Controller.sv Syst... 0
06/12/2024 05:01:16 ... Top.sv Syst... 3
06/12/2024 05:00:55 ... Datapath.sv Syst... 1

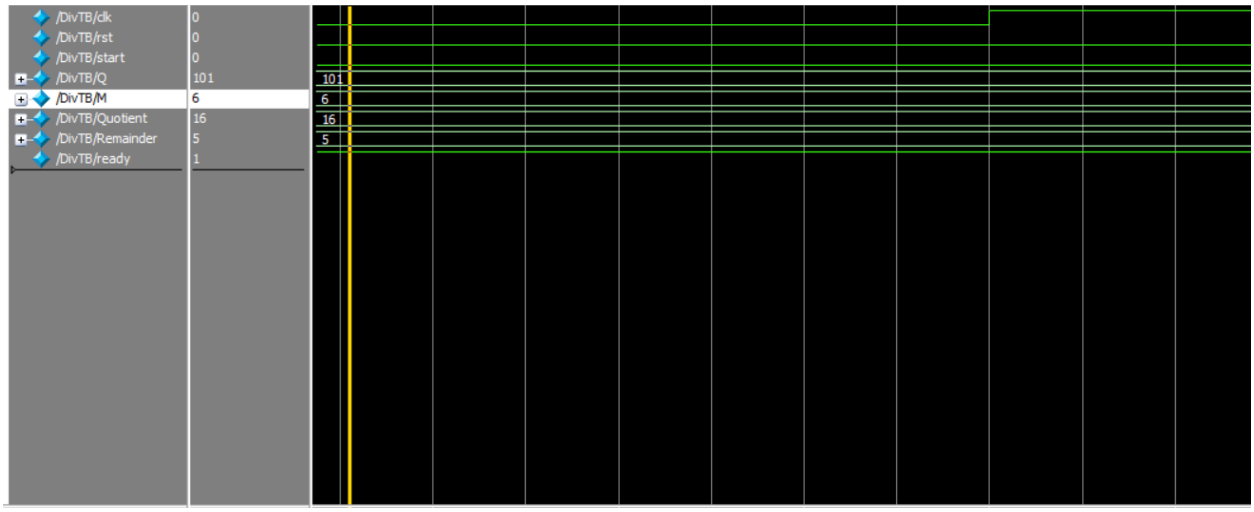

Msgs
/DivTB/clk
/DivTB/rst
/DivTB/start
/DivTB/Q
/DivTB/M
/DivTB/Quotient
/DivTB/Remainder
/DivTB/ready
1
0
0
183
11
16
7
1
Now
359 ns
Cursor 1
0 ns
50 ns
100 ns
150 ns
200 ns
250 ns
300 ns
350 ns

Post-Synthesis:



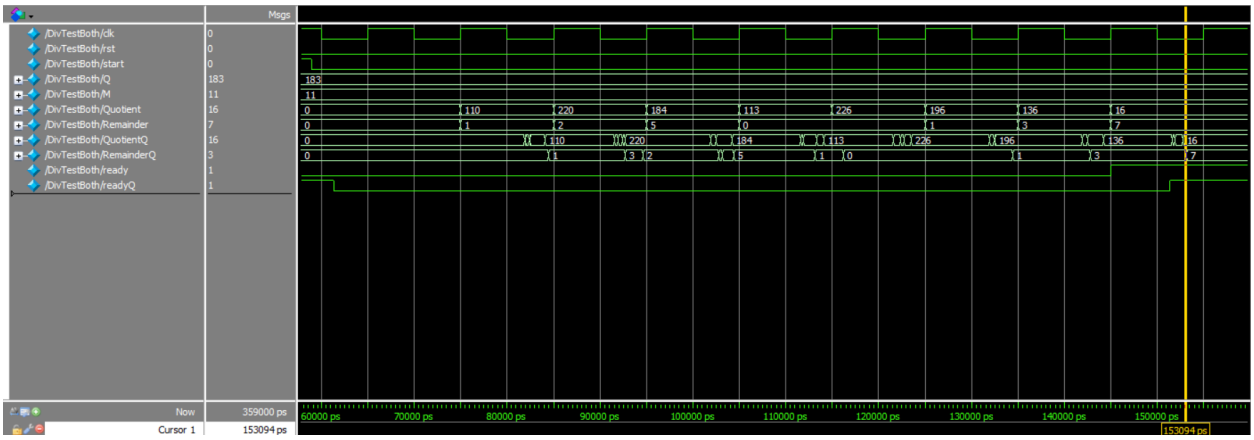

```
701         .cout());
702 // synopsys translate_off
703 defparam \cu|Selector2~0 .lut_mask = 16'h70F0;
704 defparam \cu|Selector2~0 .sum_lutc_input = "datac";
705 // synopsys translate_on
706
707 // Location: LCCOMB_X28_Y20_N14
708 cycloneive_lcell_comb \cu|Selector2~1 (
709 // Equation(s):
710 // \cu|Selector2~1_combout  = (\cu|pstate.Load~q ) # (\cu|Selector2~0_combout )
711
712         .dataa(gnd),
713         .datab(gnd),
714         .datac(\cu|pstate.Load~q ),
715         .datad(\cu|Selector2~0_combout ),
716         .cin(gnd),
717         .combout(\cu|Selector2~1_combout ),
718         .cout());
719 // synopsys translate_off
720 defparam \cu|Selector2~1 .lut_mask = 16'hFFF0;
721 defparam \cu|Selector2~1 .sum_lutc_input = "datac";
722 // synopsys translate_on
723
724 // Location: FF_X28_Y20_N15
725 dffeas \cu|pstate.Shift (
726         .clk(\clk~inputclkctrl_outclk ),
727         .d(\cu|Selector2~1_combout ),
728         .asdata(vcc),
729         .clrn(!\rst~inputclkctrl_outclk ),
730         .aload(gnd),
731         .sclr(gnd),
732         .sload(gnd)
```

```
1   `timescale 1ns/1ns
2   module DivTB();
3   logic clk = 1'b0;
4   logic rst=1'b0;
5   logic start =1'b0;
6   logic [7:0] Q, M, Quotient, Remainder;
7   logic ready;
8   top uut (clk, rst, start, Q, M, Quotient, Remainder, ready);
9   always #5 clk <= ~clk;
10  initial begin
11      #3 rst =1;
12      #3 rst=0;
13      #13 Q = 8'd101;
14      #13 M = 8'd6;
15      #3 start =1;
16      #13 start =0;
17      #300 $stop;
18
19  end
20  endmodule
```



Test-Both:

```
1   `timescale 1ns/1ns
2   module DivTestBoth();
3   logic clk = 1'b0;
4   logic rst=1'b0;
5   logic start =1'b0;
6   logic [7:0] Q, M, Quotient, Remainder;
7   logic ready;
8   DivTop cut (clk, rst, start, Q, M, Quotient, Remainder, ready);
9   top cut2 (clk, rst, start, Q, M, QuotientQ, RemainderQ, readyQ);
10  always #5 clk <= ~clk;
11  initial begin
12      #3 rst = 1;
13      #10 rst = 0;
14      #15 Q = 8'd183;
15      #15 M = 8'd11;
16      #3 start =1;
17      #13 start =0;
18      #300 $stop;
19  end
20  endmodule
```



This assignment involves designing a divider circuit using bit shifting.

Datapath:

- The datapath consists of three registers: `A`, `Q`, and `M`.
- The algorithm utilizes these registers to calculate the quotient and remainder of a division operation over a period of 8 clock cycles. This is achieved by shifting registers `A` and `Q` in a specific sequence.

Controller:

- The controller employs a 2-bit state variable that defines four distinct states for the divider.
- The controller asserts a "ready" signal upon completing the division operation.
- (**Optional:** You can mention additional details about the state transitions at first.)

Control Counter:

- The control counter has two inputs and two outputs:
  - Inputs: These might be clock and reset signals.
  - Outputs:
    - `co`: This is likely the "carry out" signal, which represents the combined overflow from all counter bits.
    - `count`: This represents the current count value of the counter.

Asynchronous Reset:

- The design incorporates an asynchronous reset mechanism to initialize the circuit.

Overall Design:

- These components, the datapath, controller, control counter, and reset, work together to form the complete divider circuit!