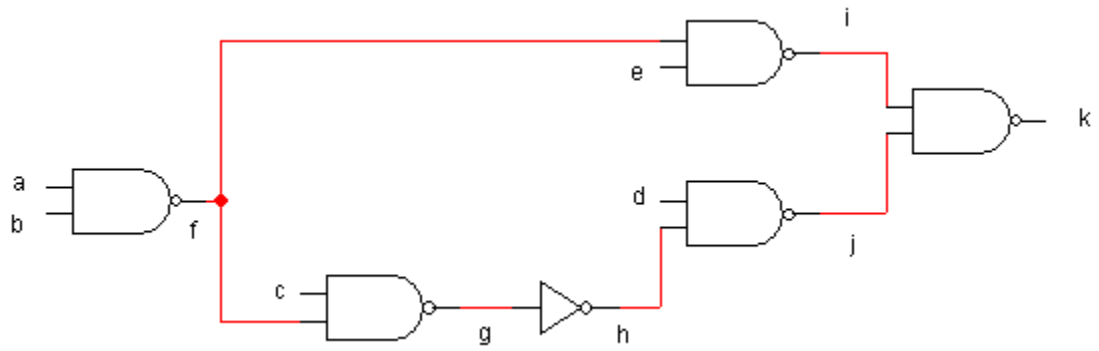
COE 405, Term 152

Design & Modeling of Digital Systems

HW# 7 Solution

Due date: Sunday, May 1

Q.1. Consider the logic network below with inputs *{a, b, c, d, e}* and output *{k}*:



Assume that the delay of the inverter gate is 1 and that the delay of the 2-input NAND gate is 2. Also, assume that the input data-ready times are zero except for input *a*, which is equal to 2.

(i) Compute the data ready times and slacks for all vertices in the network.

(ii) Determine the topological critical path.

(iii) Suggest an implementation of the function *k* using only inverters and 2-input NAND gates to reduce the delay of the circuit to the minimum possible and determine the maximum propagation delay in the optimized circuit. Has the area been affected?

(i)

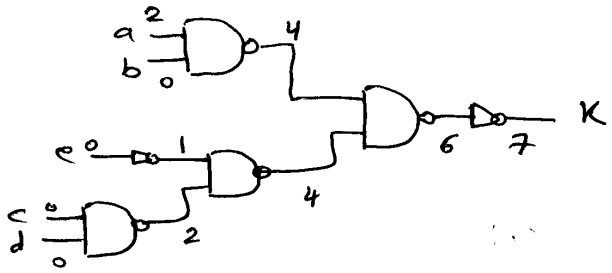
Data ready time	Data required time	Slack
$t_a = 2$	$\bar{t}_a = 4-2 = 2$	$S_a = 2-2 = 0$
$t_b = 0$	$\bar{t}_b = 4-2 = 2$	$S_b = 2-0 = 2$
$t_c = 0$	$\bar{t}_c = 6-2 = 4$	$S_c = 4-0 = 4$
$t_d = 0$	$\bar{t}_d = 9-2 = 7$	$S_d = 7-0 = 7$
$t_e = 0$	$\bar{t}_e = 9-2 = 7$	$S_e = 7-0 = 7$
$t_f = 2+2 = 4$	$\bar{t}_f = 6-2 = 4$	$S_f = 4-4 = 0$
$t_g = 4+2 = 6$	$\bar{t}_g = 7-1 = 6$	$S_g = 6-6 = 0$
$t_h = 6+1 = 7$	$\bar{t}_h = 9-2 = 7$	$S_h = 7-7 = 0$
$t_i = 4+2 = 6$	$\bar{t}_i = 11-2 = 9$	$S_i = 9-6 = 3$
$t_j = 7+2 = 9$	$\bar{t}_j = 11-2 = 9$	$S_j = 9-9 = 0$
$t_k = 9+2 = 11$	$\bar{t}_k = 11$	$S_k = 11-11 = 0$

(ii) The topological critical path in the input/output path with zero slack which is $\{a, f, g, h, j, K\}$.

(iii) $K = fe + dh = fe + fcd = f(e + cd)$

$= \overline{(ab)}(e + cd)$

This leads to the following implementation:



The maximum propagation delay in the optimized circuit is 7.

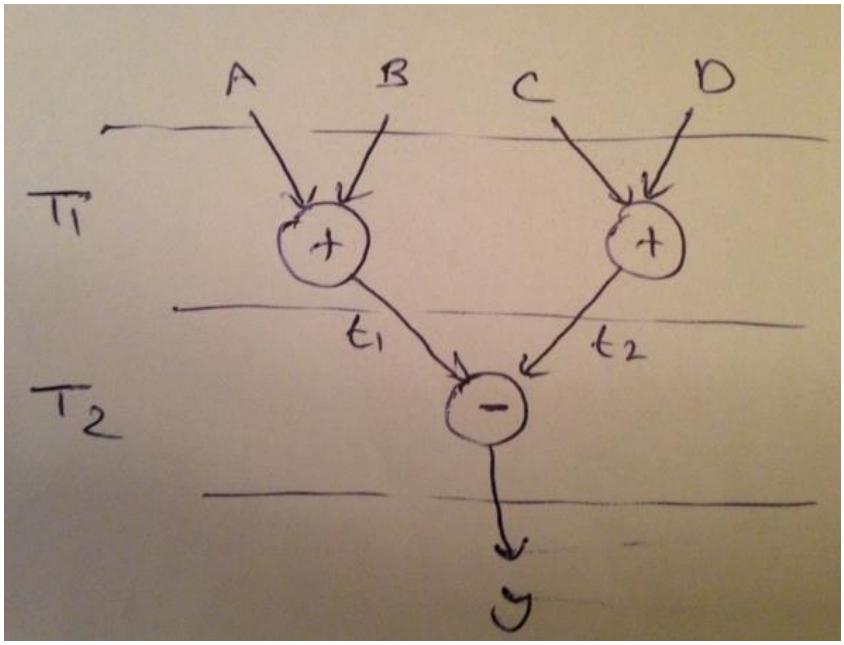
The number of literals in the original circuit is 11.

The number of literals in the optimized circuit is 6.

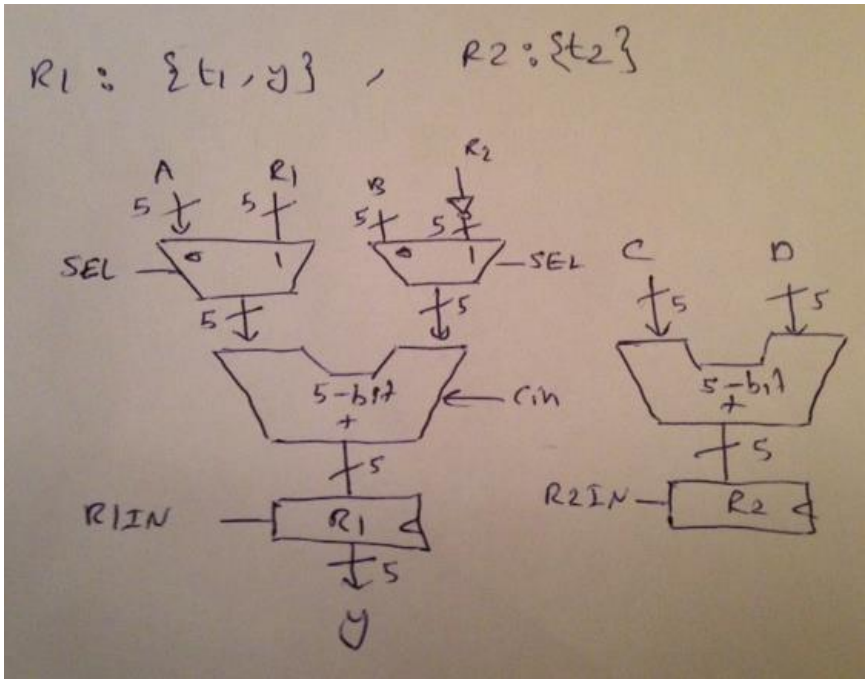
Thus, the area has also been reduced.

Q.2. It is required to design a circuit to compute the equation Y=A+B-C-D, where A, B, C and D are N-bit inputs. Assume that inputs are available only during the first cycle when a **START** input is asserted. Assume that a **DONE** signal will be set when the result is ready and the result will remain held until the next Start operation.

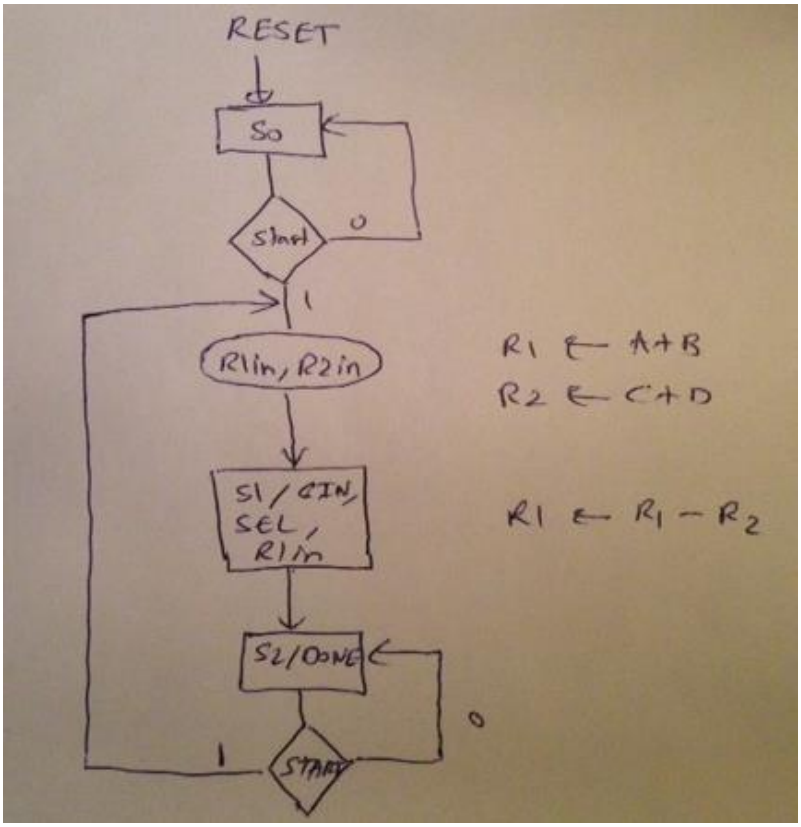
(i) Show a schedule of the operations with minimum latency (i.e., clock cycles) assuming that the clock cycle is limited by the time for performing one addition/subtraction operation. Store the output Y in a register.



(ii) Show the Data Path design of your circuit indicating all the control signals and the used adder/subtractor sizes.



(iii) Show the ASMD diagram of your control unit.



(iv) Write the necessary Verilog modules to module the data path unit, control unit and the overall circuit.

```
module HW7 #(parameter N=4) (output [N:0] Y, output DONE,
input [N-1:0] A, B, C, D, input START, RESET, CLK);


HW7_DPU #(N) M1 ( Y, A, B, C, D, SEL, R1IN, R2IN, CIN,  CLK);

HW7_CU M2 (SEL, R1IN, R2IN, CIN, DONE, START, RESET, CLK);


endmodule


module HW7_DPU #(parameter N=4) (output [N:0] Y, input [N-1:0]
A, B, C, D, input SEL, R1IN, R2IN, CIN,  CLK);


reg [N:0] R1, R2;
wire [N:0] MUX1, MUX2;

assign MUX1 = SEL? R1 : A;
assign MUX2 = SEL? ~{1'b0,R2} : B;
assign Y = R1;
```

```
always @(posedge CLK) begin
if (R1IN)
  R1 = MUX1 + MUX2 + CIN;

if (R2IN)
  R2 = C + D;

end


endmodule;



module HW7_CU (output reg SEL, R1IN, R2IN, CIN, DONE, input
START, RESET, CLK);

parameter S0 = 2'b00, S1=2'b01, S2=2'b10;

reg [2:0] state, next_state;

always @(posedge CLK, posedge RESET)
   if (RESET) state <= S0;
   else state <= next_state;

always @(state, START) begin
   SEL=0; R1IN=0; R2IN=0; CIN=0; DONE=0;
   case (state)
      S0:
        if (START)  begin
           next_state=S1; R1IN=1; R2IN=1; end
        else next_state=S0;
   S1: begin SEL=1; CIN=1; R1IN=1; next_state=S2; end
   S2: begin DONE=1;
        if (START)  begin
           next_state=S1; R1IN=1; R2IN=1; end
        else next_state=S2;
        end
       default: next_state='bx;
   endcase
end

endmodule
```

(v) Write a test bench to test the correct operation of your circuit. Include simulation snapshots.

```
module HW7_Test();



wire [4:0] Y;
wire DONE;
```

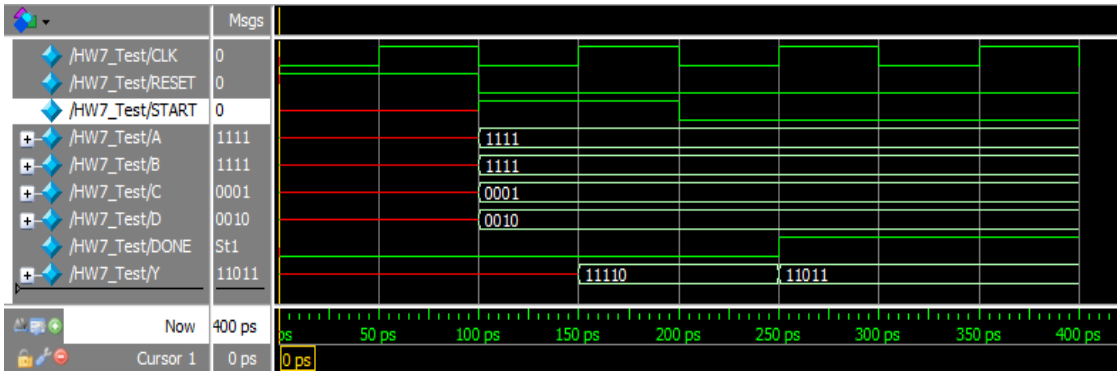
```
reg  [3:0] A, B, C, D;
reg  START, RESET, CLK;


HW7 #(4) M1 ( Y,  DONE, A, B, C, D, START, RESET, CLK);

initial begin
CLK = 0;
forever
#50 CLK = ~ CLK;
end

initial begin
RESET=1;
#100 RESET=0; START=1; A=15; B=15; C=1; D=2;
#100 START=0;
end


endmodule
```



(vi) Implement your circuit on FPGA assuming N=2 bits. Include a link for a video snapshot to demonstrate correct functionality of your circuit.