CS 4341
Fall 2019
Assigned: February 6th, 2019
Homework 2
Due: March 1st , 2019 11:59 pm

Objective:

Complete the following Exercises from the Mano book.

For the written exercises and diagrams:

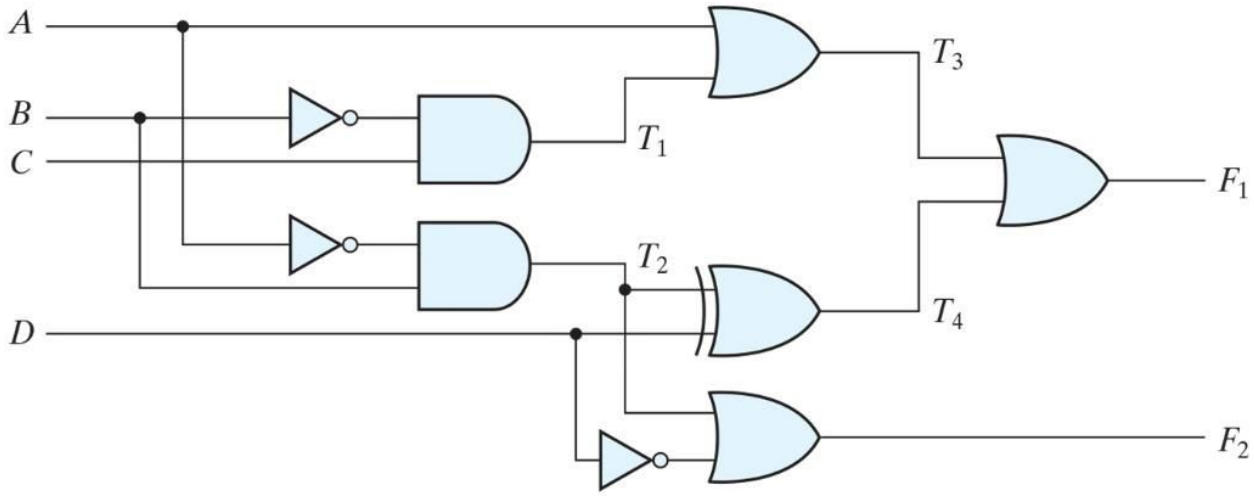
- Turn in a PDF file to the blackboard in the form of <COHORTNAME>.HW2.PDF.

For the Program and output, the following should be turned in:

- The program in the form of <COHORTNAME>.Program2.<EXTENSION>
- An output file as a text file, not a screen capture, in the form of <COHORTNAME>.Output2.txt.

Problem 1: Read a Circuit (10 points) Vishal

Given this circuit, find the formulas for T1, T2, T3, T4, F1, and F2.



- T1 = B' • C
- T2 = A' • B
- T3 = A + (B' • C)
- T4 = (A' • B) XOR D = (A' • B)'D + (A' • B)D'
- F1 = T3 + T4 = (A + (B' • C)) + ((A' • B) XOR D) = (A + (B' • C)) + ((A' • B)'D + (A' • B)D')
- F2 = T2 + D' =

Problem 2: Normalized Forms (5 points)

Simplify the equations for F1 and F2, resulting in a sum of products normalized form.

F_1

A	B	C	D	B'C	A+B'C	A'B	(A'B)'D+(A'B)D'	(A+B'C)+((A'B)'D+(A'B)D)'
0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1	1
0	0	1	0	1	1	0	0	1
0	0	1	1	1	1	0	1	1
0	1	0	0	0	0	1	1	1
0	1	0	1	0	0	1	0	0
0	1	1	0	0	0	1	1	1
0	1	1	1	0	0	1	0	0
1	0	0	0	0	1	0	0	1
1	0	0	1	0	1	0	1	1
1	0	1	0	1	1	0	0	1
1	0	1	1	1	1	0	1	1
1	1	0	0	0	1	0	0	1
1	1	0	1	0	1	0	1	1
1	1	1	0	0	1	0	0	1
1	1	1	1	0	1	0	1	1

F_1

AB \ CD	00	01	11	10
00	0	1	1	1
01	1	0	0	1
11	1	1	1	1
10	1	1	1	1

$$F_1 = A + B'D + B'C + BD'$$

F_2

A	B	C	D	(A'B)	D'	(A'B)+D'
0	0	0	0	0	1	1
0	0	0	1	0	0	0
0	0	1	0	0	1	1
0	0	1	1	0	0	0
0	1	0	0	1	1	1
0	1	0	1	1	0	1
0	1	1	0	1	1	1
0	1	1	1	1	0	1
1	0	0	0	0	1	1
1	0	0	1	0	0	0
1	0	1	0	0	1	1
1	0	1	1	0	0	0
1	1	0	0	0	1	1
1	1	0	1	0	0	0
1	1	1	0	0	1	1
1	1	1	1	0	0	0

F_2

AB \ CD	00	01	11	10
00	1	0	0	1
01	1	1	1	1
11	1	0	0	1
10	1	0	0	1

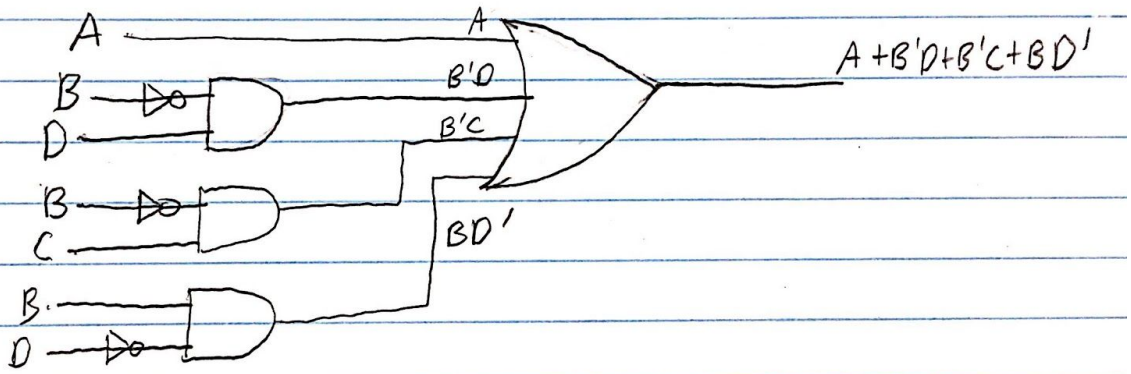
$F_2 = D' + A'B$

Problem 3: Draw Circuits (5 points)

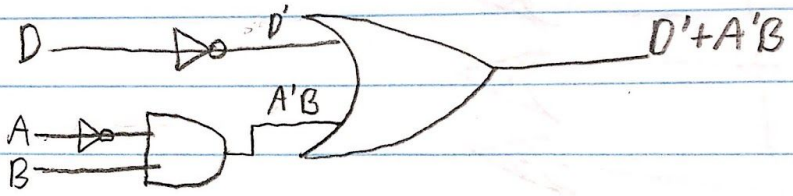
Draw a new circuit for F1 and F2

Circuits

$F_1 = A + B'D + B'C + BD'$



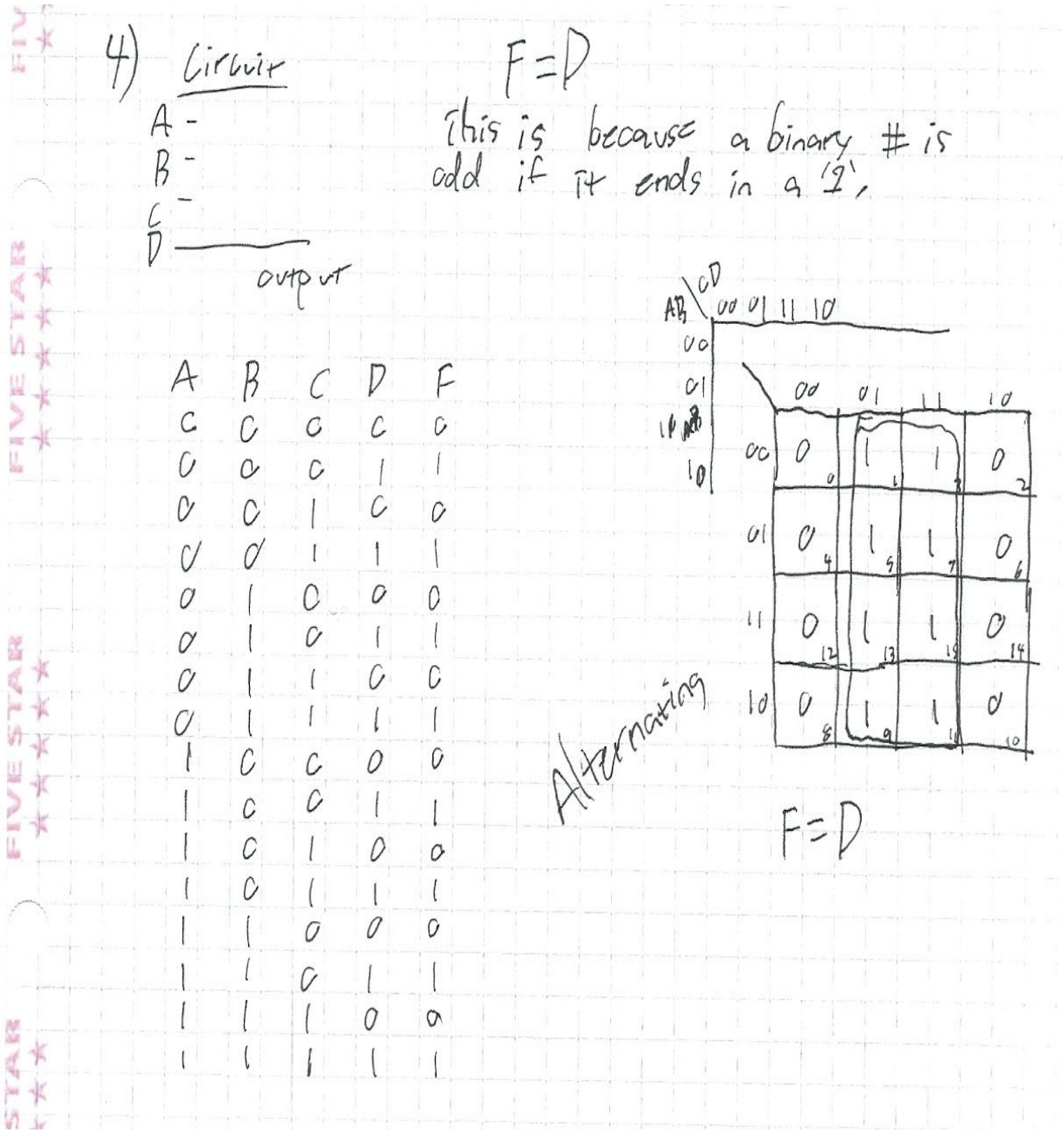
$F_2 = D' + A'B$



Problem 4: Circuit Design (5 points)

Design a combinational circuit with four input bits and one output bit. Show the truth table, the normalized equation, and the circuit. This includes gates, encoders, decoders, multiplexers, arbiters, adders, and half-adders.

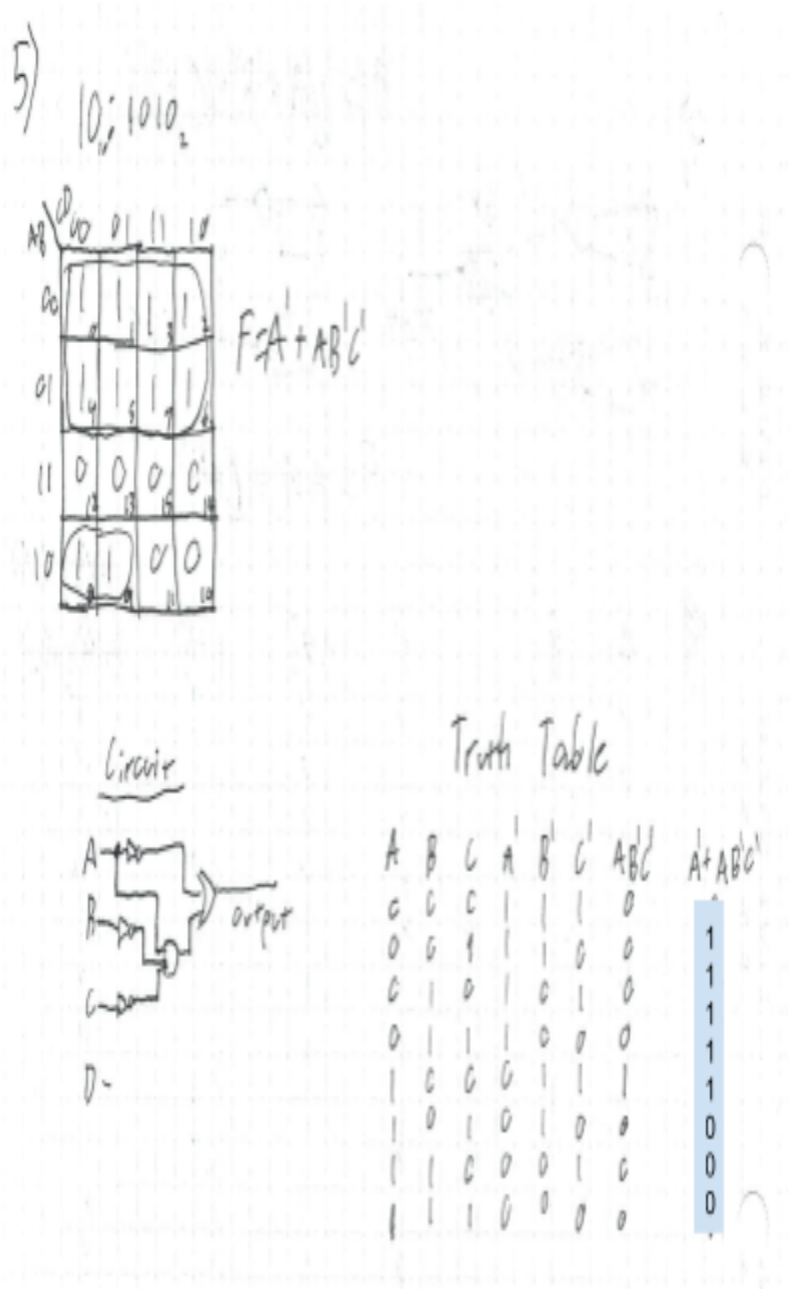
Solve: The output is 1 when the binary value of the inputs is an odd number.



Problem 5: Circuit Design (5 points)

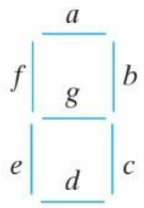
Design a combinational circuit with four input bits and one output bit. Show the truth table, the normalized equation, and the circuit. This includes gates, encoders, decoders, multiplexers, arbiters, adders, and half-adders.

Solve: The output is 1 when the binary value of the inputs is less than 10. The output is 0 otherwise.



Problem 6: The Seven Segment Decoder (25 points)

A Binary-Number to a Seven Segment decoder is a combinational circuit that converts a decimal value to an appropriate code for the selection of segments in the indicator used to display the decimal digit in a familiar form. The seven outputs of the decoder (a, b, c, d, e, f, g) select the corresponding segments in the display. Using a ***truth table of 4 input bits***, design ***a circuit*** that will run the seven segment encoder. ***Values 10 through 15 should not be included.***



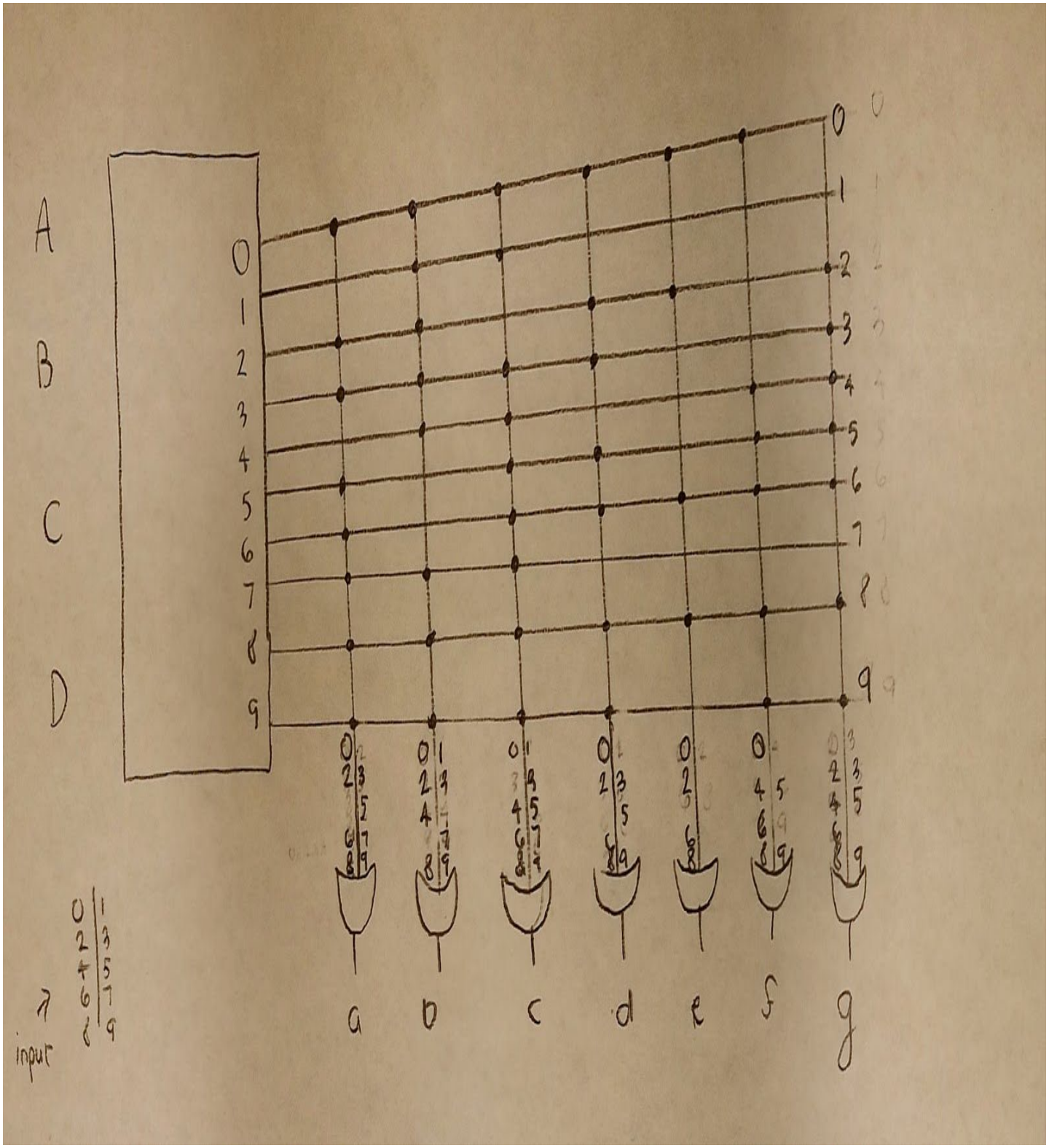
(a) Segment designation

(b) Numerical designation for display

Question 6: Seven Segment Encoder

Truth Table

Input					Output						
A	B	C	D		a	b	c	d	e	f	g
0	0	0	0	(0)	1	1	1	1	1	1	0
0	0	0	1	(1)	0	1	1	0	0	0	0
0	0	~~1~~	0	(~~2~~)	1	1	0	1	1	0	1
0	0	1	1	(3)	1	1	1	1	0	0	1
0	1	0	0	(4)	0	1	1	0	0	1	1
0	1	0	1	(5)	1	0	1	1	0	1	1
0	1	1	0	(6)	1	0	1	1	1	1	1
0	1	1	1	(7)	1	1	1	0	0	0	0
1	0	0	0	(8)	1	1	1	1	1	1	1
1	0	0	1	(9)	1	1	1	1	0	1	1
} Don't Cares 10-15											



NOTE: The box seen above is the <u>DECODER</u>

Problem 7: Delays (5 points)

Assume the XOR gates have a delay of 10 ns, AND gates have a delay of 5 ns, and OR Gates have a delay of 5 nanoseconds. What is the total delay in the four-bit adder with carry look-ahead? Hint: Look for the longest path through ***both*** circuits. This is a problem about being able to read a diagram and think through the problem. (Most students just freak out at the sight of a large circuit. Please breathe).

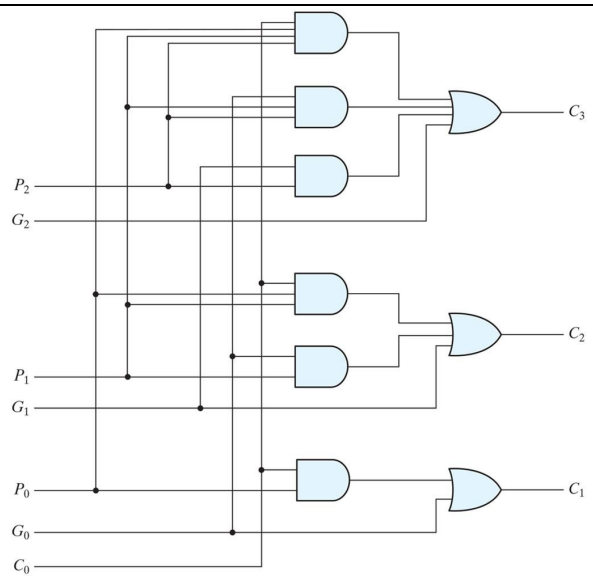


Figure 4.11
Logic diagram of carry look-ahead generator

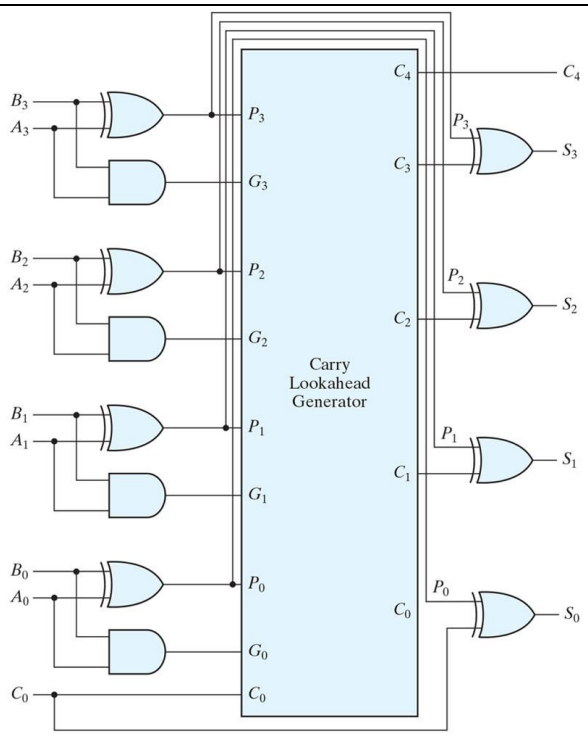


Figure 4.12
Four-bit adder with carry look-ahead

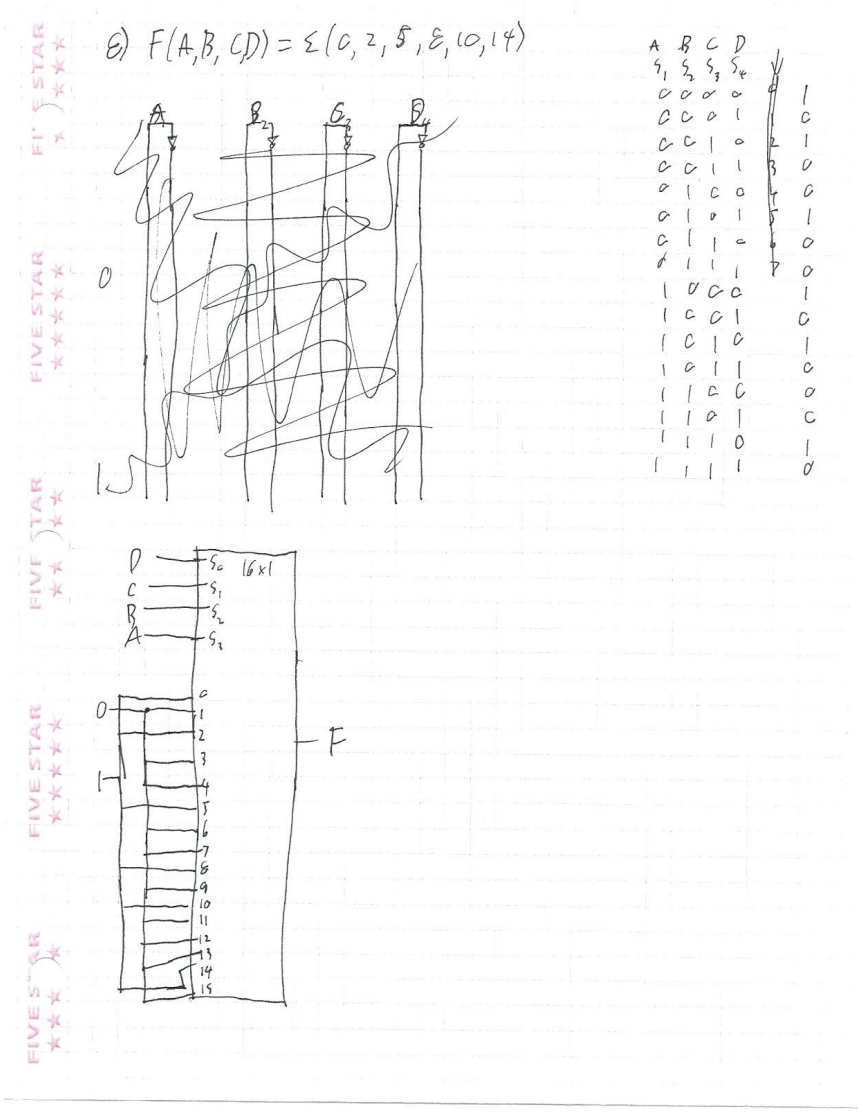
30 Nanosecond Delay

10 ns (XOR GATE Delay) + 5 ns (AND GATE Delay) + 5 ns (OR GATE Delay) +10 ns (XOR Gate Delay) = 30 ns

Problem 8: Multiplexer 1 (5 points)

Implement the following Boolean function with a **4-bit multiplexer.**

F(A,B,C,D)= Σ (0,2,5,8,10,14)

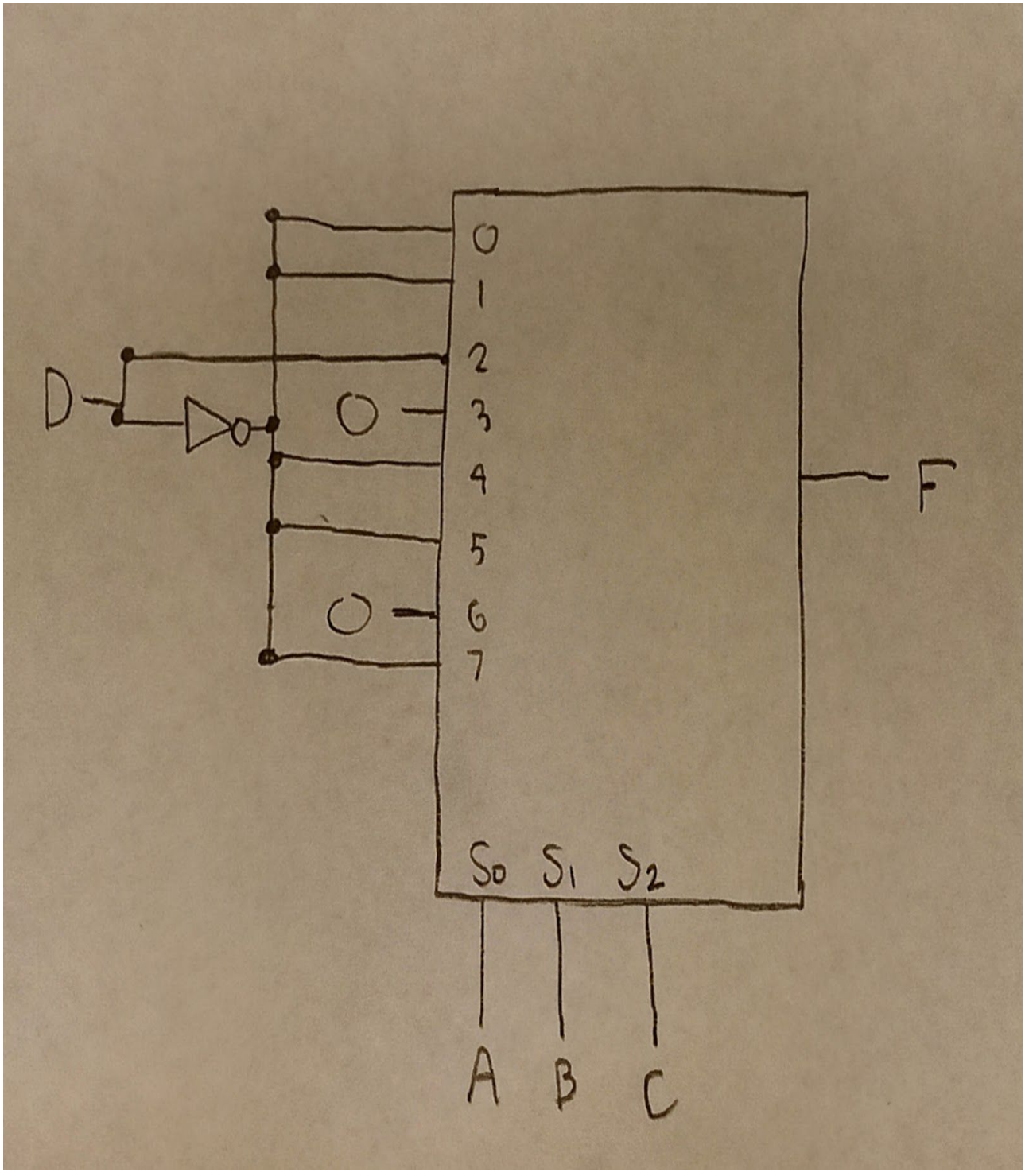


Problem 9: Multiplexer 2 (10 points)

Implement the following Boolean function with a **3-bit multiplexer** and appropriate gate logic.

F(A,B,C,D)= $\prod$ (1,3,4,6,7,9,11,12,13,15)

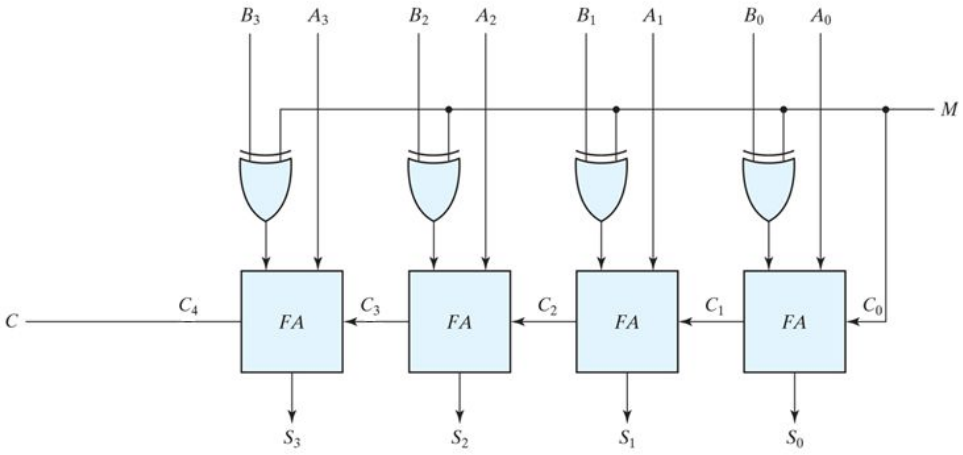
A	B	C	D	F	
0	0	0	0	1	F = D’
0	0	0	1	0	
0	0	1	0	1	F = D’
0	0	1	1	0	
0	1	0	0	0	F = D
0	1	0	1	1	
0	1	1	0	0	F = 0
0	1	1	1	0	
1	0	0	0	1	F = D’
1	0	0	1	0	
1	0	1	0	1	F = D’
1	0	1	1	0	
1	1	0	0	0	F = 0
1	1	0	1	0	
1	1	1	0	1	F = D’
1	1	1	1	0	


0
1
2
3
4
5
6
7
D
F
S0 S1 S2
A B C

Hardware Programming Language (Verilog, etc.) (25 points)

Write the HDL gate-level hierarchical description of a four-bit adder-subtractor for unsigned binary numbers similar to the following circuit. Be sure to examine the sample code on the next page.

Figure 4.13*a*, 4-Bit adder-subtractor without overflow.
Inputs: 4-Bit A, 4-Bit B, and Mode M (0=add/1=subtract)
Interfaces: Carry Bits C1, C2, C3
Outputs: Carry C (1 Bit, C4) , Sum S (4 bit)



Example Code:

What is special about this code is the system is now using *arrays* of components and bits in Verilog. This is a sample for you to study to use as a base to complete the program, not the program itself. Combine the sample code with the diagram to complete the problem

```
module Add_half (input a, b,  output c_out, sum);
  xor G1(sum, a, b);          // Gate instance names are optional
  and G2(c_out, a, b);
endmodule

module Add_full (input a, b, c_in, output c_out, sum);          // See Fig. 4.8
  wire w1, w2, w3;                                    // w1 is c_out; w2 is sum
  Add_half M1 (a, b, w1, w2);
  Add_half M0 (w2, c_in, w3, sum);
  or (c_out, w1, w3);
endmodule

module Add_rca_4 (input [3:0] a, b, input c_in output c_out, output [3:0] sum);
  wire c_in1, c_in3, c_in3, c_in4;                    // Intermediate carries
  Add_full M0 (a[0], b[0], c_in,   c_in1, sum[0]);
  Add_full M1 (a[1], b[1], c_in1, c_in2, sum[1]);
  Add_full M2 (a[2], b[2], c_in2, c_in3, sum[2]);
  Add_full M3 (a[3], b[3], c_in3, c_out, sum[3]);
endmodule

module Add_rca_8 (input [7:0] a, b, input c_in, output c_out, output [7:0] sum,)
  wire c_in4;
  Add_rca_4 M0 (a[3:0], b[3:0], c_in, c_in4, sum[3:0]);
  Add_rca_4 M1 (a[7:4], b[7:4], c_in4, c_out, sum[7:4]);
endmodule
```