

# ECE 6213 – Design of VLSI Circuits

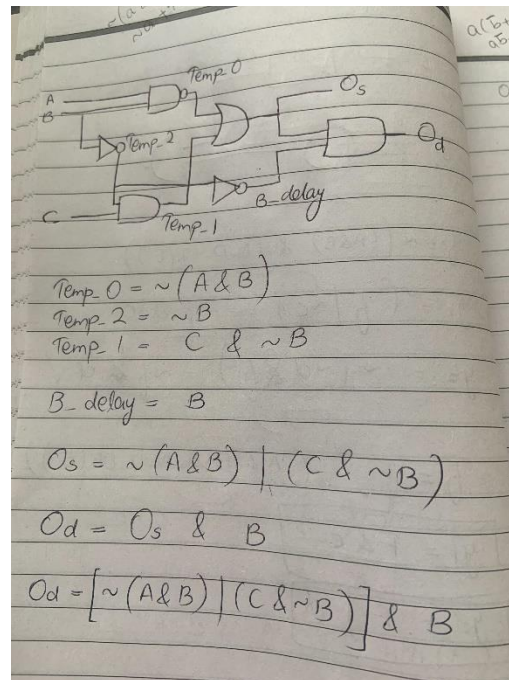
Fall 2022 – The George Washington University – Dr. Jerry Wu

## Osama Yousuf – HW 2

### Problem 1 – 35 points

- Obtain the equation and simplify it accordingly

The equation is:  $O_{dynamic} = [\sim(A \& B) | (C \& \sim B)] \& B$ . My work is attached below:



The simplified expressions are:  $O_{static} = \sim A | \sim B = \sim(AB)$ , and  $O_{dynamic} = \sim AB$ . My working is below:

Simplifying,

$$O_d = [\sim(A \& B) | (C \& \sim B)] \& B$$

Simplifying notations.

$$\begin{aligned} O_d &= (\overline{AB} + C\overline{B})B \\ O_d &= (\overline{A} + \overline{B} + C\overline{B})B \\ O_d &= \overline{A}B + \overline{B}B + C\overline{B}B \quad (\because B\overline{B} = 0) \\ O_d &= \overline{A}B \\ O_d &= \sim AB \end{aligned}$$

$$\begin{aligned} O_s &= \sim(A \& B) | (C \& \sim B) \\ O_s &= \overline{AB} + CB \\ O_s &= \overline{A} + \overline{B} + \overline{B}C \\ O_s &= \overline{A} + \overline{B}(C + 1) \\ O_s &= \overline{A} + \overline{B} \end{aligned}$$

- Discuss the possible issue with the original circuit

Let's calculate the timing delays for the two outputs:

$$Temp_0 = 2 \text{ nS}$$

$$Temp_2 = 0.8 \text{ nS}$$

$$Temp_1 = 0.8 + 2 = 2.8 \text{ nS}$$

$$O_{static} = 1.5 + 2.8 \text{ nS (longer delay out of its two inputs)} = 4.3 \text{ nS}$$

$$B_{delay} = 0.8 + 0.8 = 1.6 \text{ nS}$$

$$O_{dynamic} = 2 + 4.3 = 6.3 \text{ nS}$$

The issue thus in the circuit is that even though we wanted the  $B_{delay}$  to be the critical path for  $O_{dynamic}$ , the gate delay for  $O_{static}$  is greater and thus instead the total delay is longer than intended.

## Problem 2 – 55 points

- Create a truth table for “excess-5” code to BCD

Excess-5 input				BCD output			
w	x	y	z	a	b	c	d
0	1	0	1	0	0	0	0
0	1	1	0	0	0	0	1
0	1	1	1	0	0	1	0
1	0	0	0	0	0	1	1
1	0	0	1	0	1	0	0
1	0	1	0	0	1	0	1
1	0	1	1	0	1	1	0
1	1	0	0	0	1	1	1
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	1

Rest of the cases are don't care X.

- Derive and optimize each output (a, b, c, d) as a function of the inputs (by K-maps based on your truth table)

**K-Maps:**

Output a		y, z			
		00	01	11	10
w, x	00	X	X	X	X
	01	X			
	11		1	X	1
	10				

$$a(w, x, y, z) = wxz + wxy$$

**Verilog:** assign a = (w & x & z) | (w & x & y);

Output b		y, z			
		00	01	11	10
w, x	00	X	X	X	X
	01	X			
	11	1		X	
	10		1	1	1

$$b(w, x, y, z) = x'y + x'z + xy'z'$$

**Verilog:** assign b = (~x & z) | (~x & y) | (x & ~y & ~z);

Output c		y, z			
		00	01	11	10
w, x	00	X	X	X	X
	01	X		1	
	11	1		X	
	10	1		1	

$$c(w, x, y, z) = y'z' + yz$$

**Verilog:** assign c = (y & z) | (~y & ~z);

Output d		y, z			
		00	01	11	10
w, x	00	X	X	X	X
	01	X			1
	11	1		X	1
	10	1			1

$$d(w, x, y, z) = z'$$

**Verilog:** assign d = ~z;

- Model a converter circuit at gate level in structural style

```

8  module p2_gates(w, x, y, z, a, b, c, d);
9
10  input w, x, y, z;
11  output a, b, c, d;
12
13  wire t1, t2, t3
14
15      // model output a
16
17      and g1(t1, w, x);    // t1 = wx
18      and g2(t2, t1, y);   // t2 = wxy
19      and g3(t3, t1, z);   // t3 = wxz
20      or g4(a, t2, t3);    // a = wxy + wxz
21
22      // model output b
23      not g5(t4, x);        // t4 = x'
24      and g6(t5, t4, y);    // t5 = x'y
25      and g7(t6, t4, z);    // t6 = x'z
26
27      not g8(t7, y);        // t7 = y'
28      not g9(t8, z);        // t8 = z'
29
30      and g10(t9, x, t7);    // t9 = xy'
31      and g11(t10, t9, t8); // t10 = xy'z'
32
33      or g12(t11, t5, t6);   // t11 = x'y + x'z
34      or g13(b, t11, t10);  // b = x'y + x'z + xy'z'
35
36      // model output c
37      and g14(t12, t7, t8);  // t12 = y'z'
38      and g15(t13, y, z);    // t13 = yz
39      or g16(c, t12, t13);  // c = y'z' + yz
40
41      // model output d
42      not g17(d, z);         // d = z'
43
44  endmodule

```

### Problem 3 – 10 points

- Following is incomplete Verilog code, please finish based on the synthesis result on the attached schematic.

Problem 3 (10 pts): finish the coding.

The following is a incomplete verilog code.  
Please finish based on the synthesis result on the attached schematic.

```
module IDENTIFIERS (A, B, C, D, E, Y1, Y2);  
  Input A, B, C, D;  
  Input (7:0) E;  
  output Y1, Y2;  
  reg F, Y1, Y2;  
  
  function AND_OR_Bits;  
    Input (7:0) A;  
  begin  
    AND_OR_Bits = (A(7) & A(6) & A(5) & A(4)) &  
      (A(3) | A(2) | A(1) | A(0));  
  end  
endfunction  
  
always @ (A or B or C or D or E)  
begin  
  F =  & AND_OR_Bits(E);  
  Y1 =   
  Y2 =   
end  
endmodule
```

Jerry Wu 55

$F = A \& B \& \text{AND\_OR\_Bits}(E)$

$Y1 = F \& C$

$Y2 = F | D$