CSE 2301 Review

1. $V=10V$ $R_1=50\Omega$ $R_2=50\Omega$ $R_3=100\Omega$ $R_4=150\Omega$

a. $R_{1\|2}=\left(\frac{1}{50}+\frac{1}{50}\right)^{-1}=25\Omega$, $R_{eq}=25\Omega+100\Omega+150\Omega=\boxed{275\Omega}$

b. $V=IR_{eq}$

$10=275I$

$\boxed{.036364A}=I=\boxed{36.364mA}$

c. $P=I^2R$

$P=\boxed{.198W}=\boxed{198mW}$

2. $R_{eq}=75\Omega$ $R_3=25\Omega$ $R_1=60\Omega$ $R_2=?$

$R_{eq}=\left(\frac{1}{R_1}+\frac{1}{R_2}\right)^{-1}+R_3$

$50=\left(\frac{1}{60}+\frac{1}{R_2}\right)^{-1}$

$\frac{1}{50}=\frac{1}{60}+\frac{1}{R_2}$

$\frac{1}{50}-\frac{1}{60}=\frac{1}{R_2}$

$\boxed{300\Omega=R_2}$

3. AND

A	B	AB
0	0	0
0	1	0
1	0	0
1	1	1

OR

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

NOT

A	A'
0	1
1	0

4. AND

A	B	AB
1	1	1
1	0	1
0	1	1
0	0	0

OR

A	B	A+B
1	1	1
1	0	0
0	1	0
0	0	0

NOT

A	A'
1	0
0	1

5. SSI = Small Scale Integration
MSI = Medium Scale Integration
LSI = Large Scale Integration
VLSI = Very Large Scale Integration

These refer to chip density, or the number of gates/transistors on a chip.

6. low = 0V-.8V
undefined = .8V-2V
high = 2V-Vcc (Vcc ≈ 5V)

A voltage in the undefined range cannot be classified as either low or high; it's ambiguous.

7.

A	B	C	D	E	F	G	↓ H	↓ I	J	↓ K
0	0	0	0	0	0	0	0	1	1	0
0	0	0	1	0	0	1	0	0	0	1
0	0	1	0	0	1	0	0	1	1	0
0	0	1	1	0	1	1	0	0	0	1
0	1	0	0	0	1	1	0	0	0	1
0	1	0	1	0	1	1	0	0	0	1
0	1	1	0	0	1	1	0	0	0	1
0	1	1	1	0	1	1	0	0	0	1
1	0	0	0	0	0	0	0	1	1	0
1	0	0	1	0	0	1	0	0	0	1
1	0	1	0	1	1	0	1	1	1	0
1	0	1	1	1	1	1	1	0	1	0
1	1	0	0	0	1	1	0	0	0	1
1	1	0	1	0	1	1	0	0	0	1
1	1	1	0	1	1	1	1	0	1	0
1	1	1	1	1	1	1	1	0	1	0

8. a. 47_{10} | Binary: 101111_2 Ternary: 1202_3 Octal: 57_8 Hex: $2F_{16}$

b. 10110111_2 | Decimal: 183_{10} Ternary: 20210_3 Octal: 267_8 Hex: $B7_{16}$

c. 212_3 | Decimal: 23_{10} Binary: 10111_2 Octal: 27_8 Hex: 17_{16}

d. 67_8 | Decimal: 55_{10} Binary: 110111_2 Ternary: 2001_3 Hex: 37_{16}

e. $1F3_{16}$ | Decimal: 499_{10} Binary: 111110011_2 Ternary: 200111_3 Octal: 763_8

9. -13.375_{10} Sign bit = 1

$13.375 = 1101.011 = 1.101011 \cdot 2^3$

$\underline{E} = 127+3 = 130 = 10000010$

$M = 10101100\ldots$

Final representation: $1\ 10000010\ 101011\underbrace{00000\ldots}_{17\text{ zeros}}$

10. a.
```
  10110111  (183)
 +01000111  (71)
  11111110
```
b.
```
  10110111
 -01000111
  01110000
```

11. a. -43_{10} = 10101011_2 signed magnitude = 11010101 2's comp.

b. $+26_{10}$ = 00011010_2 signed magnitude = 00011010 2's comp.

```
  11010101
 +00011010
  11101111₂ = -17₁₀ ✓
```

12. If the signs (MSBs) of the addends are the same but the sign of the result is different, there has been an overflow. This is because the result is out of range for the number of bits you are working with. For instance:

(-3) + (-6) = (-9), but:

```
  1101   (-3)
+ 1010   (-6)
 10111 ——> we are limited to 4 bits, so this is +7.
           we have an overflow.
```

13. We use the gray code because each successive number differs by only 1 bit. If we used normal binary, you get 2 bits changing at once between certain values. While they switch at the same moment ideally, this is not the case in reality, which could cause confusion in the system. To avoid this, we have one bit change at once.

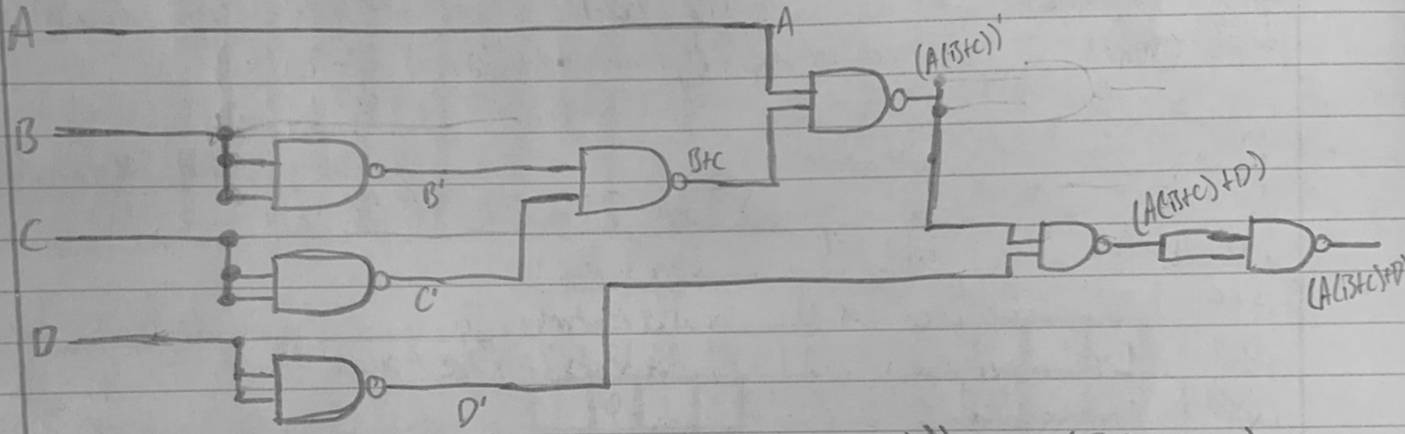
Bin	Gray	New
000	000	111
001	001	110
010	011	100
011	010	101
100	110	001
101	111	000
110	101	010
111	100	011

(essentially inverted)

14. Start with NOT: $A' = A \text{ NAND } A = (A \cdot A)' = A' + A' = \underline{A'}$

How can we do OR: $A+B = (A' \cdot B')' = A' \text{ NAND } B'$

Of course AND: $A \cdot B = ((A \cdot B)')'$



$(A(B+C)+D)' = (A \text{ NAND } ((B \text{ NAND } B) \text{ NAND } (C \text{ NAND } C))) \text{ NAND } (D \text{ NAND } D)$

NAND

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | (same)

15.

Bin	XS3
0000	0011
0001	0100
0010	0101
0011	0110
0100	0111
0101	1000
0110	1001
0111	1010
1000	1011
1001	1100

The range of XS3 is -3_{10} - 12_{10} because 0000 is the smallest 4-bit number in XS3, which would correspond to -3. Similarly, the largest 4 bits is 1111, which is 1100 in binary, or 12 in decimal.

16. We use BCD because it is limited to 0-9, which makes it a compromise between man and machine. It offers easy conversion between systems, and is easy to read for both parties.

17. A minterm is a product term with n literals which represents some binary input. For instance, $x'y'z$ would represent 001. Their relationship with truth tables is best described by an example.

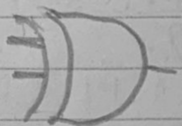
xyz	f
000	1
001	0
010	0
011	1
100	1

The minterms where f=1 are $x'y'z'$, $x'yz$, and $xy'z'$, so the logic equation for f can be written as:

$f = m_0 + m_3 + m_4$
$f = x'y'z' + x'yz + xy'z'$

18. XOR

A	B	A⊕B
0	0	0
0	1	1
1	0	1
1	1	0



XOR can be used to see if there's an odd number of 1's among a set of inputs.

19. OR = +
AND = ·
NOT = '
XOR = ⊕

$(A+B) \oplus (CD)'$

20.

DCBA	a	b	c	d	e	f	g
0000	1 ✓	1 ✓	1 ✓	1 ✓	1 ✓	1 ✓	0 ✓
0001	0 ✓	1 ✓	1 ✓	0 ✓	0 ✓	0 ✓	0 ✓
0010	1 ✓	1 ✓	0 ✓	1 ✓	1 ✓	0 ✓	1 ✓
0011	1 ✓	1 ✓	1 ✓	1 ✓	0 ✓	0 ✓	1 ✓
0100	0 ✓	1 ✓	1 ✓	0 ✓	0 ✓	1 ✓	1 ✓
0101	1 ✓	0 ✓	1 ✓	1 ✓	0 ✓	1 ✓	1 ✓
0110	1 ✓	0 ✓	1 ✓	1 ✓	1 ✓	1 ✓	1 ✓
0111	1 ✓	1 ✓	1 ✓	0 ✓	0 ✓	0 ✓	0 ✓
1000	1 ✓	1 ✓	1 ✓	1 ✓	1 ✓	1 ✓	1 ✓
1001	1 ✓	1 ✓	1 ✓	1 ✓	0 ✓	1 ✓	1 ✓

a Map:

DC \ BA	00	01	11	10
00	1	0	1	1
01	0	1	1	1
11	X	X	X	X
10	1	1	X	X

$\boxed{a = B + D + CA + C'A'}$

b Map:

DC \ BA	00	01	11	10
00	1	1	1	1
01	1	0	1	0
11	X	X	X	X
10	1	1	X	X

$\boxed{b = C' + B'A' + BA}$

f Map:

DC \ BA	00	01	11	10
00	1	0	0	0
01	1	1	0	1
11	X	X	X	X
10	1	1	X	X

$\boxed{f = D + B'A' + CA' + CB'}$

d Map:

DC \ BA	00	01	11	10
00	1	0	1	1
01	0	1	0	1
11	X	X	X	X
10	1	1	X	X

$\boxed{d = D + C'B + C'A' + BA' + CB'A}$

c Map:

DC \ BA	00	01	11	10
00	1	1	1	0
01	1	1	1	1
11	X	X	X	X
10	1	1	X	X

$\boxed{C = B' + A + C}$

g Map:

DC \ BA	00	01	11	10
00	0	0	1	1
01	1	1	0	1
11	X	X	X	X
10	1	1	X	X

$\boxed{g = D + CB' + C'B + BA'}$

e Map:

DC \ BA	00	01	11	10
00	1	0	0	1
01	0	0	0	1
11	X	X	X	X
10	1	0	X	X

$\boxed{e = C'A' + BA'}$

21. $f = C'A' + CA + DCB$

DC \ BA	00	01	11	10
00	1	0	0	1
01	0	1	1	0
11	0	1	1	1
10	1	0	0	1

22. B=1 so that the bottom pin of the 3-OR gate is always 0.

A can be either 1 or 0 so that the top pin of the 3-OR gate is always 0. This is because B=1, so B'=0, meaning the first AND gate (B'A) always outputs 0. Also, the second AND gate represents B'AA', which will always output 0 no matter what. This type of configuration itself is often a bug which can result from a misplaced connection.

23. We are attempting to propagate a change in C through the circuit to be observed as a change in D. This is called path sensitization.

24. Bits B and A comprise our test vector. In our case, the test vector BA is either 10 or 11.