William Fenton
CSF - Digital Logic
HW2

=========
Chapter 1
=========

9. What is the decimal value of the 8-bit pattern in the previous problem assuming the number is a signed integer coded in 8-bit two's complement representation?

1110 1101
0001 0010 // Flip the bits
0001 0011 // Add 1
0001 0011 = 1 + 2 + 16 = 19

-19

11. What is the hexadecimal value of the number 101910 assuming two's complement representation in 12 bits?

1 * 16^4 + 8 * 16^3 + 14 * 16^2 + 1 * 16^1 + 6 * 16^0 = 0x18E16

14. What is the decimal value of 0xFAB assuming two's complement representation in 12 bits?

1111 1010 1011 // Leftmost bit is a 1 so it's negative
15 * 16^2 + 10 * 16^1 + 11 * 16^0 = 4011 // Convert to decimal
4011 - 2^12 = -85 // Subtract 2^n where n=number of bits

-85

16. What is the hex value of the following binary number assuming 16 bit two's complement representation for signed integers?

1011 1010 1011 1110

0100 0101 0100 0001 // Flip the bits
0100 0101 0100 0010 // Add 1
---- ---- ---- ---- // Convert to hex
4 5 4 2
0x4542

17. Convert 0x871 to signed decimal. Assume two's complement representation in 12 bits.

1000 0111 0001 // Leftmost bit is a 1 so it's negative
8 * 16^2 + 7 * 16^1 + 1 * 16^0 = 2161 // Convert to decimal
2161 - 2^12 = -1935 // Subtract 2^n

-1935

19. What is the range of signed integers that can be represented in 12 bits?

2^12 = 4096

20. Convert the following binary pattern to decimal. Assume the pattern is coded in 12 bits using two's complement representation.

1111 0100 1110

0000 1011 0001 // Flip the bits
0000 1011 0010 // Add one
0x0B2 = 11 * 16^1 + 2 * 16^ = 178 // To hex then decimal

-178

21. How many bits are needed to represent the number −2271base10 in two's complement representation?

-2^(k-1) to 2^(k-1) -1 // Formula for 2's complement range
-2^(12-1) to 2^(12-1) - 1 = -2048 to 2047 // Trying 12bit
-2^(13-1) to 2^(13-1) -1 = -4096 to 4095 // -2271 is within the range where k=13

13bits

22. Convert the number −2271base10 to binary in two's complement representation. Fill out the number to the nearest nibble boundary.

-2271 + 2^13 = 5921 // Since the number was negative, add 2^n to it
0001 0111 0010 0001 // Convert to binary (13bit)

=========
Chapter 2
=========

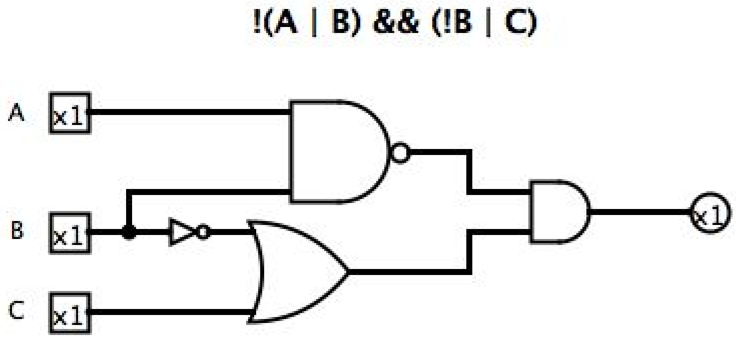
1. Evaluate the following logic expression for the variable assignment: A = 0, B = 1, C = 1, D = 0.

¬(A ∨ B) ∧ (¬B ∨ C) ∨ D

!(0 | 1) && (0 | 1) | 0 // Insert truth values
(1 && 0) && (0 | 1) | 0 // DeMorgan's law
(0) && (1) | 0 // Domination law
(0 && 1) | 0 // Rewriting to make clear precedence
0 | 0 // Domination law
------
0 | 0 == 0

3. Give a logic circuit diagram for the following logic expression.

¬(A ∨ B) ∧ (¬B ∨ C)



4. Build the truth table for the logic expression in the previous exercise.

A	B	C	!A	!B	!C	A NAND B	!B | C	P && Q
0	0	0	1	1	1	1	1	1
0	0	1	1	1	0	1	1	1
0	1	0	1	0	1	0	0	0
0	1	1	1	0	0	0	1	0
1	0	0	0	1	1	0	1	0
1	0	1	0	1	0	0	1	0
1	1	0	0	0	1	0	0	0
1	1	1	0	0	0	0	1	0

5. Give equivalent forms for the logic expression A∧((¬B)∨(¬C))∨(¬C ∧B)
using each of the three following conventions
(a) Use juxtaposition for And, ∨ for Or, and ¬ for Not.
(b) Use juxtaposition for And, + for Or, and ¬ for Not.
(c) Use juxtaposition for And, + for Or, and the overbar for Not.

A((¬B) ∨ (¬C)) ∨ (¬CB)
A((¬B) + (¬C)) + (¬CB)
A(($\overline{B}$) + ($\overline{C}$)) + ($\overline{C}$B)

6. Give a logic table for the following logic expression (from the previous

exercise).

A ∧ ((¬B) ∨ (¬C)) ∨ (¬C ∧ B)

					P	Q	R	
A	**B**	**C**	**!B**	**!C**	**!B | !C**	**!C && B**	**A && P**	**R | Q**
0	0	0	1	1	1	0	0	1
0	0	1	1	0	1	0	0	1
0	1	0	0	1	1	1	0	1
0	1	1	0	0	0	0	0	0
1	0	0	1	1	1	0	1	1
1	0	1	1	0	1	0	1	1
1	1	0	0	1	1	1	1	1
1	1	1	0	0	0	0	0	0

8. Convert the expression in the previous exercise to disjunctive normal form.

(A ∧ ((¬B) ∨ (¬C))) ∨ (¬C ∧ B)
(A && !B) | (A && !C) | (!C && B) // Distributive law

9. Give a logic expression in full disjunctive normal form for the constant logic function 1. Assume two input variables.

(A && B) | (!A && !B) // From (A->B) && (B->A)

10. Show how a Nand gate can be used to implement a Not gate.

A	B	A && B	! (A && B)	A && A	!(A && A)	!A
0	0	0	1	0	1	1
0	1	0	1	0	1	1
1	0	0	1	1	0	0
1	1	1	0	1	0	0

11. Show how an And gate can be implemented with only Nand gates.

A	B	A && B	! (A && B)	!(!(A && B)
0	0	0	1	0
0	1	0	1	0
1	0	0	1	0
1	1	1	0	1

12. Show how an Or gate can be implemented with only Nand gates.

A	B	!(A && A)	!(B && B)	(A Nand A) && (B Nand B)	!(A Nand A) && (B Nand B)	A | B
0	0	1	1	1	0	0
0	1	1	0	0	1	1
1	0	0	1	0	1	1
1	1	0	0	0	1	1