Prof. Xiang 'Anthony' Chen

Midterm Exam

Name (Last, First): Zhao, Liqi

Student Id #: 905154257

Student to Left: Wang, Michael

Student to Right: Wu, Yangchao,

Do not start working until instructed to do so.

1. *You must answer in the* ***space provided*** *for answers after every question. We will ignore answers written anywhere else in the booklet.* ***All pages in this booklet must be accounted*** *for otherwise it will not be graded.*
2. *You are permitted 1 page of notes 8.5x11 (front and back).*
3. *You may not use any electronic device.*

Following table to be filled by course staff only

	Maximum Score	Your Score
Question 1	15	
Question 2	25	
Question 3	25	
Question 4	35	
Question 5 (EC)	+5	
TOTAL	100	

Question #1 (15 pts)

Consider the following Karnaugh Map for the Boolean function, Y. A blank truth table is provided for your convenience.

B A

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

D C

A	B	C	D	Y
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

(a) Circle the prime implicants on the map. (5 pts)
How many prime implicants are there? 5

(b) Write the Boolean (sum-of-product) expression of the *essential* prime implicants of (b) (if any). (5 pts)

EssentialPrimeImplicants = $(\neg A \wedge B \wedge D) \vee (C \wedge \neg D) \vee (A \wedge \neg B \wedge \neg C) \vee (A \wedge \neg D)$

(c) Express as a minimal $\boxed{\text{sum of product}}$, $\neg Y$. (5 pts)
The K-map is provided for your convenience.

B
A
AB

CD \ **AB**	"00"	"01"	"11"	"10"
"00"	0	0	1	1
"01"	0	1	0	1
"11"	0	1	0	0
"10"	1	1	X	X

D
C

$$\neg Y = (\neg A \land \neg C \land \neg D) \lor (\neg A \land \neg B \land D)$$
$$\lor (A \land B \land D) \lor (A \land C)$$

Question #2 (25 pts)

(a) Is DeMorgan's theorem still true with more than two variables? If so, prove it in the case of three variables x, y and z. (5 pts)

Yes consider $(x \wedge y \wedge z) = \neg(\neg(x\wedge y) \vee \neg z) = \neg((\neg x \vee \neg y) \vee \neg z)$
$= \neg(\neg x \vee \neg y \vee \neg z)$

$(x \vee y \vee z) = \neg(\neg(x \vee y) \wedge \neg z) = \neg((\neg x \wedge \neg y) \wedge \neg z)$
$= \neg(\neg x \wedge \neg y \wedge \neg z)$

So the DeMorgan's theorem still works for 3 variables

(b) Rewrite the following Boolean equation in (Disjunctive) Normal form. (6 pts)

$$f = \overline{A \oplus B} + \overline{B \oplus C}$$

where $\oplus$ means XOR operation, i.e., $A \oplus B = A\bar{B} + \bar{A}B$

Answer: $f = (\overline{A\bar B + \bar A B}) + (\overline{B\bar C + \bar B C})$

$= (\overline{A\bar B} \cdot \overline{\bar A B}) + (\overline{B\bar C} \cdot \overline{\bar B C})$

$= (\bar A + B)\cdot(A + \bar B) + (\bar B + C)\cdot(B + \bar C)$

$= \cancel{\bar A A} + BA + \bar A \bar B + \cancel{B \bar B} + \cancel{\bar B B} + CB + \bar B \bar C + \cancel{C \bar C}$

$= AB + \bar A \bar B + \bar B \bar C + BC$

$= \cancel{ABC} + AB\bar C + \cancel{\bar A \bar B \bar C} + \bar A \bar B C + A\bar B \bar C + \bar A \bar B \bar C + ABC + \bar A BC$

$f =$ $AB\bar C + \bar A \bar B C + A \bar B \bar C + \bar A \bar B \bar C + ABC + \bar A BC$

(c) Simplify f from (b) to a minimum sum-of-products. List which Boolean properties you use at each step of the simplification. Hint: you may use K-map to ***verify*** your answer. (6 pts)

Answer: $AB\bar C + \bar A \bar B C + A\bar B \bar C + \bar A \bar B \bar C$
$+ ABC + \bar A BC$

$= AB + \bar B \bar C + \bar A \bar B C + \bar A BC$

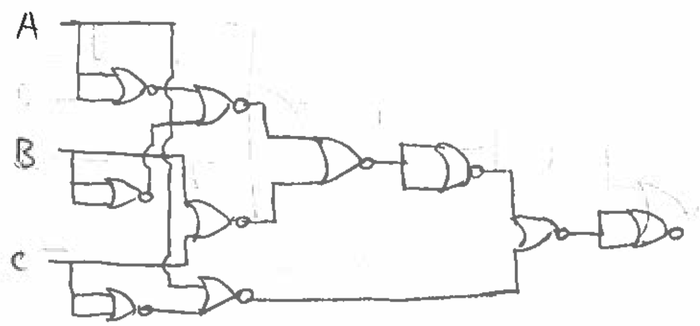
$= AB + \bar B \bar C + (\bar A \bar B + \bar A B)C$

$= AB + \bar B \bar C + \bar A C$

BC \ A	0	1
00	1	1
01	1	0
11	1	1
10	0	1

$f =$ $\bar B \bar C + AB + \bar A C$

(d) With only 2-input NOR gates, implement f with a minimal number of gates. Draw the gate diagram. (Note: no complemented inputs are given) (8 pts)



$$\bar{B}\bar{C}+AB+\bar{A}C$$
$$=\overline{(B+C)}+\overline{(\bar{A}+\bar{B})}+\overline{(A+\bar{C})}$$

10 NOR gates are needed

256
128 . . 16 . 2 1 19
0001100 1001 1
256
19 + 128 = 147

Question #3 (25 pts)

The following 12-bit word can be used to represent different numbers depending on the encoding 6 13

12b'1110_0110_1101 13 ⇒ D 1110 0110 1100

(a) If the word is 2's complement, what is the corresponding integer? (4 pts) - 403

(b) If we convert the word (treated as unsigned) into base-4, what is the represented number? (3 pts)

321231

(c) If we take answer in (b), extending how we define 1's complement for base-2, write the 3's complement of the base-4 number. (4 pts)

012103

(d) What is this word in Hexadecimal? (3 pts) E6D

(e) In base-20 system, assume each digit is now 00, 01, 02, … 09, 10, 11, …19 (each called a "vigit"). For example, 01,19 is 39 in decimal. Using 3 "vigits": $3 \times \underbrace{20^2}_{400} + 2 \times \underbrace{20^1}_{20} + 6$

How would one represent a base-10 integer 1246? (4 pts) 03, 02, 06

What's the 20's complement representation of -1246 (i.e. the 20's complement of the 1246)? (4 pts) 16, 18, 14

19, 19, 19 000001 1111111

Using the first vigit as the sign vigit, what is the most positive value in base-10 integer that can be represented? (3 pts) $20^{6}/2 - 1$

5 vigits

00, $\boxed{01}$

19 19 . 19 . 19 . 19 . 19

0 0 0 0 0 0

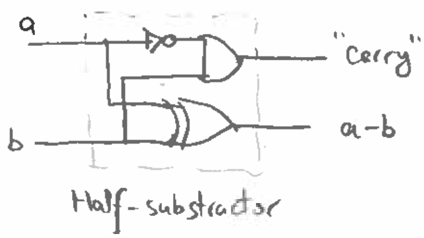
19×20^5 $\underbrace{19 \times 20^4}_{4} + \underbrace{19 \times 20^3}_{4} + \underbrace{19 \times 20^2}_{3} + \underbrace{19 \times 20}_{2} + \underbrace{19}_{1}$

Question #4 (35 pts)

(a) Implement a one-bit "half-subtractor" from gates. The carry-out of this subtractor is 1 when the result is -1. The truth table for this is shown below: (8 pts)

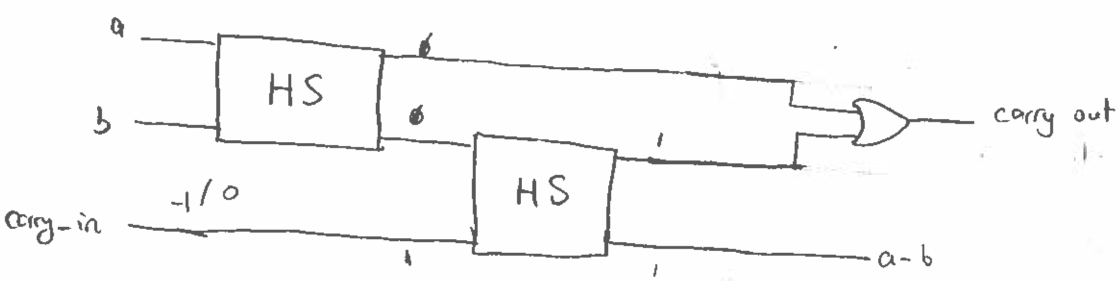
a	b	a - b	"carry"
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

a-b = a XOR b
carry = b AND $\bar{a}$



(b) Implement a "full-subtractor" from "half-subtractor" blocks. (6 pts)

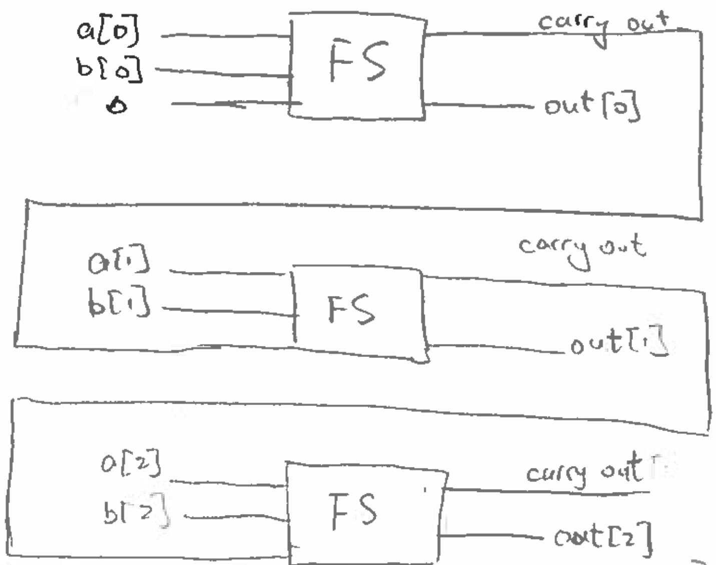
a-b < 0
b-c ≥ 0



0 - 1 - 1
= -1 0 1

1 - 1 - 1
01

(c) Implement a 3-bit "subtractor" from 1-bit "full-subtractor" blocks. (7 pts)



inputs are a[2:0], b[2:0]

The results are a 3-bit out[2:0]
and a 1-bit carry-out

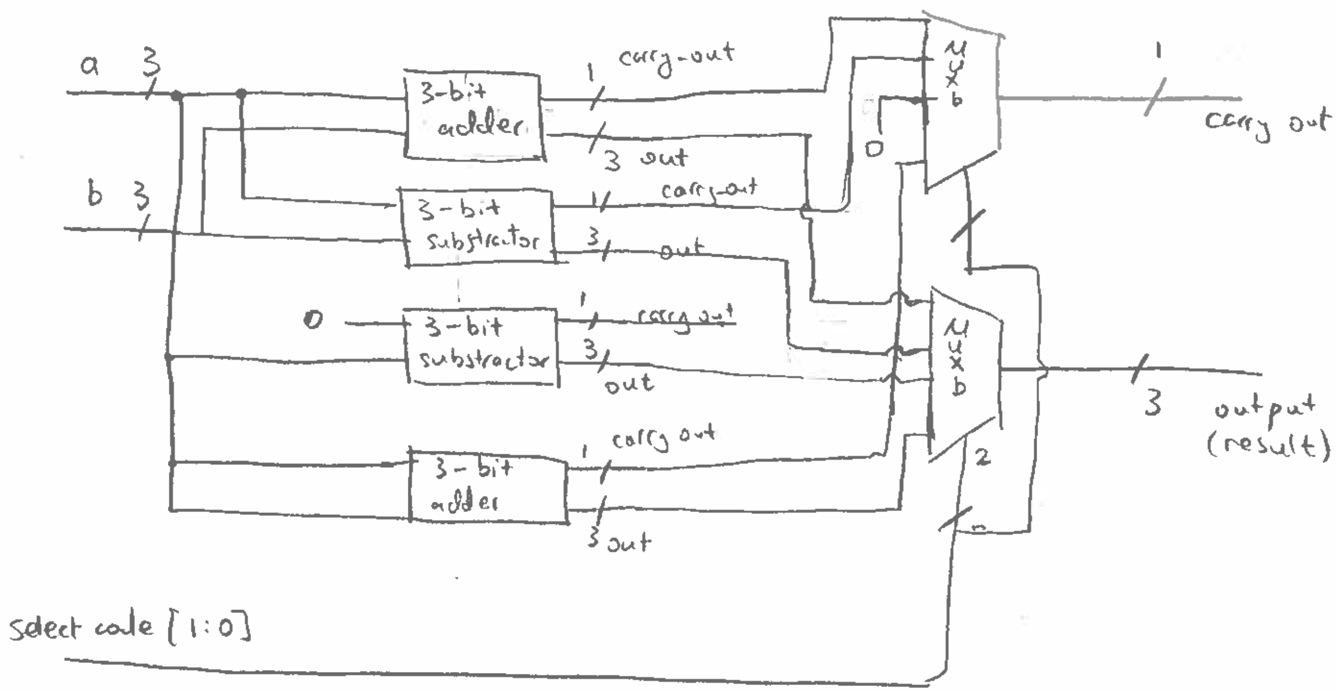
(d) Processors use a block called an ALU (Arithmetic Logic Unit) as part of their processing capability. Here we will implement a very basic ALU with a total of 4 functions, selected by a 2-bit code. Using the building blocks discussed in lecture and the 3-bit subtractor block, implement a 3-bit ALU that can add, subtract, negate one argument, and multiply by 2. The select codes are listed in the table below. Note that there are 3 inputs (3-bit a, 3-bit b, and the 2-bit select code) and 2 outputs (3-bit result and a 1-bit carry). (14 pts)

Hint: Multiplying a number is like shifting the bits to the left and using 0 as the lowest bit. An example: a = 1 = 3'b001 -> 2a = 2 = 4'b0010

Select Code	Result (3-bits)	Carry bit
00	a + b	carry out
01	a - b	carry out
10	-a	0
11	2*a	Product MSB

Not worry about overflow

0000
0111
1000



Question #5 (Extra Credit - 5 pts)

Implement a 4-bit Gray code +1 incrementor using building blocks (no gates). The 4-bit Gray codes are shown below.

Decimal Number	Gray Code
0	0000
1	0001
2	0011
3	0010
4	0110
5	0111
6	0101
7	0100
8	1100
9	1101
10	1111
11	1110
12	1010
13	1011
14	1001
15	1000

Arbiter

0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111