Please print in pen:

Waterloo Student ID Number:

WatIAM/Quest Login Userid:

UNIVERSITY OF WATERLOO

Examination Midterm Fall 2022 CS 251

Closed Book

Candidates may bring no aids (no calculators).

Times:	Tuesday 2022-10-25 at 16:30 to 18:20 (4:30 to 6:20PM)
Duration:	1 hour 50 minutes (110 minutes)
Exam ID:	5122107
Sections:	CS 251 LEC 001-003
Instructors:	Stephen Mann, Xiao-Bo Li

Instructors: Stephen Mann, Xiao-Bo Li

Student Signature: ______________________________

UW Student ID Number: ____________________

Number of Exam Pages (including this cover sheet)	22 pages
Exam Type	Closed Book
Additional Material Allowed	NO ADDITIONAL MATERIALS ALLOWED

Marking Scheme (for graders' use only):

Question	Max	Score	Grader
1	10		
2	5		
3	10		
4	6		
5	4		
6	3		
7	12		
8	6		
9	4		
10	5		
11	6		
12	8		
13	5		
14	3		
15	8		

Maximum Total Score: 95

Total Score:

Instructions:

- If you believe there is an error in the exam, notify a proctor. An announcement will be made if a significant error is found.
- It is your responsibility to properly interpret a question. **Do not ask questions regarding the interpretation of a question**; they will not be answered and you will only disrupt your neighbours. If there is a non-technical term you do not understand you may ask for a definition. If you are confused about a question, state your assumptions and proceed to the best of your abilities.
- If you require more space to answer a question, there are blank pages at the end you may use instead, but you must **clearly indicate** in the provided answer space that you have done so.
- **Do not detach any pages and do not write on the QR codes**
- In circuit drawings, assume you always have access to the inputs or their complements.
  Your circuits may also be marked for efficiency.
- You may find this Powers of 2 table useful:

2^0	1
2^1	2
2^2	4
2^3	8
2^4	16
2^5	32
2^6	64
2^7	128
2^8	256
2^9	512
2^{10}	1024

Points on this page: 10

1. (10 points) True/False

(a) (10 points) Which of the following statements are true/false (circle one of True or False to the left of each statement)? +1 for each correct, −1 per each wrong, 0 for each left unanswered.

1. True False A CMOS NAND gate has more transistors than a CMOS AND gate.
2. True False With CMOS, a 3-input NAND gate uses the same number of transistors as a 2-input AND gate.
3. True False IEEE 754 floating point representation stores the exponent in biased notation.
4. True False D-flip-flops are used in the implementation of finite state machines to store inputs and state bits.
5. True False The 64-bit ALU performs subtraction of $A - B$ as $A + \bar{B} + 1$
6. True False A Decoder is used to implement the read operation in the Register File.
7. True False Dynamic RAM is a form of memory that remembers its values when power is turned off.
8. True False A tri-state buffer can have the following three possible output values: 0, 1, short-circuit
9. True False An IEEE floating point number stores its significand in 2's complement notation.
10. True False Any Boolean function can be represented as a sum of products of literals.
11. True False A Don't Care in the output of a truth table means that the output is "floating".

2. (5 points) Consider the following ARM code:

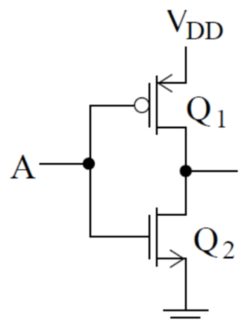
```
000: ADDI X0,X31,#6
004: SUBI X1,X31,#2
008: CBZ X0,#2
012: ADD X0,X0,X0
016: ADD X2,X1,X0
020: B #6
```

Assuming we start executing this code at line 000 and we finish execution of line 020, what are the contents of each register, and what is the contents of the PC?

- X0:
- X1:
- X2:
- PC:

Points on this page: 5

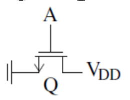
3. (10 points) **(a)** (2 points)

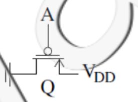


If $A = 1$ then F connects to

(a) A (b) Power (c) Ground (d) b and c (e) None of the above

You may find the following helpful.





NMOS

Input	$A = 0$	$A = 1$
Resistance	High	Low

PMOS

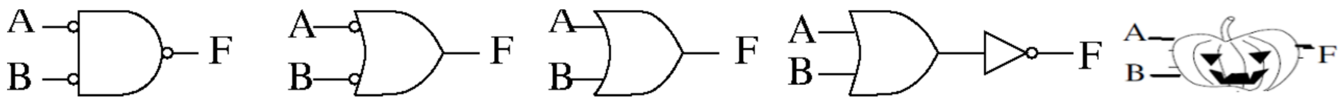
Input	$A = 0$	$A = 1$
Resistance	Low	High

(b) (2 points) Give the *unreduced, minterm* expression for F based on the following truth table.

A	B	C	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$F =$

(c) (1 point) Circle the gate(s)/circuit(s) that are equivalent to an NAND gate.

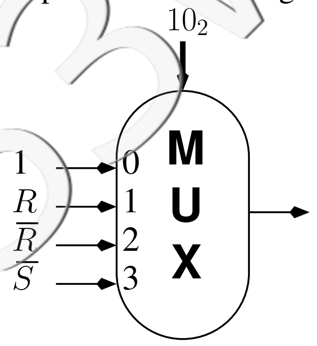


(d) (2 points) The output, C_{out}, of the binary Full Adder is given by the reduced function:

$$C_{out} = ac_{in} + bc_{in} + ab$$

Implement the circuit for C_{out} using only a minimal number of NAND gates. Be sure to label the output of your circuit.

(e) (1 point) Label the value of the output of the following circuit:



(f) (2 points) Give one reason we might prefer DRAM to SRAM; give one reason that we might prefer SRAM to DRAM.

Points on this page: 6

4. (6 points) In this question you are analyzing the following function F

$$F = \overline{\overline{A+C}+BC} + \overline{B \oplus C}$$

Give the truth table for F and deduce **a minimal** expression for F. You may use the space provided below for intermediate terms.

A	B	C	$A+C$	F
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

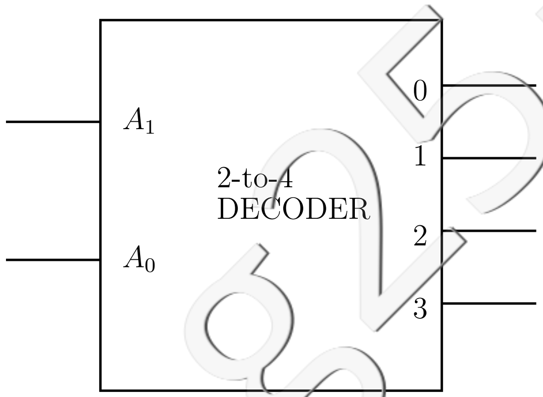
$F =$

Points on this page: 4

5. (4 points) Implement the output function *F* described in the following truth table using the inputs *B* and *C* and a 2×4 decoder. You may also use one additional *AND gate or OR gate only* (this additional gate may have up to 4 inputs). Assume you have access to the inputs or their complements and hardwired binary '0' or '1'.

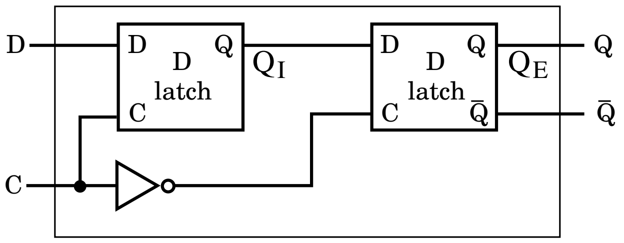
B	C	F
0	0	0
0	1	1
1	0	0
1	1	1

You may label the inputs or outputs of the decoder as needed, but you may not modify the internal implementation of the decoder. Be sure to label the output of your circuit.

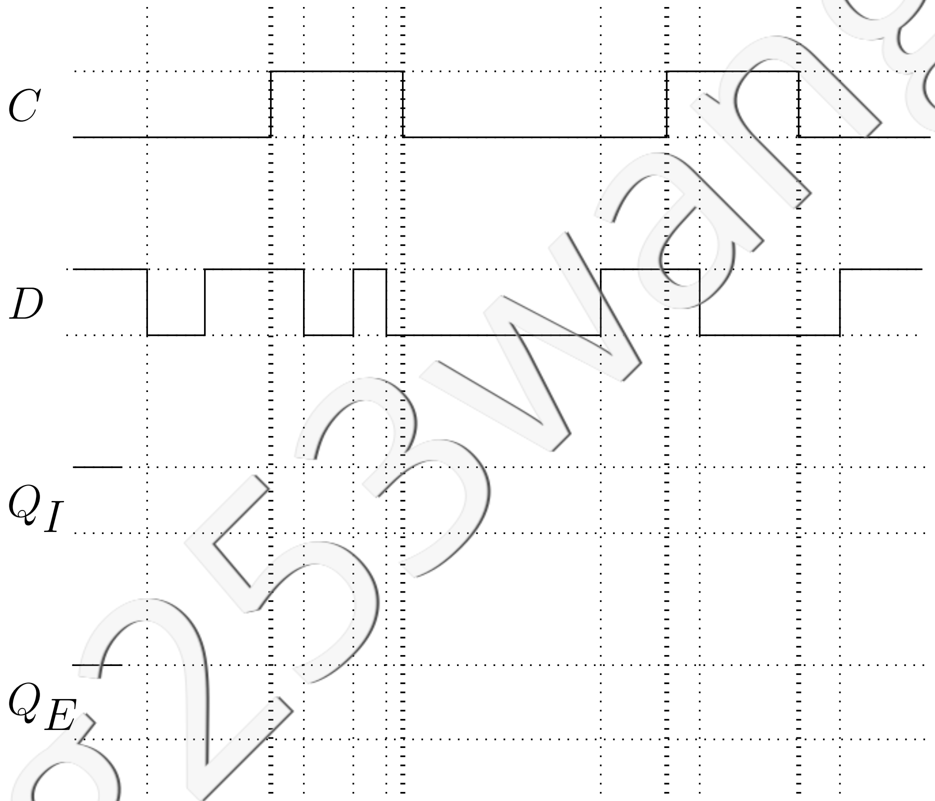


Points on this page: 3

6. (3 points) Consider a D flip flop:



In the figure below are traces of *D* and *C*; draw the resulting traces of Q_I and Q_E.



7. (12 points) **(a)** (4 points) Given the 8-bit binary number 1010 0101, give its decimal equivalent if these eight bits are interpreted as

1. an 8-bit unsigned number:
2. an 8-bit signed magnitude number:
3. an 8-bit 2's complement number:
4. an 8-bit biased number with bias 127:

(b) (2 points) Compute the following bitwise logical operations.

```
    1010 1101            1010 1101
AND 0000 1111        XOR 0000 1111
-------------        -------------
```

(c) (4 points) Using 8-bit 2's complement addition, perform the following computations. For each computation, circle if an overflow occurred or not.

```
  1010 0101           1010 0101
 +1011 1101          +0101 1010
 ----------          ----------
```

Overflow: YES \ NO   YES \ NO

(d) (2 points) Sign extend the following 4-bit 2's complement number to an 8-bit 2's complement number, and give the signed decimal value of the extended number.

Given number : 1001

Sign extension : **Signed Decimal Value :**

Points on this page: 6

8. (6 points) **(a)** (4 points) The following number is given as a 32-bit, IEEE normalized floating point number with biased exponent. Convert it to a signed decimal number.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1	1	0	0	0	0	1	1	0	0	0	0	0	1	1	1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

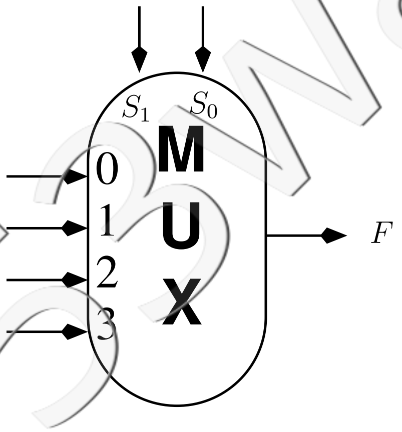
(b) (2 points)

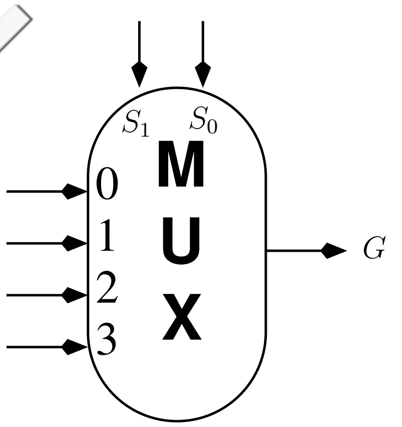
Compute the normalized binary floating point number that represents the value 1.1375×10^1
You do not need to translate the number to IEEE format.

9. (4 points) Consider the following truth table:

A_2	A_1	A_0	F	G
0	0	0	1	1
0	0	1	0	1
0	1	0	1	0
0	1	1	1	0
1	0	0	1	0
1	0	1	1	0
1	1	0	0	1
1	1	1	1	0

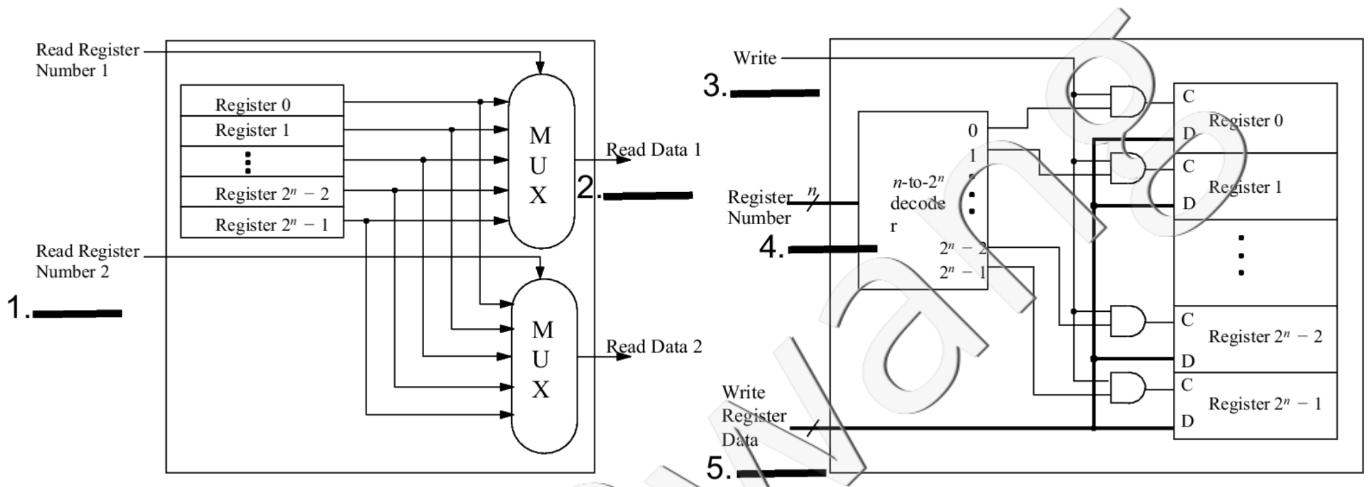
Implement a circuit for F and G using **only** the 4×1 multiplexors provided below. You may use the inputs or their complements or a hardwired '0' or'1'. You may use the inputs multiple times if needed. You may **not** use any additional gates on the inputs or outputs.





Points on this page: 5

10. (5 points) Below is the diagram for the Register File showing both read and write operations. Suppose we have an ARM 32-bit architecture (i.e, each register stores 32 bits of data) that has 64 registers in the Register File. In the figure, state the number of bits on the input or output indicated on the dark lines numbered 1-5.



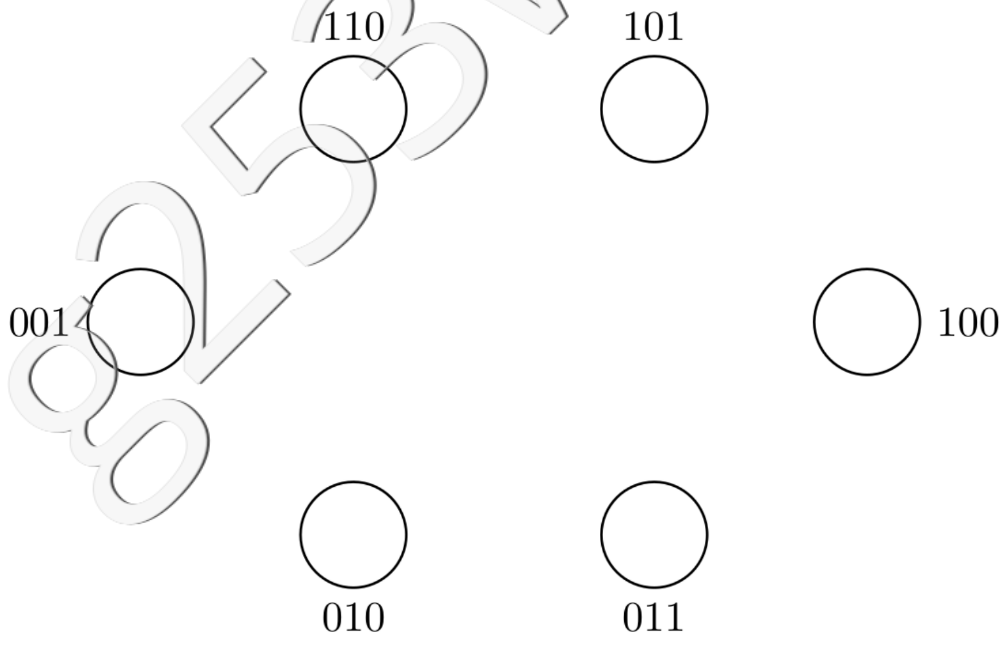
Points on this page: 6

11. (6 points) Consider the following next-state table and corresponding output table:

S_2	S_1	S_0	A	B	S_2'	S_1'	S_0'
0	0	0	X	X	X	X	X
0	0	1	X	X	0	1	0
0	1	0	X	0	0	1	1
0	1	0	X	1	1	0	0
0	1	1	X	X	0	1	1
1	0	0	0	X	0	1	1
1	0	0	1	X	1	0	0
1	0	1	0	1	1	1	0
1	0	1	1	1	0	0	1
1	0	1	X	0	1	0	0
1	1	0	X	X	0	0	1
1	1	1	X	X	X	X	X

S_2	S_1	S_0	Y	Z
0	0	0	X	X
0	0	1	0	0
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	0
1	1	0	0	1
1	1	1	X	X

Draw the finite state machine for this next state and output table using the states given below.



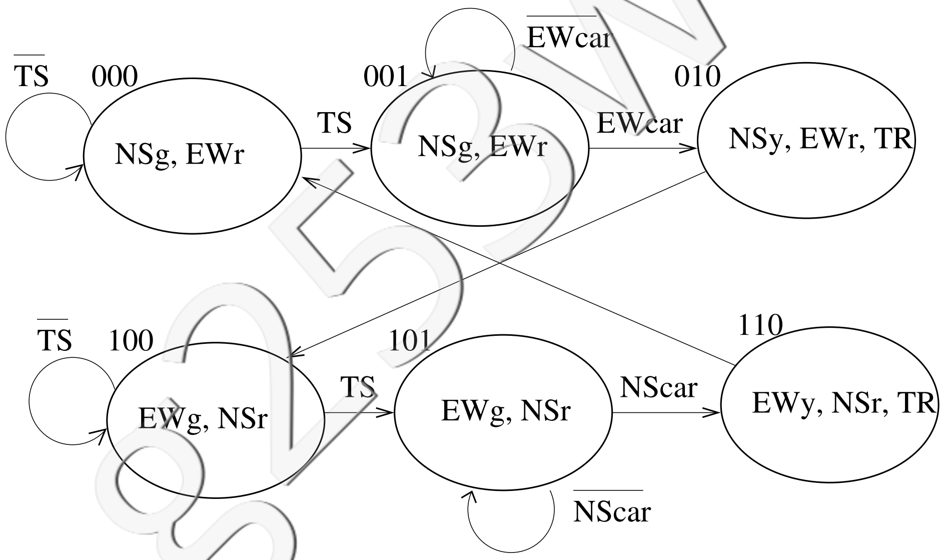
Please initial:

Points on this page: 8

12. (8 points) Provided below is the *Extended Traffic Light Controller* finite state machine studied in class. You are to extend the light to have left turn arrows in the North/South directions (the East/West directions will not get left turn arrows). One new input, **NSLeft**, and one new output, **NSLftLtG**, have been added to the system. NSLeft will be high if there is a car in the left turn lane in either the North or South directions; NSLftLtG should be high to turn on the left turn arrow on in the NS directions.

Modify the FSM diagram below to incorporate the new input and outputs according to the following description.

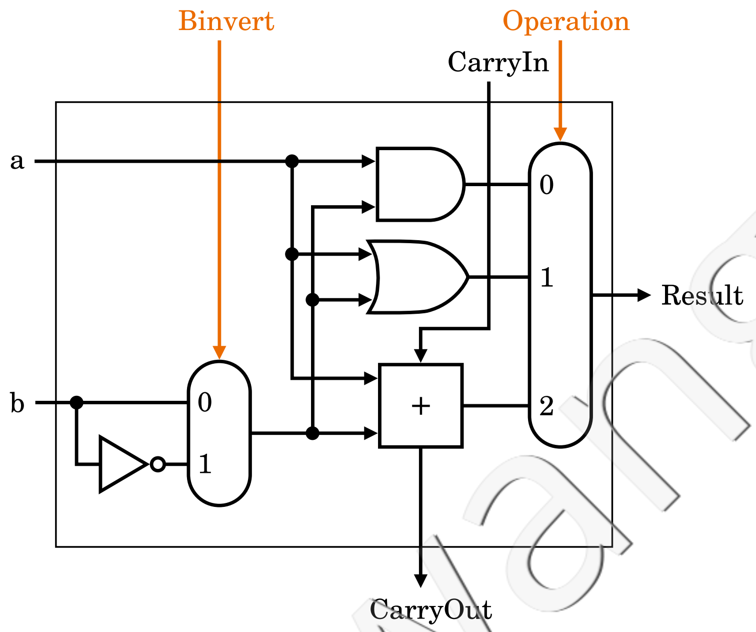
- When in state 101, if either NScar or NSLeft are high, you should transition to state 110.
- When in state 110, if NSLeft is high, you should transition to a state where NSLftLtG is high, NSr is high, and EWr is high. You should stay in this state for one clock cycle and then transition into state 000. Do not worry about TR or TS (the yellow light is stealing one clock cycle from the NS direction).
- When in state 110, if NScar is high but NSLeft is low, the FSM should function as before.



Inputs: NScar, EWCar, TS, NSLeft
Outputs: NSy, NSr, EWg, EWy, EWr, TR, NSleftLtG

Modify the diagram as needed to incorporate the changes described above. **You may modify states/transitions or add new states/transitions as required**. Be sure to label the outputs that are true in any new state as well as label new the state itself.

13. (5 points) Consider the 1-bit ALU studied in class:



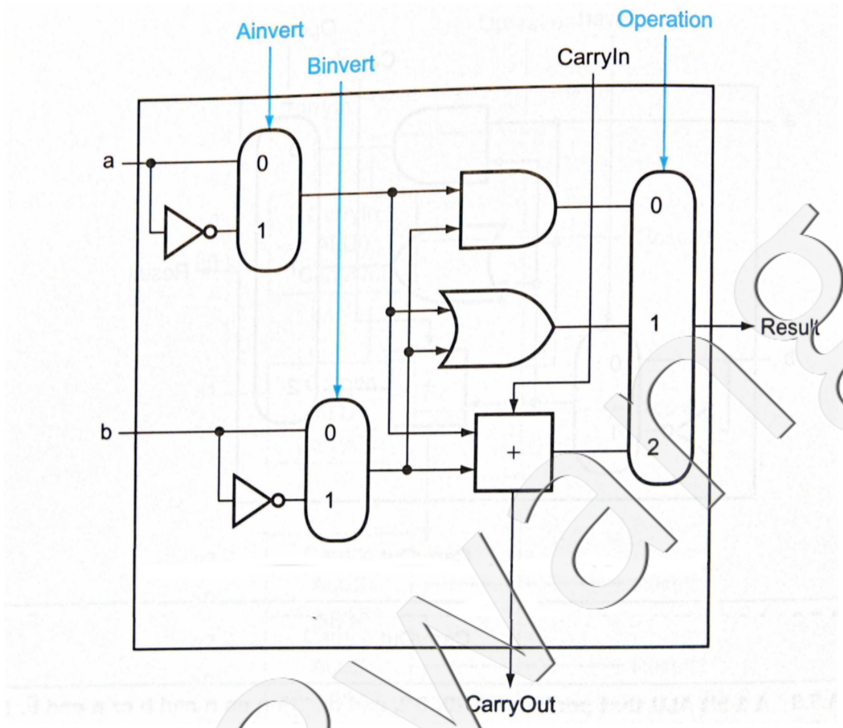
Fill in the values of CarryOut and Result in the following table (Operation is specified in binary):

a	b	CarryIn	Operation	Binvert	CarryOut	Result
1	0	0	00	0		
1	0	0	01	0		
1	1	1	10	0		
0	1	1	01	1		
1	0	1	00	1		

Please initial:

Points on this page: 3

14. (3 points) Consider the 1-bit ALU studied in class that has Ainvert:



We would like to perform the **NAND** of **a** and **b** using the 1-bit ALU given above. In the table below, list the values of the ALU inputs needed to perform NAND.

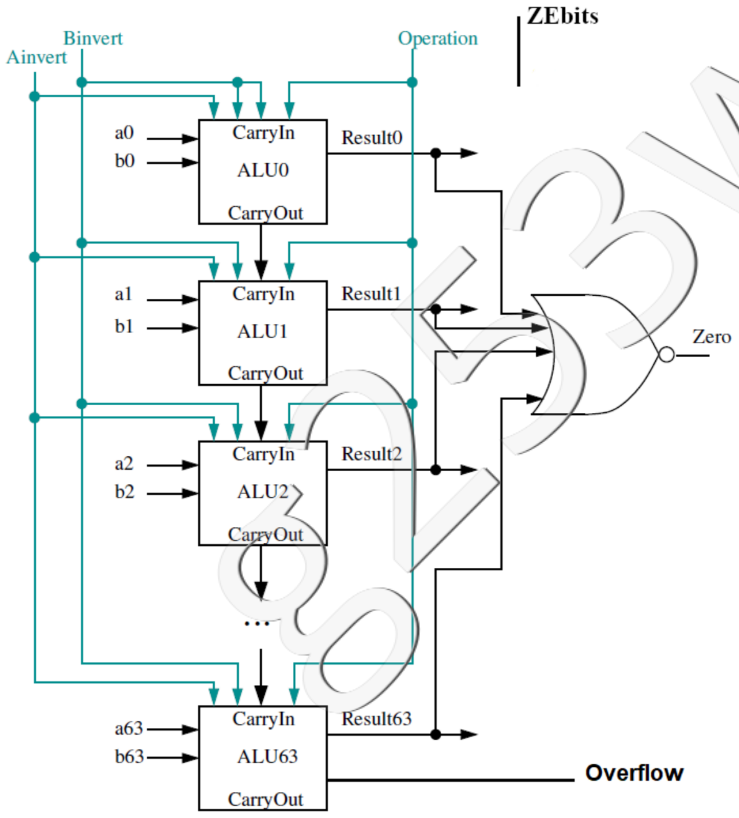
CarryIn	Operation	Ainvert	Binvert

15. (8 points) Below is the diagram for a 64-bit ALU. Modify this diagram to have the following three additional outputs. In the figure, label each output with the name given in bold in the question. You may use the Zero and Overflow bits in your solution if needed. The Result bits and operands are 2's complement numbers. You may use any of the result bits $R_0 \ldots R_{63}$ or input bits $A_0 \ldots A_{63}$, $B_0 \ldots B_{63}$, A_{invert}, B_{invert}, $Operation$, **ZEbits** (described in part c).

(a) (2 points) **Neg-Even (NE)**: This output should be HIGH (1) when the signed 64-bit result is a negative, even number.

(b) (4 points) **Greater-than-Neg-5 (GTN5)**: This output should be HIGH (1) when the signed 64-bit result is a value that is strictly greater than -5.

(c) (2 points) **ZEbits**: When this *input* is high, all the even bits of the result ($R_0, R_2, R_4, R_6, \ldots$) should be forced to be 0 while all the odd bits should remain the same. When ZEbits=0, the output should be unchanged. You will only get part marks if you use a MUX for this question.



Please initial:

This page is intentionally left blank for your use. Do not remove it from your booklet. If you require more space to answer a question, you may use this page, but you must **clearly indicate** in the provided answer space that you have done so.

This page is intentionally left blank for your use. Do not remove it from your booklet. If you require more space to answer a question, you may use this page, but you must **clearly indicate** in the provided answer space that you have done so.

This page is intentionally left blank for your use. Do not remove it from your booklet. If you require more space to answer a question, you may use this page, but you must **clearly indicate** in the provided answer space that you have done so.