1. Answers:

f = (A + C' + D")(B' + C' + D)(A + B' + C')

f = A'B'A + A'B'C' + A'B'D" + A'C'C' + A'C'D + A'D"A + A'D"C' + A'D"D + B'C'C' + B'C'D + B'DA + B'DC' + B'DD"

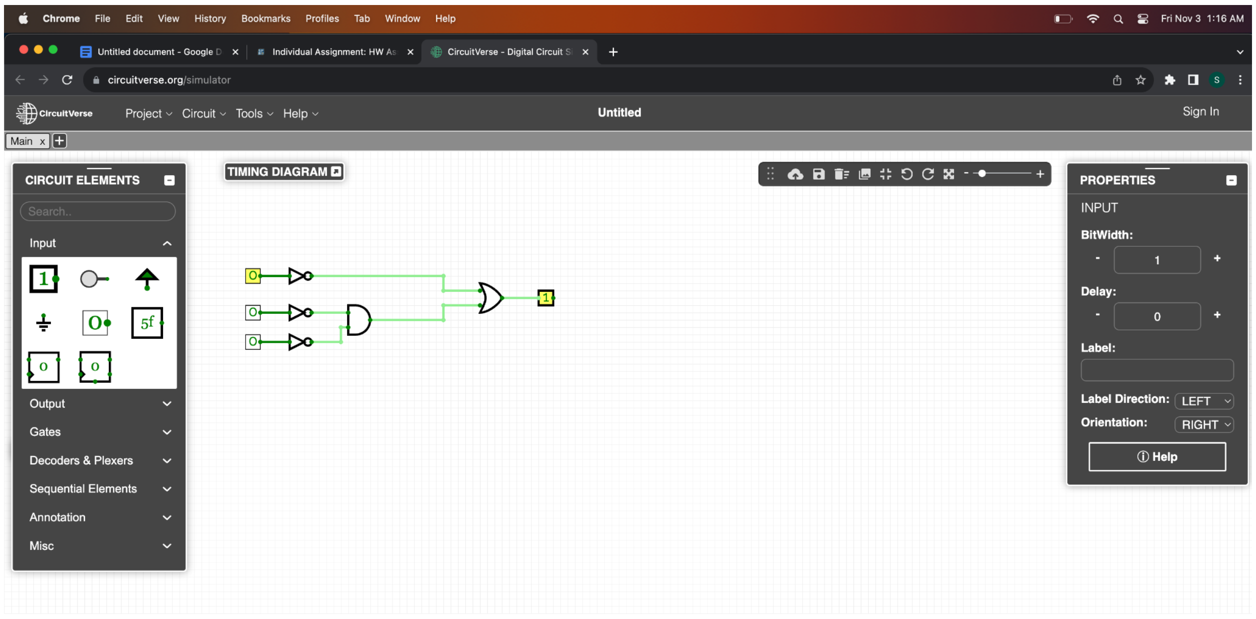
f = A'B'A + A'B'C' + A'B'D" + A'C' + A'C'D + A'D" + B'C'C' + B'C'D + B'DA + B'DC' + B'DD" [Apply the Idempotent Law (A + A = A)]

f = A'B' + A'C' + A'D" + A'C'D + A'D" + B'C' + B'DA + B'DC' + B'DD" [Apply the Absorption Law (A + AB = A)]
f = A'B' + A'C' + A'D" + B'C' + B'DA + B'DC' + B'DD" [Apply the Absorption Law]
f = A'B' + A'C' + A'D" + B'C' + B'DA + B'DC' + B'DD"
Applying the Absorption Law (A + AB = A)
A'B' + A'DA = A'B' + A'
Applying the Absorption Law (A + AB = A)
f = A' + A'C' + A'D" + B'C' + B'DA + B'DC' + B'D'
Applying the Absorption Law (A + AB = A)
A'D" + B'DC' = A' + B'DC'
f = A' + A'C' + B'C' + B'DA + B'D + B'DC'
Applying the Absorption Law (A + AB = A)
A'C' + B'DC' = A' + B'DC'
Applying the Absorption Law (A + AB = A)
B'DA + B'D = B'DA + B'D

f = A' + A' + B'C' + B'DA + B'DC' + B'DA + B'D
Applying the Idempotent Law (A + A = A)
A' + A' + B'C' + B'DA + B'DC' + B'DA + B'D = A' + B'C' + B'DA + B'DC' + B'D
f = A' + B'C' + B'D
Applying the Absorption Law (A + AB = A)
B'C' + B'D = B'C'
f = A' + B'C'

A	B	C	A'	B'C'	A'+B'C'
0	0	0	1	0	1
0	0	1	1	0	1
0	1	0	1	0	1

0	1	1	1	1	1
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	0	1	1


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri Nov 3 1:16 AM
Untitled document - Google D
Individual Assignment: HW As
CircuitVerse - Digital Circuit S
circuitverse.org/simulator
CircuitVerse
Project Circuit Tools Help
Untitled
Sign In
Main
CIRCUIT ELEMENTS
Search..
Input
Output
Gates
Decoders & Plexers
Sequential Elements
Annotation
Misc
TIMING DIAGRAM
0
0
0
1
PROPERTIES
Help

INPUT

BitWidth: 1

Delay: 0

Label:

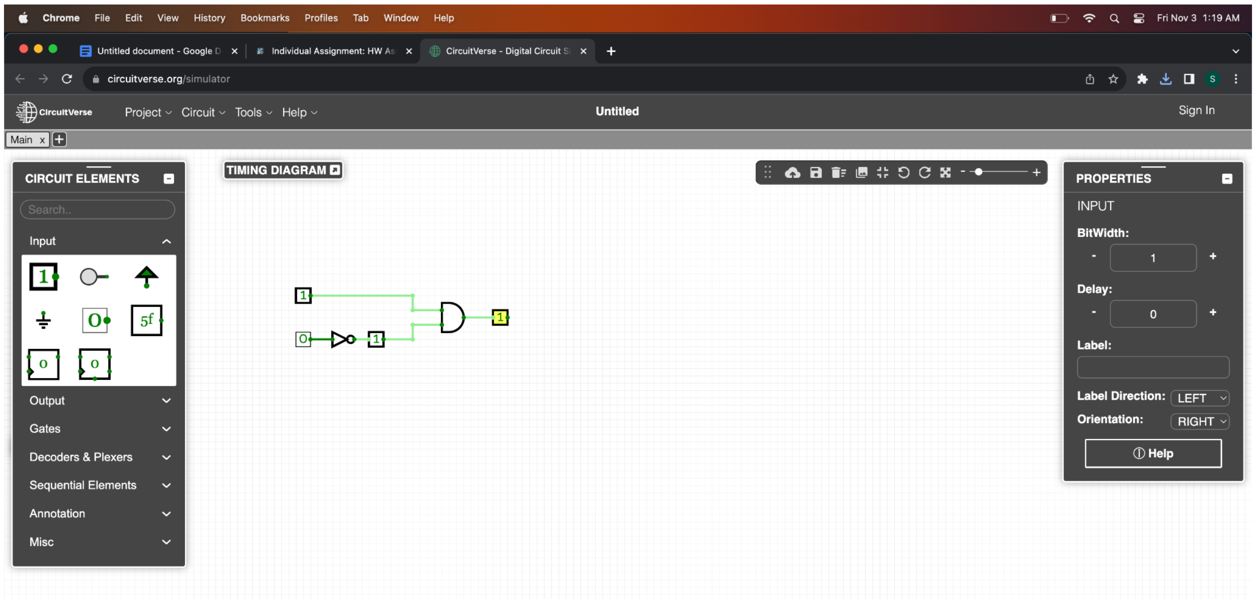
Label Direction: LEFT

Orientation: RIGHT

B. f = (Z + X)(Z' + Y')(Y' + X)

f = ZZ'Y' + ZZ'X + XY'Y' + XY'X [Using Distributive Law]
=ZZ'Y' + ZZ'X + XY'Y' + XY'X: [applying Identity Law , ZZ' =0]
=0Y' + 0X + XY'Y' + XY'X
=XY' + XY'X [Applying Idempotent Law]
=XY'(1+X) [Factoring out XY']
=XY' [Applying Identity Law

X	Y	Y'	XY'
0	0	1	0
0	1	0	0
1	0	1	1
1	1	0	0

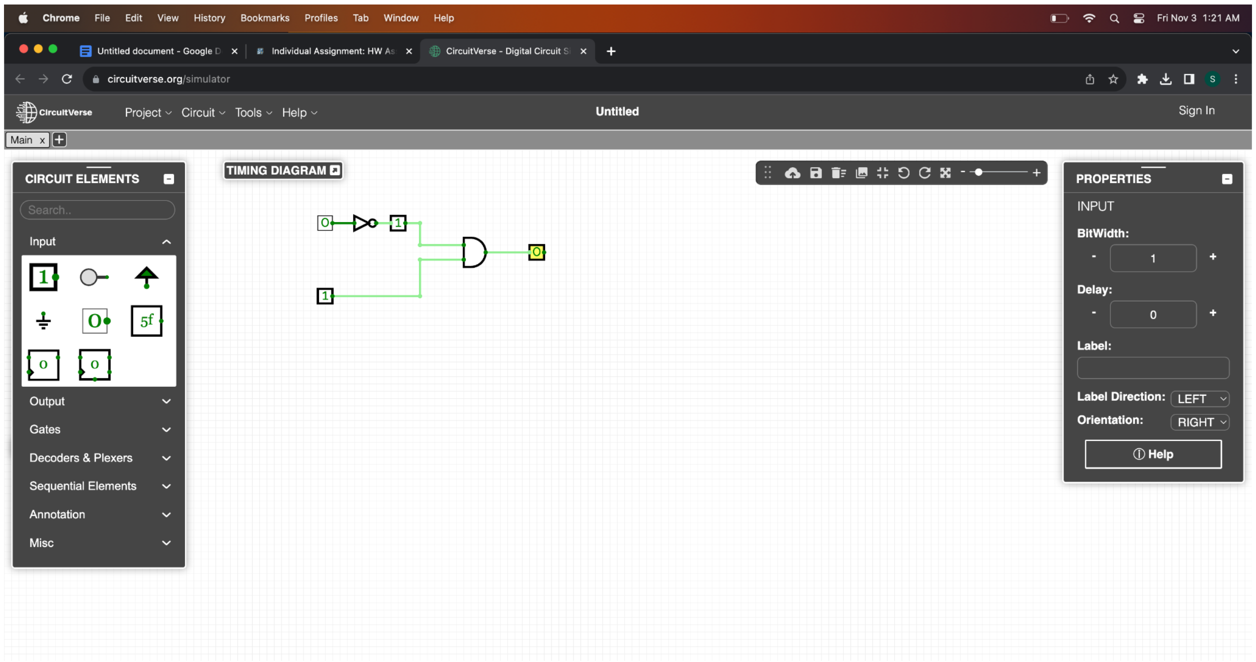

Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri Nov 3 1:19 AM
Untitled document - Google D
Individual Assignment: HW As
CircuitVerse - Digital Circuit S
circuitverse.org/simulator
CircuitVerse
Project Circuit Tools Help
Untitled
Sign In
Main
CIRCUIT ELEMENTS
Search..
Input
Output
Gates
Decoders & Plexers
Sequential Elements
Annotation
Misc
TIMING DIAGRAM
PROPERTIES
INPUT
Help

BitWidth: 1
Delay: 0
Label:
Label Direction: LEFT
Orientation: RIGHT

C. (X'Y) + (X'Y'Z')= X'Y(1 + Z') [Factoring X’Y]
=X'Y(1) [Apply the Identity Law (A + 1 = 1)]
= X'Y

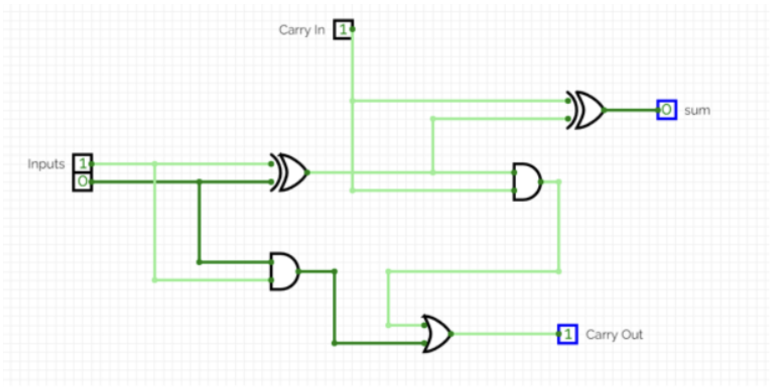
Truth table:

X	Y	X’	X’Y
0	0	1	0
0	1	1	1
1	0	0	0
1	1	0	0


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Fri Nov 3 1:21 AM
Untitled document - Google D
Individual Assignment: HW As
CircuitVerse - Digital Circuit S
circuitverse.org/simulator
CircuitVerse
Project Circuit Tools Help
Untitled
Sign In
Main
CIRCUIT ELEMENTS
Search..
Input
Output
Gates
Decoders & Plexers
Sequential Elements
Annotation
Misc
TIMING DIAGRAM
PROPERTIES
Help

PROPERTIES
INPUT
BitWidth: 1
Delay: 0
Label:
Label Direction: LEFT
Orientation: RIGHT

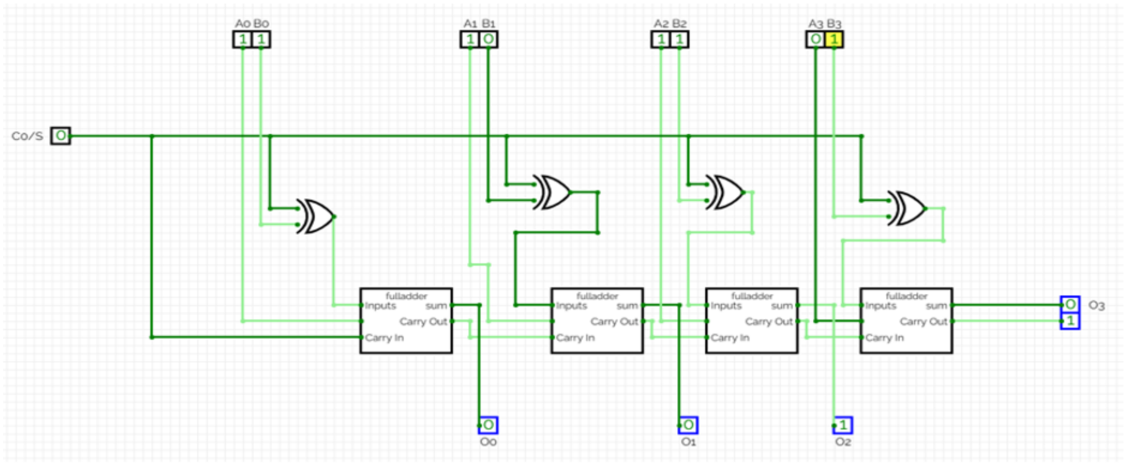
2. Full Adder Design:
   Given:
   Design of a single-bit full adder with three inputs: A, B, and a carry-in (C_in)
   Two outputs: the sum (S) and a carry-out (C_out).
   Logic Operations: AND, OR, and XOR gates.



4-bit Adder-Subtractor Design:
To execute an addition (S=0):
Bits of B number are passed directly to the adder.
Carry-in of the least significant bit (C_0) = 0.
Then, A and B are added by the adder.

To execute a subtraction (S=1):
To get 1’s complement:
Input B is inverted
To get 2’s complement:
Least significant bit adder gets Carry-in of 1.

A is added to the 2’s complement of B. [Equivalent to subtracting B from A]



DESIGN TESTING:
For the given test cases:
(a) A + B = 7 + (-3) = 4
A = 7 = 0111 [in binary]
B = -3 [2's complement of 3]
3 = 0011 [in binary 2's complement of 3]
Inverting all bits then adding 1 = 1101

When S=0 (Addition),
A = 0111
B = 1101
Expected result = 1000 [8 in binary].
The result is negative(indicated by sign bits)
Therefore, the result is -8. For the Absolute value:
2's complement of 1000 = 1000 [8 in decimal]

(b) A - B = -6 - (-1) = -5
A = -6 [2's complement of 6 = 1010 in binary]
B = -1 [2's complement of 1 = 1111 in binary]

When S=1 (Subtraction),

A = 1010
B = 1111
Expected result = 1001, which is -7 in decimal.

3. Instructions:
"Load X" = 0001[X represents the address to load from]

"Add X" is 0011 [X represents the address to add from]
"Store X" is 0010 [X represents the address to store to]
Assembly instructions:

- Load 104
- Add 105
- Store 106

When translated to machine language:

- Load 104 is translated to:

- [ ] Binary: 0001 0000 1000
- [ ] Hex: 1 08
- [ ] Add 105 translates to:
- [ ] Binary: 0011 0000 1001
- [ ] Hex: 3 09
- [ ] Store 106 translates to:
- [ ] Binary: 0010 0000 1010
- [ ] Hex: 2 0A

The required machine language = 1 08 3 09 2 0A

These instructions are represented into 4-bit format (based on the provided table) Then, remaining 12 bits are used for address in the following.

Instruction	4-bit code	Address	Machine Code
Load	0001	0000 1000	0001 0000 1000
Add	0011	0000 1001	0011 0000 1001
Share	0010	0000 1010	0010 0000 1010

Now, designing a simple circuit for the Load instruction as an example. The output will be active (1) when the 4-bit code is 0001 for a "Load" operation. Here, Inputs: A, B, C, D Outputs: Y

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

0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0

For the Load operation: $Y = D' \times C' \times B' \times A$

Now, For building the circuit, using:
NOT gates on inputs D, C, and B.
AND gate to combine the outputs of these NOT gates with input A.
Output of the circuit is 1 [only when the 4-bit code matches that of the Load operation]. Add and Store have similar designs.