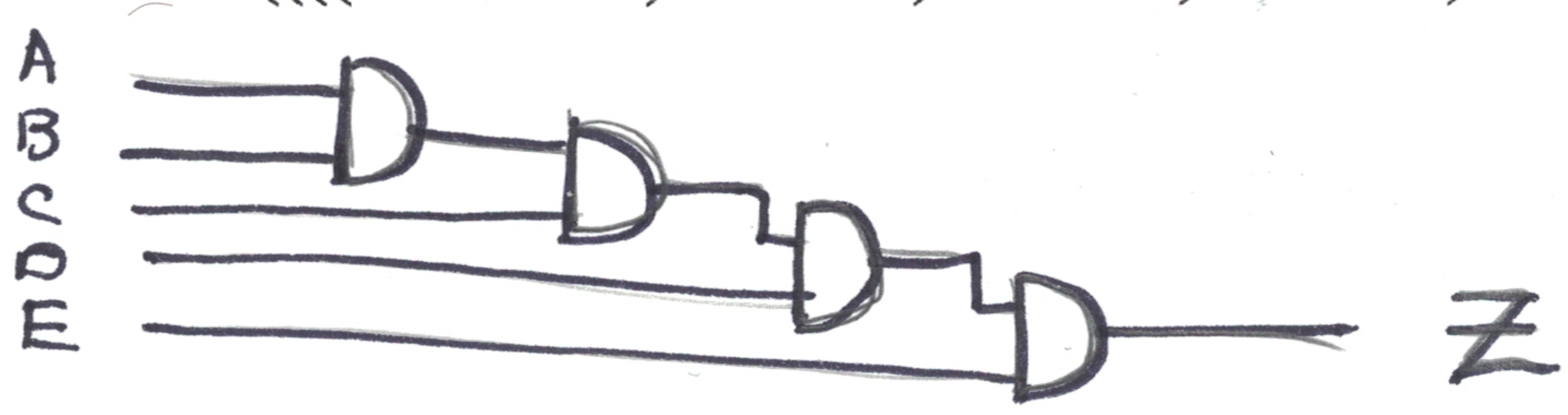
James Vrionis
JVrionis@ucsc.edu
1562232
CMPE 12/L
1-24-17

Homework #2
CMPE-12L

(5 pts) Build a 5-input and gate out of 2-input and gates.
ANSWER: Using the fact that **AND** is an associative logical operator, that means for some output Z = **((((A** AND **B)** AND **C)** AND **D)** AND **E)** with inputs A,B,C,D and E.



$\therefore ((((A \cdot B) \cdot C) \cdot D) \cdot E)$

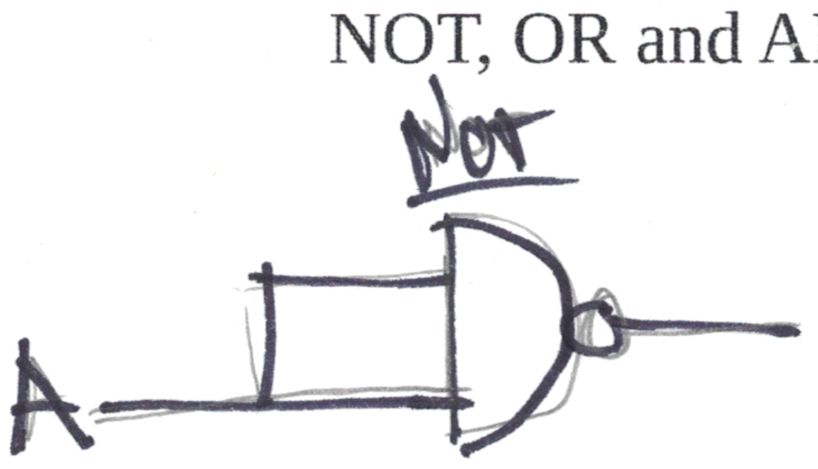
(5 pts) How many output lines will a five-input decoder have?
ANSWER: A decoder takes its inputs and converts to a "one-hot" output. With the inputs, you are selecting which output is asserted. The number of outputs in a decoder is equal to 2^n with n equal to the number of inputs. Therefore, a five-input decoder would have 32 outputs.

(5 pts) How many output lines will a 16-input multiplexer have? How many select lines will this multiplexer have?
ANSWER: A multiplexer selects one of many inputs and presents it on the output. There is one output in a multiplexer. The number of select lines necessary in a multiplexer is $\log_2$ of the number of inputs. 1 select line can select between 2 inputs, 2 can select between 4 inputs, 3 can select between 8, and 4 can select from 16. Thus, this multiplexer with have 4 select lines.

(5 pts) You know a byte is 8 bits. We call a 4-bit quantity a nibble. if a byte-addressable memory has a 14-bit address, how many nibbles of storage are in this memory?
ANSWER: $= 2^{14}$ $\because$ 14-bit address means that you can address 2^{14}
$= 2 * 2^{14}$ $\because$ Each memory location cantains 1 byte, equal to two nibbles.
$= 2^{15}$ $\because$ $2^{15} = 2 * 2^{14}$
= 32868 nibbles of storage in this memory.

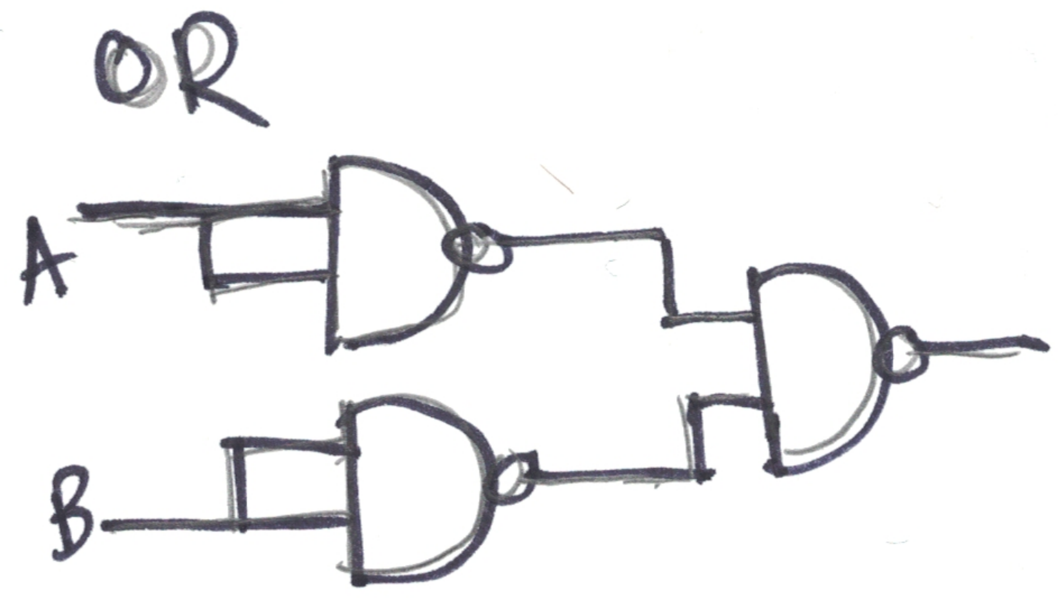
(15 pts) All Logic circuits can be created by NAND gates. Prove this by building logic circuits for NOT, OR and AND using only NAND gates.

NOT



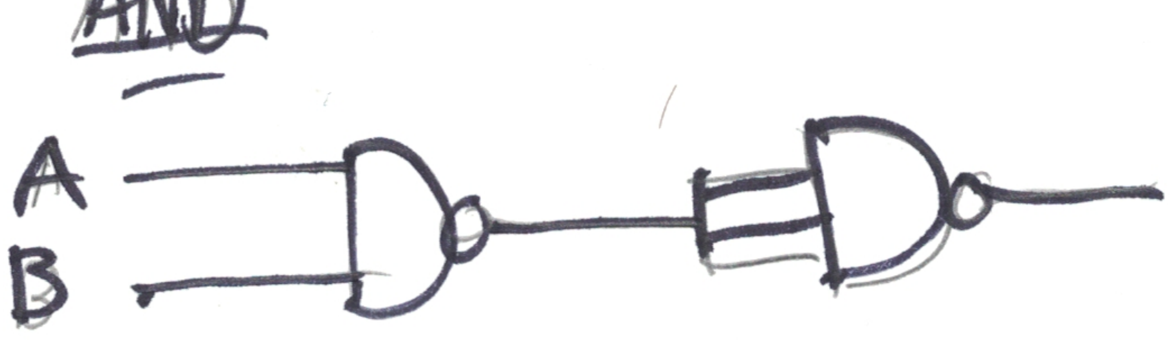
A	A'
0	1
1	0

OR



AB	A'	B'	A'B'	(A'·B')'	A+B
00	1	1	1	0	0
01	1	0	0	1	1
10	0	1	0	1	1
11	0	0	0	1	1

AND

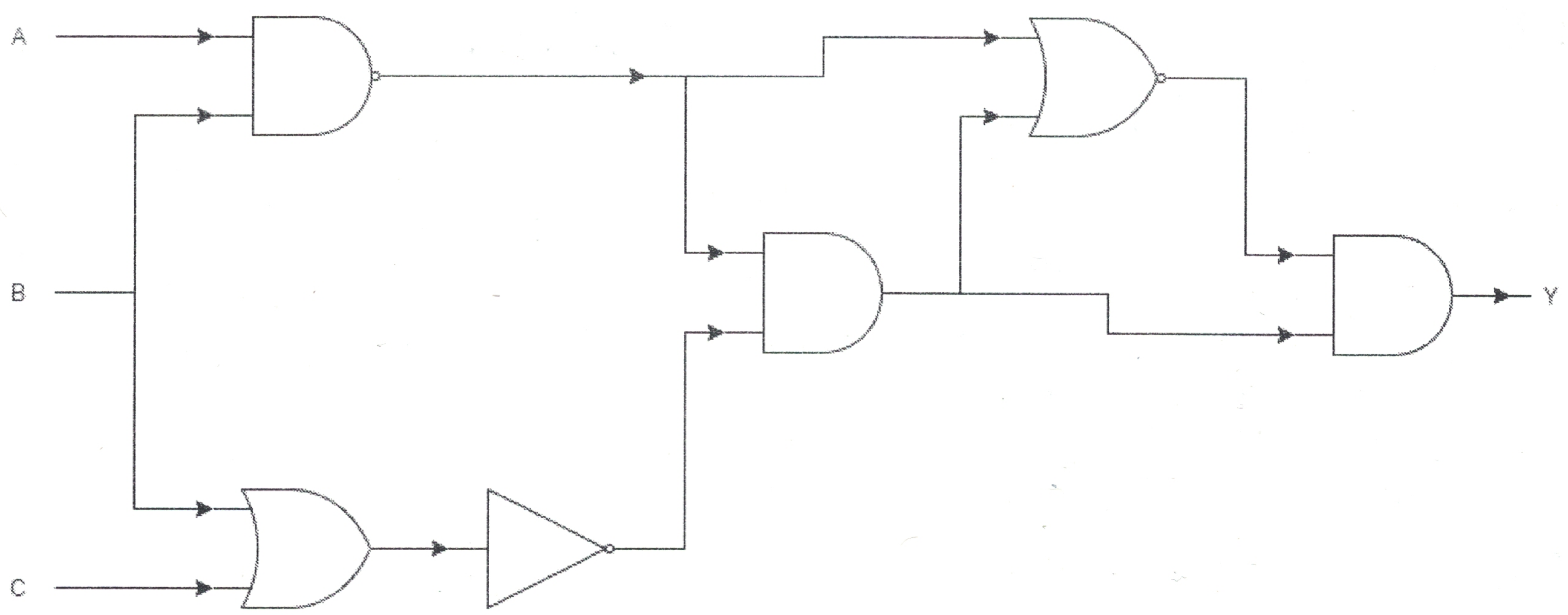


AB	A·B	(A·B)'	((AB)')'
00	0	1	0
01	0	1	0
10	0	1	0
11	1	0	1

(10 pts) Distinguish between a memory address and the memory's addressability.

Answer: Memory's addressability is the way in which a computer identifies memory locations. The width of the address bus determines how man memory locations can be addressed. For example: 1 bit address bus = 2 memory locations, 2 bit address bus = 3 memory locations, 3 bit address bus = 8 memory locations, and so on. Addressable memoy can be calculated using the number of storage locations * size of each storage location.

(15 pts) Give the logic circuit below, fill in the truth table for the output value Y.



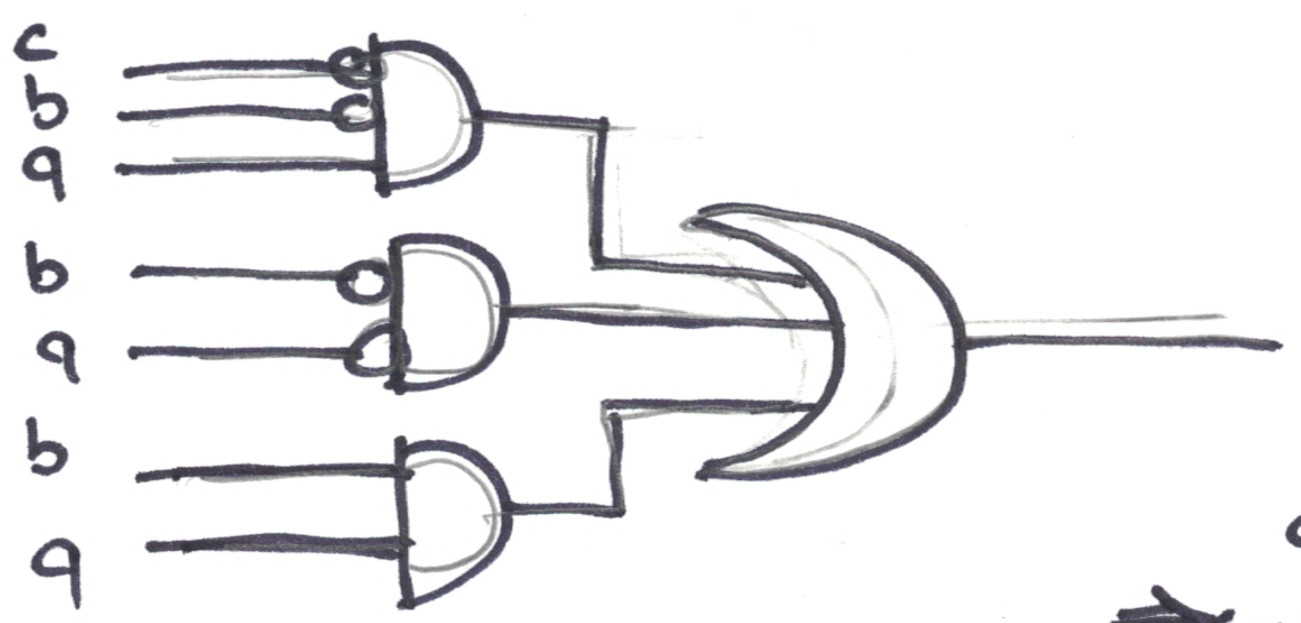
A	B	C	(A ·B)	(A ·B)'	(B+C)	(B+C)'	**(1)**	**(2)**	**(3)**	**(4)**	y = output
0	0	0	0	1	0	1	1	1	0	0	0
0	0	1	0	1	1	0	0	1	0	0	0
0	1	0	0	1	1	0	0	1	0	0	0
0	1	1	0	1	1	0	0	1	0	0	0
1	0	0	0	1	0	1	1	1	0	0	0
1	0	1	0	1	1	0	0	1	0	0	0
1	1	0	0	0	1	0	0	0	1	0	0
1	1	1	0	0	1	0	0	0	1	0	0

[(B+C)' ·(A ·B)'] = **(1)**
[[(B+C)' ·(A ·B)']+(A ·B)'] = **(2)**
[[(B+C)' ·(A ·B)']+(A ·B)'] ' = **(3)**
[[(B+C)' ·(A ·B)'] · [[(B+C)' ·(A ·B)']+(A ·B)'] '] = **(1)** · **(3)** = **(4)**
since the output **y = (4), y** is a column of all zeros.

9/10

(15 pts) Create the Logic gates for the truth Table below.

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



$a'b'c' + a'b'c + ab'c' + abc' + abc$

$\rightarrow a'b'(c+c') + ab(c+c') + ab$

$\therefore a'b' + ab + ab'c'$

$\therefore a'b'c' + a'b'c + ab'c' + abc' + abc \equiv a'b' + ab + ab'c'$

(25 pts) Convert the following numbers to binary and perform binary subtraction on them. Do not use additive inverse.

a. 39 – 22 => 00100111 – 00010110 => 00010001 => 17
b. 25 – 14 => 00011001 – 00001110 => 00001011 => 11
c. 39 – 12 => 00100111-00001100 => 00011011 => 27
d. 18- 11 => 00010010 – 00001011 => 0000111 => 7
e. 30 – 26 => 00011110- 00011010 => 00000100 => 4

a.
  00100111
− 00010110
  00010001 ✓

b.
  00011001
− 00001110
  00001011 ✓

c.
  00100111
− 00001100
  00011011 ✓

d.
  00010010
− 00001011
  00000111 ✓

e.
  00011110
− 00011010
  00000100 ✓