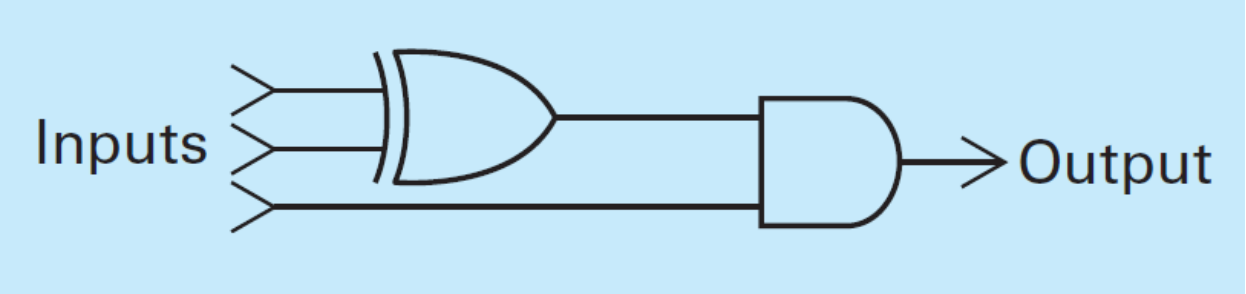
Junrui Hei
Eli Griffiths
Checked by groups 2 and 5

Data Storage- Worksheet

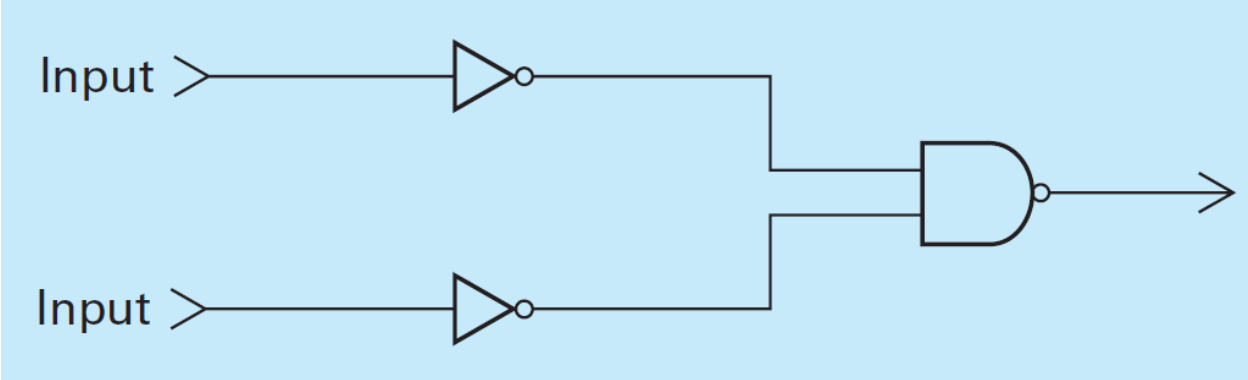
Logic Gates

1. What input bit patterns will cause the following circuit to produce an output of 1?



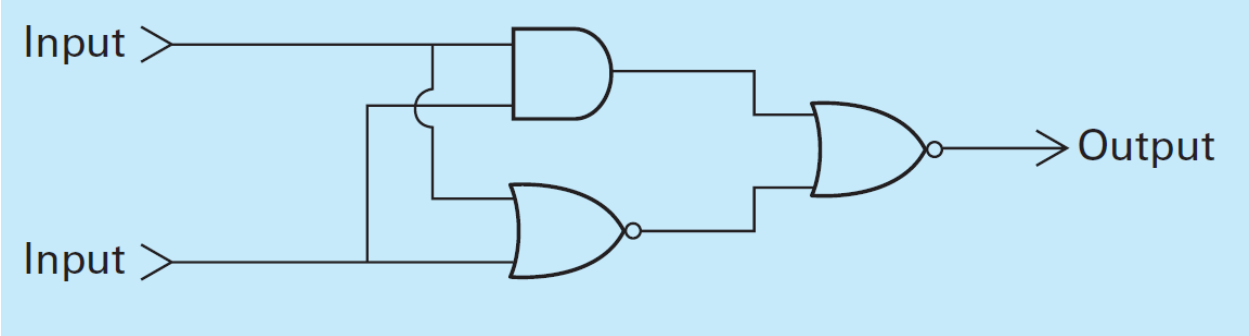
- **101 and 011**

2. If the output of an AND gate is passed through a NOT gate, the combination computes the Boolean operation called NAND, which has an output of 0 only when both its inputs are 1. The symbol for a NAND gate is the same as an AND gate except that it has a circle at its output. The following is a circuit containing a NAND gate. What Boolean operation does the circuit compute?



- **OR**

3. If the output of an OR gate is passed through a NOT gate, the combination computes the Boolean operation called NOR that has an output of 1 only when both its inputs are 0. The symbol for a NOR gate is the same as an OR gate except that it has a circle at its output. The following is a circuit containing an AND gate and two NOR gates. What Boolean operation does the circuit compute?



- **XOR**

Hexadecimal Notation

4. Use hexadecimal notation to represent the following bit patterns:
   a. `0110101011110010` ⇒ **`6AF2`**
   b. `111010000101010100010111` ⇒ **`E85517`**
   c. `01001000` ⇒ **`48`**

5. What bit patterns are represented by the following hexadecimal patterns?
   a. 5FD97 ⇒ **`0101 1111 1101 1001 0111`**
   b. 610A ⇒ **`0110 0001 0000 1010`**
   c. ABCD ⇒ **`1010 1011 1100 1101`**
   d. `0100` ⇒ **`0000 0001 0000 0000`**

ASCII Encoding for Text

6. Here is a message encoded in ASCII using 8 bits per symbol. What does it say? (*See* Appendix A)

   01000011 01101111 01101101 01110000 01110101 01110100
   01100101 01110010 00100000 01010011 01100011 01101001
   01100101 01101110 01100011 01100101

   Computer Science

7. Encode these sentences in ASCII:
   a. “Stop!” Cheryl shouted.
   00100010 01010011 01110100 01101111 01110000 00100001 00100010 00100000 01000011 01101000 01100101 01110010 01111001 01101100 00100000 01110011 01101000 01101111 01110101 01110100 01100101 01100100 00101110
   b. Does 2 + 3 = 5?
   01000100 01101111 01100101 01110011 00100000 00110010 00100000 00101011 00100000 00110011 00100000 00111101 00100000 00110101 00111111

Binary Notation

8. Convert each of the following binary representations to its equivalent base ten form:
   a. `0101` ⇒ **`5`**
   b. `1001` ⇒ **`9`**
   c. `1011` ⇒ **`11`**
   d. `0110` ⇒ **`6`**
   e. `10000` ⇒ **`16`**
   f. `10010` ⇒ **`18`**
9. Convert each of the following base ten representations to its equivalent binary form:
   a. 6 ⇒ **`0110`**
   b. 13 ⇒ **`1101`**
   c. 11 ⇒ **`1011`**
   d. 18 ⇒ **`10010`**
   e. 27 ⇒ **`11011`**
   f. 4 ⇒ **`0100`**
10. What is the largest numeric value that could be represented with three bytes if each digit were encoded using one ASCII pattern per byte? What if binary notation were used?

    The largest number stored in three bytes as ascii digits would be 999. The largest number stored in three bytes using binary would be 2^24 or 16,777,216.

11. An alternative to hexadecimal notation for representing bit patterns is **dotted decimal notation** in which each byte in the pattern is represented by its base ten equivalent. In turn, these byte representations are separated by periods. For example, 12.5 represents the pattern 0000110000000101 (the byte 00001100 is represented by 12, and 00000101 is represented by 5), and the pattern 100010000001000000000111 is represented by 136.16.7. Represent each of the following bit patterns in dotted decimal notation.
    a. `0000111100001111` ⇒ **`15.15`**
    b. `001100110000000010000000` ⇒ **`51.0.128`**
    c. `00001010101000` ⇒ **`10.160`**

12. Convert each of the following binary representations to its equivalent base ten form:
    a. 101010 ⇒ 42
    b. 100001 ⇒ 33
    c. 10111 ⇒ 23
    d. 0110 ⇒ 6
    e. 11111 ⇒ 31
13. Convert each of the following base ten representations to its equivalent binary form:
    a. 32 ⇒ 100000
    b. 64 ⇒ 1000000
    c. 96 ⇒ 1100000
    d. 15 ⇒ 1111
    e. 27 ⇒ 11011

Fractions in Binary

14. Convert each of the following binary representations to its equivalent base ten form:
    a. 11.01 ⇒ 3.25
    b. 101.111 ⇒ 5.875
    c. 10.1 ⇒ 2.5
    d. 110.011 ⇒ 6.375
    e. 0.101 ⇒ 0.625
15. Express the following values in binary notation:
    a. $4\frac{1}{2}$ ⇒ 100.1
    b. $2\frac{3}{4}$ ⇒ 10.11
    c. $1\frac{1}{8}$ ⇒ 1.001
    d. $\frac{5}{16}$ ⇒ 0.0101
    e. $5\frac{5}{8}$ ⇒ 101.101
16. Perform the following additions in binary notation:
    a. 11011 + 1100 = 100111
    b. 1010.001 + 1.101 = 1011.110
    c. 11111 + 0001 = 100000
    d. 111.11 + 00.01 = 1000

appendix

A

ASCII

The following is a partial listing of ASCII code, in which each bit pattern has been extended with a 0 on its left to produce the 8-bit pattern commonly used today. The hexadecimal value of each 8-bit pattern is given in the third column.

Symbol	ASCII	Hex	Symbol	ASCII	Hex	Symbol	ASCII	Hex
line feed	00001010	0A	>	00111110	3E	^	01011110	5E
carriage return	00001011	0B	?	00111111	3F	_	01011111	5F
space	00100000	20	@	01000000	40	`	01100000	60
!	00100001	21	A	01000001	41	a	01100001	61
”	00100010	22	B	01000010	42	b	01100010	62
#	00100011	23	C	01000011	43	c	01100011	63
$	00100100	24	D	01000100	44	d	01100100	64
%	00100101	25	E	01000101	45	e	01100101	65
&	00100110	26	F	01000110	46	f	01100110	66
’	00100111	27	G	01000111	47	g	01100111	67
(	00101000	28	H	01001000	48	h	01101000	68
)	00101001	29	I	01001001	49	i	01101001	69
*	00101010	2A	J	01001010	4A	j	01101010	6A
+	00101011	2B	K	01001011	4B	k	01101011	6B
’	00101100	2C	L	01001100	4C	l	01101100	6C
-	00101101	2D	M	01001101	4D	m	01101101	6D
.	00101110	2E	N	01001110	4E	n	01101110	6E
/	00111111	2F	O	01001111	4F	o	01101111	6F
0	00110000	30	P	01010000	50	p	01110000	70
1	00110001	31	Q	01010001	51	q	01110001	71
2	00110010	32	R	01010010	52	r	01110010	72
3	00110011	33	S	01010011	53	s	01110011	73
4	00110100	34	T	01010100	54	t	01110100	74
5	00110101	35	U	01010101	55	u	01110101	75
6	00110110	36	V	01010110	56	v	01110110	76
7	00110111	37	W	01010111	57	w	01110111	77
8	00111000	38	X	01011000	58	x	01111000	78
9	00111001	39	Y	01011001	59	y	01111001	79
:	00111010	3A	Z	01011010	5A	z	01111010	7A
;	00111011	3B	[	01011011	5B	{	01111011	7B
<	00111100	3C	\	01011100	5C	|	01111100	7C
=	00111101	3D	]	01011101	5D	}	01111101	7D