

Circuits

Find Boolean formula for:

- p, q & r are the variables.

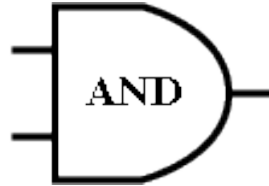
p	q	r	output
1	1	1	1
1	1	0	1
1	0	1	0
1	0	0	1
0	1	1	0
0	1	0	0
0	0	1	0
0	0	0	0

Find Boolean Formula

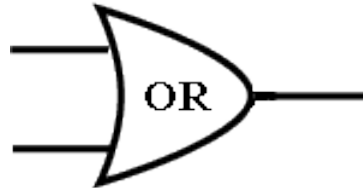
- For each row with output 1 obtain “mini-formula” which is 1 exactly on that row.
 - OR together all of the mini-formulas
-
- 111, 110, and 100 all output 1
 - $p \wedge q \wedge r$, $p \wedge q \wedge \sim r$, $p \wedge \sim q \wedge \sim r$
 - $(p \wedge q \wedge r) \vee (p \wedge q \wedge \sim r) \vee (p \wedge \sim q \wedge \sim r)$

Basic logic gates

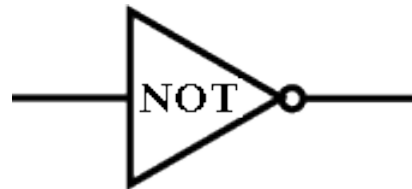
□ AND gate:



□ OR gate:

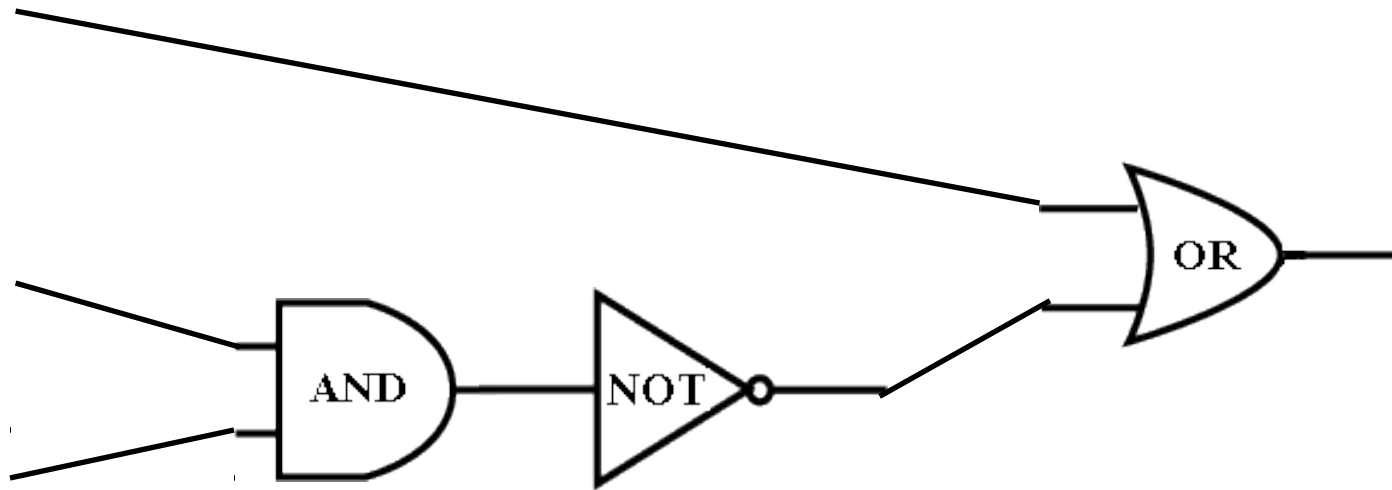


□ NOT gate:



Combining & determining I/O relationship

- From a circuit to a Boolean formula:



- $p \vee \sim(q \wedge r)$

Draw a circuit for:

- p, q & r are inputs.
- Simplify before building the circuit.

p	q	r	output
1	1	1	1
1	1	0	1
1	0	1	0
1	0	0	1
0	1	1	0
0	1	0	0
0	0	1	0
0	0	0	0

Number conversions

□ Different number system bases are used when convenient

- some commonly-used bases are 10 (decimal), 2 (binary), 8 (octal), 16 (hexadecimal)
- the base tells how many different numerals are used
- the base also determines the value of each place

□ Conversions from anything to base 10

- use the definition of the number system

□ Conversions from base 10 to anything

- use repeated integer division

Addition of binary numbers

- Carry if the number would be too large for the number system- if it is greater than 1

$$\begin{array}{r} 1001 \\ + 10 \\ \hline 1011 \end{array}$$

$$\begin{array}{r} 1001 \\ + 11 \\ \hline 1100 \end{array}$$

$$\begin{array}{r} 1011 \\ + 10 \\ \hline 1101 \end{array}$$

$$\begin{array}{r} 1101 \\ + 111 \\ \hline 10100 \end{array}$$

Addition of octal and hexadecimal numbers

- Carry if the number would be too large for the number system (larger than 7 or 15)

$$\begin{array}{r} 723_8 \\ + 12_8 \\ \hline 735_8 \end{array}$$

$$\begin{array}{r} 265_8 \\ + 33_8 \\ \hline 320_8 \end{array}$$

$$\begin{array}{r} ABC_{16} \\ + 12_{16} \\ \hline ACE_{16} \end{array}$$

$$\begin{array}{r} CDE_{16} \\ + ED_{16} \\ \hline DDC_{16} \end{array}$$

Two's complement

□ To represent negative values in binary:

1. Find the binary equivalent of the absolute value.
2. Pad on the left to completely fill the bits in the specified bit width
3. Switch all of the 1's to 0's and 0's to 1's.
4. Add 1 to the result.

• Example: find the 8-bit two's complement representation of -43:

1. $43_{10} = 101011_2$
2. 00101011_2
3. 11010100_2
4. $11010101_2 = -43_{10}$

Using a circuit for adding two bits

- Write as a logic expression
- Translate to circuits

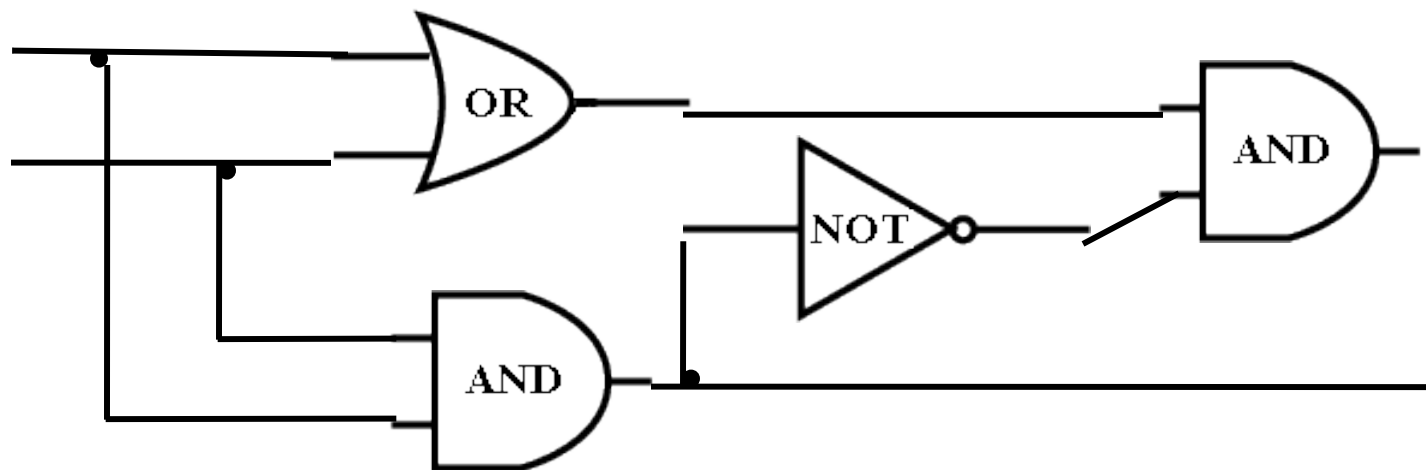
input		output	
p	q	carry	sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Half adder

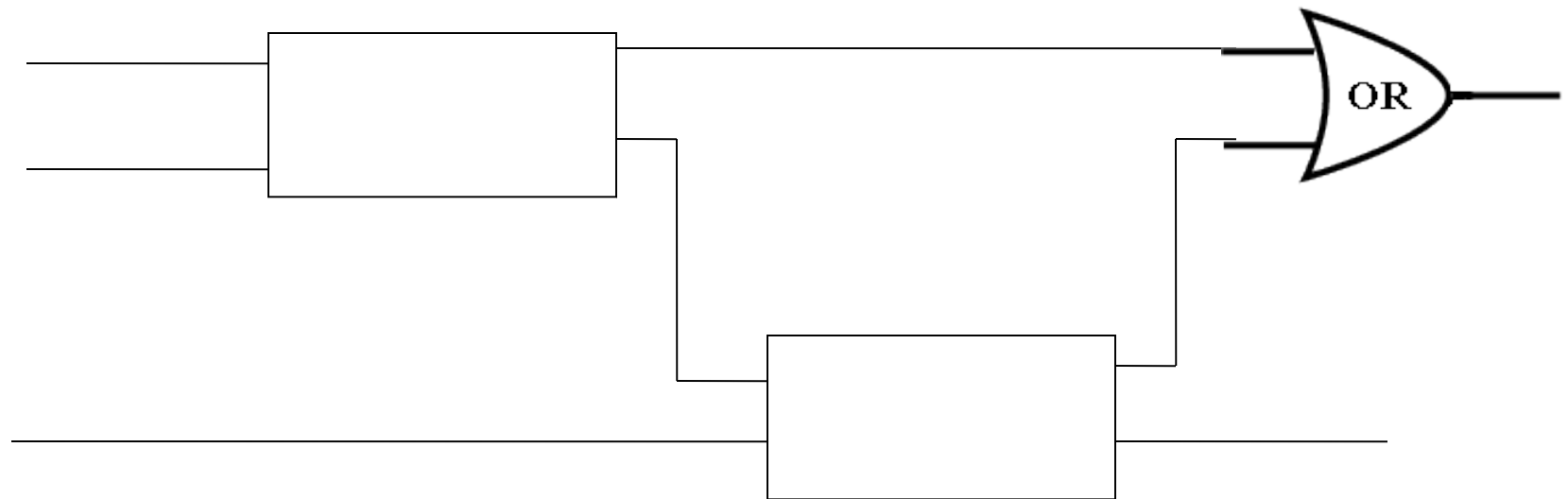
p and q are binary values (1 bit each)

$$\text{sum} = (p \vee q) \wedge \sim(p \wedge q)$$

$$\text{carry} = (p \wedge q)$$



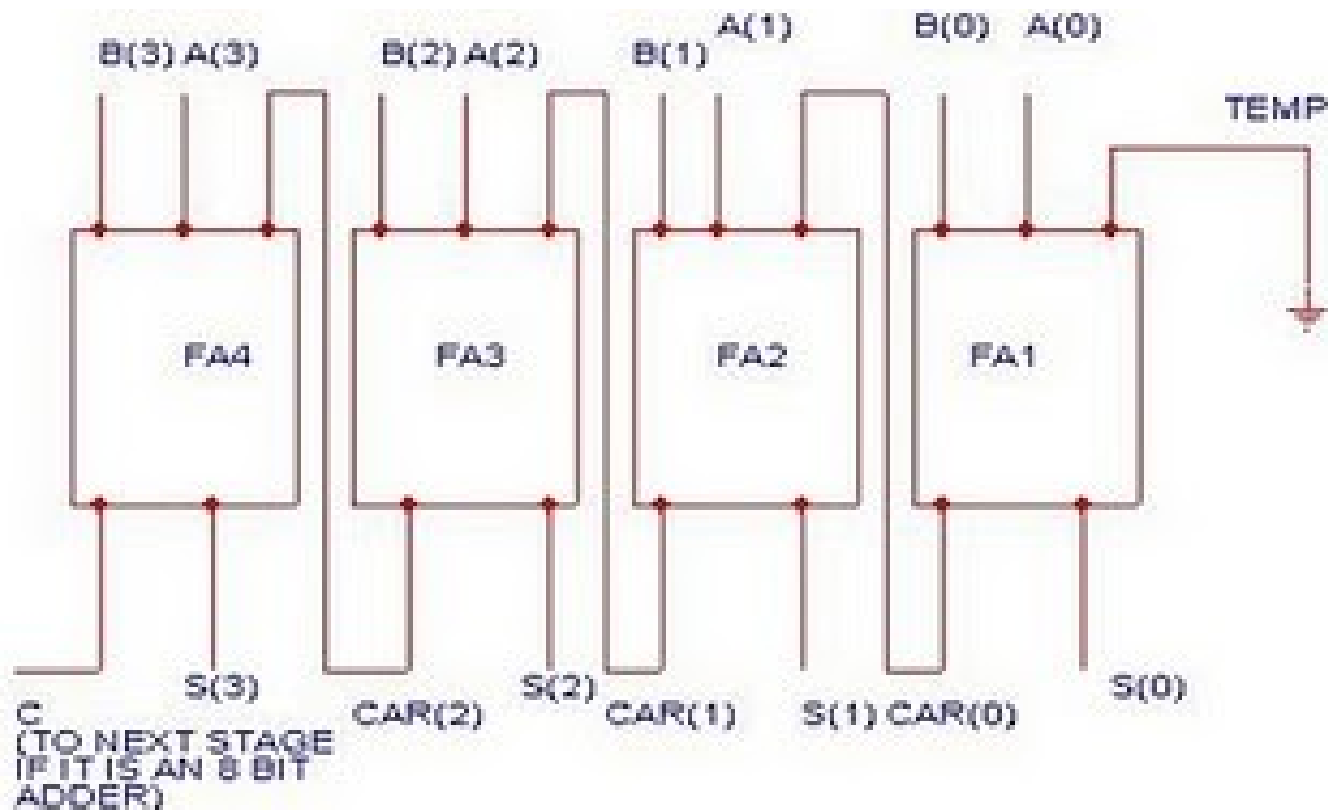
Full adder (for three bits)



- p , q , and r are binary digits
- $p + q + r$ produces a sum value and a carry value

Parallel adders

- Chain these half adders and full adders together for multi-bit addition
- $A_3A_2A_1A_0 + B_3B_2B_1B_0 = S_3S_2S_1S_0$



Draw a circuit for $x \rightarrow x^2$

a	b	c	output
0	0	0	0 0 0 0 0 0
0	0	1	0 0 0 0 0 1
0	1	0	0 0 0 1 0 0
0	1	1	0 0 1 0 0 1
1	0	0	0 1 0 0 0 0
1	0	1	0 1 1 0 0 1
1	1	0	1 0 0 1 0 0
1	1	1	1 1 0 0 0 1

Topic not covered

Simplifying circuits: there are techniques that exist (which are complex).