Fixed width addition

Fixed-width addition: adding one bit at time, using the usual column-by-column and carry arithmetic, and dropping the carry from the leftmost column so the result is the same width as the summands. *Does this give the right value for the sum?*

$$\begin{array}{r} [0\ 1\ 0\ 1]_{s,4} \\ +[1\ 1\ 0\ 1]_{s,4} \\ \hline \end{array}$$

$$\begin{array}{r} [0\ 1\ 0\ 1]_{2c,4} \\ +[1\ 0\ 1\ 1]_{2c,4} \\ \hline \end{array}$$

$$\begin{array}{r} (1\ 1\ 0\ 1\ 0\ 0)_{2,6} \\ +(0\ 0\ 0\ 1\ 0\ 1)_{2,6} \\ \hline \end{array}$$

$$\begin{array}{r} [1\ 1\ 0\ 1\ 0\ 0]_{s,6} \\ +[0\ 0\ 0\ 1\ 0\ 1]_{s,6} \\ \hline \end{array}$$

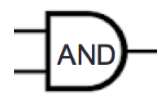
$$\begin{array}{r} [1\ 1\ 0\ 1\ 0\ 0]_{2c,6} \\ +[0\ 0\ 0\ 1\ 0\ 1]_{2c,6} \\ \hline \end{array}$$

Circuits basics

In a **combinatorial circuit** (also known as a **logic circuit**), we have **logic gates** connected by **wires**. The inputs to the circuits are the values set on the input wires: possible values are 0 (low) or 1 (high). The values flow along the wires from left to right. A wire may be split into two or more wires, indicated with a filled-in circle (representing solder). Values stay the same along a wire. When one or more wires flow into a gate, the output value of that gate is computed from the input values based on the gate's definition table. Outputs of gates may become inputs to other gates.

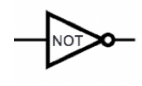
Logic gates definitions

Inputs		Output
x	y	x AND y
1	1	1
1	0	0
0	1	0
0	0	0



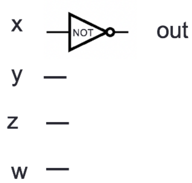
Inputs		Output
x	y	x XOR y
1	1	0
1	0	1
0	1	1
0	0	0

Input	Output
x	NOT x
1	0
0	1



Digital circuits basic examples

Example digital circuit:



Output when $x = 1, y = 0, z = 0, w = 1$ is _____
Output when $x = 1, y = 1, z = 1, w = 1$ is _____
Output when $x = 0, y = 0, z = 0, w = 1$ is _____

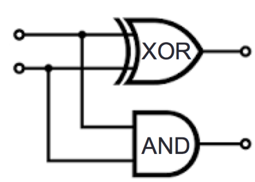
Draw a logic circuit with inputs x and y whose output is always 0. *Can you use exactly 1 gate?*

Half adder circuit

Fixed-width addition: adding one bit at time, using the usual column-by-column and carry arithmetic, and dropping the carry from the leftmost column so the result is the same width as the summands. In many cases, this gives representation of the correct value for the sum when we interpret the summands in fixed-width binary or in 2s complement.

For single column:

Input		Output	
x_0	y_0	c_0	s_0
1	1		
1	0		
0	1		
0	0		



Two bit adder circuit

Draw a logic circuit that implements binary addition of two numbers that are each represented in fixed-width binary:

- Inputs x_0, y_0, x_1, y_1 represent $(x_1 x_0)_{2,2}$ and $(y_1 y_0)_{2,2}$
- Outputs z_0, z_1, z_2 represent $(z_2 z_1 z_0)_{2,3} = (x_1 x_0)_{2,2} + (y_1 y_0)_{2,2}$ (may require up to width 3)

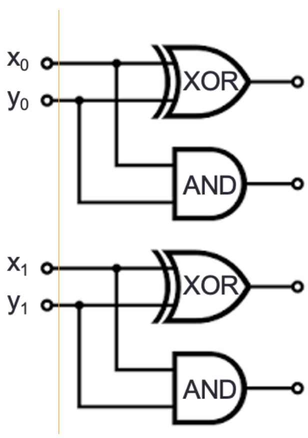
First approach: half-adder for each column, then combine carry from right column with sum of left column

Write expressions for the circuit output values in terms of input values:

$z_0 =$ ____

$z_1 =$ ____

$z_2 =$ ____



There are other approaches, for example: for middle column, first add carry from right column to x_1, then add result to y_1

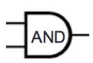
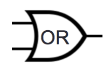
Logical operators

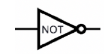
Logical operators aka propositional connectives

Conjunction	AND	$\wedge$	`\land`	2 inputs	Evaluates to T exactly when **both** inputs are T
Exclusive or	XOR	$\oplus$	`\oplus`	2 inputs	Evaluates to T exactly when **exactly one** of inputs is T
Disjunction	OR	$\vee$	`\lor`	2 inputs	Evaluates to T exactly when **at least one** of inputs is T
Negation	NOT	$\neg$	`\lnot`	1 input	Evaluates to T exactly when its input is F

Logical operators truth tables

Truth tables: Input-output tables where we use T for 1 and F for 0.

Input		Output		
		Conjunction	**Exclusive or**	**Disjunction**
p	q	$p \wedge q$	$p \oplus q$	$p \vee q$
T	T	T	F	T
T	F	F	T	T
F	T	F	T	T
F	F	F	F	F
				

Input	Output
	Negation
p	$\neg p$
T	F
F	T
	

Logical operators example truth table

Input			Output		
p	q	r		$(p \wedge q) \oplus (\ (p \oplus q) \wedge r\ )$	$(p \wedge q) \vee (\ (p \oplus q) \wedge r\ )$
T	T	T			
T	T	F			
T	F	T			
T	F	F			
F	T	T			
F	T	F			
F	F	T			
F	F	F			

Truth table to compound proposition

Given a truth table, how do we find an expression using the input variables and logical operators that has the output values specified in this table?

Application: design a circuit given a desired input-output relationship.

Input		Output	
p	q	$mystery_1$	$mystery_2$
T	T	T	F
T	F	T	F
F	T	F	F
F	F	T	T

Expressions that have output $mystery_1$ are

Expressions that have output $mystery_2$ are

Idea: To develop an algorithm for translating truth tables to expressions, define a convenient **normal form** for expressions.

Dnf cnf definition

Definition An expression built of variables and logical operators is in **disjunctive normal form** (DNF) means that it is an OR of ANDs of variables and their negations.

Definition An expression built of variables and logical operators is in **conjunctive normal form** (CNF) means that it is an AND of ORs of variables and their negations.