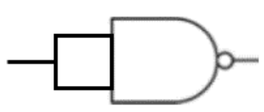
Week 2: Logic circuits

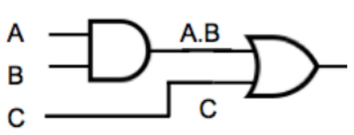
Advantages of universal logic gates

The advantage of using the NOT versions of the gates is that we don't need a special gate to implement the NOT logic. The following circuit implements a NOT gate out of a NAND gate:



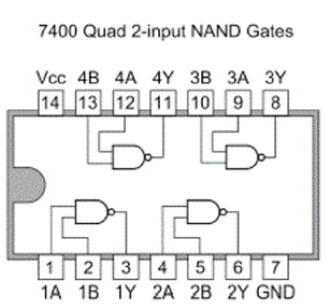
Combinational logic

Combinational logic consists of a logic circuit built with logic gates, to implement an output determined by the logic gates and the current set of inputs. There is no memory in the system, that is, the output depends exclusively of the inputs. An example combining and AND gate with an OR gate is as follows:

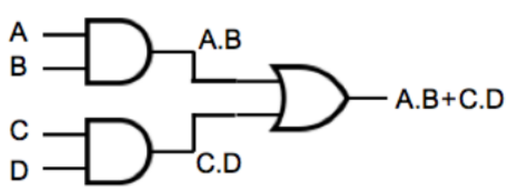


A.B+C

In practice, these gates are not provided individually but in integrated circuits, such as:



The most common situation is to build a circuit from a given Boolean expression. Recall that this is the reason why we try to simplify expressions, so circuits are easier to build. For example, the SOP expression
A . B + C . D is built very simply with two AND gates joined together with an OR gate:



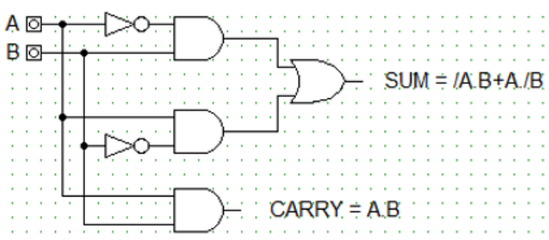
Binary adders

Half adder

To be able to add binary numbers, we need to add individual bits properly, that is, we should be able to calculate their sum with the corresponding carry. The Truth Table for a 1-bit adder - a half-adder - is as follows:

Inputs		Outputs	
A	**B**	**Sum**	**Carry**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

This is the logic for a half-adder, as implemented with standard gates:



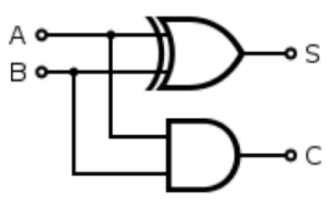
The formulas corresponding to a half-adder are as follows:

Sum = /A.B + A./B
Carry = A.B

Note that the circuit reflects exactly the truth table and the formulas. Note also that this output is the output from an XOR gate so that a half adder can be represented by:

A	B	Carry	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Sum is A⊕B
Carry is A.B

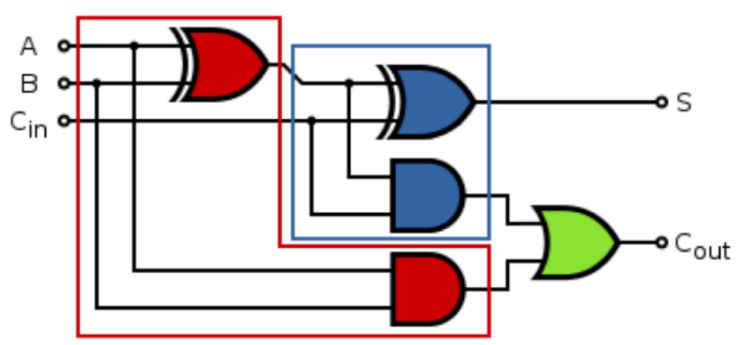


The downfall of half adders is that while they can generate a carry out output, they cannot deal with a carry in signal. This means that they can only ever be stand-alone units, and cannot be concatenated to add multiple bit numbers.

Full adder

In a more complex operation, to add two bits properly we have to make sure not only that we add the two bits, but also that we add the carry properly. The design could be carried out again via a truth table, but a better method is to use two half-adders, since this approach is based on existing components. A full-adder is able to add in the carry from the previous column in a two-digit binary number, like this:

- We could combine two half adders to make a full adder (which accepts 3 inputs: A, B and a carry-in)

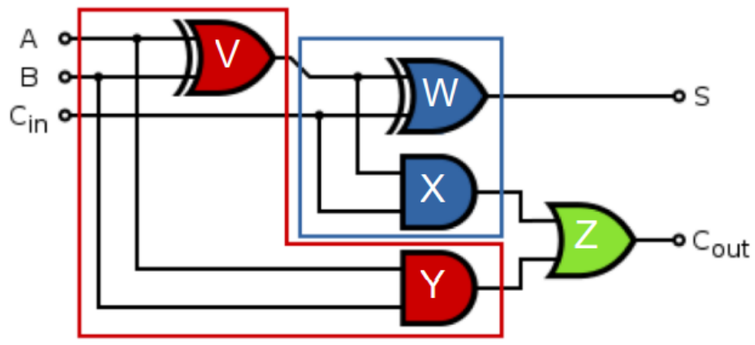


A full adder thus solves this problem by adding three numbers together - the two addends as in the half adder, and a carry in input.

A full adder can be constructed from two half adders by connecting A and B to the input of one half adder, connecting the sum from that to an input to the second adder, connecting the carry in, Cin, to the other input and ORing the two half adder carry outputs to give the final carry output, Cout.

This can be summarised in the following diagram:

- Full adders can be cascaded together to make a parallel adder that can add multi-bit binary numbers
- We can build a truth table to examine the behaviour of the full-adder



A	B	C	V A ⊕ B	sum W V ⊕ C	X V.C	Y A.B	carry Z A + B
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

V = A xor B = A ⊕ B S = (A⊕B) ⊕ C
W = V xor C = V ⊕ C C_{out} = V.C + A.B
X = V and C = V.C = (A⊕B) .C + A.B
Y = A and B = A.B
Z = X or Y = X + Y

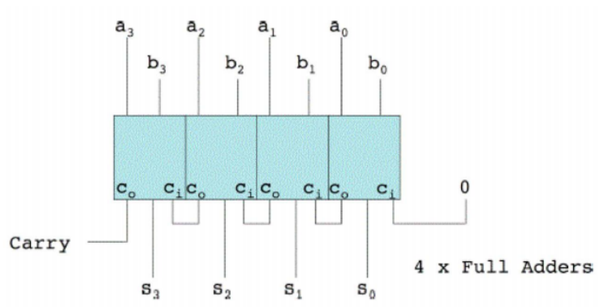
Parallel adder example

A multi-digit adder may be implemented by cascading full-adders, as shown in Figure 8.14. The .gure shows 4 full- adders, in which the carry out of each full-adder is fed into the next one. The scheme is initialised with 0 as the .rst carry in, and the last carry out is the carry out of the whole adder.

• 4 cascaded full-adders to add two 4-bit binary numbers

• the carry-out of each full-adder feeds into carry-in of the next full-adder

• the first carry-in is set to 0



Videos

PC Architecture (http://www.karbosguide.com/books/pcarchitecture/start.htm) : a book by Michael Karbo (http://www.karbosguide.com/) .