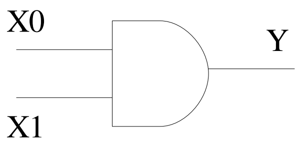
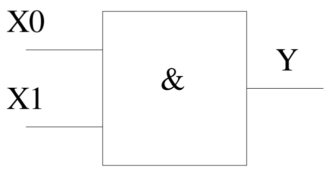
Logical and arithmetic operations

The AND gate.

An AND gate is a digital electronic circuit the output of which is logic '1' only if both (or all) inputs are also logic '1'.
The truth-table for a two input AND gate is shown below along with the two symbols commonly used to represent it. X0 and X1 are inputs, Y is the output.

X0	X1	Y
0	0	0
0	1	0
1	0	0
1	1	1

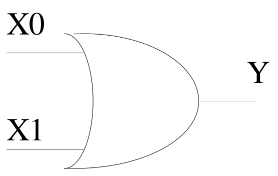


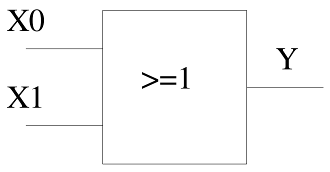


The OR gate

The output of an OR gate is logic '1' if any of its inputs is a logic '1'.
The truth-table for a two input OR gate is shown below along with the two symbols commonly used to represent it.

X0	X1	Y
0	0	0
0	1	1
1	0	1
1	1	1

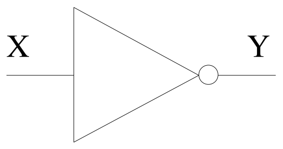


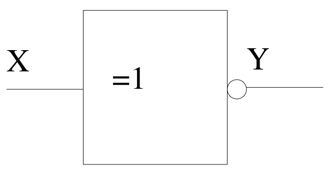


The NOT gate (inverter)

Unlike the previous two gates, the NOT gate is a single input gate. Its output is the logical inverse of the input. The truth-table for an NOT gate is shown below along with the two symbols commonly used to represent it.

X	Y
0	1
0	0

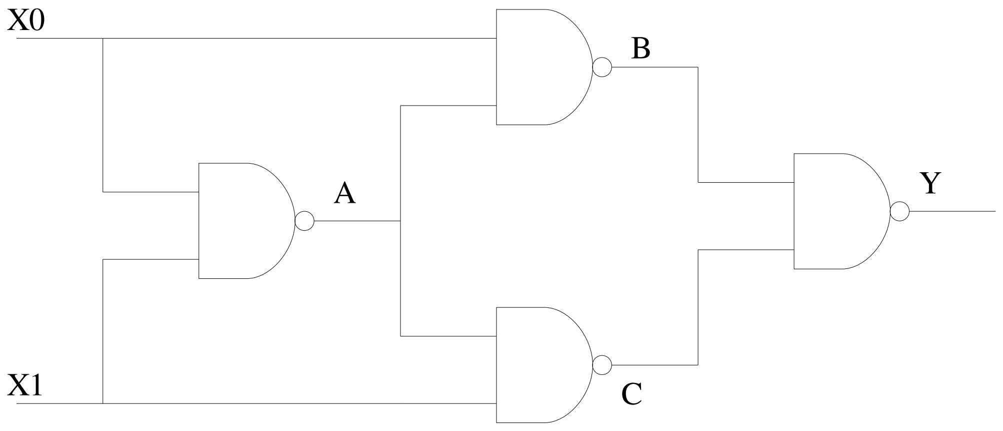




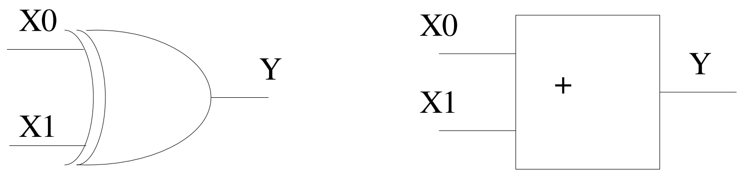
Combinational logic

It is possible to combine a number of the above gates together to form other gates. For example, if an AND gate is followed by a NOT gate, the result is a NAND gate the output of which is the inverse of the AND gate. Similarly, following an OR gate with a NOT gate results in a NOR gate. Another commonly used gate, the XOR (exclusive OR) gate is made from a combination of gate. An XOR gate can be made from a series of NAND gates as shown below. The truth table along with internal states ABC is also shown. As can be seen the output is a '1' if one and only one of the inputs is a '1'.

X0	X1	*A*	*B*	*C*	Y
0	0	*1*	*1*	*1*	0
0	1	*1*	*1*	*0*	1
1	0	*1*	*0*	*1*	1
1	1	*0*	*1*	*1*	0

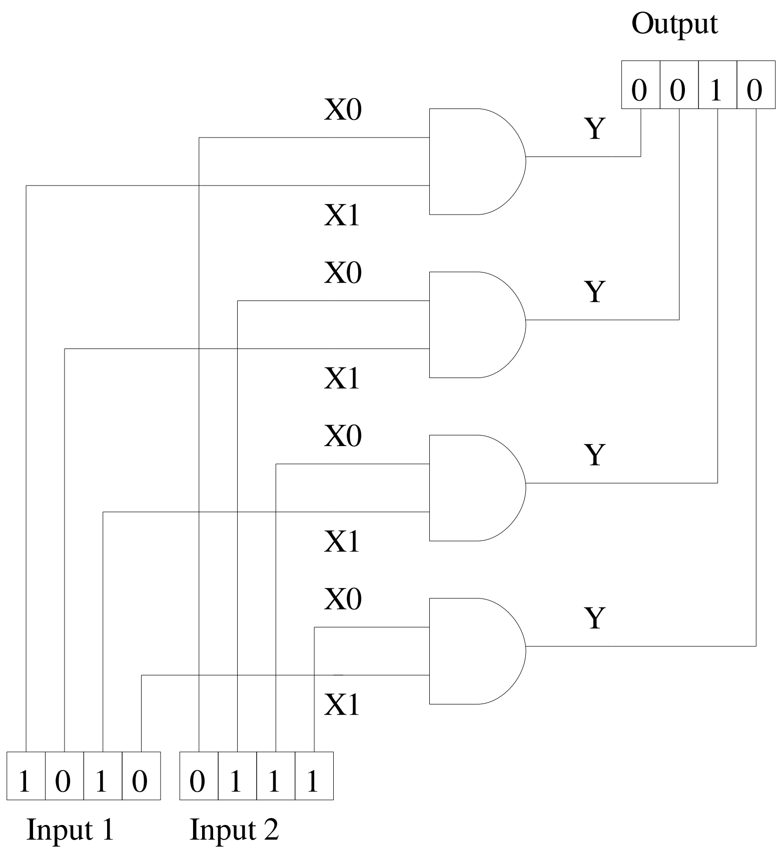


The more commonly used symbols for the XOR gate are shown below. Note the IEC symbol (the box-like one) uses a '+' to denote the operation of the XOR gate. This is because the XOR gate can be considered to be a 1 bit adder.



Arrays of gates

The gates shown above work with single bit inputs and produce a single bit output. Arrays of gates can be built such that multiple bits are operated on in one go. Thus it is possible to perform a 4 bit AND operation as follows:



Note the output contains a single logic '1' in the bit position where both corresponding inputs are logic '1'. This can of course be extended to 8, 16 and higher numbers of bits. Arrays of OR, NOT, XOR etc. gates are also possible.

The C language and logical operations

The C language supports the following logical operations:

Operation	Symbol
AND	&
OR	| (pipe char)
XOR	^
NOT	~

You should try the following short program with your favourite compiler to see these operators in action:

```
int main(void) {

     printf("5 & 4 = %d\n",5&4);
     printf("5 | 4 = %d\n",5|4);
     printf("5 ^ 4 = %d\n",5^4);
     printf("~5 = (hex) %x\n",~5);
}
```