0. Course Logistics & An introduction to Computational Systems

Notes

Digital Logic Gates

Aims

- to introduce the lowest level processing circuits in digital computers
- to show that the behaviour of a logic gate is specified by its truth table
- to show how logic gates are combined on a chip
- to show how logic gates can be combined to produce useful circuits such as multiplexors and comparators

1.0 Introduction

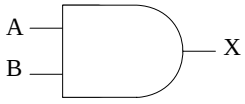
Digital computers store and process ***binary*** data. Binary data is represented by ***1***'s and ***0***'s. This simplifies the circuits that are used at the lowest level as they only have to have two stable states, one that represents a ***1*** and one that represents a ***0***.

The basis of all processing is the ***logic gate***. The logic gate is a circuit that implements a logical function. Each gate is represented by its own symbol and the symbols can be connected together to construct ***circuit diagrams***. The behaviour of a logic gate can be explained in words, with each gate behaving exactly the same given the same set of inputs. ***Truth tables*** can be used to reason about the behaviour of gates and circuits.

There are ***six basic logic gates***: **AND**, **OR**, **NOT**, **NAND**, **NOR**, **XOR**.

1.1 Gates

AND Gate

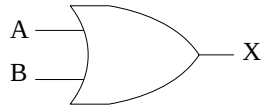


The **AND** gate outputs a 1 when all the inputs are 1
(X = 1 when both A AND B are 1).

A	B	X
0	0	
0	1	
1	0	
1	1	

Truth Table

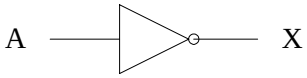
OR Gate



The **OR** gate outputs a 1 when *either* or *both* of the inputs is a 1
(X = 1 if A OR B = 1, OR both).

A	B	X
0	0	
0	1	
1	0	
1	1	

NOT Gate

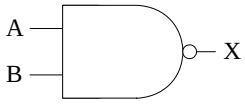


The NOT gate outputs a 1 when the input is 0.
When the input is a 1 the NOT gate outputs a 0.

A	X
0	
1	

The NOT gate is also referred to as an ***inverter***, as it inverts its inputs.

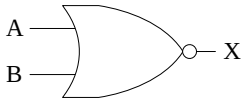
NAND Gate



The NAND gate behaves like an AND gate whose output is then passed through a NOT gate.
Thus NOT AND is shortened to give NAND.
Therefore the truth table is the opposite of an AND gate.

A	B	X
0	0	
0	1	
1	0	
1	1	

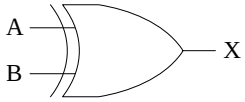
NOR Gate



The NOR gate behaves like an OR gate whose output is then passed through a NOT gate.
Thus NOT OR is shortened to give NOR.
Therefore the truth table is the opposite of an OR gate.

A	B	X
0	0	
0	1	
1	0	
1	1	

XOR Gate



The XOR gate (exclusive OR) outputs a 1 when *only one* of its inputs is a 1
(X = 1 if A OR B = 1, but not both).

A	B	X
0	0	
0	1	
1	0	
1	1	

The behaviour of a logic gate is independent of how it is actually constructed.

This allows circuit designers to implement gates how they wish.

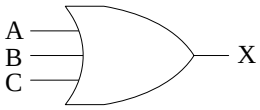
1.2 Multiple Input Gates

Each logic gate can have only one output, but any number of inputs (with the exception of the NOT gate which can only have one input).

The number of rows in the truth table depends on the number of inputs to the gate.

number of rows = 2^n where n is the number of inputs

Thus, a three input OR gate would have a truth table with eight rows.



A	B	C	X
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

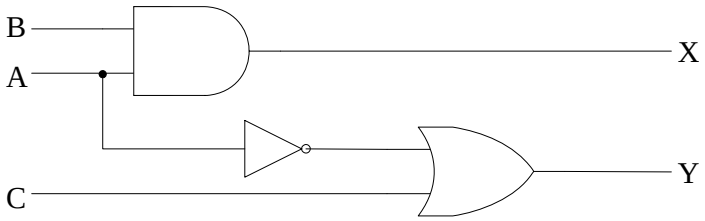
SAQ 1.1

Construct truth tables for 3 input versions of the AND, NAND and NOR gates.

answer

1.3 Simple Circuits

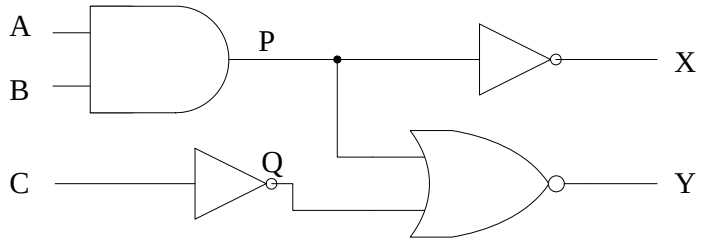
Truth tables allow us to ascertain the behaviour of circuits with ***multiple inputs*** and ***multiple outputs***.



A	B	C	X	Y
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

1.4 Intermediate Results

With more complex circuits ***intermediate results*** can be used to help construct the ***final truth table OUTPUT(s).***



A	B	C	P	Q	X	Y
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

SAQ 1.2

The **XOR** gate is similar to the **OR** gate, except that it only outputs a 1 if *EXACTLY ONE* of its inputs is a 1.

Draw a circuit diagram that creates an **XOR** gate using just ANDs, ORs and NOTs.

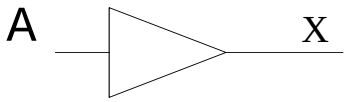
(HINT: the standard solution uses five gates).

answer

1.5 Buffer & Tri-State Gate outputs

1.5.1 Buffer

- The gate below looks like an inverter, but it is not,
  ……. because it does not have a small circle at its output.



- Such a gate is called a ***buffer*** because it copies the signal at its input to its output (therefore the ***buffer*** *gate does not change the state of the data passing through it*, unlike other gates we have examined).

1.5.2 Tri-State output

- A gate with a tri-state output has the special property that the output of the gate can be 0, 1, or a meaningless (disconnected) (third) state.

- A tri-state gate can be any of the gates previously encountered – it is not the gates logical function that is different, it is the behaviour of its output (strictly speaking we shouldn't speak of tri-state gates, we should speak of conventional gates with tri-state outputs).

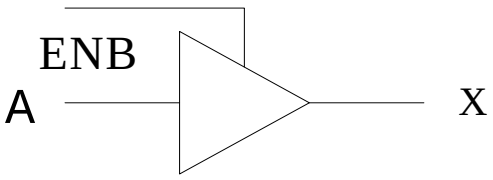
OPERATION OF A TRI-STATE GATE OUTPUT

- All tri-state gates have a special **ENABLE** input.

  ⇒ when ENABLE = 1, the gate behaves normally and its output is either a 1 or a 0, depending on its input.

  ⇒ when ENABLE = 0, the output is physically disconnected from the gate's internal circuitry. In this case we can say the output is meaningless.

- The gate below is classed as a non-inverting tri-state buffer.



ENABLE	A	Output	Description
0	0	X	Output meaningless & disconnected
0	1	X	Output meaningless & disconnected
1	0	0	Output same as Input
1	1	1	Output same as Input

References

- **Logisim** http://www.electronics-micros.com/software-hardware/logisim/