1. 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.

1.1 Introduction

- Digital computers store and process *binary* data.
  $\Rightarrow$ binary data is represented by 1's and 0's.

- The basis of all processing is the logic gate.
  $\Rightarrow$ 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.

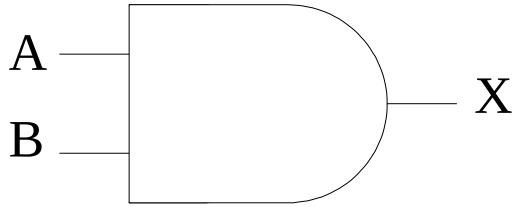
- The behaviour of a logic gate can be explained in words, with each gate behaving exactly the same given the same set of inputs.

- 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.

1.2 Gates

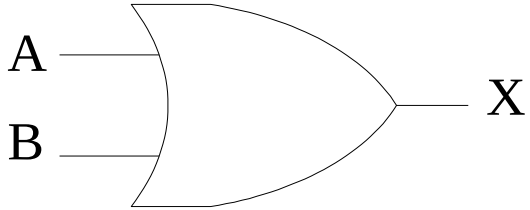
AND Gate



- 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	

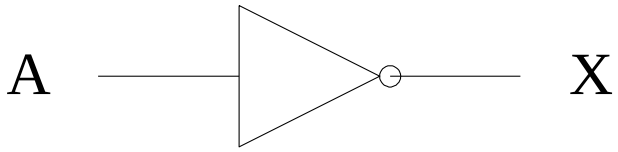
OR Gate



- 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

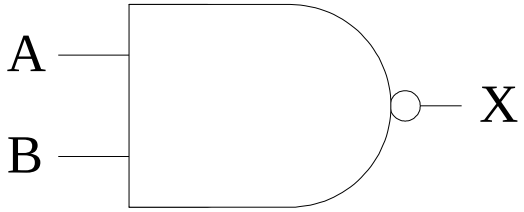


- The NOT gate outputs a 1 when the input is 0. When the input is a 1 the NOT gate outputs a 0.

A	X
0	
1	

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

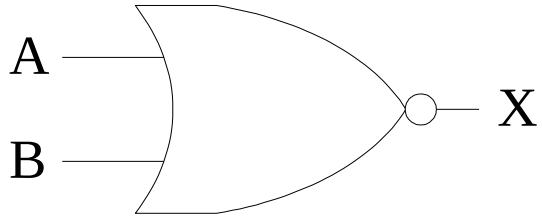
NAND Gate



- 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	

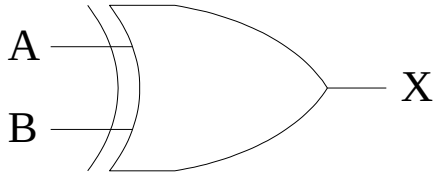
NOR Gate



- 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	

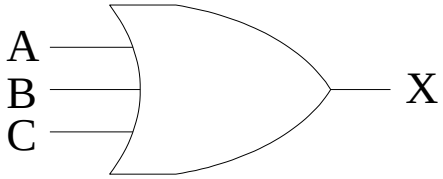
1.3 Multiple Input Gates

- 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.

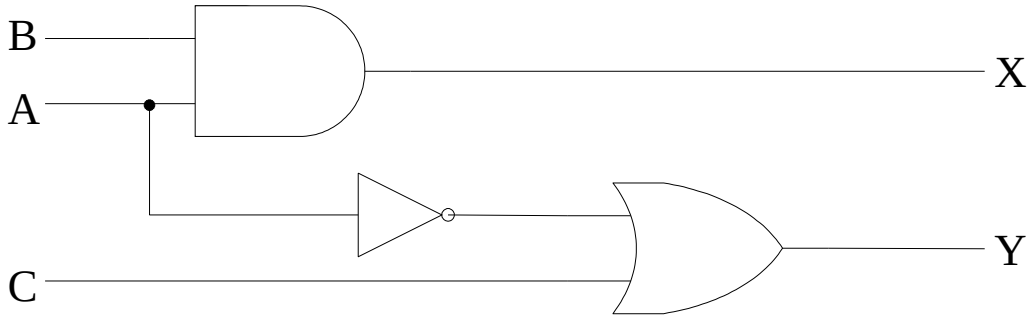
- Thus, a three input OR gate would have a truth table with eight rows.



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	

1.4 Simple Circuits

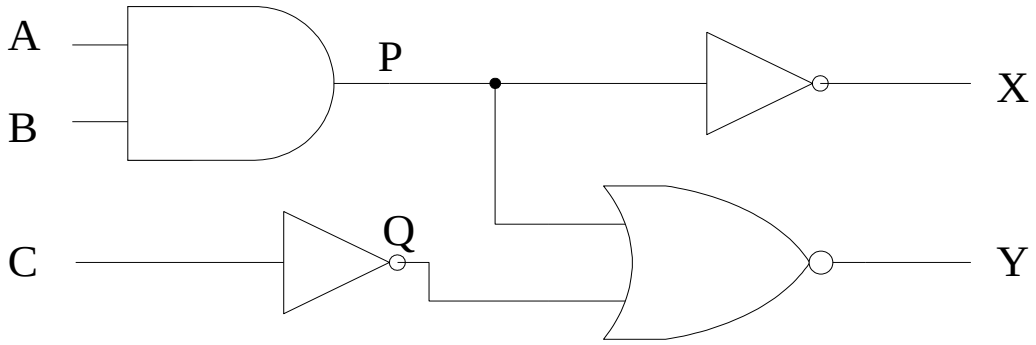
- Truth tables allow us to ascertain the behaviour of circuits, with multiple inputs and multiple outputs.



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		

1.5 Intermediate Results

- With more complex circuits intermediate results can be used to help construct the final truth table.



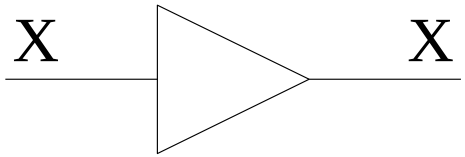
A	B	C	P	Q	X	Y
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

1.6 Tri-State Gates

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

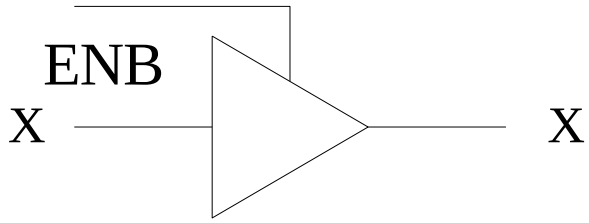
- The tri-state gate below looks like an inverter, but it is not because it does not have a small circle at its output.



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

- 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	
0	0	X	Output meaningless
0	1	X	Output meaningless
1	0	0	Output same as input
1	1	1	Output same as input

2. Combinational Circuits

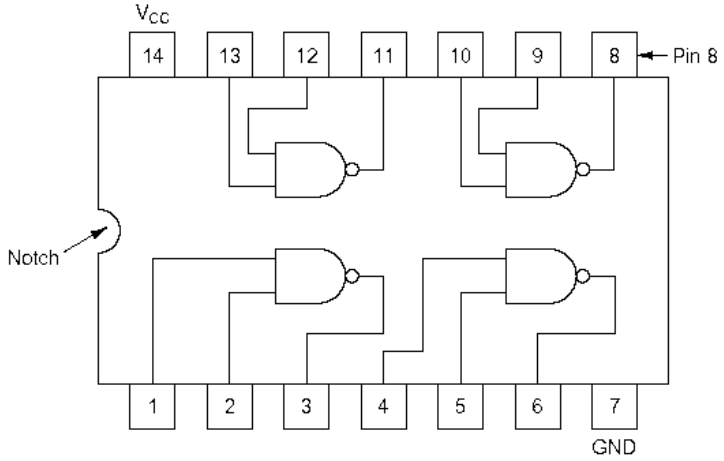
Aims

- 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.

2.1 Introduction

- Gates are not manufactured individually but in units called *integrated circuits* (ICs or chips). Chips can be divided into rough classes based on the number of gates they contain:

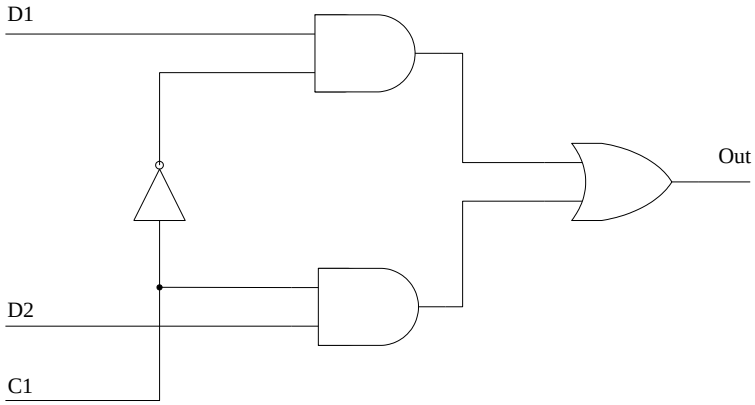
SSI (Small Scale Integrated) circuit	1 to 10 gates
MSI (Medium Scale Integrated) circuit	10 to 100 gates
LSI (Large Scale Integrated) circuit	100 to 100,000 gates
VLSI (Very Large Scale Integrated) circuit	> 100,000 gates



- In the 1970s, computers were constructed out of large numbers of these chips, but technological advances have allowed an entire CPU to be etched onto a single chip. The current state of the art allows nearly 10 million gates on a chip.

2.2 Multiplexors

- The circuit below is a multiplexor with two inputs and one output (a 2-to-1 multiplexor).



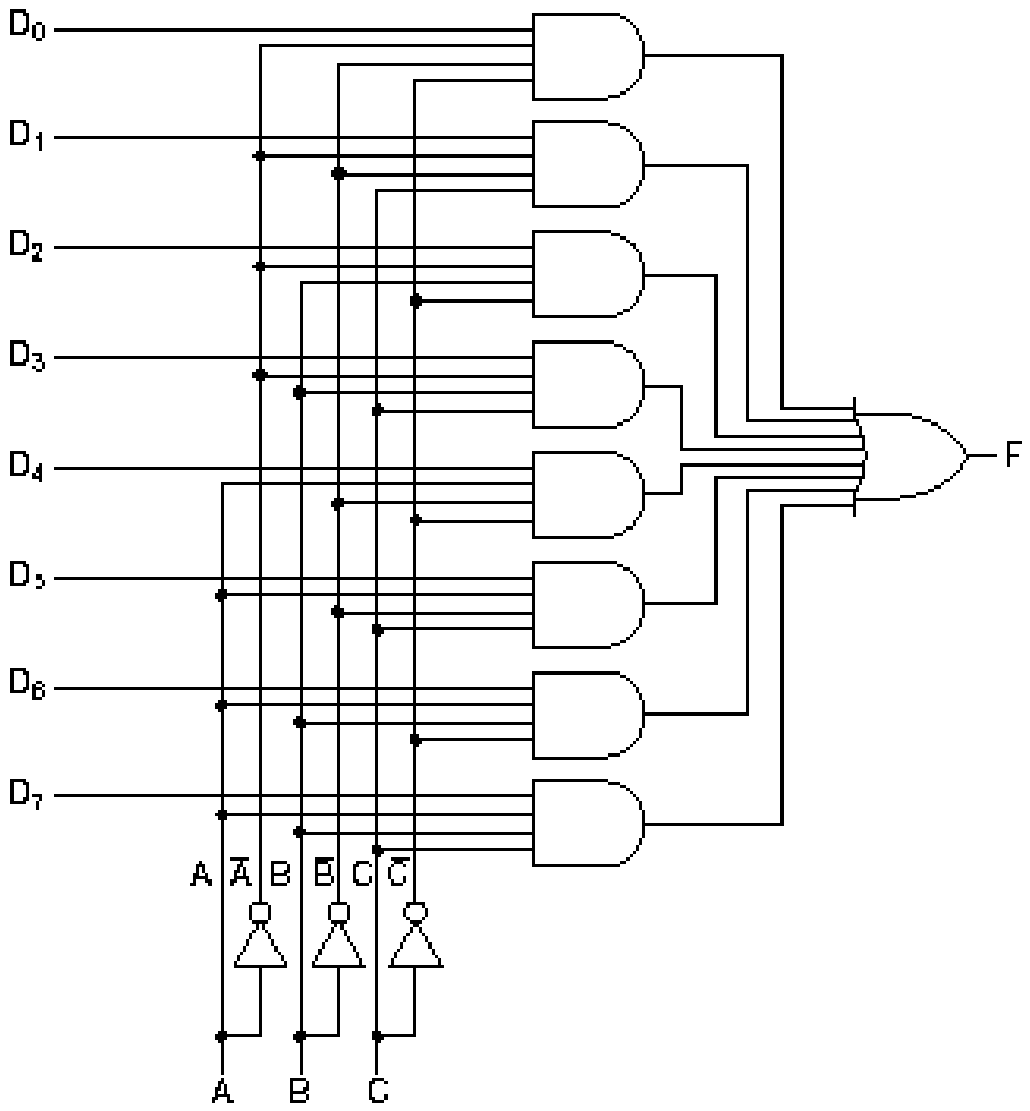
D1	D2	C1	$\overline{C1}$	D2.C1	D1.$\overline{C1}$	Out
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

- If you carefully examine the truth table you can see that the output is either D1 or D2, depending on the value of C1. Therefore, the truth table reduces to:

C1	Out
0	D1
1	D2

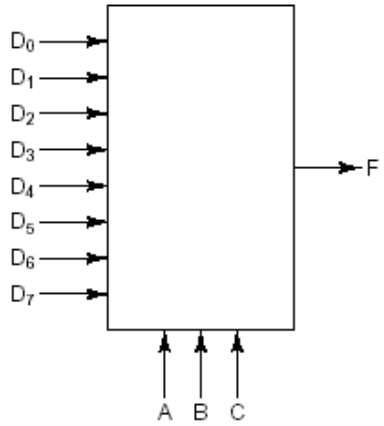
- Therefore, the function of a multiplexor is to route one of many data inputs to a single output. The control input (C1) determines which of the data inputs is passed through to the output.

- A 2^n-to-1 multiplexor has 2^n data inputs and *n* control inputs. The bit combination on the control inputs determines which data input is selected for output.



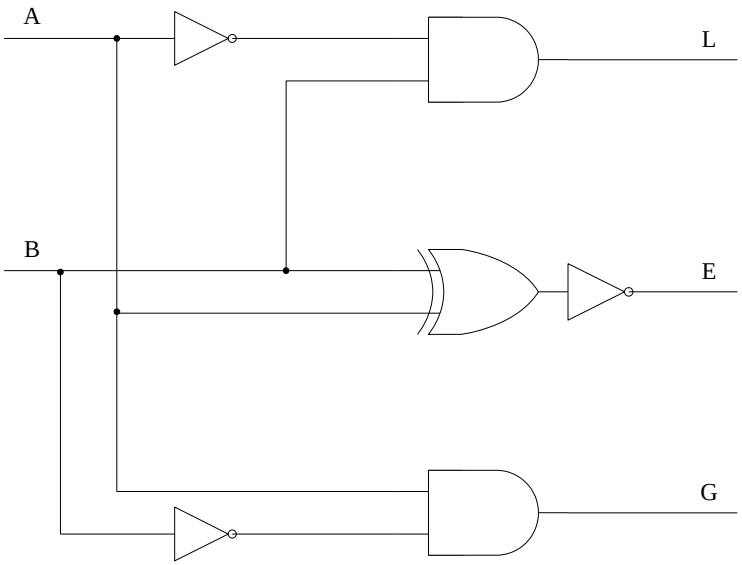
A	B	C	Output
0	0	0	D0
0	0	1	D1
0	1	0	D2
0	1	1	D3
1	0	0	D4
1	0	1	D5
1	1	0	D6
1	1	1	D7

- A simpler way to diagrammatically represent an eight-input multiplexor is shown below.

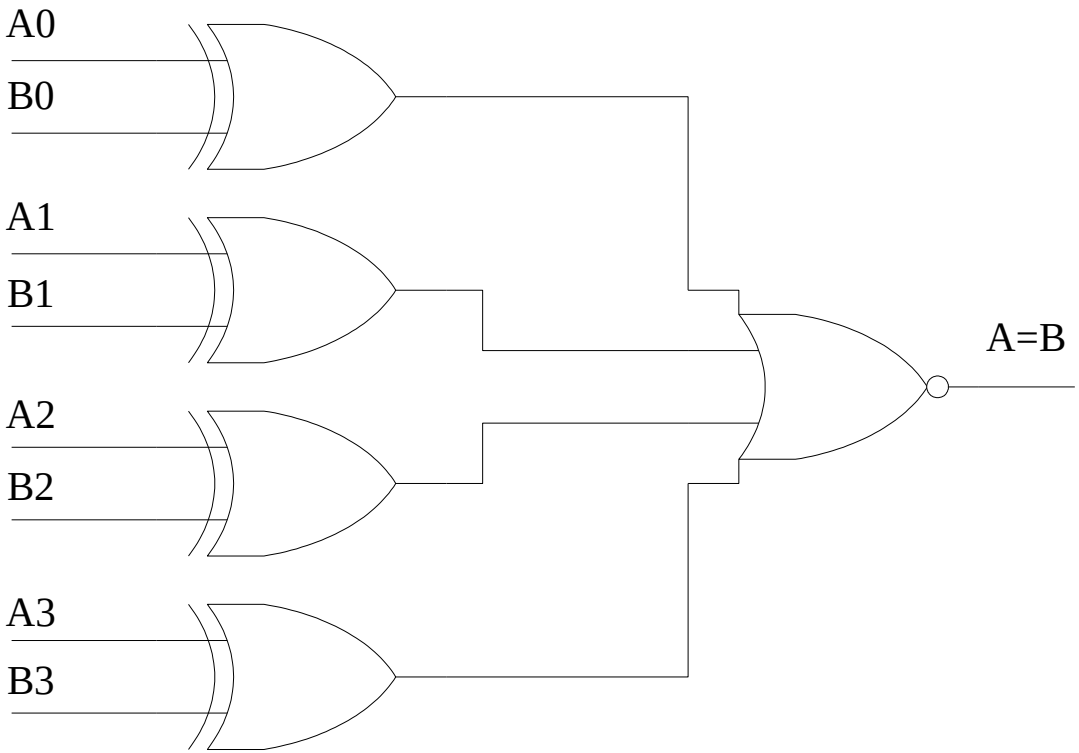


2.3 Comparators

- A comparator compares two inputs and outputs a 1 on the E output if the inputs are equal.



A	B	$\overline{A}$	$\overline{B}$	L	E	G
0	0					
0	1					
1	0					
1	1					



A0	B0	A1	B1	A2	B2	A3	B3	A0⊕B0	A1⊕B1	A2⊕B2	A3⊕B3	NOR
0	0	0	0	0	0	0	0					
0	1	0	1	0	1	0	1					
1	0	1	0	1	0	1	0					
1	1	1	1	1	1	1	1					