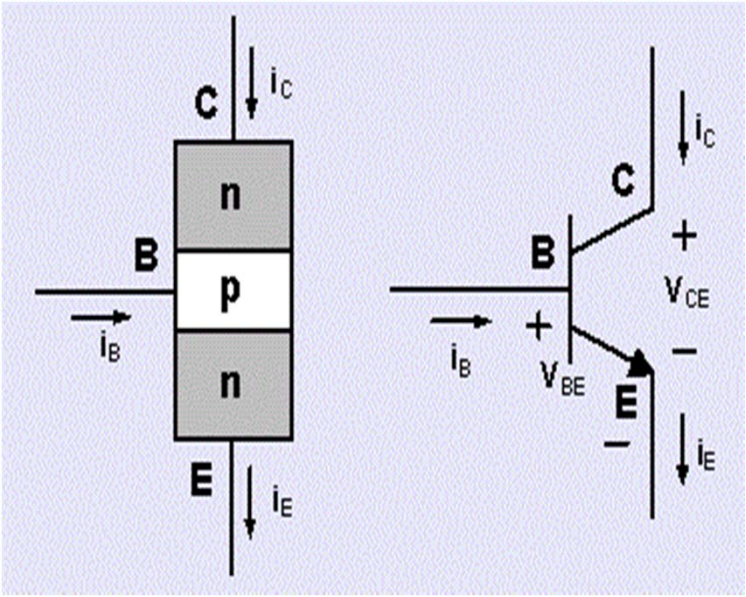
Logic Gate Circuits

- Transistor-Transistor Logic (TTL)
- Complementary Metal-Oxide Semiconductor (CMOS)
- Switch level modelling

Bipolar Junction Transistor (BJT)

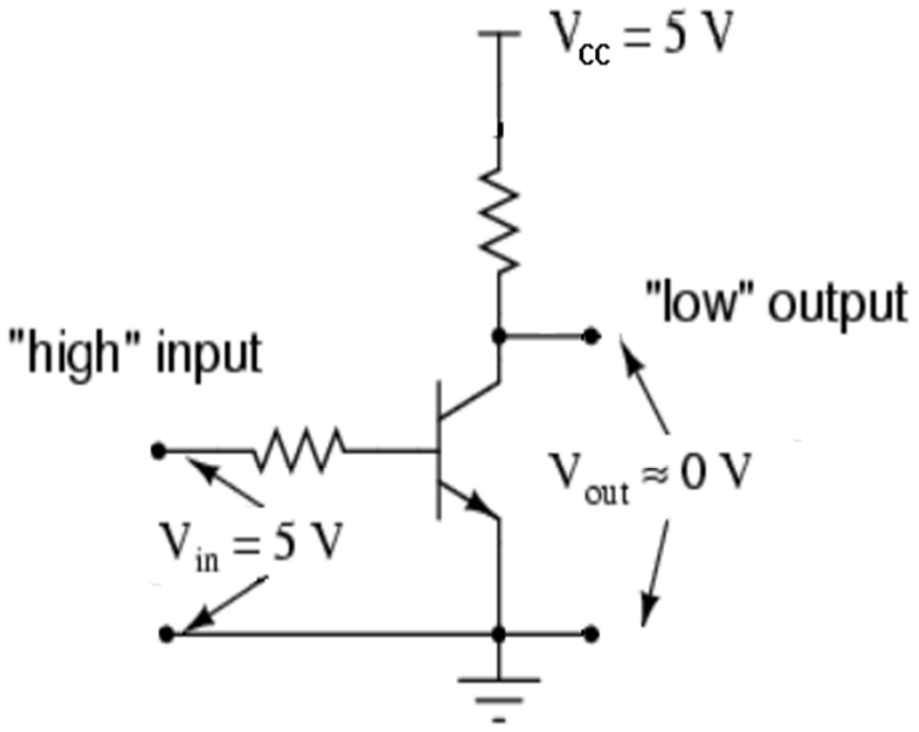
- TTL is implemented with BJTs
- active switching element is an npn transistor
- three terminals: base, emitter, collector
- two junctions: base-emitter, base collector
- will be used in cut-off and saturation modes



A simple transistor inverter:

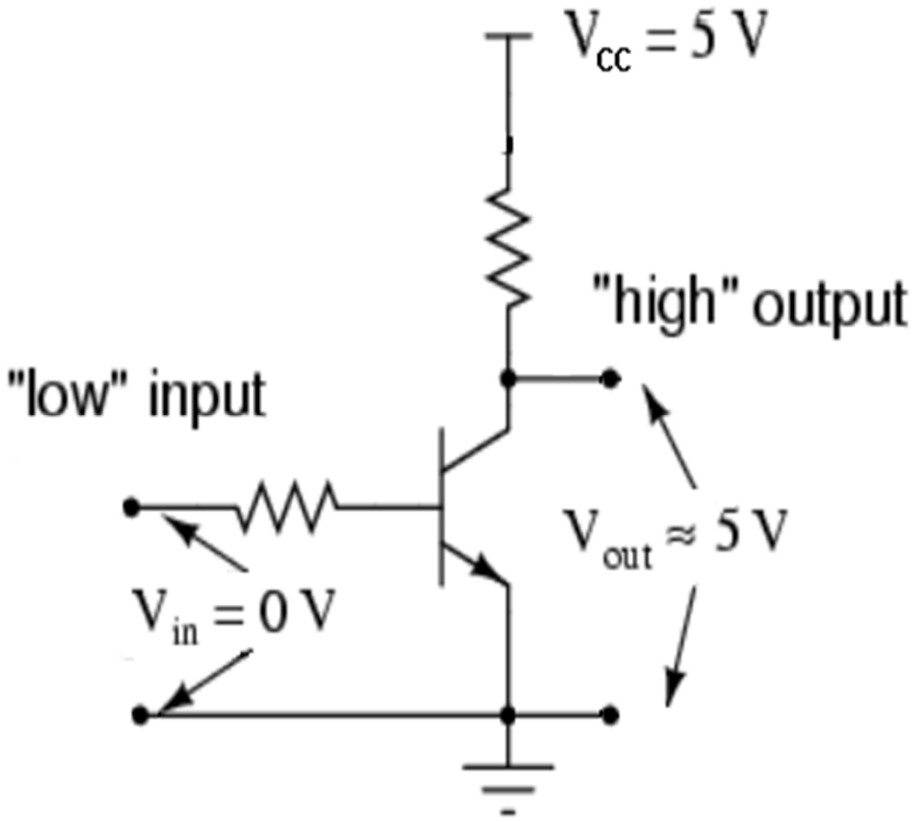
The transistor is in ‘saturation’.

The output will be almost 0V depending upon the collector-emitter voltage which is approximately 0.2V.



0 V = "low" logic level (0)

5 V = "high" logic level (1)

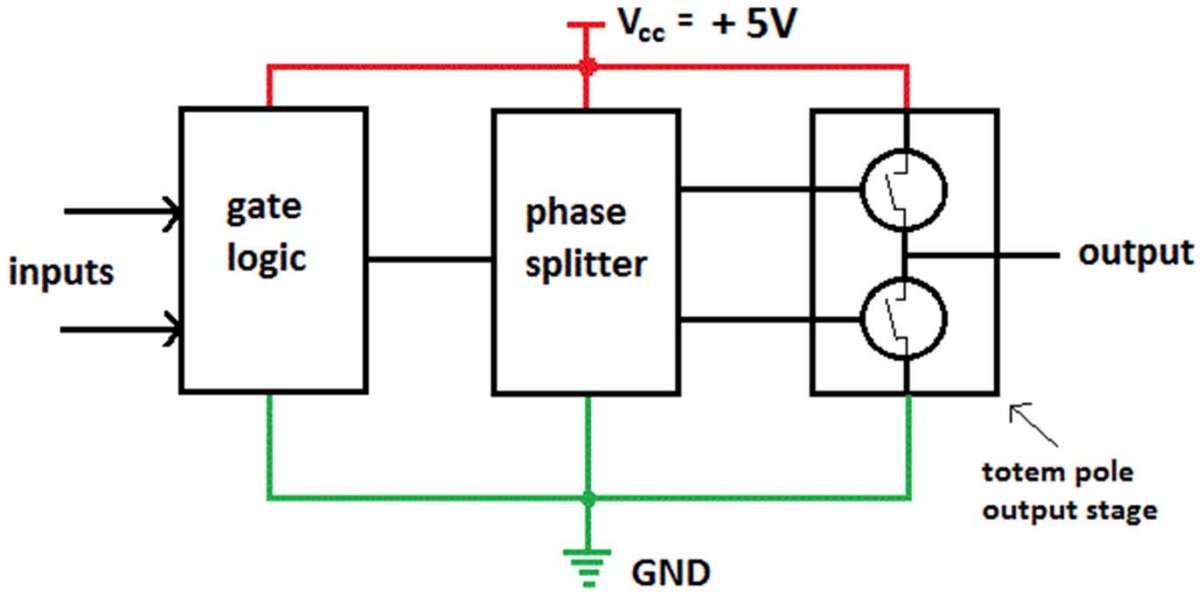


NOT gate

Input	Output
0	1
1	0

The transistor is cut-off and the output will be approximately 5V. Practical circuits build on this principle but also provide voltage gain and protection circuitry.

Transistor-Transistor Logic (TTL)



The output transistors are used to pull-up the output to V_{cc} (output HIGH) or to pull-down the output to GND (output LOW). The control signals to do this will be of the opposite phase to each other.

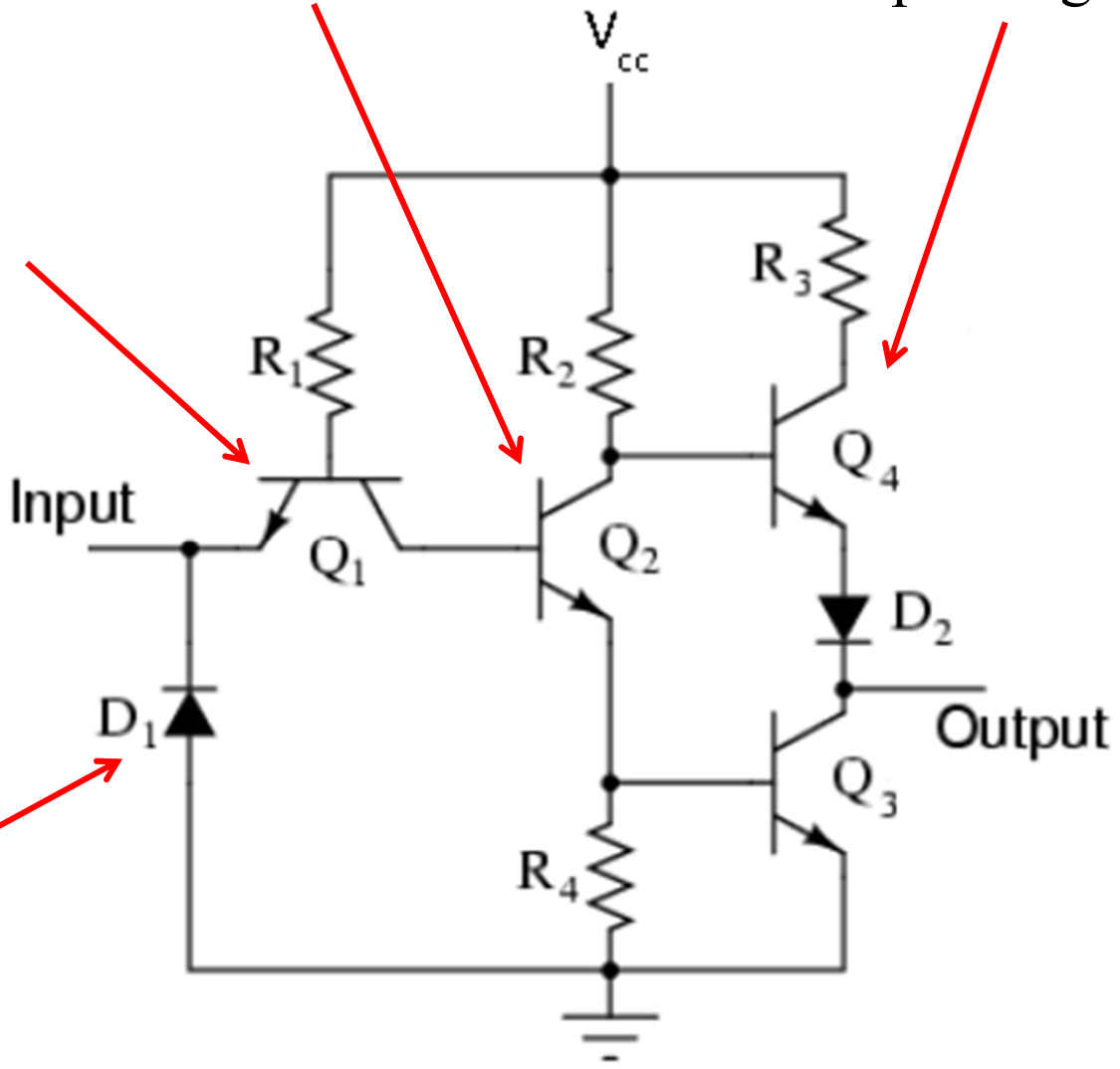
When the output is HIGH the gate is sourcing current.
When the output is LOW the gate is sinking current.

TTL Inverter

Transistor Q1 is being used as two back-to-back diodes. Depending on the logic level of the input, they control the current received at the base of transistor Q2.

Q2 acts as a phase splitter

Totem Pole output stage

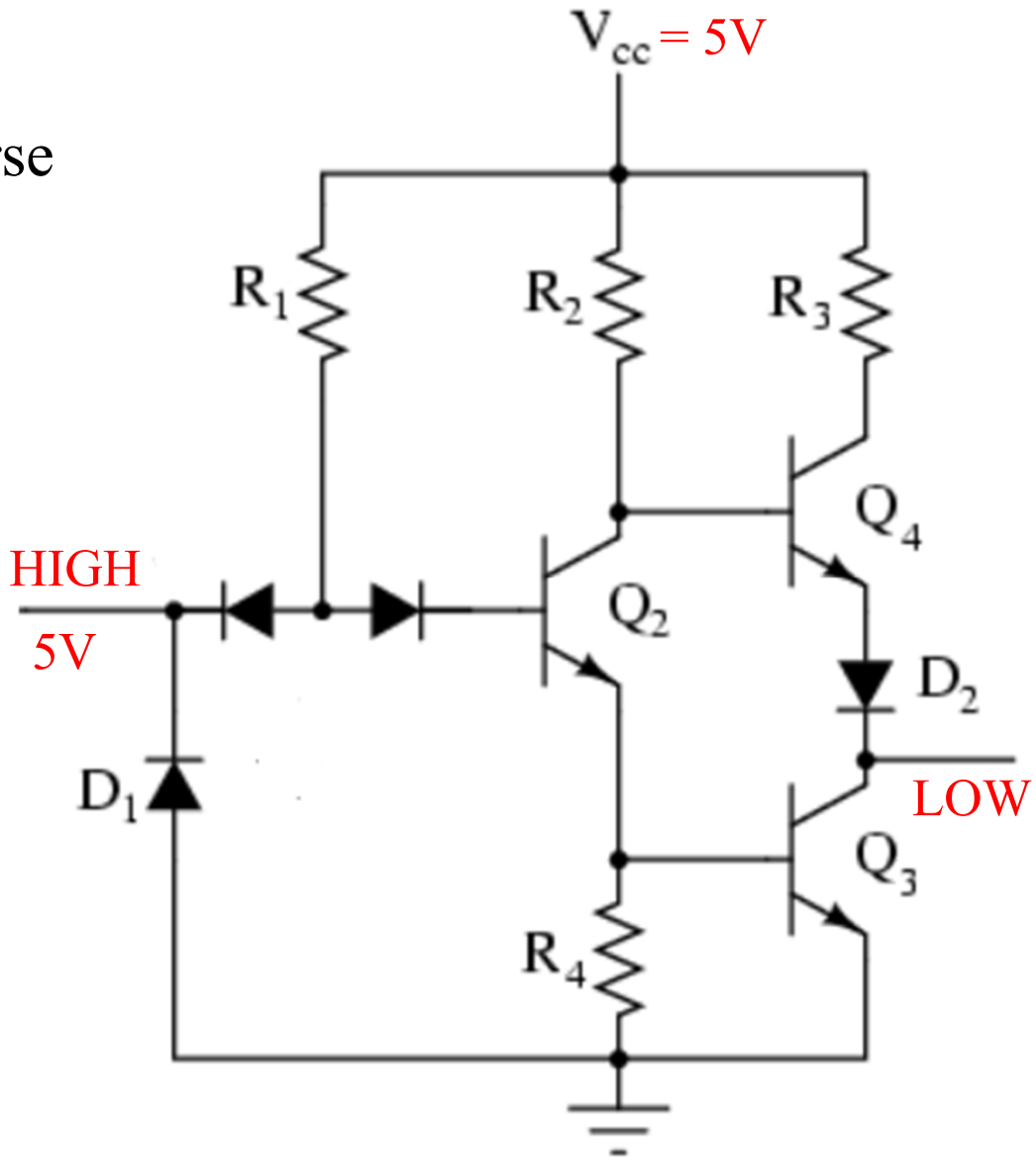


Diode D1 protects the transistor Q1 from negative going spikes on the input.

Input HIGH

The left side Q1 diode is reverse biased and will not conduct.

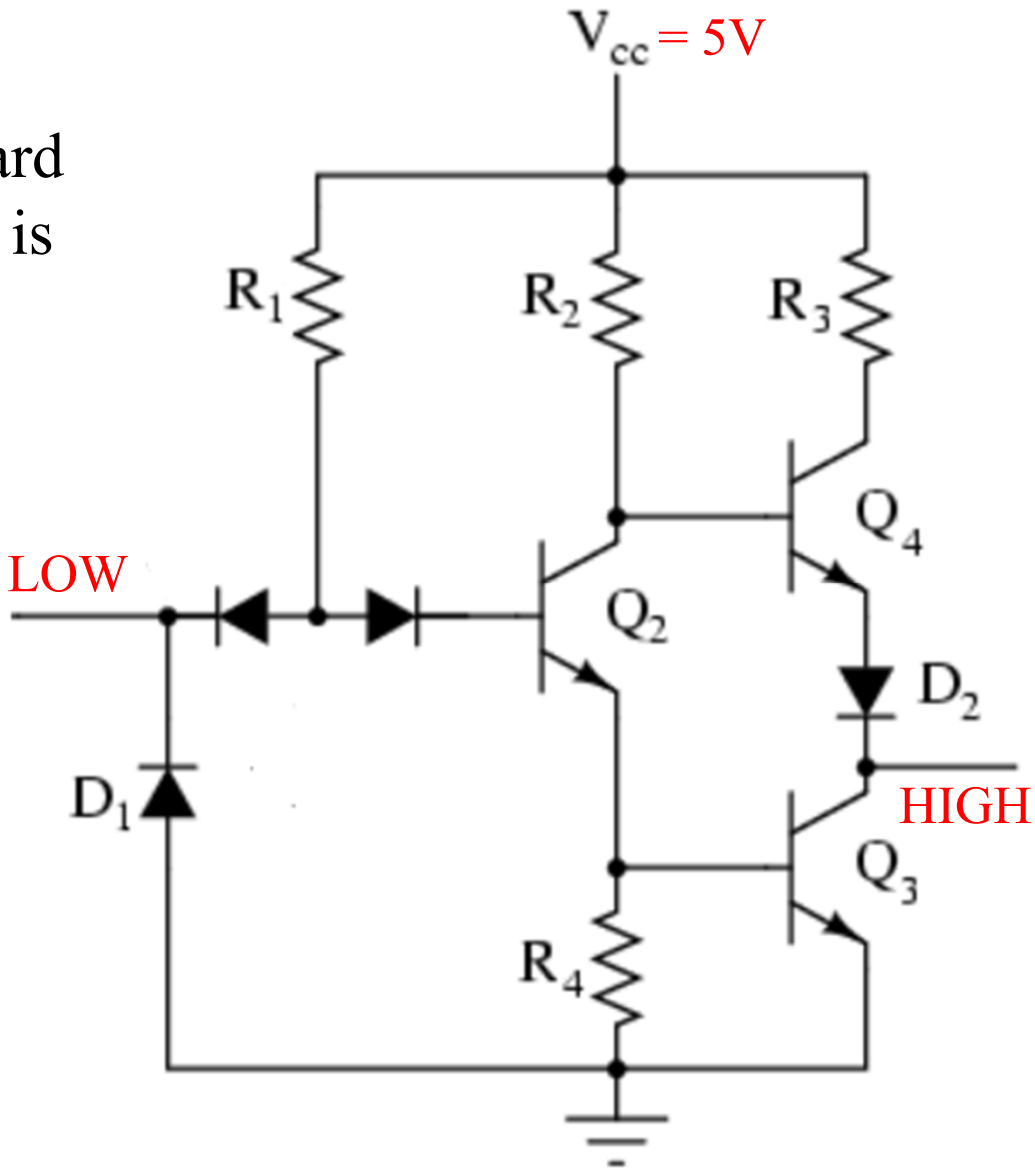
The right side Q1 diode is forward biased and will provide base current to Q2 driving it into saturation. Q3 will then be turned on by Q2. The collector voltage of Q3 and hence the output will be near ground giving a LOW output. The collector of Q2 is low enough to keep Q4 off with the help of the voltage drop across D2.



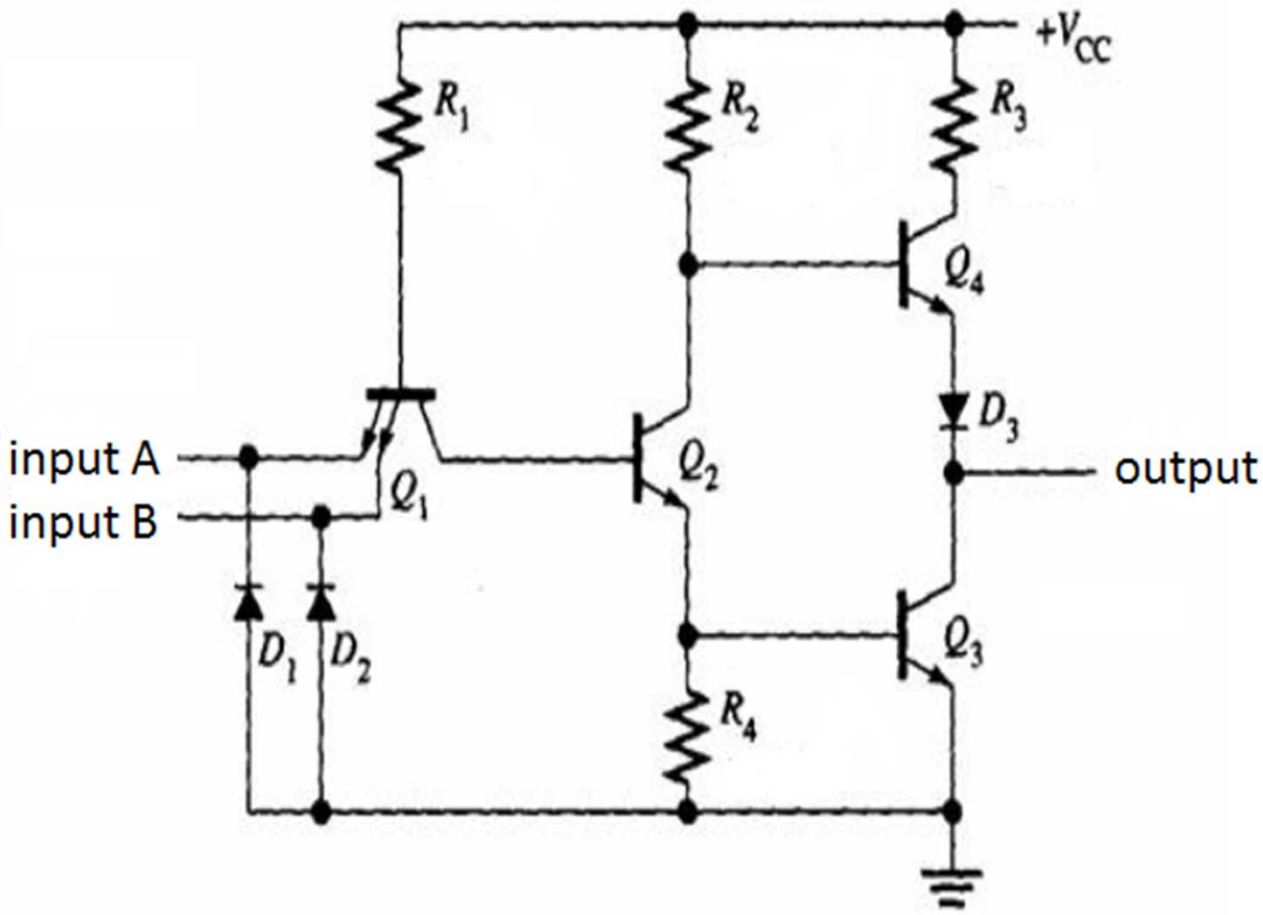
Input LOW

The left side Q1 diode is forward biased and the right side diode is reverse biased.

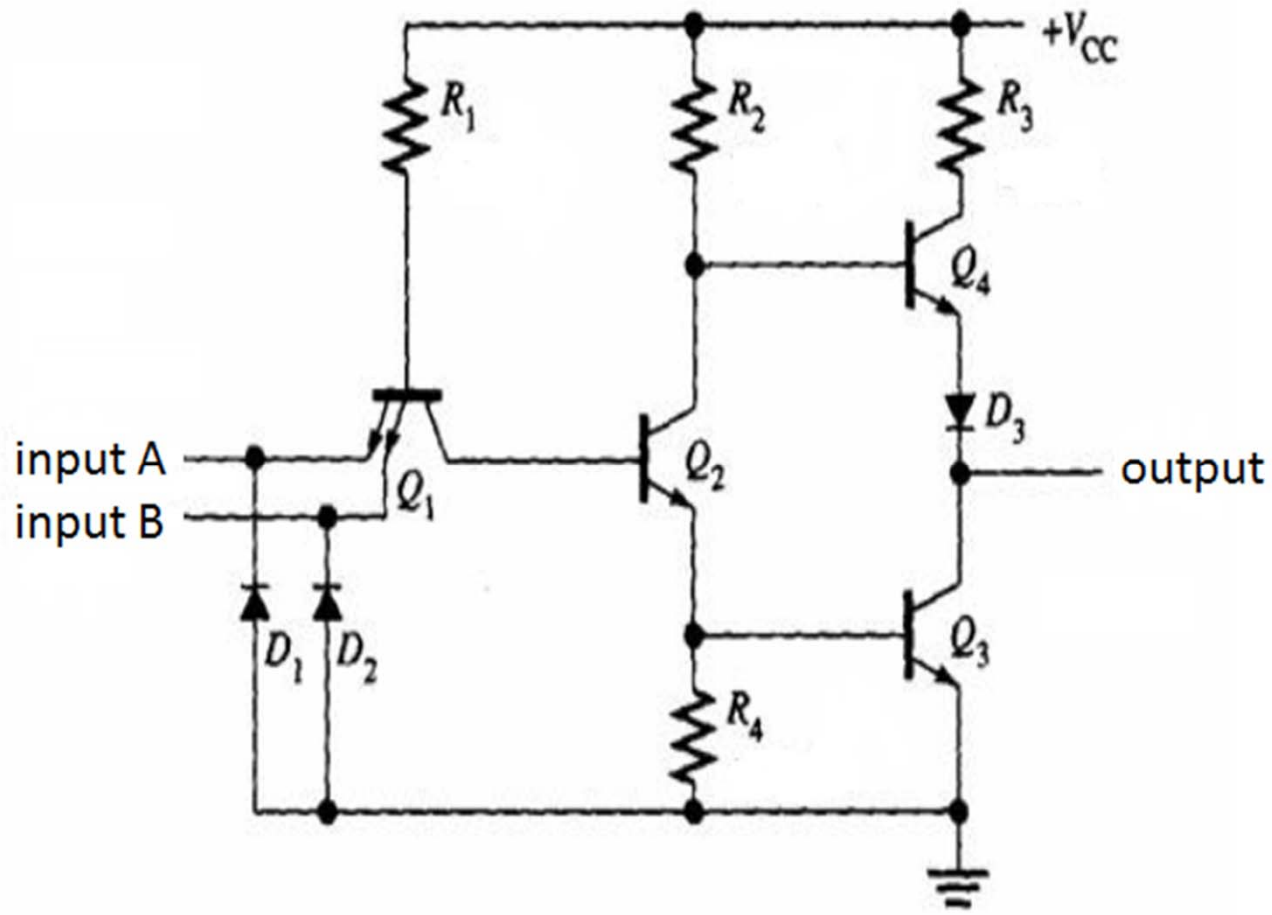
Hence, there is no base current to Q2 and it is cut off. The collector of Q2 is HIGH, turning on Q4 which will be saturated giving a low resistance path to the output which will be HIGH. The emitter of Q2 is at ground keeping Q3 off.



TTL NAND Gate



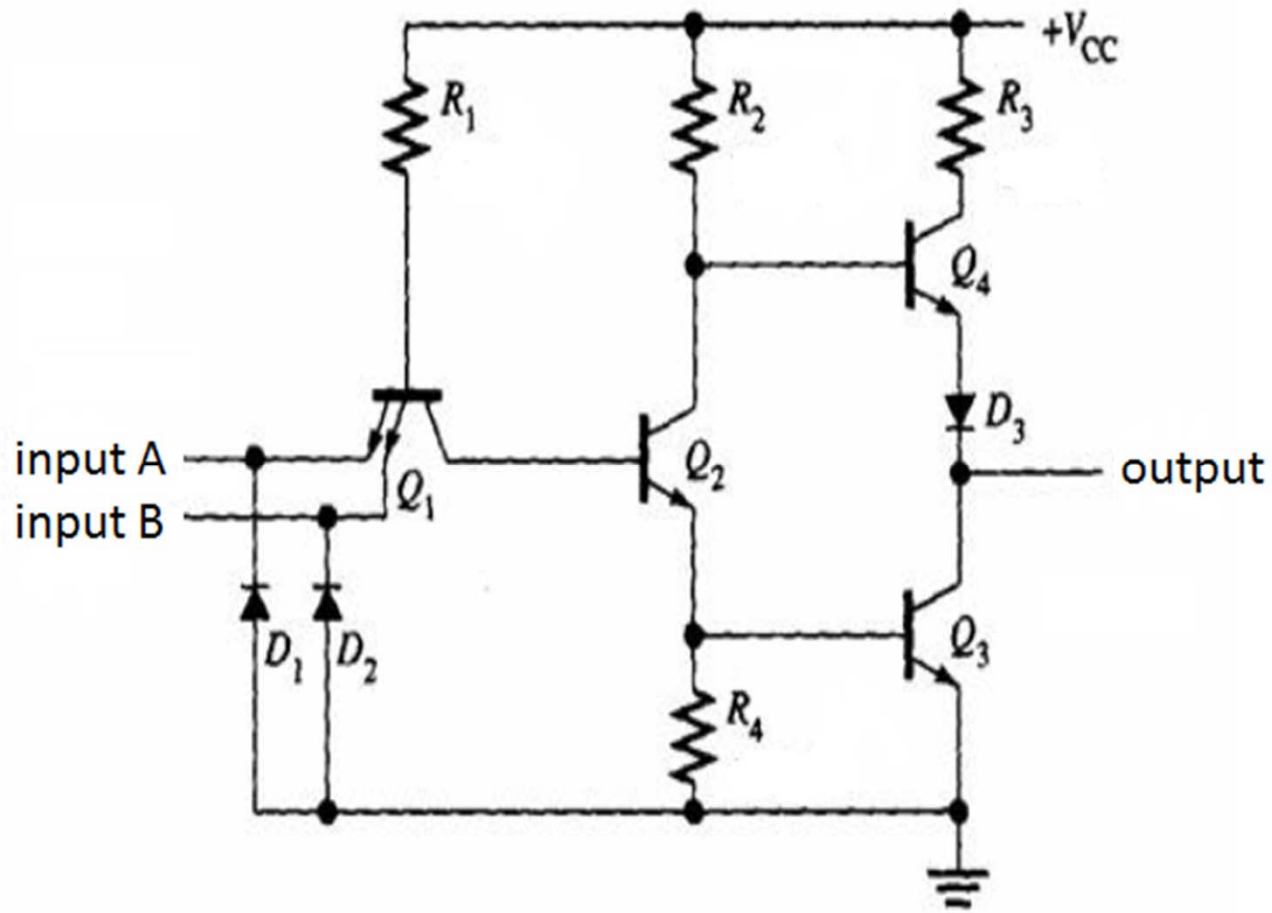
TTL NAND Gate



X	Y	$\overline{X.Y}$
0	0	1
0	1	1
1	0	1
1	1	0

$$F = \overline{A.B}$$

TTL NAND Gate

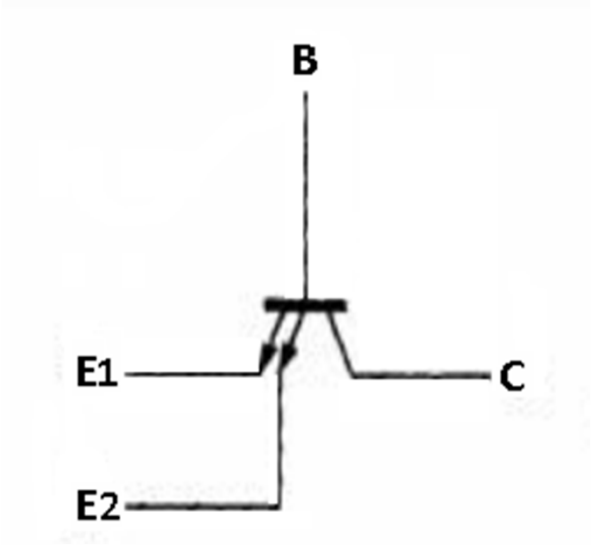


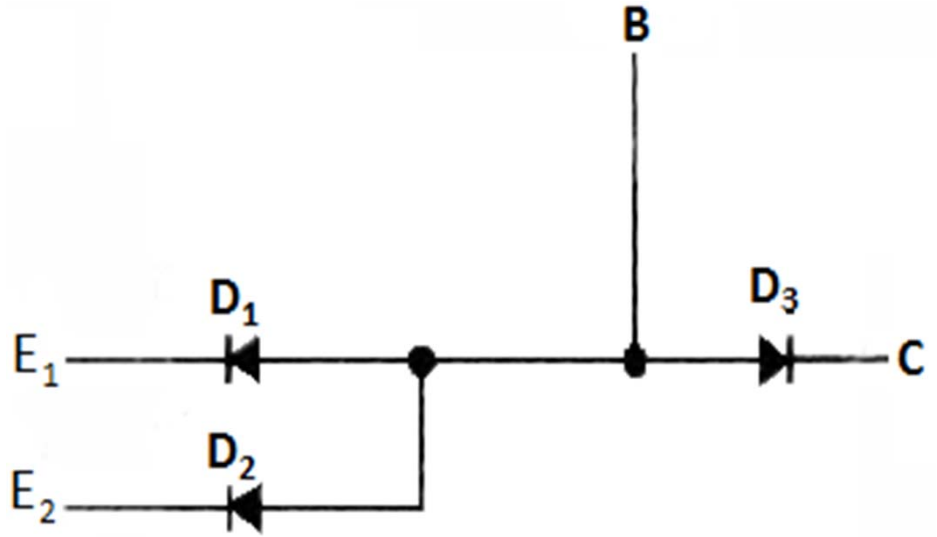
X	Y	$\overline{X.Y}$
0	0	1
0	1	1
1	0	1
1	1	0

$$F = \overline{A.B}$$

$$F = \overline{A} + \overline{B}$$

TTL multiple-emitter transistor



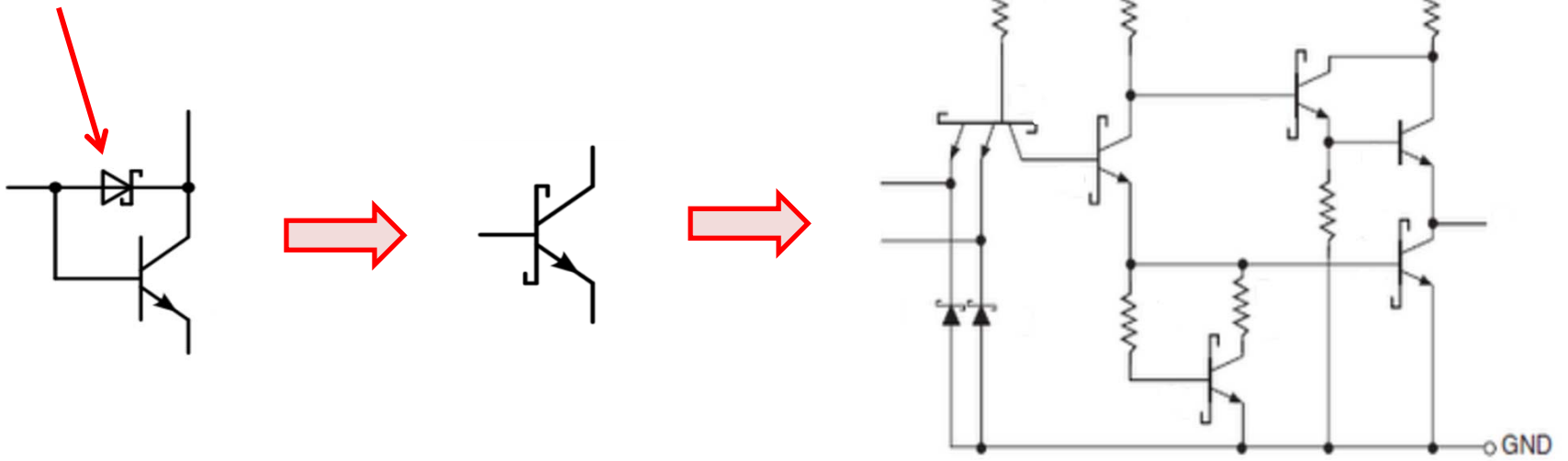


A LOW on E1 will forward bias D1, a Low on E2 will forward bias D2. In both cases, D3 will be reverse biased turning off the following transistor Q2 and giving a HIGH output for the NAND gate (as for the TTL inverter).
The same situation will occur when both inputs are LOW at the same time.

A HIGH on both inputs will reverse bias diodes D1 and D2. In this case, D3 will be forward biased, Q2 turned on and the NAND gate output will be LOW.

NAND gate with Schottky transistors

Schottky Diode: fast switching, small forward volt drop (0.2V).

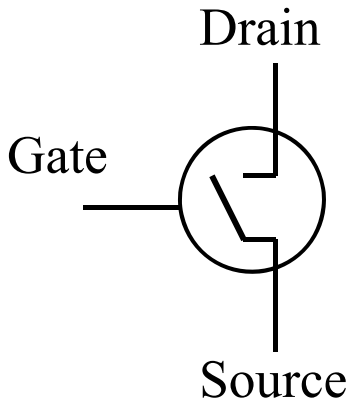


Schottky transistors are normal BJTs with a Schottky diode connected between the base and collector. They prevent the transistors from going deep into saturation which reduces the turn on and turn off time, hence a faster gate.

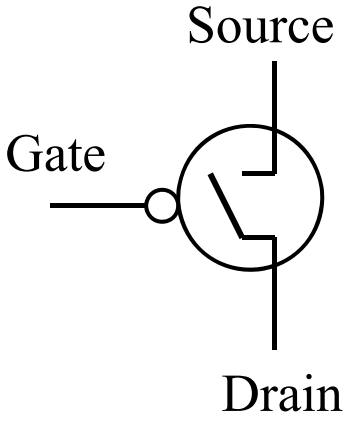
CMOS

In **MOS** technology it is possible to fabricate two complimentary transistors, a **P-transistor** and an **N-transistor**.

N-transistor: closed when the gate voltage is high and open when the gate voltage is low.

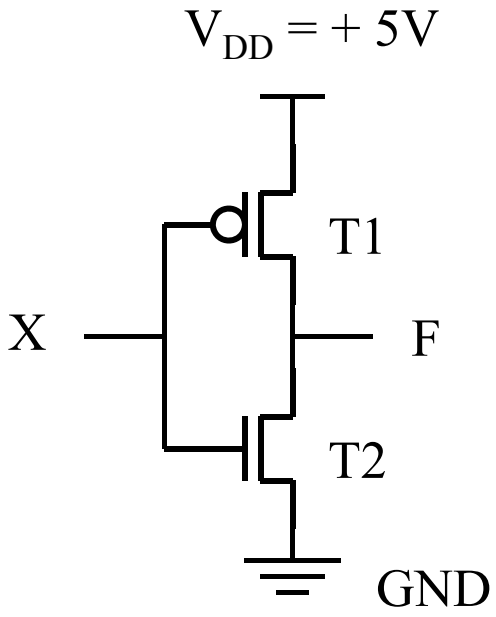


P-transistor: closed when the gate voltage is low and open when the gate voltage is high.



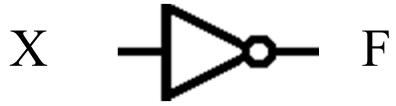
The P-transistor makes a good pull-up circuit and the N-transistor makes a good pull-down circuit.

Inverter



X	T1	T2	F
L	on	off	H
H	off	on	L

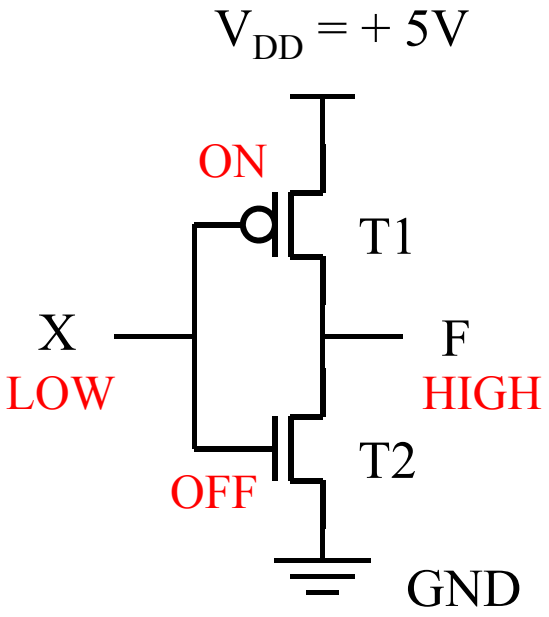
Inverter truth table



The inverter consists of two transistors. When the input is low, the pull-up transistor T1 is switched on and the pull-down transistor T2 is switched off. This gives a high output.

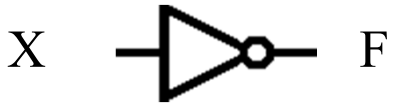
When the input is high, the pull-up transistor T1 is switched off and the pull-down transistor T2 is switched on. This gives a low output.

Inverter



X	T1	T2	F
L	on	off	H
H	off	on	L

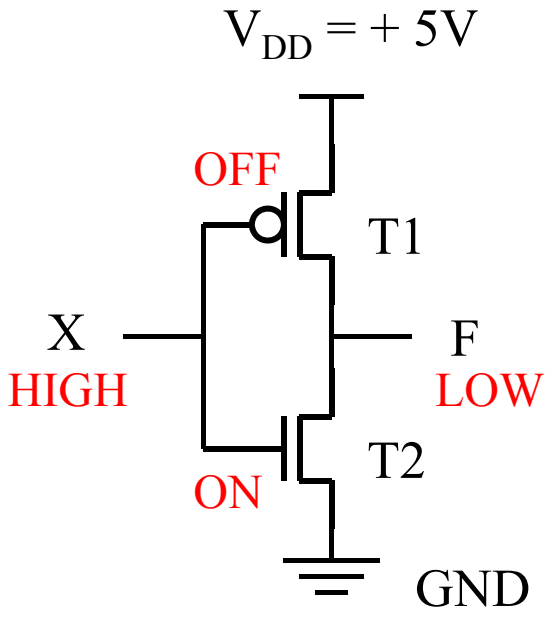
Inverter truth table



The inverter consists of two transistors. When the input is low, the pull-up transistor T1 is switched on and the pull-down transistor T2 is switched off. This gives a high output.

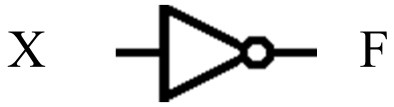
When the input is high, the pull-up transistor T1 is switched off and the pull-down transistor T2 is switched on. This gives a low output.

Inverter



X	T1	T2	F
L	on	off	H
H	off	on	L

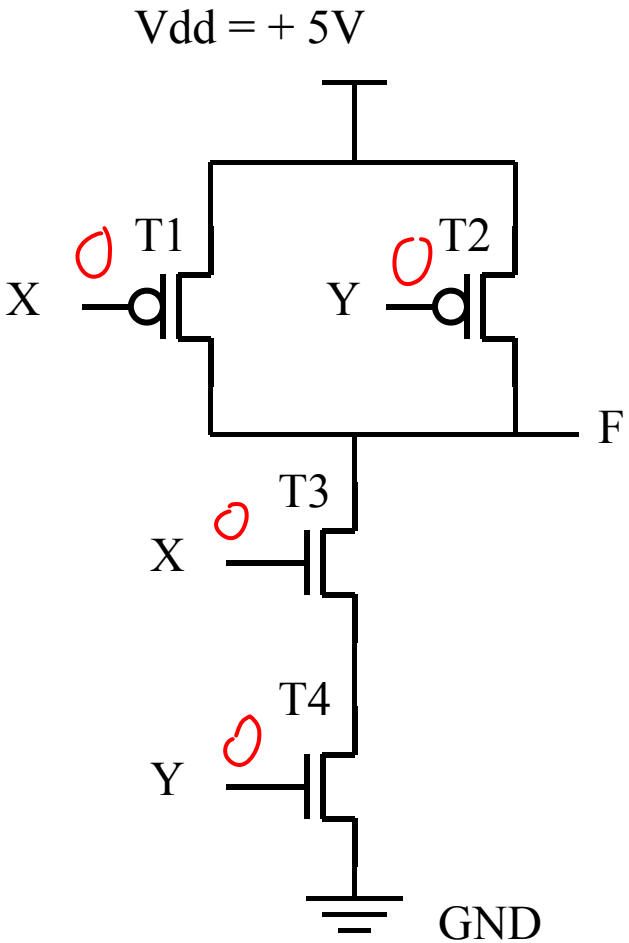
Inverter truth table



The inverter consists of two transistors. When the input is low, the pull-up transistor T1 is switched on and the pull-down transistor T2 is switched off. This gives a high output.

When the input is high, the pull-up transistor T1 is switched off and the pull-down transistor T2 is switched on. This gives a low output.

NAND Gate

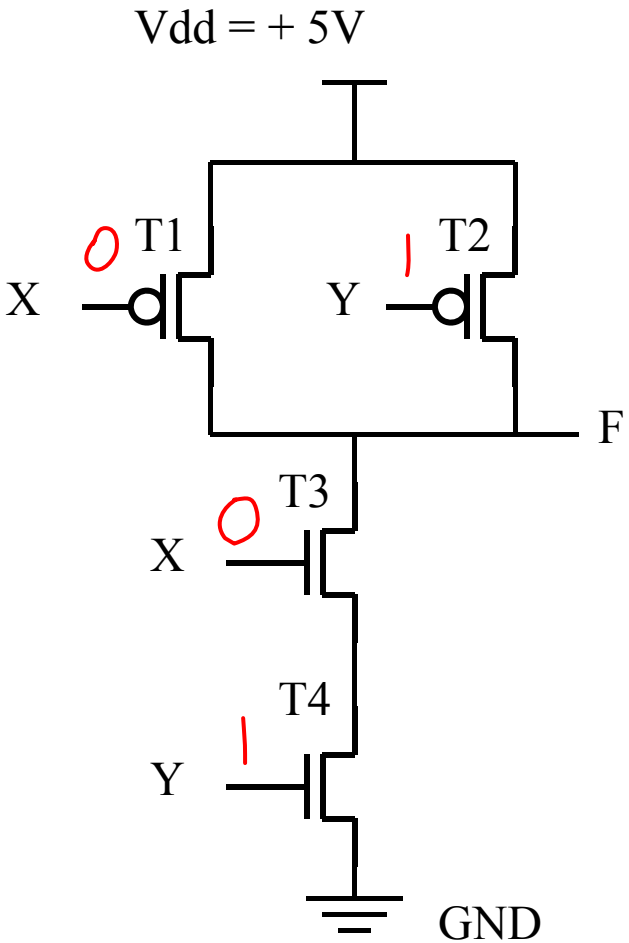


X	Y	T1	T2	T3	T4	F
L	L	on	on	off	off	H

NAND truth table

The NAND circuit consists of a parallel pull-up network and a series pull-down network.

NAND Gate

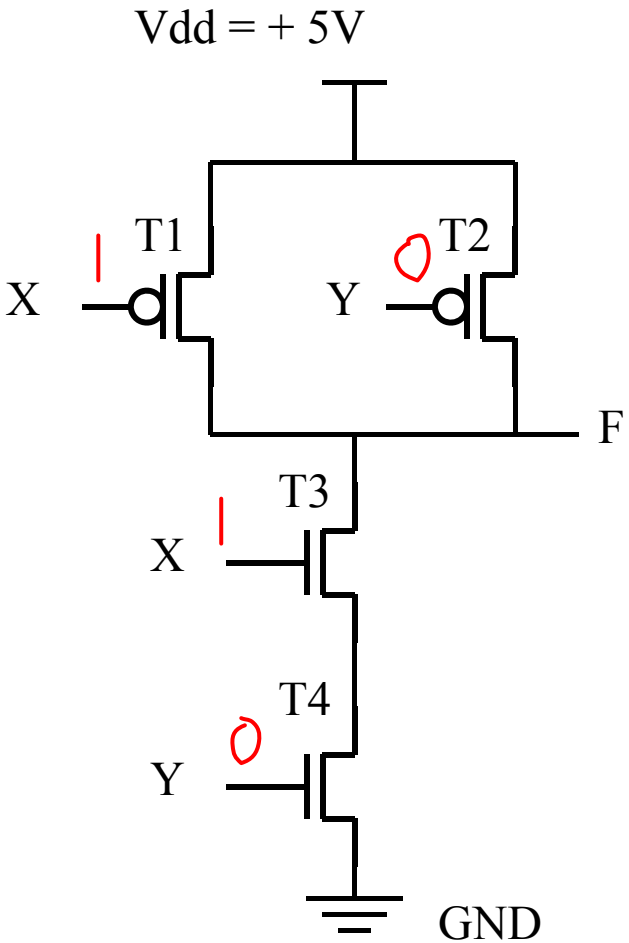


X Y	T1 T2 T3 T4	F
L H	on off off on	H

NAND truth table

The NAND circuit consists of a parallel pull-up network and a series pull-down network.

NAND Gate

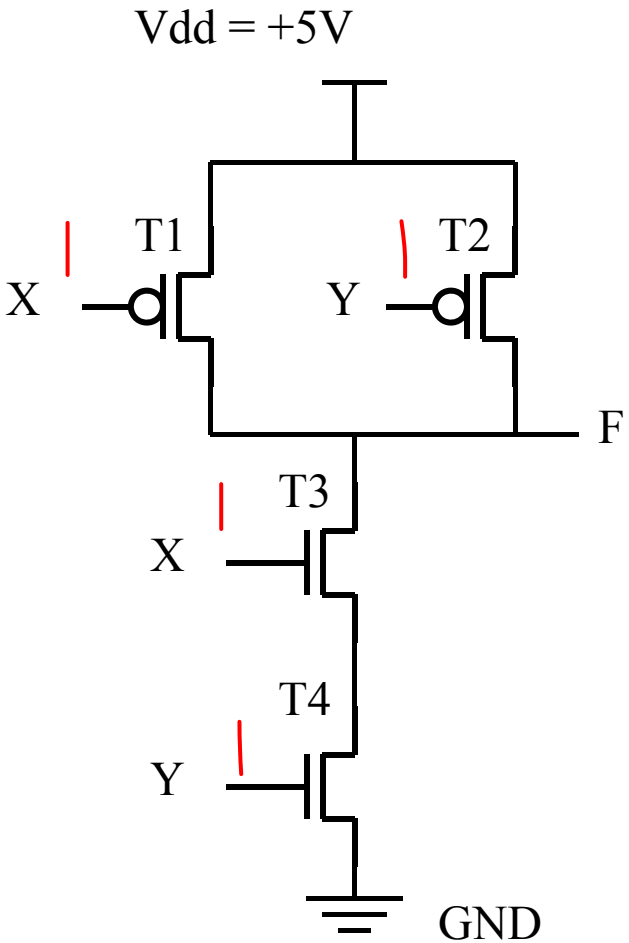


X	Y	T1	T2	T3	T4	F
H	L	off	on	on	off	H

NAND truth table

The NAND circuit consists of a parallel pull-up network and a series pull-down network.

NAND Gate

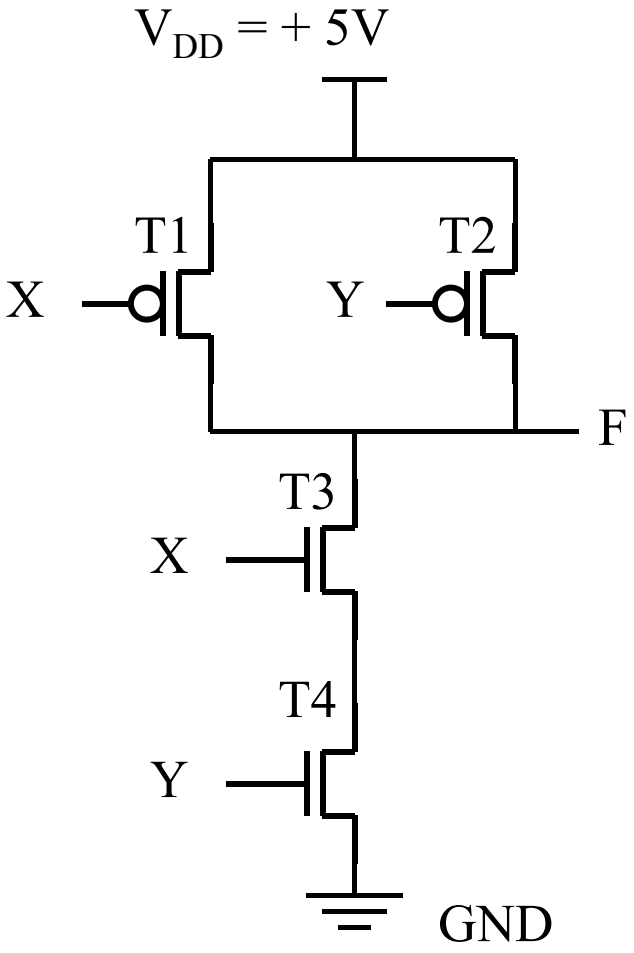


X	Y	T1	T2	T3	T4	F
H	H	off	off	on	on	L

NAND truth table

The NAND circuit consists of a parallel pull-up network and a series pull-down network.

NAND Gate

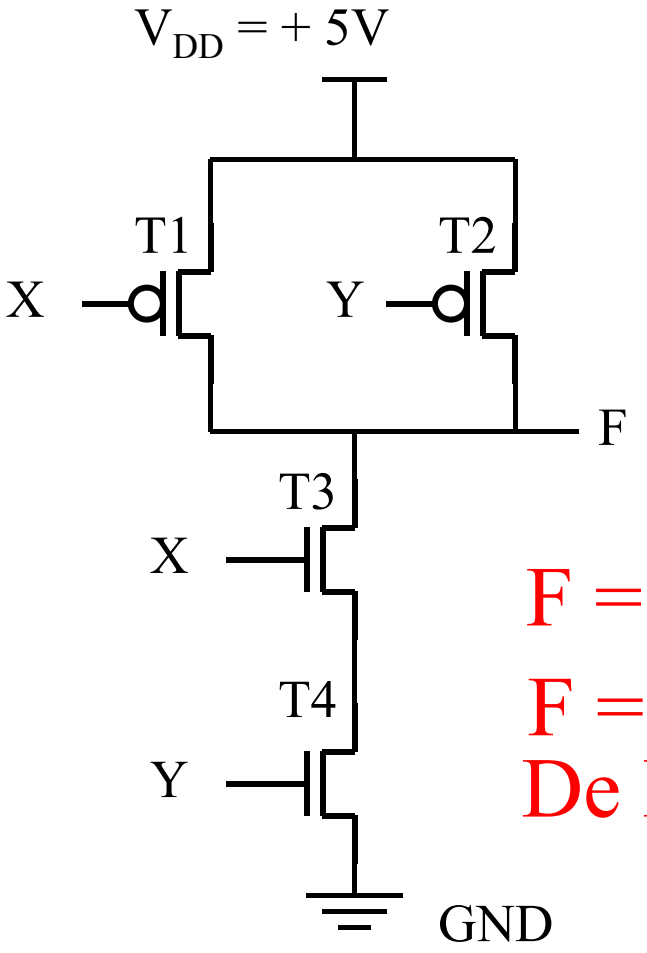


X	Y	T1	T2	T3	T4	F
L	L	on	on	off	off	H
L	H	on	off	off	on	H
H	L	off	on	on	off	H
H	H	off	off	on	on	L

NAND truth table

The NAND circuit consists of a parallel pull-up network and a series pull-down network.

NAND Gate



$$F = \overline{A.B}$$

$$F = \overline{A} + \overline{B}$$

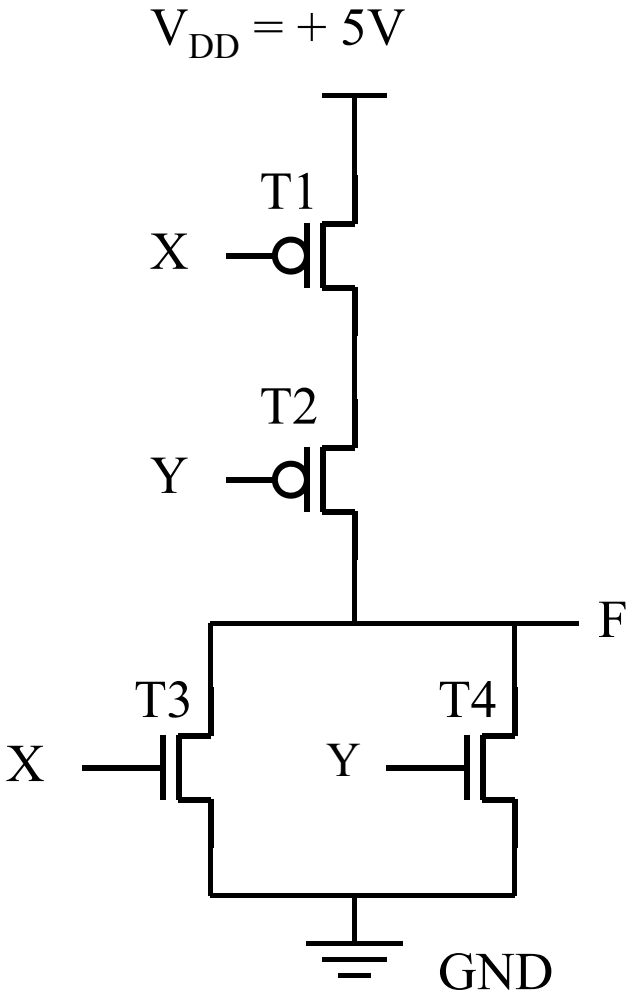
De Morgan

X	Y	T1	T2	T3	T4	F
L	L	on	on	off	off	H
L	H	on	off	off	on	H
H	L	off	on	on	off	H
H	H	off	off	on	on	L

NAND truth table

The NAND circuit consists of a parallel pull-up network and a series pull-down network.

NOR Gate

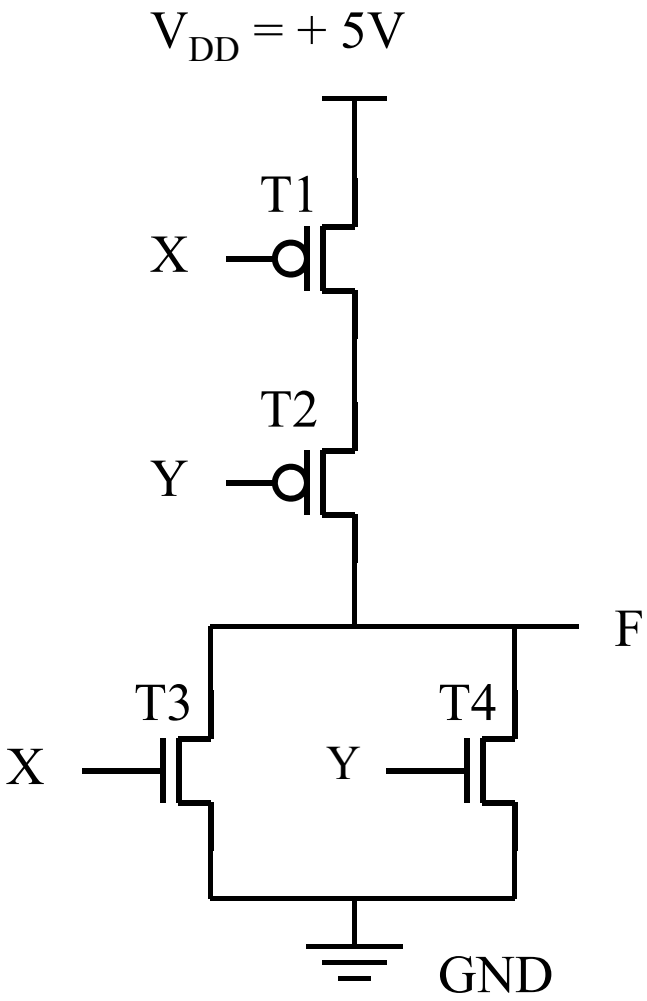


X	Y	T1	T2	T3	T4	F
L	L	on	on	off	off	H
L	H	on	off	off	on	L
H	L	off	on	on	off	L
H	H	off	off	on	on	L

NOR truth table

The NOR circuit consists of a series pull-up network and a parallel pull-down network.

NOR Gate



X	Y	T1	T2	T3	T4	F
L	L	on	on	off	off	H
L	H	on	off	off	on	L
H	L	off	on	on	off	L
H	H	off	off	on	on	L

NOR truth table

$$F = \overline{A + B}$$

$$\overline{F} = A + B$$

The NOR circuit consists of a series pull-up network and a parallel pull-down network.

Verilog

- Verilog is a Hardware Description Language (HDL)
- Describes electronic systems in a textual form
- Documents designs
- Technology Independent
- It is an IEEE standard – number 1364

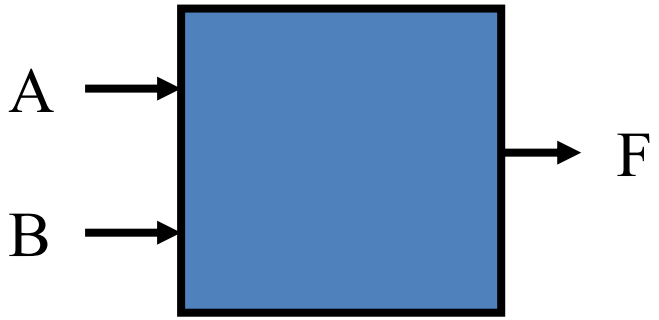
Important ! - You are only required to be able to read and understand a Verilog description.

www.xilinx.com - webpack

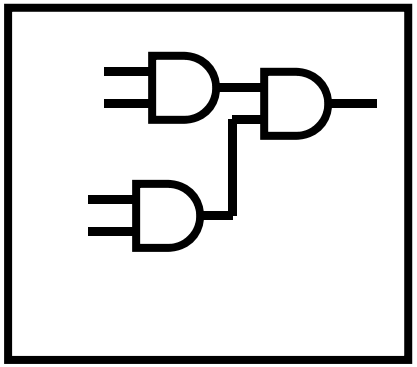
Verilog Module

The basic building block in Verilog is called a module. It is declared with the keyword **module** and always ends with the keyword **endmodule**. A digital system can be described in Verilog by a set of these modules.

Port declarations detail the interface of a module to other modules or the outside world.

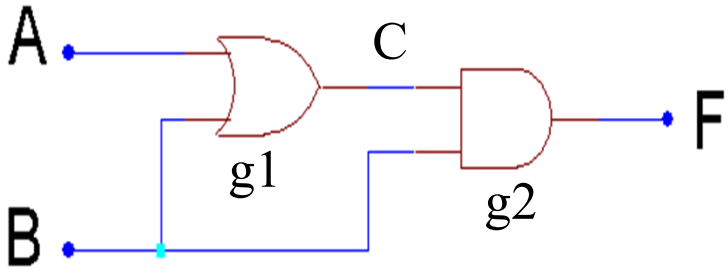


The module body describes the function of the model and the relationship between the ports.



Structural Description

The structure of a circuit can be described in Verilog using predefined gate primitives. These include **not**, **and**, **nand**, **or**, **nor**, **xor**, **xnor**.



Text after // is interpreted as a comment. Keywords are in bold.

```
module example1 (A, B, F);        // module name and port list
input A, B;                       // declare input ports
output F;                         // declare output ports
wire C;                           // declare internal connection
or g1(C, A, B);                   // or gate with optional name g1
and g2(F, C, B);                  // and gate with optional name g2
endmodule                         // gate outputs come first in list
```

Switch-Level Modelling

CMOS transistors can be specified in Verilog using the keywords **pmos** and **nmos**. They are instantiated by specifying the terminal connections.

nmos (drain, source, gate);
pmos (drain, source, gate);

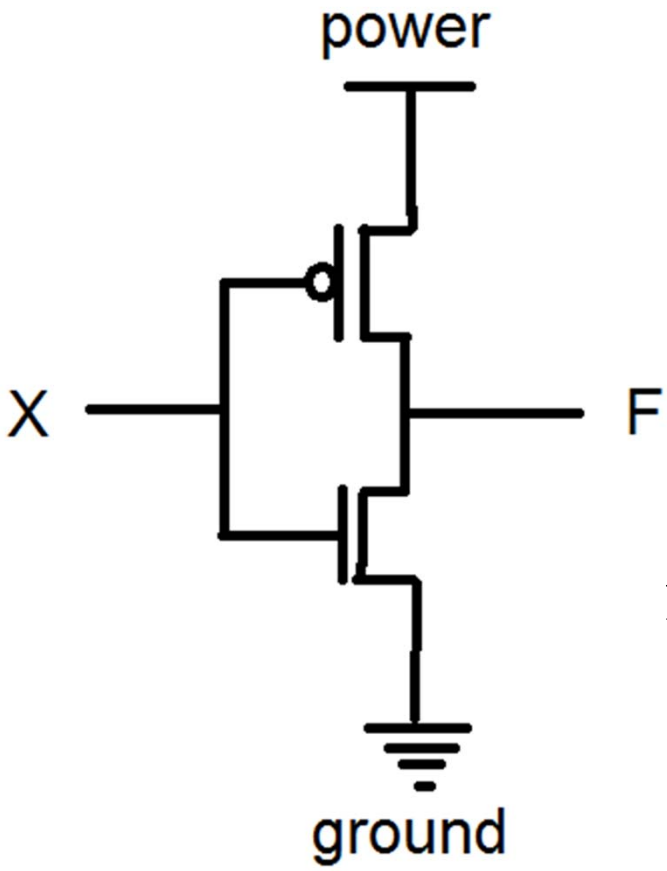
Connections to power (V_{DD}) and ground (gnd) are specified using the keywords **supply1** and **supply0**.

supply1 power;
supply0 ground;

What is the logical function of the following switch-level circuits:

Example 1

```
module gate_one (F, X);
output F;
input X;
supply1 power;
supply0 ground;
pmos (F, power, X);
nmos (F, ground, X);
endmodule
```

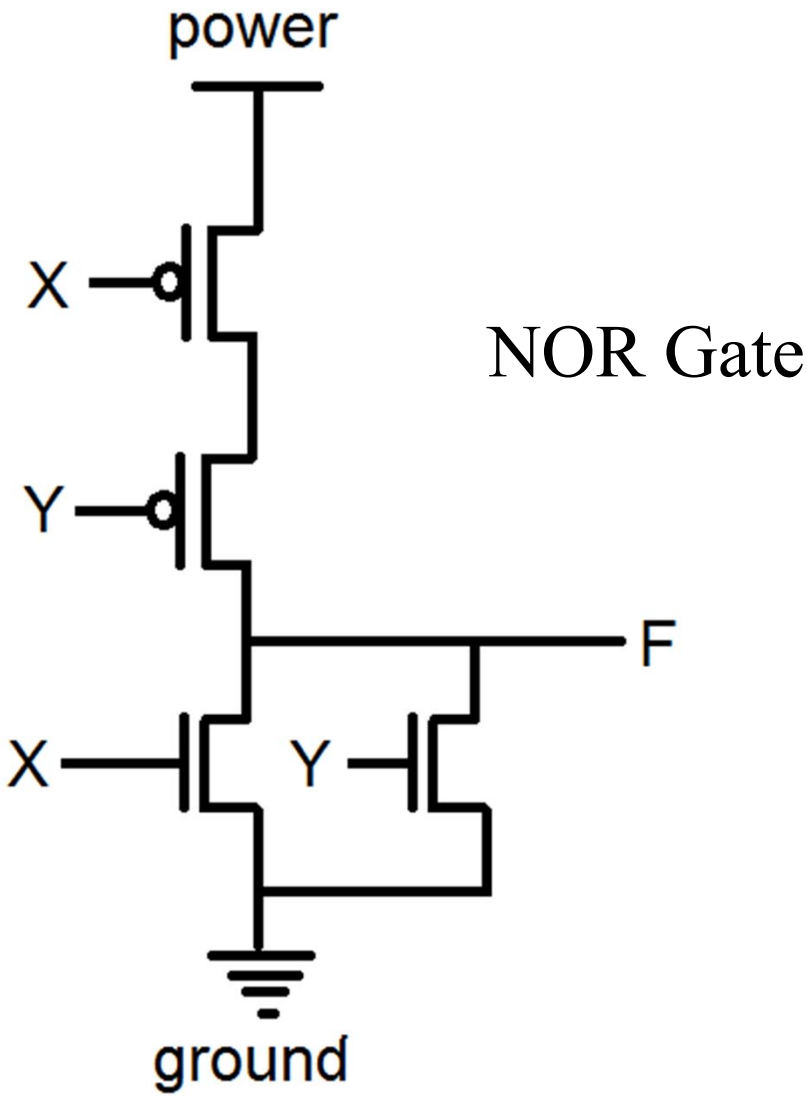


inverter

What is the logical function of the following switch-level circuits:

Example 2

```
module gate_two (F, X, Y);
output F;
input X, Y;
wire C;
supply1 power;
supply0 ground;
pmos (C, power, X);
pmos (F, C, Y);
nmos (F, ground, X);
nmos (F, ground, Y);
endmodule
```



NOR Gate