2) Base conversion

a) B to decimal

$$a_m \cdots a_2 a_1 a_0 = \sum \underbrace{a_i B^i}_{\text{expressed in decimal}}$$

$$243_6 = 2 \times 6^2 + 4 \times 6^1 + 3 \times 6^0 = 99_{10}$$

b) B to B'

$$(a_m \cdots a_2 a_1 a_0)_B = \sum \underbrace{a_i B^i}_{\text{expressed in base B'}}$$

$e.g.$ decimal to binary:

$$493_{10} = \overset{\text{4 in binary}}{100} \times (\overset{\text{10 in binary}}{1010})^{10} + \overset{9}{1001} \times (1010)^1 + \overset{3}{11} \times (1010)^0$$

$$\begin{array}{r} 1010 \\ \underline{1010} \\ 0000 \\ 1010 \\ 0000 \\ 1010 \\ \hline 100\times \quad 1100100 \end{array} \qquad\qquad \begin{array}{r} 1001 \\ \underline{1010} \\ 0000 \\ 1001 \\ 0000 \\ 1001 \\ \hline 1011010 \end{array}$$

$$493_{10} = 110010000 \quad + 1011010 \quad + 11 \quad = 111101101$$

check: $1 + 0 + 4 + 8 + 0 + 32 + 64 + 128 + 256 \quad = 493$

LSB

.

.

.

.

.

.

.

.

.

.

MSB

$$\mathrm{mod}(x,B)$$

$$\mathrm{mod}\left(\mathrm{int}\left(\frac{x}{B}\right),B\right)$$

$$\mathrm{mod}\left(\mathrm{int}\left(\frac{\mathrm{int}\left(\frac{x}{B}\right)}{B}\right),B\right)$$

.

.

.

$$493_{10} = 111101101_2$$

$$493/2 = 246 \quad \text{rem } 1$$

$$246/2 = 123 \quad \text{rem } 0$$

$$123/2 = 61 \quad \text{rem } 1$$

$$61/2 = 30 \quad \text{rem } 1$$

$$30/2 = 15 \quad \text{rem } 0$$

$$15/2 = 7 \quad \text{rem } 1$$

$$7/2 = 3 \quad \text{rem } 1$$

$$3/2 = 1 \quad \text{rem } 1$$

$$1/2 = 0 \quad \text{rem } 1$$

Octal ↔ Binary

$$a_m \cdots a_1 a_0 = \sum a_i B^i = \sum a_i \overset{8}{(1000)}^i$$

$$372_8 = (011)(1000)^{\overset{2}{10}} + 111(1000)^1 + 010$$

$$\begin{array}{r} 011\ 000\ 000 \\ 111\ 000 \\ 010 \\ \hline 011\ \ 111\ \ 010 \\ 3\quad\ \ 7\quad\ \ 2 \end{array}$$

Each octal digit can be replaced by the equivalent 3-bit binary number.

Hexadecimal ↔ Binary

$$a_m \cdots a_1 a_0 = \sum a_i B^i = \sum a_i \overset{16}{(10000)}^i$$

$$\$D49 = (1101)(10000)^{10} + 0100(10000)^1 + 1001$$

$$\begin{array}{r} 1101\ \ 0000\ \ 0000 \\ 0100\ \ 0000 \\ 1010 \\ \hline 1101\ \ \ 0100\ \ \ 1001 \\ D\qquad 4\qquad 9 \end{array}$$

Each hex digit can be replaced by the equivalent 4-bit binary number.

Decimal —> hex —> binary

Recall $493_{10} = 111101101_2$

$493/16 = 30$ rem=13 or D

$30/16 = 1$ rem=14 or E

$1/16 = 0$ rem=1

$1ED = 1 1110 1101

1 E D

3) Boolean Algebra

Algebra of two-valued variables (T, F or 1,0)

with 3 fundamental operators: logical AND, OR, NOT

a) logical AND *symbol ·*

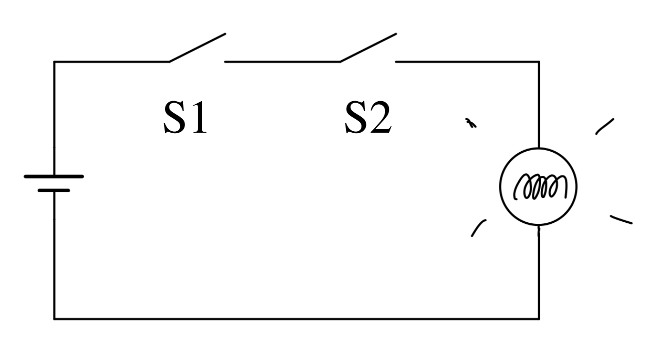
Truth table for $A \cdot B$

A	B	$A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

Other symbols: $\wedge$ $\cap$

or no symbol: $A \cdot B = AB$

e.g.



Lamp is on if S1·S2 = 1

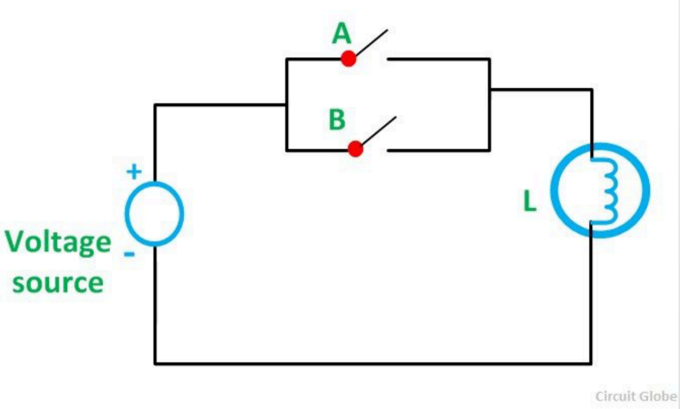
b) logical OR *(inclusive OR)* *symbol +*

Truth table for $A+B$

A	B	$A+B$
0	0	0
0	1	1
1	0	1
1	1	1

Other symbols: $\vee$ $\cup$

e.g.



Lamp is on if A+B = 1

c) logical NOT (complement)

symbol $\overline{A}$

Truth table for $\overline{A}$

A	$\overline{A}$
0	1
1	0

Other symbols: ¬ ~

$$\overline{1} = 0$$

$$\overline{0} = 1$$

d) Identities

$A \cdot A = A$	$1 \cdot 1 = 1$	$0 \cdot 0 = 0$	$A + A = A$	$1 + 1 = 1$	$0 + 0 = 0$
$\overline{A} \cdot A = 0$	$0 \cdot 1 = 0$	$1 \cdot 0 = 0$	$\overline{A} + A = 1$	$1 + 0 = 1$	$0 + 1 = 1$
$A \cdot 1 = A$	$0 \cdot 1 = 0$	$1 \cdot 1 = 1$	$A + 1 = 1$	$1 + 1 = 1$	$0 + 1 = 1$
$A \cdot 0 = 0$	$0 \cdot 0 = 0$	$1 \cdot 0 = 0$	$A + 0 = A$	$0 + 0 = 0$	$1 + 0 = 1$

e) Theorems

(i) Commutative

$$A \cdot B = B \cdot A$$

$$A + B = B + A$$

A	*B*	*A+B*
0	0	0
0	1	1
1	0	1
1	1	1

A	*B*	*A·B*
0	0	0
0	1	0
1	0	0
1	1	1

(ii) Associative

$$(A \cdot B) \cdot C = A \cdot (B \cdot C)$$

$$(A + B) + C = A + (B + C)$$

→

A	B	C	A+B	(A+B)+C	(B+C)	A+(B+C)
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	1	1	1	1
0	1	1	1	1	1	1
1	0	0	1	1	0	1
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	1	1	1	1

(iii) Distributive

$$x \cdot (y + z) = x \cdot y + x \cdot z$$

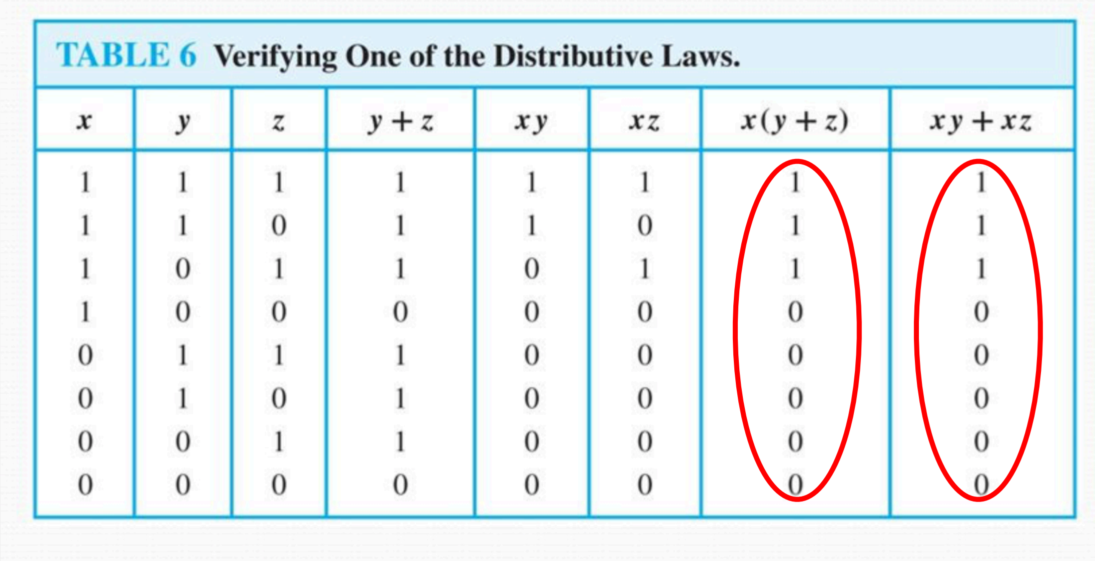


TABLE 6 Verifying One of the Distributive Laws.

x	y	z	$y+z$	xy	xz	$x(y+z)$	$xy+xz$
1	1	1	1	1	1	1	1
1	1	0	1	1	0	1	1
1	0	1	1	0	1	1	1
1	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	0	1	1	0	0	0	0
0	0	0	0	0	0	0	0

$$(A + B) \cdot (A + C) = A + B \cdot C$$

$$\begin{aligned} (A+B)\cdot(A+C) &= (A+B)\cdot A + (A+B)\cdot C \\ &= A \cdot A + B \cdot A + A \cdot C + B \cdot C \\ &= A(1 + B + C) + B \cdot C \\ &= A + B \cdot C \end{aligned}$$

(iii) De Morgan's theorem

Any binary expression is unchanged if

1) complement all variables
2) replace ORs with ANDs
3) replace ANDs with ORs
4) complement entire expression

e.g.

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

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

$$\left.\begin{array}{lll}\overline{A+B} & =\overline{\overline{\overline{A}\cdot\overline{B}}} & =\overline{A}\cdot\overline{B} \\ \overline{A\cdot B} & =\overline{\overline{\overline{A}+\overline{B}}} & =\overline{A}+\overline{B}\end{array}\right\}\ \begin{array}{l}\text{break the line}\\ \text{change the sign}\end{array}$$

Proof of two De Morgan relations

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

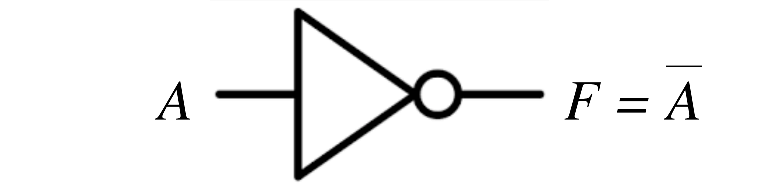
A	B	$A+B$	$\overline{A+B}$	$\overline{A}$	$\overline{B}$	$\overline{A}\cdot\overline{B}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

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

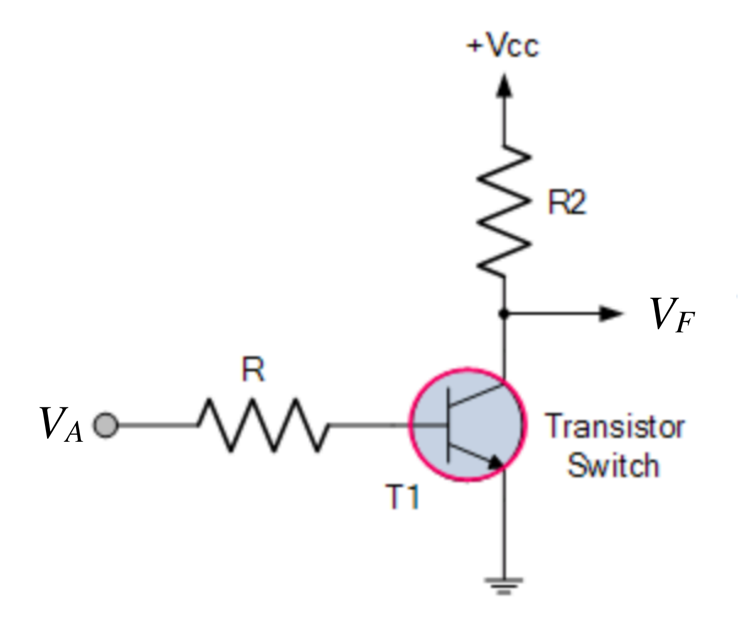
A	B	$A\cdot B$	$\overline{A+B}$	$\overline{A}$	$\overline{B}$	$\overline{A}+\overline{B}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

4) Binary gates

a) NOT



A	$\overline{A}$
0	1
1	0



V_A	V_F
0	5 V
5 V	0.1 V

positive logic:

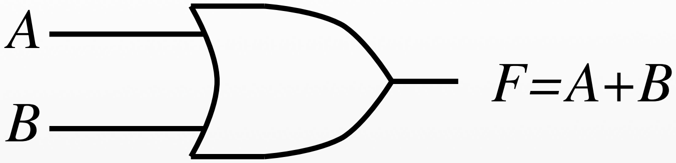
0 —> ~ 0 V

1 —> ~ 5V

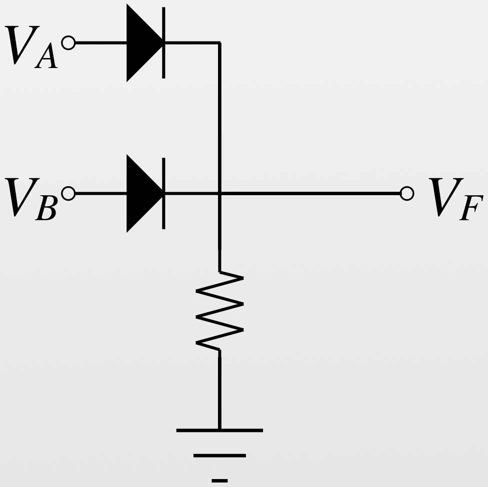
If the input is high, the transistor turns on, grounding the output.

If the input is low, the transistor turns off, leaving the output high.

b) OR



A	*B*	*A+B*
0	0	0
0	1	1
1	0	1
1	1	1



V_A	V_B	V_F
0	0	0
0	5V	4.4V
5V	0	4.4V
5V	5V	4.4V

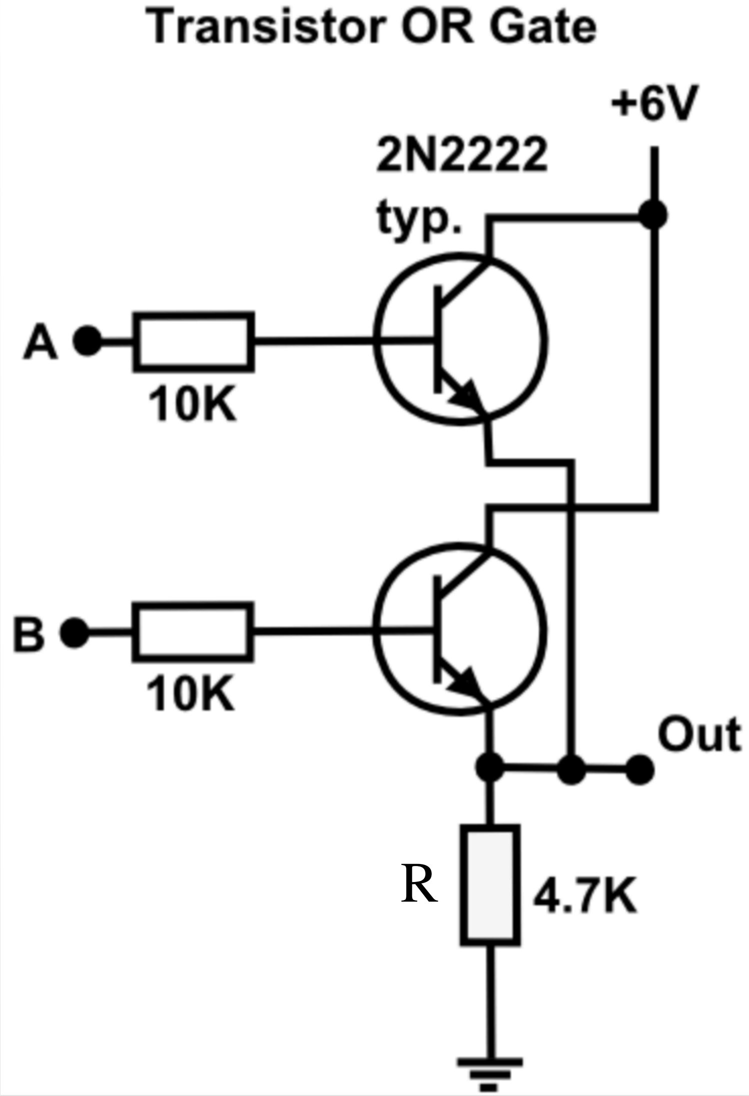
positive logic:

0 —> 0V

1 —> ~5V

A high on either input turns on the respective diode, raising the output high.

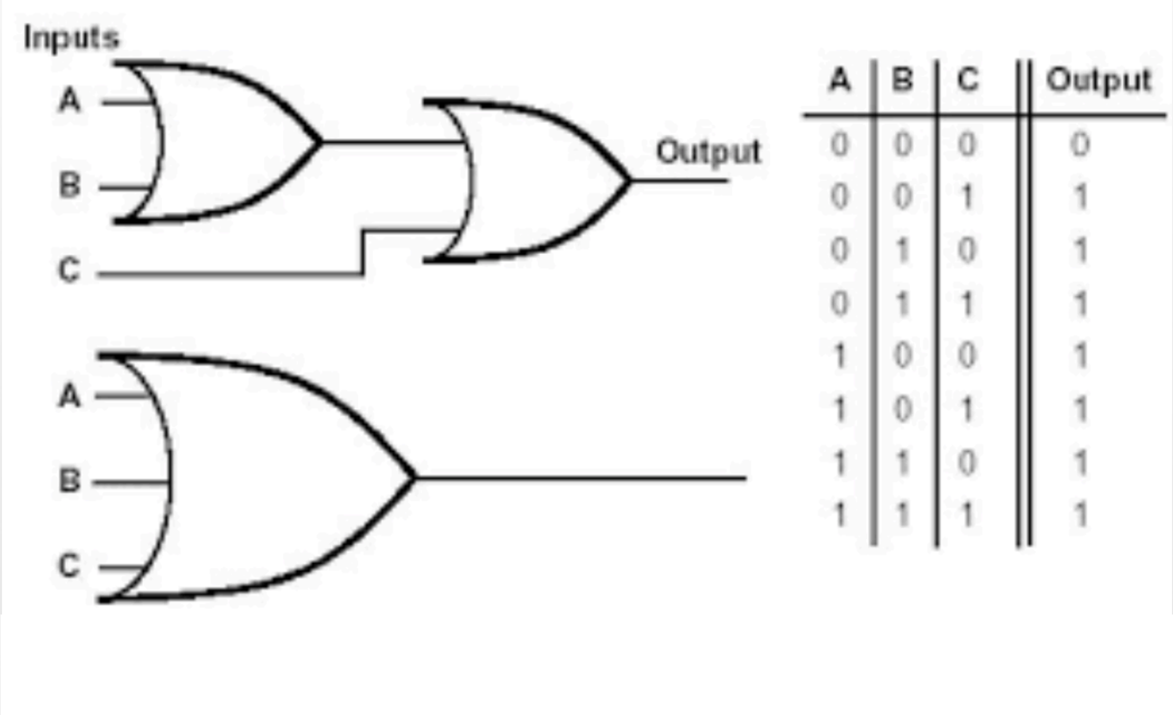
If both inputs are low, the diodes are both off, so no current flows, and the output is low.



If either transistor turns on
(i.e. if either A or B is high)
then the output is tied to Vcc = 6 V.

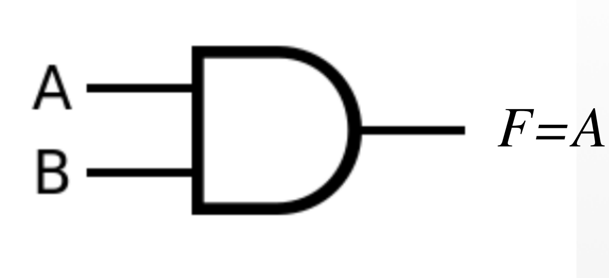
If both transistors are off
(i.e. if both A and B are low)
then there is no current through R,
so the output is low (~ zero).

extends easily to multiple inputs:

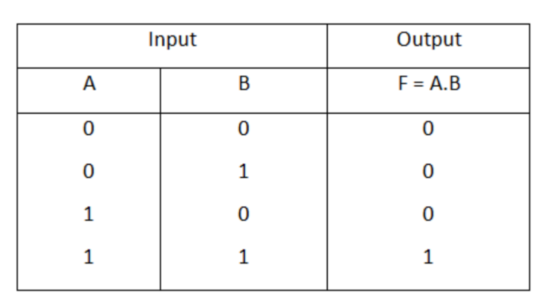


A	B	C	Output
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

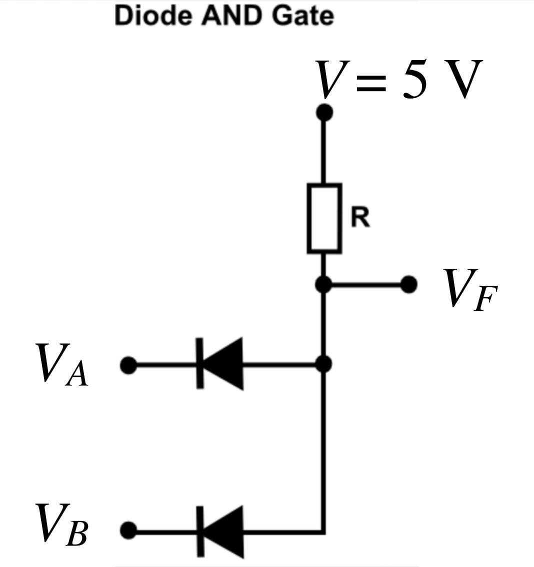
c) AND



$F{=}A{\cdot}B$



Input		Output
A	B	F = A.B
0	0	0
0	1	0
1	0	0
1	1	1



V_A	V_B	V_F
0	0	.6 V
0	5V	.6 V
5V	0	.6 V
5V	5V	5 V

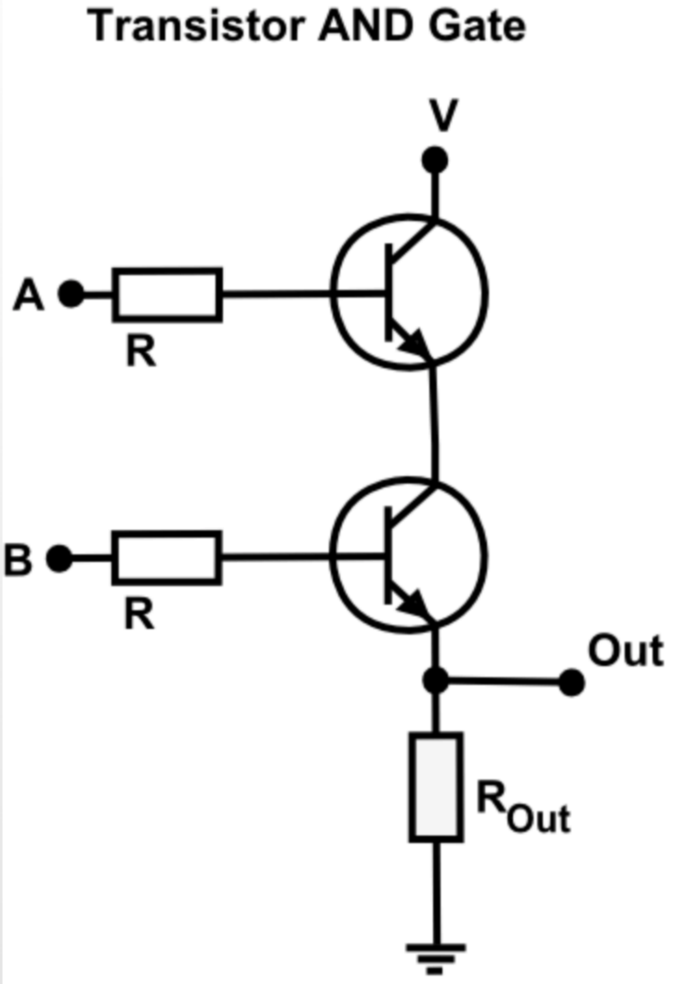
positive logic:

0 —> ~ 0V (<~ .6V)

1 —> ~5V

A low on either input turns on the respective diode, bringing the output to low.

If both inputs are high, the diodes are both off, so no current flows, and the output is high.

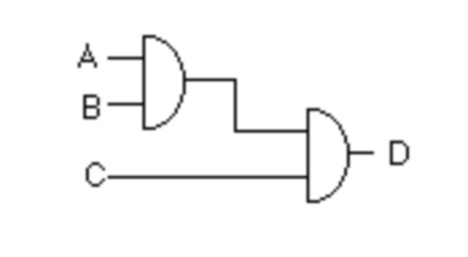


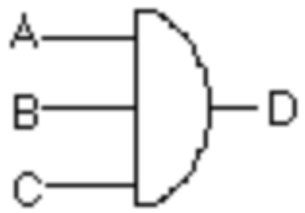
Transistor AND Gate

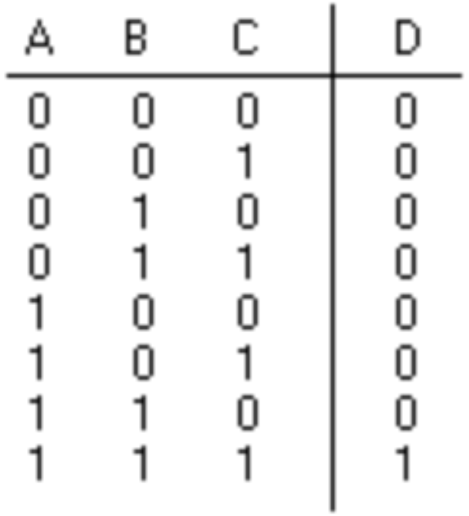
If both transistor turn on
(i.e. if both A and B are high)
then the output is tied to Vcc (high).

If either transistors is off
(i.e. if either A or B is low)
then there is no current through R,
so the output is low (~ zero).

extends easily to multiple inputs:

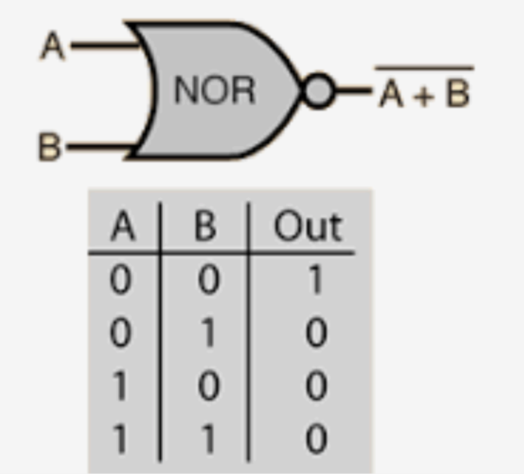




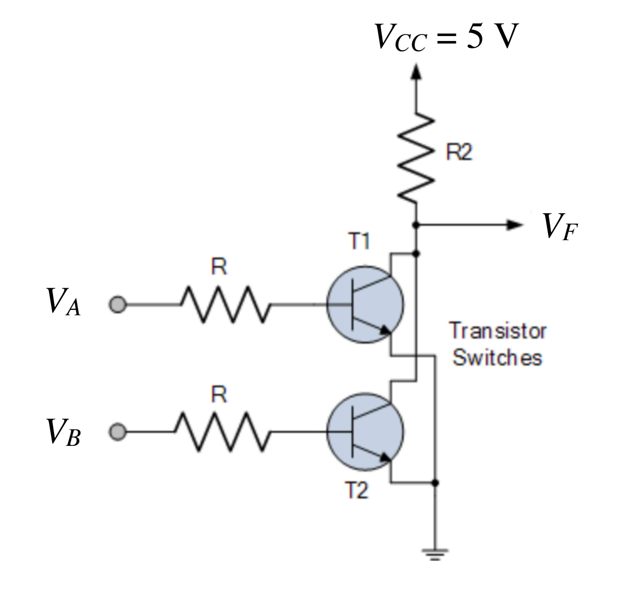


A	B	C	D
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

d) NOR (NOT OR)



A	B	Out
0	0	1
0	1	0
1	0	0
1	1	0



V_A	V_B	V_F
0	0	5 V
0	5V	0.1 V
5V	0	0.1 V
5V	5V	0.1V

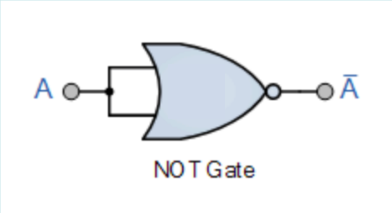
If either input is high, the respective transistor turns on, and the output is switched to ground (low).

If both inputs are low, both transistors turns off,
so no current flowa and
the output is high.

NOR gate as universal gate:

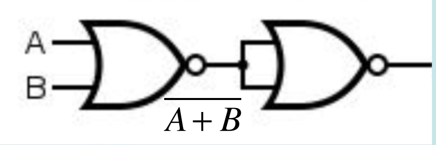
NOT

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



OR

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

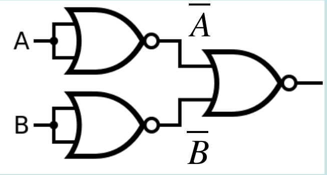


$A + B$

AND

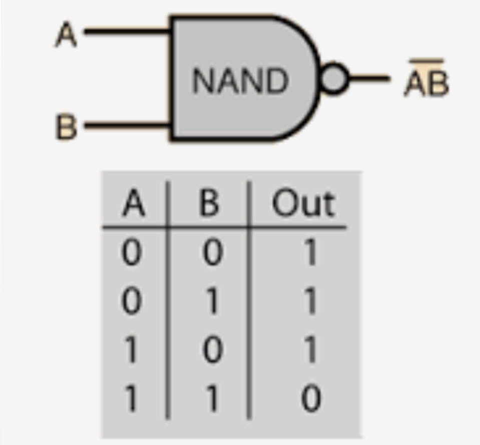
$$A \cdot B = \overline{\overline{A} + \overline{B}}$$

De Morgan

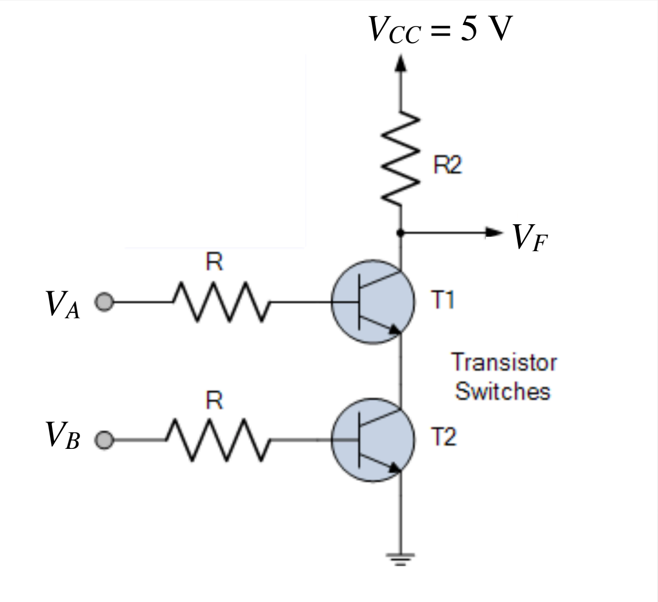


$\overline{\overline{A} + \overline{B}} = A \cdot B$

e) NAND (NOT AND)



A	B	Out
0	0	1
0	1	1
1	0	1
1	1	0



V_A	V_B	V_F
0	0	5 V
0	5V	5 V
5V	0	5 V
5V	5V	0.1V

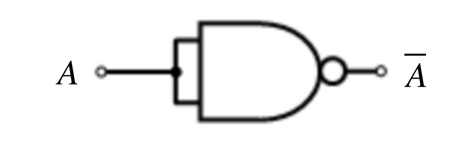
If either input is low, the respective transistor turns off, so no current flows, and the output is high.

If both inputs are high, both transistors turns on,
and the output is switched to ground (low).

NAND gate as universal gate:

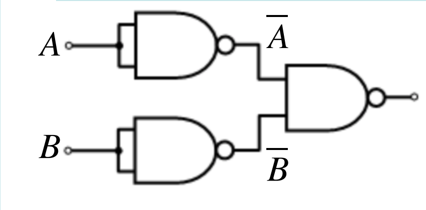
NOT

$$\overline{A} = \overline{A \cdot A}$$



OR

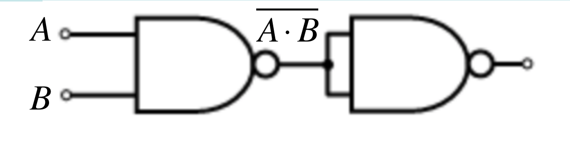
$$A + B = \overline{\overline{A} \cdot \overline{B}}$$



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

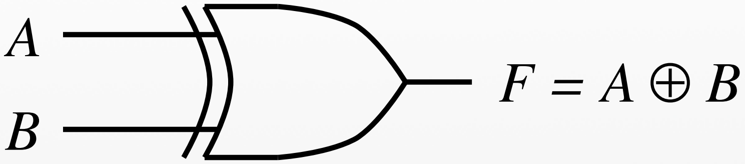
AND

$$A \cdot B = \overline{\overline{A \cdot B}}$$



$A \cdot B$

f) Exclusive OR (XOR)



A	B	$F = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

- either, but not both

$$A \oplus B = (A+B)\bullet(\overline{A \bullet B})$$

check:

A	B	$A+B$	$A \cdot B$	$\overline{A \bullet B}$	$(A+B)\bullet(\overline{A \bullet B})$
0	0	0	0	1	0
0	1	1	0	1	1
1	0	1	0	1	1
1	1	1	1	0	0

$$A \oplus B = (A+B)\bullet(\overline{A \bullet B})$$

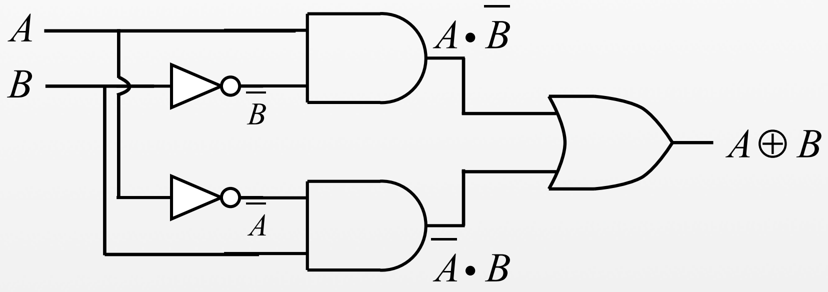
$$= A \bullet \overline{A \bullet B} + B \bullet \overline{A \bullet B} \qquad \text{dist.}$$

$$= A \bullet (\overline{A} + \overline{B}) + B \bullet (\overline{A} + \overline{B}) \qquad \text{De Morgan}$$

$$= 0 + A \bullet \overline{B} + B \bullet \overline{A} + 0 \qquad A \bullet \overline{A} = 0$$

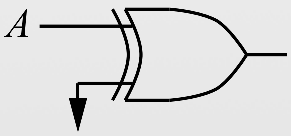
$$\boxed{A \oplus B = A \bullet \overline{B} + B \bullet \overline{A}}$$

$$A \oplus B = A \bullet \overline{B} + B \bullet \overline{A}$$



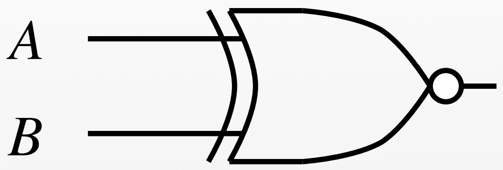
- can be constructed with all NOR or all NAND gates.

Controllable inverter:



Control: high —> inverter
low —> leave as is

g) Exclusive NOR



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

A	B	$A \oplus B$	$\overline{A \oplus B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

$$\overline{A \oplus B} = \overline{A \bullet \overline{B} + \overline{A} \bullet B}$$

$$= \overline{A \bullet \overline{B}} \bullet \overline{\overline{A} \bullet B}$$

De Morgan

$$= (\overline{A} + B) \bullet (A + \overline{B})$$

De Morgan

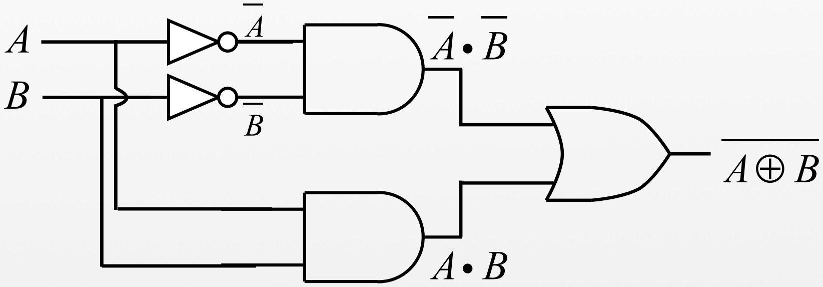
$$= \overline{A} \bullet A + \overline{A} \bullet \overline{B} + B \bullet A + B \bullet \overline{B}$$

dist.

$$\boxed{\overline{A \oplus B} = \overline{A} \bullet \overline{B} + A \bullet B}$$

$A \bullet \overline{A} = 0$

$$\overline{A \oplus B} = \overline{A} \bullet \overline{B} + A \bullet B$$



6) TTL Logic

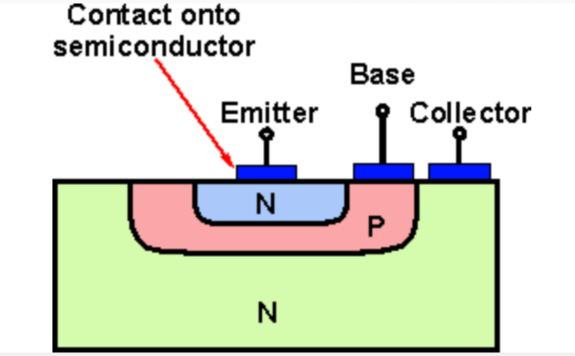
Transistor - transistor logic

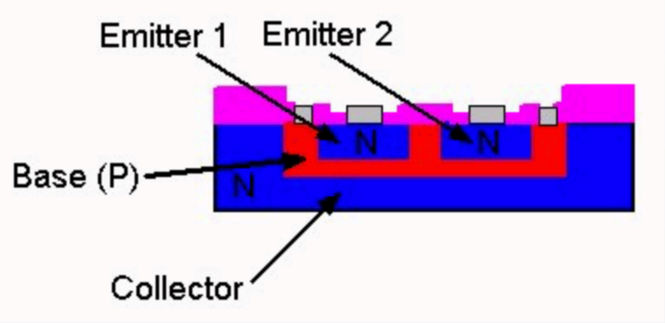
TTL: transistor used for logic and amplification

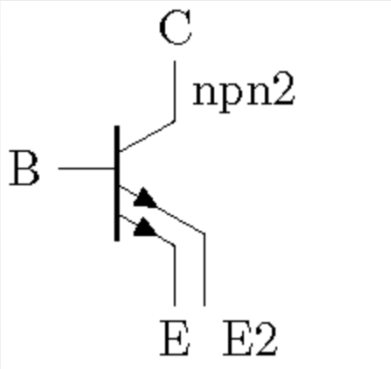
Others:

DL - diode logic
RTL - resistor transistor logic
DTL - diode transistor logic)

a) two emitter transistor



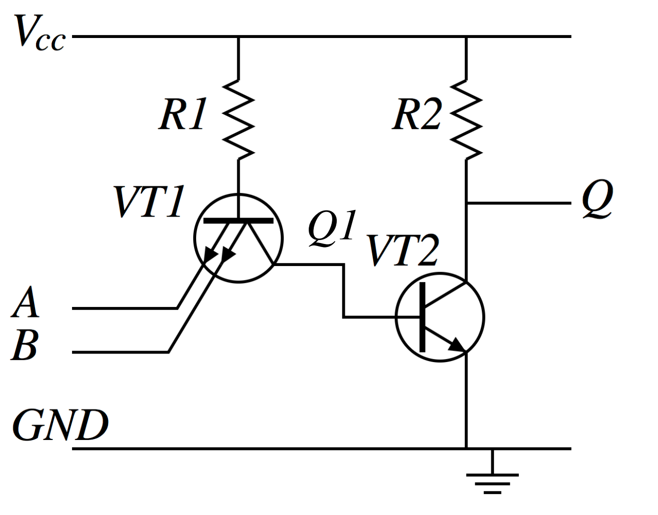




If either b-e junction is on:

- minority carriers injected into base
- transistor on
- V_{CE} low

b) basic TTL NAND gate



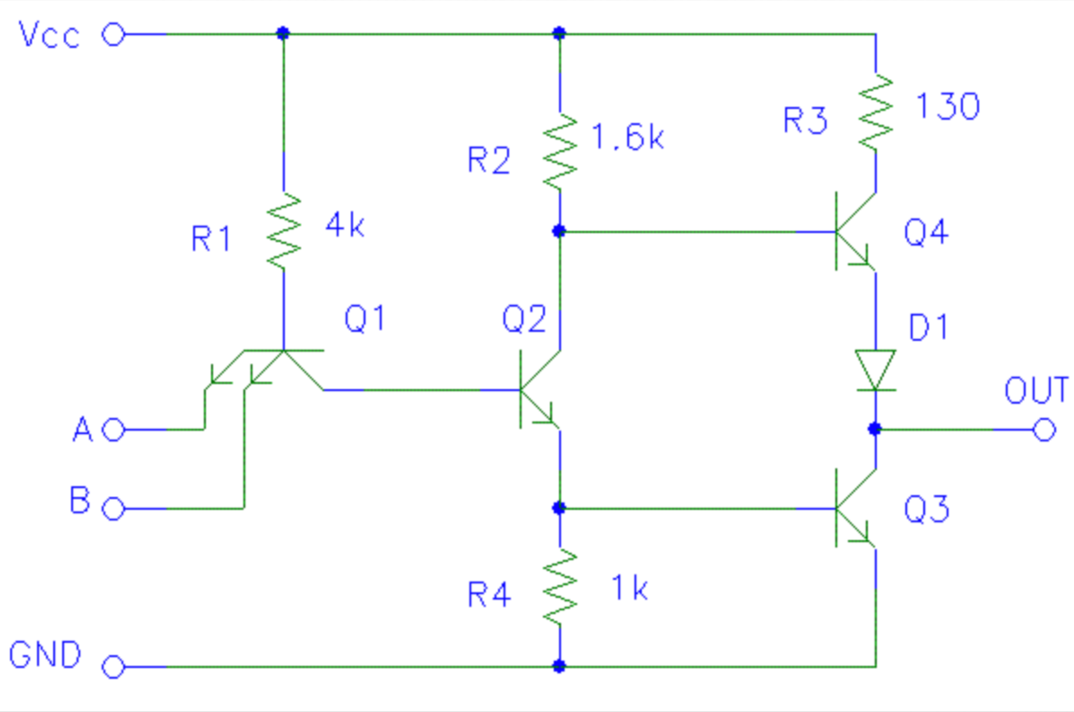
- If A and B are high VT1 is off
  - Then Q1 is high
- Otherwise VT1 is on
  - and Q1 is low
- VT2 is the familiar NOT gate

A	B	$Q1=A\cdot B$	$Q=\overline{A\cdot B}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

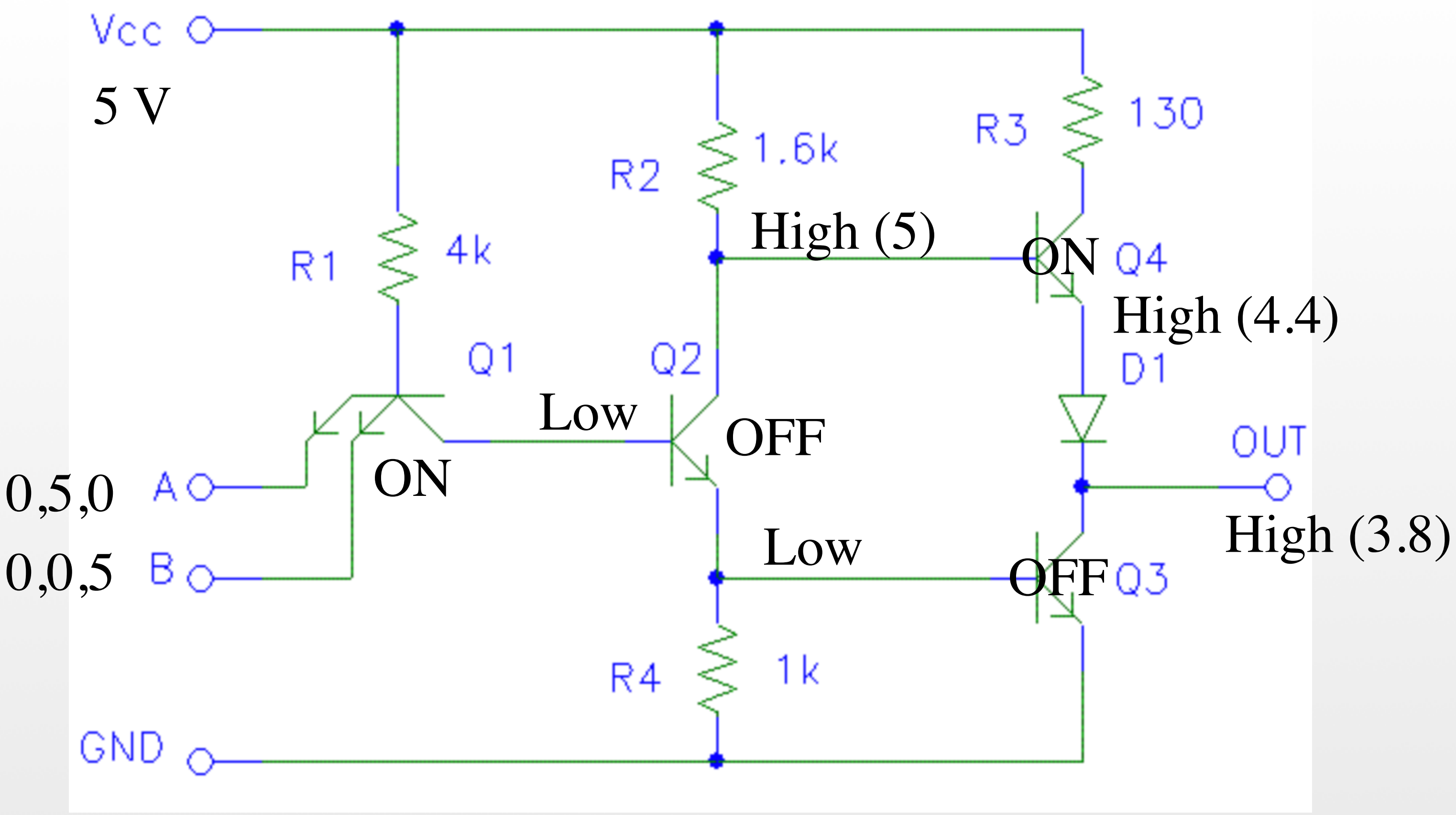
High output impedance (R2):

- slow to turn off (go high)
- output current limited
- reducing R2 consumes power

c) TTL NAND gate with totem pole output stage

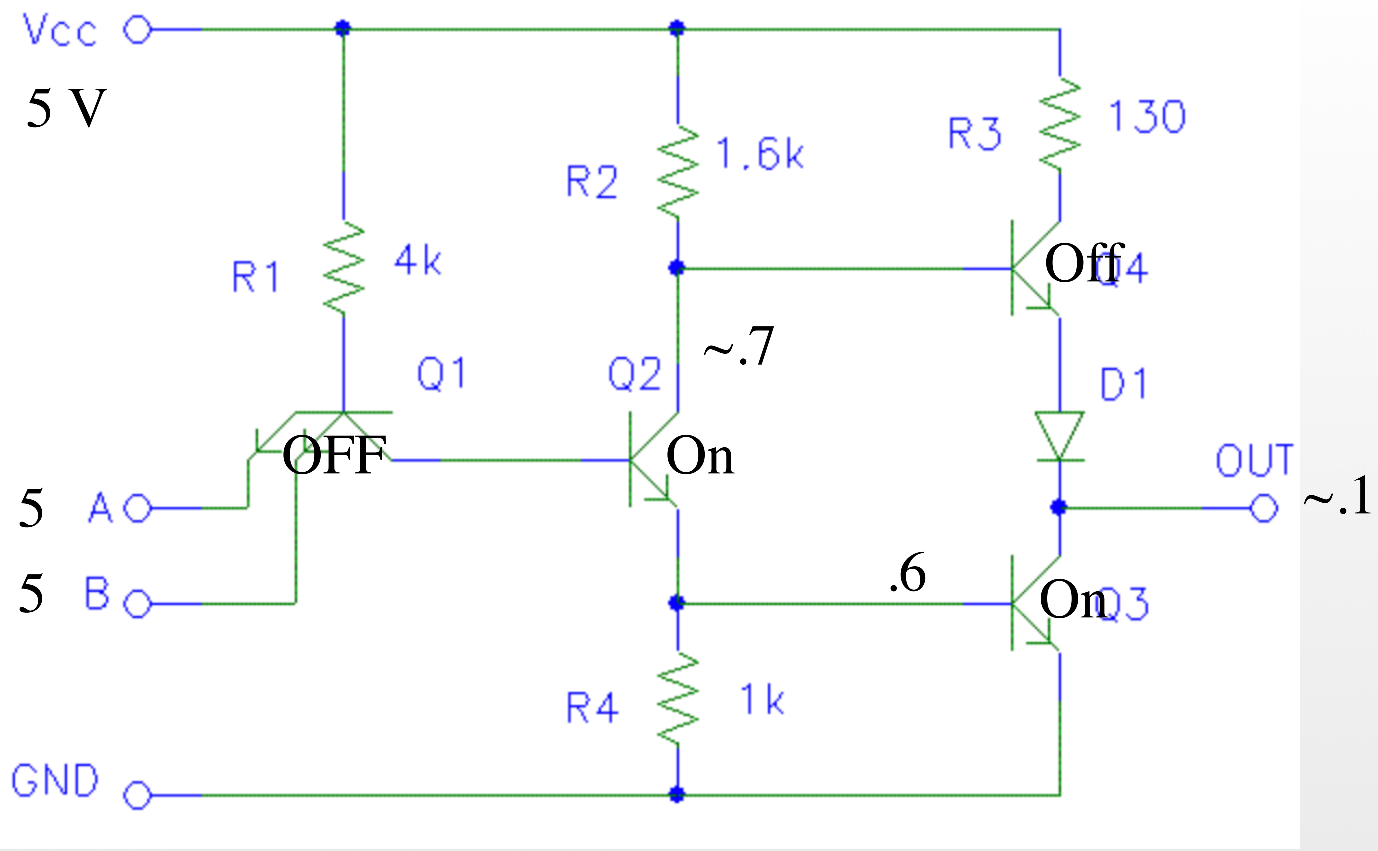


- Output signal amplified for more speed
- Output load capacitance is charged and discharged through active transistors instead of resistors



Q2 off —> Vc2 high —> Q4 on —> Ve4 high (through R3; faster) —> OUT high

—>Vb3 low —> Q3 off —> leaves OUT high



Q2 on —> Q3 on —> Vc3 ~ .1 V

—>Vb3 = .6 V —> Vc2 ~.7V —> Q4 off

(because of diode)

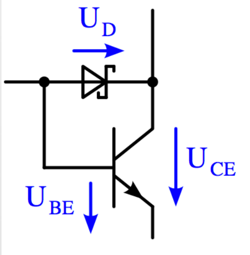
d) TTL logic

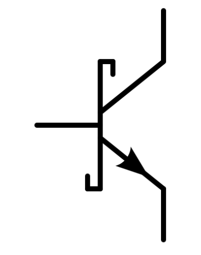
logic 0: < 0.8 V; ideally 0; typically 0.1

logic 1: > 2.0 V; ideally 5; typically 3.6

regular TTL: 7400

Schottky TTL: 74S00





Schottky diode: semiconductor - metal

- fast, lower power

low power Schottky: 74LS00

7) *Digital Addition*

a) Binary addition

```
 1 111 1
 110011010
 010111011
 ---------
1001010101
```

Define addition for
2 1-bit variables:

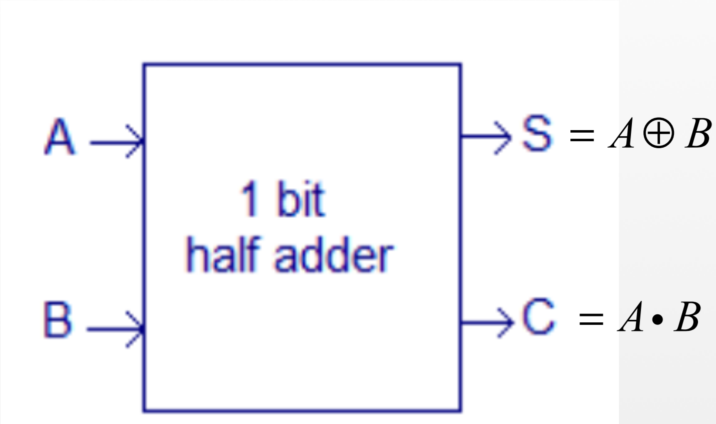
A	B	Sum $A\oplus B$	Carry $A\bullet B$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

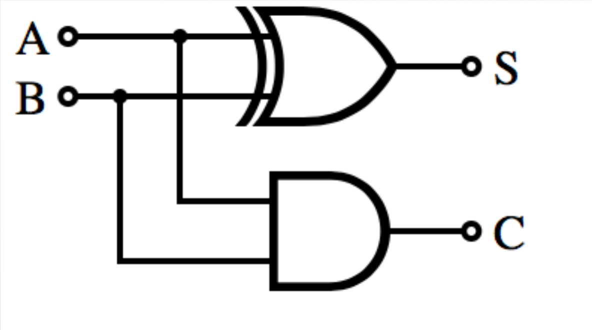
Define addition for
3 1-bit variables:

A	B	C	S_{AB} $A\oplus B$	Carry 1 $A\bullet B$	S_{ABC} = sum(S_{AB},C) $A\oplus B\oplus C$	Carry 2 $(A\oplus B)\bullet C$	Carry $(A\oplus B)\bullet C+A\bullet B$
0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	1	0	0	1	1
1	0	0	1	0	1	0	0
1	0	1	1	0	0	1	1
1	1	0	0	1	0	0	1
1	1	1	0	1	1	0	1

b) Half adder add 2 1-bit variables

A	B	Sum $A \oplus B$	Carry $A \bullet B$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

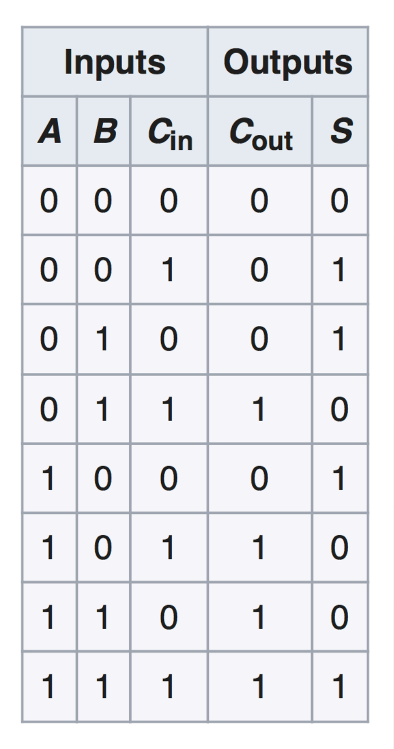




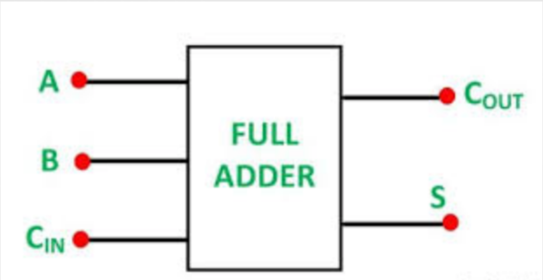
c) 1-bit Full adder

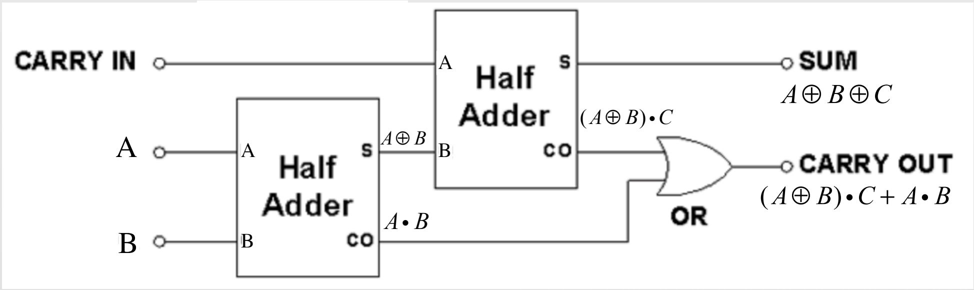
Adds column with carry from previous column

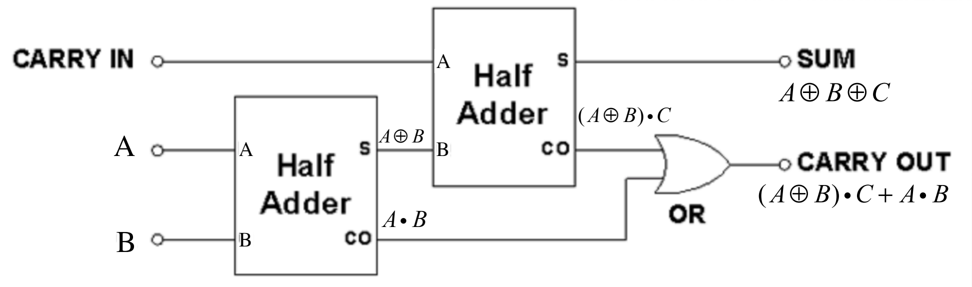
A	B	C	S_{AB} $A\oplus B$	Carry 1 $A\bullet B$	S_{ABC} = sum(S_{AB},C) $A\oplus B\oplus C$	Carry 2 $(A\oplus B)\bullet C$	Carry $(A\oplus B)\bullet C + A\bullet B$
0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	1	0	0	1	1
1	0	0	1	0	1	0	0
1	0	1	1	0	0	1	1
1	1	0	0	1	0	0	1
1	1	1	0	1	1	0	1

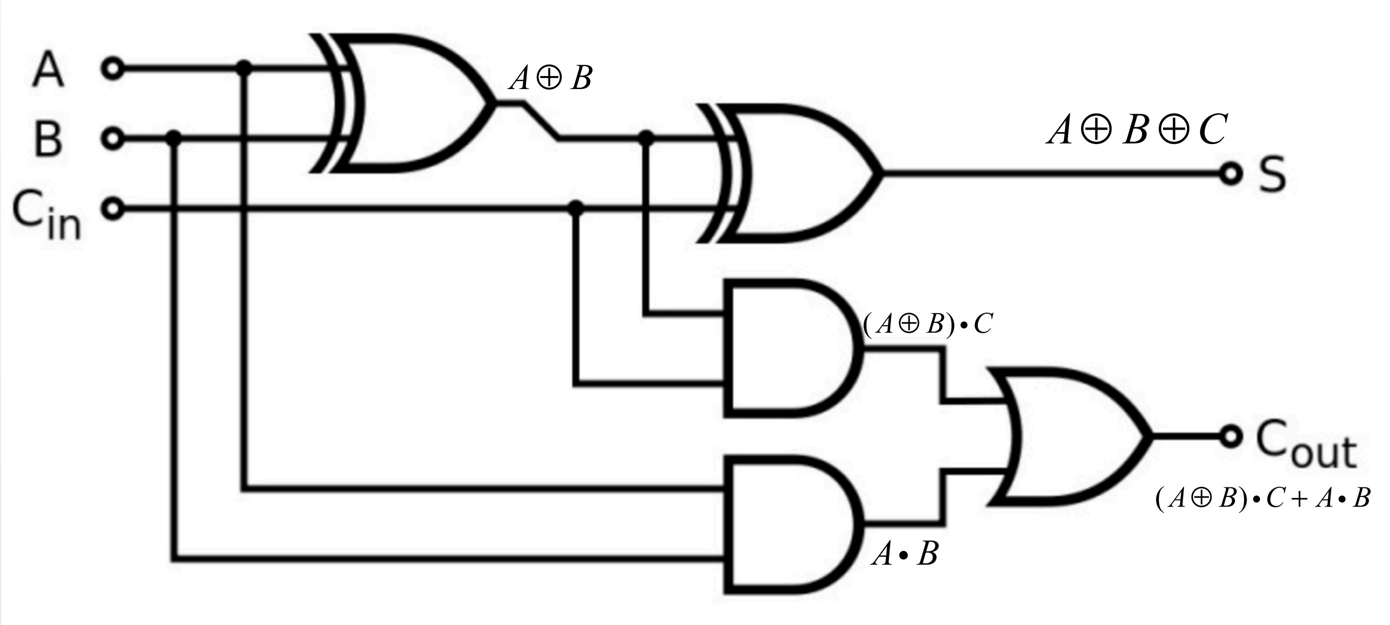


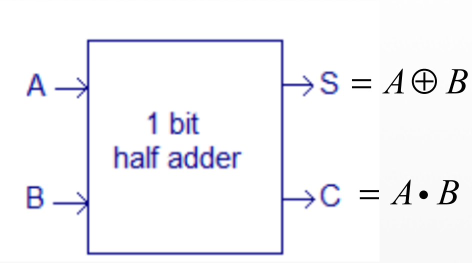
Inputs			Outputs	
A	***B***	***C_{in}***	***C_{out}***	***S***
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

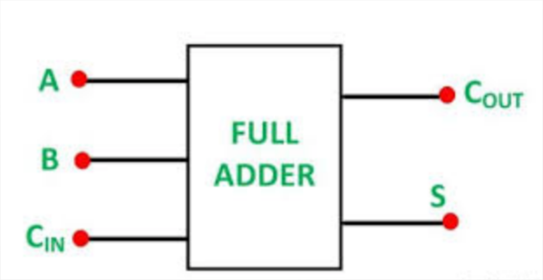


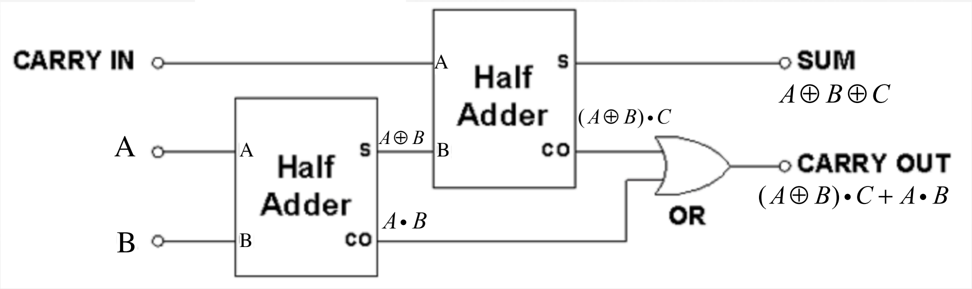


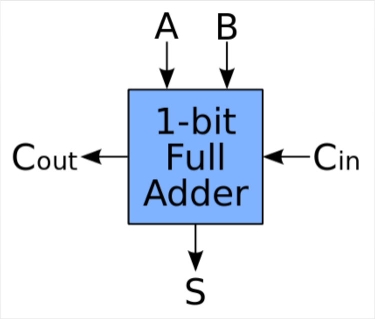

CARRY IN
A
B
A
B
Half Adder
S
CO
$A \oplus B$
$A \bullet B$
A
B
Half Adder
S
CO
$(A \oplus B) \bullet C$
OR
SUM
$A \oplus B \oplus C$
CARRY OUT
$(A \oplus B) \bullet C + A \bullet B$


A
B
C_{in}
$A \oplus B$
$A \oplus B \oplus C$
S
$(A \oplus B) \bullet C$
$A \bullet B$
C_{out}
$(A \oplus B) \bullet C + A \bullet B$


A
B
1 bit
half adder
S $= A \oplus B$
C $= A \bullet B$

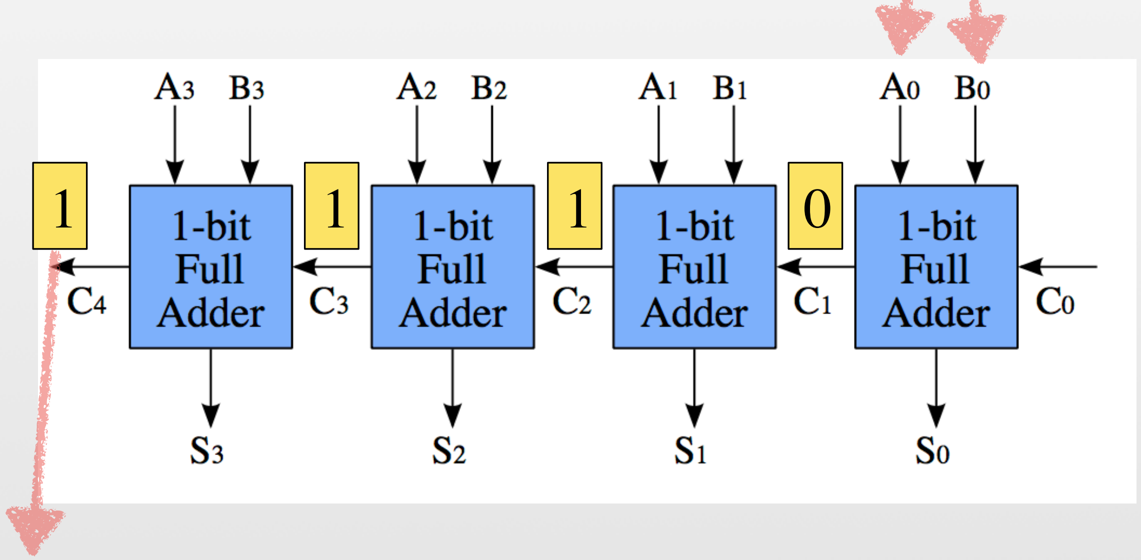

A
B
C_{IN}
FULL
ADDER
C_{OUT}
S


CARRY IN
A
B
A
B
Half
Adder
S
CO
$A \oplus B$
$A \bullet B$
A
B
Half
Adder
S
CO
$(A \oplus B) \bullet C$
OR
SUM
$A \oplus B \oplus C$
CARRY OUT
$(A \oplus B) \bullet C + A \bullet B$


A
B
Cout
1-bit
Full
Adder
Cin
S

d) Parallel Addition

A		1	0	1	1
B		1	1	1	0



SUM	1	1	0	0	1

8) Digital Subtraction

Two possible methods:

1) 1/2 subtractor and full subtractor (with borrow)

√ 2) using 2's complement

a) Ones' complement: each bit is complemented

e.g. $X = 1011 \longrightarrow X_1 = 0100$

Note: $X + X_1 = 1111$

b) Two's complement: add 1 to ones' complement

procedure: *leave all right bits as is up to and including first 1, and complement the rest*

$X_2 = X_1 + 1$ *e.g.* $X = 1011 \longrightarrow X_1 = 0100 \longrightarrow X_2 = 0101$

Since $X + X_1 = 1111$, $X + \underbrace{X_1 + 1}_{X_2} = 10000 = 10^4$ (or 2^4 in decimal)

So, $\boxed{X^2 = 2^N - X}$ where N is the number of bits

c) Two's complement as negative

Recall $X_2 = 2^N - X$

—> $X_2 + X = 2^N$

—> $X_2 + X = 0$ *(ignoring the carry, or considering only N bits)*

—> $X_2 = -X$ *or* $X = -X_2$

So, $Y - X = Y + (-X) = Y + X_2$

i.e. subtract X by adding X_2

(and ignore the carry)

Problem: if $X > Y$, how to recognize negative value?

d) Subtraction using 2’s complement

$X_2 = 2^N - X$ —> $X = 2^N - X_2$

$$Y - X = Y - (2^N - X_2)$$

$$\boxed{= (Y + X_2) - 2^N} \quad = \quad -[2^N - (Y + X_2)] \quad \boxed{= -(Y + X_2)_2 \ ??}$$

but 2^N has no representation in N bits,
and Z_2 is not defined if $Z > 2^N$

$Y - X$ $= (Y + X_2) - 2^N$ $= -(Y + X_2)_2$??

but 2^N has no representation in N bits, and Z_2 is not defined if $Z > 2^N$

Consider 3-bit numbers (N = 3)

e.g. $Y = 3 = 011;\ X = 2 = 010;\ X_2 = 110\ (=6)$

$Y - X = (Y + X_2) - 2^N$

3 - 2 = (3 + 6) - 8 = 1

$$\left.\begin{array}{r} 011 - 010 = 011 \\ \underline{110} \\ (1)001 \end{array}\right\} - 1000 = 1$$

For every $Y > X$, $Y + X_2 > 2^N$ so the carry is 1

—> If Y > X (or if the carry is 1), $Y - X = Y + X_2$ to 3 bits

(In this case, $(Y + X_2)_2$ is not defined)

$Y - X$ $= (Y + X_2) - 2^N$ $= -(Y + X_2)_2$??

but 2^N has no representation in N bits,
and Z_2 is not defined if $Z > 2^N$

Consider 3-bit numbers (N = 3)

if the carry is 1 $Y - X = Y + X_2$ to 3 bits

e.g. $Y = 2 = 010;\ X = 3 = 011;\ X_2 = 101\ (=5)$

$Y - X = (Y + X_2) - 2^N$

2 - 3 = (2 + 5) - 8 = -1

$$\left.\begin{array}{r} 010 - 011 = 010 \\ \underline{101} \\ 111 \end{array}\right\} - 1000 = -1$$

$Y - X = -(Y + X_2)_2$??

$010 - 011 = -(111)_2 = -(001)$

For every $Y < X$, $Y + X_2 < 2^N$ —> carry is 0

—> If $Y < X$ (or if the carry is 0), $Y - X = -(Y + X_2)_2$

$Y - X \quad = Y + X_2$ if the carry is 1

$\quad\quad\quad\;\; = -(Y + X_2)_2$ if the carry is 0

e) Positive and Negative representation

4-bit —> $2^4 = 16$ numbers 0 to 15 (regular binary representation)

-8 to +7

-7 to +8

Options: 1) sign, magnitude: low 3 bits for magnitude, high bit for sign

—> subtraction has a conditional as above

2) 2’s complement for negative

- high bit for sign (1 negative)
- positive: low 3 bits in binary
- negative: 4-bit 2’s complement of absolute value

2) 2’s complement for negative

- high bit for sign (1 negative)
- positive: low 3 bits in binary
- negative: 4-bit 2’s complement of absolute value

x	X	sign-magnitude
7	0111	0111
6	0110	0110
5	0101	0101
4	0100	0100
3	0011	0011
2	0010	0010
1	0001	0001
0	0000	0000=1000
-1	1111	1001
-2	1110	1010
-3	1101	1011
-4	1100	1100
-5	1011	1101
-6	1010	1110
-7	1001	1111
-8	1000	

If $X = A_3A_2A_1A_0$ is a 4-bit representation of x, then

$$x = \begin{matrix} X_b \text{ if } A_3 = 0 \\ -X_2 \text{ if } A_3 = 1 \end{matrix}$$

Then,

$$y - x = Y + X_2 \qquad \text{if } y > x$$

$$y - x = -(Y + X_2)_2 \qquad \text{if } y < x$$

but $-(Y + X_2)_2 = Y + X_2$

so $\boxed{y - x = Y + X_2}$ without conditionals

x	X	sign-magnitude
7	0111	0111
6	0110	0110
5	0101	0101
4	0100	0100
3	0011	0011
2	0010	0010
1	0001	0001
0	0000	0000=1000
-1	1111	1001
-2	1110	1010
-3	1101	1011
-4	1100	1100
-5	1011	1101
-6	1010	1110
-7	1001	1111
-8	1000	

$$\boxed{y - x = Y + X_2}$$

3 - 2 = 1

$$\left.\begin{array}{r} 0011 - 0010 = 0011 \\ 1110 \\ \hline (1)0001 \end{array}\right\} = 1$$

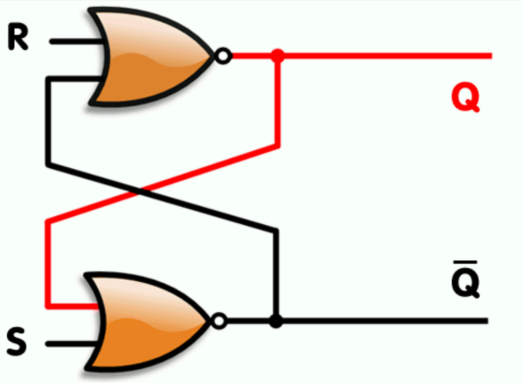
2 - 3 = -1

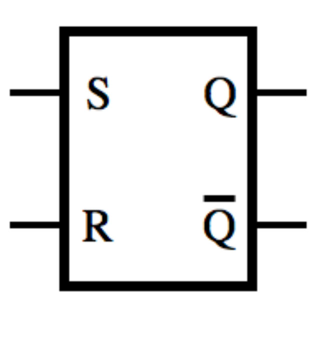
$$\left.\begin{array}{r} 0010 - 0011 = 0010 \\ 1101 \\ \hline 1111 \end{array}\right\} = -1$$

9) Flip Flops

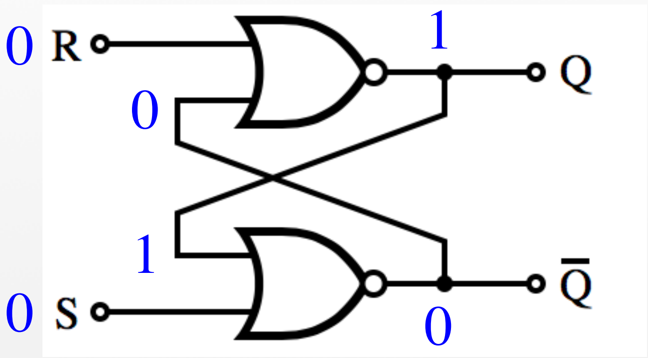
- 2 stable states for storing binary information

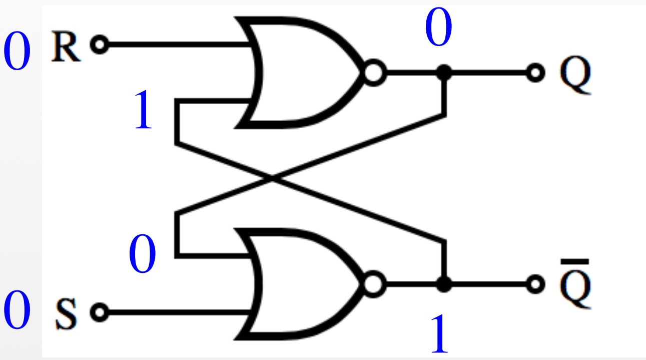
a) SR flip flop (NOR), S-set; R-reset (SR latch)





R = S = 0

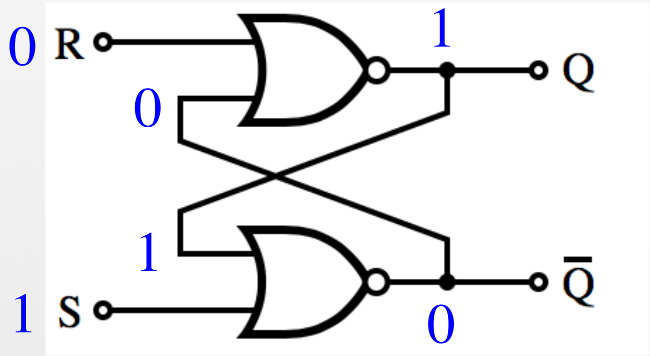




2 possible states with R = S = 0

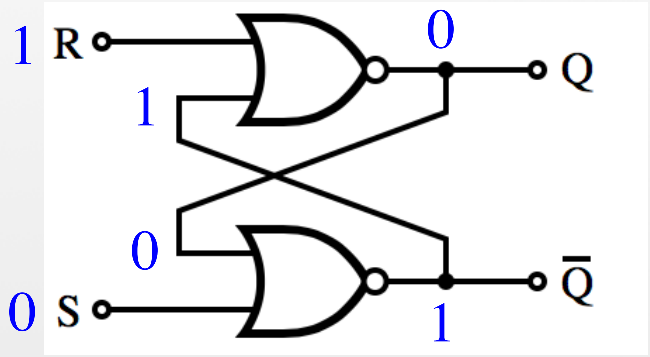
the state is “read” with R = S = 0

$S = 1, R = 0$



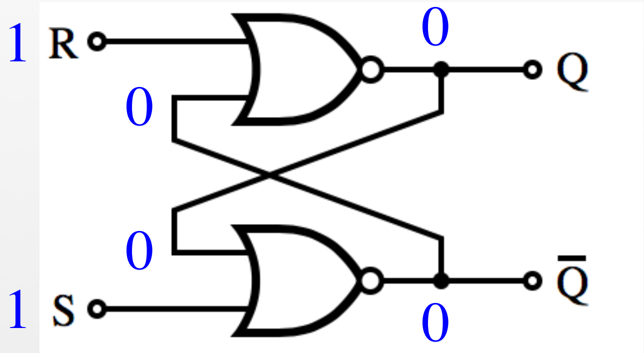
sets $Q = 1, \overline{Q} = 0$

S = 0, R = 1



(re)sets Q = 0, $\overline{\text{Q}}$ = 1

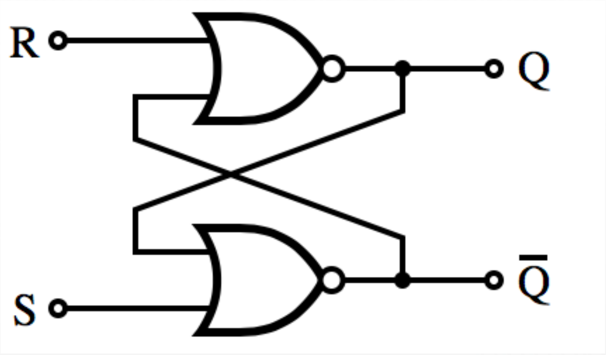
S = 1, R = 1



both outputs 0

indeterminate when R, S set to 0

not allowed (not used)



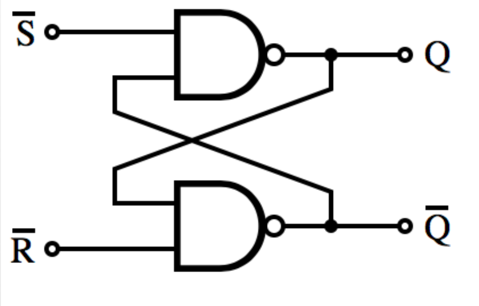
Truth table

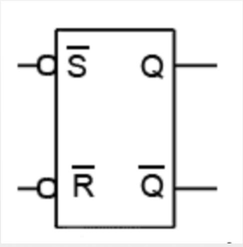
S	R	Q	$\overline{Q}$
0	0	NC	NC
1	0	1	0
0	1	0	1
1	1	X	X

no change
(data storage)

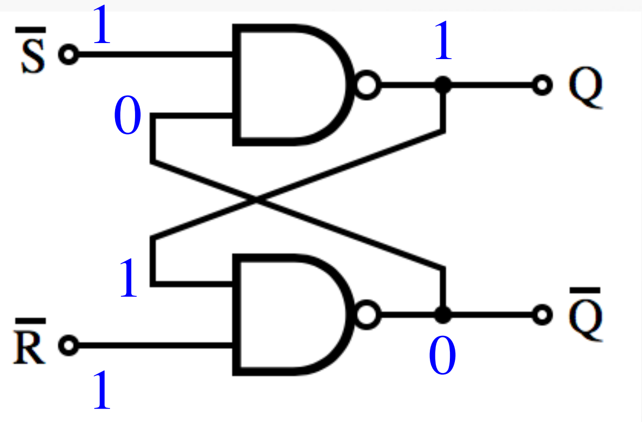
not allowed;
indeterminate when
read with 0,0

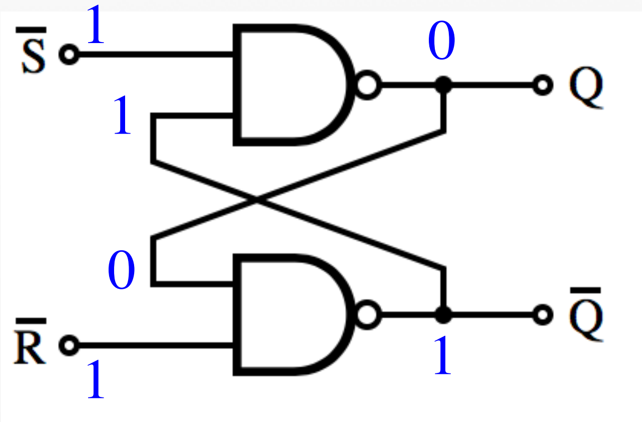
b) SR NAND latch





$R = S = 0; \overline{R} = \overline{S} = 1$

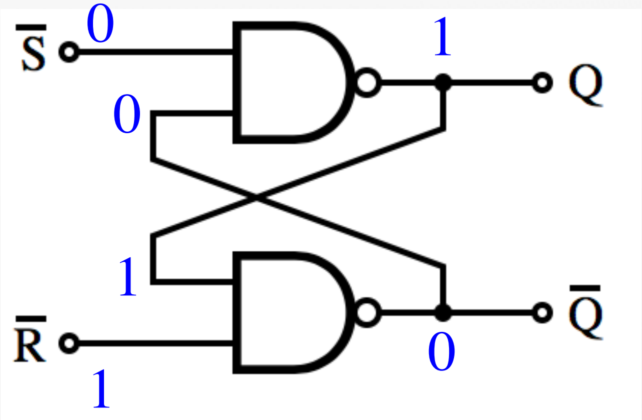




2 possible states with R = S = 0

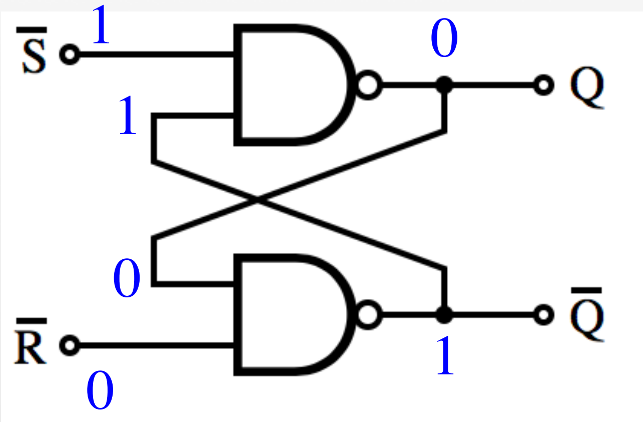
the state is “read” with R = S = 0

$$S = 1,\ R = 0;\quad \overline{S} = 0,\ \overline{R} = 1$$



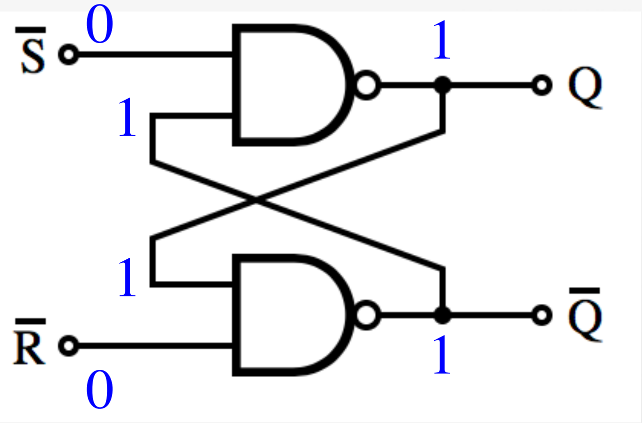
$$\text{sets } Q = 1, \overline{Q} = 0$$

$S = 0,\ R = 1;\ \ \overline{S} = 1,\ \overline{R} = 0$



(re)sets $Q = 0, \overline{Q} = 1$

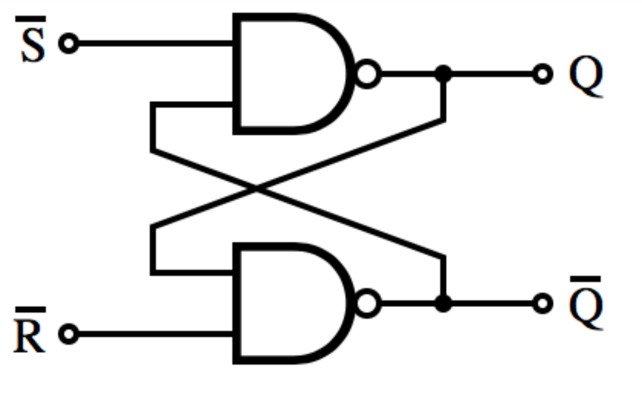
$$S = 1,\ R = 1;\ \ \overline{S} = 0,\ \overline{R} = 0$$



both outputs 1

indeterminate when R, S set to 0

not allowed

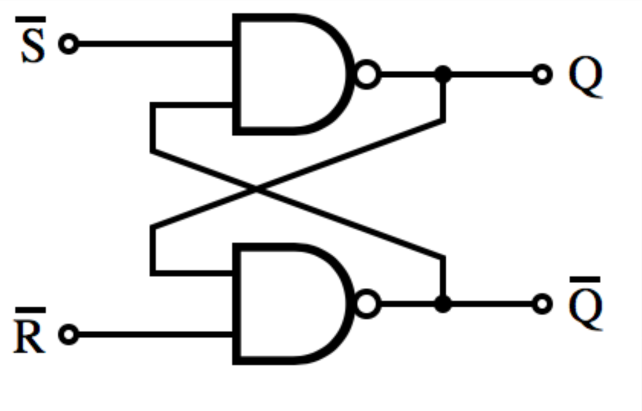


Truth table

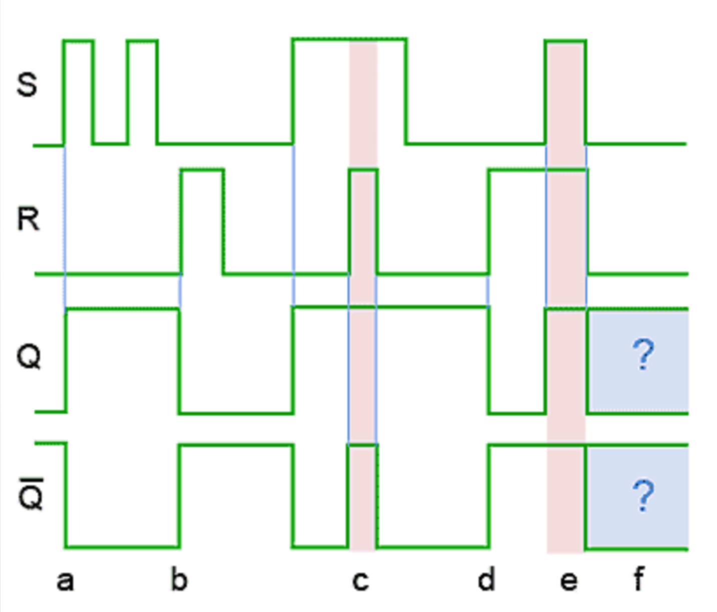
S	R	$\overline{S}$	$\overline{R}$	Q	$\overline{Q}$
0	0	1	1	NC	NC
1	0	0	1	1	0
0	1	1	0	0	1
1	1	0	0	X	X

no change
(data storage)

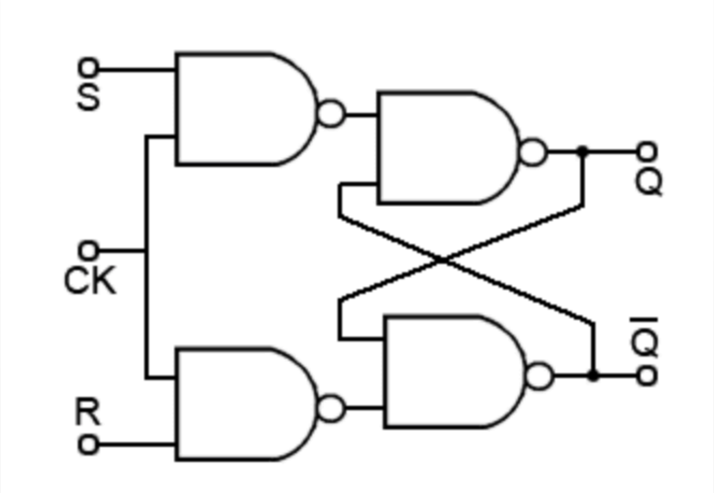
not allowed;
indeterminate when
read with 0,0

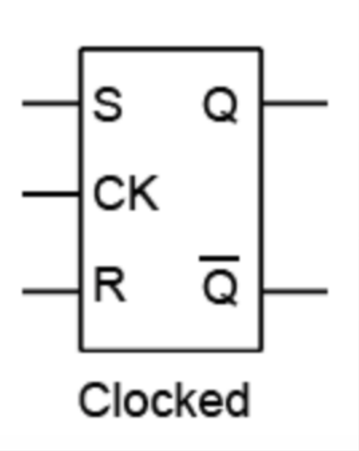


S	R	$\overline{S}$	$\overline{R}$	Q	$\overline{Q}$
0	0	1	1	NC	NC
1	0	0	1	1	0
0	1	1	0	0	1
1	1	0	0	X	X

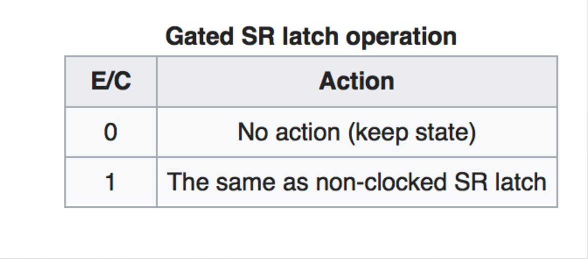


c) Clocked (or gated) SR latch



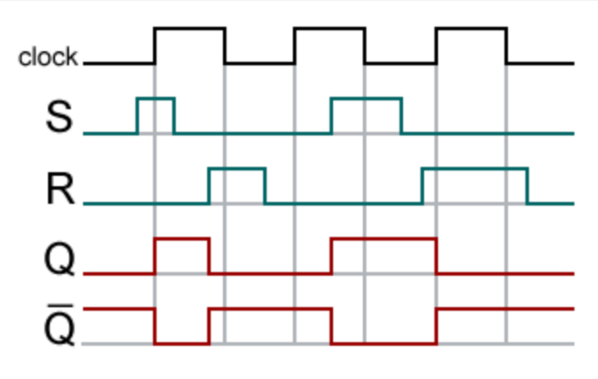


Clock must be present (high) to enable R and S

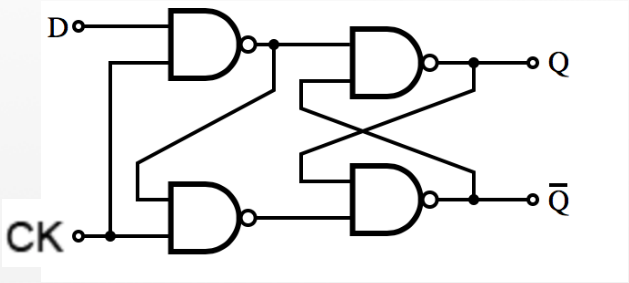


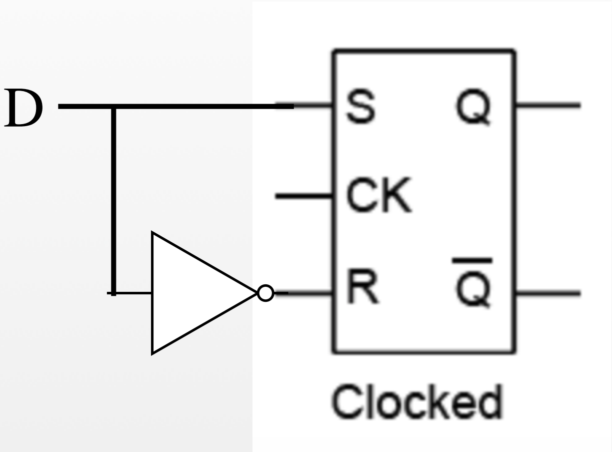
Gated SR latch operation

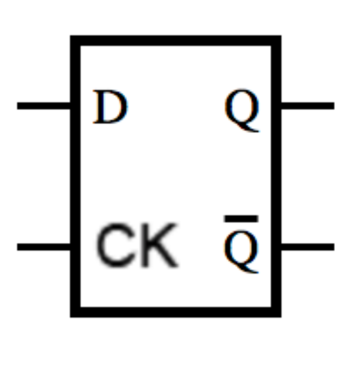
E/C	Action
0	No action (keep state)
1	The same as non-clocked SR latch



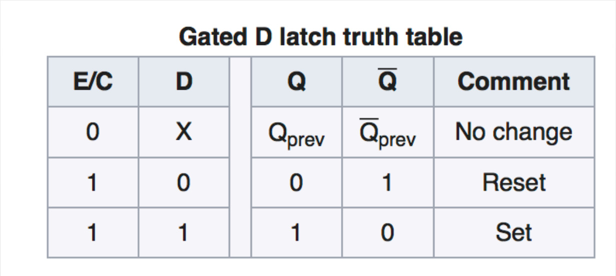
d) D flip-flop (data latch)







Stores state of D at last clock high.



Gated D latch truth table

E/C	D	Q	$\overline{Q}$	Comment
0	X	Q_{prev}	$\overline{Q}_{prev}$	No change
1	0	0	1	Reset
1	1	1	0	Set