Chapter 1 Basic Logic Gate Practice

1-1 Practice purpose

1. To understand the usage of logic gate
2. To understand the conversion and appliance of logic gate

1-2 Practice theory

This chapter is mainly introducing how to use logic gate. Logic gate is the most basic element in the digitalal system. Basic logic gate includes AND, OR, NAND, NOR XOR and Exclusive OR.

Digitalal logic circuit is a binary system of IN and OUT voltage respectively 1 and 0. We know that 0 and 1 respectively have specific voltage range and the characteristic of logic circuit allows us to use Boolean algebra as a tool to analyze and design the digital system.

In this chapter, we are going to discuss: (1) how to explain logic calculation with Boolean Algebra; (2) how to combine logic gate in order to produce logic circuit; (3) how to explain and analyze logic circuit with Boolean Algebra and also explain how the Karnough map and table method simplify logic circuit so that our readers have a clear idea about CPLD digital experiments.

1-2-1 Logic function

The difference between Boolean Algebra and normal algebra is that Boolean Algebra only allows two possible numerical values which means 0 and 1. Boolean Algebra can be 0 and 1 at different times. Boolean Algebra is always presented as input/output voltages. For example, when 0's value is between 0V and 0.8V in some digital system and 1 is between 2V and 5V. The voltage between 0.8V and 2V is not defined (neither 0 or 1) and it does not happen under normal circumstances.

Therefore, Boolean's 0 and 1 do not present any actual number but only the status of different voltages or so called logic level.

The voltage in digital circuit is called logic level 0 and logic level 1 which depends on the numerical value. For digital logic field, technical terms have the same meaning as 0 and 1. Table 1-1 shows the common ones in which we always use 0/1 and low/high as technical terms.

Table 1-1

Logic 0	Logic 1
False	true
close	open
low	high
no	yes
switch on	switch off

Boolean Algebra expresses the logic input effects at different digital circuits and it deals with logic variables so that it can decide the best way to execute the circuit function. The following topic is about how to use an alphabet symbol to present logic variables. For example, A can mean some digital circuit's input or output and also at any time A equals either 0 or 1.

As there are only two possible numbers for Boolean Algebra, therefore it is easier to operate than normal algebra; there are no fractions, denary scale, negative numbers, square roots, cube roots, logarithms, or imaginary numbers. In fact, there are only three basic operations:

(1) Logic addition: Also called OR addition or OR operation. OR operation symbols are: f (A,B) = A+B = A∪B = A∨B

This book applies the first symbol A+B, when dealing with OR function, avoids using PLUS.

(2) Logic multiplication: Also called AND multiplication or AND operation, symbols are: f (A, B) = A·B = AB = A∩B = A∧B

AND symbols are A·B or AB

(3) Logic complementary or inversion: Also called NOT operation, symbols are: f (A)= $\overline{A} = A' = A^*$

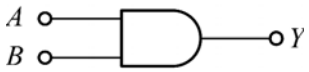
NOT operation symbols are $^-$, $\overline{A}$ *which is read as A BAR.*

1-2-2 Logic Gate

Basically, logic operations can be present from various logic circuits and these logic circuits are AND, OR, NOT, NAND, NOR, XOR, and XNOR.

1. AND Gate

As figure 1-1, there are symbols and truth table of two input variables AND gate, figure 1-2 is the symbols and truth table of input variable AND gate. Therefore, we can tell that when AND gate truth table shows 1 as all inputs, the outputs are 1 as well. Y=A·B.



A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Figure 1-1

A	B	C	Y
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

Figure 1-2

2. OR Gate:

When any of A or B input variables respectively connect to OR gate output, the output ends present as variable Y, then the relation among A, B, and Y:

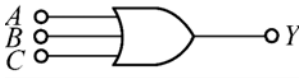
$$Y=A+B$$

As figure 1-3 shows the symbols and truth table of two input variables, figure 1-4 shows the symbols and truth table of three input variables. Therefore, we can tell that the truth table of OR gate is when any of the input variables present as 1, the output is 1 as well.

A	B	Y

0	0	0
0	1	1
1	0	1
1	1	1

Figure 1-3

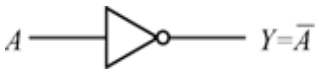


A	*B*	*C*	*Y*
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Figure 1-4

3. NOT Gate

NOT is also called inverter, NOT only supplement-operates one variable. The symbols and truth table are as figure 1-5.



A	*Y*
0	1
1	0

Figure 1-5

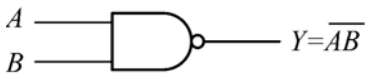
4.NAND Gate

NAND gate is combined with AND and NOT. Its name is NAND meaning NOT-AND.

NAND operation is: $f(A \cdot B) = (A \text{ NAND } B) = \overline{AB}$

Logic symbols and value figure as figure 1-6

Input	AND	NAND



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

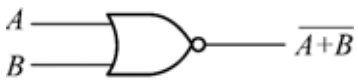
Figure 1-6

5. NOR Gate:

The combination of NOR and OR is the reason for its name NOR. NOR is capable of operating one or more Boolean Algebra.

NOR operation is: $f(A, B) = (A \text{ NOR } B) = \overline{A+B}$

Its truth table is:



Output	OR	NOR
A B	$A+B$	$\overline{A+B}$
0 0	0	1
0 1	1	0
1 0	1	0
1 1	1	0

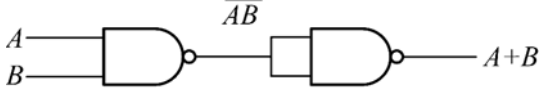
Figure 1-7

6. The basic logic gate combination of NAND and NOR:

NAND and NOR can be replaced with three basic logic gates by simple circuit, AND, OR and INVERTOR. In fact, it will be found that using the logic circuit combined with NAND and NOR is a simple and low cost method. That is why we call these two logic gates “Universal Logic Gate”, which means they can make up other logic gates.

Now let us have a look at how it works:

NAND to AND



NOR to AND

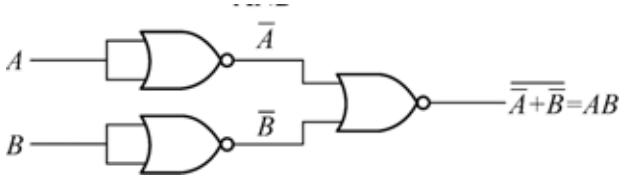
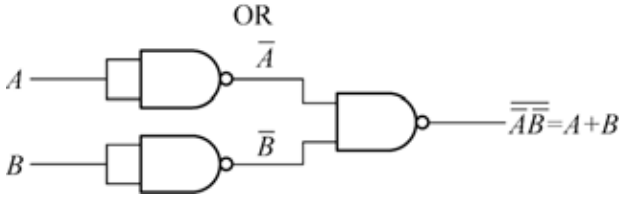


Figure 1-8 NAND and NOR make AND gate

NAND to OR



OR

NOR to OR

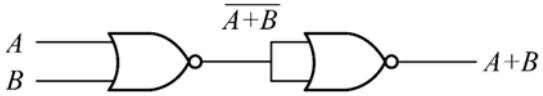


Figure 1-9 AND and NOR make OR gate

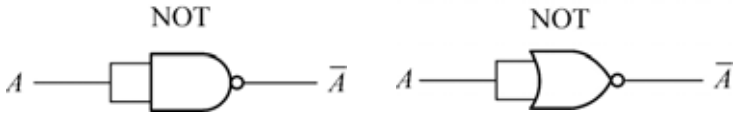
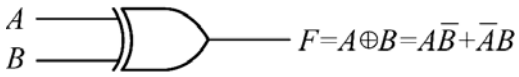


Figure 1-10 NAND and NOR make NOT logic gate

7. XOR Gate:

In the digital system, there is another common logic gate, Exclusive-OR or also called XOR gate. XOR gate is not a basic gate, but a logic circuit combined with some logic gates. Figure 1-11 is the logic symbols and truth table. XOR has a logic symbol and also it has its IC, therefore its usage is very wide. We sometimes see it as a basic logic gate and apply it on circuit. XOR gate usually has two input ends. But normally two or more output ends are not very ecconomical, therefore, we can use XOR to operate serial input.



A B	F
0 0	0
0 1	1
1 0	1
1 1	0

Figure 1-11

There are many ways to use logic gate and combine XOR gate, the most common one is as figure 1-12 (a): Figure 1-12 is combined with AND-OR-NOT logic gates. The output function F is $F = A\overline{B} + \overline{A}B$

$= A \oplus B$

($\oplus$ is XOR operation symbol)

Figure 1-12 (b) is made with NAND gate which can lead to output function F as:

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

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

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

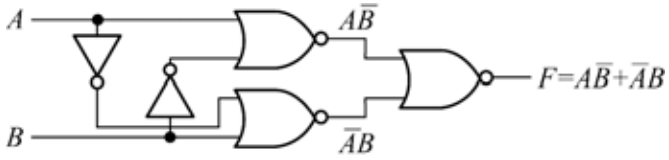
Figure 1-12 (c) is made with NOR gate which can lead to output function F as:

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

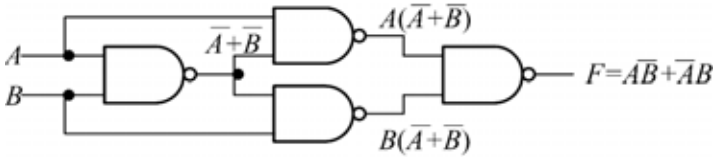
$= \overline{A}A + \overline{A}B + \overline{B}A + \overline{B}B$

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

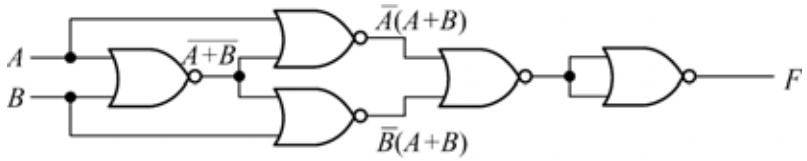
$= A \oplus B$



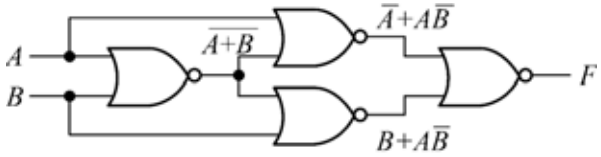
(a)



(b)



(c)



(d)

Figure 1-12

Figure 1-12 (d) is made with NOR gate which can lead to output function F as:

$$
\begin{aligned}
F &= (\overline{A} + A\overline{B})(B + A\overline{B}) \\
&= \overline{A}B + A\overline{B}B + \overline{A}A\overline{B} + A\overline{B}A\overline{B} \\
&= \overline{A}B + A\overline{B} \\
&= A \oplus B
\end{aligned}
$$

XOR gate poses interaction and combination which means that three or more variable functions can be directly present without parentheses.

$$
\begin{aligned}
A \oplus B \oplus C &= (A \oplus B) \oplus C = A \oplus (B \oplus C) \\
&= B \oplus A \oplus C = B \oplus C \oplus A
\end{aligned}
$$

8. XNOR Gate:

A	B	F
0	0	1
0	1	0
1	0	0
1	1	1

Figure 1-13

A similar logic gate such as XOR is called equivalence gate. The character of XOR gate is that when two input ends are the same, the output end is 0, and when two input ends are different, the output is 1 which is totally opposite to equivalence gate. When two input ends are the same,

the output end is 1 and when two input ends are different, the output end is 0. Figure 1-13 is the logic symbol and truth table of equivalence gate. Meanwhile we can also see that its logic symbol is to add an inventor at the output end of exclusive-OR's logic symbol. As follows:

$$\begin{aligned} F &= \overline{A \oplus B} \\ &= \overline{A\overline{B} + \overline{A}B} \\ &= (\overline{A} + B)(A + \overline{B}) \\ &= \overline{A}A + \overline{A}\overline{B} + BA + B\overline{B} \\ &= AB + \overline{A}\overline{B} \\ &= A \odot B \end{aligned}$$

$\odot$ is the operation symbol of equivalence gate. Equivalence gate is also called Exclusive-NOR. Equivalence gate is the supplement of exclusive-NOR. Therefore they have the character of A+B's interaction and combination. By using truth tables, we realize that $A \oplus B \oplus C = A \odot B \odot C$ and $\overline{A \oplus B} = A \odot B$.

As for four or more variables of Boolean Algebra, the formulas are as follows:

$$\begin{aligned} A \oplus B \oplus C \oplus D &= A \oplus (B \oplus C \oplus D) \\ &= A \oplus (B \odot C \odot D) \\ &= \overline{A \odot (B \odot C \odot D)} \\ &= \overline{(A \odot B \odot C \odot D)} \end{aligned}$$

$$\begin{aligned} A \oplus B \oplus C \oplus D \oplus E &= A \oplus (B \oplus C \oplus D \oplus E) \\ &= A \oplus \overline{(B \odot C \odot D \odot E)} \\ &= A \odot (B \odot C \odot D \odot E) \\ &= A \odot B \odot C \odot D \odot E \end{aligned}$$

From the above formula, we are able to know as for N variables, the character is still able to establish, the conclusion is:

(1)When the variable is an even number, exclusive-NOR and equivalence gate are supplements meaning $A \oplus B \oplus C \oplus D \cdots\cdots = \overline{A \odot B \odot C \odot \cdots}$

(2)When the variable is an odd number, exclusive – NOR and equivalence gate have the same meaning $A \oplus B \oplus C \oplus D \cdots\cdots = A \odot B \odot C \ldots..$

If the variable functions are odd numbers, when they apply for the supplement, the formula below is used:

$$\overline{A \oplus B \oplus C} = \overline{(A \oplus B) \oplus C} = A \oplus B \odot C$$

$$= \overline{A \oplus (B \oplus C)} = A \odot B \oplus C$$
$$= \overline{(A \oplus C) \oplus B} = A \oplus C \odot \overline{B}$$
$$= \overline{A \oplus (C \oplus B)} = A \odot C \oplus B$$
$$= \overline{A \odot B \odot C}$$

$$\overline{A \oplus B \oplus C \oplus D \oplus E} = \overline{(A \oplus B \oplus C \oplus D) \oplus E}$$
$$= A \oplus B \oplus C \oplus D \odot E$$
$$= \overline{A \oplus (B \oplus C \oplus D \oplus E)}$$
$$= A \odot B \oplus C \oplus D \oplus E$$
$$= A \oplus B \odot C \oplus D \oplus E$$

Table 1-2

A B C	$A \oplus B$	$A \oplus B \oplus C$	$A \odot B$	$A \odot B \odot C$
0 0 0	0	0	1	0
0 0 1	0	1	1	1
0 1 0	1	1	0	1
0 1 1	1	0	0	0
1 0 0	1	1	0	1
1 0 1	1	0	0	0
1 1 0	0	0	1	0
1 1 1	0	1	1	1

A

B

Even variables

$A \oplus B$ and $A \odot B$

The truth table is the supplement.

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

Odd variables

$A \oplus B \oplus C$ and $A \odot B \odot C$

The truth table is the same meanin

$A \oplus B \oplus C = A \odot B \odot C$

Table 1-2 uses real value to show the relationship between exclusive NOR and equivalence gate. It is rare to use exclusive NOR and equivalence gate to operate Boolean Algebra, except in the digital system design where it often happens, especially for arithmetic calculations and circuit corrections.

1-3 Practice items

1-3-1 7404

Experiment steps

1. Inverter circuit, as figure 1-14.

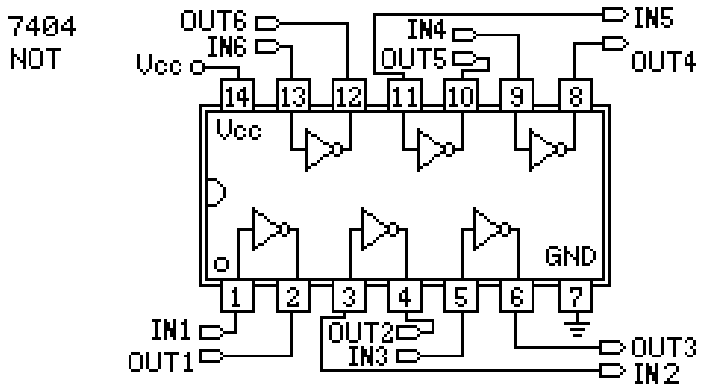


Figure 1-14

2. INPUT: Connect CON7 S1 to CON1 IN1

   OUTPUT: Connect CON4 OUT1 to LED DISPLAY CON15 Q1

3. Switch on S1 when finished connecting, as table 1-3. 0 means low logic LED not on, 1 means high logic LED on.
4. Record LED changes in table 1-3

A(S1)	Y(Q1)
0	
1	

table 1-3

5. Connect a serial inverter circuit as figure 1-15

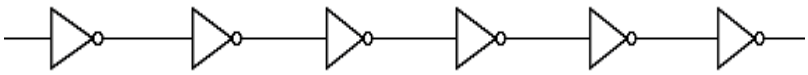


figure 1-15

6. Inputs: Connect CONT7 S1 to CON1 IN1

   Connect 2's OUT1 to CON2 A1

   Connect 3's IN2 to CON3 B1

   Connect A1 and B1

   Connect 4 OUT2 to CON2 A2

   Connect 5 IN3 to CON3 B2

   Connect A2 and B2

   Connect 6 OUT3 to CON2 A3

Connect 9 IN4 to CON3 A4

Connect A3 and B3

Connect 8 OUT4 to CON2 A4

Connect 11 IN5 to CON3 B4

Connect 10 OUT5 to CON2 A5

Connect 13 IN6 to CON3 B5

Connect A5 and B5

Outputs: Connect 12 OUT6 to LED DISPLAY CON15 Q1

7. Switch on S1, as table 1-4, after finished connecting. 0 means low logic LED off, 1 means high logic, LED on.
8. Record LED changes in table 1-4.

A(S1)	Y(Q1)
0	
1	

table 1-4

1-3-2 7432

Experiment steps

1. 2 Input OR circuit as figure 1-16.

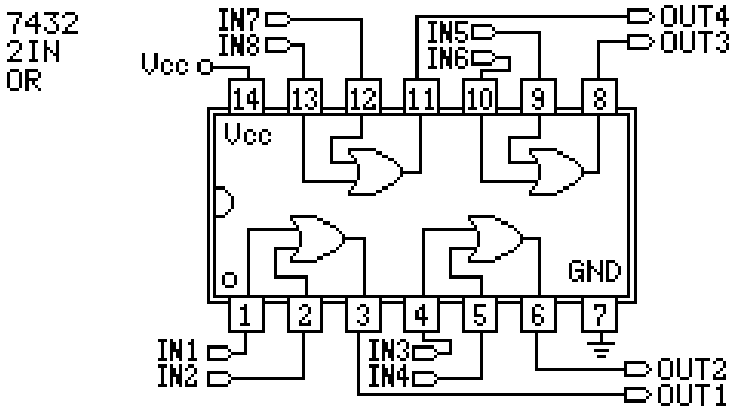


Figure 1-16

2. Inputs: Connect CON7 S1 to CON1 IN1
   Connect CON7 S2 to CON1 IN2
   Outputs: Connect CON4 OUT1 to LED DISPLAY CON15 Q1
3. Switch on S1 and S2, as table 1-5, after finished connecting. 0 means low logic, LED off, 1 means high logic, LED on.
4. Record LED changes in table 1-5.

A(S1)	B(S2)	F(Q1)

0	0	
0	1	
1	0	
1	1	

Table 1-5

5. Connect a 2-connections 3 inputs OR.

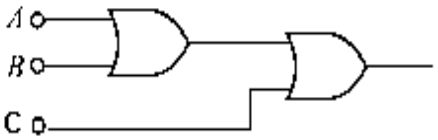


Figure 1-17

6. Inputs: Connect CON7 S1 to CON1 IN1
   Connect CON7 S2 to CON1 IN2
   Connect CON7 S3 to CON1 IN3
   Connect 3 OUT1 to CON2 A1
   Connect 5 in4 to CON3 B1
   Connect A1and B1
   Outputs: Connect 6 OUT2 to LED DISPPLAY CON15 Q1
7. Switch on S1, S2 and S3, as table 1-6, when finished connecting. 0 means low logic LED off, 1 means high logic LED on
8. Record LED changes in table 1-6.

A(S1)	B(S2)	C(S3)	Y(Q1)
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Table 1-6

1-3-3 7408

Experiment steps

1. 2 Input AND circuit, as figure 1-18.

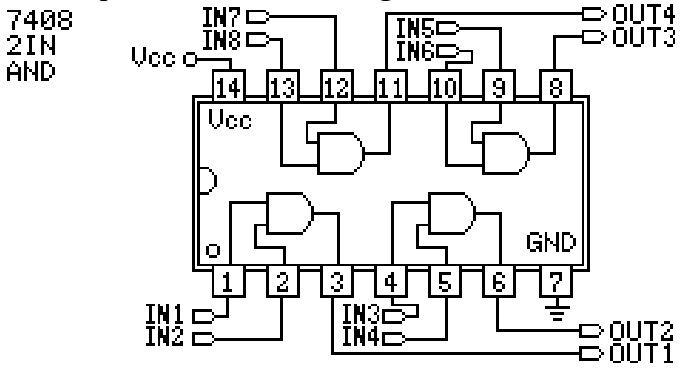


figure 1-18

2. Inputs: Connect CON7 S1 to CON1 IN1
   Connect CON7 S2 to CON1 IN2
   Outputs: Connect CON4 OUT1 to LED DISPLAY CON15 Q1
3. Switch on S1, S2,as table 1-7. 0 means low logic, LED off, 1 means high logic, LED on.
4. Record LED changes in table 1-7.

A(S1)	B(S2)	F(Q1)
0	0	
0	1	
1	0	
1	1	

Table 1-7

5. Connect 2 connections 3 input AND circuit as figure 1-19.

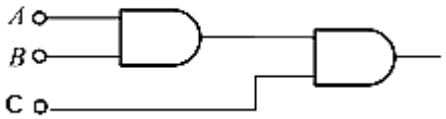


Figure 1-19

6. Inputs: Connect CON7 S1 to CON1 IN1
   Connect CON7 S2 to CON1 IN2
   Connect CON7 S3 to CON1 IN3
   Connect 3 OUT1 to CON2 A1
   Connect 5 IN4 TO CON3 B1
   Connect A1 and B1
   Outputs: Connect 6 OUT2 to LED DISPLAY CON15 Q1
7. Switch S1, S2 and S3, as table 1-8. 0 means low logic, LED off, 1 means high logic, LED ON.
8. Record LED changes in table 1-8.

A(S1)	B(S2)	C(S3)	Y(Q1)
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Table 1-8

1-3-4 7402

Experiment steps:

1. 2 input NOR circuit, as figure 1-20.

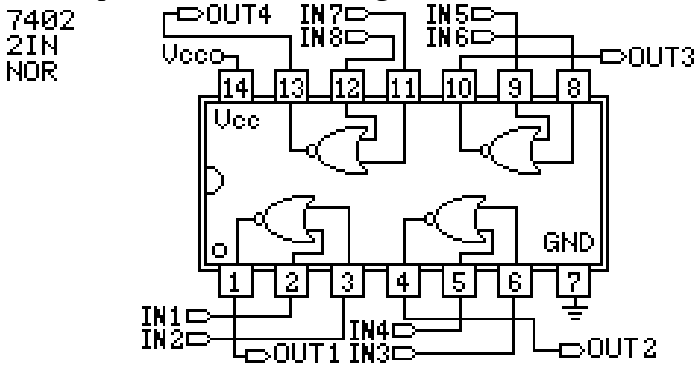


Figure 1-20

2. Inputs: Connect CON7 S1 to OCN1 IN1.
   Connect CON7 S2 to CON1 IN2
   Outputs: Connect CON4 OUT1 to LED DISPLAY CON15 Q1
3. Switch SW, S2, as table 1-9. 0 means low logic, LED off, 1 means high logic, LED ON.
4. Record LED changes in table 1-9

A(S1)	B(S2)	F(Q1)
0	0	
0	1	
1	0	
1	1	

Table 1-9

5. Circuit 1-21

Figure 1-21

6. Inputs: Connect CON1 IN1 to CON21
   Connect IN2 to CON21
   Output: Connect CON4 OUT1 to LED DISPLAY CON15 Q1
7. Switch S1, as table 1-10, after finished connecting. 0 means low logic, LED off, 1 means high logic, LED on.
8. Record LED changes in table 1-10

A(S1)	Y(Q1)
0	
1	

Table 1-10

1-3-5 7400

Experiment steps

1. 2 input NAND circuit, as figure 1-22

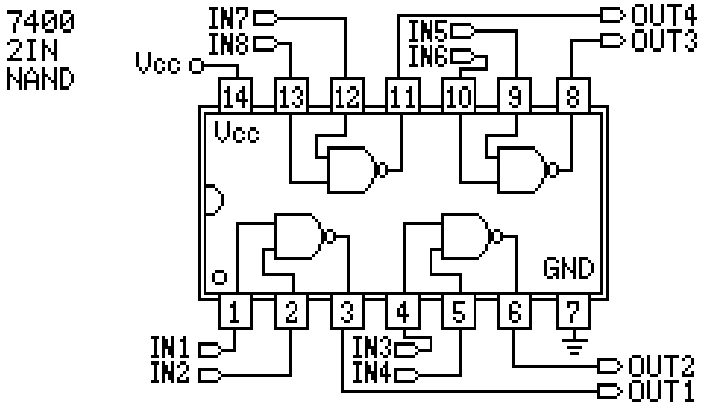


Figure 1-22

2. Inputs: Connect CON7 S1 to CON1 IN1
   Connect CON7 S2 to CON1 IN2
   Output: Connect CON4 OUT 1 to LED DISPLAY CON15 Q1
3. Switch S1, S2, as table 1-11 after finished connecting. 0 means low logic, LED off. 1 means high logic, LED on.

4. Record LED changes in table 1-11

A(S1)	B(S2)	F(Q1)
0	0	
0	1	
1	0	
1	1	

Table 1-11

5. Circuit 1-23

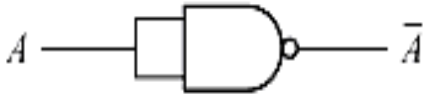


Figure 1-23

6. Inputs: Connect CON1 IN1 to CON2 1
   Connect CON1 IN2 to CON2 1
   Output: Connect CON4 OUT1 to LED DISPLAY CON 15 Q1
7. Switch on S1, as table 1-12, after finished connecting. 0 means low logic, LED off. 1 means high logic, LED on.
8. Record changes in table 1-12

A(S1)	Y(Q1)
0	
1	

Table 1-12

1-3-6 7486

Experiment steps

1. Input XOR circuit, as figure 1-24

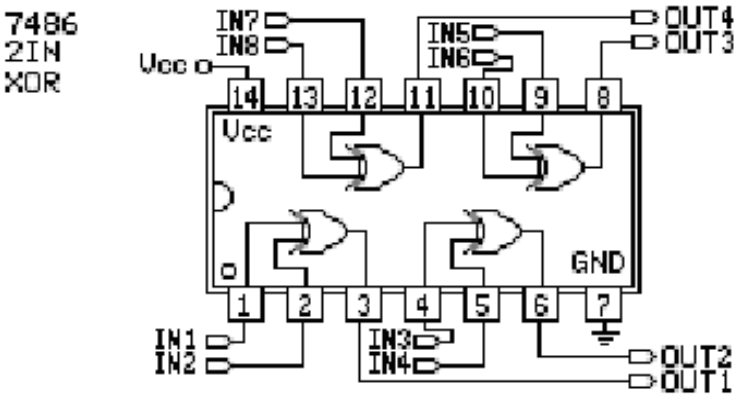


Figure 1-24

2. Inputs: Connect CON7 S1 to CON1 IN1
   Connect CON7 S2 to CON1 IN2
   Output: Connect CON4 OUT 1 to LED DISPLAY CON15 Q1
3. Switch S1, S2, as table 1-13 after finished connecting. 0 means low logic, LED off. 1 means high logic, LED on.
4. Record LED changes in table 1-13

A(S1)	B(S2)	F(Q1)
0	0	
0	1	
1	0	
1	1	

Table 1-13

1-3-7 7411

Experiment steps

1. 3 Input AND circuit, as figure 1-25

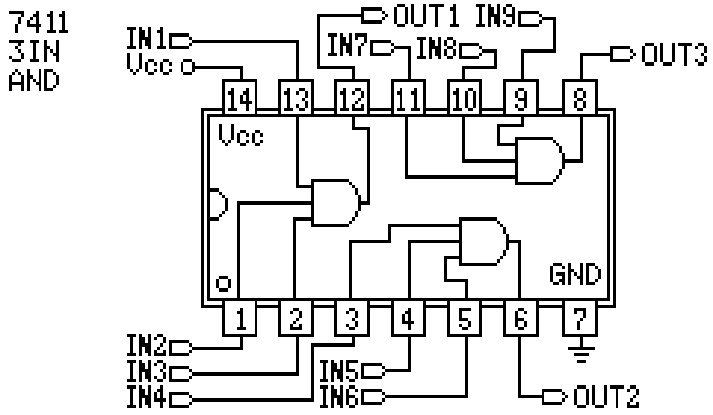


Figure 1-25

2. Inputs: Connect CON7 S1 to CON1 IN1
   Connect CON7 S2 to CON1 IN2
   Connect CON7 S3 to CON1 IN3
   Output: Connect CON4 OUT 1 to LED DISPLAY CON15 Q1
3. Switch S1, S2 and S3, as table 1-14 after finished connecting. 0 means low logic, LED off. 1 means high logic, LED on.

4. Record LED changes in table 1-14

A(S1)	B(S2)	C(S3)	Y(Q1)
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Table 1-14

1-4 Questions & discussion

1. Try to draw NOT, OR, AND, NOR, NAND, EXOR, EXNOR's Gate symbols, real value tables and function formula.
2. Try to define positive and negative logic systems.
3. If input value is N, then how many rows are there in the real value table?

⊙ ⊙ ⊙ ⊙