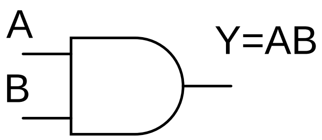
Lecture 1 - Introduction to Discrete Digital Logic

1 Basic Logic Gates
2 TTL & CMOS technology
3 Logic Trainer
4 Elementary Theorem-Identity
5 Creating a 2-Input NOR from a 3-Input NOR
6 K-Maps

Basic Logic Gates

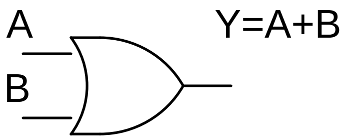
AND Gate
The Notation for AND of A and B is $A * B$ or AB

A	B	Y=AB
0	0	0
0	1	0
1	0	0
1	1	1



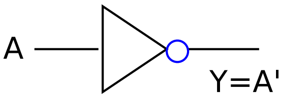
OR Gate
The Notation for OR of A and B is $A + B$

A	B	Y=A+B
0	0	0
0	1	1
1	0	1
1	1	1



Inverter
The notation for the complement or not of A is A'

A	Y=A'
0	1
1	0



TTL & CMOS Technology

TTL stands for Transistor-to-Transistor Logic. In this lab TTL chips will be used to implement basic logic gates such as AND, OR, NAND and others. On TTL technology gates are made of bipolar transistors (see figure 1). Main characteristics of these gates include their relatively high switching speed, immunity to noise and high consumption of power.

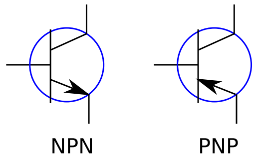
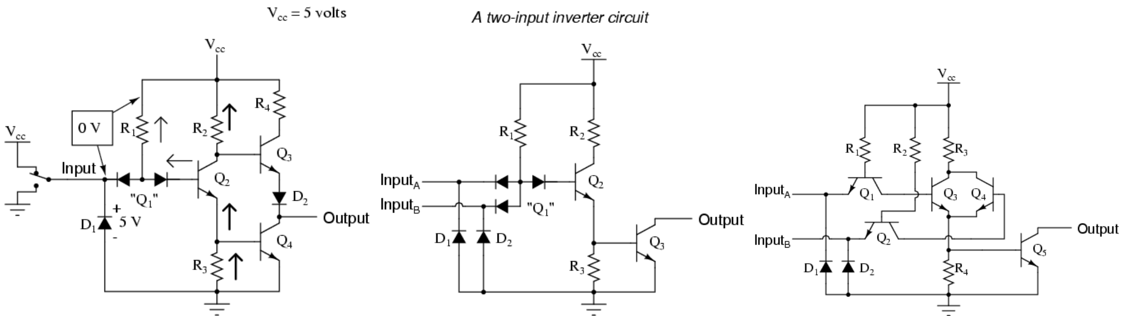


Figure 1: bipolar transistors

TTL chips can be identified by its code, which will always start with the numbers 74 or 54 followed by some more numbers or characters that will serve to uniquely identify the chip.



(a) not (b) nand (c) nor

Figure 2: bipolar basic gates

CMOS stands for complementatry Metal-Oxide Semiconductor. Chips implementing gates with this technology use MOS transistors. Figure 3 shows the symbol for this kind of transistor. In general, a MOS transister is a gate voltage controlled switch. CMOS's power consumption is small, which make them ideal for today's digital applications.

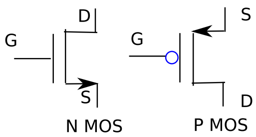
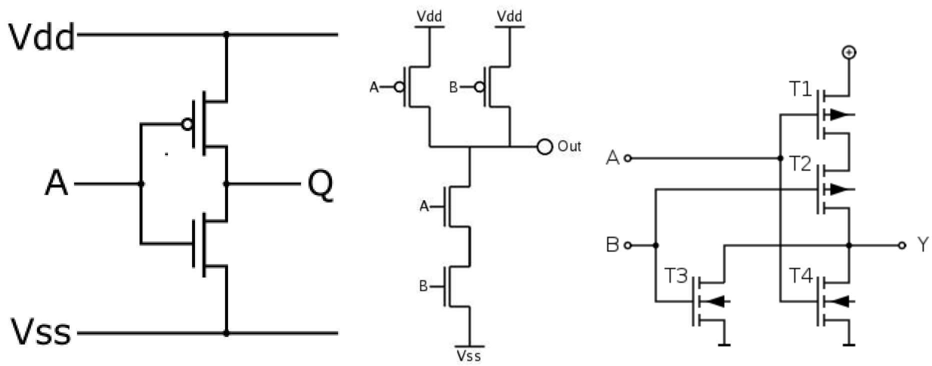


Figure 3: MOS transistors

The basic gates based on CMOS is drawn in following figure:



(a) not (b) nand (c) nor

Figure 4: CMOS basic gates

Our experiment is based on TTL chips. It is necessary to take a look at the datasheet before operation. There are chip schemes on the website, and some example operating conditions are shown in figure 5.

recommended operating conditions

		SN5400			SN7400			UNIT
		MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH}	High-level input voltage	2			2			V
V_{IL}	Low-level input voltage			0.8			0.8	V
I_{OH}	High-level output current			−0.4			−0.4	mA
I_{OL}	Low-level output current			16			16	mA
T_A	Operating free-air temperature	−55		125	0		70	°C

Figure 5: example of information from 7400 TTL datasheet

Another important difference with TTL and CMOS chips is that CMOS can support a wider range of voltage. For instance, while TTL input digital zero levels range from 0 to 0.8 volts, CMOS can go up to 4 volts, Figure 6 shows the voltage ranges from CMOS and TTL.

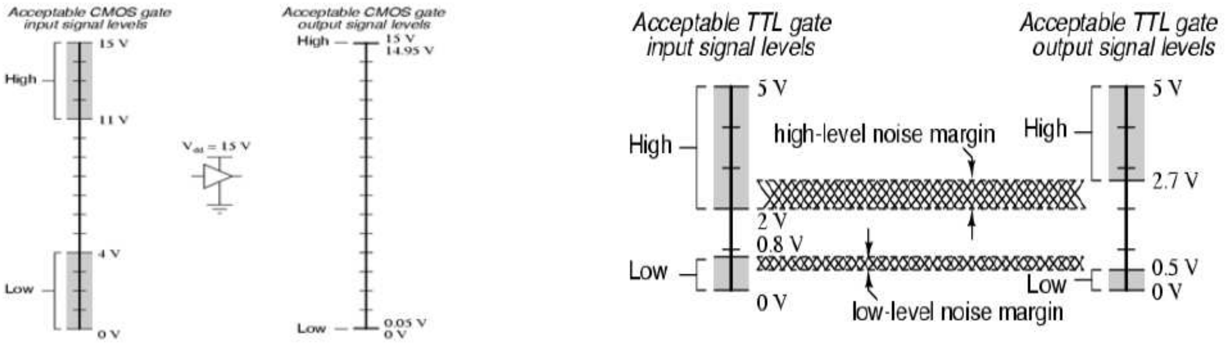


Figure 6: Example of CMOS and TTL voltage Levels

Logic Trainer

A description of basic features for the Logic Trainer (Figure 7) follows:

1 The power switch is in the top left hand corner, Leave the power off while building and modifying the circuit. If you blow a fuse, notify your TA.

2 Sources for +5 Volts and Ground are in the top right hand corner.

3 Switches for inputs are along the bottom of the logic Trainer. Up is true(logic 1). Use jumper wires to connect the switches to the main surface area of the Logic Trainer.

4 The main surface area is made up of white plastic pieces with lots of holes in them. This area is where the TTL chips go. See below for details on TTL chip placement. TTL chips are connected with jumper wires to input switches, other TTL chips, and output LEDs.

5 LEDs for output are on the right side of the Logic Trainer. They light for true (logic 1).

6 There is a logic probe that is used to check for the presence of a high (5 volts) or low volts). The probe is your best tool for verifying that you have voltage where you think you have it. Below (figure 7) is an indication of five volts at the test point.

Figure 7: Operation on Logic Trainer

Elementary Theorem-Identity

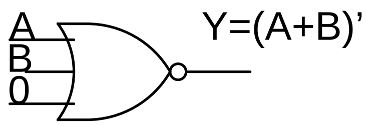
The Identity Theorem specifies that when performing oepration involving any element and the identity element, the result is always the original element. For AND, the identity element is 1; therefore x * 1 = x. For OR, the identity element is 0; therefore x + 0 = x.

K-Maps

This example involves a 2-input NOR function. However, the 2-input function will be implemented using a 3-input NOR gate. In order to determine what the third input of the gate should be we must create a truth table for the 2-input function and a truth table for two 3-input functions one with a third input of 0 and the other with a third input of 1. Comparing the truth table for the two 3-input functions to the truth table for the 2-input function will allow us to find the identity element of the OR operation. The identity element will be the third input of the NOR gate. Here is the truth table for (A+B)', (A+B+0)', and (A+B+1)'.

1	2	3	4	5	6	7
AB	A+B	(A+B)'	A+B+0	(A+B+0)'	A+B+1	(A+B+1)'
00	0	1	0	1	1	0
01	1	0	1	0	1	0
10	1	0	1	0	1	0
11	1	0	1	0	1	0

Column 3 is the function to be implemented. Column 5 is the result of tying one input of a 3-input NOR gate to 0. Column 7 is the result of tying one input of a 3-input NOR gate to 1. By inspection of the truth tables, a 3-input NOR gate with one input tied to 0 will implement a 2-input NOR gate. Here is the logic diagram.



Creating a 2-Input NOR from a 3-Input NOR

This section will show you how to transition from complex(larger) equations to smaller equations and then to hardware. Let's start with a three variable problem and then progress to a four variable.

Three Variables

Boolean function of three variables: $F3 = (Y + Z')X + X'YZ$

a. Truth Table For F3

miniterm	X Y Z	X'	Z'	Y+Z'	(Y+Z')X	X'YZ	F3
m_0	000	1	1	1	0	0	0
m_1	001	1	0	0	0	0	0
m_2	010	1	1	1	0	0	0
m_3	011	1	0	1	0	1	1
m_4	100	0	1	1	1	0	1
m_5	101	0	0	0	0	0	0
m_6	110	0	1	1	1	0	1
m_7	111	0	0	1	1	0	1

The most important information that this truth table gives you is contained in the second and last columns. The second column represents each distinct set of inputs. The last column represents the behavior of the output to each set of inputs. The other columns represent an isolated part of the equations and are useful in determining the final output.

b. K-map For F3

		YZ			
		00	01	11	10
X	0	0	0	1	0
	1	1	0	1	1

X, Y and Z are the inputs. The values in the yellow area are the outputs. You fill in the K-Map by placing each output in the box corresponding to its inputs.

c. Simplify the Boolean function

The K map will allow you to simplify the function, Combine the adjacent 1s in number of two's power. You can warp up the table, which means you can combine ones across the edges.

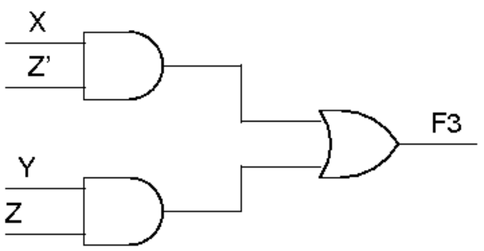
		YZ			
		00	01	11	10
X	0	0	0	1	0
	1	1	0	1	1

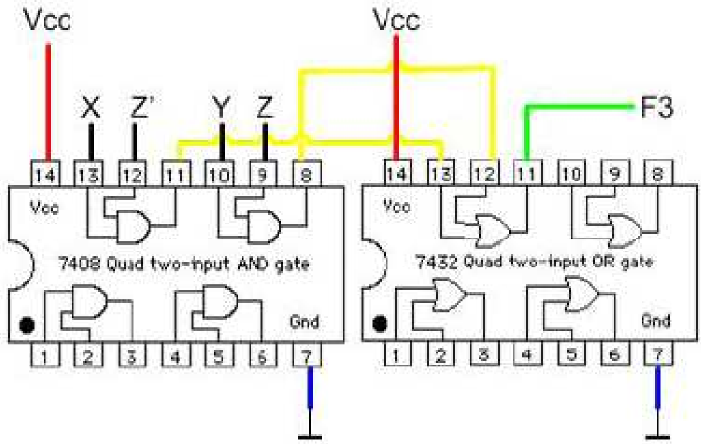
$F3 = YZ + XZ'$

The simplified equations results from two pairs of ones. Each term in the equation is color coded to match the pair that represents it in the K-Map.

d. Design an AND/OR circuit

Based on previous caculation, the logic and layout diagrams are as follows:





Four Variables

Here is the boolean function of four variables: $F4 = B'CE' + A'B'E' + AC'E' + ABC'E$

a. Truth Table for F4

miniterm	ABCE	A'	B'	C'	E'	B'CE'	A'B'E'	AC'E'	ABC'E	F4
m_0	0000	1	1	1	1	0	1	0	0	1
m_1	0001	1	1	1	0	0	0	0	0	0
m_2	0010	1	1	0	1	1	1	0	0	1
m_3	0011	1	1	0	0	0	0	0	0	0
m_4	0100	1	0	1	1	0	0	0	0	0
m_5	0101	1	0	1	0	0	0	0	0	0
m_6	0110	1	0	0	1	0	0	0	0	0
m_7	0111	1	0	0	0	0	0	0	0	0
m_8	1000	0	1	1	1	0	0	1	0	1
m_9	1001	0	1	1	0	0	0	0	0	0
$m_1$0	1010	0	1	0	1	1	0	0	0	1
$m_1$1	1011	0	1	0	0	0	0	0	0	0
$m_1$2	1100	0	0	1	1	0	0	1	0	1
$m_1$3	1101	0	0	1	0	0	0	0	1	1
$m_1$4	1110	0	0	0	1	0	0	0	0	0
$m_1$5	1111	0	0	1	0	0	0	0	0	0

b. K-map For F4

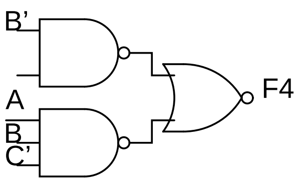
		CE			
		00	01	11	10
AB	00	1	0	0	1
	01	0	0	0	0
	11	1	1	0	0
	10	1	0	0	1

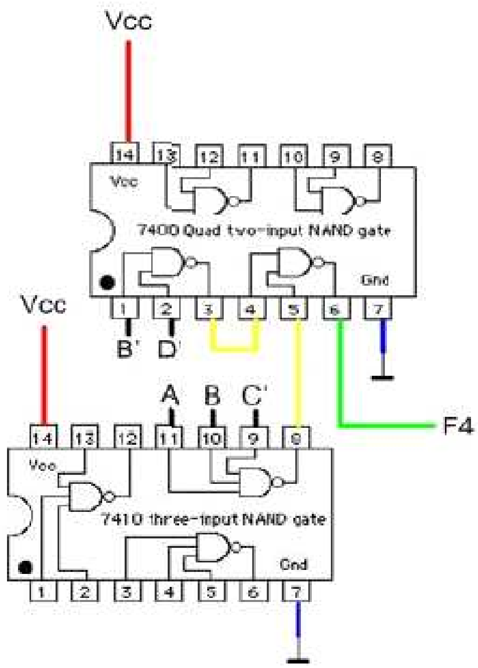
c. Simplify the Boolean function

		CE			
		00	01	11	10
AB	00	1	0	0	1
	01	0	0	0	0
	11	1	1	0	0
	10	1	0	0	1

$$F3 = B'E' + ABC'$$

d. Design a NAND circuit
Draw a logic diagram and a layout diagram





This doc is based on alonzo's work in 2004