L12 : Boolean Algebra & K-Map

We introduced Logic gates in last lecture. Basically, they are the fundamental unit for building digital circuit.

What we are going to do today is to see how we can implement certain "logic function" by using logic gates, [NOT, AND. OR].

1. History of logic

There are two foundations for digital circuit $\begin{cases} \text{Boolean Algebra [Math]} \\ \text{Gates [Physics].} \end{cases}$

Boolean Algebra was coined in the 18xx. It was not well recognized for 100 years, until 1937. In that year, Shannon published his master thesis, identifying the connection between Boolean Algebra & logic gates (switch).

It took another 30 years to combine Boolean Algebra & electronics, and the first computer was built by IBM in 1953. You know what happened after that ...

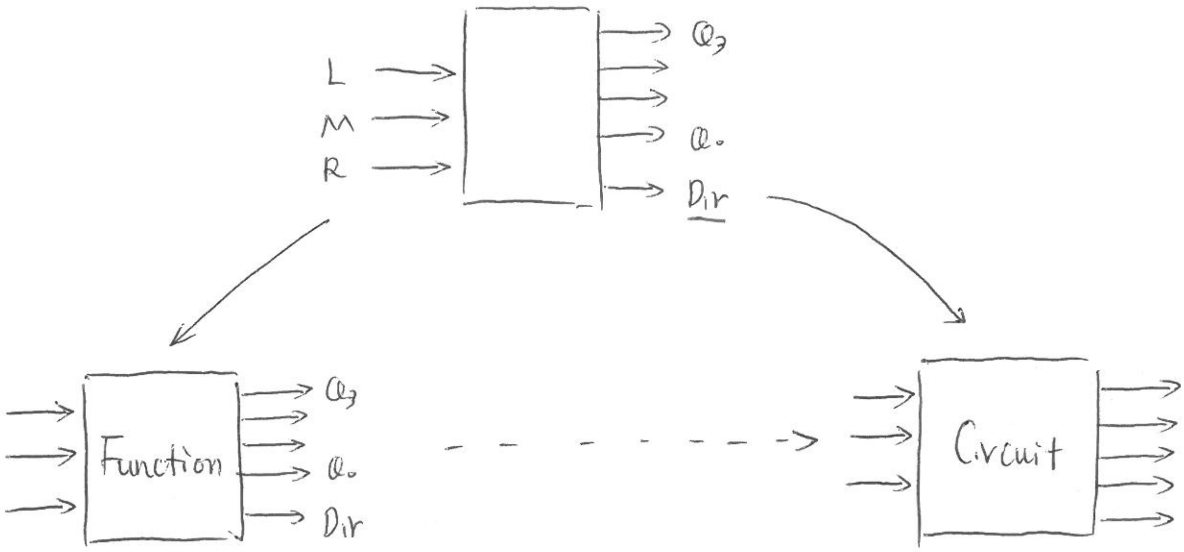
See how Math + Physics ⇒ Engineering?

$$\underset{18xx}{\text{Math}} + \underset{193x}{\text{Physics}} \Rightarrow \underset{1953}{\text{Engineering}}$$

Now, what is our problem here?

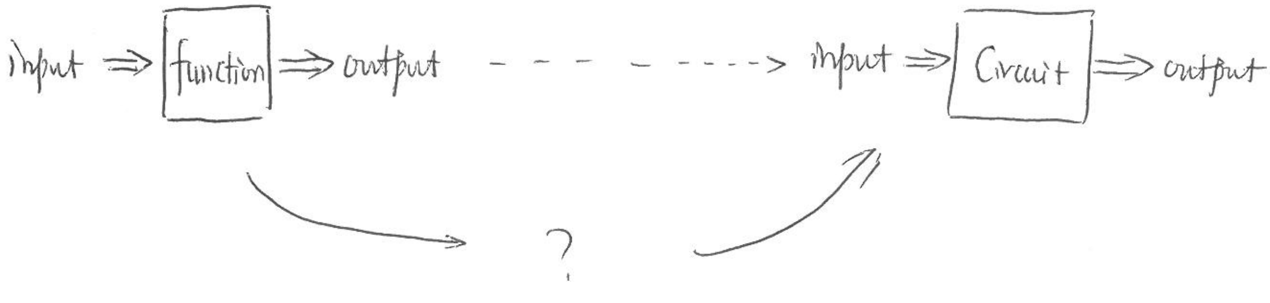
Our target is to build a circuit with a "desired" function.

For example, we want the car to move forward with full power if the input is "1. 0. 1". That means if the input is "1 0 1", then the output should be $Q_3 Q_2 Q_1 Q_0 = 1111$, $Dir = 0$.



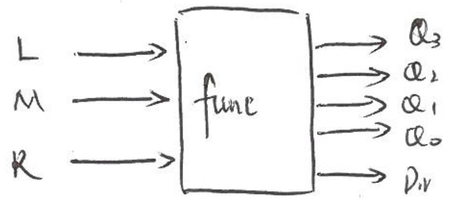
Basically, we have certain functions that we want to achieve, and we hope we can build a circuit that implements those functions.

This process of "from Function to Circuit" is called Circuit Design.



Let's first look at the possible "form" of the circuit.

2. Combinational Logic



We want to determine the output "$Q_3 Q_2 Q_1 Q_0$" & "Dir" for all possible "Combinations" of the inputs "L M R" = "000, 001, --- 111"

We call this "combinational logic"

1) Standard form : Boolean equations can be written in two standard forms :

(Three basic operations NOT, AND, OR)

① Sum of Products (SOP)

NAO

$$X = \underline{A \cdot B} + \underline{B \cdot C \cdot D} + \underline{\overline{E}\,\overline{F}}$$

$L M + L M R + \overline{L}\,\overline{R}$

AND OR

[Note both include the three basic operations]

② Product of Sums (POS)

NOA

$$\overline{X} = (\overline{A} + \overline{B}) \cdot (\overline{B} + \overline{C} + \overline{D}) \cdot (E + F)$$

$(\overline{L} + \overline{M})(\overline{L} + \overline{M} + \overline{R})(\overline{M} + \overline{R})$

OR AND

2) Change of forms

We can change between two forms by using the properties of Boolean Algebra

$$X = (A + B)(C + D) \qquad \text{POS}$$

$\Downarrow$

$$= AC + AD + BC + BD \qquad \text{SOP}$$

$\Downarrow$

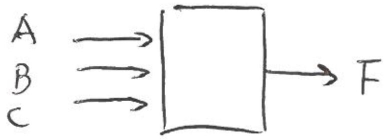
Can be implemented by hardware.

3. How can we design the circuit for a given "function".

Step 1: How is "function" specified? "Truth Table"

Step 2: How to translate "Truth Table" to a circuit?

Consider one example:



A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	$\boxed{1 \rightarrow \bar{A}BC}$
1	0	0	$1 \rightarrow A\bar{B}\bar{C}$
1	0	1	$1 \rightarrow A\bar{B}C$
1	1	0	$1 \rightarrow AB\bar{C}$
1	1	1	$1 \rightarrow ABC$

input output

we can write the product term for each "1" of F.

$$\Rightarrow F = \bar{A}BC + A\bar{B}\bar{C} + A\bar{B}C + AB\bar{C} + ABC$$

(SOP)

Func $\Longrightarrow$ Circuit

$\downarrow$ $\uparrow$

Truth Table $\longrightarrow$ SOP

We can simplify the circuit by Boolean algebra.

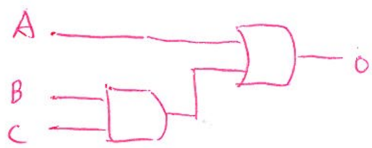
$$\bar{A}BC + ABC = (\bar{A}+A)BC = BC$$

$$A\bar{B}\bar{C} + A\bar{B}C = A\bar{B}(C+\bar{C}) = A\bar{B}$$

$$AB\bar{C} + ABC = AB$$

$$A\bar{B} + AB = A$$

$$= A + BC$$



But, do we have a systematic way to do this?

4. K-Map

An example without the need to do simplification

1) Example : Half adder

Input : two 1-bit numbers

output : one 2-bit number

$$\begin{array}{r} A \\ +\ B \\ \hline C_1\ C_0 \end{array}$$

A	B	C_1	C_0
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$C_0 = \bar{A}B + A\bar{B}$

$C_1 = AB$

Nothing to simplify.

✓ Another example :

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

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

$= \underbrace{\bar{A}\bar{B}} + \bar{A}\bar{B} + \bar{A}B + A\bar{B}$

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

$= \bar{A}(\bar{B} + B) + (\bar{A} + A)\bar{B}$

$= \bar{A} + \bar{B}$ We can simply it.

What's the difference.

(Find common factor)

with only one element different

Can we rearrange the truth table in another way? K-Map

rearrange the SOP.

	$\bar{A}$	A
$\bar{B}$	$1_{\bar{A}\bar{B}}$	$1_{A\bar{B}}$
B	$1_{\bar{A}B}$	0_{AB}

In the above simplification, we duplicate $\bar{A}\bar{B}$ and use it to combine with other two terms.

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

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

Common element.

Then, how about the half adder example.

C_1

	A	$\bar{A}$
B	1	0
$\bar{B}$	0	0

$C_1 = AB$

C_0

	A	$\bar{A}$
B	0	1
$\bar{B}$	1	0

$C_0 = A\bar{B} + \bar{A}B$

no common element

no simplification..

2) How about 3 variables?

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

one & only one difference.

F:

	$\bar{A}\bar{B}$	$\bar{A}B$	AB	$A\bar{B}$
$\bar{C}$	1	0	0	1
C	0	1	1	0

How to simplify?

$C\bar{A}B + CAB = CB(\bar{A}+A) = CB$

$\bar{A}\bar{B}\bar{C} + A\bar{B}\bar{C} = (\bar{A}+A)\bar{B}\bar{C} = \bar{B}\bar{C}$

So they are also adjacent (only one difference)

$\bar{A}\bar{B} \quad \bar{A}B \quad AB \quad A\bar{B}$

3) How about 4 variables?

	$\bar{A}\bar{B}$	$\bar{A}B$	AB	$A\bar{B}$
$\bar{C}\bar{D}$	0	0	0	0
$\bar{C}D$	0	1	1	0
CD	0	1	1	0
$C\bar{D}$	0	0	0	0

$\bar{A}B\bar{C}D + AB\bar{C}D = B\bar{C}D$

$\bar{A}BCD + ABCD = BCD$

$+ \Rightarrow BD(\bar{C}+C) = BD$

Now, steps to use K-Map for simplying circuit.

1) Begin with isolated "1" ⟹ No simplification

2) Find "two cell" group ⟹ Simplify

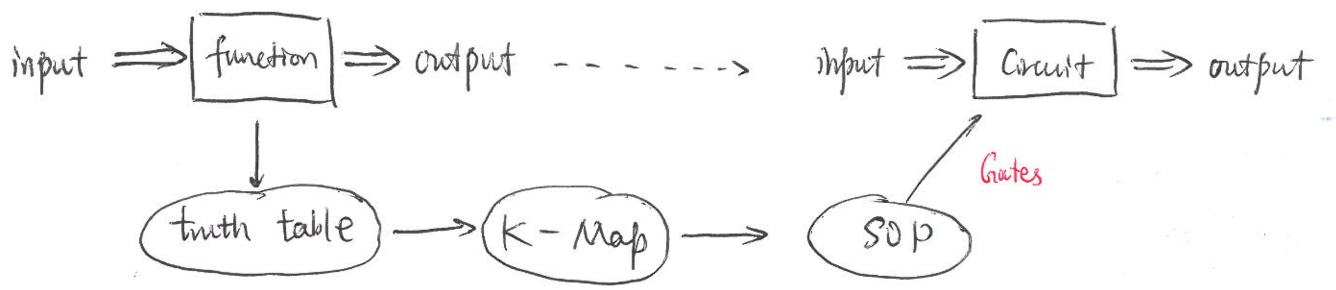
3) " "four " " ⟹ Simplify

⋮

4) Sum them together ⟹ Done!

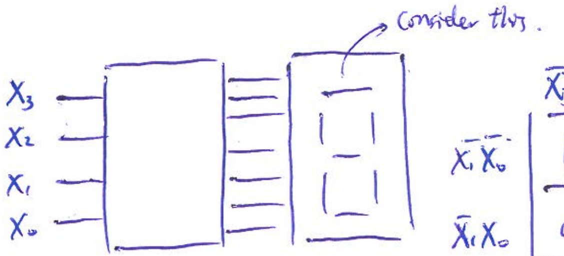
5.

Finally: steps for circuit design:



6. I don't Care.

In logic design, there are cases where some of the input combinations are not possible/allowed. As a result, we don't need to care about them. For example. consider the system where we want to display 0 ~ 9 by a 7-segment display. Obviously, only the combinations 0000 ~ 1001 matter to us. For the other cases, we don't care. How do we simplify K-MAP for such cases.?



	$\bar X_3\bar X_2$	$\bar X_3 X_2$	$X_3 X_2$	$X_3\bar X_2$
$\bar X_1\bar X_0$	1	0	X	1
$\bar X_1 X_0$	0	1	X	1
$X_1 X_0$	1	1	X	X
$X_1\bar X_0$	1	1	X	X

1	0	0	1	$\bar X_3\bar X_2\bar X_1$
0	1	0	1	$\bar X_2\bar X_1\bar X_0$
1	1	0	0	$\bar X_3 X_2 X_0$
1	1	0	0	$\bar X_3 X_1$

Compare

1	0	1	1	$\bar X_2\bar X_0$
0	1	1	1	$X_2 X_0$
1	1	1	1	X_3
1	1	1	1	X_1