Development of Truth Tables

Goal: Given a word problem, develop a truth table. Why? Truth table → Minterm or Maxterm list → Simplified SOP or POS function → Circuit.

Ex1. A system has 3 inputs *A, B,* and *C*, and one output *Z*, such that *Z* = 1 iff exactly two of the inputs are HIGH (1). Create its truth table.

Ex2. A system has 4 inputs and three outputs. The first 2 inputs (X_1, X_0) represent a two 2-bit binary number, and the last 2 inputs (Y_1, Y_0) represent another 2-bit number. The 3 outputs are defined as follows:
F = 1 iff the two numbers differ by exactly 2.
G = 1 iff the two numbers are equal.
H = 1 iff the second number is larger that the first.
Create its truth table.

2.5: From Truth Table to Algebraic Expressions

Goal: Given the truth table of a function, derive its algebraic expression.

Minterm: a **canonical product term** (a *product term* where every input variable appears once).

For a **2**-input function, there are 4 minterms. Ex: For the function *F*(*a,b*)

Minterm	Looks like	Designation
a'b'		
a'b		
a b'		
a b		

For minterms:
0 = variable is primed
1 = no prime

For a **3**-input function, there are 8 minterms. Ex: For the function *F*(*a,b,c*)

Minterm	Looks like	Designation
a'b'c'		
a'b'c		
a'b c'		
a'b c		
a b'c'		
a b'c		
a b c'		
a b c		

Maxterm: a **canonical sum term** (a *sum term* where every input variable appears once).

For a **2**-input function, there are 4 maxterms. Ex: For the function *F*(*a,b*)

Maxterm	Looks like	Designation
a + b		
a + b'		
a'+ b		
a'+ b'		

For maxterms:
0 = no prime
1 = variable is primed

For a **3**-input function, there are 8 maxterms. Ex: For the function *F*(*a,b,c*)

Maxterm	Looks like	Designation
a + b + c		
a + b + c'		
a + b'+ c		
a + b'+ c'		
a'+ b + c		
a'+ b + c'		
a'+ b'+ c		
a'+ b'+ c'		

- **Sum of minterms (also known as minterm list)**
  Look for the 1's in the truth table.

a	*b*	*F*
0	0	0
0	1	1
1	0	1
1	1	1

$F(a,b) =$

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

$F(A,B,C) =$

- **Product of maxterms (also known as maxterm list)**
  Look for the 0's in the truth table.

a	*b*	*F*
0	0	0
0	1	1
1	0	1
1	1	1

$F(a,b) =$

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

$F(A,B,C) =$

Summary

A function can be expressed in minterms/SOP ***or*** in maxterms/POS forms:

- Look for 1's in truth table → minterm list → leads to an SOP expression
- Look for 0's in truth table → maxterm list → leads to an POS expression

Once the SOP or POS expression is obtained, a logic circuit can be derived.

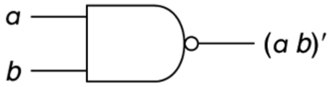
More Examples:

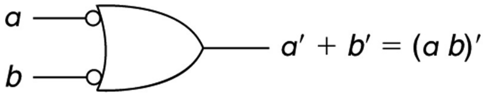
Ex 1. (see p. 62). Given $F(w,x,y,z) = \sum m(0,1,5,9,11,15)$, derive:
1) SOP equation 2) Maxterm list 3) POS equation

Ex 2. Given $F(x,y,z) = \pi M(0,1,4,7)$, derive:
1) SOP equation 3) Minterm list
2) POS equation 4) Complement function $F'(x,y,z)$ in minterm list form

2.6: NAND, NOR and XOR Logic Functions (Derived Functions)

NAND = NOT-AND (AND gate followed with a NOT gate)

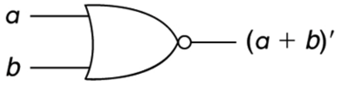


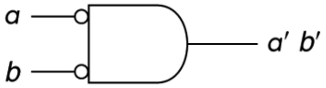


Logic Symbol Alternate (Equivalent) Logic Symbol

Truth Table Timing Diagram

NOR = NOT-OR (OR gate followed with a NOT gate)

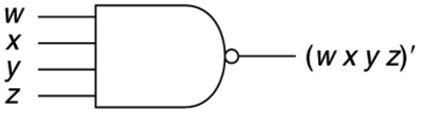




Logic Symbol Alternate (Equivalent) Logic Symbol

Truth Table Timing Diagram

NAND gates and NOR gates can have more than 2 inputs.
Example: 4-input NAND



Truth table has 16 combinations. Output is 0 only for one input combination. Which one?

In practice, most circuits are implemented using only either NAND gates or NOR gates!!! [NAND is most preferred, because it is the fastest gate.]

Circuit Implementation using NAND gates only

Any two-level AND/OR circuit can be converted into a NAND-gate circuit implementation.

1. Replace all gates by NAND gates
2. Any input going directly into original OR gates are complemented.

Ex: $F(x,y,z) = x'y + x\,y' + z$ [SOP form ⇒ Two-level AND/OR circuit]

Circuit Implementation using NOR gates only

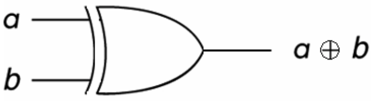
Any two-level OR/AND circuit can be converted into a NAND-gate circuit implementation.

1. Replace all gates by NOR gates
2. Any input going directly into original AND gates is complemented.

Ex: $G(x,y,z) = (x + y')(y' + z)(x')$ [POS form ⇒ Two-level OR/AND circuit]

Multi-level circuits are trickier to implement using single-type gate.
Ex: $F(\mathrm{w},x,y,z) = wx(y+z) + x'y$ [Neither SOP nor POS form]

XOR = Exclusive OR (can have 2 or more inputs)



Logic Symbol Truth Table Timing Diagram

Definition of XOR function: $\boldsymbol{a \oplus b = a\,b' + a'\,b = \Sigma m(1,2) = \prod M(\quad)}$
XOR function is both commutative and associative.
$a \oplus b = b \oplus a$ $a \oplus b \oplus c = (a \oplus b) \oplus c = a \oplus (b \oplus c)$

Useful properties:
$(a \oplus b)' =$

$a' \oplus b =$

$a \oplus b' =$

$a \oplus \mathbf{1} =$

$a \oplus \mathbf{0} =$

Example using XOR: $G(x,y,z) = (x \oplus y')(z')$ What is the minterm list?