COMPUTER LOGIC AND DIGITAL CIRCUIT DESIGN (PHYS306/COSC330): UNIT 2.3

Jordan Hanson

March 13, 2018

Whittier College Department of Physics and Astronomy

SUMMARY

Reading: DF Chapter 3-4 (Moodle)

1. Logic Gates
    - Circuit diagram
    - Truth table
    - Timing diagram
    - Boolean logic
2. Boolean algebra I
3. IC Circuits, **data sheets**
4. Boolean algebra II

Homework: Chapter 3, ex. 1-22 (two weeks)

BOOLEAN ALGEBRA BASICS

Good paper topic: George Boole and Claude Shannon in the mid-19th century, early 20th century.

Boolean or logical algebra - *An Investigation of the Laws of Throught, on Which Are Founded the Mathematical Theories of Logic and Probabilities* (Boole's title)

What does *an algebra* contain?

1. Variables or *literals*
2. *complements* (resembles *identities, reciprocals, and nulls*)
3. Sums are ORs
4. Products are ANDs

An algebra has many objects, properties, and rules. An extensive list is outside our scope.

Boolean or logical algebra has these *objects*:

1. Variables: A is a variable, either TRUE or FALSE
2. complements: $\overline{A}$ is the *complement* of A, inverting the input

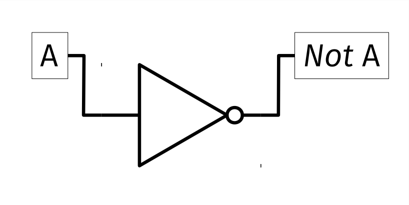


Figure 1: The inverter performs the complement operation at gate-level, $\overline{A} = NOT\ A$.

AND is multiplication, and OR is addition. Refer to Chapter 2 DF.

Boolean or logical algebra has these *properties*:

1. Property of commutativity
   - **Addition version:** $A + B = B + A$
   - (*an OR gate does not distinguish between inputs*)
   - **Mutliplication version:** $AB = BA$
   - (*an AND gate does not distinguish between inputs*)
2. Property of Associativity
   - **Addition version:** $(A + B) + C = A + (B + C)$
   - *OR gate order does not matter.*
   - **Mutliplication version:** $(AB)C = A(BC)$
   - *AND gate order does not matter.*
3. Property of Distribution
   - $A(B + C) = AB + BC$ (*gate input and gate order*)

Boolean or logical algebra has these *rules*:

1. $A + 0 = A$
2. $A + 1 = 1$
3. $A \cdot 0 = 0$
4. $A \cdot 1 = A$
5. $A + A = A$
6. $A + \overline{A} = 1$
7. $AA = A$
8. $A\overline{A} = 0$
9. $\overline{\overline{A}} = A$

Boolean or logical algebra has these *special rules*:

1. $A + AB = A$
2. $A + \overline{A}B = A + B$
3. $(A + B)(A + C) = A + BC$

Exercises:

- Prove special rule (1) with a truth table
- Why is special rule (1) intuitive?
- Prove special rule (2) with a truth table
- Why is special rule (2) intuitive?
- **Group board exercise:** Prove special rule (3) *Hint: this requires rules 2, 4, and 7. This is good practice at **gate simplification**.*

BOOLEAN ALGEBRA BASICS

De Morgan's Theorems

- **First theorem**: The complement of a product of literals is equal to the sum of the complements of the literals.
- **Second theorem**: The complement of a sum of literals is equal to the product of the complements of the literals.

- First theorem: $\overline{AB} = \overline{A} + \overline{B}$
- Second theorem: $\overline{A + B} = \overline{A}\,\overline{B}$

For example: $\overline{(AB + BC)} = \overline{AB}\;\overline{BC} = (\overline{A} + \overline{B})(\overline{B} + \overline{C})$

Group board exercise: Show that

$$\boxed{\overline{(A+B)(B+C)} + \overline{(A+\overline{A}B)(B+\overline{B}A)} = \overline{A}+\overline{B}+\overline{C}} \qquad (1)$$

- *Hint*: use De Morgan's two theorems
- *Hint*: use special rules
- *Hint*: consider the regular rules

Group board exercise: Create a circuit using logic gates that represents the left-hand side of Eq. 1, and do the same for the right-hand side. How many gates fewer are there in the right-hand solution?

Group board exercise: Show that

$$\boxed{\begin{aligned}&(AB + C(B + A)) + \overline{(AB + C(B + A))}(AC + B(C + A))\\ &\qquad = AB + AC + BC\end{aligned}} \tag{2}$$

- *Hint*: use special rules
- *Hint*: use De Morgan's two theorems

Group board exercise: Create a circuit using logic gates that represents the left-hand side of Eq. 2, and do the same for the right-hand side. How many gates fewer are there in the right-hand solution?

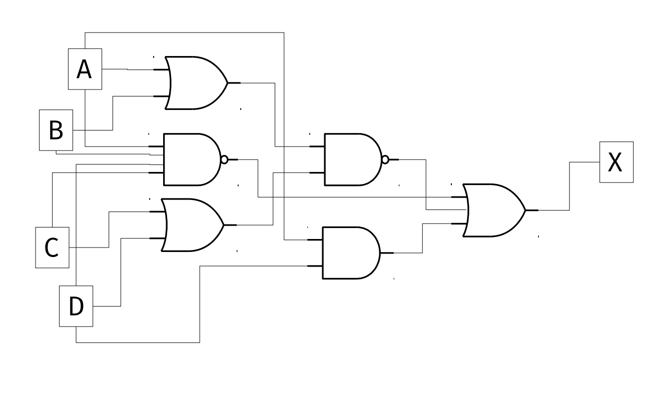


Figure 2: (1) Derive the logical algebraic expression for *X*. (2) Proceed left to right. (3) Simplify the result using simple rules. (4) Draw the equivalent circuit using gates.

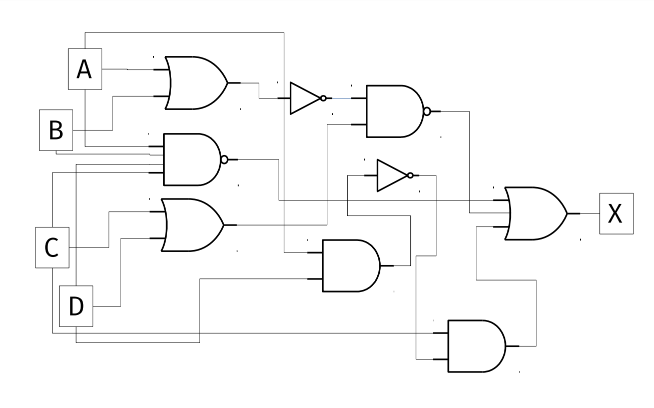


Figure 3: (1) Derive the logical algebraic expression for *X*. (2) Proceed left to right. (3) Simplify the result using simple rules. (4) Draw the equivalent circuit using gates.

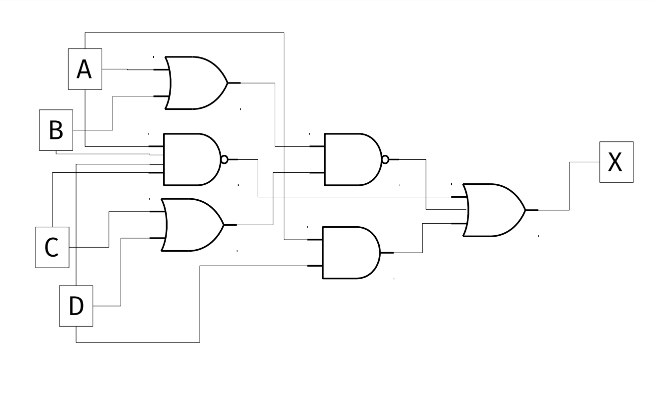


Figure 4: Construct the truth table.

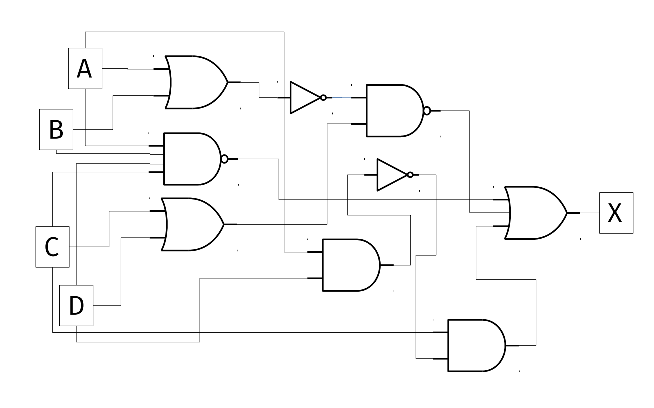


Figure 5: Construct the truth table.

SOP AND POS MINIMIZATION

Is there a standard way for performing these actions?

SOP - *Sum of products* form.

POS - *Product of sums* form.

Rules for SOP form:

1. All terms must be grouped by OR operations
2. No complements of literal groups, e.g. $\overline{ABD}$
3. Example: $A + \overline{B} + AB + ABCD$

SOP - *Sum of products* form.

S-SOP - **Standard** *Sum of products* form.

Rules for S-SOP form:

1. All terms must be grouped by OR operations
2. No complements of literal groups, e.g. $\overline{ABD}$
3. Each term must *span the domain* of literals
4. Example: $ABC + A\overline{B}C + AB\overline{C}$

Notice that S-SOP is not necessarily the simplest form, but it has a different usage.

Build the truth table for this S-SOP expression:

$$X = \overline{A}BCD + A\overline{B}CD \tag{3}$$

1. How many input combinations are there?
2. How many input combinations result in $X = TRUE$?

$$X = \overline{A}BCD + A\overline{B}CD \tag{4}$$

A	B	C	D	X
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	**1**	**1**	**1**	**1**
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	**0**	**1**	**1**	**1**
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

Table 1: Truth table for Eq. 4. S-SOP makes identifying TRUE final states trivial by identifying TRUE codes for individual terms.

How do we convert to SOP and then S-SOP form?

1. SOP: expand terms using De Morgan's theorems and simple rules
2. S-SOP: use rule 6: $A + \overline{A} = 1$ to complete the domain of each OR term

---

Example: The domain is $ABCD$

$$X = A\overline{B} + BC(A + D) \tag{5}$$

$$X = A\overline{B} + ABC + BCD \tag{6}$$

$$X = A\overline{B}(C + \overline{C})(D + \overline{D}) + ABC(D + \overline{D}) + (A + \overline{A})BCD \tag{7}$$

$$X = A\overline{B}CD + A\overline{B}\,\overline{C}\,\overline{D} + ABCD + ABC\overline{D} + ABCD + \overline{A}BCD \tag{8}$$

(1) How many unique TRUE states exist for X? (2) Which input codes make X TRUE?

Is there a standard way for performing these actions?

SOP - *Sum of products* form.

POS - *Product of sums* form.

Rules for POS form:

1. All terms must be grouped by AND operations
2. No complements of literal sums, e.g. $\overline{A+D}$
3. Example: $(A+C)(A+\overline{B}+C)$

POS - *Product of sums* form.
S-POS - **Standard** *Product of sums* form.

Rules for S-POS form:

1. All terms must be grouped by AND operations
2. No complements of literal sums, e.g. $\overline{A+D}$
3. Each term must *span the domain* of literals
4. Example: $(A+B+C)(A+\overline{B}+C)$

Notice that S-POS is not necessarily the simplest form, but it has a different usage.

Build the truth table for this S-POS expression:

$$X = (A+B+C+D)(\overline{A}+B+C+D)(A+\overline{B}+C+D)(A+B+\overline{C}+D) \quad (9)$$

1. How many input combinations are there?
2. How many input combinations result in $X = FALSE$?

$$X = (A+B+C+D)(\overline{A}+B+C+D)(A+\overline{B}+C+D)(A+B+\overline{C}+D) \quad (10)$$

A	B	C	D	X
0	**0**	**0**	**0**	**0**
0	0	0	1	1
0	**0**	**1**	**0**	**0**
0	0	1	1	1
0	**1**	**0**	**0**	**0**
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	**0**	**0**	**0**	**0**
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Table 2: Truth table for Eq. 10. S-POS makes identifying FALSE final states trivial by identifying FALSE codes for individual terms.

How do we convert to POS and then S-POS form?

1. POS: Factor terms using De Morgan's theorems and simple rules
2. S-POS: Add a rule 8 pair to terms not spanning domain: $A\overline{A}$
3. Apply rule 12 to this term: $A + BC = (A + B)(A + C)$
4. (Looks like this: $A + B\overline{B} = (A + B)(A + \overline{B})$)

Example: The domain is $ABCD$

$$X = (A\overline{B} + C + D) + (A + C) \tag{11}$$

$$X = (A\overline{B} + C + D) + (A + C + B\overline{B}) \tag{12}$$

$$X = K + (Q + B\overline{B}) \tag{13}$$

$$X = K + (Q + B)(Q + \overline{B}) \tag{14}$$

$$\ldots\ (\textit{continued}) \tag{15}$$

How do we convert to POS and then S-POS form?

1. POS: Factor terms using De Morgan's theorems and simple rules
2. S-POS: Add a rule 8 pair to terms not spanning domain: $A\overline{A}$
3. Apply rule 12 to this term: $A + BC = (A + B)(A + C)$
4. (Looks like this: $A + B\overline{B} = (A + B)(A + \overline{B})$)

Example: The domain is $ABCD$

$$X = K + (Q + B)(Q + \overline{B}) \tag{16}$$

$$X = (K + Q + B)(K + Q + \overline{B}) \tag{17}$$

$$X = (A + B + C + D)(A + \overline{B} + C + D) \tag{18}$$

(1) How many unique FALSE states exist for X? (2) Which input codes make X FALSE?

Conversions between S-SOP and S-POS

SOP to POS	POS to SOP
1. Determine domain size	1. Determine domain size
2. Identify TRUE codes	2. Identify FALSE codes
3. Remaining codes are the FALSE codes for POS	3. Remaining codes are the TRUE codes for SOP
4. Write sum terms corresponding to the FALSE codes	4. Write product terms corresponding to the TRUE codes
5. Multiply all sum terms	5. Add all product terms

The bottom line: a code cannot give both a TRUE and FALSE answer.

Exercise: Convert the following from SOP to POS:

$$\overline{A}\ \overline{B}\ \overline{C} + \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}C + ABC = X \tag{19}$$

Exercise: Convert the following from SOP to POS:

$$\overline{A}\ \overline{B}\ \overline{C} + \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}C + ABC = X \tag{20}$$

- TRUE codes: 000, 010, 011, 101, 111
- FALSE codes for POS: 001, 100, 110
- First false code sum-term: $(A + B + \overline{C})$, ...

Using similar strategies, SOP and POS expressions can be derived from a truth table (TT). The converse is simpler: convert to S-SOP or S-POS and fill in the table.

TT to S-SOP

1. Locate TRUE final states in TT
2. Write the input codes as product terms, replacing 1's with literals and 0's with complements
3. Sum the product terms

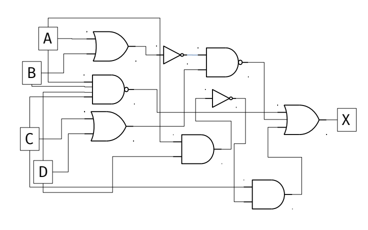
TT to S-POS

1. Locate FALSE final states in TT
2. Write the input codes as sum terms, replacing 1's with complement and 0's with literals.
3. Multiply the sum terms

Where it gets interesting:

Design requirements → TT → S-SOP or S-POS → Simplification via De Morgan's/simple and special rules → digital circuit design

"Ya know Bob..I think we need it to do this new action..." →



Before going there, we will discuss Karnaugh maps next time.

CONCLUSION

Reading: DF Chapter 3-4 (Moodle)

1. Logic Gates
    - Circuit diagram
    - Truth table
    - Timing diagram
    - Boolean logic
2. Boolean algebra I
3. IC Circuits, **data sheets**
4. Boolean algebra II

Homework: Chapter 3, ex. 1-22 (two weeks)