MODULE 3: Boolean Algebra and Digital Logic

# Lecture 3.2
# Representing Boolean Functions


Prepared By:

- Scott F. Midkiff, PhD
- Luiz A. DaSilva, PhD
- Kendall E. Giles, PhD

Electrical and Computer Engineering

Virginia Tech

# Lecture 3.2 Objectives

- Represent Boolean functions in sum-of-product (SOP) form
- Represent Boolean functions in product-of-sum (POS) form
- Define “minterm” and “maxterm”

# Example Logic Function and Minterms

- Consider function f(a,b,c) specified by the truth table below
- Every row in the truth table corresponds to a unique combination of values for inputs a, b, and c
- Every row corresponds to a "minterm" containing all variables

| a | b | c | f | minterm |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | a'b'c' |
| 0 | 0 | 1 | 1 | a'b'c |
| 0 | 1 | 0 | 0 | a'bc' |
| 0 | 1 | 1 | 1 | a'bc |
| 1 | 0 | 0 | 1 | ab'c' |
| 1 | 0 | 1 | 0 | ab'c |
| 1 | 1 | 0 | 0 | abc' |
| 1 | 1 | 1 | 1 | abc |

# More on Minterms

- Example function f(a,b,c) is TRUE (“1”) if and only if the minterm in each row with f=1 is TRUE
- For example, f = 1 if ab’c’ = 1, i.e., a=1, b=0, c=0

| a | b | c | f | minterm |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | a’b’c’ |
| 0 | 0 | 1 | 1 | a’b’c |
| 0 | 1 | 0 | 0 | a’bc’ |
| 0 | 1 | 1 | 1 | a’bc |
| 1 | 0 | 0 | 1 | ab’c’ |
| 1 | 0 | 1 | 0 | ab’c |
| 1 | 1 | 0 | 0 | abc’ |
| 1 | 1 | 1 | 1 | abc |

# Sum-of-Product (SOP) Form

- Example function f(a,b,c) can be expressed as the “OR” of the true minterms
- f(a,b,c) = a’b’c’ + a’b’c + a’bc + ab’c’ + abc
- This is the canonical sum-of-products form of function f, using two-level AND/OR logic

| a | b | c | f | minterm |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | a’b’c’ |
| 0 | 0 | 1 | 1 | a’b’c |
| 0 | 1 | 0 | 0 | a’bc’ |
| 0 | 1 | 1 | 1 | a’bc |
| 1 | 0 | 0 | 1 | ab’c’ |
| 1 | 0 | 1 | 0 | ab’c |
| 1 | 1 | 0 | 0 | abc’ |
| 1 | 1 | 1 | 1 | abc |

# NAND Function

- NAND function is realized by applying AND and then NOT Output is TRUE (“1”) if and only if any input is FALSE (“0”)

| a | b | $F=\overline{ab}$ |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

$$F = (ab)' = \overline{ab}$$

# NAND/NAND Form

- Example function f(a,b,c) can also be implemented in NAND/NAND form using DeMorgan's Theorem, (x+y)' = x'y'
- So, f(a,b,c) can be realized using AND and OR operations or using only NAND operations

```
Step 1: f = a'b'c' + a'b'c + a'bc + ab'c' + abc
Step 2: f = [(a'b'c' + a'b'c + a'bc + ab'c' + abc)']'
Step 3: f = [(a'b'c')' (a'b'c)' (a'bc)' (ab'c')' (abc)']'
```

# CHECK POINT

As a checkpoint of your understanding, please pause the video and make sure you can do the following:

- For the truth table on slide 3, by hand write the AND/OR sum-of-product form of function f, then convert that function into the NAND/NAND form.

If you have any difficulties, please review the lecture video before continuing.

# f', Inverse of Function f

- Consider the inverse of example function f(a,b,c)
- We can determine the SOP form of function f'
    - f(a,b,c)'=a'bc'+ab'c+abc'

| | a | b | c | f | f ' | minterm |
|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 1 | 0 | a'b'c' |
| | 0 | 0 | 1 | 1 | 0 | a'b'c |
| ⇒ | 0 | 1 | 0 | 0 | 1 | a'bc' |
| | 0 | 1 | 1 | 1 | 0 | a'bc |
| | 1 | 0 | 0 | 1 | 0 | ab'c' |
| ⇒ | 1 | 0 | 1 | 0 | 1 | ab'c |
| ⇒ | 1 | 1 | 0 | 0 | 1 | abc' |
| | 1 | 1 | 1 | 1 | 0 | abc |

# Product-of-Sums (POS) Form

- Function f' can be manipulated to yield the product-of-sums form for function f using OR and AND operations

```
Step 1: f' = a'bc' + ab'c + abc'
Step 2: f = (a'bc' + ab'c + abc')'
Step 3: f = (a'bc')' (ab'c)' (abc')'
Step 4: f = (a''+b'+c'') (a'+b''+c') (a'+b'+c'')
Step 5: f = (a+b'+c) (a'+b+c') (a'+b'+c)
```

# Canonical POS Form (1)

- A “maxterm” can be associated with each row of the truth table
- Maxterm is an OR expression
- Maxterm is FALSE for the input combination, but TRUE for all others

| | a | b | c | f | maxterm |
|---|---|---|---|---|---|
| | 0 | 0 | 0 | 1 | a+b+c |
| | 0 | 0 | 1 | 1 | a+b+c’ |
| ⇒ | 0 | 1 | 0 | 0 | a+b’+c |
| | 0 | 1 | 1 | 1 | a+b’+c’ |
| | 1 | 0 | 0 | 1 | a’+b+c |
| ⇒ | 1 | 0 | 1 | 0 | a’+b+c’ |
| ⇒ | 1 | 1 | 0 | 0 | a’+b’+c |
| | 1 | 1 | 1 | 1 | a’+b’+c’ |

# Canonical POS Form (2)

- Function f(a,b,c) can be written as the product of the maxterms where f = 0
  - f = (a+b'+c) (a'+b+c') (a'+b'+c)
  - This is the canonical product-of-sums form

| | a | b | c | f | maxterm |
|---|---|---|---|---|---|
| | 0 | 0 | 0 | 1 | a+b+c |
| | 0 | 0 | 1 | 1 | a+b+c' |
| ⇒ | 0 | 1 | 0 | 0 | a+b'+c |
| | 0 | 1 | 1 | 1 | a+b'+c' |
| | 1 | 0 | 0 | 1 | a'+b+c |
| ⇒ | 1 | 0 | 1 | 0 | a'+b+c' |
| ⇒ | 1 | 1 | 0 | 0 | a'+b'+c |
| | 1 | 1 | 1 | 1 | a'+b'+c' |

# NOR Operation

- NOR function is realized by applying OR and then NOT
- Output is TRUE (“1”) if and only if all inputs are FALSE (“0”)

| a | b | $F=\overline{a+b}$ |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

$$F = (a+b)' = a+b$$

# NOR/NOR Form

- Example function f(a,b,c) can also be implemented in NOR/NOR form using DeMorgan’s Theorem, (xy)’ = x’+y’
- So, f(a,b,c) can be realized using only NOR operations

```
Step 1: f = (a+b′+c) (a′+b+c′) (a′+b′+c)
Step 2: f = {[(a+b′+c) (a′+b+c′) (a′+b′+c)]′}′
Step 3: f = {(a+b′+c)′ + (a′+b+c′)′ + (a′+b′+c)′}′
```

# CHECK POINT

As a checkpoint of your understanding, please pause the video and make sure you can:

- On slide 10 write out by hand the function f in sum-of-product form, then work through the steps to convert f into the product-of-sums form, then work through the steps to represent f in the NOR/NOR form.

If you have any difficulties, please review the lecture video before continuing.

# Summary

- Any logic expression can represented and implemented as a two-level logic function
- Two basic forms of representation based on OR and AND operations
  - Sum-of-products: AND/OR
  - Product-of-sums: OR/AND
- Alternate forms based on NAND gates or on NOR operations
  - Sum-of-products: NAND/NAND
  - Product-of-sums: NOR/NOR

MODULE 3: Boolean Algebra and Digital Logic

# Lecture 3.2
# Representing Boolean Functions


Prepared By:

- Scott F. Midkiff, PhD
- Luiz A. DaSilva, PhD
- Kendall E. Giles, PhD

Electrical and Computer Engineering

Virginia Tech