Unit 4

Applications of Boolean Algebra

Logic Circuits (Spring 2022)

Conversion to Boolean Equations

- The first step in designing a logic circuit
- Conversion procedure
  - Break down each sentence into phrases and associate a Boolean variable with each phrase
  - If a phrase can have a value of true or false, we can represent that phrase by a Boolean variable
  - Phrases can be either true or false, or have no truth value

Conversion to Boolean Equations

- Example

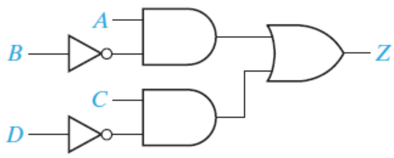
> The alarm will ring if and only if
> the alarm switch is turned on and the door is not closed, or
> it is after 6 p.m. and the window is not closed.

- Conversion

$\underbrace{\text{The alarm will ring}}_{Z}$ iff $\underbrace{\text{the alarm switch is on}}_{A}$ and

$\underbrace{\text{the door is not closed}}_{B'}$ *or* $\underbrace{\text{it is after 6 P.M.}}_{C}$ and

$\underbrace{\text{the window is not closed.}}_{D'}$



- Logic expression $Z = AB' + CD'$

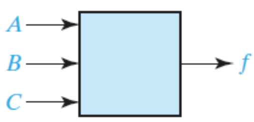
Logic Design With a Truth Table

- Example

> Three binary digits, A, B, C are given to represent a binary number N
> The output $f = 1$ iff $N \geq 011_2$ and $f = 0$ iff $N < 011_2$

- Truth table representation

A	B	C	f	f'
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

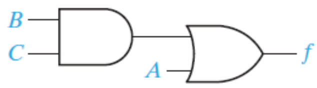


Logic Design With a Truth Table

- Building a SOP expression
  - OR-ing the terms that yield value 1
    $f = A'BC + AB'C' + AB'C + ABC' + ABC$
  - The expression can be simplified
    $f = A'BC + AB' + AB = A'BC + A = A + BC$

A	B	C	f	f'
0	0	0	0	1
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

- Building a POS expression
  - AND-ing the terms that yield value 0
    $f = (A + B + C)(A + B + C')(A + B' + C)$
  - The expression can be simplified
    $f = (A + B)(A + B' + C) = A + BC$



- Building a POS expression using DeMorgan's theorem
  - $f' = A'B'C' + A'B'C + A'BC'$
  - $f = (A + B + C)(A + B + C')(A + B' + C)$

Minterm Expansions

- *Minterm* of *n* variables
  - A *product* of *n* literals
  - Each variable appears exactly once in either true or complemented form, but not both
- Minterms are often written in abbreviated form
  - $A'B'C'$ is designated m_0
  - $A'B'C$ is designated m_1
- Minterm m_i corresponds to row i of the truth table
- *Minterm expansion*
  - Represent a function as a sum of minterms
  - Standard SOP(sum of products)
- Example

$$f(A, B, C) = m_3 + m_4 + m_5 + m_6 + m_7$$
$$f(A, B, C) = \Sigma m(3, 4, 5, 6, 7)$$

Maxterm Expansions

- *Maxterm* of *n* variables
  - A *sum* of *n* literals
  - Each variable appears exactly once in either true or complemented form, but not both
- Minterms are often written in abbreviated form
  - $A+B+C$ is designated M_0
  - $A+B+C'$ is designated M_1
- A maxterm is the complement of the corresponding minterm
- *Maxterm expansion*
  - Represent a function as a product of maxterms
  - Standard POS(products of sums)
- Example

$$f(A, B, C) = M_0 M_1 M_2$$

$$f(A, B, C) = \Pi\, M(0, 1, 2)$$

Minterm and Maxterm

Row No.	A B C	Minterms	Maxterms
0	0 0 0	$A'B'C' = m_0$	$A + B + C = M_0$
1	0 0 1	$A'B'C = m_1$	$A + B + C' = M_1$
2	0 1 0	$A'BC' = m_2$	$A + B' + C = M_2$
3	0 1 1	$A'BC = m_3$	$A + B' + C' = M_3$
4	1 0 0	$AB'C' = m_4$	$A' + B + C = M_4$
5	1 0 1	$AB'C = m_5$	$A' + B + C' = M_5$
6	1 1 0	$ABC' = m_6$	$A' + B' + C = M_6$
7	1 1 1	$ABC = m_7$	$A' + B' + C' = M_7$

Minterm Expansions of the Complement

- Minterm representation of f

  $f(A, B, C) = m_3 + m_4 + m_5 + m_6 + m_7$

  $f(A, B, C) = \Sigma m(3, 4, 5, 6, 7)$

- Maxterm representation of f

  $f(A, B, C) = M_0 M_1 M_2$

  $f(A, B, C) = \Pi\, M(0, 1, 2)$

A B C	f	f′
0 0 0	0	1
0 0 1	0	1
0 1 0	0	1
0 1 1	1	0
1 0 0	1	0
1 0 1	1	0
1 1 0	1	0
1 1 1	1	0

- Minterm expansions for the complement of f

  $f' = m_0 + m_1 + m_2 = \Sigma\, m(0, 1, 2)$

- Applying DeMorgan's theorem to derive the complement of f

  $f' = (M_0 M_1 M_2)' = M_0' + M_1' + M_2' = m_0 + m_1 + m_2$

Minterm Expansions: Example

Example

Find the minterm expansion of $f(a, b, c, d) = a'(b' + d) + acd'$.

$$\begin{aligned} f &= a'b' + a'd + acd' \\ &= a'b'(c + c')(d + d') + a'd(b + b')(c + c') + acd'(b + b') \\ &= a'b'c'd' + a'b'c'd + a'b'cd' + a'b'cd + \cancel{a'b'c'd} + \cancel{a'b'cd} \\ &\quad + a'bc'd + a'bcd + abcd' + ab'cd' \end{aligned}$$

$$f = a'b'c'd' + a'b'c'd + a'b'cd' + a'b'cd + a'bc'd + a'bcd + abcd' + ab'cd'$$

0000 0001 0010 0011 0101 0111 1110 1010

$$f = \Sigma\, m(0, 1, 2, 3, 5, 7, 10, 14)$$

The maxterm expansion for f can then be obtained by listing the decimal integers (in the range 0 to 15) which do not correspond to minterms of f:

$$f = \Pi\, M(4, 6, 8, 9, 11, 12, 13, 15)$$

Utilizing Minterm Expansions: Example

Example

Show that $a'c + b'c' + ab = a'b' + bc + ac'$.

We will find the minterm expansion of each side by supplying the missing variables. For the left side,

$$\begin{aligned} &a'c(b + b') + b'c'(a + a') + ab(c + c') \\ &\quad = a'bc + a'b'c + ab'c' + a'b'c' + abc + abc' \\ &\quad = m_3 + m_1 + m_4 + m_0 + m_7 + m_6 \end{aligned}$$

For the right side,

$$\begin{aligned} &a'b'(c + c') + bc(a + a') + ac'(b + b') \\ &\quad = a'b'c + a'b'c + abc + a'bc + abc' + ab'c' \\ &\quad = m_1 + m_0 + m_7 + m_3 + m_6 + m_4 \end{aligned}$$

Because the two minterm expansions are the same, the equation is valid.

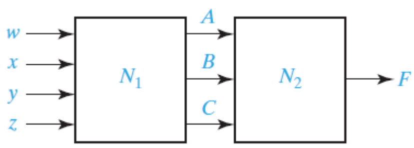
Review on Minterm/Maxterm Expansion

DESIRED FORM

GIVEN FORM	Minterm Expansion of f	Maxterm Expansion of f	Minterm Expansion of f'	Maxterm Expansion of f'
$f = \Sigma\, m(3, 4, 5, 6, 7)$	______	$\Pi\, M(0, 1, 2)$	$\Sigma\, m(0, 1, 2)$	$\Pi\, M(3, 4, 5, 6, 7)$
$f = \Pi\, M(0, 1, 2)$	$\Sigma\, m(3, 4, 5, 6, 7)$	______	$\Sigma\, m(0, 1, 2)$	$\Pi\, M(3, 4, 5, 6, 7)$

Incompletely Specified Functions

- "*Incompletely specified*"
  - Certain combinations of inputs will never occur
  - We "don't care" what the value is
  - The function is then considered incompletely specified
- Example



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

- Possible expressions on incompletely specified functions

$F = A'B'C' + A'BC + ABC = A'B'C' + BC$

$F = A'B'C' + A'B'C + A'BC + ABC = A'B' + BC$

$F = A'B'C' + A'B'C + A'BC + ABC' + ABC = A'B' + BC + AB$

Incompletely Specified Functions

- Minterm representation

  $F = \Sigma\, m(0, 3, 7) + \Sigma\, d(1, 6)$

- Maxterm representation

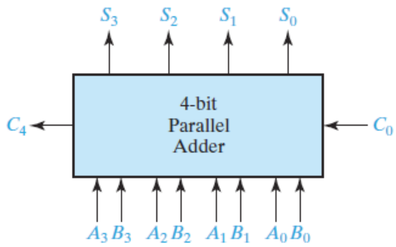
  $F = \Pi\, M(2, 4, 5) \cdot \Pi\, D(1, 6)$

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

- *d* and *D* specify "don't cares" in minterm and maxterm expansions

4-bit (Binary) Adder

- 4-bit parallel adder
  - Adds two 4-bit binary numbers and a carry input
  - Produces a 4-bit sum and a carry output

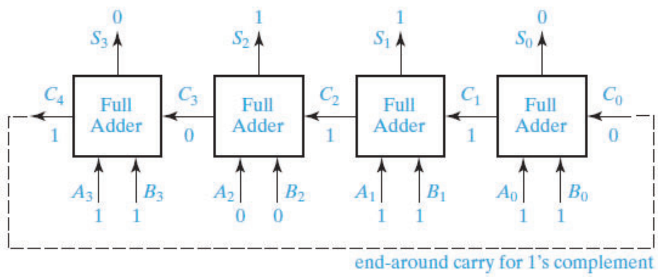


- Two design methods
  - To construct a truth table with 9 inputs and 5 outputs ⇒ very difficult
  - To design a logic module (*full adder*) to add two bits and a carry, and connect four modules together (*hierarchical design*)

How the 4-bit Adder Works

- Example case: 1011 + 1011

```
  10110   (carries)
   1011
 + 1011
 ------
  10110
```



1-bit (Binary) Adder: Half Adder

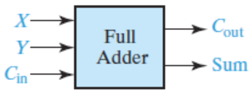
- A simple binary adder that adds two 1-bit binary numbers
  - Produces a 2-bit sum

A	B	X	Y
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$X = AB$ and $Y = A'B + AB' = A \oplus B$

- May need another half adder to perform a bit addition completely

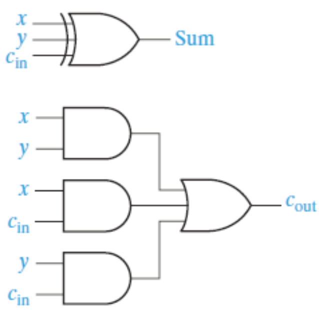
1-bit (Binary) Adder: Full Adder



X	Y	C_{in}	C_{out}	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$\begin{aligned} Sum &= X'Y'C_{in} + X'YC'_{in} + XY'C'_{in} + XYC_{in} \\ &= X'(Y'C_{in} + YC'_{in}) + X(Y'C'_{in} + YC_{in}) \\ &= X'(Y \oplus C_{in}) + X(Y \oplus C_{in})' = X \oplus Y \oplus C_{in} \end{aligned}$$

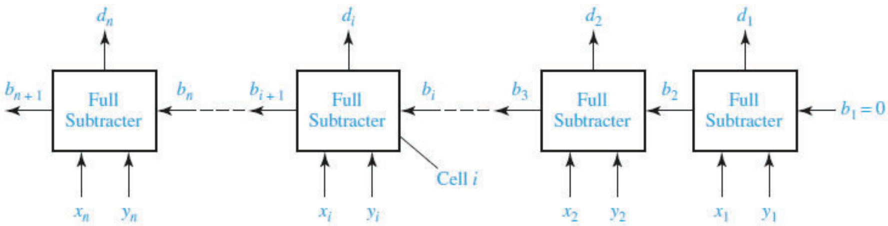
$$\begin{aligned} C_{out} &= X'YC_{in} + XY'C_{in} + XYC'_{in} + XYC_{in} \\ &= (X'YC_{in} + XYC_{in}) + (XY'C_{in} + XYC_{in}) + (XYC'_{in} + XYC_{in}) \\ &= YC_{in} + XC_{in} + XY \end{aligned}$$



Binary Subtracter

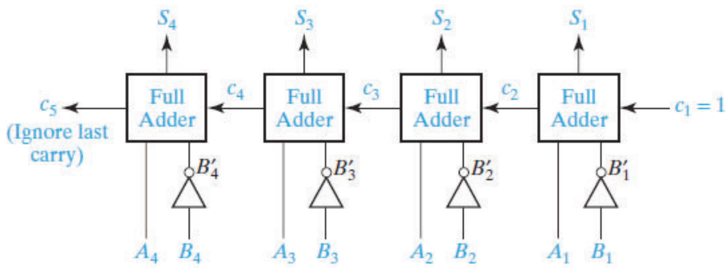
- 4-bit parallel subtracter
  - Subtracts Y from X
  - Produces a 4-bit difference and a borrow output
- 1-bit subtracter
  - Inputs: x_i, y_i, and b_i
  - Outputs: d_i, and b_{i+1}

x_i	y_i	b_i	b_{i+1}	d_i
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1



Binary Subtracter

- Circuit to perform A−B using 2's complement



Example

$A = 0110$ (+6)
$B = 0011$ (+3)

The adder output is

$$\begin{array}{rrl} & 0110 & (+6) \\ & +1100 & (\text{1's complement of 3}) \\ & +\quad 1 & (\text{first carry input}) \\ \hline (1) & 0011 & = 3 = 6 - 3 \end{array}$$