Chapter 1

DATA REPRESENTATION

Analog vs. Digital

Analog quantities

- can vary over a continuous range of values.
- Examples: voltage, room thermostat

Digital quantities

- represented by symbols called digits.
- Example: digital watch

Digital Number Systems

- Many number systems are in use in digital technology.
- The most common are the *decimal, binary, octal, and hexadecimal systems.*

Conversion: Base-r to Decimal

Procedure

- **Step 1:**

  Multiply each coefficient with the corresponding power of r.

- **Step 2:**

  Get the sum.

Conversion: Decimal to Base-r

Procedure

- **Step 1:**
  Separate integer from fraction
- **Step 2:**
  Convert integer to base-r
- **Step 3:**
  Convert fraction to base-r

Integer to base-r

- Divide integer by r
- Accumulate remainders

Fraction to base-r

- Multiply fraction by r
- Accumulate integers

Conversion: Binary to octal or hexadecimal

Binary To Octal

Procedure:

- Partition binary number into groups of 3 digits

Conversion: Binary to octal or hexadecimal

Binary To Hexadecimal

Procedure:

- Partition binary number into groups of 4 digits

Conversion: Octal or hexadecimal to binary

Octal To Binary

Procedure:

- Each octal digit is converted to its 3-digit binary equivalent

Conversion: Octal or hexadecimal to binary

Hexadecimal to Binary

Procedure:

- Each hexadecimal digit is converted to its 4-digit binary equivalent

Fixed-Point Representation

Unsigned Number

- leftmost bit is the most significant bit
- Example:
  - 01001 = 9
  - 11001 = 25

Signed Number

- leftmost bit represents the sign
- Example:
  - 01001 = +9
  - 11001 = - 9

Systems Used to Represent Negative Numbers

Signed-Magnitude representation

- A number consists of a magnitude and a symbol indicating whether the magnitude is positive or negative.

Examples:

$+85 = 01010101_2$ $-85 = 11010101_2$

$+127 = 01111111_2$ $-127 = 11111111_2$

Systems Used to Represent Negative Numbers

Signed-Complement System

- This system negates a number by taking its complement as defined by the system.
- Types of complements:
  - Radix-complement
  - Diminished Radix-complement

Chapter 2

COMPUTER ARITHMETIC

Arithmetic in various Number Systems

- Addition of numbers in any number system
  - Add numbers starting at the least significant digit.
  - Perform addition on numbers of the same number base.
- Subtraction of numbers
  - Must use complements

Binary Addition

- To add binary numbers: (X + Y)
  - Get the SCR of the negative numbers
  - Add the two numbers
  - If the SCR used is:
    - 2's C: Discard end carry
    - 1's C: Add the end carry to the sum

Binary Subtraction

- To subtract binary numbers: (X – Y)
  - Take the complement of the subtrahend.
  - Then, add the two numbers.
  - If the complement used is:
    - 1's C: Add the end carry to the sum
    - 2's C: Discard end carry

(X – Y) >>> X + (complement of Y)

Error Detection

- Overflow
  - occurs when an arithmetic operation yields a result that is greater than the range's positive limit

Error Detection

- Underflow
  - occurs when an arithmetic operation yields a result that is lesser than the range's negative limit

BCD Addition

- Sum less than or equal to 9
  - Normal binary addition
- Sum greater than 9
  - Add the codes
  - Add a correction value of 0110 to any sum

Chapter 3

BOOLEAN ALGEBRA, LOGIC FUNCTIONS and LOGIC GATES

Boolean Operations

- AND
  - represented by a dot or the absence of an operator
  - 0 dominates

x	y	xy
0	0	0
0	1	0
1	0	0
1	1	1

Boolean Operations

- OR
  - represented by a plus sign
  - 1 dominates

x	y	x+y
0	0	0
0	1	1
1	0	1
1	1	1

Boolean Operations

- NOT
  - represented by a prime
  - inversion or complementation

x	x'
0	1
1	0

Boolean Operations on constants

AND	OR	NOT
$0 \cdot 0 = 0$	$0 + 0 = 0$	$0' = 1$
$0 \cdot 1 = 0$	$0 + 1 = 1$	$1' = 0$
$1 \cdot 0 = 0$	$1 + 0 = 1$	
$1 \cdot 1 = 1$	$1 + 1 = 1$	

Boolean Operations on one variable

AND	OR	NOT
$A \cdot 0 = 0$	$A + 0 = A$	$A'' = A$
$A \cdot 1 = A$	$A + 1 = 1$	
$A \cdot A = A$	$A + A = A$	
$A \cdot A' = 0$	$A + A' = 1$	

Boolean Operations On Two or More Variables

- Commutative laws

  $A + B = B + A$

  $AB = BA$

- Associative laws

  $A+(B+C) = (A+B)+C$

  $A(BC) = (AB)C$

- Distributive laws

  $A(B + C) = AB + AC$

  $A+(BC) = (A+B)(A+C)$

- De Morgan's laws

  $(A + B)' = A'B'$

  $(AB)' = A' + B'$

- Laws of Absorption

  $A + AB = A$

  $A(A+B) = A$

Boolean Functions

- Boolean functions are expressions formed with binary variables and boolean operators
- Representations of boolean functions:
  - algebraic expression
  - truth table

Minterms and Maxterms for 3 variables

			MINTERM		MAXTERM	
x	y	z	Term	Designation	Term	Designation
0	0	0	x’y’z’	m0	x+y+z	M0
0	0	1	x’y’z	m1	x+y+z’	M1
0	1	0	x’yz’	m2	x+y’+z	M2
0	1	1	x’yz	m3	x+y’+z’	M3
1	0	0	xy’z’	m4	x’+y+z	M4
1	0	1	xy’z	m5	x’+y+z’	M5
1	1	0	xyz’	m6	x’+y’+z	M6
1	1	1	xyz	m7	x’+y’+z’	M7

Forms of Boolean Functions

- Canonical Form
  - Sum of minterms

  $F(x,y,z) =$

  $xyz' + x'yz$

  - Product of maxterms

  $F(x,y,z) =$

  $(x'+y'+z)(x+y+z')$

- Standard Form
  - Sum of products

  $F(x,y,z) = xz'+y$

  - Product of sums

  $F(x,y,z) = (x+y')z$

Chapter 4

SIMPLIFICATION of LOGIC CIRCUITS

Ways to simplify Boolean functions

- Boolean Algebra
- Graphical method (Karnaugh Map)
- Tabular method (Quine-McCluskey)

Simplification: Boolean Algebra

Simplify $x'y' + xyz + x'y$

Simplification: Boolean Algebra

Simplify $x'y' + xyz + x'y$

$= x'(y' + y) + xyz$ Comm / Dist.

Simplification: Boolean Algebra

Simplify $x'y' + xyz + x'y$

$= x'(y' + y) + xyz$ Comm / Dist.

$= x' + xyz$ Inv / Iden

Simplification: Boolean Algebra

Simplify $x'y' + xyz + x'y$

$= x'(y' + y) + xyz$ Comm / Dist.

$= x' + xyz$ Inv / Iden

$= (x' + x)(x' + yz)$ Dist.

Simplification: Boolean Algebra

Simplify $x'y' + xyz + x'y$

$= x'(y' + y) + xyz$	Comm / Dist.
$= x' + xyz$	Inv / Iden
$= (x' + x)(x' + yz)$	Dist.
$= (x' + yz)$	Inv / Iden

Simplification: Graphical method

- Karnaugh map (K-map)
  - alternate way of representing Boolean functions
  - a graphical tool for assisting in the general simplification procedure
  - a simpler way to handle most jobs of manipulating logic functions

General Steps of K-Map Simplification

- Express function in canonical form
- Map expression on a K-Map
- Group 1’s or 0's
- Determine the minimum expression

Simplification Example

- Simplify B'C'D' + A'BC'D' + A'B'CD + AB'CD + A'B'CD' + A'BCD' + ABCD' + AB'CD' + ABC'D'

Simplification Example

- Simplify B'C'D' + A'BC'D' + A'B'CD + AB'CD + A'B'CD' + A'BCD' + ABCD' + AB'CD' + ABC'D'

AB \ CD	0 0	0 1	1 1	10
0 0				
0 1				
1 1				
1 0				

Simplification Example

- Simplify B’C’D’ + A’BC’D’ + A’B’CD + AB’CD + A’B’CD’ + A’BCD’ + ABCD' + AB’CD’ + ABC'D'

AB \ CD	0 0	0 1	1 1	10
0 0	1		1	1
0 1	1			1
1 1	1			1
1 0	1		1	1

Simplification Example

- Simplify B’C’D’ + A’BC’D’ + A’B’CD + AB’CD + A’B’CD’ + A’BCD’ + ABCD' + AB’CD’ + ABC'D'

AB \ CD	0 0	0 1	1 1	10
0 0	1		1	1
0 1	1			1
1 1	1			1
1 0	1		1	1

Simplification Example

- Simplify B’C’D’ + A’BC’D’ + A’B’CD + AB’CD + A’B’CD’ + A’BCD’ + ABCD' + AB’CD’ + ABC'D'

AB \ CD	0 0	0 1	1 1	10
0 0	1		1	1
0 1	1			1
1 1	1			1
1 0	1		1	1

Simplification Example

- Simplify B’C’D’ + A’BC’D’ + A’B’CD + AB’CD + A’B’CD’ + A’BCD’ + ABCD' + AB’CD’ + ABC'D'

AB \ CD	0 0	0 1	1 1	10
0 0	1		1	1
0 1	1			1
1 1	1			1
1 0	1		1	1

D'

Simplification Example

- Simplify B'C'D' + A'BC'D' + A'B'CD + AB'CD + A'B'CD' + A'BCD' + ABCD' + AB'CD' + ABC'D'

AB \ CD	0 0	0 1	1 1	10
0 0	1		1	1
0 1	1			1
1 1	1			1
1 0	1		1	1

B'C

D'

Simplification Example

- Simplify B'C'D' + A'BC'D' + A'B'CD + AB'CD + A'B'CD' + A'BCD' + ABCD' + AB'CD' + ABC'D'

= B'C + D'

AB \ CD	0 0	0 1	1 1	10
0 0	1		1	1
0 1	1			1
1 1	1			1
1 0	1		1	1

B'C

D'

Simplification Example

- Simplify (B+C+D) (A+B+C'+D) (A'+B+C+D') (A+B'+C+D) (A'+B'+C+D)

AB \ CD	0 0	0 1	1 1	10
0 0				
0 1				
1 1				
1 0				

Simplification Example

- Simplify (B+C+D) (A+B+C’+D) (A’+B+C+D’) (A+B’+C+D) (A’+B’+C+D)

AB \ CD	0 0	0 1	1 1	10
0 0	0			
0 1				
1 1				
1 0	0			

Simplification Example

- Simplify (B+C+D) (A+B+C’+D) (A’+B+C+D’) (A+B’+C+D) (A’+B’+C+D)

AB \ CD	0 0	0 1	1 1	10
0 0	0			0
0 1				
1 1				
1 0	0			

Simplification Example

- Simplify (B+C+D) (A+B+C'+D) (A'+B+C+D') (A+B'+C+D) (A'+B'+C+D)

AB \ CD	0 0	0 1	1 1	10
0 0	0			0
0 1				
1 1				
1 0	0	0		

Simplification Example

- Simplify (B+C+D) (A+B+C'+D) (A'+B+C+D') (A+B'+C+D) (A'+B'+C+D)

AB \ CD	0 0	0 1	1 1	10
0 0	0			0
0 1	0			
1 1				
1 0	0	0		

Simplification Example

- Simplify (B+C+D) (A+B+C'+D) (A'+B+C+D') (A+B'+C+D) (A'+B'+C+D)

AB \ CD	0 0	0 1	1 1	10
0 0	0			0
0 1	0			
1 1	0			
1 0	0	0		

Simplification Example

- Simplify (B+C+D) (A+B+C'+D) (A'+B+C+D') (A+B'+C+D) (A'+B'+C+D)

AB \ CD	0 0	0 1	1 1	10
0 0	0			0
0 1	0			
1 1	0			
1 0	0	0		

Simplification Example

- Simplify (B+C+D) (A+B+C'+D) (A'+B+C+D') (A+B'+C+D) (A'+B'+C+D)

AB \ CD	0 0	0 1	1 1	10
0 0	0			0
0 1	0			
1 1	0			
1 0	0	0		

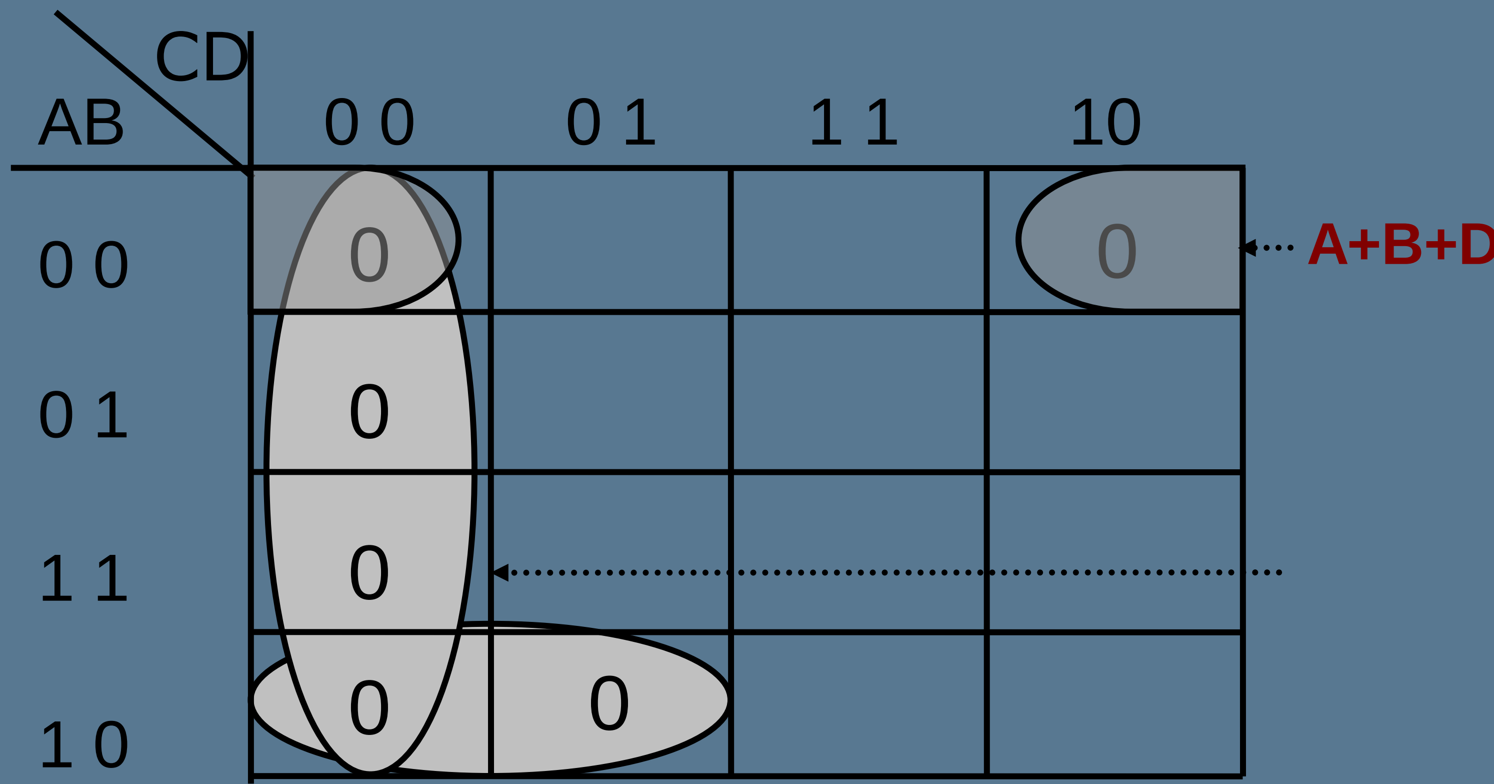
Simplification Example

- Simplify (B+C+D) (A+B+C'+D) (A'+B+C+D') (A+B'+C+D) (A'+B'+C+D)

AB \ CD	0 0	0 1	1 1	10
0 0	0			0
0 1	0			
1 1	0			
1 0	0	0		

Simplification Example

- Simplify (B+C+D) (A+B+C'+D) (A'+B+C+D') (A+B'+C+D) (A'+B'+C+D)



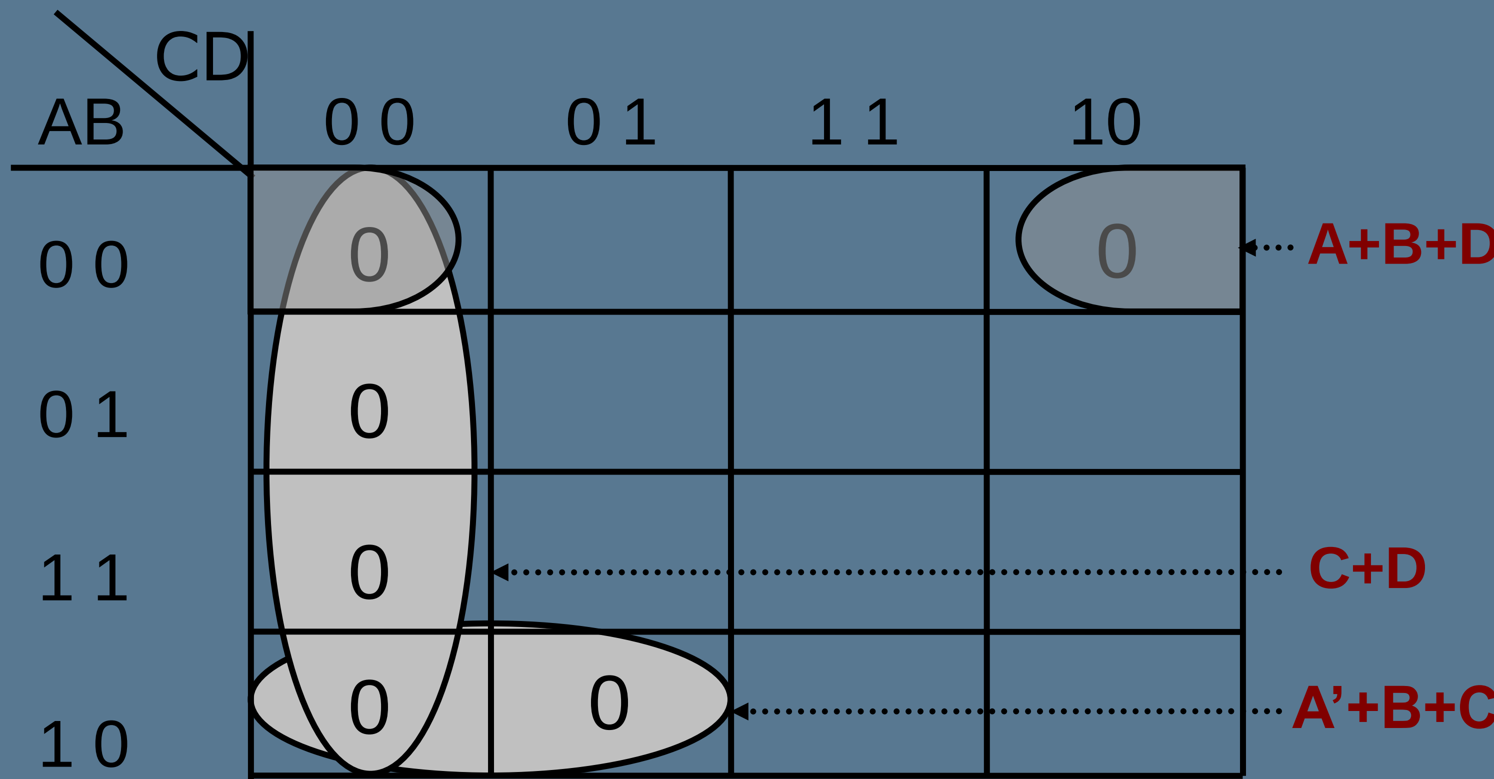
Simplification Example

- Simplify (B+C+D) (A+B+C'+D) (A'+B+C+D') (A+B'+C+D) (A'+B'+C+D)

AB \ CD	00	01	11	10	
00	0			0	A+B+D
01	0				
11	0				C+D
10	0	0			

Simplification Example

- Simplify (B+C+D) (A+B+C'+D) (A'+B+C+D') (A+B'+C+D) (A'+B'+C+D)



Simplification Example

- Simplify (B+C+D) (A+B+C'+D) (A'+B+C+D') (A+B'+C+D) (A'+B'+C+D) **=(A+B+D)(C+D)(A'+B+C)**

AB \ CD	0 0	0 1	1 1	10
0 0	0			0
0 1	0			
1 1	0			
1 0	0	0		

A+B+D

C+D

A'+B+C

Simplification: Quine-McCluskey

- Advantages
  - Specific step by step procedure
  - Can be applied to problems with many variables
  - Suitable for machine computation

Simplification: Quine-McCluskey

- Steps
  - Construct prime implicants table
  - Construct prime implicants chart
  - Select all essential prime implicants
  - Select a minimal cover from the remaining prime implicants

Constructing Prime Implicants Table

- List terms in a column using their binary representation
  - Group terms so that each group contains minterms with the same number of 1's
  - Place groupings which differ by only one literal adjacent to one another

Example

- Simplify $F=\sum m(1,3,7,8,14,15)$

1	0001
3	0011
7	0111
8	1000
14	1110
15	1111

Example

- Simplify $F=\sum m(1,3,7,8,14,15)$

			Column 1	
1	0001		1	0001
3	0011		8	1000
7	0111			
8	1000		3	0011
14	1110			
15	1111		7	0111
			14	1110
			15	1111

Constructing Prime Implicants Table

- Perform exhaustive search for logically adjacent terms between adjacent groups
  - Each term should be checked off
  - Combine each pair of terms into a single term replacing the differing literal with '-'
  - Repeat procedure until no further terms can be created
  - All unchecked terms are prime implicants

Example

- Simplify $F=\sum m(1,3,7,8,14,15)$

		Column 1		Column 2
1	0001	1	0001	
3	0011	8	1000	
7	0111			
8	1000	3	0011	
14	1110			
15	1111	7	0111	
		14	1110	
		15	1111	

Example

- Simplify $F=\sum m(1,3,7,8,14,15)$

			Column 1			Column 2	
1	0001	1	0001	√		1,3	00-1
3	0011	8	1000				
7	0111						
8	1000	3	0011	√			
14	1110						
15	1111	7	0111				
		14	1110				
		15	1111				

Example

- Simplify $F=\sum m(1,3,7,8,14,15)$

				Column 1			Column 2	
1	0001		1	0001	√		1,3	00-1
3	0011		8	1000				
7	0111						3,7	0-11
8	1000		3	0011	√			
14	1110							
15	1111		7	0111	√			
			14	1110				
			15	1111				

Example

- Simplify $F=\sum m(1,3,7,8,14,15)$

		Column 1			Column 2	
1	0001	1	0001	√	1,3	00-1
3	0011	8	1000			
7	0111				3,7	0-11
8	1000	3	0011	√		
14	1110				7,15	-111
15	1111	7	0111	√		
		14	1110			
		15	1111	√		

Example

- Simplify $F=\sum m(1,3,7,8,14,15)$

		Column 1			Column 2	
1	0001	1	0001	√	1,3	00-1
3	0011	8	1000			
7	0111				3,7	0-11
8	1000	3	0011	√		
14	1110				7,15	-111
15	1111	7	0111	√	14,15	111-
		14	1110	√		
		15	1111	√		

Example

- Simplify $F=\sum m(1,3,7,8,14,15)$

		Column 1			Column 2	
1	0001	1	0001	√	1,3	00-1
3	0011	8	1000			
7	0111				3,7	0-11
8	1000	3	0011	√		
14	1110				7,15	-111
15	1111	7	0111	√	14,15	111-
		14	1110	√		
		15	1111	√		

Prime implicants:
AB'C'D'
A'B'D
A'CD
BCD
ABC

Construct Prime Implicants Chart

- Terms are listed horizontally
- Prime implicants are listed vertically
- Place an X whenever a prime implicant covers a minterm

Example

		1	3	7	8	14	15
8	1000						
1,3	00-1						
3,7	0-11						
7,15	-111						
14,15	111-						

Example

		1	3	7	8	14	15
8	1000				X		
1,3	00-1	X	X				
3,7	0-11		X	X			
7,15	-111			X			X
14,15	111-					X	X

Select Essential Prime Implicants

		1	3	7	8	14	15
8	1000				X		
1,3	00-1	X	X				
3,7	0-11		X	X			
7,15	-111			X			X
14,15	111-					X	X

Select Essential Prime Implicants

		1	3	7	8	14	15
8	1000				X		
1,3	00-1	X	X				
3,7	0-11		X	X			
7,15	-111			X			X
14,15	111-					X	X

Select Essential Prime Implicants

		1	3	7	8	14	15
8	1000				X		
1,3	00-1	X	X				
3,7	0-11		X	X			
7,15	-111			X			X
14,15	111-					X	X
		√			√	√	

Select Minimum Cover

		1	3	7	8	14	15
8	1000				X		
1,3	00-1	X	X				
3,7	0-11		X	X			
7,15	-111			X			X
14,15	111-					X	X
		√	√		√	√	√

Minimum Expression

- Essential prime implicants + the prime implicants that cover the columns that were not removed
- Hence
  - $F = AB'C'D' + A'B'D + ABC$

Select Minimum Cover

			1	3	7	8	14	15
	8	1000				X		
	1,3	00-1	X	X				
	3,7	0-11		X	X			
√	7,15	-111			X			X
	14,15	111-					X	X
			√	√		√	√	√

Minimum Expression

- Essential prime implicants + the prime implicants that cover the columns that were not removed
- Hence
  - F = AB'C'D' + A'B'D + ABC +BCD

Converting Between POM & SOM Using K-Map

- (A’+B’+C+D) (A+B’+C+D) (A+B+C+D’) (A+B+C’+D’) (A’+B+C+D’) (A+B+C’+D)

AB \ CD	0 0	0 1	1 1	1 0
0 0				
0 1				
1 1				
1 0				

Converting Between POM & SOM Using K-Map

- (A’+B’+C+D) (A+B’+C+D) (A+B+C+D’) (A+B+C’+D’) (A’+B+C+D’) (A+B+C’+D)

AB \ CD	0 0	0 1	1 1	1 0
0 0		0	0	0
0 1	0			
1 1	0			
1 0		0		

Converting Between POM & SOM Using K-Map

- (A'+B'+C+D) (A+B'+C+D) (A+B+C+D') (A+B+C'+D') (A'+B+C+D') (A+B+C'+D)

AB \ CD	0 0	0 1	1 1	1 0
0 0	1	0	0	0
0 1	0	1	1	1
1 1	0	1	1	1
1 0	1	0	1	1

Converting Between POM & SOM Using K-Map

- (A’+B’+C+D) (A+B’+C+D) (A+B+C+D’) (A+B+C’+D’) (A’+B+C+D’) (A+B+C’+D)

AB \ CD	0 0	0 1	1 1	1 0
0 0	1	0	0	0
0 1	0	1	1	1
1 1	0	1	1	1
1 0	1	0	1	1

Converting Between POM & SOM Using K-Map

- (A'+B'+C+D) (A+B'+C+D) (A+B+C+D') (A+B+C'+D') (A'+B+C+D') (A+B+C'+D)

AB \ CD	0 0	0 1	1 1	1 0
0 0	1	0	0	0
0 1	0	1	1	1
1 1	0	1	1	1
1 0	1	0	1	1

Converting Between POM & SOM Using K-Map

- (A’+B’+C+D) (A+B’+C+D) (A+B+C+D’) (A+B+C’+D’) (A’+B+C+D’) (A+B+C’+D)

AB \ CD	0 0	0 1	1 1	1 0
0 0	1	0	0	0
0 1	0	1	1	1
1 1	0	1	1	1
1 0	1	0	1	1

Converting Between POM & SOM Using K-Map

- (A’+B’+C+D) (A+B’+C+D) (A+B+C+D’) (A+B+C’+D’) (A’+B+C+D’) (A+B+C’+D)

AB \ CD	0 0	0 1	1 1	1 0
0 0	1	0	0	0
0 1	0	1	1	1
1 1	0	1	1	1
1 0	1	0	1	1

Converting Between POM & SOM Using K-Map

- (A'+B'+C+D) (A+B'+C+D) (A+B+C+D') (A+B+C'+D') (A'+B+C+D') (A+B+C'+D)

AB \ CD	0 0	0 1	1 1	1 0
0 0	1	0	0	0
0 1	0	1	1	1
1 1	0	1	1	1
1 0	1	0	1	1

Converting Between POM & SOM Using K-Map

- (A'+B'+C+D) (A+B'+C+D) (A+B+C+D') (A+B+C'+D') (A'+B+C+D') (A+B+C'+D)

AB \ CD	0 0	0 1	1 1	1 0
0 0	1	0	0	0
0 1	0	1	1	1
1 1	0	1	1	1
1 0	1	0	1	1

B'C'D'

Converting Between POM & SOM Using K-Map

- (A’+B’+C+D) (A+B’+C+D) (A+B+C+D’) (A+B+C’+D’) (A’+B+C+D’) (A+B+C’+D)

AB \ CD	0 0	0 1	1 1	1 0
0 0	1	0	0	0
0 1	0	1	1	1
1 1	0	1	1	1
1 0	1	0	1	1

B’C’D’

BD

Converting Between POM & SOM Using K-Map

- (A'+B'+C+D) (A+B'+C+D) (A+B+C+D') (A+B+C'+D') (A'+B+C+D') (A+B+C'+D)

AB \ CD	0 0	0 1	1 1	1 0
0 0	1	0	0	0
0 1	0	1	1	1
1 1	0	1	1	1
1 0	1	0	1	1

B'C'D'

BD

BC

Converting Between POM & SOM Using K-Map

- (A'+B'+C+D) (A+B'+C+D) (A+B+C+D') (A+B+C'+D') (A'+B+C+D') (A+B+C'+D)

AB \ CD	0 0	0 1	1 1	1 0
0 0	1	0	0	0
0 1	0	1	1	1
1 1	0	1	1	1
1 0	1	0	1	1

B'C'D'

BD

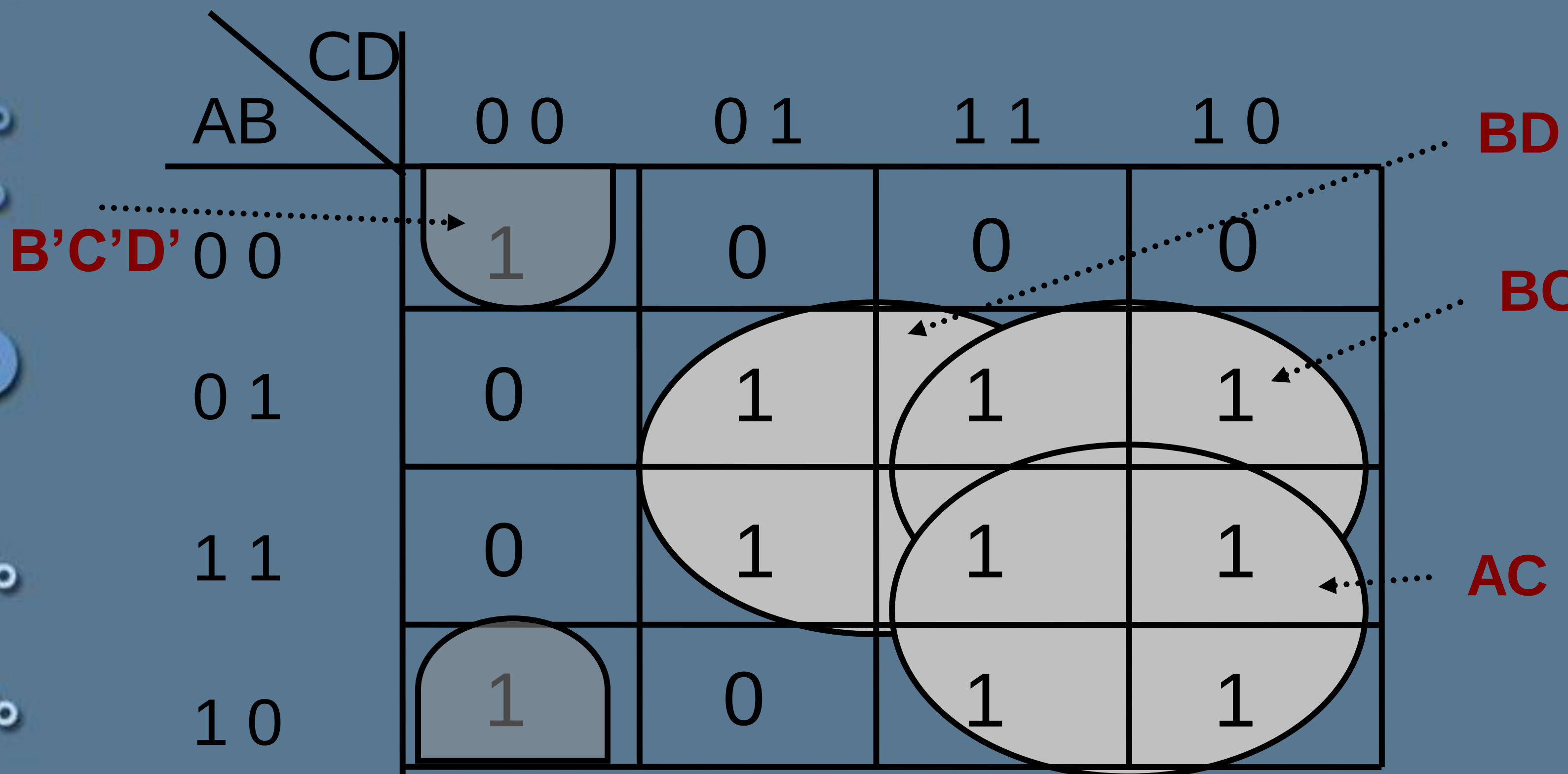
BC

AC

Converting Between POM & SOM Using K-Map

= B’C’D’ + BD + BC + AC

- (A’+B’+C+D) (A+B’+C+D) (A+B+C+D’) (A+B+C’+D’) (A’+B+C+D’) (A+B+C’+D)



Don’t Care conditions

- The unspecified minterms (maxterms) of an incompletely specified function
- An **X** inside a map represents a don’t care condition

Don’t Care conditions

- There are 2 cases when this occurs.
  - The input combination never occurs
    - E.g. The BCD code does not use the 6 remaining codes.
  - The input combinations are expected to occur, but we do not care what the outputs are

Representation of Don't Cares

- e.g:

  $F(W,X,Y,Z) = \sum m(0,1,2,4,6,7,8,10)$

  $d(W,X,Y,Z) = \sum d(12,13,14,15)$

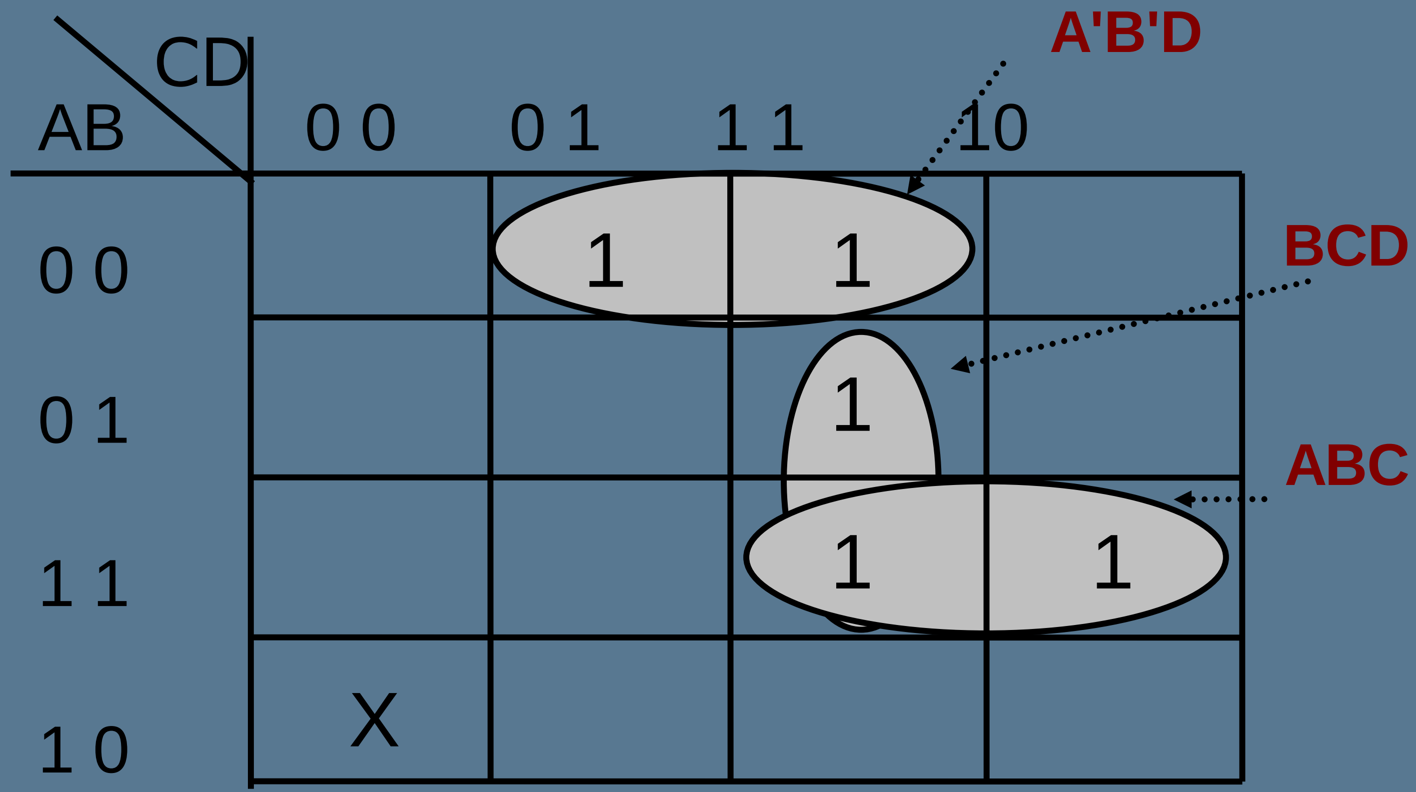
- It could also be represented as:

  $F(W,X,Y,Z) = \sum m(0,1,2,4,6,7,8,10)$

  $+ \sum d\ (12,13,14,15)$

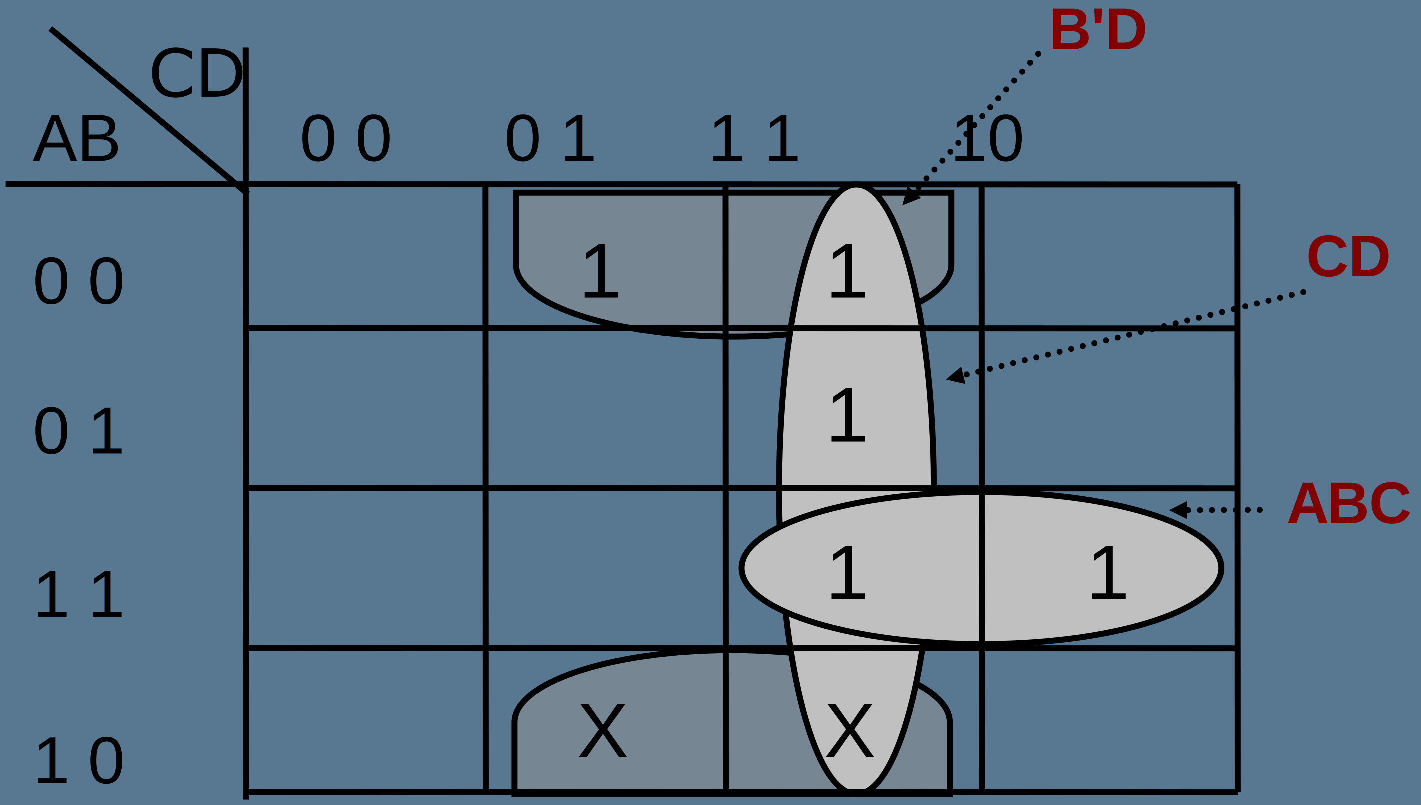
Example

- Simplify $F = \sum m(1,3,7,14,15) + d(8)$



Example

- Simplify $F=\sum m(1,3,7,14,15) + d(9,11)$



Example

- F (A,B,C,D) = $\sum m(0,2,4,8,9,12)$ + $\sum d\ (6,13,15)$

AB \ CD	0 0	0 1	1 1	1 0
0 0				
0 1				
1 1				
1 0				

Example

- $F(A,B,C,D) = \sum m(0,2,4,8,9,12) + \sum d(6,13,15)$

AB \ CD	0 0	0 1	1 1	1 0
0 0	1			1
0 1	1			
1 1	1			
1 0	1	1		

Example

- $F(A,B,C,D) = \sum m(0,2,4,8,9,12) + \sum d(6,13,15)$

AB \ CD	00	01	11	10
00	1			1
01	1			X
11	1	X	X	
10	1	1		

Example

- F (A,B,C,D) = $\sum m(0,2,4,8,9,12) +$
  $\sum d\ (6,13,15)$

AB \ CD	0 0	0 1	1 1	1 0
0 0	1			1
0 1	1			X
1 1	1	X	X	
1 0	1	1		

Example

- $F(A,B,C,D) = \sum m(0,2,4,8,9,12) + \sum d(6,13,15)$

AB \ CD	0 0	0 1	1 1	1 0
0 0	1			1
0 1	1			X
1 1	1	X	X	
1 0	1	1		

Example

- F (A,B,C,D) = $\sum m(0,2,4,8,9,12) +$
  $\sum d\ (6,13,15)$

AB \ CD	0 0	0 1	1 1	1 0
0 0	1			1
0 1	1			X
1 1	1	X	X	
1 0	1	1		

Example

- F (A,B,C,D) = $\sum m(0,2,4,8,9,12) +$
  $\sum d\ (6,13,15)$

AB \ CD	0 0	0 1	1 1	1 0
0 0	1			1
0 1	1			X
1 1	1	X	X	
1 0	1	1		

Example

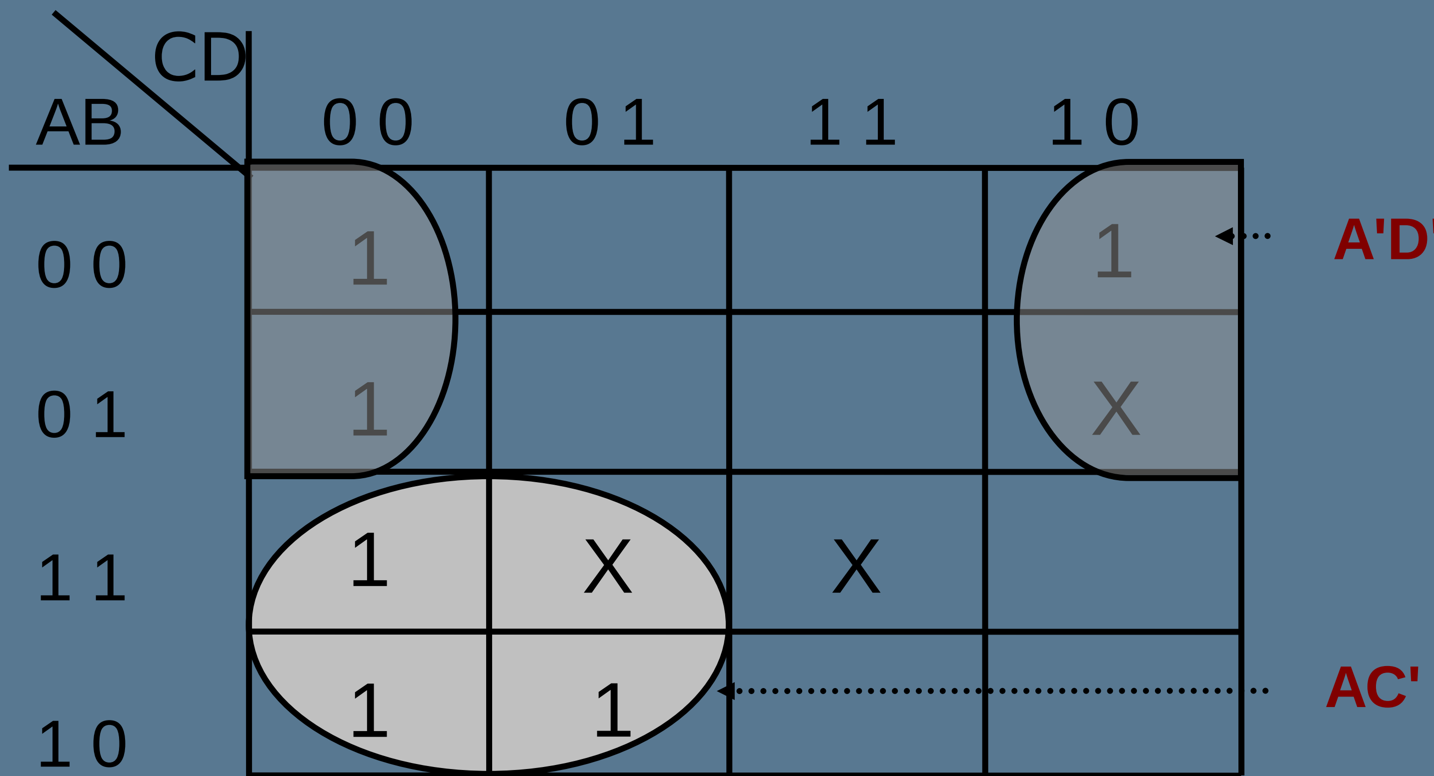
- F (A,B,C,D) = ∑m(0,2,4,8,9,12) +
  ∑d (6,13,15)

AB \ CD	0 0	0 1	1 1	1 0
0 0	1			1
0 1	1			X
1 1	1	X	X	
1 0	1	1		

A'D'

Example

- F (A,B,C,D) = $\sum m(0,2,4,8,9,12) +$
  $\sum d\ (6,13,15)$



Example

- F (A,B,C,D) = $\sum$m(0,2,4,8,9,12) + $\sum$d (6,13,15)

=A'D' + AC'

AB \ CD	0 0	0 1	1 1	1 0
0 0	1			1
0 1	1			X
1 1	1	X	X	
1 0	1	1		

A'D'

AC'