Lect 08

K-Maps and QM Methods

CS221: Digital Design

Dr. A. Sahu
Dept of Comp. Sc. & Engg.
Indian Institute of Technology Guwahati

1

Outline

- Canonical form of Function
  - SOP and POS
- Gray Code and Hamming Distances
- K-Maps : Graphical
- QM Methods : Tabular

2

Converting among Representations

Q: Convert to equation

a	b	F	Term
0	0	**1**	**a'b'**
0	1	**1**	**a'b**
1	0	**0**	
1	1	**0**	

F = a'b' + a'b

Expressing Functions with Minterms

- Boolean function can be expressed algebraically from a give truth table
  - Forming sum of ALL the minterms that produce 1 in the function

Example : Consider the function defined by the truth table

F(X,Y,Z)= X'Y'Z' + X'YZ' + XY'Z + XYZ

$= m_0 + m_2 + m_5 + m_7$

$= \sum m(0, 2, 5, 7)$

X	Y	Z	m	F
0	**0**	**0**	m_0	**1**
0	0	1	m_1	0
0	**1**	**0**	m_2	**1**
0	1	1	m_3	0
1	0	0	m_4	0
1	**0**	**1**	m_5	**1**
1	1	0	m_6	0
1	**1**	**1**	m_7	**1**

Expressing Functions with Maxterms

- Boolean function : Expressed algebraically from a give truth table
- By forming logical product (AND) of ALL the maxterms that produce 0 in the function

Example:

Consider the function defined by the truth table

$F(X,Y,Z) = \Pi M(1,3,4,6)$

Applying DeMorgan

$F' = m_1 + m_3 + m_4 + m_6 = \sum m(1,3,4,6)$

$F = F'' = [m_1 + m_3 + m_4 + m_6]'$

$= m_1'.m_3'.m_4'.m_6'$

$= M_1.M_3.M_4.M_6$

$= \Pi M(1,3,4,6)$

X	Y	Z	M	F	F'
0	0	0	M_0	1	0
0	**0**	**1**	M_1	**0**	**1**
0	1	0	M_2	1	0
0	**1**	**1**	M_3	**0**	**1**
1	**0**	**0**	M_4	**0**	**1**
1	0	1	M_5	1	0
1	**1**	**0**	M_6	**0**	**1**
1	1	1	M_7	1	0

Note the indices in this list are those that are missing from the previous list in $\Sigma m(0,2,5,7)$

Sum of Minterms vs Product of Maxterms

- A function can be expressed algebraically as:
  - The sum of minterms
  - The product of maxterms
- Given the truth table, writing F as
  - $\sum m_i$ – for all minterms that produce 1 in the table, or
  - ΠM_i – for all maxterms that produce 0 in the table
- Minterms and Maxterms are complement of each other.

Example: minterm & maxterm

- Write E = Y' + X'Z' in the form of $\sum m_i$ and $\prod M_i$?
- **Method1**

First construct the Truth Table as shown

E = $\sum$m(0,1,2,4,5), and

E = $\prod$M(3,6,7)

X	Y	Z	m	M	E
0	0	0	m_0	M_0	**1**
0	0	1	m_1	M_1	**1**
0	1	0	m_2	M_2	**1**
0	1	1	m_3	M_3	0
1	0	0	m_4	M_4	**1**
1	0	1	m_5	M_5	**1**
1	1	0	m_6	M_6	0
1	1	1	m_7	M_7	0

SOP and POS Conversion

SOP → POS

F = AB + CD

= (AB+C)(AB+D)

= (A+C)(B+C)(AB+D)

= (A+C)(B+C)(A+D)(B+D)

Hint 1: Use id15: X+YZ=(X+Y)(X+Z)

Hint 2: Factor

POS → SOP

F = (A'+B)(A'+C)(C+D)

= (A'+BC)(C+D)

= A'C+A'D+BCC+BCD

= A'C+A'D+BC+BCD

= A'C+A'D+BC

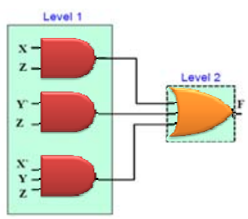
Hint 1: Use id15 (X+Y)(X+Z)=X+YZ

Hint 2: Multiply

Implementation of SOP

F(X,Y,Z) = XZ+Y'Z+X'YZ

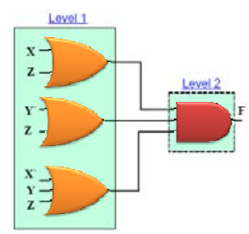
- Any SOP expression can be implemented using 2-levels of gates
- The 1st level consists of AND gates, and the 2nd level consists of a single OR gate
- Also called 2-level Circuit



Implementation of POS

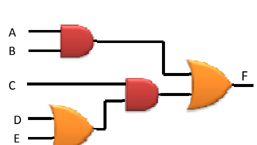
F(X,Y,Z) = (X+Z)(Y'+Z)(X'+Y+Z)

- Any POS expression can be implemented using 2-levels of gates
- The 1st level consists of OR gates, and the 2nd level consists of a single AND gate
- Also called 2-level Circuit

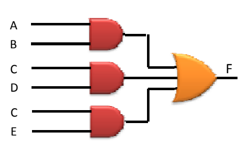


Implementation of SOP

- Consider F = AB + C(D+E)
  - This expression is NOT in the sum-of-products form
  - Use the identities/algebraic manipulation to convert to a standard form (sum of products), as in F = AB + CD + CE
- Logic Diagrams:



3-level circuit



2-level circuit

Gray Codes

- Gray code is a binary value encoding in which adjacent values only differ by one bit

2-bit Gray Code
00
01
11
10

Filling of Karnaugh Map

Why not: A'B', A'B, AB', AB

00, 01, 10, 11

Only two adjacent can be grouped

Group Reduce a variable: AB'+AB=A(B'+B)=A

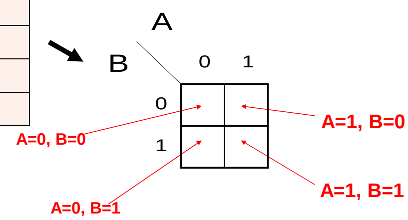
A'B', A'B, AB, AB'

00, 01, 11, 01

All 4 Adjacent can be grouped

2-Variable Karnaugh Map

A	B	F
0	0	
0	1	
1	0	
1	1	



A different way to draw a truth table: by folding it

Karnaugh Map: 2, 3, 4 Variable
Gray coding

00	01
10	11

000	001	011	010
100	101	111	110

0000	0001	0011	0010
0100	0101	0111	0110
1100	1101	1111	1110
1000	1001	1011	1010

Karnaugh Map: 2, 3, 4 Variable
Gray coding, Decimal

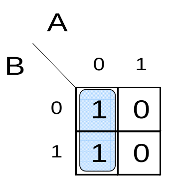
0	1
2	3

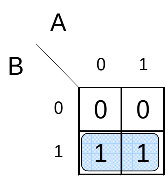
0	1	3	2
4	5	7	6

0	1	3	2
4	5	7	6
12	13	15	14
8	9	11	10

Karnaugh Map

- In a K-map, physical adjacency **does** imply gray code adjacency



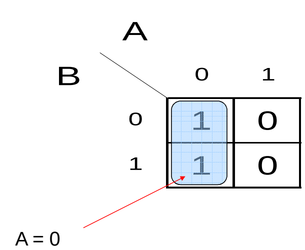


F =A'B' + A'B = A'

F = A'B + AB = B

2-Variable K Map: Grouping

A	B	index	F
0	0	0	1
0	1	1	1
1	0	2	0
1	1	3	0



F = A'

Another Example

A	B	index	F
0	0	0	0
0	1	1	1
1	0	2	1
1	1	3	1

B \ A	0	1
0	0	1
1	1	1

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

Yet Another Example

A	B	index	F
0	0	0	1
0	1	1	1
1	0	2	1
1	1	3	1

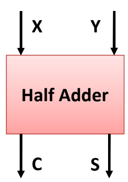
B \ A	0	1
0	1	1
1	1	1

F = 1

Groups of more than two 1's can be combined

HALF ADDER: One bit adder

X	Y	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



Y \ X	0	1
0	0	1
1	1	0

S = A'B + AB'
= A XOR B

Y \ X	0	1
0	0	0
1	0	1

C = AB

K-Map of three variable

A	B	C	index	F
0	0	0	0	1
0	0	1	1	0
0	1	0	2	1
0	1	1	3	0
1	0	0	4	1
1	0	1	5	1
1	1	0	6	1
1	1	1	7	1

F=∑m(0,2,4,5,6,7)

A \ BC	B'C'	B'C	BC	BC'
A'	1 (0)	0 (1)	0 (3)	1 (2)
A	1 (4)	1 (5)	1 (7)	1 (6)

F= A + B'C' + BC'

Group of 4 m(4,5,7,6)

Group of 2 m(0,4)

Group of 2 M(2,6)

3-Variable Karnaugh Map Showing Minterm Locations

Note the order of the B C variables:

0 0
0 1
1 1
1 0

BC \ A	0	1
00	m0	m4
01	m1	m5
11	m3	m7
10	m2	m6

ABC = 101

ABC = 010

Adjacencies

- Adjacent squares differ by exactly one variable

BC \ A	0	1
00		AB'C'
01	A'B'C	AB'C
11		ABC
10		ABC'

There is wrap-around: top and bottom rows are adjacent

Truth Table to Karnaugh Map

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

BC \ A	0	1
00	0	0
01	0	1
11	1	1
10	1	0

Minimization Example

BC \ A	0	1
00	0	0
01	0	1
11	1	1
10	1	0

A'BC+A'BC' = A'B

AB'C+ABC = AC

F = A'B + AC

Another Example

BC \ A	0	1
00	0	1
01	1	0
11	1	0
10	0	1

AB'C'+ABC' = AC'

A'B'C+A'BC = A'C

F = A'C + AC' = A ⊕ C

Minterm Expansion to K-Map

$F = \sum m(1, 3, 4, 6)$

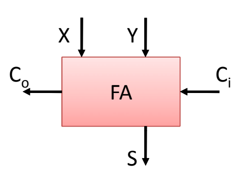
BC \ A	0	1
00	m0	m4
01	m1	m5
11	m3	m7
10	m2	m6

BC \ A	0	1
00	0	1
01	1	0
11	1	0
10	0	1

Minterms are the 1's, everything else is 0

Full Adder Example: Minterms

Ci	X	Y	index	S	Co
0	0	0	0	0	0
0	0	1	1	1	0
0	1	0	2	1	0
0	1	1	3	0	1
1	0	0	4	1	0
1	0	1	5	0	1
1	1	0	6	0	1
1	1	1	7	1	1



$S = \sum m(1, 2, 4, 7)$

$Co = \sum m(3, 5, 6, 7)$

Full Adder Output

$S = \sum m(1, 2, 4, 7)$

$Co = \sum m(3, 5, 6, 7)$

XY \ Ci	0	1
00	0	1
01	1	0
11	0	1
10	1	0

XY \ Ci	0	1
00	0	0
01	0	1
11	1	1
10	0	1

S=Ci'X'Y+Ci'XY'+CiX'Y'+CiXY

Co=XY+CiX+CiY

30

Maxterm Expansion to KMap

$F = \prod M(0, 2, 5, 7)$

BC \ A	0	1
00	M0	M4
01	M1	M5
11	M3	M7
10	M2	M6

BC \ A	0	1
00	0	1
01	1	0
11	1	0
10	0	1

Maxterms are the 0's, everything else is 1

Yet Another Example

BC \ A	0	1
00	1	1
01	1	1
11	0	1
10	0	0

2^n 1's can be circled at a time
1, 2, 4, 8, … OK
3 not OK

A'B'C'+AB'C'+
A'B'C+AB'C = B'

AB'C+ABC = AC

F = B' + AC

**The larger the group of 1's
the simpler the resulting product term**

Boolean Algebra to Karnaugh Map

Plot: ab'c' + a' + bc

BC \ A	0	1
00		1
01		
11		
10		

BC \ A	0	1
00	1	1
01	1	0
11	1	1
10	1	0

Remaining spaces are 0

Boolean Algebra to Karnaugh Map

Plot: ab'c' + bc + a'

BC \ A	0	1
00	1	1
01	1	0
11	1	1
10	1	0

Now minimize

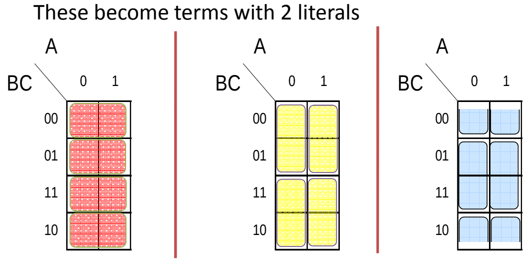
B'C' + BC + A'

This is a simpler equation than we started with.
Do you see how we obtained it?

Mapping Sum of Product Terms

The 3-variable map has 12 possible groups of 2 spaces

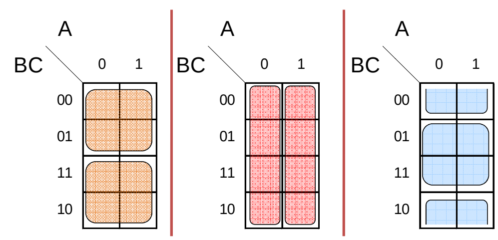
These become terms with 2 literals



Mapping Sum of Product Terms

The 3-variable map has 6 possible groups of 4 spaces

These become terms with 1 literal



4-Variable Karnaugh Map

AB \ CD	00	01	11	10
00	m0	m1	m3	m2
01	m4	m5	m7	m6
11	m12	m13	m15	m14
10	m8	m9	m11	m10

AB \ CD	00	01	11	10
00	0	0	0	1
01	1	1	1	1
11	1	1	1	1
10	0	1	0	0

A'CD'

B

AC'D

F = AC'D + A'CD' + B

Note the row and column orderings. Required for adjacency

Find a POS Solution

CD \ AB	00	01	11	10
00	1	0	0	1
01	0	0	0	0
11	1	1	1	1
10	1	1	1	0

A'D

A'B

AB'CD'

F' = A'D + A'B + AB'CD'
F = (A+D)(A+B')(A'+B+C'+D)

Find solutions to groups of 0's to find F'
Invert to get F then use DeMorgan's

Don't Care

- A **don't-care term** is an input to a function that the designer does not care about
- Because that input would never happen
- Example:
  - BCD number (0-9, A-F) are 4 bits, don't care about input A-F
  - Suppose a system have 5 type of input
    - Unfortunately we can't have 2 input line
    - Make 3 input line and last 3 sequence as don't care
    - S0, S1, S2,S3,S4, X,X,X == > 000, 001….,111

39

Dealing With Don't Cares

F = Σm(1, 3, 7) + Σd(0, 5)

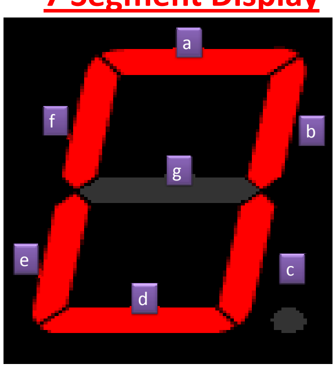
BC \ A	0	1
00	x	0
01	1	x
11	1	1
10	0	0

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

F = C

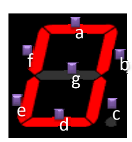
Circle the x's that help get bigger groups of 1's
Don't circle the x's that don't

7 Segment Display



41

Activation of LEDs



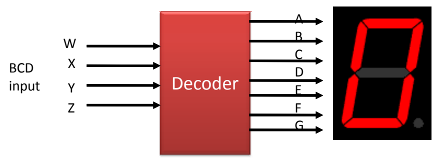
F_a(W,X,Y,Z)= Σm(0,2,3,5,6,7,8,9) + Σd(10, 11,12,13,14,15)

- 0 : a,b,c,d,e,f
- 1: b,c
- 2:a,b,g,e,d
- 3:a,b,g,c,d
- 4:f,g,b,c
- 5:a,f,g,c,d
- 6:a,f,g,c,d,e
- 7:a,b,c
- 8:a,b,c,d,e,f,g
- 9:a,b,c,d,f,g

42

BCD to 7 Segment Display

- BCD are 4 bit
- Design a decoder to drive 7 segment LED



43

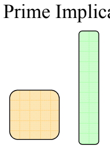
Prime Implicants

- A group of one or more 1's which are adjacent and can be combined on a Karnaugh Map is called an implicant.
- The *biggest* group of 1's which can be circled to cover a given 1 is called a prime implicant.
  - They are the only implicants we care about.

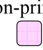
Prime Implicants

CD \ AB	00	01	11	10
00	0	0	0	1
01	0	0	1	1
11	0	1	1	1
10	0	1	1	1

Prime Implicants



Non-prime Implicants

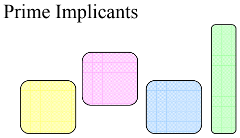


Are there any additional prime implicants in the map that are not shown above?

All The Prime Implicants

CD \ AB	00	01	11	10
00	0	0	0	1
01	0	0	1	1
11	0	1	1	1
10	0	1	1	1

Prime Implicants



When looking for a minimal solution – ***only*** circle prime implicants…
A minimal solution will ***never*** contain non-prime implicants

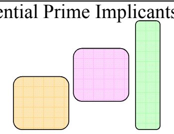
Essential Prime Implicants

CD \ AB	00	01	11	10
00	0	0	0	1
01	0	0	1	1
11	0	1	1	1
10	0	1	1	1

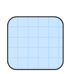
Not all prime implicants are required…

A prime implicant which is the only cover of some 1 is ***essential*** – a minimal solution requires it.

Essential Prime Implicants



Non-essential Prime Implicants

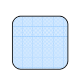


A Minimal Solution Example

CD \ AB	00	01	11	10
00	0	0	0	1
01	0	0	1	1
11	0	1	1	1
10	0	1	1	1

F = AB' + BC + AD

Minimum



Not required…

Another Example

CD \ AB	00	01	11	10
00	1	0	0	1
01	1	1	0	0
11	1	1	1	0
10	1	0	0	1

Another Example

CD \ AB	00	01	11	10
00	1	0	0	1
01	1	1	0	0
11	1	1	1	0
10	1	0	0	1

F = A'D + BCD + B'D'

Minimum

A'B' is not required…
Every one one of its locations is covered by multiple implicants

After choosing essentials, everything is covered…

Finding the Minimum Sum of Products

1. Find each essential prime implicant and include it in the solution.
2. Determine if any minterms are not yet covered.
3. Find the minimal # of remaining prime implicants which finish the cover.

Yet Another Example (Use of non-essential primes)

CD \ AB	00	01	11	10
00	1	1	0	0
01	0	0	1	1
11	1	1	1	1
10	1	1	0	0

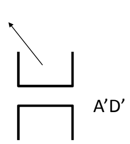
Yet Another Example (Use of non-essential primes)

CD \ AB	00	01	11	10
00	1	1	0	0
01	0	0	1	1
11	1	1	1	1
10	1	1	0	0

AD

CD

A'C



Essentials: A'D' and AD
Non-essentials: A'C and CD
Solution: A'D' + AD + A'C
or
A'D' + AD + CD

5-Variable Karnaugh Map

DE \ BC	00	01	11	10
00	m0	m4	m12	m8
01	m1	m5	m13	m9
11	m3	m7	m15	m11
10	m2	m6	m14	m10

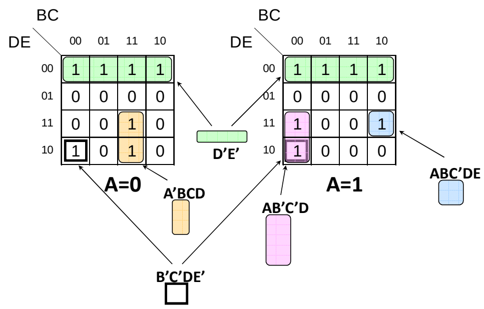
This is the A=0 plane

DE \ BC	00	01	11	10
00	m16	m20	m28	m24
01	m17	m21	m29	m25
11	m19	m23	m31	m27
10	m18	m22	m30	m26

This is the A=1 plane

The planes are adjacent to one another (one is above the other in 3D)

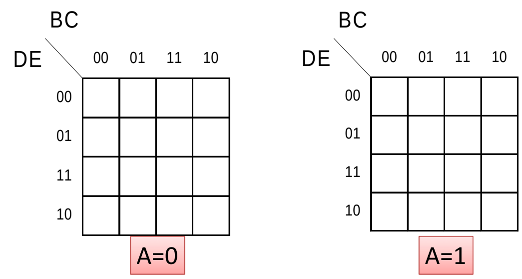
Some Implicants in a 5-Variable KMap



Some of these are not prime…

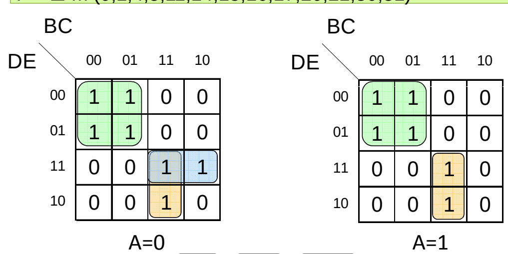
5-Variable KMap Example

Find the minimum sum-of-products for:
$F = \Sigma$ m (0,1,4,5,11,14,15,16,17,20,21,30,31)



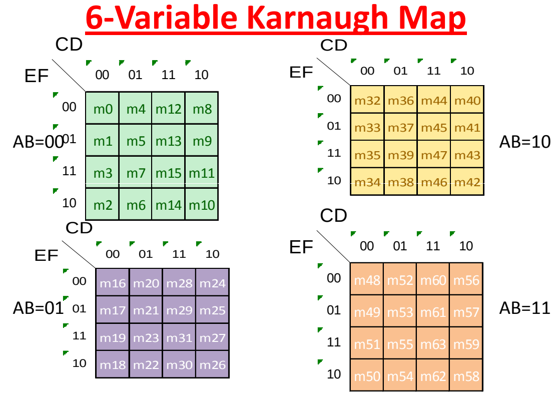
5-Variable KMap Example

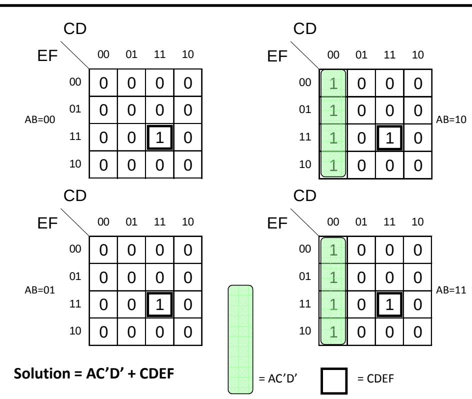
Find the minimum sum-of-products for:
$F = \Sigma$ m (0,1,4,5,11,14,15,16,17,20,21,30,31)



F = B'D' + BCD + A'BDE

6-Variable Karnaugh Map





Solution = AC'D' + CDEF

KMap Summary

- A Kmap is simply a folded truth table
  - where physical adjacency implies logical adjacency
- KMaps are most commonly used hand method for logic minimization
- KMaps have other uses for visualizing Boolean equations
  - you may see some later.

Quine-McCluskey (QM) Method for Logic Minimization

Quine-McCluskey Method for Minimization

- KMAP methods was practical for at most 6 variable functions
- Larger number of variables: need method that can be applied to computer based minimization
- **Quine-McCluskey** method
- For example:

$$\sum m(0,1,2,3,5,7,13,15)$$

QM Method

- Phase I : finding Pis
  - Tabular methods: Grouping and combining
- Phase II: Covers minimal PIs

63

QM Method

- Minterms that differ in one variable's value can be combined.
- Thus we list our minterms so that they are in groups with each group having the same number of 1s.
- So the first step is ordering the minterms according to their number of 1s (0-cube list)
- only minterms residing in adjacent groups have the chance to be combined.):

QM Method

$$\sum m(0,1,2,3,5,7,13,15)$$

0_Cube	
0	0000
1	0001
	0010
2	0011
	0101
3	0111
	1101
4	1111

QM Method: Combining Adjacent

- Compare minterms of a group with those of an **adjacent one to form 1-cube list**.
- Skip multiple adjacent
  - **0010 of Gr 1 and 0101 of G2 have Hamming distance 3, so cannot be combined.**
  - **HD value 1 is GRAY adjacent**
- When doing the combining, we put checkmark alongside the minterms in the 0-cube list that have been combined.

QM Method

0_Cube	
0	0000 √
1	0001 √
	0010 √
2	0011 √
	0101 √
3	0111 √
	1101 √
4	1111 √

1_Cube		
0	000X	0,1
	00X0	
1	00X1	1,2
	0X01	
	001X	
2	0X11	2,3
	01X1	
	X101	
3	X111	3,4
	11X1	

QM Method: Combining Adjacent

- Do same combination of comparing adjacent group minterms
  - To form 2-cubes, 3-cubes and so on.
  - Compare minterms of a group with those of an **adjacent one to form 1-cube list**.
  - Skip multiple adjacent
    - **0010 of Gr 1 and 0101 of G2 have Hamming distance 3, so cannot be combined.**
    - **HD value 1 is GRAY adjacent**
- Only minterms of adjacent groups have the chance of being combined
  - **Which have an X in the same position.**

QM Method

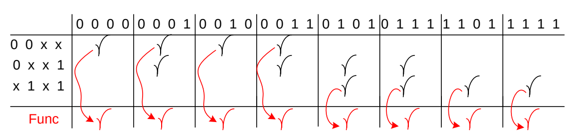
1_Cube	
0	000X √
	00X0 √
1	00X1 √
	0X01 √
	001X √
2	0X11 √
	01X1 √
	X101 √
3	X111 √
	11X1 √

2_Cube	
0	00XX *
1	0XX1 *
2	X1X1 *

Q-M Method: Cover PIs

- PIs : terms left without checkmarks.
- After identifying our PIs, we list them against the minterms needed to be covered

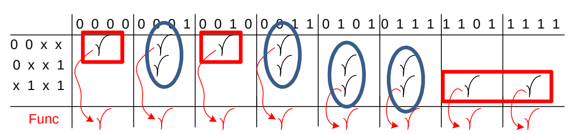
$$\sum m(0,1,2,3,5,7,13,15)$$



QM Method : Covers

- To find a minimal cover, we first need to find essential Pis
- To do this we need to find columns that only have one checkmark in them, the according row will thus show the essential PI.
- After identifying essential PIs, that are necessarily part of the cover, we cover any remaining minterms using a minimal set of PIs.

QM Method : Covers



Essential I | Essential II | Redundant

In this example:

F = A'B' +BD