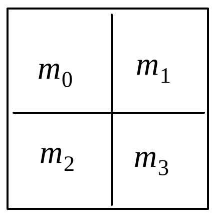
Systems I: Computer Organization and Architecture

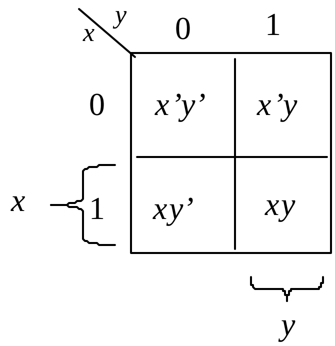
Lecture 4 - Karnaugh Maps

Mapping Boolean Functions

- Boolean expressions can be fairly complex.
  - This leads to overly-complex digital circuits.
  - This necessitates simplification of Boolean expressions.
- These expressions can become too complex for simplification by Boolean algebra.
- The use of maps such as Karnaugh maps makes it much easier to simplify such expressions.

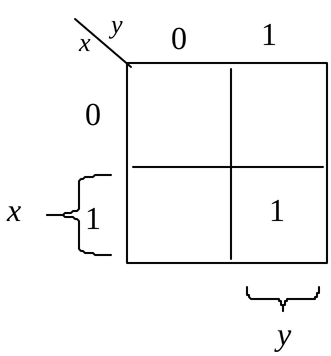
Two-variable maps



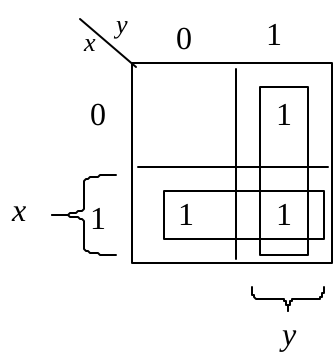


Each square represents a minterm.

Representing 2-Variable Functions



$F = xy$



$F = x + y$

$= x'y + xy' + xy$

Three-variable maps

- With only two dimensions on the page, we need to graph more than one dimension together whenever we go beyond two-variable maps.
- We arrange the minterms in an order that resembles Gray codes, where only one bit varies between adjacent squares.
- This allows us to recognize simpler terms quickly.

3-Variable Karnaugh Maps

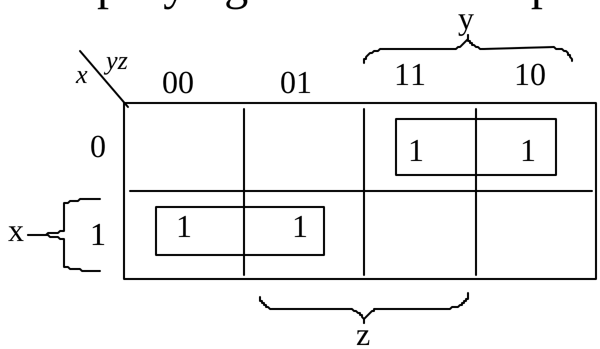
m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6

y

x \ yz	00	01	11	10
0	x’y’z’	x’y’z	x’yz	x’yz’
x 1	xy’z’	xy’z	xyz	xyz’

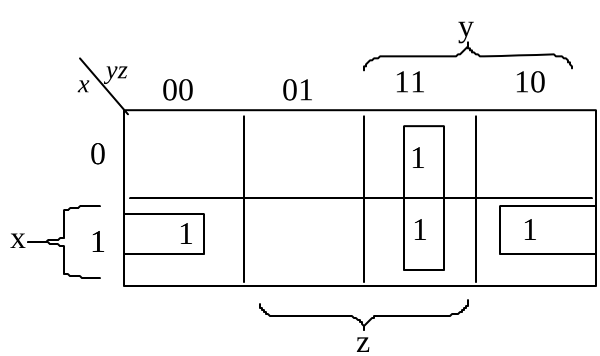
z

Simplifying 3-Variable Expressions



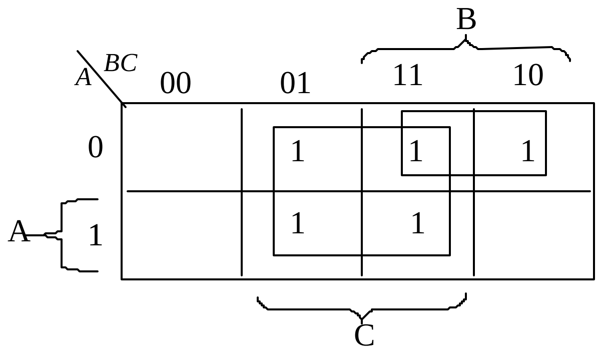
$$F = x'yz + x'yz' + xy'z' + xy'z$$
$$= xy' + x'y$$

Simplifying 3-Variable Expressions (continued)



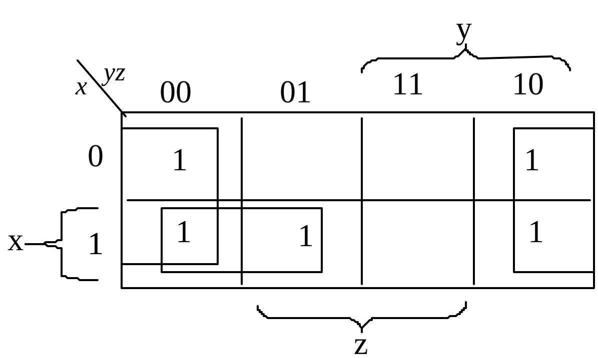
$$F = x'yz + xy'z' + xyz + xyz'$$
$$= xz' + yz$$

Simplifying 3-Variable Expressions (continued)



F = A’C+A’B + AB’C+BC
= C + A’B

Simplifying 3-Variable Expressions (continued)



F = Σ (0, 2, 4, 5, 6)
= z’ + xy’

Four-Variable Maps

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}

wx \ yz	00	01	11	10
00	w’x’y’z’	w’x’y’z	w’x’yz	w’x’yz’
01	w’xy’z’	w’xy’z	w’xyz	w’xyz’
11	wxy’z’	wxy’z	wxyz	wxyz’
10	wx’y’z’	wx’y’z	wx’yz	wx’yz’

y (columns 11, 10); x (rows 01, 11); w (rows 11, 10); z (columns 01, 11)

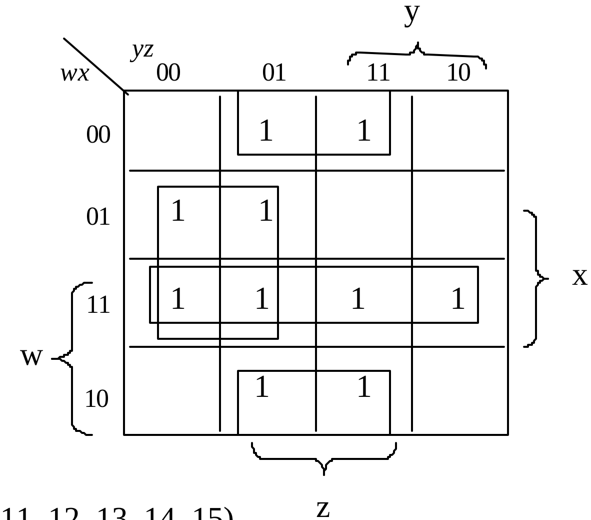
Simplifying 4-variable Maps

wx \ yz	00	01	11	10
00	1	1		1
01	1	1		1
11	1	1		1
10	1	1		

y (columns 11, 10); x (rows 01, 11); w (rows 11, 10); z (columns 01, 11)

F = Σ(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14)
= y’ + w’z’ + xz’

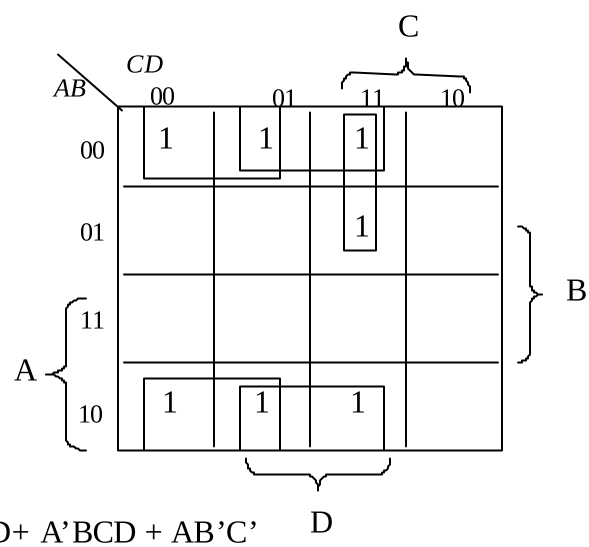
Simplifying 4-variable Maps (continued)



$F = \Sigma(1, 3, 4, 5, 9, 11, 12, 13, 14, 15)$

$= wx + xy' + x'z$

Simplifying 4-variable Maps (continued)



$F = A'B'C' + B'CD + A'BCD + AB'C'$

$= B'D + B'C' + A'CD$

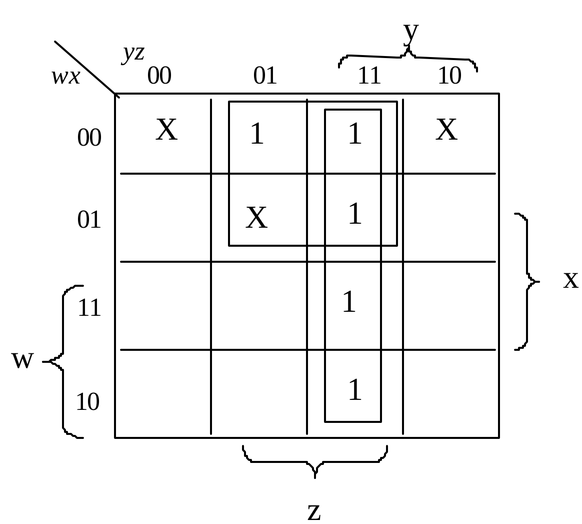
Don't Care Conditions

- Until now, all the spaces on a Karnaugh map indicates where the function has a value of "1" or "0" (assumed by the square being left empty).
- Sometimes we don't care what value the square holds – it is irrelevant.
- We can't leave it blank (assumed to be 0), 0 or 1, so we mark it with an "X" to indicate that we ***don't care*** about its value.

Don't-Care Conditions – An Example

F = Σ(1,3,7,11,15)
d = Σ(0, 2, 5)

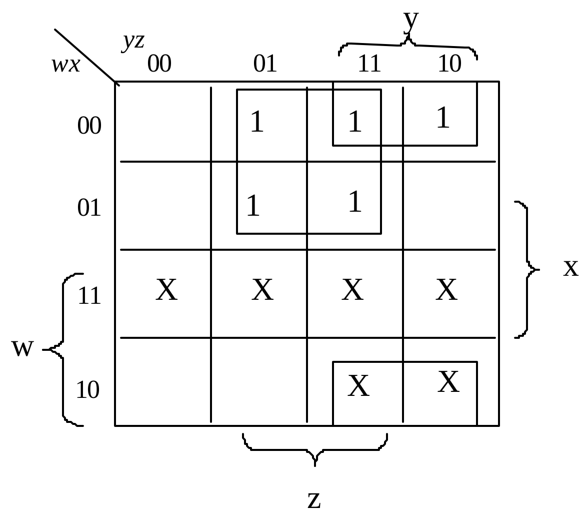
F = w'z + yz
= z (w' + y)



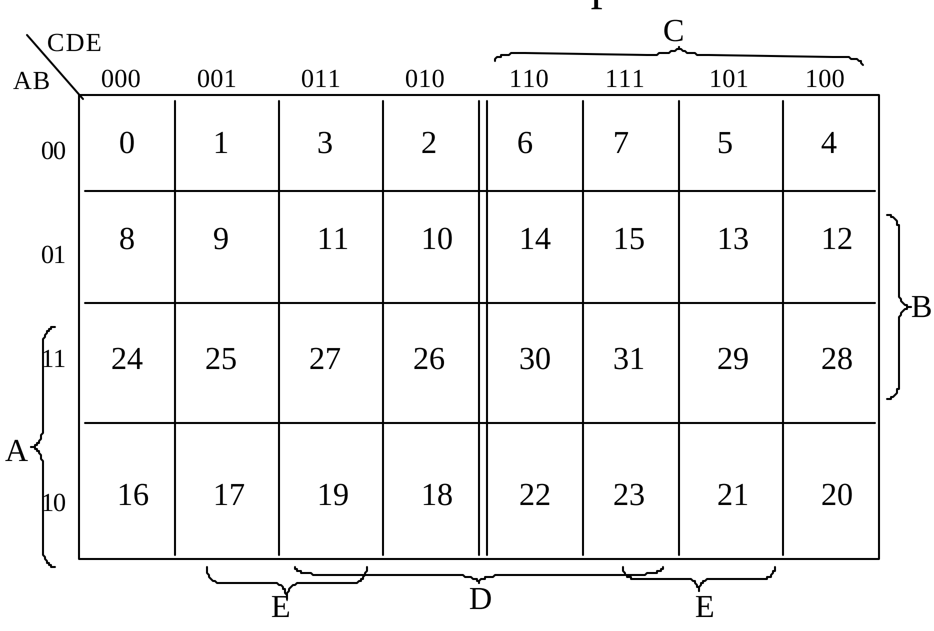
Don’t-Care Conditions – An Example

F = Σ(1,2,3,5,7)
d = Σ(10,11,12, 13
14, 15)

F = w’z + x’y



5-Variable Maps



6-Variable Maps

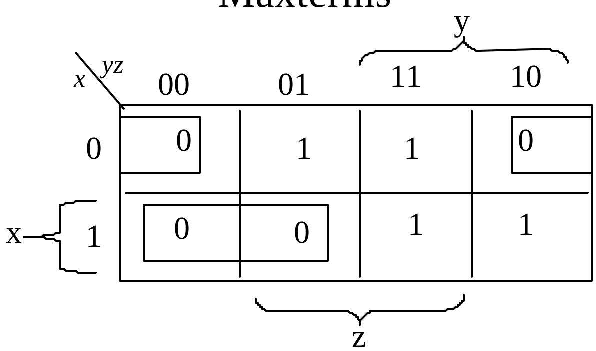
ABC \ DEF	000	001	011	010	110	111	101	100
000	0	1	3	2	6	7	5	4
001	8	9	11	10	14	15	13	12
011	24	25	27	26	30	31	29	28
010	16	17	19	18	22	23	21	20
110	48	49	51	50	54	55	53	52
111	56	57	59	58	62	63	61	60
101	40	41	43	42	46	47	45	44
100	32	33	35	34	38	39	37	36

Product of Maxterms

- Normally, we express Boolean function as a sum of minterms, e.g.,
  - xy + x'z
  - A+ B'C
- Each of the 2^{2n} functions of n binary variables can be rewritten as a product of maxterms, e.g,:

  xy + x'z = (x'+y)(x+z)(y+z)

Using Karnaugh Maps to Find Product of Maxterms



F’ = xy’ + x’z’

F = (x’+y)(x+z)

Using Karnaugh Maps and Don’t-Care Conditions to Find Product of Maxterms

F’ = z’ + wy’

F = z(w’+y)