

Simplifying Digital Logic Functions

Introduction to Combinatorial Logic

CS 64: Computer Organization and Design Logic
Lecture #13

Ziad Matni
Dept. of Computer Science, UCSB

Administrative

- Lab# 6 is due tomorrow
- Remaining on the calendar... *This supersedes anything on the syllabus*

DATE	TOPIC	ASSIGNMENTS
Thu. 3/1	Simplifying Digital Logic Functions	Lab 6 (due Fri. 3/2)
Tue. 3/6	Combinatorial Logic	
Thu. 3/8	Sequential Logic	Lab 7 (due Fri. 3/9)
Tue. 3/13	Finite State Machines	
Thu. 3/15	Ethics	Labs 8 and 9 (due Fri. 3/16)

Lecture Outline

- Logic Functions and their Simplifications:
Truth Table Use vs. **Karnaugh Maps**

Digital Circuit Design Process

CAN THIS PROCESS BE REVERSED?

Function
definition

adder

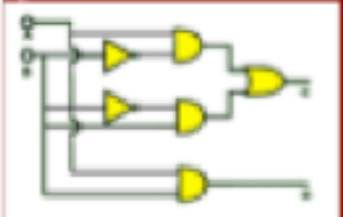
Truth
table

A	B	Carry	Sum
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Boolean
expression

$$\text{Sum} = (A\bar{B}) + (\bar{A}B)$$
$$\text{Carry} = AB$$

Logic
block



Boolean Logic Laws

Circuit Equivalence - each law has 2 forms that are duals of each other.

Name	AND form	OR form
Identity law	$1A = A$	$0 + A = A$
Null law	$0A = 0$	$1 + A = 1$
Idempotent law	$AA = A$	$A + A = A$
Inverse law	$A\bar{A} = 0$	$A + \bar{A} = 1$
Commutative law	$AB = BA$	$A + B = B + A$
Associative law	$(AB)C = A(BC)$	$(A + B) + C = A + (B + C)$
Distributive law	$A + BC = (A + B)(A + C)$	$A(B + C) = AB + AC$
Absorption law	$A(A + B) = A$	$A + AB = A$
De Morgan's law	$\overline{AB} = \bar{A} + \bar{B}$	$\overline{A + B} = \bar{A}\bar{B}$

More Simplification Examples

Simplify the Boolean expression:

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

Simplify the Boolean expression and write it out on a truth table as proof

- $XZ + Z(X' + XY)$

Use DeMorgan's Theorem to re-write the expression below using at least one OR operation

- $\text{NOT}(X + YZ)$

Scaling Up Simplification

- When we get to *more* than 3 variables, it becomes challenging to use truth tables
- We can instead use ***Karnaugh Maps*** to make it immediately apparent as to what can be simplified

Example of a K-Map

0
1
2
3

A	B	f(A,B)
0	0	a
0	1	b
1	0	c
1	1	d

B \ A	0	1
0	a	c
1	b	d

B \ A	0	1
0	0	2
1	1	3

A	B	f(A,B)
0	0	0
0	1	1
1	0	1
1	1	1

B \ A	0	1
0	0	1
1	1	1

K-Maps with 3 or 4 Variables

AB		A			
		00	01	11	10
C	0	0	2	6	4
	1	1	3	7	5

Diagram illustrating a 3-variable K-map (3x4 grid) for variables A, B, and C. The horizontal axis represents variable A (00, 01, 11, 10) and the vertical axis represents variable C (0, 1). The grid contains values 0, 2, 6, 4 in the top row and 1, 3, 7, 5 in the bottom row. Blue brackets indicate the horizontal axis is labeled A and the vertical axis is labeled C. A blue bracket labeled B is shown under the bottom row, indicating the horizontal axis is also labeled B.

Note the adjacent placement of:

00 01 11 10

It's NOT:

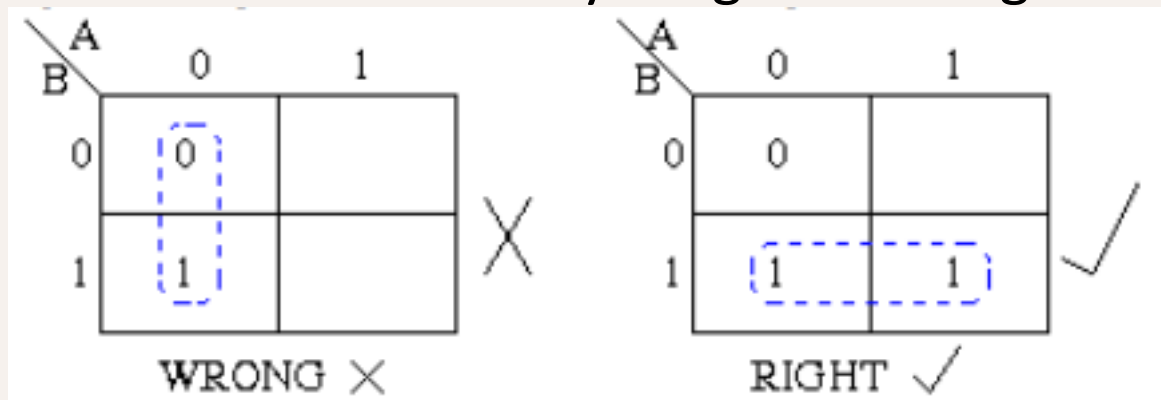
00 01 10 11

AB		A			
		00	01	11	10
CD	00	0	4	12	8
	01	1	5	13	9
	11	3	7	15	11
	10	2	6	14	10

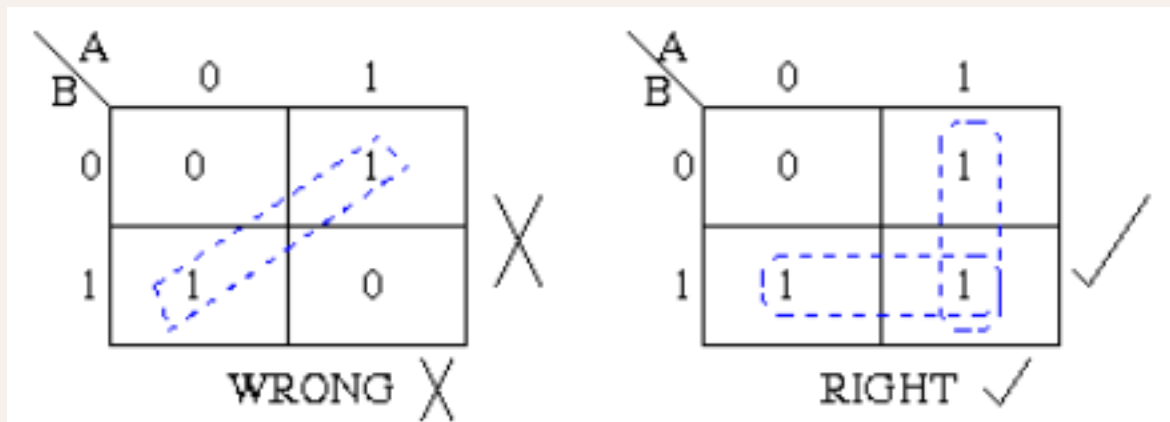
Diagram illustrating a 4-variable K-map (4x4 grid) for variables A, B, C, and D. The horizontal axis represents variable A (00, 01, 11, 10) and the vertical axis represents variable C (00, 01, 11, 10). The grid contains values 0, 4, 12, 8 in the top row; 1, 5, 13, 9 in the second row; 3, 7, 15, 11 in the third row; and 2, 6, 14, 10 in the bottom row. Blue brackets indicate the horizontal axis is labeled A and the vertical axis is labeled C. A blue bracket labeled B is shown under the bottom row, indicating the horizontal axis is also labeled B. A blue bracket labeled D is shown to the right of the bottom row, indicating the vertical axis is also labeled D.

Rules for Using K-Maps for Simplification

1. Group together **adjacent cells** containing “1”
2. Groups should **not include** anything containing “0”



3. Groups may be horizontal or vertical, but **not diagonal**



Rules for Using K-Maps for Simplification

4. Groups must contain 1, 2, 4, 8, or in general 2^n cells.

A \ B	0	1
0	1	1
1	0	0

Group of 2

RIGHT ✓

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

Group of 3

WRONG ✗

A \ B	0	1
0	1	1
1	1	1

Group of 4

RIGHT ✓

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

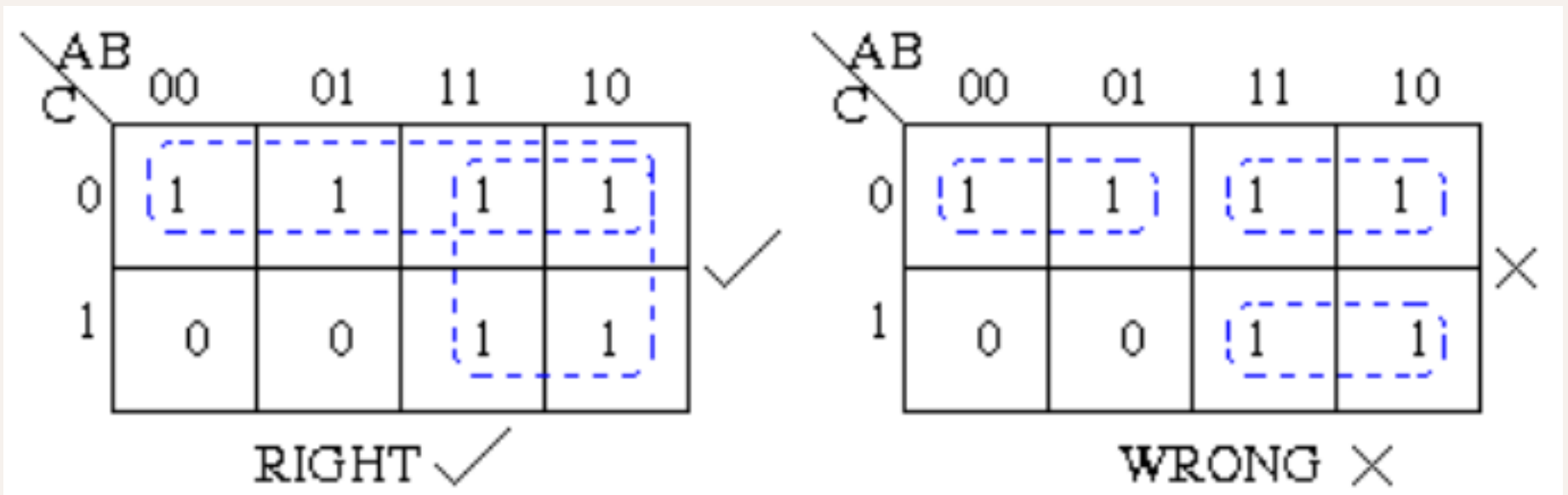
Group of 5

WRONG ✗

Rules for Using K-Maps for Simplification

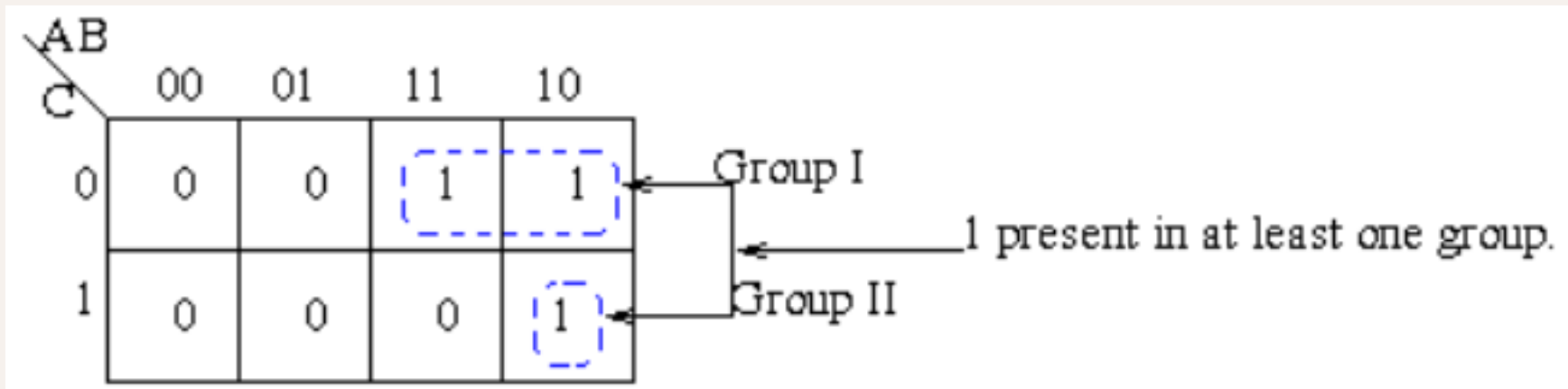
5. Each group must be as large as possible

(Otherwise we're not being as minimal as we can be, even though we're not breaking any Boolean rules)



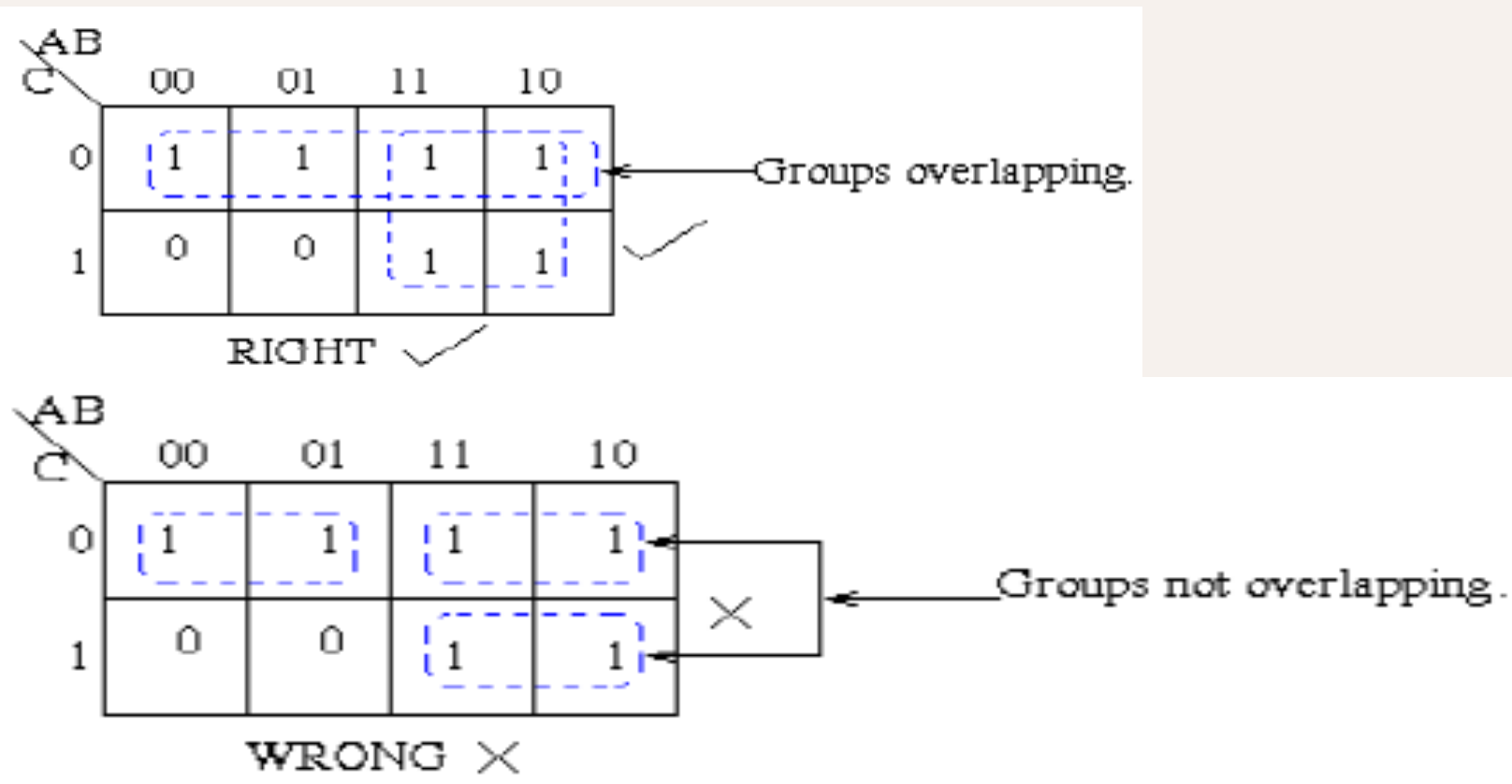
Rules for Using K-Maps for Simplification

6. Each cell containing a “1” must be at least in one group



Rules for Using K-Maps for Simplification

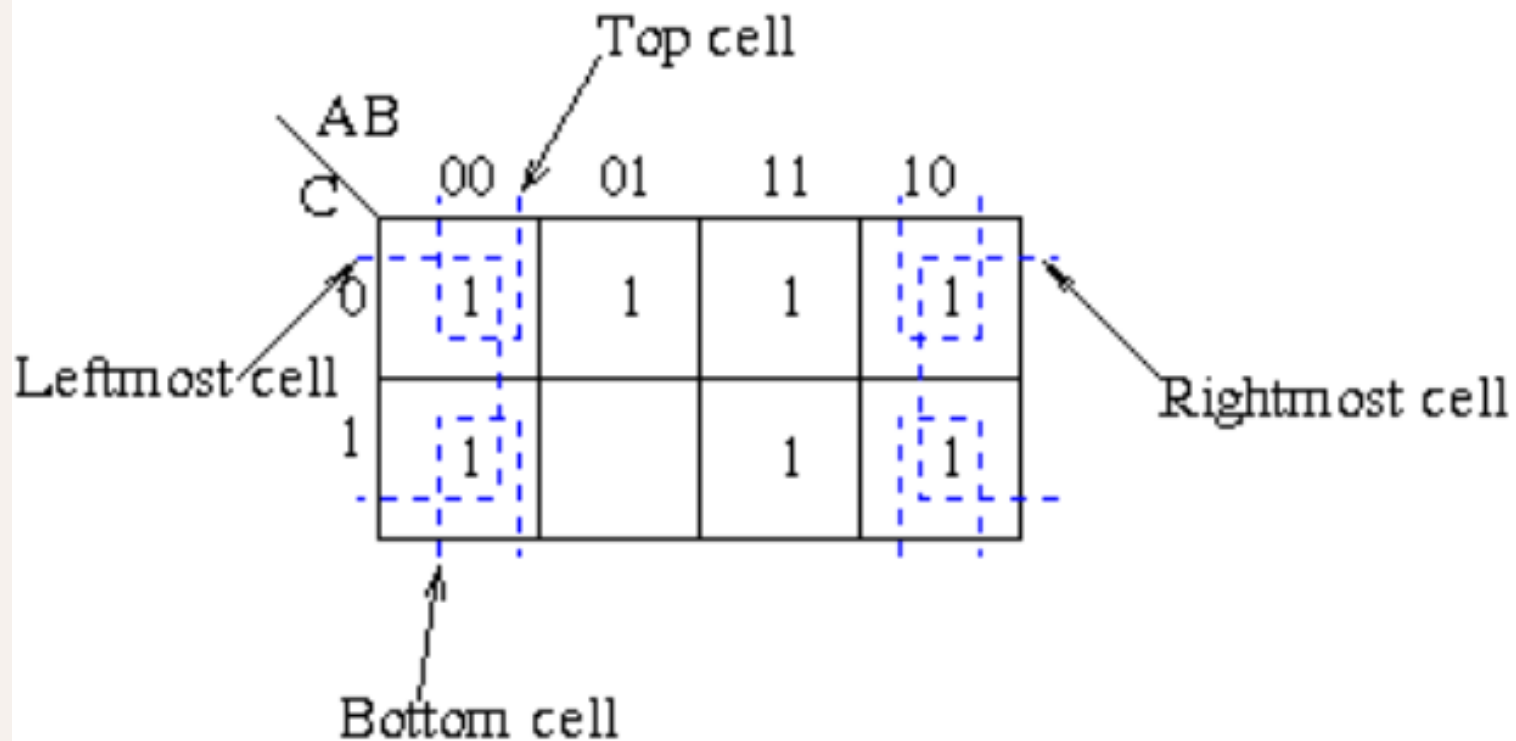
7. Groups may overlap esp. to maximize group size



Rules for Using K-Maps for Simplification

8. Groups may wrap around the table.

The leftmost cell in a row may be grouped with the rightmost cell and the top cell in a column may be grouped with the bottom cell.



Example 1

2 vars

$F(X,Y)$

$$= XY + Y$$

$$= Y (X + 1)$$

$$= Y$$

Y = 1 column

$\begin{array}{c} Y \\ \diagdown \\ X \end{array}$		0	1
0			1
1			1

$F(X,Y) = Y$

Example 2

3 vars

$F(X,Y,Z)$

$$= XZ + Z(X' + XY)$$

$$= XZ + ZX' + ZXY$$

$$= Z(X + X' + XY)$$

$$= Z(1 + XY)$$

$$= Z$$

A Karnaugh map for the function F(X,Y,Z) with variables X, Y, and Z. The map is a 2x4 grid. The columns are labeled with XY values: 00, 01, 11, 10. The rows are labeled with Z values: 0 and 1. The cell at (Z=1, XY=00) contains a red 1. The cells at (Z=1, XY=01), (Z=1, XY=11), and (Z=1, XY=10) also contain red 1s. A red dashed rectangle encloses these four cells. Above the map, a blue bracket labeled 'Y = 1' spans the columns 01 and 11. A yellow bracket labeled 'X = 1' spans the columns 11 and 10. A red arrow points from the text 'F(X,Y,Z) = Z' to the red 1s in the Z=1 row.

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

$F(X,Y,Z) = Z$ ←

Example 3

3 vars

Class Ex.

$$!A!B!C + !A!BC + !ABC + !AB!C + A!B!C + AB!C$$

		AB			
		00	01	11	10
C	0	1	1	1	1
	1	1	1		

$$F(X,Y,Z) = !C + !A$$

Example 4

4 vars

$$F(A,X,Y,Z)$$

$$= AX + Z(X+A'+Y)$$

$$= AX + ZX + ZA' + ZY$$

$$F(A,X,Y,Z) = ZA' + AX + ZY$$

A 4x4 Karnaugh map for the function F(A,X,Y,Z). The columns are labeled XY (00, 01, 11, 10) and the rows are labeled AZ (00, 01, 11, 10). The map contains 1s in the following cells: (01,01), (01,11), (01,10), (11,01), (11,11), (11,10), (10,11), and (10,10). These 1s are grouped into four sets: a group of four 1s where Z=1 (rows 01 and 11, columns 01 and 11), a group of four 1s where A=1 (rows 11 and 10, columns 01 and 11), a group of two 1s where Y=1 (columns 01 and 11, rows 01 and 11), and a group of two 1s where X=1 (columns 11 and 10, rows 01 and 11). Red dashed boxes highlight the Z=1 and A=1 groups. Blue and yellow brackets above the map indicate the Y=1 and X=1 groups respectively. A red arrow points from the Z=1 group to the term ZA' in the simplified equation.

AZ \ XY	00	01	11	10
00				
01		1	1	1
11		1	1	1
10			1	1

Example 4

4 vars

Class Ex.

$F(A,B,C,D)$

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

$$F(A,B,C,D) = A'B' + D + ABC$$

A 4-variable Karnaugh map for F(A,B,C,D). The columns are labeled AB (00, 01, 11, 10) and the rows are labeled CD (00, 01, 11, 10). The map contains 1s in the following cells: (00,00), (00,01), (00,11), (00,10), (01,00), (01,01), (01,11), (01,10), (11,00), (11,01), (11,11), (11,10), (10,00), (10,01), (10,11), (10,10). The map is annotated with groupings and prime implicants: a blue bracket above the top row is labeled B=1; a yellow bracket above the top row is labeled A=1; a blue bracket to the left of the middle two rows is labeled D=1; a yellow bracket to the left of the bottom two rows is labeled C=1; a red dashed box encloses the 4x2 area where C=1; a red arrow points from the text 'ABC' in the simplified expression to the cell (01,11).

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

K-Map Rules Summary

- Groups can contain only 1s
- Only 1s in adjacent groups are allowed (no diagonals)
- The number of 1s in a group must be a power of two (1, 2, 4, 8...)
- Groups must be as large AND as few in no.s as “legally” possible
- All 1s must belong to a group, even if it’s a group of one element
- Overlapping groups are permitted
- Wrapping around the map is permitted

Exploiting “Don’t Cares”

- An output variable that’s designated “don’t care” (symbol = X) means that it could be a **0** or a **1** (i.e. we “don’t care” which)
 - That is, it is **unspecified**,
usually because of invalid inputs

Example of a Don't Care Situation

- Consider coding all decimal digits (say, for a digital clock app):
 - 0 thru 9 --- requires how many bits?
 - 4 bits
 - But! 4 bits convey more numbers than that!
 - Don't forget A thru F!
- Not all binary values map to decimal



Example Continued...

Binary	Decimal
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7

Binary	Decimal
1000	8
1001	9
1010	X
1011	X
1100	X
1101	X
1110	X
1111	X

Don't Care: So What?

- Recall that in a K-map, we can only group 1s
- Because the value of a don't care is irrelevant, we can treat it as a 1 ***if it is convenient to do so*** (or a 0 if that would be more convenient)

Example

- A circuit that calculates if the 4-bit binary coded single digit decimal **input % 2 == 0**
- So, although 4-bits will give me numbers from 0 to 15, I don't care about the ones that yield 10 to 15.

I3	I2	I1	I0	R
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

Example as a K-Map

$I_1 I_0$					
$I_3 I_2$		00	01	11	10
	00	1	0	1	0
	01	1	0	1	0
	11	X	X	X	X
	10	1	0	X	X

If We Don't Exploit “Don't Cares”

$$R = \neg I_1 \neg I_0 I_3 + I_1 I_0 I_3 + \neg I_0 \neg I_1 I_2$$

$I_1 I_0$					
$I_3 I_2$		00	01	11	10
00		1	0	1	0
01		1	0	1	0
11		X	X	X	X
10		1	0	X	X

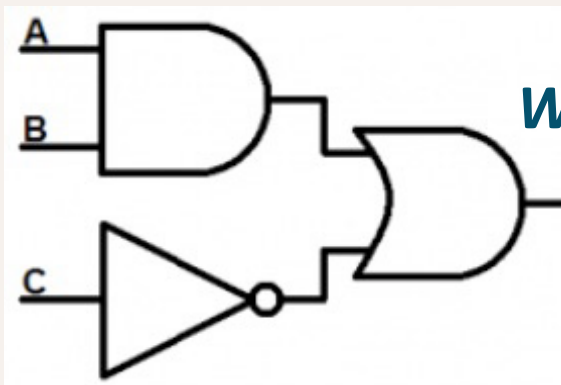
If We **DO** Exploit “Don’t Cares”

$$R = !I_1!I_0 + I_1I_0$$

I_1I_0		00	01	11	10
I_3I_2					
00		1	0	1	0
01		1	0	1	0
11		X	X	X	X
10		1	0	X	X

Combinatorial Logic Designs

- When you *combine* multiple logic blocks together to form a more complex logic function/circuit



What is the output?

$$A.B + \overline{C}$$

What is its K-Map?

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

What is its truth table?

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

Exercise 1

- Given the following truth table, draw the resulting logic circuit
 - **STEP 1:** Draw the K-Map and simplify the function
 - **STEP 2:** Construct the circuit from the now simplified function

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

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

Exercise 1 – Step 1

Get the simplified function

		AB			
CD		00	01	11	10
	00		1	1	
	01				
	11			1	1
	10			1	1

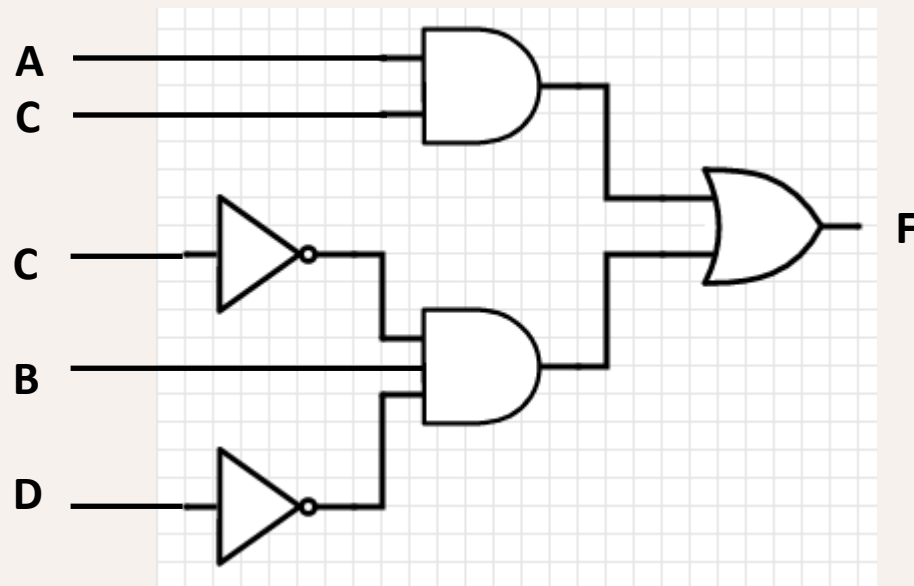
$B = 1$ (grouping 01 and 11 in row 00)
 $A = 1$ (grouping 11 and 10 in row 00)
 $D = 1$ (grouping 00 and 01 in column 11)
 $C = 1$ (grouping 11 and 10 in column 11)

$$F(A,B,C) = B.C'.D' + A.C$$

Exercise 1 – Step 2

Draw the logic circuit diagram

$$F(A,B,C) = B.C'.D' + A.C$$



Exercise 2

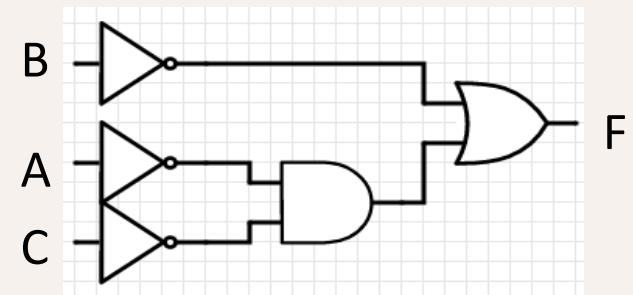
Class Ex.

- Given the following truth table, draw the resulting logic circuit

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

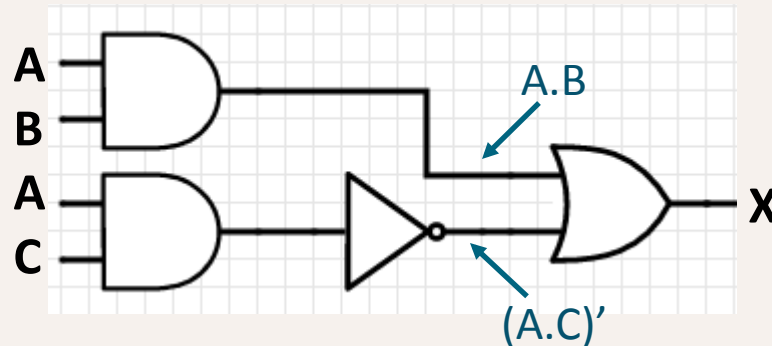
		AB			
		00	01	11	10
C	0	1	1		1
	1	1			1

$$F(A,B,C) = B' + A'.C'$$



Exercise 3

- Given the following schematic of a circuit, (a) write the function and (b) fill out the truth table:



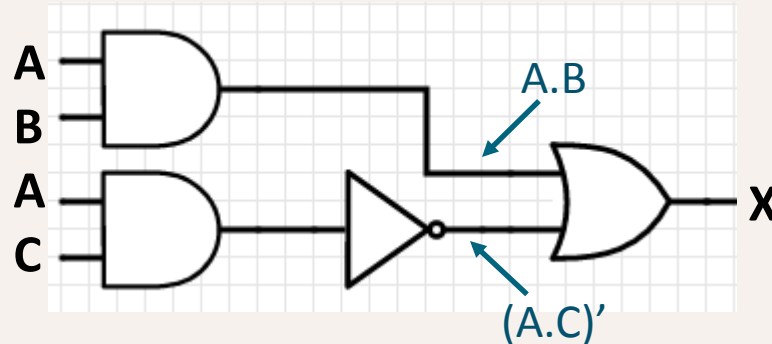
$$X = A.B + (A.C)'$$

(note that also means: $X = A.B + A' + C'$)

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

Exercise 3

- Given the following schematic of a circuit, (a) write the function and (b) fill out the truth table:



$$X = A.B + (A.C)'$$

(note that also means: $X = A.B + A' + C'$)

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

Your To Dos

- Lab #6 is due end of day Friday

</LECTURE>