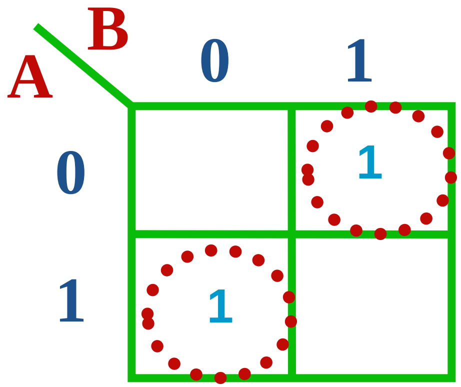


K-MAP SIMPLIFICATION

We know how to derive the output expression from …

- ✓ Logic circuits

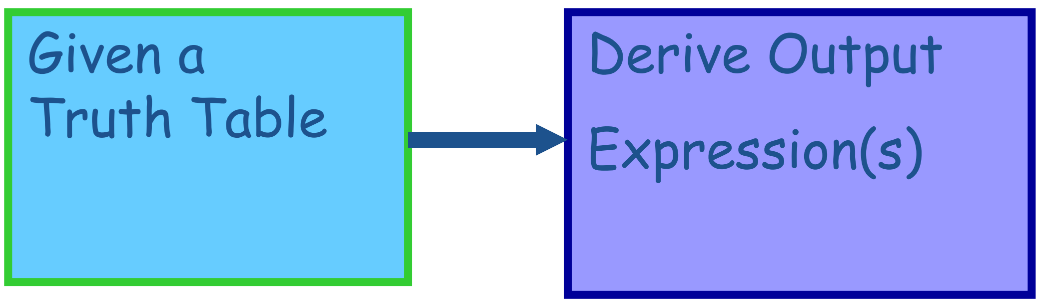
We will learn how to derive the output expression from …

- ✓ Logic circuits
- **Truth tables**

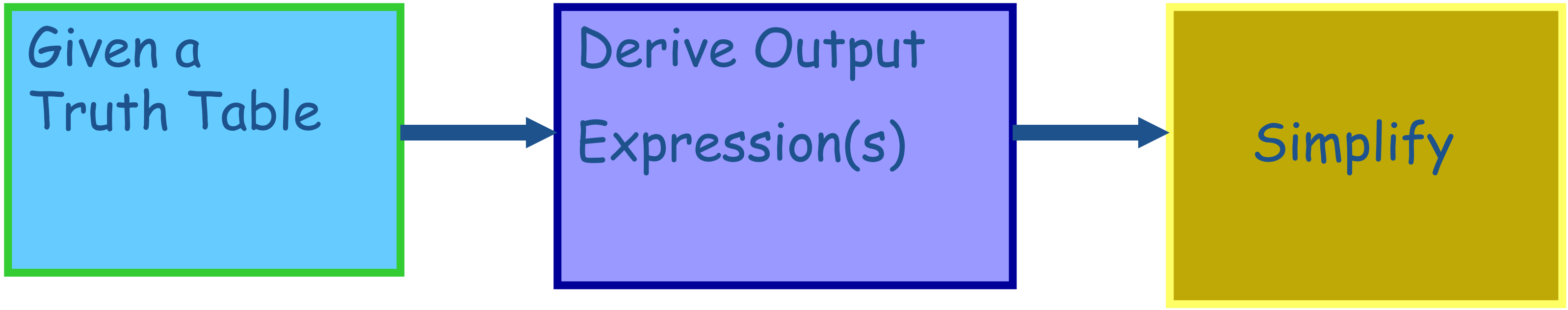
Truth table …

Given a Truth Table

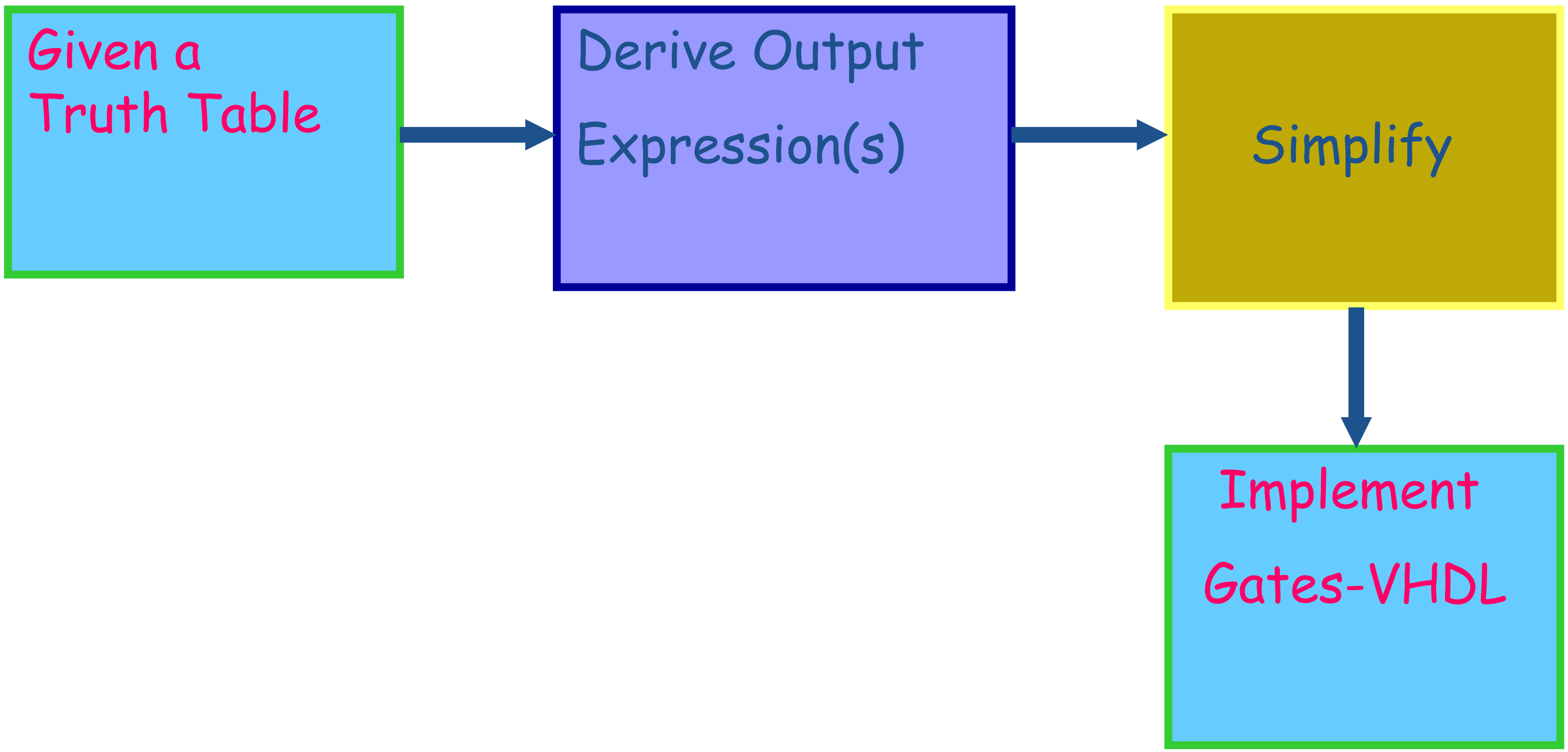
Output expressions …



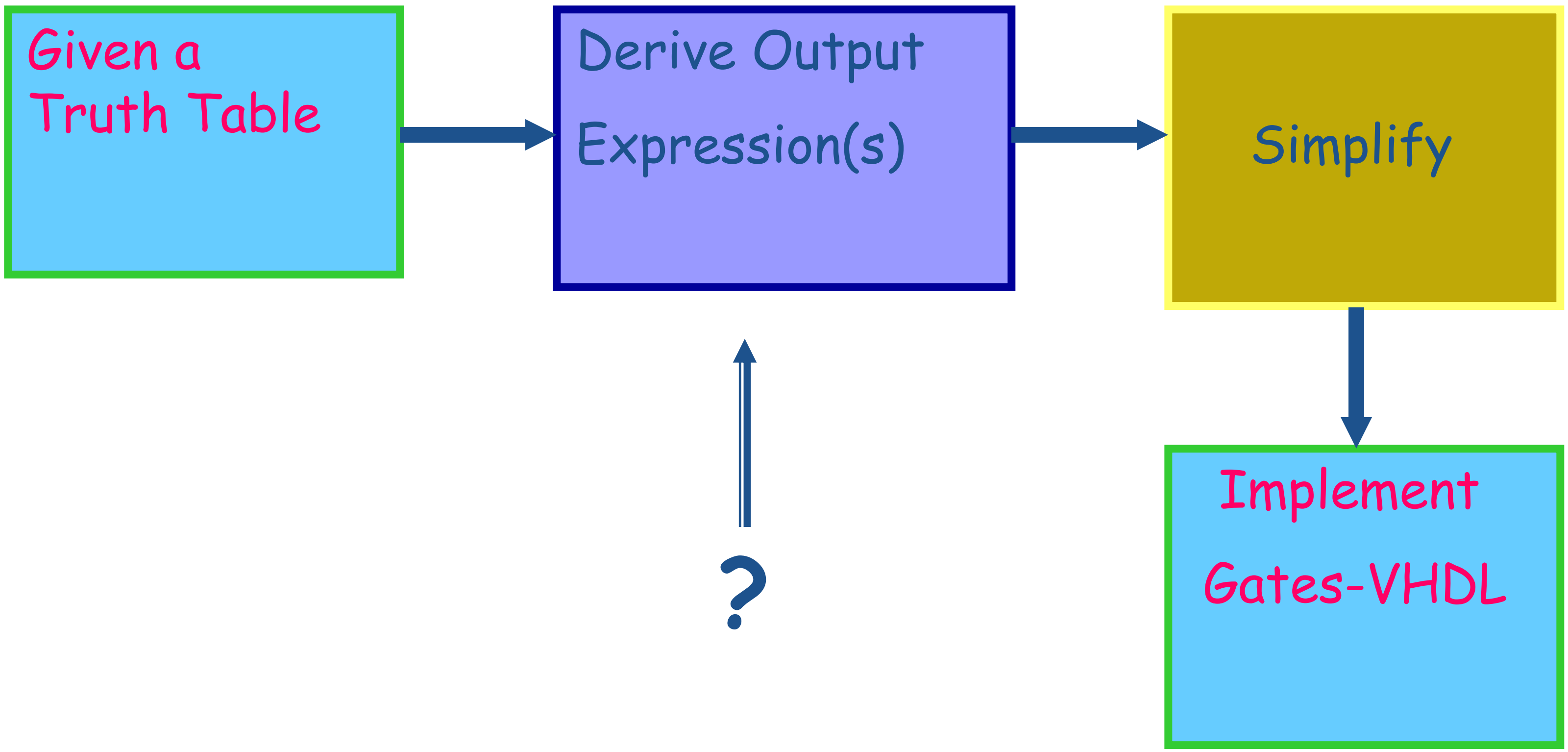
Simplify …



Gates and VHDL coding



Truth table → Simplified circuit



How can we derive an output expression from a Truth Table?

How can we derive an output expression from a Truth Table?

Algorithm:

1. Write an AND term (Boolean expression) for each case in the truth table the output is logic 1.

Algorithm is an anagram of logarithm ?

Truth table → output logic expression(s)

Algorithm:

1. Write an AND term (Boolean expression) for each case in the truth table the output is logic 1.

2. All the AND terms are then ORed together to produce the final output expression

Example: Derive the Truth Table

Word Problem:
For a three-input (A,B,C) binary system. If we have more than one high(1) inputs the output (X) is 1, otherwise is zero(0).

Example: Truth Table

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	

Word Problem:
For a three-input (A,B,C) binary system. If we have more than one high(1) inputs the output (X) is 1, otherwise is zero(0).

Example: Truth Table (done)

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

Word Problem:
For a three-input (A,B,C) binary system. If we have more than one high(1) inputs the output (X) is 1, otherwise is zero(0).

Example: Write Terms

A	B	C	X	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\overline{A}\,B\,C$
1	0	0	0	
1	0	1	1	$A\,\overline{B}\,C$
1	1	0	1	$A\,B\,\overline{C}$
1	1	1	1	$A\,B\,C$

Example: Output expression (SOP)

A	B	C	X	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\overline{A}BC$
1	0	0	0	
1	0	1	1	$A\overline{B}C$
1	1	0	1	$AB\overline{C}$
1	1	1	1	ABC

SOP = Sum-Of-Products

$$X = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

Simplify the logic expression

$$X = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

Add two ABC terms

$$X = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

$$= \overline{A}BC + \overline{AB}C + AB\overline{C} + ABC + ABC + ABC$$

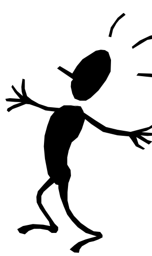
Result

$$X = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

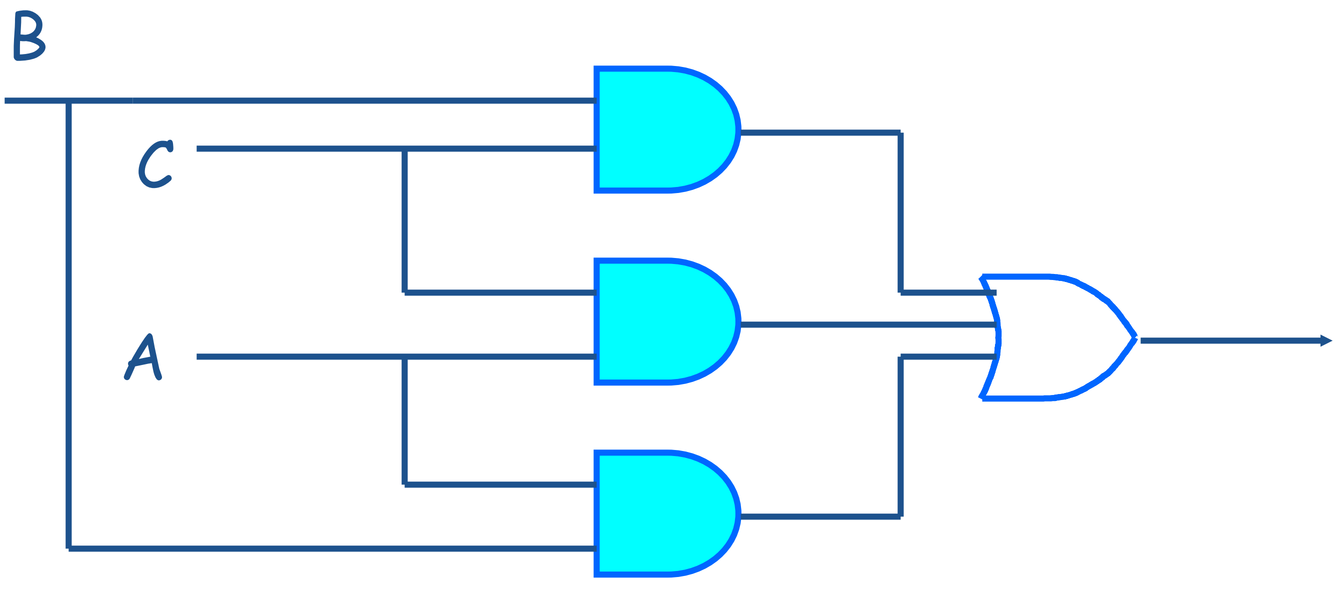
$$= \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC + ABC + ABC$$

$$= BC(\overline{A}+A)+AC(\overline{B}+B)+AB(\overline{C}+C)$$

$$= BC + AC + AB$$



Implementation: Logic Circuit



BC + AC + AB

Conclusion

The algebraic simplification procedure is very **unsystematic**…

A “more” systematic way…

- There is a more systematic way to simplify logic expressions: Karnaugh maps or K-maps.

Karnaugh maps (K-maps)

Maurice Karnaugh, "*The Map Method for Synthesis of Combinational Logic Circuits*", Trans. AIEE. part I, 72(9):593-599, November 1953.

Karnaugh maps (K-maps)

K-map is a symbolic representation of a truth table that enables us to simplify a logic expression.

- 2-variable K-map
- 3-variable K-map
- 4-variable K-map
- ...

2-variable K-map setup

A \ B	0	1
0		
1		

4-cells having values: 0 or 1

3-variable K-map setup

A \ BC	00	01	11	10
0				
1				

or …

3-variable K-map setup

AB \ C	0	1
00		
01		
11		
10		

The 00, 01, 11, 10 are not in ascending order. This is the **Gray Code**...

4-variable K-map setup

AB \ CD	00	01	11	10
00				
01				
11				
10				

K-map: 2-variable set up

A \ B	0	1
0		
1		

given:

A	B	X	
0	0	1	$\rightarrow \overline{A}\,\overline{B}$
0	1	0	
1	0	0	
1	1	1	$\rightarrow AB$

K-map: 2-variable set up

A \ B	0	1
0	1	0
1	0	1

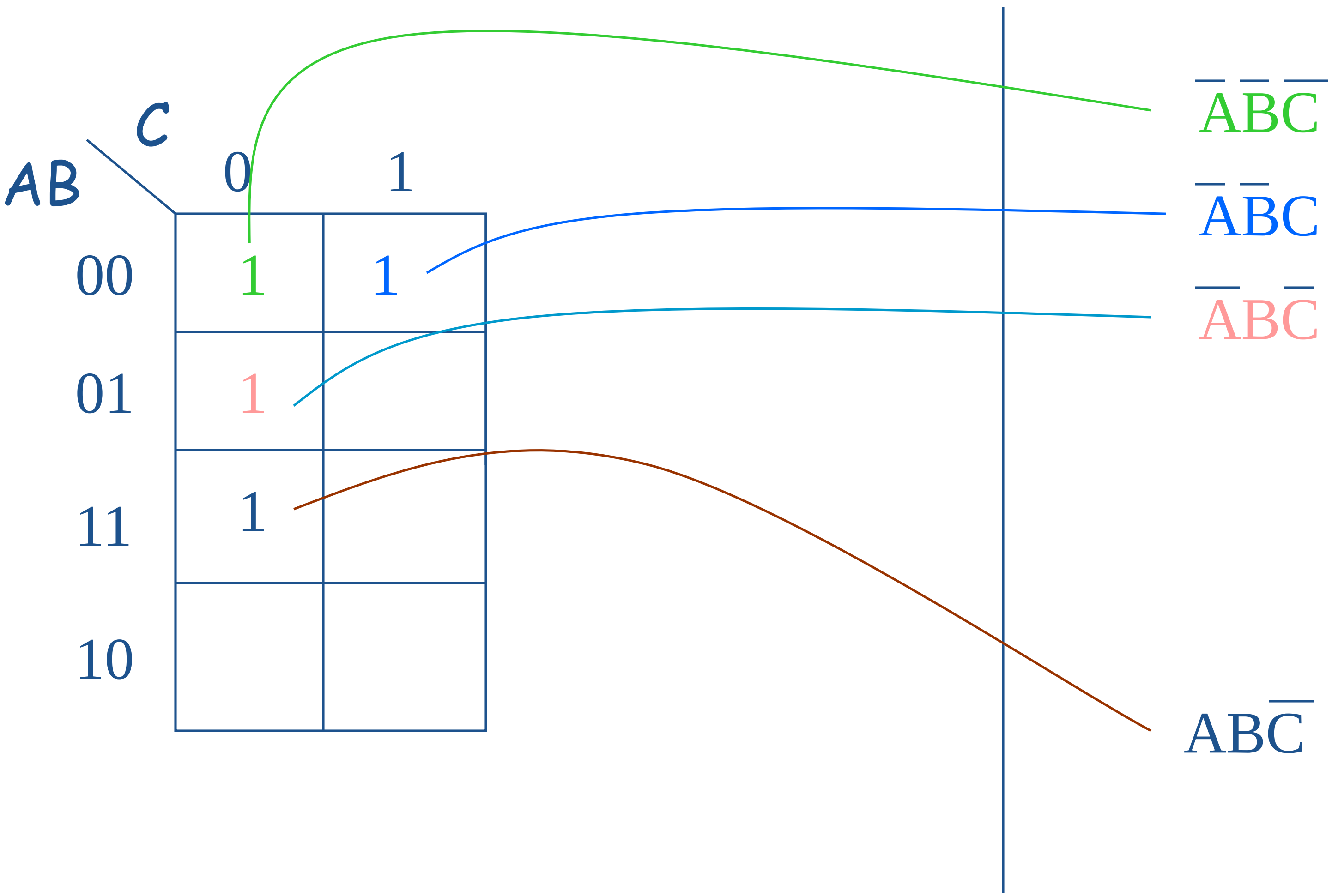
K-map: 3-variable example set-up

AB \ C	0	1
00		
01		
11		
10		

given:

A	B	C	X	
0	0	0	1	→ $\overline{A}\,\overline{B}\,\overline{C}$
0	0	1	1	→ $\overline{A}\,\overline{B}C$
0	1	0	1	→ $\overline{A}B\overline{C}$
0	1	1	0	
1	0	0	0	
1	0	1	0	
1	1	0	1	→ $AB\overline{C}$
1	1	1	0	

K-map: 3-variable example set-up



Four variable K-map: Example

AB \ CD	00	01	11	10
00				
01				
11				
10				

$$X = \overline{A}\,\overline{B}\,\overline{C}\,D + \overline{A}\,B\,\overline{C}\,D + A\,B\,\overline{C}\,D + A\,B\,C\,D$$

Four variable K-map: Example

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

$$X = \overline{A}\,\overline{B}\,\overline{C}\,D + \overline{A}\,B\,\overline{C}\,D + A\,B\,\overline{C}\,D + A\,B\,C\,D$$

How can we simplify using K-maps?

Use looping

looping is a process of combining 1's

Looping: Process of combining 1's

The looping is done in groups of …

Two (pair)

Four (quad)

Eight (octel)

1) Looping: Pair (2 … 1's)

- Looping a pair of adjacent 1's in a K-map table eliminates one variable that appears in *complemented* (A') and *uncomplemented* (A) form.

Uniting Theorem

- Looping a pair of adjacent 1's in a K-map table eliminates one variable that appears in *complemented* (A') and *uncomplemented* (A) form.

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

Example1: 2 logically adjacent 1's

AB \ C	0	1
00		
01	1	
11	1	
10		

X = ?

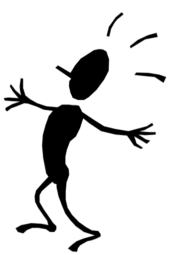
Example1

AB \ C	0	1
00		
01	1	
11	1	
10		

$$X = \bar{A}B\bar{C} + AB\bar{C}$$

$$= B\bar{C}(\bar{A} + A)$$

$$= B\bar{C}$$



Logically adjacent …

- Two terms (minterms) are logically adjacent if they differ in only one variable position (A) (Gray Code)

$$X = \overline{A}B\overline{C} + AB\overline{C}$$
$$= B\overline{C}(\overline{A} + A)$$
$$= B\overline{C}$$

- Logically adjacent terms can be looped (combined)
- A'BC', ABC' are minterms (m5, m6)

Example2

AB \ C	0	1
00		
01	1	1
11		
10		

X = ?

Example2

AB \ C	0	1
00		
01	1	1
11		
10		

$$X = \overline{A}\,B$$

Example3

AB \ C	0	1
00	1	
01		
11		
10	1	

X = ?

Cyclic property…

AB \ C	0	1
00	1	
01		
11		
10	1	

$$X = \overline{B}\,\overline{C}$$

Top and bottom rows are considered to be logical adjacent

Example4

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

X = ?

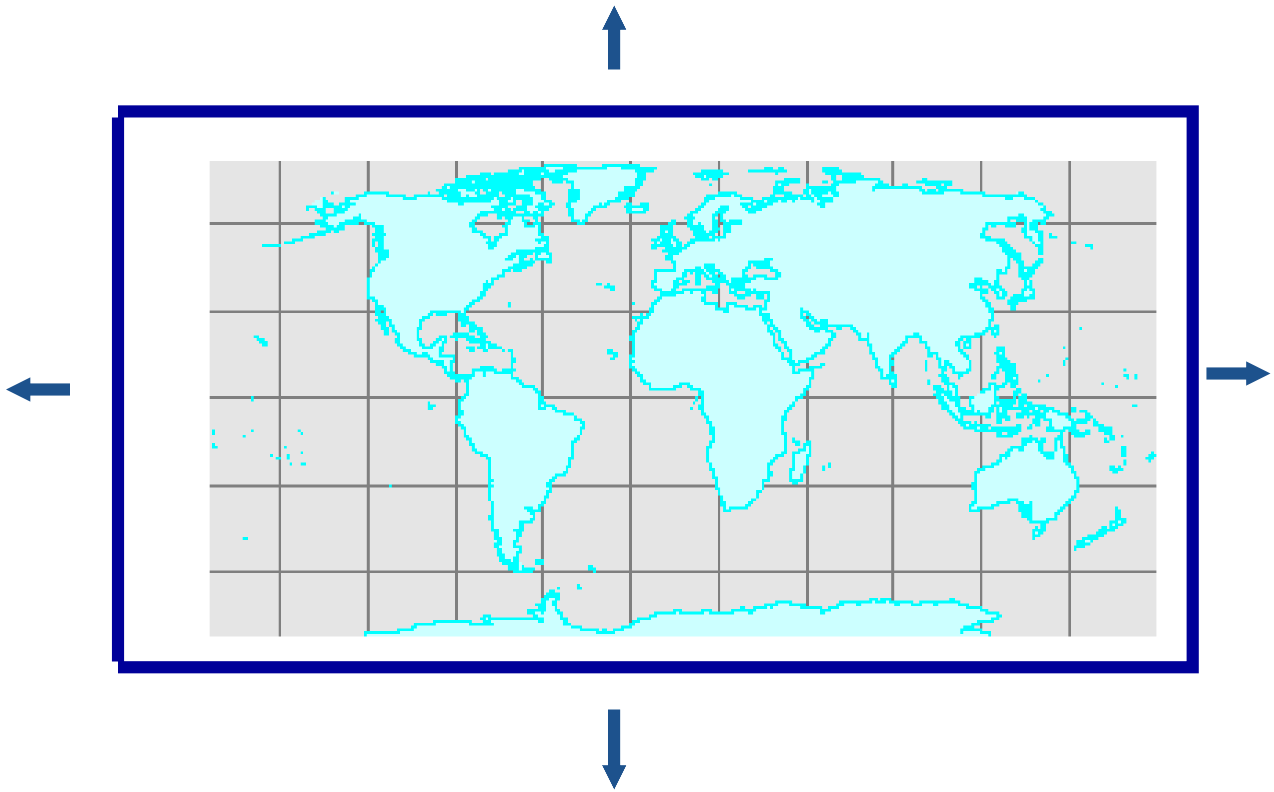
Cyclic property … again

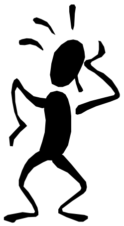
AB \ CD	00	01	11	10
00			1	1
01				
11				
10	1			1

$$X = A\overline{B}\,\overline{D} + \overline{A}\,\overline{B}C$$

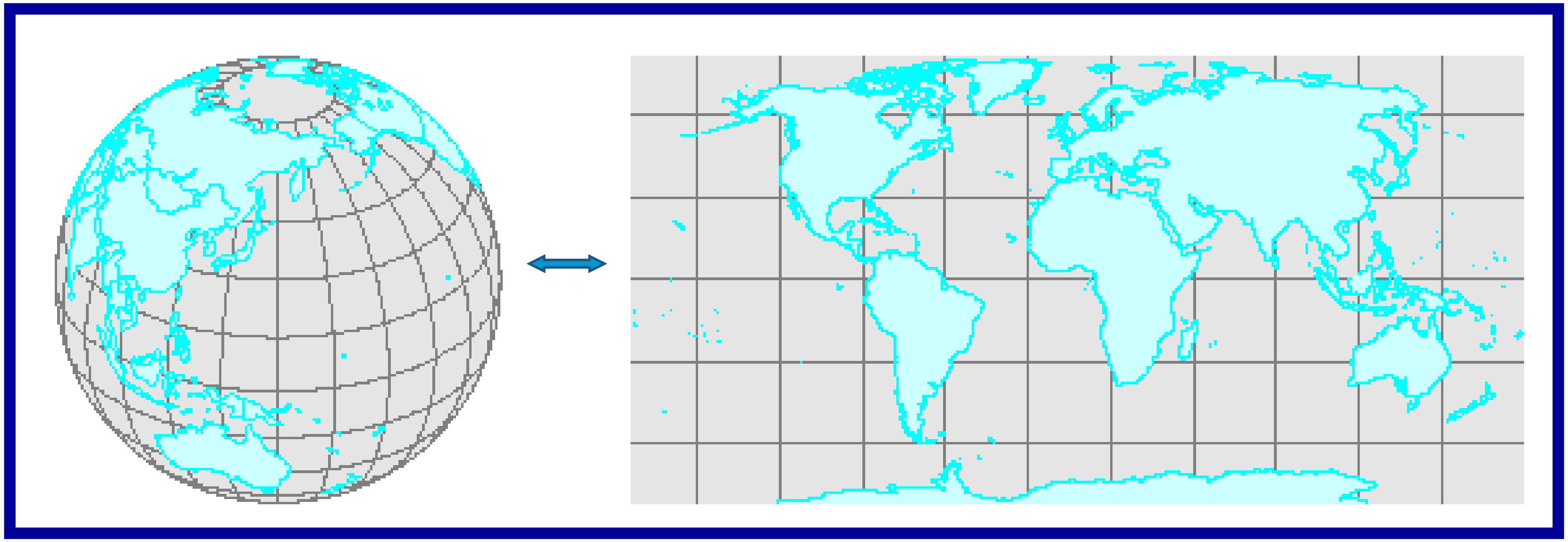
Left and right columns are considered to be logical adjacent…

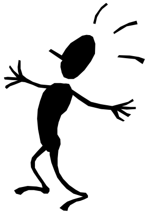
Adjacent left-right and top-bottom





Earth





2) Looping: Quad (4 … 1's)

- Looping (Combining) a quad, of logically adjacent 1's in a K-map, eliminates two variables that appear in complemented and uncomplemented form.

3-variable K-Map: Example1

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

X = ?

3-variable K-Map: Example1

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

$X = C$

Four variable K-map: Example2

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

X = ?

Four variable K-map: Example2

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

X = AB

Four variable K-map: Example3

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

$X = ?$

Four variable K-map: Example3

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

X = BD

Four variable K-map: Example4

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

X = ?

Left and Right pairs are adjacent

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

$$X = A\overline{D}$$

Four variable K-map: Example5

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

X = ?

Cyclic property: All 1's are adjacent

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

$$X = \overline{D}\,\overline{B}$$

3) Looping: Octel (8 … 1's)

- Looping (combining) an octel, of logically adjacent 1's, in a K-map eliminates three variables that appear in complemented and uncomplemented form
- In general, looping 2^m terms…eliminates m variables.

Four variable K-map: Example1

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

X = ?

Solution

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

X = B

Four variable K-map: Example2

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

X = ?

Solution

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

$$X = \overline{C}$$

Four variable K-map: Example3

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

X = ?

Solution

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

$$X = \overline{B}$$

Four variable K-map: Example4

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

X = ?

Solution

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

$$X = \overline{D}$$

More Examples-1

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

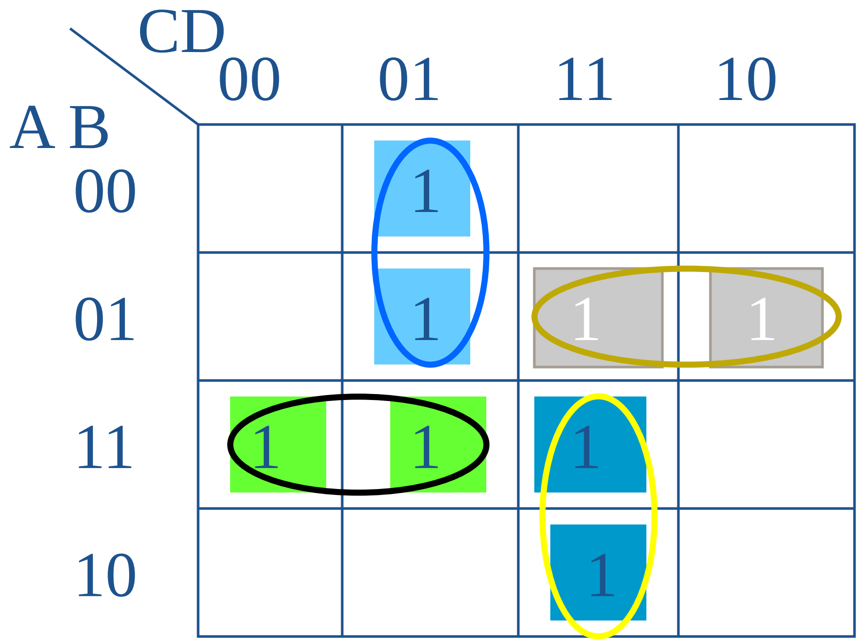
X = ?

Looping…

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

$$X = \overline{A}\,\overline{C}D + \overline{A}BC + AB\overline{C} + ACD + BD$$

BD is not needed

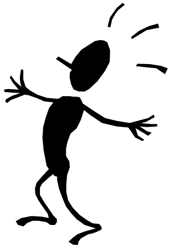


$$X = \bar{A}\bar{C}D + \bar{A}BC + AB\bar{C} + ACD + \cancel{BD}$$

Optimal Minimal Simplification

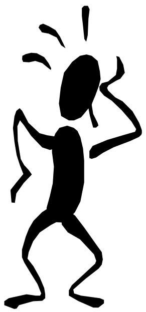
AB \ CD	00	01	11	10
00		1		
01		1	1	1
11	1	1	1	
10			1	

$$X = \overline{A}\,\overline{C}D + \overline{A}BC + AB\overline{C} + ACD$$



More Examples-2

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



X = ?

Minimal simplification

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

$$X = ACD + BD + \overline{A}\,\overline{B}C\overline{D}$$

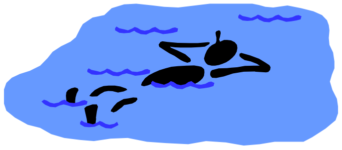
More Examples-3

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

X = ?

Minimal simplification

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



$$X = \overline{A}CD + \overline{A}B + B\overline{C}$$

Summary: Looping (K-Map)

- Loop the isolated 1's (those not logically adjacent to any other 1's). Look for the 1's that are adjacent to any loops and loop any pair containing such 1's. Each 1 must be looped at least once. However, it may be covered more than once (optimal).
  - Loop any octels (optimal)
  - Loop any quads (optimal)
  - Loop any pairs (optimal)
  - Form the OR sum of all terms in the loops