ENGIN 112

Intro to Electrical and Computer Engineering

Lecture 8

Minimization with Karnaugh Maps

Overview

- K-maps: an alternate approach to representing Boolean functions
- K-map representation can be used to minimize Boolean functions
- Easy conversion from truth table to K-map to minimized SOP representation.
- Simple rules (steps) used to perform minimization
- Leads to minimized SOP representation.
    - Much faster and more more efficient than previous minimization techniques with Boolean algebra.

Karnaugh maps

- **Alternate way of representing Boolean function**
  - **All rows of truth table represented with a square**
  - **Each square represents a minterm**
- **Easy to convert between truth table, K-map, and SOP**
  - **Unoptimized form: number of 1’s in K-map equals number of minterms (products) in SOP**
  - **Optimized form: reduced number of minterms**

$F = S(m_0,m_1) = x'y + x'y'$

			y
	x \ y	0	1
	0	x’y’	x’y
x	1	xy’	xy

x \ y	0	1
0	1	1
1	0	0

x	y	F
0	0	1
0	1	1
1	0	0
1	1	0

Karnaugh Maps

- **A Karnaugh map is a graphical tool for assisting in the general simplification procedure.**

- **Two variable maps.**

A \ B	0	1
0	0	1
1	1	0

$$F = AB' + A'B$$

A \ B	0	1
0	0	1
1	1	1

$$F = AB + A'B + AB'$$

- **Three variable maps.**

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

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

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

Rules for K-Maps

- We can reduce functions by circling 1's in the K-map
- Each circle represents minterm reduction
- Following circling, we can deduce minimized and-or form.

Rules to consider

- Every cell containing a 1 must be included at least once.
- The largest possible "power of 2 rectangle" must be enclosed.
- The 1's must be enclosed in the smallest possible number of rectangles.

Example

Karnaugh Maps

- **A Karnaugh map is a graphical tool for assisting in the general simplification procedure.**

- **Two variable maps.**

A \ B	0	1
0	0	1
1	1	0

$$F = AB' + A'B$$

A \ B	0	1
0	0	1
1	1	1

$$F = AB + A'B + AB'$$

$$F = A + B$$

- **Three variable maps.**

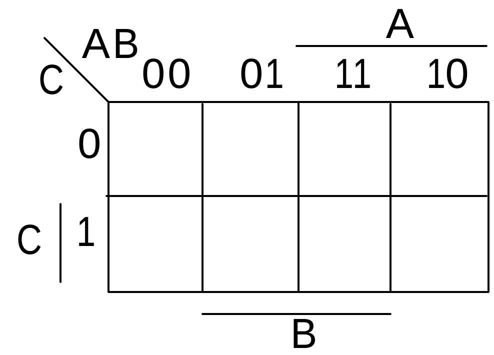
A \ BC	00	01	11	10
0	0	1	0	1
1	1	1	1	1

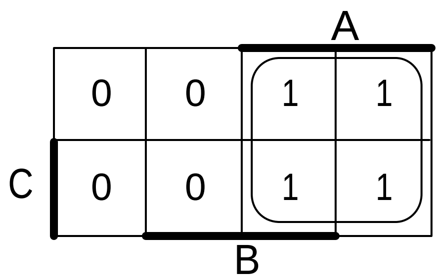
$$F = A + B'C + BC'$$

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

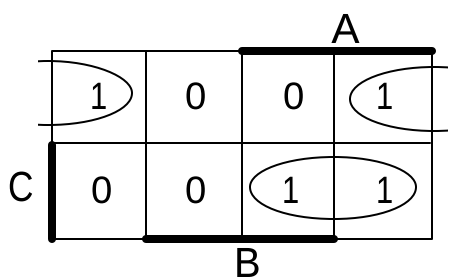
Karnaugh maps

- **Numbering scheme based on Gray–code**
  - **e.g., 00, 01, 11, 10**
  - **Only a single bit changes in code for adjacent map cells**
  - **This is necessary to observe the variable transitions**





$G(A,B,C) = A$



$F(A,B,C) = \Sigma m(0,4,5,7) = AC + B'C'$

More Karnaugh Map Examples

- **Examples**

b \ a	0	1
0	0	1
1	0	1

f = a

b \ a	0	1
0	1	1
1	0	0

g = b'

c \ ab	00	01	11	10
0	0	0	1	0
1	0	1	1	1

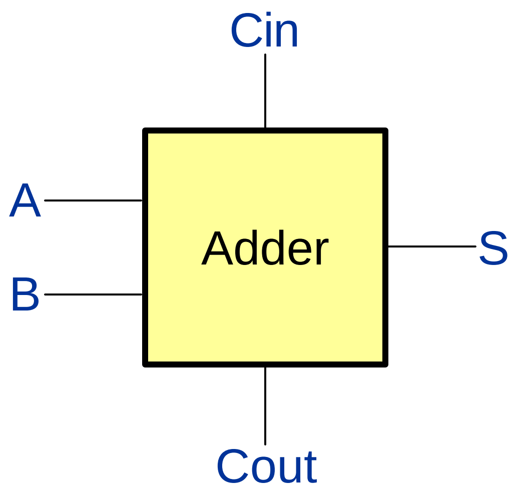
cout = ab + bc + ac

c \ ab	00	01	11	10
0	0	0	1	1
1	0	0	1	1

f = a

1. Circle the largest groups possible.
2. Group dimensions must be a power of 2.
3. Remember what circling means!

Application of Karnaugh Maps: The One-bit Adder



A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

How to use a Karnaugh Map instead of the Algebraic simplification?

S = A'B'Cin + A'BCin' + A'BCin + ABCin

Cout = A'BCin + A B'Cin + ABCin' + ABCin

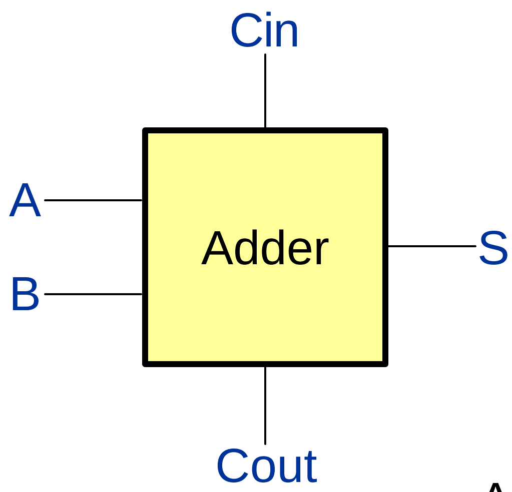
= A'BCin + ABCin + AB'Cin + ABCin + ABCin' + ABCin

= (A' + A)BCin + (B' + B)ACin + (Cin' + Cin)AB

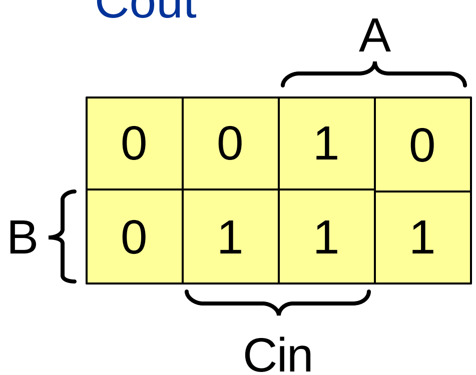
= 1·BCin + 1· ACin + 1· AB

= BCin + ACin + AB

Application of Karnaugh Maps: The One-bit Adder



A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

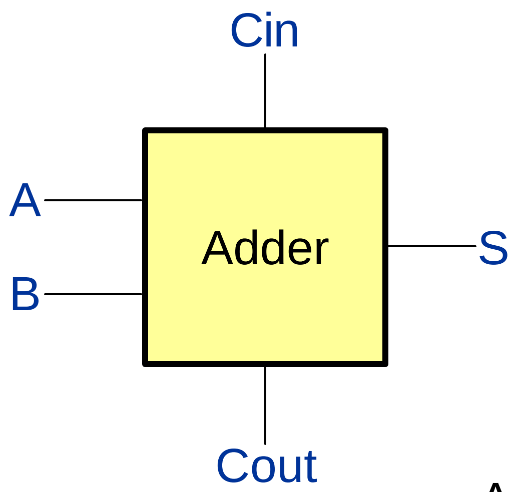


0	0	1	0
0	1	1	1

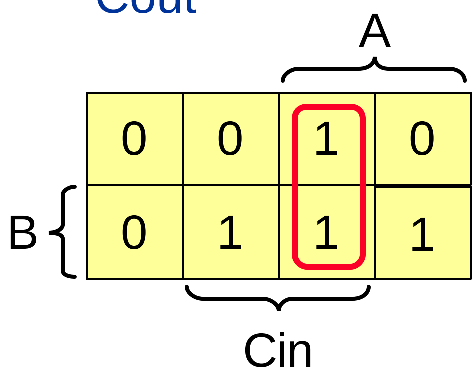
Karnaugh Map for Cout

Now we have to cover all the 1s in the Karnaugh Map using the largest rectangles and as few rectangles as we can.

Application of Karnaugh Maps: The One-bit Adder



A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

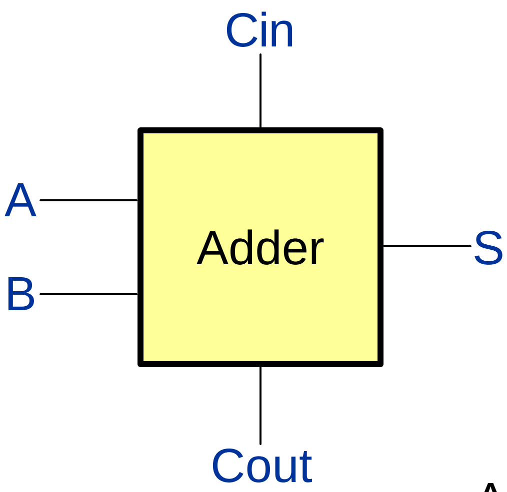


Karnaugh Map for Cout

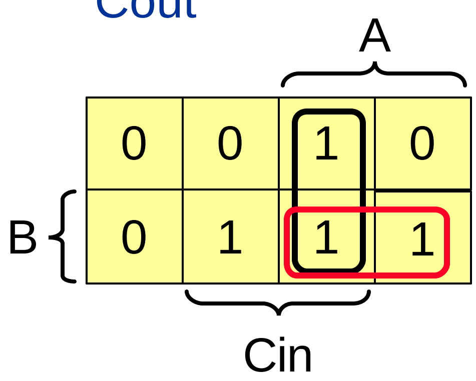
Now we have to cover all the 1s in the Karnaugh Map using the largest rectangles and as few rectangles as we can.

Cout = ACin

Application of Karnaugh Maps: The One-bit Adder



A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

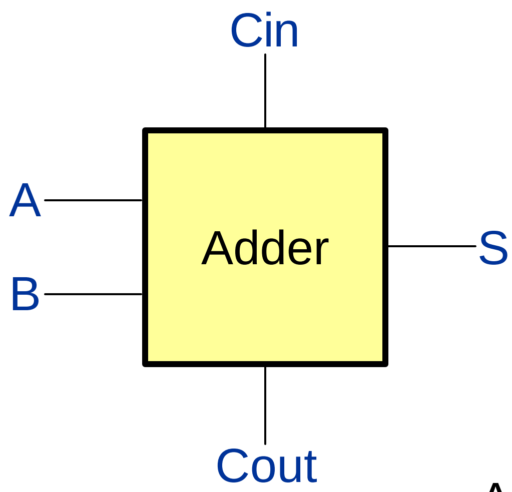


Karnaugh Map for Cout

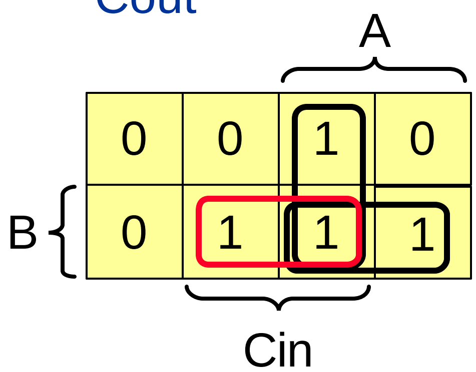
Now we have to cover all the 1s in the Karnaugh Map using the largest rectangles and as few rectangles as we can.

Cout = Acin + AB

Application of Karnaugh Maps: The One-bit Adder



A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

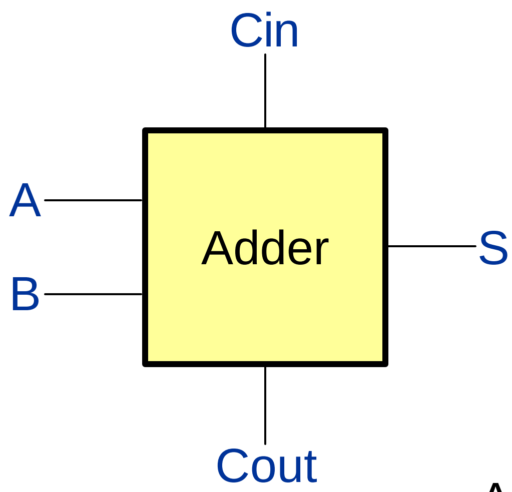


Karnaugh Map for Cout

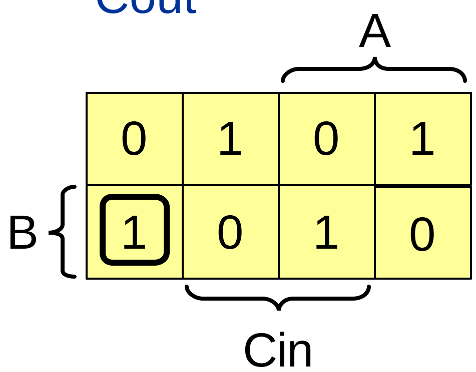
Now we have to cover all the 1s in the Karnaugh Map using the largest rectangles and as few rectangles as we can.

Cout = ACin + AB + BCin

Application of Karnaugh Maps: The One-bit Adder



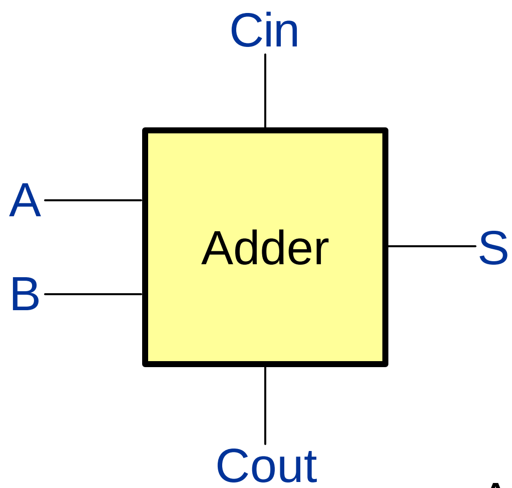
A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



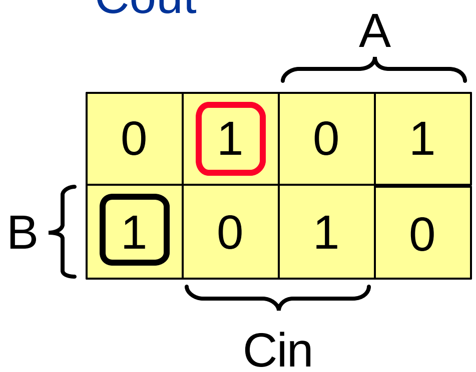
Karnaugh Map for S

S = A'BCin'

Application of Karnaugh Maps: The One-bit Adder



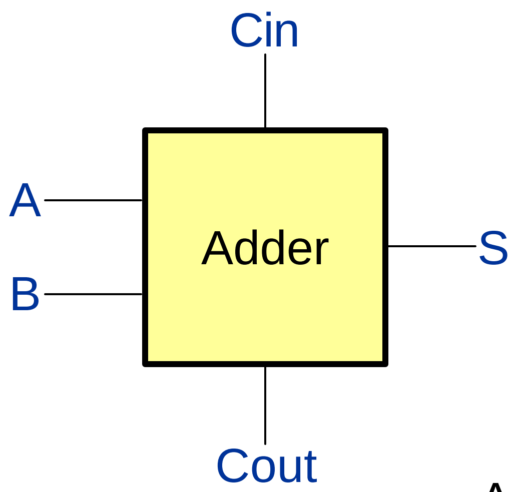
A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



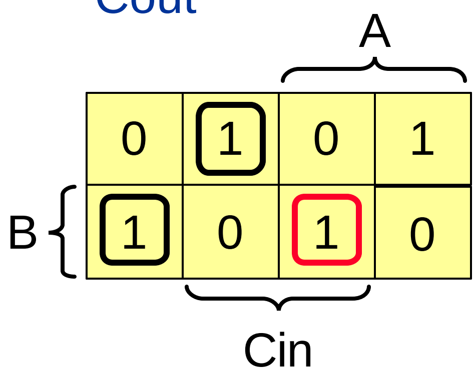
Karnaugh Map for S

S = A'BCin' + A'B'Cin

Application of Karnaugh Maps: The One-bit Adder



A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

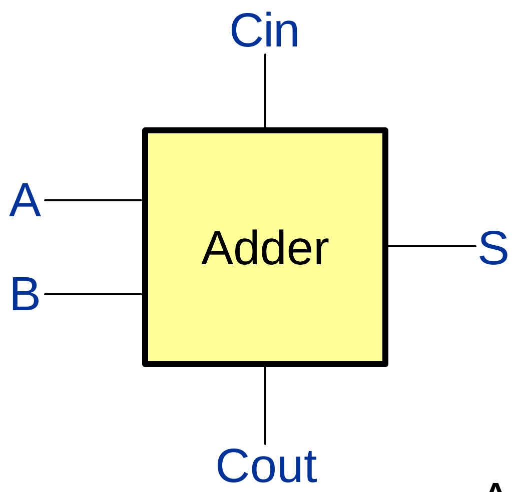


Karnaugh Map for S

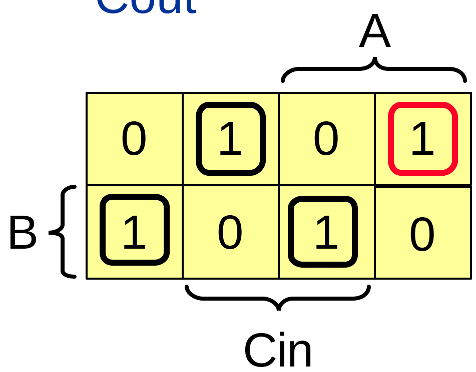
S = A’BCin’ + A’B’Cin + ABCin

Application of Karnaugh Maps: The One-bit Adder

Can you draw the circuit diagrams?



A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



Karnaugh Map for S

S = A'BCin' + A'B'Cin + ABCin + AB'Cin'

No Possible Reduction!

Summary

- Karnaugh map allows us to represent functions with new notation
- Representation allows for logic reduction.
  - Implement same function with less logic
- Each square represents one minterm
- Each circle leads to one product term
- Not all functions can be reduced
- Each circle represents an application of:
  - Distributive rule -- $x(y + z) = xy + xz$
  - Complement rule – $x + x' = 1$