Reading 07

9.1 Two-level combinational logic simplification

- Logic simplification (or logic minimization) means to simplify a Boolean expression before converting the expression to a circuit to yield a smaller circuit.

- Look for $i(j + j')$ opportunities since $i(j+j') = i(1) = i$

↳ Ex $y = cd + c$

$y = cd + c(d+d')$

$y = cd + cd + cd'$ Idempotent Property

$y = cd + cd'$

$y = c(d+d')$

$y = c(1)$

$y = c$

- You can do this:

$ab + a'b + a'b'$

↳ $ab + a'b + a'b + a'b'$

9.2 K-maps: Introduction

- A K-map is a graphical function representation that eases the simplification process for expressions involving a few variables by adjacently placing minterms that differ by exactly one variable.
  - ↳ A K-map is a reoriented truth table.
  - ↳ Ex:

y = a'b' + ab'

a \ b	0	1
0		a'b
1	ab'	ab

a	b	y
0	0	1
0	1	0
1	0	1
1	1	0

a \ b	0	1
0	1	0
1	1	0

  - A 1 is placed in cells for function midterms.

- K-maps make it obvious to see i(j+j') opportunities.
  - ↳ If there are 2 adjacent 1's, the designer can eliminate the differing variable
  - ↳ Ex:

y = ab' + ab
y = a

a \ b	0	1
0	0	0
1	1	1

ab' + ab
a(b' + b)
a(1)
a

  - ↳ Ex!

y = ab + a'b + a'b'
y = a' + b

a \ b	0	1
0	1	1
1	0	1

a'b' + a'b
a'(b' + b)
a'

a'b + ab
b(a' + a)
b

- Rules for simplifying sum-of-minterms with a K-map:
  - Rule 1: Cover every 1 at least once using circles. Add the circle's term to the expression.
  - Rule 2: Use the fewest and largest circles possible to achieve the simplest expression.

9.3 3 and 4 variable K-maps:

- A K-map for 3 variables has 2 variables across the top.
  - ↳ Ex:

y = ab'c' + ab'c + a'bc + a'bc'

just showing what minterm corresponds to what box

a \ bc	00	01	11	10
0		a'b'c	a'bc	
1	ab'c'		abc	abc'

a \ bc	00	01	11	10
0	0	0	1	1
1	1	1	0	0

a	b	c	y
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

↳ Ex Simplification of 3 variable K-map:

y = ab'c' + ab'c + a'bc + a'bc'

y = ab' + a'b

a \ bc	00	01	11	10	
0	0	0	1	1	a'b
1	1	1	0	0	
	ab'				

ab'c' + ab'c

ab'(c' + c)
ab'(1)
ab'

a'bc + a'bc'

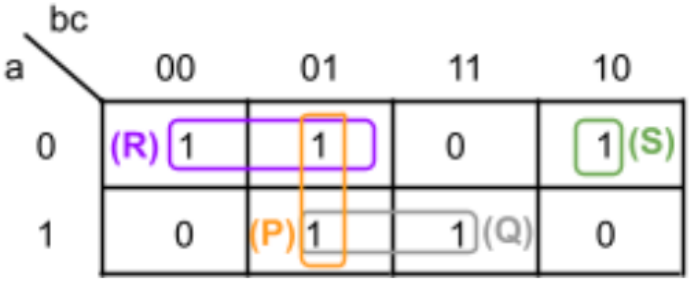
a'b(c + c')
a'b(1)
a'b

↳ Ex

a \ bc	00	01	11	10
0	1 (M)	0	0	1
1	0	1 (P)	0	0

- Purple loop is VALID
- Orange loop is NOT VALID

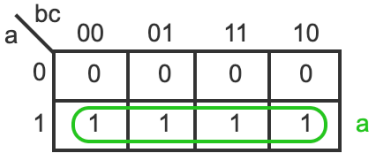
↳ Ex :



- Circle P is NOT necessary since both of the ones it contains have already been captured by other circles.

- A circle may encompass four adjacent 1s, removing two variables rather than just one

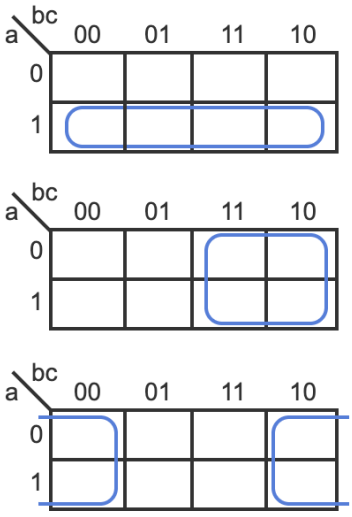
↳ Ex



y = ab'c' + ab'c + abc + abc'

y = a (b'c' + b'c + bc + bc')

y = a(b'(c' + c) + b(c + c'))
y = a(b'(1) + b(1))
y = a(b + b')
y = a(1)
y = a



- 4-variable K-map

↳ Top and bottom rows are adjacent (as are left and right columns). Valid circle sizes are 1, 2, 4, 8, or 16 cells.

↳ Ex