3.19 K-maps: Introduction

K-maps

A ***K-map*** is a graphical function representation that eases the simplification process for expressions involving a few variabl placing minterms that differ in exactly one variable. . K-map is short for ***Karnaugh map***. "Map" is used like how a country m next to each other. A K-map lays out minterms instead.

A K-map lays out possible minterms as adjacent cells (boxes). Adjacent minterm cells differ by exactly one variable. Each f cell gets a 1; other cells get 0.

A K-map is a reoriented truth table.

PARTICIPATION ACTIVITY 3.19.1: A 2-variable K-map: Adjacent cells differ in exactly one variable.

Start 2x speed

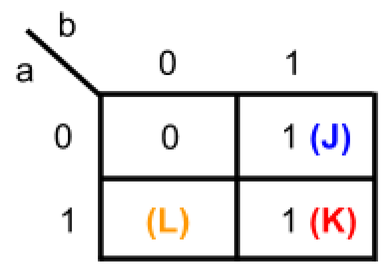
y = a'b' + ab'

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

PARTICIPATION ACTIVITY 3.19.2: 2-variable K-map basics.



Given function y = ab + a'b, represented in the above figure's K-map.

1) (J) corresponds to which minterm?

Check Show answer

2) (K) corresponds to which minterm?

Check Show answer

3) (L) should have what value (0 or 1)?

Check Show answer

4) Cells (J) and (K) differ in what variable: a, or b?

Check Show answer

5) Cells (L) and (K) differ in what variable: a, or b?

Check Show answer

6) Cells (L) and (J) differ in how many variables?

Check Show answer

Simplifying an expression with a K-map

Because adjacent minterm cells differ in exactly one variable, a K-map's key benefit is to make i(j + j') simplification opportu
Adjacent 1's are an i(j + j') opportunity. Circling two adjacent 1's graphically represents the algebraic simplification i(j + j') = i
drawing such a circle, a designer can write a product term with the differing variable omitted.

PARTICIPATION ACTIVITY 3.19.3: Simplification with a 2-variable K-map: i(j + j') opportunities are obvious.

Start 2x speed

y = ab' + ab

y = a

a \ b	0	1
0	0	0

ab' + ab
a(b' + b)

1	1	1

a(1)
a

A powerful feature of a K-map is how easily replicating a minterm is achieved (recall an earlier section's example), merely b twice.

PARTICIPATION ACTIVITY 3.19.4: Circling a 1 twice is like replicating a minterm to create i(j + j') opportunities.

Start 2x speed

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

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

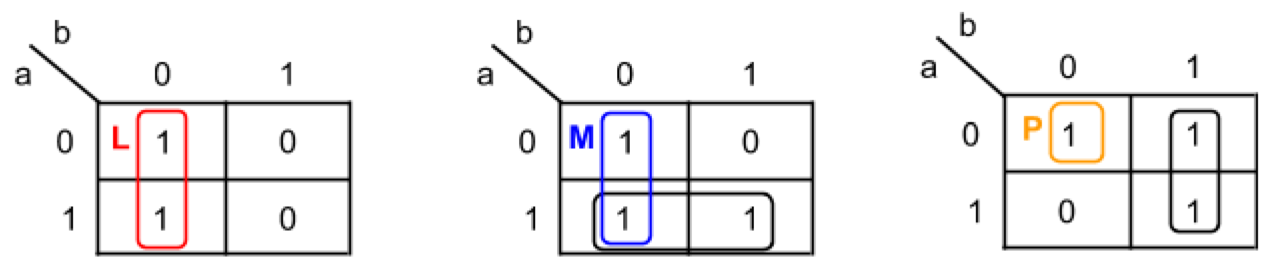
a'b' + a'b
a'(b' + b)
a'

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

Table 3.19.1: Rules for simplifying a sum-of-minterms expression with a K-map.

Rule 1:	*Cover every 1* at least once using circles. Add circle's term to expression.
Rule 2:	*Use fewest and largest circles* possible, to achieve simplest expression.

PARTICIPATION ACTIVITY 3.19.5: Basic 2-variable K-map.



Consider the K-maps in the figure above.

1) Circle (L) is what simplified term?

Check Show answer

2) Is circle (M) necessary? Type: yes or no

Check Show answer

3) Is circle (P) a good circle? Type: yes or no

Check Show answer

Example: Out-of-bed alarm

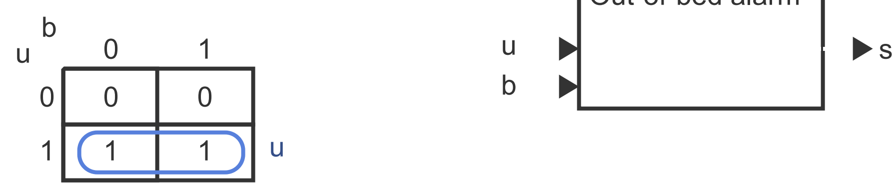
An example in an earlier section involved sounding an alarm (s = 1) if a person was up from bed (u = 1) and a button presse
person was up and button was not pressed. The captured equation was s = ub + ub'. A K-map can be used to simplify the e

PARTICIPATION ACTIVITY 3.19.6: Simplifying with a K-map: Out-of-bed alarm.

Start 2x speed

s = ub + ub'

s = u



PARTICIPATION ACTIVITY 3.19.7: Out-of-bed alarm system.

Consider the example above.

1) The designer captured behavior as s = ub + ub', but simplification yielded s = u. Thus, the designer incorrectly captured the original behavior.

- True
- False

2) The simplification on the K-map was quite obvious.

True

False

Example: Motion-sensing light

An earlier section captured a motion-sensing lamp's behavior and then simplified algebraically. That example can more-eas
by a K-map instead.

PARTICIPATION ACTIVITY 3.19.8: Simplifying with a K-map: Motion-sensing light.

Start 2x speed

Inputs: m: motion sensed
t: test mode
Outputs: i: illuminate lamp
Goal: Illuminate lamp if motion and not test mode,
or if test mode and no motion, or if test mode and motion

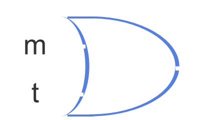
Algebraic simplification

i = mt' + tm' + tm
i = mt' + m't + mt
i = mt' + m't + mt + mt
i = mt' + mt + m't + mt
i = m(t' + t) + (m' +m) t
i = m(1) + (1)t
i = m(1) + t(1)
i = m + t

K-map simplification

i = mt' + m't + mt
i = m + t

m \ t	0	1
0		1
1	1	1



PARTICIPATION ACTIVITY 3.19.9: Motion-sensing light system.

Consider the example above.

1) How many equations were involved using algebraic simplification?
   - 2
   - 8

2) How many circles were drawn using K-map simplification?
   - 2
   - 3

3) K-maps help with simplification by not obeying algebraic properties.
   - True
   - False

Provide feedback on this section