# 3.20 Prime implicants and minimal covers

## On-sets and implicants

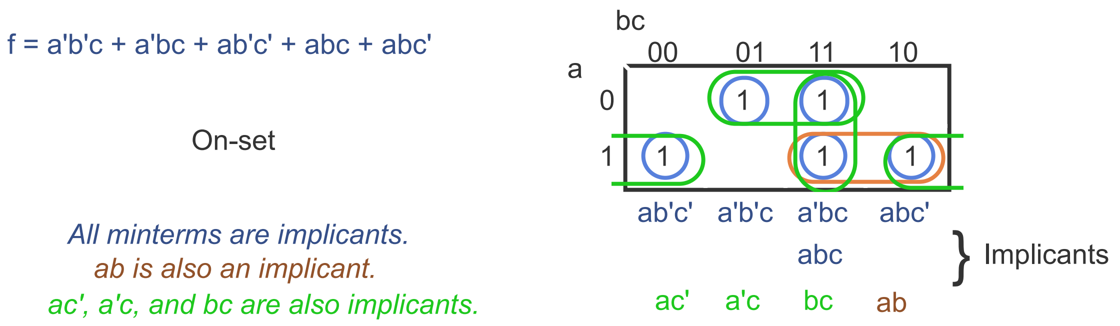
Several mathematical concepts underlie techniques for two-level logic minimization.

- A function's ***on-set*** is the set of minterms that define when the function is 1. Ex: f = a'b'c + a'bc + ab'c' + abc + abc' has the function's on-set, as listed.
- A term ***covers*** a minterm if the term evaluates to 1 whenever the minterm does. Ex: Term ab covers minterm abc, as abc'.
- An ***implicant*** of a function is a term that covers only minterms in the function's on-set. Ex: For the above function, ab due to covering abc and abc', both of which are in the function's on-set , and covering no other minterms. a'b' is not ar to covering minterm a'b'c', which is not in the function's on-set.

On a K-map, each 1 represents a minterm in a function's on-set, and each valid circle (not necessarily largest) is an implicar

**PARTICIPATION ACTIVITY** 3.20.1: Implicants of a function. Each valid K-map circle represents an implicant.

Start 2x speed


f = a'b'c + a'bc + ab'c' + abc + abc'
On-set
All minterms are implicants.
ab is also an implicant.
ac', a'c, and bc are also implicants.
bc
a
00
01
11
10
0
1
1
1
1
1
1
1
ab'c'
a'b'c
a'bc
abc'
abc
Implicants
ac'
a'c
bc
ab

**PARTICIPATION ACTIVITY** 3.20.2: Identifying implicants of a function.

Consider the following K-map for function F = ab'c' + a'bc + abc + a'bc' + abc'.

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

1) The function's on-set consists of how many minterms?

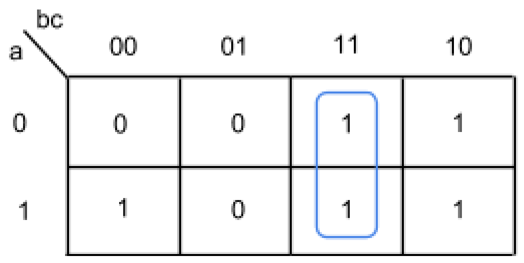
- ○ 2
- ○ 5
- ○ 8

2) What implicant of the function does the shown circle represent?

| a \ bc | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 |
| 1 | $\boxed{1}$ | 0 | 1 | 1 |

- ○ a
- ○ ab'c'
- ○ abc

3) What implicant of the function does the shown circle represent?



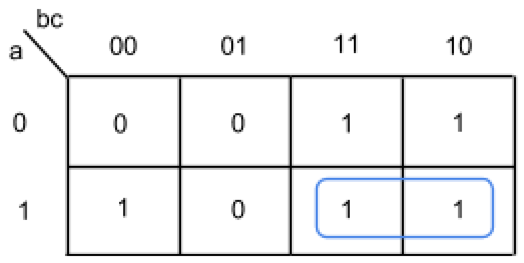
| a \ bc | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 |

○ bc

○ a'bc + abc

○ Not an implicant

4) What minterms are covered by the shown implicant?



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

○ ab

○ abc, abc'

○ Not an implicant

5) Is b an implicant of the function?

○ Yes

○ No

6) Is a' an implicant of the function?

○ Yes

○

No

7) How many possible implicants does the function have?

- ○ 2
- ○ 5
- ○ 11

## Prime implicants

A ***prime implicant*** of a function is an implicant that cannot have a literal removed without becoming a non-implicant. On a K representation of a function, each *largest* possible valid circle represents a *prime* implicant.

**PARTICIPATION ACTIVITY** 3.20.3: Prime implicants: Largest possible K-map circles.

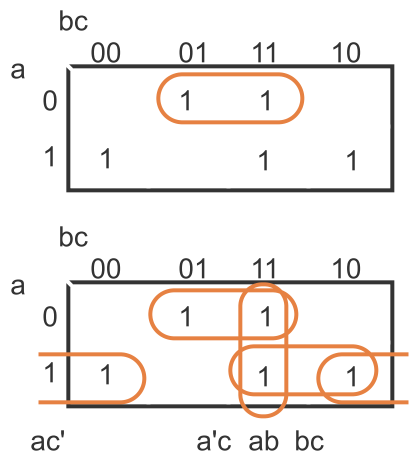
Start ☐ 2x speed

f = a'b'c + a'bc + ab'c' + abc + abc'

a'b'c: implicant, but not prime (can remove b')

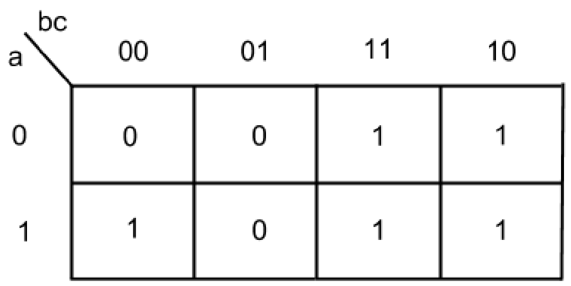
a'c: prime implicant (can't remove any literal)

a' c



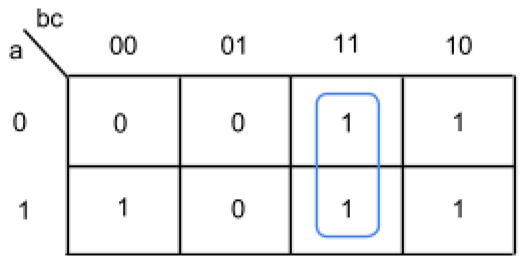
**PARTICIPATION ACTIVITY** 3.20.4: Prime implicants.

Consider the following K-map for function F = ab'c' + a'bc + abc + a'bc' + abc'.



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

1) Does this circle represent a prime implicant?

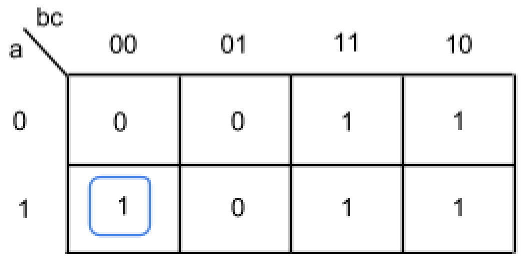


| a \ bc | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 |

○ Yes

○ No

2) Does this circle represent a prime implicant?



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

○ Yes

○

No

## Essential prime implicants and minimal covers

A minimal cover is useful to create the smallest circuit for the function. A function's ***minimal cover*** is an expression for the the fewest terms, each with the fewest literals.

Defining prime implicants (PI's) above was useful because each term in a minimum cover *must* be a prime implicant (else, minimal). On a K-map, a minimal cover is a cover using the fewest, largest circles, and largest circles are PI's.

An ***essential prime implicant*** is the only prime implicant to cover a particular minterm in a function's on-set. On a K-map, ar largest circle that is the only circle to cover a particular 1. Essential PI's must be in the function's cover.

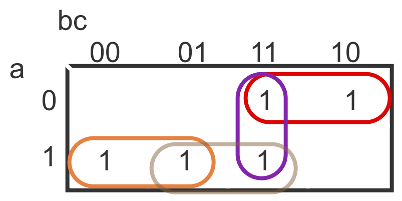
An algorithm to create a minimal cover is:

1. *Generate PI's:* Find all PI's by drawing all largest possible circles.
2. *Add essential PI's:* Find any essential PI's and add them to the function's cover.
3. *Cover remaining:* Select PI's to cover any 1's not covered by the essential PI's.

More than one minimal cover may exist, as in the animation below (either the purple or brown circle could be used).

PARTICIPATION ACTIVITY 3.20.5: Essential prime implicants and minimal cover.

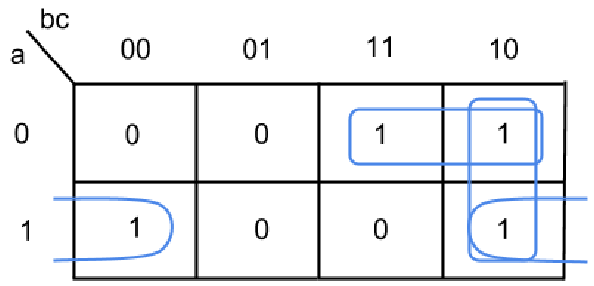
Start 2x speed



Minimal cover: ab' + a'b + bc

**PARTICIPATION ACTIVITY** 3.20.6: Essential prime implicants and minimal covers.

Consider the given K-map and prime implicants.



1) Which prime implicant is essential?

- ○ ac'
- ○ bc'
- ○ b

2) Which prime implicant is essential?

- ○ bc'
- ○ a'b
- ○ a'bc

3) After including essential prime implicants in the minimal cover, how many remaining prime implicants must be added?

- ○ 0
- ○

1

○ 2

## K-map minimization usually combines steps

*When minimizing a function by hand using a K-map, designers typically combine the above steps, drawing essential PI circles first (covering obviously standalone 1's), then drawing the fewest largest circles to cover remaining 1's. But the above 3 steps are useful when automating the minimization like in the "Quine-McCluskey algorithm" (described in another section), which is especially important for larger functions such as those with 5 inputs or more.*

**Provide feedback on this section**