the `k-mapper` package 1.0.0

Karnaugh map
/ˈkɑːnɔː/
noun

a diagram consisting of a rectangular array of squares each representing a different combination of the variables in a Boolean function

introduction

`k-mapper` is a Typst package for adding customizable Karnaugh maps of 2 by 2, 2 by 4, and 4 by 4 grid sizes to your Typst projects.

This Manual has been typeset in Typst, using the `k-mapper` package, and is intended for the 1.0.0 version of `k-mapper`. See the source code on the Github repository for the project here.

using `karnaugh()`

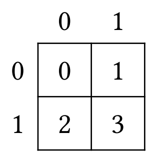
The main function of this package is the `karnaugh()` function, which allows you to create and customize all sizes of Karnaugh maps.

gray code position

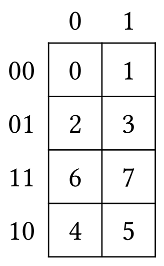
The position of implicants in `k-mapper` are declared via *Gray code position*. This is similar to Karnaugh map packages in LaTeX.

The Gray code position of a cell in a Karnaugh map can be determined by looking at the Gray code labels of the Karnaugh map: the Gray code position is the decimal equivalent of the binary number formed from the number(s) on the left and the number(s) on the top.

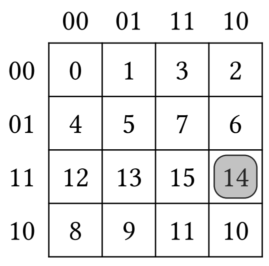
The empty maps below show each cell's Gray code position. Note that the Gray code position for a cell differs depending on the Karnaugh map's grid size.



	0	1
0	0	1
1	2	3



	0	1
00	0	1
01	2	3
11	6	7
10	4	5



	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

For example, the shaded cell above's Gray code position (`14`) can be determined by concatenating the binary numbers to its left on the y-axis (`11`) and above it on the x-axis (`10`), giving `1110` which equals `14` in decimal.

function arguments

name	default	description	examples
grid-size int	required	The size of the Karnaugh map's grid. This value can be only 4 (2 by 2), 8 (2 by 4), or 16 (4 by 4). Any other values will throw an error.	4 8 16

x-label
 content
	`$$`	The label (usually a variable name) to go on the top (x-axis) of the Karnaugh map.	`$A$`
			
			 `[foo]`

y-label
 content
	`$$`	The label (usually a variable name) to go on the left (y-axis) of the Karnaugh map.	`$B$`
			
			 `[bar]`

minterms
 (int)
 none
	none	The `array` of Gray code positions[1] where at that position is a minterm (`0`).
		 Mutually exclusive with `maxterms` and `manual-terms`.
			`(3, 4, 6)`
			
			 `(1, )`

maxterms
 (int)
 none
	none	The array of Gray code positions[1] where at that position is a maxterm (1).
		 Mutually exclusive with `minterms` and `manual-terms`.
			`(0, 1, 2, 3,`
			 `5, 11, 12`
			
			 `(7, )`

manual-terms
 (content)
 none
	none	The `array` of `content` in each cell in order of Gray-code position[1]. The length of this `array` *must* equal the `grid-size`.
		 Mutually exclusive with `minterms` and `maxterms`.
			`// Grid-size 4`
			 `(0, "X", 1, 1)`

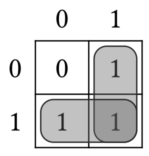
implicants
 ((int, int),)
	()	An `array` where each element is an array of two `int`s, where each `int` is a Gray code position[1] corner of a *rectangular* implicant.	`((0, 3), (1, 1))`
			
			 `((0, 2), )`

horizontal-implicants
 ((int, int),)
	()	An `array` where each element is an array of two `int`s, where each `int` is a Gray code position[1] corner of a	`// Grid-size 16`
			 `((0, 6), (8, 10))`

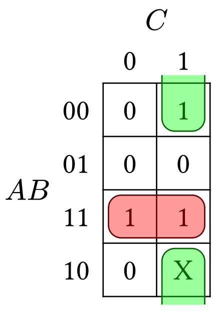
name	default	description	examples
		horizontal split implicant — that is, one which wraps around the vertical edges of the Karnaugh map.	
`vertical-implicants` `((int, int), )`	`()`	An `array` where each element is an `array` of two `int`s, where each `int` is a Gray code position[1] corner of a *vertical split* implicant — that is, one which wraps around the horizontal edges of the Karnaugh map.	`// Grid-size 8` `((0, 4), )` `// Grid-size 16` `((0, 9), (2, 10))`
`corner-implicants` `bool`	`false`	A `bool` which indicates whether the Karnaugh map contains a `corner split` implicant — that is, one which wraps around both vertical and horizontal edges of the Karnaugh map.	`true`
`cell-size` `length`	`20pt`	The size of an individual cell in the Karnaugh map.	`1cm`
`stroke-size` `length`	`0.5pt`	The stroke width of the Karnaugh map grid.	`0.2pt`
`colors` `(color)`	array of: red green blue cyan magenta yellow	An array of RGBA `colors` to be used in displaying implicants. The first implicant uses the first `color` in the array, the second implicant the second color, etc. If there are more implicants than there are colors, each subsequent implicant will use the least recently used color (i.e. it wraps around). By default, all `colors` in `colors` have alpha values of `100`.	`// Grayscale K-map` `(rgb( 200, 200, 200, 100 ), )`
`implicant-inset` `length`	`2pt`	The inset of implicants within each cell.	`3pt`
`edge-implicant-overflow` `length`	`5pt`	How much *split implicants* (horizontal, vertical, corner) overflow the bounds of the grid.	`2mm`

name	default	description	examples
`implicant-radius` `length`	`5pt`	The corner radius of implicants.	`3mm`

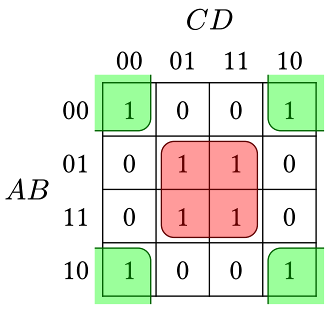
examples



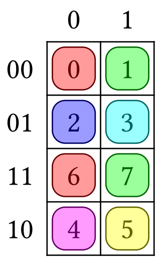
```
// Grayscale Karnaugh map
#karnaugh(
  4,
  minterms: (0, ),
  implicants: ((1, 3), (2, 3)),
  colors: (rgb(100, 100, 100, 100), ) // <-
)
```



```
#karnaugh(
  8,
  x-label: $C$,
  y-label: $A B$,
  manual-terms: (0, 1, 0, 0, 0, "X", 1, 1),
  implicants: ((6, 7), ),
  vertical-implicants: ((1, 5), )
)
```

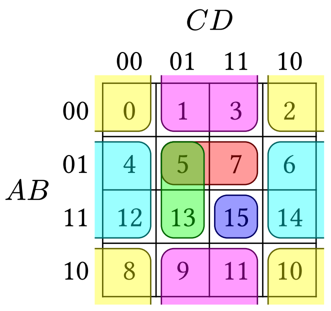


```
#karnaugh(
  16,
  x-label: $C D$,
  y-label: $A B$,
  maxterms: (0, 2, 5, 7, 13, 15, 8, 10),
  implicants: ((5, 15), ),
  corner-implicants: true
)
```

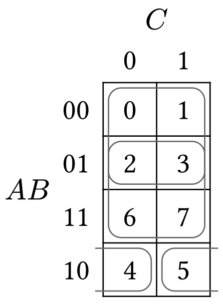


```
#karnaugh(
  8,
  manual-terms: (0, 1, 2, 3, 4, 5, 6, 7),
  implicants: (
    (0, 0), (1, 1), (2, 2), (3, 3),
    (4, 4), (5, 5), (6, 6), (7, 7)
  )
)
```

[1]See p. 1.



```
#karnaugh(
  16,
  x-label: $C D$,
  y-label: $A B$,
  manual-terms: (
    0, 1, 2, 3, 4, 5, 6, 7, 8,
    9, 10, 11, 12, 13, 14, 15
  ),
  implicants: ((5, 7), (5, 13), (15, 15)),
  vertical-implicants: ((1, 11), ),
  horizontal-implicants: ((4, 14), ),
  corner-implicants: true,
)
```


C
0 1
AB
00
01
11
10
0 1
2 3
6 7
4 5