



PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE  
ESCUELA DE INGENIERÍA  
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

IIC2343 — Arquitectura de Computadores 2019-2

## Tutorial – Diseño de Circuitos Combinacionales

Tutorial para diseñar Circuitos Combinacionales con Mapas de Karnaugh

### Introducción:

En el contexto del diseño de circuitos combinacionales, existen muchas formas de enfrentarse a la lógica booleana. Una forma sencilla de enfrentarse a esta lógica fue descubierta por Maurice Karnaugh, con lo que se conoce como **mapas de Karnaugh**.

Los mapas de Karnaugh te permite obtener la forma reducida de una función booleana con compuertas **and**, **or** y **not**, lo cual representará la salida de un bit.

A continuación se presentará el tratamiento de un problema de diseño, y su resolución en Vivado. Esta herramienta le será útil para el proceso de sus entregas. **Recordar que esta materia NO entra para las ies.**

### Obtener la información

#### Enunciado

Se requiere un programa que identifique si un número entero positivo es primo, este número debe ser menor o igual a 13. En caso de ser primo, debe encender un led.

##### 1. Identificar entradas y salidas.

Se identifica del enunciado las siguientes entradas y salidas:

- in: numero[4 bits]
- out: led[1 bit]

##### 2. Obtener la tabla de verdad

La tabla de verdad se observará todas las posibles combinaciones de las 4 entradas:

Las **X** corresponden al dato llamado: *Don't Care*, son datos que no importan su valor pues no corresponden al rango del problema. En este caso, los números mayores de 13 de 4 bits.

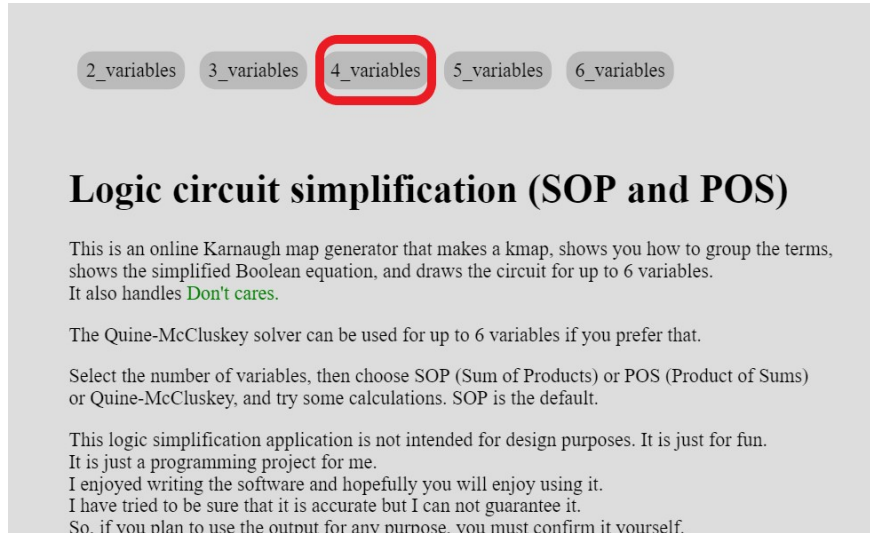
Número	binario[3:0]				Led
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	x
15	1	1	1	1	x

Table 1: Tabla de verdad

### 3. Obtener el Mapa de Karnaugh

Para obtener el Mapa de Karnaugh utilizaremos una herramienta online, puede utilizar el siguiente link: [32x8.com](http://32x8.com)

En esta página primero debemos seleccionar la opción dependiendo de la cantidad de variables, en nuestro caso: 4.



2\_variables 3\_variables **4\_variables** 5\_variables 6\_variables

## Logic circuit simplification (SOP and POS)

This is an online Karnaugh map generator that makes a kmap, shows you how to group the terms, shows the simplified Boolean equation, and draws the circuit for up to 6 variables. It also handles **Don't cares**.

The Quine-McCluskey solver can be used for up to 6 variables if you prefer that.

Select the number of variables, then choose SOP (Sum of Products) or POS (Product of Sums) or Quine-McCluskey, and try some calculations. SOP is the default.

This logic simplification application is not intended for design purposes. It is just for fun. It is just a programming project for me. I enjoyed writing the software and hopefully you will enjoy using it. I have tried to be sure that it is accurate but I can not guarantee it. So, if you plan to use the output for any purpose, you must confirm it yourself.

Luego seleccionamos los valores de la tabla de verdad:

**Truth Table**

Submit					Y		
	A	B	C	D	0	1	x
0	0	0	0	0	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
1	0	0	0	1	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
2	0	0	1	0	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
3	0	0	1	1	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
4	0	1	0	0	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
5	0	1	0	1	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
6	0	1	1	0	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
7	0	1	1	1	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
8	1	0	0	0	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
9	1	0	0	1	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
10	1	0	1	0	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
11	1	0	1	1	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
12	1	1	0	0	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>
13	1	1	0	1	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
14	1	1	1	0	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
15	1	1	1	1	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>

Submit

Reset

☒ Highlight groups

☒ SOP  
☐ POS  
☐ Quine-McCluskey Method (SOP)

☒ CSS circuit drawing  
☐ ASCII circuit drawing

Try Tab and arrow keys for keyboard input.

Puedes elegir las siguientes opciones, ambas son igual de correctas:

- SOP (Suma de Productos)
- POS (Producto de Sumas)
- Quine-McCluskey Method (SOP)

Posterior a eso, presionamos *Submit*, y obtenemos el mapa, la función booleana reducida y el circuito correspondiente.

**Map**

	$\bar{C}\bar{D}$	$\bar{C}D$	$CD$	$C\bar{D}$
$\bar{A}\bar{B}$	0	0	1	1
$\bar{A}B$	0	1	1	0
$AB$	0	1	x	x
$A\bar{B}$	0	0	1	0

**Map Layout**

	$\bar{C}\bar{D}$	$\bar{C}D$	$CD$	$C\bar{D}$
$\bar{A}\bar{B}$	0	1	3	2
$\bar{A}B$	4	5	7	6
$AB$	12	13	15	14
$A\bar{B}$	8	9	11	10

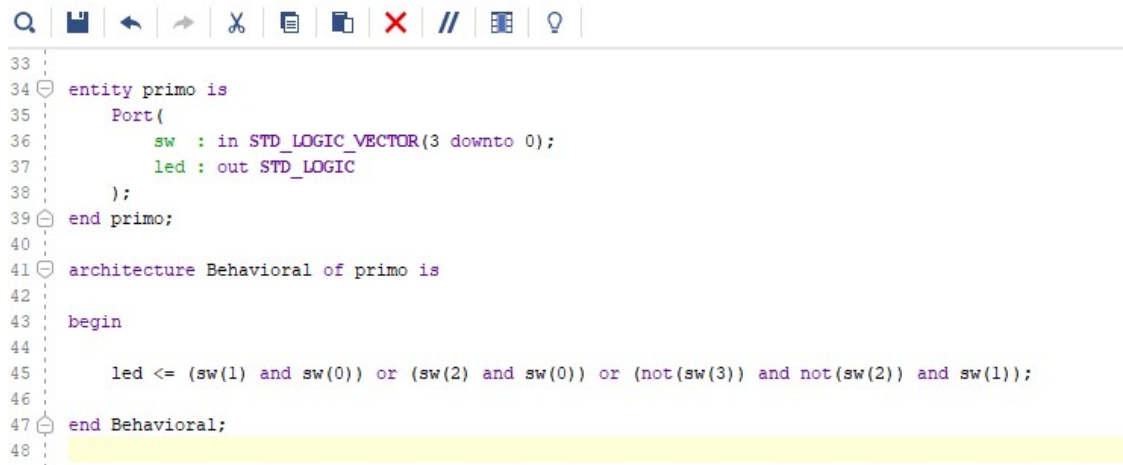
**Groups**

(3,7,11,15)	$CD$
(5,7,13,15)	$BD$
(2,3)	$\bar{A}\bar{B}C$

$y = CD + BD + \bar{A}\bar{B}C$

#### 4. Escribir la función en Vivado.

Finalmente, pasamos la función obtenida al proyecto de Vivado.



```
33
34 entity primo is
35     Port(
36         sw : in STD_LOGIC_VECTOR(3 downto 0);
37         led : out STD_LOGIC
38     );
39 end primo;
40
41 architecture Behavioral of primo is
42
43     begin
44
45         led <= (sw(1) and sw(0)) or (sw(2) and sw(0)) or (not(sw(3)) and not(sw(2)) and sw(1));
46
47     end Behavioral;
48
```

## Bibliografía:

Para más información revisar los siguientes enlaces

- <http://www.ee.surrey.ac.uk/Projects/Labview/minimisation/karnaugh.html>
- <https://web.archive.org/web/20090107120802/http://www.embedded.com/columns/programmerstoolbox/29111968>