# *Diseño Digital Moderno*

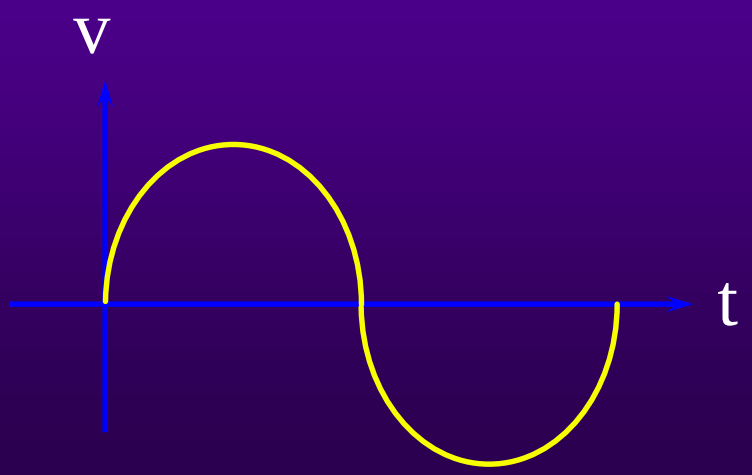
M.I. Norma Elva Chávez Rodríguez

# *Introducción a los Sistemas Digitales*

Señales { Analógicas
Digitales

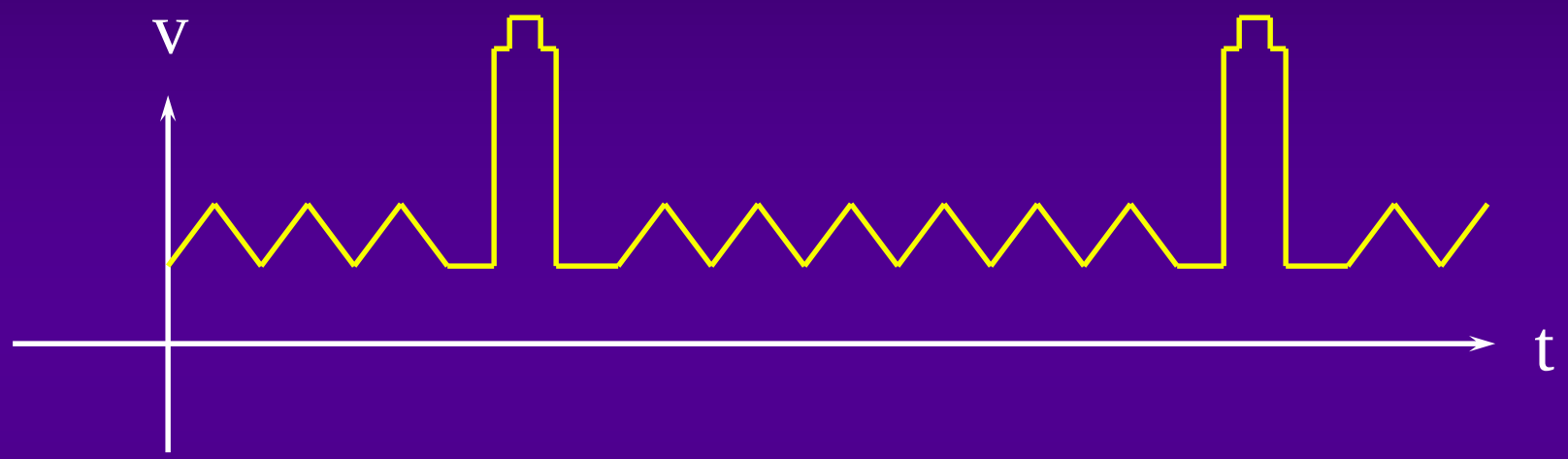
Una ***señal analógica*** es la representación de alguna cantidad que puede variar continuamente en el tiempo. Por ejemplo:

1) Onda senoidal



# *Introducción a los Sistemas Digitales*

2) Señal de televisión

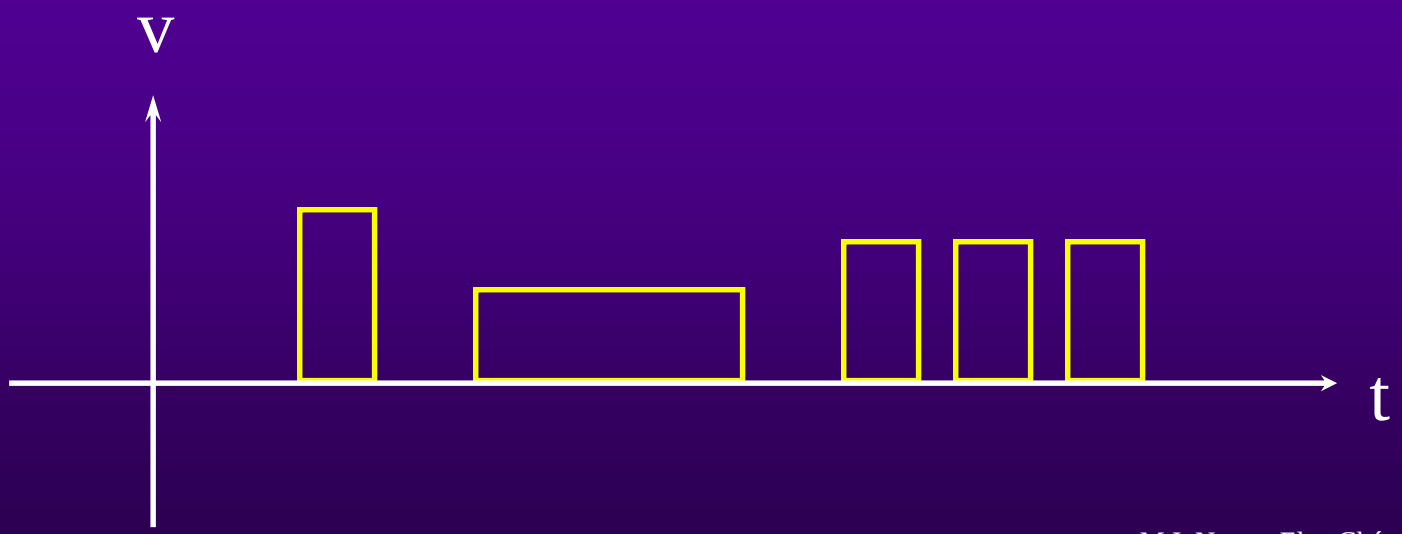



3) Señal de audio
4) Señal de temperatura
5) Velocímetro analógico

Así que, al haber señales analógicas, es equivalente a hablar de señales continuas en el tiempo.

# *Introducción a los Sistemas Digitales*

Una ***señal digital*** es la representación de alguna cantidad que varía en forma discreta (muestras de una señal continua). Por ejemplo:



# *Introducción a los Sistemas Digitales*

Algunos dispositivos digitales son:

1. Reloj digital
2. Display digital
3. Calculadoras
4. Computadoras

Analógico


Analógico

Electrónica analógica

Electrónica digital

# *Sistemas numéricos y conversiones*

En forma general:

$$S = a_n r^n + a_{n-1} r^{n-1} + \ldots + a_0 r^0 + a_{-1} r^{-1} + \ldots + a_{-m} r^{-m}$$

donde:

S = cantidad

a = dígito

m, n = posición

r = base

# *Sistemas numéricos y conversiones*

Sistema binario: (0, 1)

$(110110)_2$ ⇨ $1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$

$= 32 + 16 + 0 + 4 + 2$

$= (54)_{10}$

$(0.1101)_2$ ⇨ $1 \times 2^{-1} + 1 \times 2^{-2} + 0 \times 2^{-3} + 1 \times 2^{-4}$

$= 0.5 + 0.25 + 0 + 0.0625$

$= (0.8125)_{10}$

# *Sistemas numéricos y conversiones*

Sistema octal: (0, 1, 2, 3, 4, 5, 6, 7)

$(756)_8$ ⇨ $7 \times 8^2 + 5 \times 8^1 + 6 \times 8^0$

$= 448 + 40 + 6$

$= (494)_{10}$

Sistema hexadecimal: (0, 1, 2, 3, … , 8, 9, A, B, C, D, E, F)

$(C54B.FE)_H$ ⇨ $12 \times 16^3 + 5 \times 16^2 + 4 \times 16^1 + 11 \times 16^0 + 15 \times 16^{-1} + 14 \times 16^{-2}$

$= 49152 + 1280 + 64 + 11 + 0.9375 + 0.0547$

$= (50507.992)_{10}$

# *Sistemas numéricos y conversiones*

En general, para cualquier base tenemos:

| | |
|---|---|
| 2 | 0, 1 |
| 3 | 0, 1, 2 |
| 4 | 0, 1, 2, 3 |
| 5 | 0, 1, 2, 3, 4 |
| 6 | 0, 1, 2, 3, 4, 5 |
| 7 | 0, 1, 2, 3, 4, 5, 6 |
| 8 | 0, 1, 2, 3, 4, 5, 6, 7 |
| 9 | 0, 1, 2, 3, 4, 5, 6, 7, 8 |

# *Sistemas numéricos y conversiones*

Continuación:

| | |
|---|---|
| 10 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, |
| 11 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A |
| 12 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B |
| 13 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C |
| 14 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D |
| 15 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E |
| 16 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F |

# *Sistemas numéricos y conversiones*

En forma general:

$$S = a_n r^n + a_{n-1} r^{n-1} + \ldots + a_0 r^0 + a_{-1} r^{-1} + \ldots + a_{-m} r^{-m}$$

donde:

S = cantidad

a = dígito

m, n = posición

r = base

# *Sistemas numéricos y conversiones*

Sistema binario: (0, 1)

$(110110)_2$ ⇨ $1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$

$= 32 + 16 + 0 + 4 + 2$

$= (54)_{10}$

$(0.1101)_2$ ⇨ $1 \times 2^{-1} + 1 \times 2^{-2} + 0 \times 2^{-3} + 1 \times 2^{-4}$

$= 0.5 + 0.25 + 0 + 0.0625$

$= (0.8125)_{10}$

# *Sistemas numéricos y conversiones*

Sistema octal: (0, 1, 2, 3, 4, 5, 6, 7)

$(756)_8$ ⇨ $7 \times 8^2 + 5 \times 8^1 + 6 \times 8^0$

$= 448 + 40 + 6$

$= (494)_{10}$

Sistema hexadecimal: (0, 1, 2, 3, … , 8, 9, A, B, C, D, E, F)

$(C54B.FE)_H$ ⇨ $12 \times 16^3 + 5 \times 16^2 + 4 \times 16^1 + 11 \times 16^0 + 15 \times 16^{-1} + 14 \times 16^{-2}$

$= 49152 + 1280 + 64 + 11 + 0.9375 + 0.0547$

$= (50507.992)_{10}$

# *Sistemas numéricos y conversiones*

En general, para cualquier base tenemos:

| | |
|---|---|
| 2 | 0, 1 |
| 3 | 0, 1, 2 |
| 4 | 0, 1, 2, 3 |
| 5 | 0, 1, 2, 3, 4 |
| 6 | 0, 1, 2, 3, 4, 5 |
| 7 | 0, 1, 2, 3, 4, 5, 6 |
| 8 | 0, 1, 2, 3, 4, 5, 6, 7 |
| 9 | 0, 1, 2, 3, 4, 5, 6, 7, 8 |

# *Sistemas numéricos y conversiones*

Continuación:

| | |
|---|---|
| 10 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, |
| 11 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A |
| 12 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B |
| 13 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C |
| 14 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D |
| 15 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E |
| 16 | 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F |

# *Sistemas numéricos y conversiones*

1. Convierta $(15A75.AF)_{16}$ a base 10

$(15A75.AF)_{16}$ ⇨ $1 \times 16^4 + 5 \times 16^3 + 10 \times 16^2 + 7 \times 16^1 + 5 \times 16^0 + 10 \times 16^{-1} + 15 \times 16^{-2}$

$= 65536 + 20480 + 2560 + 112 + 5 + 0.625 + 0.0586$

$= (88693.683)_{10}$

# *Sistemas numéricos y conversiones*

2. Convierta $(11011001.101)_2$ a base 10

$(11011001.101)_2$ ⇨ $1 \times 2^7 + 1 \times 2^6 + 0 \times 2^5 + 1 \times 2^4$
$+ 1 \times 2^3 + 0 \times 2^2 + 0 \times 16^1 + 1 \times 16^0$
$+ 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$
$= 128 + 64 + 16 + 8 + 1 + 0.5 + 0.625$
$= (217.625)_{10}$

# *Sistemas numéricos y conversiones*

3. Convierta $(A3DE.F)_{16}$ a base 10

$(A3DE.F)_{16}$ ⇨

$10 \times 16^3 + 3 \times 16^2 + 13 \times 16^1 + 14 \times 16^0$
$+ 1 \times 2^3 + 0 \times 2^2 + 0 \times 16^1 + 1 \times 16^0$
$+ 15 \times 16^{-1}$
$= 40960 + 768 + 208 + 14 + 0.9375$
$= (41950.937)_{10}$

# *Sistemas numéricos y conversiones*

4. Convierta $(37AB.B)_{12}$ a base 10

$(37AB.B)_{12}$ ⇨ $3 \times 12^3 + 7 \times 12^2 + 10 \times 12^1 + 11 \times 12^0 + 11 \times 12^{-1}$

$= 5184 + 1008 + 120 + 11 + 0.9167$

$= (6323.9167)_{10}$

# *Tarea #1: Sistemas numéricos y conversiones*

Obtenga la representación en decimal
de los siguientes números

1. $(417.3)_8$
2. $(110111.111)_2$
3. $(23FA.CD)_{16}$
4. $(1485.156)_9$
5. $(AB167.B9)_{12}$
6. $(13467.A)_{13}$
7. $(1011000111.10101)_2$
8. $(2312.33)_4$
9. $(2112.122)_3$
10. $(4134.43)_5$
11. $(541.553)_6$
12. $(1654.36)_7$
13. $(A179.AA)_{11}$
14. $(DC9A.DC)_{14}$
15. $(EE459.E9)_{15}$
16. $(2567.856)_{16}$
17. $(4732.71)_8$
18. $(111101101.10111)_2$
19. $(13AFF.DEF)_{16}$
20. $(32112.312)_4$

M.I. Norma Elva Chávez Rodríguez

# *Conversión de base decimal a base r*

Si deseamos convertir un número ***de base decimal a cualquier otra base***, sólo dividimos el número decimal entre la base a la que lo queremos convertir y se van acomodando lo residuos, obteniendo la cantidad convertida.

# *Conversión de base decimal a base r*

1. Convierta $(48.123)_{10}$ a base 2 y a base 8

| | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 1 | 2 | 3 | 2 | 6 | 2 | 12 | 2 | 24 | 2 | 48 | .123 | 2 | .246 | 2 | .492 | 2 | .984 | 2 | .968 | 2 | | |
| | 1 | | 1 | | 0 | | 0 | | 0 | | 0 | . 0 | | 0 | | 0 | | 1 | | | | | |

Por lo tanto

$(48.123)_{10}$ ⇒ $(110000.0001)_2$

| | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8 | 6 | 8 | 48 | .123 | 8 | .984 | 8 | .872 | 8 | .976 | 8 | | | | | | | | | | | | |
| | 0 | | 0 | . 0 | | 7 | | 6 | | | | | | | | | | | | | | | |

Por lo tanto

$(48.123)_{10}$ ⇒ $(60.076)_8$

M.I. Norma Elva Chávez Rodríguez

# *Conversión de base decimal a base r*

## 2. Convierta $(2950)_{10}$ a base 16

$16\overline{)2950}$ $16\overline{)184}$ $16\overline{)11}$

6 8

Por lo tanto

$(2950)_{10}$ ⇒ $(B86)_{16}$

## 3. Convierta $(710)_{10}$ a base 2

$2\overline{)710}$ $2\overline{)355}$ $2\overline{)177}$ $2\overline{)88}$ $2\overline{)44}$ $2\overline{)22}$ $2\overline{)11}$ $2\overline{)5}$ $2\overline{)2}$ $2\overline{)1}$

0 1 1 0 0 0 1 1 0

Por lo tanto

$(710)_{10}$ ⇒ $(1011000110)_2$

# *Conversión de base decimal a base r*

Para convertir un número fraccionario de base decimal a otra base se hace mediante multiplicaciones sucesivas. Los siguientes ejemplos ilustran el método.

1. Convierta $(0.546)_{10}$ a base 2

| .546 \| 2 | .092 \| 2 | .184 \| 2 | .368 \| 2 | .736 \| 2 | .472 \| 2 | . . . |
|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | | . . . |

Por lo tanto

$(0.546)_{10}$ ⇒ $(0.10001)_2$ aproximadamente

# *Conversión de base decimal a base r*

2. Convierta $(0.546)_{10}$ a base 16

| .546 \| 16 | .736 \| 16 | .776 \| 16 | .416 \| 16 | .656 \| 16 | . . . |
|---|---|---|---|---|---|
| 8 | B | C | 6 | | . . . |

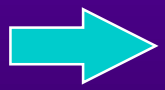
Por lo tanto

$(0.546)_{10}$ ⇒ $(0.8BC6)_{16}$ aproximadamente

# *Conversión de base r a base decimal*

Para convertir un número real ***de base decimal a otra base*** se realiza primero la parte entera y después la parte fraccionaria para, finalmente, sumar ambos resultados.

Realice las siguientes conversiones de acuerdo con el ejemplo.

$1.(4315.718)_{10}$ 

$2 = (100001101011.1011)_2$

$5 = (11423.324)_5$

$13 = (1C6C.944)_{13}$

$16 = (10DB.B7CE)_{16}$

M.I. Norma Elva Chávez Rodríguez

# *Conversión de base r a base decimal*

2. $(8349.159)_{10}$ → 2 =
   4 =
   8 =
   16 =

3. $(935.75)_{10}$ → 2 =
   4 =
   8 =
   16 =

La conversión entre bases se realiza pasando primero por base decimal.

M.I. Norma Elva Chávez Rodríguez

# *Tarea #2: Conversiones entre bases*

Desarrolla un programa en lenguaje C, Pascal, Fortran o Basic para la conversión de números de una base a otra. Estructura el programa de tal forma que maneje su información por medio de ventanas y menús.

# *Operaciones aritméticas*

Complementos { A la base
A la base disminuída

Complemento a la base. ***Definición:***

$\Lambda^* = 10^n - \Lambda$ para $\Lambda \neq 0$

$\Lambda^* = 0$ para $\Lambda = 0$

donde:

$\Lambda^*$ = cantidad en complementos a la base

n = número de dígitos enteros de $\Lambda$

$\Lambda$ = cantidad

M.I. Norma Elva Chávez Rodríguez

# *Operaciones aritméticas*

Ejemplos: Obtenga el complemento a la base de los siguientes números

1. $(52520)_{10}$
2. $(0.3267)_{10}$
3. $(101100)_{2}$
4. $(0.10110)_{2}$
5. $(AB2373)_{16}$
6. $(347823)_{11}$

# *Operaciones aritméticas*

$$\Lambda^* = 10^n - \Lambda$$

1. $\Lambda^* = 10^5 - 52520_{10}$
   $= 100000_{10} - 52520_{10}$
   $= 47480_{10}$

2. $\Lambda^* = 10^0 - 0.3267_{10}$
   $= 1_{10} - 0.3267_{10}$
   $= 0.6733_{10}$

# *Operaciones aritméticas*

## $\Lambda^* = 10^n - \Lambda$

3. $\Lambda^* = 10^6 - 101100_2$

$$\begin{array}{r} 1000000_{\,2} \\ -\ 101100_{\,2} \\ \hline 010100_{\,2} \end{array}$$

$\Lambda^* = 010100_2$

4. $\Lambda^* = 10^0 - 0.10110_2$

$$\begin{array}{r} 1.00000_{\,2} \\ -\ 0.10110_{\,2} \\ \hline 0.01010_{\,2} \end{array}$$

$\Lambda^* = 0.01010_2$

# *Operaciones aritméticas*

$$\Lambda^* = 10^n - \Lambda$$

5. $\Lambda^* = 10^6 - AB2373_{16}$

$$\begin{array}{r} 1000000_{16} \\ -\ AB2373_{16} \\ \hline 054DC8D_{16} \end{array}$$

$\Lambda^* = 54DC8D_{16}$

6. $\Lambda^* = 10^6 - 347823_{11}$

$$\begin{array}{r} 1000000_{11} \\ -\ 347823_{11} \\ \hline 763288_{11} \end{array}$$

$\Lambda^* = 763288_{11}$

# *Operaciones aritméticas*

Complemento a la base disminuída. ***Definición:***

$$\bar{\Lambda} = 10^n - 1 - \Lambda$$

Ejemplos:

1. $(52520)_{10}$

$$\bar{\Lambda} = 10^5 - 1 - 52520_{10}$$
$$= 99999_{10} - 52520_{10}$$
$$\bar{\Lambda} = 47479_{10}$$

2. $(0.0110)_{10}$

$$\bar{\Lambda} = 10^0 - 1 - 0.0110_2$$
$$0.1111_{\,2}$$
$$-\ 0.0110_{\,2}$$
$$0.1001_{\,2}$$
$$\bar{\Lambda} = 0.1001_2$$

# *Operaciones aritméticas*

3. $(347823)_{11}$

$$\bar{\Lambda} = 10^6 - 1 - 347823_{11}$$
$$= AAAAAA_{11} - 347823_{11}$$
$$\bar{\Lambda} = 763287_{11}$$

4. $(1011011)_2$

5. $(AFC192)_{16}$

6. $(1101101)_2$

# *Representación de datos*

Magnitud y signo

Formato

Signo { 0 positivo
1 negativo

# *Representación de datos*

M.I. Norma Elva Chávez Rodríguez

Si n=3

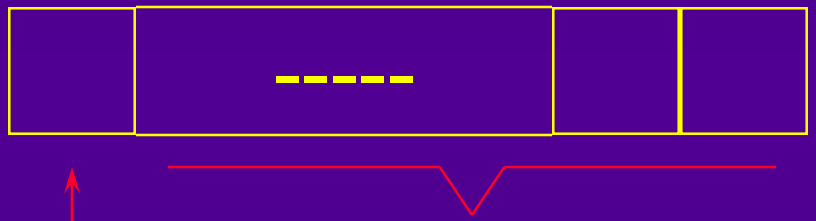
| | | | | | |
|---|---|---|---|---|---|
| 0000 | +0 | 0110 | +6 | 1101 | -5 |
| 0001 | +1 | 0111 | +7 | 1110 | -6 |
| 0010 | +2 | 1001 | -1 | 1111 | -7 |
| 0011 | +3 | 1010 | -2 | | |
| 0100 | +4 | 1011 | -3 | | |
| 0101 | +5 | 1100 | -4 | | |

Cantidad { mayor: $2^n - 1$
menor: $-(2^n - 1)$

# *Representación de datos*

Complementos a 2

Formato N . . . . . 1 0




Signo { 0 positivo
1 negativo

# *Representación de datos*

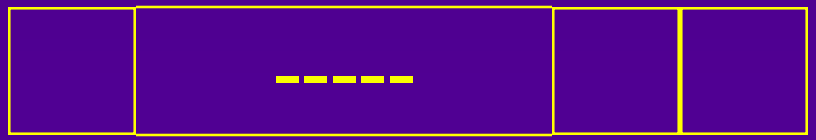
| Si n=3 | | Complemento a 2 | |
|---|---|---|---|
| 0000 | +0 | 1111 | -1 |
| 0001 | +1 | 1110 | -2 |
| 0010 | +2 | 1101 | -3 |
| 0011 | +3 | 1100 | -4 |
| 0100 | +4 | 1011 | -5 |
| 0101 | +5 | 1010 | -6 |
| 0110 | +6 | 1001 | -7 |
| 0111 | +7 | 1000 | -8 |

Cantidad { mayor: $2^n - 1$
menor: $- 2^n$

# *Representación de datos*

Complementos a 1

Formato N . . . . . 1 0




magnitud

signo

Signo { 0 positivo
1 negativo

# *Representación de datos*

| Si n=3 | | Complemento a 1 | |
|---|---|---|---|
| 0000 | +0 | 1111 | -0 |
| 0001 | +1 | 1110 | -1 |
| 0010 | +2 | 1101 | -2 |
| 0011 | +3 | 1100 | -3 |
| 0100 | +4 | 1011 | -4 |
| 0101 | +5 | 1010 | -5 |
| 0110 | +6 | 1001 | -6 |
| 0111 | +7 | 1000 | -7 |

Cantidad { mayor: $2^n - 1$
menor: $-(2^n - 1)$

# *Tarea #4: Operaciones aritméticas*

Investigar la utilización de los procedimientos para sumar dos números en complemento a uno y en complemento a dos.

# *Operaciones aritméticas*

M.I. Norma Elva Chávez Rodríguez

Las dos operaciones básicas son:

- ***la suma***
- ***la resta***

El procedimiento para realizar sumas en bases diferentes a la decimal es muy similar al usado para hacer sumas y restas en este sistema. Por ejemplo:

$$\begin{array}{r} 8_{10} \\ +\ 1_{10} \\ \hline 9_{10} \end{array} \qquad \begin{array}{r} 2_{4} \\ +\ 1_{4} \\ \hline 3_{4} \end{array} \qquad \begin{array}{r} 5_{8} \\ +\ 2_{8} \\ \hline 7_{8} \end{array} \qquad \begin{array}{r} 1_{2} \\ +\ 1_{2} \\ \hline 1\,0_{2} \end{array} \qquad \begin{array}{r} 6_{16} \\ +\ 9_{16} \\ \hline F_{16} \end{array}$$

carry generado

# *Operaciones aritméticas*

M.I. Norma Elva Chávez Rodríguez

$$\begin{array}{r} 3_7 \\ +\ 4_7 \\ \hline 1\ 0_7 \end{array}$$

carry generado

$$\begin{array}{r} 2_{11} \\ +\ 9_{11} \\ \hline 1\ 0_{11} \end{array}$$

carry generado

$$\begin{array}{r} 4_6 \\ +\ 5_6 \\ \hline 1\ 3_6 \end{array}$$

carry generado

$$\begin{array}{r} F_{16} \\ +\ F_{16} \\ \hline 1\ E_{16} \end{array}$$

carry generado

$$\begin{array}{r} 111111 \\ +\ 1011011_2 \\ \ \ 0101111_2 \\ \hline 10001010_2 \end{array} \Leftarrow$$

carry generado

carry generado
fuera de las posiciones

# *Operaciones aritméticas*

Ejemplos:

1.

$$\begin{array}{r} 1111 \\ +\ 1A69F2_{16} \\ 21A93F_{16} \\ \hline 3C1331_{16} \end{array}$$

⇦ carry generado

2.

$$\begin{array}{r} 111 \\ +\ 2546_{7} \\ 3461_{7} \\ \hline 6340_{7} \end{array}$$

⇦ carry generado

# *Operaciones aritméticas*

El procedimiento para llevar a cabo restas se ilustra a continuación:

1.

```
    1 2
  0 2 0
  1 1 0 1 0 1 1 . 1 1 2
- 1 0 0 1 1 0 1 . 0 1 2
-------------------------
  0 0 1 1 1 1 0 . 1 0 2
```

2.

```
   13  9 15 10  9    12 11
 7  4  0  6  1  0    3  2 10
 8 5 1 7 2 1 . 4 3 1 9
-7 8 4 8 3 2 . 5 6 7 9
---------------------
 0 5 5 7 7 7 . 7 5 6 9
```

# *Operaciones aritméticas*

3.

```
                    28
      E 23 4 17     9  C 18
  E F 7 5 1 A . A D 2 F 16
- D 9 F 3 B 4 . 2 E 7 1 16
-----------------------
  1 5 8 1 6 6 . 7 E B E 16
```

4.

```
  A 4 5 C 2 5 . 0 F 2 16
- F 1 B F 4 1 . 1 C D 16
-----------------------
```

# *Códigos*

Un código es un conjunto de símbolos que representan número, letra o palabras.

Códigos {
- BCD
- Exceso 3
- GRAY
- ASCII

# *Códigos*

## Código BCD ( Binary - Coded Decimal )

| Decimal | BCD |
|---|---|
| 0 | 0000 |
| 1 | 0001 |
| 2 | 0010 |
| 3 | 0011 |
| 4 | 0100 |
| 5 | 0101 |
| 6 | 0110 |
| 7 | 0111 |
| 8 | 1000 |
| 9 | 1001 |

# *Códigos*

Ejemplo:

Convierta $(1492.15)_{10}$ a BCD

0001 0100 1001 0010 . 0001 0101 en BCD

Ejemplo:

Convierta $(95.7)_{10}$ a BCD

1001 0101 . 0111 en BCD

# *Códigos*

## Código Exceso 3

| Decimal | BCD |
|---|---|
| 0 | 0011 |
| 1 | 0100 |
| 2 | 0101 |
| 3 | 0110 |
| 4 | 0111 |
| 5 | 1000 |
| 6 | 1001 |
| 7 | 1010 |
| 8 | 1011 |
| 9 | 1100 |

# *Códigos*

Ejemplo:

Convierta $(43.2)_{10}$ a Exceso 3

1001 0010 . 0001 en Exceso 3

## Código Gray

Es un código de cambio mínimo, en el cuál sólo un bit del código cambia cuando se pasa de una etapa a la siguiente. El código Gray es un código sin valor.

# *Códigos*

## Código Gray

| Decimal | Gray | Decmal | Gray |
|---|---|---|---|
| 0 | 0000 | 8 | 1100 |
| 1 | 0001 | 9 | 1101 |
| 2 | 0011 | 10 | 1111 |
| 3 | 0010 | 11 | 1110 |
| 4 | 0110 | 12 | 1010 |
| 5 | 0111 | 13 | 1011 |
| 6 | 0101 | 14 | 1001 |
| 7 | 0100 | 15 | 1000 |

# *Funciones booleanas*

Las ***funciones booleanas*** están constituidas de variables booleanas que pueden tomar los valores de cero lógico ó uno lógico.

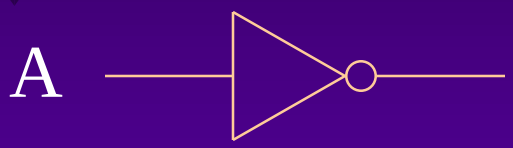
Operadores booleanos básicos:

| | | |
|---|---|---|
| 1. NOT | $\overline{\ }$ | F(A) = ***NOT*** A = $\overline{A}$ |
| 2. AND | • | F(A,B) = A ***AND*** B = A • B |
| 3. OR | + | F(A,B) = A ***OR*** B = A + B |

# *Funciones booleanas*

M.I. Norma Elva Chávez Rodríguez

## NOT $\overline{\ }$

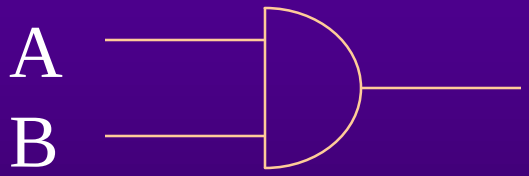



Símbolo

Tabla de verdad

| A | F = $\overline{A}$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

## AND •



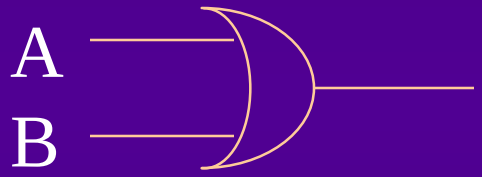

Símbolo

Tabla de verdad

| A | B | F = A • B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

# *Funciones booleanas*

OR +




Símbolo

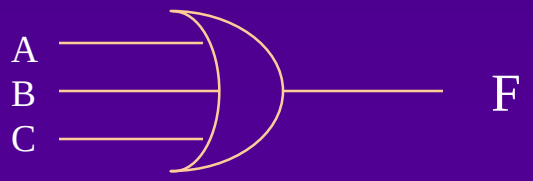
F = A + B

Tabla de verdad

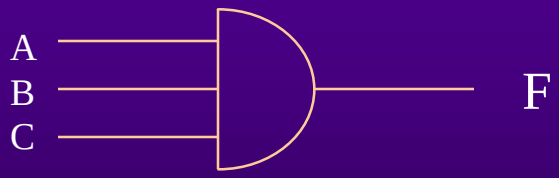
| A | B | F = A + B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

# *Funciones booleanas*

## Compuertas AND y OR de tres variables




$F = A \cdot B \cdot C$

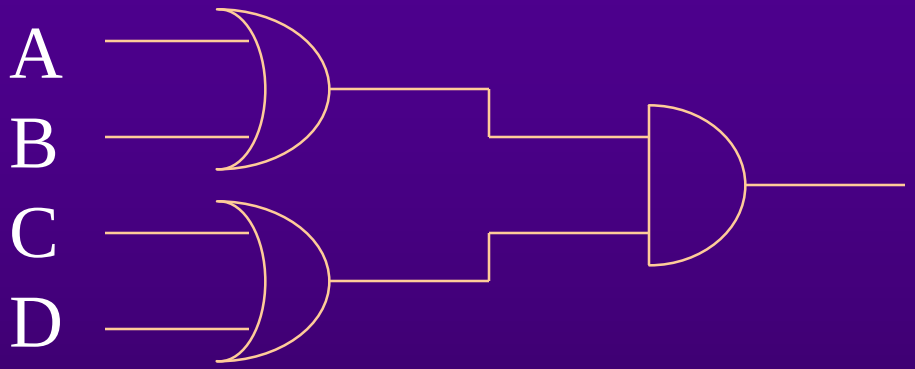



$F = A + A + C$

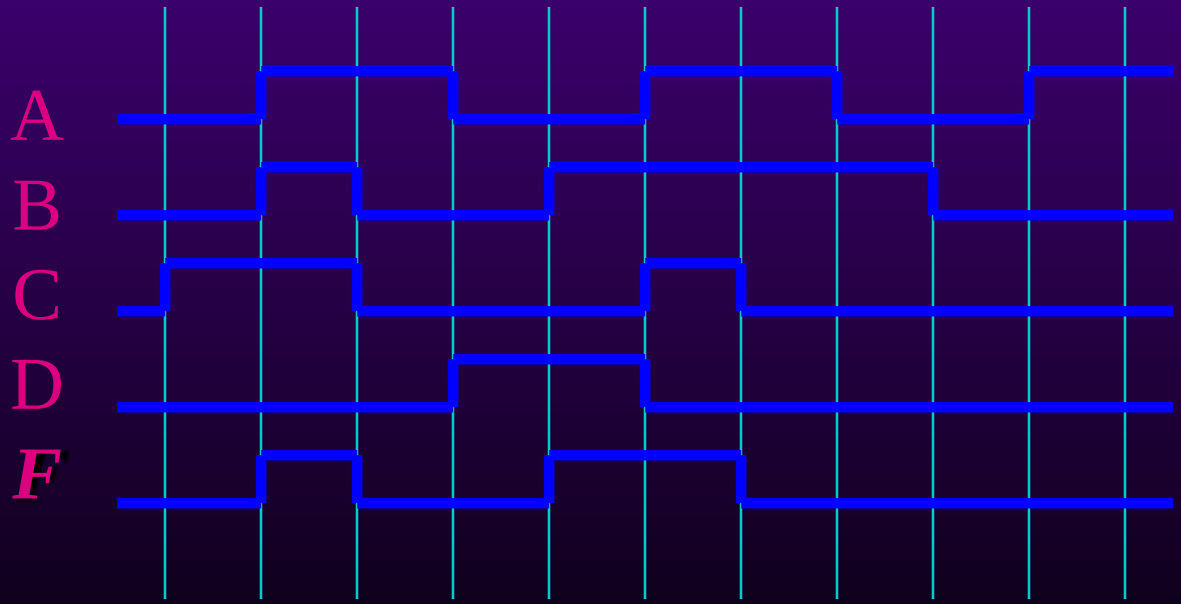
Tabla de verdad

| A | B | C | $F = A \cdot B \cdot C$ | $F = A + A + C$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

# *Funciones booleanas*




$$F = (A+B)(C+D)$$



# *Funciones booleanas*

## ***Jerarquía de los operadores***

1. NOT
2. AND
3. OR

Los paréntesis se resuelven de adentro hacia afuera.

Ejemplos:

Muestre el circuito lógico de la siguiente función y tabla de verdad.

1. $F(A,B,C,D) = [(ABC + \overline{B}C)(AB + \overline{C}D)]$

$[(ABCD + AB)(AC + BD)]$

# *Funciones booleanas*

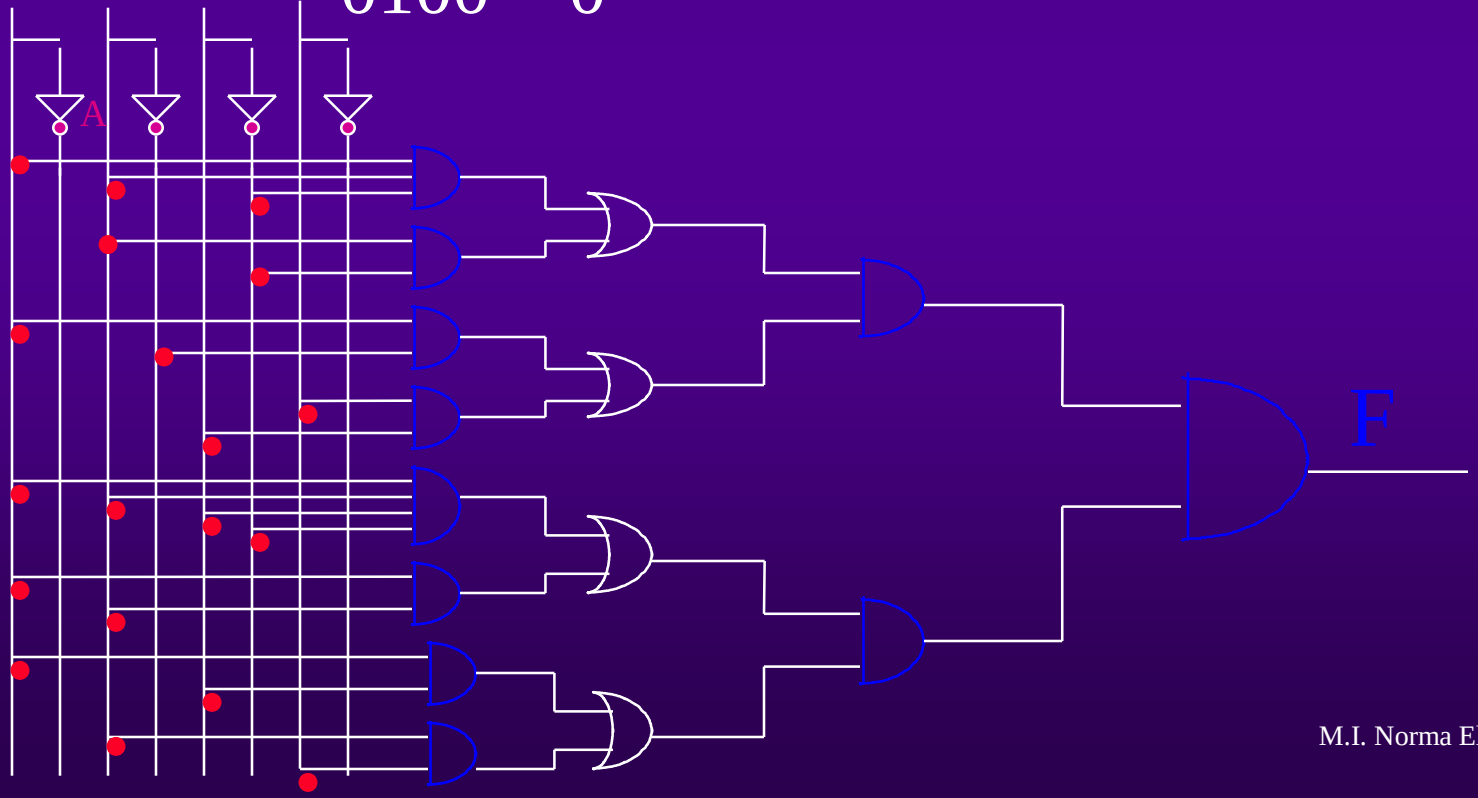
Si A=1 B=0 C=0 D =1

Determine su valor lógico.

F = [(100 + 00)(10 + 01)] [(1001 + 10)(10 + 01)]

= [(101 + 00)(11 + 01)] [(1001 + 10)(10 + 01)]

A B C D = 0100 = 0



# *Funciones booleanas*

2. $F(A,B,C) = AB + B\overline{C} + A\overline{B}\overline{C}$

A = 1 B = 1 C = 0

3. $F(X,Y,Z) = X + \overline{Y}Z + X\overline{Y}\overline{Z}$

X = 1 Y = 0 Z = 1

***Término:*** Un término es una o más variables unidas por el operador AND.

***Minitérmino:*** Para una función de n variables, el conjunto de las N variables puede tomar $2^n$ valores diferentes.

3 variables → $2^n = 2^3$ valores diferentes

# *Funciones booleanas*

| A | B | C | minitérminos |
|---|---|---|---|
| 0 | 0 | 0 | $\overline{A}\,\overline{B}\,\overline{C}$ |
| 0 | 0 | 1 | $\overline{A}\,\overline{B}\,C$ |
| 0 | 1 | 0 | $\overline{A}\,B\,\overline{C}$ |
| 0 | 1 | 1 | $\overline{A}\,B\,C$ |
| 1 | 0 | 0 | $A\,\overline{B}\,\overline{C}$ |
| 1 | 0 | 1 | $A\,\overline{B}\,C$ |
| 1 | 1 | 0 | $A\,B\,\overline{C}$ |
| 1 | 1 | 1 | $A\,B\,C$ |

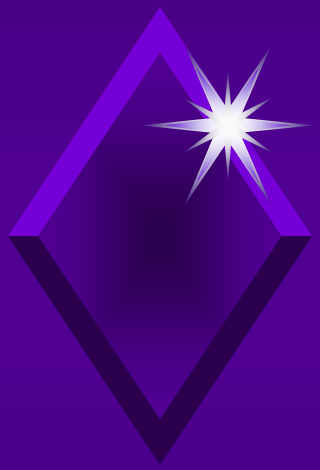
Un ***minitémino*** es un término que contiene todas las variables de la función unidas por el operador AND.

# *Funciones Booleanas.*

Maxitérmino:

Un maxitérmino es una operación OR de N términos, cada término contiene una de las variables y todas las variables deberán estar presentes en el maxitérmino

# *Funciones Booleanas.*



| A | B | C | maxitérminos |
|---|---|---|---|
| 0 | 0 | 0 | $A + B + C$ |
| 0 | 0 | 1 | $A + B + \overline{C}$ |
| 0 | 1 | 0 | $A + \overline{B} + C$ |
| 0 | 1 | 1 | $A + \overline{B} + \overline{C}$ |
| 1 | 0 | 0 | $\overline{A} + B + C$ |
| 1 | 0 | 1 | $\overline{A} + B + \overline{C}$ |
| 1 | 1 | 0 | $\overline{A} + \overline{B} + C$ |
| 1 | 1 | 1 | $\overline{A} + \overline{B} + \overline{C}$ |

# *Funciones Booleanas.*

Formas Canónicas:

Cuando una función booleana se expresa como suma de minitérminos o producto de maxitérminos, se dice que la función se encuentra en su forma canónica.

Cualquier función booleana se puede expresar como suma de productos o producto de sumas

# *Funciones Booleanas.*

EJEMPLO:

$F(A,B,C) = AB + BC$

| A | B | C | F |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

$$F(A,B,C) = \bar{A}\bar{B}C + A\bar{B}C + AB\bar{C} + ABC$$

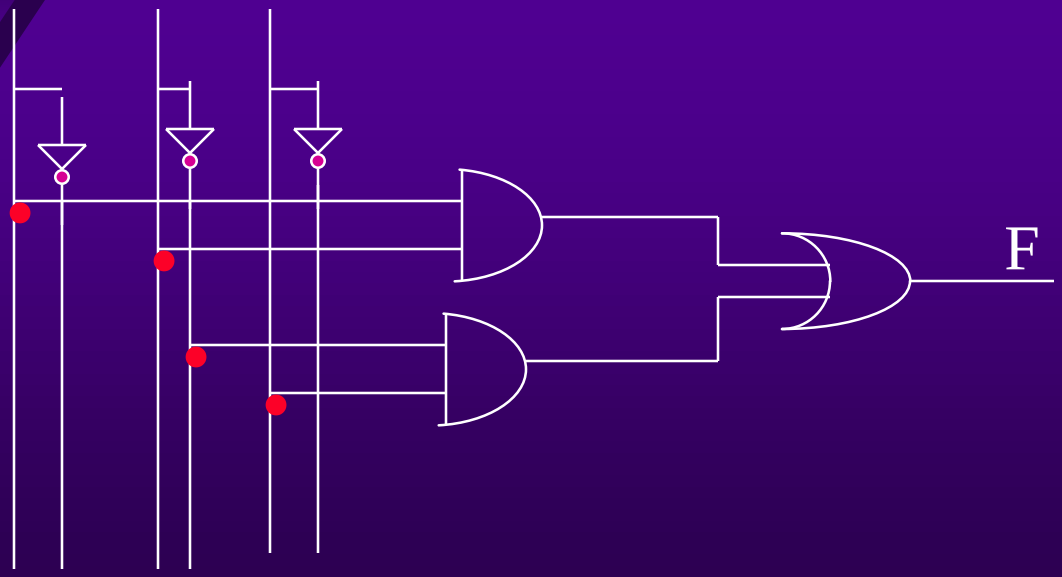
# *Funciones Booleanas.*

La función que representa los ceros es la siguiente:

$F(A,B,C)=(A+B+C)(A+\overline{B}+C)(A+\overline{B}+\overline{C})(\overline{A}+B+C)$

C. Lógico:

A B C



# *Funciones Booleanas.*

Muestra la tabla de verdad de las siguientes funciones:

$F_1 (A,B,C) = A$

$F_2 (A,B,C) = A\bar{B} + A\bar{C} + ABC$

$F_3 (A,B,C) = \bar{A}\bar{B}C + A\bar{B}C + \bar{A}BC + ABC$

# *Funciones Booleanas.*

| A | B | C | $F_1$ | $F_2$ | $F_3$ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 |

Dos o más funciones son equivalentes si y solo si tengan la misma tabla de verdad

# *Teoremas del Algebra Booleana.*

| | | |
|---|---|---|
| 1.- | $x \cdot 1 = x$ | $x+0 = x$ |
| 2.- | $x \cdot x = x$ | $x+x = x$ |
| 3.- | $x \cdot 0 = 0$ | $x+1 = 1$ |
| 4.- | $x \cdot \bar{x} = 0$ | $x+\bar{x} = 1$ |
| 5.- | $\bar{\bar{x}} = x$ | |
| 6.- | $\overline{(x \cdot y)} = \bar{x}+\bar{y}$ | $\overline{x+y} = \bar{x} \cdot \bar{y}$ |

# *Teoremas del Algebra Booleana.*

## Demostración del Teorema 6

| x y | $\overline{x\,y}$ | $\overline{x}\,\overline{y}$ | $\overline{x} + \overline{y}$ |
|---|---|---|---|
| 0 0 | 0 | 1 | 1 |
| 0 1 | 0 | 1 | 1 |
| 1 0 | 0 | 1 | 1 |
| 1 1 | 1 | 0 | 0 |

| x y | x + y | $\overline{x + y}$ | $\overline{x}\,\overline{y}$ |
|---|---|---|---|
| 0 0 | 0 | 1 | 1 |
| 0 1 | 1 | 0 | 0 |
| 1 1 | 1 | 0 | 0 |
| 1 0 | 1 | 0 | 0 |

# *Teoremas del Algebra Booleana.*

7.- $XY = YX$ $X+Y = Y+X$

8.- $XYZ = X(YZ) = (XY)Z$ $X+Y+Z = X+(Y+Z)$

9.- $X(Y+Z) = XY + YZ$ $X+(YZ) = (X+Y)(X+Z)$

10.- $X(X+Y) = X$ $X+(XY) = X$

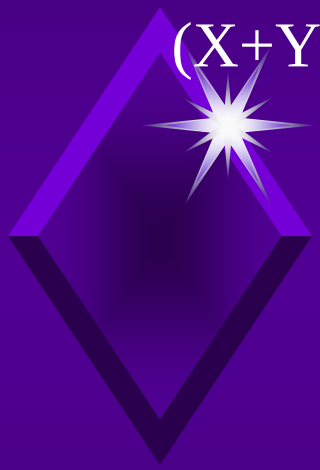
11.- $(X+Y)(X+Y') = X$ $XY+XY' = X$

12.- $X(X'+Y) = XY$ $X+X'Y = X+Y$

13.- $XY+X'Z+YZ = XY+X'Z$ $(X+Y)(x'+Z)(Y+Z) = (X+Y)(X'+Z)$

# *Teoremas del Algebra Booleana.*

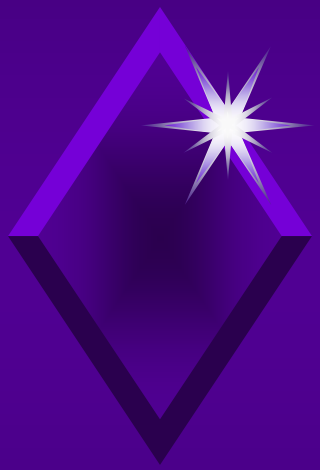
## Demostración 9b



| XYZ | YZ | X+YZ | X+Y | X+Z | (X+Y)(X+Z) |
|---|---|---|---|---|---|
| 000 | 0 | 0 | 0 | 0 | 0 |
| 001 | 0 | 0 | 0 | 1 | 0 |
| 010 | 0 | 0 | 1 | 0 | 0 |
| 011 | 1 | 1 | 1 | 1 | 1 |
| 100 | 0 | 1 | 1 | 1 | 1 |
| 101 | 0 | 1 | 1 | 1 | 1 |
| 110 | 0 | 1 | 1 | 1 | 1 |
| 111 | 1 | 1 | 1 | 1 | 1 |

# *Teoremas del Algebra Booleana.*

## Demostración 11a



| XY | X+Y | X+Y’ | (X+Y)(X+Y’) |
|---|---|---|---|
| 00 | 0 | 1 | 0 |
| 01 | 1 | 0 | 0 |
| 10 | 1 | 1 | 1 |
| 11 | 1 | 1 | 1 |

# *Simplificación de funciones.*

1.- $F(x,y,z) = x'y'z' + xyz' + xyz$

$= x'y'z' + xy(z' + z)$

$= x'y'z' + xy$ 4.B

2.- $F(x,y,z) = x + x'y + xy + xy'$

$= x + y + x(x + y')$ 12.B

$= x + y + x$ ~~2B~~ 4B

$= x + y$

# *Simplificación de funciones.*

3.- $F_1 = xy + x'z + yz$ 13.A

$= xy + x'z$

4.- $F_2 = AB'C'D+ABCD+A'B'C'+ACD+B'C'D$

$+ ACD$

$= ACD(B'+B+1) +A'BC'+B'C'D$

$= ACD+A'BC'+BC'D$ 2.B Y 3.B

5.- $F_3 =$ $AB+ABCD+ABC+ACD+B(C'+D')$

$= AB\ (1+CD+C)+ACD+B(C'+D')$

$= AB +ACD+B(C'+D')$ 13.A

XZ YX Z X'

$= CDA+B\ (C'+D')$

# *Simplificación de funciones.*

6.- F(A,B,C,D) =AB+AB’C’+BCD+AB(C’+D’)

=AB+AC’C’+BCD+ABC’D’

=AB(1+C’D’)+AB’C’+BCD

=AB+AB’C’+BCD

=A(B+B’C’)+BCD 12.B

=AB+AC’+BCD

7.- F(A,B,C,D) = (A+B)(A’+C)(B+C) 13.B

= (A+B)(A’+C)(B+C+AA’) 4.A

=(A+B)(A’+C)(A+B+C)(A’+B+C) 9.B

= (A+B)(A’+C)

# *Simplificación de funciones.*

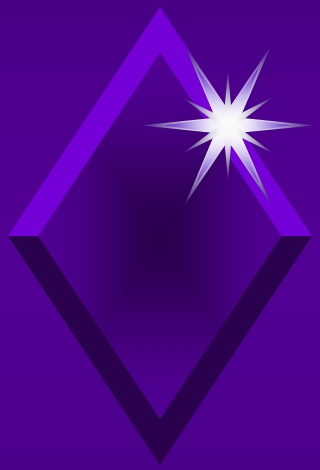
8.- $A+B'+A'B+(A+B')A'B = 1$

$A+B'+A'B(1+(A+B')) = 1$

$A+B'+A'B = 1$

$A+B+B' = 1$

$1 = 1$



9.-$(W'+X+Y'+Z')(W'+X+Y'+Z)(W'+X+Y+Z')(W'+X+Y+Z)=W'+X$

11.A

$(W'+X+Y')(W'+X+Y) = W'+X$ 11.A

$W'+X = W'+X$

# *Simplificación de funciones.*

10.- (A+B)(B+C+D')(B'+C+D') = (A+B)(C+D')

APLICANDO 11.A

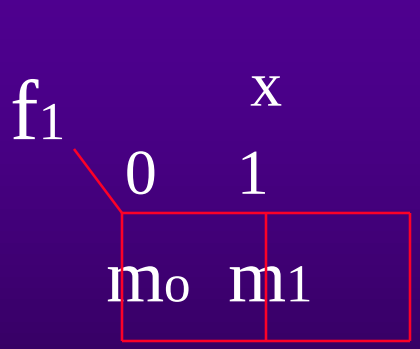
(A+B)(C+D') = (A+B)(C+D')

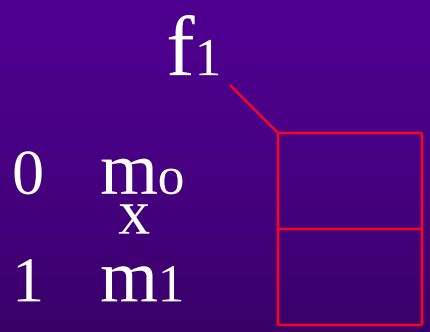
# *Mapas de Karnaugh.*

Un mapa de Karnaugh. Es otra forma de representar la tabla de verdad consistiendo de $2^N$ casillas donde cada casilla contiene un minitermino ó un máxitermino.

Para una variable

| x | $f_1$ |
|---|---|
| 0 | $m_0$ |
| 1 | $m_1$ |



# *Mapas de Karnaugh.*

Para dos variables:

| x | y | f |
|---|---|---|
| 0 | 0 | mo |
| 0 | 1 | m1 |
| 1 | 0 | m2 |
| 1 | 1 | m3 |

f

| x \ y | 0 | 1 |
|---|---|---|
| 0 | mo | m1 |
| 1 | m2 | m3 |

f

| x \ y | 0 | 1 |
|---|---|---|
| 0 | mo | m2 |
| 1 | m1 | m3 |

# *Mapas de Karnaugh.*

Para tres variables:

| x | y | z | f |
|---|---|---|---|
| 0 | 0 | 0 | mo |
| 0 | 0 | 1 | m1 |
| 0 | 1 | 0 | m2 |
| 0 | 1 | 1 | m3 |
| 1 | 0 | 0 | m4 |
| 1 | 0 | 1 | m5 |
| 1 | 1 | 0 | m6 |
| 1 | 1 | 1 | m7 |

f yz

| x | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | mo | m1 | m3 | m2 |
| 1 | m4 | m5 | m7 | m6 |

f z

| xy | 0 | 1 |
|---|---|---|
| 00 | m0 | m1 |
| 01 | m2 | m3 |
| 11 | m6 | m7 |
| 10 | m4 | m5 |

# *Mapas de Karnaugh.*

Para cuatro variables:

| w | x | y | z | f |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | mo |
| 0 | 0 | 0 | 1 | m1 |
| 0 | 0 | 1 | 0 | m2 |
| 0 | 0 | 1 | 1 | m3 |
| . | . | . | . | . |
| . | . | . | . | . |
| 1 | 1 | 1 | 1 | m15 |

| f | yz | | | |
|---|---|---|---|---|
| x | 00 | 01 | 11 | 10 |
| 00 | mo | m1 | m3 | m2 |
| 01 | m4 | m5 | m7 | m6 |
| 11 | m12 | m13 | m15 | m14 |
| 10 | m6 | m9 | m11 | m10 |

# *Mapas de Karnaugh.*

Para 5 variables:

| w | x | y | z | t | f |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | mo |
| 0 | 0 | 0 | 0 | 1 | m1 |
| 0 | 0 | 0 | 1 | 0 | m2 |
| . | . | . | . | . | . |
| 1 | 1 | 1 | 1 | 1 | m31 |

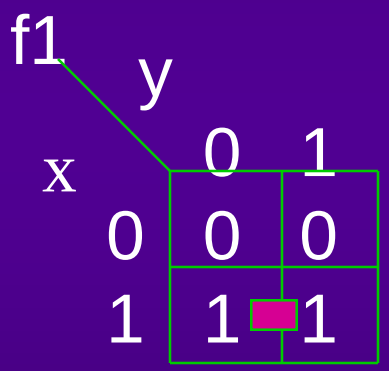
| t / yzt | 000 | 001 | 011 | 010 |
|---|---|---|---|---|
| wx 00 | mo | m1 | m3 | m2 |
| 01 | m8 | m11 | m9 | m10 |
| 11 | m24 | m25 | m27 | m26 |
| 10 | m16 | m17 | m19 | m18 |

| 110 | 11 | 101 | 100 |
|---|---|---|---|
| m6 | m7 | m5 | m4 |
| m14 | m15 | m13 | m12 |
| m30 | m31 | m29 | m28 |
| m22 | m23 | m21 | m20 |

# *Mapas de Karnaugh.*

P. ej. : Dada la siguiente tabla de verdad, representarla en un mapade Karnaugh y minimizarla.

| x | y | f1 |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |




$f1 = \overline{x}y + xy$

$= \overline{x}(y + y)$

$= x$

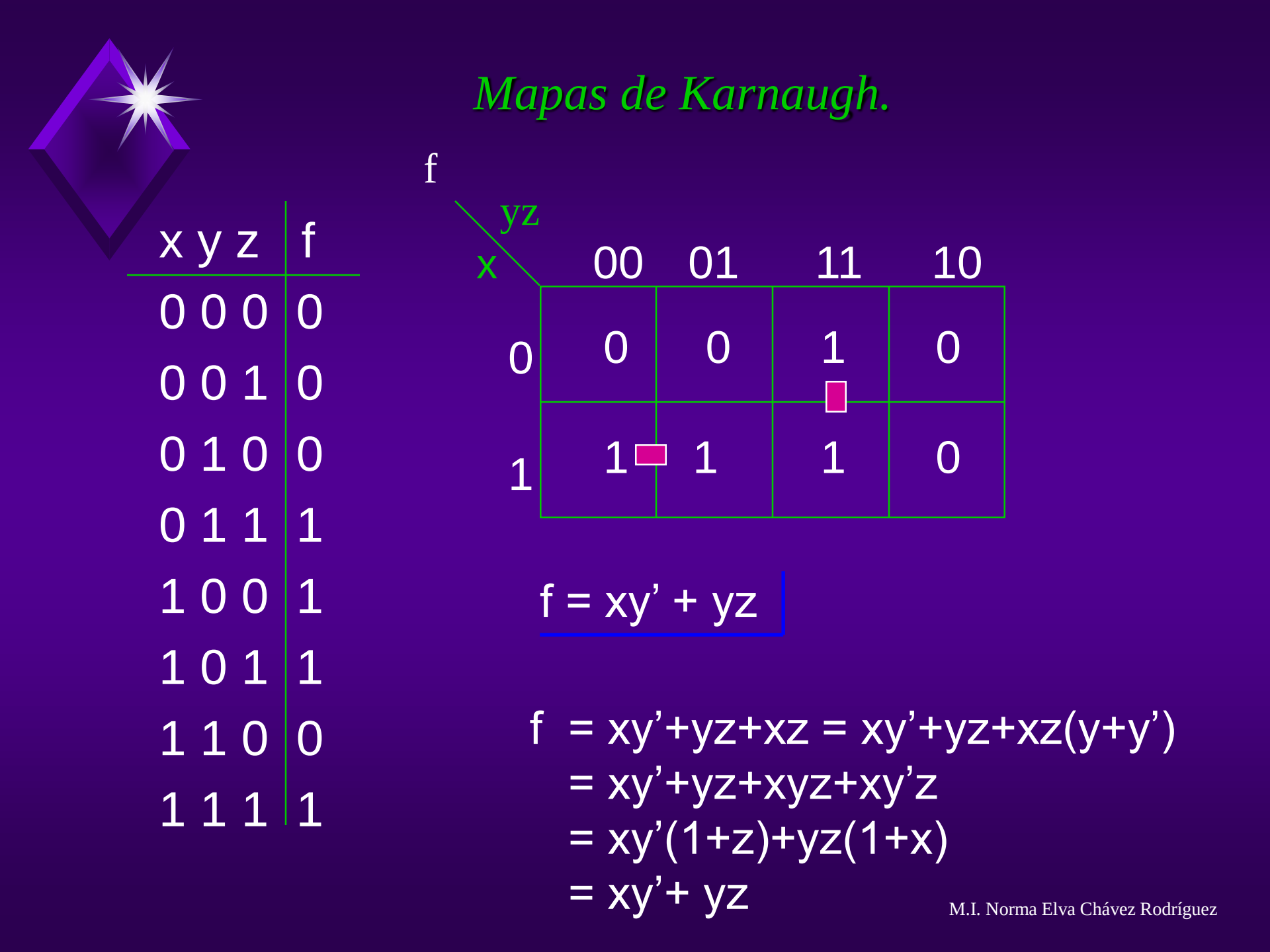
$f1 = x$

# *Mapas de Karnaugh.*

| x y z | f |
|---|---|
| 0 0 0 | 0 |
| 0 0 1 | 0 |
| 0 1 0 | 0 |
| 0 1 1 | 1 |
| 1 0 0 | 1 |
| 1 0 1 | 1 |
| 1 1 0 | 0 |
| 1 1 1 | 1 |

f

| x \ yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 |



f = xy’ + yz

f = xy’+yz+xz = xy’+yz+xz(y+y’)
= xy’+yz+xyz+xy’z
= xy’(1+z)+yz(1+x)
= xy’+ yz

# *Mapas de Karnaugh.*

- P.ejemplo: Dada las siguientes funciones presentarlas en un mapa de karnaugh.

A). f(x,y,z) = xy'+ xz + y'z

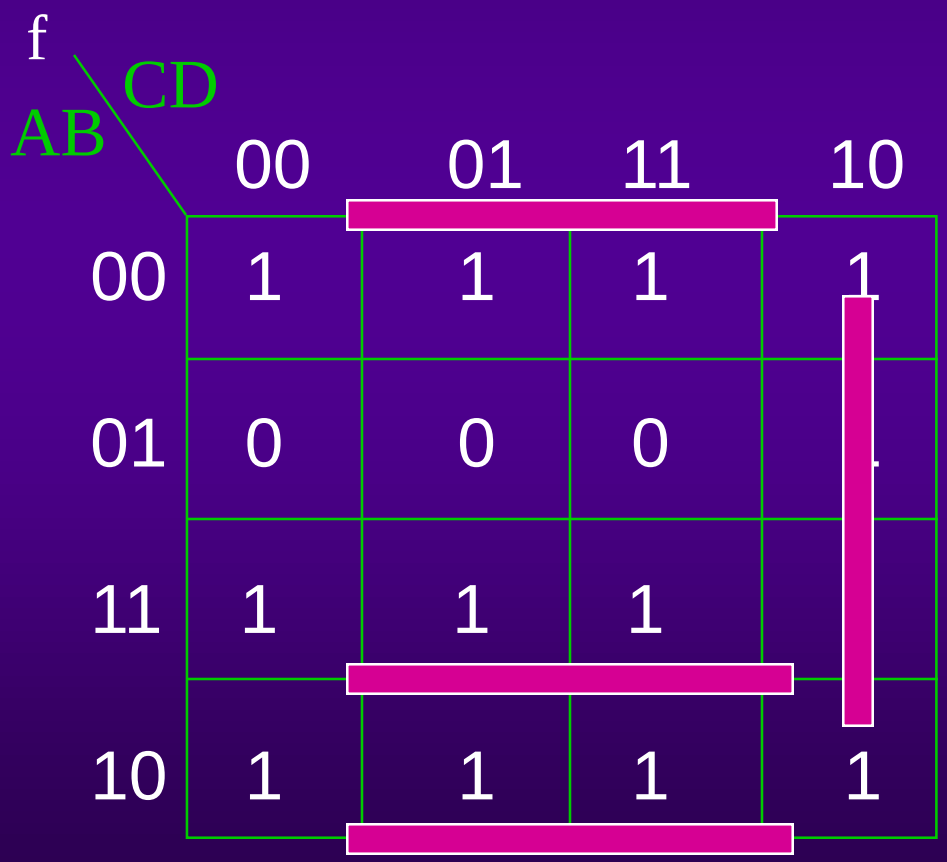
| f | yz | 00 | 01 | 11 | 10 |
|---|---|---|---|---|---|
| x | 0 | 0 | 1 | 0 | 0 |
| | 1 | 1 | 1 | 1 | 0 |

# *Mapas de Karnaugh.*

M.I. Norma Elva Chávez Rodríguez

- B). f(A,B,C,D) = AB + AC'+ CD'+ B'D + B'C'

f = A + B' + CD'



# *Mapas de Karnaugh.*

- Tarea:

c). f (A,B,C,D) = C’D+AB’C’+A’BCD’+A’BD’

d). f (W,X,Y,Z) = XYZ+W’XYZ’+WX’Y’Z+XY’+ WY’Z+ W’YZ’ + YZ

e). f(A,B,C) = ABC+AB’+A’B’C’+AB’+AC’

f). f(X,Y,Z) = XY+Y+Z+X’Y’Z’+X’YZ’

P. ejem. Representar la sig. tabla en un mapa de Karnaugh.

# *Mapas de Karnaugh.*

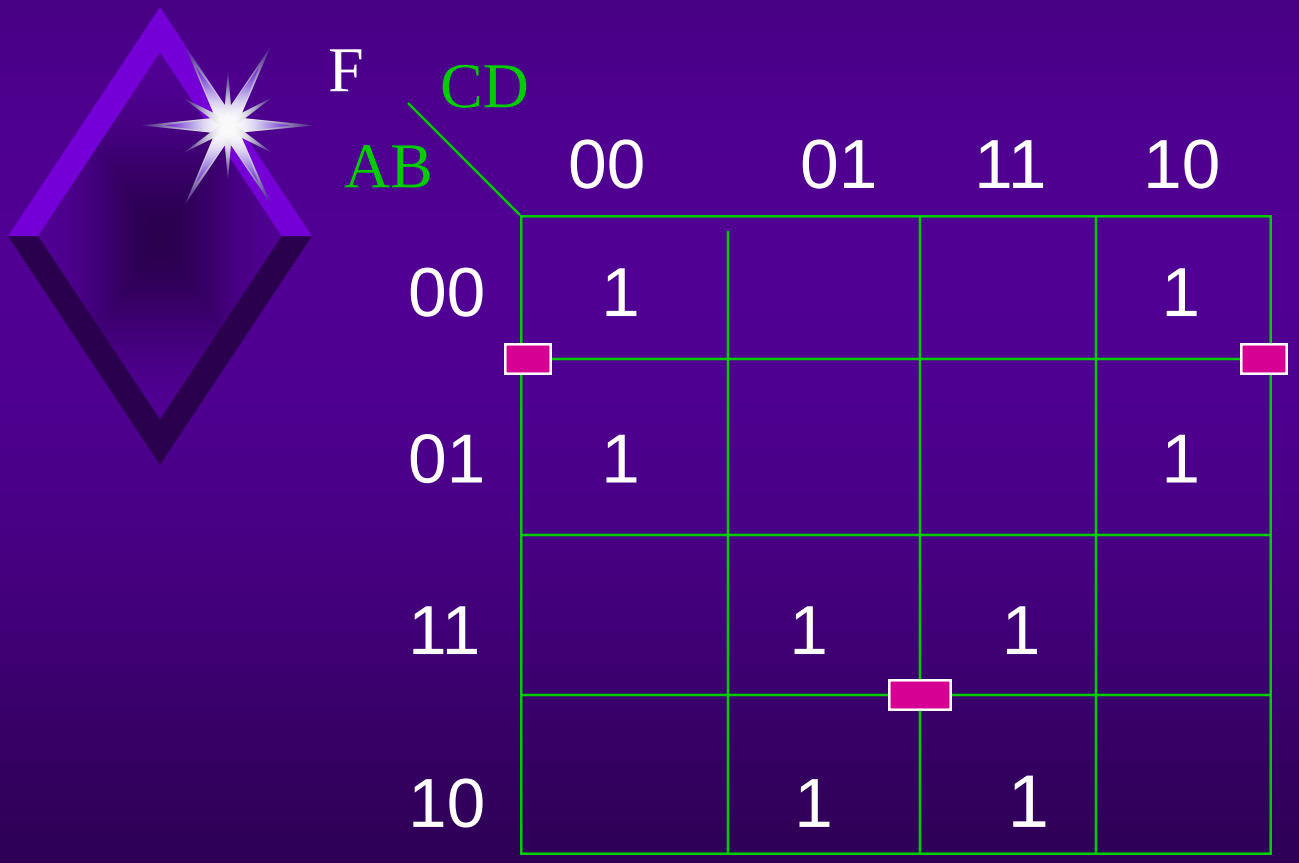
M.I. Norma Elva Chávez Rodríguez

| A | B | C | D | F |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | * |
| 1 | 0 | 1 | 1 | * |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | * |
| 1 | 1 | 1 | 1 | * |

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | 0 | 1 | 0 | 0 |
| 01 | 1 | 1 | 0 | 1 |
| 11 | 0 | 1 | * | * |
| 10 | 0 | 1 | * | * |

# *Mapas de Karnaugh.*

P. ejemplo: Reducir la siguiente función por el método de mapas de Karnaugh.

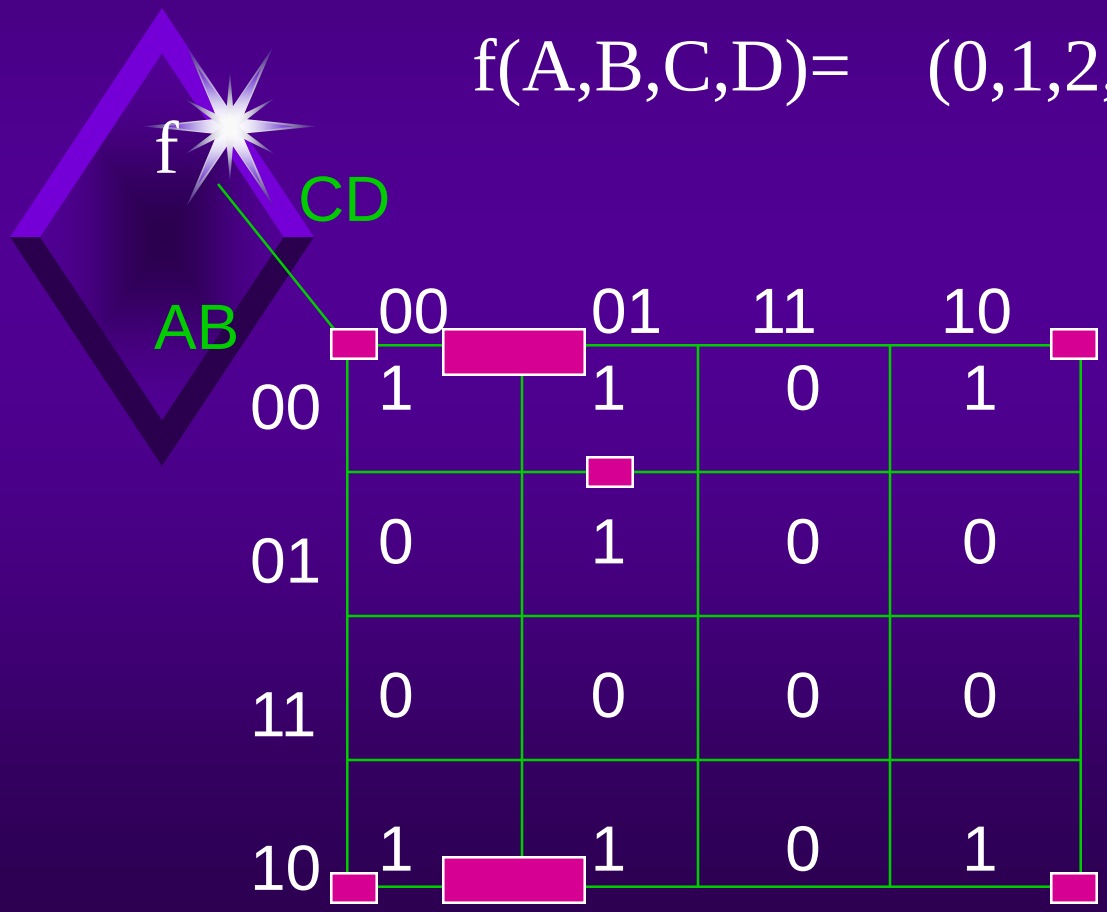



$$F = \bar{A}D + A\bar{D}$$

# *Mapas de Karnaugh.*

P. ejem: Simplificar la siguiente función en suma de productos y productos de sumas.

$$f(A,B,C,D) = (0,1,2,5,8,9,10)$$




$$f = \overline{B}\,\overline{C} + \overline{B}\,\overline{D} + \overline{A}CD$$

# *Mapas de Karnaugh.*

Tarea: Simplificar la siguiente función:

$$f(A,B,C,D,E) = \Sigma\,(0,2,4,6,9,11,13,15,17,21,25,27,29,31)$$

Ejemplo: Hay 4 personas que actúan como jueces en una competencia dada. Cada uno de acuerdo a sus acciones de la empresa tienen cierto peso en su votación Juan=40%, Pedro= 30%, Pablo =20%, José = 10%.
Si el porcentaje es mayor que el 50% se considera aceptado, si no es rechazado. Diseñar un circuito que muestre el resultado, transparente para los votantes.

# *Mapas de Karnaugh.*

| 40 Ju | 30 Pe | 20 Pa | 10 Jo | S1 |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 |

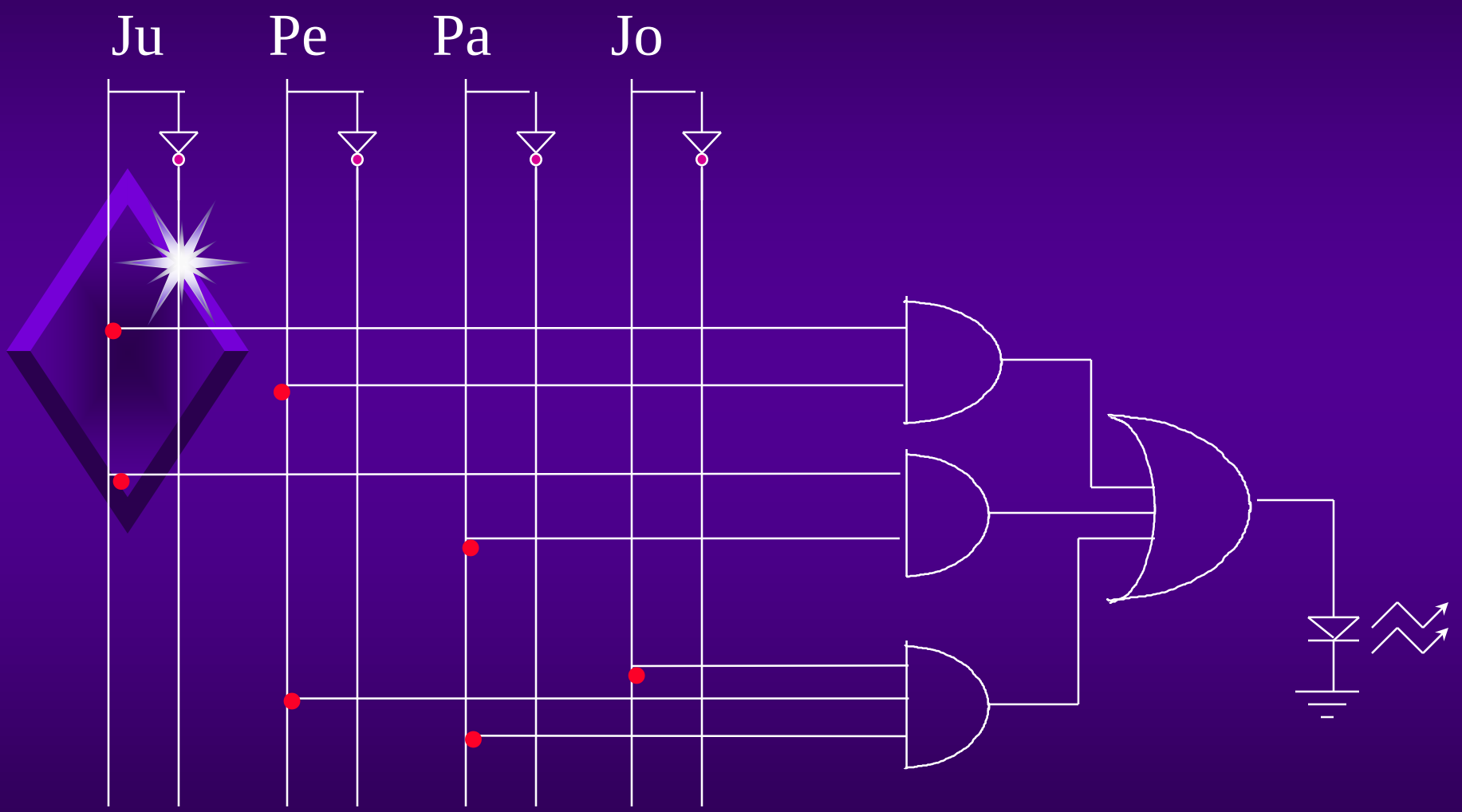
| 40 Ju | 30 Pe | 20 Pa | 10 Jo | S1 |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

M.I. Norma Elva Chávez Rodríguez

# *Mapas de Karnaugh.*

$S_1 = Ju\ Pe + Ju\ Pa + Pe\ Pa\ Jo$

# *Mapas de Karnaugh.*




M.I. Norma Elva Chávez Rodríguez

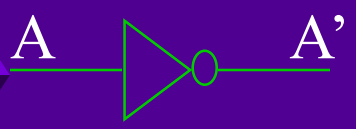
# Universalidad de las compuertas NAND y NOR.

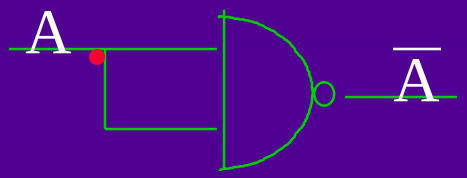
Cualquier función se puede representar con compuertas NAND y NOR.

M.I. Norma Elva Chávez Rodríguez

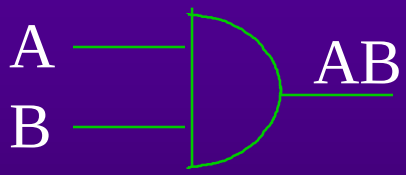
Con NAND

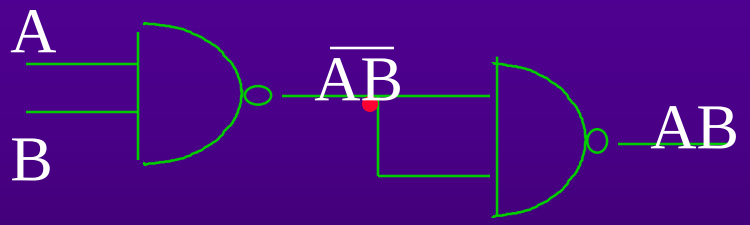
F = A’



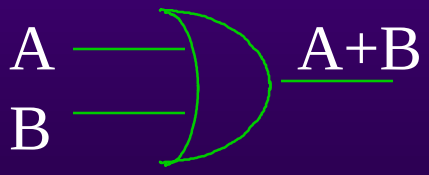



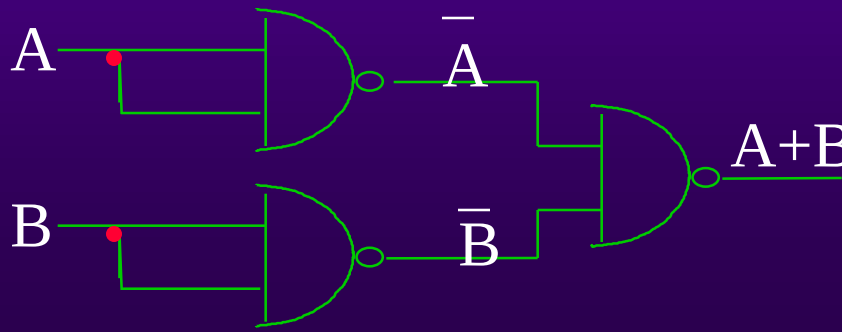

F = AB




F = A+B



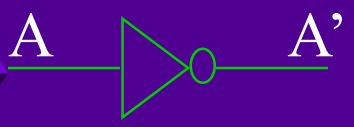
# Universalidad de las compuertas NAND y NOR.

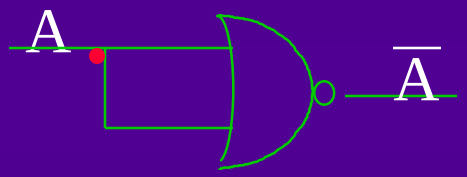
Cualquier función se puede representar con compuertas NAND y NOR.

M.I. Norma Elva Chávez Rodríguez

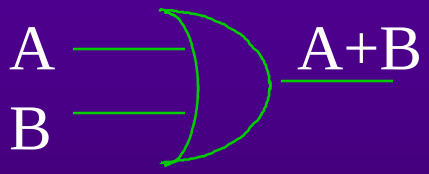
Con NOR

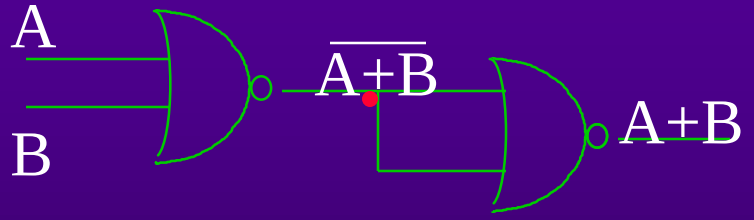
F = A’



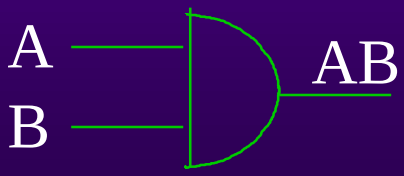



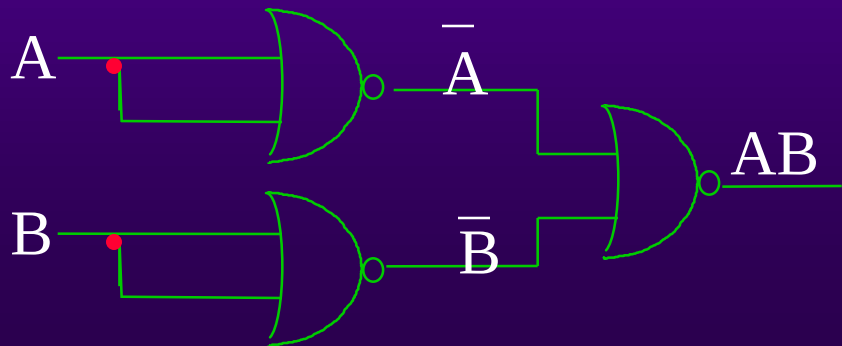

F = A+B




F = AB



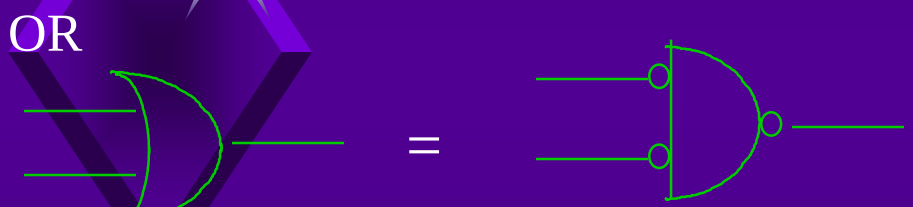



# Universalidad de las compuertas NAND y NOR.

Resumen:

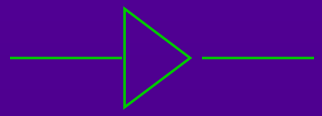
AND

OR



NOT

BUFFER



NAND

NOR

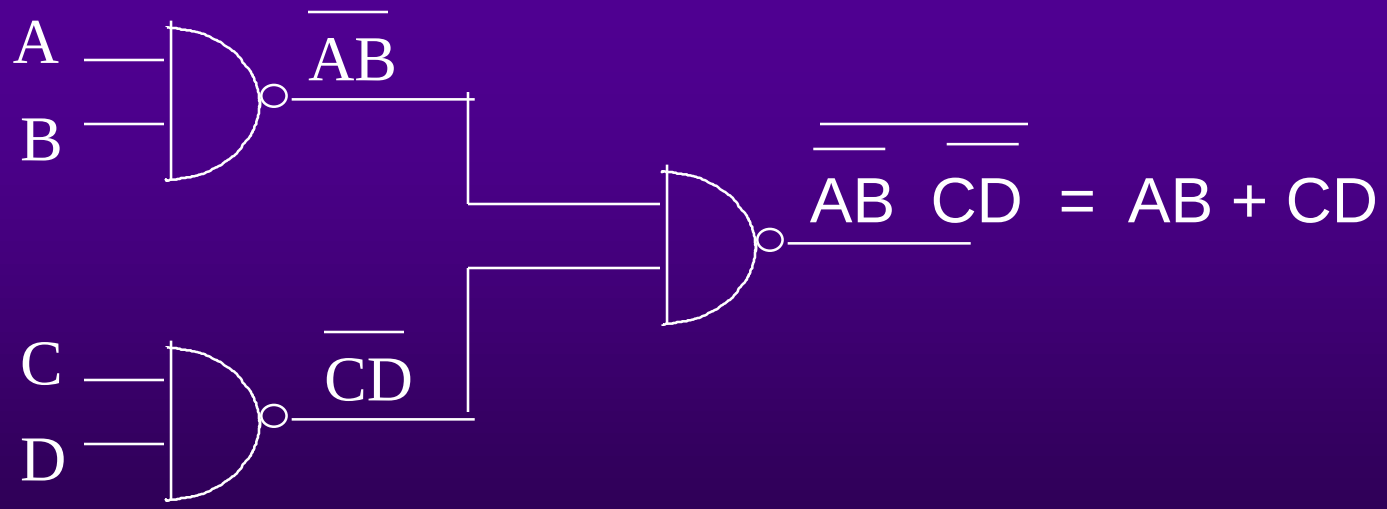
M.I. Norma Elva Chávez Rodríguez

# Universalidad de las compuertas NAND y NOR.

Por ejemplo:

Representa la siguiente función con compuertas NAND:

$$F(A,B,C,D)=AB + CD = \overline{\overline{AB+CD}} = \overline{\overline{AB}\ \overline{CD}}$$



# Universalidad de las compuertas NAND y NOR.

Ejercicio:

Para el siguiente ejercicio construya el circuito lógico usando sólo compuertas NAND ó NOR.

A) $F=AB\,(C+D) = AB + (C+D)$

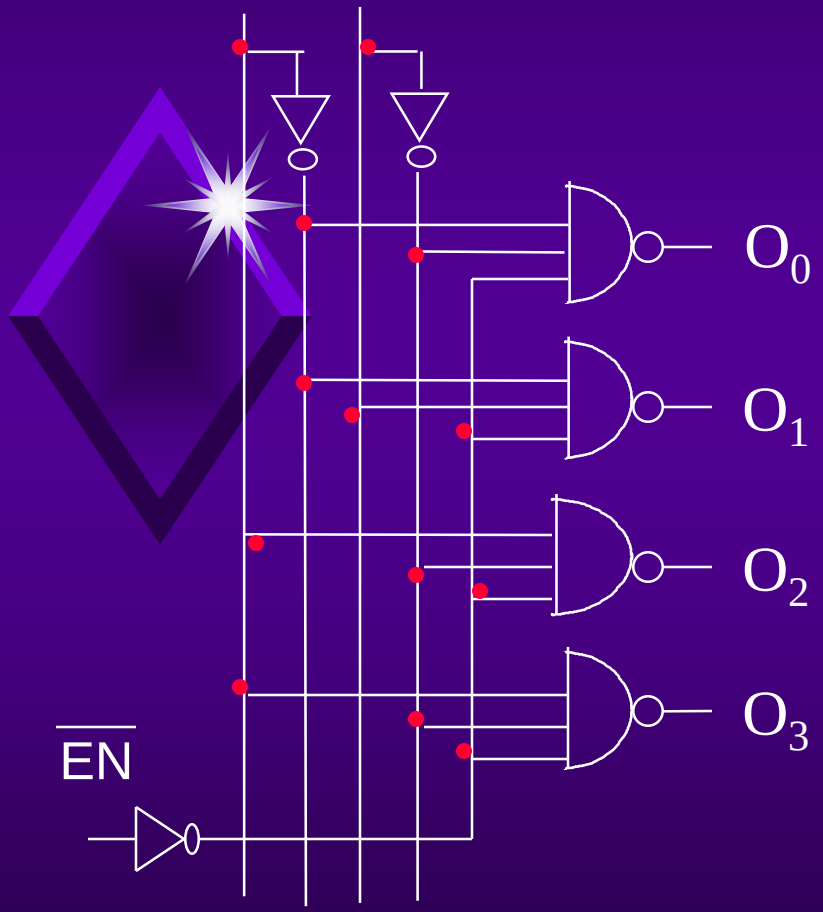
# Decodificadores.

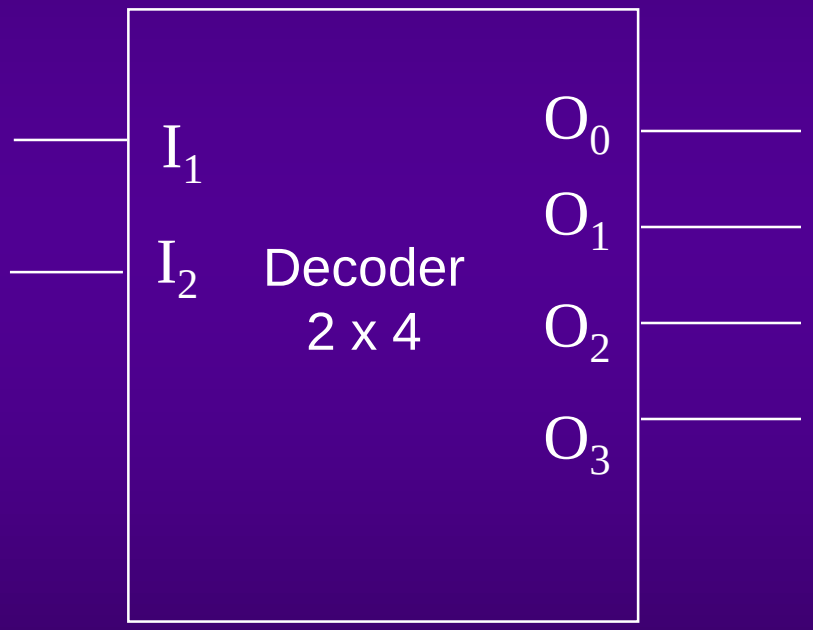
Un decodificador es un circuito combinacional que convierte información binaria de N entradas a $2^N$ salidas; que con frecuencia se les refiere como decodificadores NxM donde $M = 2^N$.

Un decodificador genera $2^N$ minitérminos.

Supongamos que tenemos dos entradas por lo tanto tendremos 2 salidas. Esto es un decodificador 2 x 4.

# Decodificadores.



# Decodificadores.

Tabla de verdad.

| X | Y | D0 | D1 | D2 | D3 |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 |

Un decodificador nos puede servir para representar funciones.

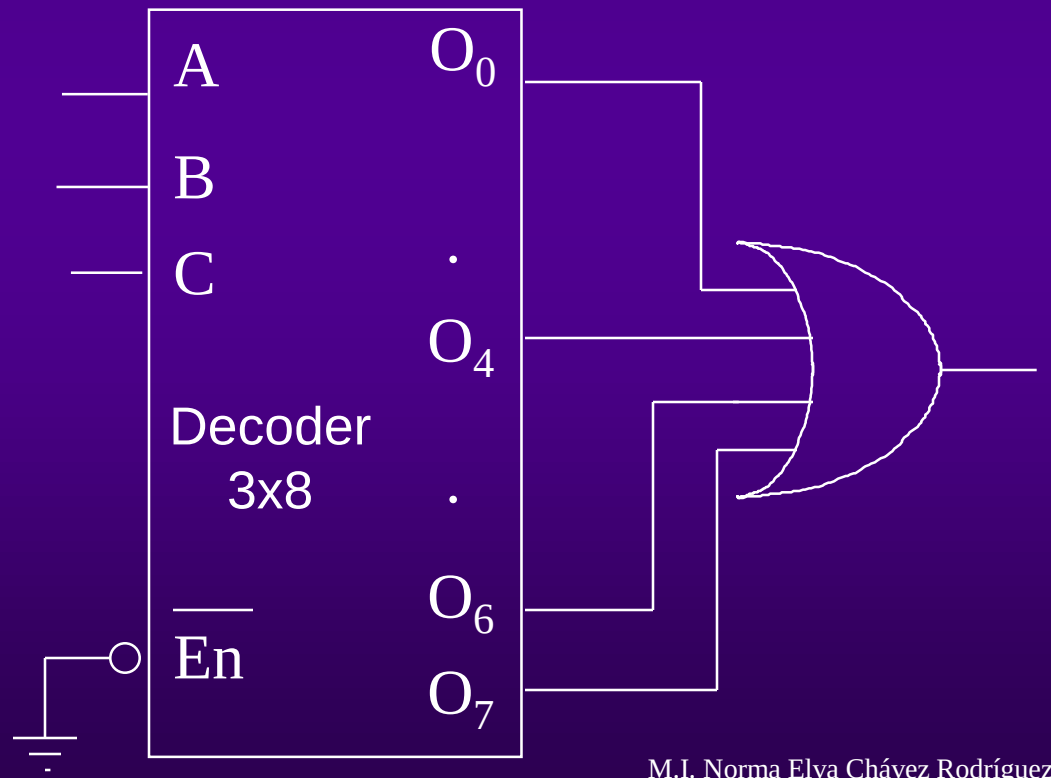
# Decodificadores.

Ejemplo 1:

Implemente la siguiente función utilizando un decoder de 3x8.

$$F(A,B,C) = AB + A\overline{C} + ABC + \overline{A}\,\overline{B}\,\overline{C}$$

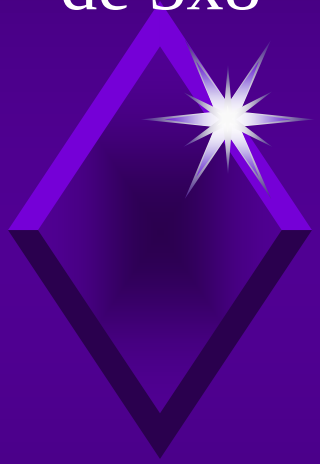
| A | B | C | F |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |



# Decodificadores.

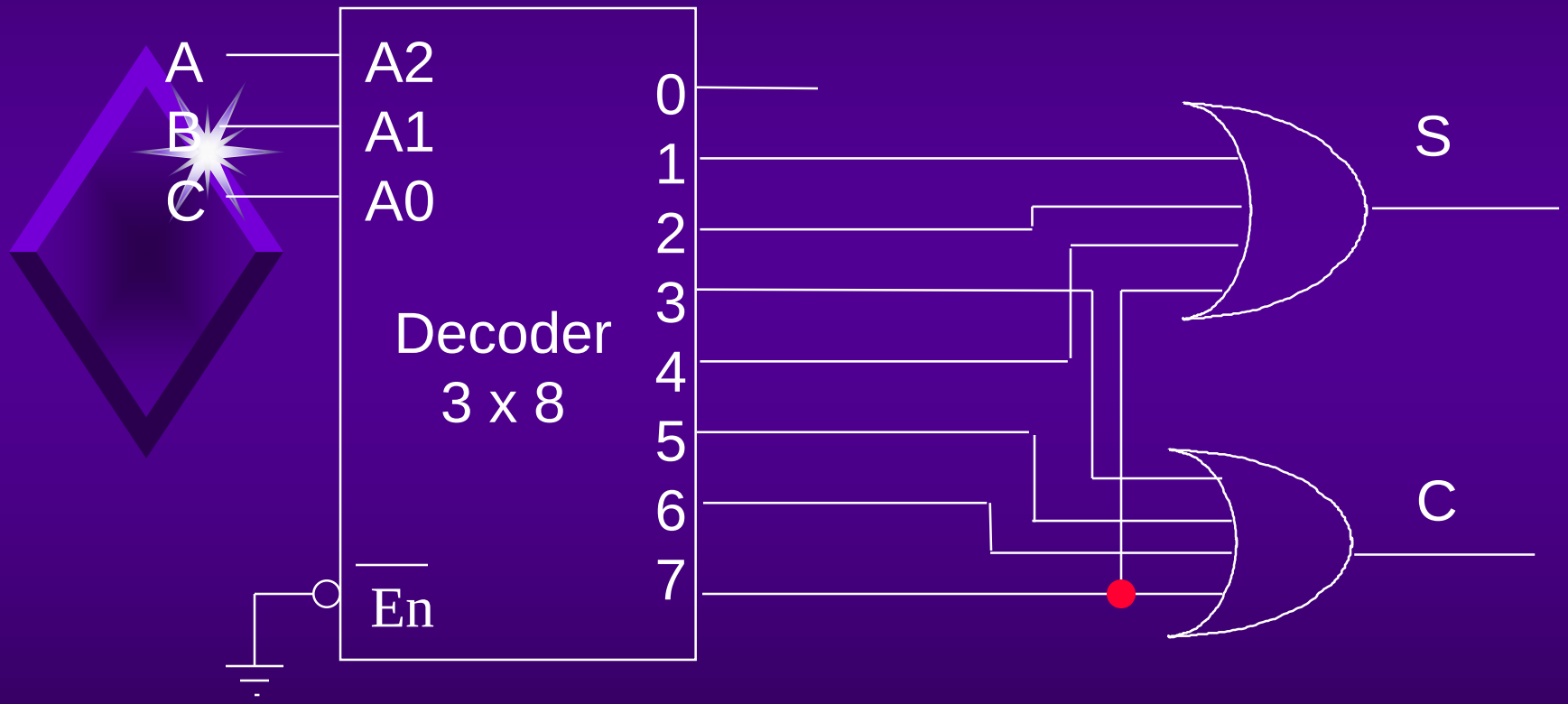
Ejemplo 2

Diseñe un sumador completo utilizando un decodificador de 3x8



| A | B | C | S | C |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

# Decodificadores.



# Decodificadores.

Decodificadores con entradas “Enable” .

Estas entradas sirven para controlar la operación del decodificador. El 74L5138 tiene salidas negadas.

# Decodificadores.

Circuito interno

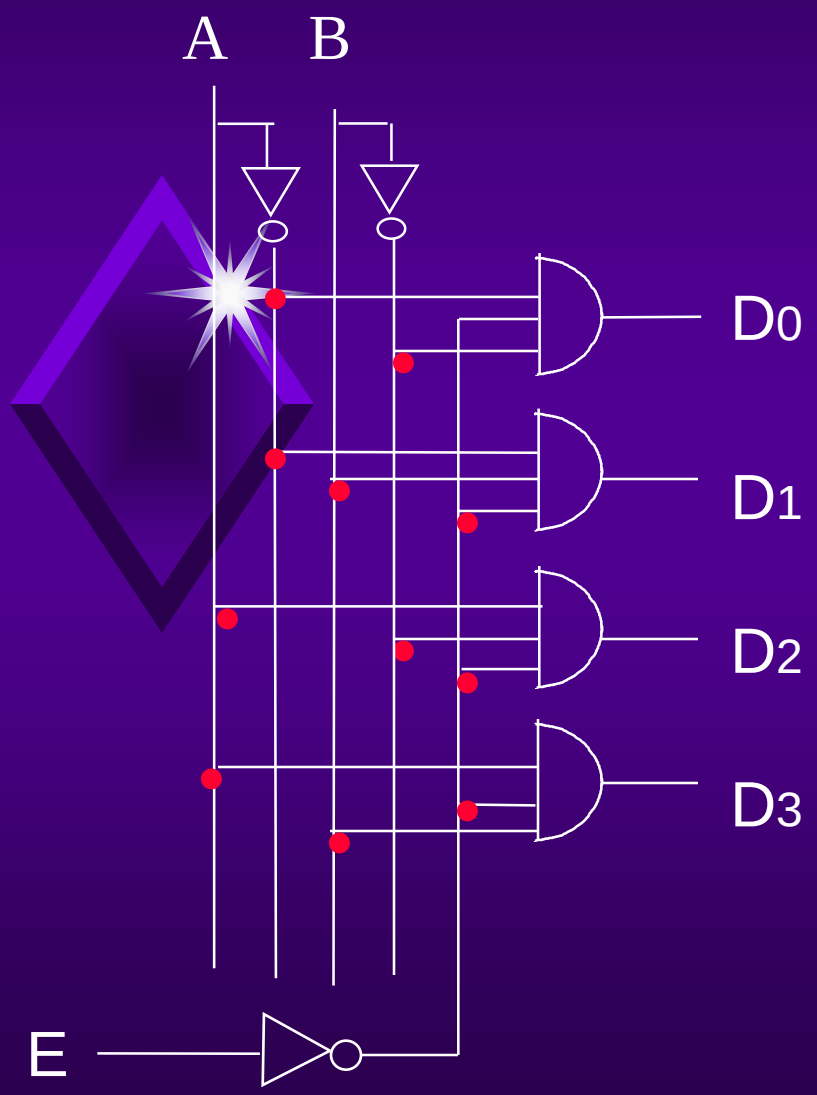
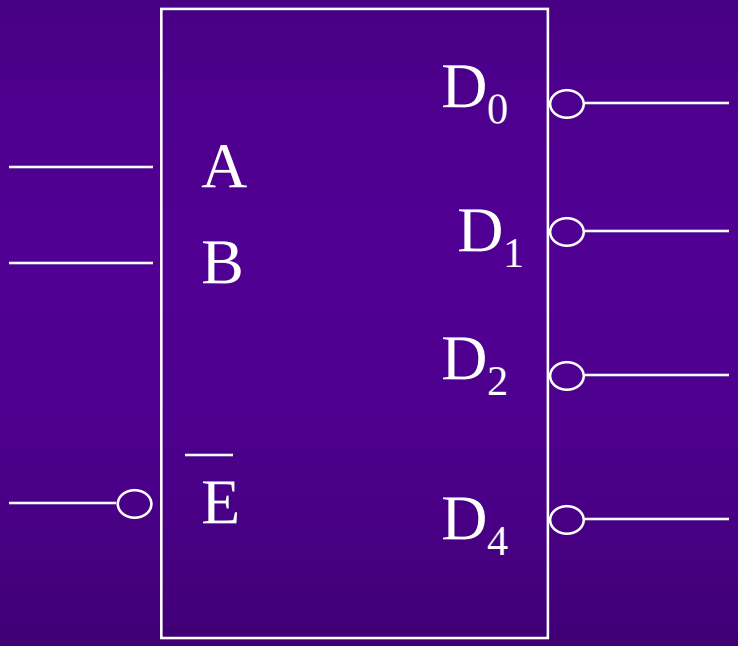



Tabla de verdad

| E | A | B | Do | D1 | D2 | D3 |
|---|---|---|---|---|---|---|
| 1 | x | x | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |

# Decodificadores.

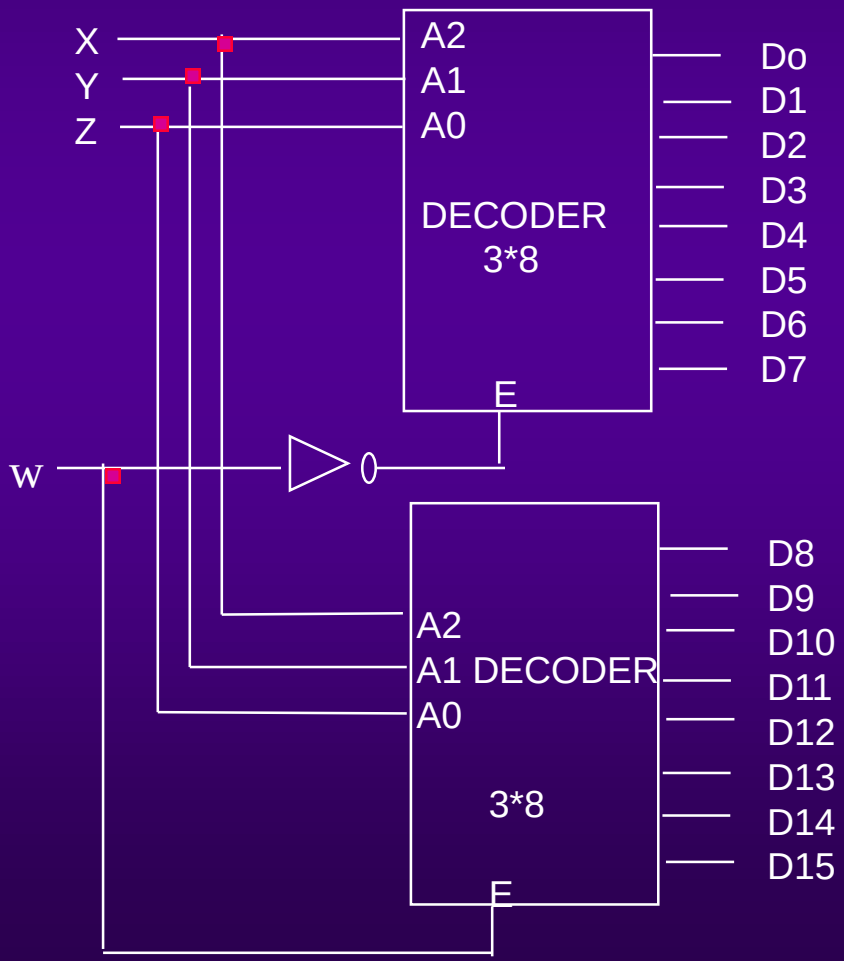
## Simbolo del decodificador 74L5138




Decoder de 2x4

# Decodificadores.

Diseñe un decodificador de 4x16 usando dos decoder's de 3x8 con entrada enable




Algunos decodificadores importantes
74LS139
74AC154
74LC138

# Decodificadores.

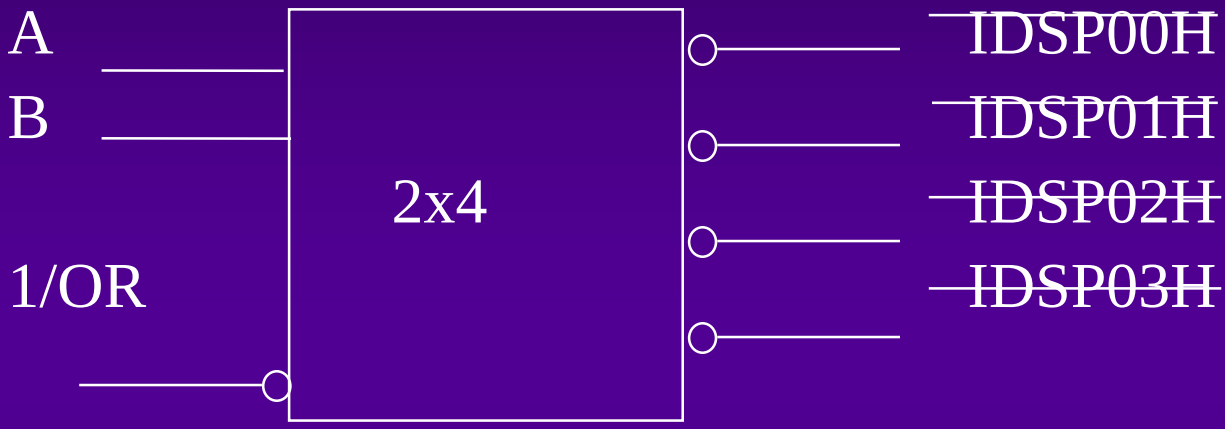
## Tabla de verdad

| X YZW | D0 | D1 | D2 | D3 | D4 | D5 | D6 | D7 | D8 | D9 | D10 | D11 | D12 | D13 | D14 | D15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 0 0 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 0 0 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 0 1 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 0 1 1 | | | | | | | | | | | | | | | | |
| 0 1 0 0 | | | | | | | | | | | | | | | | |
| 0 1 0 1 | | | | | | | | | | | | | | | | |
| 0 1 1 0 | | | | | | | | ▪ | | | | | | | | |
| 0 1 1 1 | | | | | | | | ▪ | | | | | | | | |
| 1 0 0 0 | | | | | | | | ▪ | | | | | | | | |
| 1 0 0 1 | | | | | | | | | | | | | | | | |

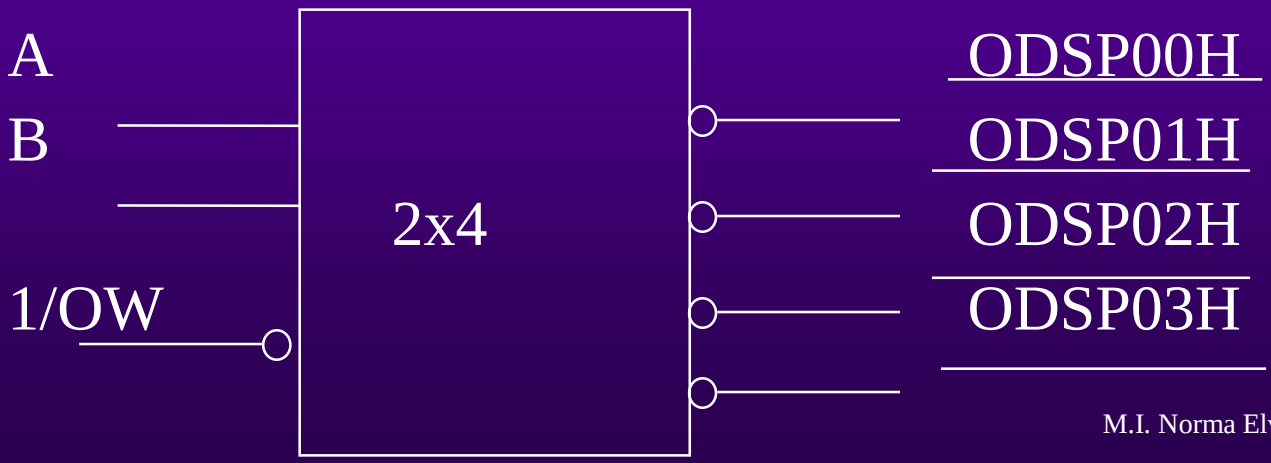
# Decodificadores.

Para los puertos de entrada.

74LS139

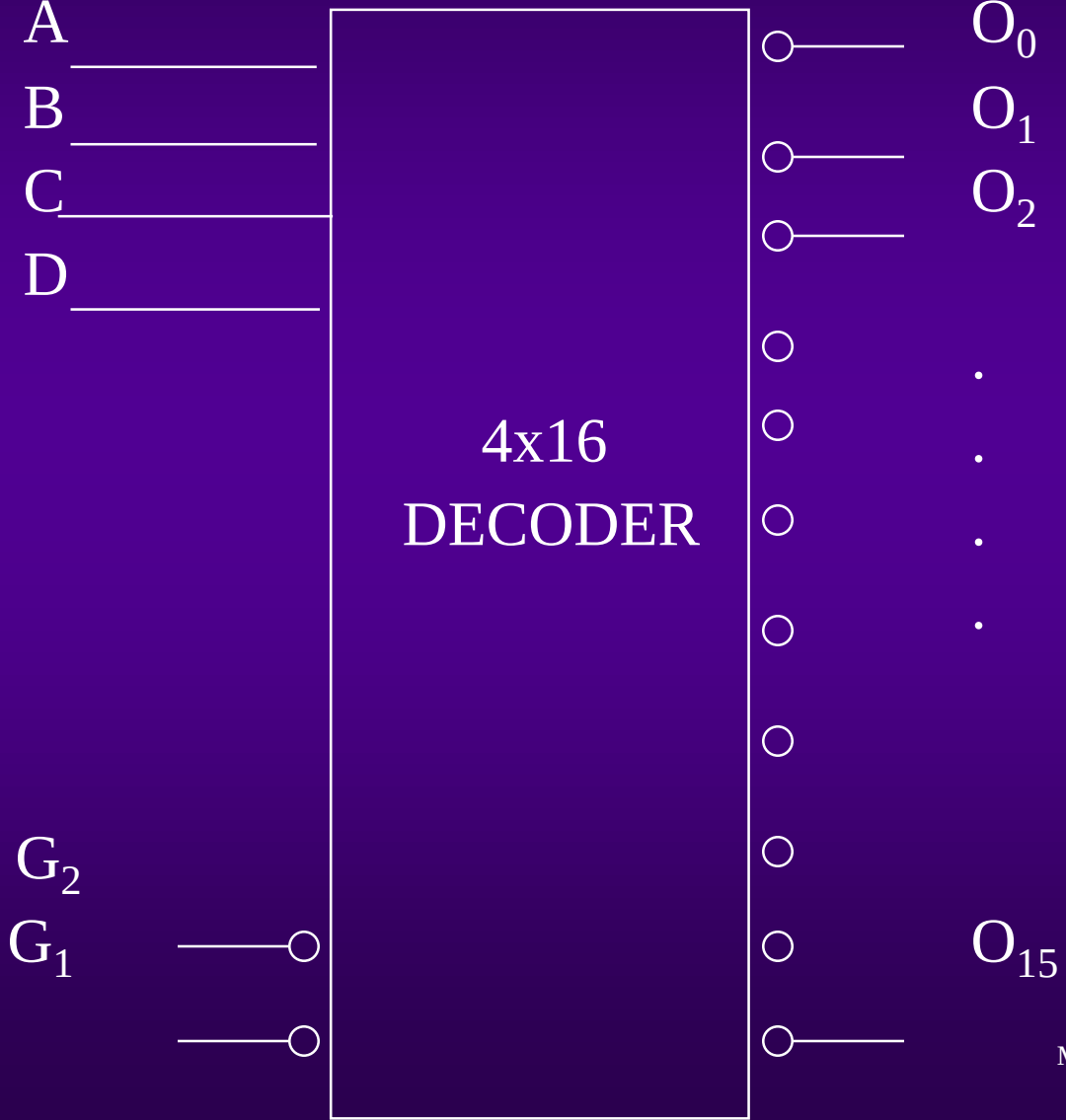



Para los puertos de salida



# Decodificadores.

## 74C154

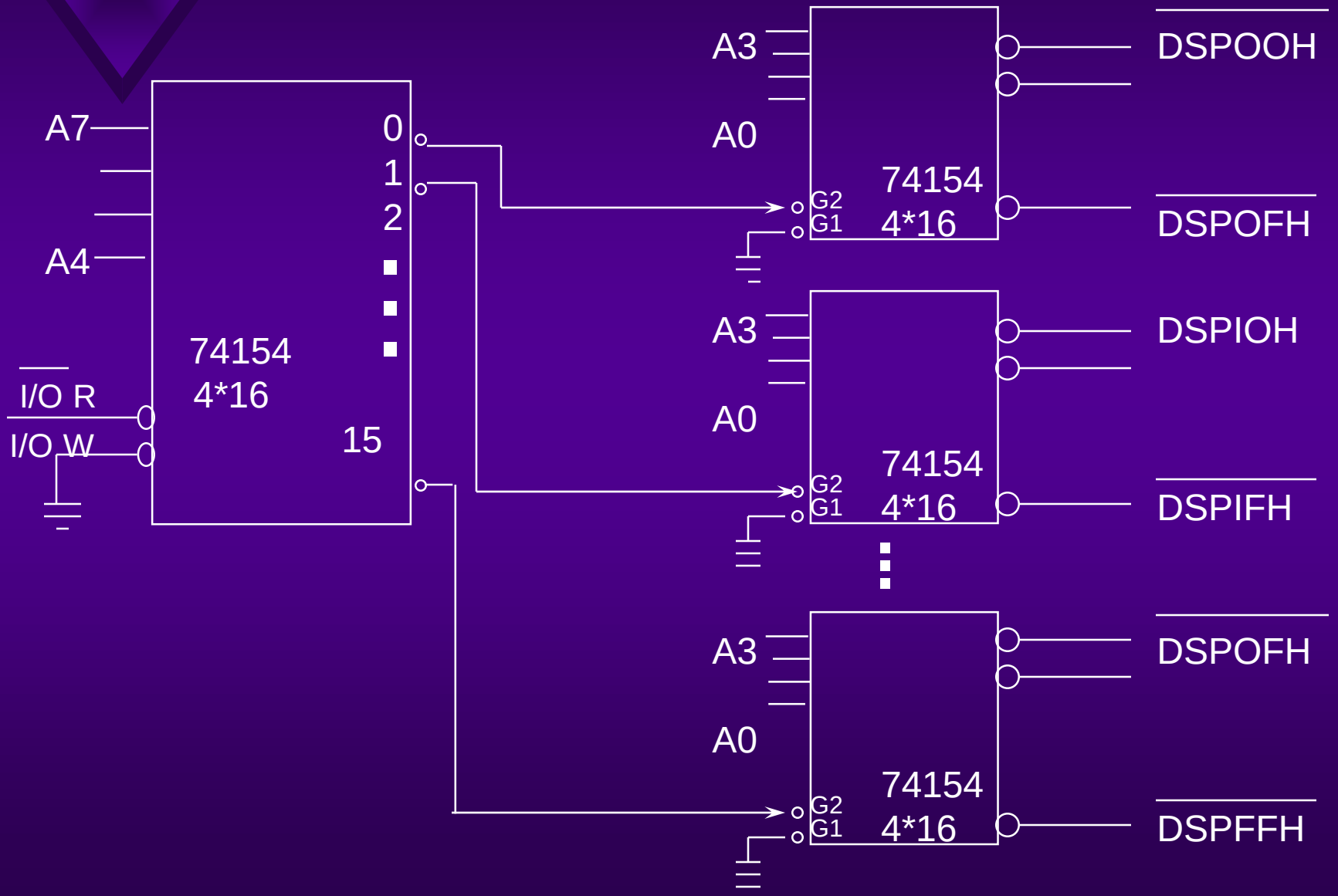



M.I. Norma Elva Chávez Rodríguez

# Decodificadores.

M.I. Norma Elva Chávez Rodríguez

Generación de los pulsos selectores de los dispositivos.



# Multiplexor.

Un multiplexor es un circuito combinacional que selecciona una línea de entrada de entre varias.

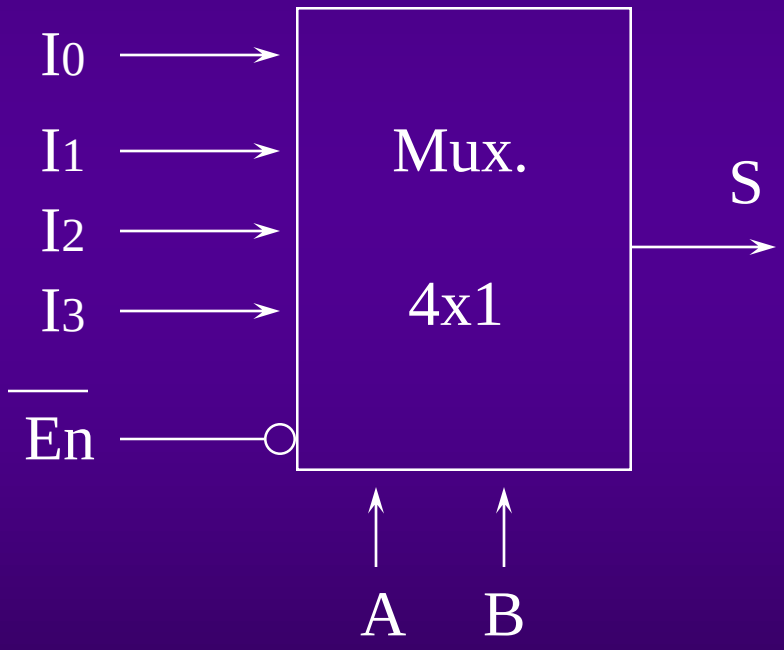
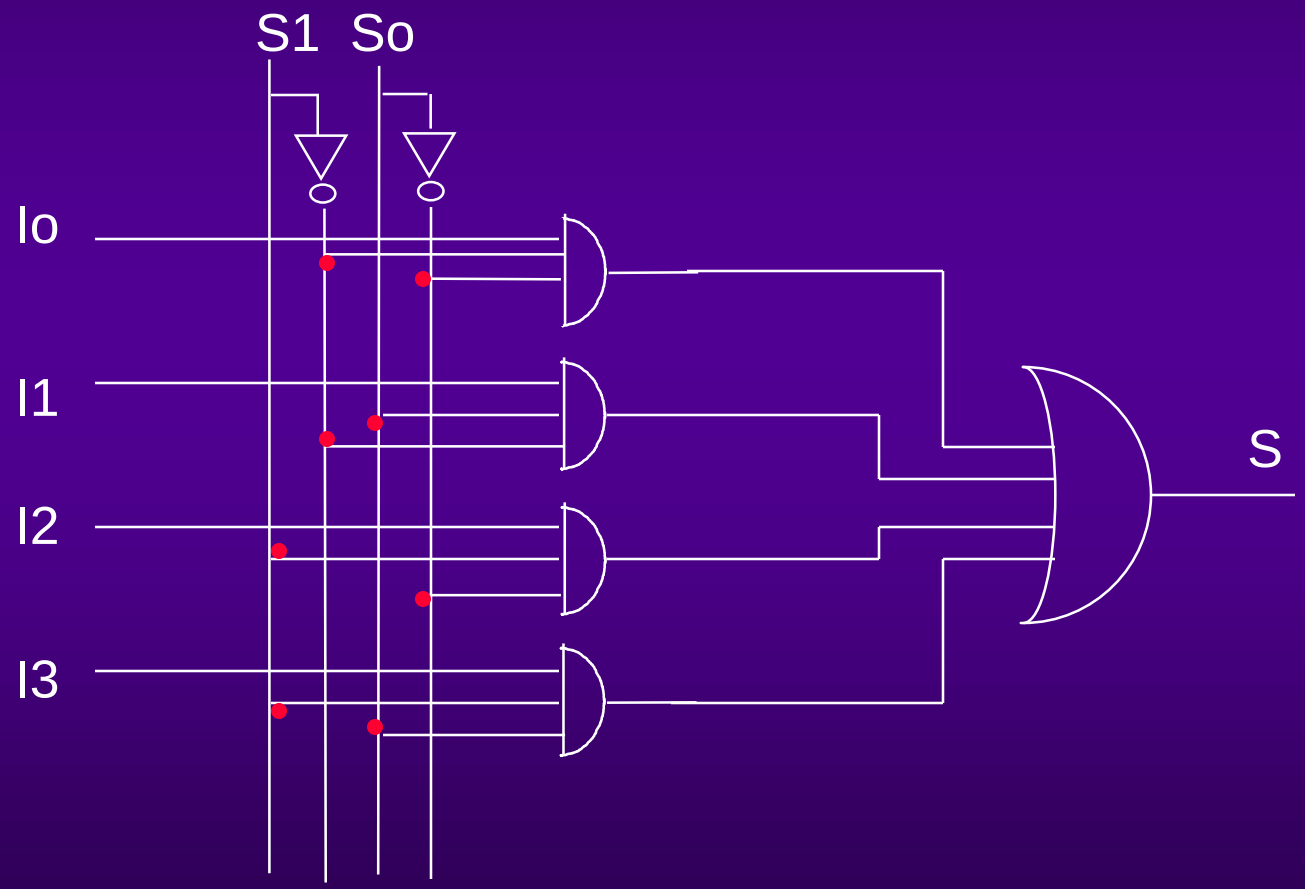



Tabla de verdad

| S1 | S0 | S |
|---|---|---|
| 0 | 0 | I0 |
| 0 | 1 | I1 |
| 1 | 0 | I2 |
| 1 | 1 | I3 |

# Multiplexor.

## Circuito Interno



# Multiplexor.

Para implementar una función booleana de N variables se necesita un multiplexor de N - 1 entradas de selección .

Las primeras N-variables (las de menor peso) de la función se conectan a las entradas de la selección del multiplexor . La variable que resta de la función se utiliza para la entrada de datos .

# Multiplexor.

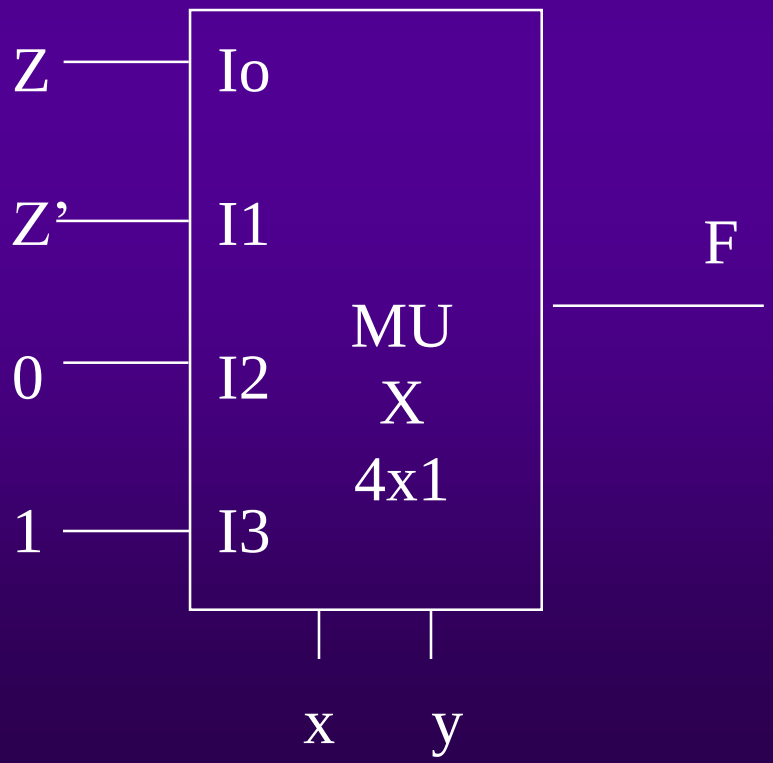
M.I. Norma Elva Chávez Rodríguez

Por ejemplo:

Implemente la siguiente función con un MUX de 4x1 y una variable residual.

F (x,y,z) = (1,2,6,7)

| x | y | z | F | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | |
| 0 | 0 | 1 | 1 | z |
| 0 | 1 | 0 | 1 | |
| 0 | 1 | 1 | 0 | z’ |
| 1 | 0 | 0 | 0 | |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | |
| 1 | 1 | 1 | 1 | 1 |

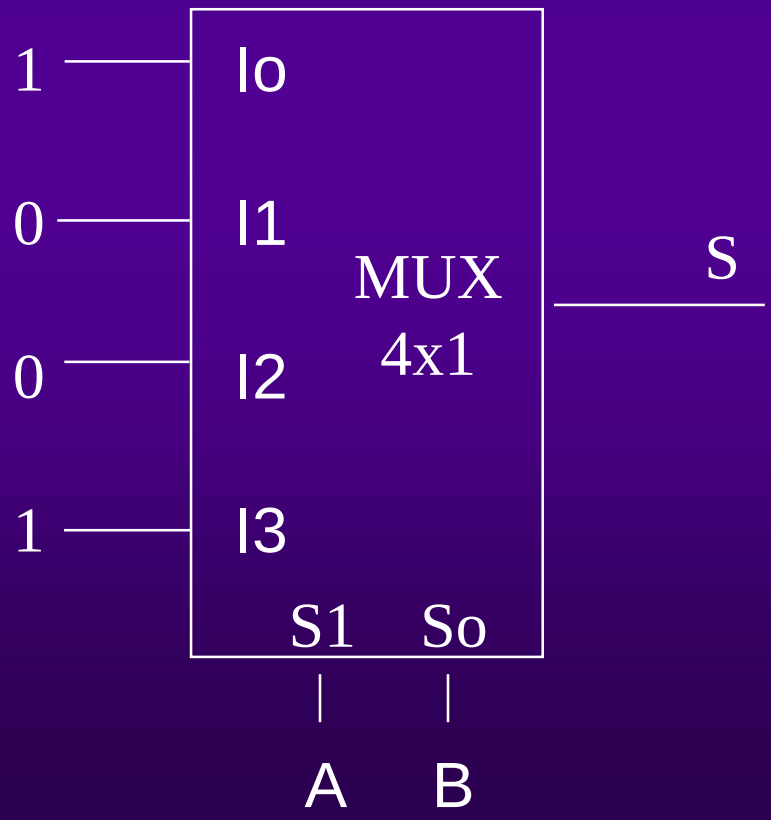


# Multiplexor.

Ejemplo: Implemente la siguiente función con un MUX de 4x1 y una variable residual

F(A,B,C)=A'B'C+A'B'+AB

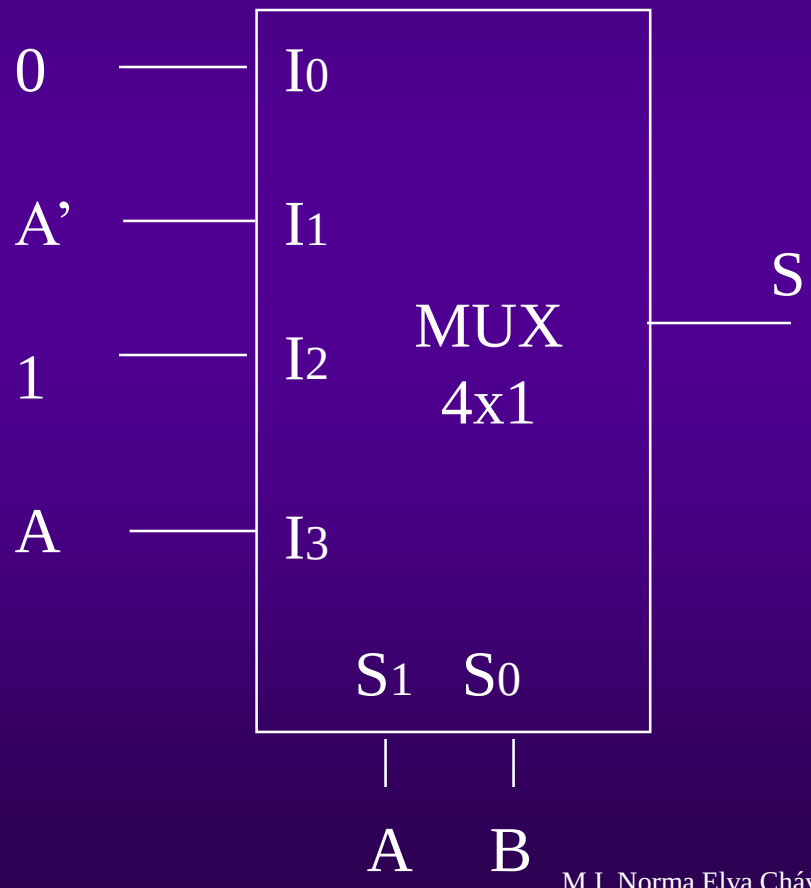
| A | B | C | F | |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | |
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | |
| 1 | 1 | 1 | 1 | 1 |



# Multiplexor.

ó también:

| | I0 | I1 | I2 | I3 |
|---|---|---|---|---|
| A’ | 0 | 1 | 1 | 0 |
| A | 0 | 0 | 1 | 1 |
| | 0 | A’ | 1 | A |



# Multiplexor.

Ejemplo: Implemente la siguiente función con un multiplexor de 8x1.

$F(A,B,C,0)=\sum(0,1,3,4,8,9,15)$

| A | B | C | D | F | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | |
| 0 | 0 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | |
| 0 | 0 | 1 | 1 | 1 | D |
| 0 | 1 | 0 | 0 | 1 | |
| 0 | 1 | 0 | 1 | 0 | D’ |
| 0 | 1 | 1 | 0 | 0 | |
| 0 | 1 | 1 | 1 | 0 | 0 |

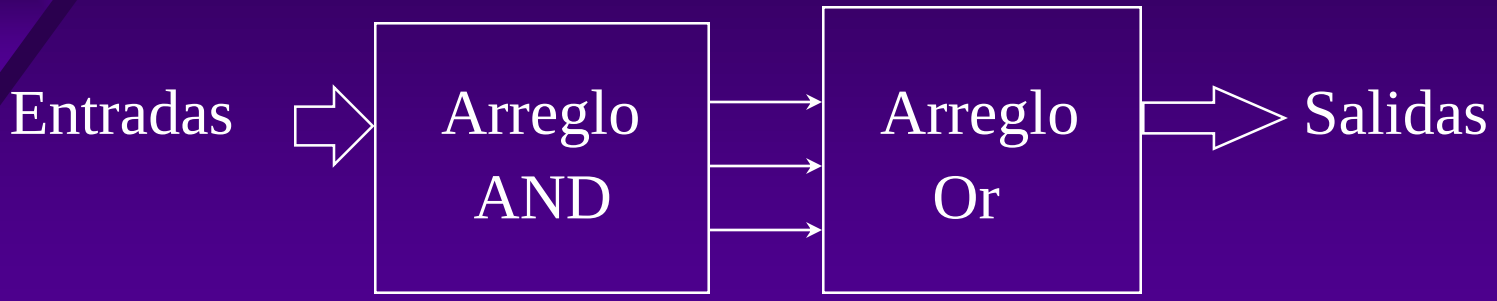
| A | B | C | D | F | |
|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 | |
| 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 | |
| 1 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | |
| 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | |
| 1 | 1 | 1 | 1 | 1 | D |

# Dispositivos Lógicos Programables.

| | |
|---|---|
| 1.- PAL’s | Programmable Array Logic. |
| 2.- PLA’s | Programmable Logic Arrays. |
| 3.- ROM´s | Read only Memory |
| 4.- EPROM´s | Erase Programmable read only memory. |

Los arreglos lógicos programables son dispositivos con múltiples entradas y múltiples salidas organizadas en un sub-arreglo AND y otro OR.

# Dispositivos Lógicos Programables.



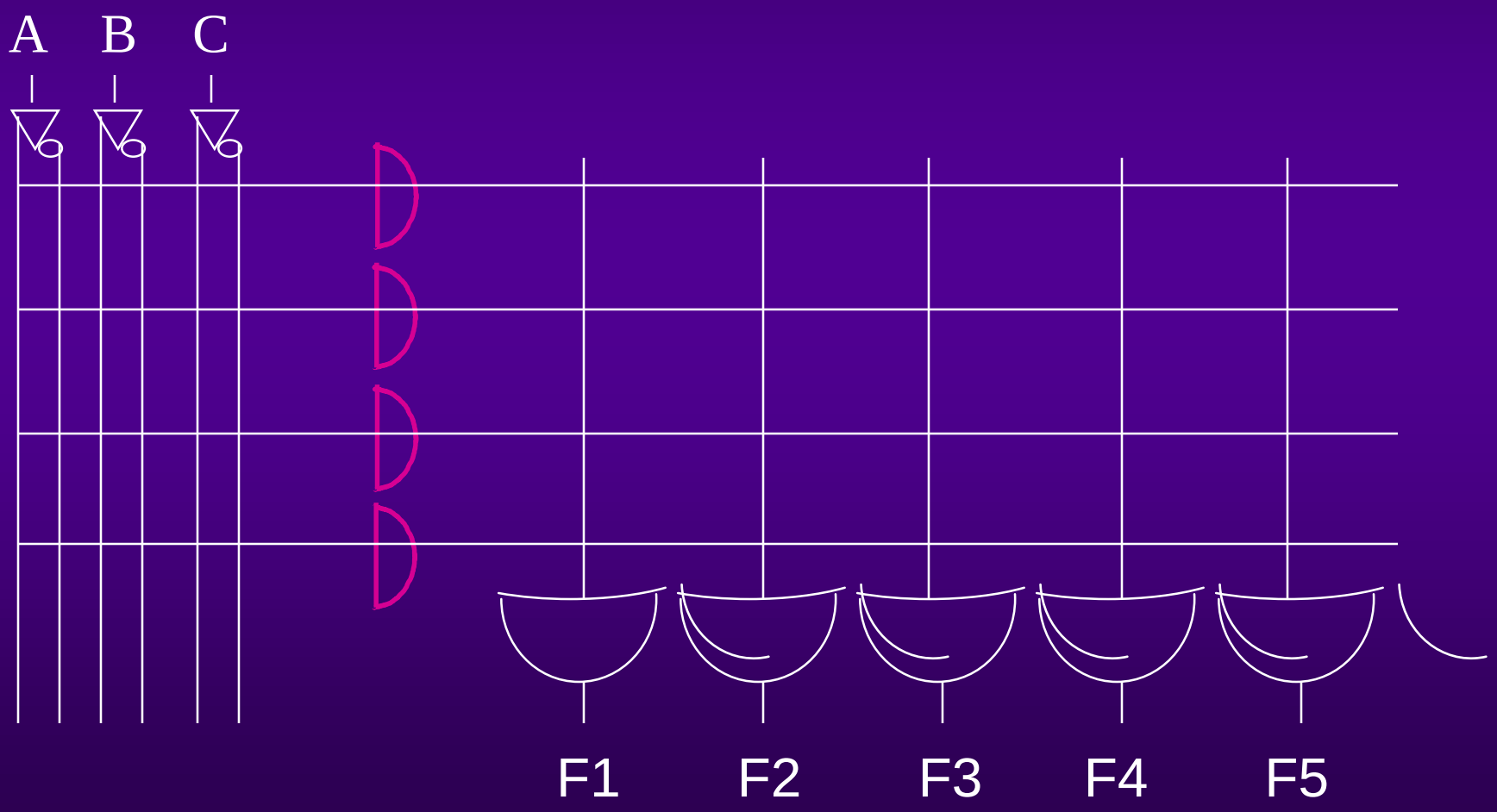

Programación

| | AND | OR |
|---|---|---|
| PLA | Programmable | Programmable |
| PAL | “ | Fijo |
| PROM | Fijo | Programmable |

# Dispositivos Lógicos Programables.

## Ejemplo de un PLA de 3 entradas y 5 salidas:

M.I. Norma Elva Chávez Rodríguez



# Dispositivos Lógicos Programables.

Un PLA TTL Típico tiene 16 entradas 48 productos y 8 salidas en un chip de 24 pines.

Ejemplo:

Use un PLA para diseñar un control de BCD a 7 segmentos.

# Dispositivos Lógicos Programables.

$F_a = A + BD + C + B'D'$

$F_b = A + C'D' + CD + B'$

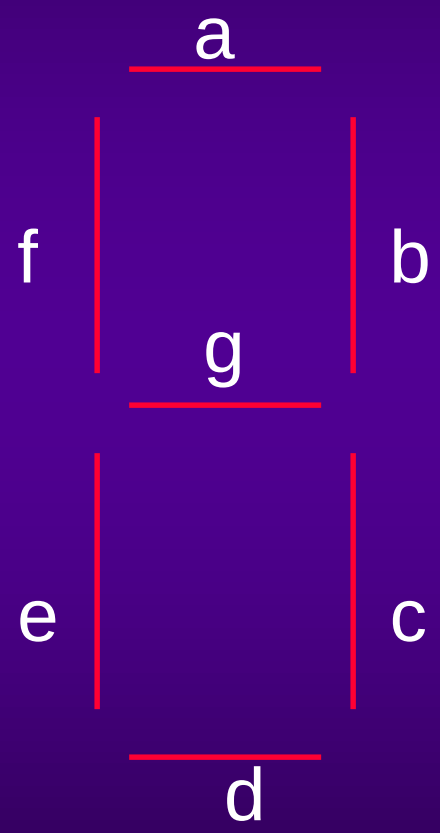
$F_c = A + B + C' + D$

$F_d = B'D' + CD' + BC'D + B'C$

$F_e = B'D' + CD'$

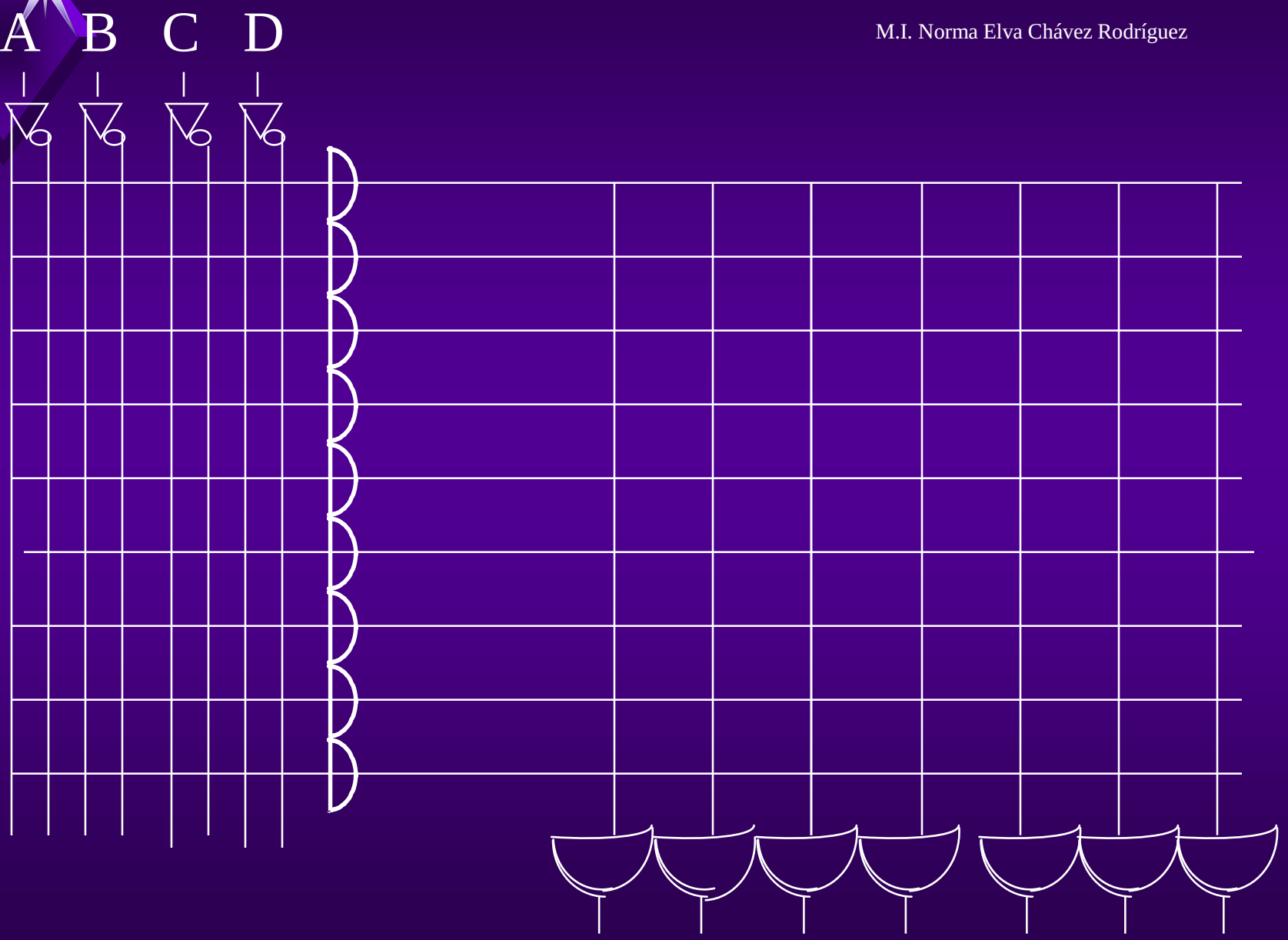
$F_f = A + C'D' + BD' + BC'$

$F_g = A + CD' + BC' + B'C$



# Dispositivos Lógicos Programables.

M.I. Norma Elva Chávez Rodríguez



# Dispositivos Lógicos Programables.

Memorias de sólo lectura son un tipo de memoria de semiconductor que están diseñadas para retener datos que son permanentes o que no cambian con mucha frecuencia .

En algunas ROM los datos se tienen que grabar a la hora de la fabricación, algunos otros pueden programarse eléctricamente.

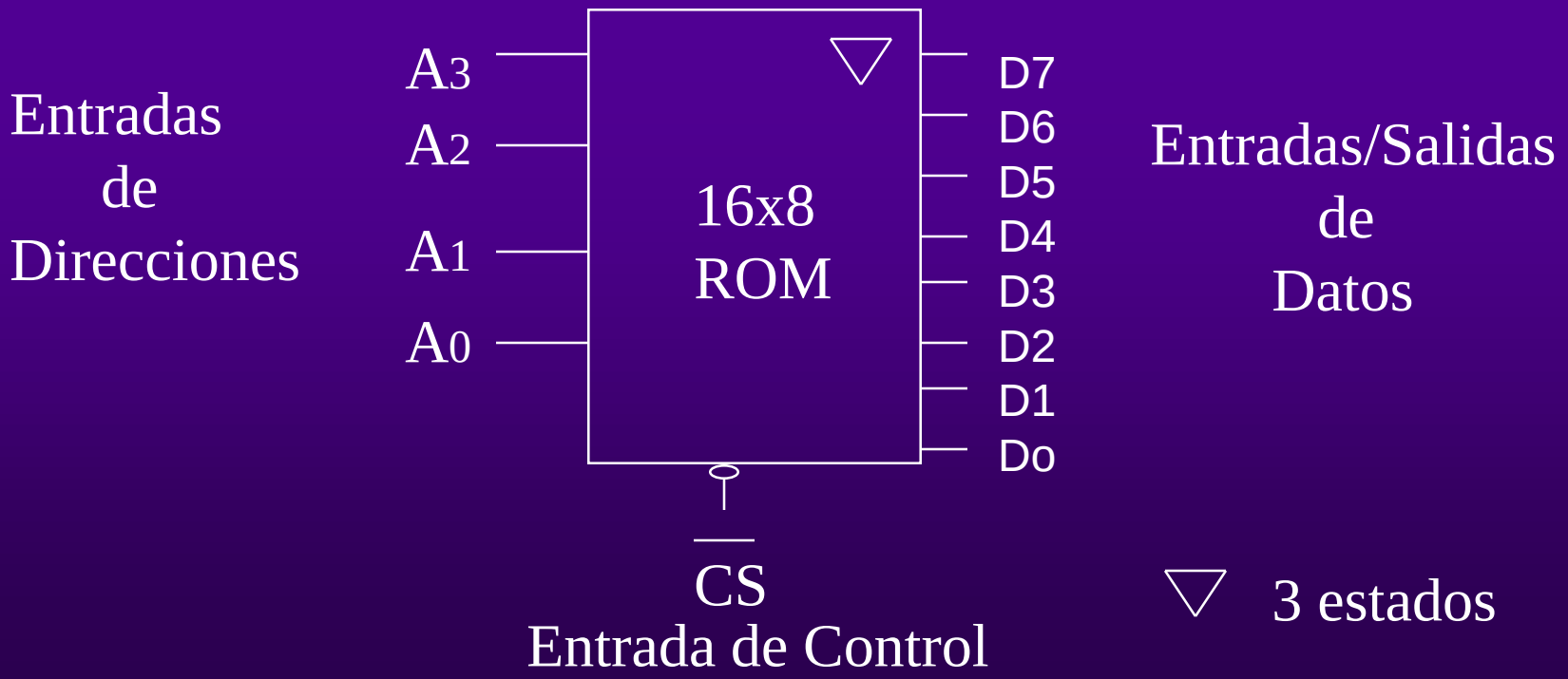
La ROM se usa para almacenar datos e información que no cambiará durante la operación de un sistema.

# Dispositivos Lógicos Programables.

M.I. Norma Elva Chávez Rodríguez

## Diagrama a bloques de una ROM

Una ROM tiene tres conjuntos de señales : entradas/salidas de datos, entradas de direcciones y líneas de control.



# Dispositivos Lógicos Programables.

Esta Rom almacena 16 palabras , ya que tiene $2^4$ = 16 posibles direcciones y cada palabra 8 bits , puesto que hay 8 salidas de datos. Por lo tanto , esta es una ROM de 16x8 . La entrada de control CS significa selección de CI.

## Operación de Lectura.

Suponga que la ROM ha sido programada con los siguientes datos:

| Palabra | A3 | A2 | A1 | Ao | D7 | D6 | D5 | D4 | D3 | D2 | D1 | Do | D7-DO |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | DE |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 3A |
| 2 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 85 |
| 3 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | AF |
| 4 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 19 |
| 5 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 7B |
| 6 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 00 |
| 7 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | ED |
| 8 | 1 | 0 | 0 | 0 | | | | . | | | | | 3C |
| 9 | 1 | 0 | 0 | 1 | | | | . | | | | | FF |
| 10 | 1 | 0 | 1 | 0 | | | | . | | | | | BC |
| 11 | 1 | 0 | 1 | 1 | | | | | | | | | C7 |
| 12 | 1 | 1 | 0 | 0 | | | | | | | | | 27 |
| 13 | 1 | 1 | 0 | 1 | | | | | | | | | 6A |
| 14 | 1 | 1 | 1 | 0 | | | | | | | | | D2 |
| 15 | 1 | 1 | 1 | 1 | | | | | | | | | 5B |

M.I. Norma Elva Chávez Rodríguez

## *Dispositivos Lógicos Programables.*

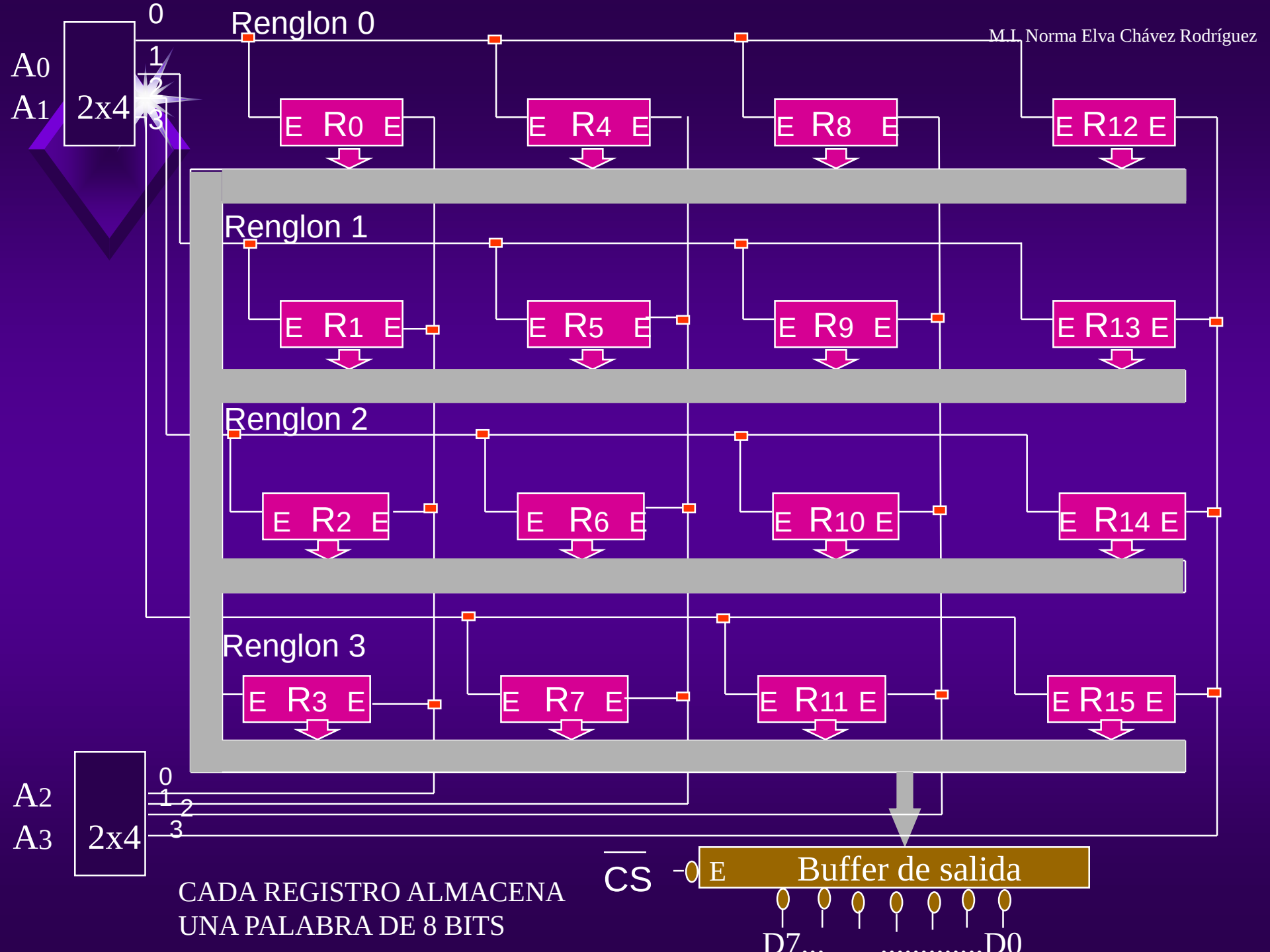
P. ejemplo: La palabra de datos almacenada con la localidad 0011 es 10101111 . Los datos que se almacenan en binario en la ROM , pero con mucha frecuencia se usa notación hexadecimal para mostrar los datos almacenados.
A fin de leer una palabra de datos de la ROM, se necesita hacer dos cosas : Aplicar las entradas de direcciones adecuadas y luego activar las entradas de control.

# *Dispositivos Lógicos Programables.*

Arquitectura de la ROM .

La arquitectura de un CI ROM es muy complicado y no necesitamos conocer todos sus detalles . Sin embargo , es constructivo observar un diagrama simplificado de la arquitectura interna para la ROM de 16x8 . Existen cuatro partes básicas : Decodificador de renglones , decodificador de columnas , disposición de registros y buffers de salida:

M.I. Norma Elva Chávez Rodríguez




CADA REGISTRO ALMACENA
UNA PALABRA DE 8 BITS

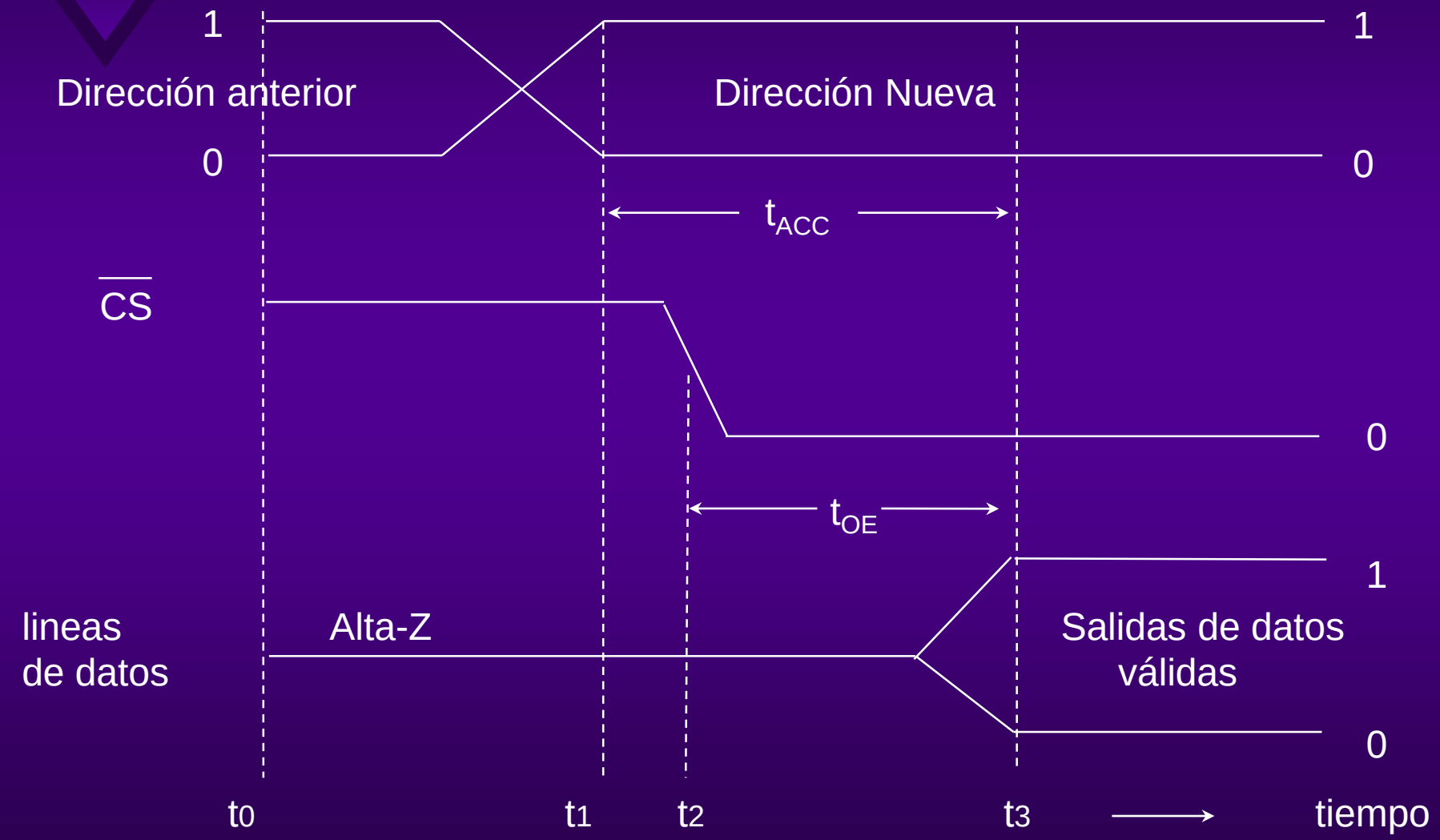
# *Dispositivos Lógicos Programables.*

Temporización de la ROM.

Tiempo de acceso (tacc): Es el retardo en la propagación entre la aplicación de entradas de una ROM y la aparición de las salidas de datos durante una operación de lectura. Es una medida de la velocidad de operación de la ROM.

# *Temporización de lectura en ROM.*

M.I. Norma Elva Chávez Rodríguez



# *Temporización de lectura en ROM.*

$t_{ACC}$

- 30 —90 nS en ROM bipolares (bJT)
- 35 —500 nS en NMOS

$t_{OE}$ = Tiempo de habilitación de salida.

$t_{OE}$

- 10 a 20 nS para ROM bipolares
- 25 a 100 nS para ROM MOS

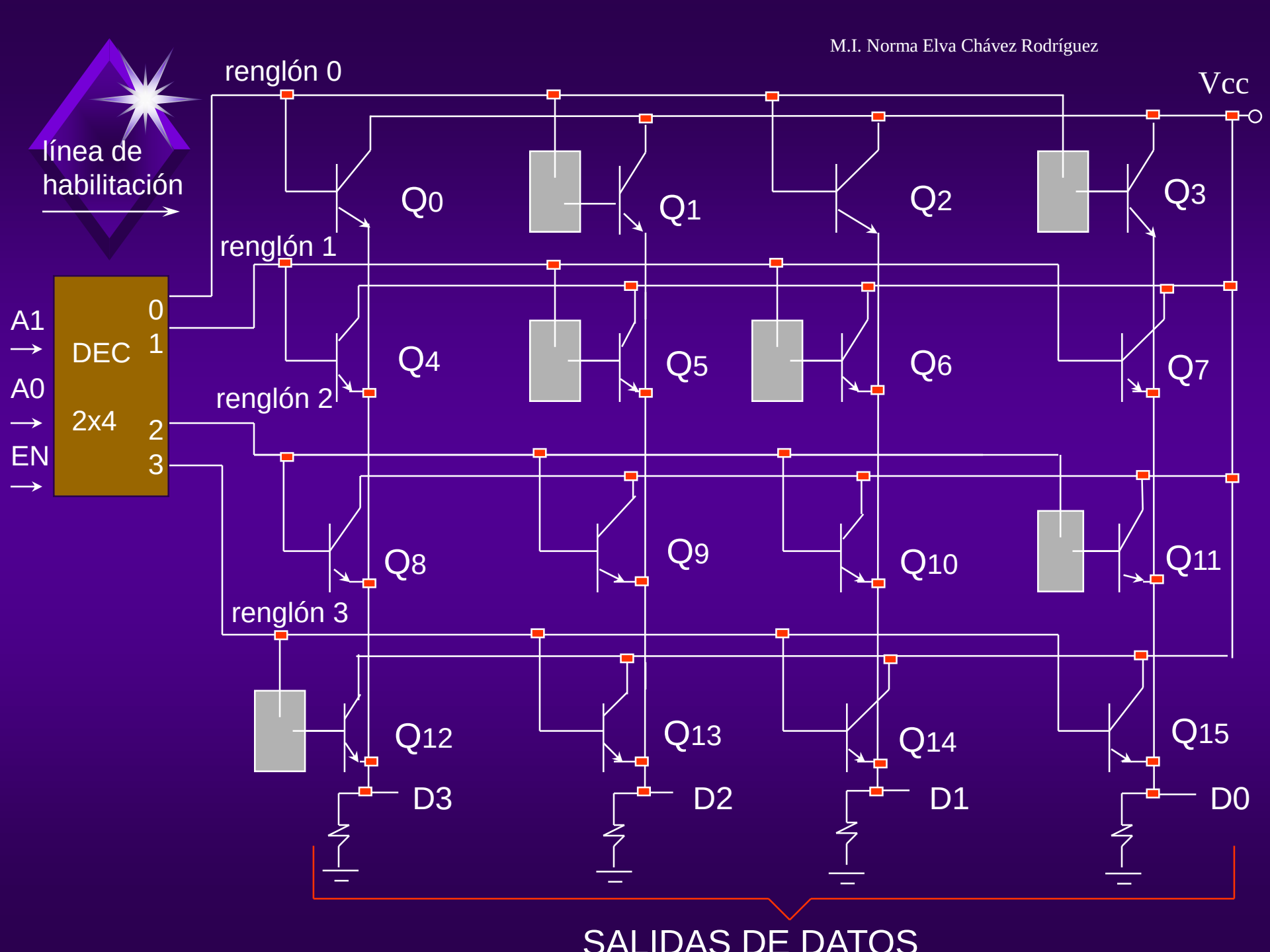
# *Dispositivos Lógicos Programables.*

Tipos de ROM

ROM programada por mascarilla : Este tipo de ROM tiene sus localidades de almacenamiento escritas (programadas) por el fabricante según las específicaciones del cliente (MROM).

Un ejemplo de una ROM bipolar se presenta en el siguiente dibujo:

M.I. Norma Elva Chávez Rodríguez



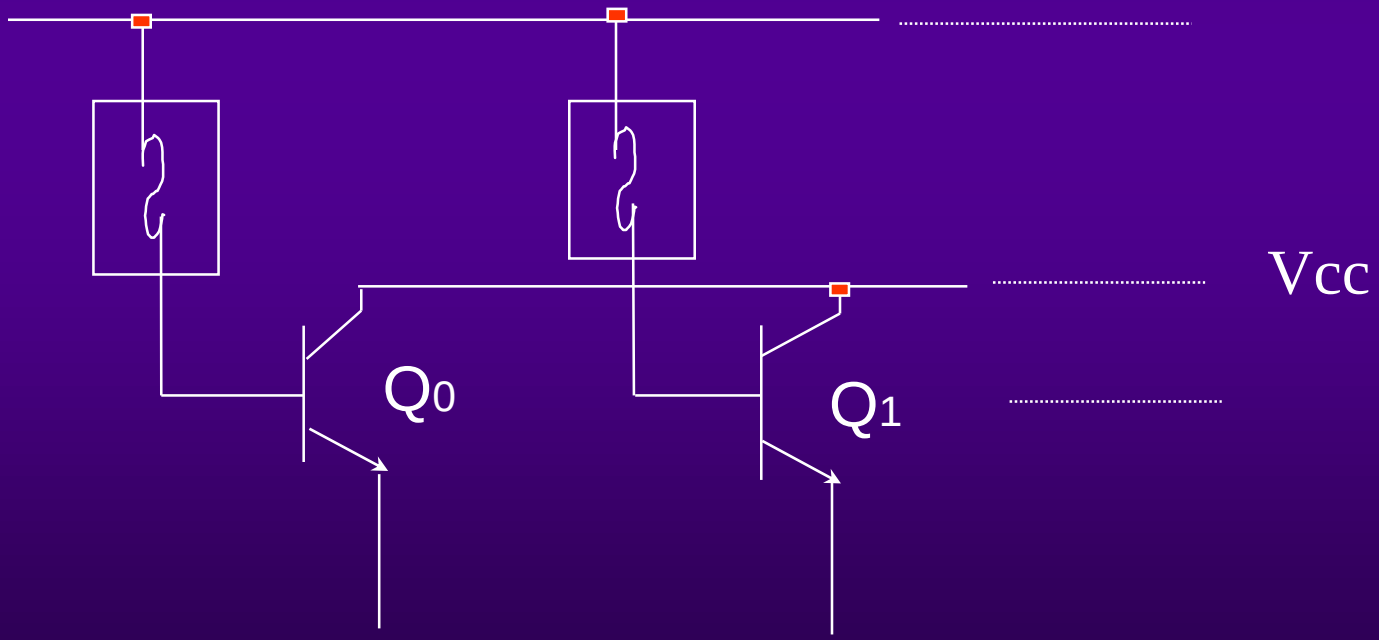
# *Dispositivos Lógicos Programables.*

| Dirección | | Dato | | | |
|---|---|---|---|---|---|
| $A_1$ | $A_0$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
| 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 |

Las MROM se encuentran disponibles en varias capacidades, todas ellas pequeñas. Una de las más populares es la 74187, es una memoria de 256x4 con un $t_{acc}$ = 40ns con salidas de colector abierto. Otra ROM es la 7488A con capacidad de 32x8 y $t_{acc}$ = 45ns.

# *Dispositivos Lógicos Programables.*

ROM Programables (PROM). Este tipo de ROM se han creado con conexión fusible de forma tal que pueden ser programadas por el usuario, solo que pueden ser programadas una sola vez igual que las MROM.



# *Dispositivos Lógicos Programables.*

PROM por ej. 74186 de 64x8 con $t_{acc}$ = 50ns TBP28S166 de 2Kx8 PROM MOS por ej.: TMS27PC256 de 32Kx8 con $t_{acc}$ = 120 a 250ns.

ROM programable y borrable (EPROM). Este tipo de ROM puede ser programada y borrada por el usuario tantas veces como quiera. Una vez programada, la EPROM es una memoria no volátil.

EPROM 2732 de 4Kx8 $t_{acc}$ = 45ns.

PROM eléctricamente borrable (EEPROM): 2864 de 8Kx8 $t_{acc}$ = 250ns

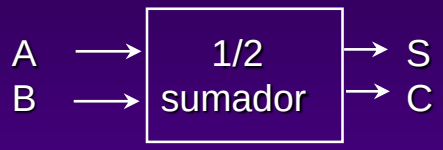
# *Dispositivos Lógicos Programables.*

## Aplicaciones de las ROM.

- Almacenamiento de programas en microcomputadora programación en firme (firrnware).
- Memoria de arranque.
- Tablas de datos.
- Convertidor de datos.
- Generador de caracteres.
- Generador de funciones.

# Sumadores y Restadores.

## 1/2 SUMADOR
## A+B



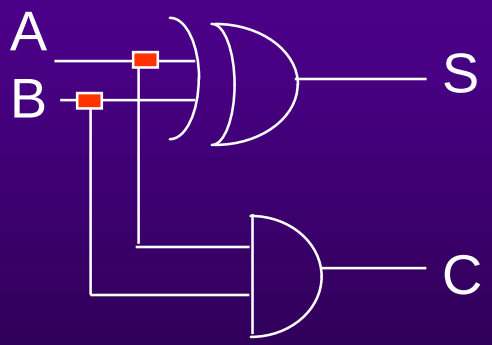

| A | B | S | C |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

$$S = \bar{A}B + A\bar{B}$$
$$S = A \oplus B$$

$$C = AB$$

| S | B=0 | B=1 |
|---|---|---|
| A=0 | 0 | 1 |
| A=1 | 1 | 0 |

| C | B=0 | B=1 |
|---|---|---|
| A=0 | 0 | 0 |
| A=1 | 0 | 1 |



# Sumadores y Restadores.

## 1/2 RESTADOR
## A-B

| A | B | D | R |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 |

D

| A \ B | 0 | 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 0 | 0 |

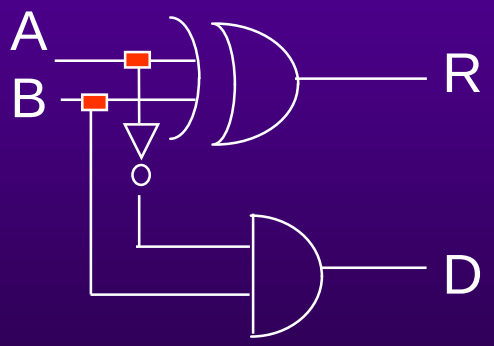
R

| A \ B | 0 | 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 1 | 0 |

$D = \bar{A}B$

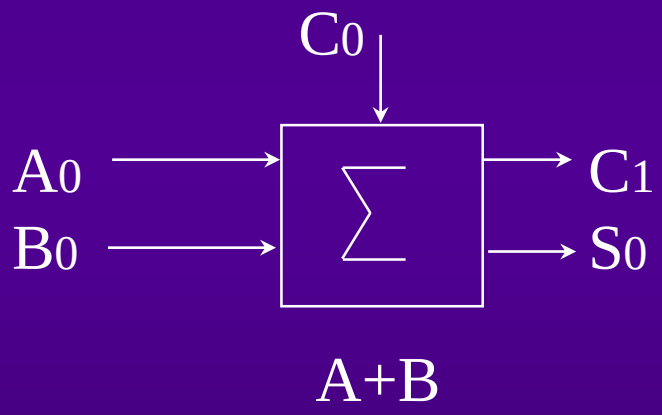
$R = \bar{A}B + A\bar{B}$

$R = A \oplus B$



# Sumadores y Restadores.

## SUMADOR COMPLETO




| | $A_2$ | $A_1$ | $A_0$ |
|---|---|---|---|
| A = | 1 | 0 | 1 |
| | $B_2$ | $B_1$ | $B_0$ |
| B = | 0 | 1 | 0 |
| | $S_2$ | $S_1$ | $S_0$ |

$C_3$ $C_2$ $C_1$ $C_0$

# Sumadores y Restadores.

| $A_0$ | $B_0$ | $C_0$ | $S_0$ | $C_1$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

| $S_0$ | $B_0C_0$ 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| $A_0$ 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |

$$S_0=\bar{A}_0\bar{B}_0C_0+\bar{A}_0B_0\bar{C}_0+A_0\bar{B}_0\bar{C}_0+A_0B_0C_0$$

$$=\bar{A}_0(\bar{B}_0C_0+B_0\bar{C}_0)+A_0(\bar{B}_0\bar{C}_0+B_0C_0)$$

$$=\bar{A}_0(B_0\oplus C_0)+A_0(\bar{B}_0\bar{C}_0+B_0C_0)$$

$K$ $\quad$ $\bar{K}$

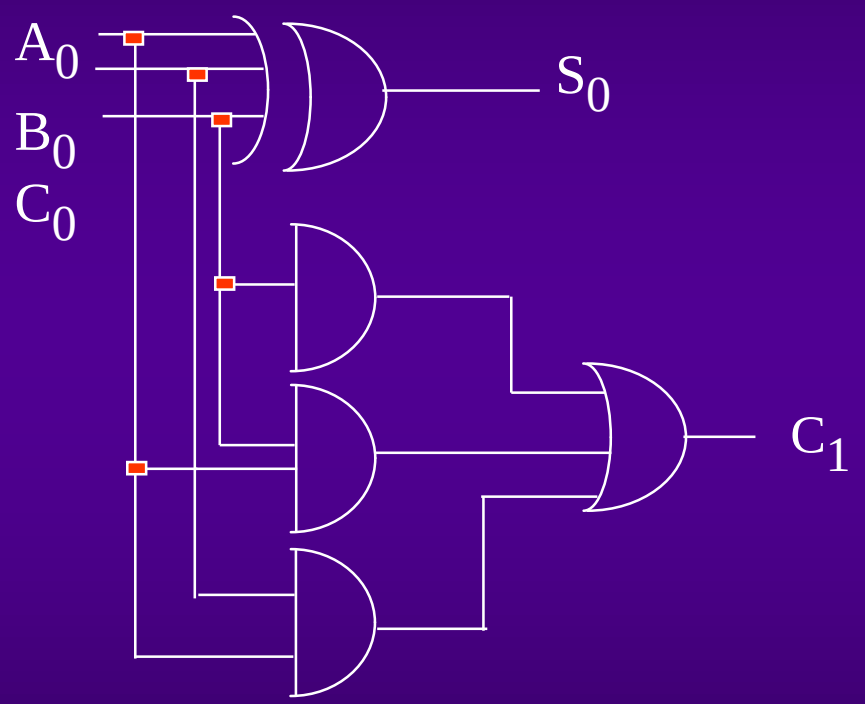
# Sumadores y Restadores.

$= A_0\bar{K} + \bar{A_0}K$

$= A_0 \oplus K = A_0 \oplus (B_0 + C_0)$

$= A_0 \oplus B_0 \oplus C_0$

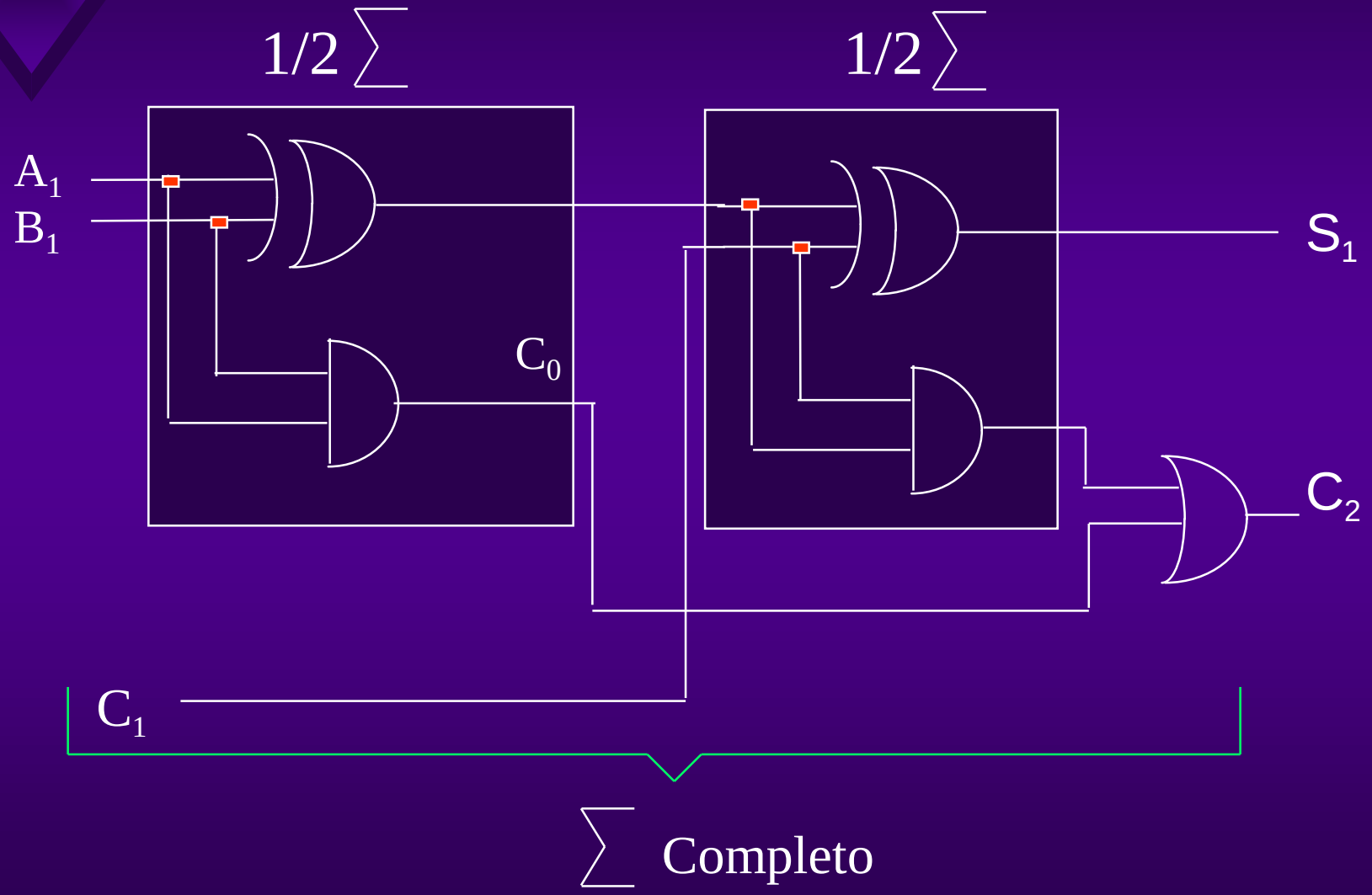
| $A_0$ \ $C_1$ \ $B_0C_0$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 |



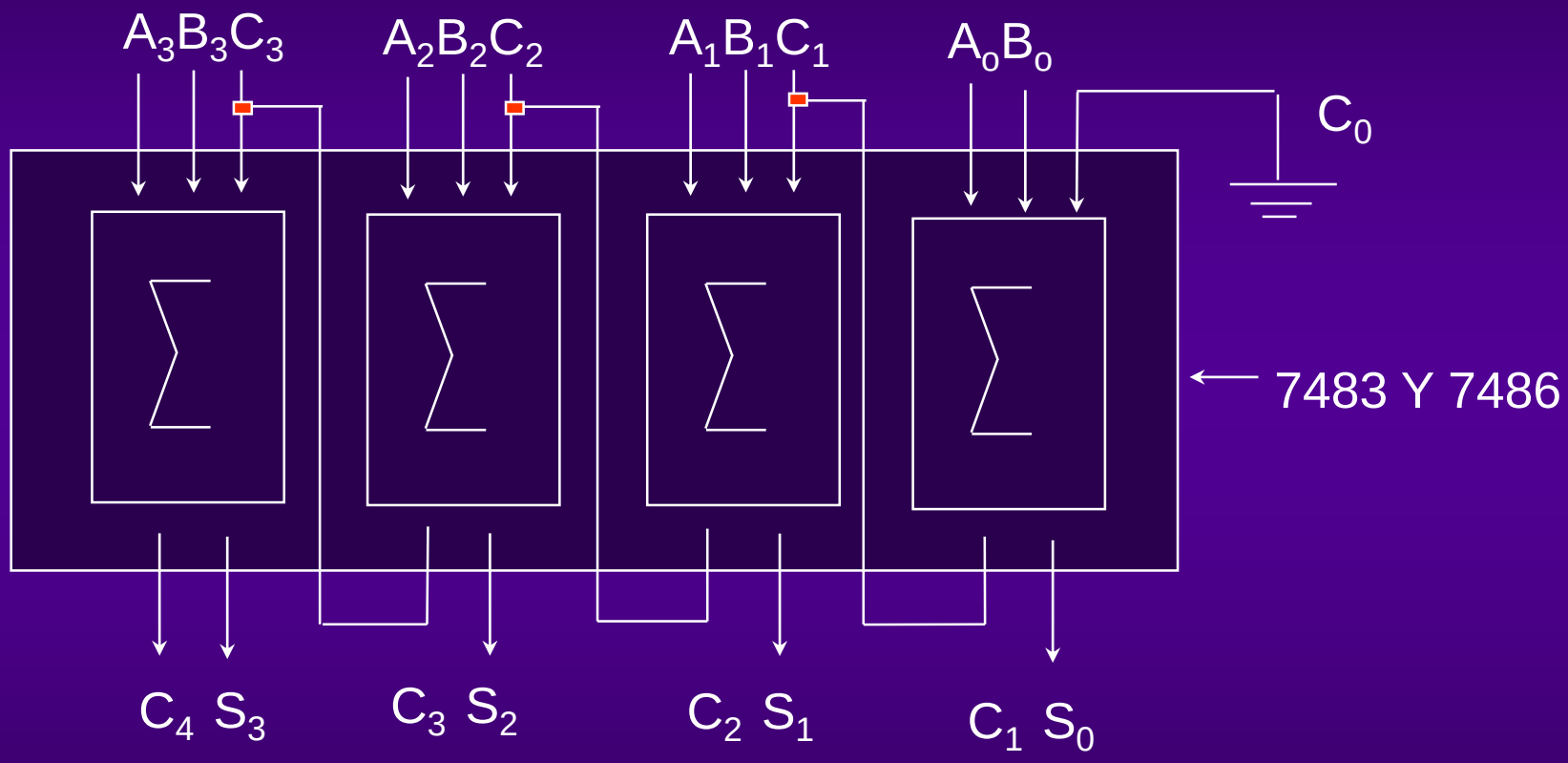

$C_1 = B_0C_0 + A_0C_0 + A_0B_0$

$C_1 = B_0C_0 + A_0 (C_0 + B_0)$

# Sumadores y Restadores.



# Sumadores y Restadores.




A = 1 0 1 0

B = 1 0 1 1

# Sumadores y Restadores.

## Restador Completo

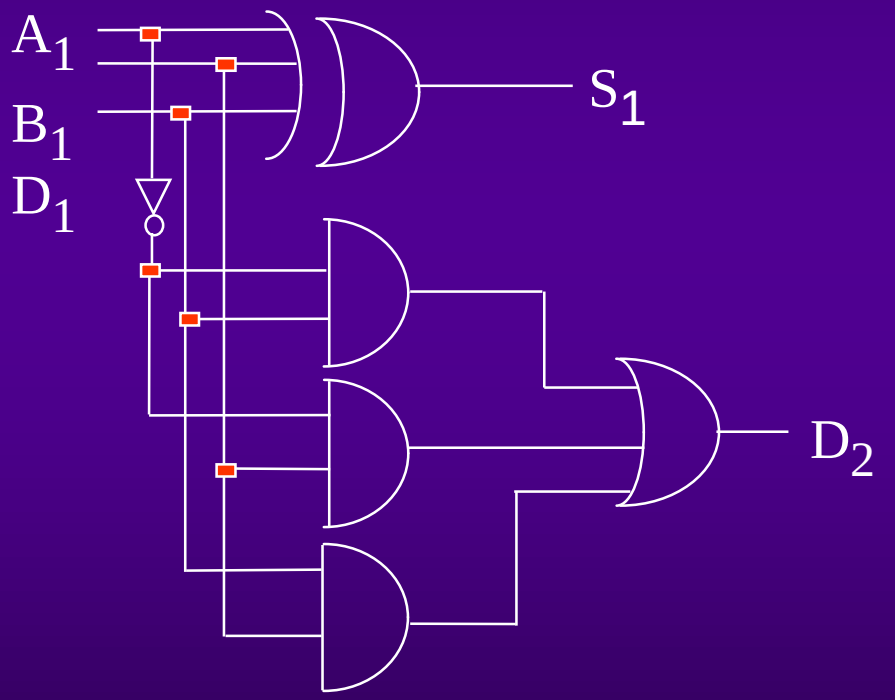
| $A_0$ | $B_0$ | $D_0$ | $D_1$ | $S_0$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

| $S_1$ | $B_1D_1$ 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| $A_1$ 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |

$$S_0 = A_1 \oplus B_1 \oplus D_1$$

# Sumadores y Restadores.

| $D_0$ / $B_0D_0$ / $A_0$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 |

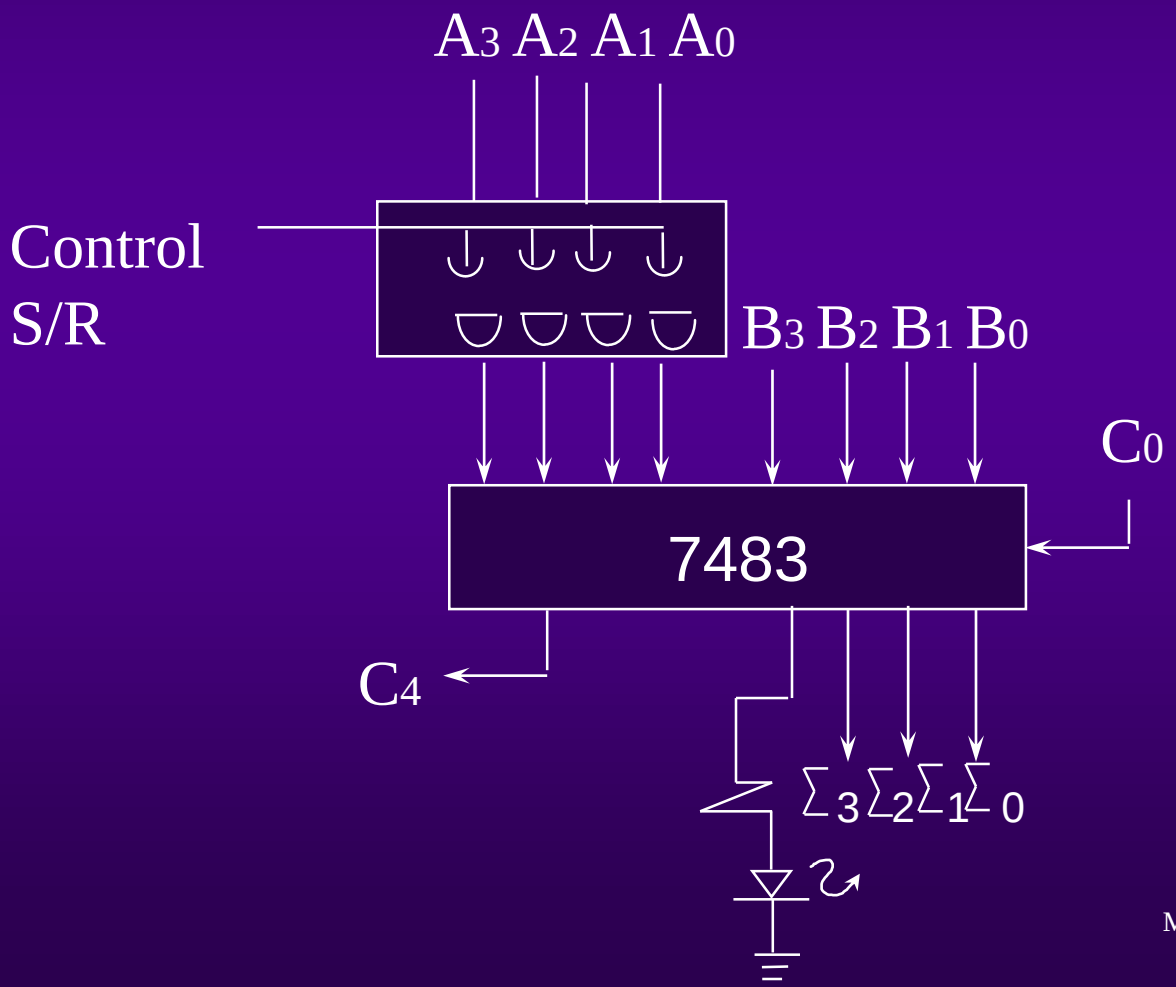



$$D_2 = A'_1D_1 + A'_1B_1 + B_1D_1$$

$$D_2 = A'_1(D_1 + B_1) + B_1D_1$$

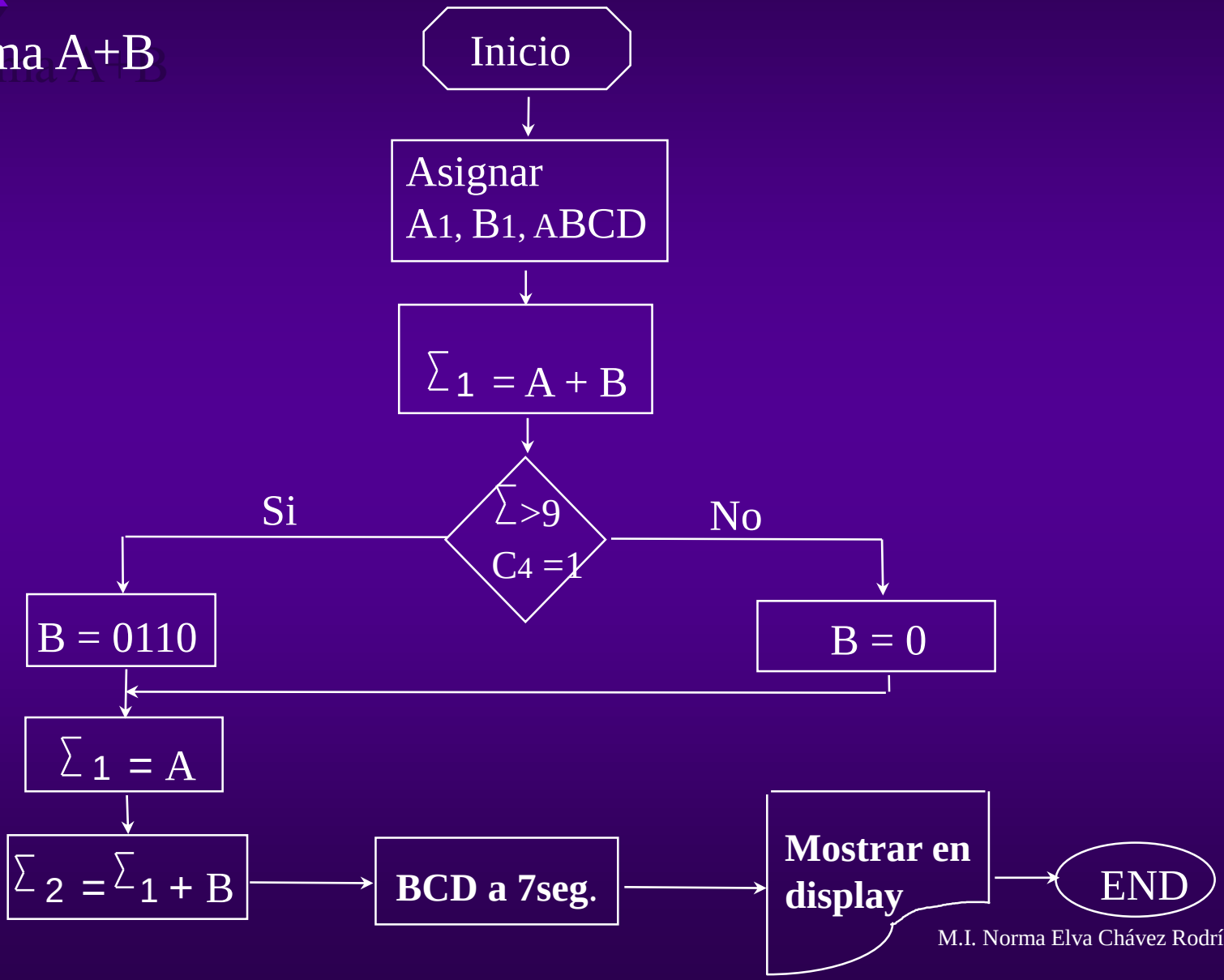
# Sumadores y Restadores.

El 7483 es un sumador de 4 bits y se puede implementar para restar un número de la siguiente forma:



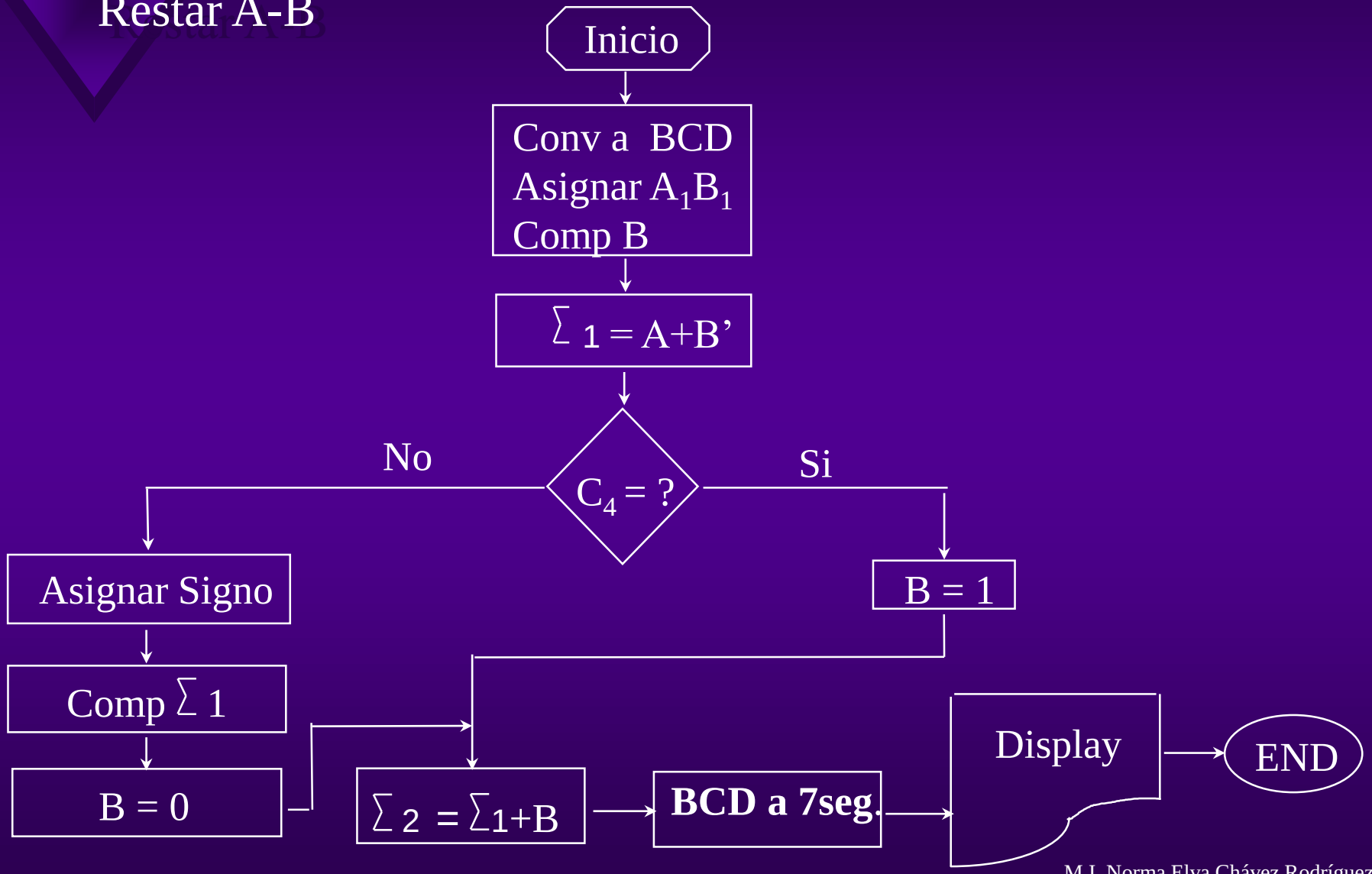
# Algoritmo para realizar la suma de dos variables de 4 bits en código BCD, utilizando sumadores completos:

Suma A+B



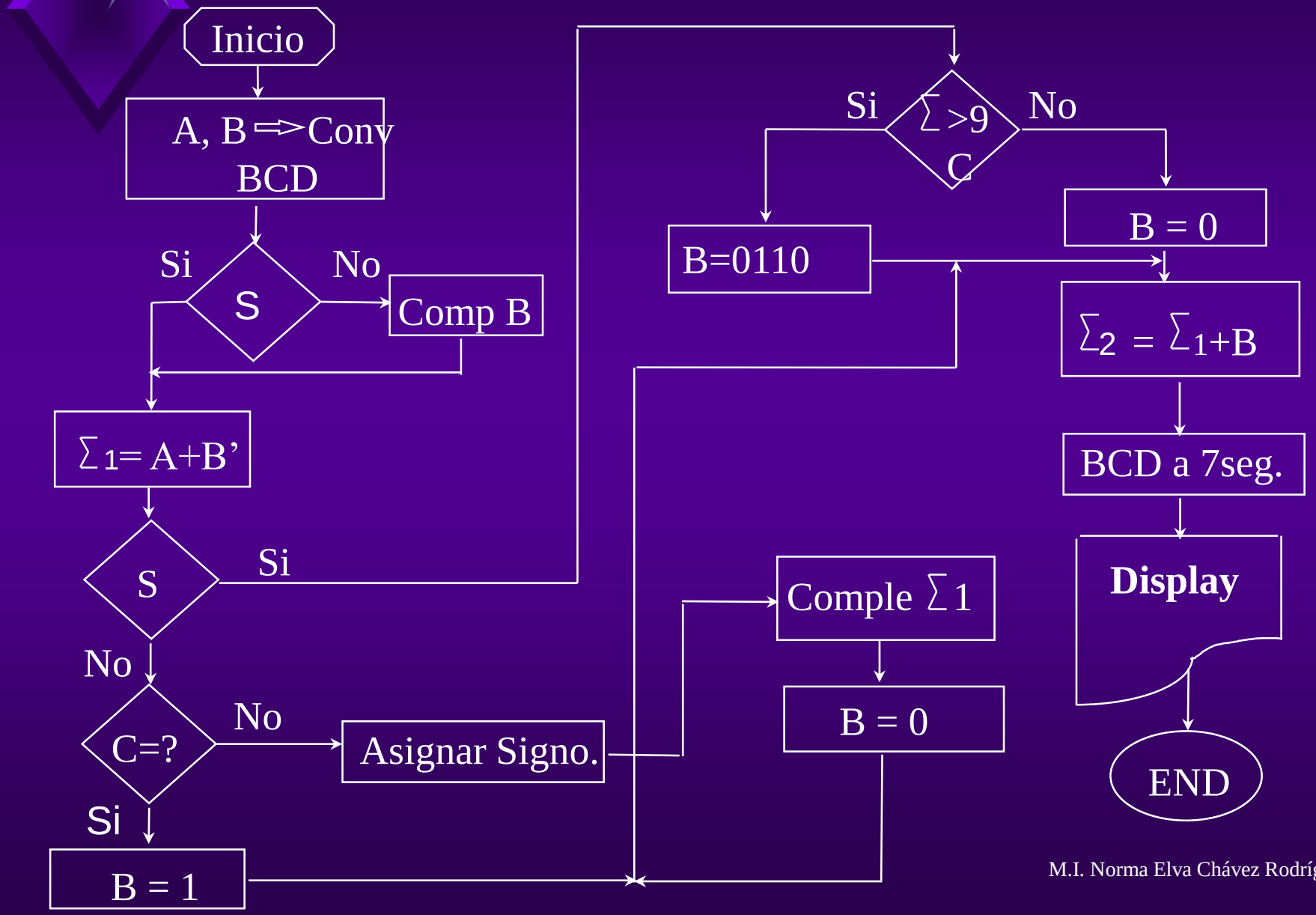
# Algoritmo para realizar la resta de dos variables de 4 bits en código BCD , utilizando sumadores completos

Restar A-B




M.I. Norma Elva Chávez Rodríguez

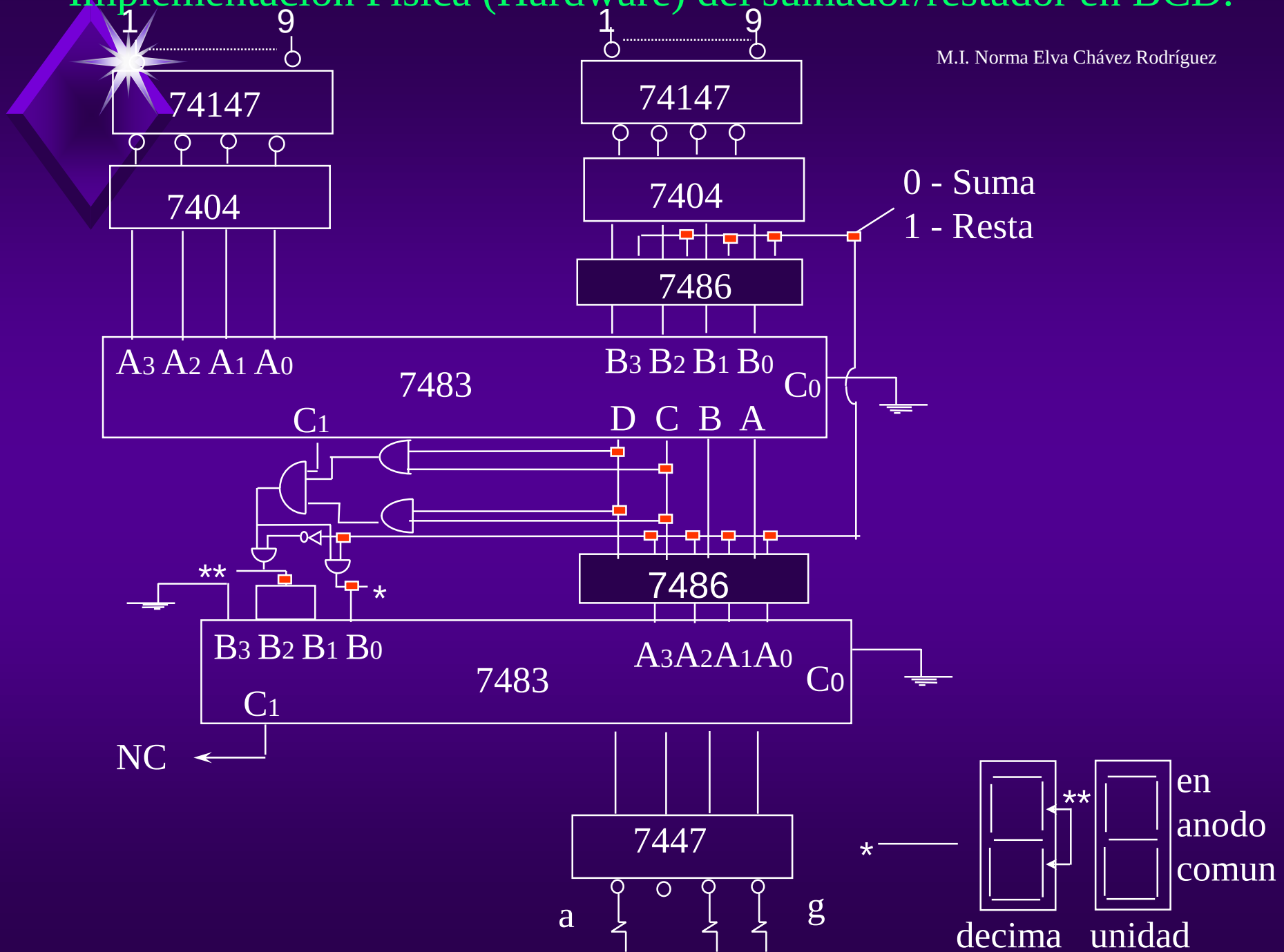
# Algoritmo para realizar la suma o resta de dos variables de 4 bits utilizando codigo BCD.




M.I. Norma Elva Chávez Rodríguez

# Implementación Física (Hardware) del sumador/restador en BCD.

M.I. Norma Elva Chávez Rodríguez



# *Lógica secuencial.*

Cto. combinacionales: Son aquellos en el que las salidas en un instante dado de tiempo son enteramente dependientes de las entradas presentes en ese mismo tiempo.

La mayoría de los sistemas digitales contienen lógica combinacional e incluyen también elementos de memoria, los cuales requieren que el sistema se describa en términos de lógica secuencial.

# *Lógica secuencial.*

Los elementos de memoria son capaces de almacenar información binaria dentro de ellos. La información binaria almacenada en un tiempo dado define el estado del cto. secuencial.

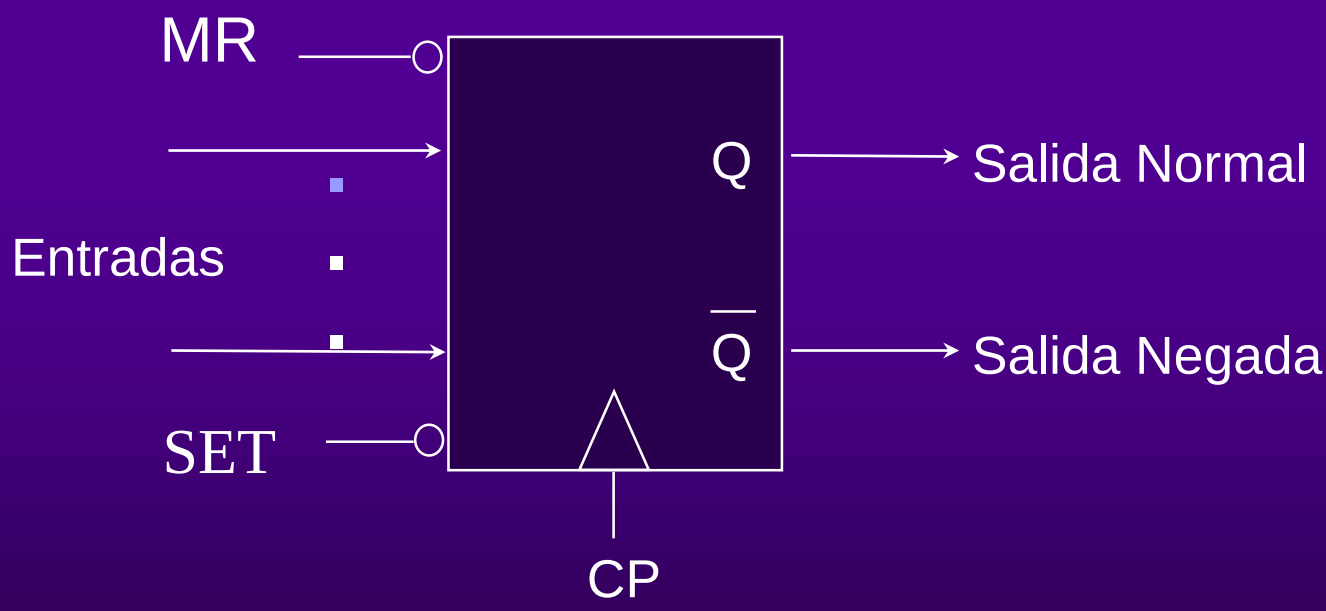
hay 2 tipos de cto. secuenciales: Su clasificación depende del tiempo de sus señales:

1. Cto. sec. sincrono: Es un sistema cuyo comportamiento puede definirse a partir del conocimiento de sus señales en instantes discretos de tiempo.
2. Cto. sec. asincrono: Su comportamiento depende del orden en que cambien las señales de entrada y pueden ser afectadas en un instante dado de tiempo.

# *Lógica secuencial.*

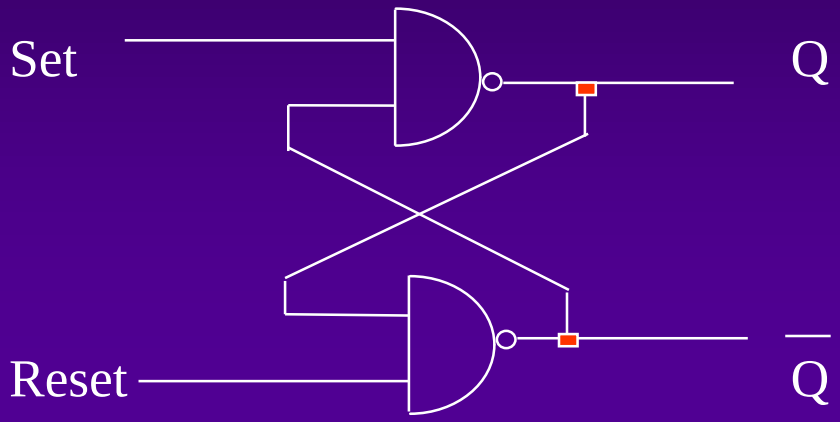
El elemento de memoria más importante es el flip-flop, que está formado por un ensamble de compuertas lógicas.

El símbolo general para el flip-flop es el siguiente:



# *Lógica secuencial.*

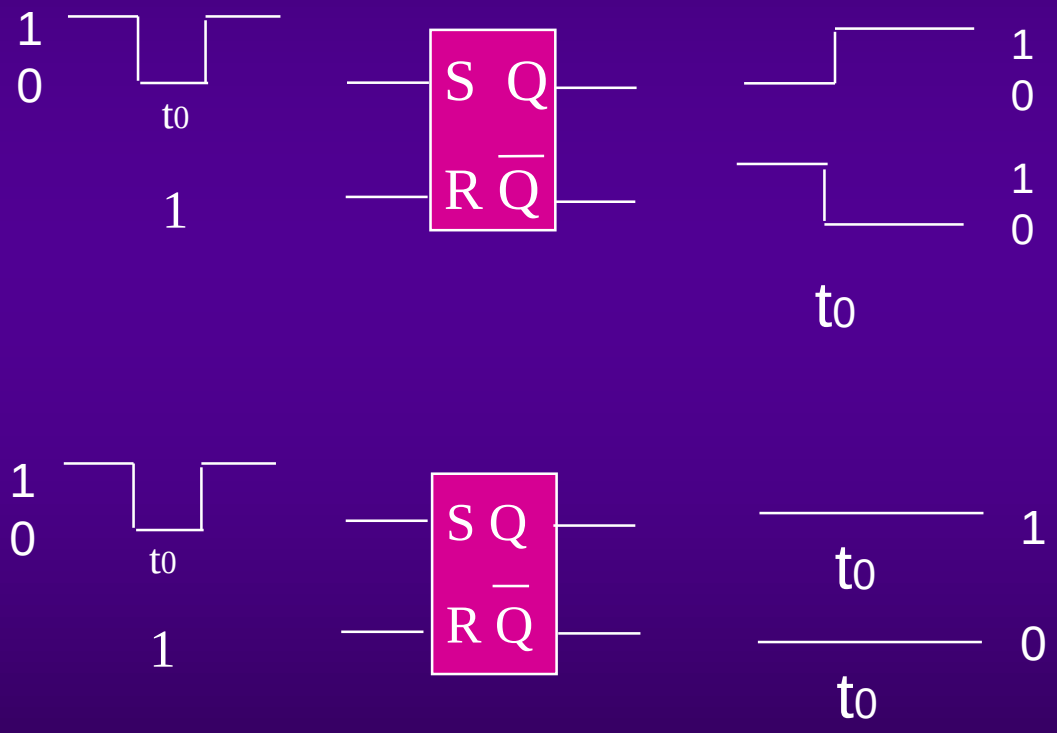
Flip-Flop básico construido con compuertas NAND




Paso 1: Si consideramos primero que tenemos Q = 1 y $\overline{Q}$ = 0 y ponemos Set = Reset = 1 no existe cambio alguno en las salidas.
De la misma forma si consideramos Q = 0 y $\overline{Q}$ = 1 teniendo como entradas Set = Reset = 1, no existe cambio alguno ni en Q ni en $\overline{Q}$.

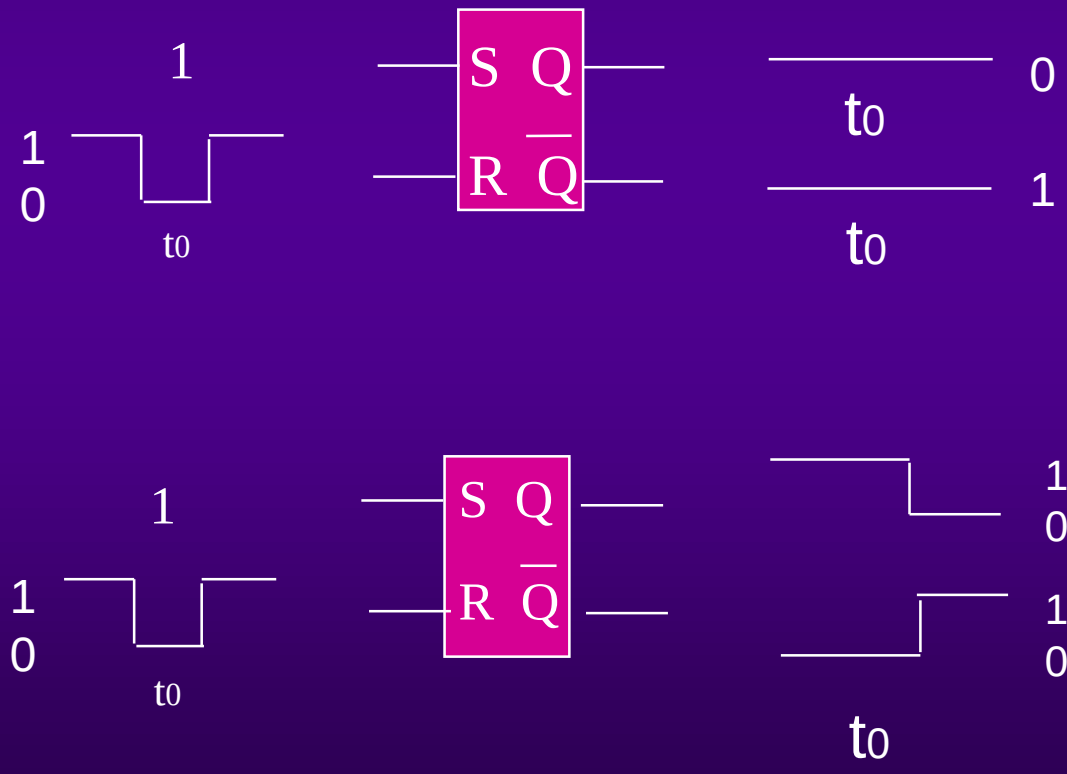
# *Lógica secuencial.*

Paso 2: Si ahora cambiamos en un instante de tiempo Set a cero que sucede:



# *Lógica secuencial.*

Paso 3: Borrado del registro básico. Si ahora cambiamos en un instante de tiempo reset a bajo estando a 1 set que sucede:



# *Lógica secuencial.*

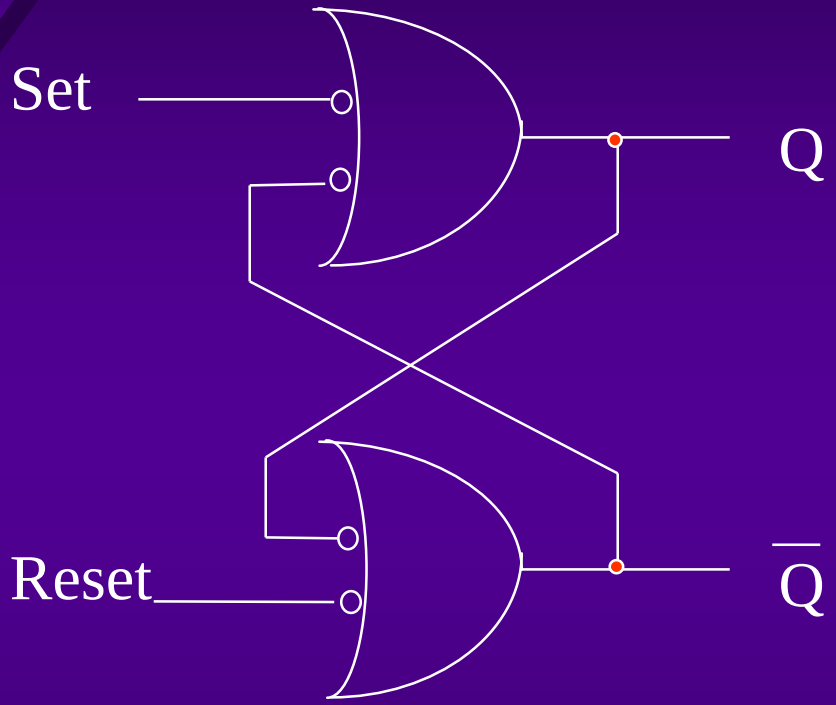
Paso 4: La última condición que falta probar es cuando set = reset = 0. Cuando se envían a cero en forma simultánea Q = Q’ = 1. Con toda claridad se observa que es una condición no deseada, ya que las salidas se suponen una la inversa de la otra. Además cuando las entradas retornen al estado alto, la salida dependerá de cual entrada cambio primero a alto. Ya que transiciones simultáneas a 1 producirán resultados impredecibles. Por lo tanto el registro básico con NAND no utiliza esta condición.

# *Lógica secuencial.*

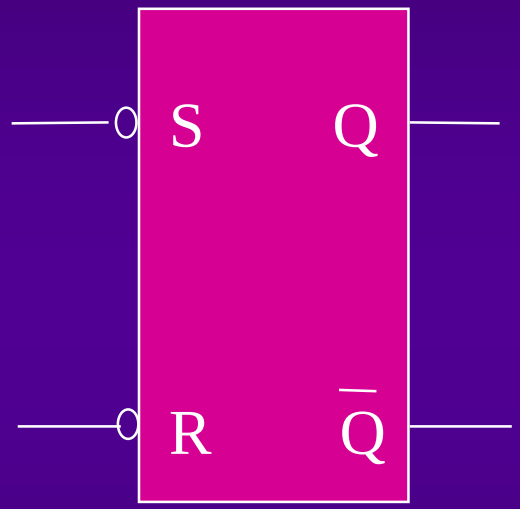
Por lo tanto la tabla de verdad de este registro podemos resumirlo como:

| Set | Reset | Salida |
|---|---|---|
| 1 | 1 | no hay cambio |
| 0 | 1 | Q = 1 y Q’ = 0 |
| 1 | 0 | Q = 0 y Q’ = 1 |
| 0 | 0 | invalido (produce Q = Q’ = 1) |

# Lógica secuencial.



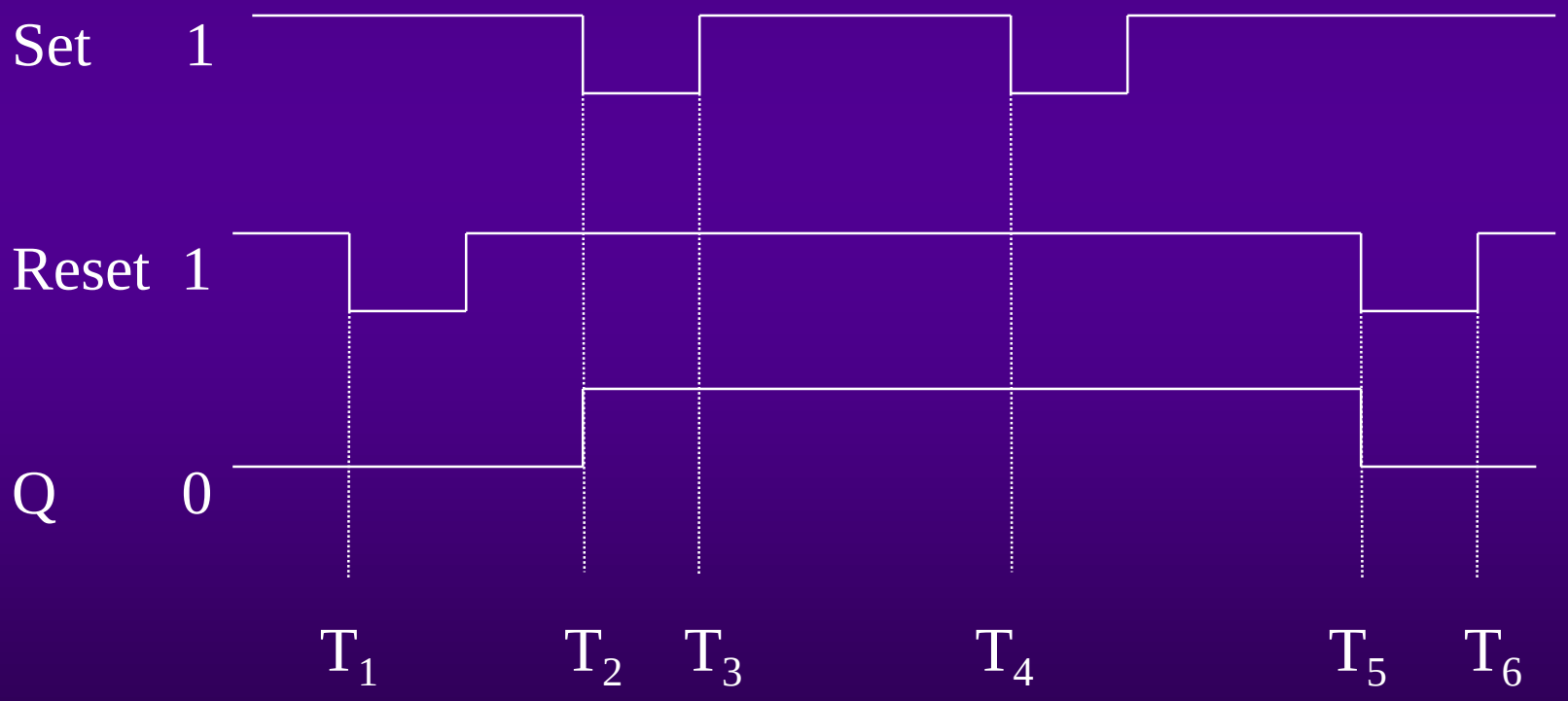

Registro Básico Nand
representación equivalente




Simbolo simplificado

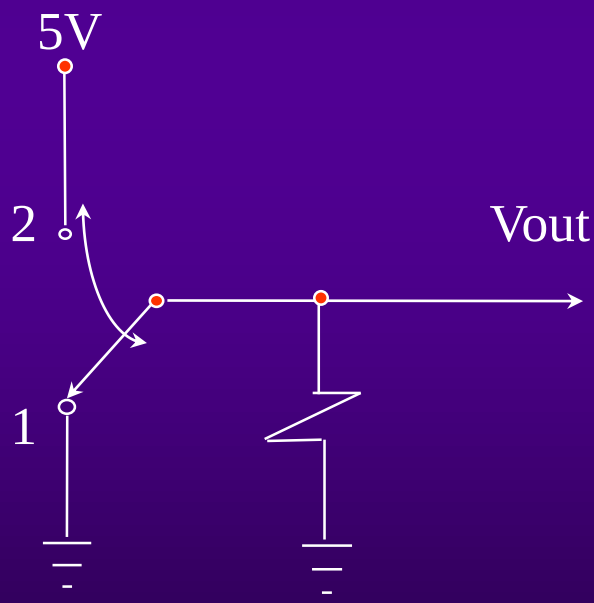
# Lógica secuencial.

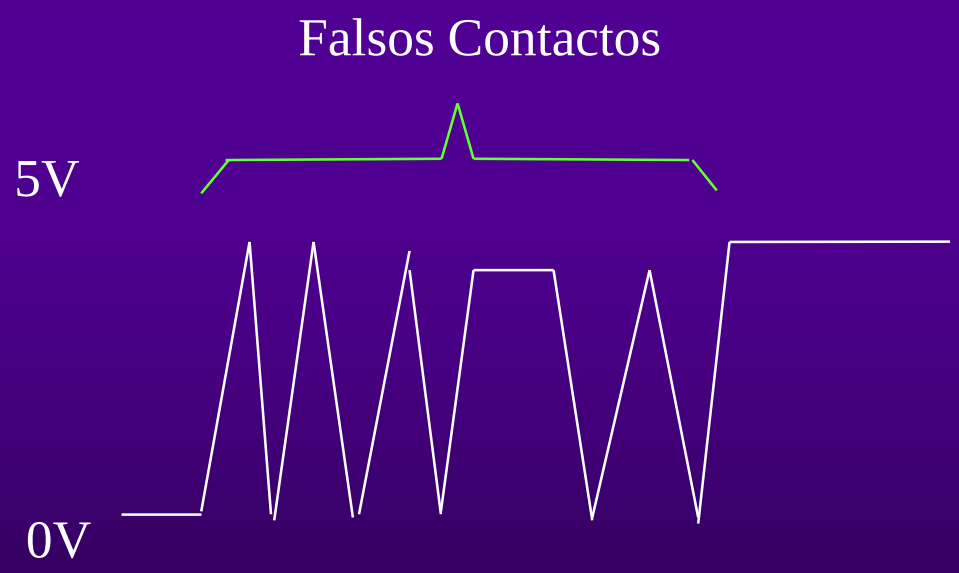
Ejemplo: Suponga que se aplican las siguientes formas de onda a un flip-flop básico con compuertas NAND . Suponga que inicialmente Q = 0 , determinar la forma de onda de Q.



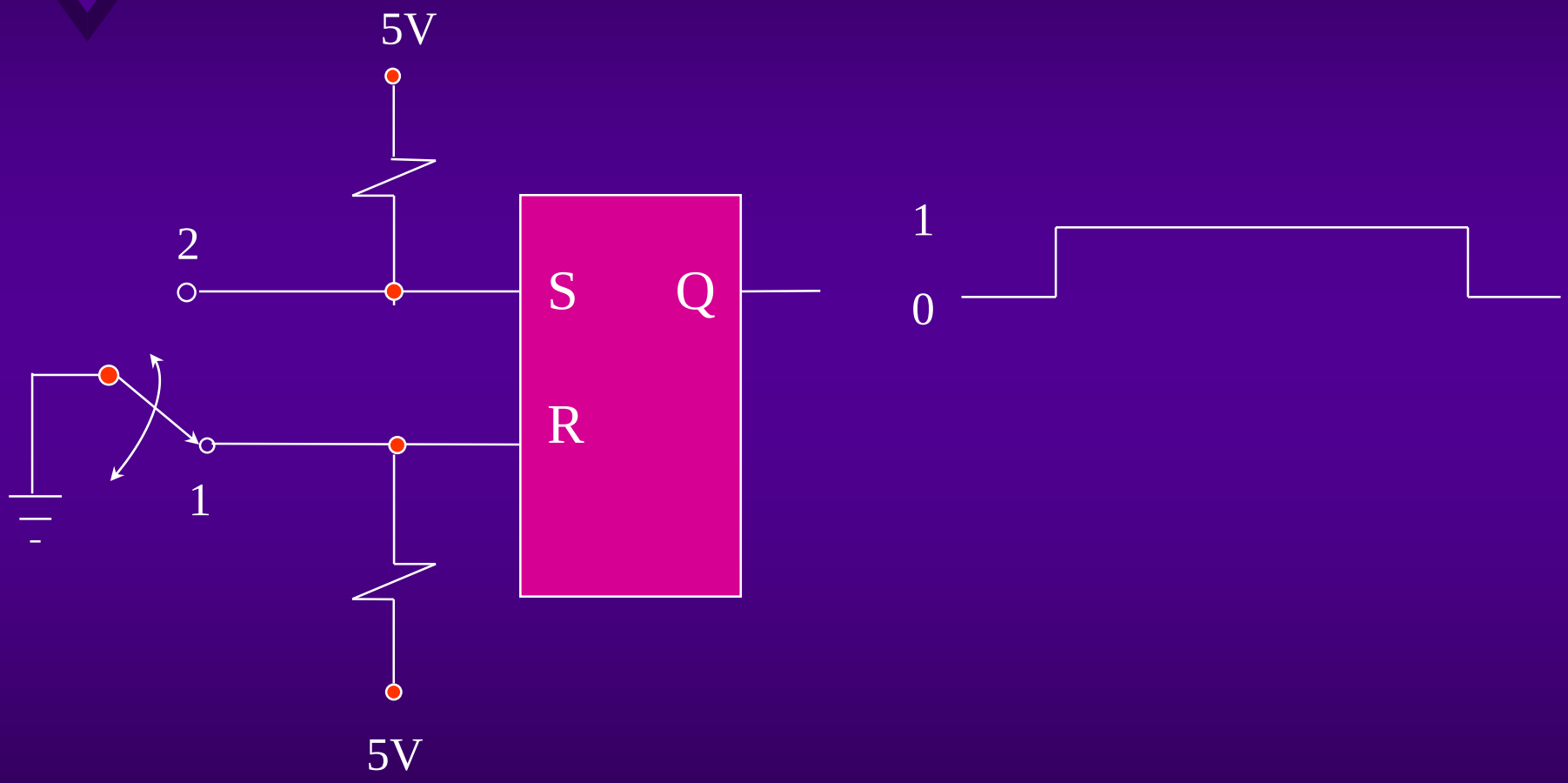
# Lógica secuencial.

Ejemplo: En un interruptor de contacto es casi imposible obtener una transición de voltaje “limpia” debido al fénomeno de oscilación (“rebote”) de contacto.



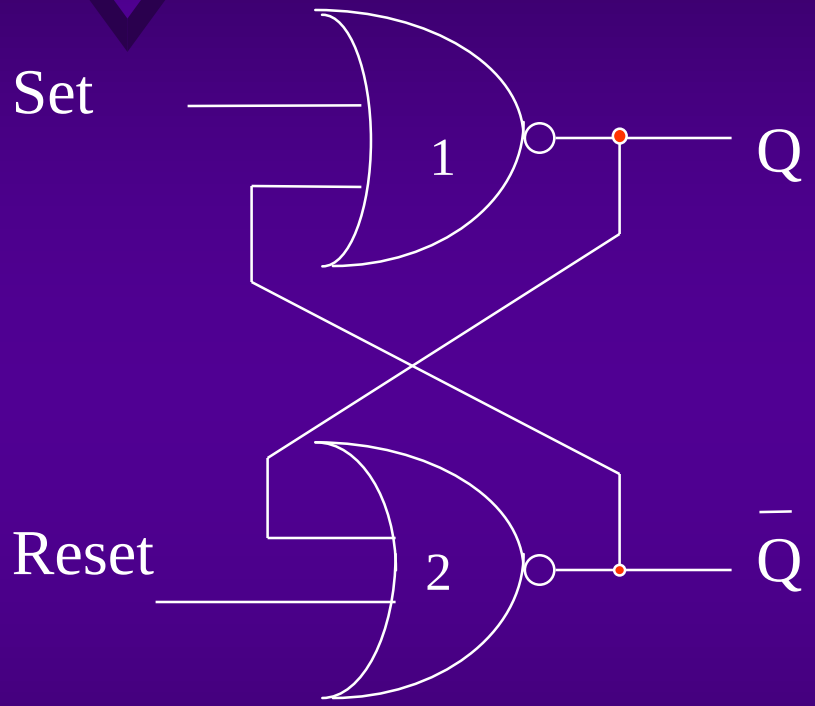
# Lógica secuencial.




M.I. Norma Elva Chávez Rodríguez

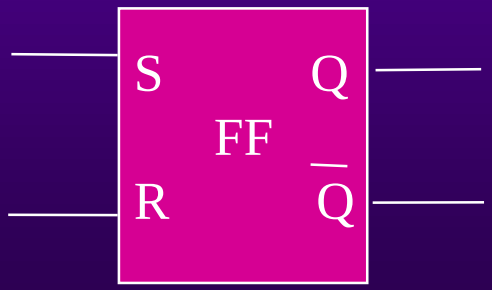
# Lógica secuencial.

## Registro básico con compertas NOR



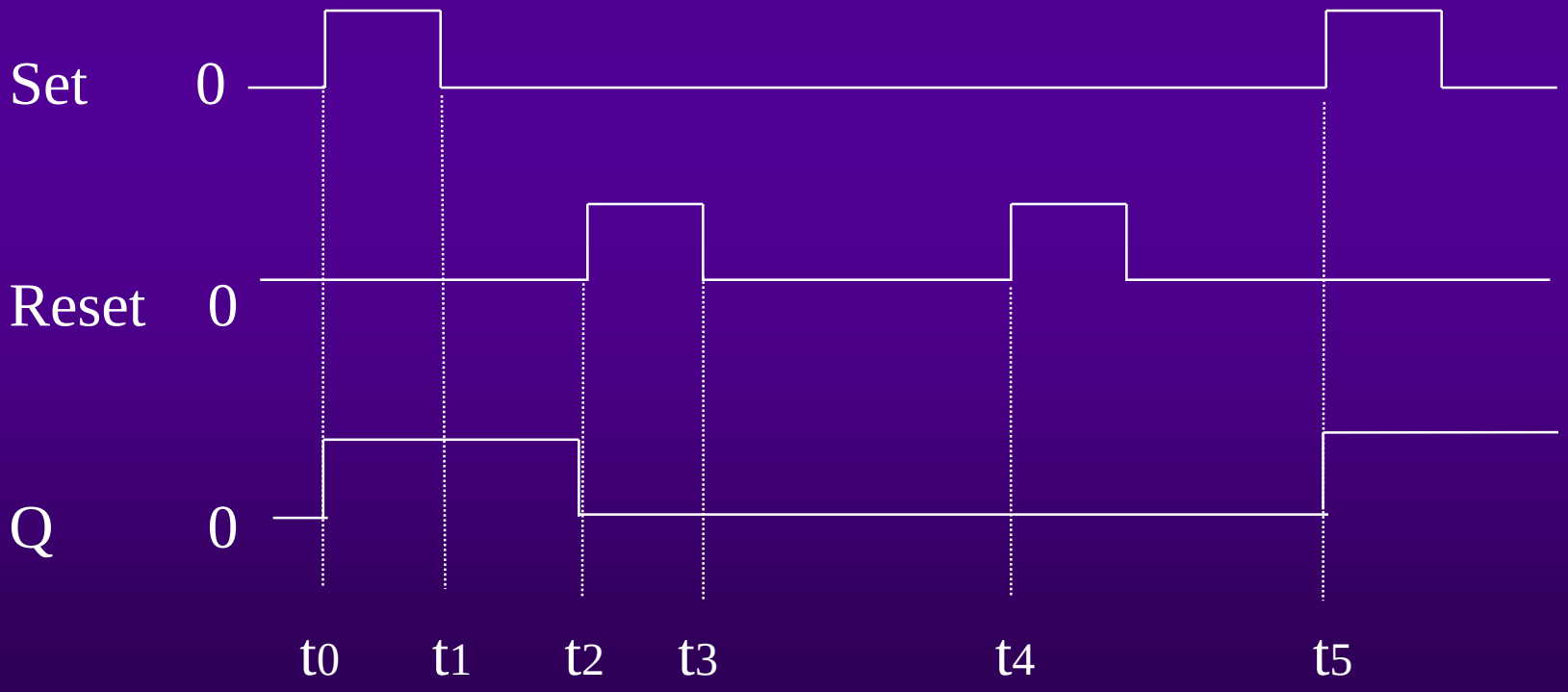

| RESET | SET | SALIDA |
|---|---|---|
| 0 | 0 | No hay cambio |
| 1 | 0 | Q = 1 |
| 0 | 1 | Q = 0 |
| 1 | 1 | inválido * |

*produce Q = $\overline{Q}$ = 0



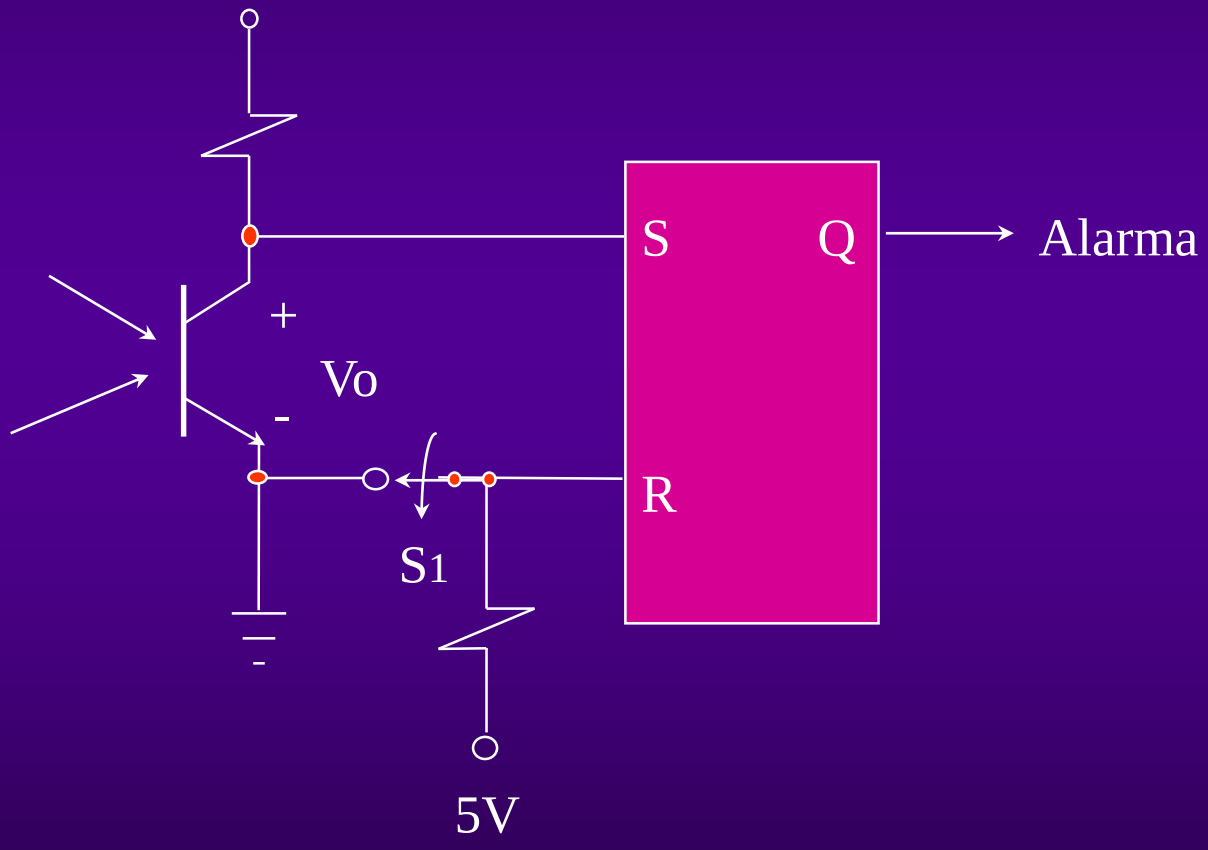
# Lógica secuencial.

Ejemplo: Suponga que inicialmente Q = 0 y determine la formas de onda de Q para las entradas del registro básico NOR.

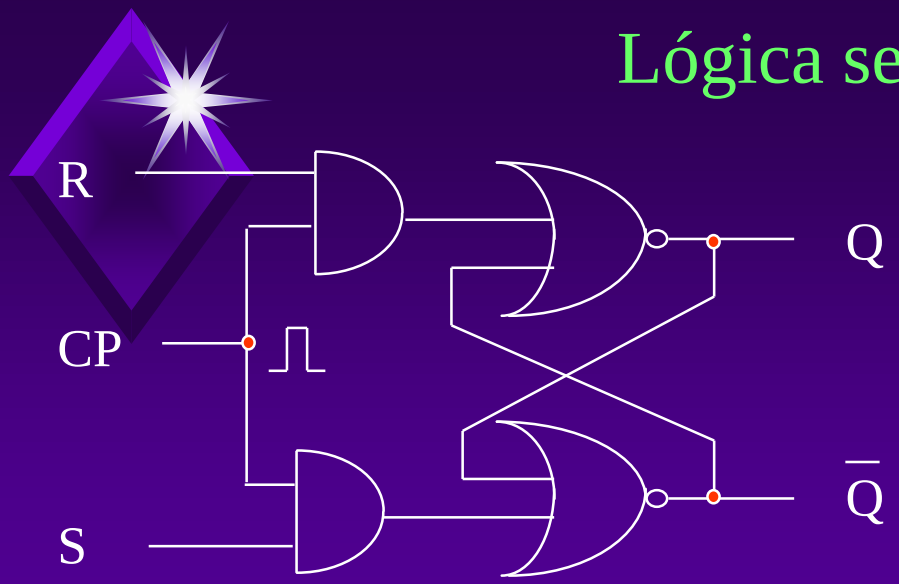


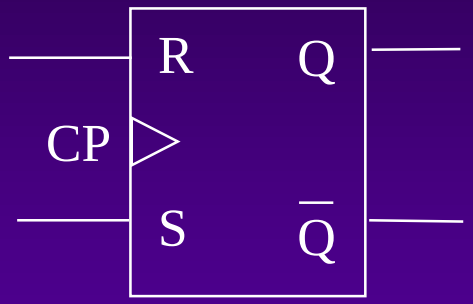
# Lógica secuencial.

Ejemplo:

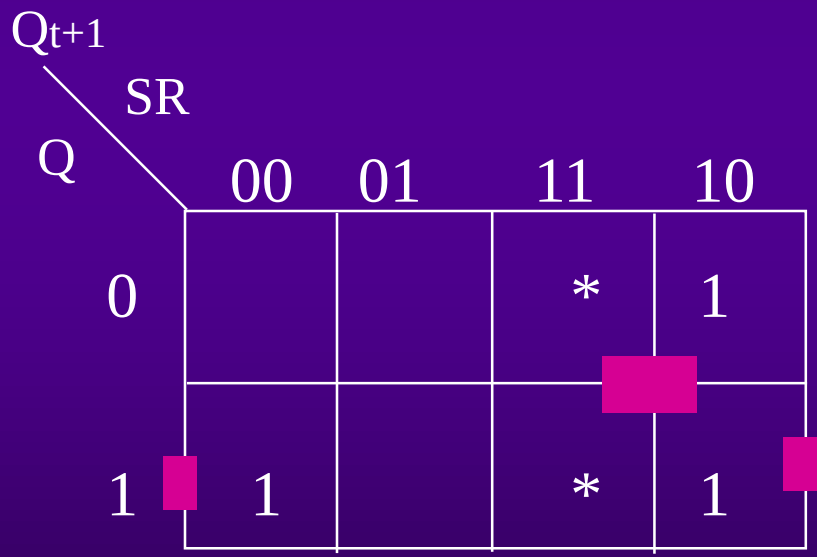


# Lógica secuencial.




| Q | S | R | $Q_{t+1}$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | Indeterminado |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | Indeterminado |




$Q_{t+1} = S + R\bar{Q}$

M.I. Norma Elva Chávez Rodríguez

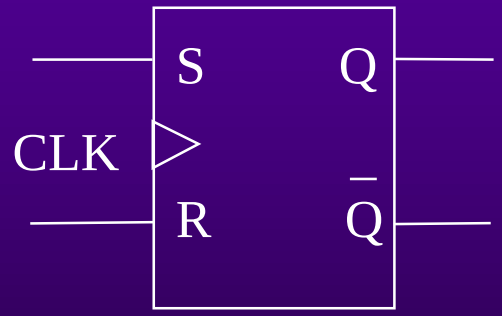
# Lógica secuencial.

Tabla característica del F.F. SR

| S | R | Q(t+1) |
|---|---|---|
| 0 | 0 | Qt |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | Indeterminado |

Tabla de excitación del F.F. SR

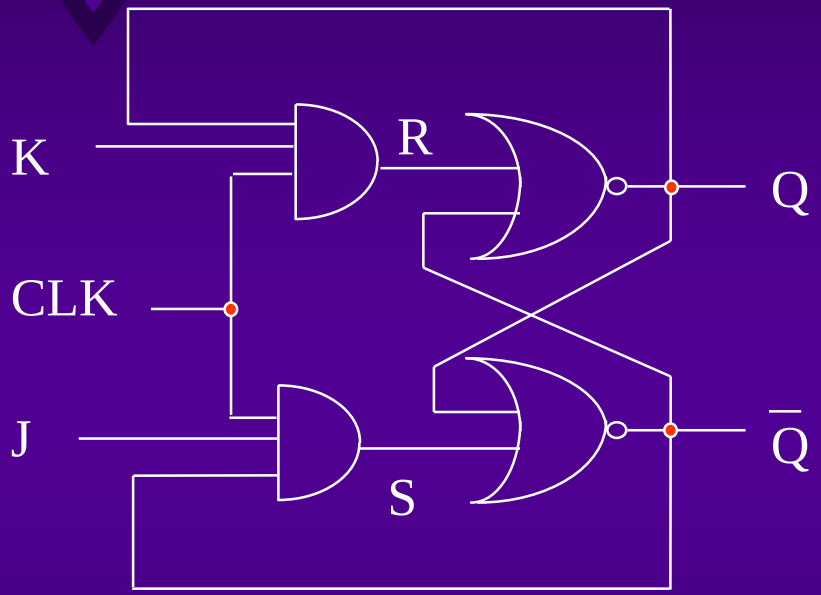
| Qt | Qt+1 | S | R |
|---|---|---|---|
| 0 | 0 | 0 | * |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | * | 0 |




Símbolo

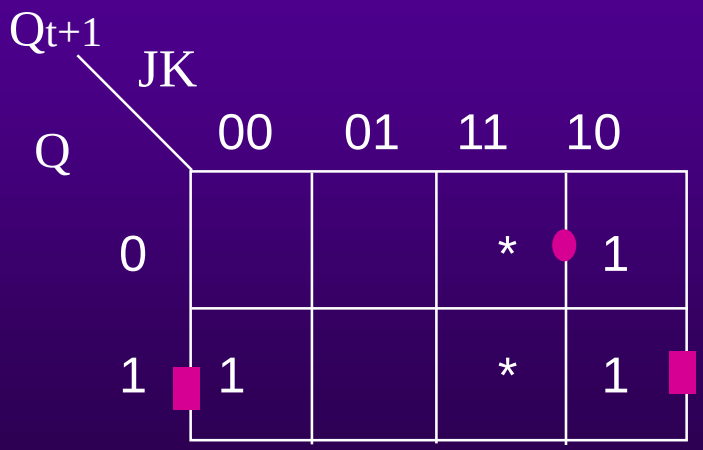
# Lógica secuencial.

Flip-Flop JK
Diagrama interno




| Q | J | K | Qt+1 |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

$$Q_{t+1} = J\overline{Q} + \overline{K}\overline{Q}$$




M.I. Norma Elva Chávez Rodríguez

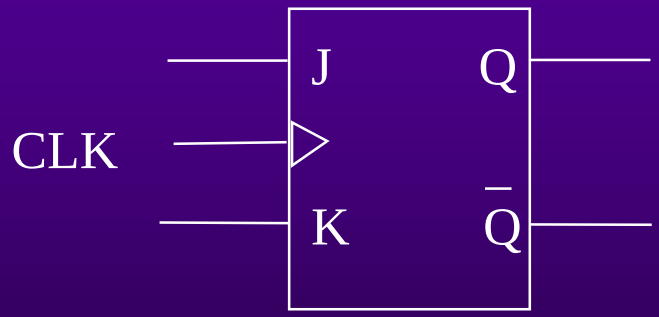
# Lógica secuencial.

## Tabla característica

| J | K | $Q_{t+1}$ |
|---|---|---|
| 0 | 0 | $Q_t$ |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | $\overline{Q_t}$ |

## Tabla de excitación

| Qt | $Q_{t+1}$ | J | K |
|---|---|---|---|
| 0 | 0 | 0 | * |
| 0 | 1 | 1 | * |
| 1 | 0 | * | 1 |
| 1 | 1 | * | 0 |




Símbolo

# Lógica secuencial.

Flip-Flop tipo D (Data).

Este Flip-Flop es una combinación del SR ó JK tal como se muestra:

Diagrama Interno

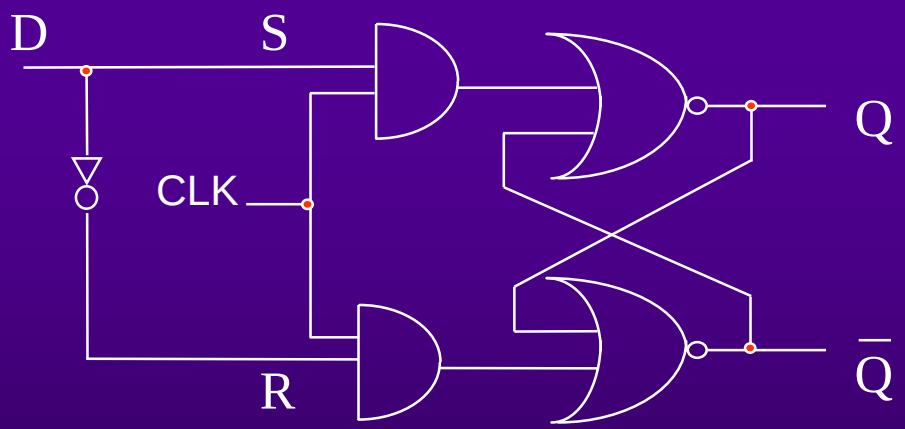



Tabla de verdad

| Q | D | Qt+1 |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

# Lógica secuencial.

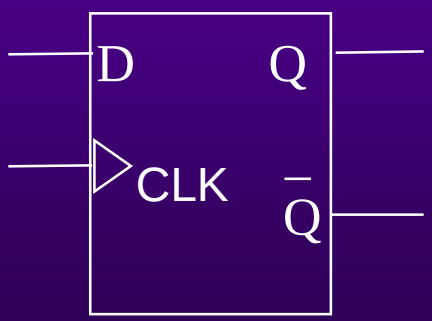
## Tabla característica

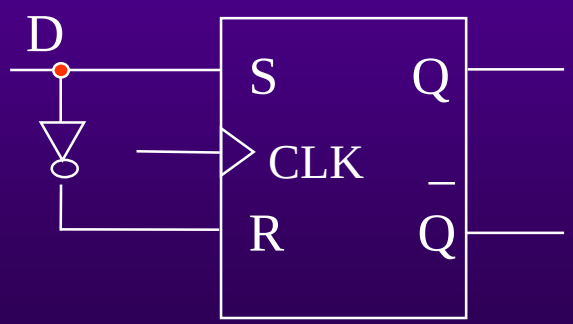
| D | $Q_{t+1}$ |
|---|---|
| 0 | 0 |
| 1 | 1 |

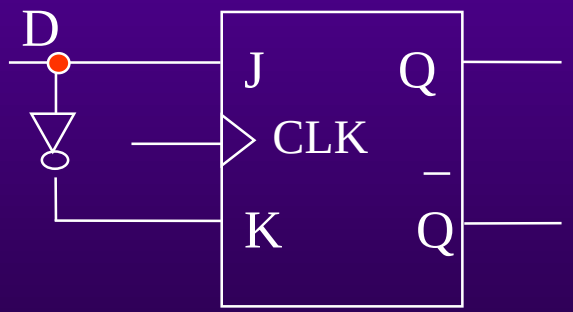
## Tabla de excitación

| $Q_t$ | D | $Q_{t+1}$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

## Símbolo



# Lógica secuencial.

M.I. Norma Elva Chávez Rodríguez

## Flip-Flop tipo T (toggle)

Este Flip-Flop es una modificación del JK como lo muestra la siguiente figura:

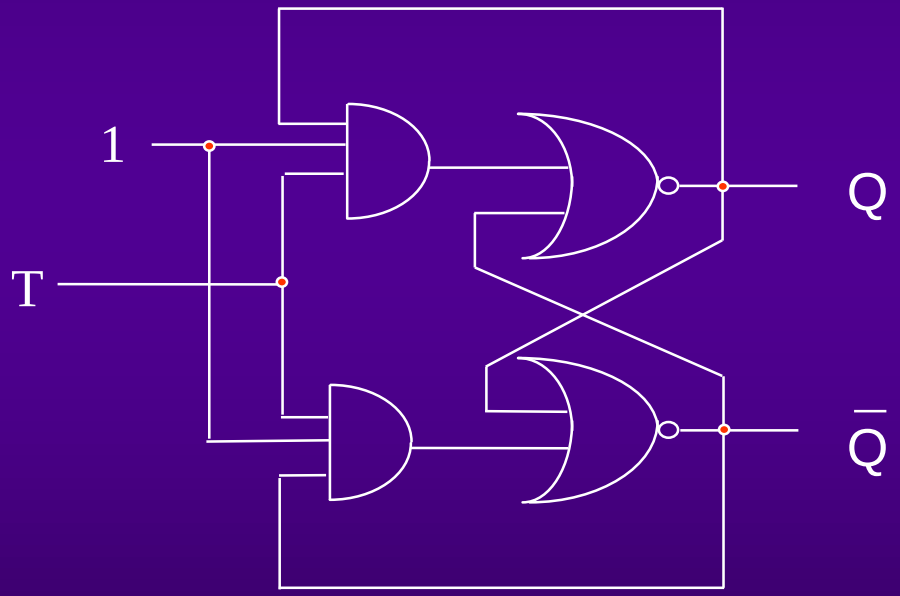



Tabla de verdad

| Q | T | $Q_{t+1}$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

| $Q_{t+1}$ / Q | T = 0 | T = 1 |
|---|---|---|
| 0 | 0 | 1 |
| 1 | 1 | 0 |

$$Q_{t+1} = Q \oplus T$$

# Lógica secuencial.

## Tabla característica

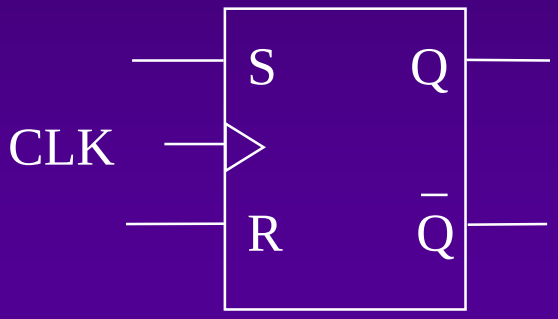
| T | $Q_{t+1}$ |
|---|---|
| 0 | $Q_t$ |
| 1 | $\overline{Q_t}$ |

## Tabla de excitación

| $Q_t$ | $Q_{t+1}$ | T |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

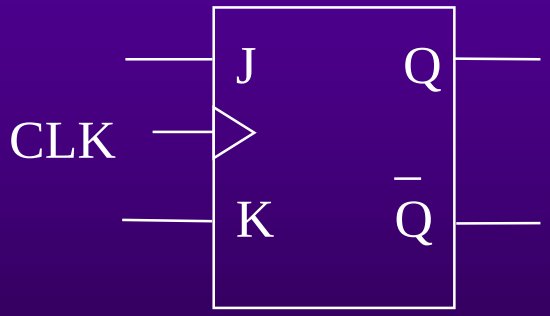
# Lógica secuencial.

## Resumen de Flip-Flop's




| S | R | $Q_{t+1}$ |
|---|---|---|
| 0 | 0 | $Q_t$ |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | ? |

| $Q_t$ | $Q_{t+1}$ | S | R |
|---|---|---|---|
| 0 | 0 | 0 | * |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | * | 0 |

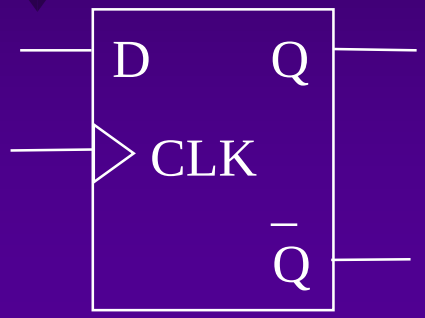



| J | K | $Q_{t+1}$ |
|---|---|---|
| 0 | 0 | $Q_t$ |
| 0 | 1 | 0 |
| 1 | 0 | 1 |
| 1 | 1 | $Q_t$ |

| $Q_t$ | $Q_{t+1}$ | J | K |
|---|---|---|---|
| 0 | 0 | 0 | * |
| 0 | 1 | 1 | * |
| 1 | 0 | * | 1 |
| 1 | 1 | * | 0 |

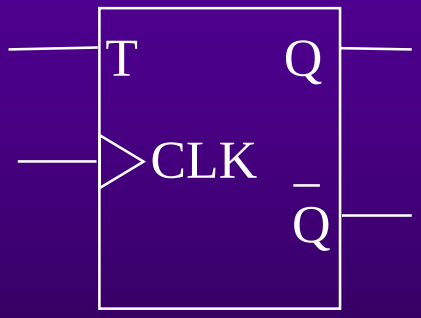
# Lógica secuencial.

## Resumen de Flip-Flop's




| D | $Q_{t+1}$ |
|---|---|
| 0 | 0 |
| 1 | 1 |

| $Q_t$ | $Q_{t+1}$ | D |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |




| T | $Q_{t+1}$ |
|---|---|
| 0 | $Q_t$ |
| 1 | $Q_t$ |

| $Q_t$ | $Q_{t+1}$ | T |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

# Lógica secuencial.

## Flip-Flops

| | |
|---|---|
| 7474 | Dual D-Type positive-edge-triggerd |
| 74LS74 | Flip-Flop with preset and clear |
| 74AS74 | |
| 74109 | Dual J-R positive-edge-triggered Flip-Flop |
| 74LS109 | with preset and clear |
| 74H101 | Gated J-R negative edge triggered Flip-Flop with preset |
| 74H102 | gated J-K negative edge triggered Flip-Flop with preset and clear |
| 74H103 | Dual J-R negative edge triggered Flip-Flop with clear |
| 74H106 | Dual J-K negative edge triggered Flip-Flop with clear |
| 74107 | Dual J-K Master/slave Flip-Flop with clear |
| 74LS107 | Dual J-R negative edge triggered Flip-Flop with clear |