INSTITUTO POLITÉCNICO NACIONAL
C.E.C. y T. No 1 "GONZALO VÁZQUEZ VELA"
SISTEMAS DIGITALES

APUNTES DE :

CIRCUITOS LÓGICOS COMBINATORIOS

ELABORADOS POR:
ING.ANGEL CRUZ ANTONIO

UNIDAD 1
SISTEMAS DE NUMERACIÓN Y CÓDIGOS

RAP # 1: Resuelve operaciones aritméticas utilizando diversos sistemas numéricos afines a los sistemas digitales.

RAP # 2: Establece relaciones entre el sistema binario y los códigos como una manera de representar información.

1.1 INTRODUCCIÓN

En la ciencia, la tecnología, la administración y de hecho en otros campos de la actividad humana, constantemente se manejan cantidades. Estas se miden, monitorean, registran, se manipulan aritméticamente, se observan, se utilizan en muchos sistemas físicos. Existen básicamente dos maneras de representar el valor numérico de las cantidades: la analógica y la digital.

La analógica es una cantidad que se denota por medio de otra que es directamente proporcional a la primera, quiere decir que es continua.

En la digital las cantidades se denotan no por cantidades proporcionales, sino por símbolos denominados dígitos o discreto, que varia paso a paso.

Los números y los códigos son el lenguaje básico de los microprocesadores.

NÚMERO DECIMAL

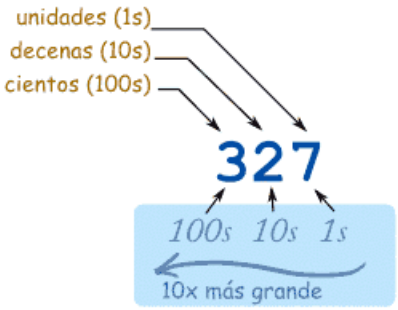
.Un número decimal (en base **10**) contiene un **punto decimal**.

Valor posicional

Cuando escribimos números, la **posición** (o "**lugar**") de cada número es importante.

En el número **327**:

- El "7" está en la posición de las **unidades**, así que vale 7 (o 7 "1"s),
- El "2" está en la posición de las **decenas**, así que son 2 dieses (o veinte),
- Y el "3" está en la posición de las **centenas**, así que vale 3 cientos.



"Trescientos veintisiete"

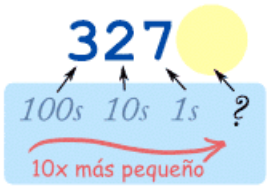
Cuando vamos a la izquierda, cada posición vale **¡10 veces más!**

De unidades, a decenas, a centenas

... y...

Cuando vamos a la derecha, cada posición es **10 veces más pequeña**.

De centenas, a decenas, a unidades



¿Pero qué pasa si seguimos antes de las unidades?

¿Qué es **10 veces más pequeño** que las unidades?

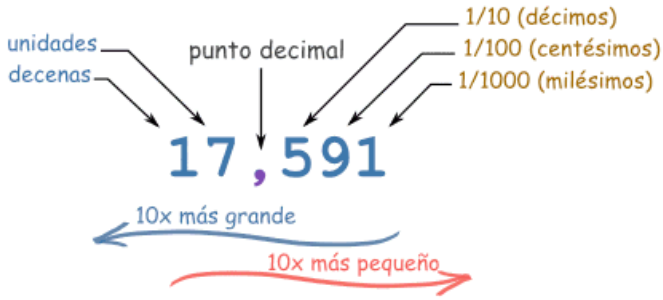
¡1/10 (décimos)!

Pero tenemos que poner un **punto decimal** (o coma decimal, depende de dónde vivas), para que sepamos exactamente dónde está la posición de las unidades:

"Trescientos veintisiete **y cuatro décimos**"

Y eso es un número decimal
Cada dígito de un número decimal va en una "posición", y el **punto decimal** nos dice qué posición es cada una.

La posición *justo a la izquierda* del punto son las "unidades". Cada vez que nos movemos a la izquierda vale 10 veces más, y a la derecha vale 10 veces menos:



Pero esto sólo es una manera de escribir números. Hay otras maneras como los números romanos, binarios, octal, hexadecimal.

El sistema decimal de numeración también se llama "base 10", porque se basa en el número 10.
En decimal hay diez símbolos (0 a 9), pero fíjate en esto: no hay un símbolo para el "diez". "10" son en realidad dos símbolos juntos, un "1" y un "0":

En decimal contamos 0,1,2,3,4,5,6,7,8,9, entonces decimos "me he quedado sin símbolos, así que empiezo otra vez con 0, pero primero voy a añadir 1 a la izquierda".
Pero **no es obligatorio** usar 10 como "base". Se podrá usar 2 ("binario"), 16 ("hexadecimal"), ¡o cualquier número que se quiera! Sólo se sigue la misma regla:
Cuenta hasta justo antes de la "base", después vuelve al 0, pero añadiendo 1 a la izquierda.

1.2 NUMERACIÓN BINARIA

Esta numeración opera con dos números el 0 y el 1 se utilizan para representar cualquier cantidad.

Es una numeración en base 2, donde los símbolos 0 y 1 vistos anteriormente asumen el valor numérico 0 y 1.

Los números binarios son en "base 2" en lugar de "base 10". Empieza contando 0, después 1, ¡ya se acabaron los dígitos! Así que vuelve al 0, pero aumenta en 1 el número de la izquierda.

Funciona así:

00**0**		0
00**1**		1
0**10**	No hay "2" en binario, así que volvemos al 0... ... y sumamos 1 a la cifra de la izquierda	2
01**1**		3
100	Volvemos otra vez al 0, y sumamos 1 a la izquierda... ... pero ese número ya es 1 así que vuelve a ser 0... ... y el 1 se suma al siguiente número a la izquierda	4
10**1**		5
1**10**	etc...	6

En el sistema binario:

- con 1 bit el valor más alto que se puede expresar es el 1;
- con 2 bits el valor más alto que se puede expresar es el 3;
- con 3 bits el valor más alto que se puede expresar es el 7;
- con 4 bits el valor más alto que se puede expresar es el 15;
- con **N** bits el valor más alto que se puede expresar es el $(2^N) - 1$.

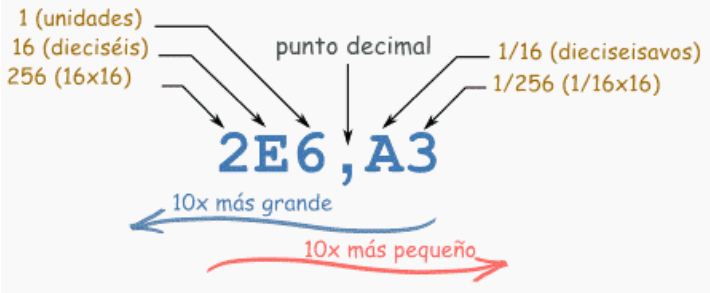
Ejemplo en número binario 1 1 1 1 = 15 decimal.

1.3 NUMERACIÓN OCTAL
Esta numeración opera con ocho números (símbolos) 0,1,2,3,4,5,6,7 se utilizan para representar cualquier número, su base es ocho.

El número mayor en base octal es el 7.
Para contar arriba de 7 se coloca un cero nuevamente en la posición de las unidades y se continúa contando 10, 11, 12, 13, 14, 15, 16, 17.

Después de 17 se coloca un cero nuevamente en la posición de las unidades y los siguientes números son: 20, 21, 22, 23, 24, 25, 26, 27 y así sucesivamente.

1.4 HEXADECIMALES
Un número hexadecimal es en base **16.**



Este es 2×16×16 + 14×16 + 6 + 10/16 + 3/(16×16)
Lee más abajo para averiguar por qué

Cada cifra se coloca a la izquierda o derecha del punto, para indicar valores más grandes o más pequeños que uno:

La que está justo a la izquierda del punto es un número entero, y a esa posición la llamamos **unidades**.

Cuando nos movemos a la izquierda, cada posición vale **16 veces más**.

La primera cifra a la derecha del punto vale **un dieciseisavo** (1/16).

Cuando nos movemos a la derecha, cada posición vale **16 veces menos** (un dieciseisavo de la anterior).

16 valores diferentes
Los números **hexadecimales** son como los números decimales hasta el 9, pero también se usan letras ("A","B","C","D","E","F") para los valores 10 a 15:

Decimal:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimal:	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Así que un dígito hexadecimal puede tomar 16 valores diferentes en lugar de 10.
Definición de hexadecimal

La palabra "hexadecimal" quiere decir "en base 16" (Del griego hexa*:* "seis" y del latín décima: "la décima parte").

SISTEMAS NUMÉRICOS

DEFINICIÓN.- Cualquier número en un sistema de base "n" puede ser representado de la siguiente forma:

$$a_m\ a_{m-1}\ a_{m-2} \ldots a_0 = \sum_{i=m}^{0} a_i\ b^i = (a_m\ b^m) + (a_{m-1}\ b^{m-1}) + \ldots + (a_0\ b^0)$$

Donde **b** es la base del sistema.

Con la expresión anterior cualquier número de cualquier base se puede convertir a un número decimal.

1.5 CONVERSIONES DE NÚMEROS ENTEROS Y RACIONALES DE UN SISTEMA DE NUMERACIÓN A OTRO.

Ejemplos:
DE DECIMAL EN DECIMAL

Convertir el número decimal **528** en decimal
5 centenas + 2 decenas + 8 unidades, es decir:
$5 \cdot 10^2 + 2 \cdot 10^1 + 8 \cdot 10^0$ o, lo que es lo mismo:
500 + 20 + 8 = 528

Por ejemplo, el número **8245,97** se calcularía como:

8 millares + 2 centenas + 4 decenas + 5 unidades + 9 décimos + 7 céntimos
$8 \cdot 10^3 + 2 \cdot 10^2 + 4 \cdot 10^1 + 5 \cdot 10^0 + 9 \cdot 10^{-1} + 7 \cdot 10^{-2}$, es decir:
8000 + 200 + 40 + 5 + 0,9 + 0,07 = 8245,97

BINARIO EN DECIMAL

El número binario 1011 tiene un valor en decimal, que se calcula así:

$1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$, es decir:
8 + 0 + 2 + 1 = 11

Y para expresar que ambas cifras describen la misma cantidad lo escribimos así:

$1011_2 = 11_{10}$

Convertir el número binario 1010011_2 a decimal:

$1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 83$
$1010011_2 = 83_{10}$

Ejemplo de número binario 10110. a Decimal
Así, tenemos que el número binario 10110 o en base 2, equivale a: 22 en decimal
$1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 16 + 0 + 4 + 2 + 0 = (22)_{10}$

$(10110)_2 = (22)_{10}$

$101111_2 = 1.2^5+0.2^4+1.2^3+1.2^2+1.2^1+1.2^0 = 45_{10}$

$10101_2 = 1.2^4+0.2^3+1.2^2+0.2^1+1.2^0 = 21_{10}$

CONVERTIR OCTAL EN DECIMAL

El número octal $\mathbf{273_8}$ tiene un valor que se calcula así:
$2 \cdot 8^3 + 7 \cdot 8^2 + 3 \cdot 8^1 = 2 \cdot 512 + 7 \cdot 64 + 3 \cdot 8 = 1496_{10}$

$273_8 = 1496_{10}$

Por ejemplo, para convertir el número $\mathbf{237_8}$ a decimal basta con desarrollar el valor de cada dígito:
$2 \cdot 8^2 + 3 \cdot 8^1 + 7 \cdot 8^0 = 128 + 24 + 7 = 159_{10}$

$237_8 = 159_{10}$

$740_8 = 7.8^2+4.8^1+4.8^0 = 484_{10}$

CONVERTIR HEXADECIMAL EN DECIMAL

Ejemplo 1:¿Cuánto es 2E6 (hexadecimal)? En decimal
El "2" está en la posición de "16×16", así que vale $2 \times 16 \times 16 = 2 \cdot 16^2$
La "E" está en la posición de "16", así que vale $14 \times 16 = 14 \cdot 16^1$
El "6" está en la posición de las "unidades" así que vale $6. = 6 \cdot 16^0$
Respuesta: $2E6 = 2 \cdot 16^2 + 14 \cdot 16^1 + 6 \cdot 16^0 = 2 \times 16 \times 16 + 14 \times 16 + 6 = 742$ en decimal)
$(2E6)_{16} = (742)_{10}$

Ejemplo 2: ¿Cuánto es 2,3 (hexadecimal)? En decimal
A la izquierda del punto hay "2", esa es la parte entera.
El 3 está en la posición de los "dieciseisavo", así que vale "3 dieciseisavos", que son 3/16
Así, 2,3 es "2 y 3 dieciseisavos" (=2,1875 en decimal)
$(2.3)_{16} = (2.1875)_{10}$

Calculemos el valor del número hexadecimal 1A3F16 en decimal.

$1A3F_{16} = 1 \cdot 16^3 + A \cdot 16^2 + 3 \cdot 16^1 + F \cdot 16^0$

$1 \cdot 4096 + 10 \cdot 256 + 3 \cdot 16 + 15 \cdot 1 = 6719$

$1A3F_{16} = 6719_{10}$

Convertir el número $31F_{16}$ a decimal.
$31F_{16} = 3x16^2 + 1x16 + 15 \times 16^0 = 3x256 + 16 + 15 = 768 + 31 = 799_{10}$

PARA CONVERTIR UN NÚMERO DECIMAL A BINARIO, OCTAL O HEXADECIMAL BASTA CON DIVIDIR ENTRE EL NÚMERO QUE SE DESEA CONVERTIR.

CONVERSIÓN DE NÚMEROS DECIMALES EN BINARIOS

Convertir un número decimal al sistema binario es muy sencillo: basta con realizar divisiones sucesivas por 2 y escribir los restos obtenidos en cada división en orden inverso al que han sido obtenidos.
Por ejemplo, para convertir al sistema binario el número 77_{10} haremos una serie de divisiones que arrojarán los restos siguientes:
77 : 2 = 38 Resto: 1
38 : 2 = 19 Resto: 0
19 : 2 = 9 Resto: 1
9 : 2 = 4 Resto: 1
4 : 2 = 2 Resto: 0
2 : 2 = 1 Resto: 0
1 : 2 = 0 Resto: 1
Y, tomando los restos en orden inverso obtenemos la cifra binaria:

$77_{10} = 1001101_2$

Convertir de decimal a binario

325		510		1012		835		435	
162	1	255	0	506	0	417	1	217	1
81	0	127	1	253	0	208	1	108	1
40	1	63	1	126	1	104	0	54	0
20	0	31	1	63	0	52	0	27	0
10	0	15	1	31	1	26	0	13	1
5	0	7	1	15	1	13	0	6	1
2	1	3	1	7	1	6	1	3	0
1	0	1	1	3	1	3	0	1	1
0	1	0	1	1	1	1	1	0	1
				0	1	0	1		

101000101
111111110
1111110100
1101000011
110110011

Convertir 107.635 decimal a binario

Nº Decimal	Base	Cociente	Resto
107	2	53	1
53	2	26	1
26	2	13	0
13	2	6	1
6	2	3	0
3	2	1	1

$107_{10} = 1101011_2$

Cuando tengamos un número con decimales seguiremos el siguiente procedimiento: multiplicaremos por 2 la parte decimal y se toma como dígito binario su parte entera. El proceso se repite con la fracción decimal resultante del paso anterior, hasta obtener una fracción decimal nula, o bien hasta obtener el número de cifras binarias que se desee. Ejemplo: 107,645. Como anteriormente convertimos 107 a binario, el resultado de la conversión quedaría así:
$1101011,\ 10100101_2$

Fracción decimal	Multiplicado por:	Resultado	Dígito binario
0,645	2	1,290	1
0,290	2	0,580	0
0,580	2	1,160	1
0.160	2	0,320	0
0,320	2	0.64	0
0.64	2	1.28	1
0.28	2	0.56	0
0.56	2	1.12	1

CONVERTIR DECIMAL EN OCTAL
Convertir el número 465_{10} a octal. Dividir siempre entre 8

Número N	N ÷ 8	Parte decimal	Parte decimal x 8	Peso
465	58	0,125	1	LSB
58	7	0,25	2	
7	0	0,875	7	MSB

El resultado en octal de $\mathbf{465_{10}}$ es $\mathbf{721_8}$

Convertir 93_{10} a octal
93
11 5
1 3
0 1
$\mathbf{93_{10} = 135_8}$

Convertir 128_{10} a octal
128
16 0
2 0
0 2
$\mathbf{128_{10} = 200_8}$

CONVERTIR DECIMAL EN HEXADECIMAL
Para convertir a hexadecimal del número $\mathbf{1735_{10}}$ será necesario hacer las siguientes divisiones:

1735 : 16 = 108 Resto: 7
108 : 16 = 6 Resto: C es decir, 12_{10}
6 : 16 = 0 Resto: 6
De ahí que, tomando los restos en orden inverso, resolvemos el número en hexadecimal:
$\mathbf{1735_{10} = 6C7_{16}}$

Convertir el número 1869_{10} a hexadecimal.

1868: 16 = 13 Resto D es decir 13
116: 16 = 7 resto 4
7 : 16 = 0 resta 7
$\mathbf{1869_{10} = 74\ D_{16}}$

CONVERTIR BINARIO EN OCTAL

Se toma los grupos de tres bits y se sustituyen por su equivalente octal:
Por ejemplo, convertir el número binario $\mathbf{101001011_2}$ a octal
$101_2 = 5_8$
$001_2 = 1_8$
$011_2 = 3_8$
Y, de ese modo: $\mathbf{101001011_2 = 513_8}$

Convertir el número $\mathbf{01010101_2}$ a octal.
001 010 101 = $\mathbf{125_8}$
1 2 5

Ejemplo: $10011111,11111_2$

Resultado: $237,76_8$
Observa como ha sido necesario añadir un cero en la última agrupación de la parte entera y otro en la parte fraccionaria para completar los grupos de 3 dígitos.

Agrupación	Equivalente octal
010	2
011	3
111	7
,	,
111	7
110	6

CONVERTIR OCTAL EN BINARIO
Tomar el equivalente binario de cada uno de sus dígitos del número octal:

Por ejemplo, para convertir el número octal 750 a binario,
$7_8 = 111_2$
$5_8 = 101_2$
$0_8 = 000_2$
Y, por tanto: $\mathbf{750_8 = 111101000_2}$

Convertir 55.35 octal a binario

Carácter octal	Nº binario
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Ejemplo: $55,35_8$
Resultado: 101, 101,011 101_2

CONVERTIR BINARIO EN HEXADECIMAL
Bastará con tomar grupos de cuatro bits, empezando por la derecha, y reemplazarlos por su equivalente hexadecimal:

Por ejemplo, para expresar en hexadecimal el número binario 101001110011_2
$1010_2 = A_{16}$
$0111_2 = 7_{16}$
$0011_2 = 3_{16}$
Y, por tanto: $\mathbf{101001110011_2 = A73_{16}}$

En caso de que los dígitos binarios no formen grupos completos de cuatro dígitos, se deben añadir ceros a la izquierda hasta completar el último grupo. Por ejemplo:
$\mathbf{101110_2 = 00101110_2 = 2E_{16}}$

Convertir el número 10011101010 a hexadecimal.
0100 1110 1010 = 4 E A_{16}
4 E A

CONVERTIR HEXADECIMAL EN BINARIO
Para convertir a binario, se tomaran grupos de cuatro bits por cada número hexadecimal.
Ejemplo convertir $\mathbf{1F6_{16}}$ a binario
$\mathbf{1_{16} = 0001_2}$
$\mathbf{F_{16} = 1111_2}$
$\mathbf{6_{16} = 0110_2}$
Y, por tanto: $\mathbf{1F6_{16} = 000111110110_2}$
Convertir el número $1F0C_{16}$ a binario.
$\mathbf{1F0C_{16} = 1111100001100_2}$

Convertir 5F.C4 hexadecimal a binario

Sistema binario	Sistema Hexadecimal
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Ejemplo: $1011111,110001_2$
Agrupando obtenemos el siguiente resultado:
$0101\ 1111,\ 1100\ 0100_2$
Sustituyendo según la tabla logramos la conversión esperada:
$5F,\ C4_{16}$

Convertir:
$69DE_{16}$= $0110\ 1001\ 1101\ 1110_2$

TABLA DE CONVERSIÓN-DECIMAL, HEXADECIMAL, OCTAL, BINARIO

Dec	Hex	Oct	Bin	Dec	Hex	Oct	Bin	Dec	Hex	Oct	Bin	Dec	Hex	Oct	Bin
0	0	000	00000000	16	10	020	00010000	32	20	040	00100000	48	30	060	00110000
1	1	001	00000001	17	11	021	00010001	33	21	041	00100001	49	31	061	00110001
2	2	002	00000010	18	12	022	00010010	34	22	042	00100010	50	32	062	00110010
3	3	003	00000011	19	13	023	00010011	35	23	043	00100011	51	33	063	00110011
4	4	004	00000100	20	14	024	00010100	36	24	044	00100100	52	34	064	00110100
5	5	005	00000101	21	15	025	00010101	37	25	045	00100101	53	35	065	00110101
6	6	006	00000110	22	16	026	00010110	38	26	046	00100110	54	36	066	00110110
7	7	007	00000111	23	17	027	00010111	39	27	047	00100111	55	37	067	00110111
8	8	010	00001000	24	18	030	00011000	40	28	050	00101000	56	38	070	00111000
9	9	011	00001001	25	19	031	00011001	41	29	051	00101001	57	39	071	00111001
10	A	012	00001010	26	1A	032	00011010	42	2A	052	00101010	58	3A	072	00111010
11	B	013	00001011	27	1B	033	00011011	43	2B	053	00101011	59	3B	073	00111011
12	C	014	00001100	28	1C	034	00011100	44	2C	054	00101100	60	3C	074	00111100
13	D	015	00001101	29	1D	035	00011101	45	2D	055	00101101	61	3D	075	00111101
14	E	016	00001110	30	1E	036	00011110	46	2E	056	00101110	62	3E	076	00111110
15	F	017	00001111	31	1F	037	00011111	47	2F	057	00101111	63	3F	077	00111111

Dec	Hex	Oct	Bin
64	40	100	01000000
65	41	101	01000001
66	42	102	01000010
67	43	103	01000011
68	44	104	01000100
69	45	105	01000101
70	46	106	01000110
71	47	107	01000111
72	48	110	01001000
73	49	111	01001001
74	4A	112	01001010
75	4B	113	01001011
76	4C	114	01001100
77	4D	115	01001101
78	4E	116	01001110
79	4F	117	01001111
80	50	120	01010000
81	51	121	01010001
82	52	122	01010010
83	53	123	01010011
84	54	124	01010100
85	55	125	01010101
86	56	126	01010110
87	57	127	01010111
88	58	130	01011000
89	59	131	01011001
90	5A	132	01011010
91	5B	133	01011011
92	5C	134	01011100
93	5D	135	01011101
94	5E	136	01011110
95	5F	137	01011111
96	60	140	01100000
97	61	141	01100001
98	62	142	01100010
99	63	143	01100011
100	64	144	01100100
101	65	145	01100101
102	66	146	01100110
103	67	147	01100111
104	68	150	01101000
105	69	151	01101001
106	6A	152	01101010
107	6B	153	01101011
108	6C	154	01101100
109	6D	155	01101101
110	6E	156	01101110
111	6F	157	01101111
112	70	160	01110000
113	71	161	01110001
114	72	162	01110010
115	73	163	01110011
116	74	164	01110100
117	75	165	01110101
118	76	166	01110110
119	77	167	01110111
120	78	170	01111000
121	79	171	01111001
122	7A	172	01111010
123	7B	173	01111011
124	7C	174	01111100
125	7D	175	01111101
126	7E	176	01111110
127	7F	177	01111111

Dec	Hex	Oct	Bin
128	80	200	10000000
129	81	201	10000001
130	82	202	10000010
131	83	203	10000011
132	84	204	10000100
133	85	205	10000101
134	86	206	10000110
135	87	207	10000111
136	88	210	10001000
137	89	211	10001001
138	8A	212	10001010
139	8B	213	10001011
140	8C	214	10001100
141	8D	215	10001101
142	8E	216	10001110
143	8F	217	10001111
144	90	220	10010000
145	91	221	10010001
146	92	222	10010010
147	93	223	10010011
148	94	224	10010100
149	95	225	10010101
150	96	226	10010110
151	97	227	10010111
152	98	230	10011000
153	99	231	10011001
154	9A	232	10011010
155	9B	233	10011011
156	9C	234	10011100
157	9D	235	10011101
158	9E	236	10011110
159	9F	237	10011111
160	A0	240	10100000
161	A1	241	10100001
162	A2	242	10100010
163	A3	243	10100011
164	A4	244	10100100
165	A5	245	10100101
166	A6	246	10100110
167	A7	247	10100111
168	A8	250	10101000
169	A9	251	10101001
170	AA	252	10101010
171	AB	253	10101011
172	AC	254	10101100
173	AD	255	10101101
174	AE	256	10101110
175	AF	257	10101111
176	B0	260	10110000
177	B1	261	10110001
178	B2	262	10110010
179	B3	263	10110011
180	B4	264	10110100
181	B5	265	10110101
182	B6	266	10110110
183	B7	267	10110111
184	B8	270	10111000
185	B9	271	10111001
186	BA	272	10111010
187	BB	273	10111011
188	BC	274	10111100
189	BD	275	10111101
190	BE	276	10111110
191	BF	277	10111111

Dec	Hex	Oct	Bin
192	C0	300	11000000
193	C1	301	11000001
194	C2	302	11000010
195	C3	303	11000011
196	C4	304	11000100
197	C5	305	11000101
198	C6	306	11000110
199	C7	307	11000111
200	C8	310	11001000
201	C9	311	11001001
202	CA	312	11001010
203	CB	313	11001011
204	CC	314	11001100
205	CD	315	11001101
206	CE	316	11001110
207	CF	317	11001111
208	D0	320	11010000
209	D1	321	11010001
210	D2	322	11010010
211	D3	323	11010011
212	D4	324	11010100
213	D5	325	11010101
214	D6	326	11010110
215	D7	327	11010111
216	D8	330	11011000
217	D9	331	11011001
218	DA	332	11011010
219	DB	333	11011011
220	DC	334	11011100
221	DD	335	11011101
222	DE	336	11011110
223	DF	337	11011111
224	E0	340	11100000
225	E1	341	11100001
226	E2	342	11100010
227	E3	343	11100011
228	E4	344	11100100
229	E5	345	11100101
230	E6	346	11100110
231	E7	347	11100111
232	E8	350	11101000
233	E9	351	11101001
234	EA	352	11101010
235	EB	353	11101011
236	EC	354	11101100
237	ED	355	11101101
238	EE	356	11101110
239	EF	357	11101111
240	F0	360	11110000
241	F1	361	11110001
242	F2	362	11110010
243	F3	363	11110011
244	F4	364	11110100
245	F5	365	11110101
246	F6	366	11110110
247	F7	367	11110111
248	F8	370	11111000
249	F9	371	11111001
250	FA	372	11111010
251	FB	373	11111011
252	FC	374	11111100
253	FD	375	11111101
254	FE	376	11111110
255	FF	377	11111111

1.6 OPERACIONES BINARIAS

Suma binaria
La suma binaria se puede realizar cómodamente siguiendo las tres reglas descritas:
Si el número de unos (en sentido vertical) es par el resultado es 0.
Si el número de unos (en sentido vertical) es impar el resultado es 1.
Acarreo tantos unos como parejas (completas) de números 1 haya.
Por ejemplo:
0 + 0 = 0, con acarreo 0
0 + 1 = 1, con acarreo 0
1 + 0 = 1, con acarreo 0
1 + 1 = 10 se pone 0 y se acarrea un 1 a la posición siguiente.

Para sumar 1 0 1 0 (que en decimal es 10) y 1111 (que en decimal es 15) 10 + 15 =25

```
  1 0 1 0
 +1 1 1 1
 ---------
 1 1 0 0 1
```

También sumar 152 + 21 = 173

```
  1 0 0 1 1 0 0 0
+ 0 0 0 1 0 1 0 1
_________________
  1 0 1 0 1 1 0 1
```

RESTA BINARIA

Las cuatro reglas básicas para la resta de números binarios son:

0 - 0 = 0
1 – 1 = 0
1 – 0 = 1
0 – 1 = 1 (con acarreo negativo de 1)

Al restarse números algunas veces se genera un acarreo negativo que pasa a la siguiente columna de la izquierda. En binario sólo se produce este acarreo cuando se intenta restar 1 de 0 (4ª regla).

Ejemplo sobre esta situación, restar 011 de 101:

101 – 011 = 010

```
 1 0 1
-0 1 1
--------
 0 1 0
```

1. En la columna derecha se realiza la resta de 1 – 1 = 0
2. En la columna central se produce un acarreo negativo de 1 a la columna siguiente (4ª regla) que da lugar a 1 en esta columna, luego 0 - 1 = 1 con acarreo de 1 a la siguiente columna.
3. en la columna izquierda, se resta 1 del acarreo producido en la anterior columna y da como resultado 0, luego se resta 0 – 0 = 0

Restar los siguientes números:

1 0 0 0 1 = 17 **1 1 0 1 1 0 0 1 = 217**
- 0 1 0 1 0 = 10 **-1 0 1 0 1 0 1 1 = 171**
0 0 1 1 1 = 7 **0 0 1 0 1 1 1 0 = 46**

En sistema decimal sería: 17 - 10 = 7 y 217 - 171 = 46.

Para simplificar las restas y reducir la posibilidad de cometer errores hay varios métodos:

- Dividir los números largos en grupos. En el siguiente ejemplo, vemos cómo se divide una resta larga en tres restas cortas:

1 0 0 1 1 0 0 1 1 1 0 1		**1 0 0 1**	**1 0 0 1**	**1 1 0 1**
- 0 1 0 1 0 1 1 1 0 0 1 0		**-0 1 0 1**	**-0 1 1 1**	**-0 0 1 0**
0 1 0 0 0 0 1 0 1 0 1 1	=	**0 1 0 0**	**0 0 1 0**	**1 0 1 1**

RESTA POR COMPLEMENTO A 1
Consiste en cambiar “1” por el “0” y el “0” por el “1”
Ejemplo
1001 su complemento a unos es 0110

MÉTODO DE RESTA POR COMPLEMENTO A 1
1.-COMPLEMENTAR A 1 EL SUSTRAENDO
2.-SUMAR EL NÚMERO COMPLEMENTO CON EL MINUENDO
3.-SI EXISTE ACARREO SE SUMA CON EL RESULTADO, SI NO EXISTE ACARREO EL RESULTADO ES NEGATIVO Y SE COMPLEMENTA A 1

Ejemplo
11010 = 26 (1) **el complemento a unos** (2) **11010 como hay acarreo se**
- 10111 =23 **de 10111 es 01000** **+ 01000 le suma al resulta**
------ **------- do**
00011 = 3 **100010**

(3) **00010**
+00001

00011 = 3 es positivo, resultado final

10111 = 23 el complemento a 1 es **10111** como no hay acarreo el
-11010 = 26 00101 **+ 00101** resultado se complementa a
-------- -------- 1 o sea **00011 = 3** es negativo.
-00011 = -3 **11100** Resultado final.

101110 = 26 el complemento es **101110** como no hay acarreo, el resultado
-110011 = 51 001100 **+001100** se complementa a 1
--------- --------- **000101 = 5** negativo.
-000101 = -5 **111010** Resultado final.

101111 = 47 el complemento a 1 **101111** como hay acarreo **000100**
-101010 = 42 010101 **+010101** **+ 1**
---------- ---------- ----------
000101 = 5 **1000100** **000101 = 5**

RESTA POR COMPLEMENTO A 2
Complemento a 2 es realizar el complemento a 1 de un número y sumarle 1 10111

01000
+ 1

01001

LA METODOLOGÍA DE LA RESTA POR COMPLEMENTO A 2 ES:
1.-OBTENER EL COMPLEMENTO A 2 DEL SUSTRAENDO
2.-SUMAR EL MINUENDO CON EL RESULTADO ANTERIOR.
3.-SI EXISTE ACARREO EL RESULTADO ES POSITIVO.
4.-SI NO HAY ACARREO EL RESULTADO ES NEGATIVO, Y SE
COMPLEMENTA A 2.

10011 = 19 el complemento a 2 es **10011**
-01001 = 9 10110 **+10111**
-------- + 1 --------
01010 = 10 -------- **101010 = 10** positivo por el acarreo
10111

101011 = 43 el complemento a 2 es **101011**
-100100 = 36 011011 **+011100**
---------- + 1 ----------
= 7 ---------- **1000111 = 7** positivo por el acarreo
011100

100100 =36 el complemento a 2 es **100100**
-101011 =43 010100 **+010101**
--------- + 1 ---------
= -7 --------- **111001 =** no hay acarreo
010101

Como no hay acarreo 111001 se complementa a 2, o sea:

0 0 0 1 1 0
+ 1
----
0 0 0 1 1 1 = 7 y es negativo

La siguiente resta, 91 - 46 = 45, en binario es:

1011011		**1011011**
-0101110	el C2 de 0101110 es 1010010	**+1010010**
0101101		**10101101**

En el resultado nos sobra un BIT, que se desborda por la izquierda. Pero, como el número resultante no puede ser más largo que el minuendo, el BIT sobrante se desprecia.

Un último ejemplo: vamos a restar 219 - 23 = 196, directamente y utilizando el complemento a dos:

11011011		**11011011**
-00010111	el C2 de 00010111 es 11101001	**+11101001**
11000100		**111000100**

Y, despreciando el BIT que se desborda por la izquierda, llegamos al resultado correcto: 11000100 en binario, 196 en decimal.

PRODUCTO DE NÚMEROS BINARIOS

El algoritmo del producto en binario es igual que en números decimales; aunque se lleva a cabo con más sencillez, ya que el 0 multiplicado por cualquier número da 0, y el 1 es el del producto.
Por ejemplo, multipliquemos 10110 por 1001:

10110 = 22
1001 = 9

10110
00000
00000
10110

11000110= 198

Multiplicar 11101111 por 111011

```
      11101111
        111011
      ________
       11101111
      11101111
     00000000
    11101111
   11101111
  11101111
 ______________
 11011100010101
```

DIVISIÓN BINARIA

Reglas de la división binaria: 0/0 no permitida, 1/0 no permitida, 0/1=0, 1/1=1.
División: Se hace igual como el sistema decimal.
Ejemplos

```
        10               110110              0111
      ______           __________          ________
101 /  1010       10 / 1101100       1001 / 1000110
       101             10                   1001
       0000            010                  10001
                        10                   1001
                        0011                 10000
                          10                 01001
                           10                 00111
                           10
                           00
```

1.7 CÓDIGOS BINARIOS

Código BCD decimal codificado en binario (binary coded decimal)

Para poder compartir información, que está en formato digital, es común utilizar las representaciones binaria y hexadecimal.
Hay otros métodos de representar información y una de ellas es el **código BCD**.
Con ayuda de la **codificación BCD** es más fácil ver la relación que hay entre un número decimal (base 10) y el número correspondiente en **binario** (base 2).

El **código BCD** utiliza **4 dígitos binarios.**
Para poder obtener el equivalente **código BCD**, se asigna un "peso" o "valor" según la posición que ocupa.
Este "peso" o "valor" sigue el siguiente orden: **8 - 4 - 2 - 1.** (Es un código ponderado o pesado).
Ejemplo: se observa que el número 5 se representa como: 0 1 0 1 donde.
El primer "0" corresponde al **8**,
El primer "1" corresponde a **4**,
El segundo "0" corresponde a **2**, y...
El segundo "1" corresponde a **1**.

De lo anterior: **0** x **8** + **1** x **4** + **0** x **2** + **1** x **1** = **5**

Al **código BCD 8 4 2 1** que tiene los "pesos" o "valores" antes descritos se le llama: **Código BCD natural.**

El **código BCD** cuenta como un número binario normal del 0 al 9, pero del diez (1010) al quince (1111) no son permitidos pues no existen, para estos números, el equivalente de una cifra en decimal.
Este código es utilizado, entre otras aplicaciones, para la representación de las cifras de los números decimales en displays de 7 segmentos.

DECIMAL	BCD
	8 4 2 1
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1

CÓDIGO 2421 (Aiken)
El código BCD Aiken es un código similar al código BCD natural con los "pesos" o "valores" distribuidos de manera diferente.
En el código BCD natural, los pesos son: 8 - 4 - 2 - 1, en el código Aiken la distribución es: 2 - 4 - 2 - 1
La razón de esta codificación es la de conseguir simetría entre ciertos números.
Analizar la tabla siguiente:
Ver la simetría en el **código Aiken** correspondiente a los decimales: 4 y 5, 3 y 6, 2 y 7, 1 y 8, 0 y 9.

Cada cifra es el complemento a 9 de la cifra simétrica en todos sus dígitos. (los "1" se vuelven "0" y los "0" se vuelven "1")

Ejemplo: 3 (0011) y 6 (1100). Tomar en cuenta los nuevos "pesos" en este código.
El **código Aiken** es muy útil para realizar operaciones de resta y división.

DECIMAL	Aiken
	2 4 2 1
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	1 0 1 1
6	1 1 0 0
7	1 1 0 1
8	1 1 1 0
9	1 1 1 1

1.9 CÓDIGO BCD EXCESO 3

Código no pesado porque cada BIT no tiene un peso especial.
El **código BCD Exceso 3** se obtiene sumando **3** a cada combinación del **código BCD natural**. Ver la tabla inferior a la derecha.

El **código BCD exceso 3** es un código en donde la ponderación no existe (no hay "pesos" como en el **código BCD natural** y **código Aiken**).

Al igual que el **código BCD Aiken** cumple con la misma característica de simetría. Cada cifra es el complemento a 9 de la cifra simétrica en todos sus dígitos.

DECIMAL	BCD	Exceso 3
	8 4 2 1	
0	0 0 0 0	0 0 1 1
1	0 0 0 1	0 1 0 0
2	0 0 1 0	0 1 0 1
3	0 0 1 1	0 1 1 0
4	0 1 0 0	0 1 1 1
5	0 1 0 1	1 0 0 0
6	0 1 1 0	1 0 0 1
7	0 1 1 1	1 0 1 0
8	1 0 0 0	1 0 1 1
9	1 0 0 1	1 1 0 0

Ver la simetría en el **código exceso 3** correspondiente a los decimales: 4 y 5, 3 y 6, 2 y 7, 1 y 8, 0 y 9
Es un código muy útil en las operaciones de resta y división.

1.10 EL CÓDIGO GRAY

El **código Gray** no es pesado (los dígitos que componen el código no tienen un peso asignado).
Su característica es que entre una combinación de dígitos y la siguiente, sea ésta anterior o posterior, sólo hay una diferencia de un dígito.
Por eso también se le llama **código progresivo**.
Esta progresión sucede también entre la última y la primera combinación. Por eso se le llama también código cíclico.

El **código GRAY** es utilizado principalmente en sistemas de posición, ya sea angular o lineal. Sus aplicaciones principales se encuentran en la **industria y en robótica**.

En **robótica** se utilizan unos **discos codificados** para dar la información de posición que tiene un eje en particular. Esta información se da en **código GRAY**.
Analizando la tabla se observa que:

Decimal	Binario	GRAY
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1
10	1 0 1 0	1 1 1 1
11	1 0 1 1	1 1 1 0
12	1 1 0 0	1 0 1 0
13	1 1 0 1	1 0 1 1
14	1 1 1 0	1 0 0 1
15	1 1 1 1	1 0 0 0

Cuando un **número binario** pasa de:
0111 a 1000 (de 7 a 8 en decimal) o de
1111 a 0000 (de 16 a 0 en decimal) cambian todas las cifras.

Para el mismo caso pero en **código Gray**:
0100 a 1100 (de 7 a 8 en decimal) o de
1000 a 0000 (de 16 a 0 en decimal) sólo ha cambiado una cifra.

La característica de pasar de un **código** al siguiente cambiando sólo un dígito asegura menos posibilidades de error.

CÓDIGO ALFANUMÉRICO

1.11CÓDIGO ASCCI

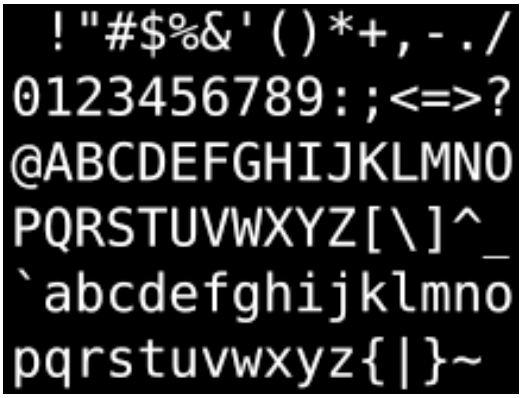
El código **ASCII** (**A**merican **S**tandard **C**ode for **I**nformation **I**nterchange — (Código Estadounidense Estándar para el Intercambio de Información), pronunciado generalmente [áski].

Representa números, letras y algunos símbolos.

7 bits

127 combinaciones
Alt. + 64 = @

Siempre que se usa el código ASCII primero es ALT.



1.12 CONVERSIÓN ENTRE DIFERENTES CÓDIGOS

CONVERSIÓN DE BINARIO A CÓDIGO GRAY
La conversión entre el código binario y el código Gray a veces es muy útil. Primeramente, mostraremos cómo convertir un número binario a un número de código Gray. Se aplican las siguientes reglas:
El BIT más significativo (el más a la izquierda) en el código Gray es el mismo que el MSB correspondiente en el número binario,
Yendo de izquierda a derecha, sume cada par adyacente de bits del código binario, para obtener el siguiente BIT del código Gray. Descarte acarreos.

Ejemplo: Paso 1. El dígito del código Gray más a la izquierda es el mismo que el dígito del código binario más a la izquierda.

1	0	1	1	0	Binario
1					Gray

Paso 2. Sume el BIT de código binario más a la izquierda al BIT adyacente.

1 + 0		1	1	0	Binario
1	1				Gray

Paso 3. Sume el siguiente par adyacente.

1	0 + 1		1	0	Binario
1	1	1			Gray

Paso 4. Sume el siguiente par adyacente y descarte el acarreo.

1	0	1 + 1		0	Binario
1	1	1	0		Gray

Paso 5. Sume el último par adyacente.

1	0	1	1 +0		Binario
1	1	1	0	1	Gray

La conversión ha sido completada; el código Gray es 11101.

CONVERSIÓN DE GRAY A BINARIO

Para convertir de código Gray a Binario, se utiliza un método similar, pero con algunas diferencias. Se aplican las siguientes reglas:
El BIT más significativo (el más a la izquierda) es el código binario es el mismo que el BIT correspondiente en el código Gray.
Sume cada BIT generado del código binario al BIT del código Gray en la siguiente posición adyacente. Descarte acarreos.

Ejemplo:
Paso 1. El dígito del código binario más a la izquierda es el mismo que el digito del código Gray más a la izquierda.

1	1	0	1	1	Gray
1					Binario

Paso 2. Sume el último BIT del código binario que se acaba de generar al BIT del código Gray en la siguiente posición. Descarte acarreos. (En negrilla BIT que se suman).

1	**1**	0	1	1	Gray
1	0				Binario

Paso 3. Sume el último BIT del código binario que se acaba de generar al BIT del código Gray en la siguiente posición.

1	1	**0**	1	1	Gray
1	**0**	0			Binario

Paso 4. Sume el último BIT del código binario que se acaba de generar al BIT del código Gray en la siguiente posición.

1	1	0	**1**	1	Gray
1	0	**0**	1		Binario

Paso 5. Sume el último BIT del código binario que se acaba de generar al BIT del código Gray en la siguiente posición. Descarte acarreos.

1	1	0	1	**1**	Gray
1	0	0	**1**	0	Binario

La conversión ha sido completada; el código binario es 10010

Ejercicios:

11000110B = Gray **R/=** 10100101

10101111G = Binario **R/=** 11001010

0101B = Gray **R/=** 0111

00111B = Gray **R/=** 00100

101011B = Gray **R/=** 111110

UNIDAD 2
CIRCUITOS DE MEDIANA ESCALA DE INTEGRACIÓN

RAP # 1: Diseña funciones lógicas a través de la simbología técnica.

RAP # 2: Analiza el comportamiento de circuitos lógicos a través de tablas de verdad.

RAP # 3: Construye circuitos lógicos, utilizando circuitos integrados de mediana escala de integración, apoyándose en hojas de especificaciones técnicas.

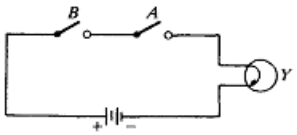
2.1 COMPUERTAS LÓGICAS
2.2 INTRODUCCIÓN

La compuerta lógica es el elemento básico en los sistemas digitales. Las compuertas lógicas operan con números binarios. Por esta razón, a las compuertas lógicas se les llama compuertas lógicas binarias. Todos los voltajes usados en las compuertas lógicas serán ALTO o BAJO, un ALTO voltaje significará un 1 binario y un BAJO voltaje significará un 0 binario. Estos circuitos electrónicos responderán sólo ALTOS voltajes (llamados 1 unos) ó BAJO (tierra) voltaje (llamado 0 ceros).
Todos los sistemas digitales se construyen usando sólo tres compuertas lógicas. A estas compuertas lógicas se les conoce como compuertas AND, OR y NOT.

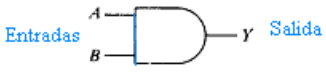
Integración a baja escala (SSI), son unos cuantos componentes los que se integran para formar un circuito completo. Como guía, SSI se refiere a los CI con menos de 12 componentes integrados. La mayoría de los chips SSI utilizan resistores, diodos y transistores bipolares integrados.
La integración a media escala (MSI) se refiere a los CI que tienen de 12 a 100 componentes integrados por chip. Transistores bipolares o transistores MOS (MOSFET en modo de enriquecimiento) se pueden emplear como transistores integrados de un CI. De nueva cuenta, la mayoría de los chips MSI utilizan componentes bipolares.
La importancia de conocer las compuertas lógicas es entender las operaciones básicas lógicas and, or, not. Para entender otras compuertas, es necesario conocer su símbolo, tabla de verdad y función de salida.

2.3 COMPUERTA AND:

A la compuerta AND se le llama la compuerta “todo o nada”. El esquema de la figura 2.1 la muestra la idea de la compuerta AND. La lámpara (Y) se encenderá sólo cuando ambos interruptores de entrada (A y B) están cerrados. En la figura 2.2 se muestran todas las posibles combinaciones para los interruptores A y B. A la tabla en esta figura se le llama tabla de verdad. La tabla de verdad muestra que la salida (Y) es habilitada sólo cuando ambas entradas estén cerradas.



a) Circuito AND usando conmutadores



a) Símbolo de la compuerta AND

Computadoras de entrada		Luz de salida
B	A	Y
abierto	abierto	no
abierto	cerrado	no
cerrado	abierto	no
cerrado	cerrado	si

b) Tabla de verdad
fig. 2.1

Entradas		Salida
B	A	Y
0	0	0
0	1	0
1	0	0
1	1	1

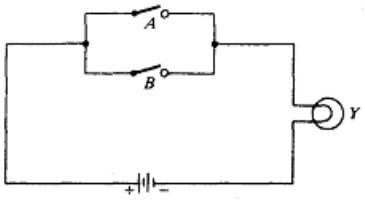
0 = bajo voltaje
1= alto voltaje

b) Tabla de verdad para AND
fig. 2.2

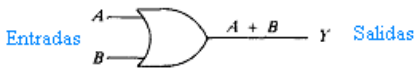
El símbolo de operación algebraico de la función AND es el mismo que el símbolo de la multiplicación de la aritmética ordinaria (·).
Las compuertas AND pueden tener más de dos entradas y por definición, la salida es 1 si todas las entradas son 1
A · B = y
En la figura 2.2 se muestra el símbolo lógico convencional de la compuerta AND. Éste símbolo señala las entradas como A y B. A la salida se le señala como Y. Éste es el símbolo para una compuerta AND de dos entradas. La tabla de verdad para la compuerta AND de dos entradas se muestra en la fig. 2.2b. Las entradas se representan como dígitos binarios (bit). Advierta que sólo cuando ambas entradas AyB son 1 la salida será 1.
El punto representa la función lógica AND en álgebra booleana, no la multiplicación como en el álgebra regular.

2.4 COMPUERTA OR:

A la compuerta OR se le llama compuerta de " cualquiera o todo" El esquema de la figura 2.3 a muestra la idea de la compuerta OR. La lámpara (Y) se encenderá cuando cualquier interruptor A ó B esté cerrado. La lámpara se encenderá cuando los dos interruptores A y B estén cerrados. La lámpara (Y) no se encenderá cuando ambos interruptores (A y B) se encuentren abiertos. Todas las posibles combinaciones de los interruptores se encuentran en la figura 2.3b. La tabla de verdad muestra en detalle la función OR del circuito de interruptor y lámpara. La salida del circuito OR estará habilitada cuando alguno o todos de los interruptores esté cerrado.



a) Circuito OR usando conmutador



a) Símbolo de la compuerta OR

Conmutadores de entrada		Salida luminosa
B	A	Y
abierto	abierto	no
abierto	cerrado	si
cerrado	abierto	si
cerrado	cerrado	si

b) Tabla de verdad
fig. 2.3

Entradas		Salidas
B	A	Y
0	0	0
0	1	1
1	0	1
1	1	1

0 = bajo voltaje
1= alto voltaje

b) Tabla de verdad para OR
fig. 2.4

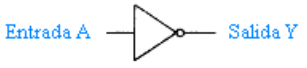
El símbolo lógico convencional para la compuerta OR se muestra en la figura 2.4 Note que la compuerta OR tiene diferente forma. La compuerta OR tiene dos entradas, llamadas A y B. A la salida se le llama Y. La expresión Booleana "taquigráfica" para esta función OR está dada por A + B = Y. Nótese que el signo (+) significa OR en álgebra booleana. La expresión (A + B = Y) se lee como A OR B igual a la salida Y. Note que el signo más no significa suma como en el álgebra regular.

La tabla de verdad para la compuerta OR de dos entradas se muestra en la figura 2.4b. Las variables de entrada (A y B) se muestran a la izquierda. La salida resultante se muestra en la columna de la derecha de la tabla.

La compuerta OR es habilitada (la salida es 1) cada vez que aparece un 1 en alguna o todas las entradas. Igual que anteriormente, un 0 se define como BAJO voltaje (tierra). Un 1 en la tabla de verdad representa ALTO voltaje (+ 5V).

2.5 COMPUERTA NOT:

A la compuerta NOT también se le conoce como un inversor. La compuerta NOT, o inversor, es una compuerta no usual. La compuerta NOT tiene solamente una entrada y una salida. En la figura 2.5 se muestra el símbolo lógico para el inversor o compuerta NOT.



a) Símbolo de la compuerta NOT

Entrada	Salida
A	Y
0	1
1	0

b) Tabla de la compuerta NOT

$A = \overline{A}$

c) Expresión booleana de NOT

fig. 2.5

El proceso de inversión es simple. La figura 2.5b muestra la tabla de verdad para la compuerta NOT. La entrada es cambiada por su opuesto. Si la entrada es 0, la

compuerta NOT dará su complemento u opuesto que es 1. Si la entrada en la compuerta NOT es 1, el circuito dará un 0. Esta inversión también se llama negación o complemento. Los términos complementación, negación e inversión, significan la misma cosa.

La expresión booleana para la inversión se muestra en la figura 2.5c La expresión A = A' se lee como A es igual a la salida no A. La barra ó coma sobre la A significa complemento de A.

2.6 COMPUERTA NAND:

Observe el diagrama de símbolos lógicos en la parte superior de la figura 2.6. Una compuerta AND se encuentra conectada a un inversor. A las entradas A y B se les aplica la operación AND para formar la expresión Booleana A · B. Este A · B se invierte después por acción de la compuerta NOT. A la derecha del inversor, se ve que se ha añadido la barra sobre la expresión booleana. La expresión booleana para el circuito completo es A · B = Y. Se dice que éste es un circuito no AND o circuito NAND.

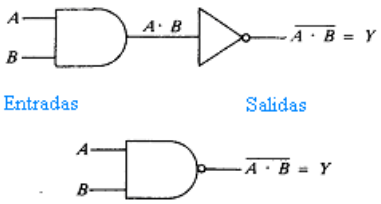


fig 2.6 La compuerta NAND

El símbolo lógico estándar para la compuerta NAND se muestra en la parte inferior del diagrama de la figura 2.6 Advierta que el símbolo NAND es un símbolo AND con un pequeño círculo en la salida. A este círculo se le denomina a veces círculo inversor. El círculo inversor es un método simplificado para representar a la compuerta NOT mostrada en la parte superior del diagrama de la figura 2.7.

Entradas		Salidas	
B	A	AND	NAND
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

fig. 2.7 Tablas de verdad para las compuertas AND y NAND

La tabla de verdad describe la operación exacta de una compuerta lógica. La tabla de verdad para la compuerta NAND se ilustra en las columnas de la figura 2.7 También se proporciona la tabla de verdad de la compuerta AND, para mostrar cómo se invierte cada salida para dar la salida NAND.

2.7 COMPUERTA NOR:

Considere el diagrama lógico de la figura 4-6. Se ha conectado un inversor a la salida de una compuerta OR. La expresión booleana a la entrada del inversor es A + B. Luego, el inversor complementa los términos a los que se aplicó el operador OR, mismos que se muestran en la expresión booleana con una barra arriba, esto es, A + B = Y. Ésta es una función no OR. La función no OR puede dibujarse con un sólo símbolo lógico, conocido como compuerta NOR. En el diagrama inferior de la figura 4-6, se ilustra el símbolo convencional para la compuerta NOR. Note que se añadió un círculo inversor al símbolo OR para formar el símbolo NOR.

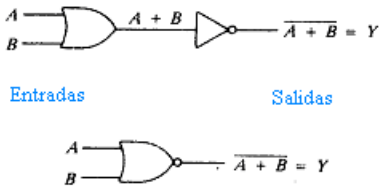


fig. 2.8 La compuerta NOR

La tabla de verdad de la figura 4-7 ilustra detalladamente la operación de la compuerta NOR. Ad vierta que la columna de la salida de la compuerta NOR es el complemento (se ha invertido) la columna sombreada OR. En otras palabras, la compuerta NOR pone un cero donde la compuerta OR hubiera puesto un 1. El círculo inversor en la salida del símbolo NOR sirve como un recordatorio de la idea de la salida O.

Entradas		Salidas	
B	A	OR	NOR
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

fig. 2.7 Tablas de verdad para las compuertas OR y NOT

2.8 COMPUERTA OR-EXCLUSIVA

A la compuerta OR exclusiva se le conoce como la compuerta “algunos pero no todos”. El término OR exclusivo se abrevia XOR. En la figura 2.8 se muestra una tabla de verdad para la función XOR. Se observa que si ambas entradas tienen el mismo valor las salidas serán 0 y si son diferentes la salida será 1. La compuerta XOR puede considerarse un circuito rectificador de bit impares.

Entradas		Salidas
B	A	XOR
0	0	0
0	1	1
1	0	1
1	1	0

fig.2.8 Tablas de verdad para las compuertas OR exclusiva

De la tabla de verdad de la figura 2.8 se puede desarrollar una expresión booleana para la compuerta XOR. La expresión sería $A \cdot B' + A' \cdot B = Y$. Con esta expresión Booleana puede desarrollarse un circuito lógico que utilice compuertas AND, OR e inversores. En la figura 2.9a se dibuja tal circuito. Este circuito lógico realizaría la función lógica XOR.

En la figura 2.9b muestra el símbolo lógico convencional para la compuerta XOR. Ambos diagramas de símbolos lógicos de la figura 2.9, producirían la misma tabla de verdad (XOR). La expresión booleana a la derecha de la figura 2.9b, es una expresión XOR simplificada. El símbolo representa a la función XOR en álgebra booleana.

a) Circuito lógico que realiza la función XOR b) Símbolo lógico estandar para la compuerta XOR

fig 2.9

2.9 LA COMPUERTA NOR EXCLUSIVA

En la figura 2.10 se invierte la salida de una compuerta XOR. A la salida del inversor de la derecha se le llama función NOR exclusivo (XNOR). La compuerta XOR produce la expresión $A \oplus B$ Al invertir ésta se forma la expresión booleana $(A \oplus B)' = Y$. Ésta es la expresión booleana para la compuerta XNOR. El símbolo lógico convencional para la compuerta XNOR se muestra en la parte inferior de la figura 2.10. Note que el símbolo es un XOR con un círculo inversor en la salida.

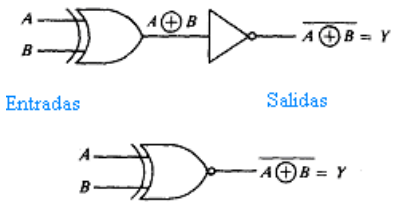


fig. 2.10 La compuerta XNOR.

Entradas		Salidas	
B	A	XOR	XNOR
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

fig. 2.11 Tablas de verdad para las compuertas XOR y XNOR

La columna derecha de la tabla de verdad de la figura 2.11 muestra detalladamente la operación de la compuerta XNOR. Advierta que todas las salidas de la compuerta

XNOR son los complementos de las salidas de la compuerta XOR. Mientras que la compuerta XOR es un detector de número impar de 1, la compuerta XNOR es un detector de par de bits iguales. La compuerta XNOR producirá una salida de 1 cuando en su entrada aparezca un número par de bits iguales..

2.10 COMPUERTA SEPARADOR (YES): O BUFFER

El símbolo en forma de triángulo se le designa como un circuito separador, el cual no produce ninguna función lógica particular, el valor binario de la salida es el mismo de la entrada.

Este circuito se utiliza para amplificación de la señal. Por ejemplo, un separador que utiliza 5 volt para el binario 1, producirá una salida de 5 volt cuando la entrada es 5 volt. Sin embargo, la corriente producida a la salida es muy superior a la corriente suministrada a la entrada de la misma.

De ésta manera, un separador puede excitar muchas otras compuertas que requieren una cantidad mayor de corriente.

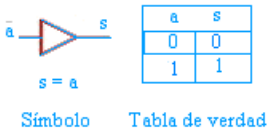
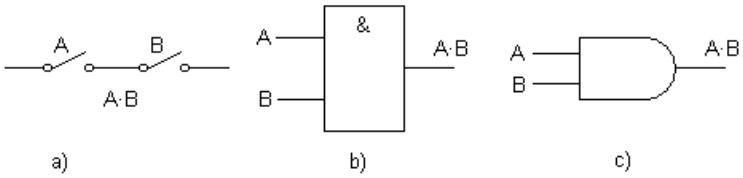
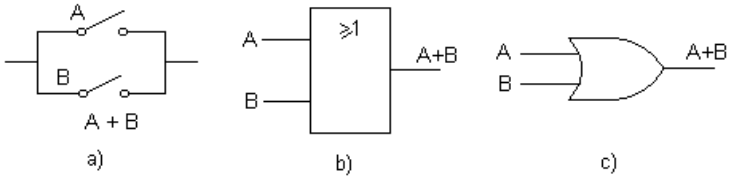


fig. 2.12 Buffer

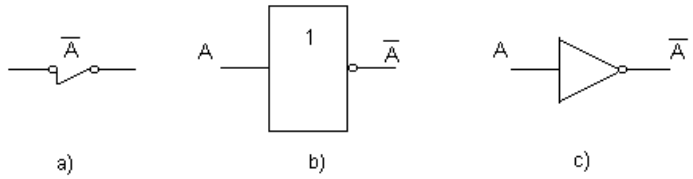
2,11 SÍMBOLOS



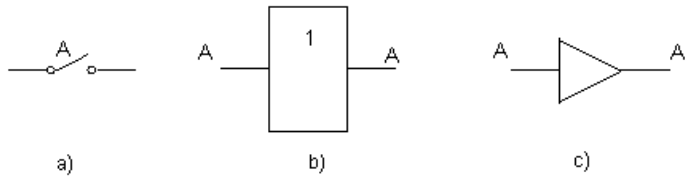
Símbolo de la función lógica Y a) Contactos, b) Normalizado y c) No normalizado



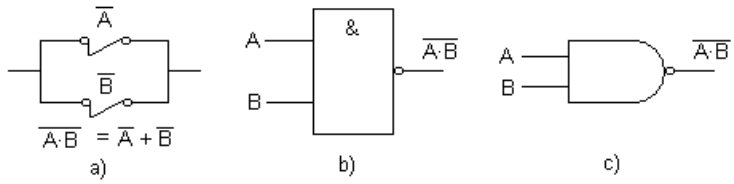
Símbolo de la función lógica O a) Contactos, b) Normalizado y c) No normalizado



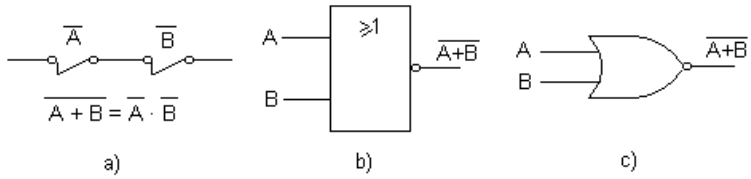
Símbolo de la función lógica NOT a) Contactos, b) Normalizado y c) No normalizada



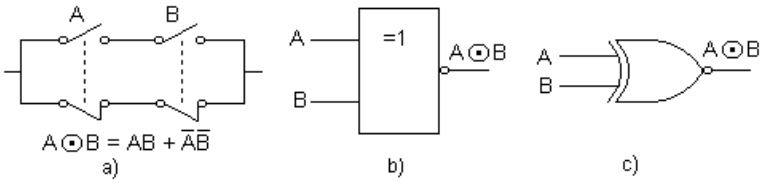
Símbolo de la función lógica SI a) Contactos, b) Normalizado y c) No normalizado



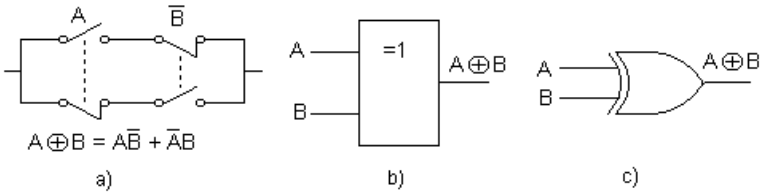
Símbolo de la función lógica NO-Y. a) Contactos, b) Normalizado y c) No normalizado



Símbolo de la función lógica NO-O. a) Contactos, b) Normalizado y c) No normalizado



Símbolo de la función lógica equivalencia. NOR exclusiva (XNOR): a) Contactos, b) Normalizado y c) No normalizado



Símbolo de la función lógica O-exclusiva. (XOR) a) Contactos, b) Normalizado y c) No normalizado.

2.12 FUNCION LÓGICA

Una función booleana es una expresión formada por variables binarias, los dos operadores binarios AND y OR y el operador unitario NOT, paréntesis y signo igual. Para un valor dado dé variables (ó combinación) la función puede ser 1 ó 0.

La forma de representar las funciones es por medio de tablas de verdad, para esto se necesitan 2^n combinaciones de $1^{'}s$ y $0^{'}s$ de las n variables binarias, y una columna que muestre aquellas combinaciones para las cuales la función es 1 ó 0.

Ejemplos : De funciones lógicas

$F_1 = x\,y\,z$

$F_2 = x + y^{'}\,z$

$F_3 = x^{'}\,y^{'}\,z^{'} + x^{'}\,y\,z + x\,y^{'}$

Pero estas expresiones algebraicas no son únicas, ya que como habíamos mencionado la manipulación del álgebra booleana es posible encontrar expresiones más simples para la misma función
Sea la función F_4 : $F_4 = x\,y^{'} + x^{'}\,z$ Parte de la explicación anterior y tabulemos la tabla de verdad de todas ellas:

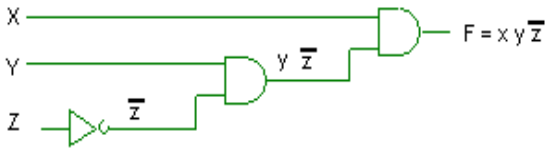
xyz	F_1	F_2	F_3	F_4
000	0	0	0	0
001	0	1	1	1
010	0	0	0	0
011	0	0	1	1
100	0	1	1	1
101	0	1	1	1
110	1	1	0	0
111	0	1	0	0

De aquí podemos notar que $F_3 = F_4$.
Una función booleana se puede transformar de una expresión algebraica en un diagrama lógico de compuertas AND, OR y NOT.

Podemos ver que F_4 requiere menos entradas y menos compuertas. Lo que constituye la mejor forma de una función booleana depende de la aplicación particular. Aquí tomaremos en consideración el criterio de minimización de compuertas.

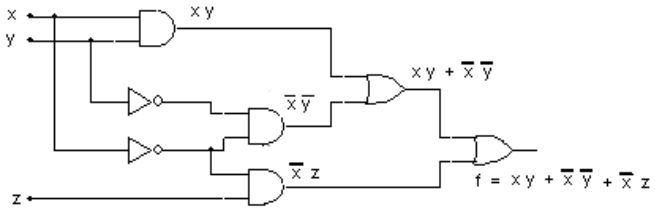
2.13 CONSTRUCCIÓN DE FUNCIONES LÓGICAS CON DIFERENTES ARREGLOS DE COMPUERTAS

¿Cual será la función f del siguiente circuito?



El resultado es: $f = x\ y\ \overline{z}$

¿Cual será la función f del siguiente circuito?



La función es: $f = x\ y + \overline{x}\ \overline{y} + \overline{x}\ z$

2.14 TABLAS DE VERDAD

La tabla de verdad es un instrumento utilizado para la simplificación de circuitos digitales a través de su ecuación booleana. Las tablas de verdad pueden tener muchas columnas, pero todas las tablas funcionan de igual forma.
Hay siempre una columna de salida (última columna a la derecha) que representa el resultado de todas las posibles combinaciones de las entradas.

Tabla de verdad	
Columna(s)de entrada	Columnas de salida
Entrada (interruptor)	Salida (lámpara)
Abierto	Apagado
Cerrado	Encendido

El número total de columnas en una tabla de verdad es la suma de las entradas que hay + 1 (la columna de la salida).
El número de filas de la tabla de verdad es la cantidad de combinaciones que se pueden lograr con las entradas y es igual a 2^n, donde n es el número de columnas de la tabla de verdad (sin tomar en cuenta la columna de salida)

Ejemplo: en la siguiente tabla de verdad hay 3 columnas de entrada, entonces habrán: 2^3 = 8 combinaciones (8 filas)

Un circuito con 3 interruptores de entrada (con estados binarios "0" o "1"), tendrá 8 posibles combinaciones. Siendo el resultado (la columna salida) determinado por el estado de los interruptores de entrada.

Tabla de verdad			
Switch 1	Switch 2	Switch 3	Salida
0	0	0	?
0	0	1	?
0	1	0	?
0	1	1	?
1	0	0	?
1	0	1	?
1	1	0	?
1	1	1	?

Los circuitos lógicos son básicamente un arreglo de interruptores, conocidos como "compuertas lógicas" (compuertas AND**,** NAND, OR, NOR, NOT**,** etc.). Cada compuerta lógica tiene su tabla de verdad.

Si pudiéramos ver con más detalle la construcción de las "compuertas lógicas", veríamos que son circuitos constituidos por transistores, resistencias, diodos, etc., conectados de manera que se obtienen salidas específicas para entradas específicas
La utilización extendida de las compuertas lógicas, simplifica el diseño y análisis de circuitos complejos. La tecnología moderna actual permite la construcción de circuitos integrados que se componen de miles (o millones) de compuertas lógicas.

2.15 RESUELVA PROBLEMAS DE CIRCUITOS LÓGICOS

1) Escriba la función lógica para una compuerta AND de cuatro entradas
Solución:

$$x \cdot y \cdot z \cdot w = S$$

b)Dibuje una tabla de verdad de tres entradas para una compuerta AND

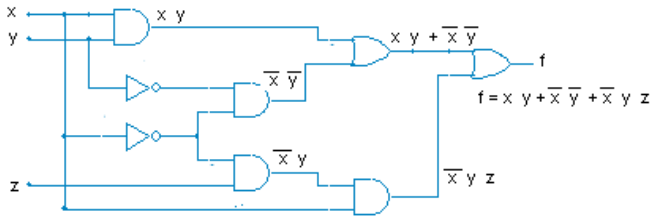
Entradas			Salidas
x	y	z	S
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

2) Partiendo del enunciado de un determinado problema, se tiene la siguiente expresión:

$F(x, y, z) = \bar{x}\,\bar{y} + x\,y + x\,\bar{y}\,z$

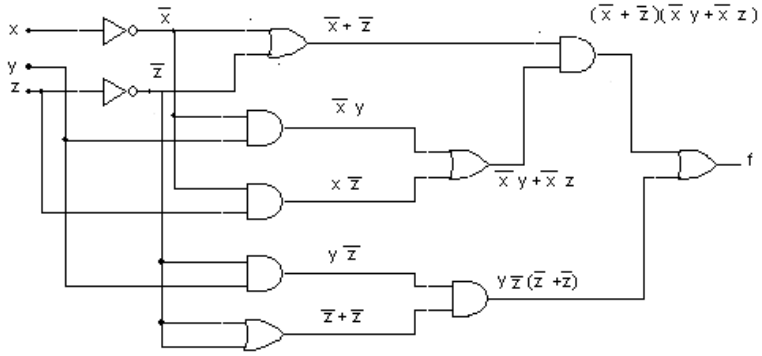
Deseamos obtener el diagrama del circuito lógico que realice esta función, las variables x,y,z serán las entradas del circuito y F será la salida, de la expresión observamos que se tienen 3 términos, cada uno de los cuales requiere de una compuerta Y (AND) , las dos primeras de 2 entradas y una tercera de 3 entradas. La salida de cada una de estas las compuertas es la entrada de una compuerta O (OR). A la salida de esta compuerta se tendrá la función de salida. Pero antes, por cada variable testada que se tenga, se requiere que ésta pase por un inversor ó también como se aprecia en el circuito, usando 4 compuertas AND de 2 entradas y una OR de 2 entradas.

Al diagrama lógico en estas notas le denominaremos "logigrama"
El logigrama que representa la función, queda de la siguiente forma.



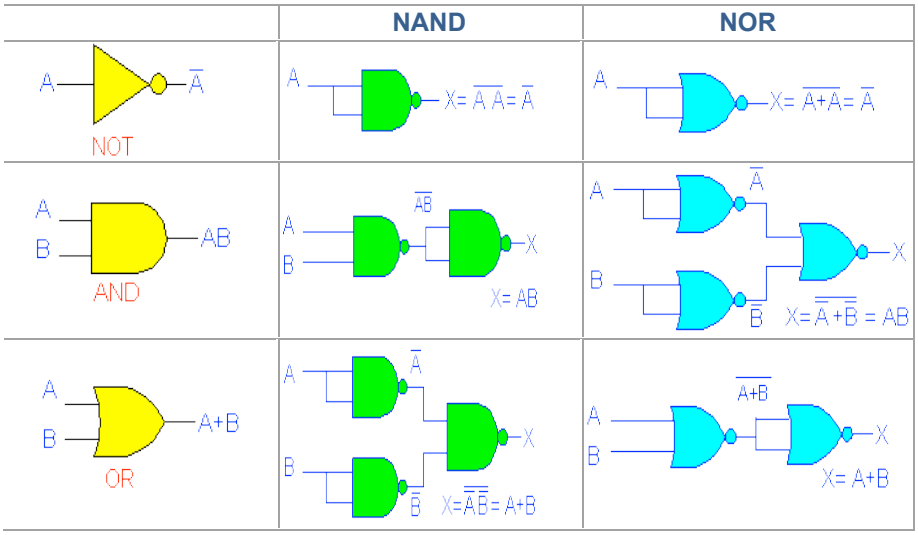
La función anterior es factible simplificarla aplicando postulados y teoremas.
3) Otro ejemplo, dibujar el logigrama de la siguiente función

$f/x,y,z) = (\bar{x} + \bar{z})(\bar{x}\,y + \bar{x}\,z) + y\,\bar{z}\,(\bar{z} + \bar{z})$



Universalidad de las compuertas NAND Y NOR

Estas compuertas se dicen que son "universales" puesto que con cada una de las dos familias podemos realizar todas las funciones lógicas.
En la tabla a continuación se muestran los operadores lógicos en función de sólo compuertas NOR y sólo compuertas NAND.



2.16 CARACTERÍSTICAS Y FUNCIONAMIENTO DE COMPUERTAS LÓGICAS

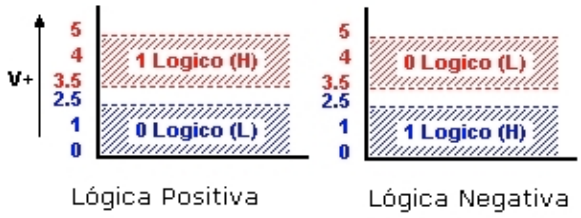
En la literatura de los fabricantes de CI aparecen algunos términos que ayudan al técnico al usar o comparar las familias lógicas. Se esquematizarán algunos de los términos y sus características más importantes usados en los CI digitales.

En la figura se observa una compuerta nor de la familia TTL de CI. Los fabricantes especifican que para una operación adecuada, una entrada, BAJO, debe tomar valores entre tierra y 0.8 V. De la misma manera, una entrada ALTO debe encontrarse entre 2.0 V y 5.0 V

Todos los sistemas digitales funcionan de manera binaria, los voltajes de entrada y salida son (dependiendo de su valor), separados en tres bloques:

Estado ALTO (1) Entre 2 y 5V, Suponiendo que la alimentación es de 5V.
Estado BAJO (0) Entre 0 y 0.8V, Suponiendo que la alimentación es de 5V.
Estado Indefinido (Cualquier voltaje entre 0.9 y 1.99V).

(Estos valores pueden variar dependiendo la tecnología utilizada en las compuertas)



Existen compuertas lógicas con más de dos entradas

Existen otras que tienen una entrada extra, es como un swich que al desconectar minimizan el consumo de energía (entrada de inhibición). A estas se les llama Compuertas con salida de tres estados.

A la salida de las compuertas sería para una salida BAJO sería 0.1 normalmente, pero puede llegar a 0.4 V. Una salida ALTO normal sería 3.5 V pero puede llegar a ser tan baja como 2.4 V. La salida ALTO depende del valor de la resistencia de carga en la salida. Entre mayor sea la corriente de carga, menor es el voltaje de salida ALTO. La parte no sombreada del voltaje de salida es la región prohibida.

En la tecnología CMOS una nivel lógico de "0", será interpretado como tal, mientras el valor de voltaje de la salida esté entre 0V. y 1.5V

Un voltaje de entrada nivel alto se denomina VIH
Un voltaje de entrada nivel bajo se denomina VIL
Un voltaje de salida nivel alto se denomina VOH
Un voltaje de salida nivel bajo se denomina VOL

Además de los niveles de voltaje, también hay que tomar en cuenta, las corrientes presentes a la entrada y salida de las compuertas digitales.

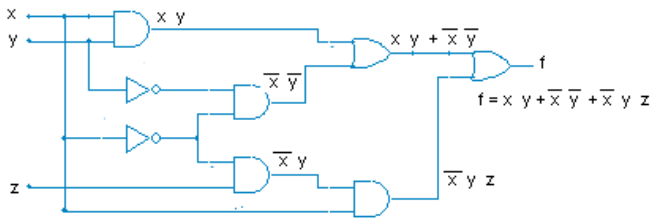
La corriente de entrada nivel alto se denomina: IIH
La corriente de entrada nivel bajo se denomina IIL
La corriente de salida nivel alto se denomina: IOH
La corriente de salida nivel bajo se denomina IOL

2.17 CONSTRUCCIÓN DE CIRCUITOS LÓGICOS

Construya el siguiente circuito lógico con compuertas AND, OR y NOT.
Partiendo del enunciado de un determinado problema, se tiene la siguiente expresión:

$F(x, y, z) = \bar{x} \cdot \bar{y} + x\,' y + \bar{x} \cdot y \cdot z$

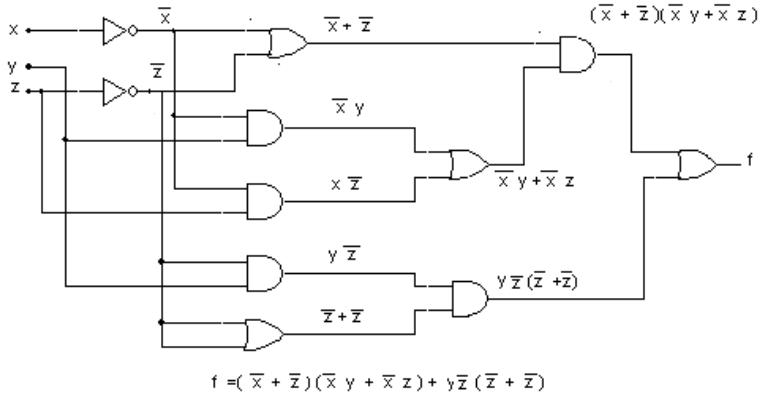
Deseamos obtener el diagrama del circuito lógico que realice esta función, las variables x,y,z serán las entradas del circuito y F será la salida, de la expresión observamos que se tienen 3 términos, cada uno de los cuales requiere de una compuerta Y (AND) , las dos primeras de 2 entradas y una tercera de 3 entradas. La salida de cada una de estas las compuertas es la entrada de una compuerta O (OR). A la salida de esta compuerta se tendrá la función de salida. Pero antes, por cada variable testada que se tenga, se requiere que ésta pase por un inversor ó también como se aprecia en el circuito, usando 4 compuertas AND de 2 entradas y una OR de 2 entradas.
Al diagrama lógico en estas notas le denominaremos "logigrama"
El logigrama que representa la función, queda de la siguiente forma.



La función anterior es factible simplificarla aplicando postulados y teoremas.

Otro ejemplo, construya la siguiente función

$$f/x,y,z) = (\bar{x} + \bar{z})(x\ y + \bar{x}\ z) + \bar{y}\ \bar{z}(z + \bar{z})$$



Para alimentar a los circuitos integrados de las siguientes figuras, la fuente de alimentación se conecta el positivo en la terminal 14 y el negativo en la terminal 7

UNIDAD 3

ALGEBRA DE BOOLEANA Y MAPAS DE KARNAUGH

RAP # 1: Correlaciona postulados y teoremas del algebra de Boole para simplificar funciones lógicas

RAP # 2: Aplica los mapas de Karnaugh para la obtención de la expresión mínima de funciones lógicas.

3.1 INTRODUCCIÓN

Es un sistema deductivo, puede definirse como un conjunto de elementos, un conjunto de operadores y un número de axiomas no probados o postulado. Se llama álgebra de Boole debido a George Boole quien lo desarrollo a mediados del año 1800. En 1938 C. E. Shanon introdujo un álgebra booleana de dos valores denominada álgebra de interruptores, en donde demostró que las propiedades de los circuitos eléctricos y estables con interruptores pueden representarse con esta álgebra. Para la definición formal de álgebra booleana se utilizan los postulados formulados por E. V. Huntiton en 1904.

Un conjunto de elementos es una colección de objetos que tienen una propiedad común. Si B es un conjunto y *x* y *y* son objetos ciertos, entonces $x \in B$ denota que *x* es un miembro del conjunto B y $y \in B$ denota que y no es un elemento de B. Un conjunto con un número finito de elementos se representa por medio de llaves: A= {1, 2, 3, 4} es decir los elementos del conjunto A son los números 1, 2, 3 y 4. Un operador binario definido en un conjunto B de elementos, es una regla que asigna a cada par de elementos de B un elemento único de B. Por ejemplo, considérese la relación $a \cdot b = c$. Se dice que $\cdot$ es un operador binario si éste especifica una regla para encontrar c de un par (a, b) y también si a, b, $c \in B$. Por otra parte, $\cdot$ no es un operador binario si a, $b \in B$ mientras que la regla encuentra que $c \in B$.

3.1 POSTULADOS

Como cualquier sistema matemático deductivo se puede definir como un conjunto de elementos, un conjunto de operadores y un numero de axiomas no probados o postulados, mediante los cuales es posible deducir las reglas, teoremas, y propiedades del sistema, dentro de un conjunto de elementos B, junto con dos operadores binarios + y $\cdot$, siempre que satisfagan los siguientes postulados (de Huntington)

1. a) Conjunto cerrado con respecto al operador +. Se define una regla de combinación "+", en tal forma que el resultado (x + y) siga perteneciendo a B, siempre que tanto x como y estén en B.

1. b) Conjunto cerrado con respecto al operador $\cdot$. Se define una regla de combinación "$\cdot$", en tal forma que el resultado $x \cdot y$) siga perteneciendo a B, siempre que tanto x como y estén en B.

2. a) Elemento identidad con respecto a +, designado por 0, tal que: $x + 0 = x$

2. b) Elemento identidad con respecto a $\cdot$, designado por 1, tal que: $x \cdot 1 = x$

3 a) Conmutativo con respecto a + : $x + y = y + x$.

3. b) Conmutativo con respecto a $\cdot$: $x \cdot y = y \cdot x$.

4. a) es distributivo con respecto a + : $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$

4. b) + es distributivo con respecto a · : x + (y · z) = (x + y) · (x + z)

5. Para cada elemento x ϵ B, existe un elemento x' ϵ B (denominado complemento de x), tal que:

a) x + x' = 1 y b) x · x' = 0

6. Existen cuando menos dos elementos x, y ϵ B, tales que: x = / = y

Se pueden ver diferencias entre esta álgebra y álgebra ordinaria, como en el postulado 4 (b) , que no es valido en el algebra de los números reales; en el álgebra. booleana no hay sustracción o división porque no tiene inverso aditivo o multiplicativo; y vemos que el operador complemento no se encuentra en el álgebra ordinaria

Aun cuando puedan formularse muchas álgebras booleanas dependiendo de la elección de los elementos y las reglas de operación, el álgebra booleana ó de la lógica es de sólo 2 valores (ó elementos del conjunto B), con las reglas para dos operadores binarios + y · definidas en las tablas de verdad de AND y OR, y un operador complemento equivalente al operador NOT.

DUALIDAD

Puede verse en 10 postulados de Huntington, que se listaron en pares, ya que una parte puede obtenerse de la otra si se intercambian los operadores binarios y los elementos identidad. Esta propiedad se llama principio de dualidad y establece que si una función es verdadera, entonces su dual también lo es.

Se listarán teoremas básicos en el álgebra de Boole, al igual que los postulados se listan en pares, cada relación es el dual de su pareja. Los postulados son axiomas básicos de la estructura algebraica y no necesitan prueba. Los teoremas deben probarse mediante los postulados, ejemplo:

Teorema 1 a) x + x = x

x + x = (x + x) 1	; por el postulado 2 (b)
= (x + x) (x+ x')	; por el postulado 5 (a)
= x + xx'	; por el postulado 4 (b)
= x + 0	; por el postulado 5 (b)
= x	; por el postulado 2 (a)

En el caso del teorema 1 b) se puede derivar en forma dual de esta prueba.

TEOREMAS BÁSICOS DEL ÁLGEBRA DE BOOLEANA

T1. a) x + x = x b) x · x = x
T2. a) x + 1 = 1 b) x +0 = 0
T3. de INVOLUCIÓN a) (x')' x
T4. ASOCIATIVO a) x + y + z = (x + y) + z = x + (y + z) b) x · y · z = (x· y) · z = x · (y·z)
T5. de de MORGAN a) (x + y)' = x' · y' b) (x · y)' = x' + y'
76. de ABSORCIÓN a) x + (x · y) = x
T7. a) y · (x + y') = x · y b) y' + (x · y) = x + y'
T8. a) (x + y) · (x' + z) · (y + z) = (x + y) · (x' + z)
b) (x · y) + (x'+ z) +(y · z) = (x · y)+ (x' · z)

T9. $(x + y) \cdot (x' + z) = x \cdot z + x' \cdot y$

3.3 SIMPLIFICACIÓN DE FUNCIONES

El número de las literales puede minimizarse por operaciones algebraicas, como en los siguientes ejercicios.

1) $x + xyz + xy + x' yz = x + xy + x' yz$;por el T6 de absorción
$x + xy + x' yz$;por el T6 de absorcion
$= x + x' yz$;por el T7 (b) $y' + xy = x + y'$

2) $x+x'y = x+x'y$:por el p4 (b) distributivo
$= (x + x')(x +y)$;por el p5 (a) complemento
$= 1 \cdot (x + y)$;por el p2 (b) elemento identidad
$= x + y$

3) $x(x' + y) = x(x' + y)$; por el P4 (a) distributivo
$= xx' + xy)$; por e1 P5 (b) complementos
$= 0 + xy$; por el P2 (a) elemento identidad
$= xy$

4) $x' y'z + x'yz + xy' = x'y'z + x'yz + xy'$; aplicando el P4(a)
$= x'z(y' + y) + xy'$; por el P5(a) complementos
$= x'z + x y'$

5) $xv + x'z + yz = xy + x'z + yz \cdot 1$; por el P2 (b) y el P5 (a)
$= xv + x'z + vz(x + x')$; por el P4 (a) distributivo
$= xy + x'z + xyz + x'yz$; aplicando el P4 (a)
$= :xy (1+z) +x'z(1+y)$; por el T2(a)
$= xv+x'z$

6) $(x + y)(x' + z)(y + z) = (x + y)(x' +z)$;por la dualidad de la función 5.

Vemos en las funciones de los incisos 2 y 3, que son duales una de la otra y que se utilizan expresiones duales en los pasos correspondientes. La función del inciso 4 nuestra la igualdad de las funciones Fa y F,, mostradas con anterioridad. En el Inciso 5 se nuestra que un incremento en el número de literales puede, alguna veces, conducir a una expresión final más simple. En el inciso 6 no se hace en forma directa, se utiliza el principio de dualidad.

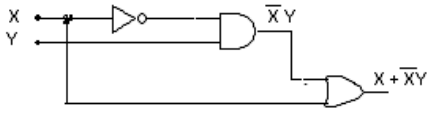
7) $x + x.y.z + x.y + x^i y.z = x + x.y.z + x.y + x^i y.z$ (t 4b)
$= x + xy + x^i yz$ (t 4b)
$= x + x^i yz$ (t 14a)
$= yz + x$

8) $x + x^i y = (x + x^i)(x + y)$ (p 5a)
$= 1.\ (x + y)$ (p 6a)
$= (x + y)$ (p 3b)

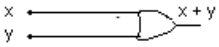
9) $x^i y^i z + x^i y z + x y^i = x^i z (y^i + y) + x y^i$ (p 5b)
$= x^i z.\ 1 + x y^i$ (p 3b)
$= x^i z + x y^i$

3.4CIRCUITOS LÓGICOS SIMPLIFICADOS

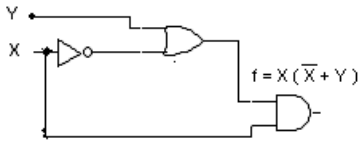
1) $X + X^I.Y = X + Y$



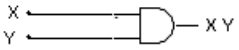
CIRCUITO SIMPLIFICADO



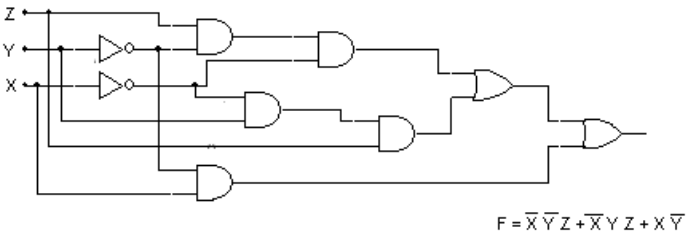
2) $X(X^I + Y) = XY$

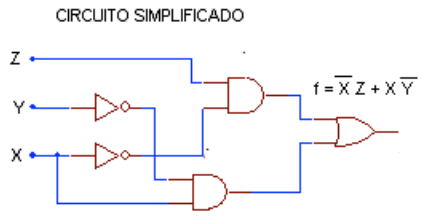


CIRCUITO SIMPLIFICADO



3) $X^I.Y^I.Z + X^I.Y.Z + X.Y^I = X^I.Z + X.Y^I$





3.5 MINTÉRMINOS Y MAXTÉRMINOS

Los métodos de diseño y simplificación de circuitos lógicos requieren que la expresión lógica esté en forma de suma de productos. Algunos ejemplos de ésta forma son:

1. xyz + x' y z'
2. xy + xyz + zw + w
3. xy + zw' + yz' w

Cada una de estas expresiones de suma de productos consta de dos o más términos AND (productos) que se operan con OR juntos. Cada término AND consta de una o más variables que aparecen en forma complementada o no complementada. Por ejemplo, en la expresión de suma de productos xyz + x' y z', el primer producto AND tiene las variables x, y y z en su forma no complementada (no invertida). El segundo término AND tiene a x y a z en su forma complementada (invertida). Nótese que en una expresión con suma de productos, un signo de inversión no puede aparecer en más de una variable en un término (por ejemplo, no podemos tener (xyz)')

3.6 MAXTÉRMINOS es el complemento de su Mintérmino.

Minterminos Maxterminos para tres variables binarias

			Mintermino		Maxtermino	
X	Y	Z	Termino	Designacion	Termino	Designacion
0	0	0	$X^I\ Y^I\ Z^I$	m_0	X+Y+Z	M_0
0	0	1	$X^I\ Y^I\ Z$	m_1	$X+Y+Z^I$	M_1
0	1	0	$X^I\ Y\ Z^I$	m_2	$X+Y^I+Z$	M_2
0	1	1	$X^I\ Y\ Z$	m_3	$X+Y^I+Z^I$	M_3
1	0	0	$X\ Y^I\ Z^I$	m_4	X^I+Y+Z	M_4
1	0	1	$X\ Y^I\ Z$	m_5	X^I+Y+Z^I	M_5
1	1	0	$X\ Y\ Z^I$	m_6	X^I+Y^I+Z	M_6
1	1	1	X Y Z	m_7	$X^I+Y^I+Z^I$	M_7

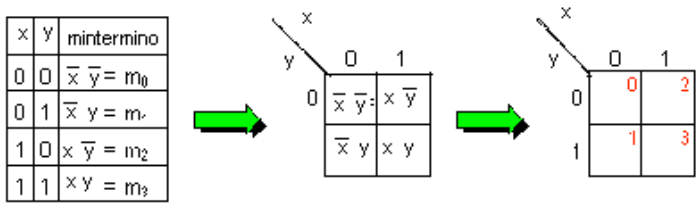
3-7 MAPAS DE KARNAUGH

El álgebra booleana es la base para cualquier simplificación de circuitos lógicos. Una de las técnicas más fáciles de usar para simplificar circuitos lógicos es el método de mapas de Karnaugh. Este método gráfico se basa en los teoremas booleanos, y sólo es uno de varios métodos utilizados por los diseñadores lógicos para simplificar circuitos lógicos.

Mapas de Karnaugh de 2 variables

Sea f una función de 2 variables f (x,y)
Para elaborar el mapa de Karnaugh se debe de tener 2^n cuadrados para n variables, en el caso de 2 variables, tendremos $2^2 = 4$ cuadrados, en la figura se muestra la tabla de verdad con la lista de los mintérminos y el lugar que ocupa cada uno de ellos en un mapa.

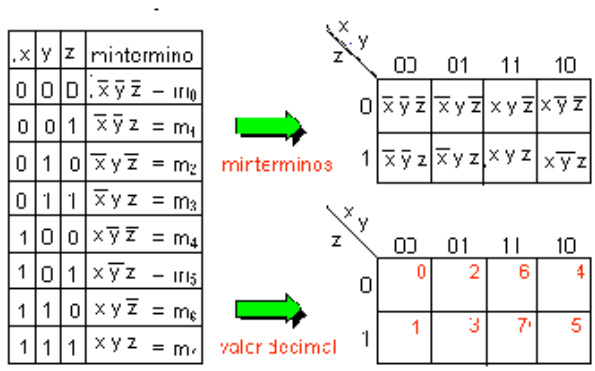
x y mintérmino
0 0 $x' y' = m_0$
0 1 $x' y = m_1$
1 0 $x y' = m_2$
1 1 $x y = m_3$



La función de este mapa de karnaugh es: $f = x' y' + x' y + x' y + x' y$

Mapas de Karnaugh de 3 variables
Sea f una función de 3 variables (x, y, c)

Para elaborar el mapa de Karnaugh tendremos $2^3 = 8$ combinaciones. Al igual que antes cada casilla del mapa corresponde a un mintérmino de la tabla de verdad.
Es importante colocar las variables en el orden indicado de más significativo a menos significativo (x, y, c) de otra forma el valor decimal de las casilla sería diferente.



Note que en las columnas x y no se sigue el orden progresivo de valores, 00, 01, 10,11, si no 00, 01, 11,10

Es muy importante, que el proceso de minimización depende de la ubicación de las casillas en el mapa de k. Esto se hace para que entre una casilla y otra, en forma horizontal o vertical sólo cambie una variable, lo que llamamos ADYACENCIA LÓGICA.

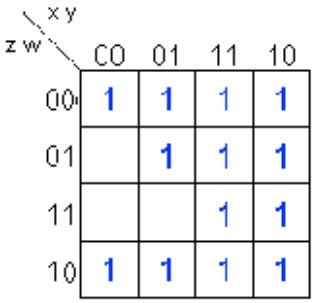
Por ejemplo la casilla 2 (010) es adyacente a las casillas 0 (000) (cambia y), a la 3 (011) (cambia z)y a la 6 (110) (cambia x).
¿Cuales son las casillas adyacentes a la casilla 4? Note que además de la 6 y la 5 también es adyacente a la 0 (entre 100 (4) y 000 (0) cambia A)

Mapa de cuatro variables
Sea f una función de 4variables (x,y,z,w)

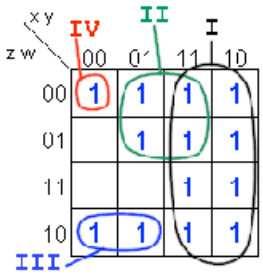
Para este caso el mapa de Karnaugh es de 2^4= 16 combinaciones. cada casilla del mapa corresponde a un mintérmino de la tabla de verdad.

x y z w	Mintérmino
0 0 0 0	1= m_0
0 0 0 1	0= m_1
0 0 1 0	1= m_2
0 0 1 1	0= m_3
0 1 0 0	1= m_4
0 1 0 1	1= m_5
0 1 1 0	1= m_6
0 1 1 1	0= m_7
1 0 0 0	1= m_8
1 0 0 1	1= m_9
1 0 1 0	1= m_{10}
1 0 1 1	1= m_{11}
1 1 0 0	1= m_{12}
1 1 0 1	1= m_{13}
1 1 1 0	1= m_{14}
1 1 1 1	1= m_{15}

La función está expresada en forma canónica, por lo que cada mintérmino "colocará" un 1 en su casilla correspondiente como se muestra en el mapa de Karnaugh ó 0 según el caso.



Ahora agrupar los "unos" del mapa de Karnaugh como se muestra en la figura, Según esto tenemos cuatro grupos que son :



grupo I x (agrupa 8 unos y es de 1 variable)

grupo II $y\bar{x}$ (agrupa 4 unos y es de 2 variables)

grupo III $\bar{x}y\bar{z}$ (agrupa 2 unos y es de 3 variables)

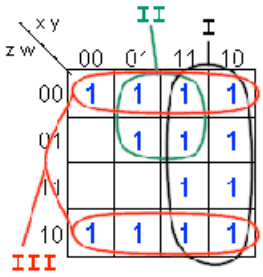
grupo IV $\bar{x}\bar{y}\bar{z}\bar{w}$ (agrupa 1 uno y es de 4 variables)

Bajo este agrupamiento de unos la función queda sin reducirse:

$f = x' y' z' w' + x' y z' + y z' + x$

Puede verse que a medida que agrupamos mayor cantidad de "unos", el término tiene menos literales. El agrupamiento se hace con una cantidad de "unos" que son potencias de 2. Así agrupamos 2 mintérminos, 4 mintérminos y 8 mintérminos. Cada vez que aumentamos, el término va eliminando una variable. En una función de 4 variables, un término que tenga un sólo "uno" tendrá las cuatro variables. De hecho es un término canónico. Al agrupar dos mintérminos eliminaremos una variable y el término quedará de tres variables. Si agrupamos cuatro "unos" eliminaremos dos variable quedando un término de dos variables y finalmente si agrupamos ocho "unos" se eliminaran tres variables para quedar un término de una variable.

Todo esto se debe a la adyacencia entre casillas y cada vez que agrupamos, se eliminan las variables que se complementan.



Finalmente la función mínima es: $f = x + w' + y z'$

Es importante que al "tomar" un uno, se agrupe con todos los unos adyacentes, aunque estos unos sean parte de otros grupos. El mintérmino 13 (1100_2) es común a los tres términos.

Al combinar las casillas en un mapa de Karnaugh, agruparemos un número de mintérminos que sea potencia de dos. Así agrupar dos casillas eliminamos una variable, al agrupar cuatro casillas eliminamos dos variables, y así sucesivamente. En general, al agrupar 2^n casillas eliminamos n variables.

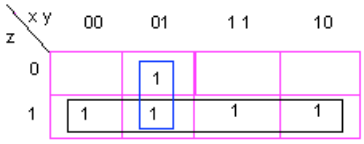
Debemos agrupar tantas casillas como sea posible; cuanto mayor sea el grupo, el término producto resultante tendrá menos literales. Es importante incluir todos los "unos" adyacentes a un mintérmino que sea igual a uno.

Para que haya menos términos en la función simplificada, debemos formar el menor número de grupos posibles que cubran todas las casillas (mintérminos) que sean iguales a uno. Un "uno" puede ser utilizado por varios grupos, no importa si los grupos se solapan. Lo importante es que si un grupo está incluido completamente en otro grupo, o sus "unos" están cubiertos por otros grupos, no hace falta incluirlo como término.

Ejemplo 1: De la siguiente tabla de verdad, obtener la función simplificada.

x y z	Mintérmino
0 0 0	0= m_0
0 0 1	1= m_1
0 1 0	1= m_2
0 1 1	1= m_3
1 0 0	0= m_4
1 0 1	1= m_5
1 1 0	0= m_6
1 1 1	1= m_7

Colocar unos en el mapa



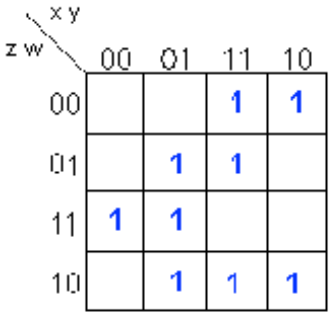
La función simplificada es:

$f = z + x' y$

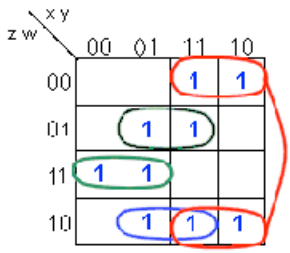
Ejemplo 2: Simplificar la siguiente tabla de verdad

x y z w	Mintérmino
0 0 0 0	0= m_0
0 0 0 1	0= m_1
0 0 1 0	0= m_2
0 0 1 1	0= m_3
0 1 0 0	1= m_4
0 1 0 1	1= m_5
0 1 1 0	1= m_6
0 1 1 1	1= m_7
1 0 0 0	1= m_8
1 0 0 1	0= m_9
1 0 1 0	1= m_{10}
1 0 1 1	0= m_{11}
1 1 0 0	1= m_{12}
1 1 0 1	1= m_{13}
1 1 1 0	1= m_{14}
1 1 1 1	0= m_{15}

Colocar unos en el mapa de Karnaugh



Hacer grupos de unos



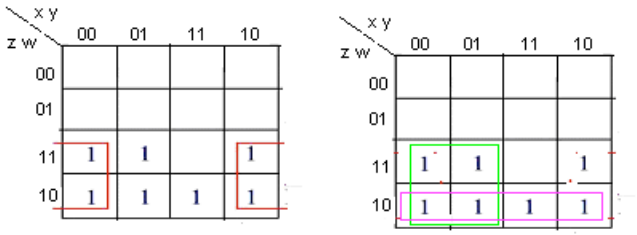
Función simplificada

$f = y z' w + x w' + x' z w + y z w'$

Ejemplo 3 : De la siguiente tabla de verdad por medio de mapas de karnaugh obtener la función simplificada.

x y z w	Mintérmino
0 0 0 0	0= m_0
0 0 0 1	0= m_1
0 0 1 0	0= m_2
0 0 1 1	0= m_3
0 1 0 0	1= m_4
0 1 0 1	1= m_5
0 1 1 0	1= m_6
0 1 1 1	1= m_7
1 0 0 0	1= m_8
1 0 0 1	0= m_9
1 0 1 0	1= m_{10}
1 0 1 1	0= m_{11}
1 1 0 0	1= m_{12}
1 1 0 1	1= m_{13}
1 1 1 0	1= m_{14}
1 1 1 1	0= m_{15}

Colocar unos en el mapa



Agrupar unos en el mapa, obtener la función

La función simplifica queda así:

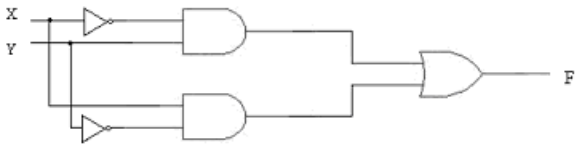
$f = x y' + y' z' + y' w'$

3.8 CIRCUITOS LÓGICOS SIMPLIFICADOS POR MAPAS DE KARNAUGH

1). Simplifique la siguiente función por mapas de karnaugh

$f = x' y z + x' y z' + x y' z' + x y' z$

Ahora aplicando las compuertas lógicas (en este caso tenemos a x' y que están multiplicando y sumando a la ves con x y') entonces la función. Simplificada queda así.
1) f= x' y + x y'

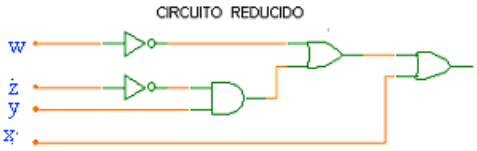


circuito reducido de:
f = x' y z + x' y z' + x y' z' + x y' z

2). Mostrar el circuito minimizado por mapas de karnaugh de la siguiente función:

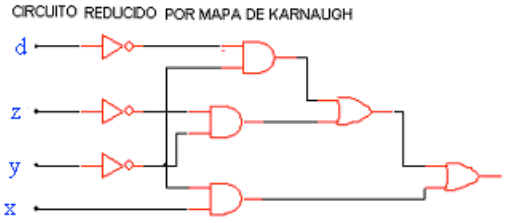
f = x' y' z' w' + x' y z' w + x y z' w'+ x y' z' w'+ x' y z' w + x y z' w + x y' z' w + x y z w + x y' z w + x' y' z' w' + x' y z w' + x y z w' + x y' z w'

función reducida anteriormente por mapas es:
f = x+ w' +y z'



3). Mostrar el circuito minimizado de la siguiente función:
F= x y' z' w + x y' z w + x' y' z' w + x y' z' w + x y' z w' + x' y' z w' + x' y' z' w'

Función minimizada anteriormente por mapas de k es:
f = x y' + y' z' + y' w' su circuito es:



UNIDAD 4

APLICACIÓN DE LOS CIRCUITOS DE MEDIANA ESCALA DE INTEGRACIÓN

RAP # 1: Aplica los codificadores y decodificadores que requiere para resolver una necesidad detectada en su entorno.

RAP # 2: Resuelve problemas elementales de su entorno mediante la aplicación de multiplexores y demultiplexores.

RAP # 1: Demuestra el funcionamiento del sumador y restador para realizar operaciones de dos números de al menos 4 bits cada uno.

4-1 INTRODUCCIÓN

Una aplicación de las compuertas lógicas en sistemas digitales sería la de convertidores de código. Los códigos comúnmente utilizados son el binario, BCD (8421), octal, hexadecimal y, por supuesto, el decimal. Mucho del "misterio" que rodea a las computadoras y a otros sistemas digitales proviene del lenguaje poco conocido de los circuitos digitales. Los dispositivos digitales sólo pueden procesar los bit O y 1, sin embargo, para los seres humanos es difícil entender cadenas muy largas de ceros y unos. Por esta razón son necesarios los convertidores de código para convertir el lenguaje humano a lenguaje de máquina.

4.2 ESTABLECER RELACIONES EN LA CONSTRUCCIÓN FUNCIONAMIENTO Y APLICACIÓN

Considérese el diagrama de bloque de una calculadora de mano, como el de la figura 4-1. El sistema• de entrada a la izquierda es el conjunto de teclas. Entre este conjunto y la unidad del procesador central (CPU) de la calculadora existe un codificador que traduce el número decimal de la tecla oprimida a un código binario, tal como el BCD (8421). El CPU ejecuta la operación en binario y produce el resultado en código binario. El decodificador traduce del código binario de la CPU a un código especial que ilumina los segmentos apropiados en la pantalla de siete segmentos. Es decir, que el decodificador traduce de binario a decimal. En este sistema, el codificador y el decodificador son traductores electrónicos de código. El codificador se puede pensar como un traductor de lenguaje humano a lenguaje de máquina, mientras que el decodificador hace lo contrarío: traduce de lenguaje de máquina a lenguaje humano.

4.3 CODIFICADORES Y DECODIFICADORES

El codificador se puede pensar como un traductor de lenguaje humano a lenguaje de máquina, mientras que el decodificador hace lo contrario: traduce de lenguaje de máquina a lenguaje humano.

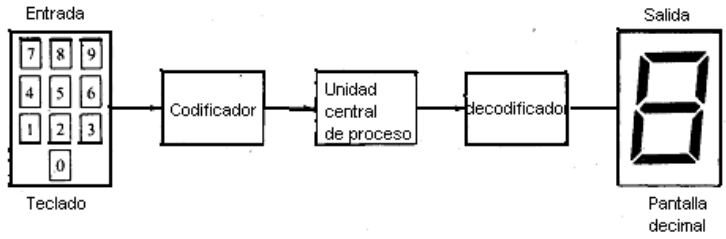


Fig. 4.1diagrama a bloque básico de una calculadora

El trabajo del codificador en una calculadora es traducir de una entrada decimal a un número en BCD 8421. La forma simplificada de un diagrama lógico para un codificador decimal a BCD se muestra la figura 4.2 El codificador tiene diez entradas a la izquierda y cuatro salidas a la derecha.

El codificador puede tener sólo una entrada activa que a su vez produce una salida única. En la figura 4.2 se activa la entrada decimal 7 que produce la salida 0111 en BCD, como lo muestra el indicador de la derecha.

En la figura 4.3a se muestra un diagrama de bloque para un codificador comercial decimal a BCD.

Un hecho fuera de lo común, son los círculos en las entradas y salidas. Los de las entradas significan que están activadas por el cero lógico ó BAJO, y los de las salidas significan que las salidas están normalmente en ALTO, o en 1 lógico, pero cuando se activan pasan a ser BAJO, o al O lógico. Se agregan cuatro invertidores al circuito para invertir la salida a su forma normal. Otro hecho poco común de este codificador es que no existe la entrada cero. La entrada decimal cero significa la salida 1111 (en D, C, B yA), que es verdadera cuando todas las entradas (1-9) no están conectadas. Cuando esto sucede se dice que las entradas están flotando. En este caso están flotando en ALTO.

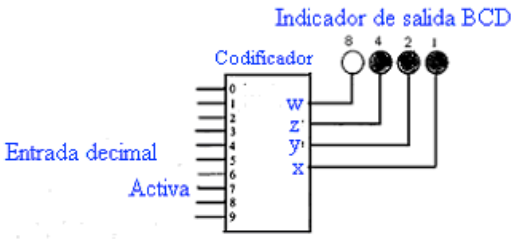
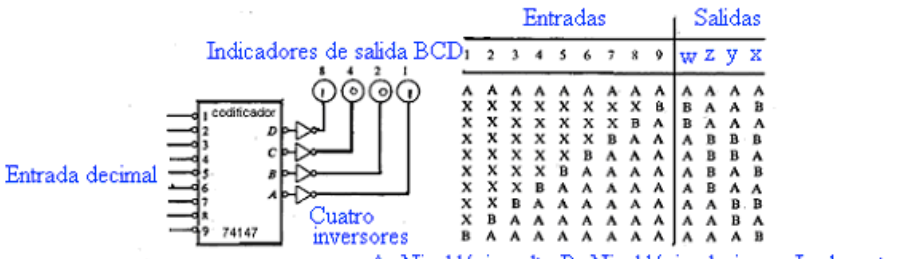


Fig.4.2 Símbolo lógico para un decimal a BCD



A=Nivel lógico alto,B=Nivel lógico bajo x = Irrelevante

a) Símbolo lógicocon indicadores de salida b) Tabla de verdad

Fig. 4.3 Codificador comercial para un codificador decimal a BCD 74147

El fabricante del codificador que se muestra en la figura 4.3 lo llama codificador de prioridad de diez líneas a cuatro líneas. A este dispositivo TTL se le denomina codificador 74147, cuya tabla de verdad se muestra en la figura 4.3b. La primera línea de la tabla de verdad es para cuando no hay entradas. Cuando todas las entradas flotan en ALTO, las salidas flotan en ALTO, y esto se interpreta como 0000 en los indicadores de salida BCD de la figura 4-3a. La segunda línea de la tabla de verdad en la figura 4-3b muestra la entrada decimal 9 siendo activada con un BAJO o 0, lo que produce BAAB en las salidas D, C, B y A; los cuatro inversores invierten BAAB y los indicadores BCD leen 1001, que es la forma de representar al 9 decimal en BCD.

La segunda línea de la tabla de verdad de la figura 4-3b muestra las entradas 1 a 8 marcadas con una X, que significa irrelevante. Una entrada irrelevante puede ser ALTO o bien, BAJO. Este codificador tiene un mecanismo de prioridad que activa el número mayor que tiene entrada BAJO. Si se colocará un BAJO en las entradas 9 y 5, la salida sería 1001, correspondiente al 9 decimal. El codificador sólo activa la salida del número mayor.

El diagrama lógico para el codificador 74147 según Texas Instruments, Inc., se muestra en la figura 4.4, en donde se ilustran las 30 compuertas. Primero trata de activar la entrada 9 decimal (BAJO en la entrada 9). Esta entrada O se invierte por medio del invertidor 1, y se aplica un 1 a las compuertas NOR 2 y 3 que se activan y producen BAJO. Las compuertas NOR 4 y 5 se desactivan por la presencia de ceros.

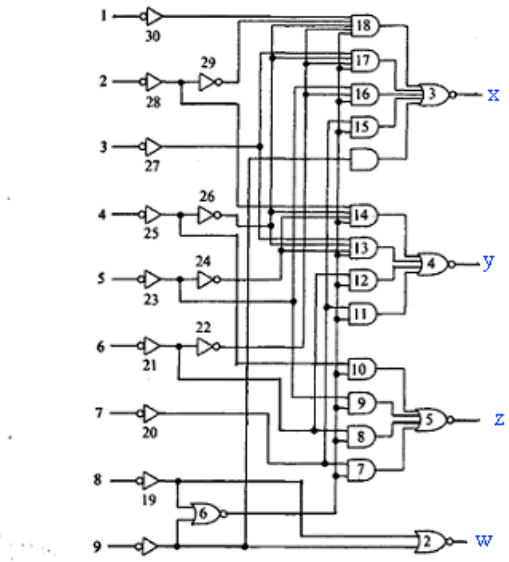


fig.4.4 Diagrama lógico de un codificador 74147 de prioridad decimal a BCD

En las entradas de las compuertas desactivadas AND de la 7 a la 18. Estas compuertas AND (7 a 18), se desactivan por los ceros en las entradas inferiores producidas por la compuerta NOR 6. Las compuertas AND (7 a 18) aseguran que la entrada decimal mayor tiene prioridad sobre los números menores.

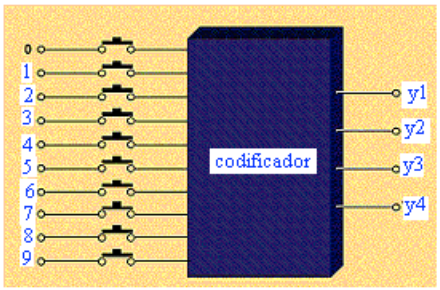


fig.4.5 Diagrama de bloques de un codificador de 10 entradas y 4 salidas

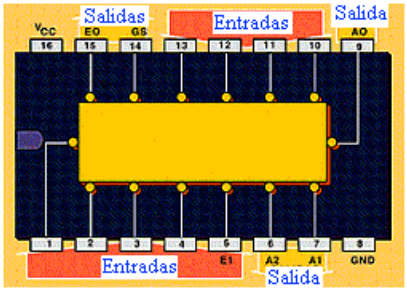


Fig. 4.6 Circuito integrado típico de un codificador con prioridad y señal de rehabilitación.

Entre las aplicaciones de este tipo de codificadores destacan la codificación de pequeños teclados, la conversión analógica a digital y el control de perturbaciones en los ordenadores.

DECODIFICADORES: BCD A DECIMAL

Podría decirse que un decodificador es lo opuesto a un codificador. Si se invierte el proceso descrito en la sección anterior se produciría un decodificador que traduce del código BCD a decimales. En la figura 4-7 se muestra un diagrama de bloque de dicho decodificador. El código BCD (8421) constituye la entrada a la izquierda del decodificador. Las líneas de las diez salidas se muestran a la derecha. A cualquier tiempo dado, solamente se activa una línea de salida, y para aclarar qué salida se activa se colocan unos indicadores (LED o lámparas) a estas líneas de salida.

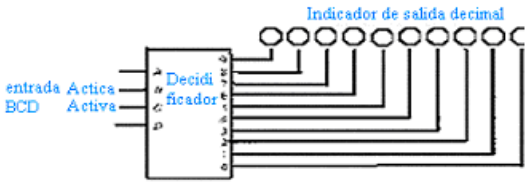
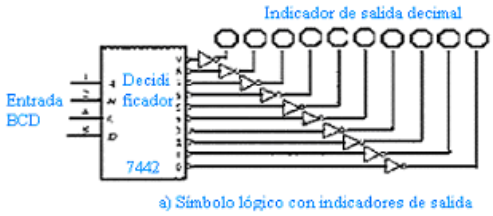


fig. 4.7 Símbolo lógico para un decodificador BCD a decima



Línea	No.	Entradas BCD				Salida decimal									
		D	C	B	A	0	1	2	3	4	5	6	7	8	9
1	0	B	B	B	B	A	A	A	A	A	A	A	A	A	A
2	1	B	B	B	A	B	A	A	A	A	A	A	A	A	A
3	2	B	B	A	B	A	B	A	A	A	A	A	A	A	A
4	3	B	B	A	A	A	A	B	A	A	A	A	A	A	A
5	4	B	A	B	B	A	A	A	B	A	A	A	A	A	A
6	5	B	A	B	A	A	A	A	A	B	A	A	A	A	A
7	6	B	A	A	B	A	A	A	A	A	B	A	A	A	A
8	7	B	A	A	A	A	A	A	A	A	A	B	A	A	A
9	8	A	B	B	B	A	A	A	A	A	A	A	B	A	A
10	9	A	B	B	A	A	A	A	A	A	A	A	A	B	A
11	No válido	A	B	A	B	A	A	A	A	A	A	A	A	A	A
12		A	B	A	A	A	A	A	A	A	A	A	A	A	A
13		A	A	B	B	A	A	A	A	A	A	A	A	A	A
14		A	A	B	A	A	A	A	A	A	A	A	A	A	A
15		A	A	A	B	A	A	A	A	A	A	A	A	A	A
16		A	A	A	A	A	A	A	A	A	A	A	A	A	A

A= ALTO B= BAJO

b) Tabla de verdad

fig. 4.8 Decodificador excitador comercial 7442 BCD a decimal.

En la figura 4.7 se activan las entradas B y C (B en el lugar de los 2 y C en el lugar de los 4). Esto activa la salida decimal 6, que se enseña con el indicador 6 iluminado. Si no hay entradas activadas, entonces el indicador de la salida O se ilumina. La entrada BCD 0011 activa el indicador de la salida 3.

En la figura 4.8a se muestra un decodificador comercial BCD a decimal. El fabricante de este dispositivo TTL le dio el número 7442. Las cuatro entradas BCD a la izquierda del símbolo lógico se marcan con las etiquetas D, C, B y A, en donde la entrada D corresponde a los 8, mientras que la entrada A corresponde a los 1. El 1 lógico, o ALTO, activa las entradas. A la derecha, salen diez líneas de salida del decodificador, y los círculos adjuntos al símbolo lógico indican que las salidas están activas en BAJO; normalmente flotan en ALTO, excepto cuando se activan. Por conveniencia se agregan diez inversores al circuito para manejar las luces de los indicadores decimales. Una salida activa se invierte al 1 lógico en los indicadores de salida.

La tabla de verdad para el decodificador 7442 se muestra en la figura 4.8b La primera línea, (que representa al O decimal) muestra todas las entradas en BAJO (B). Con la entrada BBBB (0000), la salida decimal O se activa a un estado BAJO. El inversor del extremo inferior complementa esta salida a ALTO que ilumina al indicador de la salida decimal 0. Los demás indicadores permanecen apagados. De la misma manera, la quinta línea (que representa al 4 decimal) muestra la entrada BCD como BABB (0100). La salida 4 se activa a BAJO, que se invierte iluminado el indicador decimal 4. Este decodificador tiene entonces entradas activadas en ALTO y salidas activadas en BAJO.

Considérese la línea 11 de la figura 4.8b, cuya entrada es ABAB 1010 y que normalmente representaría al decimal 10. Ya que el código no contiene este número, entonces ésta es una entrada no válida y ningún indicador de salida se ilumina (ninguna salida se activa). Hay que notar que las seis últimas líneas de la tabla de verdad muestran entradas no válidas, sin salidas activadas.

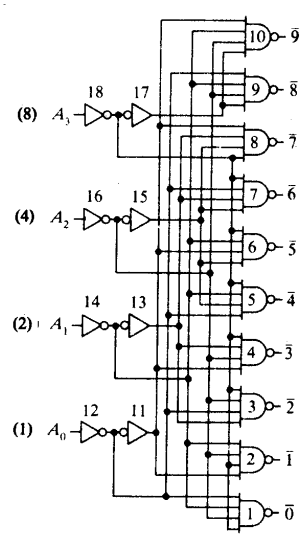


Fig. 4.9 Diagrama lógico del decodificador 7442 BCD a decimal

El diagrama de símbolos lógicos para el decodificador BCD 7442 a decimal se muestra en la figura 4-9. Las entradas BCD se encuentran a la izquierda y las salidas a la derecha. Las etiquetas de las entradas son algo diferentes de las que se usaron con anterioridad.

La entrada A_3 es el BIT más significativo o la entrada de los 8, mientras que la entrada A_0 corresponde al BIT menos significativo, o la entrada de los 1. Las salidas se etiquetan con números decimales, las salidas activadas BAJO del decodificador se muestran con una barra sobre las salidas decimales ($\bar{9}$, $\bar{8}$, y así sucesivamente). Suponga la entrada BCD BBBB (0000) en el decodificador de la figura 4.9 Si se sigue cuidadosamente el camino desde las cuatro entradas a través de los inversores 12, 14, 16 y 18, se observa que cuatro 1 lógicos se aplican a la compuerta NAND 1, que activa la compuerta produciendo un 0 lógico. Las demás compuertas NAND se desactivan con ceros en alguna de sus entradas. De la misma manera, cada combinación de entradas puede verificarse por medio de un análisis del diagrama lógico del decodificador 7442 de la figura 4.9 Las 18 compuertas de la figura 4-9 están contenidas en un circuito integrado (CI) al que se llama decodificador 7442. Como es usual, las conexiones (V_{CC} y GND al CI no se muestran en el diagrama lógico.

Decodificador de BCD a Siete Segmentos

El display de siete segmentos

El *display* está formado por un conjunto de 7 leds conectados en un punto común en su salida. Cuando la salida es común en los ánodos, el *display* es llamado de ánodo común y por el contrario, sí la salida es común en los cátodos, llamamos al *display* de cátodo común. En la figura 4.10, se muestran ambos tipos de dispositivos. En el *display* de cátodo común, una señal alta encenderá el segmento excitado por la señal. La alimentación de cierta combinación de *leds*, dará una imagen visual de un dígito de 0 a 9.

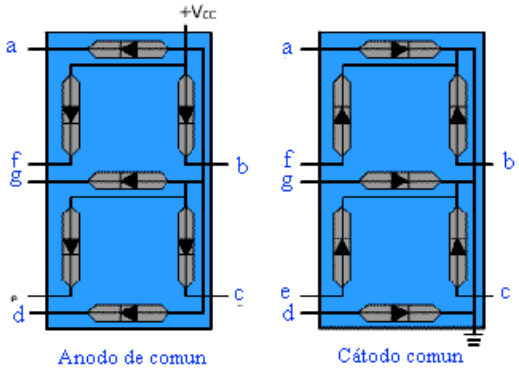


Fig.4.10 Display de ánodo común y Cátodo común

El decodificador requiere de una entrada en código decimal binario *BCD* y siete salidas conectadas a cada segmento del *display*. La figura 4.11 Representa en un diagrama de bloques el decodificador de *BCD* a 7 segmentos con un display de cátodo común.

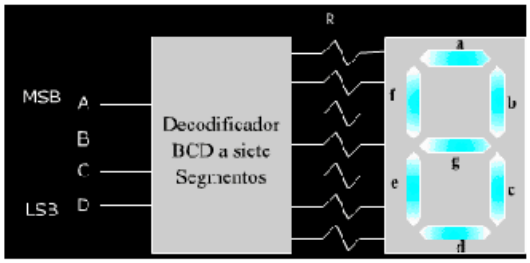


Figura 4.11 Diagrama de bloques de un decodificador BCD a siete segmentos

Suponiendo que el visualizador es un display de cátodo común, se obtiene una tabla cuyas entradas en código BCD corresponden a A, B, C y D y unas salidas correspondientes a los leds que se encenderían en cada caso para indicar el dígito decimal. La tabla 4.12 muestra el caso de ejemplo.

Valor decimal	Entradas	Salidas
	A B C D	a b c d e f g
0	0 0 0 0	1 1 1 1 1 1 0
1	0 0 0 1	0 1 1 0 0 0 0
2	0 0 1 0	1 1 0 1 1 0 1
3	0 0 1 1	1 1 1 1 0 0 1
4	0 1 0 0	0 1 1 0 0 1 1
5	0 1 0 1	1 0 1 1 0 1 1
6	0 1 1 0	1 0 1 1 1 1 1
7	0 1 1 1	1 1 1 0 0 0 0
8	1 0 0 0	1 1 1 1 1 1 1
9	1 0 0 1	1 1 1 0 0 1 1
10	1 0 1 0	x x x x x x x

Tabla 4.12

Los valores binarios 1010 a 1111 en BCD nunca se presentan, entonces las salidas se tratan como condiciones de no importa.

La simplificación de la información contenida en la tabla 4.12 requiere de siete tablas de verdad, que se pueden separar para cada segmento. Por consiguiente, un *1* en la columna indica la activación del segmento y varios de estos segmentos activados indican visualmente el número decimal requerido.

Según la información de la tabla de verdad, se puede obtener la expresión para cada segmento en suma de productos o producto de sumas según la cantidad de unos y ceros presentes.

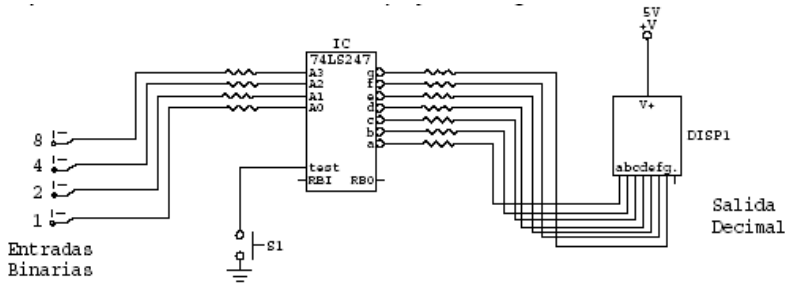


Fig. 4.13 Circuito para la salida e del decodificador BCD a siete segmentos

4.4ESTABLECER RELACIÓN EN LA CONSTRUCCIÓN , FUNCIONAMIENTO Y APLICACIÓN

4.5 MULTIPLEXOR Y DEMULTIPLEXORES

Un selector de datos es la versión electrónica de un conmutador rotatorio de un sentido. A la izquierda de la figura 4.14 se muestra un conmutador rotatorio de ocho posiciones

y un polo único. Las ocho entradas (0-7) se muestran a la izquierda, mientras que la única salida (Y) se etiqueta a la derecha. A la derecha se muestra un selector de datos. El dato en la entrada 2 (un 1 lógico) está siendo transferido a través de los contactos del conmutador rotatorio el dato en la entrada 2 (un 1 lógico) está siendo transferido a través de los circuitos del selector de datos a la derecha. La posición de los datos se selecciona girando mecánicamente el rotor del interruptor giratorio. La posición de los datos se selecciona en el selector de datos colocando los números binarios adecuados en las entradas selectoras de datos (C, B, A). El selector de datos permite pasar a los datos únicamente de entrada a salida, mientras que el interruptor rotatorio permite que los datos fluyan en ambas direcciones. Un selector de datos puede considerarse similar a un conmutador rotatorio de un sentido.

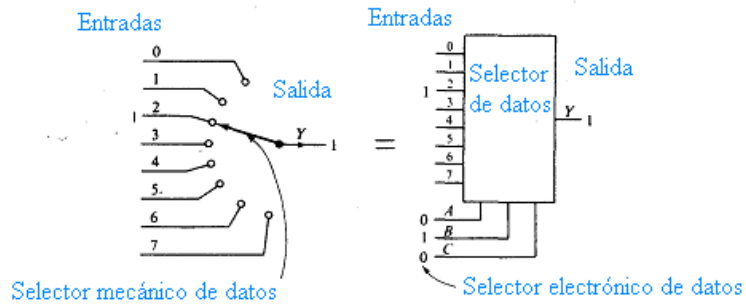
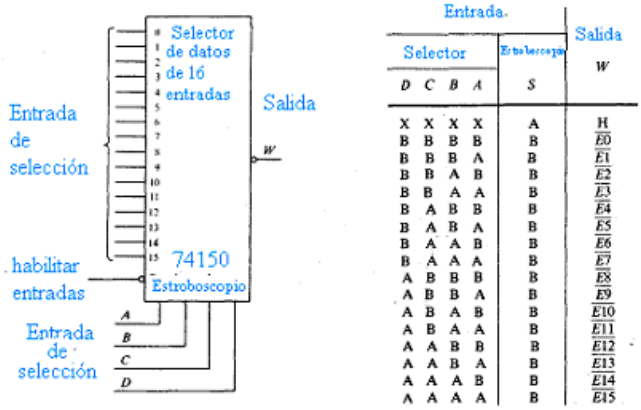


fig. 4.14 Comparación de un conmutador rotatorio con un selector de datos

En la figura 4.15a se muestra un selector de datos comercial en forma de diagrama de bloques. Este Cl TTL se identifica como un selector de datos/multiplexor 74150 de 16 entradas por los fabricantes. Note las 16 entradas en la parte superior izquierda. El 74150 tiene una sola salida invertida identificada como W. En la parte superior izquierda de la figura 4.15a, se identifican 4 entradas de selección de datos (D, C, B, A). Un BAJO en la entrada del estroboscopio habilitará al selector de datos y puede considerarse como un apagador principal.

Considere la tabla de verdad para el selector de datos 74150 de la figura 4.15b. La línea 1 muestra la entrada del estroboscopio (habilitar) en ALTO, lo cual desactiva a la unidad. La línea 2 nos muestra todas las entradas de selección de datos como BAJO, al igual que la entrada del estroboscopio. Esto permite que la información en la entrada de datos O sea transmitida a la salida W. La salida W se presentará en su forma invertida, como se simboliza con el E0 en la columna de la salida en la tabla de verdad. Al crecer la cuenta binaria (0001, 0010, 0011, etc.) hacia abajo en la tabla de verdad, cada entrada de datos se conecta consecutivamente a la salida W del selector de datos.



Entrada					Salida
Selector				Estroboscopio	
D	C	B	A	S	W
X	X	X	X	A	H
B	B	B	B	B	$\overline{E0}$
B	B	B	A	B	$\overline{E1}$
B	B	A	B	B	$\overline{E2}$
B	B	A	A	B	$\overline{E3}$
B	A	B	B	B	$\overline{E4}$
B	A	B	A	B	$\overline{E5}$
B	A	A	B	B	$\overline{E6}$
B	A	A	A	B	$\overline{E7}$
A	B	B	B	B	$\overline{E8}$
A	B	B	A	B	$\overline{E9}$
A	B	A	B	B	$\overline{E10}$
A	B	A	A	B	$\overline{E11}$
A	A	B	B	B	$\overline{E12}$
A	A	B	A	B	$\overline{E13}$
A	A	A	B	B	$\overline{E14}$
A	A	A	A	B	$\overline{E15}$

a) Símbolo lógico de bloque b) Tabla de verdad

fig. 4.15 EL CI TTL 74150 Multiplexor

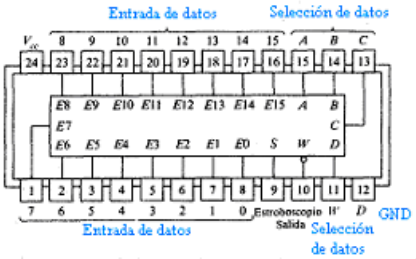


fig. 4.16 Diagrama de clavijas para el CI 74150 selector datos ó Multiplexor

El 74150 se presenta en un paquete de 24 clavijas. El diagrama de clavijas para este CI se muestra en la figura 4.16 Además de las 21 entradas y una salida que se muestran en el diagrama de bloque, el diagrama de clavijas también muestra la conexión a la fuente de poder (Vi y GND). Siendo un CI TTL, el 74150 requiere una fuente de poder de 5V.

Note el uso del término selector de datos multiplexor para identificar el CI 74150. Un multiplexor digital 74150 puede ser usado para transmitir una palabra paralela de 16 bit en serie. Esto se realiza conectando un contador a la entrada de selección de datos y contando de 0000 a 1111.

La palabra paralela de 16 bit en la entrada de datos (0-15) se transfiere a la salida en serie (una a la vez).

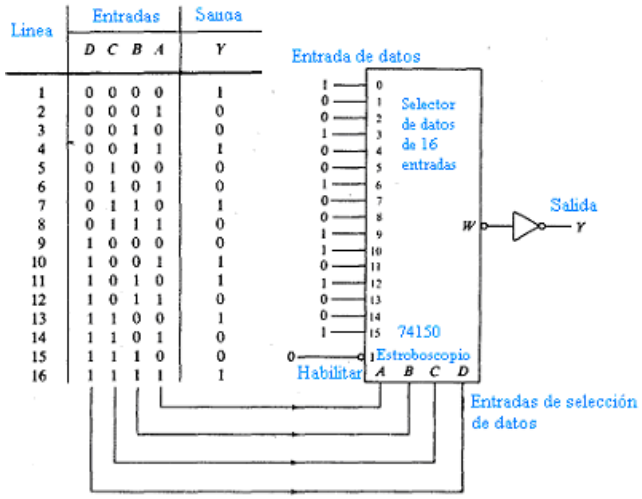


Fig. 4.17 Uso del selector de datos 74150 para resolver un problema de lógica combinacional.

También se puede usar, el selector de datos/multiplexor para resolver problemas difíciles de lógica combinacional. Considere la tabla de verdad a la izquierda de la figura 4.17 La expresión Booleana simplificada para esta tabla de verdad es ABCD + ABCD + ABCD + ABCD + ABCD + ABCD + ABCD = Y. Se necesitarían muchos circuitos integrados para implementar esta complicada expresión si se usaran los circuitos convencionales AND-OR o NAND. El selector de datos es un método fácil de resolver, éste, de otra forma, difícil problema.

En la figura 4.17 se presenta un problema de lógica combinacional. Para resolver el problema se usa un selector de datos de 16 entradas. Las 16 entradas de datos (0-15) en el CI 74150 tienen niveles lógicos aplicados correspondientes a la columna de salida de la tabla de verdad. La línea 1 de la tabla de verdad tiene una entrada binaria de 0000 (0 decimal) con una salida de 1. Se aplica entonces el 1 a la entrada 0 del selector de datos. La línea 2 en la tabla de verdad tiene una entrada binaria de 0001 (1 decimal) con una salida de 0. Se aplica entonces el 0 a la entrada 1 del selector de datos. Los niveles lógicos de entrada (D, C, B, A) de la tabla de verdad se aplican a las entradas de selección del selector de datos 74150. La entrada habilitar del CI 74150 se coloca en 0, y la unidad resuelve el problema lógico de la tabla de verdad. Nótese en la figura 11-9 que debido a la salida inversa del selector de datos 74150 se añade un inversor a la derecha. La solución del selector de datos a este problema de lógica combinacional fue una solución fácil y rápida en un so lo paquete.

DEMULTIPLEXORES

Un multiplexor toma varias entradas y transmite una de ellas a la salida. Un demulti plexor efectúa la operación contraria; toma una sola entrada y la distribuye en varias salidas. La figura 4.18 muestra el diagrama general de un demultiplexor (DEMUX).

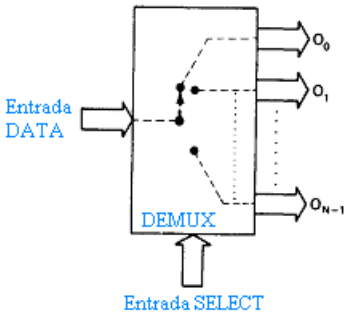


Fig. 4.18 Demultiplexor general

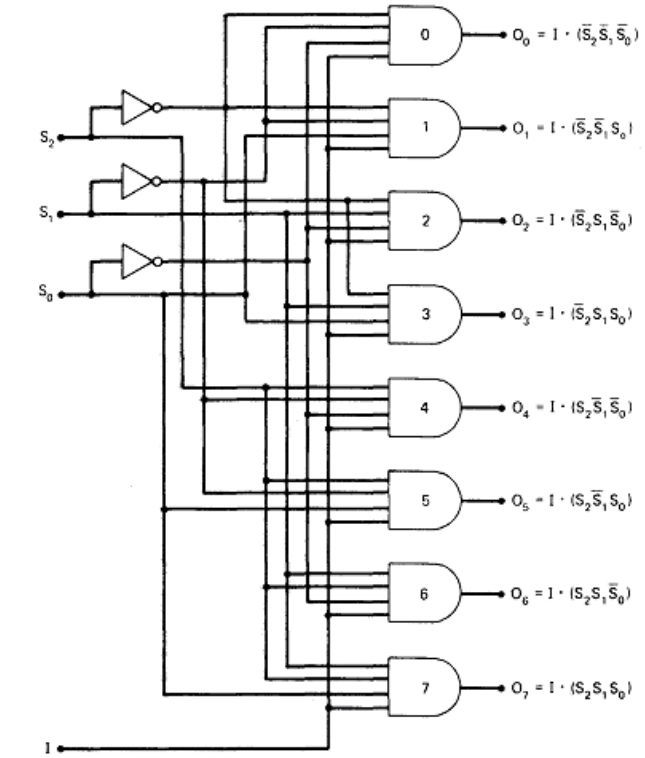
La entrada DATA se transmite solamente a una de las salidas como lo determina el código de entrada de selección

Las flechas grandes que corresponden a entradas y salidas pueden representar una o más líneas. El código de entrada de selección determina hacia qué salida se transmitirá la entrada DATA. En otras palabras, el demultiplexor o distribuidor de datos toma una fuente de datos de entrada y la distribuye selectivamente a 1 de N canales de salida, igual que un interruptor de múltiples posiciones.

Demultiplexor de 1 a 8 líneas, la figura 4.19 muestra el diagrama lógico de un distribuidor de datos que distribuye una línea de entrada a ocho líneas de entrada. La línea de entrada de datos individual 1 se conecta a las ocho compuertas AND, pero sólo una de estas compuertas será activada por las líneas de entrada SELECT. Por ejemplo, con S = 000, solamente la compuerta AND O será activada, y la entrada de datos 1 figurará en la salida 00. Otros códigos SELECT ocasionan que la entrada 1 llegue a las otras salidas. La tabla de verdad resume la operación.

Ya antes se observó la forma en que se utiliza el 74LS 138 como decodificador 1 de 8. La figura 4.20 muestra cómo puede emplearse para que funcione como demultiplexor o distribuidor de datos. La entrada activada E' se usa como la entrada de datos 1, en tanto que las otras dos entradas activadas se mantienen en sus estados activos. Las entradas A_2 A_1 A_0 sirven como código de selección. Para ilustrar la operación, supongamos que las entradas de selección son 000. Con este código de entrada, la única salida que puede activarse es O'_0, en tanto que todas las otras salidas son ALTAS. O'_0, pasará a BAJA sólo si E'_1 cambia a BAJA y será ALTA si E'_1 cambia a ALTA. Dicho de otra manera, O'_0 seguirá la señal en E (es decir, la entrada de datos, 1) mientras todas las otras salidas permanecen ALTAS. En forma análoga, un código de selección diferente aplicado a A_2 A_1 A_0 ocasionará que la salida correspondiente siga la entrada de datos, 1.

Demultiplexor con cronómetro muchas aplicaciones del principio de la distribución de datos son posibles. La figura 4.21 muestra el demultiplexor 74LS 138 que se usa como demultiplexor con cronómetro. Con el control de las líneas SELECT, la señal del cronómetro es guiada al destino deseado. Por ejemplo, con S_2 S_1 S_0 = 000, la señal del reloj aplicada a 1 figurará en la salida O'_O Y Con S_2 S_1 S_0 = 101, la señal del cronómetro ocurrirá en . O'_5



Código select			Salidas							
S_2	S_1	S_0	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0	0	0	0	0	0	0	I
0	0	1	0	0	0	0	0	0	I	0
0	1	0	0	0	0	0	0	I	0	0
0	1	1	0	0	0	0	I	0	0	0
1	0	0	0	0	0	I	0	0	0	0
1	0	1	0	0	I	0	0	0	0	0
1	1	0	0	I	0	0	0	0	0	0
1	1	1	I	0	0	0	0	0	0	0

Fig. 4.19 Demultiplexor de 1 a 8 líneas

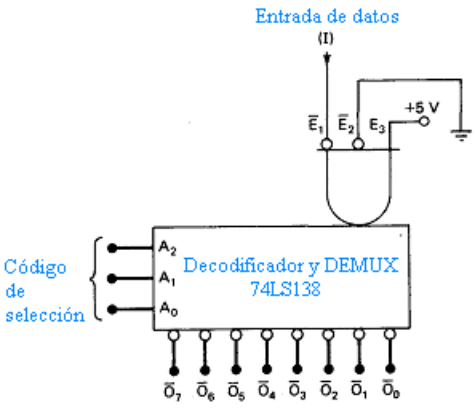


Fig. 4.20 Decodificador 74LS138 que se usa como demultiplexor

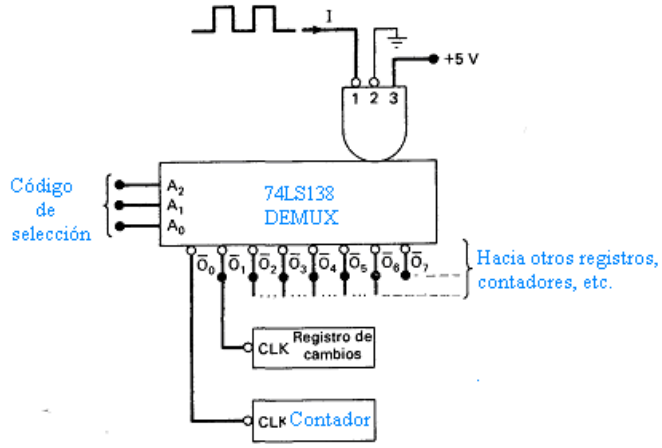


Fig. 4.21 Demultiplexor con cronómetros

4.6 RECONOCE LOS CIRCUITOS ARITMÉTICOS

El diseño de sistemas digitales involucra el manejo de operaciones aritméticas. En esta lección se implementarán los circuitos de suma y resta de números binarios.

4.7 Sumador Medio. El circuito combinacional que realiza la suma de dos bits se denomina sumador medio. La figura 4.22 muestra el símbolo lógico de sumador medio. En el circuito las entradas son A y B y la salida S corresponde a la suma y C_{out} al acarreo de salida.

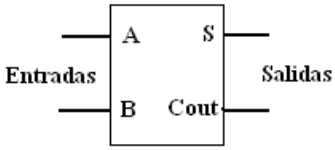


Figura 4.22 Símbolo lógico del sumador medio.

La tabla de verdad 4.37 está dada por las reglas de la suma binaria.

x	y	C_{OUT}	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Tabla 4.23 Tabla de verdad del sumador medio

La salida obtenida a partir de la tabla de verdad es:

$$X + Y = C_{out}\, S$$

El BIT de acarreo C_{out} es 1, sólo cuando A y B tienen el valor de 1; por tanto entre A y B se puede establecer una operación AND:

$$C_{out} = A \cdot B$$

El BIT de suma *S* es 1, sólo si las variables A y B son distintas. El BIT de acarreo es 0 a no ser que ambas entradas sean 1. Por consiguiente, la salida S puede expresarse en términos de la operación OR – Exclusiva:

$$S = A' \cdot B + A \cdot B = A \oplus B$$

El circuito se muestra en la figura 3.9.2.

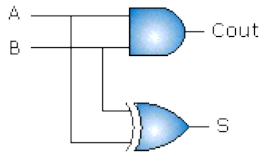


Figura 4.24 Circuito lógico del sumador medio.

Sumador Completo

El sumador completo acepta dos bits y un acarreo de entrada y genera una suma de salida junto con el acarreo de salida. La tabla 4.39 muestra la tabla de verdad del sumador completo. Las entradas A, B y C_{in} denotan al primer sumando, el segundo sumando y el acarreo de entrada. Las salidas S y C_{out} representan a la suma y el acarreo de salida.

A	B	C_{in}	C_{out}	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Tabla 4.25 Tabla de verdad del sumador completo

La salida *S* en la tabla de verdad corresponde a la operación OR- Exclusiva:

$$S = A \cdot B' \cdot C_{in}' + A' \cdot B \cdot C_{in}' + A \cdot B \cdot C_{in} + A' \cdot B' \cdot C_{in}$$

$$S = C_{in}' \cdot (A \cdot B' + A' \cdot B) + C_{in} \cdot (A \cdot B + A' \cdot B')$$

$$S = C_{in}' \cdot (A \cdot B' + A' \cdot B) + C_{in} \cdot (A' \cdot A + A' \cdot B' + A \cdot B + B \cdot B')$$

$$S = C_{in}' \cdot (A \cdot B' + A' \cdot B) + C_{in} \cdot ((A' + B) \cdot (A + B'))$$

$$S = C_{in}' \cdot (A \cdot B' + A' \cdot B) + C_{in} \cdot ((A \cdot B')' \cdot (A' \cdot B)')$$

$$S = C_{in}' \cdot (A \cdot B' + A' \cdot B) + C_{in} \cdot (A \cdot B' + A' \cdot B)'$$

$$S = (A \oplus B) \oplus C_{in}$$

El mapa de karnaugh de la salida C_{out} se muestra en la figura 4.40.

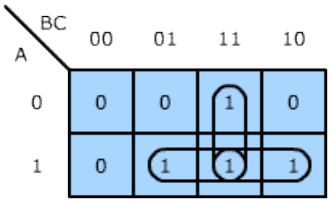


Figura 4.26. Mapa para la salida Cout de un sumador completo.

La salida C_{out} está dada por:

$$C_{out} = A \cdot B + A \cdot C_{in} + B \cdot C_{in}$$

El circuito se muestra en la figura 4.27

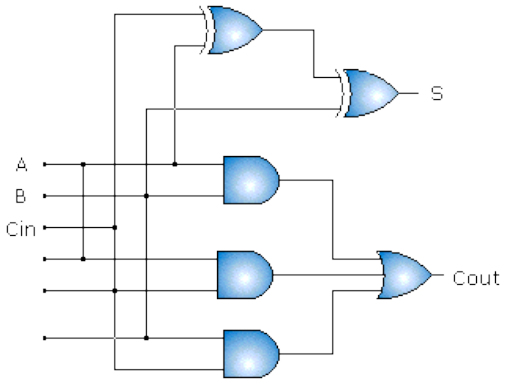


Figura 4.27 Circuito lógico del sumador completo.

4.8 Restador

En la diferencia, cada BIT del sustraendo se resta de su correspondiente BIT del minuendo para formar el BIT de la diferencia. El préstamo ocurre cuando el BIT del minuendo es menor al BIT del sustraendo, de tal forma que se presta un 1 de la siguiente posición significativa.

La resta se implementa mediante un sumador. El método consiste en llevar al minuendo a una de las entradas y el sustraendo en complemento 2 a la otra entrada.

Restador Medio
El circuito combinacional que realiza la resta de dos bits se denomina Restador medio. El circuito tiene dos entradas binarias y dos salidas. La figura 4.42 muestra el símbolo lógico de restador medio. En el circuito las entradas son *A* (minuendo) y *B* (sustraendo) y la salida *D* corresponde a la diferencia y *P* al préstamo de salida.

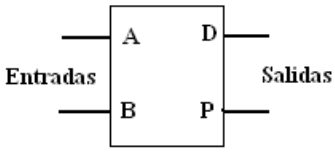


Figura 4.28 Símbolo lógico del restador medio.

Si A- B, existen tres posibilidades 0-0=0, 1-0=1 y 1-1=0. El resultado es el BIT de diferencia D. Si A<B se tiene 0-1 y es necesario prestar un 1 de la siguiente posición significativa de la izquierda. El préstamo agrega 1 al BIT del minuendo de manera similar cuando en el sistema decimal se agrega 10 al dígito del minuendo.

La tabla de verdad 4.29 está dada por las reglas de la resta binaria.

A	B	P	D
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

Tabla 4.29 Tabla de verdad del restador medio.

La salida D coincide con la operación OR- Exclusiva y se puede expresar de la siguiente forma:

$$D = A' \cdot B + A \cdot B'$$

La salida *P* está dada por la suma de productos de los términos presentes en el renglón 2 de la tabla de verdad:

$$P = A' \cdot B$$

El circuito se muestra en la figura 4.30

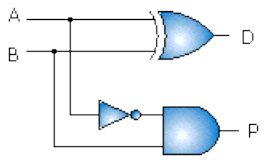


Figura 4.30 Circuito lógico del restador medio.

Restador completo

El Restador completo realiza la resta entre dos bits, considerando que se ha prestado un *1* de un estado menos significativo. En la tabla 3.9.4. las entradas *A, B* y *C* denotan el minuendo, el sustraendo y el BIT prestado. Las salidas *D* y *P* representan a la diferencia y el préstamo.

CECyT 1 9/3/12 13:45
Comentario [1]: Que e donde salió

A	B	C	P	D
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

Tabla 4.31 Tabla de verdad del restador completo.

En las combinaciones del mapa donde C=0, se tienen las mismas condiciones para el sumador medio.

La función de la salida D de un restador es la misma que la salida de un sumador completo:

$$D = A'\cdot B'\cdot C + A'\cdot B\cdot C' + A\cdot B'\cdot C' + A\cdot B\cdot C = (A \oplus B) \oplus C_{in}$$

El mapa de karnaugh de la salida P se muestra en la figura 4.42

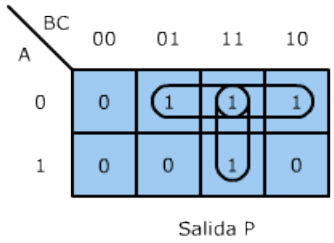


Figura 4.32 Mapa para la salida P de un restador completo

La salida P está dada por:

$$P = A'\cdot B + A'\cdot C + B\cdot C$$

El circuito se muestra en la figura 4.33

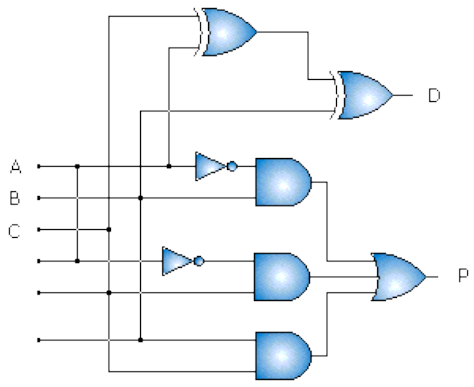


Figura 4.33 Tabla de verdad del restador completo.

CONTENIDO

BIBLIOGRAFÍA

SISTEMAS DIGITALES PRINCIPIOS Y APLICACIONES	Ronald J. Tocci	Prentice may
PRINCIPIOS DIGITALES	Roger L. Tokheim	Mc Graw Hill
DISEÑO DIGITAL	M. Morris Mano	Prentice may
FUNDAMENTOS DE SISTEMAS DIGITALES	Thomas L. Floyd	Prentice may
FUNDAMENTOS DE LÓGICA DIGITAL CON DISENO	Brown Stephen	Mc Graw Hill
LÓGICA DIGITAL Y DISEÑO DE COMPUTADORES	M. Morris Mano	Prentice Hall
TEORÍA DE COMPUTACIÓN Y DISEÑO LÓGICO	Frederick J. Hill	Limusa
ELECTRONICA DIGITAL MODERNA	J.M.Angulo	Paraninfo
CIRCUITOS DE COMPUTADORA	Saul Ritterman	Mc. Graw Hill
FUNDAMENTOS DE LOS MICROPROCESADORES	Roger L.tokheim	Mc. Graw Hill

PÁGINAS WEB

http://apuntesgratis.oposicionesyempleo.com/electronicadigital.pdf
http://www.academiasigloxxi.com/biblioteca/Tecnicas%20digitales%20-%20Apuntes1.pdf
http://www.econ.uba.ar/www/departamentos/humanidades/plan97/logica/Legris/apuntes/AP-Circuitos.pdf
http://fcqi.tij.uabc.mx/docentes/jjesuslg/LAB-D1-00-2.PDF
http://lc.fie.umich.mx/~jrincon/elec3-cap7.pdf
http://fisica.udea.edu.co/~gicm/lab_electronica/compuertas_logicas.pdf
http://www.control-systems-principles.co.uk/whitepapers/spanishwp/17DigitalSystemsSP.pdf
http://www.el.bqto.unexpo.edu.ve/~ltarazona/digitales/tema4A_4.pdf
http://ocw.ucv.ve/facultad-de-ingenieria/logica-digital-6/materiales/ALGEBRADEBOOLEYFUNCIONESLOGICA.pdf
http://www.aves.edu.co/ovaunicor/recursos/1/index_4_Guia_compuertas_y_Karnaugh.pdf

GLOSARIO DE LOS TÉRMINOS UTILIZADOS CON MÁS FRECUENCIA

ASCII

Éstas siglas que significan American Standard of Computer Information In terchange, es una norma de codificación de los signos universalmente adoptados para los microordenadores. Cada signo, carácter alfanumérico, de puntuación u otro, posee un código numérico denominado «código ASCII». A título de ejemplo, la letra “A” posee el código 65 y “z” el código 122. El juego de caracteres estándar que utilizan la mayor parte de las máquinas MS-DOS comporta 255 signos (se trata de un «ASCII extendido», con relación al de los inicios que estaba constituido por 64).

BINARIO

En un ordenador, todos los datos, números, caracteres, etc. están representados por dos «estados» eléctricos elementales: el estado «alto», que corresponde al paso de corriente o «interruptor cerrado», al que se da el valor 1. Y el estado «bajo», que es la ausencia de corriente, «interruptor abierto», al que se da el valor 0. Este sistema de contar con dos signos se denomina sistema binario, de la misma forma que nuestro sistema habitual con diez signos se denomina sistema decimal. Todas las operaciones internas de los ordenadores se efectúan en el sistema binario.

BIT

Es la más pequeña unidad de información que existe, que sólo puede tomar los valores O ó 1. Para facilitar la manipulación de los bits, normalmente éstos se agrupan en palabras de ocho bits, u octetos. Se habla de ordenadores de 8 o 16 bits, pues los primeros sólo manipulan un octeto a la vez, mientras que los segundos son capaces de gestionar dos simultáneamente.

CIRCUITO INTEGRADO

Se trata de una pequeña pastilla de silicio, que incluye de algunas decenas a algunas decenas de miles de transistores. La casi totalidad de los elementos activos del ordenador: unidad central, memorias, sintetizador de voz, etc., están constituidos por circuitos integrados.

CMOS

Siglas de Complementary Metal Oxide Semiconductor. Se trata de una tecnología particular de fabricación de los circuitos integrados. Presenta la ventaja de consumir poca corriente, lo que hace posible la conservación de datos almacenados en memorias CMOS durante muy largos períodos, con ayuda de acumuladores. La memoria viva de numerosos ordenadores está constituida por circuitos CMOS.

SISTEMA OCTAL

Esta numeración opera con ocho números (símbolos) 0,1,2,3,4,5,6,7 se utilizan para representar cualquier número, su base es ocho.
El número mayor en base octal es el 7.
Para contar arriba de 7 se coloca un cero nuevamente en la posición de las unidades y se continúa contando 10, 11, 12, 13, 14, 15, 16, 17.
Después de 17 se coloca un cero nuevamente en la posición de las unidades y los siguientes números son: 20, 21, 22, 23, 24, 25, 26, 27 y así sucesivamente.

HEXADECIMAL

El sistema hexadecimal es un sistema numérico, como el decimal o el binario. Comprende 16 signos, es decir, los diez del sistema decimal más las letras& B. C. D, E y F. Los números se expresan a menudo en hexadecimal cuando un programa en lenguaje máquina, ya que se facilita enormemente la conversión 1 binario (lo que no sucede en el caso del sistema decimal).

OCTETO (BYTE)

Para facilitar los diálogos con la máquina, así como las manipulaciones de datos, se utilizan «palabras» de 8 bits llamadas octetos. Un octeto puede tomar 256 (2 valores diferentes. Dado que la totalidad de los caracteres del juego ASCII extendido están codificados con un solo octeto, a menudo se asocia un octeto a un carácter. Esto sólo está justificado parcialmente, ya que si bien un carácter ocupa un octeto cuando se almacena mediante su código ASCII, claramente puede ocupar menos espacio cuando se utiliza una forma de almacenamiento binario (compactada). MS-DOS es capaz de utilizar los dos métodos.

EL CÓDIGO GRAY

El código Gray no es pesado (los dígitos que componen el código no tienen un peso asignado).
Su característica es que entre una combinación de dígitos y la siguiente, sea ésta anterior o posterior, sólo hay una diferencia de un dígito.
Por eso también se le llama código progresivo.

Esta progresión sucede también entre la última y la primera combinación. Por eso se le llama también código cíclico.

BUFFER

Un símbolo triángulo por sí mismo designa un circuito separador, el cual no produce ninguna función lógica particular puesto que el valor binario de la salida es el mismo de la entrada. Este circuito se utiliza simplemente para amplificación de la señal. Por ejemplo, un separador que utiliza 5 volt para el binario 1, producirá una salida de 5 volt cuando la entrada es 5 volt. Sin embargo, la corriente producida a la salida es muy superior a la corriente suministrada a la entrada de la misma.

MULTIPLEXORES

Un Multiplexor o "Selector de datos**"** es un circuito lógico que acepta varias entradas de datos y permite que sólo una de ellas pase a un tiempo a la salida. El enrutamiento de la entrada de datos hacia la salida está controlado por las entradas de selección (a las que se hace referencia a veces como las entradas de dirección).

DEMULTIPLEXORES

Es lo inverso a un multiplexor. Los demultiplexores o DEMUX tienen una entrada que es transferida a una de las m posibles líneas de salida. La línea m vendrá direccionada por los n bits de selección donde lo normal es que 2^nm.cada salida del demultiplexor corresponde con el término mínimo del número binario que se encuentra en las líneas de selección

ELABORADOS POR
ING. ANGEL CRUZ ANTONIO