INTRODUCCION AL PROCESAMIENTO DIGITAL

(del 5 al 9 de septiembre de 1977)

Fecha	Duración	Tema	Profesor
5 de sept.	9 a 13 h	1.- INTRODUCCION 2.- CONCEPTOS BASICOS	M. en C. Alejandro Guarda Auras
	13 a 14 h	comida	
	14 a 18 h	3.- CIRCUITOS COMBINACIONALES	M. en C. Alejandro Guarda Auras
6 de sept.	9 a 11 h	3.- CIRCUITOS COMBINACIONALES	M. en C. Alejandro Guarda Auras
	11 a 13 h	4.- BIESTABLES Y REGISTROS DE CORRIMIENTO	Ing. Arturo González Hermosillo
	13 a 14 h	comida	
	14 a 18 h	4.- BIESTABLES Y REGISTROS DE CORRIMIENTO	Ing. Arturo González Hermosillo
7 de sept.	9 a 13 h	5.- CIRCUITOS SINCRONOS	Ing. Mario Rodríguez M.
	13 a 14 h	comida	
	14 a 18 h	6.- CIRCUITOS ASINCRONOS	M. en C. Angel Kuri M.
8 de sept.	9 a 13 h	7.- UNIDADES ARITMETICAS	M. en C. Angel Kuri M.
	13 a 14 h	comida	
	14 a 18 h	8.- FAMILIAS LOGICAS E INTERCONEXION	Dr. José Fco. Albarrán Núñez

'pmc.

INTRODUCCION AL PROCESAMIENTO DIGITAL

(del 5 al 9 de septiembre de 1977)

Fecha	Duración	Tema	Profesor
9 de sept.	9 a 12 h	8.- FAMILIAS LOGICAS E INTERCONEXION	Dr. José Fco. Albarrán Núñe
	12 a 13 h	9.- REFLEXIONES Y CARRERAS	M. en C. José Ruiz Ascencio
	13 a 14 h	comida	
	14 a 17 h	9.- REFLEXIONES Y CARRERAS	M. en C. José Ruiz Ascencio
	17 a 18 h	10.- PERSPECTIVAS. EL CASO DE LOS MICROPROCESADORES	TODOS LOS PROFESORES
		CLAUSURA	

'pmc.

DIRECTORIO DE PROFESORE S DEL CURSO INTRODUCCION AL PROCESA_

MIENTO DIGITAL

DR. JOSE F. ALBARRAN NUÑEZ
TITULAR "A"
DESFI, UNAM
TEL. 550.18.24

ING. ARTURO GONZALEZ HERMOSILLO MELGAREJO
INVESTIGADOR ASOCIADO
I I M A S
CIUDAD UNIVERSITARIA
TEL.: 550.52.15 E. 4582

M. EN C. ALEJANDRO GUARDA AURAS
INVESTIGADOR
IIMAS
CIUDAD UNIVERSITARIA
TEL. 550.52.15 E. 4573

M. EN C. ANGEL KURI MORALES
INVESTIGADOR
INST. DE INV. MATEMATICAS APLICADAS
Y EN SISTEMAS
CIUDAD UNIVERSITARIA
TEL. 548.33.60

ING. MARIO RODRIGUEZ MANZANERA
INVESTIGADOR ASOCIADOS
IIMAS
CIUDAD UNIVERSITARIA
TEL. 550.52.15 E.4582

M. EN C. JOSERUIZ ASCENCIO
INVESTIGADOR ASOCIADO
IIMAS
CIUDAD UNIVERSITARIA

centro de educación continua
división de estudios superiores
facultad de ingeniería, unam

INTRODUCCION AL PROCESAMIENTO DIGITAL

M. en C. Alejandro Guarda Auras

Septiembre, 1977

INDICE

CAPITULO 1: INTRODUCCION.

1.1.- Técnicas Digitales versus Técnicas Analógicas:

La Electrónica puede concebirse como aquella área de la técnica que trata con el procesamiento de señales. Hasta aproximadamente diez años atrás, dicho procesamiento se hacía fundamentalmente mediante técnicas analógicas en las que "codificaba" una señal en función de su amplitud, frecuencia o fase.

Las técnicas digitales, hasta la época mencionada, se consideraban demasiado costosas, aún cuando se reconocían sus ventajas en que el tipo de codificación que empleaba, mediante niveles de voltaje, era mucho más versatil que el analógico.

Desde hace una década, con la producción de los primeros circuitos integrados digitales (compuertas RTL) se observó que los problemas de costo, tamaño, etc., podían ser superados y desde entonces, las técnicas digitales han evolucionado al punto que no hay lugar del dominio analógico donde no haya aparecido un reemplazo digital. Vemos así que en relojes [1], T.V. [2], teléfonos [3], controles industriales [4], artículos domésticos [5], etc., cada día aparecen nuevos dispositivos, sin mencionar el campo de las computadoras que le es propio.

Indudablemente, que en un universo analógico, las técnicas analógicas no van a desaparecer. Sin embargo, se observa claramente que su función se está concentrando en dispositivos para la adecuación de señales y la adquisición de datos. Es decir, hacia transformar señales analógicas en digitales para posteriormente ser procesadas digitalmente.

¿A qué se debe esta tendencia actual? A nuestro juicio, hay tres factores prinicpales.

Exactitud
Inmunidad al ruido
Estandarización.

A continuación examinaremos estas cualidades de los sistemas digitales.

1.2.- Exactitud e Inmunidad al ruido

Habíamos dicho que en un sistema analógico la codificación de una señal se establece a través de variaciones continuas de la amplitud, frecuencia o fase de la señal. Por lo tanto, la exactitud en un sistema analógico estará limitada por: la mínima señal detectable -nivel de ruido-, la máxima señal transferible -saturación del sistema- y la distorsión propia del sistema. En condiciones muy ideales, un amplificador de bajo ruido y con un rango dinámico de 120 db, puede procesar una señal con una exactitud de una parte por millón, lo que corresponde aproximadamente a 2^{20}.

Un sistema digital, en cambio, trabaja con señales binarias (1 ó 0), las que son empleadas para codificar la información en forma numérica. Esto hace que la exactitud con que se represente la información estará limitada exclusiva-

mente por el número de dígitos binarios, o bits, empleados; es decir por la longitud de palabra*. Como ejemplo podemos considerar la computadora CDC 6400/6600 que tiene una palabra de 64 bits y que por lo tanto, permite representar números menores o iguales a $(2^{64}-1)$ es decir, del orden de 1.8×10^{19}, lo que significa contar con una exactitud de una parte en 10^{19}; es decir, 10^{13} veces más exacto que el amplificador citado.

De lo anterior, podemos decir que la exactitud en un sistema digital está limitada solamente por cuestiones económicas (que también son muy válidas), ya que eventualmente podríamos trabajar con longitudes de palabras aún mayores. De hecho, si en la CDC 6400/6600 programáramos empleando doble precisión, contaríamos con una exactitud de una parte en 10^{38}, lo cual, aún para trabajos astronómicos resulta suficiente. En lo que respecta al ruido, es sabido como éste afecta a una señal analógica y que al ser amplificada, también estamos amplificando ruido.

En sistemas digitales, la especificación de los niveles de 0 y 1 permite ciertas fluctuaciones en torno al nivel de operación. Esto equivale a trabajar con "bandas" de señal, como se muestra en la figura 1-1.

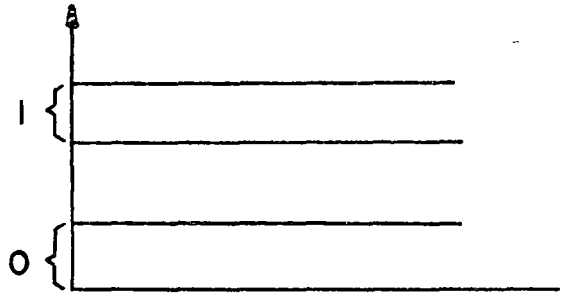


FIGURA 1-1

Esto hace que los sistemas digitales tengan cierta "tolerancia" al ruido. Es decir, si la señal llega distorsionada, pero dentro de los niveles establecidos, será reconocida como un 1 o un 0 según el caso.

Esto resulta,evidentemente, la ventaja más clara de un sistema digital con respecto a uno analógico, desde el punto de vista de operación.

1.3.- Estandarización:

Una característica de las técnicas digitales es que es posible (aunque no recomendable) diseñar casi cualquier sistema con el mismo tipo de compuerta y el mismo tipo de flip-flop. De hecho, en una época, esa fue la tendencia. Esto permitió la estandarización del tipo de componente empleado y como consecuencia, la producción masiva del mismo, con el consiguiente abatimiento de costos.

---

* La longitud de palabra, es el nivel primario en que se agrupan los dígitos binarios (bits) en una computadora.

Como resultado de ésto, se tiene que por ejemplo, hay un sólo tipo de compuerta NAND, y la única variación está en el número de entradas (2,3,4 y 8) y en el tipo de salida (totem pole y colector abierto).

Si como comparación consideramos el amplificador operacional, veremps que existe una gama enorme de variaciones sobre el tema: bajo ruido, entrada con FET's, instrumentación.

Esto es de esperarse puesto que las señales de entrada a los sistemas digitales consisten de pulsos estándar, y en cambio el tipo de señales que se procesan analógicamente, cubre un espectro sumamente amplio. Por esta razón, en el primer caso se puede tener un mismo tipo de componente operando en sistemas cuyas funciones sean completamente diferentes, lo cual no es el caso de los sitemas analógicos.

Un ejemplo muy claro de los efectos de la estandarización y que trasciende al ámbito de la electrónica es el caso de los relojes. El reloj analógico (mecánico) está compuesto por una gran cantidad de piezas y cada una, aunque en principio sea la misma que se empleé en otro reloj, es susceptible a variaciones: por ejemplo el número de dientes en un engrane. El reloj electrónico en cambio está hecho de un circuito integrado estándar, del que se fabrican miles con el mismo proceso. Esto se refleja en los costos actuales de un reloj digital en los que el costo del circuito integrado es una fracción mínima (menos del 10%).

Otro ejemplo de estandarización es el caso de los microprocesadores. Hasta su introducción al mercado, la mayoría de los componentes más complejos, de uso específico y de fabricación masiva, eran ordenados por un cliente a una fábrica de semiconductores para su fabricación. Es lo que se llama "custom design", en la que la fábrica de circuitos desempeña la función de "sastre", diseñando un circuito adecuado a las necesidades precisas del cliente.

Con los microprocesadores se ha puesto en el mercado un dispositivo reconfigurable en su función de acuerdo a las necesidades del usuario y dicha reconfiguración es realizada por el mismo usuario. Es decir, con un circuito estándar, producido masivamente, se pueden realizar funciones completamente diferentes como las de una calculadora hasta el control de una central privada de teléfonos.

A todo lo anterior se suma la enorme ventaja de disminuir los costos de mantenimiento y servicio al requerirse un stock más reducido de componentes.

1.4.- Algunas consideraciones sobre diseño de sistemas digitales.

Se ha estimado en estudios realizados que el costo de los circuitos integrados y componentes de un sistema, representa menos del 10% del costo total del sistema. Esto es muy significativo y nos hace revisar las metas de diseño que se tenían hasta hace 6 ó 7 años atrás.

Las primeras computadoras y aún las de principios de los 60, eran construídas empleando componentes discretos -diodos, transistores, resistores, capacitores, etc.- y por lo tanto, uno de los aspectos más estudiados en los centros de investigación y más enseñados en las Universidades, eran precisamente técnicas tendientes a minimizar el número de componentes, que equivale a minimizar funciones booleanas. El objetivo perseguido con estos métodos de minimización, era el de reducir el número de compuertas y flip-flop y de esta manera reducir el número de componentes. En la actualidad, los circuitos comerciales traen varias compuertas en cada circuito integrado -específicamente, 4 compuertas NAND de dos entradas, 3 compuertas NAND de 3 entradas, 2 compuertas NAND de 4 entradas, etc.- y en este contexto,el minimizar el número de compuertas pierde bastante sentido, por que si originalmente podíamos implementar una función con 8 compuertas y después de invertir cierto tiempo logramos implementarla con 5, la ganancia habrá sido cero porque en ambos casos requeriremos de dos circuitos integrados y en cambio sí hemos aumentado los costos de desarrollo, porque se invirtió el tiempo de un ingeniero en algo inútil.

Sin embargo, la cifra que dimos al principio, que el costo de todos los circuitos empleados en el sistema representa sólo el 10% de los mismos, hace aún más obsoletas las metas originales de diseño. Veamos, muy superficialmente, qué es lo que consume más dinero en un sistema y nos daremos cuenta que dichas metas deben ser modificadas drásticamente.

Para estos efectos, hemos empleado los datos proporcionados por Blakeslee |6| y que reproducimos en la Tabla 1-1. Según el autor, los datos corresponden a costos de 1974 y corresponden a una computadora de tamaño mediano y con un volumen moderado de producción.

De la tabla se observan dos cosas: que no hay un solo item en el que se pueda concentrar la minimización de costos y que todos los costos son casi proporcionales al número de circuitos integrados del sistema. Podría decirse que, un sistema con la mitad de los circuitos integrados, requeriría la mitad de circuitos impresos, la mitad de chasis, la mitad de la energía, la mitad de los esfuerzos de diseño, etc. De lo anterior parece evidente que la única forma real de minimizar costos es minimizar el número de circuitos integrados. Esto a su vez sugiere que si en un paquete de circuito integrado pudiéramos incluir funciones más complejas que la de una compuerta o de un flip-flop, lograríamos disminuír el total de circuitos integrados. Esto es posible hacerlo si se emplean circuitos MSI o LSI (integración a mediana y gran escala), con lo cuál aunque el precio por circuito aumenta, el costo total del sistema disminuye.

En resumen, la mejor forma de abatir costos en un sistema digital, es disminuir el número de circuitos integrados mediante el uso de circuitos de integración a mediana y gran escala. Otros factores que deben ser considerados son confiabilidad y simplicidad de mantenimiento y reparación, los cuales no vamos a analizar.

TABLA 1-1

Costos proporcionales al número de circuitos integrados, en un sistema digital.

Item	Cost/Unit	ICs/Unit[a]	Cost/IC[b]
PC board	$20/board	50/board	$0.40
PC connector	$5/board	50/board	0.10
Subrack (metalwork, guides, labor)	$200/subrack	1000/subrack	0.20
Backplane (subrack ground-power plane)	$100/subrack	1000/subrack	0.10
Wire wrap (automatic)	6¢/wire	1/wire	0.06
Power supplies ($1/W, 70% utilized)	$1.50/W	10/W	0.15
Rack (including doors, fans, power distributor)	$500/rack	4000/rack	0.13
IC ordering, receiving, and inventory	2¢/IC	1	0.02
IC testing	8¢/IC	1	0.08
PC board testing	$5/board	50/board	0.10
System checkout	$800/rack	4000/rack	0.20
Bypass capacitors (including insertion labor)	$1/board	50/board	0.02
IC insertion and soldering	6¢/IC	1	0.06
Interconnecting cables	$80/rack	4000/rack	0.02
Maintenance panel	$300/rack	4000/rack	0.07
System assembly	$200/rack	4000/rack	0.05
System packing and shipment	$200/rack	4000/rack	0.05
System design, drafting and prototype checkout (÷ 100 systems)	$2000/rack	4000/rack	0.50
Service cost for 5 years	$4000/rack	4000/rack	1.00
Total overhead cost per IC			= $3.31

[a] Since it is impractical to fill PC boards completely, an average (rather than maximum) number of ICs per board, subrack, and so on, is assumed.

[b] $\text{Cost/IC} = \dfrac{(\text{cost/unit})}{(\text{ICs/unit})}$

En los dos capítulos que siguen, estudiaremos los conceptos básicos de sistemas numéricos y álgebra de boole, en los cuales se fundamentan las técnicas digitales.

Referencias:

|1| G.M. Walker "Choosinn Sides in digital Watch Technology", Electronics Vol. 49, No. 12, June 10, 1976. pp. 91-99.

|2| A.A. Goldberg, "Digital Techniques promise to clarify the Television Picture", Electronics, Vol. 49 No.3, Feb. 5, 1976, pp 94-100.

|3| R. Gunlach, "Large-Scale Integration is ready to answer the call of telecommunications", Electronics, Vol. 50, No. 9, April 28, 1977, pp 93-108.

|4| D. M. Auslander, Y. Takahashi and M. Tonizuka, "The Next Generation of Single Loop Controllers", Journal of Dynamic Systems, Measurement and Control, Trans. ASME, Series G, Vol. 97, Sept. 1975, pp 280-282.

|5| G.M. Walker, "LSI Controls gaining in home appliances", Electronics Vol. 50, No. 8, April 14, 1977, pp 91-99.

|6| T. R. Blakeslee, "Digital Design with Standrad MSI and LSI", Wiley-Interscience, 1975.

CAPITULO 2: CONCEPTOS BASICOS DE SISTEMAS NUMERICOS Y DISEÑO LOGICO.

2.1.- Notación Posicional:

El sistema numérico que nosotros empleamos es el sistema decimal. En dicho sistema numérico, tenemos 10 símbolos para representar una cantidad, a saber: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. A estos símbolos les llamaremos dígitos.

Un número decimal, representando una cantidad de objetos, está compuesto por dígitos ubicados en posiciones que representan las unidades, decenas, centenas, etc. Por ejemplo, el número decimal 547 nos representa una cantidad de objetos, de la cual tenemos 5 centenas, 4 decenas y 7 unidades.

Este concepto de considerar a un número formado por dígitos en distintas posiciones, nos sugiere que podríamos utilizar otro tipo de representación. Consideremos el número decimal 47821 y representémoslo de la siguiente forma:

$$47821 = 40000 + 7000 + 800 + 20 + 1$$

Esta representación es equivalente a:

$$47821 = 4 \times 10000 + 7 \times 1000 + 8 \times 100 + 2 \times 10 + 1$$

o bién, empleando potencias de 10, podemos escribir

$$47821 = 4 \times 10^4 + 7 \times 10^3 + 8 \times 10^2 + 2 \times 10^1 + 1 \times 10^0$$

Podemos establecer entonces, que un número decimal de n dígitos, es la suma de coeficientes, cada uno con un peso asignado de acuerdo a su posición. En general, un número decimal de n dígitos se representa:

$$N_{10} = a_{n-1}(10)^{n-1} + a_{n-2}(10)^{n-2} + \ldots + a_1(10)^1 + a_0(10)^0 \quad (2.1)$$

o bién,

$$N_{10} = a_{n-1}\, a_{n-2} \cdots a_1 a_0 \quad (2.2)$$

que es la forma en que tradicionalmente escribimos un número.

En la expresión (2.2), la base 10 está sólo implicada. Pero, así como empleamos la base 10 para construir el sistema decimal, podríamos emplear cualquier otra base r. La ecuación (2.1) podemos generalizarla para un sistema base r, de la forma siguiente:

$$N_r = a_{n-1}\, r^{n-1} + a_{n-2}\, r^{n-2} + \ldots + a_1 r^1 + a_0 r^0 \quad (2.3)$$

El requisito para generar números en una base r es que requerimos de r símbolos (incluyendo el 0). En la tabla siguiente se presentan los símbolos de algunos de los sistemas numéricos más comunes:

TABLA 2-1

Base	Sistema Numérico	Símbolos
2	Binario	0, 1
8	Octal	0, 1, 2, 3, 4, 5, 6, 7
10	Decimal	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
16	Hexadecimal	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.

Hasta el momento, hemos considerado solamente números enteros. Para el caso de números fraccionarios, podemos extender las consideraciones anteriores. Consideremos el número decimal 0.234

$$0.234 = \frac{2}{10} + \frac{3}{100} + \frac{4}{1000}$$

o bién

$$0.234 = 2 \times 10^{-1} + 3 \times 10^{-2} + 4 \times 10^{-3}$$

o bién, en general para un número fraccionario decimal tenemos:

$$N_{10} = a_{-1} \times 10^{-1} + a_{-2} \times 10^{-2} + \ldots + a_{-(m-1)} \times 10^{-(m-1)} + a_{-m} \times 10^{-m} \quad (2.4)$$

Finalmente, para un número fraccionario base r

$$N_r = a_{-1} r^{-1} + a_2 r^{-2} + \ldots + a_{-(n-1)} r^{-(n-1)} + a_{-m} \times r^{-m} \quad (2.5)$$

Combinando las ecuaciones (2.3) y (2.5), obtenemos la representación de un número mixto (que tiene parte entera y parte fraccionaria), base r.

$$N_r = a_{n-1} r^{n-1} + a_{n-2} r^{n-2} + \ldots + a_1 r^1 + a_o r^0 + a_{-1} r^{-1} + a_{-2} r^{-2} + \ldots + a_{-m} r^{-m} \quad (2.6)$$

o bién

$$N_r = \sum_{i=-m}^{n} a_i \, r^i \quad (2.7)$$

A continuación explotaremos la representación posicional desarrollada para estudiar algunos sistemas numéricos de uso más frecuente. En particular, estudiaremos los sistemas binario, octal y hexadecimal.

2.2.- El Sistema Binario:

Este sistema es de gran importancia en sistemas digitales. Sabemos que en un sistema digital, la información se representa mediante dos niveles: 1 y 0, que pueden representar un voltaje y la ausencia de éste, o una intensidad de campo magnético y la ausencia de ésta, etc. En general, la filosofía de un sistema digital es la forma discreta y bivaluada en que se representa la información.

De lo anterior surge en forma obvia la importancia del sistema binario: el sistema binario emplea dos símbolos para representar una cantidad. Por lo tanto, debemos encontrar los mecanismos que nos permitan representar un número decimal en binario y viceversa. Debemos además establecer las reglas aritméticas de este sistema numérico. De esta forma, tendremos herramientas adecuadas para efectuar procesamientos numéricos mediante sistemas digitales.

2.2.- Conversión de Binario a Decimal:

El procedimiento para convertir un número binario a su equivalente decimal está basado en la ecuación (2.7).

Consideremos el número binario 110101101 y representémoslo en forma posicional:

$$110101101_2 = 1x2^8 + 1x2^7 + 0x2^6 + 1x2^5 + 0x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1x2^0$$

$$110101101_2 = 1x(256) + 1x(128) + 0x(64) + 1x(32) + 0x(16) + 1x(8) + 1x(4) + 0x(2) + 1x(1)$$

$$110101101 = 256 + 128 + 0 + 32 + 0 + 8 + 4 + 0 + 1$$

$$110101101 = 429_{10}$$

Si el número fuera fraccionario, procederíamos de acuerdo con la misma ecuación de la forma siguiente:

$$\cdot\ 101101 = 1x2^{-1} + 0x2^{-2} + 1x2^{-3} + 1x2^{-4} + 0x2^{-5} + 1x2^{-6}$$

$$\cdot\ 101101 = 1x(0.5) + 0x(0.25) + 1x(0.125) + 1x(0.0625) + 0x(0.03125) + 1x(0.0.15625)$$

$$\cdot\ 10101 = 0.5 + 0.125 + 0.0625 + 0.015625$$

$$\cdot\ 101101 = 0.703125$$

Podemos resumir entonces que:

PARA CONVERTIR UNA NUMERO BINARIO A DECIMAL, REPRESENTAMOS EL NUMERO EN SU FORMA POSICIONAL Y EFECTUAMOS LAS OPERACIONES INDICADAS.

De esta regla se pueden derivar otras alternativas que pueden resultar más simples. Por ejemplo, sumar las potencias de dos correspondientes a las posiciones distintas de cero. Consideremos el caso siguiente:

$$101101_2 = (\quad)_{10}$$

En este caso las potencias de dos correspondiente a las posiciones distintas de cero son: comenzando desde la posición menos significativa (el dígito del extremo derecho) tenemos: 2^0, 2^2, 2^3, 2^5; es decir sumamos: 1, 4, 8 y 32. El decimal equivalente será 45.

2.2.2.- Conversión de Decimal a Binario

Nuevamente, el procedimiento de conversión debemos buscarlo a partir de la ecuación (2.6). En este caso, el número se encuentra en decimal y queremos obtener su representación binaria. Luego, podemos escribir

$$N_{10} = a_{n-1}x2^{n-1} + a_{n-2}x2^{n-2} + ... + a_1x2^1 + a_0x2^0 \qquad (2.8)$$

Nuestro problema consiste en determinar los coeficientes a_{n-1}, a_{n-2}, ..., a_1, a_0. Como el sistema es binario, sabemos que podrán tomar sólo los valores 1 ó 0. Es decir, debemos determinar cuales coeficientes valen 1 y cuales valen 0.

Si dividimos ambos lados de la ecuación (2.8) por 2, obtenemos un cociente Q_1 que es número entero y un residuo a_0:

$$\frac{N_{10}}{2} = Q_1 + a_0 2^{-1} \qquad (2.9)$$

donde

$$Q_1 = a_{n-1}x2^{n-2} + a_{n-2}x2^{n-3} + ... + a_2x2^1 + a_1x2^0 \qquad (2.10)$$

El residuo a_0 obtenido de esta manera, es dígito menos significativo de la representación binaria de N_{10}. Si ahora procedemos de igual forma con Q_1, obtendremos un cociente entro Q_2 y un residuo a_1:

$$\frac{Q_1}{2} = Q_2 + a_1 \times 2^{-1} \qquad (2.11)$$

donde

$$Q_2 = a_{n-1} \times 2^{n-3} + a_{n-2} \times 2^{n-4} + ... + a_3 \times 2^1 + a_2 \times 2^0 \qquad (2.12)$$

Si procedemos de esta forma hasta que el cociente resultante sea cero, habremos obtenido todos los coeficientes y por lo tanto, la representación binaria del número en cuestión. Por ejemplo, encontremos la representación binaria del número decimal 429_{10}.

Cociente	Residuo	Posición
429 ÷ 2	1	a_0
214 ÷ 2	0	a_1
107 ÷ 2	1	a_2
53 ÷ 2	1	a_3
26 ÷ 2	0	a_4
13 ÷ 2	1	a_5
6 ÷ 2	0	a_6
3 ÷ 2	1	a_7
1 ÷ 2	1	a_8
0		

Luego

$$429_{10} = 110101101$$

Otra forma de proceder es buscar la potencia de dos más alta que esté contenida en el número original; por ejemplo, si el número es 489_{10}, la potencia más cercana es $2^8 = 256$. A continuación buscamos la potencia de 2 más alta contenida en $489-2^8 = 233$, que en este caso es $2^7 = 128$. Luego, buscamos la potencia de 2 más alta contenida en $233-2^7 = 105$, que es 2^6. Procediendo de esta forma, encontraremos las potencias de 2 cuya suma es igual al número original. En el ejemplo que estábamos analizando, dichas potencias son:

$$\begin{aligned} 2^8 &= 256 \\ 2^7 &= 128 \\ 2^6 &= 64 \\ 2^5 &= 32 \\ 2^3 &= 8 \\ 2^0 &= 1 \\ &\overline{489} \end{aligned}$$

De esta forma, obtenemos todos los coeficientes iguales a 1 de la representación binaria del número original. Todas las potencias de 2 que no aparecen en nuestra lista, serán aquellas correspondientes a los coeficientes iguales a cero.

En nuestro ejemplo, tendremos:

$$489_{10} = 111101001_2$$

El problema de convertir a binario un número decimal fraccionario, se resuelve en forma similar. Si para números enteros, los coeficientes de la representación binaria los obteníamos por medio de divisiones sucesivas, para números fraccionarios los obtenemos mediante multiplicaciones sucesivas. Esto se puede comprobar observando la ecuación (2.5)

$$N_{10} = a_{-1}x2^{-1} + a_{-2}x2^{-2} + ... + a_{-m}x2^{-m}$$

donde N_{10} es el número decimal fraccionario y los coeficientes a_{-1}, a_{-2}, ..., a_{-m} corresponden a la representación binaria de N_{10} y son los que debemos determinar. Multiplicando ambos lados de la ecuación, obtenemos:

$$2xN_{10} = a_{-1} + a_{-2}x2^{-1} + ... + a_{-m}x2^{-m+1}$$

Es decir, hemos obtenido el coeficiente a_{-1}, que podrá ser 0 ó 1. Procediendo de esta forma podemos obtener los coeficientes restantes. Consideremos el siguiente ejemplo:

Convertir 0.703125_{10} a binario.

$0.703125 \times 2 = 1 + 0.406250$ $\quad a_{-1} = 1$
$0.406250 \times 2 = 0 + 0.812500$ $\quad a_{-2} = 0$
$0.812500 \times 2 = 1 + 0.625000$ $\quad a_{-3} = 1$
$0.625000 \times 2 = 1 + 0.250000$ $\quad a_{-4} = 1$
$0.250000 \times 2 = 0 + 0.500000$ $\quad a_{-5} = 0$
$0.500000 \times 2 = 1 + 0.000000$ $\quad a_{-6} = 1$
$0.000000 \times 2 = 0 + 0.000000$

Luego

$$0.703125_{10} = 101101_2$$

En el ejemplo vemos que los coeficientes iguales a 1 son generados cuando al multiplicar el número fraccionario se obtiene una parte entera. Además se observa que el proceso termina cuando el número fraccionario se hace 0. Esto último no es siempre el caso. Consideremos el ejemplo siguiente:

Convertir 0.59_{10} a binario

. $39 \times 2 = 0 + 0.78$ $\quad a_{-1} = 0$
. $78 \times 2 = 1 + 0.56$ $\quad a_{-2} = 1$
. $56 \times 2 = 1 + 0.12$ $\quad a_{-3} = 1$
. $12 \times 2 = 0 + 0.24$ $\quad a_{-4} = 0$
. $24 \times 2 = 0 + 0.48$ $\quad a_{-5} = 0$
. $48 \times 2 = 0 + 0.96$ $\quad a_{-6} = 0$
. $96 \times 2 = 1 + 0.92$ $\quad a_{-7} = 1$

. $92 \times 2 = 1 + 0.84$ $\quad a_{-8} = 1$

. $84 \times 2 = 1 + 0.68$ $\quad a_{-9} = 1$

. $68 \times 2 = 1 + 0.36$ $\quad a_{-10} = 1$

. $36 \times 2 = 0 + 0.72$ $\quad a_{-11} = 0$

. $72 \times 2 = 0 + 0.44$ $\quad a_{-12} = 1$

Como puede observarse, aún cuando sigamos con el proceso éste no terminará. Es decir, 0.39 no se puede representar en forma exacta en binario; en este caso,

$$0.39 \simeq 0.011000111101_2$$

El error introducido en la representación será menor que el dígito menos representativo, es decir:

$$\varepsilon < 2^{-12} = 1/4096 = 0.000244141$$

Si convertimos a decimal el número binario obtenido, tendremos:

$$0.011000111101_2 = 0.389892578$$

Es decir, el error es de 0.000107472

Este problema se traduce en un problema de exactitud de la representación y existen varias formas de atacarlo. Por lo general la limitación más grande es el número de bits o "longitud de palabra" con que se cuente para representar un número. Por ejemplo, en la mayoría de los microprocesadores, dicha longitud de palabra es de 8 bits, en cambio, en computadoras grandes de propósito general, la longitud de palabra llega a 64 bits, como es el caso de la CDC 6400/6600.

2.2.3.- Aritmética Binaria:

Así como tenemos aritmética decimal, existe una aritmética binaria, cuyas reglas son idénticas a la decimal pero en este caso tenemos solo dos símbolos: 0 y 1 para representar un número.

A continuación se dan dos tablas aritméticas: suma y Multiplicación

Tabla 2.2: Suma binaria

Augendo	+ Adendo	= Resultado	+ Acarreo
0	0	0	
0	1	1	
1	0	1	
1	1	0	+ 1

Tabla 2.3: Multiplicación Binaria

X	0	1
0	0	0
1	0	1

Veamos algunos ejemplos de suma y multiplicación de números binarios

1) Sumar 1011 + 1101 (= 11 + 13)

```
(1)  (1)  (1)  (1)      ← Acarreo
      1    0    1    1
      1    1    0    1
-----------------------
 1    1    0    0    0   ( = 34)
```

2) Multiplicar 1001 x 1011 (9 x 11)

```
1001 x 1011
   1001
  1001
 0000
1001
-------
1100011      (= 99)
```

La substracción binaria será analizada posteriormente cuando veamos la representación de números negativos. Finalmente, la división no tiene mayor interés práctico, ya que puede ser implementada mediante restas.

2.3.- Sistemas Octal y Hexadecimal:

Anteriormente se hizo notar que el sistema binario es el de mayor importancia para trabajar con sistemas digitales. Sin embargo, los humanos estamos acostumbrados a trabajar con el sistema decimal. Resulta bastante difícil tener una idea de los órdenes de magnitud de cantidades representadas en binario. Por esto se ha buscado un sistema numérico lo más cercano al decimal y que a la vez sea muy cercano al decimal. Dos de estos sistemas numéricos son el octal (base 8) y el hexadecimal (base 16). El primero resulta obvio que se acerca bastante al decimal: cuenta con ocho símbolos para representar una cantidad. En el caso del sistema hexadecimal, esta ventaja ya no es tan obvia puesto que requerimos de 16 símbolos y por lo tanto hay que emplear 6 letras (A, B, C, D, E, F) para suplir los símbolos numéricos faltantes. Esto obviamente introduce un grado más de complejidad y, aparentemente, nos hace más difícil la vida. Sin embargo, estudiemos un poco estos dos sistemas nu-

méricos y veremos que es posible justificar su empleo, más aún cuando se está haciendo cada vez más empleado con el advenimiento de los microprocesadores.

La característica más importante de los sistemas octal y hexadecimal es que sus bases (8 y 16) son potencias de 2 y por lo tanto, se relacionan directamente con el sistema binario. En efecto,

$$8 = 2^3$$
$$16 = 2^4$$

Estas relaciones nos sugieren que para representar un número octal (base 8) en binario, requerimos de tres dígitos (o bits - de binary digit) y para representar uno hexadecimal, requerimos cuatro dígitos. Esto se ilustra en la tabla siguiente:

Tabla 2.4

Hexadecimal	Octal	Binario
0	0	0 0 0 0 0
1	1	0 0 0 0 1
2	2	0 0 0 1 0
3	3	0 0 0 1 1
4	4	0 0 1 0 0
5	5	0 0 1 0 1
6	6	0 0 1 1 0
7	7	0 0 1 1 1
8	10	0 1 0 0 0
9	11	0 1 0 0 1
A	12	0 1 0 1 0
B	13	0 1 0 1 1
C	14	0 1 1 0 0
D	15	0 1 1 0 1
E	16	0 1 1 1 0
F	17	0 1 1 1 1
10	20	1 0 0 0 0
11	21	1 0 0 0 1
12	22	1 0 0 1 0
:	:	
1F	37	1 1 1 1 1

De la tabla vemos que los ocho símbolos octales son representados por las ocho combinaciones diferentes de tres bits. Cuando la cantidad octal requiere de más de un dígito para ser expresada, su representación en binario se obtiene simplemente reemplazando cada dígito octal por sus tres bits correspondientes a su representación en binario. Veamos esto con un ejemplo:

Representar 712_8 en binario:

Procedemos reemplazando cada dígito octal por su representación en binario:

$7_8 = 111_2$

$1_8 = 001_2$

$3_8 = 011_2$

Luego

$713_8 = 111\ 001\ 011_2$

Comprobemos ésto convirtiendo ambas cantidades a decimal:

$$713_8 = 7 \times 8^2 + 1 \times 8^1 + 3 \times 8^0 = 7 \times 64 + 8 + 3 =$$

$$= 448 + 8 + 3 = 459_{10}$$

$$111001011 = 1x2^8 + 1x2^7 + 1x2^6 + 0x2^5 + 0x2^4 + 1x2^3 +$$
$$+ 0x2^2 + 1x2^1 + 1x2^0$$
$$= 256 + 128 + 64 + 0 + 0 + 8 + 0 + 2 + 1 =$$
$$= 459_{10}$$

Esto mismo nos sugiere que para convertir un número binario a octal, podemos proceder agrupando (desde la derecha hacia la izquierda) los dígitos binarios en grupos de tres y luego reemplazar directamente por su equivalente en octal.

Convertir 101011011_2 a octal

$$\underbrace{101}_{5}\ \underbrace{011}_{3}\ \underbrace{011}_{3}$$

Luego:

$101011011_2 = 533_8$

¿Qué sucede cuando el último grupo tiene menos de tres bits? Muy simple: agregamos tantos ceros a la izquierda como sean necesarios. Los ceros a la izquierda no alteran la cantidad.

Convertir 1101101111_2 a octal

$$\underbrace{0\ 0\ 1}_{1}\ \underbrace{101}_{5}\ \underbrace{101}_{5}\ \underbrace{111}_{7}$$

Luego:

$1101101111_2 = 1557_8$

Para el caso de números hexadecimales, los procedimientos de conversión son idénticos, excepto que ahora cada dígito hexadecimal requiere de cuatro bits y viceversa.

1) Convertir $E70A_{16}$ a binario

$$E_{16} = 1110_2$$
$$7_{16} = 0111_2$$
$$0_{16} = 0000_2$$
$$A_{16} = 1010_2$$

Luego

$$E70A_{16} = 1110011100001010_2$$

2) Convertir 1111101011101 a hexadecimal

$$\overbrace{0001}^{1}\ \overbrace{1111}^{F}\ \overbrace{0101}^{5}\ \overbrace{1101}^{D}$$

Luego

$$1111101011101_2 = 1F5D_{16}$$

Aún cuando existen métodos para convertir entre cualquier par de bases, y por lo tanto, existen para convertir de octal a decimal y viceversa y de hexadecimal a decimal y viceversa, éstos no serán considerados en las presentes notas. El interesado puede consultar, por ejemplo, el libro de A. Barna y D.I. Porat [1].

Sin embargo, aún cuando no es muy eficiente, si se desea convertir un número decimal a octal o hexadecimal, se puede convertir primero de decimal a binario mediante el procedimiento de la sección 2.2.2. y luego convertir de binario a octal o hexadecimal, que no es más que un paso mecánico.

Regresando a nuestra justificación del empleo de los sistemas octal y hexadecimal, vemos que cada dígito octal requiere de tres bits para su representación y en cambio cada dígito hexadecimal requiere de cuatro bits. Este hecho tan simple es la base de la preferencia por el sistema hexadecimal. Casi todos los sistemas digitales están organizados en base a registros o palabras con un número de bits que es una potencia de dos. Así vemos que muchos microprocesadores (I8080, M6800, COSMAC, SC/MP, etc.) son de ocho bits; muchas minicomputadoras son de 16 o 32 bits y lo mismo sucede con varias de las computadoras grandes (IBM: 32 bits, CDC: 64 bits). Esto hace que sea muy simple agrupar bits en grupos de 4 y representarlos directamente en hexadecimal. No sucede así con el octal, en que la organización de palabras en múltiplos de 3 no se encuentran muy frecuentemente en sistemas digitales. De hecho, las instrucciones de la gran mayoría de los microprocesadores vienen especificados en hexadecimal.

2.4.- Representación de Números Negativos:

Hasta el momento hemos considerado solamente el problema de representar números positivos. Sin embargo, con mucha frecuencia debemos trabajar con números negativos. Una forma de atacar el problema podría ser emplear un bit adicional y hacerlo 0 cuando la cantidad sea positiva y 1 cuando sea negativa. Aún cuando esto es perfectamente válido, tiene el inconveniente que requerimos de un bit extra que significa contar con un elemento electrónico adicional en todas aquellas partes del sistema en que debamos "recordar" el signo de la cantidad que se esté almacenando. Esto no es muy deseable desde el punto de vista económico.

Afortunadamente existe una forma de representar números negativos que no requiere de un bit extra y que además tiene grandes cualidades desde el punto de vista aritmético. Esta forma se llama complemento a la base.

El problema puede ser visto de la forma siguiente: si tenemos que restar dos números:

$$7 - 2 = 5$$

podemos invertir el problema y hacerlo de la forma siguiente:

$$7 + (10 - 2) = 7 + 8 = \cancel{1}5$$

El proceso de restarle a 10 el sustraendo es lo que se conoce como el complemento a 10. Veamos otro ejemplo:

$$374 - 254 = ?$$

en primer lugar obtenemos el complemento a 10 del sustraendo 254:

$$\begin{array}{r} 1000 \\ 254 \\ \hline 746 \end{array}$$

Ahora sumamos:

$$374 + 746 = \cancel{1}120$$

el resultado lo obtenemos ignorando el acarreo a la posición más significativa.

Todo esto parece hacer más complicado (para nosotros) el simple proceso de restar dos números. Sin embargo, veremos que aún cuando para nosotros puede resultar complicado, puede implementarse en forma muy simple en un sistema digital.

Anteriormente vimos el complemento a la base decimal es decir, el complemento a 10. Veamos ahora el complemento a la base binaria, o complemento a dos. El complemento a dos lo obtenemos restando la cantidad a la potencia de dos inmediatamente superior. Obtengamos el complemento a dos de 1101:

$$
\begin{array}{ll}
10000 & \leftarrow 2^4 \\
-1101 & \leftarrow \text{Cantidad a complementar} \\
\hline
0011 & \leftarrow \text{Complemento a dos}
\end{array}
$$

Todavía sigue siendo complicado. Sin embargo, se puede simplificar, Consideremos el mismo número y cambiemos los 0 por 1 y los 1 por 0 y luego sumémosle 1:

$$
\begin{array}{ll}
1101 & \leftarrow \text{Cantidad original} \\
0010 & \leftarrow \text{Complemento} \\
+\ 1 & \\
\hline
0011 & \leftarrow \text{Complemento a dos}
\end{array}
$$

Lo que hicimos en este ejemplo fue obtener el complemento a 1 de la cantidad original y luego sumarle. El complemento a 1, formalmente se obtiene restando de 2^n-1 la cantidad a complementar. Por ejemplo, obtengamos el complemento a 1 de 101101:

$$
\begin{array}{ll}
1\ 1\ 1\ 1\ 1\ 1 & \leftarrow 2^6 - 1 \\
-\ 1\ 0\ 1\ 1\ 0\ 1 & \leftarrow \text{Cantidad a complementar} \\
\hline
0\ 1\ 0\ 0\ 1\ 0 & \leftarrow \text{Complemento a uno.}
\end{array}
$$

Sin embargo, nos podemos dar cuenta que este procedimiento es innecesario, ya que si "negamos" cada dígito (cambiamos los 1 por 0 y viceversa) estamos logrando el mismo efecto.

Podemos concluír que: El complemento a 2 de un número binario se obtiene sumando 1 al complemento a 1 de dicho número.

Regresemos ahora al problema de la resta de dos números. Consideremos primero la resta binaria siguiente:

$$
\begin{array}{ll}
\text{Minuendo} & 10001 \\
\text{Sustraendo} & -\ 01011 \\
\hline
 & 00110
\end{array}
$$

La misma operación la podemos realizar sumando al minuendo el complemento a dos del sustraendo:

$$
\begin{array}{ll}
\text{Minuendo} & 10001 \\
\text{Complemento a 2 del Sustraendo.} & 10101 \\
\hline
 & \not{1}00110
\end{array}
$$

el resultado lo obtenemos ignorando el dígito de acarreo. El resultado es entonces: 00110.

Consideremos otro ejemplo:

Minuendo:	11001	11001	Minuendo
Sustraendo:	-101	+ 11011	Complemento a dos del Sustraendo
	= 10100	X10100	

Cuando la magnitud del sustraendo es mayor que el minuendo, no habrá acarreo y el resultado final será el negativo del complemento a 2 del resultado de la suma. Para ver más claramente lo anterior, consideremos una versión decimal de lo anterior:

$$2 - 9 = 2 + (10-9) = 2 + 1 = 3$$

Luego, el resultado final será -(10-3) = 7. Veamos un ejemplo binario:

101	Minuendo	101	
- 11011	Sustraendo	+00101	Complemento a 2
- 10110		01010	Resultado de la suma
		-10110	Complemento a 2 del resultado

Podemos concluír lo siguiente:

AL EFECTUAR UNA RESTA EMPLEANDO COMPLEMENTO A DOS, EL ACARREO A LA POSICION MAS SIGNIFICATIVA NOS DARA EL SIGNO DEL RESULTADO. SI EL ACARREO ES 1, LA RESPUESTA SERA POSITIVA (CASO EN QUE EL MINUENDO > SUSTRAENDO); SI EL ACARREO ES 0, EL RESULTADO FINAL SERA EL NEGATIVO DEL COMPLEMENTO A DOS DEL RESULTADO DE LA SUMA.

Para concluír, cabe hacer notar que la obtención del complemento a dos de un número es muy simple de implementar digitalmente y que esto nos evita el tener que efectuar restas.

2.5.- Algunos códigos más usados

Dado que internamente, una computadora o un sistema digital, almacena y procesa información en binario, es necesario contar con códigos que permitan traducir números y caracteres a una forma compatible con el tipo de representación interna de un sistema digital. En particular, nos interesan los códigos más comunes: BCD, hexadecimal y alfanumérico.

2.5.1.- Código BCD:

El código BCD ("Binary Coded Decimal") se emplea para representar directamente los números decimales en binarios. La representación en BCD se obtiene reemplazando cada dígito de un número decimal, por un grupo de cuatro dígitos binarios. En la tabla siguiente se muestra la representación en BCD de los nueve dígitos decimales.

Decimal	B C D
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

Ejemplo: Representar 375 en BCD. De la tabla anterior obtenemos la representación en BCD de cada dígito decimal y lo reemplazamos directamente:

$$\underbrace{0\ 0\ 1\ 1}\ \underbrace{0\ 1\ 1\ 1}\ \underbrace{0\ 1\ 0\ 1}_{BCD} = 375_{10}$$

En binario, el mismo número se representa como sigue:

$$1\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 1_2 = 375_{10}$$

Ejemplo: Convertir el número BCD 0101011110000001 a decimal.

Agrupamos de a cuatro dígitos binarios el número BCD y reemplazamos cada grupo de dígitos por su equivalente decimal.

$$\underbrace{0\ 1\ 0\ 1}\ \underbrace{0\ 1\ 1\ 1}\ \underbrace{1\ 0\ 0\ 0}\ \underbrace{0\ 0\ 0\ 1}_{BCD} = 5781_{10}$$

Ejemplo: convertir el número BCD 0 1 0 1 1 1 0 0 0 0 0 1 a decimal.

Procediendo como en el ejemplo anterior

$$\underbrace{0\ 1\ 0\ 1}_{5}\ \underbrace{1\ 1\ 0\ 0}_{\text{Ilegal}}\ \underbrace{0\ 0\ 0\ 1}_{1}$$

En este caso, el grupo 1100 no representa ningún dígito decimal codificado en BCD y por lo tanto es un código ilegal.

2.5.2.- Código Hexadecimal:

El código binario puro de cuatro bits, en el que se usan todas las combinaciones posibles constituye el código hexadecimal. Las seis combinaciones no usadas en BCD se simbolizan con letras, A a F.

DECIMAL	HEXADECIMAL	DENOMINACION
0	0 0 0 0	0
⋮	⋮	⋮
9	1 0 0 1	9
10	1 0 1 0	A
11	1 0 1 1	B
12	1 1 0 0	C
13	1 1 0 1	D
14	1 1 1 0	E
15	1 1 1 1	F

Ejemplo: El número binario 1 1 0 0 0 1 0 1 corresponde a C 5 hexadecimal y a 12 x 16 + 5 = 197 decimal.

2.5.3.- Códigos Alfanuméricos:

Este tipo de códigos permite representar en binario cualquier tipo de caracter: dígitos, letras y símbolos especiales. Los dos códigos alfanuméricos más empleados son el ASCII (American Standard Code for Information Interchange Code). El código ASCII es empleado por todos los fabricantes de minicomputadores y microcomputadoras.

Usa 7 bits (más uno de paridad, para seguridad en las comunicaciones). Además de información alfanumérica, el código contiene posiciones que son interpretadas por las terminales de comunicaciones para realizar ciertas funciones ("retorno del carro", "avance del papel", "conectar perforador de cinta", etc.)

Se adjunta una tabla del código ASCII.

2.6.- Algebra de Boole

Con esta sección iniciamos el estudio de otro aspecto básico de las técnicas digitales: Las herramientas matemáticas que nos permitirán describir sistemas digitales en forma exacta.

En general, un sistema digital puede caracterizarse por dos aspectos: capacidad de tomar decisiones y capacidad de almacenar información.

En esta sección estableceremos las bases para lograr la implementación física del proceso de toma de decisiones lógicas.

Para implementar un proceso de toma de decisiones, debemos en primer lugar establecer un modelo de dicho proceso. Este modelo surgió de la forma en que toma decisiones el hombre. De este estudio surgió la lógica proposicional y posteriormente, la lógica matemática.

Uno de los primeros en formular una teoría fue George Boole, quién en 1849 publicó su libro "An Investigation of the Laws of Thought, on which are Founded the Mathematical Theories of Logic and Probability", en donde presentó una formulación algebráica del proceso del pensamiento lógico y del razonamiento. Esta formulación se ha llamado Algebra de Boole y se emplea extensivamente en el estudio de circuitos combinacionales.

2.6.1.- Postulados de Huntington:

Como toda teoría deductiva, el Algebra de Boole está basada en un conjunto de postulados. Estos postulados fueron establecidos por Huntington en 1904.

Postulado No. 1: Existe un conjunto K de elementos, sujetos a una relación de equivalencia denotada "=", que satisface el principio de substitución.

(por substitución se entiende que si a = b, se puede substituir a por b en cualquier expresión que contenga a b, sin afectar la validéz de la expresión).

Postulado No. 2: Se define una regla de combinación "+", tal que si $A \in K$ y $B \in K$, $(A + B) \in K$.

Postulado No. 3: Se define una regla de combinación ".", tal que $A \cdot B \in K$ si $A \in K$ y $B \in K$.

Postulado No. 4: Existe un elemento "0" tal que: $A + 0 = A \; \forall \; A \in K$.

Postulado No. 5: Existe un elemento "1" tal que $A \cdot 1 = A \; \forall \; A \in K$.

Postulado No. 6: Si $A \in K$ y $B \in K$, luego: conmutación de + $A + B = B + A$

Postulado No. 7: Si $A \in K$ y $B \in K$, luego: conmutación de · $A \cdot B = B \cdot A$

Postulado No. 8: Si $A \in K$, $B \in K$ y $C \in K$, luego : distributiva $A+(B \cdot C) = (A+B) \cdot (A+C)$

Postulado No. 9: Si $A \in K$, $B \in K$ y $C \in K$, luego: distributiva $A \cdot (B+C) = A \cdot B + A \cdot C$

Postulado No. 10: Para todo elemento $A \in K$, existe un elemento $\overline{A}$ tal que:

$$A \cdot \overline{A} = 0$$
$$A + \overline{A} = 1$$

Postulado No. 11: Existen al menos dos elementos $x \varepsilon K$ y εK, tales que $x \neq y$.

2.6.2.- Diagramas de Venn:

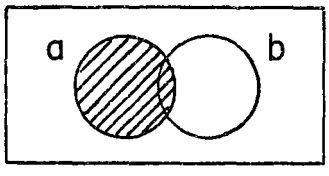
Dado que (la teoría de conjuntos) el álgebra de conjuntos es un álgebra de Boole, podemos utilizar los diagramas de Venn, empleados en teoría de conjuntos.

La correspondencia entre álgebra de conjuntos y de Boole es la siguiente

Algebra de Boole		Algebra de conjuntos	
·	(and)		(intersección)
+	(or)		(unión)
1		I	(universo)
0		Ø	(vacío)

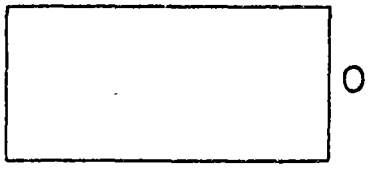
En los diagramas de Venn, los conjuntos se muestran como entornos cerrados (cuadrados, círculos, elipses, etc.)

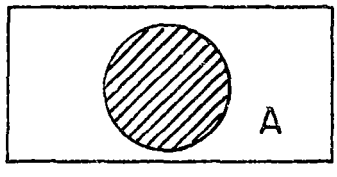
Ejemplo:



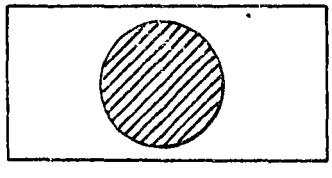
El conjunto a es el sombreado

Postulado No. 4: $A + 0 = A$





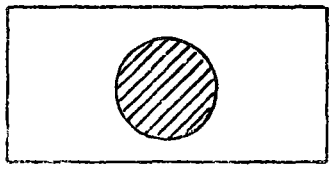
Unión



$A + 0$

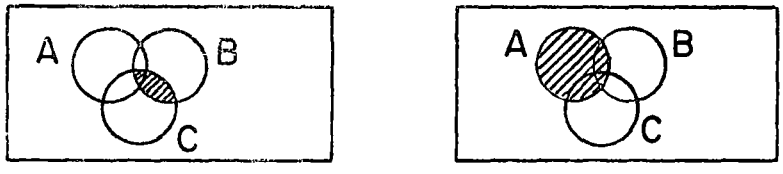
Postulado No. 5: $A \cdot 1 = A$

Intersección

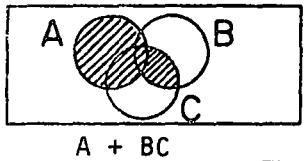


$A \cdot 1$

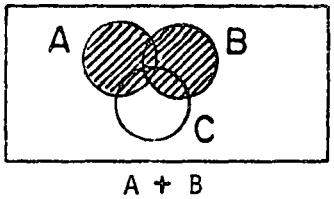
Postulado No. 8: $A + BC = (A+B)(A+C)$



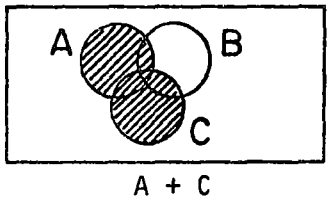
Unión



$A + BC$

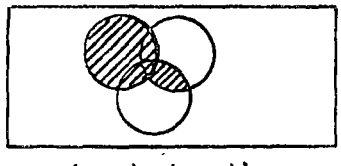


$A + B$



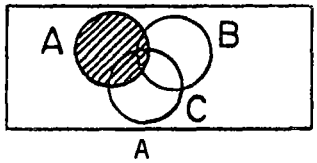
$A + C$

Intersección

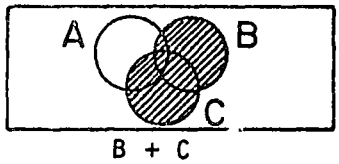


$(A+B)\ (A+C)$

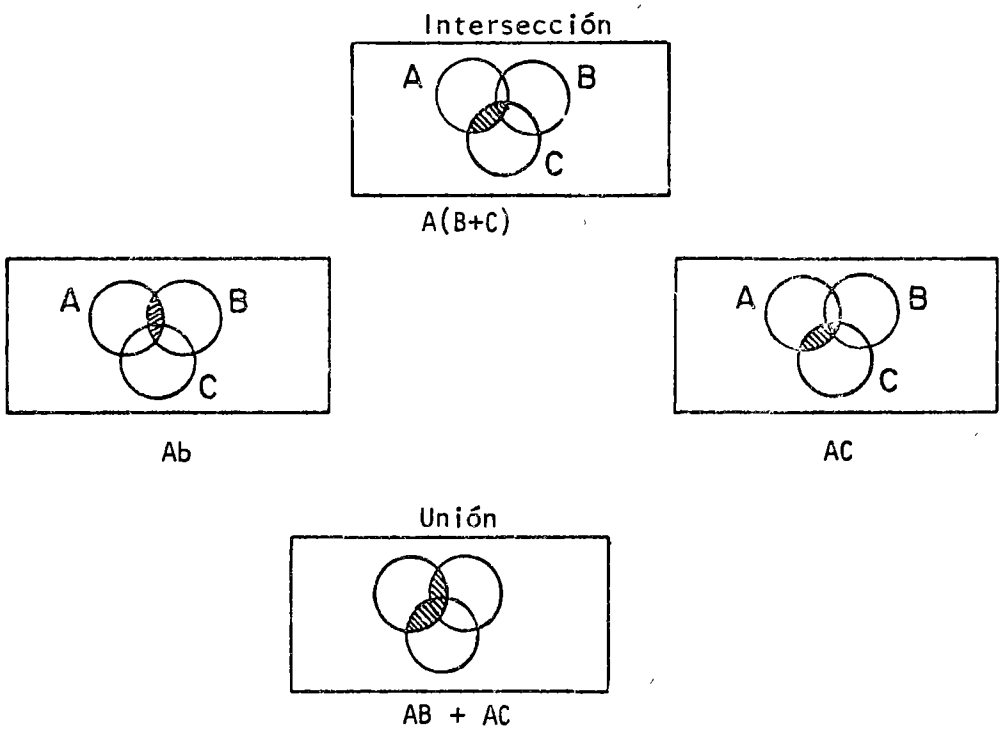
Postulado No. 9: $A \cdot (B+C) = AB + AC$



A



$B + C$



2.6.2.- Teoremas Fundamentales del Algebra de Boole:

A continuación presentamos, sin demostración, aquellos teoremas del álgebra de Boole que nos serán útiles en el trabajo subsecuente. Varios de estos teoremas pueden ser verificados empleando los diagramas de Venn.

T.1.- El 0 y el 1 de los postulados 4 y 5 son elementos únicos.

T.2.- Para todo $a \in K$

i) $a + a = a$

ii) $a \cdot a = a$

T.3.- Para todo $a \in K$

i) $a + 1 = 1$

ii) $a \cdot 0 = 0$

T.4.- Los elementos 1 y 0 son distintos, y $\overline{1} = 0$.

T.5.- Para todo par de elementos a y b en K,

i) $a + a \cdot b = a$

ii) $a(a + b) = a$

T.6.- El elemento $\bar{a}$ definido en el postulado 10 es único, para todo $a \in K$.

T.7.- Para todo $a \in K$, $\bar{\bar{a}} = a$.

T.8.- $a|(a+b) + c| = |(a+b) + c|a = a$

T.9.- Para todo $a \in K$, $b \in K$ y $c \in K$,

i) $a + (b + c) = (a + b) + c$
ii) $a \cdot (b \cdot c) = (a \cdot b) \cdot c$

T.10.- Para todo $a \in K$ y $b \in K$,

i) $a + \bar{a}b = a + b$
ii) $a(\bar{a} + b) = ab$

T.11.- Leyes de De Morgan:
Para todo $a \in K$ y $b \in K$

i) $\overline{a + b} = \bar{a} \cdot \bar{b}$
ii) $\overline{a \cdot b} = \bar{a} + \bar{b}$

T.12.- Para todo $a \in K$, $b \in K$ y $c \in K$,

i) $ab + \bar{a}c + bc = ab + \bar{a}c$
ii) $(a + b)(\bar{a}+c)(b + c) = (a + b)(\bar{a} + c)$

T.13.- Para todo $a \in K$ y $b \in K$,

i) $ab + a\bar{b} = a$

ii) $(a + b)\ (a + \bar{b}) = a$

T.14.- Para todo $a \in K$, $b \in K$ y $c \in K$,

i) $ab + a\bar{b}c = ab + ac$
ii) $(a + b)(a + \bar{b} + c) = (a+b)(a + c)$

T.15.- Para todo $a \in K$, $b \in K$ y $c \in K$,

i) $ab + \bar{a}c = (a + c)(\bar{a} + b)$
ii) $(a + b)(\bar{a} + c) = ac + \bar{a}b$

2.6.3.- Funciones Booleanas

Los postulados y teoremas del álgebra Booleana presentados en las secciones anteriores, se dieron en términos generales y sin especificar los elementos del conjunto K, por lo cual, los resultados son válidos para cualquier álgebra de Boole. En la discusión que sigue, nos concentraremos en un álgebra donde $K = 0,1$. Esta formulación es comúnmente llamada "álgebra de conmutación".

El concepto de función es conocido del álgebra ordinaria. Las funciones booleanas representan el concepto correspondienta para álgebra de Boole y se pueden definir como sigue:

Sean $X_1, X_2, \ldots, X_n$ símbolos que llamaremos variables, cada uno de las cuales puede representar un 1 o un 0 de un álgebra Booleana (se dice que 0 ó 1 son el valor de la variable).

Sea $f(X_1, X_2, \ldots, X_n)$ una función Booleana de las variables $X_1, X_2, \ldots, X_n$.

La función f representa el valor 0 ó 1 dependiendo de los valores asignados a $X_1, X_2, \ldots, X_n$.

Una función Booleana se puede describir mediante una expresión booleana como sigue:

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

Si A = 1, B = C = 0, entonces f = 1 como se verifica a continuación:

$$\begin{aligned} f(1,0,0) &= 1\cdot 0 + \overline{1}\cdot 0 + 1\cdot\overline{0} \\ &= 1\cdot 0 + 0\cdot 0 + 1\cdot 1 = 0 + 0 + 1 \\ &= 1 \end{aligned}$$

Una alternativa en la descripción de funciones Booleanas, se puede dar mediante el uso de una tabla de verdad.

Una tabla de verdad presenta todas las combinaciones posibles de valores posibles de f.

Si evaluamos la función $f(A,B,C) = AB + \overline{A}C + A\overline{C}$ para todas las combinaciones posibles de las variables encontramos:

$$\begin{aligned} f(0,0,0) &= 0 \\ f(0,0,1) &= 1 \\ f(0,1,0) &= 0 \\ f(0,1,1) &= 1 \\ f(1,0,0) &= 1 \\ f(1,0,1) &= 0 \\ f(1,1,0) &= 1 \\ f(1,1,1) &= 1 \end{aligned}$$

Si estos valores se listan en forma tabular, obtenemos la tabla de verdad.

A B C	f(A,B,C)
0 0 0	0
0 0 1	1
0 1 0	0
0 1 1	1
1 0 0	1
1 0 1	0
1 1 0	1
1 1 1	1

En la tabla siguiente se muestra la tabla de verdad para $f(X_1, X_2, \ldots, X_n)$

X_1 X_2 ..... X_n	$f(X_1, X_2, \ldots, X_n)$
0 0 0	a_0
0 0 1	a_1
⋮	⋮
1 1 0	a_{2^n-2}
1 1 1	a_{2^n-1}

Como hay n variables y cada variable tiene dos valores posibles, hay 2^n formas de asignar valores a las n variables; por lo tanto, la tabla de verdad tendrá 2^n files.

Además, para cualquier combinación de las variables $X_1, X_2, \ldots X_n$, hay dos valores posibles para la función general $f(X_1, X_2, X_3, \ldots, X^n)$; por lo tanto podemos hacer 2^N tablas de verdad para n variables, donde $N = 2^n$.

Es decir, para n variables hay 2^{2^n} funciones Booleanas posibles.

Aún para un número pequeño de variables, el número de funciones Booleanas posibles es enorme, como lo demuestra la tabla siguiente:

n	2^n	2^{2^n}
0	1	2
1	2	4
2	4	16
3	8	256
4	16	65.536
⋮	⋮	⋮

A continuación se dan las tablas para funciones de 0, 1 y 2 variables:

n = 0: $f_1 = 0$
$f_2 = 1$

n = 1:

A	f_1 (A)	f_2 (A)	f_3 (A)	f_4 (A)
0	0	1	0	1
1	0	0	1	1

$f_1(A) = 0$
$f_2(A) = \bar{A}$
$f_3(A) = A$
$f_4(A) = 1$

A B	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0 0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0 1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1 0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1 1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

$f_0(A,B) = 0$ = CERO
$f_1(A,B) = A\bar{B}$ = AND
$f_2(A,B) = A\bar{B}$ = INHIBIDORA
$f_3(A,B) = A\bar{B} + AB = A$ = IDENTIDAD A
$f_4(A,B) = \bar{A}B$ = INHIBIDORA
$f_5(A,B) = \bar{A}B + AB = B$ = IDENTIDAD B
$f_6(A,B) = A\bar{B} + \bar{A}B = A + B$ = OR EXCLUSIVO
$f_7(A,B) = \bar{A}B + A\bar{B} + AB = A+B$ = OR
$f_8(A,B) = \bar{A}\bar{B} = \overline{A + B}$ = NOR
$f_9(A,B) = AB + \bar{A}\bar{B} = A \cdot B$ = NOR EXCLUSIVO
$f_{10}(A,B) = \bar{A}\bar{B} + A\bar{B} = \bar{B}$ = NOT B
$f_{11}(A,B) = \bar{A}\bar{B} + A\bar{B} + AB = A + \bar{B}$ = IMPLICACION
$f_{12}(A,B) = \bar{A}\bar{B} + \bar{A}B = \bar{A}$ = NOT A
$f_{13}(A,B) = \bar{A}\bar{B} + \bar{A}B + AB = \bar{A} + B$ = IMPLICACION
$f_{14}(A,B) = AB + \bar{A}B + A\bar{B} = A + B = \overline{A \cdot B}$ = NAND
$f_{15}(A,B) = AB + \bar{A}B + A\bar{B} + \bar{A}\bar{B} = 1$ = UNO

Anteriormente vimos comos se podían evaluar funciones Booleanas usando el álgebra de Boole. Una variación de esa técnica es efectuar el álgebra directamente en la tabla de verdad.

Ejemplo: $f(A,B,C) = AB + \bar{A}C + A\bar{C}$

La tabla de verdad será:

A	B	C	$\bar{A}$	$\bar{C}$	AB	$\bar{A}C$	$A\bar{C}$	AB + $\bar{A}C$ + $A\bar{C}$
0	0	0	1	1	0	0	0	0
0	0	1	1	0	0	1	0	1
0	1	0	1	1	0	0	0	0
0	1	1	1	0	0	1	0	1
1	0	0	0	1	0	0	1	1
1	0	1	0	0	0	0	0	0
1	1	0	0	1	1	0	1	1
1	1	1	0	0	1	0	0	1

Ejemplo:

Un campseino tiene un ayudante no muy inteligente' El ayudante está encargado de cuidar que una oveja no se meta a un granero cuando la puerta está abierta, ya que en el se guarda maíz. Además, hay un coyote en los alrededores, con intenciones de comerse a la oveja.

El campesino decide diseñar una caja con tres interruptores:

- uno para "puerta abierta"
- uno para "oveja cercana al granero"
- uno para "coyote a la vista"

El ayudante deberá cerrar los interruptores de acuerdo a lo que esté sucediendo en un momento determinado. Si la combinación de sucesos resulta "peligrosa", los interruptores harán sonar una alarma, con la cual el campesino tomará las medidas adecuadas.

Se supone que:

1. La situación no es peligrosa cuando el coyote y la oveja no están a la vista.

2. Al coyote no le gusta el maíz

Asignemos letras a las variables:

P = Puerta abierta
C = Oveja a la vista
S = Coyote a la vista

P	C	S	Situaciones peligrosas
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

$f = \bar{P}CS + PC\bar{S} + PCS$
$f = \bar{P}CS + PC(\bar{S} + S)$ Post. 9
$f = \bar{P}CS + PC$ Post. 10
$f = C(P + \bar{P}S)$ Post. 9
$f = C(P + S)$ Teor. 10
$f = CP + CS$ Post. 9

2.6.4.- Simplificación de Funciones Booleanas:

Para cualquier función Booleana, podemos intentar la reducción o simplificación de la función para satisfacer algún criterio. Por lo general dicho criterio es el disminuir el número de componentes. Esto se traduce en la reducción del número de términos en la expresión representando una función booleana. Un término se define como cada ocurrencia de una variable a su complemento en una expresión booleana.

Ejemplo: $f(X, Y, Z) = \bar{X}Y(Z + \bar{Y}X) + \bar{Y}Z$ siete términos
$g(A, B, C) = A\bar{B} + \bar{A}B + AC$ seis términos

La simplificación de funciones se efectúa aplicando los postulados y teoremas del álgebra de Boole.

Ejemplo: $f(X, Y, Z) = \bar{X}Y(Z + \bar{Y}X) + YZ$
$= \bar{X}YZ + \bar{X}Y\bar{Y}X + \bar{Y}Z$
$= \bar{X}YZ + \bar{Y}Z$
$= Z(\bar{Y} + Y\bar{X})$
$= \overline{XY}Z$

$f(A, B, C, D) = ABC + ABD + \bar{A}B\bar{C} + CD + B\bar{D}$
$= ABC + \bar{A}D\bar{C} + CD + B(\bar{D} + AD)$
$= ABC + \bar{A}B\bar{C} + CD + B(\bar{D} + A)$
$= ABC + \bar{A}B\bar{C} + CD + AB + B\bar{D}$
$= (\bar{A}\bar{C} + A)B + ABC + CD + B\bar{D}$
$= (\bar{C} + A)B + ABC + CD + B\bar{D}$
$= B\bar{C} + AB + ABC + CD + B\bar{D}$
$= AB(1 + AC) + D\bar{C} + CD + B\bar{D}$
$= AB + CD + B(\bar{C} + \bar{D})$
$= AB + CD + B\,\overline{CD}$
$= AB + CD + B$
$= B(1 + A) + CD$
$= B + CD$

Comprobación mediante tabla de verdad:

A B C D	A C D	ABC	ABD	ABC	CD	BD	f(A.B,C,D)	
0 0 0 0	1 1 1	0	0	0	0	0	0	
0 0 0 1	1 1 0	0	0	0	0	0	0	
0 0 1 0	1 0 1	0	0	0	0	0	0	
0 0 1 1	1 0 0	0	0	0	1	0	1	- CD
0 1 0 0	1 1 1	0	0	1	0	1	1	
0 1 0 1	1 1 0	0	0	1	0	0	1	
0 1 1 0	1 0 1	0	0	0	0	1	1	- B
0 1 1 1	1 0 0	0	0	0	1	0	1	
1 0 0 0	0 1 1	0	0	0	0	0	0	
1 0 0 1	0 1 0	0	0	0	0	0	0	
1 0 1 0	0 0 1	0	0	0	0	0	0	
1 0 1 1	0 0 0	0	0	0	1	0	1	- CD
1 1 0 0	0 1 1	0	0	0	0	1	1	
1 1 0 1	0 1 0	0	1	0	0	0	1	
1 1 1 0	0 0 1	1	0	0	0	1	1	- B
1 1 1 1	0 0 0	1	1	0	1	0	1	

2.8.- Formas Algebráicas de Funciones Combinacionales:

Cualquier función booleana se puede expresar en una de dos formas estándar o canónicas. Estas formas, que serán de utilidad en procesos de simplificación a discutirse posteriormente, conducen a expresiones que pueden ser implementadas mediante circuitos de dos niveles. No existe una teoría general que conduzca a una expresión minimizada para circuitos de n niveles cuando $n > 2$.

2.8.1.- Formas Suma de productos y Productos de sumas:

Las funciones en forma de Suma de Productos SP se construyen mediante la disjunción de términos productos y los términos productos se obtienen mediante la conjunción de variables complementadas y no complementadas.

Ejemplos:

a) Términos productos: $A\bar{B}C$, $\bar{B}\bar{D}$

b) Función en forma SP

$$f = A\bar{B}C + \bar{B}\bar{D} + \bar{A}C\bar{D}$$

Las funciones Productos de Sumas PS se construyen mediante la conjunción de términos sumas y los términos sumas se obtienen mediante la disyunción de variables complementadas y sin complementar.

Ejemplos:

a) Términos suma: $\bar{A} + B + \bar{C}$
$\bar{B} + C + \bar{D}$
$A + \bar{C} + D$

b) Función en forma PS:

$$f = (\bar{A} + B + C)(\bar{B} + C + \bar{D})(A + \bar{C} + D)$$

2.8.2.- Formas Canónicas:

Las formas canónicas de funciones booleanas son formas SP y PS, con características especiales.

Mintérminos: Si un término producto de una función de n variables contiene las n variables en forma complementada o sin complementar, dicho término producto se llama mintérmino. Si la función está compuesta solamente de mintérminos, se dice que la función está en forma canónica de SP.

Ejemplo: $f(A,B,C) = \bar{A}B\bar{C} + AB\bar{C} + \bar{A}BC + ABC$

es una función de tres variables en forma canónica con cuatro mintérminos.

Para simplificar la notación de mintérminos, las variables se codifican de la siguiente forma:

variables no complementadas: 1
variables complementadas: 0

Usando este código en la función del ejemplo anterior, los mintérminos se pueden escribir de las siguientes formas equivalentes:

	Código	Número de lista
$\bar{A}\ B\ \bar{C}$	010	m_2
$A\ B\ \bar{C}$	110	m_6
$\bar{A}\ B\ C$	011	m_3
$A\ B\ C$	111	m_7

Los mintérminos se escriben en la forma m_i, donde i es el entero decimal del código binario correspondiente. Así, la función del ejemplo anterior se puede escribir en forma compacta como:

$$f(A,B,C) = m_2 + m_6 + m_3 + m_7$$
$$= m_2 + m_3 + m_6 + m_7$$

Un paso más en la simplificación, es escribir la función en la forma de lista de mintérminos

$$f(A,B,C) = \sum m\ (2,\ 3,\ 6,\ 7)$$

El orden en que se escriban las variables en la notación funcional es muy importante, ya que dicho orden se emplea para decodificar los mintérminos:

Ejemplo: $f(BCA) = \sum\ m\ (2,3,6,7)$

$$= m_2 + m_3 + m_6 + m_7$$
$$010 \quad 011 \quad 110 \quad 111$$
$$= BC\bar{A} + \bar{B}CA + BC\bar{A} + BCA$$
$$= \bar{A}\bar{B}C + A\bar{B}C + \bar{A}BC + ABC$$

Obsérvese que esta ecuación no es igual a la del ejemplo anterior, aún cuando las listas de mintérminos son iguales,

Consideremos la función:

$$f(A,B,C) = \underset{001}{\bar{A}\bar{B}C} + \underset{011}{\bar{A}BC} + \underset{101}{A\bar{B}C} + \underset{111}{ABC}$$
$$= m_1 + m_3 + m_5 + m_7$$
$$= \sum m(1,3,5,7)$$

construyamos su tabla de verdad:

		m_1	m_3	m_5	m_7	
Fila No.	A B C	$\bar{A}\bar{B}C$	$\bar{A}BC$	$A\bar{B}C$	ABC	f(A,B,C)
0	0 0 0	0	0	0	0	0
1	0 0 1	1	0	0	0	1
2	0 1 0	0	0	0	0	0
3	0 1 1	0	1	0	0	1
4	1 0 0	0	0	0	0	0
5	1 0 1	0	0	1	0	1
6	1 1 0	0	0	0	0	0
7	1 1 1	0	0	0	1	1

Obsérvese en la tabla que cada fila está numerada de acuerdo

al código decimal, y que los únicos 1 que aparecen en la tabla son aquellos en la fila i, producidos por el mintérmino m.

Por lo tanto, podemos eliminar todos los pasos intermedios y escribir la tabla de verdad directamente de la lista de mintérminos.

Ejemplo:

Fila No.	A B C	f(ABC)
0	0 0 0	0
1	0 0 1	0
2	0 1 0	1
3	0 1 1	1
4	1 0 0	0
5	1 0 1	0
6	1 1 0	1
7	1 1 1	1

$f(ABC) = \sum m(2,3,6,7)$

Otro aspecto importante es observar la tabla de verdad para $\bar{f}(A,B,C)$

Fila No.	A B C	f(A,B,C)	$\bar{f}(A,B,C)$
0	0 0 0	0	1
1	0 0 1	0	1
2	0 1 0	1	0
3	0 1 1	1	0
4	1 0 0	0	1
5	1 0 1	0	1
6	1 1 0	1	0
7	1 1 1	1	0

$\bar{f}(A,B,C) = \sum m(0,1,4,5)$

La tabla indica que $\bar{f}(A,B,C)$ tiene 1 en las filas 0,1,4, y 5, y por lo tanto:

$$\bar{f}(A,B,C) = \sum m(0,1,4,5)$$

y

$$f(A,B,C) = \sum m(2,3,6,7)$$

Obsérvese que todos los mintérminos compuestos de tres variables (8 en total) están contenidos en una de las dos listas de

mintérminos: $f(A,B,C)$ o $\bar{f}(A,B,C)$.

En general, los 2^n mintérminos de n variables aparecerán siempre en la forma canónica de SP para $f(X_1, X_2, X_3, \ldots X_n)$ o $\bar{f}(X_1, X_2, \ldots X_n)$.

Por ejemplo, si:

$$f(A,B,C,D) = \sum m(0,1,6,7)$$

el complemento tendrá $2^4 - 4 = 12$

$$\bar{f}(A,B,C,D) = \sum m(2,3,4,5,8,9,10,11,12,13,14,15)$$

$$= \sum m(2\text{-}5,\ 8\text{-}15)$$

Finalmente, del álgebra de Boole

$$f(X_1\ X_2\ \ldots\ldots X_n) + \bar{f}(X_1\ X_2\ \ldots\ldots\ X_n) = 1$$

Pero como

$$f(X_1X_2\ldots X_n) + \bar{f}(X_1X_2\ldots X_n) = \sum_{i=0}^{2^n-1} m_i$$

luego:

$$\sum_{i=0}^{2^n-1} m_i = 1$$

En otras palabras la disyunción de todos los mintérminos de n variables es igual a 1.

Maxtérminos: Si un término suma de una función de n variables contine las n variables en forma complementada o sin complementar, dicho término-suma se llama maxtérmino.

Si una función está compuesta de productos de términos-suma, siendo cada uno de los cuales un maxtérmino, se dice que la función está en forma canónica de suma de producto.

Ejemplo: $f_1(A_1B_1C) = (A+B+C)(A+B+\bar{C})(\bar{A}+B+C)(A+B+\bar{C})$

f_1 es una función en forma canónica con tres variables y cuatro maxtérminos.

Tal como para el caso de los mintérminos, existe una notación especial para maxtérminos.

Variables sin complementar: 0
Variables complementadas: 1

Con este código, los maxtérminos de la función f_1 del problema anterior quedan:

	Código Maxtérmino	Lista
$A + B + C$	0 0 0	M_0
$A + B + \bar{C}$	0 0 1	M_1
$\bar{A} + B + C$	1 0 0	M_4
$\bar{A} + B + \bar{C}$	1 0 1	M_5

Los maxtérminos se abrevian como M_i, donde i es el decimal entero correspondiente al código binario del maxtérmino. Luego, la función f_1 del ejemplo quedaría:

$$f_1(A,B,C) = M_0 M_1 M_4 M_5$$

O bién, escribiéndola en forma de lista de maxtérminos:

$$f_1(A_1B_1C) = \overline{1\,1}\ M(0,1,4,5)$$

Las dos últimas expresiones para f_1 están en forma canónica de suma de productos.

Tal como en el caso de mintérminos, el orden de las variables en la notación funcional es muy importante.

La tabla de verdad para la función f_1 anterior es:

		M_0	M_1	M_4	M_5	
Fila No.	A+B+C	$A+B+\bar{C}$	$\bar{A}+B+C$	$\bar{A}+B+\bar{C}$	A+B+C	$f_1(A_1B_1C)$
0	0 0 0	0	1	1	1	0
1	0 0 1	1	0	1	1	0
2	0 1 0	1	1	1	1	1
3	0 1 1	1	1	1	1	1
4	1 0 0	1	1	0	1	0
5	1 0 1	1	1	1	0	0
6	1 1 0	1	1	1	1	1
7	1 1 1	1	1	1	1	1

Obsérvese que los únicos ceros que aparecen en la tabla, están en la fila i y son producidos por el maxtérmino M_i. Por lo tanto, tal como en el caso de los mintérminos, la tabla de verdad puede ser generada por inspección de la lista de maxtérminos.

Examinemos la función siguiente:

$$f(A,B,C) = \underbrace{(A+B+\bar{C})}_{001}\underbrace{(A+\bar{B}+\bar{C})}_{011}\underbrace{(\bar{A}+B+\bar{C})}_{101}\underbrace{(\bar{A}+\bar{B}+C)}_{111}$$

$$f(A,B,C) = M_1M_3M_5M_7$$

$$f(A,B,C) = \Pi M(1,3,5,7)$$

Los maxtérminos de la función ubican los ceros en las filas 1, 3, 5 y 7 de la tabla de verdad.

Fila	A B C	F(A,B,C)
0	0 0 0	1
1	0 0 1	0
2	0 1 0	1
3	0 1 1	0
4	1 0 0	1
5	1 0 1	0
6	1 1 0	1
7	1 1 1	0

Observando la tabla de verdad, vemos que $f(A,B,C) = \sum m(0,2,4,6)$.

Luego

$$\bar{f}(A,B,C) = \sum m(1,3,5,7)$$

$$= \underset{001}{m_1} + \underset{011}{m_3} + \underset{101}{m_5} + \underset{111}{m_7}$$

$$= \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C + ABC$$

Consecuentemente,

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

$$= \overline{\bar{A}\bar{B}C} \cdot \overline{\bar{A}BC} \cdot \overline{A\bar{B}C} \cdot \overline{ABC}$$

$$= \underset{001}{(A+B+\bar{C})}\underset{011}{(A+\bar{B}+\bar{C})}\underset{101}{(\bar{A}+B+\bar{C})}\underset{111}{(\bar{A}+\bar{B}+\bar{C})}$$

$$= M_1M_3M_5M_7$$

$$= \Pi M(1,3,5,7)$$

Es decir, hemos demostrado que:

$$f(A,B,C) = \Pi M(1,3,5,7) = \sum m(0,2,4,6)$$

lo cual es evidente al inspeccionar la tabla de verdad.

En las manipulaciones algebráicas del ejemplo anterior, resultaron aparentes ciertas relaciones entre mintérminos y maxtérminos

$$\overline{m_1} = \overline{\bar{A}\bar{B}C} = \underbrace{A + B + \bar{C}}_{0\ 0\ 1} = M_1$$
$$001$$

$$\overline{m_3} = \overline{\bar{A}BC} = \underbrace{A + \bar{B} + \bar{C}}_{0\ 1\ 1} = M_3$$

En general: $\overline{m}_i = M_i$

$$\overline{M}_i = m_i$$

y por lo tanto los mintérminos y maxtérminos: son complementos el uno del otro.

Observemos la tabla de verdad del complemento de la función del ejemplo anterior:

$$f(A,B,C) = \Pi M(1,3,5,7)$$

Fila	A B C	f(A,B,C)	$\bar{f}$(A,B,C)
0	0 0 0	1	0
1	0 0 1	0	1
2	0 1 0	1	0
3	0 1 1	0	1
4	1 0 0	1	0
5	1 0 1	0	1
6	1 1 0	1	0
7	1 1 1	0	1

de la tabla vemos que los ceros de $\bar{f}(A,B,C)$ están en las filas 0,2, 4 y 6. Luego:

$$\bar{f}(A,B,C) = \Pi M(0,2,4,6)$$

y

$$(A,B,C) = \Pi M(1,3,5,7)$$

es decir todos los maxtérminos generados por tres variables, aparecen en $f(A,B,C)$ o en $\bar{f}(A, B, C)$. Además, del álgebra de Boole, tenemos que:

$$f(A,B,C) \cdot \bar{f}(A,B,C) = 0$$

luego:

$$(M_0 M_2 M_4 M_6)(M_1 M_3 M_5 M_7) = 0$$

o bién

$$\prod_{i=0}^{2^3-1} M_i = 0$$

En general, para n variables tenemos:

$$\prod_{i=0}^{2^n-1} M_i = 0$$

Resumen:

Resumiremos los resultados obtenidos mediante la función - - $f(A,B,C) = \bar{A}B\bar{C} + \bar{A}BC + AB\bar{C} + ABC$

$$f(A,B,C) = \sum m\,(2,3,6,7) = \Pi M(0,1,4,5)$$

$$\bar{f}(A,B,C) = \sum m\,(0,1,4,5) = \Pi M(2,3,6,7)$$

El significado de lo anterior, se ilustra mediante la función siguiente:

$$f(W,X,Y,Z) = \Pi M(0,4 - 8, 10, 12, 15)$$

La función es de cuatro variables, W, X, Y y Z y está especificada en forma canónica de productos de sumas o en forma de lista de maxtérminos.

Con esta información, podemos escribir la función en forma de lista de mintérminos

$$f(W,X,Y,Z) = \Pi M(0,4 - 8, 10, 12, 15)$$

$$= \sum m(1,2,3,9,11,13,14)$$

$$\bar{f}(W,X,Y,Z) = \Pi M(1,2,3,9,11,13,14)$$
$$= \sum m(0,4-8,10,12,15)$$

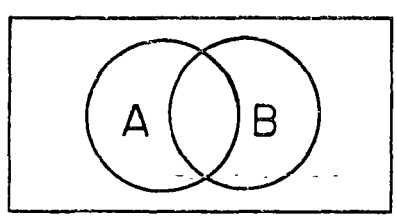
2.9.- Mapas de Karnough:

Anteriormente vimos que es posible simplificar funciones booleanas ya sea empleando los teoremas del álgebra de Boole, o a partir de la tabla de verdad de la función. El empleo de la tabla de verdad, es un procedimiento gráfico que puede resultar poco eficiente. Sin embargo, se la tabla de verdad es modificada un tanto, puede obtenerse una representación gráfica de la función que resulta muy útil para efectos de minimización. Esta forma es el mapa de Karnough.

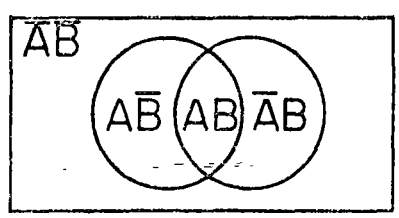
El mapa de Karnough no es más que una extensión de los conceptos de: tablas de verdad, diagramas de Venn y mintérminos.

Para hacer explícita esta extensión, transformemos un diagrama de Venn en un mapa de Karnough.

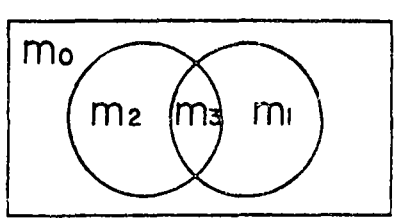
Consideremos el diagrama de Venn de la figura (a) siguiente. Las dos variables A y B se representan mediante subdivisiones del conjunto universal. En la figura (b) se ilustra el hecho que cada subdivisión disjunta del diagrama de Venn está formada por las intersecciones AB, $\bar{A}B$, $A\bar{B}$, $\bar{A}\bar{B}$, las que son los mintérminos de dos variables y por lo tanto podemos redesignar las



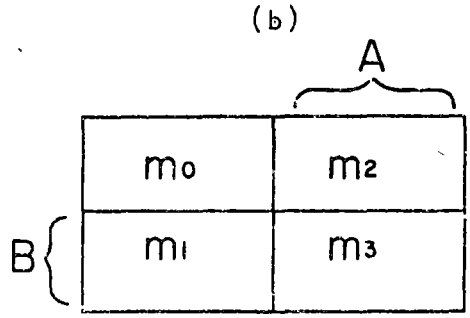
(a)



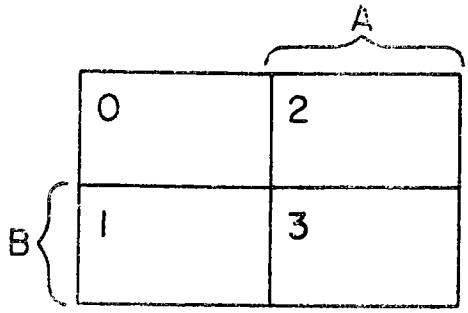
(b)



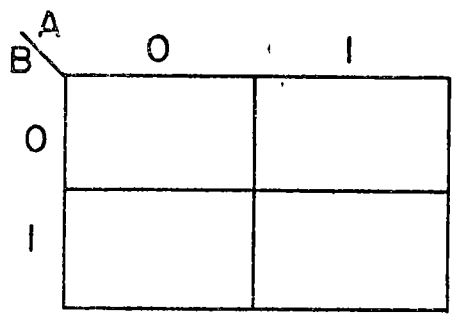
(c)



(d)



(e)



(f)

las subdivisiones del diagrama de Venn como los mintérminos m_0, m_1, m_2 y m_3, lo cual se ilustra en la figura (c). En la figura (d) se ha redibujado el diagrama de Venn de la figura (c), con las áreas de cada subdivisión iguales. Obsérvese que las áreas adyacentes del diagrama de Venn son también áreas adyacentes en la figura (d). Sin embargo, en la figura (d), una mitad del diagrama representa a la variable A y otra mitad representa a la variable B. Como la notación de mintérminos se identifica con cada cuadrado del diagrama, podemos omitir la letra m y dejar solamente el subíndice, como se muestra en la figura (e). Esta figura muestra una forma del mapa de Karnaugh En la figura (f) se muestra una segunda forma del mapa, que es la más usada.

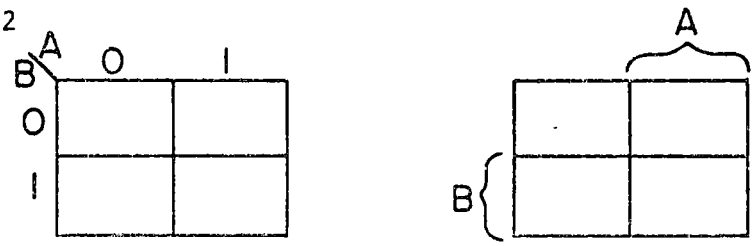
Es importante observar que el mapa de Karnaugh es una representación gráfica de una tabla de verdad y existe por lo tanto una correspondencia uno a uno entre ambas. La tabla de verdad tiene una fila para cada mintérmino, mientras el mapa de Karnaugh tiene un cuadrado para cada mintérmino.

Los requisitos que debe satisfacer un mapa de Karnaugh son los siguientes : 1) debe haber un cuadrado para cada combinación de variables; es decir deber haber 2^n cuadrados para n variables 2) Los cuadrados deben estar organizados de tal forma que cualquier par de cuadrados inmediatamente adyacentes el uno del otro (horizontal y verticalmente) deben corresponder a condiciones de combinaciones de variables lógicamente adyacentes: es decir, que difieran en una sola variable.

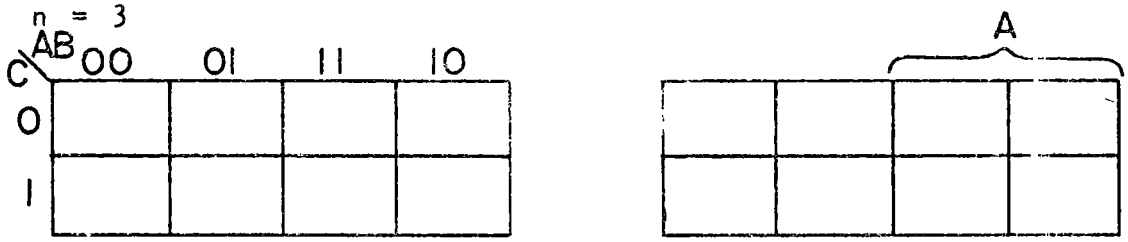
Observemos el paso de la figura (c) a la (d); consideremos el mintérmino m_0 en la figura (c); dicho mintérmino es adyacente a m_1, m_2 y m_4, lo cual satisface el requisito 2) mencionado anteriormente. Sin embargo, en la figura (d), m_0 no está físicamente adyacente a m_4. Para conciliar esta inconsistencia con el requisito 2) se considera que los extremos izquierdo y derecho del mapa son adyacentes. Es decir, el mapa-K viene a ser un cilindro.

En la serie de figuras siguientes, se encuentran los mapas de Karnough para 2, 3, 4 y 5 variables.

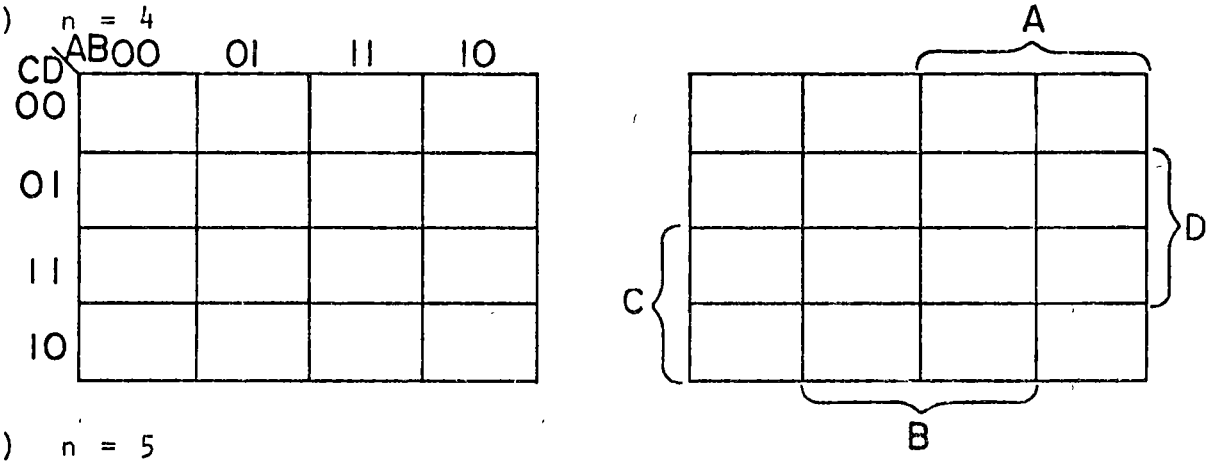
1) n = 2



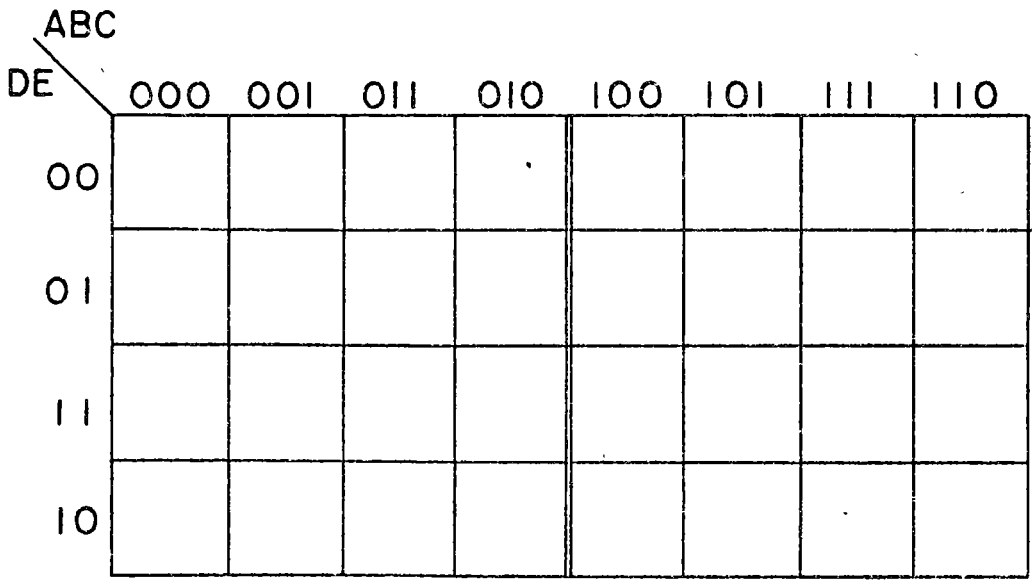
2) n = 3

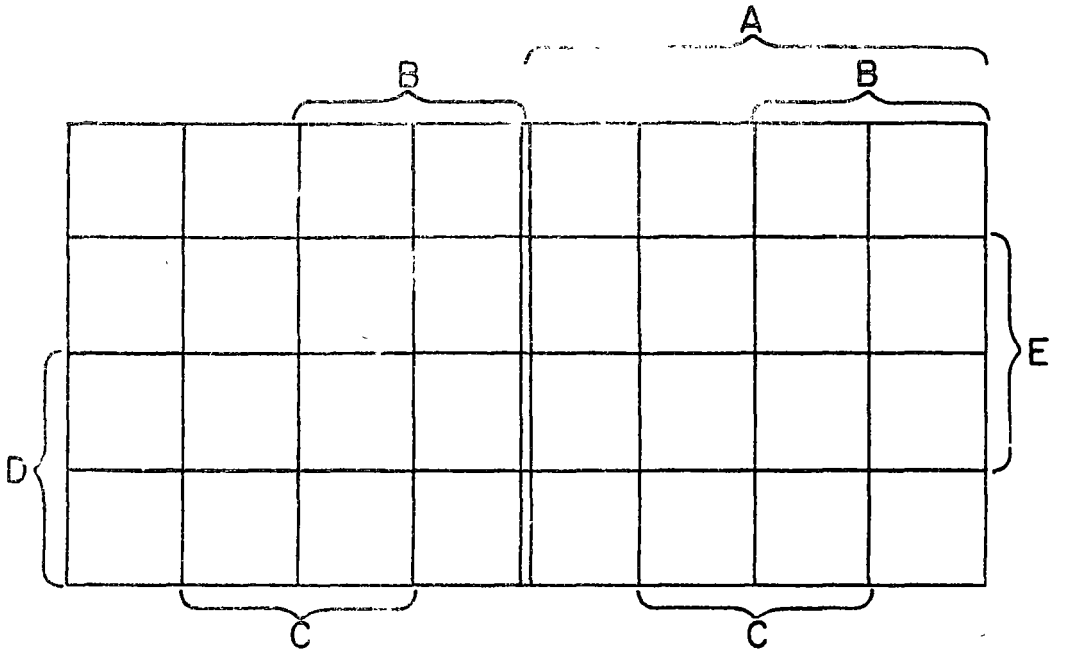


3) n = 4



4) n = 5





2.9.1.- Representación de Funciones en Mapas - K:

Una función booleana puede ser representada fácilmente en un mapa-K si se encuentra en una de sus formas canónicas.

1) Si la función está representada en forma de lista de mintérminos, se ponen 1 en los cuadrados correspondientes a cada mintérmino de la lista.

2) Si la función está representada en forma de lista de maxtérminos, se ponen 0 en los cuadrados correspondientes a cada maxtérmino de la lista o 1 en los cuadrados correspondientes a maxtérminos que no estén en la lista.

Ejemplos:

1) Representar en un mapa-K la función

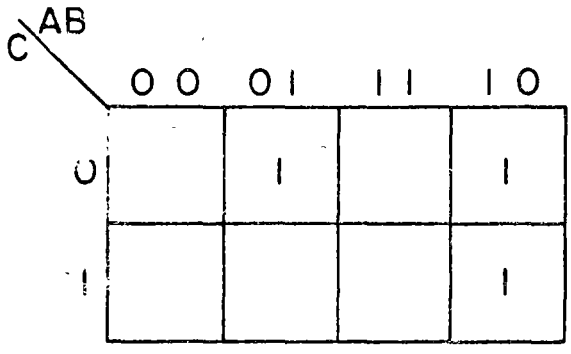
$$f(A,B,C,D) = \sum m(0,3,5,7,10 - 15)$$

CD \ AB	00	01	11	10
00	1		1	
01		1	1	
11	1	1	1	1
10			1	1

2) Representar en un mapa - K la función

$$f(A,B,C) = \Pi M(0,1,3,6,7)$$

C \ AB	00	01	11	10
0	0		0	
1	0	0	0	

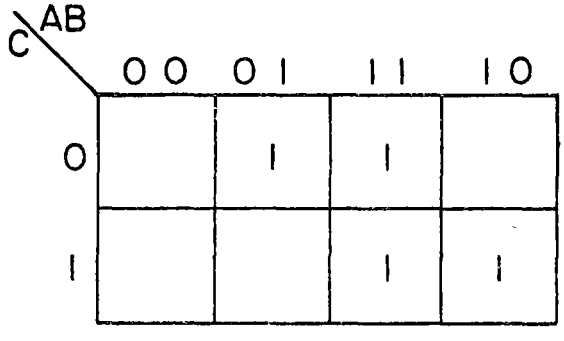


Para el caso de funciones que no estén en forma de lista de mintérminos o maxtérminos, se presentan dos casos:

1) La función está en forma de suma de productos. En este caso, lo más conveniente es representar directamente la función en un mapa de Karnough; sin expandirla.

Ejemplo:

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



2) La función está en forma de productos de suma. En este caso, lo más conveniente es pasar a suma de productos y luego representarla en el mapa-K.

2.9.2.- Simplificación de Funciones:

La simplificación de una función debe estar orientada hacia la minimización de algún objetivo de diseño. Estos objetivos de diseño pueden ser varios, pero en general hay dos que tienen gran importancia: costo y velocidad.

Estos objetivos de diseño obligan a establecer un compromiso entre ambos ya que invariablemente existen las relaciones:

Alta velocidad => Alto costo
Bajo costo => Baja velocidad

Consideremos el caso de la función:

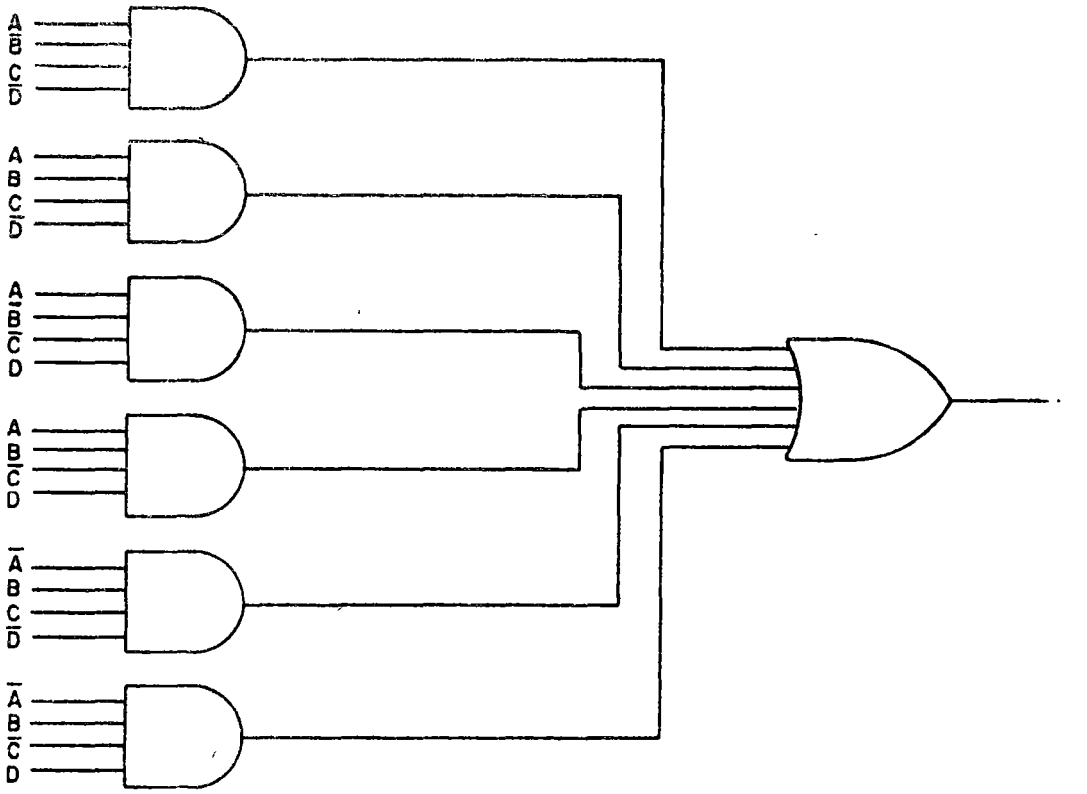
$$f = (A,B,C,D) = \sum m(5,6,9,10,13,14) \qquad (A)$$

$$= AC\bar{D} + A\bar{C}D + BC\bar{D} + B\bar{C}D \qquad (B)$$

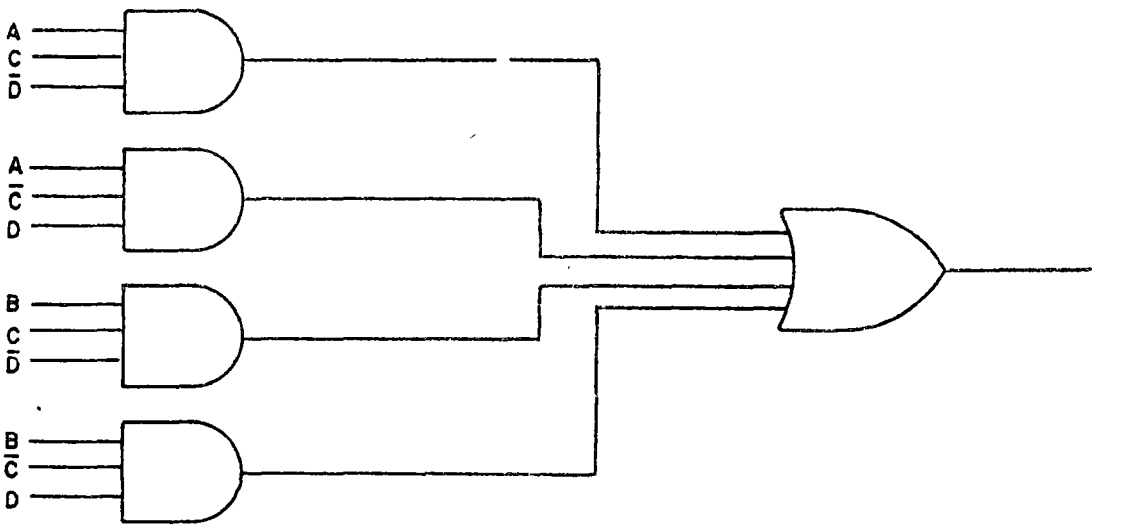
$$= (A + B)(C\bar{D} + \bar{C}D) \qquad (C)$$

La realización de cada una de las formas de f se muestra en la siguiente figura.

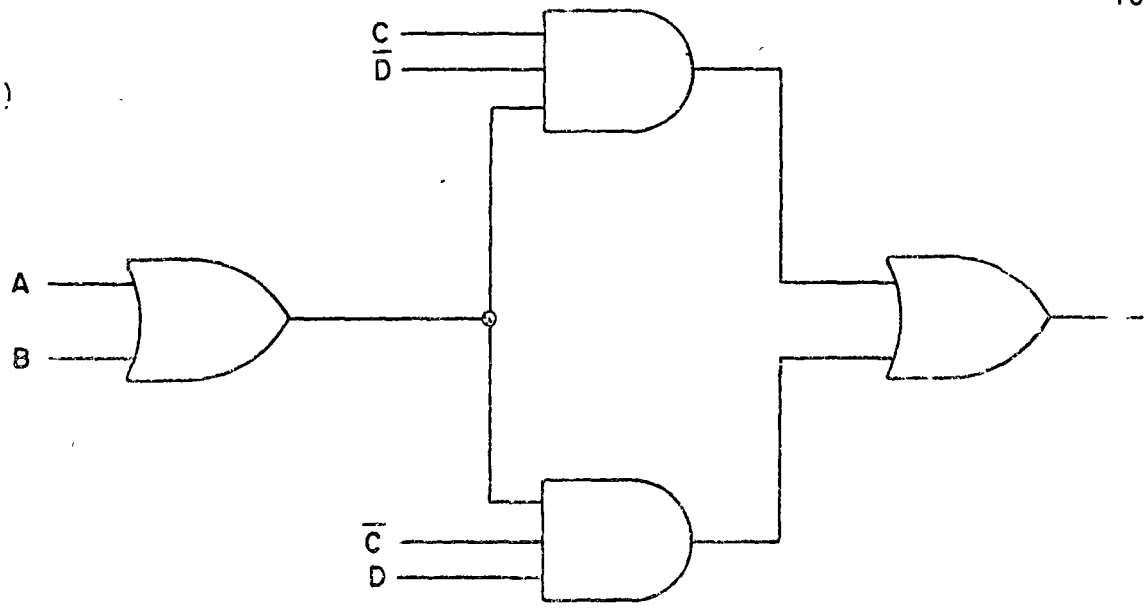
A)



B)



C)



Obviamente el circuito A es el más complejo: requiere de 7 compuertas. Comparando los circuitos B y C vemos que el más simple es el C, ya que requiere de 4 compuertas contra 5 del B. Sin embargo, una señal de entrada al circuito C experimentará 3 retardos de tiempo antes de salir, ya que hay 3 niveles de compuertas; en cambio en el B hay sólo dos niveles. Aquí represente el compromiso entre costo y velocidad.

Nuestro criterio de minimización será el de optimizar velocidad. Es decir, trataremos de obtener circuitos de dos niveles de compuerta:

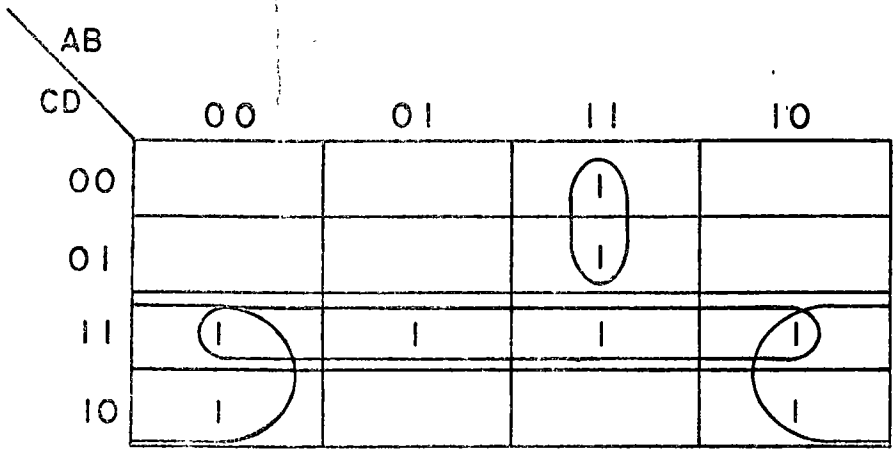
La simplificación mediante mapas-K se basa en el hecho que aquellos conjuntos de mintérminos que se puedan combinar en términos-producto más simples, deben ser adyacentes o aparecer en patrones simétricos en el mapa-K.

Los mapas K están ordenados de tal forma que los mintérminos en cuadrados adyacentes son idénticos excepto por una variable. Dicha variable aparece en forma sin complementar en un mintérmino y complementada en el otro y por lo tanto, el valor de la función será independiente del valor de dicha variable.

Cualquier par de mintérminos de n variables, adyacentes en un mapa-K se puede combinar en un término-producto de (n-1) variables.

Ejemplo:

$$f(A,B,C,D) = \sum m(2,\ 3,\ 7,\ 10,\ 11,\ 12,\ 13,\ 15)$$



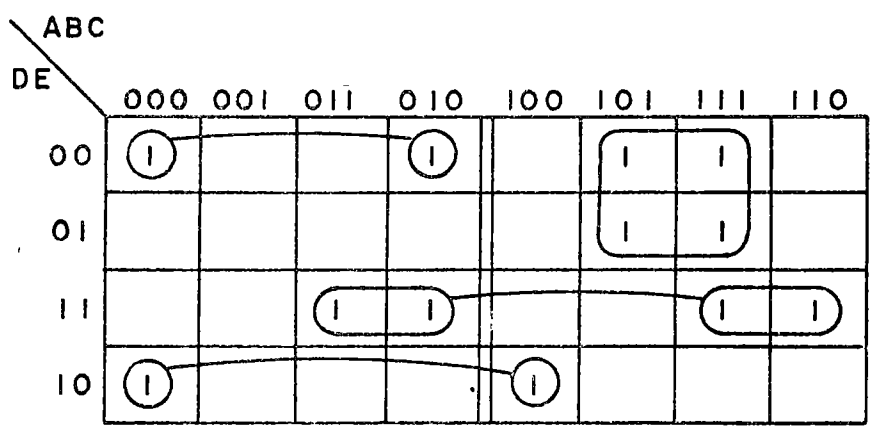
$f = AB\bar{C} + CD + \bar{B}C$

Las adyacencias deben ser de filas o de columnas, pero no de diagonales.

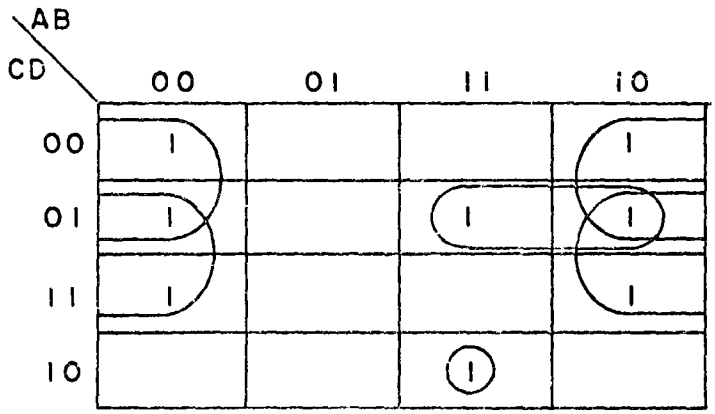
Como los conjuntos de dos mintérminos se combinan para eliminar una variable y los de cuatro mintérminos, para eliminar dos, los de ocho se combinarán para eliminar tres.

Ejemplos:

1) $f = \sum m(0,2,8,11,15,18,20,21,27,28,29,31)$

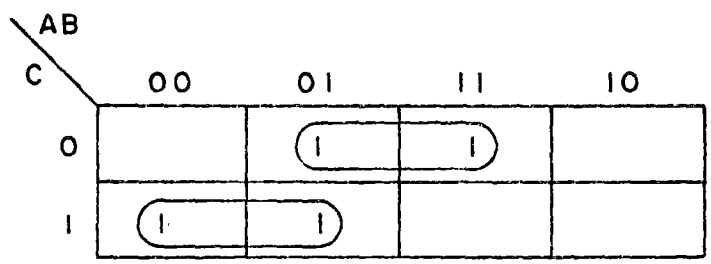


2) $f(A,B,C,D) = \sum m\ (0,1,3,8,9,11,13,14)$



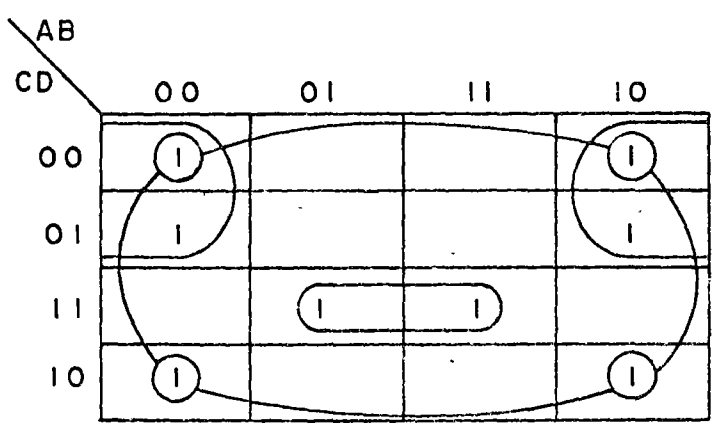
$f = ABC\bar{D} + A\bar{C}D + \bar{B}\bar{C} + \bar{B}D$

3) $f = \sum m(1,2,3,6)$



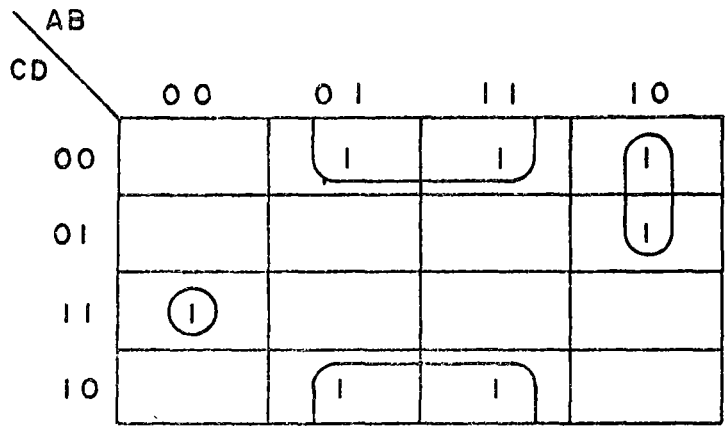
$f = B\bar{C} + \bar{A}C$

4) $f = \sum m\ (0,1,2,7,8,9,10,15)$



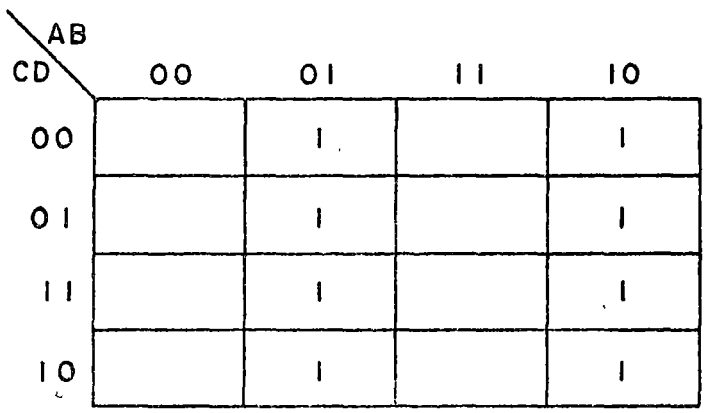
$f = \bar{B}\bar{C} + \bar{B}\bar{D} + BCD$

5) $f(A,B,C,D) = \sum m\ (3,4,6,8,9,12,14)$



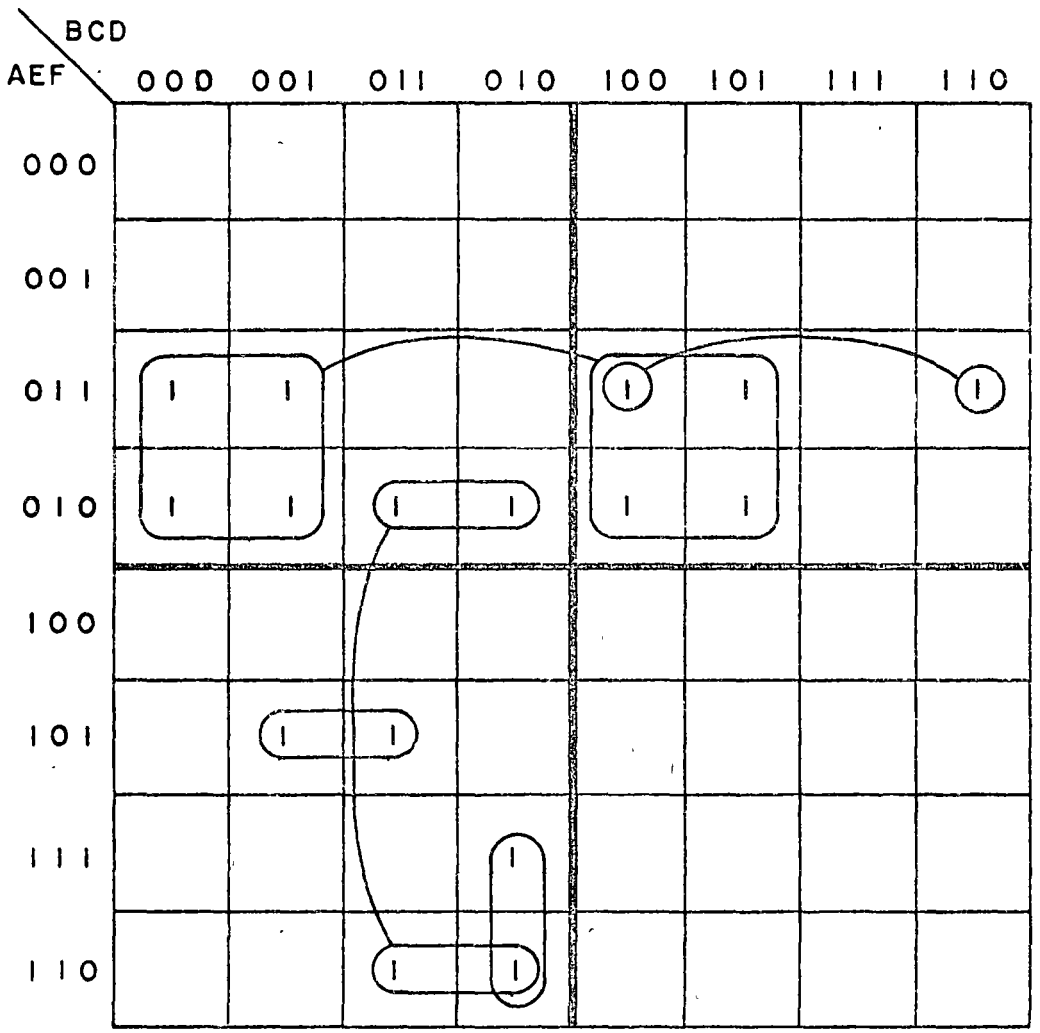
$f = \bar{A}\bar{B}\,CD + A\bar{B}\bar{C} + B\bar{D}$

6) Se requiere producir una salida siempre que ocurrra un número entre 4 y 11



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

7) $f(A,B,C,D,E,F) = \sum m(2,3,6,7,10,14,18,19,22,23,27,37,42,43,45,46)$



$$f = \bar{A}B\bar{D}FE + A\bar{B}D\bar{E}F + A\bar{B}C\bar{D}E + \bar{B}CE\bar{F}$$

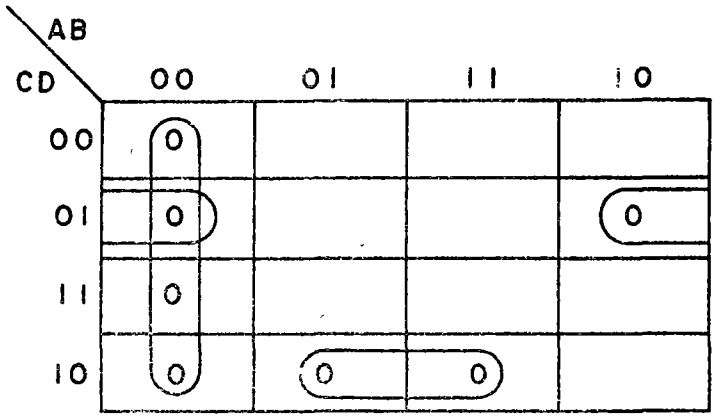
2.9.3.- Simplificación de Funciones Productos de Suma:

La simplificación de funciones en forma de productos de suma se puede realizar en forma muy simple de la siguiente forma:

1. Representar la función (PS o SP) en el mapa-K.
2. Complementar el mapa. Ahora se tiene $\bar{f}$.
3. Minimizar $\bar{f}$ en forma de SP.
4. Complementar $\bar{f}$ y obtenemos f en forma de PS.

Ejemplos:

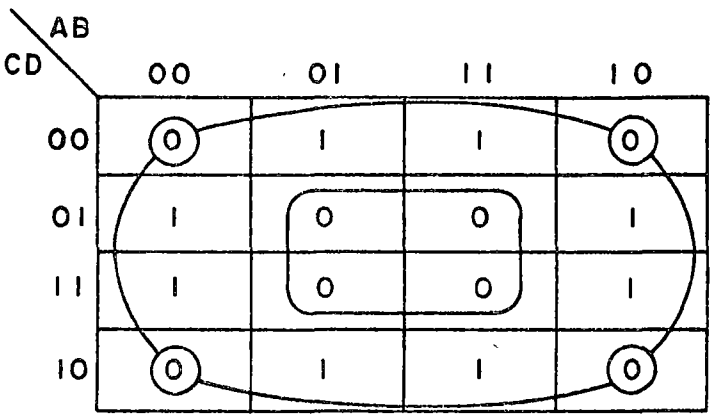
1) $f(A,B,C,D) = \Pi\, M(0,1,2,3,6,9,14)$



$$\bar{f} = BC\bar{D} + \bar{B}\bar{C}D + \bar{A}\bar{B}$$

$$f = (\bar{B} + \bar{C} + D)(B + C + D)(A + B)$$

2) $f(A,B,C,D) = \sum m(1,3,4,6,9,11,12,14)$



$$f = (B + D)\ (\bar{B} + \bar{D})$$

2.9.4.- Funciones Incompletamente Especificadas:

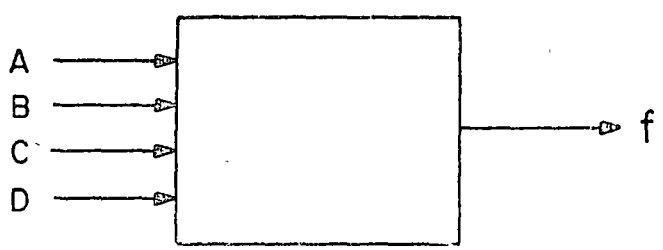
En el diseño de circuitos digitales es frecuente encontrar casos en que la función booleana no está completamente especificada. Es decir, puede darse el caso de una función de la que se requieran ciertos mintérminos, se omitan otros y los restantes sean opcionales. Esto puede ser el caso de un circuito que sea una parte de un sistema más grande en el que ciertas entradas ocurrirán bajo circunstancias tales que la salida del circuito no tendrá influencia sobre el sistema en global. También puede ser el caso de que cierta combinación de entradas no se producirá nunca debido a ciertas restricciones externas.

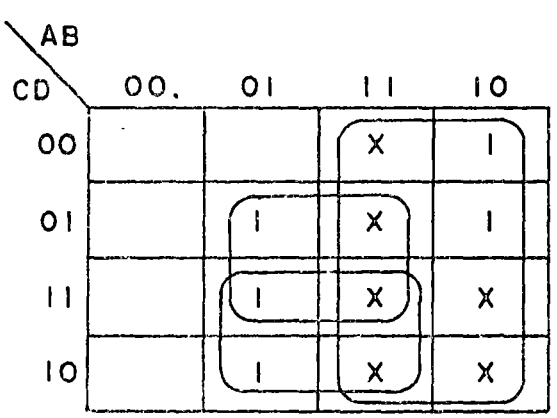
Cuando la salida no tenga efecto sobre el resto del sistema, obviamente no nos importará si ésta es un 0 o un 1. En tal condición, decimos que la salida no está especificada y lo indicamos mediante una X en la tabla de verdad.

Ejemplo:

La entrada de un circuito es la representación en BCD de los números decimales 0-9. El circuito se usa como detector de redondeo, es decir, producirá una salida cuando la entrada sea 5, 6, 7, 8 ó 9.

ABCD	f	m
0000	0	m_0
0001	0	m_1
0010	0	m_2
0011	0	m_3
0100	0	m_4
0101	1	m_5
0110	1	m_6
0111	1	m_7
1000	1	m_8
1001	1	m_9
1010	X	m_{10}
1011	X	m_{11}
1100	X	m_{12}
1101	X	m_{13}
1110	X	m_{14}
1111	X	m_{15}





$f = A + BC + BD$

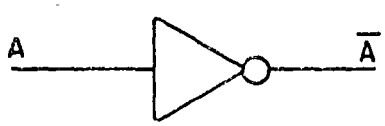
CAPITULO 3: CIRCUITOS COMBINACIONALES

3.1.- Compuertas Lógicas

En este capítulo estudiaremos algunos aspectos de análisis y diseño de circuitos combinacionales. En primer lugar, estudiaremos los dispositivos más comunes en el diseño lógico: las compuertas.

3.1.1.- Inversor:

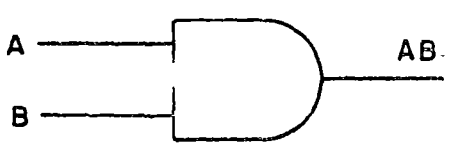
El inversor es un circuito de una entrada; a la salida produce el complemento del estado lógico de la entrada. Por ejemplo, si la entrada es 0, la salida es 1 y viceversa.



A	$\bar{A}$
0	1
1	0

3.1.2.- AND:

El circuito AND tiene dos o más entradas. en la figura siguiente se muestra su símbolo y su tabla de verdad.

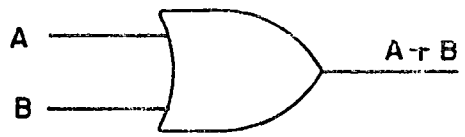


A B	A·B
0 0	0
0 1	0
1 0	0
1 1	1

De la tabla de verdad vemos que la salida del circuito será 1 sólo cuando ambas entradas, A y B, sean 1. Cualquier otra combinación a la entrada producirá un 0 a la salida.

3.1.3.- OR

El circuito OR tiene dos o más entradas. En la figura se muestra su símbolo y su tabla de verdad.



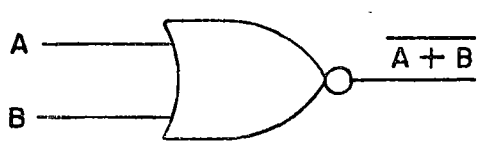
A B	A + B
0 0	0
0 1	1
1 0	1
1 1	1

De la tabla de verdad vemos que la salida del circuito será 1 cuando cualquiera (o todas) las entradas sean 1, y será 0 sólo cuando todas las entradas sean 0.

Los circuitos AND, OR e Inversor son los básicos de un sistema lógico y de hecho con estos tres es posible (aunque no es aconsejable) diseñar una computadora. Estos circuitos se combinan para producir funciones más elaboradas, algunas de las cuales veremos a continuación.

3.1.4.- NOR:

El circuito NOR es una combinación del circuito OR y del Inversor. En la figura se muestra su símbolo y su tabla de verdad.

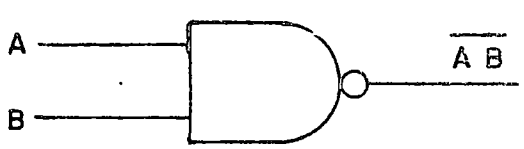


A B	$\overline{A + B}$
0 0	1
0 1	0
1 0	0
1 1	0

De la tabla de verdad vemos que la salida será 1 sólo cuando todas las entradas sean 0 y será 0 cuando cualquiera de las entradas sea 1.

3.1.5.- NAND:

El circuito NAND es una combinación del circuito AND y el Inversor. En la figura se muestra su símbolo y su tabla de verdad.

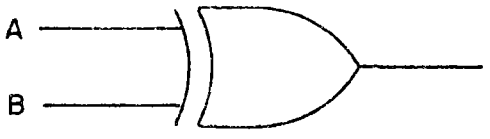


A B	$\overline{A \cdot B}$
0 0	1
0 1	1
1 0	1
1 1	0

La salida será o cuando todas las entradas sean 1 y será 1 cuando cualquiera de ellas sea 0.

3.1.6.- OR Exclusivo

El circuito OR Exclusivo realiza la función OR con la excepción de que cuando ambas entradas son 1, la salida es 0.



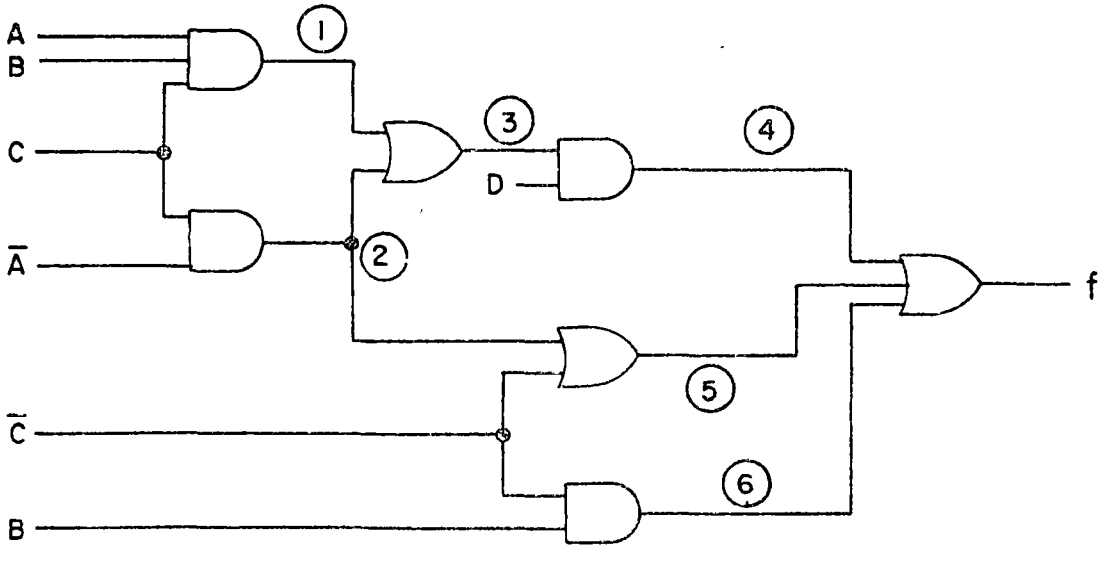
A B	A ⊕ B
0 0	0
0 1	1
1 0	1
1 1	0

3.2.- Análisis de Circuitos Combinacionales:

El análisis de circuitos combinacionales se refiere a circuitos cuya salida depende exclusivamente de sus entradas y están formados por interconexiones de compuertas.

El análisis de tales circuitos requiere obtener la ecuación booleana que lo representa y luego la caracterización completa de la función resultante, para todas las combinaciones posibles de entrada:

Ejemplos:



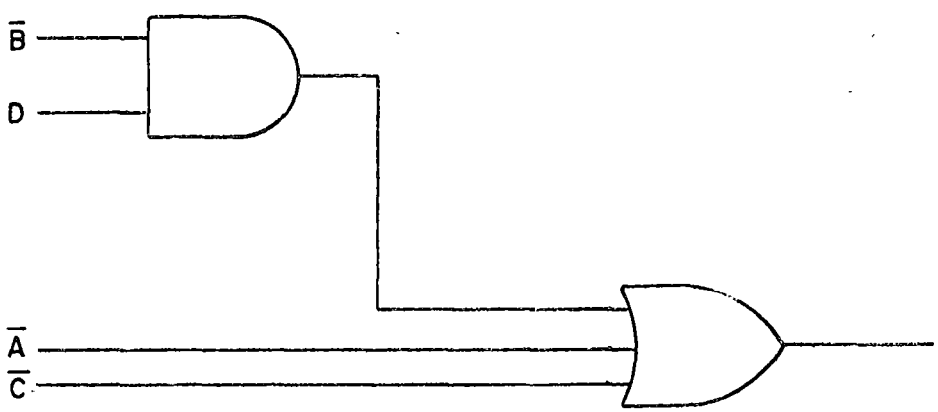
$f = 4 + 5 + 6$
$4 = D \cdot 3$ $\qquad 3 = 1 + 2 \qquad 1 = A \cdot \bar{B} \cdot C$
$5 = \bar{C} + 2$ $\qquad 2 = \bar{A} \cdot C$
$6 = B \cdot \bar{C}$

Luego:

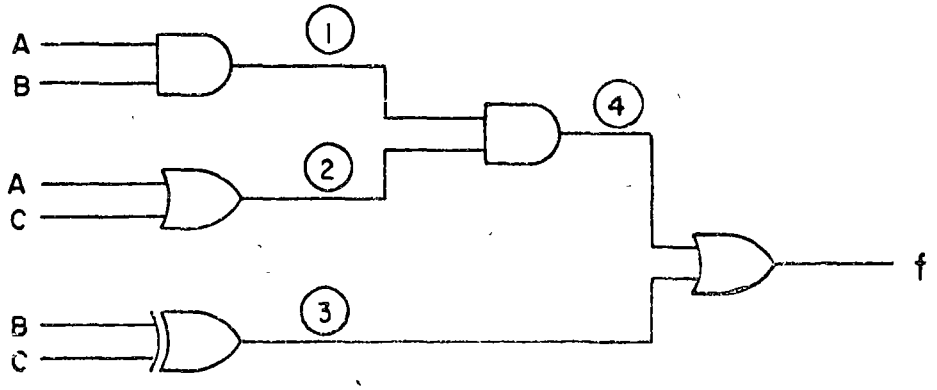
$f = D \cdot 3 + \bar{C} + 2 + B\bar{C}$
$f = D(1 + 2) + \bar{C} + \bar{A}C + B\bar{C}$
$f = D(A\bar{B}C + \bar{A}c) + \bar{C} + \bar{A}C + B\bar{C}$

Pero esta función podemos simplificarla como sigue:

$f = A\bar{B}CD + \bar{A}CD + \bar{A}C + B\bar{C} + \bar{C}$
$f = CD(\bar{A} + A\bar{B}) + \bar{A}C + \bar{C}(B + 1)$
$f = CD(\bar{A} + \bar{B}) + \bar{A}C + \bar{C}$
$f = \bar{A}CD + \bar{B}CD + \bar{A}C + \bar{C}$
$f = \bar{A}C(D + 1) + \bar{B}DC + \bar{C}$
$f = \bar{A}C + B\bar{D} + \bar{C}$
$f = \bar{A}C + \bar{C} + \bar{B}D$
$f = \bar{A} + \bar{C} + \bar{B}D$



2)



$f = \overline{③ + ④}$

$4 = ① \cdot ②$

$3 = B \oplus C$

$① = \overline{AB}$

$② = \overline{\bar{A} + C}$

$f = \overline{(B \oplus \bar{C}) + ① \cdot ②}$

$f = \overline{(B \oplus \bar{C} + \overline{AB}\ (\overline{\bar{A} + C})}$

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

Luego:

$f = \overline{(\bar{B}\bar{C} + BC) + \overline{AB}\ (\bar{A} + C}$

$\bar{f} = (\bar{B}\bar{C} + BC) + \overline{AB}\ (\overline{\bar{A} + C})$

$\bar{f} = \bar{B}\bar{C} + BC + (\bar{A} + \bar{B})\ A\bar{C}$

$\bar{f} = \bar{B}\bar{C} + BC + A\bar{B}\bar{C}$

$\bar{f} = \bar{B}\bar{C}(1 + A) + BC$

$\bar{f} = \bar{B}\bar{C} + BC$

$\bar{f} = B \odot C$ COINCIDENCIA

$f = B \oplus C$

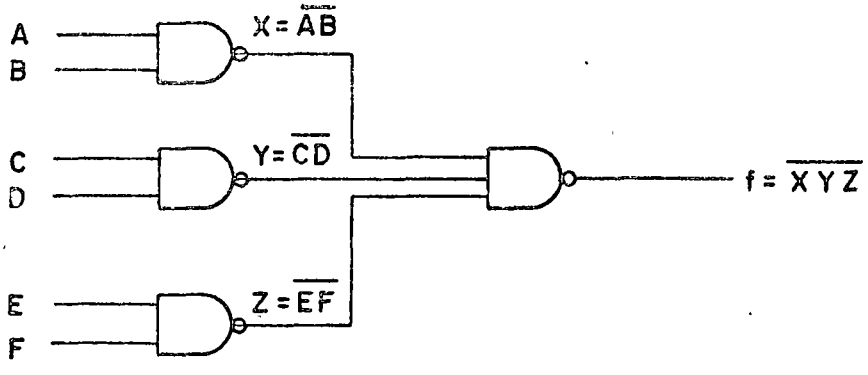
En los ejemplos anteriores vimos como se podía obtener un conjunto de ecuaciones para una estructura dada. En dichas ecuaciones se describían el tipo de compuerta y sus interconexciones. Por lo tanto, se pueden utilizar conjuntos de ecuaciones en vez de diagramas lógicos.

3.3.- Propiedades de las funciones NAND y NOR

Al revisar las dieciseis funciones definidas por las combinaciones de dos variables, observamos que todas ellas se pueden implementar mediante las funciones AND, OR y NOT. En general, cualquier función lógica puede ser implementada mediante las tres compuertas mencionadas.

Sin embargo, cabe preguntarse si no es posible implementar cualquier función con menos de tres tipos diferentes de compuertas.

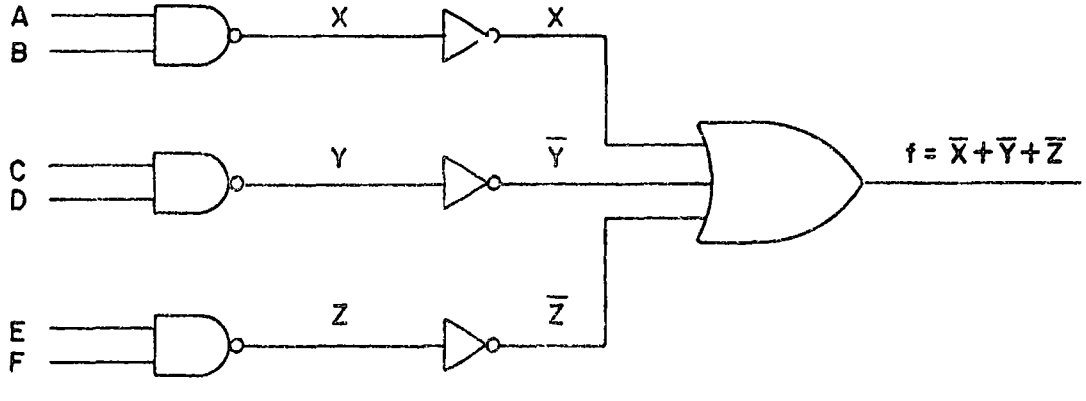
Consideremos el circuito de la figura, que consiste de tres compuertas NAND, cuyas salidas están conectadas a la entrada de otra compuerta NAND.



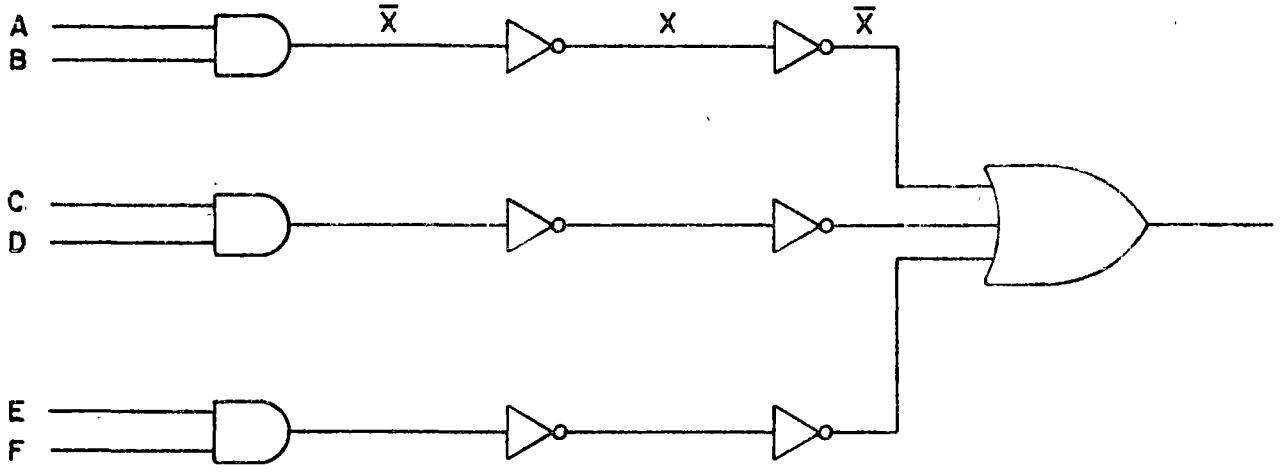
De las leyes de Morgan tenemos que:

$$f = \overline{XYZ} = \bar{X} + \bar{Y} + \bar{Z}$$

Dicha función también puede ser implementada mediante el siguiente circuito:



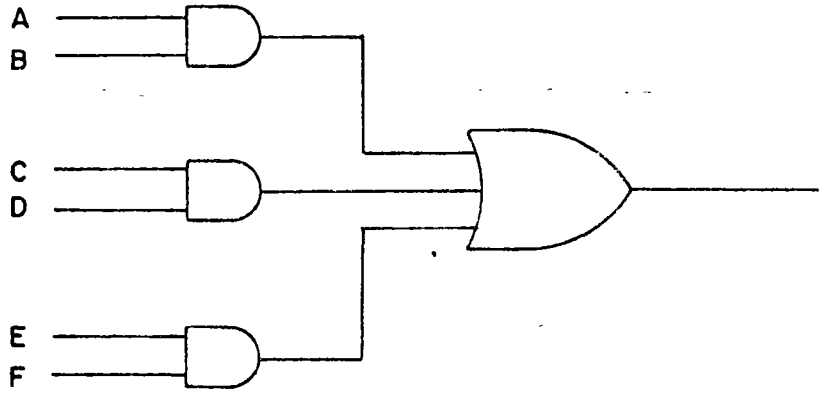
o bién, podemos además hacer lo siguiente



Pero tener dos inversores en cascada, equivale algebráicamente a:

$$\overline{\overline{X}} = X$$

Luego:

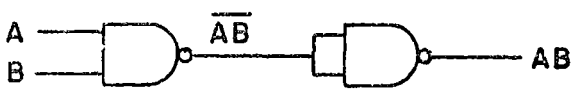


Es decir, utilizando sólo compuertas NAND, se implementó un circuito que utilizaba compuertas AND y OR. Esto sugiere que podemos implementar dichas

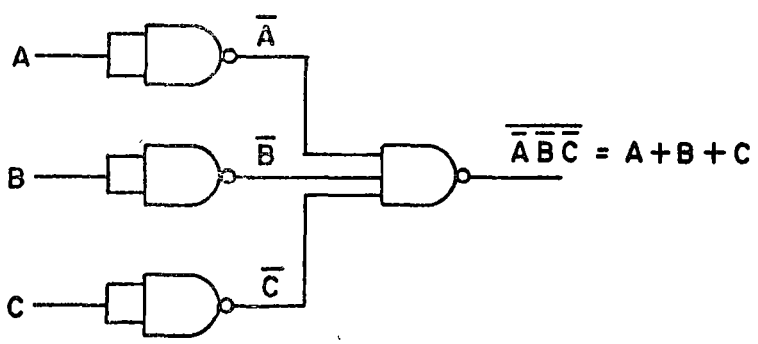
funciones (AND y OR) mediante compuertas NAND, y como además, ésta tiene implícita la función NOT, tendremos una compuerta mediante la cual se puede implementar cualquier función:

NOT

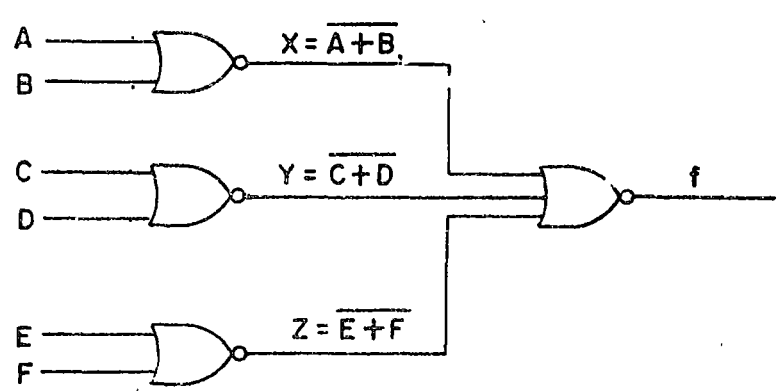
AND:



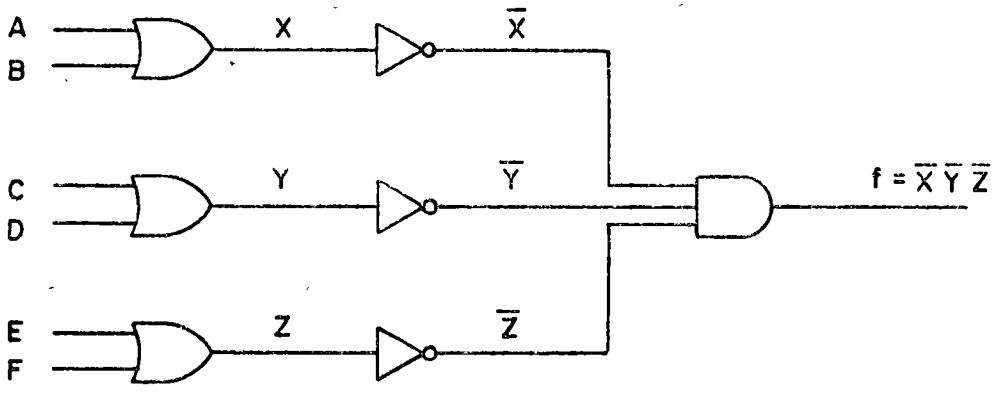
OR:



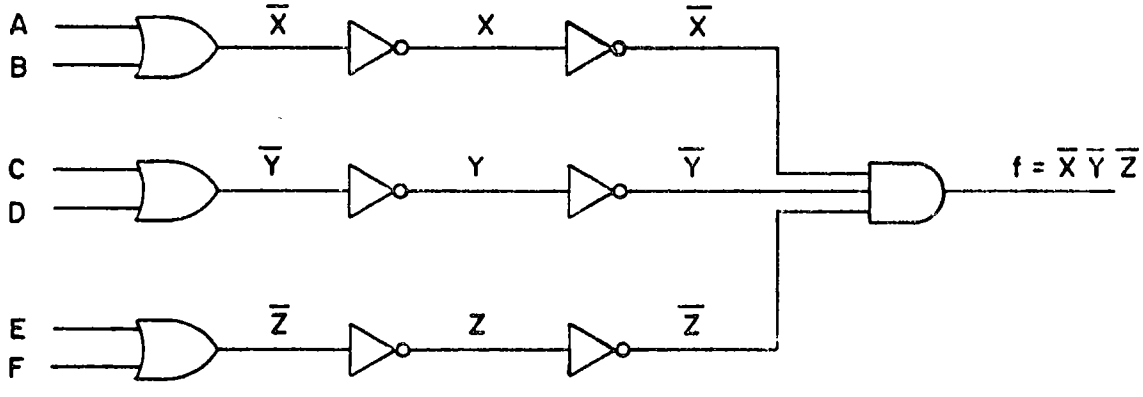
Lo anterior también se cumple para circuitos NOR como veremos a continuación: Consideremos nuevamente un circuito de segundo orden, pero a base de compuertas NOR:



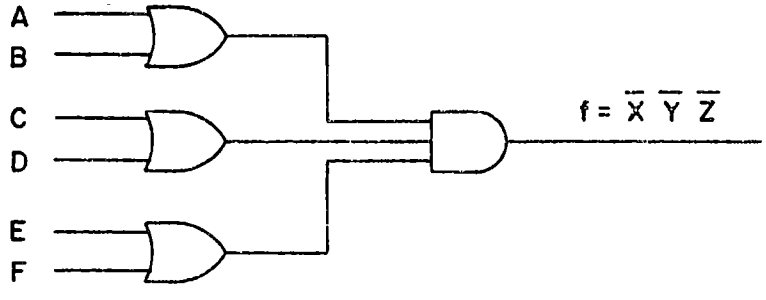
Dicha función se puede también implementar de la siguiente forma:



o bién



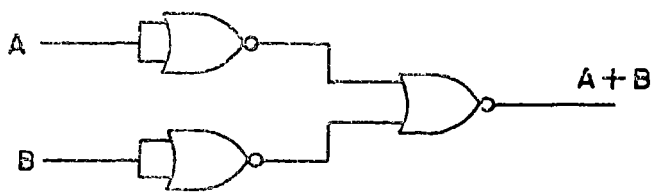
o bién



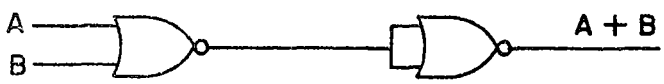
Las funciones NOT, AND y OR implementadas mediante compuertas NOR, toman las siguientes formas:

NOT:

AND:



OR:



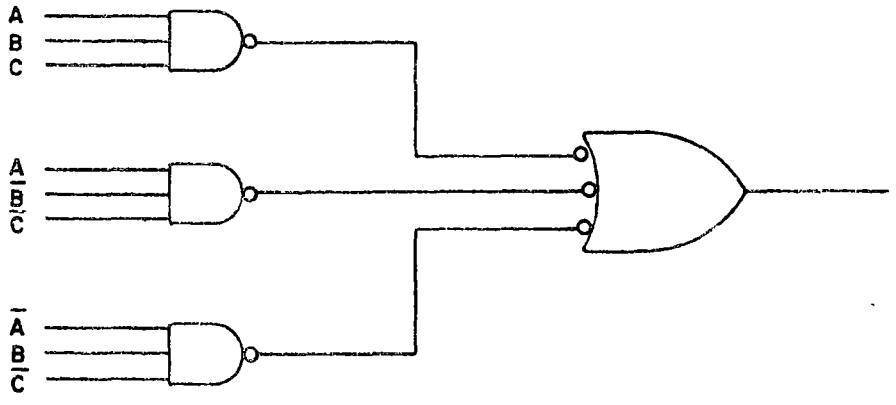
3.4.- Análisis de Circuitos NAND y NOR.

En la sección anterior vimos que es posible implementar cualquier función Booleana empleando solamente compuertas NAND o compuertas NOR. Vimos que cada una de las funciones básicas NOT, AND y OR tienen un circuito equivalente NAND y otro NOR. Una posibilidad sería entonces utilizar dichos circuitos equivalentes para convertir un circuito AND-OR a NAND o un circuito OR-AND a NOR o viceversa. Obviamente, aún cuando es posible hacerlo, la solución es impráctica debido a que aumenta considerablemente el número de compuertas usadas. A continuación veremos que el procedimiento es bastante más simple:

Consideremos la siguiente función:

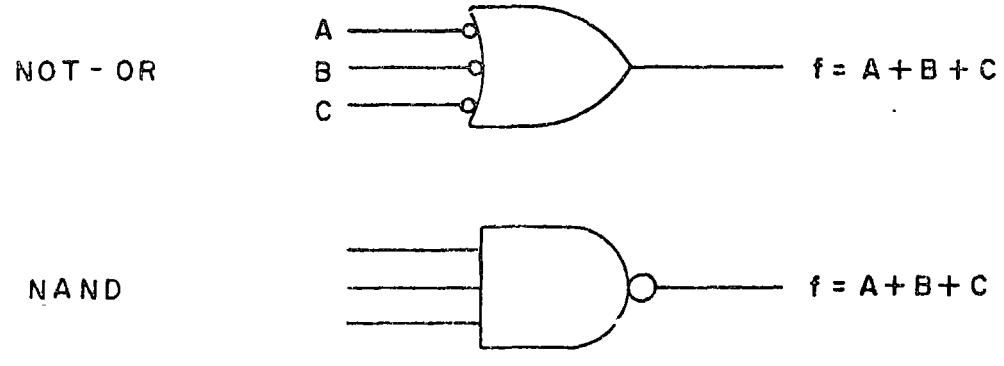
$$f(A,B,C) = ABC + A\bar{B}\bar{C} + \bar{A}B\bar{C}$$

En el circuito anterior tenemos una compuerta AND seguida de un inversor, lo que equivale a tener una compuerta NAND, por lo tanto el circuito se modifica de la siguiente forma:

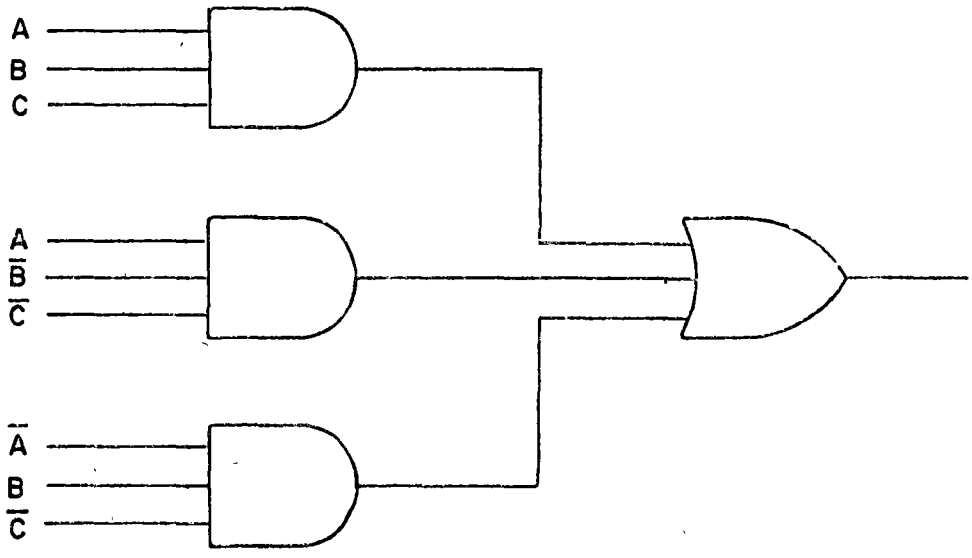


De esta forma hemos obtenido un circuito compuesto de tres compuertas NAND y una compuerta que invierte y ORea las señales a la entrada. Esta última compuerta no es familiar; sin embargo, como veremos es otra compuerta NAND, lo cual se deduce de las leyes de De Morgan:

Consideremos la compuerta NOT-OR Y LA NAND.



es decir, ambas compuertas son idénticas, luego el circuito queda finalmente:

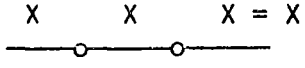


Dicha función está en forma de suma de productos y su implementación se realiza mediante compuertas AND/OR como se muestra en la figura.

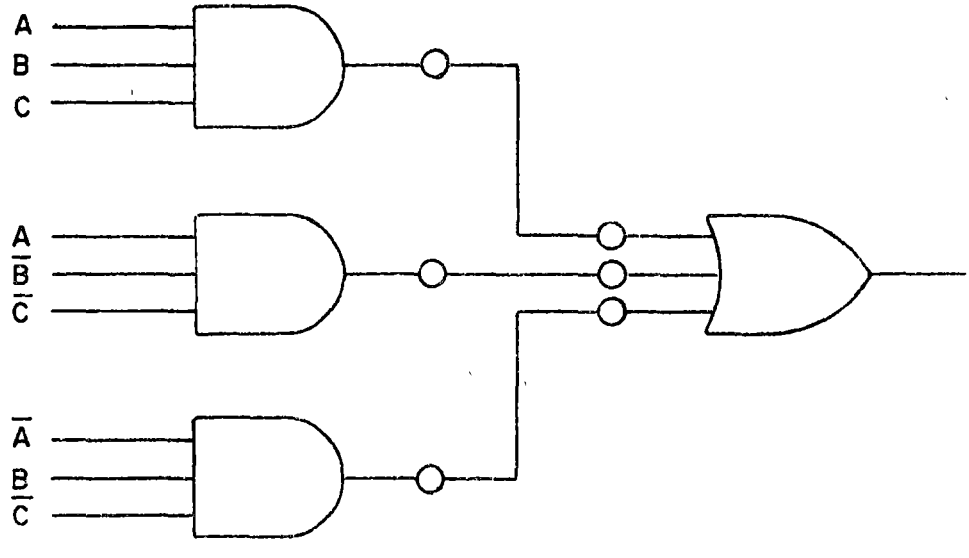
Del álgebra de boole sabemos que;

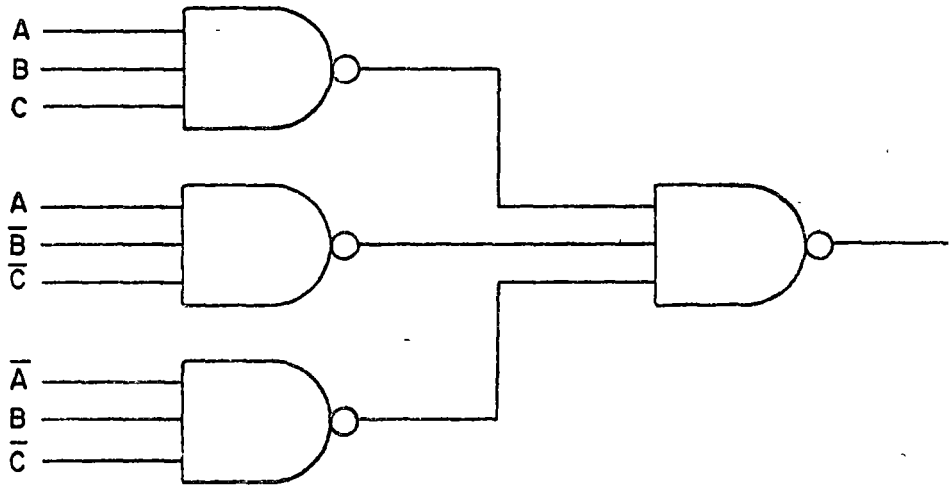
$$\overline{\overline{X}} = X$$

usaremos círculos pequeños para indicar la complementación, por ejemplo, en una línea tenemos:



Con esta conversión, podemos modificar el circuito anterior sin alterarlo, de la siguiente forma:





Es decir, para obtener un circuito con compuertas NAND a partir de AND-OR, simplemente reemplazamos las compuertas por NAND. A-nque lo anterior es cierto para la función anterior y en general para todas las funciones en forma de suma de productos, el procedimiento no es totalmente general y hay que observar ciertas reglas.

En general, podemos resumir lo siguiente:

1.- UNA EXPRESION LOGICA EN FORMA DE SUMA DE PRODUCTOS SE PUEDE REALIZAR DE DOS FORMAS MEDIANTE DOS NIVELES DE COMPUERTAS.

i) UN NIVEL DE AND SEGUIDO POR UN NIVEL DE OR.

ii) DOS NIVELES DE NAND

2.- UNA EXPRESION LOGICA EN FORMA DE PRODUCTO DE SUMAS se puede realizar de dos formas mediante dos niveles de computar.

i) UN NIVEL DE OR SEGUIDO POR UN NIVEL DE AND.

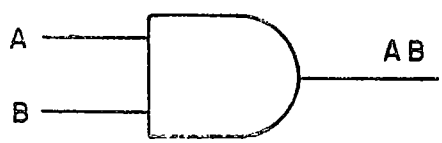
ii) DOS NIVELES DE NOR.

En este punto conviene definir un nuevo concepto: el de lógica positiva y lógica negativa, que nos permitirá aclarar la dualidad de las compuertas del ajemplo anterior.

3.4.1.- Lógica Positiva y Negativa

Al usar una tabla de verdad para describir operaciones lógicas, no hacemos referencia a los nivels de voltaje con respecto a tierra. Sin embargo, al asignar el "1" o el "0" lógico a un nivel, cambiará la función lógica de la compuerta, según la forma en que se asignen los valores lógicos a los niveles de voltaje.

Consideremos el caso siguiente: típicamente los voltajes de trabajo de las compuertas TTL so de 0.2V y 3.0V. Supongamos que una compuerta AND la hemos definido asignando a 0.2V el "0" lógico y a 3.0V el "1" lógico. Tenemos entonces las siguientes tablas de verdad.



0.2 → 0

3.0 → 1

A	B	A·B
0.2	0.2	0.2
0.2	3.0	0.2
3.0	0.2	0.2
3.0	3.0	3.0

A	B	AB
0	0	0
0	1	0
1	0	0
1	1	1

Si ahora para la misma compuerta cambiamos nuestra conversión y hacemos que 0.2V le corresponda el "1" lógico y que a 3.0V le corresponda el "0" lógico, la tabla de verdad queda:

A	B	A·B
1	1	1
1	0	1
0	1	1
0	0	0

Veremos que ahora obtenemos una función diferente referida al caso anterior.

En el primer caso, cuando 0.2V → "0" y 3.0 → "1" se tiene Lógica positiva y en el segundo caso cuando 0.2V → "1" y 3.0 → "0" se tiene Lógica negativa.

En los cuadros siguientes se muestran las tablas de verdad de las funciones más comunes, en lógica positiva y lógica negativa.

LOGICA POSITIVA

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

LOGICA NEGATÍVA

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

Comparando ambos cuadros vemos que existen las siguientes relaciones:

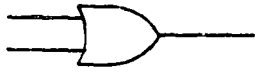
LOGICA POSITIVA		LOGICA NEGATIVA
AND	equivale →	OR
OR	equivale →	AND
NAND	equivale →	NOR
NOR	equivale →	NAND

Estos resultados tienen implicaciones prácticas muy importantes, ya que mejoran substancialmente la flexibilidad de diseño. En lo que sigue veremos como utilizar estas propiedades.

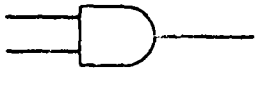
Símbolos

LOGICA POSITIVA

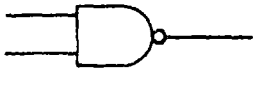
OR



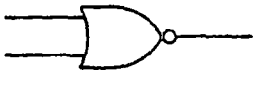
AND



NAND



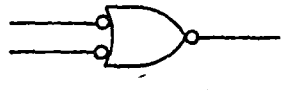
NOR



LOGICA NEGATIVA

AND

OR



NOR

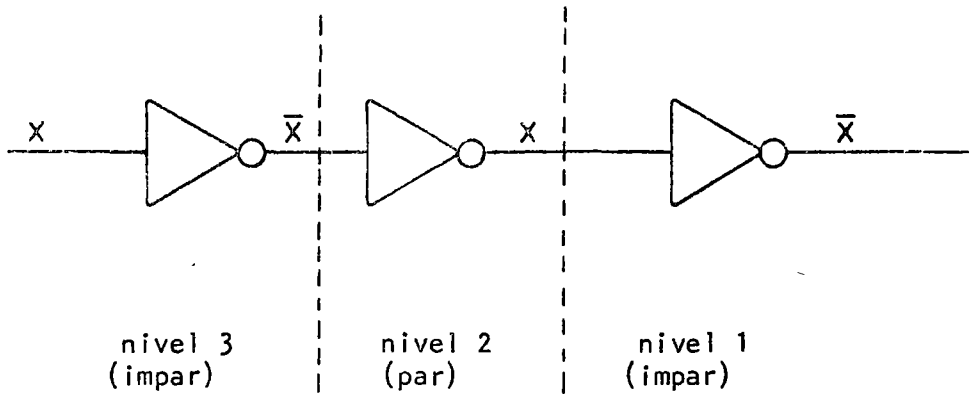
NAND

XOR 

XOR 

3.4.2.- Consderaciones para el análisis de circuitos combinacionales con NAND/NOR.

Consideremos el circuito de la figura, que consiste de una cadena de inversores:



Observemos en dicho circuito que la seañ de entrada X se obtiene en forma complementada en ciertos niveles y sin complementar en otros .

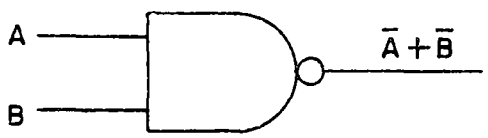
En general:

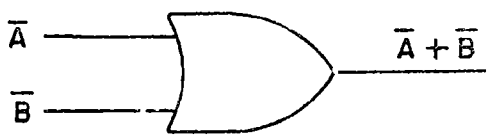
> SI LA VARIABLE PASA A TRAVES DE LOS NUMEROS PAR DE NIVELES DE INVERSION, APARECERA A LA SALIDA EN LA MISMA FORMA QUE A LA ENTRADA, SI EN CAMBIO PASA POR UN NUMERO IMPAR DE NIVELES DE INVERSION APARECERA A LA SALIDA COMO EL COMPLEMENTO DE LA ENTRADA.

Para aplicar esta regla, comenzamos a numerar los niveles desde la salida sin complementar y las entradas en la forma correspondiente.

La regla anterior nos será de mucha utilidad para analizar y sintetizar circuitos con compuertas NAND o NOR.

Analicemos algunas características de la compuerta NAND. Anteriormente vimos que una compuerta NAND se comportaba como una compuerta OR con las entradas complementadas.





Esto lo podemos interpretar en términos de niveles de inversión y de lógica positiva y negativa, de la siguiente forma:

Para una compuerta en un nivel impar las variables de entrada a la compuerta están complementadas. En términos de lógica positiva, están en su nivel bajo. Si usamos lógica mezclada (positiva y negativa) podemos usar lógica negativa en los niveles impares de inversión. Como la compuerta es de lógica postiva, tendremos lo siguiente:

A	B	NAND
0	0	1
0	1	1
1	0	1
1	1	0

A	B	"OR"
1	1	1
1	0	1
0	1	1
0	0	0

} Lógica negativa a la entrada.

y vemos que la compuerta NAND se comporta como una compuerta OR cuando asignamos lógica negativa a las entradas, lo que hacemos en los niveles impares de inversión.

Un argumento similar se puede aplicar al caso de compuertas NOR y veremos que ésta se comporta como una compuerta AND en niveles impares de inversión.

Similarmente, para una compuerta NAND en un nivel par, sus entradas estarán sin complementar y su salida estará complementada; si consideramos la salida como lógica negativa y las entradas como lógica positiva tendremos:

A	B	NAND
0	0	1
0	1	1
1	0	1
1	1	0

A	B	"AND"
0	0	0
0	1	0
1	0	0
1	1	1

} Lógica negativa a la salida.

De donde vemos que bajo dichas condiciones la compuerta NAND se comporta como una AND en niveles pares de inversión.

Resumiendo:

	Comportamiento	
	Nivel par	Nivel impar
NAND	AND	OR
NOR	OR	AND

Utilizando estos resultados podemos elaborar el siguiente conjunto de reglas para obtener la expresión de salida de un circuito que utilice compuertas inversoras:

1.- CONSIDERAR LA COMPUERTA DESDE LA CUAL SE OBTENDRA LA SEÑAL DE SALIDA COMO EL PRIMER NIVEL DE INVERSION (IMPAR): LA COMPUERTA PRECEDENTE COMO SEGUNDO NIVEL, ETC.

2.- CONSIDERAR TODAS LAS COMPUERTAS NAND EN NIVELES IMPARES REALIZANDO LA FUNCION OR.

3.- CONSIDERAR TODAS LAS COMPUERTAS NAND EN NIVELES PARES, REALIZANDO LA FUNCION AND.

4.- CONSIDERAR TODAS LAS COMPUERT AS NOR EN NIVELES IMPARES REALIZANDO LA FUNCION AND.

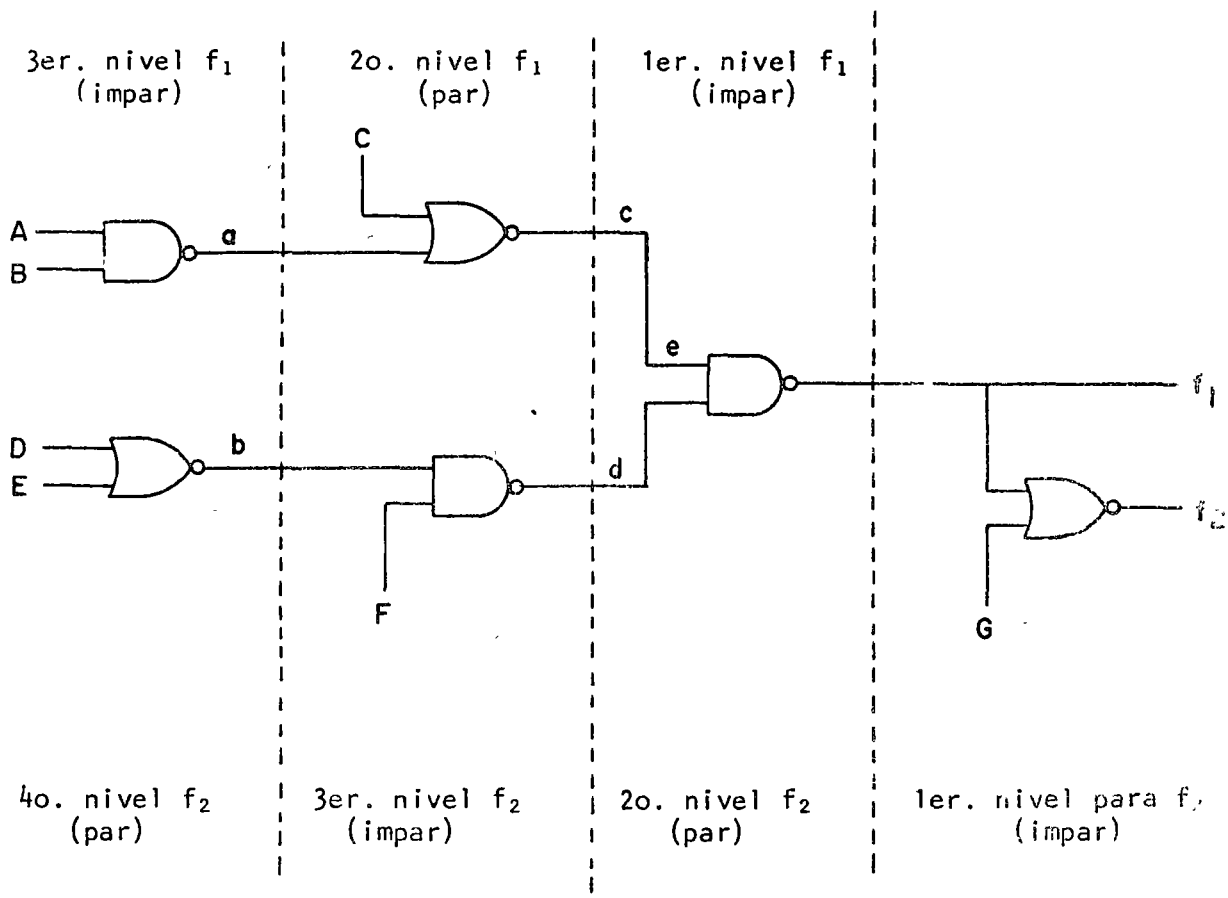
5.- CONSIDERAR TODAS LAS COMPUERTAS NOR EN NIVELES PARES REALIZANDO LA FUNCION OR.

6.- TODAS LAS VARIABLES DE ENTRADA A COMPUERTAS EN NIVELES IMPARES DEBEN APARECER EN FORMA COMPLEMENTADA EN LA EXPRESION DE LA SEÑAL DE SALIDA.

7.- TODAS LAS VARIABLES DE ENTRADA A COMPUERTAS EN NIVEL PAR DEBEN APARECER SIN COMPLEMENTAR EN LA EXPRESION DE LA SEÑAL DE SALIDA.

Observación: Estas reglas se utilizan para pasar de una configuración AND-OR (OR-AND) a NAND (NOR) o para derivar la expresión de la señal de salida de un circuito de compuertas NAND-NOR en términos de compuertas AND-OR.

Ejemplo: Derivar las expresiones para f_1 y f_2



Para f_1:

a) La compuerta NAND # 5 está en el primer nivel (impar) y se debe considerar como realizando la función OR.

b) La compuerta NAND # 4 está en el segundo nivel (par) y se debe considerar como realizando la función AND; además la variable F debe aparecer sin complementar en la expresión para f_1.

c) La compuerta NOR # 3 está en el segundo nivel (par) y se debe considerar como realizando la función OR; la variable C debe aparecer sin complementar en la expresión para f_1.

d) NOR # 2 está en el tercer nivel para f_1 (impar), se debe considerar como realizando la función AND. Las variables D y E deben aparecer complementadas en la expresión para f_1.

e) NAND # 1 está en el tercer nivel para f_1 (impar), se debe considerar como realizando la función OR. Las variables A y B deben aparecer complementadas en la expresión para f_1.

Luego:

1er. nivel NAND # 5

$$f_1 = \overbrace{(\underbrace{\bar{A} + \bar{B}}_{\text{3er.n.}} + \underbrace{C}_{\text{2o.n.}}) + \underbrace{\bar{D}\,\bar{E}}_{\text{3 n.}}\,\underbrace{F}_{\text{2o. n}}}$$

NAND1 NOR3 NOR2 NAND4

De otra manera habríamos tenido

$$f_1 = \overline{c\,d} \qquad c = \overline{c + a} \qquad a = \overline{AB}$$

$$d = \overline{Fb} \qquad b = \overline{D + E}$$

$$f_2 = \overline{(\overline{c + a})(\overline{Fb})} = \overline{\overline{(C + \overline{AB})}\ \overline{(F(\overline{D + E}))}}$$

$$f_2 = \overline{\overline{(C + \overline{AB})}} + \overline{\overline{(F(\overline{D + E}))}}$$

$$f_2 = (C + \bar{A} + \bar{B}) + F\bar{D}\bar{E}$$

Para f_2:

a) NOR # 6: nivel impar; considerar como AND; la variable B debe aparecer complementada en expresión para f_2.

b) NAND # 5: nivel par; considerar como AND

c) NAND # 4: nivel impar; considerar como OR; la variable F debe aparecer complementada en la expresión para f_2.

d) NOR # 3: nivel impar; considerar como AND; la variable C debe aparecer complementada en f_2.

e) NOR # 2: nivel par; considerar como OR; las variables D y E deben aparecer sin complementar en f_2.

f) NAND # 1: nivel par; considerar como AND, las variables A y B deben aparecer sin complementar en f_2.

Luego:

$$f_2 = \bar{G}\left[(\bar{F} + (D + E))\ (\bar{C}(A \cdot B))\right] = AB\bar{C}\bar{G}\ (D + E + F)$$

De otra manera habríamos tenido que proceder como sigue:

$$f_2 = \overline{e + G} \qquad e = \overline{cd} \qquad c = \overline{C + a} \qquad a = \overline{AB}$$

$$d = \overline{Fb} \qquad b = \overline{D + E}$$

$$f_2 = \overline{\overline{cd} + G}$$

$$= \overline{(\overline{\overline{C + a}})\ \overline{FB} + G}$$

$$= \overline{(C + \overline{Ab})\ F\ \overline{(\overline{D + E})} + G}$$

$$= \bar{G}\left[\overline{C + \overline{AB}}\ \ (F\overline{(D + E)})\right]$$

$$= \bar{G}\ (\bar{C}AB)\ (\bar{F} + D + E)$$

$$= AB\bar{C}\bar{G}\ (D + E + \bar{F})$$

o bién:

$$f_2 = \overline{f_1 + G} = \bar{f}_1\ \bar{G}$$

pero $f_1 = (\bar{A} + \bar{B} + C) + \bar{D}\bar{E}F$

luego $\bar{f}_1 = ABC\ (D + E + \bar{F})$

y $f_2 = AB\bar{C}\bar{G}\ (D + E + \bar{F})$

En el ejemplo anterior vimos una aplicación de las reglas establecidas anteriormente. En este caso se trató de derivar las expresiones de las funciones de salida de un circuito NAND-NOR, refiriendo las compuertas a NAND y OR según el caso.

A continuación veremos síntesis de circuitos NAND y NOR y extenderemos la aplicación de las reglas para emplearlas en la síntesis de circuitos NAND y NOR.

3.5.- Diseño con compuertas NAND y NOR.

De lo que se ha visto hasta este punto, se puede intuir un proceso de síntesis de circuitos implementados con NAND y NOR. A continuación veremos dos métodos de síntesis:

3.5.1.- Método de Doble complemento: El método más simple para realizar una función booleana con compuertas NAND es mediante la aplicación del teorema de DeMorgan a una función en forma de suma de productos. La expresión resultante se puede realizar directamente mediante compuertas NAND.

Ejemplo:

$$f = A\bar{C} + BCD + \bar{A}D$$

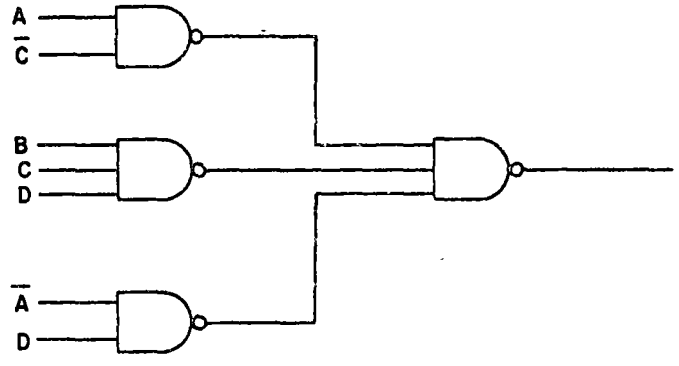
Si complementamos dos veces la función, obtenemos nuevamente la función:

$$f = \overline{\overline{A\bar{C} + BCD + \bar{A}D}}$$

Usando una de las leyes de DeMorgan, la funció queda:

$$f = \overline{A\bar{C}\ \overline{BCD}\ \overline{AD}} = \overline{X_1X_2X_3}$$

En esta expresión reconocemos inmediatamente que puede ser realizada mediante compuertas NAND.



3.5.2.- Método de Transformación:

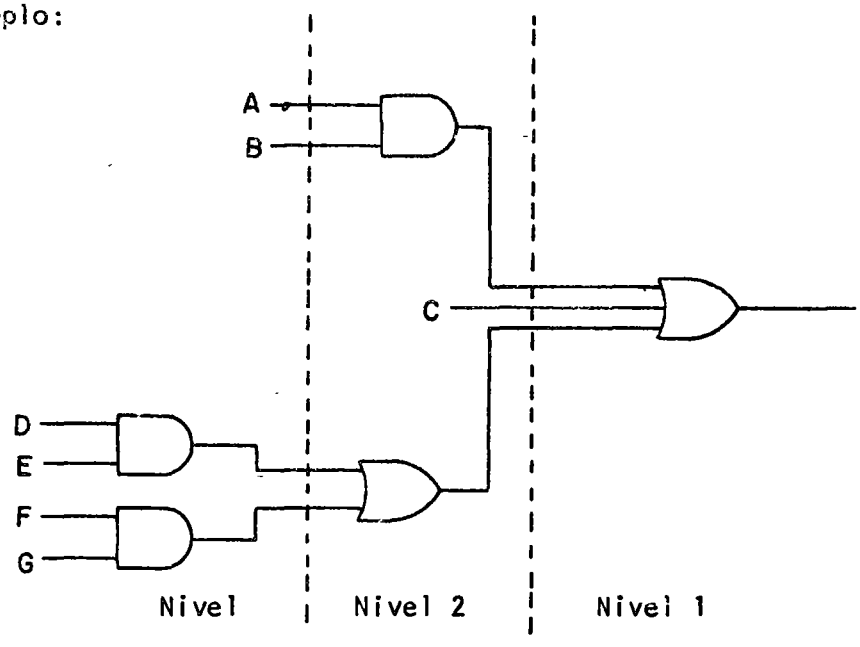
Este método consiste en la aplicación de las reglas desarrolladas en la sección anterior, con el objeto de transformar un circuito AND-OR en NAND. El procedimiento se resume en:

Regla:

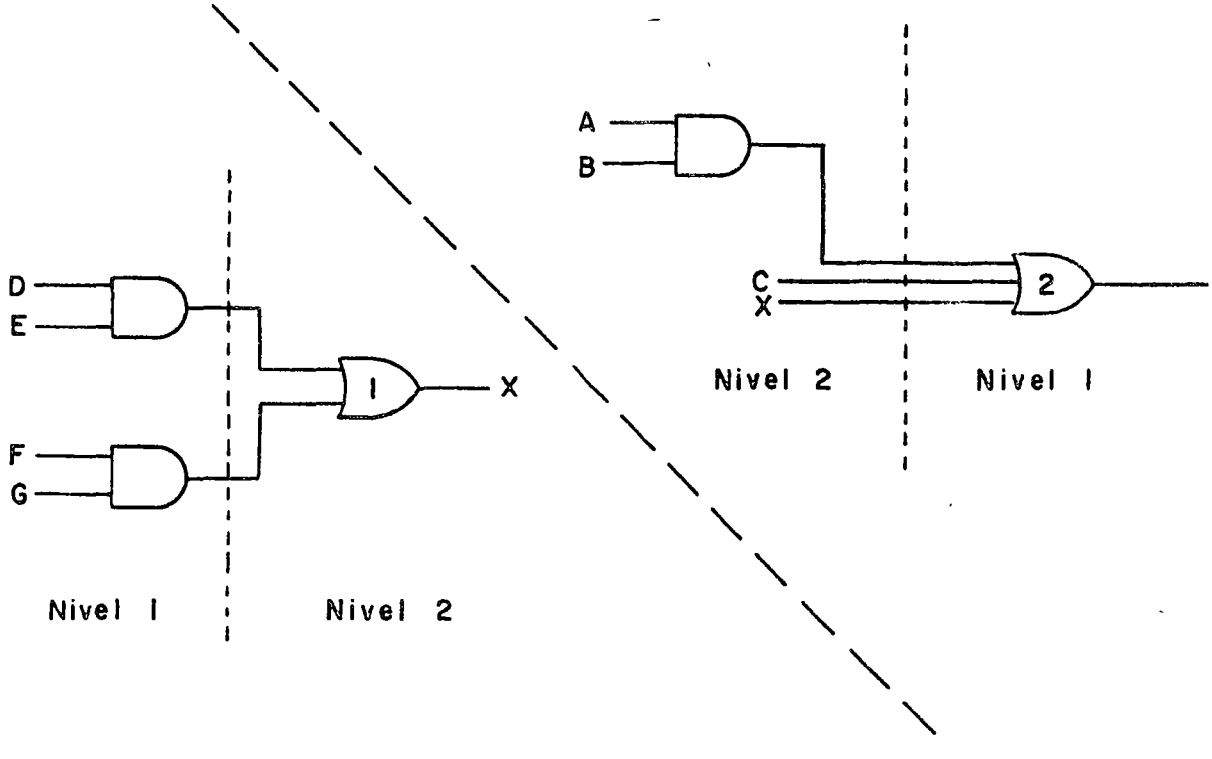
Arreglar el circuito en forma de niveles alternados de compuertas AND y OR,

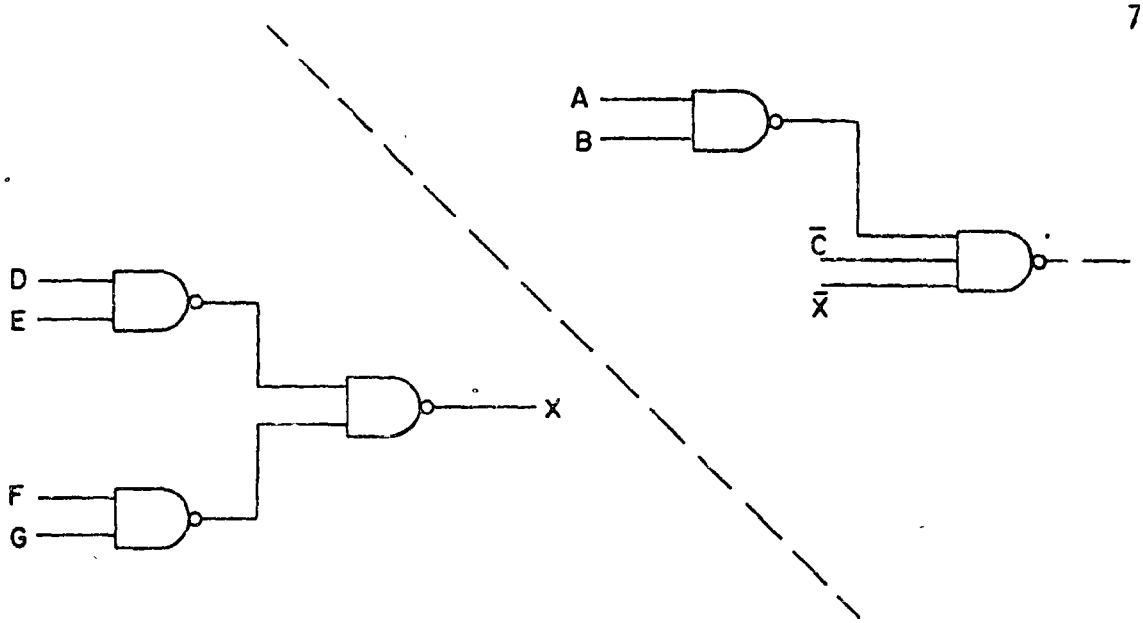
con una salida OR (AND). Reemplazar todas las compuertas AND y OR por compuertas NAND (NOR): Complementar todas las entradas a compuertas en niveles impares.

Ejemplo:

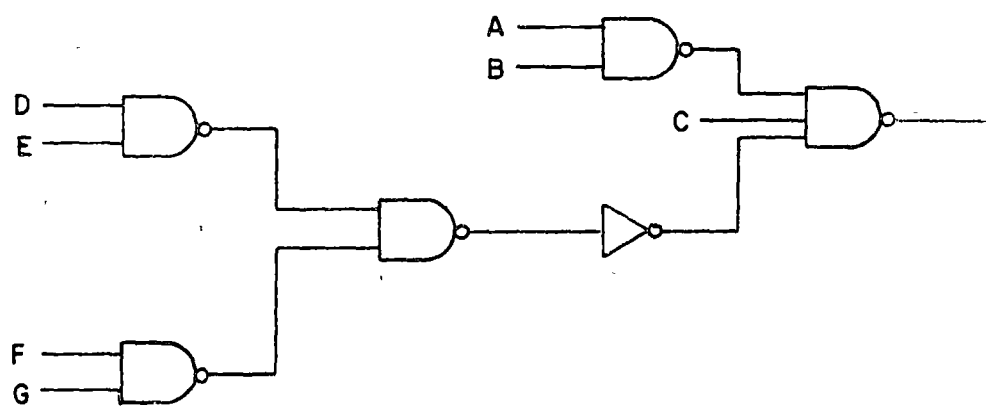


Este circuino m se puede arreglar en la forma contemplada por la Regla anterior debido a los dos niveles sucesivos de compuertas OR, por lo tanto, dividimos el circuito en dos subcircuitos de dos niveles AND-OR.





Obsérvese que la salida del OR-1 debe entrar complementada al OR 2, luego necesitamos poner un inversor.



Como hemos visto en los dos métodos anteriores, ambos implican la minimización utilizando mapas de Karnaugh para llegar a una forma de Suma de Productos o de Productos de Suma. Posteriormente se debe analizar si es posible obtener una forma más simplificada aún al factorizar la función.

3.5.- Circuitos NAND de tres niveles:

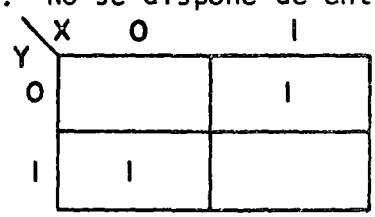
Hasta el momento, en nuestros métodos de minimización se han orientado hacia la obtención de circuitos con el mínimo número de compuertas y con el mínimo número total de entradas a compuertas, suponiéndose que se cuenta

con las variables de entrada en sus formas complementada y sin complementar. Hemos visto que se pueden obtener circuitos NAND en forma de suma de productos directamente del mapa de Karnough. Aún cuando estos circuitos son interesantes, su utilidad está basada en una suposición falsa: que se dispone de todas las entradas en su forma complementada y sin complementar, lo cual es un caso muy raro en la práctica. Si se usan soluciones en forma de suma de productos, nos encontraremos con que se requerirán varios inversores y nuestra solución dejará de ser mínima.

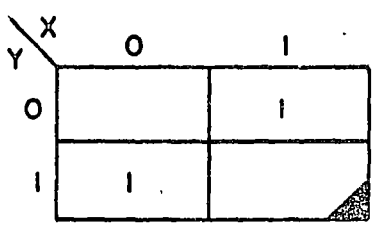
El problema de inversores se puede eliminar en gran parte usando circuitos de tres niveles. El procedimiento para obtener circuitos de tres niveles emplea el mapa de Karnough como herramienta básica. El procedimiento es similar al requerido para una solución en forma de suma de productos, con algunas modificaciones que se ilustran en el ejemplo desarrollado a continuación.

Ejemplo: Implementar la función $f = X\bar{Y} + \bar{X}Y$. No se dispone de entradas complementadas.

1.- Dibujar en mapa-K de la función deseada.

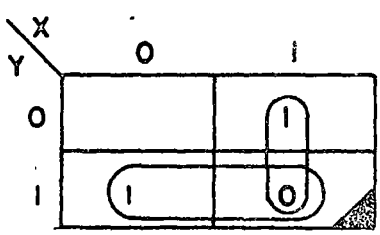


2.- Indicar en el mapa el o los cuadrados correspondientes a las entradas disponibles.



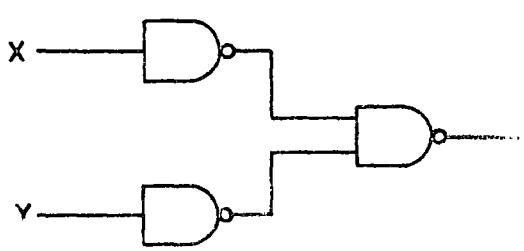
Sólo X y Y están disponibles.

3.- Todos los loops dibujados en el mapa deben incluir al cuadrado marcado. Dibujar loops implicante en el mapa, para cubrir todos los 1. Asegurarse que cada loop cubra un cuadrado marcado aún cuando encierre algunos 0s.

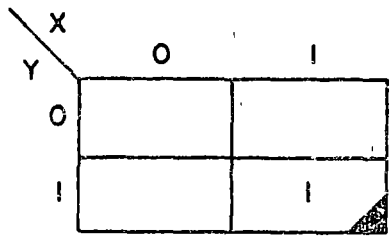


$f = X + y$

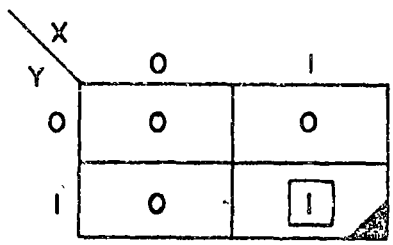
4.- Los loops se tratan temporalmente como un conjunto normal de implicantes y se construye un circuito. Se requiere un NAND por cada lazo más un NAND adicional para la salida. Las líneas de entrada a las NANDS se obtienen refiriéndose a lazos en el mapa.



5.- En este punto, el circuito es correcto excepto por los ceros que se han encerrado en los lazos. Dichos ceros deben ser inhibidos. Esto se logra considerando cicnos ceros como un nuevo problema y mapeándolos como 1 en un nuevo mapa.

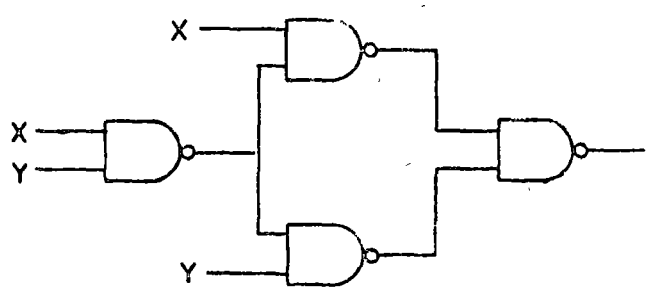


6.- Nuevamente, dibujar loops implicantes que cubran todos los 1 del mapa. Asegurarse que cada loop cubra un cuadrado marcado. En este punto ya no es necesario encerrar ceros.



7.- Se asigna una compuerta NAND a cada uno de los lazos. Las entradas a dichas compuertas se toman de los lazos en el mapa.

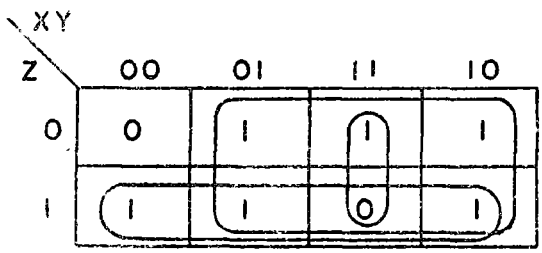
8.- Las salidas de estas compuertas se pueden considerar como términos inhibidores. Cada NAND se relaciona con un 0 específico o con un conjunto de ceros que se agruparon previamente en un lazo. Las salidas de estos bloques se conectan selectivamente a las NAND's previamente conectadas. Asegurarse qje cada cero en cada lazo es inhibido.

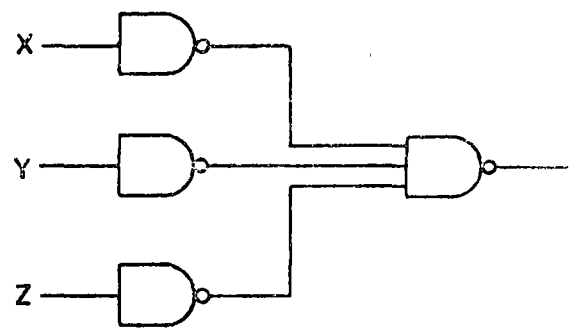


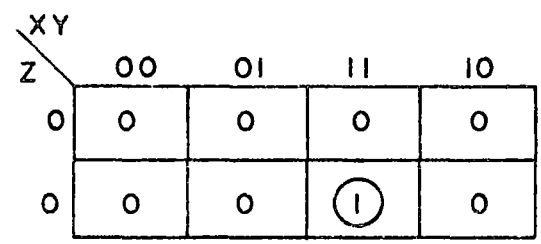
Si un cero es encerrado por más de un loop se le debe inhibir en cada uno de los loops que lo encerraba.

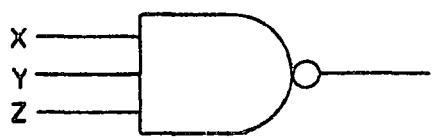
Ejemplo:

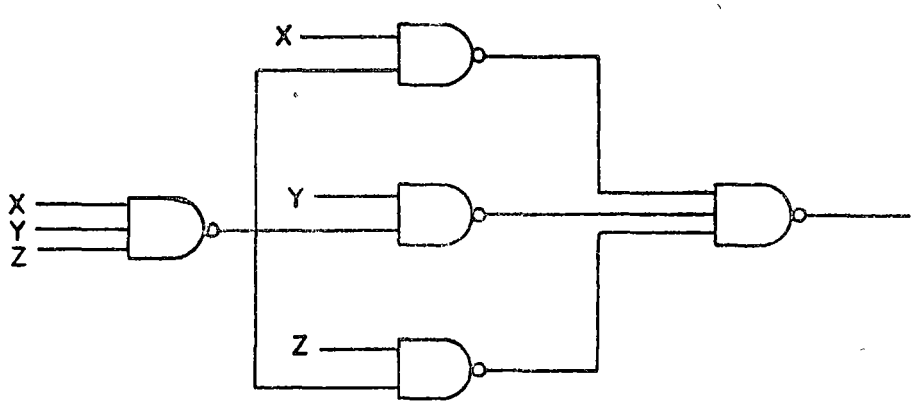
$f = \bar{X}\bar{Y}Z + \bar{X}Y\bar{Z} + \bar{X}YZ + X\bar{Y}\bar{Z} + X\bar{Y}Z + XY\bar{Z}$











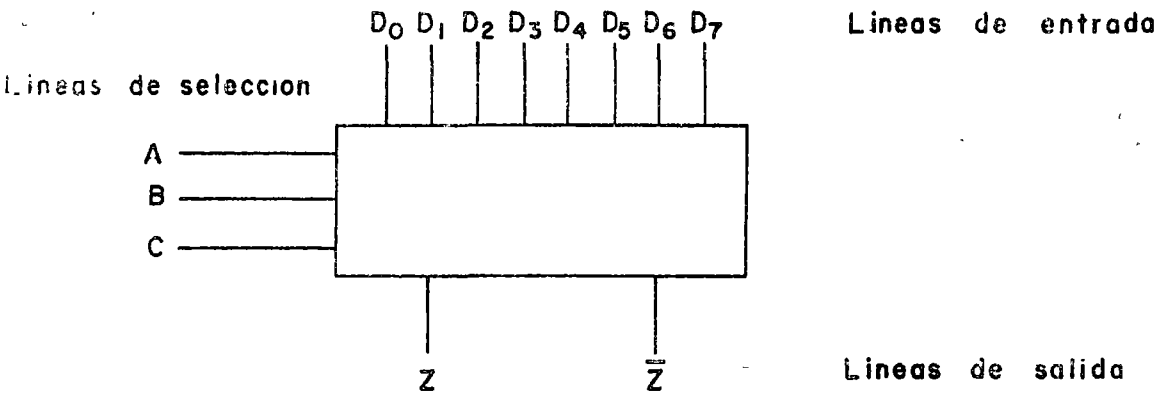
3.6.- Diseño con Multiplexores:

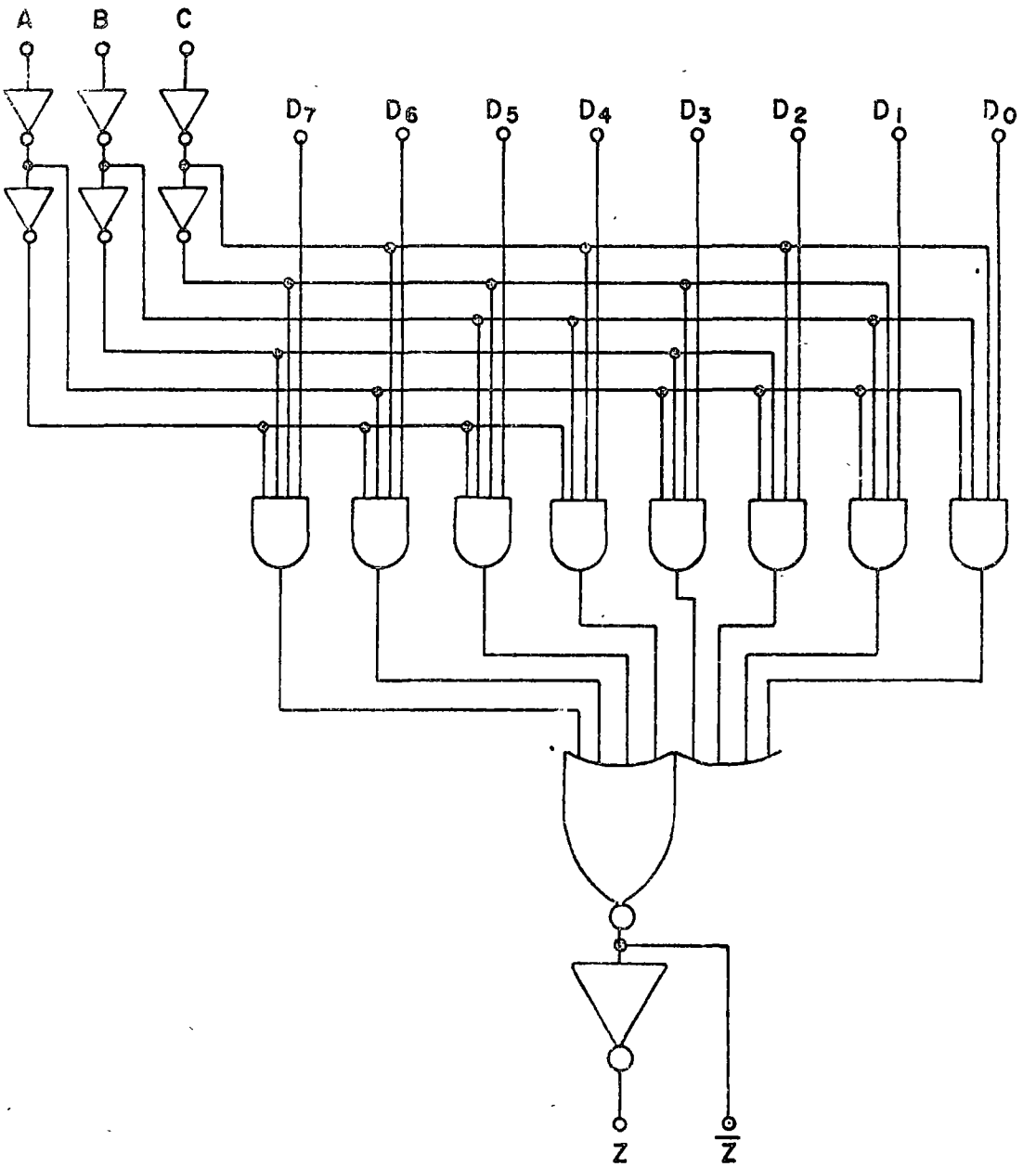
El principal atractivo que ofrecen los multiplexores es la posibilidad de "empacar" varias funciones lógicas en un mismo circuito integrado.

Básicamente, un multiplexor es un dispositivo con 2^n entradas, una salida (y su complemento), y n líneas de selección, que determinan cual de las 2^n entradas será conectada a la salida.

Comercialmente, se encuentran multiplexores de 2 entradas (4 Mux/circuito integrado), 4 entradas (2 mux/circuito integrado), 8 entradas (1 mux/circuito integrado) y 16 entradas. En la figura siguiente se muestra el símbolo de un multiplexor de 8 entradas, su tabla de verdad y el circuito a nivel de compuertas.

A	B	C	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	Z
0	0	0	0	X	X	X	X	X	X	X	0
0	0	0	1	X	X	X	X	X	X	X	1
0	0	1	X	0	X	X	X	X	X	X	0
0	0	1	X	1	X	X	X	X	X	X	1
0	1	0	X	X	0	X	X	X	X	X	0
0	1	0	X	X	1	X	X	X	X	X	1
0	1	1	X	X	X	0	X	X	X	X	0
0	1	1	X	X	X	1	X	X	X	X	1
1	0	0	X	X	X	X	0	X	X	X	0
1	0	0	X	X	X	X	1	X	X	X	1
1	0	1	X	X	X	X	X	0	X	X	0
1	0	1	X	X	X	X	X	1	X	X	1
1	1	0	X	X	X	X	X	X	0	X	0
1	1	0	X	X	X	X	X	X	1	X	1
1	1	1	X	X	X	X	X	X	X	0	0
1	1	1	X	X	X	X	X	X	X	1	1


D0 D1 D2 D3 D4 D5 D6 D7
Lineas de entrada
Lineas de seleccion
A
B
C
Z
$\overline{Z}$
Lineas de salida


A
B
C
D7
D6
D5
D4
D3
D2
D1
D0
Z
$\overline{Z}$

La ecuación de salida del multiplexor de 8 entradas es la siguiente:

$$Z = D_0\bar{A}\bar{B}\bar{C} + D_1\bar{A}\bar{B}C + D_2\bar{A}B\bar{C} + D_3\bar{A}BC + D_4A\bar{B}\bar{C} + D_5A\bar{B}C + D_6AB\bar{C} + D_7ABC$$

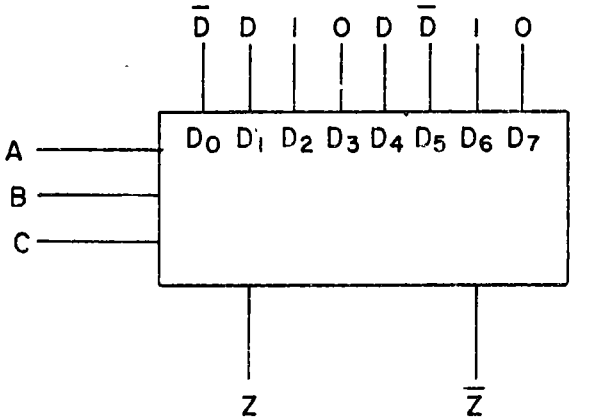
Analizando esta ecuación y observando al dispositivo, vemos que contamos con una entrada para cada una de las 8 combinaciones de las líneas de seleccion A,B y C. Por lo tanto, para una función de n variables, podemos "factorizar" A, B y C de la función y obtener ocho funciones separadas, cada una de n-3 variables, que deberán conectarse a la entrada correspondiente del multiplexor.

Por ejemplo consideremos la función:

$$f(A,B,C,D) = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}CD + \bar{A}B\bar{C}D + \bar{A}B\bar{C}\bar{D} + A\bar{B}\bar{C}D + A\bar{B}C\bar{D} + AB\bar{C}\bar{D} + ABC\bar{D}$$

Hagamos la siguiente tabla:

Entrada	Dirección	Otras Variables	
D_0	$\bar{A}\ \bar{B}\ \bar{C}$	$\bar{D}$	$\bar{D}$
D_1	$\bar{A}\ \bar{B}\ C$	D	D
D_2	$\bar{A}\ B\ \bar{C}$	$D + \bar{D}$	1
D_3	$\bar{A}\ B\ C$		0
D_4	$A\ \bar{B}\ \bar{C}$	D	D
D_5	$A\ \bar{B}\ C$	$\bar{D}$	D
D_6	$A\ B\ \bar{C}$	$\bar{D} + D$	1
D_7	$A\ B\ C$		0



$$Z = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}CD + \bar{A}B\bar{C} + A\bar{B}\bar{C}D + A\bar{B}CD + ABC$$

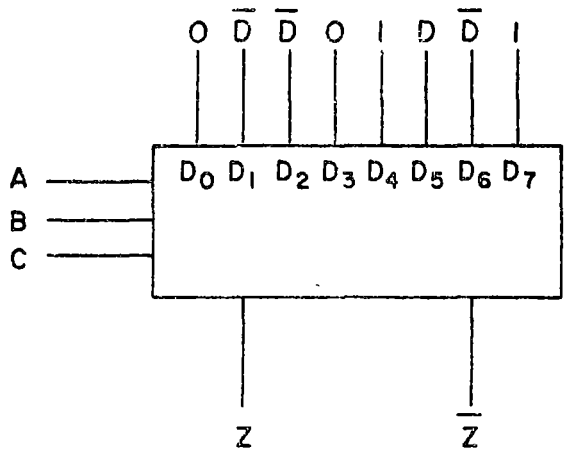
Consideremos otro ejemplo y veamos las distintas alternativas disponibles para su implementación.

$f(A,B,C,D) = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}B\bar{C}D + A\bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C}D + A\bar{B}CD + AB\bar{C}\bar{D} + ABC\bar{D} + ABCD$

1a. Alternativa: Emplear un multiplexor de 8 entradas

$f = \bar{A}\bar{B}CD + \bar{A}B\bar{C}D + A\bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C}D + A\bar{B}CD + AB\bar{C}\bar{D} + ABC\bar{D} + ABCD$

Entrada	Dirección	Otras Variables	
D_0	$\bar{A}\ \bar{B}\ \bar{C}$	0	0
D_1	$\bar{A}\ \bar{B}\ C$	$\bar{D}$	$\bar{D}$
D_2	$\bar{A}\ B\ \bar{C}$	D	D
D_3	$\bar{A}\ B\ C$	0	0
D_4	$A\ \bar{B}\ \bar{C}$	$\bar{D} + D$	1
D_5	$A\ \bar{B}\ C$	D	D
D_6	$A\ B\ \bar{C}$	$\bar{D}$	$\bar{D}$
D_7	$A\ B\ C$	$\bar{D} + D$	1

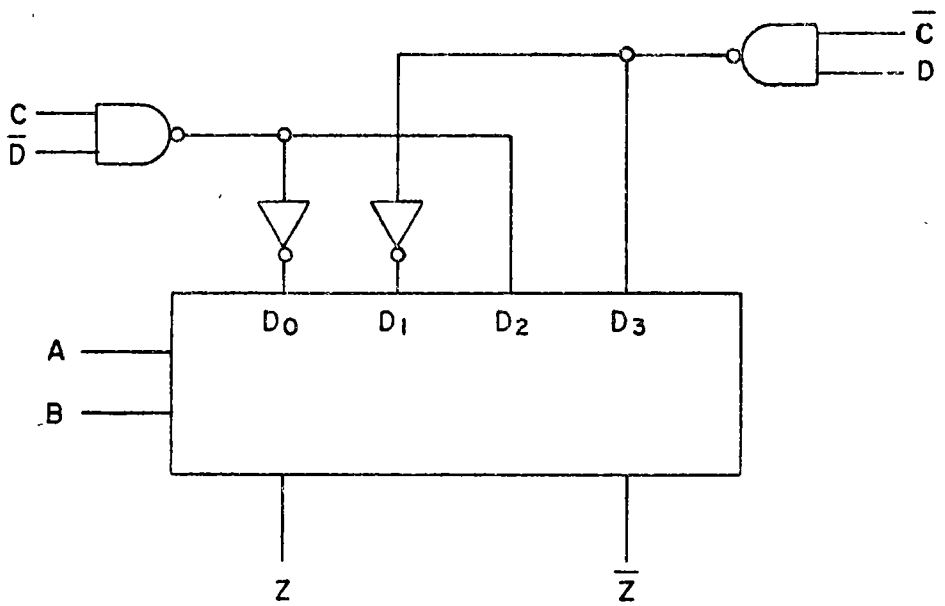


$Z = \bar{A}\bar{B}C\bar{D} + \bar{A}B\bar{C}D + A\bar{B}\bar{C} + A\bar{B}CD + AB\bar{C}\bar{D} + ABC$

2a. Alternativa: Un multiplexor de 4 entradas y lógica discreta

Entrada	Dirección	Otras Variables	
D_0	A B	C D	C D
D_1	A B	C D	C D
D_2	A B	C D + C D + C D	C + C D = C + D
D_3	A B	C D + C D + C D	C + C D = C + D

minimizar



3a. Alternativa: Emplear 4 mux de 2 entradas y 1 de 4 entradas

Para esto, generaremos cuatro tablas de verdad, cada una de las cuales servirá para definir las entradas del primer nivel de multiplexores.

A = D; B = 0

C	D	f_1
0	0	0
0	1	0
1	0	1
1	1	0

A = 0; B = 1

C	D	f_2
0	0	0
0	1	1
1	0	0
1	1	0

A = 1; B = 0

C	D	f_3
0	0	1
0	1	1
1	0	0
1	1	1

A =1; B = 1

C	D	f_4
0	0	1
0	1	0
1	0	1
1	1	1

A partir de estas cuatro sub-funciones, construimos las tablas de entrada a los multiplexores:

f_1:

Entrada	Dirección	Otras Variables
D_0	$\bar{C}$	0
D_1	C	$\bar{D}$

Salida: $C\bar{D}$

f_2:

Entrada	Dirección	Otras Variables
D_0	$\bar{C}$	D
D_1	C	0

Salida: CD

f_3:

Entrada	Dirección	Otras Variables
D_0	$\bar{C}$	1
D_1	C	D

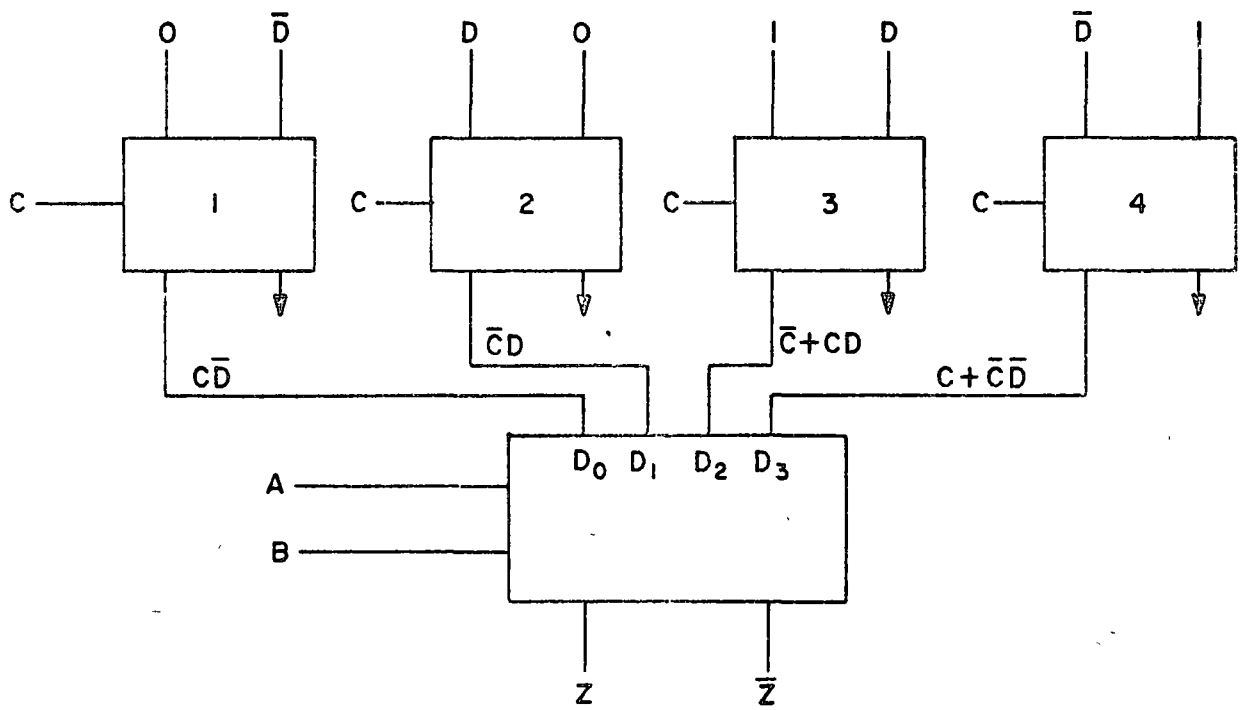
Salida: $C + CD$

f_4

Entrada	Dirección	Otras Variables
D_0	$\bar{C}$	$\bar{D}$
D_1	C	1

Salida: $C + \bar{C}\bar{D}$

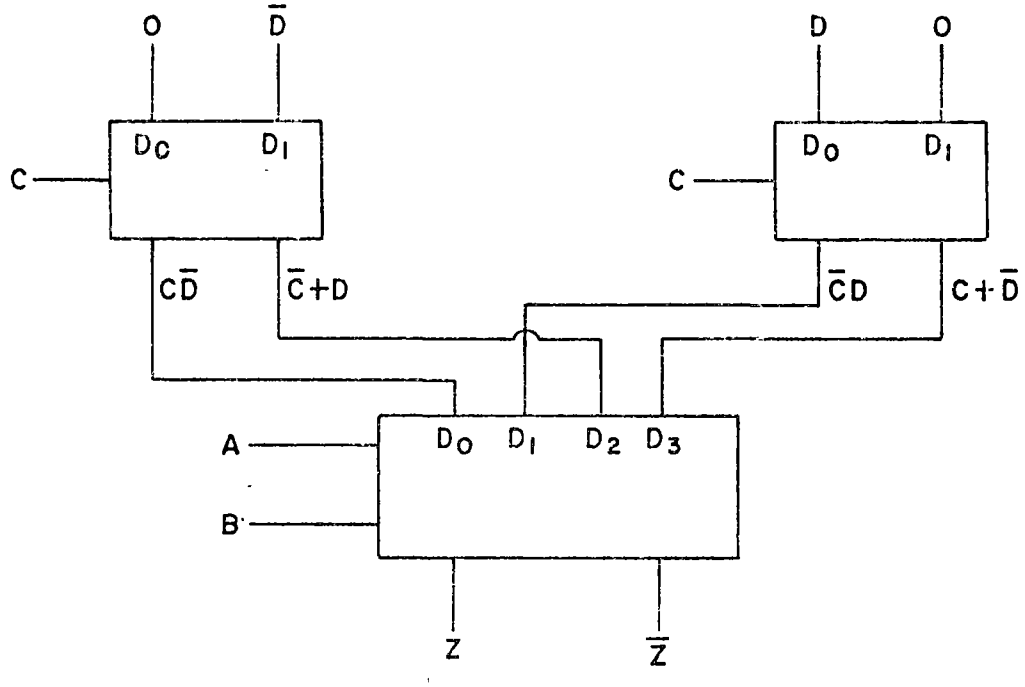
El circuito quedará como se muestra en la figura siguiente:



Sin embargo, si analizamos el complemento de las salidas de los multiplexores 1 y 2 vemos que:

$$\overline{C\bar{D}} = \bar{C} + D = \bar{C} + CD$$
$$\overline{\bar{C}D} = C + \bar{D} = C + \bar{C}\bar{D}$$

Luego, el circuito puede simplificarse aún más:



De los ejemplos anteriores podemos establecer la siguiente regla para el diseño de circuitos de dos niveles empleando multiplexores:

Regla:

1.- DIVIDIR LA FUNCION EN DOS PARTES ($g_1 = g_1(A,B,C)$ Y $g_2 = g_2(C,D,E)$ REDUCIENDO EL PROBLEMA DE 6 VARIABLES A 2 DE 3 VARIABLES.

2.- AGRUPAR LOS TERMINOS DE $g_1(A,B,C)$ CORRESPONDIENTES A CADA COMBINACION DE LAS VARIABLES D, E, F.

3.- IMPLEMENTAR CADA TERMINO DE $g_1(A,B,C)$ CON MUX DE 4 ENTRADAS.

4.- ALIMENTAR LAS 8 ENTRADAS DEL MUX DE SALIDA CON LAS SALIDAS DE LOS MUX DEL PRIMER NIVEL.

Veamos a continuación un último ejemplo, más complejo, que ilustre el uso de la regla anterior. El sistema está definido por la siguiente tabla de verdad.

A	B	C	D	E	F	f
0	0	0	0	0	0	1
0	0	0	0	0	1	0
0	0	0	0	1	0	1
0	0	0	0	1	1	0
0	0	0	1	0	0	1
0	0	0	1	0	1	1
0	0	0	1	1	0	1
0	0	0	1	1	1	0
0	0	1	0	0	0	1
0	0	1	0	0	1	1
0	0	1	0	1	0	1
0	0	1	0	1	1	1
0	0	1	1	0	0	1
0	0	1	1	0	1	1
0	0	1	1	1	0	1
0	0	1	1	1	1	1
0	1	0	0	0	0	0
0	1	0	0	0	1	1
0	1	0	0	1	0	0
0	1	0	0	1	1	0
0	1	0	1	0	0	0
0	1	0	1	0	1	0
0	1	0	1	1	0	0
0	1	0	1	1	1	0
0	1	1	0	0	0	1
0	1	1	0	0	1	1
0	1	1	0	1	0	1
0	1	1	0	1	1	0
0	1	1	1	0	0	1
0	1	1	1	0	1	1
0	1	1	1	1	0	1
0	1	1	1	1	1	1

A	B	C	D	E	F	f
1	0	0	0	0	0	1
1	0	0	0	0	1	1
1	0	0	0	1	0	0
1	0	0	0	1	1	0
1	0	0	1	0	0	0
1	0	0	1	0	1	1
1	0	0	1	1	0	1
1	0	0	1	1	1	0
1	0	1	0	0	0	0
1	0	1	0	0	1	0
1	0	1	0	1	0	0
1	0	1	0	1	1	1
1	0	1	1	0	0	1
1	0	1	1	0	1	1
1	0	1	1	1	0	0
1	0	1	1	1	1	0
1	1	0	0	0	0	1
1	1	0	0	0	1	0
1	1	0	0	1	0	0
1	1	0	0	1	1	1
1	1	0	1	0	0	0
1	1	0	1	0	1	0
1	1	0	1	1	0	0
1	1	0	1	1	1	1
1	1	1	0	0	0	0
1	1	1	0	0	1	0
1	1	1	0	1	0	1
1	1	1	0	1	1	1
1	1	1	1	0	0	1
1	1	1	1	0	1	0
1	1	1	1	1	0	0
1	1	1	1	1	1	1

En primer lugar tenemos que dividir la función original en dos funciones parciales g_1 y g_2. Este es el problema más complejo ya que dependiendo de la forma en que se agrupen las variables para formar g_1 y g_2, se puede obtener un circuito más simple o más complejo. Optemos ahora por la agrupación más obvia:

$g_1(A,B,C)$ y $g_2(D,E,F)$.

El segundo paso es agrupar los términos de $g_1(A,B,C)$ correspondientes a cada combinación de D, E y F.

Revisando la tabla de verdad, vemos que la función es 1 en las 8 combinaciones posibles de D, E y F. Por lo tanto, tendremos ocho grupos de términos para $g_1(A,B,C)$.

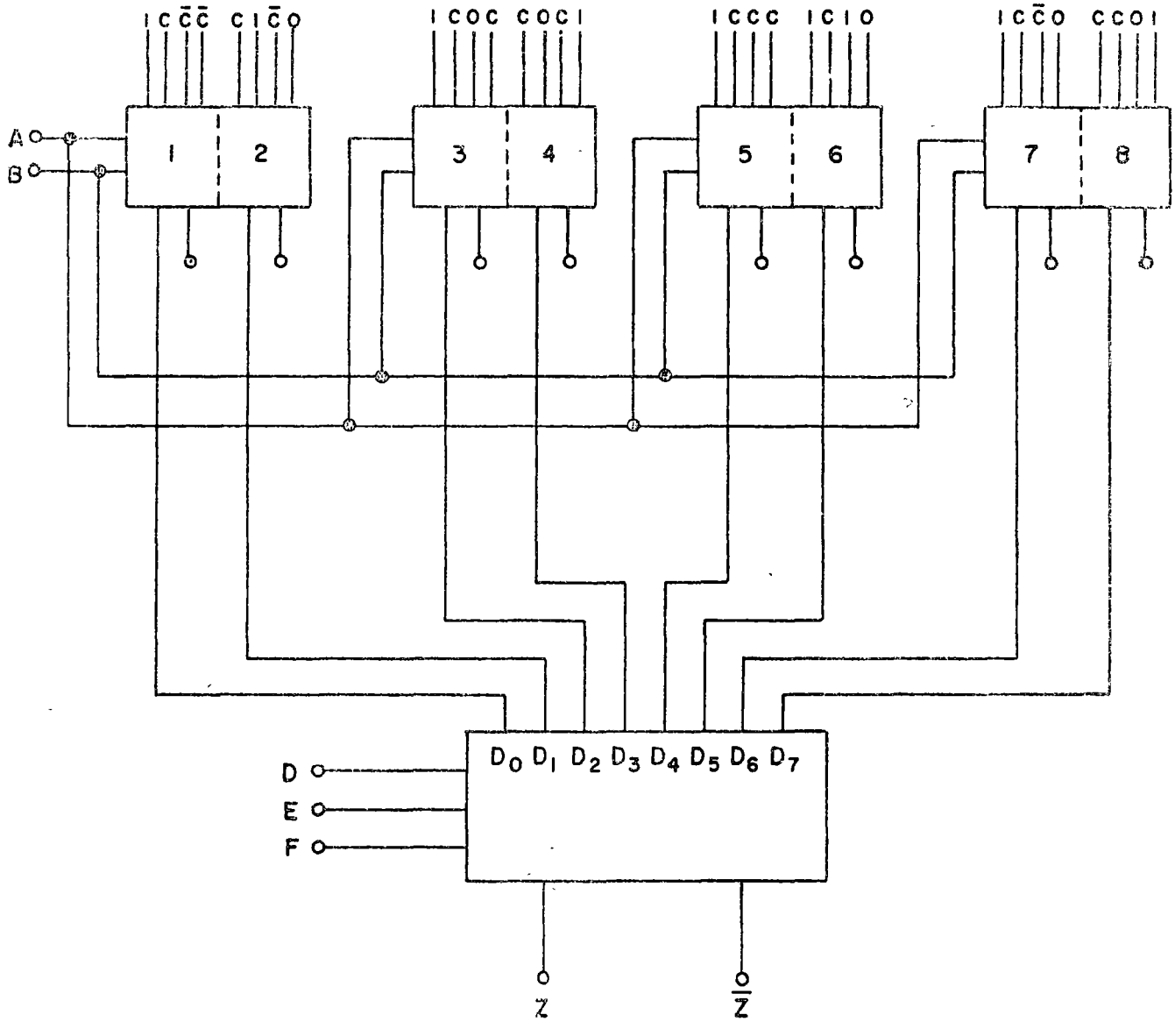
En la tabla siguiente se muestra lo anterior:

	A	B	C	D	E	F			Direcc. AB	
→ D_0	0	0	0	0	0	0		D_0	00	1
	0	0	1	0	0	0		D_1	01	C
	0	1	1	0	0	0	Mux 1:	D_2	10	$\bar{C}$
	1	0	0	0	0	0		D_3	11	$\bar{C}$
	1	1	0	0	0	0				
→ D_1	0	0	1	0	0	1		D_0	00	C
	0	1	0	0	0	1		D_1	01	1
	0	1	1	0	0	1	Mux 2:	D_2	10	$\bar{C}$
	1	0	0	0	0	1		D_3	11	0
→ D_2	0	0	0	0	1	0		D_0	00	1
	0	0	1	0	1	0		D_1	01	C
	0	1	1	0	1	0	Mux 3:	D_2	10	0
	1	1	1	0	1	0		D_3	11	C
→ D_3	0	0	1	0	1	1		D_0	00	C
	1	0	1	0	1	1		D_1	01	0
	1	1	0	0	1	1	MUX 4:	D_2	10	C
	1	1	1	0	1	1		D_3	11	1
→ D_4	0	0	0	1	0	0				
	0	0	1	1	0	0		D_0	00	1
	0	1	1	1	0	0	MUX 5:	D_1	01	C
	1	0	1	1	0	0		D_2	10	C
	1	1	1	1	0	0		D_3	11	C
→ D_5	0	0	0	1	0	1				
	0	0	1	1	0	1		D_0	00	1
	0	1	1	1	0	1	MUX 6:	D_1	01	C
	1	0	0	1	0	1		D_2	10	1
	1	0	1	1	0	1		D_3	11	0
→ D_6	0	0	0	1	1	0		D_0	00	1
	0	0	1	1	1	0		D_1	01	C
	0	1	1	1	1	0	MUX 7:	D_2	10	C
	1	0	0	1	1	0		D_3	11	0
→ D_7	0	0	1	1	1	1		D_0	00	C
	0	0	1	1	1	1		D_1	01	C
	1	1	0	1	1	1	MUX 8:	D_2	10	0
	1	1	1	1	1	1		D_3	11	1

Entradas al mux de salida.

Debemos realizar la primer columna mediante mux de 4 entradas. Necesitaremos un mux por cada entrada del mux de salida, ya que de la tabla vemos que no hay agrupaciones iguales que pudieran ser generadas con un sólo mux, ni tampoco hay agrupaciones que sean el complemento de la otra.

Como se decía, el problema más complejo, estriba en las agrupaciones que se hagan de las variables con el objeto de obtener las funciones g_1 y g_2. De esta agrupación dependerá si el número de multiplexores empleados es mínimo.



Como ejemplo de esto, repitamos el ejemplo con otra agrupación:

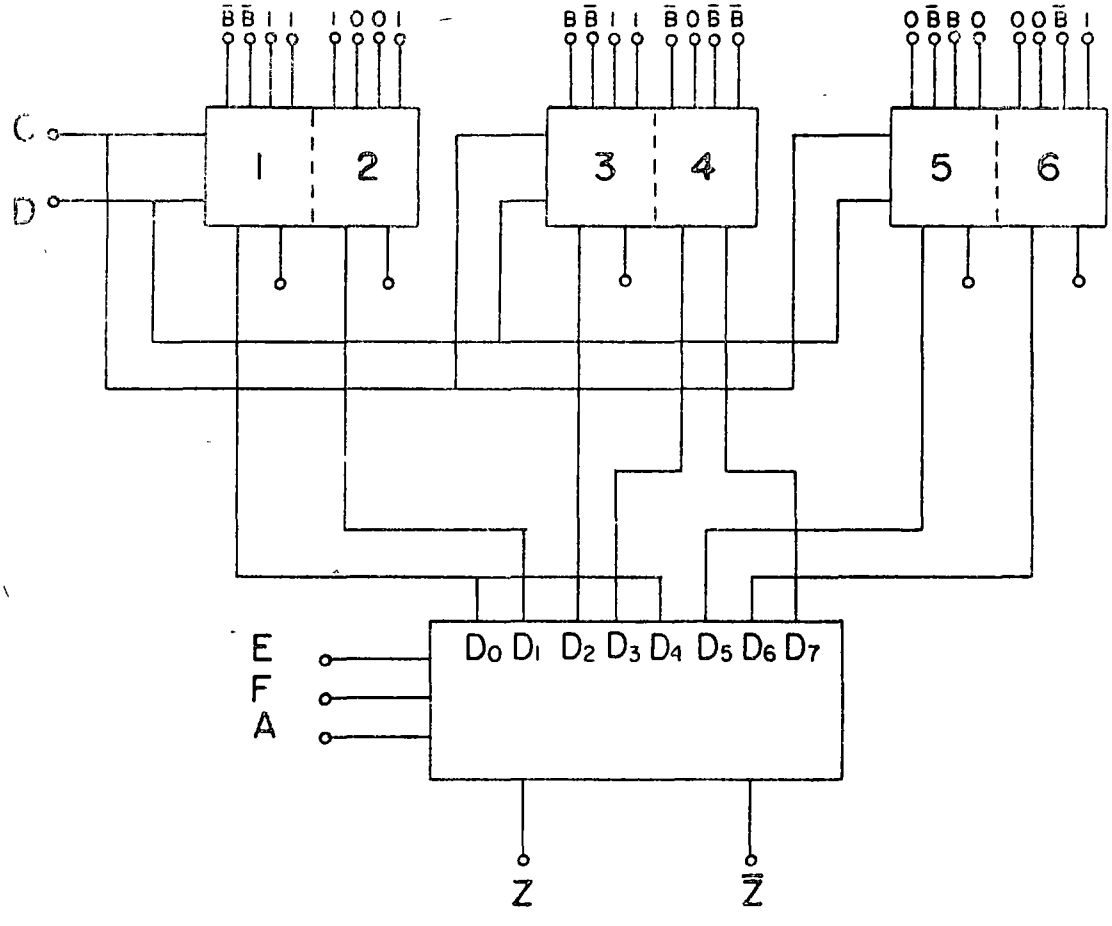
	C	D	B	E	F	A				
	0	0	0	0	0	0		CD		
	1	0	0	0	0	0	D_0	00	$\bar{B}$	
	0	1	0	0	0	0	D_1	01	$\bar{B}$	
D_0	1	0	1	0	0	0	D_2	10	1	MUX 1
	1	1	0	0	0	0	D_3	11	1	
	1	1	1	0	0	0				
	0	0	0	0	0	1	D_0	00	1	
	1	1	0	0	0	1	D_1	01	0	
D_1	0	0	1	0	0	1	D_2	10	0	MUX 2
	1	1	1	0	0	1	D_3	11	1	
	1	0	0	0	1	0				
	0	1	0	0	1	0	D_0	00	B	
	1	1	0	0	1	0	D_1	01	$\bar{B}$	
D_2	0	0	1	0	1	0	D_2	10	1	MUX 3
	1	0	1	0	1	0	D_3	11	1	
	1	1	1	0	1	0				
	0	0	0	0	1	1	D	00	$\bar{B}$	
	0	1	0	0	1	1	D	01	$\bar{B}$	
D_3	1	1	0	0	1	1	D	10	0	MUX 4
							D	11	$\bar{B}$	
	0	0	0	1	0	0				
	1	0	0	1	0	0	D_0	00	$\bar{B}$	
	0	1	0	1	0	0	D_1	01	B	
D_4	1	0	1	1	0	0	D_2	10	1	MUX 1 = MUX 5
	1	1	0	1	0	0	D_3	11	1	
	1	1	1	1	0	0				
D_5	0	1	0	1	0	1	D_0	00	0	D_2 10 B
	1	0	1	1	0	1	D_1	01	$\bar{B}$	D_3 11 D — MUX 6
	1	0	0	1	1	0	D_0	00	0	
D_6	1	1	0	1	1	0	D_1	01	0	
							D_2	10	$\bar{B}$	MUX 7
	1	1	1	1	1	0	D_3	11	1	
	1	0	0	1	1	1				
	0	0	1	1	1	1	D_0	00	B	
	1	0	1	1	1	1	D_1	01	B	
D_7	0	1	1	1	1	1	D_2	10	1	MUX 4 = MUX 8
	1	1	1	1	1	1	D_3	11	B	

Vemos entonces que con esta agrupación obtenemos un ahorro de 2 mux, ya que:

Mux 1 = Mux 5

Mux 8 = Mux 4

El circuito queda:



Criterios:

1) Utilizar señales multiplexadas que se empleen en más de una de las entradas del Mux de salida.

2) Seleccionar combinaciones que maximicen pares de funciones multiplexadas que sean complementos una de la otra.

Por último cabe mencionar que la misma función se podría realizar con 4 mux de 8 entrdas en el primer nivel y uno de 4 en el 2o. nivel. . Sin embargo, en el primer nivel conviene emplear mux más baratos y por lo tanto, para el ejemplo, los más convenientes son los de 4 entradas (vienen 2/ circuito integrado)

centro de educación continua
división de estudios superiores
facultad de ingeniería, unam

INTRODUCCION AL PROCESAMIENTO DIGITAL

CAPITULO 4: BIESTABLES Y REGISTROS DE CORRIMIENTO

Ing. Arturo González Hermosillo

Septiembre, 1977

Capítulo IV Biestables y Registros de Corrimiento.

IV.0 Introducción. Dentro de la electrónica digital, se utilizan diferentes tipos de elementos, de los más importantes en la división de los circuitos secuenciales, son los biestables. Un biestable es un elemento con capacidad de almacenar información; dicho de otra forma es un "elemento de memoria", no siendo estos los únicos con esta capacidad; entre otros, se pueden enunciar las cintas magnéticas, cintas de papel, tambores magnéticos, etc. pero por los requerimientos en el proceso de información, el grado de avance de la tecnología y sobre todo el estado actual en la microintegración de circuitos, los biestables tienen un especial interés. En la actualidad, existen circuitos integrados de no más de 2 cm^2 en cuyo interior se encuentra un arreglo de memoria con más de 4000 biestables; una computadora puede tener un porcentaje alto de biestables dentro de los elementos electrónicos que utiliza; el funcionamiento de un reloj electrónico de pulsera es en base de biestables. En general dentro de la electrónica digital, es dificil encontrar un sistema secuencial en donde su memoria no esté implementada con biestables.

Hay diferentes tipos de biestables; por su comportamiento en función de sus entradas de control reciben un nombre en especial. La forma de entender el comportamiento o el tipo de biestable, es a través de su tabla de transición. Esta tablade transición, nos determina el estado sucesor del biestable a un vector entrada.

IV.1 Biestable Básico. La forma más sencilla de pensar en la contrucción de un biestable, es con la realimentación lógica de dos inversores, suponiendo que el primer inversor se encuentra en un estado lógico $\bar{A}$; esto forzará a la salida del segundo a dar la entrada negada A, realimentandose al primero y así el arreglo permanecerá en un estado estable (figura 4-1).

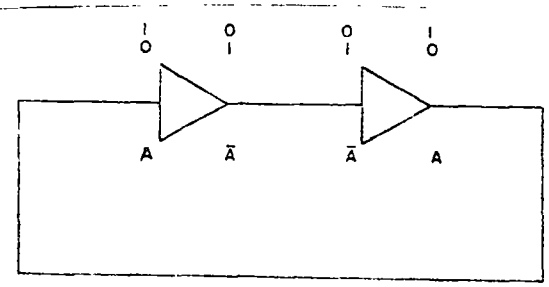


Figura (4-1)

Los inversores en su versión más sencilla es como lo muestra la Figura 4-2 de tal forma que un biestable básico queda implementado como se muestra en la figura 4-3.

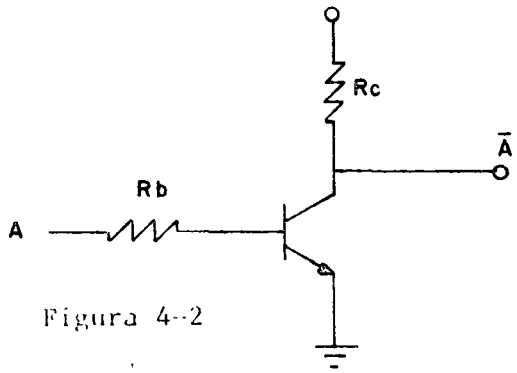


Figura 4-2

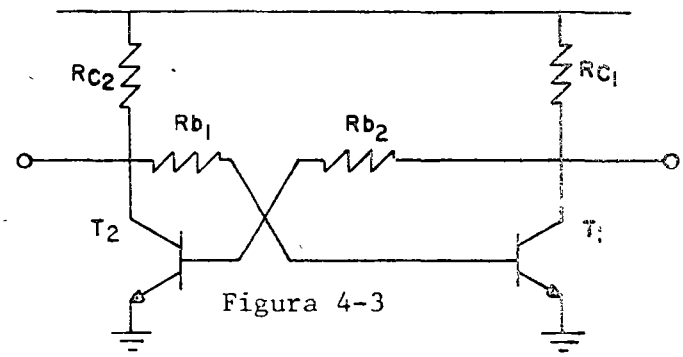


Figura 4-3

Como es sabido, la forma de sensar el estado del biestable es a través de niveles de voltaje. Para este caso, los valores de resistencia con relación a la beta mínima del transistor, deben de ser tales, que cuando un transistor se encuentre en corte, produzca la saturación del otro; entonces se debe asegurar que la corriente de base alimentada al transistor en saturación, sea mayor que la corriente requerida para su saturación. Estando un transistor en saturación, produce el corte en el otro, de tal forma que las salidas pueden ser directamente los colectores de los transistores, y si se encuentran trabajando en forma complementaria al llamar a un colector salida Q, el colector del otro transistor será $\bar{Q}$.

Las entradas de control para el circuito son como las mostradas en la figura 4-4, la acción que producen estas entradas con un voltaje alto en "S" o "R", es la saturación del transistor correspondiente.

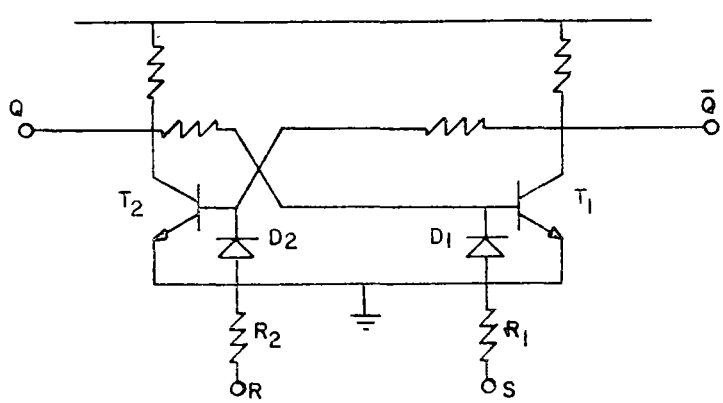


Figura 4-4

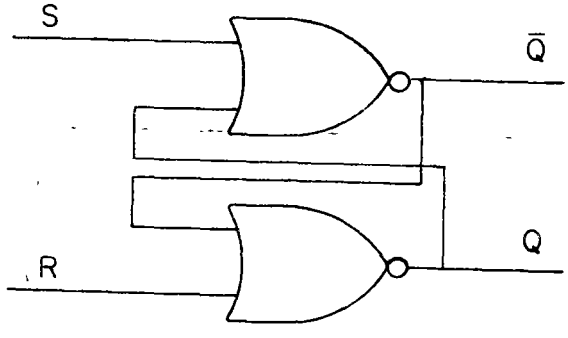
Si S es un voltaje alto, (V_H), D_1 conduce y satura T_1 ; al saturar T_1 corta a T_2 y la salida Q será alta mientras $\overline{Q}$ es baja, entonces al tener alta la entrada S produce la acción de poner al biestable alto ($Q=V_H$) y se le llama entrada "SET". Por simetría del circuito, la acción de R, para cuando se le aplica un voltaje alto será saturar a T_2 quedando Q baja y $\overline{Q}$ alta por lo que a esta entrada se le denomina "RESET".

Es obvio que no se podrán poner ambos transistores en saturación simultáneamente; esto implica que no podrán ser ambas entradas SET y RESET altas al mismo tiempo. De lo anterior se puede inferir la tabla de transisión de este biestable.

S	R	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	*

* No definido

IV.2 Latch. El Latch es un arreglo de dos compuertas realimentadas, que produce estados estables. De aquí que con la ayuda de algunas compuertas más sea utilizado como biestable.



Circuito Latch.

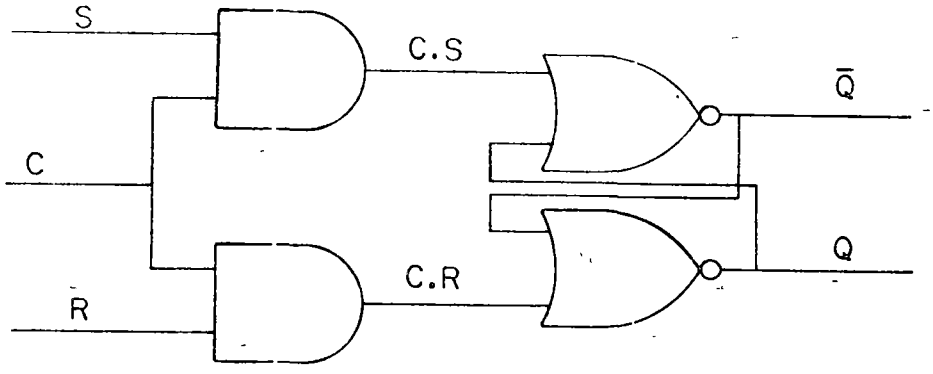
Figura 4-5

El análisis del funcionamiento es como sigue: Asumimos que la salida Q=0 y S=R=0, entonces la compuerta superior tendrá entradas Q=S=0; su salida será entonces $\overline{0+0}$ =1 y le llamaremos $\overline{Q}$. La compuerta inferior, tiene entradas $\overline{Q}$=1 y R=0; su salida será $\overline{0+1}$ =0 que es el estado donde

habíamos empezado.

Suponiendo que R cambia rápidamente a 1. La compuerta inferior tiene entonces entradas $\overline{Q}$=R=1 y su salida será Q= $\overline{\overline{Q+R}}$ =$\overline{1+1}$ =0 y permanece sin cambio, que es la condición de RESET. Sin embargo si S es la que cambia a 1, la salida de la compuerta superior será $\overline{Q}$= $\overline{Q+S}$ =$\overline{0+1}$ =0 y $\overline{Q}$ cambiará a cero. Como $\overline{Q}$ ha cambiado a cero, las entradas de la compuerta inferior son afectadas como sigue: $\overline{Q}$ =0, R=0, la salida Q será Q= $\overline{\overline{Q}+R}$ =$\overline{0+0}$ =1 cambiará a 1, condición de SET.

Biestable SR. A un latch, al agregársele dos compuertas AND como lo muestra la figura 6, se podrá tener una señal de sincronía para el cambio del latch, quedando un biestable S.R.



Biestable S.R.
Figura 4-6.

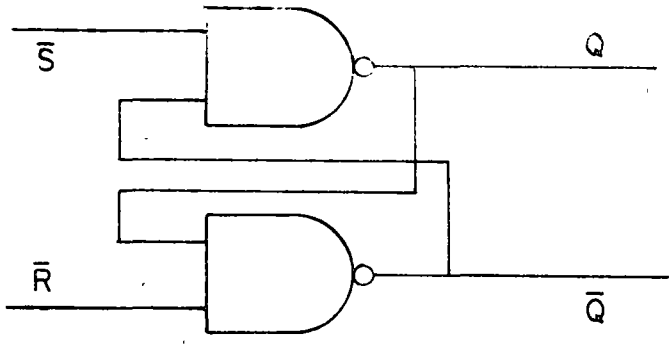
Para cuando C=0 no importa el estado de S y R, C.S=C.R=0 y el latch no cambia su estado; para cuando C=1 el valor de C.S y C.R está en función de S y R; el estado del latch será entonces dependiente de sus entradas. A la entrada C se le llama control de reloj; Q_{t+1} se entiende como el estado del biestables después de un pulso de reloj en C. La tabla de transición de este biestable será:

S	R	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	*

* No determinado

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

El mismo análisis de latch, se puede hacer con compuertas NAND quedando el arreglo y tabla de transición de la forma:



$\overline{S}$	$\overline{R}$	Q_{t+1}
0	0	*
0	1	1
1	0	0
1	1	Q_t

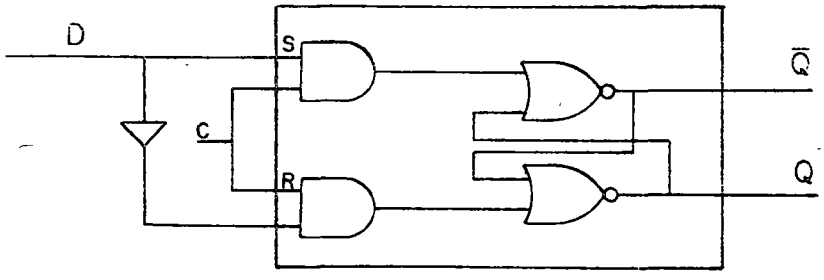
* No definido.

Figura 4-7

IV.3 Biestable D. El nombre de biestable D es tomado de "Delay", la lógica de este biestable es que la entrada que le sea presentada, será el sucesor de la salida Q del biestable; su tabla de transición será:

D	Q_{t+1}
0	0
1	1

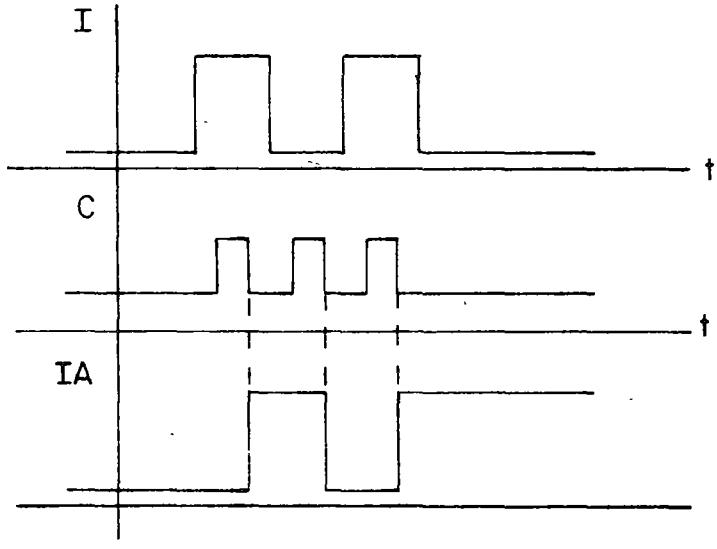
Observando la tabla de transición de un biestable SR, si trabajamos únicamente para cuando $S=\overline{R}$ podremos construir un biestable D.



Biestable D.
Figura 4-8.

De tal forma que si D=0 implica que S=0; R=1; la salida Q=0, Si D=1 implica que S=1, R=0 y la salida será Q=1, Luego entonces cumple con la tabla de transición propuesta.

IV.4 Principio de Maestro Esclavo. En ocaciones el conectar biestables en "cadena" como los que hasta ahora hemos visto, nos produce salidas no deseables. Un ejemplo de un efecto no deseado es el siguiente: Supóngase que se quiere construir un registro en donde se desea almacenar información binaria de 3 bits, se requiere que la forma de almacenarla sea en serie como lo muestra la siguiente carta de tiempos:



I es una señal digital que se encuentra variando en el tiempo; se sincroniza un reloj C que determina el tiempo donde I es estable y puede ser almacenada; IA es la información almacenada en los biestables. Si se construye un arreglo de biestables de los que se han estudiado como lo muestra la figura 4-9 para almacenar esta información, al momento de ser cierto C los tres biestables tomarán la información; esto se debe a que la salida de los biestables no está desacoplada de sus entradas.

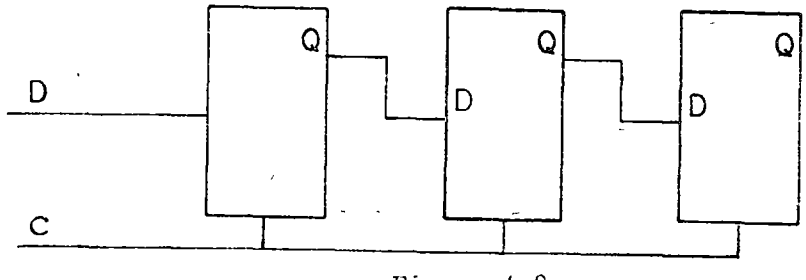


Figura 4-9.

Si se hace un desacoplamiento de las entradas de los biestables con sus salidas, de tal forma que las entradas censen la información a la subida del reloj y la pasen a la salida de los mismos en la bajada del reloj, tendremos:

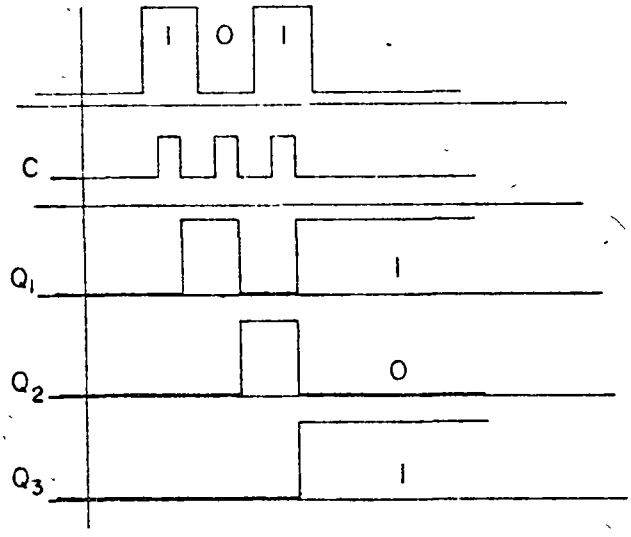


Figura 4-10

La información queda entonces almacenada en las salidas Q de los biestables.

La siguiente forma de desacoplar las salidas de las entradas se llama Maestro-esclavo y su funcionamiento es a través de dos latches como lo muestra la figura 4-11; el primero forma la sección maestro, y el segundo la sección esclavo.

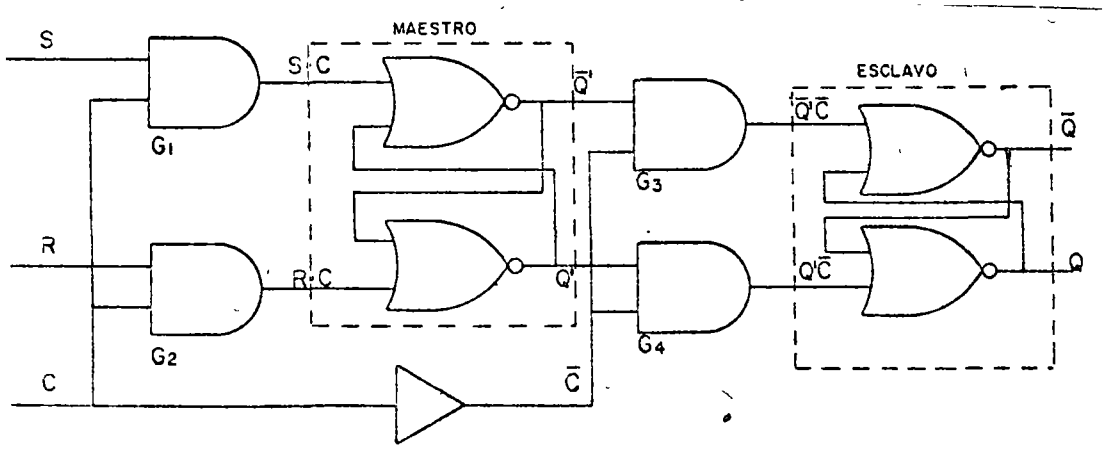


Figura 4-11

Para cuando C=0 las compuertas G_1 y G_2 se encuentran "abiertas"; S.C y R.C son ambas iguales a cero, haciendo que maestro no cambie de estado; G_3 y G_4 estarán "cerradas" y esclavo tomará la información de Maestro. En la transición de C =1, G_3 y G_4 se abren y $\overline{C}=0$; $\overline{Q}^1\ \overline{C}=Q^1\overline{C}=0\overline{Q}^1$ y esclavo almacena la información; C=1, S.C=S, R.C=R y maestro cambiará en función de S y R, al tiempo de que C cambia de uno a cero, maestro es "bloqueado" y "habilitado" esclavo; hasta este tiempo Q será función de S y R. El arreglo de la figura 10 nos representa un biestable "SET RESET MAESTRO ESCLAVO".

IV.5 Biestable Toggle. El comportamiento del biestable Toggle (T) es que en la transición del reloj, ya sea de 0 1 ó de 1 0, dependiendo de su implementación, la salida Q del biestable sea el negado del estado anterior; el funcionamiento es mostrado en el diagrama siguiente.

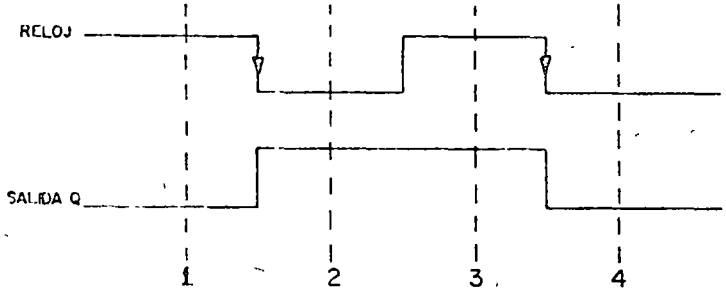


FIG. 11 a

Para nuestro análisis tomaremos al biestable que cambia en la bajada del reloj. El análisis del circuito se hace con los métodos convencionales. Primero del comportamiento del biestable construimos una tabla de estados; esta tabla nos da la información de los estados sucesores y los valores de Q.

	$\overline{C}$	C	Q
1	2	1	0
2	2	3	1
3	4	3	1
4	4	1	0

Haciendo la asignación de estados llegamos a la tabla de excitación del biestable.

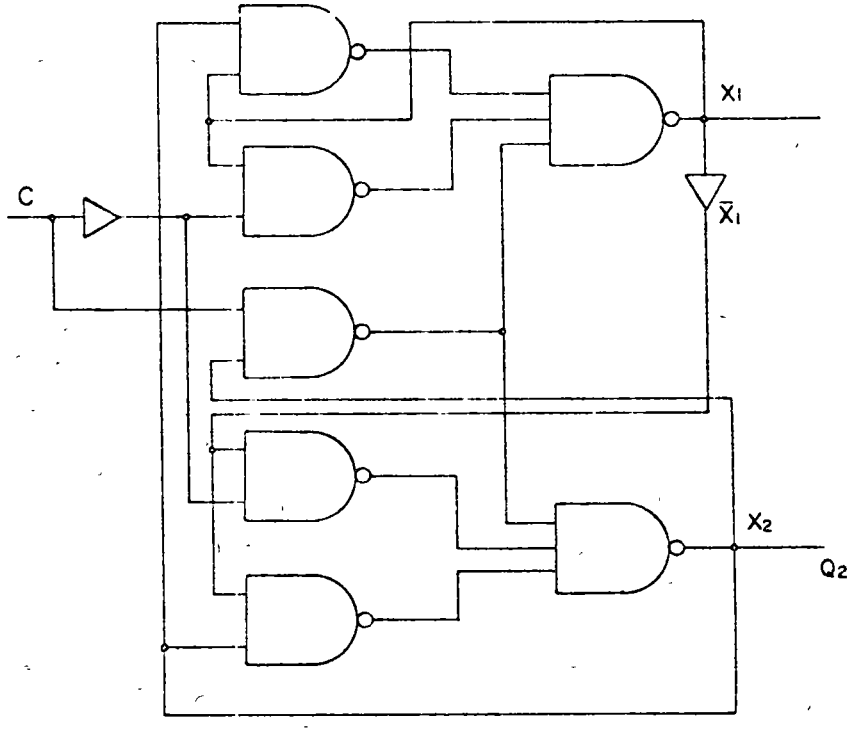
X_1X_2	C=0	C=1	Q
0	0	01	00	0
0	1	01	11	1
1	1	10	11	1
1	0	10	00	0

Resolviendo por Karnaugh.

$$X_1 = \bar{C}X_1 + CX_2 + X_1X_2$$

$$X_2 = \bar{C}\bar{X}_1 + CX_2 + \bar{X}_1X_2$$

Y el circuito correspondiente a las ecuaciones es mostrado en la figura 10.

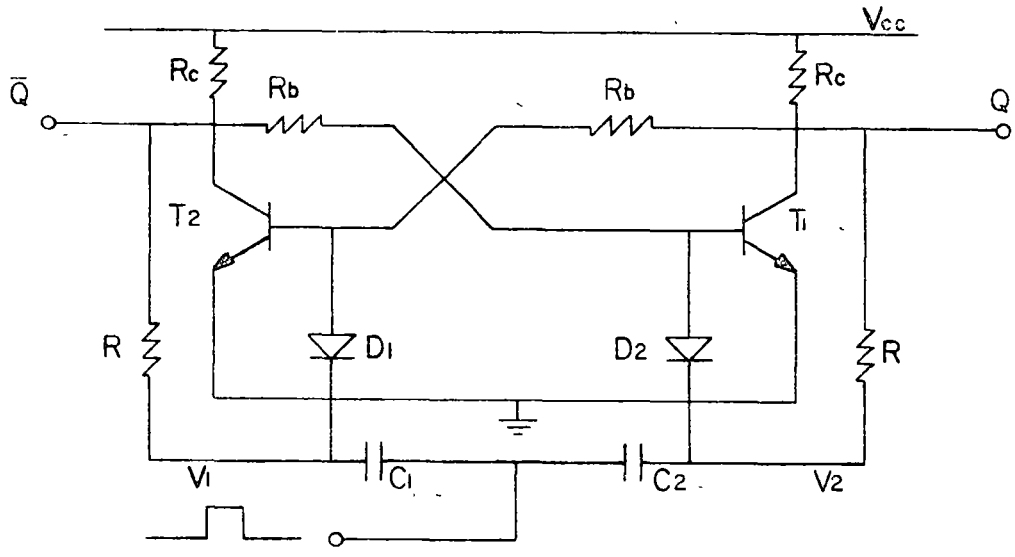


Biestable Toggle.
Figura 4-12.

Otra forma de implementar un biestable Toggle, es a partir de una celda básica; el circuito de disparo para este caso, trabaja con la diferenciación de la señal de sincronía o reloj.

Los pulsos obtenidos de esta diferenciación son negativos y actúan directamente sobre las bases de los transistores de la celda; en cada ciclo de reloj, se produce el pulso que corta al transistor en saturación,

saturándose el otro; al siguiente ciclo, el transistor que se encuentra en saturación, ahora es cortado y así sucesivamente, para cada ciclo de reloj.



Biestable Toggle.

Figura 4-13.

La figura 4-13 muestra una celda básica, con un circuito de disparo formado por D_1, D_2, C_1, C_2, R_1 y R_2, con este circuito la celda básica actúa como un biestable Toggle. La interacción de la celda y el circuito de disparo es del modo siguiente: Suponiendo que T_1 se encuentra en saturación, para el frente de onda positivo del reloj se tendrán pulsos positivos en V_1 y V_2, los díodos D_1 y D_2 se encontrarán entonces en circuito abierto y no habrá cambio en las condiciones del biestable; antes de producirse el frente de onda negativo del reloj, si T_1 está en saturación, el capacitor C_1 se encuentra cargado en forma positiva hacia la entrada del reloj, ya que por su otro extremo tendremos un voltaje casi igual a cero, que es el voltaje de colector de saturación de T_1; al momento de la transición o frente de onda negativo, por la carga de C_1 se tendrá un pulso negativo en V_2 que hace que conduzca D_2, y forza a la base de T_1 a un voltaje negativo, cortándose este transistor. Para el siguiente ciclo de reloj, y por simetría del circuito en forma recíproca pasará de corte a saturación T_1. Esta acción de T_1 se muestra gráficamente en la figura 4-14.

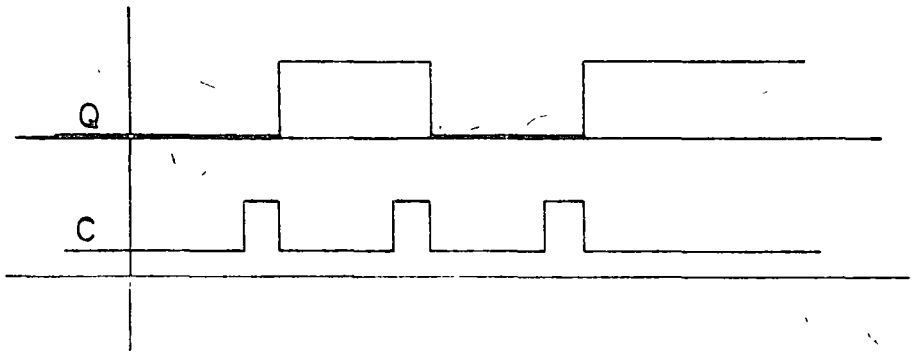


Figura 4-14

IV.6 Biestable J-K. Los biestables JK, posiblemente sean los más usados dentro de la familia de los biestables, es debido a que en sus dos entradas, J y K es posible conectarlas ambas a uno lógico, rompiéndose la ambigüedad para cuando se conectaban SET y RESET altas al mismo tiempo. Para cuando J=K=1, la transición del biestable es el negado de su estado anterior, la figura 4-15, presenta su tabla de transición.

J	K	Q_{t+1}
0	0	Q_t
0	1	0
1	0	1
1	1	$\bar{Q}_t$

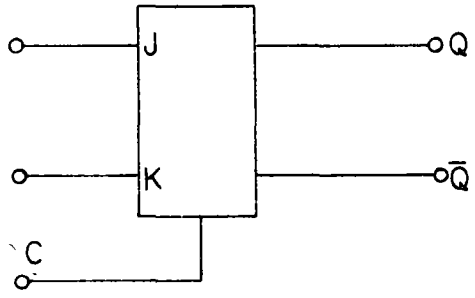


Figura 4-15

La construcción de este biestable, es partiendo de un biestable "SET" "RESET" maestro esclavo y permitiendo con lógica implementada, la combinación de ambas entradas altas, esto es mostrado en la figura 4-16.

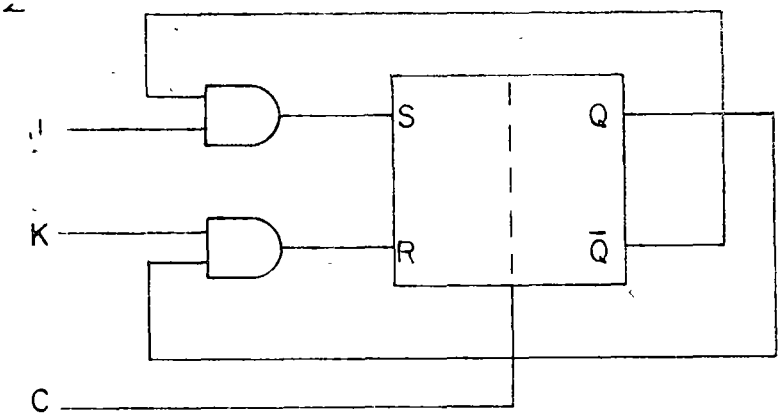


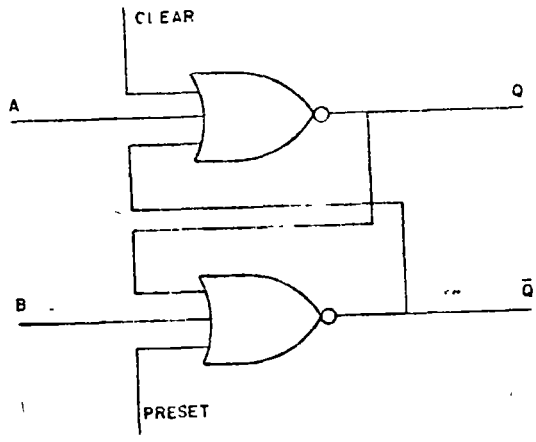
Figura 4-16

Se observa que $S=J\bar{Q}$; $R=KQ$ de tal forma que si se quiere que Q=Qn implica que S=0 y R=0 y si J=0 y K=0 entonces S=R=0 cumpliendo; si se quiere que Q=0 se debe de tener J=0, K=1, de donde no importando Q, S=0, y si Q=1, R=1 haciendo que Q=0 y si Q=0, R=0 y queda que $Qn+1=Q_n=0$ que también cumple; para hacer Q=1, las entradas deben de ser J=1, K=0 implica que R=0 y $S=\bar{Q}_n$ de tal forma que si $Q_n=1$, S=0 y $Q_{n+1}=Q_n=1$, y si $Q_n=0$, S=1 y $Q_{n+1}=1$ cumpliendo para ambos casos; Por último si J=K=1 tendremos que si $Q_n=0$, implica que R=0, S=1 y $Q_{n+1}=1=\bar{Q}_n$; si $Q_n=1$ implica que R=1, S=0 y $Q_{n+1}=0=\bar{Q}_n$ cumpliendo el último caso.

IV.6.1 Clear y Preset. En ocasiones, los biestables tienen una o dos entradas llamadas Clear y Preset.

La acción del Clear es fijar la salida Q del biestable a un cero lógico; la acción del Preset es fijar la salida Q al uno lógico, ambas entradas funcionan en modo asíncrono, esto es que son independientes del reloj. Por la acción que tienen estas entradas, es ovbio que no podrán ser activadas simultaneamente.

La implementación del Clear y el Preset, se hace directamente sobre el latch de salida del biestable (Figura 4-17) de tal forma que es independiente del tipo de biestable que se trate.



Para cuando Clear y Preset son ceros, el funcionamiento del latch no se ve afectado y la salida Q será función de A y B como ya se analizó; para cuando A = B = 0 y Clear es alto, se forza $Q = A+\bar{Q}+C = 0+\bar{Q}+1 = 0$ y $\bar{Q}=\overline{Q+B+P}= \overline{0+0+0} = 1$ siendo el efecto esperado del Clear; para cuando A= B= 0 y el Preset es alto, se tiene que $\bar{Q} = \overline{B+Q+P} = \overline{0+Q+1} = 0$ y $Q = \overline{C+B+\bar{Q}} = \overline{0+0+0} = 1$ siendo el efecto esperado del Preset. Como ya se dijo, A y B tienen que ser ceros, así que la implementación del biestable es tal que, el latch de salida se debe de encontrar cerrado para cuando el reloj está bajo.

IV.7 Registros de Corrimiento. Un registro de corrimiento es un arreglo de biestables conectados en cascada; la salida de los biestables, es conectada directamente a la entrada del siguiente. Suponiendo que se tiene un registro de n biestables, el biestable $i<n$; tendrá a su entrada la Qi-1 y a su salida la entrada i+1. Por características de la tecnología en circuitos integrados, los registros de corrimiento integrados tienen limitaciones en cuanto al número de puertos de acceso a la lógica interna del arreglo de biestables, esto es que el número de patas de un circuito integrado está limitado; por lo anterior, los registros en donde se desea accesar las salidas Q de todos los biestables, no podrán ser arreglos mayores de 8 biestables y en el caso donde se desea accesar a Qi y $\bar{Q}$i no podrá ser mayor a 4 biestables.

Para la implementación de registros de corrimiento en base a biestables integrados, el número de biestables conectados en cascada, prácticamente no tienen límite. Para este tipo de registros lo común es utilizar biestables J.K., conectados como lo muestra la figura 4-17.

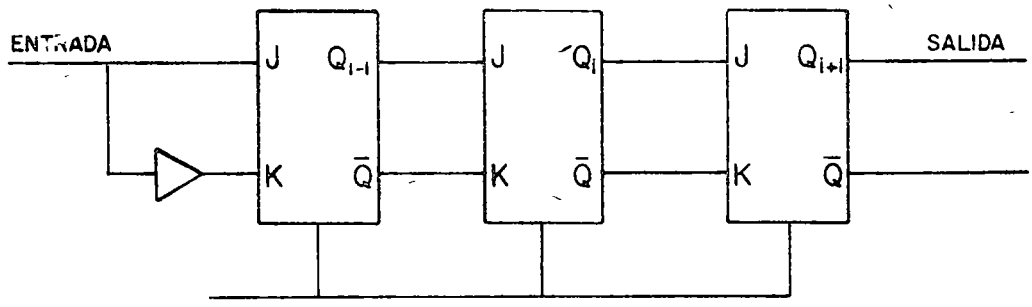


Figura 4-17

Como se observa de la figura, las entradas de los biestables están conectadas de tal forma que $J=\overline{K}$. Esto de la tabla de transición de los biestables implica que si J=1, Q_{n+1} = 1 y si J = 0, Q_{n+1} =0; entonces $J=Q_{n+1}$.

El funcionamiento es de la forma siguiente: Al tener los biestables el reloj común, cuando el reloj hace la tansición de cero a uno lógico, las secciones maestro de los biestables se abren y censan la información del biestable anterior; al estar el reloj alto, la sección esclavo de los biestables es cerrada y los biestables no producen cambio en sus salidas; al hacer el reloj la transición del uno al cero lógico, las secciones maestro de cada uno de los biestables se cierran y capturan el dato para cuando el reloj se encontraba alto; en este mismo tiempo, la sección esclavo se abre y toma el dato capturado por maestro, produciéndose que el dato que se encontraba en la salida Q_{i-1} del biestable i-1, fue censado y capturado por el biestable i. También se pueden implementar registros de corrimiento, con biestbles D y S.R. El funcionamiento es análogo para el caso cuando son implementados con biestables J.K., de la tabla de transición del biestable D, se sabe que Q_{n+1} =D de tal forma que con biestables D ahora queda el circuito implementado como en la figura 4-18.

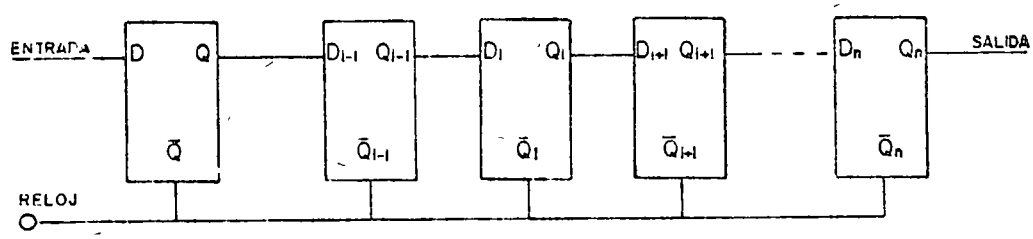


Figura 4-18.

Para la implementación con biestables S y R, la figura 4-19 muestra la interconexión haciendo trabajar al biestable i con $S = \bar{R}$, y si S =1; R=0 implica que $Q_{n+1}=1$, y si S = 0 R=1 implica que $Q_{n+1}=0$, concluyendo que $Q_{n+1}=S$.

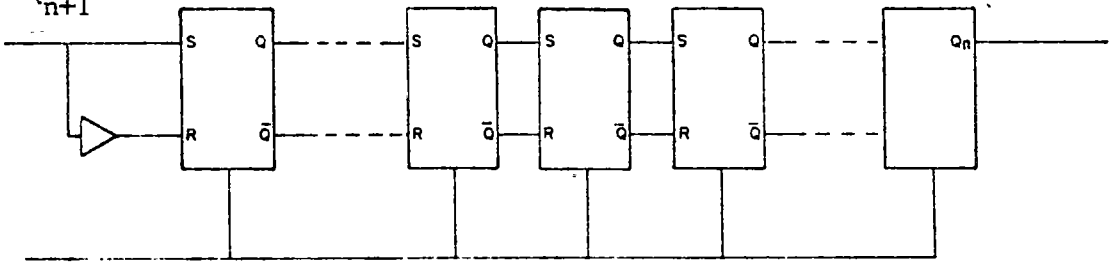


Figura 4-19.

Un registro de corrimiento puede ser cargado en paralelo y posteriormente hacerse el corrimiento. Las señales por medio de las cuales se hace la carga en paralelo, son el "Clear" y el "Preset". Estas señales fijan un cero y un uno lógico respectivamente en la salida del biestable. La figura 4-20 muestra un registro con carga en paralelo.

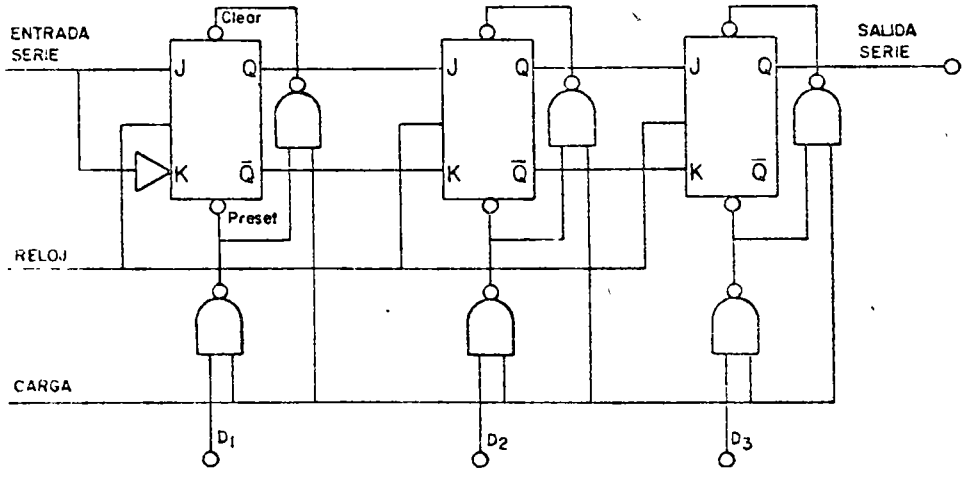


Figura 4-20.

Las funciones lógicas para el Clear y Preset son diseñadas en la siguiente forma: Se requiere que para cuando carga sea igual a cero, el Clear y Preset sean altos para que el biestable opere en modo síncrono y hacer el corrimiento; para cuando carga es alto, el dato debe de ser cargado en el biestable correspondiente, activándose el Clear o el Preset del mismo. La lógica anterior se interpreta de la siguiente tabla de verdad.

Carga	Dato	Clear	Preset
0	0	1	1
0	1	1	1
1	0	0	1
1	1	1	0

de la tabla de verdad se observa que para el Preset,

$$\overline{P} = C \circ D$$

$$P = \overline{C \circ D}$$

para el Clear,

$$\overline{C} = C\overline{D}$$

$$C = \overline{C\overline{D}}$$

$$C = \overline{C} \circ (\overline{C}+\overline{D})$$

$$C = C \circ \overline{C\overline{D}}$$

Un arreglo de biestables junto con lógica combinacional, puede dar la implementación de un registro de corrimiento en donde el corrimiento sea a la derecha o izquierda, a elección de una señal de modo de operación (figura 4-21).

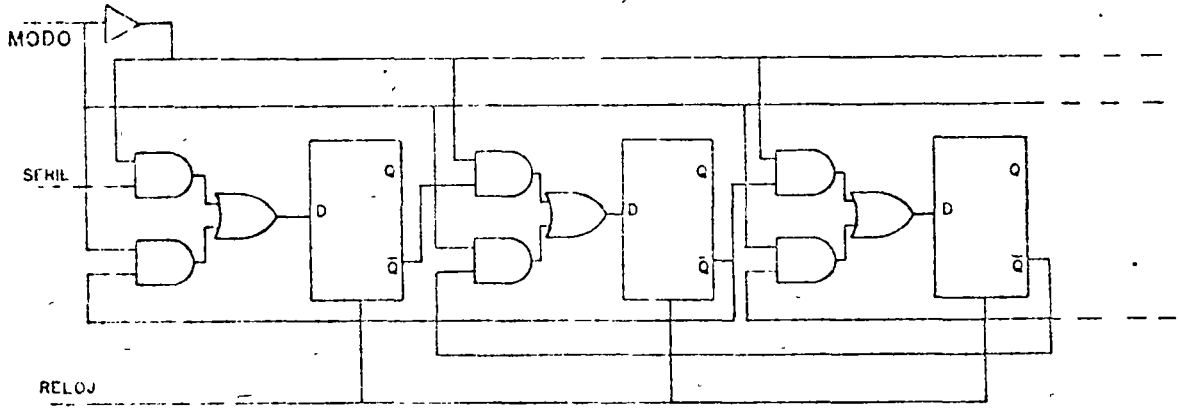


Figura 4-21.
Registro de corrimiento
a la izquierda y derecha.

El diseño de la función lógica para la entrada D_n es; Si el modo es alto, el corrimiento se hace hacia la izquierda, y si el modo es bajo el corrimiento se hace a la derecha. La forma de operación de este Registro se representa en la siguiente tabla.

Modo	Q_{n-1}	Q_{n-1}	Q_n
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

de la tabla se obtiene la implementación de D_n con mapas de Karnaught.

	Q_{n+1} Q_{n-1}			
Modo	00	01	11	10
0	0	1	1	0
1	0	0	1	1

$$\overline{D} = M\,Q_{n+1} + \overline{M}\,\overline{Q_{n-1}}$$

$$D = \overline{M\,Q_{n+1} + \overline{M}\,\overline{Q_{n-1}}}$$

Bibliografía.

1. Texas Instruments Incorporated
   Designing with Integrated Circuits
   Mc Graw-Hill
2. Kohonen Teuvo
   Digital Circuits and Devices
   Prentice-Hall, Inc.

3. H. Troy Nagle Jr.
   B.D. Caroll
   J. David Irwin
   An Introduction to Computer Logic
   Prentice Hall

centro de educación continua
división de estudios superiores
facultad de ingeniería, unam

INTRODUCCION AL PROCESAMIENTO DIGITAL

CAPITULO 5: DISEÑO DE CIRCUITOS SECUENCIALES

Ing. Mario Rodríguez Manzanera

Septiembre, 1977

V Diseño de Circuitos Secuenciales:

V.1 Introducción: Una de las partes de mayor importancia en el diseño lógico es la correspondiente a la realización de circuitos secuenciales.

V1.1 Definición: Cuando en un circuito el valor presente en sus salidas (z), no es dependiente únicamente del valor presente en sus entradas (X) sino que lo es tambien de la historia del circuito (Q), este será denominado 'secuencial'.

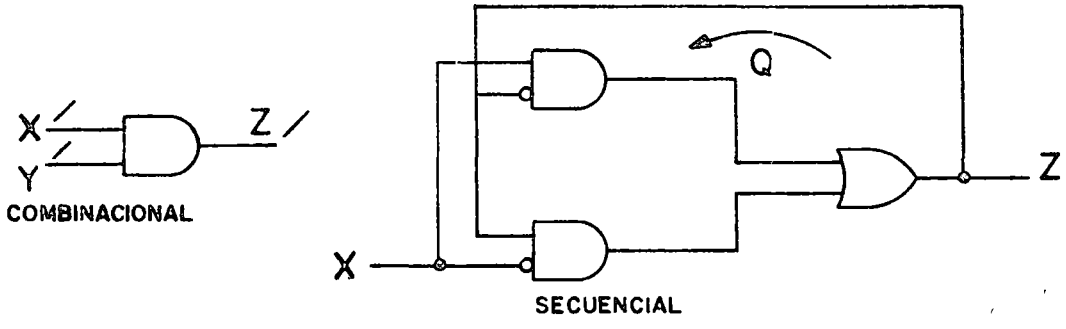


Figura V.1

VI.2 Estado: Es una sencilla representación de la historia de un circuito, la cual puede considerarse como una secuencia de estados. Un estado es una propiedad inherente al circuito.

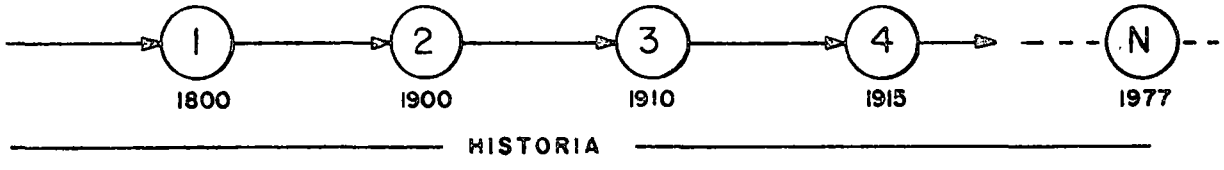


Figura V.II

VI.3 Circuitos secuenciales síncronos y asíncronos.

Un circuito secuencial es síncrono cuando sus cambios de estado pueden llevarse a cabo únicamente en coincidencia con una señal de reloj es decir en sincronía con este.

Un circuito secuencial asíncrono es aquel que no requiere de una señal de reloj fija para cambiar de estado.

V.2 Circuitos secuenciales síncronos.

V2.1 Representación de circuitos secuenciales:

Para plantear un problema el diseño lógico se auxilia de la teoría de autómatas y de la teoría de gráficas con el fin de facilitar su solución.

V2.1.1 Representación gráfica o diagrama de flujo.

En la representación gráfica se utilizarán los elementos de teoría de gráficas siendo estos: círculos denominados nodos, Flechas denominadas ramas o trayectorias.

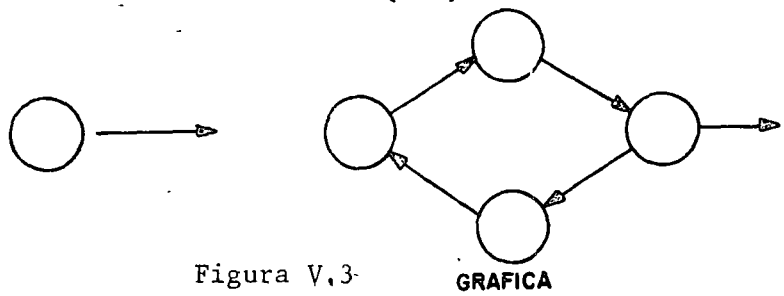


Figura V.3

Cada nodo representará un estado y será etiquetado con el nombre de este, por ejemplo los estados A, B, C

Cada rama será etiquetada con el nombre de la entrada al circuito que afecta a un cierto estado sobre la salida del circuito al realizarse el paso de un nodo a otro, es decir:

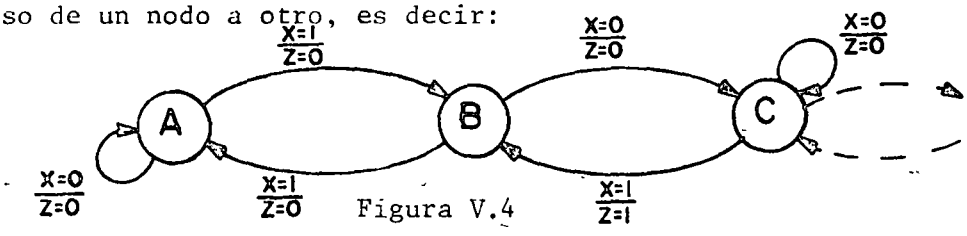


Figura V.4

En la figura V.4 se tienen todos los elementos para definir un autómata esto es, se tiene:

Un conjunto de estados $\{A, B, C,..\} = \{Q\} = \{q_1, q_2, ..\}$

Un conjunto de entradas $\{X_1 ... \} = \{X\} = \{x_1, x_2, ..\}$

Un conjunto de salidas $\{Z_1 ... \} = \{Z\} = \{z_1, z_2, ..\}$

y dos funciones, una llamada δ y otra λ.

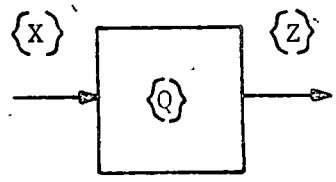
δ es la función de transición y se define como:

$$\delta(q,x) = q^{t+1}$$

y λ la función de salida:

$$\lambda(q,x) = Z$$

Así en la figura V.4.



$Q = A,B,C$

$X = X$

$Z = Z$

$\delta(A,x=o) = A \qquad \lambda(A,x=o) = 0$

$\delta(A,x=1) = B \qquad \lambda(A,x-1) = 0$

$\delta(B,x=1) = A \qquad \lambda(B,x=o) = 0$

$\delta(B,x=1) = C \qquad \lambda(B,x=o) = 0$

$\delta(C,x=o) = C \qquad \lambda(C,x=o) = 0$

$\delta(C,x=1) = B \qquad \lambda(C,x=1) = 1$

Entonces el siguiente estado de B al tener una entrada X=0 será el estado C y el circuito produce una salida Z=0.

El empleo de esta representación facilitará grandemente el planteamiento de cualquier problema más no así su solución. Con este fin, se introducen las tablas de transición.

V.2.1.2. Tablas de Transición. Otra forma de representación de un autómata son las tablas de transición en donde cada uno de los elementos anteriormente introducidos se incluyen.

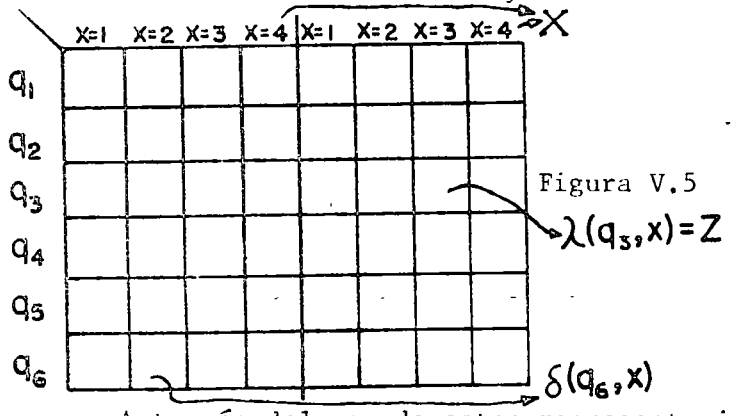


Figura V.5

	X=0	X=1	X=0	X=1
A	A	B	0	0
B	C	A	0	0
C	C	B	0	1
	$\delta(q,x)$		$\lambda(q,x)$	

$\delta(B,x=0)=C$

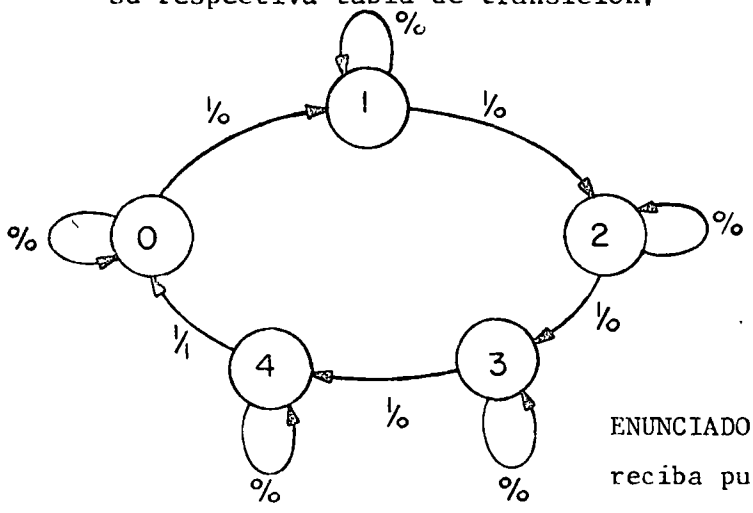
$\lambda(B,x=0)=0$

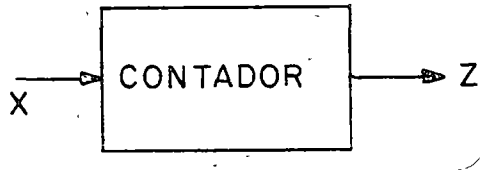
A través del uso de estas representaciones el diseñador puede en forma sencilla plantear los problemas de diseño en circuitos secuenciales.

V.3. Ejemplos de Diseño: A continuación se incluyen diversos diseños

para circuitos necesarios en múltiples aplicaciones.

V.3.1. Contadores. El número estados de un contador está limitado por la base de este; así, para un contador base 5 serán necesarios 5 estados. En la Figura V.6 se presenta el diagrama de flujo para este contador y su respectiva tabla de transición.





ENUNCIADO: Realizar un circuito el cual reciba pulsos por una entrada X y produzca una salida Z cada 5 pulsos.

Q	X=0	X=1	X=0	X=1
$q_0 = 0$	0	1	0	0
$q_1 = 1$	1	2	0	0
$q_2 = 2$	2	3	0	0
$q_3 = 3$	3	4	0	0
$q_4 = 4$	4	0	0	1
		δ		λ

Z

Figura V.6

V.3.1.2. Asignación de estados: El problema de la asignación de estados corresponde a etiquetar cada estado con un número representado en Binario.

ABC	X=0	X=1	X=0	X=1
000	000	001	0	0
001	001	010	0	0
010	010	011	0	0
011	011	100	0	0
100	100	000	0	1

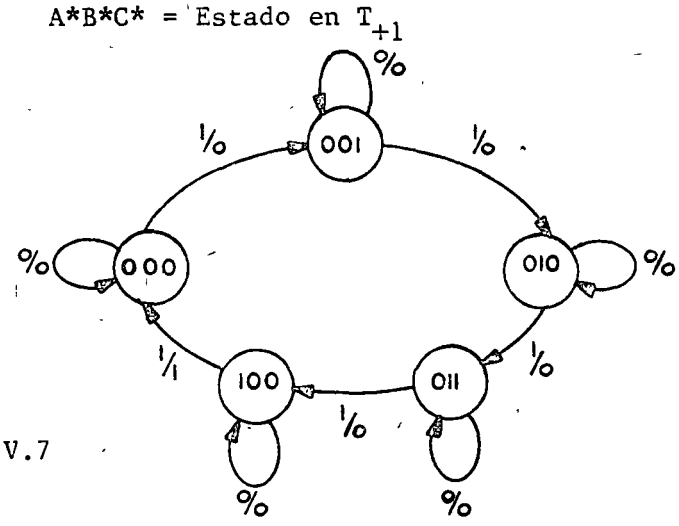


Figura V.7

Habiendo asignado un valor a cada estado en la tabla de transición se ha dado un nombre a cada columna teniéndose C,B,A.

V.3.1.3. Tabla de Verdad. A partir de la tabla de transiciones será posible generar la tabla de verdad de las variables A,B y C.

Procedimiento:

Coloque las entradas como columnas (X) a continuación las variables (ABC) y sus combinaciones como se muestra en la Figura V.7; coloque una línea vertical y vierta las funciones δyλ.

	XABC	A*B*C*	Z
	0 000	000	0
	0 001	001	0
	0 010	010	0
	0 011	011	0
	0 100	100	0
	0 101	***	*
No definido	0 110	***	*
	0 111	***	*
	1 000	001	0
	1 001	010	0
	1 010	011	0
	1 011	100	0
	1 100	000	1

Figura V.7.

V.3.1.4. Reducción: Una vez planteada la tabla de verdad el problema se reduce a obtener las funciones estrella o asterisco A*, B* y C*, para lo cual utilizamos los mapas de Karnaugh, (Figura V.8.).

XA \ BC	00	01	11	10
00	0	0	0	0
01	1	*	*	*
11	0	*	*	*
10	0	0	1	0

$$A^* = XA + X\,BC = A(X) + XBC$$

XA \ BC	00	01	11	10
00	0	0	1	1
01	0	*	*	*
11	0	*	*	*
10	0	1	0	1

$$B^* = X\,B\,C + XB + BC = B(X + C) + B\,(XC)$$

XA \ BC	00	01	11	10
00	0	1	1	0
01	0	*	*	*
11	0	*	*	*
10	1	0	0	1

$$C^* = \overline{X}C + X\,\overline{BC} + XAC = C\,(\overline{X}) + C\,(XB + \overline{XA})$$

V.3.1.5. Funciones. Una vez encontradas las funciones estrella tenemos:

$$A^* = A(\overline{X}) + XBC$$

$$B^* = B(X+C) + B(XC)$$

$$C^* = C(X) + C\,(XB + XA)$$

y podemos observar que en las funciones B* y C* se tiene el valor real y el negado de la variable sin estrella; para lograr que A* tenga la misma forma utilizamos el siguiente artificio:

$$A^* = A(\overline{X}) + XBC\ (A + \overline{A})$$

De donde:

$$A^* = A(\overline{X}+XBC) + \overline{A}\ (XBC)$$

Y así:

$$A^* = A(\overline{X} + BC) + \overline{A}\ (XBC)$$

Quedando la función con su real y su negado resumendo:

$$A^* = A(\overline{X}+BC) + \overline{A}\ (XBC)$$
$$B^* = B(\overline{X}+\overline{C}) + \overline{B}\ (XC)$$
$$C^* = C(\overline{X}) + \overline{C}\ (XB + X\overline{A})$$

V.3.1.6. Implementación. Al observar la forma de las funciones encontradas recordamos la clásica representación de un Flip-Flop que en función de Q nos queda:

$$Q^* = g_1\ Q + \overline{g}_2\overline{Q}$$

y para el Flip-Flop JK:

$$Q^* = J\overline{Q} + \overline{K}Q$$

De donde para la variable A* se tiene:

$$Q^* = \overline{Q}\ J + \overline{K}$$
$$A^* = \overline{A}\ (XBC) + A\ (\overline{X}+BC)$$

De donde:

$$J_a = XBC \quad ; \quad K_a = \overline{\overline{X}+BC}$$

Y para B y C:

$$J_b = XC \quad ; \quad K_b = \overline{\overline{X}+\overline{C}}$$
$$J_c = XB+\overline{A} \quad ; \quad K_c = \overline{X}$$

Obteniéndose las ecuaciones de diseño:

$$J_a = XBC \quad K_a = X(\overline{B}+\overline{C})$$
$$J_b = XC \quad K_b = XC \quad \therefore\ J_b = K_b = XC$$
$$J_c = X(\overline{A}+B) \quad K_c = X$$

De aquí es posible realizar el diagrama lógico:

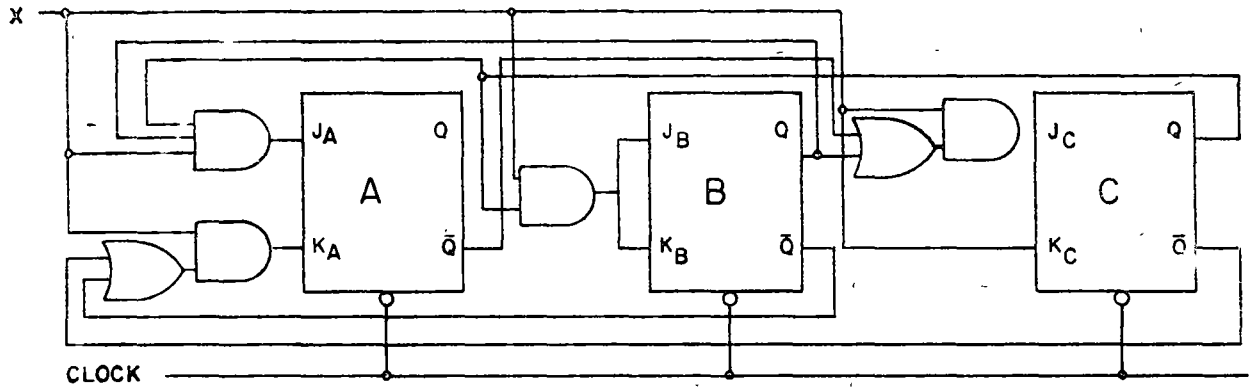


Figura V9

V.3.1.7. Función de Salida. Resta por el momento obtener la función de salida Z, utilizando el mismo procedimiento tenemos:

XA \ BC	00	01	11	10
00	0	0	0	0
01	0	X	X	X
11	1	X	X	X
10	0	0	0	0

Z=XA

esto es:

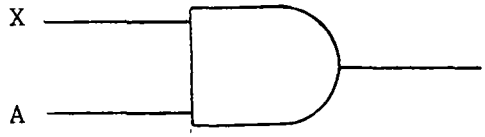


Figura V.10

Rearreglando el circuito y agregando la salida tenemos:

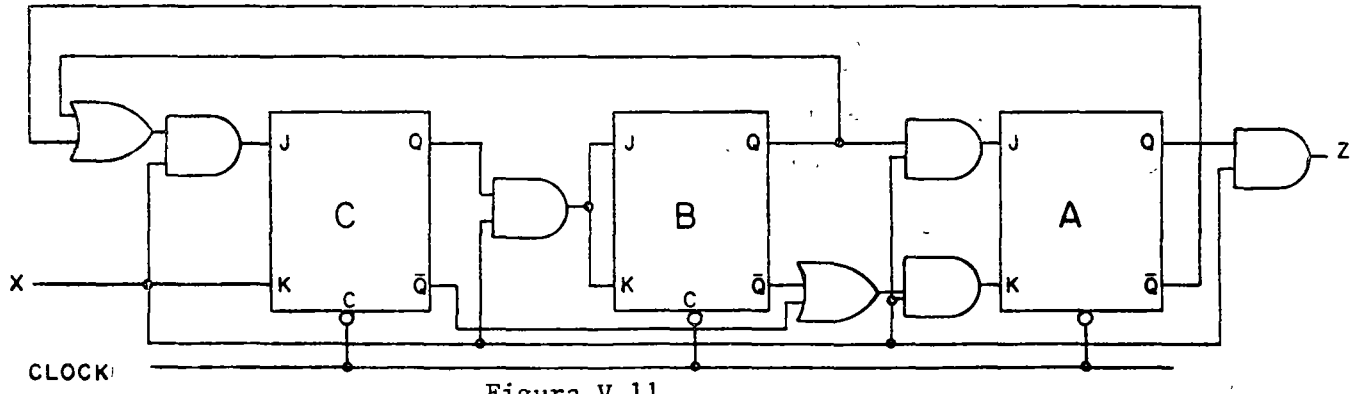


Figura V.11

V.3.1.8. Posibles Reducciones. Cuando en el diseño no enteresa que debe suceder al circuito cuando la entrada X es cero, es posible realizar una reducción en el diseño anterior.

es decir,

ABC	X=1	X=1
000	001	0
001	010	0
010	011	0
011	100	0
100	000	1

Figura V.12

de donde:

X ABC	A*B*C*
0 000	X X X
0 001	X X X
0 010	X X X
0 011	X X X
0 100	X X X
0 101	X X X
0 110	X X X
0 111	X X X
1 000	0 0 1
1 001	0 1 0
1 010	0 1 1
1 011	1 0 0
1 100	0 0 0
1 101	X X X
1 110	X X X
1 111	X X X

XA \ BC	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	0	X	X	X
10	0	0	1	0

$A^* = BCX$

00	01	11	10
X	X	X	X
X	X	X	X
0	X	X	X
0	1	0	1

$B = (\bar{B}C + B\bar{C})X$

XA \ BC	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	0	X	X	X
10	1	0	0	1

$C=\overline{C}AX$

En la misma forma:

$A^*= BC\ (A+\overline{A}) = A(BC_x) + \overline{A}(BC_x) \quad \therefore\ J_a=BC_x \quad K_a=\overline{BCX}$

$B^*= \longrightarrow B(C_{\overline{x}}) + (\overline{B}C_x) \quad \therefore\ J_b=C_x \quad K_b=C_x$

$C^*= \overline{C}(\overline{A}) = 0+\overline{C}(\overline{A}_x) \quad \therefore\ J_c=A_x \quad K_c=\overline{0}=1$

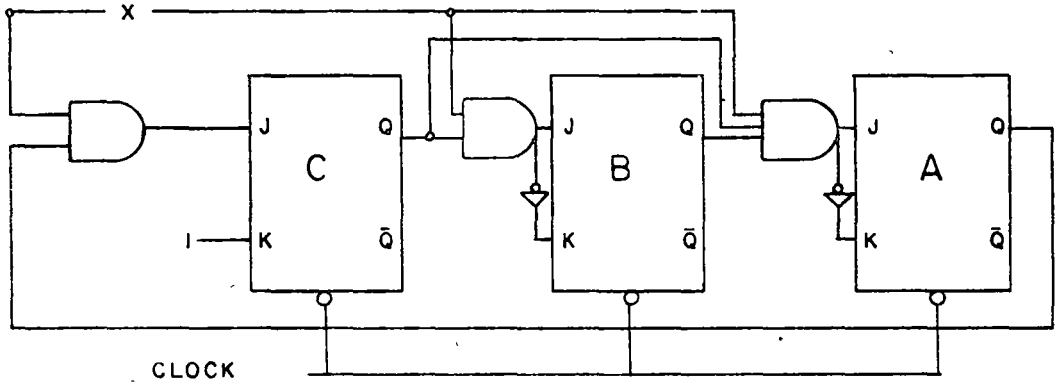


Figura V-14

Si se desea tener un contador dependiente únicamente de la señal de reloj:

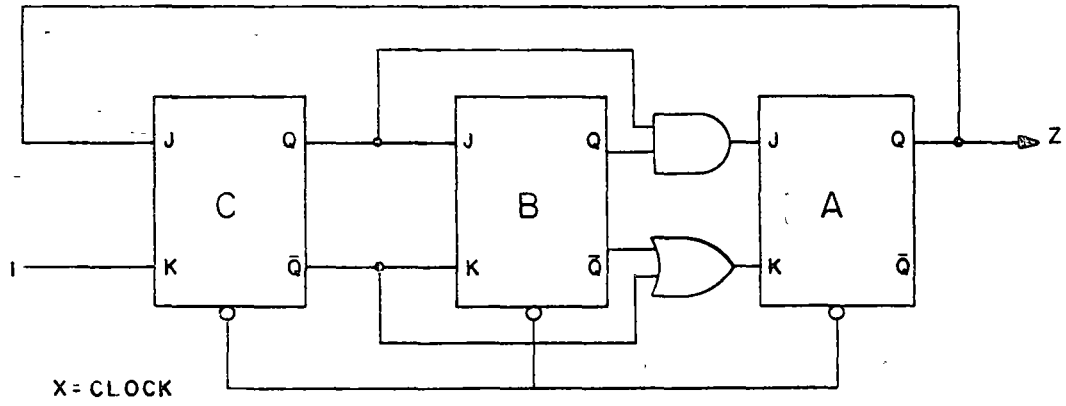
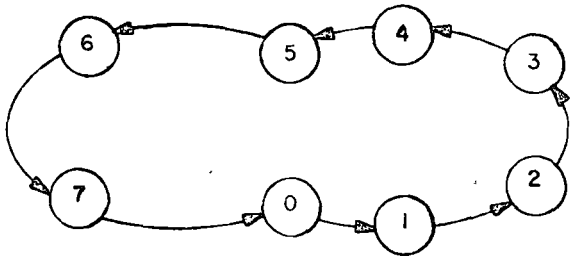


Figura V-15

V.3.2. Ejemplo. Diseñar un contador base 8 utilizando FFT, JK.

Diagrama:



X ABC	A*	B*	C*	Z
0 000	0	0	0	0
0 001	0	0	1	0
0 010	0	1	0	0
0 011	0	1	1	0
0 100	1	0	0	0
0 101	1	0	1	0
0 110	1	1	0	0
0 111	1	1	1	0
1 000	0	0	1	0
1 001	0	1	0	0
1 010	0	1	1	0
1 011	1	0	0	0
1 100	1	0	1	0
1 101	1	1	0	0
1 110	1	1	1	0
1 111	0	0	0	1

BC

XA	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	1	1	0	1
10	0	0	1	0

$A^* = \overline{X}A + A\overline{B} + A\overline{C} + x\overline{A}BC$

$A^* = A(\overline{x}+\overline{B}+\overline{C}) + \overline{A}\ (XBC)$

BC

XA	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	0	1	0	1
10	0	1	0	1

$B^* = B\overline{C} + B\overline{X} + BC_x$

$B^* = B(\overline{c} + \overline{x}) + \overline{B}(CX)$

Figura V-16

BC

XA	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	1	0	0	1
10	1	0	0	1

$C^* = \overline{X}C + X\overline{C}$

BC

XA	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	1	0
10	0	0	0	0

$Z = XABC$

De las ecuaciones: $Q^* = J\bar{Q} + \bar{K}Q$

$J_a = XBC \quad K_a = \overline{(x+B+C)} = xBC \quad J_a = K_a \rightarrow T_a$

$J_b = XC \quad K_b = \overline{(c+x)} = XC \quad J_b = K_b \rightarrow T_b$

$J_c = X \quad K_c = \bar{X} = X \quad J_c = K_c \rightarrow T_c$

Solución:

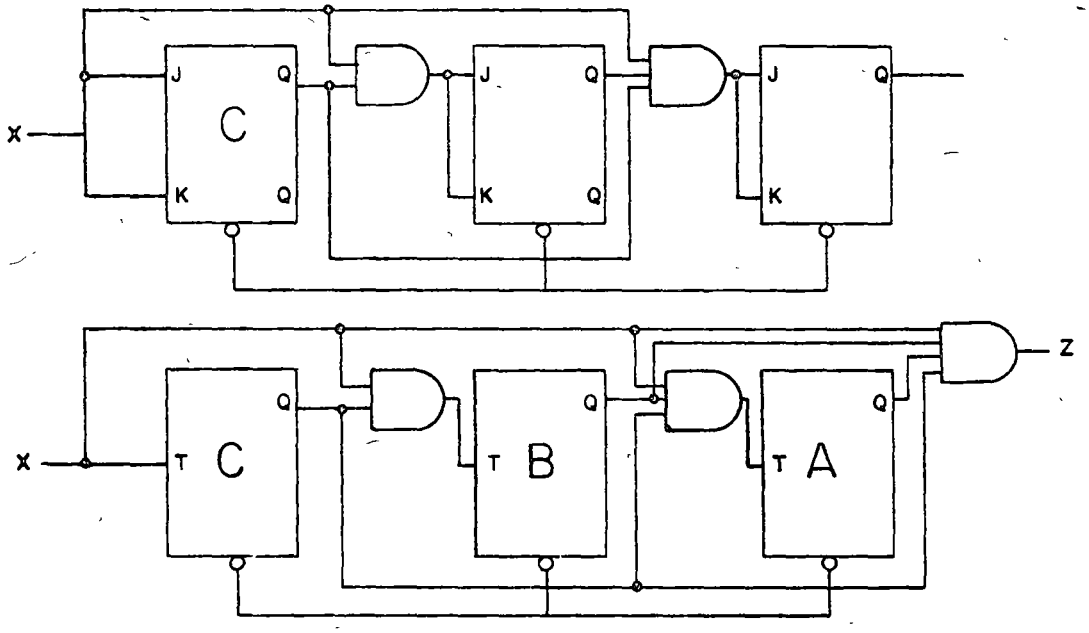


Figura V-17.

Si ahora sólo tomamos los estados iniciales como no importantes tenemos:

$A^* = A(\bar{B}+\bar{C}) + \bar{A}BC \quad \therefore \quad J_a = K_a = T_a = B_c$

$B^* = BC + \bar{B}C \quad \therefore \quad J_b = K_b = T_b = C$

$C^* = C\bar{x} + \bar{C}x \quad \therefore \quad J_c = K_c = T_c = x$

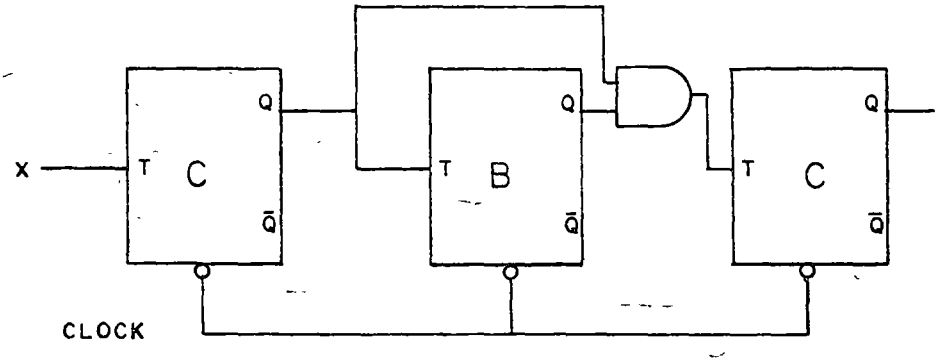


Figura V-18

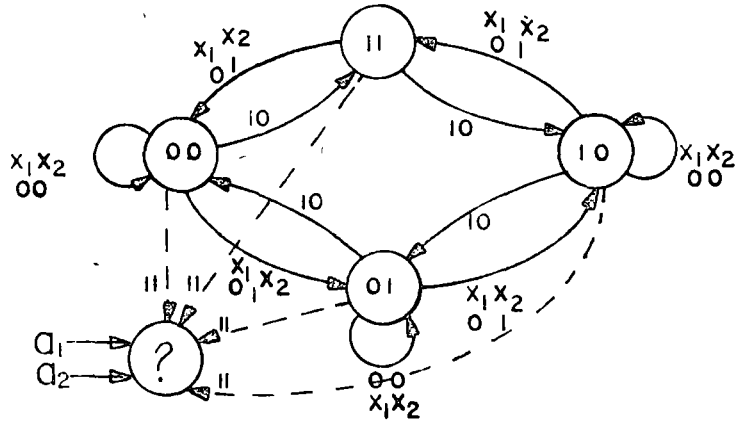
V.32. Contadores ascendentes descendentes con cualquier secuencia y programables.

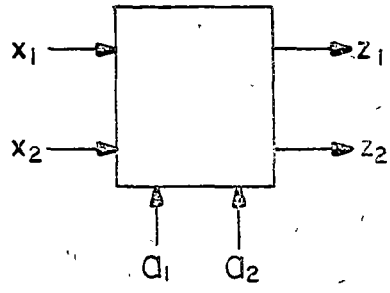
V.3.2.1. Contadores ascendentes y descendentes. Hasta ahora el diseño ha tomado únicamente en consideración contadores que incrementan su secuencia es decir: 0→1→2→3→4→5→0

Para construir un contador de secuencia descendente bastará con tomar de un diseño ya construído las salidas Q. En donde se encuentra el complementao del diseño es decir: 15-14→13→12→11→15 respectivamente.

V.3.2.2. En la práctica muchas veces es indispensable fijar el número desde el cual se requiere efectuar una cuenta ascendente o descendente. Para esto se utiliza un contador programable en donde bajo ciertas señales de control, se decide cuando se está programando, cuando sube o cuando baja la cuenta.

V.3.2.3. Ejemplo: Diseño de un contador programable base 4 que cuente hacia arriba y hacia abajo.





	X_1X_2	X_1X_2	X_1X_2	X_1X_2
A B	0 0	0 1	1 0	1 1
0 0	0 0	0 1	1 1	A_1A_2
0 1	0 1	1 0	0 0	A_1A_2
1 0	1 0	1 1	0 1	A_1A_2
1 1	1 1	0 0	1 0	A_1A_2

Programa

Cuenta abajo

Cuenta arriba

Permanece

Figura V.19.

X_1	X_2	A	B	A*B*
0	0	0	0	0 0
0	0	0	1	0 1
0	0	1	0	1 0
0	0	1	1	1 1
0	1	0	0	0 1
0	1	0	1	1 0
0	1	1	0	1 1
0	1	1	1	0 0
1	0	0	0	1 1
1	0	0	1	0 0
1	0	1	0	0 1
1	0	1	1	1 0
1	1	0	0	A_1A_2
1	1	0	1	A_1A_2
1	1	1	0	A_1A_2
1	1	1	1	A_1A_2

AB

X_1X_2

	00	01	11	10
00	0	0	1	1
01	0	1	0	1
11	A_1	A_1	A_1	A_1
10	1	0	1	0

$$A^* = A(\overline{X}_1\overline{B}+\overline{X}_2B) + \overline{A}(\overline{X}_1X_2B+X_1\overline{X}_2\overline{B}) + X_1X_2A_1$$

AB

X_1X_2

	00	01	11	10
00	0	1	1	0
01	1	0	0	1
11	A_2	A_2	A_2	A_2
10	1	0	0	1

$$B^* = B(\overline{X}_1\overline{X}_2)+ \overline{B}\ (\overline{X}_1X_2+X_1\overline{X}_2) + X_1X_2A_2$$

$\therefore$

$$A^* = A(\overline{X}_1\overline{B}+\overline{X}_2B+X_1X_2A_1)+\overline{A}\ (\overline{X}_1X_2B+X_1\overline{X}_2\overline{B}+X_1X_2A_1)$$

$$B^* = B(\overline{X}_1\overline{X}_2+X_1X_2A_2) + \overline{B}\ (\overline{X}_1X_2=X_1\overline{X}_2=X_1X_2A_2)$$

Figura V.20

Así:

$$J_a = \overline{X}_1X_2B+X_1\overline{X}_2\overline{B}+X_1X_2A_1)$$

$$K_a = \overline{X}_1B+X_2B+X_1X_2A_1$$

$$J_b = \overline{X}_1X_2+X_1\overline{X}_2+X_1X_2A_2=X_1+X_2+X_1X_2A_2$$

$$K_b = \overline{\overline{X}_1\overline{X}_2+X_1\ X\ A}$$

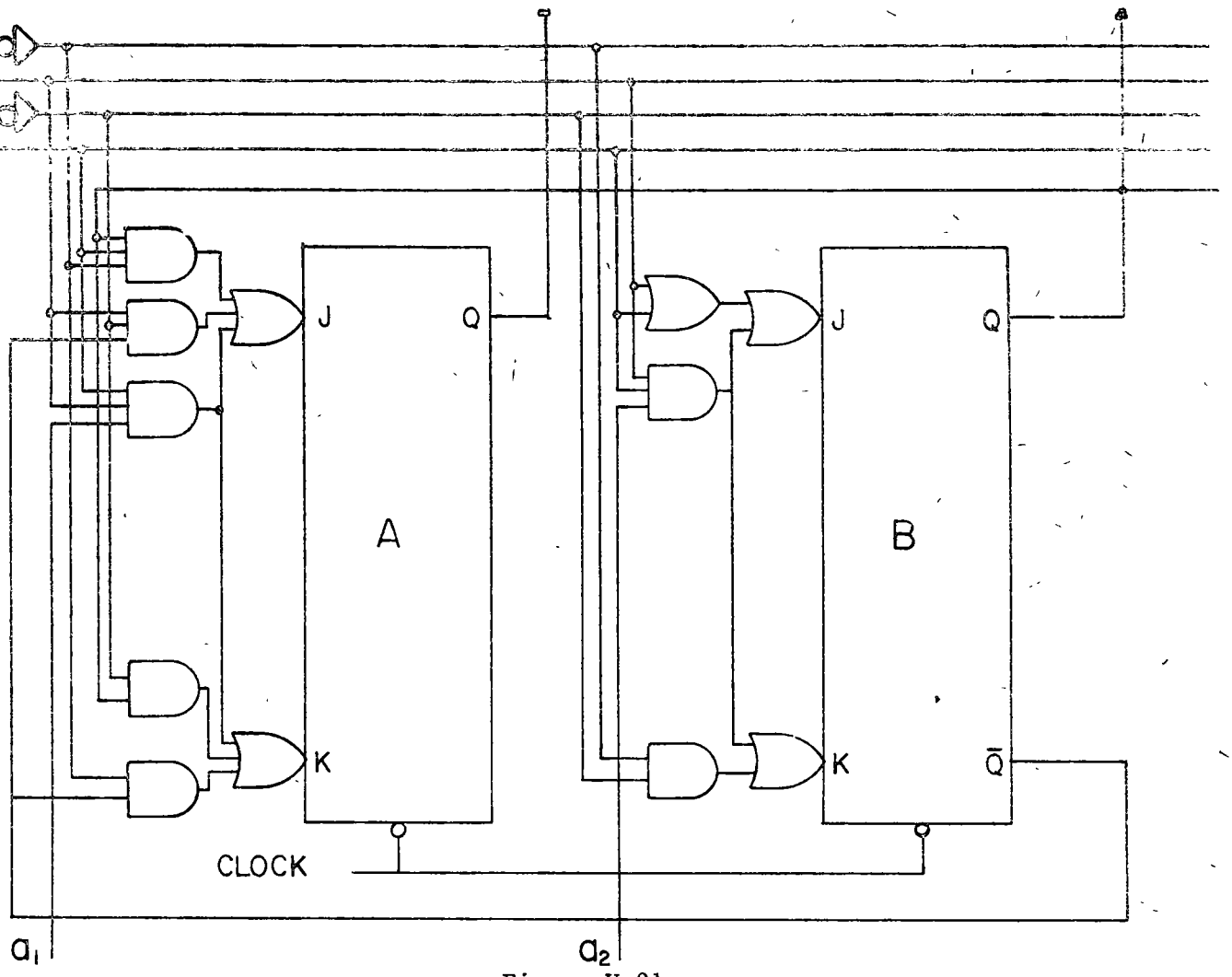
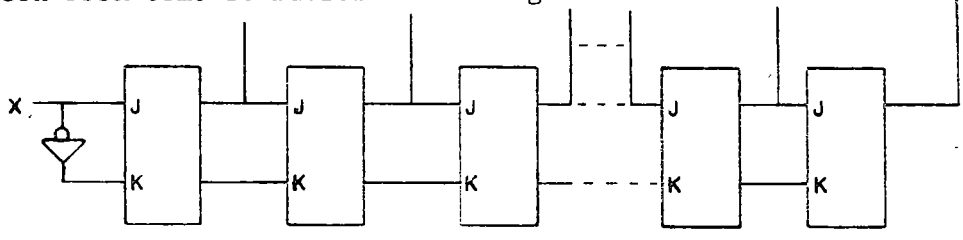


Figura V.21

V.4. Registros de Corrimiento.

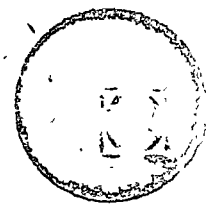
V.4.1. Los registros de corrimiento por otros de los elementos más utilizados en el diseño lógico y su máxima aplicación se encuentra en los sistemas de cómputo los cuales manejan paquetes o conjuntos (Buts) de información.

Un registro de corrimiento es fácilmente diseñado a través de las técnicas presentadas en el capítulo V. Supóngamos por ejemplo una realización con FFJK como se muestra en la figura V.22.



Este constituye uno de los casos más simples de corrimiento: "corrimiento a la derecha"; en la práctica existen registros de corrimiento a la derecha é izquierda al igual que en forma circular.

centro de educación continua
división de estudios superiores
facultad de ingeniería, unam



INTRODUCCION AL PROCESAMIENTO DIGITAL

CAPITULO 6: CIRCUITOS SECUENCIALES ASINCRONOS

CAPITULO 7: UNIDADES ARITMETICAS

M. en C. Angel Kuri M.

Septiembre, 1977

Capítulo VI, Circuitos Secuenciales Asíncronos.

6.1. Introducción. En muchas situaciones prácticas los pulsos de sincronía del reloj no están presentes y es necesario el diseño de circuitos llamados asíncronos (es decir, que no utilizan señal de sincronía). Por otro lado, debe resultar claro que la frecuencia de un reloj resulta una limitante en la velocidad de un circuito digital, excepto en los casos en que dicha frecuencia es igual a, o mayor que, la velocidad de conmutación de cicho circuito.

Ejemplo 6.1.

Un circuito digital realizado con tecnología ECL (emitter coupled logic) tiene tiempos de conmutación en sus compuertas del orden de 1-2 nS$\left(1\text{-}2\ x10^{-9}\ \text{seg}\right)$. ¿Cuál es la frecuencia que debe tener el reloj del sistema para que el circuito funcione a su máxima velocidad teórica ?

Tiempo de conmutación

$T=2x10^{-9}\text{seg}$

$\therefore$ frecuencia

$$F=\frac{1}{2x10^{-9}\text{seg.}}=\frac{1}{2}\ x10^{9}\text{seg}^{-1}$$

$$=.50x10^{9}\ \text{seg}^{-1}=500\ \text{MHZ}$$

$$\underline{F=500\ \text{MHZ}}$$

Dichas frecuencias no son comunes, son relativamente difíciles de obtener de circuitos convencioneales y tienen la desventaja adicional de hacer que los efectos de capacidades espúreas, que a bajas frecuencias pueden considerarse despreciables, se hagan de consideración. Por otro lado, en sistemas de cómputo de cierta complejidad suele suceder que dos circuitos de un sistema, dada su separación física, pierdan sincronía a pesar del reloj del sistema. Por ejemplo, para que una señal eléctrica se propague a través de una distancia de 10 m (suponiendo $v=c=3x10^{8}$ m/seg) se requieren del orden de 35 ns. ¡En este intervalo, el reloj del ejemplo anterior habrá cambiado de "estado" 17 veces!

Por dichas razones es conveniente hablar de las propiedades básicas de los circuitos asíncronos y de los métodos de síntesis de los mismos.

6.2. Circuitos de modo fundamental. Representaremos al circuito asíncrono con el esquema de la figura 1.

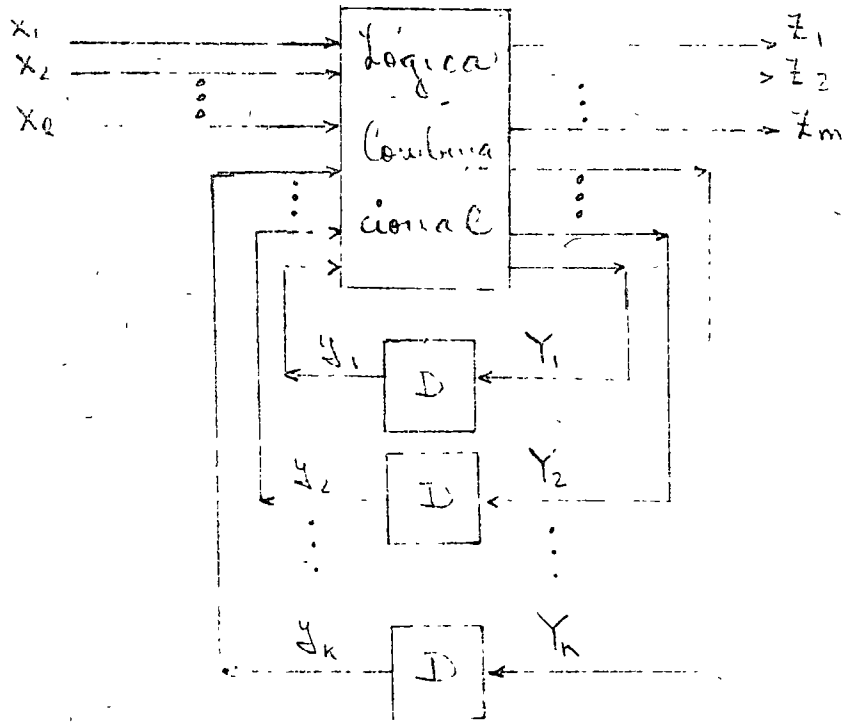


Figura 1. Circuito secuencial asíncrono.

Este circuito tiene la característica de que sus entradas (X_1, X_2, ..., X_e) pueden cambiar en cualquier instante y sus entradas y salidas están representadas por niveles y no por pulsos.

Internamente se caracteriza por utilizar elementos de retardo ('delays') como dispositivos de memoria.

Las variables de estado (y_1, y_2, ..., y_k) definen el estado interno del circuito y las variables de entrada (X_1, X_2, ..., X_e) definen su estado externo. Las variables Y_1, Y_2,...,Y_k se conocen como variables de excitación.

Cuando $y_i = Y_i$ para toda i, se dice que el circuito está en un estado estable. Cuando hay un cambio en las entradas, la lógica combinacional produce un nuevo conjunto de valores para las variables de excitación y el circuito entra en un estado inestable hasta que las variables de es-

tado (después de un retardo) asumen al mismo valor de las variables de excitación.

Supondremos que las variables de entrada cambian sólo después de un tiempo Δt suficiente para que el circuito se estabilice. A este tipo de operación se le llama modo fundamental.

Otro modo de operación, conocido como modo pulso será discutido posteriormente.

En la práctica, debido a las características físicas de los circuitos electrónicos, es imposible asegurar que el cambio en dos o más entradas será simultáneo. Por ello se especifican las siguientes restricciones:

a) Sólo una de las entradas puede variar a un tiempo.
b) El tiempo de cambio entre dos entradas será mayor o igual al mínimo tiempo de conmutación de los elementos del circuito.

Es de hacer notar que, en ciertos casos, los elementos de retardo, o "memoria" del circuito pueden ser dados por los mismos retardos inherentes de la lógica combinacional. Por claridad, únicamente, los representamos como elementos de retardo independientes del resto del circuito.

6.3 Síntesis. Al igual que en el caso de los circuitos síncronos, el primer paso del diseño es pasar del planteamiento verbal a la descripción precisa del comportamiento del circuito bajo cualquier combinación de entrada. Para ello utilizaremos el análogo del diagrama de estados de un circuito síncrono, que en este caso es la tabla de flujo.

Consideremos un circuito con dos entradas X_1 y X_2 y una salida Z. El estado inicial es $X_1 = X_2 = 0$. El circuito responde con un '1' a la salida sólo si el estado de entrada es $X_1 = X_2 = 1$ y el estado de entrada precedente es $X_1 = 0$, $X_2 = 1$. Algunas posibles secuencias y sus correspondientes salidas se muestran el la figura 2.

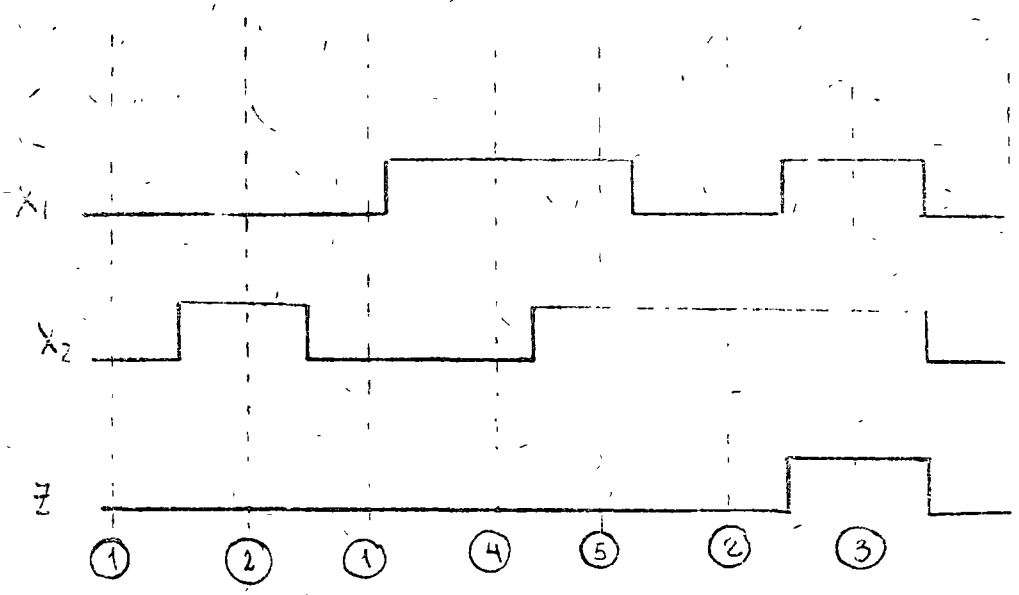


Figura 2. Secuencia de entrada-salida.

Para construir la tabla de flujo empezamos por "etiquetar" las columnas con las combinaciones de entrada. Luego, de acuerdo con la secuencia de entrada-salida, determinamos los estados estables, que denotamos con un número encerrado en un círculo, de la manera que se muestra en la figura 3.

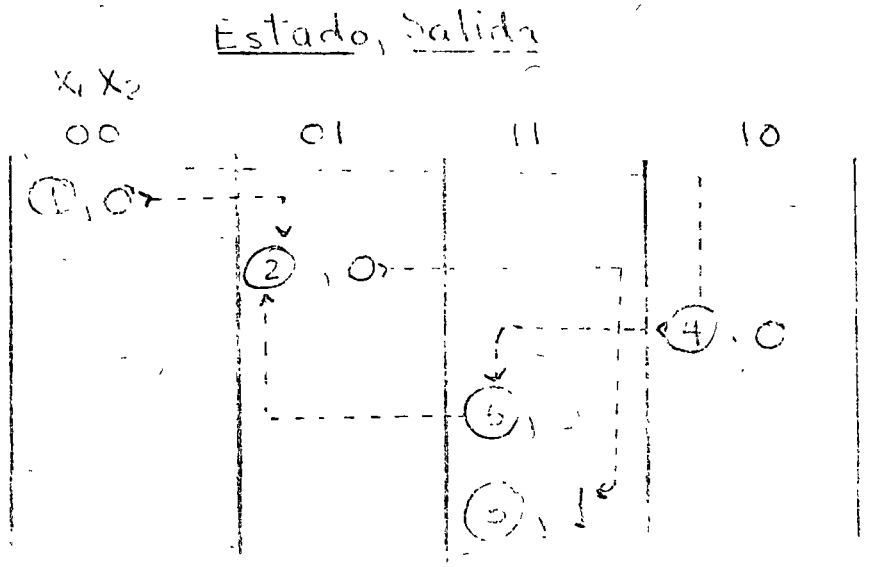


Figura 3. Tabla de flujo parcial.

Notése que en esta tabla solamente se han indicado los estados estables. Por ejemplo, el paso del estado 1 al 2 necesariamente implica una transición por un estado inestable que no está indicado. Lo mismo ocurre en las transiciones 1 a 4, 5 a 2, etc. La conclusión lógica es que el estado inestable 1 a 2 sea el estado que denotaremos como 2; o el estado 4 en el caso de 1 a 4, etc. En los estados inestables la salida no está especificada. Convencionalmente puede suponerse que la salida en un estado inestable sea la misma del estado destino. Con ello se asegura que el cambio a la salida (en caso de existir) debe ser tan rápido como sea posible. Por otra parte, ciertas transiciones serán prohibidas. Este es el caso en él cual haya dos o más cambios en las variantes de entrada. Considerando estas restricciones es posible desarrollar la tabla primitiva de flujo, como se muestra en la figura 4.

Estado, Salida

$x_1 x_2$ 00	01	11	10
(1), 0	2, 0	—	4, 0
1, 0	(2), 0	3, 1	—
1, 0	—	5, 0	(4), 0
—	2, 0	(5), 0	4, 0
—	2, 0	(2), 1	4, 0

Figura 4. Tabla de flujo primitiva.

En esta tabla hay un estado estable por renglón y las salidas de los estados inestables se han tomado de manera que las transiciones de dichas salidas sean inmediatas.

6.4. Reducción de Tablas de Flujo. Como en el caso de los circuitos síncronos, es deseable eliminar los estados "redundantes" de manera que el circuito final sea mínimo. En este sentido hay dos diferencias fundamentales, que son:

a) Algunos de los estados del circuito no están especificados.

b) Propiamente hablando, en el caso de los circuitos asíncronos no puede hablarse de una transición de "estado actual" a "siguiente estado" puesto que dichas transiciones son irregulares. (No dependen de un reloj).

Sin embargo, para propósito de diseño, puede suponerse que el circuito, en un momento dado, está en un estado y que hay una transición marcada a un estado posterior, a pesar de que, en realidad, dichos cambios pueden ser totalmente irregulares. Esto se puede observar de la figura 2, en donde se ve claramente que la duración de un estado cualquiera es aleatoria. Puede entonces desarrollarse una tabla de flujo como la siguiente:

"Estado Actual"	$x_1 x_2$ 00	01	11	10
(1)	(1),0	2,0	—	4,0
(2)	1,0	(2),0	3,1	—
(3)	—	2,0	(3),1	4,0
(4)	1,0	—	5,0	(4),0
(5)	—	2,0	(5),0	4,0

Figura 5. Tabla de Flujo Completa.

6.4.1. Reducción de un autómata incompletamente especificado. Sin discutir la teoría subyacente, bosquejaremos aquí el método de reducción de los llamados autómatas incompletamente especificados. Todos los circuitos asíncronos de modo fundamental dan orígen a este tipo de autómatas. El lector interesado en la justificación formal del método que va a discutirse puede consultar las referencias [1] y [2].

La diferencia básica entre un autómata completamente especificado y uno como el que nos ocupa, que no lo es, es que ciertos estados que especifican el comportamiento del circuito no están determinados. Esto puede ocurrir porque no se conocen o, porque, como en el presente caso, no pueden presentarse.

El procedimiento a seguir consta de los siguientes pasos:

1) Se determina una tabla de conjunción.
2) Se encuentran los compatibles máximos.
3) Se busca un cubrimiento mínimo.
4) Se desarrolla el autómata mínimo.

Cada uno de estos pasos será ejemplificado con el ejemplo anterior y con otros de los que nos ocuparemos más adelante.

1) Tabla de Conjunción. La tabla de conjunción se forma etiquetando una gráfica en escalera con M-1 peldaños (M= número de estados) en sentido horizontal y M-1 peldaños en sentido vertical de modo que el primero y último peldaños verticales correspondan a los estados del 2 al M respectivamente; en sentido horizontal los peldaños primero y último corresponden a los estados del 1 al M-1 respectivamente. Para el ejemplo de la figura 5 la tabla será la siguiente:

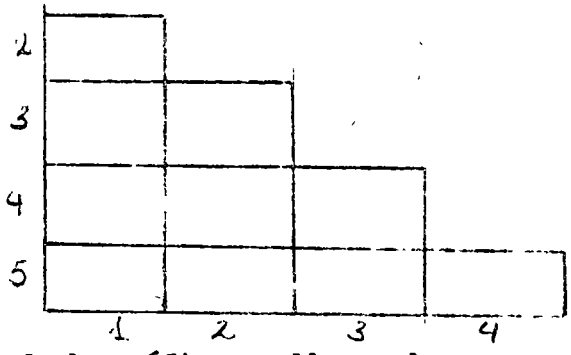


Figura 6.

Las casillas de la gráfica se llenan de manera que aparezca una paloma (√) en aquéllas que correspondan a la intersección de dos estados cuyas salidas sean iguales (o no especificadas) para todas las combinaciones de entrada y una cruz (X) en aquéllas correspondientes a estados con salidas diferentes. Para el ejemplo:

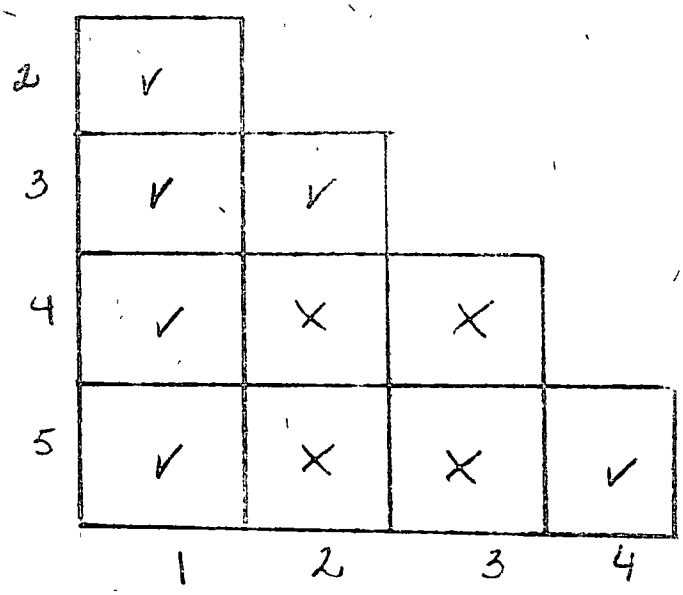


Figura 7.

2) Para encontrar los compatibles máximos de la tabla se procede de derecha a izquierda asociando por parejas a aquéllos estados que sean compatibles (es decir, que tengan una paloma en su intersección). En el ejemplo, (4,5) son compatibles así como (2,3), (1,2), (1,3), (1,4) y (1,5). Lo que deseamos es encontrar grupos de 2 o más estados compatibles entre sí. Para que tres estados sean compatibles (digamos los estados A, B y C), se requiere que las parejas AB, BC y AC sean todas compatibles entre sí. Sólo en ese caso podrán considerarse compatibles ABC, (en un grupo de 3).

De esta manera, sabemos que existen los siguientes compatibles máximos para nuestro ejemplo.

(4,5) (1,4) (1,5) → (1,4,5)

(2,3) (1,3) (1,2) → (1,2,3)

3) Lo que llamamos un cubrimiento mínimo es un conjunto de compatibles tales que cubran (o abarquen) a todos los estados del circuito original. Se puede demostrar que cada conjunto de compatibles es, en sí, una representación de un circuito cuyas características de funcionamiento son equivalentes al circuito original, en donde cada compatible del cubrimiento mínimo del circuito corresponde a un estado del circuito mínimo. (Es decir, aquel con menor número de estados).

Para el ejemplo, los conjuntos

(4,5) (1,2,3); (1,4,5) (1,2,3)

(1,4,5) (2,3)

son todos cubrimientos mínimos.

Si hacemos

(1,2,3)→ A , (4,5)→ B, de la figura 5 podemos encontrar la siguiente tabla de flujo reducida, correspondiente al circuito mínimo:

	$X_1 X_2$ 00	01	11	10
A	A, 0	A, 0	A, 1	B, 0
B	A, 0	A, 0	B, 0	B, 0

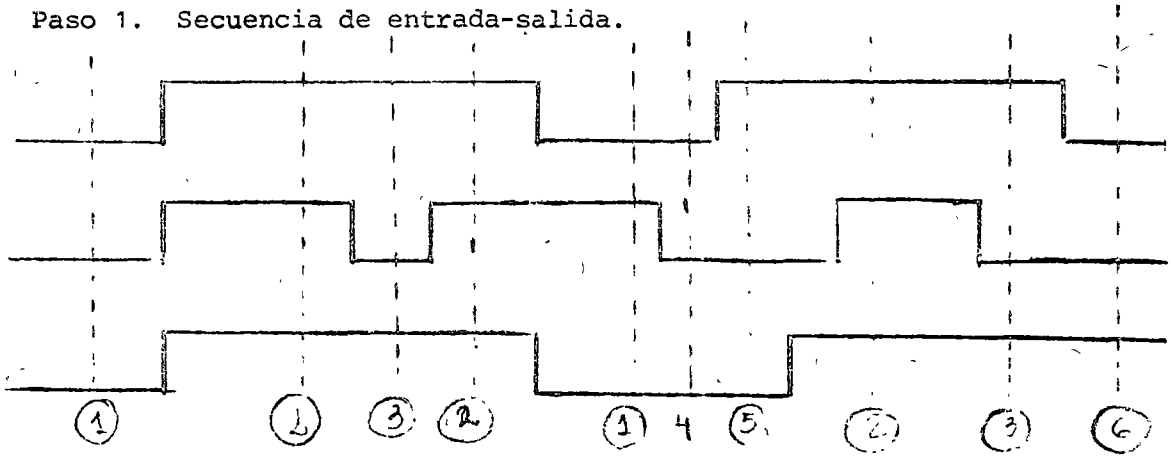
Figura 8. Tabla de flujo reducida.

De aquí, utilizando las técnicas desarrolladas en el capítulo anterior, es elemental obtener el diseño del circuito.

Como un ejemplo ilustrativo de la técnica antes señalada, consíderese el diseño del siguiente circuito.

Ejemplo. Un circuito tiene dos entradas X_1, X_2, y una salida Z. Cuando X_2= 1 el valor de Z es igual al valor de X_1; cuando X_2=0 la salida permanece fija a su último valor previo a que X_2 se hiciera 0.

Paso 1. Secuencia de entrada-salida.

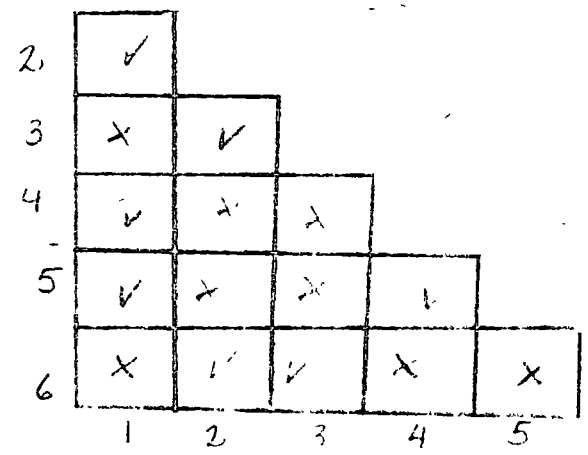


Paso 2. Tabla de flujo.

(Siguiente Página)

	X_1X_2 00	01	11	10
(1)	4,0	(1),0	2,1	—
(2)	—	1,0	(2),1	3,1
(3)	6,1	—	2,1	(3),1
(4)	(4),0	1,0	—	5,0
(5)	4,0	—	2,1	(5),0
(6)	(6),1	1,0	—	3,1

Paso 3. Tabla de conjunción.



Paso 4. Compatibles máximos. (Examinando la tabla de derecha a izquierda).

(5)
(45)
(45) (36)
(45) (236)
(145) (12) (236)

Paso 5. Cubrimiento mínimo.

(145) (236)

Paso 6. Tabla de flujo reducida.

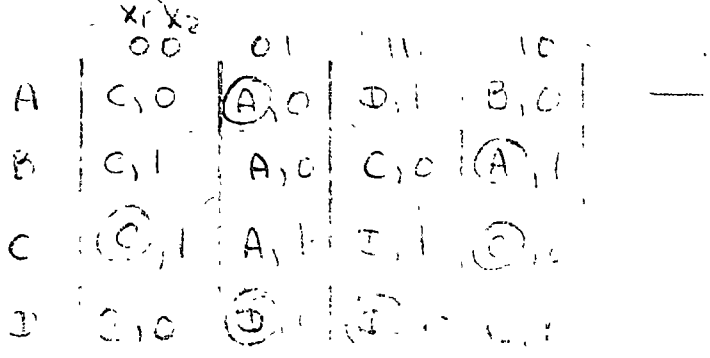
	X_1X_2 00	01	11	10
(145) → A	A,0	A,0	[illegible]	B,0
(236) → B	B,1	A,0	B,1	B,1

Es de hacerse notar que, en circuitos asíncronos, la tabla reducida, a diferencia del caso de los circuitos síncronos no es única.

6.5. Asignación de estados en circuitos asíncronos.

El paso siguiente en el diseño de un circuito asíncrono es, a partir de la tabla reducida, asignar las variables secundarias a los estados de dicha tabla. Como se indicó anteriormente, es imposible asegurar que dos variables cambiarán sus valores simultáneamente. Por ello, la asignación de variables secundarias a las filas de una tabla reducida debe ser tal que el circuito funcione correctamente aún si se asocian diferentes retardos a los elementos secundarios.

6.5.1 Carreras y ciclos. Considérese la siguiente tabla, correspondiente a un circuito reducido:



Para este circuito se requieren, al menos, dos variables. Supongamos la siguiente asignación (no consideraremos las salidas por comodidad).

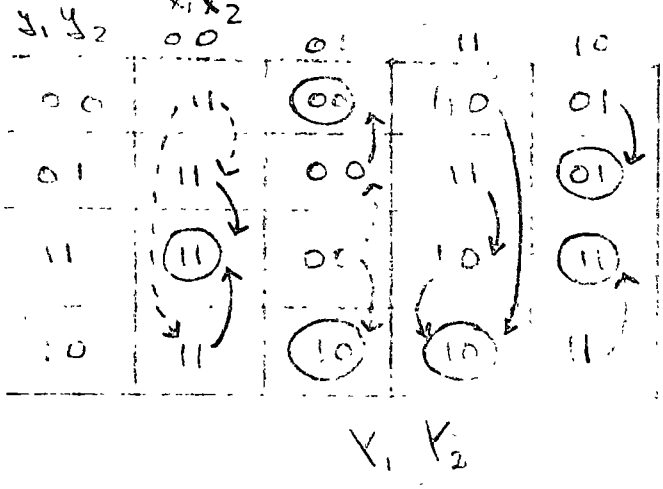


Figura 9. Ilustración de carreras y ciclos.

Cuando ambas entradas son iguales a 0 y y_1y_2=00, la trancisión requerida al estado y_1y_2=11 implica el cambio de valor en dos de las variables secundarias. Si estos cambios ocurren simultáneamente, la transición deseada tendrá lugar. Sim embargo, si y_1 ó y_2 cambia primero, en lugar de ir al estado (11) el circuito irá al estado 01 o 10. Afortunadamente, dado que tanto los estados 01 como 10, bajo una entrada X_1X_2=00, van al estado (11), el circuito finalmente se estabilizará en el estado (11). A esta situación, en donde se requiere el cambio de más de una variable, se le llama una 'carrera'. Si el comportamiento del circuito no depende del orden en que cambien las variables (como en el caso anterior), la carrera se conoce como 'no-crítica'.

Supongamos ahora que el circuito está en el estado y_1y_2=11 y que X_1X_2=01. La transición requerida es el estado y_1y_2=00. Si y_1 cambia antes que y_2, el circuito irá al estado 01 y de ahí al 00. Si y_2 cambia antes que y_2, el circuito va al estado (10), y permanecerá ahí puesto que el estado total y_1y_2=10; X_1X_2=01 es estable. En este caso el circuito no funciona adecuadamente. A esta situación se le llama 'carrera crítica'.

Las carreras pueden a veces evitarse usando estados inestables intermedios. Por ejemplo, para el estado total y_1y_2=01; X_1X_2=11, se desea pasar a (10). Los pasos, sin embargo son y_1y_2=01 →11→ 10. De esta forma, usando el estado inestable 11 como pivote, se logra el paso del estado 01 a (10) sin incurrir en carreras críticas. A este caso, en donde el circuito pasa a través de una secuencia única de estados inestables se conoce como un 'ciclo'. Si un ciclo no contiene un estado estable, el circuito oscilará hasta que haya un cambio en las entradas. Claramente, esta situación debe ser evitada. Una asignación (para el circuito de la figura 9) en la cual no hay carreras críticas ni ciclos espúreos se conoce como 'válido' y se muestra en la figura 10.

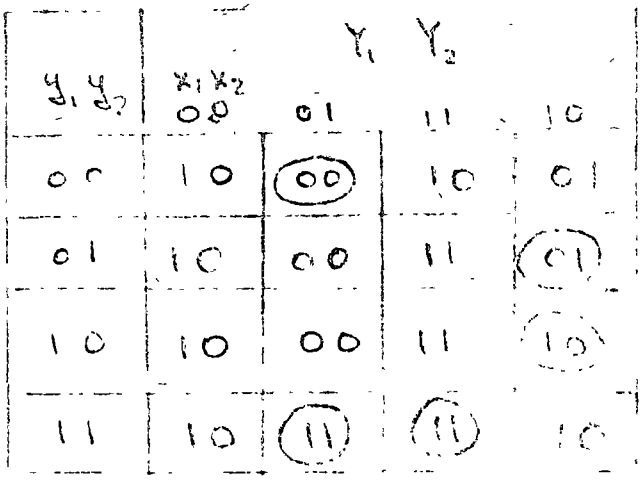


Figura 10. Una asignación válida.

6.5.2 Métodos de asignación secundaria. Para sistematizar la asignación de variables secundarias es conveniente definir a aquéllos estados que difieren entre si en un sólo valor de las variables de estado como estados adyacentes. Consideremos la tabla de la figura 11.

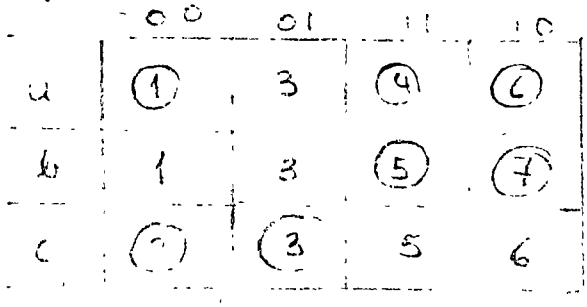


Figura 11. Tabla de flujo reducida.

Examinando la columna 00, concluímos que las filas a y b deben ser adyacentes, puesto que la transición de 1 a 1 debe involucrar un solo cambio de variable. De manera similar concluímos:

Columna 00: b debe ser adyacente a a
01: a y b deben ser adyacentes a c
11: c debe ser adyacente a b
10: c debe ser adyacente a a

En forma diagramática esto lo podemos expresar de la siguiente forma:

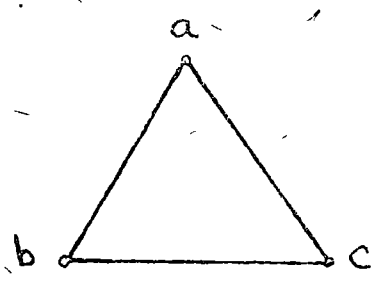


Figura 11. Gráfica de adyacencias.

Para 4 estados se requieren, al menos, dos variables (y_1 y_2), lo que la gráfica indica es que todos los estados de la figura 10 (a, b y c) son mútuamente adyacentes.

Consideremos la asignación:
a →00, b→ 01, c→ 11
En ese caso, se viola la adyacencia a ↔ c. Consideremos, entonces:
a→ 00, b→ 01, c→ 10
En este caso es b ↔ c la adyacencia que se viola. Existen 24 formas de asignación y es fácil convencerse de que ninguna es satisfactoria. Las alternativas son:

1) Utilizar más variables de estado
2) Utilizar combinaciones no especificadas

como estados inestables "pivote".

En general, la alternativa de (2) es más económica. Consideremos, pues, la asignación:
a→ 11, b→ 10, c→ 01. De este modo:

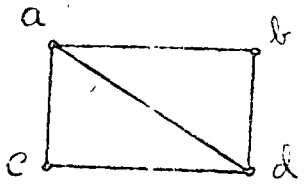
	X_1X_2 00	01	11	10
a→ 11	(11)	01	(11)	(11)
b→ 10	11	01	(10)	(10)
c→ 01	01	01	00	11
00	-	-	01	-

Como se vé, usamos el estado no especificado y_1y_2=00 para generar un ciclo que conduce al circuito del estado 01 al 10. Consideremos ahora el siguiente ejemplo:

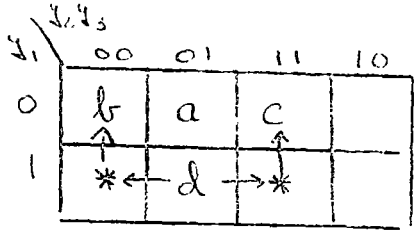
	X_1X_2 00	01	11	10
a	①	②	4	⑥
b	1	3	④	⑦
c	1	2	⑤	⑧
d	1	③	5	6

Figura 12. Ejemplo de asignación secundaria.

Obtengamos la gráfica de adyacencias, de modo que tenemos:



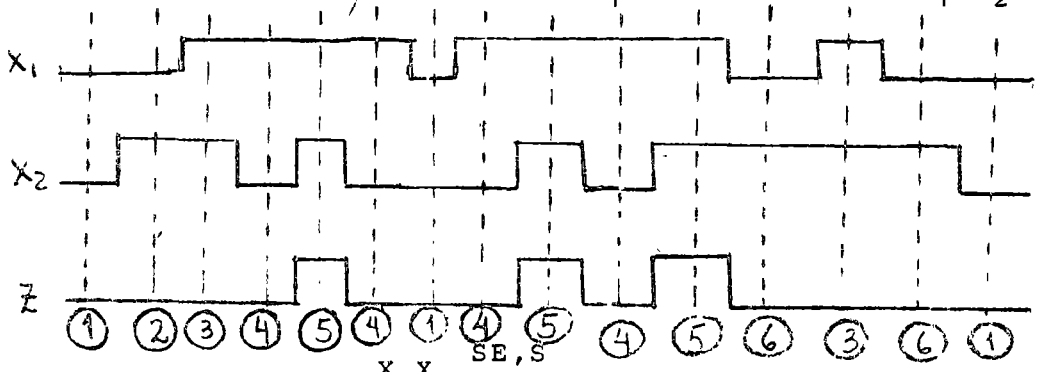
Para encontrar una asignación que cumpla con estas adyacencias podemos usar un mapa de Karnaugh, en donde es fácil ver la relación entre las diversas asignaciones. Recordando que para 4 estados se requiere de 2 variables, tenemos que concluir que las adyacencias se tienen que violar en el caso que nos ocupa a menos que agreguemos una tercera variable. De esa forma podemos usar el siguiente mapa:



En donde los asteriscos indican un punto de pivote. La tabla correspondiente es la siguiente:

$y_1y_2y_3$	X_1X_2 = 00	01	11	10
a → 001	(001)	(001)	000	(001)
b → 000	000	100	(000)	(000)
c → 011	001	001	(011)	(011)
d → 101	001	(101)	111	001
100	–	101	–	–
111	–	–	011	–

6.6 Un diseño asíncrono. Estamos ahora en condición de diseñar un circuito asíncrono de modo fundamental hasta llegar a los componentes electrónicos. Para ello consideremos el siguiente problema: un circuito con dos entradas (X_1X_2) y una salida (Z) en el cual Z=1 si y sólo si $X_1=X_2=1$ y el penúltimo cambio a la entrada fue de X_1. Suponga que $X_1=X_2=0$ al inicio.



E.A.	X_1X_2 = 00	01	11	10
1	(1),0	2,0	–	4,0
2	1,0	(2),0	3,0	–
3	–	6,0	(3),0	4,0
4	1,0	–	5,1	(4),0
5	–	6,0	(5),1	4,0
6	1,0	(6),0	3,0	–

2	✓				
3	✓	✓			
4	✓	X	X		
5	✓	X	X	✓	
6	✓	✓	✓	X	X
	1	2	3	4	5

(45) (236)

(145) (1236)

SE,S

EA	X_1X_2 00	01	11	10
45→ A	1,0	6,0	(5),1	(4),0
1236→ B	(1),0	(6),0	3,0	4,0

y	X_1X_2 00	01	11	10
a→ 0	1,0	1,0	0,1	0,0
b→ 1	1,0	1,0	1,0	0,0

X_1X_2

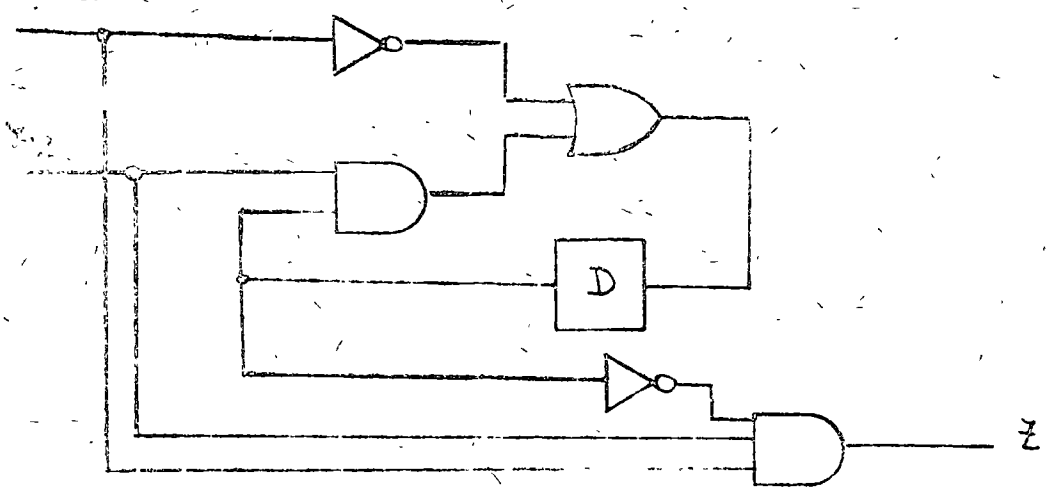
y	00	01	11	10
0	1	1		
1	1	1	1	

$$Y = \overline{X_1} + yX_2$$

X_1X_2

y	00	01	11	10
0			1	
1				

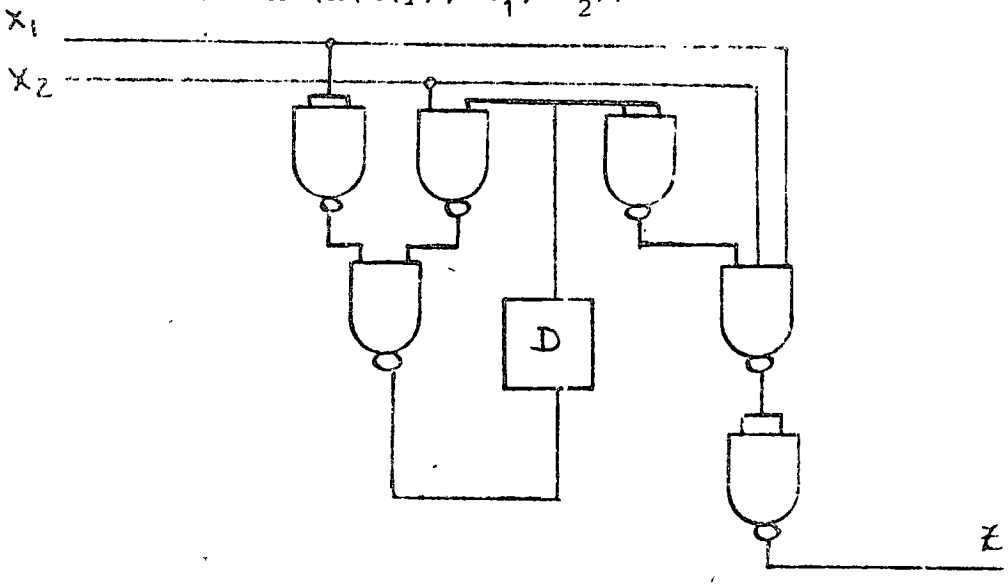
$$Z = \overline{y}X_1X_2$$



$Y = M(\bar{X}_1, M(y_1, X_2))$

$Y = M(M(X_1), M(Y_1, X_2))$

$Z = M(M(M(y), X_1, X_2))$



6.7. Diseño usando 'Flip-Flops'. En general, distinguiremos un Flip-Flop (elemento de memoria con reloj) de un circuito de 'cerrojo' (latch). Este último se muestra en la figura 12.

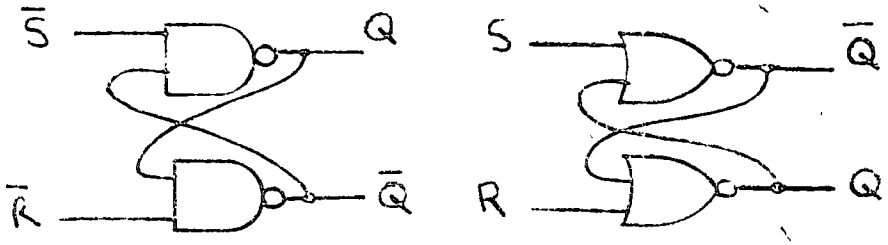


Figura 12. Cerrojo ('latch') S-R implementado con NAND y NOR.

Cuando se diseñan circuitos asíncronos sólo se pueden utilizar elementos de retardo o circuitos como los anteriores.

El diseño de un circuito asíncrono usando RS's se ilustra a continuación.

Problema: Un circuito tiene dos entradas (J,K) y una salida Q. Si J=1 y K=0, entonces Q=1. Si J=0 y K=1 entonces Q=0. Si J=K=0 el valor de Q debe ser igual al valor que tenía antes de que J o K cambiara (de 1 a 0). Si J=K=1, el valor de Q debe ser el complemento del que tenía antes de que J o K cambiara (de 0 a 1). Diseñe usando sólo compuertas Nand.

Tabla de Flujo.

	SE,Q			
E.A.	X_1X_2 00	01	11	10
A	(A),0	B,0	-	C,1
B	A,0	(B),0	D,1	-
C	E,1	-	F,0	(C),1
D	-	B,0	(D),1	C,1
E	(E),1	B,0	-	C,1
F	-	B,0	F,0	C,1

Tabla de conjunción.

B	✓				
C	X	X			
D	✓	✓	X		
E	X	X	✓	✓	
F	✓	X	✓	X	✓
	A	B	C	D	E

Máximos compatibles: (CEF) (DE) (ABD) (AF)

Cubrimiento mínimo: (ABD) (CEF)

Tabla de flujo reducida

	00	01	11	10
ABD	A,1	B,0	D,1	C,1
CEF	E,1	B,0	F,0	C,1

Hagamos ahora ABD=0; CEF=1

JK

y	00	01	11	10
0	0,0	0,0	0,1	1,1
1	1,1	0,0	1,0	1,1

Ahora obtengamos el mapa de Y y Q.

JK

y	00	01	11	10
0				1
1	1		1	1

y

Y

J,K

y	00	01	11	10
0			1	1
1	1			1

Q

De donde es fácil obtener las ecuaciones de S y R con el siguiente procedimiento.

Para S.

1. Háganse todos los 1's en la sección del mapa correspondiente a la variable (en este caso y) iguales a no-importa.
2. Diséñese con los 1's restantes.

De esta manera,

$S = J\overline{K}$

Para R.

1. Háganse todos los 0's en la sección del mapa correspondientes a la variable negada (es decir $\overline{y}$) eguales a no-importa.
2. Diséñese con los 0's restantes.

De donde,

$R = \overline{J}K$

Por otro lado, del mapa de Q se ve que: $Q = J\overline{y} + \overline{K}y$

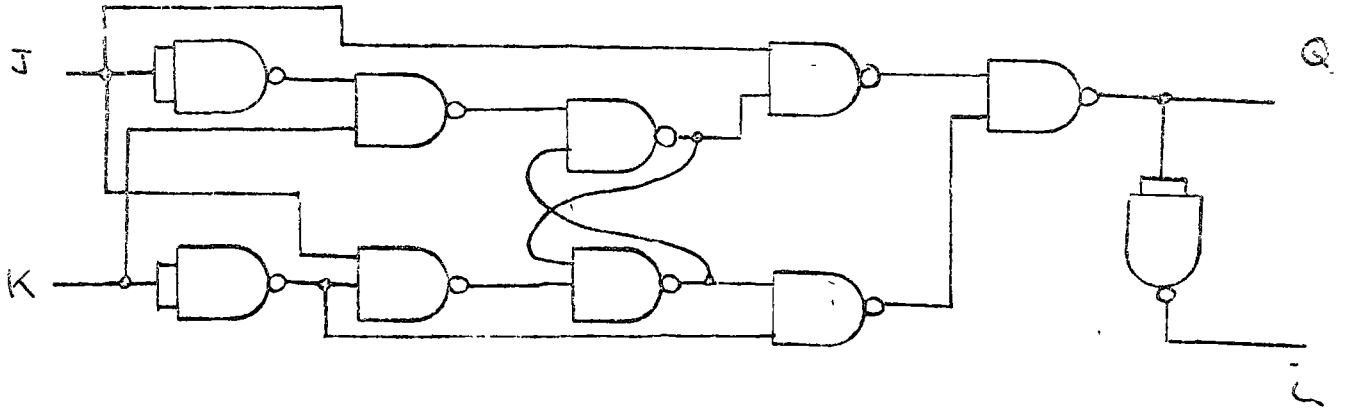
$\overline{S} = M[M(K),J] = \overline{JK}$

$\overline{R} = M[M(J),K] = \overline{JK}$ $\quad \{M(X,y) = \overline{Xy}\}$

y, para Q,

$Q = M[M(\overline{J},\overline{y}),\ M(\overline{K},y)]$

de modo que:



Este circuito se puede implementar utilizando 3 circuitos integrados 7400.

6.8. Circuitos asíncronos modo de pulso. En ciertos casos prácticos es conveniente diseñar circuitos asíncronos cuyas entradas son pulsos, es decir, la ausencia de nivel lógico no contiene ninguna información.

Los pulsos de entrada deben ser suficientemente largos para permitir el cambio en los elementos de memoria y suficientemente cortos para evitar combios dobles. Cuando llega un pulso, se dispara el circuito y se cambia de un estado estable a otro. Por la naturaleza de los pulsos, se debe de diseñar usando flip-flops asíncronos como elementos de memoria.

Debido a que los flip-flops estabilizan las salidas, el método de modo de nivel ya no es válido y, por ello, utilizamos una versión modificada de diseño de circuitos síncronos.

Puesto que la ausencia de pulso es irrelevante, el número de columnas de "siguiente estado" de la tabla de transición es igual al número de terminales de entrada.

Utilizaremos el siguiente ejemplo para ilustrar la técnica de diseño.

6.9. Diseño de un Circuito modo de pulso. Diseñaremos un circuito que controlará un servidor automático de refrescos. Dicho circuito recibe monedas de 25,50 y 100 centavos. El precio del refresco es de 1.25 pesos. Cuando se alcanza (o sobrepasa) dicha cantidad, el circuito genera un nivel de salida Z=1, que libera a la botella. Cuando ésta se retira, se produce un pulso de reinicio que restaura las condiciones iniciales del circuito. La tabla de transición es, pues, la siguiente:

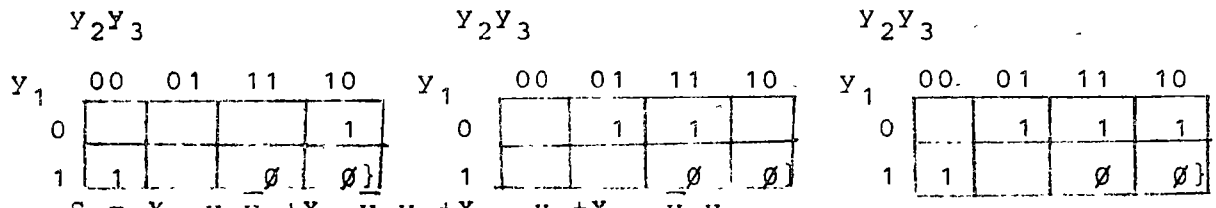
CANTIDAD	E.A.	SE X_{25}	X_{50}	X_{100}	X_r	Z
0	A	B	C	D	A	0
25	B	G	E	F	A	0
50	G	E	D	F	A	0
75	E	D	F	F	A	0
100	D	F	F	F	A	0
125 o más	F	B	C	D	A	1

El circuito, como es claro, es mínimo. Podemos hacer la asignación correspondiente. Ya que hay 6 estados, se requieren 3 flip-flops.

	X_{25}	X_{50}	X_{100}	X_r	Z
A→ 000	001	010	011	000	0
B→ 001	010	100	101	000	0
C→ 010	100	011	101	000	0
D→ 011	01	101	101	000	0
E→ 100	101	101	101	000	0
F→ 101	001	010	011	000	1
110					
111					

En este caso, la variable de entrada está presente como factor la expresión de los SR, de modo que hay que obtener 3 mapas de S_1, 3 para R_1, etc. (1 por variable).

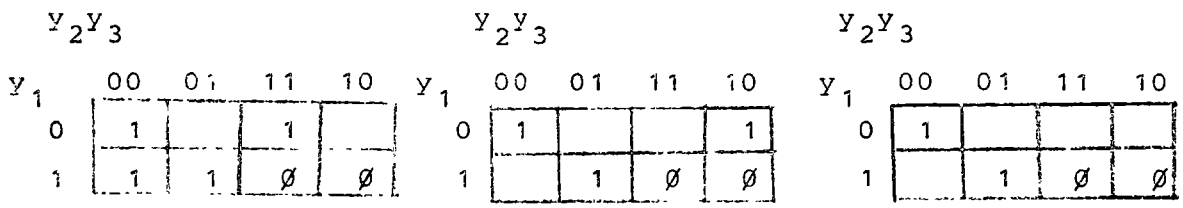
Para S_1; R_1:



$S_1 = X_{25}y_r\bar{y}_3 + X_{50}y_1y_3 + X_{100}y_2 + X_{100}y_1y_3$

$R_1 = X_{25}y_3 + X_{50}y_1y_3 + X_{100}y_1y_3$

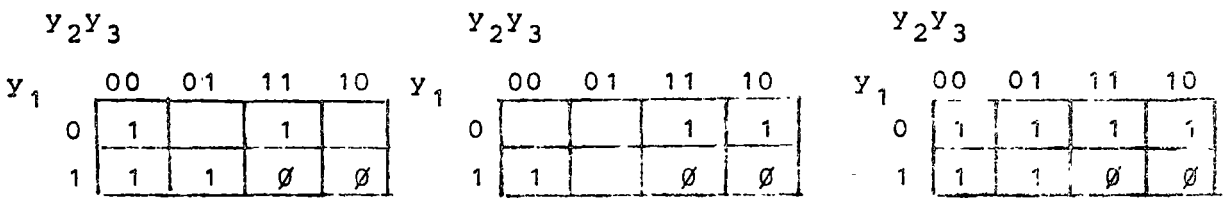
Para S_2; R_2:



$S_2 = X_{25}\bar{y}_1y_3 + X_{50}\bar{y}_1\bar{y}_3 + X_{50}y_1y_3 + X_{100}\bar{y}_1\bar{y}_2\bar{y}_3 + X_{100}y_1y_3$

$R_2 = X_{25}y_3 + X_{50}\bar{y}_1y_3 + X_{100}y_2$

Para S_3; R_3:



$$S_3 = X_{25}\bar{y}_2\bar{y}_3 + X_{50}y_2 + X_{50}y_1\bar{y}_3 + X_{100}$$

$$R_3 = X_{25}\bar{y}_1\bar{y}_2y_3 + X_{50}\bar{y}_2y_3$$

BIBLIOGRAFIA:

1. Introduction to Boolean Algebra and Logic Design. Hoerves and Heilweil, McGraw-Hill, 1964.

2. Digital Electronics with Engineering Applications. Franklin Kuo, Prentice Hall, 1970.

3. Introduction to Switching Theory and Logical Design. Hill & Peterson, Wiley, 1968.

4. Switching and Finite Automata Theory, McGraw-Hill, 1970. (Zvi Kohavi)

Capítulo VII.

Unidades Aritméticas. Un elemento aritmético es aquél que realiza una suma, resta, multiplicación, división o alguna otra función aritmética, generalmente con números binarios. Aunque no se restringe a computadoras o calculadoras, es en éstas dos aplicaciones en donde se usan estos elementos con mayor asiduidad. Una unidad aritmética es el conjunto de uno o más elementos aritméticos, y éstos son el tema de la discusión subsecuente.

7.1. El Sumador Completo y Algoritmos. A pesar de qué tan compleja pueda hacerse una operación, ésta puede siempre expresarse en función de un elemento básico llamado el sumador completo (full adder). Usando uno o más sumadores completos (SC) en una secuencia bien difinida de pasos (llamada algoritmo) es posible implementar funciones tan complejas como se desee.

Debido a la existencia de estos algoritmos, se requiere de la utilización de elementos lógicos auxiliares, como son:

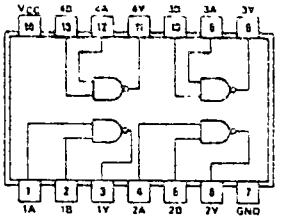
1) Registros de corrimiento
2) Contadores
3) Codificadores
4) Decodificadores
5) Multiplexores
6) Demultiplexores
7) Comparadores

Todos estos elementos pueden diseñarse utilizando las técnicas ya mencionadas en otros capítulos. Lo más común, sin embargo, es usar circuitos ya existentes en el mercado. En las siguientes páginas se incluyen los diagramas lógicos de algunos de los integrados más comunes. De estos hay que destacar el 74198, el 74181 y el 7400. Es posible diseñar cualquier circuito aritmético usando únicamente estos tres tipos de integrado. (De hecho lo mismo podría afirmarse del 7400 en particular, pero aquí no deseamos sino enfatizar la universalidad de estos 3 circuitos). Básicamente el 74198 representa una memoria (registro de corrimiento universal)

7400

QUADRUPLE 2-INPUT
POSITIVE-NAND GATES

positive logic
$Y = \overline{AB}$

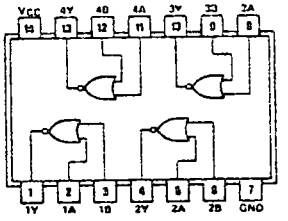


7400

7402

QUADRUPLE 2 INPUT
POSITIVE-NOR GATES

positive logic
$Y = \overline{A+B}$

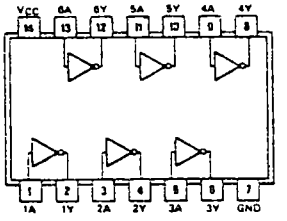


7402

7404

HEX INVERTERS

positive logic
$Y = \overline{A}$

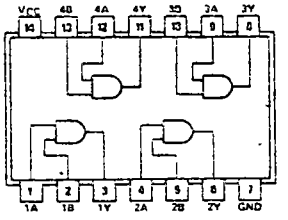


7404

7408

QUADRUPLE 2 INPUT
POSITIVE AND GATES

positive logic
$Y = AB$

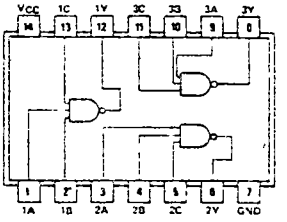


7408

7410

TRIPLE 3 INPUT
POSITIVE NAND GATES

positive logic
$Y = \overline{ABC}$



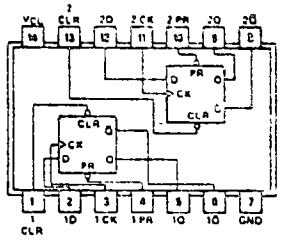
7410

7474

DUAL D TYPE POSITIVE EDGE TRIGGERED
FLIP-FLOPS WITH PRESET AND CLEAR

FUNCTION TABLE

INPUTS				OUTPUTS	
PRESET	CLEAR	CLOCK	D	Q	$\bar{Q}$
L	H	X	X	H	L
H	L	X	X	L	H
L	L	X	X	H*	H*
H	H	↑	H	H	L
H	H	↑	L	L	H
H	H	L	X	Q_0	$\bar{Q}_0$



7474

7475

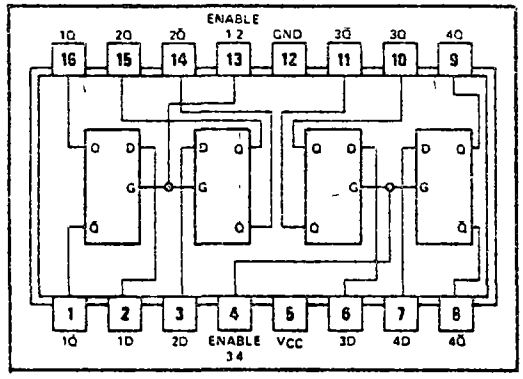
QUADRUPLE D TYPE LATCH

FUNCTION TABLE
(Each Latch)

INPUTS		OUTPUTS	
D	G	Q	$\bar{Q}$
L	H	L	H
H	H	H	L
X	L	Q_0	$\bar{Q}_0$

H = high level L = low level, X = irrelevant
Q_0 = the level of Q before the high-to-low transition of G

(TOP VIEW)



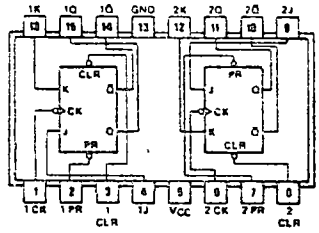
7475

7476

DUAL J-K FLIP FLOPS
WITH PRESET AND CLEAR

FUNCTION TABLE

INPUTS					OUTPUTS	
PRESET	CLEAR	CLOCK	J	K	Q	$\bar{Q}$
L	H	X	X	X	H	L
H	L	X	X	X	L	H
L	L	X	X	X	H*	H*
H	H	⊓	l	L	Q_0	$\bar{Q}_0$
H	H	⊓	H	L	H	L
H	H	⊓	L	H	L	H
H	H	⊓	H	H	TOGGLE	

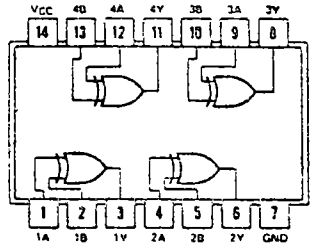


7476

7486

QUADRUPLE EXCLUSIVE-OR GATES

positive logic $Y = A \oplus B = \bar{A}B + A\bar{B}$

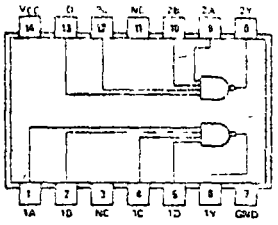


7486

7420

DUAL 4-INPUT
POSITIVE-NAND GATES

positive logic:
$Y = \overline{ABCD}$

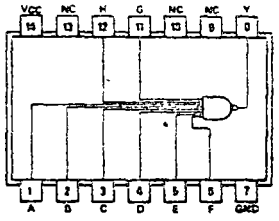


7420

7430

8-INPUT
POSITIVE-NAND GATE

positive logic:
$Y = \overline{ABCDEFGH}$

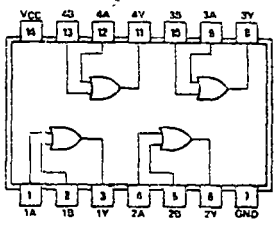


7430

7432

QUADRUPLE 2 INPUT
POSITIVE-OR GATES

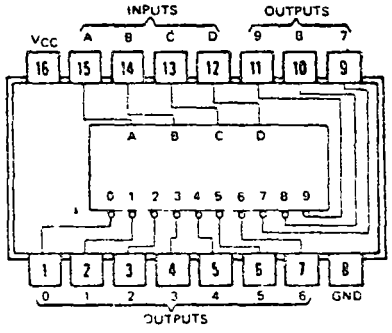
positive logic
$Y = A+B$



7432

7442

BCD-TO-DECIMAL
DECODER

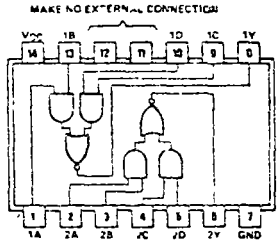


7442

7451

DUAL 2-WIDE 2-INPUT
AND-OR INVERT GATES

$Y = \overline{AB+CD}$



7451

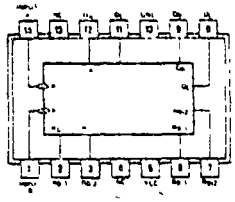
7490

DECADE COUNTERS

Output Q_A is connected to input B for BCD count

BCD COUNT SEQUENCE

COUNT	OUTPUT			
	Q_D	Q_C	Q_B	Q_A
0	L	L	L	L
1	L	L	L	H
2	L	L	H	L
3	L	L	H	H
4	L	H	L	L
5	L	H	L	H
6	L	H	H	L
7	L	H	H	H
8	H	L	L	L
9	H	L	L	H



7490

RESET/COUNT FUNCTION TABLE

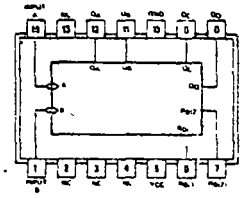
RESET INPUTS				OUTPUT			
$R_{0(1)}$	$R_{0(2)}$	$R_{9(1)}$	$R_{9(2)}$	Q_D	Q_C	Q_B	Q_A
H	H	L	X	L	L	L	L
H	H	X	L	L	L	L	L
X	X	H	H	H	L	L	H
X	L	X	L	COUNT			
L	X	L	X	COUNT			
L	X	X	L	COUNT			
X	L	L	X	COUNT			

7492

DIVIDE-BY-TWELVE COUNTER

Output Q_A is connected to input B.

COUNT	OUTPUT			
	Q_D	Q_C	Q_B	Q_A
0	L	L	L	L
1	L	L	L	H
2	L	L	H	L
3	L	L	H	H
4	L	H	L	L
5	L	H	L	H
6	H	L	L	L
7	H	L	L	H
8	H	L	H	L
9	H	L	H	H
10	H	H	L	L
11	H	H	L	H



7492

RESET/COUNT FUNCTION TABLE

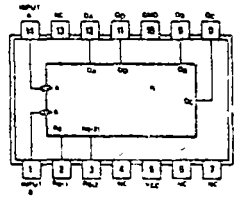
RESET INPUTS		OUTPUT			
$R_{0(1)}$	$R_{0(2)}$	Q_D	Q_C	Q_B	Q_A
H	H	L	L	L	L
L	X	COUNT			
X	L	COUNT			

7493

4-BIT BINARY COUNTERS

COUNT SEQUENCE

COUNT	OUTPUT			
	Q_D	Q_C	Q_B	Q_A
0	L	L	L	L
1	L	L	L	H
2	L	L	H	L
3	L	L	H	H
4	L	H	L	L
5	L	H	L	H
6	L	H	H	L
7	L	H	H	H
8	H	L	L	L
9	H	L	L	H
10	H	L	H	L
11	H	L	H	H
12	H	H	L	L
13	H	H	L	H
14	H	H	H	L
15	H	H	H	H



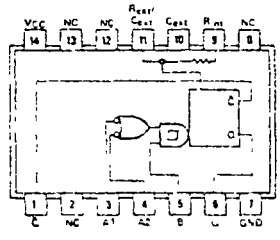
7493

74121

MONOSTABLE MULTIVIBRATORS

FUNCTION TABLE

INPUTS			OUTPUTS	
A1	A2	B	Q	$\bar{Q}$
L	X	H	L	H
X	L	H	L	H
X	X	L	L	H
H	H	X	L	H
H	↓	H	⎍	⎎
↓	H	H	⎍	⎎
↓	↓	H	⎍	⎎
L	X	↑	⎍	⎎
X	L	↑	⎍	⎎



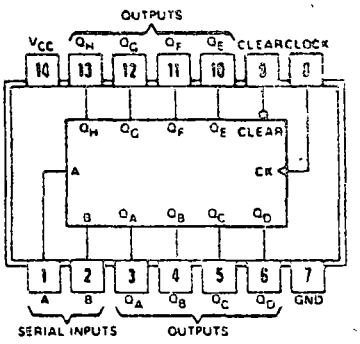
74121

74164

8-BIT PARALLEL-OUT SERIAL SHIFT REGISTERS

FUNCTION TABLE

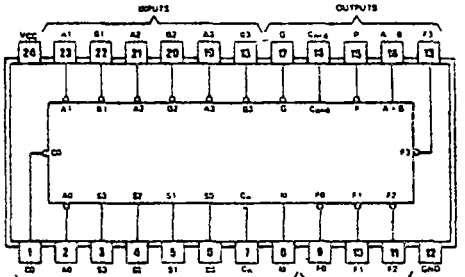
INPUTS				OUTPUTS			
CLEAR	CLOCK	A	B	Q_A	Q_B	...	Q_H
L	X	X	X	L	L		L
H	L	X	X	Q_{A0}	Q_{B0}		Q_{H0}
H	↑	H	H	H	Q_{An}		Q_{Gn}
H	↑	L	X	L	Q_{An}		Q_{Gn}
H	↑	X	L	L	Q_{An}		Q_{Gn}



74164

74181

ARITHMETIC LOGIC UNITS/ FUNCTION GENERATORS



74181

SELECTION	ACTIVE HIGH DATA		
	M = H	M = L ARITHMETIC OPERATIONS	
S3 S2 S1 S0	LOGIC FUNCTIONS	C_n = H (no carry)	C_n = L (with carry)
L L L L	F = $\overline{A}$	F A	F A PLUS 1
L L L H	F = $\overline{A+B}$	F A + B	F (A + B) PLUS 1
L L H L	F = $\overline{A}B$	F A + $\overline{B}$	F = (A + $\overline{B}$) PLUS 1
L L H H	F 0	F MINUS 1 (2's COMPL)	F ZERO
L H L L	F = $\overline{AB}$	F A PLUS A$\overline{B}$	F A PLUS A$\overline{B}$ PLUS 1
L H L H	F = $\overline{B}$	F (A + B) PLUS A$\overline{B}$	F (A + B) PLUS A$\overline{B}$ PLUS 1
L H H L	F = A ⊕ B	F A MINUS B MINUS 1	F A MINUS B
L H H H	F = A$\overline{B}$	F A$\overline{B}$ MINUS 1	F A$\overline{B}$
H L L L	F = $\overline{A}$ + B	F A PLUS AB	F A PLUS AB PLUS 1
H L L H	F = A ⊕ B	F A PLUS B	F A PLUS B PLUS 1
H L H L	F B	F (A + B) PLUS AB	F (A + B) PLUS AB PLUS 1
H L H H	F AB	F AB MINUS 1	F AB
H H L L	F 1	F A PLUS A	F A PLUS A PLUS 1
H H L H	F = A + B	F (A + B) PLUS A	F (A + B) PLUS A PLUS 1
H H H L	F = A $\overline{B}$	F (A + $\overline{B}$) PLUS A	F (A + $\overline{B}$) PLUS A PLUS 1
H H H H	F A	F A MINUS 1	F A

SELECTION	ACTIVE LOW DATA		
	M = H	M = L ARITHMETIC OPERATIONS	
S3 S2 S1 S0	LOGIC FUNCTIONS	C_n = L (no carry)	C_n = H (with carry)
L L L L	F = $\overline{A}$	F A MINUS 1	F A
L L L H	F = $\overline{A+B}$	F AB MINUS 1	F AB
L L H L	F = $\overline{A}$ B	F A$\overline{B}$ MINUS 1	F A$\overline{B}$
L L H H	F 1	F MINUS 1 (2's COMPL)	F ZERO
L H L L	F = $\overline{A \cdot B}$	F A PLUS (A + $\overline{B}$)	F A PLUS (A + $\overline{B}$) PLUS 1
L H L H	F = $\overline{B}$	F AB PLUS (A + $\overline{B}$)	F AB PLUS (A + $\overline{B}$) PLUS 1
L H H L	F = $\overline{A \oplus B}$	F A MINUS B MINUS 1	F A MINUS B
L H H H	F = A + $\overline{B}$	F A + $\overline{B}$	F (A + $\overline{B}$) PLUS 1
H L L L	F = $\overline{A}$B	F A PLUS (A + B)	F A PLUS (A + B) PLUS 1
H L L H	F = A ⊕ B	F A PLUS B	F A PLUS B PLUS 1
H L H L	F B	F A$\overline{B}$ PLUS (A + B)	F A$\overline{B}$ PLUS (A + B) PLUS 1
H L H H	F = A + B	F A + B	F (A + B) PLUS 1
H H L L	F 0	F A PLUS A	F A PLUS A PLUS 1
H H L H	F = A$\overline{B}$	F AB PLUS A	F = AB PLUS A PLUS 1
H H H L	F = AB	F A$\overline{B}$ PLUS A	F A$\overline{B}$ PLUS A PLUS 1
H H H H	F A	F A	F A PLUS 1

*Each bit is shifted to the next more significant position

altamente versátil; el 74181 un elemento aritmético muy poderoso (unidad lógico-aritmética o ALU); el 7400 corresponde a la lógica combinacional más usada (compuertas NAND).

Cada uno de los elementos aritméticos básicos se puede adquirir como un circuito integrado. En combinación con otros elementos se integra un ALU (tal como el 74181). Los circuitos más sofisticados son aquéllos de los microprocesadores en donde, en un sólo 'chip' se incluye toda la lógica de una unidad de proceso central.

7.1. Suma Binaria. El diagrama de un SC se muestra en la figura 1.

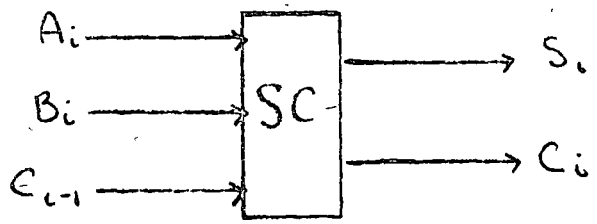


Figura 1. Sumador completo.

La tabla de verdad del SC es la siguiente:

C_{i-1}	A_i	B_i	S_i	C_i
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

El SC tiene tres entradas: dos (A_i y B_i) corresponden al i-ésimo bit de los dos sumandos (A y B) y una correspondiente al bit de acarreo de la (i-1) ésimo etapa de suma. Tiene dos salidas que corresponden a ésimo bit de suma (S_i) y al ésimo bit de acarreo (C_i).

In the above example each adder has to be modified to handle three inputs: A, B, and the carry C. Such a modified circuit, Figure 7-3, is called a full adder.

Although the full adder can be constructed in many different ways, it always performs the same function -- it adds two bits and a previous carry and generates a sum, as well as a carry. This circuit is used in calculators, computers, and many other arithmetic circuits to perform not only addition, but also multiplication, division, and subtraction. All these operations can be performed by utilizing special algorithms and the full adder circuit.

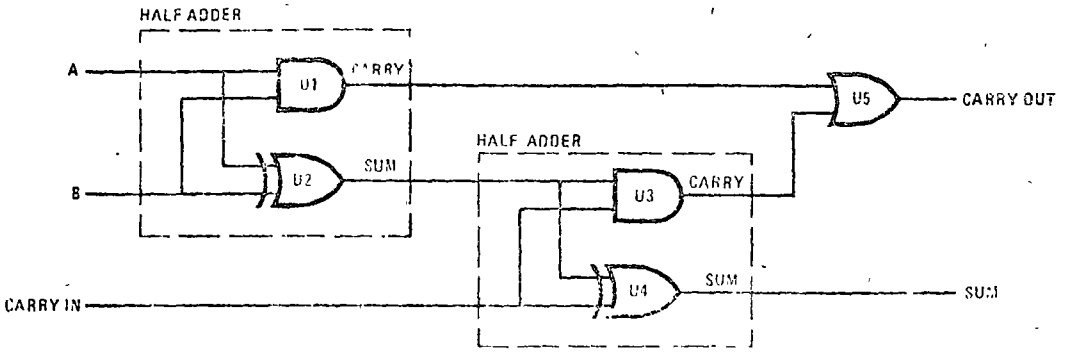


Figure 7-3. Full Adder.

PARALLEL ADDER

The parallel adder consists of several full adder circuit stages that are interconnected so that the carry output of one stage becomes the carry input to the succeeding stage, as shown in Figure 7-4. Therefore, a four stage parallel adder will handle the propagation of all carries and can be used to add any two 4-bit binary numbers.

Several significant features in the parallel adder should be understood. First, even though it is called parallel, the adder works in a sequential manner. If we are to add the numbers:

1 1 1 1	C	(carries)
0 1 1 1	A	(augend)
+ 1 0 0 1	B	(addend)
1 0 0 0 0	S	(sum)

a carry is generated by each stage of the addition. The first adder U1 must complete the addition of A_1 and B_1 in order to generate the carry C_1 input to the second stage U2. Therefore, U2 cannot correctly perform its part of the addition sequence until U1 has completed adding and generating carry C_1. Likewise, U3 has to wait for U2 to generate carry C_2 and so on.

The carries pass sequentially through all stages of the parallel adder and the last sum output is not correct until the last carry is generated. This carry propagation is similar to propagation through the stages of a ripple counter. Therefore, the adder in Figure 7-4 is referred to as a parallel adder with ripple carry

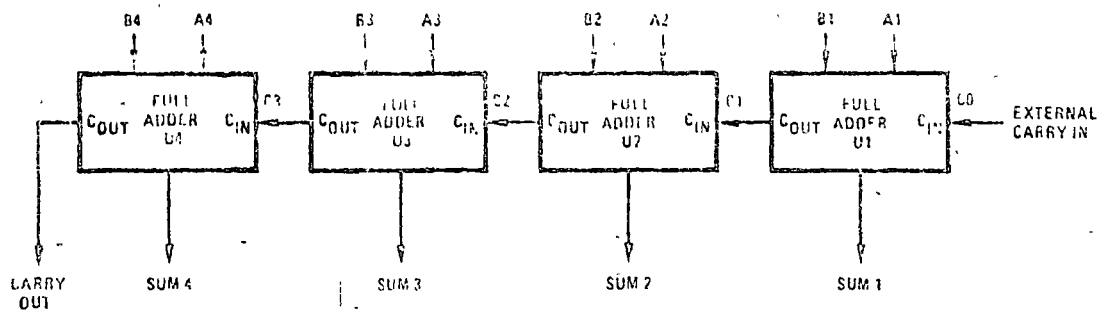


Figure 7-4. Parallel 4-bit Adder.

Notice that every stage of this adder performs an addition as soon as its A and B inputs are present, but the outputs of any stage may not be correct until a carry input has been processed. Thus, an iterim incorrect sum output may be generated that cannot be allowed to enter into other circuits of the system. For this reason parallel adders often have sum output gates that are disabled until the adder has had sufficient time to generate the correct sum signals.

A parallel adder with ripple carry requires a relatively long time for all the carries to be generated and processed. Consequently, these adders are considered to be slow (although they are more than fast enough for most applications outside large computers). In the previous example of the addition of 0111 and 1001 the total time to complete the addition can be calculated as follows: if each stage requires 30 ns from 4-bit adder will require 4 x 30 ns = 120 ns. If, however, the following two numbers are added,

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

a carry will be generated only by the last stage and no stage will have to wait on the previous one to generate a carry. Consequently, all additions will be performed in parallel and the total time to complete the addition will be equal to that for one stage, or 30 ns.

This difference in addition time implies results of some additions can be sampled sooner than those of others. In most systems, however, the adder output is sampled synchronously and the clock pulses which do the sampling cannot occur more frequently than at 120 ns intervals to allow time for the longest possible addition. Thus, no decrease of cycle time can be realized.

On the other hand, if the system is such that the adder outputs can be sampled asynchronously, any sum can be output as soon as it becomes available. Unfortunately, there is no easy way to tell when the addition is actually complete – the adder itself has no signal that indicates this. It is possible to construct what is known as carry-complete logic from auxiliary gates, but this increases the complexity of the adder and the auxiliary gates themselves introduce an additional delay (resulting in an increase of the maximum cycle length). The net benefits of carry-complete logic are no greater than those of a different kind of adder, called the look-ahead carry adder. This circuit is described later in this chapter.

SERIAL ADDER

The basic serial adder is structured around a single full adder circuit that adds each pair of bits sequentially. The auxiliary circuitry that constitutes the rest of the serial adder is arranged so that the augend word is entered from an external memory storage into the A shift register, Figure 7-5, and the addend is entered into the B register — both with the least significant bits on the right. The least significant bit from each register is shifted into the full adder. Then, the sum is shifted into the sum register and the next two bits from the A and B registers are shifted to the full adder on the same clock pulse. If the first addition produces a carry, the carry is stored in FF1 and becomes an input to the full adder during the next addition. A pair of binary words of any length can be added in a series of such additions, starting with the least significant bit and ending with the most significant, clocking all shift registers with the same clock signal.

Several modifications can be added to this basic serial adder for different applications. If it is desired to add several different numbers in sequence to the same augend, the

Figura 3.

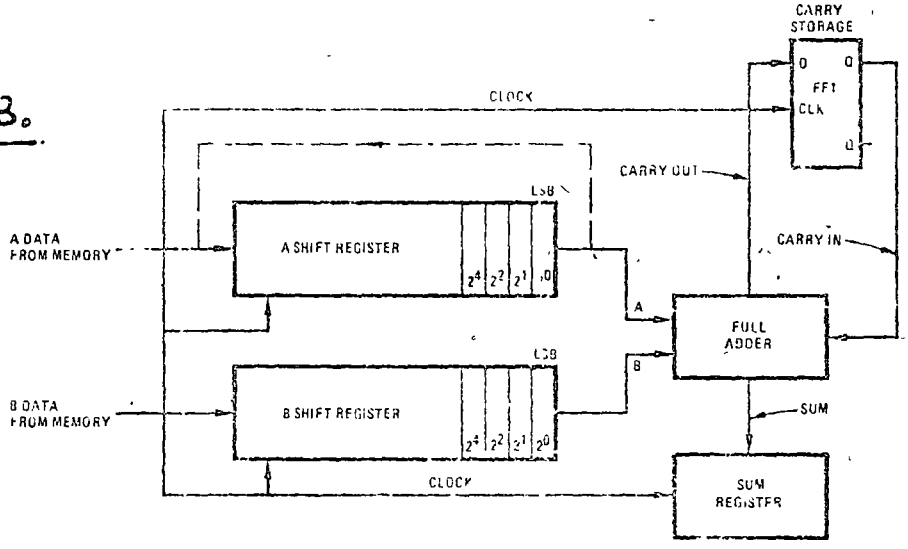


Figure 7-5. Serial Adder.

Figura 4.

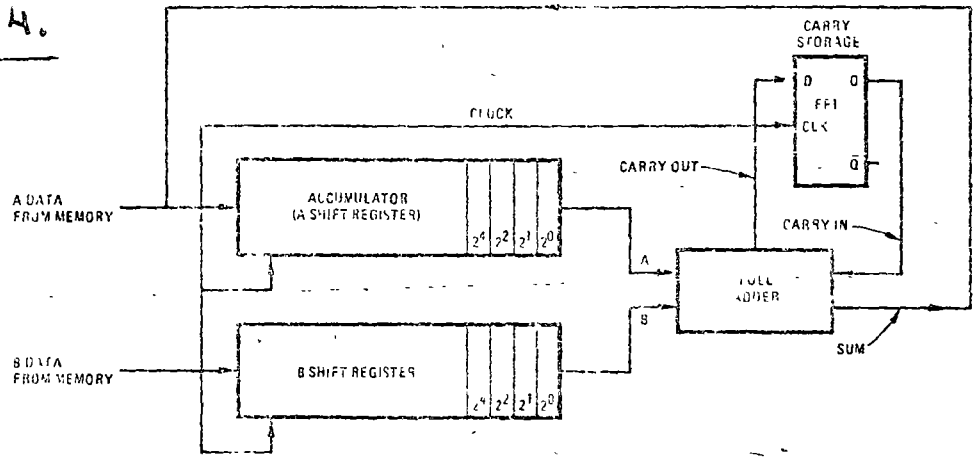


Figure 7-6. Serial Adder with Accumulator.

Conectando 4 S.C. en serie se pueden sumar dos palabras de 4 bits como se muestra en la figura 2.

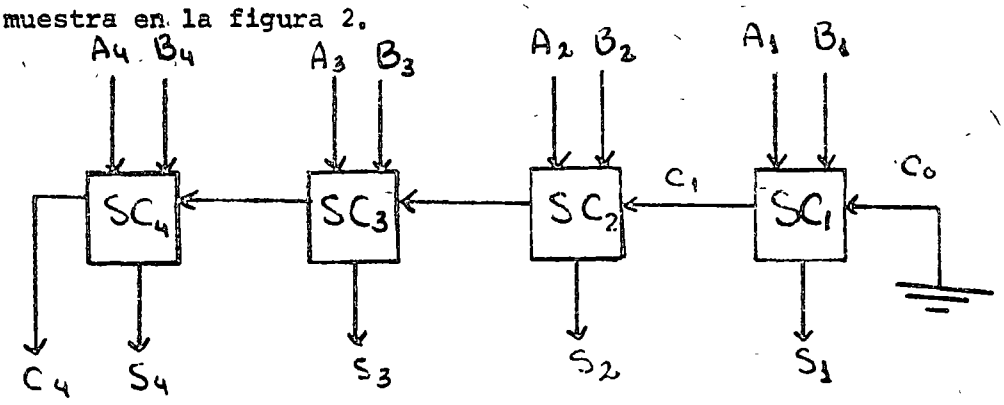


Figura 2. Sumador paralelo de 4 bits.

La alternativa al circuito anterior es la suma en serie. En este caso, los sumandos son alimentados desde dos registros de corrimiento a un solo SC y la suma de 'n' bits toma 'n' períodos de reloj. Dos circuitos que implementan este tipo de suma se muestran en la figura 3 y 4.

En la figura 3, el resultado se almacena en un registro diferente de los registros A y B, mientras que en la figura 4, el resultado se realimenta al registro A. Este es el caso delacumulador de las llamadas 'computadorasde una dirección' (tales como los microprocesadores INTEL 8080, MOTOROLA 6800, la mini PDP-8, etc.)

7.1.1 El concepto 'carry look-ahead'. De la figura 2 se ve que, para que se genere el acarreo C_4, se tiene que suceder el acarreo C_3, C_2, C_1. El tiempo máximo para que esto ocurra el de 4 ΔT, en donde Δt es el tiempo de propagación a través de un SC. Por ello, la velocidad de suma del circuito es de 5 ΔT. Para un par de sumandos de 16 bits, el retardo asociado a su suma es 6 ΔT.

Para que se produzca un acarreo de SC_1 debe cumplirse la ecuación:

$$C_1 = (\bar{A}_1 B_1 + A_1 \bar{B}_1)\, C_o + A_1 B_1$$

En general:

$$C_n = (\bar{A}_n B_n + A_n \bar{B}_n)\, C_{n-1} + A_n B_n$$

$$C_n = (A_n + B_n)\, C_{n-1} + A_n B_n$$

Si hacemos:

$$G_n = A_n B_n$$

y:

$$P_n = (A_n + B_n)$$

tenemos que: $C_n = G_n + P_n C_{n-1}$

Para: $N=1$

$$C_i = G_i + P_i C_o$$

Podemos entonces hacer la siguiente consideración:

$$C_2 = G_2 + P_2 C_1 = G_2 + P_2 (G_1 + P_1 C_o)$$

$$C_3 = G_3 + P_3 C_2 = G_3 + P_3 \left[G_2 + P_1 (G_1 + P_1 C_o) \right]$$

$$C_4 = G_4 + P_4 C_3 = G_4 + P_4 \{ G_3 + P_3 \left[G_2 + P_1 (G_1 + P_1 C_o) \right] \}$$

en donde C_1 C_2 C_3 y C_4 están en función de C_o y por ello en sólo Δt habremos generado la suma. Compárese Δt contra 4 Δt. En la figura 5 se muestra este sumador. (7-9).

Continuar en esta forma es inprâctico, pues el número de compuertas aumenta desmedidamente.

Sin embargo, es posible considerar a este sumador de 5 bits como una 'sección' de manera que cada grupo de 4 SC sea una sección y 4 de dichas secciones pueden conectarse comu se muestra en la figura 6.

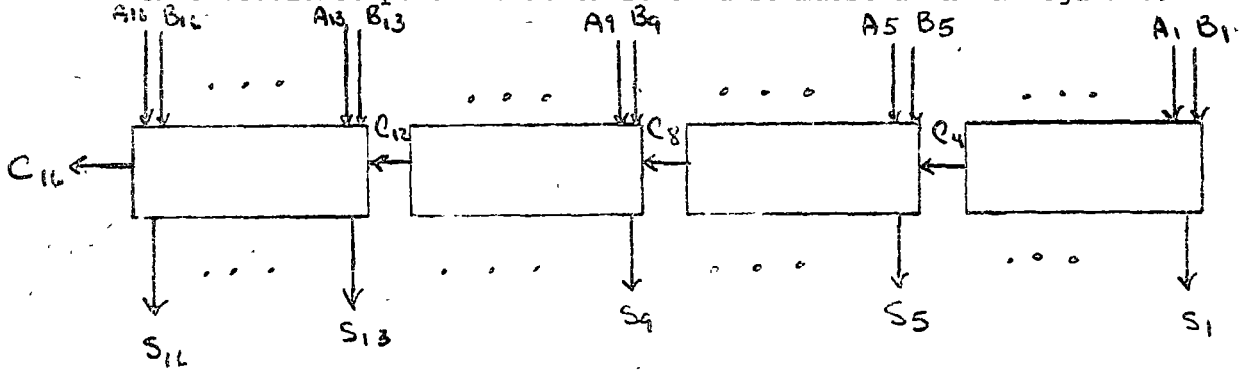


Figura 6.

El tiempo de suma de 16 bits es de 4 Δt (compárese con 16 Δt).

Figura 5.

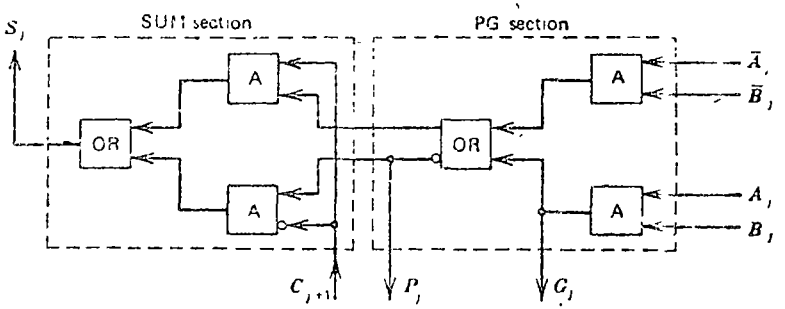


FIGURE 11.4. Full-adder circuit.

The sum equations obviously all have the same form so we shall implement them with a special form of full-adder circuit, as shown in Fig. 11.4. For later convenience, we shall divide this circuit into two sections, the *PG section* and the *SUM section*, as shown.

The carry-in terms to the sum circuits will be developed as shown in the following equations:

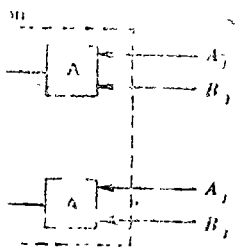
$$C_{63} = G_{63} \vee (P_{63} \wedge C_{61}) \tag{11.14}$$

$$\begin{aligned} C_{62} &= G_{62} \vee (P_{62} \wedge C_{63}) \\ &= G_{62} \vee (P_{62} \wedge G_{63}) \vee (P_{62} \wedge P_{63} \wedge C_{61}) \end{aligned} \tag{11.15}$$

$$\begin{aligned} C_{61} &= G_{61} \vee (P_{61} \wedge C_{62}) \\ &= G_{61} \vee (P_{61} \wedge G_{62}) \vee (P_{61} \wedge P_{62} \wedge G_{63}) \vee (P_{61} \wedge P_{62} \wedge P_{63} \wedge C_{61}) \end{aligned} \tag{11.16}$$

These three equations are implemented in the carry look-ahead (CLA) unit, shown in Fig. 11.5. (This unit also implements some additional equations, which will be discussed shortly.) Since the CLA unit may be used with any set of four bit-positions, we have used generalized subscripts in Fig. 11.5. For implementation of the above equations, $j = 63$.

The interconnection of the CLA unit with the adder units for bits 60–63 is shown in Fig. 11.6. Noting that each unit (SUM, PG, CLA) is a second-order circuit, we can analyze the propagation delays. Let us consider the worst case, a carry generated in bit 63 and propagated through to bit 0. The carry is generated in PG_{63} with a delay of $2\Delta t$, propagates through the CLA to C_{61} in $2\Delta t$, and propagates through SUM_{60} to develop S_{60} in $2\Delta t$, for a total delay of $6\Delta t$. This compares with a delay of $8\Delta t$ for a ripple through four bits. This is only a small improvement; but [illegible] the logic of the design as we shall see



G_j

uit.

so we shall implement
n in Fig. 11.4. For later
ons, the *PG section* and

eloped as shown in the

(11.14)

(11.15)

$P_0 \wedge P_{0,0} \wedge C_{-1})$ (11.16)

look-ahead (CLA) unit,
e additional equations
may be used with any
subscripts in Fig. 11.5.

ier units for bits 60-63
PG, CLA) is a second-
ed to apply to bit 60
propagated through the
to develop S_{60} in $2\Delta t$
of $8\Delta t$ for a
nt, but this is just the

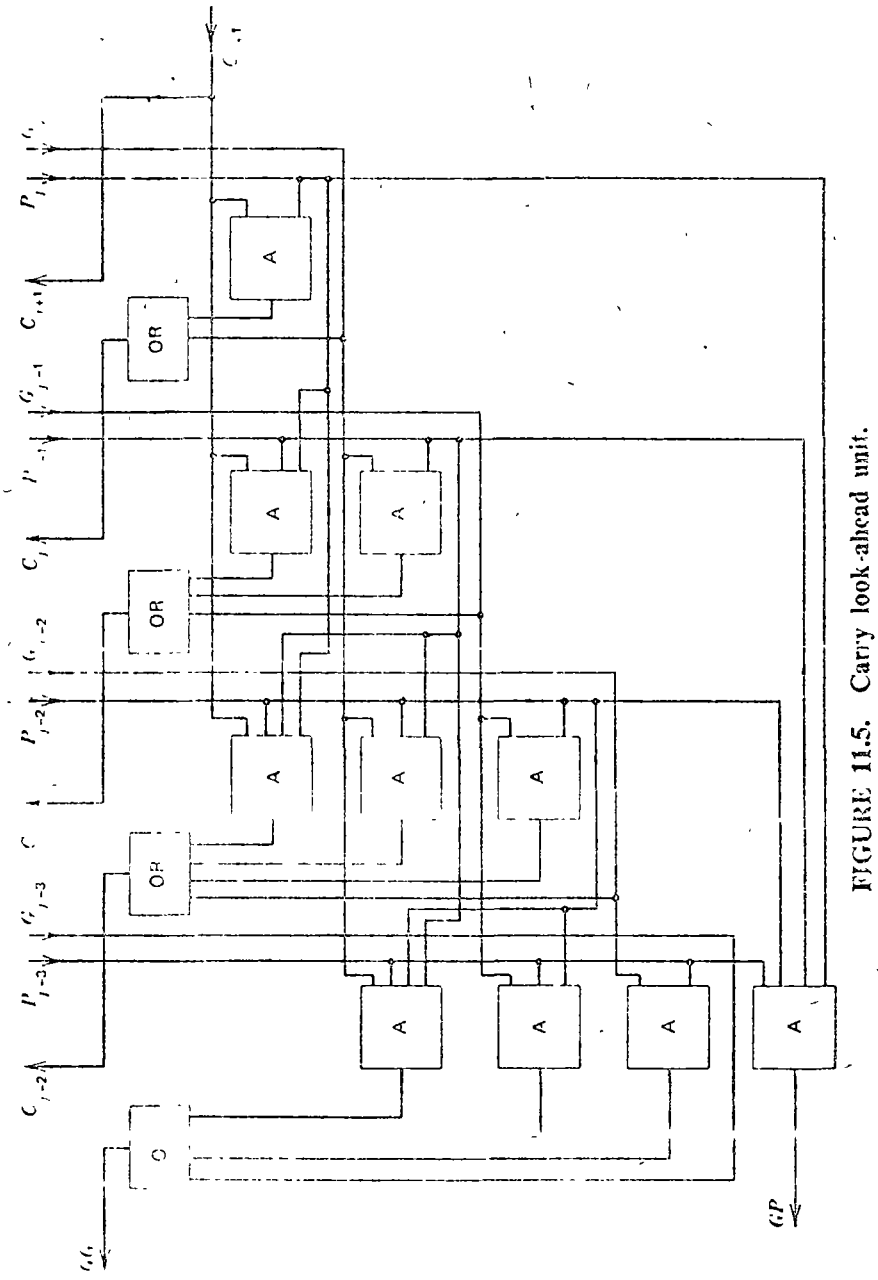


FIGURE 11.5. Carry look-ahead unit.

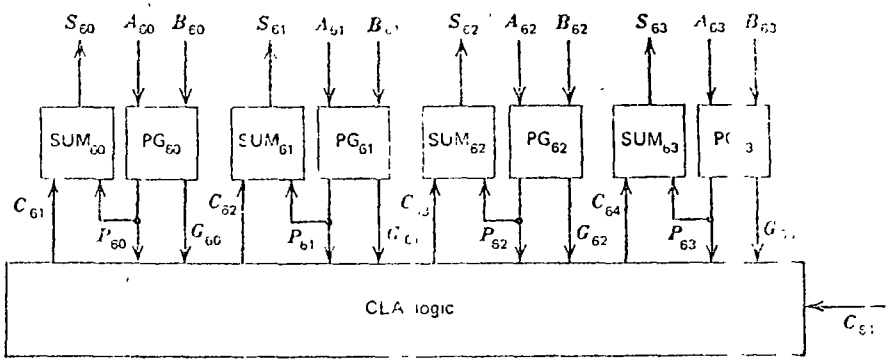


FIGURE 11.6. Complete adder for bits 60–63.

If we examine Eqs. 11.14, 11.15, 11.16, we see that they are iterative in form, and there is no reason why we could not continue the same process to write equations for C_{56}, C_{59}, etc. These equations would also be second-order, so the CLA unit could be extended to cover more bits, with no increase in delay. However, as we increase the number of bits the size and number of gates also increases. C_{60} requires 4-input gates, C_{59} would require 5-input gates, C_{59} would require 6-input gates, etc. So the number of bits the CLA unit can cover is limited by the fan-in capability of our gates. Circuit technology makes it generally impractical to go beyond about eight bits in the basic CLA unit.

11.5. GROUP CARRY LOOK-AHEAD

As the next step in our design, we shall divide the 64-bit adder into 4-bit groups, bits 0–3 comprising group 0, bits 4–7, group 1, etc. We then define *group generate*, GG, and group *propagate*, GP, terms, as shown below for group 15 (bits 60–63). The group generate term corresponds to the situation

$$GG_{15} = G_{63} \vee P_{63} \wedge G_{62} \vee P_{63} \wedge P_{62} \wedge G_{61} \vee P_{63} \wedge P_{62} \wedge P_{61} \wedge G_{60} \tag{11.17}$$

$$GP_{15} = P_{63} \wedge P_{62} \wedge P_{61} \wedge P_{60} \tag{11.18}$$

where a carry has been generated somewhere in the group and all more significant positions are in the propagate condition, so that the carry propagates on out of the group. The group propagate corresponds to the condition where all bits in the group are in the propagate condition, so that a carry into the group should pass straight through the group. Note that these terms are implemented by the left-most five gates in the CLA unit in Fig. 11.5.

Next we note that there is a carry out of the group if a carry is ge

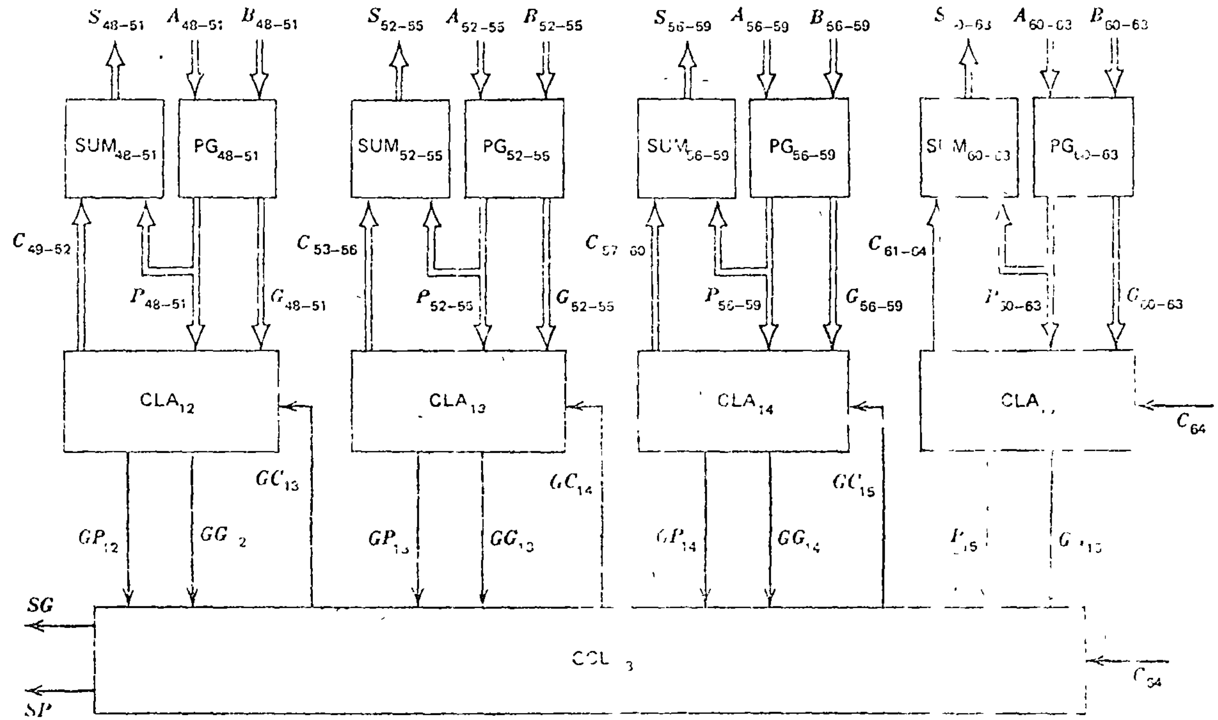


FIGURE 11.7. 16-Bit adder with group carry look-ahead.

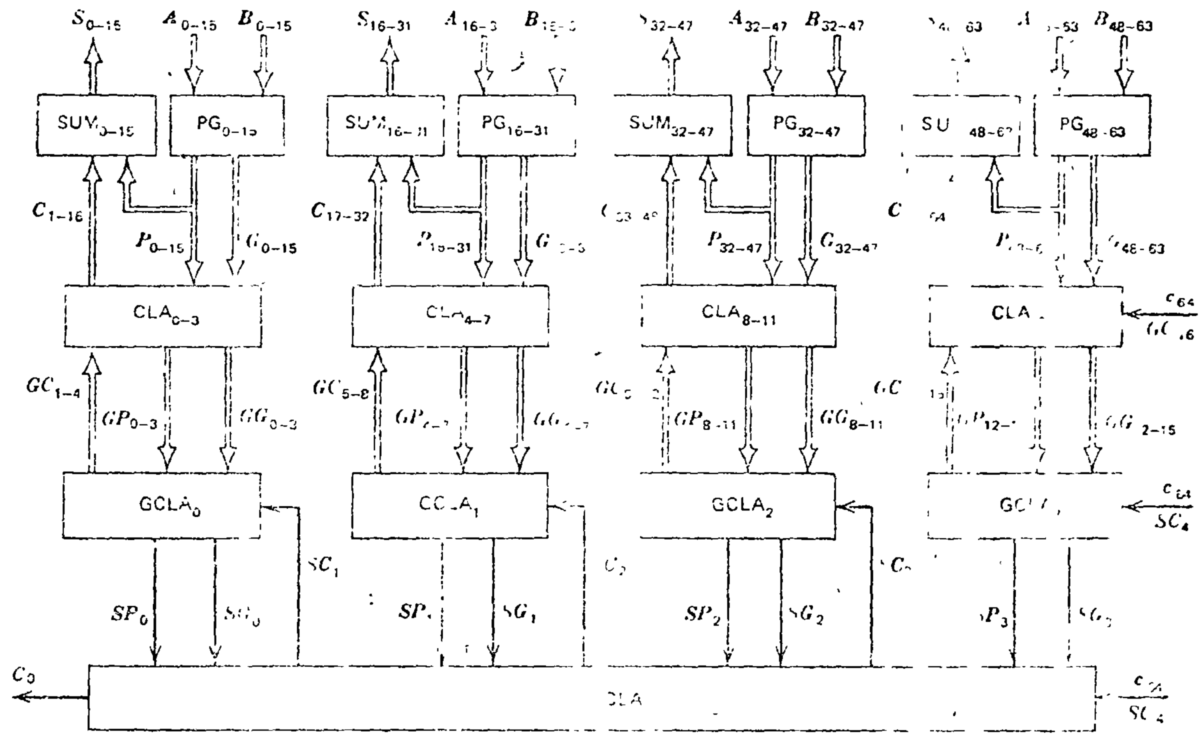


FIGURE 11.8. 64-Bit adder with three levels of carry look ahead.

Existen varias formas de intercambiar complejidad con velocidad. Para una discusión más completa del principio de 'look-ahead' se pueden consultar las referencias [3] y [4].

7.2 Resta Binaria. La resta binaria se puede considerar como un complemento de la suma. Para ello puede utilizarse el restador completo (RC) que se muestra en la figura 7. |7-11|. Sin embargo, en la mayoría de las aplicaciones prácticas se sigue un procedimiento diferente que permite tratar a la suma y la resta como una suma algebráica.

7.2.1. Representación complementada. Para lograr el tratamiento general de dos sumandos con signo se acostumbra usar, convencionalmente, un bit que representa el signo (1 para (-) y 0 para (+)). De esta forma, en una palabra de 8 bits, tenemos:

0.000 1000 = +8

1.000 1000 = -8

Básicamente existen tres tipos de representación:

a) Signo y magnitud.

b) Complemento a 1.

c) Complemento a 2.

La lógica de estas representaciones, así como los circuitos que implementan cada caso se muestran en las siguientes páginas (7-12,13,14,15).

El tipo de representación que se elige depende simplemente de las preferencias del diseñador. Sin embargo, debe hacerse notar que es más sencillo trabajar con aritmética complemento a 2.

7.3. Multiplicación. Las multiplicaciones binarias pueden hacerse utilizando tres métodos básicos:

a) Sumas repetidas

b) Sumas y corrimientos

c) Con tablas de productos parciales

El primer método es el más elemental posible y consiste en sumar el multiplicando "multiplicador" veces consigo mismo. Este método es el más lento de todos y no lo discutiremos aquí.

the carry logic to do some preliminary ANDing of the G and P terms. These additional gates develop what are referred to as first level auxiliary functions.

All of the above methods of optimization are discussed in more detail in the references listed at the end of the chapter and in IC manufacturers' literature.

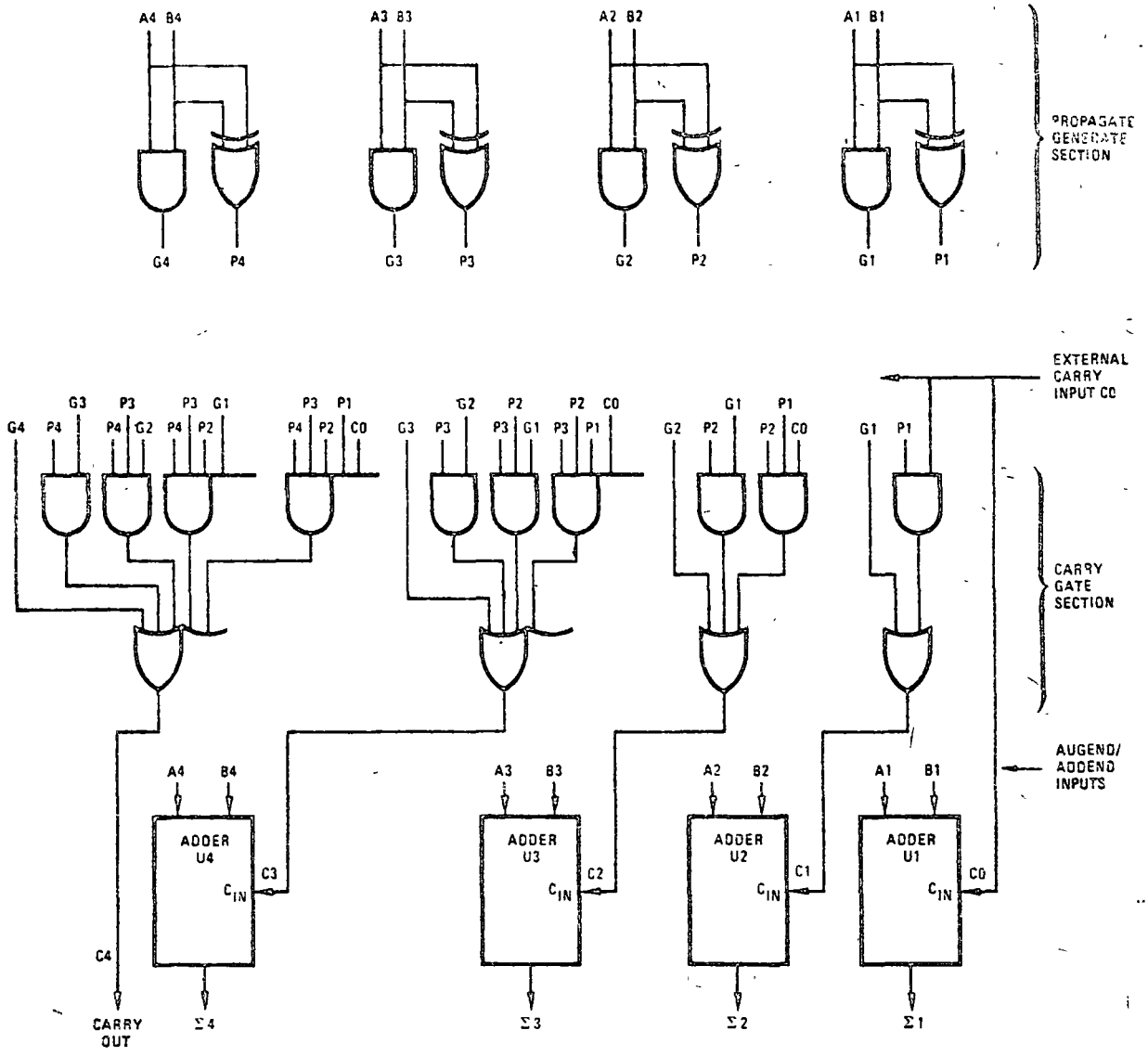


Figure 7-7. Look-ahead Carry Adder.

BINARY SUBTRACTION

Whereas in binary addition there were only four simple algorithms for the addition of positive integers, subtraction can be done in several different ways. First, to perform direct subtraction of one binary number from another, rules or algorithms can

HALF AND FULL SUBTRACTER

To implement the first set of subtraction algorithms above (direct binary subtraction), a **half subtracter** and a **full subtracter** can be constructed that correspond to the half and full adders for addition. Figure 7-8 shows one version of a full subtracter, but just as in adder circuits, various types of gates can be used to achieve the same result. In fact, the adder and subtracter circuits are so similar that it is also possible to construct a combined adder/subtracter circuit, with an ability to switch from one operation to the other by means of a single control signal.

A full subtracter circuit can be used in much the same manner as the full adder. That is full subtracters can be connected in multiple parallel stages, or a single subtracter can be used in a serial circuit. Look-ahead carry type logic can also be used with subtracters.

Figura 7.

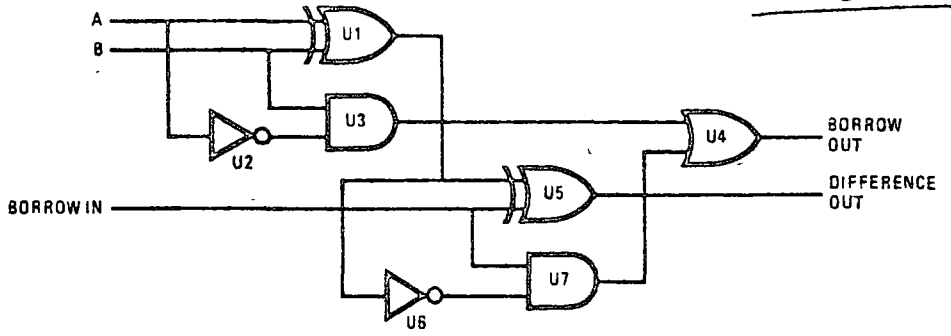


Figure 7-8. Full Subtracter.

In practice, however, the full subtracter circuit is not used very often because most calculators and computers perform subtraction by the addition of a complement, using only full adder circuits for all addition and subtraction operations. This reduces the amount of logic gating required in any one circuit, increases speed of operation, and simplifies programming.

ADDITION AND SUBTRACTION WITH COMPLEMENTS

Because subtraction can be performed by the addition of a complement, in typical computer logic the difference between addition and subtraction looses its clear distinction. A typical computer simply recognizes positive and negative numbers and adds these numbers by the method it has been designed to use.

Up to this point, two specific constraints have been implicit in addition and subtraction: the numbers all had to be positive and, in subtraction, the minuend had to be larger than the subtrahend. If negative numbers are allowed, and some variations on the algorithm for subtraction by adding a complement are introduced, actual methods of subtraction and addition in computers can now be described.

To distinguish between positive and negative numbers, a simple notation is usually used: if a number is positive, its + sign is indicated by a 0 in the leftmost digit of the number, if the number is negative, the - sign is indicated by a 1. For the purposes of

Table 7-1. Addition and Subtraction by 1's Complements.

EXAMPLES	ALGORITHMS
TWO NUMBERS OF SAME SIGN	
(Positive) (Negative) +13 0.01101 +11 0.01011 +24 0.11000	*To augend magnitude add addend magnitude,* *retain existing sign*
-13 1.10010 -11 1.10100 -24 1.00110 → 1 1.00111	*To 1's complement of augend magnitude add 1's complement of addend magnitude,* *add end-around carry,* *retain existing sign (answer in 1's complement form)*
TWO NUMBERS OF OPPOSITE SIGN	
-13 1.10010 +11 0.01011 +11 0.01011 -13 1.10010 -2 1.11101 -2 1.11101	*To magnitude of augend (in 1's complement form, if negative) add magnitude of addend (in 1's complement form, if negative).*
TWO NUMBERS OF OPPOSITE SIGN	
+13 0.01101 -11 1.10100 -11 1.10100 +13 0.01101 +2 0.00001 +2 0.00001 → 1 → 1 0.00010 0.00010	*To magnitude of augend (in 1's complement form, if negative) add magnitude of addend (in 1's complement form, if negative);* *add end-around carry.*

Note All examples shown in the table are defined to be additions. Subtraction is defined to be for example, +13 − (−11). Subtraction requires the complementing of the subtrahend in a true/complement circuit, such as shown in Figures 7-9 and 7-10.

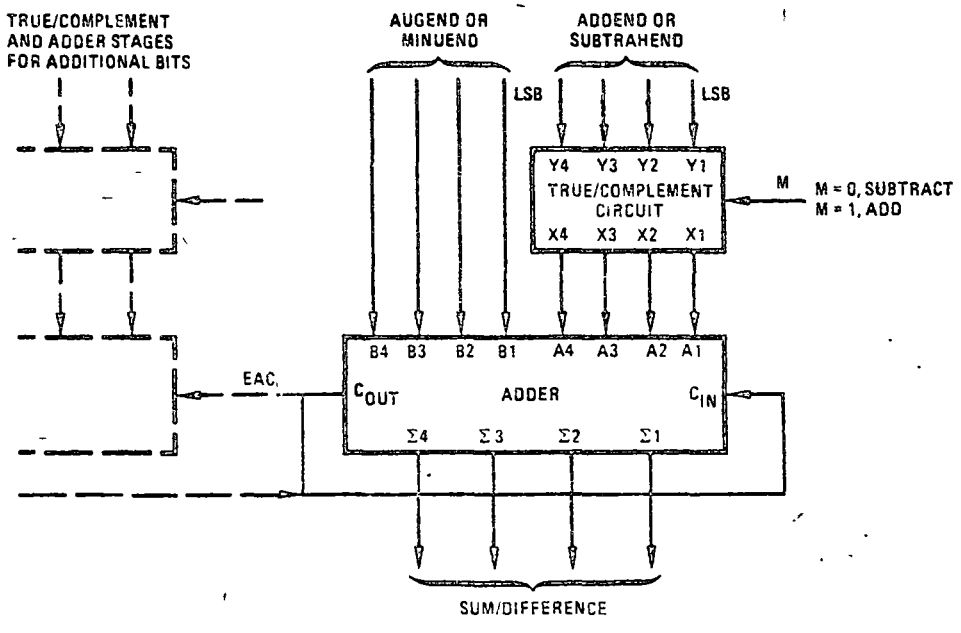


Figure 7-9. Typical 1's Complement Adder/Subtracter.

Table 7-2. Addition and Subtraction by 2's Complements.

EXAMPLES				ALGORITHMS
TWO NUMBERS OF SAME SIGN				
(Positive)		(Negative)		
+13	0.01101			*To augend magnitude add addend magnitude;*
+11	0 01011			*retain existing sign*
+24	0.11000			
		-13	1 10011	*To 2's complement of augend magnitude add 2's complement*
		-11	1 10101	*of addend magnitude, neglect last carry;*
		-24	1.01000	*retain existing sign*
TWO NUMBERS OF OPPOSITE SIGN – AUGEND MAGNITUDE LARGER				
+13	0.01101			*To magnitude of augend add 2's complement of addend,*
-11	1 10101			*add sign bits also,*
+2	0.00010			*but neglect carry from sign bits.*
		-13	1 10011	*To 2's complement of augend magnitude add magnitude of*
		+11	0 01011	*addend, add sign bits a so,*
		-2	1.11110	*but neglect carry from sign bits.*
TWO NUMBERS OF OPPOSITE SIGN – ADDEND MAGNITUDE LARGER (OR EQUAL)				
-11	1.10101			*To 2's complement of augend magnitude add magnitude of*
+13	0 01101			*addend, add sign bits also,*
+2	0.00010			*but neglect carry from sign bits*
		+11	0 01011	*To magnitude of augend add 2's complement of addend,*
		-13	1 10011	*add in sign bits also,*
		-2	1 11110	*but neglect carry from sign bits*

Note See footnote to Table 7 1 for a definition of addition and subtraction

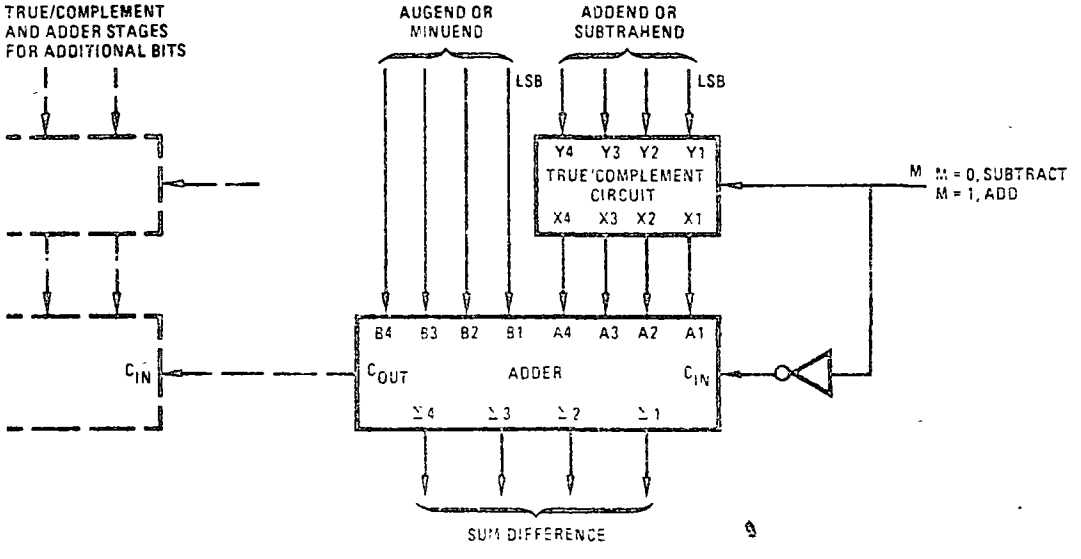


Figure 7-10. Typical 2's Complement Adder/Subtracter.

Table 7-3. Addition and Subtraction by Sign and Magnitude.

EXAMPLES				ALGORITHMS
TWO NUMBERS OF SAME SIGN				
(Positive)		(Negative)		
+13	0 01101	-13	1 01101	*To magnitude of augend add magnitude of addend*
+11	0 01011	-11	1 01011	
+24	0 11000	-24	1 11000	*retain existing sign*
TWO NUMBERS OF OPPOSITE SIGN – AUGEND MAGNITUDE LARGER				
+13	0 01101	-13	1 01101	
-11	1.01011	+11	0 01011	
+2		-2		
	01101		01101	*To magnitude of augend add 1's complement of addend magnitude.*
	10100		10100	
	00001		00001	
	↪1		↪1	*end-around carry*
	00010		00010	
	0 00010		1 00010	*retain sign of augend*
TWO NUMBERS OF OPPOSITE SIGN – ADDEND MAGNITUDE LARGER (OR EQUAL)				
+11	0 01011	-11	1 01011	
-13	1.01101	+13	0 01101	
-2		+2		
	01011		01011	*To magnitude of augend add 1's complement of addend magnitude*
	10010		10010	
	11101		11101	*(there is no end-around carry),*
	1.00010		0.00010	*complement result and retain sign of addend*

Note See footnote to Table 7-1 for a definition of addition and subtraction

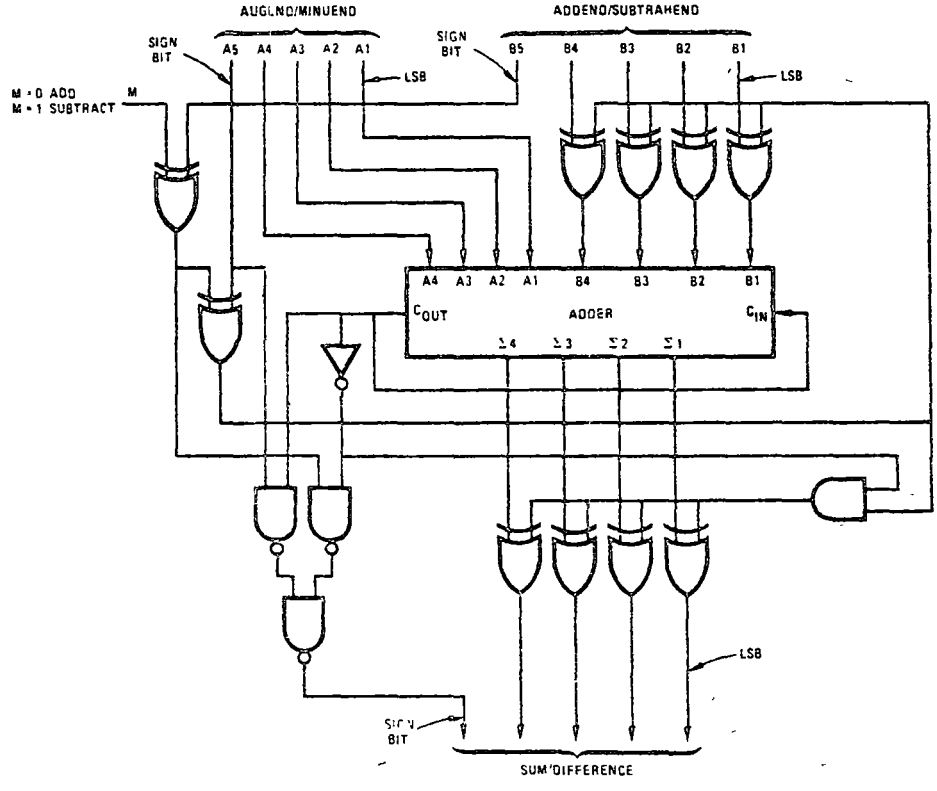


Figure 7-11. Typical Sign and Magnitude Adder/Subtracter.

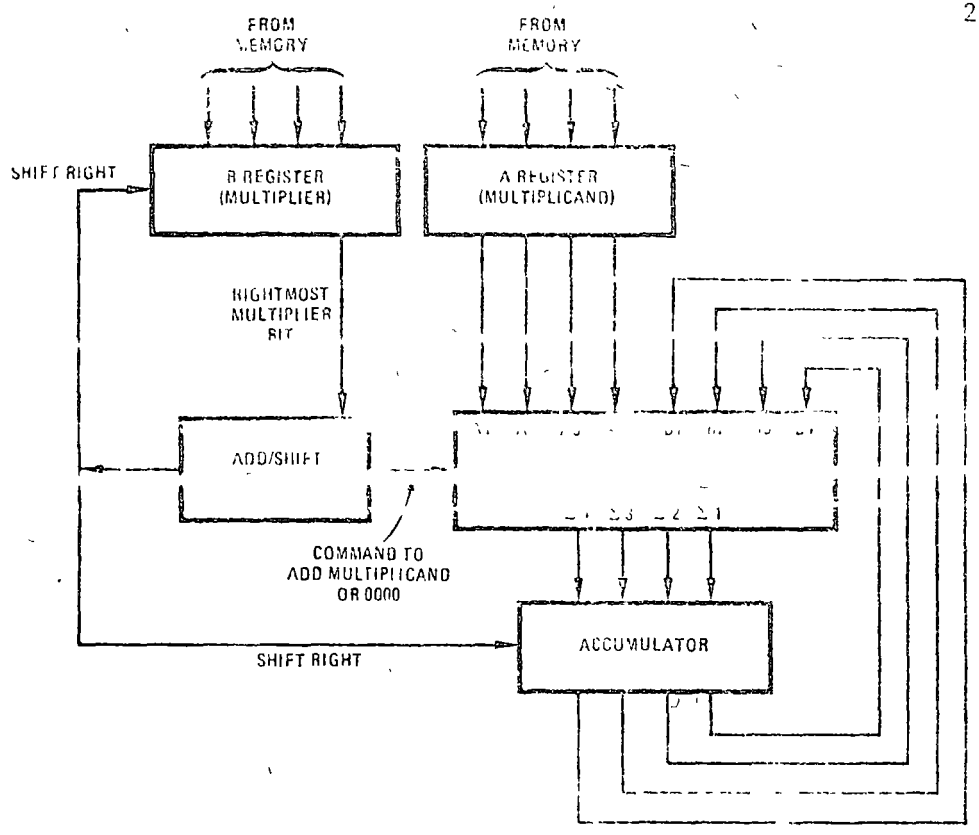


Figura 8.

STEP		CONTENTS OF B REGISTER (MULTIPLIER)	CONTENTS OF ACCUMULATOR	
1.	ADD 1101	1 0 1 1	0 1 1 0 1	*first partial product*
			space for carry bit	
2.	SHIFT RIGHT	1 1 0 1	0 1 1 0 1	
3.	ADD 1101	0 1 0 1	1 0 0 1 1 1	*sum of first two partial products*
4	SHIFT RIGHT	0 1 0 1	0 1 0 0 1 1 1	
5.	ADD 0000	1 0 1 0	0 1 0 0 1 1 1	*sum of first three partial products*
6	SHIFT RIGHT	0 1 0 1	0 0 1 0 0 1 1 1	
7.	ADD 1101	1 0 1 0	1 0 0 0 1 1 1 1	*final product*

Figure 7-15 Binary Multiplier.

El segundo método es el más empleado, y consiste en implementar, para números binarios, el algoritmo que se usa normalmente en operaciones efectuadas con "lápiz y papel". En binario, la multiplicación se reduce a sumar o correr el resultado parcial, por lo que este algoritmo es atractivo y sencillo.

Supongamos el producto:

```
  17
X 42
```

El primer producto parcial arroja:

```
  34
```

y el segundo 680,

de manera que el producto final es: 714.

Al multiplicar en binario se sigue el mismo proceso. Supongamos el producto:

```
   0001 0000
   0000 1011
   0001 0000
  00010 000
 0001000 0
 0001011 0000
```

Como se vé, en binario un 1 en el multiplicador implica "suma" y corrimiento del multiplicando; un cero simplemente implica un corrimiento del multiplicando. (Recordemos que un de un número binario es equivalente a una multiplicación por 2).

Un circuito que realiza una multiplicación de dos palabras de 4 bits se muestra en la figura 8. (7-22)

Para multiplicar por el método (C) debe hacerse la siguiente consideración: cuando se efectúa el producto de dos números en alguna base (diferente a 2) es necesario recordar todos los posibles productos de todas las combinaciones de los dígitos de la base. Por ejemplo, si la base es 3, hay que 'recordar' los productos

0 X 0=0	1 X 0=0	2 X 0=0
0 X 1=0	1 X 1=1	2 X 1=2
0 X 2=0	1 X 2=2	2 X 2=4

Teniendo lo anterior en mente, es posible obtener el producto $52_{10} \times 5_{10} = 1221_3 \times 12_3$ de la siguiente manera:

$$\begin{array}{r} 1221_3 \\ 12_3 \\ \hline 10212_3 \\ 1221_3 \\ \hline 100122_3 \end{array}$$

Lo que se pretende ilustrar es que para cualquier caso no trivial (en donde la base es mayor que 2) hay que 'recordar' ciertos productos parciales. De hecho, lo que se hace es almacenar en una tabla en memoria (ROM) todas las posibles combinaciones de productos para dos dígitos de la base escogida. En la práctica no se seleccionan bases como las del ejemplo anterior , sino bases multiplo de 2 (8,16, etc) con una importante excepción que es la base decimal (BCD).

Un ROM típico de 256 palabras de 8 bits tiene un tiempo de acceso de 500ns (nanosegundos). Para efectuar la multiplicación de dos números hexadecimales lo único que tiene que hacerse es almacenar los 256 posibles productos en ROM y luego sumar los productos parciales obtenidos de la tabla. Por ejemplo, si se desea multiplicar los números:

$$\begin{array}{rr} & 6C \\ \times & 7A \end{array}$$

se requieren 4 accesos a memoria y dos sumas. Suponiendo que cada suma (de 16 bits) requiere 4 Δt y que Δt=60 ns, el tiempo total es de $2 \times 240 + 2000 = 2480$ us. En contraste, por el método de (b) se requieren 16 sumas y 16 corrimientos, par un total aproximado de $48 \times 120 = 5760$ us. Es decir, el método de (c) es aproximadamente un 120% más rápido que él de (b). Por otro lado, las memorias más rápidas tienen tiempos de acceso mejores que 100 ns, lo cual reduciría el tiempo de multiplicación a alrededor de 1600 ns, (en el caso del ejemplo anterior).

7.4. <u>División</u>. La división es, tradicionalmente, la más lenta de las 4 operaciones básicas. Como en el caso de la multiplicación, puede lograrse restando repetidamente el divisor del dividendo. Por ejemplo, la división:

$$2526/6 = 421$$

Se puede lograr restando 421 veces el número 6 del dividendo.

digits are multiplied in binary form in a multiplication circuit much like that described above, but then the partial products are combined in a particular way with respect to each other in shift registers and added.

To understand how this is done, consider two ways in which the partial products of a multiplication can be written:

Common longhand method	*BCD method*	
4 6 7	4 6 7	
6	6	Partial Product
4 2	2 3 4	Stored in Shift Register A
3 6	4 6 2	Stored in Shift Register B
2 4	2 8 0 2	Sum, after adding A and B
2 8 0 2		

The first method, of course, is more familiar to us, but the second method, in which all partial products are written diagonally, is more useful in logic circuits. It requires only two shift registers to store the partial products and the addition can be carried out by a BCD adder, such as shown in Figure 7-12.

Another common and increasingly popular method used in BCD multiplication uses **look-up tables**. All possible multiples of two numbers between 0 and 9 are stored in a read-only memory (ROM) which then serves as the look-up table. Multiplication can be simplified by reading the product of any two digits out of the ROM directly, thus eliminating the digit multiplication entirely and reducing the process to repeated additions. (In the above example, 6 x 7 = 42, as well as other partial products are read out of the ROM.)

DIVISION

Because division is the opposite of multiplication, it can be performed by repeated subtraction. The same decimal number examples used for multiplication can illus[illegible]

1 ← (quotient)

(divisor) → 10) 10 ← (dividend)

10	
-10	1st subtraction
30	
-10	2nd subtraction
20	
-10	3rd subtraction
10	
-10	4th subtraction
00	

The example shows that division can be performed by subtracting 10 from 40 four times – the quotient being the number of times the subtraction is performed. This method can be extended to division of large numbers, in a manner that is similar to longhand division.

```
    421
6 ) 2526
    24
    --
     12
     12
     --
      6
      6
      -
      0
```

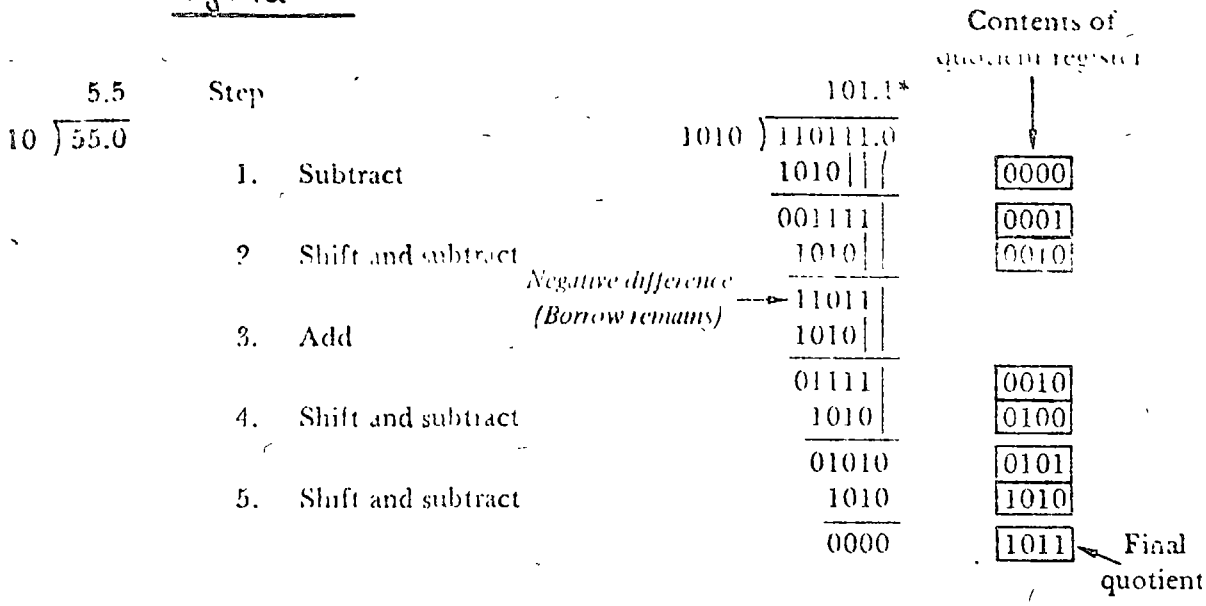
can also be performed as follows:

Step	6) 2526	Add to quotient	Quotient (total)
1.	600	100	100
	1926		
2.	600	100	200
	1326		
3.	600	100	300
	726		
4.	600	100	400
	126		
5.	60	10	410
	66		
6.	60	10	420
	6		
7.	6	1	421
	0		

The method on the right used repeated subtraction also, but resembles more closely the method used in a computer. Instead of subtracting 6 from 2526 a total of 421 times, the 6 has been *shifted* left by two places (thus multiplying it by 100). Then, it has been subtracted four times (steps 1 through 4), until the remainder 126 is too small to allow a fifth subtraction of 600. With the longhand method (on the left) the same four steps are accomplished with the first subtraction: 6 will go into 25 four times, and 6 x 400 = 2400 is subtracted from 2526 in a single step. (The procedure for subsequent subtraction steps 5, 6, and 7 is in principle the same, except the 6 is shifted right to subtract 60 and then 6.)

A version of this process of division by repeated subtractions and shifting is used in arithmetic circuits of computers and calculators. Of course, the computer does its arithmetic in binary, rather than decimal form, but the procedure can be used in either system. Another example will show how a division is performed in binary arithmetic. (In this example division involving fractions will be introduced.)

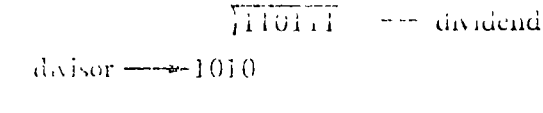
Figura 9.



Binary Division Algorithm (restoring method):

- *a. Subtract divisor from dividend*
- *a(1) If result is a positive number, put 1 in rightmost digit of quotient register; if result is a negative number, add divisor back to dividend.*
- *b. Shift quotient left by one digit and shift divisor right by one digit (or dividend left)*
- *c. Repeat steps a., a(1), and b*
- *d. Continue steps a. through c. until a subtraction yields a difference of all 0 bits, until required accuracy is obtained, or until all available bit positions in quotient register are filled.*

The above example uses essentially the same procedure as that described in the decimal division example of 2526 ÷ 6. However, note two points. First, as 600 was chosen in the first subtraction in the decimal example, the position of the divisor was chosen in this example. A computer, however, does not have the ability to examine in the same way and must start with the highest possible value of the divisor (shifting it left as far as possible):



Then, it must perform three trial subtractions and, with each unsuccessful subtraction (negative difference), shift the divisor right until it is in the position we started with.

*Binary fractions are converted to decimal form by placing the binary number following the decimal (i.e. binary) point as the numerator above the total number of states that the digits can define. Examples: 0.1 = 1/2, 0.10 = 2/4, 0.1011 = 11/16, 0.1111 = 16/16, etc.

Una forma más racional (y más rápida) de lograr el mismo resultado es multiplicar el divisor (haciendo corrimientos a la izquierda) y restar el el divisor así miltiplicado del dividendo hasta obtener un número negativo de las restas. Entonces se hace un corrimiento en sentido inverso y repitiendo el proceso hasta llegar a un residuo final (que puede ser diferente de cero).

Un ejemplo de este algoritmo (llamado algoritmo de restauración)se muestra en la figura 9. (7-25).

Un tercer algoritmo de división supone que el divisor es mayor que el dividendo. El diagrama de flujo de dicho algoritmo se muestra en la figura 9.

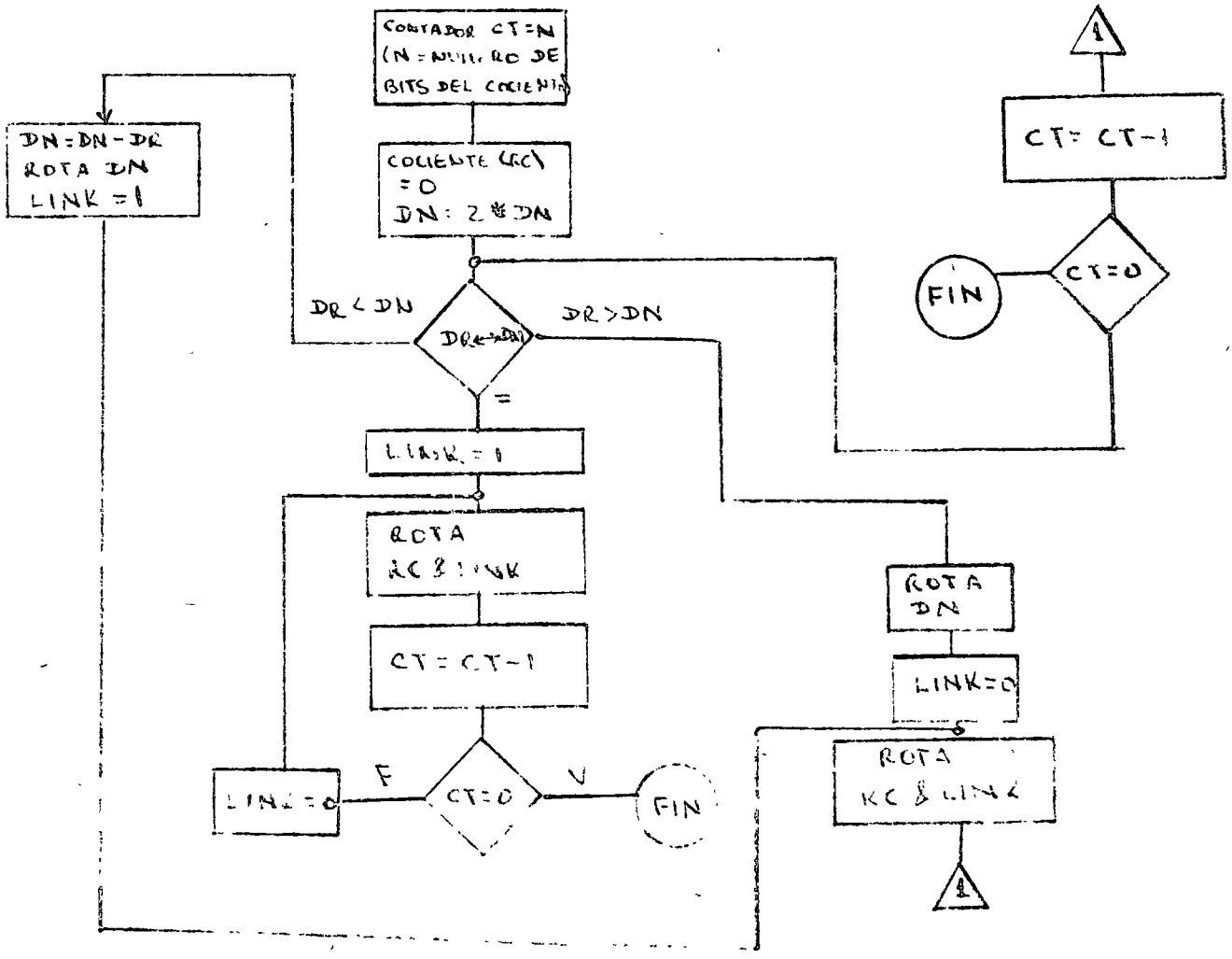


Figura 9.

En este algoritmo todas las rotaciones que se indican son a la izquierda y LINK es un Flip-Flop auxiliar asociado al registro de resultado RC. Como se vé, hay 2N rotaciones y N/2 restas. Para palabras de 16 bits de precisión en el resultado esto implica 40 operaciones y un total aproximado de 40 X 240 = 9600 us.

Para generalizar este algoritmo es suficiente con hacer una normalización previa del registro DR, de manera que se asegure que DN<DR y usar un contador auxiliar que lleve registro del número de rotaciones previas a que se cumpla la condición anterior. El algoritmo entonces es cómo se muestra en la figura 10.

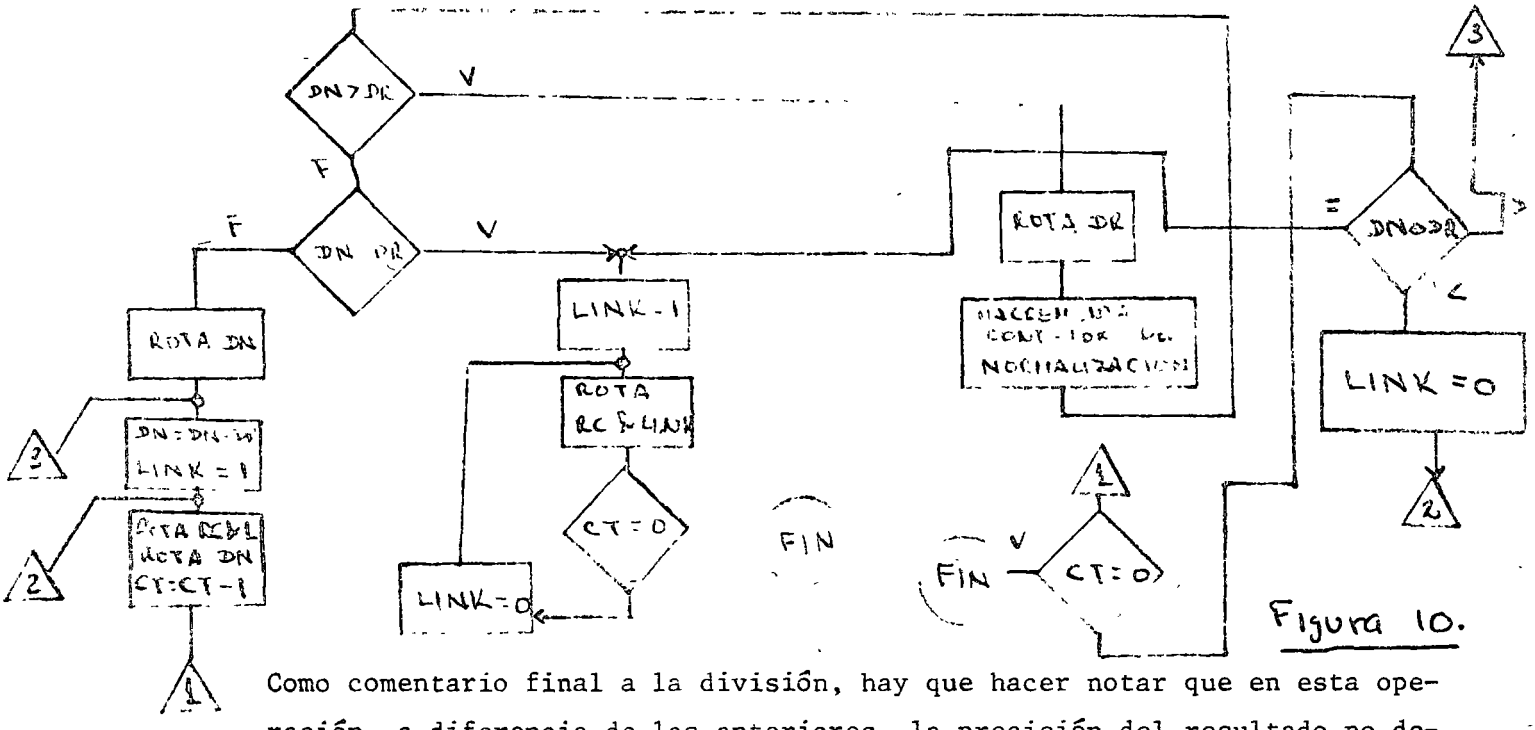


Figura 10.

Como comentario final a la división, hay que hacer notar que en esta operación, a diferencia de las anteriores, la precisión del resultado no depende de la precisión de los operandos.

7.5. Unidades lógico-aritméticas. Aunque, como se dijo, es posible implementar todas las operaciones a base de SC's, existen las llamadas ALU's, tales como las que se muestra en las hojas de componentes (74181). Con estos circuitos integrados es relativamente sencillo implementar un buen número de funciones programando sus entradas.

En la figura 11 se muestra un multiplicador implementado con un 74181 y otras componentes.

EXPERIMENT 22

ARITHMETIC ELEMENTS

Multiplication

CONCEPT

Multiplication is a process of successive additions in which the product is the sum of the partial products, properly shifted with respect to one another. There are two basic types of multiplication:

simultaneous addition of all partial products

```
  101   → multiplicand
  111   → multiplier
 -----
  101
  101   } partial products
 101
 -----
100011   all partial products added simultaneously
```

successive addition of each partial product

```
 111
 101 ←── if the first digit of the multiplier is a 1, data is
 ---     transferred to the output as a partial product
 111 ←──
 0111 ←── if the next least significant bit is a 0,
  0111 ←─ the product is shifted right one position
 111     ─ if MSD is a 1, the product is shifted right
 ------    one position and added to the multiplicand
100011
```

Successive addition of each partial product is the process most commonly used when constructing a binary multiplier. The multiplier circuit is configured as shown in the Multiplier Block Diagram.

MULTIPLIER BLOCK DIAGRAM

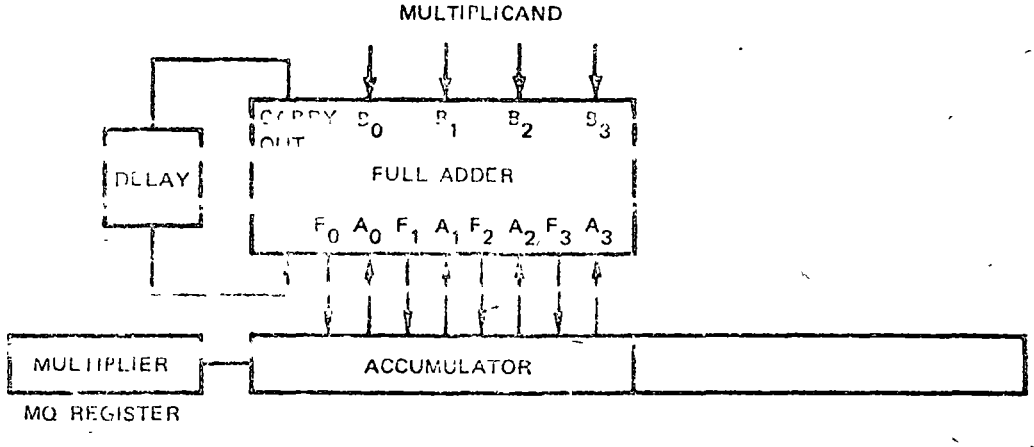
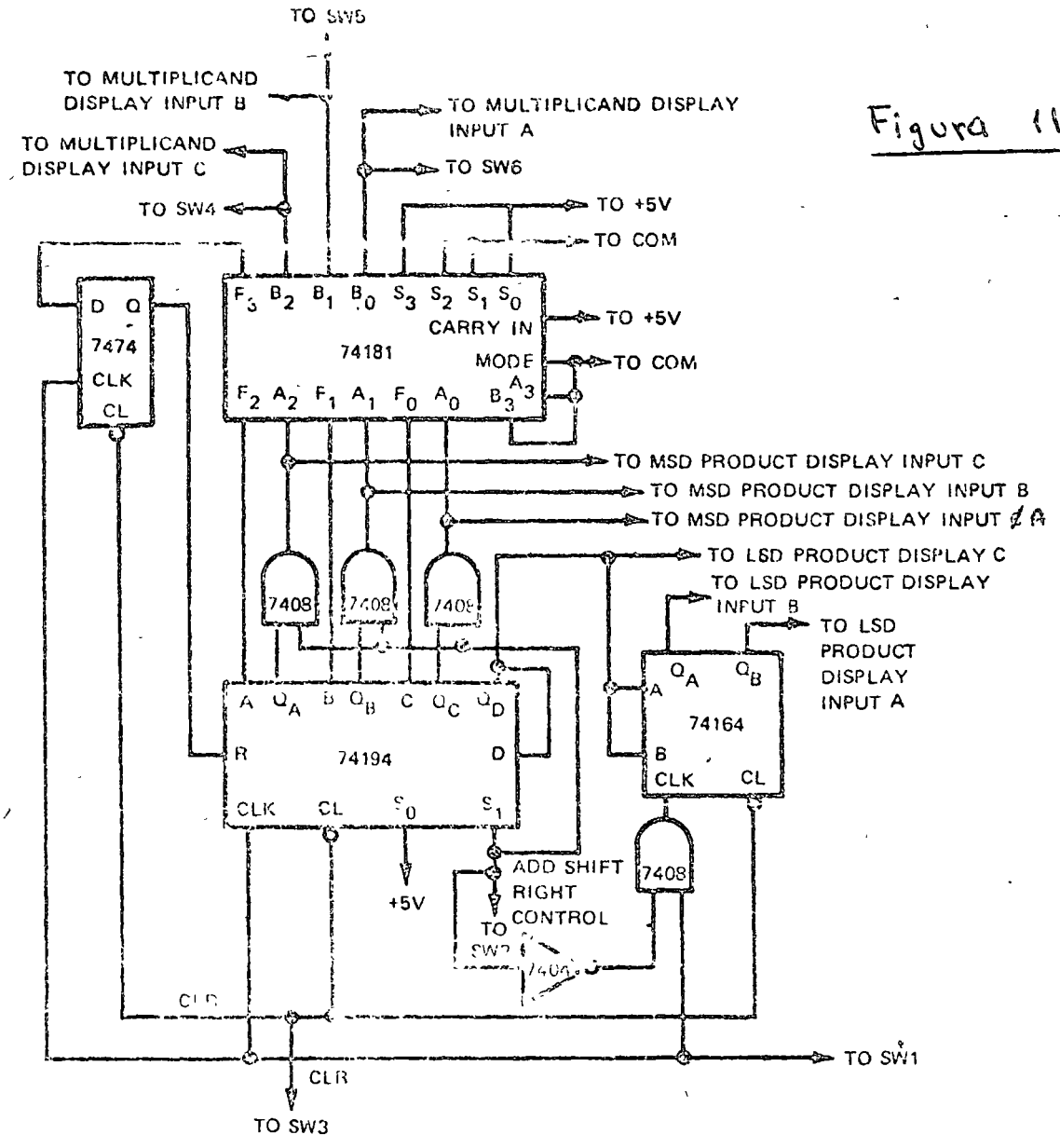
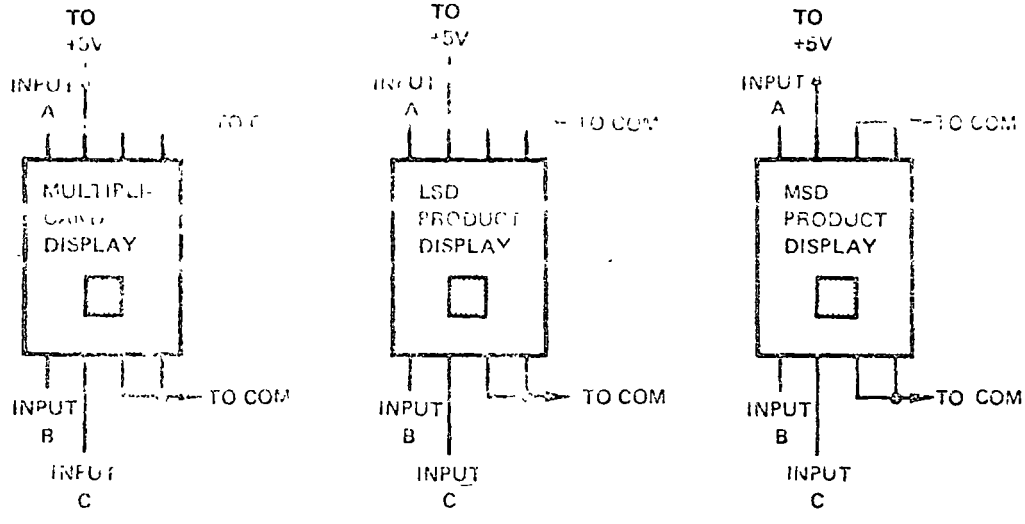


Figura 11.

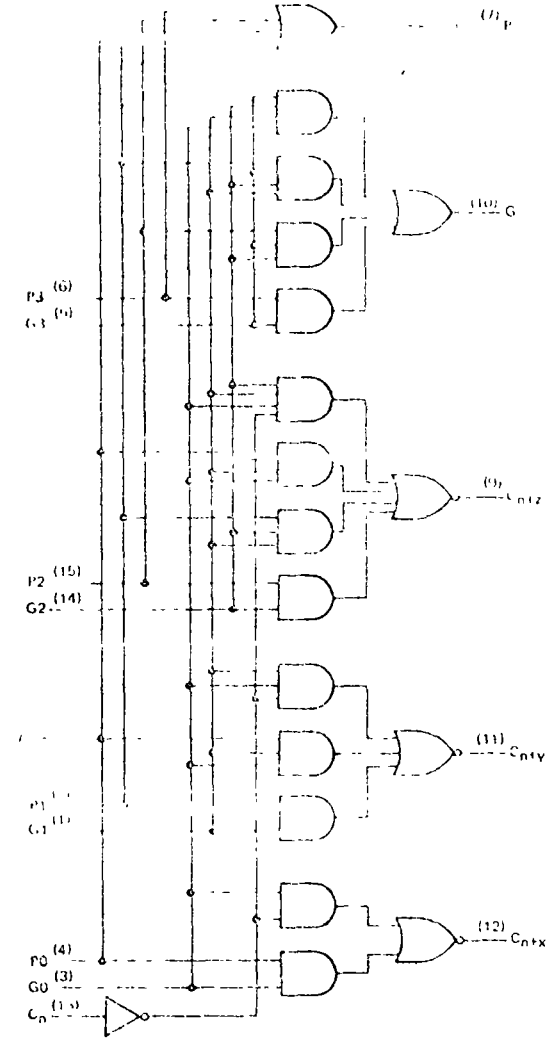

TO SW5
TO MULTIPLICAND DISPLAY INPUT B
TO MULTIPLICAND DISPLAY INPUT A
TO MULTIPLICAND DISPLAY INPUT C
TO SW6
TO SW4
TO +5V
TO COM
D Q
7474
CLK
CL
F3 B2 B1 B0 S3 S2 S1 S0
CARRY IN
74181
MODE
F2 A2 F1 A1 F0 A0 B3 A3
TO +5V
TO COM
TO MSD PRODUCT DISPLAY INPUT C
TO MSD PRODUCT DISPLAY INPUT B
TO MSD PRODUCT DISPLAY INPUT A
TO LSD PRODUCT DISPLAY C
TO LSD PRODUCT DISPLAY INPUT B
TO LSD PRODUCT DISPLAY INPUT A
7408
7408
7408
A QA B QB C QC QD
R
74194
D
CLK CL S0 S1
QA QB
A
74164
B
CLK CL
+5V
ADD SHIFT RIGHT CONTROL
TO SW2
7408
7408
CLR
TO SW1
CLR
TO SW3


TO +5V
INPUT A
MULTIPLI-
CAND
DISPLAY
TO COM
INPUT B
INPUT C
TO +5V
INPUT A
LSD
PRODUCT
DISPLAY
TO COM
INPUT B
INPUT C
TO +5V
INPUT A
MSD
PRODUCT
DISPLAY
TO COM
INPUT B
INPUT C

Las salidas G y P del 74181 corresponden a las salidas de 'look-ahead'. Estas pueden utilizarse para hacer una suma de 64 bits (que es el caso que se ilustra) en un máximo de $7\Delta t$ (7 X 60 us = 420 us). Para la generación de las señales del 'look ahead' se puede utilizar un 74182 como el que se muestra en la figura 12 (392) y que se pueden conectar como se ve en la figura 13 (395).

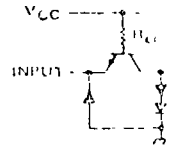
Figura 13.

functional block diagram and schematics of inputs and outputs



'182

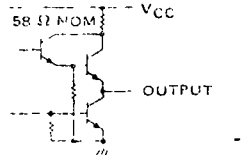
EQUIVALENT OF EACH INPUT



INPUT	R_{eq} NOM
C_n	2.8 kΩ
P3	1.4 kΩ
P2	940 Ω
P0, P1, G3	700 Ω
G0, G4	400 Ω
G1	350 Ω

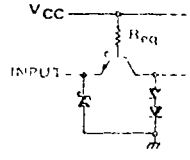
'182

TYPICAL OF ALL OUTPUTS



'S182

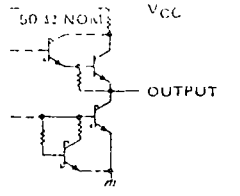
EQUIVALENT OF EACH INPUT



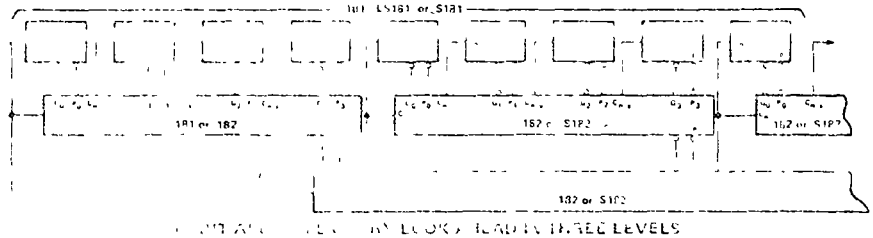
INPUT	R_{eq} NOM
C_n	2.8 kΩ
P3	1.4 kΩ
P2	940 Ω
P0, P1, G3	700 Ω
G0, G4	400 Ω
G1	350 Ω

'S182

TYPICAL OF ALL OUTPUTS



TYPICAL APPLICATION DATA



A and B inputs and F outputs of '181, 'LS181, and S181 are not shown.

Figure 12.

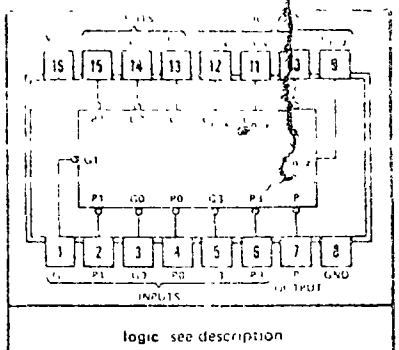
TYPES SN54182, SN54S182, SN74182, SN74S182
LOOK-AHEAD CARRY GENERATORS

BULLETIN NO. DL-S 7211983, DECEMBER 1972

PIN DESIGNATIONS

DESIGNATION	PIN NO.	FUNCTION
$\bar{G}_0$ $\bar{G}_1$ $\bar{G}_2$ $\bar{G}_3$	1 2 14 15	ACTIVE LOW CARRY GENERATE INPUTS
P_0 P_1 P_2 P_3	4 5 6 9	ACTIVE LOW CARRY PROPAGATE INPUTS
C_n	13	CARRY INPUT
C_{n+x} C_{n+y} C_{n+z}	11 12 10	CARRY OUTPUTS
G	3	ACTIVE LOW CARRY GENERATE OUTPUT
P	7	ACTIVE LOW CARRY PROPAGATE OUTPUT
V_{CC}	16	SUPPLY VOLTAGE
GND	8	GROUND

J OR N DUAL-IN-LINE PACKAGE (TOP VIEW)



TYPE	TYPICAL AVERAGE PROPAGATION DELAY TIME	TYPICAL POWER DISSIPATION
'182	13 ns	180 mW
'S182	7 ns	260 mW

description

The SN54182, SN54S182, SN74182, and SN74S182 are high-speed, look-ahead carry generators capable of anticipating a carry across four binary adders or group of adders. They are cascadable to perform full look-ahead across n-bit adders. Carry, generate-carry, and propagate-carry functions are provided as enumerated in the pin designation table above.

When used in conjunction with the '181, 'LS181, or 'S181 arithmetic logic unit (ALU), these generators provide high-speed carry look-ahead capability for any word length. Each '182 or 'S182 provides look-ahead (anticipated carry) across a group of four ALU's and, in addition, other carry look-ahead circuits may be employed to anticipate carry across sections of four look-ahead packages up to n bits. The method of cascading '182 or 'S182 circuits to perform multi-level look-ahead is illustrated under typical application data.

Carry input and output of the '181, 'LS181, and 'S181 ALU's are in their true form and the carry propagate (P) and carry generate (G) are in complemented form; therefore, the carry functions (inputs, outputs, generate, and propagate) of the look-ahead generators are implemented in the complemented forms for direct connection to the ALU. Reinterpretations of carry functions as explained on the '181, 'LS181, and S181 data sheet are also applicable to and compatible with the look-ahead generator. Positive logic equations for the '182 and 'S182 are

$$C_{n+x} = \bar{G}_0 + P_0 C_n$$
$$C_{n+y} = \bar{G}_1 + \bar{P}_1 \bar{G}_0 + P_1 P_0 C_n$$
$$C_{n+z} = \bar{G}_2 + P_2 \bar{G}_1 + P_2 P_1 \bar{G}_0 + \bar{P}_2 \bar{P}_1 \bar{P}_0 C_n$$
$$G = \bar{G}_3 (\bar{P}_3 + \bar{G}_2)(P_3 + P_2 + \bar{G}_1)(P_3 + \bar{P}_2 + \bar{P}_1 + \bar{G}_0)$$
$$\bar{P} = P_3 P_2 P_1 P_0$$

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC} (see Note 1) . . . . . 7 V
Input voltage . . . . . 5.5 V
Interemitter voltage (see Note 2) . . . . . 5.5 V
Operating free-air temperature range: SN54', SN54S' Circuits . . . . . −55°C to 125°C
SN74', SN74S' Circuits . . . . . 0°C to 70°C
Storage temperature range . . . . . −65°C to 150°C

NOTES: 1. Voltage values are with respect to network ground terminal.
2. This is the voltage between two emitters of a multiple-emitter input. For this device, it applies to the G input in conjunction with any other G input or in conjunction with any P input.

TTL MSI

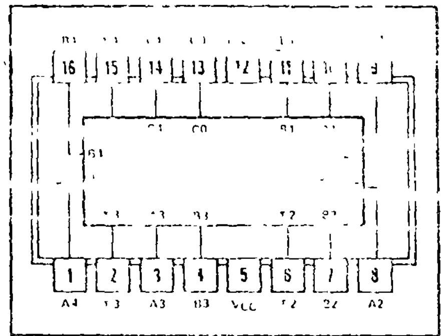
TYPES SN5483A, SN54LS83, SN7483A, SN74LS83 4-BIT BINARY FULL ADDERS

BULLETIN NO. DL-S 7211857, DECEMBER 1972

- For applications in:
  - Digital Computer Systems
  - Data Handling Systems
  - Control Systems
- SN54283/SN74283 Are Recommended For New Designs as They Feature Supply Voltage and Ground on Corner Pins to Simplify Board Layout

TYPE	TYPICAL ADD TIMES: TWO 8-BIT WORDS	TYPICAL ADD TIMES: TWO 16-BIT WORDS	TYPICAL POWER DISSIPATION PER 4-BIT ADDER
'83A	23 ns	43 ns	310 mW
'LS83	89 ns	165 ns	75 mW

J OR N DUAL-IN-LINE OR W FLAT PACKAGE (TOP VIEW)



positive logic: see function table

description

These full adders perform the addition of two 4-bit binary numbers. The sum (Σ) outputs are provided for each bit and the resultant carry (C4) is obtained from the fourth bit. The adders are designed so that logic levels of the input and output, including the carry, are in their true form. Thus the end-around carry is accomplished without the need for level inversion. Designed for medium-to-high-speed, the circuits utilize high-speed, high-fan-out transistor-transistor logic but are compatible with both DTL and TTL families.

The '83A circuit features full look-ahead across four bits to generate the carry term in typically 10 nanoseconds to achieve partial look-ahead performance with the economy of ripple carry.

The 'LS83 can reduce power requirements to less than 20 mW/bit for power-sensitive applications. These circuits are implemented with single-inversion, high-speed, Darlington-connected serial-carry circuits within each bit.

Series 54 and 54LS circuits are characterized for operation over the full military temperature range of −55°C to 125°C; Series 74 and 74LS are characterized for 0°C to 70°C operation.

FUNCTION TABLE

INPUT				OUTPUT: WHEN C0 = L			WHEN C0 = H		
A1 / A3	B1 / B3	A2 / A4	B2 / B4	Σ1 / Σ3	Σ2 / Σ4	C2 / C4	Σ1 / Σ3	Σ2 / Σ4	C2 / C4
L	L	L	L	L	L	L	H	L	L
H	L	L	L	H	L	L	L	H	L
L	H	L	L	H	L	L	L	H	L
H	H	L	L	L	H	L	H	H	L
L	L	H	L	L	H	L	H	H	L
H	L	H	L	H	H	L	L	L	H
L	H	H	L	H	H	L	L	L	H
H	H	H	L	L	L	H	H	L	H
L	L	L	H	L	H	L	H	H	L
H	L	L	H	H	H	L	L	L	H
L	H	L	H	H	H	L	L	L	H
H	H	L	H	L	L	H	H	L	H
L	L	H	H	L	L	H	H	L	H
H	L	H	H	H	L	H	L	H	H
L	H	H	H	H	L	H	L	H	H
H	H	H	H	L	H	H	H	H	H

H = high level, L = low level

NOTE: Input conditions at A1, B1, A2, B2, and C0 are used to determine outputs Σ1 and Σ2 and the value of the internal carry C2. The values at C2, A3, B3, A4, and B4 are then used to determine outputs Σ3, Σ4, and C4.

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage, V_{CC} (see Note 1)	7 V
Input voltage	5.5 V
Interemitter voltage (see Note 2)	5.5 V
Operating free-air temperature range: SN54', SN54LS' Circuits	−55°C to 125°C
SN74', SN74LS' Circuits	0°C to 70°C
Storage temperature range	−65°C to 150°C

NOTES: 1. Voltage values, except interemitter voltage, are with respect to network ground terminal.
2. This is the voltage between two emitters of a multiple-emitter transistor. For '83A circuits, this rating applies to each A input in conjunction with its respective B input. For the 'LS83, this rating applies between the following pairs: A1 and B1, A2 and B2, A3 and B3, A4 and B4. For the 'LS83, this rating applies between the following pairs: A1 and B1, A1 and C0, B1 and C0, A3 and B3.

functional block diagram

schematics of inputs

TEXAS INSTRUMENTS INCORPORATED
POST OFFICE BOX 5012 • DALLAS, TEXAS 75222

BIBLIOGRAFIA:

1. Digital Systems: Hardware Organization and Design. Hill & Peterson, Wiley, 1973.

2. Practical Digital Electronics, an introductory course. Blukes and Baker, Hewlett-Packard, 1974.

3. Practical Digital Electronics, laboratory workbook. Bird and Schmidt, Hewlett-Packard, 1974.

4. Computer Organization. Ivan Flores, Prentice Hall, 1969.

5. Design of a Computer, the CDC 6600. Thorton, Scott-Foresman and Co., 1970.

VIII. FAMILIAS LOGICAS

8.0 INTRODUCCION:

8.0.1 Una clasificación.- El procesamiento digital moderno está basado en forma absoluta en la electrónica de estado sólido desde la década de los 50's con transistores, pasando por los 60's y los primeros circuitos integrados de pequeña-mediana escala, a la década actual en la que los circuitos integrados de gran escala han hecho realidad el microprocesador.
En la actualidad existen tres formas principales de distinguir los circuitos integrados digitales: en base a la tecnología de fabricación, en base al funcionamiento interno y en base al grado de integración. La primera distinción tiene dos grandes grupos: Bipolar y MOS (Metal-oxido-semiconductor). La segunda en cierto modo combina en parte el proceso de fabricación y el mecanismo de acción interna del circuito, distinguiendo en la actualidad varias "familias" de las cuales las principales son: DTL, TTL, STTL, I^2L, ECL, NMOS, PMOS, CMOS y CCD. Finalmente, el grado de integración, aunque no está muy bien definido puede ser: pequeño (SSI), con menos de 100 elementos por circuito, mediano (MSI) con menos de 1000 elementos por circuito y grande (LSI).
En el resto de esta sección se describirán someramente algunas de las características de estas familias lógicas.
Por el momento, la tabla 8.1 muestra la interrelación entre las diferentes clasificaciones de familias lógicas.

	BIPOLAR			MOS			
	TTL/DTL/STTL	ECL	I^2L/MTL	CMOS	N-MOS	P-MOS	CCD
SSI	✓	✓	NO	✓	NO	✓	NO
MSI	✓	✓	✓	✓	✓	✓	NO
LSI	NO	NO	✓	✓	✓	✓	✓

En dicha tabla se muestra como algunas familias no son empleadas en ciertos grados de integración, ya sea por imposibilidad tecnológica o por inconveniencia comercial.

8.0.2.- Característica de Transferencia y Margen de Ruido.- La característica de transferencia ideal de una compuerta lógica (por ejemplo una compuerta NAND con todas sus entradas conectadas entre sí), debe hacer una clara distinción entre dos niveles de voltaje, los cuales representan un 1 o un 0. Por ejemplo, la figura 8.1 muestra la característica de transferencia ideal para una compuerta NAND de dos entradas (conectadas entre sí).

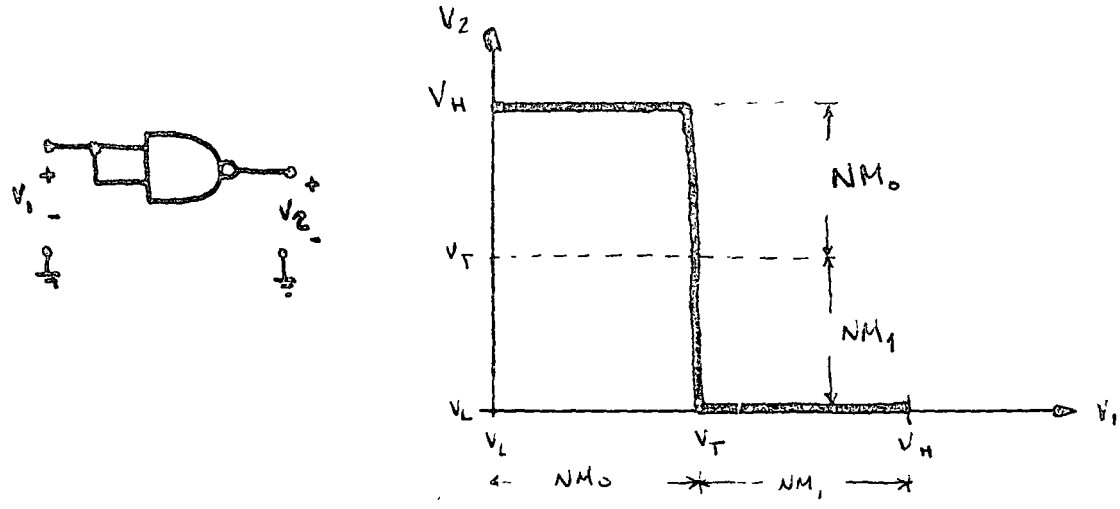


FIGURA 8.1

Los voltajes están referidos a un punto común (tierra). Mientras la entrada tenga un voltaje cercano a cero, la salida tiene un voltaje alto (V_H). Cuando el voltaje de entrada sobrepase el voltaje de umbral (V_T), la salida cambiará a un voltaje bajo (V_L). Como se supone que esta compuerta está siendo excitada por otra compuerta idéntica, el voltaje de entrada tiene como límites los voltajes máximo (V_H) y mínimo (V_L) de salida.

Se define como Margen de Ruido (NM) a la diferencia entre el voltaje máximo (mínimo) obtenible a la salida de la compuerta excitadora y el mínimo (máximo) tolerable antes de que el nivel lógico sea irreconocible. En este caso ideal, el margen de ruido para un CERO a la salida (NM_o) es ($V_H - V_T$), mientras que el margen de ruido para un nivel UNO a la salida (NM_1) es ($V_T - V_L$). También en este caso ideal, para que ambos márgenes

de ruido sean máximos se debe tener que $V_T = \frac{1}{2}(V_H - V_L)$.

En la práctica, no es posible tener el caso ideal, aunque algunos circuitos (como los CMOS) se aproximan bastante.

8.0.3.- Tiempo de conmutación y potencia.- La conmutación de un CERO a un UNO, o a la inversa, en un compuerta lógica no sucede en forma instantánea. El tiempo que tarda en conmutar depende del diseño del circuito y de la carga que tenga conectada a la salida. Las causas que producen este fenómeno, así como el análisis del mismo, se salen del propósito de estas notas, por lo que se limitarán a describir la forma que este parámetro se especifica.

En la figura 8.2 se muestra un diagrama en el tiempo de las señales de entrada y salida de una compuerta inversora (NAND o NOR)

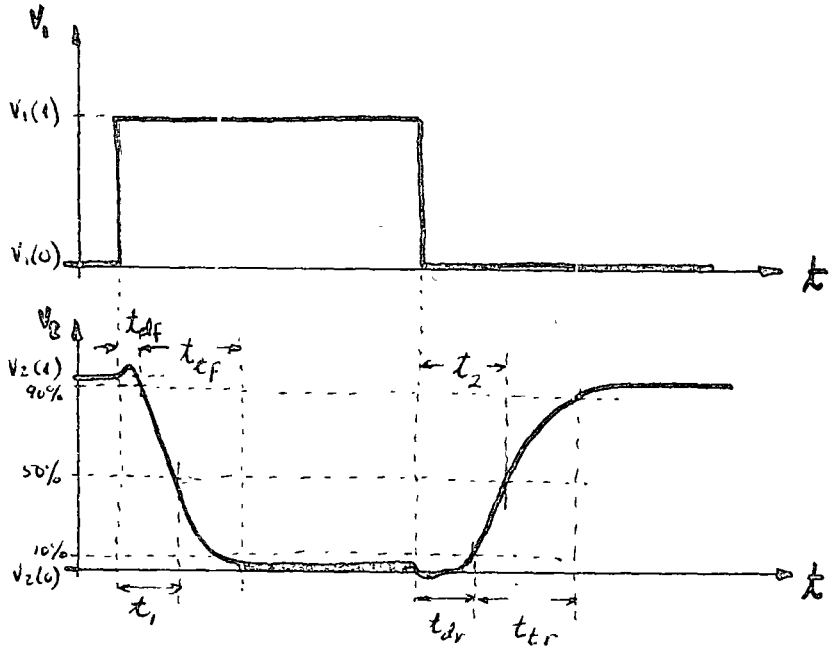


FIGURA 8.2

El tiempo de conmutación de un estado a otro se considera compuesto de dos etapas: tiempo de retardo inicial (t_d) y tiempo de transición, (t_t). El primero representa el tiempo que tarda el circuito en empezar a cambiar, mientras que el segundo marca el tiempo que tarda en llegar de un 10% del valor final a un 90% de dicho valor.

Otra forma de medir el retardo de la señal es medir el tiempo que tarda en llegar al 50% de la transición total, desde un CERO (t_1) y desde un UNO (t_2), dándose como "tiempo de propagación" el promedio de ambas $t_p = \frac{1}{2}(t_1 + t_2)$.

La especificación de estos tiempos de propagación, retardo, etc., se hacen considerando al circuito en ciertas condiciones de carga; normalmente se especifican con una carga equivalente a 10 compuertas similares, aunque esto no es siempre así.

La potencia disipada en un circuito depende en mucho de su diseño y de la tecnología de fabricación. Igualmente depende del valor de la fuente de poder que emplee y de la carga capacitiva que esté conmutando.

La potencia "estática" es aquella que disipa la compuerta en estado de reposo (o sea sin conmutar), ya sea estando en un CERO o en un UNO a la salida.

La potencia dinámica es aquella que disipa la compuerta en promedio, conmutando a su máxima de velocidad con una onda cuadrada a la salida (es decir con un "ciclo de trabajo" del 50%). Estas potencias no son iguales y aunque a veces son parecidas, la potencia dinámica es normalmente mayor.

Existe también una relación entre la velocidad de conmutación de un circuito y la potencia dinámica que disipa. Normalmente, para una familia dada, el producto Potencia Dinámica x Retardo (PXD) es constante y está dado por:

$$P \times D = \frac{1}{2} C V_{cc} AV_L$$

En donde: C es la carga capacitiva a la salida del circuito, V_{cc} es el valor de la fuente de alimentación y AV_L es la excursión de voltaje entre UNO y el CERO Lógicos del circuito.

8.1 FAMILIAS LOGICAS BIPOLARES

Las familias lógicas bipolares con las que el diseñador actual se encontrará en la práctica son, principalmente: TTL, STTL, ECL, DTL. Aunque I^2L está teniendo un gran auge, su presencia en un circuito es "transparente" al usuario, ya que siempre aparece con interfaces apropiados para interconectarse con TTL.

8.1.1 DTL (Diode-transitor-Logic).- Este tipo de circuito lógico apareció primeramente en forma discreta (es decir no-integrada) en los albores del procesamiento digital moderno. Junto con RTL (resistor-transistor-Logic), formaban las familias lógicas más usadas, y aún en la actualidad, en ocasiones se hace necesario usar algunos de estos circuitos en problemas sencillos. El uso de circuitos discretos tiene ciertas ventajas, como flexibilidad de operación, a cambio de una gran cantidad de desventajas: costo, dificultad de diseño (para el que no conoce los rudimentos del diseño electrónico), espacio ocupado, baja velocidad. Por lo tanterior, sólo se usa en contadas ocasiones.

El principio de funcionamiento de la compuerta DTL se ilustrará por medio del circuito que se muestra en la figura 8.3 y se describe a continuación.

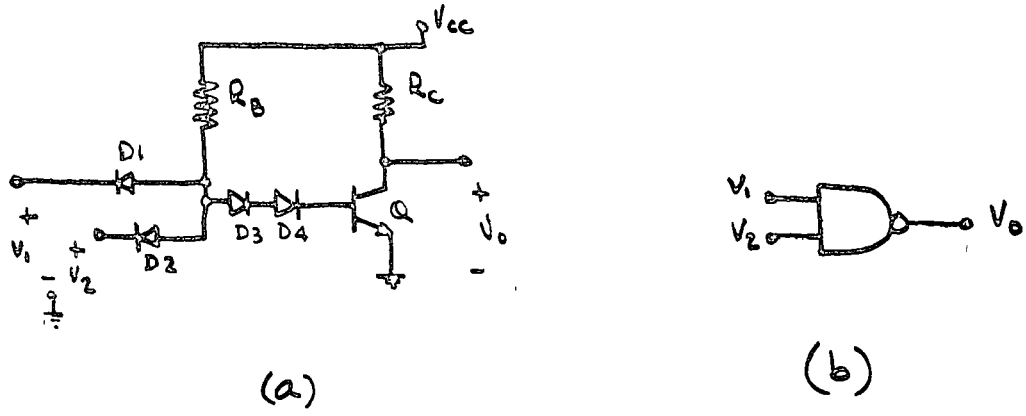


FIGURA 8.3

Primeramente, en forma cualitativa, se puede apreciar que la función lógica de este circuito es la de una compuerta NAND (considerada al nivel 1 como

un voltaje alto positivo, y al nivel 0 como un voltaje cercano a cero). Cuando los dos voltajes de entrada (V_1 y V_2) son altas (por ejemplo $V_1 = V_2 \cong V_{cc}$), los diodos D_1 y D_2 se encuentran en inversa, y por lo tanto el transistor Q queda encendido a través de R_β, D_3 y D_4. En cambio si cualquiera de las entradas es un CERO (por ejemplo $V_1 = 0$), el potencial en el nodo B es demasiado pequeño para que Q pudiera encenderse, quedando éste apagado. Cuando el transistor Q está encendido, la corriente que circula a través de R_c forza al voltaje de salida (V_0) a ser bajo (o sea un CERO). Por el contrario, al estar Q apagado, la corriente a través de R_c es cero y el voltaje de salida es $V_0 = V_{cc}$ (o sea un uno). Por lo tanto, en este circuito se tendrá un UNO a la salida siempre que una de las entradas sea un CERO, y sólo se tendrá un CERO a la salida cuando ambas entradas sean UNO; es decir es una compuerta NAND.

Para mostrar el margen de ruido de esta compuerta es necesario añadir la excitación y la carga a que será sujeta. Ambas son normalmente compuertas del mismo tipo, así que se empleará el circuito de la figura 8.4, en la que se ha añadido como carga al circuito de entrada de otra compuerta DTL, y se han propuesto valores para las resistencias. En añadidura se requiere conocer los parámetros del transistor.

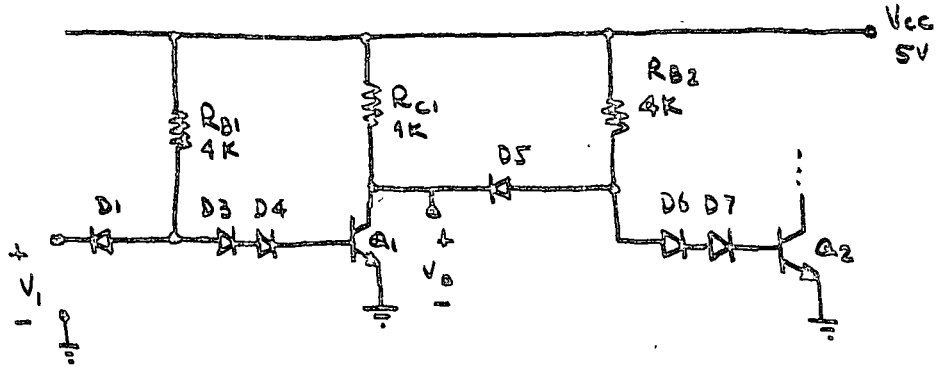


FIGURA 8.4

En este caso, para simplificar al problema se considerarán las siguientes características para el transistor: $V_{CE}(sat) = 0.2V$, $V_{BE}(encendido) = 0.6V$ y $\beta(min) = 40$.

Para los diodos se considerará que: V_D(encendido) = 0.6V y que en reversa llevan una corriente de fuga I_R = 10μA.

Con estos datos se puede proceder a analizar el circuito. Primero se considerará el caso en que V_1 es alto ($V_1 \cong V_{cc}$). En esta condición el circuito se puede representar por la figura 8.5, en la que se han eliminado los diodos y transistores que queden apagados.

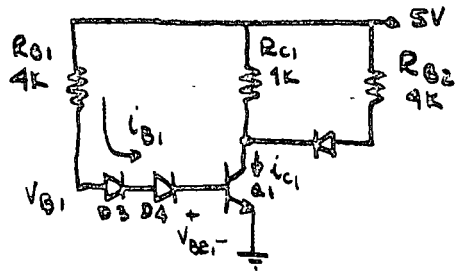


FIGURA 8.5

De la figura se puede obtener que:

$$i_{B1} = \frac{V_{cc} - V_{D3} - V_{D4} - V_{BE1}}{R_{B1}} = \frac{3.2V}{4K} = 0.8mA$$

y, suponiendo que Q_1 queda saturado (suposición que se comprobará posteriormente).

$$i_{C1} \cong \frac{V_{cc} - V_{ce1}(sat)}{R_{C1}} + \frac{V_{cc} - V_{DS} - V_{ce1}(sat)}{R_{B2}}$$

$$= \frac{4.8V}{1K} + \frac{4.2V}{4K} = 0.585\ mA$$

Obviamente, $\beta i_{B1} > i_{C1}$, con lo que se comprueba que Q_1 está saturado.

La solución de punto de operación anterior dá el primer punto mostrado en figura 8.6.

Ahora se propone disminuir el voltaje V_1 hasta llegar al punto en el que Q_1 sale de saturación y se encuentra en la región activa. Este punto será aquel en el que: $V_1 \cong (V_{BE1} + V_{D3} + V_{D4}) - V_{D1} = 1.2V$ (ver punto 2 en la figura 8.6) De este punto, la ganancia del inversor determinará a qué voltaje se apaga el transistor. Este voltaje es típicamente del orden de 1.1V a 1.0V.

Cuando Q_1 ha sido apagado (punto 3 en la figura 8.6), la única corriente que circuita por R_{c_1} es la corriente de fuga en los diodos, con lo que queda:

$$V_o(1) = Vcc - R_c I_R = 4.99\ V$$

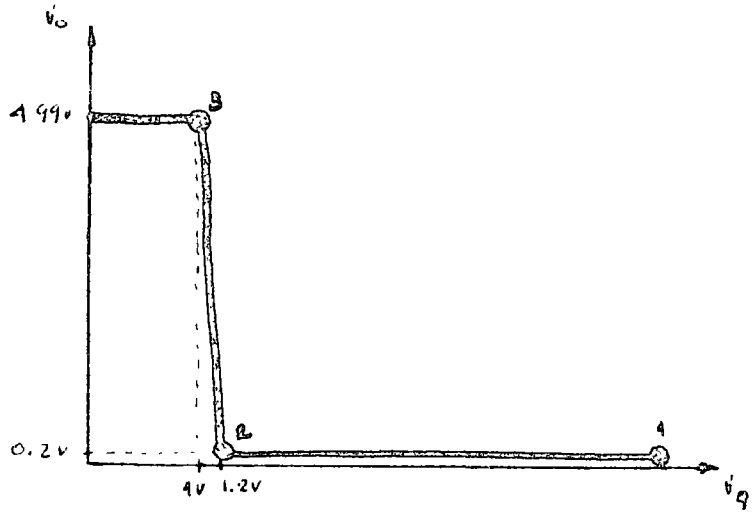


FIGURA 8.6

Aunque el análisis anterior ha ido simplificando algunos problemas del circuito, representa el método usual de análisis manual de las características estáticas de la compuerta. Existen variaciones en los parámetros del circuito que cambian la característica de transferencia, tales como: temperatura, número de compuertas conectadas a la salida y variaciones propias de las componentes del circuito (tolerancias, envejecimiento, etc.). Algunas de estas variaciones se especifican normalmente en las compuertas comerciales, como se mostrará más adelante.

El margen de ruido de este circuito, para las condiciones "ideales" en lo que ha sido presentado es:

$$NM_1 = (4.99 - 1.2) \cong 3.8\ V.$$

$$NM_0 = (1.0 - 0.2) = 0.8V.$$

Sin embargo, estos márgenes se verán afectados notablemente por las variaciones mencionadas en los párrafos anteriores.

Los tiempos de retardo y propagación no se pueden definir aquí y dependerán del circuito empleado. La potencia estática es máxima para el cero a la salida y vale alrededor de 6.5MW.

8.1.2.- TTL (Transistor - Transistor Logic.

Sin duda, esta es la compuerta más empleada actualmente a nivel de SSI y MSI. Tiene varios años de desarrollo y ofrece al diseñador una amplia gama de funciones lógicas como: multiplexores, demultiplexores, contadores, compuertas, flip-flops, etc.

Desde el punto de vista de circuito, es una compuerta DTL un poco modificada. El circuito de la figura 8.7 es prácticamente equivalente a DTL de la figura 8.3, con las siguientes modificaciones: primero, los diodos D_1 y D_2 son reemplazados por dos diodos formados por uniones base-emisor (este procedimiento es válido solo para circuitos integrados), el diodo D_3 es reemplazado por el diodo formado por la unión base-colector del transistor de entrada (Q_1); el diodo D_4 es reemplazado por un transistor, aumentando la β efectiva del transistor de salida. La adición del Resistor R_1 permite una ruta de descarga a la base de Q_3, con lo que mejora la velocidad de conmutación; esta resistencia está normalmente presente también en un DTL práctico.

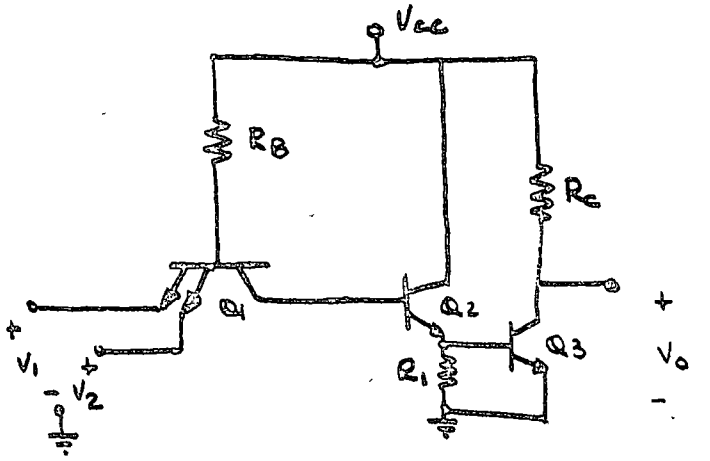


FIGURA 8.7

Aunque esta compurte TTL es común, aunque por lo general no incluya a Rc (por lo que se le llama circuito de "colector abierto"), el circuito TTL por excelencia emplea una etapa de salida del tipo pugh-pull, llamada "Totoem pole". Este circuito, ya clásico, se muestra en la figura 8.8. El transistor de entrada Q_1 integra los didos que forman la esencia de la compuerta DTL. Los diodos D_2 y D_3 sirven para fijar excursiones negativas

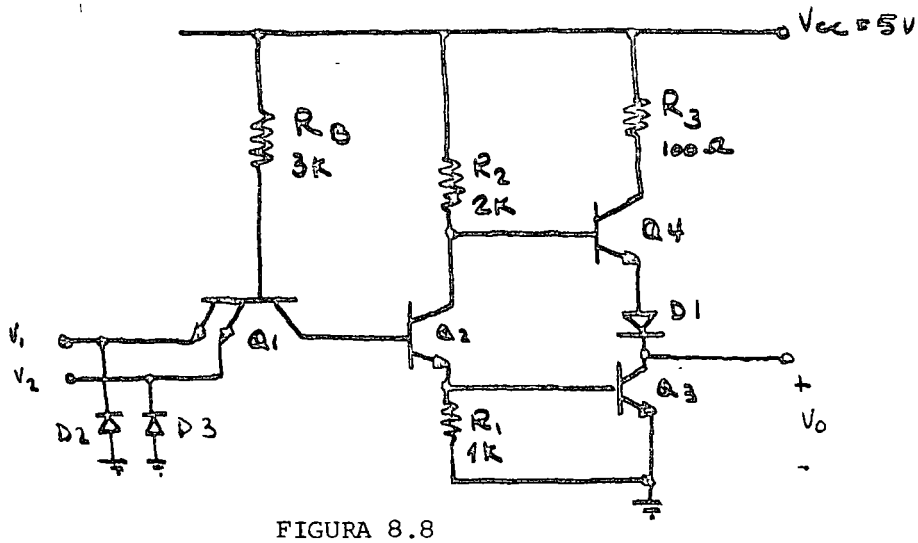


FIGURA 8.8

en el voltaje de entrada, y evitar la destrucción de la compuerta. El transistor Q_2 actúa en parte como un amplificador, proporcionando a las bases de Q_3 y Q_4 señal en contrafase (es decir, que cuando el voltaje en la base de Q_3 aumenta, el de la base de Q_4 disminuye, y viceversa). Los transistores de salida, Q_3 y Q_4, proporcionan la corriente necesaria para accionar las demás compuertas a que está conectada ésta.

El diodo D_1 tiene por objeto evitar que Q_4 se encienda mientras Q_3 está encendido en estado estable.

El principal propósito del circuito Totem-pole es tener la capacidad de proporcionar mucha corriente al circuito de carga, con el propósito de aumentar la velocidad de propagación del circuito.

A continuación se describe este circuito, con algunos valores típicos empleados en la conocida serie 7400 de compuertas TTL (ver figura 8.8).

Los parámetros de los transistores, por ilustración serán iguales a los especificados para la compuerta TTL. Al igual que para la DTL, se "cargará" a la compuerta, sólo que esta vez se utilizará el equivalente a diez compuertas TTL, para ilustrar el efecto de la carga. La carga se simulará con el circuito de la figura 8.9, que es también lo recomendado por los fabricantes para esta prueba. El diodo DA representa la unión base-emisor de entrada; los diodos DB, DC y DD representan los voltajes de la fijación de:

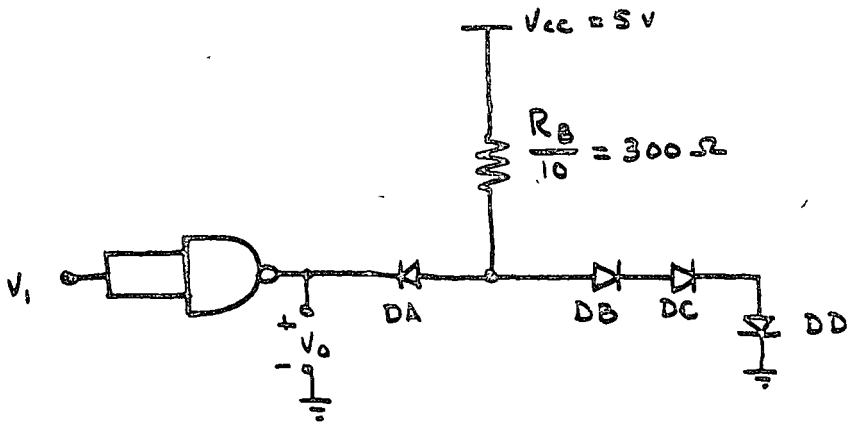


FIGURA 8.9

base colector de Q_1 y base emisor de Q_2 y Q_3. La resistencia se ha igualado a la resistencia $R_B/10$, para simular la carga de diez compuertas en paralelo.

Para analizar el circuito de supondrá primeramente que el voltaje de entrada es alto ($V_1 = V_2 = V_{cc}$).

Con esto, al eliminar el circuito a los elementos que queden cortados o apagados, queda el circuito de la figura 8.10

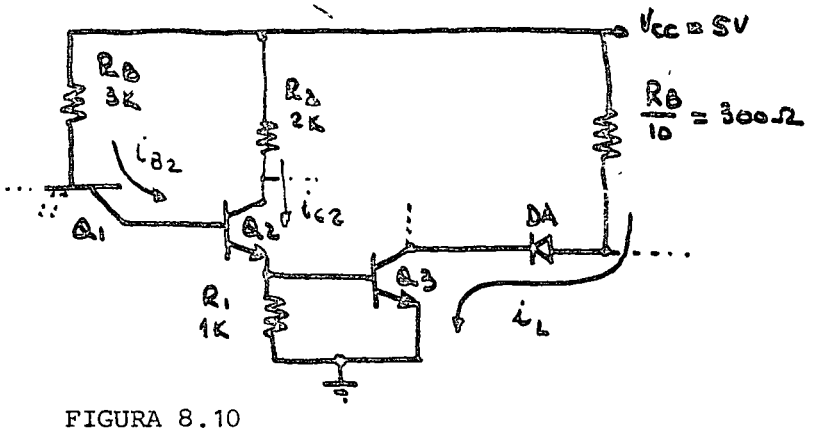


FIGURA 8.10

Obviamente, Q_2 y Q_3 están encendidos, y por lo tanto: $V_{B_3} = 0.6V$ y $V_{B_2} = 1.2V$. Con esto se conoce que $i_{B_2} = \frac{V_{cc} - V_{BC_1} - 1.2V}{R_B} \simeq 1mA$.

Con esta corriente de base, Q_2 queda saturado, por lo que el voltaje en

voltaje en su colector es aproximadamente de 0.8V. Por lo anterior se puede encontrar que $i_{C_2} = \frac{4.2V}{2K} = 2.1mA$. Como la corriente a través de R_1 es $0.6V/R_1 = .6mA$, la corriente a la base Q_3 será:

$$i_{B_3} = i_{C_2} + i_{B_2} - \frac{0.6V}{R_1} \cong 2.5\ mA$$

La corriente de carga está dada por:

$$i_L = \frac{V_{cc} - V_{D_1} - V_{CE_3}(sat)}{\frac{R_B}{10}} = \frac{4.2V}{.30K} \cong 14mA$$

ya que $\beta i_{B_3} > i_L$, se comprueba que Q_3 está saturado. Con esto, el voltaje a través de la base-emisor de Q_4 y D_1 es $V_{C_2} - V_{C_3} = 0.6V$, lo que asegura que Q_4 está apagado.

Para encender a la unión base-emisor de Q_1, se requiere que $V_1 = V_2 \cong V_{BE_3} + V_{BE_2} + V_{BC_1}) - V_{BE_1}$, o sea del orden de 1.2V. A partir de entonces, el circuito tiende a apagarse en función de la ganancia de Q_3. Cuando Q_3 se apaga, la salida depende de Q_4 - D_1, los cuales a su vez dependen del voltaje V_{C_2}. Este voltaje, tiene un valor conocido una vez que se apaga Q_3, ya que al no llevar éste corriente de base, la corriente en el colector de Q_2 es casi igual a la de su emisor, lo que a su vez está dada por:

$$i_{C_2} \cong \frac{V_{B_3}}{R_1}$$

Con esto se tiene que:

$$V_{C_2} \cong V_{CC} - V_{B_3}\frac{R_2}{R_1}$$

Al acabarse de apagar Q_3, su voltaje base emisor es muy poco inferior a 0.6V, por lo que en ese punto $V_{C_2} \cong 5 - 0.6\ \frac{2}{1} = 3.8V$. En este punto, el voltaje de salida vale: $V_0 = 3.8V - V_{BE_4} - V_{D_1} = 2.6V$. Este es el punto 2 de la figura 8.11.

A partir de este momento, la ganancia del circuito está dada por el cociente R_2/R_1, hasta que Q_2 y el diodo base-colector de Q_1 se apagan. Esto sucede cuando el voltaje de entrada es un poco menor que: $V_1=V_2\cong V_{BC_1} + V_{BE_2} - V_{BE_1} \cong 0.6V$. Esto representa el punto 3 en la gráfica de la fig. 8.11

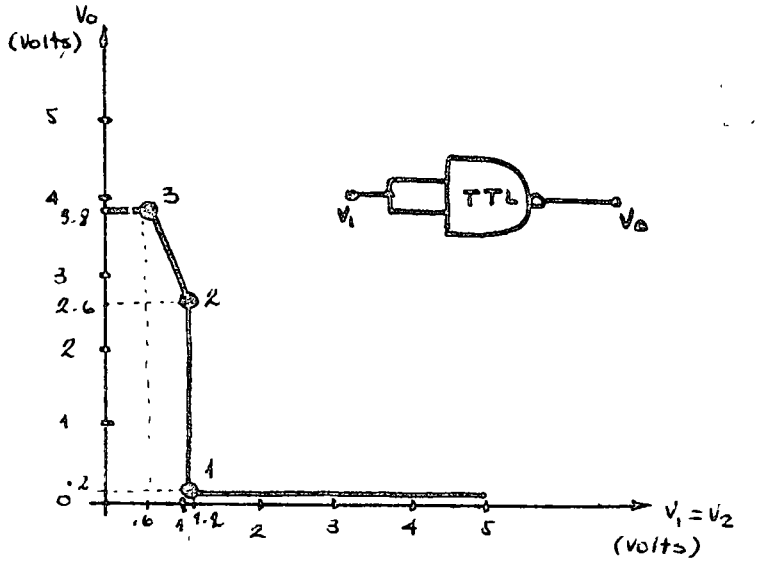


FIGURA 8.11

Para tensiones menores a ésta, el voltaje de salida permanece aproximadamente en $V_0(\S) = V_{cc} - V_{BE_4} - V_{D_1} = 3.8V$.

Conociendo los voltajes límites de la compuerta se puede calcular el margen de ruido. Aquí existe, sin embargo, la disyuntiva de escoger el punto 2 ó el 3 de la gráfica como el 1 lógico de la compuerta. Si se escoge el 2, se tendrá que: $NM(1) \cong 0.9V$ y $NM(0) \cong 1.5V$; en camibio si se escoge el 3, se tiene que $NM(1) = .54$ y $NM(0) = 2.6V$.

Nuevamente se hace notar que el ejemplo anterior deja de lado consideraciones de segundo orden que son de máxima importancia para el diseñador del circuito. Algunas de estas consideraciones son: variaciones en los parámetros de los transistores, efecto de la temperatura y aspectos referentes al tiempo de conmutación y disipación de potencia del circuito.

8.1. 3.- STTL (Schottky TTL)

Esta compuerta emplea en añadidura un elemento llamado diodo Schottky, cuyo símbolo se muestra en la figura 8.12a. Este diodo tiene como características fundamentales las siguientes: es fabricable en forma integrada junto con transistores bipolares, su tiempo de propagación es muy inferior al de un diodo de unión PN, y su voltaje de encendido es del orden de 0.4V. Con estas características, el principal uso de este diodo es el de fijador de voltaje para evitar la saturación de los transistores. Al

evitar esta saturación, aumenta considerablemente la velocidad de propagación de la señal en estas compuertas.

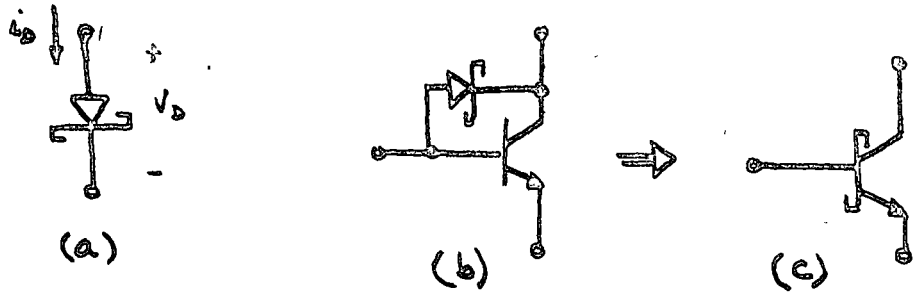


FIGURA 8.12

El uso extensivo del transistor con fijador Schottky ha hecho que se utilice el símbolo que se muestra en la figura 8.12c. Empleando esta simbología la figura 8.13 muestra una compuerta STTL de las más recientes.

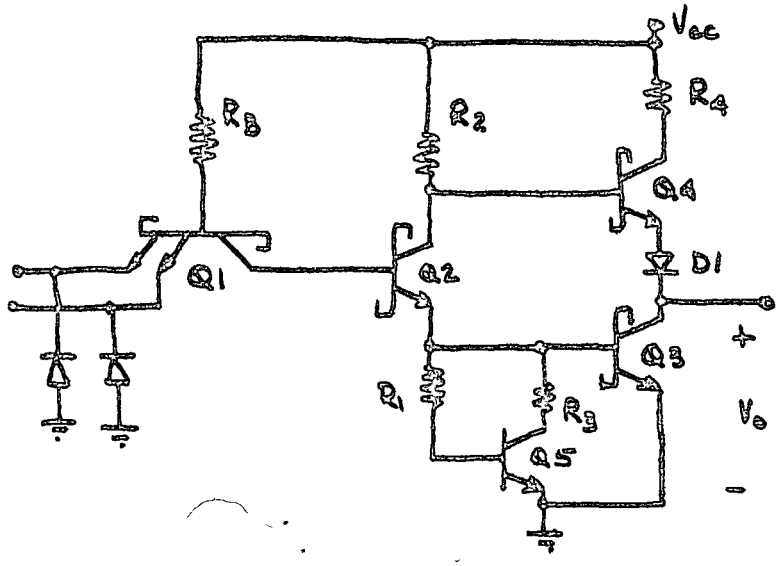


FIGURA 8.13

Las principales diferencias entre este circuito y el TTL típico son:

a) El uso extensivo de fijadores Schottky, con el objeto de reducir el tiempo de transición del circuito.

b) Distintos valores de resistencias con el propósito de reducir la potencia disipada en el circuito, así como presentar una impedancia más apropiada a las líneas de 50Ω.

c) La carga "activa" Q5-R_1-R_3, que aumenta la ganancia de Q_2, haciendo más empinada la función de transferencia (entre los puntos 2 y 3 de la gráfica de la figura 8.11), aumentando el margen de ruido.

Al ir aumentando su volumen de ventas, la familia STTL ha ido reduciendo sus precios y aumentando la diversidad de sus funciones, estimando que en un futuro próximo STTL dominará a TTL en su empleo dada sus mejores características de conmutación.

El producto P X D del STTL es del orden de 110 pJoules, mientras que su retardo típico es de 5 nseg.

8.2.- Familia Lógica CMOS (Complementary Metal-Oxide-Semiconductor)

La única familia lógica con transistores MOS que se puede usar en forma MSI y SSI es la familia CMOS. Los circuitos de MOS canal simple se emplean principalmente en circuitos de gran escala (LSI), y su funcionamiento interno es "transparente" al usuario, es decir que la entrada y la salida se parecen a las de las familias de uso común (particularmente TTL).

La familia CMOS en cambio, tiene características particulares que la hacen materia de estudio en estecurso. Por un lado son accesibles comercialmente a través de varios fabricantes, y por otro lado ofrecen una alternativa muy significante a las familias bipolares (particularmente TTL), dando una mayor flexibilidad al arte de diseñar circuitos digitales.

Evitando entrar a detalles de fabricación y diseño, en los párrafos siguientes se describe a un inversor CMOS como el de la figura 8.14.

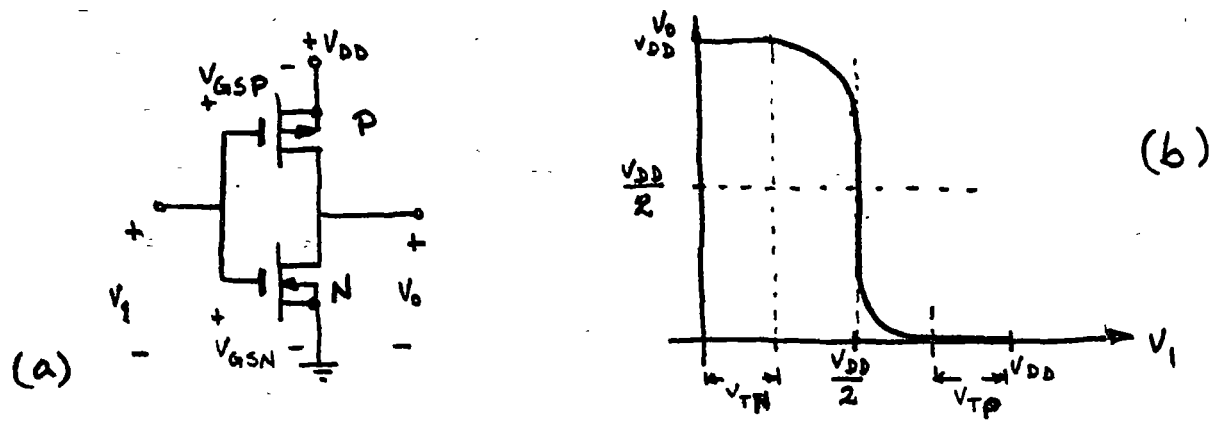


FIGURA 8.14

Cuando el voltaje de entrada es bajo ($V_I = 0$), el MOS canal N está apagado y el canal P está encendido ($V_{GSP} = -V_{DD}$). En estas condiciones, el voltaje de salida es $V_0 = V_{DD}$, siempre y cuando la "carga" al circuito sea otro circuito CMOS (en cuyo caso la impedancia de carga es muy alta).

Cuando el voltaje de entrada es igual al voltaje de umbral del MOS canal N (V_{TN}), dicho transistor empieza a encenderse, "cargando" al MOS canal P (ver punto (2) en la figura 8.14B).

Al seguir subiendo el potencial a la entrada, el voltaje de salida disminuye: primero en forma ligera, y muy abruptamente cuando $V_I \cong V_{DD}/2$ (ver puntos (3) y (4) en la figura 8.14b). Para $V_I > V_{DD}/2$, el voltaje de salida disminuye paulatinamente hasta llegar a cero cuando $V_I \cong V_{DD} - V_{TP}$, en cuyo caso el MOS canal P se apaga.

Varias características interesantes son deducibles de la operación del inversor CMOS. Primero, que su margen de ruido es amplio, ya que se puede considerar cuando menos del orden de $V_{TP} \cong V_{TN}$, que para la mayoría de los dispositivos comerciales es del orden de 1.5 Volts; en todo caso es posible considerar que el margen de ruido sea parecido a $V_{DD}/2$ (tanto para el 0 como para el 1), siempre y cuando se sea cuidadoso en el diseño del sistema. Segundo, que en los extremos lógicos la corriente que se drena de la fuente de alimentación es mínima, ya que será solo la corriente de fuga de los dispositivos, la cual es del orden de las nanoamperes; esto significa una disipación de energía sumamente bajo en estado estático (no confundir con la energía que se gasta en el caso dinámico). Tercero que la impedancia de entrada del circuito es sumamente alta (característica inherente al dispositivo MOS), lo que permite un "Fan-out" ilimitado.

Otra importante característica de este inversor en su alta flexibilidad en cuanto al valor de la fuente de alimentación (V_{DD} puede variar desde 3 volts hasta 15 volts en la mayoría de los dispositivos comerciales).

La disipación dinámica del dispositivo depende de la carga capacitiva del mismo (C), de la frecuencia a que opera (f_0) y de la fuente de alimentación (V_{DD}) en la siguiente relación:

$$P(\text{dinámica}) = \frac{1}{2} V_{DD}^2 f_0 C_L$$

Para la familia CMOS, el producto P x D es del orden de 1 nJ para $C_L \cong 20pF$ y $V_{DD}=10V$ o 250 pJ para $V_{DD}=5V$ y $C_L=20pF$. La velocidad de propagación de las compuertas CMOS es típicamente de 20 a 50 nseg para las condiciones de carga y alimentación señaladas antes.

Así es que en general, la flexibilidad de la familia CMOS se ve compensada por su baja velocidad, además de ser relativamente fragil, ya que la carga estática del cuerpo humano puede destruir a un circuito integrado MOS, si no se tiene la precaución debida.

8.2.1.- Compuertas CMOS.

Basadas en el funcionamiento del inversor CMOS se pueden generar compuertas NAND y NOR, así como interruptores (también llamadas compuertas de transmisión). Una breve descripción de estas compuertas siguen en las secciones posteriores.

8.2.1.1.- Interruptor o compuerta de transmisión.

Una característica importante del dispositivo MOS es su bilateralidad. A diferencia del bipolar, el circuito MOS, una vez que está encendido, puede conducir en ambas direcciones; es decir que su Drenaje y su Compuerta son intercambiables. Esta propiedad se muestra en la figura 8.15, en la que aparece la característica voltaje-corriente de un MOS canal N.

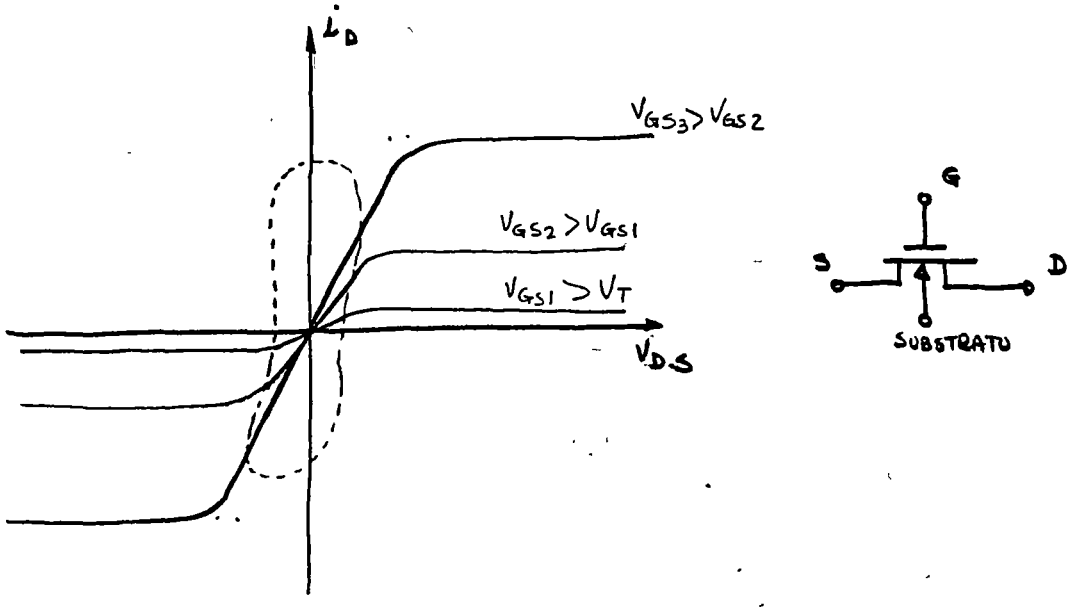


FIGURA 8.15

Mientras al potencial en la compuerta sea mayor que el potencial en el drenaje y en la fuente, el dispositivo estará encendido. Además, para bajos voltajes entre drenaje y fuente, el MOS se comporta como una resistencia (ver zona punteada en la figura 8.15).

Una compuerta de transmisión CMOS se realiza en la forma mostrada en la figura 8.16 a.

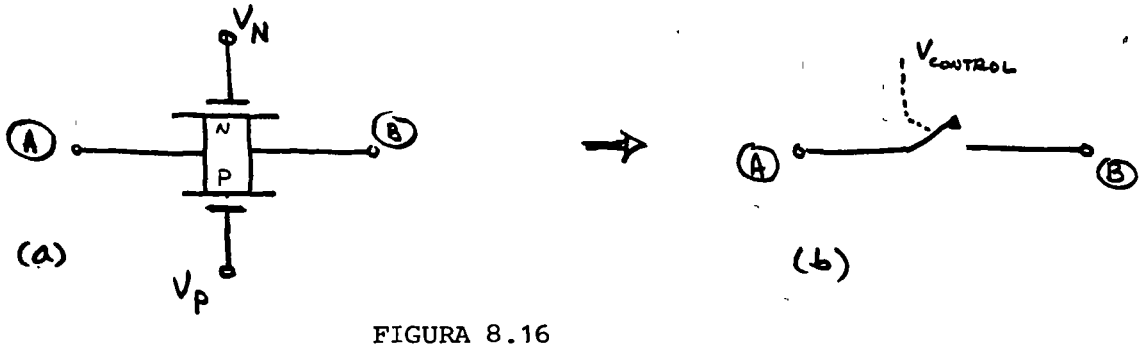


FIGURA 8.16

Cuando $V_N = V_{DD}$ y $V_p = 0$, la compuerta está encendida y puede transmitir (o sea que equivale a un interruptor cerrado). Cuando $V_N = 0$ y $V_p = V_{DD}$ la compuerta está inhibida (o sea el interruptor está abierto). En la figura 8.16b se muestra el modelo equivalente, o sea un interruptor que es controlado por un voltaje V_C.

En la familia CMOS es muy conocido el circuito 4016 que consiste de cuatro interruptores con su respectivo control. Cada interruptor tiene el circuito mostrado en la figura 8.17. El inversor CMOS adicional se emplea para asegurarse de que V_p y V_N están siempre en estados lógicos opuestos.

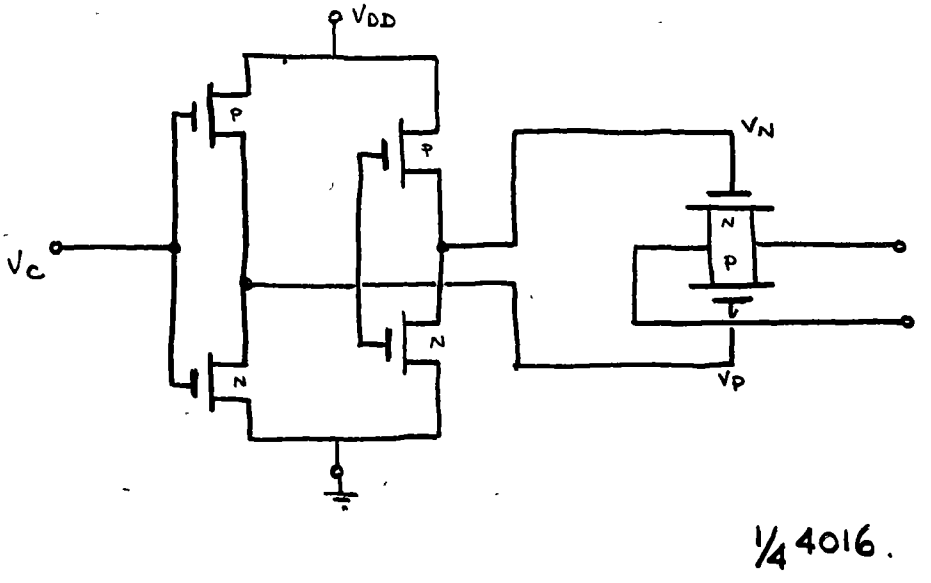


FIGURA 8.17

8.2.1.2.- Compuerta NAND.

La compuerta NAND se muestra en la figura 8.18. Su funcionamiento estático es como sigue. Cuando $V_1 = V_2 = 0$, M1 y M2 están apagados, mientras que M3 y M4 están encendidos, con lo que $V_0 = V_{DD}$. En virtud de que M1 y M2 están conectadas en serie, la única forma en la que la salida pude ser 0 es si ambas están encendidas; por esto es que solo cuando $V_1 = V_2 = V_{DD}$, se tiene que $V_0 = 0$,

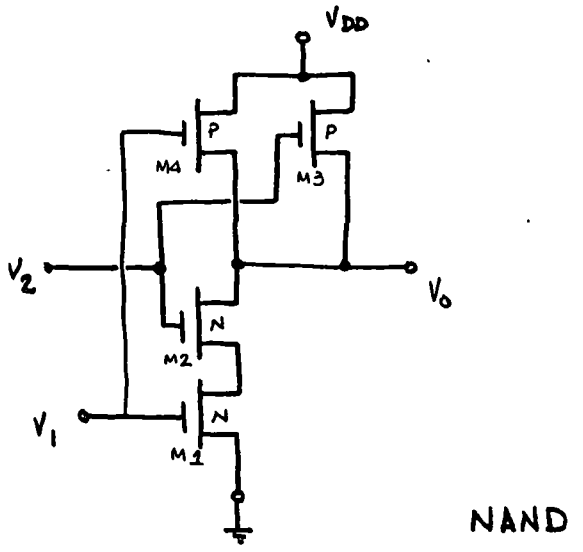


FIGURA 8.18

8.2.1.3.- Compuerta NOR

Esta compuerta es el dual de la anterior y se muestra en la figura 8.19. En ésta, los dispositivos del canal N (M1 y M2) están en paralelo, por lo que su salida será baja siempre que alguna de las entradas sea alta. La salida será alta solamente cuando $V_1 = V_2 = 0$.

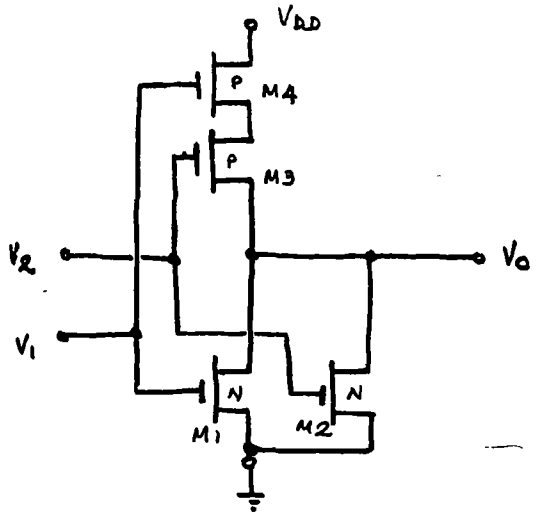


FIGURA 8.19

8.4.- INTERCONEXION.

8.4.1.- El concepto de BUS.

En muchas ocasiones se desea minimizar el cableado de interconexión en un sistema digital. En estas ocasiones es común usar un cable al que se conectan las salidas de varias compuertas y las entradas de otras compuertas. El objetivo es usar este cable para transmitir información de una sola salida a una o más entradas. Por supuesto, durante el tiempo en que una transmisión dada sea efectuada, las demás compuertas no deberán interferir; por lo tanto es necesario inhibir la acción de las compuertas que no deben estar enviando información y de aquellas que no deban recibir dicha información.

En la figura 8.20 se muestra un arreglo en el que tres transmisoras se comunican con dos receptoras.

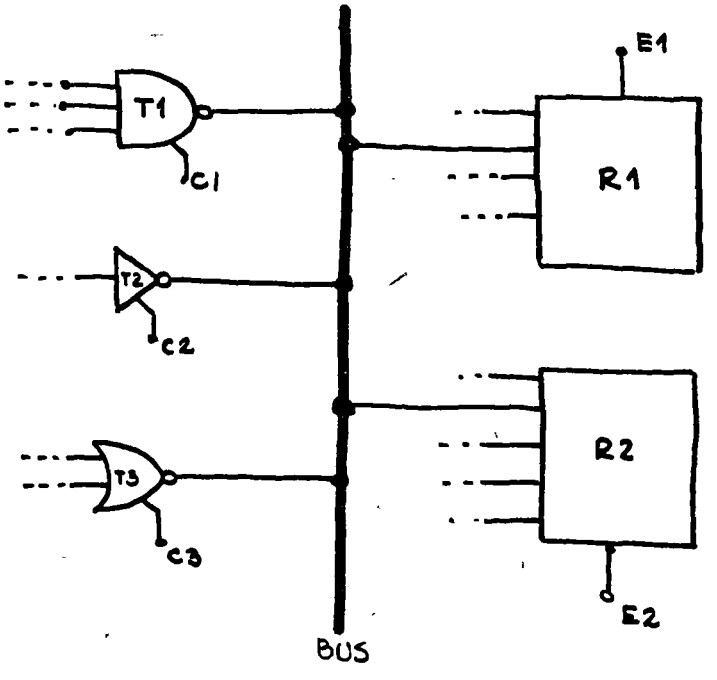


FIGURA 8.20

La acción transmisora del elemento Ti se inhibe con la señal de Control C_i, mientras que la acción receptora del elemento R_i se inhibe con el control E_i.

Supóngase que un cero en la señal de control inhibe, y que un UNO habilita. Entonces para comunicar a T2 con R1, se deben tener las señales de control: C1 = 0, C2 = 1, C3 = 0, E1 = 1 y E2 = 0. Si por el contrario, se desea que T3 transmita tanto a R1 como a R2, se deberá tener: C1 = 0, C2 = 0, C3 = 1, E1 = 1 y E2 = 1.

De esta forma, a expensas de tener que compartir una ruta de transmisión, se puede reducir la interconexión en muchos circuitos. Una de las principales desventajas de este sistema es su lentitud, ya que mientras ocurre una transmisión, las demás deben esperar. Por supuesto, debe existir algún procesador que controle el uso del BUS, es decir, que opere las señales de control de transmisión y recepción.

Desde el punto de vista de los circuitos, existe un importante problema para la implementación de un BUS. Si una compuerta digital solo puede cambiar entre un CERO y un UNO, al conectar varias compuertas a la salida se puede tener un estado indeterminado. Esto pasaría muy claramente si se conectasen directamente a la salida dos compuertas TTL con totem-pole o bien dos compuertas CMOS del tipo descrito anteriormente.

La solución común es de dos tipos:

a) usar compuertas del tipo "colector abierto"

b) usar compuertas especiales denominadas "TRI-STATE". A continuación se describe el principio de funcionamiento de estas compuertas.

8.4.2.- Interconexión a través de "colector abierto".

En la figura 8.21 se muestran tres circuitos con "colector abierto", conectados a la misma línea de BUS.

Cuando las señales de control son un 0 lógico, los transistores se encuentran apagados y por lo tanto, su colector-emisor equivale a un circuito abierto. En este estado, la línea se encuentra "en reposo" con un estado alto (o sea un 1 lógico). La señal se transmite por la línea de BUS , sin que los otros dos intervengan. Por supuesto, no se puede permitir que dos señales de control sean 1 simultáneamente, ya que en ese caso la señal en el BUS es el "AND" de ambas señales.

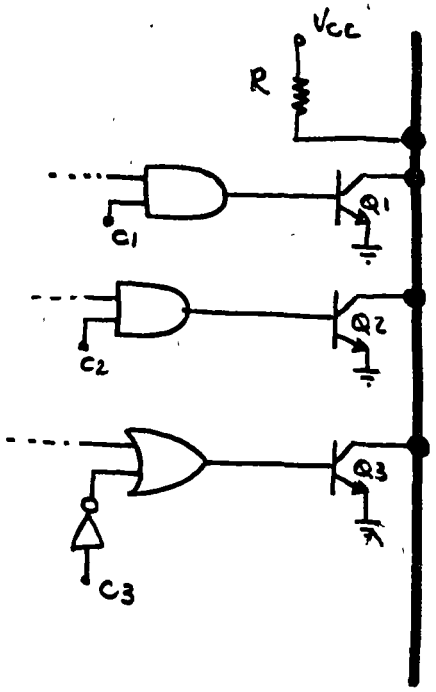


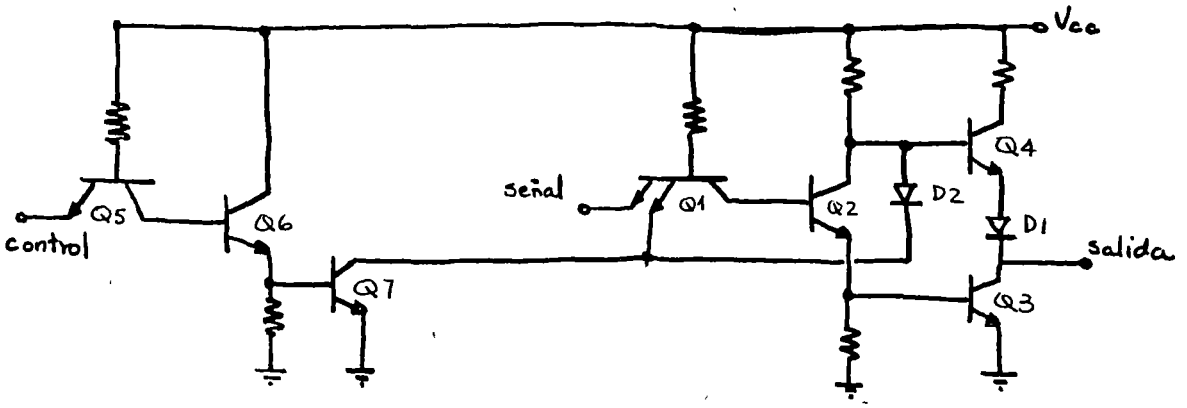
FIGURA 8.21

8.4.3.- TRI-STATE.

El sistema a base de "colector abierto" tiene algunos problemas en lo referente a potencia, retardo y número de receptores a que puede dar servicio. Ciertas modificaciones a las compuertas TTL (totem-pole) y CMOS, permite realizar la interconexión al BUS. Estas modificaciones convierten a estas compuertas en las llamadas TRI-STATE debido a que tienen tres estados : los dos lógicos (1 y 0) y un estado que equivale un circuito abierto y que es llamado "de alta impedancia"

8.4.3.1.- TTL/TRI-STATE.

La figura 8.22 muestra la modificación más comúnmente usada por los fabricantes de TTL para convertir compuertas al tipo TRI-STATE.
Cuando la señal de control es un 1, el transistor Q7 se halla encendido, con lo que tanto Q3 como Q4-D1 se encuentran apagados. Al estar encendido Q7, equivale a tener un cero en la compuerta NAND/TTL, lo que produce que Q3 se apague. En añadidura, a través del diodo D2, Q4-D1 son apagados. En este estado la salida del totem-pole equivale a un circuito abierto (excepción hecha de la corriente de fuga de Q3, que es normalmente inferior a 40μA), es decir que queda en el estado de alta impedancia.



TTL/TRI-STATE.

FIGURA 8.22

Cuando el control está en un nivel CERO, el transistor Q7 está apagado y la señal se transmite normalmente a la salida.

8.4.3.2.- CMOS/TRI-STATE.

La forma más común de usar un CMOS en TRI-STATE es añadiendo una compuerta de transmisión a la salida de las compuertas lógicas comunes. Cuando la compuerta de transmisión se abre se tiene el estado de alta impedancia; y cuando ésta se cierra, se puede tener alguno de los estados lógicos. La figura 8.23 muestra este circuito para un inversor.

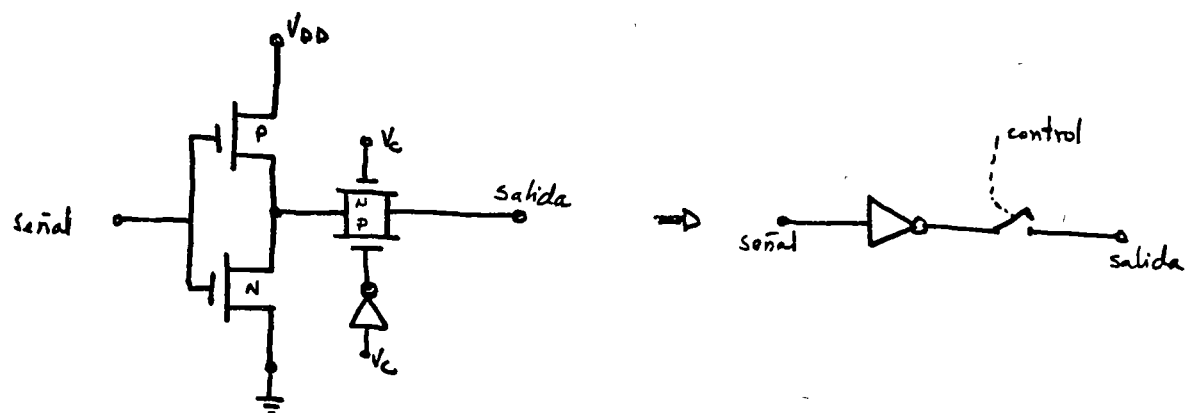


FIGURA 8.23

Una alternativa consiste en añadir elementos en serie con los transistores MOS de salida, como se muestra en la figura 8.24. Cuando los dispositivos en serie se encuentran apagados (C=0), el circuito se encuentra en el estado de alta impedancia.

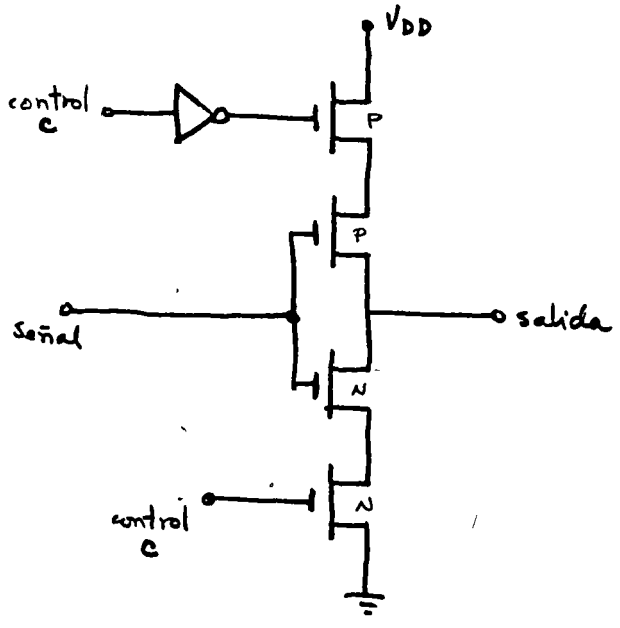


FIGURA 8.24

centro de educación continua
división de estudios superiores
facultad de ingeniería, unam

INTRODUCCION AL PROCESAMIENTO DIGITAL

CAPITULO 8: FAMILIAS LOGICAS E INTERCONEXION

Dr. José F. Albarrán Núñez

Septiembre, 1977

A los alumnos del curso: "INTRODUCCION AL PROCESAMIENTO DIGITAL"

Tengo el placer de incluir en las notas que corresponden al Capítulo 8 ("Familias lógicas e interconexión), una parte de la versión preliminar del libro escrito por el Dr. Isaac Schnadower y editado por Mc Graw Hill de México: "Circuitos Electrónicos Digitales".

Este libro representa el primer esfuerzo serio sostenido en nuestro país por publicar un libro de categoría internacional, si bien adecuado a las necesidades particulares de nuestro medio. Indudablemente que la trayectoria científica del Dr. Schnadower, así como su importante experiencia docente, han fructificado en un magnífico libro de texto y consulta, que pone de manifiesto el grado de desarrollo que la electrónica está adquiriendo en nuestro país.

Agradezco la gentileza de Mc Graw Hill de México y del Dr. Schnadower, al permitir la reproducción parcial de dicho libro. Finalmente, me permito recordarles que toda reproducción de estas notas queda prohibida, excepto con el consentimiento escrito de la casa editora.

Atentamente,

JOSE F. ALBARRAN

4.- COMPUERTAS LÓGICAS CON TRANSISTORES BIPOLARES.

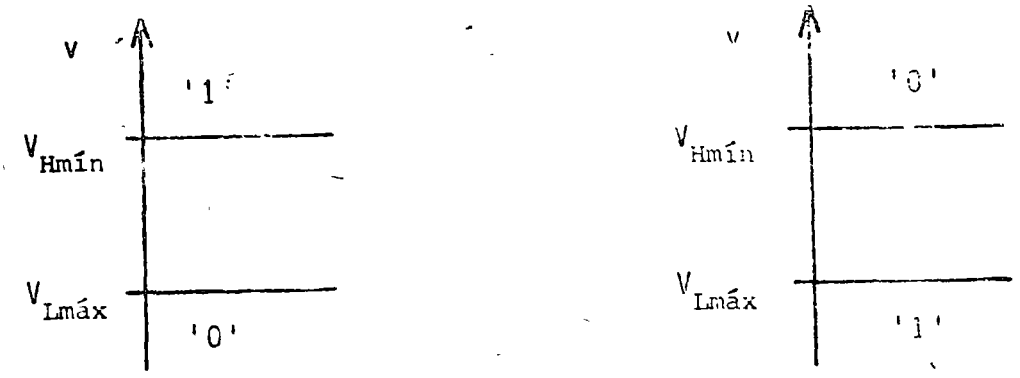
En el presente capítulo se estudiará el comportamiento de circuitos de conmutación no regenerativos requeridos para realizar funciones lógicas. Estos circuitos, y los circuitos regenerativos, que se tratarán en capítulos posteriores, son básicos para el funcionamiento de los computadoras y equipo de medición y control digital.

4.1.- Señales y circuitos lógicos.

Los circuitos digitales procesan información binaria, respondiendo a un conjunto de excitaciones con dos niveles definidos, que corresponden a estados lógicos '1' y '0'. Se habla de lógica positiva cuando al estado lógico '1' corresponde un voltaje alto, mayor que cierto valor $V_{H\text{mín}}$, y el estado '0' corresponde a un voltaje bajo, menor que $V_{L\text{máx}}$. En sistemas con lógica negativa, el '1' corresponde al nivel alto del voltaje y el '0' al bajo (Fig. 4-1-1).

---

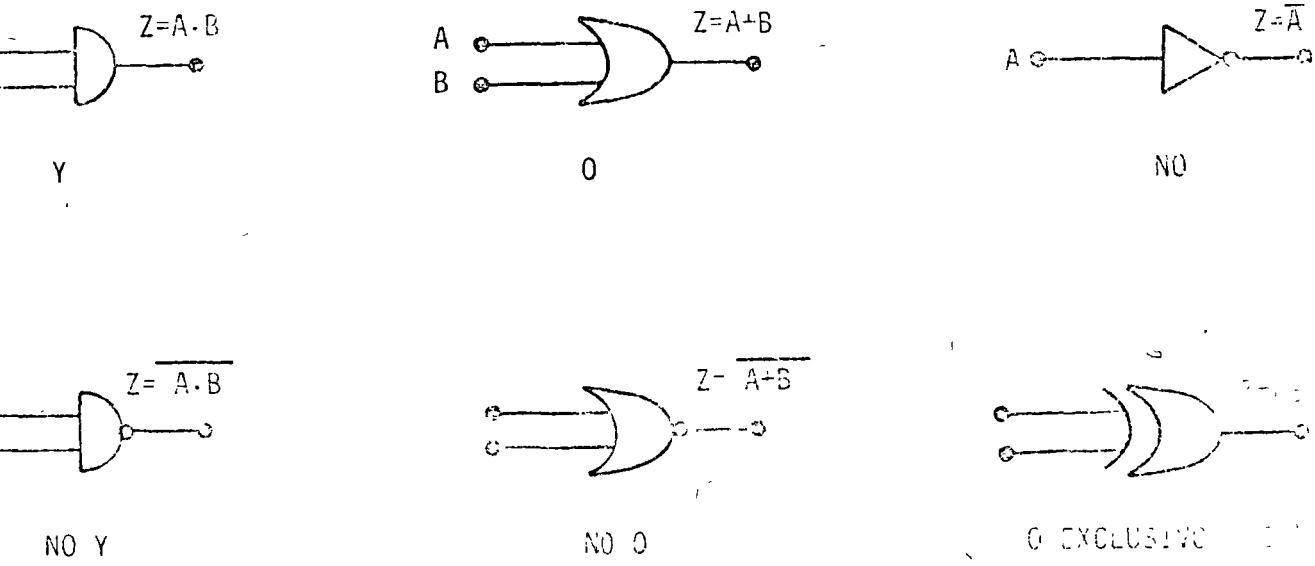
* También llamados sistemas binarios).



lógica positiva lógica negativa

Figura 4-1-1.- Niveles lógicos.

Una compuerta es un circuito para realizar operaciones lógicas con una o más variables (entradas) obteniendo un resultado (salida Z). Las operaciones más comunes se encuentran definidas en la Fig. 4-1-2, utilizando en cada caso una tabla de verdad, que indica el resultado de la operación correspondiente.



Y O NO

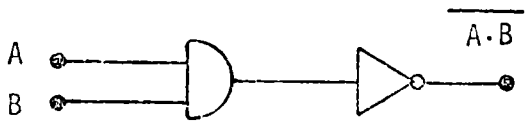
NO Y NO O O EXCLUSIVO

A B	Y A · B	O A + B	NO Y $\overline{A \cdot B}$	NO O $\overline{A + B}$	OEX $\overline{A \oplus B}$
0 0	0	0	1	1	0
0 1	0	1	1	0	1
1 0	0	1	1	0	1
1 1	1	1	0	0	0

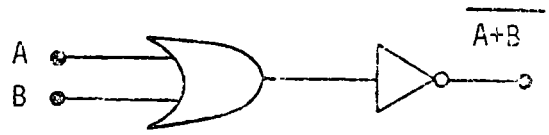
A	NO Z = $\overline{A}$
0	1
1	0

Figura 4-1-2.- Operaciones lógicas más comunes y sus tablas de verdad.

La función NO opera sobre una sola variable, y da como resultado un valor lógico invertido o complementado. La operación NO Y produce justamente la función Y negada, y puede implementarse como se muestra en la Fig. 4-1-3a., efectuando la inversión después de la operación Y.



NO Y



NO O

Figura 4-1-3.- Implementación de las funciones NO Y y NO O.

Al aplicar la operación Y, el valor de la variable de salida será '1' si todas las variables de entrada valen '1'. La operación O, en cambio, produce un '1' si al menos una de las variables de entrada vale '1'. La operación OEX, aplicable a 2 variables de entrada solamente, da como resultado '1' siempre y cuando una sola variable sea '1'.

Si representamos a cada variable lógica por un interruptor simple de contacto, asociando el estado o valor '1' al interruptor cerrado, y el '0' al interruptor abierto, y si definimos los valores '1' y '0' de la salida Z por los estados 'prendido' y 'apagado' de un foco incandescente, resulta por demás sencillo que las operaciones Y, O y NO pueden realizarse como lo muestra la Fig. 4-1-4.

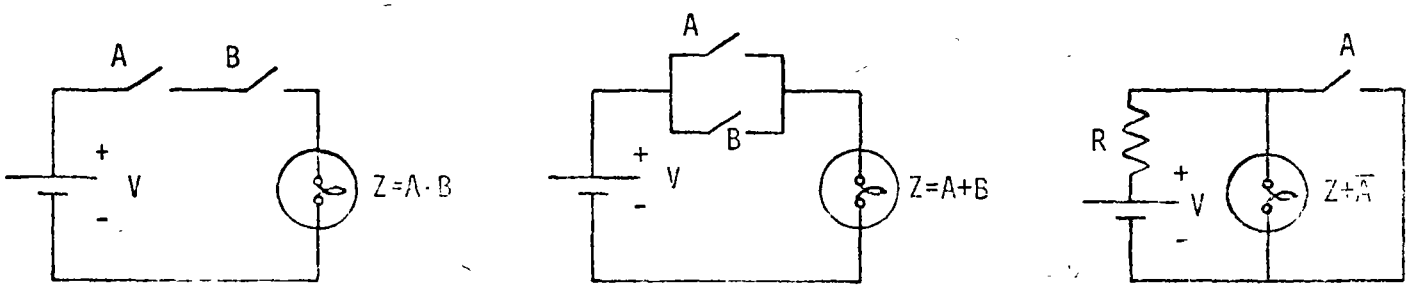


Figura 4-1-4.- Implementación de las operaciones Y, O y NO por medio de interruptores.

Los valores lógicos resultantes de funciones lógicas como $Z = A \cdot (A \cdot B + C) + B$ pueden obtenerse, además de utilizar una tabla de verdad, aplicando los postulados y teoremas del algebra de Boole, desarrollada por el matemático George Boole en conexión con el estudio de la lógica formal:

1.- Propiedades del '0'	$'0' + A = A \quad , \quad '0' \cdot A = '0'$
2.- Propiedades del '1'	$'1' + A = '1' \quad , \quad '1' \cdot A = A$
3.- Leyes conmutativas	$A+B = B+A \quad , \quad A \cdot B = B \cdot A$
4.- Leyes asociativas	$A + (B+C) = (A+B) + C$ $A \cdot (B \cdot C) = (A \cdot B) \cdot C$
5.- Leyes distributivas	$A \cdot (B+C) = A \cdot B + A \cdot C$ $A + (B \cdot C) = (A+B) \cdot (A+C)$
6.- Reglas de idempotencia	$A + A = A \quad , \quad A \cdot A = A$
7.- Propiedades del complemento	$A + \overline{A} = '1' \quad , \quad A \cdot \overline{A} = '0'$ $\overline{(\overline{A})} = A \quad , \quad '\overline{1}' = '0'$
8.- Leyes del complemento - (de De Morgan)	$\overline{A + B} = \overline{A} \cdot \overline{B}$ $\overline{A \cdot B} = \overline{A} + \overline{B}$

Las leyes del algebra de Boole permiten simplificar expresiones lógicas, a fin de realizarlas en la forma más económica posible (con un número mínimo de compuertas, por ejemplo). Supongamos que se pretende realizar la función de 3 variables $Z = A \cdot (A \cdot B + C) + B$; el proceso de simplificación puede llevarse a cabo como sigue:

$Z = A \cdot A \cdot B + A \cdot C + B$ por (5)

$Z = A \cdot B + A \cdot C + B$ por (6)

$Z = A \cdot B + A \cdot C + B \cdot 1$ por (2)

$Z = A \cdot B + A \cdot C + B \cdot (A + \overline{A})$ por (7)

$Z = A \cdot B + A \cdot C + B \cdot A + B \cdot \overline{A}$ por (5)

$Z = A \cdot B + A \cdot C + \overline{A} \cdot B$ por (6,3)

$Z = (A + \overline{A}) \cdot B + A \cdot C$ por (5)

$Z = B + A \cdot C$ por (2,7)

La función simplificada, $Z = B + A \cdot C$, puede implementarse como se muestra en la Fig. 4-1-5.

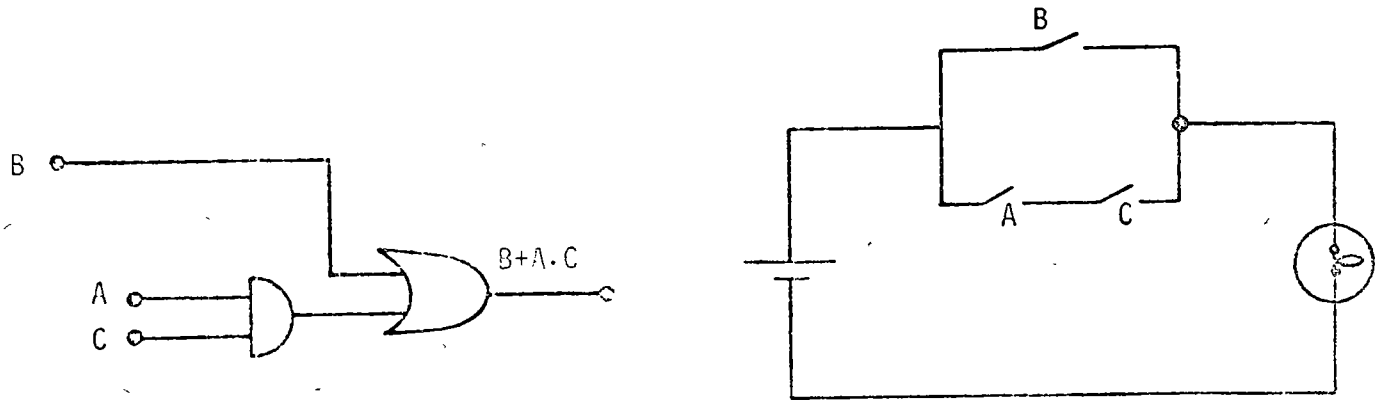
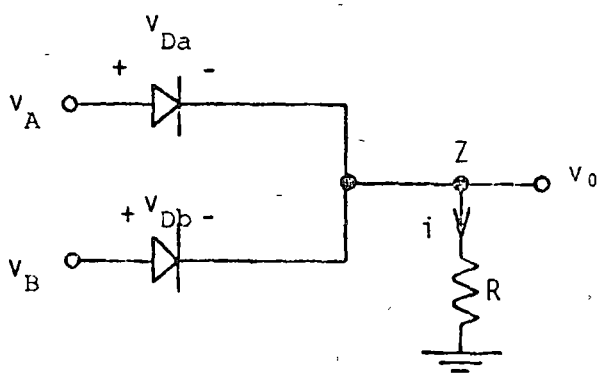


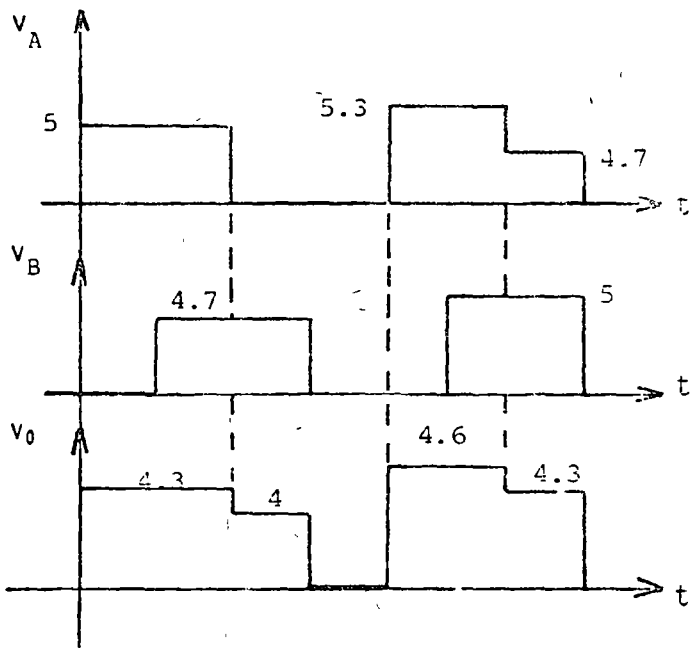
Figura 4-1-5.- Realización de la función $Z = B + A \cdot C$.

Funciones de mediana complejidad (hasta de 6 variables) suelen simplificarse por métodos gráficos, utilizando los llamados mapas de Karnaugh. Para simplificar funciones más complejas, es conveniente seguir métodos sistemáticos, requiriéndose en ocasiones implementarlos en una computadora. Estos temas se mencionan extensamente en la literatura, y no se tratarán en el presente texto.

4.2.- Compuertas pasivas con diodos.

En la Fig. 4-2-1a. se muestra un circuito selector de voltaje máximo; la salida, v_o, es igual al voltaje máximo de entrada, $V_{máx}$, menos el voltaje del diodo en conducción correspondiente v_D, si $V_{máx}$ es mayor que su voltaje de arranque, V_a. De lo contrario, v_o es aproximadamente igual a 0 V. Esto es, la salida será baja, de valor lógico '0' sólo si todas las entradas son bajas, y alta ('1' lógico) si al menos un voltaje de entrada es alto, como se muestra en la Fig. 4-2-1b., y corresponde por tanto a la salida de una compuerta O.





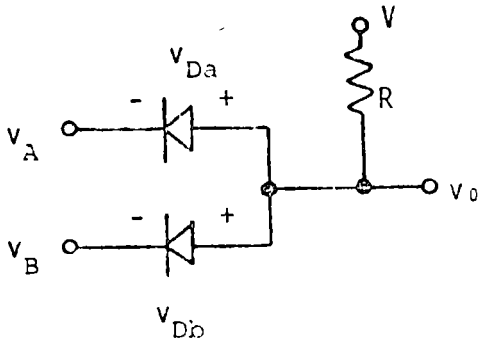
$v_0 = \text{máx}\ (v_A, v_B) - v_D$

$Z = A + B$

Figura 4-2-1.- Compuerta 'O' con diodos.

Con $v_D = 0.7$ V para cualquier diodo en conducción, el valor '1' lógico de la salida fluctúa entre 4 y 4.6 V. - La corriente i por R es igual a v_0/R; si conducieran dos diodos simultáneamente fluiría por cada uno de los mismos - una fracción de la corriente total. Nótese que el valor - máximo del voltaje de entrada bajo, $V_{Lmáx}$, no debe exceder de 0.5 a 0.6 V; de lo contrario se arrancaría indebidamente a alguno de los diodos.

La operación Y puede realizarse utilizando un selector de voltaje mínimo, como el circuito de la Fig. 4-2-2. La salida v_0 excede en v_D volts al menor de los voltajes de entrada, si dicho voltaje es menor de $V - v_D$ volts. Así, conque una sola de las entradas sea baja ('0'), la salida será baja; sólo si todas las entradas son altas ('1'), la salida será alta. Aún en este último caso, es conveniente que conduzca algún diodo, a fin de obtener una resistencia baja de salida; de lo contrario existirán serios problemas de acoplamiento.



$v_0 = \min (V_a, V_b) + v_D$

Figura 4-2-2.- Compuerta 'Y' con diodos.

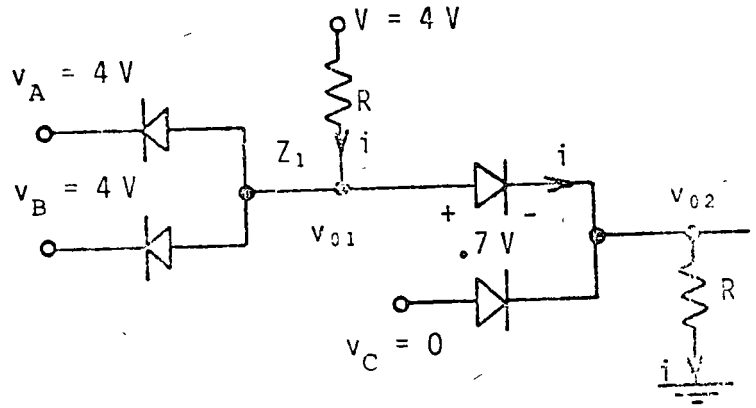


Figura 4-2-3.- Implementación de la función $Z_2 = A \cdot B + C$.

En la Fig. 4-2-3 se muestra una compuerta Y excitando a una compuerta O. Si $v_A = v_B = 4\,V$, y $v_C = 0\,V$, los diodos de la compuerta Y se encuentran cortados. Despreciando sus corrientes de fuga, $v_{02} = 4 - Ri - 0.7 = Ri$; luego $i = 1.65/R$, y $v_{02} = 1.65\,V$, voltaje considerablemente menor que el valor estándar de 4V para el estado lógico '1'.

Con un voltaje de batería $V = 5V$, sin embargo, la operación mejora notablemente, pues los diodos de la compuerta Y conducen, y $v_{01} = 4.7\,V$ aproximadamente; por tanto, $v_{02} \simeq 4\,V$. La batería debe entregar una corriente substancialmente mayor que en el caso anterior. Nótese que las caídas de voltaje a través de los diodos de las compuertas Y y O quedan canceladas; al conectar varias compuertas del mismo tipo en cascada, en cambio, el nivel del voltaje de salida estará desplazado respecto a los de entrada. Tres compuertas Y en cascada, por ejemplo, podrían producir un voltaje de salida 2.1 a 2.3 V mayor que el menor de los voltajes de entrada.

Es mediante el uso de elementos activos que se puede obtener un conjunto de niveles de voltaje más uniformes, y la importante operación NO (inversión).

4.3.- Estructura y características de las compuertas lógicas.

Las compuertas que se emplean actualmente presentan una estructura como la que se muestra en la Fig. 4-3-1a.

La red E de entrada permite seleccionar un voltaje, mismo que controla a un interruptor; el interruptor se realiza normalmente por un circuito activo, tan sencillo como un transistor en emisor común (inversor), que permite la complementación o inversión (operación NO) del voltaje de control, como se muestra en la Fig. 4-3-1b.

Se obtienen así compuertas NO Y y NO O; las funciones Y y O se realizan mediante el circuito de entrada E, por medio de diodos, resistores o transistores.

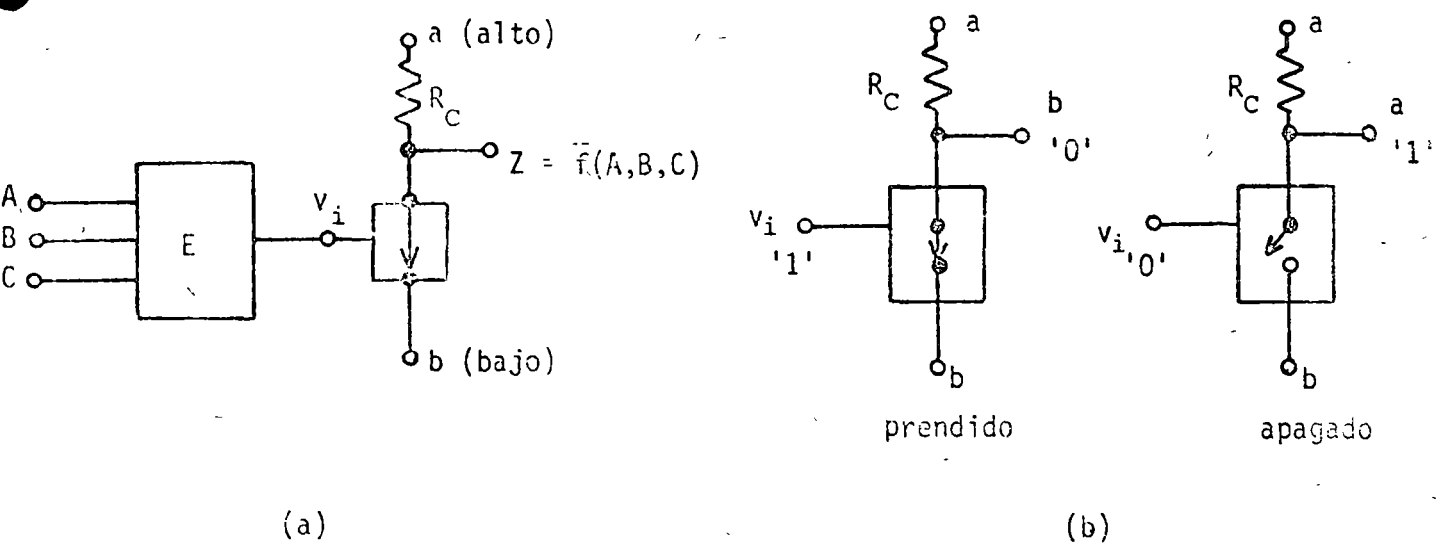


Figura 4-3-1.- Estructura de una compuerta con un inversor.

Otras configuraciones posibles se muestran en la Fig. 4-3-2. En la primera de ellas se utilizan interruptores en paralelo, conectados a un voltaje alto a través de un resistor común, R_c. Para obtener una salida baja, basta conque una de las entradas sea alta, lográndose por tanto la operación NO O para lógica positiva, que estamos utilizando en el texto. La segunda configuración utiliza interruptores en serie, y se requiere que ambas entradas sean altas a fin de que la salida sea baja (NO Y).

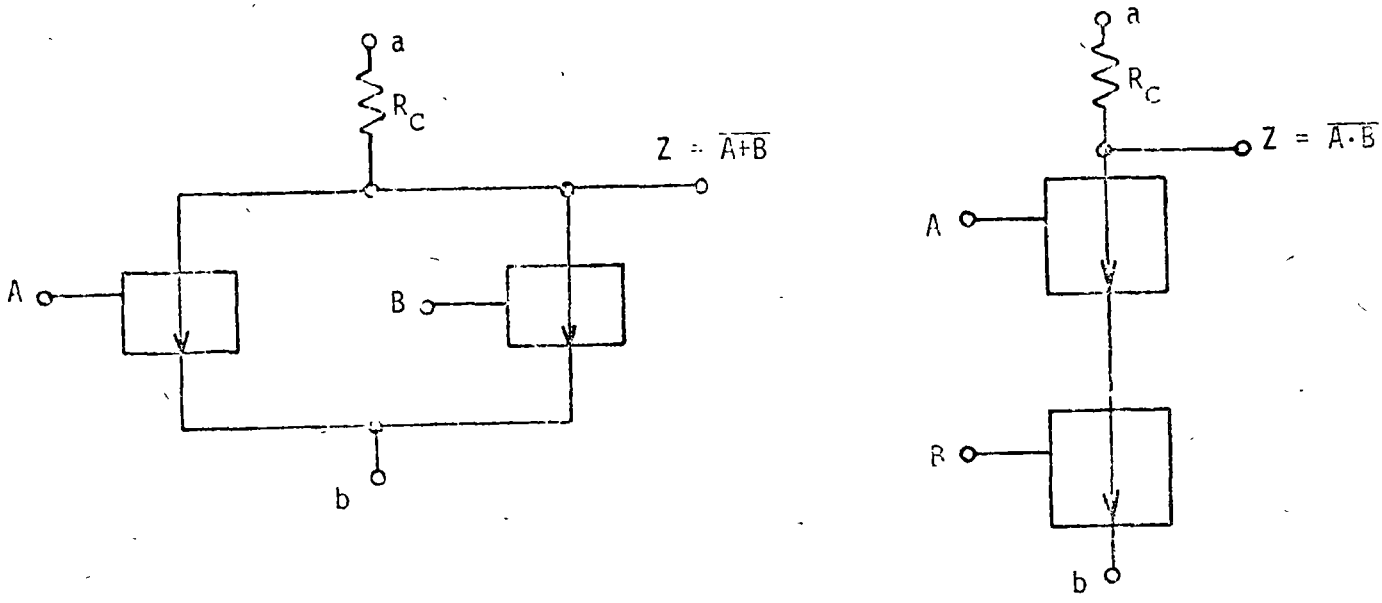
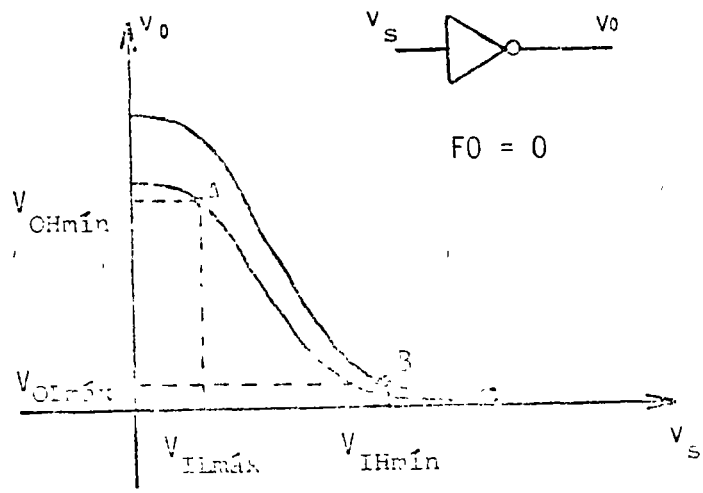


Figura 4-3-2.- Estructuras con inversores en paralelo y en serie.

Características de transferencia.- En general, la característica de transferencia de una compuerta es una función no lineal de los voltajes de entrada: $v_0 = f(v_A, v_B, v_C, \ldots)$. En el caso particular de un inversor, como el de la Fig. 3-4-1, la salida es una función de una sola entrada, v_s. Debido a variaciones de temperatura, voltajes de fuentes y valores de parámetros y componentes, no es posible referirse a una característica de transferencia única; sin embargo, al considerar casos extremos de las variaciones (los peores), es posible definir las características límites o envolventes, que dependen de la carga de la compuerta.

Esta carga se especifica como el número de compuertas del mismo tipo excitadas simultáneamente por la compuerta en cuestión, y se denomina 'abanico de salida' (en inglés, FAN-OUT, abreviado FO). Este valor afecta a las envolventes de la característica de transferencia, como se muestra en la Fig. 4-3-3.



(6)

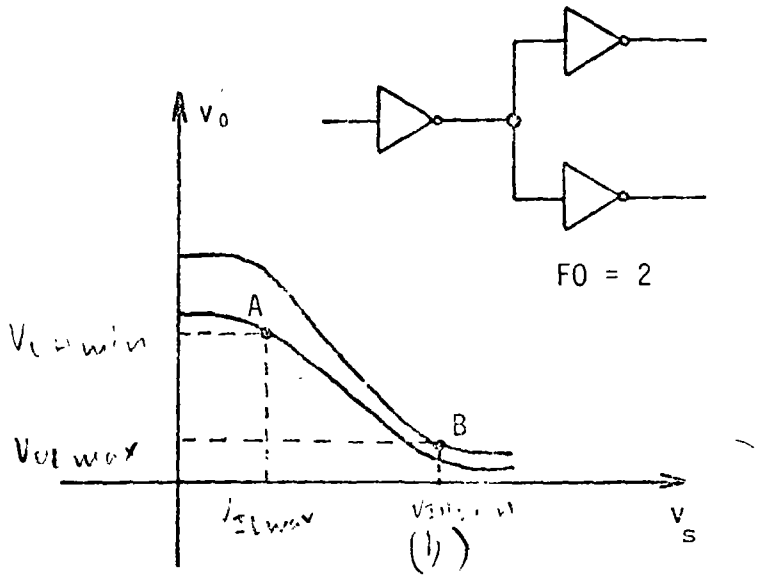


Figura 4-3-3.- Envolventes de la función de transferencia de un inversor.

En la rodilla superior de la envolvente inferior, donde la pendiente es igual a 1, se definen las coordenadas del punto $A = (V_{OHmín}, V_{ILmáx})$ como:

$V_{OHmín}$ = voltaje mínimo permisible de salida en estado alto.

$V_{ILmáx}$ = voltaje máximo permisible de entrada en estado bajo.

En el punto B de la envolvente superior, de pendiente igual a 1, se tienen las coordenadas

$V_{OLmáx}$ = voltaje máximo permisible de salida en estado bajo.

$V_{IHmín}$ = voltaje mínimo permisible de entrada en estado alto.

Es claro entonces que los niveles de voltaje para el '1' y el '0' están limitados por los valores:

	Entrada	Salida	
'1'	$v_s > V_{IHmín}$	$v_0 > V_{OHmín}$	4-3-1
'0'	$v_s < V_{ILmáx}$	$v_0 < V_{OLmáx}$	

a fin de garantizar el correcto funcionamiento del inversor aún en el peor de los casos. Dado que el voltaje v_0 de salida de un inversor (u otro tipo de compuerta) es a su vez el voltaje de entrada de otros inversores (o compuertas). se requiere que

$$V_{OHmín} > V_{IHmín} \quad y \quad V_{OLmáx} < V_{ILmáx} \qquad 4\text{-}3\text{-}2$$

Margen de ruido.- En todo sistema existe ruido; éste puede provenir del exterior, o bien ser generado interna

mente (por acoplamiento capacitivo o inductivo entre líneas del sistema, ruido de componentes, etc.). Cualquiera que sea su naturaleza, dicho ruido puede añadirse a una señal lógica y provocar la operación incorrecta de una o varias compuertas, ocasionando incluso la falsa aparición de un '1' por un '0' o viceversa.

El margen de ruido, NM, es una medida del ruido que puede tolerarse sin que se alteren los niveles lógicos '1' y '0'. El margen de ruido para el estado bajo, por lo tanto, resulta

$$NM_0 = V_{ILmáx} - V_{OLmáx} \qquad 4\text{-}3\text{-}3$$

y el margen de ruido para el estado alto resulta

$$NM_1 = V_{OHmín} - V_{IHmín} \qquad 4\text{-}3\text{-}4$$

En la Fig. 4-3-4 se muestran los niveles, destacándose gráficamente los márgenes de ruido para los estados bajo y alto.

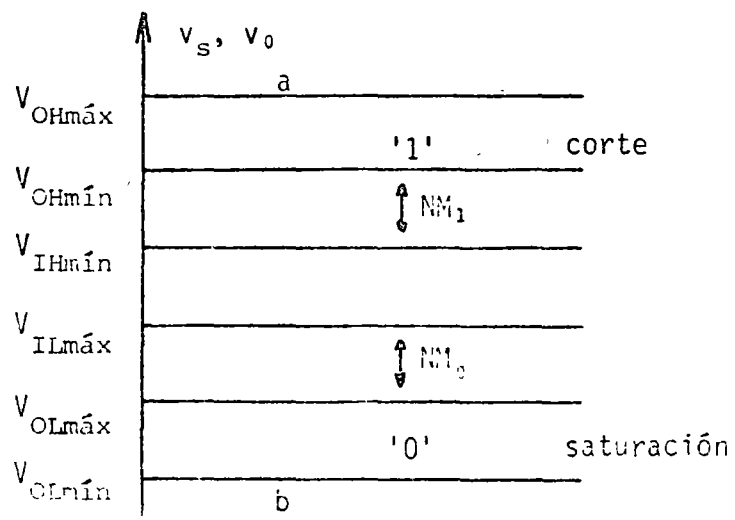


Figura 4-3-4.- Niveles lógicos y márgenes de ruido

Cabe aclarar que, para una compuerta con M entradas, es posible especificar las envolventes de la función transferente utilizando a la compuerta como un inversor; para una compuerta NO Y, por ejemplo, M-1 entradas se excitan con un nivel alto ('1'); para una compuerta NO O, M-1 entradas se excitan con un nivel bajo, ('0').

Tiempo de propagación.- El tiempo de propagación, t_p, indica el tiempo requerido para que un pulso se propague por un inversor, y se define como la diferencia entre los tiempos t_b y t_a requeridos para que tanto la salida como la entrada lleguen al 50% de sus respectivos valores finales (Fig. 4-3-5). Considerando que el tiempo de encendido, t_{on}, difiere por lo general del tiempo de apagado, t_{off}, t_p varía de acuerdo a si el nivel cambia de '1' a '0' ó de '0' a '1'.

Por consiguiente, se utilizan dos etapas para determinar t_p*:

$$t_p = \frac{t_c - t_a}{2} \simeq \frac{t_d + t_r/2 + t_s + t_f/2}{2} \qquad 4\text{-}3\text{-}5$$

Si, como es frecuente, $t_r >> t_d$, y $t_s >> t_f$, entonces

$$t_p \simeq \frac{t_{on}}{4} + \frac{t_{off}}{2} \qquad 4\text{-}3\text{-}6$$

---

* Ver capítulo 3.

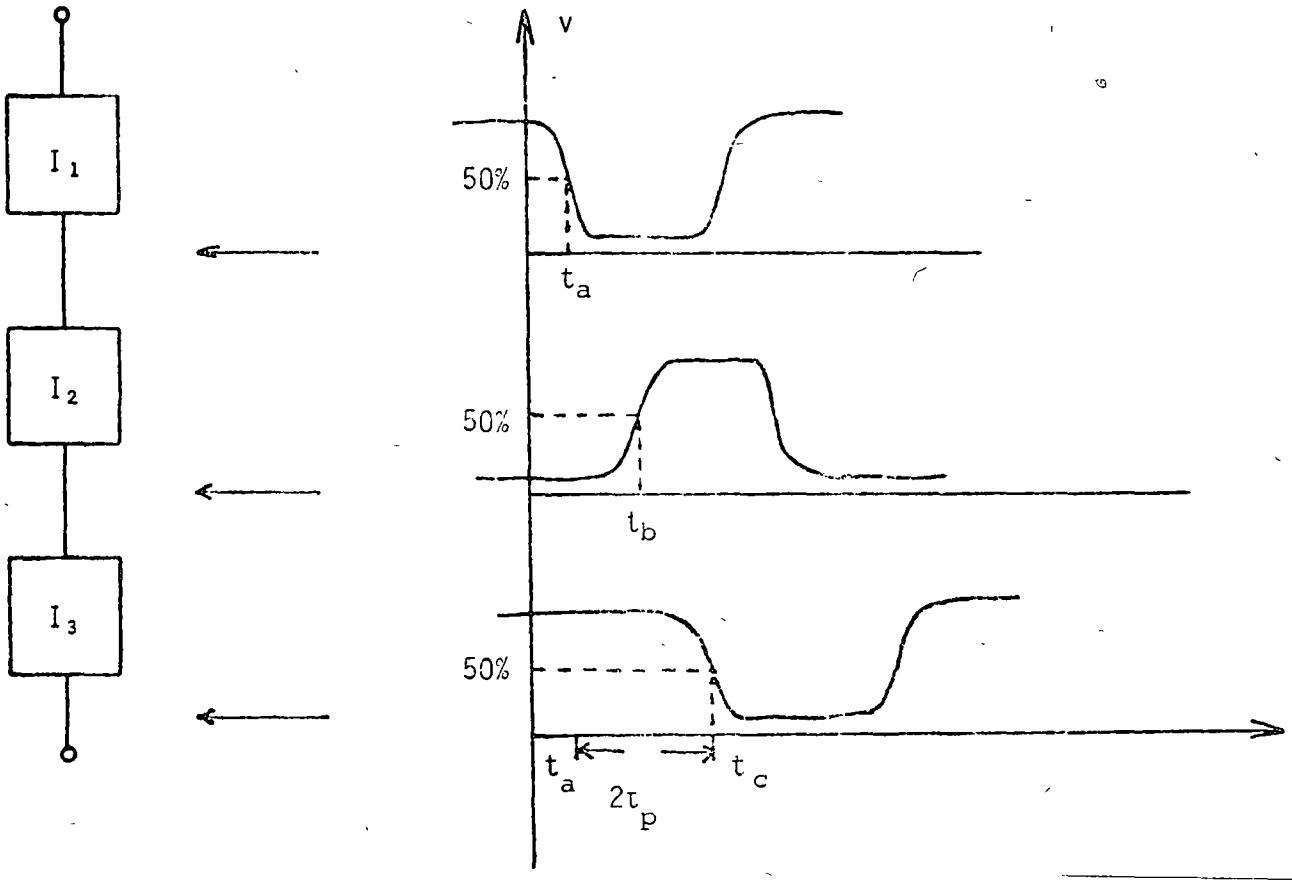


Figura 4-3-5.- Tiempo de propagación.

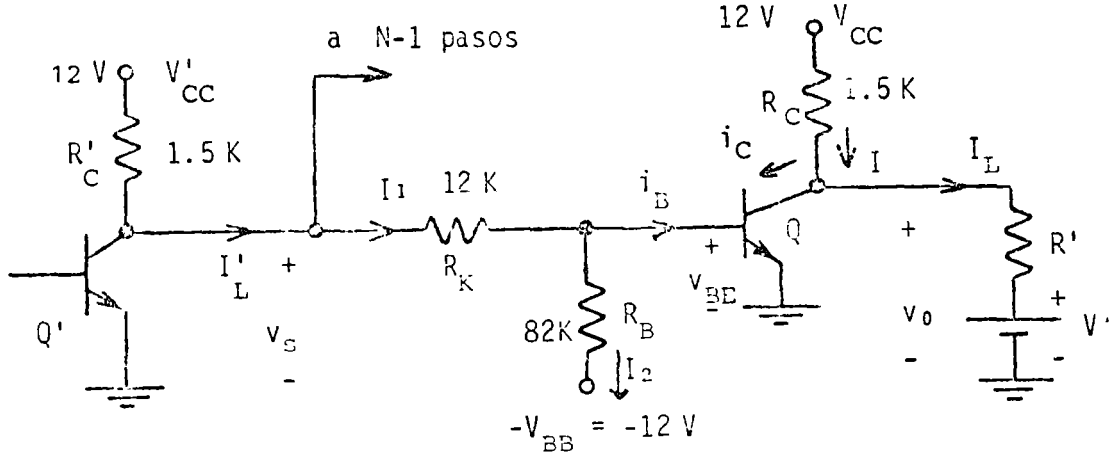
El tiempo de propagación limita el ancho de los pulsos, y por tanto, la frecuencia de operación (velocidad).

Otras características importantes de las compuertas, son: El consumo o disipación de potencia, factor de peso - al considerar el diseño de un sistema digital con un número grande de componentes, costo, baja generación de ruido y - flexibilidad lógica, que se refiere a la posibilidad de realizar convenientemente distintas operaciones lógicas.

Las compuertas pueden diseñarse con componentes discretos, aunque actualmente se utilizan extensamente en circuitos integrados.

4-4.- Análisis y diseño de un inversor RTL discreto.-

A fin de ilustrar los conceptos expuestos en la sección anterior, procederemos a analizar un inversor RTL, que significa lógica de resistores y transistores. El inversor Q de la Fig. 4-4-1 es idéntico al de la Fig. 3-4-1, sólo que consideramos ahora un circuito excitador (otro inversor con N salidas), y la carga de Q, representada por una resistencia R' y un voltaje V', que dependen del estado de Q: R'_L y V'_L cuando el inversor se encuentra saturado (nivel lógico '0'), y R'_H, V'_H cuando el inversor está cortado (nivel lógico '1')'



$V_{be} \simeq 0.7$ V, $V_{ces} \simeq 0.2$ V, $\beta_{min} = 30$, T = 27°C, $I_{CO} = 10^{-9}$ A.

Figura 4-4-1.- Inversor RTL discreto.

Ecuaciones para el caso en que Q' está cortado, y Q saturado.

$$i_C = I_{csat} \quad , \quad v_0 = V_L = V_{ces} \quad , \quad v_{BE} = V_{be}$$

$$\frac{v_s - V_{be}}{R_k} = \frac{V_{be} + V_{BB}}{R_B} + i_B \qquad 4\text{-}4\text{-}1$$

$$\frac{V_{CC} - V_{ces}}{R_C} = I_{csat} + I_L = I_{csat} + \frac{V_{ces} - V'_L}{R'_L} \qquad 4\text{-}4\text{-}2$$

Ecuaciones para el caso en que Q' está saturado, y Q cortado.

$$v_0 = V_H \quad , \quad v_s = V_{ces} \quad , \quad v_{BE} = V_{off} < 0 \quad , \quad i_B = -i_C = -I_{CO}$$

$$\frac{V_{ces} - V_{off}}{R_k} = \frac{V_{off} + V_{BB}}{R_B} - I_{CO} \qquad 4\text{-}4\text{-}3$$

$$\frac{V_{CC} - V_H}{R_C} = I_{CO} + I_L = I_{CO} + \frac{V_H - V'_H}{R'_H} \qquad 4\text{-}4\text{-}4$$

Las ecuaciones anteriores pueden resolverse siempre y cuando se conozcan los valores de v_s, R' y V'. Si suponemos que ambos inversores son idénticos, y excitan cada -

uno de ellos a N inversores idénticos, se tendrá entonces - que

$$I_L = NI_1 \qquad 4\text{-}4\text{-}5$$

y la carga de cada inversor equivale a la combinación en - paralelo de N redes de entrada. Para un inversor cortado, los inversores de carga se encuentran saturados, y los valo_ res R'_H, V'_H, I_L y V_H (voltaje alto del inversor cortado), pueden determinarse del circuito equivalente, mostrado en - la Fig. 4-4-2.

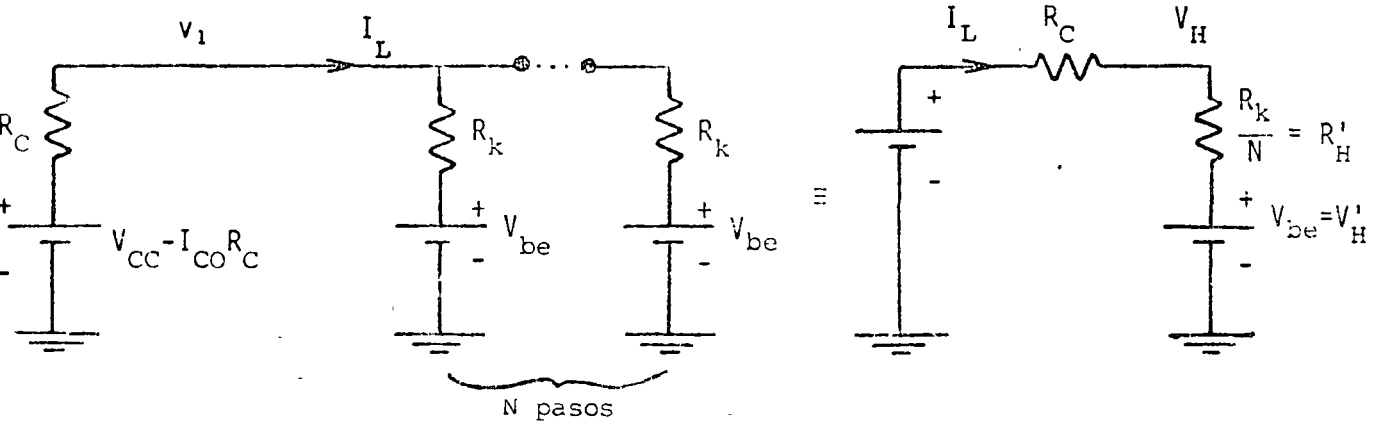


Figura 4-4-2.- Inversor cortado con un - abanico de salida = N.

Esto es, $R'_H = R_k/N$, $V'_H = V_{be} \approx 0.7\ V$, y V_H resulta

$$V_H = V_{be} + \frac{R_k\,(V_{CC} - I_{CO}R_C - V_{be})}{R_k + NR_C} \qquad 4\text{-}4\text{-}6$$

Sustituyendo este valor por v_s en la Ec. 4-4-1, ésta puede reescribirse como

$$I_1 = \frac{V'_{CC} - R'_C I_{CO} - V_{be}}{NR'_C + R_k} = \frac{V_{be} + V_{BB}}{R_B} + i_B \qquad 4\text{-}4\text{-}7$$

de donde se obtiene el valor de i_B. Para un paso saturado, los valores de R'_L y V'_L pueden obtenerse con ayuda del circuito equivalente 4-4-3, obteniéndose

$$R'_L = \frac{R_k + R_B}{N} \qquad 4\text{-}4\text{-}8$$

$$V'_L = -V_{BB} + R_B I_{CO} \qquad 4\text{-}4\text{-}9$$

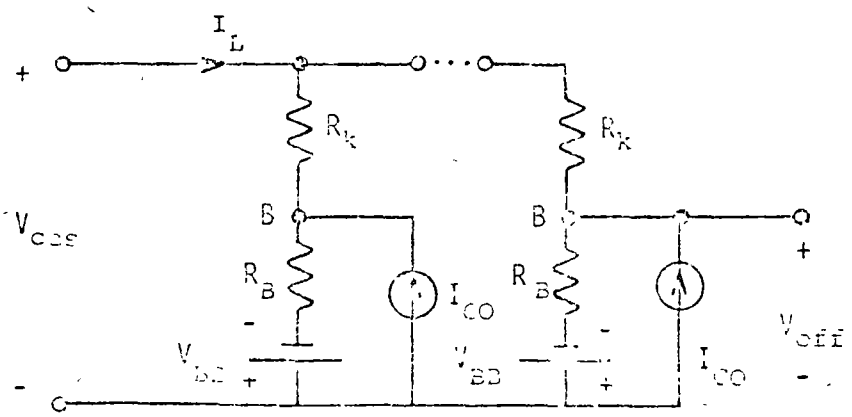


Figura 4-4-3.- Inversor saturado con FO - N.

La ecuación 4-4-2 aplicada ahora a Q, resulta

$$\frac{V_{CC} - V_{ces}}{R_C} \simeq I_{csat} + N \frac{V_{ces} + V_{BB} - R_B I_{CO}}{R_k + R_B} \qquad 4\text{-}4\text{-}10$$

Ejemplo 4-4-1.- Evaluar el funcionamiento del inversor Q sin carga ($FO_Q = 0$), excitado por Q' con un abanico de salida $FO_{Q'} = 1$, con valores nominales para las resistencias, fuentes y parámetros de los transistores a 27°C.

Solución.- Verifiquemos primeramente que Q esté cortado si Q' está saturado; de 4-4-3, despreciando I_{CO}

$$V_{off} \simeq (R_k || R_B) \left(\frac{V_{ces}}{R_k} - \frac{V_{BB}}{R_B}\right) = -1.06\ V$$

luego Q está cortado. Procedemos ahora a verificar que Q esté efectivamente saturado con Q' cortado; de 4-4-10, con $N = 0$:

$$I_{csat} = \frac{V_{CC} - V_{ces}}{R_C} = 7.867\ mA.$$

Por otro lado, de 4-4-7, con $N = 1$, obtenemos i_B:

$$I_B \simeq \frac{V'_{CC} - V_{be}}{R'_C + R_k} - \frac{V_{be} + V_{BE}}{R_B} = 0.837 - 0.155 \simeq 0.68\ mA.$$

Dado que se requiere al menos una corriente $I_{bsat} = I_{csat}/\beta_{Fmín} = 0.26$ mA., el transistor Q se encuentra saturado, excitado por un voltaje $v_s = V_H$ que resulta, de 4-4-6 (con N = 1)

$$V_H \simeq V_{be} + \frac{R'_k\,(V'_{CC} - V_{be})}{R'_k + R_C} = 10.74\ V$$

Efecto del abanico de salida de Q'.- Supongamos ahora que incrementamos el abanico de salida de Q'. Tanto V_H como I_1 tienden a disminuir, hasta que la corriente de base i_B de Q no alcance a saturar al transistor Q (ocurriendo lo propio con otros pasos del abanico de salida de Q').

A fin de determinar el valor máximo permisible del abanico de salida, $\overline{FO}$, se consideran las peores condiciones, que en este caso son las que propician una corriente de colector máxima, $\overline{I_{csat}}$, y una corriente de base mínima. $\underline{i_B}$. Es necesario asegurar que incluso bajo tales condiciones extremas se cumpla la condición de saturación $\overline{I_{csat}} \leq \underline{\beta_F}\ \underline{I_B}$, o bien $\underline{i_B} \geq I_{bsat}$* .

---

* La barra superior indica valor máximo. La inferior indica valor mínimo.

I_{cs} es máxima si Q no está cargado, o sea, si $I_L = 0$ (no consideramos por el momento valores negativos de I_L). - Se requiere también que $V_{CC} = \overline{V_{CC}}$, $R_C = \underline{R_C}$ y $V_{ces} = \underline{V_{ces}}$, de donde

$$\overline{I_{csat}} = \frac{\overline{V_{CC}} - \underline{V_{ces}}}{\underline{R_C}} \qquad 4\text{-}4\text{-}11$$

Las condiciones bajo las cuales el valor de i_B de Q es mínima ocurrirán claramente cuando I_1 sea mínima, y - una gran proporción de la misma circule hacia la fuente - $-V_{BB}$ en vez de proceder a la base; para ello se requiere - además que la mayor proporción de la corriente I'_L de carga suministrada por la batería V'_{CC} circule por los N-1 pasos restantes del abanico de salida; se tiene por tanto - el circuito equivalente de la Fig. 4-4-4, de donde

$$\underline{i_B} = \frac{\dfrac{(\underline{V'_{CC}} - \overline{I_{CO}}\,\overline{R'_C} - \underline{V_{be}})\,\underline{R_k}}{\underline{R_k} + (N-1)\,\overline{R'_C}} + \underline{V_{be}} - \overline{V_{be}}}{(\overline{R'_C} \,||\, \frac{R_k}{N-1}) + \overline{R_k}} - \frac{\overline{V_{be}} + \overline{V_{BB}}}{\underline{R_B}} \qquad 4\text{-}4\text{-}12$$

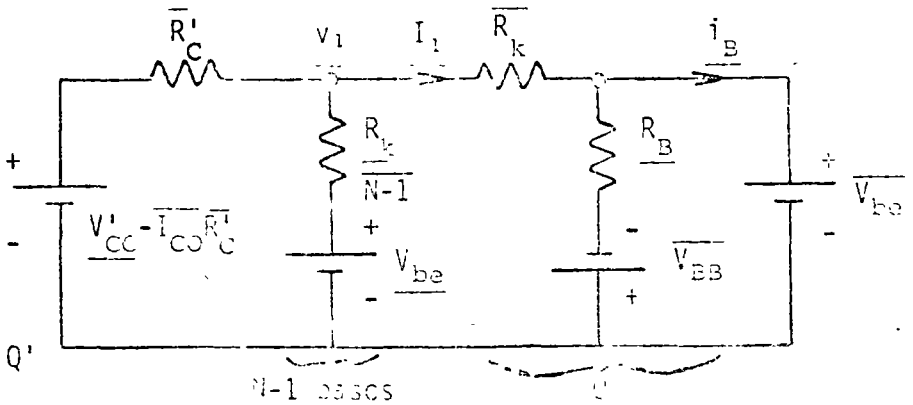


Figura 4-4-4.- Circuito que produce una corriente i_B mínima para Q.

Nótese que si consideramos N pasos iguales, obtendríamos, de 4-4-7, una expresión más sencilla, aunque no corresponda al caso extremo:

$$\underline{i_B} = \frac{\underline{V'_{CC}} - \overline{I_{CO}}\,\overline{R'_C} - \overline{V_{be}}}{N\,\overline{R'_C} + \overline{R_k}} - \frac{\underline{V_{be}} + \overline{V_{BB}}}{\underline{R_B}} \qquad 4\text{-}4\text{-}13$$

Es necesario aclarar, en todo caso, que en la práctica algunas condiciones extremas no podrían ocurrir en virtud de las inevitables correlaciones entre algunos parámetros. No podría esperarse, por ejemplo, grandes diferencias entre los valores de V_{CC} de dos compuertas montadas en la misma tablilla; por otro lado, los parámetros de los transistores, e incluso las resistencias de los resistores dependen de la temperatura. Para propósitos de análisis y diseño, por tanto, podrá ser necesario efectuar los cálculos bajo diversas condiciones de operación.

Supongamos que se permiten variaciones de $\pm$ 0.5 V para las fuentes, de $\pm$ 20% para los resistores, con los valores extremos de los parámetros del transistor mostrados en la Tabla 4-4-1, para las temperaturas extremas T = 25°C y T = 60°C.

T	V_{ces}(V) mín	V_{ces}(V) máx	V_{be}(V) mín	V_{be}(V) máx	β_F	I_{CC}(nA)
25°C	.10	.3	.68	.75	30	5
60°C	.25	.45	.61	.67	42	160

I_c = 10 mA

Tabla 4-4-1.

Como β_F es el parámetro dominante para obtener el máximo valor de N de la condición de saturación, y su valor mínimo ocurre a 25°C, consideraremos los parámetros correspondientes a tal temperatura de operación.

En la tabla 4-4-2 se presentan los valores máximos de N, despejados de la ecuación 4-4-13* con i_B sustituído por $\bar{I}_{bsat} = \bar{I}_{csat}/\beta_F$, e $\bar{I}_{csat}$ dado por 4-4-11, utilizando para ello los valores siguientes de V_{CC} y V'_{CC}:

a) $V_{CC} = V'_{CC} = 12$ V b) $V_{CC} = V'_{CC} = 12.5$ V c) $V_{CC} = V'_{CC} = 11.5$ V

* La Ec. 4-4-13 no corresponde al peor de los casos, pero se presta más fácilmente a los cálculos que la Ec. 4-4-11.

	$\bar{I}_{csat}$ (mA)	$\bar{I}_{bsat}$ (mA)	$N_{máx}$	
a	9.85	0.328	3.79	(3)
b	10.27	0.342	4	(4)
c	9.43	0.314	3.57	(3)

Tabla 4-4-2.

Del análisis anterior se concluye por tanto que N no debe exceder de 3 (no. entero inmediato inferior del caso - c). Sustituyendo dicho valor en la Ec. 4-4-12, que refleja el caso peor posible, sin embargo, se obtiene $\underline{i_B} = 0.29$; luego no se satisface realmente la condición de saturación en condiciones extremas; con $N = 2$, en cambio, $\underline{i_B} = 0.36$ mA, asegurándose la correcta operación del inversor aún en tales condiciones.

Margen de ruido.- El voltaje mínimo permisible de entrada al inversor Q, $\underline{V_H}$, puede determinarse del circuito 4-4-4 (peor de los casos), con $i_B = \bar{I}_{bsat} = 0.314$ mA:

$$v_s = \underline{V_H} = V_{I\text{mín}} = \overline{R_k}\,\underline{I_1} = \overline{R_k}\left(\bar{I}_{bsat} + \frac{\bar{V}_{be} + \bar{V}_{EB}}{\underline{R_B}}\right) = 8.18\ V$$

Por otro lado, el voltaje de salida mínimo de Q' con $N = 2$, para el cual $\underline{i_B} = 0.36$ mA, resulta

$$V_{Clmín} = \bar{R}_k\left(\underline{i_B} + \frac{\bar{V}_{be} + \bar{V}_{BB}}{\underline{R_B}}\right) + V_{be} = 8.87\ V \qquad 4\text{-}4\text{-}14$$

luego se tiene un margen de ruido en estado alto $NM_1 = 0.69$ V.

Para determinar el margen de ruido en estado bajo, - es preciso determinar las condiciones de corte de Q en el - peor de los casos; esto ocurre cuando la salida de Q' en estado bajo es máxima, $V_{OLmáx} = \overline{V_{ces}}$, y cuando se propicia - un voltaje de base máximo, lo que ocurre cuando $R_B = \overline{R_B}$, $V_{BB} = \underline{V_{BB}}$ e $I_{CO} = \overline{I_{CO}}$. como se muestra en la Figura - 4-4-5.

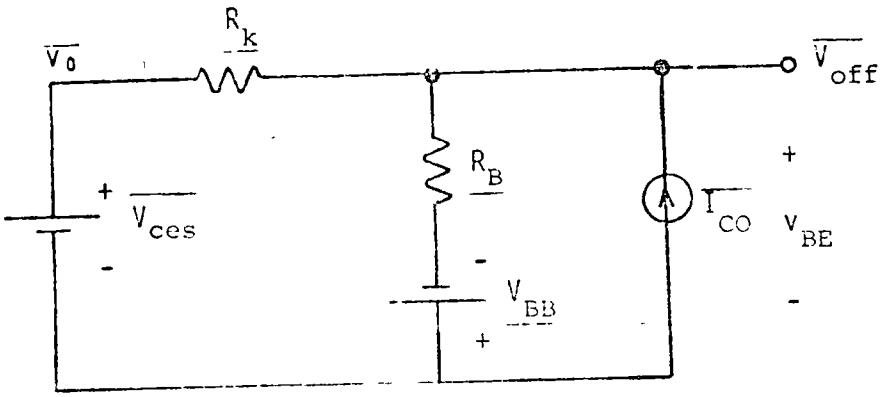


Figura 4-4-5.- Circuito que produce un voltaje menos negativo en la base de Q.

Estas condiciones ocurren, desde luego, a la temperatura máxima de operación. De la Fig., o bien de la Ec. - 4-4-3 se tiene

$$\overline{V}_{off} = (\underline{R_k} \,||\, \overline{R_B}) \left(\frac{\overline{V}_L}{\underline{R_k}} - \frac{V_{BB}}{\overline{R_B}} + \overline{I}_{CO}\right) \qquad 4\text{-}4\text{-}15$$

Para $\overline{V}_L = \overline{V}_{ces}$, $\overline{V}_{off} = -0.57$ V, asegurándose por tanto el corte de Q en el peor de los casos. Si el voltaje mínimo - de arranque $\underline{V_a}$ es igual a 0.3 V, y se utiliza dicho valor por V_{off} en 4-4-15, se obtiene el valór máximo permisible de entrada a Q en estado bajo

$$\overline{v}_s = V_{ILmáx} = \underline{R_k} \left(\frac{\underline{V_a}}{\underline{R_k} \,||\, \overline{R_B}} + \frac{V_{BB}}{\overline{R_B}} - \overline{I}_{CO}\right) \qquad 4\text{-}4\text{-}16$$

Obteniéndose el valor $I_{ILmáx} = 1.4$ V; el margen de ruido - en estado bajo resulta entonces $NM_0 = 1.4 - 0.45 = 0.95$ V.

Diseño de un inversor (Caso nominal).- Se ilustra - a continuación el diseño de un inversor con un transistor - NPN con las características de la tabla 4-4-1 para $I_c = 10$ mA. Se dispone de fuentes de 10 V, y se pretende operar al transistor con una corriente $I_{csat} = 10$ mA, un voltaje V_{off} de corte en la base de -0.5 V, y un abanico de salida - $N_{máx} = 3$. El diseño se llevará a cabo con válores nominales a la temperatura $T = 25°C$.

Procedimiento:

1.- Determinar R_C:

$$R_C = \frac{V_{cc} - V_{ces}}{I_{csat}} = 0.98 \text{ K} \quad (R_C = 1 \text{ K}\Omega)$$

2.- Especificar la corriente de base:

$i_B > I_{bsat} = I_{csat}/\beta_F = 0.33$ mA.

Sea $i_B = 0.5$ mA.

3.- Sustituir el valor de i_B en la Ec. 4-4-7.

$$I_1 \approx \frac{10 - 0.7}{3 + R_k} = \frac{0.7 + 10}{R_B} + 0.5 \qquad (a)$$

4.- Sustituir el valor de $V_{off} = -0.5$ V en la Ec. 4-4-3.

$$\frac{0.2 + -0.5}{R_k} = \frac{-0.5 + 10}{R_B} \qquad (b)$$

De (b), se tiene que $R_B = 13.57\ R_k$. Sustituyendo este valor en (a), se obtiene la ecuación algebraica de segundo grado

$$13.57\ R_k^2 - 190.3\ R_k + 64.2 = 0$$

satisfecha por los valores $R_K = 13.7\ K\Omega$ y $0.33\ K\Omega$. Consideramos el resultado mayor, para reducir el consumo de corriente y potencia de las baterías; los valores nominales más cercanos son 12 K y 15 K; con $R_k = 12\ K\Omega$ se asegura una corriente de base mayor; de (b) se obtiene entonces $R_B = 164$ K (sea 150 K, a fin de obtener mejores condiciones para el corte).

- Diseño en el peor de los casos.- Consideramos ahora las posibles variaciones de las fuentes, resistores y parámetros del transistor; un procedimiento de diseño consiste en determinar los límites permisibles de R_B ($\overline{R_B}$ y $\underline{R_B}$); como $\underline{i_B} \geq \overline{I_{bsat}}$, e $\underline{i_B}$ está dada por la Ec. 4-4-12, despejamos a $\underline{R_B}$ de la misma:

$$\underline{R_B} \geq \frac{\overline{V_{be}} + \overline{V_{BB}}}{\dfrac{(\underline{V_{CC}} - \underline{V_{CE}})\underline{R_C}}{R(\underline{R_A} + (N+1)\overline{R_C})} + \dfrac{\underline{V_{BB}} - \overline{V_{BE}}}{R} - \overline{I_{bsat}}} \qquad 4\text{-}4\text{-}17$$

siendo $R = \overline{R_A} + \overline{R_C}\ \|\ (\dfrac{R_k}{N-1})$. Con tolerancias de $\pm 10\%$ para las resistencias, utilizando un valor consistente de V_{CC} ($V_{CC} = \underline{V_{CC}} = 9.5$ V) para el cual $I_{bsat} = 0.341$ mA, se obtiene el valor límite inferior de R_B: $R_B \geq 66.5\ K\Omega$.

El límite superior de R_B ($\overline{R_B}$) puede determinarse de la Ec. 4-4-3 en condiciones extremas (Ec. 4-4-15), imponiendo un valor máximo a V_{off}:

$$\underline{R_B} \leq \frac{\overline{V_{off}} + \underline{V_{BB}}}{\frac{\overline{V_{ces}} - \overline{V_{off}}}{\underline{R_k}} + \overline{I_{CO}}} \qquad 4\text{-}4\text{-}18$$

$$T = T_{máx}$$

Si $\overline{V_{off}} = -0.3\ V$, se obtiene $\overline{R_B} \leq 132\ K\Omega$. Con un valor $R_B = 100\ K \pm 10\%$, el circuito operará satisfactoriamente incluso en el peor de los casos.

En general, no es posible satisfacer siempre las ecuaciones 4-4-17 y 4-4-18; en dichos casos es preciso rediseñar el circuito, empleando componentes con tolerancias más bajas, o bien transistores con ganancias mayores.

4.5.- Compuertas RTL (NO O).

Existen dos tipos básicos de compuertas RTL: Discretos e integrados. El circuito de la Fig. 4-5-1a. (compuerta discreta) se utilizó extensamente antes del desarrollo de los circuitos integrados, y se diferencia del inversor RTL

por el número de entradas, M, llamado el abanico de entrada (FI).

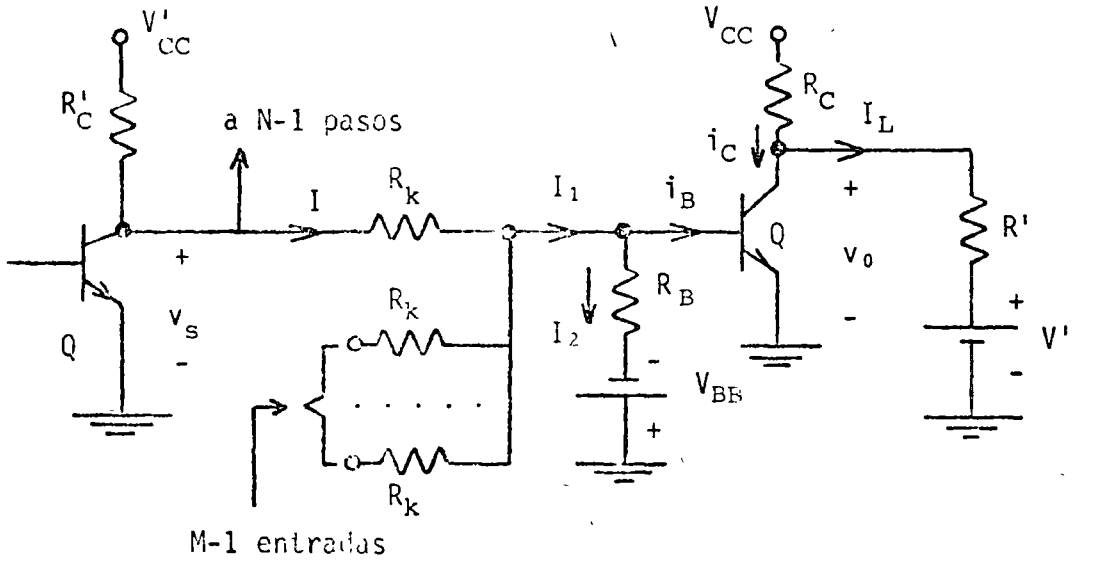


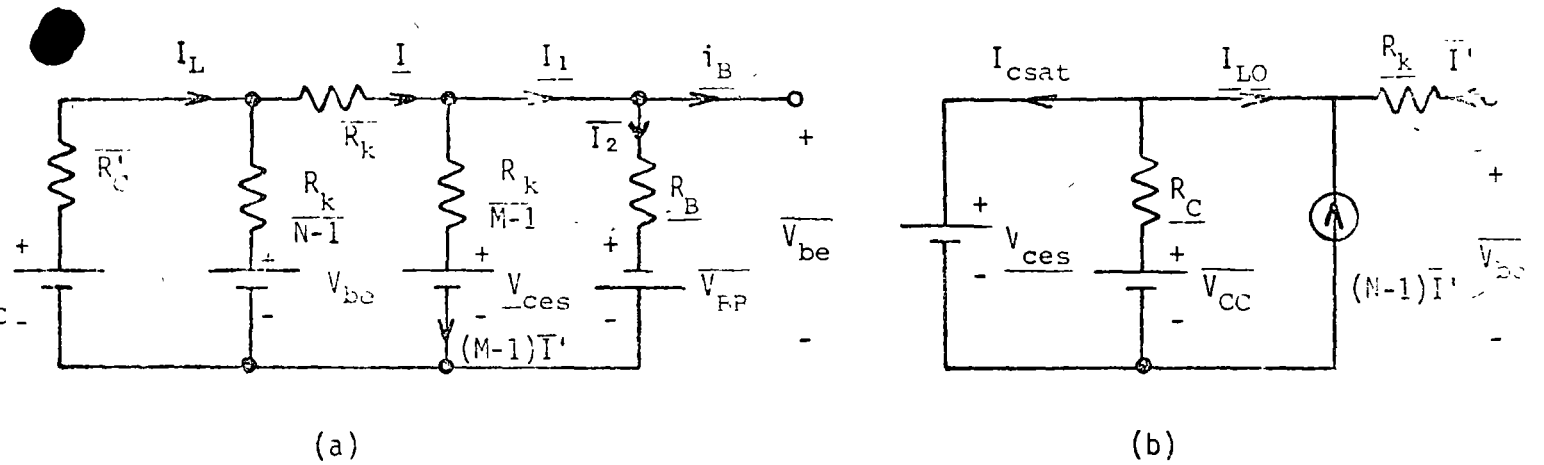
Figura 4-5-1.- Compuerta RTL discreta.

El transistor deberá estar cortado si todas las entradas son bajas, y deberá saturarse si al menos una de las entradas es alta, realizándose por tanto la operación NO O. Para propósitos de diseño se procede en forma similar al de un inversor, si bien es preciso considerar además los factores siguientes:

1.- Cuando sólo una de las entradas es alta, parte de la corriente I a través del resistor R_k correspondiente circula hacia los colectores de los M - 1 pasos saturados restantes; por consiguiente I_1, y por tanto i_B disminuyen al aumentar M; i_B resulta por tanto

$$i_B = I - I_2 - (M-1)I' = I - I_2 - (M-1)\frac{V_{be} - V_{ces}}{R_k}$$

4-5-1



(a) (b)

Figura 4-5-2.- Circuitos de entrada y salida de Q' (saturado) en casos extremos.

2.- La corriente de carga de un transistor saturado que excita a otro paso saturado (debido a alguna entrada alta a éste último) es negativa; esto es, circula hacia el colector aumentando la corriente de saturación (Fig. 4-5-2b). La corriente adicional es igual a NI', y se tiene

$$I_{csat} = \frac{V_{CC} - V_{ces}}{R_C} + NI' = \frac{V_{CC} - V_{ces}}{R_C} + N\frac{V_{be} - V_{ces}}{R_k} \qquad 4\text{-}5\text{-}2$$

El abanico de entrada limita entonces el valor del abanico de salida N, y por tanto el margen de ruido, pues produce una reducción de i_B y un aumento de I_{csat}. Otro efecto indeseable consiste en que el tiempo de propagación t_p aumenta notablemente al saturarse en exceso el transistor, si todas las entradas son altas. Existe, desde luego, la posibilidad de conectar capacitores de conmutación en paralelo con cada resistor de entrada R_k para reducir t_p; la compuerta se denomina en dicho caso RCTL (C por capacitores).

3.- Se menciona, por último, que el abanico de entrada influye también sobre el voltaje V_{off} de corte del transistor.

El circuito equivalente con todas las entradas bajas se muestra en la Fig. 4-3-3 (en el peor de los casos), con una resistencia R_k/M equivalente a la de M resistores en paralelo. Con valores nominales, se tiene

$$V_{off} = \left(\frac{R_k}{M} \,||\, R_B \right) \left(M \frac{V_{ces}}{R_k} - \frac{V_{BB}}{R_B} + I_{CO} \right) \qquad 4\text{-}5\text{-}3$$

que resulta evidentemente menos negativo conforme aumenta M.

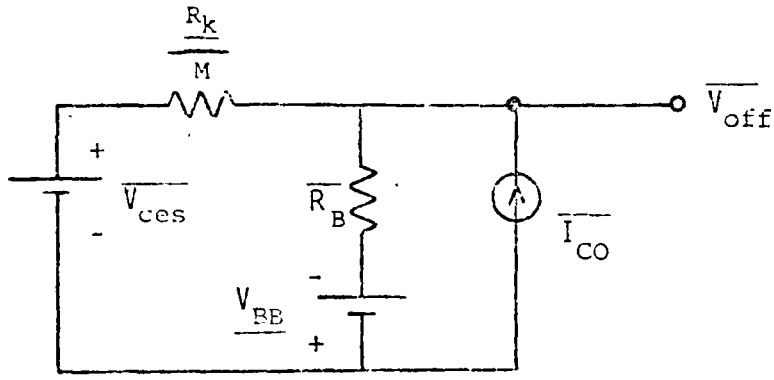


Figura 4-5-3.

Compuerta integrada.- En su versión integrada, cada entrada excita a su propio transistor, eliminándose por tanto la excesiva saturación de un solo transistor con todas las entradas altas.

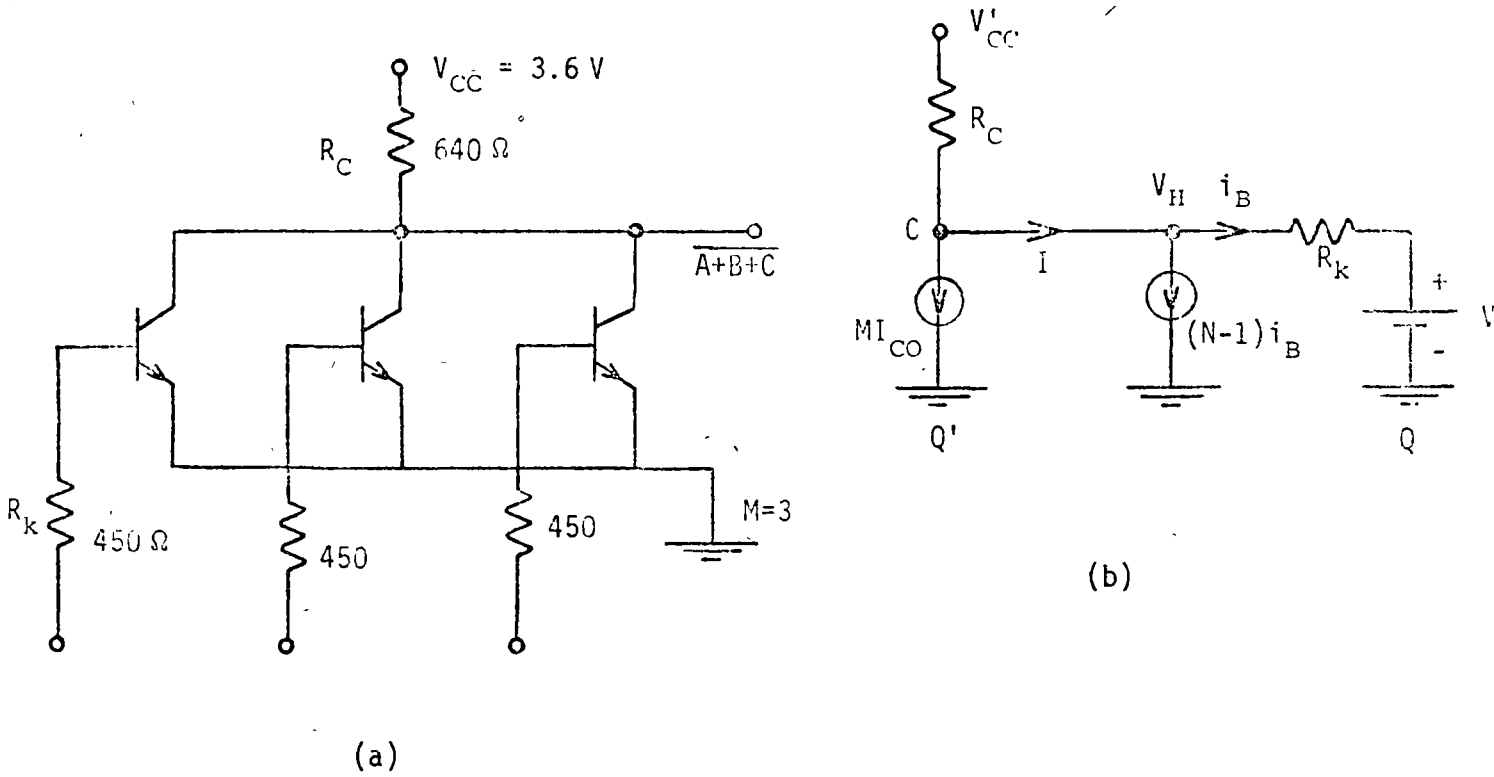


Figura 4-5-4.- Compuerta RTL integrada y circuito equivalente de entrada a un inversor saturado.

Con todas las entradas a un nivel bajo, el voltaje de salida V_H depende del abanico de salida N; del circuito equivalente 4-5-4b., despreciando la corriente de fuga I_{CO} se tiene

$$V_H = V_{be} + \frac{V_{CC} - V_{be}}{R_k + NR_C} R_k \qquad 4\text{-}5\text{-}4$$

Si $V_{be} = 0.75\,V$, $V_H = 1.92\,V$ si $N = 1$ mientras que $V_H = 1.29\,V$ si $N = 3$. En el peor de los casos V_H puede ser insuficiente para producir la saturación de alguno de los pasos de carga, por lo que es preciso limitar el valor de N. Del mismo circuito

$$i_B = \frac{V_H - V_{be}}{R_k} = \frac{V_{CC} - V_{be}}{R_k + NR_C} \qquad 4\text{-}5\text{-}5$$

y resulta igual a 1.2 mA si N = 3; esta corriente debe ser suficiente para saturar al transistor. Su corriente de colector es máxima si los otros transistores de la compuerta están cortados, en cuyo caso

$$I_{csat} = \frac{V_{CC} - V_{ces}}{R_C} \qquad 4\text{-}5\text{-}6$$

que resulta igual a 5.31 mA si $V_{ces} = 0.2\,V$. Es evidente que se requiere que β_F exceda de $5.31/1.2 = 4.4$ para saturar al transistor con la entrada alta.

Ejemplo 4-5-1.- Determinar el margen de ruido en estado alto si $\beta_F = 30$, $V_{D\gamma} = 0.76\,V$, $V_{Be} = 0.7\,V$,

$\underline{V_{ces}}$ = 0.2 V, V_{CC} = 4 ± 0.5 V, N = 4 y las tolerancias - de los resistores son de ± 20%.

Solución.- Determinamos primeramente $\overline{I_{csat}}$:

$$\overline{I_{csat}} \simeq \frac{\overline{V_{CC}} - \underline{V_{ces}}}{\underline{R_C}} \simeq 8.4 \text{ mA}$$

luego $\overline{I_{bsat}} = \overline{I_{csat}}/\underline{\beta_F} \simeq 0.28$ mA. El voltaje mínimo alto de entrada al transistor resulta, por tanto

$$V_{IHmín} = \overline{R_k}\ \overline{I_{bsat}} + \overline{V_{be}} \simeq 0.911 \text{ V}$$

Se requiere determinar a continuación la corriente - de base mínima en el peor de los casos, cuando V_{CC} = 3.5 V, R_C = 768 Ω, y los N - 1 pasos restantes del abanico de - salida toman la mayor parte de la corriente I*; esto es

$$\underline{i_B} = I - (N-1)\ \frac{V_H - \underline{V_{be}}}{\underline{R_k}} = \frac{V_H - \overline{V_{be}}}{\overline{R_k}}$$

---

* Este efecto se denomina "estrangulamiento". Ocurre cuando no están bien parecadas las características de entrada de los transistores, y uno de ellos recibe menor corriente de base que V_{BB} / R_B igual para todos.

De la ecuación anterior, y considerando que - $I = \frac{V_{CC} - V_H}{R_C}$, se obtiene

$$\underline{i_B} \simeq \frac{\frac{(\underline{V_{cc}} - \underline{V_{be}})\,\underline{R_k}}{\overline{R_k} + (N-1)\,\overline{R_C}} + \underline{V_{be}} - \overline{V_{be}}}{\overline{R_k} + \frac{\overline{R_c}\,\underline{R_k}}{(N-1)\overline{R_c} + \underline{R_k}}} \simeq 0.495 \text{ mA}$$

luego el voltaje de salida mínimo permisible del transistor excitador resulta

$$V_{OHmín} = \overline{R_k}\,\underline{i_B} + \overline{V_{be}} = 1.027 \text{ V}$$

y $NM_1 = 0.116$ V, margen relativamente reducido; obsérvese, sin embargo, que se han utilizado valores extremos de V_{CC} en los cálculos; por otra parte, es posible aumentarlo utilizando valores más elevados de V_{CC} (5 ó 6 voits).

Aunque no se utilizan en la actualidad con mucha frecuencia, los circuitos integrados RTL son baratos, fáciles de fabricar, y por tanto son de utilidad en aplicaciones comerciales de baja velocidad.

4.6.- Compuertas DTL (NO Y).

En su versión discreta, mostrada en la Fig. 4-6-1, una compuerta DTL incluye una compuerta pasiva Y con diodos conectada a un inversor, obteniéndose por tanto la función NO Y de las variables lógicas de entrada.

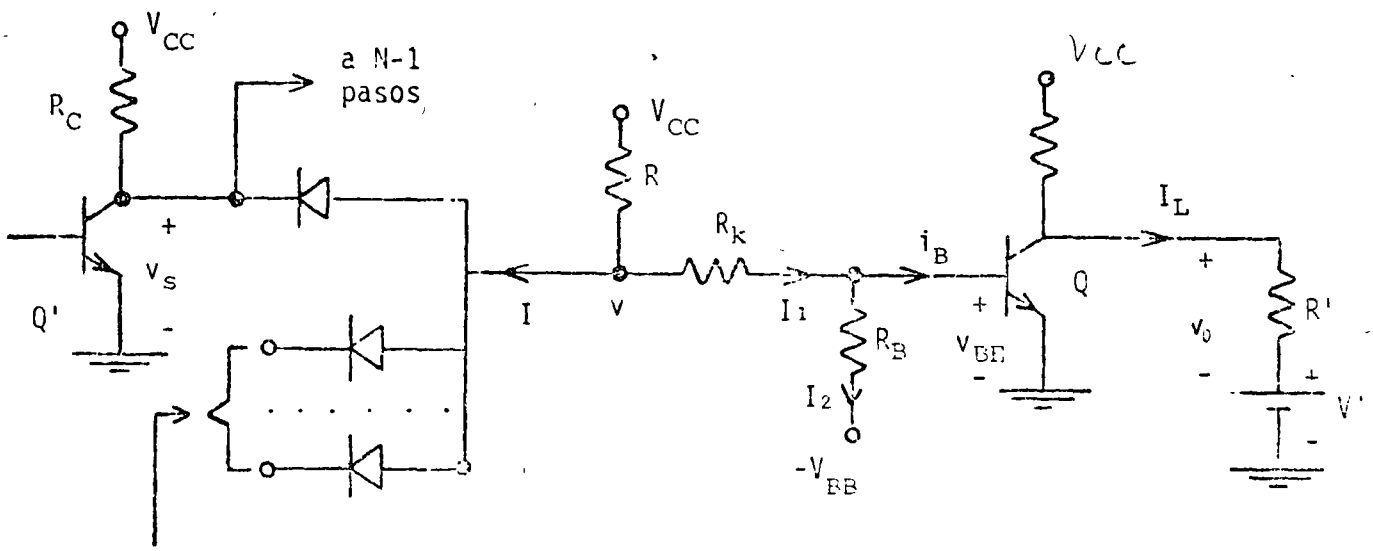


Figura 4-6-1.- Compuerta DTL discreta.

El voltaje v de entrada al inversor es bajo si cualquiera de las entradas es baja, cortando por tanto el inver-

sor; si todas las entradas son altas, en vambio, el voltaje v es alto, y el transistor Q deberá saturarse.

La corriente de carga de un transistor cortado hacia el abanico de salida es igual a NI_s, siendo I_s la corriente de saturación de los diodos, y es prácticamente despreciable. Un transistor saturado, en cambio, recibe una corriente NI_d, siendo I_d la corriente en directa de los diodos del abanico de salida. Esto es, I_L es negativa, y la corriente de colector máxima resulta por tanto

$$\overline{I_{csat}} = \frac{\overline{V_{CC}} - V_{ces}}{R_C} + N\overline{I_d} \qquad 4\text{-}6\text{-}1$$

La corriente de un diodo es máxima cuando el resto de los diodos del abanico de entrada se encuentran cortados, como se muestra en la Fig. 4-6-2, de donde obtenemos

$$\overline{I_d} = \frac{\overline{V_{th}} - V_d - V_{ces}}{R_{th}} + (M - 1)\overline{I_s} \qquad 4\text{-}6\text{-}2$$

siendo

$$\overline{V_{th}} = \overline{V_{CC}} - \frac{R}{R + \overline{R_K} + \overline{R_B}} (\overline{V_{CC}} + V_{BB} + \overline{R_B}\overline{I_{CO}}) \qquad 4\text{-}6\text{-}3$$

$$R_{th} = R \,||\, (\overline{R_c} + R_B) \qquad 4\text{-}6\text{-}4$$

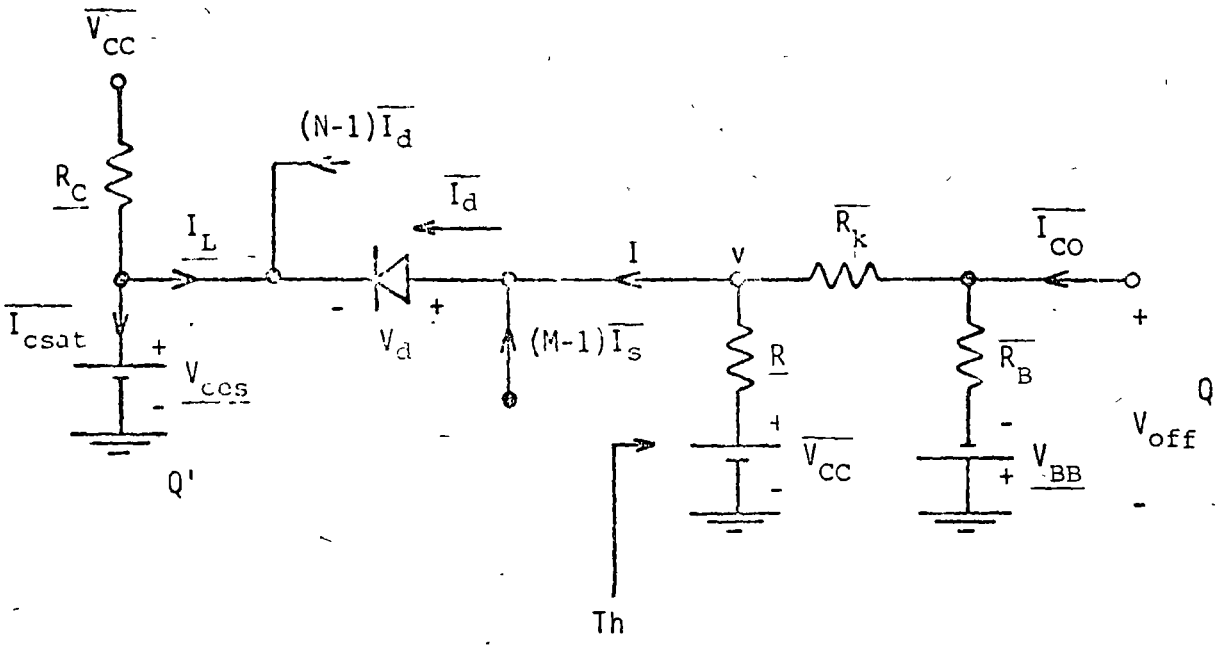


Figura 4-6-2.- Circuito equivalente para determinar la corriente - de colector máxima en el peor de los casos.

Para saturar a un transistor, por otra parte, se requiere que todos los diodos del abanico de entrada se encuentren cortados. Luego $I = -MI_s$. La corriente de base - mínima puede determinarse de la Fig. 4-6-3, y resulta

$$\underline{i_B} = \frac{\underline{V_{CC}} + RMI_s - \overline{V_{be}}}{\overline{R} + \overline{R_K}} - \frac{\overline{V_{be}} + \overline{V_{BB}}}{\underline{R_B}} \qquad 4\text{-}6\text{-}5$$

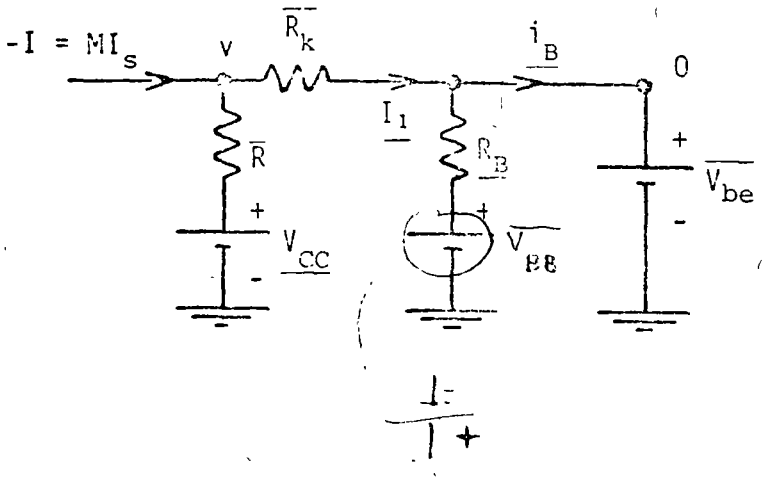


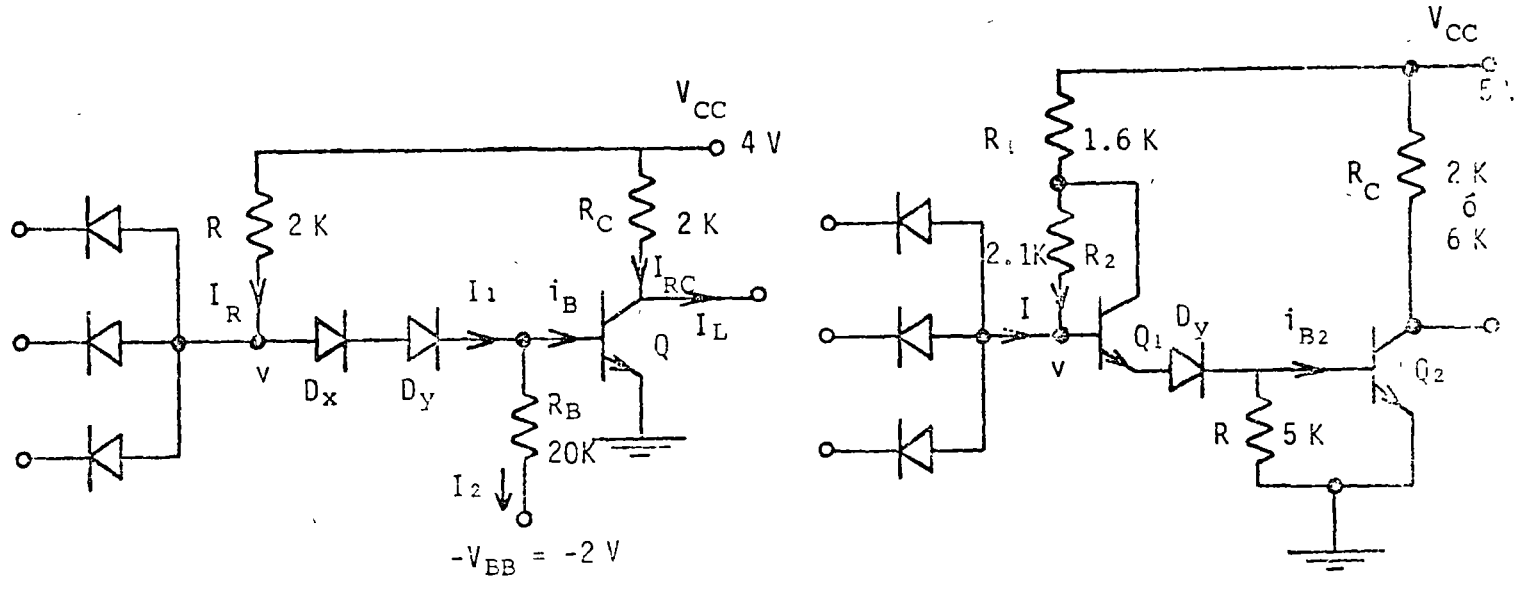
Figura 4-6-3.- Circuito equivalente de entrada al transistor saturado Q.

Resta sólo por considerar la condición de corte de Q; el valor máximo (menos negativo) de V_{off} puede determinarse de la Fig. 4-6-2, utilizando el valor mínimo de R_k y los valores máximos de V_{ces}, V_d e I_{CO} (a la máxima temperatura de operación); dado que $v = \overline{V_d} + \overline{V_{ces}}$, se tiene

$$V_{off} = \frac{\overline{V_d} + \overline{V_{ces}} + \overline{R_B}\ \overline{I_{CO}} - V_{BB}}{R_k + R_B}\ \overline{R_B} - V_{BB} + \overline{R_B}\ \overline{I_{CO}} \qquad 4\text{-}6\text{-}6$$

Las ecuaciones anteriores se utilizan para el diseño del inversor, considerando la condición de saturación - $i_B > \overline{I_{b,sat}}$ e imponiendo un valor limite superior al voltaje V_{off} de la base para el corte. Nótese que el abanico de entrada no influye negativamente sobre la condición de - saturación de una manera tan decisiva como en el caso de la compuerta discreta RTL; en su versión discreta, sin embargo, la compuerta DTL es más costosa, por el precio de los - diodos.

Compuertas DTL integradas.



(a) básica

(b) modificada -

Figura 4-6-4.- Compuertas integradas DTL.

En la compuerta básica se utilizan dos diodos, D_x y D_y, en vez de la resistencia R_k; si cualquier entrada es baja ($v_s = V_{cc}$), el voltaje $v = V_d + V_{cc}$ es del orden de 1 V; dado que se requiere un voltaje $v = V_{Dxa} + V_{Dya} + V_a \simeq 1.6$ V para arrancar al transistor, éste se encuentra cortado, con un voltaje de salida alto

$$v_0 = V_H = V_{CC} - R_C (I_{CO} + NI_s) \simeq V_{CC} \qquad 4\text{-}6\text{-}7$$

Con todas las entradas altas, $v = V_{Dx} + V_{Dy} + V_{be} \simeq 2.1$ V, los diodos de entrada se encuentran cortados, y Q se encuentra saturado con una corriente de base

$$i_B = I_R - MI_s - I_2 \simeq \frac{V_{CC} - v}{R} - \frac{V_{be} + V_{BB}}{R_B} \qquad 4\text{-}6\text{-}8$$

Con los valores de la Fig. 4-6-4a., $i_B \simeq 0.82$ mA si $v = 2.1$ V. Si $\beta = 20$, I_{csat} no debe exceder por tanto de 16.4 mA: la corriente de colector, por otro lado, aumenta con el abanico de salida

$$I_{csat} = I_{RC} - I_L = \frac{V_{CC} - V_{ces}}{R_C} + NI_d \qquad 4\text{-}6\text{-}9$$

La corriente I_d de un diodo es máxima si el resto de los diodos del abanico de entrada están cortados; esto es

$$I_d = (M - 1)I_s + I_R - I_1 \qquad 4\text{-}6\text{-}10$$

siendo $I_R = \dfrac{V_{cc} - v}{R}$, y $v \simeq 1$ V; la corriente I_1 circula por los diodos D_x y D_y de cada compuerta del abanico de salida hacia la fuente negativa; despreciando la corriente de fuga de la base, se tiene

$$I_1 \simeq \frac{v - V_{Dx} - V_{Dy} + V_{BB}}{R_B} \simeq .08 \text{ mA}.$$

Despreciando I_s, $I_d \simeq 1.42$ mA, e $I_{csat} = 1.85 + 1.42\ N$; el valor máximo permisible del abanico de salida resulta entonces

$$N \leq \frac{16.4 - 1.85}{1.42} = 10.2$$

o sea, $N \leq 10$.

El margen de ruido en estado bajo, NM_0, depende primordialmente de las características de los diodos y de V_{ces}, y varía, como NM_1, con la temperatura y el abanico de salida, dado que V_{ces} aumenta al aumentar I_{csat}. A fin de obtener en forma más precisa el valor máximo de V_{ces}, consideraremos la resistencia óhmica r_{cs} del colector (Fig. 4-6-5); llamando V_{ceso} el voltaje de saturación cuando $N = 0$, $V_{ces}(N) = V_{ceso} + r_{cs} N I_d$; en el peor de los casos, este valor corresponde a $V_{OLmáx}$.

Por otro lado, si los voltajes de arranque de los diodos a la máxima temperatura de operación son de 0.5 V, se tiene entonces

$$V_{ILmáx} = - V_{Da} + v = -V_{Da} + V_{Dxa} + V_{Dya} + V_a \simeq 1\ V$$

y se tiene entonces $NM_0 = 1 - V_{OLmáx}$; si $r_{cs} = 20\ \Omega$ y $V_{ceso} = 0.2\ V$, y se especifica un margen de ruido mínimo de 600 mV, $V_{OLmáx} = 0.4\ V$, y el valor máximo permisible de N resulta

$$N = \frac{V_{OLmáx} - V_{ceso}}{r_{cs}\ I_d} \simeq \frac{0.2 \times 10^3}{20 \times 1.42} \simeq 7$$

Nótese que los diodos D_x y D_y contribuyen a mejorar el nivel de ruido, al aumentar el voltaje de entrada máximo $V_{ILmáx}$.

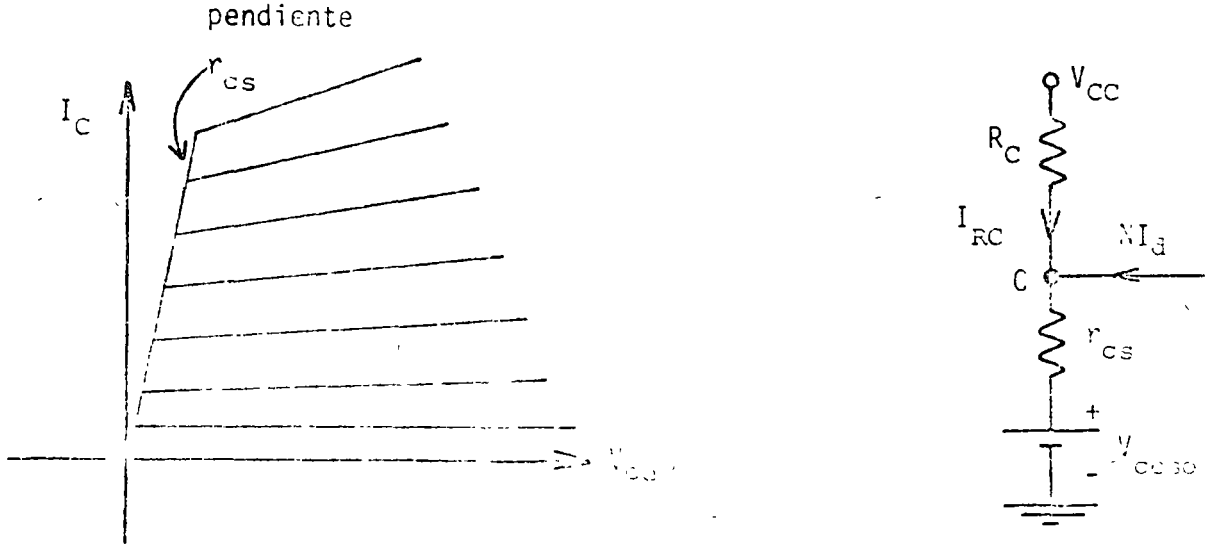
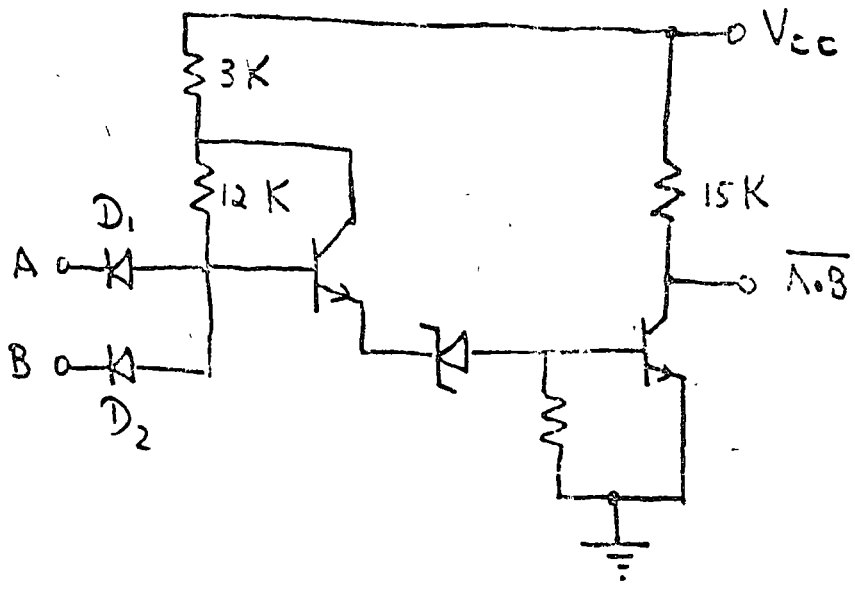


Figura 4-6-5.- Efecto de la resistencia óhmica del colector en saturación.

En la compuerta modificada (Fig. 4-6-4b.) se ha eliminado la batería negativa del transistor de salida; además, el diodo D_x en serie se ha sustituido por el transistor Q_1 (seguidor de emisor). El voltaje v consiste aún de 3 voltajes de unión en conducción: $v = V_{be1} + V_{Dy} + V_{be2} \simeq 2.1\,V$, y $v \simeq 1.6\,V$ para iniciar el arranque como en la compuerta básica, pero se dispone ahora de una corriente de base i_{B2} para el transistor Q_2 mucho mayor, debido a la amplificación proporcionada por Q_1; así, para el mismo valor de β_F, se obtiene mayor capacidad para el abanico de salida; éste último estará restringido solamente por el margen de ruido en estado bajo, NM_0.

Lógica HTL.- Existen diversas aplicaciones para las cuales se requiere un margen de ruido apreciablemente mayor que 600 mV; éste puede incrementarse utilizando valores mayores de V_{CC}, y un diodo Zener (con $V_z = 6\,V$) por D_y, como se muestra en la Fig. 4-6-6. A fin de limitar el consumo de potencia, se utilizan resistencias mayores que las de las compuertas DTL, con el aumento consiguiente del tiempo de propagación t_p.

Estas compuertas se utilizan primariamente en circuitos de control de motores, sistemas de comunicación acopladas a relevadores, y otros sistemas ruidosos de baja velocidad requeridos en la industria.



Compuerta HTL.

Figura 4-6-6.

4.7.- Compuertas TTL.

La familia lógica TTL (o T^2L) es una de las más utilizadas en la actualidad. Su principio de operación es similar al de las compuertas DTL, pero dan lugar a tiempos de propagación sustancialmente menores a costa de consumos mo-

derados de potencia. En vez de utilizar diodos convencionales en el circuito de entrada como en las compuertas DTL, se utilizan las uniones emisor-base de un transistor multiemisor NPN con un área común para la base, como se muestra en la Fig. 4-7-1b. La construcción del circuito de entrada de una compuerta DTL se aprecia a guisa de comparación en la Fig. 4-7-1a.; la región de la base es también común, pero está conectada eléctricamente al colector n. La capacitancia C_{cs} en ambos casos se debe a la juntura entre el colector n y el material básico p (sustrato) donde se efectúan las difusiones.

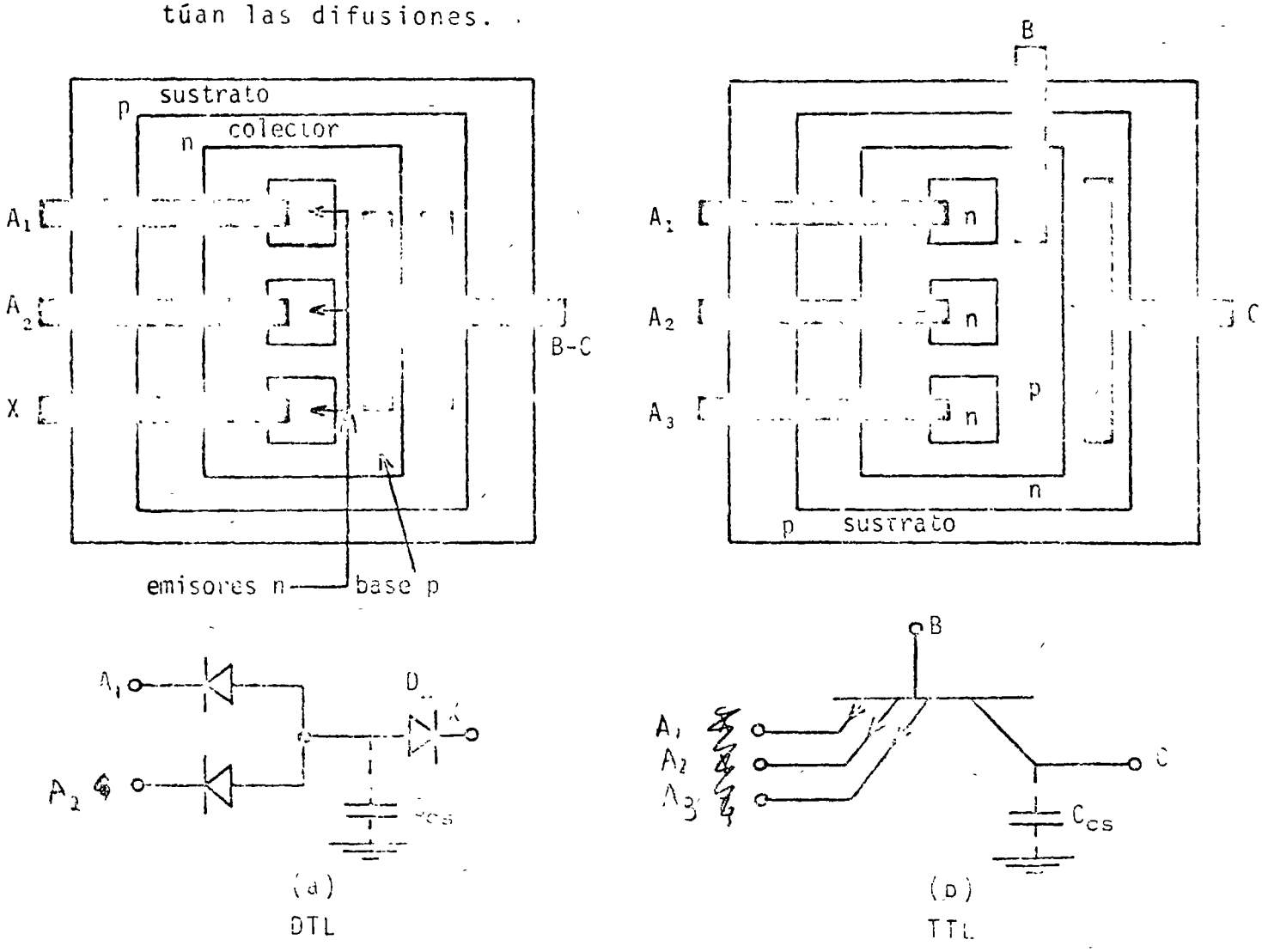


Figura 4-7-1.- Construcción de los circuitos de entrada de compuertas DTL y TTL con un área común para la base.

La compuerta TTL NO Y estándar (serie 54-74) y su característica de transferencia típica se muestran en la Fig. 4-7-2; la compuerta consta del paso de entrada (transistor-multiemisor), un paso divisor de fase y un paso de salida, conocido como poste totémico.

Análisis con las entradas altas.- Con las entradas altas (o desconectadas) existe una corriente I_{b2} a través de la unión PN (base-colector) de Q_1 suficiente para saturar a Q_2; al conducir Q_2, permite a su vez la saturación de Q_4 (Fig. 4-7-2).

El voltaje v_{B1} de la base de Q_1 es aproximadamente igual a 2.1 V, luego $I_{b2} \simeq 0.725$ mA, despreciando las corrientes en reversa por los emisores. Como $v_{E2} = v_{BE4} \simeq 0.7$ V, $v_{C2} \simeq 1$ V, voltaje insuficiente para arrancar a Q_3 y D_1; el voltaje v_0 de salida (colector de Q_4) es igual por tanto a $V_{ces4} \simeq 0.3$ V, circulando hacia el colector una corriente negativa de carga proporcional al abanico de salida.

Para completar el análisis, se observa que $I_{c2} \simeq I_{R2} = \frac{V_{CC} - v_{C2}}{R_{C2}} \simeq 2.5$ mA, luego $I_{e2} \simeq 3.225$ mA; $I_{RE2} = v_{E2}/R_{E2} \simeq 0.7$ mA, luego $I_{b4} = I_{e2} - I_{RE2} \simeq 2.525$ mA.

Es de hacerse notar el alto valor de la corriente de base I_{b4}, que propicia un abanico de salida grande, y una rápida descarga de una carga capacitiva.

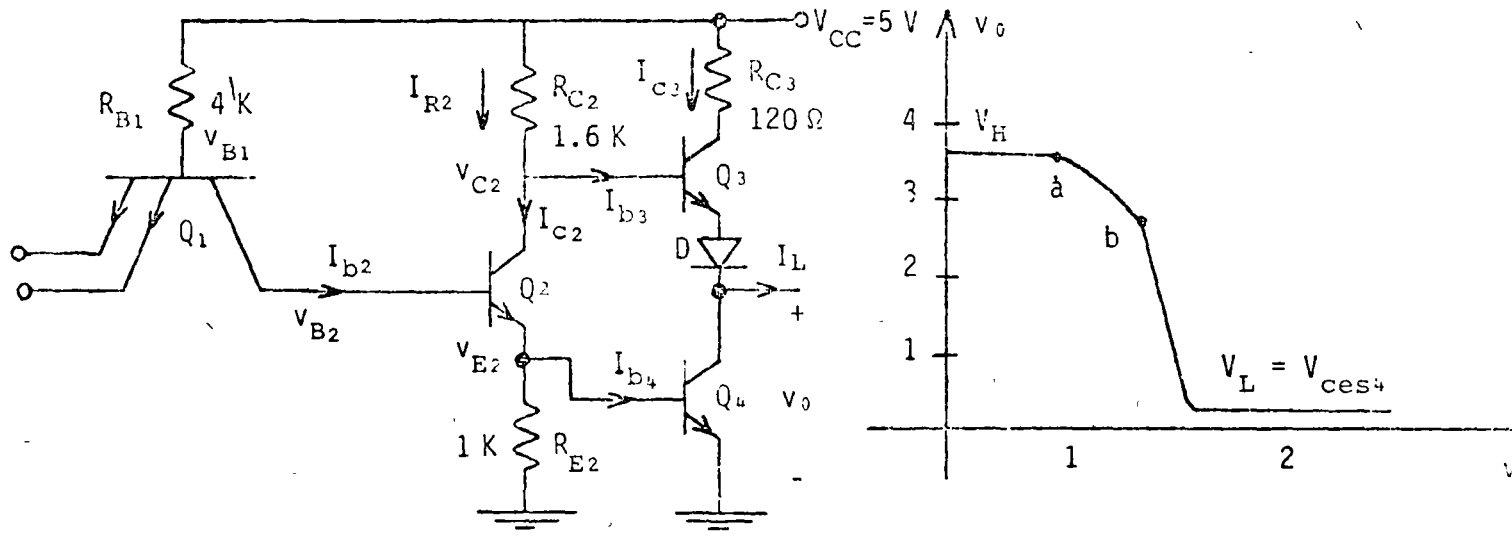


Figura 4-7-2.- Compuerta estándar (54-74) y característica de transferencia.

Análisis con una entrada alta y una baja.- Basta con que una sola entrada sea baja (V_E -) para que por el emisor correspondiente circule corriente hacia el paso excitador, y Q_1 se sature.

Ahora $v_{B1} = V_{ces} + V_{be1} \simeq 1\,V$, y $v_{B2} = v_{C1} \simeq 0.6\,V$; este voltaje es insuficiente para arrancar tanto a Q_2 como a Q_4. Con Q_2 cortado, I_{b1} es negativa, contribuyendo a extraer con rapidez las cargas almacenadas en la base de Q_2 si se encontraba previamente prendido, fenómeno que contribuye notablemente a la reducción del tiempo de desaturación de Q_2.

Con Q_2 y Q_4 cortados, $I_{R2} \simeq I_{b3}$, y la corriente del emisor de Q_3, que circula también por el diodo D, es aproximadamente igual a la corriente I_L de carga; Q_3 actúa por tanto como un seguidor de emisor, permitiendo una carga rápida de V_{ces} a un voltaje de salida $v_0 = V_H$ alto.

El voltaje alto (V_H) de salida depende de I_L; se tiene que $I_L \simeq I_{e3} = (\beta_{F3}+1)I_{b3}$, y además $v_{C2} = V_{be3}+V_d+V_H \simeq 1.4+V_H$; como $I_{b3} \simeq I_{R2} = (V_{CC} - v_{C2})/R_{C2}$, entonces

$$V_H \simeq V_{CC} - R_{C2}\,(1 - \alpha_{F3})I_L - 1.4 \qquad 4\text{-}7\text{-}1$$

Dado que I_L es pequeña, $V_H \simeq 3.6\,V$, valor típico del voltaje de salida de una compuerta TTL en estado alto.

A fin de determinar la corriente I_L de carga tanto en estado bajo como en estado alto, es preciso analizar con detalle el funcionamiento del transistor multiemisor, Q_1: éste se muestra nuevamente en la Fig. 4-7-3, con un voltaje v_s aplicado a una entrada con las demas entradas altas.

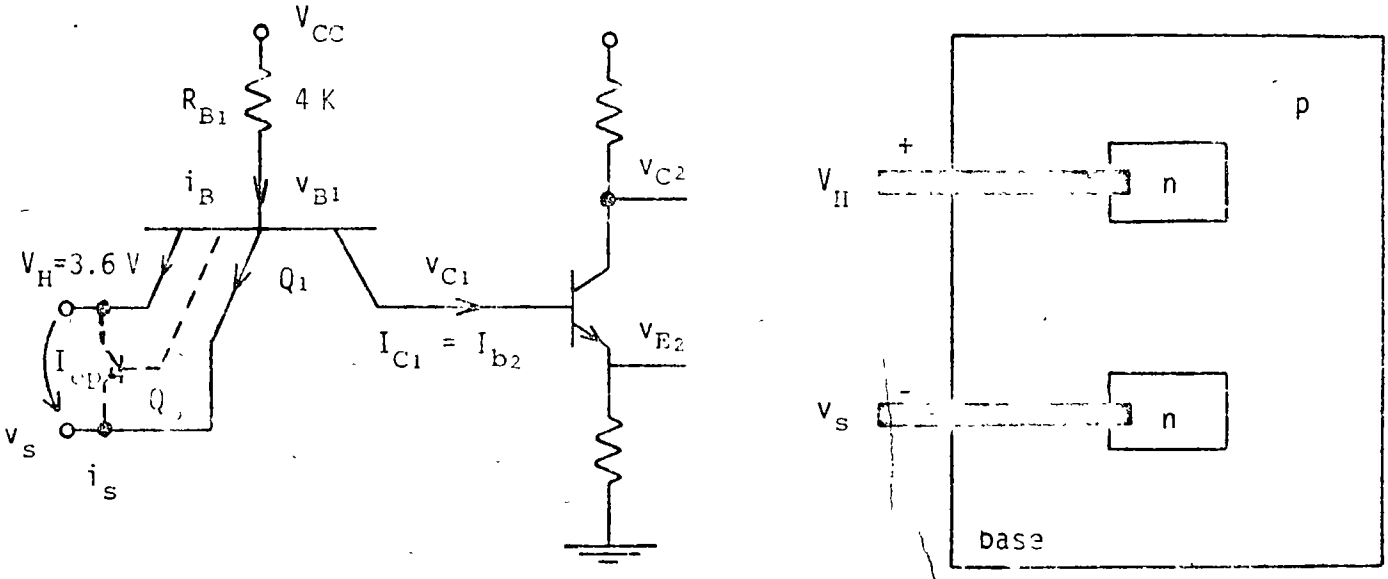


Figura 4-7-3.- Paso de entrada y transistor parásito npn.

Para v_s bajo ($v_s = V_L$), i_s es negativa ($-I_{so}$), comportándose la unión base emisor correspondiente como un diodo, con una corriente

$$i_B = \frac{V_{CC} - V_{B1}}{R_{B1}} \sim \frac{V_{CC} - (v_s + 0.7)}{R_{B1}} \qquad 4\text{-}7\text{-}2$$

Así, para $v_s = 0.3$ V, $i_B \simeq 1$ mA. A esta corriente se añade una corriente I_{ep} debida al transistor parásito Q_p npn, formado por ambos emisores y la base; I_{ep} es igual a $\beta_{Fp}\, i_B$, siendo β_{Fp} el factor de amplificación de corriente de Q_p. Al fabricar la compuerta, se procura que β_{Fp} sea sumamente pequeña, del orden de .01; así, $I_{ep} = .01\, i_B = .01$ mA. Si se tienen M colectores de entrada, existen M-1 transistores parásitos, y la corriente I_{so} resulta .

$$-i_s = I_{so} = i_B(1 + (M-1)\beta_{Fp}) \simeq i_B(1 + .01(M-1)) \qquad 4\text{-}7\text{-}3$$

Conforme aumenta v_s, aumenta v_{B1} y disminuye consecuentemente la magnitud de i_s; cuando v_{B1} sea igual a la suma de los voltajes de arranque de 2 diodos (1 a 1.2 V), la unión base-colector de Q_1 se polariza en directa, y se inicia la conducción de Q_2 (punto a de la característica de salida); tanto el voltaje v_{C2} como v_o disminuyen gradualmente al crecer v_s, en tanto Q_4 no inicie su conducción. Cuando $v_{B1} \simeq 1.8$ V, Q_4 arranca (punto b de la característica) y poco después se satura.

La magnitud de i_s, entre tanto, disminuye al crecer v_{B1}, hasta establecerse plena conducción de Q_1, Q_2 y Q_4; v_{B1} queda prácticamente fijado a 2.1 V, y la unión base-emisor queda polarizada en reversa al crecer v_s. Los emisores se comportan ahora como colectores, con una corriente por cada uno de ellos igual a

$$i_s = I_{s1} = \frac{\beta_R}{M} i_B \qquad 4\text{-}7\text{-}4$$

En el proceso de fabricación se procura que el factor de amplificación en reversa β_R sea pequeño, lográndose incluso valores iguales a 0.1 o menores. Como $v_{B1} \simeq 2.1$ V, $i_{B1} \simeq 0.725$ mA, y el máximo valor de i_s si $\beta_R = 0.1$ resulta igual a 72.5 μA/M. Esta corriente es sustancialmente menor que la magnitud de la corriente negativa cuando v_s es bajo.

Abanico de salida.- El abanico de salida está limitado por el factor de amplificación β_{F4} del transistor Q_4. Por el colector de Q_4 circula la corriente $I_{csat4} = -I_{LO}$ que en el peor de los casos equivale a NI_{so}, luego

$$I_{LO} = -Ni_B(1 + \beta_{FP}(M-1)) \qquad 4\text{-}7\text{-}5$$

Si $M = 3$, $-I_{LO}$ resulta aproximadamente igual a $-1.02N$ mA, ya que $i_B = 1$ mA si $v_s = V_{CES} = 0.3$ V. Recordando que $I_{b4} \simeq 2.525$ mA, y si $\beta_{F4} = 10$, se obtiene el límite superior de N:

$$N = \frac{\beta_{F4} I_{b4}}{I_{csat}} \simeq 24$$

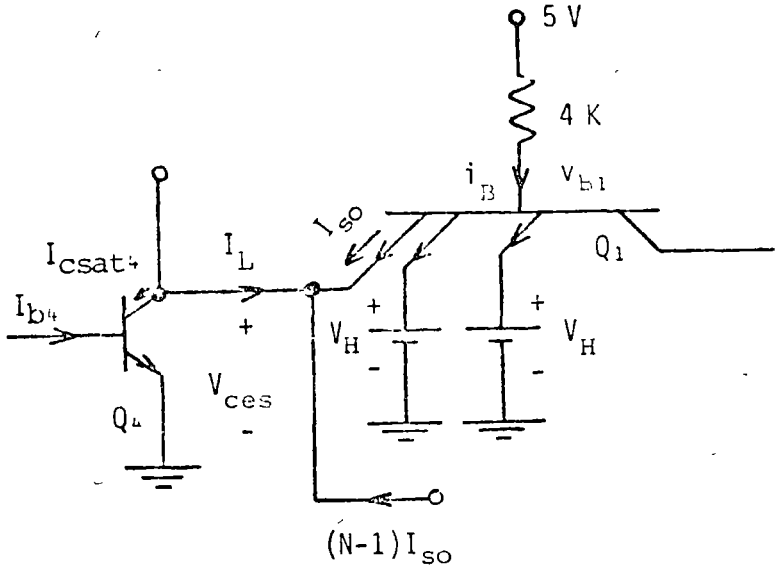


Figura 4-7-4.- Circuito para el cálculo del abanico de salida.

En la práctica no se recomienda un abanico de salida superior a 10, a fin de no reducir excesivamente el margen de ruido.

Ejemplo 4-7-1.- Determinar el abanico de salida para un margen de ruido $NM_0 = 0.7$ V si $\overline{r_{cs}} = 25\ \Omega$. $\overline{V_{ceso}} = 0.2$ V a una corriente $I_c = 1$ mA, $M = 3$, V_{be} = 0.66 V y $V_a = 0.5$ V para todas las uniones base-emisor.

Solución.- Determinemos primeramente el voltaje de entrada bajo máximo permisible, $V_{ILmáx}$, que corresponde al punto de la característica de salida, justo antes de que arranque Q_4:

$$V_{ILmáx} = -V_{be1} + V_{bc1} + V_{be2} + V_{a4} = 1.16\ V$$

Para determinar NM_0, requerimos ahora el voltaje bajo máximo de salida, $V_{OLmáx}$, que depende de N; este es el voltaje máximo de saturación, que aumenta con la corriente $I_{csat} = -Ni_S$:

$$V_{OLmáx} = V_{ceso} + (I_{csat} - .001)r_{cs4} = 0.175 + 25\ I_{csat}$$

Por otro lado, i_S depende de i_B (Ec. 4-7-3), e i_B está dada por 4-7-2; sustituyendo v_S por $V_{OLmáx}$ en ésta última, tendremos:

$$I_{csat} = 1.04\ Ni_B = 1.04\ N\ \frac{5 - (V_{OLmáx} + 0.66)}{4 \times 10^3}$$

Luego $I_{csat} = \dfrac{4.165\ N}{3846 + 25\ N}$, y $V_{OLmáx}$ resulta

$$V_{OLmáx} = 0.175 + \frac{104.125\ N}{3846 + 25\ N}$$

Dado que $NM_0 = V_{ILmáx} - V_{OLmáx} = 0.7$ V, - - $V_{OLmáx} = 0.46$ V, y se obtiene $N = 11.3$ ($N = 11$).

Especificaciones.- Es de harse notar que en el ejemplo anterior no se han considerado variaciones de la fuente V_{CC} y de las resistencias de la compuerta. Al utilizar - compuertas integradas, se consideran las especificaciones - garantizadas por el fabricante (Tabla 4-7-1).

$V_{IH} \geq$ 2 V	FO $\leq$ 10	$\|I_{LO}\| \leq$ 16 mA. a 0.4 V
$V_{IL} \leq$ 0.8 V	NM $\geq$ 400 mV	$I_{L1} \leq$ 400 µA a 2.4 V
$V_{OH} \geq$ 2.4 V	$t_p <$ 20 nseg.	$I_{s1} \leq$ 40 µA a 2.4 V
$V_{OL} \leq$ 0.4 V	$\|I_{os}\| \leq$ 55 mA.	$I_{so} \leq$ 1.6 mA. a 0.4 V

Tabla 4-7-1.- Algunas especificaciones garantizadas de las compuertas TTL de la serie 54/74.

La corriente I_{os} es la corriente en corto circuito con Q_3 prendido, limitada por la resistencia R_{c3}.

Respuesta transitoria.- Las compuertas TTL son las más rápidas de las compuertas saturantes. Por un lado el -

poste totemico de salida permite una rápida descarga de la carga capacitiva de salida hacia el transistor Q_4 saturado, y una carga rápida al voltaje alto debido a que el transistor Q_3 actúa como un seguidor de emisor.

Otro efecto de suma importancia es el corte rápido de Q_2 (y Q_4) al cambiar una entrada de 1 a 0, con el resto de las entradas altas. Justo antes de la transición, $v_{B1} = 2.1\,V$ y $v_{B2} = 1.4\,V$, con una carga almacenada por la capacitancia parásita C_{cs} y la capacitancia de entrada de Q_2, C_{12}.

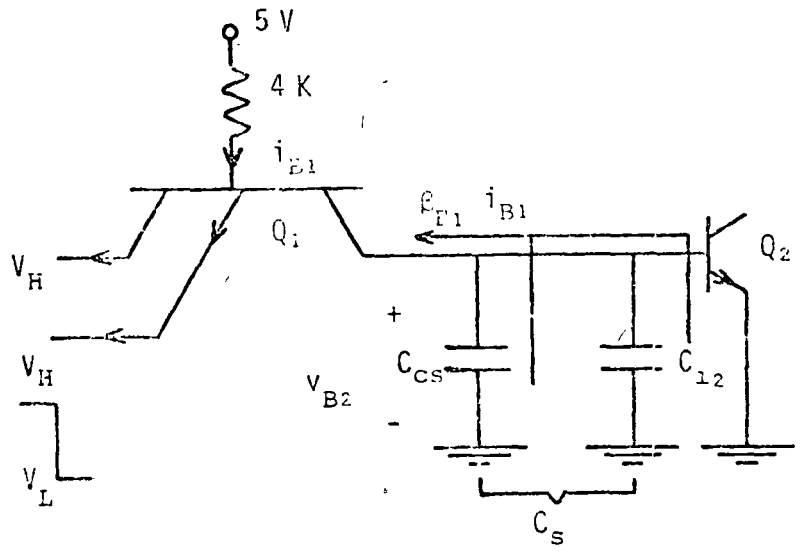


Figura 4-7-5.- Corte de Q_2.

Al efectuarse la transición, fluye una corriente $\beta_{F1}\, i_{B1}$ de colector, proporcionada justamente por la carga almacenada en la capacitancia total C_s hacia la unión colector-base de Q_1 en directa. Dicha corriente es fija, y dura el tiempo requerido para descargar a Q_2.

Sea $v_{B1}(0_+) = V_L + V_{BB1} \simeq 0.9$ V. Luego $i_{B1} = 1.075$ mA. y si $\beta_{F1} = 20$, $i_{C1} = -i_{B2} = 21.5$ mA. El voltaje de la base v_{B2} resulta por tanto

$$v_{B2}(t) = v_{B2}(0) - \frac{\beta_{F1}\, i_{B1}\, t}{C_s} = 1.4 - 0.43 \times 10^{-9}\, t$$

si $C_s = 50$ pF. El tiempo requerido para que v_{B2} sea igual a cero volts resulta entonces

$$t = \frac{v_{B2}(0)\, C_s}{\beta_{F1}\, i_{B1}} \simeq 3.26 \text{ nseg.}$$

Al cesar la descarga, $I_c = I_{CO}$, y la corriente del emisor bajo es suministrada prácticamente en su totalidad por la batería V_{CC}.

Pico de corriente.- Al cortar el transistor Q_2, aumenta el voltaje v_{C2} arrancando al transistor Q_3 y al diodo D. Simultáneamente, el voltaje v_{B2} tiende a decrecer, pero el transistor Q_1 requiere de unos cuantos nanosegundos para desaturarse, dependiendo de su carga almacenada y la resistencia R_{E2}. Ocurre así que por unos instantes tanto Q_3 como Q_4 se encuentran prendidos, propici-

una corriente por el emisor de Q_3 y el diodo D hacia el colector de Q_4 (a un voltaje V_{ces}).

La corriente suministrada por V_{CC} al colector de Q_3, I_{C3}, aumenta durante dicho periodo a un valor muy alto (hasta 40 mA), estando limitada solamente por la resistencia R_{C3}. Este pico de corriente puede producir ruido apreciable en otras compuertas, debido a la imposibilidad de reducir a cero la impedancia de salida de la fuente V_{CC}.

Variantes de la familia TTL.- La compuerta básica (54/74) ha sufrido diversas modificaciones para obtener fundamentalmente mayor velocidad (menor tiempo de propagación), o bien menor potencia de consumo.

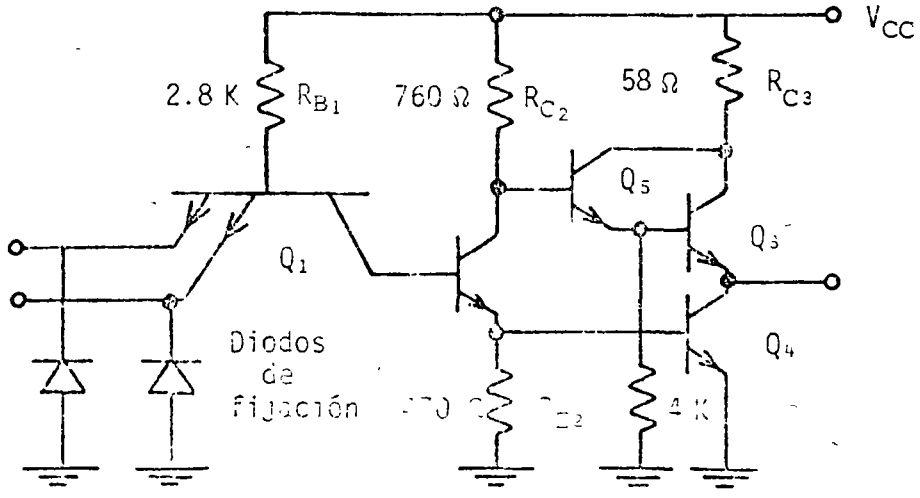


Figura 2-7-6 - Compuerta TTL de alta velocidad (54H/74H).

El circuito de la Fig. 4-7-6 es una compuerta de alta velocidad (serie 54H/74H), con tiempos de propagación de 12 ó 10 nseg. Los transistores Q_3 y Q_5 forman una configuración conocida como 'Darlington', caracterizado por un factor de amplificación β_F equivalente al producto de β_{F3} por β_{F5}. Este alto factor de amplificación contribuye a disminuír la impedancia de salida de Q_3, y por tanto a la reducción del tiempo de carga de la capacitancia del nodo de salida.

Por otro lado, los valores de las resistencias son menores que los de la compuerta estándar, a fin de reducir las constantes de tiempo y por tanto los tiempos de conmutación, a expensas de un mayor consumo de potencia.

Cada uno de los emisores de entrada está conectado al cátodo de un diodo que fija o limita la excursión negativa de los pulsos de entrada; dicha excursión es causada por oscilaciones o reflexiones en las pistas conductoras de los circuitos impresos, mismos que se comportan como líneas de transmisión con inductancia apreciable a frecuencias altas de operación. Al limitar el voltaje negativo de entrada de un emisor bajo a -0.7 V, se evita la ruptura de la unión en reversa de la base con una entrada no utilizada (conectada a V_{CC}).

Un circuito semejante al de la Fig. 4-7-6, pero con resistencias mucho mayores es el de la serie 54L/74L, utilizado en sistemas de baja potencia (1 mW por compuerta, en contraste con 25 mW consumidos por la compuerta 54H/74H).

TTL con diodos Schottky (54S/74S).- A fin de obtener tiempos de propagación aún menores, debe evitarse la satura

ción de los transistores. Como se estudió en el Capítulo 3, ello puede lograrse con diodos fijadores; la conmutación de un diodo fijador debe ser extremadamente rápida, lo cual puede lograrse con diodos de efecto Schottky de barrera (SBD), formados por una juntura metal-semiconductor. El comportamiento eléctrico de un SBD es similar al de un diodo convencional, pero se distingue de éste último por operar con cargas mayoritarias (sin almacenar cargas minoritarias) y por su menor voltaje V_d en directa, del orden de 0.3 V.

El SBD se fabrica entre la base y el colector de un transistor, como se muestra en la Fig. 4-7-7, impidiendo que V_{bc} sobrepase de 0.3 a 0.35 V, y evitándose por tanto la saturación plena del transistor.

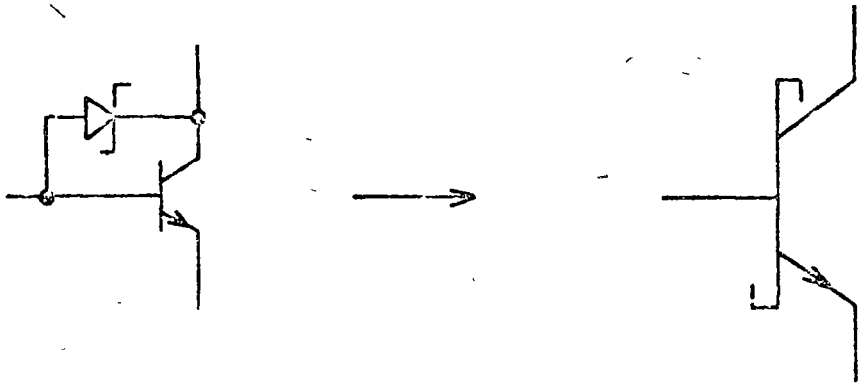
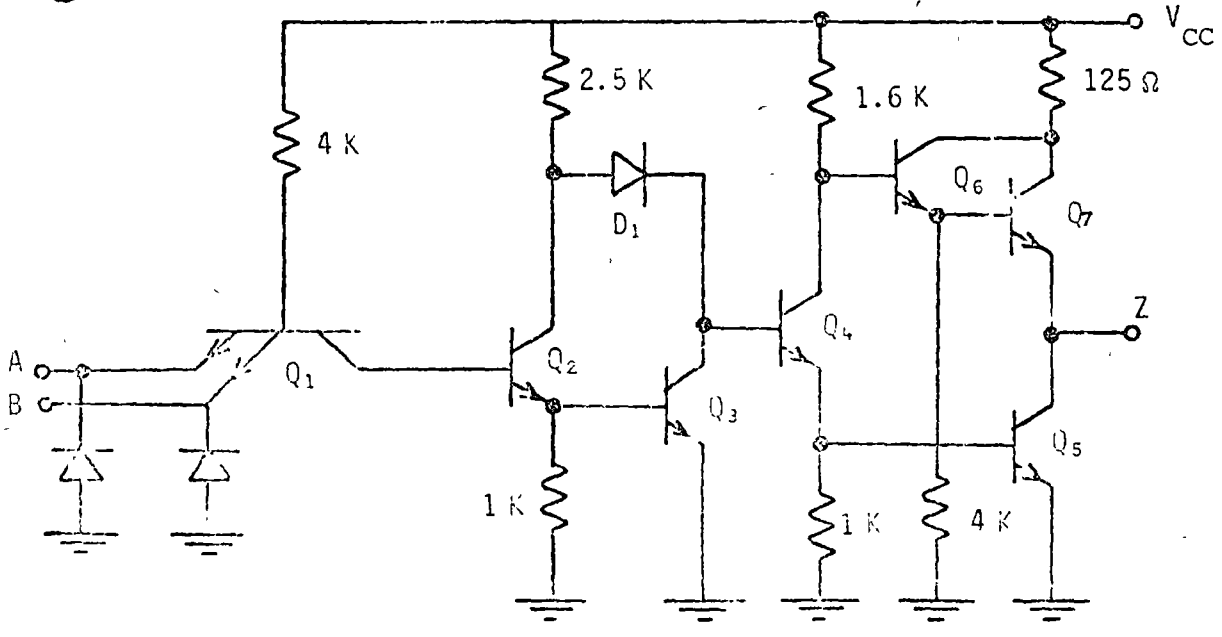


Figura 4-7-7.- Uso y símbolo de un diodo Schottky como fijador.

Compuertas TTL Y, O, NO O e Y-O-NO.- La extraordinaria aceptación de la familia lógica TTL ha propiciado el desarrollo de diversas funciones lógicas con la misma tecnología.

En la Fig. 4-7-9 se muestra una compuerta Y; si cualquiera de las entradas es baja (V_L), los transistores Q_2 y Q_3 están cortados, Q_4 se satura y la salida en Q_5 es baja. Cuando ambas entradas son altas, Q_2 y Q_3 conducen, Q_4 está cortado y activa a la configuración Darlington Q_6 y Q_7, produciendo una salida alta.



A	B	Z
0	0	0
0	1	0
1	0	0
1	1	1

Figura 4-7-9.- Compuerta TTL 'Y'.

Esta tecnología se emplea para las compuertas de la serie 54S/74S (Fig. 4-7-8), lográndose tiempos de propagación de 3 a 4 nseg. Una característica adicional es el empleo del transistor Q_6 actuando como carga del emisor de Q_2 en vez de un resistor R_{E2} conectado a tierra. Con ello, se impide la conducción de Q_2 antes de que su voltaje de base permita conducir asimismo a Q_4, lográndose una característica más abrupta a partir del valor $v_s = 1.3$ a 1.4 V. con el consiguiente aumento del margen de ruido.

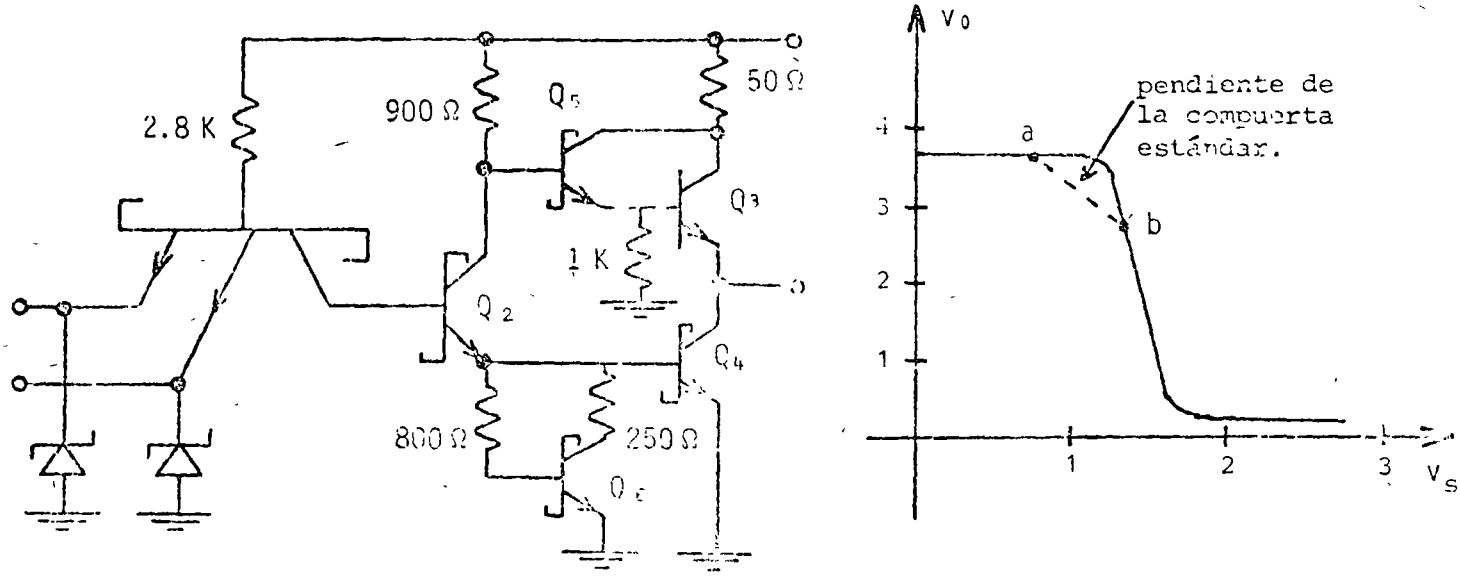
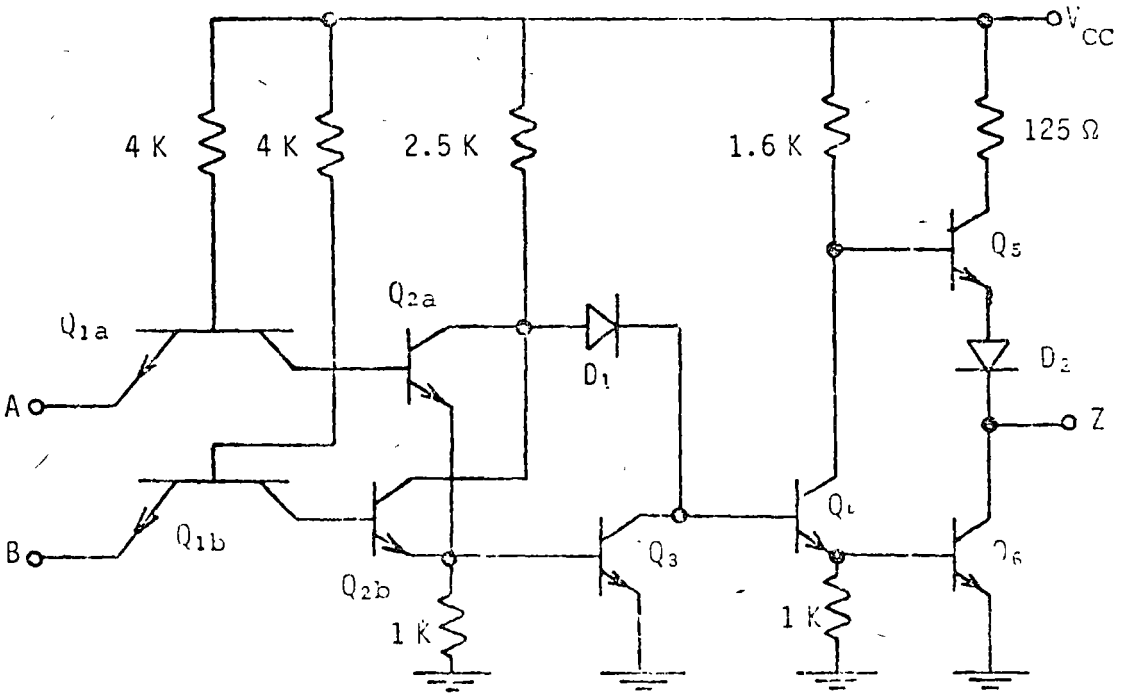


Figura 4-7-8.- Serie 54S/74S y características de transferencia

La función 'O' puede lograrse utilizando un transistor con un emisor por cada entrada lógica (Fig. 4-7-10). - Cualquier entrada alta hace conducir a Q_3, corta a Q_4 - y por tanto produce una salida alta.



A	B	Z
0	0	0
0	1	1
1	0	1
1	1	1

Figura 4-7-10.- Compuerta TTL 'O'.

En estas compuertas se propicia un corte sumamente - rápido del transistor Q_4 por la extracción de sus cargas almacenadas hacia el transistor Q_3 en saturación.

El circuito de la Fig. 4-7-11 permite realizar la función $Z = \overline{A_1B_1 + A_2B_2}$. Si $A_2 = A_1$ y $B_2 = B_1$ se obtiene la función O exclusiva. La salida es alta si tanto Q_3 como Q_4 están cortados, para lo cual se requiere que al menos una entrada de cada paso de entrada sea baja.

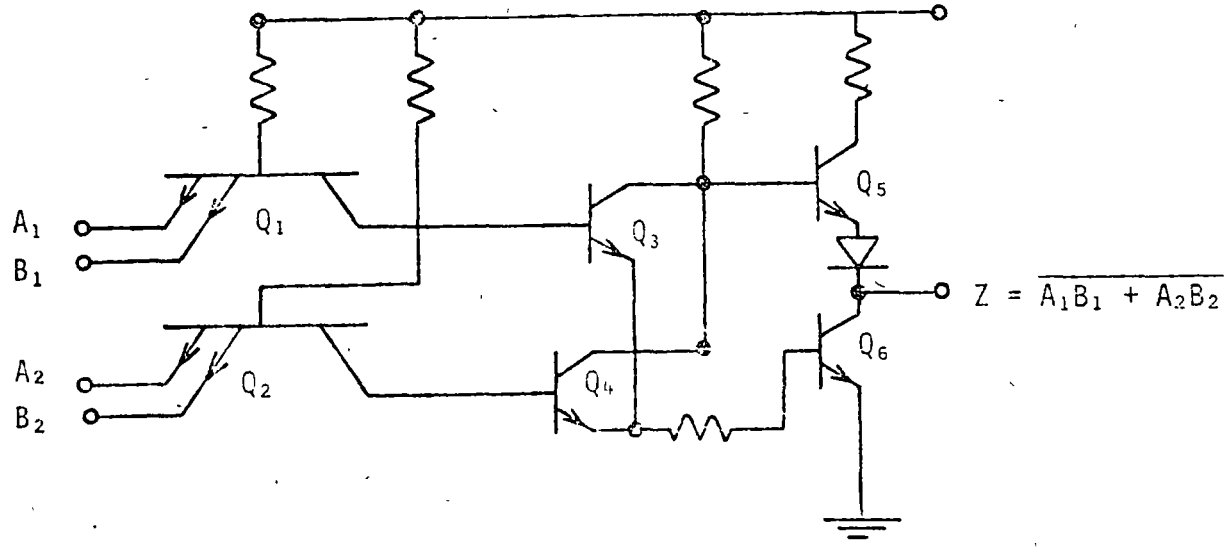


Figura 4-7-11.- Compuerta Y-O-NO.

4-8.- Compuertas ECL (NO O).

Todas las compuertas estudiadas hasta el momento, con excepción de la familia TTL con diodos Schottky operan con transistores saturados, y por tanto presentan retardos

relacionados con el tiempo de desaturación. Las compuertas ECL (lógica acoplada por emisor, también denominadas por - CML) operan, por el contrario, bajo el principio de la conmutación de una corriente, entre las regiones de corte y activas de sus transistores.

La compuerta ECL se muestra en la Fig. 4-8-1. Las - entradas se aplican a las bases de Q_1, Q_2 y Q_3. El transistor Q_4 tiene aplicado a su base un voltaje fijo V_R de referencia. Con todas las entradas bajas (menores que V_R), Q_4 conduce, circulando por la resistencia R_E una corriente del emisor de Q_4 igual a

$$I_g = I_{E4} = \frac{V_R - 0.7 + V_{EE}}{R_E} \qquad 4\text{-}8\text{-}1$$

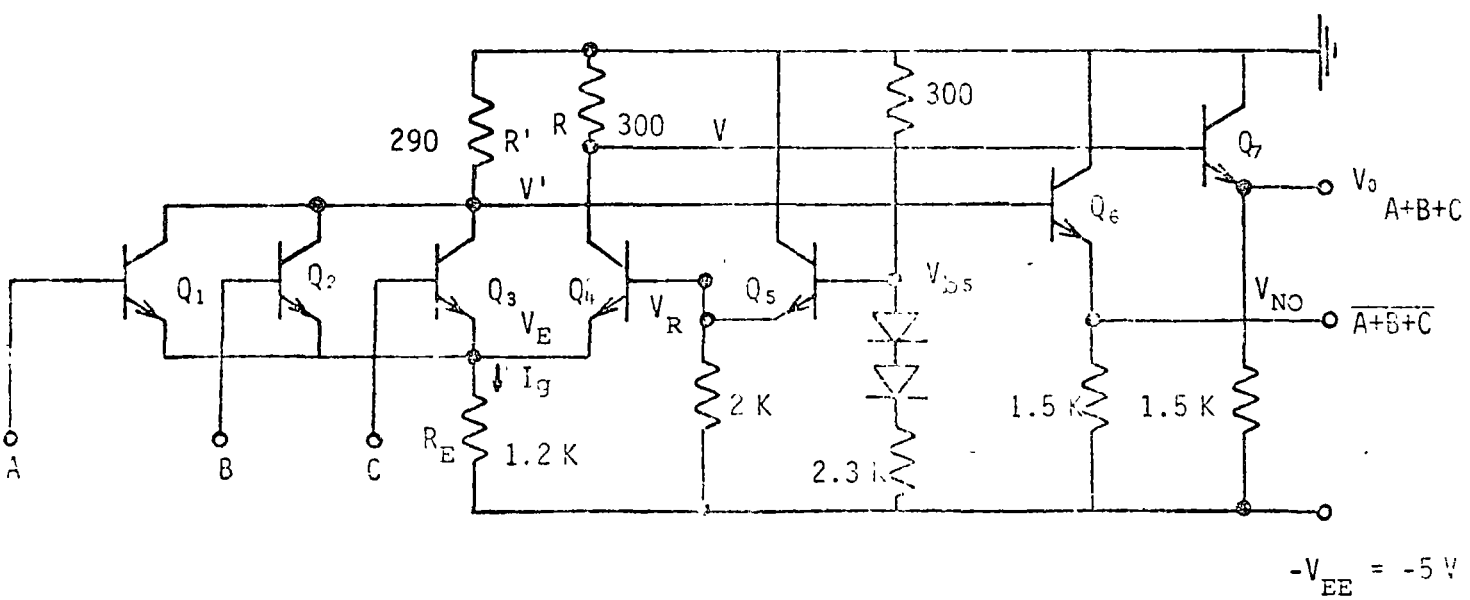


Figura 4-8-1.- Compuerta ECL.

El voltaje V_R esta fijado por Q_5 : $V_R = V_{bs} - V_{be5}$, y $V_{bs} \simeq -0.45$ V, luego V_R se encuentra entre -1.15 y -1.2 V (-1.175 V es el valor estándar). Despreciando el efecto de la resistencia del emisor de Q_7, circula por la resistencia R la corriente de colector $I_{c4} = \alpha_4 I_g$, y $V \simeq -RI_g \simeq -0.85$ V; el voltaje V_O de salida resulta por tanto

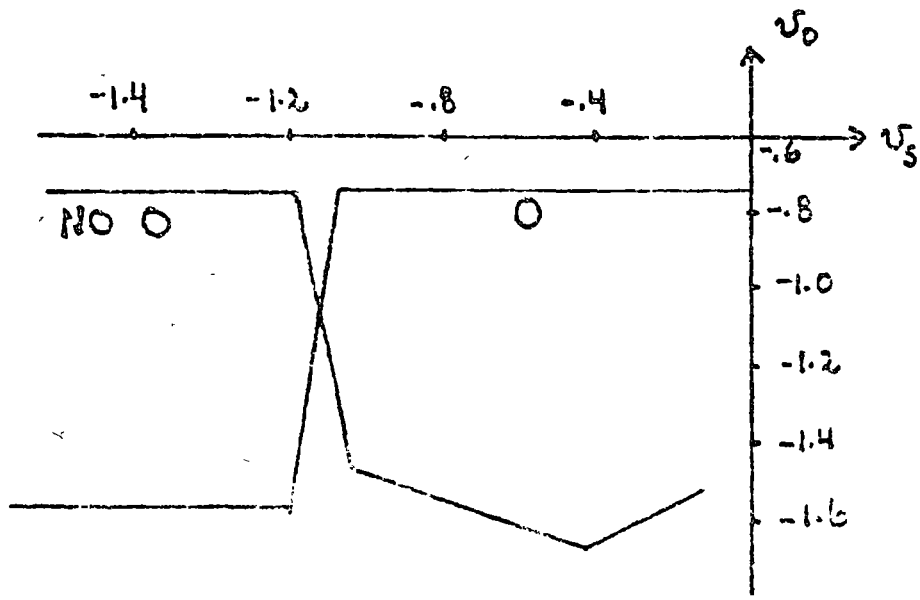
$$V_O = V - V_{be7} \simeq -1.55 \text{ V} \quad \text{(bajo)}$$

Por otro lado, V' es ligeramente inferior a 0V, y el voltaje V_{NO} es aproximadamente igual a -0.75 V (alto).

Supongamos ahora que el voltaje de una de las entradas (V_A) aumenta hasta el valor V_R; la corriente I_g se divide entre dos emisores disminuyendo V, V_O y aumentando V', V_{NO}. Conforme V_A aumenta un poco más, se efectúa la conmutación, circulando por R_E la corriente del emisor de Q_1, que depende del voltaje de entrada. El voltaje V'. y por tanto V_{NO}, disminuyen hasta que Q_1 se sature (curva b de la Fig. 4-8-2). El voltaje V, por el contrario, queda fijo a un valor ligeramente inferior a 0V, y $V_O \simeq -0.75$ V, realizándóse así la operación lógica O .

De las características de transferencia, mostradas en la Fig. 4-8-2, es posible apreciar que la diferencia entre los niveles lógicos es sumamente reducida (0.8 V); ésto ocasiona que el margen de ruido sea también pequeño, de 200 a 300 mV; la compuerta, sin embargo, posee ciertas ventajas, además de su extrema rapidez ($t_p < 5$ nseg.).

En primer lugar, posee una baja impedancia de salida, y una altísima impedancia de entrada, permitiendo un abanico de salida relativamente grande. Por otro lado, la corriente total entregada por la fuente es casi constante, reduciendo a un mínimo el ruido generado por picos de corriente. Además, se cuenta con la salida normal O y su complemento NO O, evitando así la necesidad de utilizar inversores. Las compuertas ECL suelen utilizarse en sistemas digitales de alto rendimiento y velocidad, pero su alto consumo de potencia no se presta para sistemas compactos como minis y microcomputadoras.



Características de transferencia de la compuerta ECL.

Figura 4-8-2.

4.9.- Funciones implícitas o alambradas.

Las salidas de dos o más compuertas con inversores clásicos (no postes totémicos) pueden conectarse en paralelo a fin de implementar implícitamente una función Y (Y alambrado), como se muestra en la Fig. 4-9-1. Al efectuar la conexión, predomina el menor de los voltajes; si uno de ellos es bajo, V_L, y el otro es alto, V_H, el inversor inicialmente saturado habrá de recibir una corriente adicional del paso originalmente cortado.

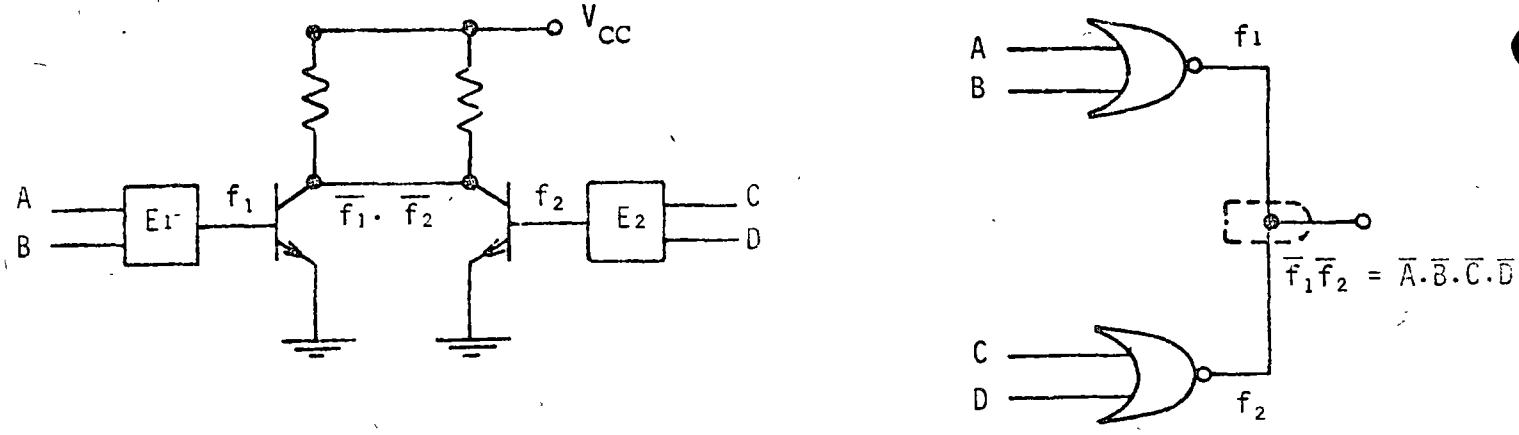


Figura 4-9-1.- Función Y implícita.

En el caso de compuertas RTL (NO O) la conexión ocasiona simplemente una expansión del abanico de entrada. Para compuertas DTL (NO Y), la conexión permite ahorrar efec-

tivamente un paso de lógica.

Al conectar dos o más salidas de seguidores de emisor (con carga resistiva), prevalece el voltaje mayor, obteniéndose por tanto la función implícita 'O' (Fig. 4-9-2).

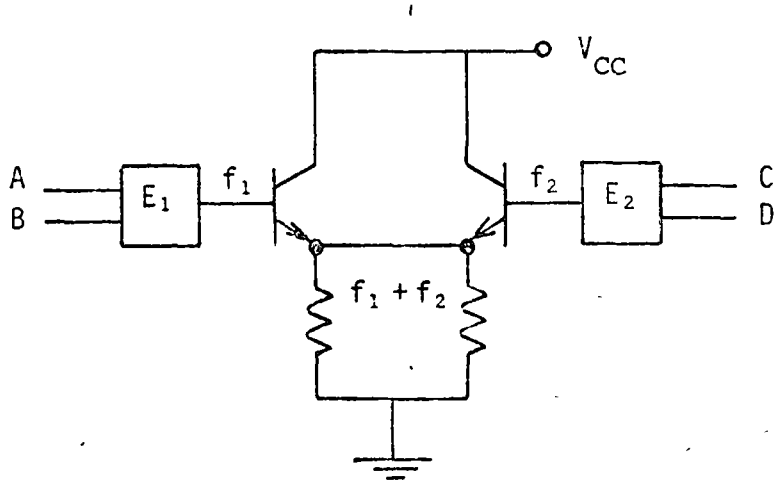
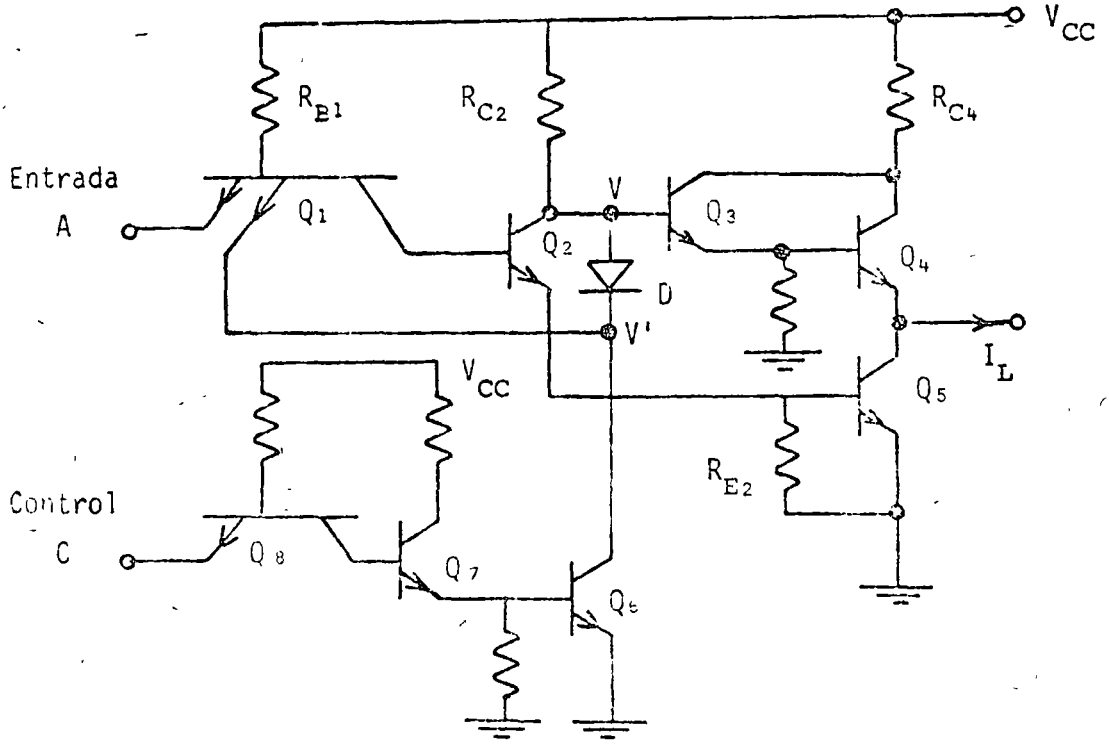


Figura 4-9-2.- Función 'O' implícita.

Este tipo de conexiones no es permitido cuando la salida de una compuerta es un poste totémico (TTL y algunas versiones de DTL), por la posibilidad de que conduzcan los dos transistores del poste y se establezca un voltaje intermedio entre V_H y V_L, que representaría un estado lógico ambiguo, no permisible.

Compuerta de tres estados lógicos (TSL).- La compuerta TSL, con tecnología TTL, contiene un estado eléctrico adicional al '0' y al '1', denominado de alta impedancia, y se utiliza fundamentalmente como interfaz (acoplamiento) entre sistemas TTL ó DTL y líneas para transmitir señales o datos lógicos, 'bus'). A estas líneas suelen conectarse decenas o cientos de elementos lógicos, cuyas salidas quedarían por tanto en paralelo; sólo un elemento lógico, habilitado por una señal de control apropiada puede 'transmitir' su estado a la línea; los demás elementos deben permanecer 'abiertos', o en un estado de alta impedancia, a fin de no degradar la señal lógica presente.



C	función
0	Inversión normal, $\overline{A}$.
1	Alta impedancia, Z. ($\|I_L\| < 40$ uA).

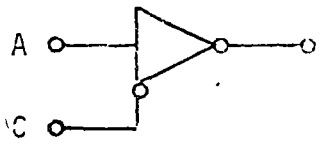
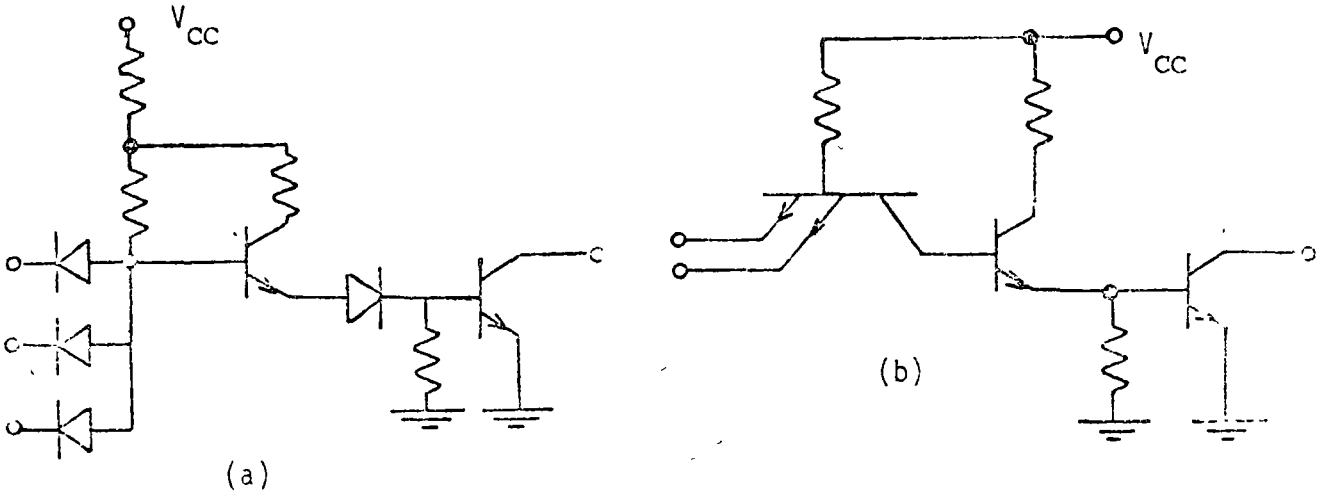


Figura 4-9-3.- Compuerta TSL.

Con la señal de control baja (compuerta habilitada), Q_7 y Q_6 se encuentran cortados, y la salida lógica de la compuerta es $\overline{A}$: al subir la señal de control, Q_6 y Q_7 se saturan; El voltaje V' es igual a $V_{ces_6} \approx 0.1$ a 0.2 V, y V queda fijado entre 0.8 y 0.9 V, impidiendo la conducción de Q_3 y Q_4. De esta manera, todos los transistores del poste totémico de salida están cortados, circulando hacia Q_5 sólo una pequeña corriente de fuga.

Considerando que la compuerta en estado alto puede entregar al menos 5.2 mA de corriente de carga, resulta posible conectar un mínimo de 5.2/.04 = 130 compuertas TSL a una línea de datos.

Compuertas con colectores abiertos.- En la Fig. 4-9-4 se muestra una compuerta DTL y una compuerta TTL que se caracterizan por su salida, que consiste en ambos casos de un inversor con el colector abierto. Ello permite, mediante un resistor externo R_{EXT}, implementar la función implícita Y conectando los colectores de M compuertas.



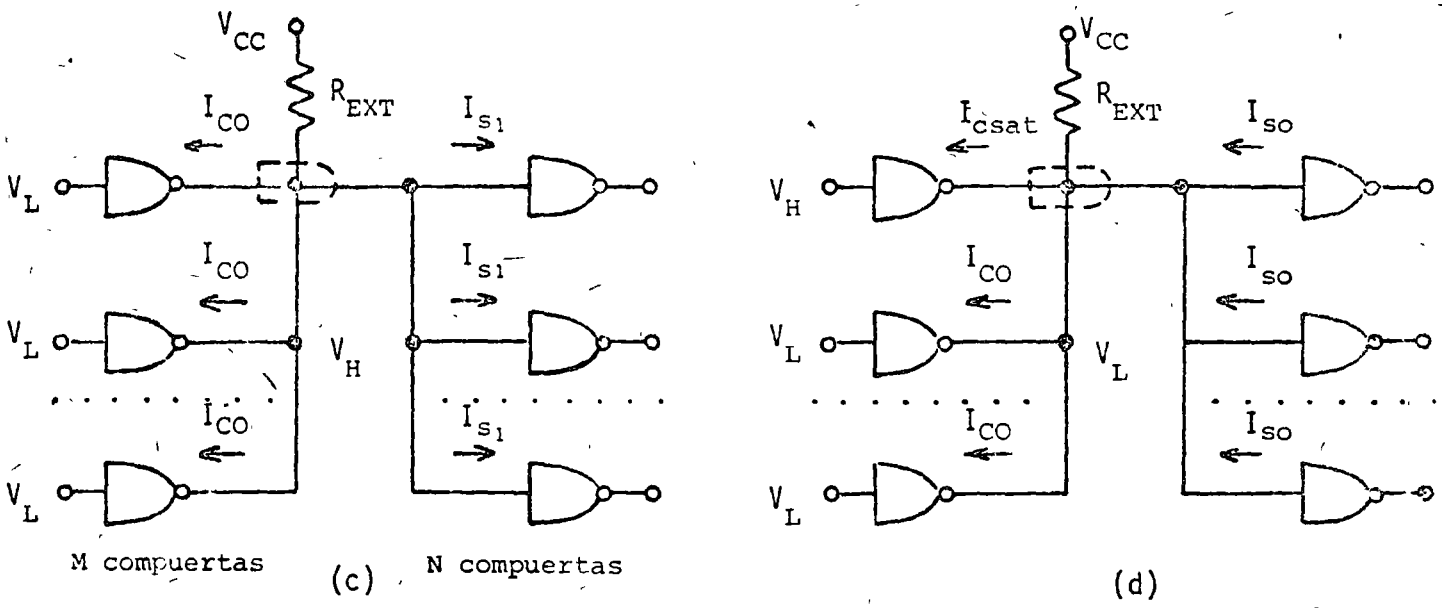


Figura 4-9-4.- Compuertas con colectores abiertos.

El valor de R_{EXT} debe limitarse a fin de que el voltaje V_H sea superior al mínimo requerido; para compuertas TTL, se requiere que $V_H > 2.4$ V a fin de garantizar un márgen de ruido superior a 400 mV si el abanico de salida FO = N = 10. La corriente que circula por R_{EXT} es igual a NI_{S1} (corriente total de las compuertas del abanico de salida) más la corriente de fuga MI_{CO} que circula hacia los colectores; como $V_H = V_{CC} - R_{EXT} I_{REXT}$, el máximo valor de R_{EXT} resulta

$$\overline{R_{EXT}} = \frac{V_{CC} - \underline{V_H}}{M\overline{I_{CO}} + N\overline{I_{s1}}} \qquad 4\text{-}9\text{-}1$$

Si $I_{CO} = 250\ \mu A$, $I_{s1} = 40\ \mu A$ para $\underline{V_H} = 2.4\ V$, $M = 3$ y $N = 5$, se obtiene $R_{EXT} = 2.6/0.95 \simeq 2.74\ K\Omega$ - ($V_{CC} = 5\ V$).

El valor mínimo debe acotarse también, ya que V_L no debe exceder de 400 mV, voltaje para el cual corresponde una corriente máxima de colector de 16 mA. Por otro lado, la corriente mínima I_{so} de cada emisor de carga es aproximadamente igual a 1 mA, luego

$$\underline{R_{EXT}} = \frac{V_{CC} - \overline{V_L}}{\overline{I_{csat}} - \underline{NI_{so}}} \simeq 416\ \Omega \qquad 4\text{-}9\text{-}2$$

Los circuitos con colectores abiertos se utilizan asimismo como elementos de acoplamiento o interfaz entre familias lógicas diferentes (con niveles lógicos distintos), o bien entre TTL, por ejemplo, y sistemas electromecánicos; el circuito de la Fig. 4-9-4b., conocido como 8T80, o bien 8T90 con un solo emisor (inversor), se utiliza con frecuencia con dicho propósito, como se muestra en la Fig. 4-9-5. El sistema electromecánico, que requiere de voltajes y corrientes más altos, puede a su vez acoplarse a los niveles lógicos de TTL por medio de otro circuito de interfaz como el 8T18 de Signetics Co.

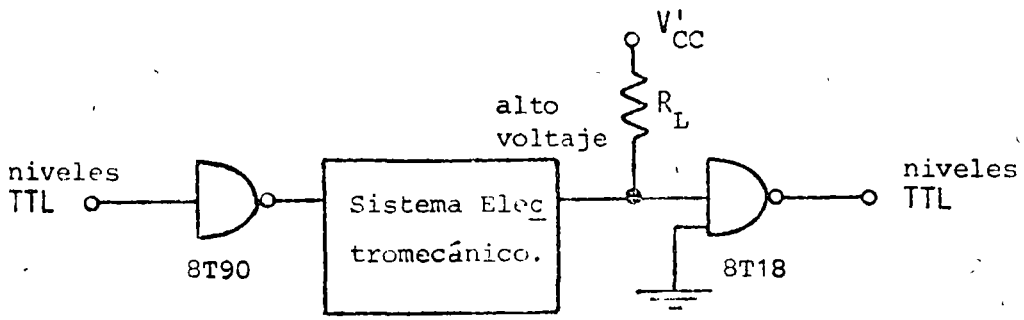


Figura 4-9-5.- Aplicación de circuitos de acoplamiento.

4.10.- Circuitos integrados de pequeña y mediana escala.

Un circuito integrado de pequeña escala (SSI) contiene menos de 12 ó 10 compuertas en una sola pieza de silicio encapsulada. Por lo general, contienen 2, 4 ó 6 compuertas simples o circuitos biestables*, con 14 ó 16 patas de cone-

* Ver Capítulo 5.-

xión. En la Fig. 4-10-1 se muestran dos circuitos típicos: El primer dispositivo contiene 4 compuertas NO Y TTL de - 2 entradas (54/7400A), y el segundo incluye 6 inversores - (5404/7404A).

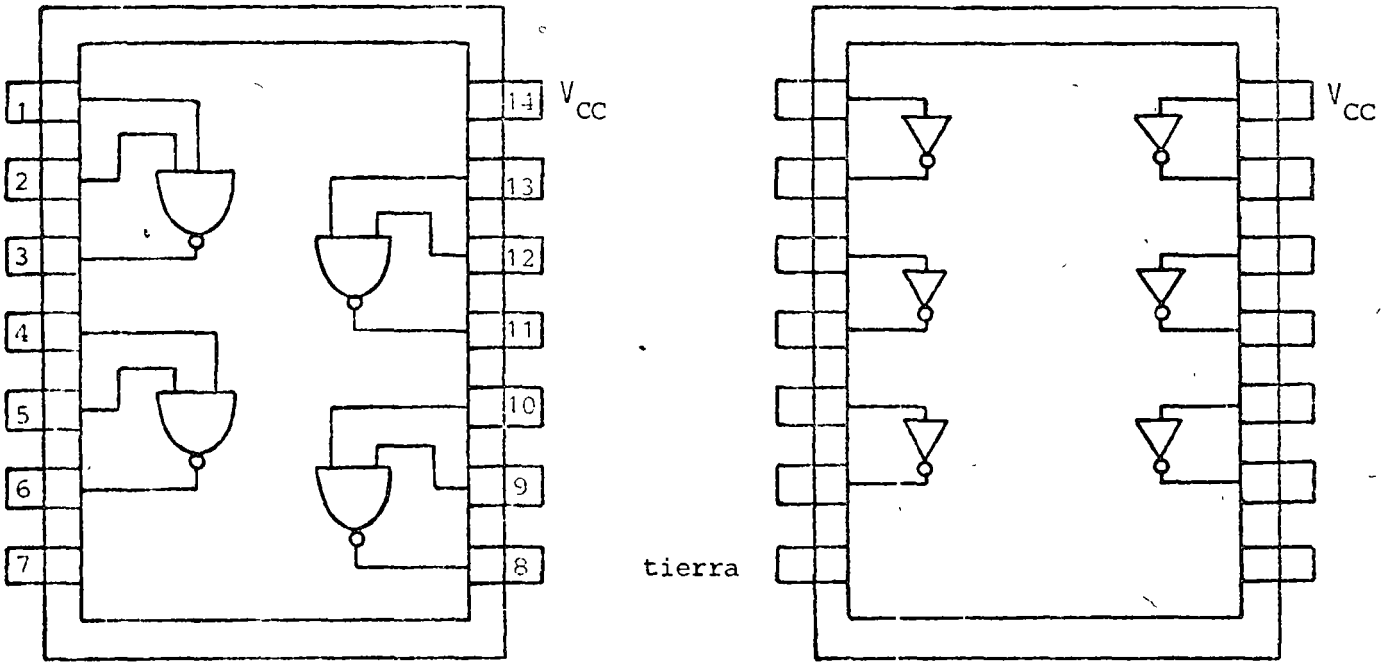


Figura 4-10-1.- Distribución de compuertas en circuitos integrados SSI.

Se denominan circuitos integrados de mediana escala (MSI) aquellos que contienen de 12 a 100 compuertas, como - decodificadores, selectores, unidades aritméticas de 2 ó 4 bits, así como circuitos secuenciales a base de biestables como memorias pequeñas, contadores, registros de corrimiento, etc.

En la Fig. 4-10-2 se ilustra un circuito selector, llamado también multiplexor, que se utiliza para seleccionar una de ocho entradas, D_0 a D_7; esto se logra por medio

de un código o clave binaria de tres bits, A, B y C, como se muestra en la tabla de verdad. La señal $\overline{S}$ permite controlar al circuito; cuando $\overline{S} = 0$, éste queda habilitado.

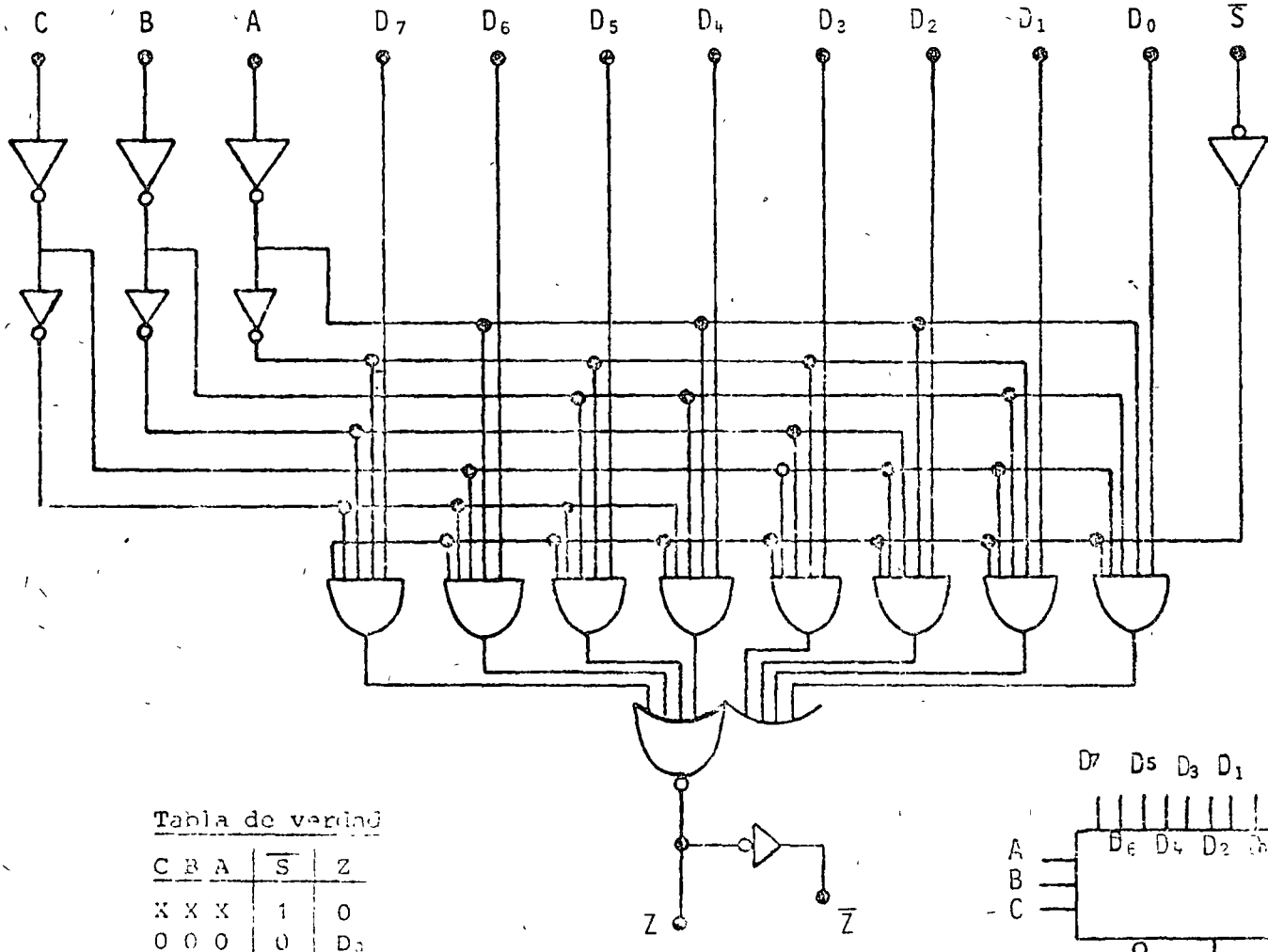


Tabla de verdad

C	B	A	$\overline{S}$	Z
X	X	X	1	0
0	0	0	0	D_0
0	0	1	0	D_1
0	1	0	0	D_2
0	1	1	0	D_3
1	0	0	0	D_4
1	0	1	0	D_5
1	1	0	0	D_6
1	1	1	0	D_7

(X indica que la entrada es irrelevante).

Figura 4-10-2.- Selector de ocho entradas, tabla de verdad y símbolo correspondiente.

En la Fig. 4-10-3 se muestra un sumador binario de dos bits (54/7482). Las entradas son A_1, B_1 (bits menos significativos), A_2, B_2 (bits más significativos) y C_{in1} (bit de acarreo). Las salidas son S_1, S_2 y un bit de acarreo C_0. Las reglas de adición de un sistema binario se muestran en la tabla 4-10-1; la suma será 1 si una o tres entradas son 1s. El bit de acarreo será 1 si al menos dos entradas son 1s.

$$S = (A\overline{B} + \overline{A}B)\overline{C}_{in} + (\overline{A}\overline{B} + AB)C_{in}$$

$$C_0 = AB + (A\overline{B} + \overline{A}B)C_{in}$$

A	B	C_{in}	S	C_0
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Tabla 4-10-1.- Tabla de verdad y ecuaciones de un sumador de un bit.

En el sumador de dos bits, el bit de acarreo de salida C_{01} se utiliza como entrada C_{in2} para la suma de la segunda pareja de bits. El circuito integrado contiene 14

patas de conexión, pero se utilizan solamente 10 de las mismas (5 entradas, 3 salidas, V_{CC} y tierra).

En la fabricación de circuitos integrados MSI se pretende incrementar la densidad, o sea, el número de compuertas por área de silicio. Para ello, se procura eliminar componentes pasivas y activas en las compuertas internas, que no requieren de especificaciones tan estrictas como las de salida. Dado que su abanico de salida es fijo (no suele exceder de 3), y no se requiere impulsar cargas capacitivas considerables, una compuerta TTL interna no requiere del poste totémico de salida. A fin de no reducir su velocidad, se procura evitar la saturación de uno o dos de sus transistores.

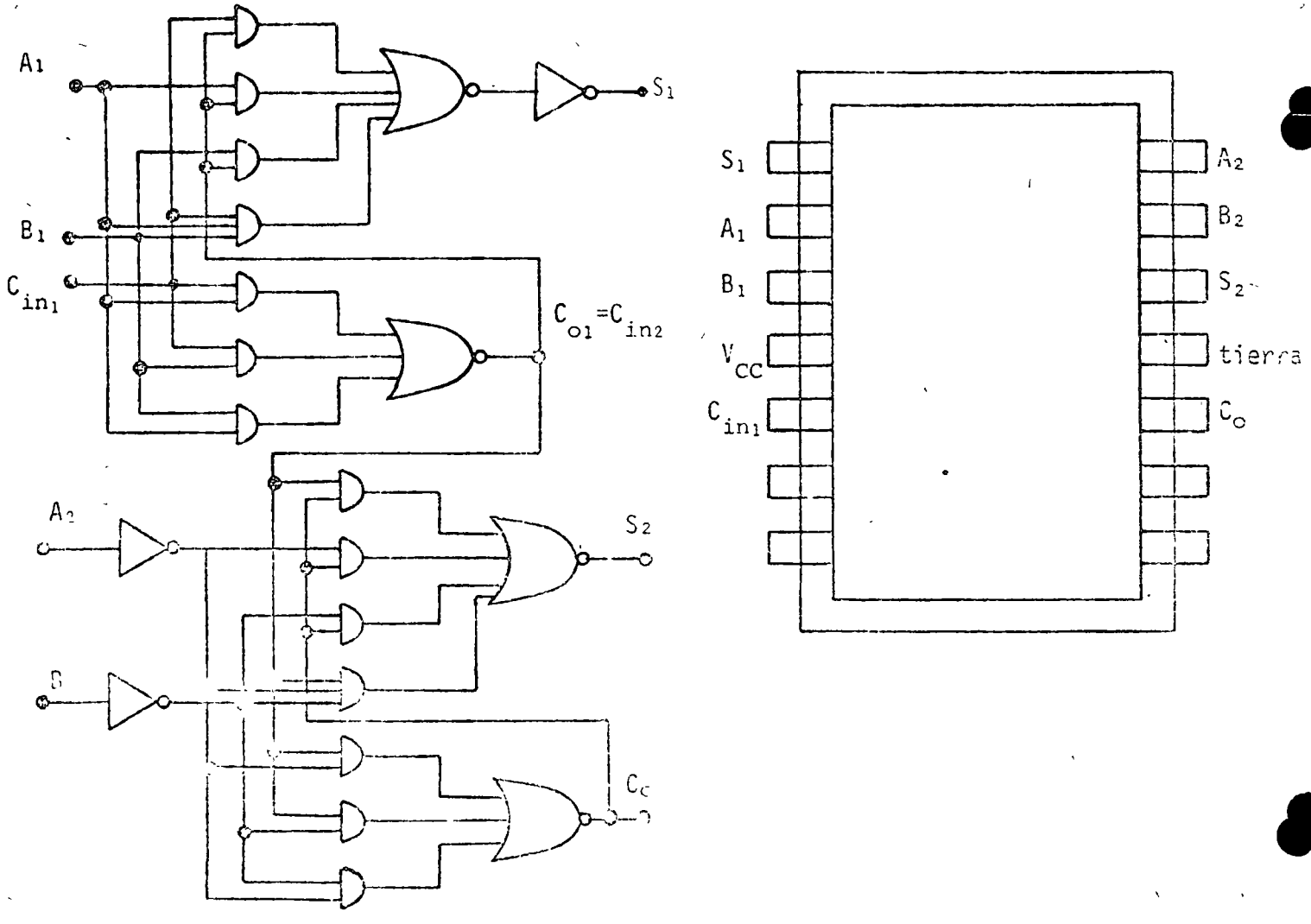


Figura 4-10-3.- Sumador binario de 2 bits 54/7482.

En la Fig. 4-10-4a. se muestra un inversor interna - con un solo transistor Q_3 de salida; si el voltaje de entrada es alto, tanto Q_2 como Q_3 operan en su región activa. Como $v_{B1} \approx 2.1\,V$, $v_{C2} = v_X \approx 1.4\,V$ (v_{C2} queda fijado por el diodo base-emisor de Q_1); el diodo D, a su - vez, ocasiona que el voltaje de salida v_0 sea de 0.7 V - aproximadamente.

En el circuito de la Fig. 4-10-4b., el transistor - Q_2 posee un segundo emisor conectado a la salida. Con las entradas altas, $v_{C2} \approx 1.4\,V$, luego $v_X = v_0 = 0.7\,V$, conservando a Q_3 fuera de saturación.

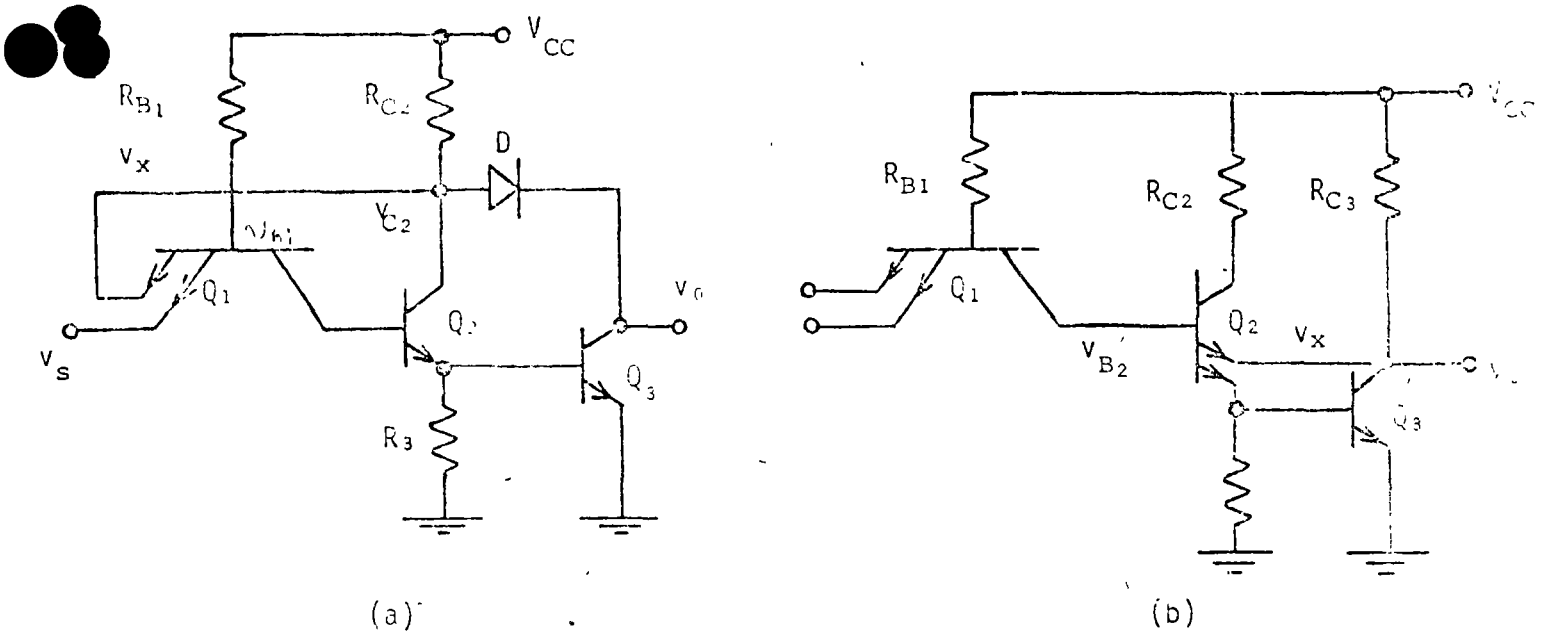


Figura 4-10-4.- Compuertas TTL internas con diodos fijadores.

4.11.- Circuitos integrados de gran escala (LSI).

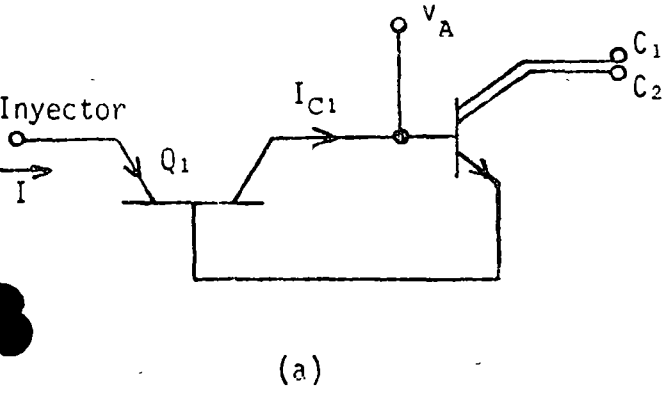
Los circuitos integrados de gran escala contienen más de 100 compuertas en una sola pieza o pastilla de silicio; se utilizan principalmente para memorias, unidades aritméticas y lógicas (ULAs) complejas, arreglos lógicos programables, calculadoras y los modernos microprocesadores.

Si bien se han realizado circuitos LSI con tecnología bipolar (TTL e incluso ECL), los circuitos integrados en gran escala han sido fabricados preferentemente con tecnología MOS (ver el Capítulo 7), que permite densidades mucho mayores a la vez que consumos de potencia apreciablemente inferiores a los que requieren las familias lógicas ECL y TTL, si bien la velocidad de éstos últimos es superior.

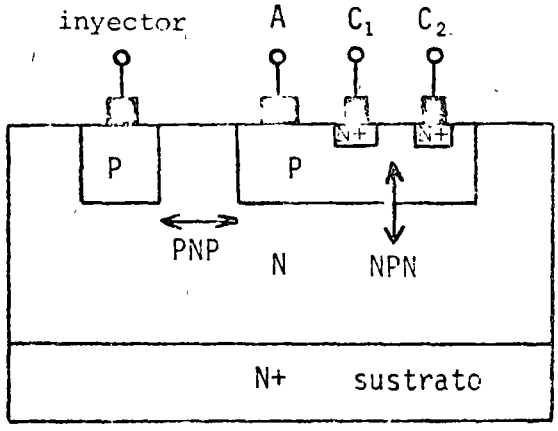
Es justamente la mayor velocidad de la tecnología bipolar la que ha propiciado una actividad febril de investigación para lograr densidades altas, bajo precio y bajo consumo de potencia con dicha tecnología; la investigación ha culminado recientemente con el desarrollo de la familia I^2L (lógica de inyección).

El principio en que se basan las compuertas I^2L es sumamente sencillo; se pretende conmutar la corriente de una fuente de corriente, por medio de una señal lógica A; la fuente de corriente la constituye un transistor PNP de alta ganancia, Q_1; su corriente de colector, $I_c \approx I$ circulará hacia la base del transistor Q_2 (NPN) que hace las veces de inversor, si $v_A > 0.7$ V; de lo contrario, la corriente fluirá hacia la salida de la compuerta excitadora, y Q_2 estará cortado (Fig. 4-11-1).

Las salidas, formadas por los colectores de Q_2, se conectan con las salidas de otras compuertas para formar implícitamente diversas funciones lógicas



(a)



(b)

Figura 4-11-1.- Estructura básica de las compuertas I^2L.

El circuito de la Fig. 4-11-2, por ejemplo, permite obtener las funciones $\overline{A} \cdot \overline{B}$ y $A + B$. Si ambas entradas son bajas ($A = B = '0'$), tanto Z_1 como Z_2 son altas, y por tanto Z_3 es baja. Si cualquier entrada es alta, tanto Z_1 como Z_2 son bajas, y $Z_3 = '1'$.

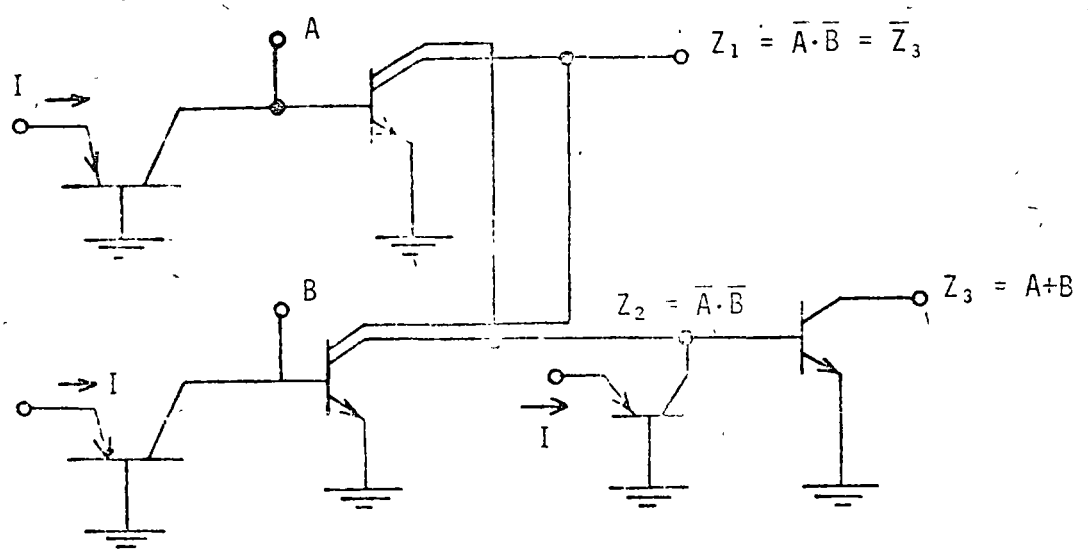


Figura 4-11-2.- Compuerta con funciones NOO y O.

El principio de conmutación permite eliminar resistores, que ocupan un área considerable; este factor, aunado al empleo de áreas comunes en la fabricación (Fig. 4-11-1b.) permite un grado de integración del orden de 150 compuertas por mm^2, equivalente al de la tecnología MOS, a la cual la tecnología I^2L supera en velocidad, habiéndose logrado tiempos de propagación menores de 10 nseg. con corrientes de 10 a 20 nA.

Aún cuando está en plena fase de desarrollo, la tecnología I^2L se ha aplicado ya exitosamente en relojes digitales, y en microprocesadores.

4-12.- Problemas.

1.- Dibuje los diagramas lógicos de las siguientes funciones

a) $AC + \overline{A}B$

b) $AB + \overline{A}C + B$

c) $A(B + \overline{CB})$

utilizando compuertas Y, O y NO.

2.- Un semisumador es un circuito para sumar dos dígitos binarios, sin entrada de acarreo C_{in}. Demuestre que el circuito de la Fig. 4-12-1 es un semisumador, y que el circuito de la Fig. 4-12-2 permite realizar una suma completa.

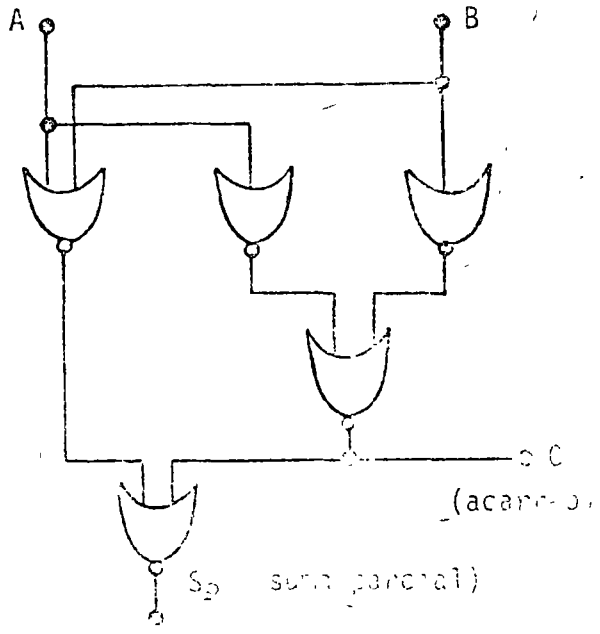


Figura 4-12-1.

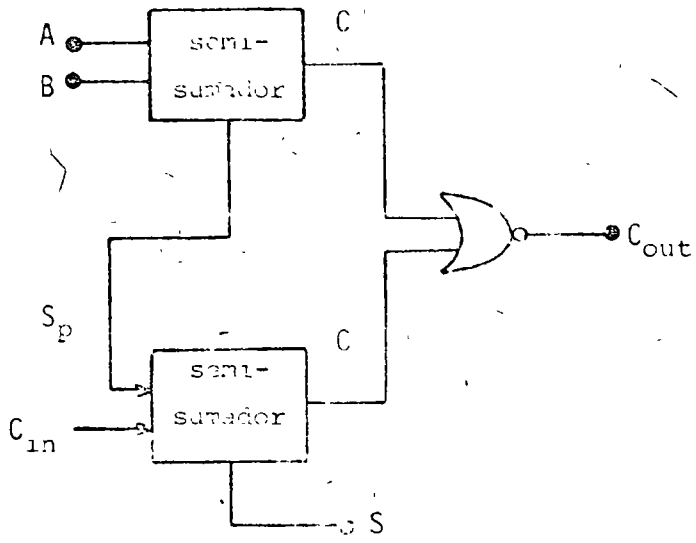
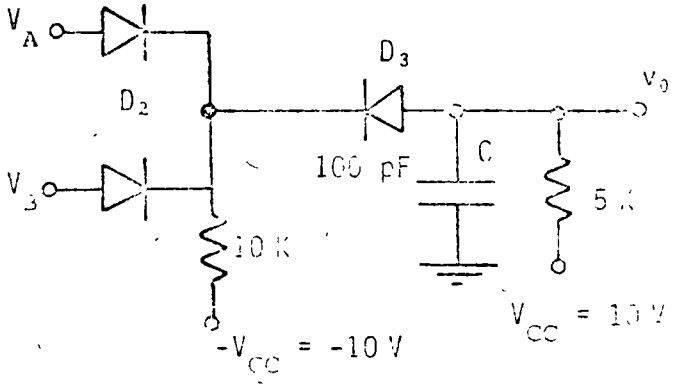


Figura 4-12-2.

3.- Demuestre que el circuito de la Fig. 4-12-3 se comporta como una compuerta O . Determine el voltaje de salida si V_A = 0 V, y V_b varía como se muestra en la figura; los diodos conducen con un voltaje de 0.7 V.



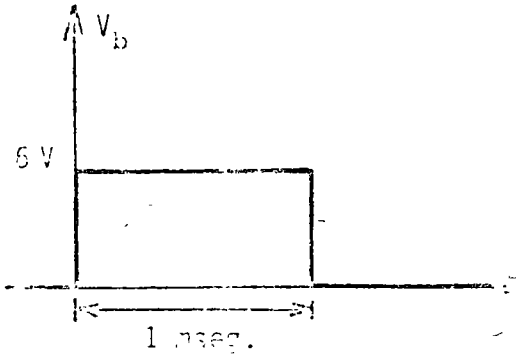
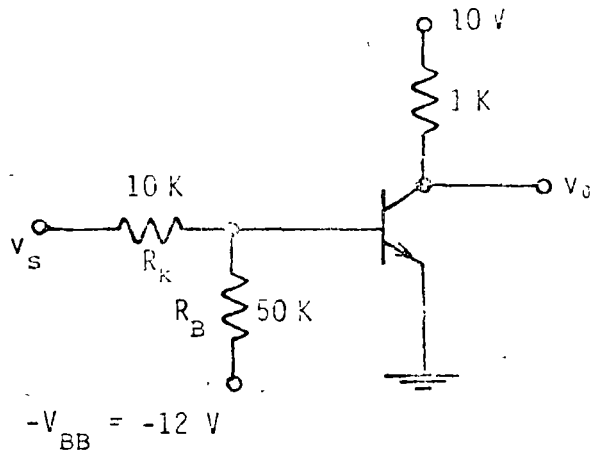


Figura 4-12-3.

4.- Determine el margen de ruido en estado alto del inversor de la Fig. 4-12-4, con sus valores nominales, si el abarico de salida es igual a 2 y $\beta_F = 20$.



$V_a = 0.5$ V

$V_{be} = 0.7$ V

$V_{ces} = 0.2$ V

$I_{CO} \simeq 0$

Figura 4-12-4.

5.- Repita el problema anterior si se añade al circuito una entrada adicional.

6.- Describa la operación de la compuerta DTL de la Fig. 4-12-1, demuestre que es una compuerta NO O.

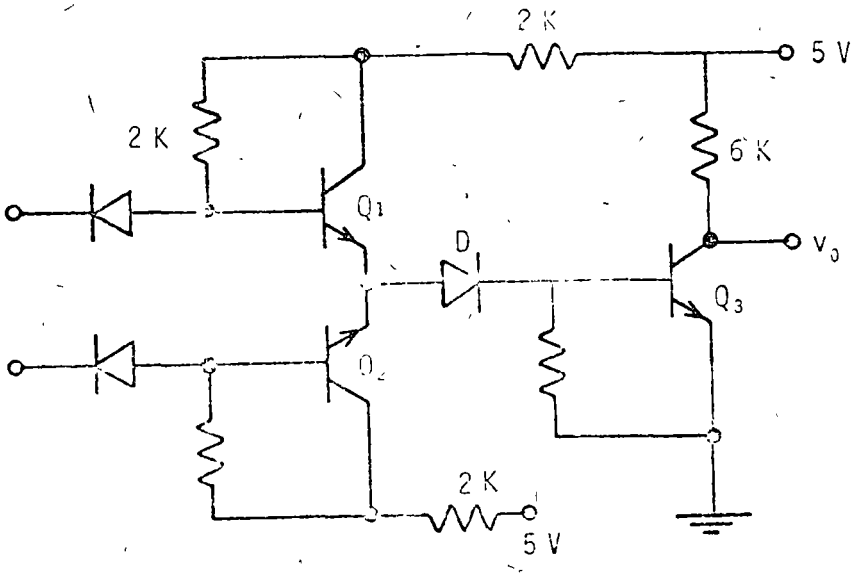
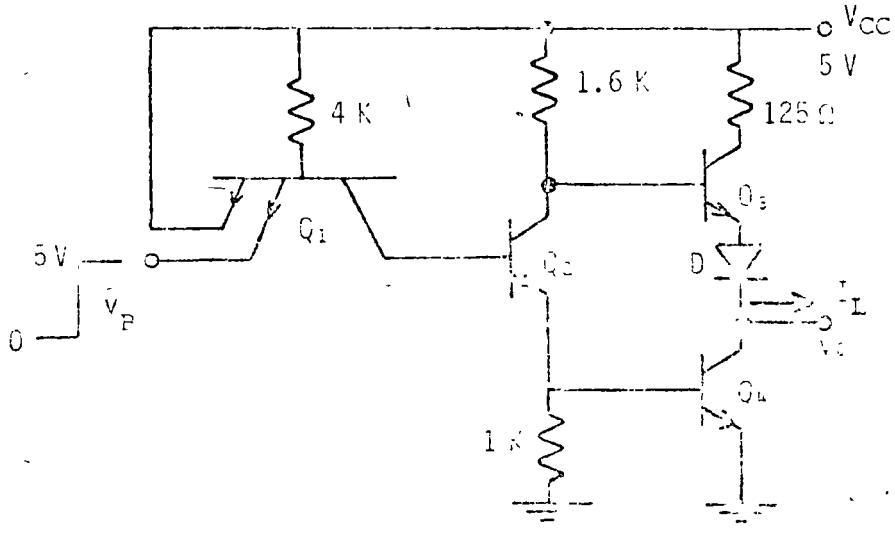


Figura 4-12-5.

7.- Determine la potencia total entregada por la batería V_{CC} a la compuerta TTL estándar si: a) la salida es baja, e $I_L = -10$ mA; b) la salida es alta, $I_L = 400$ μA y $V_{OH} = 3.56$ V.



$V_{be} = V_d = 0.7$ V

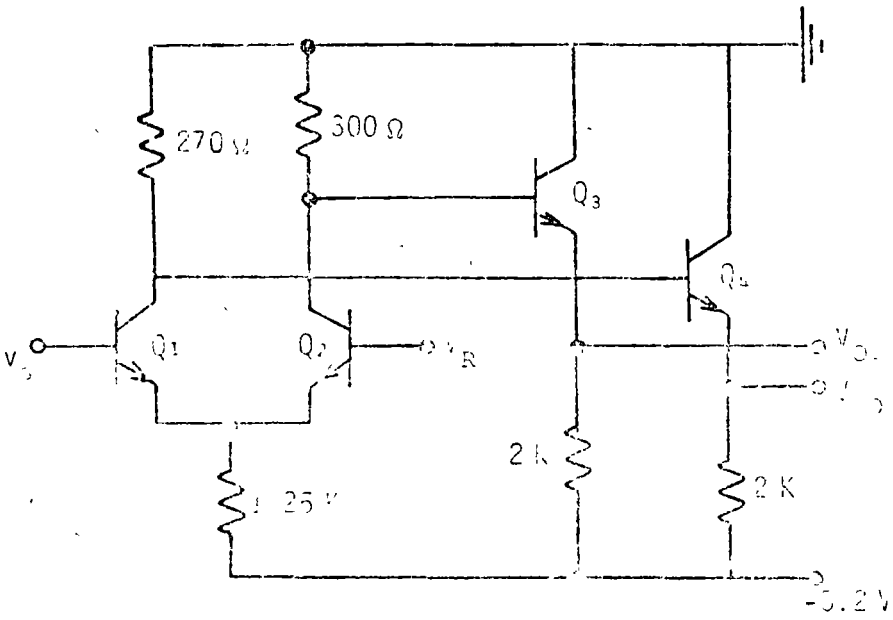
$V_{ces} = 0.3$ V

$V_{bc1} = 0.7$ V

Figura 4-12-6

8. Determine el valor de $\underline{\beta_F}$ requerido para el transistor Q_4 de la compuerta TTL estándar (Fig. 4-12-6) si $I_L = -I_{csat.} = -12$ mA.

9.- Obtenga las funciones de transferencia V_0 vs. v_s y V_{NO} vs. v_s del inversor ECL mostrado en la Fig. 4-12-7 si v_s varía de -2 a 0 V, $V_R = 1.25$ V, y suponiendo que se requiere un voltaje de 150 mV para que se lleve a cabo totalmente la conmutación.



$\beta_F = 50$

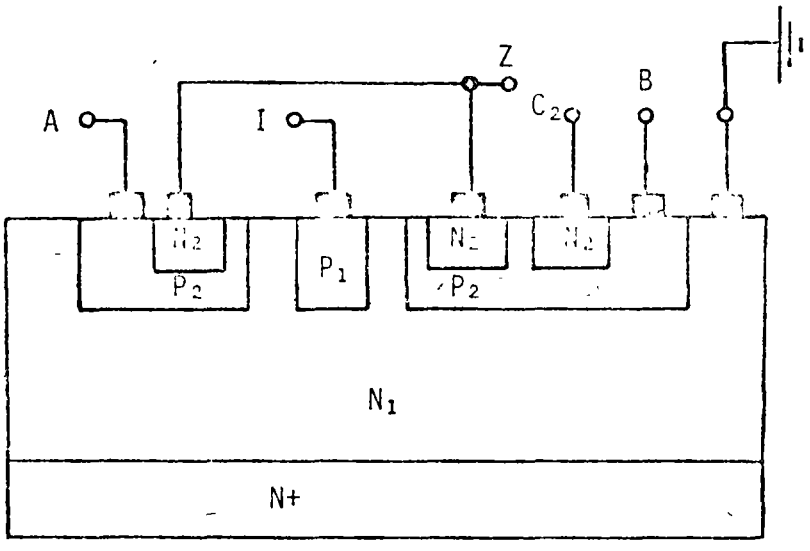
$V_{be} = 0.75$ V

$V_{ces} = 0.2$ V

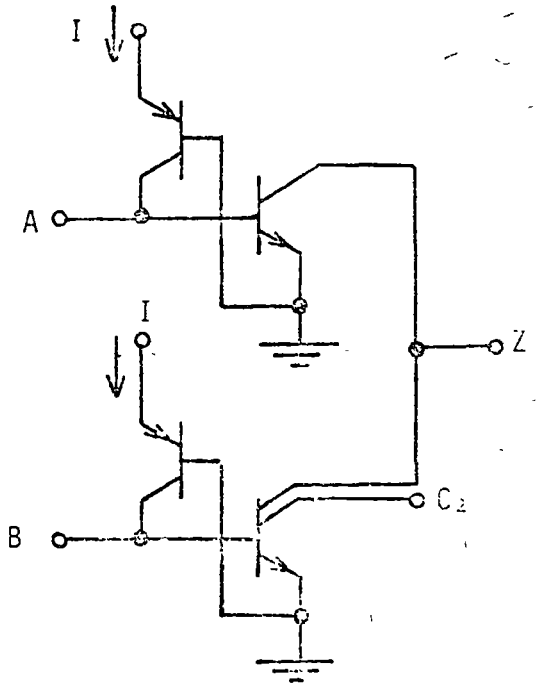
para todos los transistores.

Figura 4-12-7

10.- Muestre que la estructura de la Fig. 4-12-8a, corresponde al circuito 4-12-8b. ¿Qué función lógica se implementa?



(a)



(b)

Figura 4-12-8.

BIBLIOGRAFIA:

1.- Strauss, L.:
'Wave generation and shaping'
Mc. Graw-Hill, 2a. Ed., 1970.

2.- Casasent, D.:
'Digital Electronics'
Quantum Publishers Inc., 1974.

3.- Texas Instruments Inc.:
'Designing with TTL Integrated Circuits'
Mc. Graw-Hill, 1971.

4.- Hnateck, E.R.:
'A Users Handbook of Integrated Circuits'
John Wiley & Sons, 1973.

5.- Kohonen, T.:
'Digital Circuits and Devices'
Prentice-Hall Inc., 1972.

6.- Garrett, L.S.:
'Integrated Circuit Digital Logic Families'
IEEE Spectrum, Oct. 1970, pp 46-58; Nov. 1970, pp 63-72.

7.- Hart C.M., Slob A., Wulms E.J.:
'Bipolar LSI takes a new direction with integrated injection logic'
Electronics, Oct. 3 1974, pp 111-118.

8.- Altman, L.:
'The new LSI'
Electronics, Jul 10 1975, pp 81-92.

DIRECTORIO DE ASISTENTES AL CURSO INTRODUCCION AL PROCESAMIENTO DIGITAL. (DEL 5 AL 9 DE SEPTIEMBRE DE 1977).

	NOMBRE Y DIRECCION	EMPRESA Y DIRECCION
1.	GUSTAVO CORTES POPOLUS Claveles # 317 Col. Frac. La Florida Edo. de México Tel. 562-54-73	PUROMEX, S. A. Filiberto Gómez # 204 Z. I. Tlalnepantla Edo. de México Tel. 565-86-41
2.	HECTOR SALGADO GALICIA Pánuco # 314 Col. Bellavista Salamanca, Gto. Tel. 8-19-79	PETROLEOS MEXICANOS, REFINERIA EN SALAMANCA, GTO. Arbol Grande s/n Domicilio Conocido Salamanca, Gto. Tel. 8-00-97
3.	ING. FRANCISCO TALAN GONZALEZ Valle de Zumpango # 20 Col. El Mirador Naucalpan, Edo. de México Tel. 560-01-10	I.P.N. Edificio 6 U. P. Zacatenco Col. Lindavista México 14, D. F.
4.	JOSE LUIS VAZQUEZ GUEVARA Av. Colón Ote. # 1034 Int. 6 Col. Moctezuma Orizaba, Ver Tel. 530-33 Ext. 224	CERVECERIA MOCTEZUMA, S. A. Sur 10 y Poniente 9 Orizaba, Ver.