Sistemas Digitales

Manual de asignatura

Sistema de Universidades Tecnológicas

ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL

Programa 2004

Créditos

Elaboró:
MC Marco Antonio Ramírez Barrientos

Revisó:

Colaborador:

Autorizó:

Contenido

Objetivo general

CONOCER LOS DISPOSITIVOS ELECTRONICOS PARA EL CONTROL DE POTENCIA

Habilidades por desarrollar en general

Escribir la habilidad propuesta que se debe desarrollar en esta asignatura.

Introducción al Diseño Digital

1.1 INTRODUCCIÓN.

Saber en la Teoría (12 hrs.)

Cuando la mayoría de nosotros escucha el término 'digital' inmediatamente pensamos en una calculadora digital' o 'computadora digital'. Lo anterior probablemente puede atribuirse a la forma tan impresionante en que ahora la persona promedio tiene acceso a poderosas computadoras y calculadoras, a bajo costo. Es importante señalar que ambas representan sólo una de las muchas aplicaciones de los circuitos y principios digitales. Los circuitos digitales se emplean en productos electrónicos tales como juegos de video, hornos de microondas y sistemas de control para automóviles, así como en equipos de prueba como medidores, generadores y osciloscopios. Además, las técnicas digitales han reemplazado muchos de los 'circuitos analógicos' utilizados en productos de consumo como radios, televisores y equipos para grabación y reproducción de alta fidelidad.

1.2 CARACTERÍSTICAS DE LA SEÑAL ANALÓGICA Y DIGITAL.

1.2.1 Señal Analógica.

El mundo real es fundamentalmente analógico y la mayor parte de las cantidades físicas son de naturaleza analógica, y a menudo estas cantidades son las entradas y salidas de un sistema que las monitorea, que efectúa operaciones con ellas y que las controla. Algunos ejemplos son la temperatura, la presión, la posición, la velocidad, el nivel de un líquido, la rapidez de flujo y varias más.
Para aprovechar las técnicas digitales cuando se tienen entradas y salidas analógicas, deben seguirse tres pasos:

1. Convertir las entradas analógicas del 'mundo real" a la forma digital.
2. Procesar (realizar operaciones con) la información digital.
3. Convertir las salidas digitales a la forma analógica del mundo real.

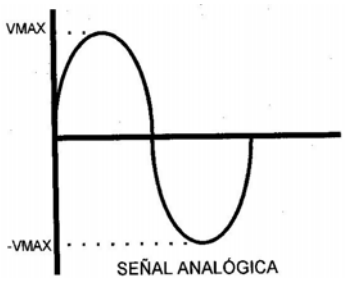


Figura 1.1

1.2.2 Señal Digital.

Las señales digitales basan su funcionamiento en dos estados únicamente, nivel alto (uno lógico) y nivel bajo (cero lógico), aunque la lógica de trabajo puede ser de dos clases que podrían definirse como complementarias:

- **Positivas:** funciona a nivel alto.
- **Negativas:** funciona a nivel bajo.

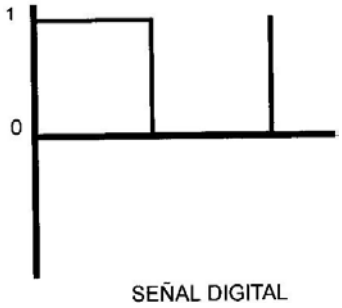


Figura 1.2

1.2.3 Sistemas Lógicos digitales.

Un sistema digital es una combinación de dispositivos diseñado para manipular cantidades físicas o información que estén representadas en forma digital; es decir, que sólo puedan tomar valores discretos. La mayoría de las veces, estos dispositivos son electrónicos, pero también pueden ser mecánicos, magnéticos o neumáticos.
Los niveles alto y bajo dependerán de las técnicas de construcción con que se fabrican los circuitos integrados que albergan los componentes lógicos digitales. En lógica positiva son:

TTL	ECL	MOS
"1"… 5V (2.4 V mínimo)	"1"… 0V	"1"… 15V (5 V mínimo)
"0"… 0V (0.8 V máximo)	"0"… -5.2V	"0"… 0V (3.5 V máximo)

Para garantizar la compatibilidad entre los circuitos lógicos por diferentes tecnologías existen integrados adaptadores de nivel, aunque en la actualidad la tecnología MOS dispone integrados

lógicos totalmente compatibles con los TTL. Se puede establecer una relación de ventajas e inconvenientes de unas tecnologías frente a otras:

- TTL y ECL poseen unos márgenes de tensión mas reducidos, aunque ello exige una mayor estabilización en las alimentaciones.
- TTL permite trabajar con corrientes más elevadas.
- ECL presenta el inconveniente de necesitar una alimentación negativa.
- ECL es la que permite trabajar a mayores velocidades, mientras que MOS es más lenta.
- MOS presenta menor consumo de potencia.
- El costo de los integrados TTL es menor frente a los ECL.

Los circuitos integrados lógicos pueden ser clasificados de varias formas:

- **Por su proceso de fabricación:**

  Monolíticos: Todos los componentes son formados en la misma pastilla semiconductora.
  Peliculares: Los componentes se van formando sobre la superficie de un sustrato aislante (vidrio, cerámico).
  Multilaminares: Los componentes se forman en capas diferentes y son unidos por un sustrato común.
  Híbridos: utilizan todas las técnicas anteriores para formar los componentes.

- **Por el tipo de transistor empleado:**

  Bipolares: transistores NPN o PNP.
  MOS: transistores MOS.

- **Por su complejidad:**

  Pequeña escala de integración: SSI, un máximo de 12 unidades lógicas.
  Media escala de integración: MSI, entre 13 y 100 unidades lógicas.
  Alta escala de integración: LSI, entre 101 y 1000 unidades lógicas.
  Muy alta escala de integración: VLSI, entre 1001 y 1 000 000 de unidades lógicas.

1.3 CÓDIGOS DE NUMERACIÓN.

1.3.1 Sistemas de numeración.

La necesidad de establecer cantidades para poder ponderar magnitudes, contar y operar con ellas, hace que se prevalezcan unos sistemas de numeración a través de unos códigos perfectamente estructurados que facilitaran dichas tareas. El sistema de numeración mas utilizado en la actualidad es el decimal, el cual se compone de 10 dígitos, del 0 al 9 de forma que al combinarlos se forman diferentes números:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9
10, 11, 12, 13,……19
20, 21, 22, 23,……29
.
.
90, 91, 92, 93,……99
100, 101, 102,……...109
.
.
1000, 1001, 1002,….1009

- TTL y ECL poseen unos márgenes de tensión mas reducidos, aunque ello exige una mayor estabilización en las alimentaciones.
- TTL permite trabajar con corrientes más elevadas.
- ECL presenta el inconveniente de necesitar una alimentación negativa.
- ECL es la que permite trabajar a mayores velocidades, mientras que MOS es más lenta.
- MOS presenta menor consumo de potencia.
- El costo de los integrados TTL es menor frente a los ECL.

Los circuitos integrados lógicos pueden ser clasificados de varias formas:

- **Por su proceso de fabricación:**

  Monolíticos: Todos los componentes son formados en la misma pastilla semiconductora.
  Peliculares: Los componentes se van formando sobre la superficie de un sustrato aislante (vidrio, cerámico).
  Multilaminares: Los componentes se forman en capas diferentes y son unidos por un sustrato común.
  Híbridos: utilizan todas las técnicas anteriores para formar los componentes.

- **Por el tipo de transistor empleado:**

  Bipolares: transistores NPN o PNP.
  MOS: transistores MOS.

- **Por su complejidad:**

  Pequeña escala de integración: SSI, un máximo de 12 unidades lógicas.
  Media escala de integración: MSI, entre 13 y 100 unidades lógicas.
  Alta escala de integración: LSI, entre 101 y 1000 unidades lógicas.
  Muy alta escala de integración: VLSI, entre 1001 y 1 000 000 de unidades lógicas.

1.4 CÓDIGOS DE NUMERACIÓN.

1.3.2 Sistemas de numeración.

La necesidad de establecer cantidades para poder ponderar magnitudes, contar y operar con ellas, hace que se prevalezcan unos sistemas de numeración a través de unos códigos perfectamente estructurados que facilitaran dichas tareas. El sistema de numeración mas utilizado en la actualidad es el decimal, el cual se compone de 10 dígitos, del 0 al 9 de forma que al combinarlos se forman diferentes números:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9
10, 11, 12, 13,……19
20, 21, 22, 23,……29
.
.
90, 91, 92, 93,……99
100, 101, 102,……...109
.
.
1000, 1001, 1002,….1009
.
.
A cada número le corresponde un único código, y para ello se puede establecer una regla para formarlos, donde cada dígito se descompone en un coeficiente, una base y un exponente:

$$186 = 1 \cdot 10^2 + 8 \cdot 10^1 + 6 \cdot 10^0 = 100 + 80 + 6 = 186$$

centenas decenas unidades

Coeficientes: 1, 8, 6
Base: 10 (decimal)
Exponente: corresponde al número de orden que ocupa cada coeficiente desde la derecha y empezando por el cero.

Otro de los sistemas de numeración utilizados es el octal, el cual lo componen 8 dígitos (del 0 al 7) y cuya forma de componer sus códigos es idéntica al decimal:

0, 1, 2, 3, 4, 5, 6, 7
10, 11, 12, 13, 14, 15, 16, 17
20,…..

Se puede establecer una regla para pasar siempre de cualquier código al decimal.

- *Para poder pasar de un código de cualquier sistema al correspondiente en decimal, se multiplicara cada coeficiente por la base elevada al exponente que ocupa y posteriormente se sumara todo.*

$$186_8 = 2 \cdot 8^2 + 7 \cdot 8^1 + 2 \cdot 8^0 = 186$$

De igual forma existe una regla que permite pasar de un código en sistema de decimal a cualquier otro sistema.

- *Para pasar de un código en sistema decimal al correspondiente en otro sistema, se procede a dividir sucesivamente el código decimal por la base del nuevo sistema, así hasta que el cociente ya no sea divisible. Entonces se tomara como digito mayor el último cociente y los siguientes dígitos los formaran los restos obtenidos hasta el primero.*

Aplicando esta última regla al código anterior:

186	8	
26	23	8
2	7	2

El resultado obtenido es: 272_8

1.3.3 Códigos Binarios.

Los sistemas lógicos digitales basan su funcionamiento en dos estados (cero y uno), por tanto será necesario construir un código basado en dos dígitos que permitan ponderar magnitudes y operar con ellas, en definitiva construir números. Al código más utilizado se le llama binario natural, el cual toma como base el dos y se codifica de forma:

$$C_n b^n C_{n-1} b^{n-1} ..... \; C_3 b^3 C_2 b^2 C_1 b^1 C_0 b^0$$

1 0 1 1, es un numero binario natural de 4 dígitos o bits.

- ***Bit.*** *unidad básica de información binaria.*

Siguiendo las reglas para pasar de un sistema cualquiera al decimal y viceversa se obtiene:

$$1110 = 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 8 + 4 + 2 + 0 = 14$$

14	2	
0	7	2
	1	3 2
		1 **1**

..... 1110

Hasta ahora sólo se ha visto como se representan códigos con parte entera, pero en los sistemas de numeración también existe una parte fraccionaria que debe ser codificada. En decimal la parte fraccionaria se representa a la derecha de la parte entera y separada de esta por una coma:

$$24{,}375 = 2 \cdot 10^1 + 4 \cdot 10^0 , 3 \cdot 10^{-1} + 7 \cdot 10^{-2} + 5 \cdot 10^{-3} = 20 + 4{,}0.3 + 0.07 + 0.005 = 24{,}375$$

En la parte fraccionaria viene afectada la base por exponentes negativos. En este caso también existen reglas para pasar de un sistema a otro.

> ➢ *Para pasar la parte fraccionaria de un código en sistema decimal a otro cualquiera, se procede a multiplicar dicha parte por la base del sistema al que se va a pasar, tomándose la parte entera obtenida de las sucesivas multiplicaciones hasta que acabe con la parte fraccionaria.*

$$24{,}375 = 11000{,}011$$
$$0{,}375 \cdot 2 = 0.75$$
$$0{,}75 \cdot 2 = 1.5$$
$$0{,}5 \cdot 2 = 1$$

> ➢ *Para pasar la parte fraccionaria de un código binario a su correspondiente en decimal, se procede a multiplicar dicha parte por las potencias de dos y al final se suma todo.*

$$0{,}011 = 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} = 0 + 0.25 + 0.125 = 0.375$$

Dentro de los códigos binarios existen aquellos que para formar sus codificaciones siguen unas reglas no ponderadas, como el ser **continuos**, como ocurre con el código **GRAY** o binario reflejado y el **JOHNSON**, que forma sus códigos de forma adyacente (se diferencian en un bit) y además la primera codificación y la última también son adyacentes, lo que hace que sean **cíclicos**.

Decimal	GRAY (4 Bits)	JOHNSON (8 Bits)
0	0 0 0 0	00000000
1	0 0 0 1	00000001
2	0 0 1 1	00000011
3	0 0 1 0	00000111
4	0 1 1 0	00001111
5	0 1 1 1	00011111
6	0 1 0 1	00111111
7	0 1 0 0	01111111
8	1 1 0 0	11111111
9	1 1 0 1	11111110
10	1 1 1 1	11111100
11	1 1 1 0	11111000
12	1 0 1 0	11110000
13	1 0 1 1	11100000
14	1 0 0 1	11000000
15	1 0 0 0	10000000

Tabla 1.1

1.3.4 Códigos BCD.

Los BCD (Binary Codex Decimal) son códigos particulares que trabajan en decimal, pero están codificados en binario, por ellos tienen 10 codificaciones diferentes que se corresponden con los del sistema decimal. Algunos de los más importantes se pueden en la Tabla 1.2

Decimal	BCD - natural 8 4 2 1	BCD-Exceso 3	AIKEN 2 4 2 1
0	0 0 0 0	0 0 1 1	0 0 0 0
1	0 0 0 1	0 1 0 0	0 0 0 1
2	0 0 1 0	0 1 0 1	0 0 1 0
3	0 0 1 1	0 1 1 0	0 0 1 1
4	0 1 0 0	0 1 1 1	0 1 0 0
5	0 1 0 1	1 0 0 0	1 0 1 1
6	0 1 1 0	1 0 0 1	1 1 0 0
7	0 1 1 1	1 0 1 0	1 1 0 1
8	1 0 0 0	1 0 1 1	1 1 1 0
9	1 0 0 1	1 1 0 0	1 1 1 1

Tabla 1.2

Para operar con los códigos BCD se procede a agruparlos en tétradas (grupos de 4 bits), de forma que cada tétrada corresponde a un dígito en decimal. Así, pasar al sistema decimal o viceversa se convierte en tarea fácil y rápida.

Decimal	BCD - natural	BCD-Exceso 3	AIKEN
45	0100 0101	0111 1000	0100 1011

Tabla 1.3

1.4 FAMILIAS LÓGICAS.

Los circuitos lógicos digitales tienen como estructura básica las unidades o puertas lógicas, que son elementos electrónicos integrados en circuitos y cuyo funcionamiento (respuesta de la salida en función de las entradas) viene determinado por las funciones lógicas básicas. Cada una de estas funciones da nombre a una puerta, mostrando la evolución de su salida a través de su tabla de verdad.

1.4.1 FUNCIÓN IDENTIDAD.

En esta función la entrada y la salida son iguales, su misión por tanto consiste en proporcionar ganancia de corriente.

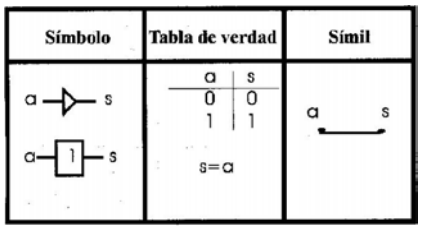


Figura 1.3a

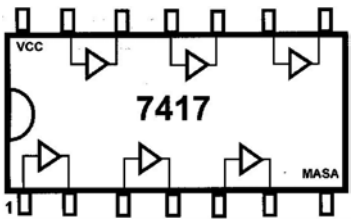


Figura 1.3b

Nota: a partir de ahora se considerarán las entradas como variables y serán referenciadas con las letras del abecedario.

1.4.2 FUNCIÓN INVERSORA.

La salida es lo contrario de la entrada. Es la negación lógica.

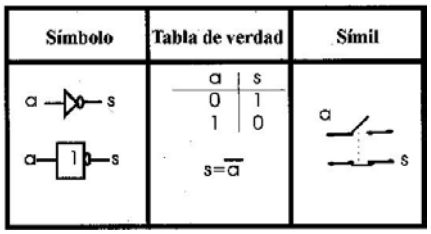


Figura 1.4a

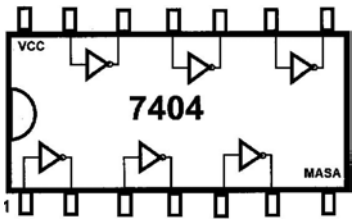


Figura 1.4b

1.4.3 FUNCIÓN AND.

Responde a la función que da como resultado a su salida el producto lógico, es decir, sólo dará uno a su salida si todas las entradas están a uno también.

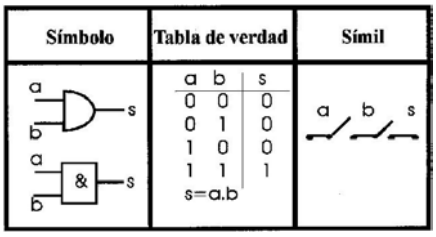


Figura 1.5a

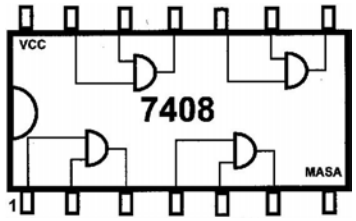


Figura 1.5b

1.4.4 FUNCIÓN NAND.

Esta función es la complementaria de la AND, de manera que sólo cuando todas las entradas valen uno la salida es cero.

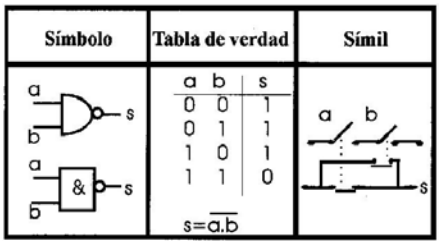


Figura 1.6a

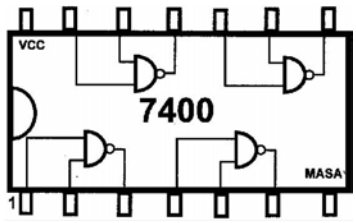


Figura 1.6b

1.4.5 FUNCIÓN OR.

Responde a la función que da como resultado a su salida la suma lógica, es decir, dará uno a su salida siempre que alguna de las entradas esté a uno.

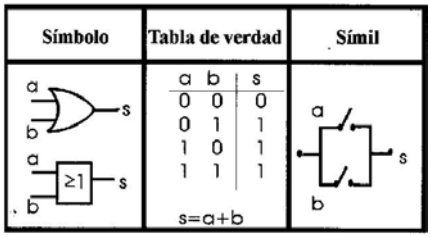


Figura 1.7a

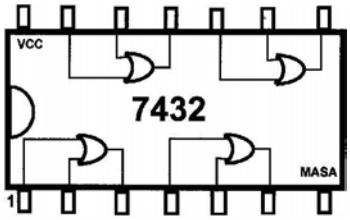


Figura 1.7b

1.4.6 FUNCIÓN NOR.

Es la función complementaria de la OR, de manera que sólo valdrá uno la salida cuando todas las entradas estén a cero.

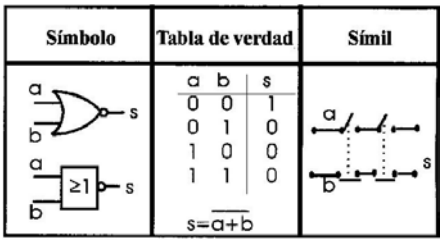


Figura 1.8a

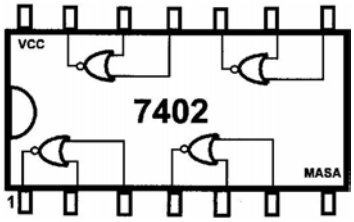


Figura 1.8b

1.4.7 FUNCIÓN OR-EXCLUSIVA.

Se dice que es una función generadora de paridad par, ya que da a su salida uno cuando el número de unos en sus entradas es impar.

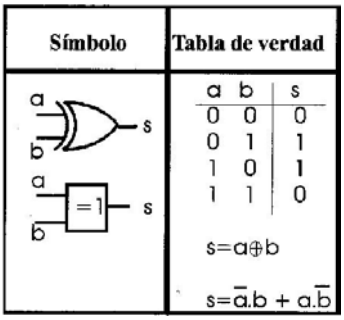


Figura 1.9a

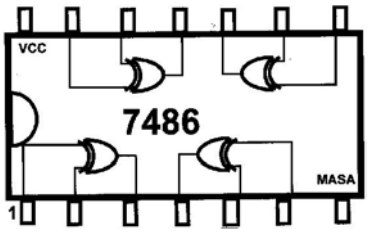


Figura 1.9b

1.4.8 FUNCIÓN NOR-EXCLUSIVA.

Se dice que es una función generadora de paridad impar, ya que da a su salida uno cuando el número de unos en sus entradas es par.

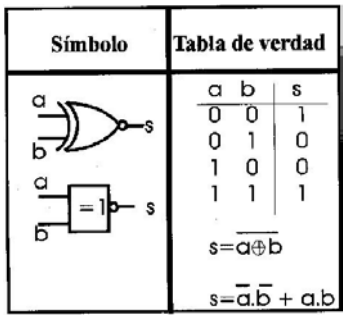


Figura 1.10a

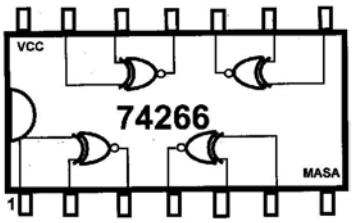


Figura 1.10b

II
Circuitos Lógicos Combinacionales

2.1 ÁLGEBRA DE BOOLE.

Saber en la Teoría (14 hrs.)

A mediados del siglo XIX, George Boole, filósofo y matemático se basó en el sistema binario de numeración, es decir, tomando como únicos elementos el cero y el uno lógico. Este sistema establece una serie de propiedades, postulados y teoremas respecto de la suma y el producto para operaciones con las funciones lógicas. Con todo esto se consigue que la resolución de problemas con automatismos electrónicos sea mucho más sencilla.

2.1.1 Propiedades.

		Suma	Producto
i.	**Conmutativa:**	$a+b=b+a$	$a\cdot b=b\cdot a$
ii.	**Elemento neutro:**	$a+0=a$	$a\cdot 1=a$
iii.	**Elemento simétrico:**	$a+\bar{a}=1$	$a\cdot\bar{a}=0$
iv.	**Asociativa:**	$(a+b)+c=a+(b+c)$	$(a\cdot b)\cdot c=a\cdot(b\cdot c)$
v.	**Distributiva:**	$a+(b\cdot c)=(a+b)\cdot(a+c)$	$a\cdot(b+c)=(a\cdot b)+(a\cdot c)$

2.1.2 Postulados.

i. Dualidad: establece que en cualquier igualdad si cambiamos los unos por ceros o las sumas por productos y viceversa, lo que se obtiene también se cumple.

	Suma	Producto
ii. Idempotencia:	$a+a=a$	$a\cdot a=a$
iii. Absorvente:	$a+1=1$	$a\cdot 0=0$
iv. Doble negación:	$\bar{\bar{a}}=a$	
v. Simplificación:	$a+a\cdot b=a$	$a\cdot(a+b)=a$
vi. Leyes de Morgan:	$\overline{a+b+c}=\bar{a}\cdot\bar{b}\cdot\bar{c}$	$a\cdot(b+c)=(a\cdot b)+(a\cdot c)$

2.1.3 Teoremas.

i. Si $a+b=c$ entonces $\overline{a+b}=\bar{c}$

ii. Si $a\cdot b=c$ entonces $\overline{a\cdot b}=\bar{c}$

iii. $a + (\bar{a} \cdot b) = a + b$

iv. $(a + \bar{b}) \cdot b = a \cdot b$

v. $a + b = \overline{\bar{a} \cdot \bar{b}}$

vi. $a \cdot b = \overline{\bar{a} + \bar{b}}$

2.2 FORMA CANÓNICA DE UNA FUNCIÓN BOOLEANA.

Una función booleana es la que muestra la relación en términos de sumas y productos lógicos existente entre las variables de entrada de dicha función, pues bien, se dice que una función esta en forma canónica cuando en cada uno de sus términos aparecen todas las variables de entrada, bien sean negadas o sin negar.
Los términos representan las condiciones de las variables de entrada para que se cumpla la función ("1" en lógica positiva). Se puede ver un ejemplo para una función cuya salida (F), es función de tres entradas (a, b, c):

$$F = a \cdot b \cdot c + \bar{a} \cdot b \cdot \bar{c} + a \cdot \bar{b} \cdot \bar{c}$$

Existen dos maneras de representar las funciones booleanas en su forma canónica, en forma Minterm y en forma de Maxterm.

Minterm: es la forma de representar una función canónica mediante términos en forma de sumas de productos de todas sus variables.

$$F_3 = a \cdot b \cdot c + a \cdot \bar{b} \cdot c + a \cdot b \cdot \bar{c}$$

Maxterm: es la forma de representar una función canónica mediante términos en forma de productos de sumas de todas sus variables.

$$F_3 = (a + \bar{b} + c) \cdot (a + b + \bar{c}) \cdot (\bar{a} + \bar{b} + c) \cdot (\bar{a} + b + c)$$

Toda función no canónica puede ser representada en forma de Minterm o Maxterm siguiendo unas reglas:

Pasar a Minterm: se debe multiplicar cada término al que le falte una variable por "uno", entendiendo por "uno" la suma de la variable que falta más ella misma negada.

$$F = a \cdot b + a \cdot \bar{b} \cdot c = a \cdot b \cdot (c + \bar{c}) + a \cdot \bar{b} \cdot c = a \cdot b \cdot c + a \cdot b \cdot \bar{c} + a \cdot \bar{b} \cdot c$$

Pasar a Maxterm: se debe sumar a cada término que le falte una variable "cero", entendiendo por "cero" el producto de la variable que falta por sí misma negada.

$$F = (a + b) \cdot (a + \bar{b} + c) = (a + b) + (c \cdot \bar{c}) \cdot (a + \bar{b} + c) = (a + b + c) \cdot (a + b + \bar{c}) \cdot (a + \bar{b} + c)$$

NOTA: Se puede observar como en ambos casos se ha aplicado la propiedad distributiva del álgebra de Boole.
De igual Forma, existen unas normas para que cualquier función canónica pueda ser representada en forma de minterm o maxterm indistintamente. Para ello se debe partir de la tabla de verdad de una función, y se obtendrán los minterm para los valores en que la función vale "uno":

No. orden	a b c	F
0	0 0 0	1
1	0 0 1	1
2	0 1 0	0
3	0 1 1	0
4	1 0 0	0
5	1 0 1	1
6	1 1 0	1
7	1 1 1	0

Tabla 2.1

$F = \bar{a}\cdot\bar{b}\cdot\bar{c} + \bar{a}\cdot\bar{b}\cdot c + a\cdot\bar{b}\cdot c + a\cdot b\cdot\bar{c}$, se puede representar como $F = \sum_3(0,1,5,6)$

Minterm y Maxterm:

1. Obtener los términos de la función negada.
2. Aplicar Morgan.
3. Aplicar a los términos obtenidos la ecuación $\overline{m_i} = M_{2^n-1-i}$

Se pueden aplicar estos tres puntos a la función anterior, la cual se cumplía para los minterm 0, 1, 5 y 6, siendo por tanto la función negada:

$\overline{F} = \sum_3(2,3,4,7)$
$\overline{\overline{F}} = \sum_3(\overline{2,3,4,7})$, se aplicará Morgan a los términos negados.
$F = \prod_3(\bar{2},\bar{3},\bar{4},\bar{7})$, se aplica la regla descrita a todos los términos, para el dos será: $\overline{m_2} = M_{2^3-1-2} = M_5$
$F = \prod_3(5,4,3,0)$
$F_3 = (a+\bar{b}+c)\cdot(a+\bar{b}+\bar{c})\cdot(\bar{a}+b+c)\cdot(\bar{a}+\bar{b}+\bar{c})$, así queda la función en forma de Maxterm con todas sus variables.

Implementación de funciones.

Al paso de una función lógica a su circuito mediante compuertas lógicas se le llama implementar la función, si se aplica a la función vista en el apartado anterior.

$$F = a \cdot b \cdot c + \bar{a} \cdot b \cdot \bar{c} + a \cdot \bar{b} \cdot \bar{c}$$

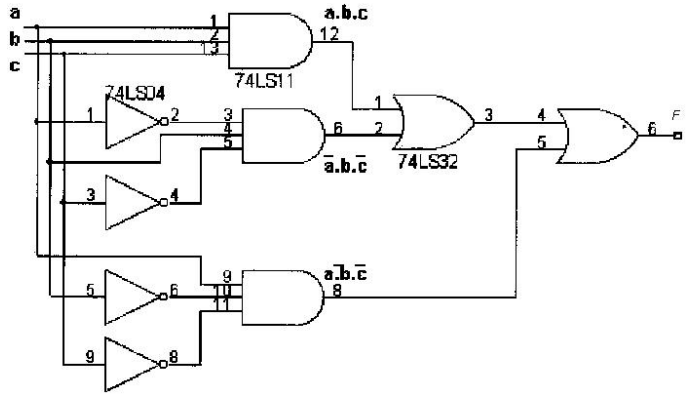


Figura 2.1

Existe una manera especial de implementar funciones aplicando las leyes de Morgan. Con este sistema se consigue el circuito de una función mediante compuertas NAND o NOR únicamente. Esto se puede ver con algo tan sencillo como es realizar la función lógica AND implementada tanto por compuertas NAND como NOR:

$F = a \cdot b$, ésta es la función AND o producto lógico.

$F = \overline{a \cdot b}$, esto es lo mismo por el postulado de la doble negación y la tenemos en compuertas NAND.

$F = \overline{\bar{a} + \bar{b}}$, este es el resultado aplicando Morgan y la tenemos mediante compuertas NOR.

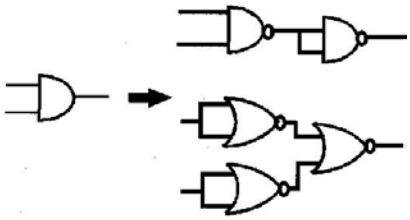


Figura 2.2

Obsérvese el siguiente ejemplo:

$$F1 = a \cdot b \cdot c + a \cdot \bar{b} \cdot c = \overline{a \cdot b \cdot c + a \cdot \bar{b} \cdot c}$$
$$F1 = \overline{\overline{a \cdot b \cdot c}} + \overline{\overline{a \cdot \bar{b} \cdot c}}$$
$$F2 = (a + b + c) \cdot (a + b + \bar{c}) = \overline{(a + b + c) \cdot (a + b + \bar{c})}$$
$$F2 = \overline{\overline{(a + b + c)}} + \overline{\overline{(a + b + \bar{c})}}$$

Se puede deducir que, aplicando dobles negaciones y deshaciendo las que interesen, es posible implementar cualquier función mediante compuertas NAND o NOR

2.3 SIMPLIFICACION DE FUNCIONES LOGICAS.

Es probable que en muchas ocasiones aparezcan diferentes caminos para realizar una misma tarea, y que además estos caminos no sean idénticos, si no que unos resulten más sencillos o inmediatos que otros, aunque el fin conseguido sea el mismo.

2.3.1 Método algebraico.

De la observación de las propiedades, postulados y teoremas del álgebra de Boole, se puede apreciar que muchos de los términos de una función lógica pueden ser redundantes en si mismos o en algunas de sus variables, de tal manera que apoyándose en estas cuestiones se puede obtener una expresión minimizada de la función, pero que a la vez hace que se cumpla correctamente. A la aplicación de este sistema se le llama método algebraico, y se pueden ver algunos ejemplos sencillos:

- $F1 = a \cdot b \cdot c + a \cdot b \cdot \bar{c} = a \cdot b \cdot (c + \bar{c})$; aplicando la propiedad distributiva y elemento simétrico.
  $F1 = a \cdot b$
- $F2 = (a + b + c) \cdot (a + b + \bar{c}) = (a + b) \cdot (c + \bar{c})$
  $F2 = a + b$
- $F3 = a \cdot b + a \cdot b \cdot \bar{c}$; aplicando el postulado de la simplificación.
  $F3 = a \cdot b$
- $F4 = a \cdot b + \overline{a \cdot b} \cdot c$; aplicando el teorema numero 3.
  $F4 = a \cdot b + c$

Genéricamente se puede aplicar como método de simplificación el siguiente: siempre que existan dos términos de la misma expresión, el término mayor será redundante. De igual manera en dos términos de la misma expresión la variable complementada del término mayor es redundante.

$$F5 = a \cdot b + a \cdot b \cdot c = a \cdot b$$
$$F6 = a \cdot \bar{b} \cdot c + a \cdot b \cdot c \cdot d = a \cdot \bar{b} \cdot c + a \cdot c \cdot d$$

Para ver un ejemplo más complejo se va a partir de una tabla de verdad de una función cualquiera:

Decimal	a b c d	F
0	0 0 0 0	1
1	0 0 0 1	1
2	0 0 0 1	1
3	0 0 1 1	1
4	0 1 0 0	1
5	0 1 0 1	1
6	0 1 1 0	0
7	0 1 0 1	0
8	1 0 0 0	0
9	1 0 0 1	0
10	1 0 1 0	1

11	1 0 1 1	1
12	1 1 0 0	0
13	1 1 0 1	0
14	1 1 1 0	0
15	1 1 1 1	0

Tabla 2.2

Se procede primero a obtener los Minterm de la función

$$F = \bar{a}\cdot\bar{b}\cdot\bar{c}\cdot\bar{d} + \bar{a}\cdot\bar{b}\cdot\bar{c}\cdot d + \bar{a}\cdot\bar{b}\cdot c\cdot\bar{d} + \bar{a}\cdot\bar{b}\cdot c\cdot d + \bar{a}\cdot b\cdot\bar{c}\cdot\bar{d} + \bar{a}\cdot b\cdot\bar{c}\cdot d + a\cdot\bar{b}\cdot c\cdot\bar{d} + a\cdot\bar{b}\cdot c\cdot d$$

$$F = \bar{a}\cdot\bar{b}\cdot\bar{c} + \bar{a}\cdot\bar{b}\cdot c + \bar{a}\cdot b\cdot\bar{c} + a\cdot\bar{b}\cdot c$$

$$F = \bar{a}\cdot\bar{c} + \bar{b}\cdot c$$

La elección de los términos a la hora de simplificar es importante, ya que cualquier otra combinación conduciría a un resultado no simplificado al máximo. Este es el principal inconveniente de este método, ya que existen varias alternativas a la hora de proceder a eliminar variables redundantes pero no todas llevan a una simplificación máxima al final.

$$F = \bar{a}\cdot\bar{b}\cdot\bar{c} + \bar{a}\cdot\bar{b}\cdot c + \bar{a}\cdot b\cdot\bar{c} + a\cdot\bar{b}\cdot c$$

$$F = \bar{a}\cdot\bar{b} + \bar{a}\cdot b\cdot\bar{c} + a\cdot\bar{b}\cdot c = \bar{a}\cdot(\bar{b} + b\cdot\bar{c}) + a\cdot\bar{b}\cdot c$$

$F = \bar{a}\cdot\bar{c} + a\cdot\bar{b}\cdot c$; se llega a una función no simplificada al máximo.

Y respondería al siguiente circuito de compuertas lógicas:

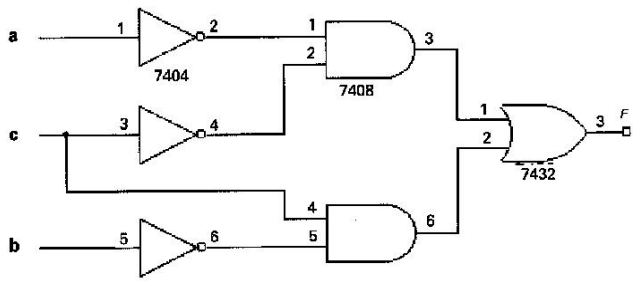


Figura 2.3

2.4 MAPAS DE KARNAUGH.

Entre los sistemas más utilizados para simplificar funciones de 4 variables está el **método de Karnaugh**, el cual por su forma de utilización estructurada hace que se llegue con mayor rapidez y fiabilidad a la función final simplificada. Se basa en la utilización de unas tablas donde quedan reflejados todos los términos de la función. En la figura se pueden observar los mapas para 2, 3 y 4 variables.

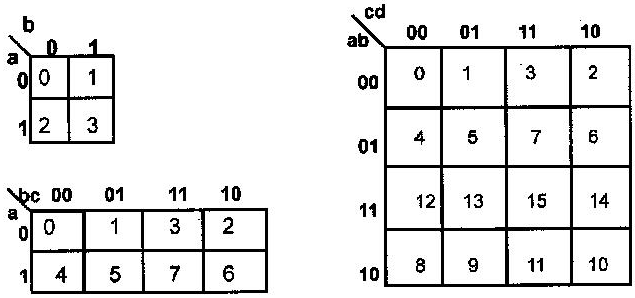


Figura 2.4

El procedimiento que hay que seguir sería:

1. Obtener la función en forma de Minterm.
2. Colocar los unos de la función en el correspondiente mapa de Karnaugh.
3. Formar agrupaciones de 2, 4, 8 ó 16 unos que sean adyacentes (se diferencian en un bit) tanto en el centro, derecha e izquierda o arriba y abajo. Estas agrupaciones deben ser las mayores posibles.
4. Obtener los términos simplificados, de manera que desaparezcan las variables de las agrupaciones realizadas donde aparezcan complementadas y sin complementar.

Se puede observar este método con la misma función utilizada en el método algebraico, con lo que se debe llegar al mismo resultado.

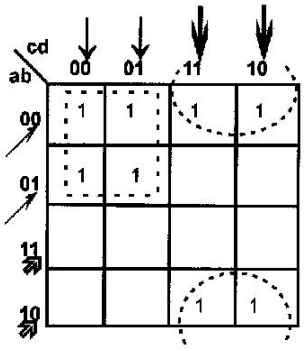


Figura 2.5

$$F = \bar{a} \cdot \bar{c} + \bar{b} \cdot c$$

Se realizan las dos agrupaciones mayores posibles que engloben a todos los unos donde se cumple la función y se obtienen los términos simplificados eliminando las variables que aparecen complementadas y sin complementar en cada una de ellas.
Para simplificar funciones de 5 variables mediante este sistema se recurre a dos mapas de 4 variables donde la variable de mayor peso estaría sin complementar en el primer mapa y complementada en el segundo.

2.4.1 Simplificación de funciones incompletas.

Se dice que una función es incompleta cuando su salida no queda definida para determinados valores de entrada, es decir, que la función puede valer "1" ó "0", e incluso no tener valor definido para estos términos a los cuales se les llama redundantes o inoperantes. Para simplificar estas funciones se pondrá una "X" en el mapa de Karnaugh donde corresponda a uno de estos términos redundantes, y se formarán agrupaciones con ellos sólo si interesa hacerlo.

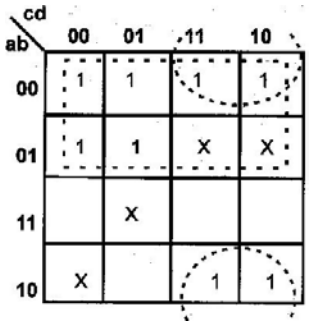


Figura 2.6

$$F = \bar{a} + \bar{b} \cdot c$$

2.4.2 Simplificación de multifunciones.

Cuando un mismo circuito lógico tiene más de una salida, se puede recurrir a un sistema de simplificación conjunto donde se observará que términos idénticos pudieran pertenecer a más de una función, de manera que no se simplificarían y pasarían a pertenecer de forma común a ambas salidas, con lo cual se ahorrarían compuertas lógicas.

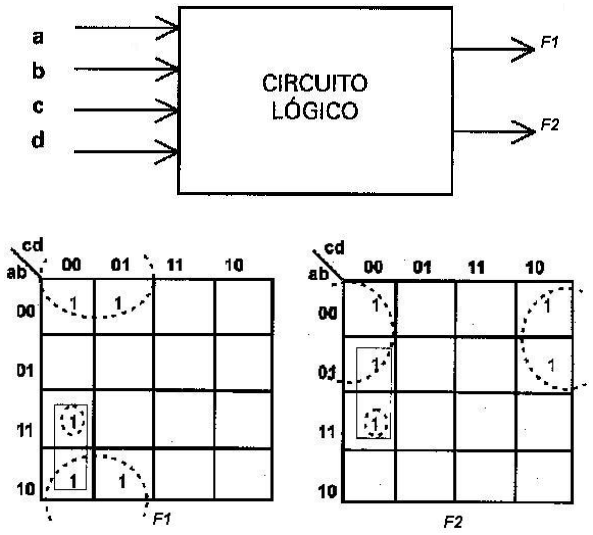


Figura 2.7

$F1 = \bar{c} \cdot \bar{b} + a \cdot b \cdot \bar{c} \cdot \bar{d}$; de esta manera se renuncia al paquete cuadrado donde daría $F1 = \bar{c} \cdot \bar{b} + a \cdot \bar{c} \cdot \bar{d}$

$F2 = \bar{a} \cdot \bar{d} + a \cdot b \cdot \bar{c} \cdot \bar{d}$; renunciando también a la simplificación $F2 = \bar{a} \cdot \bar{d} + b \cdot \bar{c} \cdot \bar{d}$

A simple vista parecería que la segunda opción está más simplificada que la primera, y es cierto, pero a la hora de implementar ambas funciones mediante compuertas lógicas se aprecia que mediante la primera simplificación pueden compartir la compuerta correspondiente al segundo término y por tanto esta sería la forma aconsejable de proceder. Esto se puede apreciar en la Figura 2.8

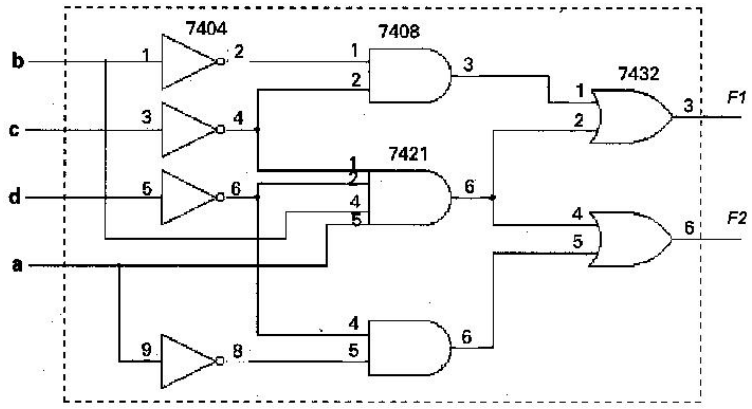


Figura 2.8

III
Aplicación de la Lógica Combinacional

3.1 INTRODUCCIÓN.

Saber en la Teoría (7 hrs.)

La información binaria, además de un álgebra para ser operada, necesita también de un lenguaje para ser interpretada. Este es el formado por los códigos binarios y los BCD, pero para que dicha información tenga sentido se necesita de unos circuitos digitales capaces de agruparla bajo la estructura de un código o partiendo de una información codificada desagrupada para poder operar con ella. A estos circuitos se les llama codificadores y decodificadores y se encuentran comercializados en circuitos integrados.

3.2 CONTROL DE ENTRADA/SALIDA Y CONVERSIÓN DE CÓDIGO.

3.2.1 Codificadores.

Son circuitos lógicos Combinacionales (su respuesta final no depende del tiempo ni de estados anteriores) que disponen de 2^n líneas de entrada sin codificar y proporcionan a su salida un código de **n** líneas. Generalmente siempre existirá una sola entrada activa que generara su correspondiente código de salida. Su principal aplicación es la de pasar líneas de señales sin orden aparente a códigos binarios para posteriormente operar con ellos.

a b c d	$F1$	$F2$
0 0 0 1	0	0
0 0 1 0	0	1
0 1 0 0	1	0
1 0 0 0	1	1

Tabla 3. 1

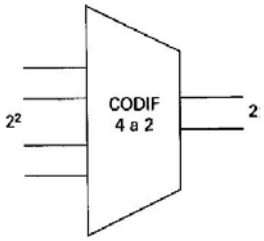


Figura 3. 1

$$F1 = \overline{a} \cdot \overline{b} \cdot c \cdot \overline{d} + a \cdot \overline{b} \cdot \overline{c} \cdot \overline{d} = \overline{b} \cdot \overline{d} \cdot (a \oplus c)$$

$$F2 = \overline{a} \cdot \overline{b} \cdot \overline{c} \cdot \overline{d} + a \cdot \overline{b} \cdot \overline{c} \cdot \overline{d} = \overline{c} \cdot \overline{d} \cdot (a \oplus b)$$

La implementación del circuito la podemos realizar de dos maneras, compartiendo el segundo termino en forma de multifunción (Figura 3.2a), o según el resultado final (Figura 3.2b), siendo esta la que interesa mas al utilizar el mismo numero de compuertas para economizar en integrados.

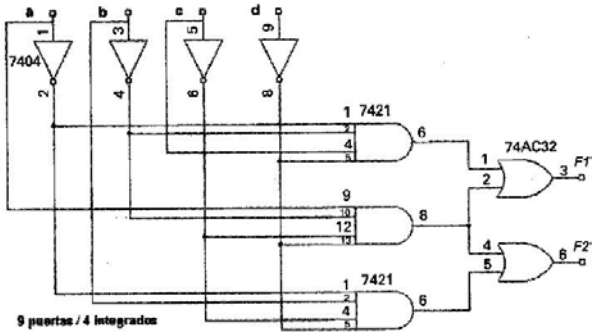


Figura 3. 2a

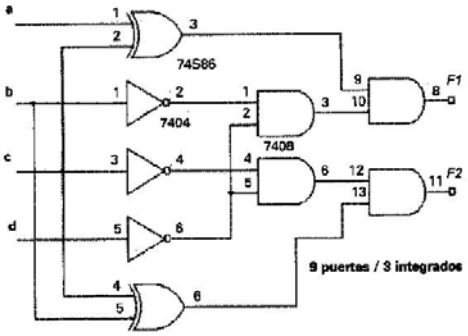


Figura 3. 2b

La codificación se puede realizar atendiendo a una prioridad de las señales de entrada, es decir, que si se activa más de una entrada a la vez la salida corresponderá al código de la de menor peso. Además, también pueden disponer de una entrada de habilitación como se puede observar en la tabla 3.2 y el circuito de compuertas de la Figura 3.3.

I	a b c d	$F1$	$F2$
0	x x x 1	0	0
0	x x 1 0	0	1
0	x 1 0 0	1	0
0	1 0 0 0	1	1
1	x x x x	0	0

Tabla 3. 2

$$F1 = c \cdot \overline{d} + a \cdot \overline{b} \cdot \overline{c} \cdot \overline{d}$$
$$F2 = b \cdot \overline{c} \cdot \overline{d} + a \cdot \overline{b} \cdot \overline{c} \cdot \overline{d}$$

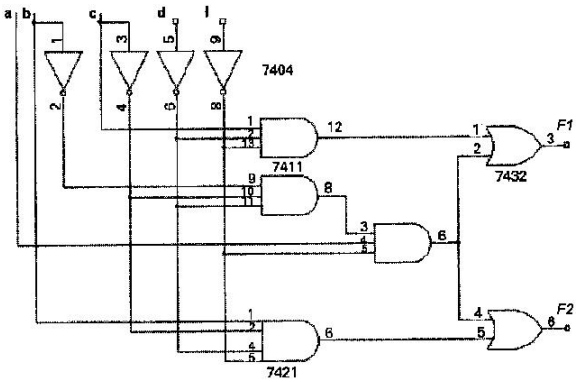


Figura 3. 3

3.2.2 Codificador decimal-BCD (74147).

Estos circuitos se encuentran integrados en chip, encontrándose entre los más importantes el 74147, que es un codificador de decimal a BCD con prioridad (de 10 a 4 líneas) y que funcionan con lógica negativa. La tabla 3.3 muestra su funcionamiento:

1	2	3	4	5	6	7	8	9	D	C	B	A
1	1	1	1	1	1	1	1	1	1	1	1	1
x	x	x	x	x	x	x	x	0	0	1	1	0
x	x	x	x	x	x	x	0	1	0	1	1	1
x	x	x	x	x	x	0	1	1	1	0	0	0
x	x	x	x	x	0	1	1	1	1	0	0	1
x	x	x	x	0	1	1	1	1	1	0	1	0
x	x	x	0	1	1	1	1	1	1	0	1	1
x	x	0	1	1	1	1	1	1	1	1	0	0
x	0	1	1	1	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	1	0

Tabla 3. 3

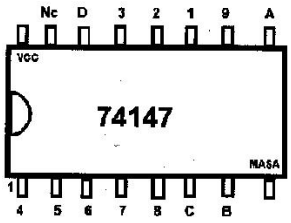


Figura 3. 4

3.3 Decodificadores.

Son circuitos combinacionales que disponen de n líneas de entrada, generalmente correspondientes al código binario, y proporcionan 2^n líneas de salida, de las cuales sólo habrá una en activo. Es por tanto el circuito que realiza la función contraria del codificador, de modo que permite deshacer un código generando líneas de señales que pueden ser utilizadas para activar o desactivar otros dispositivos.

I	a b	$F4\ F3\ F2\ F1$
0	0 0	0 0 0 1
0	0 1	0 0 1 0
0	1 0	0 1 0 0
0	1 1	1 0 0 0
1	x x	0 0 0 0

Tabla 3. 4

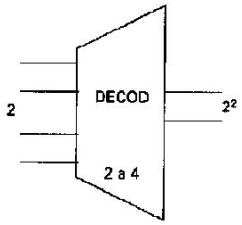


Figura 3. 5

$F1 = \bar{a} \cdot \bar{b}$
$F2 = \bar{a} \cdot b$
$F3 = a \cdot \bar{b}$
$F4 = a \cdot b$

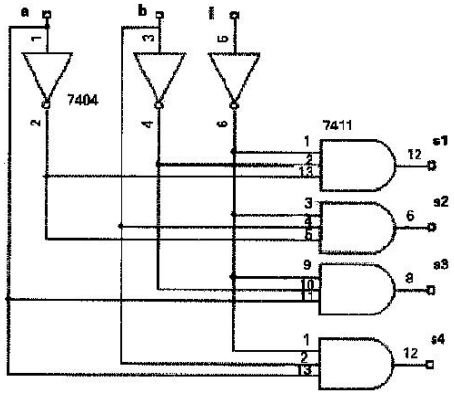


Figura 3. 6

Otra de las aplicaciones importantes de los decodificadores es la actuar como generador de funciones asociado a una compuerta OR, de tal manera que a la salida habrá un "1" siempre que se cumpla la función al introducir cada uno de los códigos que la hacen valida:

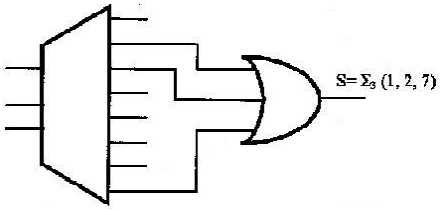


Figura 3. 7

En la Figura 3.7 se observa que la salida F corresponde a los minterm 1,2 y 7, que son aquellos donde la función vale uno.

3.3.1 Decodificador BCD-Decimal (7441).

Este decodificador, que se encuentra integrado en el circuito TTL 7441, funciona con lógica negativa de salida y no viene provisto de entrada de habilitación. La tabla 5.5 muestra su funcionamiento:

D	C	B	A	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1	1	1
0	1	1	0	1	1	1	1	1	1	0	1	1	1
0	1	1	1	1	1	1	1	1	1	1	0	1	1
1	0	0	0	1	1	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0
1	0	1	0	1	1	0	1	1	1	1	1	0	1
1	0	1	1	1	1	1	0	1	1	1	1	1	0
1	1	0	0	1	1	1	1	0	1	1	1	1	1
1	1	0	1	1	1	1	1	1	0	1	1	1	1
1	1	1	0	1	1	1	1	1	1	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	0	1	1

Tabla 3. 5

Las salidas a partir de la decodificación del 10 en binario no tienen sentido, ya que en BCD sólo existen codificaciones del 0 al 9.

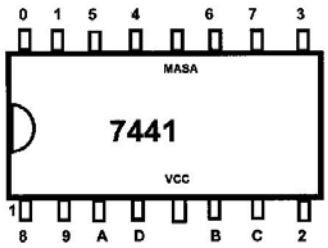


Figura 3. 8

3.3.2 Decodificador BCD-7 Segmentos (7447).

No todos los decodificadores responden a la definición dada anteriormente, existen algunos de uso especial donde las líneas de salida responden al código de entrada activándose mas de una a la vez, obedeciendo por tanto a la utilidad a la que han sido diseñados. Este es el caso del decodificador BCD a 7 segmentos, cuya utilidad es la atacar a un display de 7 segmentos presentando el digito en decimal correspondiente al código BCD de la entrada. La Tabla 3.6 muestra su funcionamiento:

No. dec	$\overline{LT}$	$\overline{RBI}$	D	C	B	A	$\overline{BI}$ / $\overline{RBO}$	a	b	c	d	e	f	g
0	1	1	0	0	0	0	1	0	0	0	0	0	0	1
1	1	x	0	0	0	1	1	1	0	0	1	1	1	1
2	1	x	0	0	1	0	1	0	0	1	0	0	1	0
3	1	x	0	0	1	1	1	0	0	0	0	1	1	0
4	1	x	0	1	0	0	1	1	0	0	1	1	0	0
5	1	x	0	1	0	1	1	0	1	0	0	1	0	0
6	1	x	0	1	1	0	1	1	1	0	0	0	0	0
7	1	x	0	1	1	1	1	0	0	0	1	1	1	1
8	1	x	1	0	0	0	1	0	0	0	0	0	0	0
9	1	x	1	0	0	1	1	0	0	0	1	1	0	0
$\overline{BI}$	x	x	x	x	x	x	0	1	1	1	1	1	1	1
$\overline{RBI}$	1	0	0	0	0	0	0	1	1	1	1	1	1	1
$\overline{LT}$	0	x	x	x	x	x	1	0	0	0	0	0	0	0

Tabla 3. 6

Observaciones:

- El funcionamiento de salida es por lógica negativa, por tanto trabajara con displays de ánodo común, según se puede ver en la Figura 3.10 (el 7448 funciona por lógica positiva para los de cátodo común).
- Las codificaciones del 10 al 15 no dan a la salida nada que tenga sentido.
- $\overline{BI}$ es la entrada de blanking que puesta a cero sirve para apagar el display (salidas a nivel alto).
- $\overline{LT}$ es la entrada de prueba (lamptest), de manera que al llevarla a cero y estando $\overline{BI}$, fuerza todas las salidas a nivel bajo con lo que todos los segmentos del display se encenderán.

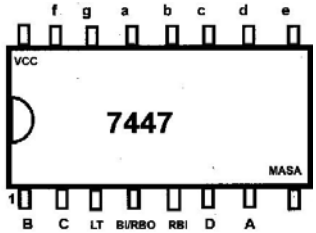


Figura 3. 9

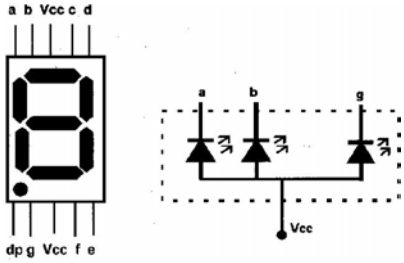


Figura 3. 10

> ➢ *El patillaje de los displays se puede conocer con el multímetro en el modo de continuidad, si el display es de ánodo común entonces la punta roja se conectará al ánodo y la punta negra a cada uno de los pines, de forma que cada segmento se encenderá al polarizarlo correctamente mediante la batería del multímetro (si el display es de cátodo común se intercambian las puntas y se realiza el mismo procedimiento).*

3.4 Multiplexores y Demultiplexores.

La transmisión de señales se puede realizar mediante una o varias líneas, en especial cuando se trata de información codificada en digital. El sistema elegido dependerá fundamentalmente del factor económico y de la distancia a la cual se deben transmitir las señales, recurriéndose por tanto a la línea única cuando se trata de economizar en largas distancias y buscando la rapidez de trabajar con varias líneas de información en cortas distancias.

3.4.1 Multiplexores.

Son circuitos cuya prestación es la de poder gobernar 2^n líneas de información de entrada mediante **n** líneas de selección, de tal manera que serán canalizadas hacia una única salida. En esencia, funcionan como si de un conmutador eléctrico se tratase, según se puede observar en la Figura 3.11:

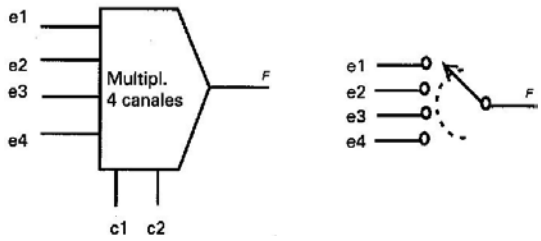


Figura 3. 11

Entradas de selección		Salida
c1	c2	F
0	0	e1
0	1	e2
1	0	e3
1	1	e4

Tabla 3. 7

De la observación de la tabla de funcionamiento podemos sacar directamente el circuito de compuertas lógicas:

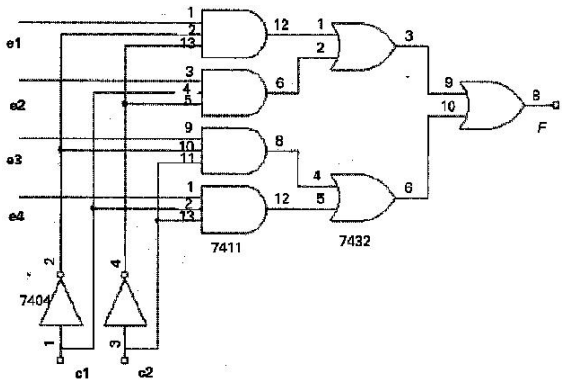


Figura 3. 12

3.4.1.1 Multiplexor selector de datos de 8 a 1 (74151).

Este circuito multiplexor se encuentra integrado dentro de la familia TTL en el chip 74151, dispone de tres líneas de selección para controlar 8 entradas y también esta provisto de una entrada de habilitación (S) llamada **strobe**. Su salida esta disponible también complementada.

Selección				Información								Salida	
C	B	A	S	D0	D1	D2	D3	D4	D5	D6	D7	Y	W
x	x	X	1	x	x	x	x	x	x	x	x	0	1
0	0	0	0	0	x	x	x	x	x	x	x	0	1
0	0	0	0	1	x	x	x	x	x	x	x	1	0
0	0	1	0	x	0	x	x	x	x	x	x	0	1
0	0	1	0	x	1	x	x	x	x	x	x	1	0
0	1	0	0	x	x	0	x	x	x	x	x	0	1
0	1	0	0	x	x	1	x	x	x	x	x	1	0
0	1	1	0	x	x	x	0	x	x	x	x	0	1
0	1	1	0	x	x	x	1	x	x	x	x	1	0
1	0	0	0	x	x	x	x	0	x	x	x	0	1
1	0	0	0	x	x	x	x	1	x	x	x	1	0
1	0	1	0	x	x	x	x	x	0	x	x	0	1
1	0	1	0	x	x	x	x	x	1	x	x	1	0
1	1	0	0	x	x	x	x	x	x	0	x	0	1
1	1	0	0	x	x	x	x	x	x	1	x	1	0
1	1	1	0	x	x	x	x	x	x	x	0	0	1
1	1	1	0	x	x	x	x	x	x	x	1	1	0

Tabla 3. 8

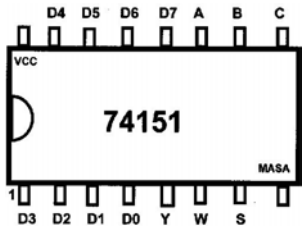


Figura 3. 13

3.4.1.2 Multiplexor doble de 4 a 1 (74153).

Este es un circuito multiplexor dual de 4 entradas, es decir, lo componen dos circuitos multiplexores de 4 a 1 que disponen además de dos entradas independientes de habilitación (1G, 2G) y entrada común de selección. Las salidas están amplificadas, pudiendo atacar hasta 10 unidades lógicas.

Selección			Información				Salida
B	A	G	D0	D1	D2	D3	Y
x	x	1	x	x	x	x	1
0	0	0	0	x	x	x	0
0	0	0	1	x	x	x	1
0	1	0	x	0	x	x	0
0	1	0	x	1	x	x	1
1	0	0	x	x	0	x	0
1	0	0	x	x	1	x	1
1	1	0	x	x	x	0	0
1	1	0	x	x	x	1	1

Tabla 3. 9

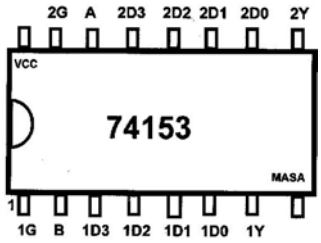


Figura 3. 14

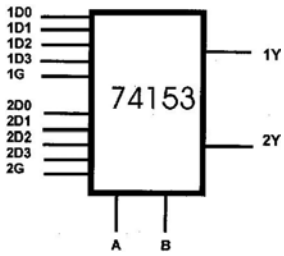


Figura 3. 15

3.4.2 Demultiplexores.

Son circuitos cuya prestación es la de poder transmitir una única entrada de información hacia 2^n líneas de selección, según se puede observar en la Figura 3.16:

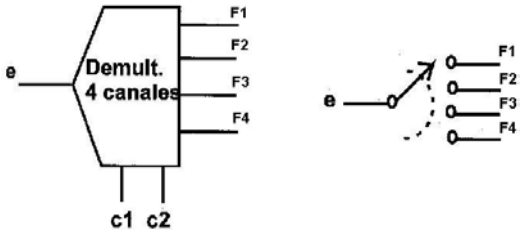


Figura 3. 16

Entradas de selec.		Salidas			
c1	c2	F1	F2	F3	F4
0	0	e	0	0	0

0	1	0	e	0	0
1	0	0	0	e	0
1	1	0	0	0	E

Tabla 3. 10

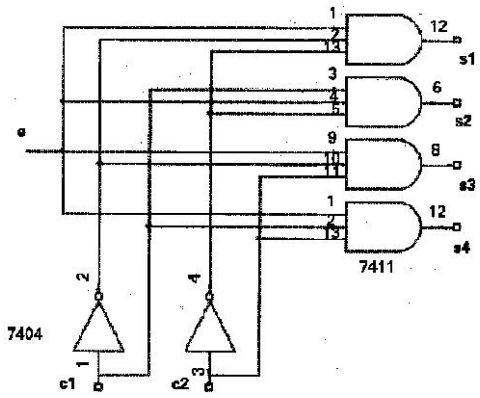


Figura 3. 17

IV
Circuitos Secuénciales

4.1 INTRODUCCIÓN A LOS CIRCUITOS SECUENCIALES.

Saber en la Teoría (10 hrs.)

Hasta el momento se han estudiado circuitos pertenecientes a la lógica combinacional, es decir, aquéllos, cuya salida depende única y exclusivamente del valor de las entradas, pero también existen otros circuitos que pertenecen a la llamada lógica secuencial, y son aquéllos cuya salida no sólo depende del valor de las entradas sino también de la historia del circuito, esto es, de los valores y estados que tuviera anteriormente y que han quedado almacenados. Los Circuitos secuenciales básicos son los Flip-Flop's o biestables.

4.2 BIESTABLES.

El Flip-Flop es un multivibrador biestable, que constituye a su vez el circuito secuencial básico, capaz de almacenar a modo de memoria la unidad de información básica (bit).

Los biestables o Flip-Flop respecto de su funcionamiento se pueden clasificar en síncronos y asíncronos:

Síncronos: son aquellos que necesitan de una base de tiempos (reloj) para ser activados. La activación puede producirse bien por el nivel lógico ("0","1") o bien por el flanco de la señal de sincronismo (subida ↥, bajada ↧).
Asíncronos: son las que se activan y funcionan sin necesidad de una base de tiempos.

4.3 FLIP-FLOP TIPO RS.

Los Flip-Flop, aunque se comercializan integrados, están formados por compuertas lógicas conectados de forma que se establece una realimentación de la salida hacia la entrada dando como resultado circuitos con dos estados estables. Estos circuitos, como ya se ha mencionado, tendrán la particularidad de evolucionar en función de sus entradas y teniendo en cuenta el estado en que se encuentra, proporcionando una salida y su complementaria.

El Flip-Flop RS es el más sencillo y recibe su nombre de sus entradas, **Set** (puesto a uno) y **Reset** (puesto a cero). Seguidamente se puede observar su tabla de funcionamiento y circuito eléctrico.

R	S	Q^t	Q^{t+1}
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	x
1	1	1	x

- Las entradas a cero no producen variación del estado del Flip-Flop Q^2.
- Un uno por ambas entradas creará una situación de indeterminación.

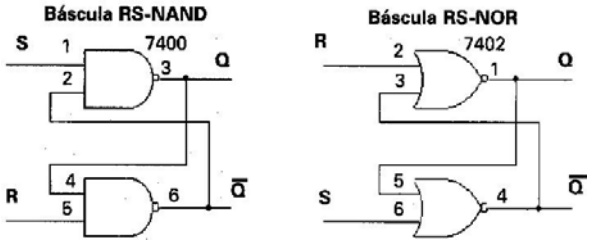


Figura 4.1

Funcionamiento RS-NOR:

- Partiendo de un estado inicial $Q=0$, $\overline{Q}=1$, $R=1$, $R=0$.
- Para salir se provoca que $S=1$ y $R=0$, con lo cual $\overline{Q}=0$ y $Q=1$.
- El cambio $S=0$ y $R=1$, volvería al primer estado.
- La báscula RS-NAND funcionaría por lógica negativa, es decir, que un cero por Set pone la báscula a "1".
- La versión síncrona de un Flip-Flop RS se obtiene agregando una entrada de reloj que funcione conjuntamente con las entradas de control, según se muestra en la Figura 4.2.

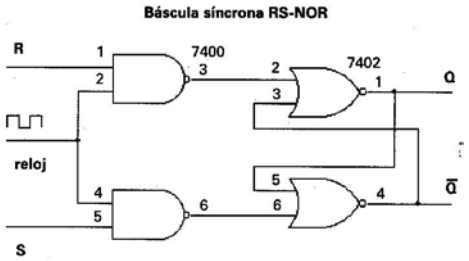


Figura 4.2

Se puede establecer el cronograma de tiempos de la Figura 4.3 para el estudio de su funcionamiento, donde se aprecia que la puesta a uno o la puesta a cero no tienen efecto hasta la llegada de la señal de reloj.

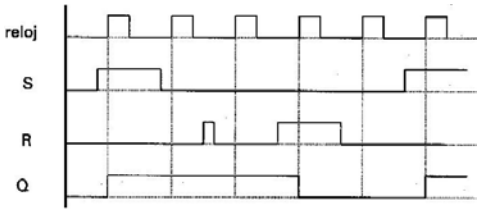


Figura 4.3

Los Flip-Flop RS comerciales se pueden encontrar en los integrados: 74118, 74119, 74279, 4043, 4044.

4.4 FLIP-FLOP TIPO JK.

El uso de los Flip-Flop RS se ve sujeto a evitar pasar por los estados en que la salida es indeterminada. Como resultado de evitar la posibilidad de que existan estos estados surge el Flip-Flop tipo JK, que es en realidad un RS realimentado, según puede observarse en la Figura 4.4.

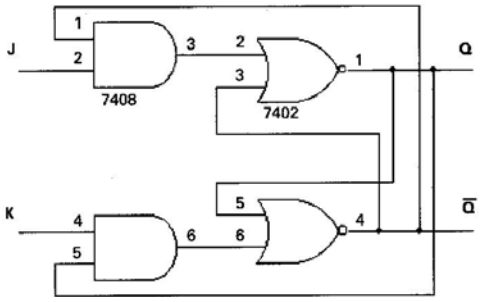


Figura 4.4

La tabla de funcionamiento muestra cómo desaparecen los estados indeterminados, de manera que en todo momento se puede saber cómo es la evolución de la salida. J seria equivalente a Set y K a Reset, así cuando ambos están a uno se convierte en un verdadero Flip-Flop, ya que la salida cambia al estado negado al que tiene, de manera que un Flip-Flop JK con las entradas unidas se convierte en un Flip-Flop tipo T, según se muestra en la Figura 4.5.

K	J	Q^{t+1}
0	0	Q^t
0	1	1
1	0	0
1	1	$\overline{Q^t}$

Tabla 4.2

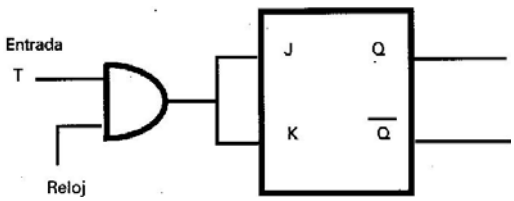


Figura 4.5

La solución síncrona de este Flip-Flop se obtendría añadiendo dos compuertas AND a las entradas J y K junto con la señal de reloj. Los Flip-Flop JK comerciales se pueden encontrar en los integrados: 7470 y 4027.

4.5 FLIP-FLOP TIPO D.

Otra variante de los Flip-Flop JK lo formarían los llamados Flip-Flop tipo D (también se podrían realizar a partir de RS), que son JK síncronos con una única entrada, aunque en realidad la entrada se lleva tal cual a J y negada a K. en esta situación el Flip-Flop no cambiará hasta que llegue la señal de reloj, por lo que almacenan la información existente hasta ese instante. Esto es lo que hace que se las llame match ó células de memoria. La tabla 4.3 y la Figura 4.6 muestran lo dicho.

D	reloj	Q^{t+1}
0	↑	0
1	↑	1
x	0	Q^t

Tabla 4.3

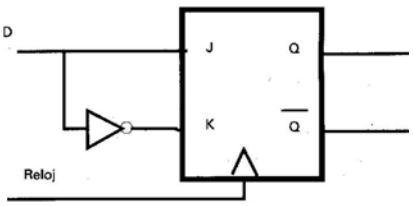


Figura 4.6

4.6 FLIP-FLOP MASTER-SLAVE.

Los Flip-Flop master-slave (maestro-esclavo) también llamados principal-secundario. Están formados por dos RS, JK o D síncronos en los que se ha creado un retardo de la señal de reloj entre el primero y el segundo, de tal manera que el primero evolucionará con el flanco de subida y el segundo con el flanco de bajada. La Tabla 4.4 y las Figuras 4.7 y 4.8 muestran su funcionamiento y esquema eléctrico mediante compuertas.

J	K	reloj	Q^{t+1}
0	0	↓	Q^t
0	1	↓	1
1	0	↓	0
1	1	↓	$\overline{Q^t}$
X	X	0	Q^t

Tabla 4.4

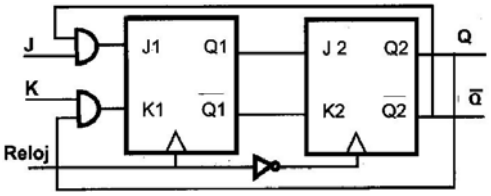


Figura 4.7

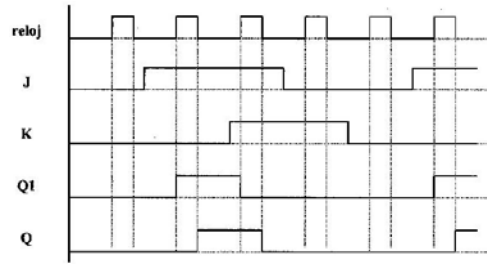


Figura 4.8

Comercialmente se encuentran en los integrados: 7472, 7473, 7476, 74104, 74105, 74107, 74109, 74110, 74111, 74115, 74135, 74176, 100151, etc.

4.7 FLIP-FLOP COMERCIALES.

Seguidamente, en las Figuras 4.9 a 4.13 se pueden ver algunos de los integrados comerciales de Flip-Flop más usuales.

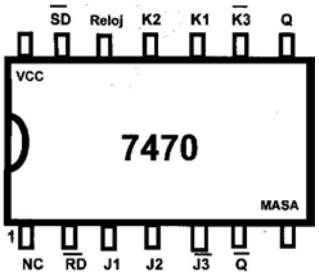


Figura 4.9

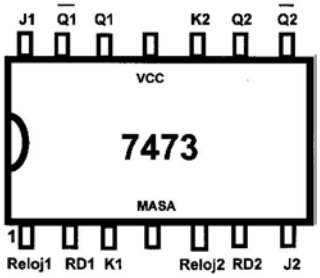


Figura 4.10

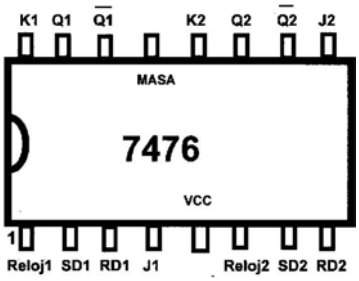


Figura 4.11

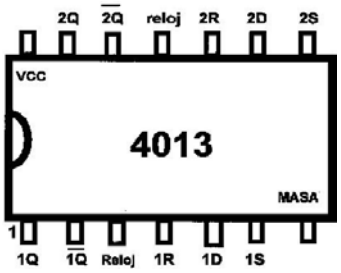


Figura 4.12

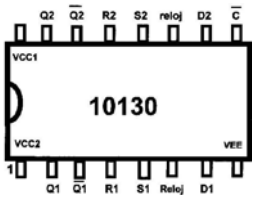


Figura 4.13

4.8 CONTADORES.

4.8.1 Introducción a los contadores.

Al igual que ocurre con los registros, los contadores son circuitos secuenciales que están formados por la unión de varios Flip-Flop's, pero con funciones bastantes diferentes. Básicamente lo que hacen es contar pulsos de la señal de reloj de forma encadenada, de manera que dependiendo de la conexión interna de sus Flip-Flop's tendrán una función u otra. Entre las aplicaciones más destacadas están la de contadores de códigos binarios y de décadas, divisiones de frecuencia y medidores de tiempos.

Son circuitos lógicos secuenciales compuestos por Flip-Flop's y que funcionan gobernados por una señal de reloj. Estos Flip-Flop's pueden pasar por una serie de estados sucesivos, en los que cada uno dependerá del estado anterior y de la forma de funcionamiento del circuito contador. Los contadores admiten varias clasificaciones:

Respecto al funcionamiento.
a) Asíncronos.
b) Síncronos.

Respecto al código.
a) Binarios.
b) Décadas.

Respecto de las prestaciones.
a) Ascendentes y descendentes.
b) Programables.

4.8.2 Contadores según su funcionamiento.

4.8.2.1 Contadores Asíncronos.

Son aquéllos en los que los pulsos de reloj sólo atacan al primer Flip-Flop del contador, de manera que la entrada de reloj de las siguientes será la salida Q del Flip-Flop anterior. Se dice que la propagación es encadenada, por tanto se sufrirá un retardo de dos ciclos de reloj por cada una de los Flip-Flop's. La Figura 4.14 y 4.15 muestran la estructura de estos contadores y su cronograma de funcionamiento.

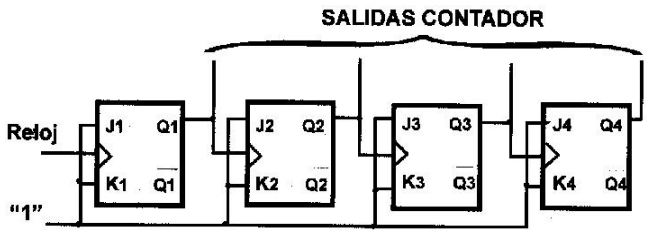


Figura 4.14

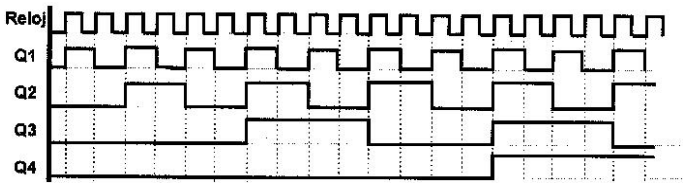


Figura 4.15

En el esquema eléctrico se puede apreciar que con la llegada de una transición de uno a cero (podría ser también de cero a uno) de una señal por la entrada de reloj la salida Flip-Flop correspondiente cambiará al estado contrario al que se encontraba. Si se observa el cronograma de tiempos se verá que esto es así y por tanto la salida de cada Flip-Flop necesitará del cambio de la anterior para hacerlo también, creándose así un retardo que para el último Flip-Flop será de 8 ciclos de reloj respecto de la anterior. Este tiempo que se necesita para la propagación condicionará la frecuencia máxima de reloj a la que puede trabajar el contador.

4.8.2.2 Contadores Síncronos.

Son aquéllos en que la señal de entrada de reloj produce el cambio a la vez en todos los Flip-Flop que están preparados para hacerlo, esto es, los Flip-Flop cambiarán de estado sólo si las que le preceden se encuentran ya a uno. Se verá con más claridad si se analiza tanto la estructura como el cronograma de funcionamiento. Figuras 4.16 y 4.17.

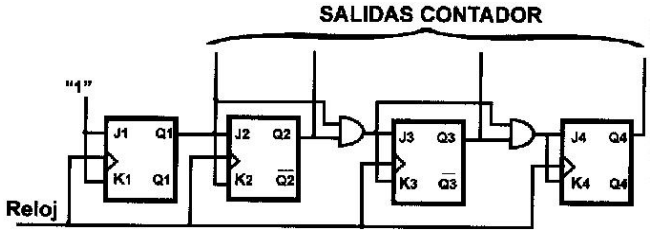


Figura 4.16

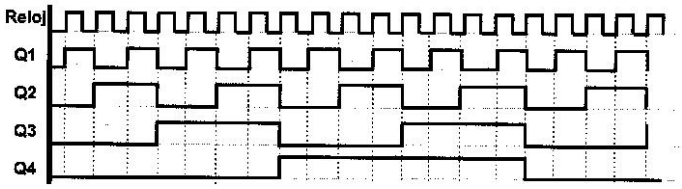


Figura 4.17

Se observa que el primer Flip-Flop cambia con los pulsos de la señal de reloj, pero las siguientes para cambiar deben recibir un uno de las precedentes además del pulso de reloj que, como se ve, llega a todas a la vez. En este caso existe la ventaja de no tener tiempo de propagación; si acaso se pierde una fracción mínima en la respuesta de las puertas AND. En consecuencia, las velocidades de trabajo serán mayores que en los contadores asíncronos.

4.9 MOTORES A PASOS.

Sus principales aplicaciones se pueden encontrar en robótica, tecnología aeroespacial, control de discos duros, flexibles, unidades de CD-ROM o de DVD e impresoras, en sistemas informáticos, manipulación y posicionamiento de herramientas y piezas en general. Desde el punto de vista de su construcción existen 3 tipos de motores paso a paso:

De imán permanente: es el modelo que hemos analizado anteriormente. El rotor es un imán permanente en el que se mecanizan un número de dientes limitado por su estructura física. Ofrece como principal ventaja que su posicionamiento no varía aún sin excitación y en régimen de carga.
De reluctancia variable: Los motores de este tipo poseen un rotor de hierro dulce que en condiciones de excitación del estator y bajo la acción de su campo magnético, ofrecen menor resistencia a ser atravesado por su flujo en la posición de equilibrio. Su mecanización es similar a los de imán permanente y su principal inconveniente radica en que en condiciones de reposo (sin excitación) el rotor queda en libertad de girar y, por lo tanto, su posicionamiento de régimen de carga dependerá de su inercia y no será posible predecir el punto exacto de reposo.
Híbridos: Son combinación de los dos tipos anteriores; el rotor suele estar constituido por anillos de acero dulce dentado en un número ligeramente distinto al del estator y dichos anillos montados sobre un imán permanente dispuesto axialmente.

Motores Unipolares: En este tipo de motores, todas las bobinas del estator están conectadas en serie formando cuatro grupos. Estos, a su vez, se conectan dos a dos, también en serie, y se montan sobre dos estatores diferentes, tal y como se aprecia en la Figura 4. Según puede apreciarse en dicha figura, del motor paso a paso salen dos grupos de tres cables, uno de los cuales es común a dos bobinados. Los seis terminales que parten del motor, deben ser conectados al circuito de control, el cual, se comporta como cuatro conmutadores electrónicos que, al ser activados o desactivados, producen la alimentación de los cuatro grupos de bobinas con que está formado el estator. Si generamos una secuencia adecuada de funcionamiento de estos interruptores, se pueden producir saltos de un paso en el número y sentido que se desee.

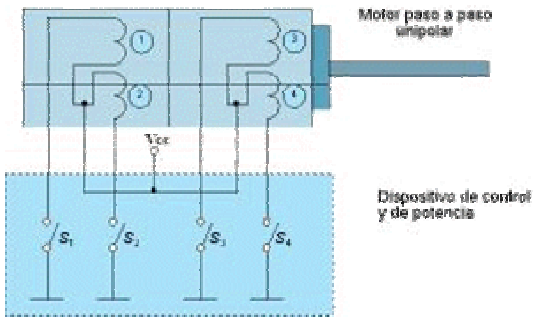


Figura 4.18 Control de motor Unipolar.

Motores Bipolares: En este tipo de motores las bobinas del estator se conectan en serie formando solamente dos grupos, que se montan sobre dos estatores, tal y como se muestra en

la Figura 4.23. Según se observa en el esquema de este motor salen cuatro hilos que se conectan, al circuito de control, que realiza la función de cuatro interruptores electrónicos dobles, que nos permiten variar la polaridad de la alimentación de las bobinas. Con la activación y desactivación adecuada de dichos interruptores dobles, podemos obtener las secuencias adecuadas para que el motor pueda girar en un sentido o en otro.

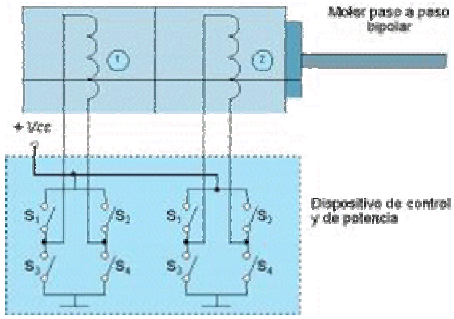


Figura 4.19 Control de motor Bipolar.

La existencia de varios bobinados en el estator de los motores de imán permanente, da lugar a varias formas de agrupar dichos bobinados, para que sean alimentados adecuadamente. Estas formas de conexión permiten clasificar los motores paso a paso en dos grandes grupos:

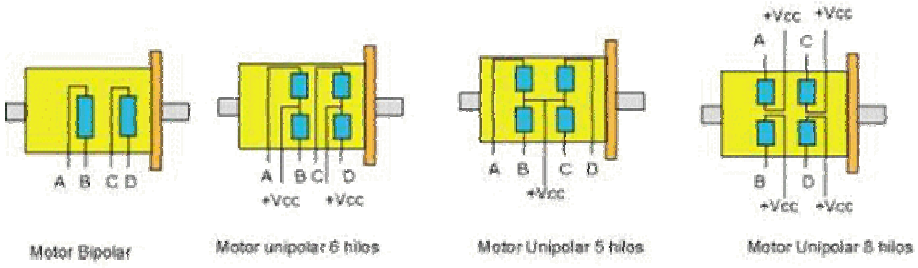


Figura 4.20 Disposición de las bobinas de motores paso a paso

a) bipolar b) unipolar con 6 hilos c) unipolar a 5 hilos d) unipolar a 8 hilos.

Hay que tener en cuenta que los motores unipolares de seis u ocho hilos, pueden hacerse funcionar como motores bipolares si no se utilizan las tomas centrales, mientras que los de cinco hilos no podrán usarse jamás como bipolares, porque en el interior están conectados los dos cables centrales.

En el caso de los unipolares lo normal es encontrarnos con cinco, seis u ocho terminales, ya que además de los bobinados hay otros terminales que corresponden con a las tomas intermedias de las bobinas, los cuales se conectan directamente a positivo de la fuente de alimentación para su correcto funcionamiento. En la figura 3b, 3c y 3d pueden apreciar como están conectados internamente los terminales de estos tipos de motores.

Secuencia del circuito de control.
Existen dos formas básicas de hacer funcional los motores paso a paso atendiendo al avance del rotor bajo cada impulso de excitación:
Paso completo (*full step*): El rotor avanza un paso completo por cada pulso de excitación y para ello su secuencia ha de ser la correspondiente a la expuesta anteriormente, para un motor como el de la Figura 2, y que se presentada de forma resumida en la Tabla 1 para ambos sentidos de giro, las X indican los interruptores que deben estar cerrados (interruptores en ON), mientras que la ausencia de X indica interruptor abierto (interruptores en OFF).

Paso	S1	S2	S3	S4
1	X			X
2			X	X
3		X	X	
4	X	X		
1	X			X
Sentido horario (a)				

Paso	S1	S2	S3	S4
1	X	X		
2		X	X	
3			X	X
4	X			X
1	X	X		
Sentido antihorario (b)				

Tabla 1.- Secuencia de excitación de un motor paso a paso completo

Medio paso (*Half step*): Con este modo de funcionamiento el rotor avanza medio paso por cada pulso de excitación, presentando como principal ventaja una mayor resolución de paso, ya que disminuye el avance angular (la mitad que en el modo de paso completo). Para conseguir tal cometido, el modo de excitación consiste en hacerlo alternativamente sobre dos bobinas y sobre una sola de ellas, según se muestra en la Tabla 2 para ambos sentidos de giro.

Paso	Excitación de Bobinas			
	S1	S2	S3	S4
1	X			X
2				X
3			X	X
4			X	
5		X	X	
6		X		
7	X	X		
8	X			
1	X			X
Sentido horario (a)				

Paso	Excitación de Bobinas			
	S1	S2	S3	S4
1	X	X		
2		X		
3		X	X	
4			X	
5			X	X
6				X
7	X			X
8	X			
1	X	X		
Sentido antihorario (b)				

Tabla 2.- Secuencia de excitación de un motor Paso a Paso en medio paso

Según la Figura 2 al excitar dos bobinas consecutivas del estator simultáneamente, el rotor se alínea con la bisectriz de ambos campos magnéticos; cuando desaparece la excitación de una de ellas, extinguiéndose el campo magnético inducido por dicha bobina, el rotor queda bajo la acción del único campo existente, dando lugar a un medio desplazamiento.
Sigamos, por ejemplo, la secuencia presentada en la Tabla 2 : en el paso 1, y excitadas las bobinas L1 y L2 de la Figura 2 mediante la acción de S1 y S2, el rotor se situaría en la posición indicada en la Figura 2 a; en el paso 2, S1 se abre, con lo que solamente permanece excitada L2 y el rotor girará hasta alinear su polo sur con el norte generado por L2. Suponiendo que este motor tenía un paso de 90 grados, en este caso sólo ha avanzado 45 grados. Posteriormente, y en el paso 3, se cierra S3, situación representada en la Figura 2 b, con lo que el rotor ha vuelto a avanzar otros 45 grados. En definitiva, los desplazamientos, siguiendo dicha secuencia, son de medio paso.

4.10 TIPOS DE MEMORIAS.

Las memorias empleadas en los sistemas electrónicos microprocesadores se pueden clasificar en dos grandes grupos:

1. Memorias RAM (Ramdom Acces Memory), son memorias de acceso aleatorio, pero la característica que más las define es que son volátiles. En estas memorias se puede realizar tanto escrituras como lecturas.
   - RAM estáticas ó SRAM. Cada una de las celdas están construidas mediante Flip-Flop's, que permanecen indefinidamente en un estado, mientras no se elimine la alimentación.
   - RAM dinámicas o DRAM. Cada una de las celdas están construidas por condensadores que almacenan la información mediante la carga o descarga de estos. Los condensadores pierden cargas y esto obliga a realizar continuas reescrituras de la misma información para evitar perder la información. A este proceso se le conoce como "refresco" (refresh).
2. Memorias ROM (Read Only Memory), son memorias de acceso aleatorio como las RAM, pero la principal característica por la que son identificadas es que no pierden los datos aunque no tengan aplicada la tensión de alimentación, por tanto son memorias no volátiles. Estas memorias sólo permiten la lectura, ya que para la escritura se requieren unos procesos más complejos (hay algunos casos en que sí es permitida la escritura). Dependiendo del proceso de grabación, las ROM se clasifican en:
   - ROM, sin ninguna sigla más. Se entiende que son memorias programables por máscara, esto es, los datos se graban durante el proceso de grabación. Una de las máscaras utilizadas es el proceso fotográfico se diseña para producir en la memoria los datos deseados en las posiciones necesarias. Para cada conjunto de información se requiere un tipo de máscara distinta. Ya que las máscaras son muy costosas, este tipo de ROM sólo es rentable cuando se realizan grandes series de un tipo de máscara.
   - PROM, son memorias ROM programables. Dichas memorias se pueden grabar por el usuario final, pero una sola vez. El método usado es el de fundir fusibles para provocar el valor del dato deseado, por tanto es una grabación irreversible. También se hace referencia a estas memorias como ROM "programables una sola vez". Con este tipo de memoria si que es rentable el realizar series de cantidades pequeñas.
   - EPROM, Son memorias ROM borrables y programables. Son similares a las PROM, pero el proceso de grabación no es tan destructivo, pudiendo borrar la EPROM mediante la aplicación de luz ultravioleta a través de una "ventana" que posee en la parte superior del encapsulado. Tras este proceso de borrado se vuelven a grabar los mismos datos. Los datos permanecen grabados durante un largo tiempo. El proceso de borrado/escritura esta determinado a un número de veces.
   - EAROM o EEPROM (E2PROM), son memorias similares a las EPROM, pero en este caso se realizan por procedimientos eléctricos. Actualmente tienen una importancia fundamental en cualquier diseño. Dentro de este tipo hay que mencionar a las de acceso serie, que actualmente es la que más aceptación tiene en los diseños con microprocesadores y microcontroladores. Dentro del acceso serie tenemos las que utilizan el protocolo de comunicación serie I2C.

Veamos una representación esquemática de la clasificación anterior:

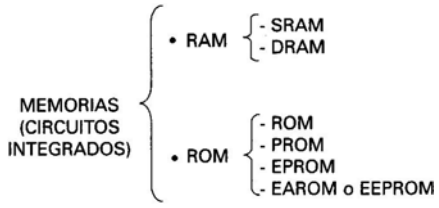


Figura 4.21

V
Dispositivos Lógicos Programables

5.1 TEORIA DE OPERACIÓN DE LOS PLD'S.

Saber en la Teoría (4 hrs.)

Los circuitos digitales de función fija o estándar, sirvan como ejemplo los de la serie 7400 de la familia TTL, o los de la serie 4000 de la familia CMOS, han sido los utilizados en todos los sistemas digitales desde principios de la década de los setenta. El objetivo de estos circuitos integrados era satisfacer las necesidades de un gran número de aplicaciones. Para aplicaciones complejas el número de circuitos integrados de función fija era elevado. La solución a este inconveniente era la utilización de circuitos integrados diseñados a medida, para dicha aplicación. Estos circuitos integrados también reciben el nombre de ASCI (Aplication Specific Integrated Circuit), circuito integrado de aplicación especifica.

5.2 CLASIFICACIÓN DE LO ASCI.

Los ASCI pueden ser clasificados de la siguiente forma:

1. Totalmente a medida (full custom), todo el proceso se realiza en fábrica, para conseguir un rendimiento máximo del silicio utilizado.
2. Casi a medida (semi custom), el fabricante del silicio proporciona al usuario la documentación y el software necesarios para que este optimice el funcionamiento del Circuito Integrado. Dentro de este tipo se puede realizar la siguiente clasificación:
    - Redes de puerta (Gate Array), cuando las funciones básicas que el usuario puede configurar son: funciones lógicas básicas, latches, Flip-Flop's, buffers, osciladores, registros, decodificadores, multiplexores, etc.
    - Células estándar (Estándar Cell), básicamente son idénticas a las redes de puertas. La diferencia fundamental es que no se trabaja con simples funciones, si no con un conjunto de combinaciones de éstas, que han sido configuradas y depuradas previamente por el fabricante.
3. Los circuitos integrados son programables por el usuario mediante herramientas comerciales, luego quien finaliza la fabricación del circuito integrado (programación) es el usuario. Dentro de este apartado se incluyen:
    - Memorias.
    - Microcontroladores (o microprocesadores).
    - PLD.
    - ASPLD.
    - FPGA.

La clasificación anterior la podemos ver de forma esquemática en la Figura 5.1:

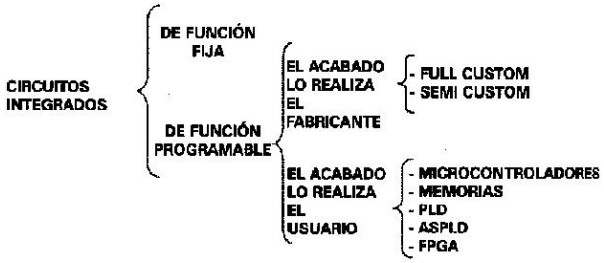


Figura 5.1

5.2.1 Los PLD.

Los PLD (Programmable Logic Device), Dispositivos Lógicos Programables. La denominación PLD es genérica a muchos tipos de componentes. La mayoría de los PLD consisten en una matriz de compuertas AND seguidas de una matriz de OR.

5.2.2 Los ASPLD.

Los ASPLD (Aplication Specific Programmable Logic Device), Dispositivos Lógicos Programables para Aplicaciones Específicas. Están diseñados para realizar funciones específicas con características especiales. El usuario programará el ASPLD para satisfacer la necesidad del diseño, pero el ASPLD está basado en una función específica. Un ejemplo seria utilizar un ASPLD cuya función sea un decodificador. El usuario podrá personalizarlo, pero sigue siendo un decodificador.

5.2.3 Los FPGA.

Los FPGA (Field Programmable Gate Array), arrays de compuertas de campo programables. Estos dispositivos contienen bloques lógicos independientes, con una complejidad igual a la de algunas de las PAL mas complejas. Estos bloques lógicos se pueden interconectar mediante conexiones programables, para obtener circuitos más complejos.

5.3 CLASIFICACIÓN DE LOS PLD.

El concepto básico de funcionamiento lo podemos en la Figura 5.2 en la que se identifican los fusibles que realizan o no la conexión entre varias líneas.

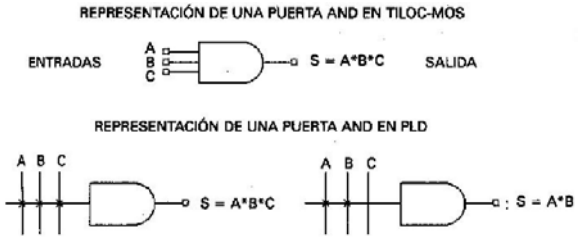


Figura 5.2

La cruz indica la existencia de una conexión entre líneas, la ausencia de cruz indica que no hay contacto entre las dos líneas que se cruzan. Por tanto podríamos decir que un PLD es virgen cuando tiene todas sus conexiones hechas, esto es, todos sus fusibles intactos. Para materializar una función bastará con eliminar las conexiones sobrantes, fundiendo los fusibles que correspondan a dichas funciones.
Partiendo de este pequeño ejemplo podemos ver cómo es la estructura básica de un PLD. Para poder seleccionar cualquier entrada habrá que construir una matriz con todas las combinaciones posibles y para las salidas también formaremos otra matriz. La matriz de entrada se construye con compuertas AND y la de la salida con compuertas OR, tal y como se puede ver en la Figura 5.3
La clasificaron de los PLD se realiza en función de cómo se pueden configurar dichas matrices. Se pueden enumerar como sigue:

- PAL (Programmable Logic Array).
- FPLA (Field Programmable Logic Array).
- PROM (Programmable Read Only Memory).

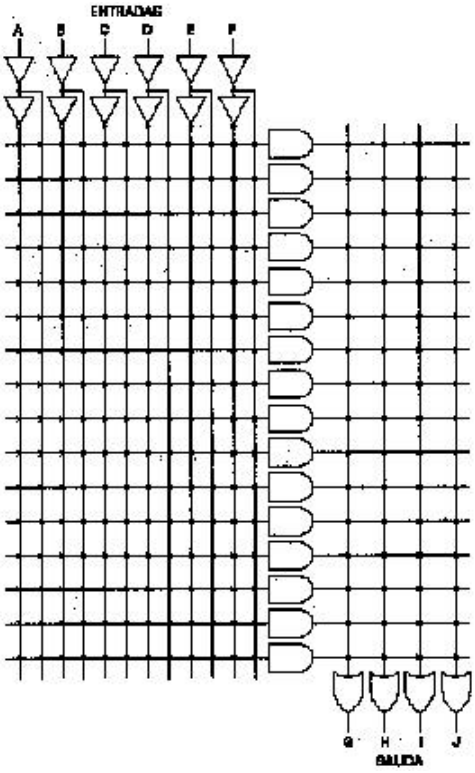


Figura 5.3

5.3.1 PAL.

Dentro de los dispositivos lógicos programables, los más antiguos y también los más conocidos son las PAL. Realmente este no es un nombre genérico, si no una marca registrada por su inventor, la sociedad MMI. En ésta nombraremos a los dispositivos PAL, como nombre genérico de este tipo de dispositivo, por ser la denominación mas extendida de todas.

La característica es que se programa la matriz que forman las uniones de las compuertas AND, y es fija la matriz que forman las uniones de las compuertas OR. En la Figura 5.3 se puede ver esta característica. Obsérvese que la representación de una unión fija, esto es, que no se puede configurar por el usuario, esta representada por un punto.
La PAL representada en la Figura 5.3 dispone de seis entradas (A, B, C, D, E, F) y cuatro salidas (G, H, I, J) y se podrán configurar un total de 16 * 12 = 192 fusibles.

5.3.2 FPLA.

La característica que distingue a este tipo de PLD es que se programa tanto la matriz que forman las uniones de las compuertas AND como la matriz que forman las uniones que forman las uniones de las compuertas OR. En la Figura 5.4 se puede ver esta característica.
El número de entradas y salidas es idéntico que en el anterior dispositivo, pero por tener las dos matrices configurables tendremos 16 * 16 = 256 fusibles para configurar. Son los dispositivos más flexibles, pero también los más lentos, por el número de transistores utilizados.

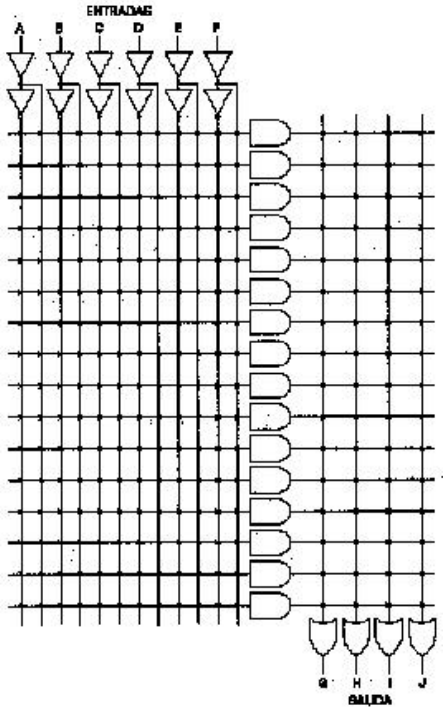


Figura 5.4

5.4 WinCUPL (Compilador Universal para Lógica Programable).

El CUPL, es un lenguaje de descripción y simulación. Por medio de éste se puede describir el funcionamiento de un circuito lógico (combinacional o secuencial), por medio de sus ecuaciones lógicas, por diagramas de estados o tablas.
Después de conocer las ecuaciones lógicas que rigen un circuito, o su diagrama de estados (para secuenciales), éste puede llevarse a CUPL, compilarse y obtenerse un archivo (.jed), que permite, por medio de un programador universal mediante el software ALL-11, programar un

dispositivo (pld), con la función que requiera el usuario. Por medio del CUPL, también se pueden hacer pruebas al circuito para comprobar que el diseño esté óptimo o dado el caso, hacer las correcciones necesarias.
CUPL tiene palabras y símbolos reservados, es decir, tienen un uso restringido dentro del programa, por ejemplo no pueden ser parte del nombre de ninguna variable. Para lógica combinacional se tienen símbolos especiales que representan las funciones lógicas básicas not, and, or y xor.

OPERACIONES LÓGICAS EN CUPL		PALABRAS RESERVADAS	SÍMBOLOS RESERVADOS
Operación	Símbolo	APPEND FORMAT PIN ASSEMBLY FUNCTION PINNODE ASSY IF PRESENT COMPANY JUMP LOCATION REVISION DATE MACRO SEQUENCE DESIGNER MIN DEVICE NAME SEQUENCERS ELSE NODE SEQUENCET FIELD OUT TABLE DEFINITION NEXT ORDER VECTORS PARNOT INPUTS EQUATIONS OUTPUTS JEDEC FLD PARNTNO IN	! & # $ * /* */ ( ) [ ] { } ^ / . , ; : = @ + - ‘
NOT	!		
AND	&		
OR	#		
XOR	$		

Tabla 5.1

VARIABLES

Una variable puede contener símbolos numéricos o alfabéticos, si contiene caracteres numéricos debe tener por lo menos un carácter alfabético.
La extensión máxima permitida es de 31 caracteres.
Una variable no puede contener símbolos o palabras reservadas, es permitido el símbolo (_), no pueden haber espacios en blanco intermedios.
El programa diferencia entre minúsculas y mayúsculas.
Ejemplos:

123a	variable	correcta
125	variable	incorrecta
ab_c	variable	correcta
entra	variable incorrecta	

VARIABLES DE ENTRADA

Son aquéllas que entran información al circuito, son asignadas a los pines dedicados a entrada en el dispositivo. Ejemplo: el reloj, un reset, set o un habilitador.

VARIABLES DE SALIDA

Son aquéllas que se utilizan para sacar información del circuito, son asignadas a los pines dedicados a salidas en el dispositivo. Ejemplo: la salida de una máquina de estados, una ecuación lógica x=a&b; en este caso x es una variable de salida.

VARIABLES INTERMEDIAS:

Son variables que se definen dentro del programa, y tienen como función abreviar términos, es decir, si se tiene un término que se repite en varias ecuaciones, este se puede definir como una variable intermedia, evitándose repetir éste innecesariamente.

Una de las ventajas de las variables intermedias, es que permite eliminar productos término, situación que es muy importante, ya que algunos PLD tienen un número fijo de productos términos para sus pines de salida.

USO DE BASES NUMÉRICAS

CUPL puede trabajar en base decimal, hexadecimal, octal o binaria, siempre por defecto usa el sistema hexadecimal. Si se desea usar una base específica se le debe informar al compilador; utilizando la siguiente notación: prefijo de base + número

Ejemplo:

'b' 010 base binaria
'h' f8 base hexadecimal
'o' 73 base octal

Nota: el prefijo de la base puede ser letra mayúscula o minúscula, se reconocerá cualquiera de ellas.

DEFINICIONES

CUPL permite dar un valor específico a una determinada variable en cualquiera de las bases ya explicadas.

Esto es muy importante cuando se hace uso de máquinas de estado y se necesita definir cada uno de los estados por los cuales ella puede pasar.

Para hacer una definición se utiliza la palabra reservada *define* y se le antepone el símbolo "$".

Ejemplo:

$ define s0 'b' 00 …. En este caso se define el estado s0 con el valor binario cero.

También es muy común definir un estado alto de una variable (*high*), por un valor uno, o un estado bajo (*low*) por un valor cero.

$ *define* H 'b'1
$ *define* L 'b'0

COMENTARIOS

Cuando se desea documentar un programa realizado, o se desea hacer un comentario en un lugar específico, se puede hacer uso de la siguiente sintaxis.

/* comentario */

Ejemplo:

/* esto es un comentario en *CUPL* */

Conocidas algunas generalidades del programa, podemos empezar a conocer sus plantillas básicas y la forma de utilizar éstas.

En CUPL existen dos archivos muy importantes que deben ser creados por el usuario, y en los cuales esta toda la información pertinente al diseño, como es la descripción del circuito y los valores de simulación, estos dos archivos tienen extensión (.pld) para el de descripción y (.si) para el de simulación.

Después de haber ejecutado el programa desde Windows, se accede a la primera opción en la barra de tareas. Esta es la opción *file*, dentro de ella se activa la opción *new*; a continuación se desplegará una ventana como la siguiente:

PLANTILLA .pld

```
Partno XXXXX;
Date XX/XX/XX;
Revision XX;
Name XXXXX;
Designer XXXXX;
Company XXXXX;
Assembly XXXXX;
Location XXXXX;
Device XXXXX;
/*************************************************************/
/*************************************************************/
/* Allowable Target Device Types: */
/*************************************************************/
/** Inputs **/
Pin = ; /* */
Pin = ; /* */
Pin = ; /* */
Pin = ; /* */
Pin = ; /* */
/** Outputs **/
Pin = ; /* */
Pin = ; /* */
Pin = ; /* */
Pin = ; /* */
Pin = ; /* */
/** Declarations and Intermediate Variable Definitions **/
/** Logic Equations **/
```

Veamos la segunda plantilla utilizada para dar los vectores de simulación al programa, esta es la plantilla (.si), para obtener ésta se realizan los pasos que se hicieron para obtener la anterior pero se le harán uno cambios y debe quedar con la información que presenta la siguiente plantilla:

PLANTILLA .si

```
Name XXXXX;
Partno XXXXX;
Date XX/XX/XX;
Revision XX;
Designer XXXXX;
Company XXXXX;
Assembly XXXXX;
Location XXXXX;
Device XXXXX;
/*************************************************************/
/* */
/*************************************************************/
/* Allowable Target Device Types: */
/*************************************************************/
order:
vectors:
```

Estas dos plantillas deben estar abiertas simultáneamente, al momento de guardar se debe tener la precaución de dar el mismo nombre a las dos plantillas, lo único que las diferenciará será la extensión. Ejemplo: practica.pld, practica.si
También se debe tener en cuenta el asignar al archivo de descripción la extensión .pld y al de simulación la extensión .si.

DESCRIPCIÓN DEL ENCABEZAMIENTO DE LOS ARCHIVOS

Name, este primer campo es de uso obligatorio, ya que el programa utilizará éste para darle nombre al archivo que se utilizará para programar el dispositivo, este archivo tendrá extensión .jed
Estos siete campos que se tienen a continuación se usan para documentar el programa, por lo tanto su uso no es obligatorio, esto queda a consideración del diseñador.

Partno
Date
Revision
Designer
Company
Assembly
Location

Device, este campo es de vital importancia, ya que en él se define el tipo de dispositivo a utilizar, es de uso obligatorio.
Además a este encabezado se le debe añadir un campo más que es el siguiente:
Format j: en este campo se define el tipo de archivo que generará el programa para programar el dispositivo. El resto de línea "j;" le indica que el formato es jed

NOTA: todas las instrucciones de programa deben terminar con un punto y coma, excepto las definiciones y los vectores.
IMPORTANTE: los encabezados de los dos archivos (.si) y (.pld) deben de ser exactamente iguales, de lo contrario se generarán errores de compilación. Se recomienda, realizar uno y copiarlo al otro archivo.

EJEMPLO

Este es un sencillo ejemplo de lógica combinacional donde se aprenderá a compilar un programa; en el ejemplo se realizarán las compuertas básicas.
Para ello se partirá de dos señales de entrada y con ellas se generarán cada una de las funciones.
Nota: los ejemplos se compilarán para el dispositivo g22v10.

Asignación de pines

Recuerde que se debe conocer con qué dispositivo se va a trabajar para conocer cuáles son sus pines de entrada y salida.
Los pines se pueden asignar una variable a cada uno o por campos, esto es, tanto para los de salida como para los de entrada.
Ejemplo: definición pines de entrada en el problema.
pin 1=A;
pin 2=B;
o por campo así:

pin [1,2]=[A,B];

Ahora veamos como quedaría la plantilla .pld

```
Name basicas;
Partno 01;
Date 01/02/01;
Revision 01;
Designer alumno;
Company eafit;
Assembly no;
Location cdigital;
Device g22v10;
Format j;
/*****************************************************************/
/*Compuertas basicas en cupl */
/* not, and, or, xor, nand, nor, xnor */
/*****************************************************************/
/** Inputs **/
Pin 1=A;                /*variable de entrada*/
Pin 2=B;                /*variable de entrada */
/** Outputs **/
Pin 14=nota;            /*variable de salida negacion de a*/
Pin 15=AandB;  /*variable de salida operacion and entre A y B */
Pin 16=AorB;            /*variable de salida operacion or entre A y B */
Pin 17=AxorB;   /*variable de salida operacion xor entre A y B*/
Pin 18=AnandB;          /*variable de salida operacion nand entre A y B*/
Pin 19=AnorB;   /*variable de salida operacion nor entre A y B*/
Pin 20=AxnorB;          /*variable de salida operacion xnor entre A y B*/
/** Declarations and Intermediate Variable Definitions **/
/*debido a lo simple del programa no se usan variables intermedias*/
/** Logic Equations **/
nota=!A;
AandB=A&B;
AorB=A#B;
AxorB=A$B;
AnandB=!(A&B);
AnorB=!(A#B);
AxnorB=!(A$B);
```

Ahora veamos los comandos que aparecen en la plantilla (.si).
Estos son: *ORDER, VECTORS*
order, este comando le indica al programa de simulación en qué orden debe tomar las variables de entrada y salida, este orden es dado por el usuario de acuerdo al peso de ellas o a la significancia que tenga dentro del problema; cuando existe reloj este siempre será la primera variable en indicarse.

Para indicarle el orden al programa se da el siguiente formato *order:* %n,variable1,%n,variable2; donde:
n es un número entero[0..9] que indicará al compilador cuántos espacios deja entre las columnas de la tabla que él creará.
Las variables deben ir separadas entre si por una coma, y se debe terminar toda la línea con punto y coma.
Vectors este comando le indica al compilador que lo que encuentra a continuación son vectores, los cuales usará en la simulación.
Estos vectores asignan valores a las variables de entrada para probar el circuito; las variables de salida se colocan como (*) lo cual indica al compilador que se está esperando una respuesta.

Continuando con el ejemplo veamos como debe quedar el archivo (.si)

```
Name basicas;
Partno 01;
Date 01/02/01;
Revision 01;
Designer alumno;
Company eafit;
Assembly no;
Location cdigital;
Device g22v10;
Format j;
/*********************************************************************/
/*Compuertas basicas en cupl */
/* not, and, or, xor, nand, nor, xnor */
/*********************************************************************/
order:
%2,A,%2,B,%2,nota,%2,AandB,%2,AorB,%2,AxorB,%2,AnandB,%2,AnorB,%2,AxnorB;
vectors:
00 *******
01 *******
10 *******
11 *******
```

Luego de haber realizado los dos archivos, se procederá a compilarlos, veamos a continuación como se compila un archivo en CUPL.

COMPILACIÓN DE LÓGICA COMBINATORIA

La compilación se puede hacer sobre cualquiera de los dos archivos creados. Para ello en el menú *option* se activa la opción (*Compiler Options*), se desplegará la siguiente ventana:

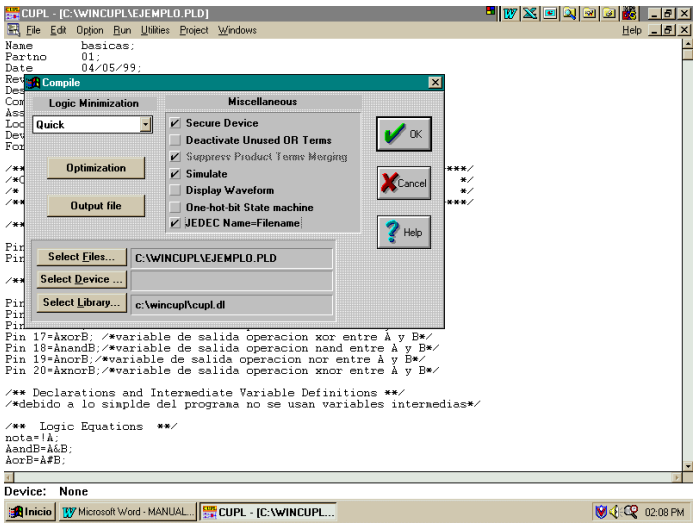


Figura 5.5

En la Figura 5.5, en el recuadro perteneciente a *Miscellaneous* se deben activar las siguientes opciones: *Secure Device, Simulate, Jedec Name=Filename.*
Luego de haber activado las anteriores, se continúa con la opción de esta ventana llamada *Output file*, se desplegará la siguiente ventana:

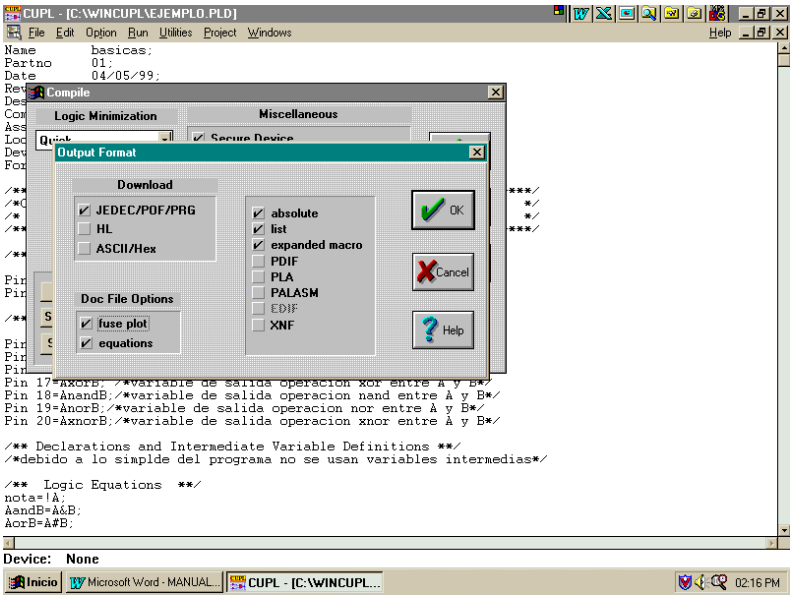


Figura 5.6

En la Figura 5.6 se presentan las siguientes opciones en las que se activaran las siguientes opciones:
Download, sólo se activa *jedec/pof/prg.*
Output, se activarán *absolute, list y expanded macro.*
Doc File options, se activarán *fuse plot y equations.*

Luego de presionar *OK* se regresa a la ventana principal donde se activará la opción *select device*, donde se desplegará la siguiente ventana:

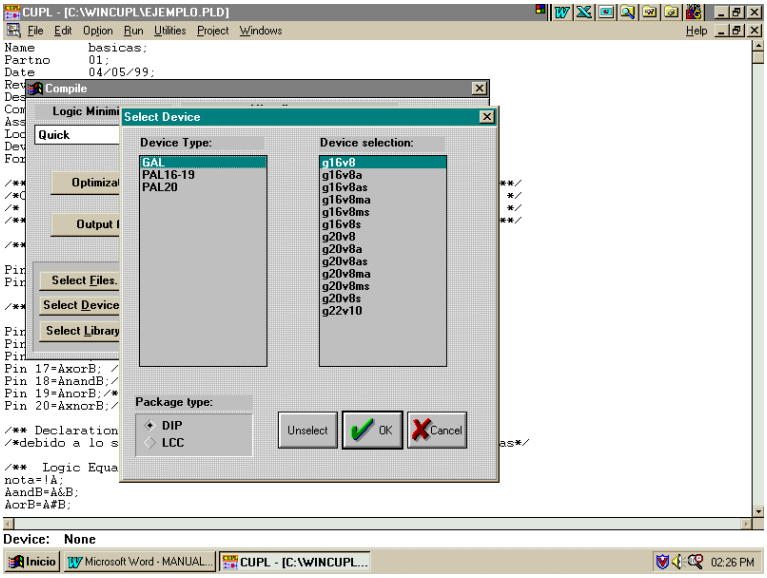


Figura 5.7

En este se seleccionará el dispositivo que desea usar en su circuito, luego se presiona *OK* y se regresará a la ventana principal donde también se presionará *OK*, ver Figura 5.7.
Luego en el menú *option* se activará la opción (*Simulator options*) se activará la siguiente ventana:

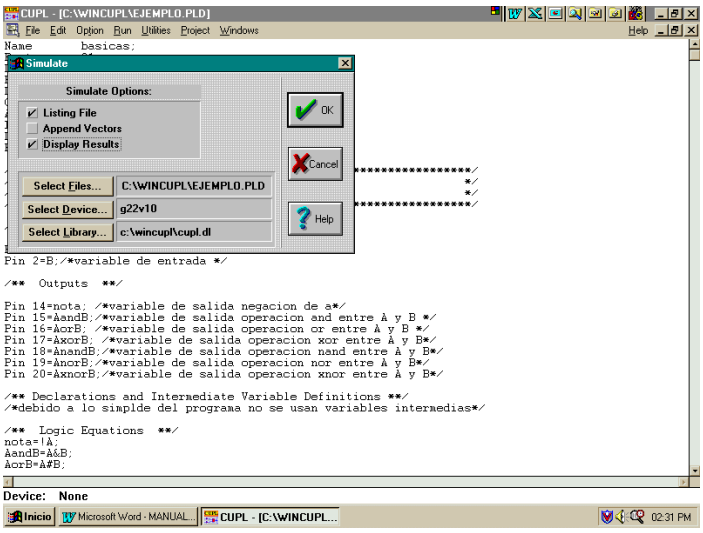


Figura 5.8

En la ventana *Simulate options* se activará: *Listing file y Display results*, luego se presiona *OK*.
A continuación en el menú run se activa la opción (*Device specific compile*) y el programa ejecutará la compilación.
Si la compilación fue exitosa el programa creará los archivos con extensión (.so) y (.jed).

VI
Proyecto

Horas prácticas (14 hrs.)

Objetivo particular de la unidad

Que el alumno aplique los conocimientos adquiridos durante el desarrollo de la materia en la aplicación de un sistema con uso práctico.

Proyectos sugeridos:

MAQUINAS DE ESTADOS

1.- Se desea controlar dos bombas B1 y B2 que vierten liquido a un depósito, de acuerdo con el nivel de líquido existente en él. El depósito depende de dos sensores, d arriba y c abajo, que se activan al detectar líquido. Su funcionamiento ha de ser el siguiente:

- Cuando el nivel de líquido se encuentra comprendido entre los dos sensores "c" y "d" debe funcionar la bomba B1 o B2 de forma alternativa, es decir, una vez B1 y otra vez B2, y se parará cuando se active el sensor "d"(deposito lleno).
- Si el nivel de líquido se encuentra por debajo de "c" se deben activar ambas bombas. En caso de funcionamiento anormal de los sensores del depósito (se active "d" cuando no lo esté "c"), ambas bombas se pararán.
- Además, ambas bombas cuentan con sendos detectores de temperatura "a" y "b" para B1 y B2 respectivamente, de tal forma que si la temperatura de su motor supera un cierto limite, el detector se activará y la correspondiente bomba se parará.

Se pide diseñar el circuito de control completo, diagrama de estados, reducción de estados, diagrama transición-estado y diagrama del circuito implementado mediante biestables J-K.

2.- Se desea diseñar el control del vaivén de un móvil, que cumpla las siguientes condiciones:

a) Si el móvil se encuentra situado en el extremo izquierdo, pulsando "C", iniciará el movimiento hacia la derecha (QB activado), de forma que cuando llegue al extremo derecho presionará el final de carrera "B" y volverá automáticamente a la izquierda (QA activado), deteniéndose cuando active el final de carrera "A".

b) Si el móvil se encuentra detenido en cualquier punto intermedio del recorrido (final de carrera "A" desactivado), pulsando "C" regresará al extremo izquierdo (QA activado) y se parará.

c) Si el móvil se encuentra en movimiento, actuar sobre "C" no supondrá ningún cambio en el ciclo que esté realizando.

d) Adicionalmente se dispondrá de una pulsador de emergencia "D" que detendrá el móvil.

Se pide:

i) Diagrama de estado reducido.

ii) Diagrama del circuito realizado mediante memoria ROM y contenido de esta.

3.- Para accionar un motor de c.a. se utilizan cuatro contactores, los cuales están controlados por otros tantos relés: Q1 (giro a izquierda), Q2 (giro a derecha), Q3 (conexión en estrella) y Q4 (conexión en triángulo). Dicho sistema va provisto de tres pulsadores "A", "B" y "C", de forma que pulsando "A" el motor debe girar a izquierdas y se debe conectar en estrella; pulsando "B" girará a derechas y se conectará en triángulo. Pulsando "C" el motor se ha de parar, y pulsando "A" y "B" simultáneamente éste ha de permanecer en el estado anterior.
Se pide el diagrama de estados y el circuito que implementa este diagrama. Emplear la construcción de la máquina de estados que se considere más idónea.

4.- Empleando un contador binario y operadores lógicos, diseñar un circuito divisor de frecuencia por 1001 veces la de entrada, es decir que cuando el circuito haya detectado 1001 pulsos a su entrada genere un pulso a la salida.

5.- Diseñar una pila LIFO (Last in, First Out: El último en entrar es el primero en salir) de cuatro posiciones que es capaz de guardar datos de cuatro bits cada uno. Las entradas apilar y desapilar serán activas durante un ciclo de reloj y se producirá un error cuando:
- Se activen al mismo tiempo apilar y desapilar.
- Se active apilar y la pila este llena.
- Se active desapilar y la pila esté vacía.

Se pide diagrama de estado del controlador, tabla de estado, diagrama lógico usando un registro y una PAL.

6.- Diseñar un dispositivo que permita visualizar el "turno" de espera en un establecimiento comercial mediante dos dígitos decimales. Dicho sistema debe ir provisto de un mecanismo que permita saltar uno o varios números sin que aparezcan visualizados, siempre que el vendedor así lo desee. Calcular el número de salidas registradas necesarias antes de comenzar el diseño. Seleccionar y usar el mínimo número de dispositivos secuenciales.

7.- Diseñar un contador síncrono ascendente/descendente de módulo 10, con entradas de precarga controladas por una señal de validación (L). El circuito ha de poseer también una entrada de puesta a cero (R) y de cuenta ascendente/descendente (U/D) ambas activas a nivel bajo. Emplear los biestables adecuados.

8.- Diseñar un dispositivo capaz de simular un dado digital. El dispositivo dispondrá de un display para poder visualizar el número seleccionado, además de un pulsador de reset (R) y otro de marcha (M), de tal forma que el número obtenido dependa del tiempo que esté actuando sobre este último. Emplear los dispositivos secuenciales más adecuados.

9.- Diseñar un dispositivo que genere la siguiente secuencia de salida: "101->100->110->010->011->001->101->...".
Representar el cronograma correspondiente obtenido a la salida del circuito.

10.- Diseñar un registro de aproximaciones sucesivas para un convertidor A/D:

El registro de aproximaciones sucesivas comienza poniendo a 1 el bit de más peso (MSB), quedando el resto a cero, o sea, forma el valor 1000..0, que corresponde a la mitad de la máxima tensión de salida. Este valor es transformado a señal analógica, Vc, que a su vez se introduce al comparador.
Si la señal VC es mayor que VI, el comparador bascula dando lugar a una señal que hace que el registro varíe su contenido, sustituyendo el 1 del bit de más peso por un 0 y colocando en el bit de peso inmediatamente inferior un 1, quedando inalterado el resto de los bits (0100..0).
Si la señal VC fuese menor que VI, el registro no modificaría el bit de más peso, pero coloca el bit de peso inmediatamente inferior a 1, dejando a 0 el resto de los bits (1100..0). Tanto en un caso como en otro, se efectúa una nueva conversión.
Resolver el diseño del registro de aproximaciones sucesivas mediante los biestables y dispositivos secuenciales adecuados.

11.- Diseñar un contador binario de 8 bits U/D que se bloquee al llegar a la última cuenta y genere una señal. La señal RESET se emplea para inicializar a cero y desbloquear el contador.

12
.- Se desea controlar una puerta corredera, la cual puede moverse de izquierda a derecha y viceversa. Su recorrido está limitado por dos finales de carrera, uno en la parte izquierda (B) que se activará cuando la puerta esté completamente abierta y otro en la parte derecha (C) que indicará que está completamente cerrada. Inicialmente la puerta se encuentra cerrada y presionando sobre el final de carrera (C), de tal forma que actuando sobre un pulsador (A), ésta comenzará un ciclo de apertura moviéndose hacia la izquierda hasta que alcance el final de carrera (B). A partir de este momento, la puerta invertirá su movimiento, desplazándose hacia la derecha hasta estar completamente cerrada, es decir, hasta activar el final de carrera (C). Si por cualquier circunstancia se pulsa (A) durante el movimiento de la puerta, ésta no debe detenerse.

Anexo A

Prácticas de Sistemas Digitales

Saber Hacer en la practica (2 hrs.)

PRÁCTICA NO 1. Compuertas Not, And, Or, Xor.

DATOS DE LA PRÁCTICA

CARRERA	ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL			
CUATRIMESTRE	**TERCERO**		**GRUPO**	
TIPO PRÁCTICA	☐ **LABORATORIO** ☐ **SIMULACION**		**FECHA**	
ASIGNATURA	**SISTEMAS DIGITALES**			
UNIDAD TEMÁTICA				
No DE ALUMNOS POR PRÁCTICA	**3**	**No. DE ALUMNOS POR REPORTE**		**3**
NOMBRE DEL PROFESOR				
NOMBRE (S) DE ALUMNO (S)				
TIEMPO ESTIMADO	**2h**	**VO. BO. DEL PROFESOR**		
COMENTARIOS				

OBJETIVO:

Conocer el funcionamiento de las compuertas lógicas y realizar su comprobación con su respectiva tabla de verdad.

RECURSOS MATERIALES UTILIZADOS (CONSUMIBLES):

- 1 CI TTL 74LS04
- 1 CI TTL 74LS08
- 1 CI TTL 74LS32
- 1 CI TTL 74LS86
- 1 LED
- 1 Dipswitch (de 4 u 8 interruptores)
- 2 Resistencias 10 KΩ.
- 1 Resistencia 330 Ω.

HERRAMIENTAS Y EQUIPO:

- 1 Fuente de 5 Vcc.
- 1 Protoboard ó Tablilla de conexiones.

DESARROLLO O PROCEDIMIENTO:

Conectar alguna de las compuertas de cada CI y aplicar niveles de voltaje de 5 V a cada una de las entradas, conectar un led a la salida de la compuerta y aterrizarlo a través de una resistencia, como se muestra en el diagrama de la Figura A1.1, y posteriormente sustituir cada una de las compuertas para poder obtener la tabla de verdad de cada una de ellas.

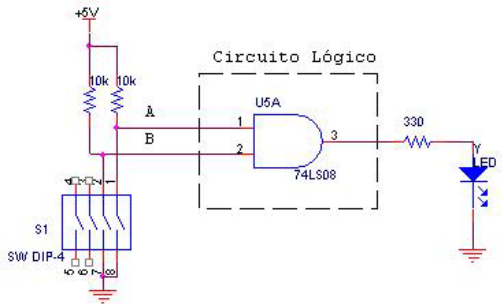


Figura A1.1

AND		
A	B	Y
0	0	
0	1	
1	0	
1	1	

Tabla A1.1a

OR		
A	B	Y
0	0	
0	1	
1	0	
1	1	

Tabla A1.1b

XOR		
A	B	Y
0	0	
0	1	
1	0	
1	1	

Tabla A1.1c

NOT	
A	A'
0	
1	

Tabla A1.1d

CUESTIONARIO.

- Con los circuitos integrados que se utilizaron, ¿cuántas compuertas tiene cada integrado?
- Si a la salida de cada compuerta XNor se conecta un inversor, ¿Qué tabla se obtiene?

RECOMENDACIONES.

Para complementar el funcionamiento de todas las compuertas lógicas se recomienda realizar el mismo procedimiento, para verificar la función de las compuertas NAND, NOR y X-NOR (7400, 7402 y 74266), ya sea adquiriendo los modelos respectivos de cada CI ó verificar si con los que ya se tienen se pueden realizar dichas compuertas lógicas.

Nota: Se dejan las tablas de verdad para ser llenadas.

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

Tabla A1.2a

NOR		
A	B	Y
0	0	
0	1	
1	0	
1	1	

Tabla A1.2b

XNOR		
A	B	Y
0	0	
0	1	
1	0	
1	1	

Tabla A1.2c

Saber en la Teoría (7 hrs.)

PRÁCTICA NO 2. Construcción de circuitos lógicos utilizando compuertas básicas.

DATOS DE LA PRÁCTICA

CARRERA	ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL			
CUATRIMESTRE	**TERCERO**		**GRUPO**	
TIPO PRÁCTICA	☐ **LABORATORIO** ☐ **SIMULACION**		**FECHA**	
ASIGNATURA	**SISTEMAS DIGITALES**			
UNIDAD TEMÁTICA				
No DE ALUMNOS POR PRÁCTICA	**3**	**No. DE ALUMNOS POR REPORTE**		**3**
NOMBRE DEL PROFESOR				
NOMBRE (S) DE ALUMNO (S)				
TIEMPO ESTIMADO	**2h**	**VO. BO. DEL PROFESOR**		
COMENTARIOS				

OBJETIVO:

Obtener un circuito lógico a partir de una función dada, utilizando el método de mapas de Karnaugh.

RECURSOS MATERIALES UTILIZADOS (CONSUMIBLES):

- Circuitos Integrados 74LSXX (serán determinados).
- 1 LED.
- 1 Dipswitch (de 4 u 8 interruptores)
- 4 Resistencias 10 KΩ.
- 1 Resistencia 330 Ω.

HERRAMIENTAS Y EQUIPO:

- 1 Fuente de 5 Vcc.
- 1 Protoboard ó Tablilla de conexiones

DESARROLLO O PROCEDIMIENTO:

a) Diseñar y conectar en laboratorio un circuito lógico generado a partir de la función: $F(a,b,c,d) = \sum(0,1,2,4,5,6,8,9,12,13,14)$.

b) Se deberá llenar la tabla de verdad que se muestra en la Figura A2.1, utilizando el método de mapas de Karnaugh.

c) Así mismo completar el diagrama de la Figura A2.2

a	b	c	d	F
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

Tabla A2.1 Tabla de verdad.

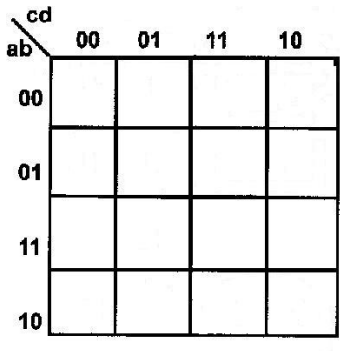


Figura A2.1 Mapa de Karnaugh.

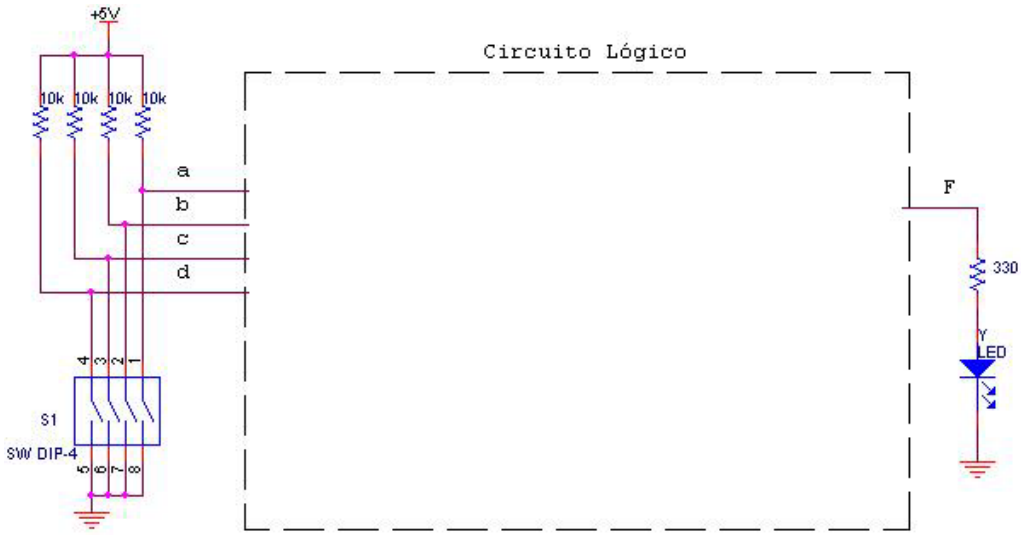


Figura A2.2

CUESTIONARIO:

- ¿Cuántas compuertas lógicas se utilizaron?
- Al comparar con los demás equipos, ¿Se obtuvieron los mismos circuitos lógicos?

RECOMENDACIONES:

Para esta práctica de contador binario se recomienda agregar un decodificador 74LS48 o 74LS47 para poder controlar un display de siete segmentos de cátodo o ánodo común respectivamente, en el display se mostraran las 16 posibles combinaciones secuencialmente.

PRÁCTICA NO 3. Circuito lógico de conversión de código Binario a Gray.

DATOS DE LA PRÁCTICA

CARRERA	ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL			
CUATRIMESTRE	**TERCERO**		**GRUPO**	
TIPO PRÁCTICA	☐ **LABORATORIO** ☐ **SIMULACION**		**FECHA**	
ASIGNATURA	**SISTEMAS DIGITALES**			
UNIDAD TEMÁTICA				
No DE ALUMNOS POR PRÁCTICA	**3**	**No. DE ALUMNOS POR REPORTE**		**3**
NOMBRE DEL PROFESOR				
NOMBRE (S) DE ALUMNO (S)				
TIEMPO ESTIMADO	**2h**	**VO. BO. DEL PROFESOR**		
COMENTARIOS				

OBJETIVO:

Que el alumno aplique compuertas básicas para construir un circuito lógico que realice la conversión de código Binario a Gray.

RECURSOS MATERIALES UTILIZADOS (consumibles):

- 3 TTL 7486.
- 4 Microswitch's.
- 4 Led.
- 4 Resistencias 10K.
- 1 Resistencia 330.

HERRAMIENTAS Y EQUIPO:

- 1 Fuente de 5 Vcc.
- 1 Tablilla de conexiones o protoboard.

DESARROLLO O PROCEDIMIENTO:

a) Conectar el diagrama de la Figura A3.1 que corresponde al conversor de código binario a Gray.
b) Comprobar el funcionamiento con las tablas de verdad A3.1a y A3.1b.
c) Hallar las Funciones lógicas que se muestra en la Figura A3.2, por el método de Mapas de Karnaugh.

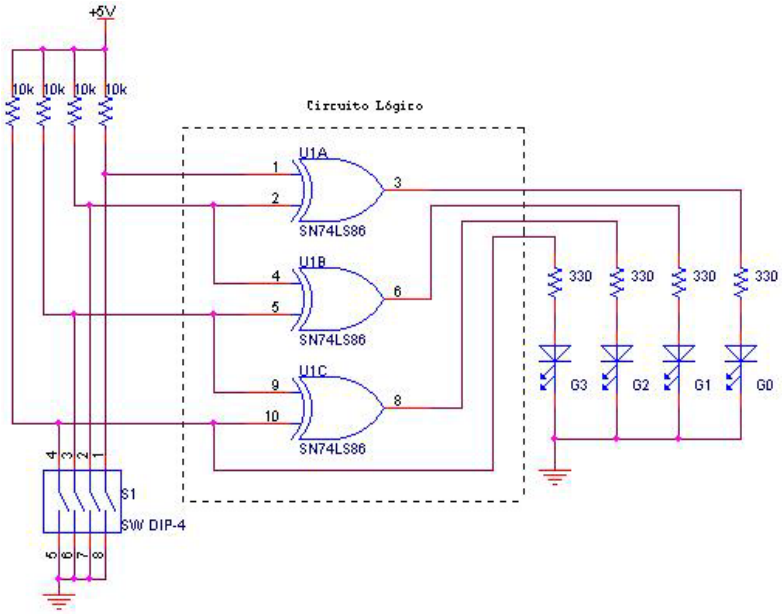


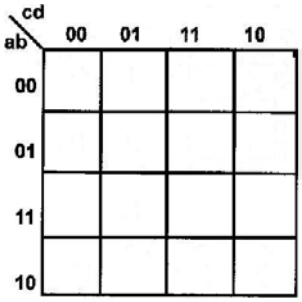
Figura A3.1

Equivalente decimal	Código binario $B_3B_2B_1B_0$	Cód. reflejado ó Gray $G_3G_2G_1G_0$
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0

Tabla A3.1a

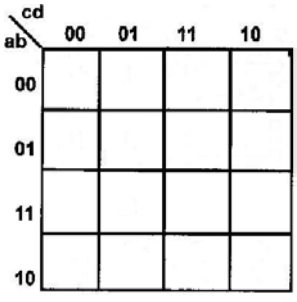
Equivalente decimal	Código binario $G_3G_2G_1G_0$	Cód. reflejado ó Gray $G_3G_2G_1G_0$
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1
10	1 0 1 0	1 1 1 1
11	1 0 1 1	1 1 1 0
12	1 1 0 0	1 0 1 0
13	1 1 0 1	1 0 1 1
14	1 1 1 0	1 0 0 1
15	1 1 1 1	1 0 0 0

Tabla A3.1b



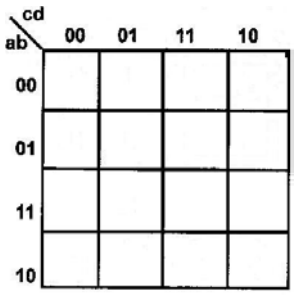
$G_0 =$

Figura A3.2ª



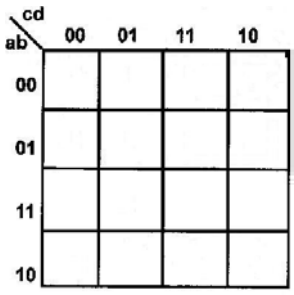
$G_1 =$

Figura A3.2b



$G_2 =$

Figura A3.2c



$G_3 =$

Figura A3.2d

CUESTIONARIO:

- ¿Investigar en qué aplicaciones es utilizado el circuito de conversión de Binario a Gray?
- ¿Investigar en qué aplicaciones es utilizado el circuito de conversión de Gray a Binario?

RECOMENDACIONES:

Se recomienda obtener el circuito lógico de conversión de código Gray a binario.

PRÁCTICA NO 4. Diseño y montaje de un circuito lógico generado a partir de un diagrama eléctrico de control.

DATOS DE LA PRÁCTICA

CARRERA	ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL			
CUATRIMESTRE	**TERCERO**		**GRUPO**	
TIPO PRÁCTICA	☐ **LABORATORIO** ☐ **SIMULACION**		**FECHA**	
ASIGNATURA	**SISTEMAS DIGITALES**			
UNIDAD TEMÁTICA				
No DE ALUMNOS POR PRÁCTICA	**3**	**No. DE ALUMNOS POR REPORTE**		**3**
NOMBRE DEL PROFESOR				
NOMBRE (S) DE ALUMNO (S)				
TIEMPO ESTIMADO	**2h**	**VO. BO. DEL PROFESOR**		
COMENTARIOS				

OBJETIVO:

Que el alumno aplique los conocimientos obtenidos sobre compuertas lógicas para construir un circuito lógico que realice función de un diagrama eléctrico de control.

RECURSOS MATERIALES UTILIZADOS (CONSUMIBLES):

- CI's TTL (serán determinados).
- 3 Microswitch's.
- 3 Led.
- 3 Resistencias de 10K.
- 3 Resistencias de 330.

HERRAMIENTAS Y EQUIPO:

- 1 Fuente de 5 Vcc.
- 1 Tablilla de conexiones o protoboard.

DESARROLLO O PROCEDIMIENTO:

a) Obtener el circuito lógico para el diagrama eléctrico de control de la Figura A4.1
b) Montar el circuito y comprobar que cumple la operación de paro y arranque.

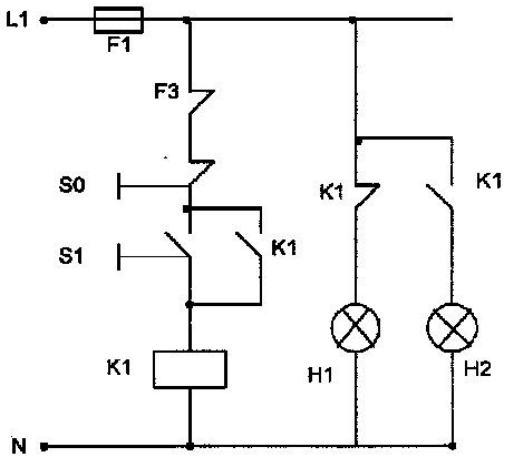


Figura A4.1

CUESTIONARIO:

- ¿Este circuito de control en qué tipo de aplicaciones puede ser utilizado?
- Para acoplar motores, lámparas que trabajen a 120 ó 220 V ¿Qué dispositivos son necesarios?

RECOMENDACIONES:

Se recomienda obtener el circuito lógico para el control de un motor trifásico que trabaje en forma reversible (cambio de giro).

Saber en la Práctica (14 hrs.)

PRÁCTICA No. 5. Manejo de Display de siete segmentos.

DATOS DE LA PRÁCTICA

CARRERA	ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL			
CUATRIMESTRE	**TERCERO**		**GRUPO**	
TIPO PRÁCTICA	☐ **LABORATORIO** ☐ **SIMULACION**		**FECHA**	
ASIGNATURA	**SISTEMAS DIGITALES**			
UNIDAD TEMÁTICA				
No DE ALUMNOS POR PRÁCTICA	**3**	**No. DE ALUMNOS POR REPORTE**		**3**
NOMBRE DEL PROFESOR				
NOMBRE (S) DE ALUMNO (S)				
TIEMPO ESTIMADO	**2h**	**VO. BO. DEL PROFESOR**		
COMENTARIOS				

OBJETIVO:

Que el alumno comprenda el funcionamiento de un decodificador BCD a 7 segmentos, así como el manejo de los display's de siete segmentos.

RECURSOS MATERIALES UTILIZADOS (consumibles):

- 1 TTL 7447.
- 1 TTL 7448.
- 1 Display de ánodo común.
- 1 Display de cátodo común.
- 4 Interruptores.
- 4 Resistencias de 10K.
- 7 Resistencias de 330.

HERRAMIENTAS Y EQUIPO:

- 1 Fuente de 5 Vcc.
- 1 Tablilla de conexiones o protoboard.

DESARROLLO O PROCEDIMIENTO:

En esta práctica se plantea la conexión de un decodificador BCD a 7 segmentos, así como el manejo de los dos diferentes tipos de display's de siete segmentos.

a) Conectar el diagrama de la Figura **A5.1** , donde se muestra la conexión de un 7447 a un display de ánodo común.

b) Conectar el mismo diagrama, sólo que ahora se realiza un reemplazo del 7447 por el 7448 y del display de ánodo común por el cátodo común.

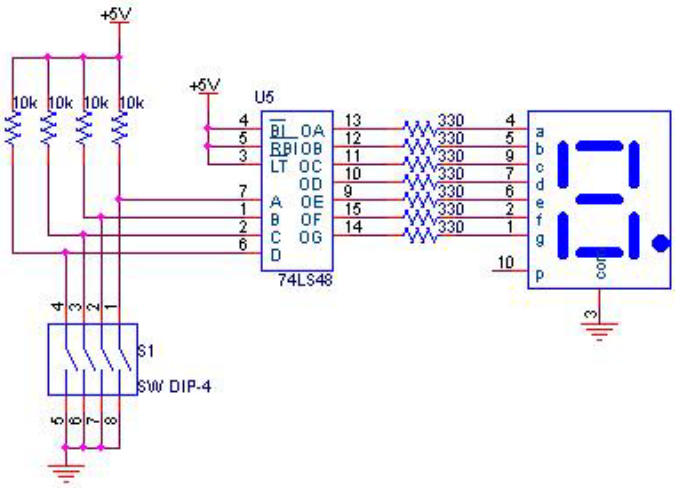


Figura A5.1

CUESTIONARIO:

- ¿Cuál es la diferencia principal entre el 7447 y el 7448?
- ¿Cuál es la diferencia principal entre un display de ánodo común y uno de cátodo común?

RECOMENDACIONES:

Se recomienda obtener el circuito lógico para el control de un motor trifásico que trabaje en forma reversible (cambio de giro).

PRÁCTICA No. 6. Circuito de control de entrada/Salida y conversión de código.

DATOS DE LA PRÁCTICA

CARRERA	ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL			
CUATRIMESTRE	**TERCERO**		**GRUPO**	
TIPO PRÁCTICA	☐ **LABORATORIO** ☐ **SIMULACION**		**FECHA**	
ASIGNATURA	**SISTEMAS DIGITALES**			
UNIDAD TEMÁTICA				
No DE ALUMNOS POR PRÁCTICA	**3**	**No. DE ALUMNOS POR REPORTE**		**3**
NOMBRE DEL PROFESOR				
NOMBRE (S) DE ALUMNO (S)				
TIEMPO ESTIMADO	**2h**	**VO. BO. DEL PROFESOR**		
COMENTARIOS				

OBJETIVO:

Que el alumno compruebe el funcionamiento de un codificador Decimal – BCD.

RECURSOS MATERIALES UTILIZADOS (consumibles):

- 1 TTL 74147.
- 1 TTL 7404.
- 1 TTL 7447.
- 1 Display de ánodo común.
- 4 Interruptores.
- 4 Led's.
- 7 Resistencias de 330.

HERRAMIENTAS Y EQUIPO:

- 1 Fuente de 5 Vcc.
- 1 Tablilla de conexiones o protoboard.

DESARROLLO O PROCEDIMIENTO:

Se plantea el diseño de un circuito digital para incorporar al panel de control de un automóvil de manera que al abrir las puertas del vehículo se encienda un led mostrando la puerta o puertas abiertas y además visualice en un display de 7 segmentos un número asignado a la misma, el esquema quedaría como se muestra en la Figura A6.1.

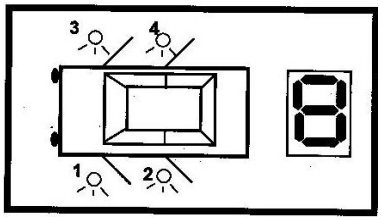


Figura A6.1

La resolución del circuito seria la siguiente:

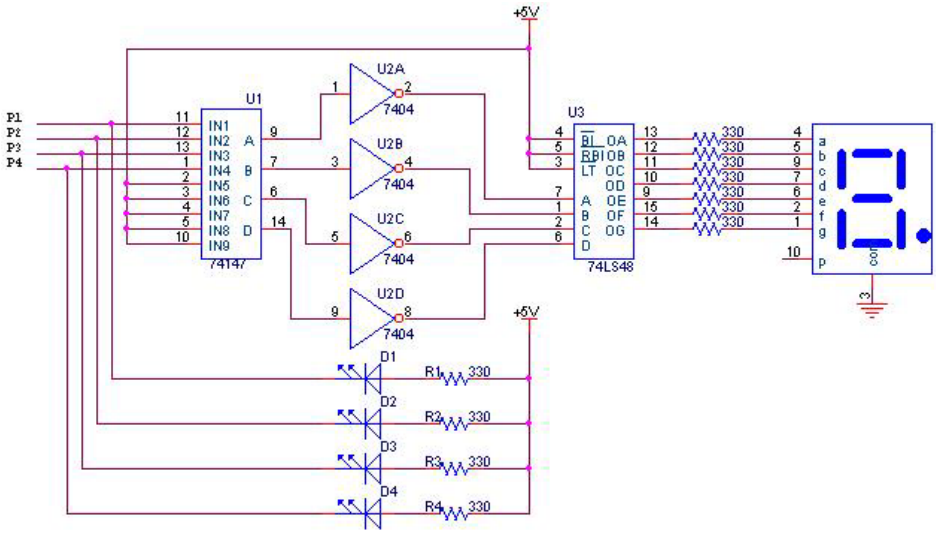


Figura A6.2

Se parte de que los pulsadores de las puertas se cierran a tierra al abrirlas. También será necesario adaptar a 5 voltios los 12 voltios que proporciona la batería de un automóvil, o se podría optar por resolver el circuito con integrados de la familia CMOS.

CUESTIONARIO:

- ¿Cuál es la función de los inversores?
- ¿Cómo funciona el CI 74147?

RECOMENDACIONES:

Se recomienda repetir esta práctica en la unidad de circuitos secuenciales, de tal manera que los cuatro bits de entrada sean cambiados automáticamente.

PRÁCTICA No. 7. Diseño y montaje de un circuito lógico para el control de cualquier segmento en un display de siete segmentos.

DATOS DE LA PRÁCTICA

CARRERA	ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL			
CUATRIMESTRE	**TERCERO**		**GRUPO**	
TIPO PRÁCTICA	☐ **LABORATORIO** ☐ **SIMULACION**		**FECHA**	
ASIGNATURA	**SISTEMAS DIGITALES**			
UNIDAD TEMÁTICA				
No DE ALUMNOS POR PRÁCTICA	**3**	**No. DE ALUMNOS POR REPORTE**		**3**
NOMBRE DEL PROFESOR				
NOMBRE (S) DE ALUMNO (S)				
TIEMPO ESTIMADO	**4h**	**VO. BO. DEL PROFESOR**		
COMENTARIOS				

OBJETIVO:

Que el alumno realice el diseño de un circuito lógico para controlar cualquier segmento en un display de siete segmentos mediante algún método de simplificación utilizando compuertas lógicas.

RECURSOS MATERIALES UTILIZADOS (consumibles):

- 1 TTL 7448.
- CI's de circuito lógico serán determinados
- 1 Display de cátodo común.
- 4 Interruptores.
- 4 Resistencias de 10K.
- 7 Resistencias de 330.

HERRAMIENTAS Y EQUIPO:

- 1 Fuente de 5 Vcc.
- 1 Tablilla de conexiones o protoboard.

DESARROLLO O PROCEDIMIENTO:

Diseñar un circuito lógico generado a partir del segmento que se quiera controlar, de acuerdo a los números que tienen que ser mostrados en el display de siete segmentos como se muestra en la Figura 3.20. En la Tabla A7.1 se muestra la columna para el control del segmento **f,** se sugiere llenar toda la Tabla A7.1, y posteriormente una vez seleccionado el segmento a controlar utilizar el método de mapas de Karnaugh para hallar la función simplificada, en esta práctica se propone dividir al grupo en equipos y que cada equipo realice un diseño de circuito lógico para un segmento, siendo diferentes para cada equipo.

Tabla de Verdad

Dec.	A	B	C	D	g	f	e	d	c	b	a
0	0	0	0	0		1					
1	0	0	0	1		0					
2	0	0	1	0		0					
3	0	0	1	1		0					
4	0	1	0	0		1					
5	0	1	0	1		1					
6	0	1	1	0		1					
7	0	1	1	1		0					
8	1	0	0	0		1					
9	1	0	0	1		1					
10	1	0	1	0		0					
11	1	0	1	1		0					
12	1	1	0	0		1					
13	1	1	0	1		1					
14	1	1	1	0		1					
15	1	1	1	1		0					

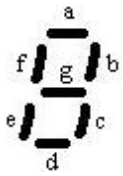


Tabla A7.1

Figura A7.1

En el circuito electrónico de la Figura 3.21, se muestra el ejemplo de control para el segmento **f**, generado a partir de la columna **f** de la Tabla A7.1

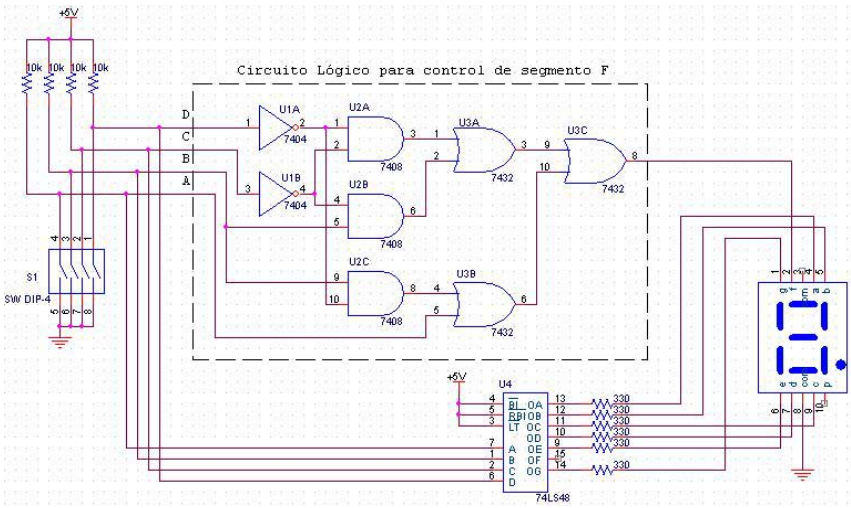


Figura A7.2

CUESTIONARIO:

- Comparar con los demás equipos para verificar, ¿Cuál circuito lógico contiene más compuertas y cuál menos?
- Si el display es de ánodo común ¿Qué es necesario modificar?

RECOMENDACIONES:

Se recomienda realizar primeramente la simulación para esta práctica y también se podría repetir esta práctica en la unidad de circuitos secuenciales, de tal manera que los cuatro bits de entrada sean cambiados automáticamente.

PRÁCTICA No. 8. Manejo de decodificador binario a decimal.

DATOS DE LA PRÁCTICA

CARRERA	ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL			
CUATRIMESTRE	**TERCERO**		**GRUPO**	
TIPO PRÁCTICA	☐ **LABORATORIO** ☐ **SIMULACION**		**FECHA**	
ASIGNATURA	**SISTEMAS DIGITALES**			
UNIDAD TEMÁTICA				
No DE ALUMNOS POR PRÁCTICA	**3**	**No. DE ALUMNOS POR REPORTE**		**3**
NOMBRE DEL PROFESOR				
NOMBRE (S) DE ALUMNO (S)				
TIEMPO ESTIMADO	**2h**	**VO. BO. DEL PROFESOR**		
COMENTARIOS				

OBJETIVO:

Que el alumno comprenda el funcionamiento de un decodificador Binario a decimal.

RECURSOS MATERIALES UTILIZADOS (consumibles):

- 1 TTL 74138.
- 4 Interruptores.
- 4 Resistencias de 10K
- 8 Resistencias de 330.
- 8 led's.

HERRAMIENTAS Y EQUIPO:

- 1 Fuente de 5 Vcc.
- 1 Tablilla de conexiones o protoboard.

DESARROLLO O PROCEDIMIENTO:

En esta práctica se plantea la conexión de un decodificador Binario a decimal.

a) Conectar el diagrama de la Figura A8.1 , donde se muestra la conexión de un 74138 conectado a 8 led's de salida para visualizar el código decimal.

b) Comprobar la tabla de verdad que se muestra en la Figura A8.2

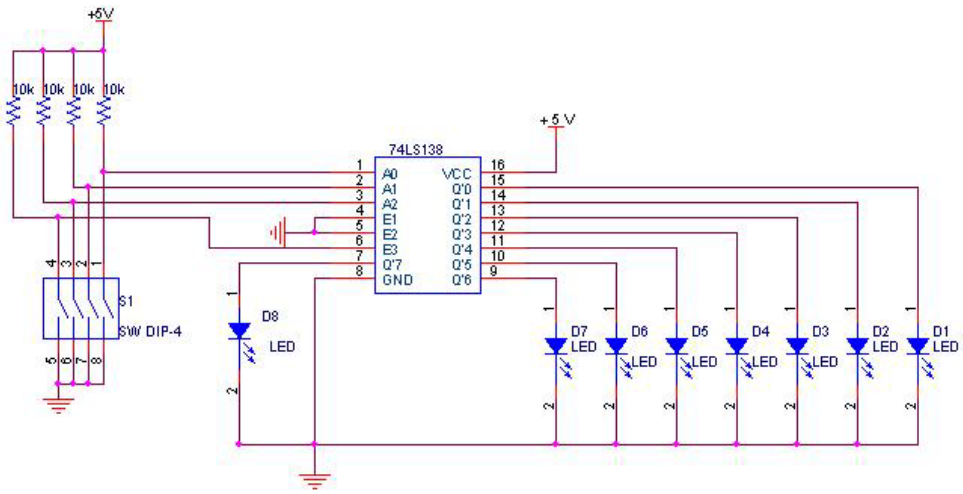


Figura A8.1

Un decodificador es un circuito lógico que acepta un conjunto de entradas que representan números binarios y que activan solamente la salida que corresponde a dicho dato de entrada. donde las salidas son activas en bajo.

código de entrada al 74LS138				código de salida							
E3	A2	A1	A0	Q'0	Q'1	Q'2	Q'3	Q'4	Q'5	Q'6	Q'7
0	X	X	X	1	1	1	1	1	1	1	1
1	0	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	0	1	1	1	1	1	1
1	0	1	0	1	1	0	1	1	1	1	1
1	0	1	1	1	1	1	0	1	1	1	1
1	1	0	0	1	1	1	1	0	1	1	1
1	1	0	1	1	1	1	1	1	0	1	1
1	1	1	0	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	1	0

E3 = entrada de habilitación

X = Condición no importa (el estado)

código de entrada al 74LS138

Figura A8.2

CUESTIONARIO:

- ¿En qué aplicaciones puede ser utilizado el circuito integrado 74138?
- ¿Qué función tiene la entrada E3?

Saber en la Teoría (11 hrs.)

PRÁCTICA No. 9. Contador binario con Flip-Flop's tipo D.

DATOS DE LA PRÁCTICA

CARRERA	ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL			
CUATRIMESTRE	**TERCERO**		**GRUPO**	
TIPO PRÁCTICA	☐ **LABORATORIO** ☐ **SIMULACION**		**FECHA**	
ASIGNATURA	**SISTEMAS DIGITALES**			
UNIDAD TEMÁTICA				
No DE ALUMNOS POR PRÁCTICA	**3**	**No. DE ALUMNOS POR REPORTE**		**3**
NOMBRE DEL PROFESOR				
NOMBRE (S) DE ALUMNO (S)				
TIEMPO ESTIMADO	**2h**	**VO. BO. DEL PROFESOR**		
COMENTARIOS				

OBJETIVO:

Que el alumno comprenda el funcionamiento de los Flip-Flop's, conectados en cascada y de forma asíncrona para un contador binario de 4 bits.

RECURSOS MATERIALES UTILIZADOS (consumibles):

- 1 LM555.
- 2 TTL 74LS74
- 1 capacitor de 10 uF.
- 1 capacitor de 0.01 uF.
- 1 Potenciómetro de 100K.
- 1 Resistencia de 10K.
- 4 Resistencias de 330.
- 4 led's.

HERRAMIENTAS Y EQUIPO:

- 1 Fuente de 5 Vcc.
- 1 Tablilla de conexiones o protoboard.

DESARROLLO O PROCEDIMIENTO:

Conectar primeramente el temporizador 555 como se muestra en la Figura A9.1, que es un dispositivo versátil y muy utilizado, por que puede ser configurado de dos modos distintos, bien como multivibrador monoestable o como multivibrador aestable (oscilador). Un multivibrador aestable no tiene estados estables y varía, por consiguiente, una y otra vez (oscila) entre dos estados inestables, sin ayuda de ningún disparador externo, aquí es importante obtener los pulsos generados en el pin 3, que serán utilizados como señal de reloj para etapas posteriores.

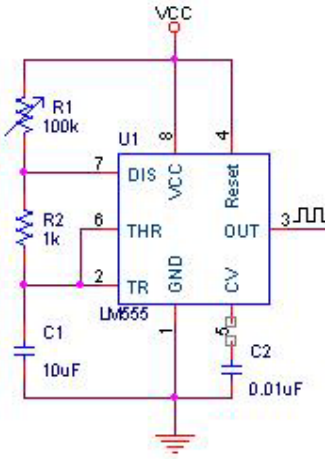


Figura A9.1

Cálculo de la señal de reloj:

Para calcular el tiempo alto de la señal, se realiza mediante la siguiente ecuación:

$$T_A = 1.44(R_1 + R_2)C_1$$

Para calcular el tiempo bajo se puede calcular con la ecuación:

$$T_B = 0.7R_2C_1$$

Por lo tanto el periodo de la señal será:

$$T = T_A + T_B = 0.7(R_1 + 2R_2)C_1$$

Y la frecuencia de la señal de reloj:

$$f = 1/T$$

A continuación conectar el circuito como se muestra en la Figura A9.2, donde se puede observar como van conectados los Flip-Flop's tipo D en cascada y de forma asíncrona para que realice la función de contador binario de 4 bits.

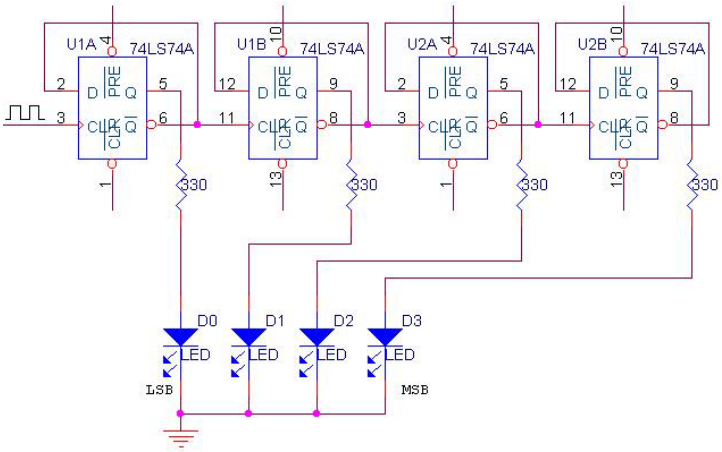


Figura A9.2

CUESTIONARIO:

- ¿De qué manera están conectados los Flip-Flop's tipo D?
- Para un contador binario de 8 bits, ¿Cuántos Flip-Flop's se utilizaran?

RECOMENDACIONES:

1. Para esta práctica de contador binario se recomienda agregar un decodificador 74LS48 ó 74LS47 para poder controlar un display de siete segmentos de cátodo o ánodo común respectivamente, en el display se mostraran las 16 posibles combinaciones secuencialmente.
2. Se puede realizar un contador binario de 8 bits y acoplarlo con la Práctica No. 9 para el surtidor de gasolina, de tal manera que trabaje como un circuito secuencial.

PRÁCTICA No. 10. Control para surtidor de gasolina.

DATOS DE LA PRÁCTICA

CARRERA	ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL			
CUATRIMESTRE	**TERCERO**		**GRUPO**	
TIPO PRÁCTICA	☐ **LABORATORIO** ☐ **SIMULACION**		**FECHA**	
ASIGNATURA	**SISTEMAS DIGITALES**			
UNIDAD TEMÁTICA				
No DE ALUMNOS POR PRÁCTICA	**3**	**No. DE ALUMNOS POR REPORTE**		**3**
NOMBRE DEL PROFESOR				
NOMBRE (S) DE ALUMNO (S)				
TIEMPO ESTIMADO	**2h**	**VO. BO. DEL PROFESOR**		
COMENTARIOS				

OBJETIVO:
Que el alumno compruebe el funcionamiento de los Flip-Flop's.

RECURSOS MATERIALES UTILIZADOS (consumibles):
- 1 CI 4518.
- 2 TTL 7448.
- 2 Display de cátodo común.
- 2 Led (rojo y verde).
- 16 Resistencias de 330.
- 1 interruptor.
- La señal de reloj con el 555.

HERRAMIENTAS Y EQUIPO:
- 1 Fuente de 5 Vcc.
- 1 Tablilla de conexiones o protoboard.

DESARROLLO O PROCEDIMIENTO:
Se plantea el circuito de control para un surtidor de gasolina, de manera que este envié la información de los litros que expende mediante 8 bits codificados en BCD a dos displays de siete segmentos, además al levantar la manguera para servir se enviara por una línea un nivel de

5 voltios y se encenderá un led rojo, pasando a cero al colgar la manguera y encendiendo un led verde.
Se parte de que el surtidor dispone de un circuito que transforma la información de los litros a un código de 8 bits, lo siguiente será almacenar esta información en latchs para decodificarla posteriormente y enviarla a dos displays de siete segmentos. Por otra parte un Flip-Flop tipo T será suficiente para controlar el encendido de los leds, sirviendo como señal de sincronismo los niveles que envía la manguera al ser colgada y descolgada. Por tanto, el circuito será el de la Figura A10.1

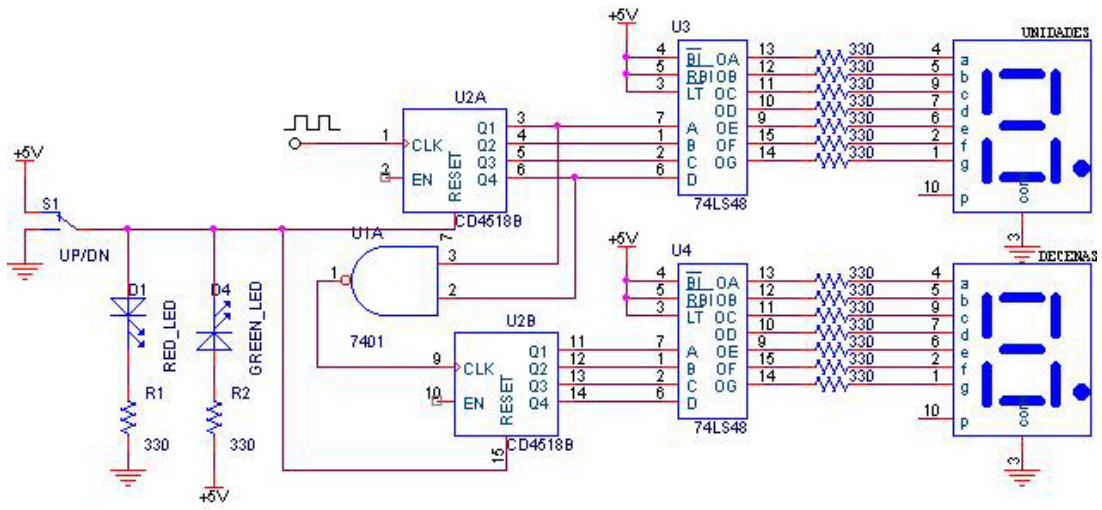


Figura A10.1

CUESTIONARIO:

- ¿De qué manera está conectado el Flip-Flop's tipo JK?
- ¿Los ocho bits de información Cuál es el consumo aproximado de corriente del motor?

RECOMENDACIONES:

Para esta práctica de control de motor a pasos también se recomienda generar la señal de reloj mediante algún otro dispositivo, como puede ser una computadora por medio de algún programa que habilite el puerto paralelo (puede ser lenguaje C).

PRÁCTICA No. 11. Circuito de control de motor a pasos.

DATOS DE LA PRÁCTICA

CARRERA	ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL			
CUATRIMESTRE	**TERCERO**		**GRUPO**	
TIPO PRÁCTICA	☐ **LABORATORIO** ☐ **SIMULACION**		**FECHA**	
ASIGNATURA	**SISTEMAS DIGITALES**			
UNIDAD TEMÁTICA				
No DE ALUMNOS POR PRÁCTICA	**3**	**No. DE ALUMNOS POR REPORTE**		**3**
NOMBRE DEL PROFESOR				
NOMBRE (S) DE ALUMNO (S)				
TIEMPO ESTIMADO	**2h**	**VO. BO. DEL PROFESOR**		
COMENTARIOS				

OBJETIVO:

Que el alumno realice la aplicación de un circuito secuencial para controlar un motor a pasos.

RECURSOS MATERIALES UTILIZADOS (consumibles):

- 1 TTL 7486.
- 1 TTL 7476.
- 1 ULN2003.
- 1 Motor a pasos unipolar.
- 1 diodo 1N4001.
- 1 interruptor.
- La señal de reloj con el 555.

HERRAMIENTAS Y EQUIPO:

- 1 Fuente de 5 Vcc.
- 1 Tablilla de conexiones o protoboard.

DESARROLLO O PROCEDIMIENTO:

Conectar el circuito que se muestra en la Figura A11.1, mediante el interruptor S1 se selecciona el sentido de giro del motor a pasos y la frecuencia de la señal de reloj dará el retardo para cada paso del motor, por lo tanto la velocidad del motor dependerá de esta señal.

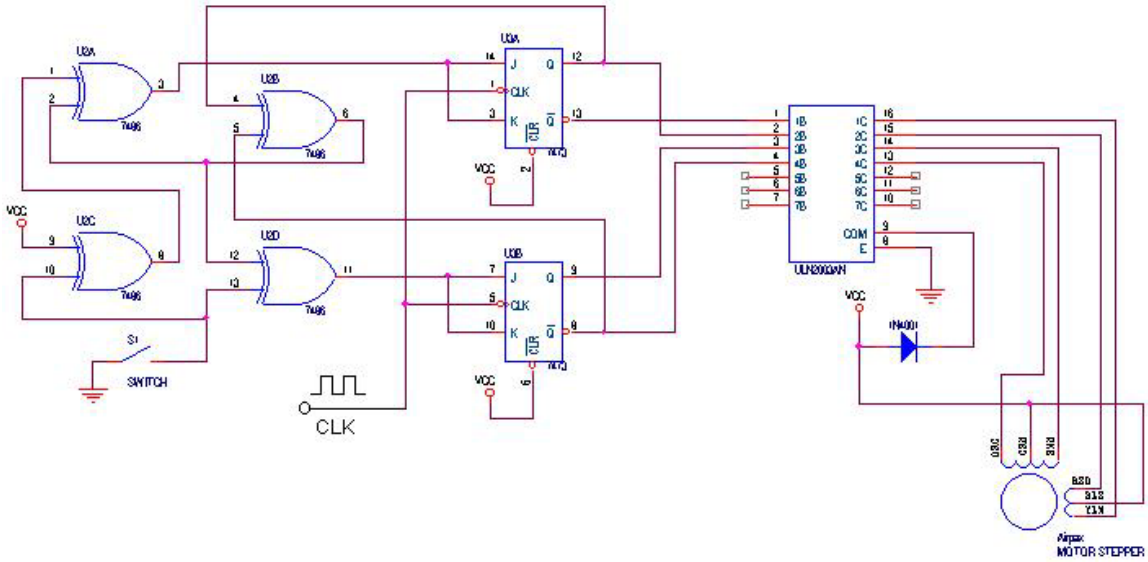


Figura A11.1

CUESTIONARIO:

- ¿De qué manera están conectados los Flip-Flop's tipo JK?
- ¿Cuál es el consumo aproximado de corriente del motor?
- ¿Qué sucede cuando aumenta demasiado la frecuencia de la señal de reloj?

RECOMENDACIONES:

Para esta práctica de control de motor a pasos también se recomienda generar la señal de reloj mediante algún otro dispositivo, como puede ser una computadora por medio de algún programa que habilite el puerto paralelo (puede ser lenguaje C).

PRÁCTICA No. 12. Contador Ascendente - Descendente.

DATOS DE LA PRÁCTICA

CARRERA	ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL			
CUATRIMESTRE	**TERCERO**		**GRUPO**	
TIPO PRÁCTICA	☐ **LABORATORIO** ☐ **SIMULACION**		**FECHA**	
ASIGNATURA	**SISTEMAS DIGITALES**			
UNIDAD TEMÁTICA				
No DE ALUMNOS POR PRÁCTICA	**3**	**No. DE ALUMNOS POR REPORTE**		**3**
NOMBRE DEL PROFESOR				
NOMBRE (S) DE ALUMNO (S)				
TIEMPO ESTIMADO	**2h**	**VO. BO. DEL PROFESOR**		
COMENTARIOS				

OBJETIVO:

Que el alumno comprenda el funcionamiento de los contadores ascendente – descendente, para poder observar en el display los dígitos hexadecimales cambiando de manera ascendente o descendente según se necesite a una frecuencia a la que se puedan observar los cambios"

RECURSOS MATERIALES UTILIZADOS (consumibles):

- 2 TTL 7448.
- 2 TTL 74192.
- 1 TTL 7404.
- 2 TTL 7401.
- 2 Display de cátodo común.
- 1 Interruptor.
- 14 Resistencias de 330.
- La señal de reloj con el 555.

HERRAMIENTAS Y EQUIPO:

- 1 Fuente de 5 Vcc.
- 1 Tablilla de conexiones o protoboard.

DESARROLLO O PROCEDIMIENTO:

Conectar el circuito que se muestra en la Figura A12.1, mediante el interruptor S1 selecciona la señal de reloj para que el conteo sea ascendente o descendente por medio de las dos compuertas NAND.

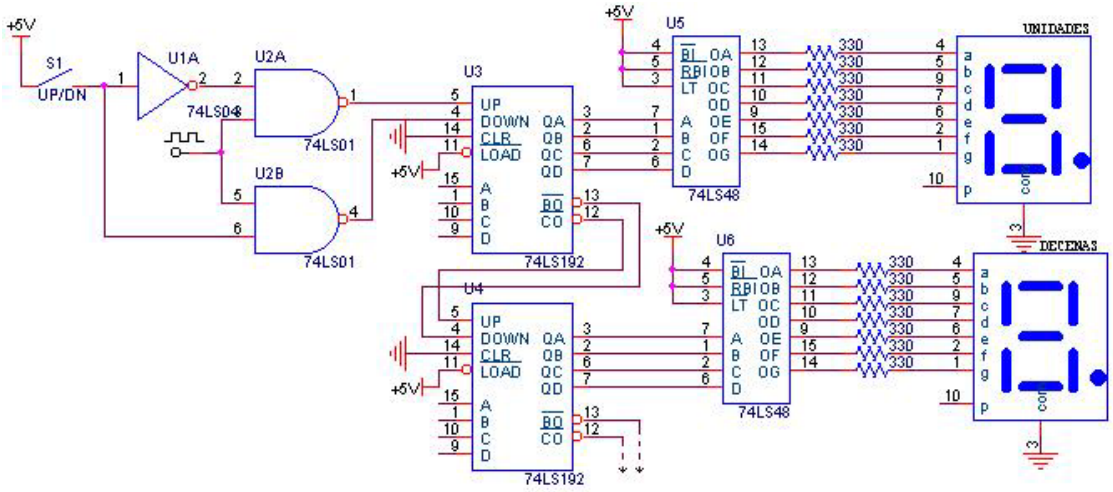


Figura A12.1

CUESTIONARIO:

- ¿Qué función tiene el circuito de entrada formado por las compuertas NAND y el inversor?
- ¿Cuál es la señal que genera el conteo ya sea ascendente o descendente?
- ¿De qué manera se puede ampliar el número de dígitos del contador?

RECOMENDACIONES:

Para esta práctica de conteo en displays de siete segmentos se recomienda acoplarla con la práctica No. 11 para el control del motor a pasos, de tal manera que se cuente el número de pasos del motor, por ejemplo si el motor gira en sentido horario (CW) el contador estará en forma ascendente y si el motor gira en sentido antihorario (CCW) el contador descenderá.

Saber en la Práctica (14 hrs.)

PRÁCTICA No. 13. Dispositivos Lógicos Programables.

DATOS DE LA PRÁCTICA

CARRERA	ELECTRICIDAD Y ELECTRÓNICA INDUSTRIAL			
CUATRIMESTRE	**TERCERO**		**GRUPO**	
TIPO PRÁCTICA	☐ **LABORATORIO** ☐ **SIMULACION**		**FECHA**	
ASIGNATURA	**SISTEMAS DIGITALES**			
UNIDAD TEMÁTICA				
No DE ALUMNOS POR PRÁCTICA	**3**	**No. DE ALUMNOS POR REPORTE**		**3**
NOMBRE DEL PROFESOR				
NOMBRE (S) DE ALUMNO (S)				
TIEMPO ESTIMADO	**2h**	**VO. BO. DEL PROFESOR**		
COMENTARIOS				

OBJETIVO:

Que el alumno comprenda el proceso de programación y grabación de los Dispositivos Lógicos Programables.

RECURSOS MATERIALES UTILIZADOS (consumibles):

- 1 C.I. GAL16V8.
- 2 interruptores
- 2 Resistencias de 10K.
- 8 Resistencias de 330.
- 8 led's.

HERRAMIENTAS Y EQUIPO:

- 1 Fuente de 5 Vcc.
- 1 Tablilla de conexiones o protoboard.

DESARROLLO O PROCEDIMIENTO:

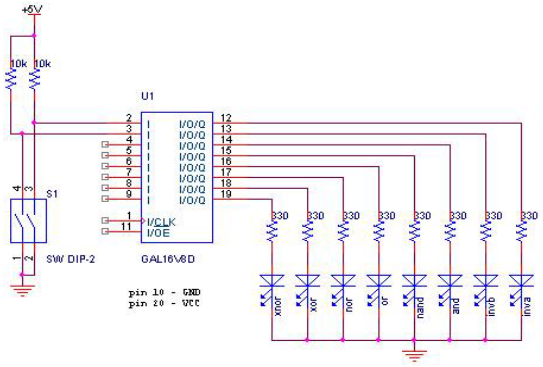
a) Escribir el código fuente en el editor de WinCUPL, dicho código contiene la implementación para que en la GAL se programen las compuertas básicas.

```
Name            Gates;
Partno          CA0001;
Revision        04;
Date            9/12/89;
Designer        G. Woolhiser;
Company         Logical Devices, Inc.;
```

```
Location        None;
Assembly        None;
Device          g16v8a;

/*****************************************************************/
/*                                                               */
/*      This is a example to demonstrate how CUPL                */
/*      compiles simple gates.                                   */
/*                                                               */
/*****************************************************************/
/*
 * Inputs:  define inputs to build simple gates from
 */

Pin 1 =  a;
Pin 2 =  b;

/*
 * Outputs:  define outputs as active HI levels
 */

Pin 12 = inva;
Pin 13 = invb;
Pin 14 = and;
Pin 15 = nand;
Pin 16 = or;
Pin 17 = nor;
Pin 18 = xor;
Pin 19 = xnor;

/*
 * Logic:  examples of simple gates expressed in CUPL
 */

inva = !a;              /* inverters */
invb = !b;
and  = a & b;           /* and gate */
nand = !(a & b);        /* nand gate */
or   = a # b;           /* or gate */
nor  = !(a # b);        /* nor gate */
xor  = a $ b;           /* exclusive or gate */
xnor = !(a $ b);        /* exclusive nor gate */
```

b) Una vez terminado, se tendrá que compilar el código para verificar si no hay errores y poder generar el archivo .jed, que es el necesario para poder grabar el chip con el programador universal.

c) Para verificar que la GAL ha sido programada se tendrá que conectar como se muestra en la siguiente Figura.



Bibliografía

- Lógica Digital y Microprogramable
  Antonio Serna Ruiz, José Vicente García Gil
  Editorial PARANINFO

- Lógica Digital y Diseño de Computadores
  M. Morris Mano
  Editorial PRENTICE HALL

- Circuitos Electrónicos Digitales
  Elias Muñoz Merino
  E.T.S.I.T.