

Materia: MATEMÁTICA DISCRETA Y DISEÑO LÓGICO

PRÁCTICOS DE GABINETE

TRABAJO PRÁCTICO DE GABINETE N°1

Contenido: SISTEMAS DE NUMERACIÓN

Utilidad del contenido y aporte a la formación:

- Interpretar especificaciones técnicas de los componentes presentes en un sistema computacional.

Objetivos:

- Convertir un número de un sistema de numeración (decimal, binario, octal o hexadecimal) a su equivalente en otro.
- Identificar las ventajas de los sistemas de numeración octal y hexadecimal.
- Contar en octal y hexadecimal.

Ejercicios:

INTRODUCCIÓN

Los números se pueden representar en distintos sistemas de numeración que se diferencian entre sí por su base.

Así el sistema de numeración decimal es de base 10, el binario de base 2, el octal de base 8 y el hexadecimal de base 16. El diseño de todo sistema digital responde a operaciones con números discretos y por ello necesita utilizar los sistemas de numeración y sus códigos. En los sistemas digitales se emplea el sistema binario debido a su sencillez.

Cualquier número de cualquier base se puede representar mediante la siguiente ecuación polinómica:

$$N = a_1 \cdot b^n + a_2 \cdot b^{n-1} + a_3 \cdot b^{n-2} + \dots + a_0 \cdot b^0 + a_{-1} \cdot b^{-1} + \dots$$

Siendo b la base del sistema de numeración. Se cumplirá que $b > 1$; a_i es un número perteneciente al sistema que cumple la siguiente condición: $0 \leq a_i < b$.

SISTEMA DECIMAL

Su origen lo encontramos en la India y fue introducido en España por los árabes. Su base es 10.

Emplea 10 caracteres o dígitos diferentes para indicar una determinada cantidad: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. El valor de cada símbolo depende de su posición dentro de la cantidad a la que pertenece. Veámoslo con un ejemplo:

$$136_{10} = 1 \cdot 10^2 + 3 \cdot 10^1 + 6 \cdot 10^0$$

$$136,42_{10} = 1 \cdot 10^2 + 3 \cdot 10^1 + 6 \cdot 10^0 + 4 \cdot 10^{-1} + 2 \cdot 10^{-2}$$

SISTEMA BINARIO

Es el sistema digital por excelencia, aunque no el único, debido a su sencillez. Su base es 2

Emplea 2 caracteres: 0 y 1. Estos valores reciben el nombre de bits (dígitos binarios).

Así, podemos decir que la cantidad 10011 está formada por 5 bits. Veamos con un ejemplo como se representa este número teniendo en cuenta que el resultado de la expresión polinómica dará su equivalente en el sistema decimal:

$$10011_2 = 1 \cdot 10^4 + 0 \cdot 10^3 + 0 \cdot 10^2 + 1 \cdot 10^1 + 1 \cdot 10^0 = 19_{10}$$

SISTEMA OCTAL

Posee ocho símbolos: 0, 1, 2, 3, 4, 5, 6, 7. Su base es 8.

Este sistema tiene una peculiaridad que lo hace muy interesante y es que la conversión al sistema binario resulta muy sencilla ya que, $8 = 2^3$. Así, para convertir un número de base 8 a binario se sustituye cada cifra por su equivalente binario en el apartado 1.5.

Conversiones se estudiará esta conversión.

SISTEMA HEXADECIMAL.

Está compuesto por 16 símbolos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Su base es 16. Es uno de los sistemas más utilizados en electrónica, ya que además de simplificar la escritura de los números binarios, todos los números del sistema se pueden expresar en cuatro bits binarios al ser $16 = 2^4$. La conversión de un número hexadecimal a uno binario es muy sencilla al igual que en el sistema octal, profundizaremos en ello en el apartado 1.5.

CONVERSIONES

CONVERSIÓN ENTRE BINARIO Y DECIMAL

Si la conversión es de binario a decimal, aplicaremos la siguiente regla: se toma la cantidad binaria y se suman las potencias de 2 correspondientes a las posiciones de todos sus dígitos cuyo valor sea 1. Veamos dos ejemplos:

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

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

Si la conversión es de decimal a binario, aplicaremos la siguiente regla: se toma la cantidad decimal dada y se divide sucesivamente entre 2. Los restos obtenidos en cada división (0, 1), forman la cantidad binaria pedida, leída desde el último cociente al

primer resto. Se presentaran los ejemplos en forma de tabla debido a la dificultad que supone utilizar el sistema tradicional de división con el editor:

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

$107_{10} = 1101011_2$

<p>Cuando tengamos un número con decimales seguiremos el siguiente procedimiento: multiplicaremos por 2 la parte decimal y se toma como dígito binario su parte entera. El proceso se repite con la fracción decimal resultante del paso anterior, hasta obtener una fracción decimal nula, o bien hasta obtener el número de cifras binarias que se desee. Ejemplo: 107,645. Como anteriormente convertimos 107 a binario, el resultado de la conversión quedaría así: 1101011, 10100101₂</p>	Fracción decimal	Multiplicado por:	Resultado	Dígito binario
	0,645	2	1,290	1
	0,290	2	0,580	0
	0,580	2	1,160	1
	0,160	2	0,320	0
	0,320	2	0,64	0
	0,64	2	1,28	1
	0,28	2	0,56	0
	0,56	2	1,12	1

CONVERSIÓN ENTRE OCTAL Y BINARIO

Si la conversión es de octal a binario cada cifra se sustituirá por su equivalente binario. Tendremos en cuenta la siguiente tabla para hacer la conversión de modo más rápido:

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

Ejemplo: 55,35₈

Resultado: 101 101, 011 101₂

Si la conversión es de binario a octal se realiza de modo contrario a la anterior conversión, agrupando los bits enteros y los fraccionarios en grupos de 3 a partir de la

coma decimal. Si no se consiguen todos los grupos de tres se añadirán, los ceros que sean necesarios al último grupo, veámoslo con un ejemplo:

Ejemplo: 11011111,11111 ₂ Resultado: 237,76 ₈ Observa como ha sido necesario añadir un cero en la última agrupación de la parte entera y otro en la parte fraccionaria para completar los grupos de 3 dígitos.	Agrupación	Equivalente octal
	010	2
	011	3
	111	7
	,	,
	111	7
	110	6

CONVERSIÓN ENTRE OCTAL Y DECIMAL

Si la conversión es de octal a decimal se procederá como observas en el ejemplo:

$$740_8 = 7 \cdot 8^2 + 4 \cdot 8^1 + 4 \cdot 8^0 = 484_{10}$$

Si la conversión es de decimal a octal se procederá de modo similar a la conversión de decimal a binario, pero dividiendo entre 8. Comprueba los resultados en el siguiente ejemplo:

$$426_{10} = 652_8$$

CONVERSIÓN ENTRE BINARIO Y HEXADECIMAL

La conversión entre binario y hexadecimal es igual al de la conversión octal y binario, pero teniendo en cuenta los caracteres hexadecimales, ya que se tienen que agrupar de 4 en 4. La conversión de binario a hexadecimal se realiza según el ejemplo siguiente:

Sistema binario	Sistema Hexadecimal	Ejemplo: 1011111,110001 ₂ Agrupando obtenemos el siguiente resultado: 0101 1111, 1100 0100 ₂ Sustituyendo según la tabla logramos la conversión esperada: 5F, C4 ₁₆
0000	0	
0001	1	
0010	2	
0011	3	
0100	4	
0101	5	
0110	6	
0111	7	
1000	8	
1001	9	
1010	A	
1011	B	

1100	C	
1101	D	
1110	E	
1111	F	

La conversión de hexadecimal a binario simplemente sustituiremos cada carácter por su equivalente en binario, por ejemplo:

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

Ejercicios

EJERCICIO 1:

Convertir a decimal los siguientes números.

- | | | |
|-------------------|---------------|---------------------|
| a) 1101010011_2 | b) $1BF_{16}$ | c) $111101,10101_2$ |
| d) 232_8 | e) $575,54_8$ | f) $2CD,5_{16}$ |

EJERCICIO 2:

Convertir a octal los siguientes números.

- | | | |
|---------------------|--------------------|-------------------|
| a) 1382_{10} | b) $7523,236_{10}$ | c) 11101001_2 |
| d) $1011011,1011_2$ | e) $45BA_{16}$ | f) $DCBA,2F_{16}$ |

EJERCICIO 3:

Convertir a hexadecimal los siguientes números.

- | | | |
|----------------------|----------------|----------------|
| a) $1001101,10011_2$ | b) 674_8 | c) $5272,32_8$ |
| d) 8324_{10} | e) 4545_{10} | f) 8963_{10} |

EJERCICIO 4:

Convertir a binario los siguientes números.

- | | | |
|----------------|------------------|-----------------|
| a) 1578_{10} | b) $359,75_{10}$ | c) $544,24_8$ |
| d) $637,43_8$ | e) $DAB,B2_{16}$ | f) $EC,9B_{16}$ |



Materia: MATEMÁTICA DISCRETA Y DISEÑO LÓGICO

PRÁCTICOS DE GABINETE

TRABAJO PRÁCTICO DE GABINETE N°2

Contenido: CÓDIGOS

Utilidad del contenido y aporte a la formación:

- Interpretar especificaciones técnicas de los componentes presentes en un sistema computacional.

Objetivos:

- Representar números decimales utilizando el código BCD. Mencionar las ventajas y las desventajas de utilizar BCD.
- Comprender la diferencia entre el código BCD y el binario directo.
- Comprender el propósito de los códigos alfanuméricos, como el ASCII.
- Explicar el método de paridad para detectar errores.
- Determinar el bit de paridad que se anexará a una cadena de información.

CÓDIGOS BINARIOS

Debido a la naturaleza biestable de los circuitos de electrónica digital, estos solo procesan códigos que constan de 0 y 1 (códigos binarios) existen muchas situaciones en la electrónica digital en la que necesitamos realizar tareas específicas, por lo tanto se necesitarán utilizar una serie de códigos que también utilizan ceros (0) y unos (1), pero sus significados pueden variar. A continuación detallaremos estos tipos de códigos.

CÓDIGOS BINARIOS CON PESO

Supongamos que queremos transformar el número decimal 89532 a su correspondiente equivalencia en binario, aplicando el método de la división sucesiva por dos, llegaremos al siguiente resultado: **10101110110111100** pero para llegar a este resultado seguro te tomará cierto tiempo y trabajo, de igual forma si queremos diseñar un sencillo circuito digital en el que la cifra introducida en el teclado sea visualizada en la pantalla, se necesitarían una gran cantidad de compuertas lógicas para construir el circuito decodificado y codificador. Los códigos binarios con peso nos resuelven este problema pues estos códigos fueron diseñados para realizar la conversión de decimal a binario de una manera mucho más fácil y rápida.

CÓDIGOS BCD

Los códigos BCD (Binary Coded Decimal) (Decimal Codificado en Binario) son grupos de 4 bits en el cual cada grupo de 4 bits solo puede representar a un único dígito decimal (del 0 al 9) Estos códigos son llamados códigos con peso ya que cada bit del grupo posee un peso o valor específico. Existen por lo tanto códigos BCD's de acuerdo al valor o peso que posea cada bit. Ejemplos de estos códigos son el BCD 8421, el BCD 4221, el BCD 5421, el BCD 7421, el BCD 6311, etc. donde la parte numérica indica el peso o valor de cada bit. Así por ejemplo el código BCD 8421 nos indica que el MSB posee un valor de 8, el segundo MSB posee un valor de 4, el tercer MSB tiene un valor de 2 y el LSB tiene un valor de 1. Para el código BCD 6311 el MSB tiene un peso o valor de 6, el segundo MSB posee un peso de 3, el tercer MSB posee un valor de 1, y el

LSB tiene un valor de 1. El código BCD 8421 es el código BCD mas utilizado, es común referenciarlo simplemente como código BCD, así en el transcurso del curso se entenderá el código BCD como el BCD 8421, a menos que se indique lo contrario.

CONVERSIÓN DE DECIMAL A BCD

Ya que cada grupo de 4 bits solo puede representar a un único dígito decimal, la conversión de un numero decimal a un numero BCD se lleva a cabo de la siguiente forma:

1. Separamos al dígito decimal en cada uno de sus dígitos
2. Cada dígito decimal se transforma a su equivalente BCD.
3. El número obtenido es el equivalente en BCD del número decimal.

Por ejemplo, para convertir el decimal 469 a BCD, según lo explicado anteriormente, tenemos que tomar cada dígito decimal y transformarlo a su equivalente BCD.

4	6	9
↓	↓	↓
0100	0110	0011

Figura 1: Conversión de decimal a BCD

De esta forma el decimal 469 equivale al BCD 010001100011

NOTA: En BCD los códigos 1010, 1011, 1100, 1101 y 1111 no tienen decimales equivalentes. Por lo tanto se les llaman códigos inválidos

CONVERSIÓN DECIMAL FRACCIONARIO A BCD

Se realiza del modo similar al anterior pero hay que tener en cuenta el punto binario, el punto del numero decimal se convertirá en el punto binario del código BCD.

Ejemplo: para convertir el decimal 74.42 a BCD:

Separamos el decimal en sus dígitos 7 4 . 4 2.

Convertimos cada dígito a decimal a BCD, y colocamos el punto binario en la misma posición del punto decimal.

7	4	.	4	2
↓	↓	↓	↓	↓
0111	0100	.	0100	0010

Figura 2: Conversión de decimal fraccionario a BCD

De esta forma el decimal 74.42 equivale al BCD 01100100. 010000101.

CONVERSIÓN DE BCD A DECIMAL

Ya que el código BCD son grupos de 4 bits, realizaremos lo siguiente:

1. A partir de la izquierda separamos al número BCD en grupos de 4 bits.
2. Cada grupo de 4 bits se convierte a su decimal correspondiente.
3. El número obtenido es el equivalente decimal del número BCD.

Ejemplo: Convertir el número BCD 010101000011 a decimal.

Separamos en grupos de 4 bits a partir de la izquierda 0101 0100 0011.

Transformamos cada grupo a decimal.

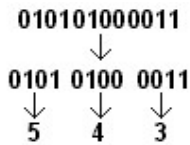


Figura 3: Conversión de BCD a decimal.

El BCD 010101000011 equivale al decimal 543

CONVERSIÓN BCD FRACCIONARIO A DECIMAL

1. A partir del punto binario separamos al número binario en grupos de 4 bits.
2. Cada grupo de 4 bits se convierte a su equivalente decimal.
3. El punto binario se convertirá en el punto decimal.
4. El número obtenido equivale en decimal al número BCD.

Ejemplo: Convertir el número BCD 01110001.0000100 a decimal.

separamos en grupo de 4 bits 0111 0001. 0000 1000.

convertimos cada grupo a decimal y colocamos el punto binario como punto decimal.

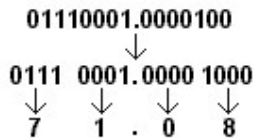


Figura 4: Conversión de BCD fraccionario a decimal.

El BCD 01110001.00001000 equivale al decimal 71.08

CONVERSIÓN BCD A BINARIO PURO

Si queremos transformar un numero BCD a su correspondiente binario llevaremos a cabo los siguientes pasos:

1. El número BCD lo transformamos a decimal.
2. Convertimos el decimal obtenido a binario mediante las técnicas ya estudiadas.
3. El binario obtenido es el equivalente en binario del número BCD.

Ejemplo: Convertir el BCD 000100000011.0101 a binario.

Convertimos 0001 0000 0011. 0101 a decimal 1 0 3. 5.

Transformamos el decimal a binario $103.5(10)=1100111.1$

CONVERSIÓN DE BINARIO PURO A BCD

1. Convertimos el número binario a número decimal.
2. Cada dígito decimal se convierte a su equivalente BCD.
3. El numero obtenido es el equivalente BCD del número binario puro.

Ejemplo: convertir el binario 10001010.101 a BCD

Se convierte primero a decimal 10001010.101

$$128 + 8 + 2 + 0.5 + 0.125 = 138.625.$$

convertimos el decimal a BCD

$$138.625 = 0001\ 0011\ 1000.\ 0110\ 0010\ 0101$$

El binario 10001010.101 es igual al BCD 000100111000.011000100101

NOTA: Seguramente ha notado que los números en código BCD contienen mayor número de bits que sus correspondientes números binarios, pero nuevamente recalamos que esta desventaja es compensada por su facilidad para convertir a decimal.

CÓDIGOS BINARIOS SIN PESO

De la misma forma que existen códigos binarios con peso, también existen códigos binarios sin peso en el cual cada bit no va a poseer un valor o ponderación por posición. Aquí detallaremos dos códigos binarios sin peso: el de exceso 3 y el código Gray.

CÓDIGO DE EXCESO 3

A pesar de ser un código binario sin peso, el código de exceso 3 guarda una estrecha relación con el código BCD 8421 por el hecho de que cada grupo de 4 bits solo pueden representar a un único dígito decimal (del 0 al 9), y deriva su nombre de exceso 3 debido a que cada grupo de 4 bits equivale al número BCD 8421 mas 3.

CONVERSIÓN DE DECIMAL A EXCESO 3

1. Se separa al número decimal en cada uno de sus dígitos.
2. Sumarle tres (3) a cada dígito decimal.
3. Convertir a BCD el número decimal obtenido.
4. El número obtenido es el equivalente en XS3 del número decimal.

Ejemplo: convertir el número decimal 18 a su equivalente XS3.

Solución: primero le sumamos 3 a cada dígito.

$$\begin{array}{r} 1 \quad 8 \\ + 3 \quad + 3 \\ \hline 4 \quad 11 \end{array}$$

luego cada resultado se transforma a BCD

$$4 = 0100$$

$$11 = 1001$$

Nota: En las conversiones de exceso 3 no se tiene en cuenta los códigos inválidos (1010, 1011, 1100, 1101, 1110, 1111) como vimos en el ejemplo anterior el número 11, el cual nos resultó de la suma de 8+3, se convirtió directamente al BCD 1001.

CONVERSIÓN BCD A XS3

Para convertir un número BCD a código de exceso 3 tenemos en cuenta los siguientes pasos:

1. A partir de la izquierda separamos al código BCD en grupos de 4 bits.
2. Sumamos 0011₂ a cada grupo de 4 bits.

3. El resultado es el equivalente en XS3 del código BCD.

Ejemplo: Convertir el BCD 00101001 a XS3

Separamos en grupos de bits. 0010 1001

Sumamos 0011_2 a cada grupo

$$\begin{array}{r} 0010 \quad 1001 \\ + 0011 \quad +0011 \\ \hline 0101 \quad 1100 \end{array}$$

El código XS3 01011100 equivale al BCD 00101001

CONVERSIÓN DE XS3 A DECIMAL

1. Dividimos a partir de la izquierda al número XS3 en grupos de 4 bits.
2. Convertimos a decimal cada grupo de 4 bits.
3. Restamos 3 a cada decimal.
4. El número obtenido es el equivalente decimal del número XS3.

Ejemplo : Convertir 10011010_{XS3} a decimal

Separamos en 4 bits 1001 1010

Convirtiendo a decimal 1001 1010

$$1001 = 9$$

$$1010 = 10$$

restamos 3 a cada resultado

$$\begin{array}{r} 9 \quad 10 \\ -3 \quad -3 \\ \hline 6 \quad 7 \end{array}$$

el número 67_{10} equivale al XS3 10011010

CÓDIGO GRAY

Observemos lo siguiente:

El decimal 5 se representa en binario por 0101

El decimal 6 se representa en binario por 0110

¿Qué has notado?

Observa que con solo aumentar un nivel en la cuenta (del 5 al 6) dos bits cambiaron de estado (el tercer MSB y el LSB de ambos números), probablemente esto no signifique nada ni nos afectaría en lo mas mínimo sin embargo existen algunas situaciones en electrónica digital en el cual solo necesitamos que al incrementarse la cuenta en un nivel solo cambie de estado (de 0 a 1 o viceversa) uno y únicamente un solo bit.

La solución esta en el código Gray, un código binario sin peso que no tiene ninguna relación con el código BCD.

Así para el ejemplo que hemos venido analizando:

el decimal 5 en binario es 0101 y en código Gray es 0 1 **1** 1

el decimal 6 en binario es 0110 y en código Gray es 0 1 **0** 1

el color azul indica el bit que cambió de estado.

Pero, ¿cuales son los pasos que se deben llevar cabo para hacer la transformación a código Gray?

CONVERSIÓN DE NUMERO BINARIO A CÓDIGO GRAY

1. El MSB del numero binario será el mismo para el código Gray.
2. Sumar el MSB del numero binario al bit situado a su derecha inmediata y anotar el resultado del numero en código Gray que estamos formando.
3. Continuar sumando bits a los bits situados a la derecha y anotando las sumas; hasta llegar al LSB.
4. El número en código Gray tendrá el mismo número de bits que el número binario.

Ilustraremos mejor esta explicación con un ejemplo:

Ejemplo: convertir el numero binario 0010 a código Gray

1. 0 0 1 0 → binario
↓
0 → gray

2. 0 0 1 0 → binario
0 + 0 = 0
0 0 → gray

3. 0 0 1 0 → binario
0 + 1 = 1
0 0 1 → gray

4. 0 0 1 0 → binario
1 + 0 = 1
0 0 1 1 → gray

Aquí finaliza la conversión dado que ya llegamos al LSB del numero binario.

Entonces el numero binario 0010 equivale al 0011 en código Gray

CONVERSIÓN DE CÓDIGO GRAY A BINARIO

1. El bit izquierdo de código Gray será el MSB del numero binario.
2. El bit obtenido es sumado al segundo bit de la izquierda del código Gray, y el resultado se anotara a la derecha del numero binario a formar.
3. Este resultado se le suma al bit situado a la derecha inmediata del ultimo bit que sumamos y el resultado será el otro bit del número binario (se ordena de izquierda a derecha).
4. Repetir el paso anterior hasta llegar al bit mas a la derecha del código Gray.
5. El número de bits del numero binario deberá coincidir con el número de bits del número en código Gray.

Ejemplo: convertir el número en código Gray 1001 a número binario

1. **1 0 0 1** → gray

1 → binario

2. **1 0 0 1** → gray

1 + 0 = 1

1 1 → binario

3. **1 0 0 1** → gray

1 + 0 = 1

1 1 1 → binario

4. **1 0 0 1** → gray

1 + 0 = 1

1 1 1 0 → binario

CÓDIGOS ALFANUMÉRICOS

Los códigos estudiados anteriormente sólo sirven para representar números, pero ; ¿y si queremos representar las letras del alfabeto o algunos símbolos? ; ¿cómo lo haríamos?.

La solución está en los códigos alfanuméricos, que no es más que un tipo de código diseñado especialmente para representar números, letras del alfabeto (mayúsculas y minúsculas), símbolos especiales, signos de puntuación y unos caracteres de control.

Un código alfanumérico muy popular y ampliamente utilizado, es el llamado [código ASCII](#) (American Standard Code for Information Interchange), que en español quiere decir: código estándar americano para el intercambio de información, el cual es un código de siete bits muy utilizado en los sistemas digitales avanzados (computadores, redes de transmisión de datos, etc.) para representar hasta 128 (2^7) piezas de información diferentes, incluyendo letras, números, signos de puntuación, instrucciones y caracteres especiales.

Ejercicios:

EJERCICIO 1:

Realizar la tabla de un código Jhonson de 6 bits. Indique que características presenta este código.

EJERCICIO 2:

Completar el cuadro, según los códigos indicados para la codificación de los números decimales enunciados. ¿Cuáles de los códigos son auto complementarios?.

Decimal	BCD 2421	BCD EXC3	BCD 3421	BCD 5421
7				
23				

67
81
95
104
237
982

EJERCICIO 3:

Realice la tabla del código Hamming para la detección y corrección de un bit, tomando como código base de información el BCD 3421.

	$2^p \geq n + p + 1$	$p = \text{n}^\circ \text{ de bits a agregar en el nuevo código}$					
3 4 2 1		$n = \text{n}^\circ \text{ de bits del código 3421 (4)}$					
0 0 0 0	$2^3 \geq 4 + 3 + 1 = 8$	los p bits los desarrollamos en binario natural					
0 0 0 1	$C_3 C_2 C_1$						
0 0 1 0	0 0 0						
0 0 1 1	b_1 0 0 1	los b_n equivalen a los bit del nuevo código y están					
0 1 0 0	b_2 0 1 0	codificados en binario natural					
0 1 0 1	b_3 0 1 1	los c_n se los denomina bits correctores de error y se					
0 1 1 0	b_4 1 0 0	calculan de la siguiente manera					
0 1 1 1	b_5 1 0 1						
1 1 0 1	b_6 1 1 0	$c_1 = b_1 \oplus b_3 \oplus b_5 \oplus b_7$					
1 1 1 0	b_7 1 1 1	$c_2 = b_2 \oplus b_3 \oplus b_6 \oplus b_7$					
		$c_3 = b_4 \oplus b_5 \oplus b_6 \oplus b_7$					
ahora considerando que no hay error al recibir el código nuevo podemos decir que c_1 , c_2 y c_3 valen cero y así debido a una propiedad de la or exclusiva podemos extrapolar b_1 , b_2 y b_4							
	$b_1 = b_3 \oplus b_5 \oplus b_7$						
	$b_2 = b_3 \oplus b_6 \oplus b_7$						
	$b_4 = b_5 \oplus b_6 \oplus b_7$						
donde ya tenemos calculados los 3 bits que agregamos al nuevo código; ahora confeccionamos el código de Hamming completo							
	b_7	b_6	b_5	b_4	b_3	b_2	b_1
	0	0	0	0	0	0	0
	0	0	0	0	1	1	1
	0	0	1	1	0	0	1
	0	0	1	1	1	1	0
	0	1	0	1	0	1	0
	0	1	0	1	1	0	1
	0	1	1	0	0	1	1
	0	1	1	0	1	0	0
	1	1	0	0	1	1	0
	1	1	1	1	0	0	0



EJERCICIO 4:

Indicar las distintas combinaciones binarias asignadas a cada uno de los siguientes números, caracteres ó símbolos especiales, en el código ASCII de 7 bits.

0 ; % ; , ; G ;) ; 3 ; + ; & ; . ; T ; ¿ ; ' ;

EJERCICIO 5:

Escribir los siguientes números decimales en BCD natural y sumar:

- a) 36921
- b) 5748

EJERCICIO 6:

Escribir los siguientes números decimales en BCD natural, BCD exceso tres y código Aiken

- a) 357
- b) 6002

EJERCICIO 7:

Encontrar el complemento a dos de los siguientes números binarios:

- a) 1010101
- b) 0011010
- c) 11100111
- d) 10011000

EJERCICIO 8:

Encontrar el complemento a uno de los siguientes números binarios:

- a) 11010
- b) 10111
- c) 00001100
- d) 10000100

Materia: MATEMÁTICA DISCRETA Y DISEÑO LÓGICO

PRÁCTICOS DE GABINETE

TRABAJO PRÁCTICO DE GABINETE N°3

Contenido: MINIMIZACIÓN DE FUNCIONES

Utilidad del contenido y aporte a la formación:

- Representar, operar e implementar funciones lógicas

Objetivos:

- Convertir una expresión lógica en una suma de productos.
- Llevar a cabo los pasos necesarios para reducir una suma de productos a su forma más simple.
- Utilizar el álgebra booleana y el mapa de Karnaugh como herramientas para simplificar.

Ejercicios:

Introducción:

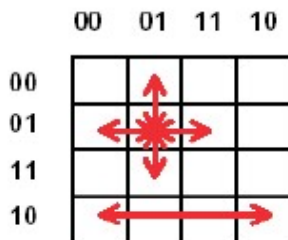
Escribir el mapa de la función:

$$F(A,B,C,D) = \Sigma(0, 1, 5, 6, 9, 13, 15)$$

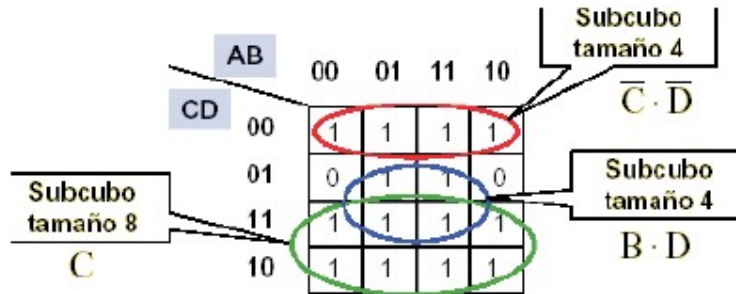
AB		00	01	11	10
CD	00	1	0	0	0
	01	1	1	1	1
	11	0	0	1	0
	10	0	1	0	0

Dos celdas se diferencian entre si por una sola variable, no hay un cambio de dos o mas unos ó ceros entre celdas contiguas

	00	01	11	10
00				
01				
11				
10				

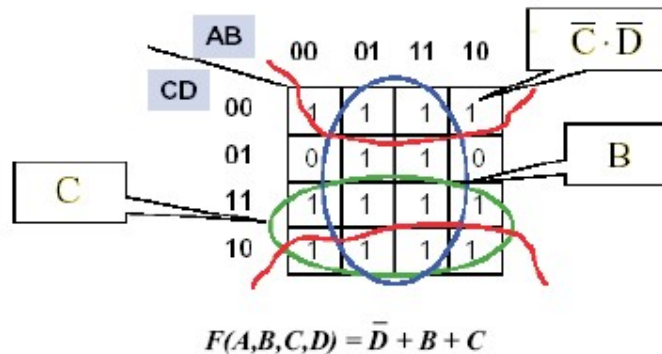


Un subcubo es un conjunto de 2^m celdas con valor 1, las cuales tienen la propiedad que cada celda es contigua a m celdas del conjunto



Una función se puede expresar como la suma de los subcubos necesarios para cubrir todos los unos del mapa. Para que una función sea mínima, hay que buscar el mínimo número de subcubos posibles, esto se logra cuando estos tienen el mayor tamaño posible, incluso si se repiten elementos ya seleccionados.

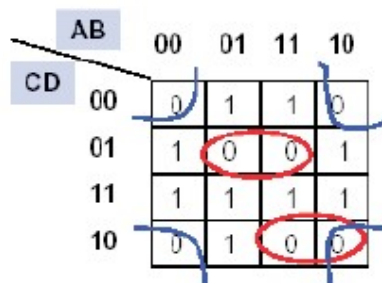
Una mejor agrupación de los subcubos sería la siguiente:



Siguiendo las leyes de Morgan podemos también expresar la función como productos de sumas, cuando sea conveniente, agrupando los ceros.

$$F(A,B,C,D) = \Pi(0,2,5,8,10,13,14)$$

Para minimizar la función, se agrupan los ceros



$$F(A,B,C,D) = (B + D) \cdot (\bar{B} + C + \bar{D}) \cdot (\bar{A} + \bar{C} + D)$$



1. Minimizar las siguientes funciones

a) $f(a,b,c) = \Sigma 0,2,4,6$

b) $f(a,b,c) = \Sigma 1,3,4,5,6,7$

c) $f(a,b,c,d) = \Sigma 0,1, 2,3,9,12,13,15$

d) $f(a,b,c,d) = \Sigma 5,7,8,9,11,12, 13,14,15$

e) $f(a,b,c,d,e) = \Sigma 0,1,4,5,8,9,10,11,14,15,16,22,24,25,26,28,29,30$

f) $f(a,b,c,d,e) = \Sigma 0,2,4,6,7,9,11,12,13,14,15,,20,21,22,23,24,25,29,30,31$

1.

Materia: MATEMÁTICA DISCRETA Y DISEÑO LÓGICO

PRÁCTICOS DE GABINETE

TRABAJO PRÁCTICO DE GABINETE N°4

Contenido: DISEÑO DE CIRCUITOS COMBINACIONALES

Utilidad del contenido y aporte a la formación:

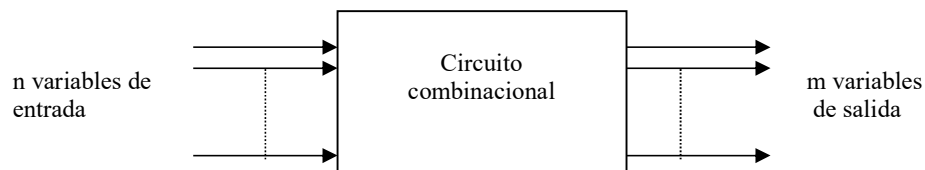
- Conocer las aplicaciones de los circuitos combinacionales.
- Identificar los componentes funcionales de los circuitos lógicos

Objetivos:

- Utilizar el álgebra booleana y el mapa de Karnaugh como herramientas para simplificar y diseñar circuitos lógicos.
- Diseñar circuitos lógicos a partir de una tabla de verdad.

Circuitos combinacionales

Un circuito combinacional es un arreglo conectado de compuertas lógicas, con un conjunto de entradas y salidas. En cualquier tiempo dado, los valores binarios de las salidas son una función de la combinación de unos y ceros de las entradas. El siguiente es diagrama en bloques de un circuito combinacional.



Un circuito combinacional transforma información binaria de datos de entrada dados a datos de salida requeridos. Los circuitos de combinación son utilizados en los sistemas digitales para generar decisiones de control binario o para proporcionar funciones digitales que se requieren en el procesamiento de datos. Muchas funciones digitales se encuentran disponibles comercialmente en circuitos integrados.

Un circuito combinacional puede describirse por una tabla de la verdad que muestra las relaciones binarias entre las n variables de entrada y las m variables de salida. Hay 2^n combinaciones posibles de los valores de las entradas binarias. La tabla de la verdad lista los valores binarios de la salida correspondientes para cada una de las 2^n combinaciones de entrada.

Procedimiento de diseño

El diseño de los circuitos combinacionales comienza con la descripción del problema y termina en un diagrama de circuito lógico. El procedimiento comprende los siguientes pasos:

1. Se enuncia el problema.
2. A las variables de entrada y de salida se les asignan símbolos de letras.
3. Se obtiene la tabla de la verdad que define las relaciones entre entradas y salidas.
4. Las funciones booleanas simplificadas se obtienen para cada una de las salidas.
5. Se dibuja el diagrama lógico.

Los siguientes son ejemplos de diseño de circuitos combinacionales. Los mismos son circuitos aritméticos digitales. Estos circuitos sirven como bloques básicos para la implementación de funciones digitales aritméticas más complejas.

Semi-sumador

La función digital más básica es la suma de dos dígitos binarios. Un circuito combinacional que realiza esta suma aritmética de dos bits, se denomina un semi-sumador. Uno que realiza la suma de tres bits (dos bits y uno de acarreo) se denomina sumador completo.

Las variables de entrada de un semi-sumador se denominan bit sumando y bit sumador. Las variables de salida se denominan bit de suma y bit de acarreo. Es necesario especificar dos variables de salida ya que la suma de 1+1 es el binario 10. Si asignamos los símbolos x e y las dos variables de entrada, y S para la suma y C para el acarreo; la tabla de la verdad para el semi-sumador será:

x	y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Las funciones booleanas para las dos salidas se obtienen directamente de la tabla de la verdad:

$$S = \bar{x} \cdot y + x \cdot \bar{y} = x \oplus y$$

$$C = x \cdot y$$

Sumador completo

Los pasos para el diseño de un sumador completo serán:

1. Se enuncia el problema.

El sumador completo es un circuito combinacional que realiza la suma aritmética de tres bits. Tiene tres variables de entrada (dos bits que representan los bits significativos que se suman y un bit que representa el acarreo de la posición significativa previa más baja) y dos variables de salida (un bit para la suma y otro para el acarreo).

2. A las variables de entrada y de salida se les asignan símbolos de letras.

Las dos variables de entrada que representan los bits que se suman se denominan x e y . La variable de entrada que representa el acarreo previo se denomina z .

Las variables de salida serán S y C , que representan la suma y el acarreo respectivamente.

3. Se obtiene la tabla de la verdad que define las relaciones entre entradas y salidas.

La tabla de la verdad será entonces:

entradas			Salidas	
x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

4. Las funciones booleanas simplificadas se obtienen para cada una de las salidas.

Las funciones booleanas simplificadas serán:



$$S = x \oplus y \oplus z$$

$$C = x.y + x.z + y.z$$

5. Se dibuja el diagrama lógico.

Ejercicios:

Ejercicio 1:

Diseñe un sistema combinacional que accione el motor de un limpiaparabrisas sabiendo que éste funciona cuando la llave general del auto está en contacto, y además, se activa el control del limpiaparabrisas, pero si desactiva este último, el motor sigue funcionando hasta que las escobillas lleguen al costado izquierdo.

Obtener la solución más simple, e implementarla con las compuertas correspondientes.

Ejercicio 2:

Realizar el diseño e implementación de dos llaves de luz colocadas al pie y cima de una escalera; de tal manera que pueda prender la luz de la llave de abajo y apagarla con la llave de arriba y viceversa.

Ejercicio 3:

Realizar el diseño de un comparador de dos números de dos bits c/u. Este sistema tiene que tener 3 salidas la de mayor, menor e igual. Realizar el diseño con compuertas nand.

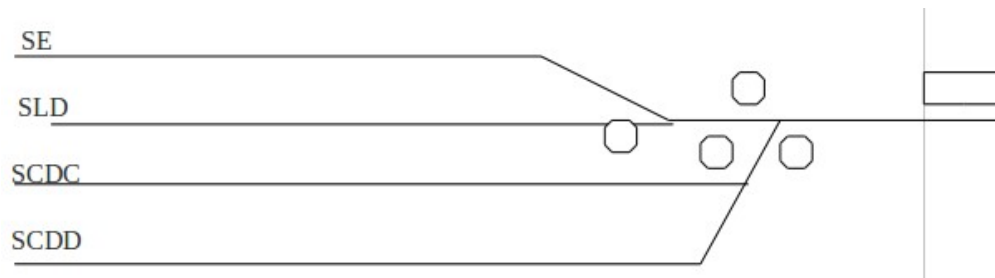
Ejercicio 4:

Diseñe un sistema digital de semaforización para una estación terminal de ferrocarril. Como se observa en la figura, a dicha estación acceden 4 líneas de diferentes tipos de servicios con las siguientes prioridades:

- 1) Servicio **E**xpreso (SE)
- 2) Servicio **C**orta **D**istancia **D**iferencial (SCDD)
- 3) Servicio **C**orta **D**istancia **C**omún (SCDC)

- Servicio Larga Distancia. (SLD)

Sensores en cada línea indicarán ó no la presencia de vehículos, y semáforos en rojo ó en verde detendrán ó habilitarán el paso respectivamente. Obtener la solución más simple, e implementarla con las compuertas correspondientes.



Materia: MATEMÁTICA DISCRETA Y DISEÑO LÓGICO

PRÁCTICOS DE GABINETE

TRABAJO PRÁCTICO DE GABINETE N°5

Contenido: DISEÑO DE CIRCUITOS SECUENCIALES

Utilidad del contenido y aporte a la formación:

- Conocer las aplicaciones de los circuitos y secuenciales
- Identificar los componentes funcionales de los circuitos lógicos

Objetivos:

- Dibujar los diagramas de tiempos de las formas de onda de salida para varios tipos de flip-flops, en respuesta a un conjunto de señales de entrada.
- Utilizar los diagramas de transición de estados para describir la operación de un contador.
- Emplear flip-flops en circuitos de sincronización.
- Utilizar flip-flops como circuitos de conteo y divisores de frecuencia.

Ejemplo de diseño con FF D

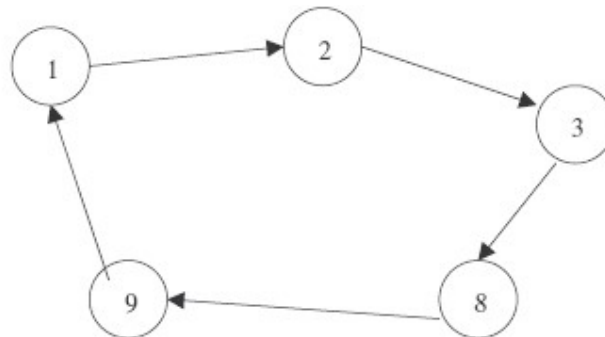


Tabla de estados:

Estados FF	1	2	3	8	9	1
FFA	0	0	0	1	1	0
FFB	0	0	0	0	0	0
FFC	0	1	1	0	0	0
FFD	1	0	1	0	1	1
Decimal	1	2	3	8	9	1

Los TNV son : 0,4,5,6,7,10,11,12,13,14,15

Las funciones de entrada a los FFs a partir de la tabla de estados:

$$D_a = \sum 3,8 + \text{TNV } (0,4,5,6,7,10,11,12,13,14,15)$$

$$D_b = \text{TNV } (0,4,5,6,7,10,11,12,13,14,15)$$

$$D_c = \sum 1,2 + \text{TNV } (0,4,5,6,7,10,11,12,13,14,15)$$

$$Dd = \sum 2,8,9 + \text{TNV } (0,4,5,6,7,10,11,12,13,14,15)$$

Minimizando estas funciones, se obtiene:

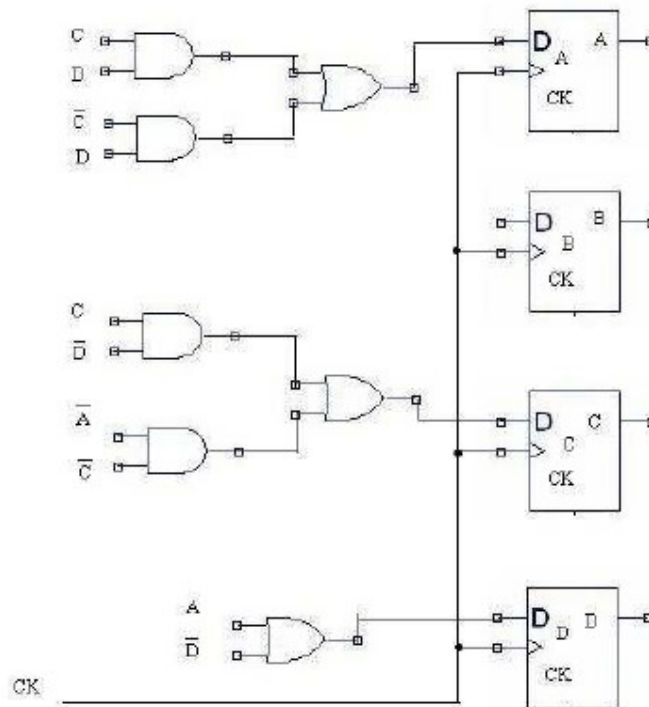
$$Da = C.D + \bar{C}.D$$

$$Db = 0$$

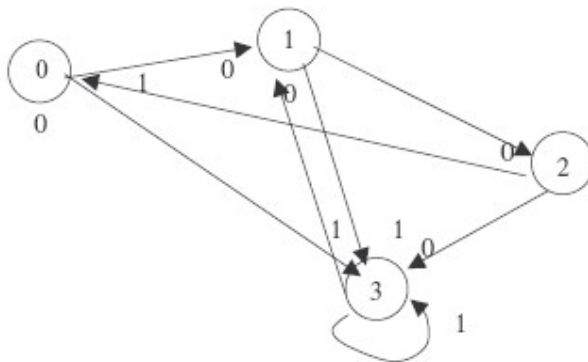
$$Dc = C.\bar{D} + \bar{A}.\bar{C}$$

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

El circuito esquemático es el siguiente:



Ejemplo con FF JK:



La tabla de estados es la siguiente:

Est. Actual	0		1		2		3	
Entrada	0	1	0	1	0	1	0	1
Próx. estado	1	3	2	3	3	0	1	3
FFA	0 ↘	↑ 0 ↘	↑ 0 ↘	↑ 0 ↘	↑ 1 ↘	↑ 1 ↓	↑ 1 ↓	↑ 1 ↘
FFB	↑ 0 ↘	↑ 0 ↘	↑ 1 ↓	↑ 1 ↘	↑ 0 ↘	0 ↘	↑ 1 ↘	↑ 1 ↘
X	0	1	0	1	0	1	0	1
decimal	0	1	2	3	4	5	6	7

Las funciones serán:

$$J_a = \sum 1,2,3 + TR(4,5,6,7)$$

$$K_a = \sum 5,6 + TR(0,1,2,3)$$

$$J_b = \sum 0,1,4 + TR(2,3,6,7)$$

$$K_b = \sum 2 + TR(0,1,4,5)$$

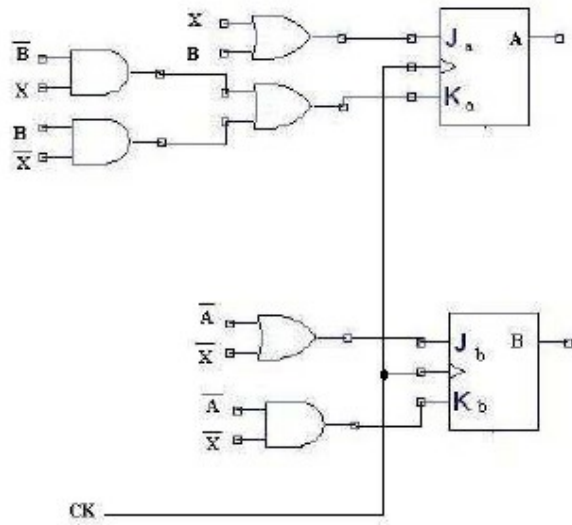
Las funciones minimizadas son:

$$J_a = X + B$$

$$K_b = \overline{B} \cdot X + B \cdot \overline{X}$$

$$J_b = \overline{A} + \overline{X}$$

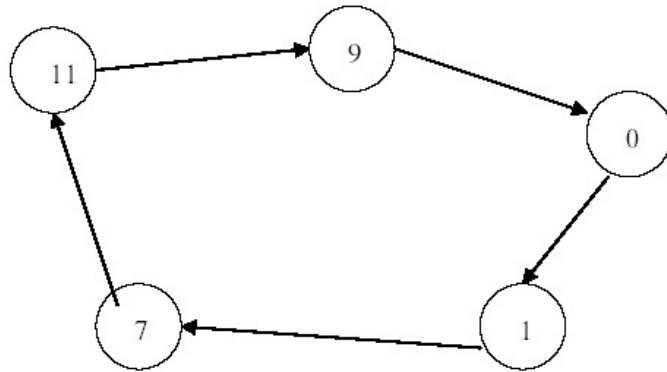
$$K_b = \overline{A} \cdot \overline{X}$$



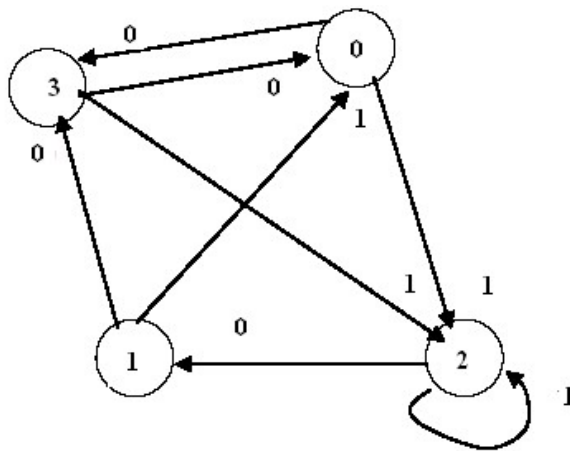
Ejercicios:

1. Representar un contador con módulo asincrónico 64 utilizando FF JK.
2. Representar un contador con módulo asincrónico 12 utilizando FF JK.
3. Representar un contador con módulo asincrónico 20 utilizando FF JK.
4. Representar un registro de desplazamiento de 3 bits con FF D
5. Representar un contador Jonhson de 5 bits.
6. Diseñar los siguientes circuitos secuenciales:

Con FF T y D



Con FF JK



7. Diseñar un contador sincrónico módulo 10.

Materia: DISEÑO LÓGICO
PRÁCTICOS DE GABINETE
TRABAJO PRÁCTICO DE GABINETE N°6
Contenido: MÁQUINA ALGORÍTMICA DE ESTADOS

Utilidad del contenido y aporte a la formación:

- Comprender procedimientos involucrados en la evolución de los componentes de las computadoras digitales.
- Describir las distintas tecnologías presentes en las unidades que constituyen la arquitectura básica de una computadora digital.

Objetivos:

- Aplicar métodos top-down para el diseño de controladores síncronos.

Diseño top – down:

Método del multiplexor

Este se basa en utilizar un multiplexor en cada una de las entradas de los FF que se requieran. A partir del diagrama MAS se realiza una tabla de estados en la cual se consideran los valores de las variables de estado en cada una de las transiciones y la condición de transición de cada estado actual al estado siguiente.

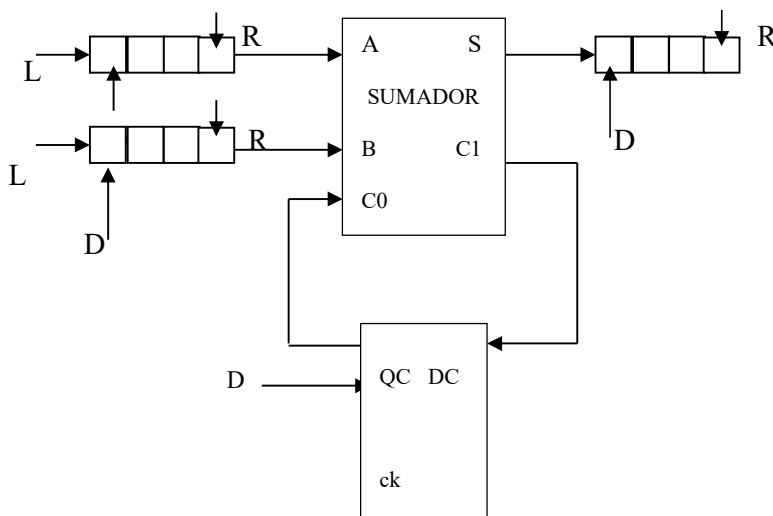
Las variables de estado son las salidas de cada uno de los FF que caracterizan a cada estado. Estas variables, además son las que actúan como entradas de selección de los multiplexores.

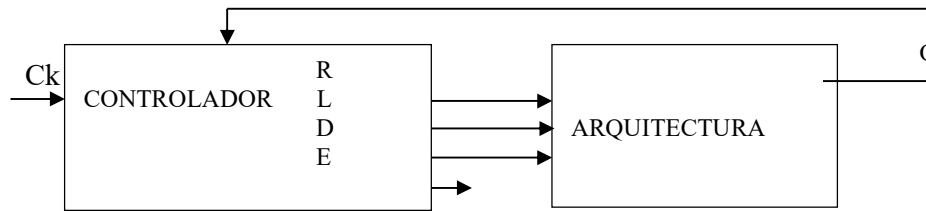
De la tabla de estados citada se obtiene las funciones de entrada a que se debe aplicar a cada una de las entradas de los multiplexores del controlador.

El número de FF necesario depende del número de estados presentes en la MAS (N° de Estados $\leq 2^n$, n = número de FF)

El siguiente es un ejemplo de la aplicación de este método:

Diseñar un controlador de la arquitectura correspondiente a un sumador serial.

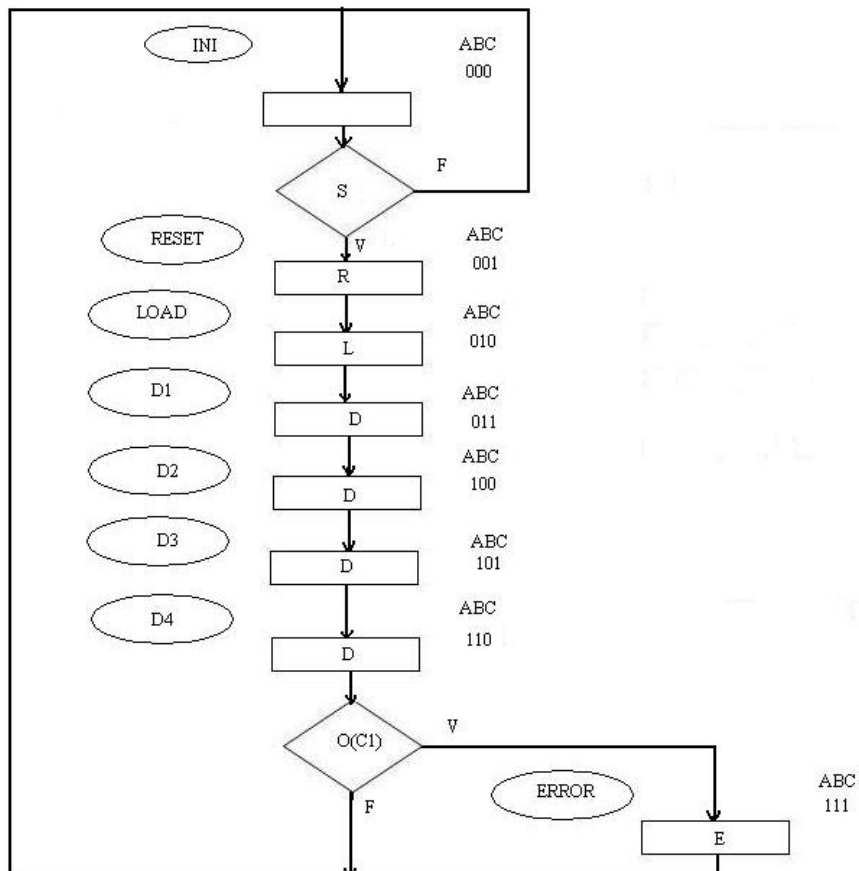




El controlador debe generar los comandos correspondientes a la inicialización de los registros, carga de los registros, desplazamientos de los bits de datos, resultado y acarreo; en la secuencia correspondiente a la suma serial. También debe generar una señal de error si se produce un sobrepasamiento en el resultado.

Utilizar el método del multiplexor.

- 1) Máquina algorítmica de estado: el siguiente diagrama representa la máquina algorítmica de estados para el sumador serial, la variable S es la de inicio y las otras variables son las indicadas en la descripción de la arquitectura.



2) método del multiplexor:

Est. Actual N°	nombre	Próx. Estado Nombre	A B C	Condición de transición
0	INI	INI RESET	0 0 0 0 0 1	\overline{S} S
1	RESET	LOAD	0 1 0	Siempre
2	LOAD	D1	0 1 1	Siempre
3	D1	D2	1 0 0	Siempre
4	D2	D3	1 0 1	Siempre
5	D3	D4	1 1 0	Siempre
6	D4	ERROR INI	1 1 1 0 0 0	\overline{O} \overline{O}
7	ERROR	INI	0 0 0	Siempre

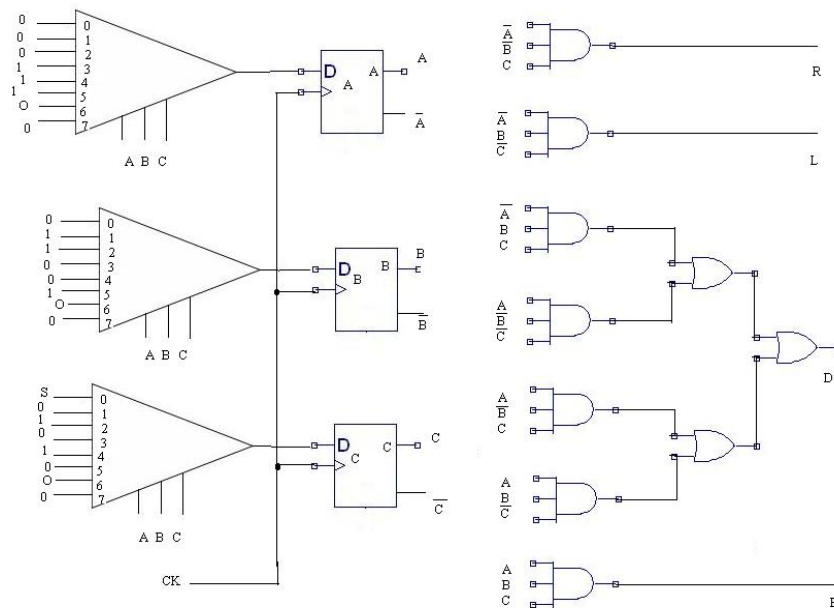
Las entradas de los tres multiplexores son:

MUXA(0)=0
MUXA(1)=0
MUXA(2)=0
MUXA(3)=1
MUXA(4)=1
MUXA(5)=1
MUXA(6)=0
MUXA(7)=0

MUXB(0)=0
MUXB(1)=1
MUXB(2)=1
MUXB(3)=0
MUXB(4)=0
MUXB(5)=1
MUXB(6)=0
MUXB(7)=0

MUXC(0)=S
MUXC(1)=0
MUXC(2)=1
MUXC(3)=0
MUXC(4)=1
MUXC(5)=0
MUXC(6)=0
MUXC(7)=0

El circuito resultante es el siguiente:



Método uno en alto

Este método se basa en que cada estado se caracteriza por el estado en alto del FF que identifica ese estado.

Para la MAS del ejemplo anterior se hace una tabla de estados como la siguiente:

Próximo estado	Estado actual	Condición de transición
INI	INI D4 INI	\overline{S} . INI \overline{O} . D4 ERROR
RESET	INI	S. INI
LOAD	RESET	RESET
D1	LOAD	LOAD
D2	D1	D1
D3	D2	D2
D4	D3	D3
ERROR	D4	O. D4

De esta tabla se obtienen las funciones de entrada a los 8 FF necesarios (este método se requieren un FF por cada uno de los estados). La funciones son:

$$D_{ini} = \overline{S}.INI + \overline{O} . D4 + ERROR$$

$$D_{reset} = S. INI$$

$$D_{load} = RESET$$

$$D_{d1} = LOAD$$

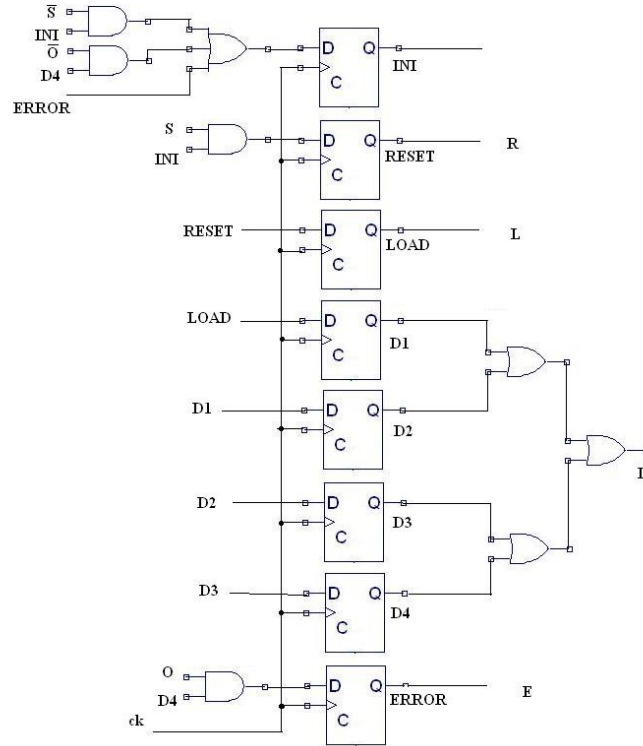
$$D_{d2} = D1$$

$$D_{d3} = D2$$

$$D_{d4} = D3$$

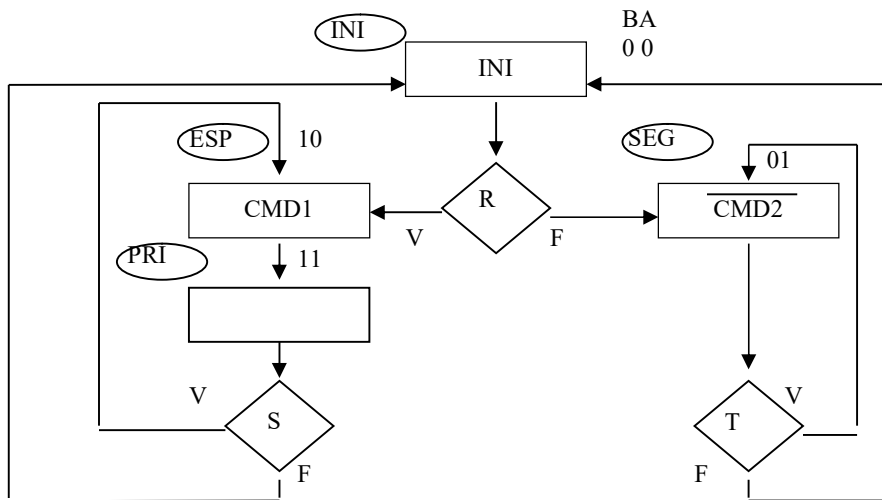
$$D_{error} = O. D4$$

El controlador se implementa con el siguiente circuito:



Ejercicios:

1. Diseñar el controlador de la MAS por el método del multiplexor:



2. Diseñar el controlador de la MAS por el método uno en alto:

