PUCE Virtual
EDUCACIÓN FLEXIBLE

Circuitos y Sistemas Digitales

Circuitos combinacionales

Clase 6

La excelencia no se improvisa

1. Introducción

Los circuitos combinacionales son la piedra angular de los sistemas digitales modernos, diseñados para generar salidas específicas a partir de una combinación determinada de entradas en un mismo instante. A diferencia de los circuitos secuenciales, carecen de memoria, por lo que su comportamiento depende exclusivamente de los valores actuales de las señales de entrada. Estos circuitos, como multiplexores, decodificadores y sumadores, son fundamentales en aplicaciones que van desde procesadores hasta dispositivos electrónicos cotidianos. Para comprender su funcionamiento, es indispensable adentrarse en el Álgebra Booleana, un marco matemático desarrollado por George Boole que define operaciones lógicas básicas (AND, OR, NOT) y permite modelar y simplificar circuitos mediante leyes y teoremas estructurados. Este lenguaje algebraico no solo facilita el diseño eficiente de sistemas digitales, sino que también establece las bases para optimizar su rendimiento y complejidad.

Antes de profundizar en el diseño de circuitos combinacionales, es crucial dominar los sistemas de numeración y la aritmética binaria, pilares de la electrónica digital. Los sistemas binarios, basados en dígitos 0 y 1, representan la información de forma compatible con la lógica de circuitos integrados. La aritmética binaria, por su parte, define operaciones como la suma, resta y multiplicación en este sistema, esenciales para implementar funciones avanzadas en hardware. Comprender estos conceptos no solo permite interpretar cómo los circuitos procesan datos, sino que también sienta las bases para explorar temas posteriores, como la conversión entre sistemas numéricos y el manejo de errores en comunicaciones digitales. Así, la combinación de teoría y práctica en estos temas abre la puerta al diseño de sistemas electrónicos robustos y eficientes.

Clase 6: Circuitos combinacionales.

Resultado o resultados de aprendizaje que será abordado con el contenido de la clase. **(No agregue los criterios)**

Comprender los conceptos fundamentales de los sistemas digitales combinacionales y su operación.

3. Circuitos combinacionales.

Los circuitos combinacionales son sistemas digitales cuyas salidas dependen únicamente de las combinaciones presentes en sus entradas en un momento dado, sin considerar estados previos. A diferencia de los circuitos secuenciales, no poseen elementos de memoria (como flip-flops), lo que simplifica su diseño pero limita su aplicación a funciones estáticas. Su estructura se compone de puertas lógicas interconectadas (AND, OR, NOT, NAND, NOR, XOR) que implementan funciones booleanas.

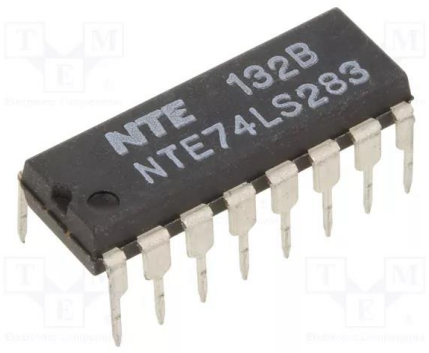
Los circuitos combinacionales resaltan con las siguientes características:

- Funcionalidad determinista: La salida se calcula mediante una tabla de verdad que relaciona todas las entradas posibles con sus respectivas salidas.
- Propagación de retardo: El tiempo que tarda una señal en atravesar las puertas lógicas afecta la estabilidad de la salida.
- Ejemplos típicos:
  - Multiplexores (MUX): Seleccionan una de varias entradas para dirigirla a una salida, usando líneas de control.
  - Decodificadores: Activan una salida específica según el código binario de entrada (ejemplo: decodificador de 3 a 8 líneas).
  - Sumadores: Realizan operaciones aritméticas binarias, como el sumador completo (Full Adder), que considera acarreos de entrada y salida.
  - Sistemas de control en dispositivos IoT.
  - Conversores de código (ejemplo: BCD a 7 segmentos).

De manera ilustrativa la Figura 1 muestra un circuito integrado con lógica combinacional que corresponde al sumador de 4 bits con acarreos NTE74LS283, mientra que la Figura 2 muestra su diagrama de pines.

Figura 1

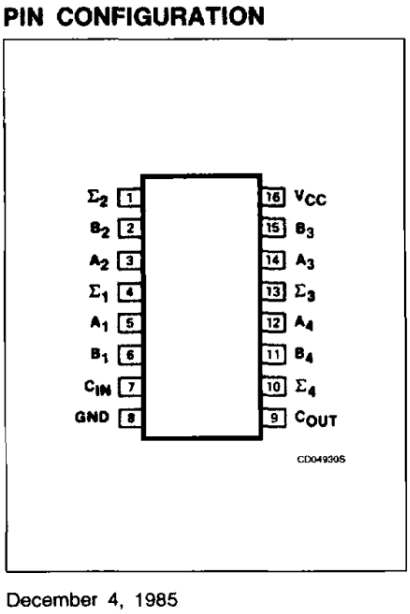
Sumador de 4 bits 74LS282.



Nota: Recuperado de NTE Electronics, 2025.

Figura 2

Configuración de pines del sumador 74LS282.



Nota: Recuperado de Signetics Data Sheet, 1985.

Para diseñar los circuitos combinacionales se deben definir las entradas, salidas y relaciones mediante tablas de verdad, considerando todas las opciones posibles tanto de las entradas como de las salidas.

Una vez que se tenga la tabla de verdad se procede con la minimización, que consiste en reducir la expresión booleana usando técnicas como mapas de Karnaugh o el algoritmo de Quine-McCluskey.

Una vez que se tenga la expresión booleana reducida se procede con la implementación que no es otra cosa que traducir la expresión simplificada a un esquema de puertas lógicas.

Para verificar que el diseño funcione según las especificaciones se puede realizar pruebas de escritorio probando con entradas aleatorias y analizando el comportamiento de las compuertas y la señal de salida, adicionalmente con herramientas modernas como Ktechlab se puede simular el circuito para garantizar la coherencia con la tabla de verdad original y luego finalmente realizar la implementación física. En este curso llegaremos hasta la etapa de simulación y usaremos para ello Ktechlab.

3.1. Algebra Booleana.

El álgebra booleana, formulada por George Boole en 1847, es la base matemática para analizar y diseñar circuitos digitales. Opera con variables binarias (0 y 1) y tres operaciones fundamentales:

Tabla 1

Operaciones lógicas básicas.

Operación	Descripción	Símbolo
AND	Producto lógico	$A \wedge B$
OR	Suma lógica	$A \vee B$
NOT	Complemento	$\neg A$

Nota: Elaborado por Escobar C. (2025)

3.1.1. Tabla de verdad del operador AND

Tabla 2

Tabla de verdad AND.

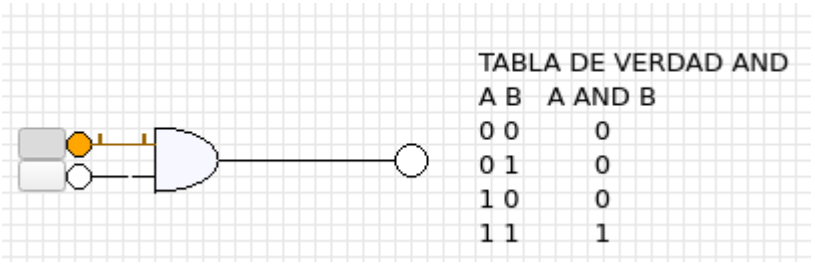
A	B	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

Nota: Elaborado por Escobar C. (2025)

Las Figura 3 y Figura 4 muestran respectivamente una compuerta AND y el esquemático de un circuito integrado comercial que empaqueta 4 compuertas de este tipo.

Figura 3

Compuerta lógica AND, Simulación KTechlab.



Nota: Elaborado por Escobar C. (2025)

Figura 4

DM74LS08 Quad 2-Input AND Gates

DM74LS08
Quad 2-Input AND Gates

General Description

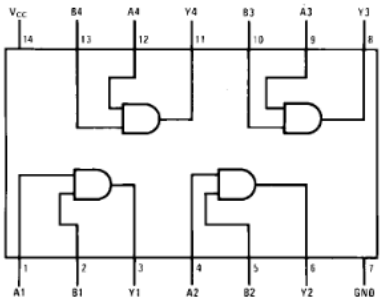
This device contains four independent gates each of which performs the logic AND function.

Ordering Code:

Order Number	Package Number	Package Description
DM74LS08M	M14A	14-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-120, 0.150 Narrow
DM74LS08SJ	M14D	14-Lead Small Outline Package (SOP), EIAJ TYPE II, 5.3mm Wide
DM74LS08N	N14A	14-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Connection Diagram



Function Table

Y = AB

Inputs		Output
A	B	Y
L	L	L
L	H	L
H	L	L
H	H	H

H = HIGH Logic Level
L = LOW Logic Level

Nota: Recuperado de Fairchild Semiconductor Datasheet, 2000b.

3.1.2. Tabla de verdad del operador OR

Tabla 3

Tabla de verdad OR (Escobar C, 2025)

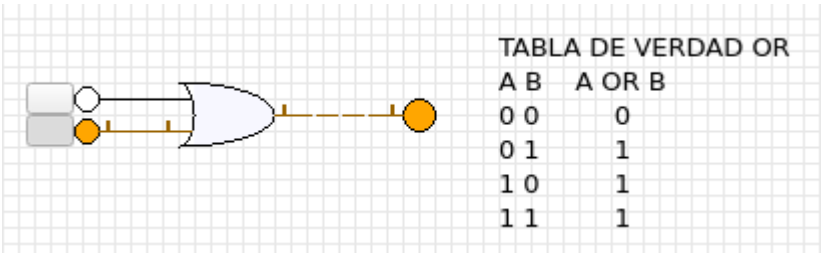
A	B	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

Nota: Elaborado por Escobar C. (2025)

Las Figura 5 y Figura 6 muestran respectivamente una compuerta OR y el esquemático de un circuito integrado comercial que empaqueta 4 compuertas de este tipo.

Figura 5

Compuerta lógica OR, Simulación KTechlab.



Nota: Elaborado por Escobar C. (2025)

Figura 6

DM74LS32 Quad 2-Input OR Gate.

FAIRCHILD SEMICONDUCTOR™

June 1986
Revised March 2000

DM74LS32
Quad 2-Input OR Gate

General Description

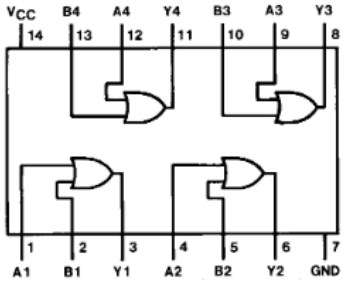
This device contains four independent gates each of which performs the logic OR function.

Ordering Code:

Order Number	Package Number	Package Description
DM74LS32M	M14A	14-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-120, 0.150 Narrow
DM74LS32SJ	M14D	14-Lead Small Outline Package (SOP), EIAJ TYPE II, 5.3mm Wide
DM74LS32N	N14A	14-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Connection Diagram



Function Table

Y = A + B

Inputs		Output
A	B	Y
L	L	L
L	H	H
H	L	H
H	H	H

H = HIGH Logic Level
L = LOW Logic Level

Nota: Recuperado de Fairchild Semiconductor Datasheet, 2000c

3.1.3. Tabla de verdad del operador NOT

Tabla 4

Tabla de verdad NOT.

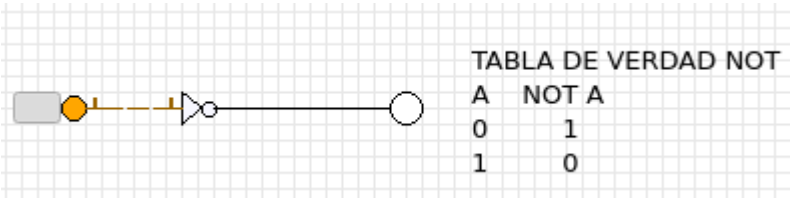
A	$\neg A$
0	1
1	0

Nota: Elaborado por Escobar C. (2025)

Las Figura 7 y Figura 8 muestran una compuerta NOT y el esquemático de un circuito integrado comercial que empaqueta 6 compuertas de este tipo.

Figura 7

Compuerta lógica NOT, Simulación KTechlab.



Nota: Elaborado por Escobar C. (2025)

Figura 8

DM74LS04 Hex Inverting Gates.

FAIRCHILD SEMICONDUCTOR™

August 1986
Revised March 2000

DM74LS04
Hex Inverting Gates

General Description

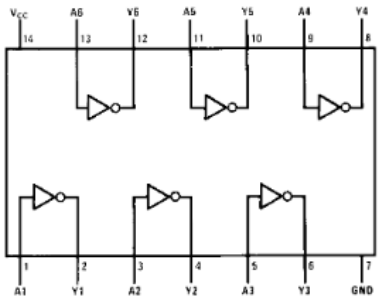
This device contains six independent gates each of which performs the logic INVERT function.

Ordering Code:

Order Number	Package Number	Package Description
DM74LS04M	M14A	14-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-120, 0.150 Narrow
DM74LS04SJ	M14D	14-Lead Small Outline Package (SOP), EIAJ TYPE II, 5.3mm Wide
DM74LS04N	N14A	14-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Connection Diagram



Function Table

$Y = \overline{A}$

Input	Output
A	Y
L	H
H	L

H = HIGH Logic Level
L = LOW Logic Level

Nota: Recuperado de Fairchild Semiconductor Datasheet, 2000a.

3.1.4. Leyes y teoremas esenciales

El aporte de George Boole es invaluable para las ciencias de la computación y el mundo como lo conocemos actualmente. Nada de lo que existe tecnológicamente alrededor de los circuitos digitales y la computación existirían. De manera complementaria mire el documental dedicado a George Boole como fundador del a Ciencia de la Computación.

Las leyes y teoremas esenciales del álgebra booleana son reglas matemáticas que gobiernan las operaciones lógicas entre variables binarias (0 y 1). Estas normas permiten simplificar expresiones booleanas, optimizar circuitos digitales y garantizar coherencia en el diseño lógico. Son la base teórica para construir sistemas digitales eficientes y libres de ambigüedades. A continuación, se detallan las más relevantes:

A. Leyes Fundamentales

a) Leyes de Identidad

Operación	Descripción	Explicación
$A \vee 0 = A$	Identidad para la suma lógica	Al sumar lógicamente (OR) una variable A con 0, el resultado siempre es A
$A \wedge 1 = A$	Identidad para el producto lógico	Al multiplicar lógicamente (AND) una variable A, el resultado siempre es A

b) Leyes de Complemento

Operación	Descripción	Explicación
$A \vee \neg A = 1$	Identidad para la suma lógica	Al sumar lógicamente (OR) una variable A con 0, el resultado siempre es A
$A \wedge 1 = A$	Identidad para el producto lógico	Al multiplicar lógicamente (AND) una variable A, el resultado siempre es A

B. Leyes Estructurales

a) Leyes Conmutativas

Operación	Descripción	Explicación
$A \vee B = B \vee A$	Conmutatividad para OR	Al sumar lógicamente (OR), el orden de las variables no altera el resultado.
$A \wedge B = B \wedge A$	Conmutatividad para AND	Al multiplicar lógicamente (AND), el orden de las variables no altera el resultado.

b) Leyes Asociativas

Operación	Descripción	Explicación
$A \vee (B \vee C) = (A \vee B) \vee C$	Asociativa para OR	Al sumar lógicamente (OR), la agrupación de variables no afecta el resultado..
$A \wedge (B \wedge C) = (A \wedge B) \wedge C$	Asociativa para AND	Al multiplicar lógicamente (AND), la agrupación de variables no afecta el resultado.

c) Leyes Distributivas

Operación	Descripción	Explicación
$A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$	Distributividad de AND sobre OR	Permite expandir o factorizar expresiones con el álgebra booleana.
$A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$	Distributividad de OR sobre AND	Permite expandir o factorizar expresiones con el álgebra booleana.

C. Teorema de De Morgan

Teorema de De Morgan

Operación	Descripción	Explicación
$\neg(B \vee C) = \neg A \wedge \neg B$	El complemento de una suma es el producto de los complementos	Permite convertir compuertas OR en AND.
$\neg(B \wedge C) = \neg A \vee \neg B$	El complemento de un producto es la suma de los complementos	Permite convertir compuertas AND en OR.

Con el teorema de De Morgan se pueden convertir puertas OR en AND (y viceversa) usando inversores, lo que es clave para implementar circuitos con NAND/NOR universales.

D. Leyes de Absorción

Leyes de Absorción

Operación	Explicación
$A \vee (A \wedge B) = A$	Una variable "absorbe" términos redundantes en una expresión.
$A \wedge (A \vee B) = A$	Una variable "absorbe" términos redundantes en una expresión.

E. Principio de Dualidad

Toda expresión booleana tiene un "dual" obtenido intercambiando OR ↔AND y 0 ↔1., por ejemplo el dual de $A \vee 0 = A$ es $A \wedge 1 = A$.

Si bien es cierto las leyes y teoremas del álgebra booleana se pueden demostrar analíticamente, también se pueden simular los dos lados de las ecuaciones y comprobar que las salidas son las mismas. Para profundizar sus conocimientos en el simulador KTechlab, revise su documentación en este enlace.

3.2. Sistemas de numeración y arimética binaria.

Los sistemas de numeración y la aritmética binaria constituyen la base de la electrónica digital y la computación, ya que permiten representar y manipular información utilizando solo dos dígitos: 0 y 1. Este sistema, alineado con el comportamiento de los circuitos electrónicos (donde 0 puede representar 0V y 1 puede representar 5V), es la base de operaciones en microprocesadores, memorias y dispositivos digitales.

3.2.1. Sistemas de Numeración

Un sistema de numeración define cómo se representan los números mediante símbolos y reglas. Los más relevantes en computación son:

a) Sistema Binario (Base 2)
b) Sistema Decimal (Base 10)
c) Sistema Hexadecimal (Base 16)
d) Sistema Octal (Base 8)

La base del un sistema de numeración representa el número de digitos disponibles en ese sistema.

a) Sistema Binario (Base 2)

Dígitos: 0 y 1.
Valor posicional: Cada posición representa una potencia de 2.
Ejemplo: $1011_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 11_{10}$
El sistema binario es usado de facto para la representación de datos en los computadores y las operaciones lógicas en CPU, así como también en el hardware implementado con fines de operar circuitos con lógica binaria (ejemplo: estados de un transistor).

b) Sistema Decimal (Base 10)

Dígitos: 0 1 2 3 4 5 6 7 8 9.
Valor posicional: Cada posición representa una potencia de 10.
Ejemplo: $345_{10} = 3 \cdot 10^2 + 4 \cdot 10^1 + 5 \cdot 10^0$
Este es el sistema natural para humanos, y se usa como interfaz para interactuar con el software, lo que hace necesario contar con mecanismos de conversión entre sistemas d enumeración.

c) Sistema Hexadecimal (Base 16)

Dígitos: 0 1 2 3 4 5 6 7 8 9 A B C D E F.
Valor posicional: Cada posición representa una potencia de 16.
Ejemplo: $2F16 = 2 \cdot 16^1 + 15 \cdot 16^0 = 47_{10}$

El sistema hexadecimal se usa por su facilidad en la representación de números binarios para la interpretación humana, ya que existe una correspondencia directa en una secuencia de bits agrupada cada 4 y una cadena hexadecimal (ejemplo: $1101\_1110_2 = DE_{16}$). También es común el uso sistema hexadecimal para el manejo de direcciones de memoria y representación de colores (ejemplo: #FF0000 = rojo).

d) Sistema Octal (Base 8)
Dígitos: 0 1 2 3 4 5 6 7.
Valor posicional: Cada posición representa una potencia de 8.
Ejemplo:
$75_8 = 7 \cdot 8^1 + 5 \cdot 8^0 = 61_{10}$
El sistema octal se usa por su facilidad en la representación de números binarios para la interpretación humana, ya que existe una correspondencia directa en una secuencia de bits agrupada cada 3 y una cadena hexadecimal (ejemplo: $110\_001 = 61_8$).

3.2.2. Conversiones entre Sistemas

a) Decimal a binario
Método: Divisiones sucesivas 2.
Ejemplo: Convertir 25_{10} a binario

Para convertir el número 25 en base 10 a binario se realizan divisiones sucesivas para 2 y luego se recupera en orden inverso los residuos, cuya serie constituye el número binario convertido.

$$\frac{25}{2} = 12(residuo = 1)$$
$$\frac{12}{2} = 6(residuo = 0)$$
$$\frac{6}{2} = 3(residuo = 0)$$
$$\frac{3}{2} = 1(residuo = 1)$$
$$\frac{1}{2} = 0(residuo = 1)$$

Para convertir el número 25 en base 10 a binario se realizan divisiones sucesivas para 2 y luego se recupera en orden inverso los residuos, cuya serie constituye el número binario convertido.
En este caso 11001_2 , el cual es el binario equivalente a 25.

b) Binario a Decimal

Método: Suma de potencias de 2.

Ejemplo: Convertir 11001_2 a decimal.

Para convertir el número 11001_2 a decimal se realiza una suma de potencias de cada uno de los dígitos multiplicado por la base elevado a la ubicación del dígito, dónde el bit menos significativo está elevado a la cero y el bit mas significativo está elevado a la n-1, es decir:

$11001_2 \rightarrow X_{10}$

$1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 16 + 8 + 0 + 0 + 1 = 25_{10}$

c) Hexadecimal a Binario

Método: Al partir de bases relacionada a través de potenciación, es decir:

[Base origen] = [Base destino]^n,

en este caso;

$16 = 2^4$

se usa la regla en la que cada dígito de la base origen hexadecimal equivale a la combinación secuencial de 4 bits de la base destino binario.

Ejemplo:

$BFA76_{16} = 1011\_1111\_1010\_0111\_0110_2 = 10111111101001110110_2$

d) Binario a Hexadecimal

Método: Para convertir un número binario a hexadecimal, dado que son bases relacionadas a través de potenciación, dónde:

[Base destino] = [Base origen] ^{n},

es decir;

$16 = 2^4$

se usa la regla en la que cada dígito de la base destino hexadecimal equivale a la combinación secuencial de 4 bits de la base origen binario, lo que quiere decir que la base origen debe tener un número dígitos múltiplo de 4, para lo cual en caso de ser necesario se rellena con ceros a la izquierda.

Ejemplo:

$1011011101111_2 \rightarrow X_{16}$

Al separarlo en grupos de 4 bits partiendo desde la derecha:

$1\ 0110\ 1110\ 1111_2$

se encuentra que hace falta rellenar de tres ceros al primer grupo de la izquierda que solo tiene un dígito, es decir:

$0001\ 0110\ 1110\ 1111_2$

Dónde ya se puede aplicar la regla de equivalencia directa cada cuatro bits:

$0001\ 0110\ 1110\ 1111_2 = 1\_6\_E\_F_{16} = 16EF_{16}$

e) Octal a Binario

Método: Al partir de bases relacionada a través de potenciación, es decir:

[Base origen] = [Base destino]^n,

en este caso;

$8 = 2^3$

se usa la regla en la que cada dígito de la base origen hexadecimal equivale a la combinación secuencial de 3 bits de la base destino binario.
Ejemplo:

$10376_8 = 001\_000\_011\_111\_110_2 = 0010000111111110_2$

f) Binario a Octal

Método: Para convertir un número binario a octal, dado que son bases relacionadas a través de potenciación, dónde:

[Base destino] = [Base origen] ^{n},

es decir;

$8 = 2^3$

se usa la regla en la que cada dígito de la base destino octal equivale a la combinación secuencial de 3 bits de la base origen binario, lo que quiere decir que la base origen debe tener un número dígitos múltiplo de 3, para lo cual en caso de ser necesario se rellena con ceros a la izquierda.
Ejemplo:

$1011011101111_2 \rightarrow X_8$

Al separarlo en grupos de 3 bits partiendo desde la derecha:

$1\ 011\ 011\ 101\ 111_2$

se encuentra que hace falta rellenar de dos ceros al primer grupo de la izquierda que solo tiene un dígito, es decir:

$001\ 011\ 011\ 101\ 111_2$

Dónde ya se puede aplicar la regla de equivalencia directa cada tres bits:

$001\ 011\ 011\ 101\ 111_2 = 1\_3\_3\_5\_7_8 = 13357_8$

3.2.3. Aritmética Binaria

a) Suma binaria.

Reglas básicas de la suma:

- 0 + 0 = 0
- 0 + 1 = 1
- 1 + 0 = 1
- 1 + 1 = 0 con acarreo 1, es decir 10
- 1 + 1 +1 = 1 con acarreo 1, es decir 11

En el siguiente ejemplo se aprecia la suma de dos números binarios:

1	1	1	1		(Acarreos)	Decimal
	1	0	1	0	Operador 1	10
+	0	1	1	1	Operador 2	7
1	0	0	0	1	Resultado	17

b) Resta Binaria con Complemento a 2

Para realizar la resta binaria con complemento a 2, el sustraendo es convertido a su complemento a 2 y sumado al minuendo.

En el siguiente ejemplo se realiza una resta binaria usanod complemento a uno:

	1	0	1	0	Minuendo	10
-	0	1	1	1	Sustraendo	7
	?	?	?	?	Resultado	?

Complemento a uno del sustraendo sumado 1:

	0	1	1	1	Sustraendo
	1	0	0	0	Complemento a 1
+				1	Sumado 1
	1	0	0	1	Complemento a 2

Suma del minuendo con el complemento a 2 del sustraendo:

					(Acarreos)	Decimal
	1	0	1	0	Operador 1	10
+	1	0	0	1	Operador 2	9
1	0	0	1	1	Resultado	3

En el acarreo se ignora el bit final de acarreo, por tanto el resultado es 0011

c) Multiplicación binaria.

La multiplicación binaria se realiza de manera similar a la decimal, usando sumas y desplazamientos.
Ejemplo:

Tabla 5: Multiplicación binaria

			1	0	1	1	Operador 1
		x		1	1	0	Operador 2
			0	0	0	0	1011x0
		1	0	1	1		1011x1, desplazado 1 posición
	1	0	1	1			1011x1, desplazado 2 posiciones
1	0	0	0	0	1	0	Resultado

En la multiplicación binaria se cumple que cualquier número binario multiplicado por 2, simplemente se agrega un cero a la derecha:

10 x 10 = 100
100 x 10 = 1000

d) División binaria

A nivel computacional la división binaria usa un algoritmo iterativo basado en restas y comparaciones y no forma parte de este curso.

Referencias citadas en la Clase 6.

Fairchild Semiconductor Datasheet. (2000a). DM74LS04Hex Inverting Gates. DM74LS04Hex Inverting Gates. https://www.alldatasheet.com/datasheet-pdf/view/51022/FAIRCHILD/74LS04.html

Fairchild Semiconductor Datasheet. (2000b). Quad 2-Input AND Gates. Quad 2-Input AND Gates. https://www.alldatasheet.com/datasheet-pdf/view/51024/FAIRCHILD/74LS08.html

Fairchild Semiconductor Datasheet. (2000c). Quad 2-Input OR Gate. Quad 2-Input OR Gate. https://www.alldatasheet.com/datasheet-pdf/view/51075/FAIRCHILD/74LS32.html

NTE Electronics. (2025). Sumador de 4 bits 74LS282. Sumador de 4 bits 74LS282. https://www.tme.com/pe/es/details/nte74ls283/contadores-divisores/nte-electronics/

Definición de los términos citados en la Clase 6.

Circuitos combinacionales: Los circuitos combinacionales son sistemas digitales cuyas salidas dependen únicamente de las combinaciones presentes en sus entradas en un momento dado

Sistemas hexadecimal y octal: Los sistemas hexadecimal y octal se usan por su facilidad en la representación de números binarios para la interpretación humana, ya que existe una correspondencia directa en una secuencia de bits agrupada cada 3 y 4 bits respectivamente.

PUCE Virtual
EDUCACIÓN FLEXIBLE

La excelencia no se improvisa

síguenos

MDASC - 04 - 251