

**UNIVERSIDAD TÉCNICA ESTATAL DE QUEVEDO**



**CIENCIAS DE LA INGENIERÍA  
INGENIERÍA EN SOFTWARE  
ECUADOR**

# **Sistemas Numéricos y Representación de Datos Interna de la Computadora**

**AUTORES:**

**ALMAGRO INTRIAGO LAINER PATRICIO  
FUERTES ARRAES EDSON DANIEL  
NIEVES SANCHEZ JIMMY SAMUEL  
SALTOS TELLO JOSEPH ORLANDO**

**DOCENTES:**

**ING. GUERRERO ULLOA GLEISTON CICERON**

**CURSO/PARALELO:**

**SEGUNDO SOFTWARE “B”**

**SPA 2025 – 2026**

## ÍNDICE

1	Introducción .....	3
2	Objetivo .....	3
3	Operaciones Fundamentales en Binario: .....	3
3.1	Descripción de la Base 2 y su Importancia, procesamiento de datos en computadoras .....	3
3.2	Suma Binaria .....	3
3.3	Resta Binaria .....	4
3.4	Multiplicación Binaria .....	5
3.5	División Binaria .....	5
4	Operaciones en Octal y Hexadecimal .....	6
4.1	Uso práctico en programación .....	6
4.2	Operaciones aritméticas con octal y hexadecimal .....	6
4.3	Operaciones Aritméticas (Suma, Resta, Multiplicación, División) .....	6
5	Flujo de Datos dentro de una Computadora .....	8
5.1	Arquitectura clásica de un computador modelo Von Neumann .....	8
5.2	Dispositivo de operación .....	9
5.3	Unidad de control .....	9
5.4	Memoria del dispositivo .....	9
5.5	Unidad central de procesamiento .....	9
5.6	Unidad Aritmética Lógica (ALU) .....	9
5.7	Registros internos .....	9
5.8	Sistema de interconexión (Buses) .....	10
5.9	Periféricos .....	11
5.10	Entrada/Salida .....	11
5.11	Acceso directo a la memoria (DMA) .....	11
5.12	Técnicas de Interrupción .....	11
5.13	Tratamiento de las E/S como posiciones de memoria .....	11
6	Códigos de Representación Numérica y No Numérica: .....	11
6.1	Código Binario Puro .....	12
6.2	Código BCD (Binary Coded Decimal) .....	12
6.3	IEEE 754 .....	12
6.4	Códigos De Representación No Numérica .....	12
6.5	ASCII (American Standard Code for Information Interchange) .....	12
6.6	EBCDIC .....	12
7	Conclusión .....	13
8	Bibliografía .....	13
9	Anexos .....	15 9.1
	Git-Hub .....	15

## **1 Introducción**

Los sistemas de numeración binarios, octales y hexadecimales son esenciales para el funcionamiento de las computadoras. Las que las operaciones aritméticas con números binarios forman la base del procesamiento digital. Los sistemas octales y hexadecimales facilitan la lectura y escritura de datos binarios y son muy usados en arquitectura de computadoras.

El flujo de datos de los componentes de una computadora hace referencia al movimiento de información entre registros, buses, memoria y la unidad de procesamiento, haciendo posible la ejecución de instrucciones. Los códigos de representación como ASCII, BCD permiten almacenar y proteger datos numéricos y no numéricos, asegurando su correcta manipulación.

## **2 Objetivo**

Desarrollar habilidades para realizar operaciones en sistemas numéricos binario, octal y hexadecimal, así como entender el flujo y representación de datos dentro de una computadora.

## **3 Operaciones Fundamentales en Binario:**

### **3.1 Descripción de la Base 2 y su Importancia, procesamiento de datos en computadoras**

El sistema binario utiliza solo dos símbolos: el 0 y el 1, las computadoras interpretan estos símbolos como estados eléctricos: apagado y encendido.

Claude Shannon propuso usar interruptores para representar operaciones lógicas, esta propuesta permitió aplicar principios de álgebra booleana a circuitos eléctricos.

Según Gregorio et al. (2020), los procesadores modernos emplean el sistema binario para almacenar y procesar datos. Esta codificación facilita la construcción de sistemas digitales eficientes y precisos [1].

### **3.2 Suma Binaria**

La suma binaria se realiza de derecha a izquierda como en el sistema decimal, el sistema combina bits y aplica reglas simples para generar resultados. Cuando se suman  $1 + 1$ , el

sistema produce un 0 y genera un *carry* hacia la izquierda.

Este acarreo se añade a la siguiente columna en la operación.

La presencia del *acarreo* asegura que la suma binaria funcione correctamente entre columnas cada acarreo se propaga según sea necesario en el cálculo.

EJEMPLO:

$$\begin{array}{r}
 10011000 \\
 + 00010101 \\
 \hline
 10101101
 \end{array}$$

acarreo  $\longrightarrow$ 

$$\begin{array}{r}
 1 \quad 1 \quad 1 \\
 1 \quad 0 \quad 1 \quad 1 \quad 0 \\
 + 0 \quad 1 \quad 1 \quad 1 \quad 0 \\
 \hline
 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0
 \end{array}$$

Figura 1 Suma binaria

### 3.3 Resta Binaria

La resta binaria también se realiza de derecha a izquierda, usando reglas básicas. El sistema reconoce cuándo se requiere un *borrow* para continuar la operación.

Cuando se intenta restar 1 de 0, el sistema pide prestado un 1 de la posición siguiente, esto convierte el 0 en 10 (binario) y permite completar la resta.

El *borrow* garantiza que las restas se realicen correctamente en operaciones más largas.

El sistema propaga el préstamo si la posición siguiente también es 0.

Ejemplo:

$$\begin{array}{r}
 10001 \\
 - 01010 \\
 \hline
 00111
 \end{array}
 \qquad
 \begin{array}{r}
 11011001 \\
 - 10101011 \\
 \hline
 00101110
 \end{array}$$

Figura 2 Resta binaria

### 3.4 Multiplicación Binaria

La multiplicación binaria solo tiene cuatro combinaciones posibles entre ceros y unos, estas reglas hacen que el proceso sea rápido y predecible.

El sistema trata al 1 como el elemento neutro de la multiplicación, cuando se multiplica  $1 \times 1$ , el resultado siempre es 1. La multiplicación completa requiere desplazar resultados parciales y sumarlos estos pasos se parecen al proceso manual de multiplicación decimal.

Ejemplo:

$$\begin{array}{r} 10100 \\ \times 1101 \\ \hline 10100 \\ 00000 \\ 10100 \\ 10100 \\ \hline 100000100 \end{array}$$

*Figura 3 Multiplicación Binaria*

### 3.5 División Binaria

La división binaria funciona como la decimal, pero con números binarios, el sistema usa restas binarias para obtener el cociente y el residuo. El divisor se compara con partes del dividendo, y se resta si es menor o igual coloca un 1 en el cociente cuando realiza una resta exitosa. Si no se puede restar, el sistema coloca un 0 en el cociente y baja el siguiente bit, el proceso se repite hasta terminar con todos los bits del dividendo.

Ejemplo:

$$\begin{array}{r} \text{Dividendo} \\ \uparrow \\ 101011 \quad | 100 \rightarrow \text{Divisor} \\ -100 \quad 1010, 11 \rightarrow \text{Cociente} \\ \hline 101 \\ -100 \\ \hline 110 \\ -100 \\ \hline 100 \\ -100 \\ \hline 0 \rightarrow \text{Resto o residuo} \end{array}$$

*Figura 4 División binaria*

## 4 Operaciones en Octal y Hexadecimal

### 4.1 Uso práctico en programación

El sistema octal y el sistema hexadecimal tienen un papel fundamental en la hora de la programación. Se desarrolla más en las áreas del sistema embebidos redes, gráficos por computadora y bajo nivel. En los programadores se manejan mejores con símbolos y numero costo. Esos 2 sistema aporta una anotación más adaptable para la construcción de valores en el sistema binario 0 y 1 siendo utilizado por el computador [2].

### 4.2 Operaciones aritméticas con octal y hexadecimal

El manejo del sistema de numeración tiene una composición de 8 dígitos desde el inicio del 0 hasta el 7: [ 0,1,2,3,4,5,6,7]. Ese sistema fue de gran utilidad en los inicios de la computadora ya que equivalía a 3 binarios. El sistema dio apertura a los lenguajes de programación antiguos [3].

El sistema hexadecimal, está relacionado a 16 dígitos que es de gran utiliza en el área digital. Mientras que el sistema decimal se cuenta del 1 al 10, El hexadecimal se cuenta del 0 al 9. Después del número 9 vienen A, B, C, D, E, F [4].

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

Figura 5 Tabla Hexadecimal

### 4.3 Operaciones Aritméticas (Suma, Resta, Multiplicación, División)

Las operaciones más comunes son suma, resta, multiplicación y división en sistemas octales y hexadecimales. Su procedimiento es muy similar al sistema decimal que comúnmente realizamos(base 10), pero las reglas de "acarreo" y "préstamo" se basan en la base del sistema (8 para octal, 16 para hexadecimal) en lugar de 10 [5].

**Suma:** Se suman los dígitos columna por columna de derecha a izquierda. Si la suma de una columna es igual o mayor que la base (8 u 16), se escribe el residuo de la división por la base y se acarrea el cociente a la siguiente columna de la izquierda [6].

$$\begin{array}{r}
 2 \ 1 \ 1 \leftarrow \text{Acarreo} \\
 \text{F} \ 3 \ \text{B} \ \text{C} \\
 9 \ \text{D} \ \text{D} \ 0 \\
 + 3 \ \text{A} \ 0 \ 6 \ 0 \\
 \hline
 5 \ 3 \ 1 \ \text{E} \ \text{C}
 \end{array}$$

Figura 6 Suma Hexadecimal

$$\begin{array}{r}
 1 \ 1 \\
 7 \ 7 \\
 + \quad 1 \\
 \hline
 1 \ 0 \ 0
 \end{array}$$

Figura 7 Suma Octal

**Resta:** Se restan los dígitos columna por columna de derecha a izquierda. Si un dígito es menor que el que se está restando, se pide "prestado" a la columna de la izquierda. Un préstamo de una columna de valor  $B$  añade  $B$  (8 u 16) al dígito actual.

$$\begin{array}{r}
 \text{A} \ \text{F} \ 3 \ \text{B} \ \text{C} \\
 - 3 \ \text{A} \ 0 \ 6 \ 0 \\
 \hline
 7 \ 5 \ 3 \ 5 \ \text{C}
 \end{array}$$

Figura 8 Resta Hexadecimal

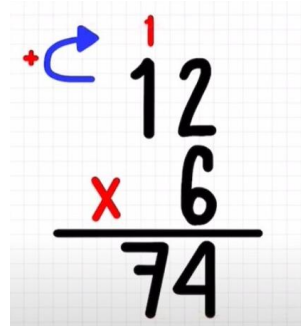
$$\begin{array}{r}
 -1 \ +8 \\
 6 \ 5 \\
 - 3 \ 6 \\
 \hline
 2 \ 7
 \end{array}$$

Figura 9 Resta Octal

**Multiplicación:** Se multiplica de forma similar a la multiplicación decimal, generando productos parciales. La suma de estos productos parciales debe seguir las reglas de suma de la base correspondiente, manejando los acarreos [7].

$$\begin{array}{r}
 3B57 \\
 \times \quad A3 \\
 \hline
 + \quad B205 \\
 25166 \\
 \hline
 25C865
 \end{array}$$

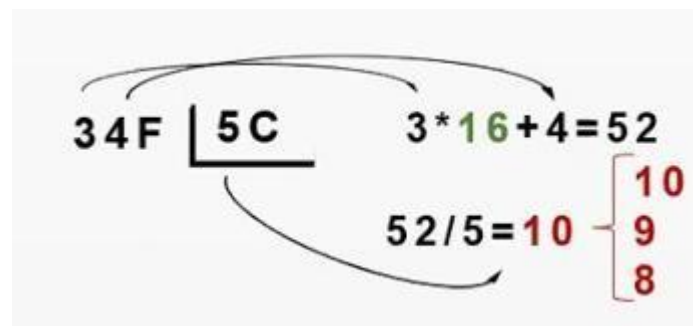
Figura 10 Multiplicación Hexadecimal



$$\begin{array}{r}
 12 \\
 \times 6 \\
 \hline
 74
 \end{array}$$

Figura 11 Multiplicación Octal

**División:** Se realiza mediante un proceso similar a la división larga decimal, pero las restas y multiplicaciones internas se efectúan siguiendo las reglas del sistema octal o hexadecimal.



$$\begin{array}{r}
 34F \overline{) 5C} \\
 \hline
 3 \times 16 + 4 = 52 \\
 52 / 5 = 10
 \end{array}$$

Figura 12 División Hexadecimal



$$\begin{array}{r}
 30 \\
 2 \overline{) 60} \\
 \underline{-6} \phantom{0} \\
 00 \\
 \underline{-0} \\
 00
 \end{array}$$

Figura 13 División Octal

## 5 Flujo de Datos dentro de una Computadora

Según Rodríguez, en el libro Evaluación Y Arquitectura Del Computador una instrucción no es más que una combinación de unos y ceros que, según como se combinen, van a indicar que operaciones tiene que hacer el computador [8].

### 5.1 Arquitectura clásica de un computador modelo Von Neumann

La principal característica de este modelo es que almacena los datos e instrucciones en una misma memoria.

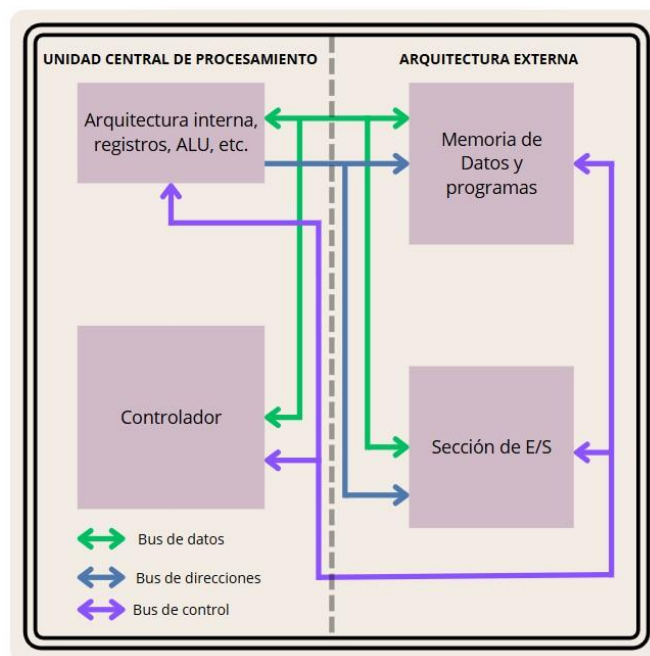


Figura 14 Arquitectura Von Neumann

El computador modelo Von Neumann consta varios componentes:

## **5.2 Dispositivo de operación**

Ejecuta un sistema de instrucciones sobre segmentos de información almacenada y separa la memoria del CPU.

## **5.3 Unidad de control**

Implementa constantemente algoritmos de decodificación de instrucciones que vienen de la memoria del dispositivo y realiza funciones de dirección general de todos los nodos del computador. Por lo tanto, el dispositivo de operación y la unidad de control conforman una estructura llamada La unidad central de procesamiento (CPU).

## **5.4 Memoria del dispositivo**

Son un conjunto de celdas con identificadores únicos, que contienen instrucciones y datos.

## **5.5 Unidad central de procesamiento**

La unidad central de procesamiento , es el hardware dentro del computador el cual traduce las instrucciones de un programa mediante operaciones aritméticas básicas. Su operación fundamental sigue siendo la misma.

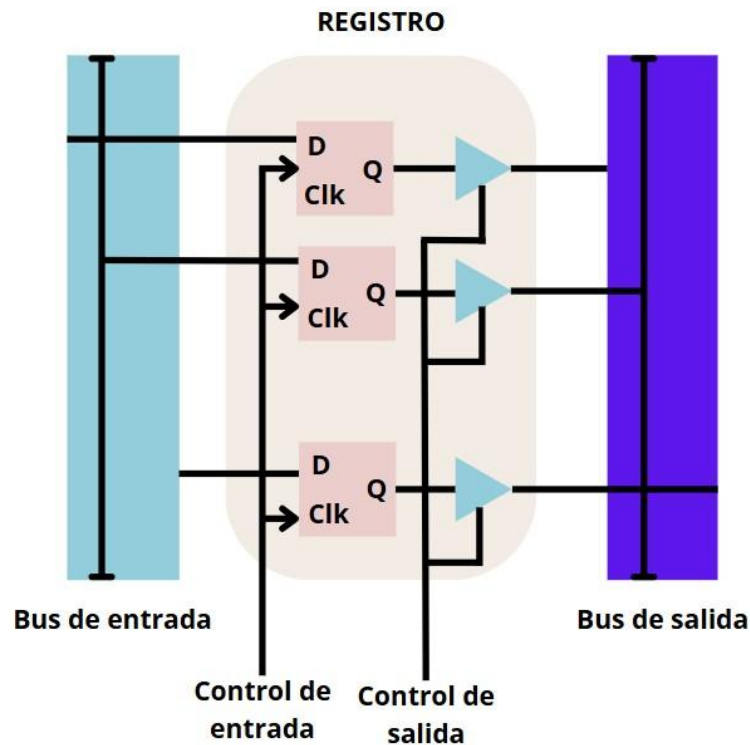
La unidad de control (UC) se encarga de leer las instrucciones de la memoria y que operaciones tiene que ejecutar. La UC genera señales eléctricas que marcaron el flujo de datos en todo el computador e internamente en la CPU.

## **5.6 Unidad Aritmética Lógica (ALU)**

Esta unidad es la encargada de realizar las transformaciones de los datos y está ubicada en el CPU.

## **5.7 Registros internos**

Según Osio et al. (2020), en el libro de Sistemas digitales basados en microcontroladores, sustentan que los registros internos son bloques biestables que almacenan información básica con la cual, el CPU, va a trabajar, ejercitando instrucciones del programa. El almacenamiento de las instrucciones en la memoria puede ser lentos, haciendo que el rendimiento baje, además también se almacena la configuración interna del CPU en los registros internos [9].



*Figura 15 Estructura interna de un registro*

Los principales registros de una CPU son:

**Contador de programa:** almacena la dirección de la siguiente instrucción a ejecutar.

**Registro de instrucción:** almacena la instrucción capturada en la memoria y la ejecutada en el momento.

**Registro de estado:** formada por una serie de bits los cuales dan el resultado obtenido en la última operación del ALU.

**Registro acumulador:** almacena los resultados de las operaciones aritméticas y lógicas.

## 5.8 Sistema de interconexión (Buses)

Es el mecanismo que permite el flujo de datos entre los componentes del computador. Desde aquí se distribuyen las señales eléctricas que se interpretan como unos y ceros lógicos. Cuando hay más de un dispositivo en el mismo bus, uno enviará una señal que será procesada por los demás módulos. Si se mandan datos al mismo tiempo habrá un error o una contención del bus. La funcionalidad de los buses se divide en:

**Buses de datos:** se utiliza para transmitir datos entre los diferentes componentes de un computador.

**Buses de direcciones:** indica la posición del dato que se requiere acceder.

**Bus de control:** seleccionan al emisor y al receptor en una transacción del bus.

**Bus de alimentación:** proporciona los distintos voltajes requeridos a los dispositivos.

## **5.9 Periféricos**

Son dispositivos que permiten la entrada y salida de datos del computador una vez procesada. Puede definirse como un conjunto de transductores entre la información física externa y la información binaria interpretable por el computador.

### **5.10 Entrada/Salida**

Maneja los datos entre el exterior y el computador, en este se encuentran los controladores de periféricos que construyen la interfaz, como la memoria y el procesador.

Existen varios métodos para manejar los dispositivos de entrada y salida.

#### **5.10.1 Acceso directo a la memoria (DMA)**

La CPU ordena los buses de direcciones y datos en triestado. Un dispositivo controlador de DMA controla los buses y pasa los datos entre los dispositivos de E/S y la memoria directamente.

#### **5.10.2 Técnicas de Interrupción**

El periférico enciende las líneas de interrupción de la CPU, deteniendo el programa en ejecución y cargando el contador del programa con la dirección de inicio.

#### **5.10.3 Tratamiento de las E/S como posiciones de memoria**

Emplea las mismas instrucciones para acceso a memoria que para E/S. una parte de la memoria es reservada para los registros de los dispositivos de E/S. estas zonas se llaman Puertos de E/S.

## **6 Códigos de Representación Numérica y No Numérica:**

En los sistemas digitales, su información se representa en base a 0 y 1 llamados bits, los cuales se utilizan para distintos códigos de representación pueden clasificarse entre numéricos y no numéricos.

En la representación numérica podemos denotar que hay distintos subconjuntos como los son:

## **6.1 Código Binario Puro**

Esta es la manera más básica de representar los números en un sistema informático. Se basa en el sistema de numeración binaria la cual cada cifra es un bit que puede llegar a tener solo 2 valores: 0 y 1. Este permite representar cualquier número en base 2.

## **6.2 Código BCD (Binary Coded Decimal)**

Este código se representa con cada dígito decimal por sí mismo en binario, se utiliza 4 bits por dígito. Útil en apps donde se requieran iterar la separación de cifras decimales como las calculadoras digitales [10].

## **6.3 IEEE 754**

Llamado también punto flotante IEEE754 se usa para representar números reales (decimales o de punto flotante) es el estándar en ello, en formato binario permite realizar operaciones básicas como suma, resta, multiplicación y divisiones en ellas, ayuda a profesionales en su área hagan con precisión y coherencia en todos los sistemas informáticos [11].

## **6.4 Códigos De Representación No Numérica**

Estos como su propio nombre lo dice no numéricos se usan en si representándose en letras, signos de puntuación, caracteres especiales y otros símbolos no numéricos [12].

Permiten que los comandos y textos sean interpretados de manera digital de los cuales los más conocidos son:

## **6.5 ASCII (American Standard Code for Information Interchange)**

El más utilizado por no decir estándar para representar caracteres en computadoras. Para representar 128 caracteres solo hace falta usar 7 bits incluye letras tanto mayúsculas como minúsculas, números, signos de puntuación y además de algunos caracteres de control. Existe una versión extendida la cual aparte de incluir caracteres adicionales ocupa de 8 bits [13].

## **6.6 EBCDIC**

Creada por IBM por los años 60, para su línea principal de computadoras industriales, esta permite codificar letras, números signos de puntuación y caracteres de control manejan y usando solo 8 bits por carácter, cosa que da un total de 256 combinaciones totales [14].

DE TEXTO A BINARIO Y VICEVERSA											
Letra	Binario	Letra	Binario								
A	1000001	O	1001111								
B	1000010	P	1010000								
C	1000011	Q	1010001								
D	1000100	R	1010010								
E	1000101	S	1010011								
F	1000110	T	1010100								
G	1000111	U	1010101								
H	1001000	V	1010110								
I	1001001	W	1010111								
J	1001010	X	1011000								
K	1001011	Y	1011001								
L	1001100	Z	1011010								
M	1001101	(espacio)	1000000								
N	1001110		1001111								

H	O	L	A	M	U	N	D	O
1001000	1001111	1001100	1000001	1001101	1010101	1001110	1000100	1001111

1000011	1001111	1000100	1001001	1000111	1001111
C	O	D	I	G	O

Figura 16 Conversiones entre bases numéricas

## 7 Conclusión

El estudio de los sistemas numéricos, el flujo de datos y los códigos de representación permite comprender los principios fundamentales del procesamiento digital. Estos conceptos son la base sobre la cual se construyen los sistemas informáticos modernos, y su dominio es esencial para estudiantes y profesionales de las ciencias computacionales.

## 8 Bibliografía

- [1] E. Gregorio, Q. Gutiérrez, V. Manuel, H. Suárez, and A. Morales González, "UN MUNDO DE UNOS Y CEROS. NUEVOS RECURSOS DIDÁCTICOS DEL SISTEMA DE NUMERACIÓN BINARIO EN EDUCACIÓN PRIMARIA," 2020. Accessed: Jun. 07, 2025. [Online]. Available: [https://wp.ull.es/fpiem/wpcontent/uploads/sites/158/2023/07/12\\_10-quevedo-hernandez-morales-unmundo.pdf](https://wp.ull.es/fpiem/wpcontent/uploads/sites/158/2023/07/12_10-quevedo-hernandez-morales-unmundo.pdf)
- [2] M. Beltrán-Escobar, T. E. Alarcón, J. Y. Rumbo-Morales, S. López, G. Ortiz-Torres, and F. D. J. Sorcia-Vázquez, "A Review on Resource-Constrained Embedded Vision Systems-Based Tiny Machine Learning for Robotic Applications," *Algorithms*, vol. 17, no. 11, Nov. 2024, doi: 10.3390/a17110476.
- [3] H. Nan, J. Jiang, J. Zhang, R. Liu, and A. Wang, "Conversion between Number Systems in Membrane Computing," *Applied Sciences (Switzerland)*, vol. 13, no. 17, Sep. 2023, doi: 10.3390/app13179945.
- [4] C. Şimşek, U. Erkan, A. Toktas, Q. Lai, and S. Gao, "Hexadecimal permutation and 2D cumulative diffusion image encryption using hyperchaotic sinusoidal exponential memristive system," *Nonlinear Dyn*, Jul. 2025, doi: 10.1007/s11071025-11001-w.
- [5] J. Xu, B. Zhao, and Z. Wu, "Research on Color Image Encryption Algorithm Based on Bit-Plane and Chen Chaotic System," *Entropy*, vol. 24, no. 2, Feb. 2022, doi: 10.3390/e24020186.

- [6] N. A. Mawla and H. K. Khafaji, "Enhancing Data Security: A Cutting-Edge Approach Utilizing Protein Chains in Cryptography and Steganography," *Computers*, vol. 12, no. 8, Aug. 2023, doi: 10.3390/computers12080166.
- [7] J. Kodua Wiredu, B. Atiyire, N. Seidu Abuba, R. Wiredu Acheampong, and R. Wiredu Acheampong Effi, "Analysis and Optimization Techniques for Base Conversion Algorithms in Computational Sys-tems," *Int J Innov Sci Res Technol*, vol. 9, no. 8, pp. 235–244, 2024, doi: 10.38124/ijisrt/IJISRT24AUG066.
- [8] J. César and R. Casas, "EVALUACIÓN Y ARQUITECTURA DEL COMPUTADOR," 2021. Accessed: Jun. 07, 2025. [Online]. Available: <https://repositorio.usam.ac.cr/xmlui/bitstream/handle/11506/1781/LEC%20ING%20SIST%200001%202021.pdf?sequence=1&isAllowed=y>
- [9] J. R. Osio, W. J. Aróztegui, and J. A. Rapallini, *Sistemas digitales basados en microcontroladores*. Editorial de la Universidad Nacional de La Plata (EDULP), 2020. doi: 10.35537/10915/95305.
- [10] K. Dharamvir and P. Manoranjan, "Area-Optimized BCD-4221 VLSI Adder Architecture for High-Performance Computing," Burla, India, 2024. Accessed: Jun. 07, 2025. [Online]. Available: [https://hit.alljournals.cn/jhit\\_cn/ch/reader/create\\_pdf.aspx?file\\_no=20240303&flag=1&year\\_id=2024&quarter\\_id=3](https://hit.alljournals.cn/jhit_cn/ch/reader/create_pdf.aspx?file_no=20240303&flag=1&year_id=2024&quarter_id=3)
- [11] S. Boldo, C. P. Jeannerod, G. Melquiond, and J. M. Muller, "Floating-point arithmetic," *Acta Numerica*, vol. 32, pp. 203–290, May 2023, doi: 10.1017/S0962492922000101.
- [12] V. Novaković, "Accurate complex Jacobi rotations," Aug. 2023, doi: 10.1016/j.cam.2024.116003.
- [13] D. Hayatpur, B. Hempel, K. Chen, W. Duan, P. J. Guo, and H. Xia, "Taking ASCII Drawings Seriously: How Programmers Diagram Code," in *Conference on Human Factors in Computing Systems - Proceedings*, Association for Computing Machinery, May 2024. doi: 10.1145/3613904.3642683.
- [14] A. Taneja, R. K. Shukla, and R. S. Shukla, "Improvisation of RSA Algorithm in Respect to Time and Security with the Proposed (AEA) Algorithm," in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Aug. 2021. doi: 10.1088/17426596/1998/1/012036.

## **9 Anexos**

### **9.1 Git-Hub**

<https://github.com/MiloSaurio4kHD/SISTEMASNUMERICOSGE>