**Circuitos Lógicos y Sistemas Digitales**

Los circuitos digitales son sistemas electrónicos que operan con señales binarias, 0 y 1 o de voltaje bajo y voltaje alto. Se basan en el álgebra booleana, lo que permite ejecutar operaciones lógicas y aritméticas a través de compuertas lógicas [1]. Estos circuitos son fundamentales en sistemas digitales como computadoras, teléfonos móviles y dispositivos de comunicación, y han evolucionado desde circuitos integrados simples (CI) a chips de alta complejidad como los VLSI y ULSI [2].

**Circuitos secuenciales**

Los circuitos secuenciales son aquellos en los que su salida no solo depende de lo que entra en ese momento, sino también de lo que les pasó antes, es decir, el estado anterior del sistema. Estos guardan información internamente, lo que les da una memoria y hace que su comportamiento cambie con el tiempo. Esto se logra utilizando componentes como Flip-Flops, que retienen un valor hasta que una señal de reloj indique que debe cambiar [3].

Los Flip-Flops se clasifican según su comportamiento: El Flip-Flop SR actúa como una memoria de 1 bit con entradas SET y RESET. RESET pone la salida en 0, y SET la pone en 1. Los D Flip-Flops conservan el valor de su entrada mientras está sincronizado con una señal de reloj, y los JK Flip-Flops son similares a los SR, pero tienen la ventaja de incluir una entrada de reloj y compuertas adicionales que evitan resultados incorrectos. Puede operar en cuatro modos diferentes: establecer la salida 1, a 0, mantener su estado anterior o alternar (toggle) entre 0 y 1 [4].

**Circuitos combinacionales**

Los circuitos combinacionales se forman interconectando varias puertas lógicas. Para que estas funcionen sin problemas deben ser concatenables. La concatenabilidad significa que la señal de salida de una puerta debe ser compatible con la entrada de otras puertas o incluso de circuitos secuenciales, lo que permite que se realicen operaciones lógicas continuas. Para evitar conflictos, tanto las entradas como las salidas deben usar la misma magnitud física. Por ejemplo, si se emplean niveles de voltaje para codificar las entradas, las salidas también deben expresarse mediante voltajes para asegurar una conexión directa entre los componentes [5].

**Sistemas combinacionales básicos**

**Sumadores**

Un sumador es un circuito combinacional constituido por puertas lógicas que recibe dos entradas binarias (o tres entadas, si se incluye un acarreo de entrada o carry-in), y su resultado es el bit de la suma, y un posible bit de acarreo [6].

Existen algunos tipos de sumadores. El sumador medio (Half Adder) realiza la suma de dos bits sin considerar el acarreo de entrada, mientras que el sumador completo (Full Adder) amplia esta función sumando tres bits, incluyendo el acarreo de entrada. Según Nagaraj et al. (2025), el Full Adder puede ser optimizado mediante técnicas como ECRAAL con el objetivo de reducir el consumo energético en diseño VLSI. A partir de este último, nacen estructuras más complejas como el Sumador con acarreo en cascada (Ripple Carry Adder), el Sumador con anticipación de acarreo (Carry Lookahead Adder) y Sumadores aproximados (Approximate Adder), cada uno con ventajas en velocidad, complejidad o eficiencia energética [6].

**Ejemplo de sumadores:**

**Tabla 1.** Tabla de Verdad Sumador

|  |  |  |  |
| --- | --- | --- | --- |
| **A** | **B** | **S** | **Cout** |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 |

Imagen que contiene Tabla

El contenido generado por IA puede ser incorrecto.

**Figura 1.** Sumador

**Restadores**

Los restadores son circuitos combinacionales que realizan la operación de resta binaria, ya sea entre dos bits individuales o entre números binarios de múltiples bits. Hay versiones simples como el restador medio (Half Subtractor), que resta dos bits sin préstamo previo, y el restador completo (Full Subtractor), que si considera un bit de préstamo de entrada (borrow in) [7].

Según Sanadhya y Sharma (2020), los restadores completos pueden implementarse eficientemente usando lógica adiabática, lo que permite disminuir tanto el consumo energético como el retardo en comparación con diseños CMOS comunes [8].

**Ejemplo de restadores:**

**Tabla 2.** Tabla de Verdad

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A** | **B** | **Bin** | **D** | **Bout** |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

Diagrama, Esquemático

El contenido generado por IA puede ser incorrecto.

**Figura 2**. Restador

**Multiplexores**

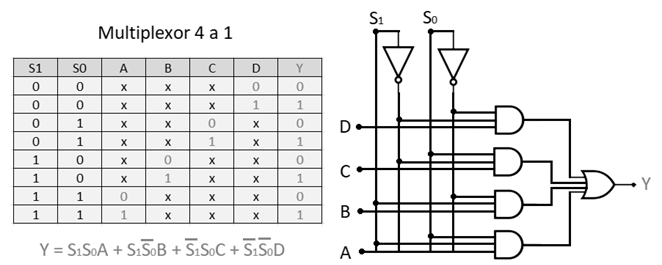
Un multiplexor es un circuito lógico combinacional que permite la selección de una única salida entre múltiples señales de entrada, utilizando un conjunto de líneas de control binarias. Su función principal es optimizar el uso de recursos al permitir que varios datos compartan un mismo canal de transmisión o línea de procesamiento. Los tipos más comunes de multiplexores son los de 2:1, 4:1 y 8:1, donde el número que sigue a los dos puntos indica cuántas entradas de datos son supervisadas por las líneas de control [9].

Según Bansal, Singh y Sharma (2021), estas estructuras son esenciales en sistemas digitales como microprocesadores y redes de comunicación, debido a que permiten reducer el número de líneas físicas necesarias y optimizar el flujo de datos en aplicaciones embebidas [9].

**Ejemplo de multiplexores:**

**Tabla 3**. Tabla de Verdad Multiplexor

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S1** | **S0** | **A** | **B** | **C** | **D** | **Y** |
| 0 | 0 | x | X | X | 0 | 0 |
| 0 | 0 | X | X | X | 1 | 1 |
| 0 | 1 | X | X | 0 | X | 0 |
| 0 | 1 | X | X | 1 | X | 1 |
| 1 | 0 | X | 0 | X | X | 0 |
| 1 | 0 | X | 1 | X | X | 1 |
| 1 | 1 | 0 | X | X | X | 0 |
| 1 | 1 | 1 | x | X | X | 1 |



**Figura 3.** Multiplexor

**Microprocesador y Microcontroladores**

**Microprocesador**

Un microprocesador es un circuito integrado de tamaño reducido, compuesto por miles o millones de transistores, el cual tiene como función principal recibir datos de entrada, procesarlos siguiendo un conjunto de instrucciones específicas y entregar resultados como salida. Se encuentra en el núcleo de la CPU de una computadora, así como en otros sistemas electrónicos, siendo el responsable de ejecutar operaciones aritméticas, lógicas y de control con gran velocidad y precisión [10].

Para Abdullayevich (2020), el microprocesador es el elemento central del sistema computacional, encargado de coordinar y ejecutar los procesos internos de la computadora [11].

Según Antonov y Krasnyuk (2021), el microprocesador está constituido por diversos componentes que operan en conjunto para ejecutar instrucciones de manera eficaz. Entre estos se destaca la unidad de control que interpreta y dirige la ejecución de instrucciones, la unidad aritmético-lógica (ALU por sus siglas en inglés) encargada de las operaciones matemáticas y lógicas, registros internos que almacenan datos temporalmente, y la memoria caché que acelera el acceso a información frecuente [12].

**Velocidad de procesamiento y ciclo de instrucciones**

La velocidad de procesamiento de un microprocesador depende de varios factores que dependen uno de otros, como la frecuencia de reloj, la eficiencia de instrucciones y la arquitectura interna [13].

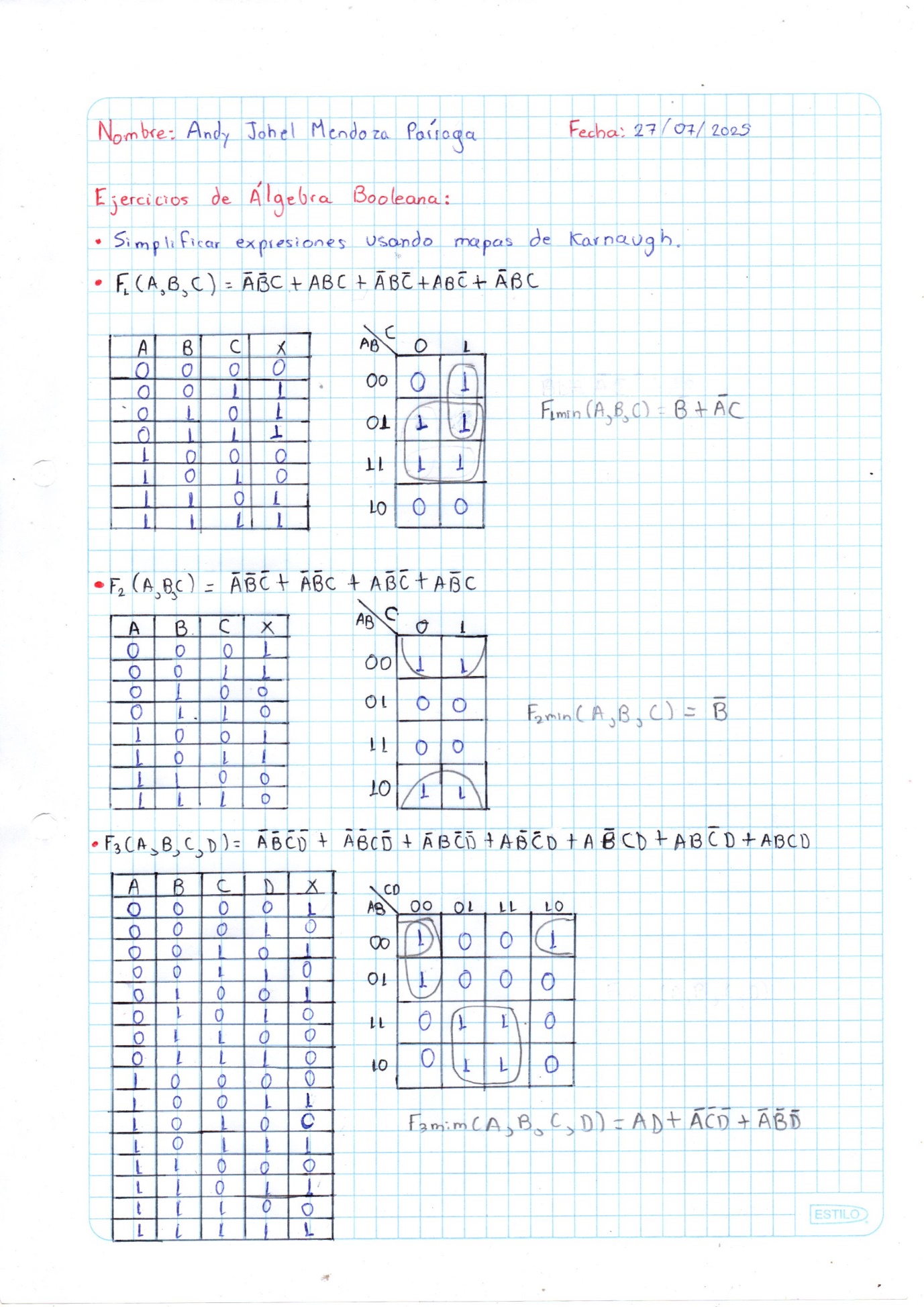
Suarez, Almeida y Blanco (2024), determinaron que, aunque las implementaciones de la arquitectura RISC-V puede presentar un menor consumo promedio de energía en comparación con ARM, esto no quiere decir que sea superior en el rendimiento por vatio. Es decir, una menor potencia consumida no siempre se traduce como mayor eficiencia energética si el procesador no puede ejecutar instrucciones de manera eficiente [13].

El ciclo de instrucciones en los microprocesadores consta de cuatro etapas: búsqueda de la instrucción en memoria, decodificación para interpretar la operación, ejecución de la instrucción, acceso a memoria si es necesario y escritura del resultado en registros. Nikolic et al. (2022), señalan que la duración y eficiencia de este ciclo afectan directamente a la velocidad de procesamiento, debido a que cuanto más corto sea un ciclo, más instrucciones se pueden ejecutar por segundo, además que el uso de técnicas como la segmentación o pipelining, mejoran el rendimiento al procesar múltiples instrucciones de forma simultánea en diferentes fases del ciclo [14].

**Procedimientos**

**Ejercicios de Álgebra Booleana:**

**Simplificar expresiones usando mapas de Karnaugh**



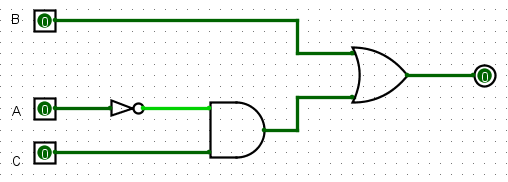
**Diseñar y probar circuitos lógicos básicos utilizando software de simulación.**

Para realizar la simulación de los circuitos lógicos, se empleó el software Open Source “Logisim Evolution”, y se diseñó los circuitos lógicos con el resultado de la simplificación de expresiones usando mapas de Karnaugh del ejercicio anterior.

* **F1(A, B, C) =**

El resultado de la simplificación es: , lo que nos quiere decir que la salida será alta cuando B sea de nivel alto o cuando A sea de nivel bajo y C sea de nivel alto.

Circuito:



Cuando B es de nivel alto:

Diagrama

El contenido generado por IA puede ser incorrecto.

Cuando A es de nivel bajo y C es de nivel alto:

Diagrama

El contenido generado por IA puede ser incorrecto.

Cuando A, B y C son de nivel alto:

Diagrama

El contenido generado por IA puede ser incorrecto.

Todo esto comprueba de que si se cumple lo que dice la expresión simplificada.

* **F2(A, B, C) =**

El resultado de la simplificación es: , lo que quiere decir que la salida será de nivel alto únicamente cuando B sea de nivel bajo (negada), y en el circuito como A y C no inciden en el resultado de salida se las manda a tierra.

Circuito:

Diagrama

El contenido generado por IA puede ser incorrecto.

Cuando B es de nivel alto:

Diagrama

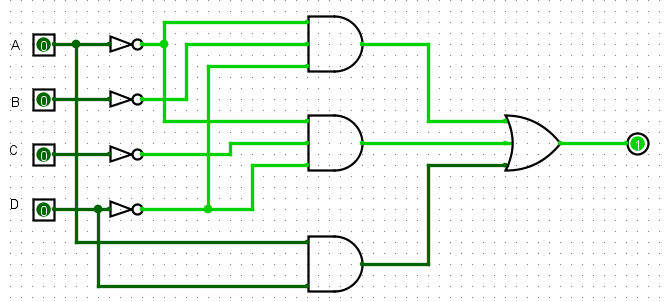
El contenido generado por IA puede ser incorrecto.

Esto refleja que si se cumple lo que está en la expresión simplificada

* **F3(A, B, C, D) =**

El resultado de la simplificación es: , lo que quiere decir que la salida será de nivel alto cuando A y D sean de nivel alto, o A, C y D sean de nivel bajo (negadas), o cuando A, B y D sean de nivel bajo (negadas).

Circuito:



Cuando A y D son de nivel alto:

Diagrama

El contenido generado por IA puede ser incorrecto.

Cuando A, C y D son de nivel bajo (negadas):

Diagrama

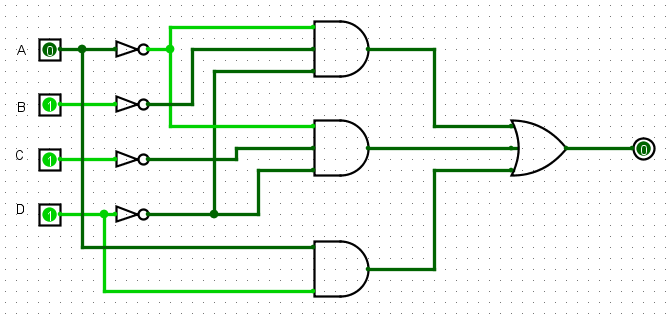
El contenido generado por IA puede ser incorrecto.

Cuando A, B y D son de nivel bajo (negadas):

Diagrama, Esquemático

El contenido generado por IA puede ser incorrecto.

Cuando A es de nivel bajo y las demás son de nivel alto:



Todo esto comprueba de que la expresión simplificada es correcta.

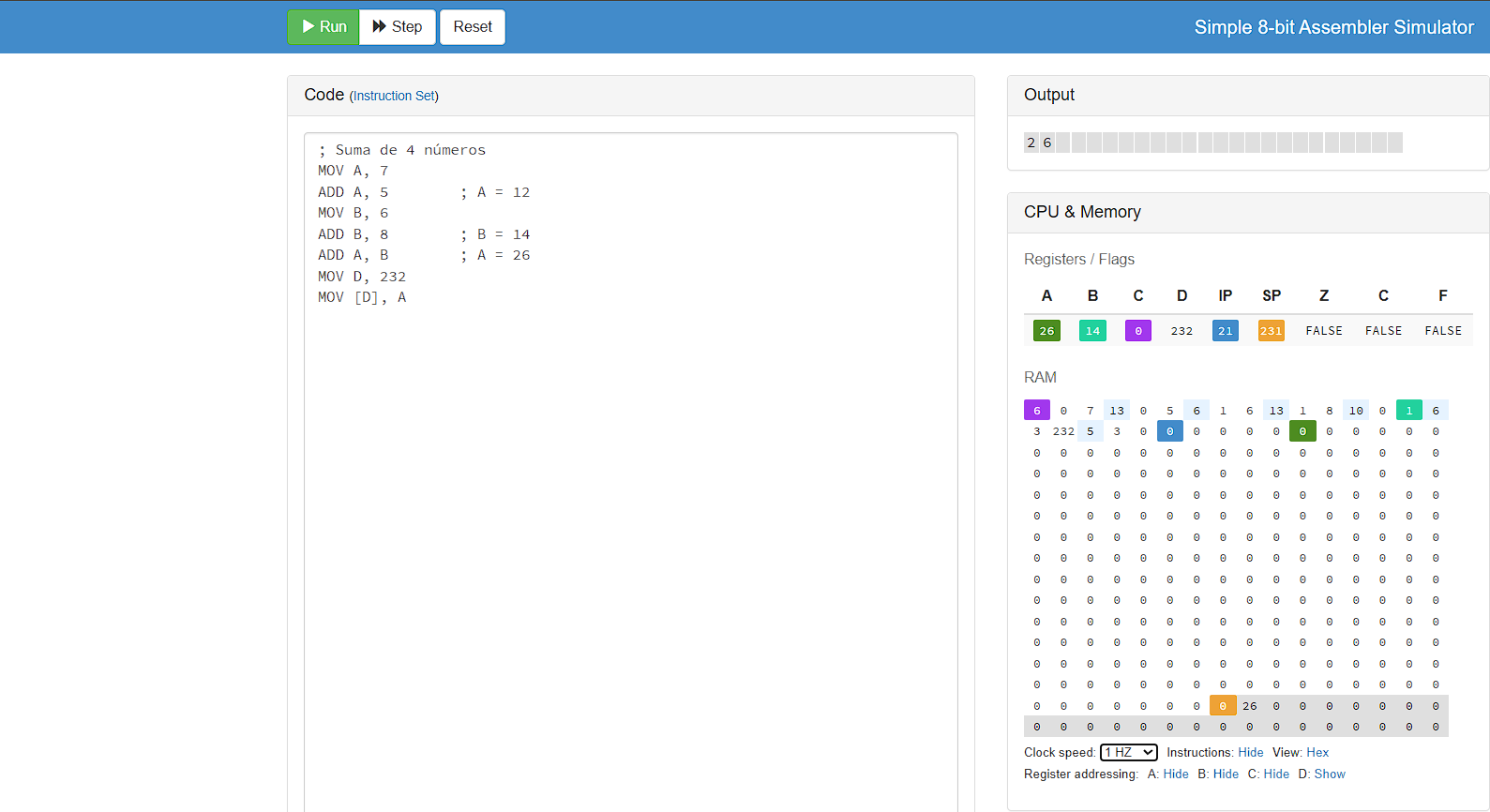
**Estudio de Microprocesadores y Microcontroladores:**

**Analizar el ciclo de instrucciones y la velocidad del microprocesador con ejemplos prácticos.**

Para realizar este análisis se hicieron simulaciones prácticas en un simulador online de arquitectura de computadoras denominado “Simple 8-bit Assembler Simulator”, que emula una arquitectura básica similar a la del Intel 8086.

Para el análisis se codificó en lenguaje ensamblador utilizando instrucciones como MOV, ADD, entre otras, que permitieron observar el funcionamiento básico del ciclo de instrucciones y el cambio en los registros durante la ejecución, con el objetivo de determinar la velocidad del microprocesador en distintas velocidades de reloj como 1Hz, 4Hz, 8Hz y 16Hz.

**Primer ejemplo: Suma de 4 números**

****

En este ejemplo existen 7 ciclos de instrucción, ya que en ensamblador cada instrucción ejecutable se considera un ciclo de instrucción.

Al evaluar la ejecución a 1Hz, el tiempo que le tomó es de 7 segundos, ya que 1Hz equivale a una instrucción por segundo. A una frecuencia de 4 Hz, el tiempo de ejecución fue de 1.75 segundos. Al evaluarlo en 8 Hz, la ejecución tardó en finalizar en 875 milisegundos. En 16 Hz le tomó un tiempo de 437.5 milisegundos. Y si lo evaluamos con la frecuencia promedio de un microprocesador de la actualidad que va a 3.7 GHz, solo le tomaría 1.8919 nanosegundos.

**Ejemplo 2: Suma 1 en el registro A hasta que B sea 0**

Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

Este ejemplo cuenta con 2 ciclos de instrucción iniciales, 3 en el loop que multiplicado por 5 iteraciones serían 15 ciclos de instrucciones y 1 en el final, en total 18 ciclos de instrucciones.

Al evaluar la ejecución a una frecuencia de 1Hz le tomó 18 segundos. A una frecuencia de 4 Hz tardó 4.5 segundos. Con una frecuencia de 8 Hz demoró 2.25 segundos en finalizar la ejecución. Y a una frecuencia de 16 Hz solo le tomó 1.125 segundos en finalizar. Si lo evaluamos a una frecuencia promedio de un microprocesador de la actualidad que va a 3.7 GHz, sería prácticamente imperceptible tomándole solo 4.87 nanosegundos aproximadamente.

**Referencias**

[1] N. I. Georgiev, V. V. Bakov, and V. B. Bojinov, “A Tutorial Review on the Fluorescent Probes as a Molecular Logic Circuit—Digital Comparator,” *Molecules*, vol. 28, no. 17, p. 6327, Aug. 2023, doi: 10.3390/molecules28176327.

[2] S. S. Srikant and P. K. Chaturvedi, *Basic Electronics Engineering*, 1st ed. Singapore: Springer Singapore, 2020. doi: 10.1007/978-981-13-7414-2.

[3] J. F. Groote, R. Morel, J. Schmaltz, and A. Watkins, *Logic Gates, Circuits, Processors, Compilers and Computers*. Cham: Springer International Publishing, 2021. doi: 10.1007/978-3-030-68553-9.

[4] Neha M Harapanhalli, Kavipriya M, Ishwari Jigajinni, and Dr. Keerti Kulkarni, “Implementation of Combinational and Sequential Logic Circuits using Quantum Computing,” *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, pp. 430–439, Jun. 2023, doi: 10.32628/CSEIT23903106.

[5] S. Bandyopadhyay, “Nanomagnetic Boolean Logic—The Tempered (and Realistic) Vision,” *IEEE Access*, vol. 9, pp. 7743–7750, 2021, doi: 10.1109/ACCESS.2021.3049333.

[6] S.Nagaraj, G. M. Sreerama Reddy, and S. Aruna Mastani, “Design and Analysis of Novel Full Adder using ECRAAL,” *International Journal of Computational and Experimental Science and Engineering*, vol. 11, no. 1, Jan. 2025, doi: 10.22399/ijcesen.848.

[7] A. Lloris Ruiz, E. Castillo Morales, L. Parrilla Roure, A. García Ríos, and M. J. Lloris Meseguer, *Arithmetic and Algebraic Circuits*, vol. 201. Cham: Springer International Publishing, 2021. doi: 10.1007/978-3-030-67266-9.

[8] M. Sanadhya and D. K. Sharma, “Design and implementation of full subtractor using different adiabatic techniques,” in *2020 IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE)*, IEEE, Dec. 2020, pp. 102–106. doi: 10.1109/WIECON-ECE52138.2020.9397967.

[9] M. Bansal, H. Singh, and G. Sharma, “A Taxonomical Review of Multiplexer Designs for Electronic Circuits &amp; Devices,” *Journal of Electronics and Informatics*, vol. 3, no. 2, pp. 77–88, Apr. 2021, doi: 10.36548/jei.2021.2.001.

[10] A. Al Zaman and N. J. Monira, “AN OVERVIEW OF MICROPROCESSORS AND ASSEMBLY LANGUAGE PROGRAMMING ,” *Advances in Interconnect Technologies: An International Journal (AITIJ)* , vol. 1, no. oct, Sep. 2022, doi: 10.5281/zenodo.7081411.

[11] H. Z. Abdullayevich, “History, Structure And Types Of Microprocessors,” *The American Journal of Interdisciplinary Innovations and Research*, vol. 02, no. 11, pp. 39–46, Nov. 2020, doi: 10.37547/tajiir/Volume02Issue11-08.

[12] A. A. Antonov and A. A. Krasnyuk, “The internal structure of microprocessors for industrial control and data processing systems,” *IOP Conf Ser Mater Sci Eng*, vol. 1061, no. 1, p. 012003, Feb. 2021, doi: 10.1088/1757-899X/1061/1/012003.

[13] D. Suárez, F. Almeida, and V. Blanco, “Comprehensive analysis of energy efficiency and performance of ARM and RISC-V SoCs,” *J Supercomput*, vol. 80, no. 9, pp. 12771–12789, Jun. 2024, doi: 10.1007/s11227-024-05946-9.

[14] G. Nikolic, B. Dimitrijevic, T. Nikolic, and M. Stojcev, “Fifty years of microprocessor evolution: from single CPU to multicore and manycore systems,” *Facta universitatis - series: Electronics and Energetics*, vol. 35, no. 2, pp. 155–186, 2022, doi: 10.2298/FUEE2202155N.