TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE TLAXIACO

# INVESTIGACIÓN 3. “INTERPRETACIÓN DE CEROS Y UNOS A NIVEL DE HARDWARE”:


**Presenta:**

- Gonzalez Lita Daisy - 22620056


**Carrera:**

Ingeniería en Sistemas Computacionales

**Asignatura:**

Arquitectura de Computadoras

**Docente:**

Ing. Edward Osorio Salinas

Tlaxiaco, Oaxaca, a 10 de octubre de 2024.

*“Educación, Ciencia y Tecnología, Progreso día con día”®*

# ÍNDICE

# INTRODUCCIÓN

En el presente trabajo se presenta una recopilación de información acerca de la interpretación de ceros y unos a nivel de hardware. La estructura es la siguiente en primer lugar se encuentra el objetivo, en segundo lugar, los materiales ocupados, en tercer lugar, se encuentra el desarrollo, en donde se encuentra toda la información recopilada, en cuarto lugar, una conclusión y, por último, las referencias bibliográficas. Este trabajo corresponde a la materia de Arquitecturas de Computadoras, de la carrera de Ingeniería en Sistemas Computacionales.

## OBJETIVO

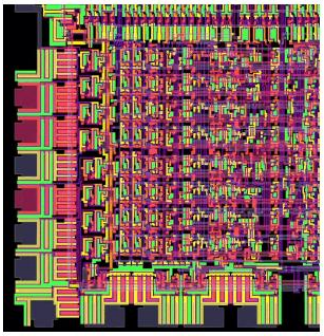
- ✓ Recopilar información acerca de la interpretación de ceros y unos a nivel de hardware.

## MATERIALES

- ➢ Computadora con acceso a internet.
- ➢ Word.

## DESARROLLO

### Los bits dentro de la computadora



Nuestras computadoras digitales hablan el lenguaje de los bits. Un bit es la unidad mínima de información y solo puede tener dos valores (que convencionalmente fijamos en 0 y 1).

Por muy complejas que parezcan ser, las computadoras solo pueden manejar ceros y unos. Pero no existe un número cero o uno dibujado en los cables de las computadoras. Tampoco hay lamparitas adentro de los chips para indicar que se ha recibido un 1 o un 0.

Los valores lógicos, en la electrónica digital, se manejan como magnitudes de voltajes altos o bajos. Los valores que representan al valor lógico 1 están cerca de 5V (o algo menos) y los valores del valor lógico o están cerca de 0V.

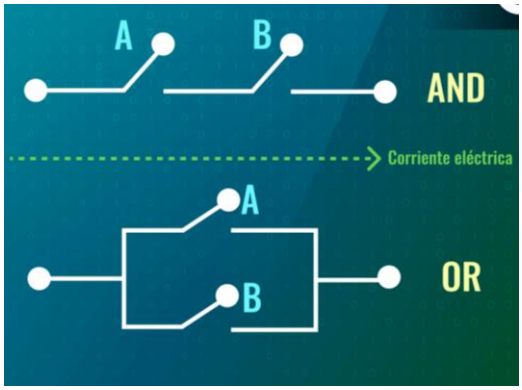
Como nosotros no podemos ver voltajes; no podemos ver los bits fluir por las patitas de los chips o por las pistas de los circuitos impresos. Pero, aunque pudiéramos ver voltaje, la velocidad con que los bits se mueven por los conductores sería tan alta que probablemente solo veríamos unos cuantos destellos. (Hinostroza, 2023)

### El sistema binario en las computadoras

En 1937, Claude Shannon publicó el artículo “Análisis simbólico de circuitos de relés y de conmutación”. Una obra que ha sido considerada la base de la informática moderna, pues unía el álgebra booleana con los circuitos. Mientras trabajaba en los

laboratorios Bell quedó maravillado de cómo se usaban los relés y los interruptores para manejar el tráfico de las llamadas telefónicas y redireccionarlas. Y se le ocurrió que usando ese principio y el álgebra de Boole se podría ejecutar cualquier operación matemática usando interruptores, que no son más que ceros (apagado) y unos (encendidos).

Por ejemplo, una operación AND consistiría de dos interruptores en serie. Tienen que estar los dos encendidos (1) para que pase la corriente. Mientras que un OR serían dos interruptores en paralelo. Basta que uno de los dos esté prendido (1) para que pase la corriente.




Este principio tan sencillo, replicado cientos o miles de veces en muchos interruptores, podía resolver cualquier problema matemático o lógico y sería la base para las computadoras.

De este trabajo derivan las famosas puertas lógicas, que son una forma de implementar en un circuito electrónico el álgebra booleana usando los principios de Claude Shannon.

**Del binario al ensamblador y a los lenguajes de programación**

El binario es excelente para las computadoras porque pueden hacer cálculos a velocidades increíbles, pues solo emplean dos valores (1 y 0) en lugar de cientos o miles. Pero, para los programadores es una tortura escribir en binario.

Así que el siguiente paso fue crear un lenguaje que fuera más fácil de entender y de recordar y así nació el lenguaje ensamblador. Este lenguaje es un conjunto de

órdenes directas al procesador (como hacer cálculos matemáticos) escritas en abreviaturas que hacen mucho más fácil programar.

Este lenguaje consta de dos partes: el lenguaje ensamblador en sí mismo, y el ensamblador, que es un programa que se encarga de convertir las instrucciones de este lenguaje en binario. Porque las computadoras no entienden ensamblador, solo entienden ceros y unos.

Y aunque el lenguaje ensamblador es una forma más sencilla frente a programar en binario, seguía siendo un trabajo arduo y lento, por lo que se crearon los lenguajes de programación de alto nivel. Alto nivel significa que son más fáciles de comprender para los seres humanos. Mientras que bajo nivel significa que son menos fáciles de entender, pero con más control sobre cada instrucción dada al procesador.

Estos lenguajes deben ser convertidos a ceros y unos porque las computadoras no entienden ni Python ni Javascript ni Java ni Go, ni ninguno de estos lenguajes. Solo entienden ceros y unos. Y para convertir los lenguajes de programación a binario se creó un programa llamado “compilador”.

El primer compilador fue creado por Grace Hooper en 1952 para un lenguaje llamado Flow-Matic que no llegó a ser utilizado ampliamente. El primer lenguaje de alto nivel de la historia, utilizado en muchas computadoras y empresas, fue Fortran, lanzado en 1957. Y el segundo lenguaje fue COBOL, lanzado en 1959 y que hasta la fecha se sigue usando muchísimo, sobre todo en el mundo financiero. (Chavez, 2024)

**Interruptores**

El elemento básico de todas las computadoras es un interruptor, no un interruptor de luz que debe presionar con las manos, sino un interruptor que puede cambiar de posición según el voltaje o la corriente que le ponga. El ejemplo clásico es un relé, donde el interruptor controlado por el relé está abierto o cerrado dependiendo de la entrada de corriente al relé.

Ahora, podría hacer una computadora con relés, pero tienden a ser bastante grandes o lentos. En su lugar, utilizan transistores, que pasan o no pasan corriente o voltaje entre 2 terminales, dependiendo de la señal, alto o bajo voltaje, en un tercer terminal.

Ahora, si agrupa varios de estos conmutadores de transistores, puede pensar que representan un número. Imagina que tengo 4 interruptores y pongo señales en sus entradas para que las salidas sean de bajo voltaje o alto voltaje. Para ahorrar escritura, puedo llamar a bajo voltaje "0" y alto voltaje "1", entonces los estados de mis interruptores de transistores son 0,1.

Resulta que, si combina estos interruptores de manera inteligente, puede hacer funciones lógicas como Y (una entrada de 2 1 da un 1, cualquier otra cosa da 0) y OR (una entrada de 2 0 da 0, cualquier otra cosa da 1). Incluso puede combinar los AND y los OR para hacer aritmética básica simple. El proceso de interpretar lo que está sucediendo en una computadora a niveles cada vez más altos se llama "abstracción". La computadora simplemente maneja combinaciones increíblemente complejas de estados de conmutación, que ocurren muchos millones de veces por segundo. Los humanos asignamos significado a lo que está sucediendo; de hecho, la programación es en cierto sentido el arte de construir un conjunto de instrucciones que la computadora puede ejecutar y que tendrán un significado útil para los humanos. (SaGA, 2024)

**Memoria**

La memoria del sistema (RAM, ROM, etc.) almacena ceros y unos en ubicaciones de memoria. Cada celda de memoria puede contener un bit (0 o 1), y estas celdas están organizadas en bytes (generalmente 8 bits por byte).

El contenido de las memorias no es otra cosa que dígitos binarios o bits (binary digits), que se corresponden con dos estados lógicos: el 0 (cero) sin carga eléctrica y el 1 (uno) con carga eléctrica. A cada uno de estos estados se le llama bit, que es la unidad mínima de almacenamiento de datos. (Wikipedia, 2015)

**Comunicación con periféricos**

El hardware traduce los ceros y unos en señales físicas que se transmiten entre la CPU, la memoria y los dispositivos de entrada/salida. Por ejemplo, cuando un teclado se presiona, el hardware del teclado envía una señal binaria a la CPU, la cual la interpreta y la convierte en el carácter correspondiente. (Rojas, 2024)

**Ceros físicos y lógicos**

Unos ceros lógicos (0) es una representación abstracta de la información en un sistema digital. En el contexto de circuitos lógicos y sistemas de computación, se refiere a un estado "bajo" o "apagado". En términos de lógica booleana, unos ceros lógicos se pueden considerar como un valor que indica "falso". Se utilizan en la lógica digital, diseño de circuitos, programación y en la arquitectura de computadoras.

En cambio, los ceros físicos (0) se refiere a una representación tangible en hardware. Es el estado en el que una señal o un componente electrónico está en un nivel de voltaje específico (por ejemplo, 0 voltios) que indica que el componente está apagado o no está conduciendo corriente. Se refieren a la implementación concreta en componentes electrónicos como transistores, memorias, y circuitos integrados.

Estos ceros físicos dependen de la tecnología específica utilizada (por ejemplo, CMOS, TTL) y del estado físico de los componentes. (Ortiz, 2024)

## CONCLUSIÓN

Al terminar la recopilación de información puede entender un poco más el funcionamiento de las computadoras digitales, el cual se basa en la representación binaria de los datos mediante ceros y unos, conocidos como bits, que son la unidad mínima de información. A nivel físico, estos bits son representados por señales de voltaje, donde el 0 está asociado a un voltaje bajo o nulo, y el 1 a un voltaje más alto. Aunque los humanos no podemos observar directamente el flujo de bits en los circuitos, estos operan a velocidades extremadamente altas a través de transistores y otros componentes electrónicos, que actúan como interruptores para procesar las señales.

La base teórica de esta operación se encuentra en el trabajo de Claude Shannon, quien unió el álgebra booleana con circuitos eléctricos para realizar operaciones lógicas, lo que permitió el desarrollo de las puertas lógicas fundamentales para las computadoras. Además, aunque las computadoras operan solo en binario, los lenguajes de programación como el ensamblador y los lenguajes de alto nivel se crearon para facilitar la interacción entre humanos y máquinas, convirtiendo las instrucciones comprensibles para los programadores en código binario mediante programas llamados ensambladores o compiladores.

## REFERENCIAS BIBLIOGRÁFICAS


Chavez, A. F. (14 de Octubre de 2024). *¿Por qué las computadoras solo entienden 0 y 1? (Código binario)*. Obtenido de ¿Por qué las computadoras solo entienden 0 y 1? (Código binario): https://ed.team/blog/por-que-las-computadoras-solo-entienden-0-y-1-codigo-binario

Hinostroza, T. (9 de Octubre de 2023). *Mundo Digital – Cap. 1 – Ceros y unos*. Obtenido de Mundo Digital – Cap. 1 – Ceros y unos: https://blogdetito.com/2023/10/09/mundo-digital-cap-1-ceros-y-unos/

Ortiz, D. (14 de Octubre de 2024). *Sistema binario: unos y ceros a través de la historia.* Obtenido de Sistema binario: unos y ceros a través de la historia: https://blogthinkbig.com/sistema-binario/

Rojas, J. (14 de Octubre de 2024). *Entrada-Salida (I/0) y comunicacion .* Obtenido de Entrada-Salida (I/0) y comunicacion : https://www.ele.uva.es/~ruth/DocenciaMicropro_archivos/tema5.pdf

SaGA, J. (10 de Octubre de 2024). *¿Cómo saben las computadoras qué hacer son "1" y "0"?* Obtenido de ¿Cómo saben las computadoras qué hacer son "1" y "0"?: https://es.quora.com/Cómo-saben-las-computadoras-qué-hacer-con-1-y-0-Siempre-que-trato-de-averiguarlo-es-solo-gente-que-explica-el-binario-sin-decir-nunca-cómo-sabe-una-computadora-por-ejemplo-tomar-0110-y-convertirlo-en

Wikipedia. (31 de Enero de 2015). *Memoria principal*. Obtenido de Memoria principal: https://es.wikipedia.org/wiki/Memoria_principal#:~:text=El%20contenido%20de%20las%20memorias,mínima%20de%20almacenamiento%20de%20datos.