TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE TLAXIACO

INGENIERÍA EN SISTEMAS COMPUTACIONALES

DESARROLLO DE SOFTWARE ISIE-DES-2022-01

SCD1003 -ARQUITECTURA DE COMPUTADORAS

CÁTEDRA DEL ING. OSORIO SALINAS EDWARD

ALUMNOS:

No	Nombre	No de Control
01	Cruz Ramírez Jaczibeth	22620233

GRUPO:

5BS

INVESTIGACIÓN

Tlaxiaco, Oaxaca. A 14 de Oct. Del 2024

Introducción

La interpretación de los valores binarios 0 y 1 a nivel de hardware es fundamental para entender cómo funcionan las computadoras y otros dispositivos digitales. Este concepto tiene sus raíces en el sistema de numeración binario, que fue formalizado por el matemático Gottfried Leibniz en el siglo XVIII. Aunque las primeras calculadoras mecánicas usaban el sistema decimal, la implementación del sistema binario en el hardware digital moderno se consolidó gracias al trabajo de pioneros como George Boole y Claude Shannon. Estos números representan estados eléctricos en los circuitos: 0 como apagado y 1 como encendido, lo cual es la base del procesamiento de información en la máquina.

INTERPRETACIÓN A 0 Y 1 A NIVEL DE HARDWARE

El origen de los números

Para entender a los números binarios, retrocedamos un poco en la historia. Los antiguos griegos usaban las letras de su alfabeto para contar. Alfa era uno, Beta era dos, Gama era 3 y así sucesivamente. Era un sistema complicado para cálculos complejos, pero servía para la matemática básica.

Más adelante, los romanos usaron un sistema basado en combinaciones de letras para su numeración: I para uno, V para cinco, L para cincuenta, C para 100. Así que 157 se expresaba así: CLVII. Más sencillo que el sistema griego, pero igual de complicado para la matemática avanzada.

En la Edad Media se introdujo en Europa el sistema numérico decimal que había sido inventado en la India y que usaban los árabes para comerciar. En el año 1202, Leonardo de Pisa publicó el libro “Liber abaci” (“El libro del cálculo” en español) en el cual explica la ventaja del sistema decimal indo-arabigo para actividades como el comercio, cálculos de interés o conversiones de monedas. En este libro también se presenta la famosa sucesión de Fibonacci (nombre con el que era conocido Leonardo de Pisa) en la que cada número de la serie es la suma de los anteriores.

1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987…

Desde entonces el sistema decimal ha sido el más usado en el mundo. En 1642, Blaise Pascal construyó la primera calculadora mecánica importante de la historia, la llamó Pascalina y podía realizar las cuatro operaciones (aunque la multiplicación y la división la hacía con secuencias de sumas y restas).

En 1671, inspirado por el trabajo de Pascal, Leibniz crea su propia calculadora que podía multiplicar y dividir directamente sin convertir a sumas y restas. Estas dos calculadoras serían la inspiración, más de 100 años después, para la máquina diferencial de Charles Babbage, quien es considerado el padre de las computadoras.

Sin embargo, estas máquinas trabajaban con el sistema decimal (contando del 0 al 9) no con el binario.

En 1703 el mismo Leibniz publicó el artículo “Explicación de la aritmética binaria” en el que explica cómo funciona el sistema binario (donde solo hay ceros y unos) y cómo hacer cálculos con ellos. Su trabajo estaba inspirado en los de matemáticos chinos que ya conocían este sistema y ayudó a difundir el sistema binario como en Europa.

¿Cómo funciona el sistema binario?

Para entender el sistema de numeración binario primero pensemos en el sistema decimal que usamos a diario. En él tenemos 9 dígitos (0, 1, 2, 3, 4, 5, 6, 7, 8 y 9) y vamos contando en ese orden hasta que los agotamos (o sea, cuando llegamos al nueve y no hay más dígitos). Entonces nos toca combinarlos en números de dos dígitos, así que comenzamos en el siguiente de la lista (1) y volvemos a contar del 0 al 9 (10, 11, 12, 13, 14, 15, 16, 17, 18 y 19).

Cuando terminamos esta secuencia vamos al siguiente dígito de la lista y volvemos a comenzar: 20, 21, 22, 23… y cuando terminamos, vamos al siguiente: 30, 31, 32… Y cuando terminamos con todos, creamos números de 3 dígitos: 100. Y volvemos a comenzar.

En el sistema binario no tenemos 10 dígitos, solo 2: 0 y 1. Pero el proceso es el mismo. Veamos:

Contamos 0 y 1.

Si queremos ir al siguiente número (el que en decimal sería 2), nos toca crear un número de 2 dígitos: 10 El tres es fácil porque podemos cambiar el 0 por 1 y nos queda 11. Para el cuatro nos quedamos sin dígitos, así que creamos un número de tres dígitos: 100. El cinco es fácil, solo sumamos uno: 101 Para el seis sumamos uno. Pero ya sabes que sumar 1 da 10, así que 6 es 110. El siete es sumar uno: 111. Y para el 8 nos quedamos sin dígitos, así que toca crear un número de 4 dígitos: 1000

El sistema binario en las computadoras

En 1854, George Boole publicó “An Investigation of the Laws of Thought” (una investigación sobre las leyes del pensamiento) en la que desarrolla la famosa álgebra booleana que es una teoría fundamental para la informática. El álgebra booleana consiste de tres operaciones lógicas bastante sencillas: AND, OR y NOT.

En este sistema, los resultados de las operaciones solo tienen dos valores posibles: verdadero (true o 1) y falso (false o 0). Por eso, en honor a George Boole, le llamamos booleanos (o boolean) a los tipos de datos verdaderos y falsos.

La operación AND será verdadera (1) solo cuando ambas proposiciones son verdaderas (1 y 1). Por ejemplo, si en EDteam estamos buscando un programador frontend que sepa React y TypeScript es una operación AND. Porque no le bastará saber una de las dos para pasar. Tiene que, sí o sí, saber ambas.

La operación OR, en cambio, solo requiere que una de las dos proposiciones sea verdadera para ser verdadera. Por ejemplo, si estás organizando una fiesta, puedes invitar a amigos del trabajo o a tu familia. No es necesario que sean a la vez del trabajo y de tu familia.

Mientras que la operación NOT implica negar la proposición. Por ejemplo, podrías decir que NO se admiten niños en tu fiesta.

Estas tres operaciones son la base de cómo funciona nuestro pensamiento cuando analizamos una situación y tomamos decisiones. Sin embargo, se convirtió (sin que su autor se lo proponga) en la base para la informática y 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.

¿Qué son los Niveles lógicos?

En los circuitos digitales es muy común referiste a las entradas y salidas que tienen las compuertas AND, compuerta OR y otros muchos circuitos integrados digitales, como si estas fueran altos o bajos. (Niveles lógicos altos o bajos). A la entrada alta se le asocia un “1” y a la entrada baja un “0”. Lo mismo sucede con las salidas.

Niveles lógicos en tecnologías TTL, CMOS y HC

Nivel de tensión	TTL	CMOS	HC
Bajo (0)	0V - 0.8V	0V - 1.5V	0V - 1V
Alto (1)	2V - 5V	3.5V - 5V	3.5V - 5V

www.unicrom.com

En la tabla anterior se muestran niveles de voltaje para diferentes familias lógicas y un rango de valores para el cual se acepta un nivel (sea este “0” o “1”).

En las compuertas TTL un nivel lógico de “1”, será interpretado como tal, mientras el voltaje de la entrada esté entre 2 y 5 Voltios.

En la tecnología CMOS una nivel lógico de “0”, será interpretado como tal, mientras el valor de voltaje de la salida esté entre 0V. y 1.5 V

Voltajes y corrientes de entrada y salida máximas y mínimas, para el diseño con circuitos integrados digitales

- Un voltaje de entrada nivel alto se denomina VIH
- Un voltaje de entrada nivel bajo se denomina VIL
- Un voltaje de salida nivel alto se denomina VOH
- Un voltaje de salida nivel bajo se denomina VOL

Además de los niveles de voltaje, también hay que tomar en cuenta, las corrientes presentes a la entrada y salida de las compuertas digitales.

- La corriente de entrada nivel alto se denomina: IIH
- La corriente de entrada nivel bajo se denomina IIL
- La corriente de salida nivel alto se denomina: IOH
- La corriente de salida nivel bajo se denomina IOL

Estos valores de corriente de salida pueden obtenerse con ayuda de la ley de Ohm. Io = Vo/RL, donde:

- Io: es la corriente de salida
- Vo: es el voltaje de salida
- RL: es el resistor de carga o su equivalente conectado a la salida.

Conclusión

El uso de 0 y 1 en hardware no es solo una cuestión matemática, sino también eléctrica. La evolución desde los sistemas numéricos antiguos hasta el binario ha permitido el desarrollo de circuitos digitales eficientes que utilizan voltajes y corrientes para interpretar estos estados lógicos. Gracias a los avances de Boole y Shannon, el álgebra booleana y los niveles lógicos se convirtieron en los pilares del funcionamiento de los dispositivos modernos. Esta simplicidad permite a las computadoras ejecutar operaciones complejas utilizando únicamente dos estados, demostrando el poder del sistema binario en la tecnología.

Referencias

Equipo ED. (sf). **¿Por qué las computadoras solo entienden 0 y 1? Código binario**. recuperadohttps ://ed.equipo /blog/por-que-las-computado-solo -entienden-0-y -1-do-papelera

Unicrom. (sf). **Niveles lógicos alto, bajo (0 y 1 - Bajo y Alto)**. Recuperadorhttps ://unic.com /n-logicos-Alabama-b-0--1-bajo -alto/

Llamas Cánovas, C. (sf). **Lógica combinacional**Universidad de https ://www2.en.uva.es/~cl/fi/fi-2 .pdf