

ORÍGENES Y FUNCIONAMIENTO BÁSICO DE LOS PRIMEROS ORDENADORES

La revolución en el que está inmersa la sociedad actual por la presencia en todas sus facetas de las tecnologías de la información y la comunicación (TIC) es indiscutible, pero para los no versados en la informática nos surge la pregunta de cómo hemos llegado hasta aquí. Para intentar responder en cierta forma a esa cuestión, vamos a centrarnos en analizar de forma muy somera y básica cómo la confluencia en los avances de distintas disciplinas de conocimiento y tecnologías han dado origen a las computadoras (también llamados computadores u ordenadores) hasta llegar a sus primeras generaciones.

El sistema binario

El sistema binario es el sistema de numeración que permite operar solo con los números 0 y 1, pero con los mismos resultados que el sistema decimal convencional (0,1,2,3,4,5,6,7,8 y 9).

Este sistema binario, es el empleado actualmente para manejar cualquier dispositivo informático actual debido a la sencillez (en relación al decimal) con la que permite la representación, almacenaje y transmisión de información (instrucciones y datos en forma de texto, imágenes, vídeos, audio, etc.)

Los orígenes del sistema binario datan del antiguo Egipto, donde se estableció un sistema para calcular las fracciones de superficies de cultivo. Más tarde en el libro I Ching (1.200 AC), los chinos plasmaron una suerte de sistema binario de más complejo. Por su parte, el indio Pingala, sobre el siglo III AC, elaboró un sistema binario más completo que los dos anteriormente referidos.

Pero es **Leibniz** (1646-1716) quien finalmente basándose en esos antecedentes conforma el sistema binario en su conceptualización actual como sucesivas potencias del número 2 y sus reglas aritméticas.

Handwritten table by Leibniz showing powers of 2 and their corresponding decimal values. The table is written on aged paper and includes the following data:

Power of 2	Decimal Value
2^0	1
2^1	2
2^2	4
2^3	8
2^4	16
2^5	32
2^6	64
2^7	128
2^8	256
2^9	512
2^{10}	1024

Detalle de libro de Leibniz Explicación de la aritmética binaria (1703), en el que se aprecia la correspondencia entre el sistema binario, el decimal y las potencias de 2.

Binario	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Decimal	128	64	32	16	8	4	2	1

Correspondencias de números binarios y decimales en una secuencia de ocho bits (un byte)

Para trasladar el sistema binario al decimal, se suman los valores en los que recaen los 1 del sistema binario. Por ejemplo, si tuviéramos esta secuencia de ocho bits, **00101100**, estas son las operaciones que hemos de realizar para determinar que en decimal es **44**.

Binario	0	0	1	0	1	1	0	0
Potencias de 2	$128*0$	$64*0$	$32*1$	$16*0$	$8*1$	$4*1$	$2*0$	$1*0$
Decimal	0	0	32	0	8	4	0	0

$$0+0+32+0+8+4+0+0=44$$

La aplicación práctica del sistema binario a la computación aprovecha sus semejanzas con la electricidad: si hay paso de corriente se representa como 1 y su ausencia como 0, dándose así los dos posibles valores en un bit.

Trabajar con el sistema binario supone simplificar el diseño y aumentar considerablemente la fiabilidad de las máquinas respecto al sistema decimal.

Los conceptos básicos de almacenamiento, procesamiento y transmisión de la información los bits se han acomodado a los sucesivos dispositivos físicos que han permitido que esas operaciones sean más rápidas y eficientes.

Por último, simplemente reseñar que uno de los retos que afrontan las TIC es terminar de desarrollar el sistema de computación cuántica que vendrá a superar a la binaria al permitir trascender de esa dicotomía de unos y ceros a una más versátil, el qubit (una superposición de los estados 1 y 0, que le faculta para tener un valor de 1, 0 o una combinación de ambos simultáneamente).

Lenguajes de programación

El sistema binario en una computadora viene a tener dos funciones, la representación de datos (ya sean almacenados o transmitidos) y dar instrucciones a la computadora para operar con esos datos.

Estas instrucciones básicas, agrupadas en programas, es lo que puede denominarse lenguaje de programación. Una secuencia ordenada de bits que a través de las puertas lógicas (de las que más adelante se hablará) hace que el ordenador lleve a cabo operaciones como sumar, buscar, comparar datos, reproducir audio, etc.

Que una máquina electrónica pueda programarse para realizar diferentes tareas es lo que realmente le confiere las características para denominarse computadora, tratándose así de un dispositivo de procesamiento de propósito general. A diferencia de otras máquinas digitales que únicamente pueden realizar una sola clase de operaciones o de propósito específico (como pueden ser las calculadoras).

Las instrucciones tienen una parte de bits que indican la acción a realizar (el operador) y con qué datos, ubicados en una zona específica de la memoria, realizarlos (el operando).

El lenguaje más básico es el directamente ejecutado en sistema binario y es conocido como código o lenguaje máquina. El lenguaje máquina es el de más bajo nivel (el que llega al máximo detalle y del que derivan el resto de lenguajes).

Tanto el sistema binario como el código máquina solo utilizan el 1 y el 0, pero no son exactamente lo mismo. El sistema binario representa cualquier tipo de información, mientras que el código máquina, empleando el sistema binario, es el código que entiende directamente el procesador de la computadora (mediante el procesamiento que de los bits hacen las puertas lógicas), dando como resultado operaciones aritméticas y lógicas. Haciendo un símil, podríamos decir que el sistema binario es el alfabeto y el código máquina es la gramática.

Como cada tipo de procesador tiene un diseño de circuitos (puertas lógicas) distinto, el lenguaje máquina es particular para cada modelo. De esta forma, cualquier programa en un lenguaje de alto nivel (que en contraste con los de bajo nivel, son más comprensibles e intuitivos para las personas) puede llegar a ser plasmado en binario, pero solo puede ser ejecutado por el tipo de procesador para el que está diseñado.

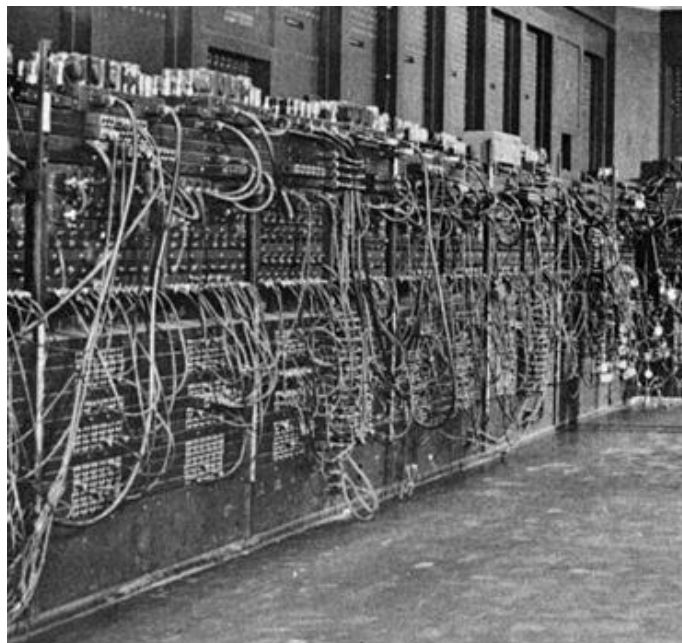
Para solventar esa situación hay unos lenguajes que intermedian entre los de alto nivel (Python, Java, etc.) y el código máquina, siendo estos:

- **Compilador:** lenguajes de alto nivel que traducen los programas escritos en alto nivel (lo que se conoce como código fuente) al lenguaje de bajo nivel ensamblador

- **Intérprete:** también de alto nivel. Realiza la misma tarea que el compilador, pero a diferencia de este otro, que traduce todo el código fuente antes de ejecutarlo, el intérprete va convirtiendo cada instrucción de código fuente para seguidamente ejecutarla y seguir ese proceso sucesivo con el resto de instrucciones del código.

- Ensamblador: este ya es de bajo nivel, pero algo más fácil de manejar por el programador que el código máquina. Traduce el código que le ha sido proporcionado por los compiladores e intérpretes. A pesar de su mayor dificultad, programar en ensamblador permite aprovechar de forma más eficiente las capacidades del procesador y es más rápido que el compilador al generar código directamente ejecutable.

En los inicios de los ordenadores estos se programaban enchufando (señal binaria 1) y desenchufando (señal binaria 0) miles de conectores para que cuando se encendiese la computadora esta supiera cuál eran las instrucciones y los datos sobre los que operar.



Detalle de unos de los módulos del computador ENIAC (1946). Se aprecian los centenares de conexiones manuales (frente) y las válvulas de vacío (arriba).

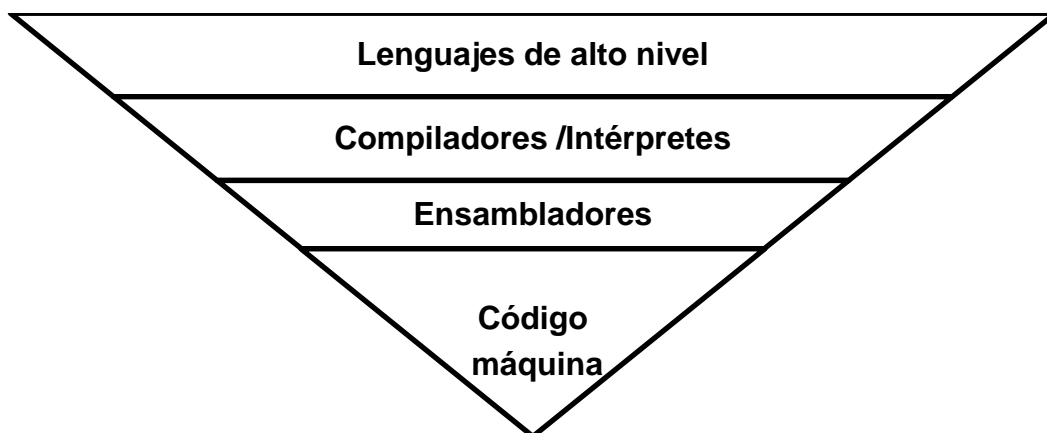


Diagrama de lenguajes de programación desde su mayor semejanza al lenguaje natural (arriba) hasta su mayor proximidad al sistema binario (abajo).

Uno de los lenguajes pioneros más paradigmáticos es FORTRAN, creado en 1954 por IBM para realizar operaciones de forma más intuitiva. Fue novedoso al ser el primer lenguaje considerado de alto nivel y ser exportable a diferentes computadoras no viéndose limitado a la arquitectura del diseño de hardware. A este le siguieron tantos otros hasta llegar a todos los empleados hoy día con diferentes especialidades, como son la creación y administración de redes, sistemas operativos, diseño gráfico, etc.

El álgebra de Boole y las puertas lógicas

Una vez expuesto de forma muy esquemática en qué consiste el sistema binario y los lenguajes de programación, el siguiente aspecto es explicar cómo ese flujo de bits proveniente de las instrucciones se convierte en operaciones a desempeñar por el ordenador.

Para ello nos remontamos a otro gran hito que condujo a la informática actual, el álgebra de Boole o “booleana”. El inglés **George Boole** publicó en 1847, *The Mathematical Analysis of Logic*, en el que planteaba la realización de lógica proposicional por medio de operaciones algebraicas.

La lógica proposicional estudia las diferentes sentencias que componen un enunciado tratando de determinar si son verdaderas o falsas y las relaciones entre ellas, llegando así a la resolución del enunciado.

Esa disyuntiva entre verdadero y falso, que se puede atribuir claramente a los valores 1 y 0 de los bits, y el procesamiento matemático en lugar de semántico son el gran aporte del álgebra booleana a la invención de la informática.

Más adelante, en 1938, **Claude Shannon**, aplicó el álgebra booleana a los circuitos eléctricos diseñando físicamente puertas lógicas conforme a las premisas de Boole.

Es por medio de la combinación de estas puertas lógicas implementadas en el propio hardware con las que el computador procesa toda la información proveniente de los bits (en forma de señal eléctrica) con operaciones aritméticas y lógicas, las cuales se pueden concatenar hasta representar físicamente los algoritmos matemáticos cuya salida es tan diversa como el audio, imágenes, vídeos, etc. Por su gran importancia, seguidamente se exponen todas las puertas lógicas:

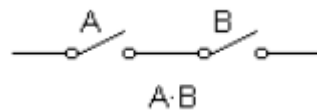
Puerta AND. Operación lógica de conjunción. Multiplica los bits de entrada.



Fórmula algebraica booleana: $S=A * B$

Tabla de verdad de AND		
Entrada A	Entrada B	Salida
0	0	0
0	1	0
1	0	0
1	1	1

Circuito de contactos físicos:



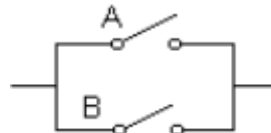
Puerta OR. Realiza la operación lógica de disyunción. Suma los bits de entrada.



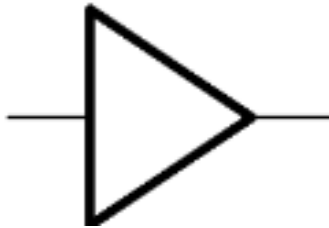
Fórmula algebraica booleana: $S=A + B$

Tabla de verdad de OR		
Entrada A	Entrada B	Salida
0	0	0
0	1	1
1	0	1
1	1	1

Circuito de contactos físicos:



Puerta IF o BUFFER. Operación lógica de igualdad.



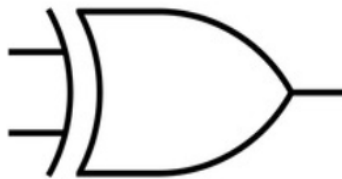
Fórmula algebraica booleana: $S=A$

Tabla de verdad de IF	
Entrada A	Salida
0	0
1	1

Circuito de contactos físicos



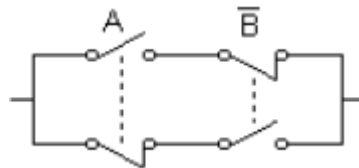
Puerta XOR: Operación lógica de disyunción exclusiva. Mismas entradas y salidas dan salida 0. Entradas distintas son 1.



Fórmulas de álgebra booleana: $S= A \oplus B$ $S=A\bar{B}+\bar{A}B$

Tabla de verdad de XOR		
Entrada A	Entrada B	Salida
0	0	0
0	1	1
1	0	1
1	1	0

Circuito de contactos físicos



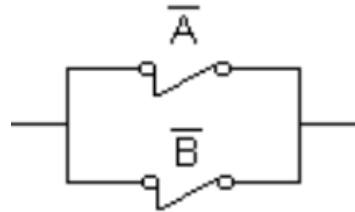
Puerta NAND: Realiza la operación lógica inversa a la conjunción AND. La puerta NAND, junto a la NOR es una puerta universal, lo que significa que tiene la propiedad que conectándola en diferentes configuraciones puede emplearse para realizar el resto de puertas lógicas.



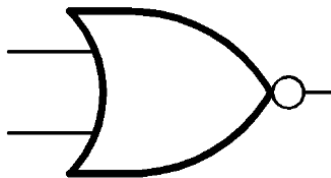
Fórmulas de álgebra booleana: $S = \overline{A \cdot B}$ $S = \overline{A} + \overline{B}$

Tabla de verdad de NAND		
Entrada A	Entrada B	Salida
0	0	1
0	1	1
1	0	1
1	1	0

Circuito de contactos físicos



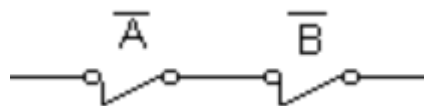
Puerta NOR: Realiza la operación lógica inversa a la disyunción OR. Es la otra puerta lógica universal.



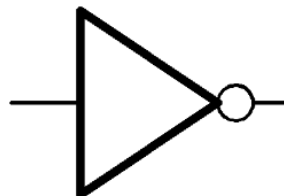
Fórmulas de álgebra booleana: $S = \overline{A + B}$ $S = \overline{A} * \overline{B}$

Tabla de verdad de NOR		
Entrada A	Entrada B	Salida
0	0	1
0	1	0
1	0	0
1	1	0

Circuito de contactos físicos



Puerta NOT: Operación lógica de negación, inversa a IF.

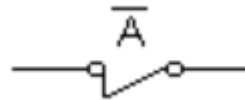


Fórmula algebraica booleana: $S = \overline{A}$

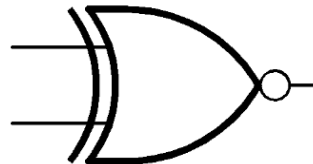
Tabla de verdad de NOT	
Entrada A	Salida
0	1
1	0

0	1
1	0

Circuito de contactos físicos



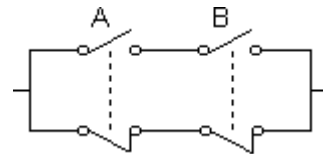
Puerta XNOR: Operación lógica inversa a la disyunción exclusiva XOR.



Fórmulas de álgebra booleana: $S = \overline{A \oplus B}$ $S = A * B + \overline{A} * \overline{B}$

Tabla de verdad de XNOR		
Entrada A	Entrada B	Salida
0	0	1
0	1	0
1	0	0
1	1	1

Circuito de contactos físicos



Almacenamiento de información. La memoria

A la par que se iban asentando esos conocimientos teóricos sobre el sistema binario y las puertas lógicas, se iban inventando y desarrollando los artefactos mecánicos que permitirían su progresiva materialización en dispositivos cada vez más cercanos a las computadoras.

En los otros apartados se ha disertado sobre cómo se procesa la información en un ordenador, pero en este punto vamos a tratar como algunos de los tipos de dispositivos almacenaban la información, que como ya se ha expuesto más arriba consta de operadores y operandos.

1. Las tarjetas perforadas

Las tarjetas perforadas, además de ejercer como memoria persistente (no volátil, aquella que no desaparece cuando el ordenador se apaga) también permitía sistematizar la entrada de información al ordenador y la salida de esta tras finalizar las operaciones.

Las tarjetas perforadas tienen su origen en diferentes ingenios mecánicos del siglo XVII, hasta que el francés Jacquard las perfeccionó a principios del siglo XIX de modo que sus tarjetas permitían que un telar plasmara el patrón contenido en las tarjetas de forma automática.



Detalle de la tarjeta perforada en el telar de Jacquard

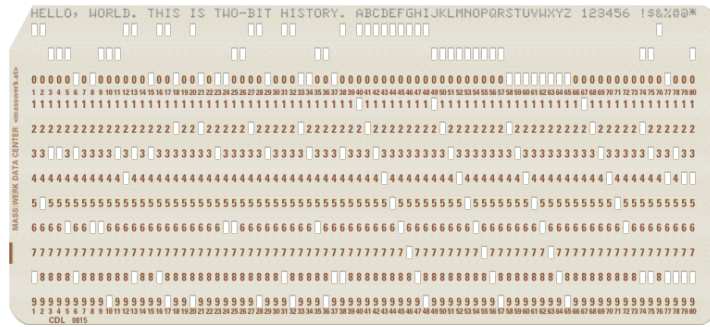
De tal modo que, a semejanza de una computadora donde, la entrada de datos se realizaba con las tarjetas (lo que hoy se realiza con un teclado, clics de ratón, órdenes orales, etc.), el telar procesaba la información de la tarjeta y la salida del proceso era la propia tela con el patrón tejido.

Posteriormente las tarjetas perforadas fueron empleadas por otros los grandes precursores de la computación, **Charles Babbage** quien a mediados del siglo XIX diseñó su “máquina analítica” que es considerada como la primera computadora de propósito general¹, pero que no pudo llegar a confeccionarse en su totalidad en esos momentos.

Más adelante, como ocurre con muchas las teorías y dispositivos vistos aquí, la configuración de las tarjetas perforadas las hizo idóneas para operar con el sistema binario (las perforaciones podían ser 1 y el resto de posiciones indemnes los 0).

Las tarjetas perforadas se introducían en lectores con agujas o contactos que pasaban por los agujeros de las tarjetas y generaban impulsos eléctricos cuando se correspondían con una perforación, convirtiendo así los agujeros en señales eléctricas interpretables por la computadora como ya se ha visto anteriormente. Por otro lado, para la salida de los resultados de las operaciones, la computadora podía perforar nuevas tarjetas con la información procesada.

¹ Aquellas que a diferencia de las de propósito específico (como una calculadora) se pueden programar para realizar distintas operaciones de toda índole.



Tarjeta perforada como las empleadas en las primeras generaciones de computadoras.

Las tarjetas perforadas fueron reemplazadas definitivamente a comienzo de 1980, conforme se terminaron de implantar dispositivos más eficaces para la entrada, el almacenamiento y la salida de datos, como pueden ser los teclados, los discos magnéticos (“blandos” y “duros”) y las pantallas o impresoras.

2. Cintas magnéticas

El empleo de las cintas y discos magnéticos (estos últimos “blandos”, por lo no hay que confundirlos con los de soporte metálico, los “duros”) que aprovechaban las propiedades magnéticas para la grabación y reproducción de sonido data de finales del siglo XIX. No obstante, no fue hasta mediados de los años 50 del siglo XX cuando comenzó a sistematizarse su uso en los ordenadores aprovechando sus propiedades para representar la información en bits, sobre todo la facilidad de escritura, lectura y borrado.

A diferencia de las tarjetas perforadas, su tremenda capacidad de crecimiento de almacenaje en cada vez menos tamaño ha evitado que se extingan, si bien su formato ha evolucionado ostensiblemente.



A la izquierda cintas magnéticas del UNIVAC 1 de mediados de los años 50 (varios Kbytes) utilizada como memoria auxiliar. En el centro un disquete de 5 1/4” de mediados de los 80 con menos de 1 mb, utilizado para la entrada y salida de información. A la derecha una cinta actual en un cartucho de copia de seguridad de 30 TB, usada como almacenamiento terciario -copia de seguridad-.

Hay otros tipos de dispositivos de almacenamiento, como las de estado sólido o las ópticas, pero al ser más recientes no vamos a analizarlos aquí.

Circuitos del procesador

De entre todos los avances dados por la informática la disminución del tamaño de los circuitos que realizan las operaciones del procesador ha supuesto sin duda el de mayor relevancia para situarnos donde estamos hoy. Tanto es así que las generaciones de ordenadores se clasifican precisamente por los saltos dados por los diferentes dispositivos con los que han ido confeccionando los circuitos lógicos.

1. El relé. Primera generación de computadoras.

El relé es un dispositivo electromecánico, inventado en 1835 por Joseph Henry, que aprovecha las propiedades del electromagnetismo, de manera que con el paso de corriente eléctrica abre o cierra interruptores en un circuito.

Como ya se ha explicado más arriba, este funcionamiento permite confeccionar las puertas lógicas y por ende, realizar operaciones lógicas y aritméticas.

El relé fue implementado en los circuitos lógicos en las primeras computadoras electromecánicas, como en la primera máquina programable y completamente automática, la Z3, creada por Konrad Zuse en 1941.

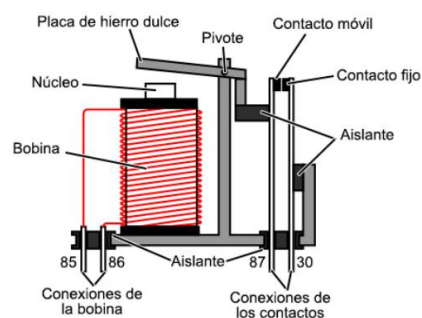


Diagrama de un relé

2. La válvula de vacío. Primera generación de computadoras.

En un proceso constante, de mejora continua, los relés fueron sustituidos por otros dispositivos con menos piezas mecánicas, más rápidos, pequeños y que aprovechaban mejor la electricidad y por tanto generaban menos calor, pero que habían sido concebidos inicialmente en 1904 para amplificar y conmutar señales eléctricas. La válvula de vacío.

La válvula de vacío viene a ser un tubo vidrio sellado al vacío con varios electrodos en su interior que permite controlar el flujo de electrones entre los electrodos mediante una diferencia de potencial.

Fue empleada por primera vez en las primeras computadoras como la ENIAC (1946) o la EDVAC (1949).



Válvula de vacío

3. El transistor. Segunda generación de computadoras.

Considerado uno de los inventos más importantes del siglo XX, fue concebido en 1947 en los laboratorios **BELL**. Es un dispositivo semiconductor que amplifica y conmuta señales eléctricas pero con muchas más eficiencia y fiabilidad que la válvula de vacío. Los transistores sustituyeron de forma definitiva en las computadoras a las válvulas de vacío en la segunda mitad de la década de los 50.



Transistor

4. El circuito integrado (microchip). Tercera generación de computadoras.

Aunque se considera que hay más generaciones, vamos a concluir aquí el estudio con la tercera generación que se inició con la implementación en la informática del circuito integrado, inventado en 1958.

El circuito integrado, como su nombre indica, es un conjunto de circuitos (puertas lógicas) confeccionados por múltiples transistores aglutinados en un solo dispositivo.

Ya en 1965 los procesos y técnicas de fabricación de circuitos integrados estaba tan avanzado, que Gordon Moore formuló la Ley de Moore que establece que el número de transistores de un circuito integrado del mismo tamaño se duplica cada dos años. A día de hoy esa ley sigue vigente y por poner un ejemplo, el procesador Intel Core i9-11900 de la 11ª generación aloja 16.000 millones de transistores con un tamaño de 37.5 mm x 37.5 mm.



Anverso y reverso del procesador Intel Core i9-11900

Hitos teóricos sobre la arquitectura de los computadores

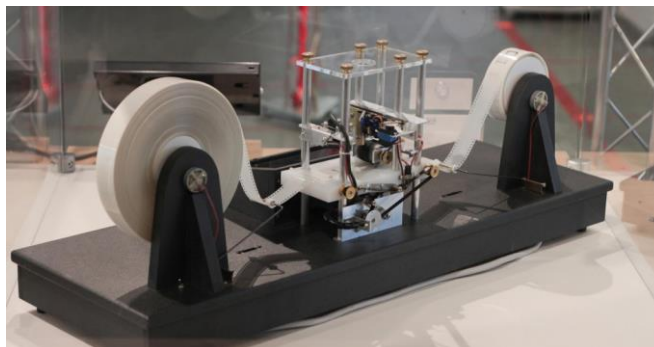
1. Máquina de Turing

Entre los grandes avances de **Alan Turing** a la ciencia de la computación se puede destacar su diseño conceptual en 1936 de lo que sería una máquina con unas características que han venido a definir el concepto moderno de computadora.

La Máquina de Turing, concebida como de propósito general constaría de estos elementos:

- Una cinta que se divide en celdas sucesivas y con una extensión indeterminada en función de la necesidad de datos, en las que se pueden grabar y leer esos datos en forma de símbolos.
- Un cabezal móvil que escribe y lee los símbolos en la cinta.
- Un registro de estado que almacena el estado de la máquina.
- Una tabla finita de instrucciones. En función del estado de la máquina y el símbolo con el que esté operando el cabezal, ordena la escritura, lectura, movimiento por las celdas y el siguiente estado.

Con todo estos elementos la Máquina de Turing podría representar y operar con operaciones lógicas, algoritmos, etc.



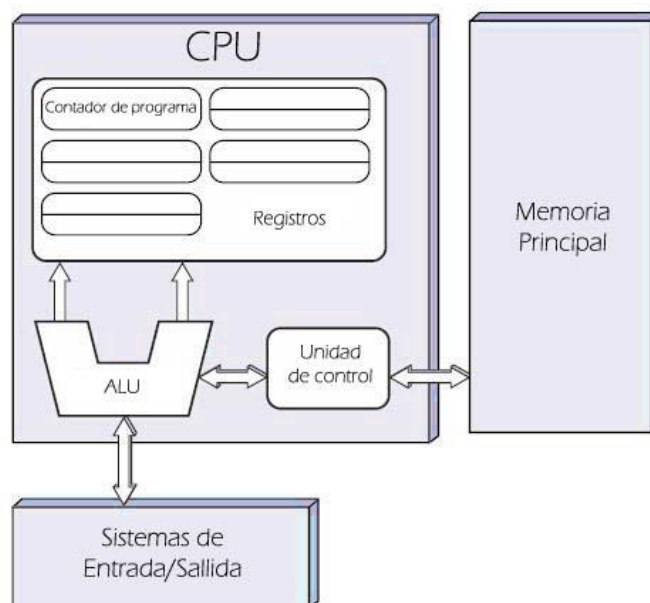
Recreación de la hipotética Máquina de Turing

La influencia de este diseño fue tal que una manera de catalogar los primeros computadores era determinar si podían emularlo como un “sistema Turing completo”.

2. Arquitectura Von Neumann

Finalizaremos este estudio citando obligatoriamente a **Von Neumann** ya que junto a otros compañeros dio el paso definitivo en el diseño físico (arquitectura) de las computadoras digitales de propósito general.

Según esta arquitectura una computadora ha de regirse por este diagrama, que seguidamente pasamos a explicar:



- Consta de unos dispositivos de entrada y salida de información. Siendo la entrada la que aporta datos (operandos) e instrucciones (operadores) y la salida la que muestra el resultado de las operaciones ejecutadas por la computadora.

- Unidad de almacenamiento principal. Es el lugar donde se almacenan las instrucciones y datos con los que trabaja la unidad central de procesamiento (CPU). Actualmente esta memoria se da en formato RAM.

La memoria principal, a diferencia de la secundaria, es volátil, por lo que cuando se apaga el dispositivo todos esos datos se pierden. Por tanto, esta información ha de ser transmitida a la memoria principal a través del sistema de entrada donde se conecta la memoria secundaria (en ordenadores convencionales actuales, esta es el disco duro magnético).

La principal novedad de esta arquitectura fue que precisamente integró en el mismo lugar físico los datos y las instrucciones, dando origen a las computadoras de programa almacenado, permitiendo así hacer mucho más rápidos los procesos y no tener que programar de inicio la computadora cada vez.

No obstante, este diseño supone que la CPU (unidad central de proceso, que hoy día se entiende como el procesador), únicamente pueda llevar a cabo una sola operación simultáneamente, ya sea leer, escribir, buscar instrucciones, etc., lo que se conoce como cuello de botella Von Neumann.

- Unidad de control: La unidad de control es la parte del procesador que se encarga de recoger las instrucciones desde la memoria, decodificarlas y ejecutarlas. También controla el flujo de datos entre los demás componentes del sistema.

- Contador de programa: Contiene la dirección de las siguientes instrucciones a ejecutar. También se ubica en el propio procesador y algunos autores los incluyen en la unidad de control.

- Registros: Es la unidad de almacenamiento más rápida, pero de menor capacidad. Almacena la instrucción que se está ejecutando. Se halla en el procesador y también se encuentra comprendido en la unidad de control.

- La unidad aritmético lógica (ALU): También se ubica en el procesador (CPU). Su función es la de realizar las operaciones matemáticas y lógicas, como sumas, restas, comparaciones, etc.

- Buses: Toda la información transmitida (en forma de flechas en el diagrama), se conduce a través de conexiones llamadas buses.

Como es evidente, este estudio podría ampliarse con mucha más información, detalles, dispositivos, ejemplos, etc., de entre toda la documentación empleada para el presente, se destacan estos enlaces de interés:

- Calculadora mecánica binaria: <https://www.youtube.com/watch?v=NQX6irk11qA>
- Sistema binario y puertas lógicas: <https://www.youtube.com/watch?v=RVGIXfC4Xeg>
- Computadora de 4 bits: <https://www.youtube.com/watch?v=wmpedd2PyjY>.
- Historia de la informática: <https://museo.inf.upv.es/wp-content/uploads/2021/04/Un-viaje-a-la-historia-de-la-informatica.pdf>

Leandro Gobierno López