Arquitectura de Computadores

La arquitectura de computadores es el estudio de la estructura, organización y funcionamiento de los sistemas computacionales. Define cómo interactúan los componentes físicos y lógicos para procesar información. Este documento explora los conceptos básicos, arquitecturas clave, ejemplos modernos y actividades prácticas para comprender su impacto en la tecnología actual.

Objetivos

- Entender los componentes esenciales de una computadora.
- Analizar arquitecturas clásicas y modernas (von Neumann, Harvard, paralelas).

Componentes Básicos de una Computadora

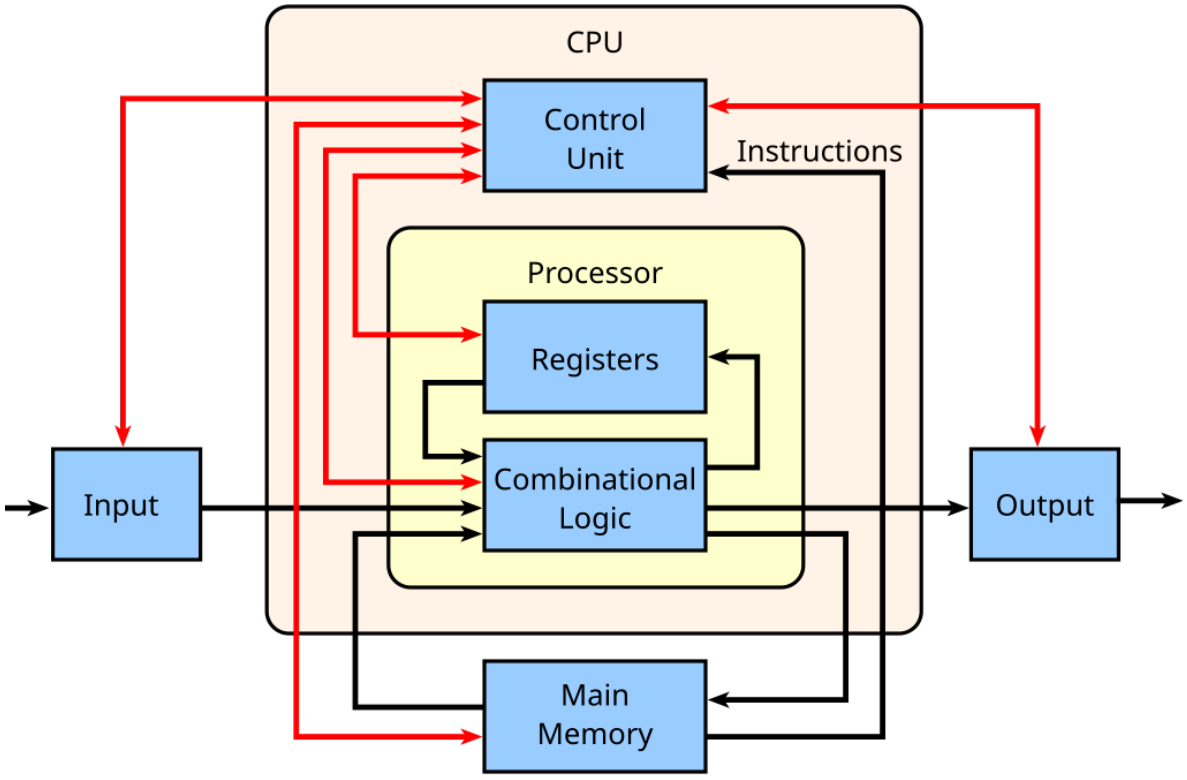
- Unidad Central de Procesamiento (CPU)
    - Función: Ejecuta instrucciones y procesa datos.
    - Subcomponentes:
        - ALU (Unidad Aritmético-Lógica): Realiza operaciones matemáticas.
        - Unidad de Control: Coordina las operaciones.
        - Registros: Almacenamiento temporal de datos e instrucciones.

- Memoria
    - Tipos:
        - RAM: Volátil, almacena datos e instrucciones en ejecución.
        - ROM: No volátil, guarda firmware (ej: BIOS).
        - Caché: Memoria rápida para acceso frecuente.

- Dispositivos de Entrada/Salida (E/S)

- Ejemplos: Teclado, monitor, discos duros, tarjetas de red.

- Buses
  - Función: Canales de comunicación entre componentes.
  - Tipos:
    - Bus de Datos: Transfiere datos.
    - Bus de Direcciones: Indica ubicaciones en memoria.
    - Bus de Control: Gestiona operaciones.



La ALU

La Unidad Aritmética Lógica (ALU, por sus siglas en inglés: Arithmetic Logic Unit) es un componente fundamental del procesador (CPU) responsable de ejecutar operaciones aritméticas y lógicas requeridas por las instrucciones de un programa y se encuentra ubicado en el circuito digital combinacional. Actúa como el "cerebro matemático" del computador, permitiendo desde cálculos básicos (suma, resta) hasta operaciones complejas (comparaciones, desplazamientos de bits). Su diseño y eficiencia impactan directamente en el rendimiento del sistema.

Componentes Principales de la ALU

La ALU está estructurada en tres bloques funcionales:

- **Unidad Aritmética**
  - Funciones:
    - Operaciones básicas: suma, resta.
    - Operaciones avanzadas (según complejidad): multiplicación, división (en ALUs modernas).
    - En ALUs simples, multiplicación/división pueden implementarse mediante software o unidades especializadas (e.g., FPU).
- **Unidad Lógica**
  - Funciones:
    - Operaciones booleanas: AND, OR, NOT, XOR.
    - Desplazamientos de bits (*shift*): lógicos (relleno con ceros) y aritméticos (conserva el signo). (https://www.youtube.com/watch?v=FphdXXiHNPE)
    - Rotaciones de bits.
- **Circuitos de Control**
  - Funciones:
    - Interpreta el *opcode* (código de operación) de la instrucción en ejecución.
    - Genera señales para seleccionar la operación a realizar (e.g., activar sumador en lugar de comparador).
- **Entradas y Salidas**
  - Entradas:

        - Dos operandos (desde registros o memoria).
        - Señales de control (definen la operación a ejecutar).
    - Salidas:
        - Resultado de la operación.
        - Banderas (Flags): Indicadores de estado como:
            - Carry (acarreo): Desbordamiento en operaciones aritméticas.
            - Zero (cero): Resultado igual a cero.
            - Overflow: Desbordamiento en operaciones con signo.
            - Negative: Resultado negativo (bit más significativo).

Operación de la ALU

Flujo de Trabajo

1. **Fetch**: La CPU carga la instrucción desde memoria.
2. **Decode**: La unidad de control identifica el opcode y configura la ALU.
3. **Execute**:
    1. Los operandos se cargan en los registros de entrada de la ALU.
    2. La ALU procesa la operación y genera el resultado + flags.
4. **Writeback**: El resultado se almacena en un registro o memoria.

Ejemplo de Operación

- Instrucción:** `ADD R1, R2, R3` (suma R2 + R3 y guarda en R1).
- La unidad de control envía señal "suma" a la ALU.
- La ALU recibe R2 y R3, realiza la suma, actualiza flags (e.g., carry si hay desbordamiento).
- El resultado se guarda en R1.

Aplicaciones de la ALU

- Procesamiento de datos: Cálculos matemáticos en programas.

- Toma de decisiones: Comparaciones (e.g., `if (a > b)`) mediante resta y flags.
- Manipulación de bits: Encriptación, compresión, gráficos.
- Ejecución de algoritmos: Desde operaciones básicas hasta IA y simulaciones.

Tendencias Modernas

- Pipelining: División de operaciones en etapas para paralelismo.
- Múltiples ALUs: Procesadores superscalares ejecutan varias operaciones simultáneas.
- Integración con FPU/GPUs: Unidades especializadas para operaciones en coma flotante o gráficos.

ALUs en Arquitecturas Actuales

- Procesadores x86 (Intel/AMD): Múltiples ALUs con soporte para SIMD (SSE, AVX).
- ARM: ALUs optimizadas para bajo consumo en dispositivos móviles.
- RISC-V: Diseño modular permitiendo ALUs personalizadas.

Futuro y Desafíos

- ALU Cuánticas: En investigación para realizar operaciones cuánticas (qubits).
- IA Integrada: Aceleradores de operaciones matriciales (e.g., TPUs de Google).
- Nanotecnología: ALUs en escala atómica para mayor eficiencia.

La ALU es un pilar de la computación moderna, permitiendo desde cálculos simples hasta procesamiento avanzado. Su diseño evoluciona con la demanda de mayor rendimiento y eficiencia, integrando innovaciones como paralelismo y especialización. Dominar su funcionamiento es esencial para ingenieros de sistemas, tanto en diseño de hardware como en optimización de software.

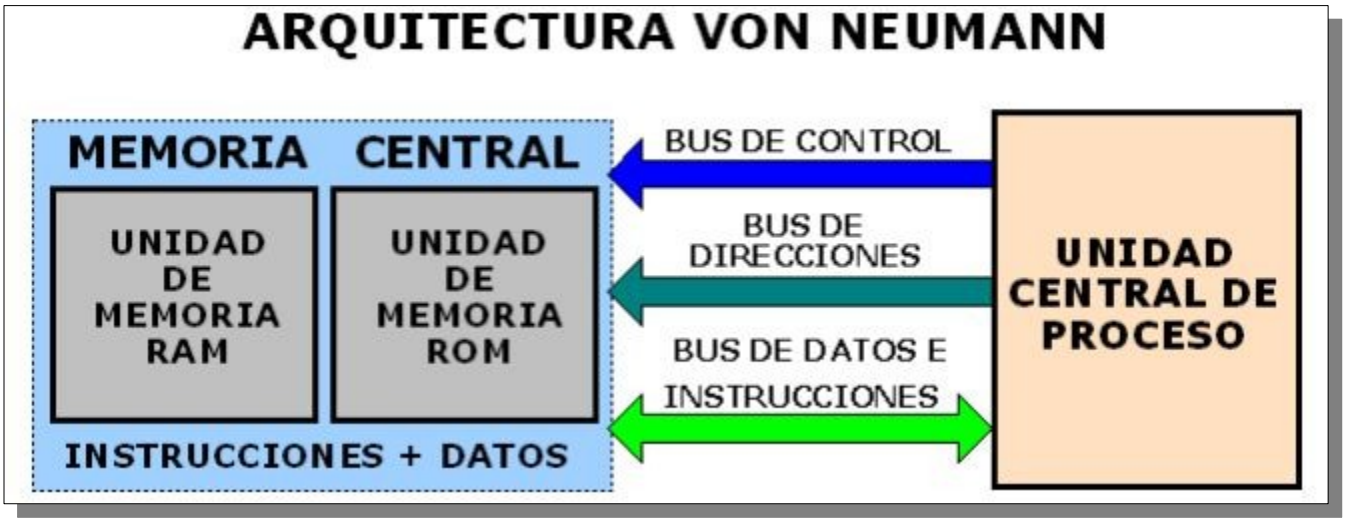
Arquitectura Von Newman y Harvard

Arquitectura de von Neumann

Tradicionalmente los sistemas con microprocesadores se basan en esta arquitectura, en la cual la unidad central de proceso (CPU), está conectada a una memoria principal única (casi siempre sólo RAM) donde se guardan las instrucciones del programa y los datos. A dicha memoria se accede a través de un sistema de buses único (control, direcciones y datos).

En un sistema con arquitectura Von Neumann el tamaño de la unidad de datos o instrucciones está fijado por el ancho del bus que comunica la memoria con la CPU. Así un microprocesador de 8 bits con un bus de 8 bits, tendrá que manejar datos e instrucciones de una o más unidades de 8 bits (bytes) de longitud. Si tiene que acceder a una instrucción o dato de más de un byte de longitud, tendrá que realizar más de un acceso a la memoria.

El tener un único bus hace que el microprocesador sea más lento en su respuesta, ya que no puede buscar en memoria una nueva instrucción mientras no finalicen las transferencias de datos de la instrucción anterior.



Las principales limitaciones que nos encontramos con la arquitectura Von Neumann son:

- La limitación de la longitud de las instrucciones por el bus de datos, que hace que el microprocesador tenga que realizar varios accesos a memoria para buscar instrucciones complejas.

- La limitación de la velocidad de operación a causa del bus único para datos e instrucciones que no deja acceder simultáneamente a unos y otras, lo cual impide superponer ambos tiempos de acceso

La arquitectura Von Neumann describe a la computadora con 4 secciones principales: la unidad lógica y aritmética (ALU), la unidad de control, la memoria, y los dispositivos de entrada y salida (E/S).

Características

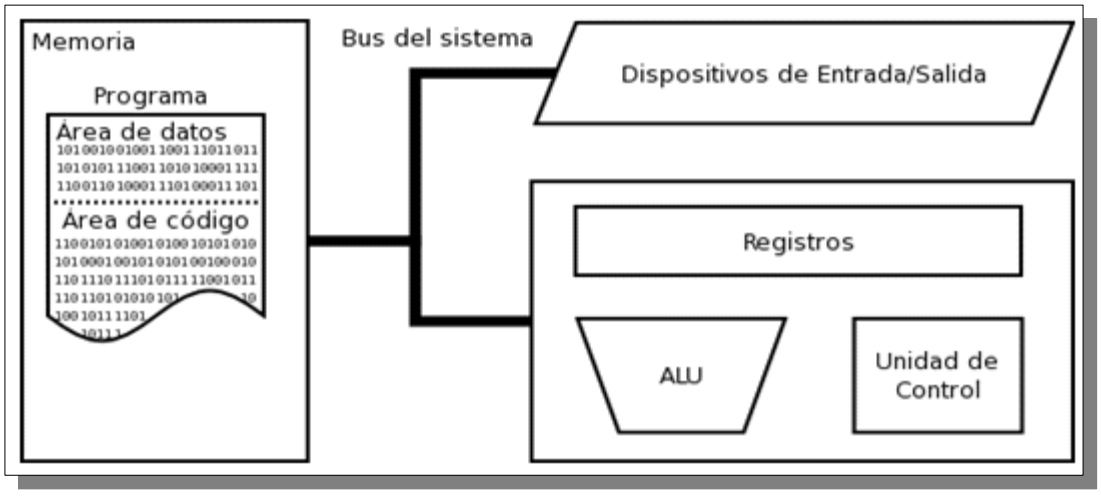
- **Memoria unificada**: Almacena datos y programas en la misma memoria.
- **Buses compartidos**: Un único bus para transferir instrucciones y datos.
- **Secuencialidad**: Las operaciones se ejecutan en serie (puede generar cuellos de botella).

Componentes

1. **Unidad Central de Proceso (CPU)**: Ejecuta instrucciones.
2. **Memoria Principal**: Almacena datos e instrucciones.
3. **Buses de Datos y Control**: Facilitan la comunicación.

Ejemplo: Computadoras personales, servidores (x86, ARM en modo general).

Los ordenadores con arquitectura Von Neumann constan de las siguientes partes:



La arquitectura Von Neumann realiza o emula los siguientes pasos secuencialmente:

1. Obtiene la siguiente instrucción desde la memoria en la dirección indicada por el contador de programa y la guarda en el registro de instrucción.

2. Aumenta el contador de programa en la longitud de la instrucción para apuntar a la siguiente.
3. Descodifica la instrucción mediante la unidad de control. Ésta se encarga de coordinar el resto de componentes del ordenador para realizar una función determinada.
4. Se ejecuta la instrucción. Ésta puede cambiar el valor del contador del programa, permitiendo así operaciones repetitivas.
5. Regresa al paso N° 1.

La mayoría de las computadoras todavía utilizan la arquitectura Von Neumann, propuesta a principios de los años 40 por John Von Neumann.

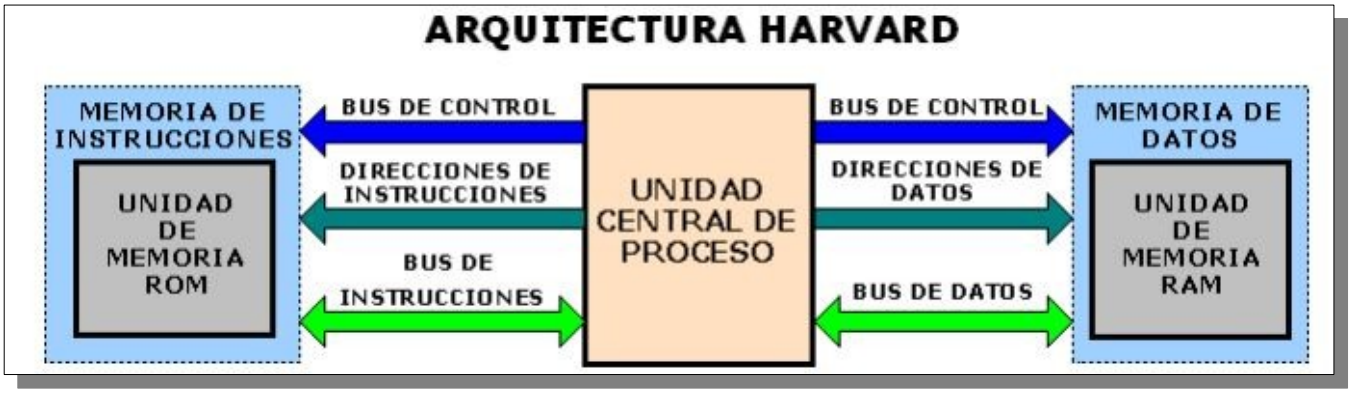
Arquitectura Harvard

Este modelo, que utilizan los Microcontroladores PIC, tiene la unidad central de proceso (CPU) conectada a dos memorias (una con las instrucciones y otra con los datos) por medio de dos buses diferentes.

Una de las memorias contiene solamente las instrucciones del programa (Memoria de Programa o Memoria de Instrucciones), y la otra sólo almacena datos (Memoria de Datos). La memoria de programa o de instrucciones lo diferencia de la arquitectura Vonn Neumann.

Ambos buses son totalmente independientes lo que permite que la CPU pueda acceder de forma independiente y simultánea a la memoria de datos y a la de instrucciones. Como los buses son independientes estos pueden tener distintos contenidos en la misma dirección y también distinta longitud.

También la longitud de los datos y las instrucciones puede ser distinta, lo que optimiza el uso de la memoria en general.



Para un procesador de Set de Instrucciones Reducido, o RISC (Reduced Instrucción Set Computer), el set de instrucciones y el bus de memoria de programa pueden diseñarse de tal manera que todas las instrucciones tengan una sola posición de memoria de programa de longitud.

Además, al ser los buses independientes, la CPU puede acceder a los datos para completar la ejecución de una instrucción, y al mismo tiempo leer la siguiente instrucción a ejecutar.

Ventajas de esta arquitectura:

- El tamaño de las instrucciones no está relacionado con el de los datos, y por lo tanto puede ser optimizado para que cualquier instrucción ocupe una sola posición de memoria de programa, logrando así mayor velocidad y menor longitud de programa.
- El tiempo de acceso a las instrucciones puede superponerse con el de los datos, logrando una mayor velocidad en cada operación.

Recibe el nombre por el ordenador Harvard Mark I, desarrollado en esa universidad de Massachussetts por Howard Aiken

Características

- **Memorias separadas**: Una para datos y otra para instrucciones.
- **Buses independientes**: Permiten acceso simultáneo a datos y código.
- **Paralelismo**: Mayor velocidad en aplicaciones específicas.

Componentes

1. **CPU**: Con buses separados para datos e instrucciones.
2. **Memoria de Instrucciones**: Almacena el programa.
3. **Memoria de Datos**: Almacena variables y resultados.

Ejemplo: Sistemas embebidos (Microcontroladores PIC, Arduino), DSPs. (digital signal processor)

Cuadro Comparativo

Característica	**Arquitectura Von Neumann**	**Arquitectura Harvard**
Memoria	Unificada (datos +	Separada (datos ≠

Característica	Arquitectura Von Neumann	Arquitectura Harvard
	instrucciones)	instrucciones)
Buses	Único bus para datos/instrucciones	Buses independientes
Rendimiento	Cuellos de botella por secuencialidad	Mayor paralelismo
Complejidad/Costo	Menor complejidad y costo	Mayor complejidad y costo
Aplicaciones Típicas	Computadoras generales	Sistemas embebidos, tiempo real
Ejemplos	Intel Core, AMD Ryzen, ARM	Microcontroladores PIC, Arduino

Ejemplos Modernos

Arquitectura	Aplicación	Ejemplo Concreto
von Neumann	Computación general	Laptops (Procesadores Intel Core, RYZEN AMD)
Harvard	Sistemas embebidos	Arduino Uno, ESP32
Paralela (SIMD)	Procesamiento gráfico/IA	NVIDIA GeForce RTX 4090
Multicore (MIMD)	Servidores y supercomputadoras	AMD EPYC, IBM POWER10

Arquitecturas Híbridas y Evoluciones Modernas

Muchos sistemas actuales combinan ambas arquitecturas:

- Arquitectura Harvard Modificada: Usada en procesadores ARM (cachés separados para datos/instrucciones, pero memoria principal unificada).
- GPUs: Utilizan buses paralelos para alta velocidad en procesamiento gráfico.

Herramientas:

- **Profiler de procesadores**: es una herramienta que analiza el uso de la CPU de una aplicación para identificar qué funciones consumen más tiempo y cómo se asignan los recursos. https://es.wikipedia.org/wiki/An%C3%A1lisis_de_rendimiento_de_software
- **Profiler de memoria**: Herramienta que mide el uso de la memoria de una aplicación y detecta fugas de memoria.
- **Profiler de E/S**: Herramienta que mide el tiempo que tarda una aplicación en realizar operaciones de entrada y salida.
- **Profiler de red**: Herramienta que mide el tiempo que tarda una aplicación en enviar y recibir datos a través de una red.
- **Profiler de rendimiento web:** Herramienta que mide el tiempo de carga de una página web y detecta cuellos de botella en el código.

Leer:

https://es.wikipedia.org/wiki/Ley_de_Moore

https://www.unir.net/revista/ingenieria/computacion-paralela/

Referencias

- Patterson, D. A., & Hennessy, J. L. (2017). *Computer Organization and Design
- IEEE Standard for Binary Floating-Point Arithmetic (IEEE 754).
- Tanenbaum, A. S. (2016). *Structured Computer Organization
- Hennessy, J. & Patterson, D. *Computer Architecture: A Quantitative Approach
- Tanenbaum, A. *Structured Computer Organization
- Documentación oficial de NVIDIA CUDA y Arduino.