Actividad 1: Introducción a la Arquitectura de Computadoras

1.1. Definición

Arquitectura de computadoras es la estructura y organización de los componentes de una computadora y cómo interactúan entre sí.

Es una rama de la ingeniería informática que se centra en el diseño y la organización de los componentes de un sistema de computación.

1.2. Aspectos relevantes de una Arquitectura de Computadoras

Rendimiento: Influye directamente en la velocidad y eficiencia con la que una computadora puede procesar datos.

Eficiencia Energética: Afecta el consumo de energía y la disipación de calor de los sistemas computacionales.

Capacidad de Expansión y Actualización: Determina cómo y cuánto puede ampliarse o mejorarse un sistema a lo largo del tiempo.

Compatibilidad: Asegura que el hardware y el software trabajen juntos de manera armoniosa.

1.3 Evolución de las Arquitecturas de Computadoras

Generación	Capacidades típicas	Lenguajes de programación	Tecnología
Primera (1940-1956)	< 0.01 MFLOPS; RAM de kilobytes	Lenguaje máquina, ensamblador	Válvulas de vacío; máquinas enormes
Segunda (1956-1963)	0.01-0.1 MFLOPS; RAM de kilobytes-mb	FORTRAN, COBOL, ALGOL	Transistores para mayor fiabilidad
Tercera (1964-1971)	0.1-1 MFLOPS; RAM de megabytes	BASIC, PL/I, Pascal, C	Circuitos integrados (ICs)

Generación	Capacidades típicas	Lenguajes de programación	Tecnología
Cuarta (1971-1989)	1-10 MFLOPS; RAM de kilobytes-mb	C, C++, Ada, Smalltalk	Microprocesadores, PCs accesibles
Quinta (1990-presente)	100+ MFLOPS a TFLOPS; RAM de gigabytes	Java, Python, C#, PHP, JavaScript, Ruby	Procesadores multinúcleo y redes
Actualidad	Petaflops+; RAM de terabytes	Python, Rust, Go, IA (TensorFlow, PyTorch)	Procesadores especializados en IA

1.4 Tipos de Arquitecturas

Existen conceptos frecuentes al hablar de los tipos de arquitectura de computadoras y su clasificación, como **von Neumann**, **Harvard**, **CISC** y **RISC**. Estos términos se refieren principalmente a aspectos de diseño interno de los microprocesadores, como la organización de la memoria o la complejidad de sus instrucciones. Como programadores, no es necesario profundizar en los detalles técnicos de estas diferencias, pero es útil conocerlos para entender mejor el funcionamiento general de los sistemas en los que desarrollaremos o configuraremos software.

1.4.1. Arquitectura de von Neumann

Memoria unificada para datos e instrucciones.

Usada en la mayoría de procesadores de propósito general, como Intel x86 y AMD Ryzen.

1.4.2. Arquitectura de Harvard

Memorias y buses separados para datos e instrucciones, lo que permite mayor velocidad.

Común en microcontroladores y dispositivos embebidos, como ARM Cortex-M.

1.4.3. CISC (Complex Instruction Set Computer)

Conjuntos de instrucciones complejos y potentes.

Ejemplo: Procesadores **x86**.

1.4.4. RISC (Reduced Instruction Set Computer)

Conjuntos de instrucciones simples y eficientes, ideales para dispositivos móviles.

Ejemplo: Procesadores **ARM**.

1.5. Instrucciones Código Máquina

Comprender la arquitectura de una computadora es esencial para los programadores, ya que las diferencias en los conjuntos de instrucciones de código máquina entre diversas arquitecturas pueden generar incompatibilidades al ejecutar software en diferentes sistemas. Por ejemplo, un programa compilado para una arquitectura particular no funcionará en otra con un conjunto de instrucciones diferente sin modificaciones o emulación, lo que puede afectar el rendimiento y la compatibilidad.

1.5.1 Definición

Son comandos binarios que el CPU ejecuta directamente. Representan operaciones básicas como cálculos o movimientos de datos.

Cada CPU tiene su propio conjunto de instrucciones, llamado **ISA (Instruction Set Architecture)**.

1.5.2 Conjuntos de instrucciones

x86 y x86-64 (x64): Usados en la mayoría de las computadoras de escritorio y laptops.

ARM: Popular en dispositivos móviles y sistemas embebidos, por su eficiencia energética.

1.5.3 Importancia en programación

Al programar o configurar tu estación de trabajo, debes saber si tu sistema operativo es de **32 o 64 bits**.

Un procesador de **32 bits** solo puede usar un sistema operativo y aplicaciones de 32 bits.

Un procesador de **64 bits** puede usar sistemas y aplicaciones tanto de 32 como de 64 bits, pero para aprovechar al máximo su capacidad, se recomienda software de 64 bits.

Si estás desarrollando software, debes considerar la **ISA** del dispositivo objetivo. Por ejemplo, un programa compilado para **x64** no se ejecutará en un dispositivo ARM, como un celular.