RESUMEN FUNDAMENTOS DE COMPUTADORES

Tema 1 - Introducción a los Computadores

Definiciones Preliminares

- Informática: Conjunto de conocimientos científicos y técnicas que permiten el tratamiento automático de la información mediante computadoras.
- Computador: Máquina electrónica que almacena y procesa información para resolver problemas. También conocido como ordenador o computadora.

Componentes de un Computador

Hardware

- Componentes físicos como procesadores, memoria, discos duros, y dispositivos de entrada/salida.

Software

- Programas e instrucciones que permiten la ejecución de tareas en un computador.
- Se encuentra almacenado físicamente en hardware pero es intangible y modificable.

Clases de Computadores

1. Personales: Uso general con equilibrio entre coste y prestaciones.
2. Supercomputadores: Alta capacidad para cálculos científicos e ingeniería.
3. Servidores: Gestionan acceso a redes con alta capacidad y confiabilidad.
4. Empotrados: Integrados en otros sistemas con restricciones de energía y coste.

Niveles de Transformación y Jerarquías

- **Las abstracciones simplifican el diseño de computadores ocultando detalles de niveles inferiores:**
    - Electrones y Transistores: Nivel físico básico.
    - Circuitos Lógicos: Implementación de operaciones digitales.
    - Microarquitectura e ISA (Arquitectura de Conjunto de Instrucciones): Interfaces clave entre hardware y software.
    - Sistemas Operativos, Compiladores y Lenguajes de Programación: Intermediarios que conectan aplicaciones con hardware.
    - Aplicaciones: Programas que resuelven problemas específicos para el usuario.

Modelo Básico del Computador

- **CPU (Unidad Central de Proceso):**
    - Unidad de control.

    - Camino de datos (incluye ALU: Unidad Aritmético-Lógica).
- **Memoria Principal: Almacena temporalmente instrucciones y datos.**
- **Dispositivos de Entrada/Salida: Permiten interacción con el usuario (teclado, pantalla, etc.).**
- **Modelo de Von Neumann: Diseño básico que define la estructura de computadores.**

Codificación de Información

- **Bits y Bytes:**
    - Bit: Unidad elemental (0 o 1).
    - Byte: Conjunto de 8 bits, permite representar hasta 256 valores diferentes.
- **Bases Numéricas: Representación en binario, octal, decimal y hexadecimal.**

Lenguajes de Programación

- Alto Nivel: Cercanos al problema, ofrecen productividad y portabilidad (ej. Python, C, Java).
- **Bajo Nivel:**
    - Ensamblador: Representación textual de instrucciones de máquina.
    - Máquina: Codificación binaria utilizada por el hardware.

Software del Sistema

- Sistemas Operativos: Gestionan recursos de hardware y protegen la ejecución de programas.
- Compiladores e Intérpretes: Traducen programas de lenguajes de alto nivel a instrucciones comprensibles por el hardware.

Rendimiento del Computador

- **Tiempo de CPU:**
    - Depende del número de instrucciones, ciclos por instrucción y frecuencia del procesador.
- **Factores Críticos:**
    - Diseño del algoritmo, eficiencia del compilador, microarquitectura del CPU y tecnología de circuitos.

Resumen detallado por temas

Tema 2: Representación de la información

Introducción

- Los computadores son máquinas diseñadas para procesar información representada en forma binaria.

- La información incluye:
    - **Caracteres alfabéticos**: Letras minúsculas y mayúsculas (A-Z, a-z).
    - **Caracteres numéricos**: Dígitos del 0 al 9.
    - **Caracteres especiales**: Símbolos como *, +, -, (,).
    - **Caracteres de control**: Fin de línea, tabulaciones, entre otros.
- La codificación convierte esta información comprensible para los humanos en secuencias binarias que la máquina puede procesar.
- **Códigos normalizados**: Estándares como ASCII y Unicode garantizan uniformidad.

Representación de enteros

- **Sistemas de numeración**:
    - **Decimal (base 10):** Sistema natural para los humanos.
    - **Binario (base 2):** Sistema fundamental para computadores; usa 0 y 1.
    - **Octal (base 8)** y **Hexadecimal (base 16):** Representaciones compactas para agrupar bits.
- **Métodos de representación interna**:
    - **Signo y magnitud:** Utiliza un bit para el signo; los restantes representan el valor.
    - **Complemento a 2:** Permite representación de enteros con signo y simplifica operaciones aritméticas.
    - **Sesgado:** Agrega un desplazamiento al valor, usado en exponentes de reales.

Representación de reales

- **IEEE 754**:
    - División en signo, exponente y mantisa.
    - Precisión simple (32 bits) y doble (64 bits).
    - Permite representar valores muy grandes, pequeños y no exactos (errores de redondeo).
- **Errores comunes:**
    - Desbordamiento: Cuando el valor excede el rango representable.
    - Subflujo: Para valores menores al mínimo.

Representación de caracteres

- **ASCII:** Define caracteres en 7 bits para textos básicos en inglés.
- **Unicode:** Codificación de hasta 32 bits; incluye emojis, idiomas globales y símbolos matemáticos.

Representación multimedia

- **Imágenes**:
    - Basadas en mallas de pixeles con profundidades de color como RGB (24 bits).
- **Sonido**:
    - Se digitaliza muestreando señales analógicas.
- **Video**:
    - Secuencia de cuadros con codificaciones como MP4 y AVI.

Tema 3: Los lenguajes del computador

ISA (Instruction Set Architecture)

- Define el conjunto de operaciones que el procesador puede realizar.
- Ejemplos:
    - **Carga (load):** Copiar datos de memoria a un registro.
    - **Almacenamiento (store):** Transferir datos de un registro a memoria.
    - **Aritmética:** Suma, resta, etc.
    - **Saltos (jumps):** Alterar el flujo de ejecución del programa.

Arquitectura RISC-V

- **Ventajas**:
    - Diseño abierto y libre de licencias.
    - Fácil aprendizaje y simplicidad en sus instrucciones.
- **Variedades soportadas:** 32, 64 y 128 bits.

Niveles de lenguajes de programación

1. **Lenguaje de alto nivel:** Legible para humanos (C, Python).
2. **Lenguaje ensamblador:** Instrucciones cercanas al hardware, mapeadas al ISA.
3. **Lenguaje de máquina:** Código binario ejecutable por la CPU.

Registros en ensamblador

- **Uso de registros**:
    - Datos almacenados temporalmente dentro de la CPU para acceso rápido.
- **RISC-V:** Tiene 32 registros de 32 bits.
- **Optimización:** Uso eficiente de registros es clave debido a su cantidad limitada.

Tema 4: Estructura del computador

Modelo básico

- **Componentes principales:**
    - CPU: Realiza operaciones y controla el flujo de datos.
    - Memoria: Almacena programas y datos para su procesamiento.
    - Entrada/Salida (E/S): Conexión con dispositivos externos.
- **Interconexiones:** Los componentes se comunican mediante buses.

Procesador (CPU)

- **Unidad aritmético-lógica (ALU):** Ejecución de operaciones matemáticas.
- **Contador de programa (PC):** Indica la próxima instrucción a ejecutar.
- **Registros internos:** Almacenan resultados temporales.

Jerarquía de memoria

1. **Registros:** Ubicados en la CPU; ultrarrápidos y pequeños.
2. **Caché:** Memoria intermedia para reducir latencias.
3. **RAM:** Principal, de mayor capacidad pero más lenta.
4. **Almacenamiento secundario:** HDDs y SSDs.

Sistemas digitales

- Construidos con puertas lógicas (AND, OR, NOT).
- **Circuitos combinacionales:** Dependen solo de las entradas actuales.
- **Circuitos secuenciales:** Guardan información pasada (memoria).

Tema 5: El sistema operativo (SO)

Concepto general

- Software intermedio entre el hardware y las aplicaciones.
- **Roles principales:**
    - Gestionar hardware (CPU, memoria, dispositivos).
    - Ofrecer servicios para que las aplicaciones funcionen.

Interfaces

1. **GUI:** Gráfico, basado en ventanas e iconos.
2. **CLI:** Basado en comandos escritos.

Gestión de procesos

- Los procesos son programas en ejecución.
- **Cambio de contexto:** Permite que el sistema gestione múltiples procesos.

Memoria virtual

- Abstracción que crea la ilusión de tener más memoria que la física.
- Divide el espacio en **páginas**, asignándolas a programas según demanda.

Tema 6: Compilación e interpretación

Compilación

- Traducción completa de código fuente a lenguaje máquina antes de ejecutar.
- Ventajas:
    - Programas rápidos y eficientes.
- Ejemplo: Lenguajes como C, C++.

Interpretación

- Traducción y ejecución línea por línea.
- Ventajas:
    - Flexibilidad para depuración.
- Ejemplo: Python.

Modelos híbridos

- Bytecode interpretado por máquinas virtuales (e.g., JVM para Java).

Uso de bibliotecas

- Funcionalidades adicionales reutilizables.
- Ejemplo en Python: os, math, numpy.

Prácticas detalladas

1. **Linux Shell:**
    - Domina comandos como ls, grep, chmod y el manejo de paquetes.
2. **Control de versiones (Git):**
    - Usa commit, push y crea ramas para colaboración.
3. **Simulaciones:**

- Usa RIPES para entender el procesamiento en caché y arquitectura RISC-V.

4. **Cómputos en binario:**
    - Convierte números y realiza operaciones con complemento a 2.