Arquitectura de Computadoras: Conexión entre Hardware y Software

Ingeniería | Ingeniería de sistemas | para estudiantes universitarios | 4 semanas

Descripción del Curso

Este curso ofrece una comprensión profunda de la arquitectura de computadoras, enfocándose en la conexión esencial entre el hardware y el software. Está diseñado para estudiantes universitarios de ingeniería de sistemas que buscan entender cómo los componentes internos de una computadora funcionan conjuntamente para ejecutar programas y cómo las decisiones de diseño impactan el rendimiento y la eficiencia.

El curso abarca desde la estructura del procesador y la jerarquía de memoria hasta los sistemas de entrada y salida y las técnicas modernas de paralelismo. Se presentan tres perfiles de enfoque: hardware y diseño, software y rendimiento, y sistemas empotrados, permitiendo a los estudiantes adaptar sus aprendizajes a distintos ámbitos profesionales. La metodología combina exposiciones teóricas con análisis de casos prácticos y ejercicios aplicados, promoviendo la reflexión crítica y la aplicación práctica.

Al finalizar, los estudiantes serán capaces de analizar y explicar el funcionamiento interno de un sistema computacional, evaluar cómo las decisiones de diseño afectan el rendimiento, y aplicar técnicas para optimizar tanto el hardware como el software en diferentes contextos tecnológicos.

Objetivos Generales

- Describir y explicar los componentes internos de una computadora y su interacción para la ejecución de instrucciones.
- Analizar la jerarquía de memoria y su influencia en la velocidad y eficiencia del sistema.
- Evaluar los sistemas de entrada y salida y su integración con el procesador y la memoria.
- Aplicar conceptos de paralelismo para mejorar el rendimiento en la ejecución de programas.
- Integrar conocimientos de hardware y software para diseñar soluciones eficientes en diferentes contextos tecnológicos.

Competencias

- Analizar la estructura y funcionamiento de la unidad central de procesamiento (CPU) y sus componentes.
- Evaluar la jerarquía de memoria y su impacto en el rendimiento del sistema.
- Diseñar y optimizar sistemas de entrada y salida para una comunicación eficiente con dispositivos externos.
- Aplicar técnicas de paralelismo para mejorar el rendimiento en sistemas computacionales.

- Integrar conocimientos de hardware y software para desarrollar soluciones eficientes en sistemas embebidos y de propósito general.
- Interpretar y seleccionar estrategias de optimización según perfiles de enfoque: hardware, software y sistemas empotrados.

Requerimientos

- Conocimientos básicos de programación y estructuras de datos.
- Fundamentos de electrónica digital y lógica combinacional.
- Acceso a simuladores o herramientas de arquitectura de computadoras (recomendado).
- Material bibliográfico básico sobre arquitectura de computadoras y sistemas operativos.

Unidades del Curso

Unidad 1: Estructura y Funcionamiento del Procesador

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de describir la arquitectura interna de la CPU identificando sus componentes principales y sus funciones específicas.
- Al finalizar la unidad, el estudiante será capaz de explicar el ciclo de instrucción detallando cada fase y su importancia en la ejecución de programas.
- Al finalizar la unidad, el estudiante será capaz de analizar el manejo de datos dentro del procesador y su flujo entre registros, unidad aritmético-lógica y memoria.
- Al finalizar la unidad, el estudiante será capaz de interpretar instrucciones de máquina y demostrar cómo estas son decodificadas y ejecutadas por el hardware.
- Al finalizar la unidad, el estudiante será capaz de evaluar el impacto de la estructura del procesador en la eficiencia de la ejecución de software mediante ejemplos prácticos.

Contenidos Temáticos

1. Arquitectura Interna de la CPU

- **Descripción general de la CPU:** Introducción a la función principal de la CPU en el sistema computacional y su relación con el hardware y software.
- **Componentes principales de la CPU:**
  - Unidad de Control (UC): funciones y operación.
  - Unidad Aritmético-Lógica (ALU): operaciones aritméticas y lógicas básicas.

    - Registros internos: tipos, propósito y ejemplos (registro acumulador, contador de programa, registro de instrucciones).
    - Buses internos: definición y función dentro de la CPU.
- **Organización interna y jerarquía:** Cómo se interconectan los componentes y la importancia de su diseño.

2. El Ciclo de Instrucción

- **Concepto y fases del ciclo de instrucción:**
    - Búsqueda (Fetch): obtener la instrucción desde la memoria.
    - Decodificación (Decode): interpretar la instrucción.
    - Ejecutar (Execute): realizar la operación indicada.
    - Almacenamiento (Store) / Escritura de resultados (si aplica).
- **Importancia de cada fase:** Cómo cada etapa garantiza la correcta ejecución del programa.
- **Ejemplos de ciclos de instrucción para instrucciones comunes:** carga, almacenamiento, operaciones aritméticas.

3. Manejo y Flujo de Datos en el Procesador

- **Flujo de datos entre registros:** transferencia interna y temporización.
- **Interacción entre la ALU y los registros:** cómo se procesan los datos.
- **Comunicación entre CPU y memoria:** lectura y escritura de datos, uso del bus de datos y direcciones.
- **Ejemplos prácticos de operaciones y movimientos de datos:** diagramas y secuencias de señales.

4. Interpretación y Ejecución de Instrucciones de Máquina

- **Formato de las instrucciones:** campos típicos (opcode, operandos, dirección).
- **Proceso de decodificación:** cómo la unidad de control identifica la acción a realizar.
- **Tipos de instrucciones y su ejecución:** instrucciones aritméticas, lógicas, de control y transferencia.
- **Ejemplos detallados de decodificación y ejecución paso a paso.**

5. Impacto de la Estructura del Procesador en la Eficiencia

- **Factores que afectan la eficiencia:** ancho de bus, cantidad y tipo de registros, complejidad de la ALU.
- **Ejemplos prácticos:** comparación entre procesadores simples y complejos, efectos en el rendimiento.
- **Optimización del ciclo de instrucción:** técnicas como pipelines y paralelismo (introducción básica).
- **Relación entre arquitectura y ejecución de software:** cómo el diseño del hardware influye en la velocidad y eficacia de los programas.

Actividades

Actividad 1: Análisis de la Arquitectura Interna de la CPU

Objetivo: Describir la arquitectura interna de la CPU identificando sus componentes principales y sus funciones específicas.

Descripción:

- El docente presenta un diagrama básico de la CPU.
- Los estudiantes, en parejas, etiquetan cada componente y describen su función.
- Realizan una breve presentación explicando cómo interactúan estos componentes.

Organización: Parejas

Producto esperado: Diagrama completo con etiquetas y descripción escrita.

Duración estimada: 1 hora

Actividad 2: Simulación del Ciclo de Instrucción

Objetivo: Explicar el ciclo de instrucción detallando cada fase y su importancia en la ejecución de programas.

Descripción:

- Se proporciona un conjunto simple de instrucciones de máquina.
- En grupos, los estudiantes simulan manualmente (o con software de simulación) el ciclo de instrucción para cada instrucción.
- Documentan las fases y resultados de cada ciclo.

Organización: Grupos de 3-4 estudiantes

Producto esperado: Informe con la simulación y explicación del ciclo para cada instrucción.

Duración estimada: 2 horas

Actividad 3: Diagrama de Flujo de Datos en la CPU

Objetivo: Analizar el manejo de datos dentro del procesador y su flujo entre registros, ALU y memoria.

Descripción:

- Los estudiantes reciben un caso práctico con una operación aritmética a realizar.
- En forma individual, elaboran un diagrama detallado que muestre el flujo de datos entre registros, ALU y memoria.
- Comparten y discuten sus diagramas en clase.

Organización: Individual, discusión en grupo

Producto esperado: Diagrama de flujo de datos detallado y presentación breve.

Duración estimada: 1.5 horas

Actividad 4: Decodificación y Ejecución de Instrucciones

Objetivo: Interpretar instrucciones de máquina y demostrar cómo estas son decodificadas y ejecutadas por el hardware.

Descripción:

- Se entrega a cada grupo una serie de instrucciones de máquina con su formato.
- Los estudiantes decodifican cada instrucción y describen la secuencia de señales y acciones que la CPU realiza para ejecutarla.
- Se pide un análisis del impacto de la estructura del procesador en el proceso.

Organización: Grupos de 3 estudiantes

Producto esperado: Reporte técnico con decodificación y análisis.

Duración estimada: 2 horas

Evaluación

Evaluación Diagnóstica

Qué se evalúa: Conocimientos previos sobre componentes básicos de la CPU y conceptos generales de arquitectura.

Cómo se evalúa: Cuestionario corto con preguntas de opción múltiple y definición de términos clave.

Instrumento sugerido: Test inicial en línea o en papel con 10 preguntas.

Evaluación Formativa

Qué se evalúa: Comprensión progresiva de la arquitectura, ciclo de instrucción, flujo de datos y decodificación de instrucciones.

Cómo se evalúa: Revisión continua de actividades prácticas, participación en discusiones y entrega de productos parciales.

Instrumento sugerido: Rúbricas para actividades, observación y retroalimentación oral y escrita.

Evaluación Sumativa

Qué se evalúa: Capacidad para describir, explicar, analizar, interpretar y evaluar los conceptos y procesos estudiados en la unidad.

Cómo se evalúa: Examen teórico-práctico que incluya:

- Preguntas de desarrollo sobre la arquitectura interna y ciclo de instrucción.
- Ejercicios de interpretación y decodificación de instrucciones.
- Análisis de casos de flujo de datos y evaluación del impacto estructural en eficiencia.

Instrumento sugerido: Examen escrito con casos prácticos y preguntas abiertas.

Unidad 2: Jerarquía de Memoria y Gestión de Datos

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de describir la estructura y organización de la memoria caché, RAM y almacenamiento secundario, identificando sus características y funciones específicas.

- Al finalizar la unidad, el estudiante será capaz de analizar cómo la jerarquía de memoria afecta la velocidad y eficiencia del sistema, explicando el impacto de cada nivel en el acceso a los datos.
- Al finalizar la unidad, el estudiante será capaz de evaluar estrategias de gestión de datos en la memoria, comparando técnicas para optimizar el uso y la transferencia de información entre niveles de memoria.
- Al finalizar la unidad, el estudiante será capaz de aplicar conceptos de jerarquía de memoria para diseñar soluciones que mejoren el rendimiento del sistema en escenarios prácticos de acceso y almacenamiento de datos.

Contenidos Temáticos

1. Introducción a la Jerarquía de Memoria

- Definición y propósito de la jerarquía de memoria en sistemas computacionales
- Importancia de la organización eficiente de la memoria para el rendimiento del sistema
- Conceptos básicos: tiempo de acceso, capacidad, costo por bit, volatilidad

2. Estructura y Organización de la Memoria Caché

- Definición y función de la memoria caché en la jerarquía de memoria
- Niveles de caché: L1, L2 y L3, sus características y diferencias
- Principios de funcionamiento: localidad temporal y espacial
- Estrategias de mapeo: directo, asociativo y asociativo por conjuntos
- Políticas de reemplazo: LRU, FIFO, aleatorio
- Manejo de coherencia y consistencia en caché

3. Memoria Principal (RAM)

- Función de la memoria RAM dentro del sistema
- Tipos de RAM: SRAM vs DRAM, características principales
- Organización interna y ciclos de acceso
- Relación de la RAM con la CPU y la caché

4. Almacenamiento Secundario

- Tipos de almacenamiento secundario: discos duros, SSD, memorias flash
- Características: capacidad, velocidad, costo y volatilidad
- El rol del almacenamiento secundario en la jerarquía de memoria
- Interfaz y acceso a datos: conceptos básicos de I/O y controladores

5. Impacto de la Jerarquía de Memoria en la Velocidad y Eficiencia del Sistema

- Concepto de tiempo promedio de acceso a memoria
- Impacto del tamaño, velocidad y ubicación de cada nivel de memoria

- Ejemplo práctico: cálculo del tiempo de acceso efectivo
- Importancia del caché para reducir la latencia en el acceso a datos

6. Estrategias de Gestión de Datos en la Jerarquía de Memoria

- Políticas de carga y escritura: write-through vs write-back
- Manejo de fallos de caché (cache misses): tipos y técnicas de mitigación
- Optimización de transferencia de datos entre niveles de memoria
- Uso de buffers y técnicas de prefetching

7. Aplicación de Conceptos de Jerarquía de Memoria para Mejorar el Rendimiento

- Diseño de soluciones basadas en la jerarquía de memoria para casos prácticos
- Análisis de escenarios: optimización del acceso a datos y almacenamiento
- Evaluación de trade-offs entre costo, velocidad y complejidad
- Herramientas y simuladores para modelar jerarquías de memoria

Actividades

Actividad 1: Mapa conceptual de la jerarquía de memoria

Objetivo: Contribuir al objetivo 1, describiendo la estructura y organización de la memoria caché, RAM y almacenamiento secundario.

Descripción:

- Los estudiantes individualmente elaborarán un mapa conceptual que incluya los niveles de memoria, sus características y funciones.
- Se proveerá material base con definiciones y diagramas para apoyar la construcción del mapa.
- Deberán destacar diferencias clave entre caché, RAM y almacenamiento secundario.
- Al finalizar, se realizará una puesta en común para discutir y comparar los mapas conceptuales.

Organización: Individual

Producto esperado: Mapa conceptual digital o en papel que refleje la organización jerárquica y características de cada nivel de memoria.

Duración estimada: 1 hora

Actividad 2: Análisis de casos prácticos de acceso a memoria

Objetivo: Desarrollar la capacidad de analizar el impacto de la jerarquía de memoria en la velocidad y eficiencia del sistema (objetivo 2).

Descripción:

- Se presentarán escenarios con diferentes configuraciones de jerarquía de memoria y patrones de acceso a datos.

- Los estudiantes en parejas calcularán el tiempo promedio de acceso y discutirán cómo las diferencias afectan el rendimiento.
- Se fomentará el análisis crítico de cómo los niveles de memoria contribuyen a la eficiencia del sistema.

Organización: Parejas

Producto esperado: Informe corto con cálculos y conclusiones sobre el impacto de la jerarquía de memoria en cada caso.

Duración estimada: 1.5 horas

Actividad 3: Comparación y debate de estrategias de gestión de datos

Objetivo: Evaluar y comparar estrategias de gestión de datos en la memoria para optimizar uso y transferencia (objetivo 3).

Descripción:

- Se dividirá a los estudiantes en grupos pequeños.
- Cada grupo investigará una estrategia de gestión (write-through, write-back, prefetching, etc.).
- Prepararán una presentación que incluya ventajas, desventajas y ejemplos de aplicación.
- Se realizará un debate en clase donde se confrontarán las estrategias y se discutirán casos de uso.

Organización: Grupos de 4-5 estudiantes

Producto esperado: Presentación grupal y resumen escrito de la estrategia asignada.

Duración estimada: 2 horas (investigación y presentación)

Actividad 4: Diseño de solución para optimización del rendimiento de memoria

Objetivo: Aplicar conceptos de jerarquía de memoria para diseñar soluciones que mejoren el rendimiento (objetivo 4).

Descripción:

- Se proporcionará a los estudiantes un escenario práctico con limitaciones de hardware y necesidades específicas de acceso a datos.
- En grupos, deberán diseñar una estrategia de jerarquía de memoria y gestión de datos que optimice el rendimiento.
- Deberán justificar su diseño con base en conceptos aprendidos, incluyendo selección de niveles de memoria, políticas de gestión y técnicas de optimización.
- Presentarán su propuesta y responderán preguntas del docente y compañeros.

Organización: Grupos de 3-4 estudiantes

Producto esperado: Reporte escrito y presentación oral del diseño propuesto.

Duración estimada: 3 horas (diseño, preparación y presentación)

Evaluación

Evaluación Diagnóstica

Qué se evalúa: Conocimientos previos sobre tipos de memoria, jerarquía y conceptos básicos de acceso a datos.

Cómo se evalúa: Cuestionario de opción múltiple y preguntas abiertas cortas.

Instrumento sugerido: Test digital o en papel al inicio de la unidad con 10-15 preguntas.

Evaluación Formativa

Qué se evalúa: Progreso en la comprensión de la estructura de la memoria, análisis de impacto en rendimiento y comparación de estrategias.

Cómo se evalúa: Revisión y retroalimentación de los mapas conceptuales, informes de análisis de casos y presentaciones grupales.

Instrumento sugerido: Rúbricas específicas para cada actividad que valoren claridad, precisión técnica, análisis crítico y argumentación.

Evaluación Sumativa

Qué se evalúa: Capacidad integral para describir, analizar, evaluar y aplicar la jerarquía de memoria y gestión de datos en un caso práctico.

Cómo se evalúa: Examen escrito teórico-práctico y entrega de un proyecto final donde se diseñe una solución optimizada de jerarquía de memoria.

Instrumento sugerido: Examen con preguntas de desarrollo y resolución de problemas, junto con rúbrica para evaluación del proyecto final.

Unidad 3: Sistemas de Entrada y Salida

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de describir los diferentes tipos de dispositivos de entrada y salida y su función en la comunicación con la computadora, utilizando terminología técnica adecuada.
- Al finalizar la unidad, el estudiante será capaz de analizar el funcionamiento de los controladores y su rol en la gestión de dispositivos externos, comparando diferentes arquitecturas de controladores bajo condiciones específicas.
- Al finalizar la unidad, el estudiante será capaz de explicar los distintos buses de entrada y salida y evaluar su impacto en la transferencia de datos entre el procesador y los dispositivos periféricos mediante ejemplos prácticos.
- Al finalizar la unidad, el estudiante será capaz de aplicar técnicas de transferencia de datos, como el acceso directo a memoria (DMA) y la interrupción, para optimizar la comunicación entre hardware y software en escenarios dados.
- Al finalizar la unidad, el estudiante será capaz de integrar conceptos de sistemas de entrada y salida con la memoria y el procesador para diseñar soluciones eficientes que mejoren el rendimiento general del sistema.

Contenidos Temáticos

1. Introducción a los sistemas de entrada y salida

- Definición y función de los sistemas de entrada y salida (E/S) en la arquitectura de computadoras
- Importancia de la comunicación entre computadora y dispositivos externos
- Terminología técnica básica: periféricos, controladores, buses, transferencia de datos

2. Tipos de dispositivos de entrada y salida

- Dispositivos de entrada: teclado, ratón, escáner, micrófono, sensores
- Dispositivos de salida: monitores, impresoras, altavoces, actuadores
- Dispositivos combinados (E/S): pantallas táctiles, unidades de almacenamiento externas
- Clasificación según velocidad, complejidad y propósito

3. Controladores de dispositivos

- Definición y función de los controladores en la gestión de dispositivos externos
- Arquitectura de controladores: tipos (controladores dedicados, integrados, universales)
- Comunicación entre controlador y CPU: registros, buffers y señales de control
- Comparación de arquitecturas de controladores bajo diferentes condiciones (latencia, ancho de banda, carga)

4. Buses de entrada y salida

- Definición y características de los buses de E/S
- Tipos de buses: paralelos y seriales
- Ejemplos de buses comunes: PCI, USB, SATA, I2C, SPI
- Impacto de los buses en la transferencia de datos: ancho de banda, velocidad, latencia
- Ejemplos prácticos de transferencia de datos utilizando diferentes buses

5. Técnicas de transferencia de datos

- Transferencia programada (polling)
- Interrupciones: concepto, manejo y ventajas
- Acceso directo a memoria (DMA): funcionamiento y beneficios
- Comparación entre técnicas de transferencia y escenarios de aplicación
- Optimización de la comunicación hardware-software mediante técnicas de transferencia

6. Integración de sistemas de E/S con memoria y procesador

- Relación entre sistemas de E/S, memoria y procesador en la arquitectura global
- Modelos de integración: memoria mapeada vs. puertos de E/S
- Diseño de soluciones eficientes para mejorar el rendimiento del sistema
- Estudio de casos: análisis y propuesta de mejoras en sistemas reales

Actividades

Actividad 1: Identificación y clasificación de dispositivos de entrada y salida

Objetivo: Describir los diferentes tipos de dispositivos de entrada y salida y su función en la comunicación con la computadora.

Descripción:

- El docente presenta una lista extensa de dispositivos periféricos.
- Los estudiantes, en parejas, investigan y clasifican cada dispositivo en entrada, salida o combinado.
- Debaten las características técnicas y su función en la comunicación con la computadora.
- Presentan un informe conciso empleando terminología técnica adecuada.

Organización: Parejas

Producto esperado: Informe escrito y presentación breve.

Duración estimada: 1.5 horas

Actividad 2: Análisis comparativo de arquitecturas de controladores

Objetivo: Analizar el funcionamiento de los controladores y comparar diferentes arquitecturas bajo condiciones específicas.

Descripción:

- El docente asigna a grupos pequeños diferentes arquitecturas de controladores para estudiar (por ejemplo, controladores dedicados vs. universales).
- Los estudiantes investigan su funcionamiento, ventajas, desventajas y casos de uso.
- Simulan escenarios con diferentes cargas o tipos de dispositivos para evaluar desempeño.
- Realizan una presentación comparativa destacando aspectos clave y conclusiones.

Organización: Grupos de 3-4 estudiantes

Producto esperado: Presentación comparativa y reporte técnico.

Duración estimada: 3 horas

Actividad 3: Demostración práctica de transferencia de datos via buses

Objetivo: Explicar los distintos buses de E/S y evaluar su impacto en la transferencia de datos mediante ejemplos prácticos.

Descripción:

- Utilizando simuladores o kits de desarrollo, los estudiantes configuran dispositivos para transferir datos vía diferentes buses (por ejemplo, USB y SPI).
- Miden y registran parámetros como velocidad y latencia.
- Analizan los resultados para discutir el impacto de cada bus en el rendimiento.
- Elaboran un informe técnico con conclusiones y recomendaciones.

Organización: Grupos de 2-3 estudiantes

Producto esperado: Informe técnico con datos experimentales.

Duración estimada: 4 horas

Actividad 4: Diseño y simulación de un sistema de E/S integrado con memoria y procesador

Objetivo: Integrar conceptos de sistemas de E/S con memoria y procesador para diseñar soluciones eficientes que mejoren el rendimiento general.

Descripción:

- Los estudiantes, en grupos, diseñan un esquema de arquitectura que integre dispositivos de E/S, controladores, buses, memoria y procesador.
- Aplican técnicas como DMA e interrupciones para optimizar la transferencia de datos.
- Utilizan herramientas de simulación para validar el diseño y evaluar rendimiento.
- Preparan un reporte detallado explicando las decisiones de diseño y resultados obtenidos.

Organización: Grupos de 4 estudiantes

Producto esperado: Diseño arquitectónico, simulación y reporte final.

Duración estimada: 6 horas

Evaluación

Evaluación diagnóstica

Qué se evalúa: Conocimientos previos sobre dispositivos de entrada/salida, controladores y conceptos básicos de buses y transferencia de datos.

Cómo se evalúa: Mediante un cuestionario escrito con preguntas de opción múltiple y de respuesta corta.

Instrumento sugerido: Test diagnóstico digital o en papel con 15 preguntas.

Evaluación formativa

Qué se evalúa: Progreso en la comprensión y aplicación de conceptos relacionados con dispositivos, controladores, buses y técnicas de transferencia.

Cómo se evalúa: Revisión continua de actividades prácticas, participación en debates, informes parciales y retroalimentación en base a ejercicios y simulaciones.

Instrumento sugerido: Rúbricas para informes y presentaciones, listas de cotejo para participación y desempeño en actividades prácticas.

Evaluación sumativa

Qué se evalúa: Dominio integral de los objetivos de la unidad, capacidad para describir, analizar, explicar, aplicar e integrar conceptos de sistemas de entrada y salida.

Cómo se evalúa: Examen teórico-práctico que incluye preguntas conceptuales, análisis de casos, resolución de problemas y diseño de soluciones.

Instrumento sugerido: Examen escrito con preguntas de desarrollo y problemas; proyecto final de diseño y simulación con entrega de informe y presentación oral.

Unidad 4: Paralelismo y Optimización del Rendimiento

Objetivos de Aprendizaje

- Al finalizar la unidad, el estudiante será capaz de identificar y describir las principales técnicas de paralelismo a nivel de hardware y software para mejorar el rendimiento de los sistemas computacionales.
- Al finalizar la unidad, el estudiante será capaz de analizar el impacto de la optimización del código en la eficiencia de ejecución mediante la aplicación de técnicas específicas en ejemplos prácticos.
- Al finalizar la unidad, el estudiante será capaz de evaluar diferentes estrategias de diseño en hardware y software orientadas a la optimización del rendimiento, justificando su aplicación en contextos tecnológicos específicos.
- Al finalizar la unidad, el estudiante será capaz de aplicar conceptos de paralelismo para diseñar soluciones que mejoren la velocidad de procesamiento en programas concurrentes, utilizando herramientas y métodos adecuados.

Contenidos Temáticos

1. Introducción al Paralelismo en Sistemas Computacionales

- Definición y justificación del paralelismo para mejorar el rendimiento.
- Clasificación del paralelismo: a nivel de bit, instrucción, dato y tarea.
- Visión general del paralelismo en hardware y software.

2. Técnicas de Paralelismo a Nivel de Hardware

- Paralelismo a nivel de instrucción (ILP): pipeline, ejecución fuera de orden, predicción de saltos.
- Paralelismo a nivel de dato (DLP): SIMD, vectorización y procesamiento en paralelo de datos.
- Paralelismo a nivel de tarea (TLP): multiprocesadores, multiprocesamiento simétrico (SMP), computación multinúcleo.
- Arquitecturas paralelas: MIMD, SIMD, SPMD y sus características.

3. Técnicas de Paralelismo a Nivel de Software

- Conceptos básicos sobre concurrencia y paralelismo en programación.
- Modelos de programación paralela: hilos, procesos, tareas y eventos.
- Herramientas y librerías para paralelismo: OpenMP, MPI, CUDA.
- Sincronización y comunicación entre hilos/procesos: mutex, semáforos, barreras.

4. Optimización del Código para Mejorar la Eficiencia

- Principios de optimización de código: reducción de instrucciones, minimización de accesos a memoria.
- Optimización a nivel de compilador: inlining, desenrollado de bucles, optimización de acceso a cache.

- Optimización manual: reordenamiento de instrucciones, uso adecuado de estructuras de datos y algoritmos.
- Análisis de impacto de optimizaciones en ejemplos prácticos.

5. Estrategias de Diseño en Hardware y Software para Optimización del Rendimiento

- Diseño de hardware orientado a la eficiencia energética y rendimiento.
- Uso de memorias cache y jerarquías de memoria como estrategia de optimización.
- Diseño de software eficiente para hardware específico: arquitecturas multinúcleo, GPUs.
- Casos de estudio: elección de estrategias según contexto tecnológico (servidores, dispositivos móviles, sistemas embebidos).

6. Aplicación Práctica de Conceptos de Paralelismo para Mejorar la Velocidad de Procesamiento

- Diseño y desarrollo de programas concurrentes utilizando técnicas de paralelismo.
- Uso de herramientas para paralelización y análisis de rendimiento.
- Identificación de cuellos de botella y puntos de mejora en programas paralelos.
- Prácticas de diseño para evitar condiciones de carrera y asegurar la coherencia de datos.

Actividades

Actividad 1: Análisis y Clasificación de Técnicas de Paralelismo

Objetivo: Identificar y describir las principales técnicas de paralelismo a nivel de hardware y software.

Descripción:

- Se proporcionará a los estudiantes un conjunto de casos prácticos y descripciones de sistemas computacionales.
- En equipos, deberán identificar las técnicas de paralelismo empleadas y clasificarlas según nivel (hardware o software).
- Presentarán un informe breve explicando cada técnica identificada y su impacto en el rendimiento del sistema.

Organización: Grupos de 3-4 estudiantes.

Producto esperado: Informe grupal con análisis y clasificación de técnicas.

Duración estimada: 2 horas.

Actividad 2: Optimización de Código con Análisis de Rendimiento

Objetivo: Analizar el impacto de la optimización del código en la eficiencia de ejecución mediante técnicas específicas.

Descripción:

- Se entregará un fragmento de código secuencial con bajo rendimiento.
- Individualmente, los estudiantes aplicarán técnicas de optimización como desenrollado de bucles, mejora en accesos a memoria y reducción de instrucciones.
- Utilizarán herramientas de profiling para medir mejoras en tiempo de ejecución y consumo de recursos.
- Compararán resultados antes y después de la optimización y redactarán un reporte con conclusiones.

Organización: Individual.

Producto esperado: Código optimizado, reporte con análisis y comparación de resultados.

Duración estimada: 3 horas.

Actividad 3: Diseño de Solución Paralela para un Problema Concurrente

Objetivo: Aplicar conceptos de paralelismo para diseñar soluciones que mejoren la velocidad de procesamiento en programas concurrentes.

Descripción:

- Se asignará un problema computacional que requiere procesamiento concurrente (por ejemplo, procesamiento paralelo de grandes volúmenes de datos o simulación).
- En parejas, los estudiantes diseñarán una solución paralela utilizando hilos o tareas y herramientas como OpenMP o MPI.
- Implementarán el programa, evaluarán el rendimiento y documentarán la estrategia de paralelismo usada.
- Discutirán posibles mejoras y riesgos asociados al diseño implementado.

Organización: Parejas.

Producto esperado: Código fuente, informe técnico con análisis de rendimiento y justificación de diseño.

Duración estimada: 4 horas.

Actividad 4: Debate y Evaluación de Estrategias de Diseño para Optimización

Objetivo: Evaluar diferentes estrategias de diseño en hardware y software para optimización del rendimiento, justificando su aplicación en contextos específicos.

Descripción:

- Se asignarán distintos contextos tecnológicos (servidores, dispositivos móviles, sistemas embebidos) a grupos pequeños.
- Cada grupo investigará y propondrá estrategias de diseño óptimas para su contexto.
- Se realizará un debate en clase donde cada grupo defenderá su propuesta y evaluará las de otros grupos.
- Finalmente, cada estudiante redactará una reflexión individual sobre las mejores prácticas y su aplicabilidad.

Organización: Grupos de 3, debate en plenaria y trabajo individual.

Producto esperado: Presentación grupal, participación en debate y reflexión escrita individual.

Duración estimada: 2.5 horas.

Evaluación

Evaluación Diagnóstica

Qué se evalúa: Conocimientos previos sobre paralelismo y optimización, familiaridad con conceptos básicos de hardware y software relacionados.

Cómo se evalúa: Cuestionario de opción múltiple y preguntas cortas al inicio de la unidad.

Instrumento sugerido: Test en línea o en papel con 15 preguntas diseñadas para identificar el nivel inicial de los estudiantes.

Evaluación Formativa

Qué se evalúa: Progreso en la identificación de técnicas de paralelismo, aplicación de optimizaciones y capacidad para diseñar soluciones paralelas.

Cómo se evalúa: Revisión continua de actividades prácticas (informes, códigos, debates), retroalimentación individual y grupal.

Instrumento sugerido: Rúbricas para evaluación de informes y códigos, listas de cotejo para participación en debates y trabajo en equipo.

Evaluación Sumativa

Qué se evalúa: Dominio integral de técnicas de paralelismo, análisis crítico de optimización, justificación de estrategias y aplicación práctica en diseño concurrente.

Cómo se evalúa: Examen escrito y proyecto final de programación paralela con informe técnico.

Instrumento sugerido: Examen con preguntas teórico-prácticas y rúbrica detallada para evaluación del proyecto y reporte final.