Programa de
ARQUITECTURA DE COMPUTADORAS

1. NOMBRE DE LA UNIDAD CURRICULAR

Arquitectura de Computadoras

2. CRÉDITOS

10

3. OBJETIVOS DE LA UNIDAD CURRICULAR

El objetivo general es presentar los fundamentos de la arquitectura de una computadora, tanto a nivel de los conceptos sobre la que se construye como su aplicación en una arquitectura real de ejemplo. Se busca que el estudiante identifique y domine los conceptos y principios fundamentales que permiten que los programas sean ejecutados en máquinas físicas, así como sea capaz de modelar el control de procesos físicos mediante computadoras.

Los objetivos específicos son:

- Explicar el concepto de distancia mínima de un código y de un sistema de codificación binario.
- Describir las características de los sistemas de codificación con capacidad de detección y corrección de errores.
- Describir las características de los sistemas de representación de caracteres, enteros y reales.
- Indicar métodos de codificación y características fundamentales (rango de representación, unicidad del cero, conservación del orden) de los sistemas de representación de enteros y reales. Calcular el error cometido al representar un número en punto fijo o punto flotante.
- Realizar operaciones aritméticas con números representados en los distintos sistemas de codificación de enteros y reales, y conversiones entre diferentes representaciones numéricas.
- Indicar las tablas de verdad de las conectivas booleanas elementales.
- Calcular la expresión booleana de un circuito lógico combinatorio en función de sus entradas.
- Describir el método de Karnaugh y fundamentar por qué logra hallar la expresión booleana mínima en dos niveles. Aplicar el método de Karnaugh para minimizar funciones booleanas.
- Escribir programas en C que manipulen la representación binaria de variables, utilizando operadores binarios bit a bit.

- Describir circuitos combinatorios paradigmáticos tales como decodificador, multiplexor y demultiplexor.
- Definir memoria ROM y describir su organización interna. Explicar la utilidad de las entradas chip select y output enable en memorias ROM.
- Calcular tamaño y organización de una ROM apropiada para guardar la tabla de verdad de una función lógica.
- Escribir programas para inicializar el contenido de una memoria ROM dada la función lógica que resuelve.
- Describir circuitos secuenciales paradigmáticos tales como flip-flops, memoria y, contadores.
- Reconocer la diferencia entre circuitos combinatorios y secuenciales.
- Modelar un problema a través de una máquina de estados.
- Detallar los bloques constructivos de la Arquitectura Von Neumann. Describir los elementos que caracterizan a una implementación particular de esta arquitectura.
- Explicar la diferencia entre Arquitectura y Organización de una computadora. Identificar qué elementos y características de una CPU son definidos a nivel de Arquitectura y cuáles a nivel de Organización.
- Definir RISC y CISC, y explicar las diferencias fundamentales entre ambas filosofías de diseño.
- Dado un conjunto de instrucciones de una arquitectura de computadoras RISC o CISC, diseñar un formato de instrucción apropiado.
- Escribir programas en Assembler simples en un set de instrucciones dado y traducirlo a lenguaje máquina.
- Describir el ciclo de instrucción clásico de cinco etapas (Fetch, Decode, Read, Execute, Write).
- Describir los modos de direccionamiento típicos. Definir registro visible, invisible y parcialmente visible. Identificar ejemplos de cada uno dentro de una arquitectura.
- Identificar y describir los diferentes componentes de hardware que intervienen en la interacción con un dispositivo de entrada/salida (E/S).
- Escribir programas en lenguaje C que interactúen con dispositivos de E/S a través de las sentencias IN y OUT.
- Describir y comparar los distintos mecanismos de identificación de interrupciones.
- Explicar las diferencias entre las técnicas de interacción con la E/S de polling e interrupciones.
- Programar rutinas de atención a la interrupción e integrarlas de manera de resolver problemas que impliquen dispositivos de E/S e interacciones con el mundo físico.
- Describir las características principales del lenguaje ensamblador de una arquitectura tomada como ejemplo, y describir el mecanismo de atención a las interrupciones.
- Compilar programas a lenguaje ensamblador de la arquitectura presentada en el curso, incluyendo el acceso a estructuras de datos, manejo de la entrada/salida y las interrupciones.
- Calcular el consumo de stack de un programa recursivo en lenguaje ensamblador
- Explicar el concepto de jerarquía de memoria, y diferenciar los diferentes niveles de memoria disponibles en una jerarquía estándar de una computadora moderna.

Reconocer las diferencias en precio por bit de almacenamiento y velocidad de acceso, entre los diferentes niveles.

- Definir los principios de localidad espacial y temporal, y explicar cómo se explotan en la jerarquía de memoria.
- Calcular hit, miss, hit rate, miss rate, hit time, miss penalty y tiempo promedio de acceso a memoria en la ejecución de programas simples.
- Describir los diferentes tipos de funciones de correspondencia utilizados en memorias caché.
- Determinar la estructura y el tamaño de los campos que componen a una dirección de memoria para acceder a la caché, dadas las características de la jerarquía de memoria y de la CPU con la que se opera.
- Definir y comparar los algoritmos de escritura write-through y write-back.
- Describir el problema de coherencia de caché y cómo resolverlo
- Describir la técnica de pipelining y explicar por qué el uso de la técnica de pipelining típicamente mejora el rendimiento de los programas.
- Definir los diferentes tipos de obstáculos que ocurren durante la ejecución de un programa en una CPU con pipeline y describir soluciones para cada uno de ellos.
- Describir diferentes tipos de técnicas de predicción de saltos y sus estructuras de hardware asociadas.
- Indicar los requerimientos de hardware mínimos de una CPU superescalar.
- Describir las estructuras de hardware, ventana de instrucciones y buffer de reorden, y explicar su utilidad en el contexto de la ejecución fuera de orden.
- Describir la técnica de renombrado de registros y las estructuras de hardware asociadas a su implementación.
- Definir paralelismo a nivel de máquina y paralelismo a nivel de instrucción y explicar la relación entre ambos conceptos.

4. METODOLOGÍA DE ENSEÑANZA

Comprende clases de teórico, clases de práctico y talleres.

Las clases teóricas son expositivas e implican cuatro horas semanales.

Las clases prácticas semanales son una combinación de clase de consulta y expositivas de dos horas de duración. Se trabaja sobre la base de hojas de ejercicios que se publican semanalmente y se espera que los estudiantes resuelvan fuera de los horarios de clases, se estima una dedicación de dos horas de trabajo no presencial. En clase, a partir de la exposición del docente sobre un ejercicio representativo y su resolución se espera una participación estudiantil activa realizando consultas, aplicando sus conocimientos y reflexionando sobre la resolución del ejercicio. El rol docente es de guía, facilitando y evaluando el aprendizaje.

Las clases de taller semanales de dos horas de duración proponen el trabajo grupal estudiantil resolviendo un ejercicio planteado en el momento. Los docentes orientan a los grupos en la resolución, respondiendo dudas y reflexionando sobre la resolución del ejercicio.

La propuesta tiene como elemento central el trabajo estudiantil activo en la resolución de ejercicios, a partir de la aplicación de conceptos teóricos. Para fortalecer esta aplicación cada semana los docentes brindarán ejercicios con dificultad incremental. Se comenzará trabajando en el encuentro práctico y se culmina en el taller con el planteamiento de un ejercicio integrador y desafiante de los contenidos de cada unidad didáctica preparatorio para la evaluación.

Trabajar colaborativamente en grupos, mostrando habilidades de cooperación, respetando las contribuciones de cada integrante y compartiendo responsabilidades.
Interactuar de manera efectiva con sus pares y docentes, formular y responder preguntas utilizando un lenguaje claro y respetuoso.

Explicar de manera clara y lógica la resolución de un ejercicio, detallando los pasos realizados y razonamientos utilizados.

Gestionar el tiempo, tareas y plazos para realizar un seguimiento adecuado del curso, reflexionando sobre el propio proceso de aprendizaje y estrategias de mejora.

5. TEMARIO

Introducción
- Presentación del Curso
- Historia de las Computadoras

Representación Interna de Datos
- Sistemas de Numeración
- Códigos y Errores
- Representación de distintos tipos de datos

Circuitos Combinatorios
- Álgebra de Boole
- Metodo de Karnaugh
- Circuito Sumador, Multiplexor, Demultiplexor
- Memorias ROM

Circuitos Secuenciales
- Flip-Flops
- Contadores
- Memorias RAM
- Maquinas de Estado

Arquitectura de la Computadora
- Máquina Lógica General
- Arquitectura von Neumann

Organización de la CPU
- Ciclo de Instrucción
- Unidad de Control
- Microprogramación

Entrada/Salida
Interrupciones
Arquitectura de ejemplo
Técnicas de Mejora de Rendimiento
Pipeline
Jerarquía de memoria
Superescalares

6. BIBLIOGRAFÍA

Tema	Básica	Complementaria
Introducción	(1)	(2, 3)
Representación Interna de Datos	(1)	(2, 3)
Circuitos Combinatorios	(1)	(2.4)
Circuitos Secuenciales	(1)	(2,4)
Arquitectura de la Computadora	(1)	(2, 3)
Organización de la CPU	(1)	(2, 3)
Entrada/Salida	(1)	(2, 3)
Interrupciones	(1)	(2, 3)
Arquitectura de ejemplo	(1)	(2)
Técnicas de Mejora de Rendimiento	(1)	(2, 3)

6.1 Básica

1. Notas de Teórico

6.2 Complementaria

2. Stallings, William (2010). Computer Organization and Architecture: Designing for Performance,8/E. Prentice Hall ISBN-10: 0136073735, ISBN-13: 9780136073734
3. Randal Bryant, David O'Hallaron (2015). Computer Systems: A Programmer's Perspective, 3/E. Pearson. ISBN-10: 013409266X ISBN-13: 978-0134092669.
4. Mano, M. Morris (1993). Computer System Architecture, 3/E. Prentice Hall ISBN-10: 0131755633, ISBN-13: 9780131755635.

7. CONOCIMIENTOS PREVIOS EXIGIDOS Y RECOMENDADOS

7.1 Conocimientos Previos Exigidos: matemática discreta, lógica, programación (estructuras de datos y algoritmos, y recursión) y física teórica básica.

7.2 Conocimientos Previos Recomendados: Programación en el lenguaje C.

ANEXO A

A1) INSTITUTO

Instituto de Computación.

A2) CRONOGRAMA TENTATIVO

Semana 1	Introducción y Sistemas de Numeración (2hrs) / Códigos y Errores y Representación Interna de Datos (2 hrs)
Semana 2	Representación Interna de Datos (2 hrs) y Circuitos Combinatorios (2 hrs)
Semana 3	Circuitos Combinatorios (4 hrs)
Semana 4	Circuitos Combinatorios (2 hrs) / Circuitos Secuenciales(2 hrs)
Semana 5	Circuitos Secuenciales (4 hrs)
Semana 6	Arquitectura de la Computadora (2 hrs) / Organización de la CPU (2 hrs)
Semana 7	Repaso y ejercicios de parcial y examen
Semana 8	Arquitectura 8086: generalidades y recursión (4 hrs)
Semana 9	Arquitectura 8086: recursión (2 hrs). Entrada/Salida (2 hrs)
Semana 10	Máquina de estados (2 hrs) / Interrupciones (2 hrs)
Semana 11	Arquitectura 8086: interrupciones (4 hrs).
Semana 12	Repaso y ejercicios de parcial y examen
Semana 13	Técnicas de Mejora de Rendimiento (4 hrs)
Semana 14	Técnicas de Mejora de Rendimiento (2 hrs)
Semana 15	Repaso y ejercicios de parcial y examen

A3) MODALIDAD DEL CURSO Y PROCEDIMIENTO DE EVALUACIÓN

Se compone de dos pruebas parciales individuales durante el semestre que aportan un total de 100 puntos.

De acuerdo a los resultados obtenidos el estudiante puede:

- ganar el cuso. obtiene al menos el 25% de los puntos en cada uno de los parciales.
- aprobar completamente el curso si obtiene al menos 40% de los puntos totales en cada uno de los parciales, y al menos 60% del total de puntos.
- reprobar si no gana el curso.

El examen consiste en una prueba donde se plantean tres partes, dos corresponden a ejercicios de desarrollo y la restante está conformada por preguntas de teórico o teórico-prácticas (pequeños ejercicios de aplicación directa de conceptos del teórico). La condición de aprobación es alcanzar la suficiencia en uno de los ejercicios de desarrollo y en la mitad de las preguntas de teórico o teórico-prácticas.

A4) CALIDAD DE LIBRE

Esta unidad curricular adhiere a la resolución de calidad de libre.

A5) CUPOS DE LA UNIDAD CURRICULAR

No hay cupos.

APROB RES CONSEJO DE FAC. ING.
30/02/24 Exp. 060120 - 000157 - 24.

ANEXO B para la carrera Ingeniería en Computación (plan 97)

B1) ÁREA DE FORMACIÓN

Arquitectura, Sistemas Operativos y Redes de Computadoras

B2) UNIDADES CURRICULARES PREVIAS

Examen aprobado de
Programación 1
y cursos aprobados de:
Programación 2,
Matemática Discreta 1 y
Lógica

Esta UC no acumula créditos con Arquitectura de Computadoras (1443)

FACULTAD DE INGENIERÍA
UDELAR

Formato Aprobado por resolución N°113 del CFI de fecha 04.07.2017

ANEXO B para la carrera Ingeniería Físico-Matemática

B1) ÁREA DE FORMACIÓN

Tecnológica

B2) UNIDADES CURRICULARES PREVIAS

Curso:
- examen aprobado de Programación 1
- curso aprobado de Programación 2
- curso aprobado de Matemática discreta 1
- curso aprobado de Lógica

Examen:
- curso aprobado de Arquitectura de computadoras