Ministerio de Cultura y Educación **(Programa del año 2026)**
Universidad Nacional de San Luis
Facultad de Ciencias Físico Matemáticas y Naturales
Departamento: Informatica
Area: Area II: Sistemas de Computacion

I - Oferta Académica

Materia	Carrera	Plan	Año	Período
ARQUITECTURA DEL PROCESADOR I	ING. EN COMPUT.	28/12	2026	2° cuatrimestre

II - Equipo Docente

Docente	Función	Cargo	Dedicación
GROSSO, ALEJANDRO LEONARDO	Prof. Responsable	P.Asoc Exc	40 Hs
ARROYUELO, MONICA DEL VALLE	Responsable de Práctico	JTP Exc	40 Hs

III - Características del Curso

Credito Horario Semanal				
Teórico/Práctico	**Teóricas**	**Prácticas de Aula**	**Práct. de lab/ camp/ Resid/ PIP, etc.**	**Total**
0 Hs	1 Hs	1 Hs	3 Hs	5 Hs

Tipificación	Periodo
B - Teoria con prácticas de aula y laboratorio	2º Cuatrimestre

Duración			
Desde	**Hasta**	**Cantidad de Semanas**	**Cantidad de Horas**
03/08/2026	13/11/2026	15	75

IV - Fundamentación

Con esta asignatura se inicia el estudio de las arquitecturas de procesadores para los alumnos de tercer año de la Ingeniería en Computación, tomando como punto de partida la diferenciación clara entre los conceptos: arquitectura, implementación y realización de un procesador. El alumno comienza con una arquitectura muy básica, experimenta con una posible implementación de tipo Von Neumann (La llamamos máquinas de primera generación con respecto a las arquitecturas y no a la tecnología utilazada en su realización) y la realización de algunas partes pequeñas de dicha implementación. Nuestro enfoque es proponer una arquitectura de procesador que pueda extenderse imitando la manera en que las arquitecturas han evolucionado históricamente. En consecuencia se le propone al alumno la utilización de una arquitectura muy simple, inspirada en la MU0 (Manchester University versión 0), que únicamente utiliza el modo de direccionamiento absoluto. Sobre esta arquitectura el alumno puede experimentar los problemas que surgen cuando intentan implementar estructuras de datos como los arreglos unidimensionales o pilas. Posteriormente, se le propone al alumno una extensión de la arquitectura que admite el modo absoluto indirecto. Utilizando dicha arquitectura experimentan que pueden resolver, de forma más simple, problemas que utilizan arreglos o pilas y además pueden evitar que las instrucciones del programa se modifiquen durante la ejecución del mismo. La siguiente propuesta de extensión es la incorporación del modo indexado, que no sólo evita que las instrucciones se tengan que modificar durante la ejecución del programa, sino que además permite que el código sea re-entrante (característica muy deseable para implementar sistemas operativos). Por último se propone una arquitectura que incorpora los modos relativo al registro contador de programa (PC) y relativo a una base almacenada en un registro base. En ésta arquitectura final el alumno puede apreciar la especialización de los registros en registros de datos y registros de dirección y el uso diferenciado que se hace de ellos en los algoritmos que implementan. Finalmente el alumno estudia procesadores comerciales que se encuentran en los sistemas de computación más utilizados y experimenta sobre su programación en bajo nivel implementando algunas estructuras de datos (arreglos n-dimensionales y listas vinculadas) y

estructuras de control (secuencia, selección, iteración), subrutinas, pasaje de parámetros a subrutinas, entrada/salida,llamadas al sistema operativo e interrupciones.
Para las extensiones propuestas se han desarrollado los ensambladores y los simuladores para que los alumnos puedan experimentar. Los simuladores utilizan una interfaz similar a la ofrecida por el depurador de código abierto gdb para el sistema operativo Linux.
El alumno estudia y resuelve:
*Problemas de representación de los datos que son manipulados por el procesador.
*Cómo diseñar dispositivos físicos que realicen el almacenamiento, transferencia y manipulación de los datos.
*Comprender cuál es el conjunto de manipulaciones provistas habitualmente por los procesadores.
*Cómo se controla la realización de estas manipulaciones a medida que transcurre el tiempo.
Esto conformará la base previa para el entendimiento de implementaciones más avanzadas que intentan mejorar el rendimiento de los procesadores secuenciales (pipelining, superscalar, multithread, etc.) o que intentan mejorar las limitaciones estructurales de los procesadores de tipo Von Neumann con su esquema de almacenamiento lineal basada en la máquina de Turing incorporando almacenamiento segmentado y/o paginado, memorias asociativas, arquitecturas tipo vector (SIMD), arquitecturas parauna base almacenada en un registro base. En ésta arquitectura final el alumno puede apreciar la especialización de los registros en registros de datos y registros de dirección y el uso diferenciado que se hace de ellos en los algoritmos que implementan. Finalmente el alumno estudia procesadores comerciales que se encuentran en los sistemas de computación más utilizados y experimenta sobre su programación en bajo nivel implementando algunas estructuras de datos (arreglos n-dimensionales y listas vinculadas) y estructuras de control (secuencia, selección, iteración), subrutinas, pasaje de parámetros a subrutinas, entrada/salida,llamadas al sistema operativo e interrupciones. Para las extensiones propuestas se han desarrollado los ensambladores y los simuladores para que los alumnos puedan experimentar. Los simuladores utilizan una interfaz similar a la ofrecida por el depurador de código abierto gdb para el sistema operativo Linux.

El alumno estudia y resuelve:
*Problemas de representación de los datos que son manipulados por el procesador.
*Cómo diseñar dispositivos físicos que realicen el almacenamiento, transferencia y manipulación de los datos.
*Comprender cuál es el conjunto de manipulaciones provistas habitualmente por los procesadores.
*Cómo se controla la realización de estas manipulaciones a medida que transcurre el tiempo. Esto conformará la base previa para el entendimiento de implementaciones más avanzadas que intentan mejorar el rendimiento de los procesadores secuenciales (pipelining, superscalar, multithread, etc.) o que intentan mejorar las limitaciones estructurales de los procesadores de tipo Von Neumann con su esquema de almacenamiento lineal basada en la máquina de Turing incorporando almacenamiento segmentado y/o paginado, memorias asociativas, arquitecturas tipo vector (SIMD), arquitecturas paralelas (MIMD).

V - Objetivos / Resultados de Aprendizaje

*Aprender a representar datos y a manipularlos usando circuitos digitales.
*Comprender cómo están diseñados los procesadores secuenciales y cómo es su ciclo de instrucción.
*Desarrollar una actitud crítica frente al diseño de distintos procesadores.
*Obtener experiencia en programación de bajo nivel. Comprender cómo interactúan los procesadores con su medio externo

VI - Contenidos

Contenidos mínimos
Sistemas Numéricos. Representación de la información: alfanumérico, punto fijo y flotante, representación signo-magnitud, complemento al y a2, etc. CPU: camino de datos, señales de control y registros. Assemblers, registros accesibles al programador, ciclos de búsqueda, ejecución de instrucción, buses internos, mecanismos de acceso a memoria, memorias entrelazadas, formato y conjunto de instrucciones, direccionamiento, subrutinas, interrupciones y excepciones. Dispositivos de: E/S: mapeados en el espacio de memoria y dedicados. Ejemplificación sobre el procesador ARM.

Contenidos
Representación de los datos a nivel máquina. Circuitos digitales: Señales analógicas vs. digitales. Funciones Booleanas como formalización de circuitos digitales. Circuitos combinacionales para la implementación de operaciones lógicas y aritméticas. Circuitos secuenciales, realización de elementos de almacenamiento e interpretación de instrucciones de un procesador. Estructura y funcionamiento de un sistema de computadoras: Diferencia entre arquitectura, implementación y realización.

Unidad de procesamiento central (CPU). Almacenamiento principal. Subsistema de entrada/salida. Interconexión. “Arquitectura Von Neumann” y “Harvard”. Evolución de las arquitecturas de procesadores. Lenguaje de ensamble y lenguaje de máquina. Programación de bajo nivel. Entrada/salida: Organización de la entrada/salida y sus consecuencias sobre la arquitectura del procesador. Protocolo de entrada y salida. Handshake. Entrada/salida programada y solapada. Interrupciones: Multiprogramación, administración dinámica de almacenamiento, protección y seguridad sobre el almacenamiento compartido, cambios de control (interrupciones, llamadas al sistema y despacho de procesos). Entrada/salida con interrupciones. Excepciones.

Unidad 1:
Repaso de las representaciones de números negativos: representación signo y magnitud, sistemas de complementos: complemento a la raíz y complemento a la raíz disminuida, en exceso. Operación de complementación. Números fraccionarios: representación en coma fija y coma flotante: rango de representación y operaciones aritméticas.

Unidad 2:
Repaso de circuitos combinacionales: semisumador, sumador completo, substractor completo, multiplexor y demutiplexor, decodificador. Circuitos secuenciales: biestable, FF-SR, FF-D, registros, contadores ascendentes y descendentes módulo 2^n, registros desplazadores a izquierda y a derecha. Máquina de Moore. Máquina de Mealy. Simlación de la especificación en VHDL de una máquina secuencial de Mealy.

Unidad 3:
Organización de una memoria RAM (Random Access Memory). Construcción de una memoria de cuatro palabras a partir de una memoria de dos palabras. Definición de contenido y continente de la información.

Unidad 4:
Ciclo de instrucción. Organización básica de una máquina de primera generación: unidad de control, unidad aritmética y lógica, registros y buses internos. Instrucciones típicas. Fase de búsqueda de instrucción. Fase de búsqueda de operando en memoria. Fase de procesamiento de los datos. Fase de almacenamiento en memoria del resultado.
Definición de Camino de datos (Datapath). Camino de datos de la máquina de primera generación. Diseño de la unidad de control de una máquina de primera generación. Máquinas de segunda generación. Elementos de una instrucción de la máquina: Operación y frases de dirección. Instrucciones de tres direcciones, dos direcciones, una y cero dirección. Tipos de operandos y tipos de operaciones. Modos de direccionamiento: conceptos generales. Modos de direccionamiento: registro, absoluto, inmediato, registro indirecto, post-incremento, pre-decremento. Modos de direccionamiento de múltiples componentes: indexado, relativos: a una base, a la próxima instrucción y a una página. Base indexado con desplazamiento. Operaciones: Manipulación de datos (manejo de datos, lógicas, aritméticas y relacionales), secuenciamiento, entrada-salida, supervisión.

Unidad 5:
Lenguaje de ensamble: conceptos generales, sintaxis y pseudo operaciones. Programa ensamblador de dos pasadas. Operaciones en tiempo de ensamble, de carga y de ejecución. Implementación en lenguaje de ensamble de las estructuras de control: if-then, if-then-else, case, while y repeat. Acceso en lenguaje de ensamble a arreglos multidimensionales. Máquinas de tercera generación. Máquinas de registros generales y Load Store. El procesador ARM-cortexM4: tipos de datos, conjunto de instrucciones, modos de direccionamiento, formato de instrucción. Llamadas a subrutina y retornos de subrutina. Uso de la pila. Pasaje de parámetros a las subrutinas: por valor, por dirección, por valor-retorno, por retorno. Formas de pasaje de parámetros: en registros, en la pila. Progamación en el microcontrolador LPC4337 multinúcleo asimétrico, compuesto por un núcleo ARM cortexM4 y el otro un cortexM0, utilizado por la plataforma abierta EDU-CIAA del proyecto Computadora Industrial Argentina Abierta desarrollada por un grupo de universidades nacionales entre otros.

Unidad 6:
Organización de un sistema de computadora: unidad central de procesamiento (CPU), memoria, entrada/salida (I/O) y sistema de interconexión (buses). Módulos de entrada/salida: interfaz con la CPU e interfaz con el dispositivo. Organización de la entrada/salida: dedicada e inmersa en el espacio de direcciones de memoria (memory mapped I/O). Protocolos de entrada/salida. Entrada/Salida programada. E/S solapada. Ejemplos de la entrada/salida. Ejemplos de entrada/salida programada en la arquitectura M0 y en el microcontralodor LPC4337 en este último se muestra la generación de un bi-tono DTMF realizando E/S programada del DAC (Digital-Analogic Converter).

Unidad 7:
Interrupciones. Acceso directo a memoria. Concepto de multiprogramación. Problemas presentados por la multiprogramación. Cambios de control necesarios para soportar multiprogramación: Interrupciones, despacho, llamadas al sistema. Condiciones asíncronas y síncronas. Atención de una interrupción asíncrona. Finalización de una interrupción síncrona. Manejo típico de una interrupción. Permiso para interrumpir: prioridades y deshabilitación de interrupciones. Acceso directo a memoria (DMA). Robo de ciclo. Muestra en microcontralador LPC4337 de E/S con interrupciones y DMA en la generación de un bi-tono DTMF utilizando el DAC.

VII - Plan de Trabajos Prácticos

Metodología de enseñanza:

Consiste en la resolución de ejercicios de representación e interpretación de números enteros y números racionales en diferentes sistemas. Adicionalmente los alumnos tienen acceso a apuntes complementarios a la bibliografía sugerida como también a las transparencias utilizadas en las clases teóricas. Sobre las unidades correspondientes a circuitos digitales, los alumnos resuelven ejercicios utilizando los conceptos introducidos en las clases teóricas y adicionalmente participan de la construcción de algunos ejercicios utilizando circuitos integrados de baja escala de integración (la cátedera ha desarrollado tableros que facilitan la construcción), finalmente se les propone describir una máquina secuencial en el lenguaje VHDL y sintetizarla sobre un FPGA para comprobar que la metodología de construcción de circuitos secuenciales es efectiva para la realización de sistemas digitales. En lo relacionado a las arquitecturas de computadoras, los alumnos resuelven ejercicios de programación en lenguaje ensamblador en grado creciente de complejidad que le permiten apreciar la utilidad de los distintos modos de direccionamientos normalmente disponibles en distintas arquitecturas. La cátedra ha desarrollado ensambladores y depuradores para que los alumnos prueben las soluciones de los ejercicios de práctico, tales herramientas implementan una arquitectura que imita la evolución histórica de las arquitecturas de computadoras. Los temas relacionados a entrada/salida e interrupciones se desarrollan para la arquitectura RISC de 32-bits de ARM y los alumnos pueden experimentar sobre placas EDU-CIAA (Computadora Industrial Abierta Argentina Educativa).
https://www.proyecto-ciaa.com.ar/devwiki/doku.php%3Fid=desarrollo:edu-ciaa:edu-ciaa-nxp.html

Práctico N° 1:
Representación y aritmética de números en punto fijo y flotante, rangos de representación.
Con esta actividad evaluamos la capacidad del alumno de identificar, formular y resolver problemas de informática, el uso de conocimientos adquiridos en la disciplina matemática aplicados a la informática y de aprender nociones para el aprendizaje continuo.

Práctico N° 2:
Circuitos digitales
Circuitos sumadores, restadores, multiplexores y demutiplexores, decodificadores y codificadores. Biestables, flip-flop S-R, Cerrojos (Latches), flip-flop D. Experimentación con algunos circuitos provistos por la cátedra que han sido realizados utilizando circuitos integrados programables. Registros. Contadores. Desplazadores. Decodificación de direcciones de memoria. Diseño de máquinas de estado finitas.

Práctico N° 3:
Programación en lenguaje de ensamble de una arquitectura de primera generación (inicial). Limitaciones de la arquitectura que sólo posee direccionamiento absoluto. Extensión de la máquina de primera generación a una de segunda generación. Resolución de problemas de programación sobre esta nueva arquitectura. Utilización de constantes inmediatas. Utilización en forma directa e indirecta de los modos registro, absoluto, indexado, relativos (a una base fija y a la próxima instrucción), base-indexado con o sin desplazamiento y con o sin escala y/o factor.

Práctico N° 4:
Programación en lenguaje de ensamble (modos de direccionamiento).
Realizar programas en los que se usen los distintos modos de direccionamiento: inmediato, registro, registro con escala, registro indirecto y indexado, base-indexado, relativos: a una base, a la próxima instrucción. Implementación de las

estructuras de control if-then, if-then-else, case, while y repeat. Implementación de arreglos multidimensionales. Analizar los listados generados por el programa ensamblador. Traducción de diferentes tipos de instrucciones. Calcular los desplazamientos que conforman algunas instrucciones y que son expresados en complemento a dos.

Práctico N° 5:
Programación en lenguaje de ensamble (subrutinas y pasaje de parámetros)
Instrucciones de llamada a subrutina y de retorno de subrutina. Manejo de la pila. Subrutinas anidadas. Implementación de las distintas formas de pasajes de parámetros: en registro, en la pila. Pasaje por valor, por dirección, por valor retorno y por retorno.

Práctico N° 6:
Entrada/salida e interrupciones
Protocolo entre los dispositivos para hacer una entrada/salida programada. Programas que realicen entrada/salida programada utilizando los protocolos sobre la M0-extendida. Entada/salida en forma programada mediante los protocolos para la utilización del conversor D/A del microcontrolador LPC4337 de la EDU-CIAA. Integración de las rutinas de E/S anteriormente para la reproducción de los bitonos “DTMF” (Dual-tone multi-frequency) de un teléfono digital. Idem al párrafo anterior pero utilizando interrpuciones para realizar la E/S. Idem a lo anterior pero utilizando DMA.
A continuación, se describe cómo se abordan y cómo se evalúan los ejes transversales trabajados en la asignatura:

Eje:
Identificación, formulación y resolución de problemas de informática

Cómo se aborda:
Se aborda a partir de la unidad 1 mediante el desarrollo de trabajos prácticos, guiados por clases teóricas, diapositivas de clase, apuntes de la materia, prácticos de aula/laboratorio y consultas grupales e individuales.

Cómo se evalúa:
Mediante un seguimiento continuo de parte de los docentes. Control de ejercicios en el pizarrón. Además, mediante una evaluación parcial en las fechas predeterminadas en el cronograma de la materia, cada evaluación posee una doble recuperación.

Eje:
Capacidad del alumno de utilizar técnicas y herramientas de aplicación en la informática

Cómo se aborda:
El estudiante debe trabajar en la interpretación y representación de la información utilizando un ambiente en el lenguaje J para verificar la solución de los ejercicios propuestos. Debe utilizar ejemplos de simulaciones de circuitos digitales básicos descriptos en VHDL para comprender el funcionamento en el tiempo de los mismos. El estudiante debe utilizar un simulador y un depuraador con una interfaz y algunos comandos similares al depurador gdb de Linux, también debe desarrollador programas en lenguaje ensamblador y luego producir el lenguaje de máquina utilizando un ensamblador desarrollado en la materia para el sistema operativo Linux. En la parte de entada-salida e interrupciones de utilizar el ambiente provisto por la plataforma abierta para la EDU-CIAA (Computadora insdustrial Abierta Argentina para entornos educativos). En dicha plataforma compilan tanto código en lenguaje ensamblador del ARM, como código C.

Cómo se evalúa:
Los docentes resuelven las consultas grupales y en la pizzara y observan las soluciones de los estudiantes de los ejercicios del práctico de laboratorio.

Eje:
Utilización de técnicas y herramientas de aplicación en la informática

Cómo se aborda:
El estudiante debe programar en lenguaje ensamblador y utilizar las técnicas introducidas en la materia para la realización de las distintas estructuras de contol, para invocar y pasar parámetros a subrutinas, utilizar protocolos para realizar entrada-salida.

Cómo se evalúa:
Los docentes observan la soluciones de los estudiantes durante los prácticos de laboratorio y se exige la entrega de algunos de los ejercicios del práctico de laboratorio.

Eje:
Fundamentos para evaluar y actuar en relación con el impacto social de su actividad en el contexto global y local

Cómo se aborda:
Desde el inicio se fomentará una actitud crítica que les permita evaluar el impacto que tendrá el sistema a desarrollar, tanto en el aumento de la productividad del trabajo con la incorporación de nuevas tecnologías y su impacto social en mundo del trabajo.

Como se evalúa:
Mediante un análisis sobre el impacto social y laboral que tendrá actividad.

Eje:
Fundamentos para el aprendizaje continuo

Cómo se aborda:
Todas las unidades tienen actividades prácticas para que los estudiantes respondan participando de las clases teóricas y clases prácticas. En cada práctico se hace un seguimiento de los ejercicios realizados por el estudiante.

Cómo se evalúa:
Cada consulta tiene una corrección por parte de los docentes .

VIII - Regimen de Aprobación

Regularización
Para regularizar la materia el alumno deberá cumplir con los siguientes requisitos: Asistir al 80% de las clases teóricas y prácticas. Aprobar dos exámenes parciales. Cada examen parcial tendrá dos recuperaciones.
Examen Final
Los alumnos regulares deberán rendir un examen final (que podrá ser oral o escrito) que consistirá en preguntas sobre los temas desarrollados durante el dictado de la materia.
Alumnos libres
Los alumnos que desean rendir libre la materia se deberán poner en contacto con la cátedra a los efectos de realizar un práctico, el cual contendrá ejercicios similares a los desarrollados en los prácticos durante el dictado de la materia. Aprobando este trabajo práctico el alumno deberá rendir un examen oral con iguales características que el de los alumnos regulares

IX - Bibliografía Básica

[1] WILLIAM STALLINGS. Computer Organization and Architecture: Designing for Performance. ED. PEARSON PRENTICE HALL [2010].
[2] JHON F. WAKERLEY. Microcomputer Architecture and Programming. Ed. JOHN WILEY AND SONS [1981].
[3] NIKLAUS WIRTH. Digital Circuit Design. An Introductory Textbook. Springer [1995].
[4] WILLIAM STALLINGS. Organización y arquitectura de computadores. ED. PEARSON PRENTICE HALL [2005].
[5] HOHL,W. and HINDS, C. ARM Assembly Language 2nd. edition. ED. CRC Press [2015].
[6] HENNESSY, J. L. & PATTERSON, D.. Computer Organization and Design.THE HARDWARE / SOFTWARE INTERFACE. RISC-V Edition. ED. MORGAN AND KAUFMANN [2018].
[7] JOHN L. HENNESSY & DAVID PATTERSON. Computer Architecture: A Quantitative Approach. 2nd Edition. ED. MORGAN AND KAUFMANN [1990].
[8] MEINADIER, J. P. Estructura y Funcionamiento de los Computadores Digitales. Editorial AC, Madrid [1980].
[9] GERRIT A. BLAAUW-FREDERICK P. BROOKS, Jr. Computer Architecture. Concepts and Evolution. ED. ADDISON-WESLEY. [1997].

X - Bibliografia Complementaria

[1] BEHROOZ PARHAMI Computer Arithmetic. Oxford University Press; 2da edición [2010].
[2] PETER J. ASHENDEN. The Designer's Guide to VHDL (Second Edition). ED. Morgan Kaufmann Publishers [2002].

XI - Resumen de Objetivos

*Aprender a representar datos y manipularlos usando circuitos digitales.
*Comprender cómo están diseñados los procesadores secuenciales y cómo es su ciclo de instrucción.
*Desarrollar una actitud crítica frente al diseño de distintos procesadores.
*Obtener experiencia en programación de bajo nivel.Comprender cómo interactúan los procesadores con su medio externo.

XII - Resumen del Programa

Representación y aritmética de números enteros y fraccionarios. Circuitos Digitales. Circuitos combinacionales básicos. Circuitos secueciales, Máquinas de estados finitas. Organización Básica de una Computadora. Unidad central de procesamiento (CPU). Memoria. Entrada / Salida. Organización interna de una CPU. Ciclo de instrucción. Conjunto de instrucciones de un procesador. Tipos de arquitecturas secuenciales. Lenguaje assembly, assembler y lenguaje de programación. Entrada / Salida. Interfaz con la CPU. Interfaz con los dispositivos. Interrupciones. Excepciones. Llamadas al sistema. Acceso directo a memoria.

XIII - Imprevistos

Comunicarse con la cátedra.
Correo electrónico: agrosso@email.unsl.edu.ar
Oficina 25. Primer Piso. Bloque II.
Sistemas de computación.
Departamento de Informática.
Facultad de Cs. Físico, Matemáticas y Naturales.
Universidad Nacional de San Luis.
Ejercito de los Andes 950. CP 5700.

XIV - Otros