
DCC

DEPARTAMENTO DE CIENCIA
DE LA COMPUTACIÓN

# IIC2343 – Arquitectura de Computadores

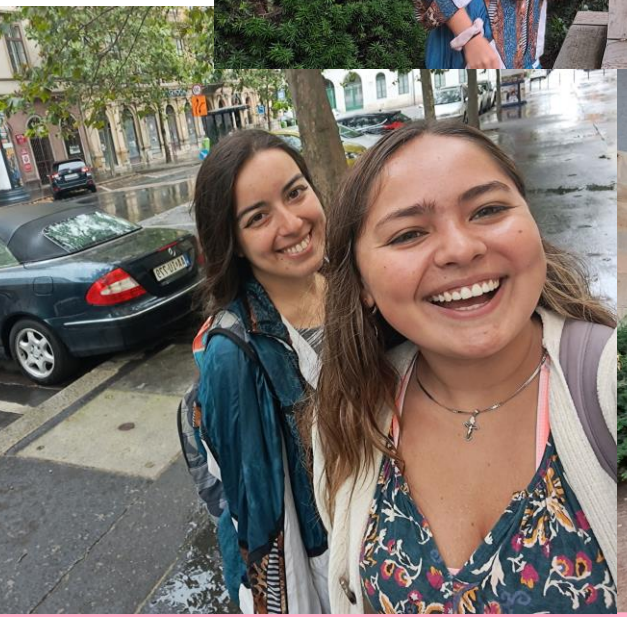
## Clase 0 – Introducción

Susana Figueroa

# Sobre mi...

- Ingeniera Civil en Ciencia de la Computación (en proceso).
- Magister (en proceso).
- Ayudante de Arqui por 5 semestres :)
- Cumpleaños 28 de septiembre

# Para sacar pica jejej



# Contenidos:

- Números enteros y de punto flotante: representación, almacenamiento y operaciones
- Lógica digital, circuitos combinacionales y la ALU
- Circuitos secuenciales, registros y memorias
- El computador básico: componentes, instrucciones y su ejecución, el datapath, control de flujo, y subrutinas
- Control de flujo y subrutinas
- La arquitectura del set de instrucciones (ISA)
- Pipelining
- Memoria principal, caches y memoria virtual
- Input/output: dispositivos, comunicación con CPU y memoria
- Paralelismo

# Evaluaciones - Teóricas:

Esta versión del curso contara con 2 evaluaciones escritas y **presenciales**.

- **I1**: 14 de Octubre
- **I2/Examen**: 6 de Diciembre

Y dos actividades prácticas que se realizaran en horario de clases, la asistencia es **obligatoria**.

- **AP1**: 12 de Septiembre
- **AP2**: 24 de Octubre

# Evaluaciones - Practicas:

A lo largo del curso desarrollaran un computador básico. Los milestones son:

- Evaluaciones **Individuales**:
  - **Laboratorio 1**: 23 de Agosto. Elaboración de un circuito sumador de 4 bits.
  - **Laboratorio 2**: 30 de Agosto. Desarrollo de una unidad aritmética lógica.

- Evaluaciones **Grupales**:
  - **Etapa 1**: 27 de Septiembre
  - **Etapa 2**: 8 de Noviembre
  - **Etapa 3**: 29 de Noviembre

# Evaluaciones - Practicas:

A lo largo del curso desarrollaran un computador básico. Los milestones son:

- Evaluaciones **Individuales**:
  - **Laboratorio 1**: 23 de Agosto. Elaboración de un circuito sumador de 4 bits.
  - **Laboratorio 2**: 30 de Agosto. Desarrollo de una unidad aritmética lógica.

- Evaluaciones **Grupales**:
  - **Etapa 1**: 27 de Septiembre
  - **Etapa 2**: 8 de Noviembre
  - **Etapa 3**: 29 de Noviembre

**Importante!!** Empiecen a descargar vivado desde ya y a formar los grupos, deben ser de la misma sección. Tienen tiempo hasta el 16 de Agosto para la instalación y los primeros 30 tienen una bonificación de 5 décimas.

# Criterios:

Para contar con el mínimo para pasar el curso, debes cumplir los siguientes criterios. El calculo de la nota de proyecto (NP) estará en el enunciado de este mismo.

- $NF \coloneqq 0.2\ I1 + 0.2\ I2 + 0.1\ Ap1 + 0.1\ Ap2 + 0.4\ NP$

- $0.5\ I1 + 0.5\ I2 \geq 3.7$
- $NP \geq 3.7$
- $NF \geq 3.9$

# Integridad Academica:

- No está permitido el uso de ChatGPT.
- El uso indebido de herramientas y/o material externo será sancionado con nota 1.1 en el promedio final, informando a la DiPre.
- Si se usa material externo a la versión **actual** del curso debe citar. En caso contrario es considerado una falta a la integridad académica.
- http://www.uc.cl/codigo-de-honor/

# Consultas y contacto:

- Cualquier consulta que me quieran hacer acerca del ramo a través de mi mail sfigueroa3+iic2343@uc.cl. Por favor indicar la sigla del curso en el asunto (para no ghostearlos).
- También pueden comunicarse con el ayudante jefe Felipe Valenzuela (frvalenzuela@uc.cl).
- Y, por último, si necesitan comunicarse conmigo de manera presencial, me pueden encontrar en la sala 19O del departamento de computación.

# Cuerpo de Ayudantes:

- Ayudante Jefe: Felipe Valenzuela (frvalenzuela@uc.cl)
- Ayudante Jefe Proyecto: Diego Rodríguez (darodriguez6@uc.cl) y Ignacio Pasten (pasten.ig@uc.cl)
- Ayudante Jefe Catedra: Alberto Agostini (alberto.agostini@uc.cl)

# Consideraciones (Importante):

- Antes de la clase leer los apuntes correspondientes.
- Si bien las salas de ayudas no son obligatorias, es muuuuuyyyy recomendable asistir.

# Bibliografia:


- “Apuntes”, de Alejandro Echeverría y Hans Löbel, “Clases” y otros, disponibles en el sitio del curso
- D.A. Patterson, J.L. Hennessy, Computer Organization and Design risc Edition: The Hardware Software Interface (2nd ed.), Morgan Kaufmann (Elsevier) 2021
- A.S. Tanenbaum, T. Austin, Structured Computer Organization (6th ed.), Pearson Education Limited 2013

# Preguntas...?

# Ahora si... Qué consideramos un computador?

Todo computador es capaz de ejecutar cuatro funciones básicas.

- Leer datos
- Escribir datos
- Procesar datos
- Almacenar datos

# Ahora si... Qué consideramos un computador?

Componentes básicas de un computador.

- CPU
- Unidad de datos (Data path)
- Unidad de control (Control Unit)
- Memoria
- I/O (input-output)

# Ahora si... Qué consideramos un computador?

- Los computadores son capaces de ejecutar aplicaciones muy complejas, pero realmente son solo secuencias de acciones básicas, como sumar dos números o desplazarlo en una posición.
- Los programas complejos son “traducidos” a estas instrucciones básicas que el computador es capaz de entender y ejecutar.
- Realmente lo que el computador puede entender son secuencias de 0s o 1s. Cada digito de esta secuencia binaria se conoce como **bit**.
- El computador va a entender esta secuencia de binaria números como una instrucción a ejecutar. Por ejemplo, 1001010100101110 puede significar que el computador debe sumar dos números.
- En un principio los programadores se “comunicaban” con el computador a través de estas secuencias de números binarios. Como era muy complicado, se crearon lenguajes simbólicos, lenguajes de **assembly**, para representar estas instrucciones. Por ejemplo,

  ADD A, B ⟶ 1001010100101110 (sumar dos números A y B)

  Ya no es necesario memorizarse el numero binario y se obtiene una notación más *human-readable.*
  El responsable de traducir el lenguaje simbólico, **assembly**, a el lenguaje de maquina es el **assembler**.

# Ahora si... Qué consideramos un computador?

- Ahora imagínense como seria programar un código que realice el cálculo del centro de masas de un cuerpo 3D que sea recibido a través de inputs por un usuario solamente utilizando instrucciones básicas como suma de dos números o comparaciones… bastante difícil verdad?
- Es por esto que se diseñaron lenguajes de alto nivel. Podemos tener instrucciones más complicadas que después un **compilador** va a traducir al lenguaje de **assembly** de la máquina que lo ejecuta.

```
swap(size_t v[], size_t k):
        size_t temp
        temp = v[k]
        v[k] = v[k+1]
        v[k+1] = temp
```

Código en un lenguaje de alto nivel

Compilador

```
swap:
        slli x6,x11,3
        add
        x6,x10,x6
        lw      x5,0(x6)
        lw      x7,4(x6)
        sw      x7,0(x6)
        sw      x5,4(x6)
        jalr x0,0(x1)
```

Código en lenguaje assembly RISC-V

Assembler

```
00000000 00000000 00000000 00010011
00000000 00000000 00000000 00110011
00000000 00000000 00000000 00000011
00000000 00000000 00000000 00000011
00000000 00000000 00000000 00100011
00000000 00000000 00000000 00100011
00000000 00000000 00000000 00100111
```

Código de maquina


DCC

DEPARTAMENTO DE CIENCIA
DE LA COMPUTACIÓN

# IIC2343 – Arquitectura de Computadores

## Clase 0 – Introducción

Susana Figueroa