Simulador de Computador Digital

Título del Proyecto: Simulación de un Computador Digital Básico

Asignatura: Arquitectura del Computador

Semestre: 5º

Observación: este proyecto se debe realizar en pareja y debe llevar a parte de la simulacion **un informe** que explique de manera detalla el código y cómo funciona el simulador desarrollado. Fecha de entrega 10 de Diciembre 2025 al email: tareas.uneg@gmail.com y la defensa será la misma fecha por Google Meet (El alumno que no defienda no tendrá nota).

Objetivo General:
Diseñar y desarrollar en **Python** una simulación funcional de un computador digital básico, enfocándose en la **interacción directa entre la CPU y la Memoria RAM** para ejecutar un programa. El simulador incluirá una interfaz gráfica simple con **Pygame** para visualizar el estado de los componentes y el ciclo de instrucción.

Objetivos Específicos:

1. Implementar un modelo de **CPU** con una **ALU** básica y una **Unidad de Control** que ejecute el ciclo de instrucción.
2. Simular una jerarquía de memoria simplificada: **Memoria RAM** y un **Disco Duro** (archivo de texto) para cargar programas.
3. Modelar dispositivos de E/S esenciales: un **teclado** para entrada y una **pantalla** de texto para salida.
4. Implementar de forma clara y visible el ciclo **Fetch-Decode-Execute**.
5. Desarrollar una visualización gráfica que muestre el estado de los registros de la **CPU**, el contenido de la **RAM** y la salida del programa.
6. Crear y ejecutar un programa simple en el lenguaje máquina del simulador para probar su funcionalidad.

Requerimientos técnicos detallados

a) Unidad Central de Procesamiento (CPU)

Deberá ser una clase CPU que contenga:

- **Registros Internos Esenciales:**

  - ✓ PC (Program Counter): Apunta a la siguiente instrucción en RAM.
  - ✓ IR (Instruction Register): Contiene la instrucción actual.
  - ✓ AC (Accumulator): Registro para operaciones y transferencia de datos.
  - ✓ FLAG_Z (Zero Flag): Se activa (1) si el resultado de una operación es cero.

b) Conjunto de Instrucciones (ISA - Simplificado):

Para mantener la viabilidad, se definirá un lenguaje máquina propio con instrucciones de longitud fija (ejemplo: 16 bits: 4 bits para el código de operación y 12 para el operando/dirección).

Opcode (Hex)	Mnemónico	Descripción
0x1	LOAD addr	Carga el dato de la dirección addr de la RAM en el AC.
0x2	STORE addr	Almacena el valor del AC en la dirección addr de la RAM.
0x3	ADD addr	Suma el dato en addr al AC. El resultado queda en el AC.
0x4	SUB addr	Resta el dato en addr al AC. El resultado queda en el AC.
0x5	JUMP addr	Cambia el PC a la dirección addr.
0x6	JZ addr	Cambia el PC a addr solo si el FLAG_Z está activado (es 1).
0x7	IN	Lee un carácter desde el "teclado" y lo almacena en el AC.
0x8	OUT	Envía el valor del AC (interpretado como un carácter ASCII) a la "pantalla".
0x9	LOADI val	Carga un valor inmediato (val) directamente en el AC.
0xF	HALT	Detiene la ejecución del programa.

- ✓ **Unidad de Control (CU):** Será la lógica que orquesta el ciclo Fetch-Decode-Execute. No necesita ser una clase separada, puede ser el método principal (run_cycle()) de la clase CPU.
- ✓ **Unidad Aritmético-Lógica (ALU):** Un conjunto de funciones que realizan las operaciones (ADD, SUB). Debe actualizar el FLAG_Z.

c) Sistema de Memoria

- ✓ **Disco Duro (Almacenamiento Persistente):**
  - **Implementación:** Un archivo de texto (ej. programa.txt). Cada línea contiene una instrucción máquina (ej. 1 255).
- ✓ **ROM / Bootloader (Función de Arranque):**
  - **Implementación:** Una función de inicialización del simulador.
  - **Función:** Al iniciar, lee el programa.txt y lo carga en la Memoria RAM. Inicializa el PC a 0.
- ✓ **Memoria RAM:**
  - **Implementación:** Una lista o array de Python de tamaño fijo (ejemplo. 256 celdas es más que suficiente). RAM = [0] * 256.

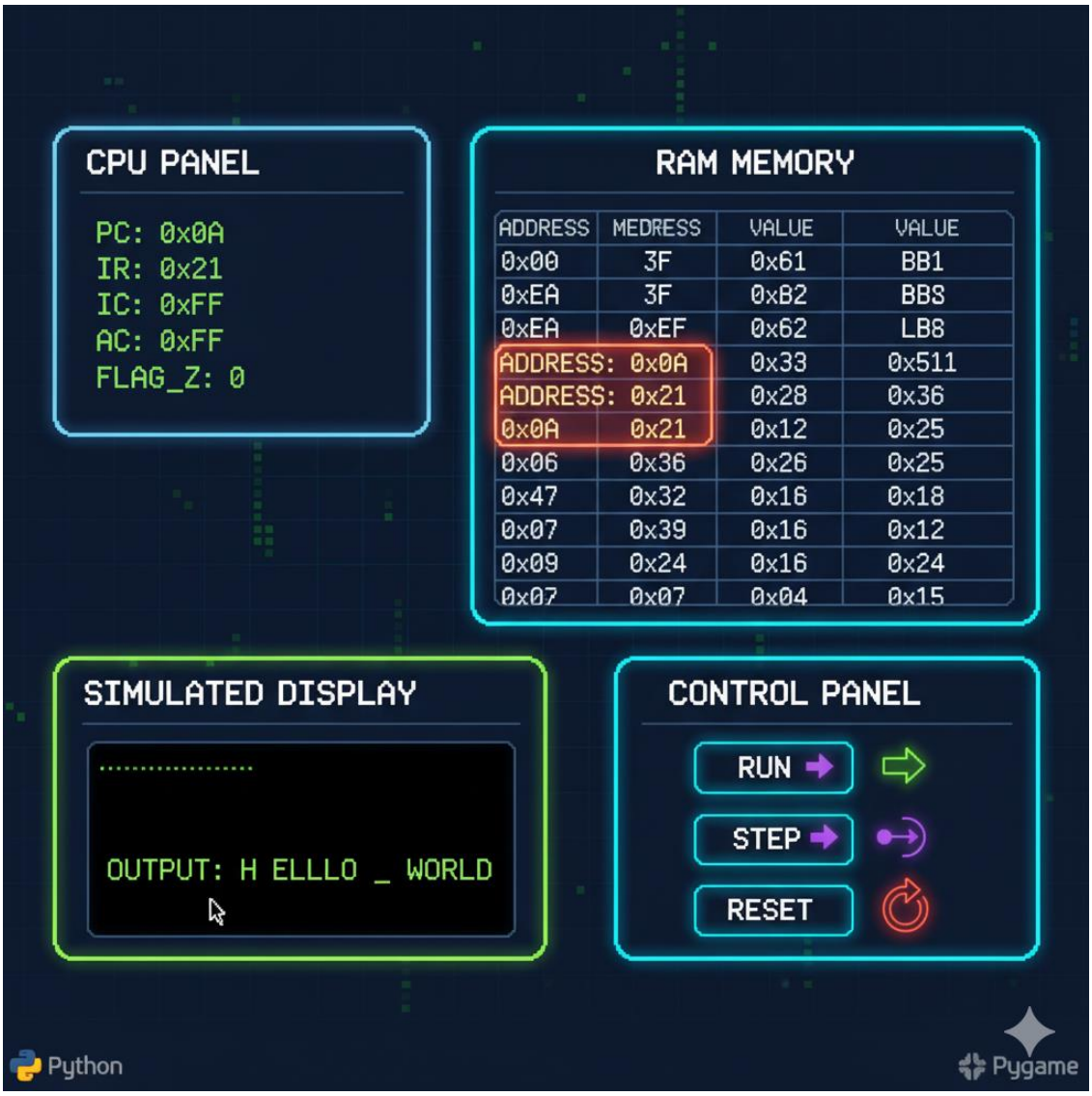
d) Dispositivos de Entrada/Salida (E/S)

- ✓ **Teclado (Entrada):**
  - **Implementación:** Utilizando el bucle de eventos de Pygame. La instrucción IN pausa la simulación hasta que se presiona una tecla. El valor ASCII de la tecla se carga en el AC.
- ✓ **Pantalla (Salida - Modelo Teletipo):**
  - **Implementación:** Un área designada en la ventana de Pygame.
  - **Funcionamiento:** Se mantiene una lista de cadenas de texto. Cuando se ejecuta la instrucción OUT, el carácter correspondiente al valor del AC se añade a la última línea de esta lista. La visualización de Pygame simplemente renderiza estas líneas de texto.

e) Visualización de la Interfaz Gráfica con Pygame

La interfaz gráfica debe ser limpia y funcional, dividida en **tres secciones claras**:

- ✓ **Panel CPU:** Muestra los valores de PC, IR, AC y FLAG_Z.
- ✓ **Panel Memoria RAM:** Una grilla que muestra las direcciones y los valores. Se debe resaltar la celda de memoria que está siendo leída o escrita en el ciclo actual.
- ✓ **Panel Pantalla Simulada:** Un cuadro de texto simple donde aparece la salida del programa (los caracteres de la instrucción OUT).
- ✓ **Panel de Control:** Botones indispensables:
  - **Ejecutar (Run):** Inicia la ejecución continua del programa hasta un HALT.
  - **Paso a Paso (Step):** Ejecuta un solo ciclo de instrucción (Fetch-Decode-Execute).
  - **Reiniciar (Reset):** Vuelve la simulación a su estado inicial, recargando el programa en la RAM.



Aplicación de la Computadora Simulada

Para que puedan probar el computador digital simulado, deben escribir un programa en el lenguaje máquina definido. La idea de programa sería:

Programa: "Calculadora de Suma Simple Interactiva"

Este programa demostrará el uso de E/S, almacenamiento en memoria y aritmética.

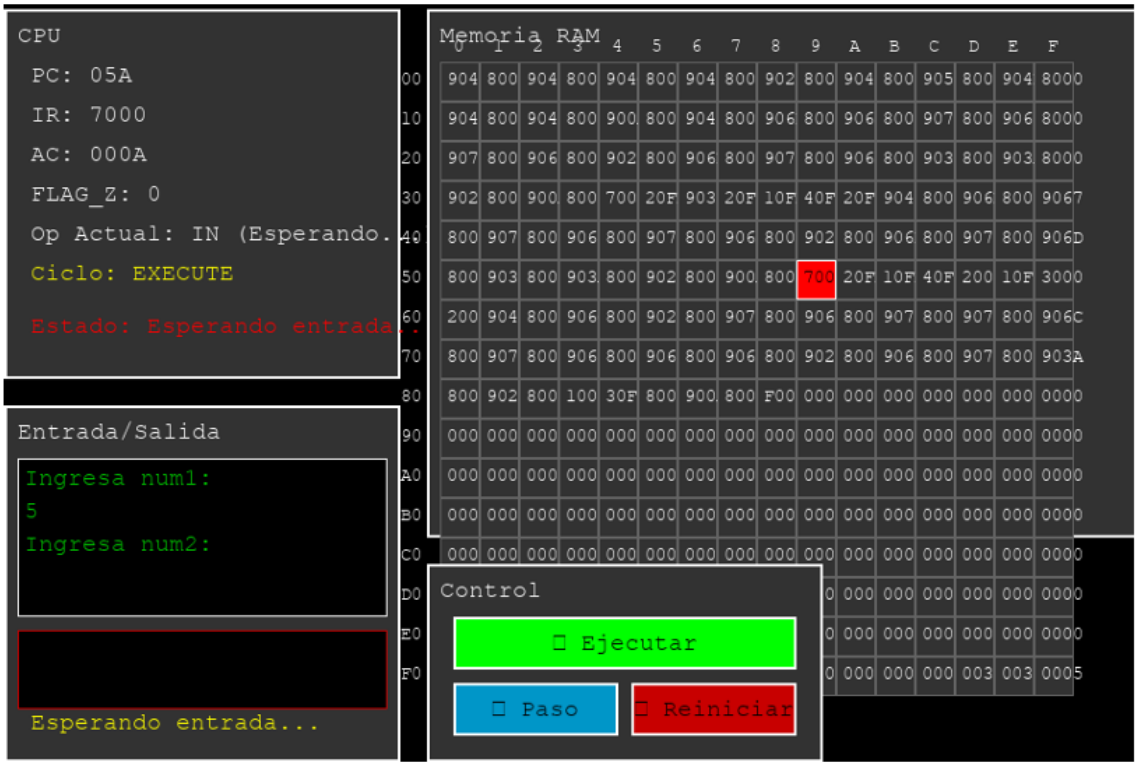
Pseudocódigo del programa:

1. Mostrar un mensaje de bienvenida en pantalla (usando varios OUT).
2. Pedir al usuario el primer número.
3. Esperar la entrada del teclado (IN).
4. Guardar el número en una dirección de memoria (ejemplo. 250) (STORE 250).

5. Pedir al usuario el segundo número.
6. Esperar la entrada del teclado (IN).
7. Sumar el segundo número al primero, que está guardado en memoria (ADD 250).
8. El resultado ahora está en el AC.
9. Mostrar un mensaje tipo "El resultado es: ".
10. Mostrar el resultado en pantalla (OUT).
11. Detener el programa (HALT).

Nota: Dado que la entrada del teclado son caracteres, los alumnos tendrán que lidiar con la conversión de caracteres '0'-'9' a sus valores numéricos (restando el código ASCII de '0'), lo cual es en sí mismo un excelente ejercicio de bajo nivel.

Ejemplo de Mini-Proyecto de Alumno 1 de semestre pasado



Ejemplo de Mini-Proyecto de Alumno 2 de semestre pasado