Ministry of Higher Education and Scientific Research
Badji Mokhtar Annaba University
Faculty of technology
Departement of electronics

Practical Work Mp and Mc

PW n° 1

Introduction to the Microprocessor Programming Environment

These practical sessions were developed for the Microprocessors and Microcontrollers lab module of L3 Automation, for the 2025/2026 academic year within the Department of Electronics Badji Mokhtar ANNABA University.

By Dr. MERABTI Nardjes

Objective

The main objective of this first practical session is to **become familiar with the tools used throughout the Microprocessors and Microcontrollers module**.

In this introductory lab, students will explore the **basic operation of an 8-bit microprocessor** and learn how to use the **GNUSim8085** software: a simple and educational simulator for the Intel 8085 assembly language for:

- Create and manage a **new project** using GNUSim8085;
- **Edit, assemble, and execute** a basic assembly program;
- **Observe the internal operation** of the microprocessor (registers, memory, and instruction cycles).

- Use the built-in **debugging and step-by-step simulation tools** to understand program execution.

- **Resources** → A PC equipped with the **GNUSim8085 software**.

❖ GNUSim8085 software by intel:

GNUSim8085 is an open-source, cross-platform simulator for the Intel 8085 microprocessor. It provides an integrated environment for **writing, assembling,** and **debugging** assembly programs. The tool simplifies learning and development by allowing users to test and understand the processor's behavior before applying it to real hardware.

1. First Steps with GNUSim8085

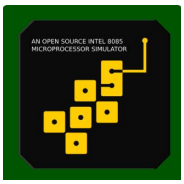
✓ **Installing the Software:** Download GNUSim8085

Go to the official GNUSim8085 page or a trusted software repository;

Download the version compatible with your operating system (Windows).

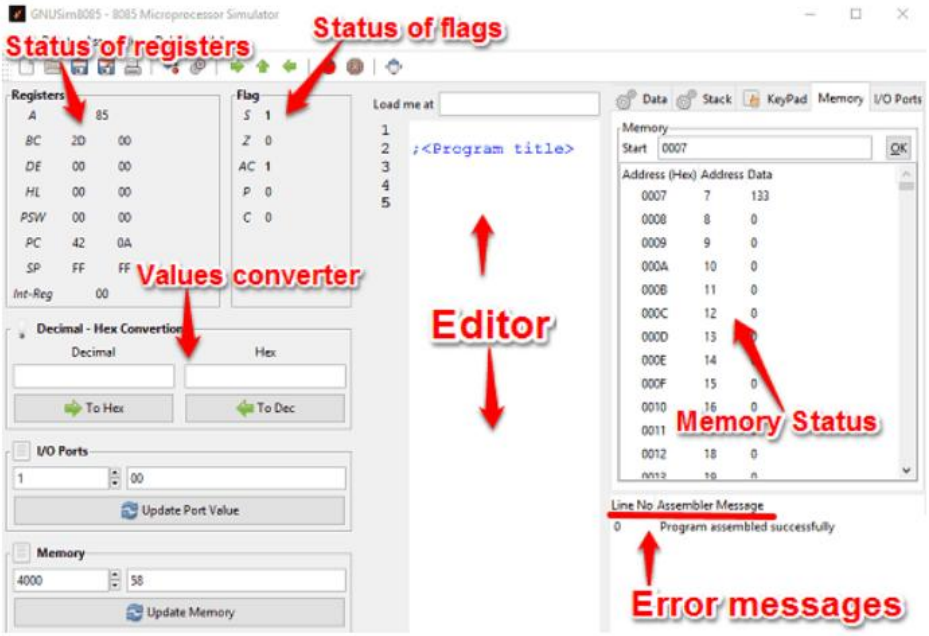
Follow the installation instructions. The process is simple and requires no special

configuration; Once installed, you should see the GNUSim8085 icon on your desktop or in your applications menu.

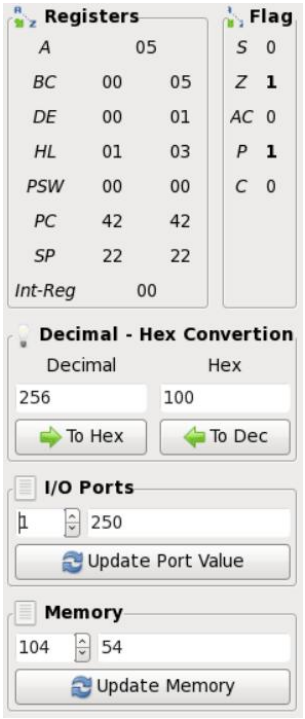


2. Features of GNUSim8085

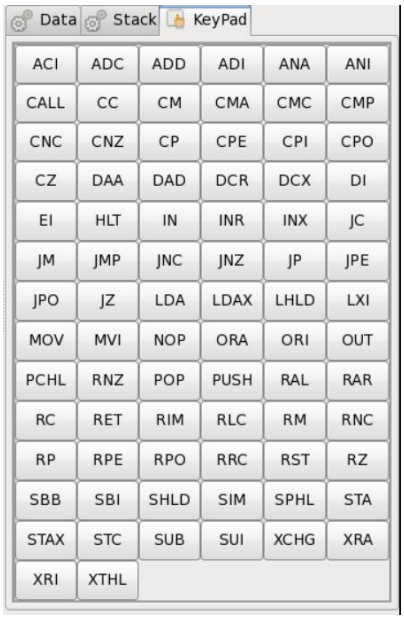
GNUSim8085 provides a digital simulation of the Intel 8085 microprocessor, displaying registers, ports, memory, and flags in real time. It includes a built-in assembler and an editor with syntax highlighting, enabling easy program creation and debugging. The software also offers I/O simulation, stack visualization, and a hexadecimal-to-decimal converter for enhanced usability.



When the software is launched, the editor starts with a template-sample code you can edit-or, you can open a **new file**, and start coding. The editor lines are numbered. The code highlighting is good, and makes it easier to read the code. To run or assemble code, you need to save it first, before compilation is allowed.

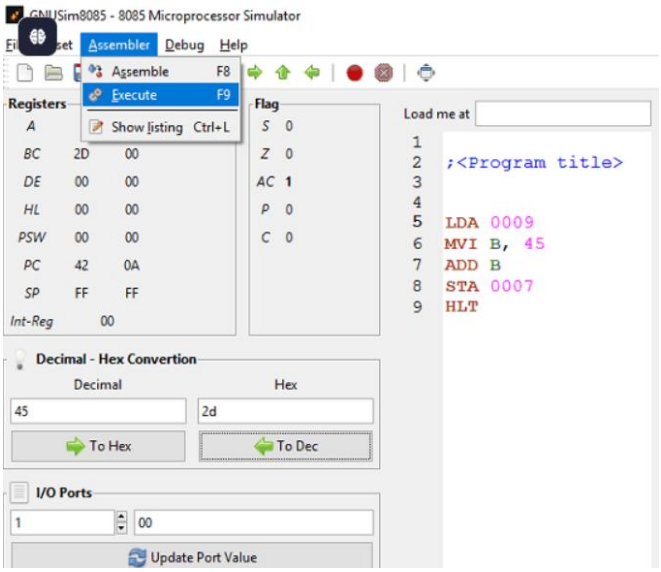


The left panel of **GNUSim8085** provides a clear display of all **registers and flags**, with their values shown in **hexadecimal format**. It also includes a convenient **hex-to-decimal converter**. Below this section, **I/O ports** and **memory spaces** are represented using spin boxes, where the index indicates the port or memory address and the adjacent field shows its current value.



The third tab contains a virtual keypad, consisting of keys corresponding to each instruction.

In GNUSim8085, the **Assemble (F8)** command converts the source code into machine code and loads it into memory. The **Execute (F9)** command runs the compiled program. The **Show Listing (Ctrl + L)** option displays the assembled machine code with addresses, opcodes, mnemonics, and comments — a useful feature for transferring and running the program on a real 8085 system, eliminating the need for manual assembly.



*** Example 1 :**

→ **Step 1**: Open the interface

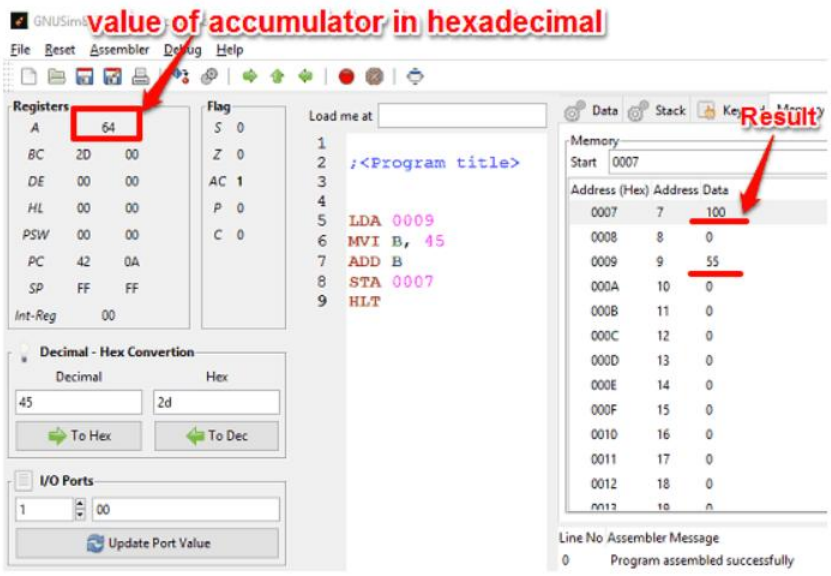
→ **Step 2:** In the editor area, you can start typing your 8085 program

```
LDA 0009H  ; Load the value from memory address 0009H into the accumulator (A)
MVI B, 45H  ; Move the immediate value 45H into register B
ADD B  ; Add the contents of register B to accumulator A (A = A + B)
STA 0007H ; Store the result from accumulator A into memory address 0007H
HLT ; Stop program execution (halt the processor)
```

when done, simply hit the **F9** key to start execution.

→ **Step 3**: If there are any errors, then they will be listed under Assembler Message section with the line number. Also, you can see the status of the used flags and registers after the execution of the program.

You can see the following screenshot, in which I have executed a very simple program that calculates the sum of 2 numbers.



*** Example 2:**

Enter and execute this program to practice using the software and simulating an assembly code.

```
MVI A, FF_H      ; Load FFH into register A
LXI B,0FF0_H     ; Load 0FF0_H into register pair BC
L1: STAX B       ; Store the contents of A at the memory location pointed to by BC
   INX B         ; Increment the BC register pair
   DCR A         ; Decrement A by 1
   JNZ L1        ; Jump to L1 if A is not zero
   STAX B        ; Store 00H at memory address FFFFH
   HLT           ; Stop the processor
```