Class:	Spring-2025	**Subject:**	Computer Architecture
Course Code:		**Lab Instructor:**	Mr. Tabassum Javed

---------- LAB 07----------

Learning Objective:

• Understand the Clear Screen (CLS) operation.

• Define and implement Complement Accumulator (CMA) in CPU Sim.

• Observe how Skip if Positive (SPA) and Skip if Negative (SNA) affect instruction flow.

• Apply CLS, CMA, SPA, and SNA in conditional execution and control flow.

Essential Tools in Our Lab:

- **Computer System:** The main machine
- **CPU Sim**: CPU Sim is a tool used for simulating simple CPU architectures, helping students understand processor design, instruction execution, and debugging.

Program-1

```
START:
INP       ; Take an input from the user
STA NUM   ; Store the input value in memory location NUM

INP       ; Take another input from the user
ADD NUM   ; Add the value stored in NUM to the second input
STA NUM   ; Store the input value in memory location NUM

SPA
ADD NUM1
OUT       ; Output the result
HLT       ; Halt (end program)
NUM: .data 1 0  ; Define memory location NUM, initialized with 0
NUM1: .data 1 100
```

Basic Machine Instructions

Basic Computer Instructions

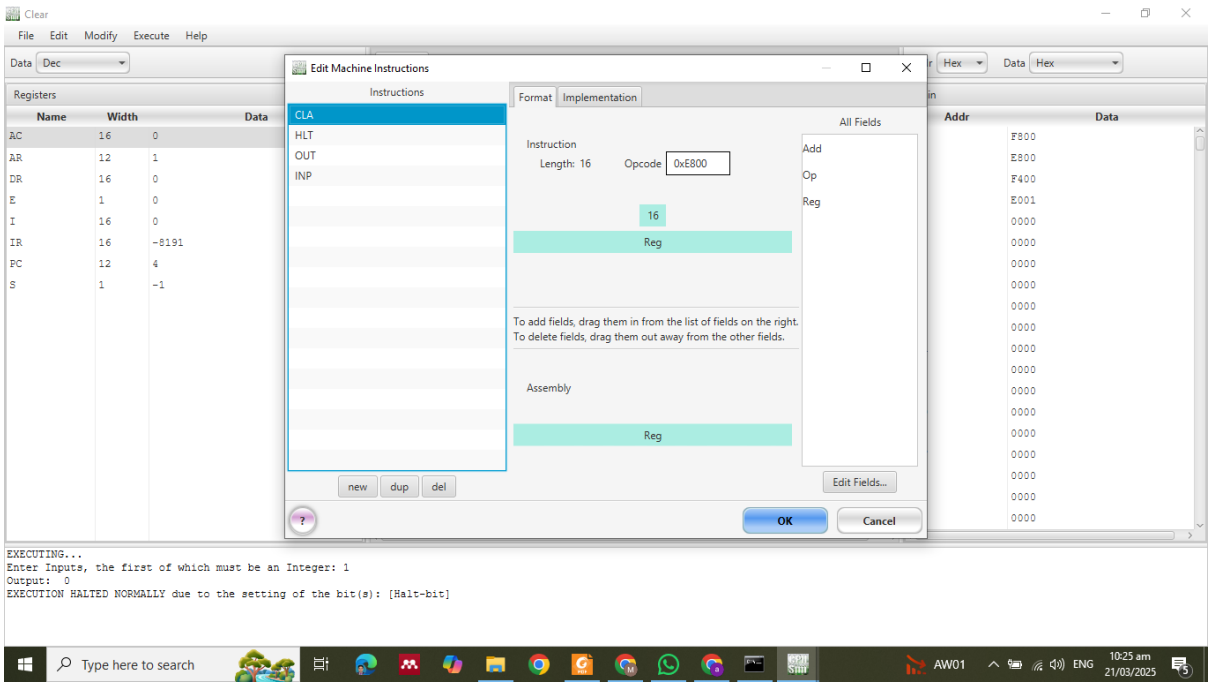
Memory Reference			Register Reference		
Symbol	Hex		Symbol	Hex	
AND	0xxx	Direct Addressing	CLA	E800	
ADD	2xxx		CLE	E400	
LDA	4xxx		CMA	E200	
STA	6xxx		CME	E100	
BUN	8xxx		CIR	E080	
			CIL	E040	
ISZ	Cxxx		INC	E020	
AND_I	1xxx	Indirect Addressing	SPA	E010	
ADD_I	3xxx		SNA	E008	
LDA_I	5xxx		SZA	E004	
STA_I	7xxx		SZE	E002	
BUN_I	9xxx		HLT	E001	
ISZ_I	Dxxx				

Step-by-Step Execution:

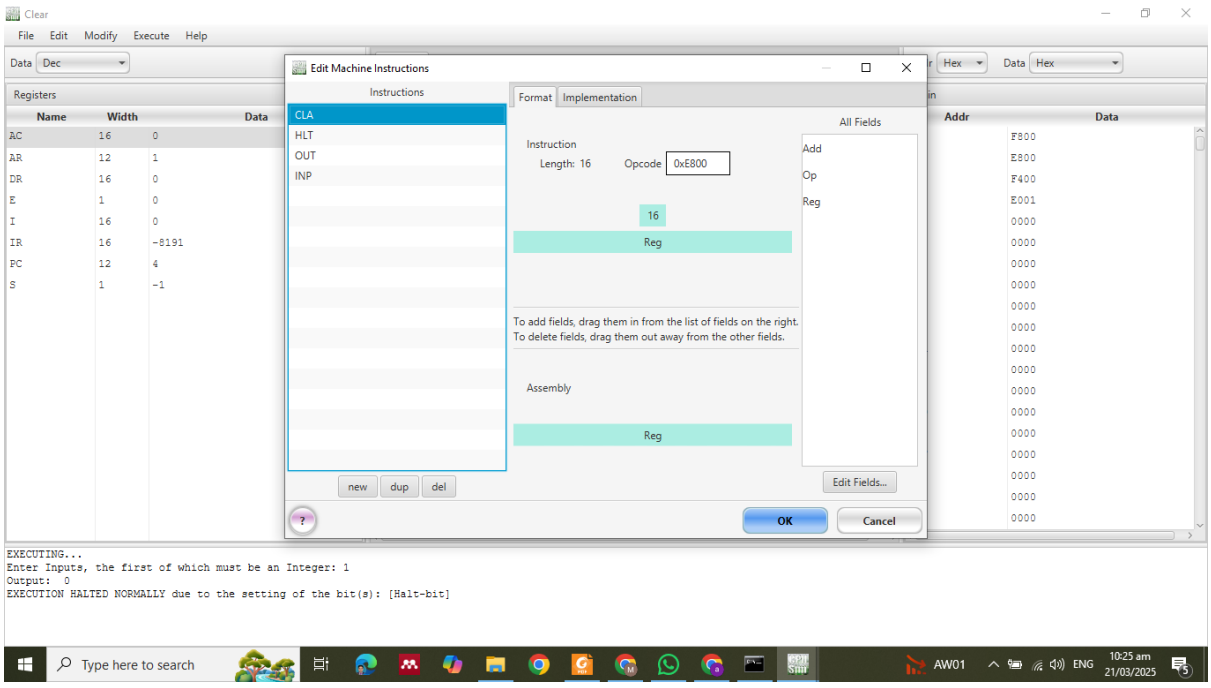
1. **ISZ 009 (Increment and Skip if Zero)**
   - **Reads the value stored at memory location 009.**
   - **Increments it by 1.**
   - **If the new value becomes 0, it skips the next instruction (OUT).**
   - **Otherwise, it proceeds to the next instruction.**
2. **OUT (Output the value in AC)**
   - **If ISZ does not skip, this instruction executes.**
   - **Outputs the value in the accumulator (AC) to the display/output device.**
3. **HLT (Halt Execution)**
   - **Stops program execution.**

Step 1: Navigate to the Microinstructions Section

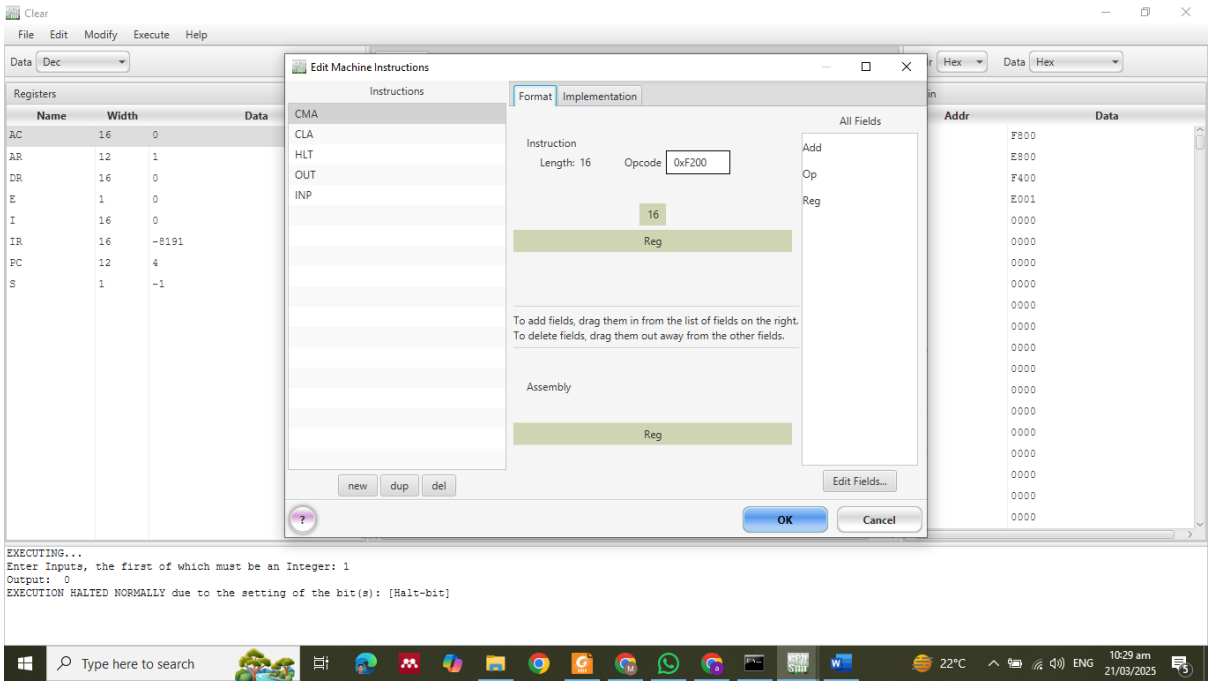
- **Open your simulator/tool and access the Microinstructions Section.**

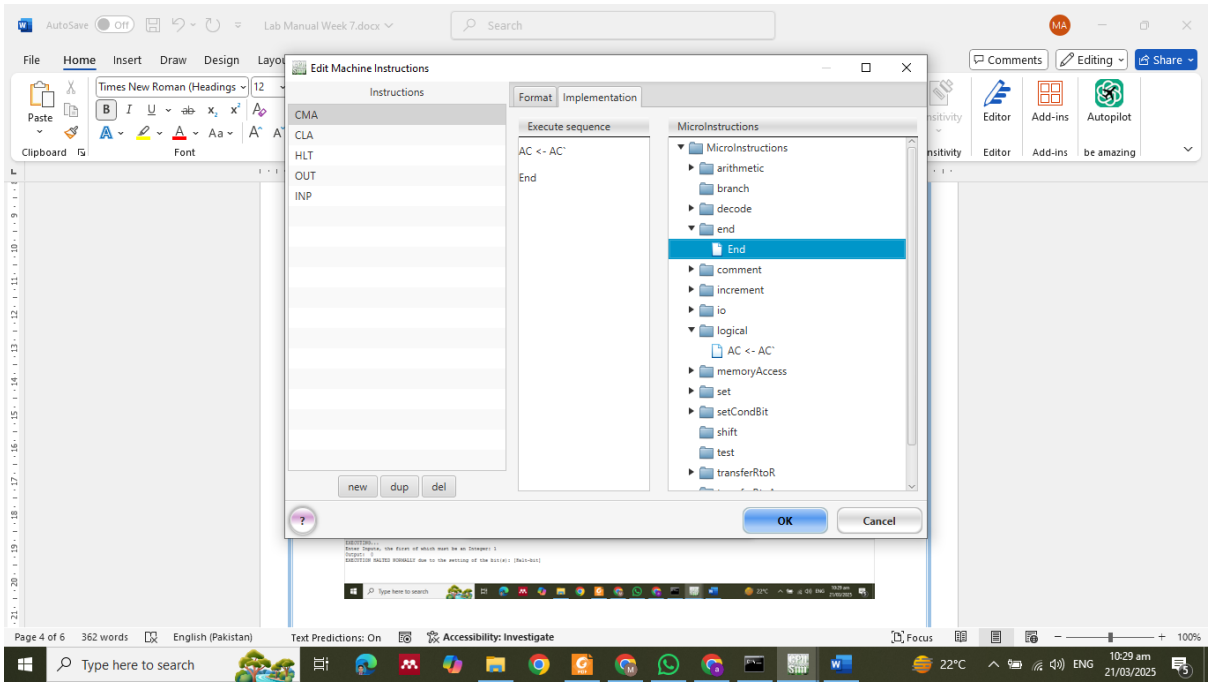


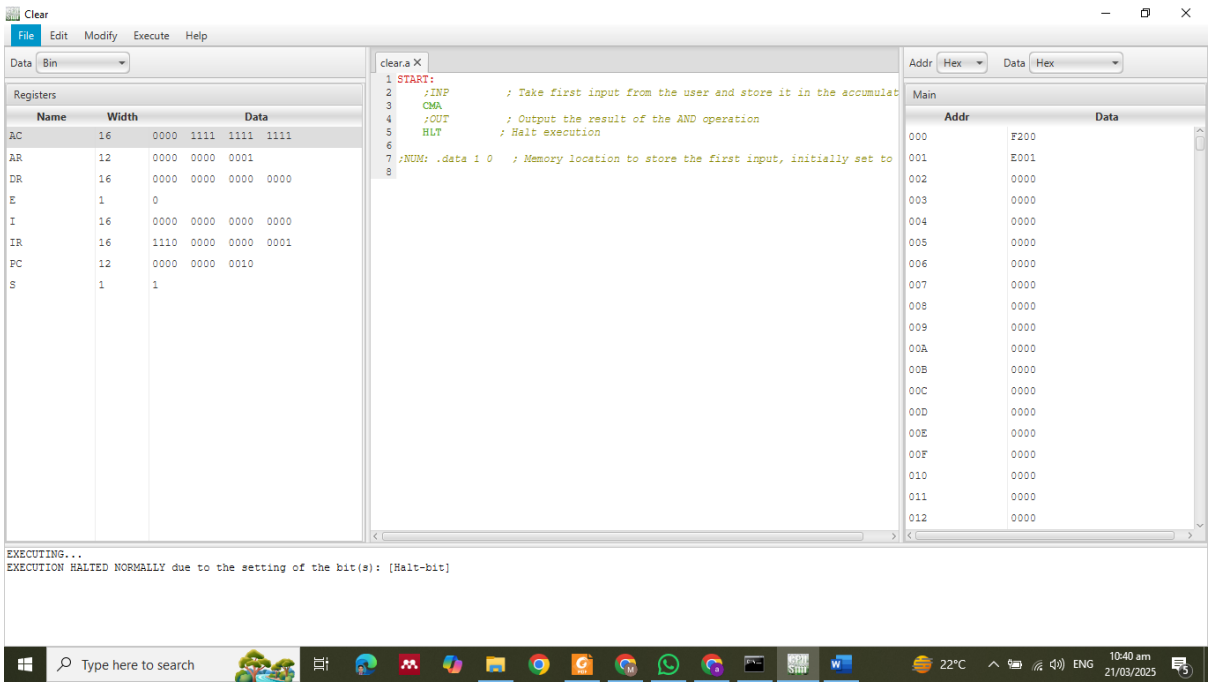
Step 2: Define CLA Microinstruction



Step 2: Define CMA Microinstruction

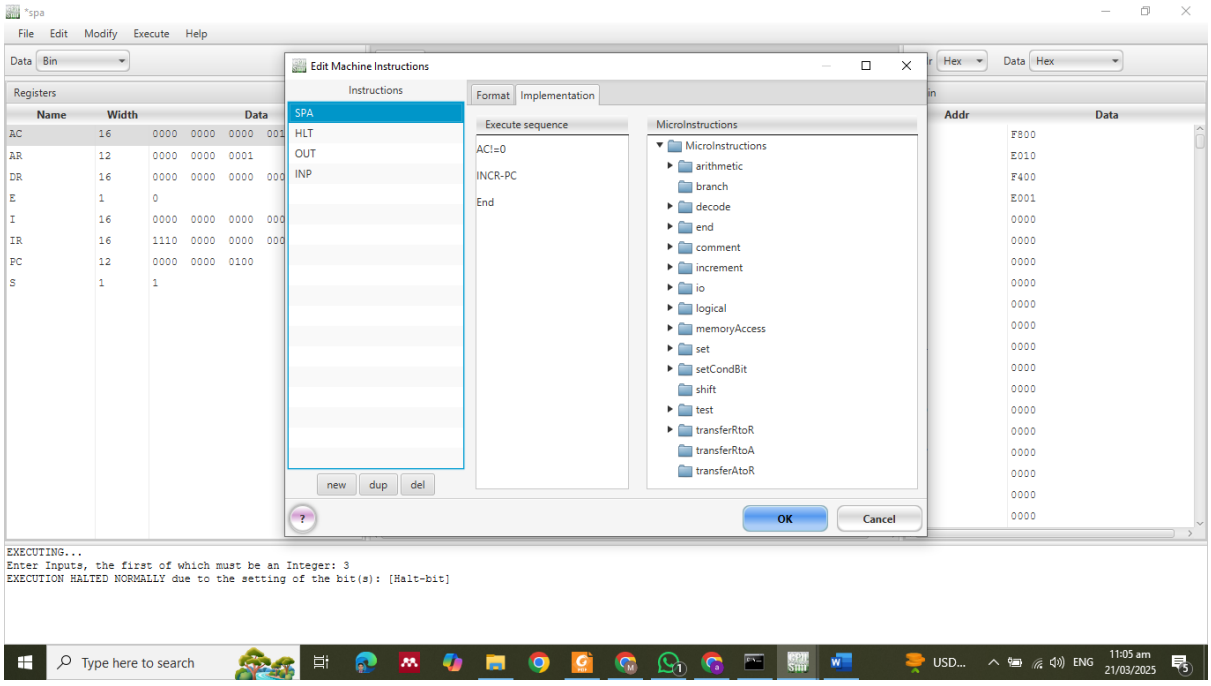



Edit Machine Instructions
Instructions
CMA
CLA
HLT
OUT
INP
Format
Implementation
Execute sequence
AC <- AC`
End
MicroInstructions
arithmetic
branch
decode
end
End
comment
increment
io
logical
AC <- AC`
memoryAccess
set
setCondBit
shift
test
transferRtoR
new
dup
del
OK
Cancel

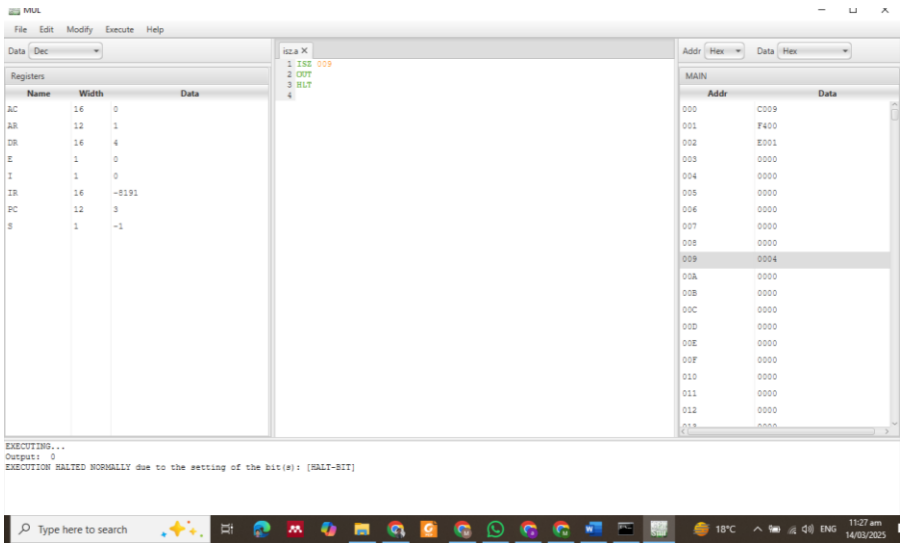

Clear
File
Edit
Modify
Execute
Help
Registers
Name
Width
Data
AC 16 0000 1111 1111 1111
AR 12 0000 0000 0001
DR 16 0000 0000 0000 0000
E 1 0
I 16 0000 0000 0000 0000
IR 16 1110 0000 0000 0001
PC 12 0000 0000 0010
S 1 1
clear.a
1 START:
2 ;INP ; Take first input from the user and store it in the accumulat
3 CMA
4 ;OUT ; Output the result of the AND operation
5 HLT ; Halt execution
6
7 ;NUM: .data 1 0 ; Memory location to store the first input, initially set to
8
Main
Addr
Data
000 F200
001 E001
002 0000
003 0000
004 0000
005 0000
006 0000
007 0000
008 0000
009 0000
00A 0000
00B 0000
00C 0000
00D 0000
00E 0000
00F 0000
010 0000
011 0000
012 0000
EXECUTING...
EXECUTION HALTED NORMALLY due to the setting of the bit(s): [Halt-bit]

Working Mechanism of SPA

1. **Checks the sign bit of the accumulator:**
    - **If the most significant bit (MSB) is 0, the value is positive, and the next instruction is skipped.**
    - **If the MSB is 1, the value is negative, and execution continues without skipping.**



Results :



Lab Task-1

Task Requirements:

1. Design a machine to solve the Program-1 given at start of the document.

Lab Task-2

Read the following code, write explanation of each line in front of each line. Implement this code using CPUSim submit screenshot and .cpu machine along with this code.

```
    START:
1.      INP
2.      STA NUM
3.      LDA NUM
4.      SPA
5.      JMP NEGATIVE

6.      CLA
7.      LDA POS_MSG
8.      OUT
9.      JMP END

10. NEGATIVE:
11.     SNA
12.     JMP ZERO
13.     CLA
14.     LDA NEG_MSG
15.     OUT
16.     JMP END

17. ZERO:
18.     CLA
19.     ISZ ZERO_MSG
20.     OUT
21. END:
22.     HLT

23. NUM:        .data 1 0
24. POS_MSG:    .data 1 1
25. NEG_MSG:    .data 1 -1
26. ZERO_MSG:   .data 1 0
```