Name: **Sahil Dattatray Mohite**

Roll No: **30** Batch: **B2**

PRN: 12010501

Lab Assignment 1

**Write ALP to display HELLO using 09h function of int 21h & explain 09h, 4ch functions of int 21h**.

Assembly language is a low-level programming language for a computer, or other programmable device specific to a particular computer architecture in contrast to most high-level programming languages, which are generally portable across multiple systems. Assembly language is converted into executable machine code an assembler like NASM, MASM etc.

Assembly language comment begins with a semicolon (;). It may contain any printable character including blank.

The **executable instructions** or simply **instructions** tell the processor what to do. Each instruction consists of an **operation code** (opcode). Each executable instruction generates one machine language instruction.

**Structure** of Assembly language is:

|  |
| --- |
| .model small  .data  <Declaration of variables>  .code  <Execution starts from here>  End |

**.model small**

* The memory model directive specifies the size of the memory the program needs. Based on this directive, the assembler assigns the required amount of memory to data and code.
* In the SMALL model all code is placed in one physical segment and all data in another physical segment. In this model, all procedures and variables are addressed as NEAR by pointing to their offsets only.

**Instructions used in this programs are**

1. **mov** — Move

The mov instruction copies the data item referred to by its first operand (i.e. register contents, memory contents, or a constant value) into the location referred to by its second operand (i.e. a register or memory). While register-to-register moves are possible, direct memory-to-memory moves are not. In cases where memory transfers are desired, the source memory contents must first be loaded into a register, then can be stored to the destination memory address.

*Syntax*  
mov <reg>, <reg>  
mov <reg>, <mem>  
mov <mem>, <reg>  
mov <con>, <reg>  
mov <con>, <mem>

1. **lea** — Load effective address

The lea instruction places the address specified by its first operand into the register specified by its second operand. Note, the contents of the memory location are not loaded, only the effective address is computed and placed into the register. This is useful for obtaining a pointer into a memory region or to perform simple arithmetic operations.

*Syntax*  
lea <mem>, <reg32>

**INT 21h**

* Int 21h is a dos interrupt. It is one of the most commonly used interrupt while writing code in 8086 assembly language.
* To use the dos interrupt 21h load **ah** with the desired sub-function. load other required parameters in other registers. and make a call to int 21h.

|  |  |
| --- | --- |
| AH=09h | WRITE STRING TO STANDARD OUTPUT |
| AH=4Ch | "EXIT" - TERMINATE WITH RETURN CODE |



