**Computer Organization & Assembly Language**

**Lab 02**

**Topics:**

1. Details of Registers
2. Assembly Instructions
3. Addressing Modes
4. ASCII codes
5. Assembler Directives
6. Variables and Strings
7. Declaration of variables in data segment

**Registers:**

* **Temporary storage of data**
* **Fastest storage area**
* **Quickly accessible by CPU**
* **Built into CPU**
* **Optimization of processing time**

**Assembly Instructions:**

* **There are different kinds of assembly instructions**
* **In order to pass any data to registers we use mov instruction**
* **To perform addition and subtraction of numbers, keywords, ADD and SUB are used**
* **Keywords INC and DEC are used**

**Addressing Modes:**

* **Register Addressing: In register addressing, both the operands are registers**
* **Immediate Addressing: In Immediate addressing, one of the operand is register and the other one is immediate value.**
* **Memory Addressing: In Memory addressing, one of the operands is register and the other access static data directly i.e (Store address of the memory)**

**Assembler Directives:**

* .MODEL directive
* .STACK directive
* .DATA directive
* .CODE directive

**Variables:**

**What are variables?**

* + Variables are used to store values.
  + The values of variable can be changed.

**Where to initialize variables in assembly program?**

* Variables are defined in .data directive of program structure.

**Naming Conventions for Variables:**

* Do not use reserved keywords for variable names.
* Reserved keywords are
  + - Operands (ADD, SUB, MUL, DIV, MOV, POP, PUSH)
    - Registers (AX, BX, CX, DX, DS, CS etc)

**Initializing a Variable:**

Variable name Variable Initializer Initializing value

Initializer directive defines the size of data. It is also known as Data type directive. Initializing value is the value assigned to the variable.

Example:

**VAR1 DB 49;** Declare a byte, referred to as location Var1, containing the value 49

**VAR2 DB ‘A’;** Declare a byte, referred to as location Var1, containing the value 65

**VAR3 DB ?;** Declare an uninitialized byte, referred to as location Var3

**Initializer Directives (Data Types)**

|  |  |  |
| --- | --- | --- |
| **NAME** | **STAND FOR** | **SIZE** |
| DB | Define Byte | 1 Byte/8 Bits |

**Variable Declaration in Program:**

Example Code:

*.DATA*

*VAR DB 49*

*.CODE*

*MOV AL, VAR*

**Strings:**

Syntax: str1 db “HELLO”,’$’

**String Operations:**

* Offset: Returns distance of a variable from the beginning of its segment
* Type: Returns an integer representing size of a variable. E.g., TYPE of word is 2
* Lengthof: Counts number of individual elements in a variable that has been defined
* Sizeof: It determines the total bytes occupied by a variable
* DUP: Allows a sequence of storage locations to be defined or reserved

**String Declaration in Program:**

Example Code:

*.DATA*

*STR1 DB “Lab 3”, ‘$’*

*.CODE*

*MOV DX, offset str1*

**Code Structure:**

.386

.model flat,stdcall

.stack 4096

.code

main PROC

; Move the values into the registers

mov al, 97

mov ax, 2

mov bx, 3

mov cx, 1

mov dx, 5

add ax, bx ; 5

add cx, dx ; 6

sub ax, cx ; The result is FFFFFFFF

main ENDP

END main

**Tasks:**

1. Write an assembly language program to Addition and Subtraction any value stored in the registers in memory. The add register will be ax or cx and subtract register will be bx, dx. Also try these on al, ah, bl, bh, cl, ch, dl, dh. Write down values of each registers in comments and also add screen shots of the watch.
2. Write an assembly language program to Addition and Subtraction any value stored in the registers in memory. The add register will be ax or cx and subtract register will be bx, dx. Also try these on al, ah, bl, bh, cl, ch, dl, dh. Those students whose roll number end on even number add and subtract value by 8 and for odd one add and subtract by 5. Write down values of each registers in comments and also add screen shots of the watch.
3. Write an assembly language program to ADD two values and see the values in memory. And Move one value to ax and other value to bx and watch the value of al, ah, ax ,bx, bh, bl. Note down each memory addressing and note down in comments. Write down values of each registers in comments and also add screen shots of the watch.
4. Write an assembly language program to add two values and see the values in memory. The two values must be one of you last 4 digits of roll number and the other must be yours friend. And Move one value to cx and other value to dx and watch the value of cl, ch, cx ,dx, dh, dl. Write down values of each registers in comments and also add screen shots of the watch.
5. Write an assembly language program to subtract two values and see the values in memory. The values must be one of the current year and the other one would your birth year. And Move one value to cx and other value to dx and watch the value of cl, ch, cx ,dx, dh, dl. Write down values of each registers in comments and also add screen shots of the watch.
6. Write assembly language to add two binaries number
7. Write an assembly language program to add two numbers in registers. One number is binary and the other one is hex. Please justify rather these add or not please specify what cause the issue along with the screenshot.
8. Write an assembly language program to add two numbers in registers. One number is binary and the other one is octal. Please justify rather these add or not please specify what cause the issue along with the screenshot.
9. Write an assembly language program to add two numbers in registers. One number is hex and the other one is octal. Please justify rather these add or not please specify what cause the issue along with the screenshot.
10. Write Assembly program to add two numbers using variables.
11. Write Assembly program to subtract two numbers using variables.
12. Declare and initialize string of your choice and store its length, size and type in separate variables.