Mahavir Polytechnic, Nashik

Department of Artificial Intelligence and Machine Learning

Year: SY **Subject: MIC (314321)**

UNIT 2: The Art Assembly Language Programming **Marks: 08**

Course Outcome 2: Use program development tools and assembler directives.

Syllabus:

2.1 Program Development Steps

• Defining Problem and constraints, writing Algorithms, flowchart, Initialization checklist

• Choosing instructions, converting algorithms into assembly language program

2.2 Assembly Language Programming Tools

• Editors, Assembler, Debugger, Linker

2.3 Assembler Directives

2.1 Program Development Steps

1. **Defining the problem:**

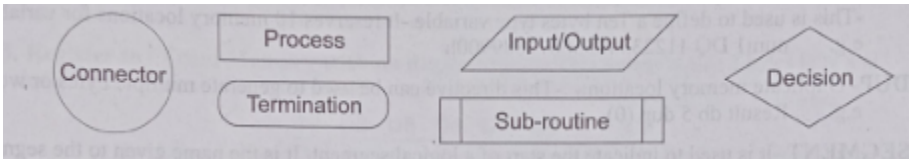
   The first step in writing program is to think very carefully about the problem that the program must solve.

2. **Algorithm:**

   The formula or sequence of operations to be performed by the program. It is step by step instruction required to solve any problem.

3. **Flowchart:**

   The flowchart is a graphically representation of the program operation or task. In this specific operation of task is represented by graphical symbol such as circle, diagonal square, parallelogram.



4. **Initialization Checklist:**

   Initialization task is to make the checklist of entire variables, constants, all the registers, flags and programmable ports, stack must be initialized properly.

   e.g. i) Data segment

   ii) Code segment

5. **Choosing Instructions:**

   Choose proper instructions that perform operations. Program should be smaller in size more efficient in execution.

6. **Converting algorithms to assembly language program:**

   Every step in the algorithm is converted into program statement using correct and efficient instructions or group of instructions.

2.2 Assembly Language Programming Tools

1. **Editor:**

   An editor in is a program which helps you to create your assembly language program in right format. We can type program using editor. The program file is created in editor is called as source file and it has .asm extension.

   Example: Editors are PC Write, Word star, DOS based editor as a turbo editor.

2. **Assembler:**

   An assembler program is used to translate the assembly language program into corresponding binary code for each instruction. i.e., machine code and generate object he with extension .obj

3. **Linker:**

   A linker is program used to join several objects files into one large object file. i.e. into one executable program.it generates .exe file.

   The linker produces a link file which contains the binary codes for all the combined modules. Linker also produces the Link Map file which contains the address info.

   Linker are TLINK Borland's Turbo.

4. **Debugger:**

   Debugger is a program that allows the execution of program in single step mode under the control of user. The process of locating and correcting errors using a debugger is known as debugging.

   Example: DOS Debug, Turbo Debugger TD.

2.3 Assembler Directives

1. **DB-Define byte (8 bits)**

   Used to declare a byte type variable of 8 bit. It also can be used to declare an array of bytes.

   e.g. a) num1 db 12h

   b) Array db 12h, 55h, 66h, 33h, 44h

2. **DW-(Define Word or Data word)**

   This is used to define a word (16-bit) type variable.

   e.g. a) num1 dw 1234h

   b) List dw 5 Dup (0)

3. **DD-(Define Double word/Data Double word)**

   This is used to define double word (32-bit variable -It reserves 4 memory locations for a variable.

   e.g. num1 DQ 11223344556677h

4. **DQ-(Define Quad Word 8 bytes -64bit)**

   This is used to define a 8 bytes type variable. It reserves 8 memory locations for a variable.

   e.g. num1 DQ 11223344556677h

5. **DT-(Define Ten bytes-80 bit)**

   This is used to define a Ten bytes type variable-It reserves 10 memory locations for variable

   e.g. num1DQ 11223344556677889900h

6. **DUP- (Duplicate Memory Location)**

   This directive can be used to generate multiple bytes or words Revalt de 5 dup (0)

   e.g. Result db 5 dup (0)

7. **SEGMENT-**

   It is used to indicate the start of a logical segment. It is the name given to the segment Code segment

   e.g. Code segment

   Ends

8. **LABEL-**

   LABEL, enables you to redefine the attribute of a data variable or function label.

   e.g. Array LABEL BYTE

9. **OFFSET-**

   It is an operator which sells the assembler to determine the offset or displacement of named data item from the start of the segment which contains it. It is used to offset of a variable into a register so that variable can be accessed with one of the addressed modes.

   e.g. MOV BX OFFSET PRICES (LEA SI, PRICES)

10. **EQU(Equate)-**

    Assigns a constant value to a symbol

11. **ORG(Origin)-**

    Sets the starting address for the following code or data.

12. **END-**

    Marks the end of the source file. It can also specify the starting address for execution.

13. **PROC(Procedure) and ENDP (End Procedure)-**

    Used to define the beginning and end of a procedure or function.

14. **ASSUME-**

    Tells the assembler which segment registers should point to specific segments.

Assignment 1

Sr. No	Questions	Attainment to which CO
1	State the Program Development Steps.	CO2
2	State the functions of Assembly Language Programming Development Tools.	CO2
3	Explain Assembler directives.	CO2