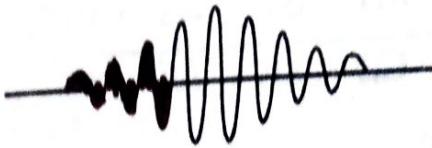


5



Debug and Assembler

DEBUG is a program which allows the programmer to write, execute and debug assembly language programs, and to examine the contents of registers and memory. It can also be used to run a program, one instruction at a time. ASSEMBLER is also a program for the same purpose but is more powerful and provides more facilities as compared to DEBUG.

5.1 DEBUG

To start DEBUG its name is typed after DOS prompt as shown below:

C:\>DEBUG

'Enter' key after a command.

If DEBUG program is in subdirectory its name is typed as follows:

C:\>CD\ DOS

C:\>DOS\ DEBUG

The hyphen shown above is DEBUG prompt.

5.2 DEBUG COMMANDS

DEBUG has a number of commands to facilitate users to write, execute and debug programs. Each command is designed to perform an important task. There are commands to assemble and execute programs, display the contents of registers and memories, to run a program in single step mode, etc.

A DEBUG command is represented by a one-letter symbol. For example: A for assemble, E for entry, D for display, etc.

5.2.1 Assemble Command, A

The A command allows you to enter mnemonics of an assembly language program and converts it into machine codes. When A command is executed, the machine codes are generated and directly stored in the memory. They are not displayed on the screen. To display machine codes on the screen the U command is used. It will be discussed in the next subsection.

After getting DEBUG prompt the starting address is initialized as - A address

Suppose we take 0100 as the starting offset address, then the above step will be written as

- A 0100

When the above command is executed the next line will show:

1584 : 0100

If offset is not provided, the DEBUG will take offset by default and it is 0100. The default address is CS : 0100. 1584 is the content of the code segment register CS. It may be some other number also. 0100 is the offset in the code segment. Now a mnemonic in this line and Enter key pressed as shown below:

1584 : 0100 MOV AX, 0012

The next line will show as:

1584 : 0103

In the above line you can enter the next mnemonic of your assembly language program. A simple program will be written as follows:

-A 0100

1584 : 0100 MOV AX, 0012
1584 : 0103 MOV BX, 1253
1584 : 0106 MOV CX, 2485
1584 : 0109 MOV DX, 1548
1584 : 010C MOV SI, 3412
1584 : 010F MOV DI, 7856
1584 : 0112 INT 3
1584 : 0113

At the memory location 1584 : 0113 the DEBUG is ready to accept another mnemonic. If you want to enter a mnemonic, you enter it and press Enter. It will go in the next line. If you do not want to enter any

mnemonic at this point, rather you want to terminate the A command and end your program, you just press Enter after 1584 : 0113. It will exit the A command and show the DEBUG prompt, a hyphen as shown in the above program.

One can change the instruction of a program at any memory address. For example, the instruction INT 3 at 1584 : 0112 can be changed to ADD AX, BX :

```
- A 0112  
1584 : 0112 ADD AX, BX  
1584 : 0114
```

5.2.2 Unassemble Command, U

The U command disassembles machine codes of specified memory addresses and gives corresponding mnemonics. It displays machine codes as well as corresponding mnemonics on the screen. Its default register is the CS : IP. Its general format is

U address range

We have already written a simple program and its machine codes are stored in the memory from 1584 : 0100 to 1584 : 0112. We can unassemble these machine codes as follows:

```
-U 0100 0112
```

This command will show the unassembled program as shown below:

```
-U 0100 0112  
1584:0100 B81200 MOV AX,0012  
1584:0103 BB5312 MOV BX,1253  
1584:0106 B98524 MOV CX,2485  
1584:0109 BA4815 MOV DX,1548  
1584:010C BE1234 MOV SI,3412  
1584:010F BF5678 MOV DI,7856  
1584:0112 CC INT 3
```

Adjacent machine codes (hex codes) are displayed in continuation without any space between them. 1584 is the content of CS. It comes by default.

The address range with U command can also be given in the following format

```
U 0100 L10
```

This command will unassemble 16 (10 hex) bytes starting from CS:0100.

If U command is given with only one address in stead of an address range, DEBUG unassembles 32 bytes starting from the given address.

```
.U 0100  
1584:0100 B81200 MOV AX,0012  
1584:0103 BB5312 MOV BX,1253  
1584:0106 B98524 MOV CX,2485  
1584:0109 BA4815 MOV DX,1548  
1584:010C BE1234 MOV SI,3412  
1584:010F BF5678 MOV DI,7856  
1584:0112 CC INT 3  
1584:0113 CC INT 3  
1584:0114 CC INT 3  
1584:0115 CC INT 3  
1584:0116 CC INT 3  
1584:0117 CC INT 3  
1584:0118 B209 MOV DL,09  
1584:011A 99 CWD  
1584:011B 007315 ADD [ BP+DI+15 ],DH  
1584:011E 7315 JNB 0135
```

If U command is given without specifying any address, DEBUG unassembles 32 bytes starting from the address contained in the IP register or it unassembles 32 bytes since last U, if any. The default register for U command is the CS:IP.

5.2.3 Register Command, R

The R command is used to display the contents of one or more registers. It also displays the status of the flags. The general format of the command is

R [register name]

If the R command is given without a name of a register, the DEBUG displays the contents of all registers.

```
.R  
AX=1254 BX=5938 CX=0000 DX=0012 SP=FFEE BP=0000 SI=0000 DI=0000  
DS=20A9 ES=20A9 SS=20A9 CS=20A9 IP=0100 NV UP EI PL NZ NA PO  
NC  
20A9:0100 B81200 MOV AX,0012
```

If R command is given with a name of a register, the name and the content of that register is displayed in the next line. A colon is then

displayed as a prompt in the subsequent line. Now a value after the colon is typed to change the register and then Enter key is pressed. Thereafter, the debug prompt is displayed. If you do not want to change the content of the register, press Enter key. The old value will remain in the register. To see the changes you have made, give R command or 'R register-name' command again.

```
-R AX
AX 1254
:2415
-R BX
BX 5938
:3498
-R IP
IP 0100
:0200
```

If you type an invalid register-name, an error message BR will be displayed.

Status Flags The status flags with their codes for SET and CLEAR are given in Table 5.1.

Table 5.1 Status Flags

Name of the flag	SET	CLEAR
Overflow	OV	NV
Direction	DN (Decrement)	UP (Increment)
Interrupt	EI (Enabled)	DI (Disabled)
Sign	NG (Negative)	PL (Positive)
Zero	ZR	NZ
Auxiliary Carry	AC	NA
Parity	PE (Even)	PO (Odd)
Carry	CY	NC

We have already seen that when R command is given without a register-name the status of flags along with the contents of all other registers are displayed. If we want to change a flag-status, we can give a command as follows:

```
-R F
```

F is register-name to see the flags. Now the status of the flags are displayed in the beginning of the next line in the form of codes as mentioned above. At the end of the list of status of the flags, the debug displays a hyphen (-). Now after the hyphen you can type changed status

of the desired flags in any order, and then press Enter key. The sequence of giving commands is as follows:

R F

It will show:

NV UP EI PL NZ NA PO NC -

Now enter desired status of the desired flags after hyphen as

NV UP EI PL NZ NA PO NC - OV CY DI

To see the changed condition give R F command again.

R F

It will show:

OV UP DI PL NZ NA PO CY -

Any flags for which new values are not given remain unchanged. If more than one value for a flag is given, debug returns an error message, DF. If an invalid flag-code is given, the debug returns a BF error message. In both the cases, the flags up to the error in the list are changed. The flags at and after the error remain unchanged.

5.2.4 Go Command, G

The G command is used to execute a program. Its general format is

G = address

The equal sign (=) put before the specified address indicates that the given address is the starting address. The default register for G command is the CS. We can take a program as shown below:

```
-A 0100
1584 : 0100 MOV AX, 0012
1584 : 0103 MOV BX, 1253
1584 : 0106 MOV CX, 2485
1584 : 0109 MOV DX, 1548
1584 : 010C MOV SI, 3412
1584 : 010F MOV DI, 7856
1584 : 0112 INT 3
1584 : 0113
```

To execute the program and to see the results, give the command as follows

G = 0100

The above command will execute the program starting from the address CS:0100, and show the contents of all register, flags, etc. as shown below:

```
-G = 0100
AX=0012 BX=1253 CX=2485 DX=1548 SP=FFFF BP=0000 SI=3412 DI=7856
DS=159C ES=159C SS=159C CS=159C IP=0112 NV UP EI PL NZ NA PO
NC
159C:0112 CC INT 3
-
-G
```

If G command is given without any address, it executes program starting from the address CS:IP.

Break-Point The G command is also used to provide a break-point in a program. The format is

G address

The above command executes the program starting from the offset address contained in the register IP. In the other words the starting address is CS:IP. The execution breaks (i.e. ends) at the given address. For example, we can give a command as follows:

```
-G 0109
```

This command will execute our program of Section 5.2.1 starting from the offset address [IP] up to the offset address 0108. With IP = 0100, the result is as follows:

```
-G 0109
```

```
AX=0012 BX=1253 CX=2485 DX=1548 SP=FFFE BP=0000 SI=0000 DI=0000
DS=159C ES=159C SS=159C CS=159C IP=0109 NV UP EI PL NZ NA PO NC
159C:0109 BA4815 MOV DX,1584
-
```

G = address address

If the above command is given it will execute program starting from the first address and provide break-point at the second address.

Example

```
-G = 0103 0109
```

The above command executes program from CS:0103 to CS:0108 and provides break-point at CS:0109.

5.2.5 Trace Command, T

The command T is used to run a program in single-step mode. The default register is the CS:IP and the general format is

T = address

For testing the above command we can write a simple program as shown below

```
-A 0100
1584:0100 MOV AX,0012
1584:0103 MOV BX,1253
1584:0106 MOV CX,2485
1584:0109 MOV DX,1584
1584:010C INT 3
1584:010D
```

To execute the above program in single-step mode the following command is given:

-T = 0100

The above command will execute the first instruction of the program and show the result as

-T = 0100

```
AX=0012 BX=0000 CX=0000 DX=0000 SP=FFFF BP=0000 SI=0000 DI=0000
DS=159C ES=159C SS=159C CS=159C IP=0103 NV UP EI PL NZ NA PO
NC
159C:0103 BB5312 MOV BX,1253
```

The last line of the result shows the next instruction which is to be executed. To execute the next instruction the T command is given in the following format:

-T

In the above command address of the instruction to be executed will be taken by default. The address which is in the register IP, i.e. 0103 will be taken by default. The results are as follows:

-T

```
AX=0012 BX=1253 CX=0000 DX=0000 SP=FFFF BP=0000 SI=0000 DI=0000
DS=159C ES=159C SS=159C CS=159C IP=0106 NV UP EI PL NZ NA PO
NC
159C:0106 B98524 MOV CX,2485
```

Using T = address command any instruction of the program can be executed. For example, if we want to execute the instruction at the memory location 1584:0106 MOV CX,2485 only, the following command will be given:

-T = 0106

Before giving this command the contents of the register AX, BX and CX have been set to zero. After execution of the above command the result is as follows:

```
-T = 0106
AX=0000 BX=0000 CX=2485 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=159C ES=159C SS=159C CS=159C IP=0109 NV UP EI PL NZ NA PO NC
20A9:0109 BA4815 MOV DX,1548
```

To execute more than one instruction by a single command in single-step mode the general format of the T command is

T = address value

Here value = n , where n is the number of instruction to be executed by single command in single-step mode.

To execute two instructions of the test program under consideration we can give command as follows:

```
-T = 0100 02
```

Before giving this command set the contents of AX, BX and CX equal to zero for testing the sample program.

The above command executes the first instruction MOV AX,0012 and prints its result. Thereafter, the second instruction MOV BX,1253 is executed and its result is printed. Results are as follows:

```
-T = 0100 02
```

```
AX=0012 BX=0000 CX=0000 DX=0000 SP=FFFF BP=0000 SI=0000 DI=0000
DS=159C ES=159C SS=159C CS=159C IP=0103 NV UP EI PL NZ NA PO NC
159C:0103 BB5312 MOV BX,1253
AX=0012 BX=1253 CX=0000 DX=0000 SP=FFFF BP=0000 SI=0000 DI=0000
DS=159C ES=159C SS=159C CS=159C IP=0106 NV UP EI PL NZ NA PO NC
159C:0106 B98524 MOV CX,2485
```

5.2.6 Display or Dump Command, D

The D command is used to display the contents of specified memory locations. The default register is DS. Its general format is

D address range or D address

With address range the D command is given as follows:

```
-D 0100 0120
```

The above command will show the contents of the memory locations from DS:0100 to DS:0120 as shown below:

```
-D 0100 0120
```

```

2000:0100 E0 23 C0 7E 2C 3D FE 00-7E 0C C7 46 E0 FE 00 B8
2000:0110 3B 00 50 E8 15 19 FF 76-E0 B8 20 00 50 C4 76 06
2000:0120 26

```

Each line displays 16 bytes. There is a hyphen between the eighth and ninth bytes.

The address range can also be given in the following format:
-D 0100 L 50

The above command will display 50 (hex) bytes beginning from the memory location DS:0100.

To see the contents of the memory locations of other segments, the command is given as follows:

-D CS:0100 0150

or

-D CS:0100 L 50

or

-1584:0100 L 50

or

-1800:0100 L 50, etc.

If the D command is given with starting address only, the DEBUG causes a default of 80 (hex) bytes. It displays 128 bytes (80 hex) starting from the given address. The command **D 0100** will display 128 bytes starting from the memory location DS:0100.

-D 0100

```

2000:0100 E0 23 C0 7E 2C 3D FE 00-7E 0C C7 46 E0 FE 00 B8
2000:0110 3B 00 50 E8 15 19 FF 76-E0 B8 20 00 50 C4 76 06
2000:0120 26 FF 74 02 26 FF 34 9A-0E 00 47 2B 83 C4 08 EB
2000:0130 12 83 7E E0 00 74 07 B8-3C 00 50 E8 ED 18 C7 46
2000:0140 E0 00 00 C4 76 0E 26 8B-04 26 FF 04 8B F0 D1 E6
2000:0150 D1 E6 C4 5E 06 26 8B 07-26 8B 57 02 C4 7E 0A 8B
2000:0160 DE 26 89 01 26 89 51 02-C4 76 06 26 8B 1C 26 8B
2000:0170 44 02 03 5E E0 26 89 1C-26 89 44 02 26 FF 04 8E

```

If D command is given without any address, the DEBUG will display 128 bytes after the last display.

5.2.7 Enter Command, E

The E command is used to enter data or machine codes. Its default register is the DS. It provides the following two options:

- (a) To enter data or machine codes sequentially.
- (b) To replace data or machine codes of certain memory locations.

The general format to enter sequential data or machine codes is

E address list

Suppose data are to be entered in DS segment starting from the offset address 0100. The command will be as

-E 0100

After writing the above command, data are entered in the same line giving space between adjacent data as shown below:

-E 0100 56 35 89 78 94

After entering the desired data Enter key is pressed. Now the DEBUG will show its prompt '-' in the next line. If data are to be entered in the next line also 'E address' command is to be given again. If only one digit of any data is typed, it is taken as a single digit data. Computer puts zero in the place of most significant position.

During data entry 24 bytes of data can be entered in a single line. Thereafter an equal (=) sign will come in the next line. Also a *bip* sound is heard. To get DEBUG prompt the return (Enter) key has to be pressed. If data entry is to be done in the next line also the command 'E address' has to be given again.

To Replace Data The command to replace data given is as follows:

-E address

Suppose we want to replace data in segment DS at offset 0100. The command to be given for this purpose is

-E 0100

This will show the contents of the memory location DS:0100 in the next line as

2000:0100 12.

2000 is the initial setting for DS, and it comes by default. 12 is the content of the memory location 2000:0100. If this data is to be changed enter the new data 26 as shown below:

2000:0100 12.26

If data of the next memory location is also to be changed, the space bar is to be pressed. It will show the content of the next memory location shown below:

2000:0100 12.26 85.

85 is the content of the memory location 0101. The new data 37 is now entered as shown:

2000:0100 12.26 85.37

In this way data replacement can be proceeded. The maximum number of bytes that can be entered in a line is 8. After replacing the desired number of data, enter key is pressed, then the debug command appears in the next line. If data replacement is continued up to the end of the line, the next line with current memory address comes automatically.

Use of Hyphen Key While replacing data if you decide to go back to the preceding byte position press hyphen key. A new line will be started. It will show the address and the content of previous byte. An example is:

```
-E 0100  
2000:0100 12.35 85.32 13.42
```

Now press hyphen after the entry of 42 as shown above.

or 2000:0100 12.35 85.32 13

Press hyphen before entering 42 as shown above a new line will be started as shown:

```
2000:0101 32.
```

Now you can enter the desired data instead of 32 in the memory address 0101.

The hyphen key does not work in sequential data entry.

While replacing data one can go to the back position using backspace key in the current byte. One cannot go to the preceding byte position by pressing backspace key. On the other hand in sequential entry one can go back to the preceding byte position using backspace key.

To Enter Machine Codes Machine codes (instruction codes) are entered in the CS segment. Suppose the starting offset is 0100. The command is given as follows:

```
-E CS:0100
```

Now machine codes (in hex) are entered in the same line as

```
-E CS:0100 B8 12 00 BB 53 12 CC
```

Now using 'U address range' command, the corresponding mnemonics along with machine codes can be seen.

To Replace Machine Codes To replace machine codes the same technique is used which has been used for replacing data. The command will be as follows:

```
-E CS:0100
```

The above command will show the contents of the memory location CS:0100 in the next line:

```
1584:0100 B8.
```

B8 is the present machine code in the memory location 1584:0100. To change it, write the new code after the decimal point in the above line.

The command can also be given as:

- E 1800:0100 for sequential entry.
- E 1600:0100 for code replacement.

In place of 1584 any other starting address may also be used.

To make changes in the preceding byte position while replacing machine codes you can use hyphen key.

5.2.8 Fill Command, F

This command is used to fill the specified range of memory locations with the values given in a list. The default segment register is the DS. The general format is

F address range list

An example is

-F 0100 0104 45 46 43 08 09

The above values from 45 to 09 will be filled in the memory locations DS:0100 0104. This can be checked using D command.

-D 0100 0104

2000:0100 45 46 43 08 09

2000 is the initially set value of DS.

The F command can also be given in the following format:

-F 0100 L 05 45 46 43 08 09

The above command fills 5 given values starting from the memory location DS:0100.

If the values are to be filled in the CS segment, the command is given as follows:

-F CS:0100 0106 B8 12 00 BB 53 12 CC

or -F CS:0100 L 07 B8 12 00 BB 53 12 CC

The command can also be given in the following forms:

-F 1800:0100 0105 1 2 3 4 5 6

The entry can be checked as

-D 1800:0100 0105

1800:0100 01 02 03 04 05 06

The command can also be given as

-F 1800:0100 L 06 01 02 03 04 05 06

If the number of data in the list is less than the number of bytes in the address range, the same list is used repeatedly until all bytes in the range are filled.

Example

```
-F 0100 010B 22 33 44 55 66
```

```
-D 0100 010B
```

```
2000:0100 22 33 44 55 66 22 33 44 55 66 22 33
```

If the number of data in the list is more than the number of bytes in the address range, the extra data are ignored.

Example

```
-F 0100 0104 22 33 44 55 66 77 88 99
```

```
-D 0100 0104
```

```
2000:0100 22 33 44 55 66
```

5.2.9 Move Command, M

The M command copies (moves) the block of data from one memory area into another. The default register is the DS. The general format is
M range address

An example of move command is

```
-M 0100 0105 0110
```

or -M 0100 L 06 0110

This command copies data from DS:0100 to DS:0105 into the address beginning from DS:0110 to DS:0115.

```
-M 0100 0105 0103
```

It is a case of overlapping. In such a case data are moved without any loss. The technique is to move data first from DS:0105 to DS:0108, then DS:0104 to DS:0107 and so on until it moves DS:0100 to DS:0103.

To move data in CS segment command is given as

```
-M CS:0100 0105 CS:0110
```

To move data from one segment to another, commands can be given:

```
-M CS:0100 0105 DS:0100
```

Commands can also be given in the following format:

```
-M 1566:0100 0105 2000:0100
```

5.2.10 Search Command, S

The search command is used to search the specified memory range for the specified list of bytes. The default register is the DS. The general format is
S range list

The list may contain one or more bytes. If the list contains only one byte, all addresses of the byte in the specified range are displayed. If the

list contains more than one byte, only the first addresses of the byte string are returned.

To Search a Byte To search a byte in a specified range the command may be given as follows:

S 0100 0105 02

Having received this command DEBUG searches the data, 02 in the memory range DS:0100 0105, and displays its address as follows:

2000:0105

This display means that 02 is at the memory location 2000:0105. Using D command the contents of the memory locations DS:0100 0105 were seen. They were as follows

-D 0100 0105

2000:0100 01 00 1F 03 E2 02

The search command can also be given in the following formats

S0100 L 06 02

S CS:0100 0105 05

S CS:0100 L 06 05

If the given byte is at more than one addresses, all addresses are displayed. DS:0100 0105 contains 01 02 02 05 07 02. The S command will display as follows:

-S 0100 0105 02

2000:0101

2000:0102

2000:0105

To Search a String of Bytes in the Specified Memory Range When S command is given to search a string (or group) of bytes within specified memory range, it displays only the first address of the string.

Example

- Suppose memory 2000:0100 contains 2 3 5 2 3 5 2 3 5 2 3 5

Search command is given to search the string of bytes 2, 3, 5 as follows:

-S 0100 010A 2 3 5

It displays:

2000:0100

2000:0103

2000:0106

2000:0109

These are starting addresses of the group of bytes 2, 3 & 5.

- (ii) Data are DS:0100 05 2 6 3 8 9 2 6 3 10 15
 Search command is given to search the group of bytes 2 6 3 as
 -S 0100 010A 2 6 3
 It displays as

2000:0101

2000:0106

5.2.11 To Save a File

Before saving a file, name is assigned to the file by N (name) command.
 Its format is

-N filename.COM

The name of the drive can also be given with N command as

-N A :filename.COM

A is for floppy drive, C for hard disk. If drive name is not given in the above command, the file is saved on hard disk.

After assigning a name to a file it can be saved using write command, W. Its format is

-W address

Before giving write command the number of bytes to be saved is to be given in the register BX:CX. The default register is the CS. If address is not given with the W command, the file is saved beginning from CS:0100.

For example,

-N BR1.COM BR1 is the name of a file

-R BX

BX 0000

:

-R CX

CX 0000

: 000A ,number of bytes of the file is 0A.

-W 0200

'Writing 000A bytes' will be shown on the screen.

The file BR1 will be saved beginning from CS:0200.

5.2.12 To Load a File

Before loading a file give name of the file using N command. Thereafter use L command to load a file.

For example,

-N A:B1.COM B1 is the name of a file

-L

The L command loads file into the memory beginning from CS:0100, and sets registers BX:CX to the number of bytes loaded.

5.2.13 Summary of DEBUG Commands

A (Assemble) converts mnemonics into machine codes. The machine codes are directly stored in the memory. The default address is CS:0100. Formats are:

A ; assembles beginning from CS:0100

A Address assembles starting from the given address

C (Compare) compares the contents of two specified blocks of memory. The default register is the DS. The general format is

C Range Address

Example

-C 0100 010A 0200

The above command compares the contents of memory locations DS:0100-DS:010A with the contents of memory locations starting from DS:0200. When contents of the two blocks are same, the debug prompt is shown in the next line.

When the contents of the two blocks of memory are not same, the addresses where contents are different are displayed.

Example

-C 0100 010A 0200

or -C 0100 L 010A 0200

2200:0105 06 15 2200:0205

2000:0107 08 13 2200:0207

In this case content at DS:0105 was 06, whereas at DS:0205 was 15. This has been displayed in a line. Similarly the content at 2200:0107 and 2200:0207 were 08 and 13 respectively. This has been displayed above in the last line.

D (Display or dump) displays the contents of specified memory locations. The default register is the DS:IP. Formats are:

D address range

or **D address**

or **D**

Examples are:

- D 0100 0120 displays the contents of the memory locations from DS:0100 to DS:0120.
- D 0100 L 50 displays 50 (hex) bytes starting from the memory location DS:0100.
- D 0100 displays 128 (80 hex) bytes starting from DS:0100.
- D displays 128 bytes after the last display.
- D CS:0100 0120 displays memory contents from CS:0100 to CS:0120.
- D 1600:0100 0150 displays from 1600:0100 to 1600:0150.

E (Enter) To enter data or machine codes. Default register is the DS:IP.
Formats are:

E address list

E address

Examples are:

- E 0100 56 35 89 78 94 Enter data starting from DS:0100
- E 0100 Replace data of DS:0100
- 2000:0100 12
- E CS:0100 B8 12 00 BB 53 12 CC Enter machine codes starting from CS:0100.
- E CS:0100 Replace code of 1584:0100
- 1584:0100 B8.
- E 1800:0100 Codes or data for sequential entry.
- E 1800:0100 For code or data replacement.

F (Fill) fills a range of memory locations with the values in the list. The default register is the DS. Formats are:

F address range list

G (Go) Executes program. Provides break-point. Default register is the CS:IP. Formats are:

G = address

Example are

- G = 0100 Executes program from CS:0100.
- G Executes program from CS:IP.
- G address** Provides break-point.
- G 0109 Executes program from CS:IP to CS:0109.

H (Hexadecimal) shows the sum and difference of two hexadecimal numbers. The maximum length of a number is of four hex digits.

(i) -H 8 5

(Result) 000D 0003

(sum) (difference)

(ii) -H 8 9

0011 FFFF (2's complement of 1)

I (Input). Inputs and displays one byte from a port. Format are:

I port address

L (Load) It is used to load a file or program. Its default register is CS. Before loading C file its name is given using N command. Then L command is given to load the file. After loading a file the DEBUG sets registers BX:CX to the number of bytes loaded.

M (Move) Copies (moves) the block of data from one memory area into another. The default register is the DS. The general format is

M range address

Examples:

-M 0100 0105 0110

Copies bytes from DS:0100 0105 into DS:0110 0115.

-M 0100 0105 0103

Copies bytes from DS:0100-DS:0105 into the address beginning from DS:0103. This is a case of overlapping.

-M CS:0100 0105 CS:0110

It is an example of moving data in CS segment.

-M CS:0100 0105 DS:0100

This is an example of moving data from CS segment to DS segment.

-M 1566:0100 0105 2000:0100

Command can also be given in this format.

N (Name) It is used to name a program or file to be read or written onto disk. The format is

-N filename.COM

When drive name is not given in the command, the file will be saved on the hard disk.

To save a file on floppy disk the command is

-N A :filename.COM

A is the name of the floppy disk drive, and C is that of the hard disk.

After assigning a name the file is saved or read using L (load) or W (write) command.

O (Output) Sends a byte to a port. Format is

O port address byte

Q (Quit) It is used to exit DEBUG. Terminates the DEBUG utility and returns to the MS-DOS without saving the file you are currently working with. The format is

-Q

R (Register) Displays the contents of one or more CPU registers, and the next instruction. Format is

-R register name

If register name is not given, it displays the contents of all registers and flags.

S (Search) Searches the specified memory range for the specified list of bytes. The default register is the DS. Format is

S range list

Examples:

-S 0100 0105 02 Searches 02 in the memory range DS:0100 0105. Displays the memory locations where 02 is stored.

or **-S 0100 L 06 02**

-S CS:0100 0105 05 Searches in CS segment.

-S CS:0100 L 06 05 Searches in CS segment.

-S 0100 010A 2 6 3 Searches the string of bytes 2, 6 and 3 in the memory range DS:0100 010A. Displays only the first address of string.

T (Trace) Executes a program in a single-step mode. After executing one instruction it displays the contents of all registers and flags. The default register is the CS:IP. The format is

T = address

Examples are:

-T = 0100 Executes instruction at the address CS:0100.

-T Executes instruction at the address CS:IP

-T = 0100 02 Executes two instructions in single-step mode. After executing each instruction, it shows registers & flags.

U (Unassemble) Unassembles machine code of specified memory range. Displays machine codes and corresponding mnemonics with address. The default address is CS:IP. General format is

U address range or U address

Examples are:

-U 0100 0112 Unassembles from CS:0100 to CS:0112.

-U 0100 Unassembles 32 bytes starting from CS:0100.

-U Unassembles 32 bytes starting from CS:IP, or Unassembles 32 bytes since last U, if any.

W (Write) It is used to save a file or program. Default register is the CS. Format is

W address

If address is not given the file is saved starting from CS:0100. Before saving a file its name is to be given using N command. The number of bytes to be saved is to be given in BX:CX. See example in the Section 5.2.11.

5.3 ASSEMBLER

An Assembler is a program which translates assembly language program into machine language program or object program. A **self-assembler** or **resident assembler** is an assembler which runs on a computer for which it produces object codes. A **cross-assembler** is an assembler which runs on a computer other than that for which it produces object codes. Some microprocessor-based systems are not provided with an assembler. For such a system cross-assembler is required. The programmer writes an assembly language program and obtains machine codes using other computer provided with a cross-assembler. Machine codes obtained from the cross-assembler are then loaded into the memory of the microprocessor-based system and the program is executed. Also, when a designer does not have an assembler for a microprocessor-based system being developed, a cross-assembler is required.

A **macro-assembler** is an assembler which allows user to define sequence of instructions as macro.

A **meta assembler** is an assembler which can handle different instruction sets. The programmer must define a particular instruction set which is to be used.

Disassembler A disassembler generates an assembly language program corresponding to an object program. Its role is complementary to an assembler. Sometimes an object program is available, and the programmer wants to understand the program. In such a situation the programmer needs a software (disassembler) to convert the object program to an assembly language program.)

One-Pass Assembler An assembler which goes through an assembly language program only once and produces its object codes is known as one-pass assembler. Such an assembler must have some means to take forward references into consideration. In assembly language programs labels are used to point forward references. The one-pass assembler must have some ways to assign addresses to the labels.

Two-Pass Assembler An assembler which goes through an assembly language program twice to produce object program (machine code program) is known as two-pass assembler. Such an assembler doesn't