Experiment #1

MS-DOS Debugger (DEBUG)

1.0 Objectives:

The objective of this experiment is to introduce the "DEBUG" program that comes with MS-DOS and Windows operating systems. This program is a basic tool to write, edit and execute assembly language programs.

In this experiment, you will learn DEBUG commands to do the following:

- Examine and modify the contents of internal registers
- Examine and modify the contents of memory
- Load, and execute an assembly language program

1.1 Introduction:

DEBUG program which is supplied with both DOS and Windows, is the perfect tool for writing short programs and getting acquainted with the Intel 8086 microprocessor. It displays the contents of memory and lets you view registers and variables as they change. You can use DEBUG to test assembler instructions, try out new programming ideas, or to carefully step through your program. You can step through the program one line at a time (called *tracing*), making it easier to find logic errors.

1.2 Debugging Functions

Some of the basic functions that the debugger can perform are the following:

- Assemble short programs
- View a program's source code along with its machine code
- View the CPU registers and flags (See Table 1 below)
- Trace or execute a program, watching variables for changes
- Enter new values into memory
- Search for binary or ASCII values in memory
- Move a block of memory from one location to another
- Fill a block of memory
- Load and write disk files and sectors

The following table shows a list of some commonly used DEBUG commands.

COMMAND	SYNTAX	FUNCTION	EXAMPLE
Register	**R** [Register Name]	Examine or modify the contents of an internal register of the CPU	-**R** AX (AX reg.) -**R** F (flags)
Dump	**D** [Address]	Display the contents of memory locations specified by Address	-**D** DS:100 200 -**D** start-add end-add
Enter	**E** [Register Name]	Enter or modify the contents of the specified memory locations	-**E** DS:100 22 33 -**E** address data data
Fill	**F** [Register name]	Fill a block of memory with data	-**F** DS:100 120 22
Assemble	**A** [Starting address]	Convert assembly lang. instructions into machine code and store in memory	-**A** CS:100 -**A** start-address
Un-assemble	**U** [Starting Address]	Display the assembly instructions and its equivalent machine codes	-**U** CS:100 105 -**U** start-add end-add
Trace	**T** [Address][Number]	Line by line execution of specific number of assembly lang. instructions	-**T**=CS:100 -**T**=starting-address
Go	**G** [Starting Address] [Breakpoint Add.]	Execution of assembly language instructions until Breakpoint address	-**G**=CS:100 117 -**G**=start-add end-add

Table 1: DEBUG commands

The Internal Registers and Status Flags of the 8086 uP are shown in the following tables.

Flag	Meaning	SET	RESET
CF	Carry	CY	NC
PF	Parity	PE	PO
AF	Auxiliary	AC	NA
ZF	Zero	ZR	NZ

Flag	Meaning	SET	RESET
SF	Sign	NG	PL
IF	Interrupt	EI	DI
DF	Direction	DN	UP
OF	Overflow	OV	NV

AX	**BX**	**CX**	**DX**	**SI**	**DI**	**SP**	**BP**
DS	**CS**	**ES**	**SS**	**IP**	**8086 Internal Registers**		

Table 2: Internal Registers and Status Flags

1.3 Pre-lab:

1. Name a few computer operating systems. Which operating system do you mostly use?
2. What is the full form for MS-DOS?
3. What is the difference between a logical address and a physical address? Show how a physical address is generated from a logical address.
4. What are the following registers used for: DS, CS, SS, SP, IP, AX
5. Define the function each of the following flag bits in the flag register: Overflow, Carry, Sign, and Zero.

1.4 Lab Work:

A. Loading the DEBUG program

1. Load the DEBUG program by typing ***debug*** at the MS-DOS prompt, as shown in the example below:

```
C:\WINDOWS>debug
```

2. You will see a dash (-) in the left-most column on the screen. This is the DEBUG prompt.
3. Type a (?) to see a list of available commands.
4. Return to MS-DOS by entering Q. What prompt do you see?

Note: You have to hit Carriage Return (CR) key (or ENTER key) on the keyboard after you type any **debug** command.

B. Examining and modifying the contents of the 8086's internal registers

1. Use the REGISTER command to display the current contents of all the internal registers by typing R.
    - List the values of the following registers:

AX		**SP**	
BX		**CS**	
CX		**DS**	
DX		**SS**	
IP		**ES**	

    - What is the address of the next instruction to be executed?
    - What is the instruction held at this address?
2. Enter the command: R AH (hit <CR>)

   What happens and why?
3. Use a REGISTER command to first display the current contents of BX and then change this value to 0020h.

4. Use a REGISTER command to first display the current contents of IP and then change this value to 0200h.
5. Use a REGISTER command to first display the current contents of the flag register and then ***set*** the parity, zero, and carry flags.
6. Redisplay the contents of all the internal registers. Compare the displayed register contents with those observed in step 1 above. What instruction is now pointed by CS: IP?

C. Examining and modifying the contents of memory

1. Use the DUMP command (D) to display the first 100 bytes of the current data segment.
2. Use the DUMP command (D) to display the first 100 bytes of the code segment starting the current value of CS: IP.
3. Use the ENTER command (E) to load locations CS:100, CS:102, and CS:104 with 11, 22, and 33, respectively.
4. Use the ENTER command (E) to load five consecutive byte-wide memory locations starting at CS:105 with data 'FF'.
5. Verify the result of steps 3 and 4 using the DUMP command.
6. Use the FILL command (F) to initialize the 16 storage locations starting at DS:10 with the value AAh, the 16 storage locations starting at address DS:30 with BBh, the 16 storage locations starting at address DS:50 with CCh, and the 16 storage locations starting at address DS:70 with DDh
7. Verify the result of step 6 using the DUMP command.

D. Coding instructions in 8086 machine language

1. Enter each of the following instructions starting at address CS:100 one-by-one using the ASSEMBLE command (A).

MOV AX,BX
MOV AX, AAAAh
MOV AX,[BX]
MOV AX,[0004H]
MOV AX,[BX+SI]
MOV AX,[SI+4H]
MOV AX,[BX+SI+4H]

2. Using the UNASSEMBLE command (U), obtain

   a. the machine code of each of the instructions in step 1

   b. the number of bytes required to store each of the machine code instructions in step 1.

   c. the starting address of each instruction.

Instruction	Machine Code	Bytes required	Starting Address
MOV AX, BX			
MOV AX, AAAAH			
MOV AX,[BX]			
MOV AX,[0004H]			
MOV AX,[BX+SI]			
MOV AX,[SI+4H]			
MOV AX,[BX+SI+4H]			

   d. Why are the starting addresses of the above instructions not consecutive?

E. Coding instructions in 8086 machine language

1. Using the ASSEMBLE command (A), load the program shown below into memory starting at address CS: 0100.

```
                MOV    SI, 0100H
                MOV    DI, 0200H
                MOV    CX, 010H
BACK:           MOV    AH, [SI]
                MOV    [DI], AH
                INC SI
                INC DI
                DEC    CX
                JNZ    BACK
```

2. Verify the loading of the program by displaying it with the UNASSEMBLE (U) command.
    a. How many bytes of memory does the program take up?
    b. What is the machine code for the DEC CX instruction?
    c. What is the address offset for the label BACK?
3. Fill 16 bytes of memory locations starting at DS: 0200 with value 45H and verify.
4. Execute the above program one instruction at a time using the TRACE command (T). Observe how the values change for registers: AX, CX, SI, DI flag register, and IP.
5. Run the complete program by issuing a single GO command (G).
    a. What is the starting address for this command?
    b. What is the ending address for this command?
6. What are the final values of registers: AX, CX, SI, and DI?
7. Describe the function of the above program.

F. Music Program

This program generates a musical tone every time a key pressed. It generates 8 tones in total and then stops.

1. Using the ASSEMBLE command (A), load the program shown below into memory starting at address CS: 0100.

```
            LEA SI, TUNE
            CLD
BACK:        MOV AH, 0
            INT 16H
            LODSW
            MOV BX, AX
            CMP AX, 0
            JZ L1
            MOV AL, 0B6H
            OUT 43H, AL
            MOV AL, BL
            OUT 42H, AL
            MOV AL, BH
            OUT 42H, AL
            IN AL, 61H
            OR AL, 3
            OUT 61H, AL
            JMP BACK

L1:         IN AL, 61H
            AND AL, 0FCH
            OUT 61H, AL
            INT 20H

TUNE:
            DW 11D1H
            DW 0FDFH
            DW 0E24H
            DW 0D59H
            DW 0BE4H
            DW 0A98H
            DW 0970H
            DW 08E9H
            DW 0000
```

2. Verify the loading of the program by displaying it with the UNASSEMBLE (U) command.
3. Run the complete program by issuing a single GO command (G).
   a. What is the starting address for this command?
   b. What is the ending address for this command?