**Experiment No. 07**

**Title: Write basic 8086 programs**

**Batch: Roll No.: Experiment No.: 07**

|  |  |
| --- | --- |
| **Aim: Write the following 8086 assembly programs:-**   1. Print “Hello World” 2. Convert an 8 bit binary number to decimal and display it. |  |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_ \_**

**Resources needed:** DOSBOX, TASM

**Theory:**

The Intel 8086 is an 16 bit microprocessor designed in the mid-80s. It has a CISC instruction set. i.e. instructions of different categories take different number of clock cycles for execution. The assembly language programs shown here are an example of how assembly code looks like. Some of the statements are assembler directives, similar to compiler pre-processor directives.

DOSBOX is a free and open source DOS emulator. The DOS operating system was soon replaced by the Windows OS . DOS is no longer used or supported, but for instructional purposes it is still needed in academic institutions. Therefore the DOS emulator DOSBOX is used to run all legacy DOS programs, games etc. We will use the old TASM (Turbo Assembler) developed by Borland.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Program1: Print “Hello World”**

; -------------------------------------------------------

; Program: Display the message "Hello World"

; Filename: Prog1.asm

; -------------------------------------------------------

.model tiny ; com program

.stack 100h

.data

message db "Hello World$",'$' ; Message to be displayed terminating with a $

.code ; code segment

org 100h ; code starts at offset 100h

start:

mov ax, @data

mov ds, ax

call method1

call crlf

call method2

mov ah,4ch ; function to terminate

int 21h ; Dos Interrupt

method1 proc near

mov ah,09h ; function to display a string

lea dx, message ; offset ofMessage string terminating with $

int 21h ; dos interrupt

ret

method1 endp

method2 proc near

lea si, message ; Make Si point to string address

loop1:

mov dl, ds:[si] ; Read First Character

cmp dl,'$'

je over ; if not '$' jump to loop1

mov ah, 02h

int 21h

inc si ; Point to next character

jmp loop1

over:

ret

method2 endp

crlf proc near

mov dl, 0ah

mov ah, 02h

int 21h

mov dl, 0dh

mov ah, 02h

int 21h

ret

crlf endp

end start

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Program2: Convert an 8 bit binary number to decimal and display it.**

; ----------------------------------------

; Program: Convert 8 bit binary number to decimal and display the result

; FileName: Prog2.asm

; ----------------------------------------

.model tiny ; com program

.stack 100h

.data

message db "Hello World$",'$' ; Message to be displayed terminating with a $

bin8 db 01100100b

ans db 0,0,0,'$'

hund db 100

ten db 10

.code ; code segment

org 100h ; code starts at offset 100h

start:

mov ax, @data

mov ds, ax

convert:

mov ah, 00

mov al, bin8

div hund

lea si, ans

mov ds:[si], al

mov al, ah

mov ah, 0

div ten

inc si

mov ds:[si], al

inc si

mov ds:[si], ah

ascii:

lea si, ans

mov al, 30h

add ds:[si], al

add ds:[si+1], al

add ds:[si+2], al

disp\_ans:

mov ah,09h ; function to display a string

lea dx, ans ; offset ofMessage string terminating with $

int 21h ; dos interrupt

over:

mov ah,4ch ; function to terminate

int 21h ; Dos Interrupt

end start

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Procedure**:

1. Download and install DOSBox and TASM in C drive
2. Type the programs given in this file and save them as .asm files.
3. Run DosBox
4. In the DosBox window, compile and run the program:

> tasm /l prog1.asm

> tlink prog1.obj

> prog1.exe

1. Verify the desired output

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Observations and Results:**

Both the programs should display the desired outputs in the DOSBOX window.

Paste the snapshots here.



**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Outcomes: CO4:**Understand the basic concepts of memory elements.

**CO5:**Understand the fundamental concepts of microprocessors.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Conclusion:**

We came to know about ASM

**Grade: AA / AB / BB / BC / CC / CD /DD**

**Signature of faculty in-charge with date**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**References:**

**Books/ Journals/ Websites:**

1. “Microprocessors and Interfacing”, by Douglas Hall Tata McGraw Hill 3rd edition.