Experiment-1

Aim: Introduction to MASM/TASM/Debugger

Experimental Requirements: PC loaded with TASM software, MASM Software

Procedure for doing **MASM** program:   
Step1:  
Open the dosbox icon placed in the desktop  
Step 2:  
type the following  
mount c c:\8086  
and press enter  
then dos box will be mounted to the local directory.  
Step 3:  
type c:  
and press enter  
You will be getting the screen as:  
c:\  
Step4:  
Now Type  
Edit  
A new window is opened now.  
Type the program in the window. Save it as "Filename.ASM" and save. Note : Give your own Filename.  
Step5:  
Check for errors and warnings. using MASM Filename.ASM  
, and press enter  
If errors and warnings correct them by going to the edit . Repeat this process until there are 0 errors and 0 warnings. When you get 0 errors and 0 warnings, Filename.obj is created.  
Step6: When they are 0 errors and 0 warnings, type the following:  
LINK Filename.obj  
, and press enter  
Now Filename.exe is created.  
Step7:  
Now type:  
DEBUG Filename.exe  
, and press enter  
Now you are going to get hyphen symbol  
-  
Step 8:  
type t  
(Single Step Execution)  
Till the completion of the program you need to repeat.

Procedure for doing **DEBUG** program:  
Step1:  
Open the dosbox icon placed in the desktop  
Step 2:  
type the following  
mount c c:\8086  
and press enter  
then dos box will be mounted to the local directory.  
Step 3:  
type c:  
and press enter  
You will be getting the screen as:  
c:\  
Step4:  
Now Type  
debug  
,and press enter  
Now you are going to get hyphen symbol  
Step5: Type  
a ,and press enter  
Type your program as shown in the attachment.  
Step6:  
Type  
R IP  
,and press enter  
The instruction pointer should point to the starting address of the program.  
If not , type the starting address of the program.For Example, 0100  
Step7:  
We need to do single step execution.  
For single step execution:  
type  
t as shown in the attachment.  
Till in the end of the program we need to repeat.

Procedure for **TASM**:

1. Switch on the PC, press windows+R then enter CMD.
2. Find the folder where TASM is located. check whether TASM.EXE, TLINK.EXE, TD.EXE are present or not
3. Enter into the directory where TASM is located by using cd... or directory name:
4. Type cd tasm in which the three files are present .Now we will be getting into c: \ or d:\ with tasm directory.
5. Type edit then a new window will be opened in which the program is entered.
6. After entering the program save the file with <filename.asm>.
7. Check for the errors or warnings by using TASM <filename> and press enter...
8. If there are no errors, then type TLINK <filename> to compile the file. If errors go back to the edit and do the necessary corrections and repeat the previous step.
9. Next type td <filename > to debug the executable file then will be getting the message program has no symbol table, press ok and then write down the instructions, registers and flags status before execution .
10. For step by step execution press F8.and for direct execution press F9 and then write down the instructions, registers and flags status after execution .Go to dump if required for noting down the required inputs and outputs.

Program:

**8-Bit Arithmetic operations using MASM**

1. Addition of 8-bit numbers :

.MODEL SMALL

.STACK

.DATA

IN1 DB 23H

IN2 DB 15H

RES DB 01H DUP(?)

.CODE

MOV AX,@DATA

MOV DS,AX

MOV AL,IN1

MOV BL,IN2

ADD AL,BL

MOV RES,AL

INT 21H

END

2. Subtraction of 8-bit numbers

.MODEL SMALL

.STACK

.DATA

IN1 DB 23H

IN2 DB 15H

RES DB 01H DUP(?)

.CODE

MOV AX,@DATA

MOV DS,AX

MOV AL,IN1

MOV BL,IN2

SUB AL,BL

MOV RES,AL

INT 21H

END

3. Multiplication of 8-bit numbers

.MODEL SMALL

.STACK

.DATA

IN1 DB 07H

IN2 DB 03H

RES DW 01H DUP(?)

.CODE

MOV AX,@DATA

MOV DS,AX

MOV AL,IN1

MOV BL,IN2

MOV AH,00H

MUL BL

MOV RES,AX

INT 21H

END

4. Division of 8-bit numbers

.MODEL SMALL

.STACK

.DATA

IN1 DB 15H

IN2 DB 05H

RES DW 01H DUP(?)

.CODE

MOV AX,@DATA

MOV DS,AX

MOV AL,IN1

MOV BL,IN2

MOV AH,00H

DIV BL

MOV RES,AX

INT 21H

END

**Introduction to DEBUG**

1.Addition

-A

072A:0100

-A 400

072A:400 MOV AL,55

072A:4002 MOV BL,32

072A:4004 ADD AL,BL

072A:4006

-R IP

IP 0100

-R IP 4000

-T

-G

2.Subraction :

-A

072A:0100

-A 4000

072A:4000 MOV AL,33

072A:4002 MOV BL,36

072A:4004 SUB AL,BL

072A:4006

-R IP

IP 0100

-R IP 4000

-T

-G

3.Multiplication:

-A

072A:0100

-A 400

072A:400 MOV AL,54

072A:4002 MOV BL,21

072A:4004 MUL,BL

072A:4006 INT 03

-R IP

IP 0100

-R IP 4000

-T

-G

4. Divison

-A

072A:0100

-A 400

072A:400 MOV AL,24

072A:4002 MOV BL,4

072A:4004 DIV,BL

072A:4006

-R IP

IP 0100

-R IP 4000

-T

-G

**TASM Programs**

1.**ADDITION**

 ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DB 78H

OPR2 DB 23H

RES DB 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AL, OPR1

MOV BL, OPR2

ADD AL, BL

MOV RES, AL

INT 03H

CODE ENDS

END START

END

2. **SUBTRACTION:**

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DB 36H

OPR2 DB 23H

RES DB 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AL, OPR1

MOV BL, OPR2

SUB AL, BL

MOV RES, AL

INT 03H

CODE ENDS

END START

END

3. **MULTIPLICATION:**

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DB 15H

OPR2 DB 05H

RES DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AL, OPR1

MOV BL, OPR2

MOV AH,00H

MUL BL

MOV RES, AX

INT 03H

CODE ENDS

END START

END

4. **DIVISION:**

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DB 20H

OPR2 DB 05H

RES DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AL, OPR1

MOV BL, OPR2

MOV AH,00H

DIV BL

MOV RES, AX

INT 03H

CODE ENDS

END START

END

Result : MASM/TASM/Debugger Programs Performed.

Experiment -2

AIM: Arithmetic operation – Multi byte Addition and Subtraction, Multiplication and Division –Signed and unsigned Arithmetic operation, ASCII – arithmetic operation.

Experimental Requirements: PC loaded with TASM software

Procedure:

1. Switch on the PC, press windows+R then enter CMD.
2. Find the folder where TASM is located. check whether TASM.EXE, TLINK.EXE, TD.EXE are present or not
3. Enter into the directory where TASM is located by using cd... or directory name:
4. Type cd tasm in which the three files are present .Now we will be getting into c: \ or d:\ with tasm directory.
5. Type edit then a new window will be opened in which the program is entered.
6. After entering the program save the file with <filename.asm>.
7. Check for the errors or warnings by using TASM <filename> and press enter...
8. If there are no errors, then type TLINK <filename> to compile the file. If errors go back to the edit and do the necessary corrections and repeat the previous step.
9. Next type td <filename > to debug the executable file then will be getting the message program has no symbol table, press ok and then write down the instructions, registers and flags status before execution .
10. For step by step execution press F8.and for direct execution press F9 and then write down the instructions, registers and flags status after execution .Go to dump if required for noting down the required inputs and outputs.

Programs:

16-BIT UNSIGNED OPERATIONS

**1. ADDITION:**

 ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DW 78BCH

OPR2 DW 23FEH

RES DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AX, OPR1

MOV BX, OPR2

ADD AX, BX

MOV RES, AX

INT 03H

CODE ENDS

END START

END

2. **SUBTRACTION:**

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DW 36BBH

OPR2 DW 23CCH

RES DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AX, OPR1

MOV BX, OPR2

SUB AX, BX

MOV RES, AX

INT 03H

CODE ENDS

END START

END

3. **MULTIPLICATION:**

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DW 1506H

OPR2 DW 0AC05H

RES1 DW 1 DUP (0H)

RES2 DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AX, OPR1

MOV BX, OPR2

MOV DX,0000H

MUL BX

MOV RES1, AX

MOV RES2,DX

INT 03H

CODE ENDS

END START

END

4. **DIVISION:**

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DW 0F506H

OPR2 DW 0AC50H

RES1 DW 1 DUP (0H)

RES2 DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AX, OPR1

MOV BX, OPR2

MOV DX,0000H

DIV BX

MOV RES1, AX

MOV RES2,DX

INT 03H

CODE ENDS

END START

END

16 BIT SIGNED ARITHMETIC OPERATIONS

1. **ADDITION:**

 ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DW 0BCDEH

OPR2 DW 0ABCDH

RES DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AX, OPR1

MOV BX, OPR2

STC

ADD AX, BX/ ADC AX, BX

MOV RES, AX

INT 03H

CODE ENDS

END START

END

2. **SUBTRACTION:**

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DW 0BCDEH

OPR2 DW 0ABCDH

RES DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AX, OPR1

MOV BX, OPR2

STC

SUB AX, BX / SBB AX,BX

MOV RES, AX

INT 03H

CODE ENDS

END START

END

3. **MULTIPLICATION:**

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DW 1356H

OPR2 DW 5687H

RES1 DW 1 DUP (0H)

RES2 DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AX, OPR1

MOV BX, OPR2

MOV DX,0000H

MUL BX/ IMUL BX

MOV RES1, AX

MOV RES2, DX

INT 03H

CODE ENDS

END START

END

4. **DIVISION:**

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DW 5687H

OPR2 DW 1356H

RES1 DW 1 DUP (0H)

RES2 DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AX, OPR1

MOV BX, OPR2

MOV DX,00H

DIV BX/ IDIV BX

MOV RES1, AX

MOV RES2, DX

INT 03H

CODE ENDS

END START

END

**ASCII OPERATIONS**

1. AAA:

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AL, 35H

MOV BL,39H

MOV AH,00H

ADD AL,BL

AAA

INT 03H

CODE ENDS

END START

END

2. AAS:

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AL, 39H

MOV BL,35H

MOV AH,00H

SUB AL,BL

AAS

INT 03H

CODE ENDS

END START

END

3. AAM:

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AL, 05H

MOV BL,09H

MOV AH,00H

MUL BL

AAM

INT 03H

CODE ENDS

END START

END

4. AAD:

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AL, 05H

MOV BL,06H

MOV AH,03H

AAD

DIV BL

INT 03H

CODE ENDS

END START

END

1.MULTI BYTE ADDITION

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DB 0BDH, 0EDH, 56H, 0DEH

OPR2 DB 0BEH, 0FEH, 76H, 0EDH

RES DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START: MOV AX, DATA

MOV DS, AX

MOV SI, OFFSET OPR1

MOV DI, OFFSET OPR2

MOV BX, OFFSET RES

MOV CX, 0004H

MOV AH, 00H

BACK: MOV AL, [SI]

MOV DL, [DI]

ADC AL, DL

MOV [BX], AL

INC SI

INC DI

INC BX

LOOP BACK

INT 03H

CODE ENDS

END START

END

2. MULTI BYTE SUBTRACTION

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DB 0BDH, 0EDH, 56H, 0DEH

OPR2 DB 0BEH, 0FEH, 76H, 0EDH

RES DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:MOV AX, DATA

MOV DS, AX

MOV SI, OFFSET OPR1

MOV DI, OFFSET OPR2

MOV BX, OFFSET RES

MOV CX, 0004H

MOV AH, 00H

BACK: MOV AL, [SI]

MOV DL, [DI]

SBB AL, DL

MOV [BX], AL

INC SI

INC DI

INC BX

LOOP BACK

INT 03H

CODE ENDS

END START

END

3. MULTI BYTE MULTIPLICATION

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DB 0BDH, 0EDH, 56H, 0DEH

OPR2 DB 0BEH, 0FEH, 76H, 0EDH

RES DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START: MOV AX, DATA

MOV DS, AX

MOV SI, OFFSET OPR1

MOV DI, OFFSET OPR2

MOV BX, OFFSET RES

MOV CX, 0004H

MOV AH, 00H

BACK: MOV AL, [SI]

MOV DL, [DI]

MUL DL

MOV [BX], AX

INC SI

INC DI

INC BX

INC BX

LOOP BACK

INT 03H

CODE ENDS

END START

END

4. MULTI BYTE DIVISION

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

OPR1 DB 0BEH, 0FEH, 76H, 0EDH

OPR2 DB 0BDH, 0EDH, 56H, 0DEH

RES DW 1 DUP (0H)

DATA ENDS

CODE SEGMENT

START:MOV AX, DATA

MOV DS, AX

MOV SI, OFFSET OPR1

MOV DI, OFFSET OPR2

MOV BX, OFFSET RES

MOV CX, 0004H

MOV AH, 00H

BACK: MOV AL, [SI]

MOV DL, [DI]

DIV DL

MOV [BX], AX

INC SI

INC DI

INC BX

INC BX

LOOP BACK

INT 03H

CODE ENDS

END START

END

Result : Arithmetic operation – Multi byte Addition and Subtraction, Multiplication and Division –Signed and unsigned Arithmetic operation, ASCII – arithmetic operation performed.

**Experiment-3**

**Aim:** Logic operations – Shift and rotate – Converting packed BCD to unpacked BCD, BCD to ASCII conversion.

Experimental Requirements : PC loaded with TASM software

Procedure:

1. Switch on the PC, press windows+R then enter CMD.
2. Find the folder where TASM is located. check whether TASM.EXE, TLINK.EXE, TD.EXE are present or not
3. Enter into the directory where TASM is located by using cd... or directory name:
4. Type cd tasm in which the three files are present .Now we will be getting into c: \ or d:\ with tasm directory.
5. Type edit then a new window will be opened in which the program is entered.
6. After entering the program save the file with <filename.asm>.
7. Check for the errors or warnings by using TASM <filename> and press enter...
8. If there are no errors, then type TLINK <filename> to compile the file. If errors go back to the edit and do the necessary corrections and repeat the previous step.
9. Next type td <filename > to debug the executable file then will be getting the message program has no symbol table, press ok and then write down the instructions, registers and flags status before execution .
10. For step by step execution press F8.and for direct execution press F9 and then write down the instructions, registers and flags status after execution .Go to dump if required for noting down the required inputs and outputs.

**Logical Instructions:**

1.AND:

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AX, 3355H

MOV BX, 5355H

AND AX, BX

INT 03H

CODE ENDS

END START

END

2. **OR:**

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AX, 3355H

MOV BX, 5355H

OR AX, BX

INT 03H

CODE ENDS

END START

END

3. **NOT:**

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AX, 3355H

NOT AX

INT 03H

CODE ENDS

END START

END

4. XOR:

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AX, 3355H

MOV BX, 5355H

XOR AX, BX

INT 03H

CODE ENDS

END START

END

Shift and Rotate Instructions

1. **SHR:**

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AX, 0ABCDH

MOV CL, 04H

SHR AX, CL

INT 03H

CODE ENDS

END START

END

**2. SHL:**

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AX, 0ABCDH

MOV CL, 04H

SHL AX, CL / SAL AX,CL

INT 03H

CODE ENDS

END START

END

**3. ROTATE RIGHT:**

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AX, 0ABCDH

MOV CL, 04H

STC / CLC

ROR AX, CL

INT 03H

CODE ENDS

END START

END

**4. ROTATE LEFT:**

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AX, 0ABCDH

MOV CL, 04H

STC / CLC

ROL AX, CL

INT 03H

CODE ENDS

END START

END

**5. ROTATE RIGHT THROUGH CARRY:**

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AX, 0ABCDH

MOV CL, 04H

STC / CLC

RCR AX, CL

INT 03H

CODE ENDS

END START

END

6. **ROTATE LEFT THROUGH CARRY:**

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AX, 0ABCDH

MOV CL, 04H

STC / CLC

RCL AX, CL

INT 03H

CODE ENDS

END START

END

**1.PACKED BCD TO UNPACKED BCD**

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AL, 56H

MOV AH, AL

SHR AH, 04H

AND AL, 0FH

INT 03H

CODE ENDS

END START

END

**2. BCD TO ASCII**

ASSUME CS: CODE

CODE SEGMENT

START:

MOV AL, 56H

MOV AH, AL

SHR AH, 04H

AND AL, 0FH

OR AX,3030H / ADD AX,3030H

INT 03H

CODE ENDS

END START END

Result : Logic operations – Shift and rotate – Converting packed BCD to unpacked BCD, BCD to ASCII conversion Performed.

Experiment-4

Aim : String operation and Instruction prefix: Move Block, Reverse string, Inserting, Deleting, Length of the string, String comparison.

Experimental Requirements : PC loaded with TASM software

Procedure:

1. Switch on the PC, press windows+R then enter CMD.
2. Find the folder where TASM is located. check whether TASM.EXE, TLINK.EXE, TD.EXE are present or not
3. Enter into the directory where TASM is located by using cd... or directory name:
4. Type cd tasm in which the three files are present .Now we will be getting into c: \ or d:\ with tasm directory.
5. Type edit then a new window will be opened in which the program is entered.
6. After entering the program save the file with <filename.asm>.
7. Check for the errors or warnings by using TASM <filename> and press enter...
8. If there are no errors, then type TLINK <filename> to compile the file. If errors go back to the edit and do the necessary corrections and repeat the previous step.
9. Next type td <filename > to debug the executable file then will be getting the message program has no symbol table, press ok and then write down the instructions, registers and flags status before execution .
10. For step by step execution press F8.and for direct execution press F9 and then write down the instructions, registers and flags status after execution .Go to dump if required for noting down the required inputs and outputs.

**STRING OPERATIONS**

**1. MOVING A BLOCK OF DATA**

ASSUME CS:CODE,DS:DATA,ES:EXTRA

DATA SEGMENT

ORG 1000H

STR1 DB 'HI FRIEND$'

COUNT EQU $-STR1

DATA ENDS

EXTRA SEGMENT

ORG 2000H

STR2 DB 1 DUP(?)

EXTRA ENDS

CODE SEGMENT

START:

MOV AX,DATA

MOV DS,AX

MOV AX,EXTRA

MOV ES,AX

MOV SI,OFFSET STR1

MOV DI,OFFSET STR2

MOV CL,COUNT-1

REP MOVSB

INT 03H

CODE ENDS

END START

END

2. **REVERSE OF A STRING**

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

ORG 1000H

STR1 DB 'HI FRIEND$'

COUNT EQU $-STR1

DATA ENDS

CODE SEGMENT

START:

MOV AX,DATA

MOV DS,AX

MOV SI,OFFSET STR1

MOV DI,OFFSET STR1+COUNT-2

MOV CL,COUNT/2

BACK:MOV AL,[SI]

XCHG [DI],AL

XCHG [SI],AL

INC SI

DEC DI

LOOP BACK

INT 03H

CODE ENDS

END START

END

3. **STRING COMPARISON**

ASSUME CS:CODE, DS:DATA,ES:EXTRA

DATA SEGMENT

ORG 1000H

STR1 DB 'HI FRIEND$'

COUNT EQU $-STR1

DATA ENDS

EXTRA SEGMENT

ORG 2000H

STR2 DB 'HIFRIEND'

EXTRA ENDS

CODE SEGMENT

START:

MOV AX,DATA

MOV DS,AX

MOV AX,EXTRA

MOV ES,AX

MOV SI,OFFSET STR1

MOV DI,OFFSET STR2

MOV CL,COUNT-1

REP CMPSB

INT 03H

CODE ENDS

END START

END

4. STRING INSERTION

ASSUME CS: CODE, DS: DATA, ES: EXTRA

DATA SEGMENT

ORG 1000H

STR1 DB 'STUDENTS MAKE NOISE$'

STRLEN EQU ($-STR1)

DATA ENDS

EXTRA SEGMENT

ORG 2000H

STR2 DB STRLEN + 1 DUP(0H)

EXTRA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AX, EXTRA

MOV ES, AX

MOV SI, OFFSET STR1

MOV DI, OFFSET STR2

MOV CL, 0DH

CLD

REP MOVSB

MOV DL, 05H

BACK:

MOV AH, 01H

INT 21H

STOS STR2

DEC DL

JNZ BACK

MOV CL, 06H

REP MOVSB

INT 03H

CODE ENDS

END START

END

5. STRING DELETION

ASSUME CS: CODE, DS: DATA, ES: EXTRA

DATA SEGMENT

ORG 1000H

STR1 DB 'CSE STUDENTS MAKE NOISE$'

STRLEN EQU ($-STR1)

DATA ENDS

EXTRA SEGMENT

ORG 2000H

STR2 DB STRLEN -1 DUP(0H)

EXTRA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AX, EXTRA

MOV ES, AX

MOV SI, OFFSET STR1

MOV DI, OFFSET STR2

MOV CL, 0DH

CLD

REP MOVSB

MOV SI, 1012H

MOV CL, 0005H

REP MOVSB

INT 03H

CODE ENDS

END START

END

6. LENGTH OF A STRING

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

ORG 1000H

STR1 DB 'HIFRIEND$'

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV SI, OFFSET STR1

MOV AL, '$'

MOV CL, 00H

BACK:

CMP AL, [SI]

JZ L2

INC CL

INC SI

JMP BACK

L2:

MOV AL, CL

INT 03H

CODE ENDS

END START

END

Result : String operation and Instruction prefix: Move Block, Reverse string, Inserting, Deleting, Length of the string, String comparison performed.

Experiment-5

Aim : DOS/BIOS programming: Reading keyboard (Buffered with and without echo) – Display characters, Strings.

Experimental Requirements: PC loaded with TASM software

Procedure:

1. Switch on the PC, press windows+R then enter CMD.
2. Find the folder where TASM is located. check whether TASM.EXE, TLINK.EXE, TD.EXE are present or not
3. Enter into the directory where TASM is located by using cd... or directory name:
4. Type cd tasm in which the three files are present .Now we will be getting into c: \ or d:\ with tasm directory.
5. Type edit then a new window will be opened in which the program is entered.
6. After entering the program save the file with <filename.asm>.
7. Check for the errors or warnings by using TASM <filename> and press enter...
8. If there are no errors, then type TLINK <filename> to compile the file. If errors go back to the edit and do the necessary corrections and repeat the previous step.
9. Next type td <filename > to debug the executable file then will be getting the message program has no symbol table, press ok and then write down the instructions, registers and flags status before execution .
10. For step by step execution press F8.and for direct execution press F9 and then write down the instructions, registers and flags status after execution .Go to dump if required for noting down the required inputs and outputs.

1. Reading keyboard (Buffered with echo) – Display characters, Strings.

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

MSG DB 'WELCOME TO PVPSIT$'

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AH, 09H

MOV DX, OFFSET MSG

INT 21H

NEXT: MOV AH, 01H

MOV DL, AL

INT 21H

CMP AL, '$'

JNE NEXT

INT 03H

CODE ENDS

END START

END

2. Reading keyboard (Buffered without echo) – Display characters, Strings.

ASSUME CS: CODE, DS: DATA

DATA SEGMENT

MSG DB 'WELCOME TO PVPSIT$'

DATA ENDS

CODE SEGMENT

START:

MOV AX, DATA

MOV DS, AX

MOV AH, 09H

MOV DX, OFFSET MSG

INT 21H

NEXT: MOV AH, 08H

MOV DL, AL

INT 21H

CMP AL, '$'

JNE NEXT

INT 03H

CODE ENDS

END START

END

Result: DOS/BIOS programming: Reading keyboard (Buffered with and without echo) – Display characters, Strings performed.

Interfacing

Experiment-1

8255 PPI Square Wave Generator

Aim:To write an assembly language program to generate square wave generator Programmable Peripherial Interface

Experiment Requirements: PC loaded with 8086E software, 8086E Kit, 8255 PPI and connecting wires and cables

PROCEDURE:

1. Switch on the PC, check for ESA term and enter into TERM86E.
2. In the kit, we need to check serial in display.
3. Reset your kit.
4. Then we will get into prompt mode.
5. Give the program starting address with the syntax:A 2000 and give enter.
6. Enter the program.
7. For termination from the program mode press ESC.
8. After that type G , give space and Then enter the starting address.
9. Then the square wave is displayed on the CRO.
10. Then observe the waveform and take readings like amplitude and time period.
11. For getting the opcodes use Z command.
12. Syntax: Z starting address , ending address

Program:

MOV AL,80

MOV DX,0FFE6

OUT DX, AL

L3: MOV AL, 0FFH

MOV DX, 0FFE0

OUT DX, AL

MOV CX, 00FF

L1: LOOP L1

MOV AL,00

OUT DX,0FFE0

OUT DX,AL

MOV CX,00FF

L2 : LOOP L2

JMP L3

INT 03

Result: Hence ALP to generate square wave using 8255 PPI by interfacing with 8086 microprocessor is performed by using tasm

Experiment-2

Stepper Motor Interfacing With 8086

Aim: To write an ALP to rotate stepper motor in clockwise and anticlockwise direction by interfacing stepper motor with 8086 microprocessor

Experiment Requirements: PC loaded with 8086E software, 8086E Kit, 8255 PPI and connecting wires and cables, Stepper Motor, Power Supply

PROCEDURE:

PROCEDURE:

1. Switch on the PC, check for ESA term and enter into TERM86E.
2. In the kit, we need to check serial in display.
3. Reset your kit.
4. Then we will get into prompt mode.
5. Give the program starting address with the syntax:A 800 and give enter.
6. Enter the program.
7. For termination from the program mode press ESC.
8. After that type G , give space and Then enter the starting address.
9. Motor will left / right depending on RCL, RCR command respectively.
10. For getting the opcodes use Z command.
11. Syntax: Z starting address , ending address

Program

MOV AL,80

MOV DX,0FFE6

OUT DX,AL

MOV AL, 11

MOV DX, 0FFE0

L2: OUT DX, AL

MOV CX, 0800

L1: LOOP L1

RCL AL,1 / RCR AL,1

JMP L2

INT 03

Result: Stepper Motor has been rotated in clockwise and anticlockwise direction using 8086 microprocessor

Experiment-3

8279 Keyboard Display

Aim : To write a program to display the string of charcaters using 8279 keyboard

Experiment Requriments : PC loaded with XT86,XT86 Kit,8279 Keyboard Display Device

Procedure:

1. Go to ESA directory.
2. Type xt86, then we will be getting a message press any key to continue.
3. Press RST from the trainer kit.
4. Press A to exit from DOT and entering into prompt mode.
5. Give SG 00 to clear all the segments.
6. Press LC to clear all the labels.
7. Next DA address, enter the program in the window.
8. The program is to be entered a space before every instruction.
9. For the execution shift+1 (or)! Is entered.
10. Then type EX and enter key.
11. enter D6,93,67,F3,F3,83 using S command to the address specified in to SI.
12. After that type G and enter key.

Program:

MOVB AL, #90

MOVW DX, #82

OUTB DX

MOVB AL, #00

OUTB DX

MOVW CX, #08

RPT: MOVB AL, #00

MOVW DX, #80

OUTB DX

LOOP 200C

MOVW CX, #06

MOVW SI, #2100

LOOP: MOVB AL, [SI]

OUTB DX

INCW [SI]

LOOP 201A

INT 03

Result: Here ALP to display string of characters using 8279 keyboard display