METHODIST

Estd:2008

COLLEGE OF ENGINEERING AND TECHNOLOGY

(Affiliated to Osmania University & Approved by AICTE, New Delhi)

LABORATORY MANUAL

MICROPROCESSOR AND MICROCONTROLLERS (CODE: PC 752 EE)

BE VII Semester (CBCS): 2020-21

NAME: _____

ROLL NO:_____

BRANCH:_____ **SEM:**_____

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

Empower youth- Architects of Future World

METHODIST

Estd:2008

COLLEGE OF ENGINEERING AND TECHNOLOGY

VISION

To produce ethical, socially conscious and innovative professionals who would contribute to sustainable technological development of the society.

MISSION

To impart quality engineering education with latest technological developments and interdisciplinary skills to make students succeed in professional practice.

To encourage research culture among faculty and students by establishing state of art laboratories and exposing them to modern industrial and organizational practices.

To inculcate humane qualities like environmental consciousness, leadership, social values, professional ethics and engage in independent and lifelong learning for sustainable contribution to the society.

METHODIST

Estd:2008

COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT
OF
ELECTRICAL AND ELECTRONICS ENGINEERING

LABORATORY MANUAL

MICROPROCESSOR AND MICROCONTROLLERS

Prepared

By

Dr. Raghu Chandra Garimella,

Associate Professor.

METHODIST

Estd:2008

COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

VISION

To become a reputed centre for imparting quality education in Electrical and Electronics Engineering with human values, ethics and social responsibility.

MISSION

- To Impart fundamental knowledge of Electrical, Electronics and Computational Technology.
- To develop professional skills through hands-on experience aligned to industry needs.
- To Undertake research in sunrise areas of Electrical and Electronics Engineering.
- To Motivate and facilitate individual and team activities to enhance personality skills.

METHODIST

Estd:2008

COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

PROGRAM EDUCATIONAL OBJECTIVES

The Graduates of the programme shall be able to:

PEO1: Utilize domain knowledge required for analyzing and resolving practical Electrical Engineering problems.

PEO2: Willing to undertake inter-disciplinary projects, demonstrate the professional skills and flair for investigation.

PEO3: Imbibe the state of the art technologies in the ever transforming technical scenario.

PEO4: Exhibit social and professional ethics for sustainable development of the society.

METHODIST

COLLEGE OF ENGINEERING AND TECHNOLOGY

Estd:2008

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

PROGRAM OUTCOMES

Engineering Graduates will be able to:

PO1: Engineering Knowledge: Apply knowledge of mathematics and science, with fundamentals of Computer Science & Engineering to be able to solve complex engineering problems related to CSE.

PO2: Problem Analysis: Identify, Formulate, review research literature and analyze complex engineering problems related to CSE and reaching substantiated conclusions using first principles of mathematics, natural sciences and engineering sciences.

PO3: Design/Development of solutions: Design solutions for complex engineering problems related to CSE and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety and the cultural societal and environmental considerations.

PO4: Conduct Investigations of Complex problems: Use research–based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern Tool Usage: Create, Select and apply appropriate techniques, resources and modern engineering and IT tools including prediction and modeling to computer science related complex engineering activities with an understanding of the limitations.

PO6: The Engineer and Society: Apply Reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the CSE professional engineering practice.

PO7: Environment and Sustainability: Understand the impact of the CSE professional engineering solutions in societal and environmental contexts and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply Ethical Principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and Team Work: Function effectively as an individual and as a member or leader in diverse teams and in multidisciplinary Settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large such as able to comprehend and with write effective reports and design documentation, make effective presentations and give and receive clear instructions.

PO11: Project Management and Finance: Demonstrate knowledge and understanding of the engineering management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multi disciplinary environments.

PO12: Life-Long Learning: Recognize the need for and have the preparation and ability to engage in independent and life-long learning the broadest context of technological change.

METHODIST

Estd:2008

COLLEGE OF ENGINEERING AND TECHNOLOGY

PROGRAM SPECIFIC OUTCOMES

Electrical and Electronics Engineering Graduates will be able to:

PSO1: Provide effective solutions in the fields of Power Electronics, Power Systems and Electrical Machines using MATLAB/MULTISIM.

PSO 2: Design and Develop various Electrical and Electronics Systems, particularly Renewable Energy Systems.

PSO 3: Demonstrate the overall knowledge and contribute for the betterment of the society.

METHODIST

Estd:2008

COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

I. PREREQUISITE(S):

Level	Credits	Semester	Prerequisites
UG	3	VI	MICROPROCESSOR AND MICROCONTROLLERS

II. SCHEME OF INSTRUCTIONS

Lectures	Tutorials	Practicals	Credits
3	1	-	3

III. SCHEME OF EVALUATION & GRADING

S. No	Component	Duration	Maximum Marks
	Continuous Internal Evaluation (CIE)		
1.	Internal Examination – I	1 hours	25
	CIE (Total)		25
2.	**Semester End Examination** (University Examination)	3 hours	50
		TOTAL	**75**

Marks Range	85-100	70 to < 85	60 to < 70	55 to < 60	50 to < 55	40 to < 50	< 40	Absent
Grade	S	A	B	C	D	E	F	Ab
Grade Point	10	9	8	7	6	5	0	-

IV. SYLLABUS

S. No.	List of Experiments	Target Hours
	For 8086: ***Section 1: Using MASM/TASM***	
1	Programs for signed/unsigned multiplication and division	2
2	Programs for finding average of N 16-bit numbers	2
3	Programs for finding the largest number in an array	2
4	Programs for code conversion like BCD numbers to 7-Segment	2
5	Programs for compute factorial of a positive integer number	2
	Section 2: Using 8086 Kit (Interfacing)	
6	8279 – Keyboard Display: Write a small program to display a string of characters	2
7	8255-PPI: Write ALP to generate triangular wave using DAC	2
8	8253- Timer/Counter: Application of different modes	2
9	8251-USART: Write a program in ALP to establish Communication between two processors	2
10	Traffic Signal Controller	2
	For 8051: ***Section 3: Using 8051 Kit (Simple Programs)***	
11	Data Transfer – Block move, Exchange, sorting, Finding largest element in an array	2
12	Arithmetic Instructions: Multibyte operations	2
13	Boolean & Logical Instructions (Bit manipulations)	2
14	Programs to generate delay, programs using serial port and on-Chip timer/Counter	2
15	Use of JUMP and CALL instructions	2
	Section 4 : Program Development using 'C' cross compiler for 8051	
16	Square Wave Generation using timers	2
17	Interfacing of keyboard and 7-segment Display Module	2
18	ADC interfacing for temperature monitoring	2
19	DAC interfacing for Generation of Sinusoidal wave	2
20	Stepper motor control (clockwise, anticlockwise and in precise angles)	2
	Total	**40**

Suggested Reading:

1. Douglas, V. Hall microprocessors and Interfacing -Tata McGraw Hill -Revised 2n`' Edition, 2006.
2. Krishna Kant - microprocessors and Microcontrollers - Architecture, Programming and System Design 8085, 8086, 8051, 8096, Prentice-Hall India - 2007.
3. Kenneth, J, Ayala—The 8051 Microcontroller Architecture Programming and Applications", Thomson publishers, 2"'Ilidition, 2007.
4. Waiter A. Triebel & Av-tar Singh - The 8088 and 8086 Microprocessor -Pearson Publishers, 4th Edition, 2007.

V. E – RESOURCES

https://nptel.ac.in/courses/108105102/

https://nptel.ac.in/courses/106108100/

https://nptel.ac.in/courses/108107029/

VI. COURSE OBJECTIVES:

Course Overview: This course will start with a discussion on a simple microprocessor, 8086. Understanding this architecture is the basis to follow any other complex CPU architecture. A number of devices can be interfaced with them to develop a complete system application. On the other hand, microcontrollers are single chip computers, integrating processor, memory and other peripheral modules into a single System-on-Chip (SoC). Apart from input-output ports, the peripherals often include timers, data converters, and communication modules. It will be followed by a complete overview of microcontrollers covering 8051. This course also explains in detail about the architecture and special function registers of 8051 and focuses on the learning way to program features like I/O ports, serial port, timers, counters and most interesting thing will be to do all with interrupts.

The objectives of this course are to impart to the following to the students:

- To be able to understand in detail about 8086-microprocessor architecture, programming and interfacing.
- To understand various interfacing memory and I/O devices with 8086.
- To be able to understand about 8051 microcontroller architecture, and programming.
- To implement the middle level programming and interfacing concepts in 8051.

VII. COURSE OUTCOMES

After completion of this course, the student will be able to:

CO No.	Course Outcome	Taxonomy Level
752.1	**Adapt** the knowledge of Architecture of 8086 and 8051, writing assembly language programming for different applications.	**Create**
752.2	**Explain** types of microcontrollers and their applications.	**Understand**
752.3	**Develop** a write programs to run on 8086 microprocessor based systems.	**Apply**
752.4	**Define** the techniques for faster execution of instructions, improve speed of operations and enhance performance of microprocessors.	**Remember**
752.5	**Interpret** the difference between Microprocessors and Microcontrollers.	**Evaluate**
752.6	**Simplify** and **design** system using memory chips and peripheral chips for 16-bit 8086 microprocessor.	**Analyse and Create**

VIII. MAPPING OF COs WITH POs & PSOs

Correlation Level: High – 3; Medium – 2; Low – 1

PO / CO	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PS0 1	PSO 2	PSO 3
C752.1	2	1	-	-	1	-	-	1	1	1	-	-	-	1	2
C752.2	3	2	-	-	1	-	-	1	1	1	-	-	-	1	3
C752.3	3	2	-	-	1	-	-	1	1	1	-	-	-	1	3
C752.4	3	3	1	1	1	-	-	1	1	1	-	-	-	1	3
C752.5	3	3	2	2	1	-	-	1	1	1	-	-	-	1	3
C752.6	3	3	3	3	1	-	-	1	1	1	-	-	-	1	3
C752	2.8	2.3	2	2	1	-	-	1	1	1	-	-	-	1	2.8

IX. Usage of chalk and board for better understanding. METHOD OF ASSESSMENT OF COs and POs:

COs	Relevant POs	Mode of Assessment
C752.1-C752.6	PO1: ENGINEERING KNOWLEDGE PO2: PROBLEM ANALYSIS PO3: DESIGN/ DEVELOPMENT OF SOLUTIONS PO4: CONDUCT INVESTIGATION ON COMPLEX PROBLEMS	Assignments, Quizzes, Internal Examinations and External Examination result
C752.1-C752.6	PO5: MODERN TOOL USAGE	Exercises to learn through ICT tools and internet websites, Usage of Excel worksheets for problem solving
C752.1-C752.6	PO8: ETHICS	Assignments, Quizzes
C752.1-C752.6	PO9: INDIVIDUAL AND TEAM WORK PO10: COMMUNICATION	Group Assignments, Writing skills in documenting assignments, Presentations

Prepared by: Dr. Raghu Chandra Garimella, Associate Professor, EEE

METHODIST

Estd:2008

COLLEGE OF ENGINEERING AND TECHNOLOGY

Laboratory Code of Conduct

1. Students should report to the labs concerned as per the scheduled time table.
2. Students, who report late to the labs will not be permitted to perform the experiment scheduled for the day.
3. Students to bring a 100 pages note book to enter the readings /observations while performing the experiment.
4. After completion of the experiment, certification of the staff in-charge concerned, in the observation book is necessary.
5. Staff member in-charge shall evaluate for 25 marks, each experiment, based on continuous evaluation which will be entered in the continuous internal evaluation sheet.
6. The record of observations, along with the detailed procedure of the experiment performed in the immediate previous session should be submitted for certification by the staff member in-charge.
7. Not more than three students in a group would be permitted to perform the experiment on the equipment-based lab set up. However only one student is permitted per computer system for computer-based labs.
8. The group-wise division made at the start of the semester should be adhered to, and no mix up with any other group would be allowed.
9. The components required, pertaining to the experiment should be collected from the stores in-charge, after duly filling in the requisition form / log register.
10. After the completion of the experiment, students should disconnect the setup made by them, and return all the components / instruments taken for the purpose, in order.
11. Any damage of the equipment or burn-out of components will be charged at cost as a penalty or the total group of students would be dismissed from the lab for the semester/year.
12. Students should be present in the lab for the total time duration, as scheduled.

13. Students are required to prepare thoroughly, before coming to Laboratory to perform the experiment.
14. Procedure sheets / data sheets provided to the students, if any, should be maintained neatly and returned after the completion of the experiment.

Microprocessor and Microcontrollers Laboratory

LIST OF EXPERIMENTS

Exp. No.	Experiment Name
	Section-I: USING MASM Software
	Addition of two 16-bit numbers using 8086 programming.
1	(a) Multiplication of two 16-bit unsigned numbers using 8086 programming. (b) Multiplication of two 16-bit signed numbers using 8086 programming. (c) Division of two 16-bit unsigned numbers using 8086 programming. (d) Division of two 16-bit signed numbers using 8086 programming.
2	Average of N 16-bit numbers using 8086 programming.
3	Program to find the largest number from a series of N numbers.
4	Program to find factorial of a number.
5	Program for BCD to 7 segment code conversion.
	Section-II: Using 8086 Kit (Interfacing)
6	Digital-to-Analog Converter IC interface to 8086 MP Kit.
	Section-III: Using 8051 Kit (Simple Programs)
7	Arithmetic operations on 8051 Microcontroller Kit.
8	Logical and Boolean Operations.
9	Jump and Call Instructions.
	Section-IV: Program Development using 'C' cross compiler for 8051
10	Digital-to-Analog Converter IC interface to 8051 MC Kit.
11	Stepper Motor interface to 8051 MC Kit.
	Section-V: Additional Experiments
12	Program to arrange a given array of bytes (numbers) in descending order. (8086 ALP)
13	Program to arrange a given array of bytes (numbers) in ascending order. (8086 ALP)
14	Stepper Motor interfacing to 8086 MP Kit. (Interfacing to 8086 MP)

Microprocessor and Microcontrollers Laboratory

INDEX

EXP. NO.	EXPERIMENT NAME	DATE	INSTRUCTOR'S SIGN	REMARKS
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				

INTRODUCTION TO MASM

The Microsoft macro assembler is an x86 high level assembler for DOS and Microsoft windows. It supports wide varieties of macro facilities and structured programming idioms including high level functions for looping and procedures.

A program called assembler used to convert the mnemonics of instructions along with the data into the equivalent object code modules, these object code may further converted into executable code using linked and loader programs. This type of program is called as ASSEMBLY LANGUAGE PROGRAMMING. The assembler converts and Assembly language source file to machine code the binary equivalent of the assembly language program. In this respect, the assembler reads an ASCII source file from the disk and program as output. The major different between compilers for a high level language like PASCAL and an Assembler is that the compiler usually emits several machine instructions for each PASCAL statement. The assembler generally emits a single machine instruction for each assembler language statement.

Attempting to write a program in machine language is not particularly bright. This process is very tedious, mistakes, and offers almost no advantages over programming in assembly language. The major disadvantages over programming in assembly language over pure machine code are that you must first assemble and link a program before you can execute it. However attempting to assemble the code by hand would take for longer than the small amount of time that the assembler takes the perform conversion for you. An assembler like Microsoft Macro Assembler (MASM) provides a large number of features for assembly language programmers. Although learning about these features take a fair amount of time. They are so useful that it is well worth the effort.

Microsoft MASM version 6.11 contains updated software capable of processing printing instructions. Machine codes and instruction cycle counts are generated by MASM for all instructions on each processor beginning with 8086. To assemble the file PROG.ASM use this command: (better to use DOS command line)

MASM PROG.ASM

The MASM program will assemble the PROG.ASM file. (To create PROG.OBJ from PROG.ASM)

To create PROG.EXE from PROG.OBJ, use this LINK command:

LINK PROG.OBJ

USING DEBUG TO EXECUTE THE 80x86 PROGRAM:

DEBUG is a utility program that allows a user to load an 80x 86 programs into memory and execute it step by step. DEBUG displays the contents of all processor registers after each instruction execute, allowing the user to determine if the code is performing the desired task. DEBUG only displays the 16-bit portion of the general purpose registers. Code view is capable of displaying the entire 32 bits. DEBUG is a very useful debugging tool. We will use DEBUG to step through a number of simple programs, gaining familiarity with Debug's commands as we do so. DEBUG contains commands that can display and modify memory, assemble instructions,

disassemble code already placed into memory, trace single or multiple instructions, load registers with data and do much more.
DEBUG loads into memory like any other program, in the first available slot. The memory space used by DEBUG for the user program begins after the end of Debug's code. If an .EXE or .COM file were specified, DEBUG would load the program according to accepted DOS conventions.

To execute the program file PROG.EXE use this command

DEBUG PROG.EXE

DEBUG uses a minus sign as its command prompt, so should see a "-" appear on display.
To get a list of some commands available with DEBUG is:

T - Trace (step by step execution)
U - Un assemble
D - Dump
G - Go (complete execution)
H - Hex

To execute the program file PROG.ASM use the following procedure:

.MASM PROG.ASM
.LINK PROG.OBJ
.DEBUG PROG.EXE

ASSEMBLER DIRECTIVES:

The limits are given to the assembler using some pre defined alphabetical strings called Assembler Directives which help assembler to correctly understand. The assembly language programs to prepare the codes.

DB	GROUP	EXTRN
DW	LABEL	TYPE
DQ	LENGTH	EVEN
DT	LOCAL	SEGMENT
ASSUME	NAME	
END	OFFSET	
ENDP	ORG	
ENDS	PROC	
EQU	PTR	

a) DB-Define Byte: The DB drive is used to reserve byte of memory locations in the available on memory.
b) DW-Define Word: The DW drive is used to reserve 16 byte of memory location available on memory.
c) DQ-Define Quad Word (4 words): The DB directives is used to reserve 8 bytes of memory locations in the memory available.
d) DT-Define Ten Byte: The DT directive is used to reserve 10 byte of memory locations in the available memory.

e) ASSUME: Assume local segment name the Assume directive is used to inform the assembler. The name of the logical segments to be assumed for different segment used in programs.
f) END: End of the program the END directive marks the end of an ALP.
g) ENDP: End of the procedure.
h) ENDS: End of the segment.
i) EQU: The directive is used to assign a label with a variable or symbol. The directive is just to reduce recurrence of the numerical values or constants in the program.
j) OFFSET: Specifies offset address. SEGMENT: The segment directive marks the starting of the logical segment.

EXECUTION OF ASSEMBLY LANGUAGE PROGRAMMING IN MASM SOFTWARE:

Assembly language programming has 4 steps.
1. Entering Program
2. Compile Program
3. Linking a Program
4. Debugging a Program

PROCEDURE:

1. Entering Program:-

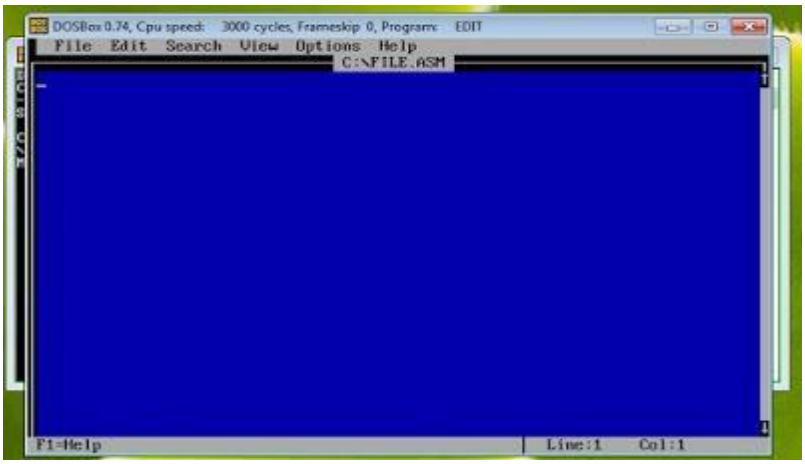
Create a new folder in local disk (C): naming it by your roll number.

Open the folder naming 8086 on the desktop.

Select DEBUG, EDIT, LINK, MASM files and copy them into the folder you created in local disk(c): 5056(say).

Now select DOS BOX 0.74 on the desktop.

After Z :\> type the following commands
mount c c:\5056.
Then press Enter and type
c:
then press Enter.
Now you are ready to write your first program. So type EDIT program_name.ASM
an editor will open like below-

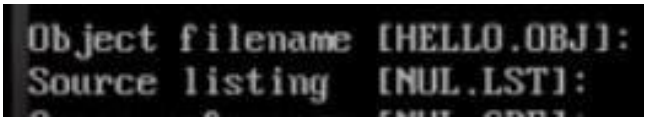


Now write your program here.
Write the program and save it as a name. I will use 'hello'. You should know that the program has an extension (.asm) so a file will create as: 'hello.asm '
after saving the program you exits from the editor.

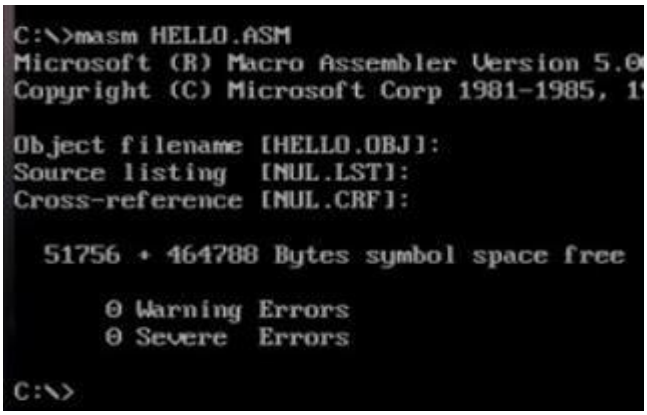
2. Compile a program:

When you exit from the editor, it will be returned back you into your previous DOSBox0.74 window. You must see - C :\>
Now you will write as like - "masm program-name.asm "
In our example, I write the following command: masm hello.asm
Then press Enter .You will see like below:-

Now again press Enter .You will see like below:-



Now again press Enter .You will see like below:-



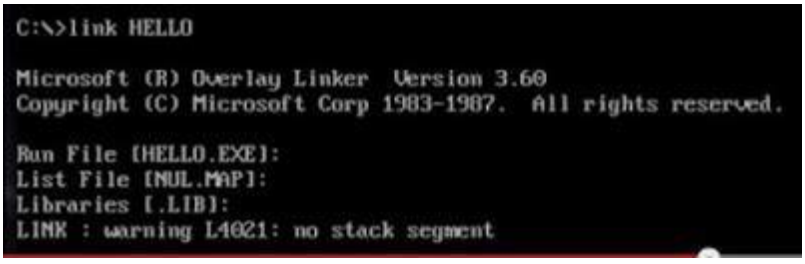
If there are any errors in your code, you must see here. If any errors are noticed than (EDIT program_name.ASM) using the command open the program file and correct the errors and warnings. Then follow the previous sequence.

3. Linking a program:

When you reach here then, type the command like “link program_name_without_extension “. In our example, we use as below -

LINK hello

Now press Enter .You will watch as like below:-



4. Debugging a program:

Now write a command as like - "debug program-name.exe"
For our practicing example-hello.exe then press Enter
-g and get your result.

To gets out from the DOSBox0.74 command window type- exit

PROGRAMS BASED ON MASM / TASM SOFTWARE

Addition of two 16-bit numbers using 8086 programming

AIM: To write an assembly language program for addition of two 16-bit numbers using MASM Tool.

APPARATUS:

1. PC with windows OS
2. MASM Tool

PROGRAM:

```
ASSUME CS: CODE, DS: DATA
DATA SEGMENT
OPER1 DW 1234H
OPER2 DW 0006H
RESULT DW 02 DUP(?)
DATA ENDS
CODE SEGMENT
START:  MOV AX, DATA
        MOV DS, AX
        MOV AX, OPER1
        MOV BX, OPER2
        CLC
        ADD AX, BX
        MOV DI, OFFSET RESULT
        MOV [DI], AX
        INT 03H
CODE  ENDS
END START
```

OUTPUT:

RESULT:

Viva Questions:

1. What is the reset address of 8086?

2. What are the special functions of AX register?

3. What is the importance of BX in 8086?

4. What are Assembler directives? Describe its use.

5. What is stack? Give stack related instructions.

6. What is function of XLAT instruction?

7. Give examples for 8 / 16 / 32 bit Microprocessor?

8. Describe ASSUME and OFFSET directive.

9. What is the use of SEGMENT and ENDS directives?

10. What is the function of 4Ch of INT 21h?

EXP. NO. 1(a) Multiplication of two 16-bit unsigned numbers using 8086 programming

AIM: To write an assembly language program to perform multiplication of two 16-bit unsigned numbers using MASM Tool.

APPARATUS:

1. PC with windows OS
2. MASM Tool

PROGRAM:

```
ASSUME CS: CODE, DS: DATA
DATA SEGMENT
OPER1 DW 1234H
OPER2 DW 0006H
RESLW  DW (?)
RESHW  DW (?)
DATA ENDS
CODE SEGMENT
START:    MOV AX, DATA
          MOV DS, AX
          MOV AX, OPER1
          MUL OPER2
          MOV RESLW, AX
          MOV RESHW, DX
          INT 03H
CODE  ENDS
END START
```

OUTPUT:

RESULT:

Viva Questions:

1. How many bit that 8086 microprocessor supports?

2. What is the size of data bus of 8086?

3. What is the size of address bus of 8086?

4. What is the maximum memory addressing capacity of 8086?

5. Which are the basic parts of 8086?

6. What is an addressing mode?

7. Explain about MOV instruction with examples.

8. List the different addressing modes in 8086.

9. What are the functions of BIU?

10. What are the functions of EU?

EXP. NO. 1(b) Multiplication of two 16-bit signed numbers using 8086 programming

AIM: To write an assembly language program to perform multiplication of two 16-bit signed numbers using MASM Tool.

APPARATUS:

1. PC with windows OS
2. MASM Tool

PROGRAM:

```
ASSUME CS: CODE, DS: DATA
DATA SEGMENT
OPER1 DW 7593H
OPER2 DW 6845H
RESLW  DW (?)
RESHW  DW (?)
DATA ENDS
CODE SEGMENT
START:    MOV AX, DATA
          MOV DS, AX
          MOV AX, OPER1
          IMUL OPER2
          MOV RESLW, AX
          MOV RESHW, DX
          INT 03H
CODE  ENDS
END START
```

OUTPUT:

RESULT:

Case-1: Two Positive Numbers

INPUT:

OPER1 = 7593H

OPER2 = 97BBH (2's complement of 6845H)

OUTPUT:

RESLW =

RESHW =

Case-2: One Positive Number & One Negative Number

INPUT:

OPER1 = 8A6DH (2's complement of 7593H)

OPER2 = 6845H

OUTPUT:

RESLW =

RESHW =

Case-3: Two Negative Numbers

INPUT:

OPER1 = 8A6DH (2's complement of 7593H)

OPER2 = 97BBH (2's complement of 6845H)

OUTPUT:

RESLW =

RESHW =

Viva Questions:

1. Which are the registers present in 8086?

2. What do you mean by pipelining in 8086?

3. How many general purpose registers are available in 8086?

4. List the 8086 instruction types.

5. Explain addition instruction in 8086 instruction set with examples.

6. Explain subtraction instruction in 8086 instruction set with examples.

7. Explain multiplication and division instruction in 8086 instruction set with examples.

8. What is the difference between instructions MUL & IMUL?

9. Explain about AND logic and give example instructions in 8086.

10. Explain about OR logic and give example instructions in 8086.

EXP. NO. 1(c) Division of two 16-bit unsigned numbers using 8086 programming

AIM: To write an assembly language program to perform division of two 16-bit unsigned numbers using MASM Tool.

APPARATUS:

1. PC with windows OS
2. MASM Tool

PROGRAM:

```
ASSUME CS: CODE, DS: DATA
DATA SEGMENT
OPER1 DW 2C58H
OPER2 DW 0056H
RESQ  DW (?)
RESR  DW (?)
DATA ENDS
CODE SEGMENT
START:    MOV AX, DATA
          MOV DS, AX
          MOV AX, OPER1
          DIV OPER2
          MOV RESQ, AX
          MOV RESR, DX
          INT 03H
CODE ENDS
END START
```

OUTPUT:

RESULT:

Viva Questions:

1. What is the supply requirement of 8086?

2. What is the relation between 8086 processor frequency & crystal Frequency?

3. Explain Functions of Accumulator or AX register?

4. What is Physical address? And how it is generated?

5. Which are pointers present in this 8086 and what its use?

6. How many segments present in it and what its use?

7. Explain the Functions of BX register?

8. Explain the Functions of CX register?

9. Explain the Functions of DX register?

10. Which is by default pointer for CS/ES?

EXP. NO. 1(d) Division of two 16-bit signed numbers using 8086 programming

AIM: To write an assembly language program to perform division of two 16-bit signed numbers using MASM Tool.

APPARATUS:

1. PC with windows OS
2. MASM Tool

PROGRAM:

```
ASSUME CS: CODE, DS: DATA
DATA SEGMENT
OPER1 DW 26F8H
OPER2 DW 00AAH
RESQ  DW (?)
RESR  DW (?)
DATA ENDS
CODE SEGMENT
START:    MOV AX, DATA
          MOV DS, AX
          MOV AX, OPER1
          IDIV OPER2
          MOV RESQ, AX
          MOV RESR, DX
          INT 03H
CODE  ENDS
END START
```

OUTPUT:

RESULT:

Case1: Two Positive Numbers

INPUT:

OPER1 = 26F8H

(DIVIDEND) OPER2 = 0056

(DIVISOR) OUTPUT:

RESLW

=

RESHW

=

Case2: One Positive Number & One Negative Number

INPUT:

OPER1 = D908H (2's complement of 26F8H)

OPER2 =

0056H

OUTPUT:

RESLW =

RESHW =

Case3: Two Negative Numbers

INPUT:

OPER1 = D908H (2's complement of 26F8H)

OPER2 = 00AAH (2's complement of 6845H)

OUTPUT:

RESLW

=

RESHW

=

Viva Questions:

1. Which register is used as COUNT in 8086 mp?

2. What is the function of INT 03 instruction?

3. What is the difference between minimum mode and maximum mode of 8086?

4. What are the process control instructions?

5. What is a difference between HALT and NOP instruction?

6. Which flag bit is effected when JNZ is executed?

7. What is the difference between CMP and SUB instructions

8. What is the difference between MOV and XCHG instruction?

9. Which are strings related instructions?

10. How LOOP instruction can function?

EXP. NO. 2 Average of N 16-bit numbers using 8086 programming

AIM: To write an assembly language program to perform average of N 16-bit numbers using MASM Tool.

APPARATUS:

1. PC with windows OS
2. MASM Tool

PROGRAM:

```
ASSUME CS: CODE, DS: DATA
DATA SEGMENT
M DW 0005H
ARRAY DW 0002H,0002H,0003H,0004H,0005H
V_AVG DB ?
DATA ENDS
CODE SEGMENT
START:      MOV AX, DATA
            MOV DS, AX
            MOV CX, M
            MOV AX, 0
            MOV DX, 0
            MOV SI, AX
START_LOOP: ADD AX, ARRAY[SI]
            ADD SI, 2
            LOOP START_LOOP
            DIV M
            MOV V-AVG, AL
            INT 03H
CODE  ENDS
END START
```

OUTPUT:

RESULT:

Viva Questions:

1. What is stack pointer

2. Describe CBW and CWD instructions

3. Explain about DAA instruction

4. Explain about CALL and RETURN instructions

5. How many address lines in a 4096*8 EPROM CHIP?

6. What are the uses of DOS and BIOS functions?

7. Give a comparison between CISC and RISC.

8. What is the function of ALE, and S0, S1 S2 pins?

9. What is the result of MUL AL? Initial value of AL = 03H.

10. What happens during DMA transfer?

EXP. NO. 3 Program to find the largest number from a series of N numbers

AIM: To write a program for finding largest number from a series of given N numbers

APPARATUS:

1. PC with windows OS
2. MASM Tool

PROGRAM:

```
ASSUME CS: CODE, DS: DATA
DATA SEGMENT
LIST DB 52H, 23H, 19H, 25H, 56H, 37H, 42H, 40H, 32H, 36H
COUNT EQU 0AH
LARGEST DW 01 DUP (?)
DATA ENDS
CODE SEGMENT
START:  MOV AX, DATA
        MOV DS, AX
        MOV SI, OFFSET LIST
        MOV CL, COUNT
        MOV AL, [SI]
AGAIN:  CMP AL, [SI+1]
        JNL NEXT
        MOV AL, [SI+1]
NEXT:   INC SI
        DEC CL
        JNZ AGAIN
        MOV DI, OFFSET LARGEST
        MOV [DI], AL
        MOV AH, 4CH
        INT 21H
CODE ENDS
  END START
```

OUTPUT:

RESULT:

Viva Questions:

1. Why 8086 processor is called a 16 bit processor?

2. What is the difference between microprocessor and microcontroller?

3. What is interrupt?

4. What is the size of flag register in 8086?

5. Which is the default segment base: offset pairs?

6. Can we use SP as offset address holder with CS?

7. What do you mean by assembler?

8. What is dd, dw, db?

9. What do you mean by 20 dup (0)?

10. What is use of EQU directive?

EXP. NO. 4 Program to find factorial of a number

AIM: To write a program for finding factorial of given number.

APPARATUS:

1. PC with windows OS
2. MASM Tool

PROGRAM:

```
ASSUME CS: CODE, DS: DATA
DATA SEGMENT
A DB 5
DATA ENDS
CODE SEGMENT
START:  MOV AX, DATA
        MOV DS, AX
        MOV AH, 00
        MOV AL, A
   L1:  DEC A
        MUL A
        MOV CL, A
        CMP CL, 01
        JNZ L1
        MOV AH, 4CH
        INT 21H
CODE ENDS
END START
```

OUTPUT:

RESULT:

Viva Questions:

1. What is meant by polling?

2. Difference between memory mapped I/O and peripheral I/O?

3. What is interfacing?

4. Explain Difference between JMP and CALL instructions

5. Explain about “LEA” instruction.

6. What is the main use of READY pin?

7. Explain about direction flag.

8. What is a port?

9. How many byte instruction is MOV AL, 23H?

10. Differentiate mov ax, 2304h and mov ax, [2304h].

EXP. NO. 5 Program for BCD to 7 segment code conversion

AIM: To write a program for arranging a given array of number in descending

order. APPARATUS:

1. PC with windows OS
2. MASM Tool

PROGRAM:

```
ASSUME CS: CODE, DS: DATA

   DATA SEGMENT

      LOOKUP DB 01H, 04H, 1AH, 09H, 02H
      KEY DB 02H

   DATA ENDS

   CODE SEGMENT

      START:  MOV  AX, DATA              ; INITIALIZE DATA SECTION
              MOV  DS, AX
              MOV  BX, OFFSET LOOKUP     ; LOAD OFFSET OF LOOKUP IN
BX
              MOV  AL, KEY               ; KEY NO. IN AL
              XLAT                       ; TRANSLATE BYTE IN AL
              MOV  BH, AL                ; AL = LOOKUP(KEY)
              MOV  CH, 02H               ; COUNT OF DIGITS TO BE DISPLAYED
              MOV  CL, 04H               ; COUNT TO ROLL BY 4 BITS
      L2:     ROL   BH, CL               ; ROLL BL SO THAT MSB COMES TO LSB
              MOV  DL, BH                ; LOAD DL WITH DATA TO BE DISPLAYED
              AND   DL, 0FH              ; GET ONLY LSB
              CMP  DL, 09                ; CHECK IF DIGIT IS 0-9 OR LETTER A-F
              JBE   L4
              ADD   DL, 07               ; IF LETTER ADD 37H ELSE ONLY ADD 30H
      L4:  ADD   DL, 30H
              MOV  AH, 02                ; FUNCTION 2 UNDER INT 21H (DISPLAY
                                           CHARACTER)
              INT    21H
              DEC   CH                   ; DECREMENT COUNT
              JNZ   L2
              MOV  AH, 4CH               ; TERMINATE PROGRAM
              INT    21H

   CODE ENDS

END START
```

OUTPUT:

Viva Questions:

1. Give Interrupts in 8086 and there function.

2. What do you mean by segment override prefix?

3. What is the difference between instructions MUL & IMUL?

4. Which flags are affected for CMP instruction?

5. Explain use of JB instruction.

6. What is the difference between Macro and procedure?

7. What is the difference between near and far procedure?

8. What is difference between shifts and rotate instructions?

9. Which Control signals are used for DMA operation?

10. Discuss the use of LAHF instruction.

EXPERIMENTS RELATED TO 8086 TRAINER KIT (*Interfacing*)

OPERATION OF 8086 KIT

1. Switch on the power supply. The following message will be displayed (make sure that DIP switches 3 & 7 are in ON position i.e. STAND-ALONE MODE operation).

   ESA 86E MONITOR V3.0

   KBD P: 86

   •–

2. Press "A" space 2000(memory location) & then "ENTER" key.
3. Now enter the program. At the end of each line press "ENTER" key.
4. After entering the last line of the program, press "ENTER" key & then "!".
5. Now press "S" space 2100(memory location) & then "ENTER" key to give inputs to the memory locations.
6. Press "G" & then "ENTER" key. Now press 2000(memory location) & "ENTER" key for execution of the program.
7. The output in the registers will be displayed.
8. Press "Z" space 2000, 200A to check the entire program written in the trainer kit.
9. (Here 2000 is the starting location and 200A is the ending location of the program)
10. Press "D" space 2100, 210A to check the inputs given to the memory locations.
11. Press "X" & then "ENTER" key to check the contents of the registers.

EXP. NO. 6 Digital-to-Analog Converter IC interface to 8086 MP Kit

AIM: To write an assembly language program for DAC-IC Interface assumed to be connected over connector J4 of the 8086 trainer kit.

APPARATUS:

1. 8086 Trainer kit
2. Power supply
3. Key board

PROGRAM:

1. **PROGRAM TO GENERATE SQUARE WAVE:**

ADDRESS		MACHINE CODE	LABEL	MNEMONIC		COMMENTS
SEGMENT	OFFSET			Opcode	Operands	
0000	2000			MOV	DX, 0FFE6	; Initialize all 8255
0000				MOV	AL, 80	; Ports as output
0000				OUT	DX, AL	
0000				MOV	DX, 0FFE2	
0000			A0:	MOV	AL, 00	; O/p data to ports
0000				OUT	DX, AL	
0000				CALL	DELAY	; Introduce delay
0000				MOV	AL, 0FF	
0000				OUT	DX, AL	
0000				CALL	DELAY	
0000				JMP	A0	
0000			DELAY:	MOV	CX, 00FF	; Delay subroutine
0000			L1:	LOOP	L1	
0000				RET		

OUTPUT:

2. PROGRAM TO GENERATE TRAINGULAR WAVE:

ADDRESS		MACHINE CODE	LABEL	MNEMONIC		COMMENTS
SEGMENT	OFFSET			Opcode	Operands	
0000	2000			MOV	DX, 0FFE6	; Initialize all 8255
0000				MOV	AL, 80	; Ports as output
0000				OUT	DX, AL	; O/p data to ports
0000				MOV	DX, 0FFE2	
0000			A0:	MOV	AL, 00	
0000			A1:	OUT	DX, AL	; O/p data to ports
0000				INC	AL	
0000				CMP	AL, 0FF	
0000				JC	A1	
0000			A2:	OUT	DX, AL	; O/p data to ports
0000				DEC	AL	
0000				CMP	AL, 00	
0000				JNBE	A2	
0000				JMP	A0	

OUTPUT:

3. PROGRAM TO GENERATE SAWTOOTH WAVE:

ADDRESS		MACHINE CODE	LABEL	MNEMONIC		COMMENTS
SEGMENT	OFFSET			Opcode	Operands	
0000	2000			MOV	DX, 0FFE6	; Initialize all 8255
0000				MOV	AL, 80	; Ports as output
0000				OUT	DX, AL	
0000				MOV	DX, 0FFE2	
0000			L2:	MOV	AL, 00	
0000			L1:	OUT	DX, AL	
0000				INC	AL	
0000				CMP	AL, 0FF	
0000				JB	L1	
0000				OUT	DX, AL	
0000				JMP	L2	

OUTPUT:

RESULT:

Viva Questions:

1. What is the role of DMA in MP?

2. Give examples of conditional branch instructions

3. Give examples of unconditional branch instructions

4. What are flag manipulation instructions? Give examples

5. What is the difference between Maskable interrupts and Non-Maskable interrupts?

6. Write the input/output feature in Mode 0 for the 8255A PPI?

7. What is the use of mode 2 in 8255A PPI?

8. What is control word to set Port A as output, Port B as input and port C as input for 8255.

9. What is the role of 8251 in computers?

10. Give comparison of 8086, 286, 386, 486 and Pentium processors w.r.t clock speed, data bus width, memory addressing capacity.

PROGRAMS RELATED TO 8051 TRAINER KIT

OPERATION PROCEDURE FOR 8051 KIT:

1. Type A space 8000 and press Enter key.
2. Type the entire program and save the program by pressing Esc key.
3. Input are given to the memory locations by MD space 9001(memory location) and save them by pressing Esc key.
4. For execution press G space 8000 and press Enter key.
5. 'User break occurred' statement will be displayed on the lcd screen, then press Enter key.
6. Now to see output in memory location , press MD space 9001(memory location) ,then press Enter key.
7. To see output in registers, press R and then Enter key. The data in the various registers can be checked by pressing Enter key.
8. To verify the program press Z space 8000 (starting memory location of the program)

EXP. NO. 7 Arithmetic operations on 8051 Microcontroller Kit

AIM: To write the following assembly language programs by addressing modes using 8051 Microcontroller Kit.

1. Program to perform addition of two numbers using immediate addressing mode.
2. Program to perform subtraction of two numbers.
3. Program to perform multiplication of two numbers.
4. Program to perform division of two numbers.

APPARATUS:

1. 8051 Trainer kit
2. Power supply
3. Key board

PROGRAM:

1. Program to perform addition of two 8-bit numbers using immediate addressing mode.

ADDRESS	MACHINE CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERANDS	
8000			MOV MOV ADD LCALL	A, #02 0F0, #04 A,0F0 0003	

INPUTS:
A = 02H
B = 04H

OUTPUTS:
A =

2. Program to perform subtraction of two 8-bit numbers using register addressing mode.

ADDRESS	MACHINE CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERANDS	
8000			MOV MOV SUBB LCALL	A, #02 0F0, #04 A,0F0 0003	

INPUTS:
A = 02H
B = 04H

OUTPUTS:
A =

3. Program to perform multiplication of two 8-bit numbers using register indirect addressing mode.

ADDRESS	MACHINE CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERANDS	
8000			MOV	DPTR, #9000	
			MOVX	A, @DPTR	
			MOV	0F0, A	
			INC	DPTR	
			MOVX	A, @DPTR	
			MUL	AB	
			LCALL	0003	

INPUTS:
9000 = 02H
9001 = 04H

OUTPUTS:
A =
B =

4. Program to perform division of two 8-bit numbers using register indirect addressing mode.

ADDRESS	MACHINE CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERANDS	
8000			MOV	DPTR, #9000	
			MOVX	A, @DPTR	
			MOV	0F0, A	
			INC	DPTR	
			MOVX	A, @DPTR	
			DIV	AB	
			LCALL	0003	

INPUTS:
9000 = 02H
9001 = 04H

OUTPUTS:
A =
B =

RESULT:

Viva Questions:

1. What is microcontroller?

2. What are the various criteria to choose the microcontroller?

3. List some of the 8051 microcontroller manufacturers?

4. List out some of the Features of 8051 Microcontroller?

5. What are the various types of memories used in microcontroller/microprocessor?

6. 8051 was developed using which Technology?

7. Why 8051 is called 8 Bit Microcontroller?

8. What is the width of Data Bus and Address Bus?

9. How Much On-Chip Ram is Available?

10. Define machine cycle.

EXP. NO. 8 Logical and Boolean Operations

AIM: To write the following assembly language programs on 8051 Microcontroller Kit.

1. Program to perform AND LOGIC of two numbers.
2. Program to perform OR LOGIC of two numbers.
3. Program to perform XOR LOGIC of two numbers.
4. Program to perform COMPLEMENT LOGIC of given number.
5. Program to perform SET and CLEAR of BIT in a given number.

APPARATUS:

1. 8051 Trainer kit
2. Power supply
3. Key board

PROGRAM:

1. Program to perform AND LOGIC of two numbers.

ADDRESS	MACHINE CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERANDS	
8000			MOV MOV ANL MOV LCALL	R0, #02 A, #04 A,R0 R1,A 0003	

INPUTS:
A = 02H
B = 04H

OUTPUTS:
A =
R1 =

2. Program to perform OR LOGIC of two numbers.

ADDRESS	MACHINE CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERANDS	
8000			MOV MOV ORL MOV LCALL	R0, #02 A, #04 A,R0 R1,A 0003	

INPUTS:
A = 02H
B = 04H

OUTPUTS:
A =
R1 =

3. Program to perform XOR LOGIC of two numbers.

ADDRESS	MACHINE CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERANDS	
8000			MOV	R0, #02	
			MOV	A, #04	
			XRL	A,R0	
			MOV	R1,A	
			LCALL	0003	

INPUTS: **OUTPUTS:**

A = 02H A =

B = 04H R1 =

4. Program to perform COMPLEMENT LOGIC of given number.

ADDRESS	MACHINE CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERANDS	
8000			MOV	A, #3F	
			CPL	A	
			MOV	R1, A	
			MOV	A, #03	
			CLR	A	
			MOV	R2, A	
			LCALL	0003	

INPUTS: **OUTPUTS:**

A = 3FH A = R1 =

A = 03H A = R2 =

5. Program to perform SET and CLEAR of BIT in a given number.

ADDRESS	MACHINE CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERANDS	
8000			MOV	A, #02	
			MOV	0F0, #03	
			SETB	C	
INPUTS:			ADDC	A, 0F0	**OUTPUTS:**
A = 3FH			MOV	R1, A A =	R1 =
B = 03H			CLR	C A =	R2 =
			ADD	A,0F0	
			MOV	R2, A	
			LCALL	0003	

RESULT:

Viva Questions:

1. How much total External Data Memory that can be interfaced to the 8051?

2. What is Special Function Registers (SFR)?

3. What are the Types of Interrupts in 8051?

4. What is use of EA pin?

5. Define DPTR.

6. What is interrupts?

7. How many timers are in 8051? Specify their names.

8. Explain DJNZ instruction of Intel 8051 microcontroller?

9. State the function of RS1 and RS0 bits in the flag register of Intel 8051 microcontroller?

10. Which two Ports combine to form the 16 bit Address for External Memory Access?

EXP. NO. 9 Jump and Call Instructions

AIM: To write the following assembly language programs on 8051 Microcontroller Kit..

1. Program to find Fibonacci series.
2. Program to find Factorial of 8-bit numbers.
3. Program to find largest number from the array of ten numbers.

APPARATUS:

1. 8051 Trainer kit
2. Power supply
3. Key board

PROGRAMS:

1. Program to find Fibonacci series.

ADDRESS	MACHINE CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERANDS	
8000		LOAD	MOV MOV MOV MOVX XCH ADD INC CJNE LCALL	A , #01 R0,#00 DPTR,#9000 @DPTR,A A,R0 A,R0 DPTR A,#13, LOAD 0003	

INPUTS:

A = 01H

OUTPUTS:

[9000] =01
=01
=02
=03
=05
=08
[9006]=0D

2. Program to find Factorial of 8-bit numbers.

ADDRESS	MACHINE CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERANDS	
8000		LOOP	MOV MOV MOV MUL DJNZ LCALL	A , #01 R0,#05 0F0, R0 AB R0, LOOP 0003	

INPUTS:

R0 = 5H

OUTPUTS:

A = 78H

3. Program to find largest number from the array of ten numbers.

ADDRESS	MACHINE CODE	LABEL	MNEMONIC		COMMENTS
			OPCODE	OPERANDS	
8000			MOV	DPTR,#9000	
8003			MOV	R0,#0A	
8005			MOV	B,#00	
8008		AGAIN	MOVX	A,@DPTR	
8009			CJNE	A,B,NE	
800C			AJMP	SKIP	
800E		NE	JC	SKIP	
8010			MOV	B,A	
8012		SKIP	INC	DPTR	
8013			DJNZ	R0,AGAIN	
8015			MOV	A,B	
8017			MOVX	@DPTR,A	
8018			LCALL	0003	

INPUTS:

[9000] = 01
= 02
= 03
= 04
= 05
= 06
= 07
= 08
= 09
[9009] = 0A

OUTPUTS:

A = 0AH

RESULT:

Viva Questions:

1. Define timer operation.

2. Define counter operation.

3. Mention the operating modes of timer/counter in 8051?

4. What is RS 232C?

5. Why are drivers bused in between RS232 and microcontroller?

6. Explain the function of each bit of SCON register.

7. Explain the function of each bit of PCON register.

8. How will you double the baud rate in 8051?

9. What is mode-0 operation in serial communication ports?

10. What is mode 3 operation in serial communication ports?

Programs Development using ‘C’ cross compiler for 8051

EXP. NO. 10 Digital-to-Analog Converter IC interface to 8051 MC Kit

AIM: To write an assembly language program for DAC-IC Interface assumed to be connected over connector J7 of the 8051 trainer kit.

APPARATUS:

1. 8051 Trainer kit
2. Power supply
3. Key board

THEORY:

The digital-to-analog converter (DAC) is a device widely used to convert digital pulses to analog signals. In this section we discuss the basics of interfacing a DAC to the 8051.

The two methods of creating a DAC: binary weighted and R/2R ladder. The vast majority of integrated circuit DACs, including the DAC0800 used in this section use the R/2R method since it can achieve a much higher degree of precision. The first criterion for judging a DAC is its resolution, which is a function of the number of binary inputs. The common ones are 8, 10, and 12 bits. The number of data bit inputs decides the resolution of the DAC since the number of analog output levels is equal to 2″, where *n* is the number of data bit inputs. Therefore, an 8-input DAC such as the DAC0800 provides 256 discrete voltage (or current) levels of output. Similarly, the 12-bit DAC provides 4096 discrete voltage levels. There are also 16-bit DACs, but they are more expensive.

The total current provided by the I_{out} pin is a function of the binary numbers at the DO – D7 inputs of the DAC0800 and the reference current (I_{ref}), and is as follows:

$$I_{out} = I_{ref}\left[\frac{D7}{2} + \frac{D6}{4} + \frac{D5}{8} + \frac{D4}{16} + \frac{D3}{32} + \frac{D2}{64} + \frac{D1}{128} + \frac{D0}{256}\right]$$

where, Iref=Reference current=2mA

D7 = MSB bit

D0=LSB bit

Example:

Assuming that R = 5K and I_{ref} = 2 mA, calculate V_{out} for the following binary inputs:

(a) 10011001 binary (99H) (b) 11001000 (C8H)

Solution:

(a) I_{out} = 2 mA (153/256) = 1.195 mA and V_{out} = 1.195 mA × 5K = 5.975 V

(b) I_{out} = 2 mA (200/256) = 1.562 mA and V_{out} = 1.562 mA × 5K = 7.8125 V

DAC 0800:

Fig 1 shows **DAC0800** series are monolithic 8-bit high-speed current output digital-to-analog converters (DAC) featuring typical settling times of 100 ns.. The noise immune inputs will accept variety of logic levels. The performance and characteristics of the device are essentially unchanged over the ±4.5V to ±18V power supply range and power consumption at only 33mW with ±5V supplies is independent of logic input levels.

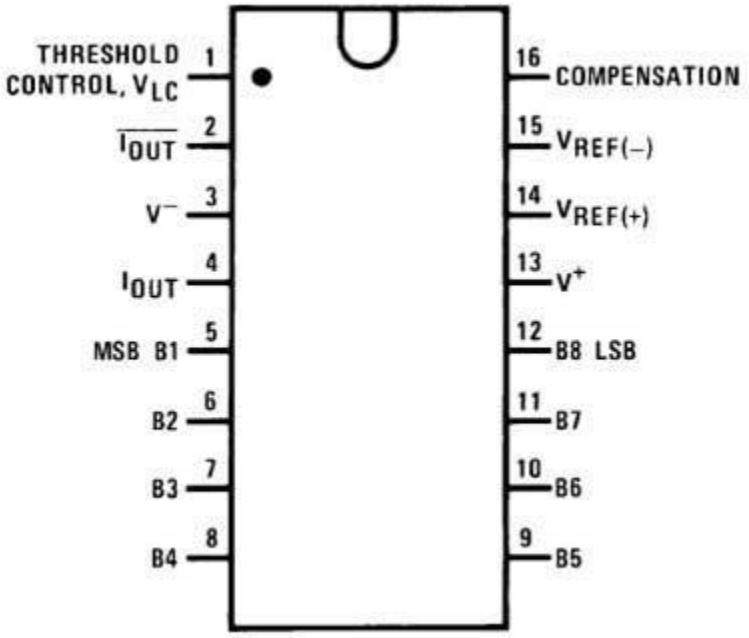


Fig 1: **pin diagram of DAC 0800**

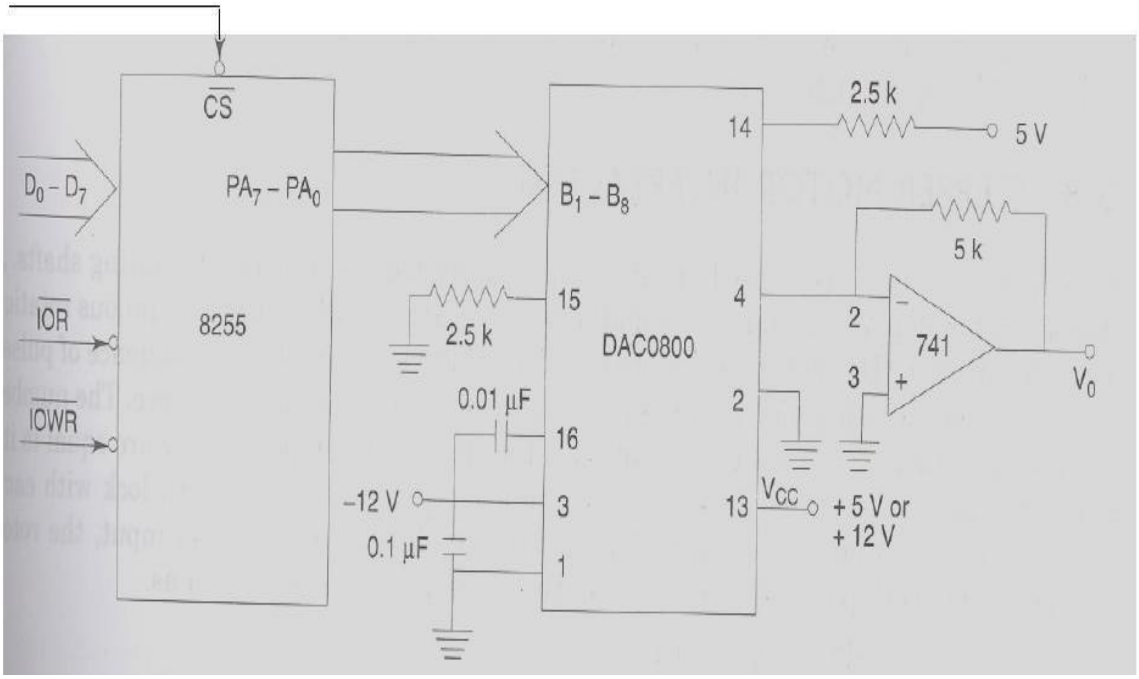


Figure: Interface diagram of DAC0800 to 8051 using 8255

The Dual DAC interface can be used to generate different interesting waveforms using microprocessor. There are two eight bit digital to analog converters provided, based on DAC 0800. The digital inputs to these DACs are provided through the port A and port B of 8255 used as output ports. The analog outputs from the DACs are given to operational amplifiers which act as current to voltage converters. Two 10K ohms pots are provided for the offset balancing of opamps. The reference voltage needed for the DACs is obtained from a onboard voltage regulator uA723. The voltage generated by this regulator is about 8V. The outputs from the DACs vary between 0 to 5V corresponding to values between 00 to FF. Different waveforms can be observed at the opamp outputs depending upon the digital input patterns.

PROGRAM:

1. **PROGRAM TO GENERATE SQUARE WAVE:**

ADDRES	MACHINE	LABE	MNEMONIC		COMMENTS
			Opcode	Operands	
8000			MOV	0A0,#0E8	; Initialize all 8255
8003			MOV	R0,#03H	; Ports as output
8005			MOV	A,#80H	
8007			MOVX	@R0,A	
8008			MOV	R0,#01	
800A		REPT:	MOV	A, #00H	; O/p data to ports
800C			MOVX	@R0, A	
800D			ACALL	DELAY	; Introduce delay
800F			MOV	A, #0FFH	;
8011			MOVX	@R0, A	
8012			ACALL	DELAY	
8014			SJMP	REPT	;
8016		DELA	MOV	R5, #0FFH	; Delay subroutine
8018		L1:	MOV	R4, #30H	
801A		L2:	DJNZ	R4,L2	
801C			DJNZ	R5,L1	
801E			RET		

OUTPUT:

2. **PROGRAM TO GENERATE TRAINGULAR WAVE:**

ADDRES	MACHINE	LABE	MNEMONIC		COMMENTS
			Opcode	Operands	
8000			MOV	0A0,#0E8	; Initialize all 8255
8003			MOV	R0,#03H	; Ports as output
8005			MOV	A,#80H	
8007			MOVX	@R0,A	
8008			MOV	R0,#01	
800A		REPT:	MOV	A, #00H	
800C		A1:	MOVX	@R0, A	
800D			INC	A	
E			CJNE	A, #0FFH,A1	
11		A2:	MOVX	@R0, A	
12			DEC	A	
13			CJNE	A, #00H,A2	
16			SJMP	REPT	

OUTPUT:

3. PROGRAM TO GENERATE SAWTOOTH WAVE:

ADDRES	MACHINE	LABE	MNEMONIC		COMMENTS
			Opcode	**Operands**	
8000			MOV	0A0,#0E8	; Initialize all 8255
8003			MOV	R0,#03H	; Ports as output
8005			MOV	A,#80H	
8007			MOVX	@R0,A	
8			MOV	R0,#01	
A		REPT:	MOV	A, #00H	
C		A1:	MOVX	@R0, A	
D			INC	A	
E			CJNE	A, #0FFH, A1	
11			SJMP	REPT	

OUTPUT:

RESULT:

Viva Questions:

1. What are the instructions used to access external RAM?

2. Mention the byte addresses of ports p0, p1, p2 and p3?

3. Discuss direct addressing mode of 8051 with examples.

4. Discuss Register indirect addressing mode of 8051 with examples.

5. Mention the SFR registers used in timer operation?

6. Explain TMOD register.

7. Explain TCON register.

8. Find the timers clock frequency for the crystal frequency of 11.0592MHz?

9. State the function of M1 and M0 bits in TMOD register?

10. What is the function of TF0 bit in TCON register?

EXP. NO. 11 Stepper Motor interface to 8051 MC Kit

AIM: To write an assembly language program for Stepper Motor Interface assumed to be connected over connector J7 of the 8051 trainer kit.

APPARATUS:

1. 8051 Trainer kit
2. Power supply
3. Key board

THEORY:

A stepper motor is stepped from one position to the next by changing the currents through the fields in the motor. The two common field connections are referred to as two phases or four phases. There are three main areas of applications for stepper motor. i. Instrumentation ii. Computer peripherals iii. Machine drives. They are used in floppy drives, dot-matrix printers, X-Y plotters, digital watches etc to rotate things in steps of small angles. The step size in typical stepper motor varies from 0.9^{o} to 30^{o}.

A stepper motor is a device used to obtain an accurate position control of rotating shafts. A stepper motor employs rotation of its shaft in terms of steps, rather than continuous rotation as in case of AC or DC motors. To rotate the shaft of the stepper motor, a sequence of pulses is needed to be applied to the windings of the stepper motor, in proper sequence. The number of pulses required for one complete rotation of the shaft of the stepper motor is equal to its number of internal teeth on its rotor. The stator teeth and the rotor teeth lock with each other to fix a position of the shaft. With a pulse applied to the winding input, the rotor rotates by one teeth position or an angle x. The angle x may be calculated as. x =360° /no. of rotor teeth after the rotation of the shaft through angle x the rotor locks itself with the next tooth in the sequence on the internal surface of stator. The internal schematic of a typical stepper motor with four windings is shown in Fig. 1.

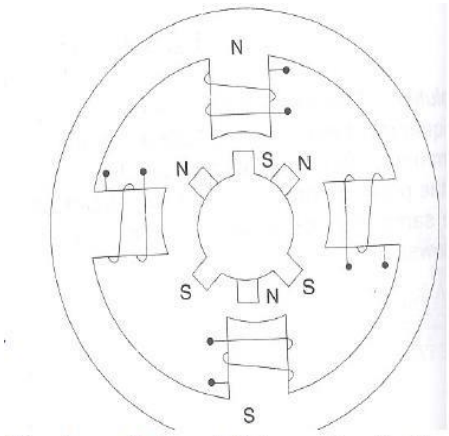


Fig. 1. Internal Schematic of a Four Winding Stepper Motor

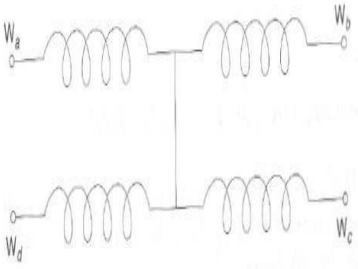


Fig. 1.1 Winding Arrangement of a Stepper Motor

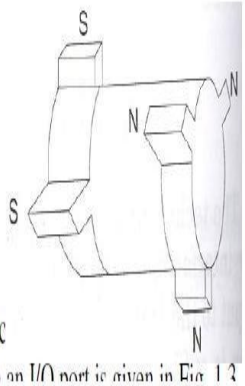


Fig. 1.2 Motor A Stepper Motor Ration

The circ
W_n with an I/O port is given in Fig. 1.3
eed
the output port.

The stepper motors have been designed to work with digital circuits. Binary level pulses of 0-5V are required at its winding inputs to obtain the rotation of shafts. The sequence of the pulses can be decided, depending upon the required motion of the shaft. Figure 1.1 shows a typical winding arrangement of the stepper motor. Figure 1.2 shows conceptual positioning of the rotor teeth on the surface of rotor, for a six teeth rotor.

A typical stepper motor may have parameters like torque 3 kg-em, operating voltage 12V, current rating 1.2A and a step angle 1.80, i.e. 200 steps/revolution (number of rotor teeth).

A simple scheme for rotating the shaft of a stepper motor is called as wave scheme. In this scheme, the windings Wa, Wb, We and Wd are applied with the required voltage pulses, in a cyclic fashion. By reversing the sequence of excitation, the direction of rotation of the stepper motor shaft may be reversed. Table 1 shows the excitation sequences for clockwise and anticlockwise rotations. Another popular scheme for rotation of a stepper motor shaft applies pulses to two successive windings at a time but these are shifted only by one position at a time. This scheme for rotation of stepper motor shaft is shown in Table 1. Table 1 excitation Sequences of a Stepper Motor Using Wave Switching Scheme.

Motion	Step	A	B	C	D
Clockwise	1	1	0	0	0
	2	0	1	0	0
	3	0	0	1	0
	4	0	0	0	1
	5	1	0	0	0
Anticlock wise	1	1	0	0	0
	2	0	0	0	1
	3	0	0	1	0
	4	0	1	0	0
	5	1	0	0	0

Stepper Motor control is a very popular application of microprocessors in control area, as stepper motors are capable of accepting pulses directly from the microprocessor and move accordingly. There are two types of stepper motors:

(a) Permanent magnet (PM)

(b) Variable reluctance (VR).

1.1 PERMANENT MAGNET STEPPER MOTORS

A practical PM stepper motor will have 1.8 degrees step angle and 50 tooth on its rotor; there are eight main poles on the stator, each having five teeth in the pole face. The step angle is given by *A = 360 / (N*K) degrees where N = number of rotor tooth K = excitation sequence factor* PM stepper motors have three modes of excitation i.e.,

Single phase mode

Two phase mode

Hybrid mode.

Single Phase Mode: Figs (a,c,e,g) illustrate the single phase mode in which only one of the motor windings is excited at a time. There are four steps in the sequence, the excitation sequence factor K=2, so that step angle is 90 degrees.

Two Phase Mode: Here both the stator phases are excited at a time as shown in figs (b,d,f,h). There are four steps in the excitation sequence, K=2 and step angle is 90 degrees. However, the rotor positions in the two-phase mode are 45 degrees away from those in single phase mode. ***Hybrid Mode:*** This is a combination of single phase and two phase modes as shown in Figs (a-h). There are eight steps in excitation sequence; K=2 and step angle = 45 degrees. From Figs (a-h), it can be observed that a voltage +V is applied to a stator winding during some steps, while voltage - V is applied during certain other steps. This requires a bipolar regulated power supply capable of

yielding +V, -V and zero outputs and a pair of SPDT switches, which is quite cumbersome. Consequently each of the two stator windings is split into two sections A1-A2, B1-B2. These sections are wound differentially as shown by the polarity dots in Fig (j). These winding sections can now be excited from a unipolar regulated power supply through switches S1 to S4 as shown in Fig (j). This type of construction is called bipolar winding construction. Bipolar winding results in reduced winding inductance and consequently improved torque stepping rate characteristics.

1.2 VARIABLE RELUCTANCE (VR) STEPPER MOTORS

The schematic diagram of a simple Variable Reluctance (VR) stepper motor is shown in fig (K). There are twelve tooth on the stator and eight on the rotor. The rotor does not carry either a permanent magnet or winding; it is assembled from soft iron punchings. The stator is also assembled from soft iron punchings, and carries stator windings A,B and C as shown in Fig (k). When stator winding A is excited, it creates a patterns of N and S poles as shown in Fig (i). The rotor then positions itself as shown in Fig (i), so as to minimize the reluctance of the magnetic circuit. When phase B is excited next, the rotor will move through 15 degrees to again seek minimum reluctance position. VR stepper motors are available as 3-5 phase motors with step angle given by

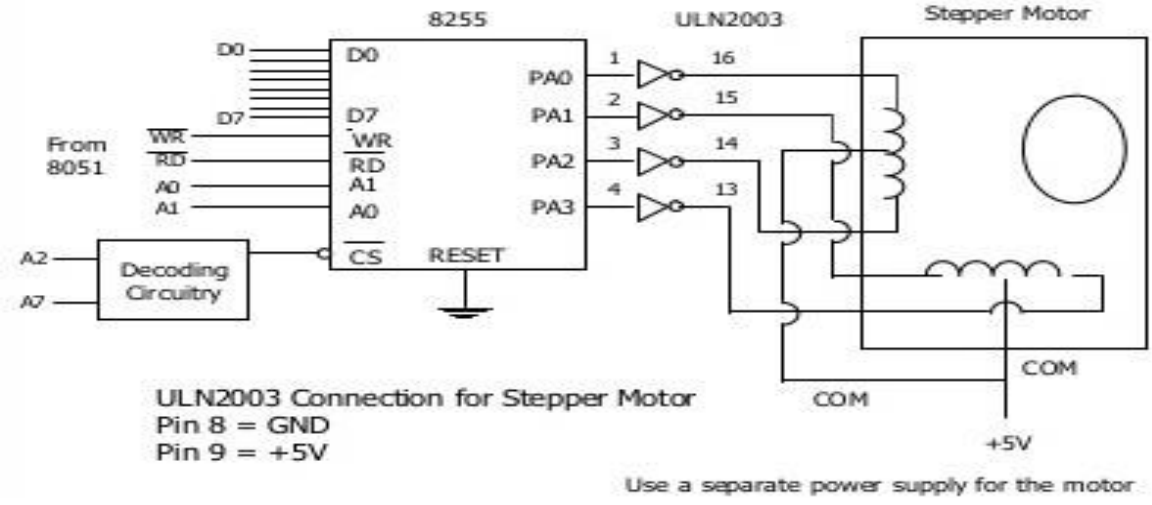
$$B = 360 (N1-N2) / (N1*N2)$$

Where N1 = number of stator tooth

*N2 = number of rotor tooth N1 and N2 are related by N1 = N2+n = p*n*

Where n = number of stator tooth per phase p = number of phases

INTERFACE DIAGRAM:



PROGRAMS:

1. Stepper Motor rotation in Clockwise direction:

ADDRES	MACHINE	LABE	MNEMONIC		COMMENTS
			Opcode	Operands	
8000			MOV	0A0,#0E8	; Initialize all 8255
8003			MOV	R0,#03H	; Ports as output
8005			MOV	A,#80H	
8007			MOVX	@R0,A	
8008			MOV	R0,#00H	; O/p data to ports
800A			MOV	A, #88H	
800C		REPT:	MOVX	@R0,A	; Introduce delay
800D			ACALL	DELAY	; rotate data byte
8010			RR	A	; rotation of motor
8011			SJMP	REPT	; Delay subroutine
8013		DELA	MOV	R5, #0FF	
8015		L1:	MOV	R4, #30	
8017		L2:	DJNZ	R4,L2	
8019			DJNZ	R5,L1	
801B			RET		

2. Stepper Motor rotation in Anti-Clockwise direction:

ADDRES	MACHINE CODE	LABEL	MNEMONIC		COMMENTS
			Opcode	Operands	
8000			MOV	P2,#0E8	; Initialize all 8255
8003			MOV	R0,#03H	; Ports as output
8005			MOV	A,#80H	
8007			MOVX	@R0,A	
8008			MOV	R0,#00H	; O/p data to ports
800A			MOV	A, #88H	
800C		REPT:	MOVX	@R0,A	
800D			ACALL	DELAY	; Introduce delay
8010			RL	A	; rotate data byte
8011			SJMP	REPT	; rotation of motor
8013		DELA	MOV	R5, #0FF	; Delay subroutine
8015		L1:	MOV	R4, #30	
8017		L2:	DJNZ	R4,L2	
8019			DJNZ	R5,L1	
801B			RET		

RESULT:

Viva Questions:

1. Explain the function of PSEN and RST pins in 8051.

2. Explain the register SP of 8051and list the stack operated instructions

3. Write about memory banks in 8051.

4. List the various applications of Microcontrollers.

5. Write the priority of interrupt in 8051.

6. Discuss immediate addressing mode of 8051 with one example.

7. Discuss Register addressing mode of 8051 with one example.

8. Explain CJNE instruction of Intel 8051 microcontroller?

9. Explain SWAP A instruction of Intel 8051 microcontroller?

10. Describe the XCHD instruction with an example.

ADDITIONAL EXPERIMENTS BEYOND THE SYLLABUS
(Based on MASM Software & Interfacing to 8086)

EXP. NO. 12 Program to arrange a given array of bytes (numbers) in descending order

AIM: To write a program for arranging a given array of number in descending

order. APPARATUS:

1. PC with windows OS
2. MASM Tool

PROGRAM:

```
ASSUME CS: CODE, DS: DATA
   DATA SEGMENT
      LIST DB 65H, 02H, 46H, 38H, 75H,
01H
      COUNT EQU 05
   DATA ENDS
   CODE SEGMENT
      START: MOV AX, DATA
             MOV DS, AX
             MOV DL, COUNT
      AGAIN:MOV CL, DL
             MOV SI, OFFSET LIST
      REPT:  MOV AL, [SI]
             CMP AL, [SI+1]
             JA NEXT
             XCHG [SI+1], AL
             XCHG [SI], AL
      NEXT:  INC SI
             LOOP REPT
             DEC DL
             JNZ AGAIN
             INT 03H
   CODE ENDS
END START
```

RESULT:

OUTPUT:

EXP. NO. 13 Program to arrange a given array of bytes (numbers) in ascending order

AIM: To write an assembly language program for arranging a given array of number in ascending order using 8086 MASM Tool.

APPARATUS:

1. PC with windows OS
2. MASM Tool

PROGRAM:

```
        ASSUME CS: CODE, DS: DATA
        DATA SEGMENT
        LIST DB 03H, 18h, 05H, 09H, 43h, 20H COUNT EQU 06
        DATA ENDS
        CODE SEGMENT
        START:      MOV AX, DATA
                    MOV DS, AX
                    MOV DL, COUNT-1
         AGAIN:     MOV CL, DL
                    MOV SI, OFFSET LIST
         REPT:      MOV AL, [SI]
                    CMP AL, [SI+1]
                    JB NEXT
                    XCHG [SI+1], AL
                    XCHG [SI], AL
         NEXT:      INC SI
                    LOOP  REPT
                    DEC DL
                    JNZ  AGAIN
                    INT 03H
            CODE ENDS
END START
```

OUTPUT:

RESULT:

EXP. NO. 14 Stepper Motor interfacing to 8086 MP Kit

AIM: To write an assembly language program for Stepper Motor Interface assumed to be connected over connector J4 of the 8086 trainer kit.

APPARATUS:

1. 8086 Trainer kit
2. Power supply
3. Key board

PROGRAM:

1. Stepper Motor rotation in Clockwise direction:

ADDRESS		MACHINE CODE	LABEL	MNEMONIC		COMMENTS
SEGMENT	OFFSET			Opcode	Operands	
0000	2000			MOV	DX, 0FFE6	; Initialize all 8255
0000				MOV	AL, 80	; Ports as output
0000				OUT	DX, AL	
0000				MOV	DX, 0FFE2	
0000				MOV	AL, 88	; O/p data to ports
0000			REPEAT	OUT	DX, AL	
0000				CALL	DELAY	; Introduce delay
0000				ROR	AL, 1	; rotate data byte
0000				JMP	REPEAT	; rotation of otor
0000			DELAY:	MOV	CX, 0800	; Delay ubroutine
0000			L1:	LOOP	L1	
0000				RET		

OUTPUT:

2. Stepper Motor rotation in Anti-Clockwise direction:

ADDRESS		MACHINE CODE	LABEL	MNEMONIC		COMMENTS
SEGMENT	OFFSET			Opcode	Operands	
0000	2000			MOV	DX, 0FFE6	; Initialize all 8255
0000				MOV	AL, 80	; Ports as output
0000				OUT	DX, AL	
0000				MOV	DX, 0FFE2	
0000				MOV	AL, 88	; O/p data to ports
0000			REPEAT:	OUT	DX, AL	
0000				CALL	DELAY	; Introduce delay
0000				ROL	AL, 1	; rotate data byte
0000				JMP	REPEAT	; rotation of motor
0000			DELAY:	MOV	CX, 0800	; Delay subroutine
0000			L1:	LOOP	L1	
0000				RET		

OUTPUT:

RESULT:

Viva Questions:

1. Define RS-232.

2. What is use of IN and OUT instruction?

3. What is the function of 8284?

4. Briefly describe how direct and indirect Jumps take place in 8086.

5. What is the function of AD0-AD15 pins in 8086?

6. Why do we need maximum mode in 8086 MP?

7. Explain the operating mode of 8255?

8. What is the use of A0 and A1 pins of 8255?

9. Write the control word format in the BSR mode.

10. Give control word to set PC-5 bit for 8255?