I

Name _____

Roll No. _____ Year 20____20____

Exam Seat No. _____

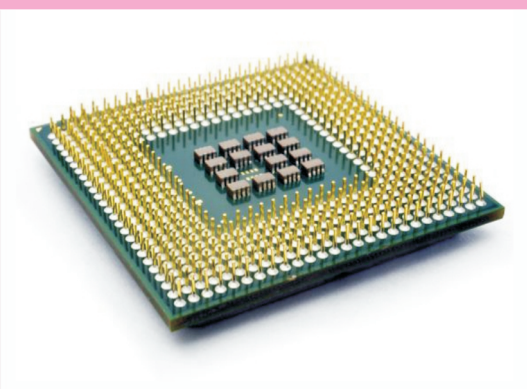
COMPUTER GROUP | SEMESTER - IV | DIPLOMA IN ENGINEERING AND TECHNOLOGY

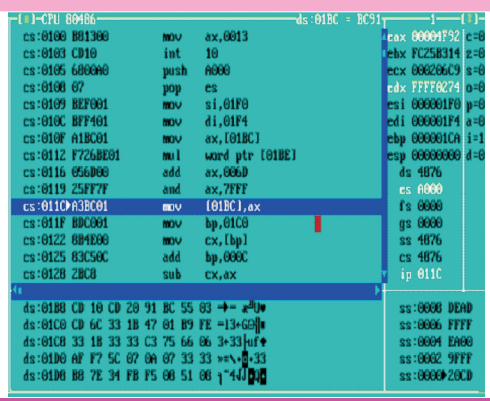
A LABORATORY MANUAL

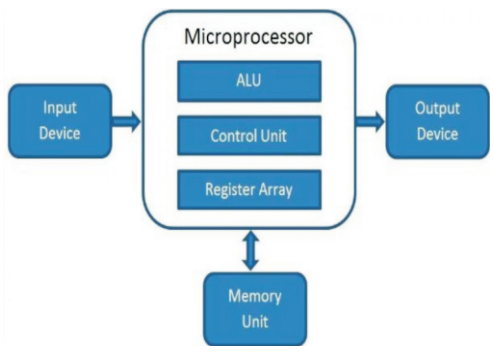
FOR

MICROPROCESSOR

(22415)







MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION, MUMBAI

(Autonomous) (ISO 9001 : 2015) (ISO / IEC 27001 : 2013)

VISION

To ensure that the Diploma level Technical Education constantly matches the latest requirements of technology and industry and includes the all-round personal development of students including social concerns and to become globally competitive, technology led organization.

MISSION

To provide high quality technical and managerial manpower, information and consultancy services to the industry and community to enable the industry and community to face the changing technological and environmental challenges.

QUALITY POLICY

We, at MSBTE are committed to offer the best in class academic services to the students and institutes to enhance the delight of industry and society. This will be achieved through continual improvement in management practices adopted in the process of curriculum design, development, implementation, evaluation and monitoring system along with adequate faculty development programmes.

CORE VALUES

MSBTE believes in the followings:

- Education industry produces live products.
- Market requirements do not wait for curriculum changes.
- Question paper is the reflector of academic standards of educational organization.
- Well designed curriculum needs effective implementation too.
- Competency based curriculum is the backbone of need based program.
- Technical skills do need support of life skills.
- Best teachers are the national assets.
- Effective teaching learning process is impossible without learning resources.

A Laboratory Manual
for

Microprocessor

(22415)

Semester – IV

(CO, CM, CW)

Maharashtra State
Board of Technical Education, Mumbai
(Autonomous) (ISO 9001:2015) (ISO/IEC 27001:2013)

Maharashtra State Board of Technical Education,
(Autonomous) (ISO 9001 : 2015) (ISO/IEC 27001 : 2013)
4th Floor, Government Polytechnic Building, 49, Kherwadi,
Bandra (East), Mumbai - 400051.
(Printed on November 2018)

Maharashtra State
Board of Technical Education

Certificate

This is to certify that Mr. / Ms. …….
Roll No…….of Fourth Semester of Diploma in ……. of Institute ……..
(Code……..) has attained predefined practical outcomes (PROs) satisfactorily in course **Microprocessor (22415)** for the academic year 20…….to 20…..... as prescribed in the curriculum.

Place ……. Enrollment No……
Date: ..... Exam Seat No. …….....

Course Teacher **Head of the Department** **Principal**

Seal of the Institute

Preface

The primary focus of any engineering laboratory/field work in the technical education system is to develop the much needed industry relevant competencies and skills. With this in view, MSBTE embarked on this innovative 'I' Scheme curricula for engineering Diploma programmes with outcome-based education as the focus and accordingly, relatively large amount of time is allotted for the practical work. This displays the great importance of laboratory work making each teacher, instructor and student to realize that every minute of the laboratory time need to be effectively utilized to develop these outcomes, rather than doing other mundane activities. Therefore, for the successful implementation of this outcome-based curriculum, every practical has been designed to serve as a ***'vehicle'*** to develop this industry identified competency in every student. The practical skills are difficult to develop through 'chalk and duster' activity in the classroom situation. Accordingly, the 'I' scheme laboratory manual development team designed the practical's to ***focus*** on ***outcomes***, rather than the traditional age old practice of conducting practical's to 'verify the theory' (which may become a byproduct along the way).

This laboratory manual is designed to help all stakeholders, especially the students, teachers and instructors to develop in the student the pre-determined outcomes. It is expected from each student that at least a day in advance, they have to thoroughly read the concerned practical procedure that they will do the next day and understand minimum theoretical background associated with the practical. Every practical in this manual begins by identifying the competency, industry relevant skills, course outcomes and practical outcomes which serve as a key focal point for doing the practical. Students will then become aware about the skills they will achieve through procedure shown there and necessary precautions to be taken, which will help them to apply in solving real-world problems in their professional life.

This manual also provides guidelines to teachers and instructors to effectively facilitate student-centered lab activities through each practical exercise by arranging and managing necessary resources in order that the students follow the procedures and precautions systematically ensuring the achievement of outcomes in the students.

A microprocessor is a general-purpose system. Several specialized processing devices have followed from the technology: They are built using digital logic. Thousands of items that were traditionally not computer-related include microprocessors. These include large and small household appliances, cars (and their accessory equipment units), car keys, tools and test instruments, toys etc. A microprocessor control program can be easily tailored to different needs of a product line, allowing upgrades in performance with minimal redesign of the product. Students will be able to write assembly language code and also write code to extend knowledge for optimizing critical sections of high level programs, implement loops in higher level at microprocessor level with the help of jump instructions, use the microprocessor at the lower level to receive input from the keyboards and the mice (through the interrupts). Students

will learn how a machine interprets instructions at low level, why memory segmentation in a process came into existence.

Although all care has been taken to check for mistakes in this laboratory manual, yet it is impossible to claim perfection especially as this is the first edition. Any such errors and suggestions for improvement can be brought to our notice and are highly welcome.

Programme Outcomes (POs) to be achieved through Practicals of this Course

Following programme outcomes are expected to be achieved significantly out of the ten programme outcomes and Computer Engineering programme specific outcomes through the practical's of the course on **Microprocessor.**

PO 1. Basic knowledge: Apply knowledge of basic mathematics, sciences and basic engineering to solve the broad-based Electronics related problems.

PO 2. Discipline knowledge: Apply Computer Programming knowledge to solve broad-based Electronics related problems.

PO 3. Experiments and practice: Plan to perform experiments and practices to use the results to solve broad-based Electronics related problems.

PO 4. Engineering tools: Apply relevant Computer programming / electrical technologies and tools with an understanding of the limitations.

PO 8. Individual and teamwork: Function effectively as a leader and team member in diverse/ multidisciplinary teams.

PO 9. Communication: Communicate effectively in oral and written form.

PO 10. Life-long learning: Engage in independent and life-long learning activities in the context of technological changes also in the Electronics engineering and allied industry.

Practical- Course Outcome matrix

Course Outcomes (COs)

a. Analyze the functional block diagram of 8086
b. Write assembly language program for the given problem.
c. Use instructions for different addressing modes.
d. Develop assembly language program using assembler.
e. Develop assembly language program procedures, macros and modular programming approach.

S. No.	Title of the Practical	CO a.	CO b.	CO c.	CO d.	CO e.
1.	Identify the various pins of the given microprocessor	√	-	-	-	-
2.	Use the assembly language programming tools and functions	√	√	-	-	-
3.	Use different addressing mode instruction in program (a) Write an Assembly Language Program (ALP) to add two given 8 and 16 bit numbers. (b) Write an Assembly Language Program (ALP) to subtract two given 8 and 16 bit numbers.	√	√	√	√	-
4.	(a) Write an ALP to multiply two given 8 and 16 bit unsigned numbers. (b) Write an ALP to multiply two given 8 and 16 bit signed numbers.	√	√	√	√	-
5.	(a) Write an ALP to perform block transfer data using string instructions (b) Write an ALP to perform block transfer data without using string instructions.	√	√	√	√	-
6.	(a) Write an ALP to compare two strings without using string instructions. (b) Write an ALP to compare two strings using string instructions	√	√	√	√	-
7.	(a) Write an ALP to divide two unsigned numbers (b) Write an ALP to divide two signed numbers	√	√	√	√	-
8.	Write an ALP to add, subtract, multiply, divide two BCD numbers.	√	√	√	√	-
9.	(a) Write an ALP to find sum of series of Hexadecimal Numbers. (b) Write an ALP to find sum of series of BCD numbers.	√	√	√	√	-

10.	(a) Write an ALP to find smallest number from array of n numbers. (b) Write an ALP to find largest number from array of n numbers.	√	√	√	√	-
11.	(a) Write an ALP to arrange numbers in array in ascending order. (b) Write an ALP to arrange numbers in array in descending order.	√	√	√	√	-
12.	(a) Write an ALP to arrange string in reverse order (b) Write an ALP to find string length. (c) Write an ALP to concatenation of two strings.	√	√	√	√	-
13.	(a) Write an ALP to check a given number is ODD or EVEN. (b) Write an ALP to count ODD and/or EVEN numbers in array.	√	√	√	√	-
14.	(a) Write an ALP to check a given number is POSITIVE or NEGATIVE (b)Write an ALP to count POSITIVE and/or NEGATIVE numbers in array.	√	√	√	√	-
15.	(a) Write an ALP to count number of '1' in a given number (b) Write an ALP to count number of '0' in a given number	√	√	√	√	-
16.	An assembly language program using procedures (a) Write an ALP for addition, subtraction, multiplication and division. (b) Write an ALP using procedure to solve equation such as Z = (A+B)*(C+D)	√	√	√	√	√
17.	An assembly language program using macros. (a) Write an ALP for addition, subtraction, multiplication and division. (b) Write an ALP using MACRO to solve equation such as Z = (A+B)*(C+D)	√	√	√	√	√

List of Industry Relevant Skills

The following industry relevant skills or the competency are expected to be developed in you by undertaking the practicals of this laboratory manual.

1. Analyze the functional block diagram of 8086 or x86 based processor
2. Develop an assembly language program using assembler for given problem
3. Use procedures and macros in assembly language programs.

Brief Guidelines to Teachers

Hints regarding strategies to be used:-

1. Teacher shall explain prior concepts to the students before starting each experiment.
2. For practical's requiring tools to be used, teacher should provide the demonstration of the practical emphasizing the skills, which the student should achieve.
3. Involve students in the activities during the conduct of each experiment.
4. Teachers should give opportunity to students for hands-on after the demonstration.
5. Assess the skill achievement of the students and COs of each unit.
6. Teacher is expected to share the skills and competencies to be developed in the students.
7. Teacher should ensure that the respective skills and competencies are developed in the students after the completion of the practical exercise.
8. Teacher may provide additional knowledge and skills to the students even though that may not be covered in the manual but are expected from the students by the industries.
9. Teacher may suggest the students to refer additional related literature of the reference books/websites/seminar proceedings etc.
10. During assessment teacher is expected to ask questions to the students to tap their knowledge and skill related to that practical.

Instructions for Students

1. Students shall listen carefully the lecture given by teacher about importance of subject, learning structure, course outcomes.
2. Students shall organize the work in the group of two or three members and make a record of all observations.
3. Students shall understand the purpose of experiment and its practical implementation.
4. Students shall write the answers of the questions during practical.
5. Student should feel free to discuss any difficulty faced during the conduct of practical.
6. Students shall develop maintenance skills as expected by the industries.
7. Student shall attempt to develop related hands on skills and gain confidence.
8. Students shall refer technical magazines; websites related to the scope of the subjects and update their knowledge and skills.
9. Students shall develop self-learning techniques.
10. Students should develop habit to submit the write-ups on the scheduled dates and time.

Content Page
List of Practicals and Progressive Assessment Sheet

Sr. No.	Title of the practical	Page No.	Date of performance	Date of submission	Assessment marks (50)	Dated sign. of teacher	Remarks (if any)
1.	Identify the various pins of the given microprocessor	1					
2.	Use the assembly language programming tools and functions	6					
3.	Use different addressing mode instruction in program (a) Write an Assembly Language Program (ALP) to add two given 8 and 16 bit numbers. (b) Write an Assembly Language Program (ALP) to subtract two given 8 and 16 bit numbers.	12					
4.	(a) Write an ALP to multiply two given 8 and 16 bit unsigned numbers. (b) Write an ALP to multiply two given 8 and 16 bit signed numbers.	20					
5.	(a) Write an ALP to perform block transfer data using string instructions (b) Write an ALP to perform block transfer data without using string instructions.	27					
6.	(a) Write an ALP to compare two strings without using string instructions. (b) Write an ALP to compare two strings using string instructions	33					
7.	(a) Write an ALP to divide two unsigned numbers (b) Write an ALP to divide two signed numbers	39					
8.	Write an ALP to add, subtract, multiply, divide two BCD numbers.	47					
9.	(a) Write an ALP to find sum of series of Hexadecimal Numbers. (b) Write an ALP to find sum	54					

Sr. No.	Title of the practical	Page No.	Date of performance	Date of submission	Assessment marks (50)	Dated sign. of teacher	Remarks (if any)
	of series of BCD numbers.						
10.	Write an ALP to find smallest number from array of n numbers. Write an ALP to find largest number from array of n numbers.	61					
11.	(a) Write an ALP to arrange numbers in array in ascending order. (b) Write an ALP to arrange numbers in array in descending order.	68					
12.	(a) Write an ALP to arrange string in reverse order (b) Write an ALP to find string length. (c) Write an ALP to concatenation of two strings.	74					
13.	(a) Write an ALP to check a given number is ODD or EVEN. (b) Write an ALP to count ODD and/or EVEN numbers in array.	80					
14.	(a) Write an ALP to check a given number is POSITIVE or NEGATIVE (b) Write an ALP to count POSITIVE and/or NEGATIVE numbers in array.	86					
15.	(a) Write an ALP to count number of '1' in a given number (b) Write an ALP to count number of '0' in a given number	92					
16.	An assembly language program using procedures (a) Write an ALP for addition, subtraction, multiplication	98					

Sr. No.	Title of the practical	Page No.	Date of performance	Date of submission	Assessment marks (50)	Dated sign. of teacher	Remarks (if any)
	and division. (b) Write an ALP using procedure to solve equation such as Z = (A+B)*(C+D)						
17.	An assembly language program using macros. (a) Write an ALP for addition, subtraction, multiplication and division. (b) Write an ALP using MACRO to solve equation such as Z = (A+B)*(C+D)	105					
Total							
Total Marks (Scaled to 50)							

- **To be transferred to Proforma of CIAAN-2017**.

Practical No. 1: Identify the various pins of the 8086 microprocessor

I Practical Significance

8086 was the first 16-bit microprocessor available in 40-pin DIP (Dual Inline Package) chip. Microprocessors are applicable to a wide range of information processing tasks, ranging from general computing to real time monitoring systems. Hence, students will be able to identify the various pins and their functions.

II Relevant Program Outcomes (POs)

PO1- Basic knowledge

PO2- Discipline knowledge

PO10- Life-long learning

III Competency and Practical Skills

"*Develop assembly language program using 8086*"

This practical is expected to develop the following skills

1. Identify the function of given pins.
2. Identify the function of pins in maximum and minimum mode.

IV Relevant Course Outcome(s)

a. Analyze the functional block diagram of 8086.

V Practical Outcomes

a. Identify the various pins of the given microprocessor.

VI Relevant Affective Domain related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices

VII Minimum Theoretical Background

The 8086 is a 16-bit HMOS microprocessor available in a 40 pin IC and works with 5 volts DC supply. Electronic circuitry of 8086 consists of 29000 transistors implemented in N-channel, silicon gate technology and offered in three versions i.e. 8086(5 MHz), 8086-2(8 MHz) and 8086-1(10 MHz). The 8086 microprocessor is no longer used, but the concept of its principles and structure is very much useful for understanding functioning of other advanced Intel microprocessors.

The 40pins of 8086 can be categorized in different group as given below.

a. Address bus
b. Data bus
c. Control bus

The 8086 has 20-bit address bus, where 16 pins AD_0-AD_{15} are multiplexed address /data bus and remaining 4 pins are multiplexed status and address lines A_{16}/S_3-A_{19}/S_6. The 8086 operates in two operating modes i.e. minimum and maximum mode. The function of control pins changes according to operating modes. In minimum mode, 8086 generates control signal such as HOLD, HLDA, ALE, $\overline{RD}$, $\overline{WR}$, M/$\overline{IO}$, DT/$\overline{R}$,

$\overline{DEN}$ and $\overline{INTA}$. In maximum mode, 8086 generates control signal such as QS_1, QS_2, $\overline{S}_0$ $\overline{S}_1$ $\overline{S}_2$, $\overline{LOCK}$, $\overline{RQ}$ / $\overline{GT}_0$, $\overline{RQ}$ / $\overline{GT}_1$.

VIII Work Situation

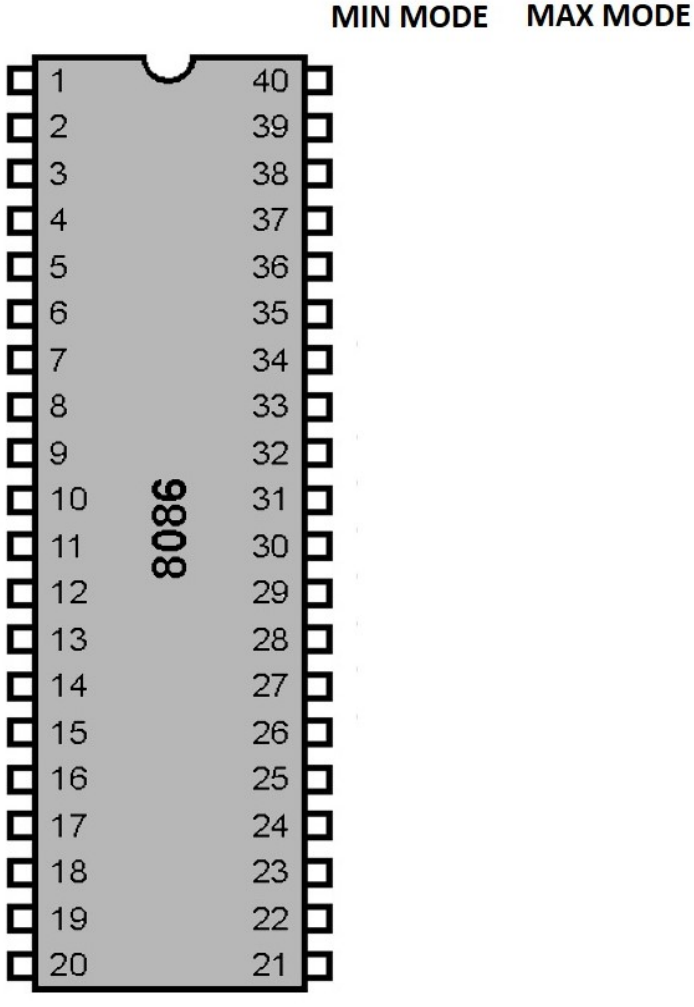
a. Faculty will demonstrate the use and function of pins of 8086 using chart or presentations.

IX Resources required (Additional)

S. No.	Instrument /Object	Specification	Quantity	Remarks
1.	Chart	8086 Microprocessor pin diagram and Block diagram	1 No.	Whichever is available

X Observations

Label pins of given diagram of 8086 in respective modes



XVI. Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Name the interrupts of 8086.

.....
.....
.....
.....

2. State DMA controller signals of 8086

.....
.....
.....
.....

3. $\overline{RD}$ Signal is active …….(low/high).

.....
.....
.....
.....

XVII. Exercise

Note: Faculty must ensure that every group of students use different pins.

(Use blank space for answers or attach more pages if needed)

1. Draw the labelled block diagram of 8086.
2. State the use of status signals in maximum mode.

(Space for answers)

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

.....

.....

.....

.....

.....

XVIII. References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX. Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1.	Pin Identification	35%
2.	Pin function	35%
Product related (15 Marks)		**30%**
3.	Practical related questions	10%
4.	Completion and submission of practical in time	10%
5.	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. …….
2. ……....
3. …….
4. …….

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 2: Use Assembly language Programming Tools and Functions

I Practical Significance

Assembly language is used to write program in the form of mnemonics that is the short form of operations i.e. for addition *add* and operands, which may be registers or memory location. In operating system, system program is normally written in assembly language using tools like assembler, linker and for debugging debugger. Hence, students will be able to use various such tools required for assembly language programming.

II Relevant Program Outcomes (POs)

PO2- Discipline knowledge
PO3- Experiments and practice
PO4- Engineering Tools

III Competency and Practical Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills

1. Use editor to write assembly language program *filename.asm* file
2. Use assembler and linker to create *filename.exe* file
3. Use debugger in single step mode to locate/trace the errors and correcting the errors

IV Relevant Course Outcome(s)

a. Use assembly language programming tools.

V Practical Outcomes

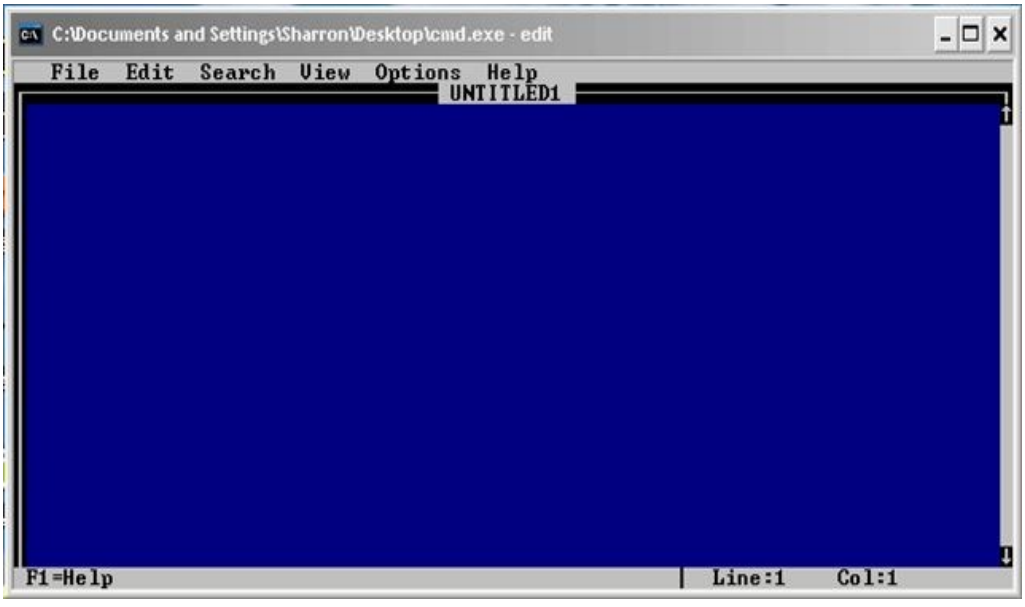
a. Use the assembly language programming tools and functions.

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

a. **Editor:** An editor is a program, which is used to construct assembly language program in appropriate format so that the assembler will translate it correctly to machine language. Therefore, you can type your program called as source program using editor. The **DOS** based editor such as **EDIT** can be used to type your program.

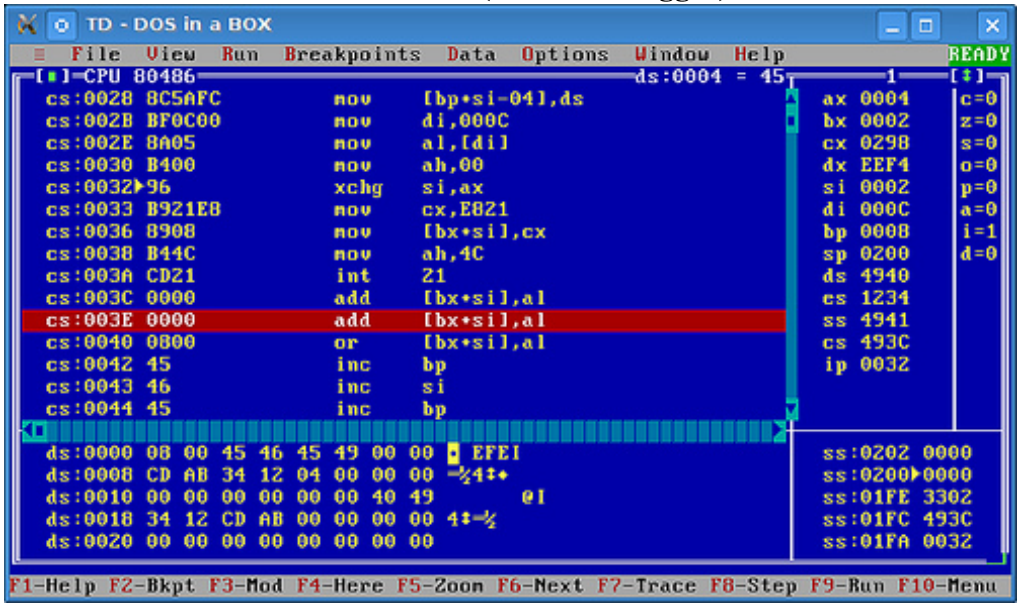


b. **Assembler:** An assembler is a program that translate assembly language program to the appropriate binary code for each instruction in program i.e. machine code and generate the file called as object file with extension **.obj**. Assembler may be TASM Borland's Turbo Assembler and MASM Microsoft Macro Assembler etc.

c. **Linker:** A linker is a program that combines, if requested, more than one separately assembled program module into one executable program and generate .exe module, and initializes it with special instructions to enable its subsequent loading the execution. Linker may be **TLINK** Borland's Turbo Linker and **LINK** Microsoft's Linker

d. **Debugger:** Debugger is a program is used to execute program in single step mode under the control of the user. The process of locating and correcting errors using a debugger is known as debugging. Some examples of debugger are DOS **Debug** command, Borland's turbo Debugger **TD**, Microsoft Debugger known as Code View **CV** etc.

View of TD (Turbo Debugger)



View of DOS Debug

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

S. No.	Instrument /Object	Specification	Quantity	Remarks
1.	Desktop PC	Pentium IV or above with Keyboard, Mouse, Monitor	1 No. /Group	Whichever is available
2.	Editor	MS-DOS EDIT or Notepad	1 No. /Group	Whichever is available
3.	Assembler	MASM or TASM	1 No. /Group	Whichever is available
4.	Linker	LINK or TLINK	1 No. /Group	Whichever is available
5.	Debugger	Debug or TD	1 No. /Group	Whichever is available

X Precautions to be followed

1. Handle computer system and peripherals with care.
2. Follow safety practices.

XI Procedure

a. Install DOSBOX TASM 1.4 or above.
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.

e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....
.....
.....
.....

XIII Observations

1) Observe and write the contents of Register using debugger TD or Debug

Table 1: Contents of Registers

Types	Registers		Flag Register		
General Purpose registers	AX		Carry Flag	CF	
	BX		Zero Flag	ZF	
	CX		Sign Flag	SF	
	DX		Overflow Flag	OF	
Index Register	SI		Parity Flag	PF	
	DI		Auxiliary Carry Flag	AF	
Base Pointer	BP		Interrupt Flag	IF	
Stack Pointer	SP		Direction Flag	DF	
Segment Register	DS				
	ES				
	SS				
	CS				
Instruction register	IP				

2) Observe and write the contents of memory location in Code Segment using debugger TD or Debug

Table 2: Contents of memory location in Code Segment

Address	Contents	Address	Contents
CS:0000		CS:0008	
CS:0001		CS:0009	
CS:0002		CS:000A	
CS:0003		CS:000B	
CS:0004		CS:000C	
CS:0005		CS:000D	

CS:0006		CS:000E	
CS:0007		CS:000F	

3) Observe and write the contents of memory location in Data Segment using debugger TD or Debug

Table 3: Contents of memory location in Data Segment

Address	Contents	Address	Contents
DS:0000		DS:0008	
DS:0001		DS:0009	
DS:0002		DS:000A	
DS:0003		DS:000B	
DS:0004		DS:000C	
DS:0005		DS:000D	
DS:0006		DS:000E	
DS:0007		DS:000F	

XIV Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write the assembly language tools used in your lab in Table 4.

Table 4: Tools Used

Sr. No.	Tools Used	Name of Tool	Version
1	Editor		
2	Assembler		
3	Linker		
4	Debugger		

2. List the files extensions that are created by the Assembler used.

.....
.....
.....
.....
.....
.....
.....

3. List the files extensions that are created by the Linker used.

.....
.....
.....
.....
.....
.....
.....

XV References / Suggestions for further Reading

a. https://www.elprocus.com/8086-assembly-language-programs-explanation/
b. http://mysc.altervista.org/beginners-guide-8086/
c. https://www.tutorialspoint.com/assembly_programming/

XVI. Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program	20%
2	Use assembler and linker to create .exe file	10%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Observations	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……
2. ……...
3. ……
4. ……

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 3: Write an assembly language program to perform addition and subtraction of two 8 and 16-bit numbers

I Practical Significance

In high-level language programming, the mathematical sign for addition (+) and subtraction (-) can be used to perform arithmetic operation. However, in assembly language the mnemonics are used to perform arithmetic operation such ADD/ADC for addition, SUB/SBB for subtraction. In operating system, system programs such as device drivers, memory management modules are normally written in assembly language where addition and subtraction is required. Hence, students will be able to use mnemonics of the instructions in assembly language program

.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge:

PO 2- Discipline knowledge

PO 3- Experiments and practice

PO 4- Engineering tools

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills

1. Use editor to create assembly language program i.e. .asm file
2. Use assembler and linker to create .exe file
3. Use debugger in single step mode to locate/trace the errors and correcting the errors

IV Relevant Course Outcome(s)

a. Write assembly language program for given problem.

V Practical Outcomes

a. Write an Assembly Language Program (ALP) to add two given 8 and 16 bit numbers.
b. Write an Assembly Language Program (ALP) to subtract two given 8 and 16-bit numbers.

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

ADD / ADC destination, source

The ADD instruction adds a number from source to a number from destination. The ADC instruction adds the carry flag into the result of addition. The source may be an immediate number, a register, or a memory location as specified by any 24 addressing modes. The destination may be a register or a memory. The source and destination must be of the same type and cannot both be memory locations. Destination should not be an immediate number.

Flag affected: OF, CF, PF, AF, SF, ZF.

Syntax & Operation:

2. **ADD <DEST> ,<SRC>**
   Destination ← destination + source
3. **ADC <DEST> ,<SRC>**
   Destination ← destination + source + CF

SUB / SBB destination, source

The SUB instruction is used to subtract the data in source from the data in destination and the stores result in destination. The SBB instruction is used to subtract the source operand and the barrow [CF], which may reflect from the result of the previous operations, from the destination operand, and the result, is stored in destination operand. Source must be a register or memory location or immediate data and the destination must be a register or a memory location. The destination operands should not be an immediate data and the source and destination both should not be memory operands.

Flag affected: OF, CF, PF, AF, SF, and ZF.

Syntax & Operation:

1. **SUB <DEST> ,<SRC>**
   Destination ← destination - source
2. **SBB <DEST> ,<SRC>**
   Destination ← destination - source – CF

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

1. Handle computer system and peripherals with care.
2. Shut down PC properly

XI Procedure

a. Write algorithm and draw flowchart for given program (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key

d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....
.....
.....
.....

XIII Observations:

1) Observe and write the contents of Register using debugger TD or Debug after the execution of program.

Table 1: Contents of Registers after the Addition of Two 16-bit numbers

Registers			Flag Register		
	After	Before			
AX			Carry Flag	CF	
BX			Zero Flag	ZF	
CX			Sign Flag	SF	
DX			Overflow Flag	OF	
SI			Parity Flag	PF	
DI			Auxiliary Carry Flag	AF	
BP			Interrupt Flag	IF	
SP			Direction Flag	DF	
DS					
ES					
SS					
CS					
IP					

2) Observe and write the contents of memory location in Code Segment using debugger TD or Debug after the execution of program.

Table 2: Contents of memory location in Code Segment

Address	Contents	Address	Contents
CS:0000		CS:0008	
CS:0001		CS:0009	
CS:0002		CS:000A	
CS:0003		CS:000B	
CS:0004		CS:000C	
CS:0005		CS:000D	
CS:0006		CS:000E	
CS:0007		CS:000F	

3) Observe and write the contents of memory location in Data Segment using debugger TD or Debug

Table 3: Contents of memory location in Data Segment

Address	Contents	Address	Contents
DS:0000		DS:0008	
DS:0001		DS:0009	
DS:0002		DS:000A	
DS:0003		DS:000B	
DS:0004		DS:000C	
DS:0005		DS:000D	
DS:0006		DS:000E	
DS:0007		DS:000F	

XIV Program Code with comments

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XV Results (Output of the Program)

(***Note: Write an Output of program assigned by teacher***)

.....

.....

.....

.....

XVI Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write the command line used to create assembly language program for addition and subtraction.

.....

.....

.....

.....

2. Write the command line used to generate .exe file of assembly language program for addition and subtraction.

.....

.....

.....

.....

3. Write assembler directives used in the program.

.....

.....

.....

.....

XVII. Exercise

Note: Faculty must ensure that every group of students use different input value.

(Use blank space provided for answers or attach more pages if needed)

1. Write the content of AL register and status of flags after execution of following code

   MOV AL, 99

   ADD AL, 01

2. Write an ALP for addition of two 32 bit numbers.

(Space for answers)

XVIII. References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX. Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1.	Use editor to create assembly language program file	10%
2.	Use assembler and linker to create .exe file	20%
3.	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4.	Practical related questions	10%
5.	Completion and submission of practical in time	10%
6.	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……...
2. ……...
3. ……...
4. ……...

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 4: Write an ALP to multiply two given 8 and 16 bit unsigned and signed numbers.

I Practical Significance

In high-level language programming, the mathematical sign for multiplication (×) is used to perform arithmetic operation. However, in assembly language the mnemonics are used to perform arithmetic operation such MUL for unsigned multiplication and IMUL for signed multiplication. In operating system, system program such as device drivers, memory management modules are normally written in assembly language where addition and subtraction is required. Hence, students will be able to use MUL instruction for unsigned and IMUL instruction for signed numbers in assembly language program.

.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge:

PO 2- Discipline knowledge

PO 3- Experiments and practice

PO 4- Engineering tools

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills

1. Use editor to create assembly language program *filename.asm* file
2. Use assembler and linker to create *filename.exe* file
3. Use debugger in single step mode to locate/trace the errors and correcting the errors

IV Relevant Course Outcome(s)

a. Use instructions for different addressing mode.

V Practical Outcomes

a. Write an ALP to multiply two given 8 and 16-bit unsigned numbers.
b. Write an ALP to multiply two given 8 and 16-bit signed numbers.

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

MUL source

MUL is used to multiply an **unsigned** byte/word from source with an **unsigned** byte/word in the AL/AX register. The source must be any register or a memory location. When a byte is multiplied with the byte in AL, then the result is stored in AX because the result of multiplication is maximum 16 bits. When a word is multiplied with the word in AX, then the MSW of result is stored in DX and LSW of result in AX register because the result of multiplication is maximum 32-bits. If the MSB or MSW of the result is zero, then CF and OF both will be set.

Flag affected by an instruction: OF, CF and PF, AF, SF, ZF are undefined.

Operation

(a) If source is byte then AX ← AL × unsigned 8 bit source.

(b) If source is word then DX**:** AX ← AX × unsigned 16 bit source.

Examples

MUL DL	Multiply AL by DL, result in AX.
MUL BX	Multiply AX by BX, result in DX**:** AX.

IMUL source

IMUL instruction is used to multiply a **signed** byte/word from source with a **signed** byte/word in the AL/AX register. The source must be a register or a memory location. When a byte is multiplied with the byte in AL, then the result is stored in AX because the result of multiplication is maximum 16 bits. When a word is multiplied with the word in AX, then the MSB result is stored in DX and LSB in AX register because the result of multiplication is maximum 32-bits. If the magnitude of the product does not requires all the bits of the destination, the unused bits are filled with the copy of the sign bit.

Flag affected by instruction: OF, CF and PF, AF, SF, ZF are undefined.

Operation

(a) If source is byte then AX ← AL × signed 8 bit source.

(b) If source is word then DX**:** AX ← AX × signed 16 bit source.

Examples

IMUL DL Multiply AL by DL, result in AX.

IMUL BX Multiply AX by BX, result in DX**:** AX.

Example of multiplication of signed byte with signed word.

MOV BX, multiplier	Load signed word multiplier in BX.
MOV AL, multiplicand	Load signed byte multiplicand in AL.
CBW	Convert Byte to Word i.e. extends sign of AL into AH.
IMUL BX	Word multiplies, result in DX**:** AX.

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

1. Handle computer system and peripherals with care
2. Shut down PC properly

XI Procedure

a. Write algorithm and draw flow-chart for given program (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....
.....
.....
.....

XIII Observations

Observe and write the contents of Register using debugger TD or Debug after the execution of program.

Table 1: Contents of Registers

Registers			Flag Register		
	After	Before			
AX			Carry Flag	CF	
BX			Zero Flag	ZF	
CX			Sign Flag	SF	
DX			Overflow Flag	OF	
SI			Parity Flag	PF	
DI			Auxiliary Carry Flag	AF	
BP			Interrupt Flag	IF	
SP			Direction Flag	DF	
DS					
ES					
SS					
CS					
IP					

Table 2: Contents of memory location in Data Segment

Address	Contents	Address	Contents
DS:0000		DS:0008	
DS:0001		DS:0009	
DS:0002		DS:000A	
DS:0003		DS:000B	
DS:0004		DS:000C	
DS:0005		DS:000D	
DS:0006		DS:000E	
DS:0007		DS:000F	

XIV Program Code with comments

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

XV Results (Output of program) (Note: Write an Output of program assigned by teacher)

.....
.....
.....
.....
.....
.....
.....
.....

XVI Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write the names of result registers of multiplication of 8/16-bits unsigned and signed numbers.

.....

.....
.....
.....

2. Which instruction you have used to extend the sign of 8-bit negative number for 8bit x 16-bit multiplication**.**

.....
.....
.....
.....

Exercise

(Use blank space provide for answers or attached more pages if needed)

1. Write the content of AX register after execution of following code.

```
MOV BL, -1
MOV AL, 1
IMUL BL
```

2. State the flag affected by IMUL instruction.
3. State the difference between MUL and IMUL.

(Space for answers)

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

.....

.....

.....

.....

.....

.....

XVII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XVIII Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program file	10%
2	Use assembler and linker to create .exe file	20%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……..
2. ……...
3. ……...
4. ……...

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 5: Write an ALP to perform block transfer operation

I Practical Significance

In operating system, system programs such as video device drivers, memory management modules are normally written in assembly language where memory block of video data is to be transferred from main memory to video memory continuously to display steady video on the screen. Hence, students will be able to use MOV and MOVS instruction for data transfer operation in assembly language program.

II Relevant Program Outcomes (POs)

PO 1- Basic knowledge

PO 2- Discipline knowledge

PO 3- Experiments and practice

PO 4- Engineering tools practice

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills

1. Use editor to create assembly language program *filename.asm* file
2. Use assembler and linker to create *filename.exe* file
3. Use debugger in single step mode to locate/trace the errors and correcting the errors

IV Relevant Course Outcome(s)

a. Use instructions for different addressing mode.

V Practical Outcomes

a. Write an ALP to perform block transfer data using string instructions
b. Write an ALP to perform block transfer data without using string instructions.

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

Block transfer operation is nothing but it is transferring of block of date from source memory locations to destination memory locations. Counter is required to perform block transfer operation which is equal to length of data block. On each transfer of data from source to destination counter must be decremented by one and memory pointer must be incremented by one or two depending on byte or word transfer. This process is repeated till the counter becomes zero.

Before Block Transfer

Source Block			Destination block	
Memory Location	Data		Memory Location	Data
DS:0000H	56H		DS:0005H	15H
DS:0001H	7BH		DS:0006H	49H
DS:0002H	62H		DS:0007H	F7H
DS:0003H	23H		DS:0008H	C9H
DS:0004H	AAH		DS:0009H	55H

After Block Transfer

Source Block			Destination block	
Memory Location	Data		Memory Location	Data
DS:0000H	56H		DS:0005H	56H
DS:0001H	7BH		DS:0006H	7BH
DS:0002H	62H		DS:0007H	62H
DS:0003H	23H		DS:0008H	23H
DS:0004H	AAH		DS:0009H	AAH

If the number of bytes or words in block is 5, then initialize this as byte counter or word counter in CX register. Then two memory pointers are required to point source block and destination block, hence use SI and DI registers respectively as source and destination memory pointers. The block can be transfer from source to destination either using string instruction i.e. MOVS/MOVSB/MOVSW or without using string instruction such as simple MOV instruction. For MOVSB/MOVSW instruction, the default memory pointer for source and destination blocks are DS:SI and ES: DI respectively. Two arrays must be declared in the array where in one array contains actual numbers and another array must be empty. To declare empty array, we can use **DUP** directive. For example, 5 dup (0) statements allocates five memory location and initialize them with 0.

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

1. Handle computer system and peripheral with care.
2. Shut down PC properly

XI Procedure

a. Write algorithm and draw flowchart for given program. (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....
.....
.....
.....

XIII Observations

1) Table 1: Observe and write the contents of Source and destination block memory location before transfer

Source Memory Block		Destination Memory Block	
Address	**Contents**	**Address**	**Contents**
DS:0000		DS:0005	
DS:0001		DS:0006	
DS:0002		DS:0007	
DS:0003		DS:0008	
DS:0004		DS:0009	

2) Table 2: Observe and write the contents of Source and destination block memory location after transfer

Source Memory Block		Destination Memory Block	
Address	**Contents**	**Address**	**Contents**
DS:0000		DS:0005	
DS:0001		DS:0006	
DS:0002		DS:0007	
DS:0003		DS:0008	
DS:0004		DS:0009	

XIV Program Code with comments

.....
.....
.....
.....
.....
.....
.....
.....
.....

XV Results (Output of Program) (Note: Write an Output of program assigned by teacher)

.....
.....
.....
.....

XVI Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write the instructions you have used to initialize memory pointer for source and destination block of data.

.....
.....
.....
.....

2. State the name of register which is used as a counte**r.**

.....
.....
.....
.....

XVII Exercise (Any One)

(Use blank space provided for answers or attach more pages if needed)

1. Write an ALP to perform block transfer in reverse order**.**
2. Write an ALP to perform block transfer of overlapping block**.**

(Space for answers)

.....
.....
.....
.....

.....

.....

.....

.....

.....

.....

.....

XVIII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program file	10%
2	Use assembler and linker to create .exe file	20%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……...
2. ……...
3. ……...
4. ……...

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 6: Write an ALP to compare two Strings

I Practical Significance

In operating system, filename normally called as ASCIIZ (ASCII string ending with zero) strings are compared during the copy operation. If source filename is already existing, then "filename already exists, overwrite it" message is displayed on screen. Hence, students will be able to use MOV and CMP or CMPS instruction to compare two strings in assembly language program.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge

PO 2- Discipline knowledge

PO 3- Experiments and practice

PO 4- Engineering tools practice

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills

1. Use editor to create assembly language program *filename.asm* file
2. Use assembler and linker to create *filename.exe* file
3. Use debugger in single step mode to locate/trace the errors and correcting the errors

IV Relevant Course Outcome(s)

a. Use instructions for different addressing mode.

V Practical Outcomes

a. Write an ALP to compare two strings without using string instructions.
b. Write an ALP to compare two strings using string instructions.

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

The string consists of numbers or characters. In assembly, the string must be declare in single quotes i.e. ' ' and end with '$' sign. The data type of the string is always byte type because assembler store ASCII value of each and every character of string in memory, as ASCII codes are 8 bit.

For Example:

```
name  db    'MSBTE$'
```

Assembler stores string characters in memory at consecutive memory locations. Hence to perform any string related operation such as comparison, length, reverse etc., the memory pointer and byte counter is required as same as used in block transfer program. For comparison, the string instruction is CMPSB as data type of string is byte. Simple CMP instruction also can be used to compare two strings. First, we will have to find the length of both strings, if lengths are equal, then strings may be equal

or unequal. Then, we will have to compare both strings character by character to find equality.

String stored at Consecutive Memory locations

Source Block	
Memory Location	Data
DS:0000H	M
DS:0001H	S
DS:0002H	B
DS:0003H	T
DS:0004H	E
DS:0005H	'$'

After the comparison of two string, we need DOS function 09H of interrupt 21H to display string such as "Strings are same$" or "Strings are not same$".
Function: 09H of INT 21H (Display Strings on Console)
Function Call with:

AH = 09H
DS:DX = Segment: Offset of string

Example:

```
MOV AH, 09H
MOV DX, offset STR
INT 21H
```

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

1. Handle computer System and peripheral with care.
2. Shut down PC properly

XI Procedure

a. Write an algorithm and draw flowchart of given program. (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key

d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....
.....
.....
.....

XIII Observations

Table 1: Observe and write the contents memory location of Source and destination strings

Source Memory Block		Destination Memory Block	
Address	**Contents**	**Address**	**Contents**
DS:0000		DS:0008	
DS:0001		DS:0009	
DS:0002		DS:000A	
DS:0003		DS:000B	
DS:0004		DS:000C	
DS:0005		DS:000D	
DS:0006		DS:000E	
DS:0007		DS:000F	

XIV Program Code with comments

.....
.....
.....
.....
.....
.....
.....
.....
.....

.....

.....

.....

.....

.....

.....

XV Results (Output of Program) *(Note: Write an Output of program assigned by teacher*)

.....

.....

.....

.....

XVI Practical Related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write the instructions you have used to initialize memory pointer for source and destination strings.

.....

.....

.....

.....

Write the string instruction used for string comparison in your program**.**

.....

.....

.....

.....

XVII Exercise

(Use blank space provided for answers or attach more pages if needed)

1. Write registers used as memory pointers for source and destination while using string compare instruction CMPS.
2. Write the flag which are used to know whether strings are equal or unequal.
3. State the use of REP prefix instruction.

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XVIII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program file	10%
2	Use assembler and linker to create .exe file	20%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……..
2. ……...
3. ……..
4. …….

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 7: Write an ALP to divide two 8 bit and two 16 bit unsigned and signed numbers.

I Practical Significance

In high-level language programming, the mathematical sign for division (/) is used to perform arithmetic operation. However, in assembly language the mnemonics are used to perform arithmetic operation such DIV for unsigned multiplication and IDIV for signed multiplication. In operating system, system program such as device drivers, memory management modules are normally written in assembly language where division operation is required. Hence, students will be able to used DIV instruction for unsigned and IDIV instruction for signed numbers the instructions in assembly language program.

.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge:

PO 2- Discipline knowledge

PO 3- Experiments and practice

PO 4- Engineering tools practice

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills

1. Use editor to create assembly language program *filename.asm* file
2. Use assembler and linker to create *filename.exe* file
3. Use debugger in single step mode to locate/trace the errors and correcting the errors

IV Relevant Course Outcome(s)

a. Use instructions for different addressing mode.

V Practical Outcomes

a. Write an ALP to divide two unsigned numbers
b. Write an ALP to divide two signed numbers

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

DIV source

DIV/IDIV instruction divides an **unsigned/signed** word by an **unsigned/signed** byte during 16/8 division, and to divide **unsigned/signed** double word i.e. 32-bits by an **unsigned/signed** word during 32/16 division. The word (dividend) must be in the AX register and a byte (divisor) may in any 8-bit register or memory location during the division of a word by a byte. After the division, 8 bit quotient will be stored in AL register and 8 bit remainder will stored in AH register

Flag affected: None and OF, CF, PF, AF, SF, ZF are undefined.

Operation

(a) If source is byte then

AL ← AL / unsigned 8 bit source (Quotient)

AH ← AL **MOD** unsigned 8 bit source. (Remainder)

(b) If source is word then

AX ← DX**:** AX / unsigned 16 bit source (Quotient)

DX←DX**:** AX **MOD** unsigned 16 bit source. (Remainder)

Examples

DIV BL ; Divide word in AX by byte in BL, quotient in AL and remainder in AH.

DIV NUM [BX]; Divide word in AX by byte in memory location pointer by [BX].

IDIV source

This instruction divides a **signed** word by a **signed** byte during 16/8 division, and to divide **signed** double word i.e. 32-bits by a **signed** word during 32/16 division.

During the division of a word by a byte, the word (dividend) must be in the AX register and a byte (divisor) may in any 8-bit register or memory location. After the division operation, 8-bit quotient will be available in AL register and 8-bit remainder will available in AH register During the division of double word by word, the dividend must be in DX: AX for double word or AX for word, but source of the divisor should be a word or byte register or a memory location.

When we want to divide a byte by a byte, we must first store dividend byte in AL and fill all bits in AH with sign bit of AL using CBW instruction. When we want to divide a word by a word, we must first store dividend word in AX and fill all bits in DX with sign bit of AX using CWD instruction.

Flag affected: None and OF, CF, PF, AF, SF, ZF are undefined.

Operation

(a) If source is byte then

AL ← AL / signed 8 bit source (Quotient)

AH ←AL **MOD** signed 8 bit source. (Remainder)

(b) If source is word then

AX ← DX**:** AX / signed 16 bit source (Quotient)

DX ← DX**:** AX **MOD** signed 16 bit source. (Remainder)

Examples

IDIV BL ; Divide a signed word in AX by a signed byte in BL, quotient in AL and remainder in AH.

IDIV NUM [BX] ;Divide a signed word in AX by a signed byte in memory location pointer by [BX].

Example of division of signed byte with signed byte.

```
MOV BL,     divisor Load signed byte divisor in    BL.
MOV AL,     dividend Load signed byte dividend in AL.
CBW         Extend sign of AL into AH.
IDIV BH     Byte division, remainder in AH and quotient in AL.
```

CBW (Convert Byte to Word)

This instruction copies the sign of byte in AL to all the bits in AH. AH is then said to be the sign extension of AL. The CBW operation must be done before performing division of a signed byte in the AL by another signed byte with IDIV instruction.

Operation

AH ← filled with 8^{th} bit of AL i.e. D_7

Example

AX = 00000000 10011011

CBW convert signed byte in AL to signed word in AX

AX = 11111111 10011011

CWD (Convert Word to Double word)

This instruction copies the sign bit of a word in AX to all the bits of the DX register. In other word, it extends the sign of AX into all of DX. The CWD operation must be done before performing division of a signed word in AX by another signed word with the IDIV instruction.

Operation

DX ← filled with 16^{th} bit of AX i.e. D_{15}.

Example

DX = 00000000 00000000

AX = 11110000 11000111.

CWD Convert signed word in AX to signed double word in DX AX.

Result after the execution of CWD.

DX = 11111111 11111111

AX = 11110000 11000111.

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

a. Handle computer System and peripheral with care.
b. Shut down PC properly

XI Procedure

a. Write an algorithm and draw flowchart of given program. (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....
.....
.....
.....

XIII Observations

1) Observe and write the contents of Register using debugger TD or Debug after the execution of program.

Table 1: Contents of Registers

Registers			Flag Register		
	After	Before			
AX			Carry Flag	CF	
BX			Zero Flag	ZF	
CX			Sign Flag	SF	
DX			Overflow Flag	OF	
SI			Parity Flag	PF	
DI			Auxiliary Carry Flag	AF	
BP			Interrupt Flag	IF	
SP			Direction Flag	DF	
DS					
ES					
SS					
CS					
IP					

2) Observe and write the contents of memory location in Data Segment

Table 2: Contents of memory location in Data Segment

Address	Contents	Address	Contents
DS:0000		DS:0008	
DS:0001		DS:0009	
DS:0002		DS:000A	
DS:0003		DS:000B	
DS:0004		DS:000C	
DS:0005		DS:000D	
DS:0006		DS:000E	
DS:0007		DS:000F	

XIV Program Code with comments

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XV Results (Output of Program) (Note: Write an Output of program assigned by teacher)

.....

.....

.....

.....

XVI Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write the result of division of signed numbers you have taken in program.

.....

.....

.....

.....

2. State the difference between DIV and IDIV instruction.

.....

.....

.....

.....

XVII Exercise (Any One)

(Use blank space provided for answers or attach more pages if needed)

1. Write an ALP to divide 8-bit signed number by 8-bit signed number.
2. Write an ALP to divide 16-bit signed number by 16-bit signed number.

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XVIII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program file	10%
2	Use assembler and linker to create .exe file	20%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……..
2. ……...
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 8: Write an ALP to perform arithmetic operation on BCD numbers

I Practical Significance

In high-level language programming, decimal numbers system is used to perform arithmetic operation. However, microprocessor performs all arithmetic operation on binary i.e. hexadecimal numbers. In assembly language program, special instructions are required to convert arithmetic operation result of decimal numbers to appropriate result in BCD format. Hence, students will be able to used DAA and DAS instruction to perform arithmetic operation on decimal (BCD) numbers in assembly language program.

.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge:

PO 2- Discipline knowledge

PO 3- Experiments and practice

PO 4- Engineering tools practice

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills

1. Use editor to create assembly language program *filename.asm* file
2. Use assembler and linker to create *filename.exe* file
3. Use debugger in single step mode to locate/trace the errors and correcting the errors

IV Relevant Course Outcome(s)

a. Develop assembly language program using assembler.

V Practical Outcomes

a. Write an ALP to add, subtract, multiply, divide two BCD numbers

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

DAA (Decimal adjust accumulator)

DAA instruction is used to convert the result of the addition of two packed BCD numbers into a packed BCD number. DAA only works on AL register. So, DAA instruction should be used after the ADD/ADC instruction. The ADD/ADC instruction adds the two BCD number in hexadecimal format and DAA instruction convert this hexadecimal result to BCD result.

Flag affected: CF, PF, AF, SF, ZF and OF is undefined.

Operation:

(a) If lower nibble of AL > 9 or AF = 1(Set), then AL = AL + 06.
(b) If higher nibble of AL > 9 or CF = 1 (Set) , then AL = AL + 60.
(c) If both above conditions are satisfied, then AL = AL + 66.

DAS (Decimal adjust after subtraction)

DAS instruction is used to convert the result of the subtraction of two packed BCD numbers to a packed BCD number. DAS instruction only works on AL register. So, DAS instruction must be used after the SUB/SBB instruction. The SUB/SBB instruction subtracts the two BCD number in hexadecimal format and DAS instruction convert this hexadecimal result to BCD result. The working of DAS instruction is given below.

Flag affected: CF, PF, AF, SF, ZF and OF is undefined.

Operation

(a) If lower nibble of AL > 9 or AF = 1 then AL = AL – 06.
(b) If higher nibble of AL > 9 or CF = 1 then AL = AL – 60.
(c) If both above conditions are satisfied then AL = AL – 66.

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

a. Handle computer System and peripheral with care.
b. Shut down PC properly

XI Procedure

a. Write an algorithm and draw flowchart of given program. (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file

f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....
.....
.....
.....

XIII Observations

1) Observe and write the contents of Register using debugger TD or Debug after the execution of program.

Table 1: Contents of Registers

Registers			Flag Register		
	After	Before			
AX			Carry Flag	CF	
BX			Zero Flag	ZF	
CX			Sign Flag	SF	
DX			Overflow Flag	OF	
SI			Parity Flag	PF	
DI			Auxiliary Carry Flag	AF	
BP			Interrupt Flag	IF	
SP			Direction Flag	DF	
DS					
ES					
SS					
CS					
IP					

2) Observe and write the contents of memory location in Data Segment after the execution of program

Table 2: Contents of memory location in Data Segment

Address	Contents	Address	Contents
DS:0000		DS:0008	
DS:0001		DS:0009	
DS:0002		DS:000A	
DS:0003		DS:000B	
DS:0004		DS:000C	
DS:0005		DS:000D	
DS:0006		DS:000E	
DS:0007		DS:000F	

XIV Program Code with comments

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

XV Results (Output of Program)

(Note: Write an Output of program assigned by teacher)

.....
.....
.....
.....

XVI Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write the flags used for BCD arithmetic operation.

.....
.....
.....
.....

2. Write the instructions that converts the result of addition and subtraction in unpacked decimal digits.

.....
.....
.....
.....

XVII Exercise (Question 2 and 3 are compulsory and 1 is optional)

(Use blank space provided for answers or attach more pages if needed)

1. Write an ALP to multiply the two BCD numbers stored in BL and CL register**.**

2. Write an output of DAA instruction in AL register of following code after the execution and also the status of CF and AF.

```
MOV AL,99H
MOV BL,01H
ADD AL, BL
DAA
```

3. Write an output of DAS instruction in AL register of following code after the execution and also the status of CF and AF.

```
MOV AL,03H
MOV BL,07H
SUB AL, BL
DAA
```

(Space for answers)

XVIII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program file	10%
2	Use assembler and linker to create .exe file	20%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……
2. ……....
3. ……....
4. ……

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 9: Implement loop to find

i. **Sum of series of Hexadecimal numbers.**
ii. **Sum of series of BCD numbers.**

I Practical Significance

In some industrial applications of assembly language programming, it is required to repeat group of instructions for specific number of times such as providing time delay while generating waves such as square, triangular, saw tooth etc. Students will be able to implement loop by using variants of Jump instructions.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge:
PO 2- Discipline knowledge
PO 3- Experiments and practice
PO 4- Engineering tools

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills

1. Use editor to create assembly language program *filename.asm* file
2. Use assembler and linker to create *filename.exe* file
3. Use debugger in single step mode to locate/trace the errors and correcting the errors

IV Relevant Course Outcome(s)

a. Develop an assembly language program using assembler.

V Practical Outcomes

a. Write an ALP to find sum of series of Hexadecimal Numbers
b. Write an ALP to find sum of series of BCD numbers.

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

The addition of the numbers in the series or array of n numbers which are stored in the memory is called as sum of series. So, byte or word counter which indicate length of series is required to read numbers from the series one by one. The result of addition may be greater than either 8 bit or 16 bit depending on numbers stored in the array.
The memory pointer and counter must be initialized to read byte or word from the series of n numbers.

Loop Instructions

Instruction	Action
LOOP Label	CX= CX-1 ; if (CX < > 0) jump to label
LOOPZ/LOOPE Label	CX= CX-1 ;if (CX <> 0) AND (ZF=1) jump to target
LOOPNZ/LOOPNE Label	CX= CX-1 ;if (CX <> 0) AND (ZF=0) jump to target
JCXZ label	CX= CX-1; if CX=0 jump to target

VIII Work Situation

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

1. Handle computer system and peripherals with care.
2. Shut down PC properly

XI Procedure

a. Write algorithm and draw flowchart for given program (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....

.....

.....

.....

XIII Observations:

1) Observe and write the contents of Register using debugger TD or Debug after the execution of program.

Table 1: Contents of Registers

Registers			Flag Register		
	After	Before			
AX			Carry Flag	CF	
BX			Zero Flag	ZF	
CX			Sign Flag	SF	
DX			Overflow Flag	OF	
SI			Parity Flag	PF	
DI			Auxiliary Carry Flag	AF	
BP			Interrupt Flag	IF	
SP			Direction Flag	DF	
DS					
ES					
SS					
CS					
IP					

2) Observe and write the contents of memory location in Data Segment after the execution of program

Table 2: Contents of memory location in Data Segment

Address	Contents	Address	Contents
DS:0000		DS:0008	
DS:0001		DS:0009	
DS:0002		DS:000A	
DS:0003		DS:000B	
DS:0004		DS:000C	
DS:0005		DS:000D	
DS:0006		DS:000E	
DS:0007		DS:000F	

XIV Program Code with comments

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

XV Results (Output of the Program)

(*Note: Write an Output of program assigned by teacher*)

.....
.....
.....
.....

XVI Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Note down the registers used as memory pointer and counter in a program?

.....
.....
.....
.....

2. State the use of INC instruction in your program.

.....
.....
.....
.....

XVII Exercise

(Use blank space provided for answers or attach more pages if needed)

1. What is the condition to terminate loop formed using LOOP instruction?
2. Write applications where loop instruction can be used?
3. Which register is used as a counter to store count for a LOOP instruction?

(Space for answers)

XVIII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
•	Use editor to create assembly language program file	10%
•	Use assembler and linker to create .exe file	20%
•	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
•	Practical related questions	10%
•	Completion and submission of practical in time	10%
•	Expected Output/Observation	10%
	Total (50 Marks)	**100%**

List of student Team Members

1. ……
2. ……....
3. ……
4. ……

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 10: Write an ALP to find smallest/largest number from an array of n numbers.

I Practical Significance

In assembly language programming, flags are affected after compare instruction. The status of the flags can be used to make decision about smaller or greater number. Student will be able to use compare instruction and decision making instruction to find smallest and largest number.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge:
PO 2- Discipline knowledge
PO 3- Experiments and practice
PO 4- Engineering tools

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills.

a. Use editor to create assembly language program *filename.asm* file.
b. Use assembler and linker to create *filename.exe* file.
c. Use debugger in single step mode to locate/trace the errors and correcting the errors.

IV Relevant Course Outcome(s)

a. Develop an assembly language program using assembler.

V Practical Outcomes

a. Write an ALP to find smallest number from an array of n numbers
b. Write an ALP to find largest number from an array of n numbers

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

Array is the set of N numbers i.e. byte or word. Hence, memory pointer and counter is required to read or write numbers from or to memory location in the array.

To find smallest/largest number from the array, the numbers in the array must be compared with each other. Array may consist of 8 bit numbers i.e. byte or 16 bit numbers i.e. word, so memory pointer is required to read numbers from the array. Also, one counter called as byte or word counter which indicates how many numbers are there in the array, is required in the program to read and compare only desired numbers from the array. In 8086, the CMP instruction is used to compare two numeric data fields.

CMP destination, source

The CMP instruction compares a byte/word from the specified source and a byte/word from the specified destination. The source and destination can be an immediate data, a

register or a memory location. However, the source and the destination should not both be memory locations. The comparison is actually done by non-destructive subtraction of the source byte or word from the destination byte or word i.e. the source and the destination will not change, but the flags will affect to specify the results of the comparison.

Flag affected: OF, CF, PF, AF, SF, ZF.

Condition	CF	ZF	SF	Meaning of flag status
AX = BX	0	1	0	Source and destination operands are equal
AX > BX	0	0	0	Destination operand is greater than source operand
AX < BX	1	0	1	Destination operand is smaller than source

Conditional Jump instruction is used to jump to certain location/memory address, after condition is satisfied

Symbol/ Instruction	**Description**	**Flags affected**
JE/JZ	Jump if Equal or Jump if Zero	ZF
JNE/JNZ	Jump if not Equal or Jump if Not Zero	ZF
JA/JNBE	Jump if Above or Jump if Not Below/Equal	CF,ZF
JAE/JNB	Jump if Above/Equal or Jump if Not Below	CF
JB/JNAE	Jump if Below or Jump if Not Above/Equal	CF
JBE/JNA	Jump if Below/ Equal or Jump if Not Above	AF,CF
JG/JNLE	Jump if Greater or Jump if not Less/Equal	OF,SF,ZF
JGE/JNL	Jump if Greater/Equal or Jump if not Less	OF,SF
JL/JNGE	Jump if Less or Jump if not Greater/Equal	OF,SF
JLE/JNG	Jump if Less/Equal or Jump if not Greater	OF,SF,ZF
JC	Jump if Carry	CF
JNC	Jump if not Carry	CF

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

1. Handle computer system and peripherals with care.
2. Shut down PC properly

XI Procedure

a. Write algorithm and draw flowchart for given program (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....
.....
.....
.....
.....

XIII Observations

1) Observe and write the contents of memory location and AL register after the execution of program.

Table 1: Contents of memory location and AL register while finding smallest number

Address	Original Contents	Loop 1	Loop 2	Loop 3	Loop 4	Loop 5
DS:0000	**12**	**AL =____**	**AL =____**	**AL =____**	**AL =____**	**AL =____**
DS:0001	**07**					
DS:0002	**25**					
DS:0003	**18**					
DS:0004	**02**					

2) Observe and write the contents of memory location and AL register after the execution of program.

Table 2: Contents of memory location and AL register while finding largest number

Address	Original Contents	Loop 1	Loop 2	Loop 3	Loop 4	Loop 5
DS:0000	**12**	**AL =____**	**AL =____**	**AL =____**	**AL =____**	**AL =____**
DS:0001	**07**					
DS:0002	**25**					
DS:0003	**18**					
DS:0004	**02**					

Program Code with comments

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XIV Results (Output of the Program)

(*Note: Write an Output of program assigned by teacher*)

.....

.....

.....

.....

XV Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Which flags are affected by CMP instruction?

.....

.....

.....

.....

2. Which instructions are used to make decision to find smallest/largest number in the program?

.....

.....

.....

.....

XVI Exercise

(Use blank space provided for answers or attach more pages if needed)

1. Show flag status after comparisons of following operands

N1	N2	CMP N1,N2			N1	N2	CMP N2,N1		
		CF	ZF	SF			CF	ZF	SF
25	45				75	36			
75	43				23	87			
234	234				100	100			

2. Write syntax of CMP instruction with suitable example.
3. Which condition jump instructions are used to find largest and smallest number?

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XVII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XVIII Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program file	10%
2	Use assembler and linker to create .exe file	20%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……..
2. ……...
3. ……..
4. …….

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No.11: Write an assembly language program to arrange numbers in ascending/descending order

I Practical Significance

Sorting is a process that organizes a collection of data into either ascending or descending order. This operation requires comparison of data and exchange the position of data depending on result of comparison. There are different algorithms for sorting data. Students will be able to use XCHG or MOV instruction while implementing sorting algorithms.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge:

PO 2- Discipline knowledge

PO 3- Experiments and practice

PO 4- Engineering tools

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills.

a. Use editor to create assembly language program *filename.asm* file.
b. Use assembler and linker to create *filename.exe* file.
c. Use debugger in single step mode to locate/trace the errors and correcting the errors.

IV Relevant Course Outcome(s)

a. Develop an assembly language program using assembler.

V Practical Outcomes

a. Write an ALP to arrange numbers in an array in ascending order
b. Write an ALP to arrange numbers in an array in descending order

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

If numbers in an array are arranged such that every n^{th} number is greater than $(n-1)^{th}$ number, then that array is in ascending order. If numbers in an array are arranged such that every n^{th} number is smaller than $(n-1)^{th}$ number, then that array is in descending order. There are many sorting algorithms such as Selection sort, Insertion sort, Bubble sort, Merge sort, Quick sort. Arranging numbers involves different operations such as comparing numbers, swapping numbers depending on result of comparison, repeating comparison operation for all numbers in an array

XCHG destination, source

This instruction exchanges the contents of a register with the contents of another register or memory location. The instruction cannot directly exchange the contents of two memory locations. A memory location can be specified as the source or as the

destination. The source and destination should both be word or they must both be byte. The segment register cannot be used in this instruction.

Operation performed by XCHG instruction: Destination ↔ Source

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

1. Handle computer system and peripherals with care.
2. Shut down PC properly

XI Procedure

a. Write algorithm and draw flowchart for given program (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (with major specifications)

.....
.....
.....
.....
.....
.....

XIII Observations

1) Observe and write the contents of memory location after the execution of program.

Table 1: Contents of memory location in ascending order operation

Address	Original Contents	Pass 1	Pass 2	Pass 3	Pass 4	Pass 5
DS:0000	**12**					
DS:0001	**07**					
DS:0002	**25**					
DS:0003	**18**					
DS:0004	**02**					

2) Observe and write the contents of memory location after the execution of program.

Table 2: Contents of memory location in descending order operation

Address	Original Contents	Pass 1	Pass 2	Pass 3	Pass 4	Pass 5
DS:0000	**12**					
DS:0001	**07**					
DS:0002	**25**					
DS:0003	**18**					
DS:0004	**02**					

XIV Program Code with comments

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

XV Results (Output of the Program)

(Note: Write an Output of program assigned by teacher)

.....

.....

.....

.....

XVI Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. What is the use of XCHG instruction in your program?

.....

.....

.....

.....

2. Which sorting algorithm is used in your program?

.....

.....

.....

.....

XVII Exercise

(Use blank space provided for answers or attach more pages if needed)

1. If numbers in an array are 07H,02H,09H,10H,06H, write the array contents in each pass while arranging numbers in ascending order .
2. If numbers in an array are 07H,02H,09H,10H,06H, write the array contents in each pass while arranging numbers in descending order.

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XVIII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program file	10%
2	Use assembler and linker to create .exe file	20%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……
2. ……....
3. ……
4. ……

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No.12: Write an assembly language program to find length of string, arrange string in reverse order and concatenate strings

I Practical Significance

String is a sequence of characters enclosed in quotes. In various applications it is required to display messages to get input from user, search particular character/word in string, arrange characters in string, combine different strings. Student will be able to perform different operations on string.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge:

PO 2- Discipline knowledge

PO 3- Experiments and practice

PO 4- Engineering tools

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills.

1. Use editor to create assembly language program *filename.asm* file.
2. Use assembler and linker to create *filename.exe* file.
3. Use debugger in single step mode to locate/trace the errors and correcting the errors.

IV Relevant Course Outcome(s)

a. Develop an assembly language program using assembler.

V Practical Outcomes

a. Write an ALP to arrange string in reverse order
b. Write an ALP to find string length
c. Write an ALP to concatenate two strings

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

The string consists of either numbers or characters. In assembly language programming, the string must be declared in single quotes i.e. ' ' and must end with '$' sign. The data type of the string is always byte because assembler store 8 bit ASCII value of every character of string in memory.

For Example

```
dept    db      'Computer Engineering$'
```

Assembler stores string characters in memory at consecutive memory locations. Hence to perform any string related operation such as comparison, length, reverse etc., the memory pointer and byte counter is required

Without byte counter, the string operations are possible. For that, you have to read character from string array and compare it with '$'. If character is not '$', then character is string character. If character is '$', then it is end of string.

Length of String:

To find the length of the string, we need one length counter and initialize memory pointer to read character from the string. Read character from the array and compare it with '$' which indicate end of the string. If the character is not '$' then increment length counter else stop reading character from the string.

String in reverse order:

The memory pointer and length counter is required to read string and then copy string in another blank string variable in reverse order. To reverse the string, first find out the length of the source string, then add this value to memory pointer register to point last character of the source string. Now copy last character from source string to first character position of destination blank string. Perform this operation continuously till first character of the source string gets transfer to destination string by decrementing memory pointer for source string and incrementing memory pointer for destination string.

Concatenation of Two Strings:

The concatenation of two strings means merging of second string in first string. For example, suppose, 'Computer' and 'Department' are two separate strings, after concatenation string will become 'Computer Department'.

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

1. Handle computer system and peripherals with care.
2. Shut down PC properly

XI Procedure

a. Write algorithm and draw flowchart for given program (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.

e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....
.....
.....
.....

XIII Observations

Table 1: Reverse of a string

	Example 1	Example 2
Input string taken	**MSBTE**	_____
Reverse string		

Table 2: Length of string

	Example 1	Example 2
Input string taken	**Microprocessor**	_____
Length of string		

Table 3: String concatenation

	Example 1	Example 2
Input string 1 taken	**Microprocessor**	_____
Input string 2 taken	**Programming**	_____
Output string		

XIV Program Code with comments

.....
.....
.....
.....
.....
.....
.....
.....
.....

XV Results (Output of the Programs)

(*Note: Write an Output of program assigned by teacher*)

.....

.....

.....

.....

XVI Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. State the registers that are used as memory pointers in string reverse and concatenation program.

.....

.....

.....

.....

2. State the role of direction flag while using string instructions?

.....

.....

.....

.....

XVII Exercise (Any one from Question 1 and 2)

(Use blank space provided for answers or attach more pages if needed)

1. Write an ALP to perform string copy operation.
2. Write an ALP to find the string is palindrome or not.
3. What is advantage of using string instructions over normal instructions

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XVIII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program file	10%
2	Use assembler and linker to create .exe file	20%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……
2. ……...
3. ……
4. ……

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 13: Write an ALP to count odd and/or even numbers in array

I Practical Significance

Decimal or hexadecimal numbers consists of Odd as well as Even numbers. Most of the times, it is required to check number is odd or even such as odd or even parity used in serial communication. Hence, students will be able to check and count odd and even numbers in array using assembly language program.

.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge:

PO 2- Discipline knowledge

PO 3- Experiments and practice

PO 4- Engineering tools practice

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills.

1. Use editor to create assembly language program *filename.asm* file.
2. Use assembler and linker to create *filename.exe* file.
3. Use debugger in single step mode to locate/trace the errors and correcting the errors.

IV Relevant Course Outcome(s)

a. Develop assembly language program using assembler.

V Practical Outcomes

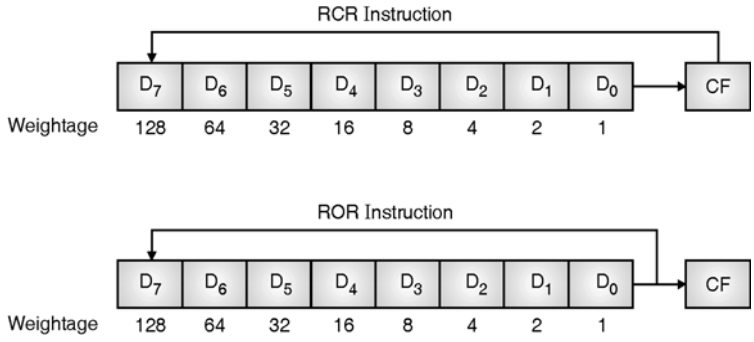
a. Write an ALP to check a given number is ODD or EVEN.
b. Write an ALP to count ODD and/or EVEN numbers in array.

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

In 8 bit or 16-bit number, the D_0 bit is used to decide the given number is odd or even because the weightage of D_0 bit in decimal is 1 i.e. odd value and the weightage of D_1, D_2 …. D_{15} bits are 2, 4, 8… i.e. even value in 8 bit or 16-bit number. When two even or odd numbers are added, then result is always even, but when odd number is added with even number, then result is always odd. Hence, when D_0 bit of any number is 1, then that number is odd and if 0 then number is even. To test any number for odd or even, check D_0 bit of that number. To check D_0 bit of any number, rotate the bits of that number toward left by 1 bit using rotate instruction i.e. ROR or RCR as shown as follows:



Then D_0 bit goes to the carry flag, hereafter by checking carry flag, number can be tested for odd or even.

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

a. Handle computer System and peripheral with care.
b. Shut down PC properly

XI Procedure

a. Write an algorithm and draw flowchart of given program. (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....

.....

.....

.....

XIII Observations

1) Observe and write the contents of Register using debugger TD or Debug after the execution of program.

Table 1: Contents of Registers

Registers			Flag Register		
	After	Before			
AX			Carry Flag	CF	
BX			Zero Flag	ZF	
CX			Sign Flag	SF	
DX			Overflow Flag	OF	
SI			Parity Flag	PF	
DI			Auxiliary Carry Flag	AF	
BP			Interrupt Flag	IF	
SP			Direction Flag	DF	
DS					
ES					
SS					
CS					
IP					

2) Observe and write the contents of memory location in Data Segment after the execution of program.

Table 2: Contents of memory location in Data Segment

Address	Contents	Address	Contents
DS:0000		DS:0008	
DS:0001		DS:0009	
DS:0002		DS:000A	
DS:0003		DS:000B	
DS:0004		DS:000C	
DS:0005		DS:000D	
DS:0006		DS:000E	
DS:0007		DS:000F	

XIV Program Code with comments

.....

.....

.....

.....

XV Results (Output of Program)

(Note: Write an Output of program assigned by teacher)

.....

.....

.....

.....

XVI Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write the flag used to check whether the number is ODD or EVEN.

.....

.....

.....

.....

2. Which bit of 8/16-bit number is used to decide if number is odd or even?

.....

.....

.....

.....

XVII Exercise (Any One)

(Use blank space provided for answers or attach more pages if needed)

1. Write an ALP to count odd as well as even numbers in array of 10 numbers.
2. Write an ALP to add the all odd numbers in array of 10 numbers**.**

(Space for answers)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

XVIII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program file	10%
2	Use assembler and linker to create .exe file	20%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……..
2. ……...
3. ……..
4. ……...

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 14: Write an ALP to count positive and/or negative numbers in array

I Practical Significance

The signed hexadecimal number is denoted in second complement format. The most significant bit (MSB) of any signed hexadecimal number represent sign of number, if MSB of signed hexadecimal number is 0, then number will positive and if MSB of signed hexadecimal number is 1, the number will be negative. Hence, students will be able to check or count positive and negative numbers in array using assembly language program.

.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge

PO 2- Discipline knowledge

PO 3- Experiments and practice

PO 4- Engineering tools practice

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills.

1. Use editor to create assembly language program *filename.asm* file.
2. Use assembler and linker to create *filename.exe* file.
3. Use debugger in single step mode to locate/trace the errors and correcting the errors.

IV Relevant Course Outcome(s)

a. Develop assembly language program using assembler.

V Practical Outcomes

a. Write an ALP to check a given number is POSITIVE or NEGATIVE

b. Write an ALP to count POSITIVE and/or NEGATIVE numbers in array..

VI Relevant Affective Domain Related Outcomes

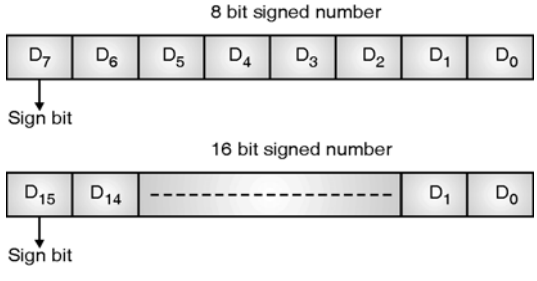
a. Follow precautionary measures.

b. Demonstrate working as a leader / a team member.

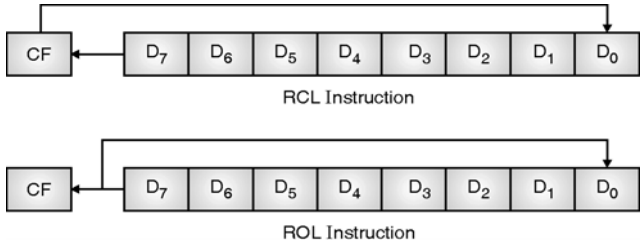
c. Follow ethical practices.

VII Minimum Theoretical Background

The most significant bit (MSB) i.e. D_7 or D_{15} in 8 bit or 16-bit signed magnitude number indicate sign of the number i.e. D_7 or D_{15} as shown Fig. given below



Hence, by checking most significant bit, we can find out a byte or word is positive or negative number. Most significant bit i.e. D_7 or D_{15} for byte or word can be checked using either ROL or RCL instruction as given in Fig. given below.



The program for checking odd or even number can be used by replacing **ROR** or **RCR** instruction with **ROL** or **RCL** instruction to check either number is positive of negative.

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

a. Handle computer System and peripheral with care.
b. Shut down PC properly

XI Procedure

a. Write an algorithm and draw flowchart of given program. (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....

.....

.....

.....

XIII Observations

1) Observe and write the contents of Register using debugger TD or Debug after the execution of program.

Table 1: Contents of Registers

Registers			Flag Register		
	After	**Before**			
AX			Carry Flag	CF	
BX			Zero Flag	ZF	
CX			Sign Flag	SF	
DX			Overflow Flag	OF	
SI			Parity Flag	PF	
DI			Auxiliary Carry Flag	AF	
BP			Interrupt Flag	IF	
SP			Direction Flag	DF	
DS					
ES					
SS					
CS					
IP					

2) Observe and write the contents of memory location in Data Segment after the execution of program.

Table 2: Contents of memory location in Data Segment

Address	Contents	Address	Contents
DS:0000		DS:0008	
DS:0001		DS:0009	
DS:0002		DS:000A	
DS:0003		DS:000B	
DS:0004		DS:000C	
DS:0005		DS:000D	
DS:0006		DS:000E	
DS:0007		DS:000F	

XIV Program Code with comments

.....
.....
.....
.....
.....
.....

XV Results (Output of Program)

(Note: Write an Output of program assigned by teacher)

.....
.....
.....
.....

XVI Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write the flag which is used to check whether the number is Positive or Negative.

.....
.....
.....
.....

2. Which bit of 8-bit and 16-bit number is used to decide if the number is Positive or Negative?

.....
.....
.....
.....

XVII Exercise (Any One)

(Use blank space provided for answers or attach more pages if needed)

1. Write an ALP to count Positive as well as Negative numbers in array of 10 numbers.
2. Write an ALP to add the all Positive numbers in array of 10 numbers**.**

(Space for answers)

.....
.....
.....
.....

.....

.....

.....

.....

.....

.....

.....

XVIII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program file	10%
2	Use assembler and linker to create .exe file	20%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……
2. ……...
3. ……
4. ……

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 15: Write an ALP to count '0's and '1's in a given number

I Practical Significance

In microprocessor based automation, the sensors output is connected to ports of microprocessor based system. Each sensor gives output on the corresponding pin of the input port. Microprocessor reads the contents of port i.e. all pins and copy it into the internal register. So, each sensor output can be checked by rotating the content of register toward left or right and find out the status of sensor connected to port pin. Hence, students will be able to check or count '0's and '1's in given numbers using assembly language program.

.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge

PO 2- Discipline knowledge

PO 3- Experiments and practice

PO 4- Engineering tools practice

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills.

1. Use editor to create assembly language program *filename.asm* file.
2. Use assembler and linker to create *filename.exe* file.
3. Use debugger in single step mode to locate/trace the errors and correcting the errors.

IV Relevant Course Outcome(s)

a. Develop assembly language program using assembler.

V Practical Outcomes

a. Write an ALP to count number of '1' in a given number
b. Write an ALP to count number of '0' in a given number.

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

The total numbers of 1's or 0's can be count in any number by rotating that number toward right or left by either 8 times for 8-bit number or 16 times for 16-bit number.

ROR or RCR or RCL or ROL instruction can be used to rotate any number to check how many ones or zeros are in the numbers.

When we rotate number once to left or right, corresponding bit i.e. D_0 or D_7 initially goes to carry flag, then we can check carry flag by using JNC or JC to count numbers of ones or zeros.

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

a. Handle computer System and peripheral with care.
b. Shut down PC properly

XI Procedure

a. Write an algorithm and draw flowchart of given program. (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

XIII Observations

1) Observe and write the contents of Register using debugger TD or Debug after the execution of program.

Table 1: Contents of Registers

Registers			Flag Register		
	After	Before			
AX			Carry Flag	CF	
BX			Zero Flag	ZF	
CX			Sign Flag	SF	
DX			Overflow Flag	OF	
SI			Parity Flag	PF	
DI			Auxiliary Carry Flag	AF	
BP			Interrupt Flag	IF	
SP			Direction Flag	DF	
DS					
ES					
SS					
CS					
IP					

2) Observe and write the contents of Register using debugger TD or Debug after the execution of program.

Table 2: Contents of memory location in Data Segment

Number 8-bit/16-bit in Hexadecimal	Nos. of '1's	Nos. of '0's
AA		
55		
88		
99		
FFFF		
AA55		
F0F0		
9898		

XIV Program Code with comments

.....

.....

.....

.....

.....

.....

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

XV Results (Output of Program)

(Note: Write an Output of program assigned by teacher)

.....
.....
.....
.....

XVI Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Write the flag used to count '1's and '0's.

.....
.....
.....
.....

2. Write the instructions used in your program to rotate and check numbers of '0' or '1'.

.....
.....
.....
.....

XVII Exercise

(Use blank space provide for answers or attached more pages if needed)

1. Modify your program to count number of '0' in AL register.
2. Write an ALP to check D_5 bit of number in BL register**.**

(Space for answers)

.....
.....
.....

.....

.....

.....

.....

.....

.....

XVIII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program file	10%
2	Use assembler and linker to create .exe file	20%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……..
2. ……...
3. ……
4. ……

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 16: Write an assembly language program using procedure.

I Practical Significance

A large program is tough to implement even if an algorithm is available, hence it should be divided into number of the independent tasks which can be easily designed and implemented. The process of dividing a large program into small tasks and designing them independently is called as modular programming. Large program is more prone to errors and it is difficult to find and segregate errors. A repeated group of instruction in a program can be organized as subprogram or subroutine or procedures in assembly language programming which permits reuse of program code. Hence, students will be able to write and use procedure in assembly language program.

.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge

PO 2- Discipline knowledge

PO 3- Experiments and practice

PO 4- Engineering tools practice

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills.

1. Use editor to create assembly language program *filename.asm* file.
2. Use assembler and linker to create *filename.exe* file.
3. Use debugger in single step mode to locate/trace the errors and correcting the errors.

IV Relevant Course Outcome(s)

a. Develop assembly language programs using procedures, macros and modular Programming approach.

V Practical Outcomes

a. Write an ALP for addition, subtraction, multiplication and division.
b. Write an ALP using procedure to solve equation such as Z=(A+B)*(C+D)

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

A procedure is a set of the program statements that can be processed independently and reuse again and again. Here are the four steps that need to be accomplished in order to call and return from a procedure.

1. Save return address
2. Call the procedure
3. Execute procedure
4. Return to calling program

The assembler directives PROC and ENDP are required to define a procedure. The directive PROC specifies the beginning of the procedure and the directive ENDP specifies the end of the procedure to the assembler. The directive PROC and ENDP must enclose the procedure code which defines the subroutine. The procedures must be defined within the code segment only.

Syntax:

```
procedure_name  PROC   [NEAR/FAR]
                :
                :
                :
               RET
               ENDP
```

The CALL instruction is used to transfer program control to a subprogram or a procedure by storing the return address on the stack. The call can be of two types

1. Inter-Segment or near call
2. Intra-Segment or far call

A near call refers to a procedure call which is in the same code segment as the CALL instruction and a far call refers to a procedure call which is in the different code segment from that of the CALL instruction.

Example:

```
CALL fact
```

The instruction RET is used to transfer program control from the procedure back to the calling program i.e. main program or procedure following the CALL. The RET instruction are of two types:

1. Near RET or inter segment return.
2. Far RET or intra segment return.

If a procedure is declared as near, the execution of the RET replaces the IP with a word from the top of the stack which contains the offset address of the instruction following the CALL instruction. Hence such return is called as near return because transfer of the control is within the segment. If a procedure is defined as far, the execution of RET instruction pops two words from the stack and places them into the registers IP and CS to transfer control to the calling program.

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

a. Handle computer System and peripheral with care.
b. Shut down PC properly

XI Procedure

a. Write an algorithm and draw flowchart of given program. (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....
.....
.....
.....

XIII Observations

1) Observe and write the contents of Register using debugger TD or Debug after the execution of program.

Table 1: Contents of Registers

Registers			**Flag Register**		
	After	**Before**			
AX			Carry Flag	CF	
BX			Zero Flag	ZF	
CX			Sign Flag	SF	
DX			Overflow Flag	OF	
SI			Parity Flag	PF	
DI			Auxiliary Carry Flag	AF	
BP			Interrupt Flag	IF	
SP			Direction Flag	DF	
DS					
ES					
SS					
CS					
IP					

2) Observe and write the contents of memory location in Data Segment after the execution of program.

2: Contents of memory location in Data Segment

Address	Contents	Address	Contents
DS:0000		DS:0008	
DS:0001		DS:0009	
DS:0002		DS:000A	
DS:0003		DS:000B	
DS:0004		DS:000C	
DS:0005		DS:000D	
DS:0006		DS:000E	
DS:0007		DS:000F	

XIV Program Code with comments

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

XV Results (Output of Program)

(Note: Write an Output of program assigned by teacher)

.....
.....
.....
.....
.....
.....
.....
.....
.....

XVI Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. Which procedure have been used in your program (Near or Far)?

.....
.....
.....
.....

2. Write the content of Instruction pointer IP before and after the execution of CALL instruction.

.....
.....
.....
.....

3. What are the advantages of using procedure?

.....
.....
.....
.....

XVII Exercise (Any one)

(Use blank space provided for answers or attach more pages if needed)

1. Write an ALP to find smallest number from the array of 10 numbers using procedure.
2. Write an ALP to count number of '1's in 8-bit number using procedure**.**

(Space for answers)

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

XVIII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program file	10%
2	Use assembler and linker to create .exe file	20%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……..
2. ……..
3. …….
4. …….

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

Practical No. 17: Write an assembly language program using macros.

I Practical Significance

In assembly language programs, small program codes of the same pattern are frequently occurring at different places of the program which perform the same operation on the different data of the same data type. Such repeated code can be written separately as a macro. The process of defining macros and using them to simplify the programming process is known as macros programming. Hence, students will be able to use macro in assembly language program.

.

II Relevant Program Outcomes (POs)

PO 1. Basic knowledge

PO 2- Discipline knowledge

PO 3- Experiments and practice

PO 4- Engineering tools practice

III Competency and Skills

"Develop assembly language program using 8086"

This practical is expected to develop the following skills.

1. Use editor to create assembly language program *filename.asm* file.
2. Use assembler and linker to create *filename.exe* file.
3. Use debugger in single step mode to locate/trace the errors and correcting the errors.

IV Relevant Course Outcome(s)

a. Develop assembly language programs using procedures, macros and modular programming approach.

V Practical Outcomes

a. Write an ALP for addition, subtraction, multiplication and division.
b. Write an ALP using MACRO to solve equation such as Z=(A+B)*(C+D)

VI Relevant Affective Domain Related Outcomes

a. Follow precautionary measures.
b. Demonstrate working as a leader / a team member.
c. Follow ethical practices.

VII Minimum Theoretical Background

When assembler encounters a macro name later in the source code, the block of code associated with the macro name is substituted or expanded at the point of call, known as macro expansion. Hence macro is called as open subroutine. Macros should be used when its body has a few program statements; otherwise, the machine code of the program will be large on account of the same code being repeated in the position where macros are used. The directive MACRO and ENDM must enclose the definition, declarations, or a small part of the code which have to be substituted at the invocation of the macro. The macro should be start with directive MACRO and end with ENDM directive.

Syntax:

macro_name MACRO *[macro variables separated by colon]*
:
:
:
ENDM

VIII Work Situation:

a. Faculty will demonstrate the use of assembly language programming tools to write and execute the program.
b. Faculty must form a group of two students.
c. Students group will use the assembly language programming tools to write and execute the programs.

IX Resources required (Additional)

Sr. No.	Instrument /Object/Software	Specification	Quantity	Remarks
1.				
2.				
3.				
4.				
5.				

X Precautions to be followed

a. Handle computer System and peripheral with care.
b. Shut down PC properly

XI Procedure

a. Write an algorithm and draw flowchart of given program. (Use blank space provided or attach more pages if needed)
b. Double clik on DOSBOX TASM 1.4 icon.
c. Type *edit filename.asm* on DOS prompt and press Enter Key
d. Type the program and save on disk.
e. Once the assembly language program is created, then type *tasm filename.*asm on the command prompt and press Enter Key to create *filename.obj* file
f. Type *tlink filename.obj or tlink filename* on command prompt and press Enter Key to create *filename .exe* file.
g. Finally, type *debug filename.exe* or *td filename.exe* on the command prompt and press Enter Key to debug your program step by step.
h. Observe the contents of registers, memory location used and status of flags.

XII Resources used (Additional)

.....
.....
.....
.....
.....

XIII Observations

1) Observe and write the contents of Register using debugger TD or Debug after the execution of program.

Table 1: Contents of Registers

Registers			Flag Register		
	After	Before			
AX			Carry Flag	CF	
BX			Zero Flag	ZF	
CX			Sign Flag	SF	
DX			Overflow Flag	OF	
SI			Parity Flag	PF	
DI			Auxiliary Carry Flag	AF	
BP			Interrupt Flag	IF	
SP			Direction Flag	DF	
DS					
ES					
SS					
CS					
IP					

2) Observe and write the contents of memory location in Data Segment after the execution of program

Table 2: Contents of memory location in Data Segment

Address	Contents	Address	Contents
DS:0000		DS:0008	
DS:0001		DS:0009	
DS:0002		DS:000A	
DS:0003		DS:000B	
DS:0004		DS:000C	
DS:0005		DS:000D	
DS:0006		DS:000E	
DS:0007		DS:000F	

XIV Program Code with comments

.....

.....

.....

.....

.....

.....

.....

.....

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

XV Results (Output of Program)

(Note: Write an Output of program assigned by teacher)

.....
.....
.....
.....

XVI Practical related Questions

Note: Below given are few sample questions for reference. Teachers must design more such questions to ensure the achievement of identified CO.

1. State the advantages and disadvantages using macro.

.....
.....
.....
.....

2. State the function of directive MACRO and ENDM.

.....
.....
.....
.....

XVII Exercise

(Use blank space provided for answers or attach more pages if needed)

1. Write an ALP to perform $\mathbf{y=a^2+b^2+c^2}$ using macro to compute square**.**

(Space for answers)

XVIII References / Suggestions for further Reading

1. https://www.elprocus.com/8086-assembly-language-programs-explanation/
2. http://mysc.altervista.org/beginners-guide-8086/
3. https://www.tutorialspoint.com/assembly_programming/

XIX Assessment Scheme

Performance Indicators		Weightage
Process related (35 Marks)		**70%**
1	Use editor to create assembly language program file	10%
2	Use assembler and linker to create .exe file	20%
3	Use debugger in single step mode to locate/trace the errors and correcting the errors	40%
Product related (15 Marks)		**30%**
4	Practical related questions	10%
5	Completion and submission of practical in time	10%
6	Expected Output/Observation	10%
Total (50 Marks)		**100%**

List of student Team Members

1. ……..
2. ……..
3. ……..
4. ……..

Marks Obtained			Dated signature of Teacher
Process Related(35)	**Product Related(15)**	**Total (50)**	

List Of Laboratory Manuals Developed by MSBTE

First Semester:

1	Fundamentals of ICT	22001
2	English	22101
3	English Work Book	22101W
4	Basic Science (Chemistry)	22102
5	Basic Science (Physics)	22102

Second Semester:

1	Bussiness Communication Using Computers	22009
2	Computer Peripherals & Hardware Maintenace	22013
3	Web Page Design with HTML	22014
4	Applied Science (Chemistry)	22202
5	Applied Science (Physics)	22202
6	Applied Machines	22203
7	Basic Surveying	22205
8	Applied Science (Chemistry)	22211
9	Applied Science (Physics)	22211
10	Fundamental of Electrical Engineering	22212
11	Elements of Electronics Engineering	22213
12	Elements of Electrical Engineering	22215
13	Basic Electronics	22216
14	C Language programming	22218
15	Basic Electronics	22225
16	Programming in C	22226
17	Fundamental of Chemical Engineering	22231

Third Semester:

1	Applied Multimedia Techniques	22024
2	Advanced Serveying	22301
3	Highway Engineering	22302
4	Mechanics of Structures	22303
5	Building Construction	22304
6	Concrete Technology	22305
7	Strength Of Materials	22306
8	Automobile Engines	22308
9	Automobile Transmission System	22309
10	Mechanical Operations	22313
11	Technology Of Inorganic Chemicals	22314
12	Object Oriented Programming Using C++	22316
13	Data Structure Using 'C'	22317
14	Computer Graphics	22318
15	Database Management System	22319
16	Digital Techniques	22320
17	Principles Of Database	22321
18	Digital Techniques & Microprocessor	22323
19	Electrical Circuits	22324
20	Electrical & Electronic Measurment	22325
21	Fundamental Of Power Electronics	22326
22	Electrical Materials & Wiring Practice	22328
23	Applied Electronics	22329
24	Electrical Circuits & Networks	22330
25	Electronic Measurments & Instrumentation	22333
26	Principles Of Electronics Communication	22334
27	Thermal Engineering	22337
28	Engineering Matrology	22342
29	Mechanical Engineering Materials	22343
30	Theory Of Machines	22344

Fourth Semester:

1	Hydraulics	22401
2	Geo Technical Engineering	22404
3	Chemical Process Instrumentation & Control	22407
4	Fluid Flow Operation	22409
5	Technology Of Organic Chemical	22410
6	Java Programming	22412
7	GUI Application Development Using VB.net	22034
8	Microprocessor	22415
9	Database Managment	22416
10	Electric Motors And Transformers	22418
11	Industrial Measurement	22420
12	Digital Electronic And Microcontroller Application	22421
13	Linear Integrated Circuits	22423
14	Microcontroller & Applications	22426
15	Basic Power Electronics	22427
16	Digital Communication Systems	22428
17	Mechanical Engineering Measurments	22443
18	Fluid Mechanics and Machinery	22445
19	Fundamentals Of Mechatronics	22048
20	Micro Project & Industrial Training Assessment Manual	22049

Fifth Semester:

1	Network Management & Administration	17061
2	Solid Modeling	17063
3	CNC Machines	17064
4	Behavioral Science(Hand Book)	17075
5	Behavioral Science (Assignment Book)	17075
6	Windows Programming using VC++	17076
7	Estimation and Costing	17501
8	Public Health Engineering	17503
9	Concrete Technology	17504
10	Design of Steel Structures	17505
11	Switchgear and Protection	17508
12	Microprocessor & Application	17509
13	A.C. Machines	17511
14	Operating System	17512
15	Java Programming	17515
16	System Programming	17517
17	Communication Technology	17519
18	Hydraulic & Pneumatics	17522
19	Advanced Automobile Engines	17523
20	Basic Electrical & Electronics	17524
21	Measurement and Control	17528
22	Power Engineering	17529
23	Metrology & Quality Control	17530
24	Computer Hardware & Networking	17533
25	Microcontroller	17534
26	Digital Communication	17535
27	Control System & PLC	17536
28	Audio Video Engineering	17537
29	Control System	17538
30	Industrial Electronics and applications	17541
31	Heat Transfer Operations	17560
32	Chemical Process Instrumentation & control	17561

Sixth Semester:

1	Solid Modeling	17063
2	Highway Engineering	17602
3	Contracts & Accounts	17603
4	Design of R.C.C. Structures	17604
5	Industrial Fluid Power	17608
6	Design of Machine Elements	17610
7	Automotive Electrical and Electronic Systems	17617
8	Vehicle Systems Maintenance	17618
9	Software Testing	17624
10	Advanced Java Programming	17625
11	Mobile Computing	17632
12	System Programing	17634
13	Testing & Maintenance of Electrical Equipments	17637
14	Power Electronics	17638
15	Illumination Engineering	17639
16	Power System Operation & Control	17643
17	Environmental Technology	17646
18	Mass Transfer Operation	17648
19	Advanced Communication System	17656
20	Mobile Communication	17657
21	Embedded System	17658
22	Process Control System	17663
23	Industrial Automation	17664
24	Industrial Drives	17667
25	Video Engineering	17668
26	Optical Fiber & Mobile Communication	17669
27	Therapeutic Equipment	17671
28	Intensive Care Equipment	17672
29	Medical Imaging Equipment	17673

Pharmacy Lab Manual

First Year:

1	Pharmaceutics - I	0805
2	Pharmaceutical Chemistry - I	0806
3	Pharmacognosy	0807
4	Biochemistry and Clinical Pathology	0808
5	Human Anatomy and Physiology	0809

Second Year:

1	Pharmaceutics - II	0811
2	Pharmaceutical Chemistry - II	0812
3	Pharmacology & Toxicology	0813
4	Hospital and Clinical Pharmacy	0816

HEAD OFFICE

Secretary,
Maharashtra State Board of Technical Education
49, Kherwadi, Bandra (East), Mumbai - 400 051
Maharashtra (INDIA)
Tel: (022)26471255 (5 -lines)
Fax: 022 - 26473980
Email: -secretary@msbte.com
Web -www.msbte.org.in

REGIONAL OFFICES:

MUMBAI

Deputy Secretary (T),
Mumbai Sub-region,
2nd Floor, Govt. Polytechnic Building,
49, Kherwadi, Bandra (East)
Mumbai - 400 051
Phone: 022-26473253 / 54
Fax: 022-26478795
Email: rbtemumbai@msbte.com

PUNE

Deputy Secretary (T),
M.S. Board of Technical Education,
Regional Office,
412-E, Bahirat Patil Chowk,
Shivaji Nagar, Pune
Phone: 020-25656994 / 25660319
Fax: 020-25656994
Email: rbtepn@msbte.com

NAGPUR

Deputy Secretary (T),
M.S. Board of Technical Education
Regional Office,
Mangalwari Bazar, Sadar, Nagpur - 440 001
Phone: 0712-2564836 / 2562223
Fax: 0712-2560350
Email: rbteng@msbte.com

AURANGABAD

Deputy Secretary (T),
M.S. Board of Technical Education,
Regional Office,
Osmanpura, Aurangabad -431 001.
Phone: 0240-2334025 / 2331273
Fax: 0240-2349669
Email: rbteau@msbte.com