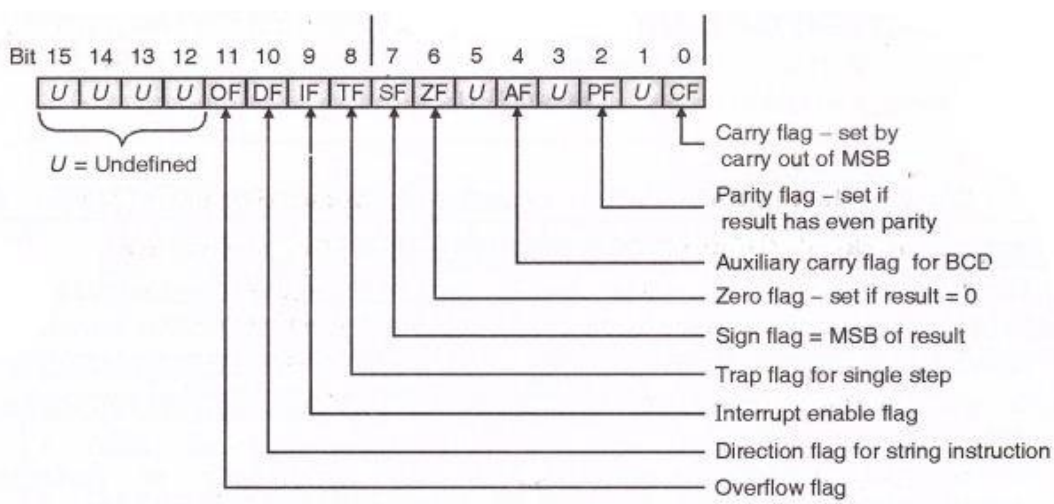
SUMMER – 2022 EXAMINATION

Subject Name: Microprocessor **Model Answer** **Subject Code:** 22415

Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills.
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.
8) As per the policy decision of Maharashtra State Government, teaching in English/Marathi and Bilingual (English + Marathi) medium is introduced at first year of AICTE diploma Programme from academic year 2021-2022. Hence if the students in first year (first and second semesters) write answers in Marathi or bilingual language (English +Marathi), the Examiner shall consider the same and assess the answer based on matching of concepts with model answer.

<table>
<tr><th>Q. No.</th><th>Sub Q. N.</th><th>Answer</th><th>Marking Scheme</th></tr>
<tr><td>1</td><td></td><td>Attempt any <u>FIVE</u> of the following:</td><td>10 M</td></tr>
<tr><td></td><td>a)</td><td>Draw the labeled format of 8086 flag register</td><td>2 M</td></tr>
<tr><td></td><td>Ans</td><td>


8086 flag register format
</td><td>Correct diagram: 2 M</td></tr>
</table>

	b)	**State any two difference between TEST and AND instructions.**	**2 M**
	Ans	(see table below)	1 M for each point of comparison

TEST	AND
This instruction logically ANDs the source with the destination but the result is not stored anywhere.	This instruction logically ANDs the source with the destination and stores the result in destination. .
e. g .TEST BL ,CL The result is not saved anywhere.	e.g. AND BL , CL The result is saved in BL register

	c)	**State the function of editor and assembler.**	**2 M**
	Ans	Editor: The editor is a program which allows the user to enter and modify as well as store a group of instructions or text under a file name. Assembler: The assembler is used to convert assembly language written by a user or a program into a machine recognizable format.	1 M for each function
	d)	**Write any two difference between NEAR and FAR procedure.**	**2 M**
	Ans	(see table below)	1 M for each point of comparison

SR.NO	NEAR PROCEDURE	FAR PROCEDURE
1.	A near procedure refers to a procedure which is in the same code segment from that of the call instruction.	A far procedure refers to a procedure which is in the different code segment from that of the call instruction.
2.	It is also called intra-segment procedure.	It is also called inter-segment procedure call.
3	A near procedure call replaces the old IP with new IP.	A far procedure call replaces the old CS:IP pairs with new CS:IP pairs.
4.	The value of old IP is pushed on to the stack. SP=SP-2 ;Save IP on stack(address of procedure)	The value of the old CS:IP pairs are pushed on to the stack SP=SP-2 ;Save CS on stack SP=SP-2 ;Save IP (new offset address of called procedure)
5.	Less stack locations are required	More stack locations are required
6.	Example :- Call Delay	Example :- Call FAR PTR Delay

	e)	**Write an ALP to add two 8 bit numbers.**	**2 M**
	Ans	.model small .data	Correct

		a db 06h
		b db 12h
		ends
		.code
		start:
		mov ax,@data
		mov ds,ax
		mov al,a
		mov bl,b
		add al,bl
		int 3
		ends
		end start
			program: 2 M

	f)	**Define immediate addressing mode with suitable example**	**2 M**
	Ans	An instruction in which 8 bit or 16 bit operand (data) is specified in instruction itself then the addressing mode of such instruction is called as immediate addressing mode. Eg. MOV AX,7120H	Definition :1M Example:1M
	g)	**State the use of DAA instruction in BCD addition.**	**2 M**
	Ans	The DAA (Decimal Adjust after Addition) instruction makes the result in Packed BCD from after BCD addition is performed. It works only on AL register.	Explanation: 2 M
2.		**Attempt any <u>THREE</u> of the following:**	**12 M**
	a)	**Describe the directives used to define the procedure with suitable example**	**4 M**
	Ans	Directives used for procedure: PROC directive: The PROC directive is used to identify the start of a procedure. The PROC directive follows a name given to the procedure. After that the term FAR and NEAR is used to specify the type of the procedure. ENDP Directive: This directive is used along with the name of the procedure to indicate the end of a procedure to the assembler. The PROC and ENDP directive are used in procedure. Example:	Description: 2 M Example: 2 M

		Procedure can be defined as Procedure_name PROC ---- ------ Procedure_name ENDP For Example Addition PROC near ------ Addition ENDP	
	b)	**Write the function of following pins of 8086:** **(i)** $\overline{BHE}$ **(ii) ALE** **(iii) READY** **(iv) RESET**	**4 M**
	Ans	(i) $\overline{BHE}$: BHE stands for Bus High Enable. It is available at pin 34 and used to indicate the transfer of data using data bus D8-D15. This signal is low during the first clock cycle, thereafter it is active. (ii) ALE: ALE stands for address Latch Enable, as address and data bus are multiplexed; ALE is used to lock either Address or Data. (iii) READY: It is used as acknowledgement from slower I/O device or memory. It is Active high signal, when high; it indicates that the peripheral device is ready to transfer data. (iv) RESET: This pin requires the microprocessor to terminate its present activity immediately	Each pin function 1 M
	c)	**Describe any four assembler directives with suitable example.**	**4 M**
	Ans	1. DB – The DB directive is used to declare a BYTE type variable – A BYTE is made up of 8 bits. Declaration examples: Num1 DB 10h	Each assembler directive 1 M

Num2 DB 37H

2. DW – The DW directive is used to declare a WORD type variable – A WORD occupies 16 bits or (2 BYTE).

   Declaration examples:

   TEMP DW 1234h

3. DD – The DD directive is used to declare a double word which is made up of 32 bits =2 Word's or 4 BYTE.

   Declaration examples:

   Dword1 DW 12345678h

4. EQU - This is used to declare symbols to which some constant value is assigned each time the assembler finds the given names in the program, it will replace the name with the value or a symbol. The value can be in the range 0 through 65535 and it can be another Equate declared anywhere above or below.

   .Num EQU 100

5. SEGMENT: It is used to indicate the start of a logical segment. It is the name given to the segment. Example: the code segment is used to indicate to the assembler the start of logical segment.

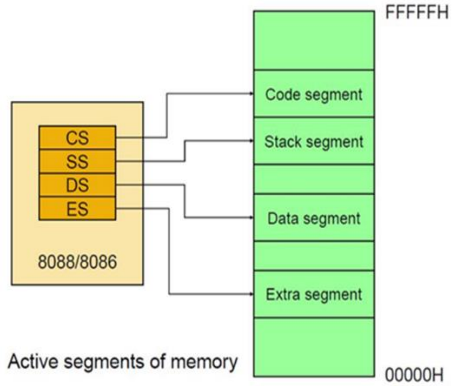
6. PROC: (PROCEDURE) It is used to identify the start of a procedure. It follows a name we give the procedure

   After the procedure the term NEAR and FAR is used to specify the procedure Example: SMART-DIVIDE PROC FAR identifies the start of procedure named SMART-DIVIDE and tells the assembler that the procedure is far.

	d)	**Describe DAS instruction with suitable example.**	**4 M**
	Ans	**DAS:** Decimal Adjust after Subtraction: - This instruction converts the result of the subtraction operation of 2 packed BCD numbers to a valid BCD number. The subtraction operation has to be only in the AL. If the lower nibble of AL is higher than the value 9, this instruction will subtract 06 from the lower nibble of the AL. If the output of the subtraction operation sets the carry flag or if the upper nibble is higher than value 9, it subtracts 60H from the AL. This instruction modifies the CF, AF, PF, SF, and ZF flags. The OF is not defined after DAS instruction. The instance is following:	Description 2 M Example 2 M

Example:

```
(i) AL = 75        BH = 46
    SUB AL,BH      ; AL ← 2 F = (AL) – (BH)
                   ; AF = 1
    DAS            ; AL ← 2 9 (as F> 9, F – 6 = 9)
```

3.		**Attempt any THREE of the following:**	**12 M**
	a)	**Describe memory segmentation in 8086 with suitable diagram.**	**4 M**
	Ans	 **Memory Segmentation**: The memory in 8086 based system is organized as segmented memory. 8086 can access 1Mbyte memory which is divided into number of logical segments. Each segment is 64KB in size and addressed by one of the segment register. The 4 segment register in BIU hold the 16-bit starting address of 4 segments. CS holds program instruction code. Stack segment stores interrupt & subroutine address. Data segment stores data for program. Extra segment is used for string data. ➢ The number of address lines in 8086 is 20, 8086 BIU will send 20bit address, so as to access one of the 1MB memory locations. ➢ The four segment registers actually contain the upper 16 bits of the starting addresses of the four memory segments of 64 KB each with which the 8086 is working at that instant of time ➢ A segment is a logical unit of memory that may be up to 64 kilobytes. Starting address will always be changing. It will not be fixed. Note that the 8086 does not work the whole 1MB memory at any given time. However, it works only with four 64KB segments within the whole 1MB memory.	Diagram: 2 M Explanation: 2 M
	b)	**Write an ALP to multiply two 16 bit signed numbers.**	**4 M**
	Ans	.model small .data A db 2222h B db 1111h	Program Code: 4 M

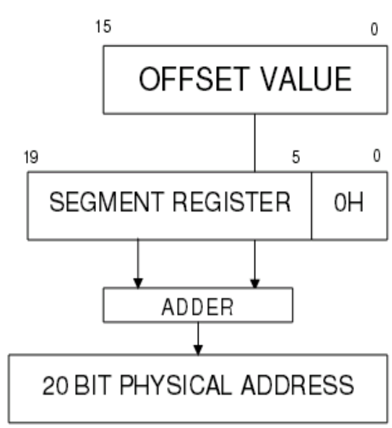
		Ends
		.code
		Mov ax,@data
		Mov ds,ax
		Mov AX,a
		Mov BX,b
		IMul BX
		Int 03h
		Ends
		End	

	c)	**Write an ALP to count odd numbers in the array of 10 numbers**	**4 M**

	Ans	. Model Small
		
		.data
		
		BLK DB 10h,40h,30h,60h
		e db ?h
		o db ?h
		 ends
		.code
		mov ax, @data
		mov ds, ax
		lea si, BLK
		mov bl, 00h
		mov bh, 00h
		mov cl, 04h
		up: mov al, [si]
		ror al, 1
		jc go
		inc bl
		jmp next
		go: inc bh
		next: inc si
		  dec cl
		   jnz up
		   mov e,bl
		   mov o,bh
		int 3
		 ends
		end
			Program Code: 4 M

	d)	**Write a MACRO to perform 32 bit by 16 bit division of unsigned numbers.**	**4 M**
	Ans	.model small Div1 macro no1,no2	Program Code: 4 M

		mov ax,no1
		div no2
		endm
		.data
		num1 dw 12346666h
		num2 dw 2222h
		.code
		mov ax,@data
		mov ds,ax
		div1 num1,num2
		ends
		end	

4.		**Attempt any <u>THREE</u> of the following:**	**12 M**
	a)	**Describe how 20 bit Physical address is generated in 8086 microprocessor with suitable example.**	**4 M**
	Ans	**Formation of a physical address:-** Segment registers carry 16 bit data, which is also known as base address. BIU attaches 0 as LSB of the base address. So now this address becomes 20-bit address. Any base/pointer or index register carry 16 bit offset. Offset address is added into 20-bit base address which finally forms 20 bit physical address of memory location. 	Describition: 2 M Example: 2 M

		Example Assume DS= 2632H, SI=4567H DS : 26320H ……...0 added by BIU(or Hardwired 0) + SI : 4567H ---------------------------- 2A887H	
	b)	**Write an ALP to find largest number in the array.**	**4 M**

	Ans	.model small
		.data
		Array db 02h,04h,06h,01h,05h
		Ends
		.code
		Start: Mov ax,@data
		Mov ds,ax
		Mov cl,04h
		Lea si,array
		Mov al,[si]
		Up : inc si
		Cmp al,[si]
		Jnc next
		Mov al,[si]
		Next : dec cl
		Jnz up
		Int 03h
		Ends
			Program Code: 4 M

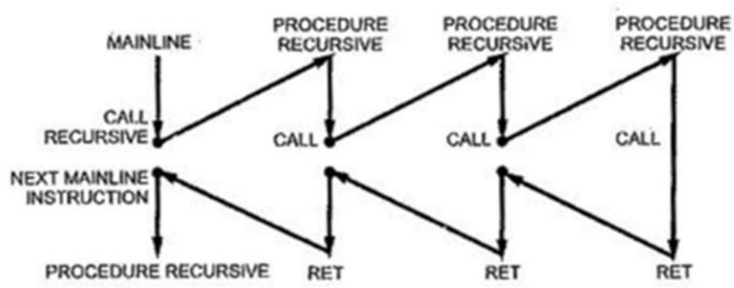
		End start	
	c)	**Write an ALP to count number of 0' in 8 bit number.**	**4 M**

	Ans	.MODEL SMALL
		.DATA
		NUM DB 08H
		ZEROS DB 00H
		.CODE
		START:
		MOV AX,@DATA
		MOV DS,AX
		MOV CX, 08H ; initialize rotation counter by 8
		MOV BX, NUM ;load number in BX
		UP: ROR BX, 1 ; rotate number by 1 bit right
		JC DN ; if bit not equal to 1 then go to DN
		INC ZEROS ; else increment ZEROS by one
		DN: LOOP UP
		;decrement rotation counter by 1 and if not zero then go to up
		MOV CX, ZEROS ;move result in cx register.
		MOV AH, 4CH
		INT 21H
		ENDS
		END ; end of program.
			Program Code: 4 M

	d)	**Write an ALP to subtract two BCD number using procedure.**	**4 M**
	Ans	.model small .data num1 db 13h num2 db 12h	Program Code: 4 M

		ends
		.code
		start:
		mov ax,@data
		mov ds,ax
		call sub1
		sub1 proc near
		mov al,num1
		mov bl,num2
		sub al,bl
		das
		sub1 endp
		mov ah,4ch
		int 21h
		ends
		end start
		end	

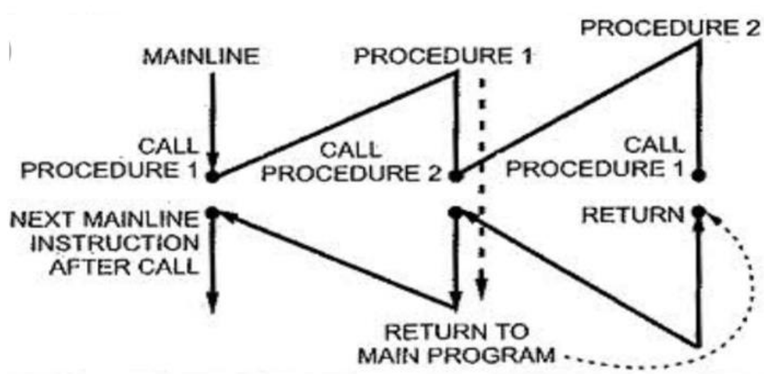
	e)	**Describe re-entrant and recursive procedure with suitable diagram.**	**4 M**
	Ans	**1)Recursive procedure**: A **recursive procedure** is procedure which calls itself. This results in the procedure call to be generated from within the procedures again and again. The **recursive procedures** keep on executing until the termination condition is reached. The recursive procedures are very effective to use and to implement but they take a large amount of stack space and the linking of the procedure within the procedure takes more time as well as puts extra load on the processor.	Recursive procedure: 2 M Re-entrant procedures: 2 M



2) Re-entrant procedures:

In some situation it may happen that Procedure 1 is called from main program Procrdure2 is called from procedure1And procedure1 is again called from procdure2. In this situation program execution flow re enters in the procedure1. These types of procedures are called re-entrant procedures.

A procedure is said to be re-entrant, if it can be interrupted, used and re-entered without losing or writing over anything.



5.		**Attempt any <u>TWO</u> of the following:**	**12 M**
	a)	**(a) Calculate the physical address if:** **(i) CS 1200H and IP = DE00OH** **(ii) SS = FFOOH and SP = 0123H** **(iii) DS 1IFOOH and BX= IA00H for MOV AX, [BX]**	**6 M**
	Ans	Physical address = segment address x 10H + offset address (i) Physical address = CS X 10H + IP = 1200H X 10H + DE00H = 12000H + DE00H	Each correct answer 2 M

		= 1FE00H (ii) Physical address = SS X 10H + SP = FF00H X 10H + 0123H = FF000H + 0123H = FF123H (iii) Physical address = DS X 10H + BX = 1F00H X 10H + 1A00H = 1F000H + 1A00H = 20A00H	
	b)	**Describe how an assembly language program is developed and debugging using program developments tools.**	**6 M**
	Ans	Assembly language development tools: **EDITOR:** It is a program which helps to construct assembly language program with a file extension .asm, in right format so that the assembler will translate it to machine language. It enables one to create, edit, save, copy and make modification in source file. **Assembler:** Assembler is a program that translates assembly language program to the correct binary code. It also generates the file called as object file with extension .obj. It also displays syntax errors in the program, if any. **Linker:** It is a programming tool used to convert Object code (.OBJ) into executable (.EXE) program. It combines, if requested, more than one separated assembled modules into one executable module such as two or more assembly programs or an assembly language with C program. **Debugger:** Debugger is a program that allows the execution of program in single step mode under the control of the user. The errors in program can be located and corrected using a debugger. Debugger generates .exe file.	Each development tool 1.5 M
	c)	**State the addressing mode of following instructions:** **(i) MOV AX, 3456H** **(ii) ADD BX, [2000H]**	**6 M**

		(iii) DAA **(iv) MOV AX, [Si]** **(v) MOV AX, BX** **(vi) SUB AX, [BX +SI +80H]**	
	Ans	(i) MOV AX , 3456H ------- IMMEDIATE ADDRESSING MODE (ii) ADD BX , [2000H] ------- DIRECT ADDRESSING MODE (iii) DAA ------ IMPLIED ADDRESSING MODE (iv) MOV AX , [SI] ------- INDEXED ADDRESSING MODE (v) MOV AX , BX ------- REGISTER ADDRESSING MODE (vi) SUB AX , [BX+SI+80H] ----- BASE RELATIVE INDEX ADDRESSING MODE	Each correct answer 1 M
6.		**Attempt any <u>TWO</u> of the following:**	**12 M**
	a)	**Describe how string instructions are used to compare two strings with suitable example.**	**6 M**

	Ans	CMPS /CMPSB/CMPSW: Compare string byte or Words.
		
		Syntax:
		CMPS destination, source
		CMPSB destination, source
		CMPSW destination, source
		Operation: Flags affected < ----- DS:[SI]- ES:[DI]
		It compares a byte or word in one string with a byte or word in another string. SI holds the offset of source and DI holds offset of destination strings. CX contains counter and DF=0 or 1 to auto increment or auto decrement pointer after comparing one byte/word.
		e.g.
			Explanation of string compare instruction 4 M
			
			And
			
			Example 2 M

Example	Explanation
CMPS m8, m8	Compares byte at address DS: SI with byte at address ES: DI and sets the status flags accordingly.
CMPS m16, m16	Compares word at address DS:SI with word at address ES:DI and sets the status flags accordingly.
CMPSB	Compares byte at address DS:SI with byte at address ES:DI accordingly.
CMPSW	Compares word at address DS:SI with word at address ES:DI and sets the status flags accordingly.

	b)	**Write an instructión to perform following operations:** **(i) Multiply BL by 88H** **(ii) Signed division of AL by BL** **(iii) Move 4000H to DS register** **(iv) Rotate content of AX register to left 4 times.** **(v) Shift the content of BX register to right 3 times.** **(vi) Load SS with FF0OH.**	**6 M**
	Ans	(1) Multiply BL by 88h **MOV AL, 88H** **MUL BL** (2) Signed division of AL by BL **IDIV BL** (3) Move 4000H to DS register **MOV DS, 4000H** (4) Rotate content of AX register to left 4 times **MOV CL,04** **ROL AX, CL** (5) Shift the content of BX register to right 3 times **MOV CL,03H** **SHR BX, CL** (6) Load SS with FF00H **MOV AX, FF00H** **MOV SS, AX**	Each correct answer 1 M
	c)	**Write an ALP to concatenate two strings.**	**6 M**
	Ans	DATA SEGMENT STR1 DB "hello$" STR2 DB "world$" DATA ENDS CODE SEGMENT START: ASSUME CS: CODE, DS:DATA MOV AX,@ DATA MOV DS, AX	Correct program 6 M

		MOV SI, OFFSET STR1
		NEXT: MOV AL, [SI]
		CMP AL,'$'
		JE EXIT
		INC SI
		JMP NEXT
		EXIT: MOV DI, OFFSET STR2
		UP: MOV AL, [DI]
		CMP AL, "$"
		JE EXIT1
		MOV [SI], AL
		INC SI
		INC DI
		JMP UP
		EXIT1: MOV AL,'$'
		MOV [SI], AL
		MOV AH, 4CH
		INT 21H
		CODE ENDS
		END START	