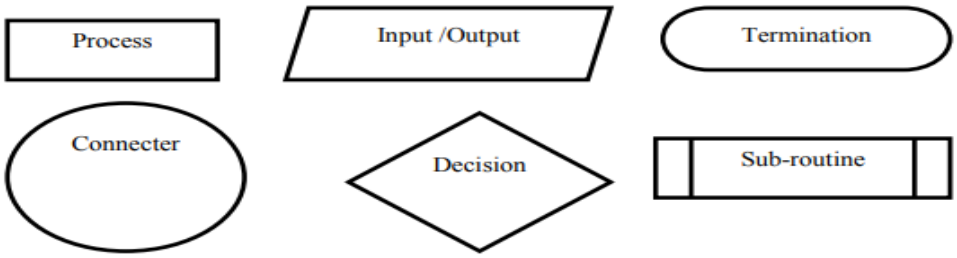
MODEL ANSWER

SUMMER– 18 EXAMINATION

Subject Title:MICROPROCESSOR AND PROGRAMMING **Subject Code:** 17431

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.

Q. No.	Sub Q.N.	Answer	Marking Scheme
Q.1		**Attempt any FIVE of the following;**	**20 Total Marks**
	a)	**Draw the symbols used in a flowchart while developing ALP, Mention the use of each symbol. (any 4)**	**4 Marks**
	Ans:		**1 Mark for Each Symbol**
	b)	**State the function of the following pin of 8085 microprocessor :** (i) ALE (ii) INTR and $\overline{INTA}$ (iii) $IO/\overline{M}$ (iv) $\overline{Reset\ IN}$	**4 Marks**
	Ans:	(i) ALE :- It is positive going pulse generated every time at the beginning of **Memory or I/O** operation when bus contains address, ALE is high , else it it low.	**1 Mark for each**

		(ii) $\overline{INTR}$ and $\overline{INTA}$ **:-** INTR is a level triggered , non vectored interrupt. Whenever INTR interrupt is given microprocessor acknowledges it by sending INTA active low signal. (iii) $IO/\overline{M}$ **:-** This signal is used to differentiate between I/O & memory operation .When it is high, it indicates I/O operation and when it is low, it indicate memory operation. (iv) $\overline{Reset\ IN}$ **:-**Active low Signal , activated during manual reset or power on Reset .This signal resets the µp. On reset Program Counter (PC) contain 0000H.Hence,Reset Vector address of 8085 is 0000H.	
	c)	**State the use of OF, TF, AF and PF flags in 8086.**	**4 Marks**
	Ans:	**Overflow flag:** This flag is set if an overflow occurs. **Trap flag:** If this flag is set the processor enters the single step execution mode **Auxiliary carry flag:**This is set if there is a carry from the lowest nibble, i.e. bit three during addition or borrow for the lowest nibble i.e. bit three during subtraction. **Parity flag:** This flag is set to 1if the lower byte of the result contains even numbers of 1s.	**1 Mark for Each**
	d)	**List of silent features of Intel 8085 Microprocessor.**	**4 Marks**
	Ans:	**Salient features of 8085:** 1. 16 address line so 2^{16}=64 Kbytes of memory can be addressed. 2. Operating clock frequency is 3MHz and minimum clock frequency is 500KHz. 3. On chip bus controller. 4. Provide 74 instructions with five addressing modes. 5. 8085 is 8 bit microprocessor. 6. Provides 5 level hardware interrupts and 8 software interrupts. 7. It can generate 8 bit I/O address so 2^8 =256 input and 256 output ports can be accessed. 8. Requires a single +5 volt supply.	**1/2 Marks for each**
	e)	**Write assembly language instruction of 8086 microprocessor to** **(i) Add 100H to the contents of AX register.** **(ii) Rotate the contents of AX towards left by 2 bits.**	**4 Marks**

	Ans:	**(i) Add 100H to the contents of AX register.**
		ADD AX,0100H
		
		(ii) Rotate the contents of AX towards left by 2 bits.
		MOV CL,02H
		ROL AX, CL
			2 Marks for Each

	f)	**State the function of STC and CMC instruction of 8086.**	**4 Marks**
	Ans:	STC instruction sets the carry flag CF=1	**2 Marks for Each**

		CMC instruction complements the carry flag CF=~CF	**Instruction**
	g)	**State the names of segment registers in 8086 microprocessor.**	**4 Marks**
	Ans:	CS – Code Segment – holds base address for all executable instructions in a program. SS -Stack segment- holds the Base address of the stack. DS – Data Segment – default base address for variables. ES – Extra Segment – additional base address for memory variables in extra segment.	**1 Mark for Each**
Q 2		**Attempt any FOUR of the following:**	**16 Marks**
	a)	**State all the control signals generated by S_0, S_1, S_2 with their functions.**	**4 Marks**
	Ans:	$\overline{S_0}, \overline{S_1}, \overline{S_2}$ of 8086 are given as input for 8288 to produce command signals as shown in table.	**4 Marks**

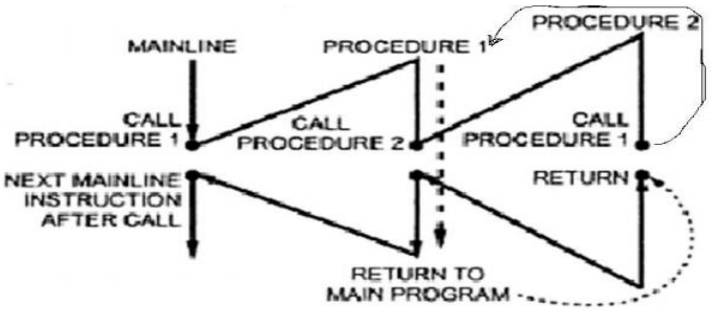
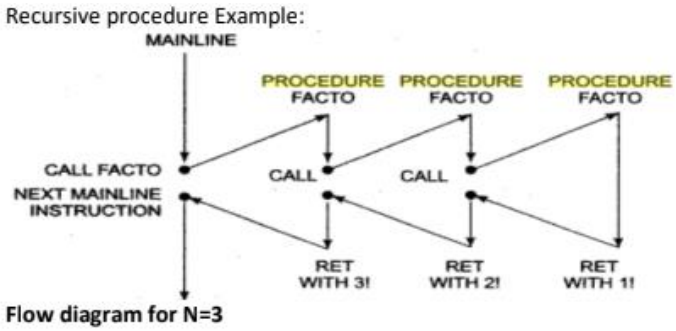
$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	Processor state	82C88 command
0	0	0	Interrupt Acknowledge	$\overline{INTA}$
0	0	1	Read I/O Port	$\overline{IORC}$
0	1	0	Write I/O Port	$\overline{IOWC}$, $\overline{AIOWC}$
0	1	1	Halt	None
1	0	0	Code Access	$\overline{MRDC}$
1	0	1	Read Memory	$\overline{MRDC}$
1	1	0	Write Memory	$\overline{MWTC}$, $\overline{AMWC}$
1	1	1	Passive	None

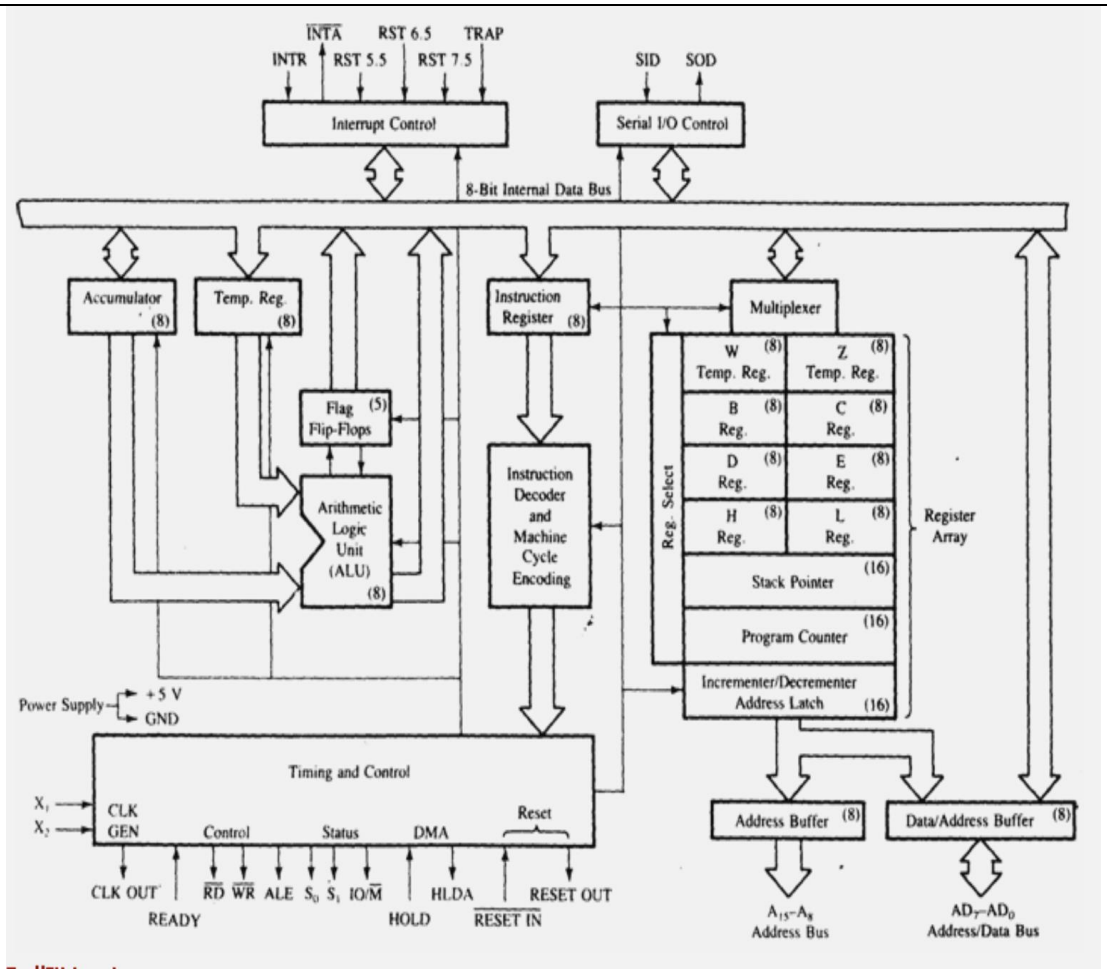
- **$\overline{MRDC}$ (Memory Read command) :It is used to read data from memory**
- **$\overline{MWTC}$(Memory Write Command): It is used to write data to memory**
- **$\overline{IORC}$(I/O Read command): It is used to read data from I/O**
- **$\overline{IOWC}$(I/O write command): It is used to write data into IO**
- **$\overline{AMWTC}$(Advanced Memory Write Command): It is similar to $\overline{MWTC}$ but it is activated one clock cycle earlier. This gives slow memory device an extra clock cycle to prepare itself to accept data.**
- **$\overline{AIOWC}$(Advanced I/O Write Command: It is similar to $\overline{IOWC}$ but it is activated one clock cycle earlier. This gives slow I/O device an extra clock cycle to prepare itself to accept data.**

	b)	**Name the general purpose register of 8086, give brief description of each.**	**4 Marks**
	Ans:	1. AX (Accumulator) – Used to store the result for arithmetic / logical operations All I/O data transfer using IN & OUT instructions use "A" register(AH / AL or AX). 2. BX – Base – used to hold the offset address or data in indirect addressing mode. 3. CX – acts as a counter for repeating or looping instructions. 4. DX – Used with AX to hold 32 bit values during multiplication and division. Used to hold address of I/O port in indirect addressing mode.	**1 Mark for Each**
	c)	**Compare 8085 microprocessor and 8086 microprocessor (with respect to)** **(i) No. of data line** **(ii) No. of address line** **(iii) Frequency of operation.** **(iv)Registers**	**4 Marks**

	Ans:	<table><tr><th>Parameter</th><th>8085</th><th>8086</th></tr><tr><td>No. of data line</td><td>8 bits</td><td>16 bits</td></tr><tr><td>No. of address line</td><td>16 bits</td><td>20 bits</td></tr><tr><td>Frequency of operation.</td><td>Min clock speed 500 KHZ ,Max Clock speed can vary 3 MHZ with 50% duty cycle.</td><td>Clock speed can vary 5 ,8 ,10 MHZ with 33%duty cycle.</td></tr><tr><td>Registers</td><td>PC & SP = Memory Registers A, B, C, D, E, H, L = General Purpose Registers</td><td>SI, DI, BX, SP, BP = Memory Offset Registers AX, BX, CX, DX = General Purpose Registers.</td></tr></table>	**1 Mark for Each**
	d)	**State function of following assembly language programming tool.** **(i) Assembler** **(ii) Linker**	**4 Marks**
	Ans:	**i) Assembler** 1. Assembler is a program that translates assembly language program to the correct binary code. 2. It also generates the file called as object file with extension .obj. 3. It also displays syntax errors in the program, if any. 4. It can be also be used to produce list(.lst) and .crf files. **ii) Linker** 1) It is a programming tool used to convert Object code into executable program. 2) It combines ,if requested ,more than one separated assembled modules into one executablemodule such as two or more assembly programs or an assembly language with C program. 3) It generates .EXE module	**2 Marks for Each**
	e)	**Explain with suitable example the instruction given below:** **(1) DAA** **(ii) AAM**	**4 Marks**
	Ans:	**(i) DAA is** Decimal Adjust after BCD Addition) This instruction is used to make sure the result of adding two packed BCD numbers is adjusted tobe a correctBCD number. The result of the addition must be in AL for DAA instruction to work correctly. If the lower nibble in AL after addition is > 9 or Auxiliary Carry Flag is set, then add	**2 Marks for Each**

		6 to lowernibble of AL. Ifthe upper nibble in AL is > 9 or Carry Flag is set, and then add 6 to upper nibble of AL. Example: - (Any Same Type of Example) if AL=99 BCD and BL=99 BCD Then ADD AL, BL 1001 1001 = AL= 99 BCD + 1001 1001 = BL = 99 BCD --- 0011 0010 = AL =32 H and CF=1, AF=1 --- After the execution of DAA instruction, the result is 0011 0010 =AL =32H AF =1 + 0110 0110 --- 1001 1000 =AL =98 in BCD& CF = 1 --- **ii) <u>AAM Instruction:</u>** (BCD Adjust After Multiply). After the two unpacked BCD digits are multiplied, the AAM instruction is used to adjust the product to two unpacked BCD digits in AX. **Examples: (Any other Equivalent Example should be given correct)** MUL CL ; AL = 0000 0100 = Unpacked BCD 4 ; CL = 0000 0110 = Unpacked BCD 6 ; AX = 0000 0000 0001 1000 = 0018H ; AL x CL Result in AX. AAM ; AX = 0000 0010 0000 0100 = 02 04H Which is unpacked BCD for 24.	
f)	**What do you mean by procedure? Explain re-entrant and recursive procedure.**	**4 Marks**
Ans:	The repeated group of instructions in a large program can be written separately from the main program.This subprogram is called as Procedure in an assembly language programming i.e. Procedure is a set ofstatements that can be processed independently from the main program. **re-entrantprocedure** The procedure which can be interrupted, used and "reentered" withoutlosing or writing over anything is called re-entrant procedure. In some situation it may happen that procedure1 is called from main program, procedure2 is called fromprocedure1 is again called from procedure2. In this situation program execution flow reenters in theprocedure1. These types of procedures are	**Definition 1 Mark, re-entrant 1 ½ Marks, recursive 1 ½ Marks**

		called reentrant procedures. The flow of program execution for reentrant procedure is shown in the diagram.  **recursive procedure** A recursive procedure is a procedure which calls itself. Recursive procedures are used to work withcomplex data structure called trees. If the procedure is called with N (recursion depth) = 3. Then the n isdecremented by one after each procedure is called until n = 0. Fig shows the flow diagram and pseudo-code for recursive procedure.  	
Q. 3		**Attempt any FOUR of the following:**	**16 Marks**
	a)	**Draw a neat labelled function block diagram of 8085. State the function of ALU.**	**4 Marks**
	Ans:		**(Diagram : 3Marks; Any one function 1Mark)**



Functions:

1. ALU performs the arithmetic operations and logical operation such as Add, Subtract, AND, OR,XOR on 8 bit data.
2. It stores the answer in the accumulator.

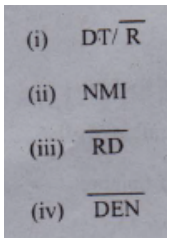
b) **Differentiate between following instructions:** **4 Marks**

(i) ROL RCL
(ii) ADD ADC
(iii)MOV LXI
(iv) JMP JNC

Ans: **(Any one Difference :1Mark)**

ROL	**RCL**
Rotate left byte or word	Rotate through carry left byte or word
Syntax: ROL Destination, Count	Syntax: RCL Destination, Count
Can be used to Swap the nibbles	Cannot be used to swap the nibbles.

ADD	**ADC**
Add byte or word	Add byte or word with carry flag

		<table><tr><td>Syntax : ADD Destination, Source</td><td>Syntax : ADC Destination, Source</td></tr><tr><td>Destination □□Source +Destination</td><td>Destination □□Source +Destination + Carry</td></tr></table> <table><tr><td>**MOV**</td><td>**LXI**</td></tr><tr><td>Transfer byte or word from source to destination</td><td>Used in 8085.</td></tr></table> <table><tr><td>**JMP**</td><td>**JNC**</td></tr><tr><td>Unconditional jump instruction</td><td>Conditional jump instruction</td></tr><tr><td>JMP Causes IP to be modified so that the nextinstruction is fetched from the location specified incode segment</td><td>JNC Causes short jump , after comparison if thecondition is true, based on the current contents ofthe status flag as carry flag</td></tr><tr><td>Allows jumping to location without any flagchecking.</td><td>Allows Short Jump to if Carry flag is set to 0.</td></tr></table>	
c)	**State the function of following pins of 8086 microprocessor.** 		**4 Marks**
Ans:	**(i)DT/ R :-** This output line is used to decide the direction of data flow through the transceivers(bidirectional buffer). When the processor sends the data, this signal is high and when theprocessor is receiving data, this signal is low. **(ii) NMI** An edge triggered signal on this pin causes 8086 to interrupt the program it is executing and executeInterrupt service Procedure. NMI is Non-maskable by software. **(iii) RD** This is an active low output control signal used to read data from memory or I/O device generated by the microprocessor. **(iv)DEN**		**(Correct one function : 1 Mark each)**

		This is active low signal, to indicate availability of valid data over AD0-AD15. Used to enable transreceivers(bi-directional buffers) 8286 or 74LS245 to separate data frommultiplexed address/data signal .	
	d)	**Write an ALP to add l6 bit BCD number.**	**4 Marks**

	Ans:	DATA SEGMENT
		N1 DW 2804H
		N2 DW 4213H
		BCD_SUM DW ?
		DATA ENDS
		CODE SEGMENT
		ASSUME CS: CODE, DS:DATA
		START:
		MOV AX, DATA
		MOV DS, AX
		MOV AX, N1
		MOV BX, N2
		ADD AL,BL
		DAA ; LOWER BYTE ADDITION
		MOV CL,AL
		MOV AL,AH
		ADD AL,BH
		DAA ; HIGHER BYTE ADDITION
		MOV CH,AL
		MOV BCD_SUM, CX
		MOV AH,4CH
		INT 21H
		CODE ENDS
		END START
			(Correct Program:4 Mark)

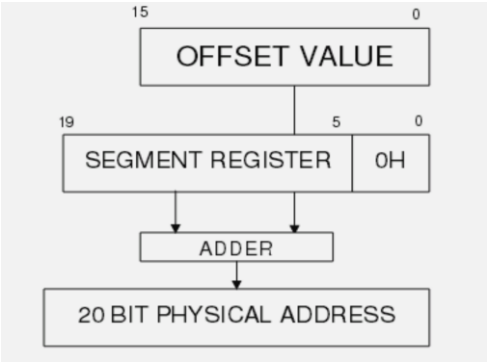
	e)	**Write an ALP to transfer a block of 10 data bytes using string instruction.**	**4 Marks**

	Ans:	DATA SEGMENT
		STRNO1 DB 10 DUP(10) (Any Value in STRNO1 should be given correct)
		DATA ENDS
		EXTRA SEGMENT
		STRNO2 DB 10 DUP(0)
		EXTRA ENDS
		CODE SEGMENT
		ASSUME CS: CODE, DS: DATA, ES: EXTRA
		START:
		MOV DX, DATA
		MOV DS, DX
		MOV DX, EXTRA
		MOV ES, DX
			(Correct Program: 4 marks)

		LEA SI, STRNO1
		LEA DI, STRNO2
		MOV CX, 000AH
		CLD
		REP MOVSB
		MOV AH, 4CH
		INT 21H
		CODE ENDS
		END START	

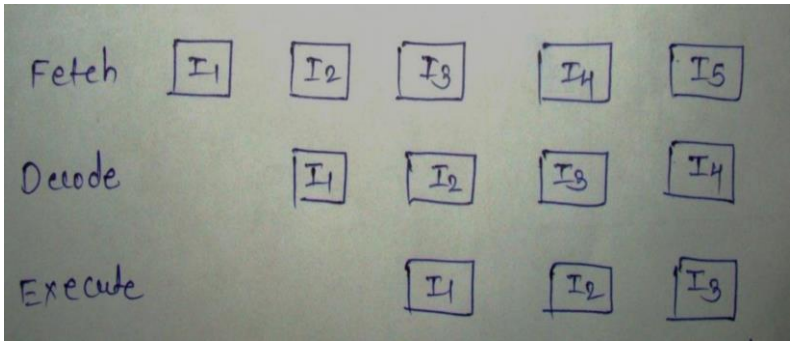
	f)	**Define MACRO with example.**	**4 Marks**

	Ans:	**Macro**
		• Small sequence of the codes of the same pattern are repeated frequently at different placeswhich perform the same operation on the different data of same data type, such repeated codecan be written separately called as Macro.
		• When assembler encounters a Macro name later in the source code, the block of codeassociated with the Macro name is substituted or expanded at the point of call, known asmacro expansion.
		• Macro called as open subroutine.
		
		Syntax:
		Macro_name MACRO[arg1,arg2,…..argN)
		…..
		Endm
		
		Example:
		MyMacro MACRO p1, p2, p3 ; macro definition with arguments
		MOV AX, p1
		MOV BX, p2
		MOV CX, p3
		ENDM ;indicates end of macro.
		
		data segment
		
		data ends
		
		code segment
		assume cs:code,ds:data
		
		start:
		mov ax,data
		mov ds,ax
		MyMacro 1, 2, 3 ; macro call
			Definition or Syntax : 2 Marks ,
			
			Macro Example :2 Marks

		MyMacro 4, 5, DX mov ah,4ch int 21h code ends end start **(OR Any Same Type of Example can be considered)**	
Q. 4		**Attempt any FOUR of the following:**	**16 Marks**
	a)	**Identify the addressing modes for the following instruction:** **(i) MOV CL,34 H** **(ii) MOV BX,[4172 H]** **(iii) MOV DS,AX** **(iv) MOV AX,[SI+BX+04]**	**4 Marks**
	Ans:	(i) MOV CL,34 H – Immediate Addressing Mode (ii) MOV BX,[4172 H]-Direct Addressing Mode (iii)MOV DS,AX-Register Addressing Mode (iv)MOV AX,[SI+BX+04] –Relative Base index addressing mode.	**(Each Addressing Mode -1M)**
	b)	**List the steps in physical address generation in 8086 microprocessor. Calculate the physical address for the given CS=3420H, IP=689AH.**	**4 Marks**
	Ans:	The 8086 addresses a segmented memory. The complete physical address which is 20-bits long is generated using segment and offset registers each of the size 16-bit.The content of a segment register also called as segment address, and content of an offset register also called as offset address. To get total physical address, put the lower nibble 0H to segment address and add offset address. The figure shows formation of 20-bit physical address.  Calculate the physical address for the given CS=3420H,IP=689AH. CS=3420H IP=689A H Zero is inserted ↓ 3 4 2 0 0 +6 8 9A 3A A9A	**(Description – 2 Marks, Calculation - 2 Marks)**

	c)	**With suitable example, explain following instruction:** **(i) INC** **(ii) XLAT** **(iii) XCHG** **(iv) AND**	**4 Marks**
	Ans:	**(i) INC** This instruction adds 1 to the indicated destination. The destination can be a register or memory location. Immediate data cannot be an operand of the instruction. Destination ← destination +1 INC AX increment the content of AX by 1. **(ii)XLAT** XLAT replaces a byte in AL register with a byte from 256 byte lookup table beginning at [BX] . AL is used as offset into this table. **Operation :- AL□[BX+AL]** **(iii)XCHG Destination, Source** This instruction exchanges Source with Destination. .It cannot exchange two memory locations directly. The source and destination can be any of the general purpose register or memory location, but not two locations simultaneously. No segment registers can be used. E.g.: XCHG DX, AX XCHG BL, CH XCHG AL,[9800] **(iv)AND (Logical AND)** This instruction logically ANDs each bit of the source byte or word with the corresponding bit in the destination and stores result in the destination. **Syntax:** AND destination, source **Examples:** AND BH,CL ;AND byte in CL with Byte in BH, result in BH. AND BX,00FFH ;AND word in BX with immediate data 00ffH AND [5000H],DX;AND word in DX with a word in memory with offset 5000 in DS.	**(1 Mark each)**
	d)	**Write an ALP for BCD to hex conversion.**	**4 Marks**
	Ans:	DATA SEGMENT BCD DB 56D HEX DB ? DATA ENDS CODE SEGMENT	**(Correct Program -4 Marks, Any other logic may**

		ASSUME CS:CODE, DS:DATA
		START:
		MOV AX,DATA
		MOV DS,AX
		MOV AL,BCD
		MOV AH,00H
		MOV BL,10H
		DIV BL
		MOV DL,AH
		MOV AH,00H
		DIV BL
		MOV CL,04H
		ROR AH,CL
		OR DL,AH
		MOV HEX,DL
		MOV AH,4CH
		INT 21H
		CODE ENDS
		END START
		
		OR
		
		DATA SEGMENT
		DEC_NUM DB 56
		HEX_NUM DW 0
		MULT_FAC DW 3e8H
		DIGIT_COUNT DW 2
		DATA ENDS
		CODE SEGMENT
		ASSUME CS:CODE,DS:DATA
		START:MOV AX,DATA
		MOV DS,AX
		MOV BX,0AH
		MOV CX,DIGIT_COUNT
		MOV SI,OFFSET DEC_NUM
		UP: MOV AL,[SI]
		AND AX,000FH
		MUL MULT_FAC
		ADD HEX_NUM,AX
		MOV AX,MULT_FAC
		MOV DX,00
		DIV BX
		MOV MULT_FAC,AX
		INC SI
		LOOP UP
		ENDS
			be considered)

		END START	
	e)	**State the advantages of pipeline architecture.**	**4 Marks**
	Ans:	In 8086, pipelining is the technique of overlapping instruction fetch and execution mechanism. • To speed up program execution, the BIU fetches as many as six instruction bytes ahead of time from memory. The size of instruction prefetch queue in 8086 is 6 bytes**.** • While executing one instruction other instruction can be fetched. Thus it avoids the waiting time for execution unit to receive other instruction. • BIU stores the fetched instructions in a 6 level deep FIFO . The BIU can be fetching instructions bytes while the EU is decoding an instruction or executing an instruction which does not require use of the buses. • When the EU is ready for its next instruction; it simply reads the instruction from the queue in the BIU. • This improves overall speed of the processor. 	**(correct advantage :1mark each)**
	f)	**Write assembly language program to divide two 16 bit unsigned numbers.**	**4 Marks**
	Ans:	Note: Since 8086 Performs 32bit / 16 bit division or 16bit / 8 bit division therefore for Two 16bit Number division we have to perform 32bit / 16bit Division. Program for Double word by word division. DATA SEGMENT NUMBER1 DD00004359H NUMBER2 DW1199H Quotient DW 1 DUP(0) Remainder DW 1 DUP(0) DATA ENDS	(**Data Declaration 1Mark; Correct Program:3 Marks)**

		CODE SEGMENT
		ASSUME CS: CODE, DS: DATA
		START:
		MOV DX , DATA
		MOV DS ,DX
		LEA SI, NUMBER1 ; Moving 32 Bit Number into DX : AX
		MOV AX , [SI]
		INC SI
		INC SI
		MOV DX, [SI]
		MOV BX ,NUMBER2 ; Moving 16 Bit Number into BX
		DIV BX; Ans = AX:Quotient,DX :Remainder
		MOV Quotient, AX
		MOV Remainder, DX
		MOV AH , 4CH
		INT 21H
		CODE ENDS
		END START	

Q.5		**Attempt any FOUR of the following.**	**16 Marks**
	a)	**Explain CALL and RET instruction.**	**4 Marks**
	Ans:	The CALL instruction is used to transfer execution to a subprogram or procedure by storing return address on stack There are two types of calls-NEAR (Inter-Segment) and FAR(Intra-segment call). Near call refers to a procedure call which is in thesame code segment as the call instruction and far call refers to a procedure call which is in different code segment from that of the call instruction. **Syntax: CALL procedure_name** (direct/indirect) **RET: (Return from procedure)** The instruction RET is used to transfer program control from the procedure back to thecalling program.i.e main program or procedural CALL. The RET instruction are of twotypes **NEAR RET(intersegment return)** **FAR RET(intra-segment return)** **Syntax:- RET**	CALL-2 Marks RET- 2 Marks
	b)	**Write an assembly language program to multiply two 8 bit number.**	**4 Marks**
	Ans:	DATA SEGMENT NUM1 DB 05H NUM2 DB 02H RESULT DW? DATA ENDS CODE SEGMENT ASSUME CS:CODE,DS:DATA	Correct program -4 marks

START:
MOV DX,DATA
MOV DS,DX
MOV AL,NUM1
MOV AH,NUM2
MUL AH
MOV RESULT,AX
CODE ENDS
END START

Correct Program with any other logic can be given marks.

c) Differentiate between minimum and maximum mode operation of 8086. — **4 Marks**

Ans:

Sr. No.	Minimum mode	Maximum mode
1.	MN/$\overline{MX}$ pin is connected to Vcc. i.e. MN/$\overline{MX}$=1.	MN/$\overline{MX}$ pin is connected to ground. i.e. MN/$\overline{MX}$ = 0.
2.	Control system M/$\overline{IO}$, $\overline{RD}$, $\overline{WR}$ is available on 8086 directly.	Control system M/$\overline{IO}$, $\overline{RD}$, $\overline{WR}$ is not available directly in 8086.
3.	Single processor in the minimum mode system.	Multiprocessor configuration in maximum mode system.
4.	In this mode, no separate bus controller is required.	Separate bus controller (8288) is required in maximum mode.
5.	Control signals such as $\overline{IOR}$, $\overline{IOW}$, $\overline{MEMW}$, $\overline{MEMR}$ can be generated using control signals M/$\overline{IO}$, $\overline{RD}$, $\overline{WR}$ which are available on 8086 directly.	Control signals such as $\overline{MRDC}$, $\overline{MWTC}$, $\overline{AMWC}$, $\overline{IORC}$, $\overline{IOWC}$ and $\overline{AIOWC}$ are generated by bus controller 8288.
6.	ALE, $\overline{DEN}$, DT/$\overline{R}$ and $\overline{INTA}$ signals are directly available.	ALE, $\overline{DEN}$, DT/$\overline{R}$ and $\overline{INTA}$ signals are not directly available and are generated by bus controller 8288.
7.	HOLD and HLDA signals are available to interface another master in system such as DMA controller.	$\overline{RQ}$/$\overline{GT0}$ and $\overline{RQ}$/$\overline{GT1}$ signals are available to interface another master in system such as DMA controller and coprocessor 8087.
8.	Status of the instruction queue is not available.	Status of the instruction queue is available on pins QS_0 and QS_1.

Any four points-1 mark each

d) Write an assembly language program to add the series of 5 number — **4 Marks**

Ans:

DATA SEGMENT

NUM1 DB10H,20H,30H,40H,50H

RESULT DB 1 DUP(0)

CARRY DB 0H

Correct program-4 marks

		DATA ENDS
		CODE SEGMENT:
		ASSUMECS:CODE,DS:DATA
		START:
		MOV DX,DATA
		MOV DS,DX
		MOV CL,05H
		MOV SI,OFFSET NUM1 OR LEA SI, NUM1
		UP:MOV AL,[SI]
		ADD RESULT,AL
		JNC NEXT
		INC CARRY
		NEXT: INC SI
		LOOP UP OR DEC CL,
		JNZ UP
		
		MOV AH,4C00H
		INT 21H
		CODE ENDS
		END START
		Correct Program with any other logic can be given marks.	

	e)	**Write a procedure to find factorial of a number.**	**4 Marks**

	Ans:	DATA SEGMENT
		A DW 0005H
		FACT_LSB DW?
		FACT_MSB DW?
		DATA ENDS
		CODE SEGMENT
		ASSUME DS:DATA,CS:CODE
		START:MOV AX,DATA
		MOV DS,AX
		CALL FACTORIAL
		MOV AH,4CH
		INT 21H
		FACTORIAL PROC
			Correct program-4 marks

```
MOV AX,A
MOV BX,AX
DEC BX
UP: MUL BX              ; MULTIPLY AX*BX
MOV FACT_LSB,AX         ;ANS DX:AX PAIR
MOV FACT_MSB,DX
DEC BX
CMP BX,0
JNZ UP
RET
FACTORIAL ENDP
```

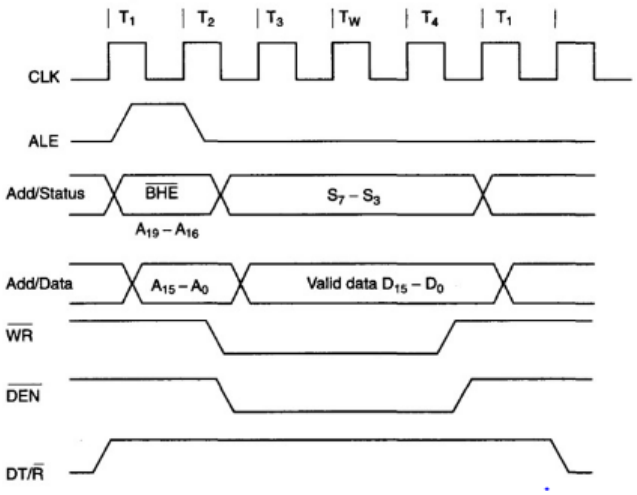
OR

```
DATA SEGMENT
      NUM DB 05H
      RES DW ?
DATA ENDS
CODE SEGMENT
   ASSUME CS:CODE,DS:DATA
START:
      MOV AX,DATA
      MOV DS,AX
      CALL FAC
      MOV RES,AX
      MOV AH,4CH
      INT 21H
 FAC   PROC
         MOV CL,NUM
         DEC CL
         MOV AL,NUM
         MOV AH,00H
         MOV BL,CL
         MOV BH,00H
  L1:   MUL BX
         DEC BX
         DEC CL
         JNZ L1
         RET
  FAC   ENDP
CODE ENDS
END START
```

Correct Program with any other logic can be given marks.

	f)	**Describe various string instructions in brief.**	**4 Marks**
	Ans:		**1 mark**

		1] MOVS/ MOVSB/ MOVSW - Move String byte or word. Syntax **MOVS destination, source** MOVSB**destination, source** MOVSW**destination, source** **Operation: ES:[DI]<----- DS:[SI]** It copies a byte or word a location in data segment to a location in extra segment. The offset of source is pointed by SI and offset of destination is pointed by DI.CX register contain counter and direction flag (DF) will be set or reset to auto increment or auto decrement pointers after one move. **2] 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. **3] SCAS/SCASB/SCASW: Scan a string byte or word.** *Syntax* **SCAS/SCASB/SCASW** **Operation: Flags affected < ----- AL/AX-ES: [DI]** It compares a byte or word in AL/AX with a byte /word pointed by ES:DI. The string to be scanned must be in the extra segment and pointed by DI. CX contains counter and DF may be 0 or 1. When the match is found in the string execution stops and ZF=1 otherwise ZF=0 **4] LODS/LODSB/LODSW**: Load String byte into AL or Load String word into AX. **Syntax: LODS/LODSB/LODSW** **Operation: AL/AX < ----- DS: [SI]** IT copies a byte or word from string pointed by SI in data segment into AL or AX.CX may contain the counter and DF may be either 0 or 1 **6] STOS/STOSB/STOSW (Store Byte or Word in AL/AX)** **Syntax** STOS/**STOSB/STOSW** **Operation: ES:[DI] < -----AL/AX** It copies a byte or word from AL or AX to a memory location pointed by DI in extra segment CX may contain the counter and DF may either set or reset. • **Operation: ES:[DI] < -----AL/AX**	**each instruction**

Q.6		Attempt any FOUR of the following.	16 Marks
	a)	**Draw the timing diagram of minimum mode memory write cycle.**	4 Marks
	Ans:	**Timing diagram of minimum mode memory write cycle**  	Correct labelled diagram-4 marks
	b)	**Write an ALP to count the number of '1' in a 16 bit number. Assume the number to be stored in BX register. Store the result in CX register.**	4 Marks

	Ans:	DATA SEGMENT
		NUM DW 0FF33H ;BINARY{ 1111 1111 0011 0011}
		ONES DB 0
		DATA ENDS
		CODE SEGMENT
		ASSUME CS:CODE,DS:DATA
		START:
		MOV AX,DATA
		MOV DS,AX
		MOV CX,16 OR MOV CX, 10H ;rotation counter
		MOV AX,NUM
		UP:ROR AX,1
		JNC DN;IF no CARRY loop
		INC ONES; else increment1's count
		DN:LOOP UP OR DEC CX ;decrement rotation counter
		JNZ UP
		
		MOV CX,ONES
		MOV AH,4CH
		INT 21H
		CODE ENDS
		END START
		Correct Program with any other logic can be given marks.
			Correct program-4 marks

	c)	**Compare between JUMP and CALL instruction in 8086 microprocessor.**	4 Marks

	Ans:		**Any four points-1 mark each**

JUMP	CALL
It is used to transfer control of execution to the specified address using 16-bit displacement or CS:IP	It is used to transfer program control to the subprogram or subroutine
Two types of jump 1) Conditional jump 2) Non-conditional jump	There are2 types of jump a)near intersegment b)far(intra segment)
If target of **JMP** is in the same code segment, it requires only IP to be changed to transfer control to the target location and known as NAER JUMP and for different code segment CS:IP value is required and known as FAR JUMP.	When 8086 executes **CALL** instruction ol IP value is pushed on to stack for NEAl CALL and old CS: IP value is pushed on t stack for FAR CALL.
Examples: JMP down ; unconditional JNC down ; conditional	Examples: 1.CALL delay 2. CALL show 3. CALL FAR PTR show

Note: Any other Correct Difference should be given Marks

d)	**Describe following assemble directive :** **(i) DB** **(ii) ASSUMS** **(iii) SEGMENT** **(iv) EQU**	**4 Marks**
Ans:	**(i)DB :Define byte(8 bits)** It is used to declare a byte type variable of 8 bit. It also can be used to declare an array of bytes.The range of values that can be stored in a byte is 0 to 255 for unsigned numbers and -128 to +127 for signed numbers. Example:NUM DB ? ; Allocate one memory location. **(ii)ASSUME:** Assume directive is used to tell Assembler the name of the logical segment it should use for the specified segment.When program is loaded the processor segment register should point to the respective logical segments. Example: - Assume CS: MSBTE_CODE, DS: MSBTE_DATA **(iii)SEGMENT:** Used to indicate the beginning of logical segment . Preceding the SEGMENT directive is the name you want to give the segment Syntax: **Segment_Name SEGMENT [Word/Public]** **(iv) EQU: Equate to** The EQU directive is used to declare the micro symbols to which some constant value is assigned. Microassembler will replace every occurrence of the symbol in a program by its value. **Syntax: \Symbol_name EQU expression** **Eg. NUM EQU 50**	**1 mark each**

<table>
<tr><td></td><td>e)</td><td>How many times LOOP1 will be executed in following program? What will be the contents of BL after the execution?
MOV BL, 00H
MOV CL, 05H
LOOP1 : ADD BL, 02H
DEC CL
JNZ LOOP1</td><td>4 Marks</td></tr>
<tr><td></td><td>Ans:</td><td>LOOP1 will be executed 5 times in the above program
The contents of BL will be 0Ah after the execution of program.</td><td>EACH Answer 2 Marks</td></tr>
<tr><td></td><td>f)</td><td>Differentiate between NEAR and FAR CALLS.</td><td>4 Marks</td></tr>
<tr><td></td><td>Ans:</td><td>
<table>
<tr><th>Sr. no</th><th>Near CALL</th><th>Far CALL</th></tr>
<tr><td>1.</td><td>A near call is in the same code segment from that of the call instruction.</td><td>A far call is in the different code segment from that of the call instruction.</td></tr>
<tr><td>2.</td><td>It is also called intra-segment call</td><td>It is also called inter-segment call .</td></tr>
<tr><td>3</td><td>A near call replaces the old IP with new IP.</td><td>A far call replaces the old CS:IP pairs with new CS:IP pairs</td></tr>
<tr><td>4.</td><td>The value of old IP is pushed on to the stack.
SP=SP-2 ;Save IP on stack(address of procedure)</td><td>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)</td></tr>
<tr><td>5.</td><td>Less stack locations are required</td><td>More stack locations are required</td></tr>
<tr><td>6.</td><td>Example :- Call Delay</td><td>Example :- Call FAR PTR Delay</td></tr>
</table>
<u>Note : Any other Correct Difference should be given Marks.</u>
</td><td>Any four points-1 mark each.</td></tr>
</table>