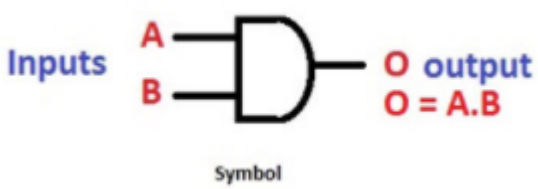
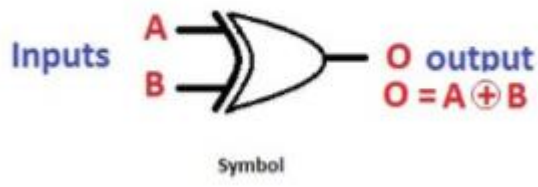
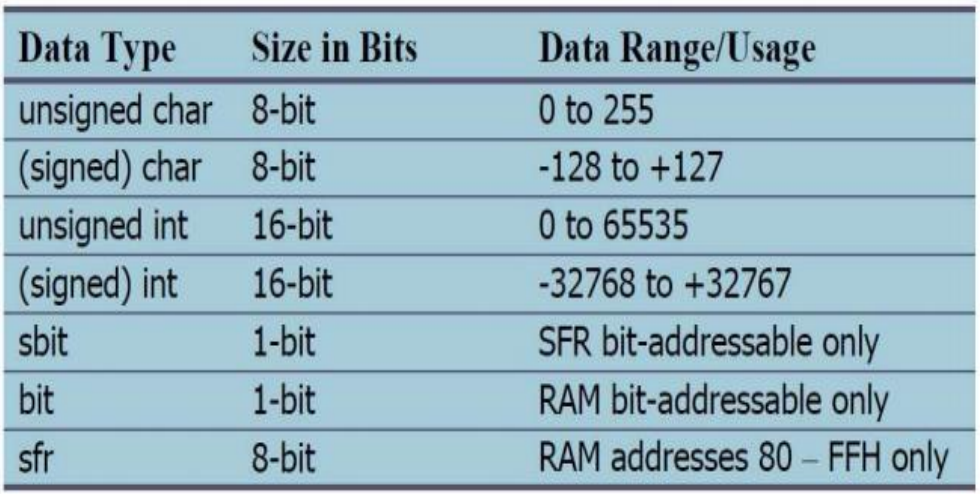
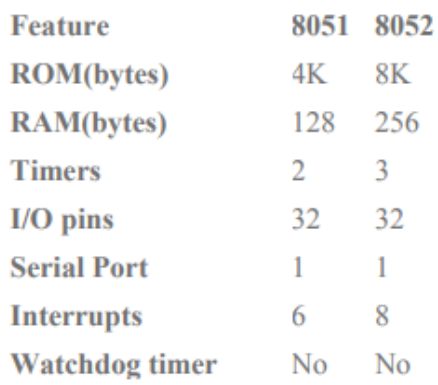
MODEL ANSWER
WINTER– 17 EXAMINATION

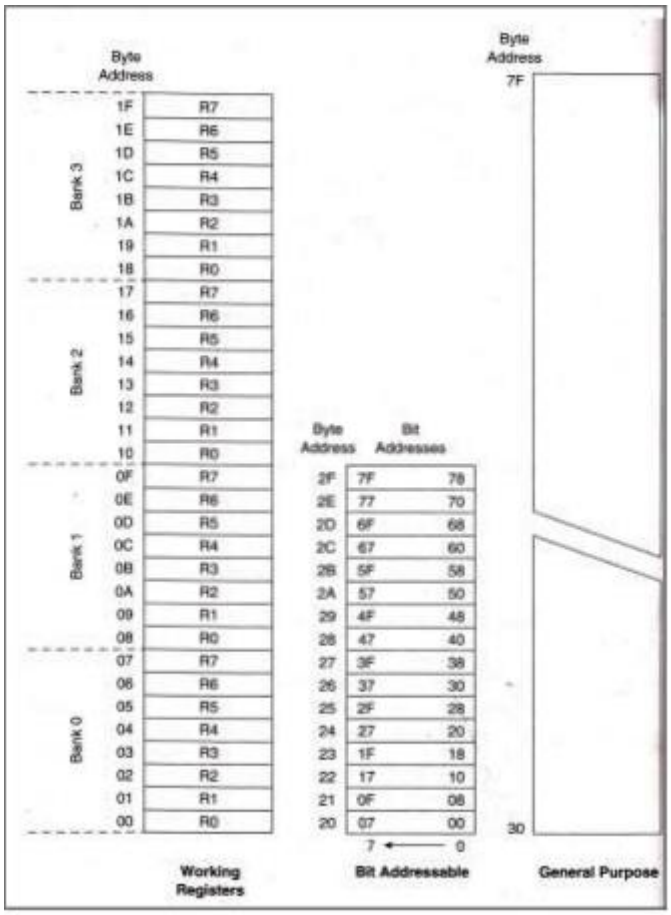
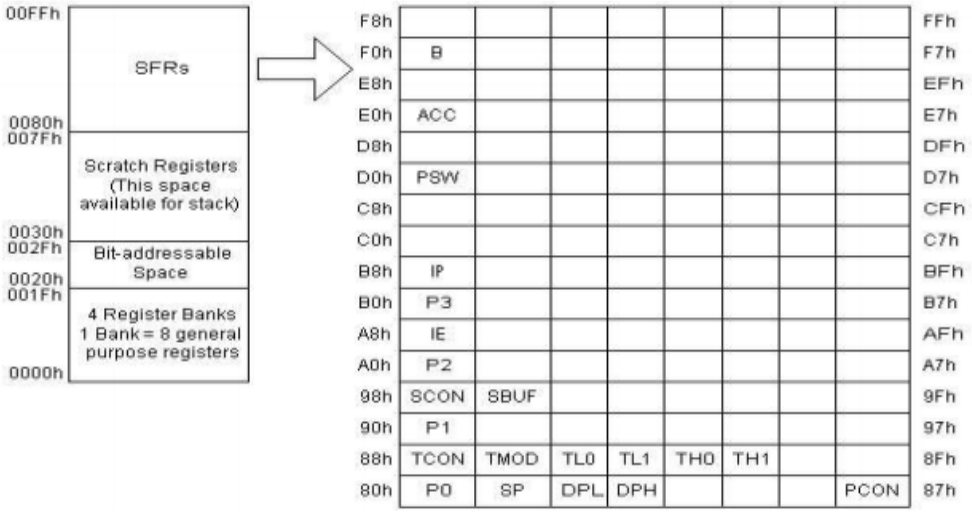
Subject Title: Microcontroller and applications **Subject Code: 17509**

Important Instructions to examiners:

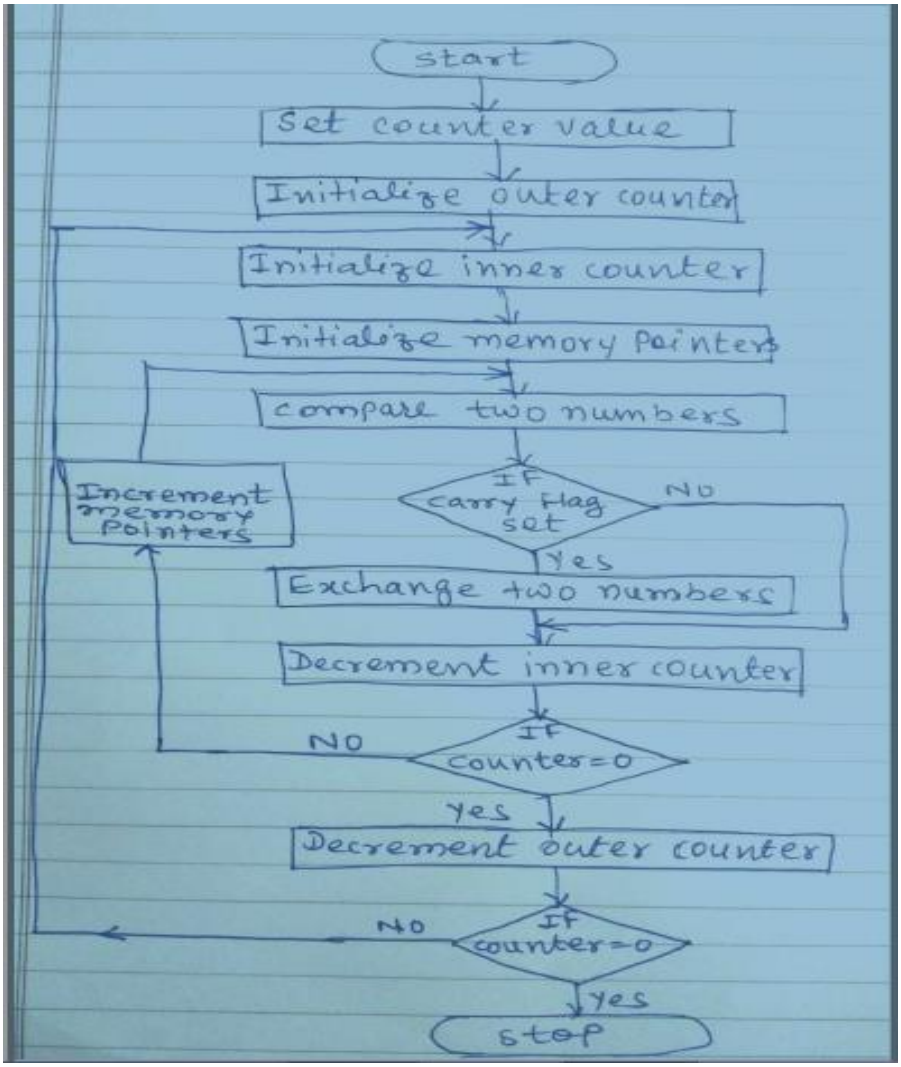
1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling 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	(A)	**Attempt any THREE:**	12-Total Marks
	a)	**Draw symbol and write truth table for AND and EXOR gate.**	4M
	Ans:	**Symbol and truth table of AND gate**  Symbol Inputs: A, B — Output: O 0 0 0 0 1 0 1 0 0 1 1 1 Truth table **Symbol and truth table of EXOR gate**  Symbol Inputs: A, B — Output: O 0 0 0 0 1 1 1 0 1 1 1 0 Truth table	Each symbol 1M Each TT 1M
	b)	**Draw the format of SCON register.**	4M

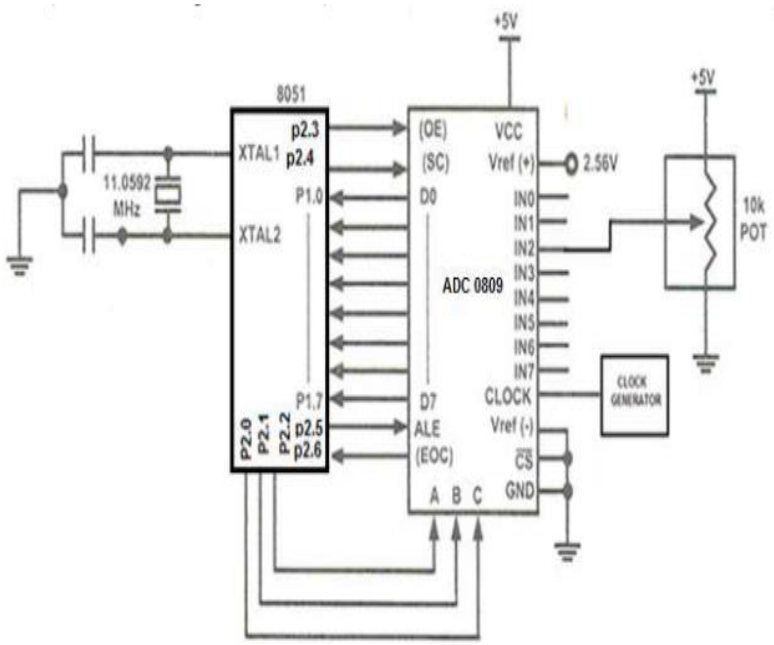
	Ans:	SCON Register format- 	**Correct format 4M**
	c)	**List any four C-data types with its range and size.**	**4M**
	Ans:		**Any 4-1M each**
	d)	**Compare 8051 and 8052 microcontrollers (four points).**	**4M**
	Ans:		**Any 4 points-1M each**
	B)	**Attempt any one.**	**6M**
	a)	**Draw the structure of internal RAM of 8051.**	**6M**
	Ans:		**Correct labeled diagram**

		 Byte Address Bank 3 1F R7 1E R6 1D R5 1C R4 1B R3 1A R2 19 R1 18 R0 Bank 2 17 R7 16 R6 15 R5 14 R4 13 R3 12 R2 11 R1 10 R0 Bank 1 0F R7 0E R6 0D R5 0C R4 0B R3 0A R2 09 R1 08 R0 Bank 0 07 R7 06 R6 05 R5 04 R4 03 R3 02 R2 01 R1 00 R0 Working Registers Byte Address Bit Addresses 2F 7F 78 2E 77 70 2D 6F 68 2C 67 60 2B 5F 58 2A 57 50 29 4F 48 28 47 40 27 3F 38 26 37 30 25 2F 28 24 27 20 23 1F 18 22 17 10 21 0F 08 20 07 00 7 ← 0 Bit Addressable Byte Address 7F 30 General Purpose **(OR)**  00FFh SFRs 0080h 007Fh Scratch Registers (This space available for stack) 0030h 002Fh Bit-addressable Space 0020h 001Fh 4 Register Banks 1 Bank = 8 general purpose registers 0000h F8h FFh F0h B F7h E8h EFh E0h ACC E7h D8h DFh D0h PSW D7h C8h CFh C0h C7h B8h IP BFh B0h P3 B7h A8h IE AFh A0h P2 A7h 98h SCON SBUF 9Fh 90h P1 97h 88h TCON TMOD TL0 TL1 TH0 TH1 8Fh 80h P0 SP DPL DPH PCON 87h	**6M**

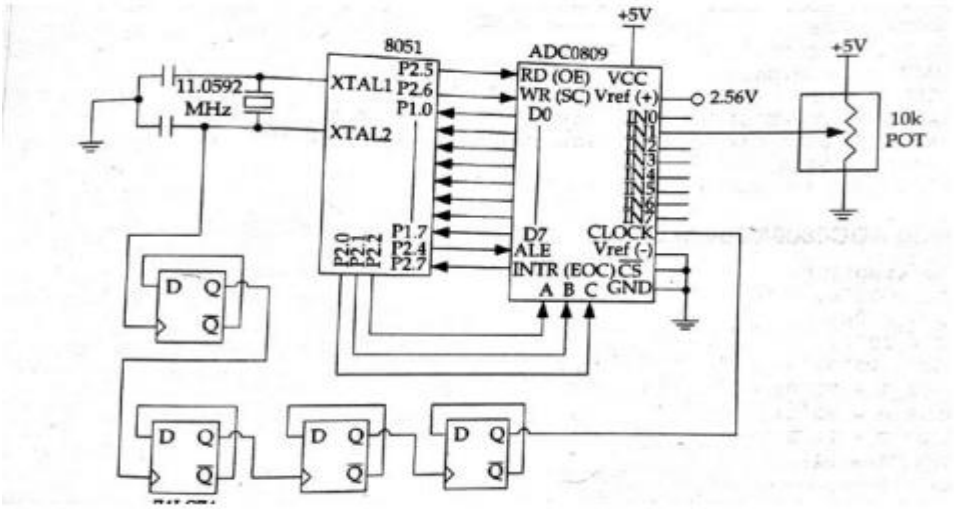
	b)	**Explain following instructions with example.** **i)MOVC A,@A+DPTR** **ii)DAA** **iii)XCHD A,@Ri**	**6M**
	Ans:	**i)MOVC A,@A+DPTR** The instruction moves data from the external code memory to the accumulator. The address of operand in this example is formed by adding the content of the DPTR register to the accumulator value. Here the DPTR value is referred to as the base address and the accumulator value is referred to as the index address. No of bytes: 1 byte .Addressing mode: register **ii)DAA** DA A -Decimal Adjust Accumulator DA adjusts the contents of the Accumulator to correspond to a BCD (Binary Coded Decimal) number after two BCD numbers have been added by the ADD or ADDC instruction. If the auxiliary carry bit is set or if the value of bits 0-3(lower nibble)exceed 9, 0x06 is added to the accumulator. If the carry bit is set or if the value of bits 4-7 (higher nibble) exceed 9, 0x60 is added to the accumulator. No of bytes: 1 byte .Addressing mode: register For eg. A=0Ch After execution A=12 BCD **iii)XCHD A,@Ri** Exchange Digit-Exchanges bits 0-3 of the Accumulator with bits 0-3 of the Internal RAM address pointed to indirectly by R0 or R1. Bits 4-7 of each register are unaffected. No of bytes: 1 byte .Addressing mode: register indirect.	**Each instruction 2M**

Q 2		Attempt any two.	16M
	(a)	**Write an ALP to arrange 10 bytes in internal RAM location in ascending order.**	8M
	Ans:	**Flow chart for ascending order:** 	**Flowchart 3M** **Program 5M**

		Program: **MOV R3,#09H** **L2:MOV R2,#09H** **MOV R0,#30H** **L1:MOV A,@R0** **INC R0** **SUBB A,@R0** **JC L0** **MOV A,@R0** **DEC R0** **XCH A,@R0** **INC R0** **MOV @R0,A** **L0:DJNZ R2,L1** **DJNZ R3,L2** **END**	
	(b)	**Interface ADC 0809 to 8051 and write c program to read analog data at CH7 and convert it to digital**	**8M**
	Ans:	**Interfacing of ADC with 8051:**	**Diagram 4M** **Program 4M**



(OR)

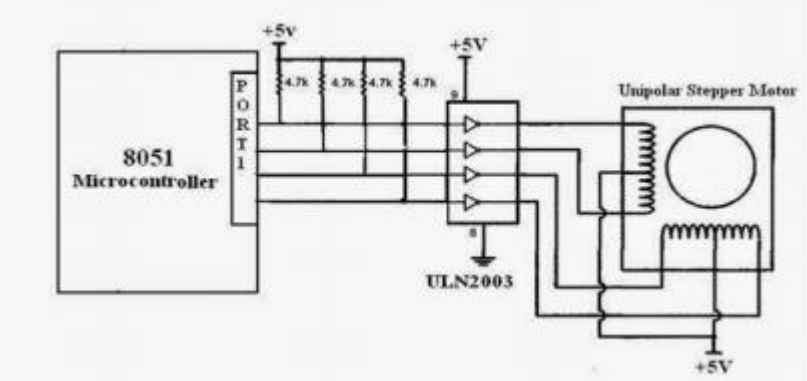
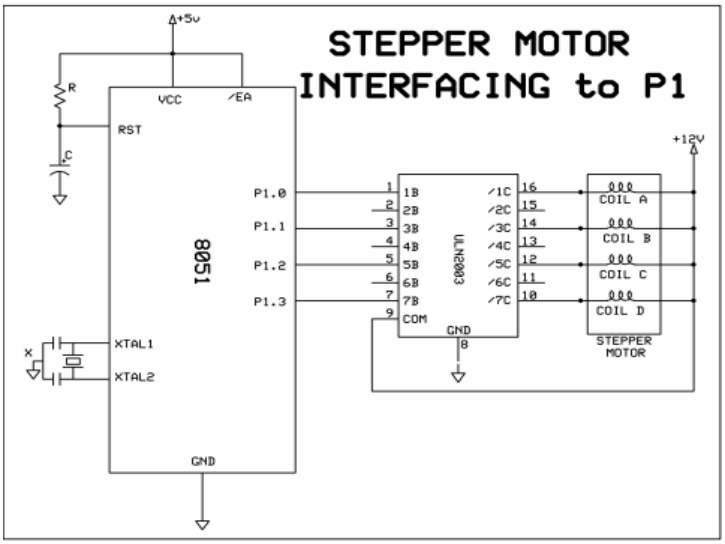


Program:

```
#include <reg 51.h>
sbit ALE = P2^4;
sbit OE = P2^5;
```

		sbit SC = P2^6;
		sbit EOC = P2^7;
		sbit ADDR_A = P2^0;
		sbit ADDR_B = P2^1;
		sbit ADDR_C = P2^2;
		Sfr MYDATA = P1;
		Void MSdelay(void);
		void main()
		{
		unsigned char value;
		MYDATA = OxFF;
		EOC = 1;
		ALE = 0;
		OE = 0;
		SC = 0;
		while(1)
		{ADDR_C = 1;
		ADDR_B = 1;
		ADDR_A = 1;
		MSDelay(1);
		ALE = 1;
		MSDelay(1);
		SC = 1;
		MSDelay (1);
		ALE = 0;
		SC = 0;
		while(EOC==1);
		while(EOC==0);
		OE = 1;
		MSDelay(1);
		value = MYDATA;
		OE = 0;
		} }
		Void Msdelay(unsigned int itime)
		{
		Int i,j;
		for(i=0;i<itime;i++)
		{
		for (j=0;j<1275;j++);
		}
		}	

	(c)	**Interface stepper motor with 8051 and explain logic to rotate it clockwise by 360. Assume step angle 1.8.(No program).**	**8M**
	Ans:	**Stepper motor interfacing-**	**Diagram 4M**

		 **(OR)**  **Logic : (any other similar logic can be considered)** • Step angle is defined as the minimum degree of rotation with a single step. • No of steps per revolution = 360° / step angle • step angle = 1.8° • No of steps per revolution = 200. • The coils need to be energized for the rotation. This can be done by sending a bits sequence to one end of the coil from microcontroller port while the other end is commonly connected. • The bit sequence sent can make either one phase ON or two phase ON for a full	**Logic 4M**

		step sequence or half step sequence. • Stepper motor rotates 1.8° for each sequence • After each sequence a delay is provided and procedure is repeated for 200 steps to get the required rotation of 360.	

Q. No.	Sub. Q. No	Answer	Marking Scheme
3		**Attempt any four**	**16 marks**
	a	**Write the function of ALE and $\overline{\text{PSEN}}$ pins of 8051.**	**4 marks**
	Ans.	1. **$\overline{\text{PSEN}}$** It is active low output control signal. It is used to fetch code from external program memory by activating enable signal OE of the external ROM/ EPROM 2. **ALE** ALE (address latch enable) is an active high output pin. It is used for demultiplexing the lower order address and data. There are two ALE pulses per machine cycle. The ALE pulse is generated every time an external memory is accessed.	**2 marks each**
	b	**What will be the content of PSW after addition of 2Bh and 9Dh?**	**4 marks**
	Ans.	111 111 2Bh- 0010 1011 9Dh- 1001 1101 After adding the two 11001000 – C8h **CY = 0** (no carry) **AC = 1** (Auxiliary is 1) **P = 1** (odd no. of 1s) **OV =0** **F0, RS1, RS2 remain unchanged**	**1 mark for calculation, 1 mark each for CY, AC & P values**
	c	**Which are the components of IDE? Write function of any 4.**	**4 marks**
	Ans.	**Components of IDE**:	**List:2 Marks,**

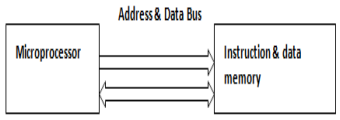
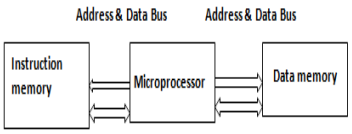
		• Compiler • Cross assembler • Cross compiler • Linker/ Locators • Loaders • Simulators • Debugger **Function: (Any four component)** **Compiler:** It is a computer program that transforms the source code written in a programming or source language into another computer language i.e. target language i.e. binary code known as object code. **Cross assembler:** It is useful to convert object codes for microcontrollers or processor to other codes for another microcontrollers or processor and vice versa. **Cross compiler:** It is used to create executable code other than one on which the compiler is run. They are used to generate executable for embedded systems or multiple platforms. **Linker/Locator:** It is used for relocation process. It is done during compilation also it can be done at run time by a relocating loader. It is a program that takes one or more objects generated by compiler and combines them into a single executable program. **Simulators:** A simulator is the software that simulates an hardware unit like emulator, peripheral, network and I/O devices on a PC. It defines a processor or processing device as well as various versions for the target system. It also monitors the detailed information as source code part with labels and symbols during the execution for each single step. It provides the detailed information of the status of memory RAM and simulated ports, simulated peripheral devices of the defined target system.	**Each Function- ½ M**

Debugger:

It is a program that is used to test and debug the target program. Debugger allows to download code to the Emulator's memory and then control from PC. It also allows the programmer to examine and modify the On-chip register and program memory data

d **Compare Von Neumann and Harvard architecture.** **4 marks**

Ans. **(Any four points)** **1 mark for each correct point**

Sr. No.	Von-Neumann architecture	Harvard architecture
1	Uses single memory for instruction and data	Uses separate memory for instruction and data
2	Same bus is used for instruction and data memory	Requires separate and Dedicated buses for instruction and data memory
3	Design is simpler	Design is complicated
4	Instruction and data has to be fetched sequentially	Instruction and data can be fetched simultaneously
5		

e **List modes of serial communication in 8051. Explain mode 3 in detail.** **4 marks**

1 mark for

Ans.

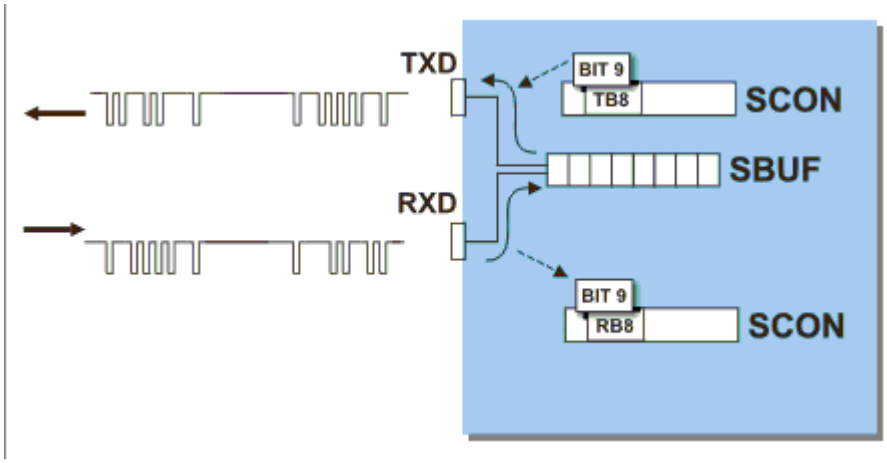
Modes of Serial communication:

SM0	SM1	Mode	Baud Rate
0	0	Mode 0	f/12
0	1	Mode 1	Variable determined by Timer 1
1	0	Mode 2	f/32 or f/64
1	1	Mode 3	Variable determined by Timer 1

Mode 3

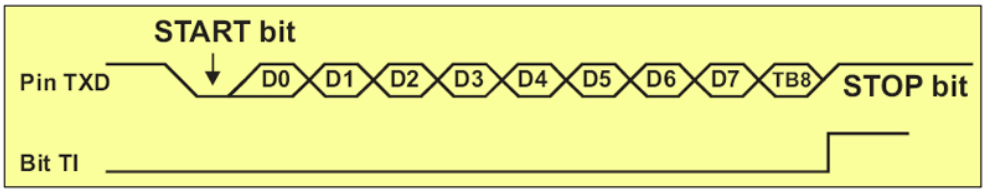
In mode 3, 11 bits are transmitted through the TXD pin or received through the RXD pin: a START bit (always 0), 8 data bits (LSB first), a programmable 9th data bit and a STOP bit (always 1). On transmit, the 9th data bit is actually the TB8 bit of the SCON register. This bit usually has a function of parity bit. On receive, the 9th data bit goes into the RB8 bit of the same register (SCON).

The baud rate is decided by Timer1 Overflow time

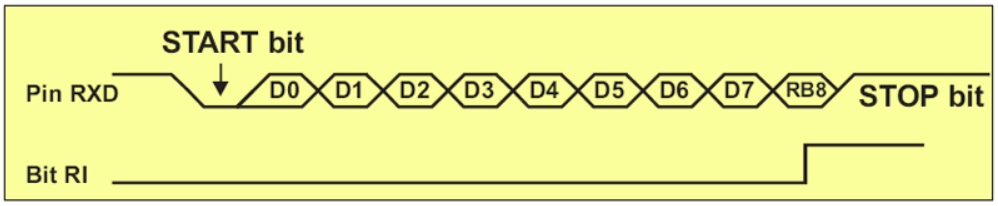


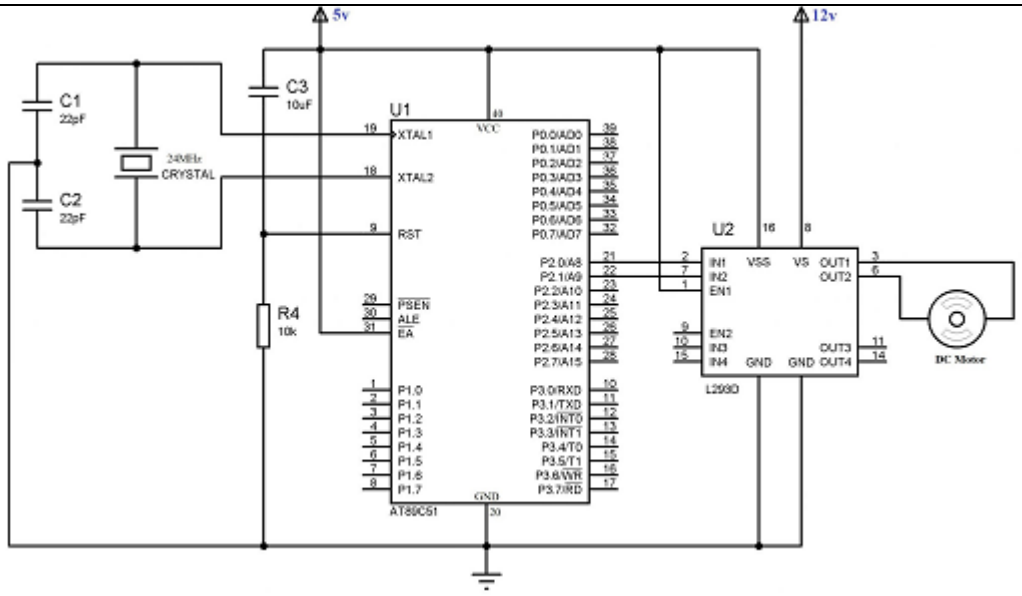
TRANSMIT – Data transmit is initiated by writing data to the SBUF register. End of data transmission is indicated by setting the TI bit of the SCON register.

list, 3 marks for correct explanation



RECEIVE – The START bit (logic zero (0)) on the RXD pin initiates data receive. The following two conditions must be met: bit REN=1 and bit RI=0. Both of them are stored in the SCON register. The RI bit is automatically set upon data reception is complete.



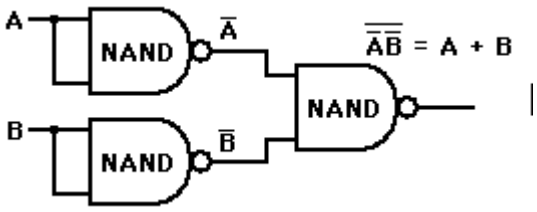
4	**A**	**Attempt any three:**	**12marks**
	a	**Interface DC motor with 8051.**	**4 marks**
	Ans		**4 marks for correct diagram**
	b	**Write C program to read P1 and send it to P2.**	**4 marks**
	Ans	'C' Language program: #include <reg51.h> void main(void) { unsigned char mybyte;	**4 marks for correct program**

```
        P1=0xFF; //make Port1 input port
        P2=0X00; // make Port2 output port
        while (1)
        {
        mybyte= P1; //get a byte from P1
        P2=  mybyte; //send compliment of it to P2
        }
}
```

c **Use NAND gate to implement AND, OR and NOT gate.** **4 marks**

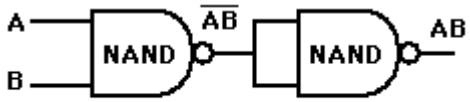
Ans.

OR gate using NAND gate

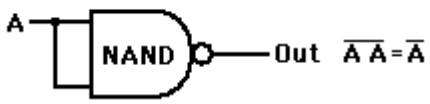


$\overline{\overline{A}}.\overline{\overline{B}} = \overline{\overline{A}} + \overline{\overline{B}} = A+B$

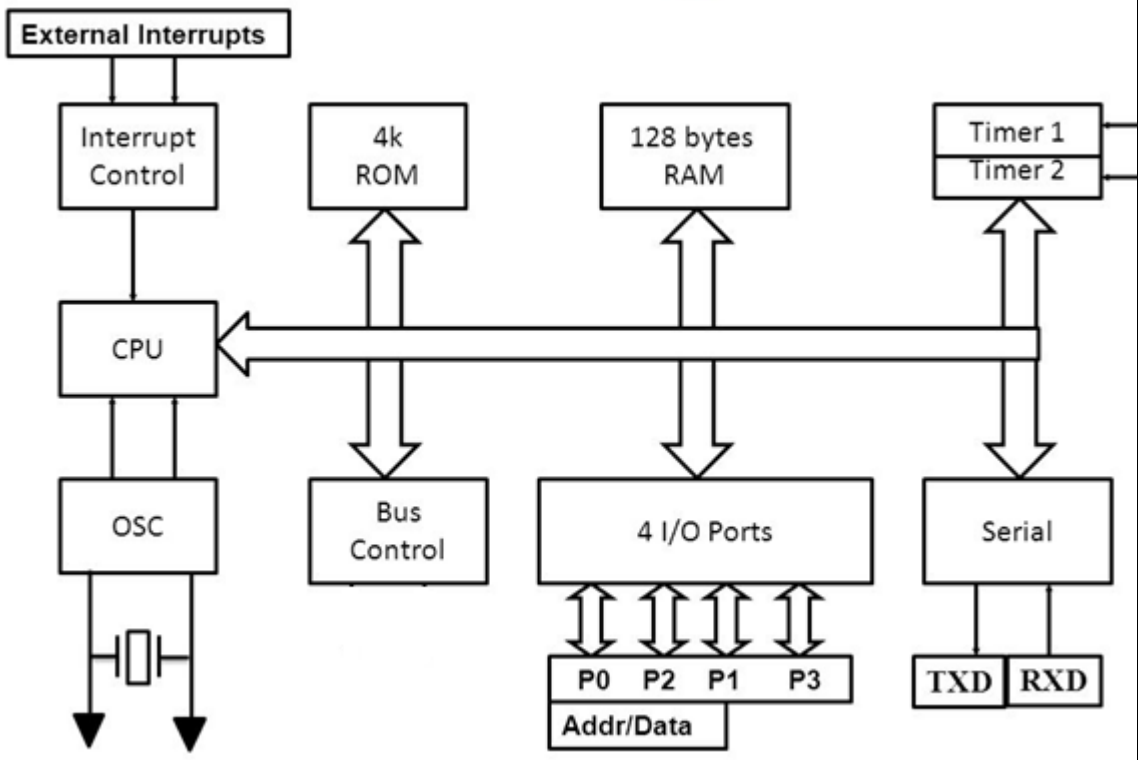
AND gate using NAND gate



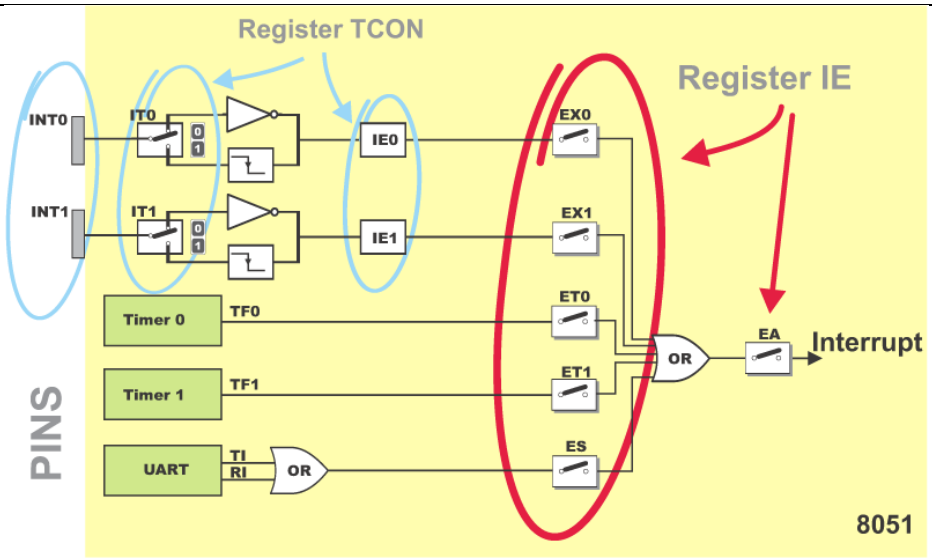
NOT gate using NAND gate

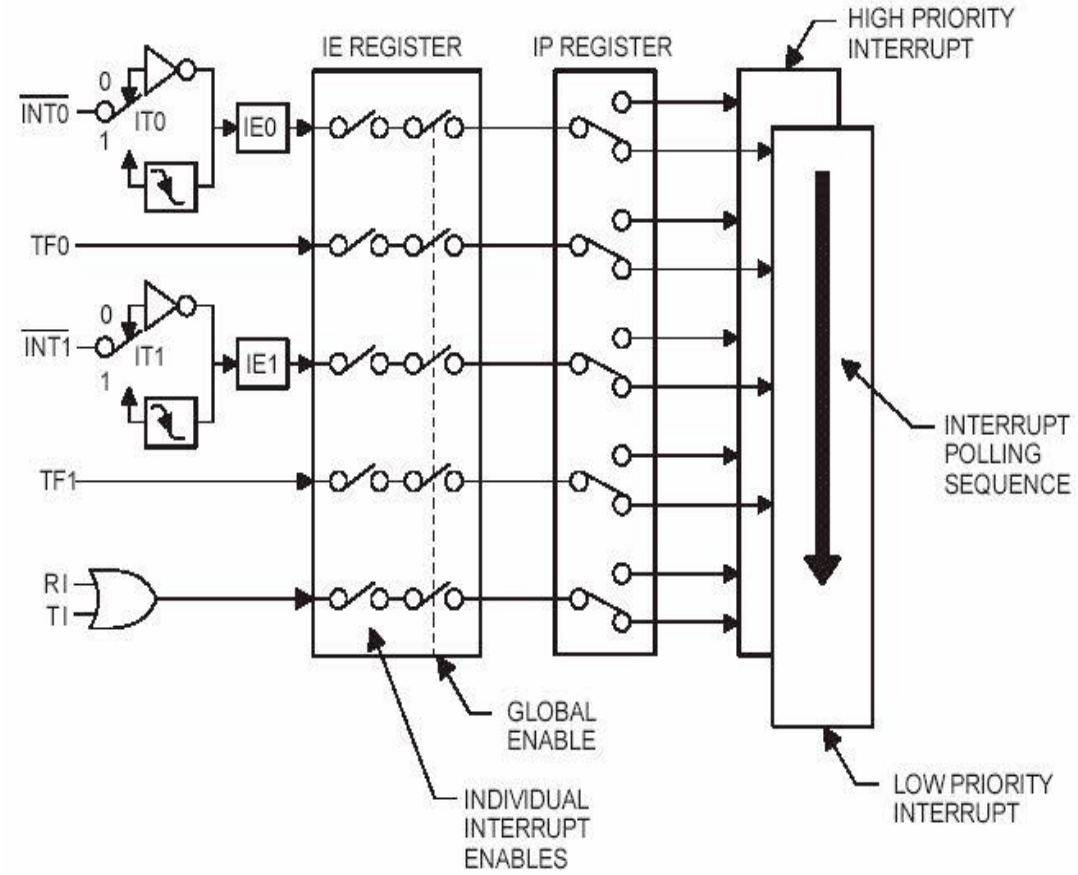


2 marks for OR and 1 mark for each AND and NOT gate

	d	**Draw block diagram of 8051.**	**4 marks**
	Ans.		**4 marks for correct diagram**
	B	**Attempt any one:**	**6 marks**
	a	**Write an ALP to generate square wave of 1 KHZ frequency. Assume f_{osc} = 12MHz**	**6 marks**
	Ans.	Square wave Frequency = 1khz Therefore Time period T = 1 / 1KHZ = 1 ms Therefore TON = TOFF = 1ms / 2 = 0.5 ms = 500 µsec Required time delay = (12 / fosc) x number of increments (N) 500 µsec = (12 / 12MHZ) x number of increments (N) = 1 µsec x N So, N = 500 Using TIMER 0 in MODE 1, COUNT = 2^{16} - N COUNT =65536 – 500 =65036	**delay calculation: 2 marks , Program : 4 marks**

		= FE0C H
		
		Assembly language program :
		ORG 0000H
		MOV TMOD, #01H ; TIMER 0, MODE 1
		REP: MOV TL0, #0CH ; Load lower byte of count
		MOV TH0, #0FEH ; Load higher byte of count
		SETB TR0 ; Start timer 0
		BACK:JNB TF0, BACK ; Check if TF0 is SET
		CLR TR0 ; Stop the Timer
		CPL P1.5 ; Complement P1.5
		CLR TF0 ; Clear TF1
		SJMP REP ; Repeat	

	b	**Draw interrupt structure of 8051 and explain it.**	**6 marks**
	Ans.	 **(OR)**	**3 marks for diagram, 3 marks for correct explanation**



When external interrupt INTX (X=0 or 1) is received,

ITX in TCON triggers the INTX, ITX=0 as level and ITX=1 as edge triggering.

IEX in TCON=1 activates the interrupt

EXX in IE=1 enables the interrupt

If EA in IE=1, interrupt is generated

For TimerX interrupt

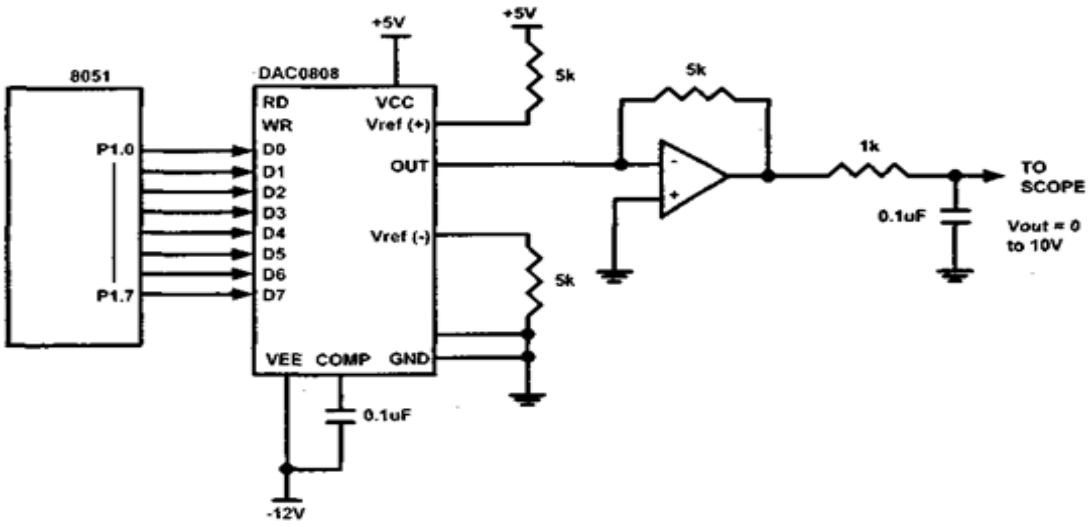
TFX in TCON = 1 activates the interrupt

ETX in IE=1 enables the interrupt

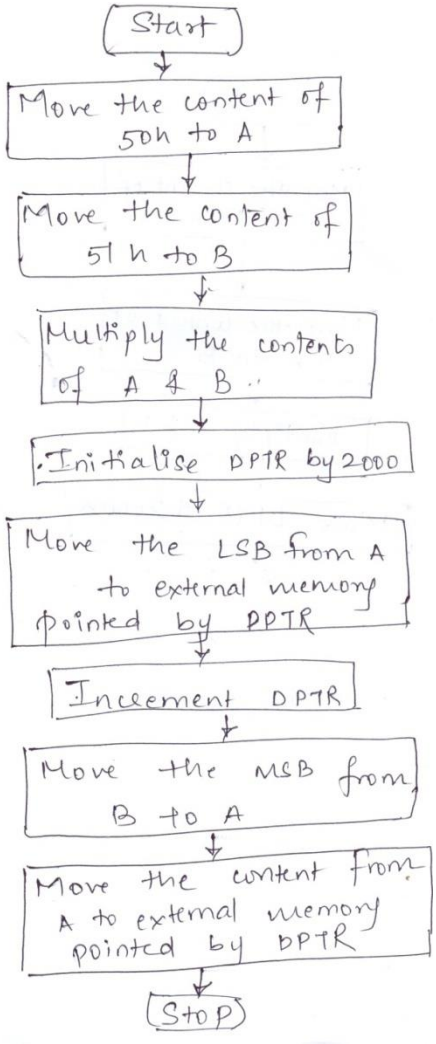
If EA in IE=1, interrupt is generated

For Serial interrupt

		RI or SI in SCON =1 activates the interrupt ES in IE=1 enables the interrupt If EA in IE=1, interrupt is generated	
Q. No.	**Sub Q.N.**	**Answer**	**Marking Scheme**
Q.5		**Attempt any TWO:**	**16M**
	a)	**Interface DAC0808 with 8051and write C-program to generate saw tooth wave on P2.1**	**8M**

	Ans:	
		#include<reg51.h>
		unsigned char d;
		void main(void)
		{
		while(1)
		{
		for(d=0; d<255; d++)
			4M

		{
		P1 = d;
		}
		}
		}
			4M

	b)	**Write an ALP to multiply content of internal RAM location 50h and 51h. Store the result at 2000h(LSB) and 2001(MSB) in external RAM.**	**8M**
	Ans:	**FLOWCHART:** 	**2M**

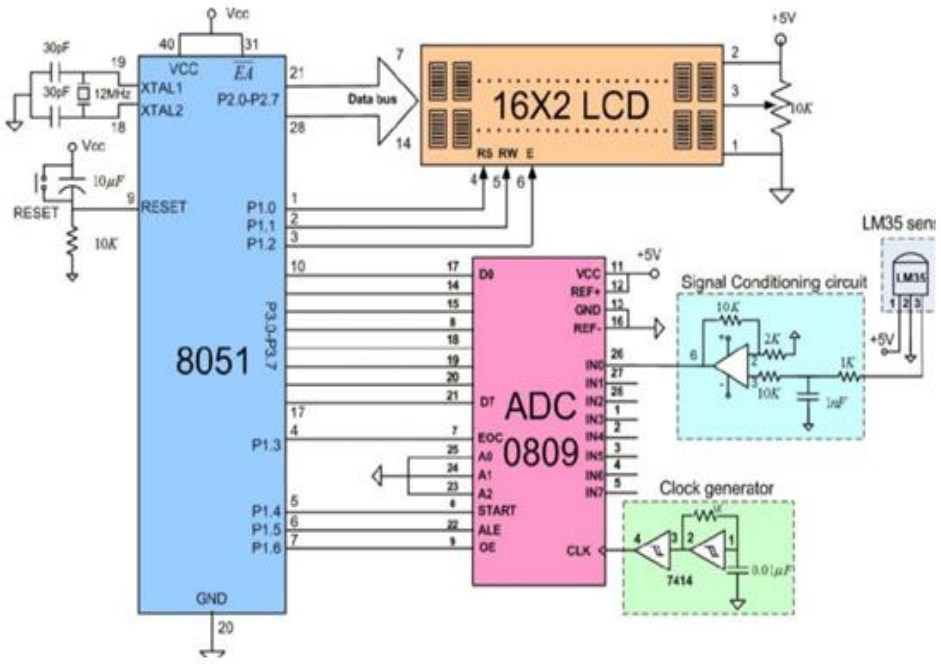
		PROGRAM:
		MOV A,50H ; Move the content of 50h to accumulator
		MOV B,51H ; Move the content of 51h to accumulator
		MUL AB ; Multiply the contents of A and B register
		MOV DPTR,#2000H ; Load DPTR with 2000h
		MOVX @DPTR,A ; Move the LSB to the External Memory pointed by DPTR
		INC DPTR ; Increment DPTR
		MOV A,B ; Move the MSB from B register to A
		MOVX @DPTR,A ; Move the MSB to the External Memory pointed by DPTR
		END ; End
			4M Program
			2M comments

	c)	**Explain bitwise shift operation for right shift and left shift with suitable examples.**	**8M**
	Ans:	**Bitwise Left Shift Operator in C:** • It is denoted by << • Bit Pattern of the data can be shifted by specified number of Positions to Left • When Data is Shifted Left , trailing zero's are filled with zero. • Left shift Operator is Binary Operator [Bi – two] **Example:** Syntax : Bitwise Left Shift Operator [variable]<<<[Number of Places] P0=0x3C<< 2 After execution of this instruction Shift number 2 times to left: 3C=0011 1100 F0=1111 0000 So, P0=0xF0 **Bitwise Right Shift Operator in C:** • It is denoted by >>	**Explanation :2M each, Syntax: 1M each, Example : 1M each**

		• Bit Pattern of the data can be shifted by specified number of Positions to Right • When Data is Shifted Right , leading zero"s are filled with zero. • Right shift Operator is Binary Operator [Bi – two]. **Example:** Syntax : Bitwise Right Shift Operator [variable]>>[number of places] P0=0x3C >> 2 After execution of this instruction Shift number 2 times to Right: 3C=0011 1100 0F=0000 1111 So, P0=0x0F **NOTE: Example May change. Please check the logic and understanding of students.**	
Q.6		**Attempt any FOUR:**	**16M**
	a)	**What is stack memory? Explain PUSH and POP instruction.**	**4M**
	Ans:	1. The stack memory is part of RAM used by the CPU to store information temporarily. 2. This information may be either data or address. 3. The CPU needs this storage area as there are only a limited amount of registers.	**Stack explanation :2M**

		4. The register used to access stack memory is called stack pointer. **PUSH Instruction:** This instruction increments Stack pointer by 1 and then copies the data at the direct address to the location pointed by SP. **POP Instruction:** This instruction copies the data at the location pointed by Stack Pointer to the direct address given in the instruction and then decrements Stack pointer by 1.	**PUSH:1M** **POP: 1M**
	b)	**Explain the function of RS,R/W and E pins in 16x4 LCD.**	**4M**
	Ans:	1. **RS:** RS is the register select pin used to write display data to the LCD (characters), this pin has to be high when writing the data to the LCD. During the initializing sequence and other commands this pin should low. 2. **R/W:** Reading and writing data to the LCD for reading the data R/W pin should be high (R/W=1) to write the data to LCD R/W pin should be low (R/W=0). 3. **EN:** Enable pin is for starting or enabling the module. A high to low pulse of about 450ns pulse is given to this pin.	**RS: 2M** **R/W: 1M** **EN: 1M**
	c)	**Explain any four assembler directives.**	**4M**
	Ans:	**DB :**The DB directive is the most widely used data directive in the assembler. It is used to define the 8-bit data. When DB is used to define data, the numbers can be in decimal, binary, hex, or ASCII formats. Example: DATA1 DB 28 (decimal data stored as 1C hex)	**1M each**

		DATA2 DB 01010101B (Binary data) DATA3 DB 5Fh Text DB "ABCDE" ASCII character array named as Text **ORG**: The ORG directive is used to indicate the beginning of the address. The number that comes after ORG can be either in hex or in decimal. If the number is not followed by H, it is decimal and the assembler will convert it to hex. Example: ORG 0000H LJMP main ORG 0030H MAIN: …. **EQU**: This is used to define a constant without occupying a memory location. When the label appears in the program, constant value will be substituted for the label. Example: NUMBER EQU 25H MOV R3,#NUMBER ; R3 = 25H as 25H will be substituted for NUMBER **END**: This indicates to the assembler the end of the source (asm) file. The END directive is the last line of an 8051 program, meaning that in the source code anything after the END directive is ignored by the assembler.	
	d)	**Write the size and function of PC and DPTR.**	**4M**
	Ans:	**PC: (Program Counter)** • It is a 16-bit register. • It holds the address of external and internal program memory.	**2M each (1M size, 1M**

		• It points next instruction to be fetched. • It is never used to hold the address of the data memory. • It cannot be used as data memory location. **DPTR: (Data Pointer)** • It is a 16-bit register. • It is used to hold the address of the external and internal data memory. • It is used to store a 16-bit data. • It is also divided into two 8-bit registers viz. DPH and DPL. • Each registers can be used to hold a 8-bit data.	**function)**
	e)	**Explain the logic to measure temperature using LM35. Draw interfacing diagram.**	**4M**
	Ans:	 • LM 35 is a temperature sensor which gives a change in output voltage of 10 mV for every degree rise in temperature i.e. if the output voltage is 350mV,the temperature ideally will be 35°C.	2M- DIAGRAM 2M - EXPLANATION

		• LM35 is connected to the input of ADC0809 or any other ADC. • ADC converts the analog output voltage of LM35 to corresponding digital value. • This digital value is proportional to the measured temperature. • It can also be converted to corresponding temperature value using look up table and can be displayed in LCD.	