SUMMER– 18 EXAMINATION

Subject Name: Micro controller **Model Answer** Subject Code: 17534

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.

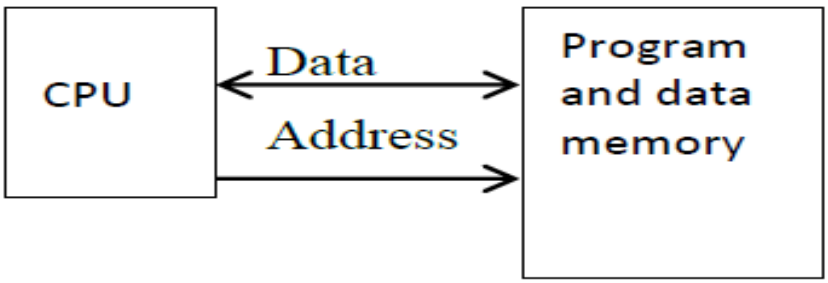
<table>
<tr><th>Q. No.</th><th>Sub Q. N.</th><th>Answers</th><th>Marking Scheme</th></tr>
<tr><td>1</td><td>a</td><td>Attempt any THREE:</td><td>12- Total Marks</td></tr>
<tr><td></td><td>i</td><td>Compare between microprocessor and microcontroller (any four points).</td><td>4M</td></tr>
<tr><td></td><td>Ans:</td><td>
<table>
<tr><th>Sr. No</th><th>Parameter</th><th>Microprocessor</th><th>Microcontroller</th></tr>
<tr><td>1</td><td>No. of instructions used</td><td>Many instructions to read/write data to/ from external memory.</td><td>Few instruction to read/ write data to/ from external memory</td></tr>
<tr><td>2</td><td>Memory</td><td>Do not have inbuilt RAM or ROM.</td><td>Inbuilt RAM /or ROM</td></tr>
<tr><td>3</td><td>Registers</td><td>Microprocessor contains general purpose registers,
Stack pointer register,
Program counter register</td><td>Microcontroller contains general purpose registers,
Stack pointer register,
Program counter register additional to that it contains Special Function Registers (SFRs) for Timer , Interrupt and serial communication</td></tr>
</table>
</td><td>1Mark each</td></tr>
</table>

			etc.
4	Timer	Do not have inbuilt Timer.	Inbuilt Timer
5	I/O ports	I/O ports are not available requires extra device like 8155 or 8255.	I/O ports are available
6	Serial port	Do not have inbuilt serial port, requires extra devices like 8250 or 8251.	Inbuilt serial port
7	Multifunction pins	Less Multifunction pins on IC.	Many multifunction pins on the IC
8	Boolean Operation	Boolean operation is not possible directly.	Boolean Operation i.e. operation on individual bit is possible directly
9	Applications	General purpose, Computers and Personal Uses.	Single purpose(dedicated application), Automobile companies, embedded systems, remote control devices.

ii **Draw neat labelled block diagram of Von-neumann and Harvard architecture.** **4M**

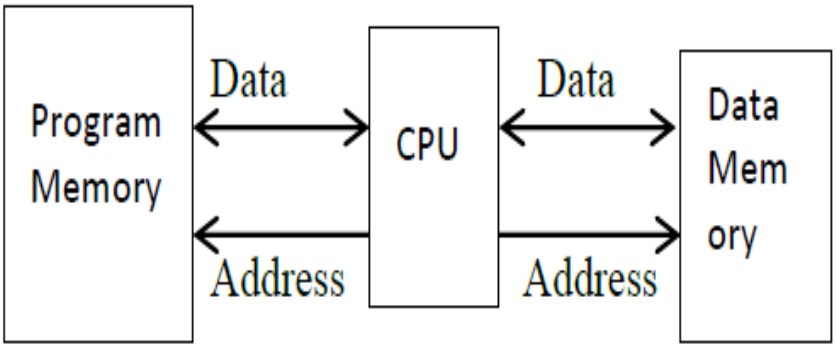
Ans:

Von-Neumann architecture



2M

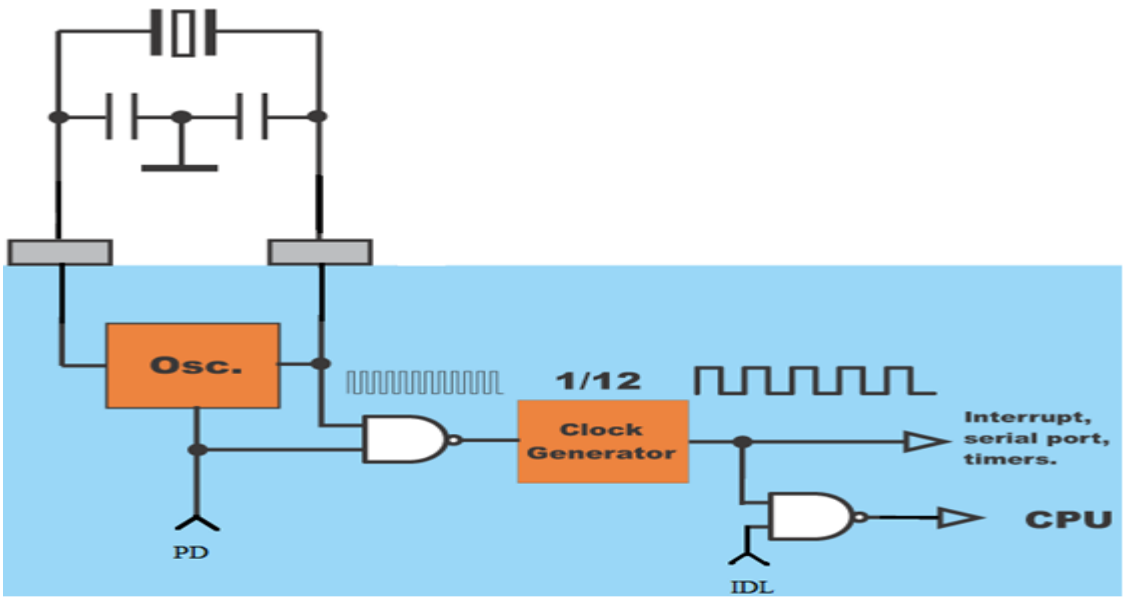
Harvard Architecture



2M

iii **Describe power saving options of 8051 microcontroller.** **4M**

Ans:



2M

For diagram and PCON

(Consider the answer for full marks even if PCON is not written)

Format of PCON:

PCON: POWER CONTROL REGISTER. NOT BIT ADDRESSABLE.

SMOD	—	—	—	GF1	GF0	PD	IDL

SMOD Double baud rate bit. If Timer 1 is used to generate baud rate and SMOD = 1, the baud rate is double when the Serial Port is used in modes 1, 2, or 3.
— Not implemented, reserved for future use.*
— Not implemented, reserved for future use.*
— Not implemented, reserved for future use.*
GF1 General purpose flag bit.
GF0 General purpose flag bit.
PD Power Down bit. Setting this bit activates Power Down operation in the 80C51BH.
IDL Idle Mode bit. Setting this bit activates Idle Mode operation in the 80C51BH.

1M

IDLE MODE

In the Idle mode, the internal clock signal is gated off to the CPU, but not to the Interrupt, Timer, and Serial Port functions.

The CPU status is preserved in its entirety, the Stack Pointer, Program Counter, Program Status Word, Accumulator, and all other registers maintain their data during Idle. The port pins hold the logical state they had at the time idle mode was activated. ALE and PSEN hold at logic high levels.

There are two ways to terminate the idle mode.

i) Activation of any enabled interrupt will cause PCON.O to be cleared and idle mode is terminated.

ii) Hard ware reset: that is signal at RST pin clears IDEAL bit IN PCON register directly. At this time, CPU resumes the program execution from where it left off.

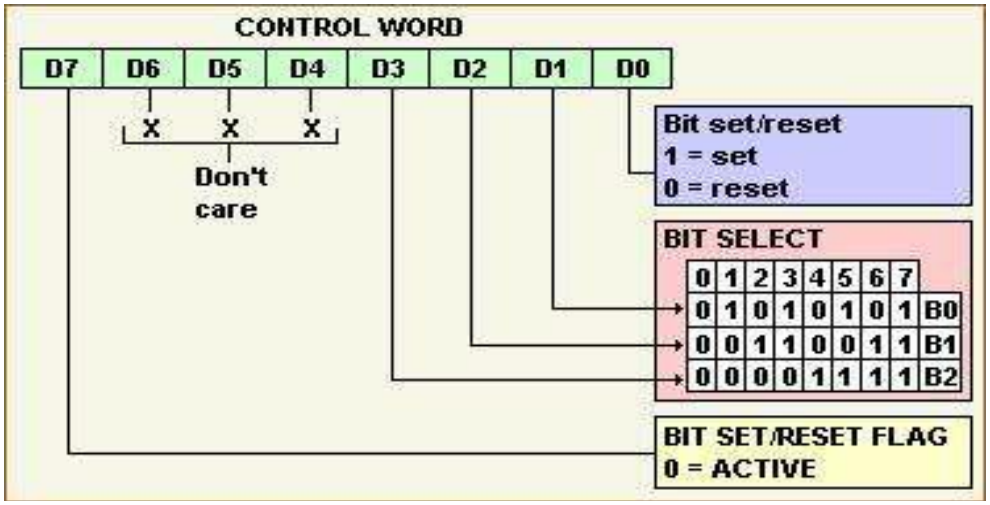
1M

POWER DOWN MODE

An instruction that sets PCON.1 causes that to be the last instruction executed before going into the Power Down mode. In the Power Down mode, the on-chip oscillator is

		stopped. With the clock frozen, all functions are stopped, but the on-chip RAM and Special Function Register are maintained held. The port pins output the values held by their respective SFRS. ALE and PSEN are held low. Termination from power down mode: an exit from this mode is hardware reset. Reset defines all SFRs but doesn't change on chip RAM	**1M**
	iv	**Write the operation of the following instructions of 8051.** **1) CJNE A, direct, rel** **2) DAA** **3) DJNZ Rn, rel.** **4) SWAP A**	**4M**
	Ans:	**1) CJNE A, direct, rel** Compare the contents of the accumulator with the 8 bit data in memory address mentioned in the instruction and if they are not equal then jump to the relative address mentioned in the instruction. Example: CJNE A, 04H, UP Compare the contents of the accumulator with the contents of 04H memory and if they are not equal then jump to the line of instruction where UP label is mentioned. **2) DA A (Decimal Adjust After Addition).** When two BCD numbers are added, the answer is a non-BCD number. To get the result in BCD, we use DA A instruction after the addition. DA A works as follows. • If lower nibble is greater than 9 or auxiliary carry is 1, 6 is added to lower nibble. • If upper nibble is greater than 9 or carry is 1, 6 is added to upper nibble. **Eg 1**: MOV A,#23H MOV R1,#55H	**1M for each instruction with example**

		ADD A, R1 // [A]=78 H DA A // [A]=78 no changes in the accumulator after da a **Eg 2**: MOV A,#53H MOV R1,#58H ADD A,R1 // [A]=ABh DA A // [A]=11, C=1 . ANSWER IS 111. Accumulator data is changed after DA A **3) DJNZ Rn, rel.(Decrement and jump if not zero)** In this instruction a byte present in register Rn is decremented, and if the result is not zero it will jump to the relative address mentioned in the instruction. Example: **DJNZ R3, HERE** Decrements the contents of the register R3, and if it is not equal to zero then jump to the line of instruction where HERE label is mentioned. **4) SWAP A** **Description**: This instruction exchanges bits 0-3 of the Accumulator with bits 4-7 of the Accumulator. This instruction is identical to executing "RR A" or "RL A" four times. **No of bytes**: 1 byte **Addressing mode**: register specific **Example:** MOV A, #59H ; A= 59H SWAP A ; A= 95H	
	v	**With control word register, explain Bit Set reset (BSR) mode of 8255.**	**4M**
	Ans:		



2M for format

Explanation :

The Bit Set/Reset (BSR) mode is available on port C only. Each line of port C (PC_7 - PC_0) can be set or reset by writing a suitable value to the control word register. BSR mode and I/O mode are independent and selection of BSR mode does not affect the operation of other ports in I/O mode.

2M - explanation

- D_7 bit is always 0 for BSR mode.
- Bits D_6, D_5 and D_4 are don't care bits.
- Bits D_3, D_2 and D_1 are used to select the pin of Port C.

Selection of port C pin is determined as follows:

D3	D2	D1	Bit/pin of port C selected
0	0	0	PC_0
0	0	1	PC_1
0	1	0	PC_2
0	1	1	PC_3
1	0	0	PC_4
1	0	1	PC_5

1	1	0	PC_6
1	1	1	PC_7

- Bit D_0 is used to set/reset the selected pin of Port C.

As an example, if it is needed that PC_5 be set, then in the control word,

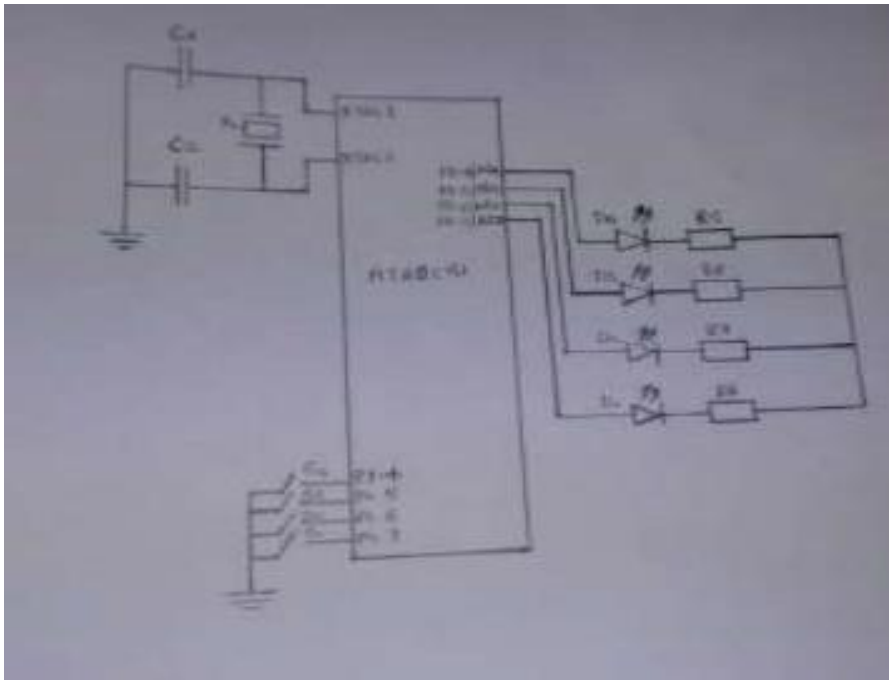
1. Since it is BSR mode, D_7 = '0'.
2. Since D_4, D_5, D_6 are not used, assume them to be '**0**'.
3. PC_5 has to be selected, hence, **D_3 = '1', D_2 = '0', D_1 = '1'**.
4. PC_5 has to be set, hence, **D0 = '1'**.

Thus, as per the above values, 0B (Hex) will be loaded into the Control Word Register (CWR).

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	0	0	1	0	1	1

b	**Attempt any ONE:**	**6- Total Marks**
i	**Write an ALP for 8051 microcontroller to find the average of ten 8 bit numbers stored in internal RAM location starting from 30 H to onwards store result at 60 H. (Assume suitable data)**	**6M**
Ans:		**6M for correct program**

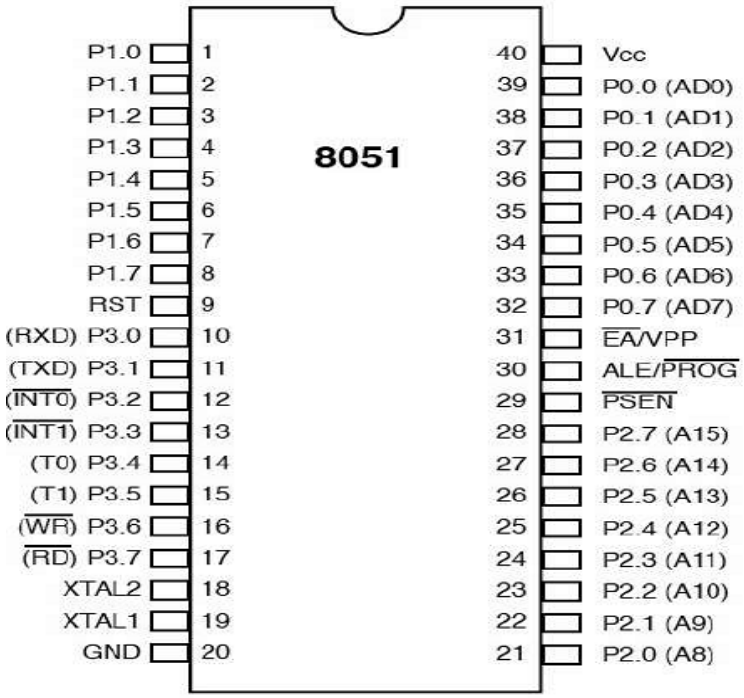
```
        ORG 0000H
        MOV R1, #0A          ;R1 stores the count of total 8 bit numbers
        MOV B, #0A           ;B is used as divisor for average
        MOV R0, #30H         ;R0 acts as pointer to the data
        MOV A, #00H          ;Clear A
BACK :  ADD A, @R0           ; Add the data to A register
```

		INC R0 ; Increment the pointer DJNZ R1,BACK ; repeat addition until R1=0 DIV AB ; divide sum to get average ; quotient is in A and remainder in B MOV 60H,A ;Ignore remainder and store average result in 60h HERE: SJMP HERE ; wait END **NOTE: Program may change. Please check the logic and understanding of students.**	
	ii	**Sketch 8051 microcontroller interfacing diagram to interface 4 LEDs and 4 switches. Interface LEDs to port 0 upper nibble and switch to port 1. Write an ALP for 8051 to read status of switches and operate LEDs as per switch status.**	**6M**
	Ans:	 **Program:** ORG 0000H MOV P1, #0F0H ; Make P1 as input START: MOV A, P1 ; Read status of the key	**3M**

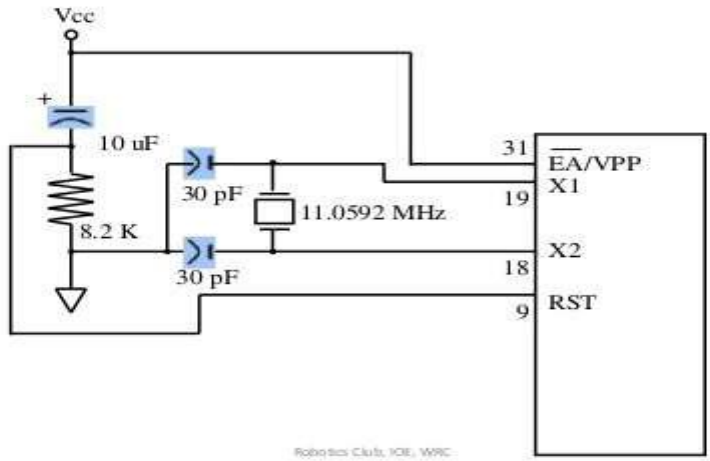
```
        CJNE A, #0F0H, CHECK1     ; Key pressed branch from Port 0

        SJMP START                ; Branch to start
CHECK1: ACALL DELAY               ; Call delay
        MOV A, P1                 ; Read data from port 0
        CPL A                     ; Complement A
        MOV P0, A                 ; Send data to LED
        AJMP START                ; Jump to start
DELAY:  MOV R6,#20H               ; Delay program
NEXT1:  MOV R7, #0FFH
NEXT2:  DJNZ R7, NEXT2
        DJNZ R6, NEXT1
        RET
        END
```

3M

NOTE: Program may change. Please check the logic and understanding of students.

Q. No.	Sub Q. N.	Answers	Marking Scheme
2		**Attempt any FOUR:**	**16- Total Marks**
	a	**Draw pin diagram of 8051 microcontroller.**	**4M**
	Ans:	**4 M for correct diagram**  	**4M**
	b	**Which pins of 8051 microcontroller are used for external memory interfacing with 8051? State their functions.**	**4M**
	Ans:	The following pins of 8051 are used for external memory interfacing: i) PSEN/:PSEN stands for —program store enable.‖ In an 8031-based system in which an external ROM holds the program code, this pin is connected to the OE pin of the ROM. In other words, to access external ROM containing program code, the 8031/51 uses the PSEN signal. When the EA pin is connected to GND, the 8031/51 fetches opcode from	**1M each**

		external ROM by using PSEN. In systems based on the 8751/89C51/ DS5000 where EA is connected to VCC, these chips do not activate the PSEN pin. This indicates that the on- chip ROM contains program code. ii) ALE: ALE stands for address latch enable. It is an output pin and is active high for latching the low byte of address during accesses to external memory. The ALE pin is used for demultiplexing the address and data by connecting to the G pin of the 74LS373 chip. iii) $\overline{EA}$: EA stands for External access pin and it is active low. When it is held high, executes instruction from the internal program memory till address 0FFFH, beyond this address the instructions are fetched from external program memory. If this pin is low , all the instructions are fetched from the external memory. During normal operation , this pin should not be floated. (Should be connected to ground). iv) RD(P3.7) and WR(P3.6): External data memory read and external data memory write.	
	c	**Draw and explain reset circuit used for 8051 microcontroller.**	**4M**
	Ans:	**Function of RESET:** 1. Pin 9 is the RESET pin. It is an input and is active high (normally low). Upon applying a high pulse to this pin, the microcontroller will reset and terminate all activities. 2. This is often referred to as a power-on reset. Activating a power-on reset will cause all values in the registers to be lost. It will set program counter to all 0s. 3. In order for the RESET input to be effective, it must have a minimum duration of two machine cycles. In other words, the high pulse must be high for a minimum of two machine cycles before it is allowed to go low.	**2M – explanation**



2M – Diagram

d **Draw the format of PSW register of 8051µC and state the function of each flag.** **4M**

Ans:

CY	AC	F0	RS1	RS0	OV	--	P

CY	PSW.7	Carry Flag.
AC	PSW.6	Auxiliary carry flag.
F0	PSW.5	Available to the user for general purpose.
RS1	PSW.4	Register bank selector bit 1.
RS0	PSW.3	Register bank selector bit 0.
OV	PSW.2	Overflow flag.
--	PSW.1	User- definable bit.
P	PSW.0	Parity flag. Set/cleared by hardware each instruction cycle to indicate and Odd/ even number of 1 bit in the accumulator.

2M – format

1. CY: Carry flag.

1. This flag is set whenever there is a carry out from the D7 bit.

2. The flag bit is affected after an 8 bit addition or subtraction.

3. It can also be set to 1 or 0 directly by an instruction such as SETB C and CLR C where SETB C stands for set bit carry and CLR C for clear carry.

2. AC: Auxiliary carry flag

If there is a carry from D3 and D4 during an ADD or SUB operation, this bit is set; it is

		cleared. This flag is used by instructions that perform BCD (binary coded decimal) arithmetic. **3. F0:.** Available to the user for general purposes **4. RS0, RS1: Register bank selects bits** 1. These two bits are used to select one of the four register banks n internal RAM in the table. By writing zeroes and ones to these bits, a group of registers R0- R7 can be used out of four registers banks in internal RAM. RS1 | RS0 | Space in RAM 0 | 0 | Bank 0 (00H- 07H) 0 | 1 | Bank 1 (08H-0FH) 1 | 0 | Bank2 (10H-17H) 1 | 1 | Bank3 (18H-1FH) **5. OV: Overflow flag** This flag is set whenever the result of a signed number operation is too large, causing the high-order bit to overflow into the sign bit. In general, the carry flag is used to detect errors in unsigned arithmetic operations. The overflow flag is only used to detect errors in signed arithmetic operations. **6. P: Parity flag** 1. The parity flag reflects the number of 1s in the A (accumulator) register only. If the A register contains an odd number of 1's, then P=1, P=0 if A has an even number of 1's.	**2M - functions**
	e	**Explain the function of following registers of 8051µC.** **i) Stack pointer** **ii) DPTR** **iii) Program counter** **iv) Accumulator**	**4M**
	Ans:	**i. Stack pointer:** The 8-bit stack pointer (SP) register is used to hold an internal RAM address that is called the top of stack. The address held in the SP register is the location in internal RAM where the last byte of data was stored by a stack operation.	**1M each**

		ii. DPTR : The DPTR register is made up of two registers named as DPH & DPL which are used to access any memory address that may be internal & external code access & external data access. The DPTR is under the program control & can also be specified as 16-bit pointer as DPTR or by individual 8- bits as DPH & DPL. DPTR does have a single address but the DPH is assigned the address as 82 H. **iii. Program counter:** The program counter (PC) is a 16-bit register. It is used to hold address of a byte in memory. Program instruction bytes are fetched from locations in memory that are addressed by PC. Program ROM may be on chip at addresses 0000H to 0FFF H, external to the chip for addresses that exceeds 0FFF h or totally external for all addresses from 0000H to FFFF H. The PC is incremented automatically after every instruction byte is fetched. The PC is the only register that does not have any address. **iv. Accumulator:** It is also called as register A. It is an 8- bit register. The CPU of 8051 is accumulator based hence it is used to hold the source operand and result of arithmetic operations like addition , subtraction, multiplication, division .However it is source as well as destination for logical operations and data movement instructions. It can be used as a look up table pointer. It is also used in RAM expansion. It is specially used for rotate, parity computation, testing for zero etc. It is bit accessible.	
f	**What is bus? Describe the function of address, data and control bus.**	**4M**
Ans:	A Bus is a set of physical connection used for communication between CPU and peripherals. There are three types of buses Address Bus, Data Bus and Control Bus **(1) Address Bus** • The address bus is unidirectional over which the microcontroller sends an address code to the memory or input/output. The size of the address bus is specified by the number of bits it can handle. • The more bits there are in the address bus, the more memory locations a microcontroller can access. A 16-bit address bus is capable of addressing (64k) addresses.	**1M**

		(2) Data Bus • The data bus is bidirectional on which data or instruction codes are transferred into the microcontroller or on which the result of on operation or computation is sent out from the microcontroller to the memory or input/output. • Depending on the particular microcontroller, the data bus can handle 8-bit or 16-bit data. **(3) Control Bus**: • The control bus is used by the microcontroller to send out or receive timing and control signals like read and write in order to co- ordinate and regulate its operation and to communicate with other devices i.e. memory or input/output.	**1M for each bus**

Q. No.	Sub Q. N.	Answers	Marking Scheme
3		**Attempt any FOUR:**	**16- Total Marks**
	a	**Explain the following directives with example.** **i) ORG** **ii) DB** **iii) EQU** **(iv) END**	**4M**
	Ans:	**ORG:-**ORG stands for Origin Syntax: ORG Address 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. Some assemblers use —.ORG (notice the dot) instead of —ORG for the origin directive. **DB:-** (Define Byte) Syntax: Label: DB Byte Where byte is an 8-bit number represented in either binary, Hex, decimal or ASCII form. There should be at least one space between label & DB. The colon (:) must present after label. This directive can be used at the beginning of program. The label will be used in program instead of actual byte. There should be at least one space between DB & a byte. **EQU**: Equate It is used to define constant without occupying a memory location.	**1M each**

		Syntax: Label EQU Numeric value By means of this directive, a numeric value is replaced by a symbol. For e.g. MAXIMUM EQU 99 After this directive every appearance of the label —MAXIMUM in the program, the assembler will interpret as number 99 (MAXIMUM=99). **END:** This directive must be at the end of every program meaning that in the source code anything after the END directive is ignored by the assembler. This indicates to the assembler the end of the source file. Once it encounters this directive, the assembler will stop interpreting program into machine code. e.g. END ; End of the program.	
	b	**State any four addressing mode of 8051 µC and explain each with example.**	**4M**
	Ans:	There are a number of addressing modes available to the 8051 instruction set, as follows: 1. Immediate Addressing mode 2. Register Addressing mode 3. Direct Addressing mode 4 Register Indirect addressing mode 5. Relative Addressing mode 6. Absolute addressing mode 7. Long Addressing mode 8. Indexed Addressing mode 1) **Immediate Addressing mode:** Immediate addressing simply means that the operand (which immediately follows the Instruction op. code) is the data value to be used. For example the instruction: MOV A, #25H ; Load 25H into A Moves the value 25H into the accumulator. The # symbol tells the assembler that the	**1M each** **(½ M - explanation,** **½ M - example)**

immediate addressing mode is to be used.

2) Register Addressing Mode:

One of the eight general-registers, R0 to R7, can be specified as the instruction Operand. The assembly language documentation refers to a register generically as Rn.

An example instruction using register addressing is :

ADD A, R5 ; Add the contents of register R5 to contents of A (accumulator) , store sum in A.

Here the contents of R5 are added to the accumulator and sum stored in A. One advantage of register addressing is that the instructions tend to be short, single byte instructions.

3) Direct Addressing Mode:

Direct addressing means that the data value is obtained directly from the memory location specified in the operand.

For example consider the instruction:

MOV R0, 40H; Save contents of RAM location 40H in R0.

The instruction reads the data from Internal RAM address 40H and stores this in theR0. Direct addressing can be used to access Internal RAM, including the SFR registers.

4) Register Indirect Addressing Mode:

Indirect addressing provides a powerful addressing capability, which needs to be appreciated.

An example instruction, which uses indirect addressing, is as follows:

MOV A, @R0; move contents of RAM location whose address is held by R0 into A

The @ symbol indicates that indirect addressing mode is used. If the data is inside the CPU, only registers R0 & R1 are used for this purpose.

5) Relative Addressing Mode:

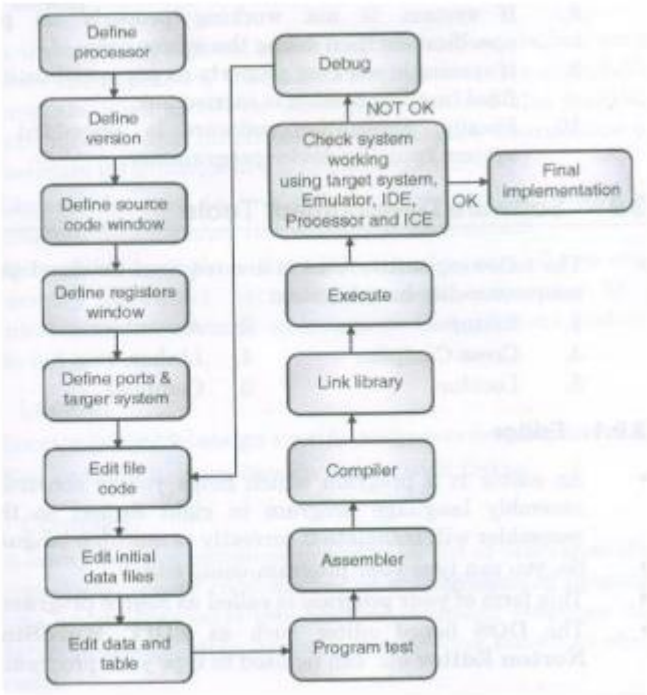
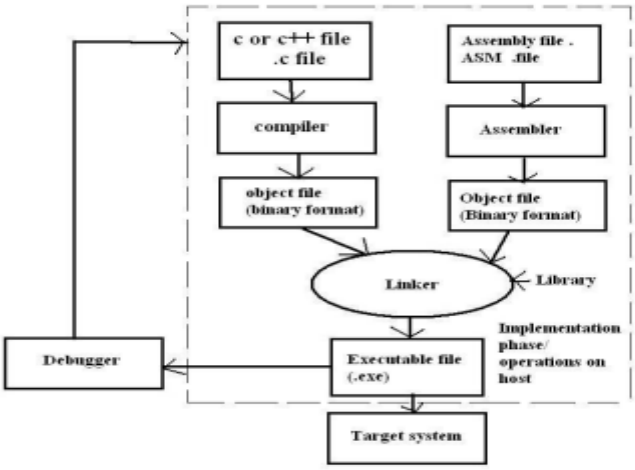
This is a special addressing mode used with certain jump instructions. The relative address, often referred to as an offset, is an 8-bit signed number, which is automatically added to the PC to make the address of the next instruction. The 8-bit signed offset value gives an address range of + 127 to –128 locations.

Consider the following example:

SJMP LABEL_X

An advantage of relative addressing is that the program code is easy to relocate in memory in

		that the addressing is relative to the position in memory. **6) Absolute addressing Mode:** Absolute addressing within the 8051 is used only by the AJMP (Absolute Jump) and ACALL (Absolute Call) instructions. **7) Long Addressing Mode:** The long addressing mode within the 8051 is used with the instructions LJMP and LCALL. The address specifies a full 16 bit destination address so that a jump or a call can be made to a location within a 64KByte code memory space (216 = 64K). An example instruction is: LJMP 5000h; full 16 bit address is specified in operand. **8) Indexed Addressing Mode:** With indexed addressing a separate register, either the program counter, PC, or the data pointer DTPR, is used as a base address and the accumulator is used as an offset address. The effective address is formed by adding the value from the base address to the value from the offset address. Indexed addressing in the 8051 is used with the JMP or MOVC instructions. Look up tables are easy to implement with the help of index addressing. Consider the example instruction: MOVC A, @A+DPTR MOVC is a move instruction, which moves data from the external code memory space. The address 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 us referred to as the index address.	
	c	**Draw the software development cycle. State the function of editor, assembler and cross compiler.**	**4M**

	Ans:	 **Fig: software development cycle.** **(OR)**  **Fig: software development cycle**	**1M diagram,** **1M each for functions**

		Assembler: An assembler is programs that translate assembly language program to the correct binary code for each instruction i. e .machine code and generate the file called as Object file with extension . obj and list file with extension .lst extension. Some examples of assembler are ASEM51, Keil A51, AX51, Intel ASM-51etc. **Editor:** An editor is a program which helps you to construct your assembly language program in right format so that the assembler will translate it correctly to machine language. So, you can type your program using editor. This form of our program is called as source program and extension of program must be .asm or .src depending on which assembler is used. The DOS based editor such as EDIT, WordStar, and Norton Editor etc. can be used to type your program. **Cross Compiler:** A cross compiler is used to create executable code for a platform other than the one on which the compiler is run. Cross compiling is compiling something for different CPU type than the one you are running on. A cross compiler is used to produce executable (or objects) for a platform other than the local host.	
	d	**Write an ALP for 16 bit multiplication. Assume numbers to be stored in External RAM.**	**4M**

	Ans:	ORG 0000H
		LCALL TRANSFER_IN
		LCALL MUL_16
		LCALL TRANSFER_OUT
		
		TRANSFER_IN:
		MOV DPTR, #ADD_FIRSTBYTE
		MOV R0, #04H
		MOV R1, #04H
		HERE: MOVX A, @DPTR
			4M

```
 MOV @R0, A
INC R0
INC DPTR
 DJNZ R1,HERE
RET


MUL_16:
;Multiply R5 by R7
MOV A,R5       ;Move the R5 into the Accumulator
MOV 0F0H,R7 ;Move R7 into F0H
MUL AB         ;Multiply the two values
MOV R2,0F0H ;Move B (the high-byte) into R2
MOV R3,A       ;Move A (the low-byte) into R3

;Multiply R5 by R6

MOV A,R5       ;Move R5 back into the Accumulator
MOV 0F0H,R6  ;Move R6 into B
MUL AB                 ;Multiply the two values
ADD A,R2               ;Add the lower byte into value in R2
MOV R2,A               ;Move the resulting value back into R2
MOV A,0F0H             ;Move the high-byte into the accumulator
 ADDC A,#00h   ;Add zero (plus the carry, if any)
MOV R1,A                ;Move the resulting answer into R1
 MOV A,#00h   ;Load the accumulator with zero
ADDC A,#00h   ;Add zero (plus the carry, if any)

MOV R0, A              ;Move the resulting answer to R0.
```

```
;Multiply R4 by R7
MOV A,R4          ;Move R4 into the Accumulator
MOV 0F0H,R7       ;Move R7 into B
MUL AB            ;Multiply the two values
ADD A,R2          ;Add the low-byte into value in R2
MOV R2,A          ;Move the resulting value back into R2
MOV A,0F0H        ;Move the high-byte into the accumulator
ADDC A,R1         ;Add the current value of R1 (plus carry)
MOV R1,A          ;Move the resulting answer into R1.
MOV A,#00h        ;Load the accumulator with zero
ADDC A,R0         ;Add the current value of R0 (plus carry)
MOV R0,A          ;Move the resulting answer to R1.

;Multiply R4 by R6

MOV A,R4          ;Move R4 back into the Accumulator
MOV 0F0H,R6       ;Move R6 into
MUL AB            ;Multiply the two values
ADD A,R1          ;Add the low-byte into the value in R1
MOV R1,A          ;Move the resulting value back into R1
MOV A,0F0H        ;Move the high-byte into the accumulator
ADDC A,R0         ;Add it to the value already in R0 + carry
MOV R0,A          ;Move the resulting answer back to R0
RET               ;Return(answer is now in R0, R1, R2,R3)

;Answer stored in r0(msb),r1,r2,r3(lsb)

TRANSFER_OUT:
MOV DPTR, #ADD_ANSBYTE
```

```
SETB PSW.3
MOV R0, #00H
MOV R1, #04H
THERE: MOV A, @R0
MOVX @DPTR, A
INC R0
INC DPTR
 DJNZ R1,THERE
BACK: SJMP BACK




;
```

(Marks can be given for any other relevant logic)

e **List interrupt of 8051 µC with their vector addresses and priorities.** **4M**

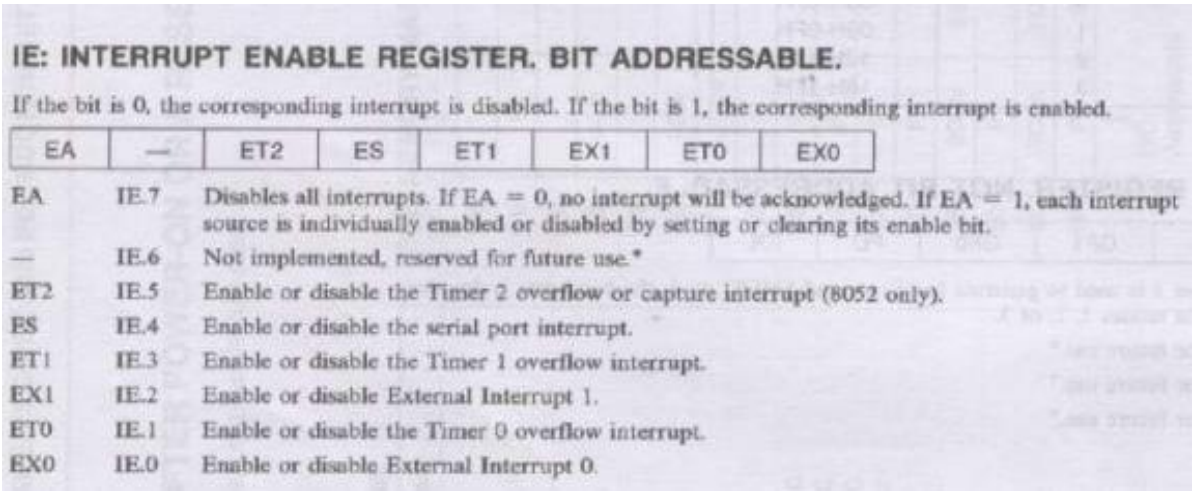
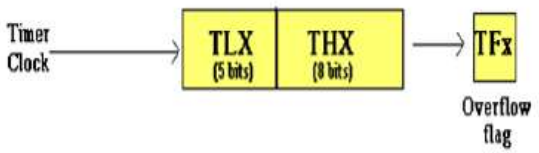
Ans:

Interrupt Source	Vector address	Interrupt priority
External Interrupt 0 – INT0	0003H	1
Timer 0 Interrupt	000BH	2
External Interrupt 1 – INT1	0013H	3
Timer 1 Interrupt	001BH	4
Serial Interrupt	0023H	5

2M- list

1M vector address , 1M priority

Q. No.	Sub Q. N.	Answers	Marking Scheme
4	**a**	**Attempt any THREE:**	**12- Total Marks**
	i	**Classify the instruction set of microcontroller 8051. List one example of each.**	**4M**
	Ans:	Instruction set can be classified as: 1. Data transfer instructions 2. Arithmetic instructions 3. Logical Instructions 4. Control transfer instructions 5. Bit manipulation instructions **Data Transfer Instructions:** MOV A, Rn: The contents of registers Rn (R0-R7) is moved to Accumulator. **(for any other data transfer instructions marks can be given)** **Arithmetic Instructions:** ADD A, Byte: Add the contents of Accumulator with byte and the result is stored in Accumulator. **(for any other arithmetic instructions marks can be given)** **Logical Instructions:** RR A: Rotate the contents of Accumulator to right. **(for any other logical instructions marks can be given)** **Control Transfer Instructions:** JNC ADDR: If carry flag CY=0, jump to the given relative address. **(for any other control transfer instructions marks can be given)** **Bit Manipulation Instructions:**	**2M- classification, 2M example**

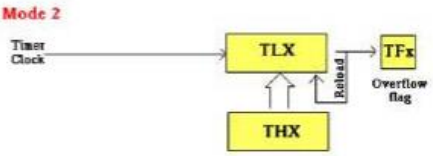
	CLR BIT: Clear the given bit. **(for any other bit manipulation instructions marks can be given)**	
ii	**Draw and describe IE register of 8051 µC.**	**4M**
Ans:	 **IE: INTERRUPT ENABLE REGISTER. BIT ADDRESSABLE.** If the bit is 0, the corresponding interrupt is disabled. If the bit is 1, the corresponding interrupt is enabled. EA — ET2 ES ET1 EX1 ET0 EX0 EA IE.7 Disables all interrupts. If EA = 0, no interrupt will be acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit. — IE.6 Not implemented, reserved for future use.* ET2 IE.5 Enable or disable the Timer 2 overflow or capture interrupt (8052 only). ES IE.4 Enable or disable the serial port interrupt. ET1 IE.3 Enable or disable the Timer 1 overflow interrupt. EX1 IE.2 Enable or disable External Interrupt 1. ET0 IE.1 Enable or disable the Timer 0 overflow interrupt. EX0 IE.0 Enable or disable External Interrupt 0.	**2M - format** **2M - description**
iii	**With the help of neat diagram, describe the timer modes of 8051 µC.**	**4M**
Ans:	M1 M0 MODE DESCRIPTION 0 0 0 13-bit timer 0 1 1 16-bit timer 1 0 2 8-bit auto-reload 1 1 3 Split mode Operating modes of Timer: The timer may operate in any of the four modes that are determined byM1 and M0 bit in TMOD register. **Mode 0:** **Mode 0**  	**1M each**

Mode 0 is a 13 bit timer, TLX is 5 bits and THX is 8 bits. The 13-bit counter can hold values between 0000H to 1FFFH in TH and TL. When the timer reaches 1FFFH, rolls over to 0000H and TF is raised.

Mode 1:

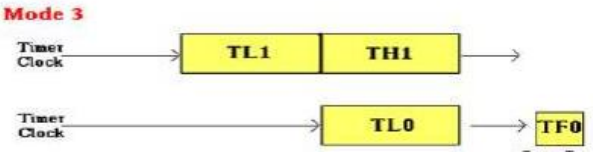
It is similar to Mode 0 except TLX is configured as a full 8-bit counter. Hence pulse input is divided by 256_{10} so that TH counts the timer flag is set when THX rolls over from FF to 00H.

Mode 2



In this mode only TLX is used as 8-bit counter. THX is used to hold the value which is loaded in TLX initially. Every time TLX overflows from FFH to 00H the timer flag is set and the value from THX is automatically reloaded in TLX register.

Mode 3



In this mode, timer 0 becomes two completed separate 8-bit timers. TL0 is controlled by gate arrangement of timer 0 and sets timer 0 flag when it overflows. TH0 receives the timer clock under the control of TR1 bit and sets TF1 flag when it overflows. Timer 1 may be used in mode 0, 1 and 2 with one important exception that no interrupt will be generated by the timer when the timer 0 is using TF1 overflow flag.

Iv	**Write the assembly language program for 8051 to generate square wave of 10KHz on port pin P1.7. Assume XTAL, frequency = 12MHz.**	**4M**
Ans:	**CALCULATION**: (Assumed Timer 0 in Mode 1) Crystal freq=12MHz Timer frequency=12MHz/12=1MHz	**1M calculation, 3M program**

Time=1/1MHz=1µs

Given frequency =10KHz,

Time Period (T)= 1/10kHz= 0.1ms,

Ton=Toff=T/2,

Delay= 0.1ms/2= 0.05ms

0.05ms/1µs=50

Therefore count to be loaded in TH0 and TL0 can be calculated as 65536- 50=65486d=FFCEH

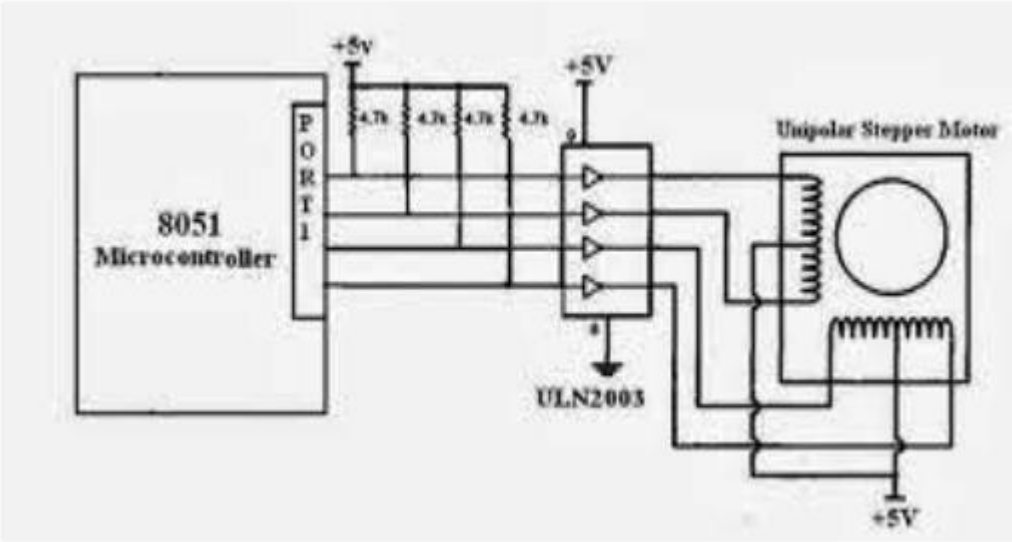
Program:

```
MOV TMOD,#01H

UP: SETB P1.7

ACALL DELAY

CLR P1.7

ACALL DELAY

SJMP UP

DELAY: MOV TH0,#0FFH

MOV TL0,# 0CEH

SETB TR0

JNB TF0,$

CLR TF0

CLR TR0

 RET

END
```

	B	Attempt any ONE:	**6- Total Marks**
	I	**Write an ALP for 8051 µC to find smallest numbers from the array of 10 numbers stored in External memory location 3000 H to onwards. Store result at 6000 H. (assume suitable data)**	**6M**

	Ans:	**Program:**
		CLR PSW.3 ; Select Bank 0 PSW.3
		MOV R1, 0AH ; Initialize byte counter
		MOV DPTR, # 3000H ; Initialize memory pointer
		DEC R1 ; Decrement byte counter by 1
		MOVX A, @DPTR ; Load number in accumulator
		MOV 40 H, A ; Store number in memory location
		UP: INC DPTR ; Increment memory pointer by 1
		MOVXA, @DTPR ; Read next number
		CJNE A, 40 H, DN ; if number≠ next number, and then go to DN
		DN: JNC NEXT ; If next number > number then go to NEXT
		MOV 40H, A ; Else replace n ext number with number
		NEXT: DJNZ R1, UP ; Decrement byte counter by 1, if byte counter≠ 0 then go to UP
		MOV DPTR,#6000H ; Increment memory pointer by 1
		MOV A, 40H
		MOVX@ DPTR, A ; Store result t external memory location
		LOOP: AJMP LOOP ; Stop
			6M

	Ii	**Draw the interfacing diagram of stepper motor with 8051 microcontroller. Write an assembly language program to rotate the stepper motor continuously in anti-clockwise direction. Assume step angle is 0.9^0 /step.**	**6M**
	Ans:		**6M**



PROGRAM:

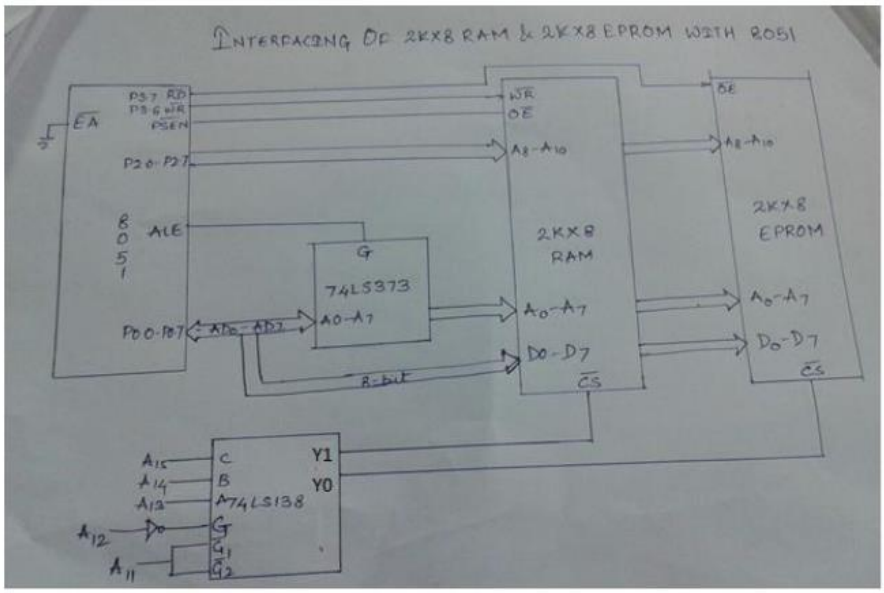
Bit pattern for code for Half Stepping (0.9^0) Stepper Motor for Anticlockwise rotation

A	B	C	D	CODE
0	0	1	1	03H
0	0	1	0	02H
0	1	1	0	06H
0	1	0	0	04H
1	1	0	0	0CH
1	0	0	0	08H
1	0	0	1	09H
0	0	0	1	01H
0	0	1	1	03H

```
        ORG 0000H
REP:    MOV DPTR, #COD
```

```
        MOV R2, #08H
UP:     MOV A, #00H
        MOVC A, @A+DPTR
        MOV P1, A
        ACALL DELAY
        INC DPTR
        DJNZ R2, UP
        SJMP REP
DELAY:  MOV R3, #0FFH
AGAIN:  MOV R4,#0FFH
        DJNZ R4,$
        DJNZ R3, AGAIN
COD:    DB 03H,02H,06H,04H,0CH,08H, 09H,01H
HERE :  SJMP HERE
```

	Iii	**Draw interfacing diagram of 2K byte EPROM and 2K byte RAM to 8051 µC. Draw memory map.**	**6M**
	Ans:		**3M interfacing, 3M memory mapping**



Memory map

	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	ADDR
Start addr of EPROM	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0000H**
End addr of EPROM	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**07FFH**
Start addr of ROM	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**2000H**
End addr of ROM	**0**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**27FFH**

Q. No.	Sub Q. N.	Answers	Marking Scheme
5		**Attempt any FOUR:**	**16- Total Marks**
	a	**List the I/o ports of 8051 microcontroller and describe alternate function of port 0.**	**4M**
	Ans:	There are four ports available with 8051 microcontroller as, 1. Port 0 2. Port 1 3. Port 2 4. Port 3 1. **Port 0:** It can be used as a) Simple input/output b) Bidirectional low order address / data bus (AD0 - AD7) for external memory.	**Listing: 2M, Alternate function: 2M**
	b	**State operating modes of serial port of IC 8051 microcontroller.**	**4M**
	Ans:	**Mode 0 – 8 bit shift Register** In this mode, the serial port works like a shift register and the data transmission works synchronously with a clock frequency of fosc /12. Serial data is received and transmitted through RXD. 8 bits are transmitted/ received at a time. Pin TXD outputs the shift clock pulses of frequency fosc /12, which is connected to the external circuitry for synchronization In Mode 0, the baud rate is fixed at Fosc. / 12. **2 Mode 1 : 8-bit UART** 10 bits are transmitted through TXD or received through RXD. The 10 bits consist of one start bit (which is usually '0'), 8 data bits (LSB is sent first/received first), and a stop bit (which is usually '1'). Once received, the stop bit goes into RB8 in the special function register SCON. The baud rate is variable. $\text{Baud Rate} = \frac{2^{SMOD}}{32} \times \frac{\text{Oscillator Frequency}}{12 \times [256 - (TH1)]}$ **Mode 2 : 9-bit UART**. In this mode 11 bits are transmitted through TXD or received through RXD.The various bits are as follows: a start bit (usually '0'), 8 data bits (LSB first), a programmable 9 th (TB8 or RB8)bit and a stop bit (usually '1'). While transmitting, the 9 th data bit (TB8 in SCON) can be assigned the value '0' or '1'. For example, if the information of parity is to be transmitted, the parity bit (P) in PSW could be moved into TB8.On reception of the data, the 9 th bit goes into RB8 in 'SCON', while the stop bit is ignored.	**1M for each mode**

Baud Rate = $\frac{2^{SMOD}}{64}$ X (Oscillator Frequency)

4. **Mode 3: 9-Bit UART with Variable Baud Rate**

In this mode 11 bits are transmitted through TXD or received through RXD. The various bits are: a start bit (usually '0'), 8 data bits (LSB first), a programmable 9 th bit and a stop bit (usually '1'). Mode-3 is same as mode-2, except the fact that the baud rate in mode-3 is variable (i.e., just as in mode-1).

Baud Rate = $\frac{2^{SMOD}}{32}$ X $\frac{\text{Oscillator Frequency}}{12 \text{ X } [256 - (\text{TH1})]}$

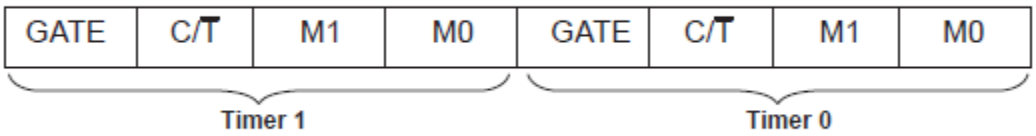
c **Explain SBUF register used with serial-communication with 8051 microcontroller.** **4M**

Ans: Serial Buffer register, SBUF is physically two separate registers at the same address. (99H). One is Write-only transmit register and other is Read-only receive register. The byte to be transmitted on the serial port is "written" into SBUF & Serial transmission starts immediately. The byte received from the serial port will be stored in SBUF once the last bit is received. **4M**

d **Draw the format of TMOD and state the function of each bit.** **4M**

Ans:

TMOD: TIMER/COUNTER MODE CONTROL REGISTER.



Format: 2M

- **GATE** : When TR_X (in TCON) is set and GATE=1 , TIMER/COUNTER will run only while INT_X pin is high(hardware control), when GATE=0 , TIMER/COUNTER will run only while TR_X=1 regardless of state of INTX pin (software control)
- **C/$\overline{T}$** **:** Timer or Counter Selector

  1 = counter – external timing signal , input from T0/T1 pin
  0 = timer – internal timing signal , input from internal system clock
- **M1 M0** These two bits selects the Timer / Counter operating mode.

Functions: 2M

M1	M0	MODE	DESCRIPTION
0	0	0	13-bit timer/counter
0	1	1	16 bit timer/counter
1	0	2	8 bit auto-reload
1	1	3	Split mode

e **Draw and explain the format of IP register of 8051 microcontroller.** **4M**

Ans: **The IP Register : INTERRUPT PRIRIOTY REGISTER** **2M**

-	-	-	PS	PT1	PX1	PT0	PX0

Bit	Name	Description
IP.7	-	Reserved
IP.6	-	Reserved
IP.5	-	Reserved
IP.4	PS	Serial Port Interrupt priority
IP.3	PT1	Timer 1 Interrupt priority (TF1)
IP.2	PX1	External Interrupt 1 priority (INT1)
IP.1	PT0	Timer 0 Interrupt priority (TF0)
IP.0	PX0	External Interrupt 0 priority (INT0)

1 = high priority level , 0 = low priority level **2M**

Q. No.	Sub Q. N.	Answers	Marking Scheme
6		**Attempt any FOUR:**	**16- Total Marks**
	a	**Write ALP for 8051 microcontroller to transmit message 'WELCOME' serially at baud rate 9600, 8 bit data, I stop bit. Assume crystal frequency is 11.0592 MHz.**	**4M**

	Ans:	MOV SCON,#50H ;8-bit, 1 stop, REN enabled
		(MODE 1)
		MOV TMOD,#20H ;timer 1,mode 2(autoreload)
		MOV TH1,#0FDH ;9600 baud rate
		SETB TR1 ;start timer 1
		MOV A,#"W" ;transfer "W"
		ACALL TRANS
		MOV A,#"E" ;transfer "E"
		ACALL TRANS
		MOV A,#"L" ;transfer "L"
		ACALL TRANS
		MOV A,#"C" ;transfer "C"
		ACALL TRANS
		MOV A,#"O" ;transfer "O"
		ACALL TRANS
		MOV A,#"M" ;transfer "M"
		ACALL TRANS
		MOV A,#"E" ;transfer "E"
		ACALL TRANS
		AGAIN:SJMP AGAIN
		TRANS:MOV SBUF,A ;load SBUF
		HERE :JNB TI,HERE ;wait for the last bit
		CLR TI ;get ready for next byte
		RET
			Correct program: 4 marks

	b	**What is the role of SMOD bit in serial communication? Write instruction to set SMOD bit.**	**4M**
	Ans:	SMOD bit Doubles the serial communication Baud rate , When set to 1 in modes 1, 2, or 3.	**2M**

		When 8051 is powered up, D7 bit i.e. SMOD bit of PCON =0, we should set D7 of PCON to double the baud rate. **PCON Register:** MSB LSB SMOD — — — GF1 GF0 PD IDL Instruction to set SMOD bit is **MOV PCON, #80H.**	**2M**
	c	**Describe the function of following handshaking signals of 8255.** **i) IBF** ii) $\overline{STB}$ iii) $\overline{ACK}$ iv) $\overline{OBF}$	**4M**
	Ans:	**i) IBF:** This is active high output signal generated by 8255 to peripheral. A high on this output indicates that data has been loaded into input latch. ii) $\overline{STB}$ This is active low input signal to 8255.A low on this input loads data into the input latch. This signal is generated by the peripheral to indicate that data is available on input ports lines. iii) $\overline{ACK}$ A low on this input informs the 8255 that the data from port A or port B is accepted by the output peripheral In essence peripheral device generates ACK signal indicating that data is accepted from the output port iv) $\overline{OBF}$ The OBF output will go low to indicate that the CPU has written data to the specified port. This signal is generated by 8255 for output peripheral to indicate data is available and latched on the port lines.	**Each signal function: 1 M**

	d	**Draw format of SFR SCON and explain each bit of same.**	**4M**
	Ans:		**2M**

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0 Serial Port Mode Bit 0,
SM1 Serial Port Mode Bit 1

SM0	SM1	Mode	Description	Baud Rate
0	0	0	shift register	$f_{osc}/12$
0	1	1	8-bit UART	variable
1	0	2	9-bit UART	$f_{osc}/64$ or $f_{osc}/32$
1	1	3	9-bit UART	variable

SM2 : enables the multiprocessor communication feature in Modes 2 and 3. In Modes 2 or 3, if SM2 is set to 1 than RI will not be activated if the received 9th date bit(RB8) is 0. In Mode 1, if SM2 = 1 then RI will not be activated if a valid stop bit was not received. In Mode 0 SM2 should be 0 **2M**

REN: Rreceive enable bit. When the REN =1, reception is enabled & REN=0, the reception is disabled.

TB8: Transfer bit 8 The 9th data bit that will be transmitted in modes 2 and 3.

RB8: Receive bit 8. The 9th data bit that was received in modes 2 and 3.In mode 1, this bit is the stop bit that was received.

TI: Transmit interrupt flag . set by hardware at end of 8th bit in mode 0 and at the beginning of the stop bit in other modes, in serial transmission. Must be cleared by software

RI: Receive interrupt flag . set by hardware at end of 8th bit in mode 0 and half way through the stop bit time in other modes in serial reception. Must be cleared by software

	e	**Describe selection factors of microcontroller.**	**4M**
	Ans:	**1. Word length:** It refers to number of bits a microcontroller can process. The word length of microcontroller is either 8, 16 or 32 bit. As the word length increases, the cost, power dissipation and speed of the microcontroller increases.	**Any four selection factors: 1 mark each**
		2. Power dissipation: It depends upon various factors like clock frequency, speed, supply voltage, VLSI technology etc. For battery operated embedded systems, we must use low power microcontrollers.	
		3. Speed/ Clock frequency: The speed of an embedded system depends upon the clock frequency. The clock frequency depends upon the application.	
		4. Instruction Set: On the basis of instructions microcontrollers are classified as CISC & RISC. CISC system improves software flexibility. Hence it is used in general purpose systems.	

		RISC improves speed of the system for the particular applications. **5. Internal resources:** The internal resources are ROM, RAM, EEPROM, FLASH ROM, UART, TIMER, watch dog timer, PWM, ADC, DAC, network interface, wireless interface etc. It depends upon the application for which microcontroller is going to be used. **6. I/O capabilities:** The capability of microcontroller to handle external devices depend upon the number of I/O ports, size and characteristics of each I/O port provided. There may be serial port or parallel ports. 7. **Architecture type:** Architecture decides complexity and hence cost of the chip. Depending upon application suitable architecture should be selected. Harvard architecture is commonly used.	