KLS Vishwanathrao Deshpande Institute of Technology

(Accredited by NAAC with "A" Grade)

(Approved by AICTE, New Delhi, Affiliated to VTU, Belagavi)

(Recognized Under Section 2(f) by UGC, New Delhi)

Udyog Vidya Nagar, Haliyal - 581 329, Dist.: Uttara Kannada

www.klsvdit.edu.in | principal@klsvdit.edu.in | hodece@klsvdit.edu.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

University / Model Question Paper Scheme & Solution

Faculty Name	:	Rajeshwari A·P / Suraj B Kadli
Course Name	:	Microcontroller
Course Code	:	BEC405A
Year of Question Paper	:	Jun / July 2024
Date of Submission	:	23/01/2025

Faculty Member

HoD

Head of the Department
Dept. of Electronic & Communication Engg.

Dean (Acad.)

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGG
KLS VDIT
HALIYAL

CBCS SCHEME

USN | | | | | | | | | | BEC405A

Fourth Semester B.E./B.Tech. Degree Examination, June/July 2024
Microcontrollers

Time: 3 hrs. Max. Marks: 100

Note: 1. Answer any FIVE full questions, choosing ONE full question from each module.
2. M : Marks , L: Bloom's level , C: Course outcomes.

			M	L	C
		Module – 1			
Q.1	a.	Bring out the difference between Microprocessor and Microcontroller.	6	L2	CO1
	b.	With a neat Architecture diagram, explain the Architecture of 8051 Microcontroller.	10	L2	CO1
	c.	Explain (i) RST (ii) INT Pins of 8051.	4	L1	CO1
		OR			
Q.2	a.	Differentiate between CISC and RISC.	6	L2	CO1
	b.	With a neat diagram, explain the Internal Memory Structure and Programming Model of 8051 Microcontroller.	10	L2	CO1
	c.	List out special features of 8051 Microcontroller.	4	L2	CO1
		Module – 2			
Q.3	a.	Define Addressing Mode. Explain different addressing modes with example.	10	L2	CO2
	b.	Write an ALP to add two 16-bit numbers loaded in R_1R_0 and R_3R_2. Store the result in R_4R_3 and R_2 from MSB to LSB.	10	L3	CO2
		OR			
Q.4	a.	Define Stack. Explain the operation of Stack using Stack Pointer, PUSH and POP Instructions.	10	L2	CO2
	b.	Write an ALP to find largest of N numbers.	10	L3	CO2
		Module – 3			
Q.5	a.	Explain (i) TMOD (ii) TCON register of 8051.	10	L2	CO3
	b.	Assume XTAL = 22 MHz. Write an ALP to generate a square of frequency 1 kHz on Pin P1.2.	10	L2	CO3
		OR			
Q.6	a.	Explain (i) SCON register (ii) Importance of TI Flag.	10	L2	CO3
	b.	Write a C program to transfer "YES" serially at 9600 baud rate. 8 bit data, 1 stop bit, do this continuously.	10	L3	CO3
		Module – 4			
Q.7	a.	Define Interrupt. List the steps involved in Executing an Interrupt.	10	L2	CO4
	b.	Explain Interrupt Vector table of 8051 Microcontroller.	5	L2	CO4
	c.	Explain Interrupt enable register.	5	L2	CO4
		OR			
Q.8	a.	Explain Interrupt Control used in 8051.	10	L2	CO4
	b.	Explain the steps involved in programming serial communication Interrupt.	5	L2	CO4

	c.	Explain **how multiple Interrupts are** handled **in 8051**.	5	L2	CO4
		Module – 5			
Q.9	a.	**Explain DAC Interface with a** neat **diagram and** also write **a program to generate staircase waveform.**	10	L3	CO5
	b.	With **a neat** diagram, write a **program** to **Interface** Stepper Motor to 8051 Microcontroller.	10	L3	CO5
		OR			
Q.10	a.	**Explain the** Interfacing of DC motor **using** C programming.	10	L3	CO5
	b.	**With a** neat diagram, write **a** ALP **to Interface** LCD to 8051 Microcontroller.	10	L3	CO5

BEC405A

Fourth Semester B.E/B.Tech Degree Examination

June/July 2024

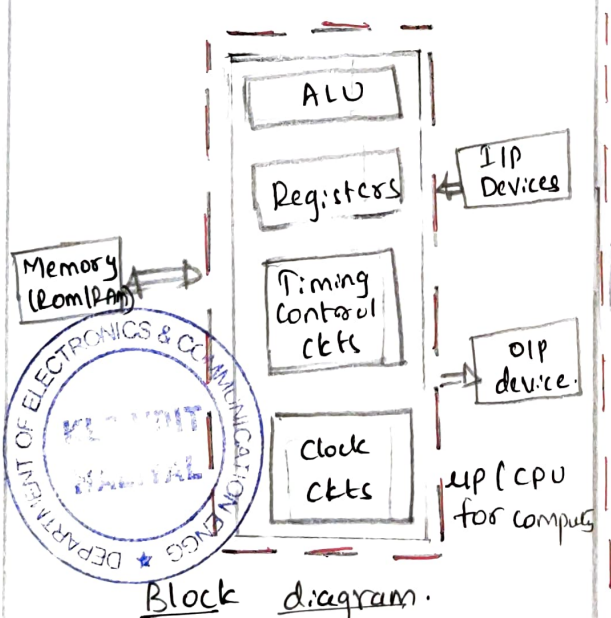
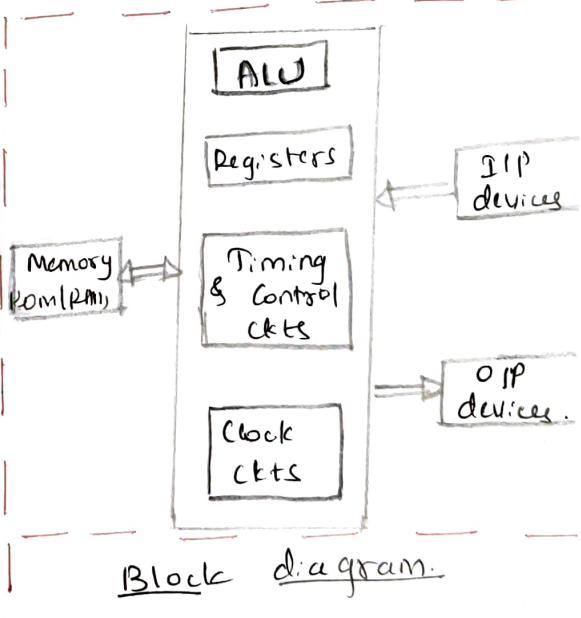
MICROCONTROLLERS.

Time: 3hrs Max Marks: 100

Module - 1

Q1. a) Bring out the difference between Microprocessor and Microcontroller - 6 marks.

Soln:

	Micro Processor	Micro Controller.
①	Its an integrated circuit or pgm logic circuit on which ALU, working registers timing & control ckts & clk ckts are fabricated on a single chip. So its CPU on chip	Its an integrated ckt on which ALU, working registers Timing & control ckts, clock ckts, Memory (ROM/RAM), I/P ports & O/P ports are fabricated on a single chip
②	 Block diagram.	 Block diagram.
③	Used for general purpose	Used for specific purpose.
④	Provides Flexibility	Don't provide flexibility
⑤	Rapid Movement of data from external memory to processor.	Rapid Movement of data within the chip itself.

Contd...

	µp	µc
(6)	It has one or two bit handling instructions.	It has many bits handling instructions.
(7)	More access time for memory & IIo devices	less access time as these are built in
(8)	µp based system requires more hardware	µc based system requires less hardware
(9)	Versatile	Non versatile.

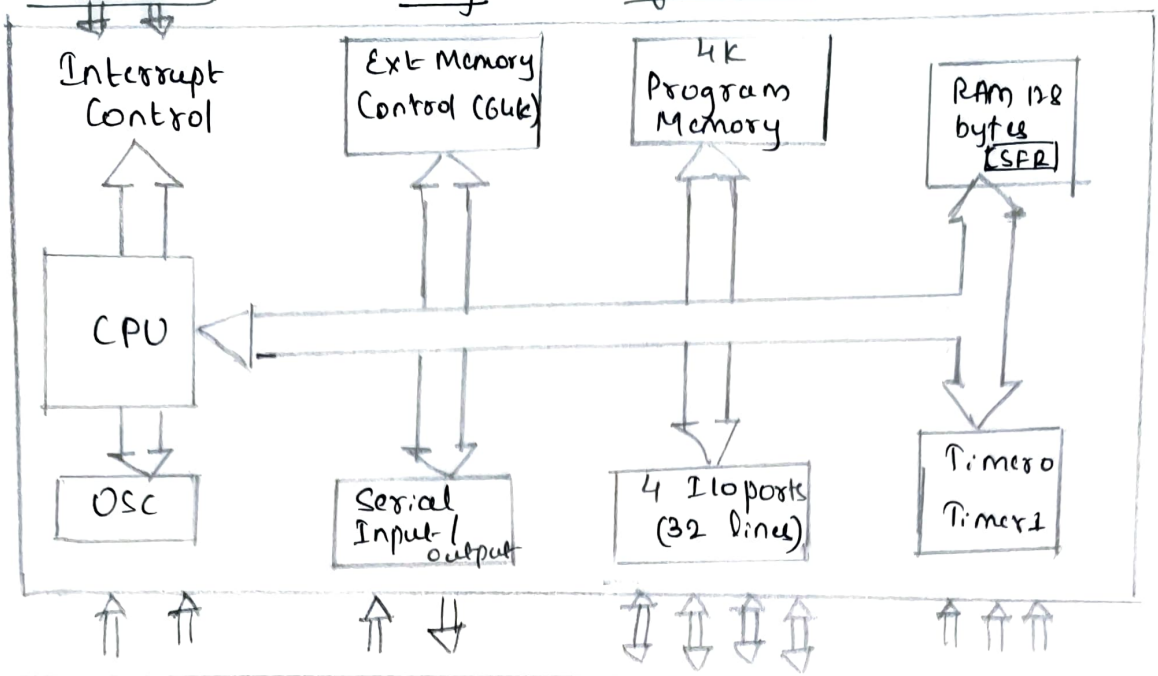
Q1 (b) With a Neat Architecture diagram, explain the Architecture of 8051 Microcontroller. – 10 marks.

Soln:

(1) 8 bit CPU with register A & B
(2) Clock circuits
(3) Internal 4k byte ROM
(4) Internal 128 byte RAM
(5) Two 16 bit timers (T_0 & T_1)
(6) Full duplex serial communication
(7) Four IIo ports (P_0, P_1, P_2, P_3)
(8) 2 External & 3 Internal interrupts (Timer flag 1 serial port)
(9) Two 16 bit registers, Program counter (PC) Data Pointer (DPTR)
(10) 8-bit stack pointer register
(11) 8 bit Program status word (PSW) register.
(12) 21 8bit special function registers (SFR's)

③

Architectural Diagram of 8051.



Q1.c) Explain: (i) RST (ii) INT Pins of 8051 -4 marks.

Soln: (i) RST: Pin 9 (Reset) RST : It is active high signal when pulse (square wave) is applied to this pin Microcontroller will terminate all its activities & Reset.

- Program Counter (PC) is loaded with 0000h.
- A voltage pulse of 2 machine cycle duration applied to this pin causes the µC to reset.
- It causes PC to resumes program execution from start.
- It also sets the contents of all SFR's to zero. except SP, P_0-P_3. & is set to 07h & P_0-P_3 -FFh.

(ii) INT Pins:

- There are assigned as Pin12 : $\overline{INT_0}$ & Pin13: $\overline{INT_1}$
- Interrupt 0 & Interrupt 1 are two interrupt pins that are triggered by external circuits.

Q12a) Difference between CISC & RISC. – 6 marks.

	CISC	RISC
①	Complex instructions set computer	Reduced Instructions set Computis.
②	Support many addressing modes for arithmetic, logical, data transfes & memory access operations.	Fewes Addressing Modes.
③	One operation can be performed by using various instructions.	Every instructions have a particular definite purpose.
④	Instructions are of variable lengths so execution times are different.	Instructions are of limited variable lengths so execution times are definite.
⑤	Ex: 8085, 8086, 8051 uc series.	PIC uc series.
⑥	Less pipelined	Highly pipelined.
⑦	Single Register set	Multiple Register set
⑧	Complex instructions taking multiple cycles.	Simple instructions taking one cycle.
⑨	Variable format instructions.	fixed format instructions
⑩	Instructions are executed by microprogram	Executed by Hardware.

Q2. (b) With a neat diagram explain the Internal Memory Structure & Programming Model of 8051 Microcontroller - 10 marks.

Soln: 8051 has 128 bytes internal RAM, The internal RAM of 8051 is organised into 3 distinct areas.

1) Working Registers
2) Bit addressible Registers.
3) General Purpose Registers.

1) Working Registers.: The first 32 byte from address 00h to 1Fh of internal RAM constitute 32 working registers.

Bank 0 → 8 registers (R0-R7) : 00h to 07h
Bank 1 → " " (R0-R7) : 08h to 0Fh
Bank 2 → " " " : 10h to 17h
Bank 3 → " " " : 18h to 1Fh

* Bits RS0 & RS1 in the PSW determine which bank of registers is currently in use.
* When 8051 is RESET, Bank 0 is selected default.

2) Bit addressible registers.: The 8051 provides 16-byte of bit addressable area. It occupies RAM area from 20h to 2Fh, forming a total of 128 address bits.

3) General Purpose registers: The RAM area above bit addressable area from 30h to 7Fh is called General purpose RAM. Its addressable as byte.

Internal ROM. It has 4K bytes of Internal ROM with address space 0000h to 0FFFh. If pgm addr. exceeds 0FFFh 8051 automatically fetches bytes from external pgm memory. It determined by pins PSEN & EA (Ext or int memory).

* Accumulator (A) - Register. It is a 8 bit register used by CPU in performing arithmetic & logical operations. & also holds the result of the operation. It also used for data movement btn ext memory to MC vice versa.

* B register: Also a 8 bit register used along with A in multiplication & division operations. Both are bit & byte addressable.

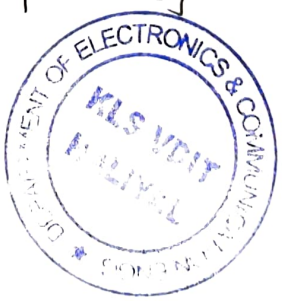
* Program counter: 16 bit register pointing to the next instruction in the program memory.

* Data Pointer (DPTR): Its a 16 bit register split into DPH (high byte) & DPL (low byte), used for external memory addressing.

* Stack Pointer (SP): Its a 8 bit register pointing to the stacks' top in Internal RAM, initialized to 07H by default.

* Special function registers: (SFRs)

- PSW (Program status word): Stores flag like carry auxillary carry, overflow etc.
- TCON, TMOD: Timer control & mode registers.
- SCON SBUF: Serial communication control & Buffer.
- IE, IP: Interrupt Enable & priority.



Q2(c) List out special features of 8051 Microcontrollers - 4marks.

Soln: ① It has 8 bit CPU with registers A & B.

② It has a clock circuit.

③ Internal ROM is of 4k bytes.

④ Internal RAM is of 128 bytes.

⑤ Two 16 bit timers are there (T_0 & T_1)

⑥ It has a full duplex serial communication.

⑦ Four I/o ports are there (P_0, P_1, P_2, P_3)

⑧ 2 External & 3 internal interrupts.

⑨ Two 16 bit registers. PC & DPTR.

⑩ 8 bit stack pointer register (SP)

⑪ 8-bit program status word (PSW) register.

⑫ 21 8bit special function register (SFR's)

Module - 2.

Q3 (a) Define Addressing Mode. Explain different address modes with example. - 10 marks.

Soln: The various methods of accessing the data are called as Addressing Modes. (AM)

① Immediate Addressing Mode: In this the source operand is a constant, the immediate data must be preceded by a "pound" sign "#"

Ex: MOV A, #0FH , MOV R3, #12h.

② Register Addressing Mode: This involves the use of registers to hold the data, that has to be manipulated. Syntax: MOV Rd, Rs.

MOV A, R0, MOV R2, A, MOV R4, DPH

③ Direct Am: The entire 128 bytes of RAM can be accessed using direct Am. Syntax: MOV Rs, add.

Ex: MOV R0, 10h , MOV A, 40h etc.

④ Register Indirect A.M. : In this a register is used to hold the address of the data.

* The register itself is not the address, but the no in the register. Syntax: MOV @Rd, Rs

Ex: MOV @R0, A, MOV 40h, @R0 etc.

⑤ Indexed AM : It is widely used in accessing data element of lookup table entries located in program ROM space.

Ex: MOVC A, @A + DPTR

MOVC A, @A + PC

⑥ Relative A.M : It is used only with conditional jump instructions Ex: SJmp.

⑦ Absolute Addressing: Used only by AJmp & ACALL instructions. These are 2 byte instructions.

Ex: AJmp loop1, ACALL loop1.

⑧ Long Addressing: Used only by LJMP & LCALL These are 3 byte instructions.

Ex: LJmp 5000h

LCALL 6000h.

9) Bit Inherent A.M: In this addressing mode address of the flag which contains the operand is implied in the opcode of the instruction.

Ex: CLR C, SETB C.

Q3)(b) Write an ALP to add 2 16 bit numbers loaded in R_1R_0 & R_3R_2. Store the result in R_6, R_5 & R_4 from MSB to LSB – 10 marks.

Soln: Assume $R_1:R_0$ = First 16 bit number (High : Low byte)

" $R_3:R_2$ = Second " " " "

Result will be stored in $R_6:R_5:R_4$ (MSB: Middle: LSB).

Contd...

```
MOV A, R0  ; load LSB of 1st no
ADD A, R2  ; Add " " 2nd "
MOV R4, A  ; Store " " result in R4.
MOV A, #00H ; Clear Accumulator
JNC NO_CARRY1 ; Jump no carry from LSB addition
INC A      ; Increment A to account for carry
NO_CARRY1:
MOV A, R1  ; load MSB of 1st no
ADDC A, R3 ; Add " " 2nd no & carry
MOV R5, A  ; Store middle byte of result in R5
MOV A, #00h ; Clear 'A' for carry
JNC NO_CARRY2; Jump if no carry for middle byte addition.
INC A.     ; Increment 'A' for carry
NO_CARRY2:
MOV R6, A  ; Store MSB of result in R6
END.
```

Q4(a) Define Stack. Explain the operation of stack pointer PUSH & POP Instructions. - 10 marks.

Soln: ① Stack pointer is the Internal RAM area used by CPU to store & retrieve data.

② Register used here is stack pointer (SP), holds an internal RAM address.

③ When 8051 is reset SP → 07h, RAM location 08h is the 1st location.

④ Storing of CPU register into stack - PUSH When data is to be placed on stack, then SP increments SP → SP+1.

⑤ Loading contents of stack back into CPU is called POP, & SP decrements SP → SP-1.

Contd.

Q4(a)) PUSH direct address.

(10)

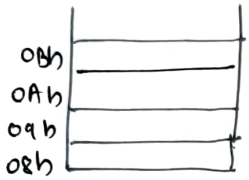
function: Push on to stack.

Description: SP is incremented by one & contents of the flags are copied onto RAM location. None of the flags are affected. 2 Bytes & 2 cycles are used. This supports only direct Am.

Ex: MOV R2, #30h

PUSH 2

Before Execution (BE)

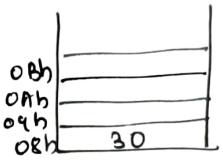


SP = 07h

SP = SP+1

SP = 08h

After Execution (AE)



SP = 08h

2) POP direct address.

function: Pop from the stack

Description: Decrements the SP by 2., flags affected none. Here 2 Bytes & 2 cycles are used.

Operation: (SP) → (SP) - 2.

Ex: MOV R2, #30h

MOV R3, #40h

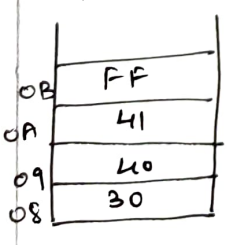
MOV R4, #41h

POP 4

POP 3

POP 2

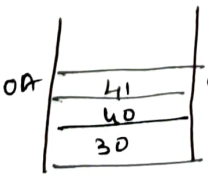
B.E



← SP = 0BH

POP-4

SP → SP-1

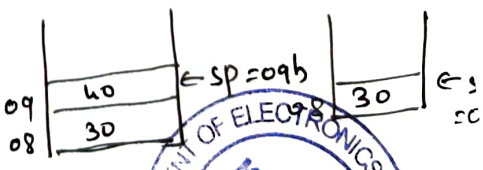


← SP = 0Ah

A.E

POP 3

POP 2



← SP = 09h

(11)

Q4(b) Write an ALP to find largest of N numbers. -10 marks.

Soln:

COUNT Equals 30H Memory location where N (count) is stored.

NUMBERS Equals 40H start of the array of numbers.

RESULT Equals 50H memory location to store the largest number

```
        ORG 0000H       ; Start pgm at memory location 0000h
START:  MOV R0, COUNT   ; Load add of COUNT into R0.
        MOV A, @R0      ; load value of COUNT(N) into 'A'
        MOV R2, A       ; Copy N to R2
        MOV R1, NUMBERS ; Loads the add of NUMBERS
                          assay into R1
        MOV A, @R1      ; Load the 1st no into 'A'
        MOV B, A        ; Store the 1st no as largest
                          (initially)
LOOP:   INC R1          ; Point to next number
        MOV A, @R1      ; Load next no to 'A'
        CJNE A, B, CHECK ; Compare 'A' with 'B'
        SJMP NEXT       ; If Equals, skip updating B
CHECK:  JC NEXT         ; If current no is smaller skip
                          updating B
        MOV B, A        ; Update largest no
NEXT:   DJNZ R2, LOOP   ; Decrement loop counter R2 &
                          repeat if no zero.
        MOV A, B        ; Load the largest number into
                          'A'
        MOV R0, RESULT  ; Load the add of RESULT
                          into R0
        MOV @R0, A      ; Store the largest no in RESULT
        END             ; END of prgm.
```

MODULE - 3

Q5 a) Explain (i) TMOD (ii) TCON register of 8051 -10 marks.

Soln: The TMOD register is an 8 bit register used to configure the mode of operation for Timer0 & Timer1. It is divided into 2 4-bit fields one for each timer.

Bit	Symbol	Description
7	GATE1	Gate control for Timer 1.
6	C/T1	Timer/Counter select for Timer 1
5, 4	M1, M0	Mode select for Timer 1
3	GATE0	Gate control for Timer0
2	C/T0	Timer/Counter select for Timer0
1, 0	M1 M0	Mode select for Timer0.

Bit functions:

- Gate: 1 → Timer operates only if the corresponding INT pin is high & TRx is set
  0 → Timer operates regardless of the INT pin
- C/T : (Counter/Timer Select)
  0 → Timer mode (incremented by the internal clock)
  1 → Counter mode (incremented by external pulse of timer pin)
- Mode select bits. (M1 M0)
- 00 : Mode 0 → 13 bit Timer/Counter
- 01 : Mode 1 → 16 bit Timer/Counter
- 10 : Mode 2 → 8 bit auto reload.
- 11 : Mode 3 → Split Timer Mode (Timer 0 is split into 2 8bit timers).

contd..

Q5a) TCON Register : Its an 8 bit register that contains flags & control bits for Timer 0 & Timer 1 and external interrupts.

Bit	Symbol	Description
7	TF1	Timer 1 overflow flag.
6	TR1	" run control bit
5	TF0	Timer 0 overflow flag
4	TR0	" run control flag
3	IE1	External interrupt 1 eadge flag
2	IT1	External interrupt 1 type contrd
1	IE0	External interrupt 0 edge flag
0	IT0	External interrupt 0 type control.

Bit functions.

- TFx (Timer overflow flags)
  1 → when the timer overflows.
  0 → cleared by software
- TRx → (Timer Run control Bits)
  1 → starts the corresponding timer
  0 → stops the " "
- IE → (Interrupt Edge flags)
  1 → (when corresponding external interrupt is triggered)
- IT (Interrupt type control)
  0 → Level triggered interrupt
  1 → Edge " "

Q5 (b) Assume XTAL = 22MHz. Write an ALP to generate a square of frequency 1kHz on Pin P1.2. (10marks)

Soln: Below is an ALP for 8051 MC to generate a square wave of frequency 1kHz on pin P1.2.
Assume Crystal frequency is 22MHz

Contd...

Q.5) (b)

Machine cycle is XTAL/12 = 22MHz/12 = 1.833 MHz

Machine cycle time = $\frac{1}{1.833\,MHz} \approx 0.546\,\mu s$

For a 1 kHz square wave, the period T is

1/1kHz = 1ms & high/low times will be = 0.5ms = $\frac{T}{2}$

Calculation of Timer Reload value

The number of machine cycles for 0.5 ms

Timer counts = $\frac{0.5\,ms}{0.546\,\mu s} \approx 916$ counts.

The timer counts down from 65536 − 916 = 64620 of which is FBDC is hex.

ALP code:

```
ORG 00H          ; Start at memory location 0
MAIN: MOV TMOD, #01H;  Timer 0 in Mode 1
      MOV TH0, #0FBH; Load high byte of Timer 0
      MOV TL0, #0DCH;   "  low  "  "  "
      SETB TR0        ; Start Timer 0
TOGGLE:
      JNB TF0, TOGGLE ; Wait for Timer 0 to overflow
       CLR TF0        ; Clear Timer 0 overflow flag
       CPL P1.2       ; Toggle P1.2
      MOV TH0, #0FBH ; Reload High Byte
      MOV TL0, #0DCH ; Reload Low Byte
       SJMP TOGGLE   ; Repeat the loop
       END.
```

Q6 (a) Explain (i) SCON register (ii) Importance of TI flag (10 marks)

Soln: SCON Register. (Serial control Register) in the 8051 uC is an 8-bit register used to configure & control the operation of the serial port. It determines the mode of operation, baud rate, & other key features of the serial communication system in the 8051.

7	6	5	4	3	2	1	0
SM_0	SM_1	SM_2	REN	TB_8	RB_8	TI	RI

SM_0 → Serial Mode bit 0: Used with SM_1 to select the serial communication mode

SM_1 → Serial Mode bit 1; Used with SM_0 to select the serial communication mode.

SM_2 → Multiprocessor communication Enable (used in Mode 2 & Mode 3)

REN → Receive Enable: Enables or disables the reception of data

TB_8 → Transmit Bit 8: Used to transmit the 9th data bit in Mode 2 & 3.

RB_8 → Receive bit 8: stores the 9th data bit received in Mode 2 & 3.

TI → Transmit Interrupt flag: Set when Transmission is complete

RI → Receive Interrupt flag: Set when a byte is received.

Modes: (1) SM0 & SM1 → Determines the mode of serial communication

(2) SM2 → In Mode 2 & 3, SM2 is used for multiprocessor communication if set the receives only responds when the 9th bit is 1

Contd...

Q6 (a) (3) REN : Receive Enable

1 → Enables reception of serial data

0 → the serial port ignores incoming data

(4) TB8 : In Mode 2 & 3 this is used to send the 9th bit

(5) RB8 : (Receive bit 8) In Mode 2 & 3 this stores the 9th bit of received data

(6) TI (Transmit Interrupt)

1 → set by hardware when byte has been transmitted

(7) RI → (Receive Interrupt)

Set to 1 by hardware when byte data is received.

(ii) Importance of TI flag: (Transmit Interrupt flag) in a uC is a crucial part of the serial communication system, typically in UART (Universal asynchronous Receiver Transmitter). It is specifically used to indicate the status of data transmission.

(1) Its a transmission status Indicator. Once the data transmission is complete the flag is set to inform the pgm that the transmitter is ready for next byte.

(2) When the TI flag is set, it ensures the transmitter buffer is empty, allowing next byte of data to be loaded without overwriting the current transmission.

(3) It helps in interrupt handling by triggering a transmit interrupt & enabling uC to execute specific actions.

(4) It allows the uC to operate in parallel on other tasks instead of waiting for transmitter to get ready.

(5) It prevents the errors & ensures data integrity by signalling when its safe to send next data.

Q6(b) Write a C program to transfer "YES" serially at 9600 baud rate, 8 bit data, 1 stop bit do this continuously. - 10 marks.

Soln:

```
# include <reg51.h>
# define BAUD_RATE 9600 // Define the desired baud rate

Void UART_Init (void); // Function to initialize UART
Void UART_Transmitt (char data); // Function to transmitt
                                 a character.
Void UART_Transmit String (chas *str); // Function to
                                 transmit a string.

Void main (void) {
    UART_Init(); // Initialize UART
    while (1) {
        UART_Transmit String ("YES"); // Transmit "YES"
                                 continuously
    }
}
    Void UART_Init (void) {
        TMOD = 0x20; // Timer 1 in mode 2 (auto-reload
                     mode)
        TH1 = 256 - (11059200 / (12 * 32 * BAUD_RATE));
                     // Set baud rate for 9600
        TR1 = 1; // Start Timer 1.
        SCON = 0x50; // Configure UART: Mode 1, 8 bit
        data, 1 stop bit
    }
    Void UART_ Transmit (chas data) {
        SBUF = data; // Load the data into UART buffer.
        while (TI==0); // Wait until the transmission
                       is complete
        TI = 0; // Clear the TI flag
    }
```

Q6(b) Contd...

```
Void UART_Transmit string (char *str) {
  while (*str) {
    UART_Transmit (*str++); // Transmit each character
                               in the string.
  }
}
```

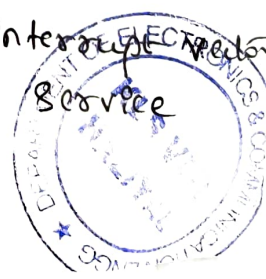
MODULE 4

Q7 (a) Define Interrupt. List the steps involved in Executing an Interrupt (10 marks)

Soln: An interrupt is a signal sent to the processor by hardware or software to indicate that an event needs immediate attention. Interrupt temporarily halt the normal execution of a program, allowing the processor to address the event after which it resumes the previous execution.

Steps involved in Executing an Interrupt.

1. An interrupt is generated by hardware or software
2. IRQ (Interrupt request), the interrupt signal is sent to the CPU notifying it of the event
3. Interrupt Acknowledgment: The CPU acknowledges the interrupt signal & determines its priority
4. The CPU saves the current state of the program, including the PC, registers & flags to ensure smooth resumption later.
5. Interrupt Vector lookup: The CPU uses an interrupt vector table to find the address of the interrupt service routine (ISR) associated with the interrupt.



Q7 (a) Contd..

(6) The CPU executes the interrupt service routine to handle the event.

(7) The ISR performs the necessary operation to address the interrupt such as reading data from a device or updating a system status.

(8) Once the ISR completes, the CPU restores the saved state, including the program counter & registers.

(9) The CPU resumes execution of the interrupt pgm from where it was paused.

Q7 (b) Explain (IVT) interrupt vector table of 8051 Microcontroller. (5 marks)

Soln: IVT in the 8051 MC is a table that contains the starting addresses of Interrupt service Routine (ISRs) Each interrupt source is assigned a specific memory where its ISR begins. When an interrupt occurs, the 8051 automatically jumps to the corresponding address in the IVT to execute the ISR for that Interrupt.

Interrupt in 8051 & their Vector Addresses.

Interrupt	Interrupt flag	Vector Address	Priority (default)
Reset	—	0x0000	Highest
External Interr-upt 0 (INT0)	IE_0	0x0003	Low
Timer 0 Overflow (TF0)	TF_0	0x000B	Low
External Interrupt 1 (INT1)	IE_1	0x0013	Low
Timer 1 Over flow (TF1)	TF_1	0x001B	Low
Serial Communication (RI|TI)	RI|TI	0x0023	Low

Q7 (20) (c) Explain Interrupt enable register. (5marks)

Soln: The Interrupt Enable (IE) register in the 8051 uC is an 8 bit register used to enable or disable specific interrupts. Each bit in the register corresponds to a particular interrupt source. By setting or clearing these bits, the uC can control which interrupts are allowed to trigger an interrupt service routine (ISR)

Bit structure of the IE register.

Bit	Symbol	Description	State
7	EA	Global Interrupt Enable	0 = Disable all Interrupt 1 = Enable " "
6	—	Reserved (Not used)	Must be 0
5	ES	Serial Interrupt Enable	0 = Disable 1 = Enable
4	ET1	Timer 1 Interrupt Enable	0 = Disable 1 = Enable
3	EX1	External Interrupt 1 Enable	0 = Disable 1 = Enable
2	ET0	Timer 0 Interrupt Enable	0 = Disable 1 = Enable
1	EX0	External Interrupt-0 Enable	0 = Disable 1 = Enable
0	—	Reserved (Not used).	Must be 0

Q8(a) Explain Interrupt control used in 8051 (10marks)

Soln: In the 8051 uC interrupt control allows the processor to handle external or internal events without continuously polling for them. This improves efficiency by enabling the uC to perform its primary task while being ready to respond.

Q8(a) Contd...

Interrupt sources, priorities, & IE, Interrupt vector table will be explained in Q7(b) & (c).

Interrupt handling process refer Q(7)(a).

Q8(b) Explain the steps involved in programming serial communication Interrupt. (5 marks).

Soln: Programming the serial communication interrupt in the 8051 involves several steps to ensure the microcontroller can handle data transmission and reception through interrupt efficiently. The interrupt is triggered by the Serial Interface when either a byte is transmitted (TI) or received (RI). Here is how you can program the serial communication interrupt step by step.

① Step1: Configure the Serial Port: Use the SCON to configure mode of operation. Commonly used mode Mode1, which requires setting SM0 = 0 & SM1 = 1.

② Set the Baud Rate: Configure Timer 1 in Mode 2 to set the baud rate. Load the appropriate value into TH1 for the desired baud rate.

③ Enable the Serial Interrupt (IE).
ES = 1 to enable the serial interrupt
EA = 1 to " " global " system.

④ The ISR is executed when the serial interrupt occurs (TI or RI is set).
① RI = 1 A byte has been received read it from SBUF.
② TI = 1 A byte has been transmitted, load the next byte into SBUF if needed.

⑤ Enable interrupts globally. EA = 1.

⑥ Load Data for Transmission: If you intend to transmit data, load the first byte into the SBUF register to initiate transmission.

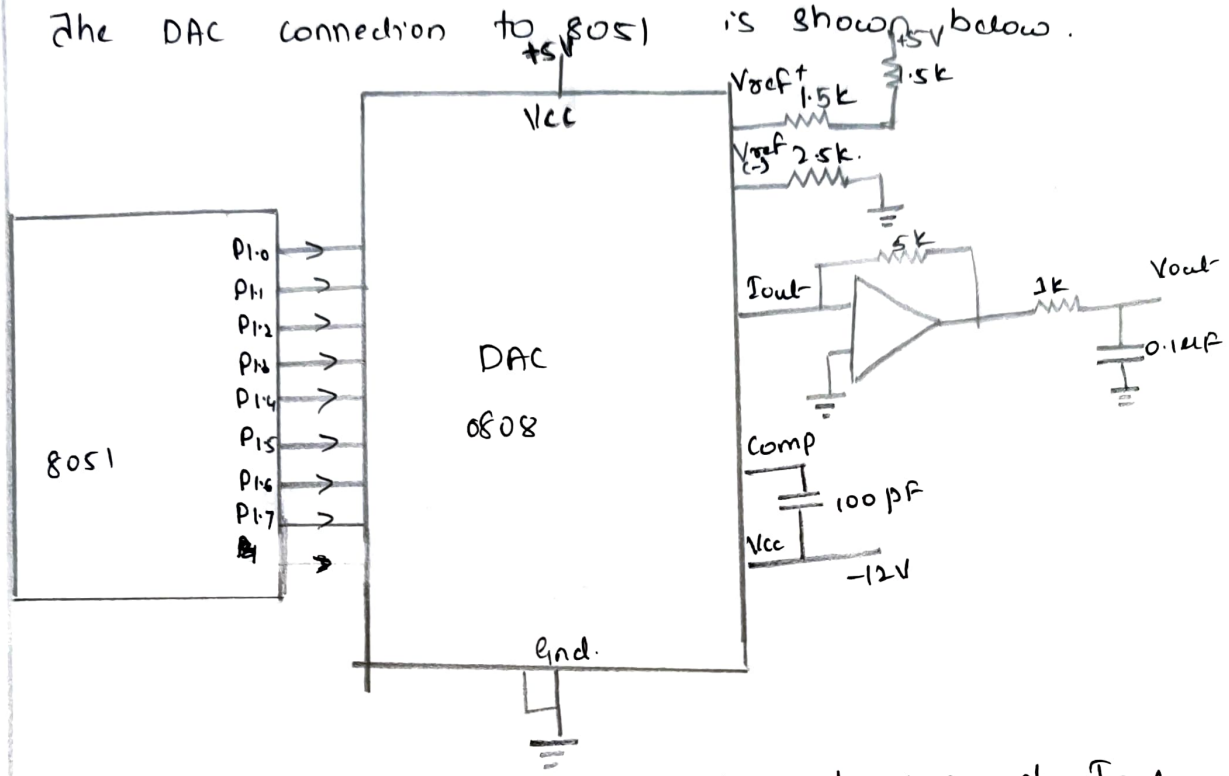
Q8(c) Explain how multiple interrupts are handled in 8051 (5 marks)

Soln: Explained in Q(7)(b)

MODULE-5

Q9(a) Explain DAC interface with a neat diagram and also write a pgm to generate staircase waveform. (10 marks)

Soln: The DAC connection to 8051 is shown below.



① The digital inputs are converted to current Iout

② Then by connecting the resistors to Iout pin we convert the result to voltage.

③ The total current Iout is a function of binary nos at D_0-D_7 i/ps of DAC 0808 & the reference current Iref.

ie $I_{out} = I_{ref}\left(\frac{D_7}{2} + \frac{D_6}{4} + \frac{D_5}{8} + \frac{D_4}{16} + \frac{D_3}{32} + \frac{D_2}{64} + \frac{D_1}{128} + \frac{D_0}{256}\right)$

usually $I_{ref} = 2mA$.

④ Ideally we connect o/p pin to resistor convert this current to vtg.

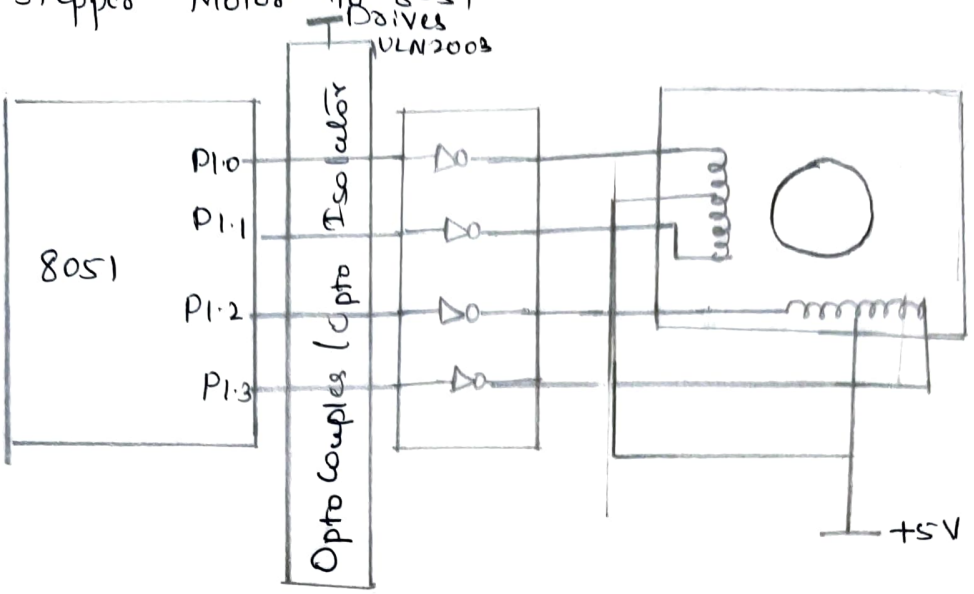
Program to generate staircase waveform.

```
        ORG 0000H
        MOV P1, #00H   ; clear the port 1.
LOOP:   MOV R0, #00H   ; Initialize R0
```

Contd...

```
INCREMENT
        MOV A,R0  ; Move value of R0 → A.
        MOV P1,A  ; Send the value to Port 1 (DAC i/p)
        ACALL DELAY; Call delay subroutine.
        INC R0    ; Increment the value in R0
        CJNE R0, #FFh, INCREMENT; continue until
                                  R0 = FFh (max value)
        SJMP LOOP ; Repeat the process for continous
                    waveform
DELAY:
        MOV R1, #0FFh ; Outer loop
D1:     MOV R2, #0FFh ; Inner loop
D2:     DJNZ R2, D2   ; Decrement R2
        DJNZ R1, D1   ; Decrement R1
        RET           ; Return from Delay
        END.
```

Q9 (b) With a neat diagram, write a program to Interface Stepper Motor to 8051 MC. (10 marks)



① Stepper Motor typically has 4 coils (unipolar type) or 2 coils (bipolar type)

② The motor drives (eg ULN2003) is used to amplify the control signals from the MC to drive the motor.

③ The driver inputs are connected to a port of 8051

④ The outputs of the drives are connected to motor windings.

⑤ The stepper motor is rotated by energizing its coils in a specific sequence. for example in full step mode the sequence is A → AB → B → BC → C → CD → D → DA.

Program:

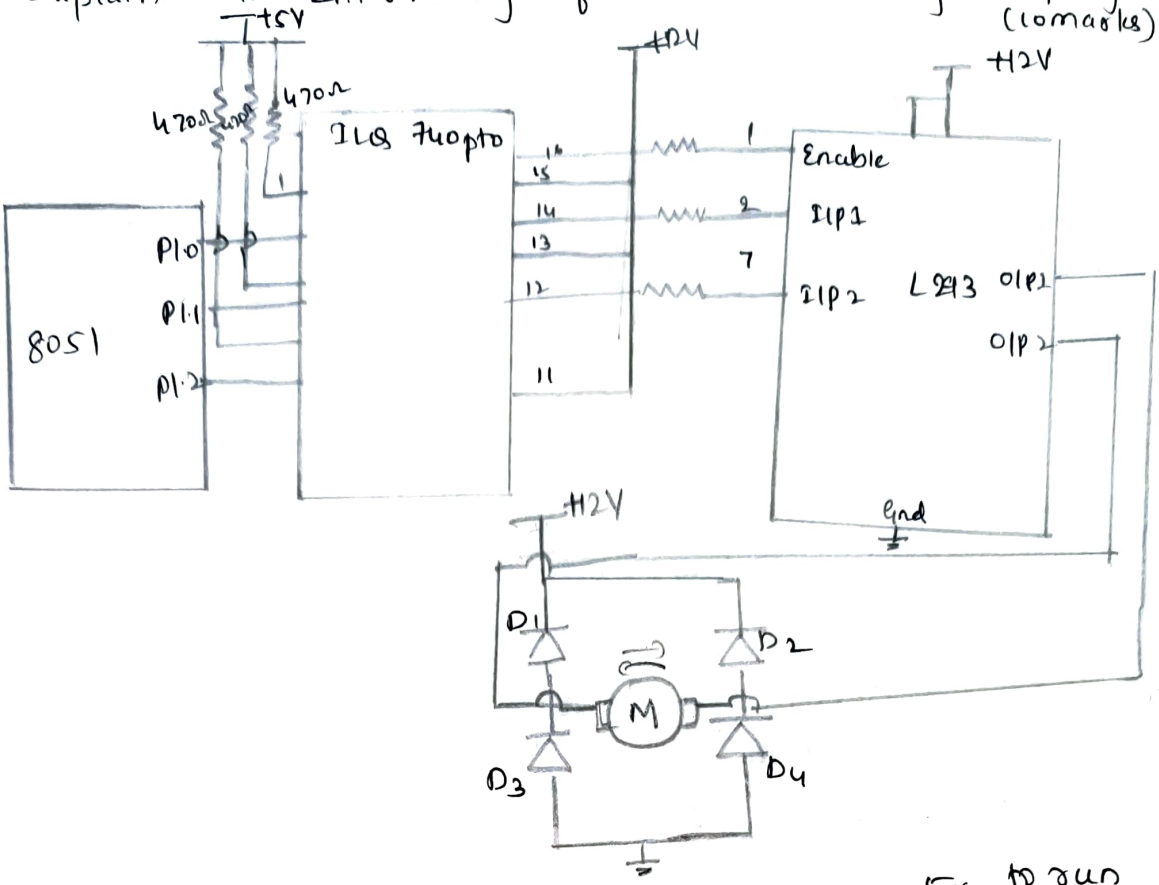
```
ORG 0000H.
MOV P2, #00H ;  Clear Port 2

MAIN_LOOP:
    ; Full Step Sequence
      MOV A, #09H ; 1st step A&D coil ON (1001)
      ACALL BELAY ; call delay subroutine
        MOV P2,A    ; Send to Port 2
        MOV A, #0CH ; 2nd step : B&D ON (1100)
        ACALL DELAY
```

Contd...

```
MOV P2, A ; Send to Port 2
MOV A, #06h ; 3rd step B & C ON (0110)
ACALL DELAY
MOV P2, A.
MOV A, #03H ; 4th step A & C ON (0011)
ACALL DELAY
MOV P2, A.
SJMP MAIN_LOOP ; Repeat the sequence.
; Subroutine for Delay
DELAY: MOV R1, #0FFh ; Outer loop
D1: MOV R2, #0FFh ; Inner "
D2: DJNZ R2, D2 ; Decrement R2
DJNZ R1, D1 ; " R1
RET ; Return from Delay
END.
```

Q.10 (a) Explain the Interfacing of DC Motor using C programming (10 marks)

(04)



(1) L293 IC Motor Drives which allows the motor to run in either direction.

(2) Opto Isolator provides additional protection of 8051

(3) Uses a separate power supply for motor & L293 than 8051

(3) Diodes are used to protect the transistors from reverse bias

(4) 16 Pin IC (2 DC motors can be controlled using IC)

(5) A switch is connected to pin P2.5

(6) If SW=0 DC motor moves clockwise.

(7) If SW=1 DC " " Anticlockwise

C- Program

```
#include <reg51.h>
  sbit  SW = P2^5;
  sbit  Enable = P1^0;
  sbit  Clockwise = P1^1;
  sbit  Anticlockwise = P1^2;
```

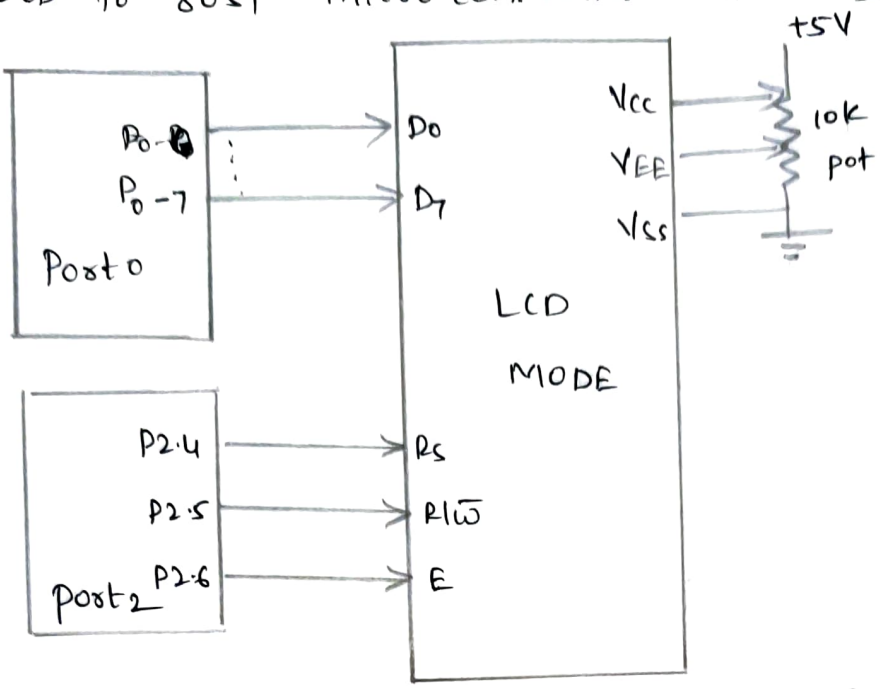
Contd...

```
Void main ()
{
    sw = 1 ;
    Enable = 0;
  clockwise = 0;
  Anticlockwise = 0;
while (1)
{
   Enable = 1;
     if (sw = = 1)
     {
       clockwise = 0;
       Anticlockwise = 1;
     }
    else
     {
       Close wise = 1;
       Anticlockwise = 0;
        }
     }
}
```

Contd...

```
; Display Message on LCD
    MOV  DPTR, # MESSAGE
    CALL LCD_WRITE
    END.
; Subroutine to send command to LCD
LCD-CMD:
    MOV  A, @DPTR
    INC  DPTR
    MOV  R0, A
    CLR  P2.0  ; RS=0
    CLR  P2.1  ; RW=0
    MOV  A, R0
    CALL LCD_NIBBLE; Send high nibble
    MOV  A, R0
    SWAP A
    CALL LCD_NIBBLE
    RET
; Subroutine to send data.
LCD_WRITE:
    MOV  A, @DPTR
    INC  DPTR
    MOV  R0, A
    SETB P2.0  ; RS=1
    CLR  P2.1  ; RW=0
    CALL LCD_NIBBLE
    MOV  A,R0
    SWAP A
    CALL LCD_NIBBLE
    RET
```

Q10(b) With a neat diagram write a ALP to interface LCD to 8051 Microcontrollers. (10marks)



① LCD is interfaced by using Port 0 & $P_{2.4}$ to $P_{2.6}$

② When Vtg is applied electrostatic field is created this allows the light to pass through segment

③ LCD needs a driving circuit.

④ LCD & driving ckts are integrated in LED display module.

⑤ The display module has one command register & one data register.

⑥ Interfacing ckt allows MC to send data to LCD & also to read.

ALP:

```
ORG 0000H;
Initialize Ports
MOV P1, #0FFH ;
MOV P2, #0FFH ;   } Set port 1 & 2 as output
; LCD Initialization
    MOV  DPTR, #LCD_INIT
    CALL LCD_CMD
```

8(c) (b) Contd...

```
; Subroutine to send a nibble to LCD
LCD_NIBBLE:
        MOV P1,A      ; Send data on P1 (D4-D7)
        SETB P2.2     ; EN=1 (Enable LCD)
        NOP           ; Small delay
        CLR P2.2      ; EN=0 (Disable LCD)
        NOP           ; Small delay
        RET

; Initialize LCD
LCD_INIT
        DB 38H ; function set
        DB 0CH ; display on
        DB 06H ; Entry mode
        DB 01H ; Clear display
        DB 02H ; Return Home
        RET
```

Head of the Department
Dept. of Electronic & Communication Engg.
K.L.S V.D.I.T., HALIYAL (U.K.)