Question 1(a) [3 marks]

List Common features of Microcontrollers.

Answer:

Feature	Purpose
CPU Core	Process instructions
Memory (RAM/ROM)	Store program and data
I/O Ports	Interface with external devices
Timers/Counters	Measure time intervals
Interrupts	Handle asynchronous events
Serial Communication	Transfer data with other devices

Mnemonic: "CRITICS: CPU ROM I/O Timers Interrupts Comm Serial"

Question 1(b) [4 marks]

Explain the functions of ALU.

Answer:

Function	Description
Arithmetic Operations	Addition, subtraction, increment, decrement
Logical Operations	AND, OR, XOR, NOT, comparison
Data Movement	Transfer between registers and memory
Flag Setting	Update status flags based on operation results

Diagram:

```
        +----------+
        |   ALU    |
        |          |
Input   |          | Output
Data    |          | Data
----->  |          |----->
        |          |
        +----------+
             |
             v
        Status Flags
        (Z, C, S, P)
```

Mnemonic: "ALFS: Arithmetic Logic Flags Status"

Question 1(c) [7 marks]

Define: Memory, Operand, Instruction Cycle, Opcode, CU, Machine Cycle, CISC

Answer:

Term	Definition
Memory	Storage unit that holds data and instructions
Operand	Data value or address used in an operation
Instruction Cycle	Complete process of fetching and executing an instruction
Opcode	Operation code that specifies the instruction type
CU	Control Unit that coordinates processor operations
Machine Cycle	Basic operation cycle consisting of T-states
CISC	Complex Instruction Set Computer with rich instruction set

- **Memory**: Organized array of storage cells with unique addresses

- **Operand**: Data elements that instructions operate upon

- **Instruction Cycle**: Fetch-decode-execute sequence for each instruction

- **Opcode**: Binary code that tells processor what operation to perform

Diagram:

```
Instruction Cycle:
+--------+    +--------+    +--------+
| Fetch  |--->| Decode |--->|Execute |
+--------+    +--------+    +--------+
```

Mnemonic: "MO-ICO-MC: Memory-Operand-Instruction-Control-Operation-Machine-Complex"

Question 1(c OR) [7 marks]

i) Define: Microprocessor. ii) Compare Von-Neumann and Harvard architecture.

Answer:

i) Microprocessor Definition:

```
An integrated circuit containing the CPU functionality of a computer,
capable of fetching, decoding, and executing instructions with ALU and
control circuitry on a single chip.
```

ii) Von-Neumann vs Harvard Architecture:

Feature	Von-Neumann	Harvard
Memory	Single shared memory	Separate program & data memory
Bus	Single bus for data & instructions	Separate buses
Speed	Slower (memory bottleneck)	Faster (parallel access)
Complexity	Simpler design	More complex
Applications	General computing	Real-time systems

Diagram:

```
Von-Neumann:
+-------+            +-------+
| CPU   |<=======>| Memory|
+-------+            +-------+

Harvard:
+-------+            +-----------+
| CPU   |=======>| Program   |
|       |            | Memory    |
|       |            +-----------+
|       |            +-----------+
|       |<=======>| Data      |
+-------+            | Memory    |
                     +-----------+
```

Mnemonic: "Harvard Has Separate Spaces"

Question 2(a) [3 marks]

Explain various Registers of 8085 microprocessor.

Answer:

Register	Size	Function
Accumulator (A)	8-bit	Main register for arithmetic & logic
General Purpose (B,C,D,E,H,L)	8-bit	Temporary data storage
Program Counter (PC)	16-bit	Address of next instruction
Stack Pointer (SP)	16-bit	Points to top of stack
Flag Register	8-bit	Status flags (Z,S,P,CY,AC)

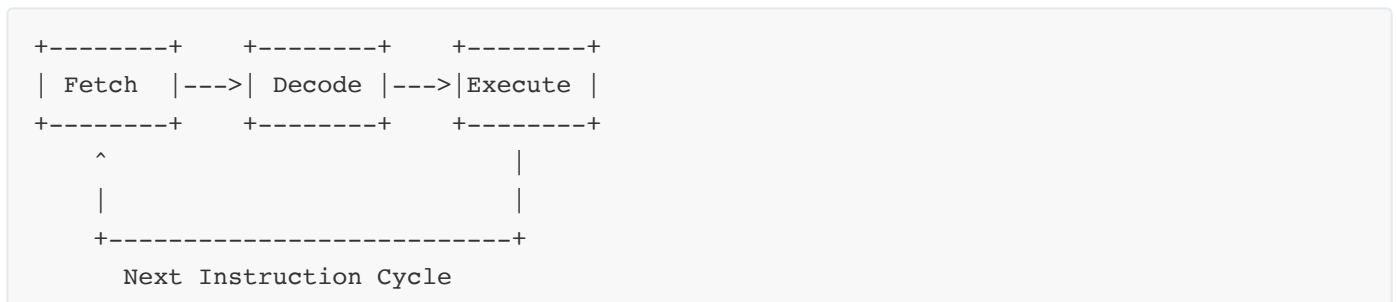**Mnemonic:** "AGSF: Accumulator-General-Stack-Flags"

Question 2(b) [4 marks]

Explain Fetching, Decoding and Execution of Instruction.

Answer:

Phase	Activity	Hardware Involved
Fetching	Get instruction from memory address in PC	PC, Address bus, Memory
Decoding	Identify operation type and operands	Instruction Register, Control Unit
Execution	Perform specified operation	ALU, Registers, Data bus

Diagram:

```
+--------+    +--------+    +--------+
| Fetch  |--->| Decode |--->|Execute |
+--------+    +--------+    +--------+
    ^                           |
    |                           |
    +---------------------------+
      Next Instruction Cycle
```

- **Fetching**: PC sends address to memory, instruction loaded into IR

- **Decoding**: Control unit interprets instruction opcode and addressing mode

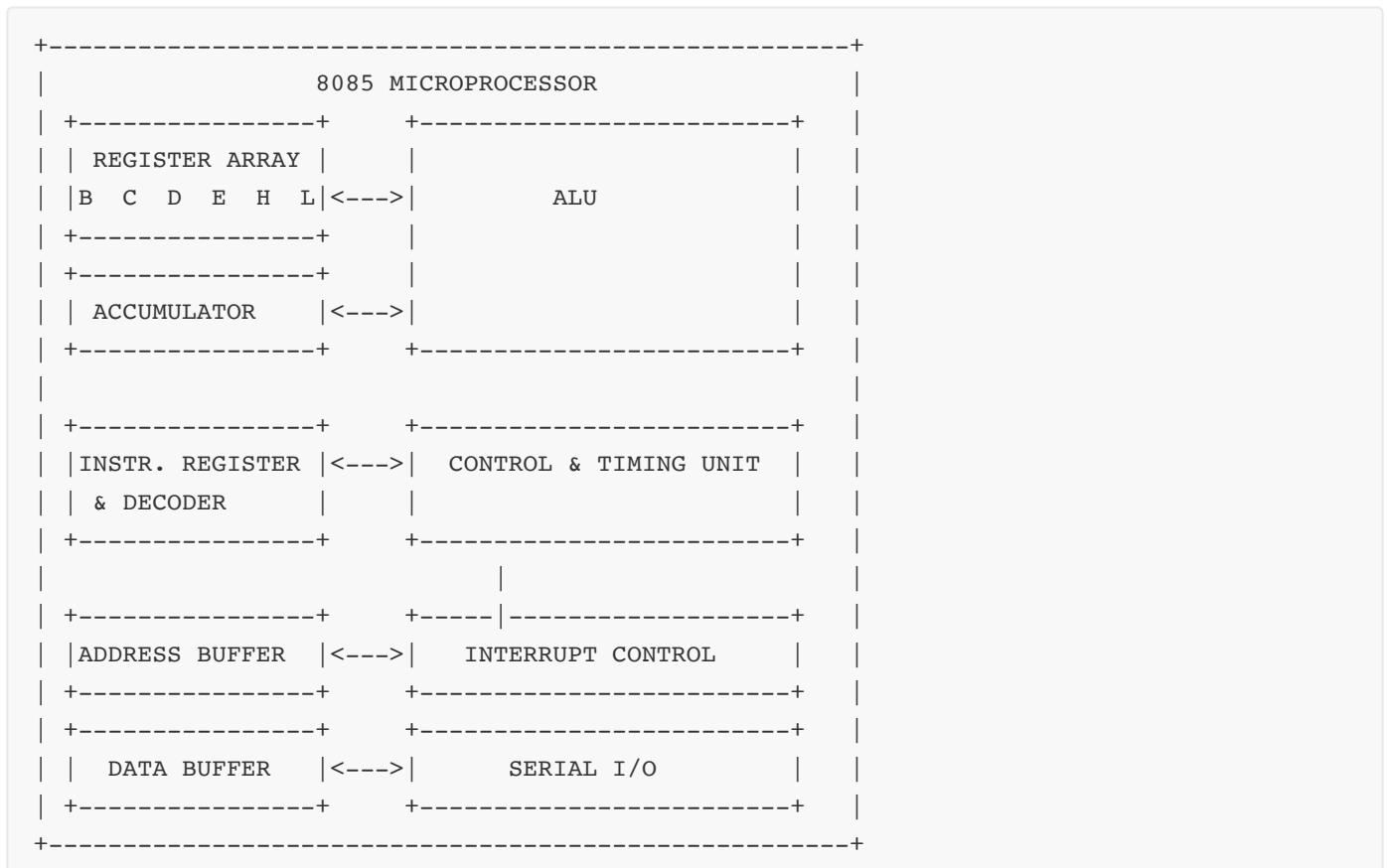- **Execution**: ALU performs arithmetic/logic, data moves between registers/memory

Mnemonic: "FDE: First Get, Then Understand, Finally Do"

Question 2(c) [7 marks]

Describe block diagram of 8085 microprocessor with the help of neat diagram.

Answer:

Block	Function
ALU	Arithmetic & logical operations
Register Array	Temporary data storage
Instruction Register & Decoder	Hold & interpret instructions
Control & Timing Unit	Generate control signals
Address Buffer	Interface with address bus
Data Buffer	Interface with data bus
Serial I/O	Communication with SID/SOD
Interrupt Control	Handle interrupt requests

Diagram:

```
+-------------------------------------------------------+
|                  8085 MICROPROCESSOR                  |
| +---------------+     +-------------------------+   |
| | REGISTER ARRAY |     |                         |   |
| |B  C  D  E  H  L|<--->|          ALU            |   |
| +---------------+     |                         |   |
| +---------------+     |                         |   |
| |  ACCUMULATOR   |<--->|                         |   |
| +---------------+     +-------------------------+   |
|                                                       |
| +---------------+     +-------------------------+   |
| |INSTR. REGISTER |<--->|  CONTROL & TIMING UNIT  |   |
| | & DECODER     |     |                         |   |
| +---------------+     +-------------------------+   |
|                              |                        |
| +---------------+     +-----|-------------------+   |
| |ADDRESS BUFFER  |<--->|    INTERRUPT CONTROL     |   |
| +---------------+     +-------------------------+   |
| +---------------+     +-------------------------+   |
| |  DATA BUFFER   |<--->|       SERIAL I/O         |   |
| +---------------+     +-------------------------+   |
+-------------------------------------------------------+
```

- **Core Components**: ALU and registers form processing core
- **Control Path**: Instructions flow through register, decoder, control unit
- **Data Path**: Data moves through buffers to/from external buses
- **Timing**: Synchronizes all operations via internal clock

Mnemonic: "RAID: Registers-ALU-Instructions-Decoders"

Question 2(a OR) [3 marks]

Compare Microprocessor & Microcontroller.

Answer:

Feature	Microprocessor	Microcontroller
Design	CPU only	CPU + peripherals
Memory	External	Internal (RAM/ROM)
I/O ports	Limited	Many built-in
Applications	General computing	Embedded systems
Cost	Higher	Lower
Example	Intel 8085/8086	Intel 8051

Mnemonic: "Micro-P Processes, Micro-C Controls"

Question 2(b OR) [4 marks]

Explain De-multiplexing of Address and Data buses for 8085 Microprocessor.

Answer:

Step	Action
1	ALE signal goes high
2	Lower address (A0-A7) appears on AD0-AD7
3	Latch captures address using ALE
4	ALE goes low, AD0-AD7 now carries data

Diagram:

```
     8085              Latch           Memory
     ----              -----           ------
                         |
AD0-AD7 ----+------> |   |-----> A0-A7 ---> Memory
            |           ---            Address
            |            ^
            |            |
            v            |
       Data Bus        ALE
```
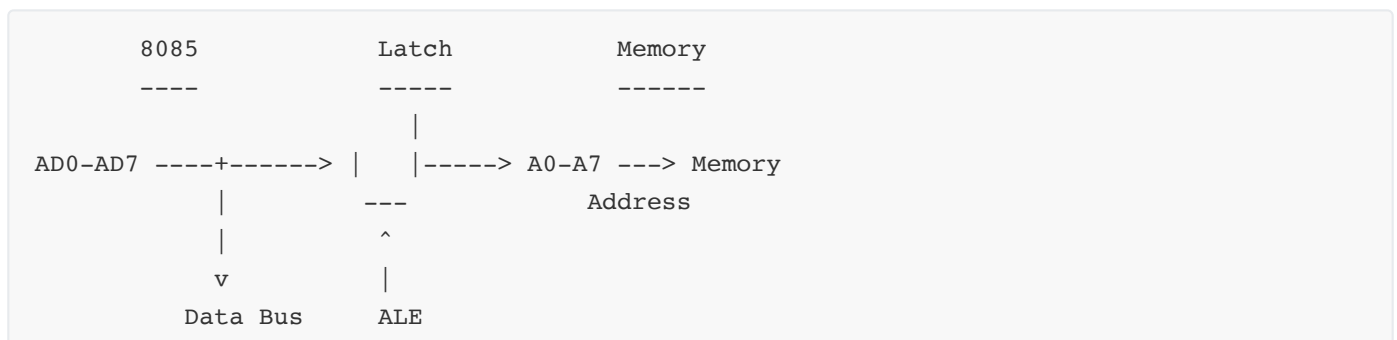
- **Multiplexing**: AD0-AD7 pins carry address and data at different times
- **ALE Signal**: Address Latch Enable controls when address is captured

- **8-bit Latch**: Holds lower address bits during entire machine cycle
- **Timing**: Address valid only during high state of ALE pulse

Mnemonic: "ALAD: ALE Latches Address before Data"

Question 2(c OR) [7 marks]

Describe Pin diagram of 8085 microprocessor with the help of neat diagram.

Answer:

Pin Group	Function
Address/Data	Multiplexed AD0-AD7, A8-A15
Control	RD, WR, IO/M, S0, S1, ALE, CLK
Interrupts	INTR, RST 5.5-7.5, TRAP
DMA	HOLD, HLDA
Power	Vcc, Vss
Serial I/O	SID, SOD
Reset	RESET IN, RESET OUT

Diagram:

```
            +-------+
    X1 -->|1    40|<-- Vcc
    X2 -->|2    39|<-- HOLD
RESET OUT-->|3    38|<-- HLDA
RESET IN -->|4    37|<-- CLK
   IO/M -->|5    36|<-- RESET IN
     S1 -->|6    35|<-- READY
     RD -->|7    34|<-- IO/M
     WR -->|8    33|<-- S1
    ALE -->|9    32|<-- RD
     S0 -->|10   31|<-- WR
    A15 -->|11   30|<-- ALE
    A14 -->|12   29|<-- S0
    A13 -->|13   28|<-- A15
    A12 -->|14   27|<-- A14
    A11 -->|15   26|<-- A13
    A10 -->|16   25|<-- A12
     A9 -->|17   24|<-- A11
     A8 -->|18   23|<-- A10
    AD7 -->|19   22|<-- A9
    AD6 -->|20   21|<-- A8
            +-------+
```

- **Address Pins**: A8-A15 (8) and multiplexed AD0-AD7 (8) for 16-bit addressing

- **Control Pins**: Generate timing and control signals for peripheral devices

- **Interrupt Pins**: Hardware interrupt handling with priority levels

- **Clock**: X1, X2 for crystal connection, CLK for synchronization

- **Power**: Vcc (+5V) and Vss (Ground) for power supply

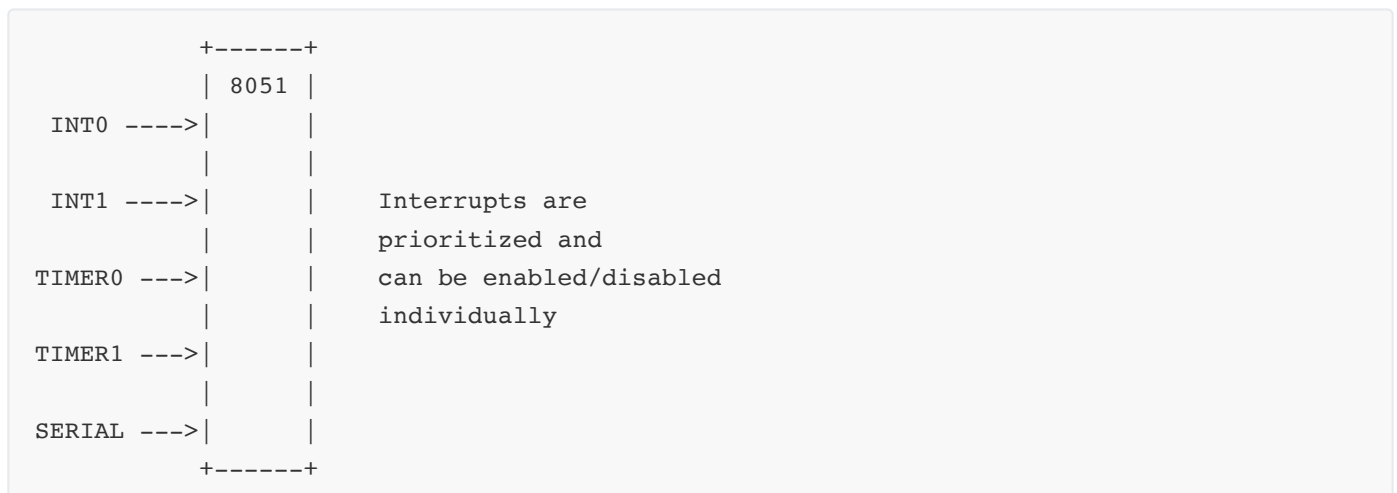**Mnemonic:** "ACID-PS: Address-Control-Interrupt-DMA-Power-Serial"

Question 3(a) [3 marks]

Explain interrupts of 8051 microcontroller.

Answer:

Interrupt	Vector	Priority	Source
External 0	0003H	1 (IP.0)	Pin INT0 (P3.2)
Timer 0	000BH	2 (IP.1)	Timer 0 overflow
External 1	0013H	3 (IP.2)	Pin INT1 (P3.3)
Timer 1	001BH	4 (IP.3)	Timer 1 overflow
Serial	0023H	5 (IP.4)	Serial port events

Diagram:

```
          +------+
          | 8051 |
  INT0 ---->|      |
          |      |
  INT1 ---->|      |   Interrupts are
          |      |   prioritized and
 TIMER0 --->|      |   can be enabled/disabled
          |      |   individually
 TIMER1 --->|      |
          |      |
 SERIAL --->|      |
          +------+
```

Mnemonic: "ETTES: External-Timer-Timer-External-Serial"

Question 3(b) [4 marks]

Draw Pin diagram of 8051 microcontroller.

Answer:

```
        8051 Microcontroller
        +------------------+
   P1.0--| 1            40 |--VCC
   P1.1--| 2            39 |--P0.0/AD0
   P1.2--| 3            38 |--P0.1/AD1
   P1.3--| 4            37 |--P0.2/AD2
   P1.4--| 5            36 |--P0.3/AD3
   P1.5--| 6            35 |--P0.4/AD4
   P1.6--| 7            34 |--P0.5/AD5
   P1.7--| 8            33 |--P0.6/AD6
   RST --| 9            32 |--P0.7/AD7
 P3.0/RXD| 10           31 |--EA/VPP
 P3.1/TXD| 11           30 |--ALE/PROG
P3.2/INT0| 12           29 |--PSEN
P3.3/INT1| 13           28 |--P2.7/A15
 P3.4/T0-| 14           27 |--P2.6/A14
 P3.5/T1-| 15           26 |--P2.5/A13
 P3.6/WR-| 16           25 |--P2.4/A12
 P3.7/RD-| 17           24 |--P2.3/A11
 XTAL2 --| 18           23 |--P2.2/A10
 XTAL1 --| 19           22 |--P2.1/A9
   VSS --| 20           21 |--P2.0/A8
        +------------------+
```

Pin Group	Function
P0	Port 0, multiplexed with address/data
P1	Port 1, general purpose I/O
P2	Port 2, upper address and I/O
P3	Port 3, special functions and I/O
XTAL	Crystal oscillator connections
Control	RST, EA, ALE, PSEN

Mnemonic: "PORT 0123: Data-General-Address-Special"

Question 3(c) [7 marks]

Explain Internal RAM Organization of 8051 microcontroller.

Answer:

RAM Area	Address Range	Usage
Register Banks	00H-1FH	R0-R7 (4 banks)
Bit-addressable	20H-2FH	128 bits (0-7FH)
Scratch Pad	30H-7FH	General purpose
SFRs	80H-FFH	Control registers

Diagram:

```
8051 Internal RAM (128 bytes):
+---------------+ 00H
| Register Bank 0|
+---------------+ 08H
| Register Bank 1|
+---------------+ 10H
| Register Bank 2|
+---------------+ 18H
| Register Bank 3|
+---------------+ 20H
| Bit-addressable|
+---------------+ 30H
|               |
| Scratch Pad   |
|               |
+---------------+ 80H

Special Function Registers:
+---------------+ 80H
|               |
| SFRs          |
| (not all bytes |
| are used)     |
|               |
+---------------+ FFH
```

- **Register Banks**: Four banks of 8 registers (R0-R7) selectable by PSW

- **Bit-addressable**: 16 bytes (128 bits) individually addressable as bits

- **General Purpose**: User variables and stack space

- **SFRs**: Control and status registers at higher addresses

Mnemonic: "RBBS: Registers Bits Buffer Special"

Question 3(a OR) [3 marks]

List SFRs with their addresses.

Answer:

SFR	Address	Function
P0	80H	Port 0
SP	81H	Stack Pointer
DPL	82H	Data Pointer Low
DPH	83H	Data Pointer High
PCON	87H	Power Control
TCON	88H	Timer Control
TMOD	89H	Timer Mode
P1	90H	Port 1
SCON	98H	Serial Control
P2	A0H	Port 2
IE	A8H	Interrupt Enable
P3	B0H	Port 3
IP	B8H	Interrupt Priority
PSW	D0H	Program Status Word
ACC	E0H	Accumulator
B	F0H	B Register

Mnemonic: "PDPT-SP: Ports-Data-Program-Timers-Serial-Prioritized"

Question 3(b OR) [4 marks]

Explain Timers/Counters logic diagram of 8051 microcontroller.

Answer:

Timer/Counter Diagram:

```
        +-----------+
TLx ---->|  8-bit    |       +------------+
        |  Register |------>|  8-bit      |
        +-----------+       |  Register   |---> Interrupt
                            |  (THx)      |
        +-----------+       +------------+
TRx ---->| Control   |            ^
        | Logic     |            |
        +-----------+            |
             ^                   |
```

```
                |                    |
                v                    v
        +--------------------+
C/T ---->| Mode Control Logic  |<----- GATE
        +--------------------+
                    ^
                    |
INTx -------------->
```
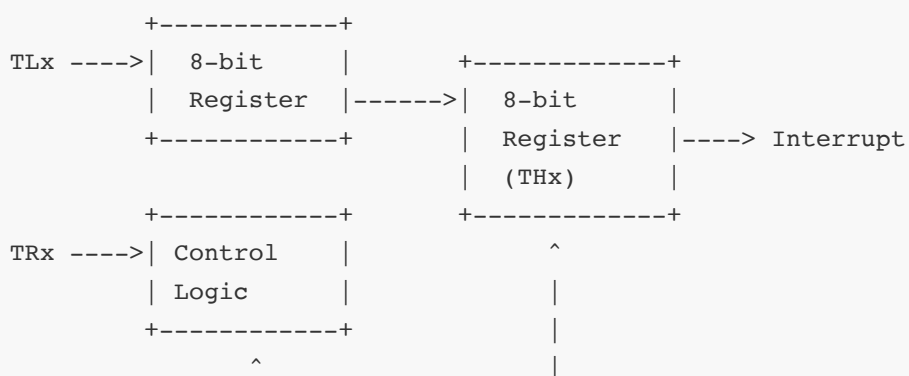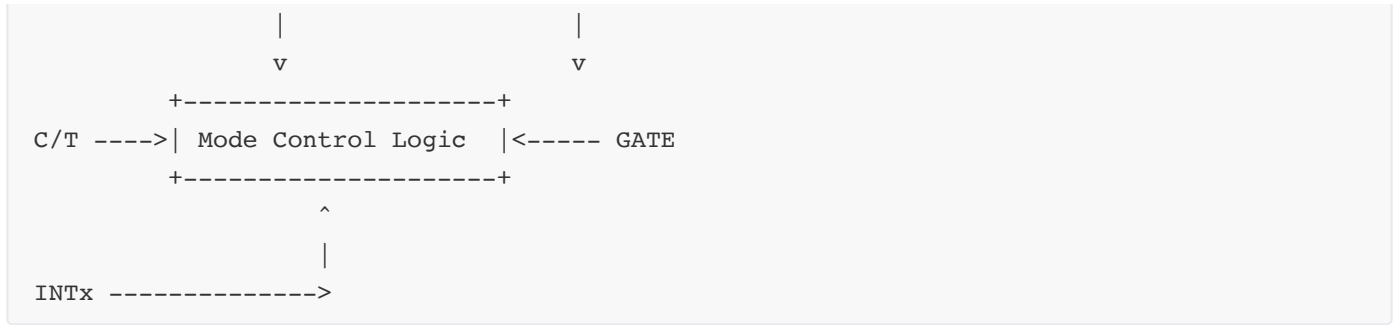
Component	Function
TLx, THx	Timer low and high byte registers
C/T	Selects Timer (0) or Counter (1) mode
GATE	External enable control
TRx	Timer run control bit
Mode Control	Selects one of four operation modes

- **Timer**: Uses internal clock, counts machine cycles
- **Counter**: Counts external events on T0/T1 pins
- **Control Bits**: Set in TMOD and TCON registers
- **Modes**: Different timer configurations (13/16/8-bit)

Mnemonic: "TCG: Timer-Counter-Gate"

Question 3(c OR) [7 marks]

Explain block diagram of 8051 microcontroller.

Answer:

Component	Function
CPU	8-bit processor with ALU
Memory	4K ROM, 128 bytes RAM
I/O Ports	Four 8-bit ports (P0-P3)
Timers	Two 16-bit timers/counters
Serial Port	Full-duplex UART
Interrupts	Five interrupt sources
Special Function Registers	Control registers

Diagram:

```
+---------------------------------------+
|                8051 MCU               |
| +-------------+       +-------------+  |
| |             |       |             | |
| |    CPU      |<------>| Program ROM |  |
| |             |       | (4K bytes)  | |
| +-------------+       +-------------+  |
|       ^                               |
|       |               +-------------+  |
|       |               |             | |
|       +-------------->| Internal RAM|  |
|       |               | (128 bytes) | |
|       v               +-------------+  |
| +-------------+       +-------------+  |
| |             |       |             | |
| |    SFRs     |<------>| I/O Ports   |  |
| |             |       | (P0,P1,P2,P3)| |
| +-------------+       +-------------+  |
|                                       |
| +-------------+       +-------------+  |
| |             |       |             | |
| | Timers/     |       | Serial Port |  |
| | Counters    |       | (UART)      | |
| +-------------+       +-------------+  |
+---------------------------------------+
```
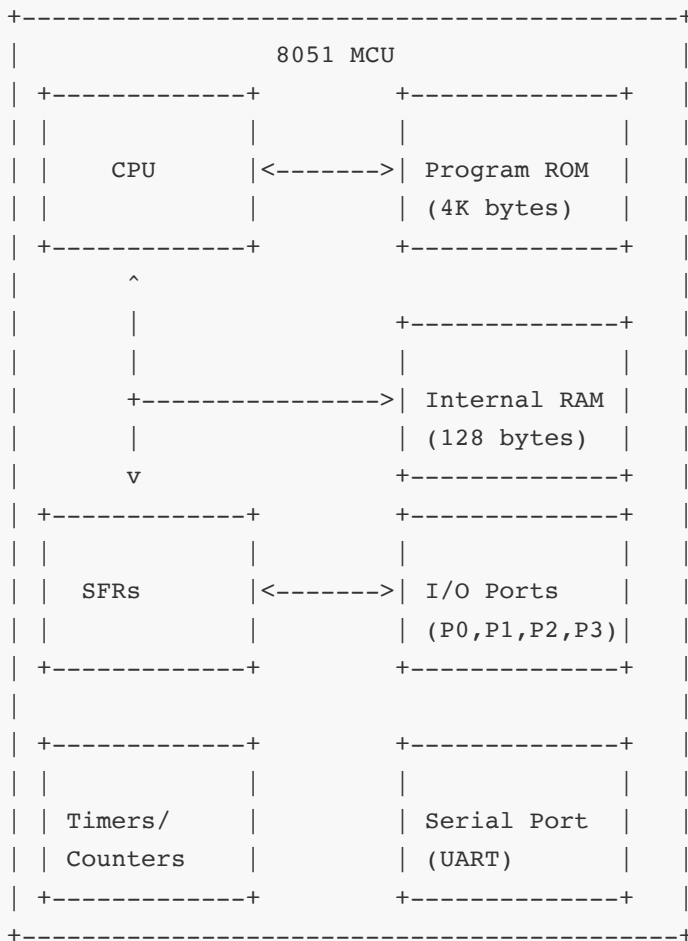
- **Harvard Architecture**: Separate program and data memory
- **CISC Design**: Rich instruction set (over 100 instructions)
- **In-built Peripherals**: No need for external components
- **Single-chip Solution**: Complete system on one chip

Mnemonic: "CAPITALS: CPU Architecture Ports I/O Timer ALU LS-Interface Serial"

Question 4(a) [3 marks]

Write an 8051 Assembly Language Program to add two bytes of data and store result in R4 register.

Answer:

```
    MOV  A, #25H    ; Load first value (25H) into accumulator
    MOV  R3, #18H   ; Load second value (18H) into R3
    ADD  A, R3      ; Add R3 to accumulator
    MOV  R4, A      ; Store result in R4 register
```

Key Steps:

- Load first operand into Accumulator
- Load second operand into register R3
- Perform addition using ADD instruction

- Store result from Accumulator to R4

Mnemonic: "LLAS: Load-Load-Add-Store"

Question 4(b) [4 marks]

Write an 8051 Assembly Language Program to OR the contents of Port-1 and Port-2 then put the result in external RAM location 0200H.

Answer:

```
    MOV  A, P1      ; Load contents of Port-1 into accumulator
    ORL  A, P2      ; OR Port-2 contents with accumulator
    MOV  DPTR, #0200H ; Load DPTR with external RAM address
    MOVX @DPTR, A    ; Store result in external RAM location 0200H
```

Key Steps:

- Read Port-1 into Accumulator

- Perform OR operation with Port-2

- Set up Data Pointer (DPTR) for external RAM

- Write result to external memory

Mnemonic: "PORT: Port-OR-Register-Transfer"

Question 4(c) [7 marks]

List Addressing Modes of 8051 Microcontroller and explain them with at least one example.

Answer:

Addressing Mode	Example	Description
Immediate	MOV A, #25H	Data is in instruction
Register	MOV A, R0	Data is in register
Direct	MOV A, 30H	Data is at RAM address
Indirect	MOV A, @R0	R0/R1 contains address
Indexed	MOVC A, @A+DPTR	Access program memory
Bit	SETB P1.3	Access individual bits
Relative	SJMP LABEL	Jumps with 8-bit offset

Examples:

- **Immediate**: `MOV A, #55H` (Load A with 55H)

- **Register**: `ADD A, R3` (Add R3 to A)

- **Direct**: `MOV 40H, A` (Store A at address 40H)

- **Indirect**: `MOV @R0, #5` (Store 5 at address in R0)

- **Indexed**: `MOVC A, @A+DPTR` (Read code memory)

- **Bit**: `CLR C` (Clear carry flag)

- **Relative**: `JZ LOOP` (Jump if A is zero)

Mnemonic: "I'M DIRBI: Immediate Register Direct Bit Indexed"

Question 4(a OR) [3 marks]

Explain following instructions: (i) DJNZ (ii) POP (iii) CJNE.

Answer:

Instruction	Syntax	Operation
DJNZ	DJNZ Rn, rel	Decrement register, Jump if Not Zero
POP	POP direct	Pop data from stack to direct address
CJNE	CJNE A, #data, rel	Compare and Jump if Not Equal

Examples and Explanation:

- **DJNZ R7, LOOP**: Decrements R7 and jumps to LOOP if R7≠0
    - Used for loop counters and delays
- **POP 30H**: Copies data from stack to address 30H
    - Increments SP after data retrieval
- **CJNE A, #25H, NOTEQUAL**: Compares A with 25H, jumps if not equal
    - Also sets Carry flag if A < data

Mnemonic: "DPC: Decrement-Pop-Compare"

Question 4(b OR) [4 marks]

For 8051 Microcontroller with a crystal frequency of 12 MHz, generate a delay of 4ms.

Answer:

```
; Crystal frequency = 12 MHz
; Machine cycle = 1 µs
; Required delay = 4 ms = 4000 machine cycles


    MOV  R7, #16    ; Outer loop counter (16 x 250 = 4000)
DELAY1: MOV  R6, #250  ; Inner loop counter
DELAY2: NOP            ; 1 machine cycle
    NOP            ; 1 machine cycle
    DJNZ R6, DELAY2 ; 2 machine cycles (250 x 4 = 1000 cycles)
    DJNZ R7, DELAY1 ; 16 x 250 = 4000 cycles total
    RET            ; Return from subroutine
```

Calculation:

- Each inner loop: 4 cycles × 250 iterations = 1000

- Outer loop: 16 iterations × 1000 cycles = 16,000 cycles

- At 12MHz, 1 machine cycle = 1µs

- Total delay = 4ms (4000 cycles)

Mnemonic: "LNDD: Load-NOP-Decrement-Decrement"

Question 4(c OR) [7 marks]

Explain any seven Logical instructions with example for 8051 Microcontroller.

Answer:

Instruction	Example	Operation
ANL	ANL A, #3FH	Logical AND
ORL	ORL P1, #80H	Logical OR
XRL	XRL A, R0	Logical XOR
CLR	CLR A	Clear (set to 0)
CPL	CPL P1.0	Complement (invert)
RL	RL A	Rotate left
RR	RR A	Rotate right

Examples with Explanation:

1. **ANL A, #0FH**: Masks high nibble (A = A AND 0FH)

   - Before: A = 95H, After: A = 05H

2. **ORL 20H, A**: Sets bits in memory (20H = 20H OR A)

   - Before: 20H = 55H, A = 0AH, After: 20H = 5FH

3. **XRL A, #55H**: Toggles specific bits (A = A XOR 55H)

   ○ Before: A = AAH, After: A = FFH

4. **CLR C**: Clears carry flag (C = 0)

   ○ Used before subtraction operations

5. **CPL A**: Inverts all bits (A = NOT A)

   ○ Before: A = 55H, After: A = AAH

6. **RL A**: Rotates accumulator left one bit

   ○ Before: A = 85H (10000101), After: A = 0BH (00001011)

7. **RR A**: Rotates accumulator right one bit

   ○ Before: A = 85H (10000101), After: A = C2H (11000010)

Mnemonic: "A-OX-CCR: AND OR XOR Clear Complement Rotate"

Question 5(a) [3 marks]

List Applications of microcontroller in various fields.

Answer:

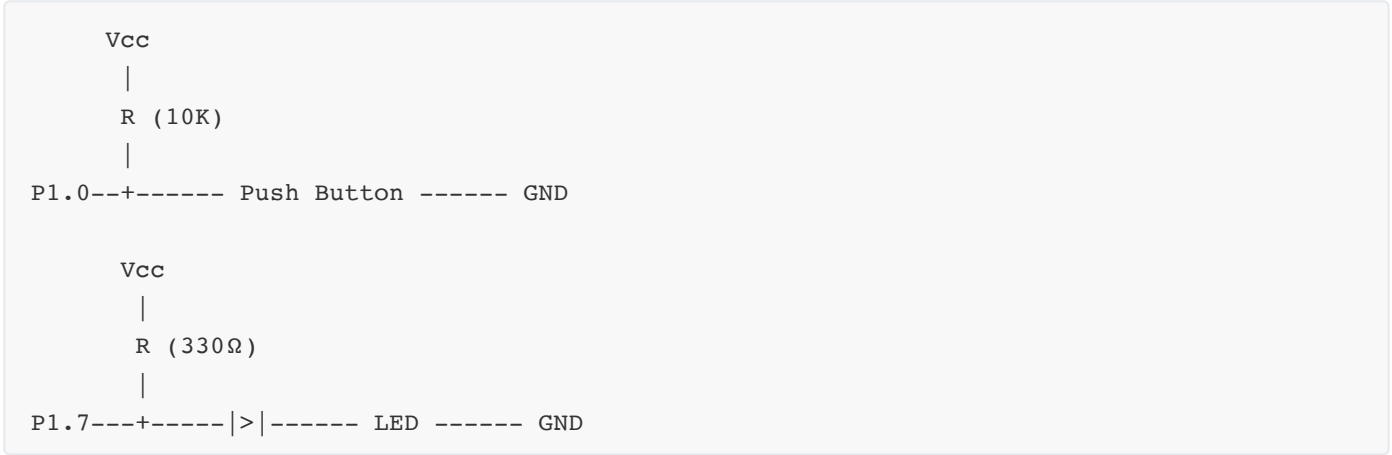| Field	Applications
Industrial	Motor control, automation, PLCs
Medical	Patient monitoring, diagnostic equipment
Consumer	Washing machines, microwaves, toys
Automotive	Engine control, ABS, airbag systems
Communication	Mobile phones, modems, routers
Security	Access control, alarm systems

Mnemonic: "I-MACS: Industrial-Medical-Automotive-Consumer-Security"

Question 5(b) [4 marks]

Interface Push button Switch and LED with 8051 microcontroller.

Answer:

Circuit Diagram:

```
     Vcc
      |
     R (10K)
      |
P1.0--+------ Push Button ------ GND


      Vcc
       |
      R (330Ω)
       |
P1.7---+-----|>|------ LED ------ GND
```

Program:

```
AGAIN:  JB    P1.0, LED_OFF  ; If P1.0=1 (not pressed), LED off
        SETB  P1.7           ; If P1.0=0 (pressed), LED on
        SJMP  AGAIN          ; Repeat
LED_OFF:CLR   P1.7           ; Turn LED off
        SJMP  AGAIN          ; Repeat
```

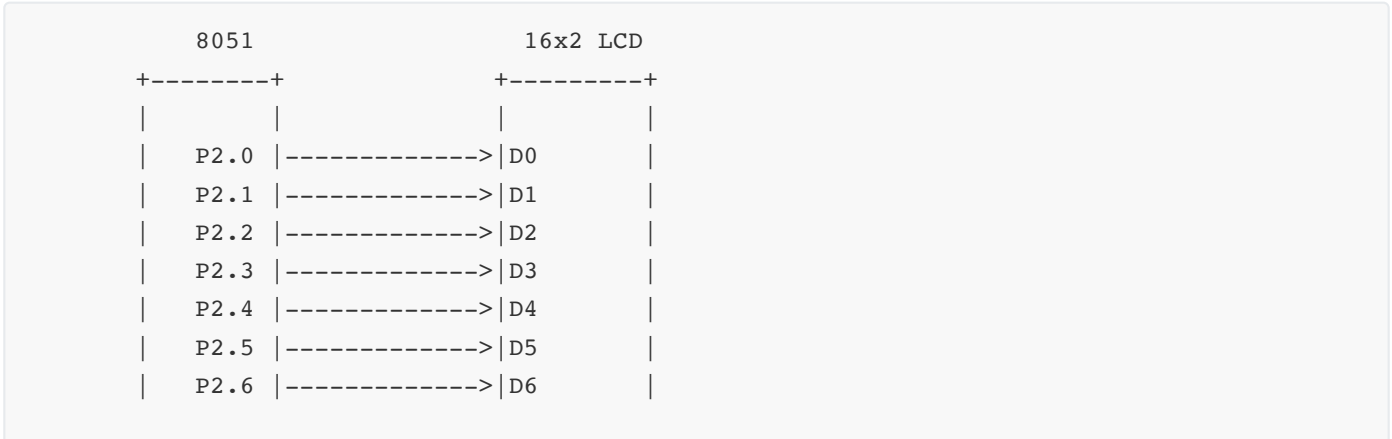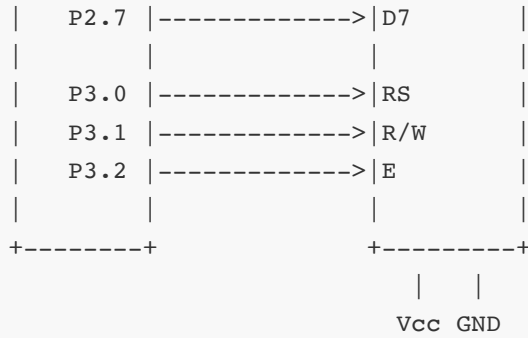| Component	Connection	Purpose
Push Button	P1.0 (input)	User input, active-low
Pull-up Resistor	10K to Vcc	Prevents floating input
LED	P1.7 (output)	Visual indicator
Current-limiting Resistor	330Ω	Protects LED

Mnemonic: "PLIC: Push-LED-Input-Control"

Question 5(c) [7 marks]

Interface LCD with microcontroller and write a program to display "HELLO".

Answer:

Circuit Diagram:

```
        8051                  16x2 LCD
    +--------+              +---------+
    |        |              |         |
    |  P2.0  |------------->|D0       |
    |  P2.1  |------------->|D1       |
    |  P2.2  |------------->|D2       |
    |  P2.3  |------------->|D3       |
    |  P2.4  |------------->|D4       |
    |  P2.5  |------------->|D5       |
    |  P2.6  |------------->|D6       |
```

```
    |   P2.7 |------------->|D7         |
    |        |              |           |
    |   P3.0 |------------->|RS         |
    |   P3.1 |------------->|R/W        |
    |   P3.2 |------------->|E          |
    |        |              |           |
    +--------+              +---------+
                             |   |
                            Vcc GND
```

Program:

```
ORG 0000H                ; Start address

; Initialize LCD
    MOV A, #38H          ; 8-bit, 2 lines, 5x7 dots
    ACALL COMMAND        ; Send command
    MOV A, #0EH          ; Display ON, cursor ON
    ACALL COMMAND        ; Send command
    MOV A, #01H          ; Clear display
    ACALL COMMAND        ; Send command
    MOV A, #06H          ; Increment cursor
    ACALL COMMAND        ; Send command
    MOV A, #80H          ; First line, first position
    ACALL COMMAND        ; Send command

; Display "HELLO"
    MOV A, #'H'          ; Load 'H'
    ACALL DISPLAY        ; Display it
    MOV A, #'E'          ; Load 'E'
    ACALL DISPLAY        ; Display it
    MOV A, #'L'          ; Load 'L'
    ACALL DISPLAY        ; Display it
    MOV A, #'L'          ; Load 'L'
    ACALL DISPLAY        ; Display it
    MOV A, #'O'          ; Load 'O'
    ACALL DISPLAY        ; Display it

    SJMP $               ; Stay here

; Command subroutine
COMMAND:
    MOV P2, A            ; Put command on data bus
    CLR P3.0             ; RS=0 for command
    CLR P3.1             ; R/W=0 for write
    SETB P3.2            ; E=1
    ACALL DELAY          ; Wait
    CLR P3.2             ; E=0
    RET                  ; Return

; Display subroutine
```

```
DISPLAY:
    MOV P2, A            ; Put data on data bus
    SETB P3.0            ; RS=1 for data
    CLR P3.1             ; R/W=0 for write
    SETB P3.2            ; E=1
    ACALL DELAY          ; Wait
    CLR P3.2             ; E=0
    RET                  ; Return


; Delay subroutine
DELAY:
    MOV R7, #50          ; Load counter
DELAY_LOOP:
    DJNZ R7, DELAY_LOOP  ; Loop until R7=0
    RET                  ; Return


END                      ; End of program
```

Component	Connection	Purpose
Data Pins	P2.0-P2.7	Transfer data/commands
RS (Register Select)	P3.0	Select command (0) or data (1)
R/W (Read/Write)	P3.1	Select write (0) or read (1)
E (Enable)	P3.2	Latch data on falling edge

- **Initialization**: Configure LCD for 8-bit, 2 lines, cursor options
- **Data Transfer**: Commands sent with RS=0, characters with RS=1
- **Characters**: ASCII values sent one by one to display text
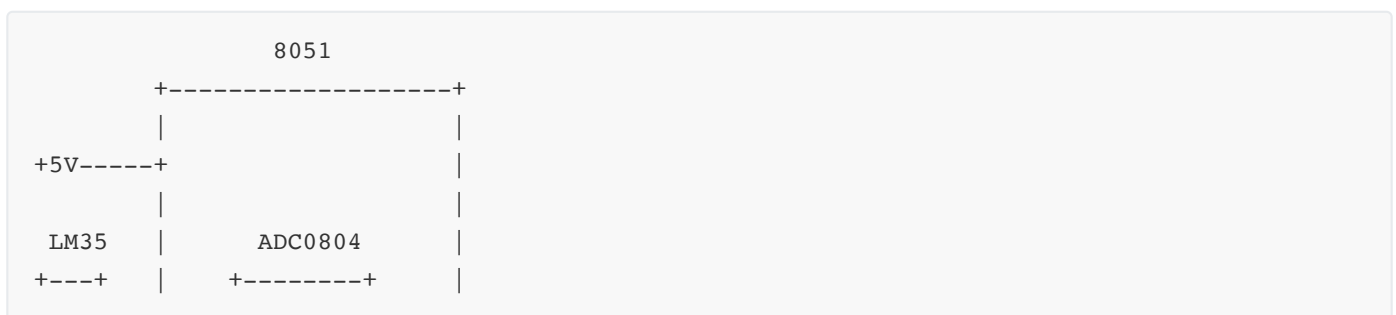- **Timing**: Delay routine ensures proper signal timing

Mnemonic: "DICE: Data-Instruction-Control-Enable"

Question 5(a OR) [3 marks]

Draw Interfacing of LM35 with 8051 microcontroller.

Answer:

Circuit Diagram:

```
            8051
      +------------------+
      |                  |
+5V-----+                |
      |                  |
 LM35 |      ADC0804     |
+---+  |    +--------+   |
```

```
 |   |   |   |        |      |
|OUT+---+--->|Vin     |      |
 |   |   |   |        |      |
+---+   |   |        |      |
 |      |   |        |      |
GND-----+----+GND     |      |
        |   |        |      |
        |   |   CS  +<----+P1.0
        |   |   RD  +<----+P1.1
        |   |   WR  +<----+P1.2
        |   |   INT +---->+P1.3
        |   |        |      |
        |   | D0-D7  +<--->+P0.0-P0.7
        |   +--------+      |
        |               |
       +-----------------+
```
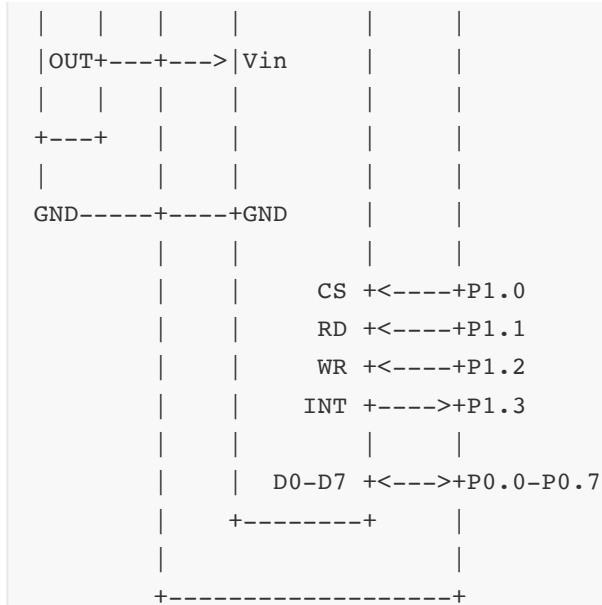
Component	Function
LM35	Temperature sensor (10mV/°C)
ADC0804	Analog-to-Digital Converter
8051	Microcontroller to read temperature

Key Points:

- LM35 produces analog voltage proportional to temperature
- ADC0804 converts analog voltage to digital value
- 8051 controls ADC and reads temperature data
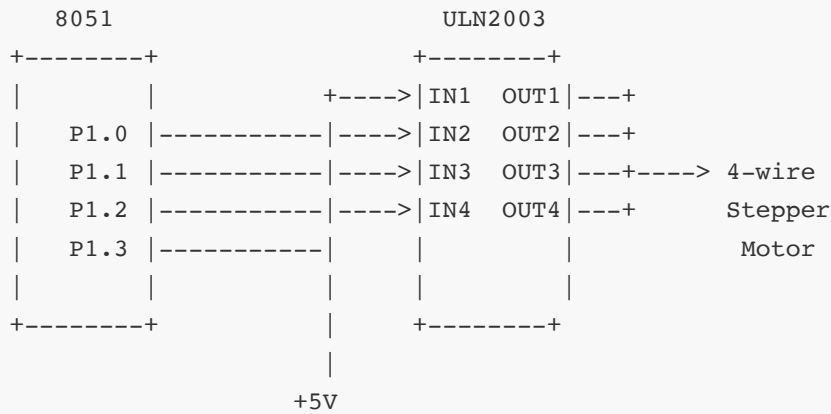- Resolution: 10mV/°C → ~0.2°C resolution with 8-bit ADC

Mnemonic: "TAC: Temperature-Analog-Convert"

Question 5(b OR) [4 marks]

Interface Stepper motor with 8051 microcontroller.

Answer:

Circuit Diagram:

```
      8051                          ULN2003
    +--------+                    +--------+
    |        |          +---->|IN1  OUT1|---+
    |  P1.0  |----------|---->|IN2  OUT2|---+
    |  P1.1  |----------|---->|IN3  OUT3|---+----> 4-wire
    |  P1.2  |----------|---->|IN4  OUT4|---+       Stepper
    |  P1.3  |----------|     |    |         |         Motor
    |        |          |     |    |         |
    +--------+          |     +--------+
                        |
              +5V
```

Program:

```
      ORG 0000H

; Stepper Motor Sequence for Clockwise Rotation
SEQ:  DB 00001000B  ; Step 1
      DB 00001100B  ; Step 2
      DB 00000100B  ; Step 3
      DB 00000110B  ; Step 4
      DB 00000010B  ; Step 5
      DB 00000011B  ; Step 6
      DB 00000001B  ; Step 7
      DB 00001001B  ; Step 8

MAIN: MOV R0, #00H  ; Initialize sequence pointer

STEP: MOV A, R0     ; Get current sequence number
      ANL A, #07H   ; Keep within 0-7 range (8 steps)
      MOV DPTR, #SEQ ; Point to sequence table
      MOVC A, @A+DPTR ; Get sequence pattern
      MOV P1, A     ; Output to stepper motor

      ACALL DELAY   ; Wait between steps

      INC R0        ; Next sequence
      SJMP STEP     ; Repeat

DELAY: MOV R6, #250 ; Delay loop
LOOP:  MOV R7, #250
LOOP2: DJNZ R7, LOOP2
      DJNZ R6, LOOP
      RET

      END
```

Component	Purpose
ULN2003	Driver IC with Darlington arrays
Port pins	P1.0-P1.3 for 4 motor phases
Power supply	Separate supply for motor

Key Points:

- Stepper motor requires specific sequence of pulses for rotation
- ULN2003 provides current amplification for motor coils
- 8-step sequence provides smoother rotation
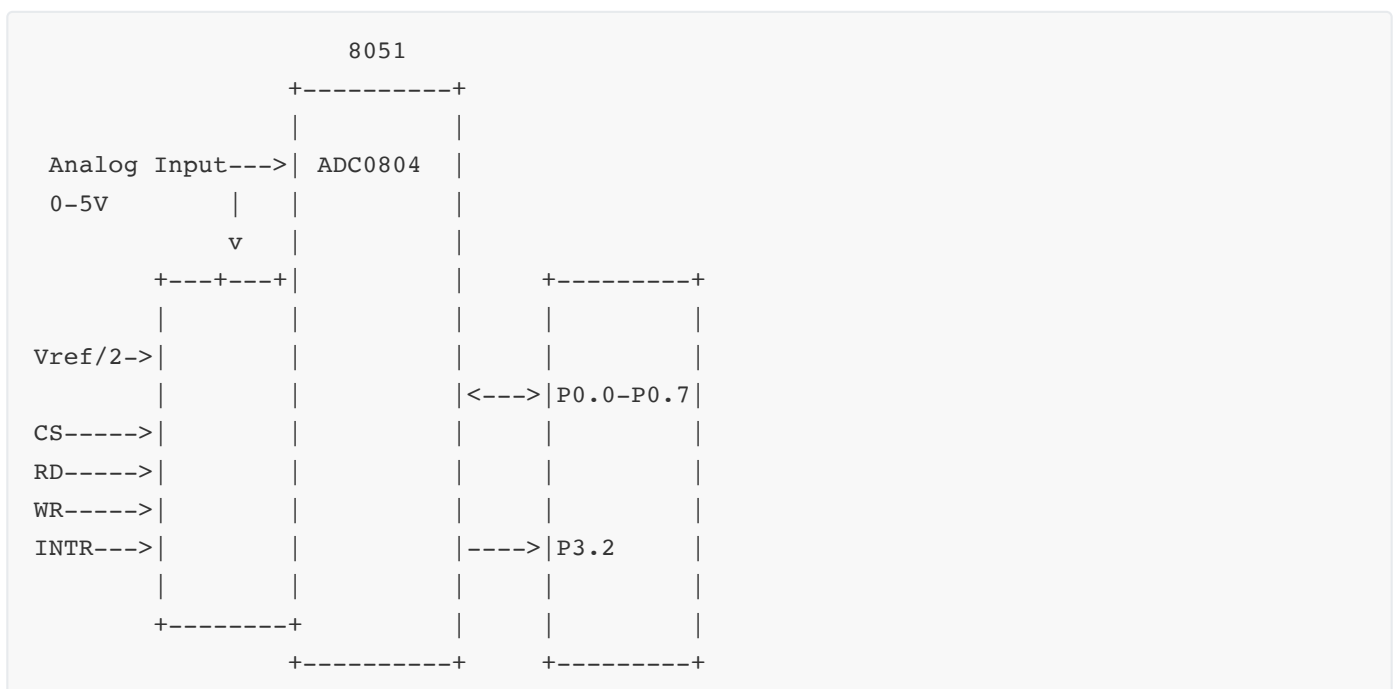- Delay between steps controls rotation speed

Mnemonic: "PDCS: Port-Driver-Current-Sequence"

Question 5(c OR) [7 marks]

Interface ADC0804 with 8051 microcontroller.

Answer:

Circuit Diagram:

```
                      8051
                  +----------+
                  |          |
  Analog Input--->| ADC0804  |
   0-5V       |   |          |
            v  |          |
       +---+---+|          |        +---------+
       |       |          |        |         |
 Vref/2->|      |          |        |         |
       |       |         |<--->|P0.0-P0.7|
 CS----->|      |          |        |         |
 RD----->|      |          |        |         |
 WR----->|      |          |        |         |
 INTR--->|      |         |---->|P3.2     |
       |       |          |        |         |
       +--------+         |        |         |
                +----------+     +---------+
```

Program:

```
ORG 0000H

START:  CLR P1.0      ; CS = 0 (Chip Select active)
        CLR P1.1      ; RD = 0
        CLR P1.2      ; WR = 0
```

```
        SETB P1.2    ; WR = 1 (Start conversion)

WAIT:   JB P1.3, WAIT ; Wait for conversion (INTR = 0)

        CLR P1.1     ; RD = 0 (Read data)
        MOV A, P0    ; Read converted data into A
        MOV 30H, A   ; Store result in RAM

        SETB P1.0    ; CS = 1 (Chip deselect)

PROCESS:
        ; Process the data (e.g., display, compare, etc.)
        ; ...

        ACALL DELAY  ; Wait before next conversion
        SJMP START   ; Repeat

DELAY:  MOV R7, #200  ; Delay routine
DLOOP:  DJNZ R7, DLOOP
        RET

END
```

Connection	8051 Pin	ADC0804 Pin
Data Bus	P0.0-P0.7	D0-D7
CS	P1.0	CS
RD	P1.1	RD
WR	P1.2	WR
INTR	P1.3	INTR

ADC0804 Features:

- 8-bit resolution (256 steps)
- 0-5V input range
- Single-channel operation
- ~100μs conversion time
- Interface protocol:
    1. Activate CS, pulse WR to start conversion
    2. Wait for INTR to go low (conversion complete)
    3. Activate RD to read data
    4. Deactivate CS when done

Mnemonic: "CRIW: Control-Read-Interrupt-Write"