

# Subject Name (Gujarati)

1333202 -- Winter 2023

Semester 1 Study Material

Detailed Solutions and Explanations

## પ્રશ્ન 1(અ) [3 ગુણ]

માઇક્રોપ્રોસેસર ની વ્યાખ્યા આપો.

### જવાબ

માઇક્રોપ્રોસેસર એ એક સિંગલ ચિપ CPU છે જેમાં digital computer ના central processing unit ના કાર્યો કરવા માટે જરૂરી બધા arithmetic, logic અને control circuits હોય છે.

Table 1: માઇક્રોપ્રોસેસર ની મુખ્ય વિશેષતાઓ

વિશેષતા	વર્ણન
<b>Single Chip Processing Unit Control Logic</b>	એક integrated circuit પર સંપૂર્ણ CPU instructions execute કરે છે અને calculations કરે છે system operations અને data flow ને manage કરે છે

- **Central Processing Unit:** મુખ્ય component જે instructions execute કરે છે
- **Integrated Circuit:** બધા functions એક જ silicon chip પર combined
- **Programmable Device:** stored instructions આધારે વિવિધ programs execute કરી શકે છે

### મેમરી ટ્રીક

“Single Chip CPU = Smart Computer Processor Unit”

## પ્રશ્ન 1(બ) [4 ગુણ]

માઇક્રોપ્રોસેસર ના ફ્લેગ રજિસ્ટર ને સમજાવો.

### જવાબ

Flag register માં ALU દ્વારા કરવામાં આવેલા arithmetic અને logical operations ના result વિશે status information store થાય છે.

Table 2: 8085 Flag Register Bits

Flag	Position	હેતુ
<b>S (Sign)</b>	Bit 7	Result નું sign દર્શાવે છે (1=negative, 0=positive)
<b>Z (Zero)</b>	Bit 6	Result zero હોય ત્યારે set થાય છે
<b>AC (Auxiliary Carry)</b>	Bit 4	Bit 3 થી bit 4 માં carry
<b>P (Parity)</b>	Bit 2	Even parity flag
<b>CY (Carry)</b>	Bit 0	MSB માંથી carry

- **Status Indicator:** છેલ્લા operation result ની condition બતાવે છે
- **Conditional Instructions:** Branching અને decision making માટે ઉપયોગ થાય છે
- **5 Active Flags:** Sign, Zero, Auxiliary Carry, Parity અને Carry flags

### મેમરી ટ્રીક

“Flags Show Zero, Sign, Parity, Auxiliary, Carry”

### પ્રશ્ન 1(ક) [7 ગુણ]

માઇક્રોપ્રોસેસર નું instruction format ઉદાહરણ સાથે સમજાવો.

જવાબ

Microprocessor instructions માં opcode અને operand fields હોય છે જે operation અને data locations specify કરે છે.

Table 3: 8085 Instruction Format Types

Format	Size	Structure	Example
1-Byte	8 bits	Opcode only	MOV A,B
2-Byte	16 bits	Opcode + 8-bit data	MVI A,05H
3-Byte	24 bits	Opcode + 16-bit address	LDA 2000H

ડાયાગ્રામ:

Mermaid Diagram (Code)

```
{Shaded}
{Highlighting}[]
graph LR
    A[Instruction Format] --{-}{-}{-} B[Opcode Field]}
    A --{-}{-}{-} C[Operand Field]}
    B --{-}{-}{-} D[Operation Code{}br/{}Operation Specify    ]}
    C --{-}{-}{-} E[Data/Address{}br/{}Source/Destination Specify    ]}
{Highlighting}
{Shaded}
```

- **Opcode Field:** કયું operation કરવું છે તે define કરે છે (ADD, MOV, JMP)
- **Operand Field:** Data, register અથવા memory address information હોય છે
- **Variable Length:** Instructions 1, 2 અથવા 3 bytes ની હોઈ શકે છે
- **Addressing Modes:** Operand location specify કરવાની વિવિધ રીતો

મેમરી ટ્રીક

“Opcode Operations + Operand Objects = Complete Commands”

### પ્રશ્ન 1(ક OR) [7 ગુણ]

માઇક્રોપ્રોસેસરમાં ALU, Control Unit અને CPU સમજાવો.

જવાબ

CPU માં ત્રણ મુખ્ય functional units છે જે instructions execute કરવા માટે સાથે મળીને કામ કરે છે.

Table 4: CPU Components અને Functions

Component	Primary Function	Key Operations
ALU	Arithmetic & Logic Operations	ADD, SUB, AND, OR, XOR
Control Unit	Instruction Control	Fetch, Decode, Execute
CPU	Overall Processing	બધા operations coordinate કરે છે

## સાચાગ્રામ:

### Mermaid Diagram (Code)

```
{Shaded}
{Highlighting}[]
graph TD
    A[CPU] --{} B[ALU{}br/{}Arithmetic Logic Unit]
    A --{} C[Control Unit]
    A --{} D[Register Array]
    B --{} E[Math Operations{}br/{}Logic Operations]
    C --{} F[Instruction Control{}br/{}Signal Generation]
{Highlighting}
{Shaded}
```

- **ALU Functions:** બધા arithmetic calculations અને logical operations કરે છે
- **Control Unit Tasks:** Instruction execution cycle manage કરે છે અને control signals generate કરે છે
- **CPU Coordination:** Complete processing માટે ALU અને Control Unit ને integrate કરે છે

## મેમરી ટ્રીક

## “ALU Adds, Control Commands, CPU Coordinates”

**પ્રશ્ન 2(અ) [3 ગુણ]**

**ALE signal નું કાર્ય સમજાવો.**

**જલ્દી**

ALE (Address Latch Enable) signal નો ઉપયોગ lower-order address અને data lines ને demultiplex કરવા માટે થાય છે.

### Table 5: ALE Signal Functions

Function	વર્ણન
<b>Address Latching</b>	Lower 8-bit address capture કરે છે
<b>Demultiplexing</b>	Address ને data થી separate કરે છે
<b>Timing Control</b>	Timing reference પ્રદાન કરે છે

## સાચાગ્રામ:

```
+{-}{-}{-}{-}{-}{-}{-}{-}{-}{-}+      ALE      +{-}{-}{-}{-}{-}{-}{-}{-}{-}{-}+
|   8085   |{-}{-}{-}{-}{-}{-}{-}{-}{-}{-}| Latch  |}
|           |               | 74373 |
| AD0{-7    |{-}{-}{-}{-}{-}{-}{-}{-}{-}{-}|         |}
+{-}{-}{-}{-}{-}{-}{-}{-}{-}{-}+          +{-}{-}{-}{-}{-}{-}{-}{-}{-}{-}+
                                           |
                                          AO{-A7 (Address)}
```

- **Active High Signal:** T1 state દરમિયાન ALE high જાય છે
- **External Latching:** Address hold કરવા માટે 74373 latch સાથે ઉપયોગ થાય છે
- **System Timing:** External devices માટે reference પ્રદાન કરે છે

## મેમરી ટ્રીક

## "ALE Always Latches External Addresses"

**પ્રશ્ન 2(બ) [4 ગુણ]**

**માઇક્રોપ્રોસેસર અને માઇક્રોકંટ્રોલર ની સરખામણી કરો**

Table 6: Microprocessor vs Microcontroller Comparison

Parameter	Microprocessor	Microcontroller
<b>Design</b>	General purpose	Application specific
<b>Memory</b>	External RAM/ROM	Internal RAM/ROM
<b>I/O Ports</b>	External interface	Built-in I/O ports
<b>Timers</b>	External	Built-in timers
<b>Cost</b>	વધુ system cost	ઓછો system cost
<b>Power</b>	વધુ consumption	ઓછો consumption

- **Integration Level:** Microcontroller માં વધુ integrated components હોય છે
- **Application Focus:** Microprocessor computing માટે, microcontroller control માટે
- **System Complexity:** Microprocessor ને વધુ external components જોઈએ છે
- **Design Flexibility:** Microprocessor વધુ expandability આપે છે

## મેમરી ટ્રીક

“Microprocessor = More Power, Microcontroller = More Control”

## પ્રશ્ન 2(ક) [7 ગુણ]

માઇક્રોપ્રોસેસરનો બ્લોક ડાયાગ્રામ દોરો અને સમજાવો.

8085 microprocessor માં કેટલાક functional blocks છે જે સાથે મળીને કામ કરે છે.

ડાયાગ્રામ:

graph TB

```

A[Accumulator] --- B[ALU]
C[Temp Register] --- B
B --- D[Flag Register]
E[Instruction Register] --- F[Instruction Decoder]
F --- G[Timing & Control Unit]
H[Program Counter] --- I[Address Buffer]
J[Stack Pointer] --- I
K[Register Array{B,C,D,E,H,L}]
I --- L[Address Bus A8-A15]
M[Address/Data Buffer] --- N[Address/Data Bus AD0-AD7]

```

Table 7: Block Functions

Block	Function
<b>ALU</b>	Arithmetic અને logical operations
<b>Register Array</b>	Temporary data storage (B,C,D,E,H,L)
<b>Control Unit</b>	Instruction execution control
<b>Address Buffer</b>	Address bus lines drive કરે છે

- **Data Path:** Internal bus દ્વારા registers વચ્ચે information flow થાય છે
- **Control Signals:** Timing અને control unit દ્વારા generate થાય છે
- **Bus Interface:** External memory અને I/O devices સાથે connect કરે છે
- **Register Operations:** Operands અને results માટે temporary storage

## મેમરી ટ્રીક

“Blocks Build Better Processing Systems”

પ્રશ્ન 2(અ OR) [3 ગુણ]

માઇક્રોપ્રોસેસરના 16 bits registers સમજાવો.

જવાબ

8085 માં 8-bit register pairs ને combine કરીને બનેલા ત્રણ 16-bit registers છે.

Table 8: 16-bit Registers

Register	Formation	Purpose
PC	Single 16-bit	Program Counter - next instruction address
SP	Single 16-bit	Stack Pointer - stack ની top નું address
HL	H + L registers	Memory pointer - data address

- **Program Counter:** આપમેળે next instruction પર increment થાય છે
- **Stack Pointer:** Stack પર last pushed data તરફ point કરે છે
- **HL Pair:** Memory addressing માટે સૌથી વધુ વપરાતું

મેમરી ટ્રીક

“PC Points Program, SP Stacks Properly, HL Holds Location”

પ્રશ્ન 2(બ OR) [4 ગુણ]

માઇક્રોપ્રોસેસર માં lower order address અને data lines ને de-multiplexing કરવાનું સમજાવો.

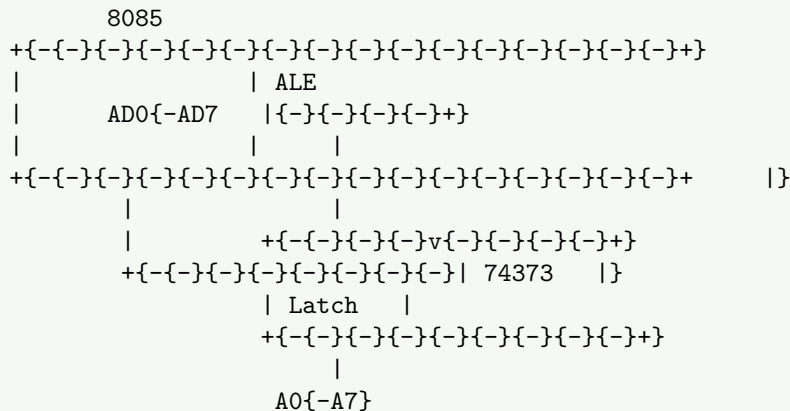
જવાબ

8085 pin count ઘટાડવા માટે lower 8-bit address ને data lines સાથે multiplex કરે છે.

Table 9: Multiplexed Lines

Lines	T1 State	T2-T4 States
AD0-AD7	Lower Address A0-A7	Data D0-D7
ALE Signal	High	Low

ડાયાગ્રામ:



- **Time Division:** સમાન lines પહેલા address પછી data carry કરે છે
- **External Latch:** ALE high હોય ત્યારે 74373 address capture કરે છે
- **Signal Separation:** અલગ address અને data buses બનાવે છે

મેમરી ટ્રીક

“ALE Always Latches External Address Elegantly”



```

      |      |
    30pF  30pF
      |      |
      GND   GND
  
```

Reset Circuit:

```

    +5V
      |
    10K
      |
RST  {-{-}+{-}{-}{-}||{-}{-}{-}GND}
      10pF
  
```

Table 11: Circuit Components

Component	Value	Purpose
Crystal	11.0592 MHz	Clock generation
Capacitors	30pF દરેક	Crystal stabilization
Reset Resistor	10K $\Omega$	Reset માટે pull-up
Reset Capacitor	10 $\mu$ F	Power-on reset delay

- **Clock Frequency:** Serial communication માટે સામાન્ય રીતે 11.0592 MHz
- **Reset Duration:** ઓછામાં ઓછા 2 machine cycles માટે high હોવું જોઈએ
- **Power-on Reset:** Power apply થાય ત્યારે automatic reset

### મેમરી ટ્રીક

“Crystals Create Clock, Resistors Reset Reliably”

## પ્રશ્ન 3(બ) [4 ગુણ]

માઇક્રોકંટ્રોલર ની આંતરીક RAM સમજાવો.

### જવાબ

8051 માં વિવિધ sections માં organize થયેલા 256 bytes નો internal RAM છે.

Table 12: Internal RAM Organization

Address Range	Size	Purpose
00H-1FH	32 bytes	Register Banks (4 banks $\times$ 8 registers)
20H-2FH	16 bytes	Bit-addressable area
30H-7FH	80 bytes	General purpose RAM
80H-FFH	128 bytes	Special Function Registers (SFRs)

## ડાયાગ્રામ:

### Mermaid Diagram (Code)

```
{Shaded}
{Highlighting} []
graph TD
    A[Internal RAM 256 Bytes] --{} B[Bank 0{-}3{-}br/{}00H{-}1FH]
    A --{} C[Bit Addressable{}br/{}20H{-}2FH]
    A --{} D[General Purpose{}br/{}30H{-}7FH]
    A --{} E[SFRs{}br/{}80H{-}FFH]
{Highlighting}
{Shaded}
```

- **Register Banks:** દરેકમાં 8 registers (R0-R7) વાળા ચાર banks
- **Bit Addressing:** 20H-2FH area માં individual bits address કરી શકાય છે
- **Stack Area:** સામાન્ય રીતે general purpose RAM area માં હોય છે
- **Direct Access:** બધા locations direct addressing દ્વારા accessible છે

## મેમરી ટ્રીક

“RAM Registers, Bits, General, Special Functions”

## પ્રશ્ન 3(ક) [7 ગુણ]

8051 નો બ્લોક ડાયાગ્રામ સમજાવો.

## જવાબ

8051 microcontroller એક જ chip પર CPU, memory અને I/O integrate કરે છે.

## ડાયાગ્રામ:

```
graph TB
    A[CPU Core] --{} B[ALU]
    A --{} C[Accumulator]
    A --{} D[B Register]
    E[Program Memory{br/4KB ROM}] --{} F[Program Counter]
    G[Data Memory{br/256B RAM}] --{} H[Data Pointer DPTR]
    I[Timer/Counter] --{} J[Timer 0/1]
    K[Serial Port] --{} L[UART]
    M[Interrupt System] --{} N[5 Interrupt Sources]
    O[I/O Ports] --{} P[P0, P1, P2, P3]
    Q[Oscillator] --{} R[Clock Circuit]
```

Table 13: મુખ્ય Blocks

Block	Function
<b>CPU</b>	Instruction execution અને control
<b>Memory</b>	4KB ROM + 256B RAM
<b>Timers</b>	બે 16-bit timer/counters
<b>I/O Ports</b>	ચાર 8-bit bidirectional ports
<b>Serial Port</b>	Full-duplex UART
<b>Interrupts</b>	5-source interrupt system

- **Harvard Architecture:** અલગ program અને data memory spaces
- **Built-in Peripherals:** Timers, serial port, interrupts integrated
- **Expandable:** External memory અને I/O add કરી શકાય છે
- **Control Applications:** Embedded control tasks માટે optimized



### મેમરી ટ્રીક

“Complete Control Chip Contains CPU, Memory, I/O”

### પ્રશ્ન 3(અ OR) [3 ગુણ]

DPTR અને PC નું કાર્ય સમજાવો.

#### જવાબ

DPTR અને PC એ memory addressing માટે 8051 માં મહત્વપૂર્ણ 16-bit registers છે.

Table 14: DPTR અને PC Functions

Register	Full Form	Function
DPTR	Data Pointer	External data memory તરફ point કરે છે
PC	Program Counter	Next instruction address તરફ point કરે છે

- **DPTR Usage:** External RAM અને lookup tables access કરવા માટે
- **PC Function:** Instruction fetch પછી આપમેળે increment થાય છે
- **16-bit Addressing:** બંને 64KB memory space address કરી શકે છે

### મેમરી ટ્રીક

“DPTR Data Pointer, PC Program Counter”

### પ્રશ્ન 3(બ OR) [4 ગુણ]

માઇક્રોકંટ્રોલરમાં timer ના અલગ અલગ modes સમજાવો.

#### જવાબ

8051 માં ચાર અલગ operating modes સાથે બે timers છે.

Table 15: Timer Modes

Mode	Configuration	Purpose
Mode 0	13-bit timer	8048 સાથે compatible
Mode 1	16-bit timer	Maximum count capability
Mode 2	8-bit auto-reload	Constant time intervals
Mode 3	બે 8-bit timers	Timer 0 split operation

- **Mode Selection:** TMOD register bits દ્વારા control થાય છે
- **Timer 0/1:** બંને timers modes 0, 1, 2 support કરે છે
- **Mode 3 Special:** ફક્ત Timer 0 જ mode 3 માં operate કરી શકે છે
- **Applications:** Delays, baud rate generation, event counting

### મેમરી ટ્રીક

“Modes Make Timers Tremendously Versatile”

### પ્રશ્ન 3(ક OR) [7 ગુણ]

માઇક્રોકંટ્રોલર ની interrupts સમજાવો.

#### જવાબ

8051 માં external events handle કરવા માટે 5-source interrupt system છે.

Table 16: 8051 Interrupt Sources

Interrupt	Vector Address	Priority	Trigger
Reset	0000H	સૌથી વધુ	Power-on/External
External 0	0003H	વધુ	INT0 pin
Timer 0	000BH	મધ્યમ	Timer 0 overflow
External 1	0013H	મધ્યમ	INT1 pin
Timer 1	001BH	ઓછું	Timer 1 overflow
Serial	0023H	સૌથી ઓછું	Serial communication

સાચાશ્રામ:

#### Mermaid Diagram (Code)

```
{Shaded}
{Highlighting}[]
graph TD
    A[Interrupt System] --{-}{-} B[External INT0]}
    A --{-}{-}{ C[Timer 0 Overflow]}
    A --{-}{-}{ D[External INT1] }
    A --{-}{-}{ E[Timer 1 Overflow]}
    A --{-}{-}{ F[Serial Port]}
    G[Interrupt Control] --{-}{-}{ H[IE Register]}
    G --{-}{-}{ I[IP Register]}
{Highlighting}
{Shaded}
```

- **Interrupt Enable:** IE register individual interrupt enables control કરે છે
- **Priority Control:** IP register interrupt priorities set કરે છે
- **Vector Addresses:** દરેક interrupt નું fixed vector location છે
- **Nested Interrupts:** વધુ priority ઓછી priority ને interrupt કરી શકે છે

#### મેમરી ટ્રીક

“Five Interrupt Sources Serve System Efficiently”

#### પ્રશ્ન 4(અ) [3 ગુણ]

8051 ની data transfer instruction ઉદાહરણ આપી સમજાવો.

#### જવાબ

Data transfer instructions registers, memory અને I/O ports વચ્ચે data move કરે છે.

Table 17: Data Transfer Instructions

Instruction	Example	Function
MOV	MOV A,#55H	Immediate data ને accumulator માં move કરે છે
MOVX	MOVX A,@DPTR	External RAM ને accumulator માં move કરે છે
MOVC	MOVC A,@A+PC	Code memory ને accumulator માં move કરે છે

- **MOV Variants:** Register to register, immediate to register
- **External Access:** External RAM operations માટે MOVX
- **Code Access:** Program memory tables read કરવા માટે MOVC

#### મેમરી ટ્રીક

“MOV Moves data, MOVX eXternal, MOVC Code”

પ્રશ્ન 4(બ) [4 ગુણ]

માઇક્રોકંટ્રોલરના addressing modes નું list બનાવી સમજાવો.

જવાબ

8051 flexible data access માટે કેટલાક addressing modes support કરે છે.

Table 18: 8051 Addressing Modes

Mode	Example	વર્ણન
<b>Immediate</b>	MOV A,#55H	Instruction માં data specify કર્યો છે
<b>Register</b>	MOV A,R0	Register contents ઉપયોગ કરે છે
<b>Direct</b>	MOV A,30H	Direct memory address
<b>Indirect</b>	MOV A,@R0	Register માં stored address
<b>Indexed</b>	MOVC A,@A+DPTR	Base address plus offset

- **Immediate Mode:** Instruction માં constant data included છે
- **Register Mode:** Register file ઉપયોગ કરીને સૌથી ઝડપી execution
- **Direct Mode:** કોઈપણ internal RAM location access કરે છે
- **Indirect Mode:** Arrays માટે pointer-based addressing
- **Indexed Mode:** Table lookup અને array access

મેમરી ટ્રીક

“Immediate, Register, Direct, Indirect, Indexed Addressing”

પ્રશ્ન 4(ક) [7 ગુણ]

8 data block ને શરૂઆત ના address location 100h થી 200h માં copy કરવાનો program લખો.

જવાબ

Assembly Program:

```
ORG 0000H           ; Start address
MOV R0,\#100H        ; Source address pointer
MOV R1,\#200H        ; Destination address pointer
MOV R2,\#08H         ; 8 bytes counter

LOOP:
MOV A,@R0            ; Source data read
MOV @R1,A            ; Destination data write
INC R0               ; Source pointer increment
INC R1               ; Destination pointer increment
DJNZ R2,LOOP         ; Counter decrement zero jump

END                  ; Program end
```

Table 19: Register Usage

Register	Purpose
<b>R0</b>	Source address pointer (100H)
<b>R1</b>	Destination address pointer (200H)
<b>R2</b>	Loop counter (8 bytes)
<b>A</b>	Temporary data storage

- **Indirect Addressing:** Memory access માટે @R0 અને @R1
- **Loop Control:** DJNZ instruction decrements અને tests કરે છે
- **Block Transfer:** 8 consecutive bytes efficiently copy કરે છે

## મેમરી ટ્રીક

“Read, Write, Increment, Decrement, Jump Loop”

### પ્રશ્ન 4(અ OR) [3 ગુણ]

બે data bytes ને ઉમેરો અને result ને R0 register માં save કરો.

#### જવાબ

##### Assembly Program:

```
ORG 0000H          ; Start address
MOV A,\#25H         ; byte load
ADD A,\#35H         ; byte add
MOV R0,A            ; Result R0 store
END                 ; Program end
```

Table 20: Operation Steps

Step	Instruction	Result
1	MOV A,\#25H	A = 25H
2	ADD A,\#35H	A = 5AH
3	MOV R0,A	R0 = 5AH

- **Addition Result:** 25H + 35H = 5AH
- **Flag Effects:** Result > FFH હોય તો carry flag set થાય છે

## મેમરી ટ્રીક

“Move, Add, Move = Simple Addition”

### પ્રશ્ન 4(બ OR) [4 ગુણ]

Indexed addressing mode ઉદાહરણ સાથે સમજાવો.

#### જવાબ

Indexed addressing memory access માટે base address plus offset ઉપયોગ કરે છે.

Table 21: Indexed Addressing Details

Component	વર્ણન	Example
<b>Base Address</b>	DPTR અથવા PC register	DPTR = 1000H
<b>Index</b>	Accumulator contents	A = 05H
<b>Effective Address</b>	Base + Index	1000H + 05H = 1005H

##### Example:

```
MOV DPTR,\#1000H    ; Base address
MOV A,\#05H         ; Index value
MOVC A,@A+DPTR      ; Address 1005H read
```

- **Table Access:** Lookup tables અને arrays માટે આદર્શ
- **Program Memory:** MOVC ફક્ત code memory માંથી જ read કરે છે
- **Dynamic Indexing:** Execution દરમિયાન index બદલાઈ શકે છે

## મેમરી ટ્રીક

“Base + Index = Dynamic Access”

### પ્રશ્ન 4(ક OR) [7 ગુણ]

માઇક્રોકંટ્રોલરનું stack operation, PUSH અને POP instruction સમજાવો.

#### જવાબ

Stack એ temporary data storage માટે ઉપયોગમાં લેવાતું LIFO memory structure છે.

Table 22: Stack Operations

Operation	Instruction	Function
<b>PUSH</b>	PUSH 30H	Stack પર data store કરે છે
<b>POP</b>	POP 30H	Stack માંથી data retrieve કરે છે
<b>Stack Pointer</b>	SP register	Stack ની top તરફ point કરે છે

#### સાચાશ્રામ:

Stack Operation:

PUSH :	PUSH 30H :	POP 30H :
SP 07H	SP 08H	SP 07H
06H	08H: 30H	06H
05H	07H: old	05H

Stack memory upward grow

#### Example Program:

```
MOV SP, \#30H      ; Stack pointer initialize
PUSH ACC            ; Accumulator save
PUSH B              ; B register save
POP B               ; B register restore
POP ACC             ; Accumulator restore
```

- **LIFO Structure:** Last In, First Out data organization
- **SP Auto-increment:** Stack pointer આપમેળે adjust થાય છે
- **Subroutine Calls:** Stack return addresses save કરે છે
- **Register Preservation:** Register contents save/restore કરે છે

#### મેમરી ટ્રીક

“PUSH Puts Up, Stack Holds, POP Pulls Out”

### પ્રશ્ન 5(અ) [3 ગુણ]

Branching instruction ઉદાહરણ સાથે સમજાવો.

#### જવાબ

Branching instructions conditions આધારે અથવા unconditionally program flow alter કરે છે.

Table 23: Branching Instructions

Type	Instruction	Example
<b>Unconditional</b>	LJMP address	LJMP 2000H
<b>Conditional</b>	JZ address	JZ ZERO_LABEL
<b>Call/Return</b>	LCALL address	LCALL SUBROUTINE

### Example:

```
MOV A,\#00H          ; Zero load
JZ ZERO\_FOUND        ; A zero    jump
LJMP CONTINUE         ; Continue   jump
ZERO\_FOUND:
    MOV R0,\#01H      ; Flag set
CONTINUE:
    NOP                ; Execution continue
```

- **Program Control:** Execution sequence બદલે છે
- **Conditional Jumps:** Flag register status આધારે
- **Address Range:** કોઈપણ program memory location પર jump કરી શકે છે

### મેમરી ટ્રીક

“Jump Changes Control Flow”

## પ્રશ્ન 5(બ) [4 ગુણ]

માઇક્રોકંટ્રોલર સાથે 8 LEDs ને interface કરો અને તેને on અને off કરવા માટેનો program લખો.

### જવાબ

#### Circuit Diagram:

```
8051          LEDs
P1.0 {-{-{-{-} [330Ω] {-{-{-{-} LED1 {-{-{-{-} +5V}
P1.1 {-{-{-{-} [330Ω] {-{-{-{-} LED2 {-{-{-{-} +5V }
P1.2 {-{-{-{-} [330Ω] {-{-{-{-} LED3 {-{-{-{-} +5V}
P1.3 {-{-{-{-} [330Ω] {-{-{-{-} LED4 {-{-{-{-} +5V}
P1.4 {-{-{-{-} [330Ω] {-{-{-{-} LED5 {-{-{-{-} +5V}
P1.5 {-{-{-{-} [330Ω] {-{-{-{-} LED6 {-{-{-{-} +5V}
P1.6 {-{-{-{-} [330Ω] {-{-{-{-} LED7 {-{-{-{-} +5V}
P1.7 {-{-{-{-} [330Ω] {-{-{-{-} LED8 {-{-{-{-} +5V}
```

#### Program:

```
ORG 0000H
MAIN:
    MOV P1,\#0FFH      ; LEDs ON
    CALL DELAY          ; Wait
    MOV P1,\#00H       ; LEDs OFF
    CALL DELAY          ; Wait
    SJMP MAIN           ; Repeat

DELAY:
    MOV R0,\#0FFH      ; Outer loop counter
LOOP1:
    MOV R1,\#0FFH      ; Inner loop counter
LOOP2:
    DJNZ R1,LOOP2       ; Inner delay loop
    DJNZ R0,LOOP1       ; Outer delay loop
    RET                 ; Return
END
```

Table 24: Components

Component	Value	Purpose
Resistor	330Ω	Current limiting
LEDs	8 pieces	Visual indicators
Port	P1	8-bit output port

- **Current Limiting:** Resistors LEDs ને overcurrent થી protect કરે છે
- **Port Configuration:** LED control માટે P1 ને output port તરીકે ઉપયોગ
- **Delay Routine:** Visible ON/OFF timing બનાવે છે

### મેમરી ટ્રીક

“Port Controls LEDs with Resistance and Delay”

## પ્રશ્ન 5(ક) [7 ગુણ]

માઇક્રોકંટ્રોલર સાથે LCD ને interface કરો અને “welcome” display કરવાનો program લખો.

### જવાબ

#### Circuit Connections:

```
8051          16x2 LCD
P2.0 {-}{-}{-}{-}{-}{-}{-}{-} D4}
P2.1 {-}{-}{-}{-}{-}{-}{-}{-} D5 }
P2.2 {-}{-}{-}{-}{-}{-}{-}{-} D6}
P2.3 {-}{-}{-}{-}{-}{-}{-}{-} D7}
P1.0 {-}{-}{-}{-}{-}{-}{-}{-} RS (Register Select)}
P1.1 {-}{-}{-}{-}{-}{-}{-}{-} EN (Enable)}
GND  {-}{-}{-}{-}{-}{-}{-}{-} R/W (Write mode)}
```

#### Program:

```
ORG 0000H
    CALL LCD\_INIT      ; LCD initialize
    CALL DISPLAY\_MSG   ; Message display
    SJMP $              ; stop

LCD\_INIT:
    MOV P2,\#38H        ; Function set: 8{-}bit, 2{-}line}
    CALL COMMAND
    MOV P2,\#0EH        ; Display ON, Cursor ON
    CALL COMMAND
    MOV P2,\#01H        ; Display clear
    CALL COMMAND
    MOV P2,\#06H        ; Entry mode set
    CALL COMMAND
    RET

DISPLAY\_MSG:
    MOV DPTR,\#MESSAGE  ; Message point
NEXT\_CHAR:
    CLR A
    MOVC A,@A+DPTR      ; Character read
    JZ DONE             ; Zero string end
    CALL SEND\_CHAR      ; LCD character send
    INC DPTR             ; Next character
    SJMP NEXT\_CHAR

DONE:
    RET

COMMAND:
    CLR P1.0            ; Command RS = 0
    SETB P1.1           ; EN = 1
    CLR P1.1            ; EN = 0 (pulse)
    CALL DELAY
    RET
```

```

SEND\_CHAR:
    MOV P2,A           ; Data lines character put
    SETB P1.0          ; Data RS = 1
    SETB P1.1          ; EN = 1
    CLR P1.1           ; EN = 0 (pulse)
    CALL DELAY
    RET

DELAY:
    MOV R0,\#50        ; Delay routine
DELAY\_LOOP:
    MOV R1,\#255
DELAY\_INNER:
    DJNZ R1,DELAY\_INNER
    DJNZ R0,DELAY\_LOOP
    RET

MESSAGE:
    DB "WELCOME",0     ; Null terminator message string
END

```

Table 25: LCD Interface Pins

8051 Pin	LCD Pin	Function
P2.0-P2.3	D4-D7	4-bit data lines
P1.0	RS	Register select (0=command, 1=data)
P1.1	EN	Enable pulse
GND	R/W	Read/Write (write માટે ground સાથે tied)

- **4-bit Mode:** Pins save કરવા માટે ફક્ત upper 4 data lines ઉપયોગ કરે છે
- **Control Signals:** RS command/data select કરે છે, EN timing pulse આપે છે
- **Character Display:** દરેક character ASCII code તરીકે send થાય છે
- **Initialization:** Proper operation માટે જરૂરી command sequence

## મેમરી ટ્રીક

“LCD Displays Characters with Commands and Data”

## પ્રશ્ન 5(અ OR) [3 ગુણ]

Logical instruction ઉદાહરણ સાથે સમજાવો.

### જવાબ

Logical instructions data પર bitwise operations કરે છે.

Table 26: Logical Instructions

Instruction	Example	Function
ANL	ANL A,#0FH	Bitwise AND operation
ORL	ORL A,#F0H	Bitwise OR operation
XRL	XRL A,#FFH	Bitwise XOR operation



### Example:

```
MOV A,\#55H      ; A = 01010101B
ANL A,\#0FH      ; A = 00000101B (upper bits mask )
ORL A,\#FOH      ; A = 11110101B (upper bits set )
XRL A,\#FFH      ; A = 00001010B ( bits complement )
```

- **Bit Manipulation:** Bits setting, clearing અને testing માટે ઉપયોગ થાય છે
- **Masking Operations:** ANL unwanted bits clear કરે છે
- **Flag Effects:** Result આધારે parity flag update થાય છે

### મેમરી ટ્રીક

``AND Masks, OR Sets, XOR Toggles``

## પ્રશ્ન 5(બ OR) [4 ગુણ]

માઇક્રોકંટ્રોલર સાથે 7 segment ને interface કરો.

### જવાબ

#### Circuit Diagram:

```
8051      7{-Segment Display}
P1.0 {-}{-}{-}{-}{-}[330Ω]{-}{-}{-}{-}{-}a}
P1.1 {-}{-}{-}{-}{-}[330Ω]{-}{-}{-}{-}{-}b }
P1.2 {-}{-}{-}{-}{-}[330Ω]{-}{-}{-}{-}{-}c}
P1.3 {-}{-}{-}{-}{-}[330Ω]{-}{-}{-}{-}{-}d}
P1.4 {-}{-}{-}{-}{-}[330Ω]{-}{-}{-}{-}{-}e}
P1.5 {-}{-}{-}{-}{-}[330Ω]{-}{-}{-}{-}{-}f}
P1.6 {-}{-}{-}{-}{-}[330Ω]{-}{-}{-}{-}{-}g}
P1.7 {-}{-}{-}{-}{-}[330Ω]{-}{-}{-}{-}{-}dp (decimal point)}
```

#### Program to Display 0-9:

```
ORG 0000H
MOV DPTR,\#DIGIT\_TABLE ; Lookup table point
MOV R0,\#0 ; Digit 0 start

MAIN\_LOOP:
MOV A,R0 ; Current digit get
MOVC A,@A+DPTR ; 7{-segment code get }
MOV P1,A ; 7{-segment display }
CALL DELAY ; 1 second wait
INC R0 ; Next digit
CJNE R0,\#10,MAIN\_LOOP ; 10 reached check
MOV R0,\#0 ; 0 reset
SJMP MAIN\_LOOP ; Repeat

DIGIT\_TABLE:
DB 3FH, 06H, 5BH, 4FH, 66H ; 0,1,2,3,4
DB 6DH, 7DH, 07H, 7FH, 6FH ; 5,6,7,8,9
END
```

Table 27: 7-Segment Codes

Digit	Hex Code	Binary	Segments Lit
0	3FH	00111111	a,b,c,d,e,f
1	06H	00000110	b,c
2	5BH	01011011	a,b,g,e,d

- **Common Cathode:** Port pin high હોય ત્યારે segments light થાય છે
- **Current Limiting:** Resistors segment damage અટકાવે છે
- **Lookup Table:** Segment patterns નું efficient storage

## મેમરી ટ્રીક

“Seven Segments Show Digits Clearly”

**પ્રશ્ન 5(ક OR) [7 ગુણ]**

માઇક્રોકંટ્રોલર સાથે LM 35 ને interface કરો અને temperature controller નો block diagram સમજાવો.

정답이

**Circuit Diagram:**

LM35 Temperature Sensor Interface:

```
+5V {-}{-}{-}{-}{+}{-}{-}{-}{-} LM35 {-}{-}{-}{-}{+}{-}{-}{-}{-} ADC0804 {-}{-}{-}{-}{+}{-}{-}{-}{-}
    |      (Vout)   |      (Vin)       |      (P1)
    GND            GND                  |
                                         |
Relay Control:                        |
8051 P3.0 {-}{-}{-}{-}[ULN2003]{-}{-}{-}{-} Relay {-}{-}{-}{-}{-}{-}{-}{-}{+}
                                           |
                                           Load (Heater/Fan)
```

**Temperature Controller Block Diagram:**

### Mermaid Diagram (Code)

```
{Shaded}
{Highlighting}[]
graph LR
    A[LM35 Sensor] --{} B[ADC0804]}
    B --{} C[8051 Controller]}
    C --{} D[Display Unit]}
    C --{} E[Relay Driver]}
    E --{} F[Heater/Cooler]}
    F --{} G[Controlled Environment]}
    G --{} A}
{Highlighting}
{Shaded}
```

## Control Program:

```

ORG 0000H
MAIN:
    CALL READ\_TEMP      ; ADC    temperature read
    CALL DISPLAY\_TEMP   ; Display temperature show
    CALL TEMP\_CONTROL   ; Heating/cooling control
    CALL DELAY           ; Next reading    wait
    SJMP MAIN

READ\_TEMP:
    CLR P2.0             ; ADC conversion start
    SETB P2.0            ; Start    pulse
    JNB P2.1,$           ; Conversion complete    wait
    MOV A,P1             ; Temperature data read
    RET

```

```

TEMP\_CONTROL:
  CJNE A,\#30,CHECK\_HIGH    ; Setpoint (30^ ) compare
CHECK\_HIGH:
  JC TEMP\_LOW                ; A { 30      temperature low  }
  SETB P3.0                   ; Cooling (fan) ON
  CLR P3.1                    ; Heating OFF
  RET
TEMP\_LOW:
  CLR P3.0                    ; Cooling OFF
  SETB P3.1                   ; Heating ON
  RET
END

```

Table 28: System Components

Component	Function
LM35	Temperature sensor (10mV/°)
ADC0804	Analog to digital converter
8051	Main controller
Relay	High power loads switch કરે છે
Display	Current temperature show કરે છે

- **Temperature Sensing:** LM35 દરેક degree Celsius માટે 10mV આપે છે
- **ADC Conversion:** Analog voltage ને digital value માં convert કરે છે
- **Control Logic:** Setpoint સાથે compare કરે છે અને relays control કરે છે
- **Feedback System:** Continuous monitoring અને adjustment
- **Safety Features:** Over-temperature protection શક્ય છે

### મેમરી ટ્રીક

“Sense, Convert, Compare, Control Temperature Automatically”