DELHI SKILL AND ENTREPRENEURSHIP UNIVERSITY

End-Semester Examination – Set 3
B.Tech CSE – Microprocessors and Microcontrollers (BT-EC-ES511)

Max. Marks: 100 Time: 3 Hours

Instructions:

1. This question paper contains two sections: Section A and Section B.
2. Attempt **all** questions.
3. Figures to the right indicate full marks.
4. Assume necessary data if missing.

SECTION A – Short Answer Questions

Attempt all questions. Answers should be concise (3–4 lines).

Q1. Explain the function of the HOLD and HLDA pins in the 8085 microprocessor.

Answer: ***HOLD:*** *An input signal from an external device (like a DMA controller) requesting control of the system bus.*
HLDA (Hold Acknowledge): *An output signal indicating that the CPU has received the HOLD request and will surrender the bus after the current machine cycle.*

Q2. Describe the 8085 Flag Register status bits (S, Z, AC, P, CY).

Answer: *The 5 flags indicate the result of ALU operations:*

- ***S (Sign):*** *Set to 1 if MSB is 1 (Negative).*
- ***Z (Zero):*** *Set to 1 if result is zero.*
- ***AC (Aux Carry):*** *Set to 1 if carry moves from bit D3 to D4.*
- ***P (Parity):*** *Set to 1 for even parity.*
- ***CY (Carry):*** *Set to 1 if result generates a carry/borrow out of MSB.*

Q3. What is the purpose of the `LOCK` prefix in 8086 instructions?

Answer: *The `LOCK` prefix is used in multiprocessor systems. When placed before an instruction, it activates the `LOCK` output pin (active low), ensuring that the CPU retains control of the system bus for the duration of that instruction, preventing other processors from taking over the bus.*

Q4. Calculate the memory range addressed by the segment register if DS = 4000H.

Answer: *The Segment Register (DS) provides the base address.*

- ***Start Address:*** $4000H \times 10H + 0000H = 40000H$.
- ***End Address:*** $4000H \times 10H + FFFFH = 4FFFFH$.

*Thus, the range is **40000H to 4FFFFH** (64KB segment).*

Q5. Differentiate between `LEA` and `MOV` instructions in 8086.

Answer: *`MOV` transfers **data** from source to destination.*
*`LEA` (Load Effective Address) calculates the **offset address** of the source operand and loads it into a 16-bit register. Example: `LEA BX, [SI+4]` loads the calculated address into BX, not the data at that address.*

Q6. What is the function of the Gate terminal in the 8253/8254 Timer?

Answer: *The Gate terminal is used to control the operation of the counter:*

- *In some modes (e.g., Mode 0), keeping Gate Low inhibits counting.*
- *In other modes (e.g., Mode 1), a rising edge on Gate triggers the counting sequence (Hardware Trigger).*

Q7. Explain the concept of 'Key Debouncing' in microprocessor interfacing.

Answer: *Mechanical switches generate multiple transient signals (bounces) when pressed or released due to spring action. Debouncing is the process (hardware or software delay) of ignoring these transient spikes to ensure only one clean signal is registered per key press.*

Q8. What is the resolution (step size) of an 8-bit DAC if the reference voltage is 5V?

Answer: *Resolution* $= \frac{V_{ref}}{2^n - 1}$ *(or approximated as* 2^n*).*
Step Size $= \frac{5V}{2^8} = \frac{5}{256} \approx 19.53\ mV$.

Q9. Distinguish between `LJMP`, `AJMP`, and `SJMP` in 8051.

Answer:

- ***LJMP (Long Jump):*** *3-byte instruction, jumps anywhere in 64KB code space (16-bit address).*
- ***AJMP (Absolute Jump):*** *2-byte instruction, jumps within the same 2KB page (11-bit address).*
- ***SJMP (Short Jump):*** *2-byte instruction, relative jump within -128 to +127 bytes.*

Q10. What is the function of the $\overline{\text{PSEN}}$ pin in 8051?

Answer: $\overline{PSEN}$ *(Program Store Enable) is an output signal connected to the* $\overline{OE}$ *pin of external ROM. It activates (goes low) during the fetch cycle to read opcodes from external program memory.*

SECTION B – Descriptive Questions

Attempt all questions. Include diagrams where necessary.

Q11. Draw the circuit to demultiplex the address/data bus (AD0–AD7) of 8085 using a latch. Explain the timing of ALE regarding this operation.

Answer:

Operation:

(a) During **T1 state** of a machine cycle, the 8085 places the low-order address on lines $AD_0 - AD_7$.

(b) It asserts **ALE High**. This enables the latch (74LS373).

(c) When ALE goes **Low** (end of T1), the address is "latched" and stored on the output of the 74LS373. These outputs become $A_0 - A_7$.

(d) During **T2 and T3**, the bus $AD_0 - AD_7$ is used for Data ($D_0 - D_7$), while the latch holds the address stable for memory/IO.

*The 8085 uses a multiplexed bus where lower order address ($A_0 - A_7$) and data ($D_0 - D_7$) share the same pins ($AD_0 - AD_7$). To separate them, an external latch (typically 74LS373) is used, controlled by the **ALE (Address Latch Enable)** signal.*

Operation:

1. *During **T1 state** of a machine cycle, the 8085 places the low-order address on lines $AD_0 - AD_7$.*
2. *It asserts **ALE High**. This enables the latch (74LS373).*
3. *When ALE goes **Low** (end of T1), the address is "latched" and stored on the output of the 74LS373. These outputs become $A_0 - A_7$.*
4. *During **T2 and T3**, the bus $AD_0 - AD_7$ is used for Data ($D_0 - D_7$), while the latch holds the address stable for memory/IO.*

Write an 8085 assembly language program to multiply two 8-bit numbers stored at 2050H and 2051H. Store the 16-bit result at 2052H and 2053H.
Answer:

LOOP: DAD D ; HL = HL + DE (Add multiplicand to result) DCR C ; Decrement multiplier JNZ LOOP ; Repeat until C = 0

SHLD 2052H ; Store 16-bit result at 2052H (L) and 2053H (H) HLT ; Stop
**(Note: This is a simplified logic assuming unsigned numbers).* Multiplication is performed by repeated addition.*

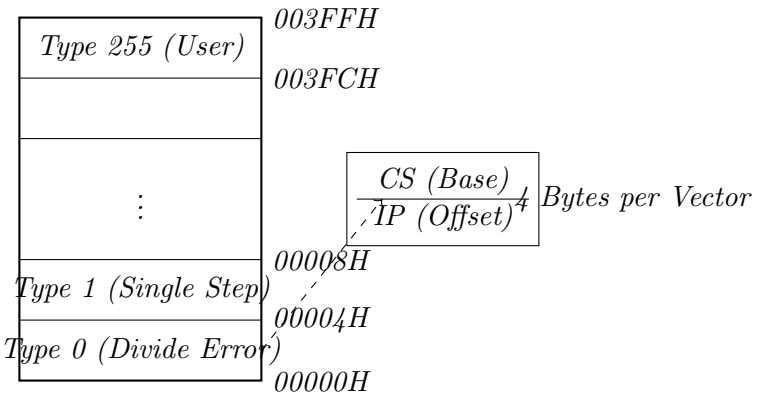
```
LOOP: DAD D ; HL = HL + DE (Add multiplicand to result) DCR C ; Decrement multiplier
JNZ LOOP ; Repeat until C = 0
SHLD 2052H ; Store 16-bit result at 2052H (L) and 2053H (H) HLT ; Stop
```

(Note: This is a simplified logic assuming unsigned numbers).

Explain the structure of the Interrupt Vector Table (IVT) in 8086. How does the processor locate the ISR for a Type-N interrupt?
Answer:

Structure:

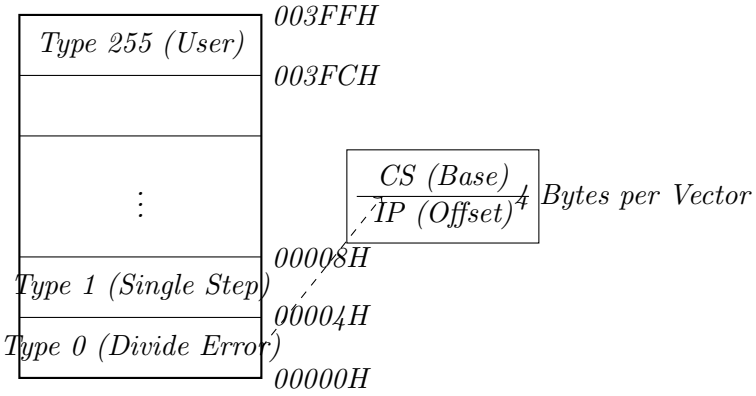
- *Located at the bottom of the memory map:* ***00000H to 003FFH*** *(First 1KB).*
- *Contains* ***256 vectors*** *(Type 0 to Type 255).*
- *Each vector is* ***4 bytes*** *long.*
  - *Lower 2 bytes:* ***IP*** *(Instruction Pointer / Offset).*
  - *Higher 2 bytes:* ***CS*** *(Code Segment).*



Locating ISR for Type-N: *1. Processor multiplies Interrupt Type (N) by 4. Address =* $N \times 4$.
2. Reads 2 bytes from Address into ***IP****.*
3. Reads next 2 bytes from Address + 2 into ***CS****.*
4. Program control jumps to new $CS : IP$*. The* ***Interrupt Vector Table (IVT)*** *is a memory look-up table used by the 8086 to locate Interrupt Service Routines (ISRs).*

Structure:

- *Located at the bottom of the memory map:* ***00000H to 003FFH*** *(First 1KB).*
- *Contains* ***256 vectors*** *(Type 0 to Type 255).*
- *Each vector is* ***4 bytes*** *long.*
  - *Lower 2 bytes:* ***IP*** *(Instruction Pointer / Offset).*
  - *Higher 2 bytes:* ***CS*** *(Code Segment).*



Locating ISR for Type-N: *1. Processor multiplies Interrupt Type (N) by 4. Address* $= N \times 4$.
2. Reads 2 bytes from Address into ***IP****.*
3. Reads next 2 bytes from Address $+ 2$ *into* ***CS****.*
4. Program control jumps to new $CS : IP$.

Draw the functional block diagram of 8253/8254 Programmable Interval Timer. Briefly explain the six operating modes.
Answer:

Operating Modes:

- ***Mode 0 (Interrupt on Terminal Count):*** *Output goes High when count reaches 0.*
- ***Mode 1 (Hardware Retriggerable One-Shot):*** *Output goes Low for a count duration after Gate trigger.*
- ***Mode 2 (Rate Generator):*** *Generates pulses at regular intervals (Divide-by-N).*
- ***Mode 3 (Square Wave Generator):*** *Generates square wave (useful for clocks).*
- ***Mode 4 (Software Triggered Strobe):*** *Strobe occurs after software loads count.*
- ***Mode 5 (Hardware Triggered Strobe):*** *Strobe occurs after hardware Gate trigger.*

The 8253/8254 contains three independent 16-bit counters (Counter 0, 1, 2), a data bus buffer, and control logic.

Operating Modes:

- ***Mode 0 (Interrupt on Terminal Count):*** *Output goes High when count reaches 0.*
- ***Mode 1 (Hardware Retriggerable One-Shot):*** *Output goes Low for a count duration after Gate trigger.*
- ***Mode 2 (Rate Generator):*** *Generates pulses at regular intervals (Divide-by-N).*
- ***Mode 3 (Square Wave Generator):*** *Generates square wave (useful for clocks).*
- ***Mode 4 (Software Triggered Strobe):*** *Strobe occurs after software loads count.*
- ***Mode 5 (Hardware Triggered Strobe):*** *Strobe occurs after hardware Gate trigger.*

Explain the interfacing of a Digital-to-Analog Converter (DAC) using an R-2R ladder network concept.

Answer:

R-2R Ladder Principle: *Instead of using weighted resistors (R, 2R, 4R...) which require high precision, the R-2R ladder uses only two values: R and $2R$.*

- *It divides the reference voltage/current based on the binary input bits.*
- *The output current I_{out} is proportional to the digital value.*
- *An Op-Amp (Operational Amplifier) is usually connected at the output to convert Current to Voltage (I-to-V converter).*

Application: *Used for generating waveforms (Sawtooth, Triangular, Sine) by sending varying digital values in a loop. Interfacing a DAC (like DAC0808) involves connecting the data bus of the microprocessor (via a latch like 8255 Port) to the digital inputs of the DAC.*

R-2R Ladder Principle: *Instead of using weighted resistors (R, 2R, 4R...) which require high precision, the R-2R ladder uses only two values: R and $2R$.*

- *It divides the reference voltage/current based on the binary input bits.*
- *The output current I_{out} is proportional to the digital value.*
- *An Op-Amp (Operational Amplifier) is usually connected at the output to convert Current to Voltage (I-to-V converter).*

Application: *Used for generating waveforms (Sawtooth, Triangular, Sine) by sending varying digital values in a loop.*

Detailed explanation of the Addressing Modes available in 8051 Microcontroller with examples.

Answer:

1. ***Immediate Addressing:*** *Data is provided directly in the instruction.*

- *Example:* `MOV A, #25H` *(Load 25H into Accumulator).*
- *Note: The '#' symbol indicates immediate data.*

2. ***Register Addressing:*** *Uses registers (R0–R7) from the selected bank.*

- *Example: `MOV A, R0` (Copy content of R0 to A).*
- *Faster execution as data is internal.*

3. ***Direct Addressing:*** *The address of the operand is specified directly. Only way to access SFRs and upper 128 bytes of RAM.*

- *Example: `MOV 90H, A` (Move A to Port 1, since P1 address is 90H).*
- *Example: `MOV 30H, #55H` (Move 55H to RAM address 30H).*

4. ***Register Indirect Addressing:*** *Uses a register (R0 or R1 only) to hold the address of the data.*

- *Example: `MOV A, @R0` (Move data from address held in R0 to A).*
- *The '@' symbol denotes indirection.*

5. ***Indexed Addressing:*** *Used for accessing data from Code memory (ROM), typically for lookup tables. Uses PC or DPTR.*

- *Example: `MOVC A, @A+DPTR` (Move byte from address DPTR+A to A).*

The 8051 supports 5 major addressing modes:

1. ***Immediate Addressing:*** *Data is provided directly in the instruction.*

- *Example: `MOV A, #25H` (Load 25H into Accumulator).*
- *Note: The '#' symbol indicates immediate data.*

2. ***Register Addressing:*** *Uses registers (R0–R7) from the selected bank.*

- *Example: `MOV A, R0` (Copy content of R0 to A).*
- *Faster execution as data is internal.*

3. ***Direct Addressing:*** *The address of the operand is specified directly. Only way to access SFRs and upper 128 bytes of RAM.*

- *Example: `MOV 90H, A` (Move A to Port 1, since P1 address is 90H).*
- *Example: `MOV 30H, #55H` (Move 55H to RAM address 30H).*

4. ***Register Indirect Addressing:*** *Uses a register (R0 or R1 only) to hold the address of the data.*

- *Example: `MOV A, @R0` (Move data from address held in R0 to A).*
- *The '@' symbol denotes indirection.*

5. ***Indexed Addressing:*** *Used for accessing data from Code memory (ROM), typically for lookup tables. Uses PC or DPTR.*

- *Example: `MOVC A, @A+DPTR` (Move byte from address DPTR+A to A).*

How is external Program Memory and Data Memory interfaced with 8051? Explain the control signals used.

Answer:

Control Signals:

1. $\overline{\textbf{PSEN}}$ ***(Program Store Enable):*** *Connects to the* $\overline{OE}$ *of External* ***ROM****. It is active only during code fetch.*
2. $\overline{\textbf{RD}}$ ***(Read):*** *Connects to* $\overline{OE}$ *of External* ***RAM****. Active during data read instructions (`MOVX`).*
3. $\overline{\textbf{WR}}$ ***(Write):*** *Connects to* $\overline{WE}$ *of External* ***RAM****. Active during data write instructions.*
4. ***ALE:*** *Used to demultiplex P0 into address (A0-A7) and data (D0-D7).*

Ports Usage: *Port 0 (Multiplexed Addr/Data), Port 2 (High byte Address A8-A15). The 8051 can access up to 64KB External Code (ROM) and 64KB External Data (RAM).*

Control Signals:

1. $\overline{\textbf{\textit{PSEN}}}$ ***(Program Store Enable):*** *Connects to the* $\overline{OE}$ *of External* ***ROM****. It is active only during code fetch.*
2. $\overline{\textbf{\textit{RD}}}$ ***(Read):*** *Connects to* $\overline{OE}$ *of External* ***RAM****. Active during data read instructions (`MOVX`).*
3. $\overline{\textbf{\textit{WR}}}$ ***(Write):*** *Connects to* $\overline{WE}$ *of External* ***RAM****. Active during data write instructions.*
4. ***ALE:*** *Used to demultiplex P0 into address (A0-A7) and data (D0-D7).*

Ports Usage: *Port 0 (Multiplexed Addr/Data), Port 2 (High byte Address A8-A15).*