Descending: A Series of 8-Bit Decimal Number SUM

LXI H, 2500H
MOV C, M
DCRC
MOV D, C
LXI H, 4201H
Loop: INX H

Loop: MOV A, H
INX H
ADD M
DAA
CMP H
JNC Ahead

JNC
MOV B, M
Ahead: DCRC
MOV M, B
DCX H
MOV M, B
INX H
DCR D
JNZ Loop
DCR C
JNZ
HLT

A Series of 8-Bit no sum 8-Bit:
LXI H, 2500H
MOV CM
MVI A, 00
Loop: INX H
ADD M
DCR C
JNZ Loop
STA 2450H
HLT

Find the largest to affect a series:

LXI H, 2500H
MOV C, H
SUB A
INR B
INR B
no DAA
INR B
no DAA
INR B

LXI H, 2500H
MOV A, H
Ahead: DCRC
JNZ Loop
STA 2450H
HLT

MVI A, 00
Due: Input has post has address of 01H & has 9
   data 05H to input

IN 01H
ANI 80H

After execution of two instructions, what are contents
   of flag register?

Sol: 

<table>
<thead>
<tr>
<th>S</th>
<th>Z</th>
<th>X</th>
<th>A</th>
<th>C</th>
<th>P</th>
<th>X</th>
<th>CY</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

PSW:

00000001
10000000
00000000

&

Write an assembly language program to add two 8-bit
   numbers with carry and store 3 byte result starting
   at memory location 2030H.

Sol:

LHLD 2001H
XCHG D
LHLD 2003H
MVI C,00
DAD 10
Je

No Carry
SHLD 2008H
Carry IN RC
Mova C
STA 2030H
SHLD 2008H
HLT
Modes/Features of 8255

8255 is a programmable peripheral interface which is used to interface input/output devices with microprocessors and interfacing module in which it operate is known as ports. And how it works depending upon the types of modes of operation.

8255 having 3 ports i.e. Port A, Port B, Port C

Port A = 8 bit port

Port B = 8 bit port

Port C = Divided into two part Port C upper & Port C lower

4 bit each

8255 architecture divided into two groups - Group A & Group B. Group A contain Port A and Port C upper & Group B contain Port B & Port C lower.

8255 operate in input/output modes as well as bit beat reset mode. This mode is only used for Port C.

Modes of operation:

1) Mode 0
2) Mode 1
3) Mode 2

Mode 0: It is known as simple input-output mode

Mode 1: Known as Stop (handshaking)

Mode 2: Mode 2 be at bidirectional mode
Data will be transferred as well as recieved simultaneously by input output mode.

Architecture of 8255:

- **Mode 0 features:**
  1. Simple It acts as simple I/O mode means post can be used as either input mode or output mode e.g. mouse as well as printer can be attach
  2. Inputs are not latched (not temporarily stored) but output are latched

- **Mode 1 features:**
  1. In Mode 1 posts are controlled by a device
  2. In this mode handshaking is done
  3. Port A & B are used only for input output mode for transferring data and receive under the control of port C
In this mode both input & as well as output are latched. It consists of two group. Group A & B in each group 6 bit used for input output mode & 3 bit of port c used for controlling purpose.

**Mode 2: Features**

1. Used as a bidirectional mode in which data is transferred as well as received simultaneously.
2. This mode Related with only group A. It means port B & port C is not associated with bidirectional data transferring and receiving.
3. In this mode input & output both are latched.

\[ -> \quad \text{DAD reg} \rightarrow \text{opcode fetch} \]
\[ \quad \rightarrow 2 \text{ Memory Read (not usual operation is used only status signal soft) } \]
\[ \quad 10 \text{- T state} \rightarrow \text{MR} \]

\[ -> \quad \text{LDA 16 bit Address} \rightarrow \text{opcode fetch} \]
\[ \quad \rightarrow \text{Byte} \rightarrow 3 \text{-T} \rightarrow \text{MR} \]
\[ \quad \text{operation} \rightarrow 3 \text{-T} \rightarrow \text{MR} \]

\[ \rightarrow \quad \text{CALL 16 bit address.} \]
\[ \quad \rightarrow \text{opcode fetch} \rightarrow 6 \text{-T state} \]
\[ \quad \rightarrow \text{3 bytes} \rightarrow \text{MR} \]
\[ \quad \rightarrow \text{MR} \rightarrow \text{MR} \rightarrow \text{MR} \]

1 clock cycle for ALE, 2 if 3 will be ideal (--)
RET → opcode → 4 - T-state
4 MR → 12 T-state → 16 T-state
RST n → opcode → 6 - T-state

T-state & Machine cycle for all instructions of 8085

1) MOV A, R(c-e, d-e, h-l) ÷ 4 T-state + 1 Machine cycle
2) MOV R, M ÷ 7 T-state (opcode + MR) ÷ 2 Machine cycle
3) MOVM, R ÷ 7 T-state (opcode + MR) ÷ 2
4) MVI n, data ÷ 4 T-state ÷ 2 Machine cycle
4) MOV R, data ÷ 10 T-state (opcode + MR) ÷ 3 Machine cycle
6) LDA 16 bit address ÷ 13 T-state (opcode + 3 MR)
7) LXH 16 bit data ÷ (opcode + MR + MR) 10 T-state
8) LHLD 16 bit address ÷ (opcode + 4 MR) 16 T-state
9) LDA X, rp ÷ (opcode + 2 MR) ÷ indirect way transfer
10) STA 16 bit address ÷ (opcode + 2 MR + MR + MR)
11) SHLD 16 bit address ÷ (opcode + 2 MR + 2 MW)
12) STAX rp ÷ (opcode + 2 MR)
13) XCHG ÷ (opcode) → 4 - T-state
14) \( \text{XTHL} \rightarrow (\text{opcode}) \rightarrow 4-\text{Tstate} \)

15) \( \text{STHL} \)

16) \( \text{ADD R (opcode)} \rightarrow 4-\text{Tstate} \rightarrow \text{SUB R} \)

17) \( \text{ADD M (opcode+MR)} \rightarrow 7-\text{Tstate} \rightarrow \text{SUB M} \)

18) \( \text{ADS, 8-bit data (opcode+MR)} \rightarrow 7-\text{Tstate} \rightarrow \text{SUB, 8-bit data} \)

19) \( \text{ADC R \rightarrow (opcode)} \rightarrow 4-\text{Tstate} \rightarrow \text{SB B R} \)

20) \( \text{ADC M \rightarrow (opcode+MR)} \rightarrow 7-\text{Tstate} \rightarrow \text{SB B M} \)

21) \( \text{ACI, 8-bit data (opcode+MR)} \rightarrow 7-\text{Tstate} \rightarrow \text{SB B1, 8-bit data} \)

22) \( \text{DAD, 2p \rightarrow (opcode+2MR)} \rightarrow 10-\text{Tstate} \)

23) \( \text{ANAR \rightarrow (opcode fetch)} \rightarrow 4-\text{Tstate} \rightarrow \text{ORA R} \rightarrow \text{XRA R} \)

24) \( \text{ANAM \rightarrow (opcode+MR)} \rightarrow 7-\text{Tstate} \rightarrow \text{ORA M} \rightarrow \text{XRA M} \)

25) \( \text{ANI, 8-bit data (opcode+MR)} \rightarrow 7-\text{Tstate} \rightarrow \text{OR1, 8-bit data, XR1, 8-bit data} \)

26) \( \text{MTR, CLR RAR, RLC, RRC, RAL} \rightarrow 4\text{-Tstate (opcode)} \)

27) \( \text{STE, CLE} \)

28) \( \text{CMP R \rightarrow (opcode)} \rightarrow 4\text{-Tstate} \)

29) \( \text{CMP M \rightarrow (opcode+MR)} \rightarrow 7\text{-Tstate} \)

30) \( \text{CPG, 8-bit data \rightarrow (opcode+HR)} \rightarrow 7\text{-Tstate} \)

31) \( \text{JMR R \rightarrow 4\text{-Tstate} / DCR R} \)

32) \( \text{JMR M \rightarrow (opcode+MR)} \rightarrow 7\text{-Tstate} / \text{DCR M} \)
33) ANX rep + (opcode-fetch) → 6 T-State → DEX, rep

34) JMP, 16 bit address (opcode + 2MR) → 10 T-State

35) Conditional Jump: J → 10 T-State will be used

36) CALL 16 bit address + (opcode + 4MR) → 18 T-State

37) Conditional Call: In conditional Call 9, \( \frac{18}{6} \) T-State will be used

38) RET 16 bit address + (opcode + 4MR) → 16 T-State

39) RET (opcode + 2MR) → 10 T-State → if address is not given.

40) Conditional Return: \( \frac{6}{12} \) T-State will be used for define conditions

41) IN 8-bit data + (opcode + MR + Input/read operation) → 10 T-State

42) OUT 8-bit data + (opcode + MR + I/O write operation) → 10 T-State

43) NOP + (opcode) → 4 T-State

43) HLT + (opcode-fetch) → 5 T-State → 1 cycle required

44) RIM, SIM, DI, ES + (opcode-fetch) → 4 T-State
Features of 8253 (Programmable interval timer): 8253 is used to generate accurate time periods.

3. This timer IC having 3 three 16 bit counters.

3. If counting of sequence is greater than 16 bit two counters in combination can also be used.

4. Instead of using loops to generate delays directly counters are initialized which is documented up to a terminal point and at the end high clock pulse is generated which interrupt the CPU until next sequence is unloaded.

5. It operates at frequency of 2.6 MHz

6. It is based on N-MOS technology. For example for generating 1 sec time delay either using 8085 or with the help of 8253. In both cases count is required means up to how much time timer of 8253 or register of 8085 will be decremented for 8085 count is equal to time required for delay divided by time for execution of 1 clock cycle

\[
\text{Count} = \frac{\text{Time required for delay}}{\text{Time for execution of 1 clock cycle}}
\]

For 3.07 MHz

\[
\frac{1}{3.07 \times 10^6} = 1 \times 307 \times 10^5 \times 3.07 \times 10^5 \text{ counts}
\]

eg: 3.07 MHz

\[
5 \times 10^{-3} \times 30 \times 3.07 = 50 \text{ ms} \quad 15 \times 10^4 \times 3.07 = 5739 \quad \text{in Hexa format}
\]

\[
10 \times 15 \times 3.50 = 525 \text{ counts}
\]

\[
15 \times 3.50 = 52.5 \text{ counts}
\]

\[
15 \times 3500 = 52500
\]
100 mill sec = frequency - 2 MHz

\[
\frac{100}{MHz} = \frac{100 \times 2 \times 10^6 \times 10^{-3}}{2 MHz} = \frac{200 \times 10^3}{2 MHz}
\]

\[
30 D40H = 200000
\]

18 6 14 equals 18,614

\[
\begin{array}{c|c|c}
2 & 200000 & 200000 \\
2 & 100000 & 100000 \\
2 & 50000 & 50000 \\
2 & 25000 & 25000 \\
2 & 12500 & 12500 \\
2 & 6250 & 6250 \\
2 & 3125 & 3125 \\
\end{array}
\]
LHLD 16 bit address
XOR DCX H
loop TLZ
MOV 03H
DCX H

It will be repeated $2 \times 10^5$ minus 1 time $2 \times 10^{5-1}$
Delay will be generated for

Step 1: Check counts (Convert into Hexadecimal)
2. Check bit 0
3. Depending upon no of bit use register or register pair
   - If bit of count is to greater than 16 bit use another
     register for decrement
   - If this delay will be used as subroutine than last instruction
     will be RET

These counters in 8253 are presettable & can be
programmed for either of binary count or BCD count.

Operation of 8253:
1. Depends upon Counter
2. Depends upon mode of Operation
3. Similar to 8255, 8253 counting is described by
   address line A1 & A0 & mode of operation decided
   by control word register which is 8 bit long.
Control word Register format:

```
<table>
<thead>
<tr>
<th>BCD</th>
<th>M0</th>
<th>M1</th>
<th>M2</th>
<th>Rwno</th>
<th>Rwhnt</th>
<th>Ezo</th>
<th>Zt</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>
```

- **BCD**: If Do or LSB = 1, Count will be BCD if L.S.B = 0.
- **Count as Binary part**

### Modes

<table>
<thead>
<tr>
<th>Mb</th>
<th>M0</th>
<th>M1</th>
<th>M2</th>
<th>M4</th>
<th>MODES</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Mode0</td>
</tr>
<tr>
<td>001</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Mode1</td>
</tr>
<tr>
<td>010</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Mode3</td>
</tr>
<tr>
<td>011</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Mode4</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Mode5</td>
</tr>
</tbody>
</table>

- **Rw=0**
- **Rw=1**

#### Operation
- Counter latch command
- Execute

#### Least Significant Byte
- 0
- 1

#### Most Significant Byte
- 0
- 1

**LSB 1st then MSB**
Write a program to find largest no from an array.

```
LX 1 H, 2500 H
MOV C, M. Related with Counter
INX H
MOV A, M
DC R C
INX H
CMP M
JNC AHEAD. Related with largest & smallest
MOV A, M.
and DC R C
```
Write a program using counter to generate frequency of 50 kHz if 8253 having frequency of 1 MHz.

\[ f = \frac{50 \text{ MHz}}{50 \times 10^3} = 0.02 \times 10^{-3} \text{ sec} = 20 \mu \text{sec} \]

\[ f = 1 \text{ MHz} \Rightarrow T = 1 \times 10^{-6} \text{ sec} = 1 \mu \text{sec} \]

Counter = 20 = 20 counts

<table>
<thead>
<tr>
<th>2</th>
<th>20</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

D7 D6 D5 D4 D3 D2 D1 B0
[Sc | Sc | Rw1 | Rwo | M2 | M1 | M0 | BCD]

0 1 1 5 0 0 1 0 = 72H

MVIA 72H
OUT 13H 13 & address of control word register
MVIA 14H Put value of 14 in counter
OUT 11H
MVIA 00H
OUT 14H Put value of 14 in counter
HLT
8253 is only related with generating of clock pulses either square wave or delay or plus of any layer.

8255 is only related with input interfacing of input output device eg. LED.

8086 o
Feature:
1. 16 bit MoP
2. operating at frequency of 5 MHz depending upon its version otherwise max. frequency range lie up to 10 MHz
3. It is having 16 data lines & 20 address line.
4. It can address up to \(2^{20} \times 2^{16}\) memory location (2 Mega byte).
5. H-Mos (high width) Technology is used for 8086 due to its speed of operation.
6. 8086 operating at +5V source with having very lower power consumption.
7. 8086 can operate in 2 different operating mode known as min. mode and max. mode.
8. It is 14 pin IC's package.
**Diagram:**

- **GND (Ground)**
- **VCC + 5V**
- **AD0 - AD15**

Address/Data bus which is bidirectional and these buses are time multiplexed. It means depending upon address latch enable signal at one clock pulse address will be send another two clock pulse uses for data if address on the bus. These lines define by Symbol D. These buses contain low order address line which A16 to A19 higher order address line which are multiplexed with status signals S3, S4, S5, S6.

S3 & S4 signals are segment identifiers S5 is related with interrupts and S6 is used for future purpose as a status signal.

- **RD** – Active low signal used for read operation
- **Reset** – Used to reset the system.
NMI: Non Maskable Interrupt Request

Ready: It sends acknowledgement that input output device ready for processing & transferring of data.

Operating mode of 8086:

1. Min mode and Max modes
   - Min mode: Signals M0 P is used in this mode
   - Max mode: Multiple processes or one used for process

Pin description for Min:

- MN1Mx for this mode will be connected to +Ve

   1. INTA → After receiving interrupt request signal, this pin will activate and send acknowledge signal having zero in its pin. It is active low signal.

   2. AE (Address latch enable): for 1 clock pulse address will be send another 5 clock pulse used to send data. It is an active signal

   3. DE (Data enable): It is active low signal
      - It defines that data is enable for its processing

   4. DT/R: Data transmit/Receive. When this pin is high data is transmitted and when this pin data low data is received.
5) M150: If address violated with memory then this pin will be high, if low then read/write, fetch operations only related with I/O devices.

6) WR: active low signal CPU performs Memory or I/O output with operation.

7) HOLD (M1D): If Hold Request is received by a processor, it releases the Control of buses.

8) HLDACK: when request is received it will be acknowledged by processor or with this pin which is active.

Pin description of Maximum Mode:

<table>
<thead>
<tr>
<th>So</th>
<th>S1</th>
<th>S0</th>
<th>Instruction Queue Status Logic Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>No operation</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
<td>First byte of opcode from Queue is fetched</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
<td>Empty the Queue</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td>Subsequent byte from Queue</td>
</tr>
</tbody>
</table>

S0, S1, S2 → Status Signals, defined these are define depending upon their combination signal.

<table>
<thead>
<tr>
<th>S0</th>
<th>S1</th>
<th>S2</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Interrupt acknowledge</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Read data from Input output port</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Write data from I/O Port</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Halt operation</td>
</tr>
</tbody>
</table>
So \( s_1 s_2 \) operation

1 0 0 \( \rightarrow \) Opcode fetch
1 0 1 \( \rightarrow \) memory read
1 1 0 \( \rightarrow \) memory write
1 1 1 \( \rightarrow \) Ideal State

9) \( l_{ock} \) = active low signal when does all interrupt are mask and no request will be received by hold pin.

10) \( R_O \) \( \cap T_2 \) \& \( R_O \) \( \cap T_1 \); These are known as bus priority control, other processors ask the CPU through this pin to release token bus. \( R_O \) \( \cap T_1 \) is having highest priority than \( R_O \) \( \cap T_2 \).

Functional description of 8086 or Architecture of 8086;
8086 divided into two different blocks.
1) One is known as Bus Interface unit where data is transferring is done between processor and memory & input output devices.
2) Execution unit = This unit related with execution of an instruction by fetching opcode the decode it and execute it.

Bus Interface Unit contain = Bus Interface Logic
Segment Register, Memory address logic.

This 6 byte instruction is defined by two status of 1.

Execution Unit contain data & Address registers, ALU & Control Unit.
Registers of 8086:

1) General purpose register
2) Pointer and index register
3) Segment register
4) Instruction pointer
5) Flag

1) General purpose register: There are 4 16-bit general purpose registers, AX, BX, CX, DX (AH, AL, BH, BL, CH, CL, DH, DL). AH is 8-bit higher order.
   AL is 8-bit lower register.
   AX register acts as an accumulator in 8086. BX, CX, DX are used for general purpose operations in addition to general purpose they also act as special purpose register. As a special purpose DX acts as base register for computation of memory addresses.

   In 8086 memory addresses are to be calculated using the contents of Segment register. Only in case of multiprocessing

   CX acts as a counter in case of multiplication instruction if more than one program is executed (if instruction is executed more than one)

2) Pointer and Index register: Four registers DX, SP, BP, SI, DF (Stack pointer (SP), Base pointer (BP), Source index (SI), Destination index (DI)) Segment.

3) Segment register: CS, DS, ES, SS code segment (CS), data segment (DS), extra segment (ES), stack.
Segment (ss) Code Segment of memory holds the instruction code of program data, variables, and constants are given in the program held in data segment of memory.

- Stack Segment: it holds the data & address of subroutine.
- Segment Register points to starting address of memory which is currently used. Maximum capacity of segment may be up to 64 Kbyte.

2) 8086 instruction specify 16-bit memory address but the actual address is of 20-bit because 8086 having 20 Address line and they are calculated using content of segment register and effective memory address.

4) Instruction Pointer: In 8086 instruction pointer act as a program counter it points to the address of next instruction to be executed. It is automatically incremented after execution of an instruction.

5) Status Flag: 8086 contain 16-bit status flag register also known as program status word (PSW).

- Direction flag
- Interrupt flag
- Sign flag
- Auxilary flag
- Parity flag
- Carry flag
- Overflow flag
- When TR is set to zero, the program can run in multistep. If = 1, the program runs in single step.

- Interrupt flag = 1, use to enable INT0. Interrupt flag 0, use INT0 will be disable.

- If it is used in saving operation, the STD instruction is used to activate DF flag.

- Lock signal is used to make an instruction of 8086 non-interruptable. In multiprocessor system, each processor can use the system bus for its operation when processor is operating with an instruction it is not utilizing a system bus. It should not be interrupted by other processors. At that time lock pin become low, which mean active all the hold request received by other processor until previous processor is executed.

1) Addition, Subtraction, Multiplication, Division, Arrangement (no arrangement), series of no delays, led
Addressing mode of 8086 register

1) Register addressing mode: Register to Register data is transferred
2) Immediate addressing mode: operand is specified in the instruction MOV CX, 16 bit data

Remaining 6 addressing mode specifies the location of an operand which is placed in memory location. Memory consist of two part

Starting address & offset (effective address)

1) Direct addressing mode: Direct address will be given MOV AX, [2000H]

2) Register indirect addressing mode: MOV AX [BX]

3) Based addressing mode: MOV AX [BX, Displacement (8, 16 bit)]

Operand offset is the sum of 8 or 16 bit displacement and contents of base register & base pointer.

DX is used as base register in data segment.

BP is used as a stack base register or stack segment.

Indexed Addressing mode: MOV AX, [SI + Displacement]

Operand offset is the effective address in the Sum
Based Indexed Addressing:

\[
\text{Mov AX}[\text{BX + SI}]
\]

\[
[\text{BP + DI}] \\
\text{BX + SI}
\]

\[
\text{Mov AX} [\text{BX}, \text{SI}]
\]

8086 Instruction:

1. Data transfer instructions:
   - Mov R, R
   - Mov R, M
   - Mov R, Date
   - Mov H, Data
   - Mov R/M, M/R

Memory related with based index mode, based indirect addressing mode, base addressing mode, direct addressing mode. No flag affected.

Load/Store/Exchange instructions:

1. LEA R, M: This instruction load a word (16 bit data) from specified memory location into specified register.

2. LEA: This instruction load offset address into specified register.

3. LDS R, M: Load 16 bit data from specified memory location into specified register.