Serial Input / Output data using 8085

8085 Microprocessor has two Serial Input/Output pins that are used to read/write one bit data to and from peripheral devices.

SID (Serial Input Data) line

- There is an One bit Input line inside the 8085 CPU (Pin number 5)
- 1 bit data can be externally read and stored using this SID line
- The data that is read is stored in the A7th bit of the Accumulator
- RIM instruction is used to read the SID line

Example Pseudocode:
1) RIM
2) $\text{A7} \leftarrow \text{SID}$

As seen from the figure 1, if the SID line is connected with +5V and RIM instruction is executed, then the Accumulator’s MSB bit will be loaded with a Logic 1 (Fig 2)

$$\begin{array}{c}
1 \\
\end{array} = 80H$$

Fig: 2: Accumulator Content after the RIM instruction is executed

if the SID line is connected with 0V (GND) and RIM instruction is executed, then the Accumulator’s MSB bit will be loaded with a Logic 0 (Fig 3)

$$\begin{array}{c}
0 \\
\end{array} = 00H$$

Fig: 3: Accumulator Content after the RIM instruction is executed
**SOD (Serial Output Data) Line**

- There is a One bit Output port inside the 8085 CPU (Pin number 4
- 1 bit data can be externally written in this port.
- To write data into this port, SIM instruction is used.
- The data that is to be written in this port must be stored in the A7th bit of the Accumulator.
- Bit A6 of the Accumulator is known as SOE (Serial output Enable). This bit Must be set to 1 to enable Serial data output.

Fig: 4

Example Pseudocode:

To write a *logic 1* in this SOD line, Load the accumulator with C0H

\[
\begin{array}{cccccc}
1 & 1 & 1 & 1 & 1 & 1 \\
\end{array} = C0H
\]

Data SOE

Fig: 5: Accumulator Content after the RIM instruction is executed

Pseudocode:

\[
\begin{align*}
A & \leftarrow C0H \\
SIM & \\
SOD & \leftarrow (A7)
\end{align*}
\]

To write a *logic 0* in this SOD line, Load the accumulator with 40H

\[
\begin{array}{cccccc}
0 & 1 & 1 & 1 & 1 & 1 \\
\end{array} = 40H
\]

Data SOE

Fig: 6: Accumulator Content after the RIM instruction is executed
Pseudocode:
A ← 40H
SIM
SOD ← (A7)

Problem:

The SID Pin of an 8085 microprocessor is connected with a 0V/+5V source and the SOD Pin is connected with a LED. Write down the Pseudocode that will read the SID pin of the microprocessor and glow the LED if the SID pin is connected with a +5V source otherwise turn the Led off if the SID Pin is connected with GND (0V).

\[\text{The Solution is not provided here, See the Class lecture for the solution.}\]
The 8085 Programming Model

- The 8085 is an 8-bit microprocessor made by Intel.
- It has:
  - 6 general purpose Programmable registers. There are other Temporary registers. But, they are not available to the programmer for programming.
  - An accumulator
  - A flag register
  - A stack pointer
  - A program counter

<table>
<thead>
<tr>
<th>Accumulator</th>
<th>Flags</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>D</td>
<td>E</td>
</tr>
<tr>
<td>H</td>
<td>L</td>
</tr>
<tr>
<td>Program Counter</td>
<td></td>
</tr>
<tr>
<td>Stack Pointer</td>
<td></td>
</tr>
</tbody>
</table>

The Registers

- The 6 general purpose registers are 8-bits wide each.
  - They are to be used as needed.
  - They are called B, C, D, E, H, and L.
  - They can be used as 16-bit register pairs: BC, DE, HL.

- The accumulator is technically part of the ALU.
  - It is 8-bits wide.
  - It is one of the inputs to every ALU operation.
  - The result of any operation is always stored in it.
  - It is known as Register A.

- There are Five flip-flops known as Flag register that are set (Set to Logic 1) or reset (Set to Logic 0) after an operation.
  - They are Z (zero), CY (carry), S (sign), P (parity) and AC (Auxiliary Carry).
  - These flags are used when the microprocessor tests for data conditions.

❖ See the Class lecture / Book for the detailed explanation of the Flag register
The 8085 Instructions Classification

Instructions:
- Instruction is a binary pattern to perform a specific function
- It is a command to the microprocessor to perform a given task

Since the 8085 is an 8-bit device it can have up to $2^8$ (256) instructions.

However, the 8085 only uses 246 combinations that represent a total of 74 instructions.
- Most of the instructions have more than one format.

These instructions can be grouped into five different Classes:

1. Data Transfer Operations
2. Arithmetic Operations
3. Logic Operations
4. Branch Operations
5. Machine Control Operations

Details are not provided here. See the Class lecture / Book (Page 34-36, Section: 2.2) for the detailed explanation of this instruction classes.

Addressing Modes

The microprocessor has different ways of specifying the data for the instruction. These are called “addressing modes”.

The 8085 has Five addressing modes:

1) **Implied Addressing mode:** In this type of addressing mode, No operand (register or data) is specified in the instruction. The operand is inherent to the instruction.
   Example: CMA (Complement Accumulator), SIM, RIM etc

2) **Immediate Addressing Mode:** In this type of addressing mode, immediate data byte is provided with the instruction. Example: MVI A 47H, MVIB CFH etc.

3) **Direct Addressing Mode:** In this type of addressing mode, the 16bit memory address is directly provided with the instruction.
   Example: LDA C5 00 etc

4) **Indirect Addressing Mode:** In this type of addressing mode, the 16bit memory address is indirectly provided with the instruction using a register pair.
   Example: LDAX B
   (Load the accumulator with the contents of the memory location whose address is stored in the register pair BC)

5) **Register Addressing mode:** This type of addressing mode specifies register or register pair that contains data.
   Example: ADD B, MOV B A
8085 Instruction Set

Study the use of the following 8085 instructions

Data Transfer Instructions

**MOV reg, reg**
- i.e. MOV B A

**MVI reg 8bit data (immediate data)**
- i.e. MVI A 32H

**LDA 16bit address**
- i.e. LDA C0 05
  - Load the Accumulator with the content of the memory location specified by 16bit address

**STA 16bit address**
- i.e. STA C0 05
  - Store the content of Accumulator to the Memory location specified by the 16bit address

**LDAX Rp ( Rp = Register pair)**
- i.e. LDAX B
  - (Load the accumulator with a data contained in a memory location where the 16bit memory location is stored in the BC register pair)

**STAX Rp ( Rp = Register pair)**
- i.e. STAX B
  - (Store the content of the accumulator to a memory location where the 16bit memory location is stored in the BC register pair)

**IN portaddr**
- i.e. IN 00 (Reads data from the Input Switch, 00 represents the port address of the input switch)

**OUT portaddr**
- i.e. OUT 00 (Writes data to the Display device where 00 represents the Port address of the display)

Arithmetic Instructions

**Addition**

**ADD reg**
- i.e. ADD B
  - \([A] \leftarrow [A] + [B]\)

**ADI 8bit data**
- i.e. ADI 47H
  - \([A] \leftarrow [A] + 47H\)
**Subtraction**

**SUB reg**
- i.e. SUB C
  
  \[ A \leftarrow [A] - [C] \]

**SUI reg**
- i.e. SUI 5F;
  \[ A \leftarrow [A] - 5F \]

**Increment/Decrement**

**INR reg**
- Increment the Content of a Register specified by reg
  - i.e: INR B
    - Increments the Content of Register B by 1

**DCR reg**
- Decrement the Content of a Register specified by reg
  - i.e: DCR D
    - Decrements the Content of Register D by 1

**INX Rp**
- Increment the Content of a Register pair specified by Rp
  - i.e: INX B
    - Increments the Content of Register pair BC by 1

**DCX Rp**
- Decrement the Content of a Register pair specified by Rp
  - i.e: DCX B
    - Decrement the Content of Register pair BC by 1

**Logical Instructions**

**ORA reg**

**ORI 8bit data**

**ANA reg**

**ANI 8bit data**

**XRA reg**

**XRI 8bit data**

**CMA**
- Completes the Content of the Accumulator (Performs the NOT Operation)

**CMP reg**
- Compares the Content of the Accumulator with the register specified by ‘reg’
  - If \( B > A \), then Carry flag is set
  - If \( B = A \), then Zero flag is set
  - If \( A > B \), then No flag is set

**CPI 8bit data**
- Compares the Content of the Accumulator with the immediate data specified by the 8bit data
Branching Instructions

**JMP** 16 bit address  [Unconditional Jump]
The program sequence jumps unconditionally to a memory location defined by 16bit address

**JC** 16 bit address  [conditional Jump]
The program sequence jumps to a memory location defined by 16bit address if the Carry Flag is set

**JNC** 16 bit address  [conditional Jump]
The program sequence jumps to a memory location defined by 16bit address if the Carry Flag is not set

**JZ** 16 bit address  [conditional Jump]
The program sequence jumps to a memory location defined by 16bit address if the Zero Flag is set

**JNZ** 16 bit address  [conditional Jump]
The program sequence jumps to a memory location defined by 16bit address if the Zero Flag is not set

Machine Control Instructions

**HLT**

**NOP**

odynamicons For Details, See Section 2.5 (Page number: 46-50 (Fifth Edition) of your Text Book)

OPCode Formatting:

odynamicons See Section 2.3.2 (Page number: 39-41 (Fifth Edition) of your Text Book)
16 bit Operations

Memory location C0F0, C0F1, C0F2 contains 05H, 07H, 09H data. Write a program to display these three data one by one.

```
MVI H 03H
MVI D C0H
MVI E F0H
LDAX D
OUT 00
Add a DELAY here
INX D
DCR H
JNZ
HLT
```

<table>
<thead>
<tr>
<th>Memory Location</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>C0F0</td>
<td>42H</td>
</tr>
<tr>
<td>C0F1</td>
<td>47H</td>
</tr>
<tr>
<td>C0F2</td>
<td>A9H</td>
</tr>
</tbody>
</table>

To Load Register Pair with immediate data

LXI Rp 16bit Memory address

i.e. LXI H C0F0

Loads the HL register pair with C050 data, Register H is loaded with C0 and Register L is loaded with data F0

Similar Examples: LXI B C0F0
LXI D C056

There is no LDAX / STAX instruction for HL Register Pair

To Load Register Pair HL, MOV instruction is used

```
MOV A M : A \rightarrow ((HL))
MOV M A : ((HL)) \rightarrow A  [ Equivalent to STAX Rp instruction ]
```

8085 Advanced Instruction

16bit Data transfer

LHLD 16bit address

Example:
```
LHLD C050
Load data from C050 memory location to Register L and Loads data from C051 memory location to Register H
```
SHLD 16bit address

**Example:**

\[
\text{SHLD C050}
\]
Stores data of *Register L* to memory location C050.  
and Stores data of *Register H* to memory location C051.

**XCHG**

Exchanges the content of HL and DE Register Pair

![XCHG Diagram]

**Problem:** Memory Location 2050\(^H\) and 2051\(^H\) contains 3F\(^H\) and 47\(^H\) respectively. Register pair DE contains 856F\(^H\). Write assembly code to exchange the content of DE with the content of the memory locations

Soln:

\[
\text{LHLD 2050} \\
\text{XCHG} \\
\text{SHLD 2050}
\]

*Study the following problems and solve them*

**Problem 1:** There are 10 data stored in 10 consecutive memory locations. Write 8085 assembly code to find the maximum of these 10 numbers and display the maximum number in the display device.

**Problem 2:** There are 5 data stored in 5 consecutive memory locations starting from C0F0. Write 8085 assembly code to copy these data bytes in reverse order to some different memory location starting from CF10.

**Problem 3:** There are 10 data stored in 10 consecutive memory locations. Write 8085 assembly code to display the numbers that are in the range 50\(^H\)-70\(^H\)
16 bit addition instructions

DAD Rp: Adds the content of the HL register with the content of the Register pair specified by Rp and stores the result in the HL register pair.

Example:
DAD B
Adds the content of the HL register with the content of the BC register pair stores the result in the HL register pair.

\[ \text{HL} \leftarrow \text{HL} + \text{BC} \]

<table>
<thead>
<tr>
<th>H</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>41</td>
<td>01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>06</td>
<td>02</td>
</tr>
</tbody>
</table>

DAD B will result in:

<table>
<thead>
<tr>
<th>H</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>47</td>
<td>03</td>
</tr>
</tbody>
</table>

What DAD B does:

\[ \begin{align*}
4101 & \quad \text{[HL]} \\
+0602 & \quad \text{[DE]} \\
\hline
4703 & \quad 0100 0111 0000 0011
\end{align*} \]

Rotating instructions:

RLC
RRC
RAL
RAR

RLC (Rotate Accumulator Left)

Every bit of the Accumulator is shifted one bit left and the MSB bit of the Accumulator is copied into the Carry flag and into the A0th bit.

After the RLC instruction is executed the content of the Accumulator and the carry flag will be
RRC (Rotate Accumulator Right)

Every bit of the Accumulator is shifted one bit Right and the LSB bit of the Accumulator is copied into the Carry flag and into the A7th bit.

After the RRC instruction is executed the content of the Accumulator and the carry flag will be

RAL (Rotate Accumulator Left through Carry)

Every bit of the Accumulator is shifted one bit left and the MSB bit of the Accumulator is copied into the Carry flag and the Carry flag value is copied into the A0th bit.

After the RAL instruction is executed the content of the Accumulator and the carry flag will be
RAR (Rotate Accumulator Right through Carry)

Every bit of the Accumulator is shifted one bit Right and the LSB bit of the Accumulator is copied into the Carry flag and the Carry flag value is copied into the A\textsubscript{7}th bit

After the RAR instruction is executed the content of the Accumulator and the carry flag will be