MICROPROCESSOR & INTERFACING LAB

(EE-329-F)

LABORATORY MANUAL

V - SEMESTER

Prepared by
VIKRANT VERMA(A.P.)

BRCM College of Engineering and Technology

Department of Electrical & Electronics &Engineering
Bahal, Bhiwani - 127028
# LIST OF EXPERIMENTS

<table>
<thead>
<tr>
<th>S.No</th>
<th>Name of Experiments</th>
<th>Page No</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Introduction of microprocessor 8085 trainer kit – 85AD</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>Write an ALP to perform the addition of two 8 bit numbers.</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>Write an ALP to perform the subtraction of two 8 bit numbers.</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>Write an ALP to perform the addition with carry of two 8 bit numbers.</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>Write an ALP to perform the subtraction with borrow of two 8 bit numbers.</td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td>Write an ALP to perform the addition of two BCD numbers.</td>
<td></td>
</tr>
<tr>
<td>7.</td>
<td>Write an ALP to perform the subtraction of two BCD numbers.</td>
<td></td>
</tr>
<tr>
<td>8.</td>
<td>Write an ALP to perform the multiplication of two 8 bit numbers by repeated addition method.</td>
<td></td>
</tr>
<tr>
<td>9.</td>
<td>Write an ALP to perform the multiplication of two 8 bit numbers by bit rotation method.</td>
<td></td>
</tr>
<tr>
<td>10.</td>
<td>Write an ALP to perform the division of two 8 bit numbers by repeated addition method.</td>
<td></td>
</tr>
<tr>
<td>11.</td>
<td>Write an ALP to perform the division of two 8 bit numbers by bit rotation method.</td>
<td></td>
</tr>
<tr>
<td>12.</td>
<td>Write an ALP to find the square of given numbers in array.</td>
<td></td>
</tr>
<tr>
<td>13.</td>
<td>Write an ALP to find largest number in an array.</td>
<td></td>
</tr>
<tr>
<td>14.</td>
<td>Study of 8086 microprocessor kit</td>
<td></td>
</tr>
<tr>
<td>15.</td>
<td>Write an ALP to perform addition of two 16 bit numbers.</td>
<td></td>
</tr>
</tbody>
</table>
EXPERIMENT NO.1

AIM:

To study about introduction of microprocessor 8085 trainer kit - 85AD.

APPARATUS REQUIRED:

8085 - Microprocessor kit.

THEORY:

The system has got 8085 as the Central Processing Unit. The clock frequency for the system is 3.0MHz and is generated from a crystal of 6.14MHz. 8085 has got 8 bit data lines and 16 bit address lines. The lower 8 address lines and 8 bit data lines are multiplexed. Since the lower 8 address bits appear on the bus during the first clock cycle of a machine cycle and the 8 bit data appears on the bus during the 2nd and 3rd clock cycle, it becomes necessary to latch the lower 8 address bits during the first clock cycle so that the 16 bit address remains available in subsequent cycles. This is achieved using a latch 74LS373.

The training kit which we are going to use in this lab is STUDENT-85AD which communicates with the outside world through a general purpose IBM PC Compatible ASCII keyboard and 16x2 Liquid Crystal Display (LCD). The kit also has the capability of interfacing with CRT terminal through the interface provided on the board.

The on board resident system monitor software is very powerful and provides various software utilities. The kit provides support for powerful software commands like INSERT, DELETE, BLOCK MOVE, RELOCATE, STRING, FILL and MEMORY COMPARE etc. The kit is configured around the internationally adopted standard STD bus which is most popular bus for process control and real time applications. All the address, data and control lines are available at the edge connector. The kit is fully expandable for any kind of application.

MEMORY:

8085 kit provides 8/32K bytes of RAM using 6264/62256 chip and 8K bytes of EPROM for monitor. There is one memory space provided on kit. This one space can be defined any address slots from 8000 - DFFF depending upon the size of the memory chip to be used. Total onboard memory can be extended to 64K bytes.

I/O DEVICES

The various I/O chips used in STUDENT-85AD microprocessor kit are 8255, 8253 & 8155. The functional role of all these chips is given below:

8255(Programmable Peripheral Interface)

8255 is a programmable peripheral interface (PPI) designed to use with 8085 Microprocessor. This basically acts as a general purpose I/O device to interface peripheral equipments to the system bus. It is not necessary to have an external logic to interface with peripheral devices since the functional configuration of 8255 is programmed by the system software. It has got three Input/Output
ports of 8 lines each (PORT-A, PORT-B & PORT-C). Port C can be divided into two ports of 4 lines each named as Port C upper and Port C lower. Any Input output combination of Port A, Port B, Port C upper and lower) can be defined using the appropriate software commands. The kit provides 24 Input/output ports using 8255 chips.

8253 (Programmable Internal Timer)

This chip is a programmable interval Timer/Counter and can be used for the generation of accurate time delays under software control. Various other functions that can be implemented with this chip are programmable rate generator, Even Counter, Binary rate Multiplier, Real Time Clock etc. This chip has got three in dependent 16 bit counters each having a count rate of up to 2 KHz. The first Timer/Counter (i.e. Counter 0) is being used for Single Step operation. However, its connection are also brought at connector space C4. For single step operation CLKO signal of Counter 0 is getting a clock frequency of 1.535 MHz. The counter 1 is used to generate clock for 8251. Counter 1 & Counter 2 are free for the user. Clock for the CLK1, CLK2 is to be given externally.

8155 (Programmable I/O Port & Timer Interface) – Optional

8155 is a programmable I/O ports and timer interface designed to use with 8085 Microprocessor. The 8155 includes 256 bytes of R/W memory, three I/O ports and a Timer. This basically acts as a general purpose I/O device to interface peripheral equipments to the system bus. It is not necessary to have an external logic to interface with peripheral devices since the functional configuration of 8155 is programmed by the system software. It has got two 8-bit parallel I/O port (Port-A, Port-B) and one 6-bit (Port-C). Ports A & B also can be programmed in the handshake mode, each port using three signals as handshake signals from Port-C. The timer is a 14 bit down counter and has four modes.

<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>Command</th>
<th>Description</th>
<th>Command Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>M</td>
<td>Examine/ Modify Memory</td>
<td>[M]&lt;ADDRESS&gt;[DATA] [DATA]…[,]</td>
</tr>
<tr>
<td>2.</td>
<td>E</td>
<td>Enter a memory block</td>
<td>[E]&lt;ADDRESS&gt;[DATA] [DATA]…[,]</td>
</tr>
<tr>
<td>3.</td>
<td>R</td>
<td>Examine/ Modify register</td>
<td>[R] &lt;REG. IDENTIFIER&gt; [$]</td>
</tr>
<tr>
<td>4.</td>
<td>S</td>
<td>Single Step</td>
<td>[S] &lt;Starting Address&gt;[,]</td>
</tr>
<tr>
<td>5.</td>
<td>G</td>
<td>Go</td>
<td>[G] &lt; Starting Address&gt;$</td>
</tr>
<tr>
<td>6.</td>
<td>B</td>
<td>Block Move</td>
<td>[B]&lt; Starting Address of source&gt;[,]&lt; End Address of source &gt;[,]&lt; Starting Address of destination&gt;[,$]</td>
</tr>
<tr>
<td>7.</td>
<td>I</td>
<td>Insert</td>
<td>[I]&lt; Starting Address of the program&gt;[,]&lt; End Address of the program &gt;[,]&lt;Address from where the byte or bytes are to be entered&gt;[,]&lt;No. of bytes&gt;[,][DATA][$,]</td>
</tr>
<tr>
<td></td>
<td>Command</td>
<td>Description</td>
<td>Details</td>
</tr>
<tr>
<td>---</td>
<td>---------</td>
<td>-------------</td>
<td>---------</td>
</tr>
<tr>
<td>8.</td>
<td>D</td>
<td>Delete</td>
<td>[D]&lt;Starting Address of the program&gt;[,]&lt;Ending address of the program&gt;[,]&lt;starting address from where the bytes are to be deleted&gt;[,]&lt;Ending address till where the bytes are to be deleted&gt;[$]</td>
</tr>
<tr>
<td>9.</td>
<td>N</td>
<td>Insert Data</td>
<td>[N]&lt; Starting Address of the program/ data area &gt;[,] &lt;Ending Address of the program&gt; [,] &lt;Starting Address at which the bytes are to be entered&gt;,&lt;No. of bytes&gt; [,][DATA][,.][.][$]</td>
</tr>
<tr>
<td>10.</td>
<td>O</td>
<td>Delete Data</td>
<td>[O]&lt; Starting Address of the program or data area&gt; [,] &lt;End Address of the program/ data area&gt; [,] &lt;Starting Address from where the deletion should start&gt; [,] &lt;End address till where bytes are to be deleted&gt;[$]</td>
</tr>
<tr>
<td>11.</td>
<td>F</td>
<td>Fill</td>
<td>[F]&lt; Starting Address of program/ data area&gt; [,] &lt;End Address&gt; [,] &lt;Constant to be filled&gt;[$]</td>
</tr>
<tr>
<td>12.</td>
<td>H</td>
<td>Relocate</td>
<td>[H]&lt; Starting Address of the program&gt; [,] &lt;End Address of the program&gt; [,] &lt;Destination Address&gt;[$]</td>
</tr>
<tr>
<td>13.</td>
<td>J</td>
<td>Memory Compare</td>
<td>[J]&lt; Starting Address of the first block&gt; [,] &lt;End Address of the first block&gt; [,] &lt;Starting Address of second block&gt;[$]</td>
</tr>
<tr>
<td>14.</td>
<td>K</td>
<td>String</td>
<td>[K]&lt; Starting Address of the program&gt; [,] &lt;End address of the program&gt; [,] &lt;Address of the location at which first byte of the string lies&gt; [,] &lt;Address of the location at which last byte of the string lies&gt;[$]</td>
</tr>
</tbody>
</table>

*Table: 1 Details of the commands used in STUDENT-85AD*
<table>
<thead>
<tr>
<th>Sr. No.</th>
<th>Register Identifier</th>
<th>Register Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>A</td>
<td>Register A or accumulator</td>
</tr>
<tr>
<td>2.</td>
<td>B</td>
<td>Register B</td>
</tr>
<tr>
<td>3.</td>
<td>C</td>
<td>Register C</td>
</tr>
<tr>
<td>4.</td>
<td>D</td>
<td>Register D</td>
</tr>
<tr>
<td>5.</td>
<td>E</td>
<td>Register E</td>
</tr>
<tr>
<td>6.</td>
<td>F</td>
<td>Register F</td>
</tr>
<tr>
<td>7.</td>
<td>I</td>
<td>Interrupt Mask Register</td>
</tr>
<tr>
<td>8.</td>
<td>H</td>
<td>Register H</td>
</tr>
<tr>
<td>9.</td>
<td>L</td>
<td>Register L</td>
</tr>
<tr>
<td>10.</td>
<td>S</td>
<td>Stack Pointer MSB</td>
</tr>
<tr>
<td>11.</td>
<td>P</td>
<td>Program Counter MSB</td>
</tr>
</tbody>
</table>

Table: 2 Details of the Register Identities used in STUDENT-85AD

Modes of Operation:

In STUDENT-85AD there is an onboard facility of assembler/ disassemble. On pressing ‘1’ key, STUDENT-85AD comes in assembler/ disassemble mode depending upon the next key you press i.e.:

‘A’- Assembler mode
‘C’- Disassembler mode
**Assembler Mode:**

On pressing the key ‘A’ STUDENT-85AD comes into the assembler mode. As soon as ‘A’ is pressed, kit asks RAM address. This will be the starting address of the program to be entered. After entering the starting address, press <CR> key, it displays the entered starting address in the upper line of the LCD screen. Now it waits for the mnemonics entry.

One can enter all the valid mnemonics of 8085 and the Pseudo commands. If the entered alphabets do not form a valid mnemonics or a Pseudo command, the carriage goes to same line and prints the address of the previous line. Hence the entry of the wrong mnemonic is indicated by giving the same line to the user.

Entry of a space completes one field of entry; and processing of that field is done immediately by the command. By field, we mean, mnemonic as one field, operand or label as another field.

Using this mode one can write or feed the program using assembly language mnemonics.

**Format:**

```
The display will show

| RAM ADRR: |
```

Type the desired RAM ADDRESS e.g. 2000, the display looks like figure below

| RAM ADRR: 2000 |

(Note: RAM Address is always a16-bit hexadecimal number.)

Press <ENTER> and the display will show the RAM Address blank against it.

| 2000: |

One can type the command mnemonics in blank space. After that on pressing ENTER or SPACE key, the RAM Address will be automatically incremented. In case of invalid mnemonics the monitor software will erase the mnemonic and will remain on the same RAM Address until the command typed is a valid mnemonic. Then the display will appear like figure below:

| 2000: MOV B, A |

In this manner one can type all a whole program which automatically gets stored in the RAM of the kit and RUN the program using ‘GO’ command thereafter.

**DISASSEMBLER MODE:**

Disassembler mode can be selected by pressing the ‘1’ and then ‘C’. This command disassembles the program as specified by the STARTING address and END address. In case one
wants to proceed further, press <CR> key, otherwise <Esc> key will exit from the disassemble mode.

MEMORY MAPPING:

STUDENT-85AD kit provides 8/32 KB of RAM and 16 KB of EPROM. The total onboard memory can be expanded to 64 KB. For the system operation the monitor should start from address 0000H. A minimum of 8KB RAM should be there on the board with starting address as 2000H.

INPUT/OUTPUT MAPPING:

<table>
<thead>
<tr>
<th>Device</th>
<th>Active range Port Addresses</th>
<th>Port Numbers</th>
<th>Selected Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>8255-I</td>
<td>00-07</td>
<td>00 and 04</td>
<td>PPI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>01 and 05</td>
<td>Port-A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>02 and 06</td>
<td>Port-B</td>
</tr>
<tr>
<td></td>
<td></td>
<td>03 and 07</td>
<td>Control Word</td>
</tr>
<tr>
<td>8255-II</td>
<td>08-0B</td>
<td>08</td>
<td>PPI</td>
</tr>
<tr>
<td></td>
<td></td>
<td>09</td>
<td>Port-A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0A</td>
<td>Port-B</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0B</td>
<td>Control Word</td>
</tr>
<tr>
<td>8253</td>
<td>10-17</td>
<td>10 and 14</td>
<td>Counter 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>11 and 15</td>
<td>Counter 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12 and 16</td>
<td>Counter 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>13 and 17</td>
<td>Control Word</td>
</tr>
<tr>
<td>LCD</td>
<td>38-3F</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Developing/Debugging Software:

STUDENT-85AD kit provides software features like Relocate, String, Insert, Delete, Assembler, Disassembler, Programming etc. which find extensive application in developing/debugging software. The various steps involved in developing software are:

1. Define the problem in the form of a flow chart.
2. Write the program in Assembly Language of 8085.
3. Assemble the program through Assembler command.
4. Enter the program in RAM area and RUN it.

It is likely that the program may not run in one shot because some mistakes can be there in it. The process of finding these mistakes and removing them is called the debugging of the program.

One way of entering the program is in HEX code of the mnemonics and the other way is through assembler. In assembler mode you can write the program in mnemonics form and inspect the disassembly form with its HEX code.

One way of finding the mistakes in the program is to run the program in single instruction mode and after each step compare what the program is doing and what is it supposed to do. In the
process of this one might have to examine the contents of the memory locations or the content of internal registers after the execution of each instruction.

During this process of debugging, at some time user might just like to examine the status of the program at a particular point. If this point is near the beginning of the program, one can reach this point by single instruction facility. But if the point is quite far from the beginning of the program, it is time saving to make use of BREAK POINT facility. For this introduce a RST5 instruction (EF) at the point to be examined and run the program at full speed using ‘GO’ command. When during the execution of the program, this instruction is encountered; the control of the processor is transferred to the monitor. The monitor saves the user registers and displays a sign ‘STUDENT-85AD’ on the LCD screen. No one can examine the status of any memory location or any internal register. One can change the content of memory location or register if necessary.

Sometimes while debugging user may find that certain instructions are to be added to the program or to be deleted from the program. The program written for one memory area can be made operative for some other area using the RELOCATE command.

Sometimes it is required to execute the program in single cycle mode. For this one can make use of single cycle facility (optional) on the board of STUDENT-85AD kit.
EXPERIMENT NO. 2

Aim

Write a well-documented program using 8085 for addition of two 8-bit numbers.

Apparatus

8085 microprocessor kit (STUDENT-85AD), ASCII (PS-2) keyboard.

Theory :-

2501H is the address of memory location for the 1st number. 2nd number is stored at next memory location. By the virtue of the program given below we first move the 1st number into accumulator register then add the 2nd number to this and use the next memory location i.e. 2503h to store the result. The program given below is self explanatory.

Program:-

<table>
<thead>
<tr>
<th>MEMORY</th>
<th>MACHINE</th>
<th>MNEMONICS</th>
<th>OPERANDS</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>21,01,25</td>
<td>LXI</td>
<td>H,2501</td>
<td>Get address of 1st no. in HL pair</td>
</tr>
<tr>
<td>2003</td>
<td>7E</td>
<td>MOV</td>
<td>A,M</td>
<td>Move 1st no. in accumulator</td>
</tr>
<tr>
<td>2004</td>
<td>23</td>
<td>INX</td>
<td>H</td>
<td>HL points the address 2502H</td>
</tr>
<tr>
<td>2005</td>
<td>86</td>
<td>ADD</td>
<td>M</td>
<td>Add the 2nd no.</td>
</tr>
<tr>
<td>2006</td>
<td>23</td>
<td>INX H</td>
<td></td>
<td>HL points 2503H</td>
</tr>
<tr>
<td>2007</td>
<td>77</td>
<td>MOV</td>
<td>M,A</td>
<td>Store result in 2503H.</td>
</tr>
<tr>
<td>2008</td>
<td>CF</td>
<td>RST 1</td>
<td></td>
<td>Terminate</td>
</tr>
</tbody>
</table>

Procedure:-

1. Connect the kit to the power supply.
2. Connect the PS-2 keyboard to the PS-2 (Female) connector CN-7 of the kit.
3. Switch ON the kit.
4. LCD displays “STUDENT-85”, if not so check power connections.
5. If backlight is glowing and display shows something else press the RESET key provided on the kit. If still the persists contact Lab staff.
6. To Feed the program and data in the RAM, for machine language follow the steps 7 to 9 and for assembly language programming steps 10 to 14.
8. The LCD display will show the address and its content. Modify the content according to the program using keyboard then press [ENTER] or [SPACEBAR]. The content will be saved and display will move the next address. Modify if required.
9. Use UP/DOWN arrow keys move between the memories locations.
(11) The display will show ‘RAM ADR: [BLANK]’, asking for the starting address which is 2000H in the above program.

(12) Type [2000] and press [ENTER].

(13) Display will show ‘2000: [BLANK]’. Type the mnemonics of the command operand etc. and press [ENTER].

(14) The display will show the next location. Type desired mnemonics and operand. Keep on doing this until the end of the program.

(15) Feed the data using the [M] command.

(16) To RUN/ EXECUTE the program Press [G]<Starting Address>$[].

(17) Go to memory locations used for result storage using [M] command and verify the results.

**Figure:** Flow-Chart of the Process

```
START
Get the first No.
Get the second No.
Add. Two numbers
Store the result
END
```

**Input Data:**
- 2501- 67h
- 2502- 22h

**Output Data:**
- 2503- 89h

**Precautions:-**

(1) Make sure proper handling of equipments/ kits.

(2) Make sure that all the machine codes/ mnemonics are as per the program.

**Result:-**

Addition of 67H and 22H is 89H. Hence experiment for the addition of two 8-bit numbers has been successfully performed.
EXPERIMENT NO. 3

Aim:-

Write a program using 8085 for subtraction of two 8-bit numbers.

Apparatus:-

8085 microprocessor kit (STUDENT-85AD), ASCII (PS-2) keyboard.

Theory:-

2501h is the address of memory location for the 1st number. 2\textsuperscript{nd} number is stored at next memory location. By the virtue of the program given below we first move the 1st number into accumulator register then subtract the 2\textsuperscript{nd} number from this and use the next memory location i.e. 2503h to store the result. The program given below is self explanatory.

Program: -

<table>
<thead>
<tr>
<th>Memory address</th>
<th>Opcode</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>21,01,25</td>
<td>LXI</td>
<td>H, 2501</td>
<td>Get address of 1st no. in HL pair</td>
</tr>
<tr>
<td>2003</td>
<td>7E</td>
<td>MOV</td>
<td>A, M</td>
<td>Move 1st no. in accumulator</td>
</tr>
<tr>
<td>2004</td>
<td>23</td>
<td>INX</td>
<td>H</td>
<td>HL points 2502H.</td>
</tr>
<tr>
<td>2005</td>
<td>96</td>
<td>SUB</td>
<td>M</td>
<td>Subtract 2nd no. from 1st no.</td>
</tr>
<tr>
<td>2006</td>
<td>23</td>
<td>INX</td>
<td>H</td>
<td>HL points 2503H.</td>
</tr>
<tr>
<td>2007</td>
<td>77</td>
<td>MOV</td>
<td>M, A</td>
<td>Move contents of acc. to memory</td>
</tr>
<tr>
<td>2008</td>
<td>CF</td>
<td>RST 1</td>
<td></td>
<td>Stop</td>
</tr>
</tbody>
</table>

Procedure:-

1. Connect the kit to the power supply.
2. Connect the PS-2 keyboard to the PS-2 (Female) connector CN-7 of the kit.
3. Switch ON the kit.
4. LCD displays “STUDENT-85”, if not so check power connections.
5. If backlight is glowing and display shows something else press the RESET key provided on the kit. If still the persists contact Lab staff.
6. To feed the program and data in the RAM, for machine language follow the steps 7 to 9 and for assembly language programming steps 10 to 14.
8. The LCD display will show the address and its content. Modify the content according to the program using keyboard then press [ENTER] or [SPACEBAR]. The content will be saved and display will move the next address. Modify if required.
9. Use UP/DOWN arrow keys move between the memory locations.
(11) The display will show ‘RAM ADR: [BLANK]’, asking for the starting address which is 2000H in the above program.

(12) Type [2000] and press [ENTER].

(13) Display will show ‘2000: [BLANK]’. Type the mnemonics of the command operand etc. and press [ENTER].

(14) The display will show the next location. Type desired mnemonics and operand. Keep on doing this until the end of the program.

(15) Feed the data using the [M] command.

(16) To RUN/EXECUTE the program Press [G]<Starting Address>$.

(17) Go to memory locations used for result storage using [M] command and verify the results.

**Flow-Chart of the process :-**

```
Start

Get 1\textsuperscript{st} number from memory

Get 2\textsuperscript{nd} number from memory

Subtract 2nd number from the 1st

Store result

Stop
```
**Input Data:**

<table>
<thead>
<tr>
<th>Case</th>
<th>2501h-67h, 2502h-22h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case-II</td>
<td>2501h-22h, 2502h-67h</td>
</tr>
</tbody>
</table>

**Output Data:**

<table>
<thead>
<tr>
<th>Case</th>
<th>2503h-45h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case-II</td>
<td>2503h-BBh</td>
</tr>
</tbody>
</table>

**Precautions:**

1. Make sure proper handling of equipments/ kits.
2. Make sure that all the machine codes/ mnemonics are as per the program.

**Result:**

Subtraction of 22h from 67h is 45h. If we are subtracting a bigger number from a smaller number i.e. 22h-67h, then result is two’s complement of 45h i.e. BBh. Hence experiment for the subtraction of two 8-bit numbers has been successfully performed.
EXPERIMENT NO. 4

Aim:-

Write a well-documented program using 8085 for addition of two 8-bit numbers with carry.

Apparatus:-

8085 microprocessor kit (STUDENT-85AD), ASCII (PS-2) keyboard.

Theory:-

2501h is the address of memory location for the 1st number, 2nd number is stored at next memory location. By the virtue of the program given below we first move the 1st number into accumulator register then add the 2nd number to this and if a carry is there we use C register to store carry. We use next two memory locations i.e. 2503h & 2504h to store the result and carry respectively. The program given below is self explanatory.

Program:-

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Machine Code</th>
<th>Labels</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>21,01,25</td>
<td>LXI</td>
<td>H,2501</td>
<td></td>
<td>Get address of 1st no. in HL pair</td>
</tr>
<tr>
<td>2003</td>
<td></td>
<td>MVI</td>
<td>C,00</td>
<td></td>
<td>MSB of sum, Initial value=00</td>
</tr>
<tr>
<td>2005</td>
<td>7E</td>
<td>MOV</td>
<td>A,M</td>
<td></td>
<td>Move 1st no. in accumulator</td>
</tr>
<tr>
<td>2006</td>
<td>23</td>
<td>INX</td>
<td>H</td>
<td></td>
<td>HL points the address 2502H</td>
</tr>
<tr>
<td>2007</td>
<td>86</td>
<td>ADD</td>
<td>M</td>
<td></td>
<td>Add the 2nd no.</td>
</tr>
<tr>
<td>2008</td>
<td>D2,0D,25</td>
<td>JNC</td>
<td>AHEAD</td>
<td></td>
<td>If no carry jump to AHEAD</td>
</tr>
<tr>
<td>200B</td>
<td>0C</td>
<td>INR</td>
<td>C</td>
<td></td>
<td>If carry increment C</td>
</tr>
<tr>
<td>200C</td>
<td>32,03,25</td>
<td>AHEAD</td>
<td>STA</td>
<td>2503h</td>
<td>Store result at 2503h.</td>
</tr>
<tr>
<td>200F</td>
<td>79</td>
<td>MOV</td>
<td>A,C</td>
<td></td>
<td>Get carry in acc.</td>
</tr>
<tr>
<td>2010</td>
<td>32,04,25</td>
<td>STA</td>
<td>2504h</td>
<td></td>
<td>Store carry at 2504h</td>
</tr>
<tr>
<td>2013</td>
<td>76</td>
<td>HLT</td>
<td></td>
<td></td>
<td>Stop</td>
</tr>
</tbody>
</table>

Procedure:-

(1) Connect the kit to the power supply.
(2) Connect the PS-2 keyboard to the PS-2 (Female) connector CN-7 of the kit.
(3) Switch ON the kit.
(4) LCD displays “STUDENT-85”, if not so check power connections.
(5) If backlight is glowing and display shows something else press the RESET key provided on the kit. If still the persists contact Lab staff.
(6) To Feed the program and data in the RAM, for machine language follow the steps 7 to 9 and for assembly language programming steps 10 to 14.
(7) Press [M]<Starting Address >[ENTER]. Starting address here is 2000H, so syntax will be
(8) The LCD display will show the address and its content. Modify the content according to the program using keyboard then press [ENTER] or [SPACEBAR]. The content will be saved and display will move the next address. Modify if required.

(9) Use UP/DOWN arrow keys move between the memory locations.

(10) Press [1][A] to select assembler mode.

(11) The display will show ‘RAM ADR: [BLANK]’, asking for the starting address which is 2000H in the above program.

(12) Type [2000] and press [ENTER].

(13) Display will show ‘2000: [BLANK]’. Type the mnemonics of the command operand etc. and press [ENTER].

(14) The display will show the next location. Type desired mnemonics and operand. Keep on doing this until the end of the program.

(15) Feed the data using the [M] command.

(16) To RUN/EXECUTE the program Press [G]<Starting Address>[$].

(17) Go to memory locations used for result storage using [M] command and verify the results.

**Flow-Chart of the process:**

1. **Start**
2. Get 1\(^{st}\) number from memory
3. Get 2\(^{nd}\) number from memory
4. Add 2\(^{nd}\) number to the 1st
5. Is there a Carry?
   - If No, go back to step 4
   - If Yes, Increment C reg.
8. Store result
9. **Stop**
**Input Data:**

2501- 67h  
2502- 22h

**Output Data:**

2503- 89h

**Precautions:-**

(1) Make sure proper handling of equipments/ kits.

(2) Make sure that all the machine codes/ mnemonics are as per the program.

**Result:-**

Addition of 67h and 22h is 89h. Hence experiment for the addition of two 8-bit numbers has been successfully performed.
EXPERIMENT NO. 5

Aim:
Write a well-documented program using 8085 for subtraction of two 8-bit numbers with carry.

Apparatus:
8085 microprocessor kit (STUDENT-85AD), ASCII (PS-2) keyboard.

Theory:
2501h is the address of memory location for the 1st number. 2nd number is stored at next memory location. By the virtue of the program given below we first move the 1st number into accumulator register then subtract the 2nd number from this and if a borrow is there we use C register to store it. We use next two memory locations i.e. 2503h & 2504h to store the result and borrow respectively. The program given below is self explanatory.

Program:

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Machine Code</th>
<th>Labels</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>21,01,25</td>
<td>LXI</td>
<td>H,2501</td>
<td></td>
<td>Get address of 1st no. in HL pair.</td>
</tr>
<tr>
<td>2003</td>
<td>0E,00</td>
<td>MVI</td>
<td>C,00</td>
<td></td>
<td>MSB of sum, Initial value=00</td>
</tr>
<tr>
<td>2005</td>
<td>7E</td>
<td>MOV</td>
<td>A,M</td>
<td></td>
<td>Move 1st no. in accumulator</td>
</tr>
<tr>
<td>2006</td>
<td>23</td>
<td>INX</td>
<td>H</td>
<td></td>
<td>HL points the address 2502H</td>
</tr>
<tr>
<td>2007</td>
<td>86</td>
<td>SUB</td>
<td>M</td>
<td></td>
<td>Subtract 2nd no. from acc.</td>
</tr>
<tr>
<td>2008</td>
<td>D2,0D,25</td>
<td>JNC</td>
<td>AHEAD</td>
<td></td>
<td>If no borrow jump to AHEAD</td>
</tr>
<tr>
<td>200B</td>
<td>0C</td>
<td>INR</td>
<td>C</td>
<td></td>
<td>If borrow increment C</td>
</tr>
<tr>
<td>200C</td>
<td>32,03,25</td>
<td>AHEAD</td>
<td>STA</td>
<td>2503h</td>
<td>Store result at 2503h.</td>
</tr>
<tr>
<td>200F</td>
<td>79</td>
<td>MOV</td>
<td>A,C</td>
<td></td>
<td>Get carry in acc.</td>
</tr>
<tr>
<td>2010</td>
<td>32,04,25</td>
<td>STA</td>
<td>2504h</td>
<td></td>
<td>Store carry at 2504h</td>
</tr>
<tr>
<td>2013</td>
<td>76</td>
<td>HLT</td>
<td></td>
<td></td>
<td>Stop</td>
</tr>
</tbody>
</table>

Procedure:
1. Connect the kit to the power supply.
2. Connect the PS-2 keyboard to the PS-2 (Female) connector CN-7 of the kit.
3. Switch ON the kit.
4. LCD displays “STUDENT-85”, if not so check power connections.
5. If backlight is glowing and display shows something else press the RESET key provided on...
the kit. If still the persists contact Lab staff.

(6) To feed the program and data in the RAM, for machine language follow the steps 7 to 9 and for assembly language programming steps 10 to 14.


(8) The LCD display will show the address and its content. Modify the content according to the program using keyboard then press [ENTER] or [SPACEBAR]. The content will be saved and display will move the next address. Modify if required.

(9) Use UP/DOWN arrow keys move between the memory locations.

(10) Press [1][A] to select assembler mode.

(11) The display will show ‘RAM ADR: [BLANK]’, asking for the starting address which is 2000H in the above program.

(12) Type [2000] and press [ENTER].

(13) Display will show ‘2000: [BLANK]’. Type the mnemonics of the command operand etc. and press [ENTER].

(14) The display will show the next location. Type desired mnemonics and operand. Keep on doing this until the end of the program.

(15) Feed the data using the [M] command.

(16) To RUN/ EXECUTE the program Press [G]<Starting Address>[$].

(17) Go to memory locations used for result storage using [M] command and verify the results.

**Input Data:**

2501- 67h  
2502- 22h

**Output Data:**

2503- 89h

**Precautions:-**

(1) Make sure proper handling of equipments/ kits.

(2) Make sure that all the machine codes/ mnemonics are as per the program.
Flow-Chart of the process:

Result:

Addition of 67h and 22h is 89h. Hence experiment for the addition of two 8-bit numbers has been successfully performed.
EXPERIMENT NO. 6

Aim:-

Write a program using 8085 for addition of two BCD (8-bit) numbers.

Apparatus:-

8085 microprocessor kit (STUDENT-85AD), PS-2 keyboard.

Theory:

2501h is the address of memory location for the 1st number. 2nd number is stored at next memory location. By the virtue of the program given below we first move the 1st number into accumulator register then add the 2nd number to this and use the DAA (Decimal Adjust Accumulator) instruction to convert result to BCD. The result is then stored into the memory location 2503h. The comments in program give explanation of what happens after the execution of that particular instruction.

Procedure:-

(1) Connect the kit to the power supply.
(2) Connect the PS-2 keyboard to the PS-2 (Female) connector CN-7 of the kit.
(3) Switch ON the kit.
(4) LCD displays “STUDENT-85”, if not so check power connections.
(5) If backlight is glowing and display shows something else press the RESET key provided on the kit. If still the persists contact Lab staff.
(6) To feed the program and data in the RAM, for machine language follow the steps 7 to 9 and for assembly language programming steps 10 to 14.
(8) The LCD display will show the address and its content. Modify the content according to the program using keyboard then press [ENTER] or [SPACEBAR]. The content will be saved and display will move the next address. Modify if required.
(9) Use UP/DOWN arrow keys move between the memory locations.
(10) Press [1][A] to select assembler mode.
(11) The display will show ‘RAM ADR: [BLANK]’, asking for the starting address which is 2000H in the above program.
(12) Type [2000] and press [ENTER].
(13) Display will show ‘2000: [BLANK]’. Type the mnemonics of the command operand etc. and press [ENTER].
(14) The display will show the next location. Type desired mnemonics and operand. Keep on doing this until the end of the program.
(15) Feed the data using the [M] command.
(16) To RUN/ EXECUTE the program Press [G]<Starting Address>$.
(17) Go to memory locations used for result storage using [M] command and verify the results.
### Program:

<table>
<thead>
<tr>
<th>Memory address</th>
<th>Op-code</th>
<th>Labels</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>21,01,25</td>
<td></td>
<td>LXI</td>
<td>H, 2501</td>
<td>Get address of 1st no. in H-L Pair</td>
</tr>
<tr>
<td>2003</td>
<td>0E,00</td>
<td></td>
<td>MVI</td>
<td>C.00h</td>
<td>Initialize C reg. to store MSB of result.</td>
</tr>
<tr>
<td>2005</td>
<td>7E</td>
<td></td>
<td>MOV</td>
<td>A, M</td>
<td>Move 1st no. in Accumulator</td>
</tr>
<tr>
<td>2006</td>
<td>23</td>
<td></td>
<td>INX</td>
<td>H</td>
<td>H-L points to 2502h.</td>
</tr>
<tr>
<td>2007</td>
<td>86</td>
<td></td>
<td>ADD</td>
<td>M</td>
<td>Add 2(^{\text{nd}}) no. to 1st no.</td>
</tr>
<tr>
<td>2008</td>
<td>27</td>
<td></td>
<td>DAA</td>
<td></td>
<td>Decimal Adjust Accumulator</td>
</tr>
<tr>
<td>2009</td>
<td>D2,0D,20</td>
<td></td>
<td>JNC</td>
<td>AHEAD</td>
<td>If no carry go to AHEAD</td>
</tr>
<tr>
<td>200C</td>
<td>0C</td>
<td></td>
<td>INR</td>
<td>C</td>
<td>If carry increment C reg.</td>
</tr>
<tr>
<td>200D</td>
<td>32,03,25</td>
<td>AHEAD</td>
<td>STA</td>
<td>2503h</td>
<td>LSB of the sum in 2503h.</td>
</tr>
<tr>
<td>2010</td>
<td>79</td>
<td></td>
<td>MOV</td>
<td>A,C</td>
<td>MSB of sum in acc.</td>
</tr>
<tr>
<td>2011</td>
<td>32,04,25</td>
<td></td>
<td>STA</td>
<td>2504h</td>
<td>Store MSB at 2504h</td>
</tr>
<tr>
<td>2014</td>
<td>76</td>
<td></td>
<td>HLT</td>
<td></td>
<td>Stop</td>
</tr>
</tbody>
</table>
Flow-Chart of the process:

Start

Get 1\textsuperscript{st} number from memory

Initialize reg. C for Carry
Get 2\textsuperscript{nd} number from memory

Add 1\textsuperscript{st} and 2\textsuperscript{nd} number

Decimal Adjust Accumulator

Yes

Is there a carry?

Increment C register

Store result

Stop

Input Data: 2501-67h, 2502-22h

Output Data: 2503-89h

Precautions:

1. Make sure proper handling of equipments/kits.
2. Make sure that all the machine codes/mnemonics are as per the program.

Result:

Addition of 22h from 67h is 89h. Hence experiment for the addition of two 8-bit BCD numbers has been successfully performed.
EXPERIMENT NO. 7

**Aim:-**

Write a program using 8085 for subtraction of two BCD (8-bit) numbers.

**Apparatus:-**

8085 microprocessor kit (STUDENT-85AD), PS-2 keyboard.

**Theory:-**

2501h is the address of memory location for the 1st number. 2<sup>nd</sup> number is stored at next memory location. By the virtue of the program given below we first move the 1st number into accumulator register then subtract the 2<sup>nd</sup> number from this and use the DAA (Decimal Adjust Accumulator) instruction to convert result to BCD. The result is then stored into the memory location 2503h. The comments in program give explanation of what happens after the execution of that particular instruction.

**Program:-**

<table>
<thead>
<tr>
<th>Memory address</th>
<th>Op-code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>21,02,25</td>
<td>LXI</td>
<td>H, 2502</td>
<td>Get address of 2&lt;sup&gt;nd&lt;/sup&gt; no. in H-L pair</td>
</tr>
<tr>
<td>2005</td>
<td>96</td>
<td>SUB</td>
<td>M</td>
<td>9’s complement of 2&lt;sup&gt;nd&lt;/sup&gt; number</td>
</tr>
<tr>
<td>2006</td>
<td>3C</td>
<td>INR</td>
<td>A</td>
<td>10’s complement of 2&lt;sup&gt;nd&lt;/sup&gt; number</td>
</tr>
<tr>
<td>2007</td>
<td>2B</td>
<td>DCX</td>
<td>H</td>
<td>Decrement H-L pair</td>
</tr>
<tr>
<td>2008</td>
<td>86</td>
<td>ADD</td>
<td>M</td>
<td>Add 1&lt;sup&gt;st&lt;/sup&gt; number to 10’s complement of 2&lt;sup&gt;nd&lt;/sup&gt; number</td>
</tr>
<tr>
<td>2009</td>
<td>27</td>
<td>DAA</td>
<td></td>
<td>Decimal Adjust Accumulator</td>
</tr>
<tr>
<td>200A</td>
<td>32,03,25</td>
<td>STA</td>
<td>2503h</td>
<td>Store result at 2503h</td>
</tr>
<tr>
<td>200D</td>
<td>CF</td>
<td>RST 1</td>
<td></td>
<td>Stop</td>
</tr>
</tbody>
</table>

**Procedure:-**

1. Connect the kit to the power supply.
2. Connect the PS-2 keyboard to the PS-2 (Female) connector CN-7 of the kit.
3. Switch ON the kit.
4. LCD displays “STUDENT-85”, if not so check power connections.
5. If backlight is glowing and display shows something else press the RESET key provided on the kit. If still the persists contact Lab staff.
6. To Feed the program and data in the RAM, for machine language follow the steps 7 to 9 and for assembly language programming steps 10 to 14.
7. Press [M]<Starting Address >[ENTER]. Starting address here is 2000H, so syntax will be
[M]<2000>[ENTER].

(8) The LCD display will show the address and its content. Modify the content according to the program using keyboard then press [ENTER] or [SPACEBAR]. The content will be saved and display will move the next address. Modify if required.

(9) Use UP/DOWN arrow keys move between the memory locations.

(10) Press [1][A] to select assembler mode.

(11) The display will show ‘RAM ADR: [BLANK]’, asking for the starting address which is 2000H in the above program.

(12) Type [2000] and press [ENTER].

(13) Display will show ‘2000: [BLANK]’. Type the mnemonics of the command operand etc. and press [ENTER].

(14) The display will show the next location. Type desired mnemonics and operand. Keep on doing this until the end of the program.

(15) Feed the data using the [M] command.

(16) To RUN/EXECUTE the program Press [G]<Starting Address>[$].

(17) Go to memory locations used for result storage using [M] command and verify the results.

Flow-Chart of the process :-

![Flow-Chart](image-url)
**Input Data:**
2501- 67h (0110 0111), 2502- 22h (0010 0010)

**Output Data:**
2503- 45h (0100 0101)

**Precautions:-**

1. Make sure proper handling of equipments/ kits.
2. Make sure that all the machine codes/ mnemonics are as per the program.

**Result:-**
Subtraction of 22h from 67h is 45h. Hence experiment for the subtraction of two 8-bit BCD numbers has been successfully performed.
EXPERIMENT NO. 8

Aim:-
Write a program using 8085 for multiplication of two 8-bit numbers by repeated addition method.

Apparatus:-
8085-micro-processor kit (STUDENT-85AD), ASCII (PS-2) keyboard.

Theory:
Repeated Addition method of multiplication is simplest method of multiplying two numbers. For use in microprocessors, we first store multiplicand and multiplier in any of its general purpose registers. We use a register pair for the storage of result. Then we keep on adding the multiplicand to result and decrement the multiplier every time we add. This cycle continues until multiplier becomes zero. Then the result is moved to any desired memory location. A well documented program for this purpose is given below.

Program:-

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Machine Code</th>
<th>Labels</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>0E,10</td>
<td>MVI</td>
<td>C,08h</td>
<td>Move the multiplicand in reg. C</td>
<td></td>
</tr>
<tr>
<td>2002</td>
<td>1E,20</td>
<td>MVI</td>
<td>E,07h</td>
<td>Move the multiplier in reg. E</td>
<td></td>
</tr>
<tr>
<td>2004</td>
<td>06,00</td>
<td>MVI</td>
<td>B,00</td>
<td>Load 00h in B reg.</td>
<td></td>
</tr>
<tr>
<td>2006</td>
<td>21,00,00</td>
<td>LXI</td>
<td>H,0000</td>
<td>Initial Product=0000</td>
<td></td>
</tr>
<tr>
<td>2009</td>
<td>09</td>
<td>UP1:</td>
<td>DAD</td>
<td>B,HL+BC=&gt;HL</td>
<td></td>
</tr>
<tr>
<td>200A</td>
<td>1D</td>
<td>DCR</td>
<td>E</td>
<td>Decrement reg. E</td>
<td></td>
</tr>
<tr>
<td>200B</td>
<td>C2,09,20</td>
<td>JNZ</td>
<td>UP1</td>
<td>Jump if not zero to location up1</td>
<td></td>
</tr>
<tr>
<td>200E</td>
<td>22,01,25</td>
<td>SHLD</td>
<td>2501h</td>
<td>Store result at 2501h</td>
<td></td>
</tr>
<tr>
<td>2012</td>
<td>76</td>
<td>HLT</td>
<td></td>
<td>Stop</td>
<td></td>
</tr>
</tbody>
</table>

Procedure:-

(1) Connect the kit to the power supply.
(2) Connect the PS-2 keyboard to the PS-2 (Female) connector CN-7 of the kit.
(3) Switch ON the kit.
(4) LCD displays “STUDENT-85”, if not so check power connections.
(5) If backlight is glowing and display shows something else press the RESET key provided on the kit. If still the persists contact Lab staff.
(6) To Feed the program and data in the RAM, for machine language follow the steps 7 to 9 and for assembly language programming steps 10 to 14.
(8) The LCD display will show the address and its content. Modify the content according to the program using keyboard then press [ENTER] or [SPACEBAR]. The content will be saved and display will move the next address. Modify if required.

(9) Use UP/DOWN arrow keys move between the memory locations.

(10) Press [1][A] to select assembler mode.

(11) The display will show ‘RAM ADR: [BLANK]’, asking for the starting address which is 2000H in the above program.

(12) Type [2000] and press [ENTER].

(13) Display will show ‘2000: [BLANK]’. Type the mnemonics of the command operand etc. and press [ENTER].

(14) The display will show the next location. Type desired mnemonics and operand. Keep on doing this until the end of the program.

(15) Feed the data using the [M] command.

(16) To RUN/EXECUTE the program Press [G]<Starting Address>$.

(17) Go to memory locations used for result storage using [M] command and verify the results.

**Flow-Chart of the process:-**
Input Data: C reg.- 08h (0000 1000), E reg.- 07h (0000 0111)
Output Data: 2501- 38h (0011 1000)

Precautions:-
(1) Make sure proper handling of equipments/ kits.
(2) Make sure that all the machine codes/ mnemonics are as per the program.

Result:-
Multiplication of 08h and 07h is 38h. Hence experiment for the multiplication of two 8-bit numbers by repeated addition method has been successfully performed.
EXPERIMENT NO.9

Aim:-

Write a program using 8085 for multiplication of two 8-bit numbers by bit-rotation method.

Apparatus:-

8085- micro-processor kit (STUDENT-85AD), ASCII (PS-2) keyboard.

Theory:-

In binary multiplication we see that when a multiplicand is multiplied by 1 the product is same as the multiplicand. When a multiplicand is multiplied by zero, the product is zero. The procedure for multiplication is that multiplicand is rotated to left by one bit if the MSB of multiplier is 1. Partial product is stored. We keep on rotating the multiplier, decrementing cycle-counter and add the multiplicand to the partial product until the cycle-counter which is set to 8 becomes zero. When cycle-counter is 0 that means we have completed the multiplication and we store the result as our final result. The method of multiplication applied here is known as BIT ROTATION METHOD.

Program:-

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Machine Code</th>
<th>Labels</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2003</td>
<td>EB</td>
<td>XCHG</td>
<td></td>
<td>Exchange HL pair with DE pair</td>
<td></td>
</tr>
<tr>
<td>2004</td>
<td>3A,03,25</td>
<td>LDA</td>
<td>7503 H</td>
<td>Get 2nd no. in acc.</td>
<td></td>
</tr>
<tr>
<td>2007</td>
<td>21,00,00</td>
<td>LXI</td>
<td>H,0000</td>
<td>Initial product in HL=00</td>
<td></td>
</tr>
<tr>
<td>200A</td>
<td>0E,08</td>
<td>MVI</td>
<td>C,08H</td>
<td>Count=08 in reg. C</td>
<td></td>
</tr>
<tr>
<td>200C</td>
<td>29</td>
<td>LOOP</td>
<td>DAD</td>
<td>H</td>
<td>Shift partial product left by 1 bit</td>
</tr>
<tr>
<td>200D</td>
<td>17</td>
<td>RAL</td>
<td></td>
<td></td>
<td>Rotate multiplier by 1bit.</td>
</tr>
<tr>
<td>200E</td>
<td>D2,12,20</td>
<td>JNC</td>
<td>AHEAD</td>
<td>If no carry, go to AHEAD</td>
<td></td>
</tr>
<tr>
<td>2011</td>
<td>19</td>
<td>DAD</td>
<td>D</td>
<td>Product=Product +Multiplicand</td>
<td></td>
</tr>
<tr>
<td>2012</td>
<td>0D</td>
<td>AHEAD</td>
<td>DCR</td>
<td>C</td>
<td>Decrement Count</td>
</tr>
<tr>
<td>2013</td>
<td>C2,0C,20</td>
<td>JNZ</td>
<td>LOOP</td>
<td>If C≠0, go to LOOP</td>
<td></td>
</tr>
<tr>
<td>2016</td>
<td>22,04,25</td>
<td>SHLD</td>
<td>7504</td>
<td>Store result</td>
<td></td>
</tr>
<tr>
<td>2019</td>
<td>76</td>
<td>HLT</td>
<td></td>
<td>Stop</td>
<td></td>
</tr>
</tbody>
</table>

Procedure:-

(1) Connect the kit to the power supply.
(2) Connect the PS-2 keyboard to the PS-2 (Female) connector CN-7 of the kit.
(3) Switch ON the kit.
(4) LCD displays “STUDENT-85”, if not so check power connections.

(5) If backlight is glowing and display shows something else press the RESET key provided on the kit. If still the persists contact Lab staff.

(6) To feed the program and data in the RAM, for machine language follow the steps 7 to 9 and for assembly language programming steps 10 to 14.


(8) The LCD display will show the address and its content. Modify the content according to the program using keyboard then press [ENTER] or [SPACEBAR]. The content will be saved and display will move the next address. Modify if required.

(9) Use UP/DOWN arrow keys move between the memory locations.

(10) Press [1][A] to select assembler mode.

(11) The display will show ‘RAM ADR: [BLANK]’, asking for the starting address which is 2000H in the above program.

(12) Type [2000] and press [ENTER].

(13) Display will show ‘2000: [BLANK]’. Type the mnemonics of the command operand etc. and press [ENTER].

(14) The display will show the next location. Type desired mnemonics and operand. Keep on doing this until the end of the program.

(15) Feed the data using the [M] command.

(16) To RUN/EXECUTE the program Press [G]<Starting Address>[$].

(17) Go to memory locations used for result storage using [M] command and verify the results.

**Input Data:**
C reg.- 08h (0000 1000), E reg.- 07h (0000 0111)

**Output Data:**
2501- 38h (0011 1000)

**Precautions:**

(1) Make sure proper handling of equipments/ kits.

(2) Make sure that all the machine codes/ mnemonics are as per the program.
Flow-Chart of Process:-

Result:-

Multiplication of 08h and 07h is 38h. Hence experiment for the multiplication of two 8-bit numbers by repeated addition method has been successfully performed.
EXPERIMENT NO. 10

Aim:

Write a program using 8085 for division of a number by another number (8-bit) using repeated subtraction method.

Apparatus:

8085-micro-processor kit (STUDENT-85AD), ASCII (PS-2) keyboard.

Theory:

Repeated subtraction method of division is the simplest method of dividing two numbers. For use in microprocessors, we can store dividend and divisor in any of its general purpose registers. If the dividend is bigger than 8-bit we can use a register pair for its storage. Then we keep on subtracting the divisor form dividend or remainder of subtraction and increment the quotient every time we subtract. This cycle continues until remainder becomes zero or less than divisor. Then the result is moved to any desired memory location. A well documented program for this purpose is given below.

Program:

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Machine Code</th>
<th>Labels</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>21,00,21</td>
<td></td>
<td>LXI</td>
<td>H,2100h</td>
<td>Address of divisor in H-L pair</td>
</tr>
<tr>
<td>2003</td>
<td>46</td>
<td></td>
<td>MOV</td>
<td>B,M</td>
<td>Divisor in reg. B</td>
</tr>
<tr>
<td>2004</td>
<td>23</td>
<td></td>
<td>INX</td>
<td>H</td>
<td>Increment H-L pair</td>
</tr>
<tr>
<td>2005</td>
<td>7E</td>
<td></td>
<td>MOV</td>
<td>A,M</td>
<td>Dividend in Acc.</td>
</tr>
<tr>
<td>2006</td>
<td>23</td>
<td></td>
<td>INX</td>
<td>H</td>
<td>Increment H-L pair</td>
</tr>
<tr>
<td>2007</td>
<td>0E,00</td>
<td></td>
<td>MVI</td>
<td>C,00h</td>
<td>Initialize C reg. for quotient storage</td>
</tr>
<tr>
<td>2009</td>
<td>B8</td>
<td>LP1</td>
<td>CMP</td>
<td>B</td>
<td>Compare acc. With B reg.</td>
</tr>
<tr>
<td>200A</td>
<td>DA,13,20</td>
<td></td>
<td>JMP</td>
<td>LP1</td>
<td>Jump back and repeat from to LP1</td>
</tr>
<tr>
<td>200D</td>
<td>90</td>
<td>SUB</td>
<td>B</td>
<td></td>
<td>Subtract divisor from dividend</td>
</tr>
<tr>
<td>200E</td>
<td>0C</td>
<td>INR</td>
<td>C</td>
<td></td>
<td>Increment C reg.</td>
</tr>
<tr>
<td>200F</td>
<td>C3,09,20</td>
<td></td>
<td>JMP</td>
<td>LP1</td>
<td>Jump back and repeat from to LP1</td>
</tr>
<tr>
<td>2012</td>
<td>77</td>
<td>LOOP</td>
<td>MOV</td>
<td>M,A</td>
<td>Store remainder at 2102h</td>
</tr>
<tr>
<td>2013</td>
<td>23</td>
<td>INX</td>
<td>H</td>
<td></td>
<td>Increment H-L pair</td>
</tr>
<tr>
<td>2014</td>
<td>71</td>
<td>MOV</td>
<td>M,C</td>
<td></td>
<td>Store result at 2103h</td>
</tr>
<tr>
<td>2015</td>
<td>76</td>
<td>HLT</td>
<td></td>
<td></td>
<td>Stop</td>
</tr>
</tbody>
</table>
Procedure:

1. Connect the kit to the power supply.
2. Connect the PS-2 keyboard to the PS-2 (Female) connector CN-7 of the kit.
3. Switch ON the kit.
4. LCD displays “STUDENT-85”, if not so check power connections.
5. If backlight is glowing and display shows something else press the RESET key provided on the kit. If still the persists contact Lab staff.
6. To Feed the program and data in the RAM, for machine language follow the steps 7 to 9 and for assembly language programming steps 10 to 14.
8. The LCD display will show the address and its content. Modify the content according to the program using keyboard then press [ENTER] or [SPACEBAR]. The content will be saved and display will move the next address. Modify if required.
9. Use UP/DOWN arrow keys move between the memory locations.
11. The display will show ‘RAM ADR: [BLANK]’, asking for the starting address which is 2000H in the above program.
12. Type [2000] and press [ENTER].
13. Display will show ‘2000: [BLANK]’. Type the mnemonics of the command operand etc. and press [ENTER].
14. The display will show the next location. Type desired mnemonics and operand. Keep on doing this until the end of the program.
15. Feed the data using the [M] command.
16. To RUN/EXECUTE the program Press [G]<Starting Address>[$].
17. Go to memory locations used for result storage using [M] command and verify the results.

Input Data: 2100h - 38h (0011 1000), 2101h- 07h (0000 0111)
Output Data: 2102h - 00h (0000 0000), 2103h- 08h (0000 1000)

Precautions:-

(1) Make sure proper handling of equipments/ kits.
(2) Make sure that all the machine codes/ mnemonics are as per the program.
Flow-Chart of the process:

Result:--

Multiplication of 08h and 07h is 38h. Hence experiment for the multiplication of two 8-bit numbers by repeated addition method has been successfully performed.
**EXPERIMENT NO.11**

**Aim:**

Write a well document program using 8085 for division of two 8-bit numbers by bit-rotation method.

**Apparatus:**

8085-micro-processor kit (STUDENT-85AD), ASCII (PS-2) Keyboard.

**Theory:**

The computer performs division by trail subtraction. The divisor is subtracted from 8 MSBs of dividend. If there is borrow, the bit of quotient is set to 1; otherwise 0. To line up the dividend and quotient properly the dividend is shifted left by one bit before each trail of subtraction. The dividend and quotient share a 16-bit register. Due to shifting of dividend one bit register falls vacant in each step. The quotient is stored in vacant bit positions. The program and flow-chart are given below.

**Program:**

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Machine Code</th>
<th>Labels</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>2A, 01,25</td>
<td></td>
<td>LHLD</td>
<td></td>
<td>Enter the 16 bit address in HL pair</td>
</tr>
<tr>
<td>2003</td>
<td>3A, 03,25</td>
<td></td>
<td>LDA</td>
<td>2503 H</td>
<td>Get divisor from 2503</td>
</tr>
<tr>
<td>2006</td>
<td>47</td>
<td></td>
<td>MOV</td>
<td>B, A</td>
<td>Divisor in register B</td>
</tr>
<tr>
<td>2007</td>
<td>0E, 08</td>
<td></td>
<td>MVI</td>
<td>C, 08</td>
<td>Count = 08 in register C.</td>
</tr>
<tr>
<td>2009</td>
<td>29</td>
<td>LOOP</td>
<td>DAD</td>
<td>H</td>
<td>Shift dividend and quotient left by one bit.</td>
</tr>
<tr>
<td>200A</td>
<td>7C</td>
<td></td>
<td>MOV</td>
<td>A, H</td>
<td>Most significant bits of dividend in acc.</td>
</tr>
<tr>
<td>200B</td>
<td>90</td>
<td></td>
<td>SUB</td>
<td>B</td>
<td>Subtract divisor from MSB of dividend.</td>
</tr>
<tr>
<td>200C</td>
<td>DA,11,24</td>
<td></td>
<td>JC</td>
<td>AHEAD</td>
<td>Is MSB of dividend&gt;divisor? No, go to AHEAD.</td>
</tr>
<tr>
<td>200F</td>
<td>67</td>
<td></td>
<td>MOV</td>
<td>H, A</td>
<td>MSB of dividend in reg. H</td>
</tr>
<tr>
<td>2010</td>
<td>2C</td>
<td></td>
<td>INR</td>
<td>L</td>
<td>Yes, add 1 to quotient.</td>
</tr>
<tr>
<td>2011</td>
<td>0D</td>
<td>AHEAD</td>
<td>DCR</td>
<td>C</td>
<td>Decrement count.</td>
</tr>
<tr>
<td>2012</td>
<td>C2,09,24</td>
<td></td>
<td>JNZ</td>
<td>LOOP</td>
<td>Is count=0? No, jump to loop.</td>
</tr>
<tr>
<td>2015</td>
<td>22,04,25</td>
<td></td>
<td>SHLD</td>
<td>2504 H</td>
<td>Store quotient in 2504 and remainder in 2505 H.</td>
</tr>
<tr>
<td>2018</td>
<td>76</td>
<td></td>
<td>HLT</td>
<td></td>
<td>Stop.</td>
</tr>
</tbody>
</table>
Procedure:-

(1) Connect the kit to the power supply.
(2) Connect the PS-2 keyboard to the PS-2 (Female) connector CN-7 of the kit.
(3) Switch ON the kit.
(4) LCD displays “STUDENT-85”, if not so check power connections.
(5) If backlight is glowing and display shows something else press the RESET key provided on the kit. If still the persists contact Lab staff.
(6) To Feed the program and data in the RAM, for machine language follow the steps 7 to 9 and for assembly language programming steps 10 to 14.
(8) The LCD display will show the address and its content. Modify the content according to the program using keyboard then press [ENTER] or [SPACEBAR]. The content will be saved and display will move the next address. Modify if required.
(9) Use UP/DOWN arrow keys move between the memory locations.
(10) Press [1][A] to select assembler mode.
(11) The display will show ‘RAM ADR: [BLANK]’, asking for the starting address which is 2000H in the above program.
(12) Type [2000] and press [ENTER].
(13) Display will show ‘2000: [BLANK]’. Type the mnemonics of the command operand etc. and press [ENTER].
(14) The display will show the next location. Type desired mnemonics and operand. Keep on doing this until the end of the program.
(15) Feed the data using the [M] command.
(16) To RUN/EXECUTE the program Press [G]<Starting Address>[$].
(17) Go to memory locations used for result storage using [M] command and verify the results.

Input Data: 2100h - 38h (0011 1000), 2101h- 07h (0000 0111)
Output Data: 2102h - 00h (0000 0000), 2103h- 08h (0000 1000)

Precautions:-

(1) Make sure proper handling of equipments/ kits.
(2) Make sure that all the machine codes/ mnemonics are as per the program.
Flow-Charts of the process:-

Start

Get dividend, get divisor
Count =08, Quotient =00

Shift dividend left one bit
Shift divisor left by one bit

No

Is MSBs of Dividend > divisor?

Quotient = Quotient+1

8MSBs of dividend =8MSBs of dividend -divisor

Count = count -1

No

Is count =0?

Yes

Store remainder and quotient

Stop

Result:-

Division of 38h by 07h is 08h. Hence experiment for the division of an 8-bit numbers by repeated addition method has been successfully performed.
EXPERIMENT NO.12

Aim:
To write an ALP for finding the square of a number from look up table method using 8085.

Apparatus:
8085-micro-processor kit (STUDENT-85AD), ASCII (PS-2) keyboard.

Theory:
The squares of number are stored in certain memory locations in tabular form. This table is called look-up table. Here squares of numbers from 00 to 09 are stored at locations 2600 to 2609h. All these values are in decimal numbers. The program given below is self explanatory.

Program:

<table>
<thead>
<tr>
<th>Memory</th>
<th>Machine</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>3A,00,25</td>
<td>LDA</td>
<td>2500 H</td>
<td>Get number in accumulator</td>
</tr>
<tr>
<td>2003</td>
<td>6F</td>
<td>MOV</td>
<td>L,A</td>
<td>Move From A into reg. L</td>
</tr>
<tr>
<td>2004</td>
<td>26,26</td>
<td>MVI</td>
<td>H,26H</td>
<td>Get 26 in reg. H</td>
</tr>
<tr>
<td>2006</td>
<td>7E</td>
<td>MOV</td>
<td>A,M</td>
<td>Square of data in accumulator</td>
</tr>
<tr>
<td>2007</td>
<td>32,01,25</td>
<td>STA</td>
<td>2501 H</td>
<td>Store square in 2501 H.</td>
</tr>
<tr>
<td>200A</td>
<td>76</td>
<td>HLT</td>
<td></td>
<td>Stop</td>
</tr>
</tbody>
</table>

Look-Up table

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>2600h</td>
<td>00</td>
</tr>
<tr>
<td>2601h</td>
<td>01</td>
</tr>
<tr>
<td>2602h</td>
<td>04</td>
</tr>
<tr>
<td>2603h</td>
<td>09</td>
</tr>
<tr>
<td>2604h</td>
<td>16</td>
</tr>
<tr>
<td>2605h</td>
<td>25</td>
</tr>
<tr>
<td>2606h</td>
<td>36</td>
</tr>
<tr>
<td>2607h</td>
<td>49</td>
</tr>
<tr>
<td>2608h</td>
<td>64</td>
</tr>
<tr>
<td>2609h</td>
<td>81</td>
</tr>
</tbody>
</table>
Procedure:-

(1) Connect the kit to the power supply.
(2) Connect the PS-2 keyboard to the PS-2 (Female) connector CN-7 of the kit.
(3) Switch ON the kit. LCD displays “STUDENT-85”, if not so check power connections. If backlight is glowing and display shows something else press the RESET key provided on the kit. If still it persists contact Lab staff.
(4) To Feed the program and data in the RAM, for machine language follow the steps 7 to 9 and for assembly language programming steps 10 to 14.
(6) The LCD display will show the address and its content. Modify the content according to the program using keyboard then press [ENTER] or [SPACEBAR]. The content will be saved and display will move the next address. Modify if required.
(7) Use UP/DOWN arrow keys move between the memory locations.
(8) Press [1][A] to select assembler mode.
(9) The display will show ‘RAM ADR: [BLANK]’, asking for the starting address which is 2000H in the above program.
(10) Type [2000] and press [ENTER].
(11) Display will show ‘2000: [BLANK]’. Type the mnemonics of the command operand etc. and press [ENTER].
(12) The display will show the next location. Type desired mnemonics and operand. Keep on doing this until the end of the program.
(13) Feed the data using the [M] command.
(14) To RUN/EXECUTE the program Press [G]<Starting Address>[\$].
(15) Go to memory locations used for result storage using [M] command and verify the results.

Flow-Chart of the process:-

```
Start
  ↓
Get number from memory
  ↓
Form the address of Square from the Look-up table
  ↓
Find the square of the number
  ↓
Store result
  ↓
Stop
```
**Input Data:** 2500h- 07d  
**Output Data:** 2503h- 49d

**Precautions:**  
(1) Make sure proper handling of equipments/ kits.  
(2) Make sure that all the machine codes/ mnemonics are as per the program.

**Result:**  
Square of the number 07d is 49d. Hence experiment for the calculation of square of given number using look-up table has been successfully performed.
EXPERIMENT NO. 13

Aim:
Write a program using 8085 for finding largest number in a data array.

Apparatus:
8085 microprocessor kit (STUDENT-85AD), ASCII (PS-2) Keyboard.

Program:

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>7000</td>
<td>21,00,75</td>
<td>LXI H, 7500H</td>
<td>Address for count in H-L pair</td>
</tr>
<tr>
<td>7003</td>
<td>4E</td>
<td>MOV C, M</td>
<td>Count in register C</td>
</tr>
<tr>
<td>7004</td>
<td>23</td>
<td>INX H</td>
<td>Address of 1st number in HL Pair</td>
</tr>
<tr>
<td>7005</td>
<td>7E</td>
<td>MOV A,M</td>
<td>1st no. in accumulator</td>
</tr>
<tr>
<td>7006</td>
<td>0D</td>
<td>DCR C</td>
<td>Decrement count</td>
</tr>
<tr>
<td>7007</td>
<td>23 Loop</td>
<td>INX H</td>
<td>Address of next number</td>
</tr>
<tr>
<td>7008</td>
<td>BE</td>
<td>CMP M</td>
<td>Is next number  &gt; previous no.</td>
</tr>
<tr>
<td>7009</td>
<td>D2,0D,70</td>
<td>JNC Ahead</td>
<td>If not carry, jump to ahead</td>
</tr>
<tr>
<td>700C</td>
<td>7E</td>
<td>MOV A,M</td>
<td>Get larger no. into acc.</td>
</tr>
<tr>
<td>700D</td>
<td>0D Ahead</td>
<td>DCR C</td>
<td>Decrement count</td>
</tr>
<tr>
<td>700E</td>
<td>C2,07,70</td>
<td>JNZ Loop</td>
<td>Jump if no zero to loop</td>
</tr>
<tr>
<td>7011</td>
<td>32,50,74</td>
<td>STA 7450 H</td>
<td>Store result at 7450.</td>
</tr>
<tr>
<td>7014</td>
<td>CF</td>
<td>RST 1</td>
<td>Terminate.</td>
</tr>
</tbody>
</table>

Procedure:-
1. Connect the kit to the power supply.
2. Connect the PS-2 keyboard to the PS-2 (Female) connector CN-7 of the kit.
3. Switch ON the kit.
4. LCD displays “STUDENT-85”, if not so check power connections.
5. If backlight is glowing and display shows something else press the RESET key provided on the kit. If still the persists contact Lab staff.
6. To Feed the program and data in the RAM, for machine language follow the steps 7 to 9 and for assembly language programming steps 10 to 14.
8. The LCD display will show the address and its content. Modify the content according to the program using keyboard then press [ENTER] or [SPACEBAR]. The content will be saved and
display will move the next address. Modify if required.

(9) Use UP/DOWN arrow keys move between the memory locations.

(10) Press [1][A] to select assembler mode.

(11) The display will show ‘RAM ADR: [BLANK]’, asking for the starting address which is 2000H in the above program.

(12) Type [2000] and press [ENTER].

(13) Display will show ‘2000: [BLANK]’. Type the mnemonics of the command operand etc. and press [ENTER].

**Flow Chart:**

```
START

Take the count in reg. C

Enter first No.

Enter next No.

Is first No. > next No?

YES

NO

Replace 2nd No. in acc.

Decrement Count

No

IS Count = 0?

YES

Store largest No. in acc.

END
```
<table>
<thead>
<tr>
<th>Input Data</th>
<th>7500-03 (Counter)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>7501-</td>
</tr>
<tr>
<td></td>
<td>7502-</td>
</tr>
<tr>
<td></td>
<td>7503-</td>
</tr>
<tr>
<td>Output Data:</td>
<td>7450-</td>
</tr>
</tbody>
</table>

**Precautions:**

1. Make sure proper handling of equipments/ kits.
2. Make sure that all the machine codes/ mnemonics are as per the program.

**Result:**

Thus the largest number in array stored in respective memory location and verified the data.
EXPERIMENT NO.14

AIM:
To study of 8086 Microprocessor Kit.

APPARATUS:
VMC-8609 8086 microprocessor trainer kit.

THEORY:
The 8086 is a 16-bit, N-channel, HMOS microprocessor. The term HMOS is used for "high-speed MOS". The 8086 uses 20 address lines and 16 data lines. It can directly address up to $2^{20} = 1$ Mbytes of memory. The 16-bit data word is divided into a low-order byte and a high-order byte. The 20 address lines are time multiplexed lines. The 16 low-order address lines are time multiplexed with data, and the 4 high-order address lines are time multiplexed with status signals.

OPERATING MODES OF 8086
There are two modes of operation for Intel 8086, namely the minimum mode and the maximum mode. When only one 8086 CPU is to be used in a microcomputer system the 8086 is used in the minimum mode of operation. In this mode the CPU issues the control signals required by memory and I/O devices. In case of maximum mode of operation control signals are issued by Intel 8288 bus controller which is used with 8086 for this very purpose. When MN/MX is high the CPU operates in the minimum mode. When it is low the CPU operates in the maximum mode.

Pin Description for Minimum Mode
For the minimum mode of operation the pin MN/MX is connected to 5V d.c.supply. The description of the pins from 24 to 31 for the minimum mode is as follows:

- **INTA(Output)**: Pin no. 24 Interrupt acknowledge. On receiving interrupt signal the processor issues an interrupt acknowledge signal. It is active LOW.

- **ALE(Output)**: Pin no. 25 Address latch enable. It goes HIGH during T1. The microprocessor sends this signal to latch the address into the Intel 8282/8283 latch.

- **DEN(Output)**: Pin no. 26 Data enable. When Intel 8286/8287 octal bus transceiver is used this signal acts as an output enable signal. It is active LOW.

- **DT/R(Output)**: Pin no. 27 Data Transmit/Receive. When Intel 8286/8287 octal bus transceiver is used this signal controls the direction of data flow through the transceiver. When it is High data are sent out. When it is LOW data are received.

- **M/IO(Output)**: Pin no. 28. Memory or I/O access. When it is HIGH the CPU wants to access memory. When it is LOW the CPU wants to access I/O device.

- **WR (Output)**: Pin no. 29. Write. When it is LOW the CPU performs memory or I/O write Operation.

- **HLDA (Output)**: Pin no. 30. HOLD acknowledge. It is issued by the processor when it receives HOLD signal. It is active HIGH signal. When HOLD request is removed HLDA goes LOW.
HOLD (Output) : Pin no. 31. Hold. When another device in microcomputer system wants to use the address and data bus, it sends a HOLD request to CPU through this pin. It is an active HIGH signal.

Pin Description for Maximum Mode

For the maximum mode of operation the pin MN/MX is made LOW. It is grounded. The description of the pins from 24 to 31 is as follows:

QS1,QS0(Output): Pin no. 24,25 Instruction Queue status. Logic are given below:

\[
\begin{array}{cc}
0 & 0 \quad \text{No operation}
\\
0 & 1 \quad \text{St} \\
1 & 0 \quad \text{Empty the queue}
\\
1 & 1 \quad \text{Subsequent byte from queue}
\end{array}
\]

S0,S1,S2(Output) : Pin nos. 26,27,28.status signals. These signals are connected to the bus controller Intel 8288. The bus controller generates memory and I/O access control signals. Table for status signals is:

\[
\begin{array}{ccc}
S2 & S1 & S0 \\
0 & 0 & 0 \quad \text{Interrupt acknowledge}
\\
0 & 0 & 1 \quad \text{Read data from I/O port}
\\
0 & 1 & 0 \quad \text{Write data into I/O port}
\\
0 & 1 & 1 \quad \text{Halt}
\\
1 & 0 & 0 \quad \text{Opcode fetch}
\\
1 & 0 & 1 \quad \text{Memory read}
\\
1 & 1 & 0 \quad \text{Memory write}
\\
1 & 1 & 1 \quad \text{Passive state.}
\end{array}
\]

LOCK(Output) : Pin no. 29. It is an active LOW signal. When it is LOW all interrupts are masked and no HOLD request is granted. In a multiprocessor system all other processors are informed by this signal that they should not ask the CPU for relinquishing the bus control.

RQ / GT1, RQ / GT0 (Bidirectional) : Pin no. 30,31. Local bus Priority control. Other processors ask the CPU through these lines to release the local bus. RQ / GT1 has higher priority than RQ / GT0

FUNCTIONAL UNITS OF 8086 :

The 8086 contains two functional units: a bus interface unit (BIU) and an execution unit (EU). The general purpose registers, stack pointer, base pointer and index registers, ALU, flag register (FLAGS), instruction decoder and timing and control unit constitute execution unit (EU). The segment registers, instruction pointer and 6-byte instruction queue are associated with the bus interface unit (BIU).
**BLOCK DIAGRAM OF 8086:**

![Block Diagram of Intel 8086 Microprocessor](image)

**REGISTERS OF 8086**: The Intel 8086 contains the following registers:

a) General Purpose Register
b) Pointer and Index Registers
c) Segment Registers
d) Instruction Registers
e) Status Flags

**Introduction to VMC-8609 (8086 Microprocessor Trainer Kit)**

**General Description:**

VMC-8609 is a single board Microprocessor Training/Development kit. Configured around the INTEL-8086 microprocessor. This kit can be used to train engineers, to control any industrial process and to develop software for 8086 system.

The kit has been designed to operate in the Maximum or Minimum mode. Co-processors 8087 or 8089 can be added if required. The 8086 can also be replaced by 8088 microprocessor.

The kit communicates with the outside world through an IBM PC compatible keyboard and LCD display.
VMC-8609 is packed up with powerful monitor in 32KB of factory programmed EPROMs and 32KB of RAM for user. This memory can be expanded up to 256KB each. The system has 72 programmable I/O lines. The serial I/O communication is made possible through 8251.

For control applications three 16-bit timer/counters are available through 8253. For real time applications, 8 level of interrupt are provided through 8259. VMC-8609 provides onboard battery backup for RAM. This saves the user program in case of power failure.

The onboard resident SYSTEM MONITOR SOFTWARE is very powerful. It provides various software commands like BLOCK MOVE, INSERT, DELETE, FILL etc. which are helpful in debugging/developing software. An onboard Assembler/Disassembler is also provided on VMC-
8609. The kit also supports MASM. VMC-8609 also has onboard buzzer for self testing of hardware and software. This kit is also provided with a Centronix Printer port to take out the prints of the program written in the RAM of kit. A Real-Time Clock is provided onboard for real time applications.

**OPERATION:**

The operation of the kit is very simple. The operation instructions will be displayed when the device is being switched ON or RESET.

After power on the system, it will display as follows:

```
VMC-8609
ENTER RETURN KEY . . .
```

After pressing [ENTER], the operating commands will be displayed:

```
VMC-8609 . . 7F F F
>A. D. F. G. I. M. P. T. U.
```

**COMMAND DESCRIPTION:**

1) [A] **ASSEMBLE** : Press the key [A] and the LCD is shown as:

```
VMC-8609 . . 7F F F
A
```

Now the user enters the segment address and effective address simultaneously as follows:

```
VMC-8609 . . 7F F F
A 0000:0400
```

Now press [ENTER] key, the effective address will appear

```
0400
```

From now onwards user can type the program in assembly language as shown below:
Pressing [ENTER] from here will store the instruction at the effective address specified in the instruction and display will show next address. Then next instruction can be typed and stored in same manner. The sequence will continue till the end of the particular program.

2) [D] DISPLAY OR MODIFY THE RAM'S CONTENT:

Press [RESET] and [ENTER] to return to the command menu. Select [D] from this menu the display will show as follows:

```
VMC-8609..7FFF
D
```

Type the segment address and offset address of the desired location you wish to modify

```
VMC-8609..7FFF
D1234:0300
```

(Note: Segment address will be taken as 0000H until and unless specified.)

Press [ENTER], the display will show the content of RAM as shown below:

<table>
<thead>
<tr>
<th>Segment</th>
<th>Offset</th>
<th>Content of total Base Address 8 bytes of RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1234:0300B3C09A78</td>
<td>F000F0BB</td>
<td></td>
</tr>
</tbody>
</table>

From here on content of any RAM location can be modified. UP/ DOWN and RIGHT/ LEFT arrow keys can be used to go to next or previous memory location.

3) [F] FILL CONSTANT DATA INTO RAM:

Press [RESET] and [ENTER] to return to the command menu. Select [F] from this menu the display will show as follows:

```
VMC-8609..7FFF
F
```

Type the address and data in the manner shown below:

```
VMC-8609..7FFF
F0000:0400045057
```
Segment  Starting  Ending  Constant
Base     Address  Address  to be filled

Press [ENTER] and the value 57H will be stored at offset 0400H to 0450H.

4) [G] GO FOR RUN/ EXECUTION:

Press [RESET] and [ENTER] to return to the command menu. Select [G] from this menu the display will show as follows:

```
VMC - 8609 .. 7FFF
G
```

Type the Segment Address and Offset Address of the starting point of the program. Press [ENTER] [F7] and [ENTER] again thereafter in order to RUN the program. The code will run and the end results will be stored at the memory locations specified in the program which can be verified using [D] command.

5) [I] INTERRUPT:

Press [RESET] and [ENTER] to return to the command menu. Select [I] from this menu the display will show as follows:

```
VMC - 8609 .. 7FFF
INTP: 0000 . 0000 . 0000
```

First Int.  Second  Third Int.
Offset  Int. Offset  Offset
Address  Address  Address

Three INTERRUPTs (Offset Addresses) can be set in for the program execution, the CPU will continuously make a single step subprogram for checking IP values. When the IP registers has the same value as the INTERRUPTs addresses, it will enter the INTERRUPT’s subprogram.

6) [M] MOVING DATA:

The command MOVE is used to move data in the memory from a specified address to another address by inputting the starting address, the ending address and the desired target address. A return key is then used to execute the changes.
Press [RESET] and [ENTER] to return to the command menu. Select [M] from the menu, the display will show as follows:

```
Segment | Starting | Ending | Base | Offset | Offset
Source   |          |        |      |        |        
M  S    : 0000 : 0000 8088
T        : 0000 : 0000
```

Target

7) [P] PRINT:

This command allows the user to print the output. Printer to be connected to the I/O port of 8255.

Press [RESET] and [ENTER] to return to the command menu. Select [P] from the menu, the display will show as follows:

```
PRESS S / F FOR SPEED
```

If the printer is not connected at this time, press key [P] so the LCD would show ‘PRINTER ERROR!’ in the second line. Here S/F indicates ‘SLOW’ or ‘FAST’. Now according to the printer, press ‘S’ or ‘F’ key, then following display will be shown on the screen.

```
PRINTER ON !
```

UNASSEMBLE
DUMP
BUFFER

Description of the above commands is as follows:
[D].....command will allow the printer to print the machine code.
[U].....command will allow the printer to print the assembly program.
[B].....command sends data of RAM directly to the printer without going through any modification.
The starting address and ending address need to be entered first before using commands [B], [D] and [U]. Followed by an enter or ARROW UP key in order to print the output.

8) [T] TRACE PROGRAM (AN N-STEP DESIGNED COMMAND):

This command is used for program execution in any desired number of steps. TRACE will enter the interrupt subprogram every time the program is executed. N has a decimal range from 1-99 with 10 as the rounding off number and only operates if N is not 0; otherwise it will clear the function.

After feeding the program, Press [F7] key, then menu display will show as follows:

```
040E
>A. D. F. G. I. M. P. T. U.
```

Press the key [T], the screen displays as follows:

```
040E
T 00 -STEP
```

Here user can enter any number of steps from 00 to 99 and start execution using [G] command. For example, 01 is entered in place of 00 in the above command the program will be interrupted after execution of every instruction and if 02 entered in place of 00 then the program will be interrupted after execution of every two instructions and so on.

9) [U] UNASSEMBLE:

The UNASSEMBLE command decodes the value of a group of memory location mnemonics and displays on the display.

Press [RESET] and [ENTER] to return to the command menu. Select [U] from the menu, the display will show as follows:

```
VMC - 8609 . 7F FF
U 0400
```

Here 0400 is the address from where the assembly language code will be unassembled. Press [ENTER] and the display will show as:

```
Effective Address  Machine Code
0000:0400  B0 30
MOV A L, 30
```

Assembly mnemonics
User can use UP/ DOWN arrow keys to go to previous or next address respectively.

MEMORY MAPPING:

<table>
<thead>
<tr>
<th>Memory Locations</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000:0000 – 0000:FFFF</td>
<td>RAM AREA (Odd and Even RAM)</td>
</tr>
<tr>
<td>F000:0000 – F000:FFFF</td>
<td>ROM AREA (Odd and Even ROM)</td>
</tr>
</tbody>
</table>

RAM MEMORY

<table>
<thead>
<tr>
<th>Memory Locations</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000:0000</td>
<td>Interrupt Vector Section (INT1, INT2, INT3) have the arranged Interrupt Section and Stack Segment.</td>
</tr>
<tr>
<td>0000:0390</td>
<td>Buffer</td>
</tr>
<tr>
<td>0000:039B</td>
<td>System Data</td>
</tr>
<tr>
<td>0000:93E0</td>
<td>Buffer (Only if needed)</td>
</tr>
<tr>
<td>0000:0400 – 0000:7FFF</td>
<td>User RAM Area</td>
</tr>
</tbody>
</table>

I/O MAPPING

<table>
<thead>
<tr>
<th>Device Name</th>
<th>Port Name</th>
<th>Port Address</th>
<th>Connector</th>
</tr>
</thead>
<tbody>
<tr>
<td>8255-I</td>
<td>Port-A</td>
<td>70</td>
<td>CN3</td>
</tr>
<tr>
<td></td>
<td>Port-B</td>
<td>72</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Port-C</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CWR</td>
<td>76</td>
<td></td>
</tr>
<tr>
<td>8255-II</td>
<td>Port-A</td>
<td>80</td>
<td>CN4</td>
</tr>
<tr>
<td></td>
<td>Port-B</td>
<td>82</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Port-C</td>
<td>84</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CWR</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Port-A</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8255-III</td>
<td>8253</td>
<td>8259</td>
</tr>
<tr>
<td>----------------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>Port-B</td>
<td>Port-C</td>
<td>Counter 0</td>
<td>Data Word</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Counter 1</td>
<td>Command Word Register</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Counter 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Counter 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>00</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>02</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>04</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>06</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**RESULT:**

Basic architecture of 8086 and operation of VMC-8609 Microprocessor trainer kit successfully studied.
EXPERIMENT NO. 15

AIM:
Write a program using 8086 for addition of two 16 bit numbers.

APPARATUS:
8086 microprocessor kit, 5V power supply, Keyboard.

Program:

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Machine Code</th>
<th>Mnemonics</th>
<th>Operands</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>B8,34,12</td>
<td>MOV</td>
<td>AX,1234</td>
<td>Load 1234 in AX</td>
</tr>
<tr>
<td>1003</td>
<td>BA,65,87</td>
<td>MOV</td>
<td>DX,8765</td>
<td>Load 8765 in DX</td>
</tr>
<tr>
<td>1006</td>
<td>03,C2</td>
<td>ADD</td>
<td>AX,DX</td>
<td>Add DX with AX</td>
</tr>
<tr>
<td>1008</td>
<td>8B,C8</td>
<td>MOV</td>
<td>CX,AX</td>
<td>Move answer to CX</td>
</tr>
<tr>
<td>1009</td>
<td>CD,A5</td>
<td>INT</td>
<td>A5</td>
<td>Jump to command mode saving all registers.</td>
</tr>
</tbody>
</table>

CIRCUIT DIAGRAM / BLOCK DIAGRAM:-

```
START

Get the first No.

Get the second No.

Add. Two numbers

Store the result

END
```

PROCEDURE:-
1) Switch on the kit.
2) Go to assembler then type the mnemonics of our program in respective memory location.
3) Ensure the input weather placed in respective memory location.
4) Execute the program with starting address.
5) Verify the output data in respective memory location.

**INPUT DATA**

1000-1234(H)
1001-8765(H)

**OUTPUT DATA**

AX -9999(H)

**PRECAUTIONS:-**

Make sure that all the machine codes should be as per specified in the program.

**RESULT:-**

Thus the addition of two 16 bit numbers has performed and verified.