Page No.
Date 13 01 20

Experiment No. 1

AIM

To draw & explain
(1) Block diagram & pin diagram of 8085
(2) Instruction set of 8085

THEORY

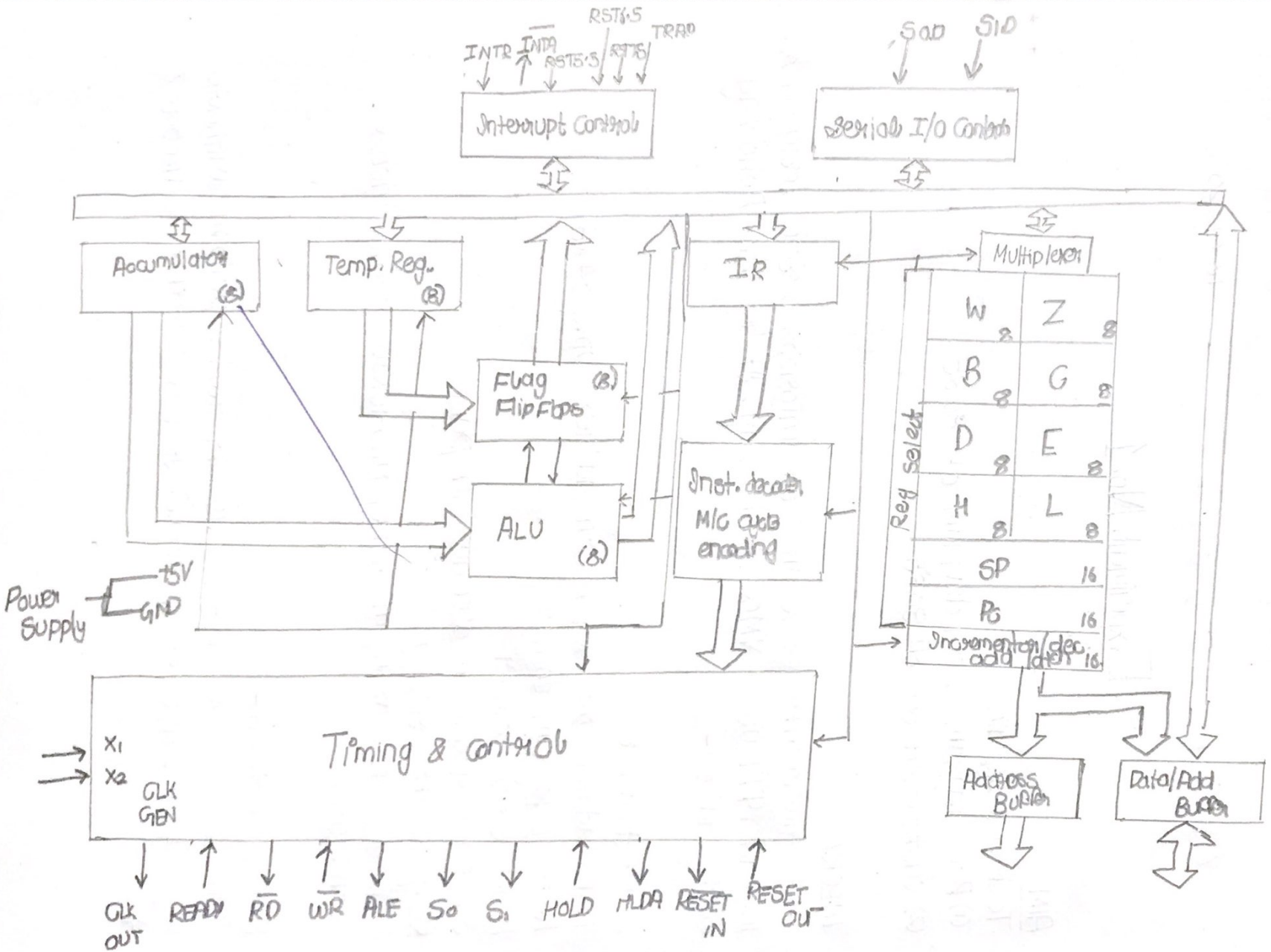
8085 microcontroller is an 8-bit microprocessor designed by intel in 1977 using NMOS technology. It has the following configuration -

- → 8-bit data bus
- → 16-bit address bus, which can address upto 64KB
- → 16-bit program counter
- → 16-bit stack pointer
- → Six 8-bit registers arranged in pairs
- → Requires +5V supply

It is used in washing machines, microwave ovens, mobile phones etc.

Functional Units -

Accumulator : It is an 8-bit register used to perform arithmetic logic, I/O & LOAD/STORE operations. It is connected to data bus & ALU


INTR
INTA
RST5.5
RST6.5
RST7.5
TRAP
SOD
SID
Interrupt Control
Serial I/O Control
Accumulator
(8)
Temp. Reg.
(8)
IR
Multiplexer
Flag Flip Flops
(8)
ALU
(8)
Inst. decoder M/C cycle encoding
W
Z
B
C
D
E
H
L
SP
PC
Reg select
Incrementer/decr. addr latch
16
Power Supply
+5V
GND
Timing & control
X1
X2
CLK GEN
Address Buffer
Data/Add Buffer
CLK OUT
READY
RD
WR
ALE
S0
S1
HOLD
HLDA
RESET IN
RESET OUT

Page No.
Date

Arithmetic Logic Unit : It performs arithmetic & logical operations like Addition, Subtraction, AND, OR etc. on 8bit data

General purpose register: There are 6 general purpose registers in 8085 processor ie. B, C, D, E, H & L. Each register can hold 8 bit data. These registers can work in pairs to hold 16 bit data & their pairing combination is like B-C, D-E & H-L.

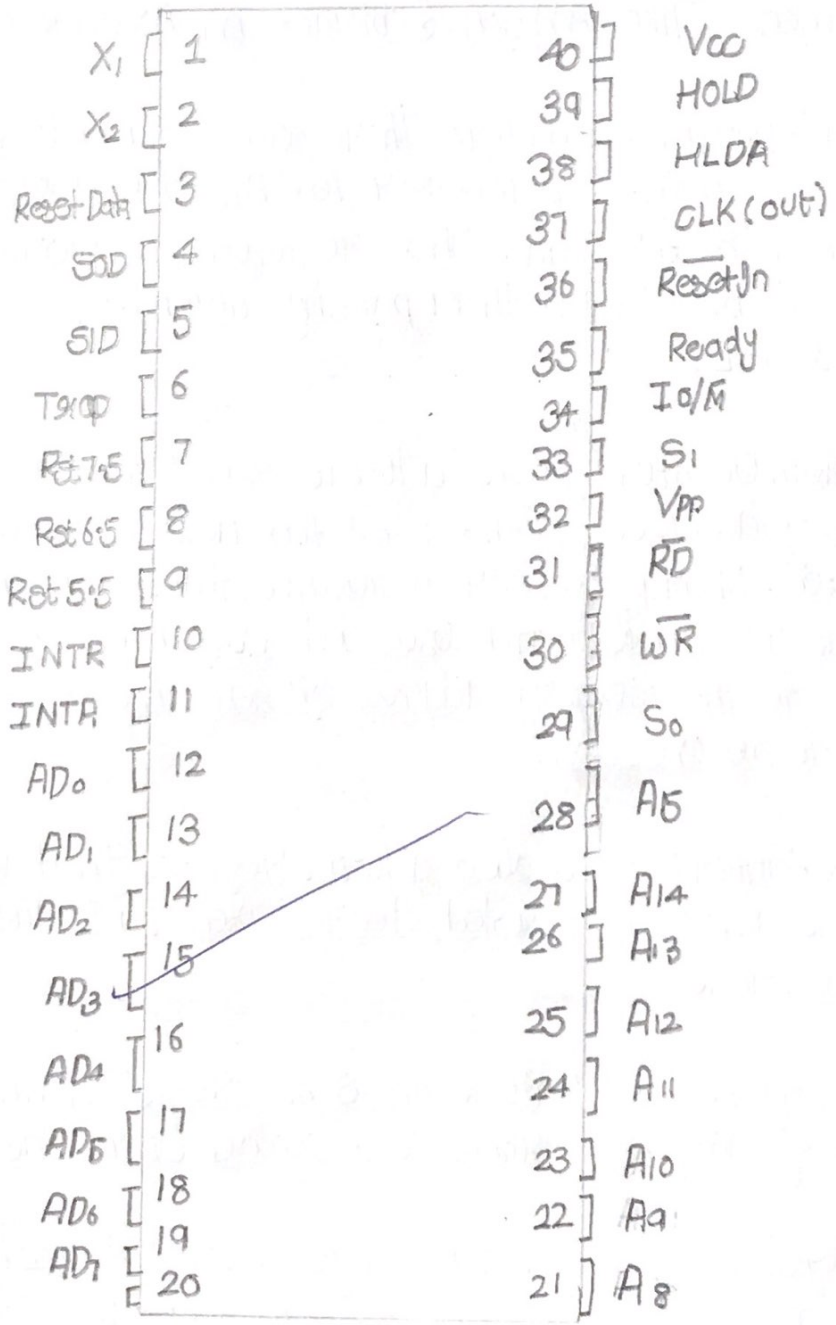
Program Counter : It is a 16-bit register used to store the memory address location of the next instruction to be executed. Microprocessor increments the program whenever an instruction is being executed, so that program counter points to the memory address of next instruction that is going to be executed.

Stack Pointer : It is also a 16-bit register that works like stack, which is always incremented/decremented by & during push & pop operations.

Temporary Register: It is an 8-bit register, which holds the temporary data of arithmetic & logical operations.

Flag Register : It is an 8-bit register having Five 1-bit Flip-Flop which holds either 0 or 1 depending upon the result stored in the accumulator

Pin Diagram of 8085 Microprocessor



Page No. _____
Date

Instruction Register & decoder: It is an 8-bit register, when an instruction is fetched from memory then it is stored in the instruction register. Instruction decoder decodes the information present in the instruction register.

Timing & Control Unit: It provides turning & control signals to the microprocessor to perform operations.
Control Signals : READY, RD', WR', ALE
Status Signals : S0, S1, IO/M'
DMA Signals : HOLD, HLDA
Reset Signals : RESET IN, RESET OUT

Interrupt Control: It controls the interrupts during a process when a microprocessor is executing a main program and whenever an interrupt occurs, the microprocessor shifts the control from the main program to process the incoming request. After the request is completed, the control goes back to the main program
5 interrupt signals : INTR, RST 7.5, RST 6.5, RST 5.5, TRAP

Serial Input/Output Control:
It controls the serial data communication by using these two instructions: SID (Serial Input Data) & SOD (Serial Output Data).

Page No.
Date

Address Buffer & Address-data buffer: The content stored in the stack pointer & program counter is loaded into the address buffer & address data buffer to communicate with the CPU. The memory & I/O chips are connected to these buses; the CPU can exchange the advised data with memory & I/O chips.

Address Bus & Data Bus: It causes the data to be stored. It is bidirection, whereas address bus causes the location to where it should be stored & it is unidirectional. It is used to transfer the data & Address I/O devices.

The pins of a 8085 microprocessor can be classified into 7 groups:

1. Address Bus - A15-A8, it carries the most significan 8-bits of memory I/O Address.

2. Data Bus - AD0-AD7, it carries the least significant 8-bit address & data bus.

3. Control & Status Signals - There are 3 control signal & 3 status signals.
RD: Signal indicates that the selected IO or memory device is to be used & is ready for accepting data available on data bus
WR: Signal indicates that the data on the data bus is to be written into a selected memory or IO location

Page No.
Date

ALE: It is a positive going pulse generated when a new operation is started by the microprocessor.
Three status signals are IO/M, S0 & S1

IO/M: This signal is used to differentiate between IO & Memory operations.

S0 & S1: These signals are used to identify type of current operation

Power supply: They are of two types, -Vcc & ground signal.
Clock Signals: X_1, X_2 - A crystal is connected to these two pins & is used to set Frequency of internal clock generator
CLK OUT: This signal is used as the system clock for devices connected with the microprocessor

Interrupts & externally initiated signals:
INTA: It is an interrupt acknowledgement signal.
RESET IN: It is used to reset the microprocessor by setting the counter to 0.
RESET OUT: It is used to reset all connected devices when the microprocessor is reset.
READY: It indicates that device is ready to send/receive data.
HOLD: It indicates that another master is requesting the use of the address & data busses.

HLDA (HOLD Acknowledge): It indicates that the CPU has received the hold request & it will relinquish the bus in the next clock signal.

Serial I/O Signals

SOD (Serial Output data line) - The output SOD is set/reset as specified by the SIM instruction

SID (Serial Input data line) - The data on this line is loaded into accumulator whenever a RIM instruction is executed.

CONSTRUCTION SET OF 8085:

Arithmetic	Data T/F	Func°	Branching Inst.		Logical	control
ADD	MOV	JMP	JC	RC	CMP	NOP
ADC	MVI	CALL	JNC	RNC	CPI	HLT
ADI	LDA	RST	JP	RP	ANA	DI
ACI	LDAX		JM	RM	ANI	EI
LXI	LXI		JZ	RZ	XRA	RIM
DAD	LHLD		JNZ	RNZ	XRI	SIM
SUB	STA		JPE	RPE	ORA	
SUI	STAX		JPO	RPO	ORI	
SBI	SHLD		CC	RST	RLC	
INR	XCHG		CNC		RRC	
INX	PUSH		CP		RAL	
DCR	POP		CM		RAR	
DCX	OUT		CZ		CMA	
DAA	IN		CNZ		CMC	
			CPE		STC	
			CPO			

Page No.
Date

Arithmetic Instructions:
ADD - To add register or memory to the accumulator
ADC - Add register to accumulator with carry
ADI - Add the immediate to the accumulator
DAD - Add the register pair to H&L registers
SUB - subtract the register/memory from accumulator
SBB - subtract the source & borrow from accumulator

Data Transfer Inst:
MOV - Copy from source to destination
MVI - Move immediate & bit.
LDA - Load the accumulator
LDAX - Load the accumulator indirectly
LXI - Load the register pair immediate
LHLD - Load H&L registers direct.
STA - 16-bit address

Branching Instructions:
JMP - Jump unconditionally
JC - Jump on carry
JNC - Jump on no carry
JP - Jump on positive
CC - Call on carry
CNC - call on no carry
CP - Call on positive
CM - Call on minus
RC - Return on carry

RP - Return on positive
RCHL - Load program counter with HL contents
RST - Restart

Logical Instructions :
CMP - Compare register/memory with accumulation
CPI - Compare Immediate with accumulator
ANA - Logical AND register/memory with accumulator
ANI - Logical AND immediate with accumulator
XRA - Exclusive OR register/memory with accumulator

Control Instructions :
NOP - No operation
HLT - Halt & enter wait state
DI - Disable Interrupts
EI - Enable Interrupts
RIM - Read Interrupts mask
SIM - Set Interrupts mask.

Result
We studied architecture, pin diagram & instruction set of 8085 microprocessor.

20/11/20

Page No.
Date 20 01 20

Experiment No. 2

AIM

Write a program of 8085 microprocessor
(a) To perform addition of two 8-bit no.s without carry.
(b) To perform addition of two 8-bit no.s with carry

SOFTWARE USED

GNUSim 8085

THEORY

a. Algorithm-

1. Load the First number to the accumulator through memory address 1.
2. Move the content of accumulator to register B.
3. Load the second number to accumulator through memory address 2.
4. Add the content of accumulator & register B and result will be stored in accumulator
5. Store the result from accumulator to memory address 3
6. Terminate the program.

Instructions-
LDA - Load number from memory to accumulator
MOV - Move content of accumulator to register B

Program Code

a. LDA 1
MOV B,A
LDA 2
ADD B
STA 3
HLT

b. LDA 1
MOV H,A
LDA 2
ADD H
MOV L,A
MVI A 00
ADC A
MOV H,A
SHLD 3
HLT

Page No.
Date

ADD - Will sum content of accumulator to memory register B.
STA - Store content of accumulator to memory
HLT - will terminate the program

b. Algorithm -

1. Load the First number From memory location 1 to accumulator
2. Move the content of accumulator to register H.
3. Load the second no. From memory location 2 to accumulator
4. Then add the content of register H & accumulator using "ADC" command & is stored at location 4,

Instructions -
MVI A 00 - moves intermediate data (ie 00 to A)
ADC - Add content of A (00), contents of register specified ie A & carry, A be default an operand & stores the result.
SHLD - moves content of L register in 3 memory location & content of H register in 4 memory location.

RESULT
We performed addition of two 8 bit no.s with & without carry.

15

8/11/22

Page No.
Date 27 01 20

Experiment No. 3

AIM

Write a program to perform:
(a) Subtraction of two 8 bit numbers without borrow.
(b) Subtraction of two 8 bit numbers with borrow

Software Used

GNUSim 8085

Theory

Algorithm

1. Load 00 in register C.
2. Load two 8 bit number from memory into registers.
3. Move one number to accumulator
4. Subtract the second number with accumulator
5. If borrow isn't equal to 1, go to step 6
6. Store accumulator content in memory
7. Move content of register into accumulator
8. Store content of accumulator in other memory location.

Program Code: (Approach 1)

```
lda 0
mov B,A
lda 1
sub B
sta 2
sbb B
sta 3
hlt
```

(Approach 2)

```
MVI     C,00
LHLD    2
MOV     A,H
SUB     L
JNC     200 B
INR     C
STA     3
MOV     A,C
STA     4
HLT
```

Page No.
Date

* LHLD - It is used to load register pair directly using 16-bit address
* INR - is used to increase register by 1 (increment)
* JNC - It is used to Jump if no borrow
* SUB - used to subtract two numbers where one number is in accumulator
* HLT - is used to halt the program

RESULT

We studied about subtraction of two 8 bit numbers with & without borrow.

3/2/22

Page No.
Date 03 02 20

Experiment No. 4

AIM

Write a program to perform multiplication of two 8 bit numbers.

SOFTWARE USED

GNUSim8085

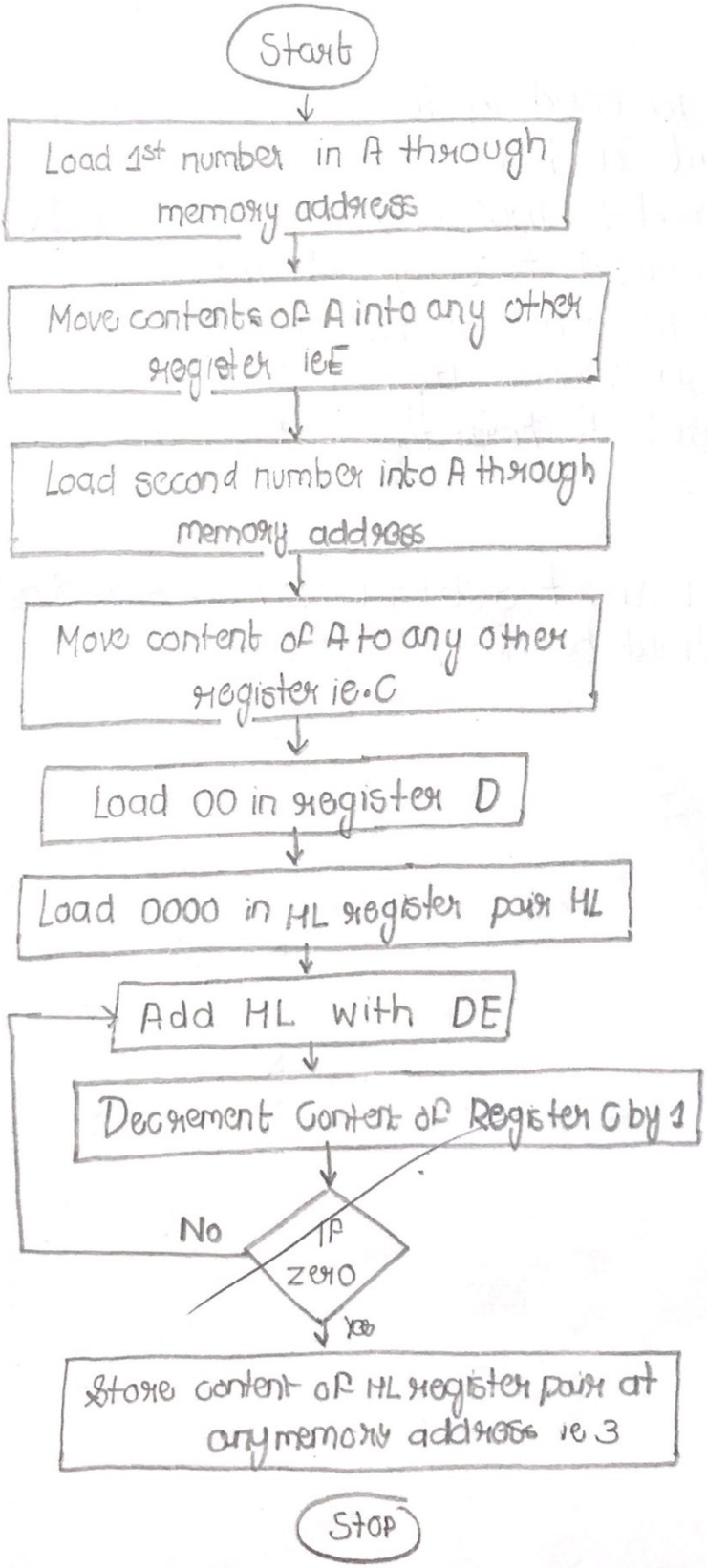
THEORY

In 8085 Microprocessor there is no command to perform multiplication. Multiplication is done through repeated addition.

Algorithm:

1. Load content at address 1 to accumulator & move to register E.
2. Load content at address 2 to accumulator & move to register C.
3. Move Immediate 8-bit to D as 00
4. Load register pair HL as 0000.
5. Add register pairs H and L to D & E.
6. Decrement C
7. Jump on no zero to perform repeated addition.
8. Store H&L registers direct at address 3
9. Halt the program.

Flowchart :


Start
Load 1st number in A through memory address
Move contents of A into any other register ie E
Load second number into A through memory address
Move content of A to any other register ie. C
Load 00 in register D
Load 0000 in HL register pair HL
Add HL with DE
Decrement Content of Register C by 1
No
IF zero
Yes
Store content of HL register pair at any memory address ie 3
Stop

Program Code :

```
LDA 1
MOV E,A
LDA 2
MOV C,A
LVI D,00
LXI H,0000
XUV: DAD D
DCR C
JNZ XUV
SHLD 3
HLT
```

Output:

	Address	Data
Input:	1	2
	2	4
Output:	3	8

Registers:

A		0A
BC	00	00
DE	00	02
HL	00	08

Flags:

S	0
Z	1
AC	0
P	1
C	0

Page No.
Date

RESULT:
We studied how to multiply two 8 bit numbers.

Page No. _____
Date 10 02 20

Experiment No. 5

Aim:
Write a program to Find smallest & largest number From the given series.

Software Used:
GNU Sim 8085

Theory:
We assume the list of numbers to be -

42H, 21H, 01H, 1FH, FFH, 25H, 32H, 34H, 0AH, ABH
2050H 2051H ---------- 2059H

hence we must get Minimum as 01H (2061H) & Maximum as FFH (2060H)

In CMP Instruction
IF Accumulator > Register then carry & zero Flags are reset
IF Accumulator = Register then zero Flag is set
IF Accumulator < Register then carry Flag is set

Algorithm:
1. Max no. is stored in register B & min. in register C
2. Load counter in D Register
3. Load starting element in Accumulator, B & C register
4. Compare Accumulator & B register
5. IF carry Flag is not set then transfer contents of Accumulator to B. Else, compare accumulator with Register C,

Program Code:

ADDRESS	LABEL	INSTRUCTION	COMMENT
2000H		LXI H, 2050H	Load starting address of list
2003H		MOV B, M	Store Maximum
2004H		MOV C, M	Store Minimum
2005H		MVI D, 0AH	Counter for 10 elements
2007H	LOOP	MOV A, M	Retieve list element in Accumulator
2008H		CMP B	Compare element with maximum
2009H		JC MIN	Jump to MIN if not maximum
200CH		MOV B, A	Transfer contents of A to B as A>B
200DH	MIN	CMP C	Compare elements with min number
200EH		JNC SKIP	Jump to skip if not maximum
2011H		MOV C, A	Transfer contant of A to C if A<min
2012H	SKIP	INX H	Increment memory
2013H		DCR D	Decrement Counter
2014H		JNZ LOOP	Jump to LOOP if D>0
2017H		LXI H, 2060H	Load address to store max
201AH		MOV M, B	Move max to 2060H
201BH		INX H	Increment Memory
201CH		MOV M, C	Move min to 2061H
201DH		HLT	Halt

Output:

Minimum: 01H at 2061H
Maximum: FFH at 2060H

Page No.
Date

if carry Flag is set then transfer contents of accumulator to C

6. Decrement D register
7. If D>0 then take next element in accumulator & go to point 4.
8. If D=0, store B&C register in memory
9. End of program

Result:
We found the minimum & maximum number from the given series.

Experiment No. 6

Aim: Write a program to find sum of series of n consecutive numbers

Software Used: GNUSim8085

Theory: The sum of first n natural numbers is given by the formula-

$$S_n = \frac{n(n+1)}{2}$$

1. With n as the input, increment it to obtain n+1.
2. Multiply n with n+1.
3. Divide the product obtained by 2.

In 8085 microprocessor, no direct instruction exists to multiply two numbers, so multiplication is done by repeated addition as 4×5 is equivalent to 4+4+4+4+4 (i.e., 5 times).
Input: 04H
Add 04H 5 times
Product: 14H(20_{10})
Similarly, in 8085 microprocessor, no direct instruction exists to divide two numbers, so division is done by repeated subtraction.
Input: 14H
Keep subtracting 2 from the input till it reduces to 0.
Since subtraction has to be performed 10_{10} times before 14H becomes 0, the quotient is 10_{10} => 0AH.

Algorithm:

1. Load the data from the memory location (201BH, arbitrary choice) into the accumulator
2. Move this data into B
3. Increment the value in the accumulator by one and move it to the register C
4. Initialise the accumulator with 0

5. Multiplication: Keep adding B to accumulator. The number of times B has to be added is equal to the value of C
6. Initialise B with 00H. B will store the quotient of the division
7. Initialise C with 02H. This is the divisor for the division
8. Division: Keep subtracting C from A till A becomes 0. For each subtraction, increment B by one
9. The final answer is in B. Move it to A. Then store the value of A in 201CH (arbitrary choice again)
10. 201CH contains the final answer.

Program Code:

ADDRESS	LABEL	MNEMONIC
2000H		LDA 201BH
2001H		
2002H		
2003H		MOV B, A
2004H		INR A
2005H		MOV C, A
2006H		MVI A, 00H
2007H		
2008H	LOOP1	ADD B
2009H		DCR C
200AH		JNZ LOOP1
200BH		
200CH		
200DH		MVI C, 02H
200EH		
200FH		MVI B, 00H
2010H		
2011H	LOOP2	INR B
2012H		SUB C
2013H		JNZ LOOP2
2014H		

2015H	
2016H	MOV A, B
2017H	STA 201CH
2018H	
2019H	
201AH	HLT

Output:

Store the value of n in 201BH. The sum can be found at 201CH.

Result:

We learned how to find sum of series of n consecutive numbers

Experiment No. 7

Aim: Write a program to find factorial of a given number

Software Used: GNUSim8085

Theory:

In 8085, there is no direct instruction to perform multiplication. We need to perform repetitive addition to get the result of the multiplication. In each step we are decreasing the value of B and multiply with the previous value of B. We are repeating these steps until B reaches 1. and B – 1 to 0. Thus, the factorial is generated.

Algorithm:

1. Load the data into register B
2. To start multiplication set D to 01H
3. Jump to step 7
4. Decrements B to multiply previous number
5. Jump to step 3 till value of B>0
6. Take memory pointer to next location and store result
7. Load E with contents of B and clear accumulator
8. Repeatedly add contents of D to accumulator E times
9. Store accumulator content to D
10. Go to step 4

Program Code:

Address	Label	Mnemonic	Comment
2000H	Data		Data Byte
2001H	Result		Result of factorial
2002H		LXI H, 2000H	Load data from memory

2005H		MOV B, M	Load data to B register
2006H		MVI D, 01H	Set D register with 1
2008H	FACTORIAL	CALL MULTIPLY	Subroutine call for multiplication
200BH		DCR B	Decrement B
200CH		JNZ FACTORIAL	Call factorial till B becomes 0
200FH		INX H	Increment memory
2010H		MOV M, D	Store result in memory
2011H		HLT	Halt
2100H	MULTIPLY	MOV E, B	Transfer contents of B to C
2101H		MVI A, 00H	Clear accumulator to store result
2103H	MULTIPLYLOOP	ADD D	Add contents of D to A
2104H		DCR E	Decrement E
2105H		JNZ MULTIPLYLOOP	Repeated addition
2108H		MOV D, A	Transfer contents of A to D
2109H		RET	Return from subroutine

Input:

Address	Data
2000H	04H

Output:

Address	Data
2001H	18H

Result:

We learned how to find factorial of a given number of 8085 Microprocessor in Assembly language.

Experiment No.8

Aim: Write a program to reverse an 8-bit number

Software Used: GNUSim8085

Theory:

Here the task is too simple. There are some rotating instructions in 8085. The RRC, RLC are used to rotate the Accumulator content to the right and left respectively without carry. We can use either RRC or RLC to perform this task.

Algorithm:

1. Load content of memory location 2050 in accumulator A
2. Use **RLC** instruction to shift content of A by 1 bit without carry. Use this instruction 4 times to reverse the content of A
3. Store content of A in memory location 3050

Program Code:

MEMORY ADDRESS	MNEMONICS	COMMENT
2000	LDA 2050	A <- M[2050]
2003	RLC	Shift content of accumulator left by 1 bit without carry
2004	RLC	Shift content of accumulator left by 1 bit without carry
2005	RLC	Shift content of accumulator left by 1 bit without carry
2006	RLC	Shift content of accumulator left by 1 bit without carry
2007	STA 3050	M[2050] <- A
200A	HLT	END

Input:

Address	Data
2050	98

Output:

Address	Data
3050	89

Result:

We learned how to reverse a number of 8085 Microprocessor in Assembly language.

Innovation

Aim: Write a Program in 8085 Assembly language for sorting a list of numbers

Software Used: GNUSim8085

Theory:

We sort the list of numbers using Bubble Sorting technique. Bubble sort is a simple sorting algorithm. This sorting algorithm is comparison-based algorithm in which each pair of adjacent elements is compared and the elements are swapped if they are not in order.

This algorithm is not suitable for large data sets as its average and worst-case complexity are of $O(n^2)$ where **n** is the number of items.

Algorithm:

1. Initialize HL pair as memory pointer
2. Get the count at 4200 into C – register
3. Copy it in D - register (for bubble sort (N-1) times required)
4. Get the first value in A – register
5. Compare it with the value at next location
6. If they are out of order, exchange the contents of A - register and Memory
7. Decrement D - register content by 1
8. Repeat steps 5 and 7 till the value in D- register become zero
9. Decrement C - register content by 1
10. Repeat steps 3 to 9 till the value in C - register becomes zero

Program Code:

```
        LXI H,5000          Set pointer for array
        MOV C, M            Load the Count
        DCR C               Decrement Count
REPEAT: MOV D, C
        LXI H,5001
LOOP:   MOV A, M            Copy content of memory location
                            to Accumulator
        INX H
        CMP M
        JC SKIP             Jump to skip if carry generated
        MOV B, M            Copy content of memory location to B -
                            Register
        MOV M, A            Copy content of Accumulator to memory
                            location
        DCX H               Decrement content of HL pair of registers
        MOV M, B            Copy content of B - Register to
                            memory location
        INX H               Increment content of HL pair of registers

SKIP:   DCR D               Decrement content of Register - D
        JNZ LOOP            Jump to loop if not equal to zero
        DCR C               Decrement count
        JNZ REPEAT          jump to repeat if not equal to zero
        HLT                 Terminate Program
```

Input:

Data 0: 05H in memory location 5000

Data 1: 05H in memory location 5001

Data 2: 04H in memory location 5002

Data 3: 03H in memory location 5003

Data 4: 02H in memory location 5004

Data 5: 01H in memory location 5005

Output:

Data 0: 05H in memory location 5000

Data 1: 01H in memory location 5001

Data 2: 02H in memory location 5002

Data 3: 03H in memory location 5003

Data 4: 04H in memory location 5004

Data 5: 05H in memory location 5005

Result:

We sorted the given numbers in ascending order for 8085 Microprocessor in Assembly Language successfully.