AARUPADAI VEEDU INSTITUTE OF TECHNOLOGY
PAIYANOOR

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

MICROCONTROLLERS LAB

2017R

LAB MANUAL

Experiment No: **Date:**

ADDITION OF TWO 8-BIT NUMBERS

AIM:

Write a program to add two 8 bit numbers and store the result in memory location 4200 and 4201.

APPARATUS REQUIRED:

(i) 8085 Microprocessor kit with keyboard
(ii) Power cable.

ALGORITHM:

Step 1: Initialize register C to account for carry.

Step 2: The addend is brought to the accumulator.

Step 3: Move the augend to any one of the register.

Step 4: Perform the addition.

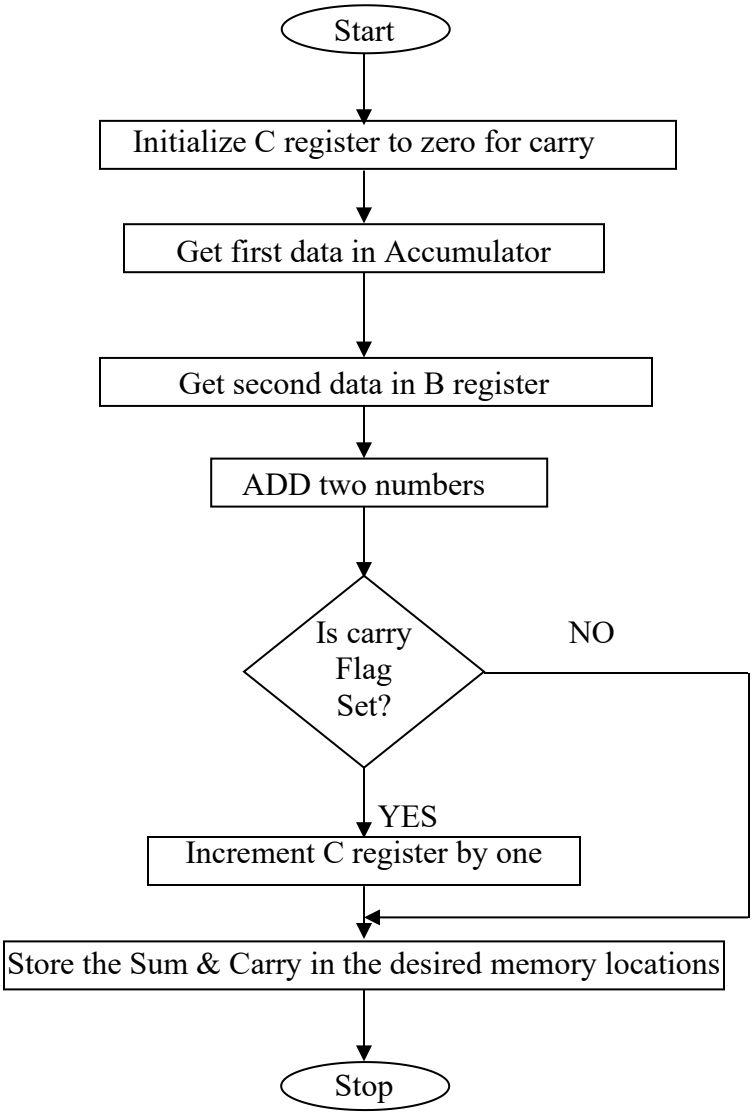
Step 5: Store the 8 bit and carry in two memory locations.

Step 6: Stop the process.

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	4150	FF	Addend
	4151	FF	Augend
OUTPUT	4200	FE	8 Bit Sum
	4201	01	Carry

FLOWCHART-ADDITION OF TWO 8-BIT NUMBERS



PROGRAM FOR 8 BIT ADDITIONS:

LABEL	ADDRESS	MNEMONIC	OPCODE	OPERAND	COMMENTS
START	4100H	MVI C,00H	0E	00	Initialize C reg. to store carry.
	4102H	LXI H,4150H	21	50,41	Initialize HL reg. pair to point 1st data.
	4105H	MOV A,M	7E		Move 1st data to accumulator.
	4106H	INX H	23		Increment HL reg. pair to point 2nd data.
	4107H	MOV B,M	46		Move 2nd data to B reg.
	4108H	ADD B	80		Add two numbers in A & B.
	4109H	JNC LOOP	D2	0D,41	Check whether Carry Flag is reset.
	410CH	INR C	0C		Increment C register.
LOOP	410DH	STA 4200H	32	00,42	Store 8 bit sum to memory location 4200.
	4110H	MOV A,C	79		Move carry to Accumulator.
	4111H	STA 4201H	32	01,42	Store carry to memory 4201.
	4114H	HLT	76		Stop the process.

RESULT:

Thus the program for addition of two 8-bit numbers has been written and executed and the sum is verified.

Experiment No: **Date:**

SUBTRACTION OF TWO 8-BIT NUMBERS

AIM:

Write a program to subtract two 8 bit numbers and store the result in memory location 4502 and 4503.

APPARATUS REQUIRED:

(i) 8085 Microprocessor kit with keyboard
(ii) Power cable

ALGORITHM:

Step 1: Initialize register C to account for Borrow.

Step 2: The Minuend is brought to the accumulator.

Step 3: Move the subtrahend to anyone register.

Step 4: Perform the subtraction.

Step 5: If the carry flag is reset go to step 7, else proceed to the next step.

Step 6: Take 2's complement of the accumulator content and increment C register.

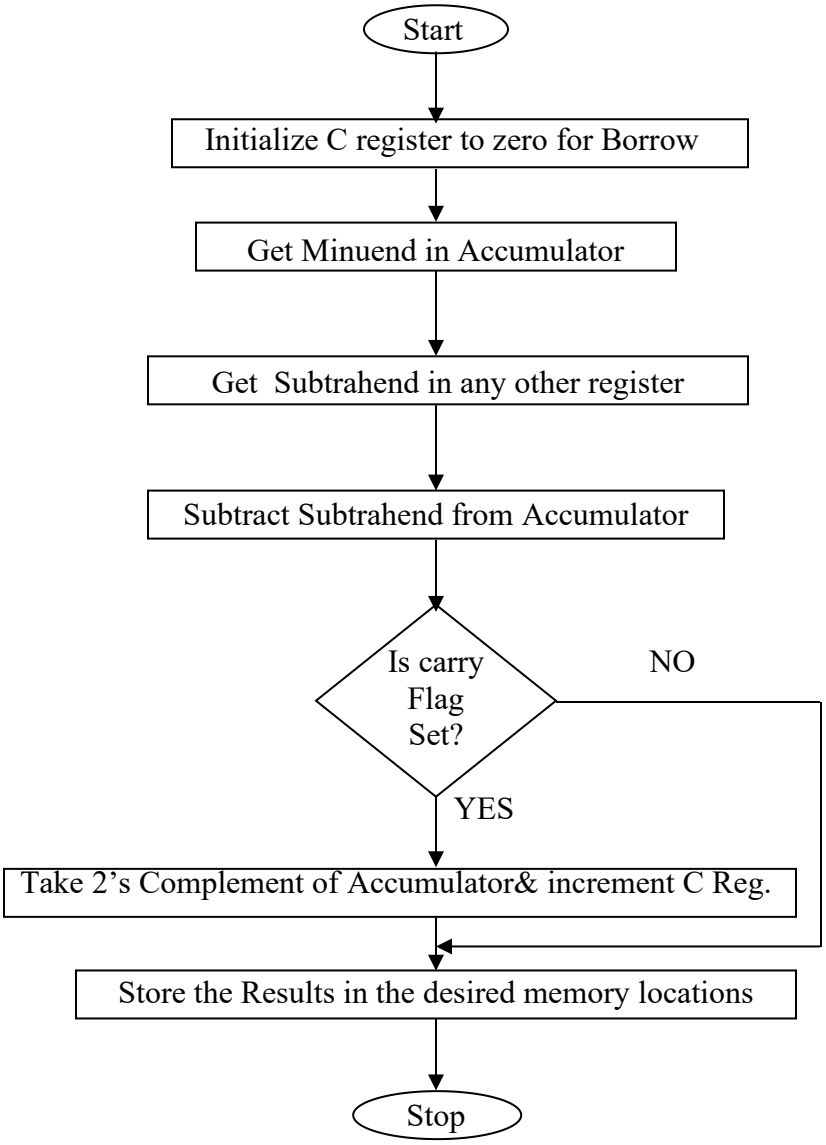
Step 7: Store the difference & borrow in memory location 4502 and 4503.

Step 8: Stop the process.

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	4500	FA	Minuend
	4501	10	Subtrahend
OUTPUT	4502	EA	Difference
	4503	00	Borrow

FLOWCHART FOR 8-BIT SUBTRACTION



PROGRAM FOR 8-BIT SUBTRACTION

LABEL	ADDRESS	MNEMONIC	OPCODE	OPERAND	COMMENTS
START	4300H	MVI C,00H	0E	00	Initialize C reg to store carry.
	4302H	LXI H,4500H	21	00,45	Initialize HL reg pair to point 1st data.
	4305H	MOV A,M	7E		Move 1st data to accumulator.
	4306H	INX H	23		Increment HL reg pair to point 2nd data.
	4307H	MOV B,M	46		Move 2nd to register B.
	4308H	SUB B	90		Subtract the subtrahend from minuend.
	4309H	JNC SKIP	D2	10,43	If Carry Flag reset, go to skip.
	430CH	CMA	2F		If the Subtraction result in borrow take 2's complement of the accumulator
	430DH	ADI 01H	C6	01	
	430FH	INR C	0C		Increment register C
SKIP	4310H	INX H	23		Store the result in the desired memory location
	4311H	MOV M,A	77		
	4312H	INX H	23		Store the carry flag to indicate whether the result is positive or negative
	4313H	MOV M,C	71		
	4314H	HLT	76		Stop the process.

RESULT:

Thus the subtraction program has been written & executed and the difference and borrow were verified.

Experiment No: **Date:**

MULTIPLICATION OF TWO 8-BIT NUMBERS

AIM:

To write an Assembly Language Program for the multiplication of two 8-bit numbers.

APPARATUS REQUIRED:

(i) 8085 Microprocessor kit with keyboard
(ii) Power cable.

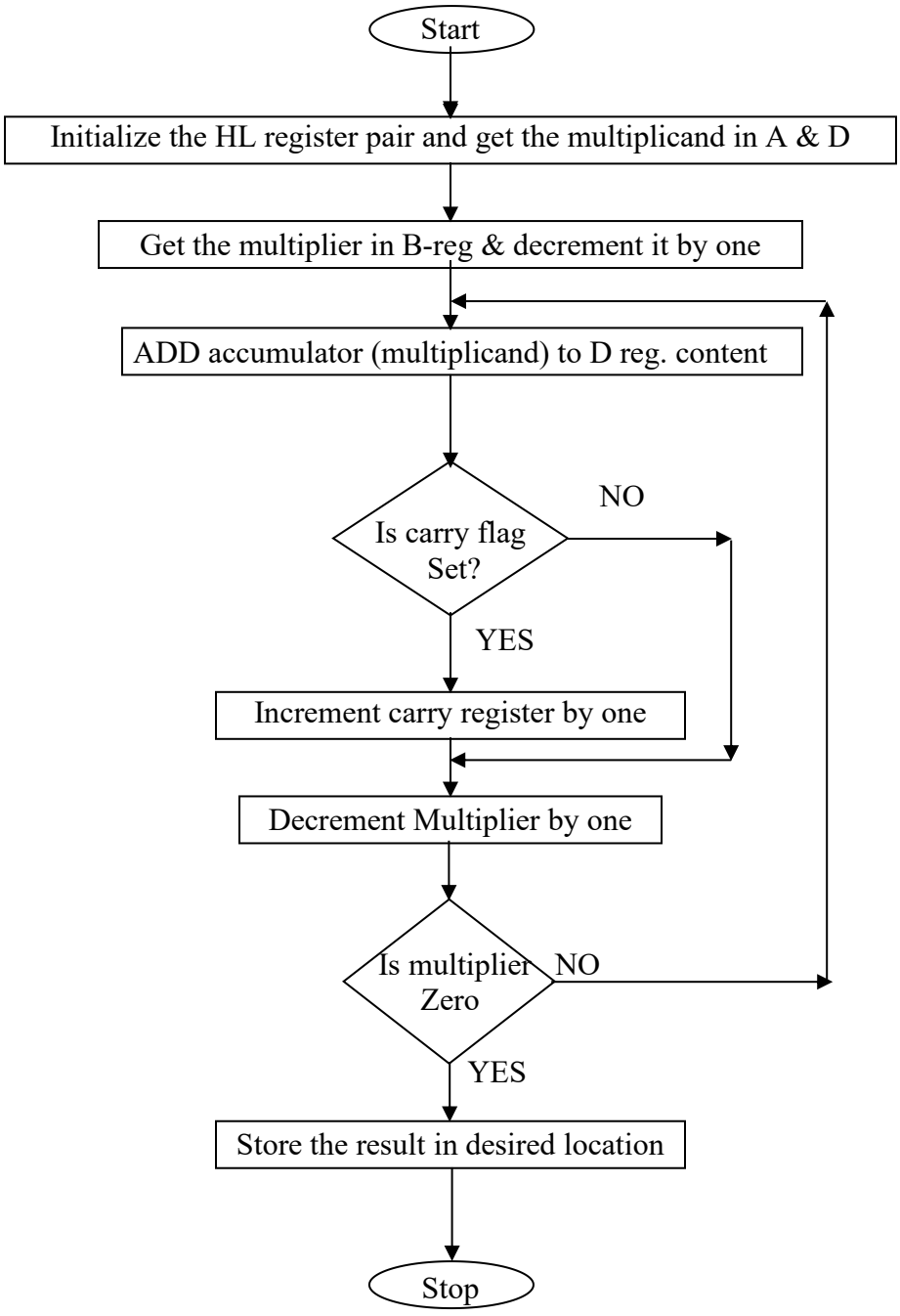
ALGORITHM:

Step 1: Initialize the HL registers pair with the address 4200H.
Step 2: Clear C register.
Step 3: Get the multiplicand in accumulator and get the multiplier in B-reg.
Step 4: Move the Multiplicand to D register.
Step 5: Add the content of A register with the content of D register.
Step 6: Check the carry flag if CY=0 go to step 7 otherwise increment the C register.
Step 7: Decrement the B register.
Step 8: If the content of B reg is not equal to zero go to step 5, otherwise go to next step.
Step 9: Store the content of accumulator in Next memory location.
Step 10: Store the higher order byte in next memory location.
Step 11; Stop the Process.

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	4200H	12	Multiplicand
	4201H	11	Multiplier
OUTPUT	4202H	32	LSB of Product
	4203H	01	MSB of Product

FLOWCHART FOR 8-BIT MULTIPLICATION



PROGRAM FOR 8-BIT MULTIPLICATION

LABEL	ADDRESS	MNEMONIC	OPCODE	OPERAND	COMMENTS
START	4100H	LXI H,4200H	21	00,42	Load the HL reg pair with the address 4200H
	4103H	MVI C,00H	0E	00	Clear C reg
	4105H	MOV A,M	7E		Move the Multiplicand to the accumulator.
	4106H	INX H	23		Increment the HL reg pair.
	4107H	MOV B,M	46		Move the Multiplier to B reg.
	4108H	DCR B	05		Decrement B reg.
	4109H	MOV D,A	57		Move the Multiplicand to D reg.
LOOP2	410AH	ADD D	82		Add the content of A reg with D reg. content.
	410BH	JNC LOOP1	D2	0F,41	If CY=0, program control jumps to label LOOP1.
	410EH	INR C	0C		Increment C reg.
LOOP1	410FH	DCR B	05		Decrement B reg.
	4110H	JNC LOOP2	C2	0A,41	If B is not equal to zero, program control goes to LOOP2; otherwise do next step
	4113H	INX H	23		Increment the HL reg. pair.
	4114H	MOV M,A	77		Store the least significant byte of the product to the memory location pointed by HL pair.
	4115H	INX H	23		Increment the HL reg.
	4116H	MOV M,C	71		Store the most significant byte of the product to the memory location pointed by HL pair.
	4117H	HLT	76		Stop the Process

RESULT:

Thus the Assembly Language Program for the multiplication of two 8-bit data had been written and executed & the results were stored in desired memory locations.

Experiment No: **Date:**

DIVISION OF TWO 8-BIT NUMBERS

AIM:

To write an Assembly Language Program for dividing two 8-bit numbers and store the result in memory locations 4502 and 4503.

APPARATUS REQUIRED:

(i) 8085 Microprocessor kit with keyboard
(ii) Power cable.

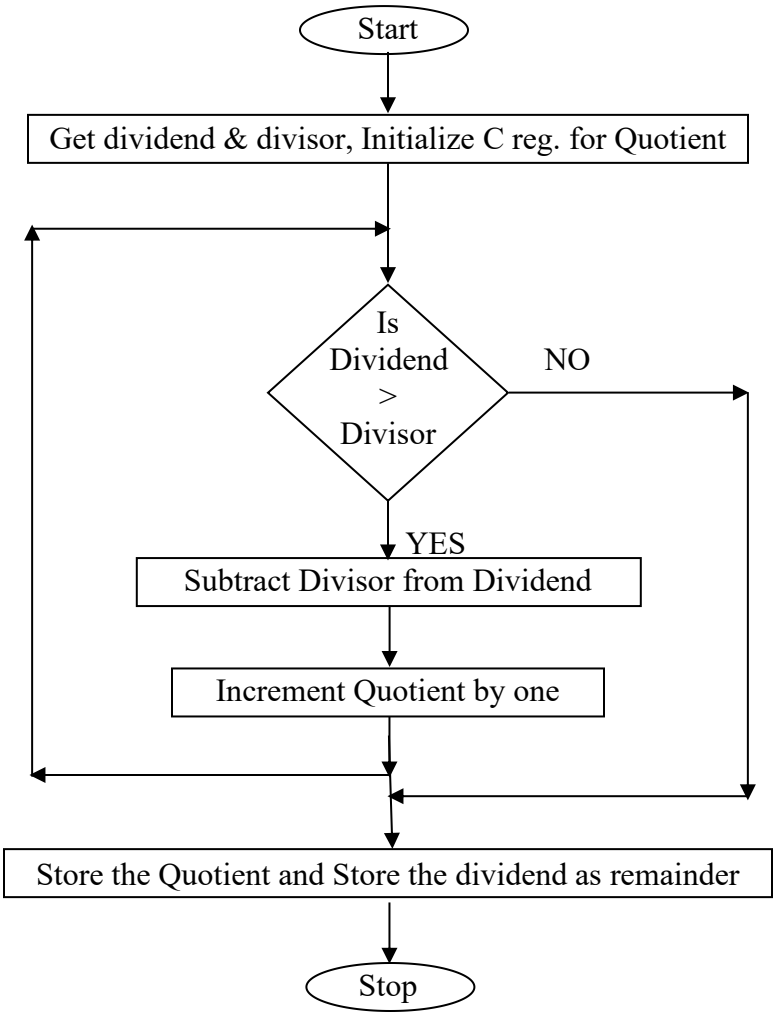
ALGORITHM:

Step 1: Initialize the HL register pair with address 4500H.
Step 2: Clear the C register.
Step 3: Get dividend in accumulator & get divisor in B register.
Step 4: Compare dividend and divisor.
Step 5: If carry occurs, go to step 8. Otherwise go to next step.
Step 6: Subtract divisor from dividend and increment C register.
Step 7: Go to step 4.
Step 8: Store the remainder in 4503H.
Step 9: store the quotient in 4502H.
Step 10: Stop the Process.

	ADDRESS	DATA	COMMENTS
INPUT	4500H	40	Dividend
	4501H	20	Divisor
OUTPUT	4502H	02	Quotient
	4503H	00	Remainder

FLOWCHART FOR 8-BIT DIVISION

\



PROGRAM FOR 8-BIT DIVISION

LABEL	ADDRESS	MNEMONIC	OPCODE	OPERAND	COMMENTS
START	4100H	LXI H,4500H	21	00,45	Load the HL reg pair with the address 4500H.
	4103H	MVI C,00H	0E	00	Clear C reg.
	4105H	MOV A,M	7E		Move the Dividend to the accumulator.
	4106H	INX H	23		Increment the HL reg pair.
	4107H	MOV B,M	46		Move the Divisor to B reg.
REPEAT	4108H	CMP B	B8		Compare B reg.
	4109H	JC SKIP	DA	11,41	If carry=1, jump to label SKIP.
	410CH	SUB B	90		Subtract the divisor from dividend.
	410DH	INR C	0C		Increment C reg.
	410EH	JMP REPEAT	C3	08,41	Jump to label REPEAT.
SKIP	4111H	STA 4503H	32	03,45	Store the remainder.
	4114H	MOV A,C	79		Move the Quotient to Accumulator.
	4115H	STA 4502H	32	02,45	Store the Quotient.
	4118H	HLT	76		Stop the Process

RESULT:

Thus the Assembly Language Program for the Division of two 8-bit data had been written and executed & the results were stored in desired memory locations.

Experiment No: **Date:**

LARGEST NUMBER IN GIVEN ARRAY

AIM:

To write an Assembly Language Program to find the largest number in the given Array.

APPARATUS REQUIRED:

(i) 8085 Microprocessor kit with keyboard.
(ii) Power cable.

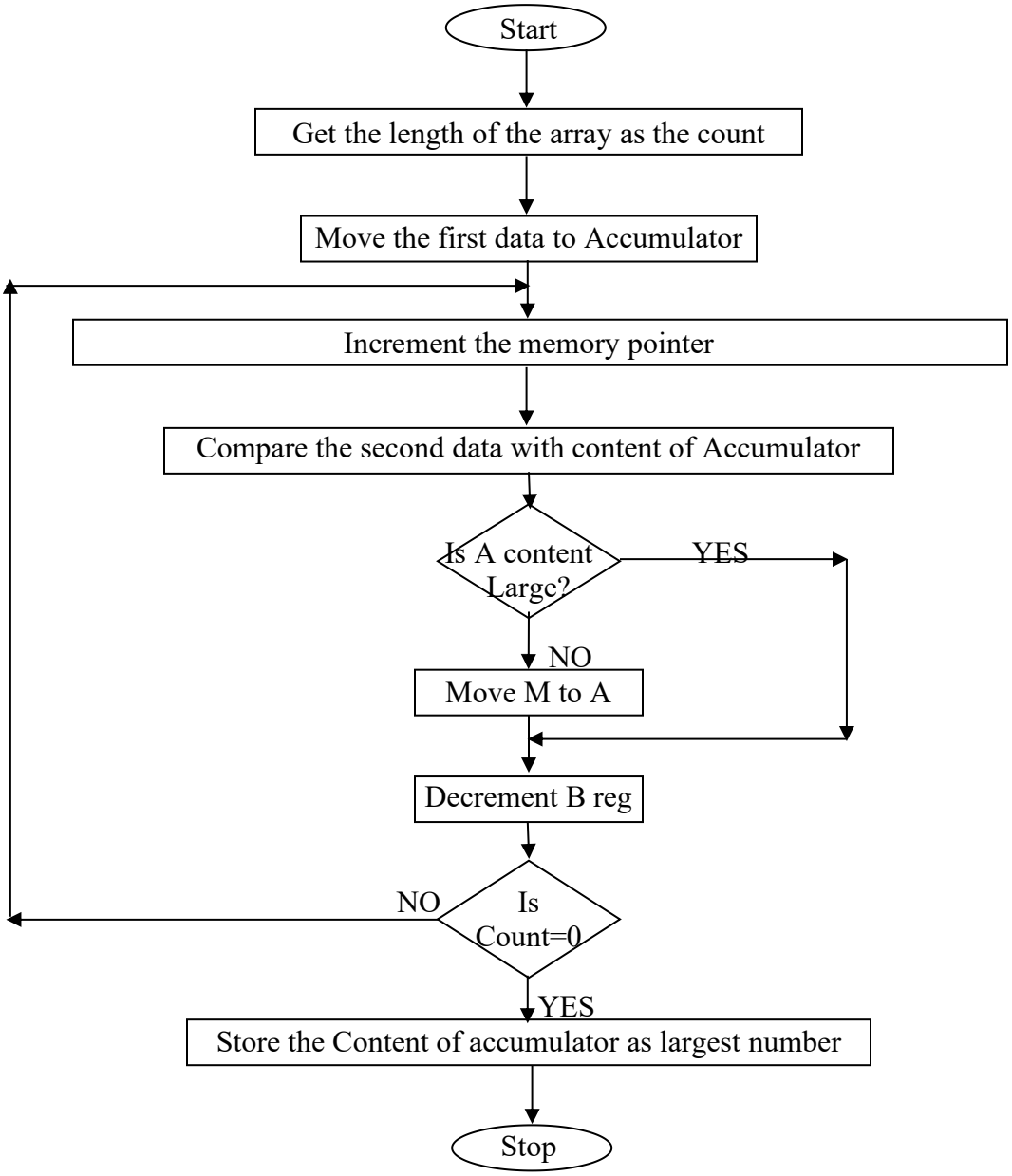
ALGORITHM:

Step 1: Initialize the HL register pair with the address 4700H.
Step 2: Get the first Data in accumulator.
Step 3: Get the count (i.e., Length of the array) in B register.
Step 4: Compare the next data from the memory address pointed by HL pair to the accumulator.
Step 5: Check the Carry Flag. If the CF is reset go to step 7 otherwise go to next step.
Step 6: Move the data from the memory address pointed by HL pair to the accumulator.
Step 7: Decrement the count (content of B register)
Step 8: If the content of B register is not equal to zero, go to step 4, otherwise go to next step.
Step 9: Store the content of accumulator in the address 5000H.
Step 10: Stop the Process.

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	4700H	05	Total number of Data
	4701H	0E	First input data
	4702H	0C	Second input data
	4703H	0F	Third input data
	4704H	03	Fourth input data
	4705H	09	Fifth input data
OUTPUT	5000H	0F	Largest Data

F**LOWCHART:**


Start
Get the length of the array as the count
Move the first data to Accumulator
Increment the memory pointer
Compare the second data with content of Accumulator
Is A content Large?
YES
NO
Move M to A
Decrement B reg
Is Count=0
NO
YES
Store the Content of accumulator as largest number
Stop

PROGRAM:

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	4100	LXI H, 4700H	21	00, 47	Initialize the memory pointer with the address 4700H
	4103	MOV B,M	46		Get the length of the array in B reg.
	4104	INX H	23		Increment the HL reg. pair to point the next memory location.
	4105	MOV A,M	7E		Get the first data in Accumulator.
	4106	DCR B	05		Decrement B reg. content by one
LOOP	4107	INX H	23		Increment the HL reg. pair to point the next memory location.
	4108	CMP M	BE		Compare the data in accumulator with data in memory.
	410B	JNC SKIP	D2	0D, 41	If CF is reset, the program control goes to label SKIP.
	410C	MOV A,M	7E		Move the data from memory to accumulator.
SKIP	410D	DCR B	05		Decrement B reg. content
	411E	JNZ LOOP	C2	07, 41	If ZF is reset go to LOOP.
	4111	STA 5000H	32	00, 50	Store the largest number in 5000H
	4114	HLT	76		Stop the process

RESULT:

Thus the Assembly Language Program to find the largest of the given data had been written and executed & the result was stored in the desired memory location.

Experiment No: **Date:**

SMALLEST NUMBER IN GIVEN ARRAY

AIM:

To write an Assembly Language Program to find the smallest number in the given Array.

APPARATUS REQUIRED:

((i) 8085 Microprocessor kit with keyboard.
(ii) Power cable.

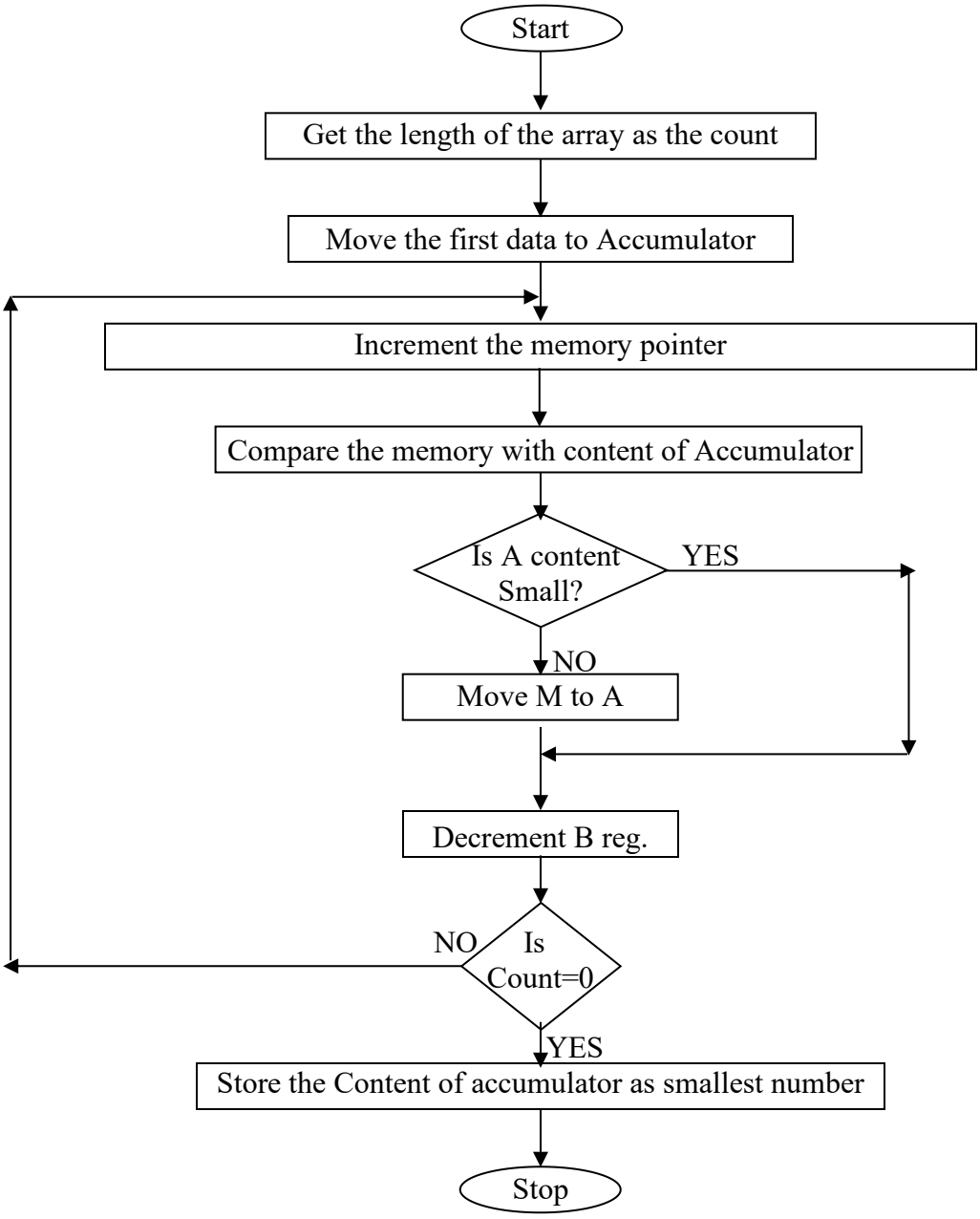
ALGORITHM:

Step 1: Initialize the HL register pair with the address 4700H.
Step 2: Get the first Data in accumulator.
Step 3: Get the count (i.e., Length of the array) in B register.
Step 4: Compare the next data from the memory address pointed by HL pair to the accumulator.
Step 5: Check the Carry Flag. If the CF is set, go to step 7 otherwise go to next step.
Step 6: Move the data from the memory address pointed by HL pair to the accumulator.
Step 7: Decrement the count (content of B register)
Step 8: If the content of B register is not equal to zero, go to step 4, otherwise go to next step.
Step 9: Store the content of accumulator in the address 5000H.
Step 10: Stop the Process.

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	4700H	05	Total number of Data
	4701H	0E	First input data
	4702H	0C	Second input data
	4703H	0F	Third input data
	4704H	03	Fourth input data
	4705H	09	Fifth input data
OUTPUT	5000H	03	Smallest Data

FLOWCHART:


Start
Get the length of the array as the count
Move the first data to Accumulator
Increment the memory pointer
Compare the memory with content of Accumulator
Is A content Small?
YES
NO
Move M to A
Decrement B reg.
NO
Is Count=0
YES
Store the Content of accumulator as smallest number
Stop

PROGRAM:

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	4100	LXI H, 4700H	21	00, 47	Initialize the memory pointer with the address 4700H
	4103	MOV B,M	46		Get the length of the array in B reg.
	4104	INX H	23		Increment the HL reg. pair to point the next memory location.
	4105	MOV A,M	7E		Get the first data in Accumulator.
	4106	DCR B	05		Decrement B reg. content by one
LOOP	4107	INX H	23		Increment the HL reg. pair to point the next memory location.
	4108	CMP M	BE		Compare the data in accumulator with data in memory.
	410B	JC SKIP	DA	0D, 41	If CF is set the program control goes to SKIP.
	410C	MOV A,M	7E		Move the data from memory to accumulator.
SKIP	410D	DCR B	05		Decrement B reg. content.
	411E	JNZ LOOP	C2	07, 41	If ZF is reset go to LOOP.
	4111	STA 5000H	32	00, 50	Store the smallest number in 5000H
	4114	HLT	76		Stop the process

RESULT:

Thus the Assembly Language Program to find the smallest of the given data had been written and executed & the result was stored in the desired memory location.

Experiment No: **Date:**

8085-ASCENDING ORDER

AIM:

To write an Assembly Language Program to arrange the given data in Ascending Order.

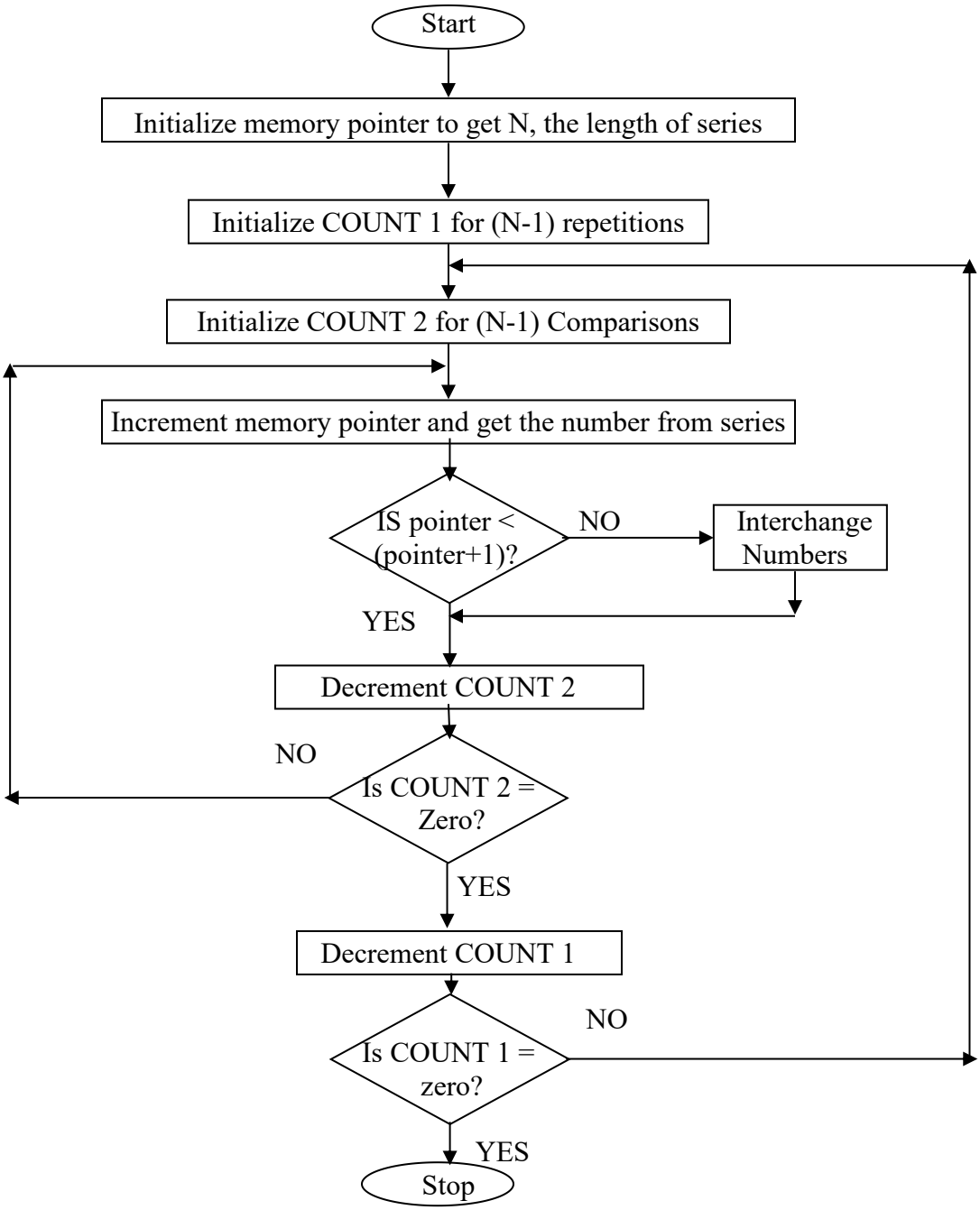
APPARATUS REQUIRED:

(i) 8085 Microprocessor kit with keyboard.
(ii) Power cable.

ALGORITHM:

Step 1: Initialize the HL register pair with the address 4700H.
Step 2: Get the count 1, number of repetitions (length of array - 1) in B register.
Step 3: Get the count 2, number of comparisons (length of array -1) in C register.
Step 4: Save the count 2 in E register.
Step 5: Move the content of E to C register.
Step 6: Initialize the HL register pair with the address 4701H.
Step 7: Move the data from memory location pointed by HL pair to the Accumulator.
Step 8: Compare the data from next memory location with the content of Accumulator.
Step 9: Check CF. If carry flag is set (accumulator content is smaller than the other number), go to step 12 otherwise do next step.
Step 10: Check ZF. If zero flag is set (accumulator content is equal to the other number), go to step 12 otherwise do next step.
Step 11: Store the smallest number to the previous memory location & greatest to the current memory location.
Step 12: Decrement the number of comparisons (count 2).
Step 13: If the number of comparison is not equal to zero, go to step 7, otherwise do the next step.
Step 14: Decrement the number of repetitions (count 1).
Step 15: If the number of repetitions is not equal to zero, go to step 5, otherwise do next step.
Step 16: Stop the Process.

FLOWCHART 8085-ASCENDING ORDER



PROGRAM FOR SORTING A SERIES IN ASCENDING ORDER

LABEL	ADDRESS	MNEMONIC	OPCODE	OPERAND	COMMENTS
START	4100	LXI H, 4700H	21	00, 47	Initialize HL reg. pair with 4700H
	4103	MOV B,M	46		
	4104	DCR B	05		B = COUNT 1 for (N-1)repetitions
	4105	MOV C,B	48		C = COUNT 2 for (N-1)Comparisons
	4106	MOV E,C	59		
LOOP	4107	MOV C,E	4B		
	4108	LXI H, 4701H	21	01, 47	
REPEAT	410B	MOV A,M	7E		
	410C	INX H	23		
	410D	CMP M	BE		Compare consecutive numbers
	411E	JC SKIP	DA	19, 41	
	4111	JZ SKIP	CA	19, 41	
	4114	MOV D,M	56		Interchange numbers if not in order
	4115	MOV M,A	77		
	4116	DCX H	2B		
	4117	MOV M,D	72		
	4118	INX H	23		
SKIP	4119	DCR C	0D		Decrement count 2
	411A	JNZ REPEAT	C2	0B, 41	
	411D	DCR B	05		Decrement count 1
	411E	JNZ LOOP	C2	07, 41	
	4121	HLT	76		Stop the Process

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	4700H	05	Total number of Data
	4701H	0E	First input data
	4702H	0C	Second input data
	4703H	0F	Third input data
	4704H	03	Fourth input data
	4705H	09	Fifth input data
OUTPUT	4700H	05	Total number of Data
	4701H	03	First output data
	4702H	09	Second output data
	4703H	0C	Third output data
	4704H	0E	Fourth output data
	4705H	0F	Fifth output data

RESULT:

Thus the Assembly Language Program to arrange the given data in ascending order had been written and executed.

Experiment No: **Date:**

8085-DESCENDING ORDER

AIM:

To write an Assembly Language Program to arrange the given data in Descending Order.

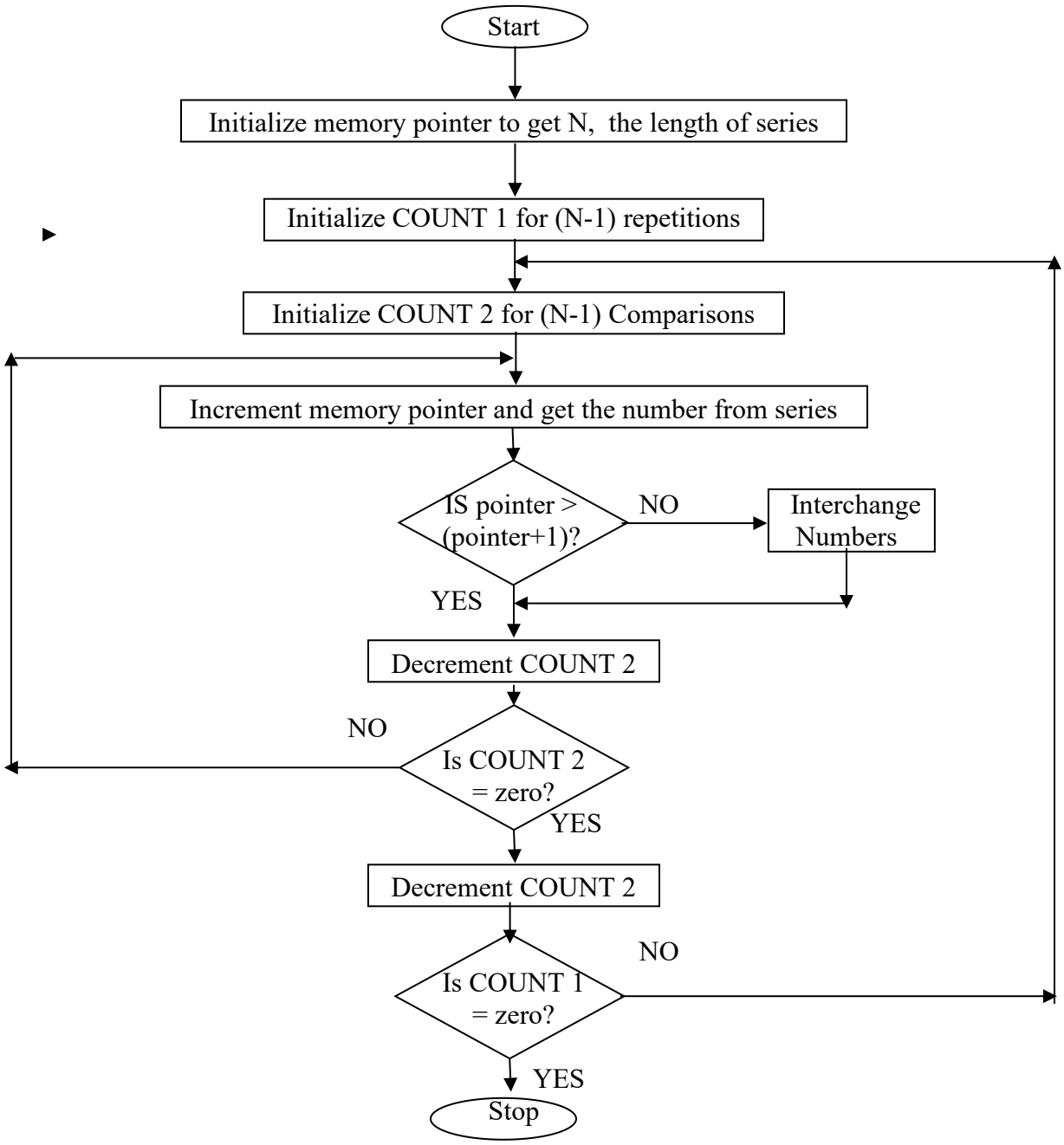
APPARATUS REQUIRED:

(i) 8085 Microprocessor kit with Keyboard.
(ii) Power cable.

ALGORITHM:

Step 1: Initialize the HL register pair with the address 4700H.
Step 2: Get the count 1, number of repetitions (length of array - 1) in B register.
Step 3: Get the count 2, number of comparisons (length of array -1) in C register.
Step 4: Save the count 2 in E register.
Step 5: Move the content of E to C register.
Step 6: Initialize the HL register pair with the address 4701H.
Step 7: Move the data from memory location pointed by HL pair to the Accumulator.
Step 8: Compare the data from next memory location with the content of Accumulator.
Step 9: Check CF. If carry flag is reset (accumulator content is larger than the other number), go to step 12 otherwise do next step.
Step 10: Check ZF. If zero flag is set (accumulator content is equal to the other number), go to step 12 otherwise do next step.
Step 11: Store the largest number to the previous memory location & smallest to the current memory location.
Step 12: Decrement the number of comparisons (count 2).
Step 13: If the number of comparison is not equal to zero, go to step 7, otherwise do the next step.
Step 14: Decrement the number of repetitions (count 1).
Step 15: If the number of repetitions is not equal to zero, go to step 5, otherwise do next step.
Step 16: Stop the Process.

FLOWCHART 8085-DESCENDING ORDER



PROGRAM FOR SORTING A SERIES IN DESCENDING ORDER

LABEL	ADDRESS	MNEMONIC	OPCODE	OPERAND	COMMENTS
START	4100	LXI H, 4700H	21	00, 47	
	4103	MOV B,M	46		
	4104	DCR B	05		B = COUNT 1 for (N-1)repetitions
	4105	MOV C,B	48		C = COUNT 2 for (N-1)Comparisons
	4106	MOV E,C	59		
LOOP	4107	MOV C,E	4B		
	4108	LXI H, 4701H	21	01, 47	
REPEAT	410B	MOV A,M	7E		
	410C	INX H	23		
	410D	CMP M	BE		Compare consecutive numbers
	411E	JNC SKIP	D2	19, 41	
	4111	JZ SKIP	CA	19, 41	
	4114	MOV D,M	56		Interchange numbers if not in order
	4115	MOV M,A	77		
	4116	DCX H	2B		
	4117	MOV M,D	72		
	4118	INX H	23		
SKIP	4119	DCR C	0D		Decrement count 2
	411A	JNZ REPEAT	C2	0B, 41	
	411D	DCR B	05		Decrement count 1
	411E	JNZ LOOP	C2	07, 41	
	4121	HLT	76		Stop the Process

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	4700H	05	Total number of Data
	4701H	0E	First input data
	4702H	0C	Second input data
	4703H	0F	Third input data
	4704H	03	Fourth input data
	4705H	09	Fifth input data
OUTPUT	4700H	05	Total number of Data
	4701H	0F	First output data
	4702H	0E	Second output data
	4703H	0C	Third output data
	4704H	09	Fourth output data
	4705H	03	Fifth output data

RESULT:

Thus the Assembly Language Program to arrange the given data in ascending order had been written and executed.

Experiment No: **Date:**

BLOCK OF DATA TRANSFER

AIM:

To write an Assembly Language Program to transfer the block of data starting from memory location 4700H to the memory location 4800H. The total number of data (length of block) is stored at 46FFH.

APPARATUS REQUIRED:

(i) 8085 Microprocessor kit with keyboard
(ii) Power cord.

ALGORITHM:

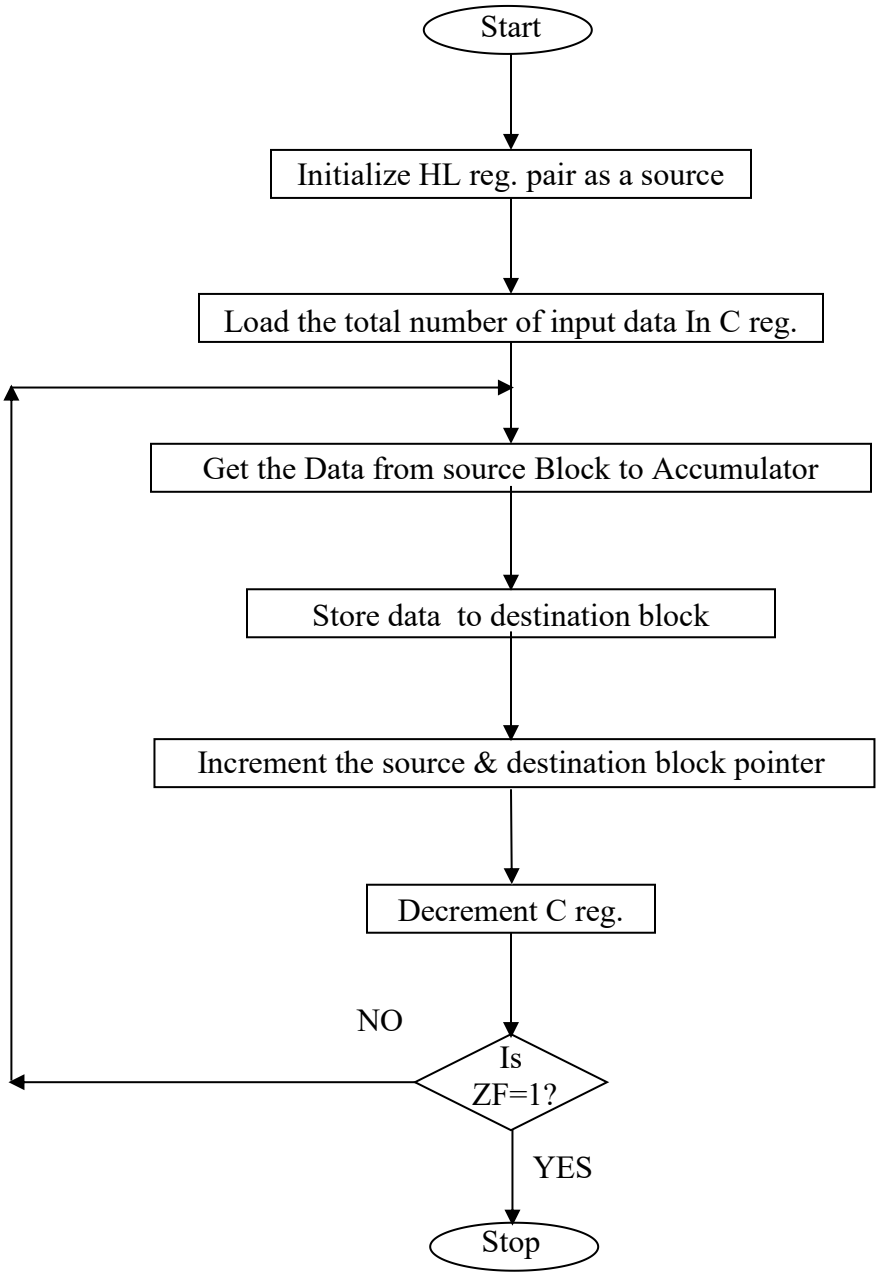
Step 1: Initialize the HL register pair with the address 46FFH.
Step 2: Move the total number of data to the C register.
Step 3: Initialize the DE register pair with the address 4800H.
Step 4: Get byte from source memory block.
Step 5: Store the first data to the destination address.
Step 6: Increment the source block pointer.
Step 7: Increment the destination block pointer.
Step 8: Decrement the C register.
Step 9: If ZF=0, go to step 4.otherwise go to next step.
Step 10: Stop the Process.

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	46FFH	05	Total number of Data
	4700H	45	First input data
	4701H	66	Second input data
	4702H	72	Third input data
	4703H	0D	Fourth input data
	4704H	0A	Fifth input data
OUTPUT	4800H	45	First output data
	4801H	66	Second output data
	4802H	72	Third output data
	4803H	0D	Fourth output data

	4804H	0A	Fifth output data

FLOWCHART OF BLOCK DATA TRANSFER



PROGRAM OF BLOCK DATA TRANSFER:

LABEL	ADDRESS	MNEMONIC	OPCODE	OPERAND	COMMENTS
START	4100H	LXI H,46FFH	21	FF,46	Initialize the source pointer with the address 4200H.
	4103H	MVI C,M	4E		Get count of total number of data bytes in C reg.
	4104H	INX H	23		Point to first source block byte.
	4105H	LXI D,4800H	11	00,48	Initialize the destination pointer with the address 4200H.
REPEAT	4108H	MOV A,M	7E		Get the data byte from source block.
	4109H	STAX D	12		Store the data byte to the destination block.
	410AH	INX H	23		Increment source block pointer.
	410BH	INX D	13		Increment destination block pointer.
	410CH	DCR C	0D		Decrement C register.
	410DH	JNZ REPEAT	C2	08,41	If ZF=0, program control jump to label REPEAT.
	4110H	HLT	76		Stop the Process

RESULT:

Thus the assembly language program to transfer the block of data from memory location 4700H to 4800H had been written and verified.

Experiment No: **Date:**

DECIMAL TO HEXADECIMAL CONVERSION

AIM:

To write an Assembly Language Program to convert a Decimal number to a Hexadecimal number. The Decimal data is stored in 4200H and Hexadecimal data is store in 4250H.

APPARATUS REQUIRED:

(i) 8085 Microprocessor kit with keyboard.
(ii) Power cable.

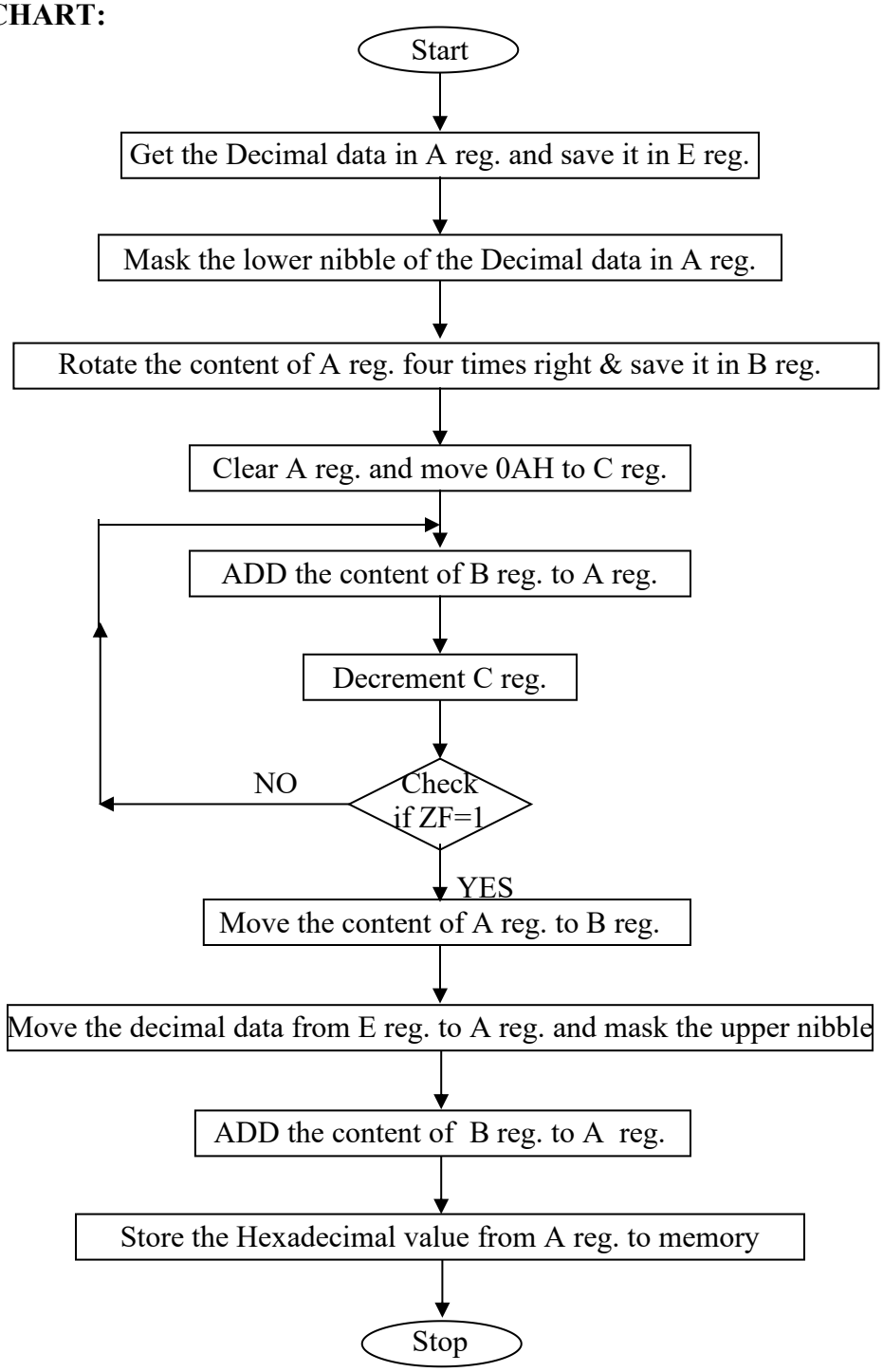
ALGORITHM:

Step 1: Get the Decimal data in A reg. and save in E reg.
Step 2: Mask the lower nibble of the decimal data in A reg.
Step 3: Rotate the upper nibble to the lower position.
Step 4: Clear the Accumulator.
Step 5: Move the 0AH to C reg.
Step 6: Add B reg. content to the A reg. Content.
Step 7: Decrement the C reg. If ZF=0 go to step 6. I f ZF=1 go to next step.
Step 8: Save the product in B reg.
Step 9: Get the decimal in A reg. from E reg. and mask the upper nibble(units).
Step 10: Add the units (A reg.) to product (B reg.).
Step 11: Stop the Process.

OBSERVATION:

	ADDRESS	**DATA**	**COMMENTS**
INPUT	4200	32	Decimal Number
OUTPUT	4250	20	Hexadecimal Number

FLOWCHART:


Start
Get the Decimal data in A reg. and save it in E reg.
Mask the lower nibble of the Decimal data in A reg.
Rotate the content of A reg. four times right & save it in B reg.
Clear A reg. and move 0AH to C reg.
ADD the content of B reg. to A reg.
Decrement C reg.
Check if ZF=1
NO
YES
Move the content of A reg. to B reg.
Move the decimal data from E reg. to A reg. and mask the upper nibble
ADD the content of B reg. to A reg.
Store the Hexadecimal value from A reg. to memory
Stop

PROGRAM:

LABEL	ADDRESS	MNEMONIC	OPCODE	OPERAND	COMMENTS
START	4100	LDA 4200H	3A	00,42	Get the data in A reg.
	4103	MOV E,A	5F		Save in E reg.
	4104	ANI F0H	E6	F0	Mask the lower nibble
	4106	RLC	07		Rotate the upper nibble to lower Nibble position save in B reg.
	4107	RLC	07		
	4108	RLC	07		
	4109	RLC	07		
	410A	MOV B,A	47		
	410B	XRA A	AF		Clear accumulator
	410C	MVI C,0AH	0E	0A	Get the product of units digit multiplied by 0AH in A reg.
REPEAT	410E	ADD B	80		
	410F	DCR C	0D		
	4110	JNZ REPEAT	C2	0E,41	
	4113	MOV B,A	47		Save the product in B reg.
	4114	MOVA,E	7B		Get the Decimal data in A reg.
	4115	ANI 0FH	E6	0F	Mask the upper nibble
	4117	ADD B	80		Get the sum of units digit and product in B reg.
	4118	STA 4250H	32	50,42	Store the Hexadecimal value in memory
	411B	HLT	76		Stop the Process

RESULT:

Thus the Assembly Language Program for Decimal to Hexadecimal conversion had been written and executed & the results were stored in desired memory location.

Experiment No: **Date:**

8086-ADDITION OF TWO 16-BIT NUMBERS

AIM:

To write an assembly language program to add two 16 bit numbers and store the sum in the memory location 1200H and carry in 1202H.

APPARATUS REQUIRED:

(i) 8086 Microprocessor kit with keyboard.
(ii) Power Cable.

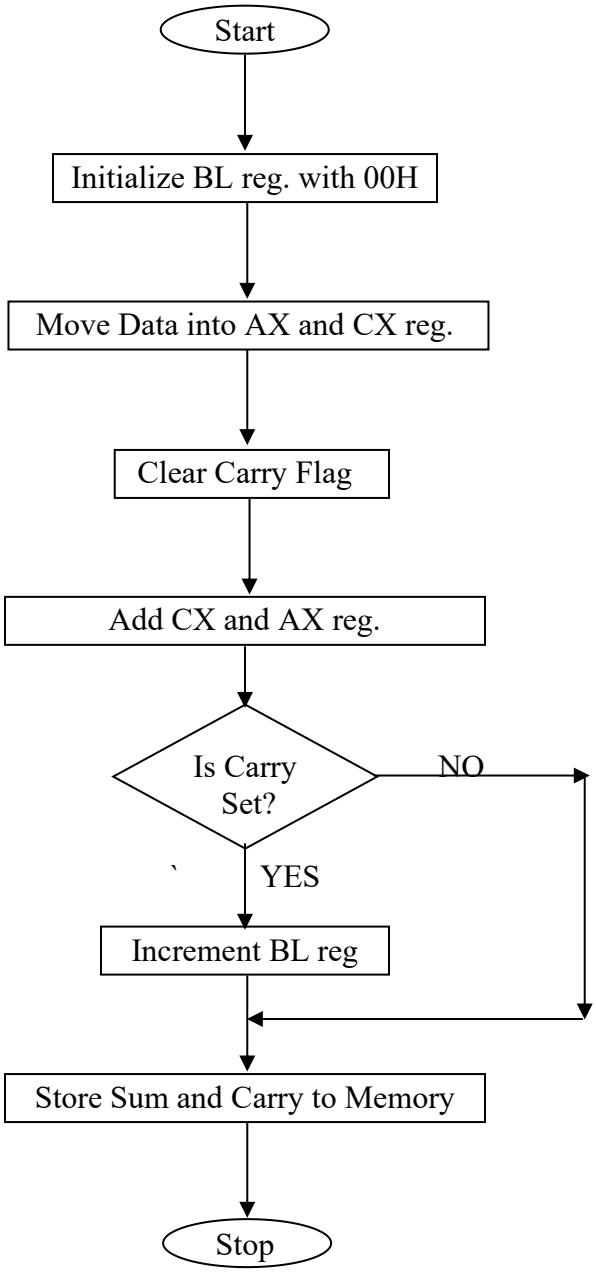
ALGORITHM:

Step 1: Initialize the BL reg. with 00H to account for carry.
Step 2: Move one of the data to AX reg.
Step 3: Move next data to CX reg.
Step 4: Clear Carry Flag.
Step 5: ADD the CX reg. content with the AX reg. Jump to step 7 if no carry occurs. Otherwise go to next step.
Step 6: Increment BL reg.
Step 7: Move the sum to 1200H.
Step 8: Move the carry to 1202H.
Step 9: Stop the Process

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	[AX]	FFFFH	First 16 bit data
	[CX]	001FH	Second 16 bit data
OUTPUT	1200H	1E	Low order byte of sum
	1201H	00	High order byte of sum
	1202H	01	carry

FLOWCHART:


Start
Initialize BL reg. with 00H
Move Data into AX and CX reg.
Clear Carry Flag
Add CX and AX reg.
Is Carry Set?
NO
YES
Increment BL reg
Store Sum and Carry to Memory
Stop

PROGRAM:

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	1000H	MOV BL,00H	C6,C3	00	Move 00H to BL reg. to account for carry.
	1003H	MOV AX,FFFFH	C7,C0	FF,FF	Move the data FFFFH to AX reg.
	1007H	MOV CX,001FH	C7,C1	1F,00	Move the data 001FH to CX reg.
	100BH	CLC	F8		Clear Carry Flag.
	100CH	ADD AX, CX	01,C8		Add the content of CX to AX.
	100EH	JNC LOOP	73	02	If no carry occurs, jump to LOOP.
	1010H	INC BL	FE,C3		Increment BL reg.
LOOP	1012H	MOV[1200H],AX	89,06	00,12	Move the AX content (sum)to 1200H & 1201H.
	1016H	MOV[1202H],BL	88,1E	02,12	Move the BL content to 1202H.
	101AH	HLT	F4		Stop the Process.

RESULT:

Thus the Assembly Language Program for Addition of two 16-bit numbers had been written and executed & the results were stored in desired memory locations.

Experiment No: **Date:**

8086-SUBTRACTION OF TWO 16-BIT NUMBERS

AIM:

To write an assembly language program to Subtract two 16 bit numbers and store the difference in the memory location 1200H and borrow in 1202H.

APPARATUS REQUIRED:

(i) 8086 Microprocessor kit with keyboard.
(ii) Power Cable.

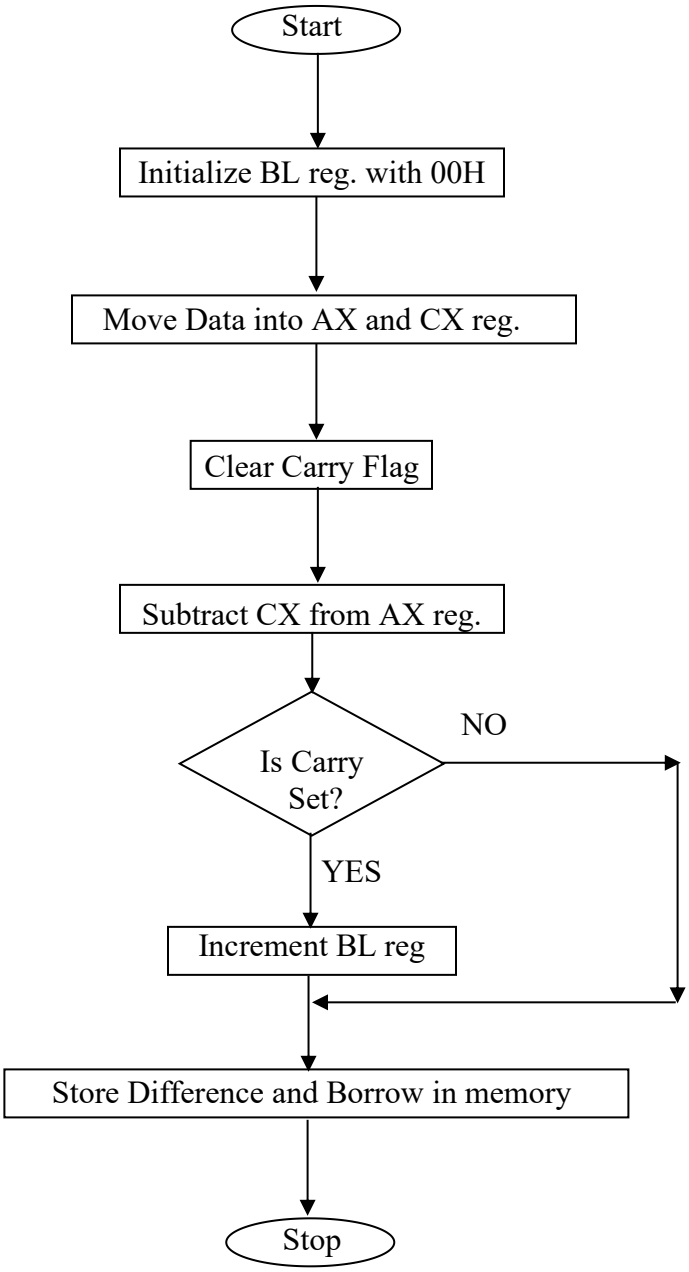
ALGORITHM:

Step 1: Initialize the BL reg with 00H to account for borrow.
Step 2: Move Minuend to AX reg.
Step 3: Move Subtrahend to CX reg.
Step 4: Clear Carry Flag.
Step 5: Subtract the CX reg. content from the AX reg. Jump to step 7 if no carry occurs. Otherwise go to next step.
Step 6: Increment BL reg.
Step 7: Move the difference to 1200H.
Step 8: Move the borrow to 1202H.
Step 9: Stop the Process

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	[AX]	FFFFH	First 16 bit data
	[CX]	001FH	Second 16 bit data
OUTPUT	1200H	E0	Low order byte of difference
	1201H	FF	High order byte of difference
	1202H	00	Borrow

FLOWCHART:


Start
Initialize BL reg. with 00H
Move Data into AX and CX reg.
Clear Carry Flag
Subtract CX from AX reg.
Is Carry Set?
NO
YES
Increment BL reg
Store Difference and Borrow in memory
Stop

PROGRAM:

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	1000H	**MOV BL,00H**	C6,C3	00	**Move 00H to BL reg. to account for carry.**
	1003H	**MOV AX,FFFFH**	C7,C0	FF,FF	**Move the data FFFFH to AX reg.**
	1007H	**MOV CX,OO1FH**	C7,C1	1F,00	**Move the data 001FH to CX reg.**
	100BH	**CLC**	F8		**Clear Carry Flag.**
	100CH	**SUB AX, CX**	29,C8		**Subtract the content of CX from AX.**
	100EH	**JNC LOOP**	73	02	**If no carry occurs, jump to LOOP.**
	1010H	**INC BL**	FE,C3		**Increment BL reg.**
LOOP	1012H	**MOV[1200H],AX**	89,06	00,12	**Move the AX content (difference) to 1200H.**
	1016H	**MOV[1202H],BL**	88,1E	02,12	**Move the BL content to 1202H.**
	101AH	**HLT**	F4		**Stop the Process.**

RESULT:

Thus the Assembly Language Program for Subtraction of two 16-bit numbers had been written and executed & the results were stored in desired memory locations.

Experiment No: **Date:**

8086-MULTIPLICATION OF TWO 16-BIT NUMBERS

AIM:

To write an assembly language program for Multiplication of two 16 bit numbers using 8086 instruction set.

APPARATUS REQUIRED:

(i) 8086 Microprocessor kit with keyboard.
(ii) Power Cable.

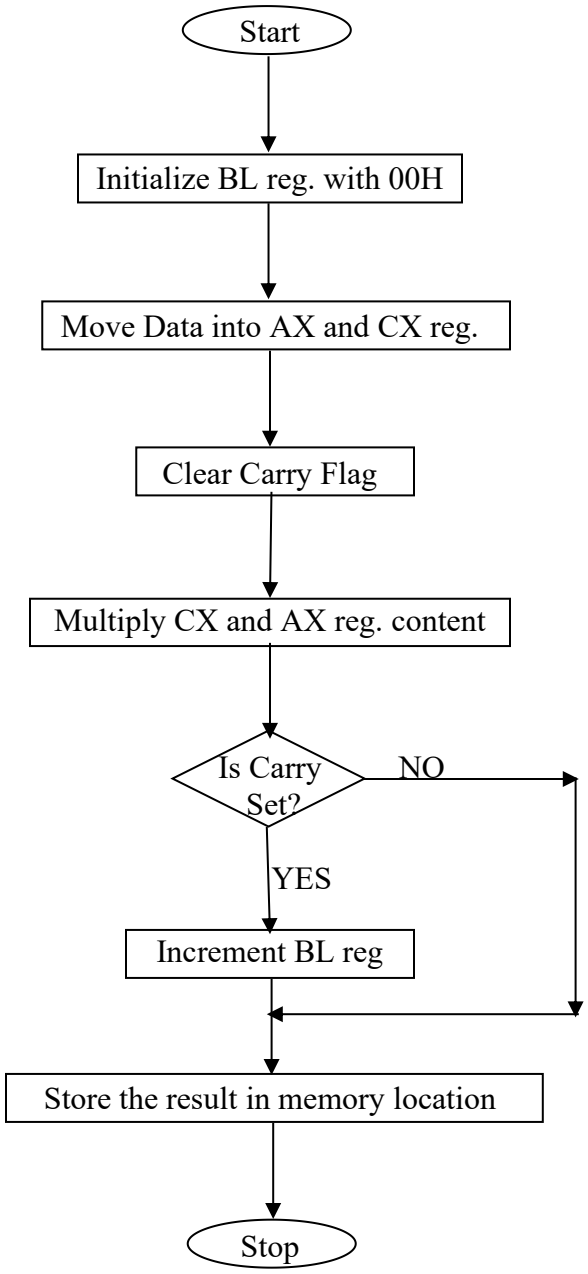
ALGORITHM:

Step 1: Initialize the BL reg. with 00H to account for carry.
Step 2: Move Multiplicand to AX reg.
Step 3: Move Multiplier to CX reg.
Step 4: Clear Carry Flag.
Step 5: Multiply the CX reg. with the AX reg. Jump to step 7 if no carry occurs. Otherwise go to next step.
Step 6: Increment BL reg.
Step 7: Move the content of AX & BL to the specified memory locations.
Step 9: Stop the Process

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	[AX]	11A2H	First 16 bit data
	[CX]	11B3H	Second 16 bit data
OUTPUT	1200H	46	Low order byte of Product
	1201H	16	Low order byte of Product
	1202H	38	High order byte of Product
	1203H	01	High order byte of Product
	1204H	00	Carry

FLOWCHART:


Start
Initialize BL reg. with 00H
Move Data into AX and CX reg.
Clear Carry Flag
Multiply CX and AX reg. content
Is Carry Set?
NO
YES
Increment BL reg
Store the result in memory location
Stop

PROGRAM:

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	1000H	MOV BL,00H	C6,C3	00	Move 00H to BL reg. to account for carry.
	1003H	MOV AX,11A2H	C7,C0	A2,11	Move the data 11A2H to AX reg.
	1007H	MOV CX,11B3H	C7,C1	B3,11	Move the data 11B3H to CX reg.
	100BH	MUL CX	F7,E1		Multiply the content of CX to AX reg.
	100DH	JNC LOOP	73	02	If no carry occurs, jump to LOOP.
	100FH	INC BL	FE,C3		Increment BL reg.
LOOP	1011H	MOV[1200H],AX	89,06	00,12	Move the AX content to 1200H & 1201H.
	1015H	MOV[1202],DX	89,16	02,12	Move the DX content to 1202H & 1203H.
	1019H	MOV[1204H],BL	88,1E	04,12	Move the BL content to 1204H.
	101DH	HLT	F4		Stop the Process.

RESULT:

Thus the Assembly Language Program for Multiplication of two 16-bit numbers had been written and executed & the results were stored in desired memory locations.

Experiment No: **Date:**

8086-DIVISION OF 32 BIT NUMBER BY 16-BIT NUMBER

AIM:

To write an assembly language program for Division of 32 bit number by 16 bit number using 8086 microprocessor kit.

APPARATUS REQUIRED:

(i) 8086 Microprocessor kit with keyboard.
(ii) Power Cable.

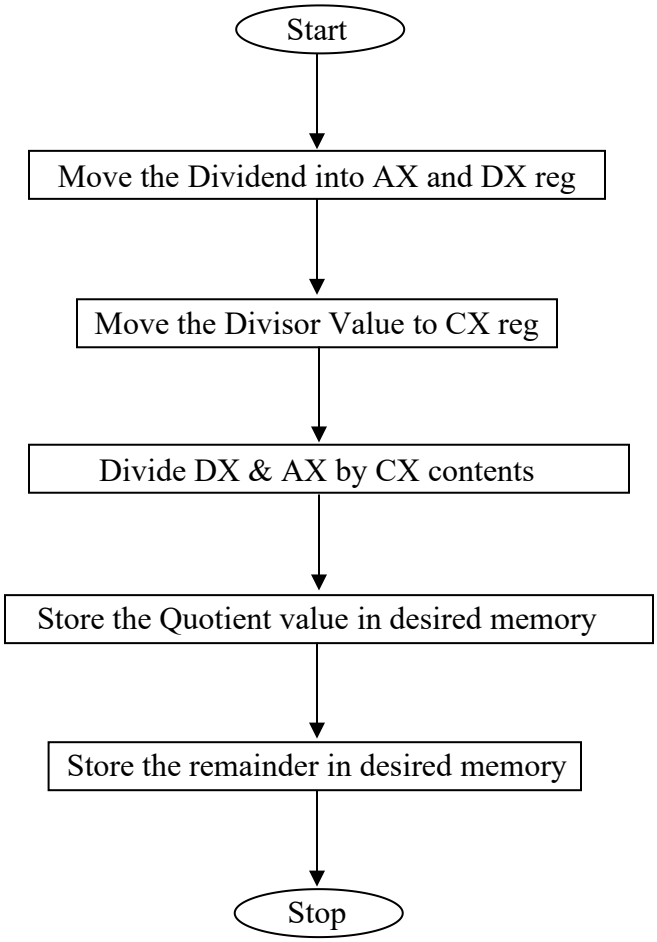
ALGORITHM:

Step 1: Move the lower order byte of dividend to AX reg. and higher order byte of dividend to DX reg.
Step 2: Move Divisor to CX reg.
Step 3: Divide the content of AX&DX by CX reg.
Step 4: Move the content of AX & DX in the specified memory locations.
Step 5: Stop the Process.

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	[AX]	FFFFH	Low order bit of 32-bit data.
	[DX]	0012H	High order bit of 32-bit data
	[CX]	01A1H	16 bit Divisor.
OUTPUT	1200	AA	Low order byte of Quotient
	1201	0B	High order byte of Quotient
	1202	15	Low order byte of Remainder
	1203	00	High order byte of Remainder

FLOWCHART:


Start
Move the Dividend into AX and DX reg
Move the Divisor Value to CX reg
Divide DX & AX by CX contents
Store the Quotient value in desired memory
Store the remainder in desired memory
Stop

PROGRAM:

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	1000H	MOV AX,FFFFH	C7,C0	FF,FF	Move the data FFFFH to AX reg
	1004H	MOV DX, 0012H	C7,C2	12,00	Move the data 0012H to DX reg
	1008H	MOV CX,01A1H	C7,C1	A1,01	Move the Divisor 01A1 to CX reg
	100CH	DIV CX	F7,F1		Divide the content of AX&DX BY CX
	100EH	MOV[1200H],AX	89,06	00,12	Move the AX content to 1200H.
	1012H	MOV[1202H],DX	89,16	02,12	Move the DX content to 1202H.
	1016H	HLT	F4		Stop the Process

RESULT:

Thus the Assembly Language Program for the Division of 32 –bit number by 16-bit number had been written and executed & the results were stored in desired memory location.

Experiment No: **Date:**

8051-ADDITION OF TWO 8-BIT NUMBERS

AIM:

To perform the addition of two 8 bit numbers using immediate addressing and store the result in memory.

APPARATUS REQUIRED:

(i) 8051 Microcontroller kit with keyboard.
(ii) Power Cable.

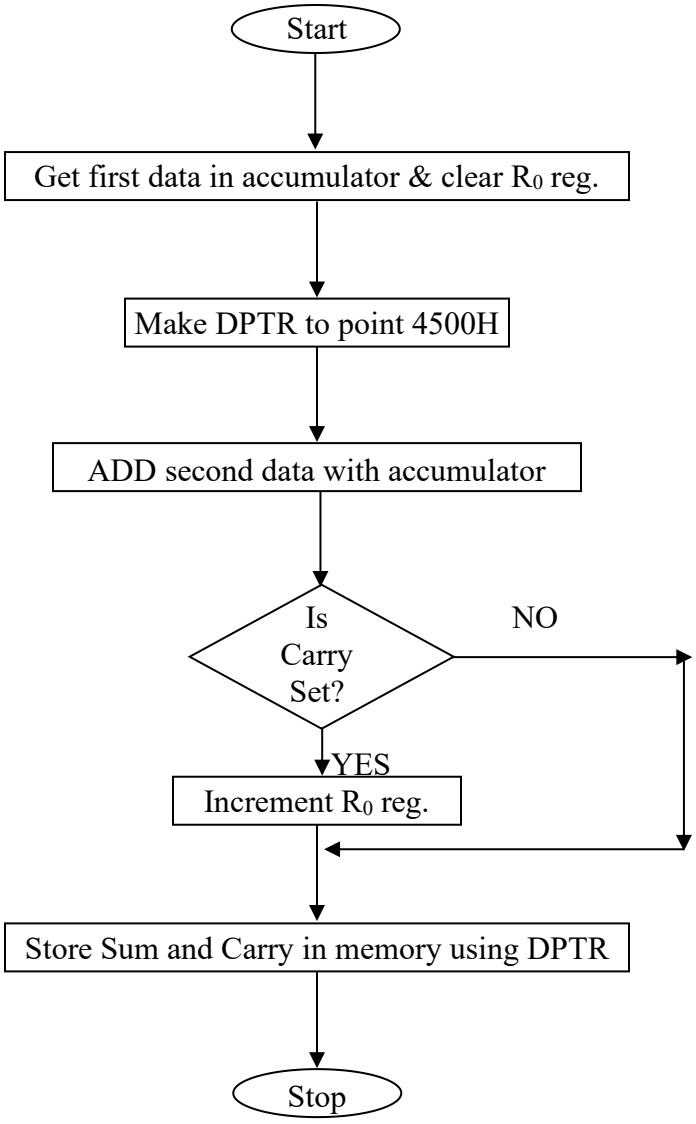
ALGORITHM:

Step 1: Clear register R_0.
Step 2: Get the first data in accumulator.
Step 3: Add the second data with the first data.
Step 4: Make the data pointer to point the address 4500H.
Step 5: Check the carry flag. If no carry occurs, go to step 7, otherwise do next step.
Step 6: Increment the register R_0.
Step 7: Store the sum (content of accumulator) in the address pointed by the data pointer.
Step 8: Make the data pointer to point the next address by incrementing the DPTR.
Step 9: Store the carry in that address.
Step 10: Stop the process.

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	4103H	31	First data
	4105H	49	Second data
OUTPUT	4500H	7A	Sum
	4501H	00	Carry

FLOWCHART:


Start
Get first data in accumulator & clear R0 reg.
Make DPTR to point 4500H
ADD second data with accumulator
Is Carry Set?
NO
YES
Increment R0 reg.
Store Sum and Carry in memory using DPTR
Stop

PROGRAM:

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	4100H	**MOV R_0, #00H**	78	00	Move the data 00H to reg R_0.
	4102H	**MOV A, #data1**	74	20	Move the first data in accumulator
	4104H	**ADD A, #data2**	24	10	Add the second data with the accumulator.
	4106H	**MOV DPTR, #4500H**	90	45,00	Move the address 4500H to DPTR.
	4109H	**JNC LOOP**	50	01	If no carry occurs go to LOOP
	410BH	**INC** R_0	08		Increment reg. R_0
LOOP	410CH	**MOV X@DPTR,A**	F0		Move the data from accumulator to the address pointed by DPTR
	410DH	**INC DPTR**	A3		Increment the DPTR
	410EH	**MOV A,** R_0	E8		Move the content from reg. Ro to the accumulator.
	410FH	**MOV X@ DPTR, A**	F0		Move the data from accumulator to the address pointed by DPTR
	4110H	**NOP**	00		No operation.

RESULT:

Thus the addition of two 8-bit data using immediate addressing was executed.

Experiment No: **Date:**

8051-SUBTRACTION OF TWO 8-BIT NUMBERS

AIM:

To perform the subtraction of two 8 bit numbers using immediate addressing and store the result in memory.

APPARATUS REQUIRED:

(i) 8051 Microcontroller kit with keyboard.
(ii) Power Cable.

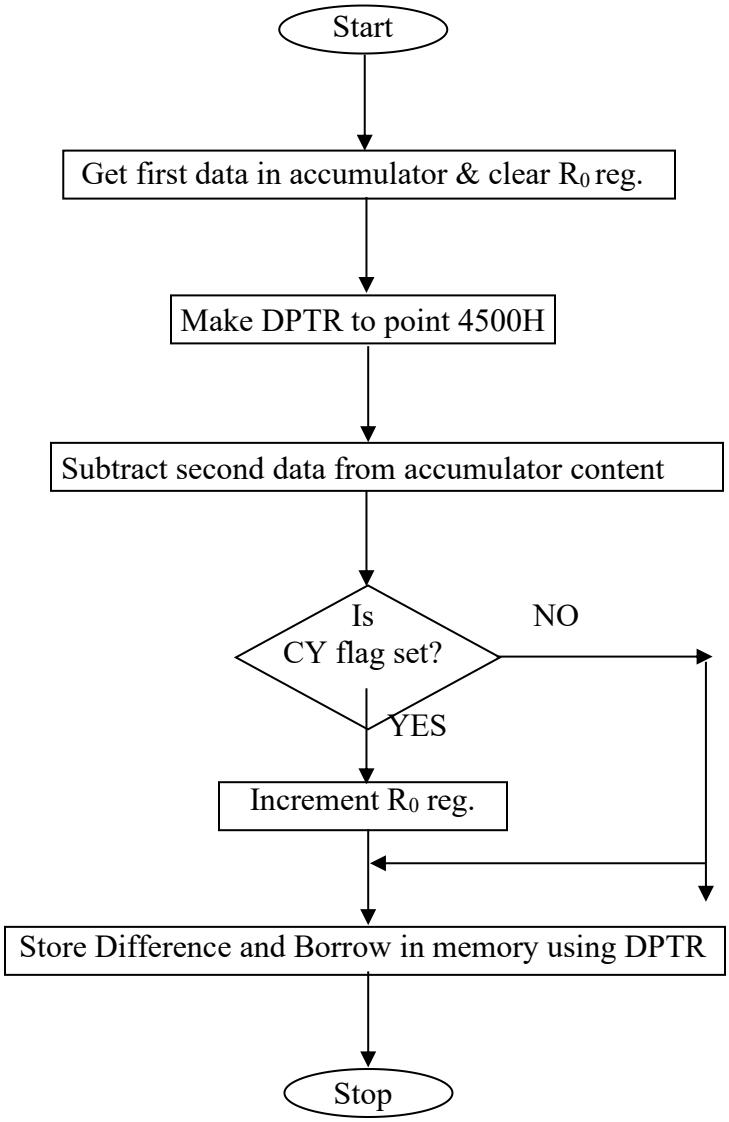
ALGORITHM:

Step 1: Clear register R_0
Step 2: Get the data in accumulator.
Step 3: Subtract the second data from the first data.
Step 4: Make the data pointer to point the address 4500H.
Step 5: Check the carry flag. If no carry occurs, go to step 7, otherwise do next step.
Step 6: Increment the register R_0.
Step 7: Store the difference (content of accumulator) in the address pointed by the data pointer.
Step 8: Make the data pointer to point the next address by incrementing the DPTR.
Step 9: Store the borrow in that address.
Step 10: Stop the process.

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	4303H	0C	First data
	4305H	A5	Second data
OUTPUT	4500H	99	Difference
	4501H	01	Borrow

FLOWCHART:


Start
Get first data in accumulator & clear R_0 reg.
Make DPTR to point 4500H
Subtract second data from accumulator content
Is
CY flag set?
NO
YES
Increment R_0 reg.
Store Difference and Borrow in memory using DPTR
Stop

PROGRAM:

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	4300H	**MOV R_0,#00H**	78	00	Move the data 00H to reg R_0
	4302H	**MOV A,#data1**	74	20	Move the first data in accumulator
	4304H	**SUB A,#data2**	94	10	Subtract the second data with the accumulator.
	4306H	**MOV DPTR, #4500H**	90	45,00	Move the address 4500H to DPTR.
	4309H	**JNC LOOP**	50	01	If no carry occurs go to LOOP
	430BH	**INC R_0**	08		Increment reg. R_0.
LOOP	430CH	**MOV X@DPTR,A**	F0		Move the data from accumulator to the address pointed by DPTR
	430DH	**INC DPTR**	A3		Increment the DPTR
	430EH	**MOV A, R_0**	E8		Move the content from reg R_0 to the accumulator.
	430FH	**MOV X@ DPTR, A**	F0		Move the data from accumulator to the address pointed by DPTR
	4310H	**NOP**	00		No operation.

RESULT:

Thus the subtraction of two 8-bit data using immediate addressing was executed.

Experiment No: **Date:**

8051-MULTIPICATION OF TWO 8-BIT NUMBERS

AIM:

To write an 8051 assembly language program to multiply two 8-bit numbers and store the result in memory.

APPARATUS REQUIRED:

(i) 8051 Microcontroller kit with keyboard.
(ii) Power Cable.

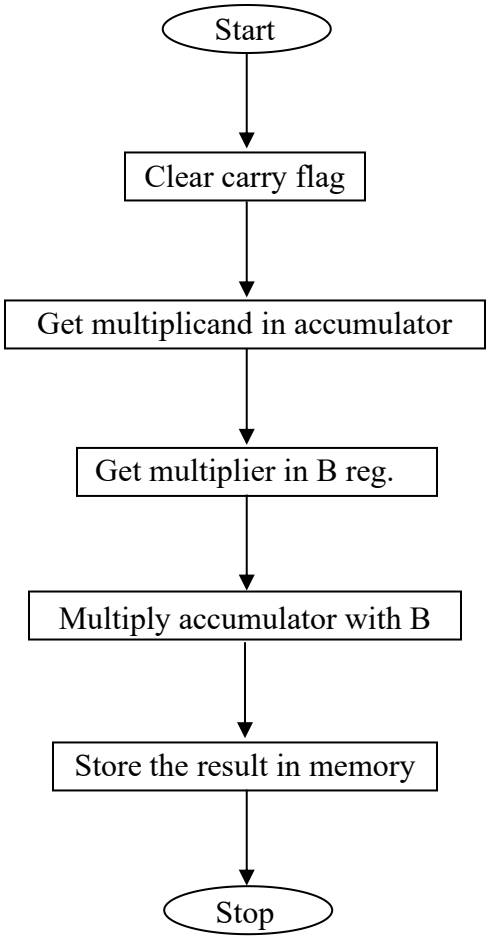
ALGORITHM:

Step 1: Get the first data in accumulator.
Step 2: Get the second data in reg B.
Step 3: Multiply two 8-bit data.
Step 4: Get the data pointer to point the address 4500H.
Step 5: Store the LSB of the result from the accumulator to the address pointed by DPTR.
Step 6: Increment the data pointer.
Step 7: Store the MSB of the result in the address pointed by DPTR.
Step 8: Stop the process.

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	4301H	23	First data
	4304H	0C	Second data
OUTPUT	4500H	A4	LSB
	4501H	01	MSB

FLOWCHART:


Start
Clear carry flag
Get multiplicand in accumulator
Get multiplier in B reg.
Multiply accumulator with B
Store the result in memory
Stop

PROGRAM:

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	4300H	**MOV A,#data1**	74	23	Move the data 1 to accumulator.
	4302H	**MOV F0,#data2**	75, F0	0C	Move the data 2 the reg B.
	4305H	**MUL B**	A4		Multiply A & B.
	4306H	**MOV DPTR, #4500H**	90	45,00	Move the address 4500H to DPTR.
	4309H	**MOV X@DPTR,A**	F0		Move the data from accumulator to the address pointed by DPTR
	430AH	**INC DPTR**	A3		Increment the DPTR.
	430BH	**MOV A,B**	E5,F0		Move the content of B reg to the accumulator.
	430DH	**MOV X@DPTR,A**	F0		Move the data from accumulator to the address pointed by DPTR.
	430EH	**NOP**	00		No operation.

RESULT:

Thus the Assembly Language Program for the multiplication of two 8-bit data had been written and executed & the results were stored in desired memory location.

Experiment No: **Date:**

8051-DIVISION OF TWO 8-BIT NUMBERS

AIM:

To write an 8051 assembly language program to divide two 8-bit numbers and store the result in memory.

APPARATUS REQUIRED:

(i) 8051 Microcontroller kit with keyboard.
(ii) Power Cable.

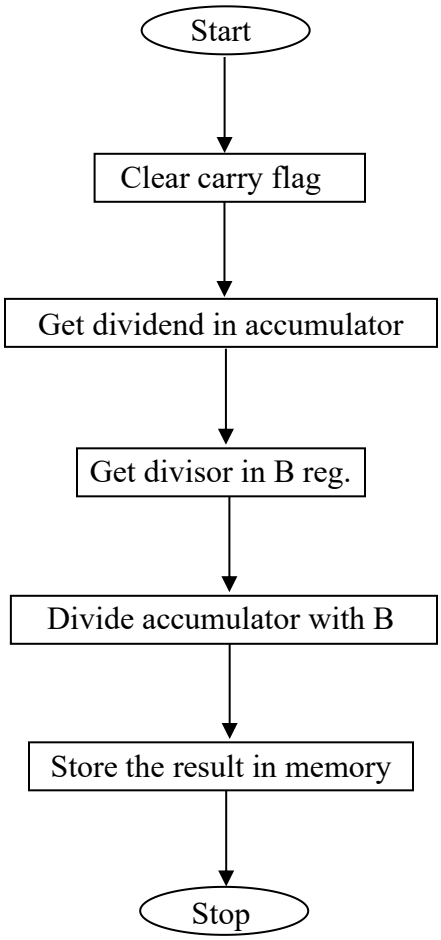
ALGORITHM:

Step 1: Get the first data in accumulator.
Step 2: Get the second data in reg B.
Step 3: Divide two 8-bit data.
Step 4: Get the data pointer to point the address 4500H.
Step 5: Store the Quotient of the result from the accumulator to the address pointed by DPTR.
Step 6: Increment the data pointer.
Step 7: Store the remainder of the result in the address pointed by DPTR.
Step 8: Stop the process.

OBSERVATION:

	ADDRESS	DATA	COMMENTS
INPUT	4301H	56	First data
	4304H	15	Second data
OUTPUT	4500H	04	Quotient
	4501H	06	Remainder

FLOWCHART:


Start
Clear carry flag
Get dividend in accumulator
Get divisor in B reg.
Divide accumulator with B
Store the result in memory
Stop

PROGRAM:

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	4300H	**MOV A,#data1**	74	56	Move the data 1 to accumulator.
	4302H	**MOV F0,#data2**	75	15	Move the data 2 the reg B.
	4305H	**DIV B**	84		Divide A by B.
	4306H	**MOV DPTR, #4500H**	90	45,00	Move the address 4500H to DPTR.
	4309H	**MOV X@DPTR,A**	F0		Move the quotient from accumulator to the address pointed by DPTR
	430AH	**INC DPTR**	A3		Increment the DPTR.
	430BH	**MOV A,B**	E5,E0		Move the content (remainder) of B reg. to the accumulator.
	430DH	**MOV X@DPTR,A**	F0		Move the remainder from accumulator to the address denoted by DPTR.
	430EH	**NOP**	00		No operation.

RESULT:

Thus the Assembly Language Program for the division of two 8-bit data had been written and executed & the results were stored in desired memory location.

Experiment No: **Date:**

TRAFFIC LIGHT CONTROLLER

AIM:

To control the traffic light system using 8085 assembly language program.

APPARATUS REQUIRED:

(i) 8085 Microprocessor kit with keyboard.
(ii) Power Cable.
(iii)Traffic Light Interface with connecting cable.

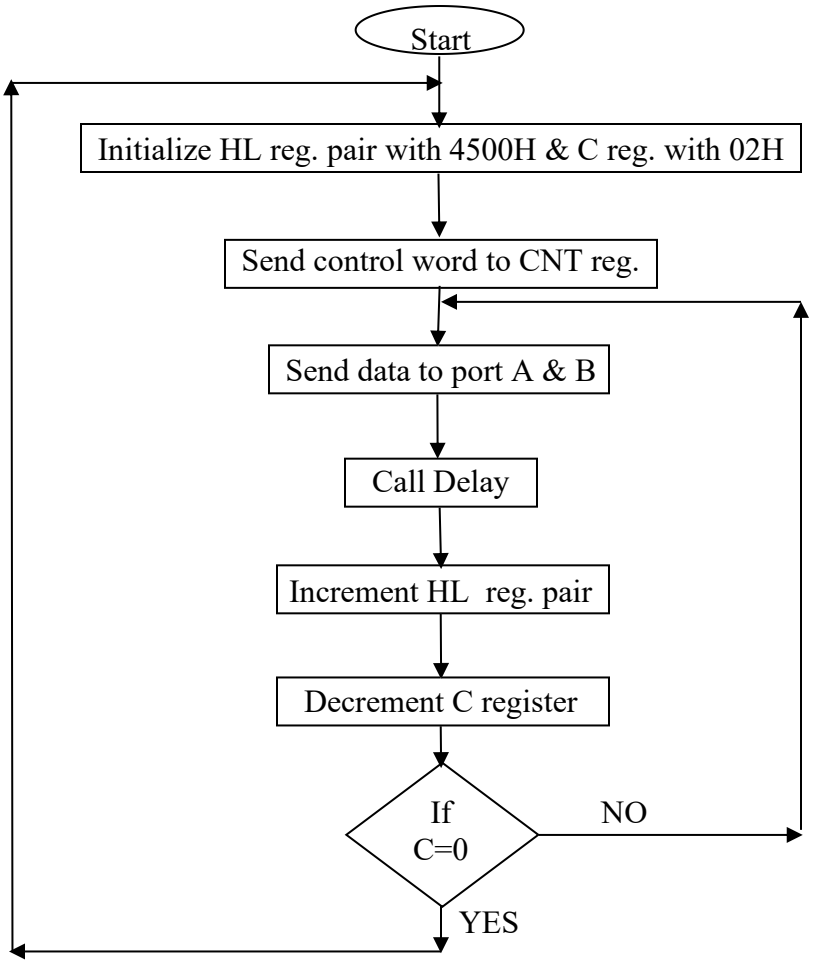
ALGORITHM:

Step 1: Initialize HL reg. pair with 4500H and C reg with 02H.
Step 2: Send control word to CNT register.
Step 3: Send data to port A & B.
Step 4: Call delay and increment HL pair.
Step 5: Decrement C reg.
Step 6: if C=0 go to step 1, otherwise go to step 3.

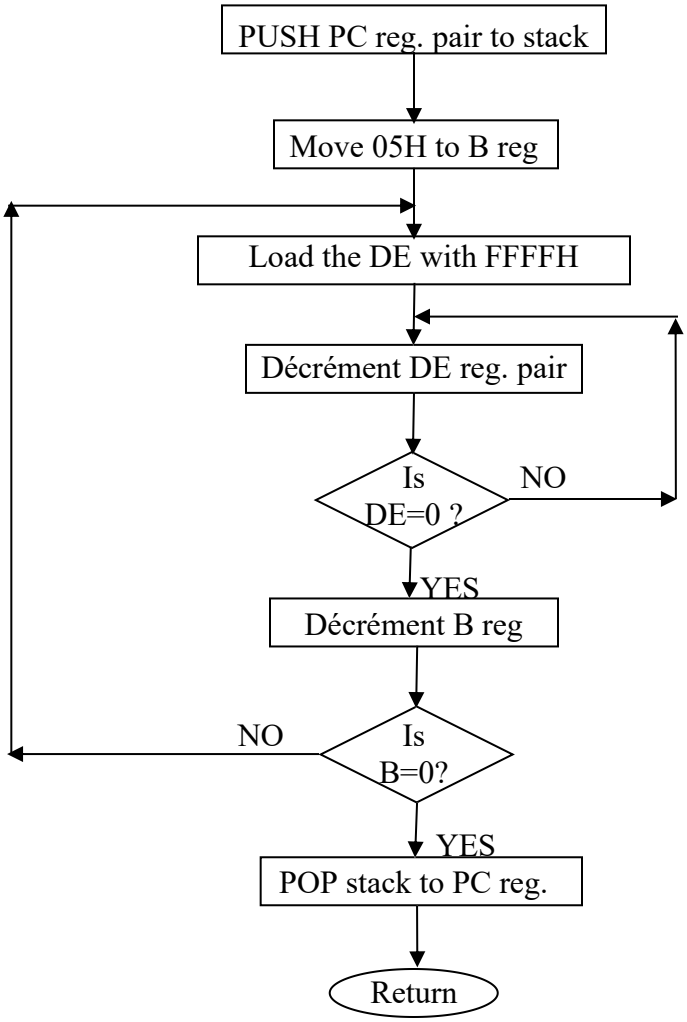
OBSERVATION:

ADDRESS	DATA	COMMENTS
4500H	80	Control Word
4501H	1A	First Step Data
4502H	A1	First Step Data
4503H	81	Second Step Data
4504H	5A	Second Step Data

FLOWCHART:


Start
Initialize HL reg. pair with 4500H & C reg. with 02H
Send control word to CNT reg.
Send data to port A & B
Call Delay
Increment HL reg. pair
Decrement C register
If
C=0
NO
YES

DELAY:


PUSH PC reg. pair to stack
Move 05H to B reg
Load the DE with FFFFH
Décrément DE reg. pair
Is DE=0 ?
NO
YES
Décrément B reg
Is B=0?
NO
YES
POP stack to PC reg.
Return

PROGRAM:

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	4100H	LXI H,4500H	21	00,45	Initialize HL pair with 4500H
`	4103H	MVI C,02H	0E	02	Move data 02H to C reg.
	4105H	MOV A,M	7E		Get data from memory to accumulator.
	4106H	OUT CNT	D3	0F	Send data to control reg.
	4108H	INX H	23		Increment HL pair reg.
LOOP1	4109H	MOV A,M	7E		Get data from memory to accumulator.
	410AH	OUT APRT	D3	0C	Send data to port A.
	410CH	INX H	23		Increment HL pair reg.
	410DH	MOV A,M	7E		Get data from memory to accumulator.
	410EH	OUT BPRT	D3	0D	Send data to port B.
	4110H	CALL DELAY	CD	1B,41	Call delay routine.
	4113H	INX H	23		Increment HL pair reg.
	4114H	DCR C	0D		Decrement HL pair.
	4115H	JNZ LOOP1	C2	09,41	If C reg value is non-zero go to loop1.
	4118H	JMP START	C3	00,41	Go to start.
DELAY	411BH	MVI B.05H	06	05	Move 05H to B reg.
LOOP3	411DH	LXI D,FFFFH	11	FF,FF	Initialize DE reg. pair with FFFFH.
LOOP2	4120H	DCX D	1B		Decrement DE reg pair.
	4121H	MOV A,D	7A		Move data from D to A reg.
	4122H	ORA E	B3		OR reg with E reg.
	4123H	JNZ LOOP2	C2	20,41	Go to loop2.
	4126H	DCR B	05		Decrement B reg.
	4127H	JNZ LOOP3	C2	1D,41	Go to loop3.
	412AH	RET	C9		Return.

RESULT:

Thus the Assembly Language Program to control the traffic light system had been written and output was verified.

Experiment No: **Date:**

STEPPER MOTOR INTERFACE

AIM:

To write an assembly language program to run a stepper motor at different speeds in two directions.

APPARATUS REQUIRED:

(i) 8085 Microprocessor kit with keyboard.
(ii) Power Cable.
(iii)Stepper Motor Interface with connecting cable.
(iv) Stepper motor

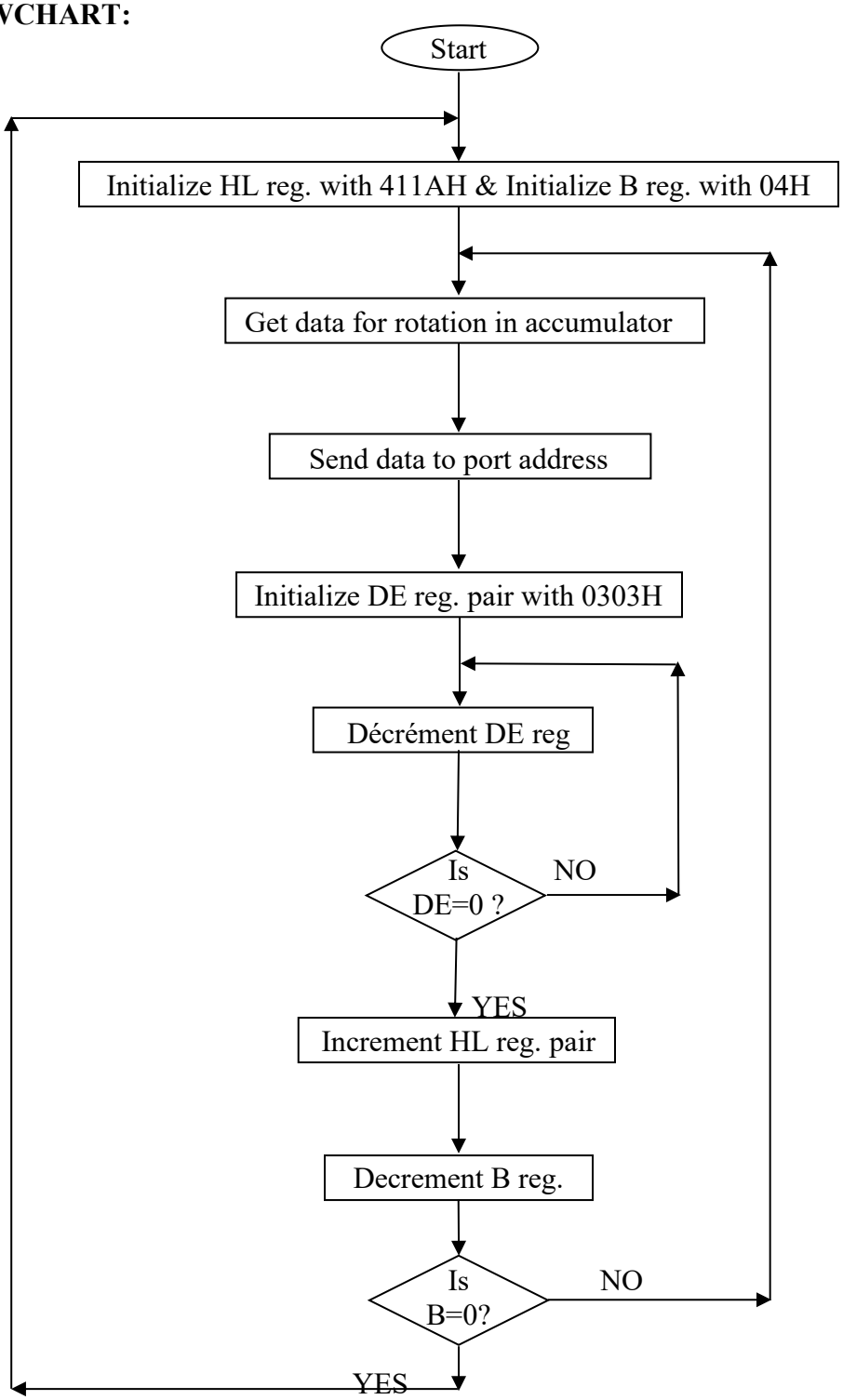
ALGORITHM:

Step 1: Initialize HL register pair with address 4150H.
Step 2: Initialize register B with total number of data for rotation.
Step 3: Get the data for rotation in accumulator.
Step 4: Send the data from accumulator to port address of stepper motor interface.
Step 5: Initialize DE register pair with data 030H.
Step 6: Decrement DE register pair.
Step 7: Check the value of DE reg pair. If it is not equal to zero, go to step 3. Otherwise go to step 1.

OBSERVATION:

	ADDRESS	**DATA**	**DATA**
INPUT	411AH	0A	09
	411BH	06	05
	411CH	05	06
	411DH	09	0A
OUTPUT	Stepper motor	Rotated in forward Direction in 4 different speeds	Rotated in reverse Direction in 4 different speeds

FLOWCHART:


Start
Initialize HL reg. with 411AH & Initialize B reg. with 04H
Get data for rotation in accumulator
Send data to port address
Initialize DE reg. pair with 0303H
Décrément DE reg
Is DE=0 ?
NO
YES
Increment HL reg. pair
Decrement B reg.
Is B=0?
NO
YES

PROGRAM:

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	4100H	LXI H,411AH	21	1A,41	Initialize HL reg. pair with 411AH.
	4103H	MVI B,04H	06	04	Load data 04H to B reg.
REPEAT	4105H	MOV A,M	7E		Send data to accumulator.
	4106H	OUT C0H	D3	C0	Send data to output port.
	4108H	LXI D,0303H	11	03,03	Initialize DE reg. pair with 0303H.
DELAY	410BH	NOP	00		No operation.
	410CH	DCX D	1B		Decrement DE reg. pair.
	410DH	MOV A,E	7B		Move data from E to A.
	410EH	ORA D	B2		OR the accumulator content with D reg.
	410FH	JNZ DELAY	C2	0B,41	If DE not equal to zero, go to DELAY.
	4112H	INX H	23		Increment HL reg. pair.
	4113H	DCR B	05		Decrement B reg.
	4114H	JNZ REPEAT	C2	05,41	If B not equal to zero, go to REPEAT.
	4117H	JMP START	C3	00,41	Go to START.

Uni polar stepping scheme:

	Clock wise				Anti clock wise			
Step	A1	A2	B1	B2	A1	A2	B1	B2
1	1	0	0	0	1	0	0	0
2	0	0	0	1	0	0	1	0
3	0	1	0	0	0	1	0	0
4	0	0	1	0	0	0	0	1

Two Phase Stepping Scheme:

	Clock wise				Anti clock wise			
Step	A1	A2	B1	B2	A1	A2	B1	B2
1	1	0	0	1	1	0	1	0
2	0	1	0	1	0	1	1	0
3	0	1	1	0	0	1	0	1
4	1	0	1	0	1	0	0	1

RESULT:

Thus the program has been written and the stepper motor was rotated in two directions at different speeds.

Experiment No: **Date:**

WAVEFORM GENERATION USING DAC

AIM:

To generate different waveforms using DAC interface with 8085 microprocessor.

APPARATUS REQUIRED:

(i) 8085 Microprocessor kit with keyboard.
(ii) Power Cable.
(iii)DAC Interface.
(iv) CRO

SQUARE WAVEFORM

ALGORITHM:

Step 1: Get the data 00H & output data at DAC1.
Step 2: Get the data FFH & output data at DAC4.
Step 3: Maintain data for sometime.
Step 4: Repeat the above steps.

SAWTOOTH WAVEFORM

ALGORITHM:

Step 1: Get the data 00H & output data at DAC1.
Step 2: Increment data to FFH.
Step 3: Repeat the above steps.
Step 4: Stop the process.

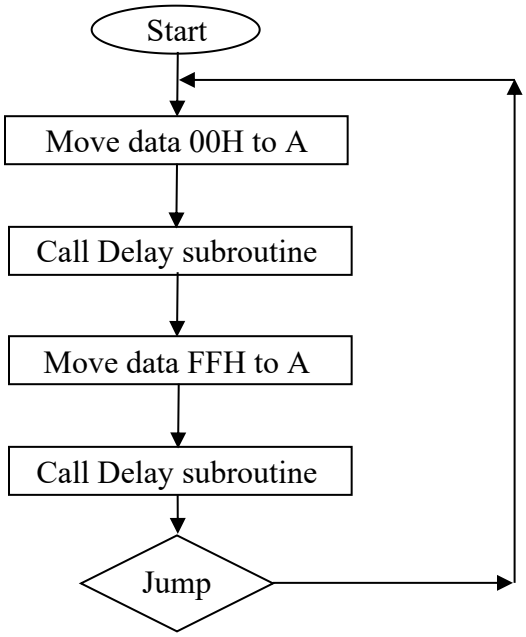
TRIANGULAR WVEFORM

ALGORITHM:

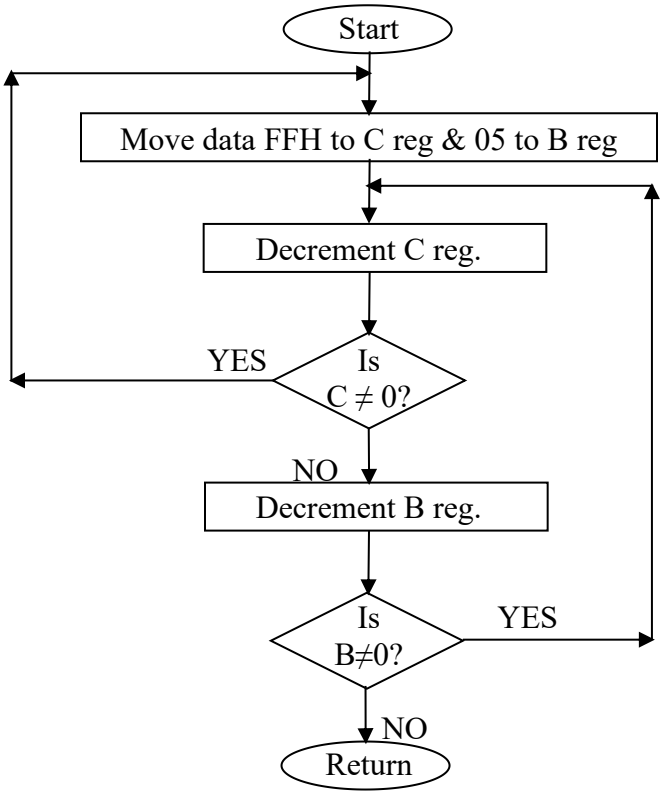
Step 1: Get the data 00H & output data at DAC1 or DAC2.
Step 2: Increment data to FFH.
Step 3: Decrement data from FFH to 00H.
Step 4: Repeat the above steps.
Step 5: Stop the process.

FLOWCHART:

SQUARE WAVEFORM

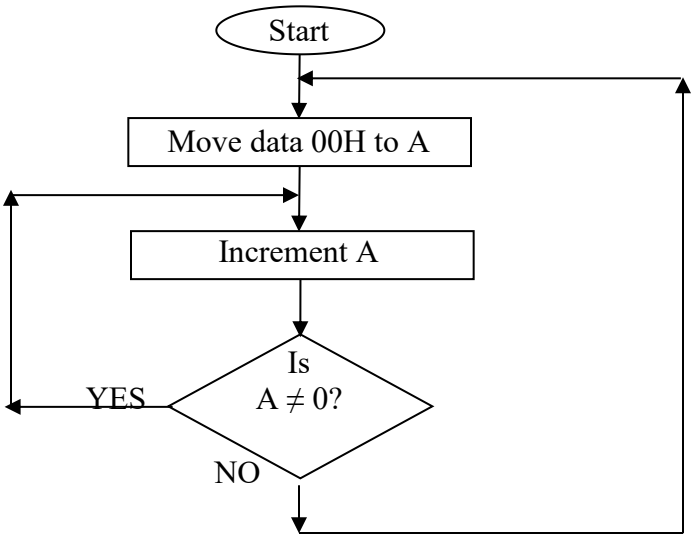


DELAY:

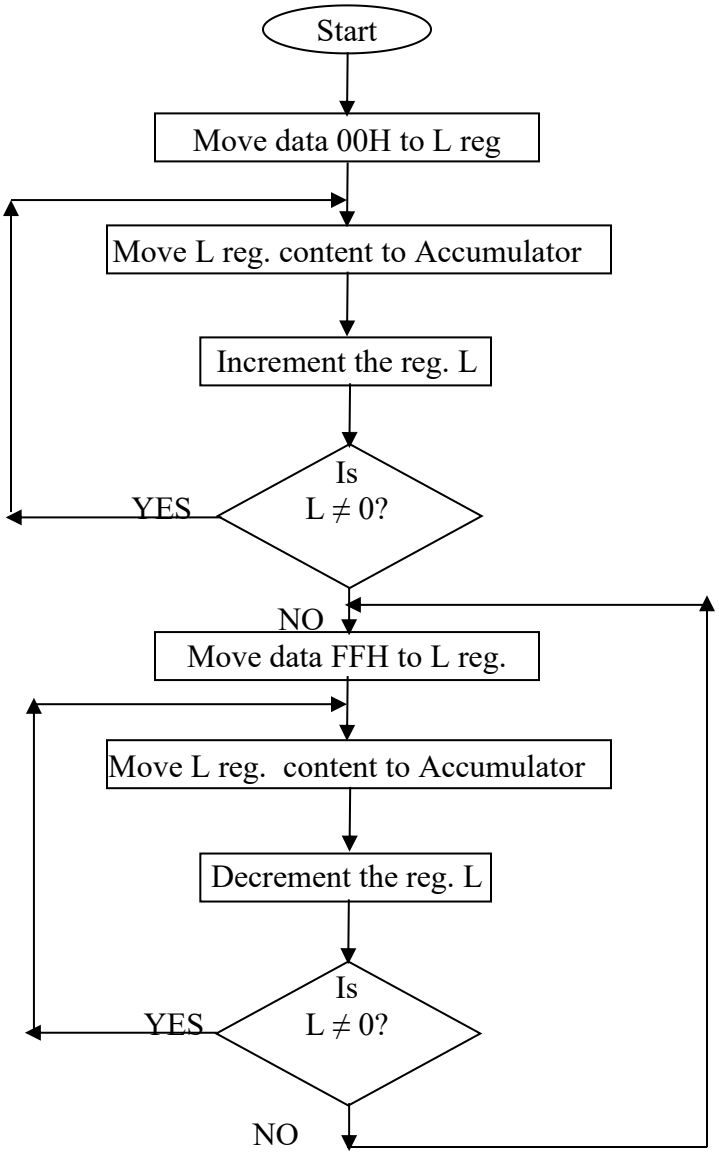


FLOWCHART:

SAWTOOTH WAVEFORM



TRIANGULAR WVEFORM



PROGRAM:

SQUARE WAVEFORM

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	4300H	MVI A,00H	3E	00	
	4302H	OUT C8H	D3	C8	
	4304H	CALL DELAY	CD	11,43	
	4307H	MVI A,FFH	3E	FF	
	4309H	OUT C8H	D3	C8	
	430BH	CALL DELAY	CD	11,43	
	430EH	JMP START	C3	00,43	
DELAY	4311H	MVI B,05H	06	05	
LOOP1	4313H	MVI C,FFH	0E	FF	
LOOP2	4315H	DCR C	0D		
	4316H	JNZ LOOP2	C2	15,43	
	4319H	DCR B	05		
	431AH	JNZ LOOP1	C2	13,43	
	431DH	RET	C9		

SAWTOOTH WAVEFORM

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	4500H	MVI A,00H	3E	00	
LOOP	4502H	OUT C0H	D3	C0	
	4504H	INR A	3C		
	4505H	JNZ LOOP	C2	02,45	
	4508H	JMP START	C3	00,45	

TRIANGULAR WVEFORM

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
START	4600H	MVI L,00H	2E	00	
LOOP1	4602H	MOV A,L	7D		
	4603H	OUT C8H	D3	C8	
	4605H	INR L	2C		
	4606H	JNZ LOOP1	C2	02,46	
	4609H	MVI L,FFH	2E	FF	
LOOP2	460BH	MOV A,L	7D		
	460CH	OUT C8H	D3	C8	
	460EH	DCR L	2D		
	460FH	JNZ LOOP2	C2	0B,46	
	4612H	JMP START	C3	00,46	

RESULT:

Thus different waveforms were generated using DAC and 8085 microprocessor.

Experiment No: **Date:**

DC MOTOR INTERFACING

AIM:

To measure the speed of DC motor using 8085 assembly language program.

APPARATUS REQUIRED:

(i) 8085 Microprocessor kit with keyboard.
(ii) Power Cable.
(iii)DC Motor interface.

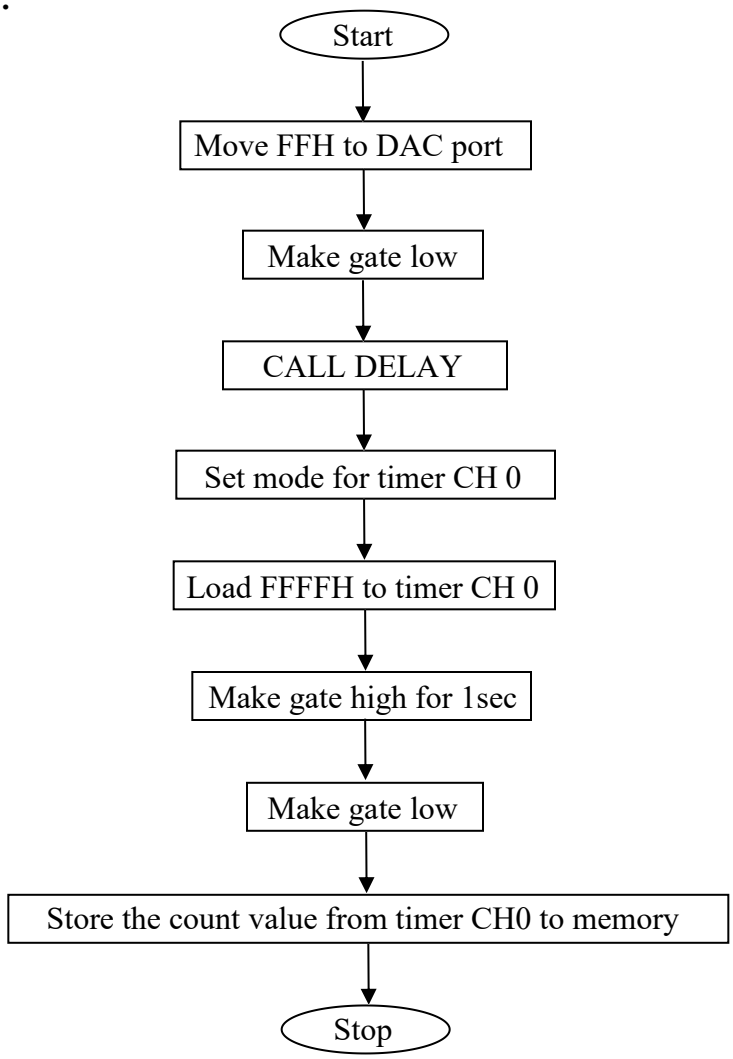
ALGORITHM:

Step 1: Run DC motor at full speed by latching FF to the DAC.
Step 2: Make the gate of channel 0 to logic 0.
Step 3: Call Delay for stable running.
Step 4: set mode as interrupt on terminal port.
Step 5: Load maximum value FFFFH to timer channel 0.
Step 6: Enable gate for one second.
Step 7: Make gate low.
Step 8: Store the count value from timer CH 0 to memory.
Step 9: Get the actual counter value & divide it by 4.
Step 10: Multiply it with 60 to convert RPM.
Step 11: Convert hexadecimal into decimal for actual RPM.

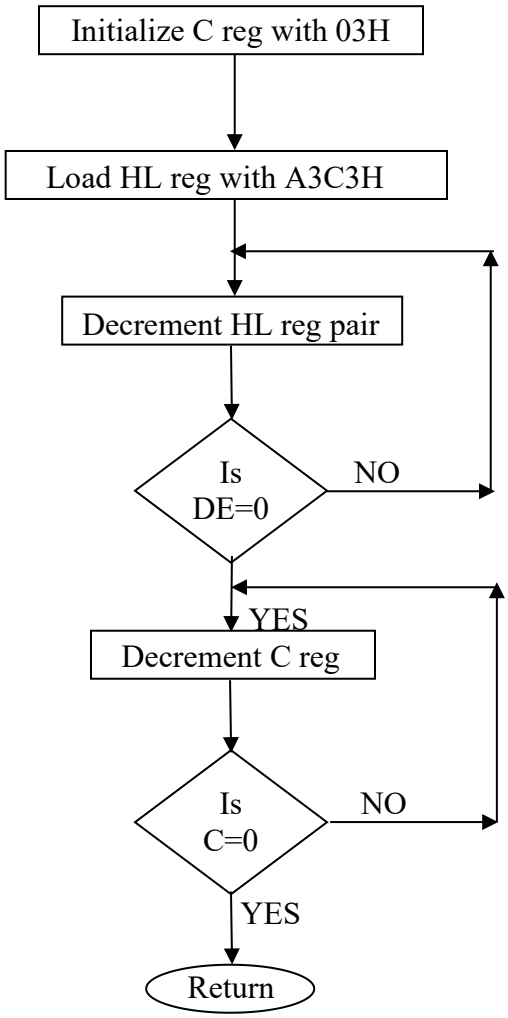
OBSERVATION:

	ADDRESS	DATA	COMMENTS
OUTPUT	4500H	87	LSB of timer channel 0
	4501H	00	MSB of timer channel 0

FLOWCHART:


Start
Move FFH to DAC port
Make gate low
CALL DELAY
Set mode for timer CH 0
Load FFFFH to timer CH 0
Make gate high for 1sec
Make gate low
Store the count value from timer CH0 to memory
Stop

DELAY:


Initialize C reg with 03H
Load HL reg with A3C3H
Decrement HL reg pair
Is
DE=0
NO
YES
Decrement C reg
Is
C=0
NO
YES
Return

PROGRAM:

LABEL	ADDRESS	MNEMONICS	OPCODE	OPERAND	COMMENTS
	4100H	**MVI A,FFH**	3E	FF	Load FF to DAC Port.
	4102H	**OUT C0H**	D3	C0	
	4104H	**MVI A,00H**	3E	00	Initialize gate as low.
	4106H	**OUT D8H**	D3	D8	
	4108H	**CALLDELAY**	CD	2B,41	Delay for stable running.
	410BH	**MVI A,30H**	3E	30	
	410DH	**OUT CEH**	D3	CE	
	410FH	**MVI A,FFH**	3E	FF	Load FFto timer CH0
	4111H	**OUT C8H**	D3	C8	
	4113H	**OUT C8H**	D3	C8	
	4115H	**MVI A,00H**	3E	00	Make gate high.
	4117H	**OUT D0H**	D3	D0	
	4119H	**CALLDELAY**	CD	2B,41	Delay for 1sec.
	411CH	**MVI A,00H**	3E	00	Make gate as low.
	411EH	**OUT D8H**	D3	D8	
	4120H	**IN C8H**	DB	C8	In data from CH0 to accumulator.
	4122H	**STA 4500H**	32	00,45	Store data in memory.
	4125H	**MVI A,00H**	3E	00	
	4127H	**STA 4501H**	32	01,45	Clear buffer
	412AH	**HLT**	76		
DELAY	412BH	**MVI C,03H**	0E	03	Load C reg with data 03H.
LOOP2	412DH	**LXI H,A3C3H**	21	C3,A3	Load HL reg with A3C3H.
LOOP1	4130H	**DCX H**	2B		Decrement HL reg.
	4131H	**MOV A,L**	1D		
	4132H	**ORA H**	B4		
	4133H	**JNZ LOOP1**	C2	30,41	If DE#0 go to loop1.
	4136H	**DCR C**	0D		Decrement C reg.
	4137H	**JNZ LOOP2**	C2	2D,41	If C#0 go to loop2
	413AH	**RET**	C9		Return to main program.
	413BH	**HLT**	76		Stop

RESULT:

Thus the program had been written and the speed of DC motor was measured as 65400rpm.