

DMI COLLEGE OF ENGINEERING

PALANCHUR, CHENNAI- 600 123



Department of Electrical and Electronics

Engineering

Subject Code: **EE8681**

Subject Name: **Microprocessor and Microcontroller laboratory**

Name :

Reg. No. :

Branch :

Year & Semester :

SYLLABUS

EE8681 MICROPROCESSOR AND MICRO CONTROLLER LABORATORY

OBJECTIVES:

To provide training on programming of microprocessors and microcontrollers and understand the interface requirements.

LIST OF EXPERIMENTS:

1. Simple arithmetic operations: addition / subtraction / multiplication / division.
2. Programming with control instructions:
 - a. Ascending / Descending order, Maximum / Minimum of numbers
 - b. Programs using Rotate instructions
 - i. Hex / ASCII / BCD code conversions.
3. Interface Experiments: with 8085
 - i. A/D Interfacing. & D/A Interfacing.
4. Traffic light controller.
5. I/O Port / Serial communication
6. Programming Practices with Simulators/Emulators/open source
7. Read a key ,interface display
8. Demonstration of basic instructions with 8051 Micro controller execution, including:
 - a. Conditional jumps, looping
 - b. Calling subroutines.
9. Programming I/O Port 8051
 - i. Study on interface with A/D & D/A
 - ii. Study on interface with DC & AC motor.
10. Application hardware development using embedded processors.

**EE6612 MICROPROCESSOR AND MICROCONTROLLER LABORATORY
LIST OF EXPERIMENTS**

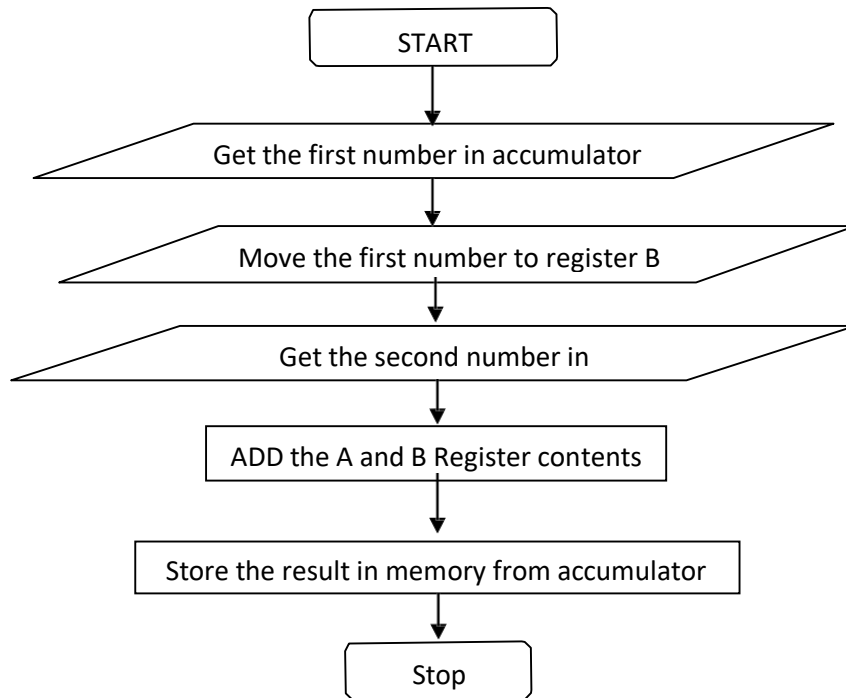
Ex.No.	Name of the Experiment
	8085 MICROPROCESSOR PROGRAMS
1	A. Addition of two 8-bit data without carry
	B. Addition of two 8-bit data with carry
2	A. Subtraction of two 8 bit data without carry
	B. Subtraction of two 8-bit data with carry
3	A. Addition of two 16-bit data
	B. Subtraction of two 16 bit data
4	A. Multiplication of two 8- bit data
	B. Division of two 8-bit data
5	A. Smallest number in an array of data
	B. Largest number in an array of data
6	A. Arrange an array of data in ascending order
	B. Arrange an array of data in descending order
7	A. Code conversion - ASCII to HEXA
	B. Code conversion - HEXA to ASCII
8	A. Code conversion BCD to HEXA
	B. Code conversion HEXA to BCD
	8051 MICROCONTROLLER PROGRAMS
9	A. Addition of two 8-bit data
	B. Subtraction of two 8-bit data
10	A. Multiplication of two 8-bit data
	B. Division of two 8-bit data
11	A. Sum of the Elements
	INTERFACING PROGRAMS
	B. Stepper motor interface using 8051 microcontroller
12	Interfacing 8279 with 8085 microprocessor (Rolling display)
13	Traffic light control system using 8085 microprocessor
14	Interfacing of D to A converter using 8085 microprocessor
15	Interfacing of A to D converter using 8085 microprocessor
16	Serial port interface using 8085 microprocessor
17	Interfacing of D to A converter using 8051 microcontroller
18	Interfacing of A to D converter using 8051 microcontroller
19	Interfacing of DC motor using 8051 microcontroller
20	Interfacing of AC motor using 8051 microcontroller
	MINI PROJECT
21	Development of a System with Higher than 8051 microcontroller

INDEX

Ex. No.	Date	Name of the Experiment	Marks	Staff Signature

8085 Microprocessor Programs

FLOW CHART:



INPUT & OUTPUT TABULATION:

Memory Address	Input data	Memory Address	Output data
8200		8202	
8201			
8200		8202	
8201			

Ex. No.:**Date :** **1.A. ADDITION OF TWO 8-BIT DATA WITHOUT CARRY****AIM:**

To add two 8 bit numbers stored at consecutive memory location using 8085 microprocessor without carry.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

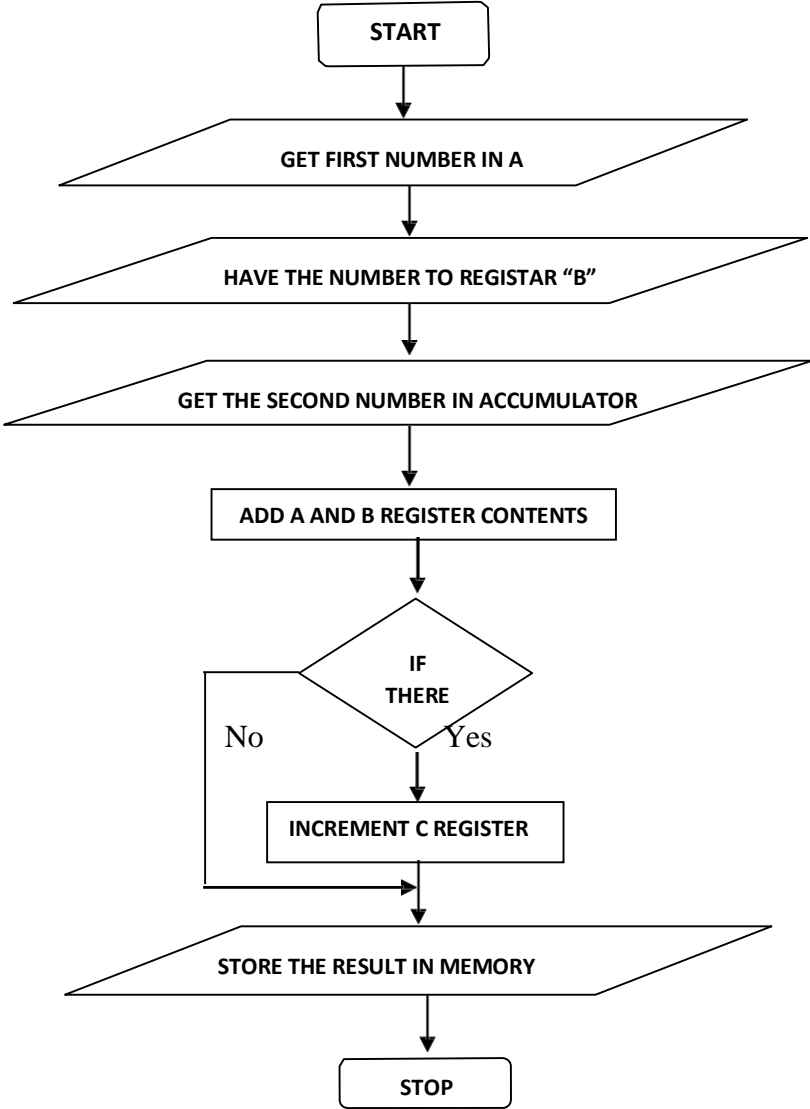
1. Start the program by initializing memory pointer to data location.
2. Get the first number and store in accumulator.
3. Move the first number to register B.
4. Get second number and store in accumulator A.
5. Add two numbers and result is in accumulator A.
6. Store the result from accumulator to memory.
7. Stop the program.

PROGRAM:

ADDRESS	LABEL	PNEUMONIC	OPCODE	COMMENTS
8100	START	LDA 8200	3A	Load the first number in accumulator from Memory
8101			00	
8102			82	
8103		MOV B, A	47	Move the data from accumulator to B
8104		LDA 8201	3A	Load the Second number in accumulator from Memory
8105			01	
8106			82	
8107		ADD B	80	Addition of B with A register values.
8108		STA 8202	32	Store the result from accumulator to Memory
8109			02	
810A			82	
810B		HLT	76	Stop the program

RESULT:

FLOWCHART:



Ex. No.:**Date :** **1.B. ADDITION OF TWO 8-BIT DATA WITH CARRY****AIM:**

To add two 8-bit numbers stored at consecutive memory location using 8085 microprocessor with carry.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Start the program by initializing the memory pointer to data location.
2. Get the first number or data in accumulator.
3. Move the first number to register B.
4. Get the second number in accumulator A.
5. Add two numbers and result is in accumulator A.
6. If carry is present, increment register C by one, otherwise go to next step.
7. Store the result in memory from accumulator and register C.
8. Stop the program.

PROGRAM:

ADDRESS	LABEL	PNEMONICS	OPCODE	COMMENTS
8100	START	LDA 8200	3A	Load First Data in Accumulator A
8101			00	
8102			82	
8103		MOV B, A	47	Move Data from Accumulator To B
8104		LDA 8201	3A	Load Second Data in Accumulator A
8105			01	
8106			82	
8107		MVI C,00	0E	Clear C Register
8108			00	
8109		ADD B	80	Addition of B With A
810A		JNC LOOP	D2	Jump to Loop , If Result does not have Carry
810B			0E	
810C			81	
810D		INR C	0C	Increment C Register
810E	LOOP	STA 8202	32	Store the Result in Memory from Accumulator
810F			02	
8110			82	

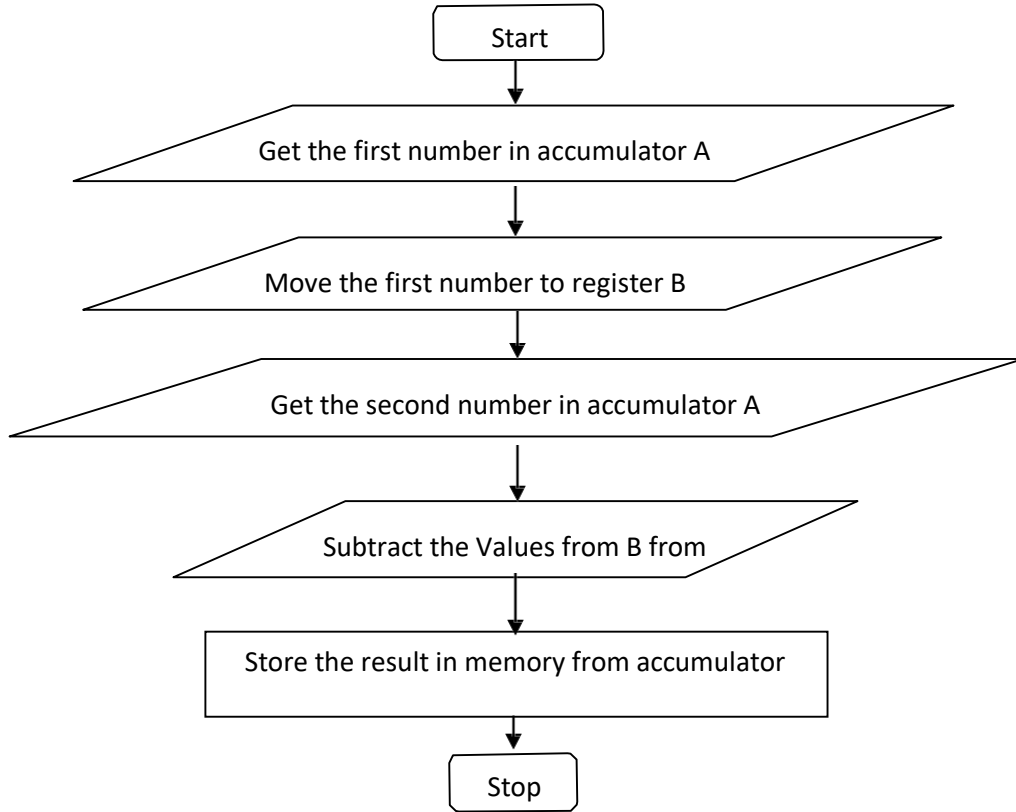
INPUT & OUTPUT TABULATION:

MEMORY ADDRESS	INPUT DATA	MEMORY ADDRESS	OUTPUT DATA
8200		8202	
8201		8203	
8200		8202	
8201		8203	

8111		MOV A,C	79	Move the Carry from C to Accumulator & Store Carry in Memory from Accumulator
8112		STA 8203	32	
8113			03	
8114			82	
8115		HLT	76	Stop the Program

RESULT:

FLOW CHART:



INPUT & OUTPUT TABULATION:

Memory address	Input data	Memory address	Output data
8200		8202	
8201			
8200		8202	
8201			

Ex. No.:

Date : 2.A. SUBTRACTION OF TWO 8 BIT DATA WITHOUT CARRY

AIM:

To subtract two 8 bit data's stored at memory location without carry using 8085 microprocessor

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

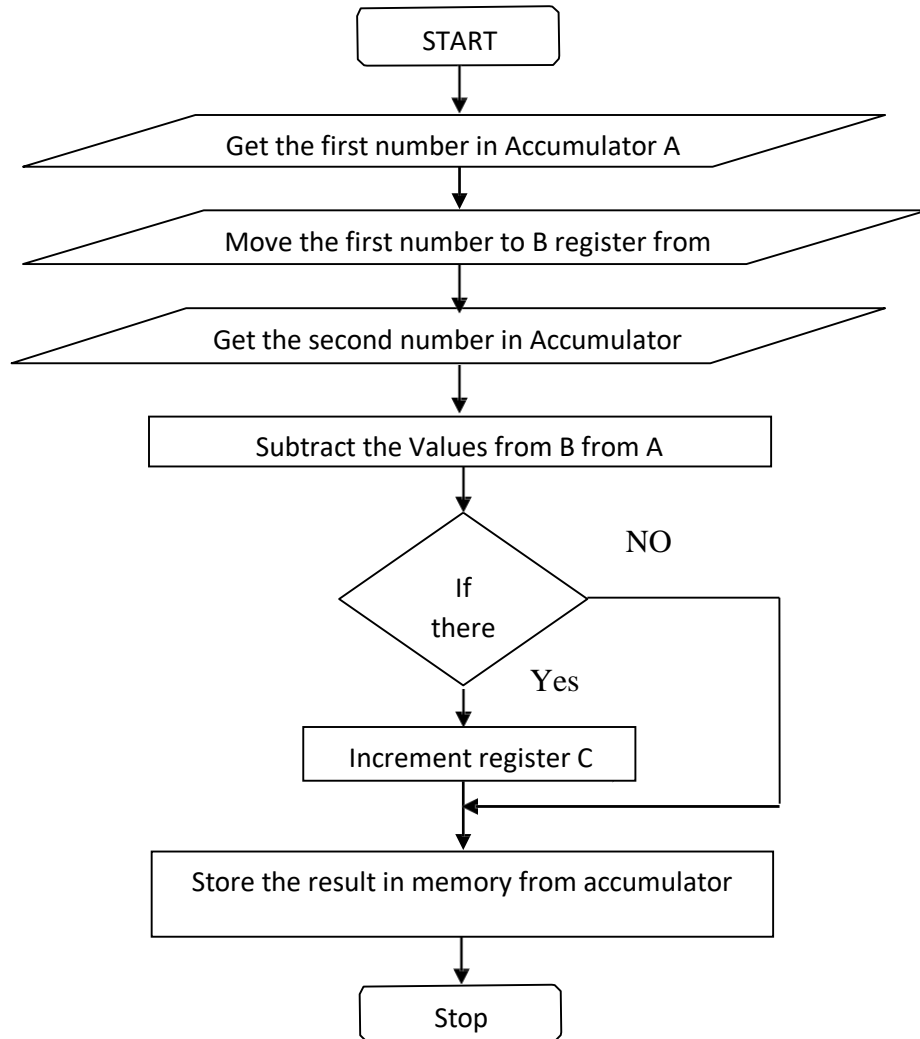
1. Start the program by initializing the memory pointer to data location
2. Get the first number from memory to accumulator
3. Move the first number to register B
4. Get the second number in accumulator from memory
5. Store the result in memory from accumulator
6. Stop the program

PROGRAM:

ADDRESS	LABEL	PNEUMONIC	OPCODE	COMMENTS
8100	START	LDA 8200	3A	Load the first data in accumulator A from memory
8101			00	
8102			82	
8103		MOV B,A	47	Move the first data to register B form accumulator A
8104		LDA 8201	3A	Load the Second data in accumulator A from memory
8105			01	
8106			82	
8107		SUB B	90	Subtract the value from B from A
8108		STA 8202	32	Store the result in memory from accumulator
8109			02	
810A			82	
810B		HLT	76	Stop the program

RESULT:

FLOWCHART:



Ex. No.:**Date :** **2.B. SUBTRACTION OF TWO 8-BIT DATA WITH CARRY****AIM:**

To subtract two 8-bit numbers stored at consecutive memory location using 8085.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Start the program by initializing the memory location to data pointer.
2. Get the first number from memory in accumulator.
3. Move the first number to register B.
4. Get the second number from memory in accumulator.
5. Subtract two numbers (B from A) and store it in accumulator.
6. Store the result in memory from accumulator.
7. Stop the program.

PROGRAM:

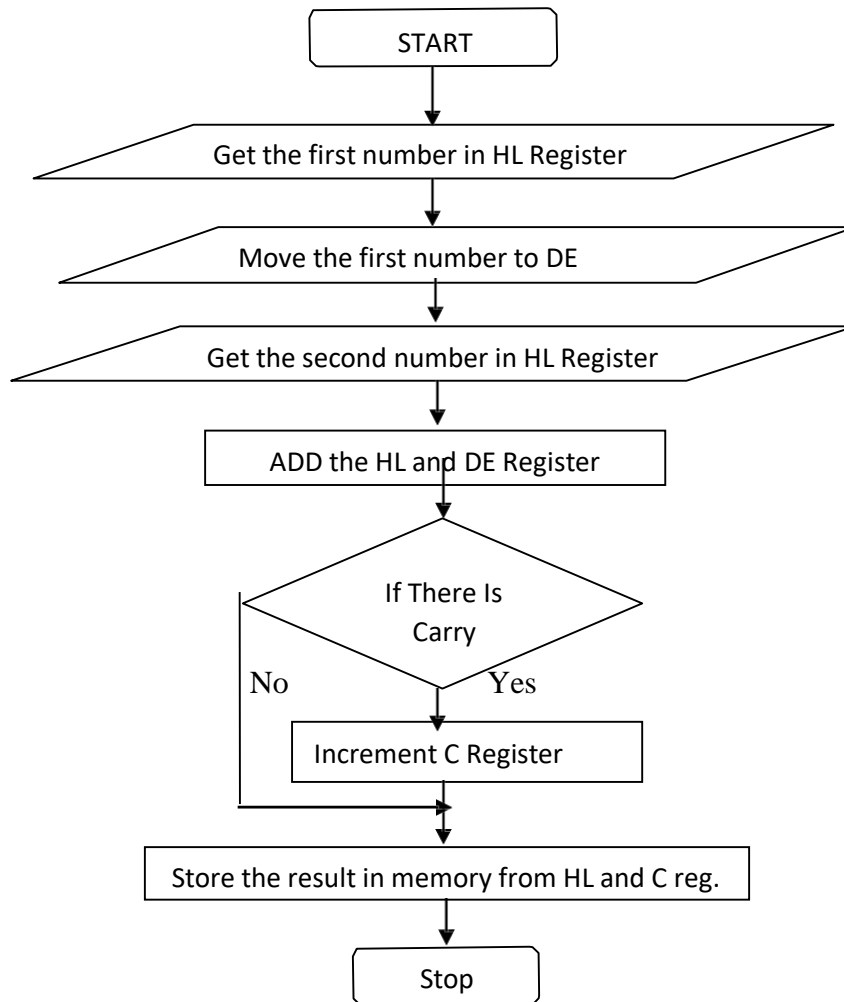
ADDRESS	LABEL	PNEUMONIC	OPCODE	COMMENTS
8100	START	LDA 8200	3A	Load the first data in accumulator A from memory
8101			00	
8102			82	
8103		MOV B,A	47	Move data from A to B
8104		LDA 8201	3A	Load the second data in accumulator A from memory
8105			01	
8106			82	
8107		MVI C,00	0E	Clear C register
8108			00	
8109		SUB B	90	Subtract B from A

INPUT & OUTPUT TABULATION:

Memory Address	Input data	Memory Address	Input data
8200		8202	
8201		8203	
8200		8202	
8201		8203	

810A		JNC LOOP	D2	Jump to location of the result doesn't have carry
810B			0E	
810C			81	
810D		INRC	0C	Increment C register
810E	LOOP	STA 8202	32	Store the result from accumulator
810F			02	
8110			82	
8111		MOV A,C	79	Move Borrow from C to A
8112		STA 8203	32	Store carry value from accumulator
8113			03	
8114			82	
8115		HLT	76	Stop the program

RESULT:

FLOW CHART:

Ex. No.:**Date :** **3.A. ADDITION OF TWO 16-BIT DATA****AIM:**

To add two 16 bit numbers stored at consecutive memory location using 8085 microprocessor with carry.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Start the program by initializing memory pointer to data location.
2. Get the first number and store in HL register.
3. Move the first number to register DE register.
4. Get second number and store in HL register.
5. Add two numbers and result is in HL register and C register.
6. Store the result from HL & C register to memory.
7. Stop the program.

PROGRAM:

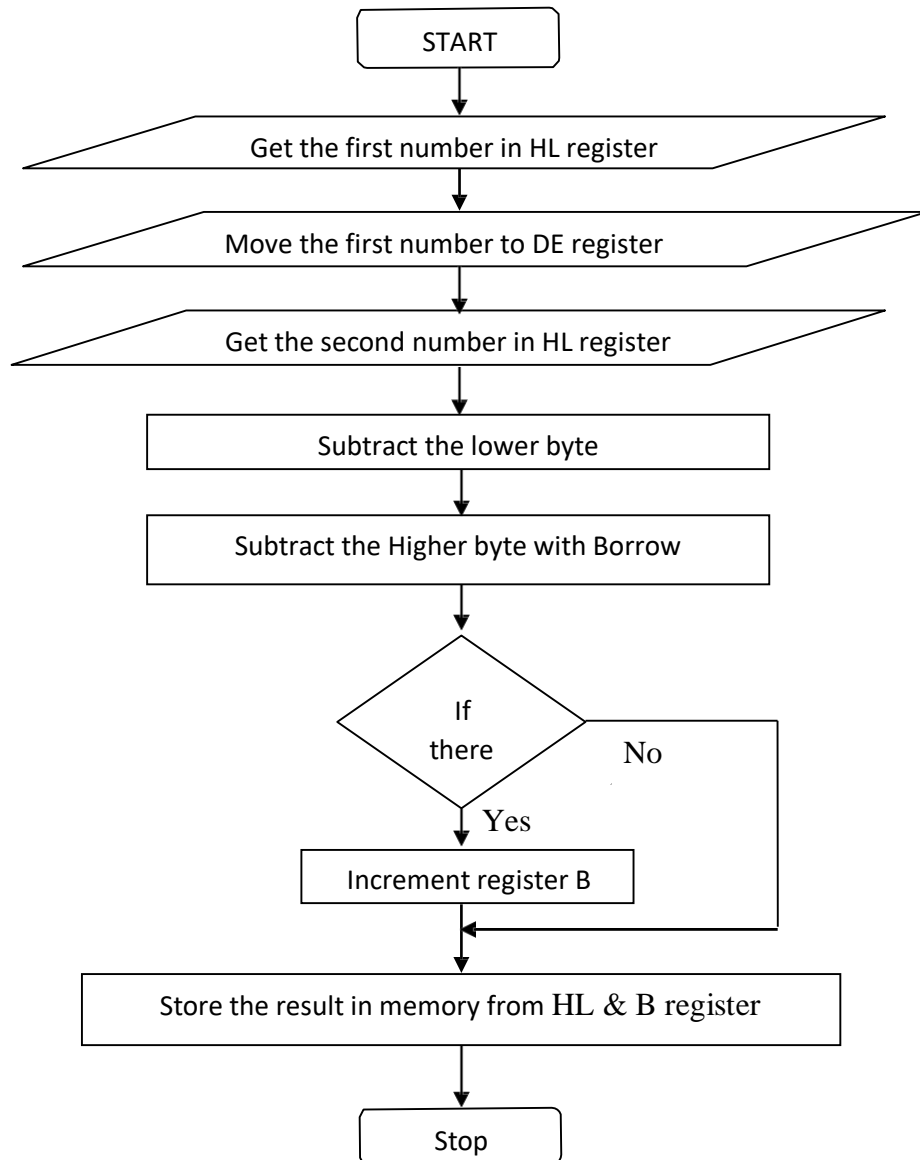
ADDRESS	LABEL	PNEMONICS	OPCODE	COMMENTS
8100	START	MVI C,00	0E	Clear C Register
8101			00	
8102		LHLD 8200	2A	Load First Data in HL register
8103			00	
8104			82	
8105		XCHG	EB	Move Data To DE register
8106		LHLD 8202	2A	Load Second Data in HL register
8107			02	
8108			82	
8109		DAD D	19	Add HL & DE registers
810A		JNC LOOP	D2	Jump to Loop , If Result does not have Carry
810B			0E	
810C			81	
810D		INR C	0C	Increment C Register
810E	LOOP	SHLD 8300	22	Store the Result in Memory from HL register
810F			00	
8110			83	

INPUT & OUTPUT TABULATION:

Memory Address	Input data	Memory Address	Output data
8200		8300	
8201		8301	
8202		8302	
8203			

8111		MOV A,C	79	Move the Carry from C to Accumulator & Store Carry in Memory from Accumulator
8112		STA 8302	32	
8113			03	
8114			82	
8115		HLT	76	Stop the Program

RESULT:

FLOWCHART:

Ex. No.:**Date :** **3.B. SUBTRACTION OF TWO 16-BIT DATA****AIM:**

To subtract two 16-bit numbers stored at consecutive memory location using 8085.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Start the program by initializing the memory location to data pointer.
2. Get the first number from memory in HL register.
3. Move the first number to DE register.
4. Get the second number from memory in HL register.
5. First Subtract Lower byte and then Higher byte with borrow.
6. If Borrow is present increment the B register.
7. Store the result in memory from HL & B register.
8. Stop the program.

PROGRAM:

ADDRESS	LABEL	PNEUMONIC	OPCODE	COMMENTS
8100		LXI B,0000	01	Clear B register
8101			00	
8102			00	
8103		LHLD 8200	2A	Load the first data in HL register from memory
8104			00	
8105			82	
8106		XCHG	EB	Move data to DE register
8107		LHLD 8202	2A	Load the second data in HL register from memory
8108			02	
8109			82	

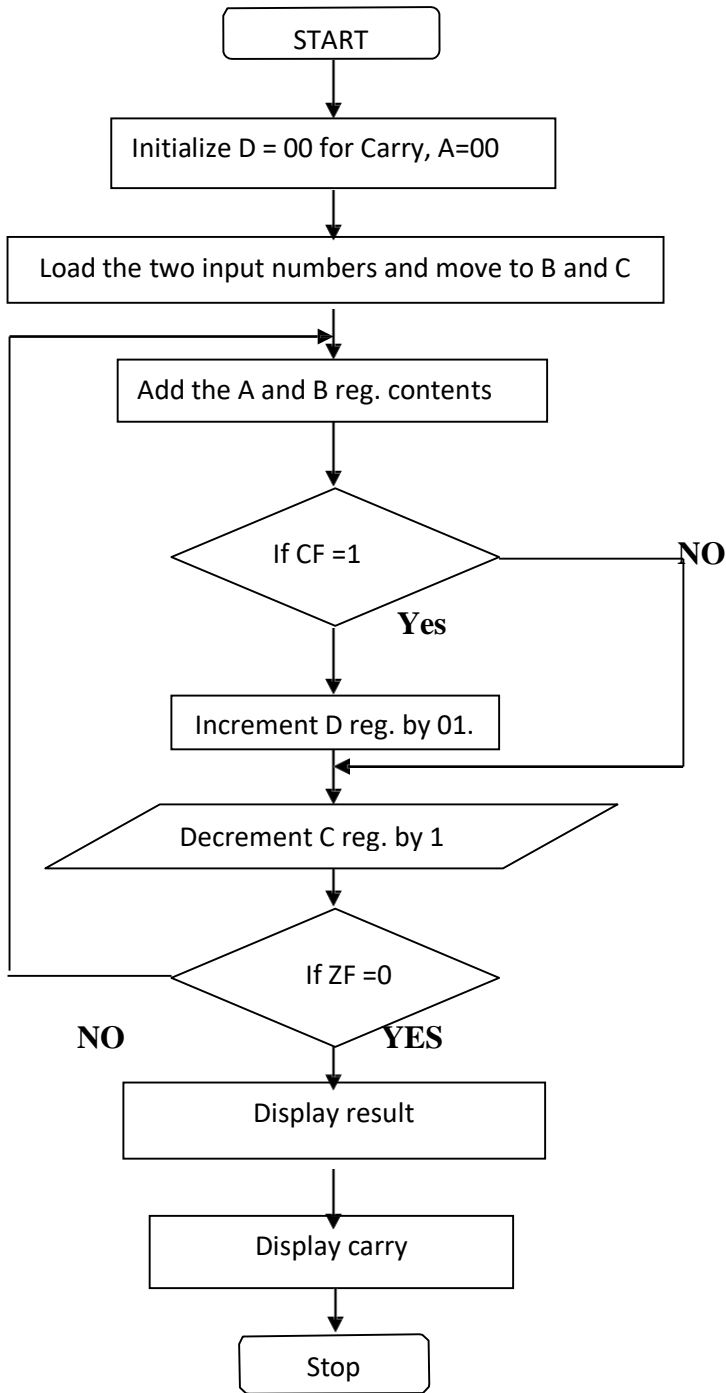
INPUT & OUTPUT TABULATION:

Memory Address	Input data	Memory Address	Input data
8200		8300	
8201		8301	
8202		8302	
8203			

810A		MOV A,E	7B	Subtract lower bytes and move lower byte result to L register.
810B		SUB L	95	
810C		MOV L,A	6F	
810D		MOV A,D	7A	Subtract higher bytes
810E		SBB H	9C	
810F		JNC LOOP	D2	Jump to location of the result doesn't have carry
8110			13	
8111			81	
8112		INX B	03	Increment B register
8113	LOOP	MOV H,A	67	Move higher byte result to H register. Finally Store the result to memory from HL register.
8114		SHLD 8300	22	
8115			00	
8116			83	
8117		MOV A,B	78	Move borrow from B to A
8118		STA 8302	32	Store Borrow value from accumulator
8119			02	
811A			83	
811B		HLT	76	Stop the program

RESULT:

FLOWCHART:



Ex. No.:**Date :** **4.A. MULTIPLICATION OF TWO 8- BIT DATA****AIM:**

To multiply two 8-bit numbers stored at consecutive memory location using 8085 microprocessor.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Start the program and Initialize D = 00 for Carry, A=00
2. Load the memory address to HL register pair
3. Move the data to a B register
4. Get the second data and move into C register.
5. Add the two register B & C contents
6. If carry is present increment the D register by 1, Otherwise go to next step.
7. Decrement the C register by 1 and repeat the step 5 until C=0.
8. Store the value of product and carry in memory location
9. Terminate the program

PROGRAM:

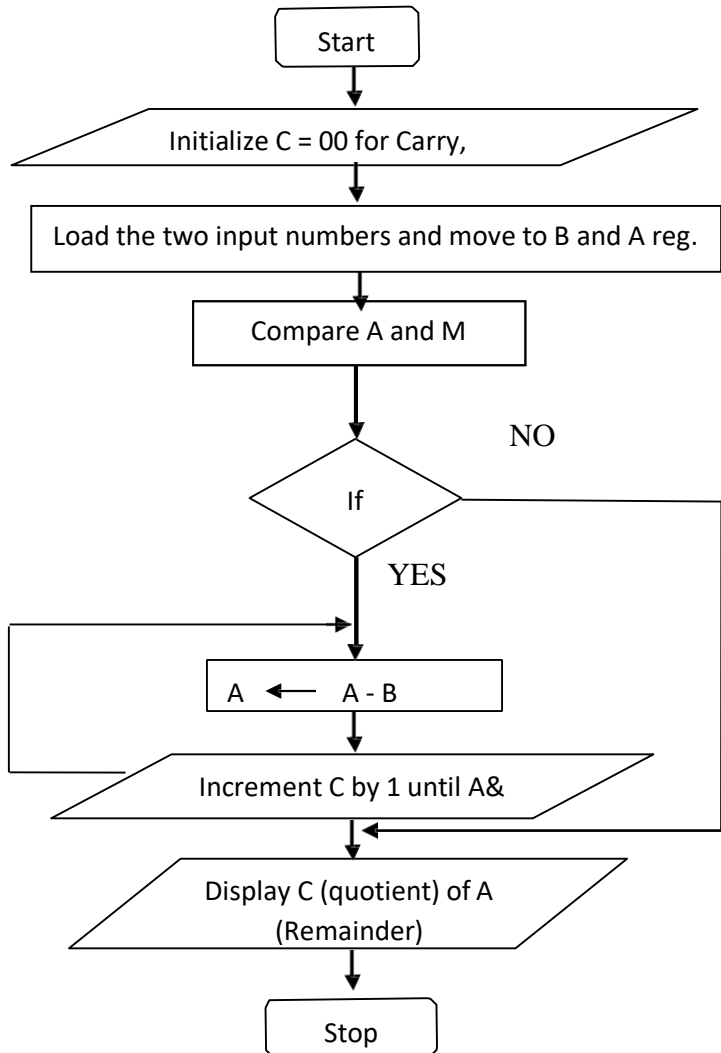
ADDRESS	LABEL	PNEUMONIC	OPCODE	COMMENTS
8100	START	MVI D,00	16	Initialize register D to 00 for carry.
8101			00	
8102		MVI A,00	3E	Initialize Accumulator content to 00
8103			00	
8104		LXI H 8200	21	Get the first number in memory
8105			00	
8106			82	
8107		MOV B,M	46	Move the first number to B- register
8108		INX H	23	Increment memory by 1
8109		MOV C,M	4E	Get the second number in C – register
810A	LOOP	ADD B	80	Add content of A register with B

INPUT & OUTPUT TABULATION:

Memory Address	Input data	Memory Address	Output data
8200		8202	
8201		8203	

810B		JNC NEXT	D2	Jump no carry to NEXT
810C			0F	
810D			81	
810E		INC D	14	Increment content of register D
810F	NEXT	DCR C	0D	Decrement content of register C
8110		JNZ LOOP	C2	Jump on no zero to LOOP
8111			0A	
8112			81	
8113		STA 8202	32	Store the result in memory
8114			02	
8115			82	
8116		MOV A,D	7A	Move D to A
8117		STA 8303	32	Store the MSB of result in memory
8118			03	
8119			82	
811A		HLT	76	Terminate the program

RESULT:

FLOW CHART:

Ex. No.:**Date :** **4.B. DIVISION OF TWO 8-BIT DATA****AIM:**

To perform the division of two 8-bit numbers using 8085 microprocessor.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Start the program by loading HL register pair with address of memory location.
2. Move the data to a register (B-register).
3. Get the second data and load into accumulator.
4. Compare the two numbers (A & B reg.) to check for carry, if carry present go to step 8.
5. Subtract the two numbers (A & B reg.).
6. Increment the value of C register for quotient.
7. If ZF=0, then repeat the step 4.
8. Store the value of remainder and Quotient in memory location.
9. Terminate the program.

PROGRAM:

ADDRESS	LABEL	PNEUMONIC	OPCODE	COMMENTS
8100	START	LXI H, 8200	21	Get the first number in memory
8101			00	
8102			82	
8103		MOV B, M	46	Get the dividend in B – register
8104		MVI C,00	0E	Clear C – register for quotient
8105			00	
8106		INX H	23	Increment memory by 1
8107		MOV A,M	7E	Get the divisor in A register
8108	NEXT	CMP B	B8	Compose A register with register B
8109		JC LOOP	DA	Jump on a carry to loop
810A			11	
810B			81	
810C		SUB B	90	Subtract A – register from B – register

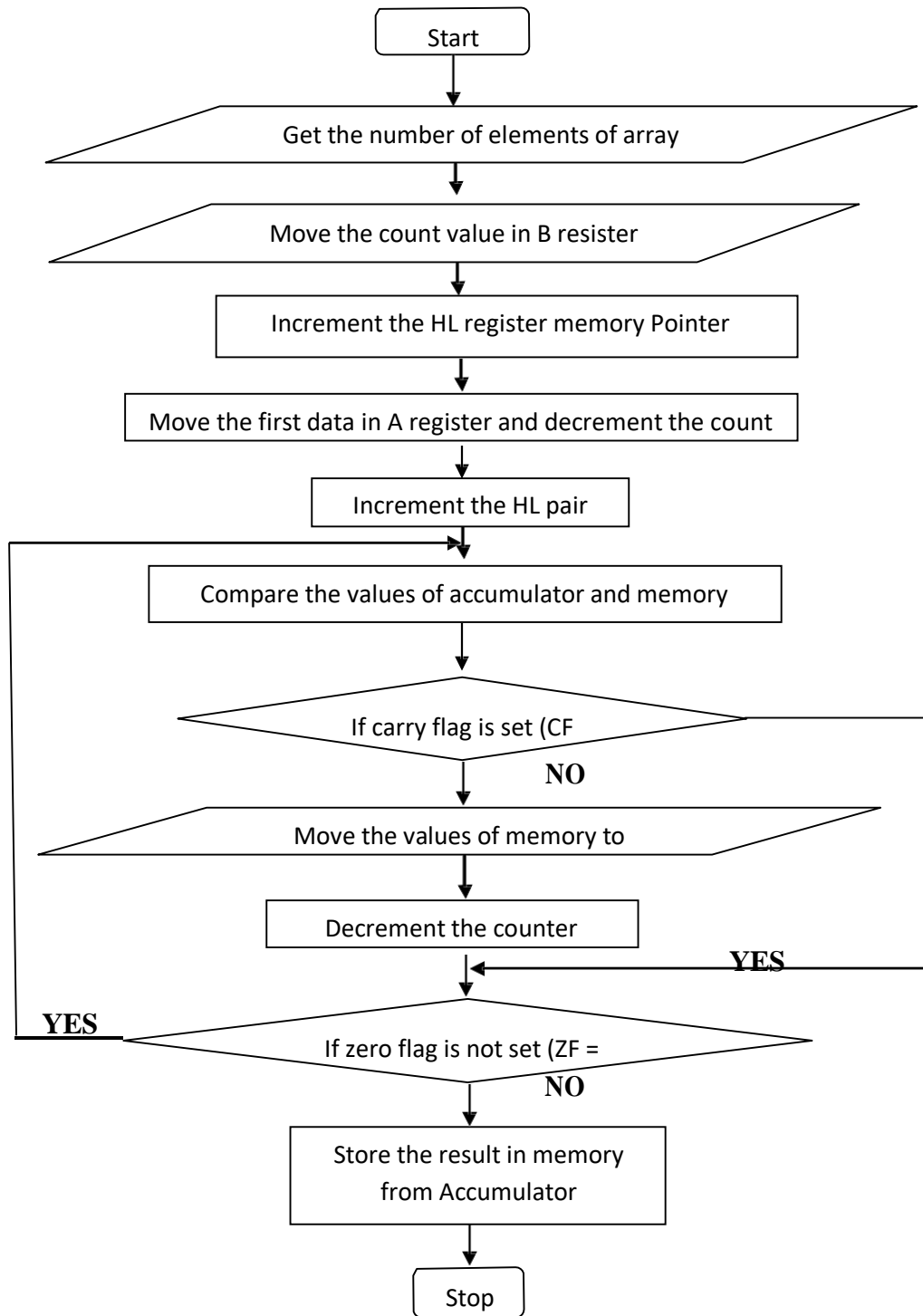
INPUT & OUTPUT TABULATION:

Memory Address	Input data	Memory Address	Output data
8200		8202	
8201		8203	

810D		INR C	0C	Increment content Of register C
810E		JNZ NEXT	C2	Jump no zero to NEXT label.
810F			08	
8110			81	
8111	LOOP	STA 8202	32	Store the remainder in memory
8112			02	
8113			82	
8114		MOV A,C	79	Move C register value to Accumulator.
8115		STA 8203	32	Store the Quotient in memory
8116			03	
8117			82	
8118		HLT	76	Stop the program

RESULT:

FLOW CHART:



Ex. No.:**Date :** **5.A. SMALLEST NUMBER IN AN ARRAY OF DATA****AIM:**

To find the smallest number in an array of datas using 8085 microprocessor

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Load the address of the first element (count) of an array in HL pair.
2. Load the count and move it in to the B –register
3. Increment the HL pair as a pointer
4. Move the first data to A-register form memory which is pointed by HL pair.
5. Decrement the count (B reg.)
6. Increment the pointer (HL reg. pair)
7. Compare the content of memory addressed by HL pair with content of A-register
8. If carry =1 go for step 10 otherwise go to step-9
9. Move the content of memory addressed by HL pair to A-register
10. Decrement the count (B reg.)
11. Check for zero of the count if ZF=0 go to step 6 otherwise go to next step
12. Store the smallest data in memory from Accumulator.

PROGRAM:

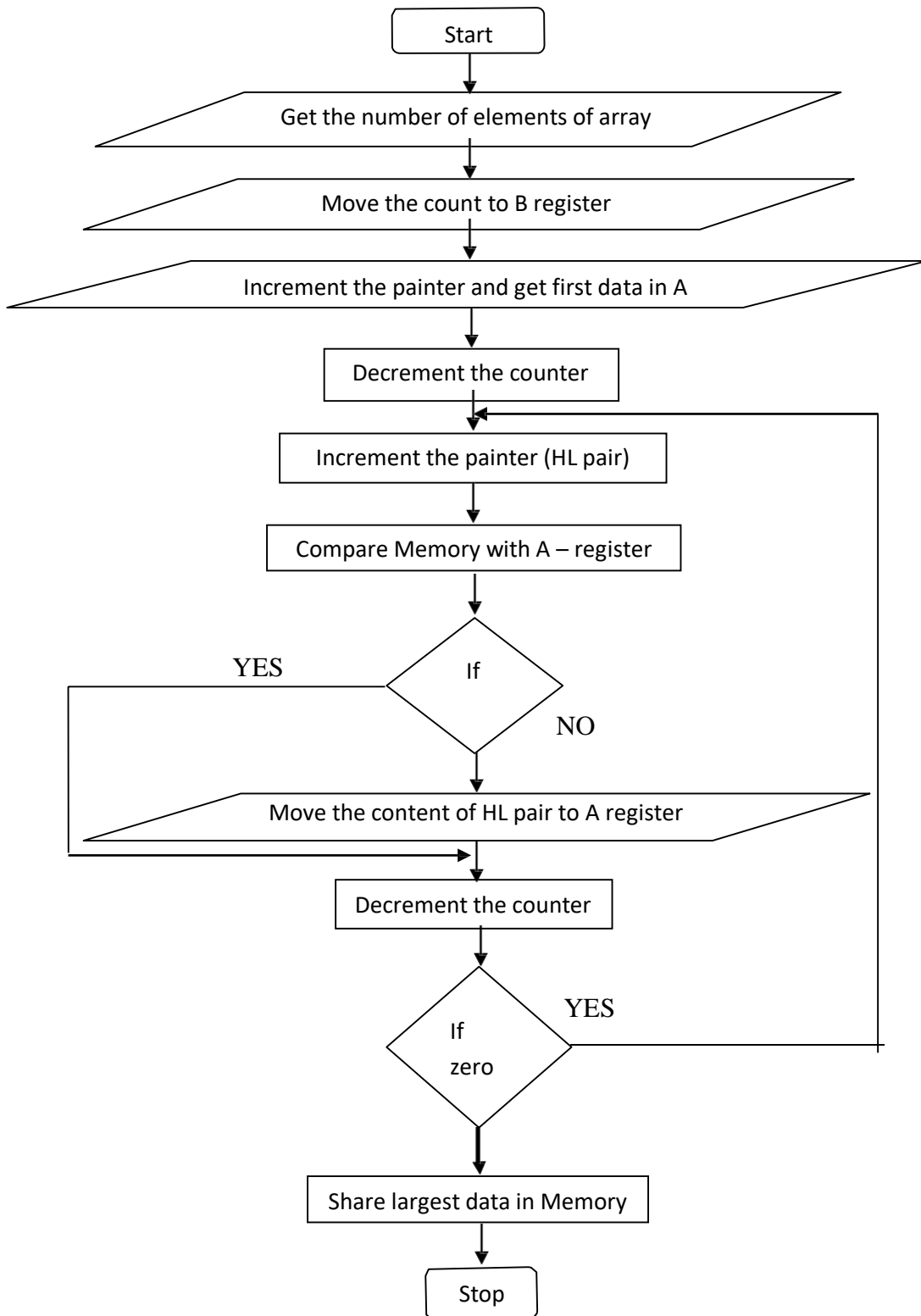
ADDRESS	LABEL	MNEMONICS	OPCODE	COMMENTS
8100	Start	LXI H, 8200	21	Set pointer for array.
8101			00	
8102			82	
8103		MOV B,M	46	Move the first data from memory to B- reg (Count)
8104		INX H	23	Increment the HL pair
8105		MOV A,M	7E	Move the second data from memory to accumulator.
8106		DCR B	05	Decrement the count

Memory Address	Input data	Memory Address	Output data
8200 (count)			
8201			
8202		8300	
8203			
8204			
8205			

INSTR & OPCODE	Label	INSTR:	Hex	Operation:
			23	Increment HL Pair
8108		CMP M	BE	Compare the content of memory with accumulator
8109		JC AHEAD	DA	If CF=1, go to Label AHEAD, otherwise go to next step.
810A			OD	
810B			81	
810C		MOV A,M	7E	Set the new values at Large
810D	AHEAD	DCR B	05	Decrement the value of B
810E		JNZ LOOP	C2	Repeat the comparison till B = 0 (ie. ZF=1)
810F			07	
8110			81	
8111		STA 8300	32	Store the largest value in memory from accumulator.
8112			00	
8113			83	
8114		HLT	76	Stop the program.

RESULT:

FLOW CHART:



Ex. No.:**Date :** **5.B. LARGEST NUMBER IN AN ARRAY OF DATA****AIM:**

To write and execute the program of largest in an array of data using 8085 microprocessor

APPRAISE REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Load the address of the first element (count) of an array in HL pair.
2. Load the count and move it in to the B –register
3. Increment the HL pair as a pointer
4. Move the first data to A-register form memory which is pointed by HL pair.
5. Decrement the count (B reg.)
6. Increment the pointer (HL reg. pair)
7. Compare the content of memory addressed by HL pair with content of A-register
8. If carry =0 go for step 10 otherwise go to step-9
9. Move the content of memory addressed by HL pair to A-register
10. Decrement the count (B reg.)
11. Check for zero of the count if ZF=0 go to step 6 otherwise go to next step
12. Store the largest data in memory from Accumulator.

PROGRAM:**INPUT & OUTPUT TABULATION:**

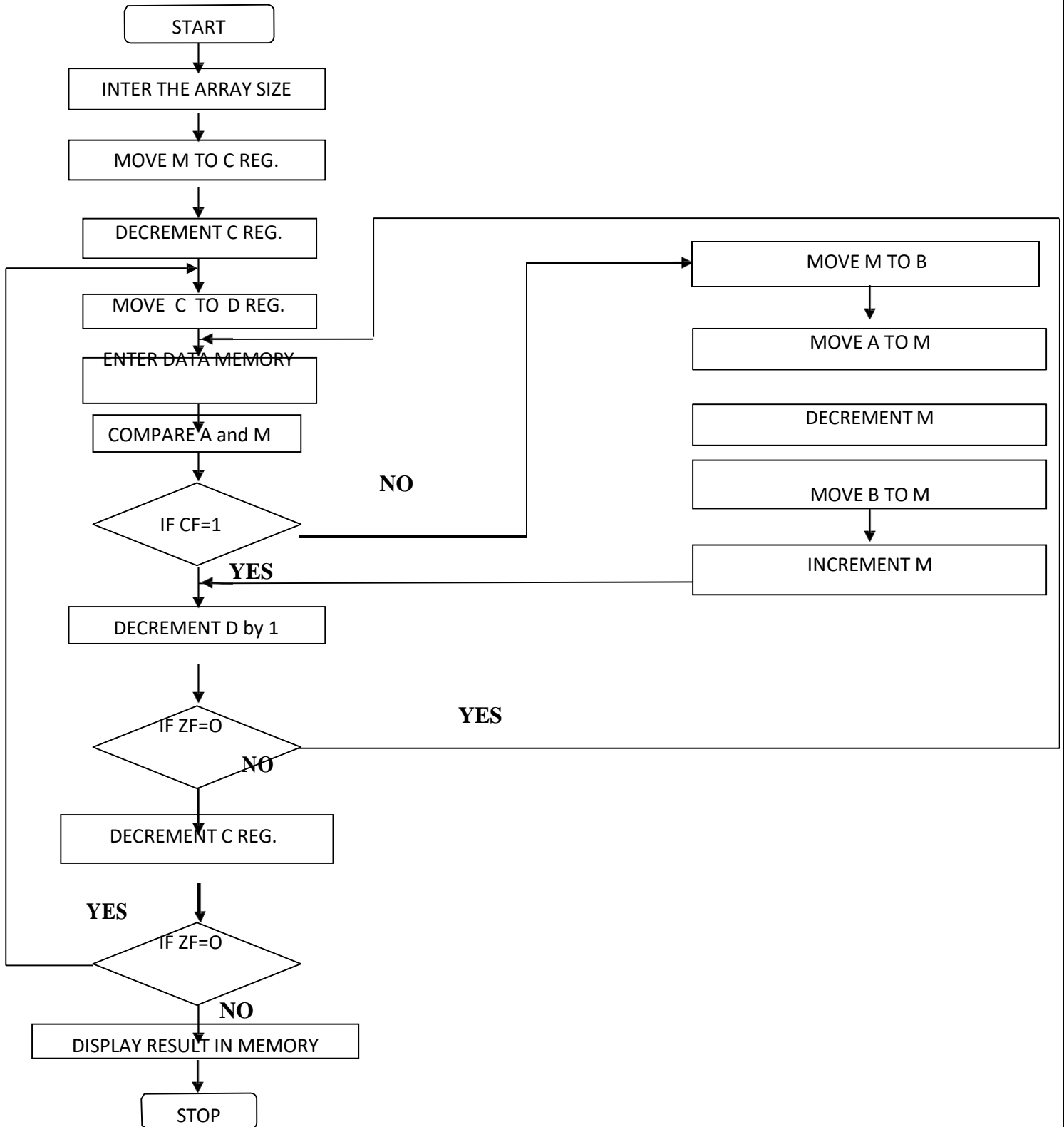
ADDRESS	LABEL	PNEUMONIC	OPCODE	COMMENTS
8100		LXI H, 8200	21	Set Pointers for array
8101			00	
8102			82	
8103		MOV B,M	46	Move the first data from (count) memory
8104		INX H	23	Increment the HL Pair
8105		MOV A,M	7E	Move the second data from memory to accumulator
8106		DCR B	05	Decrement the count

Memory Address	Input data	Memory Address	Output data
8200 (count)			
8201		8300	
8202			
8203			
8204			

8107	LOOP	INX H	23	Increment the HL pair
8108		CMP M	BE	Complements of memory with accumulator
8109		JNC AHEAD	D2	If A >M go to label AHEAD
810A			0D	
810B			81	
810C		MOV A,M	7E	Set the new values at large
810D	AHEAD	DCR B	05	Decrement the values of B
810E		JNZ LOOP	C2	Repeat the comparison till B =0
810F			07	
8110			81	
8111		STA 8300	32	Store the largest value in memory from accumulator
8112			00	
8113			83	
8114		HLT	76	Stop the program

RESULT:

FLOW CHART:



Ex. No.:**Date :** **6.A. ARRANGE AN ARRAY OF DATA IN ASCENDING ORDER****AIM:**

To write a program to arrange an array of data in ascending order by using 8085 microprocessor.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Initialize the HL pair as memory pointer
2. Move the count to C-register
3. Decrement the count
4. Copy the count in D-register
5. Load the address of the element in HL pair
6. Move the first data in A- register from memory, which is pointed by HL pair.
7. Increment the HL pointer
8. Compare the content of the memory with Accumulator
9. If they are out of order exchange the contents of A register and memory
10. Decrement D-register content by 1
11. Repeat step 9 and 10 till the value in D register becomes Zero
12. Decrement C-register content by 1
13. Repeat steps 4-12 till the value in C register becomes Zero

PROGRAM:

ADDRESS	LABEL	PNEMONICS	OPCODE	COMMENTS
8100	START	LXI H, 8200	21	Set the pointer for array
8101			00	
8102			82	
8103		MOV C, M	4E	Move the Count from Memory to C reg.
8104		DCR C	0D	Decrement the count (C reg.)
8105	REPEAT	MOV D, C	51	Move the data in C to D register
8106		LXI H, 8201	21	Load the Pointer to load next data
8107			01	
8108			82	

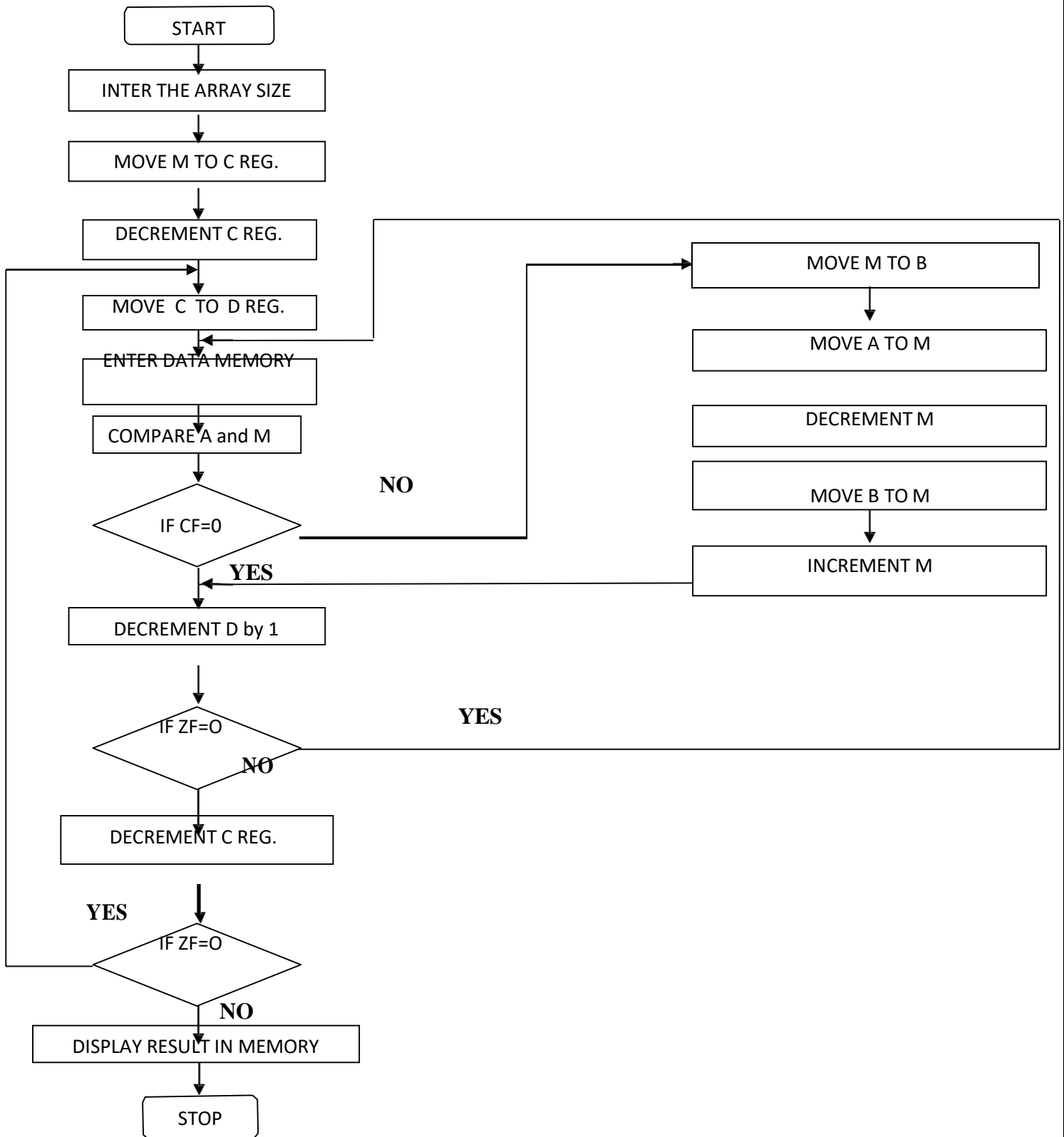
INPUT & OUTPUT TABULATION:

Memory Address	Input data	Memory Address	Output data
8200 (Count)			
8201		8201	
8202		8202	
8203		8203	
8204		8204	

8109	LOOP	MOV A ,M	7E	Set the new value of large
810A		INX H	23	Increment the HL pair.
810B		CMP M	BE	Compare the content of memory with Accumulator
810C		JC SKIP	DA	If CF=1, then go to SKIP label.
810D			14	
810E			81	Exchange the contents of A register and memory
810F		MOV B,M	46	
8110		MOV M,A	77	
8111		DCX H	2B	
8112		MOV M,B	70	
8113		INX H	23	
8114	SKIP	DCR D	15	Decrement the count(D reg.)
8115		JNZ LOOP	C2	Check for ZF, if ZF = 0 then go to LOOP label.
8116			09	
8117			81	
8118		DCR C	0D	Decrement the count
8119		JNZ REPEAT	C2	Check for ZF, if ZF = 0 then go to REPEAT label.
811A			05	
811B			81	
811C		HLT	76	Stop the program.

RESULT:

FLOW CHART:



Ex. No.:**Date :** **6.B. ARRANGE AN ARRAY OF DATA IN DESCENDING ORDER****AIM:**

To write a program to arrange an array of data in descending order by using 8085 microprocessor.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Initialize the HL pair as memory pointer
2. Move the count to C-register
3. Decrement the count
4. Copy the count in D-register
5. Load the address of the element in HL pair
6. Move the first data in A- register from memory, which is pointed by HL pair.
7. Increment the HL pointer
8. Compare the content of the memory with Accumulator
9. If they are out of order exchange the contents of A register and memory
10. Decrement D-register content by 1
11. Repeat step 9 and 10 till the value in D register becomes Zero
12. Decrement C-register content by 1
13. Repeat steps 4-12 till the value in C register becomes Zero

PROGRAM:

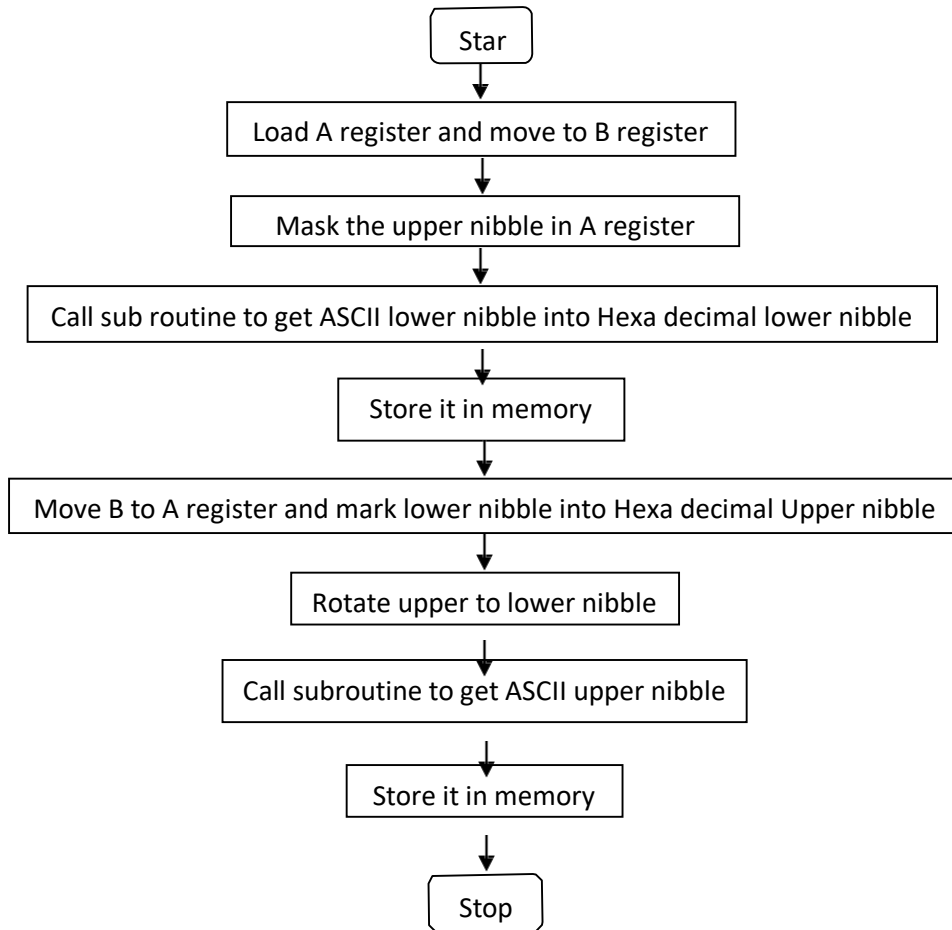
ADDRESS	LABEL	PNEMONICS	OPCODE	COMMENTS
8100	START	LXI H, 8200	21	Set the pointer for array
8101			00	
8102			82	
8103		MOV C, M	4E	Move the Count from Memory to C reg.
8104		DCR C	0D	Decrement the count (C reg.)
8105	REPEAT	MOV D, C	51	Move the data in C to D register
8106		LXI H, 8201	21	Load the Pointer to load next data
8107			01	
8108			82	

INPUT & OUTPUT TABULATION:

Memory Address	Input data	Memory Address	Output data
8200 (Count)			
8201		8201	
8202		8202	
8203		8203	
8204		8204	

8109	LOOP	MOV A ,M	7E	Set the new value of large
810A		INX H	23	Increment the HL pair.
810B		CMP M	BE	Compare the content of memory with Accumulator
810C		JNC SKIP	D2	If CF=0, then go to SKIP label.
810D			14	
810E			81	
810F		MOV B,M	46	
8110		MOV M,A	77	Exchange the contents of A register and memory
8111		DCX H	2B	
8112		MOV M,B	70	
8113		INX H	23	
8114	SKIP	DCR D	15	
8115		JNZ LOOP	C2	Check for ZF, if ZF = 0 then go to LOOP label.
8116			09	
8117			81	
8118		DCR C	0D	Decrement the count
8119		JNZ REPEAT	C2	Check for ZF, if ZF = 0 then go to REPEAT label.
811A			05	
811B			81	
811C		HLT	76	Stop the program.

RESULT:

FLOW CHART:

Ex. No.:**Date :** **7 A. CODE CONVERSION - ASCII TO HEXA****AIM:**

To write and execute the program for convert ASCII to HEXA DECIMAL number using 8085 microprocessor.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Load the given data in A register
2. Move the content of A to B register
3. Mask the upper nibble of the hexadecimal number in A register
4. Call suborning to get ASCII of lower nibble into hexadecimal lower nibble
5. Store it in memory
6. Move B register value to A- register and mask the lower nibble
7. Rotate the upper nibble to lower nibble position
8. Call subroutine to get ASCII of upper nibble in to hexadecimal
9. Store it in memory
10. Terminate the program

PROGRAM:

Address	Label	Mnemonics	Opcode	Comments
8100	START	LDA 8200	3A	Load accumulator
8101			00	
8102			82	
8103		MOV B,A	47	Move accumulator to B register
8104		ANI 0F	E6	Mask the Upper nibble
8105			0F	
8106		CALL SUB1	CD	Call suborning to get ASCII of lower nibble
8107			1A	
8108			81	
8109		STA 8201	32	Store ASCII of lower nibble in memory
810A			01	
810B			82	
810C		MOV A,B	78	Move B register to accumulator

Conversion Table for Hexadecimal, Decimal and ASCII

Hexa	Decimal	ASCII
30	48	0
31	49	1
32	50	2
33	51	3
34	52	4
35	53	5
36	54	6
37	55	7
38	56	8
39	57	9
41	65	A
42	66	B
43	67	C
44	68	A
45	69	B
46	70	C
47	71	A
48	72	B

Hexa	Decimal	ASCII
49	73	C
50	74	A
51	75	B
52	76	C
53	77	A
54	78	B
55	79	C
56	80	A
57	81	B
58	82	C
59	83	A
60	84	B
61	85	C
62	86	A
63	87	B
64	88	C
65	89	A
5A	90	B

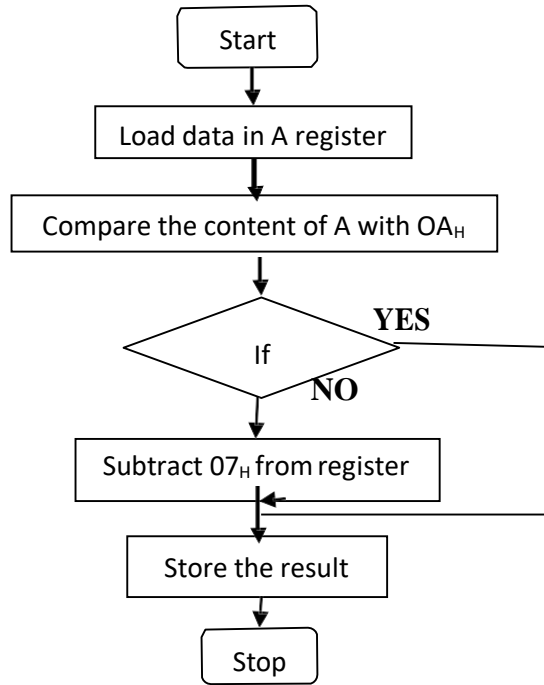
INPUT & OUTPUT TABULATION:

Memory address	Input data	Memory address	Output data
8200		8201	
		8202	

810D		ANI F0	E6	Mask the lower nibble
810E			F0	
810F		RLC	07	Rotate left through Carry
8110		RLC	07	
8111		RLC	07	
8112		RLC	07	
8113		CALL SUB1	CD	Call suborning to get ASCII of Upper nibble
8114			1A	
8115			81	
8116		STA 8202	32	Store ASCII of Upper nibble in memory
8117			02	
8118			82	
8119		HLT	76	Stop the program
811A		CPI 0A	FE	Compare A with immediate data
811B			0A	
811C		JC SKIP	DA	Jump on carry to SKIP label
811D			21	
811E			81	
811F		ADI 07	C6	Count the number , add accumulator with 07
8120			07	
8121	SKIP	ADI 30	C6	Add accumulator with immediate data
8122			30	
8123		RET	C9	Return to Main program

RESULT:

FLOW CHART



Ex. No.:**Date :** **7 B. CODE CONVERSION - HEXA TO ASCII****AIM:**

To convert given character (HEXA) in to its equivalent ASCII using 8085 microprocessor

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Load the given data in A-register
2. Subtract 30_H from A- register
3. Compose the content of A-register with 0A_H
4. If A<0A_H jump to step6, else proceed To next step
5. Subtract 07_H from A-register
6. Store the result
7. Stop the program

PROGRAM:

Address	Label	Mnemonics	Opcode	comments
8100	START	LDA 8200	3A	Load the input data into the accumulator
8101			00	
8102			82	
8103		SUI 30	D6	Subtract accumulator with immediate data
8104			30	
8105		CPI 0A	FE	Compare A with immediate data.
8106			0A	
8107		JC SKIP	DA	Jump on carry to SKIP label
8108			0C	
8109			81	
810A		SUI 07	D6	Subtract accumulator with 07
810B			07	

Conversion Table for Hexadecimal, Decimal and ASCII

Hexa	Decimal	ASCII
30	48	0
31	49	1
32	50	2
33	51	3
34	52	4
35	53	5
36	54	6
37	55	7
38	56	8
39	57	9
41	65	A
42	66	B
43	67	C
44	68	A
45	69	B
46	70	C
47	71	A
48	72	B

Hexa	Decimal	ASCII
49	73	C
50	74	A
51	75	B
52	76	C
53	77	A
54	78	B
55	79	C
56	80	A
57	81	B
58	82	C
59	83	A
60	84	B
61	85	C
62	86	A
63	87	B
64	88	C
65	89	A
5A	90	B

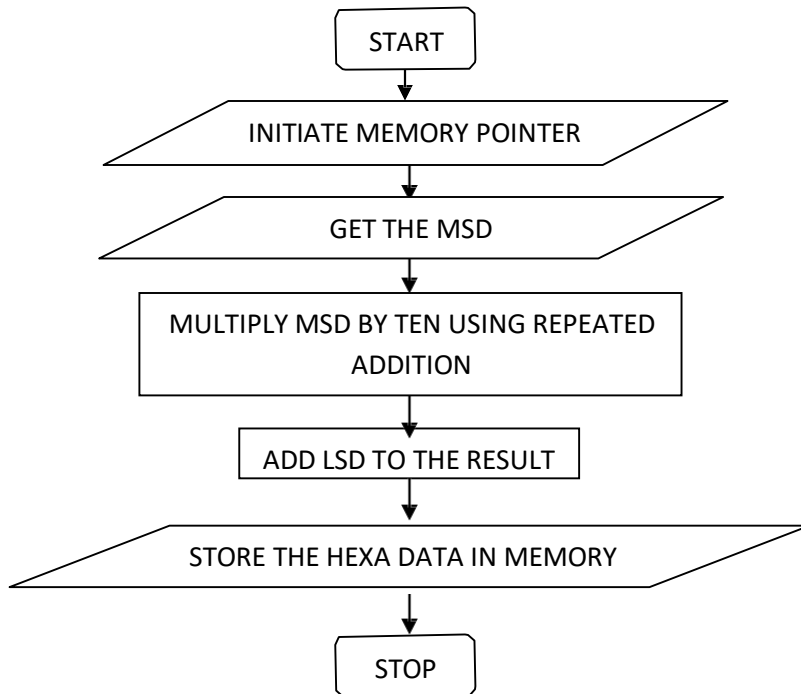
INPUT & OUTPUT TABULATION:

Memory Address	Input data	Memory Address	Output data
8200		8201	

810C	SKIP	STA 8201	32	Store the result in memory from accumulator
810D			01	
810E			82	
810F		HLT	76	Stop the Program

RESULT:

FLOW CHART:



Ex. No.:**Date :** **8 A. CODE CONVERSION - BCD TO HEXA****AIM:**

To convert two BCD numbers in memory to its equivalent HEXA number using 8085 microprocessor.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Initialize memory pointer to 8100H.
2. Load the most significant digit (MSD).
3. Multiply the MSD by ten using repeated addition.
4. Add the least significant digit (LSD) to the result obtained in previous step.
5. Store the HEXA data in memory.

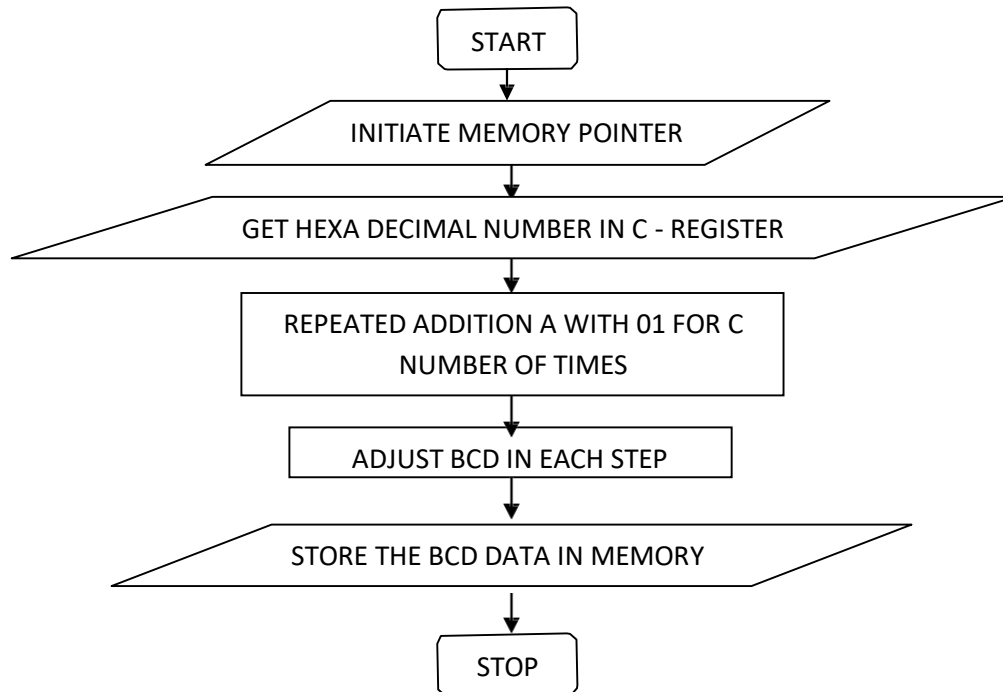
PROGRAM:

Address	Label	Pneumonic	Opcode	Comments
8100	START	LXI H,8150	21	Load the input data into the accumulator
8101			50	
8102			81	
8103		MOVA,M	7E	Move memory to accumulator
8104		ADD A	87	Add accumulator content with Accumulator. Ie) MSD*2
8105		MOV B,A	47	Move Accumulator content to B
8106		ADD A	87	MSD is multiplied by 4
8107		ADD A	87	MSD is multiplied by 8
8108		ADD B	80	Add accumulator content with B reg.
8109		INX H	23	Increment the memory
810A		ADD M	86	Add Accumulator and memory
810B		INX H	23	Increment the memory
810C		MOV M,A	77	Move the accumulator to memory for result
810D		HLT	76	Stop the program

INPUT & OUTPUT TABULATION:

Memory Address	Input data	Memory Address	Output data
8150		8152	
8151			

RESULT:

FLOW CHART:

Ex. No.:**Date :** **8 B. CODE CONVERSION - HEXA TO BCD****AIM:**

To convert given HEXA decimal number into its equivalent BCD number using 8085 microprocessor.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet

ALGORITHM:

1. Initialize memory pointer to 8100H
2. Get the hexadecimal number in C register
3. Perform repeated addition for C number of times
4. Adjust for BCD in each step
5. Store the BCD data in memory

PROGRAM:

Address	Label	Pneumonic	Opcode	Comments
8100	START	LXI H, 8150	21	Initialize memory pointer for input
8101			50	
8102			81	
8103		MVI D,00	16	Clear D register for most significant byte
8104			00	
8105		XRA A	AF	Clear accumulator
8106		MOV C,M	4E	Get Hexadecimal input data from memory
8107	LOOP2	ADI 01	C6	Count the number One by one adjust BCD count
8108			01	
8109		DAA	27	Adjust accumulator for BCD
810A		JNC LOOP1	D2	Jump on no carry to Loop1
810B			0E	
810C			81	
810D		INR D	14	Increment D register
810E	LOOP1	DCR C	0D	Decrement C register

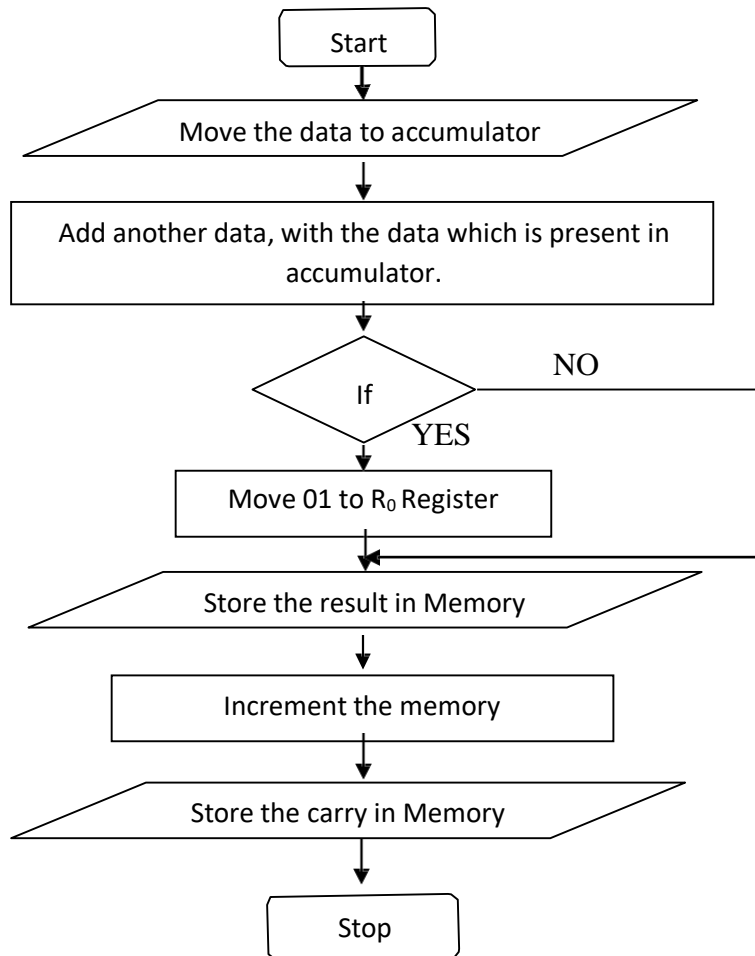
INPUT & OUTPUT TABULATION:

Memory Address	Input data	Memory Address	Output data
8150		8151	
		8152	

810F		JNZ LOOP2	C2	Jump on no zero to Loop2
8110			07	
8111			81	
8112		STA 8151	32	Store the least Significant byte in memory
8113			51	
8114			81	
8115		MOV A,D	7A	Move D to accumulator
8116		STA 8152	32	Store the most Significant byte in memory
8117			52	
8118			81	
8119		HLT	76	Termite ate the program

RESULT:

8051 Microcontroller Programs

FLOW CHART:

Ex. No.:**Date :** **9 A. ADDITION OF TWO 8-BIT DATA****AIM:**

To perform the arithmetic operation addition by using 8051 microcontroller.

APPARATUS REQUIRED:

- 8051 microcontroller kit
- Opcode sheet

ALGORITHM:

1. Start the program
2. Get the Input data at the accumulator.
3. Add the adder data with the data which is already in accumulator.
4. Move the result to 8500 memory location.
5. If any carry is available then move 01 to R₀ Register
6. Store the result in Memory
7. Stop the program

PROGRAM:

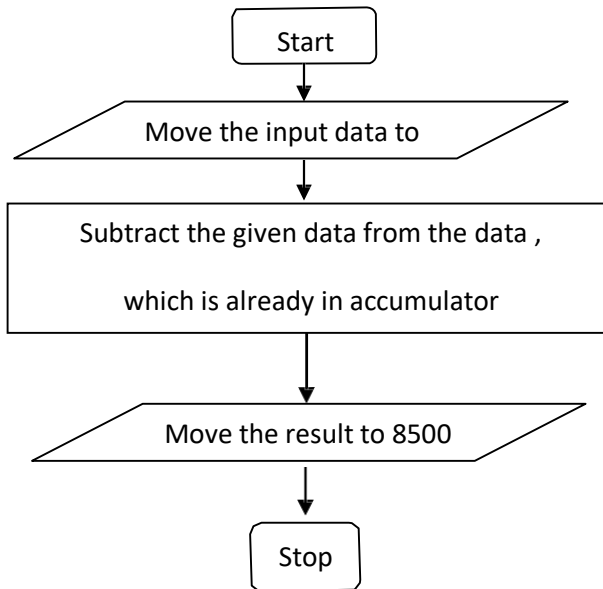
Address	Label	Pneumonic	Opcode	Comments
8100	START	MOVA,09	74	Move 09 to accumulator
8101			09	
8102		ADD A,04	24	Add 04 with accumulator
8103			04	
8104		JNC LOOP	50	On No carry Jump to 100p
8105			02	
8106		MOV R ₀ , 01	78	Move R ₀ register into 01 for carry
8107			01	

INPUT & OUTPUT TABULATION:

Memory Address	Input data	Memory address	Output data
8101		8500	
8103			
8101		8500	
8103			

8108	LOOP	MOV DPTP,#8500	90	Move the memory address to DPTR
8109			85	
810A			00	
810B		MOVX @DPTR , A	FO	Store the sum in memory
810C		INC DPTR	A3	Increment DPTR
810D		MOV A,R ₀	E8	Move R ₀ to A
810E		MOVX @DPTR , A	FO	Store the Carry in memory
810F	LOOP1	SJMP LOOP1	80	Stop the program
8110			FE	

RESULT:

FLOW CHART:**INPUT & OUTPUT TABULATION:**

Memory Address	Input data	Memory address	Output data
8101		8500	
8103			
8101		8500	
8103			

Ex. No.:**Date :** **9 B. SUBTRACTION OF TWO 8-BIT DATA****AIM:**

To subtract the two given number by using 8051 micro controller.

APPARATUS REQUIRED:

- 8051 microcontroller kit
- Opcode sheet

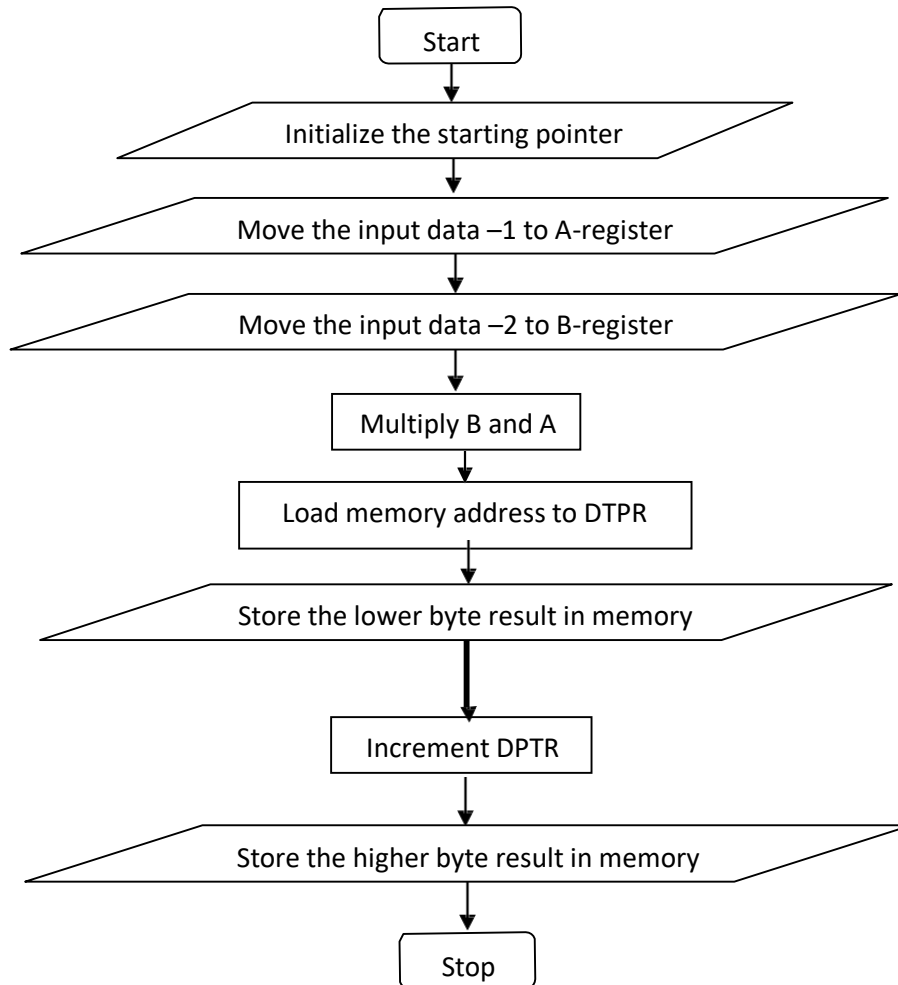
ALGORITHM:

1. Start the program.
2. Get data to the accumulator.
3. Subtract another data from the data which is already stored in the accumulator.
4. Move the result to the memory 8500.
5. If any carry, than store it in accumulator.
6. Stop the program.

PROGRAM:

Address	Label	Pneumonic	Opcode	Comments
8100	START	MOV A ,09	74	Move the data 09 to the accumulator
8101			09	
8102		SUBB A ,04	94	Subtract the data 04 with data in accumulator
8103			04	
8104		MOV DPTR, #8500	90	store the result in 8500
8105			85	
8106			00	
8107		MOV @DPTR , A	FO	Carry is stored
8108	LOOP	STMP LOOP	80	Short jump
8109			FE	Stop the program

RESULT:

FLOW CHART:**INPUT& OUTPUT TABULATION:**

MEMORY ADDRESS	INPUT DATA	MEMORY ADDRESS	OUTPUT DATA
8101		8500	
8104		8501	
8101		8500	
8104		8501	

Ex. No.:**Date :** **10 A. MULTIPLICATION OF TWO 8-BIT DATA****AIM:**

To perform 8-bit multiplication by using 8051 microcontroller

APPARATUS REQUIRED:

- 8051 microcontroller kit
- Opcode sheet

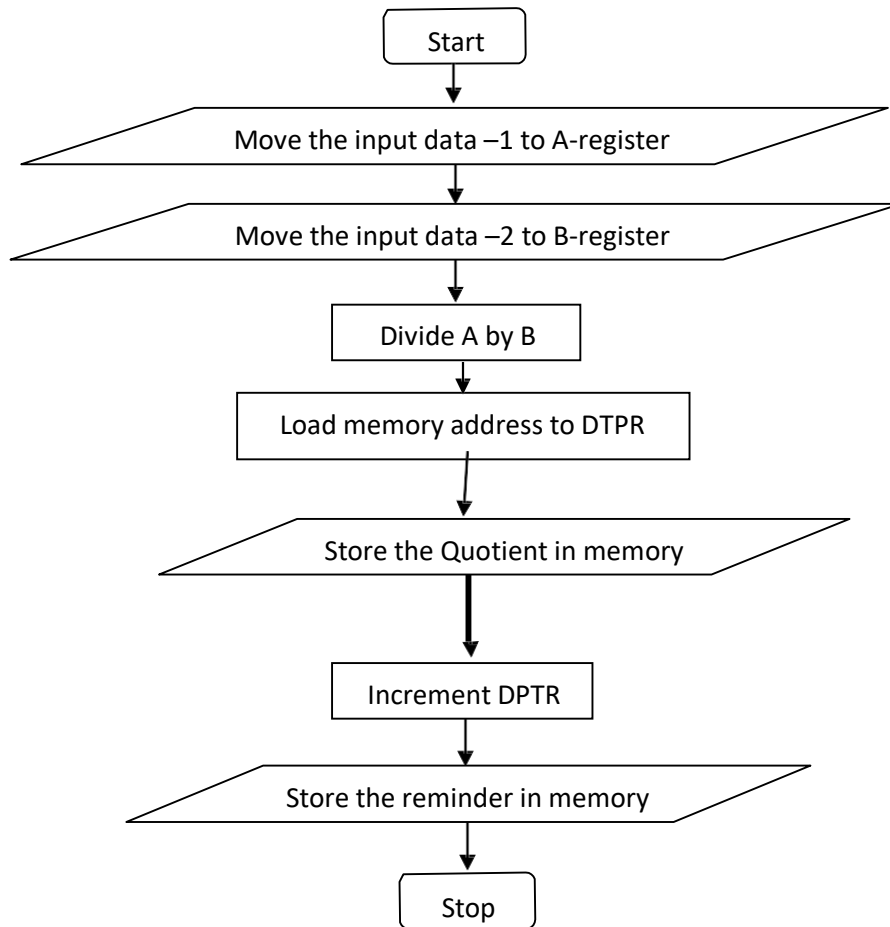
ALGORITHM:

1. Start the program.
2. Initialize the starting pointer.
3. Move the input data 1 to accumulator
4. Move the input data 2 to the B-register
5. Multiply both B and A
6. Load the memory address to DPTR and store the lower byte result.
7. Increment DPTR and move B to A.
8. Store the higher byte result
9. Stop the program

PROGRAM:

ADDRESS	LABEL	PNEUMONIC	OPCODE	COMMENTS
8100	START	MOV A , #02	74	Move the input data 1 to accumulator
8101			02	
8102		MOV B , #03	75	Move the input data 2 to the B-register
8103			F0	
8104			03	
8105		MUL A,B	A4	Multiply the input data's ie) A and B
8106		MOV DPTR , #8500	90	Load the memory address to DPTR
8107			85	
8108			00	
8109		MOVX @DPTR , A	F0	store the lower byte result
810 A		INC DPTR	A3	Increment DPTR
810 B		MOV A,B	F5	Move B to A register
810 C			F0	
810 D		MOVX @DPTR , A	F0	Store the higher byte result
810 E	LOOP	SJMP LOOP	80	Stop the program
810F			FE	

RESULT:

FLOW CHART:**INPUT & OUTPUT TABULATION:**

MEMORY ADDRESS	INPUT DATA	MEMORY ADDRESS	OUTPUT DATA
8101		8500	
8104		8501	
8101		8500	
8104		8501	

Ex. No.:**Date :** **10 B. DIVISION OF TWO 8-BIT DATA****AIM:**

To performs the 8-bit division using 8051 microcontroller

APPARATUS REQUIRED:

- 8051 microcontroller kit
- Opcode sheet

ALGORITHM:

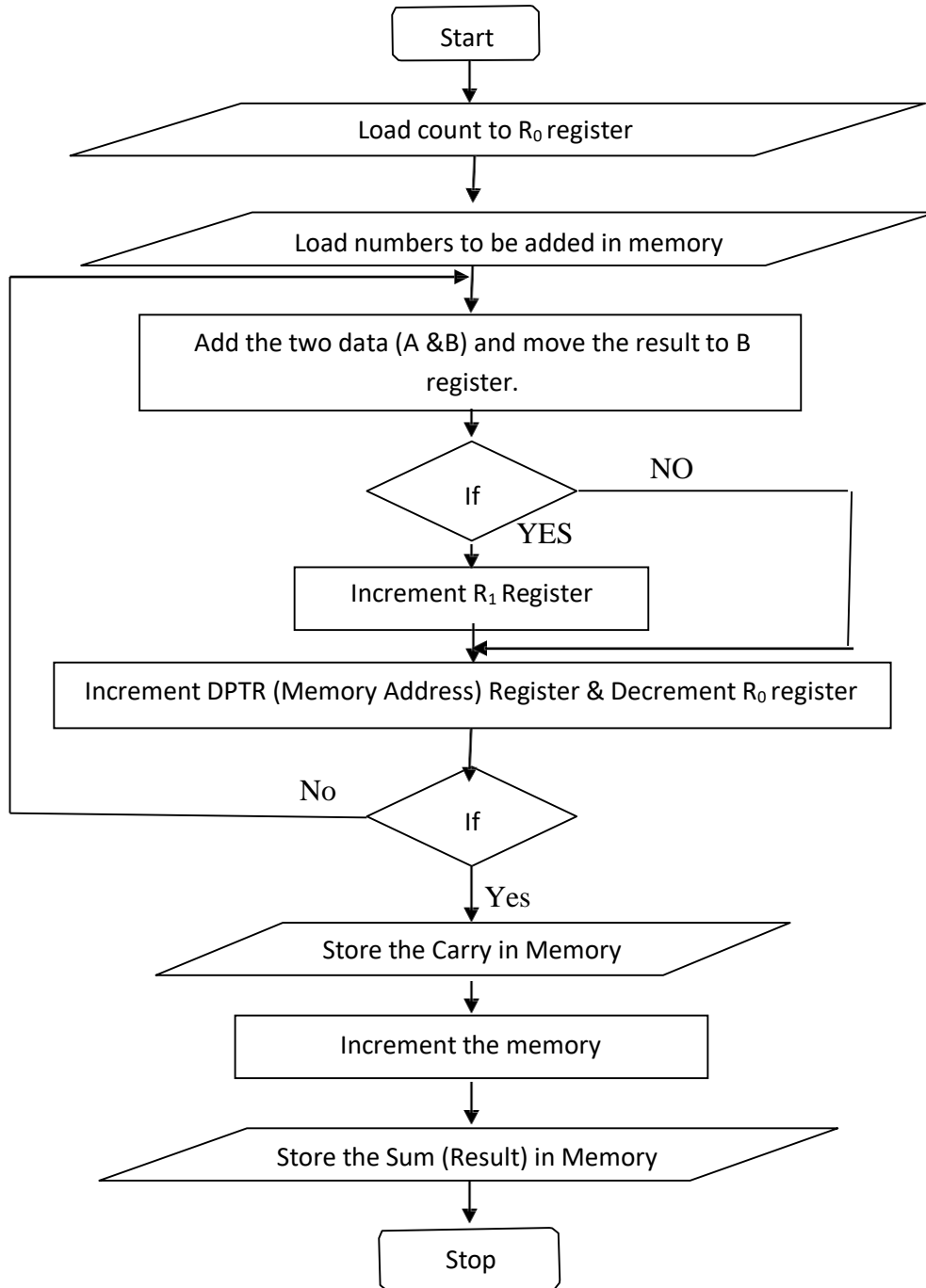
1. Start the program.
2. Initialize the starting pointer.
3. Move the input data 1 to accumulator
4. Move the input data 2 to the B-register
5. Divide the content of A by B
6. Load the memory address to DPTR and store the Quotient in memory.
7. Increment DPTR and move B to A.
8. Store the Reminder in memory.
9. Stop the program

DIVISION ON USING 8051 PROGRAM:

ADDRESS	LABLE	PREMONICS	OPCODE	COMMENTS
8100	START	MOV A,#05	74	Move the input data 1 to accumulator
8101			05	
8102		MOV B, #03	75	Move the input data 2 to the B-register
8103			FO	
8104			03	
8105		DIV A,B	84	Divide the content of A,B
8106		MOV DPTR,8150	90	Load the memory address to DPTR
8107			81	
8108			50	
8109		MOVX @DPTR , A	FO	
810A		INC DPTR	A3	Increment DPTR
810B		MOV A , B	E5	Move B to A register
810C			F0	
810D		MOV X @DPTR , A	F0	Store the reminder in memory
810E	LOOP	SJMP LOOP	80	Stop the program
810F			FE	

RESULT:

FLOW CHART:



Ex. No.:**Date :** **11 A. SUM OF THE ELEMENTS****AIM:**

To perform the sum of the numbers by using 8051 microcontroller.

APPARATUS REQUIRED:

- 8051 microcontroller kit
- Opcode sheet

ALGORITHM:

1. Start the program
2. Get the Count(R₀) & Input data in memory.
3. Add the two data and move the result to B register..
4. If Carry is present , increment R₁Register, otherwise go to next step.
5. Increment DPTR register for next data.
6. Decrement the R₀ register for count.
7. If Zero flag is not set go to step 3, otherwise go to next step.
8. Store the result in Memory
7. Stop the program

PROGRAM:

Address	Label	Pneumonic	Opcode	Comments
8100		MOV DPTR,#8200	90	Move Memory address to DPTR
8101			82	
8102			00	
8103		MOVX A, @DPTR	E0	Move the first data to acc and then R ₀ for count
8104		MOV R ₀ ,A	F8	

INPUT & OUTPUT TABULATION:

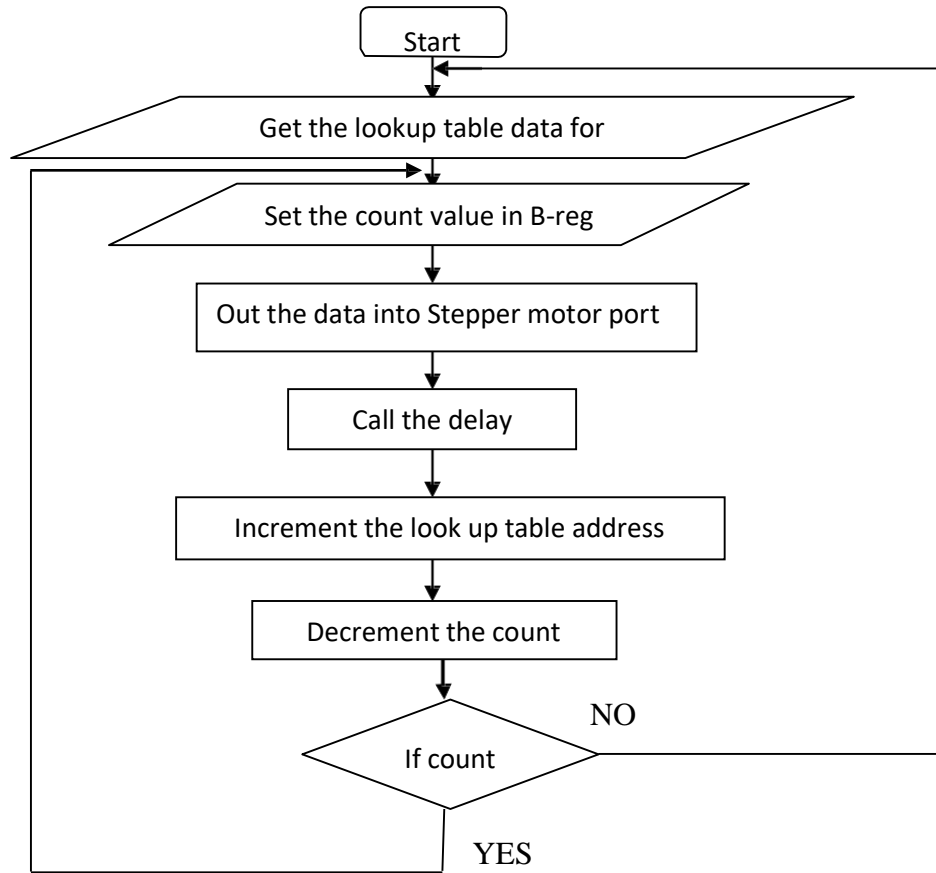
Memory Address	Input data	Memory address	Output data
8200		8500	
8201			
8202		8501	
8203			
8204			
8205			

8105		MOV B, #00	75	Clear B register
8106			F0	
8107			00	
8108		MOV R ₁ , B	A9	Move B register content to R ₁
8109			F0	
810A		INC DPTR	A3	Increment Memory address
810B	LOOP1	MOVX A, @DPTR	E0	Move the data from memory to Acc.
810C		ADD A,B	25	Add A & B registers contents.
810D			F0	
810E		MOV B,A	F5	Move the result from A to b register.
810F			F0	
8110		JNC LOOP	50	Jump no carry then LOOP label
8111			01	
8112		INC R ₁	09	Increment R ₁ for carry
8113	LOOP	INC DPTR	A3	Increment Memory address
8114		DJNZ R ₀ , LOOP1	D8	Decrement R ₀ (Count) value and if R ₀ ≠ 0 then jump to LOOP1 label.
8115			F5	
8116		MOV DPTP,#8500	90	Move the memory address to DPTR for Result
8117			85	
8118			00	
8119		MOV A,R ₁	E9	Move R ₁ to A
811A		MOVX @DPTR , A	F0	Store the Carry in memory
811B		INC DPTR	A3	Increment DPTR
811C		MOV A,B	E5	Move B to A
811D			F0	
811E		MOVX @DPTR , A	F0	Store the Sum in memory
811F	LOOP1	SJMP LOOP1	80	Stop the program
8120			FE	

RESULT:

INTERFACING PROGRAMS

FLOW CHART:



Ex. No.:**Date :** **11.B. STEPPER MOTOR INTERFACE USING 8051
MICROCONTROLLER****AIM:**

To run stepper a stopper motor at despises speed in two directions using 8051 microcontroller.

APPARATUS REQUIRED:

- 8051 Microcontroller kit
- Opcode sheet
- Sp stepper motor interface

THEORY:

A motor in which the rotor is able to assume only discrete stationery angular position is a stepper motor. The rotary motion occurs in a stepwise manner from an equilibrium position to the next. Stepper motor are widely used in (simple position control systems in the open closed loop mode)a verity of application such as complete peripherals (printers, disk drive etc)and in the areas of process control machine tools, medicine numerically controller machine robotics.

ALGORITHM:

1. Load the stepping sequence number
2. Then load the motor port addressing 8015 memory
3. Move stepping data into accumulator
4. Out the accumulator value in to the stepper motor
5. Call the delay
6. Increment the DPTR (memory address)
7. Repeat the processor for all stepping data
8. Jump to step 1and repeat all steps

PROGRAM:

Address	Label	Pneumonic	Opcode	comments
8100	START	MOV B, #04	75	Move total no of stepping data to B register
8101			F0	
8102			04	
8103		MOV R ₀ , #82	78	Move starting address of stepping sentence to R ₀ ,R ₁
8104			82	
8105		MOV R ₁ , #00	79	
8106			00	

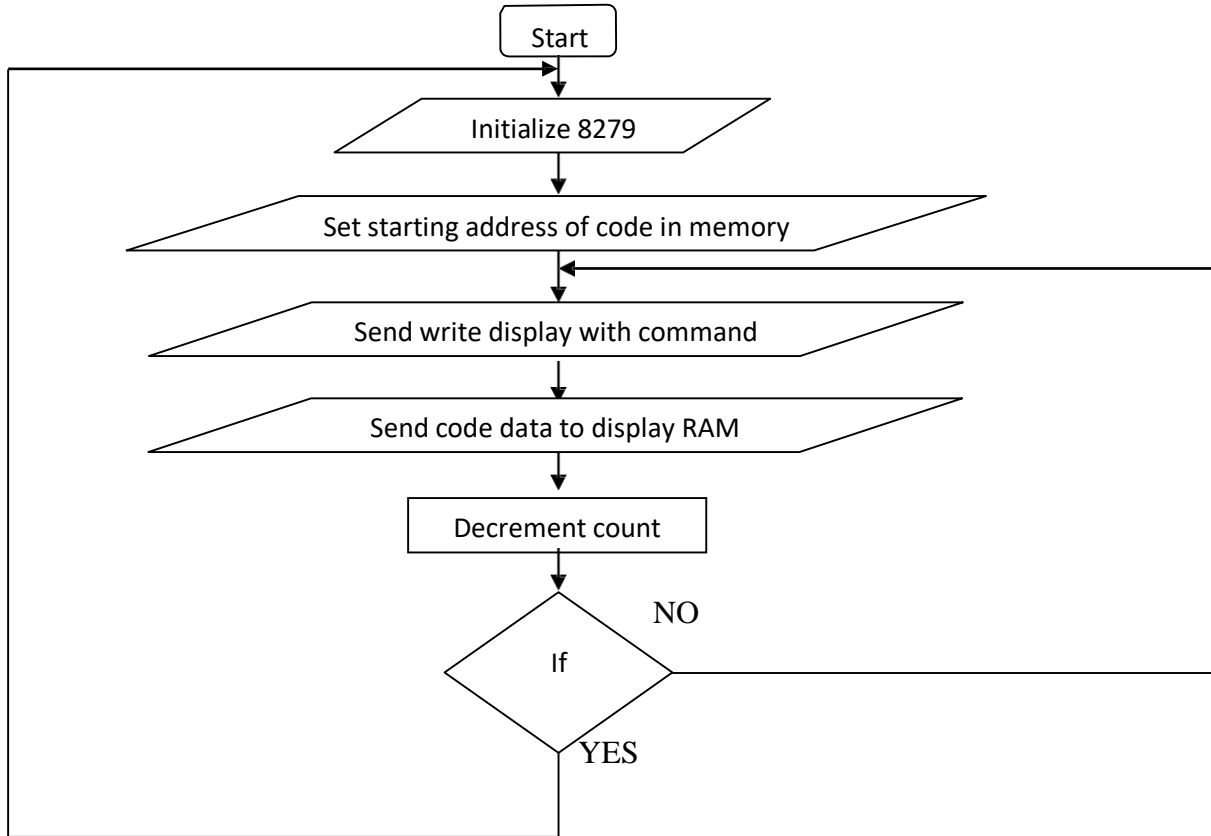
WAVE SCHEME (UNIPOLAR OPERATION)

ADDRESS	ANTI CLOCKWISE	CLOCKWISE
8200	08	02
8201	01	04
8202	04	01
8203	02	08

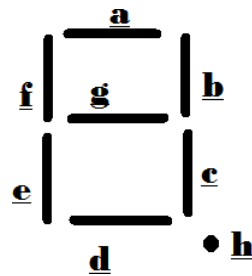
8107		MOV DPTR, #E0C0	90	Motor port address in DPTR
8108			E0	
8109			C0	
810A	LOOP	MOV DP _H , R ₀	88	Save data R ₀ ,R ₁ in data
810B			83	
810C		MOV DP _L , R ₁	89	
810D			82	
810E		MOV A , @DPTR	E0	Move stepping data to accumulator
810F		INC DPTR	A3	Increment DPTR
8100		MOV R ₀ , DP _H	A8	Save data R ₀ ,R ₁ in DPTR
8111			83	
8112		MOV R ₁ , DP _L	A9	
8113			82	
8114		MOV DPTR ,#E0C0	90	Motor port address in DPTR
8115			E0	
8116			C0	
8117		MOV X @DPTR , A	F0	Move data in accumulator to DPTR
8118		CALL DELAY	12	Call delay routine
8119			81	
811 A			21	
811B		DJNZ B , LOOP	D5	
811C			F0	Decrement B and jump to loop (810A)if B#0
811D			EC	
811E		JMP START	02	Jump to start (8100)
811F			81	
8120			00	
8121	DELAY	MOV R ₂ , #12	7A	
8122			12	
8123	DLY 1	MOV R ₃ , #FF	7	Move data to register
8124			FF	
8125	DLY 2	DJNZ R ₃ ,DLY2	DB	Decrement R ₃ ,and to DY 218125 if R ₃ ,#0
8126			FE	
8127		DJNZ R ₂ ,DLY1	DA	Decrement R AND jump to DLYLL 8123 Y R ₂ #0
8128			FA	
8129		RET	22	Return to main program

RESULT:

FLOW CHART:



SEVEN SEGMENT DISPLAY



For Example

	d	c	b	a	h	g	f	e	
	0	1	1	0	1	0	0	0	-- 68H

Ex. No.:

Date : **12. INTERFACING 8279 WITH 8085 MICROPROCESSOR**
(ROLLING DISPLAY)

AIM:

To interface 8279 programmable keyboard display controller with 8085 microprocessor and write and execute the assembly language program to roll the word to display.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- 8279 keyboard display
- Opcode sheet

ALGORITHM:

1. Start the program by initializing memory pointer
2. Initialize 8279 keyboard display controller
3. Set mode and display in 8279 IC
4. Clear display in 8279 keyboard display controller
5. Write display & Read FIFO status
6. Write display RAM from location auto-increases of mode
7. Move data input from Memory to Accumulator
8. Send code data to display Ram and call delay subroutine
9. Decrement the counter and repeat the steps from 5 to 9 until the counter becomes zero.
10. Stop the program.

PROGRAM:

ADDRESS	LABEL	PNEUMONIC	OPCODE	COMMENT
8100	START	LXI H, 8150	21	Set pointer to memory
8101			50	
8102			81	
8103		MVI D, 1C	16	Initialize counter in D register
8104			1C	
8105		MVI A,10	3E	Set mode and Display in 8279 IC
8106			10	
8107		OUT C2	D3	Clear the display
8108			C2	

INPUT & OUTPUT TABULATION:**INPUT:**

INPUT ADDRESS	INPUT DATA
8150	
8151	
8152	
8153	
8154	
8155	
8156	
8157	
8158	
8159	
815A	
815B	
815C	
815D	
815E	
815F	

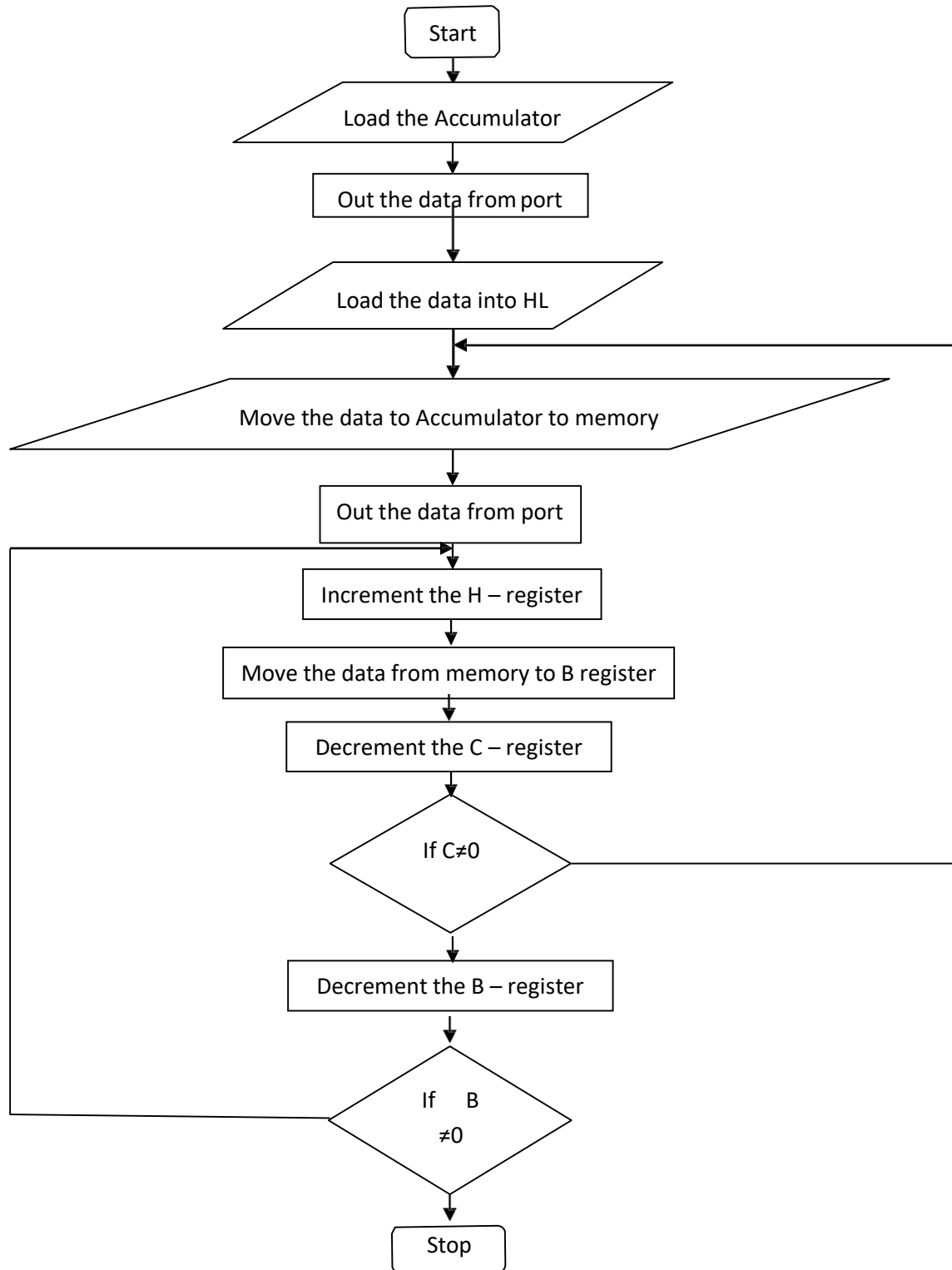
OUTPUT

8109		MVI A, CC	3E	
810A			CC	
810B		OUT C2	D3	
810C			C2	
810D		MVI A, 90	3E	Write display
810E			90	
810F		OUT C2	D3	
8110			C2	
8111	LOOP	MOV A,M	7E	
8112		OUT C0	D3	
8113			C0	
8114		CALL DELAY	CD	Call delay Subroutine
8115			1F	

8116			81	
8117		INX H	23	Increment the memory pointer
8118		DCR D	15	Decrement counter
8119		JNZ LOOP	C2	Jump if no zero to loop
811A			11	
811B			81	
811C		JMP START	C3	
811D			00	For Rolling the Display
811E			81	
811F	DELAY	MVI B, A0	06	
8120			A0	
8121	LOOP1	MVI C,FF	0E	Delay Subroutine
8122			FF	
8123	LOOP2	DCR C	0D	
8124		JNZ LOOP2	C2	
8125			23	
8126			81	
8127		DCR B	05	
8128		JNZ LOOP1	C2	
8129			21	
812A			81	
812B		RET	C9	

RESULT:

FLOW CHART:



Ex. No.:

Date : **13. TRAFFIC LIGHT CONTROL SYSTEM USING
8085 MICROPROCESSOR**

AIM:

To perform the traffic light controlling using 8085 microprocessor.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet
- Traffic light control interface board.

ALGORITHM:

1. Start the program
2. Move the data to accumulator
3. Output data from port
4. Load HL register pair with immediate data
5. Move the data content to C register
6. Move data from memory to A
7. Output data from part A
8. Increment the HL register pair & Move data from memory to A
9. Output data from port B and port C one by one
10. Move data from memory to B register
11. Call delay for some time and increment HL register
12. Decrements C registers if ZF is not set go to step 6 else repeat the whole process
13. Perform OR operation with A and D register
14. If ZF=0, call delay else decrement B register until ZF=1.
15. Stop the program

**TRAFFIC LIGHT CONTROL SYSTEM THE TRAFFIC LIGHT CONTROLLER
WORKS IN FOLLOWING SEQUENCE:**

- ❖ Provide green signal for road 1, green signal for pedestrian on road 4, red signal for other roads and other pedestrian 6secs
- ❖ Put Yellow signal for road 1, and maintain other signals in the previous state for 3 sacs.
- ❖ Provide green signal for road 2, green signal for pedestrian on road 1, red signal for other roads and other pedestrians for 6 sacs.
- ❖ Put yellow signal for road 2, and maintain other signals in the previous state for 3secs
- ❖ Provide green signal for road 3, green signal for pedestrian on road 2, red signal for other roads and other pedestrians for 6secs.

ROAD 1:

Colour	Indication	Port lines
Bi colour (Red)	Pedestrian stop	PA0
Bi colour (green)	Pedestrian Go	PA1
Green 2	Go Right	PA2
Red	Stop	PA3
Yellow	Before stop	PA4
Green 1	Go straight	PA5

ROAD 2:

Colour	Indication	Port lines
Bi colour (Red)	Pedestrian stop	PA6
Bi colour (green)	Pedestrian Go	PA7
Green 2	Go Right	PB0
Red	Stop	PB1
Yellow	Before stop	PB2
Green 1	Go straight	PB3

ROAD 3:

Colour	Indication	Port lines
Bi colour (Red)	Pedestrian stop	PB4
Bi colour (green)	Pedestrian Go	PB5
Green 2	Go Right	PB6
Red	Stop	PB7
Yellow	Before stop	PC0
Green 1	Go straight	PC1

ROAD 4:

Colour	Indication	Port lines
Bi colour (Red)	Pedestrian stop	PC2
Bi colour (green)	Pedestrian Go	PC3
Green 2	Go Right	PC4
Red	Stop	PC5
Yellow	Before stop	PC6
Green 1	Go straight	PC7

- ❖ Put yellow signal for road 3, and maintain other signals in the previous state for 3 sacs
- ❖ Provide green signal for road 4, green signal for pedestrian on road 3, red signal for other roads and other pedestrians for 6 sacs.
- ❖ Put yellow signal for road 4, and maintain other signals in the previous state for 3 sacs.
- ❖ Stop the process.

PROGRAM:

Address	Label	Mnemonics	Opcode	Comments
8100		MVI A, 80	3E	Move immediate data to accumulator
8101			80	
8102		OUT PCNT	D3	Out the data from port
8103			1B	
8104	START	LXI H,8150	21	Set pointer to the register pair
8105			50	
8106			81	
8107		MVI C,08	0E	Move immediate data to C - register
8108			08	
8109	LOOP1	MOV A,M	7E	Move data from Memory from accumulator
810A		OUT PA	D3	Out the data from port A
810B			18	
810C		INX H	23	Increment HL pair
810D		MOV A,M	7E	Move data from M to A
810E		OUT PB	D3	Out the data from port B
810F			19	
8110		INX H	23	Increment HI pair
8111		MOV B,M	46	Move content of M to B
8112		OUT PC	D3	Out the data from port C
8113			19	
8114		INX H	23	Increment HI Pair
8115		MOV B,M	46	Move the content M to B

For 6 seconds

Green signal on Road 1

Green signal for pedestrian stop on Road 4

Red signal for other Road

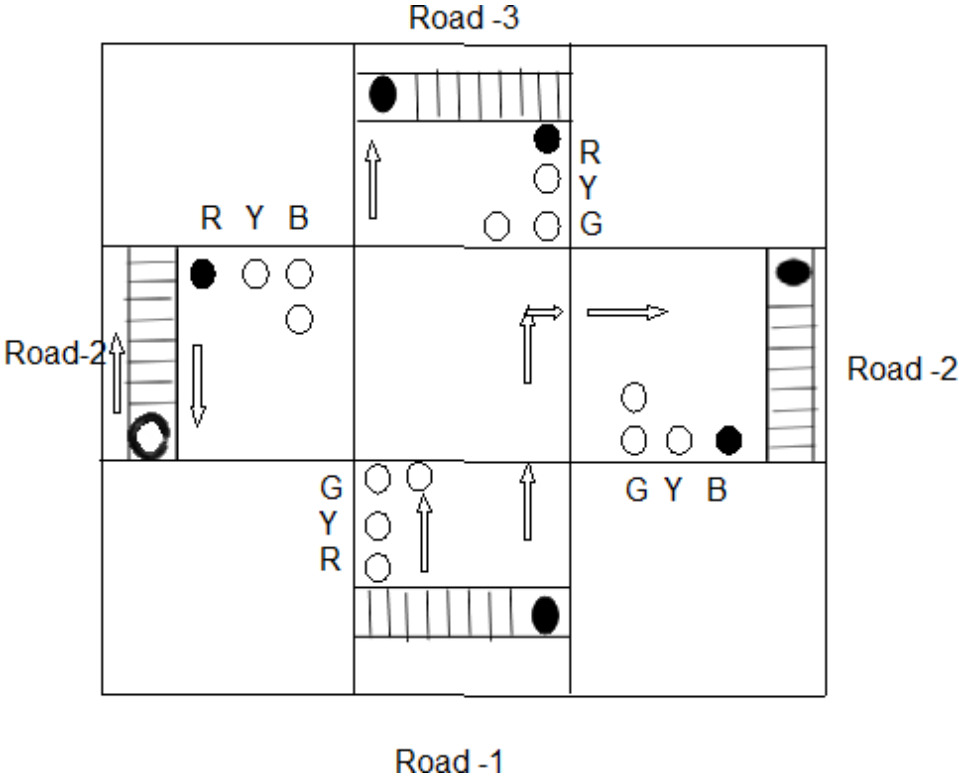
PA ₇	PA ₆	PA ₅	PA ₄	PA ₃	PA ₂	PA ₁	PA ₀	Hoax value	decimal
0	1	1	0	0	1	0	1	65 (PA)	
1	0	0	1	0	0	1	0	92(PB)	
0	0	1	0	1	0	0	0	28(PC)	
0	0	0	0	0	1	1	0	06 (time delay)	

INPUT DATA

MEMORY ADDRESS	INPUT DATA
8150	65
8151	92
8152	28
8153	06
8154	51
8155	92
8159	28
815A	03
815B	4A
815C	99
815D	24
815E	06
815F	4A
8160	94
8161	24
8162	03

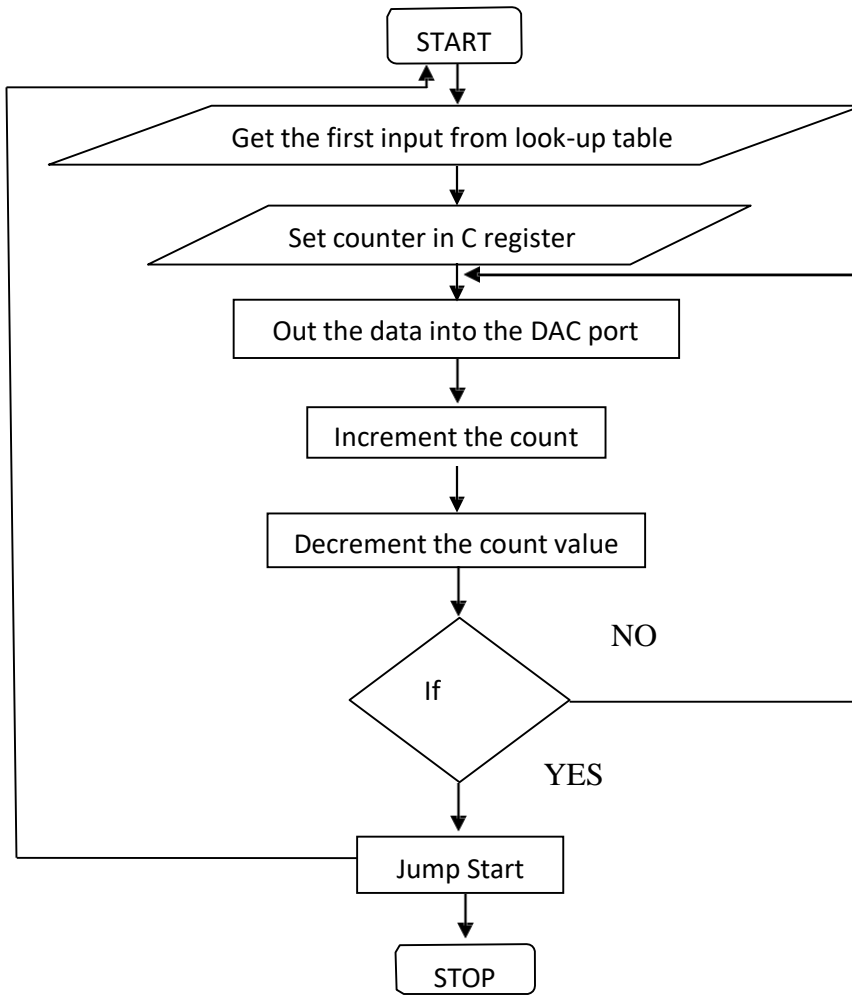
MEMORY ADDRESS	INPUT DATA
8163	89
8164	52
8165	26
8166	06
8167	89
8168	12
8169	25
816A	03
816B	49
816C	A2
816D	94
816E	06
8170	46
8171	A2
8172	44
8173	03

8116		CALL DELAY	CD	Call the delay Label
8117			21	
8118			81	
8119		INX H	23	Increment HL pair
811A		DCR C	OD	Decrement C register
811B		JNZ LOOP1	C2	Go to Loop 1 if no zero
811C			09	
811D			81	
811E		JMP START	C3	Jump to start
811F			04	
8120			81	
8121	DELAY	LXI D,FFFF	11	Delay Program Load Immediate memory content in to D register
8122			FF	
8123			FF	
8124	DLY	DCX D	1B	Decrement DE register
8125		MOV A,E	7B	Move content of E to A
8126		ORA B	B2	
8127		JNZ DLY	C2	Jump on no zero to (Delay) DLY
8128			24	
8129			81	
812A		DCR B	05	Decrement D register
812B		JNZ DELAY	C2	Jump on no zero to Delay
812C			21	
812D			81	
812E		RET	C9	Return to Main program



RESULT:

FLOW CHART:



Ex. No.:**Date : 14. INTERFACING OF D TO A CONVERTER USING 8085 MICROPROCESSOR****AIM:**

To generate triangular wave at DAC output using 8085 microprocessor.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet
- DAC interface board.

ALGORITHM:

1. Start the program
2. Get the First input from lookup table
3. Set the count in c-register
4. Out the data in to the DAC port
5. Increment the look-up table address
6. Increment the count value
7. If carry is equal to zero go to jump start
8. If no carry is go to Out data
9. Stop program

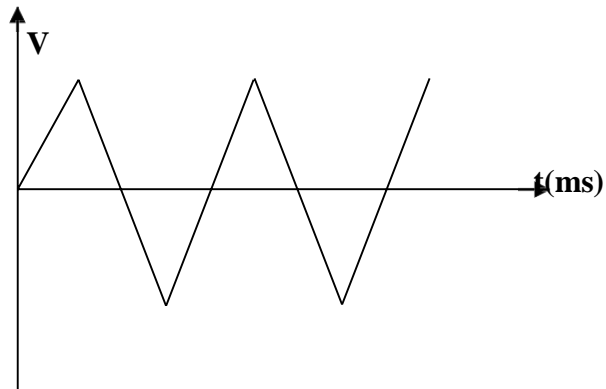
PROGRAM:

ADDRESS	LABLE	PNEMONICS	OPCODE	COMMENTS
8100	START	LXI H, 8110	21	Load the memory address for input data
8101			10	
8102			81	
8103		MVI C , 41	0E	Move 41 to C register
8104			41	
8105	LOOP	MOV A , M	7E	Move the data from M to A
8106		OUT C0	D3	Out the data from port.
8107			C0	
8108		INX H	23	Increment memory pointer

DAC 0800 is an 8-bit DAC and the output voltage varies in between -5v and +5v. The output voltage varies in steps of $10/256 = 0.04$ (approx) the digital data inputs and the corresponding output voltage are presented in the following table.

Input data in Hex	Output voltage (V)
00	0.00
01	0.04
02	0.08
.	.
.	.
.	.
7F	2.15
.	.
.	.
.	.
FD	4.92
FE	4.96
FE	5.00

Model Graph:



TABULATION:

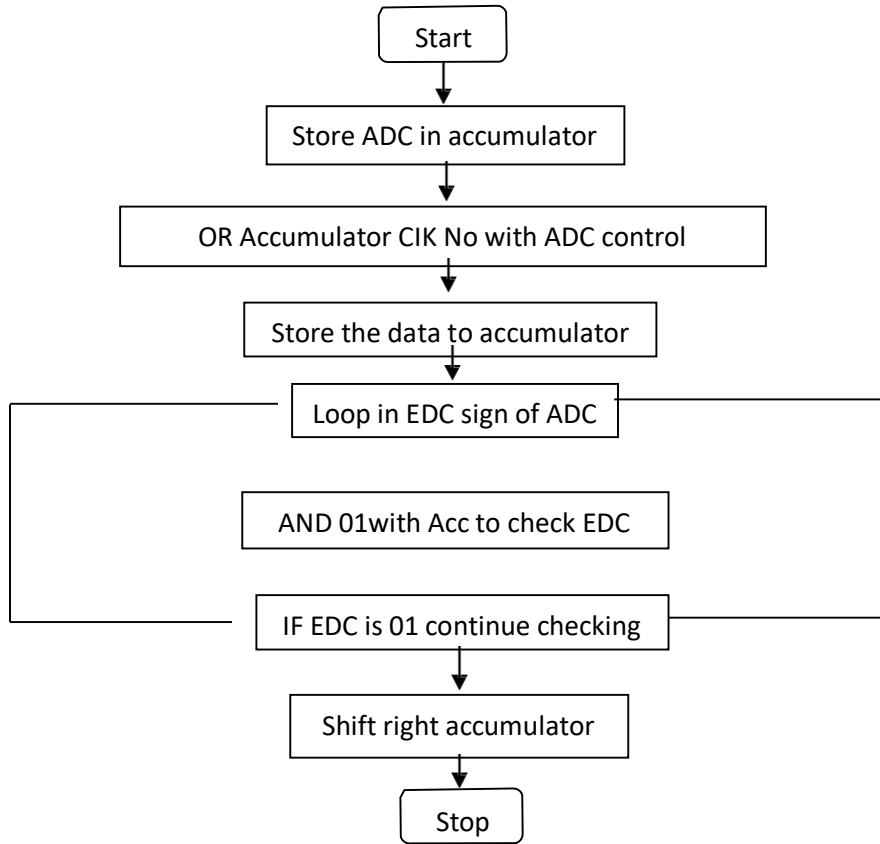
Amplitude	Time period

OUTPUT:

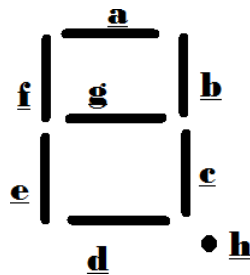
8109		DCR C	OP	Decrement C register.
810A		JNZ LOOP	C2	Jump On no Zero to LOOP
810B			05	
810C			81	
810D		JMP START	C3	Jump to start
810E			00	
810F			81	
8110		LOOK-UP TABLE	00,08,10,18	Look Up data for generation of Triangular Waveform.
8114			20,28,30,38	
8118			40,48,50,58	
811C			60,68,70,78	
8120			80,88,90,98	
8124			A0,A8,B0,B8	
8128			C0,C8,D0,D8	
812C			E0,E8,F0,F8	
8130			FF,F8,F0,E8	
8134			E0,D8,D0,C8	
8138			C0,B8,B0,A8	
813C			A0, 98, 90,88	
8140			80,78,70,68	
8144			60,58,50,48	
8148			40,38,30,28	
814C			20,18,10,08	
8150			00	

RESULT:

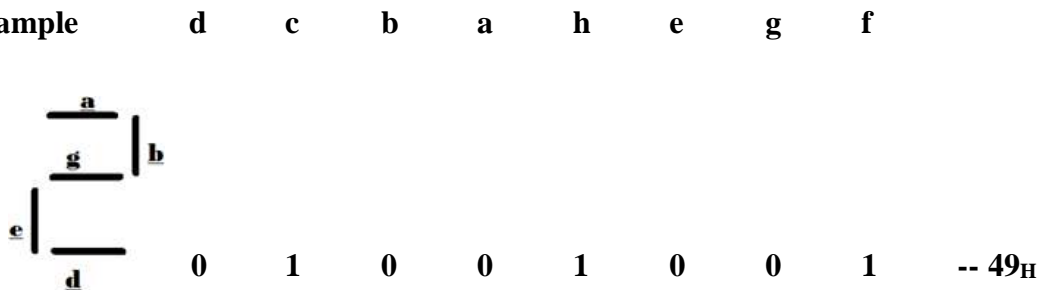
FLOWCHART:



SEVEN SEGMENT DISPLAY



For Example



Ex. No.:

Date : **15. INTERFACING OF A TO D CONVERTER USING 8085 MICROPROCESSOR**

AIM:

To write an assembly level language program to interface A to D converter using 8085 microprocessor.

APPARATUS REQUIRED:

- 8085 microprocessor kit
- Opcode sheet
- ADC interface.

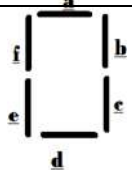
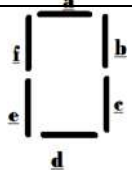
THEORY:

The A/D Conversion is a quantizing process whereby an analog signal is represented by equivalent binary states. This is opposite to b/a conversion process. Analog – to- digital converters can be classified in two general group based on the conversion technique. One technique involves comparing a given analog signal with the internally generated equivalent signal. This group includes successive approximation, counter and flash hypes converters. The second technique involves changing an analog into or frequency and comparing these new parameters against known values this group includes integrator converters and voltage to frequency converters the tradeoff between the two techniques is based on accuracy Vs speed. The successive approximation and the flash hope are faster but generally lees accurate than the in territory and the voltage to frequency hype converters.

ALGORITHM:

1. Start the program
2. Set the ADC control ward to accumulator
3. Load the input value in the accumulator
4. Out the data from ADC port
5. The output are taken in digital form
6. Store the result
7. Stop the program.

INPUT & OUTPUT TABULATION:

Memory Address	Data	7 segment display 	d	c	b	a	h	e	g	f	Hex Value (i/p data)
8200	0		0	0	0	0	1	0	1	0	0A
8201	1		1	0	0	1	1	1	1	1	9F
8202	2		0	1	0	0	1	0	0	1	49
8203	3										0D
8204	4										9C
8205	5										2C
8206	6										28
8207	7										8F

PROGRAM

Address	Label	Mnemonics	Opcode	Comments
8100	START	MVI A,00	3E	Store the ADC channel no to Acc.
8101			00	
8102		OUT C8	D3	Output the control word to ADC control reg.
8103			C8	
8104		ORI 08	F6	Logically OR with A and ALE signal through 08.
8105			08	
8106		OUT 08	D3	Output the control word to ADC control reg.
8107			C8	
8108		NOP	00	Wait for few nano sec.
8109		NOP	00	
810A		NOP	00	
810B		ANI F7	E6	Logically AND with A and Reset ALE signal through F7.
810C			F7	
810D		OUT C8	D3	Output the control word to ADC
810E			C8	
810F		NOP	00	Wait for few nano sec.
8100		NOP	00	
8111		NOP	00	
8112		MVI A,10	3E	Move control word 10 to accumulator
8113			10	
8114		OUT C8	D3	Output the control word to ADC
8115			C8	
8116		NOP	00	Wait for few nano sec.
8117		NOP	00	
8118		NOP	00	
8119		MVI A,20	3E	Move control word 20 to accumulator
811 A			20	
811B		OUT C8	D3	Output the control word to ADC
811C			C8	
811D	LOOP	IN C0	DB	Input the EOC signal from ADC
811E			C0	
811F		ANI 01	E6	Logically AND with 01 and A, to check EOC signal.
8120			01	
8121		JNZ LOOP	C2	Jump on no zero to loop
8122			ID	
8123			81	
8124		IN C4	DB	Input the digital signal from ADC
8125			C4	
8126		MOV B,A	47	Move data from A to B

Memory Address	Data	7 segment display	d	c	b	a	h	e	g	f	Hex Value (i/p data)
8208	8										08
8209	9										8C
820A	A										88
820B	B										38
820C	C										6A
820D	D										19
820E	E										68
820F	F										E8

8127		LXI H, 8200	21	Load the starting address of the lookup table
8128			00	
8129			82	
812A		MVI A,94	3E	Move control word 94 to accumulator
812B			94	
812C		OUT 01	D3	Output the control word to ADC
812D			01	
812E		MOV A , B	78	Move data from B to A
812F		ANI 0F	E6	Logically AND with 0F and A, to get MSD of the digital output.
8130			0F	
8131		RLC	07	Rotate left through Carry
8132		RLC	07	
8133		RLC	07	
8134		RLC	07	
8135		MOV L , A	0F	Move data from A to L
8136		MOV A,M	7E	Move data from M to A
8137		OUT 00	D3	Output the 1 st data to ADC
8138			00	
8139		MOV A , B	78	Move data from B to A
813A		ANI 0F	E6	Logically AND with 0F and A, to get LSD of the digital output.
813B			0F	
813C		MOV L , A	6F	Move data from A to L
813D			7E	Move data from M to A
813E		OUT 00	D3	Output the 2 nd data to ADC
813F			00	
8140		JMP START	C3	Jump to start label.
8141			00	
8142			81	
8143		HLT	76	Stop the program

RESULT:

Date : **16. SERIAL PORT INTERFACE USING 8085
MICROPROCESSOR**

AIM:

To write a program to transmit the data 55 using serial port Interface 8251

APPARATUS REQUIRED:

- 8085 micro processor kit
- Opcode sheet
- Serial port Interface

ALGORITHM

- 1) Initialize Timer for 9600 baud rate
- 2) OUT the data 00 into the USART Port
- 3) Initialize 8251
- 4) Transmit the data in to USART Port
- 5) Receive the same data through USART port
- 6) Stop the program

PROGRAM:

Address	Label	Pneumonic	Opcode	Comments
		ORG 8100H		TIMER CONT
		EQU C3H	C3	
			00	
		EQU C0	C0	CHANNAL 0
			00	
		EQU C5	C5	USART CONT
			00	
		EQU C4	C4	USART DATA
			00	
8100		MVI A , 36	3E	Move control word 36 to A – Register
8101			36	
8102		OUT TIMER CONT	D3	Output the control word to 8251 SPI
8103			C6	
8104		MVI A ,0A	3E	Move data 0A to ACC.
8105			0A	
8106		OUT CHANNAL 0	D3	Output the control word to 8251 SPI
8107			C0	

8108		MVI A , 00	3E	Clear the Accumulator
8109			00	
810A		OUT CHANNEL 0	D3	Output the control word to 8251 SPI
810B			C0	
810C		MVI A , 00	3E	Clear the Accumulator
810D			00	
810E		OUT USARTCONT	D3	Output the control word to 8251 SPI
810F			CA	
8110		OUT USARTCONT	D3	Output the control word to 8251 SPI
8111			CA	
8112		OUT USARTCONT	D3	Output the control word to 8251 SPI
8113			CA	
8114		MVI A , 40	3E	Move data 04 to accumulator
8115			40	
8116		OUT USARTCONT	D3	Output the control word to 8251 SPI
8117			CA	
8118		MVI A , 4E	3E	Move data 4 E to Accumulator
8119			4E	
811A		OUT USARTCONT	D3	Output the control word to 8251 SPI
811B			CA	
811C		MVI A , 37	3E	Move data 37 to Accumulator
811D			37	
811E		OUT USARTCONT	D3	Output the control word to 8251 SPI
811F			CA	
8120	TXDNRDY	IN USARTCONT	DB	Input the USARTCONT to SPI
8121			CA	
8122		ANI 04	E6	Logical AND with Acc and 04
8123			04	
8124		JZ TXDNRDY	CA	Jump on zero to TXDNRDY label
8125			20	
8126			81	
8127		MVI A , 55	3E	Move data 55 to Accumulator
8128			55	
8129		OUT USARTDATA	D3	Output the control word to 8251 SPI
812A			C8	
812B	RXNRDY	IN USARTCONT	DB	Input the USARTCONT to SPI

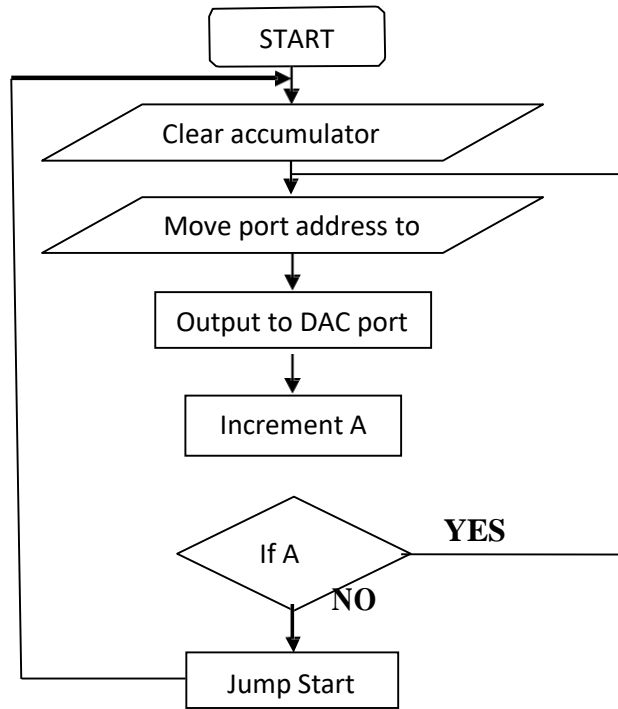
812C			CA	
812D		ANI 02	E6	Logical AND with Acc and 02
812E			02	
812F		JZ RXNRDY	CA	Jump on zero to RXNRDY label
8130			2B	
8131			81	
8132		IN USARTCONT	DB	Input the USARTCONT to SPI
8133			C8	
8134		STA 8500	32	Store the data in 8500
8135			00	
8136			85	
8137		HLT	76	Stop the Program.

INPUT & OUTPUT TABULATION:

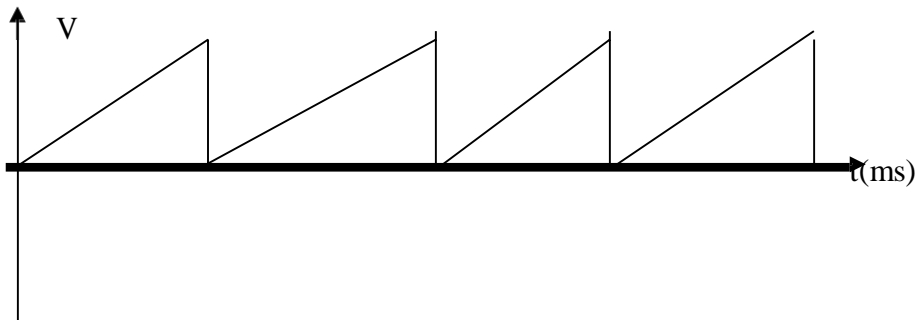
Memory Address	Input data	Memory Address	Output data
8128		8500	

RESULT:

FLOW CHART:



MODEL GRAPH:



OUTPUT:

TABULATION:

Amplitude	Time period

Ex. No.:**Date :** **17. INTERFACING D TO A CONVERTER USING 8051 MICROCONTROLLER****AIM:**

To generate saw both wave at digital to analog converter output.

Apparatus Required:

- 8051 microcontroller
- Opcode sheet
- DAC Interface Board

ALGORITHM:

1. Start the program
2. Clear accumulator
3. Move the port address to DPTR
4. Output to DAC port
5. Increment the accumulator
6. If A is not Zero go to step 4
7. Long jump to Step 1.

PROGRAM:

Address	Label	Mnemonics	Opcode	Comments
8100	START	MOVA,#00	74	Move 00 to accumulator
8101			00	
8102		MOV DPTR, # E0C0	90	DAC1, address in port
8102			E0	
8104			C0	
8105	LOOP	MOVX @DPDR , A	F0	Output to data port
8106		INC A	04	Increment A
8107		JNZ LOOP	70	If A is not zero go to l
8108			FC	
8109		LJMP START	02	Go to start
810A			81	
810B			00	

RESULT:

:

MICROCONTROLLER

Ex. No.:**18. INTERFACING A TO D CONVERTER USING 8051****Date:****AIM:**

To generate saw both wave at analog to digital converter output.

Apparatus Required:

- 8051 microcontroller
- Opcode sheet
- ADC Interface Board

PROGRAM:

Address	Label	Mnemonics	Opcode	Comments
8100	START	MOV A,#00	74,00	ADC select channel
8102		MOV DPTR, #E0C8	90 ,E0,C8	ADC Control port address
8105		MOVX @DPTR,A	F0	Out ADC Channel no to ADC control port
8106		NOP	00	
8107		NOP	00	
8108		NOP	00	
8109		MOV A,#08	74, 08	Send ALE to ADC
PORT				
810B		MOVX @DPTR,A	F0	
810C		NOP	00	
810D		NOP	00	
810E		NOP	00	
810F		MOV A,#10	74,10	Start of conversion
8111		MOVX @DPTR, A	F0	
8112		NOP	00	
8113		NOP	00	
8114		NOP	00	
8115		MOV A,#10	74,20	Output enable
8117		MOVX @DPTR,A	F0	
8118		NOP	00	
8119		NOP	00	
811A		NOP	00	

811B		MOV DPTR,#E0 C0	90, E0, C0	EOC port address
811E :		MOVX A,@DPTR	E0	Get end of the conversion
811F		ANL A,#01	54,01	
8121		JZ 811E	60,FB	If low get EOC again
8123		MOV DPTR, #E0C4	90,E0,C4	Data port address
8126		MOVX A,@DPTR	E0	
8127		MOV DPTR,#8500	90,8500	Store data
812A		MOVX @DPTR,A	F0	
812B		LIMP 8100	02, 812B	

RESULT:

Ex. No.:

Date : 19. INTERFACING OF DC MOTOR USING 8051 MICROCONTROLLER

AIM:

To control the speed of a DC motor using 8253.

ALGORITHM:

- Initialize 8253 counter 0 in mode 3 (Square wave generator). It gives frequency input to FTOV converter for the desired speed.
- Load counter 0 with count proportional to the speed required.
- Give input frequency for the speed required at 8200H in hex.

PROGRAM:

ADDR	OPCODES	MNEMONICS	COMMENTS
;To give frequency input to FTOV convertor			
8100	74 36	MOV A, #36H	; 8253 counter 0 in mode 3 ; square wave generator
8102	90 E0 0B	MOV DPTR, #E00B	
8105	F0	MOVX @DPTR, A	
;To load the count in 8253 counter 0			
8106	90 82 00	MOV DPTR, #8200H	;Read LSB count ; from 8200H
8109	E0	MOVX A, @DPTR	
810A	90 E0 08	MOV DPTR, #E008H	;counter 0 addr.
810D	F0	MOVX @DPTR, A	;Output MSB count
810E	90 82 01	MOV DPTR, #8201H	;Read MSB count from 8201H
8111	E0	MOV A, @DPTR	
8112	90 E0 08	MOV DPTR, #E008H	;counter 0 addr.
8115	F0	MOVX @DPTR, A	;output MSB count
8116	80 FE	SJMP HERE	

Verification:

Refer the verification procedure in 8085 programming enclosed at the previous section.

LOOKUP TABLE

INPUT	SPEED (RPM)
FFFF	100
FC54	150
F8A9	200
F4FE	250
F153	300
EDA8	350
E9FD	400
E652	450
E2A7	500
DEFC	550
DB51	600
D7A6	650
D3FB	700
D050	750
CCA5	800
C8FA	850
C54F	900
C1A4	950
BDF9	1000

BA4F	1050
B6A3	1100
B2F8	1150
AF4D	1200
ABA2	1250
A7F7	1300
A44C	1350
A0A1	1400
9CF6	1450
994B	1500
95A0	1550
91F5	1600
8E4A	1650
8A9F	1700
86F4	1750
8349	1800
7F9E	1850
7BF3	1900
7848	1950
749D	2000
70F2	2050
6D47	2100
699C	2150
65F1	2200
6246	2250
5E9B	2300

5AF0	2350
5745	2400
539A	2450
4FEF	2500

RESULT:

Ex. No.:

Date : 20. INTERFACING OF AC MOTOR USING 8051 MICROCONTROLLER

AIM:

To Control the speed of AC motor by controlling the firing pulses.

Requirement:

- LABTECH'S AC motor Speed Controller interface Board
- Ac motor
- MP/MC trainer kit
- 26 pin interface cable

Procedure:

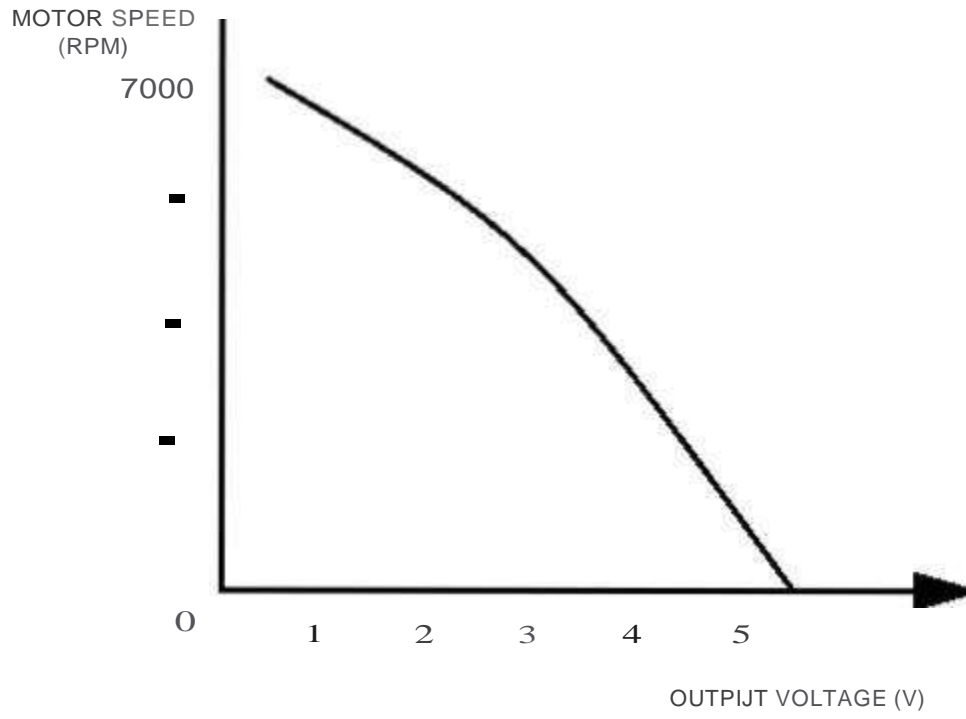
- Interface the MP/MC kit with AC Motor speed controller board using 26 pin FRC cable provided.
- Switch ON the Trainer
- Type the Program given below in the memory location with the starting address 8100H.

- ❖ Execute the following program and observe that the output voltage at DAC1. Change the value in A and observe the corresponding output voltage at DAC1. Give Digital input for the speed required at 8107H in hex.

PROGRAM:

			ORG8100H	
ADDR	OPCODE	LABEL	MNEMONICS	COMMENTS
8100	74 80		MOV A, #80H	
8102	90 E0 1B		MOV DPTR,#E01BH	
8105	F0 74 7F		MOV A, #7FH	;Move '7F' to acc
8108	90 E0 18		MOV DPTR,#E018H	;DAC1 address in DPTR
810B	F0		MOVX @DPTR,A	;Output to DAC1
810C	80 FE	HERE:	SJMP HERE	;Jump here itself

MODEL GRAPH:



DATA TABLE:

INPUT DATA IN HEX	OUTPUT VOLTAGE (V)	MOTOR SPEED (RPM)
00	0.00	20000
01	0.04	-
02	0.08	-
.	.	.
.	.	.
7F	2.50	-
.	.	.
.	.	.
FE	4.96	-
FF	5.00	0

RESULT: