SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE

SCHOOL OF COMPUTING

SECA2405 - MICROPROCESSOR AND MICROCONTROLLER LAB MANUAL

PART – A

REGULATIONS 2019

IV SEMESTER - CSE

CONTENTS

LIST OF EXPERIMENTS
(AS PER UNIVERSITY SYLLABUS)

SECA2405	MICROPROCESSOR AND MICROCONTROLLER LAB	L	T	P	Credits	Total marks
		0	0	4	2	100

Part – A- MICROPROCESSOR- 8085

1. Programs using Arithmetic Operations
2. Programs for Code Conversions
3. Largest, Smallest and Sorting of an Array (8085)

Part – B- MICROCONTROLLER- 8051

1. Data Transfer Programs
2. Programs using Logical Instructions
3. Programs using Boolean Instructions
4. Reading and Writing on a Parallel Port
5. Stepper Motor Controller
6. Timer in Different Modes
7. Serial Communication Implementation

INDEX

MICROPROCESSOR AND MICROCONTROLLER LAB

RECORD NOTEBOOK / OBSERVATION WRITING FORMAT

LEFT HAND SIDE (UNRULED)	RIGHT HAND SIDE (RULED)
1 FLOW CHART	1 AIM
2 PROGRAMS	2 REQUIRED HARDWARE AND SOFTWARE
3 MANUAL CALCULATIONS	3 ALGORITHMS
	4 PROCEDURE
	5 RESULT

ARITHMETIC OPERATIONS USING 8085

1. ADDITION OF TWO 8-BIT NUMBERS USING 8085

AIM: To write and execute an assembly language program to perform the addition of two 8-bit numbers using 8085 JUBIN simulator.

HARDWARE/SOFTWARE REQUIRED: A PC loaded with 8085 JUBIN Simulator (Windows Version)

ALGORITHM:

Step 1 : Start.
Step 2 : Clear C register for carry
Step 3 : Load the first data from memory to accumulator and move it to B register.
Step 4 : Load the second data from memory to accumulator.
Step 5 : Add the content of B register to the accumulator.
Step 6 : Check for carry. If carry = 1, go to step 7 else if carry =0, go to step 8.
Step 7 : Increment the C register.
Step 8 : Store the sum in memory
Step 9 : Move the carry to accumulator and store in memory
Step 10 : End Program

In this program Registers A, B C are used for Data Manipulation PROGRAM:

Label	Instruction		&& Comment
	Mnemonic	Operand	
	MVI	C,00	Clear C register for carry
	LDA	4100	Load the first data from memory to accumulator
	MOV	B,A	Move the data to B register.
	LDA	4101	Load the second data from memory to accumulator.
	ADD	B	Add the content of B register to the accumulator.
	JNC	L1	Check for carry. If carry = 1, increment carry register else if carry =0 store the result
	INR	C	Increment the C register for carry
L1	STA	4102	Store the sum in memory
	MOV	A,C	Move the carry to accumulator
	STA	4103	Store Carry in memory location
	HLT		End the program

MANUAL CALCULATION: (FOR THE INPUT USED AS DATA)

	BIT POSITION			8	7	6	5	4	3	2	1	0
	Carry Details			1	1	1	1	1	1			
INPUT	DATA 1		AE		1	0	1	0	1	1	1	0
	DATA 2	+	FD		1	1	1	1	1	1	0	1
OUTPUT	SUM	=	AB		1	0	1	0	1	0	1	1
	CARRY	=	01									

***** NOTE:**
This shall be considered as an example manual calculation and to be done for all arithmetic or logical programs like exercise numbers 1.1, 1.2, 1.3, 2.1 & 2.2 by students.

SAMPLE INPUT & OUTPUT:

Before Execution		
Memory Address	Data	
4100	AE	Augend
4101	FD	Addend

After Execution		
Memory Address	Data	
4102	AB	Sum
4103	01	Carry

Addition:

FD+AE=01AB

RESULT:

A program to perform addition of two 8-bit numbers using 8085 emulator is written and executed.

2. SUBTRACTION OF TWO 8-BIT NUMBERS USING 8085

AIM: To write and execute a program to perform subtraction of two 8-bit numbers using 8085 simulator

HARDWARE/SOFTWARE REQUIRED: A PC loaded with 8085 Emulator (Windows Version)

ALGORITHM:

Step 1 : Start.
Step 2 : Clear C register to account for sign of the result.
Step 3 : Load the subtrahend (the data to be subtracted) from memory to accumulator and move it to B register.
Step 4 : Load the minuend from memory to accumulator.
Step 5 : Subtract the content of B register from the content of the accumulator. (A-B)
Step 6 : Check for carry. If carry = 1, go to step 7 else if carry =0, go to step 8.
Step 7 : Increment the C register. Complement the accumulator and add 01H.
Step 8 : Store the difference in memory.
Step 9 : Move the content of C register (sign bit) to accumulator and store in memory.
Step 10 : End program

PROGRAM:

Label	**Instruction**	
	Mnemonic	**Operand**
	MVI	C,00
	LDA	4100
	MOV	B,A
	LDA	4101
	SUB	B
	JNC	L1
	INR	C
	CMA	
	INR	A
L1	STA	4102
	MOV	A,C
	STA	4103
	HLT	

SAMPLE INPUT & OUTPUT:

Before Execution		
Memory Address	**Data**	
4100	AF	Minuend
4101	EF	Subtrahend

After Execution		
Memory Address	**Data**	
4102	40	Difference
4103	00	Borrow

Subtraction: AF-EF=40

EF is loaded in Accumulator(A) and AF is loaded in B as per program.
A-B is done by ALU. Thus EF-AF is done. EF is greater than AF and hence difference is a positive number. (Borrow remains 0.)

If AF is loaded in A, and EF is loaded in B, then AF-EF is executed. Hence the answer is negative number. When negative number, Borrow becomes 1.

RESULT:

A program to perform subtraction of two 8-bit numbers using 8085 simulators is written and executed.

3. MULTIPLICATION OF TWO 8-BIT NUMBERS USING 8085

AIM:
To write and execute an assembly language program to perform multiplication of two 8-bit numbers using 8085 simulator.

HARDWARE/SOFTWARE REQUIRED: A PC loaded with 8085 simulators (Windows Version)

ALGORITHM:

Step 1 : Start.
Step 2 : Clear Reg D for Carry
Step 3 : Load 1st Number (Multiplicand in Reg B through reg A
Step 4 : Load 2nd Number (Multiplier in Reg C through reg A
Step 5 : Clear Reg A (Accumulator)
Step 6 : Now add content of Reg B with Reg A
Step 7 : Check for Carry if carry NOT exists then skip next step
Step 8 : Decrement 2nd number (multiplier) in Reg C
Step 9 : Check for Reg C for zero, If NOT zero go to step 6 Else continue
Step 10 : Store the result (product) in reg A into output memory address
Step 11 : Store the Carry in Reg D into next output memory address
Step 12 : End Program

PROGRAM:

Label	Instruction	
	Mnemonic	**Operand**
	MVI	D, 00
	LDA	4100
	MOV	B, A
	LDA	4101
	MOV	C, A
	MVI	A, 00
L2	ADD	B
	JNC	L1
	INR	D
L1	DCR	C
	JNZ	L2
	STA	4102
	MOV	A, D
	STA	4103
	HLT	

SAMPLE INPUT & OUTPUT:

Before Execution		
Memory Address	**Data**	
4100	EF	Multiplicand
4101	AF	Multiplier

After Execution		
Memory Address	**Data**	
4102	61	LSB of Product
4103	A3	MSB of the Product

Multiplication: EF*AF= A361

RESULT:

A program to perform multiplication of two 8-bit numbers using 8085 simulator is written and executed.

4. DIVISION OF TWO 8-BIT NUMBERS USING 8085

AIM:
To write and execute an assembly language program to perform the division of two 8-bit numbers using 8085 simulator.

HARDWARE/SOFTWARE REQUIRED: A PC loaded with 8085 simulator (Windows Version)

ALGORITHM:

Step 1 : Start.
Step 2 : Clear C register to account for quotient.
Step 3 : Load the divisor in accumulator and move it to B register.
Step 4 : Load the dividend in the accumulator.
Step 5 : Check whether divisor is less than dividend. If divisor is less than dividend, go to step 9, else go to next step.
Step 6 : Subtract the content of B register from accumulator.
Step 7 : Increment the content of C register (quotient).
Step 8 : Go to step 5.
Step 9 : Store the content of accumulator (remainder) in memory.
Step 10: Move the content of C register (quotient) to accumulator and store in memory.
Step 11 : End program.

PROGRAM:

Label	**Instruction**	
	Mnemonic	**Operand**
	MVI	C,00
	LDA	4101
	MOV	B,A
	LDA	4100
L2	CMP	B
	JC	L1
	SUB	B
	INR	C
	JMP	L2
L1	STA	4102
	MOV	A,C
	STA	4103
	HLT	

SAMPLE INPUT & OUTPUT:

Before Execution		
Memory Address	**Data**	
4100	EF	Divisor
4101	AF	Dividend

After Execution		
Memory Address	**Data**	
4102	40	Remainder
4103	01	Quotient

Division: EF/AF= Remainder (40) Quotient (01)

RESULT:
A program to perform division of two 8-bit numbers using 8085 simulators is written and executed.

PROGRAMS FOR CODE CONVERSIONS

5. CONVERTING A BCD NUMBER TO HEX NUMBER USING 8085

AIM:

To write a program to convert BCD number in memory to the equivalent HEX number using 8085 simulator.

HARDWARE/SOFTWARE REQUIRED: A PC loaded with 8085 Emulator (Windows Version)

ALGORITHM:

Step 1 : Start
Step 2 : Load the BCD number in accumulator (Reg A)
Step 3 : Copy it to Reg E
Step 4 : Mask the Lower Nibble of the BCD number
Step 5 : Rotate it 4 times to move Upper nibble to lower nibble place
Step 6 : Copy the number to Reg B
Step 7 : Clear Reg A
Step 8 : Copy number "0A" in Reg C (Decimal number 10)
Step 9 : Multiply the number in B by 10d
Step 10: Then move the product in Reg B
Step 11: Copy the original number in Reg E to Reg A
Step 12: Mask the Upper Nibble of the BCD Number
Step 13: Add the Number in Reg B with the number in Reg A
Step 15: The number is copied to a memory location
Step 14: End program.

PROGRAM:

Label	Instruction	
	Mnemonic	**Operand**
	LDA	4100
	MOV	E, A
	ANI	F0
	RRC	
	RRC	
	RRC	
	RRC	
	MOV	B, A
	MVI	A,00
	MVI	C,0A
L1	ADD	B
	DCR	C
	JNZ	L1
	MOV	B, A
	MOV	A, E
	ANI	0F

Label	Instruction	
	Mnemonic	**Operand**
	ADD	B
	STA	4101
	HLT	

SAMPLE INPUT & OUTPUT:

Before Execution		
Memory Address	**Data**	
4100	29	(BCD)

After Execution		
Memory Address	**Data**	
4101	1D	(HEX)

BCD (29d) = Hexadecimal (1DH)

RESULT:

A program to convert the Binary coded decimal (BCD) into Hexadecimal number (HEX) has been written and executed using microprocessor 8085 emulator.

6.CONVERTING HEXADECIMAL NUMBER TO BCD NUMBER Using 8085

AIM:

To write a program to convert HEX numbers in memory to the equivalent BCD number using 8085 simulator.

HARDWARE/SOFTWARE REQUIRED: A PC loaded with 8085 simulators

ALGORITHM:

Step 1 : Start
Step 2 : Clear Reg D and E to store tens and hundreds respectively
Step 3 : Load the HEX number in accumulator (Reg A)
Step 4 : Compare Hex number with 64h (100d)**(Check for Hundreds)**
Step 5 : If carry , HEX <64 so go to Check Tens (Step 8) NO Carry Next step
Step 6 : Subtract 64 h (100s) from the number.
Step 7 : Increment Reg E for each hundred and repeat step 6 until HEX becomes less than 100d (64H)
Step 8 : Compare Remaining Hex number with 0Ah (10d) **(Check for Tens)**
Step 9 : If carry , Remaining HEX <10 so go to Check ones (Step) NO Carry Next step
Step 10: Subtract 0A h (10s) from the number.
Step 11: Increment Reg D for each hundred and repeat step 6 until HEX becomes less than 10d (0AH)
Step 12: Now the remaining is ones and is stored in Reg C
Step 13: Now tens are copied from reg D to Reg A
Step 15: The number is rotated 4 times to move it to tens position (upper Nibble)
Step 14: Then the number is added with ones in Reg c
Step 15: Store the Combined tens and ones in reg A to memory location
Step 16: Store the Hundreds in Reg E to next memory location
Step 17: End program.

PROGRAM:

Label	Instruction	
	Mnemonic	**Operand**
	MVI	D, 00
	MVI	E, 00
	LDA	4100
L2	CPI	64
	JC	L1
	SUI	64
	INR	E
	JMP	L2
L1	CPI	0A
	JC	L3
	SUI	0A
	INR	D
	JMP	L1

L3	MOV	C, A
	MOV	A, D
	RRC	
	RRC	
	RRC	
	RRC	
	ADD	C
	STA	4101
	MOV	A, E
	STA	4102
	HLT	

SAMPLE INPUT & OUTPUT:

Before Execution		
Memory Address	**Data**	
4100	1D	(Hex)

After Execution		
Memory Address	**Data**	
4101	29	(BCD) Tens and ones
4102	00	Hundreds

Hexadecimal (1DH) = BCD (0029d)

RESULT:

The program to convert the hexadecimal number (HEX) into Binary coded decimal (BCD) has been written and executed using 8085 Simulator.

7.LARGEST OF A GIVEN SET OF NUMBERS USING 8085

AIM:

To write and execute a program to search the largest of a given set of array of numbers using 8085 emulator.

HARDWARE/SOFTWARE REQUIRED: A PC loaded with 8085 Emulator (Windows Version)

ALGORITHM:

Step 1 : Start.
Step 2 : Load the address of the first element of the array in HL register pair (pointer).
Step 3 : Move the count to B register.
Step 4 : Increment the pointer.
Step 5 : Get the first data in the accumulator.
Step 6 : Decrement the count.
Step 7 : Increment the pointer.
Step 8 : Compare the content of memory addresses by HL pair with that of accumulator.
Step 9 : If CF=0, go to step 11 else go to step 10.
Step 10: Move the content memory addressed HL to accumulator.
Step 11: Decrement the count.
Step 12: Check for zero for the count. If ZF=0, go to step 7 else go to next step.
Step 13: Store the largest data in the memory.
Step 14: End program.

PROGRAM:

Label	**Instruction**	
	Mnemonic	**Operand**
	LXI	H,4100
	MOV	B,M
	INX	H
	MOV	A,M
	DCR	B
L2	INX	H
	CMP	M
	JNC	L1
	MOV	A,M
L1	DCR	B
	JNZ	L2
	STA	4106
	HLT	

SAMPLE INPUT & OUTPUT:

Before Execution		
Memory Address	**Data**	
4100	05	Count
4101	EF	Data 1
4102	DA	Data 2
4103	FD	Data 3
4104	12	Data 4
4105	05	Data 5

After Execution		
Memory Address	**Data**	
4106	FD	Largest

RESULT:

The program to find the largest number in an array has been written and executed using 8085 simulator.

8.SMALLEST OF A GIVEN SET OF NUMBERS USING 8085

AIM:

To write and execute program to search the smallest of a given set of array of numbers using 8085 emulator.

HARDWARE/SOFTWARE REQUIRED: A PC loaded with 8085 Simulator (Windows Version)

ALGORITHM:

Step 1 : Start.
Step 2 : Load the address of the first element of the array in HL register pair (pointer).
Step 3 : Move the count to B register.
Step 4 : Increment the pointer.
Step 5 : Get the first data in the accumulator.
Step 6 : Decrement the count.
Step 7 : Increment the pointer.
Step 8 : Compare the content of memory addresses by HL pair with that of accumulator.
Step 9 : If CF=1, go to step 11 else go to step 10.
Step 10: Move the content memory addressed HL to accumulator.
Step 11: Decrement the count.
Step 12: Check for zero for the count. If ZF=0, go to step 7 else go to next step.
Step 13: Store the smallest data in the memory.
Step 14: End program.

PROGRAM:

Label	Instruction	
	Mnemonic	**Operand**
	LXI	H, 4100
	MOV	B,M
	INX	H
	MOV	A,M
	DCR	B
L2	INX	H
	CMP	M
	JC	L1
	MOV	A,M
L1	DCR	B
	JNZ	L2
	STA	4106
	HLT	

SAMPLE INPUT & OUTPUT:

Before Execution		
Memory Address	**Data**	
4100	05	Count
4101	EF	Data 1
4102	DA	Data 2
4103	FD	Data 3
4104	12	Data 4
4105	05	Data 5

After Execution		
Memory Address	**Data**	
4106	05	Smallest

RESULT:

The program to find the smallest number in an array has been written and executed using 8085 simulator.

NOTE:
Finding Largest and smallest numbers programs are the same except the commands JC and JNC.

9.ARRANGING A GIVEN SET OF NUMBERS IN ASCENDING ORDER USING 8085

AIM:

To write and execute program to arrange a given set of array of numbers in ascending order using 8085 emulator.

HARDWARE/SOFTWARE REQUIRED: A PC loaded with 8085 Emulator (Windows Version)

ALGORITHM:

Step 1 : Start.
Step 2 : Initialize HL register to 4100
Step 3 : Initialize Register B for number of comparisons.(n-1)
Step 4 : Initialize HL register to 4100 (loop 3) Redoing whole process again
Step 5 : Initialize Register C also for number of comparisons.(n-1)
Step 6 : Transfer first number to Accumulator (A) from 4100H
(Loop 2) **Step 7 :** Increment HL pair to memory location of next number (4101)
Step 8 : Compare first number in A and second number in M (HL Pair Pointer).
Step 9 : If First number (A) is less than second number (M) GO to loop L1
Step 10: Else transfer the smaller Second number from (M) to previous location (for example 4100)
Step 11: Now copy the larger first number to M (4101)
Step 12: Increment HL pair. (4101)
Step 13: Decrement C register as one comparison is done. (Loop 1)
Step 14: Check if C is zero. If NOT zero go to (loop 2), Else goto next step
Step 15: Decrement Register B
Step 16: Check if B is zero. If NOT zero go to (loop 2) and redo all the numbers again.
Step 17: Stop the program

PROGRAM:

Label	Instruction	
	Mnemonic	**Operand**
	MVI	B, 04
L3	LXI	H, 4100
	MVI	C, 04
L2	MOV	A, M
	INX	H
	CMP	M
	JC	L1
	MOV	D,M
	MOV	M,A
	DCX	H
	MOV	M,D
	INX	H
L1	DCR	C
	JNZ	L2
	DCR	B
	JNZ	L3
	HLT	

SAMPLE INPUT & OUTPUT:

Before Execution		
Memory Address	**Data**	
4100	EF	Data 1
4101	DA	Data 2
4102	FD	Data 3
4103	12	Data 4
4104	05	Data 5

After Execution		
Memory Address	**Data**	
4100	05	Data 5
4101	12	Data 4
4102	DA	Data 2
4103	EF	Data 1
4104	FD	Data 3

RESULT:

The program to arrange the array of given numbers in ascending order has been written and executed using 8085 simulator.

10.ARRANGING A GIVEN SET OF NUMBERS IN DESCENDING ORDER USING 8085

AIM:

To write and execute program to arrange a given set of array of numbers in descending order using 8085 emulator

HARDWARE/SOFTWARE REQUIRED: A PC loaded with 8085 Emulator (Windows Version)

ALGORITHM:

Step 1 : Start.
Step 2 : Initialize HL register to 4100
Step 3 : Initialize Register B for number of comparisons.(n-1)
Step 4 : Initialize HL register to 4100 (loop 3) Redoing whole process again
Step 5 : Initialize Register C also for number of comparisons.(n-1)
Step 6 : Transfer first number to Accumulator (A) from 4100H (Loop 2)
Step 7 : Increment HL pair to memory location of next number (4101)
Step 8 : Compare first number in A and second number in M (HL Pair Pointer).
Step 9 : If First number (A) is greater than second number (M) GO to loop L1
Step 10: Else transfer the larger Second number from (M) to previous location (for example 4100)
Step 11: Now copy the smaller first number to M (4101)
Step 12: Increment HL pair. (4101)
Step 13: Decrement C register as one comparison is done. (Loop 1)
Step 14: Check if C is zero. If NOT zero go to (loop 2), Else goto next step
Step 15: Decrement Register B
Step 16: Check if B is zero. If NOT zero go to (loop 2) and redo all the numbers again.
Step 17: Stop the program

PROGRAM:

Label	Instruction	
	Mnemonic	**Operand**
	MVI	B, 04
L3	LXI	H, 4100
	MVI	C, 04
L2	MOV	A, M
	INX	H
	CMP	M
	JNC	L1
	MOV	D,M
	MOV	M,A
	DCX	H
	MOV	M,D
	INX	H
L1	DCR	C
	JNZ	L2

	DCR	B
	JNZ	L3
	HLT	

SAMPLE INPUT & OUTPUT:

Before Execution		
Memory Address	**Data**	
4100	EF	Data 1
4101	DA	Data 2
4102	FD	Data 3
4103	12	Data 4
4104	05	Data 5

After Execution		
Memory Address	**Data**	
4100	FD	Data 3
4101	EF	Data 1
4102	DA	Data 2
4103	12	Data 4
4104	05	Data 5

RESULT:

The program to arrange the array of given numbers in descending order has been written and executed using 8085 Simulator.

NOTE:

Ascending and Descending programs are the same except the commands JC and JNC.