VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

BECE204P – MICROPROCESSORS AND MICROCONTROLLERS LAB RECORD

Submitted By

21BEC1851 – Rahul Karthik S

Submitted To

Dr. Prakash V

LAB – 01: 8086 – Arithmetic Operations (Addition and Subtraction)

LABTASK – 1: 8 – BIT ADDITION

AIM:

To write 8086 Assembly language program to add two 8-bit number in AL, BL registers and store the result in memory location 2000H.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

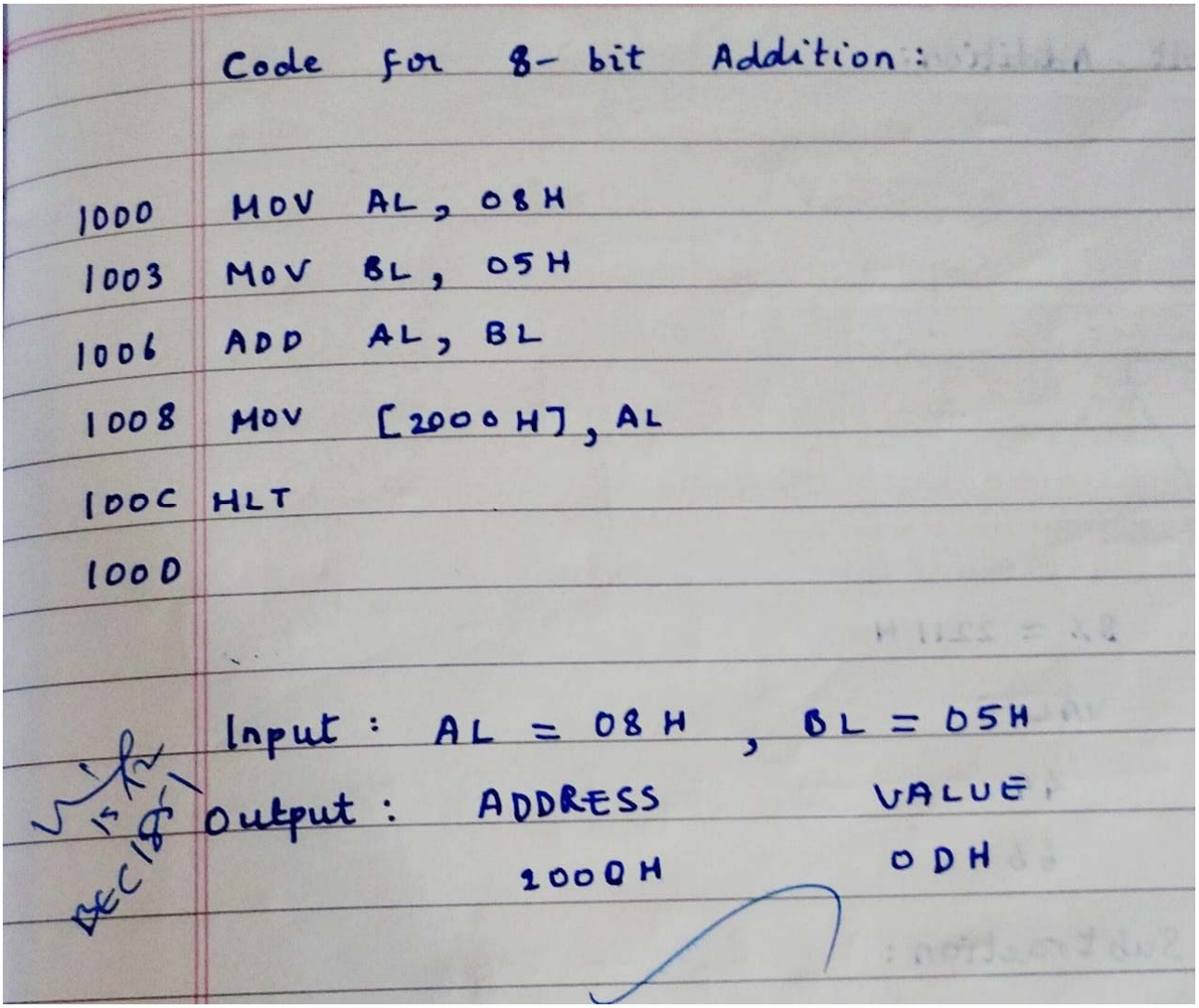
ADDRESS	MEMONICS	COMMENTS
1000	MOV AL,08H	Move data 08H to AL register
1003	MOV BL,05H	Move data 05H to BL register
1006	ADD AL, BL	Add AL and BL content
1008	MOV [2000H], AL	Move AL to the memory location 2000H

100C	HLT	Halt the program

OUTPUT:

Address	Value
2000H	0DH

OUTPUT VERIFICATION:



INFERENCE:

The output is found out to be "0DH" and stored in the address 2000H.

RESULT:

Thus the 8086 Assembly language program to add two 8-bit number in AL, BL registers and store the result in memory location 2000H is written and executed successfully.

LABTASK – 2: 16 – BIT ADDITION

AIM:

To write 8086 Assembly language program to add two 16-bit number in AX, BX registers and store the result in memory location 2000H.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

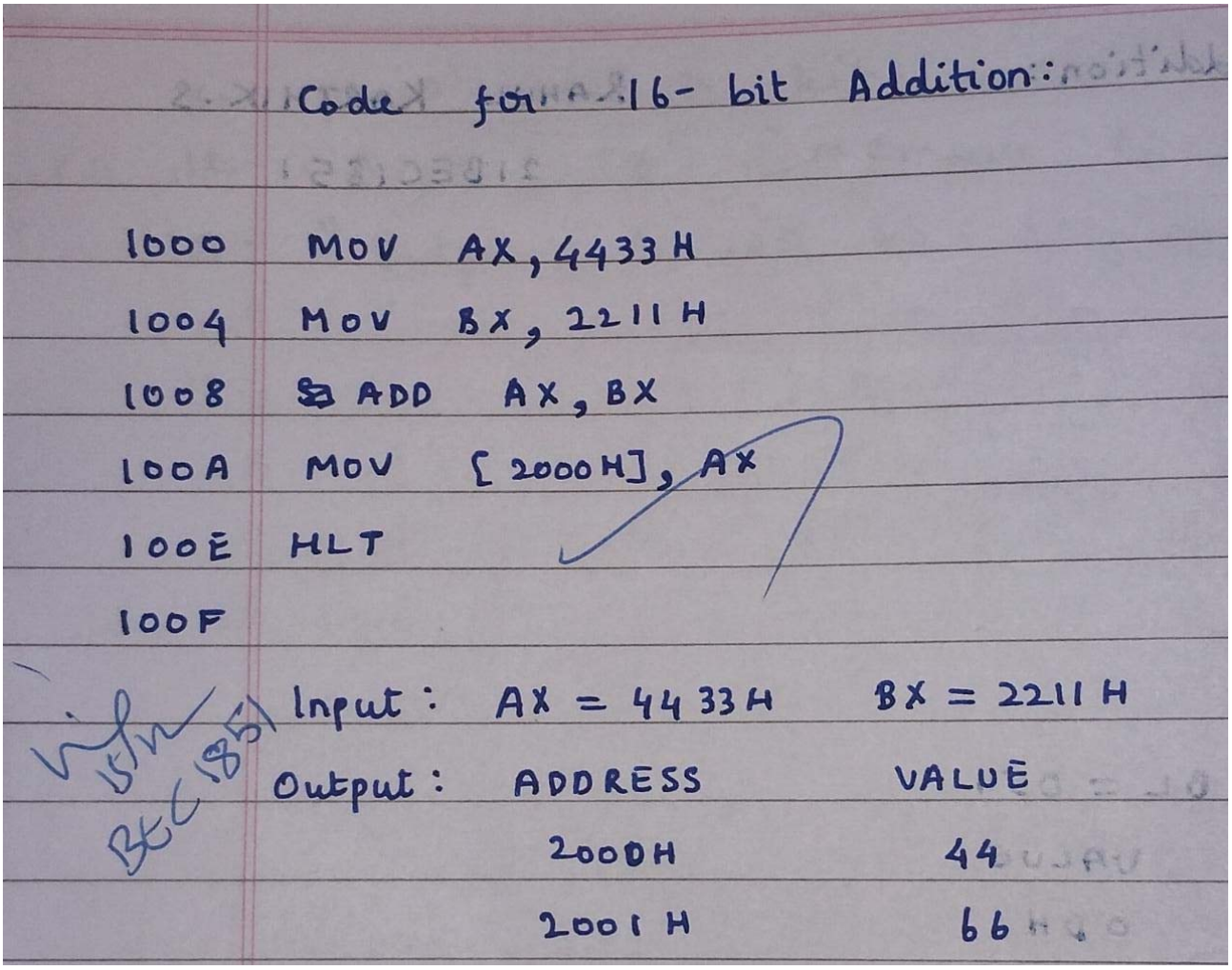
ADDRESS	MEMONICS	COMMENTS
1000	MOV AX,4433H	Move data 4433H to AX register
1004	MOV BX,2211H	Move data 2211H to BX register
1008	ADD AX, BX	Add AX and BX content
100A	MOV [2000H], AX	Move AX to the memory location 2000H
100E	HLT	Halt the program

OUTPUT:

Address	Value
2000H	44

2001H	66

OUTPUT VERIFICATION:



INFERENCE:

The output is found out to be “6644H” and lower byte is stored in the address 2000H and higher byte is stored in the address 2001H.

RESULT:

Thus the 8086 Assembly language program to add two 16-bit number in AX, BX registers and store the result in memory location 2000H is written and executed successfully.

LABTASK – 3: 8 – BIT SUBTRACTION

AIM:

To write 8086 Assembly language program to subtract two 8-bit number in AL, BL registers and store the result in memory location 2000H.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

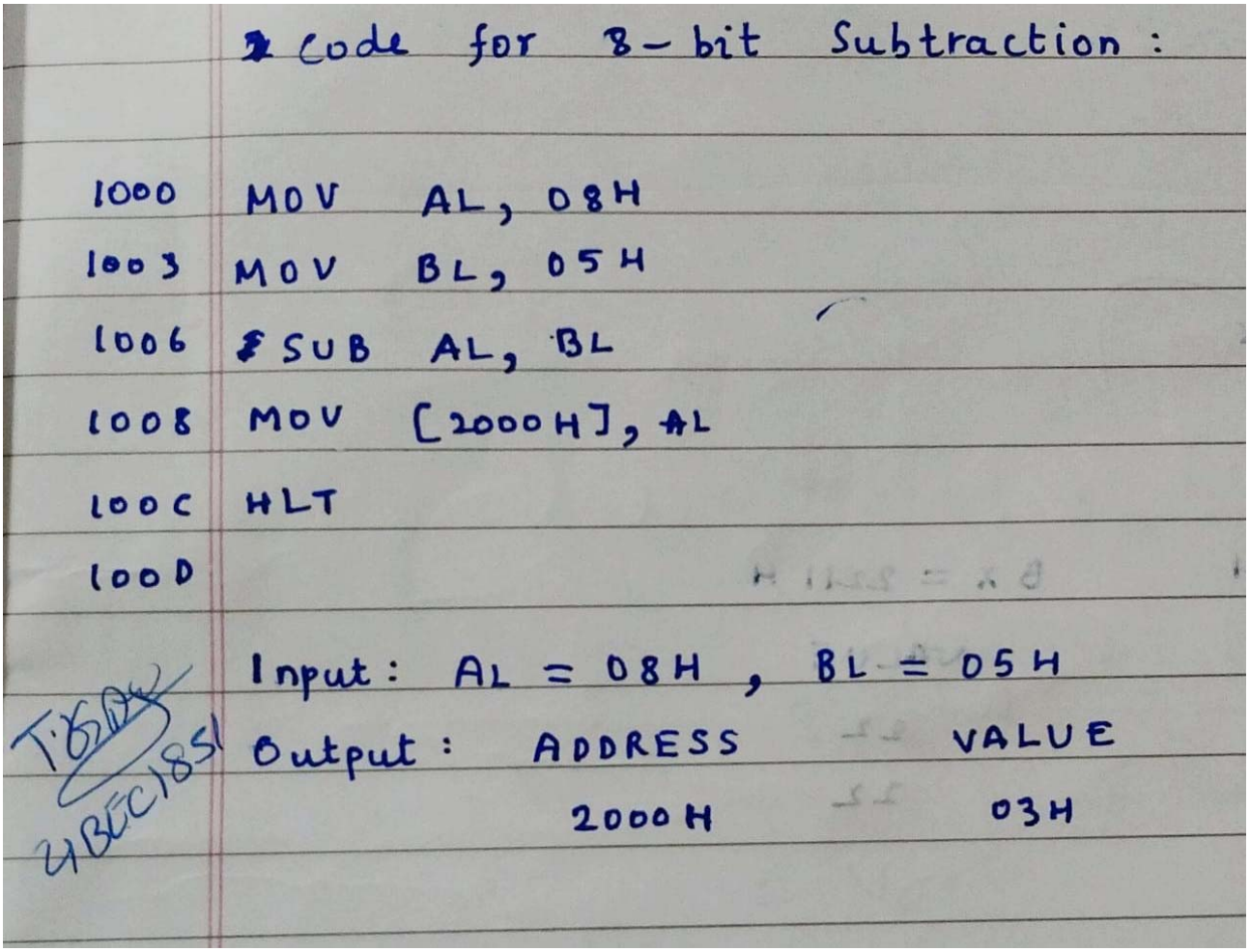
PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV AL,08H	Move data 08H to AL register
1003	MOV BL,05H	Move data 05H to BL register
1006	SUB AL, BL	Subtract AL and BL content
1008	MOV [2000H], AL	Move AL to the memory location 2000H
100C	HLT	Halt the program

OUTPUT:

Address	Value
2000H	03H

OUTPUT VERIFICATION:



INFERENCE:

The output is found out to be “03H” and it is stored in the address “2000H”.

RESULT:

Thus the 8086 Assembly language program to subtract two 8-bit number in AL, BL registers and store the result in memory location 2000H is written and executed successfully.

LABTASK – 4: 16 – BIT SUBTRACTION

AIM:

To write 8086 Assembly language program to subtract two 16-bit number in AL, BL registers and store the result in memory location 2000H.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

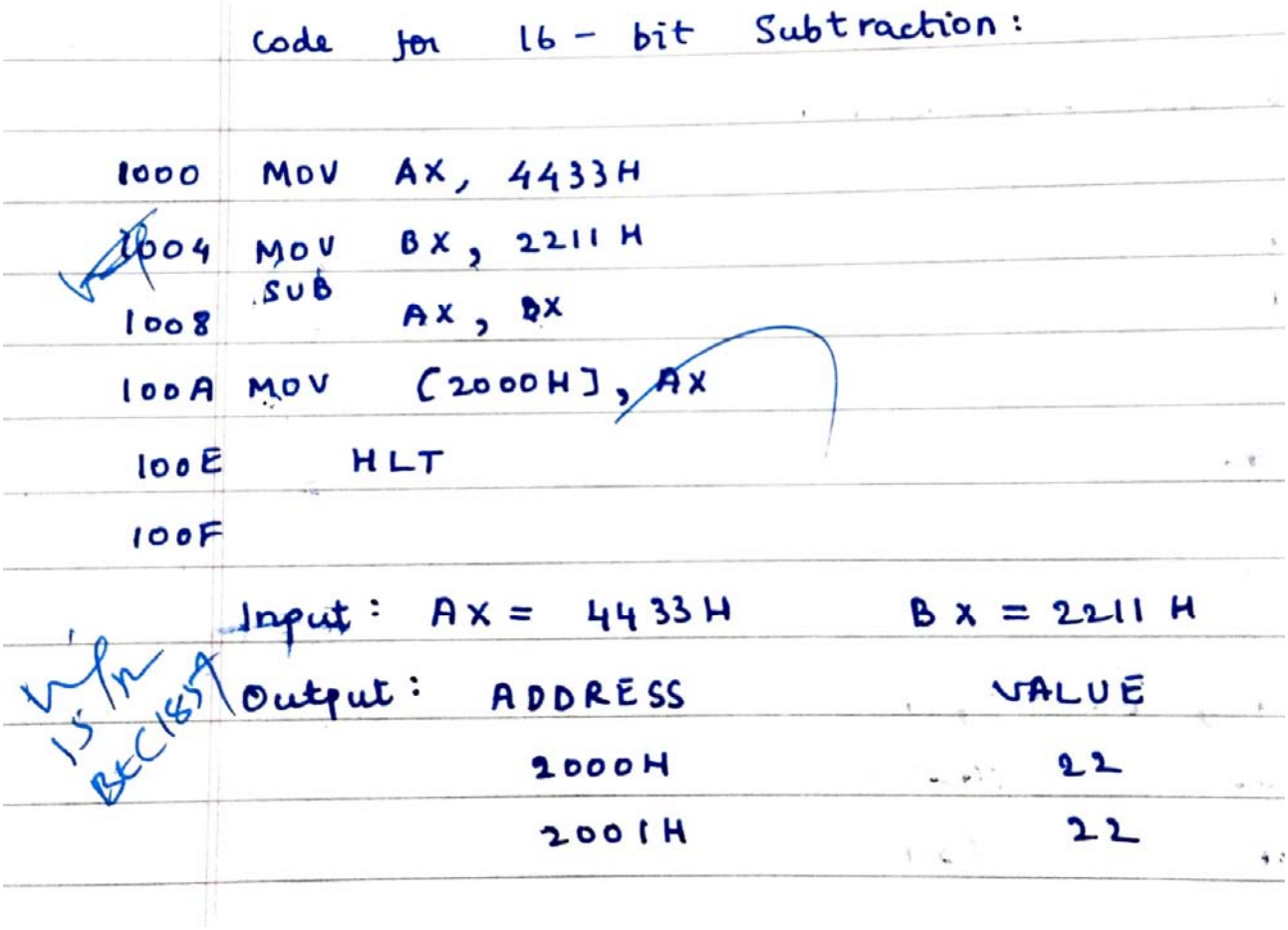
ADDRESS	MEMONICS	COMMENTS
1000	MOV AX,4433H	Move data 4433H to AX register
1004	MOV BX,2211H	Move data 2211H to BX register
1008	SUB AX, BX	Subtract AX and BX content
100A	MOV [2000H], AX	Move AX to the memory location 2000H
100E	HLT	Halt the program

OUTPUT:

Address	Value
2000H	44

2001H	66

OUTPUT VERIFICATION:



INFERENCE:

The output is found out to be “2222H” and the lower byte is stored in the address 2000H and higher byte is stored in the address 2001H.

RESULT:

Thus the 8086 Assembly language program to subtract two 16-bit number in AL, BL registers and store the result in memory location 2000H is written and executed successfully.

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

BECE204P – MICROPROCESSORS AND MICROCONTROLLERS LAB RECORD

Submitted By

21BEC1851 – Rahul Karthik S

Submitted To

Dr. Prakash V

LAB – 02: 8086: Arithmetic Operations

(Multiplication and Division)

LABTASK – 1: 8 – BIT MULTIPLICATION

AIM:

To write 8086 Assembly language to multiply two 8-bit number in AL, BL registers and store the result in memory location 2000H.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

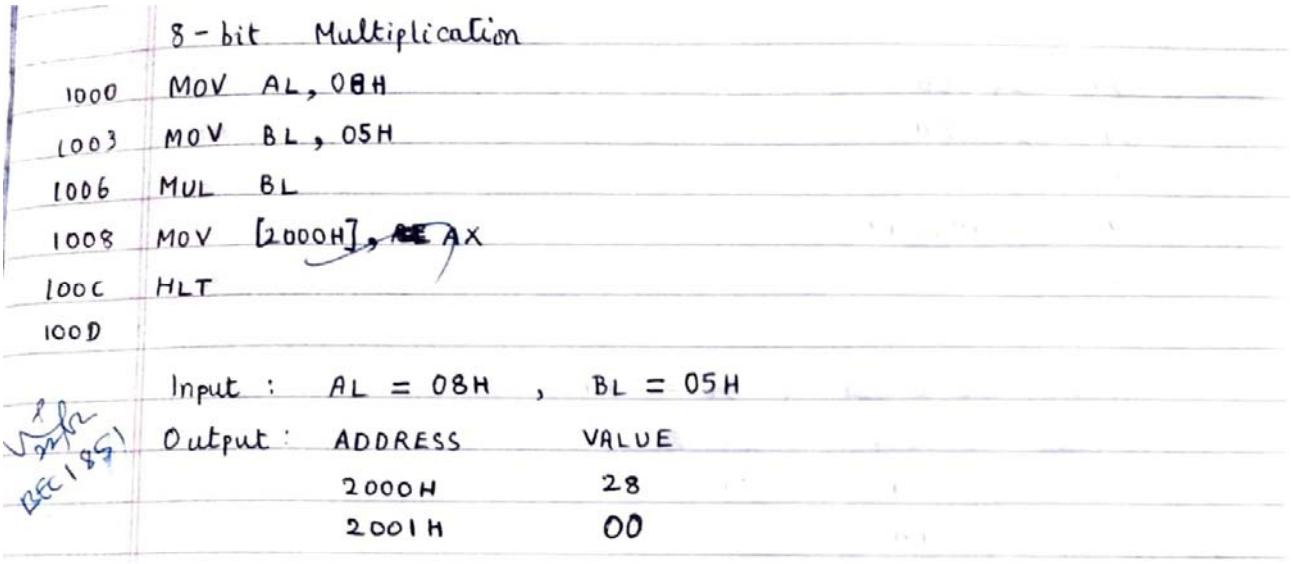
ADDRESS	MEMONICS	COMMENTS
1000	MOV AL,08H	Move data 08H to AL register
1003	MOV BL,05H	Move data 05H to BL register
1006	MUL BL	Multiply AL and BL content

1008	MOV [2000H], AX	Move AX to memory location 2000H
100C	HLT	Halt the program

OUTPUT:

Address	Value
2000H	28H
2001H	00H

OUTPUT VERIFICATION:



INFERENCE:

The output is found out to be “28H” and stored in the address 2000H.

RESULT:

Thus the 8086 Assembly language program to multiply two 8-bit number in AL, BL registers and store the result in memory location 2000H is written and executed successfully.

LABTASK-2: 16 – BIT MULTIPLICATION

AIM:

To write 8086 Assembly language to multiply two 16-bit number in AX, BX registers and store the result in memory location 2000H.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV AX,4433H	Move data 08H to AX register
1004	MOV BX,2211H	Move data 05H to BX register
1008	MUL BX	Multiply AX and BX content
100A	MOV [2000H], AX	Move AX to memory location 2000H
100E	MOV [2002H], DX	Move DX to memory location 2002H
1012	HLT	Halt the program

OUTPUT:

Address	Value
2000H	63H
2001H	4DH
2002H	13H
2003H	09H

OUTPUT VERIFICATION:

16- bit Multiplication

```
1000  MOV  AX, 4433H
1004  MOV  BX, 2211H
1008  MUL  BX
100A  MOV  [2000H], AX
100E  MOV  [2002H], DX
1012  HLT
```

Input : AX = 4433H , BX = 2211H

Output :

ADDRESS	VALUE
2000H	63
2001H	4D
2002H	13
2003H	09

INFERENCE:

The output is found out to be “0913 4D63H” and lower byte of AX is stored in the address 2000H, higher byte of AX is stored in the address 2001H, lower byte of DX is stored in the address 2002H and higher byte of DX is stored in the address 2003H.

RESULT:

Thus the 8086 Assembly language program to multiply two 16-bit number in AX, BX registers and store the result in memory location 2000H is written and executed successfully.

LABTASK – 3: 8 – BIT DIVISION

AIM:

To write 8086 Assembly language to divide two 8-bit number in AL, BL registers and store the result in memory location 2000H.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

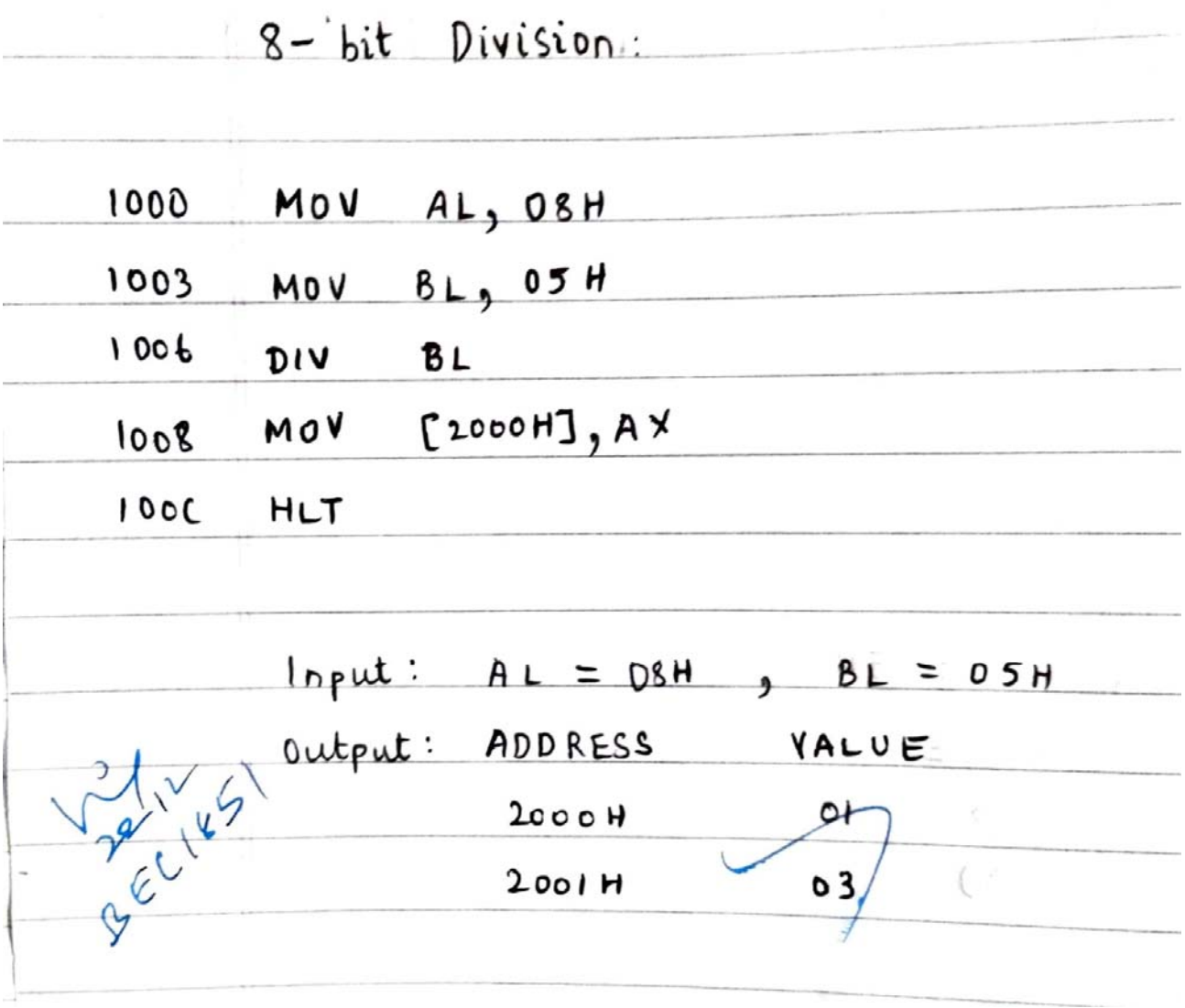
ADDRESS	MEMONICS	COMMENTS
1000	MOV AL,08H	Move data 08H to AL register
1003	MOV BL,05H	Move data 05H to BL register
1006	DIV BL	Divide AL and BL content
1008	MOV [2000H], AX	Move AX to memory location 2000H
100C	HLT	Halt the program

OUTPUT:

Address	Value

2000H	01H
2001H	03H

OUTPUT VERIFICATION:



INFERENCE:

The quotient is found out to be “01H” and stored in the address 2000H and the remainder is found out to be “03H” and stored in the address 2001H.

RESULT:

Thus the 8086 Assembly language program to divide two 8-bit number in AL, BL registers and store the result in memory location 2000H is written and executed successfully.

LABTASK – 4: 16- BIT DIVISION

AIM:

To write 8086 Assembly language to divide two 16-bit number in AX, BX registers and store the result in memory location 2000H.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV AL,4433H	Move data 08H to AL register
1004	MOV BL,2211H	Move data 05H to BL register
1008	DIV BX	Multiply AL and BL content
100A	MOV [2000H], AX	Move AX to memory location 2000H
100E	MOV [2002H], DX	Move DX to memory location 2002H
1012	HLT	Halt the program

OUTPUT:

Address	Value
2000H	02H
2001H	00H
2002H	11H
2003H	00H

OUTPUT VERIFICATION:

16- bit Division :

```
1000   MOV   AX, 4433H
1004   MOV   BX, 2211H
1008   DIV   BX
100A   MOV   [2000H], AX
100E   MOV   [2002H], DX
1012   HLT
```

Input : AX = 4433H BX = 2211H

Output :

ADDRESS	VALUE
2000H	02
2001H	00
2002H	11
2003	00

INFERENCE:

The lower byte of quotient is found out to be “02H”, higher byte of quotient is found out to be “00H” and stored in the address 2000H and 2001H respectively. The lower byte of remainder is found out to be “11H”, higher byte of remainder is found out to be “00H” and stored in the address 2002H and 2003H respectively.

RESULT:

Thus the 8086 Assembly language program to divide two 16-bit number in AX, BX registers and store the result in memory location 2000H is written and executed successfully.

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

BECE204P – MICROPROCESSORS AND MICROCONTROLLERS LAB RECORD

Submitted By

21BEC1851 – Rahul Karthik S

Submitted To

Dr. Prakash V

LAB – 03: 8086: Sum and Average of "N" 8-bit Numbers, Factorial and Celcius to Fahrenheit

LABTASK – 1: SUM OF "N" 8-BIT NUMBER

AIM:

To write 8086 Assembly language to find sum of "N" 8-bit number and store the result in memory location 0600H.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

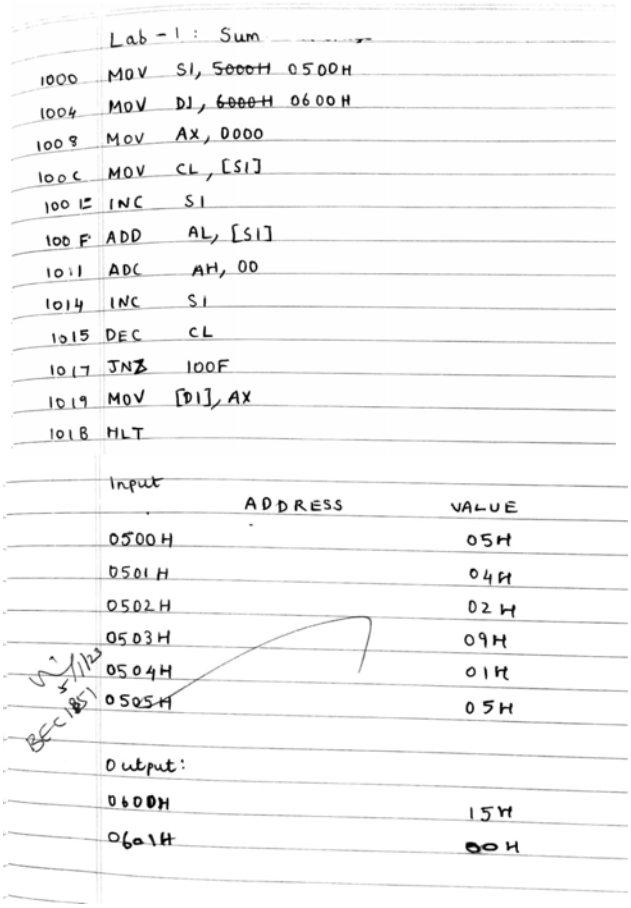
PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 0500H	Load 0500H into Source Index Register
1004	MOV DI, 0600H	Load 0600H into Destination Index Register
1008	MOV AX, 0000	Clear AX register
100C	MOV CL, [SI]	Load the block size (Value of N)
100E	INC SI	Increment SI to point next memory location
100F	ADD AL, [SI]	Add AL and data pointed by SI register
1011	ADC AH, 00	Add AH and 00H along with carry
1014	INC SI	Increment SI to point next memory location
1015	DEC CL	Decremet CL value by 1
1017	JNZ 100F	If Z=0, jump to BACK label
1019	MOV [DI], AX	Store the result of the division into the memory location pointed by DI
101B	HLT	Stop the execution

OUTPUT:

Address	Value
0600H	15H
0601H	00H

OUTPUT VERIFICATION:



Lab - 1 : Sum

```
1000  MOV  SI, 5000H 0500H
1004  MOV  DI, 6000H 0600H
1008  MOV  AX, 0000
100C  MOV  CL, [SI]
100E  INC  SI
100F  ADD  AL, [SI]
1011  ADC  AH, 00
1014  INC  SI
1015  DEC  CL
1017  JNZ  100F
1019  MOV  [DI], AX
101B  HLT
```

Input

ADDRESS	VALUE
0500H	05H
0501H	04H
0502H	02H
0503H	09H
0504H	01H
0505H	05H

Output:

0600H	15H
0601H	00H

INFERENCE:

The output is found out to be “15H” and stored in the address 0600H.

RESULT:

Thus the 8086 Assembly language program to find the sum of “N” 8-bit number is written and executed successfully.

LABTASK – 2: AVERAGE OF "N" 8-BIT NUMBER

AIM:

To write 8086 Assembly language to find average of "N" 8-bit number and store the result in memory location 0600H.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 0500H	Load 0500H into Source Index Register
1004	MOV DI, 0600H	Load 0600H into Destination Index Register
1008	MOV AX, 0000H	Clear AX register
100C	MOV CL, [SI]	Load the block size (Value of N)
100E	MOV BL, CL	Also store into BL
1010	INC SI	Increment SI to point next memory location
1011	ADD AL, [SI]	Add AL and data pointed by SI register
1013	ADC AH, 00H	Add AH and 00H along with carry
1016	INC SI	Increment SI to point next memory location
1017	DEC CL	Decremet CL value by 1
1019	JNZ BACK	If Z=0, jump to BACK label
101B	DIV BL	Otherwise divide it with BL
101D	MOV [DI], AX	Store the result of the division into the memory location pointed by DI
101F	HLT	Stop the execution

OUTPUT:

Address	Value
0600H	04H
0601H	01H

OUTPUT VERIFICATION:

Average :

```
1000  MOV  SI, 0500H
1004  MOV  DI, 0600H
1008  MOV  AX, 0000
100C  MOV  ~~AX~~ CL, [SI]
100E  MOV  BL, CL
1010  INC  SI
1011  ADD  AL, [SI]
1013  ADC  AH, 00H
1016  INC  SI
1017  DEC  CL
1019  JNZ  B
101B  DIV  BL
101D  MOV  [DI], AX
101F  HLT
1020
```

Input:	Value
0500H	05H
0501H	04H
0502H	02H
0503H	09H
0504H	01H
0505H	05H

5/1/23 BEC1851

Output:

0601H	04H	(Q)
0602H	01H	(R)

INFERENCE:

The output is found out to be “04H” (Quotient) and “01H” (Remainder) stored in the address 0600H and 0601H respectively.

RESULT:

Thus the 8086 Assembly language program to find the average of “N” 8-bit number is written and executed successfully.

LABTASK – 3: FACTORIAL

AIM:

To write 8086 Assembly language to find the factorial and to store in the memory location 0400H.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

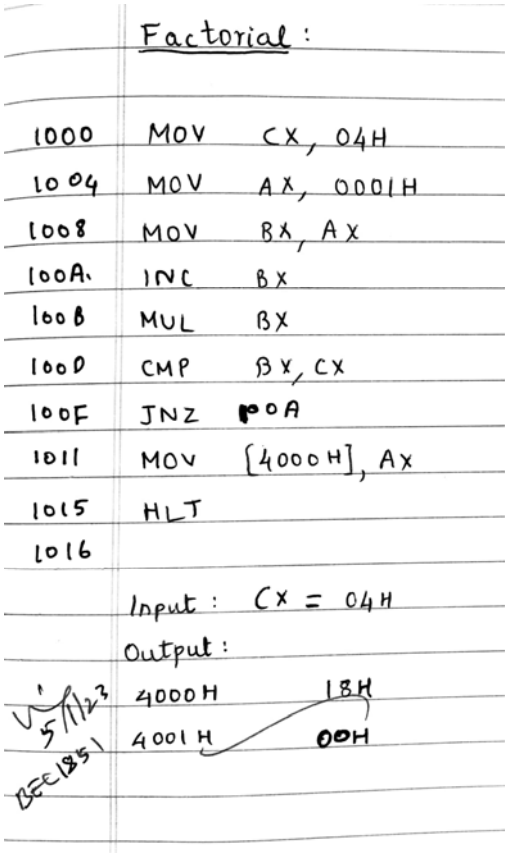
PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV CX, 04H	Load number whose factorial is to be found I CX
1004	MOV AX, 0001H	Load AX with 0001H
1008	MOV BX, AX	Copy AX into BX
100A	INC BX	Increment BX value by 1
100B	MUL BX	Multiply AX*BX = DX:AX
100D	CMP BX, CX	Compare BX with CX i.e BX-CX
100F	JNZ BACK	If Z is not zero, jump to BACK Label
1011	MOV [4000H], AX	Store AX register content to memory location 4000H
1015	HLT	Stop the execution

OUTPUT:

Address	Value
4000H	18H
4001H	00H

OUTPUT VERIFICATION:



INFERENCE:

The value is found out to be “18H” and stored in the address 4000H.

RESULT:

Thus the 8086 Assembly language program to find factorial is written and executed successfully.

LABTASK – 4: CELCIUS TO FAHRENHEIT

AIM:

To write 8086 Assembly language to convert from celcius to fahreheit and store the result in memory location 2000H.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

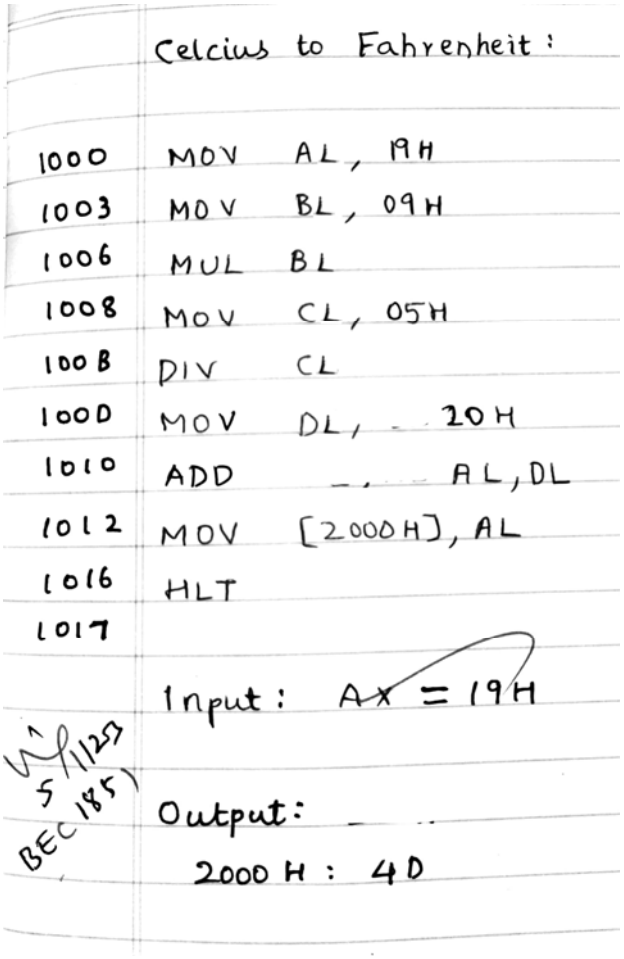
ADDRESS	MEMONICS	COMMENTS
1000	MOV AL, 19H	Load 25 degree (19 H) Celsius value into AL register
1003	MOV BL, 09H	Load BL with 09H
1006	MUL BL	Multiply BL with 25 degree celsius value and store the result in AX
1008	MOV CL, 05H	Move 05H to CL register
100B	DIV CL	Divide AX by CL, Quotient in AL and remainder in AH.
100D	MOV DL, 20H	Move 20H to DL register
1010	ADD AL, DL	Add DL along with AL (Quotient)

1012	MOV [2000H], AL	Store the final result available in AL into memory location 2000H
1016	HLT	Stop the execution

OUTPUT:

Address	Value
2000H	4DH

OUTPUT VERIFICATION:



INFERENCE:

The output "4DH" is stored in the memory location 2000H.

RESULT:

Thus the 8086 Assembly language program to convert from celcius to fahrenheit is written and executed successfully.

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

BECE204P – MICROPROCESSORS AND MICROCONTROLLERS LAB RECORD

Submitted By

21BEC1851 – Rahul Karthik S

Submitted To

Dr. Prakash V

LAB – 04: 8086: Smallest and Largest Number in an Array, Sorting an array in ascending and decending order

LABTASK – 1: SMALLEST NUMBER IN AN ARRAY

AIM:

To write 8086 Assembly language to find smallest number in an array.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type “A” in the keyboard and press enter. “Line assembler” will be displayed.
5. Starting address will be displayed in the kit. Type “1000” as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type “GO” [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. “executing” message will be displayed.
11. Press reset in the kit.
12. Give “SB” [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 0500H	Load 0500H into Source Index Register
1004	MOV DI, 0600H	Load 0600H into Destination Index Register
1008	MOV CL, [SI]	Load the block size (Value of N)
100A	INC SI	Increment SI to point next memory location
100B	MOV AL, [SI]	Add AL and data pointed by SI register
101D	DEC CL	Decrement CL value by 1
101F	INC SI	Increment SI to point next memory location
1010	MOV BL, [SI]	Get next byte of the array in BL register
1012	CMP AL, BL	Compare smallest data AL with next byte BL of the array
1014	JC 1018	If C=1, jump to 1018
1016	MOV AL, BL	Move BL to AL
1018	DEC CL	Decrement CL value by 1
101A	JNZ 100F	If Z=0, then go to 100F otherwise go to next step
101C	MOV [DI], AL	Move smallest data AL to DI
101E	HLT	Stop the execution

OUTPUT:

Address	Value
0600H	01H

OUTPUT VERIFICATION:

Lab Task 1 : Smallest Number

Address	Memonics
1000	MOV SI, 0500H
1004	MOV DI, 0600H
1008	MOV CL, [SI]
100A	INC SI
100B	MOV AL, [SI]
100D	DEC CL
100F	INC SI
1010	MOV BL, [SI]
1012	CMP AL, BL
1014	JC 1018
1016	MOV AL, BL
1018	DEC CL
101A	JNZ 100F
101C	MOV [DI], AL
101E	HLT
101F	

Input :

ADDRESS	Value
0500H	05
0501H	03
0502H	05
0503H	01
0504H	02
0505H	04

Output :

ADDRESS	VALUE
0600H	01

T.B 21BEC1851

INFERENCE:

The lowest value in the given array is “01H” which is stored in the memory 0600H.

RESULT:

Thus, the smallest number in an array code is written and executed successfully.

LABTASK – 2: LARGEST NUMBER IN AN ARRAY

AIM:

To write 8086 Assembly language to find largest number in an array.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type “A” in the keyboard and press enter. “Line assembler” will be displayed.
5. Starting address will be displayed in the kit. Type “1000” as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type “GO” [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. “executing” message will be displayed.
11. Press reset in the kit.
12. Give “SB” [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 0500H	Load 0500H into Source Index Register
1004	MOV DI, 0600H	Load 0600H into Destination Index Register
1008	MOV CL, [SI]	Load the block size (Value of N)
100A	INC SI	Increment SI to point next memory location
100B	MOV AL, [SI]	Add AL and data pointed by SI register
101D	DEC CL	Decrement CL value by 1
101F	INC SI	Increment SI to point next memory location
1010	MOV BL, [SI]	Get next byte of the array in BL register
1012	CMP AL, BL	Compare smallest data AL with next byte BL of the array
1014	JNC 1018	If C=0, jump to 1018
1016	MOV AL, BL	Move BL to AL
1018	DEC CL	Decrement CL value by 1
101A	JNZ 100F	If Z=0, then go to 100F otherwise go to next step
101C	MOV [DI], AL	Move smallest data AL to DI
101E	HLT	Stop the execution

OUTPUT:

Address	Value
0600H	05H

OUTPUT VERIFICATION:

Lab Task-2 : Largest Number

Address	Memonics
1000	MOV SI, 0500H
1004	MOV DI, 0600H
1008	MOV CL, [SI]
100A	INC SI
100B	MOV AL, [SI]
100D	DEC CL
100F	INC SI
1010	MOV BL, [SI]
1012	CMP AL, BL
1014	JNC 1018
1016	MOV AL, BL
1018	DEC CL
101A	JNZ 100F
101C	MOV [DI], AL
101E	HLT
101F	

Input :

ADDRESS	Value
0500H	05
0501H	03
0502H	05
0503H	01
0504H	02
0505H	04

Output :

ADDRESS	VALUE
0600H	: 05

T. Vijay
21BEC1851

INFERENCE:

The lowest value in the given array is “05H” which is stored in the memory 0600H.

RESULT:

The largest value in an array code is written and executed successfully.

LABTASK – 3: SORTING AN ARRAY IN ASCENDING ORDER

AIM:

To write 8086 Assembly language to sort the array in the ascending order.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 0500H	Set SI as a pointer for array
1004	MOV DI, 0600H	Set CL register as count for N-1 comparison (outer loop)
1006	DEC CL	Decrement CL register by 1
1008	MOV SI, 0500H	Initialize SI register as array pointer
100C	MOV CH, [SI]	Load CH and the value of SI
100E	DEC CH	Decrement CH value by 1
1010	INC SI	Increment the array pointer
1011	MOV AL, [SI]	Get the first element of array in AL register
1013	CMP AL, [SI]	Compare the next element of the array with AL

1016	JC 101D	Check carry flag if C = 1 go to 101D otherwise go to next step
1018	XCHG AL, [SI]	Exchange the content of memory pointer by SI and the content of memory location i.e. AL
101A	XCHG AL, [SI-1]	Exchange the content of memory pointer by SI - 1 and the content of memory location i.e. AL
101D	DEC CH	Decrement the count for comparison (CH register)
101F	JNZ 1011	Check zero flag if Z = 0 go to 1011 otherwise go to next step
1021	DEC CL	Decrement the count for repetition
1023	JNZ 1008	Check the zero flag if Z = 0, go to 1008 Otherwise go to next step
1025	HLT	Halt the execution

OUTPUT:

Address	**Value**
0500H	05H
0501H	01H
0502H	02H
0503H	03H
0504H	04H
0505H	05H

OUTPUT VERIFICATION:

Lab Task 3 : Sorting an array in ascending order

Address	Mnemonics
1000	MOV SI, 1100H
1004	MOV CL, [SI]
1006	DEC CL
1009	MOV SI, 1100H
100C	MOV CH, [SI]
100E	DEC CH
1010	INC SI
1011	MOV AL, [SI]
1013	INC SI
1014	CMP AL, [SI]
1016	JC ~~AHEAD~~ ~~1018~~ 101D
1018	XCHG AL, [SI]
101A	XCHG AL, [SI-1]
101D	DEC CH
101F	JNZ 1011
1021	DEC CL
1023	JNZ 1008
1025	HLT
1026	

Input :

0500 : 05
0501 : 03
0502 : 05
0503 : 01
0504 : 02
0505 : 04

Output :

0500 : 05
0501 : 01
0502 : 02
0503 : 03
0504 : 04
0505 : 05

T. Bhanu
21BEC1851

INFERENCE:

The array is sorted and stored in the address 0500H.

RESULT:

Thus the 8086 Assembly language program to sort the array in ascending order is written and executed successfully.

LABTASK – 4: SORTING AN ARRAY IN DESCENDING ORDER

AIM:

To write 8086 Assembly language to sort the array in the descending order.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 0500H	Set SI as a pointer for array
1004	MOV DI, 0600H	Set CL register as count for N-1 comparison (outer loop)
1006	DEC CL	Decrement CL register by 1
1008	MOV SI, 0500H	Initialize SI register as array pointer
100C	MOV CH, [SI]	Load CH and the value of SI
100E	DEC CH	Decrement CH value by 1
1010	INC SI	Increment the array pointer
1011	MOV AL, [SI]	Get the first element of array in AL register
1013	CMP AL, [SI]	Compare the next element of the array with AL

1016	JNC 101D	Check carry flag if C = 0 go to 101D otherwise go to next step
1018	XCHG AL, [SI]	Exchange the content of memory pointer by SI and the content of memory location i.e. AL
101A	XCHG AL, [SI-1]	Exchange the content of memory pointer by SI - 1 and the content of memory location i.e. AL
101D	DEC CH	Decrement the count for comparison (CH register)
101F	JNZ 1011	Check zero flag if Z = 0 go to 1011 otherwise go to next step
1021	DEC CL	Decrement the count for repetition
1023	JNZ 1008	Check the zero flag if Z = 0, go to 1008 Otherwise go to next step
1025	HLT	Halt the execution

OUTPUT:

Address	**Value**
0500H	05H
0501H	05H
0502H	04H
0503H	03H
0504H	02H
0505H	01H

OUTPUT VERIFICATION:

Lab Task 4 · Sorting an array in descending order

Address	Mnemonics
1000	MOV SI, 1100 H
1004	MOV CL, [SI]
1006	DEC CL
1008	MOV SI, 1100 H
100C	MOV CH, [SI]
100E	DEC CH
1010	INC SI
1011	MOV AL, [SI]
1013	INC SI
1014	CMP AL, [SI]
1016	JNC 101D
1018	XCHG AL, [SI]
101A	XCHG AL, [SI-1]
101D	DEC CH
101F	JNZ 1011
1021	DEC CL
1023	JNZ 1008
1025	HLT
1026	

T. Bala 21BEC1851

Input:	Output:
0500 : 05	0500 : 05
0501 : 05	0501 : 05
0502 : 04	0502 : 04
0503 : 03	0503 : 03
0504 : 02	0504 : 02
0505 : 01	0505 01

INFERENCE:

The array is sorted and stored in the address 0500H.

RESULT:

Thus the 8086 Assembly language program to sort the array in decending order is written and executed successfully.

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

BECE204P – MICROPROCESSORS AND MICROCONTROLLERS LAB RECORD

Submitted By

21BEC1851 – Rahul Karthik S

Submitted To

Dr. Prakash V

LAB – 05: 8086: Smallest and Largest Number in an Array, Sorting an array in ascending and decending order

LABTASK – 1: SMALLEST NUMBER IN AN ARRAY

AIM:

To write 8086 Assembly language to find smallest number in an array.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type “A” in the keyboard and press enter. “Line assembler” will be displayed.
5. Starting address will be displayed in the kit. Type “1000” as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type “GO” [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. “executing” message will be displayed.
11. Press reset in the kit.
12. Give “SB” [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 0500H	Load 0500H into Source Index Register
1004	MOV DI, 0600H	Load 0600H into Destination Index Register
1008	MOV CL, [SI]	Load the block size (Value of N)
100A	INC SI	Increment SI to point next memory location
100B	MOV AL, [SI]	Add AL and data pointed by SI register
101D	DEC CL	Decrement CL value by 1
101F	INC SI	Increment SI to point next memory location
1010	MOV BL, [SI]	Get next byte of the array in BL register
1012	CMP AL, BL	Compare smallest data AL with next byte BL of the array
1014	JC 1018	If C=1, jump to 1018
1016	MOV AL, BL	Move BL to AL
1018	DEC CL	Decrement CL value by 1
101A	JNZ 100F	If Z=0, then go to 100F otherwise go to next step
101C	MOV [DI], AL	Move smallest data AL to DI
101E	HLT	Stop the execution

OUTPUT:

Address	Value
0600H	01H

OUTPUT VERIFICATION:

Lab Task 1 : Smallest Number

Address	Memonics
1000	MOV SI, 0500H
1004	MOV DI, 0600H
1008	MOV CL, [SI]
100A	INC SI
100B	MOV AL, [SI]
100D	DEC CL
100F	INC SI
1010	MOV BL, [SI]
1012	CMP AL, BL
1014	JC 1018
1016	MOV AL, BL
1018	DEC CL
101A	JNZ 100F
101C	MOV [DI], AL
101E	HLT
101F	

Input :

ADDRESS	Value
0500H	05
0501H	03
0502H	05
0503H	01
0504H	02
0505H	04

Output :

ADDRESS	VALUE
0600H	01

T.B 21BEC1851

INFERENCE:

The lowest value in the given array is “01H” which is stored in the memory 0600H.

RESULT:

Thus, the smallest number in an array code is written and executed successfully.

LABTASK – 2: LARGEST NUMBER IN AN ARRAY

AIM:

To write 8086 Assembly language to find largest number in an array.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type “A” in the keyboard and press enter. “Line assembler” will be displayed.
5. Starting address will be displayed in the kit. Type “1000” as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type “GO” [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. “executing” message will be displayed.
11. Press reset in the kit.
12. Give “SB” [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 0500H	Load 0500H into Source Index Register
1004	MOV DI, 0600H	Load 0600H into Destination Index Register
1008	MOV CL, [SI]	Load the block size (Value of N)
100A	INC SI	Increment SI to point next memory location
100B	MOV AL, [SI]	Add AL and data pointed by SI register
101D	DEC CL	Decrement CL value by 1
101F	INC SI	Increment SI to point next memory location
1010	MOV BL, [SI]	Get next byte of the array in BL register
1012	CMP AL, BL	Compare smallest data AL with next byte BL of the array
1014	JNC 1018	If C=0, jump to 1018
1016	MOV AL, BL	Move BL to AL
1018	DEC CL	Decrement CL value by 1
101A	JNZ 100F	If Z=0, then go to 100F otherwise go to next step
101C	MOV [DI], AL	Move smallest data AL to DI
101E	HLT	Stop the execution

OUTPUT:

Address	Value
0600H	05H

OUTPUT VERIFICATION:

Lab Task-2 : Largest Number

Address	Memonics
1000	MOV SI, 0500H
1004	MOV DI, 0600H
1008	MOV CL, [SI]
100A	INC SI
100B	MOV AL, [SI]
100D	DEC CL
100F	INC SI
1010	MOV BL, [SI]
1012	CMP AL, BL
1014	JNC 1018
1016	MOV AL, BL
1018	DEC CL
101A	JNZ 100F
101C	MOV [DI], AL
101E	HLT
101F	

Input :

ADDRESS	Value
0500H	05
0501H	03
0502H	05
0503H	01
0504H	02
0505H	04

Output :

ADDRESS	VALUE
0600H	: 05

T. Vijay
21BEC1851

INFERENCE:

The lowest value in the given array is “05H” which is stored in the memory 0600H.

RESULT:

The largest value in an array code is written and executed successfully.

LABTASK – 3: SORTING AN ARRAY IN ASCENDING ORDER

AIM:

To write 8086 Assembly language to sort the array in the ascending order.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 0500H	Set SI as a pointer for array
1004	MOV DI, 0600H	Set CL register as count for N-1 comparison (outer loop)
1006	DEC CL	Decrement CL register by 1
1008	MOV SI, 0500H	Initialize SI register as array pointer
100C	MOV CH, [SI]	Load CH and the value of SI
100E	DEC CH	Decrement CH value by 1
1010	INC SI	Increment the array pointer
1011	MOV AL, [SI]	Get the first element of array in AL register
1013	CMP AL, [SI]	Compare the next element of the array with AL

1016	JC 101D	Check carry flag if C = 1 go to 101D otherwise go to next step
1018	XCHG AL, [SI]	Exchange the content of memory pointer by SI and the content of memory location i.e. AL
101A	XCHG AL, [SI-1]	Exchange the content of memory pointer by SI - 1 and the content of memory location i.e. AL
101D	DEC CH	Decrement the count for comparison (CH register)
101F	JNZ 1011	Check zero flag if Z = 0 go to 1011 otherwise go to next step
1021	DEC CL	Decrement the count for repetition
1023	JNZ 1008	Check the zero flag if Z = 0, go to 1008 Otherwise go to next step
1025	HLT	Halt the execution

OUTPUT:

Address	**Value**
0500H	05H
0501H	01H
0502H	02H
0503H	03H
0504H	04H
0505H	05H

OUTPUT VERIFICATION:

Lab Task 3 : Sorting an array in ascending order

Address	Mnemonics
1000	MOV SI, 1100H
1004	MOV CL, [SI]
1006	DEC CL
1009	MOV SI, 1100H
100C	MOV CH, [SI]
100E	DEC CH
1010	INC SI
1011	MOV AL, [SI]
1013	INC SI
1014	CMP AL, [SI]
1016	JC ~~AHEAD~~ ~~1018~~ 101D
1018	XCHG AL, [SI]
101A	XCHG AL, [SI-1]
101D	DEC CH
101F	JNZ 1011
1021	DEC CL
1023	JNZ 1008
1025	HLT
1026	

Input :

0500 : 05
0501 : 03
0502 : 05
0503 : 01
0504 : 02
0505 : 04

Output :

0500 : 05
0501 : 01
0502 : 02
0503 : 03
0504 : 04
0505 : 05

T. Bhanu
21BEC1851

INFERENCE:

The array is sorted and stored in the address 0500H.

RESULT:

Thus the 8086 Assembly language program to sort the array in ascending order is written and executed successfully.

LABTASK – 4: SORTING AN ARRAY IN DESCENDING ORDER

AIM:

To write 8086 Assembly language to sort the array in the descending order.

PROCEDURE:

1. Connect the power cord and keyboard with the kit.
2. Switch on the power supply.
3. Press reset in the kit.
4. Type "A" in the keyboard and press enter. "Line assembler" will be displayed.
5. Starting address will be displayed in the kit. Type "1000" as the starting address.
6. Type the program and note the address of each line of the code till HLT.
7. Press reset in the kit.
8. Type "GO" [space] starting address (e.g. 1000) for execution.
9. Press enter in the keyboard.
10. "executing" message will be displayed.
11. Press reset in the kit.
12. Give "SB" [space] memory location (e.g. 2000) for execution.
13. Output will be displayed.

Important terms:

A – Line Assembler

GO – Execution

SB – To view output

U – Disassembly

PROGRAM:

ADDRESS	MEMONICS	COMMENTS
1000	MOV SI, 0500H	Set SI as a pointer for array
1004	MOV DI, 0600H	Set CL register as count for N-1 comparison (outer loop)
1006	DEC CL	Decrement CL register by 1
1008	MOV SI, 0500H	Initialize SI register as array pointer
100C	MOV CH, [SI]	Load CH and the value of SI
100E	DEC CH	Decrement CH value by 1
1010	INC SI	Increment the array pointer
1011	MOV AL, [SI]	Get the first element of array in AL register
1013	CMP AL, [SI]	Compare the next element of the array with AL

1016	JNC 101D	Check carry flag if C = 0 go to 101D otherwise go to next step
1018	XCHG AL, [SI]	Exchange the content of memory pointer by SI and the content of memory location i.e. AL
101A	XCHG AL, [SI-1]	Exchange the content of memory pointer by SI - 1 and the content of memory location i.e. AL
101D	DEC CH	Decrement the count for comparison (CH register)
101F	JNZ 1011	Check zero flag if Z = 0 go to 1011 otherwise go to next step
1021	DEC CL	Decrement the count for repetition
1023	JNZ 1008	Check the zero flag if Z = 0, go to 1008 Otherwise go to next step
1025	HLT	Halt the execution

OUTPUT:

Address	**Value**
0500H	05H
0501H	05H
0502H	04H
0503H	03H
0504H	02H
0505H	01H

OUTPUT VERIFICATION:

Lab Task 4 · Sorting an array in descending order

Address	Mnemonics
1000	MOV SI, 1100 H
1004	MOV CL, [SI]
1006	DEC CL
1008	MOV SI, 1100 H
100C	MOV CH, [SI]
100E	DEC CH
1010	INC SI
1011	MOV AL, [SI]
1013	INC SI
1014	CMP AL, [SI]
1016	JNC 101D
1018	XCHG AL, [SI]
101A	XCHG AL, [SI-1]
101D	DEC CH
101F	JNZ 1011
1021	DEC CL
1023	JNZ 1008
1025	HLT
1026	

T. Bala 21BEC1851

Input:	Output:
0500 : 05	0500 : 05
0501 : 05	0501 : 05
0502 : 04	0502 : 04
0503 : 03	0503 : 03
0504 : 02	0504 : 02
0505 : 01	0505 01

INFERENCE:

The array is sorted and stored in the address 0500H.

RESULT:

Thus the 8086 Assembly language program to sort the array in decending order is written and executed successfully.

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

BECE204P – MICROPROCESSORS AND MICROCONTROLLERS LAB RECORD

Submitted By

21BEC1851 – Rahul Karthik S

Submitted To

Dr. Prakash V

LAB – 06: INTRODUCTION TO KEIL IDE & ASSEMBLY PROGRAMMING WITH ARITHMETIC INSTRUCTION OF 8051

LABTASK-1: SOLVING THE MATHEMATICAL EQUATION

AIM:

To write an 8051 assembly language program to solve the given mathematical equation, W = (Y+3Z-6X)/6D

PROCEDURE:

1. Create a Project (.uvproj)
2. Create and write an assembly program (.a51)
3. Build your project to check errors.
4. Select debug mode.
5. Run your code to verify the output.

PROGRAM:

```
ORG 0000H
        MOV A, #12H
        MOV B, #03H
        MUL AB
        MOV 50H, A
        MOV A, #02H
        MOV B, #06H
        MUL AB
        MOV 51H, A
        MOV A, #00H
        ADD A, #25H
        SUBB A, 51H
        MOV 52H, A
        MOV A, #03H
        MOV B, #06H
        MUL AB
```

```
MOV 53H, A
MOV A, 52H
MOV B, 53H
DIV AB
MOV 33H, A
MOV 34H, B
END
```

OUTPUT:

Address	Value
D:33H	04H
D:34H	07H

OUTPUT VERIFICATION:

1)

```
ORG 0000H
    MOV A, #12H
    MOV B, #03H
    MUL AB
    MOV 50H, A
    MOV A, #02H
    MOV B, #06H
    MOV MUL AB
    MOV 51H, A
    MOV A, #00H
    ADD A, #25H
    SUBB A, 51H
    MOV 52H, A
    MOV A, #03H
    MOV B, #06H
    MUL AB
    MOV 53H, A
    MOV A, 52H
    MOV B, 53H
    DIV AB
    MOV 33H, A
    MOV 34H, B
END
```

Output :

33H : 04

34H : 07

INFERENCE:

The answer for the given mathematical equation is a 16-bit value and it is stored in 33H and 34H memory locations.

RESULT:

Thus the 8051 Assembly language program to solve the given mathematical equation is written and executed successfully.

LABTASK- 2 : SOLVING THE MATHEMATICAL EQUATION

AIM:

To write an 8051 assembly language program to solve the given mathematical equation, $(a-b)^2 = (a^2 + b^2 – 2ab)$

PROCEDURE:

1. Create a Project (.uvproj)
2. Create and write an assembly program (.a51)
3. Build your project to check errors.
4. Select debug mode.
5. Run your code to verify the output.

PROGRAM:

```
ORG 0000H
        MOV 55H, #18H
        MOV 56H, #51H
        MOV A, 55H
        MOV B, 55H
        MUL AB
        MOV 20H, A
        MOV 21H, B
        MOV A, 56H
        MOV B, 56H
        MUL AB
        MOV 22H, A
        MOV 23H, B
        MOV A, #02H
        MOV B, 55H
        MUL AB
        MOV B, 56H
        MUL AB
        MOV 24H, A
        MOV 25H, B
        MOV A, 20H
        MOV B, 22H
```

```
ADD A, B
MOV B, 24H
SUBB A, B
MOV 58H, A
MOV A, 21H
MOV B, 23H
ADDC A, B
MOV B, 25H
SUBB A, B
MOV 57H, A
END
```

OUTPUT:

Address	Value
D:57H	0CH
D:58H	B1H

OUTPUT VERIFICATION:

②

```
ORG   0000H
MOV   55H, #18H
MOV   56H, #51H
MOV   A, 55H
MOV   B, 55H
MUL   AB
MOV   20H, A
MOV   21H, B
MOV   A, 56H
MOV   B, 56H
MUL   AB
MOV   22H, A
MOV   23H, B
MOV   A, #02H
MOV   B, 55H
MUL   AB
MOV   B, 56H
MUL   AB
MOV   24H, A
MOV   25H, B
MOV   A, 20H
MOV   B, 22H
ADD   A, B
MOV   B, 24H
SUBB  A, B
MOV   58H, A
MOV   A, 21H
MOV   B, 23H
ADDC  A, B
MOV   B, 25H
SUBB  A, B
MOV   57H, A
END
```

Output:

Address : Value

57H : 0CH

58H : B1H

INFERENCE:

The answer for the given mathematical equation is a 16-bit value and it is stored in 57H and 58H memory locations.

RESULT:

Thus the 8051 Assembly language program to solve the given mathematical equation is written and executed successfully.

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

BECE204P – MICROPROCESSORS AND MICROCONTROLLERS LAB RECORD

Submitted By

21BEC1851 – Rahul Karthik S

Submitted To

Dr. Prakash V

LAB – 07: ASSEMBLY PROGRAMMING WITH BIT-ORIENTED AND PROGRAM CONTROL INSTRUCTIONS OF 8051

CHALLENGING TASK-1: SUM OF FIRST 10 NATURAL NUMBERS

AIM:

To write an 8051 assembly language program to find sum of first 10 natural numbers.

PROCEDURE:

1. Create a Project (.uvproj)
2. Create and write an assembly program (.a51)
3. Build your project to check errors.
4. Select debug mode.
5. Run your code to verify the output.

PROGRAM:

```
ORG 0000H
        MOV R2, #0AH
        MOV A, #00H
        MOV R0, #00H
        AGAIN: INC R0
        ADD A, R0
        DJNZ R2, AGAIN
        MOV 46H, A
        END
```

OUTPUT:

Address	Value
D:46H	37H

OUTPUT VERIFICATION:

```
ORG   0000H
MOV   R2, #0AH
MOV   A, #00H
MOV   R0, #00H
AGAIN: INC R0
ADD   A, R0
DJNZ  R2, AGAIN
MOV   46H, A
END
```

T.VIGNESH
BEC1851

OUTPUT:

46H : 37

INFERENCE:

The sum of the first 10 natural numbers is stored in the memory location 46H.

RESULT:

Thus the 8051 Assembly language program to find the sum of first 10 natural numbers is written and executed successfully.

CHALLENGING TASK- 2: COMPARISON OF TWO 8- BIT NUMBERS

AIM:

To write an 8051 assembly language program to compare two 8- bit number in external memory location 8000H and 8001H respectively and reflect your result as,

- If NUM1 < NUM2, Set LSB of data RAM location 2FH.
- If NUM1 > NUM2, Set MSB of data RAM location 2FH.
- If NUM1 = NUM2, the CLR both LSB and MSB of 2FH.

PROCEDURE:

1. Create a Project (.uvproj)
2. Create and write an assembly program (.a51)
3. Build your project to check errors.
4. Select debug mode.
5. Run your code to verify the output.

PROGRAM:

```
ORG 0000H
        MOV DPTR, #8000H
        MOVX A, @DPTR
        MOV R0, A
        INC DPTR
        MOVX A, @DPTR
        MOV R1, A
        MOV A, R0
        SUBB A, R1
        JZ EQUAL
        JNC BIG
        SETB 78H
        SJMP END1
        BIG: SETB 7FH
        SJMP END1
        EQUAL: CLR 78H
        CLR 7FH
        END1: NOP
        END
```

OUTPUT 1: (NUM1 > NUM2)

Address	Value
D:2FH	80H

OUTPUT 2: (NUM1 = NUM2)

Address	Value
D:2FH	00H

OUTPUT VERIFICATION:

```
        ORG   0000H

        MOV   DPTR, #8000H
        MOVX  A, @DPTR
        MOV   R0, A
        INC   DPTR
        MOV   A, @DPTR
              MOV R1, A
              MOV A, R0
        SUBB  A, R1
        JZ    EQUAL
        JNC   BIG
        SETB  78H
        SJMP  END1
BIG:    SETB  7FH
        SJMP  END1
EQUAL:  CLR   78H
        CLR   7FH
END1:   ~~SJMP  END1~~ NOP
        END
```

O/P verified

16/1

INFERENCE:

If the NUM1 > NUM2, the output is 80H, if NUM1 < NUM2 the output is 01H and if the NUM1 = NUM2 the output is 00H.

RESULT:

Thus the 8051 Assembly language program to compare the numbers is written and executed successfully.

LABTASK- 1: STORING DATA IN EXTERNAL RAM

AIM:

To write an 8051 assembly language program to move a block of five data starting from RAM 40H to external RAM 3000H onwards and perform complement operation before storing to RAM.

PROCEDURE:

1. Create a Project (.uvproj)
2. Create and write an assembly program (.a51)
3. Build your project to check errors.
4. Select debug mode.
5. Run your code to verify the output.

PROGRAM:

```
ORG 0000H
MOV R0, #40H
MOV DPTR, #3000H
MOV R2, #05H
LOOP: MOV A, @R0
CPL A
MOVX @DPTR, A
INC R0
INC DPTR
DJNZ R2, LOOP
END
```

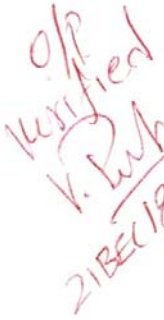
OUTPUT:

Address	Value
X:3000H	FE
X:3001H	FD
X:3002H	FC
X:3003H	FB
X:3004H	FA

OUTPUT VERIFICATION:

Lab Task - 1

```
        ORG   0000H
        MOV   R0, #40H
        MOV   DPTR, #3000H
        MOV   R2, #05H
LOOP:   MOV   A, @R0
        CPL   A
        MOVX  @DPTR, A
        INC   R0
        INC   DPTR
        DJNZ  R2, LOOP
        END
```



INFERENCE:

The data is stored external memory 3000H to 3004H after complementing.

RESULT:

Thus the 8051 Assembly language program to complement and store in external RAM is written and executed successfully.

LABTASK- 2 : SETTING THE PORT

AIM:

To write an 8051 assembly language program to set the port as high if 24H is high and set the port low if 24H is low.

PROCEDURE:

1. Create a Project (.uvproj)
2. Create and write an assembly program (.a51)
3. Build your project to check errors.
4. Select debug mode.
5. Run your code to verify the output.

PROGRAM:

```
ORG 0000H
MOV P1, #00H
AGAIN: MOV C, 24H
JNC NO
SETB P1.0
SJMP AGAIN
NO: CLR P1.0
SJMP AGAIN
END
```

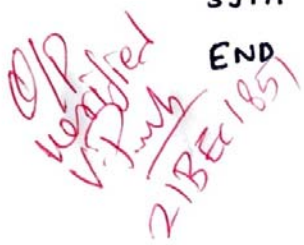
OUTPUT:

Address	Value
D:24H	00H

OUTPUT VERIFICATION:

Lab Task – 2

```
        ORG   0000H
        MOV   P1, #00H
AGAIN:  MOV   C, 24H
        JNC   NO
        SETB  P1.0
        SJMP  AGAIN
NO :    CLR   P1.0
        SJMP  AGAIN
        END
```



INFERENCE:

Thus, the port is set as high if 24H is high and set the port low if 24H is low.

RESULT:

Thus the 8051 Assembly language program to set port low and high is written and executed successfully.

LABTASK- 3 : COMPARE THE VALUES AND STORE IN REGISTER

AIM:

To write an 8051 assembly language program to compare the temperature values and store it in R2 or R3 register according to the following conditions.

PROCEDURE:

1. Create a Project (.uvproj)
2. Create and write an assembly program (.a51)
3. Build your project to check errors.
4. Select debug mode.
5. Run your code to verify the output.

PROGRAM:

```
ORG 0000H
MOV A, 55H
CJNE A, #30H, OVER
SJMP EXIT
OVER: JNC NEXT
MOV R1, A
SJMP EXIT
NEXT: MOV R2, A
EXIT: NOP
END
```

OUTPUT 1: (T < 30H)

Register	Value
R1	25H
A	25H

OUTPUT 2: (T > 30)

Register	Value
R2	35H
A	25H

OUTPUT VERIFICATION:

Lab Task - 3

```
        ORG   0000H
        MOV   A, 55H
        CJNE  A, #30H, OVER
        SJMP  EXIT
OVER :  JNC   NEXT
        MOV   R1, A
        SJMP  EXIT
NEXT:   MOV   R2, A
EXIT :  NOP
        END
```

O/P verified
V Pavi
21BEC1851

INFERENCE:

Thus, the temperature value is compared and if T = 30H, A = 30H, if T < 30H, R1 = T and if T > 30H then R2 = T.

RESULT:

Thus the 8051 Assembly language program to compare the temperature value and store in register is written and executed successfully.

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

BECE204P – MICROPROCESSORS AND MICROCONTROLLERS

LAB RECORD

Submitted By

21BEC1851 – Rahul Karthik S

Submitted To

Dr. Prakash V

LAB – 08: I/O PORT PROGRAMMING IN 8051

CHALLENGING TASK-1:

AIM:

To write an 8051 assembly language program to toggle the status of the LEDs connected at port 1 pins 0 for every 1851 ms.

PROCEDURE:

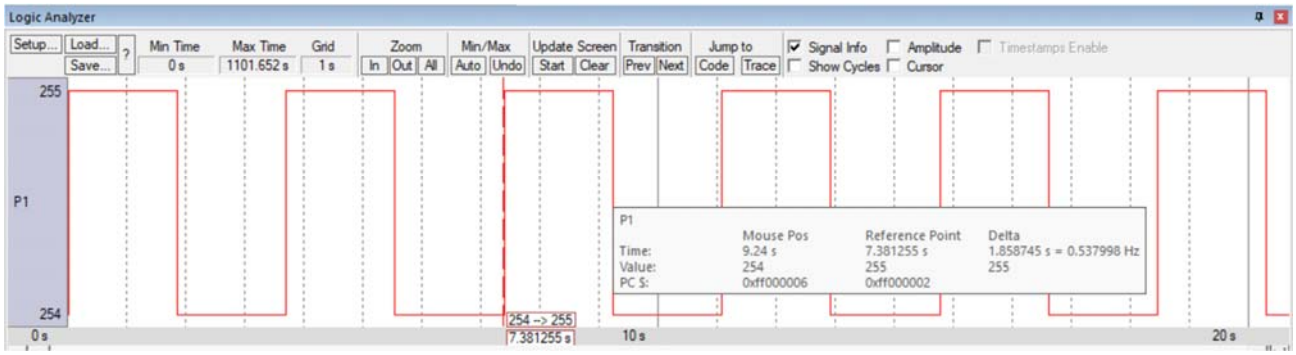
1. Create a Project (.uvproj)
2. Create and write an assembly program (.a51)
3. Build your project to check errors.
4. Select debug mode.
5. Run your code to verify the output.

PROGRAM:

```
ORG 0000H
CLR P1.0
BACK: SETB P1.0
ACALL DELAY
CLR P1.0
SJMP BACK

DELAY: MOV R1, #13
LOOP1: MOV R2, #255
LOOP2: MOV R3, #255
LOOP3: DJNZ R3, LOOP3
DJNZ R2, LOOP2
DJNZ R1, LOOP1
RET
```

OUTPUT:



OUTPUT VERIFICATION:

Challenging Task - 1

```
ORG   0000H
CLR   P1.0
BACK: SETB P1.0
ACALL DELAY
CLR   P1.0
ACALL DELAY
SJMP  BACK

DELAY: MOV  R1, #13
LOOP1: MOV  R2, #255
LOOP2: MOV  R3, #255
~~LOOP3: DJNZ R3, LOOP3 NOP~~
       ~~NOP~~
LOOP3: DJNZ R3, LOOP3
       DJNZ R2, LOOP2
       DJNZ R1, LOOP1
       RET
```

Challenging Task - 1

Crystal Freq = 11.0592

Clock Freq = $\frac{11.0592}{12}$ = 0.9216 MHz

Time Period = $\frac{1}{0.9216}$ = 1.085 μs

~~No. of~~ MOV → 1 MC → 1.085 μs

DJNZ → 2 MC → 2.17 μs

No. of cycles to create delay = $\frac{1851000}{2.17}$

= 852995.4

= 852996

$R_1 = \frac{852996}{255} = 3345.1 \simeq 3346$

$R_2 = \frac{3346}{255} = 13.12 \simeq 13 = R_3$

In loop 3 : 255 X 2 X 1.085 = 553.35 μs

In loop 2 : 255 X 553.35 + 255 (1+2) X 1.085 + 3 X 1.085

= 141934.275 ≃ 141935

In loop 1 : 13 X 141935 + 13 X (1+2) X 1.085 + 3 X 1.085

= 1845200.57 μs

≃ 1.85 s

INFERENCE:

The 8051 Assembly language program is written to generate a time delay of 1.8 seconds.

RESULT:

Thus the 8051 Assembly language program to toggle the status of the LEDs connected at port 1 pins 0 for every 1851 ms is written and executed

LAB TASK-1:

AIM:

To write an 8051 assembly language program to toggle the status of the LEDs connected at port 1 pins 0 for every 1 s.

PROCEDURE:

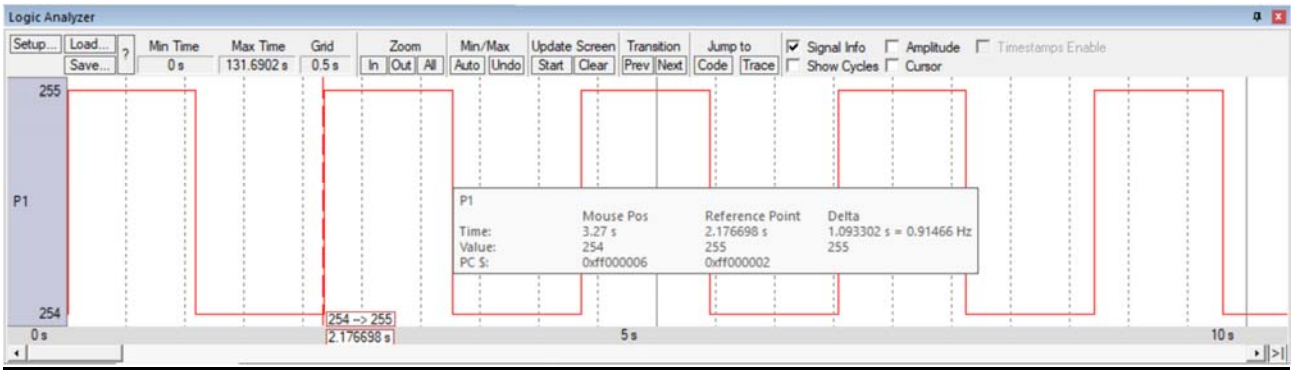
1. Create a Project (.uvproj)
2. Create and write an assembly program (.a51)
3. Build your project to check errors.
4. Select debug mode.
5. Run your code to verify the output.

PROGRAM:

```
ORG 0000H
CLR P1.0
BACK: SETB P1.0
ACALL DELAY
CLR P1.0
SJMP BACK

DELAY: MOV R1, #5
LOOP1: MOV R2, #200
LOOP2: MOV R3, #250
LOOP3: NOP
NOP
DJNZ R3, LOOP3
DJNZ R2, LOOP2
DJNZ R1, LOOP1
RET
```

OUTPUT:



OUTPUT VERIFICATION:

Lab Task - 1 :

```
ORG   0000 H
CLR   P1.0
BACK : SETB  P.10
ACALL  DELAY
CLR   P1.0
ACALL  DELAY
SJMP   BACK

;DELAY
DELAY :  MOV R1, #5
LOOP1 :  MOV  R2, # 200
LOOP2 :  MOV  R3, #250
LOOP3 :  NOP                    ; 1 MC
         NOP                    ; 1 MC
         DJNZ  R3, LOOP3        ; 2 MC
         DJNZ  R2, LOOP2        ; 2 MC
         DJNZ  R1, LOOP1        ; 2 MC
         RET                    ; 2 MC
```

O/P verified

INFERENCE:

The 8051 Assembly language program is written to generate a time delay of 1 seconds.

RESULT:

Thus the 8051 Assembly language program to toggle the status of the LEDs connected at port 1 pins 0 for every 1 s is written and executed

LAB TASK 2:

AIM:

To write an 8051 assembly language program to continuously get the status of the switch and send it to the LED.

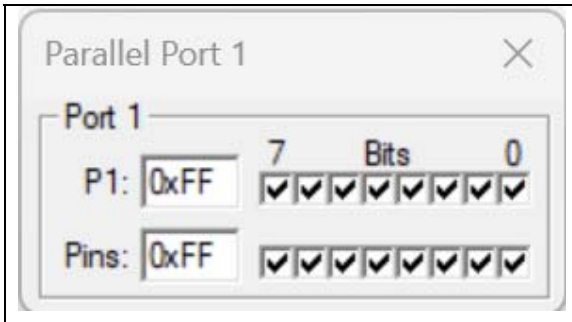
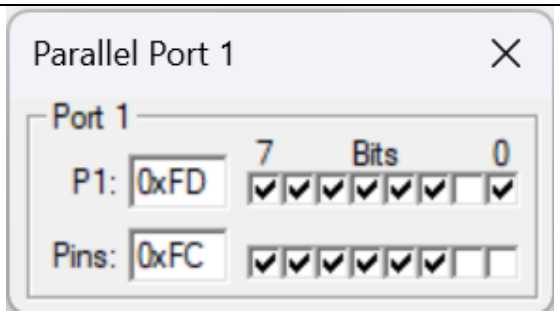
PROCEDURE:

1. Create a Project (.uvproj)
2. Create and write an assembly program (.a51)
3. Build your project to check errors.
4. Select debug mode.
5. Run your code to verify the output.

PROGRAM:

```
ORG 0000H
SETB P1.0
CLR P1.1
HERE: MOV C, P1.0
JC LEDON
CLR P1.1
SJMP HERE
LEDON: SETB P1.1
SJMP HERE
```

OUTPUT:

	
SW is HIGH	SW is LOW

OUTPUT VERIFICATION:

Lab Task - 2

```
        SETB  P1.0
        CLR   P1.1
HERE :  MOV   C, P1.0
        JC    LEDON
        CLR   P1.1
        SJMP  HERE
LEDON:  SETB  P1.1
        SJMP  HERE
```

H/W verified

25/5
BEC1851

INFERENCE:

The 8051 Assembly language program is written to get the switch and send it to LED.

RESULT:

Thus the 8051 Assembly language program to continuously get the status of the switch and send it to the LED.

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering (SENSE)

BECE204P – MICROPROCESSORS AND MICROCONTROLLERS LAB RECORD

Submitted By

21BEC1851 – Rahul Karthik S

Submitted To

Dr. Prakash V

LAB – 11: INTERUPT PROGRAMMING & LCD INTERFACING WITH 8051

LABTASK-1:

AIM:

To write a program using interrupts that continuously get 8-bit date from P0 and sends it to P1 while simultaneously creating a square wave of 200 μs period on P2.1. Use timer 0 to create the square wave. Assume that XTAL = 11.0592 MHz.

PROCEDURE:

1. Create a Project (.uvproj)
2. Create and write an assembly program (.a51)
3. Build your project to check errors.
4. Select debug mode.
5. Run your code to verify the output.

PROGRAM:

```
ORG 0000H
LJMP MAIN
ORG 0080H
CPL P2.1
RETI
ORG 0030H
MAIN: MOV P0, #0FFH
MOV P1, #00H
CLR P2.1
MOV P2.1
MOV TMOD. #02H
MOV TH0, #0AH
MOV IE, #82H
SETB TR0
BACK: MOV A, P0
MOV P1, A
SJMP BACK
END
```

OUTPUT VERIFICATION:

Lab Task-1:

```
ORG   0000H
LJMP  MAIN
ORG   000BH
CPL   P2.1
RETI
MAIN: ORG  0030H
MOV   P0, #0FFH
MOV   P1, #00H
CLR   P2.1
MOV   TMOD, #02H
MOV   TH0, #0A4H
MOV   IE, #82H
SETB  TR0
BACK: MOV  A, P0
MOV   P1, A
SJMP  BACK
END
```

30/3/23
BEC 1851

INFERENCE:

From this we can know that, P0 is an input port whereas P1 and P2 are input port.

RESULT:

Thus the 8051 Assembly language program to use interrupts that continuously get 8-bit date from P0 and sends it to P1 while simultaneously creating a square wave of 200 μs period on P2.1 is written and executed successfully.

LABTASK-2:

AIM:

To turn on LED when external interrupt is low, turn on the LED and to perform toggle operation in P1.5 with the delay of 500 ms.

PROCEDURE:

1. Create a Project (.uvproj)
2. Create and write an assembly program (.a51)
3. Build your project to check errors.
4. Select debug mode.
5. Run your code to verify the output.

PROGRAM:

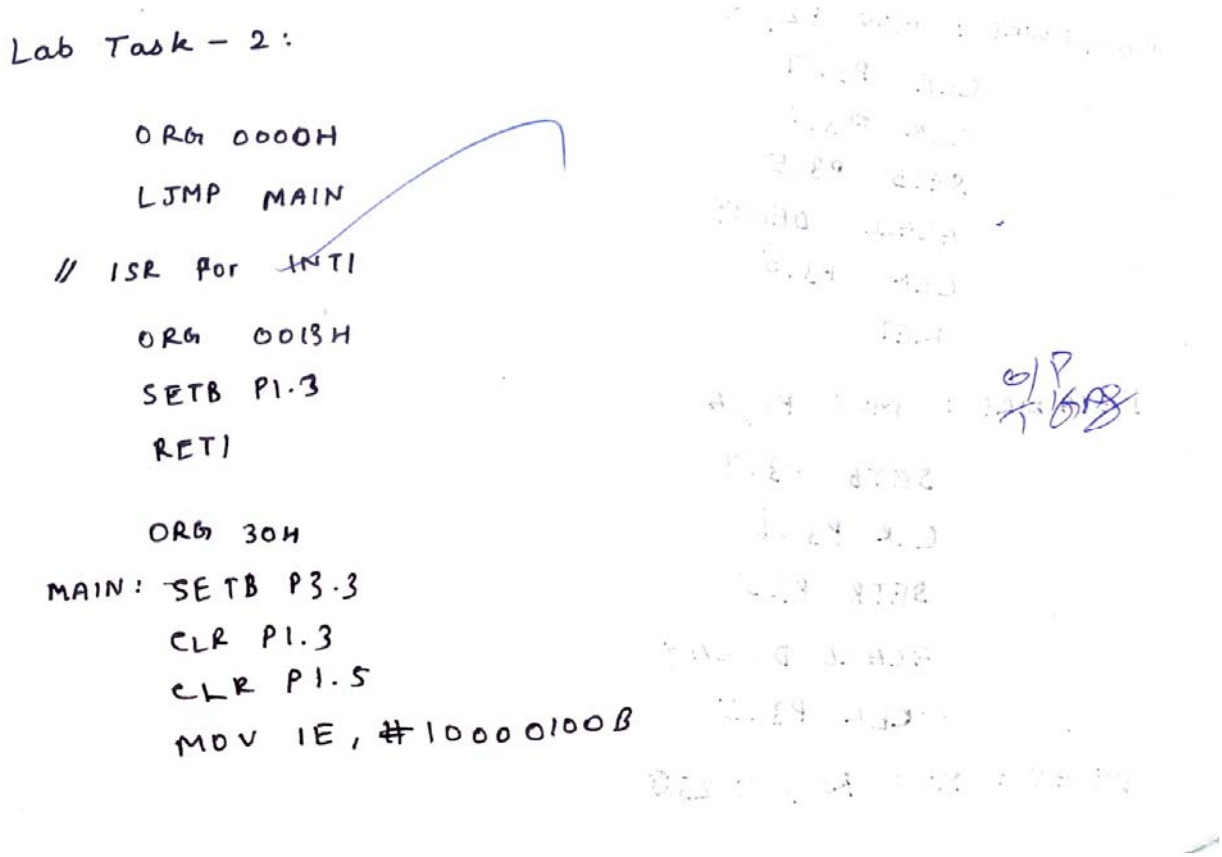
```
ORG 0000H
LJMP MAIN
// ISR for INT1
ORG 0013H
SETB P1.3
RETI
ORG 0030H
MAIN: SETB P3.3
CLR P1.3
CLR P1.5
MOV IE, #10000100B
HERE: CLR P1.3
SETB P1.5
ACALL DELAY
CLR P1.5
ACALL DELAY
SJMP HERE
// DELAY OF 500 ms
DELAY: MOV R2, #04H
HERE3: MOV R1, #0FFH
HERE2: MOV R0, #0FFH
HERE1: DJNZ R0, HERE1
DJNZ R1, HERE1
```

DJNZ R2, HERE2

RET

END

OUTPUT VERIFICATION:



INFERENCE:

The interrupt is given externally from the ESA 8051 kit, and LED is turning on and off based on the interrupt.

RESULT:

Thus the 8051 Assembly language program to turn on LED when external interrupt is low, turn on the LED and to perform toggle operation in P1.5 with the delay of 500 ms is written and executed successfully.

LABTASK-3:

AIM:

To write an 8051 assembly language program to display the message “21BEC1851”, on LCD display using DPTR.

PROCEDURE:

1. Create a Project (.uvproj)
2. Create and write an assembly program (.a51)
3. Build your project to check errors.
4. Select debug mode.
5. Run your code to verify the output.

PROGRAM:

```
ORG 0030H
MOV DPTR, #MYCOM
C1: CLR A
MOVC A, @A+DPTR
ACALL COMNWRT
ACALL DELAY
INC DPTR
JZ SEND_DAT
SJMP C1
SEND_DAT: MOV DPTR, #MYDATA
D1: CLR A
MOVC A,@A+DPTR
ACALL DATAWRT
ACALL DELAY
INC DPTR
JZ AGAIN
SJMP D1
AGAIN: SJMP AGAIN
COMNWRT: MOV P2, A
CLR P3.7
CLR P3.6
SETB P3.5
ACALL DELAY
```

```
CLR P3.5
SETB P3.5
ACALL DELAY
CLR P3.5
RET
DATAWRT: MOV P2, A
SETB P3.7
CLR P3.6
SETB P3.5
ACALL DELAY
CLR P3.5
RET
DELAY: MOV R3, #250
HERE2: MOV R4, #255
HERE: DJNZ R4, HERE
DJNZ R3, HERE2
RET
ORG 0300H
MYCOM: DB 38, 0EH, 01, 06, 84H, 0
MYDATA: DB “21BEC1851”, 0
END
```

OUTPUT VERIFICATION:

Lab Task-3:

```
ORG 0000H
LJMP MAIN
ORG 0030H
MAIN: MOV DPTR, #MYCOM
C1: CLR A
MOVC A, @A+DPTR
ACALL COMNWRT
INC DPTR
JZ SEND_DAT
SJMP C1
SEND_DATA: MOV DPTR, #MYDATA
D1: CLR A
MOVC A, @A+DPTR
ACALL DATAWRT
ACALL DELAY
INC DPTR
JZ AGAIN
SJMP D1
AGAIN: SJMP AGAIN
COMNWRT: MOV P2, A
CLR P3.7
CLR P3.6
SETB P3.5
ACALL DELAY
CLR P3.5
RET
DATAWRT: MOV P2, A
SETB P3.7
CLR P3.6
SETB P3.5
ACALL DELAY
CLR P3.5
DELAY: MOV R3, #250
```

```
HERE2 : MOV R4, # 255
HERE : DJNZ R4, HERE
       DJNZ R3, HERE2
       RET

ORG 0300 H
MYCOM : DB 38H, 0EH, 01, 06, 84H, 0
MY DATA : DB '21BEC1851', 0
END
```

INFERENCE:

Thus, the register number, “21BEC1851” is printed one by one in LCD display.

RESULT:

Thus, an 8051 assembly language program to display the message “21BEC1851”, on LCD display using DPTR is written and executed successfully.