Experiment No: 1

16 BIT ARITHMETIC OPERATIONS

Aim: To write an ALP to 8086 to perform 16-bit arithmetic operations in various Addressing modes.

Tools: PC installed with MASM/TASM

Program:

```
ASSUME CS: CODE, DS: DATA
DATA SEGMENT
OPR1 DW   5678H
OPR2 DW   1234H
SUM DW   ?
DIFF DW ?
PRODUCT1 DW  ?
PRODUCT2  DW  ?
QUOTIENT    DW  ?
REMAINDER DW ?
DATA  ENDS
CODE SEGMENT
START:        MOV   AX, DATA
              MOV   DS, AX
              MOV   AX, 5678H             ;Immediate addressing mode
              ADD   AX, 1234H             ;Direct addressing mode
              MOV   SUM, AX
              MOV   AX, OPR1
              MOV   BX, OPR2
              SUB   AX, BX
              MOV   DIFF, AX
              MOV   AX, OPR1
```

```
        MUL   BX                     ; Register addresiing mode
        MOV   PRODUCT1, AX
        MOV   PRODUCT2, DX
        MOV   DX, 0000H
        MOV   AX, OPR1
        MOV   BX, OPR2
        DIV   BX
        MOV   QUOTIENT, AX
        MOV   REMAINDER, DX
        INT   03H
    CODE ENDS
    END START
```

Result:

Experiment no: 2

SORTING AN ARRAY FOR 8086

ASCENDING ORDER

Aim: Write and execute an ALP to 8086 processor to sort the given 16-bit numbers in ascending and order.

Tools:

PC installed with masm 6.11

Program:

```
ASSUME CS:CODE,DS:DATA
DATA SEGMENT
LIST DW 0125H, 0144H, 3001H, 0003H, 0002H
COUNT EQU 05H
DATA ENDS
CODE SEGMENT
START:      MOV     AX,DATA
            MOV     DS,AX
            MOV     DX,COUNT-1
BACK:       MOV     CX,DX
            MOV     SI,OFFSET LIST
AGAIN:      MOV     AX,[SI]
            CMP     AX,[SI+2]
            JC      GO
            XCHG   AX,[SI+2]
            XCHG   AX,[SI]
GO:         INC     SI
            INC     SI
            LOOP    AGAIN
            DEC     DX
            JNZ     BACK
            INT     03H
CODE ENDS
```

END START

END

RESULT:

Experiment no: 3

SORTING AN ARRAY FOR 8086

DESCENDING ORDER

Aim: Write and execute an ALP to 8086 processor to sort the given 16-bit numbers in and descending order.

Tools:

PC installed with masm 6.11

Program:

```
ASSUME CS:CODE, DS:DATA
DATA SEGMENT
LIST DW 0125H, 0144H, 3001H, 0003H, 0002H
COUNT EQU 05H
DATA ENDS
CODE SEGMENT
START:      MOV    AX, DATA
            MOV    DS, AX
            MOV    DX, COUNT-1
BACK:       MOV    CX, DX
            MOV    SI, OFFSET LIST
AGAIN:      MOV    AX, [SI]
            CMP    AX, [SI+2]
            JNC    GO
            XCHG  AX, [SI+2]
            XCHG  AX, [SI]
GO:         INC    SI
            INC    SI
            LOOP   AGAIN
            DEC    DX
            JNZ    BACK
            INT    03H
```

```
CODE ENDS
END START
END
```

RESULT:

Experiment no: 4

SRTING MANIPULATIONS

MOVE BLOCK

Aim: To write an assembly language program to move the block of data from a source BLOCK to the specified destination BLOCK.

Tools: PC installed with MASM 6.11

Program:

```
.MODEL SMALL
.STACK 45H
ASSUME CS: CODE, DS: DATA
DATA SEGMENT
STRING  DB 04H, 0F9H, 0BCH, 98H, 40H
COUNT EQU 05H
DATA ENDS
EXTRA SEGMENT
ORG 0010H
STRING1 DB 5 DUP(?)
EXTRA ENDS
CODE SEGMENT
START:      MOV   AX, DATA
            MOV   DS, AX
            MOV   AX, DATA
            MOV   ES, AX
            MOV   SI, OFFSET STRING
            MOV   DI, OFFSET STRING1
            MOV   CL, COUNT
            CLD
            REP   MOVSB
            INT   03H
```

```
CODE ENDS
END START
END
```

Result:

Experiment no: 5

SRTING MANIPULATIONS

REVERSE STRING

Aim: To write an assembly language program to 8086 to reverse the given string.

Tools: PC installed with MASM 6.11

Program:

```
.MODEL SMALL
.STACK 48H
ASSUME CS: CODE,DS:DATA
DATA SEGMENT
STR DB 01H,02H,03H,04H
COUNT EQU 02H
DATA ENDS
CODE SEGMENT
START:      MOV   AX, DATA
            MOV   DS, AX
            MOV   CL, COUNT
            MOV   SI, OFFSET STR
            MOV   DI, 0003H
BACK:       MOV   AL, [SI]
            XCHG [DI], AL
            MOV   [SI], AL
            INC   SI
            DEC    DI
            DEC   CL
            JNZ   BACK
```

```
                    INT     03H
CODE ENDS
END START
END
```

Result:

Introduction to KEIL

STEPS TO EXECUTE 8051 PROGRAMS:

1.Click on Keil Micro Vision

2.click on **new micro vision project**,

3.Create a **new folder** and then save the project.

4.It displays all the manufacturers of Microcontroller. Select **ATMEL**, in that again select **AT89C51**.then click on OK, then a pop up window appears, select on **NO.**

5.A new target is created.

6.click on **New** and write the program

7.Save the program with **filename.a51**.

8.Right click on **source target** on the left side,select open to "**add file to source group target1**",select the respective file with **.a51** extension.

9.click on **target options** and select **target**,then change the clock frequency to **11.0598MHZ** and click **OK**.

10.Again open the **target options** and click on **OUTPUT**,click on the check box to create a **hex file.**

11.Now compile and build the hex file by clicking on **translate,build and rebuild** all options.

12.If errors exists,check with the line numbers,edit ,save and compile once again.

13.**Debug** the program.For step by step execution,click on **step in** option.For executing whole program at once,click on **run** button.

14.The registers value will be updated after the execution of the program.

15.To make note of hexcode of the program,go to the folder that was created at the starting ,and then select the **filename.lst** file.

16.Make note of the register values,code table and data table(input and ouput values with memory locations).

Experiment No: 6

Programming using arithmetic instructions of 8051

AIM: To perform Arithmetic operations on two 8-bit numbers using 8051 instructions

Tools required: Keil microvision software

Source Code:

```
ORG 00H
MOV A,#96H
MOV B,#69H
ADD A,B
MOV R0,A
MOV A,#96H
SUBB A,B
MOV R1,A
MOV A,#96H
MUL AB
MOV R2,A
MOV R3,B
MOV A,#96H
MOV B,#69H
DIV AB
MOV R4,A
MOV R5,B
END
```

Theoritical calculations:

Registers:

Code table: *****should be written in prescribed format along with comments

Result:

Thus performed arithmetic operations on two 8-bit numbers using 8051.

Experiment No: 7

Programming using logical and bit manipulation instructions of 8051

<u>AIM</u>: To perform logical operations on two 8-bit numbers using 8051 instructions

<u>Tools required</u>: Keil microvision software

<u>Source Code:</u>

```
ORG 00H
MOV A, #96H
MOV B, #69H
ANL A, B
MOV R0, A
MOV A, #96H
ORL A, B
MOV R1, A
MOV A, #96H
CPL A
MOV R2, A
END
```

<u>Theoritical calculations</u>:

<u>Registers:</u>

**<u>Code table</u>: ***should be written in prescribed format along with comments

<u>Result:</u>

Thus performed logical operations on two 8-bit numbers using 8051.

Experiment No: 8

TIMER/COUNTERS

Aim: To write an ALP in 8051 to verify timer/counter operation

Tools: **i)uxasm**
ii)tkup
iii)tkup86 kit
iv)frc cable

Program:

```
#INCLUDE "TKUP52.DEF"
ORG 0000H
START:      LJMP        MAIN
            ORG         0150H
MAIN:       MOV         SP, #060H
            MOV         TMOD, #01H
BACK:       MOV         TL0, #075H
            MOV         TH0, #0B8H
            MOV         P1, #0AAH
            LCALL       SFTDL
            ACALL       DELAY
            MOV         TL0, #00H
            MOV         TH0, #00H
            MOV         P1, #055H
            ACALL       DELAY
            LCALL       SFTDL
            SJMP        BACK
ORG 300H
DELAY:      SETB        TCON4
AGAIN:      JNB         TCON5, AGAIN
            CLR         TCON4
            CLR         TCON5
            RET

SFTDL       MOV         R4, #10H
DL3         MOV         R5, #0FFH
DL2         MOV         R6, #0FFH
DL1         DJNZ        R6, DL1
            DJNZ        R5, DL2
            DJNZ        R4, DL3
            RET
```

Result:

Experiment No: 9

INTERRUPT HANDLING

Aim: To write an ALP to verify the interrupt handling in 8051

Tools:

i) uxasm

ii) tkup

iii) tkup86 kit

iv) frc cable

Program:

```
#INCLUDE "TKUP52.DEF"
ORG 0000H
START:      LJMP  MAIN
            ORG   0150H
MAIN        MOV   SP, #50H
            MOV   IE, #85H
HERE        MOV   P1, #7EH
            SJMP  HERE
ORG 0003H                                  ;INT0 ISR
            MOV  P1, #0AAH
            LCALL DELAY
            LCALL DELAY
            LCALL DELAY
            RETI
ORG 0013H                                  ;INT1 ISR
            MOV  P1, #0A5H
            LCALL DELAY
            LCALL DELAY
            RETI
DELAY       NOP
```

```
        MOV   R4, #020H
DLY3    MOV   R3, #0FFH
DLY2    MOV   R2, #0FFH
DLY1    NOP
        NOP
        DJNZ  R2, DLY1
        DJNZ  R3, DLY2
        DJNZ  R4, DLY3
        RET
```

Result:

Experiment no: 10

UART OPERATION

Aim: To write an ALP to 8051uC for UART operation.

Tools: **i) uxasm**
ii) tkup
iii) tkup86 kit
iv)frc cable

Program:

```
;connect the rs232 from pc to tkup51 kit
;connect the tx pin of 8051 to rx of max232 and vice versa
;connect port1 to cnled

#INCLUDE "TKUP52.DEF"
ORG 0000H
START:      LJMP  MAIN
            ORG   0150H
MAIN:       MOV   SP, #060H
            MOV   IE, #85H
            MOV   TMOD, #20H
            MOV   TH1, #0FAH
            MOV   SCON, #50H
            SETB  TCON6
RPT:        MOV   SBUF, #'Y'
HERE:       JNB   SCON1, HERE
            CLR   SCON1
            MOV   A, #'A'
            MOV   P1, A
            SJMP  RPT
```

Result:

Experiment No: 11

SERIAL COMMUNICATION

AIM: To transmit data serially to the pc at baud rate 9600
REGISTERS USED: General purpose registers: AL, CX
PORTS USED: PORT A
ALGORITHM:
Step 1: Start
Step 2: Load R0 with 06H.
Step 3: Load DTPR with 4150H.
Step 4: Load SCON with 50H.
Step 5: Load TMOD with 20H.
Step 6: Load TH1 with FDH.
Step 7: Set bit TR1.
Step 8: Load A with content of DTPR.
Step 9: Load SBUF with contents of A.
Step 10: Repeat step 10 until T1=1, if check whether if T1=1, go to step8.
Step 11: Clear T1.
Step 12: Increment DTPR.
Step 13: Decrement R0, if R0≠0 go to step 8.
Step 14: Jump to step 2.
Step 15: Stop

PROGRAM:

```
AGAIN:   MOV    R0, #06H
         MOV    DPTR, #4150H
         MOV    SCON, #50H
         MOV    TMOD, #20H
         MOV    TH1, #0FDH
         SETB   8EH
NEXT:    MOVX   A,@DPTR
         MOV    SBUF, A
HERE:    JNB    99H, HERE
         CLR    T1
         INC    DPTR
         DJNZ   R0, NEXT
         SJMP   AGAIN
```