CERTIFICATE

This is to Certify that **Mr. Parmar Parth**. from R.K. University Rajkot having Enrollment Number **20SOECE13022** has completed term work of Sem-5 for subject Microprocessor and Microcontroller (EC509) During the Academic year 2021-22.
Date : 10/10/2021

Signature of Teacher..... HOD……..

EXPERIMENT-1

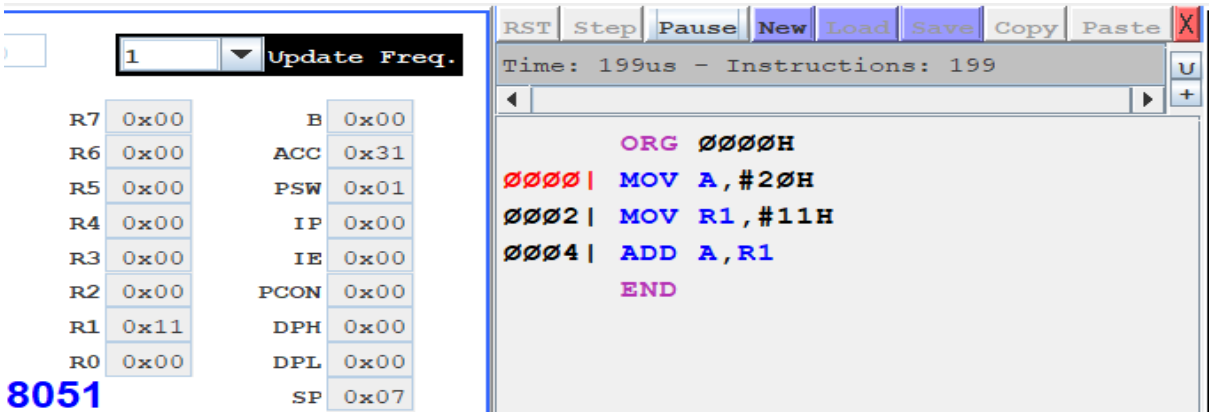
Aim: Write an Assembly Language program to

(A) Add/subtract two 8-bit numbers.

- **Code**:

```
ORG 0000H
MOV A,#20H
MOV R1,#11H
ADD A,R1
END
```

- **Output**:

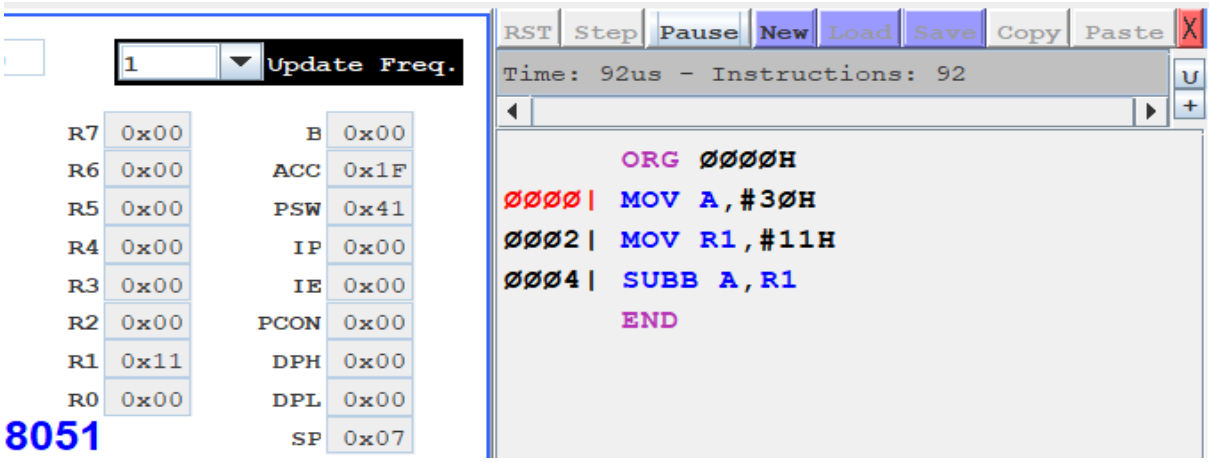


(B) Add/subtract two 16-bit numbers

- **Code**:

```
ORG 0000H
MOV A,#30H
MOV R1,#11H
SUBB A,R1
END
```

➢ **Output**:



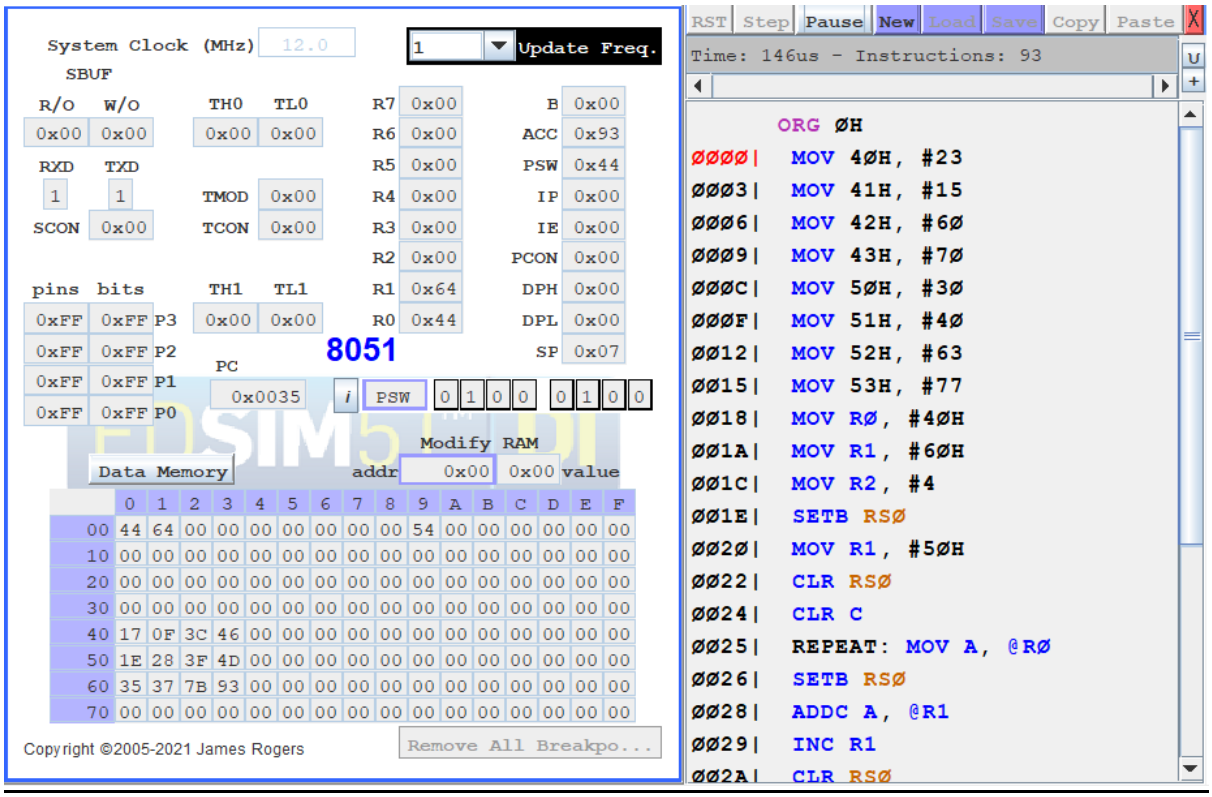
(C) Add/subtract two 32-bit numbers

➢ **Code**:

```
ORG 0H
MOV 40H, #23
MOV 41H, #15
MOV 42H, #60
MOV 43H, #70
MOV 50H, #30
MOV 51H, #40
MOV 52H, #63
MOV 53H, #77
MOV R0, #40H
MOV R1, #60H
MOV R2, #4
SETB RS0
MOV R1, #50H
CLR RS0
CLR C
REPEAT: MOV A, @R0
SETB RS0
ADDC A, @R1
INC R1
CLR RS0
MOV @R1, A
```

```
INC R0
INC R1
DJNZ R2, REPEAT
JNC EXIT
INC @R1
EXIT: NOP
SJMP $
END
```

- **Output**:



EXPERIMENT-2

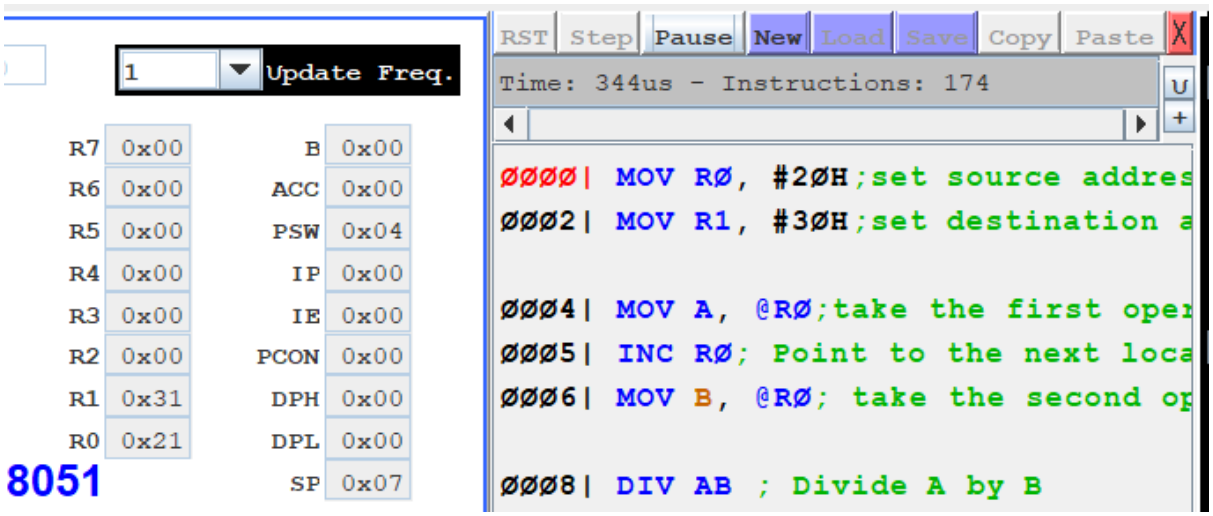
Aim: Write an Assembly Language program to

(A) Divide an 8-bit number by 8-bit number.

- **Code**:

```
MOV R0, #20H;set source address 20H to R0
MOV R1, #30H;set destination address 30H to R1

MOV A, @R0;take the first operand from source to register A
INC R0; Point to the next location
MOV B, @R0; take the second operand from source to register B

DIV AB ; Divide A by B

MOV @R1, A; Store Quotient to 30H
INC R1; Increase R1 to point to the next location
MOV @R1, B; Store Remainder to 31H
HALT:   SJMP HALT ;Stop the program
```

- **Output**:

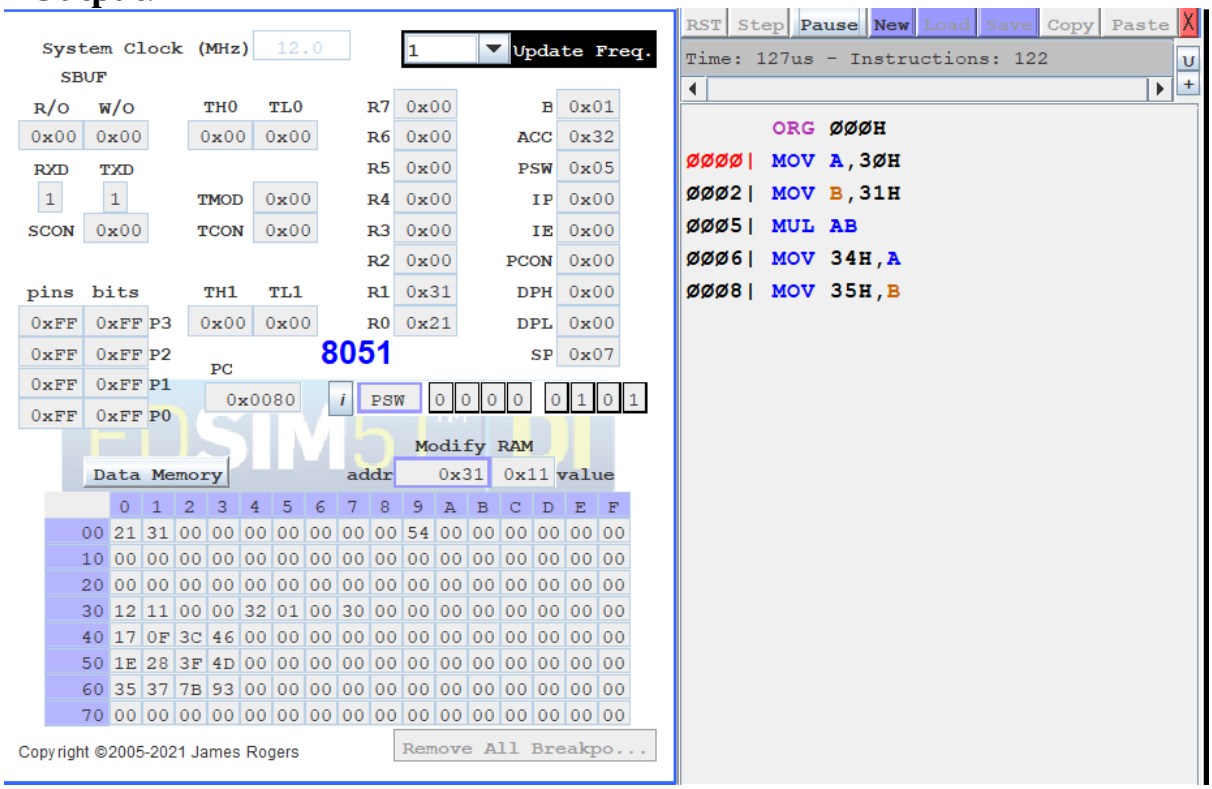


(B) Multiply two 8-bit numbers.

- **Code**:

```
ORG 000H
MOV A,30H
MOV B,31H
MUL AB
MOV 34H,A
MOV 35H,B
```

- **Output**:



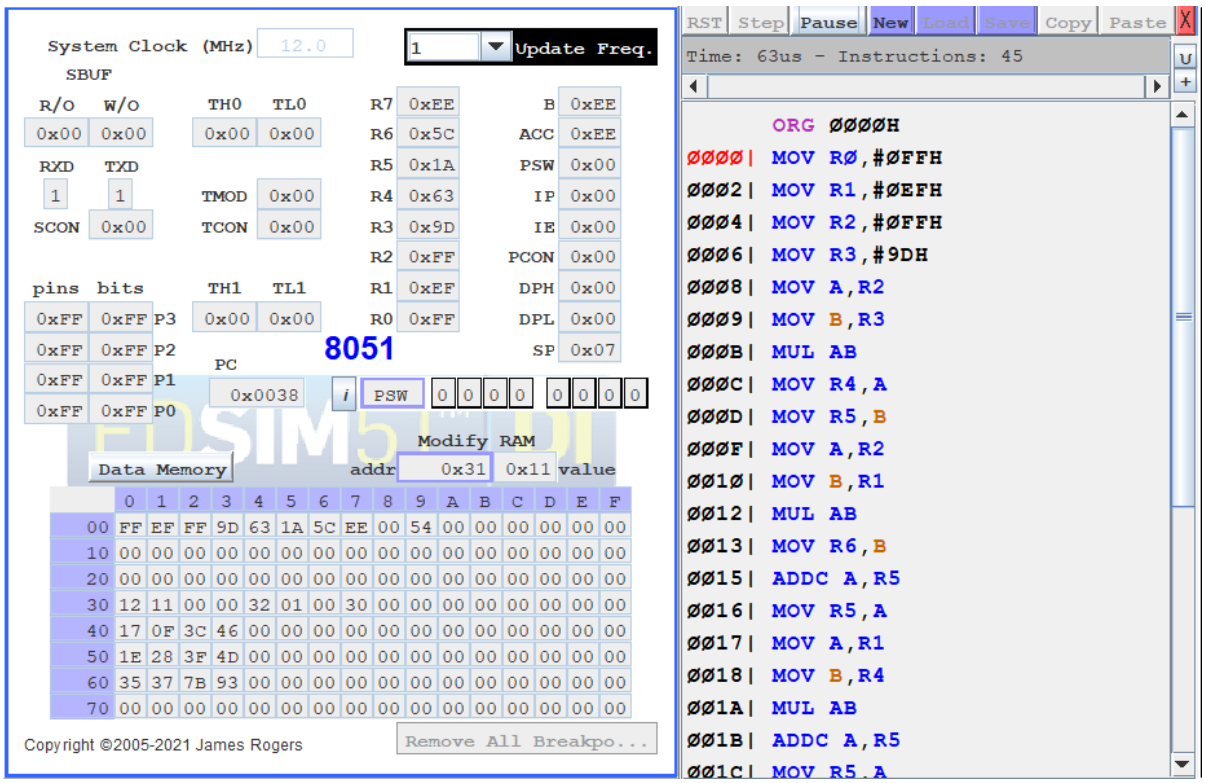
(C) Multiply two 16-bit numbers.

- **Code**:

```
ORG 0000H
MOV R0,#0FFH
MOV R1,#0EFH
MOV R2,#0FFH
MOV R3,#9DH
MOV A,R2
MOV B,R3
MUL AB
MOV R4,A
MOV R5,B
MOV A,R2
MOV B,R1
```

```
MUL AB
MOV R6,B
ADDC A,R5
MOV R5,A
MOV A,R1
MOV B,R4
MUL AB
ADDC A,R5
MOV R5,A
MOV A,B
ADDC A,R6
MOV R6,A
MOV A,R1
MOV B,R2
MUL AB
ADDC A,R6
MOV R6,A
MOV A,B
ADDC A,#00H
MOV R7,A
END
```

- **Output**:



EXPERIMENT-3

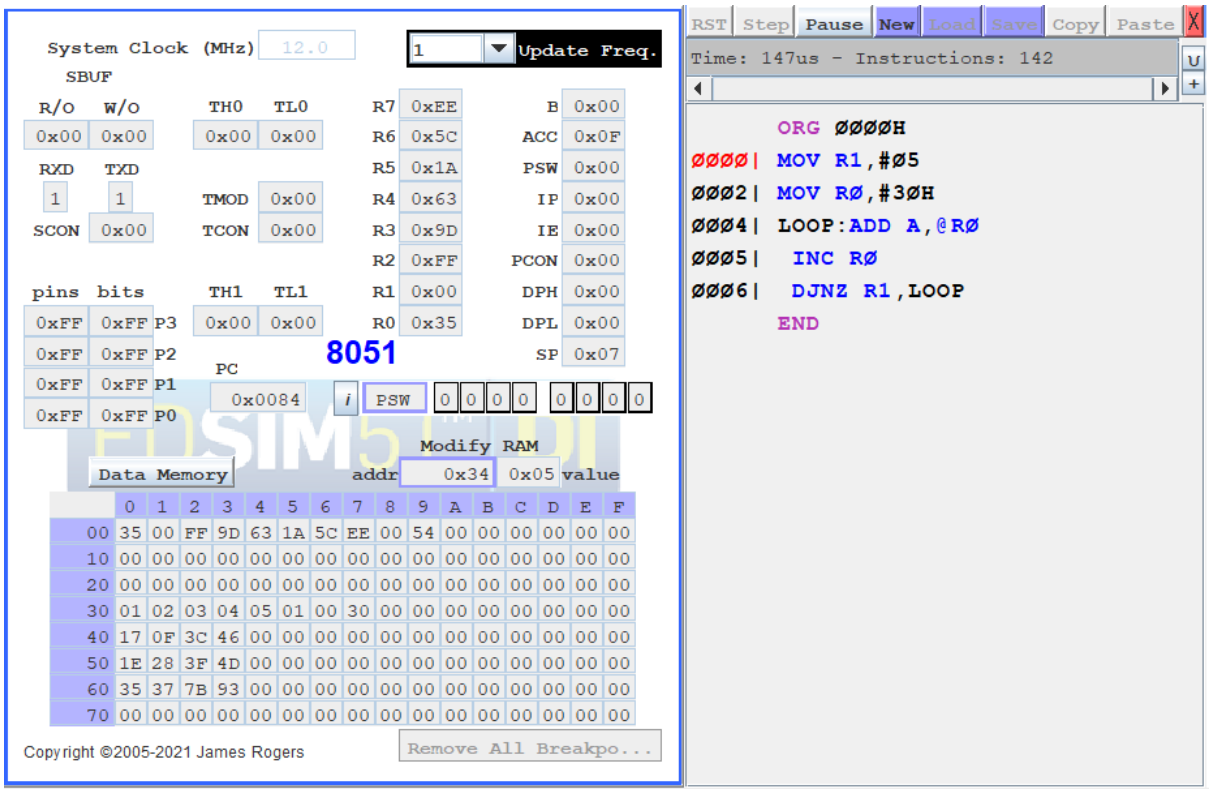
Aim: Write an Assembly Language program to

(A) Add block of data stored in internal/external memory locations.

- **Code**:

```
ORG 0000H
MOV R1,#05
MOV R0,#30H
LOOP:ADD A,@R0
 INC R0
 DJNZ R1,LOOP
END
```

- **Output**:

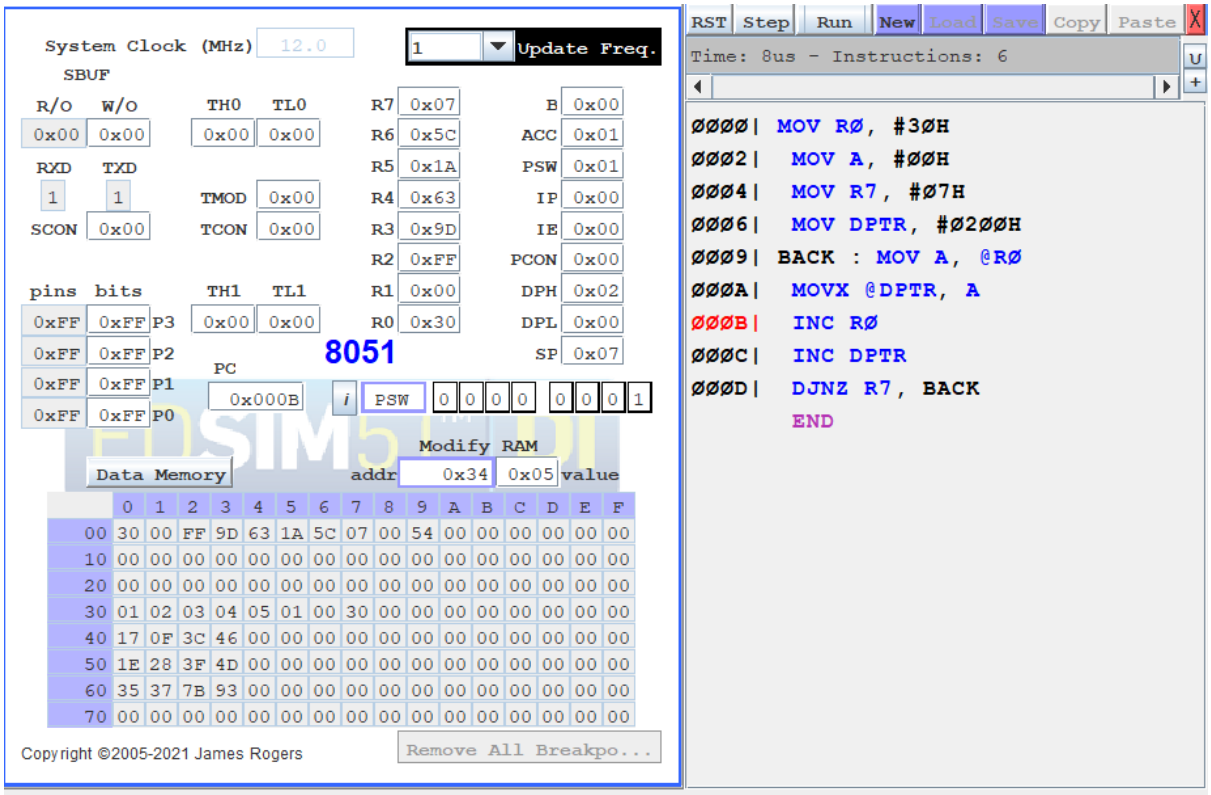


(B) Transfer block of data from internal memory locations to external memory locations.

- **Code**:

```
MOV R0, #30H
 MOV A, #00H
 MOV R7, #07H
 MOV DPTR, #0200H
BACK : MOV A, @R0
 MOVX @DPTR, A
 INC R0
 INC DPTR
 DJNZ R7, BACK
 END
```

- **Output**:



EXPERIMENT-4

Aim: Write an Assembly Language program to

(a) Find the maximum/minimum number from block of 5 data bytes stored from memory location 20h onwards.

- **Code**:

```
MOV R0, #05H
MOV R1, #20H
MOV B, @R1
LOOP:
        INC R1
        MOV A, @R1
        CMP B
        JNC SKIP
        MOV B, A
SKIP: DJNZ R0, LOOP
```

(b) Sort block of data in ascending or descending order stored from memory location 20h onwards.

- **Code**:

```
    LXI H,20H     ;Set pointer for array
    MOV C,M       ;Load the Count
    DCR C         ;Decrement Count
REPEAT: MOV D,C
    LXI H,20H     ;Load starting address of data array
LOOP:  MOV A,M    ;copy content of memory location to Accumulator
    INX H
    CMP M
    JNC SKIP      ;Jump to skip if carry not generated
    MOV B,M       ;copy content of memory location to B - Register
    MOV M,A       ;copy content of A - Register to memory location
    DCX H         ;Decrement content of HL pair of registers
    MOV M,B
    INX H         ;Increment content of HL pair of registers
SKIP:  DCR D
    JNZ LOOP      ;Jump to LOOP if not Zero
    DCR C         ;Decrement Count
    JNZ REPEAT    ;Jump to REPEAT if not Zero
    HLT           ;Terminate Program
```

EXPERIMENT-5

Aim: Write an Assembly Language program to

(A) Blinking LED at One second.

- **Code**:

```
START: CPL P1.0
 ACALL WAIT
 SJMP START
WAIT: MOV R4,#05H
WAIT1: MOV R3,#00H
WAIT2: MOV R2,#00H
WAIT3: DJNZ R2,WAIT3
 DJNZ R3,WAIT2
 DJNZ R4,WAIT1
 RET
```

- **Output**:

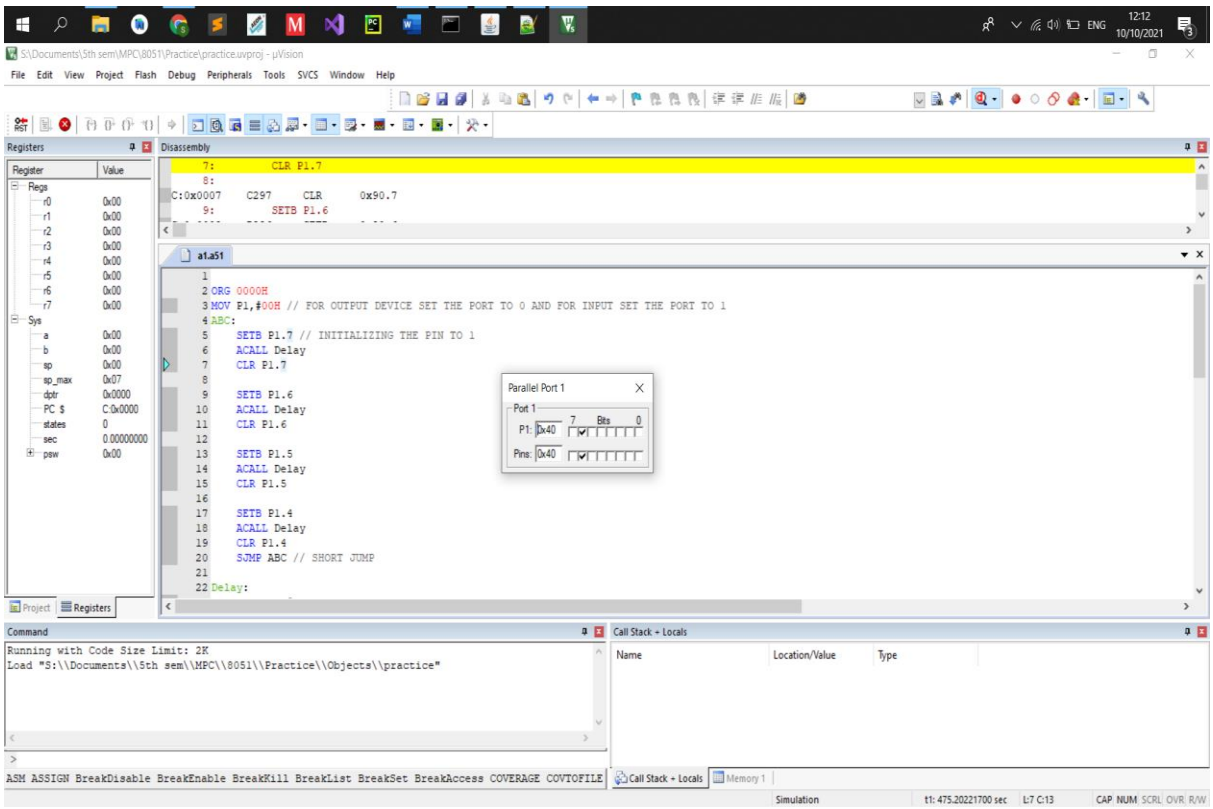
(B) Ring counter (Shifting) pattern at One second.

- **Code**:

```
ORG 0000H
MOV P1,#00H // FOR OUTPUT DEVICE SET THE PORT TO 0
AND FOR INPUT SET THE PORT TO 1
ABC:
        SETB P1.7 // INITIALIZING THE PIN TO 1
ACALL Delay
CLR P1.7

SETB P1.6
ACALL Delay
CLR P1.6

SETB P1.5
ACALL Delay
CLR P1.5

SETB P1.4
ACALL Delay
CLR P1.4
```

```
    SJMP ABC // SHORT JUMP

Delay:
    MOV R0, #255H
    Z:
        MOV R1, #255
    Y:
        MOV R2, #255
    X:
        DJNZ R2, X
    DJNZ R1, Y
    DJNZ R0, Z
    RET
    END
```

- **Output**:



EXPERIMENT-6

Aim: Write an Assembly Language program to

(A) Monitoring P1.2 until it becomes high; when P1.2 becomes high write value 45H on P0, Sent a high to low pulse to P2.3.

- **Code**:

```
CLR P1.2
MOV A,#45H
AGAIN:JNB P1.2,AGAIN
MOV P0,A
SETB P2.3
CLR P2.3
END
```

(B) Check a status of P1.7 Switch and perform the following

1. If switch = 0, send letter "N" to P2.
2. If switch = 1, send letter "Y" to P2.

- **Code**:

```
SETB P1.7
AGAIN:
 JB P1.2,OVER
MOV P2,#'N'
SJMP AGAIN
OVER:
MOV P2,#'Y'
SJMP AGAIN
```

EXPERIMENT-7

Aim: Write an Assembly Language program to.

(A) Write a program to generate 5 KHz pulse waveform of 50% duty cycle on pin 1.0 using timer 1 in mode 2.

- **Code**:

```
MOV TMOD,#01
HERE: MOV TL0,#0F2H
MOV TH0,#0FFH
CPL P1.5
ACALL DELAY
SJMP HERE
DELAY:
SETB TR0
AGAIN: JNB TF0,AGAIN
CLR TR0
CLR TF0
RET
```

(B) Write a program to generate 1 KHz pulse wave form of 70% duty cycle on pin 1.0 using timer.

- **Code**:

```
MOV TMOD, #01H
HERE: MOV TL0, #44H
MOV TH0, #0FDH
CPL P1.5
ACALL DELAY
SJMP HERE

DELAY:
SETB TR0
AGAIN: JNB TF0, AGAIN
CLR TR0
CLR TF0
RET
```

EXPERIMENT-8

Aim: Write an Assembly Language program to.

(A) Transfer letter "A" serially, continuously.

- **Code**:

```
MOV TMOD,#20H
MOV TH1,#-6
MOV SCON,#50H
SETB TR1
AGAIN:
MOV SBUF,#"A"
HERE:
JNB TI,HERE
CLR TI
SJMP AGAIN
```

(B) The message "YES" serially. Do this continuously.

- **Code**:

```
MOV TMOD,#20H
MOV TH1,#-3
MOV SCON,#50H
SETB TR1
AGAIN:
MOV A,#"Y"
ACALL TRANS
MOV A,#"E"
ACALL TRANS
MOV A,#"S"
ACALL TRANS
SJMP AGAIN
TRANS:
MOV SBUF,A
HERE:
JNB TI,HERE
CLR TI
RET
```

(C) Receive bytes of data serially, and put them in P1.

- **Code**:

```
MOV TMOD,#20H
MOV TH1,#-6
MOV SCON,#50H
SETB TR1
HERE:
JNB RI,HERE
MOV A,SBUF
MOV P1,A
CLR RI
SJMP HERE
```

EXPERIMENT-9

Aim: Write an Assembly Language program to generate a square wave of 2 Khz and also receive a byte of data serially and send to P0 continuously using interrupt method.

- **Code:**

```
MOV TMOD,#01
AGAIN: MOV TL1,#1AH
MOV TH1,#0FFH
SETB TR1
BACK: JNB TF1,BACK
CLR TR1
CLR P0.5
CLR TF1
SJMP AGAIN
```

EXPERIMENT-10

Aim: Write an Assembly Language program to display string "RK University" on 16x2 LCD

- **Code**:

```
ORG 0000H
MOV P1, #00H

MOV A,#38H ; Use 2 lines and 5x7 matrix
ACALL CMND
MOV A,#0FH ; LCD ON, cursor ON, cursor blinking ON
ACALL CMND
MOV A,#01H ;Clear screen
ACALL CMND
MOV A,#06H ;Increment cursor
ACALL CMND
MOV A,#87H ;Cursor line one , position 2
ACALL CMND

MOV A,#'R'
ACALL DISP
MOV A,#'K'
```

```
ACALL DISP

MOV A,#0C3H ;Jump to second line, position 1
ACALL CMND

MOV A,#'U'
ACALL DISP
MOV A,#'N'
ACALL DISP
MOV A,#'I'
ACALL DISP
MOV A,#'V'
ACALL DISP
MOV A,#'E'
ACALL DISP
MOV A,#'R'
ACALL DISP
MOV A,#'S'
ACALL DISP
MOV A,#'I'
ACALL DISP
MOV A,#'T'
ACALL DISP
MOV A,#'Y'
ACALL DISP
HERE: SJMP HERE

CMND: MOV P1,A
CLR P2.0
SETB P2.1
ACALL DELY
CLR P2.1
RET

DISP:MOV P1,A
setb P2.0
SETB P2.1
ACALL DELY
CLR P2.1
RET

DELY:
  MOV R1,#255
```

```
DEF:MOV R2,#255
GHI: DJNZ R2,GHI
   DJNZ R1,DEF
RET

END
```

- **Output**: