**MICRONCONTROLLERS AND ITS APPLICATIONS LAB**

## **NAME: Pratyush Raj**

## **REG.N0: 20BEC1273**

## **DATE: 31-01-2022**

**Exp-4 Assembly Programming with Logical and Bit-Oriented Instruction of 8051**

**Lab Task 1**

**Aim:**

To write an assembly language program to perform logical operations - AND, OR, XOR, on 2 eight-bit numbers stored in internal RAM locations 21H, 22H & results stored in 30H, 31H and 32H respectively.

**Software Used:**

Keil uVision4

# Program:

MOV A,21H

ANL A,22H

MOV 30H, A

MOV A,21H

ORL A,22H

MOV 31H, A

MOV A,21H

XRL A,22H

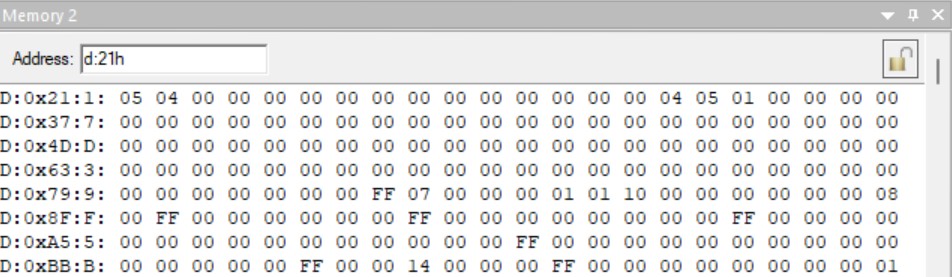
MOV 32H, A

END

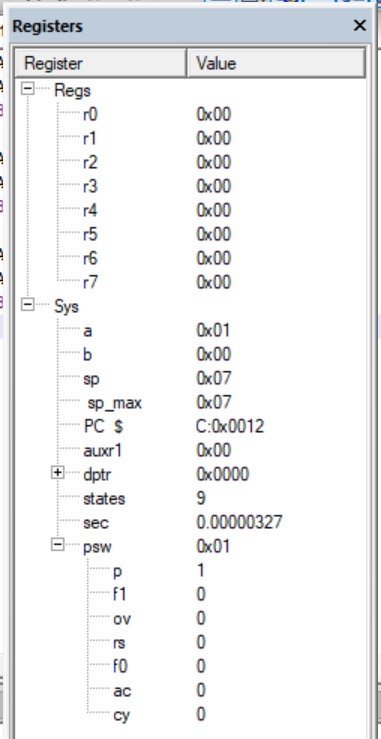
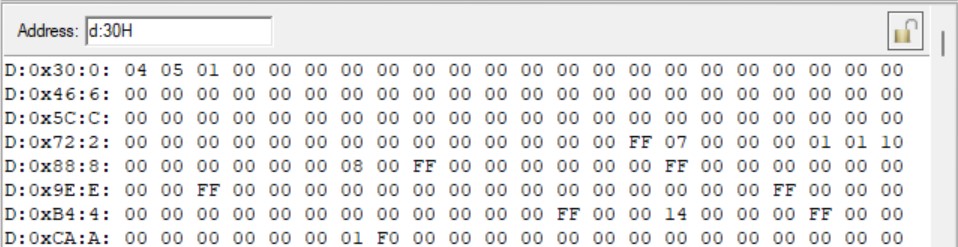
**Output:**

Input in 21H location:

Output at 30H:



**Result:**



We have successfully written a program that performs various logical operations (AND, OR, XOR) on 8-bit numbers.

------------------------------------------------------------------------------------------------

**Lab Task 2**

To write an assembly language program to compare two 8-bit number, stored in external memory locations 8000H and 8001H.

i. If NUM1<NUM2, set LSB at 2FH. ii. If NUM1>NUM2 set MSB at 2FH.

iii. If NUM1=NUM2 clear both MSB & LSB.

**Software Used:**

Keil uVision4

**Program:**

MOV DPTR,#8000H

MOVX A,@DPTR

MOV R0,A

INC DPTR

MOVX A,@DPTR

CLR C

SUBB A,R0

JZ EQUAL

JNC SMALL

## SETB 7FH

SJMP END1

SMALL: SETB 78H

SJMP END1

EQUAL: CLR 78H

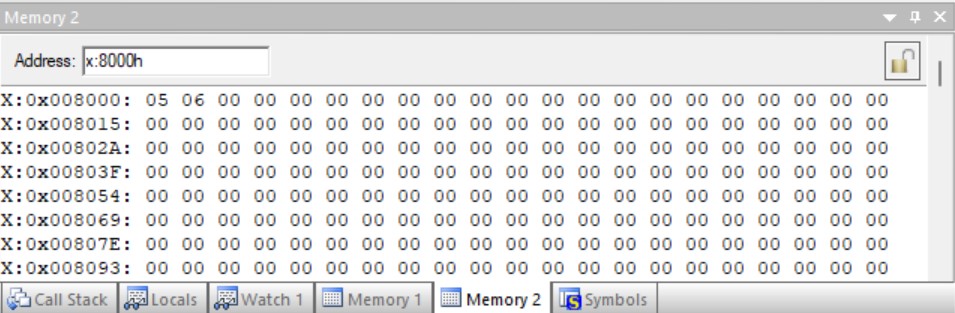
CLR 7FH

END1: NOP

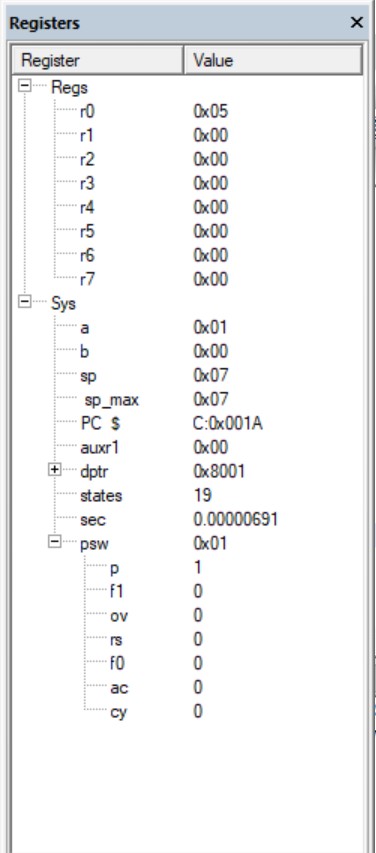
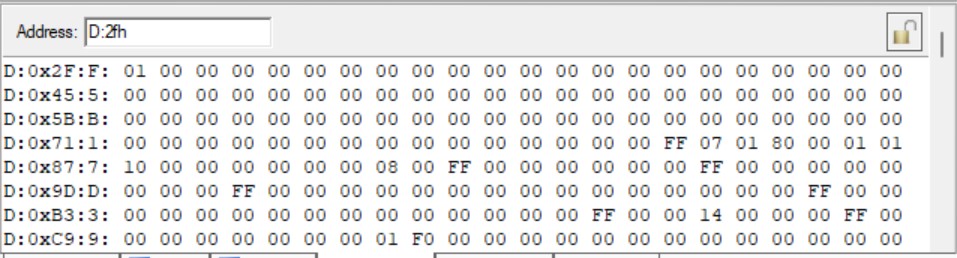
END

**Output:**

Input in 8000H location



Output at 2fh location



---------------------------------------------------------------------------------------------------------------------------

**Lab Task 3**

To write an assembly language program to implement a half-adder logic.

**Software Used:**

Keil uVision4

**Program:**

ORG 000H

CLR P2.0

CLR P2.1

## SETB P0.1

SETB P0.0

## MOV C,P0.0

ANL C,P0.1

MOV P2.0,C

MOV C,P0.0

JC INPUT1

MOV A,#00H

BACK1:MOV C,P0.1

JC INPUT2

MOV R0,#00H

BACK2:XRL A,R0

MOV C,ACC.0

MOV P2.1,C

SJMP END1

INPUT1:MOV A,#01H

SJMP BACK1

INPUT2:MOV R0,#01H

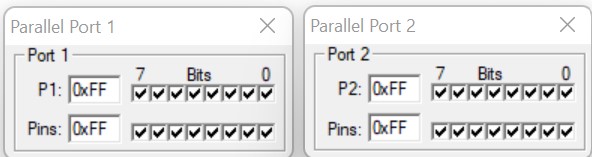
SJMP BACK2

END1:NOP

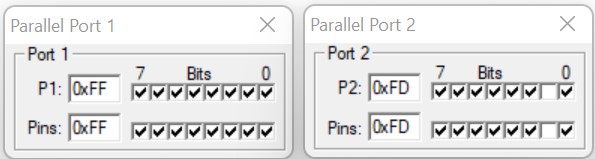
END

**Output:**

Before running the code:



After running the code:



**Result:**

We have successfully implemented a program to run a half-adder logic system.

------------------------------------------------------------------------------------------------

**Challenging Task 1**

To write an assembly language program to implement a full-adder logic.

**Software Used:**

Keil uVision4

**Program:**

ORG 0000H

CLR P1.0

CLR P1.1

SETB P0.2

SETB P0.1

SETB P0.0

MOV C,P0.2

JC INPUT1

MOV A,#00H

BACK1: MOV C,P0.1

JC INPUT2

MOV R0,#00H

BACK2:XRL A,R0

MOV C,ACC.0

ANL C,P0.0

MOV P1.0,C

MOV C, P0.2

ANL C,P0.1

ANL C,P1.0

MOV C,P0.0

JC INPUT3

MOV R0,#00H

BACK3:XRL A,R0

MOV C,ACC.0

MOV P1.1,C

SJMP END1

INPUT1: MOV A,#01H

SJMP BACK1

INPUT2: MOV R0,#01H

SJMP BACK2

INPUT3:MOV R0,#01H

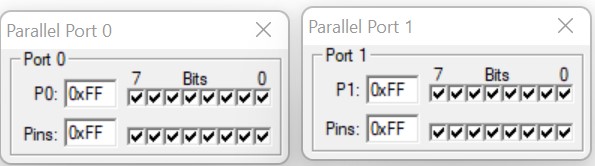
SJMP BACK3

END1: NOP

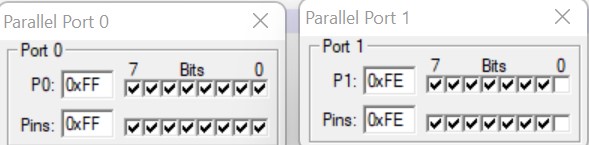
END

**Output:**

Before running the code:



After running the code:



**Result:**

We have successfully implemented a program to run a Full-Adder logic system.

------------------------------------------------------------------------------------------------

**Challenging Task 2**

To write an assembly language program to check whether a given 8-bit number is Odd or Even.

**Software Used:**

Keil uVision4

**Program:**

ORG 0000H

MOV A,55H

MOV C,ACC.0

JC ODD

MOV A,#0FFH

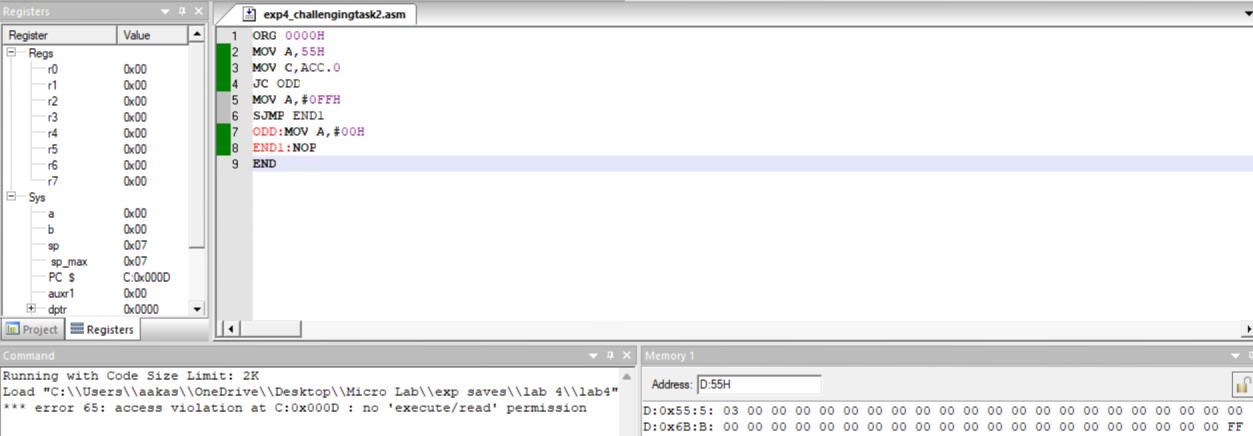
SJMP END1

ODD:MOV A,#00H

END1:NOP

END

**Output:**



**Result:**

We have successfully implemented a program to check whether a given 8-bit number is Odd or Even.

---------------------------------------------------------------------------------------------------------------------------