# **FR. Conceicao Rodrigues College of Engineering Department of Computer Engineering**

## **5.TO COUNT EVEN AND ODD NUMBERS FROM AN ARRAY OF 10 NUMBERS**.

1. **Course, Subject & Experiment Details**

| **Academic Year** | **2023-24** | **Estimated Time** | **Experiment No. 5– 02 Hours** |
| --- | --- | --- | --- |
| **Course &**  **Semester** | **S.E. (Comps)**  **– Sem. IV** | **Subject Name** | **Microprocessor** |
| **Chapter No.** | **2** | **Chapter Title** | **Instruction Set and Programming** |
| **Experiment Type** | **Software** | **Subject Code** | **CSC405** |

**Rubrics**

| **Timeline (2)** | **Practical Skill & Applied Knowledge**  **(2)** | **Output (3)** | **Postlab (3)** | **Total (10)** | **Sign** |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |

## **Aim & Objective of Experiment**

**Arrange the given numbers in Ascending/ Descending order.**

**Objective :** Program involves counting even and odd numbers from a given array. The objective of this program is to give an overview of the string instructions of 8086

## **Software Required**

TASM Assembler

# **4** . **Brief Theoretical Description**

**Pre-Requisites:** 1. Knowledge of TASM directories.

2. Knowledge of CMP and Jump Instructions of 8086.

Theory: A string is a series of bytes stored sequentially in the memory. string

Instruction operates on such ‘strings’. The SRC element is taken from the data segment using and SI register. The destination element is in extra segment pointed by DI register. These registers are incremented or decremented after each operation depending upon the direction flag in flag register.

Some of the instructions useful for program are,

1. CLC - the instruction clears the carry flag.
2. RCR destination, count- Right shifts the bits of destination. LSB is shifted into CF. CF goes to MSB. Bits Are shifted counts no of times.
3. JC: jump to specified location.
4. INC/DEC destination: add/subtract 1 from the specified destination.
5. JMP label: The control is shifted to an instruction to which label is attached.
6. JNZ label: The control is shifted to an instruction to which label is attached if ZF = 0

# **Algorithm:**

* 1. . Initialize the data segment.
  2. Initialize the array.
  3. Load the effective address of an array in any index register.
  4. Load total number of elements of the array in any register.
  5. Initialize any two registers as counter for even and odd numbers to zero.
  6. Load first element of an array in any general purpose register.
  7. Shift/rotate the contents of loaded register to right.
  8. If CF=1 increment counter for odd numbers otherwise increment counter of even numbers.
  9. Store the value of even and odd counter register to two memory locations.
  10. Stop.

# **Conclusion:**

.8086

.model small

.data

array db 2FH, 4AH, 6BH, 8CH, 5EH, 8DH, 1FH, 12H, 34H, 56H

oddno db 0

evenno db 0

.code

start:

MOV AX, @data

MOV DS, AX

LEA SI, array

MOV CX, 000AH

check:

MOV Al, [SI]

TEST Al, 1

JNZ odd

INC [evenno]

JMP next

odd:

INC [oddno]

next:

INC SI

LOOP check

MOV AH, 02H

MOV DL, [evenno]

ADD DL, 30H

INT 21H

MOV DL, '-'

INT 21H

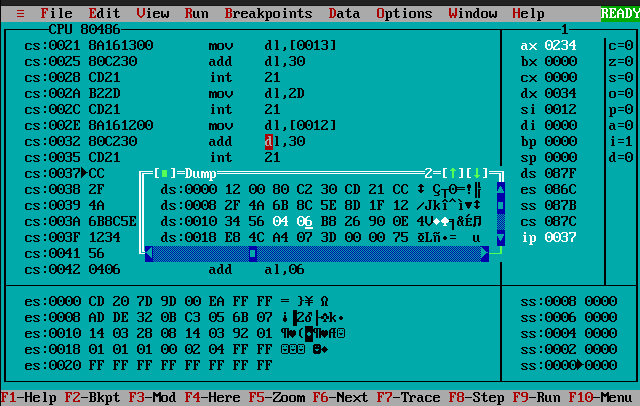
MOV DL, [oddno]

ADD DL, 30H

INT 21H

INT 03H

end start



1. **Postlab:**

# **Explain CMPSB/CMPSW , LODS/STOS instructions**

⇒

* CMPSB and CMPSW: These instructions compare two strings character by character (byte by byte) for equality.CMPSB compares one byte at a time while CMPSW compares two bytes at a time.
* LODS: This instruction loads data from memory into registers. It has three variations: LODSB, LODSW, and LODSD.
* STOS: This instruction stores data from registers into memory. It also has three variations: STOSB, STOSW, and STOSD.