Course Title: Microprocessors and Assembly Language Lab (CSE-4504)
Department of Computer Science and Engineering (CSE)
Islamic University of Technology (IUT), Gazipur

Lab # 05

*Understanding **Procedure** using Assembly Language Program.*

Objective:

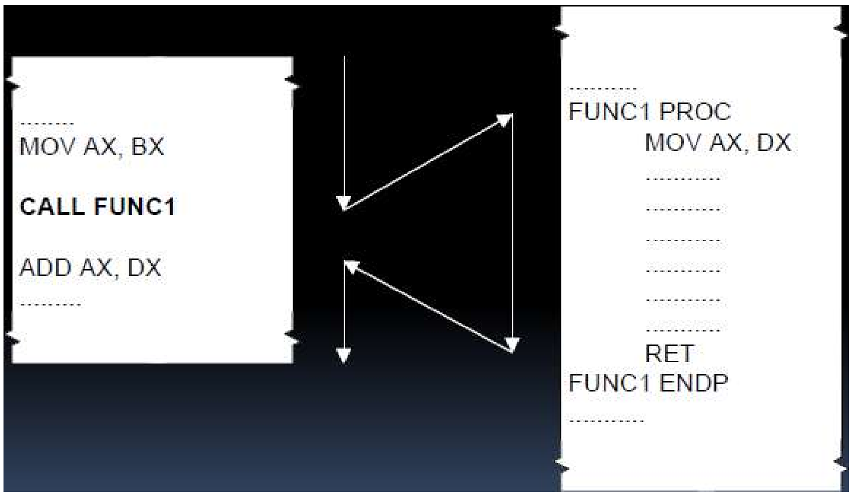
To understand 8086 instructions related to Procedure using Assembly Language Program.

Theory:

- **Procedures**
  With procedures we are able to write a separate piece of code, **call** it within our program, and return to the point that we left, having completed the code in the procedure. Procedures are also known as subroutines, functions or methods.

  Call and Return Instructions

  - We use the **CALL** instruction to transfer execution to the procedure
  - We use the **RET** instruction to return to where the procedure was called from



```
........
MOV AX, BX

CALL FUNC1

ADD AX, DX
.........
```

```
..........
FUNC1 PROC
        MOV AX, DX
        ...........
        ...........
        ...........
        ...........
        ...........
        ...........
        RET
FUNC1 ENDP
............
```

Execution of Call instruction results-
- IP is incremented to point to the next instruction and stored (on the stack)
- The address of the first instruction in the procedure is put into IP
- Execution is restarted in the procedure

Execution of Return instruction results-
- The old IP is restored (from the stack)
- Execution is restarted at the point where the procedure was called from

Assembly Language Program Example for Procedure:

```
ORG 0100H

.DATA

StrArray DB 'Hello World!!$'           ; define string to display

.CODE

MAIN PROC
        MOV AX, @DATA
        MOV DS,AX

        LEA DX, StrArray        ; set DX to point to 1st element of string array StrArray

        CALL USER               ; call procedure

        MOV AH, 4Ch
        MOV AL, 00h             ; a code after procedure call and return
        INT 21h                 ; exit to DOS
MAIN ENDP

USER PROC                       ; declare a procedure named USER
        MOV AH, 09h
        INT 21h
        RET                     ; return to MAIN procedure
USER ENDP                       ; end of procedure USER

END MAIN                        ; end of program
```

Tasks to do:

1. Write an Assembly Language code that takes any 5 of decimal digits (0 ~ 9) as input and derives the prime and non-prime digits using ***two different procedures*** named **Prime** and **Non-prime** and print those as an output.

   Sample Input / Output:

   ```
   Input: 24153

   Output:  Prime: 2 5 3
            Non-prime: 4 1
   ```