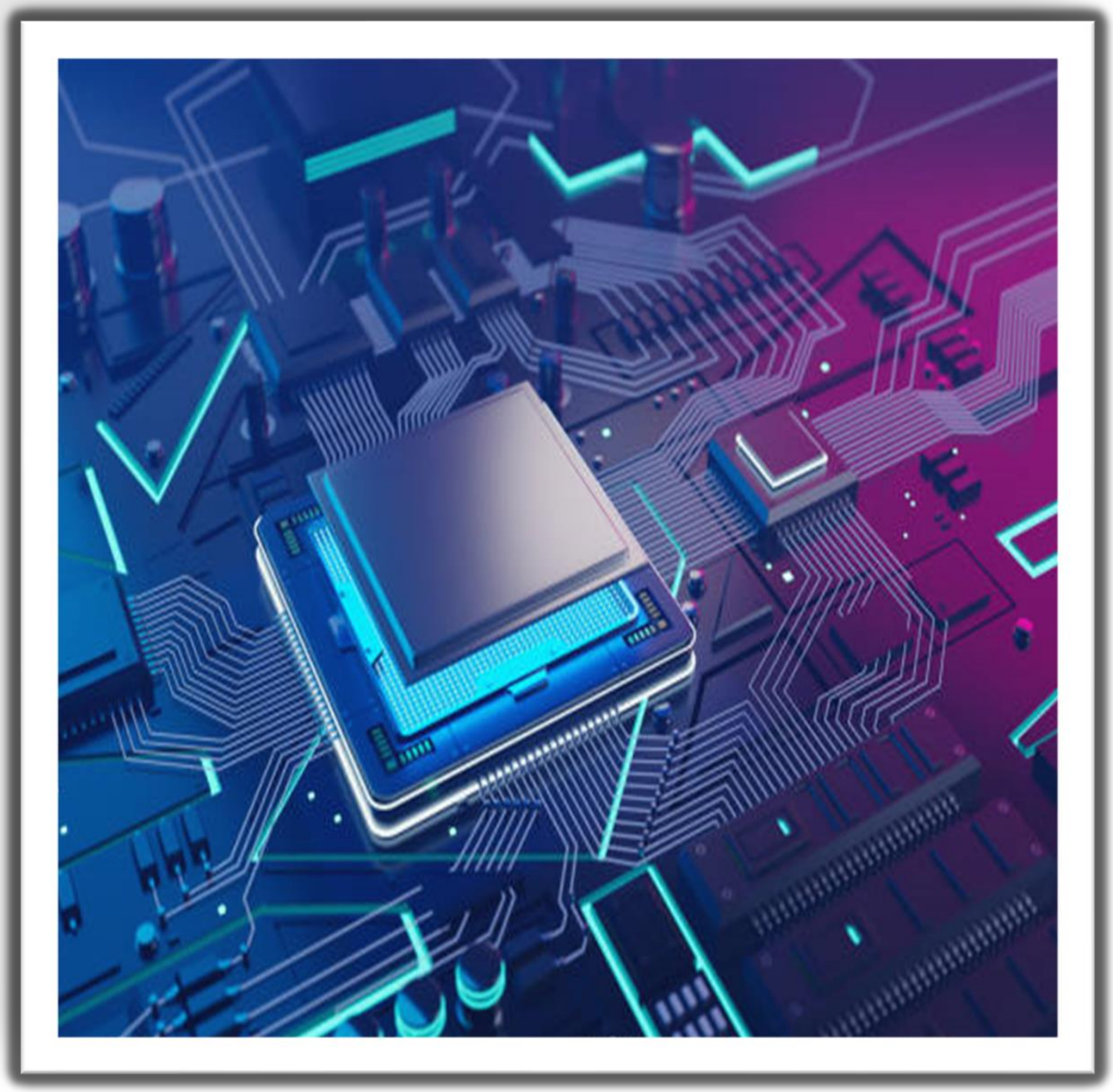

Microprocessor LAB Codes

ASIF ZAMAN RIZVE

asifzaman007@hotmail.com | Microprocessor | 2022

1. Two Single Digit Number Addition:

Solution:

First Add both numbers then use DAA instruction to find the sum if the addition result exceeds 9. after DAA the result will be stored in AL register.

```
.MODEL SMALL
.STACK 100
.DATA
.CODE
MAIN PROC
    MOV AH,01H
    INT 21H
    SUB AL,30H
    MOV BL,AL

    MOV AH,01H
    INT 21H
    SUB AL,30H

    ADD AL,BL
    DAA
    MOV CL,AL

    MOV AH,02H
    MOV DL,0AH
    INT 21H
    MOV AH,02H
    MOV DL,0DH
    INT 21H
    MOV AL,CL
    CMP AL,10H
    JE EQUAL
    JNG NOT_GREATER
```

```
GREATER:
    SUB AL,10H
    MOV BL,AL
    MOV AH,02H
    MOV DL,01H
    ADD DL,30H
    INT 21H
    MOV AH,02H
    MOV DL,BL
    ADD DL,30H
    INT 21H
    JMP EXIT
EQUAL:
    MOV AH,02H
    MOV DL,01H
    ADD DL,30H
    INT 21H
    MOV AH,02H
    MOV DL,00H
    ADD DL,30H
    INT 21H
    JMP EXIT

    NOT_GREATER:
    MOV AH,02H
    MOV DL,AL
    ADD DL,30H
    INT 21H

EXIT:
MAIN ENDP
END MAIN
```

2. Two Single Digit Number Subtraction:

Solution:

First Subtract two numbers. if the answer is negative the sign flag will be set. then sign jump will be done and 2’s complement will be taken by using NEG instruction. if the result of subtraction is positive then nothing will be happened.

```
.MODEL SMALL
.STACK 100
.DATA
.CODE
MAIN PROC

    MOV AH,01H
    INT 21H
    SUB AL,30H
    MOV BL,AL
    MOV CL,BL

    MOV AH,01H
    INT 21H
    SUB AL,30H

    SUB BL,AL
    JS NEGATIVE
    MOV DL,BL
    ADD DL,30H
    MOV BL,DL
    MOV DX,13;
    MOV AH,2
    INT 21H
    MOV DX,10;
    MOV AH,2
    INT 21H
    MOV DL,BL
    MOV AH,02H
    INT 21H
    JMP EXIT
```

```
    NEGATIVE:
    MOV DX,13;
    MOV AH,2
    INT 21H
    MOV DX,10;
    MOV AH,2
    INT 21H
    NEG BL
    MOV DL,2DH
    MOV AH,02H
    INT 21H
    MOV DL,BL
    ADD DL,30H
    MOV AH,02H
    INT 21H
    JMP EXIT

EXIT:
MAIN ENDP
END MAIN
```

3. Two Single Digit Number Division:
   Solution:

```
.MODEL SMALL
.STACK 100
.CODE
MAIN PROC
MOV AH,01H
INT 21H
SUB AL,30H
MOV CL,AL

MOV AH,02H
MOV DL,13
INT 21H
MOV AH,02H
MOV DL,10
INT 21H

MOV AH,01H
INT 21H
SUB AL,30H
MOV BL,AL
MOV AL,CL
MOV AH,00H
DIV BL;AL/BL Result:AL Remainder:AH
MOV CL,AL
MOV CH,AH

MOV AH,02H
MOV DL,13
INT 21H
MOV AH,02H
MOV DL,10
INT 21H

MOV AH,02H
MOV DL,CL
ADD DL,30H
INT 21H

MOV AH,02H
MOV DL,13
INT 21H
MOV AH,02H
MOV DL,10
INT 21H

MOV AH,02H
MOV DL,CH
ADD DL,30H
INT 21H

MAIN ENDP
END MAIN
```

4. Two Single Digit Number Multiplication:
   Solution:

```
.MODEL SMALL
.STACK 100
.DATA
.CODE
MAIN PROC
    MOV AH,01H
    INT 21H
    SUB AL,30H
    MOV CL,AL
    MOV AH,02H
    MOV DL,13
    INT 21H
    MOV AH,02H
    MOV DL,10
    INT 21H
    MOV AH,01H
    INT 21H
    SUB AL,30H
    MOV BL,AL
    MOV AL,CL
    MUL BL
    AAM
    MOV BL,AL
    MOV BH,AH
    MOV AH,02H
    MOV DL,0AH
    INT 21H
    MOV AH,02H
    MOV DL,0DH
    INT 21H
    MOV AH,02H
    ADD BH,30H
    MOV DL,BH
    INT 21H
    MOV AH,02H
    ADD BL,30H
    MOV DL,BL
    INT 21H
MAIN ENDP
END MAIN
```

5. Odd Even Check:
   Solution:

```
.MODEL SMALL
.STACK 100
.DATA
MSG1 DB "ENTER THE NUMBER:$"
MSG2 DB "THE NUMBER IS EVEN$"
MSG3 DB "THE NUMBER IS ODD$"
.CODE
MAIN PROC
    MOV AX,@DATA
    MOV DS,AX
    LEA DX,MSG1
    MOV AH,09H
    INT 21H
    MOV AH,01H
    INT 21H
    SUB AL,30H
    MOV BL,02H
    DIV BL
    CMP AH,00H
    JE EVEN
    ODD:
    MOV AH,02H
    MOV DL,0AH
    INT 21H
    MOV AH,02H
    MOV DL,0DH
    INT 21H
    MOV AX,@DATA
    MOV DS,AX
    LEA DX,MSG3
    MOV AH,09H
    INT 21H
    JMP EXIT
    EVEN:
    MOV AH,02H
    MOV DL,0AH
    INT 21H
    MOV AH,02H
    MOV DL,0DH
    INT 21H
    MOV AX,@DATA
    MOV DS,AX
    LEA DX,MSG2
    MOV AH,09H
    INT 21H
    JMP EXIT
EXIT:
MAIN ENDP
END MAIN
```

6. Largest Number among The Numbers:
   Solution:

```
.MODEL SMALL
.STACK 100
.DATA
LIST DB 25H,12H,35H,10H,11H,50H,48H,60H
.CODE
MAIN PROC
    MOV AX,@DATA
    MOV DS,AX
    LEA SI,LIST
    MOV CL,7
    MOV AL,[SI]
    TOP:
    MOV BL,[SI]
    CMP AL,BL
    JB NEXT:
    INC SI
    LOOP TOP
    JMP EXIT
    NEXT:
    MOV AL,BL
    INC SI
    JMP TOP
EXIT:
MAIN ENDP
END MAIN
```

7. Prime Non-Prime Check:
   Solution:

```
.MODEL SMALL
.STACK 100
.DATA
MSG0 DB "ENTER_THE_NUMBER:$"
MSG1 DB "PRIME$"
MSG2 DB "NOT_PRIME$"
.CODE
MAIN PROC
    MOV AX,@DATA
    MOV DS,AX
    LEA DX,MSG0
    MOV AH,09H
    INT 21H
    MOV AH,01H
    INT 21H
    SUB AL,30H

    CMP AL,02H
    JE PRIME
    CMP AL,01H
    JE EXIT
    CMP AL,00H
    JE EXIT

    MOV CL,AL
    MOV CH,AL
    SUB CH,01H
    MOV BL,2

    TOP:
    MOV AH,00H
    MOV AL,CL
    DIV BL
    CMP AH,00H
    JE NOT_PRIME
    CMP BL,CH
    JE PRIME
    INC BL
    JMP TOP
```

```
    NOT_PRIME:
    MOV AH,02H
    MOV DL,0AH
    INT 21H
    MOV AH,02H
    MOV DL,0DH
    INT 21H
    MOV AX,@DATA
    MOV DS,AX
    LEA DX,MSG2
    MOV AH,09H
    INT 21H
    JMP EXIT

    PRIME:
    MOV AH,02H
    MOV DL,0AH
    INT 21H
    MOV AH,02H
    MOV DL,0DH
    INT 21H
    MOV AX,@DATA
    MOV DS,AX
    LEA DX,MSG1
    MOV AH,09H
    INT 21H
    JMP EXIT

EXIT:
MOV AH,4CH
INT 21H
MAIN ENDP
END MAIN
```

8. Summation of 1-9 numbers:
   Solution:

```
.MODEL SMALL
.STACK 100
.CODE
MAIN PROC
MOV BL,01H ;Start of Adding Value
MOV AL,00H ;Initial Digit
MOV CL,09H ;Counter
TOP:
    ADD AL,BL ;Add
    DAA       ;Decimal Addition Adjust
    INC BL    ;Increment
LOOP TOP
MAIN ENDP
END MAIN
```

9. Display a String:
   Solution:

```
.MODEL SMALL
.STACK 100
.DATA
STRING1 DB "MARS$"
.CODE
MAIN PROC
    MOV AX,@DATA
    MOV DS,AX
    LEA DX,STRING1
    MOV AH,09H
    INT 21H
EXIT:
MAIN ENDP
END MAIN
```

10. String Upper to Lower and Lower to Upper:
    Solution:

```
.MODEL SMALL
.STACK 100
.DATA
STRING1 DB "hElLo$"
.CODE
MAIN PROC
    MOV AX,@DATA
    MOV DS,AX
    LEA SI,STRING1

    TOP:
    CMP [SI],'$'
    JZ EXIT
    CMP [SI],41H
    JL EXIT
    CMP [SI],5AH
    JLE DO_SMALL:
    CMP [SI],61H
    JL EXIT
    CMP [SI],7AH
    JG EXIT
    DO_CAPITAL:
    MOV BL,[SI]
    SUB BL,20H
    MOV [SI],BL
    JMP NEXT:
    DO_SMALL:
    MOV BL,[SI]
    ADD BL,20H
    MOV [SI],BL
    NEXT:
    INC SI
    JMP TOP

EXIT:
LEA DX,STRING1
MOV AH,09
INT 21H
MAIN ENDP
END MAIN
```

11. Display String Vertically
    Solution:

```
.MODEL SMALL
.STACK 100
.DATA
STRING1 DB "HELLO$"
.CODE
MAIN PROC
    MOV AX,@DATA
    MOV DS,AX
    LEA SI,STRING1

    TOP:
    CMP [SI],'$'
    JZ EXIT
    MOV AH,02H
    MOV DL,[SI]
    INT 21H

    MOV AH,02H
    MOV DL,13
    INT 21H
    MOV AH,02H
    MOV DL,10
    INT 21H
    INC SI
    JMP TOP

EXIT:
MAIN ENDP
END MAIN
```

12. String Reverse Display
    Solution:
12.1. Using PUSH POP instruction:

```
    .MODEL SMALL
    .STACK 100
    .DATA
    STRING1 DB "HELLO"
    .CODE
    MAIN PROC
        MOV AX,@DATA
        MOV DS,AX
        LEA SI,STRING1
        MOV CL,05H
        TOP:
        PUSH [SI]
    INC SI
        LOOP TOP
        MOV CL,00H
        MOV CL,05H
        LOOP1:
        POP DX
        MOV AH,02H
        INT 21H
        LOOP LOOP1

    EXIT:
    MAIN ENDP
END MAIN
```

12.2. Using Direction Flag SET

```
    .MODEL SMALL
    .STACK 100H
    .DATA
    STRING1 DB 'HELLO'
    STRING2 DB 5 DUP(?)
    .CODE
    MAIN PROC
        MOV AX,00H
        MOV AX,@DATA
        MOV DS,AX
        MOV ES,AX
        LEA SI,STRING1
        LEA DI,STRING2
        STD
        TOP:
        CMP DI,00H
        JE EXIT
        MOVSB
        MOV DL,[DI]
        MOV AH,2
        INT 21H
        JMP TOP
        EXIT:
    MAIN ENDP
END MAIN
```

13. Multiple Same letter Finding in a String
    Solution:

```
.MODEL SMALL
.STACK 100
.DATA
STRING1 DB 'ZAMAAAN$'
.CODE
MAIN PROC
    MOV AX,@DATA
    MOV DS,AX
    LEA SI,STRING1
    MOV AH,01H
    INT 21H
    MOV BL,AL
    MOV CL,00H
    TOP:
        CMP [SI],'$'
        JZ EXIT
        CMP [SI],BL
        JNE NEXT
        COUNT:
        INC CL
        NEXT:
        INC SI
    JMP TOP
EXIT:
MOV AH,02H
MOV DL,13
INT 21H
MOV AH,02H
MOV DL,10
INT 21H
MOV AH,02H
ADD CL,30H
MOV DL,CL
INT 21H
MAIN ENDP
END MAIN
```

14. Mismatch between two Strings
    Solution:

```
.MODEL SMALL
.STACK 100
.DATA
STRING1 DB "HELLO$"
STRING2 DB "HEPLO$"
STRING3 DB "NO MISMATCH BETWEEN TWO
STRINGS$"
STRING4 DB "THERE IS MISMATCH BETWEEN TWO
STRINGS$"
.CODE
MAIN PROC
    MOV AX,@DATA
    MOV DS,AX
    MOV ES,AX
    LEA SI,STRING1
    LEA DI,STRING2

    TOP:
    CMP [SI],'$'
    JZ DONE
    LODSB
    CMP [DI],AL
    JNE NOT_DONE
    INC DI
    JMP TOP
    DONE:
    MOV AX,@DATA
    MOV DS,AX
    LEA DX,STRING3
    MOV AH,09H
    INT 21H
    JMP EXIT
    NOT_DONE:
    MOV AX,@DATA
    MOV DS,AX
    LEA DX,STRING4
    MOV AH,09H
    INT 21H
    JMP EXIT
EXIT:
MAIN ENDP
END MAIN
```

15. Put Large Number on CL Register
    Solution:

```
.MODEL SMALL
.STACK 100H
.DATA
.CODE
MAIN PROC
    MOV AH,01H
    INT 21H
    SUB AL,30H
    MOV BL,AL
    MOV AH,01H
    INT 21H
    SUB AL,30H

    CMP BL,AL
    JE EQUAL
    JG LARGE_BL
    LARGE_AL:
    MOV CL,AL
    JMP EXIT
    LARGE_BL:
    MOV CL,BL
    JMP EXIT
    EQUAL:
    MOV CL,AL
    EXIT:
    MAIN ENDP
END MAIN
```

16. Addition of 25,12,15,10,11

Solution:

```
.MODEL SMALL
.STACK 100
.DATA
LIST DB 25H,12H,15H,10H,11H
.CODE
MAIN PROC
    MOV AX,@DATA
    MOV DS,AX
    LEA DX,LIST
    MOV CL,5
    MOV AL,00H
    TOP:
    ADD AL,[SI]
    DAA
    INC SI
    LOOP TOP
MAIN ENDP
END MAIN
```

17. BCD to Binary Conversion

Solution:

```
.MODEL SMALL
.STACK 100H
.DATA
 BCD DW 0456H
 BIN DW ?
.CODE
MAIN PROC
DATA1 EQU 0456H
MOV AX,DATA1
AND AX,0FF00H
MOV CH,AH
MOV AX,00H
MOV DX,00H
MOV BL,64H
MOV AL,CH
MUL BL
MOV DX,AX

MOV AX,DATA1
AND AX,00FFH
AND AL,0F0H
ROL AL,04H
MOV CH,AL
MOV AX,00H
MOV BL,0AH
MOV AL,CH
MUL BL
ADD DX,AX
```

```
MOV AX,DATA1
AND AX,00FFH
AND AL,0FH
MOV CH,AL
MOV AX,00H
MOV BL,01H
MOV AL,CH
MUL BL
ADD DX,AX

MOV AX,@DATA
MOV DS,AX
MOV BIN,DX
MAIN ENDP
END MAIN
```

19. $1^2 + 3^2 + 5^2 + 7^2$

Solution:

```
.MODEL SMALL
.STACK 100H
.DATA
.CODE
MAIN PROC
    MOV AL,01H
    MOV CX,05H
    TOP:
    MOV DL,AL
    MUL AL
    ADD BX,AX
    MOV AL,DL
    ADD AL,02H
    LOOP TOP
    MOV AX,BX
    DAA
MAIN ENDP
END MAIN
```