**CSE 331L / EEE 332L**

**Microprocessor Interfacing & Embedded System**

Section: 5,6 & 7, Fall 2021

Lab- 05: Library

---

## Library: emu8086.inc

To use any of the functions in emu8086.inc you should have the following line in the beginning of your source file:

include 'emu8086.inc'

**emu8086.inc** defines the following **macros**:

| | |
|---|---|
| **PUTC char** | macro with 1 parameter, prints out an ASCII char at current cursor position. |
| **GOTOXY col, row** | macro with 2 parameters, sets cursor position. |
| **PRINT string** | macro with 1 parameter, prints out a string. |
| **PRINTN string** | macro with 1 parameter, prints out a string. The same as PRINT but automatically adds "carriage return" at the end of the string. |

```
05 PRINT "HELLO"    ;prints the string
06 PUTC 'W'         ;prints 'W' next to 'O'
07
08 GOTOXY 20, 3     ;go to the given position
09
10 PRINTN "HELLO"   ;prints the string at the cursor position (20, 3) and
11                  ;prints a carriage return and newline
12
13 PUTC 65          ;prints the char whose ascii is 65
14
```

emu8086.inc also defines the following procedures:

- **PRINT_STRING** - procedure to print a null terminated string at current cursor position, receives address of **string in DS:SI register**. To use it declare: **DEFINE_PRINT_STRING** before END directive.
- **PTHIS** - procedure to print a null terminated string at current cursor position (just as PRINT_STRING), but receives address of **string from Stack**. The NULL TERMINATED string s**hould be defined just after the CALL instruction**. For example:

  ```
  CALL PTHIS
  db 'Hello World!', 0
  ```

  To use it declare: **DEFINE_PTHIS** before END directive.
- **GET_STRING** - procedure to get a null terminated string from the user, the received string is written to buffer at **DS:DI**, buffer **size should be in DX**. Procedure **stops the input when 'Enter' is pressed**. To use it declare: DEFINE_GET_STRING before END directive.
- **CLEAR_SCREEN** - procedure to clear the screen, (done by scrolling the entire screen window), and set cursor position to top of it. To use it declare: DEFINE_CLEAR_SCREEN before END directive.
- **SCAN_NUM** - procedure that gets the multi-digit SIGNED number from the keyboard, and **stores the result in CX register**. To use it declare: DEFINE_SCAN_NUM before END directive.
- **PRINT_NUM** - procedure that prints **a signed number in AX register**. To use it declare: DEFINE_PRINT_NUM and DEFINE_PRINT_NUM_UNS before END directive.
- **PRINT_NUM_UNS** - procedure that prints out an unsigned number **in AX register.** To use it declare: DEFINE_PRINT_NUM_UNS before END directive.

To use any of the above procedures you should first declare the function in the bottom of your file (but before the **END** directive), and then use **CALL** instruction followed by a procedure name. For example:

```
INCLUDE 'EMU8086.INC'
.MODEL SMALL
.STACK 100H
.DATA
    MSG DB  "This is 'print_string' procedure", 0


.CODE
    MOV AX, @DATA
    MOV DS, AX

    LEA SI, MSG             ;load offset of the string to SI to print
    CALL PRINT_STRING

    CALL PTHIS              ;print the string declared in code segment
    DB 0AH, 0DH, "This is 'pthis' procedure", 0AH, 0DH, 0

    MOV DX, 5               ;place the input buffer size
    CALL GET_STRING         ;input string

    CALL CLEAR_SCREEN       ;clear the screen and set cursor position at (0,0)

    CALL SCAN_NUM           ;get any decimal number (can be multidigit), ends with an enter

    CALL PTHIS              ;print a newline and cret
    DB 0AH, 0DH, 0

    MOV AX, CX              ;move the input number from cx to ax to print
    CALL PRINT_NUM_UNS      ;unsigned number print

    EXIT:
    MOV AH, 4CH
    INT 21H

;------------------DEFINE LIBRARY PROCUDERS-------------------------;

    DEFINE_PRINT_STRING
    DEFINE_PTHIS
    DEFINE_GET_STRING
    DEFINE_CLEAR_SCREEN
    DEFINE_SCAN_NUM
    DEFINE_PRINT_NUM
    DEFINE_PRINT_NUM_UNS

END
```

Example: take two numbers as input and display their summation.

```
01 INCLUDE 'EMU8086.INC'
02 .MODEL SMALL
03 .STACK 100H
04 .DATA
05     MSG1 DB 'Enter a number: ', 0
06     MSG2 DB 0AH, 0DH, 'Enter another number: ', 0
07 .CODE
08     MOV AX, @DATA
09     MOV DS, AX
10
11     LEA     SI, MSG1        ; ask for the number
12     CALL    PRINT_STRING
13     CALL    SCAN_NUM        ; get 1st number in CX.
14
15     MOV     AX, CX          ; copy the number to AX.
16
17     LEA     SI, MSG2        ; ask for the number
18     CALL    PRINT_STRING
19     CALL    SCAN_NUM        ; input 2nd number in CX
20
21     ADD AX, CX              ; sum in AX
22
23     CALL    PTHIS
24     DB  13, 10, 'The sum is: ', 0
25
26     CALL    PRINT_NUM       ; print number in AX
27
28     EXIT:                   ; return to operating system.
29     MOV AH, 4CH
30     INT 21H
31 ;-----------------define-----------------------------------;
32     DEFINE_SCAN_NUM
33     DEFINE_PRINT_STRING
34     DEFINE_PRINT_NUM
35     DEFINE_PRINT_NUM_UNS   ; required for print_num.
36     DEFINE_PTHIS
37 END                    ; directive to stop the compiler.
```

Example: Read a string using GET_STRING and store it in memory.

```
INCLUDE 'EMU8086.INC'

.MODEL SMALL
.STACK 100H
.DATA
    MSG DB  "This is 'print_string' procedure", 0
    STR DB 5 DUP(?)

.CODE
    MOV AX, @DATA
    MOV DS, AX

    MOV DX, 5                   ;declare the buffer size
    CALL GET_STRING             ;input string
    LEA BX, STR                 ;load the offset of memory into index reg.

    COPY:                       ;copy the input from buffer to memory
    MOV AL, [DI]                ;copy each char of the string individually
    MOV [BX], AL

    INC BX                      ;update memory offset
    INC DI                      ;update DI to get the next user input char

    CMP [DI], 0                 ;check if it is the end of user input
    JNE COPY

    MOV [BX], 0                 ;place null character at the end of the string

    CALL PTHIS                  ;print newl
    DB 0AH, 0DH, 0

    LEA SI, STR                 ;load offset of the string to SI to print
    CALL PRINT_STRING

    EXIT:
    MOV AH, 4CH
    INT 21H

;-------------------DEFINE LIBRARY PROCUDERS--------------------------;

    DEFINE_PRINT_STRING
    DEFINE_PTHIS
    DEFINE_GET_STRING

END
```