Faculdade de Computação e Engenharia Elétrica
Microprocessadores e Microcontroladores – Prof. Dr. Elton Alves
Experimento 5 – Programação Assembly no 8086/8088

- **Objetivo:**

- Utilizar as instruções macro, bibliotecas, valores negativos e funções externas no Emu8086.

- **Gerar a biblioteca BIBLIO.inc, conforme código a seguir.**

```
ESCREVA MACRO
PUSH AX
MOV AH, 9h
INT 21h
POP AX
ENDM
```

- **Em seguida o arquivo BIBLIO.inc deve ser incluído no código 13 a seguir:**

```
TITLE Teste de Segmento 5     ; identificacao de nome interno do programa

#MAKE_EXE#  ; tipo de arquivo a ser gerado .EXE
; determina a area de dados - SEGMENT e ENDS com parametro DATA

INCLUDE 'BIBLIO.inc'

DADOS SEGMENT 'DATA'
mensagem DB "Ola, Mundo$"
DADOS ENDS

PILHA SEGMENT STACK 'STACK'  ;define o tamanho da pilha
DW 0100h DUP(?)
PILHA ENDS

CODIGO SEGMENT 'CODE'
ASSUME CS:CODIGO, DS:DADOS, SS:PILHA
INICIO PROC FAR
MOV AX, DADOS
MOV DS, AX
MOV ES, AX
MOV DX, OFFSET mensagem
ESCREVA
MOV AH, 4Ch
INT 21h
RET
INICIO ENDP
```

```
CODIGO ENDS
END INICIO
```

- **Rodar o código 14 no Emu8086**

```
;transformar o procedimento mensagem na macro msg
;calculo da fatorial de um valor positivo entre 0 e 8 (16bits)

;*******************************
;* Programa: LACO3.ASM *
;*******************************
org 100h
.DATA
msg1 DB 'Entre valor decimal positivo (de 0 ate 8): ', 24h
msg2 DB 0Dh, 0Ah, 'Fatorial de ', 24h
msg3 DB ' equivale a ', 24h
msg4 DB 0Dh, 0Ah, 'Valor invalido', 24h

.CODE
LEA DX, msg1
msg
CALL entrada  ; chamada do procedimento entrada
PUSH AX       ; armazenamento da pilha do valor atual de AX (Fornecido pelo teclado)

LEA DX, msg2  ; variavel msg2
msg
POP AX  ; movimentacao do valor trazido de AL para  DL (valor que sera apresentado)
MOV DL, AL
MOV AH, 0Eh
INT 10h
SUB AL, 30h
MOV CL, AL
LEA DX, msg3   ; variavel ms3
msg
CALL fatorial
CALL valor

fim:
INT 20h
msg MACRO
MOV AH, 09h  ; exibicao da string
INT 21h
ENDM

entrada PROC NEAR
MOV AH, 01h
INT 21h
CMP AL, 30h
JL erro
```

```
CMP AL, 39h
JGE erro
JMP fim_validacao
erro:
LEA DX, msg4
msg
JMP fim
fim_validacao:

RET
entrada ENDP

fatorial PROC NEAR
MOV AX, 01h  ; menor resultado valido
CMP CX, 0h   ; verifica se e zero
JE fim_laco  ;desvia para fim_laco
repita1:   ; se nao for zero, calcul o fatorial
MUL CX
LOOPNE repita1
fim_laco:
RET
fatorial ENDP

valor PROC NEAR
PUSH AX
MOV BX, 0Ah  ; carregado com valor 10 (auxilia para conversao em notacao decimal)
SUB CX, CX  ; zerando CX
repita2:
SUB DX, DX  ; zerando DX
DIV BX
PUSH DX  ; armazena da pilha o valo de DX
INC CX     ; incremento de CX
CMP AX, 0h  ; comparacao do valor de AX
JNZ repita2  ;desvio caso o valor  de AX nao seja zero
saida:
POP AX
ADD AL, 30h
MOV DL, AL
MOV AH, 0Eh
INT 10h
DEC CX
JNBE saida
POP DX
RET
valor ENDP
```

- **Rodar o código 15 no Emu8086**

```
TITLE Teste de Segmento 7

#MAKE_EXE#

DADOS SEGMENT 'DATA'
msg1 DB 'Entre valor 1 (de 0 a 9) .....: ', 24h
msg2 DB 0Dh, 0Ah, 'Entre valor 2 (de 0 a 9) .....: ', 24h
msg3 DB 0Dh, 0Ah, 'Resultado .....: ', 24h
msg4 DB 0Dh, 0Ah, 'Valor invalido', 24h
DADOS ENDS

PILHA SEGMENT STACK 'STACK'
DW 0100h DUP(?)
PILHA ENDS

CODIGO SEGMENT 'CODE'
ASSUME CS:CODIGO, DS:DADOS, SS:PILHA
INICIO PROC FAR
MOV AX, DADOS
MOV DS, AX
MOV ES, AX

MOV DX, OFFSET msg1
MSG
CALL entrada
MOV BH, AL

MOV DX, OFFSET msg2
MSG
CALL entrada
MOV BL, AL

MOV DX, OFFSET msg3
MSG
SUB BH, BL ; Se o valor BH<BL, altera SF=1, indicando BH como negativo.
JS negativo ; verifica se SF= 1, fazendo o desvio de programa
JGE positivo ; verifica se o valor de BH>=BL, se V desvio de programa

negativo:
NEG BH   ; transforma o valor negativo (complemento de dois) em seu equivalente
positivo
MOV AL, 2Dh ; movimenta o valor 2Dh para o AL (ASCII   2Dh = -)
MOV AH, 0Eh
INT 10h
JMP mostra ; mostra e apresenta o valor numerico armazenado na memoria

positivo:
JMP mostra
```

```
mostra:
MOV AL, BH
MOV DL, AL
ADD AL, 30h
MOV AH, 0Eh
INT 10h
FIM
RET
INICIO ENDP
CODIGO ENDS

fim MACRO
MOV AH, 4Ch
INT 21h
ENDM

msg MACRO
MOV AH, 09h
INT 21h
ENDM

entrada PROC NEAR
MOV AH, 01h
INT 21h
CMP AL, 30h
JL erro
CMP AL, 40h
JGE erro
JMP fim_validacao

erro:
LEA DX, msg4
MSG
FIM
fim_validacao:
SUB AL, 30h
RET
entrada ENDP

END INICIO
```

- **Rodar o código 16 no Emu8086**

```
;*********************************
;* Programa: BIBLIOT8.ASM *
;*********************************
INCLUDE 'emu8086.inc'
org 100h
.DATA
```

```
tamanho EQU 30d + 1d  ; diretiva EQU permite constantes ; valor da constante 30d+1d
= 31d
idade DW 0
buffer DB tamanho DUP ('x')    ; tamanho da EQU
msg1 DB 'Entre seu nome ....: ', 0
msg2 DB 'Entre sua idade ...: ', 0
msg3 DB 'Ola, ', 0
msg4 DB ' voce tem ', 0
msg5 DB ' anos.', 0
.CODE

LEA SI, msg1
CALL PRINT_STRING ; manipulacao Entre com seu nome
LEA DI, buffer ; deslocamente da area do buffer
MOV DX, tamanho ; armazenamento da area do buffer
CALL GET_STRING  ; solicitacao do nome pela linha
PUTC 13d  ; programa retorna
PUTC 10d  ; programa muda de linha

LEA SI, msg2   ; exibe a mensagem 2
CALL PRINT_STRING
CALL SCAN_NUM
MOV idade, CX
PUTC 13d
PUTC 10d

CALL CLEAR_SCREEN   ; limpa a tela
LEA SI, msg3
CALL PRINT_STRING
MOV SI, DI  ; escrita da sequencia de caracter no buffer utilizando DS:SI

CALL PRINT_STRING  ; apresenta uma sequencia de caracter
LEA SI, msg4
CALL PRINT_STRING
MOV AX, idade
CALL PRINT_NUM ; apresenta os valores negativos
LEA SI, msg5
CALL PRINT_STRING
INT 20h

DEFINE_PRINT_STRING
DEFINE_GET_STRING
DEFINE_SCAN_NUM
DEFINE_CLEAR_SCREEN
DEFINE_PRINT_NUM
DEFINE_PRINT_NUM_UNS  ; apresenta valores positivos em AX
END
```

- **Rodar o código 17 no Emu8086**

```
;*******************************
;* Programa: BIBLIOT8.ASM *
;*******************************
INCLUDE 'emu8086.inc'
org 100h
.DATA
tamanho EQU 30d + 1d  ; diretiva EQU permite constantes ; valor da constante 30d+1d
= 31d
idade DW 0
buffer DB tamanho DUP ('x')    ; tamanho da EQU
msg1 DB 'Entre seu nome ....: ', 0
msg2 DB 'Entre sua idade ...: ', 0
msg3 DB 'Ola, ', 0
msg4 DB ' voce tem ', 0
msg5 DB ' anos.', 0
.CODE

LEA SI, msg1
CALL PRINT_STRING ; manipulacao Entre com seu nome
LEA DI, buffer ; deslocamente da area do buffer
MOV DX, tamanho ; armazenamento da area do buffer
CALL GET_STRING  ; solicitacao do nome pela linha
PUTC 13d  ; programa retorna
PUTC 10d  ; programa muda de linha

LEA SI, msg2   ; exibe a mensagem 2
CALL PRINT_STRING
CALL SCAN_NUM
MOV idade, CX
PUTC 13d
PUTC 10d

CALL CLEAR_SCREEN   ; limpa a tela
LEA SI, msg3
CALL PRINT_STRING
MOV SI, DI  ; escrita da sequencia de caracter no buffer utilizando DS:SI

CALL PRINT_STRING  ; apresenta uma sequencia de caracter
LEA SI, msg4
CALL PRINT_STRING
MOV AX, idade
CALL PRINT_NUM ; apresenta os valores negativos
LEA SI, msg5
CALL PRINT_STRING
INT 20h

DEFINE_PRINT_STRING
DEFINE_GET_STRING
```

```
DEFINE_SCAN_NUM
DEFINE_CLEAR_SCREEN
DEFINE_PRINT_NUM
DEFINE_PRINT_NUM_UNS  ; apresenta valores positivos em AX
END
```

- **Rodar o código 18 no Emu8086**

```
;*******************************
;* Programa: BIBLIOT4.ASM *
;*******************************
INCLUDE 'emu8086.inc'

.MODEL small
.STACK 512d

.CODE
PRINTN 'Alo Mundo 1'
PRINTN 'Alo Mundo 2'
PRINT 'Alo Mundo 3'
PRINT 'Alo Mundo 4'

INT 20h
END
```

- **Rodar o código 19 no Emu8086**

```
;*******************************
;* Programa: DECISAO1.ASM *
;*******************************
INCLUDE 'emu8086.inc'  ; chamada da biblioteca emu8086.inc

org 100h

.DATA
valor DW 0
msg1 DB 'Entre um valor numerico: ', 0   ; valor da variavel e 0; caso queira sem valor = ?
msg2 DB 'Valor acima de 10.', 0

.CODE
LEA SI, msg1  ; aponta para SI(acesso ao conteudo da string)o endereco de memoria da variavel msg1
CALL PRINT_STRING  ;  acessa o conteudo do registrador SI
CALL SCAN_NUM  ; efetua a leitura do teclado
MOV valor, CX ; armazena o valor do teclado em CX
PUTC 13d
PUTC 10d
se:
CMP valor, 10d; compara o conteudo do valor com 10
```

```
JLE fim_se  ; verifica se a condicao e V ou F ; valor>10; executa 23 ate 25
entao:
LEA SI, msg2 ; se V apresenta a mensagem msg2.
CALL PRINT_STRING

fim_se:  ; desvia se valor <10
INT 20h
DEFINE_PRINT_STRING ;   biblioteca emu8086.inc
DEFINE_SCAN_NUM     ; biblioteca emu8086.inc
END
```

- **Atividade Avaliativa**

1. Desenvolva um programa que solicite a entrada de dois valores numéricos decimais positivos de um digito, subtraia os valores e apresente o resultado da operação como sendo um valor decimal. Para isso desenvolva um programa .EXE.

- Data da entrega: 30/06/2021
- Formato de relatório com os códigos e exibição dos resultados das atividades avaliativas.