Universidade Federal do Sul e Sudeste do Pará
Faculdade de Computação e Engenharia Elétrica
Curso de Engenharia da Computação
Iago Costa das Flores

Microprocessadores e Microcontroladores
Experimento 3 – Programação Assembly no 8086/8088

Marabá
2021

Iago Costa das Flores

Microprocessadores e Microcontroladores
Experimento 3 – Programação Assembly no 8086/8088

Relatório apresentado no curso de Engenharia da Computação, turma de 2018 como obtenção de nota parcial na disciplina de microprocessadores e microcontroladores, ministrada pelo Professor Dr. Elton Alves.

Marabá
2021

Sumário

1 - Introdução

O Trabalho visa apresentar os códigos fontes e resultados de execução das três atividades avaliativas a seguir:

1. Desenvolva um código utilizando JMP para salto incondicional de seu nome.

2. No experimento 10, experimente trocar os valores das variáveis “a” e “b” e executar o programa para verificar a forma de funcionamento com valores diferentes.

2 - Atividades

As atividades demonstradas a seguir foram feitas com a ajuda do programa emu8086 para escrever, compilar e executar os códigos em assembly.

2.1 - Desenvolva um código utilizando JMP para salto incondicional de seu nome.

```
org 100h ; 64Kbytes

.DATA ; 0Dh, 0Ah e 24h mudanca de linha de cursor apos escrita da mensagem

nome1 DB 'Iago ', 0Dh, 0Ah, 24h ; 0Dh tecla <ENTER>

nome2 DB 'Costa ', 0Dh, 0Ah, 24h ;

nome3 DB 'das ', 0Dh, 0Ah, 24h ;

nome4 DB 'Flores ', 0Dh, 0Ah, 24h ;


.CODE

JMP salto4 ; salta para a linha 33

saltol:
LEA DX, nome4 ; envio nome4
CALL escreva  ; chama do procedimento escreva pelo CALL
JMP saida ;salta para a linha 38

salto2:
LEA DX, nome3 ; envio nome3
```

CALL escreva ; chama do procedimento escreva pelo CALL
JMP saltol ; salta para a linha 18

salto3:
LEA DX, nome2 ; envio nome2
CALL escreva ; chama do procedimento escreva pelo CALL
JMP salto2 ; salta para a linha 23

salto4:
LEA DX, nome1 ; envio nome1
CALL escreva ; chama do procedimento escreva pelo CALL
JMP salto3 ; salta para a linha 28

saida:
INT 20h ; Finaliza o programa

escreva PROC NEAR ; procedimento escreva
MOV AH, 09h ;apresentacao do caracter 09h
INT 21h; interrupcao 21h
RET ; retorno do procedimento para a 1 linha após sua chamada (30)

Na figura 01 temos o código comentado do algoritmo do JMP. Estão separados o .DATA e o .CODE. Na figura 01 também é possível notar que foram usadas quatro mensagens, cada uma com seu jump correspondente. Cada mensagem é um pedaço do nome.

```
01 org 100h ; 64Kbytes
02
03 .DATA ; 0Dh, 0Ah e 24h mudanca de linha de cursor apos escrita da mensagem
04
05 nome1 DB 'Iago ', 0Dh, 0Ah, 24h ; 0Dh tecla <ENTER>
06
07 nome2 DB 'Costa ', 0Dh, 0Ah, 24h ;
08
09 nome3 DB 'das ', 0Dh, 0Ah, 24h ;
10
11 nome4 DB 'Flores ', 0Dh, 0Ah, 24h ;
12
13
14 .CODE
15
16 JMP salto4 ; salta para a linha 33
17
18 saltol:
19 LEA DX, nome4 ; envio nome4
20 CALL escreva  ; chama do procedimento escreva pelo CALL
21 JMP saida ;salta para a linha 38
22
23 salto2:
24 LEA DX, nome3 ; envio nome3
25 CALL escreva  ; chama do procedimento escreva pelo CALL
26 JMP saltol ; salta para a linha 18
27
28 salto3:
29 LEA DX, nome2 ; envio nome2
30 CALL escreva ; chama do procedimento escreva pelo CALL
31 JMP salto2 ; salta para a linha 23
32
33 salto4:
34 LEA DX, nome1 ; envio nome1
35 CALL escreva ; chama do procedimento escreva pelo CALL
36 JMP salto3 ; salta para a linha 28
37
38 saida:
39 INT 20h ; Finaliza o programa
40
41 escreva PROC NEAR ; procedimento escreva
42 MOV AH, 09h ;apresentacao do caracter 09h
43 INT 21h; interrupcao 21h
44 RET ; retorno do procedimento para a 1 linha apos sua chamada (30)
45 escreva ENDP
```

Figura 01: Código fonte da atividade 01

Na figura 02, temos a execução da atividade 01, onde mostra - se todas as partes do nome imprimidas na tela utilizando o salto incondicional.

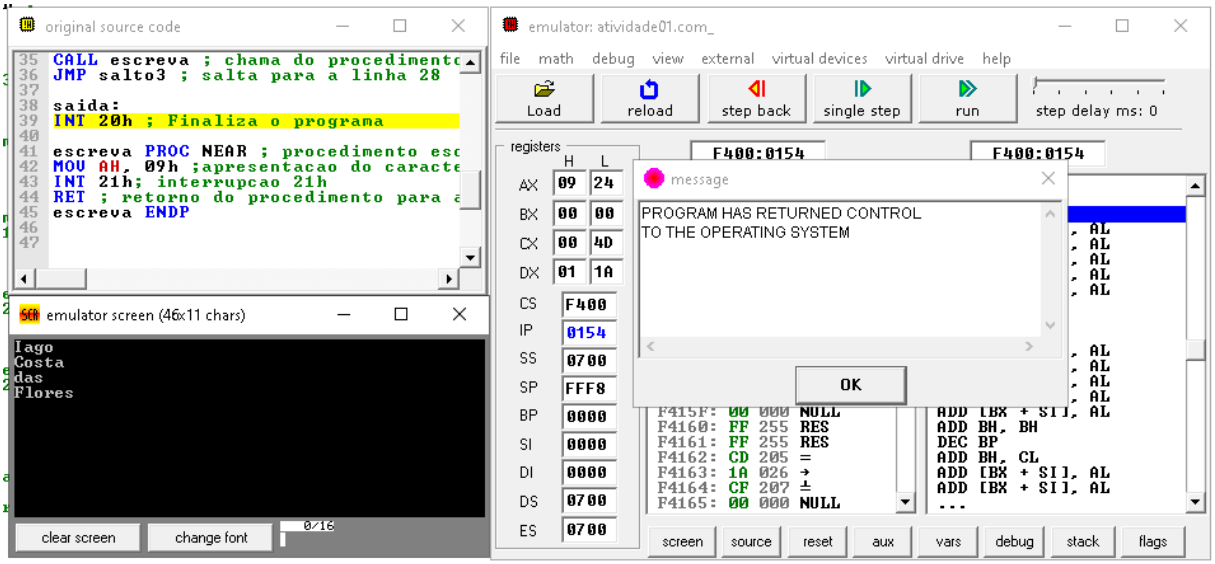

Figura 02: Execução da atividade 01

2.2 - No experimento 10, experimente trocar os valores das variáveis “a” e “b” e executar o programa para verificar a forma de funcionamento com valores diferentes.

```
.MODEL small
.STACK 512d
.DATA

a DB 8d ;8 decimal
b DB 3d

.CODE

MOV AX, @DATA
MOV DS, AX

MOV AL, a
MOV BL, b

CMP AL, BL ; SE (AL=BL) ENTAO
JE entao ; salto (=)
JMP fimse ; se V desvia para linha 26

entao:
```

```
INC AL ; incremento do valor Olh a AL
CALL apoio ; chamada do procedimento apoio
MOV DL, AL ; transferencia do valor AL para DL (07h)
CALL escreva ; chamada do procedimento escreva

fimse: ;fechamento
DEC BL ;decremento 01h sobre BL (05h)
CALL apoio ; chama o procedimento apoio
MOV DL, BL ; movimentacao do valor
CALL escreva ; chamada do procedimento escreva

INT 20h

escreva PROC NEAR
ADD DL, 30h
CMP DL, 39h
JLE valor
ADD DL, 07h

valor:
INT 21h
RET
escreva ENDP


apoio PROC NEAR ; prepara o ambiente para apresentação de um caracter por vez
MOV AH, 02h
MOV CL, 04h
SHR DL, CL
RET
apoio ENDP
```

Na figura 03 temos o código comentado do algoritmo. Estão separados o .DATA e o .CODE. Foi modificado o valor das variáveis “a = 8d” e “b = 3d”. Quando executado o código ele pula os procedimentos do “entao”, pois a condicional “JE entao” não é verdadeira. Assim, o que acontece é apenas o bloco “JMP fimse” que decrementa uma unidade do que estiver armazenado na variável BL, que no caso, é a variável “b”.

```
.MODEL small
.STACK 512d
.DATA

a DB 8d ;8 decimal
b DB 3d

.CODE

MOV AX, @DATA
MOV DS, AX

MOV AL, a
MOV BL, b

CMP AL, BL ; SE (AL=BL) ENTAO
JE entao ; salto (=)
JMP fimse ; se V desvia para linha 26

entao:
INC AL ; incremento do valor 01h a AL
CALL apoio ; chamada do procedimento apoio
MOV DL, AL ; transferencia do valor AL para DL (07h)
CALL escreva ; chamada do procedimento escreva

fimse: ;fechamento
DEC BL ;decremento 01h sobre BL (05h)
CALL apoio ; chama o procedimento apoio
MOV DL, BL ; movimentacao do valor
CALL escreva ; chamada do procedimento escreva

INT 20h

escreva PROC NEAR
ADD DL, 30h
CMP DL, 39h
JLE valor
ADD DL, 07h

valor:
INT 21h
RET
escreva ENDP


apoio PROC NEAR ; prepara o ambiente para apresentacao de um caracter por vez
MOV AH, 02h
MOV CL, 04h
SHR DL, CL
RET
apoio ENDP
```

Figura 03: Código fonte da atividade 02

Na figura 04, temos a execução do código mostrando que o valor em BL foi decrementado referente a variável b=3d.

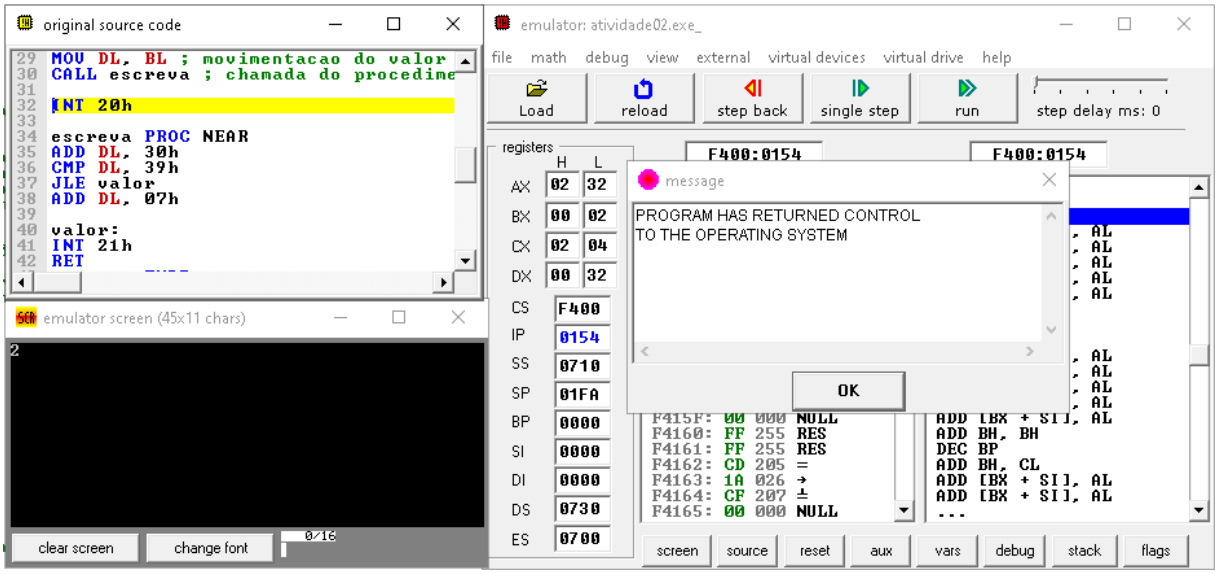

Figura 04: Execução da atividade 02

3 - Conclusão

Foram demonstradas as impressões do código e execução dos mesmos através das imagens apresentadas com suas devidas explicações. Os códigos fontes foram comentados e as 2 atividades foram feitas conforme o solicitado no comando do trabalho.

4 - Referências

MANZANO, J. A. **Programação assembly: padrão IBM - PC 8086/8088.** 6ªed. Ed.Erica, 2012.