

# Microprocessadores e Microcontroladores

## Programação Assembly

José Tarcísio Franco de Camargo

Entrada e saída  
Teclado e vídeo no simulador

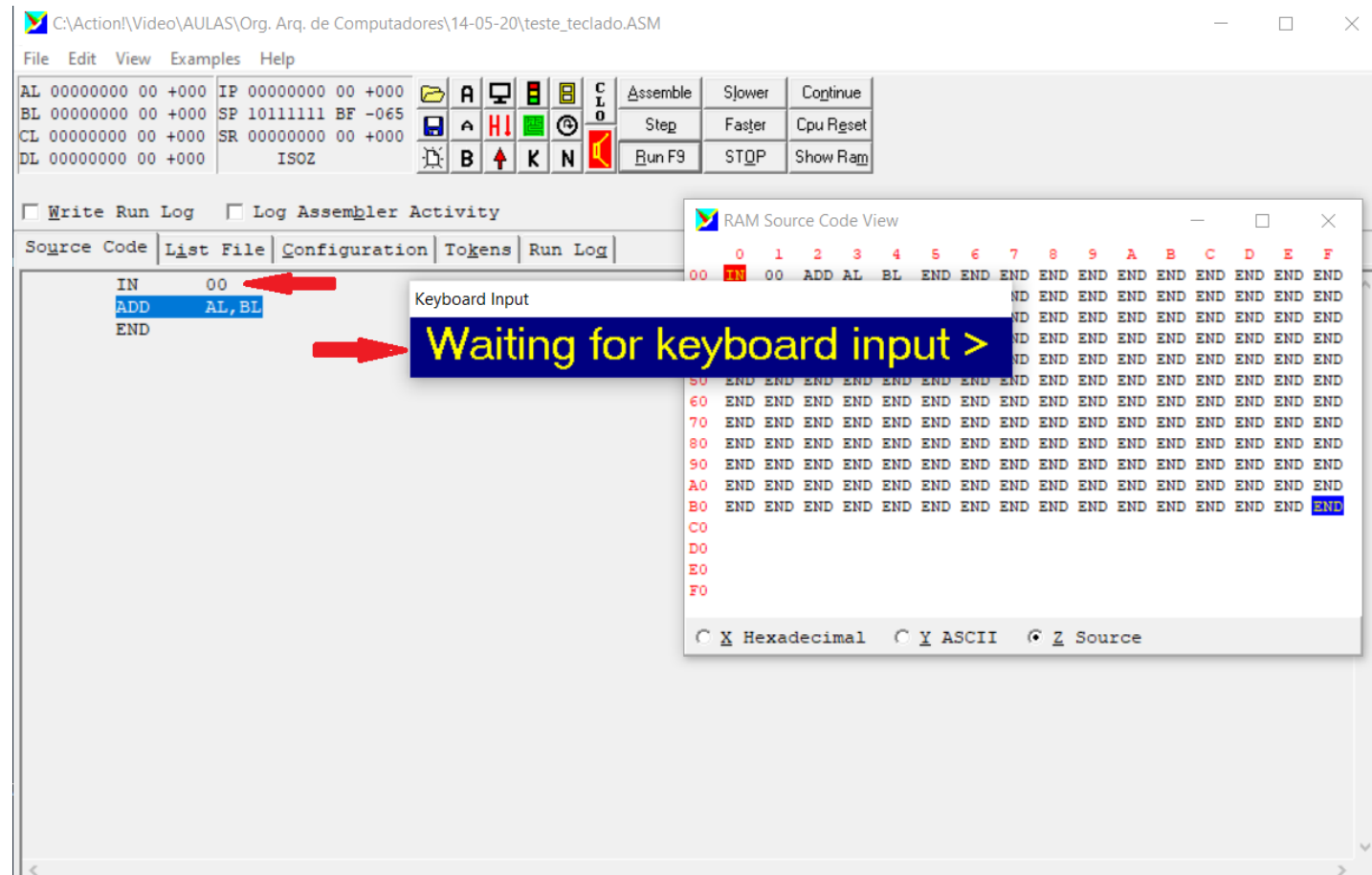
# O teclado do simulador – Versão 1

- O simulador possui um teclado associado à Porta 00.
- Para capturar um dado do teclado deve ser usada a instrução **IN 00** .
- Sempre que a instrução **IN 00** for utilizada, o programa ficará parado, aguardando que uma tecla seja pressionada.
- Após uma tecla ser pressionada, o programa continuará sua execução.

# O teclado do simulador – Versão 1

- Utilizando a instrução **IN 00** , quando uma tecla é pressionada, o equivalente ASCII da tecla pressionada é guardado no registrador AL.

# Procure conhecer a Tabela ASCII !



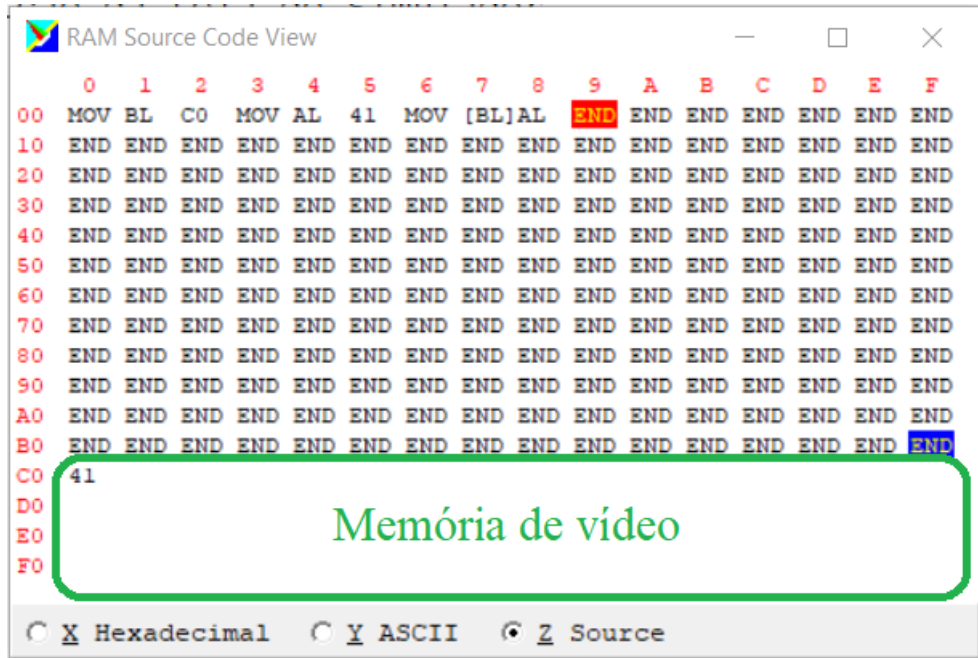
# A memória do simulador

- A memória do simulador é dividida em três áreas:
- Memória de programa e dados:
  - Vai do endereço  $00_H$  até o endereço  $AF_H$ .
  - É onde o usuário armazena seus dados e programas em execução.
- Pilha:
  - Vai do endereço  $BF_H$  até o endereço  $BO_H$ .
  - É uma área da memória onde dados são “empilhados” (armazenados) de forma temporária.

# A memória do simulador

- Memória de vídeo:
  - Vai do endereço  $C0_H$  até o endereço  $FF_H$ .
  - Esta área armazena caracteres que deverão ser apresentados na tela de vídeo do simulador.
  - Todos os bytes guardados nesta área da memória terão seus equivalentes ASCII mostrados na tela de vídeo do simulador.

# A memória de vídeo e a tela do simulador

[illegible]

# Memória de vídeo - Exemplo

The screenshot displays an x86-64 assembler simulator interface. The main window shows the assembly code for a program that stores the character 'A' in video memory. The code is as follows:

```
MOV BL, C0 ; O primeiro endereço da memória de vídeo  
; foi armazenado no registrador BL  
MOV AL, 41 ; O valor hexadecimal 41 representa o  
; caracter ASCII "A"  
MOV [BL], AL ; O valor armazenado em AL será guardado  
; na memória, no endereço indicado por BL  
; Como resultado, o caracter "A" será  
; mostrado na 1a. posição da tela do simulador.  
END
```

Below the code, a window titled "VDU CO..FF" shows the video memory. A red arrow points to the first position, which displays the character 'A'.

To the right, a window titled "RAM Source Code View" shows the memory layout. A red arrow points to the address 00000000, which contains the value 41 (hexadecimal) or 'A' (ASCII).

The RAM Source Code View window shows the following memory layout:

Address	Value
00	MOV BL C0
01	MOV AL 41
02	MOV [BL] AL
03	END
04	END
05	END
06	END
07	END
08	END
09	END
0A	END
0B	END
0C	END
0D	END
0E	END
0F	END
10	END
11	END
12	END
13	END
14	END
15	END
16	END
17	END
18	END
19	END
1A	END
1B	END
1C	END
1D	END
1E	END
1F	END
20	END
21	END
22	END
23	END
24	END
25	END
26	END
27	END
28	END
29	END
2A	END
2B	END
2C	END
2D	END
2E	END
2F	END
30	END
31	END
32	END
33	END
34	END
35	END
36	END
37	END
38	END
39	END
3A	END
3B	END
3C	END
3D	END
3E	END
3F	END
40	END
41	END
42	END
43	END
44	END
45	END
46	END
47	END
48	END
49	END
4A	END
4B	END
4C	END
4D	END
4E	END
4F	END
50	END
51	END
52	END
53	END
54	END
55	END
56	END
57	END
58	END
59	END
5A	END
5B	END
5C	END
5D	END
5E	END
5F	END
60	END
61	END
62	END
63	END
64	END
65	END
66	END
67	END
68	END
69	END
6A	END
6B	END
6C	END
6D	END
6E	END
6F	END
70	END
71	END
72	END
73	END
74	END
75	END
76	END
77	END
78	END
79	END
7A	END
7B	END
7C	END
7D	END
7E	END
7F	END
80	END
81	END
82	END
83	END
84	END
85	END
86	END
87	END
88	END
89	END
8A	END
8B	END
8C	END
8D	END
8E	END
8F	END
90	END
91	END
92	END
93	END
94	END
95	END
96	END
97	END
98	END
99	END
9A	END
9B	END
9C	END
9D	END
9E	END
9F	END
A0	END
A1	END
A2	END
A3	END
A4	END
A5	END
A6	END
A7	END
A8	END
A9	END
AA	END
AB	END
AC	END
AD	END
AE	END
AF	END
B0	END
B1	END
B2	END
B3	END
B4	END
B5	END
B6	END
B7	END
B8	END
B9	END
BA	END
BB	END
BC	END
BD	END
BE	END
BF	END
C0	41
C1	END
C2	END
C3	END
C4	END
C5	END
C6	END
C7	END
C8	END
C9	END
CA	END
CB	END
CC	END
CD	END
CE	END
CF	END
D0	END
D1	END
D2	END
D3	END
D4	END
D5	END
D6	END
D7	END
D8	END
D9	END
DA	END
DB	END
DC	END
DD	END
DE	END
DF	END
E0	END
E1	END
E2	END
E3	END
E4	END
E5	END
E6	END
E7	END
E8	END
E9	END
EA	END
EB	END
EC	END
ED	END
EE	END
EF	END
F0	END
F1	END
F2	END
F3	END
F4	END
F5	END
F6	END
F7	END
F8	END
F9	END
FA	END
FB	END
FC	END
FD	END
FE	END
FF	END



# Memória de vídeo – Exemplo 2

The image shows a screenshot of an x86-64 assembler simulator window. The title bar indicates the file path: C:\Action!\Video\AULAS\Org. Arq. de Computadores\14-05-20\testa\_video\_2.ASM. The window has a menu bar (File, Edit, View, Examples, Help) and a toolbar with icons for file operations and execution. Below the toolbar, there are checkboxes for "Write Run Log" and "Log Assembler Activity". The main window is divided into tabs: Source Code, List, File, Configuration, Tokens, and Run Log. The "Source Code" tab is active, displaying the following assembly code:

```
AL 00111010 3A +058 IP 00010010 12 +018
BL 11001010 CA -054 SP 10111111 BF -065
CL 00000000 00 +000 SR 00000010 02 +002
DL 00000000 00 +000      ISOZ
```

The code includes comments in Portuguese explaining the operations:

- `MOV BL,C0`: O primeiro endereço da memória de vídeo foi armazenado no registrador BL
- `MOV AL,30`: O valor hexadecimal 30 representa o caracter ASCII "0"
- `loop: MOV [BL],AL`: O valor armazenado em AL será guardado na memória, no endereço indicado por BL
- `INC AL`: Incrementa o conteúdo de AL
- `INC BL`: Incrementa o conteúdo de BL
- `CMP AL,3A`: Compara o valor de AL com "3A"; "3A" é o valor seguinte ao caracter "9"
- `JNZ loop`: Se não for "3A", mostra um novo caracter no vídeo do simulador
- `END`

In the bottom right corner, there is a small window titled "VDU CO.FF" which displays a video buffer with 16 slots, indexed 0 to 15. The slots are currently empty, representing a black screen.

# Juntando teclado e vídeo

```
C:\Action!\Video\AULAS\Org. Arq. de Computadores\14-05-20\teclado_video.ASM
File Edit View Examples Help
AL 00000000 00 +000 IP 00000000 00 +000
BL 00000000 00 +000 SP 10111111 BF -065
CL 00000000 00 +000 SR 00000000 00 +000
DL 00000000 00 +000 ISOZ
MOV BL,C0
Write Run Log Log Assembler Activity
Source Code | List File | Configuration | Tokens | Run Log |
MOV BL,C0 ; O primeiro endereço da memória de vídeo
; foi armazenado no registrador BL
IN 00 ; Captura um dado do teclado
; O valor capturado é guardado em AL
MOV [BL],AL ; O valor armazenado em AL será guardado
; na memória, no endereço indicado por BL
; Como resultado, o caracter "A" será
; mostrado na 1a. posição da tela do simulador.
END
```

# Sua tarefa

- Faça um programa que, em continuamente, capture caracteres do teclado e os apresente no vídeo do simulador.
- O 1º. caracter deve ser colocado no endereço  $C0_H$  da memória de vídeo; o 2º. caracter deve ser colocado no endereço  $C1_H$  da memória de vídeo e assim em diante.
- O programa deverá ser encerrado se o usuário digitar **ENTER** ou se a última posição da memória de vídeo (endereço  $FF_H$ ) for preenchida.