

Sistemas Operacionais

Simulador SMS - **S**imulator for **S**tudents



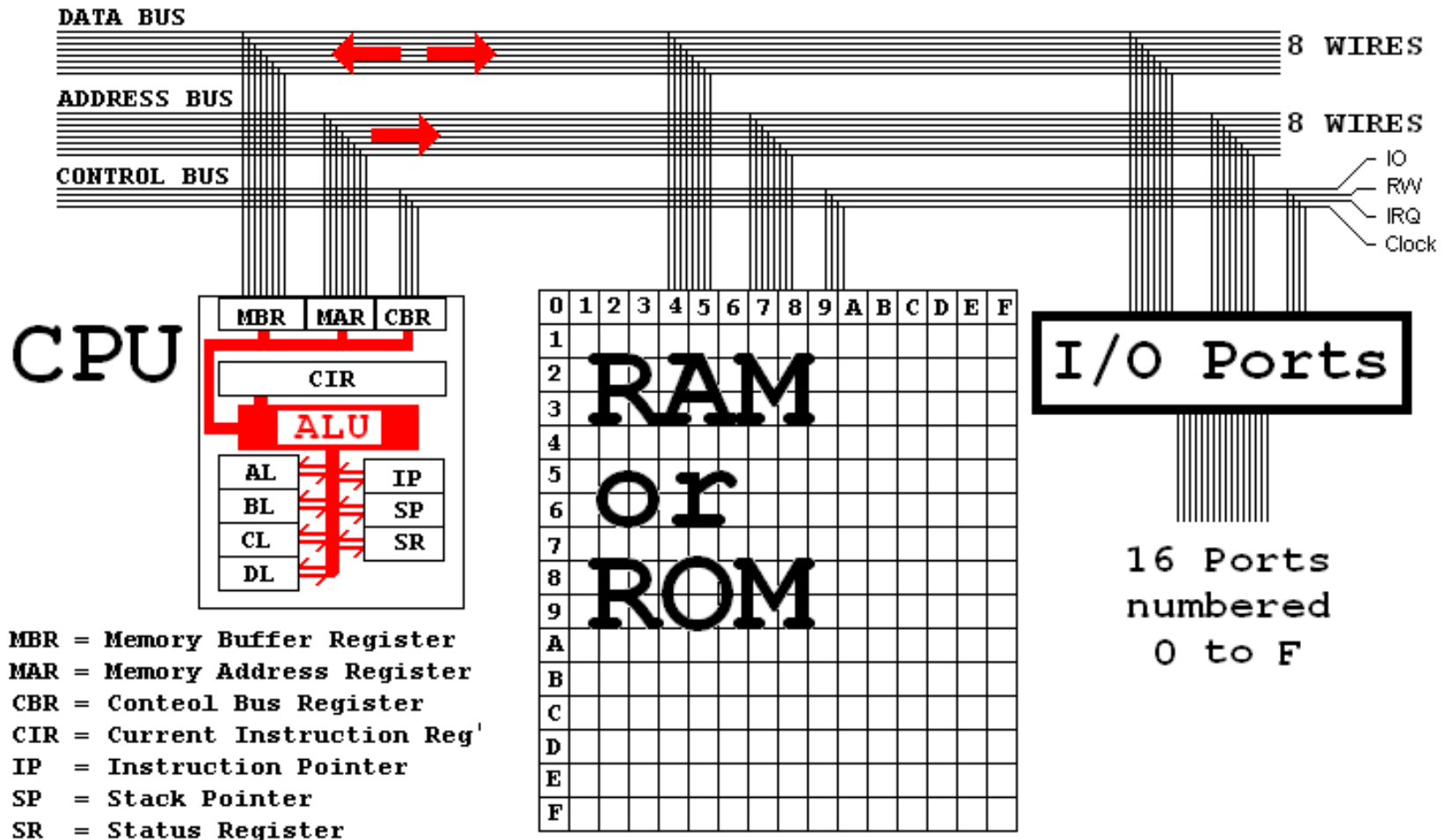
Microprocessor Simulator for Students

<http://www.softwareforeducation.com/>

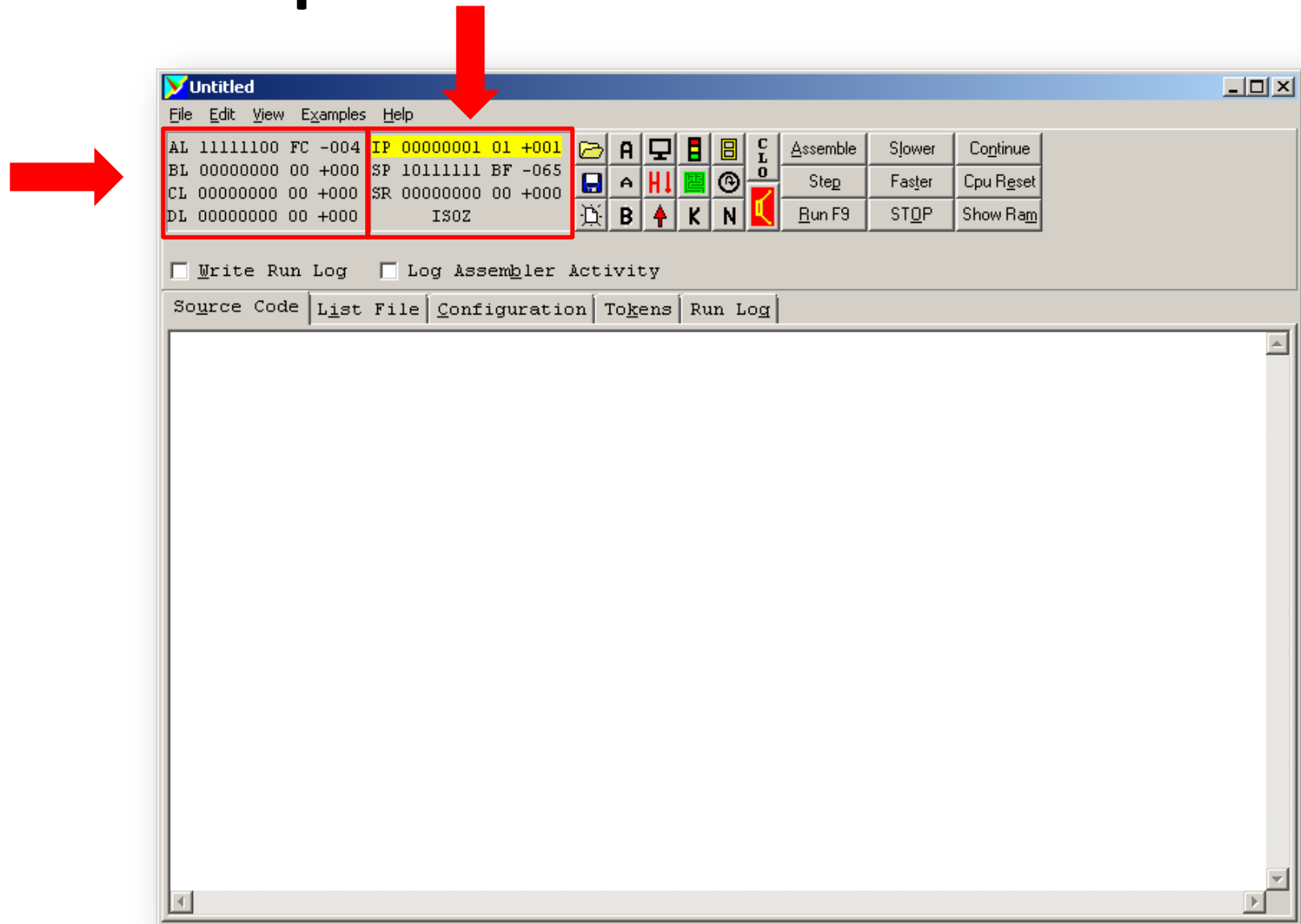
Microprocessor Simulator for Students

- Este simulador emula uma CPU de oito bits (que é semelhante aos da família de chips 80x86), e 256 bytes de RAM são simulados.
- O simulador inclui um pequeno subconjunto do conjunto completo de instruções normalmente encontrado com este estilo de processador.
- Inclui instruções avançadas como CALL, RET, INT e IRET.
- O simulador é licenciado sob GNU / GPL tornando-o gratuitamente disponível para uso por estudantes e instituições educacionais a custo zero.
- Recursos
 - CPU de 8 bits
 - 16 portas de entrada e saída. Nem todos são usados.
 - Periféricos simulados nas portas 0 a 5.

Arquitetura do SMS



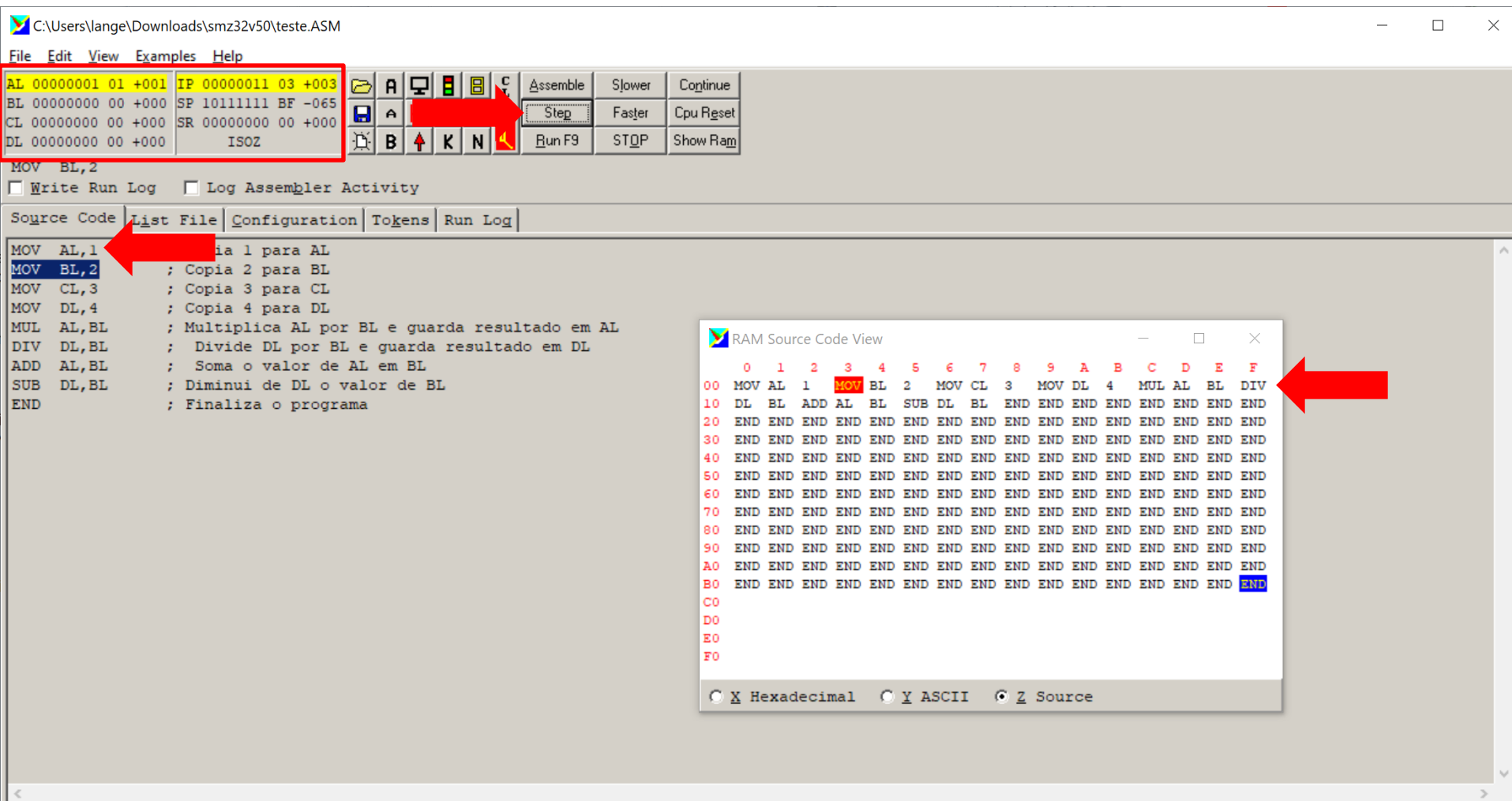
Microprocessor Simulator for Students



01 - Exemplo de Programação no SMS (Assembler)

```
MOV AL,1      ; Copia 1 para AL
MOV BL,2      ; Copia 2 para BL
MOV CL,3      ; Copia 3 para CL
MOV DL,4      ; Copia 4 para DL
MUL AL,BL     ; Multiplica AL por BL e guarda resultado em AL
DIV DL,BL     ; Divide DL por BL e guarda resultado em DL
ADD AL,BL     ; Soma o valor de AL em BL
SUB DL,BL     ; Diminui de DL o valor de BL
END           ; Finaliza o programa
```

01 - Exemplo de Programação no SMS (Assembler)



01 - Exemplo de Programação no SMS (Assembler)

C:\Users\lange\Downloads\smz32v50\teste.ASM

File Edit View Examples Help

AL 00000100 04 +004 IP 00010101 15 +021
BL 00000010 02 +002 SP 10111111 BF -065
CL 00000011 03 +003 SR 00000000 00 +000
DL 00000010 02 +002 ISOZ

Assembly Slower Continue
Step Faster Cpu Reset
Run F9 STOP Show Ram

☐ Write Run Log ☐ Log Assembler Activity

Source Code List File Configuration Tokens Run Log

```
MOV AL,1 ; Copia 1 para AL
MOV BL,2 ; Copia 2 para BL
MOV CL,3 ; Copia 3 para CL
MOV DL,4 ; Copia 4 para DL
MUL AL,BL ; Multiplica AL por BL e guarda resultado em AL
DIV DL,BL ; Divide DL por BL e guarda resultado em DL
ADD AL,BL ; Soma o valor de AL em BL
SUB DL,BL ; Subtrai de DL o valor de BL
END ; finaliza o programa
```

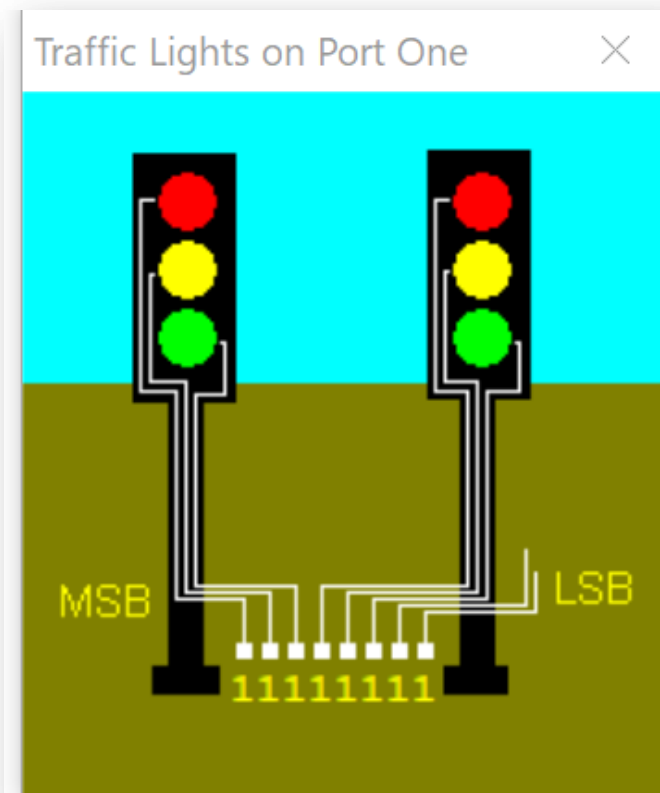
RAM Source Code View

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	MOV	AL	1	MOV	BL	2	MOV	CL	3	MOV	DL	4	MUL	AL	BL	DIV
10	DL	BL	ADD	AL	BL	SUB	DL	BL	END	END	END	END	END	END	END	END
20	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END
30	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END
40	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END
50	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END
60	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END
70	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END
80	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END
90	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END
A0	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END
B0	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END
C0																
D0																
E0																
F0																

☐ X Hexadecimal ☐ Y ASCII ☒ Z Source

02 - Exemplo de Programação no SMS (Semáforo)

- Controlador de Semáforo
 - **Porta 01:** Os semáforos estão conectados à porta 01.
 - Um valor binário faz com que uma lâmpada acenda ou que desligue.



02 - Exemplo de Programação no SMS (Assembler)

Start:

MOV AL, 55

OUT 01

MOV AL, FF

OUT 01

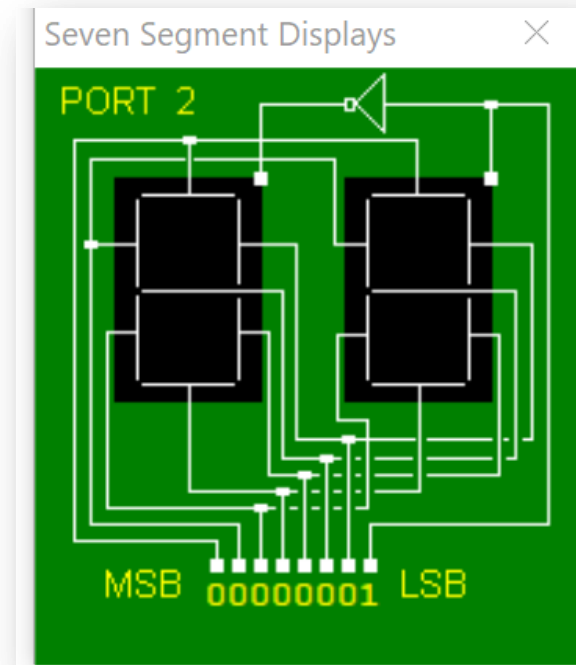
JMP Start

END

03 - Exemplo de Programação no SMS (Display)

Na porta 02 está conectado um display de sete segmentos.

O bit mais à direita controla qual dos dois grupos de segmentos está ativo. Se o bit menos significativo (LSB) for zero, o segmento esquerdo estará usado. Se o bit menos significativo (LSB) for um, o segmento da direita será usado.



03 - Exemplo de Programação no SMS (Assembler)

Start:

MOV AL,FA; 1111 1010

OUT 02;

MOV AL,0; 0000 0000

OUT 02;

MOV AL,FB; 1111 1011

OUT 02;

MOV AL,1; 0000 0001

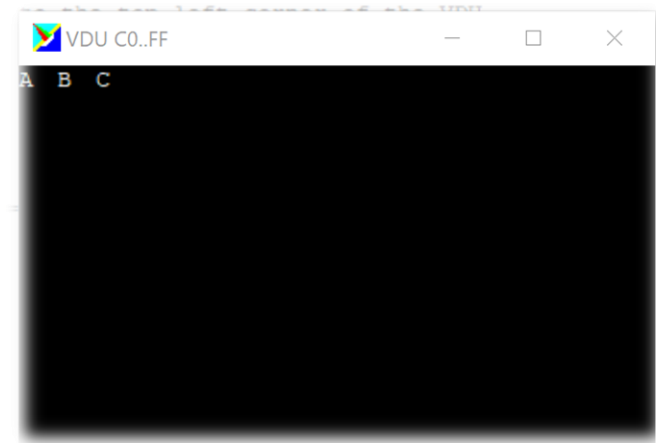
OUT 02;

JMP Start

END

04 - Exemplo de Programação no SMS (Saída Tela)

- A **Unidade Visual de Display (VDU)** é mapeada na memória. Isso significa que os locais da RAM correspondem às posições na tela.
- A localização da RAM C0 é mapeada no canto superior esquerdo do VDU. A tela tem 16 colunas e quatro linhas mapeadas para os locais de RAM C0 a FF. Quando você escreve códigos ASCII nesses locais de RAM, os caracteres de texto correspondentes aparecem e o VDU fica visível.



04 - Exemplo de Programação no SMS (Assembler)

```
MOV      AL,41  ; ASCII 'A'
MOV      [C0],AL;
MOV      AL,42  ; ASCII 'B'
MOV      [C1],AL;
MOV      AL,43  ; ASCII 'C'
MOV      [C2],AL;
END
```

05 - Exemplo de Programação no SMS (Entrada)

Entrada da porta zero. Neste simulador, a porta zero é conectada ao hardware do teclado. O simulador espera pelo pressionamento de uma tecla e copia o código ASCII da tecla pressionada no registrador AL.

Keyboard Input

Waiting for keyboard input >

05 - Exemplo de Programação no SMS (Assembler)

CLO ; Fecha as janelas – próprio do SMS

Rep:

IN 00 ; Entrada de Dados

CMP AL,0D ; Compara a entrada com ASCII 0D

JNZ Rep ; Se AL = 0D, finaliza, senão volta

END

06 - Exemplo de Programação no SMS (Terclado)

CLO ; Fecha as janelas – próprio do SMS

Rep:

IN 00 ; Entrada de Dados

CMP AL,0D ; Compara a entrada com ASCII 0D

JNZ Rep ; Se AL = 0D, finaliza, senão volta

END

06 - Exemplo de Programação no SMS (Terclado)

- **Este é um dos dispositivos mais complexos.** Para tornar o teclado visível, use **OUT 07**.
- Cada vez que uma tecla é pressionada, **uma interrupção de hardware, INT 03 é gerada.**
- **Por padrão, a CPU irá ignorar esta interrupção.** Para processar a interrupção, no início do programa, use o comando STI para definir o sinalizador de interrupção (I) no registrador de status da CPU (SR).
- Coloque um vetor de interrupção no endereço **RAM 03**. **Isso deve apontar para o código do manipulador de interrupção.**
- O manipulador de interrupção deve usar **IN 07** para ler a tecla pressionada no registrador **AL**.

-- STI: Defina o sinalizador de interrupção no Registro de Status.

-- CLI: Limpa o flag de interrupção no Registro de Status.

-- ORG: Diretiva Assembler: Gera código a partir do endereço.

--DB: Definir byte (Constante)

-- IRET: Restaura o IP da pilha e salta para ele.

06 - Exemplo de Programação no SMS (Terclado)

- Depois que o STI definir o sinalizador (I) no registrador de status (SR), as interrupções do temporizador de hardware também serão geradas.
- Estes também devem ser processados. O timer de hardware gera **INT 02**.
- Para processar essa interrupção, coloque um vetor de interrupção na localização 02 da RAM.
- Isso deve apontar para o código do manipulador de interrupção do timer.
- O código do timer pode ser tão simples quanto **IRET**. Isso causará um retorno de interrupção sem fazer nenhum outro processamento.

-- STI: Defina o sinalizador de interrupção no Registro de Status.

-- CLI: Limpa o flag de interrupção no Registro de Status.

-- ORG: Diretiva Assembler: Gera código a partir do endereço.

--DB: Definir byte (Constante)

-- IRET: Restaura o IP da pilha e salta para ele.

06 - Exemplo de Programação no SMS (Terclado)

C:\Users\lange\Downloads\smz32v50\teste.ASM

File Edit View Examples Help

AL 00000000 00 +000	IP 00110000 30 +048	Folder	A	Monitor	Stop	Clipboard	CLO	Assemble	Slower	Continue
BL 00000000 00 +000	SP 10111111 BF -065	Save	Undo	Redo	Print	Refresh	Run	Step	Faster	Cpu Reset
CL 00000000 00 +000	SR 00010000 10 +016	Close	B	Up	K	N	Run	Run F9	STOP	Show Ram
DL 00000000 00 +000	ISOZ									

jmp idle
☐ Write Run Log ☐ Log Assembler Activity

Source Code | List File | Configuration | Tokens | Run Log


```
jmp start
db 10 ; Vektor de Interrupcao de Hardware
db 20 ; Vektor de Interupcao do Teclado

org 10
nop ; Aguardando instruções
iret
org 20
CLI ; Entrada do Teclado
push al ; Carregando no registrador
pushf
in 07 ; lendo o teclado
nop ; Process the key press here
popf
pop al
STI
iret

start:
STI ; Configurando a FLAG Input
out 07 ; Mostrando o Teclado

idle:
nop ; Aguardando instruções
jmp idle
end
```

Key press generates INT 03 - Read key from port 07



06 - Exemplo de Programação no SMS (Terclado)

```
jmp      start
db       10
db       20
org      10
nop
iret
org      20
CLI
push     al
pushf
in       07
nop
popf
pop      al
STI
iret
start:
STI
out      07
idle:
nop
jmp      idle
end
```

Atividades

- 1) Fazer uma codificação que leia o Teclado e posicione o Elevador no andar indicado pelo teclado.
- 2) Fazer uma codificação que leia o teclado e escreva na tela sem sobreescrever o caractere anterior.