

Exemplos de programas para Intel usando TASM

Passos

- Escrever o programa fonte (xx.ASM)
 - Usar o seu editor de texto preferido (TXT)
- Utilizar o montador (TASM)
 - Geração de código objeto (xx.OBJ)
- Utilizar o carregador (TLINK)
 - Geração de código executável (xx.EXE)
- Utilizar o depurador (CodeView, Debug, TD)

Exemplo: teste

Programa chama uma subrotina (FRASE) para escrever mensagem na tela

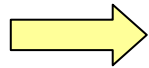
```
PILHA    SEGMENT    STACK
          DB 32 DUP ('STACK---')
PILHA    ENDS
```

parâmetro
opcional

```
DADOS    SEGMENT
MENSAGEM DB 'Hello World !',0DH,0AH
TAMANHO  EQU $-MENSAGEM
CONTADOR DB ?
DADOS    ENDS
```

CODIGO SEGMENT

ASSUME CS:CODIGO,SS:PILHA,DS:DADOS



START: MOV AX,DADOS ; Inicializa segmento de dados

MOV DS,AX

MOV CONTADOR,10

DE_NOVO: CALL FRASE

DEC CONTADOR

JNZ DE_NOVO

MOV AH, 4CH ; Retorna ao DOS

INT 21H

FRASE PROC NEAR

MOV BX,0001H

LEA DX, MENSAGEM

MOV CX, TAMANHO

MOV AH,40H

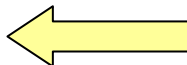
INT 21H ; Escreve mensagem

RET

FRASE ENDP

CODIGO ENDS

END START



Execução de programas

- Escrever o programa fonte (xxx.ASM)
- Chavear para modo comando DOS (cmd)
 - Iniciar -> Executar -> cmd
 - Iniciar -> Programas -> Acessórios -> Prompt de comando
- Achar o diretório correto (cd)
- Chamar o montador (TASM)
 - tasm teste
 - Geração de código objeto (xxx.OBJ)
- Utilizar o carregador (TLINK)
 - tlink teste
 - Geração de código executável (xxx.EXE)
- Executar programa

Erros de montagem e linkagem

- Em caso de erro do montador
 - Montador somente indica número da linha
 - Correção deve ser feita em um editor
 - Para facilitar localização dos erros:
`tasm /l <nome>`
 - gera <nome>.lst (listagem completa da montagem)
 - não corrigir <nome>.lst !!
- Em caso de erro do linker
 - Verifique se montou sem erros
 - Verifique se declarou segmento de pilha

Erros de execução

- Em caso de erro de execução
 - Escrever mensagens de depuração é complicado....
 - Revisar a lógica no papel é ineficiente...
 - Utilizar o depurador !!
 - CodeView, Debug, TD
 - td <nome do programa>
 - É bom ter a mão a listagem da montagem

Exemplo: leitura de arquivo

- leitura de arquivo do disco usando DOS
- programa:
 - pede nome do arquivo com mensagem na tela
 - lê nome do arquivo do teclado
 - abre arquivo para leitura
 - lê arquivo caracter a caracter escrevendo na tela
 - quando encontra LF espera a entrada de um caracter no teclado
 - quando encontra fim de arquivo termina o programa

usa rotinas do sistema DOS através da INT 21H

assume cs:codigo,ds:dados,**es:dados**,ss:pilha

CR EQU 0DH ; caractere ASCII "Carriage Return"
LF EQU 0AH ; caractere ASCII "Line Feed"

; SEGMENTO DE DADOS DO PROGRAMA

dados segment
nome db 64 dup (?)
buffer db 128 dup (?)
pede_nome db 'Nome do arquivo: ','\$'
erro db 'Erro! Repita.',CR,LF,'\$'
msg_final db 'Fim do programa.',CR,LF,'\$'
handler dw ?
dados ends

declaração do seg dados com os
todos dados necessários


\$ - para rotinas de saída em vídeo

handler vai conter o id do arquivo
(file handler)

; SEGMENTO DE PILHA DO PROGRAMA

pilha segment stack ; permite inicializacao automatica de SS:SP
 dw 128 dup(?)
pilha ends

; SEGMENTO DE CÓDIGO DO PROGRAMA

codigo segment
 inicio: ; CS e IP sao inicializados com este endereco
 mov ax,dados ; inicializa DS
 mov ds,ax ; com endereco do segmento DADOS
 mov es,ax ; idem em ES
; fim da carga inicial dos registradores de segmento
;

AH = 9 : Saída de string no vídeo
apontado por DS:DX,
terminado por '\$'

```
; pede nome do arquivo
de_novo: lea  dx,pede_nome ; endereco da mensagem em DX
          mov  ah,9         ; funcao exibir mensagem no AH
          int  21h         ; chamada do DOS
```

```
;
;
;
;
```

```
; le nome do arquivo
```

```
          lea  di, nome
entrada:  mov  ah,1
          int  21h         ; le um caracter com eco
          cmp  al,CR       ; compara com carriage return
          je   continua
          mov  [di],al     ; coloca no area de memoria reservada a nome
          inc  di
          jmp  entrada
```

```
;
;
;
```

```
continua: mov  byte ptr [di],0 ; forma string ASCIIZ com o nome do arquivo
          mov  dl,LF         ; escreve LF na tela
          mov  ah,2
          int  21h
```

```
;
```

AH = 1 : Entrada de caracter com eco
código ASCII retorna em AL e
código de varredura em AH

ASCIIZ: nome e caminho completo,
terminados por um byte em zero (0H)

AH = 2 : Saída de caracter
DL: caracter

AH = 3DH : Abre arquivo existente
AL = modo (0: **leitura**, 1:escrita, 2:leitura e escrita)
DS:DX = ponteiro para nome do arquivo, em **ASCIIZ**
Retorno: Se CF=0, **AX: file handler**
Se CF=1, AX: código de erro

```
;
; abre arquivo para leitura
    mov     ah,3dh
    mov     al,0
    lea     dx,nome
    int     21h

;
    jnc     abriu_ok
    lea     dx,erro           ; endereco da mensagem em DX
    mov     ah,9             ; funcao exibir mensagem no AH
    int     21h              ; chamada do DOS
    jmp     de_novo           ; volta a pedir nome do arquivo

;
```

AH = 9 : Saída de string no vídeo
apontado por **DS:DX**,
terminado por '\$'

AH = 3FH : Lê de arquivo
BX = file handler
CX = número de bytes a ler
DS:DX = ponteiro para área de buffer
Retorno: Se CF=0, AX contém número de bytes lidos
Se CF=1, AX: código de erro

```
abriu_ok: mov handler,ax
laco:     mov ah,3fh      ; le um caracter do arquivo
          mov bx,handler
          mov cx,1       ; cs indica numero de bytes a serem lidos
          lea dx,buffer
          int 21h

;

          cmp ax,cx      ; ax contem numero de bytes lidos (se não for 1 = erro)
          jne fim
          mov dl, buffer ; escreve caracter na tela
          mov ah,2
          int 21h

;

          mov dl, buffer ; escreve na tela até encontrar um LF (fim de linha)
          cmp dl, LF
          jne laco

;

          mov ah,8       ; espera pela digitacao de uma tecla qualquer
          int 21h
          jmp laco

;
```

AH = 2 : Saída de caracter
DL: caracter

AH = 8 : Entrada de caracter sem eco
código ASCII em AL
código de varredura em AH

AH = 3EH : Fecha arquivo (previamente aberto)
BX = file handler

```
fim:    mov    ah,3eh        ; fecha arquivo
        mov    bx,handler
        int    21h

;
;
;
;
        lea    dx,msg_final ; endereco da mensagem em DX
        mov    ah,9         ; funcao exibir mensagem no AH
        int    21h         ; chamada do DOS

;
;
;
;
        mov    ax,4c00h     ; funcao retornar ao DOS no AH
                                ; codigo de retorno 0 no AL
                                ; chamada do DOS
        int    21h
        ends
        end    inicio
```

codigo →

AH = 9 : Saída de string no vídeo
apontado por DS:DX,
terminado por '\$'

AH = 4CH : encerrar com código de retorno em AL
AL = 0 indica retorno normal;
AL <> 0 indica código (ERRORLEVEL))

Execução do programa leitura

- Escrever o programa fonte (leitura.ASM)
- Chavear para modo comando DOS (cmd)
 - Iniciar -> Executar -> cmd
 - Iniciar -> Programas -> Acessórios -> Prompt de comando
- Achar o diretório correto (cd)
- Chamar o montador (TASM)
 - tasm leitura
 - Geração de código objeto (leitura.OBJ)
- Utilizar o carregador (TLINK)
 - tlink leitura
 - Geração de código executável (leitura.EXE)
- Executar programa