Programação de Software Básico – Linguagem de montagem

MATA49

Prof. Babacar Mane

2021.1 – Aula 3

Programação de Software Básico
Linguagem de Montagem

- **Conteúdo**

- **Montadores**
- **Geração do executável**
- **Desmontadores (disassemblers)**
- **NASM: Estrutura de um programa**
- **Linguagem de montagem**

Programação de Software Básico
Linguagem de Montagem

- **Montadores**

NASM – Netwide Assembler

- Bastante usado (confiável para o desenvolvimento de aplicações de grande porte, de uso comercial e industrial)
- http://nasm.sourceforge.net/). É grátis, multi-plataforma e o código fonte está disponível gratuitamente.
- Plataformas: Windows, Linux, Mac OS X, DOS, OS/2
- Instalação: pacote nasm do Linux

$ sudo apt-get install nasm

Programação de Software Básico
Linguagem de Montagem

Geração de executável

Passo 1 – Geração do código objeto

- Usando NASM, em um computador de 32 bits:

```
$ nasm -f elf32 hello.asm
```

- Usando NASM, em um computador de 64 bits:

```
$ nasm -f elf64 hello.asm
```

Os comandos acima gerarão um arquivo hello.o.

Programação de Software Básico
Linguagem de Montagem

- **Geração de executável**

Passo 2 – Ligação (geração do código de máquina)

```
$ ld -s -o hello hello.o
```

O comando acima gerará o arquivo executável hello

Programação de Software Básico
Linguagem de Montagem

❑ Desmontadores (disassemblers)

- Um desmontador é um programa que recebe como entrada um executável ou código objeto qualquer e gera como saída um programa em linguagem de montagem

O desmontador associado ao nasm é o ndisasm

Exemplo de uso:

$ ndisasm hello.o

Programação de Software Básico
Linguagem de Montagem

- **NASM: Estrutura de um programa**
  - **Primeiro exemplo (muda o tamanho do cursor)**

```
.MODEL SMALL ;modelo de memória
.STACK       ;espaço de memória para instruções do programa na pilha
.CODE        ;as linhas seguintes são instruções do programa
    mov ah,01h  ;move o valor 01h para o registrador ah
    mov cx,07h  ;move o valor 07h para o registrador cx
    int 10h     ;interrupção 10h
    mov ah,4ch  ;move o valor 4ch para o registrador ah
    int 21h     ;interrupção 21h
.DATA
    x db 1
END ;finaliza o código do programa
```

Programação de Software Básico
Linguagem de Montagem

- **Referencias:**

- Tabela de chamadas ao sistema no Linux

  https://www.ime.usp.br/~kon/MAC211/syscalls.html

- The Netwide Assembler – NASM

  http://www.nasm.us/

- Livro The Art of Assembly Language Programming, de R. Hyde
  http://cs.smith.edu/~thiebaut/ArtOfAssembly/artofasm.html

Programação de Software Básico
Linguagem de Montagem

- **NASM: declaração de variáveis:**
- Variável é um nome simbólico para um dado atualizável pelo programa
- Cada variável possui um tipo e recebe um endereço de memória
- Usa-se pseudo-instruções para definir o tipo da variável
- O montador atribui o endereço de memória
- Nomes de variáveis, constantes e rótulos devem:
  - conter somente letras, números ou os caracteres

  '_', '$', '#', '@', ' ', '.' e ' ?'
  - iniciar por letra, '_', ' ?' ou '.' (sendo que o uso do ponto denota um rótulo local 1)
- NASM é sensível a letras maiúsculas e minúsculas nos nomes dos identificadores

Programação de Software Básico
Linguagem de Montagem

- **NASM: declaração de variáveis inicializadas:**

A declaração de variáveis inicializadas é feita na seção .data .

“Tipos” de variáveis inicializadas:

Pseudo-Instrução	Entende-se por
DB	define byte (1 byte)
DW	define word (2 bytes consecutivos)
DD	define doubleword (4 bytes consecutivos)
DQ	define quadword (8 bytes consecutivos)
DT	define ten bytes (10 bytes consecutivos)

Programação de Software Básico
Linguagem de Montagem

- **NASM: declaração de variáveis inicializadas:**

Exemplos

Nome	Tipo	Valor inicial
NUM1:	DB 64	
NUM2:	DB 0150h	;ilegal, valor > que 1 byte
NUM3:	DW 0150h	
NUM5:	DW 1234h	;byte 34h, endereço NUM5 j
		; byte 12h endereço NUM5+1
VET:	DB 10h,20h,30h	;vetor de 3 bytes
LETRAS:	DB “abC”	;vetor de caracteres ASCII
MSG:	DB 'Ola!',0Ah,0Dh	;mistura letras e números

Programação de Software Básico
Linguagem de Montagem

- **NASM: declaração de variáveis não inicializadas:**

A declaração de variáveis não inicializadas é feita na seção .bss

"Tipos" de variáveis não inicializadas:

Pseudo-Instrução	Entende-se por
RESB	reserve byte (1 bits)
RESW	reserve word (2 bytes consecutivos)
RESD	reserve doubleword (4 bytes consecutivos)
RESQ	reserve quadword (8 bytes consecutivos)
REST	reserve ten bytes (10 bytes consecutivos)

Exemplos

Nome	Tipo	Quantidade
BUFFER:	RESB 64	;reserva 64 X 1 bytes
NUM:	RESW 1	;reserva 1 X 2 bytes
VET:	RESQ 10	;reserva 10 X 8 bytes

Programação de Software Básico
Linguagem de Montagem

- **Acessando a memória por meio de variáveis:**

```
section .data
                        VAR1: DW 0F17H

section .text


MOV EAX,VAR1     ;copia em EAX o endereço de memória
                 ;associado a VAR1
MOV EAX,[VAR1]   ;copia em EAX o valor de VAR1, ou seja,
                 ;0F17H
```

Programação de Software Básico
Linguagem de Montagem

- **Declaração de constantes**

Constante é um nome simbólico para um valor constante usado com frequência no programa.

É definido por meio da pseudo-instrução EQU.

Exemplos

```
Nome        EQU         Valor
LF:         EQU         0Ah     ;LF = caracter Line Feed
CR:         EQU         0Dh     ;CR = caracter Carriage Return
LINHA:      EQU 'Digite seu nome'
MSG:        DB          LINHA,LF,CR
```

Programação de Software Básico
Linguagem de Montagem

- **Algumas diferenças entre as sintaxes da AT&T e Intel**

	AT&T	Intel
Ordem dos operandos	MOV *origem*, *dest*	MOV *dest*, *origem*
Declaração de variáveis	var1: .int *valor*	var2: DB *valor*
Declaração de constantes	const1 = *valor*	const2: EQU *valor*
Uso dos registradores	MOV %eax,%ebx	MOV ebx,eax
Uso das variáveis	MOV $var1,%eax	MOV eax,var2
Uso das constantes	MOV $const1,%eax	MOV eax,const2
Uso de imediatos	MOV $57,%eax	MOV eax,57
Num. hexadecimais	MOV $0xFF,%eax	MOV eax,0FFh
Tam. das operações	MOVB [ebx],%al	MOV al, byte [ebx]
Delimitador de comentários	# comentário AT&T	; comentário Intel

Programação de Software Básico
Linguagem de Montagem

- **Algumas operações úteis - Abertura de arquivos**

```
; declaração de constantes e variáveis
RDONLY:  equ 0                ;modo de abertura -- somente leitura
WRONLY: equ 1                 ;modo de abertura -- somente escrita
RDWR:    equ 2                ;modo de abertura -- leitura e escrita
arquivo:   db "MAC211.txt"    ;nome do arquivo a ser lido
buffer:    resb 256           ;um buffer com 256 bytes
```

Programação de Software Básico
Linguagem de Montagem

- **Algumas operações úteis - Abertura de arquivos**

```
; abertura do arquivo
;int open (const char *pathname, int flags, mode_t mode);

mov  ebx,arquivo          ;1o parâmetro: caminho + nome do arquivo
mov  ecx,RDONLY           ;2o parâmetro: modo de leitura
mov  edx,0                ;3o parâmetro: permissões de acesso,
                          ; só e' relevante na criação de arquivos
mov  eax,5                ;numero da chamada ao sistema (open)
int 80h                   ; chamada ao núcleo do SO

; apos a execução da interrupção, em caso de sucesso, o descritor do arquivo estará em ; EAX
```

Programação de Software Básico
Linguagem de Montagem

- **Algumas operações úteis - Leitura de arquivos**

```
; [continuação do programa anterior]
; leitura do arquivo
; int read(int fd, void *buf, size_t count);
mov  ebx,eax          ;1o parâmetro: descritor do arquivo
mov  ecx,buffer       ;2o parâmetro: ponteiro para o buffer
mov  edx,256          ;3o argumento: quantidade de bytes a ser lida
mov  eax,3            ;numero da chamada ao sistema (read)
int 80h               ;chamada ao núcleo do SO
;apos a execução da interrupção, a quantidade de bytes lida do arquivo estará' em EAX
```

Programação de Software Básico
Linguagem de Montagem

- **Algumas operações úteis - Escrita em arquivos**

```
;[continuação do programa anterior]
;int write(int fd, const void *buf, size_t count);
mov edx,eax                ;3o parâmetro: tamanho da mensagem
mov ebx,1                  ;1o parâmetro: stdout
mov ecx,buffer             ;2o parâmetro: ponteiro para a msg
mov eax,4                  ;numero da chamada ao sistema (write)
int 80h                    ;chamada ao núcleo do SO
;apos a execução da interrupção, a quantidade de bytes escrita estará em EAX
```

Programação de Software Básico
Linguagem de Montagem

- **Exercício**

Usando a sintaxe Intel, faça um programa em linguagem de montagem que leia um texto da entrada padrão, passe-o para letras maiúsculas e mostre o resultado na saída padrão. Caracteres que não são letras minúsculas devem permanecer inalterados.

Programação de Software Básico
Linguagem de Montagem

- **Exercício**

1. Faça um programa em linguagem de montagem que leia um texto da entrada padrão, inverta-o e mostre o resultado na saída padrão.

2. Faça um programa em linguagem de montagem que leia um arquivo texto e conte o número de caracteres e o número de palavras presentes no arquivo. Considere que o separador de palavras é o caractere de espaço (' '). Obs.: para imprimir os resultados das contagens na saída padrão, você precisará converter um número em *string*.