Assembly x86

Hugo Bessa - hrba

Paulo Serra Filho – ptvsf

Roteiro

- Assembly
- Assemblers
- Sections
- Registradores
- Registradores de Segmentos
- Principais Operações do NASM
- Funcões e Macros
- Interrupções
- Compilando um programa

Assembly

- Linguagem de baixo nível
- Traduzida por assemblers
- Baseada em mnemônicos
- Assembly x86 diferente do MIPS
  - Menos registradores
  - Muitas operações podem ser feitas com dados na momória
  - Maior abstração

Assemblers

- TurboASM - TASM
- MicrosoftASM – MASM
- NetwideASM - NASM
- Um assembler não compila o código de outro

Assemblers

- NASM: por que usar?
  - Sintaxe menos poluída
  - Mais abstrações
  - Opensource

Hello World em NASM x86

```
section .data
     hello: db 'Hello world!', 10
     hellolen: equ $$-$

section .text
     global _start


_start:
     CALL PRINTHELLO          ; chama função PRINTHELLO
     CALL FIM            ; chama função FIM

PRINTHELLO:
     MOV EAX, 4          ; read syscall
     MOV EBX, 0          ; fd = 0 (stdout)
     MOV ECX, hello      ; string
     MOV EDX, hellolen        ; string length
     INT 80H             ; syscall interruption
     RET            ; back to the line after the call

FIM:
     MOV EAX, 1          ; exit syscall
     MOV EBX, 0          ; program return
     INT 80H             ; syscall interruption
```

Sections

- .data
  - Dados inicializados
    - `db` ; Declara bytes
    - `equ` ; Resolve uma expressão e inicializa a variável com o resultado
    - `dw` ; Declara uma palavra que armazena um dado
  - Ex:

```
section .data
        message: db 'Hello world!', 10
        msglen: que 12
        buffersize: dw 1024
```

Sections

- .bss
  - Espaço reservado (variáveis)
    - `resb` ; Reserva uma quantidade de bytes
    - `resw` ; Reserva uma quantidade de palavras(2bytes)
    - `resq` ; Reserva um array de numeros reais
  - Ex:

```
section .bss
    name: resb 255
    bigNum: resw 1
    realarray: resq 10
```

Sections

- .text
  - Onde o código assembly fica
  - Ex:

```
section .text
     global _start
_start:
     POP EBX
     .
     .
     .
```

Registradores

- 32 bits
  - EAX
  - EBX
  - ECX
  - EDX
  - ESI
  - EDI
  - ESP
  - EBP
- 16 bits
  - AX
  - BX
  - CX
  - DX
  - BP
  - SI
  - DI

Registradores de segmento

- Não foram estendidos para 32 bits
- Informações sobre o código, dados, pilha
- Requerem muito cuidado quando alterados

Registradores de segmento

- Segment:offset
  - Espaço de endereçamento de 1MiB
  - (segment << 4) + offset
  - Vários endereços para cada segmento
  - Endereços repetidos e overflow
- O comando `ORG`

Principais Operações do NASM

- Load/Store: `MOV`
- Lógicas: `XOR, AND, OR`
- Aritméticos: `ADD, SUB, INC, DEC`
- Comparativas: `CMP, TEST`
- Saltos: `JE, JNE, JZ, JNZ, JMP`
- Pilha: `PUSH, POP`
- Interrupção: `INT`

Funcões e Macros

- Funções
  - CALL e RET
  - O código não se altera com a execução
- Macros
  - Sinal de %
  - Possui parâmetros
  - Altera o código assembly da mesma forma que o define em C

Funcões e Macros

- Macros
  - EX:

```
%macro prologue 1
   push ebp
   mov ebp,esp
   sub esp,%1
%endmacro
```

Interrupções

- Sinal que tipicamente resulta numa **troca de contexto**.
- Suspende temporariamente o que o processador está fazendo no momento de sua ocorrência
- Podem ser de hardware ou software, externas ou internas.

Interrupções no Linux

- `Int 80h` (0x80) = syscall
- Precisa de parâmetros
- EAX = tipo da syscall

Compilando um programa

- Baixe o pacote nasm
- Escreva o código e salve com nome.asm
- No terminal:
  - Vá para a pasta onde se encontra o código
  - nasm -f elf nome.asm
  - ld -s -o nome nome.o

Interrupções da BIOS

- Registradores de 16 bits
- Vários tipos (int 10h, int 13h,int 16h...)
- AH = valor secundário da interrupção
- Outros registradores recebem os parâmetros

Compilando um programa

- Baixe o qemu e o nasm
- Escreva o código e salve como nome.asm
- No terminal:
  - Vá para a pasta onde se encontra o código
  - $ nasm nome.asm -f bin -o nome.bin
  - $ qemu nome.bin

Referências

- Helpcc - Interrupções da bios e comandos
- http://stanislavs.org/helppc/
- Scan Codes
- http://stanislavs.org/helppc/scan_codes.html
- Tabela de Syscalls
- http://bluemaster.iu.hio.no/edu/dark/lin-asm/syscalls.html
- TECH Help - Interrupções da bios
- http://webpages.charter.net/danrollins/techhelp/0002.HTM