MATA49 - Programação de Software Básico

Apresentação da disciplina

Leandro Andrade
leandrojsa@ufba.br

Planejamento

- Avaliação
  - Uma prova
  - Um trabalho prático
  - Exercícios semanais/quinzenais
  - Seminário prático
- Média Final:

  (Pr*0,3 + Tr*0,3 + Ex*0,2 + Se*0,2)

Planejamento

- Cronograma de aulas e slides
  - Moodle - MATA49 - Programação de Software Básico
- Contatos através do e-mail:
  - leandrojsa@ufba.br
    - Busquem utilizar no assunto o prefixo: [MATA49]
      - Ensino outras disciplinas, assim facilita a identificação

Metodologia

- Aulas expositivas
  - Quartas (PAF)
- Exercícios práticos
- Aulas em laboratório
  - Segunda (laboratório 143 IME)

Programação se aprende programando!

Conteúdo Abordado

- Arquitetura Intel
- Representação dos dados em llinguagem de máquina
- Introdução a Linguagem de montagem
- Condicionais e estrutura de repetição
- Operações com Bits
- Vetores
- Instruções para manipulação de Vetores e Strings

Conteúdo Abordado

- Pilhas
- Procedimentos
- Interrupções
- Instruções sobre ponto flutuante
- Assembly com C
- Assembly 64 bits Intel
- Introdução linguagem de montagem ARM

Referências Bibliográficas

- **Paul A. Carter, PC Assembly Language**
- **Kip Irvine, Assembly Language for x86 Processors, 6th edition**
- Andrew S. Tanenbaum, Organização Estruturada de Computadores, 4ª edição, Prentice-Hall do Brasil, 2001
- Manual oficial do Nasm
- Hugo Perez Perez. Tutorial de linguagem Assembly. Information Systems General Coordination. University of Guadalajara. Traduzido para o português por Jeferson Amaral. 1995
- IA-32 Intel Architecture Software Developer's Manual volume1: Basic architecture. Intel Corporation. 2003

Objetivo

- Programação em linguagem de montagem Intel;
- Recursos de arquitetura para técnicas de programação;
  - Arquitetura Intel
- Conceitos de implementação de baixo nível;

Introdução

O que é uma linguagem de montagem?

Qual a diferença entre assembly e assembler?

Antes, alguns conceitos...

- Tradutores:
  - Programas que convertem um programa escrito em uma linguagem para outra
  - Linguagem fonte → Linguagem Alvo
  - Geração do programa objeto
- Interpretação:
  - O programa fonte é executado por um interpretador.
  - Em certos caso geração de um código intermediário (Ex: Java)

Antes, alguns conceitos...

- Quando uma linguagem de programação é uma representação simbólica de uma linguagem de máquina ela é chamada de **linguagem de montagem** ou **assembly**
  - A instrução de máquina:
    - 10110000 01100001
  - Pode ser representada por:
    - MOV AL, 61h
  - Que significa mover o número 97 para o registrador AL

Antes, alguns conceitos...

- Assim as linguagens de montagens são uma representação direta das linguagens de máquina
  - Isto é uma instrução de máquina é representada por uma instrução na linguagem de montagem

Antes, alguns conceitos...

- Por exemplo:
  - O comando na liguagem C abaixo não representa uma instrução na linguagem de máquina

    int x = y + 5
  - Já a instrução **ADD AX, 5** representa um único comando de máquina para o processador

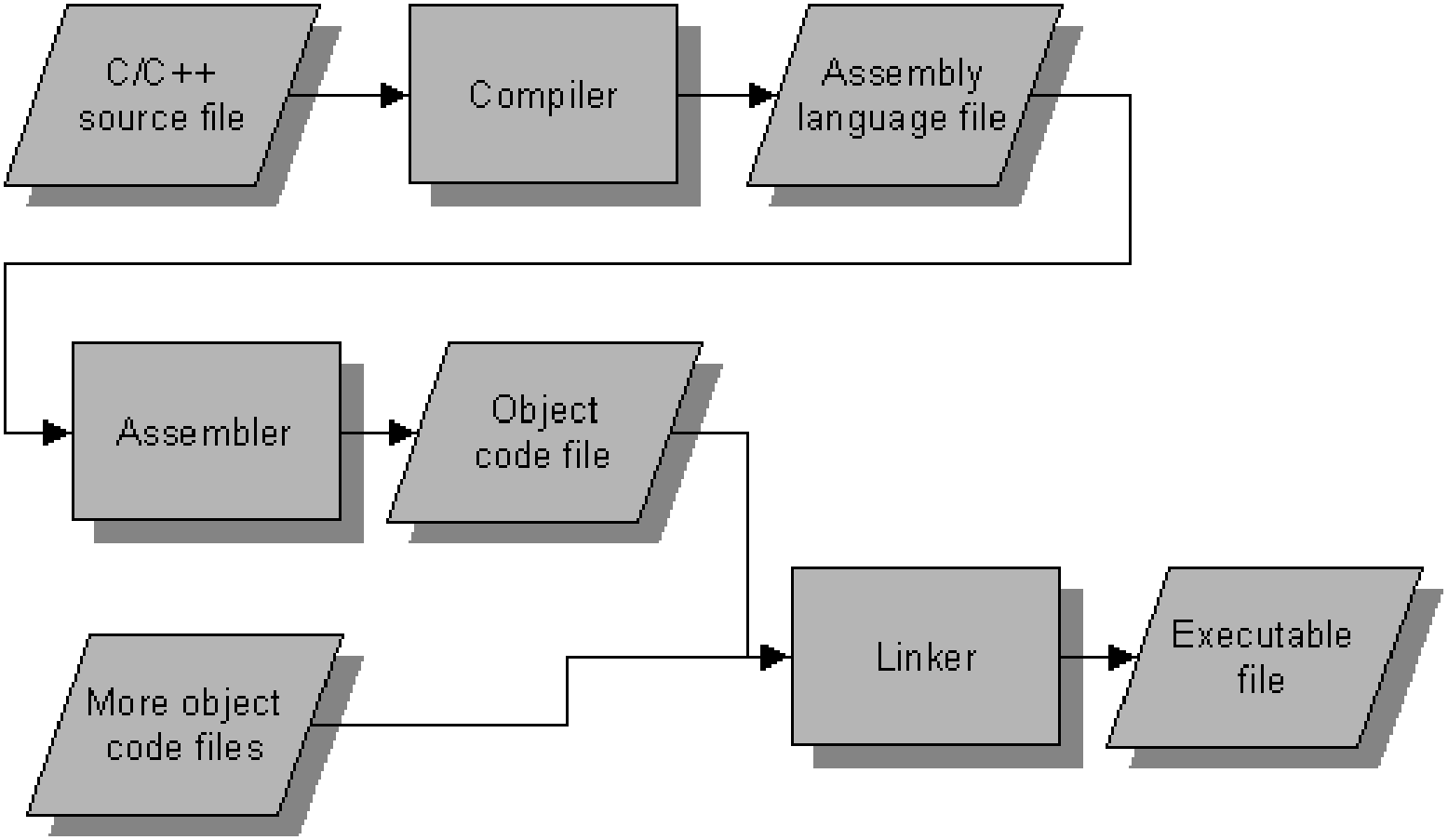
Então...

- Os tradutores podem ser:
  - Assembler (Montador): Quando a linguagem fonte é **uma representação simbólica** da linguagem de máquina
    - Neste caso a linguagem fonte é chamada de linguagem de montagem (assembly language)
  - Compilador: Quando a linguagem fonte é uma **linguagem de alto nível** e a alvo é uma **linguagem de máquina numérica** ou **uma representação simbólica**

Desse modo...

- **Assembler** é o programa que transforma a representação simbólica da linguagem de máquina em programa objeto (executável)
- **Assembly** é a linguagem fonte de representação simbólica da linguagem de máquina

Vejamos:


C/C++
source file
Compiler
Assembly
language file
Assembler
Object
code file
More object
code files
Linker
Executable
file

Linguagem de montagem

- Cada declaração produz uma instrução de máquina
- Facilita o processo de programação
  - Acredite! Pior seria usar somente binários e hexadecimais
- Diretamente ligado a arquitetura do processador
  - Dificulta o reuso para outras arquiteturas

Linguagem de montagem

Linguagem de máquina	Linguagem de montagem
BA0B01	mov dx,msg
B409	mov ah,9
B44C	mov ah,4Ch
CD21	int 21h
48656C6C6F2C20576F	msg db 'Hello, World!',0Dh,0Ah,'$'

Linguagem de montagem

- Existência de linguagens híbridas
  - São constituídas por instruções de alto nível
  - Porém permitem a execução de instruções de máquina
  - Exemplo: Linguagem C
    - Por isso é considerada por muitos autores como uma linguagem de “médio” nível

Linguagem de montagem

- As linguagens de montagens estão diretamente ligadas a arquitetura do processador.
- É possível haver mais de uma linguagem de montagem para uma única arquitetura
  - Como é o caso da arquitetura intel x86, que possui por exemplo:
    - MASM (Microsoft Macro Assembler)
    - TASM (Turbo Assembler)
    - NASM (Netwide Assembler)
    - GAS (GNUassembler)

Programação em linguagem de montagem

- Escrever um programa em linguagem de montagem demora muito mais do que escrever o mesmo em uma linguagem de alto nível
  - Isso inclui: Depuração, manutenção
- Se arquitetura muda, logo o código muda também!
- Em outras palavras, programação em linguagem de montagem **não é trivial!**

Programação em linguagem de montagem

- As linguagens de alto nível possuem uma relação 1-para-muitos com as linguagem de montagem.
  - Por exemplo:

Programa em C:

```
int   Y;
int   X = (Y + 4) * 3;
```

O mesmo programa em assembly:

```
mov    eax,Y
add    eax,4
mov    ebx,3
imul   ebx
mov    X,eax
```

Programação em linguagem de montagem

Por que ainda usar/aprender linguagem de montagem?

Por que usá-la?

- Busca por melhor desempenho
  - O código de desenvolvido em linguagem de máquina pode ser muito menor (ou mais eficiente) do que um código compilado de uma linguagem de alto nível
  - Uso de assembly em pontos críticos de desempenho
    - É comum 10% do código de um programa ser responsável por 90% do tempo de execução

Por que usá-la?

- Acesso direto ao hardware
  - Certos componentes só podem ser acessados por linguagem de montagem
    - Desenvolvimento de drivers
    - Ex: Controladores de dispositivos embutidos de tempo real

Por que usá-la?

- Desenvolvimento de sistemas operacionais
  - Na construção de S.O é imprescindível o uso de programas ou trechos de código em assembly
- Manipulação de vírus e técnicas de engenharia reversa
  - A maior parte dos vírus trabalha com códigos em assembly
  - A engenharia reversa é o processo de tornar o um programa objeto em código assembly para ter acesso ao código-fonte do programa

Por que usá-la?

- Escassez de memória
  - “escovando bits”
- Desenvolvimento de compiladores
  - Ex: Linaro para S.O Android melhoria de mais de 30% no desempenho
- Compreender o real funcionamento do computador
  - São tantas camadas de abstração que esquecemos como o funcionamento realmente é

Por que usá-la?

- Processo de emulação
  - Seja na emulação de vídeo games ou em sistemas de computadores e fundamental o uso de assembly para fazer uma conexão entre as linguagens de máquina dos sistemas

Linguagem de montagem

- Na disciplina utilizaremos a linguagem: Nasm
  - Para Linux:
    - Gcc + nasm
  - Para Windows:
    - Cygwin (inclui módulos do gcc e do nasm)
- No final utilizaremos C + GAS

Linguagem de montagem

- Na disciplina utilizaremos a linguagem: Nasm
  - Teremos aulas práticas no laboratório através de ambiente configurado para as atividades
  - Utilizaremos uma biblioteca para simplificar as instruções de entrada e saída de informações

Linguagem de montagem

Dúvidas??