Linguagem Assembly, Montadores e Ligadores

Universidade Federal de Uberlândia
Faculdade de Computação
Prof. Dr. rer. nat. Daniel D. Abdala

Na Aula Anterior ...

- Apresentação do Simulador MARS;
- Estrutura Geral de um programa em Linguagem de Montagem;

Nesta Aula

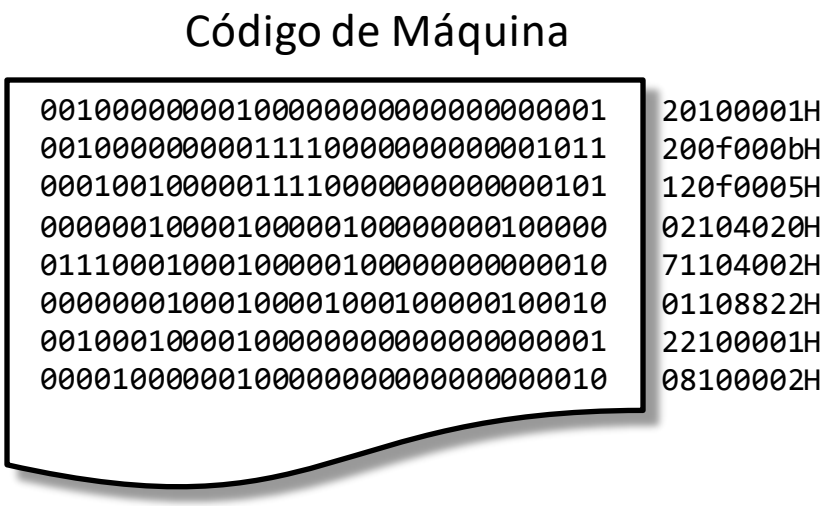
- A linguagem Assembly;
- Montadores;
- Ligadores;
  - Ligação Estática;
  - Ligação Dinâmica;
- Carregadores;
- Algumas palavras sobre Compiladores;
- Otimização de código;
- Programas executáveis;

A Linguagem Assembly – Historicamente ...

- Inicialmente Computadores não eram programáveis;
- Em um segundo momento, foi possível configurar algumas funções (chaves e conecções);
- Em um terceiro momento os computadores implementaram o conceito de programa armazenado em memória:
  - Programas podiam ser escritos e armazenados na memória;
  - A reconfiguração do hardware era executada em tempo de execução;
  - "programa" era escrito diretamente na linguagem do computador (binário).

Código de Máquina

- Processadores entendem apenas 0s e 1s;
- As instruções são especificadas utilizando um código binário;
- Programas e dados são representados desta forma;



Assembly – Linguagem de Montagem

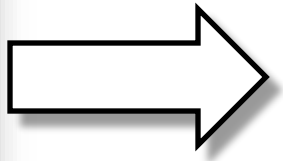
- Código de Máquina é virtualmente ilegível para seres humanos;
- A solução é utilizar um código intermediário, ao qual chamamos de código de montagem;
- Correspondência de 1-1 para o código de máquina – conversão entre código de montagem e código de máquina é trivial;
- No entanto, código de montagem é mais inteligível aos seres humanos.

Linguagem de Máquina

- Na esquerda código de montagem;
- Na direita código de máquina;

Qual é mais inteligível a nós meros humanos?

```
1   .text
2     addi  $s0, $zero, 1
3     addi  $t7, $zero, 11
4   FOR:  beq $s0, $t7, SAIFOR
5     add $t0, $s0,$s0
6     mul $t0, $t0,$s0
7     sub $s1, $t0,$s0
8     addi  $s0, $s0,1
9     j FOR
10  SAIFOR:
```



Código de Máquina

```
00100000000100000000000000000001
00100000000011110000000000001011
00010010000011110000000000000101
00000010000100000100000000100000
01110001000100000100000000000010
00000001000100001000100000100010
00100010000100000000000000000001
00001000000100000000000000000010
```

A Linguagem de Montagem

- Primeira “linguagem de programação”;
- Facilitar o processo de programação;
- Trazer a descrição de programas mais próxima para o domínio do programador;
- Essencialmente, um processo de tradução simples;
  - Linguagem de montagem → código de máquina;
- Novas Possibilidades:
  - Definição de Etiquetas;
  - Conversão automática de números em diversas bases;
  - Detecção de erros sintáticos;
  - Estender o conjunto de instruções → Pseudo-instruções;
  - Introdução do conceito de Macros;
  - Introdução de Modularização;
  - Introdução de Comentários de código.

Linguagens Inteligíveis para Seres Humanos

- Ainda que mais legível e menos susceptível a erros, a linguagem de montagem ainda é muito ligada ao hardware;
- Distante da forma como comumente os seres humanos se expressam;
- Novas linguagens foram criadas para tentar diminuir esta lacuna representacional;

Diferentes Níveis de Descrição

Ling. Matemática

$$x = \sum_{i=1}^{10} i + (i-1)$$

Ling. de Alto Nível

```
⋮
long x, i;
X = 0;
for (i = 1; i < 11; i++)
  x += i + (i-1);
⋮
```

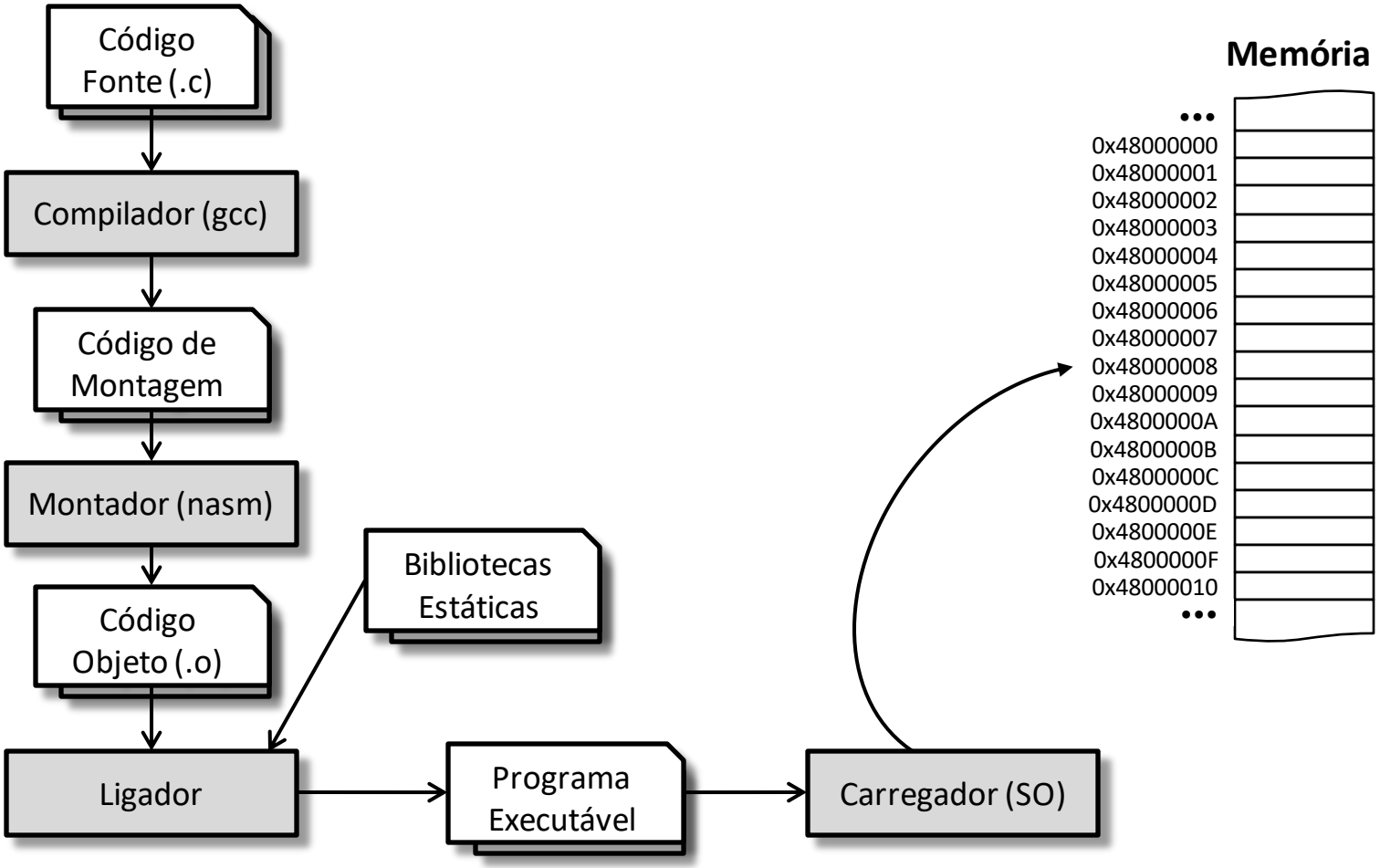
Ling. de Montagem

```
      add  $t0, $zero, $zero
      addi $t1, $zero, 1
L1:   slti $t2, $t1, 11
      beq  $t2, $zero, L2
      addi $t3, $t1, -1
      add  $t4, $t1, $t3
      add  $t0, $t0, $t4
      addi $t1, $t1, 1
      j    L1
L2: •••
```

Código de Máquina

```
00000000000000001000000010000
00100000000010010000000000001
00101001001010100000000000001011
00010001010000000000000000000101
00100001001010111111111111111111
00000001001010110110000000100000
00000001000011000100000000100000
00100001001010010000000000000001
00001000000100000000000000000010
```

Hierarquia de Tradução



O Montador

- Função → Traduzir código de montagem para código de máquina;
- Apenas um programa;
- Em sua encarnação mais simples, funciona em duas passadas:
  1. Verificar a sintaxe das instruções e identificar as etiquetas;
  2. Converter instruções para código de máquina;

O Montador

- No início o montador era simplesmente uma conveniência, para tornar o processo de programação mais “amigável” para o programador;
- Subsequentemente o montador passou a ser uma forma de expandir o conjunto de instruções (Pseudoinstruções);
- Também atua como uma linguagem intermediária entre uma linguagem de alto nível como C ou Pascal e o código de máquina.

O arquivo Objeto

- Arquivo Objeto → seis campos:
  - **Cabeçalho** → descreve o tamanho e posição dos demais campos;
  - **Segmento de Texto** → as instruções do programa em código de máquina;
  - **Segmento de Dados Estáticos** → dados a serem alocados estaticamente na memória;
  - **Informação de Relocação** → informação que identifica instruções e datos que dependem de endereços absolutos quando o programa é carregado em memória;
  - **Tabela de Símbolos** → símbolos que não são definidos no arquivo, tal como referências externas;
  - **Informação de Depuração** → informação concisa de como o módulo foi compilado de modo que o depurador possa associar instruções de máquina e código de alto nível.

Ligadores

- O montador traduz diversos arquivos fonte em arquivos objeto;
- A função do ligador é combinar todos os arquivos objetos em um único arquivo executável;
- Ele utiliza a informação de relocação e a tabela de símbolos para resolver referências entre arquivos;
- Ele também liga estaticamente (copia) o código de funções de bibliotecas pré-compiladas (.so);

Ligação Estática e Dinâmica

- Há duas formas de se utilizar funções de bibliotecas em um programa;
  - Funções de Bibliotecas Estáticas;
  - Funções de Bibliotecas Dinâmicas;
- Na Ligação Estática o código objeto da função a ser ligada é copiado para o arquivo executável;
- Na ligação dinâmica, o SO carrega a priori ou sob demanda o código das bibliotecas (.lib, .dll);
  - Símbolos externos são adicionados identificando as funções dinâmicas;
  - O endereço real destas funções é definido durante o processo de carregamento do programa.

Carregador

- Parte do Sistema Operacional;
- Responsável por carregar um programa executável em memória;
- Na realidade, O carregador faz diversas chamadas ao Sistema Operacional:
  - Cria um Descritor de Processo;
  - Selecionar um espaço de endereçamento;
  - Carregar na memória os dados estáticos;
  - Carregar na memória o texto do programa;
    - Resolver os endereços dinâmicos de chamadas de funções;
  - Agendar o novo programa como pronto para execução;

Algumas Palavras sobre Compiladores

- No início as linguagens de programação de alto nível eram uma conveniência a mais para os programadores (FORTRAN, COBOL);
- Mais simples de expressar problemas numéricos ou de processamento de dados que assembly ou código de máquina;
- No entanto, um passo de tradução adicional se fazia necessário;
- Quanto mais próximo do usuário, mais expressiva é a linguagem e consequentemente mais complexa;
- Assembly requer apenas análise léxica e sintática;
- As linguagens de alto nível requerem análise léxica, sintática e semântica;

Algumas Palavras sobre Compiladores

- Ainda assim, as linguagens de programação de alto nível não admitem ambiguidade tal como comumente encontramos em linguagens naturais como o português;
- Há um aumento de expressividade, mas a um custo:
  - A compilação é um processo intensivo que demanda poder computacional;
  - No início o código de montagem gerado pelos compiladores era de longe inferior ao gerado por um ser humano;
  - Para geração de código eficiente, os compiladores necessitam de passos de otimização;

Otimização de Código

- Geração de código de montagem é um processo mecânico;
- Diversas otimizações que seres humanos normalmente utilizam não são diretamente evidentes para o compilador;
- Solução: Implementá-las explicitamente!
- Muito custoso do ponto de vista computacional;
- Hoje em dia, compiladores modernos conseguem gerar código comparável ao gerado por um programador experiente, limitando em muito a necessidade de se programar diretamente em linguagem de Montagem;
- Geralmente não são automaticamente habilitadas na sua IDE preferida;

Programas Executáveis

- O formato ou seja, a organização dos bits que compõem um programa armazenado é dependente do sistema operacional;
- Um programa executável, deve conter diversas informações além dos dados estáticos e código do programa:
  - Tabela de relocação;

Bibliografia Comentada

- **PATTERSON, D. A. e HENNESSY, J. L. 2014.** *Organização e Projeto de Computadores – A Interface Hardware/Software.* Elsevier/ Campus 4ª edição.

- **HENNESSY, J. L. e PATTERSON, D. A. 2012.** *Arquitetura de Computadores – Uma Abordagem Quantitativa.* Elsevier/ Campus 5ª edição.

Bibliografia Comentada

- **MONTEIRO, M. A. 2001.** *Introdução à Organização de Computadores.* s.l. : LTC, 2001.

- **MURDOCCA, M. J. e HEURING, V. P. 2000.** *Introdução à Introdução de Computadores.* 2000. 85-352-0684-1.

Bibliografia Comentada

- **STALLINGS, W. 2002.** *Arquitetura e Organização de Computadores.* 2002.

- **TANENBAUM, A. S. 2007.** *Organização Estruturada de Computadores.* 2007.