Projeto de um Montador Relocador e Ligador Relocável

Sistemas de Programação – PCS3216

Luciana da Costa Marques

8987826

Introdução

Esta documentação refere-se ao projeto do montador relocável e do ligador. Para realiza-lo, foram utilizados os conhecimentos adquiridos em aula, bem como os slides de aula e explicações dadas pelo professor João José Neto.

Optou-se por utilizar a linguagem de programação c++, devido à familiaridade com a ferramenta e por ter implementações de programação orientada a objetos, recurso também utilizado na lógica de ambas as partes do projeto.

Além de programação orientada a objetos, utilizou-se também a lógica de Motor de Eventos, aprendida para o primeiro exercício programa da disciplina.

Todas as informações relevantes quanto à estrutura do programa estão devidamente descritas nos capítulos 1.1 e 1.2. Os capítulos 2 e 3 ilustram o funcionamento do montador e do ligador, respectivamente, e o capítulo 4 traz conclusões e reflexões acerca do trabalho.

Estrutura lógica do Montador

Conforme descrito na introdução deste relatório, utilizou-se programação orientada a objetos e a estrutura do Motor de Eventos na implementação do Montador.

```
//Eventos do Montador
Evento* LeituraLinha = new Evento("LEITURA LINHA");
Evento* TratarLinha = new Evento("TRATAR LINHA");
Evento* AnalisarMneumonico = new Evento("ANALISAR MNEUMONICO");
Evento* AnalisarOperando = new Evento("ANALISAR OPERANDO");
Evento* Fim = new Evento("FINAL");
```

Para o Montador, tem-se cinco eventos: LEITURA DE LINHA, TRATAR LINHA, ANALISAR MNEUMONICO, ANALISAR OPERANDO e FINAL. O papel deles na lista de eventos criada para o montador é descrita a seguir:

- **LEITURA DE LINHA:** neste evento, lê-se uma linha do arquivo. Se estiver no passo 2, a linha é listada. Adiciona-se o evento TRATAR LINHA à lista de eventos.

- **TRATAR LINHA:** capta-se o mneumonico e o símbolo escrito depois dele. Se estiver no passo 1, checa-se se o símbolo já existe na tabela de símbolos, que foi implementada como lista ligada. Se não estiver, adiciona-se o símbolo a ela como "indefinido". Se a linha estiver definindo um símbolo (caso em que encontra-se "K" no campo de mneumônico), procura-se o símbolo na tabela e atribui-se a ele o endereço armazenado em pc. Ao final deste evento, atualiza-se pc conforme o mneumonico (se "K", pc = pc + 1, e para os demais, pc = pc + 2). Se passo é 1, adiciona-se à lista de eventos TRATAR LINHA, para captar uma nova linha, e do contrário adiciona-se ANALISAR MNEUMONICO.
- **ANALISAR MNEUMONICO:** Este evento ocorre apenas se passo é 2. Primeiro percorre-se a tabela de Mneumonicos (também implementada como lista ligada). Caso o mneumonico exista e exija operando, é adicionado à lista de eventos o evento ANALISAR OPERANDO. Caso o mneumônico não esteja na lista, primeiro verifica-se se no campo de mneumônico está "K". Se sim, trata-se da definição de um símbolo, e capta-se seu código, o qual é escrito no código objeto. Por fim adiciona-se o evento LEITURA DE LINHA para fazer a leitura de uma nova linha.
- **ANALISAR OPERANDO:** este evento também só ocorro se estiver no passo 2 e se o mneumônico captado na linha exigir operando. Ele calcula o código da instrução do comando conforme o opcode do mneumônico e o endereço do símbolo escrito depois do mneumonico na linha. Feito o cálculo, escreve-se o código da instrução no código objeto. Adiciona-se o evento LEITURA DE LINHA à lista de eventos.
- **FINAL**: Este evento é inserido na lista no momento de sua criação, e sua execução indica que: ou chegou-se ao final do passo 1, tendo lido todas as linhas do programa e agora passando para o passo 2 (inserindo o evento "LEITURA DE LINHA" à lista de evento), ou que chegou-se ao final do passo 2 e tem-se o código objeto gerado, finalizando-se o programa.

Além da lista de eventos, foram também implementadas as principais estruturas de dados necessárias: tabelas de símbolos e tabela de mneumônicos, ambas como listas ligadas.

Estrutura do Arquivo Lido pelo Montador

O Montador deve fazer a leitura de um arquivo escrito em linguagem assembly, seguindo-se a estrutura:

<mneumonico><espaço><código>

No campo <mneumonico> deve haver o código do comando conforme aqueles definidos na MVN ou das pseudo-instruções comentadas em aula.

No campo <espaço> deve haver um único espaço em branco.

No campo <código> deve haver um símbolo/rótulo, que deverá ser definido nas últimas linhas do programa, seguindo-se a estrutura:

K<espaço><código><espaço><valor da constante>

Ou seja, caso a linha comece com a letra "K" seguida de um espaço, o programa entende que se está definindo uma constante/símbolo e atualiza a tabela de símbolos.

EXEMPLO DE FUNCIONAMENTO DO MONTADOR

Para demonstrar o funcionamento do montador, foi utilizado um código exemplo passado em aula e que pode ser visto no arquivo “programaexemplo3.txt”, o qual representa o programa fonte. O seu código correspondente gerado pode ser visto no arquivo “código.txt”. Junto com esta documentação, está sendo enviado também o arquivo Montador.exe, de modo que é possível alterar o arquivo do programa fonte para gerar diferentes códigos montados.

Cada linha do código montado é feito na seguinte maneira:

<endereço><espaço><instrução>

No campo de endereço está o byte de início em que a instrução deve ser gravada. Para cada instrução, deve ser armazenado na memória um espaço de dois bytes.

Para as linhas em que há definição de constantes, optou-se por deixar disponível apenas um byte e segue a seguinte estrutura:

<endereço><espaço><valor da constante>

Para o arquivo fonte original, o Montador cria a seguinte tabela de Símbolos: