Universidade Tecnológica Federal do Paraná

Campus Apucarana

ARCO3A – Arquitetura e Organização de computadores

Projeto:

Mini simulador MIPS

Arnald Souza, RA: 2271923

Carlos Eduardo da Silva Ribeiro, RA: 2271931

João Pedro Neigri Heleno, RA: 2270323

Sandro Pinheiro Christe, RA: 2270404

Professor: André Luiz Tinassi D’ Amato

Apucarana – PR

2021

**Resumo**

**Introdução**

A linguagem C é uma das linguagens de alto nível criada que mais prosperou e que é usada com mais frequência já a algum tempo. Como afirma André Backes em seu livro. (BACKES, 2013, p.2).

“A linguagem C é uma das mais bem-sucedidas linguagens de alto-nível já criadas e considerada uma das linguagens de programação mais utilizadas de todos os tempos. Define-se como linguagem de alto nível aquela que possui um alto nível de abstração relativamente elevado, que está mais próximo da linguagem humana do que do código de máquina.”

Em C, a compilação parte do código-fonte que é a maneira em que os humanos possam entender o que está sendo feito, de modo que esse código não é entendido pelo computador. Para isso, o código-fonte é traduzido para o código de máquina. O ato de traduzir é chamado de compilação.

A compilação é dividida em 4 etapas, apesar de ser entendida como apenas uma. Essas etapas são:

• **Pré-processamento**: antes de iniciar a compilação do nosso código-fonte, o arquivo é processado por um pré-processador. O arquivo de código-fonte é convertido em outro arquivo de código-fonte “expandido”. Nessa etapa, ocorrem a remoção dos comentários e a interpretação das diretivas de compilação utilizadas, as quais se iniciam com #. (BACKES, 2013, p.13).

• **Verificação sintática**: aqui se verifica se o código-fonte foi escrito corretamente, de acordo com a linguagem C. Basicamente, a verificação sintática procura por erros de sintaxe como, por exemplo, parênteses que não foram fechados, falta de ponto e vírgula no final de uma instrução etc. (BACKES, 2013, p.13).

• **Compilação**: cada arquivo de código-fonte do seu programa é processado, sendo criado um arquivo “objeto” para cada um deles. Nessa etapa, não é gerado nenhum arquivo que o usuário possa executar. Em vez disso, o compilador produz as instruções de linguagem de máquina que correspondem ao arquivo de código-fonte compilado. (BACKES, 2013, p.14).

• **Link-edição**: o trabalho do link-editor é unir todos os arquivos “objeto” que fazem parte do programa em um único arquivo executável, o programa propriamente dito. Isso inclui tanto os arquivos objeto gerados na etapa de compilação a partir dos arquivos de código-fonte do programa como os arquivos objeto que foram gerados a partir das bibliotecas usadas. (BACKES, 2013, p.14).

**Objetivos**

Exercitar e fixar os conhecimentos adquiridos sobre linguagem assembly e arquitetura de computadores. Reforçar o conhecimento sobre decisões de projeto e funcionamento de hardware. Exercitar conceitos sobre integração software e hardware por meio de simulação.

**Descrição**

Criar um simulador para uma versão do processador MIPS de 32 bits. O mesmo deverá ler um programa em código binário e executar algumas instruções.

Também deverá ter uma interface para facilitar o acompanhamento da evolução do programa e quais o estado de cada uma das instruções em execução (busca, decodificação, execução). A interface deverá mostrar qual instrução está sendo executada em cada passo. O subconjunto de instruções que deverá ser implementado no simulador é apresentado pela figura 1.

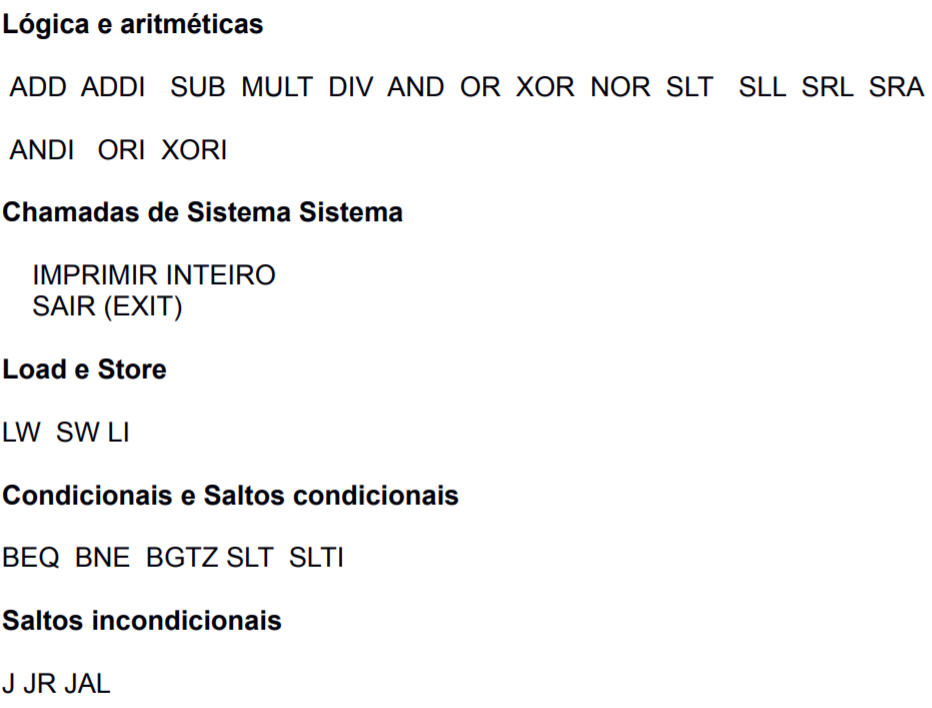


Figura 1 - Instruções a serem implantadas no MIPS 32 bits.

**Resultados obtidos**

**Conclusão**

**Referências**