

Trabalho Prático I de Organização de Computadores I

O objetivo principal do trabalho era implementar uma versão simplificada de um montador MIPS, para isso o grupo optou usar a linguagem “C”.

A implementação consiste em dois arquivos principais: *Conversores.c*, que possui as funções: *Or_Bits_finais*, *dec2bin*, *Somador_binario*, *Complemento2*, *Fprintf_registrador*, *Or_Bits_SLL_SRL*:

- ***Or_Bits_finais()***: Função criada para adicionar no fim da palavra de 16 bits obtido através do conversor para binário.
- ***dec2bin()***: Função usada para converter números decimais para binário.
- ***Somador_binario()***: Função para realizar a soma de binários.
- ***Complemento2()***: Função que transforma um número em binário em um número binário negativo. Essa função é chamada em casos de chamadas que existe a possibilidade se ser usado um número negativo.
- ***Fprintf_registrador()***: Função criada para identificar o registrador passado por parâmetro e imediatamente escrever saída em linguagem de máquina em um arquivo.
- ***Or_Bits_SLL_SRL()***: Função criada para adicionar no fim da palavra de 5 bits obtido através do conversor para binário.

Arquivo *Montador.c*: Neste está a função principal *main*, nela, está implementada toda a leitura de arquivo, o método usado foi o de conferência por strings, cada *if()* representa uma chamada de MIPS, como *add*, *sub*, *or*, *and* e outras instruções, sendo que, para cada chamada delas é feita uma formatação diferente do *fscanf()*, faz-se a escrita no arquivo de saída de acordo com cada uma das operações.

Instruções extras implementadas:

As seguintes instruções extras foram criadas para pontuação extra:

Pseudo-instruções:

- **MOVE**: Copia o valor de um registrador para outro registrador.

Instruções de carregamento de dados :

- **SW**: Instrução de transferência de dados que copia dados de um registrador para a memória.
- **LW**: Instrução de transferência de dados que copia dados da memória para um registrador.

Instruções de desvio:

- **BNE**: Branch if not equal (Desviar se não for igual).
- **BEQ**: Branch if equal (Desviar se for igual).

Instruções para execução:

Caso deseje compilar novamente os arquivos de código fonte siga as instruções:

- Em sistema operacional **linux**: dentro da pasta do projeto deve-se abrir o terminal e digitar o comando:
make
- ou:
gcc Montador.c -o Montador Conversores.c
- E em seguida para executar, o comando:
make run
ou:
./Montador
- Em sistema operacional **windows**: deve primeiramente se certificar de que o compilador GCC esteja instalado e devidamente configurado, em seguida

abra a pasta onde se encontra os códigos fontes a partir do cmd e entre com o comando:

-

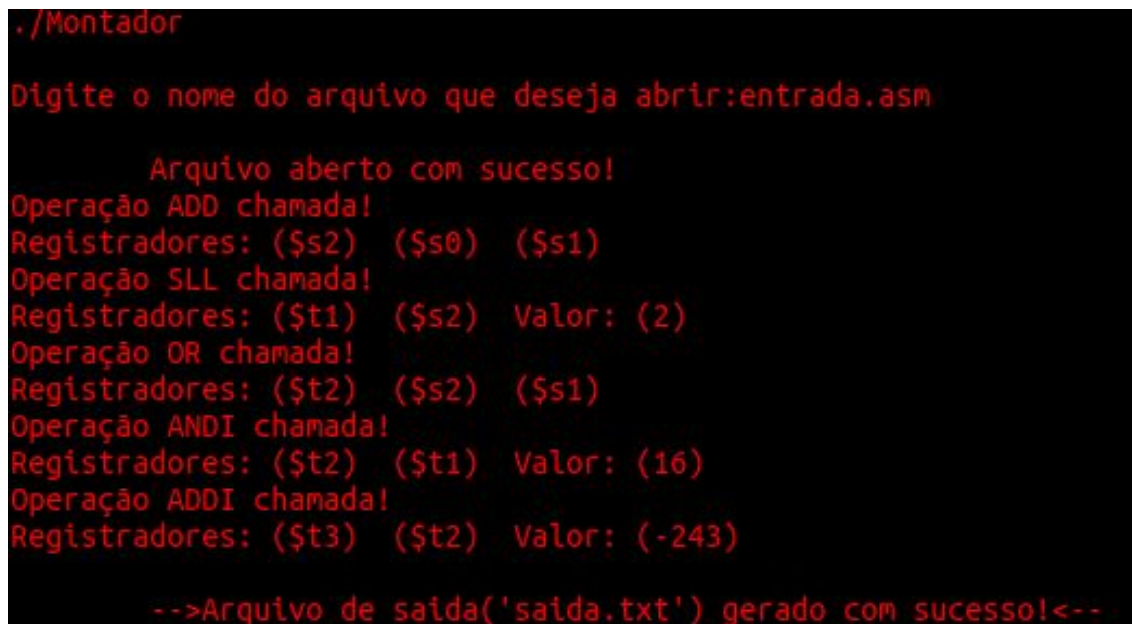
gcc Montador.c -o Montador Conversores.c

- Em seguida um arquivo “*Montador.exe*” será criado, bastando apenas executá-lo.

Lembrando que os arquivos executáveis (compilados) já se encontram previamente compilados e adicionados ao arquivo .zip .

No momento de execução do programa, inicialmente será requisitado o nome do arquivo de entrada com a extensão .asm , em seguida será impresso na tela todas as operações identificadas na linguagem em assembly, ao final teremos um arquivo de saída gerado na mesma pasta em que o programa se encontra com o nome de “*saida.txt*”.

Como exemplificado na imagem:



```
./Montador
Digite o nome do arquivo que deseja abrir:entrada.asm

    Arquivo aberto com sucesso!
Operação ADD chamada!
Registradores: ($s2) ($s0) ($s1)
Operação SLL chamada!
Registradores: ($t1) ($s2) Valor: (2)
Operação OR chamada!
Registradores: ($t2) ($s2) ($s1)
Operação ANDI chamada!
Registradores: ($t2) ($t1) Valor: (16)
Operação ADDI chamada!
Registradores: ($t3) ($t2) Valor: (-243)

-->Arquivo de saída('saida.txt') gerado com sucesso!<--
```

Grupo formado por:

Arthur De Bellis Gomes - 3503

Samuel Pedro Sena - 3494

Pablo Ferreira - 3480

Todo o desenvolvimento e distribuição do trabalho encontra-se disponível no Github no seguinte link:

https://github.com/Globson/Montador-ASSEMBLY_MIPS-TP-OC