Universidade Federal de Lavras

Departamento de Ciência da Computação
Bacharelado em Ciência da Computação
Disciplina: Arquitetura de Computadores I – GCC117 – 2022/2
Professor: Luiz Henrique A. Correia

TRABALHO PRÁTICO 2

Alunos :
Bryan De Lima Naneti Barbosa
Gustavo Soares Silva
Rafael Furtado Moraes

Lavras - MG
2023

Sumário

1. Introdução

Este trabalho prático tem como objetivo estudar o funcionamento de um processador MIPS monociclo. O desenvolvimento deste trabalho é a base para a implementação de um simulador funcional de um processador e contribui para o aprendizado dos conceitos empregados na operação de um processador. Para a implementação do simulador foi utilizada a linguagem de programação C++ , por estarmos mais familiarizados com a linguagem.

2. Implementação

A primeira parte da implementação foi o desenvolvimento do Interpretador de Instruções. Para isso utilizamos a estrutura de Mapa, da biblioteca "Map", para mapear os Labels e identificar os Opcodes de cada instrução. Também foi necessário usar a biblioteca Bitset para armazenar as instruções decodificadas em 32 bits. Cada instrução é decodificada individualmente, por linha, e armazenada num arquivo binário.

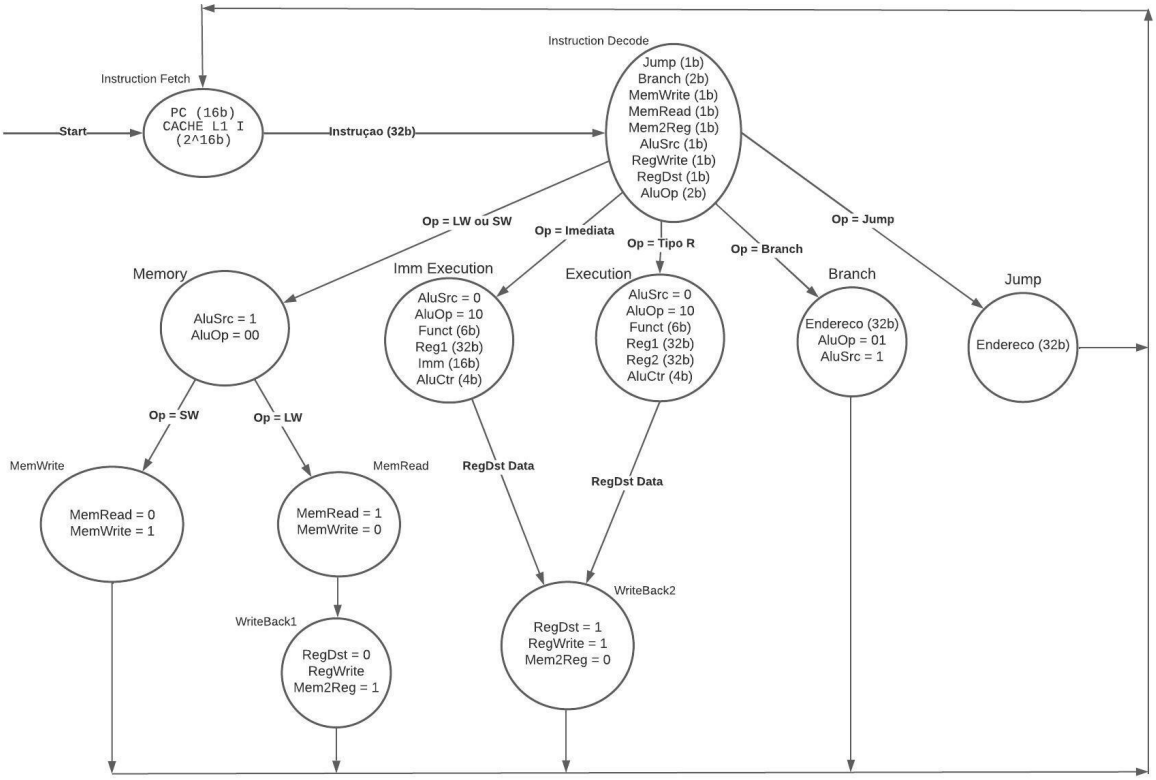
Em seguida foi criado o arquivo do processador, em que a chamada de todas as funções são feitas nele. Mas, durante o desenvolvimento desse arquivo, viu-se necessário criar um cabeçalho para armazenar o Banco de Registradores, a Memória (BancoMemoria) e outras funções que seriam usadas por cada uma das etapas do processador. Nessas classes foram criados métodos “getRegistrador”, para retornar um valor armazenado em um endereço, e “setRegistrador”, para armazenar um valor em um endereço. A “Memória Cache L1 de Instruções” não foi implementada. Ao invés disso, as instruções são buscadas diretamente no arquivo binário de entrada

No início ficamos em dúvida sobre a separação de cada etapa do processamento, pois teríamos que retornar vários valores de uma vez. Dessa forma, ao invés de usar funções, optamos por usar classes e structs. A entrada dos valores foi feita pelo construtor da classe, e a saída foi feita pelos atributos, que são utilizados como parâmetros pela próxima etapa do processamento.

Além disso, para as operações de soma, subtração, multiplicação, entre outras, foram feitas sobrecargas dos operadores “+”, “-”, “*”, “/” e “<” para a classe bitset. E também um conversor de bitset para inteiro. Também foram feitos métodos de “recorte”, que retorna um pedaço de um bitset, “signalExtension”, para estender um bitset de 16b para 32b, e “bitsetToInt”, que converte o valor de um bitset para inteiro.

E, para contornar um problema que encontramos durante o desenvolvimento, tivemos que modificar uma parte do caminho de dados do monociclo. Para funcionar as instruções “bne” e “bge” nós tivemos que adicionar um bit na flag de branch. E para executar todas as operações na execução, também tivemos que acrescentar um bit na saída do AluControl. E o cálculo do desvio não é feito no ID, e sim na Execução.

3. Diagrama da máquina de estados



4. Dificuldades encontradas

Foram encontradas dificuldades em decidir qual a melhor maneira de se estruturar o simulador funcional do processador MIPS, pois haviam muitas possíveis maneiras de se dividir o problema, tanto em classes, como em métodos, além das várias maneiras de se simular os componentes de hardware, como a memória, o contador de programa e entre outros.

Além disso, encontramos dificuldades em fazer o interpretador de instruções, pois é bem difícil pensar em todas as possibilidades de entrada escritas pelo usuário. Por isso decidimos usar um padrão para escrita do programa.

Outra dificuldade que encontramos durante a produção do programa foi desenvolver as funções “jal”, “jr” e “mult”. Isso ocorreu devido à falta de tempo por conta de diversos outros trabalhos e provas de outras disciplinas durante o período estipulado para o trabalho. A função “div” não armazena em dois registradores (HI e LO) por conta da forma com que foi pedido para fazer o trabalho, sem especificar os registradores, e sim utilizar apenas números para referenciá-los (0-31).

5. Como usar o software

Para utilizar o software é necessário seguir alguns passos. Primeiramente, escreva o código que será executado no arquivo “programa.txt”. O código escrito deverá seguir o padrão descrito abaixo (utilize os exemplos para entender melhor).

- Não utilize tabs ou espaços antes de qualquer instrução. As instruções devem ser escritas “coladas” na parte esquerda, sem espaços ou tabs.
- Labels são permitidos com a instrução logo à frente ou com uma quebra de linha (apenas uma), sem espaços antes ou depois. Todo label deve terminar com “:” (Ex: Loop:).
- Tanto Labels (caso tenha uma instrução na frente) quanto o “nome” das instruções devem ser seguidos de um espaço, separando o que vier à frente (Ex: Label: add $...).
- Os registradores são referenciados com um “$” seguido do número do registrador, logo depois uma vírgula, e se tiver algo à frente coloque um espaço para separar (Ex: add $2, $2, $3).
- Não coloque linhas vazias entre instruções, mantenha uma instrução ou label por linha.

Em seguida execute o arquivo “InterpretadorDeInstruções.cpp” e verifique se o arquivo foi Interpretado sem erros e o arquivo “programa.bin” foi gerado.

Se sim então execute o arquivo “processador.cpp”.

6. Exemplos de testes

Exemplo de loop:

```
addi $8, $0, 1
addi $9, $0, 10
loop: addi $8, $8, 1
slt $10, $8, $9
bne $10, $0, loop
```

Exemplo de Fatorial:

```
addi $3, $0, 1
addi $1, $0, 1
addi $5, $0, 11
LOOP:
slt $2, $1, $5
beq $2, $0, LSair
mul $3, $3, $1
addi $1, $1, 1
j LOOP
LSair:
```

Resultado Esperado: 3628800

Exemplo de Load e Store:

```
addi $20, $0, 100
addi $5, $0, 10
addi $1, $0, 0
loop:
beq $1, $5, exit
add $2, $1, $20
sw $1, 0($2)
addi $1, $1, 1
j loop
exit: lw $3, 5($20)
```

7. Conclusões

Neste trabalho podemos ter um entendimento melhor sobre os conceitos empregados na operação de um processador monociclo, através do simulador MIPS desenvolvido, deixando mais clara a leitura e entendimento de datapath's, a divisão dos estágios em um monociclo, bem como os demais conceitos associados à disciplina, contribuindo de forma fundamental para o aprofundamento do aprendizado.

8. Referências bibliográficas

C++ bitset and its application. Disponível em: <https://www.geeksforgeeks.org/c-bitset-and-its-application/>. Acesso em: 28 fev. 2023.

Opcodes :: Plasma - most MIPS I(TM) opcodes :: OpenCores. Disponível em: <https://opencores.org/projects/plasma/opcodes>.