Universidade Federal do Rio Grande do Norte
Centro de Tecnologia
Departamento de Engenharia de Computação e Automação

Trabalho – Simulação de *Datapath* MIPS 32 bits

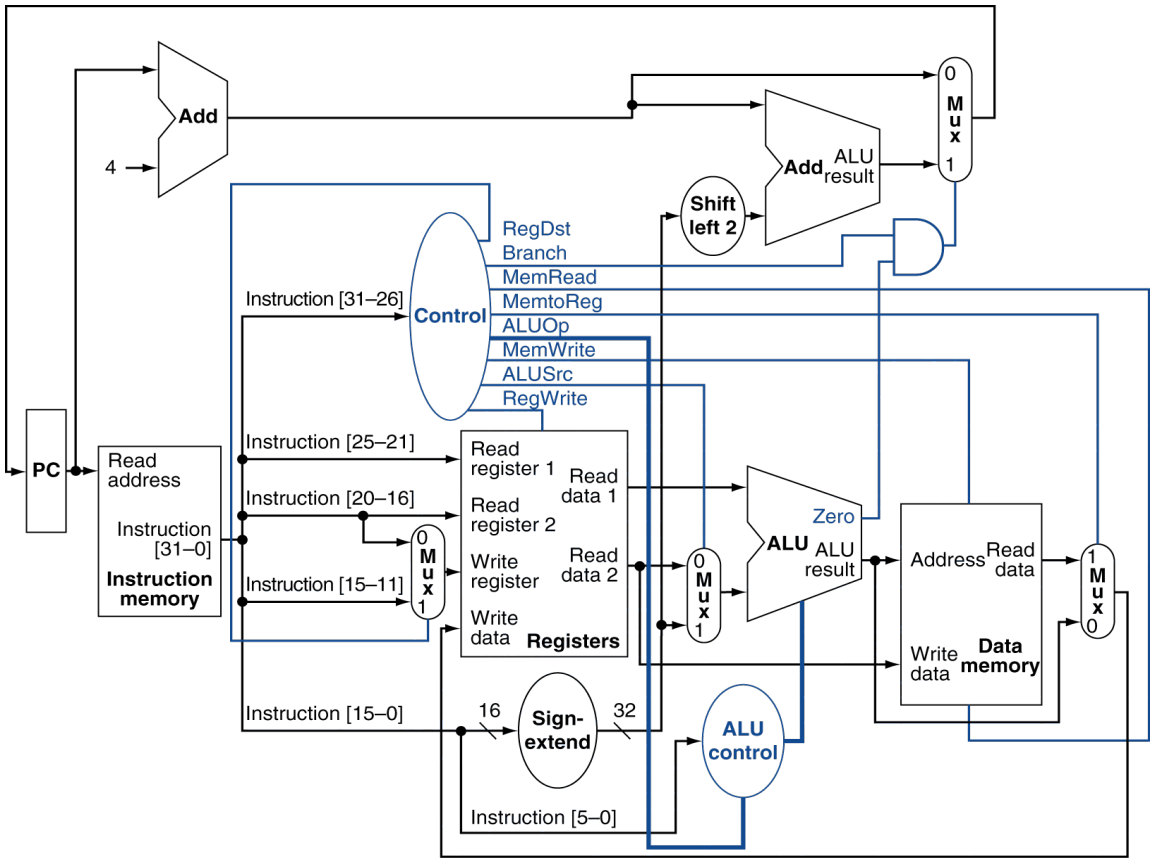
Disciplina: DCA0104 – Arquitetura de Computadores
Turma: 01, 2019.2
Horário/Local: 24M12, 4A10
Professor: Diogo Pinheiro Fernandes Pedrosa
DCA, sala 106, diogo@dca.ufrn.br

Objetivos

- Fixar a compreensão sobre o caminho de dados de processadores com arquitetura MIPS de 32 bits;
- Entender qual é o processo de execução de instruções no processador.

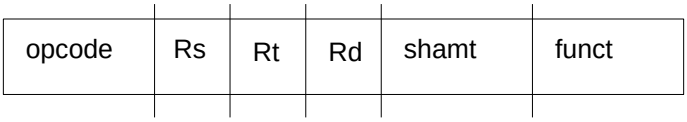
Tarefa

A tarefa principal deste trabalho é desenvolver um programa que simule o comportatmento do caminho de dados de um processador MIPS de 32 bits, cujo diagrama de blocos é apresentado na figura a seguir.

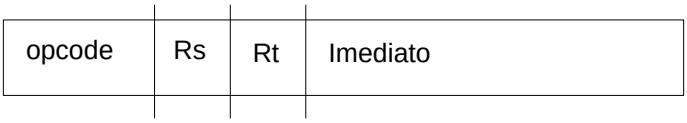


Para lembrar, as instruções MIPS são representadas por 32 bits e apresentam os seguintes formatos:

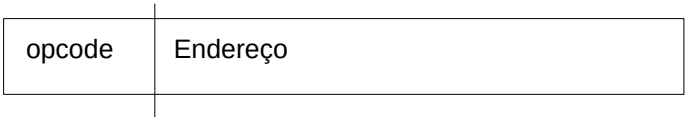
Tipo R:



Tipo R:



Tipo R:



Os registradores Rs, Rt e Rd podem ser um dos seguintes registradores, com seus respectivos identificadores:

- `$zero` → 0
- `$t0` até `$t7` → 8 até 15
- `$s0` até `$s7` → 16 até 23
- `$t8` e `$t9` → 24 e 25

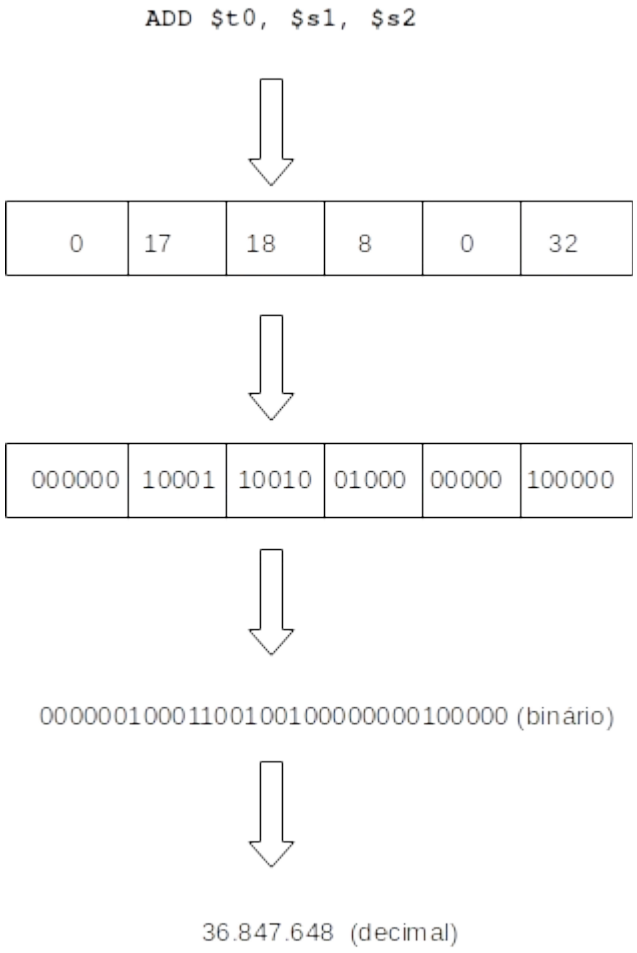
Os *opcodes* das instruções de interesse são (valores em decimal):

- `ADD` → 0 (campo FUNCT = 32) [tipo R]
- `ADDI` → 8 [tipo I]
- `BEQ` → 4 [tipo I]
- `J` → 2 [tipo J]
- `LW` → 35 [tipo I]
- `SW` → 43 [tipo I]
- `SUB` → 0 (campo FUNCT = 34) [tipo R]

O programa simulador deve ser capaz de executar essas instruções citadas (obrigatoriamente as instruções `ADD`, `ADDI` e `SUB`). O teste para validar o funcionamento do programa deve ser feito com a seguinte sequência de instruções:

```
ADDI $s1, $zero, 10
ADDI $s2, $zero, 2
ADDI $s3, $zero, 8
ADDI $s4, $zero, 6
ADD $t0, $s1, $s2
ADD $t1, $s3, $s4
SUB $s0, $t0, $t1
```

O processo de simulação deve ser feito de reproduzir, de forma mais fiel possível, o procedimento de execução das instruções acima. A forma como essas instruções alimentam o simulador é escolha do grupo de desenvolvimento. Pode ser por leitura de um arquivo texto contendo as “instruções” a serem executadas ou pode ser pelo usuário digitando a instrução na sequência (neste caso, com o usuário fazendo o papel do contador do programa e também da memória de instruções). As instruções podem ser fornecidas no modo mnemônico ou no formato numérico (sendo convertida previamente pelo grupo, como exemplificado na figura a seguir). O mais importante é que os campos que compõem as instruções sejam obtidos e usados ao longo do caminho de dados simulado.



As instruções que não aparecem no programa de teste não precisam ser reproduzidas (como `BEQ`, `J`, `LW` e `SW`). Caso o grupo deseje simulá-las (todas ou somente algumas delas), deve-se adicionar instruções ao programa de teste para que eles também possam ser validadas.

O Que Apresentar?

A apresentação dos resultados obtidos deverá ser feita por meio de um relatório elaborado pelo grupo de trabalho. Para isto, utilize os modelos de artigo científico da Sociedade Brasileira de Computação que estão disponibilizados na turma virtual, no SIGAA, para estruturar o trabalho. Neste relatório, não há necessidade dos campos *Resumo* e *Abstract*.

O relatório deve conter:

- Uma descrição breve de como se constitui o caminho de dados abordado, principalmente do modelo chamado “monociclo”;
- Descrição teórica do processo de execução dos tipos das instruções que são utilizadas no programa de exemplo (tipos R e I, somente), verificando também quais sinais de controle são mais relevantes para que elas sejam executadas;
- Descrição do processo de construção do simulador, explicando quais as estruturas, funções, tipos de dados ou outras características foram usadas no código. O código fonte pode ser posto como anexo, no relatório, ou ser disponibilizado por link em algum repositório de código na rede (como GitHub, por exemplo). A linguagem de programação usada é de total escolha do grupo;
- Resultados obtidos (quais valores ficaram armazenados nos “registradores”); e
- Uma análise teórica sobre ocorrência de falhas nos sinais de controle no processador simulado: se o sinal que habilitasse a escrita de resultados no banco de registradores ficasse permanetemente com valor lógico igual a “0” (RegWrite (ou EscreveReg) = 0), independente da instrução executada, o que aconteceria com as instruções do programa exemplo?

Este trabalho pode ser elaborado por grupos de até 4 componentes. Quem desejar, pode fazer individualmente.

Preferencialmente solicito que o relatório seja entregue fisicamente até a data final apresentada na tarefa do SIGAA.

O texto para referência está no livro “Organização e Projeto de Computadores: A Interface Hardware/Software”, de David A. Patterson e John L. Hennessy. Na 3ª edição, a teoria pode ser encontrada nos capítulos 2 e 5.