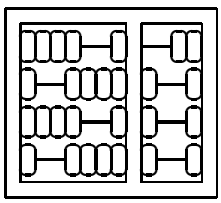

MC542
Organização de Computadores: Teoria e Prática

Trabalho 1

IC – UNICAMP

1 Entrega

Este trabalho deverá ser realizado em **grupo de dois alunos**. Um membro do grupo (**e somente um**) deverá mandar um e-mail, até 01/12/2004, para ducatte@ic.unicamp.br com *subject*: mc542: R_datapath - Grupo e no corpo da mensagem os RAs dos componentes do grupo (um RA por linha).

A data de entrega do trabalho será anunciada oportunamente na página do curso.

Escreva o seu projeto, com nome R_datapath, em um diretório cujo nome é o seu RA. Realize simulações comportamental e com *timing* e apresente os resultados em um relatório a ser entregue impresso em papel (formato carta). Além do relatório compacte o diretório do seu projeto e envie por e-mail para ducatte@ic.unicamp.br com *subject*: mc542: R_datapath.

2 Objetivo

O objetivo deste trabalho é projetar e simular alguns componentes básicos do processador MIPS usando VHDL. Neste projeto iremos projetar e simular uma versão simplificada do processador MIPS multi-ciclos descrito no capítulo 5 do livro texto de Patterson e Hennessy.

O projeto será desenvolvido usando-se a ferramenta Quartus da Altera instalada nos laboratórios 1 e 2 do IC-3. Alternativamente você pode fazer download e instalar o Quartus em uma máquina pessoal (para acesso à página do Programa Educacional da Altera consulte a página do curso).

3 Descrição do Projeto

Implemente uma versão simplificada do *datapath* do processador MIPS multi-ciclos capaz de executar instruções **R-type**, como mostrado na figura abaixo. O *datapath* a ser implementado é um subconjunto do *datapath* multi-ciclo completo desenvolvido na seção 5.4 e mostrado na figura 5.33 do livro texto.

Para simplificar o projeto iremos supor que a memória de instruções e de dados é externa ao nosso sistema, assim você não precisa modela-la (tarefa que não é trivial). Também iremos assumir que instruções e dados são de 32 bits. Não implementaremos, nesta versão simplificada,

o passo de *instruction fetch* e assumir que os valores a serem carregados em *Instruction Register* (IR) ou em *Memory Data Register* (MDR) estão disponíveis como entrada (sinais de entrada na entidade). Também iremos ignorar a parte relacionada com o PC, ou seja o calculo de PC + 4 para a busca de uma nova instrução.

Utilize como entidade base a entidade dada abaixo.

```
Entity R_datapath is
  generic(nbits : positive := 32);
  port(data     : in std_logic_vector(nbits -1 downto 0);
       clk      : in bit;
       reset    : in bit;
       Zero     : out bit;
       Overflow : out bit;
       ALUOut   : out std_logic_vector(nbits -1 downto 0));
End R_datapath;
```

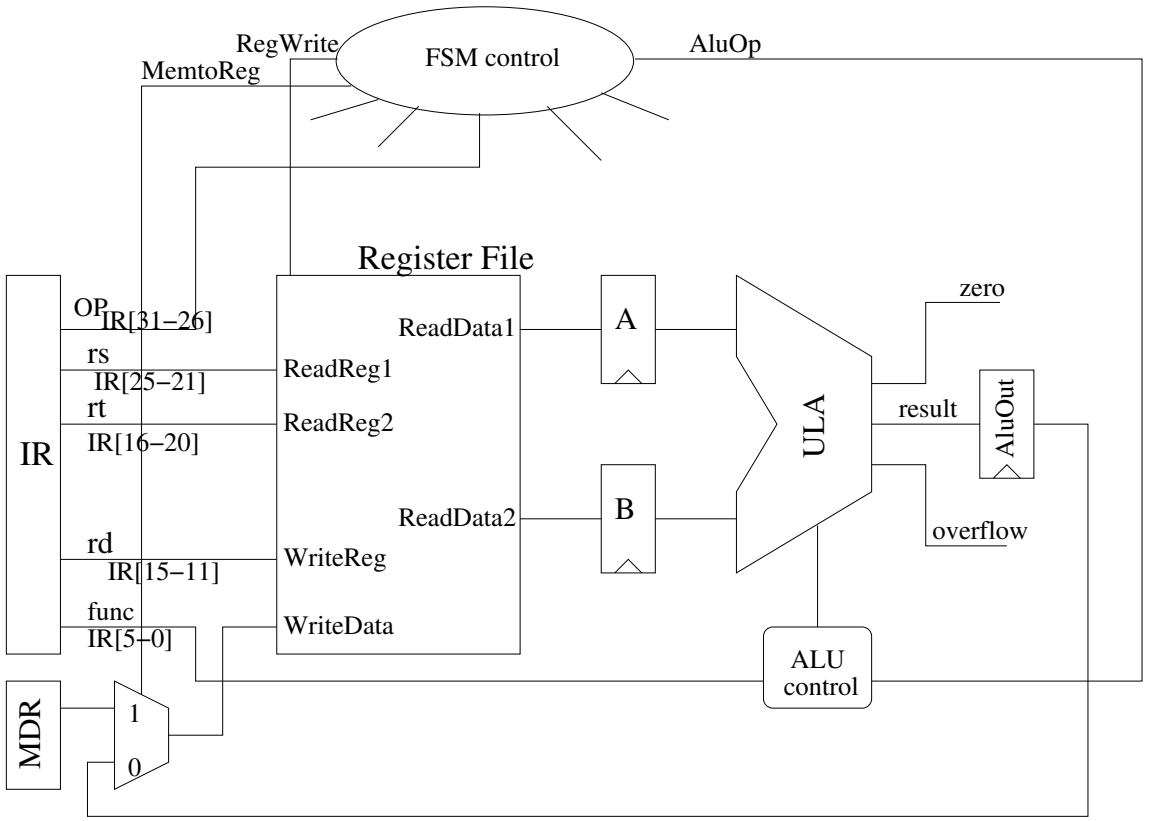

Figura 1: Datapth simplificado do MIPS para instruções R-type.

O projeto conterá um *Register File*, uma ULA, registradores auxiliares e as unidades de controle *ALU control* e *FSM control*. O *Register File* e a ULA são os componentes projetados

nos exercícios anteriores e as unidades de controle são as unidades responsáveis por gerarem os sinais de controle necessários à execução das instruções R-type.

4 Comportamento do Datapath

O *datapath* para instruções R-type deve ter o seguinte comportamento: cada fase é relacionada a um estado da *FSM control*, mostrada na figura 5.42 do livro texto. Como a nossa memória é externa o dado (instrução) estará disponível em *data* (as instruções serão editadas no arquivo de simulação) e deverão ser carregadas em IR, assim devemos partir do estado 0 e também devemos retornar ao estado 0 durante o ciclo de execução de uma instrução. Para mais detalhes consulte o livro texto.

(estado 0) Neste ciclo deverá ser guardado em IR o valor disponível na entrada *data* da entidade.

(estado 1) Neste ciclo de execução de uma instrução serão lidos os dois registradores *rs* e *rt* do *Registr File* endereçados por IR[25-21] e IR[20-16] respectivamente. Eles devem ser carregados nos registradores auxiliares A e B.

(estado 2) No próximo ciclo de execução a ALU deverá executar a operação indicada pela instrução usando como operandos os registradores A e B. ALUop = 10 e a *ALU control* usando o campo *function* IR[5-0] deverá gerar os 3 bits de controle da ULA apropriados.

(estado 3) Neste ciclo o conteúdo do registrador ALUout deve ser armazenado no registrador destino.

OBS.: Quem não implementar a instrução *load* pode usar o sinal de **reset** para além de colocar a FSM no **estado inicial** inicializar os (ou alguns) registadores do banco de registradores para realizar a simulação de instruções tipo R.

5 Concorra a Pontos Extras

Implemente extensões ao projeto básico acima, com intuito de aproxima-lo ao *datapath* multiciclo apresentado no livro texto (instruções na ULA operando com imediato, manipulação do PC, leitura de memória, instruções load, store e branch, ...). Caso seja necessário, devido às extensões implementadas, acrescente à entidade base os sinais de entrada e/ou saída necessários.