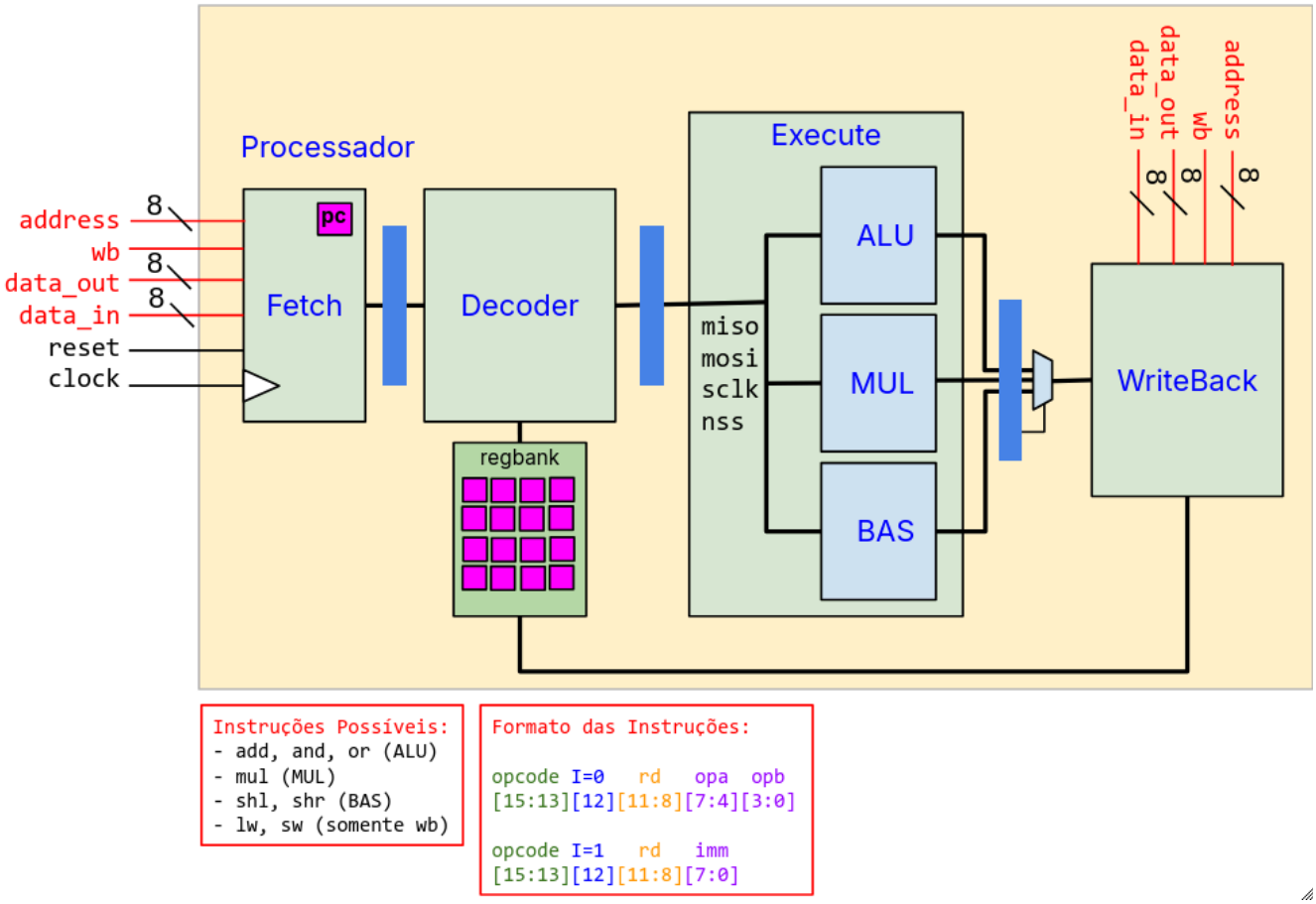
Trabalho T1 (2025-2) - Vale 30% da nota T

Objetivo: Familiarizar o aluno com o desenvolvimento de sistemas de hardware em nível de simulação RTL, utilização de ferramentas de simulação de indústria e projetos de testes básicos envolvendo testbenches e cobertura de código.

- Para este trabalho, **cada grupo de até 4 pessoas** deverá entregar somente o link para seu repositório de projeto (ex., Github, BitBucket, GitLab, AWS CodeCommit). Seu projeto deverá conter um arquivo README descrevendo os passos para executar sua simulação. Projetos sem README, que não estejam em repositório público ou que não executarem de acordo com as instruções, não serão avaliados.

- A data de envio é conforme mostrado na sala de entrega, no Moodle.

Descrição do Projeto: Considere a ilustração abaixo, que mostra os blocos de um processador de 4 estágios (sem pipeline!). Neste processador, cada instrução demora 4 ciclos para acontecer. Em cada ciclo, a instrução percorre um bloco diferente: FETCH, DECODE, EXECUTE e WRITE-BACK. No bloco EXECUTE, uma unidade lógica e aritmética (ALU), um multiplicador (MUL) e um *barrel shifter* (BAS) se conectam por um barramento SPI compartilhado, como mostra a imagem abaixo.



FETCH: Este bloco é responsável por ler uma instrução da memória. Para isso, utilizará os sinais mostrados em vermelho na imagem. O módulo de memória será fornecido pronto para uso. Ao ler uma instrução da memória, o bloco FETCH armazena a instrução em uma barreira temporal, mostrada como um retângulo azul na imagem. Esta barreira permite comunicar entre diferentes blocos ao longo da execução de uma instrução e é implementada na forma de registradores. A primeira barreira temporal se refere apenas ao registro da instrução que será utilizada pelo DECODER.

DECODE: Para cada instrução, o bloco DECODE deverá buscar o valores dos registradores envolvidos na instrução e armazená-los na próxima barreira temporal. Também deverá armazenar o opcode da instrução, para que o bloco de execução possa executar a instrução corretamente.

EXECUTE: Este bloco possuirá uma máquina de estados que lê os operandos e opcode da barreira temporal e os envia para o bloco de execução correto. O projeto prevê 3 blocos de execução: ALU, MUL e BAS. Se você realizou o exercício de SPI trabalhado nas aulas 7, 8, 10 e 11, poderá utilizar a mesma ALU e interface SPI. Se você realizou o exercício da aula 6, poderá utilizar o multiplicador que construiu. Portanto, o único bloco que resta construir é o barrel-shifter (BAS), utilizado nas instruções SHL e SHR. Mais informações sobre o BAS podem ser encontradas aqui: https://en.wikipedia.org/wiki/Barrel_shifter

WRITE BACK: Este módulo permite escrever o resultado das instruções nos registradores do processador ou na memória. Por isso, possui uma interface com a memória, assim como o bloco FETCH, mostrado em vermelho na imagem. Também é conectado ao banco de registradores.

Enunciado: Você e seu grupo deverão entregar uma implementação do processador descrito acima, utilizando de partes dos projetos já trabalhados em aula. Com exceção das barreiras temporais e da interface com a memória, todos os outros componentes foram trabalhados em aula ou estão referenciados no material.

- Caso nenhum aluno do grupo tenha realizado os exercícios propostos durante as aulas, poder-se-á utilizar os blocos de regbank e ALU disponíveis neste repositório, requerendo minimas alterações: https://github.com/andersondomingues/rre/tree/master
- O bloco de memória também está disponível neste repositório: https://github.com/andersondomingues/rre/tree/master/cores/memory

Entregáveis e Pontuação:

- ***Entregar apenas o link do repositório***, onde deve constrar arquivo README.md identificando os componentes do grupo, código-fonte e scripts de execução. O arquivo README.md deverá conter também *prints* de tela mostrando a execução de cada instrução em forma de onda.
- Cada instrução funcionando corretamente credita +1 ponto na nota final, para um total de 8 pontos.
- Os 2 pontos remanescentes serão distribuídos de acordo com a qualidade da implementação e organização do projeto e repositório.
- Projetos que não utilizarem o protocolo SPI na comunicação dos blocos de execução receberão **nota zero**. O mesmo vale para projetos que não implementarem as barreiras temporais.