UNIVERSIDADE DE BRASÍLIA
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

**116394 ORGANIZAÇÃO E ARQUITETURA DE COMPUTADORES**

Projeto da Disciplina: MIPS Uniciclo em VHDL

## OBJETIVO

O projeto da disciplina consiste no desenvolvimento de uma versão do processador MIPS Uniciclo em FPGA, utilizando a linguagem VHDL.

A plataforma de desenvolvimento é Altera. Podem ser utilizados os kits de desenvolvimento DE2-35 ou DE2-70, disponíveis no laboratório de hardware do CIC.

As ferramentas utilizadas para o desenvolvimento serão o Quartus II e ModelSim-Altera.

## DESCRIÇÃO

O diagrama esquemático da arquitetura do MIPS Uniciclo é apresentado na figura 1.

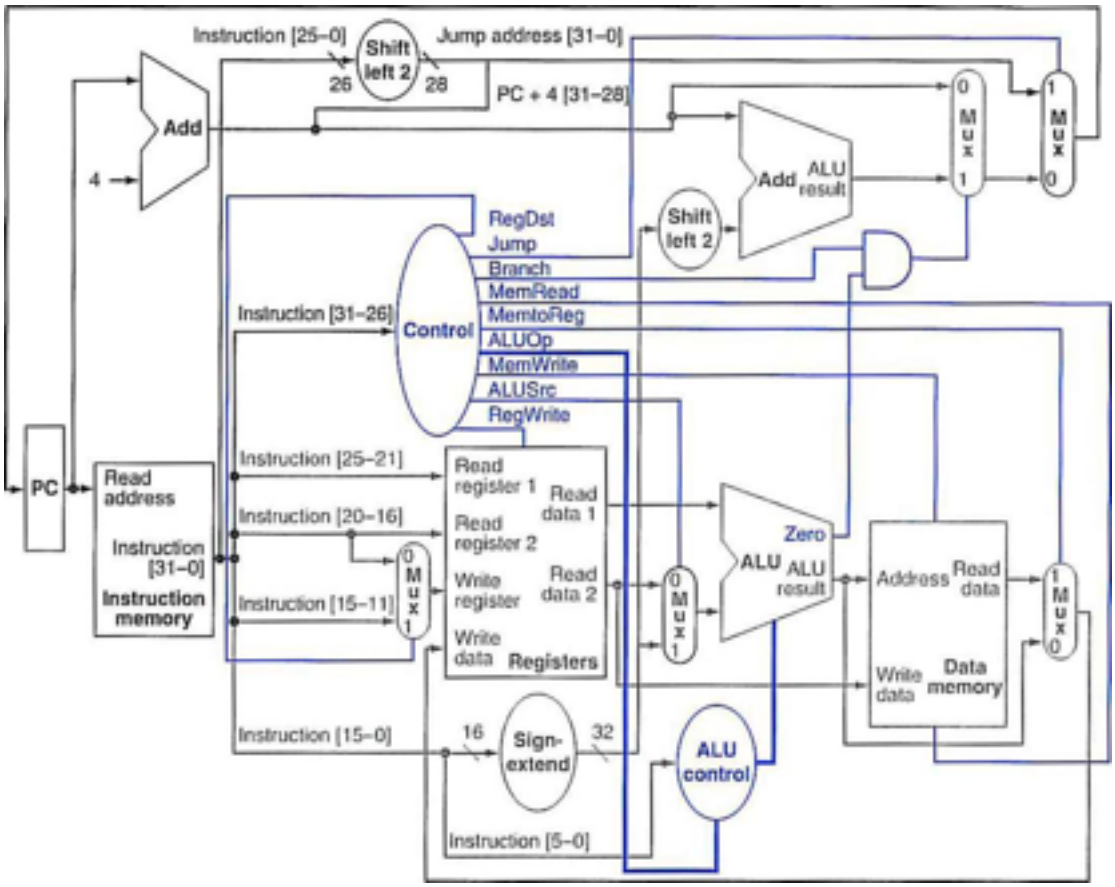


Figura 1. MIPS Uniciclo.

A implementação VHDL consiste na descrição de cada módulo e sua interligação através de sinais.

A parte operativa do MIPS é 32 bits, ou seja, os dados armazenados em memória, os registradores, as instruções e as conexões utilizam 32 bits.

Devido às restrições de memória das placas FPGA, as memórias de instruções e dados devem ser dimensionadas para suportas pequenos programas de teste. Sugere-se utilizar apenas 8 bits de endereço, de forma a prover um espaço de endereçamento de 256 palavras tanto para o programa quanto para os dados.

**Módulos principais**:

- **PC**: contador de programa. É um registrador de 32 bits. Entretanto, pelas restrições de memória adotadas, apenas o número de bits necessário deve ser enviado à memória de instruções, sendo o restante ignorado;
- **Memória de Instruções (MI)**: armazena o código a ser executado. As instruções são de 32 bits. O espaço de endereçamento é reduzido (ex: 8 bits). Cada endereço da MI armazena uma instrução de 32 bits. Idealmente, a MI deve funcionar como um bloco combinacional neste projeto, ou seja, necessita-se apenas do endereço para ler a instrução, sem sinais adicionais de controle. Essa memória não permite o endereçamento a byte. Desta forma, se forem utilizados 8 bits de endereço, deve-se utilizar os bits 2 a 9 do PC como endereço de instrução;
- **Banco de Registradores (BREG)**: é constituído por 32 registradores de 32 bits. O registrador índice zero, BREG[0], é uma constante. Sua leitura retorna sempre zero, e não pode ser escrito. O BREG tem duas entradas de endereços, permitindo a leitura de 2 registradores de forma simultânea. Uma terceira entrada de endereço é utilizada para selecionar um registrador para escrita de dados. A escrita de um dado em registrador ocorre na transição de subida do relógio.
- **Unidade Lógico-Aritmética (ULA)**: opera sobre dados de 32 bits. Provê o resultado em 32 bits, juntamente com o sinal **ZERO**, que indica que o resultado da operação realizada é zero.
    - Operações implementadas na ULA:
        - ADD, SUB, AND, OR, XOR, SLT, NOR, SLL, SRL, SRA
        - Obs: para as instruções de deslocamento, deve-se utilizar as funções de deslocamento disponíveis no pacote *numeric_std* : *shift_left* e *shift_right*, com o tipo apropriado de dados para o deslocamento lógico/aritmético.
- **Memória de Dados (MD)**: armazena os dados do programa. Pode ser lida ou escrita. Neste projeto, a MD fornece apenas palavras de 32 bits quando lida. Considerando que a MD é reduzida, deve-se selecionar apenas o número necessário de bits de endereço (8 bits, se a memória comportar 256 palavras). Os bits de endereço selecionados para acessar a MD devem permitir a leitura do segmento de dados conforme o modelo de memória compacto do MARS, onde o endereço base é 0x00002000. A memória é escrita na subida do relógio, quando o sinal de controle EscreveMem estiver acionado. O sinal de leitura LerMem faz com que o conteúdo da posição de memória endereçada seja colocado na saída de dados;
- **Multiplexadores 2 para 1**: são utilizados 4 multiplexadores com 2 entradas de 32 bits e uma saída de 32 bits;

- **Somadores**: são utilizados 2 somadores de 32 bits para operar com endereços;

## INSTRUÇÕES A SEREM IMPLEMENTADAS

- LW, SW, ADD, SUB, AND, OR, NOR, XOR, SLT, ADDI, SLL, SRL, SRA, J, BEQ, BNE

## VERIFICAÇÃO DO MIPS UNICICLO

O processador implementado deve ser capaz de executar um código gerado pelo MARS, utilizando o modelo de memória compacto.

## ENTREGA E APRESENTAÇÃO

A implementação em FPGA deverá ser apresentada em data a ser combinada com o professor.

Entregas:
- Código VHDL do projeto
- Relatório descrevendo a implementação

Entregar no Moodle em um arquivo compactado.

Prazo de entrega: 06/02/2017