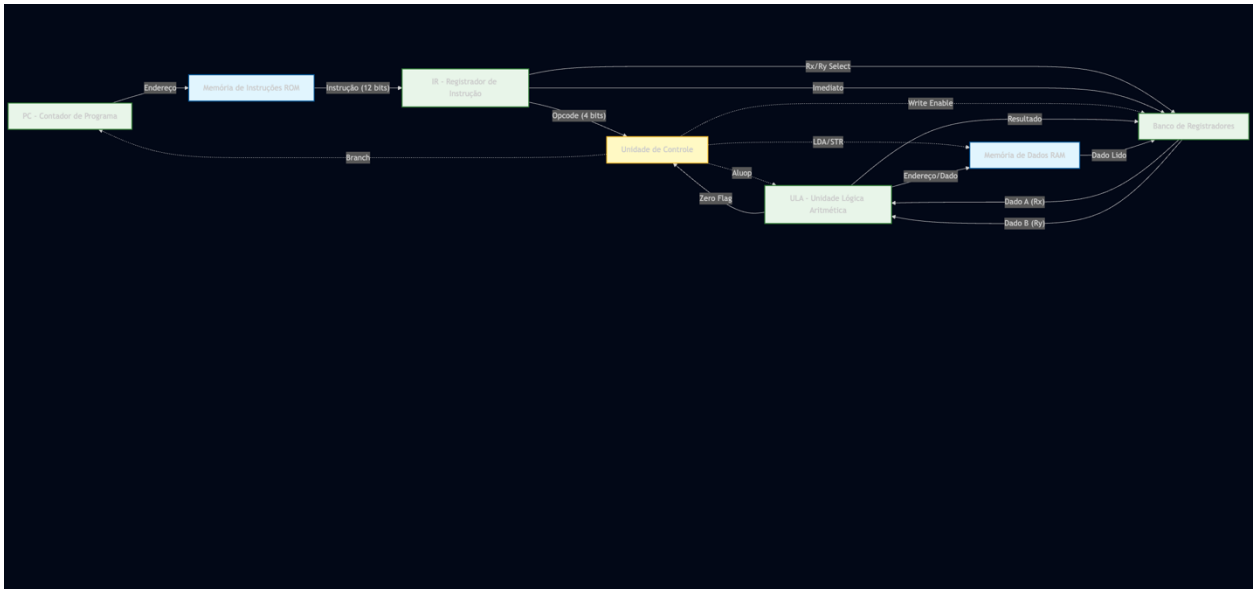
1. Diagrama de Blocos da Arquitetura Desenvolvida



2. Descrição Detalhada dos Módulos e Funcionamento

Esta parte detalha os componentes internos da arquitetura MIGS, descrevendo suas entradas, saídas e comportamento lógico conforme implementado no circuito feito no logisin.

2.1. Unidade de Controle (UC)

O módulo de controle atua como o decodificador central da CPU. Ele recebe a instrução de 12 bits proveniente do registrador de instrução (IR) e a decompõe para gerenciar o fluxo de dados.

- Entradas: Barramento de Instrução (12 bits).
- Saídas de Controle: Gera sinais discretos para as operações LDA (Load), LDI (Load Immediate), STR (Store), MOV (Move), JMP (Jump) e HLT (Halt). Além disso, emite o sinal Aluop (4 bits) que configura a operação da ULA.
- Lógica de Decodificação: Utiliza *splitters* para separar os 4 bits mais significativos (Opcode) dos operandos. Dependendo do opcode, a UC roteia os

8 bits menos significativos como um valor imediato (LDI Value) ou como endereços de registradores (#Rx, #Ry).

2.2. Unidade Lógica e Aritmética (ULA)

A ULA é responsável por todo o processamento matemático e lógico. Diferentemente de ULAs simplificadas, este projeto implementa um conjunto robusto de operações aritméticas dedicadas.

- Arquitetura Interna: O circuito da ULA contém blocos de hardware independentes para cada operação, cujos resultados são conectados a um Multiplexador de saída controlado pelo sinal Aluop.
- Operações Suportadas:
    - Aritmética: Soma (Adder) , Subtração (Subtractor) , Multiplicação (Multiplier) e Divisão (Divider).
    - Lógica: Portas AND e OR , além de um inversor (Negator).
    - Comparação: Módulo Comparador para avaliações relacionais.
- Sinais de Estado: A ULA gera um sinal de saída Zero, essencial para a lógica de desvios condicionais.

2.3. Banco de Registradores (Register File)

O banco de registradores fornece armazenamento temporário de alta velocidade para operandos da ULA.

- Capacidade: Composto por registradores de 8 bits, construídos internamente a partir de Flip-Flops tipo D.
- Portas de Acesso: O design implementa uma arquitetura de leitura dupla e escrita única:
    - Leitura: Possui dois multiplexadores de saída que permitem ler simultaneamente os valores dos registradores selecionados pelos endereços Rx Select e Ry Select.
    - Escrita: Um demultiplexador roteia o sinal de write e os dados de entrada (Data IN) para o registrador de destino selecionado (write select).

2.4. Contador de Programa (PC) e Memória

- PC: Registrador que armazena o endereço da próxima instrução. Possui lógica para incremento sequencial ou carga paralela, permitindo a execução de saltos (JMP) quando o endereço de destino é fornecido pela instrução.

- Memória de Instruções (ROM): Armazena o firmware/código. Recebe o endereço do PC e entrega uma palavra de 12 bits ao IR.
- Memória de Dados (RAM): Permite a leitura e escrita de dados de 8 bits, sendo endereçada diretamente pelo resultado da ULA ou registradores, conforme o sinal str (Store) ou ld (Load).

3. Decisões de Projeto e Justificativas Técnicas

Abaixo são apresentadas as decisões de arquitetura tomadas durante o desenvolvimento do processador MIGS e suas respectivas justificativas técnicas.

3.1. Adoção de Instruções de 12-bits em Arquitetura de 8-bits

Embora o *datapath* (ULA, Registradores e RAM) opere com palavras de 8 bits, optou-se por utilizar instruções de largura fixa de 12 bits.

- Justificativa: Uma instrução de 8 bits seria insuficiente para codificar o *Opcode* e dois operandos de registradores simultaneamente (ex: ADD R1, R2). Com 12 bits, é possível alocar 4 bits para o Opcode (16 instruções possíveis), 4 bits para o registrador de destino e 4 bits para o registrador fonte (ou valor imediato curto). Isso simplifica drasticamente a Unidade de Controle, evitando a complexidade de instruções multi-byte (tamanho variável).

3.2. Arquitetura Harvard (Memórias Separadas)

O projeto utiliza blocos distintos para Memória de Instruções (ROM) e Memória de Dados (RAM).

- Justificativa: Esta separação elimina a disputa pelo barramento de memória durante o ciclo de busca (*fetch*) e execução. Permite que a CPU leia a próxima instrução e acesse dados da RAM no mesmo ciclo lógico (em uma implementação de pipeline futuro) ou simplifique o controle em ciclo único, evitando a necessidade de estados de espera (*wait states*) ou arbitragem de barramento complexa.

3.3. Implementação da ULA com Blocos Funcionais Paralelos

Ao invés de uma ULA sequencial baseada apenas em somadores completos, a ULA foi projetada com blocos funcionais (Multiplicador, Divisor, Portas Lógicas) operando em paralelo.

- Justificativa: O uso de componentes dedicados (como o Multiplier e Divider nativos da biblioteca Logisim) aumenta a precisão e a clareza do projeto. A seleção da operação via Multiplexador de saída garante que o sinal Resultado

seja estável e definido unicamente pelo código Aluop, facilitando a depuração e garantindo previsibilidade lógica.

3.4. Banco de Registradores com Leitura Dupla (Dual-Port Read)

O banco de registradores foi desenhado para expor duas portas de saída de dados (Rx e Ry) independentes.

- Justificativa: Isso é fundamental para permitir a execução de operações aritméticas binárias (que requerem dois operandos, como soma e subtração) em um único ciclo de clock. Sem essa funcionalidade, seriam necessários múltiplos ciclos para carregar os operandos na ULA sequencialmente, reduzindo significativamente o desempenho (IPC - Instruções por Ciclo).