Relatório

Implementação microarquitetural da versão monociclo de 32 bits para subconjunto da ISA RISC-V

Pedro Henrique Marques de Lima

28 de Julho de 2024

1 Visão Geral do Projeto

O objetivo deste projeto é desenvolver uma implementação microarquitetural de uma versão monociclo (single-cycle) de 32 bits para um subconjunto da ISA (Instruction Set Architecture) RISC-V. Esta implementação será capaz de executar instruções e operações básicas do sistema, utilizando até três registradores por instrução, além de permitir a manipulação de valores imediatos (constantes).

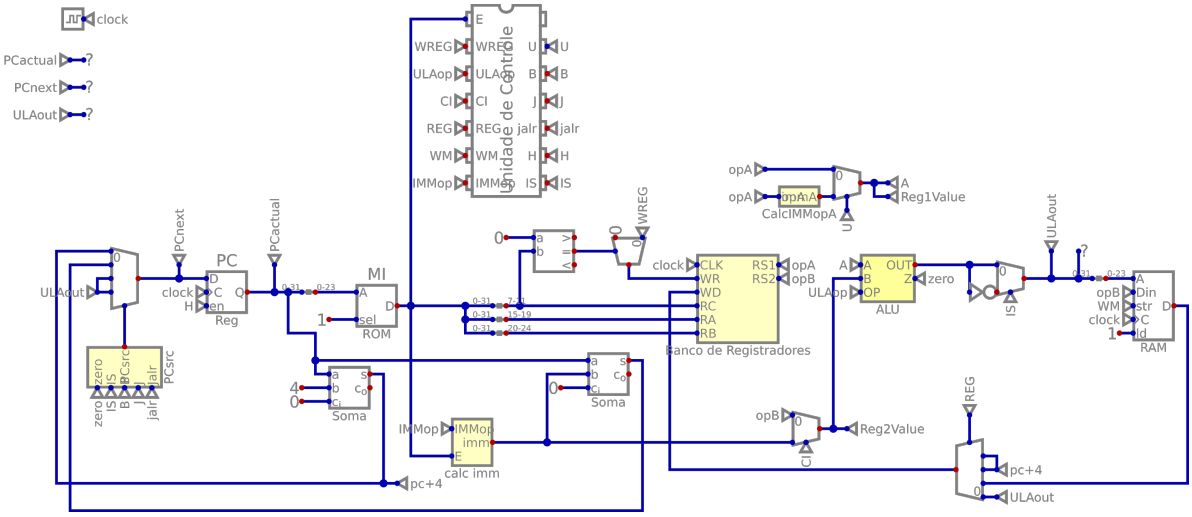

Figura 1: Circuito Geral do Processador

O projeto abrange sete formatos de instruções distintos, cada um com características e funcionalidades específicas:

- **Tipo R:** Instruções que operam diretamente com registradores, executando operações aritméticas e lógicas.
- **Tipo I:** Instruções que utilizam registradores e valores imediatos, incluindo operações de cálculo e manipulação de dados.
- **Tipo S:** Instruções responsáveis por armazenar dados na memória, transferindo valores dos registradores.
- **Tipo B:** Instruções condicionais que realizam saltos com base em comparações, permitindo a execução de blocos de código diferentes.
- **Tipo J:** Instruções de salto, que alteram o fluxo de execução do programa para um endereço específico.

- **Tipo U:** Instruções que lidam com valores imediatos grandes, principalmente utilizados para operações de carga de endereços.

Este projeto visa fornecer uma base para o estudo e compreensão da arquitetura RISC-V, focando na simplicidade e clareza da implementação de uma microarquitetura de ciclo único.

2 Unidade de Controle

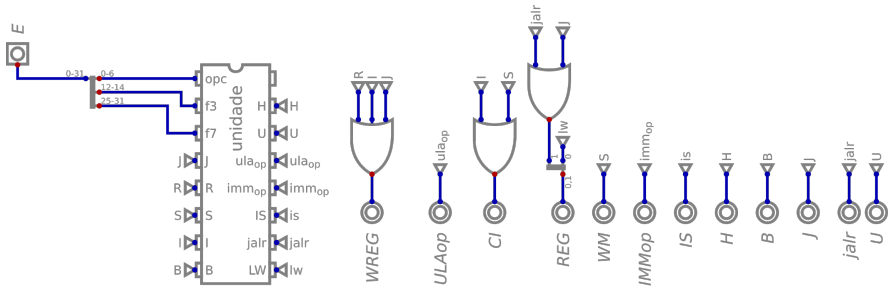

Figura 2: Circuito Geral da UC

A unidade de controle é responsável por interpretar as instruções e coordenar as operações no processador. Ela recebe como entrada uma instrução de 32 bits, da qual extrai o opcode, o funct3 e o funct7. Esses campos são cruciais para determinar a operação a ser realizada pelo processador.

As saídas da unidade de controle incluem onze sinais principais, descritos a seguir:

- **WREG (1 bit):** Indica se o valor será escrito em um registrador.
- **ULAop (3 bits):** Define qual operação será executada pela Unidade Lógica e Aritmética (ULA).
- **CI (1 bit):** Controla se a ULA utilizará um valor imediato ou um valor de registrador como entrada.
- **REG (2 bits):** Decide qual valor será armazenado no registrador: o valor lido da memória, o resultado de um salto de instrução, ou a saída da ULA.
- **WM (1 bit):** Indica se o valor será armazenado na memória.
- **IMMop (3 bits):** Define o tipo de imediato a ser utilizado na instrução.

- **IS (1 bit):** Determina se o sinal de saída será invertido ou não.
- **H (1 bit):** Sinal de halt, que interrompe a execução do programa, ativado pelo opcode da instrução de `ebreak`.
- **B (1 bit):** Sinaliza se a instrução é um branch.
- **J (1 bit):** Indica se a instrução é do tipo `jal`.
- **jalr (1 bit):** Indica se a instrução é do tipo `jalr`.
- **U (1 bit):** Identifica instruções do tipo U.

Esses sinais de controle são fundamentais para o funcionamento correto do processador, permitindo a coordenação entre as diferentes unidades funcionais e garantindo que cada instrução seja executada conforme especificado pela ISA RISC-V.

2.1 Camada Dois da Unidade de Controle

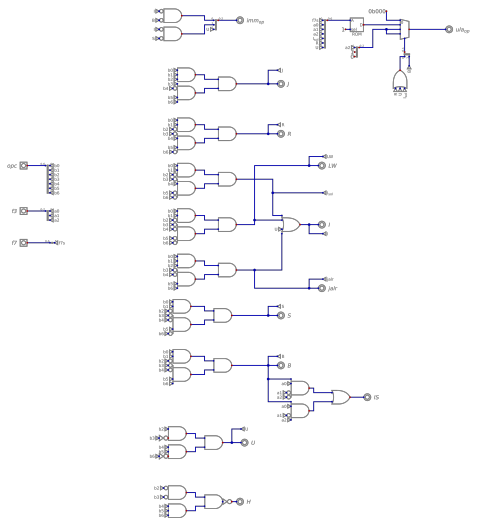

Figura 3: Circuito da camada dois da UC

Dentro da Unidade de Controle, há um segundo conjunto de funções responsáveis por identificar cada tipo de instrução e realizar a síntese apropriada. Para isso, os 7 bits do opcode foram analisados bit a bit, assim como os 3 bits do campo funct e o bit relevante do campo funct7 (bit 5), que é o único que varia, pois os demais são zeros em todas as instruções. Essa análise permite a identificação dos tipos de instrução: R, I, J, jalr, LW, S, B, e U, utilizando a detecção de sinais específicos.

Um identificador de sinal também foi utilizado para determinar as instruções que requerem inversão de valores. Por exemplo, tanto para BEQ quanto para BNE é usado o comparador de igualdade na ULA; contudo, para BNE, é necessário detectar desigualdade, o que exige a inversão do resultado. Assim, após identificar o tipo B, uma nova verificação é feita para determinar o `funct3` correspondente, e, se necessário, o sinal IS é ativado (valor 1) para realizar a inversão.

Após a identificação dos tipos de instrução, as saídas foram configuradas para selecionar o tipo de imediato a ser utilizado, de acordo com a tabela e o circuito especificados.

Para a saída da ULAop, foi implementada uma lógica de seleção que determina a operação a ser realizada. Instruções dos tipos S, LW, J, e `jalr` são mapeadas para a saída 000, que corresponde à operação de adição na ULA. Outras operações são separadas conforme os tipos R e I, com uma distinção adicional para instruções tipo B.

A lógica de seleção utiliza um multiplexador (MUX) para determinar a saída correta. Quando o seletor é 00, a saída é 000; quando é 01, uma memória ROM é acessada, onde endereços de 5 bits, representados em hexadecimal, armazenam a saída da ULA de 3 bits, também em hexadecimal.

No caso de instruções tipo B, o seletor 10 utiliza uma tabela-verdade para identificar a operação, baseando-se no bit mais significativo, enquanto outras saídas permanecem constantes.

Por fim, todas as saídas necessárias são conectadas às entradas do MUX, garantindo que cada instrução seja tratada conforme o necessário.

3 Banco de Registradores

O bloco de Banco de Registradores foi feito manualmente com recursos do Digital, onde foram utilizados 31 registradores operados por meio de variáveis de controle. Todas as entradas dos registradores são controladas pelo Demux responsável pelo controle de escolha do respectivo registrador que irá ser escrito. Vale ressaltar que as saídas de tal Demux irão para RS1 e RS2, com o registrador x0 propositalmente forçado a resultar sempre na constante 0.

4 Unidade Lógica e Aritmética (ULA)

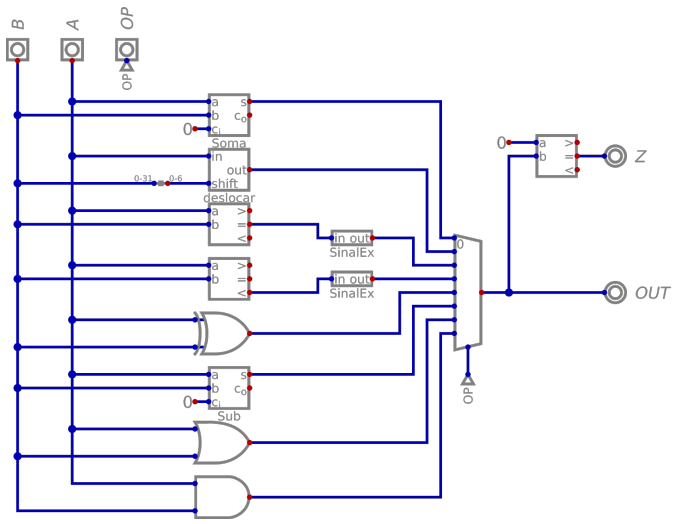

Figura 4: Circuito Geral da ULA

A Unidade Lógica e Aritmética (ULA) é responsável por realizar operações fundamentais no processador. Foram implementadas oito operações básicas: soma, deslocamento (shift), comparação de igualdade, comparação de menor que, XOR, subtração, AND e OR.

A ULA recebe como entradas dois operandos, RS1 e RS2, e o código da operação (ULAop), que atua como seletor em um multiplexador (MUX) para escolher a operação correta a ser executada. A ULA possui duas saídas: um sinal de 1 bit chamado "zero", que é utilizado para o controle de fluxo (PCsrc), e um resultado de 32 bits (R), que contém o resultado da operação.

O seletor para cada operação é determinado pelo campo `funct3`, o que simplifica o projeto do circuito e facilita a identificação da operação a ser realizada pela ULA.

5 Cálculo de Imediato

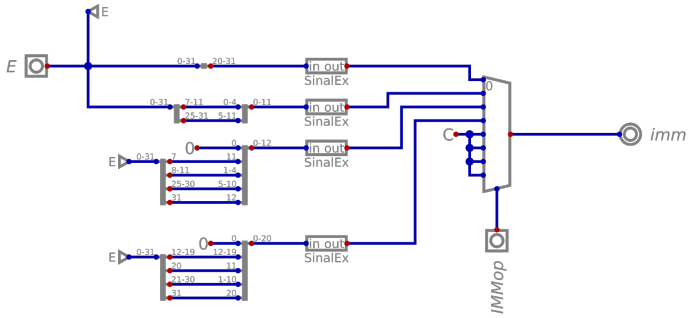

Figura 5: Circuito Geral do cálculo de imediato

Uma parte crucial do processador é o módulo de cálculo de imediato, responsável por extrair e preparar os valores imediatos das instruções. Este módulo recebe como entradas os 32 bits da instrução e o sinal IMMop, que seleciona o tipo de imediato a ser utilizado.

Os valores imediatos são determinados extraindo bits específicos conforme descrito na tabela de formatos de instrução para cada tipo de imediato. Para manipular esses valores, o módulo utiliza distribuidores e extensores de sinal. Esses mecanismos garantem que os valores imediatos sejam corretamente alinhados e sinalizados para uso nas operações subsequentes do processador.

5.1 Cálculo de imediato A

Uma subparte do cálculo do imediato ocorre na instrução do Tipo U, essa a qual destina uma constate 12 para o registrador B (rs2) e torna o registrador A (rs1) como o valor imediato. Esse cálculo só ocorre quando a variável de controle U é ativada.

6 PC

O contador de programa (PC) é responsável por determinar o endereço da próxima instrução a ser executada. Em casos de instruções do tipo branch, `jal` e `jalr`, é necessário que o endereço de salto seja calculado de forma diferente do incremento padrão de PC + 4, para direcionar o fluxo do programa corretamente.

O seletor PCsrc é utilizado para decidir o próximo endereço do PC com base no tipo de instrução:

- Quando PCsrc é 00, o PC é incrementado normalmente em 4, apontando para a próxima instrução sequencial (PC + 4).
- Quando PCsrc é 01, o PC é atualizado para PC + `imm`, indicando um salto, como em instruções do tipo J ou em branches condicionais do tipo B.
- Quando PCsrc é 10, o PC é atualizado para `rs1` + `imm`, que é o endereço calculado para instruções do tipo `jalr`.

Especial atenção é necessária para as instruções de branch. A decisão de salto depende dos sinais IS e Z (resultado da comparação na ULA):

- Se IS = 0 e Z = 0, o salto ocorre para PC + `imm`, pois a condição do branch foi satisfeita.
- Se Z = 1, a condição foi falsa, e o fluxo continua para a próxima instrução (PC + 4).
- Se IS = 1 e Z = 1, o sinal de saída da ULA é invertido, indicando que a condição foi verdadeira, então o PC é atualizado para PC + `imm`.
- Se Z = 0, a condição foi falsa, e o PC é incrementado normalmente (PC + 4).

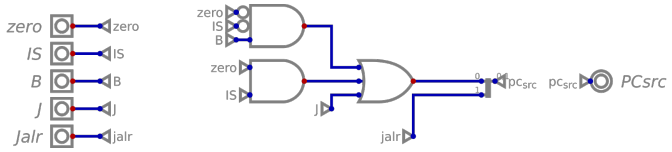

Figura 6: Circuito Geral do cálculo do PCsrc

7 Halt

O sinal de Halt é utilizado para interromper a execução do programa. Esse sinal é gerado com base no opcode da instrução `ebreak`. Quando o sinal H é ativado (H = 0), ele desativa o enable do contador de programa (PC), impedindo que qualquer outra instrução seja executada. Essa funcionalidade é essencial para finalizar o ciclo de execução de programas de forma controlada e segura.

8 Conclusão

Neste relatório, apresentamos a implementação de uma microarquitetura de processador de 32 bits baseada em um subconjunto da ISA RISC-V. A arquitetura desenvolvida inclui componentes essenciais como a Unidade de Controle, a Unidade Lógica e Aritmética (ULA), e módulos para o cálculo de imediatos e controle de fluxo (PC). Cada módulo foi detalhadamente descrito, destacando suas funções e interações dentro do processador.

A Unidade de Controle é crucial para a operação do processador, coordenando a execução de instruções através de sinais de controle específicos. A ULA, com suas múltiplas operações aritméticas e lógicas, é fundamental para o processamento de dados. O módulo de cálculo de imediato e o controle de fluxo (PC) garantem que o processador interprete corretamente os dados e siga o fluxo de instruções previsto, inclusive em casos de saltos condicionais e incondicionais.

O sistema também inclui mecanismos para interrupção segura do processamento, através do sinal de Halt, garantindo que o processador possa ser interrompido de forma controlada.

A implementação e análise desta microarquitetura proporcionam uma compreensão dos componentes e operações de um processador moderno, e reiteram a aplicabilidade da arquitetura RISC-V para projetos de hardware educacionais e industriais.