Implementação de Arquitetura RV32I Superescalar com Despacho Múltiplo Estático

Antônio Henrique Gonçalves Pereira, Millena Suiani Costa, Willian Paulo Pioker de Souza
Departamento de Informática
Universidade Federal do Paraná – UFPR
Curitiba, Brasil

***Resumo*—O presente trabalho consiste na implementação de uma arquitetura RV32I com despacho múltiplo superescalar no *software* logisim-evolution, com abordagem direta a previsão de desvios.**

***Index Terms*—RISC-V, RV32I, despacho múltiplo, superescalar, preditor de desvio.**

I. Introdução

A arquitetura *Reduced Instruction Set Computer - Five (RISC-V)* tem ganhado notoriedade por se tratar de uma alternativa simplificada e de fácil compreesão se comparada à arquiteturas diversas do mercado, além de integrar o nicho de projetos *open source*, fato que permite a colaboração entre desenvolvedores para a sua utilização, modificação, modularização e distribuição irrestrita.

Pautando-se na importância dessa, foi determinada a sua implementação simplificada como a temática central do trabalho, visando uma melhor compreensão a respeito de seu funcionamento e a possibilidade de experiência prática com o desenvolvimento de seus componentes e aplicação de testes integrados.

Além do desenvolvimento da arquitetura em si, foi aderida a proposta trazida no enunciado sobre acoplar nessa um módulo de despacho múltiplo – que se trata da execução simultânea de diferentes instruções – com fins de despertar o entendimento em torno do seu funcionamento.

II. Esqueleto do Processador

A primeira etapa de implementação do processador foi a construção do *datapath* principal com base em arquiteturas *RISC-V* disponíveis para visualização e no material [1], utilizando-se do padrão *RV32I* – que define a aplicação de instruções com 32 *bits* – por ser mais similar aos processadores *MIPS* abordados em sala de aula e desenvolvendo-o no *software logisim-evolution* por proporcionar melhores mecanismos de visualização que as suas versões anteriores.

Após finalizada a base de seu circuito principal, iniciou-se o desenvolvimento dos subcircuitos iniciais, conforme abordados nas seguintes subsessões.

A. Banco de Registradores

O banco de registradores para o padrão *RV32I* conta com 32 registradores de 32 *bits* cada, inicialmente dispondo de cinco entradas de dados, sendo elas a que carrega o valor disponível para a escrita, três representando os endereços dos registradores (tanto de leitura quanto de escrita) e a última representando o sinal lógico de *Write Enable (WE)*.

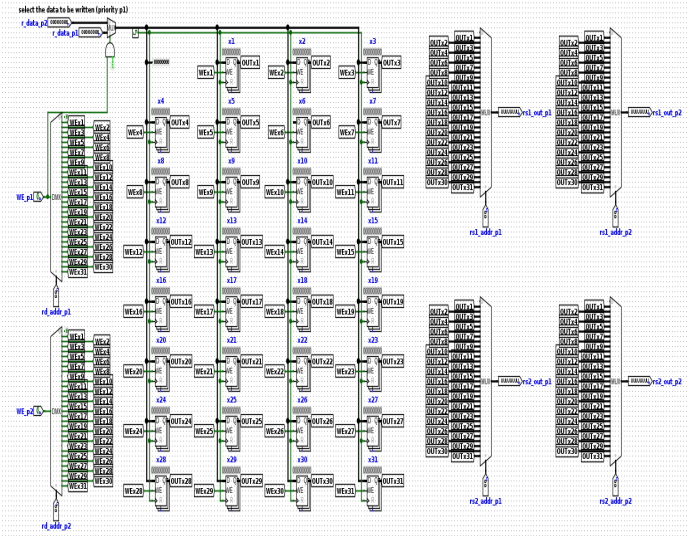

Figura 1. Banco de registradores para arquitetura RV32I

B. Arithmetic Logic Unit (ALU)

Para a construção da ALU foram escolhidos, previamente, os tipos de instrução a serem trazidos e quais delas seriam operações de lógica aritmética pautando-se no RISC-V Greencard [2].

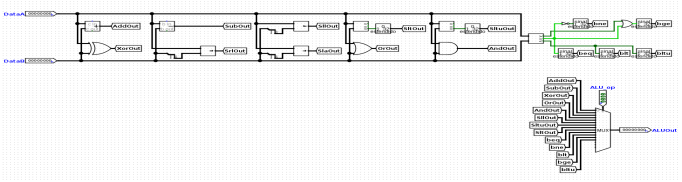

Figura 2. ALU simplificada para arquitetura RV32I

Se tratando de uma versão simplificada do RV32I, optou-se pela aplicação das instruções de tipo R, I, S e SB e, em seguida, feita a tabela I com cada uma das selecionadas e suas respectivas divisões de bits.

Tabela I
TABELA DE INSTRUÇÕES

	Divisão de bits			
Instrução	Tipo	Opcode	Funct3	Funct7
add	R	0110011	000	0000000
sub	R	0110011	000	0100000
sll	R	0110011	001	0000000
slt	R	0110011	010	0000000
sltu	R	0110011	011	0000000
xor	R	0110011	100	0000000
or	R	0110011	110	0000000
and	R	0110011	111	0000000
lw	I	0010011	010	x
addi	I	0010011	000	x
slti	I	0010011	010	x
sw	S	0100011	010	x
beq	SB	1100011	000	x
bne	SB	1100011	001	x
blt	SB	1100011	100	x
bge	SB	1100011	101	x
bltu	SB	1100011	110	x
bgeu	SB	1100011	111	x

C. Splitters e Instruction Decoder

Considerando a necessidade de lidar com quatro tipos diferentes de instrução – R, I, S e SB, como mencionados anteriormente – foi imprescindível a construção de subcircuitos especificos para a captação e separação de seus *bits* individualmente.

Os subcircuitos denominados *Splitters* separam e retornam cada tipo de *bits* por instrução nomeados por campo, visando melhor visualização. Esses, por sua vez, são utilizados dentro do subcircuito denominado *Instruction Decoder*, que recebe os 32 *bits* de instrução e, com base nos três primeiros *bits* de seu *opcode*, verifica o tipo da instrução e retorna os bits separados pelo respectivo *Splitter* dessa.

D. Branch Predictor

Para lidar com instruções com *branches* foi desenvolvida a *Branch Prediction Unit* de forma que recebe como entradas os *bits* de instrução que identificam se o formato da expressão corresponde a um desvio e os valores de endereço para o cálculo do próximo, no caso dessa correspondência ser verdadeira.

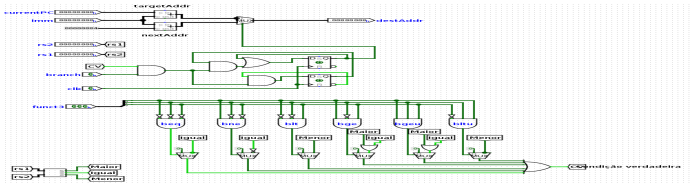

Figura 3. Branch Prediction Unit para arquitetura RV32I

E. Control Unit

Com o circuito principal estruturado foi possível tomar conhecimento de todos os sinais de controle necessarios para a correta operação desse, assim, o último dos subcircuitos desenvolvidos foi a Unidade de Controle, visando analisar cada uma das instruções passadas como parâmetro e, com base em seu formato, propagar os *bits* de controle para o seu respectivo *pipeline*.

III. POSSÍVEIS MELHORIAS

Algumas melhorias de conhecimento dos autores são possíveis para o presente circuito. A primeira delas é a implementação de uma *Forward Unit* para o aumento do desempenho do processador e tornar os seus *pipelines* mais eficientes.

A segunda melhoria cogitada foi aprimorar o tratamento de dependências entre as instruções. Em seu estado atual, por exemplo, são priorizadas as escritas em registrador para o *pipeline* focado em operações aritméticas, ou seja, caso o *pipeline* de acesso a memória requisitar uma escrita ao mesmo tempo, essa não será atendida. Para futuras melhorias, esse tratamento certamente se faria prioridade.

A última melhoria se trata de torná-lo um processador de despacho múltiplo dinâmico para que, no lugar da atribuição da tarefa de manipulação de operações que o despacho múltiplo estático realiza ao compilador, haja uma independência da CPU, fazendo com que apresente bom comportamento para quaisquer conjuntos de instruções.

IV. CONSIDERAÇÕES FINAIS

A conclusão do presente trabalho propiciou o entendimento da estrutura e funcionamento de processadores superescalares, além do desenvolvimento prático de um em formato de despacho estático considerando subcircuitos, formatos de instrução e sinais de controle.

REFERÊNCIAS

[1] "Guia prático risc-v: Atlas de uma arquitetura aberta," urlhttp://riscvbook.com/portuguese/, March 2019.

[2] "Risc-v reference data card ("green card")," urlhttps://dejazzer.com/coen2710/lectures/RISC-V-Reference-Data-Green-Card.pdf, 2021.