

Universidade Federal de Pernambuco
(UFPE)
Centro de Informática (CIn)

Relatório de Projeto

Aluno:
Gabriel Soares (gss12)

Data de Entrega: 21 de fevereiro de 2025

Relatório de Projeto: Implementação de um Processador RISC-V em Pipeline

22 de fevereiro de 2025

Sumário

| | | |
|----------|---|-----------|
| 1 | Introdução | 2 |
| 2 | Metodologia | 2 |
| 3 | Implementações | 2 |
| 3.1 | Instruções Registrador - Registrador | 2 |
| 3.2 | Instruções Registrador - Imediato | 3 |
| 3.3 | Instruções de Transferência de Controle | 3 |
| 3.4 | Instruções de Load e Store | 4 |
| 3.5 | Pseudo-Instrução | 5 |
| 4 | Simulações e Validações | 5 |
| 4.1 | Simulações de Referência | 5 |
| 4.2 | Simulações de LOAD | 6 |
| 4.3 | Simulações de Branch | 8 |
| 4.4 | Simulações de STORE | 12 |
| 4.5 | HALT | 13 |
| 5 | Conclusão | 14 |

1 Introdução

Este relatório descreve o desenvolvimento e validação de uma implementação em pipeline de um processador baseado no conjunto de instruções RISC-V. O objetivo do projeto foi projetar e testar um processador funcional que executa um subconjunto de instruções RISC-V, garantindo sua correta operação por meio de simulações no ModelSim.

2 Metodologia

A implementação foi realizada utilizando a linguagem de descrição de hardware `**SystemVerilog**`, com suporte a pipeline para execução eficiente das instruções. O projeto foi validado por meio de simulações no ModelSim, garantindo que as instruções implementadas fossem corretamente executadas.

3 Implementações

A implementação do processador incluiu suporte a diferentes tipos de instruções, categorizadas conforme feito no repositório do GitHub.

3.1 Instruções Registrador - Registrador

Para implementar as funcionalidades dessa categoria, foi necessário ajustar o controlador da ALU para considerar as partes do OPcode específicas de cada instrução, como `funct3` e `funct7`, garantindo seu correto tratamento na ALU. Neste componente, a execução da operação correspondente à instrução é realizada com base no sinal gerado pelo controlador.

- SUB

Considere o caso em que a operação é (0110) e definimos que a saída da ALU seria $ALUResult = SrcA - SrcB$.

- SLT

Considere o caso em que a operação é (1100) e dividimos a saída em três situações:

Quando o primeiro registrador possui um valor negativo e o segundo um valor positivo, o primeiro sempre será menor, portanto $ALUResult = 1$; Quando o primeiro registrador tem um valor positivo e o segundo um valor negativo, o primeiro nunca será menor que o segundo, logo $ALUResult = 0$; Para qualquer outra combinação diferente das duas anteriores, utilizei um operador ternário para verificar se $(SrcA < SrcB)$ e atribui o resultado a $ALUResult$.

- XOR

Considere o caso em que a operação é (0101) e definimos que a saída será $SrcA \oplus SrcB$.

- OR

Considere o caso em que a operação é (0001) e estabelecemos que a saída será $ALUResult = SrcA \vee SrcB$.

3.2 Instruções Registrador - Imediato

Para implementar a instrução que requer um imediato, modifiquei o Gerador de Imediato para reconhecer a instrução I-Type e o controlador para permitir o uso da ALU e da escrita do Registrador. Adicionei na ALU um novo caso para a instrução. No ALUController, usei o código da operação, a Funct3 e Funct7 de cada instrução para encaminhar para o caso da ALU correspondente à instrução. O registrador da instrução fica no fio "SrcA" e o imediato fica no fio "SrcB".

- ADDI
Considere o caso em que a operação é (0010) e defini que a saída da ALU será $ALUResult = SrcA + SrcB$.
- SLTI
Considere o caso em que a operação é (1100) e dividi a saída em três situações:
Quando o registrador possui um valor negativo e o imediato um valor positivo, o registrador será sempre menor, portanto, $ALUResult = 1$; Quando o registrador tem um valor positivo e o imediato é negativo, o registrador nunca será menor, logo, $ALUResult = 0$; Para qualquer outra configuração diferente das anteriores, utilizei um operador ternário que verifica se $(SrcA < SrcB)$ e atribui o resultado a $ALUResult$.
- SLLI
Considere o caso em que a operação é (0111) e utilizei o operador de shift lógico à esquerda do SystemVerilog, atribuindo o resultado a $ALUResult$ na ALU.
- SRLI
Considere o caso em que a operação é (1111) e apliquei o operador de shift lógico à direita do SystemVerilog, atribuindo o resultado a $ALUResult$ na ALU.
- SRAI
Considere o caso em que a operação é (1110) e empreguei o operador de shift aritmético à direita do SystemVerilog, atribuindo o resultado a $ALUResult$ na ALU.
- LUI
Considere o caso em que a operação é (1011), retornei o valor de SrcB, defini 0 nos últimos 12 bits e atribui o resultado a $ALUResult$ na ALU.

3.3 Instruções de Transferência de Controle

Para implementar as funções de desvio incondicional, o controller foi modificado para reconhecer as instruções correspondentes e enviar dois sinais ao data path: um indicando que se trata de um desvio incondicional e outro determinando que o resultado da busca nos registradores deve ser utilizado.

O sinal é processado na segunda etapa do pipeline e armazenado no banco B de registradores. Na terceira etapa, foi adicionado um multiplexador (mux) que seleciona o valor de sourceA com base nesse sinal. Esse valor, que será armazenado no registrador, pode ser tanto o conteúdo do registrador quanto o próprio $PC + 4$.

Além disso, a Branch Unit foi ajustada para considerar os valores obtidos dos registradores e os sinais das instruções do tipo J. No caso de um JALR, o próximo valor do PC é atualizado para o valor armazenado no registrador somado ao imediato (Imm).

- JAL
Quando o sinal jal está ativo, o valor de $PC + 4$ é armazenado no registrador, e o próximo PC é atualizado para $PC + Imm$.
- JALR
Quando os sinais jal e jalr estão ativos, o valor de $PC + 4$ é armazenado no registrador, e o próximo PC é atualizado para $FAmuxResult + Imm$.
- BNE
Considere o caso em que a operação é (1001) e utilizei o operador lógico `!=` do SystemVerilog para verificar se os valores armazenados nos registradores são diferentes.
- BLT
Usei o mesmo caso de operação para SLT e SLTI, uma vez que suas operações na ALU são equivalentes.
- BGE
Considere o caso em que a operação é (1010) e dividi a saída em três situações, de forma semelhante ao que foi feito para verificar se o valor de *SrcA* é menor que *SrcB*:

Quando o primeiro registrador tem um valor negativo e o segundo registrador tem um valor positivo, o primeiro registrador nunca será maior que o segundo, portanto, $ALUResult = 0$; Quando o primeiro registrador tem um valor positivo e o segundo registrador possui um valor negativo, o primeiro registrador sempre será maior que o segundo, logo, $ALUResult = 1$; Para qualquer outra configuração diferente das anteriores, utilizamos um operador ternário que verifica se $(SrcA \geq SrcB)$ e atribui o resultado a $ALUResult$.

3.4 Instruções de Load e Store

As funções de load envolvem a captura de diferentes números de bits. A escolha de implementação foi capturar a palavra completa (32 bits) da memória e resetar os bits que não são necessários, estendendo o bit de sinal (exceto no caso de lbu, que desconsidera o sinal). Dessa forma, apenas os bits necessários são escritos.

- LB
Leva em consideração o sinal e o byte menos significativo.
- LH
Leva em consideração o sinal e os 2 bytes menos significativos.
- LBU
Não leva em consideração o sinal e pega o byte menos significativo.

Para implementar os stores, alteramos o sinal "Wr" enviado para determinar qual byte será escrito na memória. No caso de uma operação half, desativamos o sinal de escrita para os dois bytes mais significativos, de forma que apenas os dois bytes menos significativos sejam gravados.

- SB
Grava o byte menos significativo da palavra fornecida na memória.

- SH
Grava os 2 bytes menos significativo da palavra fornecida na memória.

3.5 Pseudo-Instrução

- HALT
O halt foi implementado por meio de um novo módulo chamado HaltUnit. Esse módulo é responsável por controlar o opcode enviado ao controller, sendo que o opcode para o halt foi definido como 00000001. Quando um halt é detectado, a HaltUnit começa a transformar todos os opcodes enviados para o controller em 00000000, que não possui nenhuma função. Como o assembler não estava reconhecendo o halt, testei a modificação diretamente no arquivo instruction.mif.

4 Simulações e Validações

As simulações foram realizadas no ****ModelSim****, onde a execução das instruções foi analisada para verificar a correta operação do pipeline. Foram validadas todas as categorias de instruções, conforme detalhado abaixo:

4.1 Simulações de Referência

- ADD, OR, ADDI, SLL, SRL, SRA, SLT, SLTI, SLTIU, SLLI, SRLI, SRAI, XORI, ORI, ANDI, XOR

```
addi x0,x0,0
addi x1,x0,8
addi x2,x0,4
or x3,x1,x2
or x4,x2,x0
add x6,x4,x2
addi x4,x0,2
addi x5,x0,-2
sll x18,x1,x4
srl x19,x5,x4
sra x20,x5,x4
slt x21,x1,x2
sltu x22,x2,x1
sltu x23,x5,x1
sltu x24,x1,x5
slti x25,x1,8
slti x26,x1,16
addi x5,x0,-4
sltiu x27,x1,-2
sltiu x28,x5,-2
slli x29,x5,1
srl x30,x5,1
srai x31,x5,1
xori x6,x1,10
ori x7,x1,2
andi x8,x1,10
xor x9,x1,x2
```

Figura 1: Fornecido

```
45: Register [ 0] written with value: [00000000] [ 0]
55: Register [ 1] written with value: [00000008] [ 8]
65: Register [ 2] written with value: [00000004] [ 4]
75: Register [ 3] written with value: [00000000] [ 0]
85: Register [ 4] written with value: [00000004] [ 4]
95: Register [ 6] written with value: [00000000] [ 0]
105: Register [ 4] written with value: [00000002] [ 2]
115: Register [ 5] written with value: [fffffffe] [4294967294]
125: Register [18] written with value: [00000200] [ 32]
135: Register [10] written with value: [3fffffff] [1073741823]
145: Register [20] written with value: [fffffffe] [4294967294]
155: Register [21] written with value: [00000000] [ 0]
165: Register [22] written with value: [00000001] [ 1]
175: Register [23] written with value: [00000000] [ 0]
185: Register [24] written with value: [00000001] [ 1]
195: Register [25] written with value: [00000000] [ 0]
205: Register [26] written with value: [00000001] [ 1]
215: Register [ 5] written with value: [fffffffe] [4294967294]
225: Register [27] written with value: [00000001] [ 1]
235: Register [28] written with value: [00000001] [ 1]
245: Register [29] written with value: [fffffffe] [4294967294]
255: Register [30] written with value: [7fffffff] [2147483647]
265: Register [31] written with value: [fffffffe] [4294967294]
275: Register [ 6] written with value: [00000002] [ 2]
285: Register [ 7] written with value: [0000000a] [ 10]
295: Register [ 8] written with value: [00000000] [ 0]
305: Register [ 9] written with value: [0000000c] [ 12]
```

Figura 2: Esperado

```

1  addi x0,x0,0
2  addi x1,x0,8
3  addi x2,x0,4
4  or x3,x1,x2
5  or x4,x2,x0
6  add x6,x4,x2
7  addi x4,x0,2
8  addi x5,x0,-2
9  slt x21,x1,x2
10 slt x22,x2,x1
11 slti x25,x1,8
12 slti x26,x1,16
13 addi x5,x0,-4
14 slli x29,x5,1
15 srli x30,x5,1
16 srai x31,x5,1
17 xor x9,x1,x2
18

```

```

Time: 0 Instance: tb_top.riscV.dp.data_mem.mem32.memBlock3.altsyncram_component_m_default
45: Register [ 0] written with value: [00000000] | [ 0]
55: Register [ 1] written with value: [00000008] | [ 8]
65: Register [ 2] written with value: [00000004] | [ 4]
75: Register [ 3] written with value: [0000000c] | [ 12]
85: Register [ 4] written with value: [00000004] | [ 4]
95: Register [ 4] written with value: [00000000] | [ 0]
105: Register [ 4] written with value: [00000002] | [ 2]
115: Register [ 5] written with value: [fffffffc] | [4294967292]
125: Register [21] written with value: [00000008] | [ 8]
135: Register [22] written with value: [00000001] | [ 1]
145: Register [25] written with value: [00000000] | [ 0]
155: Register [26] written with value: [00000001] | [ 1]
165: Register [ 5] written with value: [fffffffc] | [4294967292]
175: Register [29] written with value: [fffffffc] | [4294967292]
185: Register [30] written with value: [fffffffc] | [4294967292]
195: Register [31] written with value: [fffffffc] | [4294967292]
205: Register [ 9] written with value: [00000000] | [ 0]

```

Figura 4: Resultado

Figura 3: Depois de tirar as instruções que não tem descrição

- SUB, AND, LUI

```

addi x1,x0,8
sub x6,x6,x1
and x7,x6,x1
lui x6,3

```

```

45: Register [ 1] written with value: [00000008] | [ 8]
55: Register [ 6] written with value: [ffffff8] | [4294967288]
65: Register [ 7] written with value: [00000008] | [ 8]
75: Register [ 6] written with value: [00003000] | [ 12288]

```

Figura 6: Esperado

Figura 5: Fornecido

```

# Time: 0 Instance: tb_top.riscV.dp.data_mem.mem32.memBlock3.altsyncram_component_m_default.altsyncram_inst
45: Register [ 1] written with value: [00000008] | [ 8]
55: Register [ 6] written with value: [ffffff8] | [4294967288]
65: Register [ 7] written with value: [00000008] | [ 8]
75: Register [ 6] written with value: [00003000] | [ 12288]

```

Figura 7: Obtido

4.2 Simulações de LOAD

- LB, LH, LW

```

addi x7,x0,1
addi x2,x0,4

or x4,x2,x0

lb x6,0(x7)

add x6,x4,x0

lb x7,0(x6)

lh x8,0(x6)

lw x9,0(x6)

```

```

45: Register [ 7] written with value: [00000001] | [ 1]
55: Register [ 2] written with value: [00000004] | [ 4]
65: Memory [ 1] read with value: [xxxxxxxx] | [ x]
65: Register [ 4] written with value: [00000004] | [ 4]
65: Memory [ 1] read with value: [ffffffff] | [4294967295]
70: Memory [ 1] read with value: [ffffff8f] | [4294967183]
75: Register [ 6] written with value: [ffffff8f] | [4294967183]
85: Register [ 6] written with value: [00000008] | [ 8]
85: Memory [ 8] read with value: [ffffff8f] | [4294967183]
85: Memory [ 8] read with value: [fffffffb] | [4294967291]
95: Register [ 7] written with value: [fffffffb] | [4294967291]
95: Memory [ 8] read with value: [fffffaaf] | [4294945531]
105: Register [ 8] written with value: [fffffaaf] | [4294945531]
105: Memory [ 8] read with value: [0001aafb] | [ 109307]
115: Register [ 9] written with value: [0001aafb] | [ 109307]

```

Figura 9: Esperado

Figura 8: Fornecido

```

# Time: 0 Instance: tb_top.riscv.op.data_mem.mem32.memBlock3.altysyncram_component_m_default.altysyncram_inst
45: Register [ 7] written with value: [00000001] | [ 1]
55: Register [ 2] written with value: [00000004] | [ 4]
65: Memory [ 1] read with value: [xxxxxxxx] | [ x]
65: Register [ 4] written with value: [00000004] | [ 4]
65: Memory [ 1] read with value: [00000003] | [ 3]
70: Memory [ 1] read with value: [00000000] | [ 0]
75: Register [ 6] written with value: [00000000] | [ 0]
85: Register [ 6] written with value: [00000004] | [ 4]
85: Memory [ 4] read with value: [00000000] | [ 0]
85: Memory [ 4] read with value: [00000003] | [ 3]
95: Register [ 7] written with value: [00000003] | [ 3]
105: Register [ 8] written with value: [00000003] | [ 3]
115: Register [ 9] written with value: [00000003] | [ 3]

```

Figura 10: Obtido

- LBU, LHU

```

addi x7,x0,1
addi x2,x0,4

or x4,x2,x0

lb x6,0(x7)

add x6,x4,x0

lbu x7,0(x6)

lhu x8,0(x6)

```

```

45: Register [ 7] written with value: [00000001] | [ 1]
45: Register [ 2] written with value: [00000004] | [ 4]
45: Memory [ 1] read with value: [xxxxxxxx] | [ x]
45: Register [ 4] written with value: [00000004] | [ 4]
45: Memory [ 1] read with value: [fffffffb] | [4294967291]
45: Register [ 6] written with value: [fffffffb] | [4294967291]
45: Register [ 6] written with value: [00000004] | [ 4]
45: Memory [ 4] read with value: [fffffffb] | [4294967291]
45: Memory [ 4] read with value: [00000003] | [ 3]
45: Register [ 7] written with value: [00000003] | [ 3]
45: Register [ 8] written with value: [00000003] | [ 3]
45: Register [ 9] written with value: [00000003] | [ 3]

```

Figura 12: Esperado

Figura 11: Fornecido


```

# Time: 0 Instance: tb_top.riscv_dp.data_mem.mem32.memBlock3.atsynsram_component.n_default.atsynsram_inst
#
# 45: Register [ 7] written with value: [00000001] | [ 1]
#
# 55: Register [ 2] written with value: [00000004] | [ 4]
# 65: Memory [ 1] read with value: [XXXXXXXX] | [ X]
#
# 65: Register [ 4] written with value: [00000004] | [ 4]
# 65: Memory [ 1] read with value: [00000003] | [ 3]
# 70: Memory [ 1] read with value: [00000000] | [ 0]
#
# 75: Register [ 6] written with value: [00000000] | [ 0]
# 85: Register [ 6] written with value: [00000004] | [ 4]
# 85: Memory [ 4] read with value: [00000000] | [ 0]
# 85: Memory [ 4] read with value: [00000003] | [ 3]
# 95: Register [ 7] written with value: [00000003] | [ 3]
# 105: Register [ 8] written with value: [00000003] | [ 3]

```

Figura 13: Obtido

4.3 Simulações de Branch

- JAL,BEQ (TAKEN)

```

addi x7,x0,1
addi x2,x0,4
jal x10,8
or x4,x2,x0
add x6,x4,x2
addi x7,x0,1
addi x8,x0,2
beq x7,x7,-8
or x4,x2,x0

```

```

45: Register [ 7] written with value: [00000001] | [ 1]
55: Register [ 2] written with value: [00000004] | [ 4]
65: Register [10] written with value: [0000000c] | [12]
85: Register [ 6] written with value: [00000004] | [ 4]
185: Register [ 7] written with value: [00000001] | [ 1]
115: Register [ 8] written with value: [00000002] | [ 2]
155: Register [ 7] written with value: [00000001] | [ 1]
165: Register [ 8] written with value: [00000002] | [ 2]
205: Register [ 7] written with value: [00000001] | [ 1]
215: Register [ 8] written with value: [00000002] | [ 2]
255: Register [ 7] written with value: [00000001] | [ 1]
265: Register [ 8] written with value: [00000002] | [ 2]
305: Register [ 7] written with value: [00000001] | [ 1]
315: Register [ 8] written with value: [00000002] | [ 2]
355: Register [ 7] written with value: [00000001] | [ 1]
365: Register [ 8] written with value: [00000002] | [ 2]
405: Register [ 7] written with value: [00000001] | [ 1]
415: Register [ 8] written with value: [00000002] | [ 2]
455: Register [ 7] written with value: [00000001] | [ 1]
465: Register [ 8] written with value: [00000002] | [ 2]
505: Register [ 7] written with value: [00000001] | [ 1]

```

Figura 15: Esperado

Figura 14: Forne-
cido

```

# Time: 0 Instance: tb_top.riscv_dp.data_mem.mem32.memBlock3.atsynsram_component.n_default.atsynsram_inst
#
# 45: Register [ 7] written with value: [00000001] | [ 1]
#
# 55: Register [ 2] written with value: [00000004] | [ 4]
# 65: Register [10] written with value: [0000000c] | [12]
#
# 95: Register [ 6] written with value: [00000004] | [ 4]
#
# 105: Register [ 7] written with value: [00000001] | [ 1]
#
# 115: Register [ 8] written with value: [00000002] | [ 2]
#
# 155: Register [ 7] written with value: [00000001] | [ 1]
# 165: Register [ 8] written with value: [00000002] | [ 2]
#
# 205: Register [ 7] written with value: [00000001] | [ 1]
# 215: Register [ 8] written with value: [00000002] | [ 2]
#
# 255: Register [ 7] written with value: [00000001] | [ 1]
# 265: Register [ 8] written with value: [00000002] | [ 2]
#
# 305: Register [ 7] written with value: [00000001] | [ 1]
# 315: Register [ 8] written with value: [00000002] | [ 2]
#
# 355: Register [ 7] written with value: [00000001] | [ 1]
# 365: Register [ 8] written with value: [00000002] | [ 2]
#
# 405: Register [ 7] written with value: [00000001] | [ 1]
# 415: Register [ 8] written with value: [00000002] | [ 2]
# 455: Register [ 7] written with value: [00000001] | [ 1]
# 465: Register [ 8] written with value: [00000002] | [ 2]
# 505: Register [ 7] written with value: [00000001] | [ 1]

```

Figura 16: Obtido

- BEQ (NOT TAKEN)

```

addi x7,x0,1
addi x2,x0,4
jal x10,8
or x4,x2,x0
add x6,x4,x2
addi x7,x0,1
addi x8,x0,2
beq x8,x7,-8
or x4,x2,x0

```

Figura 17: Fornecido

```

45: Register [ 7] written with value: [00000001] | [ 1]
55: Register [ 2] written with value: [00000004] | [ 4]
65: Register [10] written with value: [0000000c] | [12]
95: Register [ 6] written with value: [00000004] | [ 4]
105: Register [ 7] written with value: [00000001] | [ 1]
115: Register [ 8] written with value: [00000002] | [ 2]
135: Register [ 4] written with value: [00000004] | [ 4]

```

Figura 18: Esperado

```

# Time: 0 Instance: tb_top.riscv.dp.data_mem.mem32.memBlock3.altynoram_component.n_default.altynoram_inst
45: Register [ 7] written with value: [00000001] | [ 1]
55: Register [ 2] written with value: [00000004] | [ 4]
65: Register [10] written with value: [0000000c] | [12]
95: Register [ 6] written with value: [00000004] | [ 4]
105: Register [ 7] written with value: [00000001] | [ 1]
115: Register [ 8] written with value: [00000002] | [ 2]
135: Register [ 4] written with value: [00000004] | [ 4]

```

Figura 19: Obtido

- BNE (TAKEN)

```

addi x7,x0,1
addi x2,x0,4
jal x10,8
or x4,x2,x0
add x6,x4,x2
addi x7,x0,1
addi x8,x0,2
bne x8,x7,-8
or x4,x2,x0

```

Figura 20: Forne-

```

45: Register [ 7] written with value: [00000001] | [ 1]
55: Register [ 2] written with value: [00000004] | [ 4]
65: Register [10] written with value: [0000000c] | [12]
95: Register [ 6] written with value: [00000004] | [ 4]
105: Register [ 7] written with value: [00000001] | [ 1]
115: Register [ 8] written with value: [00000002] | [ 2]
155: Register [ 7] written with value: [00000001] | [ 1]
165: Register [ 8] written with value: [00000002] | [ 2]
205: Register [ 7] written with value: [00000001] | [ 1]
215: Register [ 8] written with value: [00000002] | [ 2]
255: Register [ 7] written with value: [00000001] | [ 1]
265: Register [ 8] written with value: [00000002] | [ 2]
305: Register [ 7] written with value: [00000001] | [ 1]
315: Register [ 8] written with value: [00000002] | [ 2]
355: Register [ 7] written with value: [00000001] | [ 1]
365: Register [ 8] written with value: [00000002] | [ 2]
405: Register [ 7] written with value: [00000001] | [ 1]
415: Register [ 8] written with value: [00000002] | [ 2]
455: Register [ 7] written with value: [00000001] | [ 1]
465: Register [ 8] written with value: [00000002] | [ 2]
505: Register [ 7] written with value: [00000001] | [ 1]

```

Figura 21: Esperado

```

# Time: 0 Instance: tb_top.riscv_dp.data_mem.mem32.memBlock3.altayncram_component.m_default.altayncram_inst
# 45: Register [ 7] written with value: [00000011] | [ 1]
#
# 55: Register [ 2] written with value: [00000004] | [ 4]
#
# 65: Register [10] written with value: [0000000c] | [12]
#
# 95: Register [ 6] written with value: [00000004] | [ 4]
#
# 105: Register [ 7] written with value: [00000001] | [ 1]
#
# 115: Register [ 8] written with value: [00000002] | [ 2]
#
# 155: Register [ 7] written with value: [00000001] | [ 1]
#
# 165: Register [ 8] written with value: [00000002] | [ 2]
#
# 205: Register [ 7] written with value: [00000001] | [ 1]
#
# 215: Register [ 8] written with value: [00000002] | [ 2]
#
# 255: Register [ 7] written with value: [00000001] | [ 1]
#
# 265: Register [ 8] written with value: [00000002] | [ 2]
#
# 305: Register [ 7] written with value: [00000001] | [ 1]
#
# 315: Register [ 8] written with value: [00000002] | [ 2]
#
# 355: Register [ 7] written with value: [00000001] | [ 1]
#
# 365: Register [ 8] written with value: [00000002] | [ 2]
#
# 405: Register [ 7] written with value: [00000001] | [ 1]
#
# 415: Register [ 8] written with value: [00000002] | [ 2]
#
# 455: Register [ 7] written with value: [00000001] | [ 1]
#
# 465: Register [ 8] written with value: [00000002] | [ 2]
#
# 505: Register [ 7] written with value: [00000001] | [ 1]

```

Figura 22: Obtido

- BLT (TAKEN)

```

addi x7,x0,1
addi x2,x0,4
jal x10,8
or x4,x2,x0
add x6,x4,x2
addi x7,x0,2
addi x8,x0,1
blt x8,x7,-8
or x4,x2,x0

```

```

# 45: Register [ 7] written with value: [00000001] | [ 1]
# 55: Register [ 2] written with value: [00000004] | [ 4]
# 65: Register [10] written with value: [0000000c] | [12]
# 95: Register [ 6] written with value: [00000004] | [ 4]
# 105: Register [ 7] written with value: [00000002] | [ 2]
# 115: Register [ 8] written with value: [00000001] | [ 1]
# 155: Register [ 7] written with value: [00000002] | [ 2]
# 165: Register [ 8] written with value: [00000001] | [ 1]
# 205: Register [ 7] written with value: [00000002] | [ 2]
# 215: Register [ 8] written with value: [00000001] | [ 1]
# 255: Register [ 7] written with value: [00000002] | [ 2]
# 265: Register [ 8] written with value: [00000001] | [ 1]
# 305: Register [ 7] written with value: [00000002] | [ 2]
# 315: Register [ 8] written with value: [00000001] | [ 1]
# 355: Register [ 7] written with value: [00000002] | [ 2]
# 365: Register [ 8] written with value: [00000001] | [ 1]
# 405: Register [ 7] written with value: [00000002] | [ 2]
# 415: Register [ 8] written with value: [00000001] | [ 1]
# 455: Register [ 7] written with value: [00000002] | [ 2]
# 465: Register [ 8] written with value: [00000001] | [ 1]
# 505: Register [ 7] written with value: [00000002] | [ 2]

```

Figura 24: Esperado

Figura 23: Fornecido

```

# Time: 0 Instance: tb_top.riscv_dp.data_mem.mem32.memBlock3.altayncram_component.m_default.altayncram_inst
# 45: Register [ 7] written with value: [00000001] | [ 1]
#
# 55: Register [ 2] written with value: [00000004] | [ 4]
#
# 65: Register [10] written with value: [0000000c] | [12]
#
# 95: Register [ 6] written with value: [00000004] | [ 4]
#
# 105: Register [ 7] written with value: [00000002] | [ 2]
#
# 115: Register [ 8] written with value: [00000001] | [ 1]
#
# 155: Register [ 7] written with value: [00000002] | [ 2]
#
# 165: Register [ 8] written with value: [00000001] | [ 1]
#
# 205: Register [ 7] written with value: [00000002] | [ 2]
#
# 215: Register [ 8] written with value: [00000001] | [ 1]
#
# 255: Register [ 7] written with value: [00000002] | [ 2]
#
# 265: Register [ 8] written with value: [00000001] | [ 1]
#
# 305: Register [ 7] written with value: [00000002] | [ 2]
#
# 315: Register [ 8] written with value: [00000001] | [ 1]
#
# 355: Register [ 7] written with value: [00000002] | [ 2]
#
# 365: Register [ 8] written with value: [00000001] | [ 1]
#
# 405: Register [ 7] written with value: [00000002] | [ 2]
#
# 415: Register [ 8] written with value: [00000001] | [ 1]
#
# 455: Register [ 7] written with value: [00000002] | [ 2]
#
# 465: Register [ 8] written with value: [00000001] | [ 1]
#
# 505: Register [ 7] written with value: [00000002] | [ 2]

```

Figura 25: Obtido

- BGE (TAKEN)

```

addi x7,x0,1
addi x2,x0,4
jal x10,8
or x4,x2,x0
add x6,x4,x2
addi x7,x0,2
addi x8,x0,1
bge x7,x8,-8
or x4,x2,x0

```

Figura 26: Fornecido

```

45: Register [ 7] written with value: [00000001] | [ 1]
55: Register [ 2] written with value: [00000004] | [ 4]
65: Register [10] written with value: [0000000c] | [12]
95: Register [ 6] written with value: [00000004] | [ 4]
105: Register [ 7] written with value: [00000002] | [ 2]
115: Register [ 8] written with value: [00000001] | [ 1]
155: Register [ 7] written with value: [00000002] | [ 2]
165: Register [ 8] written with value: [00000001] | [ 1]
205: Register [ 7] written with value: [00000002] | [ 2]
215: Register [ 8] written with value: [00000001] | [ 1]
255: Register [ 7] written with value: [00000002] | [ 2]
265: Register [ 8] written with value: [00000001] | [ 1]
305: Register [ 7] written with value: [00000002] | [ 2]
315: Register [ 8] written with value: [00000001] | [ 1]
355: Register [ 7] written with value: [00000002] | [ 2]
365: Register [ 8] written with value: [00000001] | [ 1]
405: Register [ 7] written with value: [00000002] | [ 2]
415: Register [ 8] written with value: [00000001] | [ 1]
455: Register [ 7] written with value: [00000002] | [ 2]
465: Register [ 8] written with value: [00000001] | [ 1]
505: Register [ 7] written with value: [00000002] | [ 2]

```

Figura 27: Esperado

```

Time: 0 Instance: tb_top.riscv.dp.data_mem.mem32.memBlock3.altysyncram_component_m_default.altysyncram_inst
45: Register [ 7] written with value: [00000001] | [ 1]
55: Register [ 2] written with value: [00000004] | [ 4]
65: Register [10] written with value: [0000000c] | [12]
95: Register [ 6] written with value: [00000004] | [ 4]
105: Register [ 7] written with value: [00000002] | [ 2]
115: Register [ 8] written with value: [00000001] | [ 1]
155: Register [ 7] written with value: [00000002] | [ 2]
165: Register [ 8] written with value: [00000001] | [ 1]
205: Register [ 7] written with value: [00000002] | [ 2]
215: Register [ 8] written with value: [00000001] | [ 1]
255: Register [ 7] written with value: [00000002] | [ 2]
265: Register [ 8] written with value: [00000001] | [ 1]
305: Register [ 7] written with value: [00000002] | [ 2]
315: Register [ 8] written with value: [00000001] | [ 1]
355: Register [ 7] written with value: [00000002] | [ 2]
365: Register [ 8] written with value: [00000001] | [ 1]
405: Register [ 7] written with value: [00000002] | [ 2]
415: Register [ 8] written with value: [00000001] | [ 1]
455: Register [ 7] written with value: [00000002] | [ 2]
465: Register [ 8] written with value: [00000001] | [ 1]
505: Register [ 7] written with value: [00000002] | [ 2]

```

Figura 28: Obtido

- JALR

```

addi x7,x0,-1
sw x7,0(x0)
lw x9,0(x0)
or x4,x2,x0
add x6,x4,x2
jalr x12,x0,12

```

Figura 29: Fornecido

```

45: Memory [ 0] written with value: [ffffff] | [4294967295]
45: Register [ 7] written with value: [ffffff] | [4294967295]
55: Memory [ 0] read with value: [xxxxxxxx] | [ x]
55: Memory [ 0] read with value: [00000000] | [ 0]
60: Memory [ 0] read with value: [ffffff] | [4294967295]
65: Register [ 9] written with value: [ffffff] | [4294967295]
75: Register [ 4] written with value: [00000000] | [ 0]
85: Register [ 4] written with value: [00000000] | [ 0]
95: Register [12] written with value: [00000018] | [ 24]
125: Register [ 4] written with value: [00000000] | [ 0]
135: Register [ 6] written with value: [00000000] | [ 0]
145: Register [12] written with value: [00000018] | [ 24]
175: Register [ 4] written with value: [00000000] | [ 0]
185: Register [ 6] written with value: [00000000] | [ 0]
195: Register [12] written with value: [00000018] | [ 24]
225: Register [ 4] written with value: [00000000] | [ 0]
235: Register [ 6] written with value: [00000000] | [ 0]
245: Register [12] written with value: [00000018] | [ 24]
275: Register [ 4] written with value: [00000000] | [ 0]
285: Register [ 6] written with value: [00000000] | [ 0]
295: Register [12] written with value: [00000018] | [ 24]
325: Register [ 4] written with value: [00000000] | [ 0]
335: Register [ 6] written with value: [00000000] | [ 0]
345: Register [12] written with value: [00000018] | [ 24]
375: Register [ 4] written with value: [00000000] | [ 0]
385: Register [ 6] written with value: [00000000] | [ 0]
395: Register [12] written with value: [00000018] | [ 24]
425: Register [ 4] written with value: [00000000] | [ 0]
435: Register [ 6] written with value: [00000000] | [ 0]
445: Register [12] written with value: [00000018] | [ 24]
475: Register [ 4] written with value: [00000000] | [ 0]
485: Register [ 6] written with value: [00000000] | [ 0]
495: Register [12] written with value: [00000018] | [ 24]

```

Figura 30: Esperado

```

# ** Note: (stop    : verif/th_top.vv(45)
#      Time: 318 ns  Iteration: 0   Instance: /th_top

```

Figura 31: Obtido

4.4 Simulações de STORE

- SW

```
addi x7,x0,-1
sw x7,0(x0)
lw x9,0(x0)
```

```
45: Memory [ 0 ] written with value: [ffffff] | [4294967295]
45: Register [ 7 ] written with value: [ffffff] | [4294967295]
55: Memory [ 0 ] read with value: [xxxxxxxx] | [ x]
55: Memory [ 0 ] read with value: [fffffaa8] | [4294945488]
68: Memory [ 0 ] read with value: [ffffff] | [4294967295]
65: Register [ 9 ] written with value: [ffffff] | [4294967295]
```

Figura 33: Esperado

Figura 32: Fornecido

```

Time: 0 Instance: [0] Instency: [0] data mem mem3 mem3lo0: [0] atpncrm_component: [0] default: [0] atpncrm_inst
45: Memory [ 0] written with value: [fffffffe] [ 4294967295]
45: Register [ 7] written with value: [fffffffe] [ 4294967295]
55: Memory [ 0] read with value: [xxxxxxxx] [  ]
55: Memory [ 0] read with value: [00000001] [ 1]
60: Memory [ 0] read with value: [fffffffe] [ 4294967295]
65: Register [ 9] written with value: [fffffffe] [ 4294967295]

** Note: Cases : verify/wh_top.v(43)
Transaction: 0 Parameters: [0]

```

Figura 34: Obtido

- SB, SH

```
addi x7,x0,0
sb x7,2(x0)
lw x9,0(x0)
sh x7,2(x0)
lw x8,0(x0)
```

Figura 35: Fornecido

```

45: Memory [ 2] written with value: [00000000] | [ 0]
45: Register [ 7] written with value: [00000000] | [ 0]
55: Memory [ 0] read with value: [xxxxxxxx] | [ x]
55: Memory [ 0] read with value: [0000ffff] | [ 65535]
60: Memory [ 0] read with value: [ff00aa80] | [4278233728]
65: Memory [ 2] written with value: [00000000] | [ 0]
65: Register [ 9] written with value: [ff00aa80] | [4278233728]
75: Memory [ 0] read with value: [ff00aa80] | [4278233728]
75: Memory [ 0] read with value: [0000ffff] | [ 65535]
80: Memory [ 0] read with value: [0000aa80] | [ 43648]
85: Register [ 8] written with value: [0000aa80] | [ 43648]

```

Figura 36: Esperado

```

Time: 0 Instance: tb_top_riscv_dp_data_mem.mem32.mem1lock.alaysprocram_component_m_default.alaysprocram
45: Memory [ 2] written with value: [00000000] | { 0}
45: Register [ 7] written with value: [00000000] | { 0}
55: Memory [ 0] read with value: [xxxxxxxx] | { x}
55: Memory [ 0] read with value: [00030000] | { 196408}
60: Memory [ 0] read with value: [00000000] | { 0}
55: Memory [ 2] written with value: [00000000] | { 0}
65: Register [ 9] written with value: [00000000] | { 0}
75: Memory [ 0] read with value: [00000000] | { 0}
75: Memory [ 0] read with value: [00030000] | { 196408}
80: Memory [ 0] read with value: [00000000] | { 0}
55: Register [ 0] written with value: [00000000] | { 0}
** Note: Setup : verify(tb_top.sv(43))

```

Figura 37: Obtido

4.5 HALT

O comportamento da pseudo-instrução ****HALT**** foi validado, garantindo que o processador interrompesse sua execução corretamente.

```
lw x1, 0(x0)
add x2, x1, x1
halt
lw x3, 4(x0)
add x4, x2, x3
sub x5, x4, x2
```

Figura 38: Utilizado

```

DEPTH = 65536;           -- The size of memory in words
WIDTH = 8;               -- The size of data in bits
ADDRESS_RADIX = DEC;     -- The radix for address values
DATA_RADIX = BIN;        -- The radix for data values
CONTENT
-- Start of (address: data pairs)
BEGIN
000: 00000001;           -- lw x1,0(x0)
001: 00100000;
002: 00000000;
003: 00000000;
004: 00100001;           -- add x2,x1,x1
005: 00000001;
006: 00010000;
007: 00000000;
008: 00000001;           -- halt
009: 00000000;
010: 00000000;
011: 00000000;
012: 10000011;           -- lw x3,x1(x0)
013: 01000001;
014: 01000000;
015: 00000000;
016: 00110011;           -- add x4,x2,x3
017: 00000001;
018: 00110001;
019: 00000000;
020: 01110011;           -- sub x5,x4,x2
021: 00000001;
022: 01000010;
023: 01000000;
END;

```

Figura 39: .MIF

[illegible]

Figura 40: Resultdo

5 Conclusão

A implementação de um processador RISC-V em pipeline foi concluída com sucesso, validando sua operação por meio do `**ModelSim**`. O projeto permitiu compreender em detalhes o funcionamento do pipeline e os desafios associados à sua implementação, como dependências de dados e controle de fluxo. Futuras melhorias podem incluir otimizações no pipeline e a adição de novas instruções para ampliar as funcionalidades do processador.