

**Universidade Federal De Roraima**

**Centro De Ciência e Tecnologia**

**Departamento de Ciência da Computação**

**CURSO DE BACHAREL EM CIÊNCIAS DA COMPUTAÇÃO**

**PROCESSADOR RISC DE 8 BITS**

Boa Vista/RR

2025

**ACADÊMICO:**

**ABRAHÃO PICANÇO NERES DE OLIVEIRA**

**JONATHAN EMERSON BRAGA DA SILVA**

PROCESSADOR RISC DE 8 BITS

Trabalho Avaliativo para matéria de Arquitetura e Organização de Computadores em Ciências da Computação da Universidade de Roraima - UFRR.

Orientador: Herbert Oliveira

Boa Vista/RR

2025

**SUMÁRIO**

[**1. INTRODUÇÃO 4**](#_168p04bzr4i3)

[**2. ESPECIFICAÇÃO 4**](#_goy1drm1twjw)

[**2.1 PLATAFORMA DE DESENVOLVIMENTO 4**](#_1fwrt4v4p4oe)

[**2.2 CONJUNTO DE INSTRUÇÕES 4**](#_bs3d65cu622r)

[**2.2.1 CAMPOS DAS INSTRUÇÕES: 4**](#_xapjvaz9z7vm)

[**2.3 DESCRIÇÃO DO HARDWARE 5**](#_a29duylyzh4p)

[**2.3.1 PC 5**](#_j0hlrd4rhs9e)

[**2.3.2 ROM 6**](#_57r5xkt6q9j4)

[**2.3.3 DECODIFICADOR 7**](#_xtptl4ba2dvv)

[**2.3.4 CONTROL 7**](#_mmsibww8nfjl)

[**2.3.5 BANCO DE REGISTRADORES 8**](#_58qrznas49xt)

[**2.3.6 ULA 9**](#_umtn4xpxthg)

[**2.3.7 MUX 10**](#_bw8fbqe0llut)

[**2.3.8 RAM 11**](#_2xhpwuwqpl47)

[**2.4 DATAPATH 11**](#_bejueg5yynfw)

[**3. TESTES 12**](#_k2re1r48k4s5)

[**3.1 ULA 12**](#_juesavisxza0)

[**3.2 UNIDADE DE CONTROLE 12**](#_tnmno8m8kmf)

[**3.3 PC 12**](#_dugz2kp0kown)

[**3.4 RAM 12**](#_2mhw7qojai60)

[**3.5 ROM 13**](#_n0tlkqgfpl8l)

[**3.6 MUX 13**](#_thrtpn90xmgb)

[**3.7 DECODIFICADOR 13**](#_mf16oxlhdm4q)

[**3.8 BANCO DE REGISTRADORES 13**](#_iz7ew5hw7n1c)

[**4. CONSIDERAÇÕES 14**](#_dzlnqlw5vrwb)

[**5. REFERÊNCIAS 15**](#_r5qycp7uhdy)

## **1. INTRODUÇÃO**

O projeto de um processador RISC de 8 bits exige um planejamento detalhado das instruções que serão suportadas, bem como a definição de sua arquitetura e funcionamento. Este documento descreve o planejamento das instruções, incluindo o conjunto de instruções, formatos, modos de endereçamento e a organização do datapath e da unidade de controle.

## **2. ESPECIFICAÇÃO**

Este trabalho apresenta o desenvolvimento de um processador básico e eficiente, implementado em VHDL, formado por componentes interligados que executam funções fundamentais.

### **2.1 PLATAFORMA DE DESENVOLVIMENTO**

Para a implementação do processador, foi utilizada a IDE Quartus Prime Lite Edition.

### **2.2 CONJUNTO DE INSTRUÇÕES**

O processador projetado (RISC-8bits) possui 4 registradores internos identificados por R0, R1, R2 e R3. Ele utiliza instruções de 8 bits organizadas em dois formatos principais:

* Formato Geral para instruções aritméticas (ADD, SUB, ADDI), acesso à memória (LW, SW), carregamento imediato (LI) e salto condicional (BEQ).
* Formato J (JUMP) para salto incondicional, com endereço específico.

#### **2.2.1 CAMPOS DAS INSTRUÇÕES:**

* Opcode (3 bits) [7-5]: Define a operação a ser executada pelo processador.
* RD (2 bits) [4-3]: Registrador de destino que armazena resultados ou recebe dados da memória.
* RS (2 bits) [2-1]: Registrador fonte, fornece o segundo operando em operações aritméticas.
* Imediato (3 bits) [2-0]: Valor imediato usado nas instruções que o requerem (LI, ADDI, LW, SW, BEQ).  
   Nota: Os bits do registrador RS [2-1] são parcialmente sobrepostos pelo campo imediato [2-0].
* Endereço Jump (5 bits) [4-0]: Valor utilizado exclusivamente para o salto incondicional (JUMP).



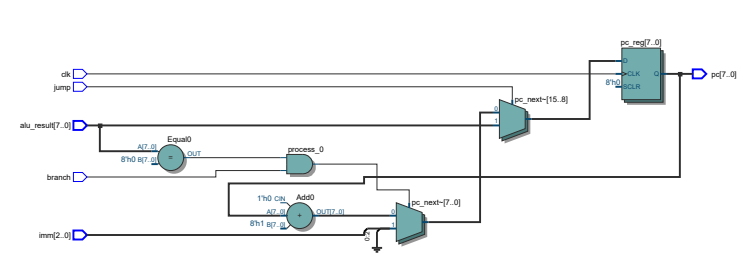
**1.Tabela**

### **2.3 DESCRIÇÃO DO HARDWARE**

Nesta seção, são apresentados os elementos de hardware que formam o processador, acompanhados de uma explicação sobre suas funcionalidades, além dos valores de entrada e saída.

#### **2.3.1 PC**

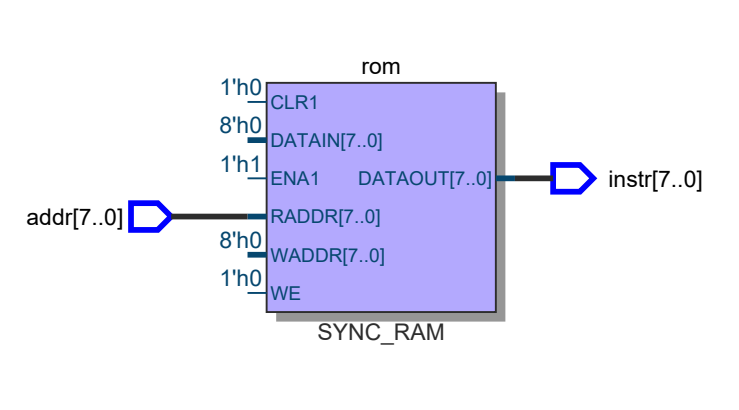
O módulo Contador de Programa mantém o endereço da próxima instrução em um registrador de 8 bits, atualizado na borda de subida do clock. Normalmente, incrementa o valor em um, mas pode assumir um resultado de 8 bits da unidade aritmética para salto incondicional ou um imediato de 3 bits estendido para 8 bits em desvio condicional, se a condição de igualdade for verdadeira. A saída reflete o endereço atualizado.



**1.Componente PC gerado pelo Quartus**

#### **2.3.2 ROM**

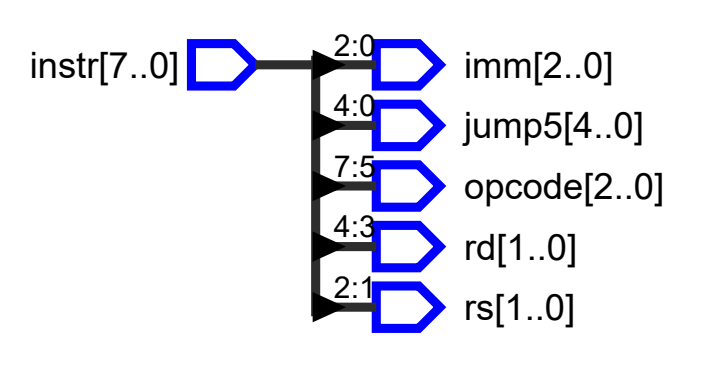
O módulo ROM armazena instruções de um programa em uma memória somente leitura com 256 posições de 8 bits. Ele recebe um endereço de 8 bits e entrega a instrução correspondente, também de 8 bits. O conteúdo é fixo, com um programa Fibonacci nas primeiras 7 posições e testes gerais nas 11 seguintes, incluindo operações como carregamento, soma, armazenamento e saltos. As posições restantes são nulas, e a instrução é obtida usando o endereço como índice da memória.



**2.Componente ROM gerado pelo Quartus**

#### **2.3.3 DECODIFICADOR**

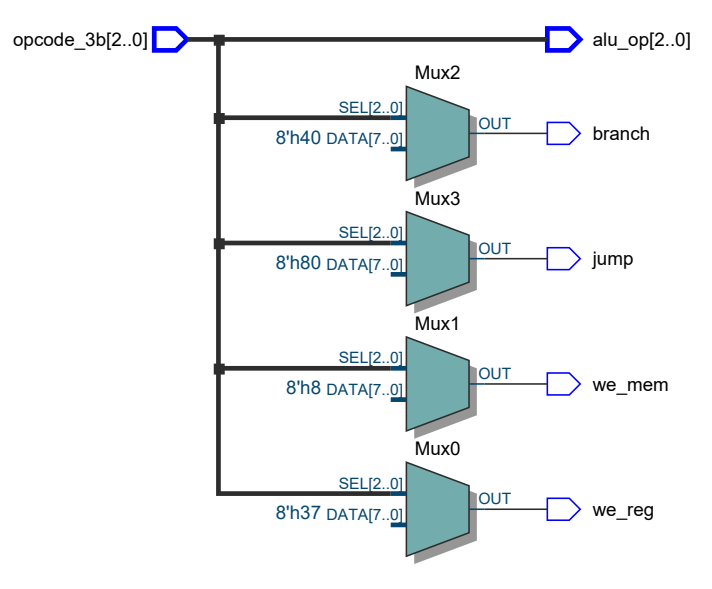
O módulo Decodificador interpreta uma instrução de 8 bits e separa seus campos: os 3 bits mais significativos para o tipo de operação, 2 bits para o registrador de destino, 2 bits para o registrador de origem, 3 bits para o valor imediato e 5 bits para o endereço de salto. A decodificação ocorre diretamente, distribuindo os bits da instrução para as saídas de maneira instantânea e sem armazenamento.

****

**3.Componente DECODIFICADOR gerado pelo Quartus**

#### **2.3.4 CONTROL**

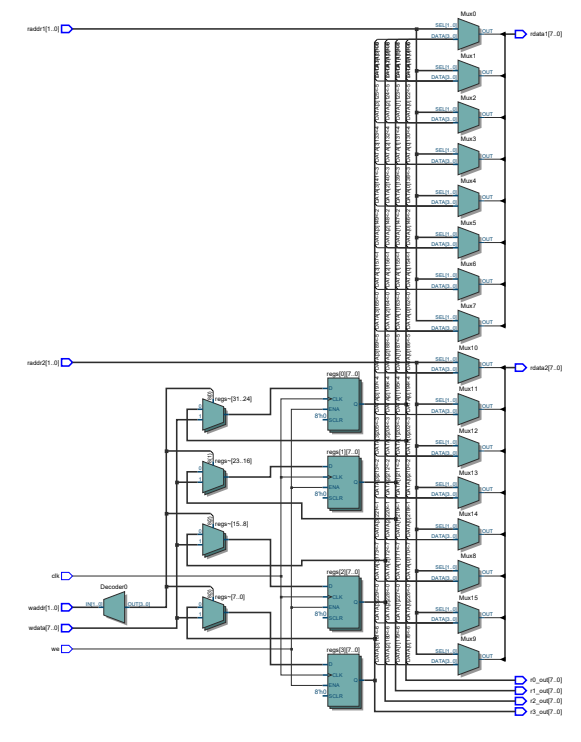
O módulo Control é a unidade de controle que decodifica um opcode de 3 bits e define sinais para operar um processador. Ele ativa a escrita em registradores para instruções como soma e carregamento, a escrita na memória para armazenamento, o salto condicional para desvio se igual, o salto incondicional para pulo direto e ajusta a operação da unidade lógica e aritmética. Em resumo, transforma instruções em comandos básicos.



**4.Componente CONTROL gerado pelo Quartus**

#### **2.3.5 BANCO DE REGISTRADORES**

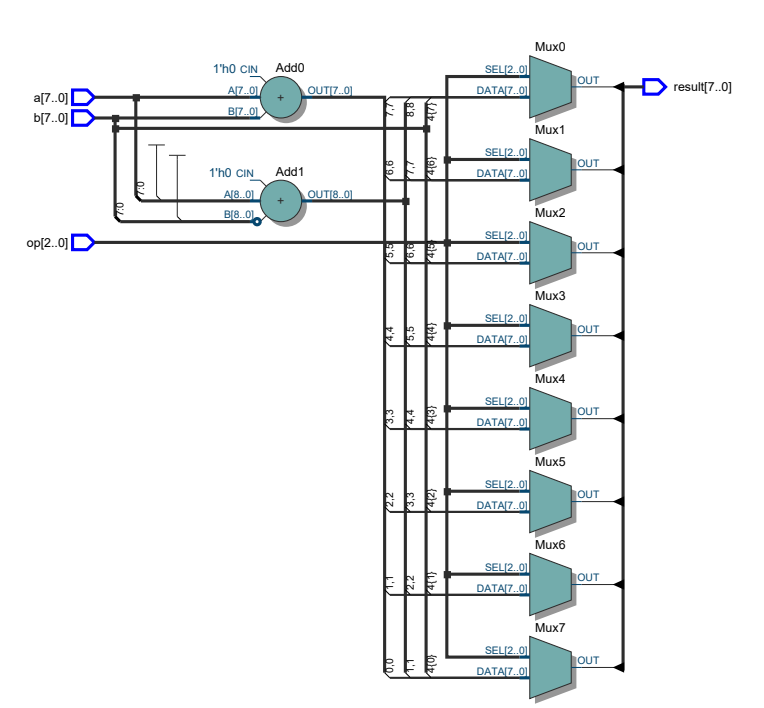
O módulo Banco de Registradores gerencia quatro registradores de 8 bits, controlados por um clock. Ele recebe um sinal de escrita, endereços de 2 bits para leitura e escrita, e dados de 8 bits para armazenar. Quando o sinal de escrita é ativo, grava os dados no registrador indicado na borda de subida do clock. Fornece duas saídas de leitura de 8 bits baseadas nos endereços fornecidos e saídas específicas para cada um dos quatro registradores.



**5.Componente BANCO DE REGISTRADORES gerado pelo Quartus**

#### **2.3.6 ULA**

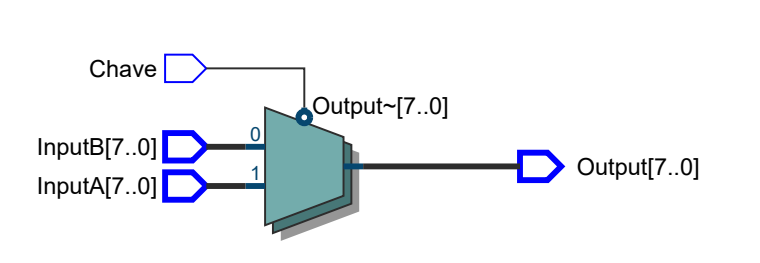
O módulo Unidade Lógica e Aritmética realiza operações em dois operandos de 8 bits, guiado por um código de 3 bits. Ele soma ou subtrai os operandos, passa um valor diretamente para carregamento ou armazenamento, verifica igualdade para desvio condicional, ou retorna um endereço para salto. O resultado de 8 bits é gerado instantaneamente com base na operação escolhida, retornando zero se o código for inválido.



**6.Componente ULA gerado pelo Quartus**

#### **2.3.7 MUX**

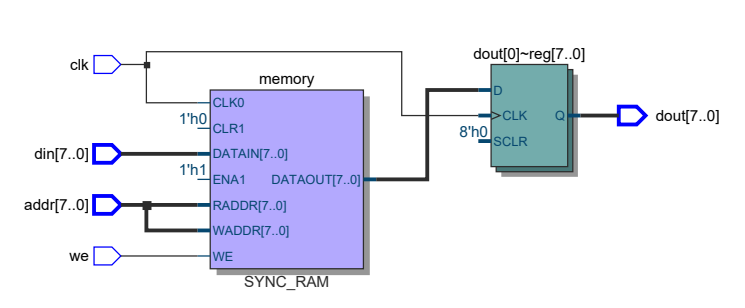
O módulo Multiplexador escolhe entre duas entradas de 8 bits com base em um sinal de controle de 1 bit. Se o sinal é zero, seleciona a primeira entrada; se é um, seleciona a segunda. A saída de 8 bits reflete a escolha feita instantaneamente, retornando zero caso o sinal de controle seja inválido.



**7.Componente ULA gerado pelo Quartus**

#### **2.3.8 RAM**

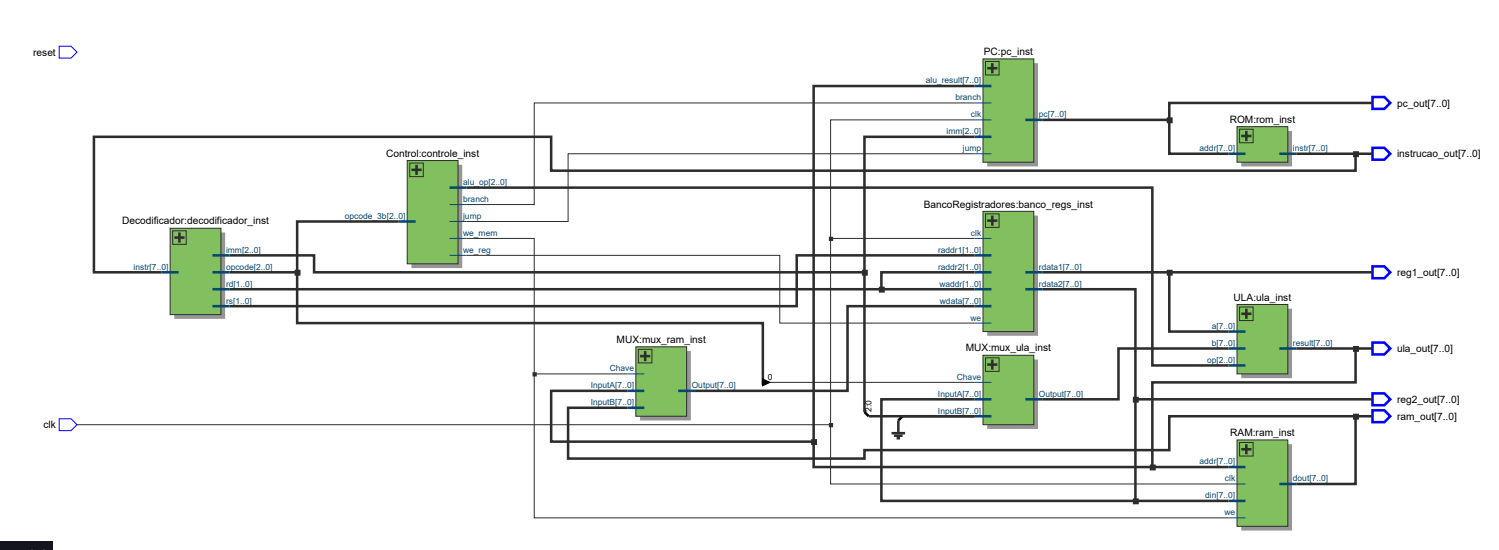
O módulo RAM armazena e acessa dados em uma memória de 256 posições de 8 bits, controlada por um clock. Ele usa um endereço de 8 bits para localizar a posição, escreve um dado de entrada de 8 bits quando o sinal de escrita é ativo na borda de subida do clock, e fornece o dado armazenado como saída de 8 bits de forma síncrona.



**8.Componente ULA gerado pelo Quartus**

#### **2.4 DATAPATH**

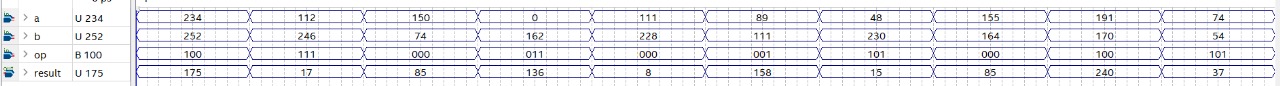
Trata-se da interligação das unidades funcionais, constituindo um caminho único para o fluxo de dados, e incorporando uma unidade de controle encarregada pela coordenação das operações executadas para distintas categorias de instruções.

****

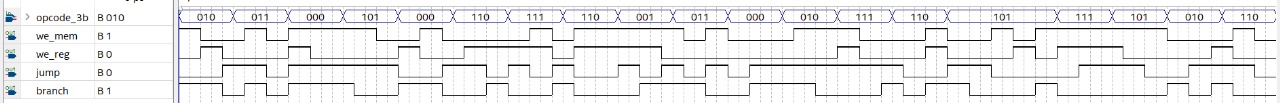
**9.Componente gerado pelo Quartus**

## **3. TESTES**

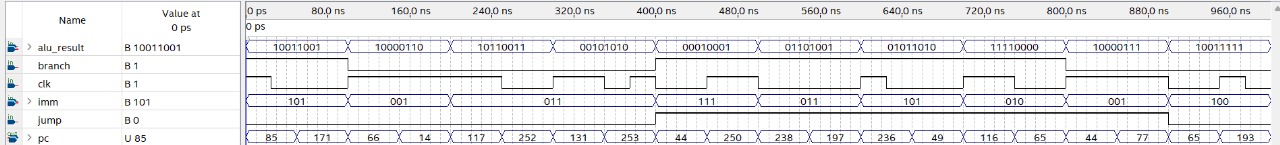
### **3.1 ULA**



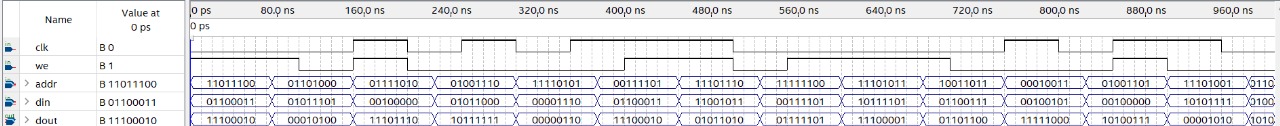
### **3.2 UNIDADE DE CONTROLE**



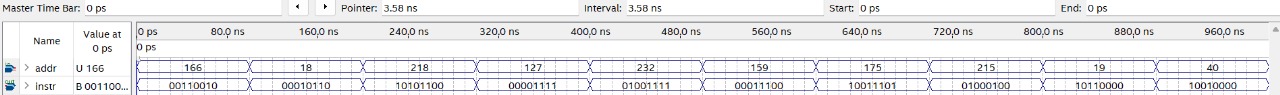
### **3.3 PC**



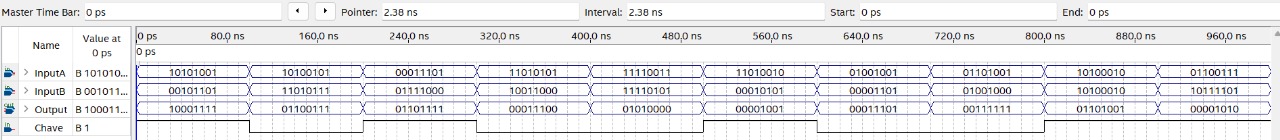
### **3.4 RAM**



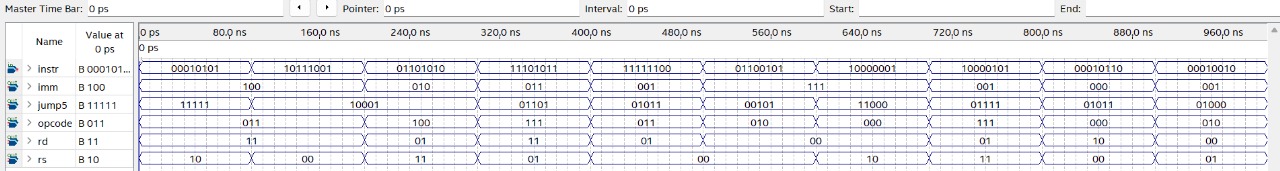
### **3.5 ROM**



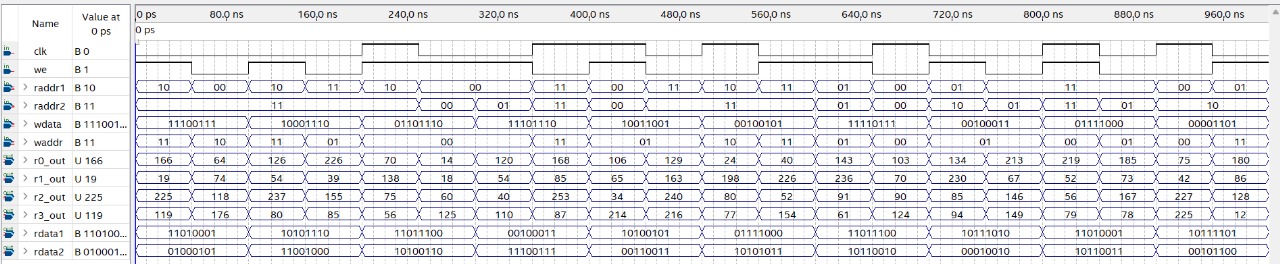
### **3.6 MUX**



### **3.7 DECODIFICADOR**



### **3.8 BANCO DE REGISTRADORES**



## **4. CONSIDERAÇÕES**

As considerações finais sobre este trabalho destacam a construção de um processador simples e funcional implementado em VHDL, composto por módulos interconectados que realizam tarefas essenciais. O Contador de Programa gerencia o fluxo de instruções, a ROM armazena programas pré-definidos como Fibonacci e testes, enquanto o Decodificador e a Unidade de Controle interpretam e direcionam as operações.

A Unidade Lógica e Aritmética executa cálculos e comparações, o Banco de Registradores armazena dados temporários, a RAM oferece memória volátil, e o Multiplexador facilita escolhas de dados. Juntos, esses componentes formam um sistema coeso, demonstrando princípios básicos de arquitetura de computadores com eficiência e clareza.

## **5. REFERÊNCIAS**

https://www.youtube.com/watch?v=8so9In8ZQUM&list=PLYE3wKnWQbHDdnb3FsDkNx2tj8xoQAPtN

https://www.youtube.com/watch?v=9DQO3H-UJCw