PODER EXECUTIVO
MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE RORAIMA
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES
RELATÓRIO DO PROJETO: PROCESSADOR LEIDEBUG

ALUNOS:
GABRIEL PEIXOTO MENEZES DA COSTA – 2020022626
NATÁLIA RIBEIRO DE ALMADA – 2017009364

Março de 2022
Boa Vista/Roraima

PODER EXECUTIVO
MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE RORAIMA
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES
RELATÓRIO DO PROJETO: PROCESSADOR LEIDEBUG

Março de 2022
Boa Vista/Roraima

Resumo

Este projeto aborda a elaboração e implementação do processador uniciclo de 8 bits denominado “LeiDeBug” baseado na arquitetura do processador MIPS. Programado na linguagem VHDL, que é utilizada de forma extensiva para o design de circuitos digitais. A implementação do projeto foi realizada na IDE Quartus Prime Lite Edition v20.1.1.720. O processador conta com Program Counter, Memória de Instruções (ROM), Banco de Registradores, Unidade Lógica Aritmética (ULA) e Memória de Dados (RAM). Os testes dos componentes do processador “LeiDeBug” foram realizados com o simulador Modelsim, tentando demonstrar o funcionamento das entradas e saídas dos componentes.

Palavras-Chave: MIPS, VHDL, Processador, 8 Bits, Quartus Prime Lite.

Abstract

This project addresses the design and implementation of a 8 bits monocycle processor named “LeiDeBug” based on a MIPS processor architecture. It was programmed on VHDL, extensively used on digital circuits design. The implementation was made on Quartus Prime Lite Edition v20.1.1.720 IDE. The processor has a Program Counter, Instruction Memory (ROM), Register Bank, Arithmetic Logic Unit (ALU) and Data Memory (RAM). The tests of the components of “LeiDeBug” processor were carried 20 out with the Modelsim simulator, trying to demonstrate the functioning of the inputs and outputs of the components.

Keywords: MIPS, VHDL, Processor, 8 Bits, Quartus Prime Lite.

Conteúdos

Lista de Figuras

1.Especificação

Nesta seção serão apresentados os itens para o desenvolvimento do processador **LeiDeBug** e a descrição de cada etapa da construção do processador.

1.1 Plataforma de desenvolvimento

Para a implementação do processador **LeiDeBug** foi utilizada a IDE: Quartus Prime 18.1 Lite Edition

Flow Status	Successful - Thu Mar 10 03:59:41 2022
Quartus Prime Version	20.1.1 Build 720 11/11/2020 SJ Lite Edition
Revision Name	LeiDeBug8Bits
Top-level Entity Name	LeiDeBug8BitsProcessador
Family	Cyclone V
Device	5CGXFC7C7F23C8
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	24
Total pins	110
Total virtual pins	0
Total block memory bits	0
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0
Total DLLs	0

1.2 Conjunto de instruções

O processador **LeiDeBug** possui os registradores S0 e S1, que combinam-se a 3 formatos de instruções do tipo R, I e J de 8 bits:

Opcode: operação básica executada pelo processador.

Reg1: o registrador contendo o primeiro operando, é o registrador de destino;

Reg2: o registrador contendo o segundo operando, é a fonte;

Funct: formato para escrita em código binário:

Tipo de Instruções:

Formato do tipo R: Instruções de operações aritméticas.

OPCODE	R1	R2	FUNCT
3 BITS	1 BIT	1 BIT	3 BITS
7-5	4	3	2-0

Formato do tipo I: Instruções de Load, Store, Load Immediately e Branch.

OPCODE	R1	FUNCT
3 BITS	1 BIT	4 BITS
7-5	4	3-0

Formato do tipo J: Instrução do tipo Jump.

OPCODE	ENDEREÇO
3 BITS	5 BITS
7-5	4-0

Visão geral das instruções do Processador **LeiDeBug** :

São 4 bits do campo das instruções Opcod, então:
(*Bit(0e1)NumeroTodaldeBitsdoOpcode* ∴ 2X = X)) totalizam 8 Opcodes que preenchem de 0-7

OPCODE	Nome	Formato	Breve Descrição	Exemplo
000000	ADD	R	Soma	add $S0,$S1
000001	SUB	R	Subtração	sub $S0,$S1
000100	MULT	R	Multiplicação	mult $S0,$S1
001	LW	I	Load	lw $S0,5
110	LI	I	Load immediately	li $S0,20
010	SW	I	Store	sw $S0,3
011	BNE	I	Branch Not Equal	bne $S0,$S1,LABEL
100	BEQ	I	Branck Equal	beq $S0,$S1,LABEL
111	JUMP	J	Jump	jump Endereço

1.3 Descrição do Hardware

Nesta seção são descritos os componentes do hardware que compõem o processador **LeiDeBug**, incluindo uma descrição de suas funcionalidades, valores de entrada e saída.

1.3.1 ALU ou ULA

O componente ULA (Unidade Lógica Aritmética) tem como principal objetivo efetuar as principais operações aritméticas de números inteiros, e também efetua operações de comparação de valor como igual ou diferente. A ULA recebe três valores:

a - dado de 8bits para operação,

b - dado de 8bits para operação;

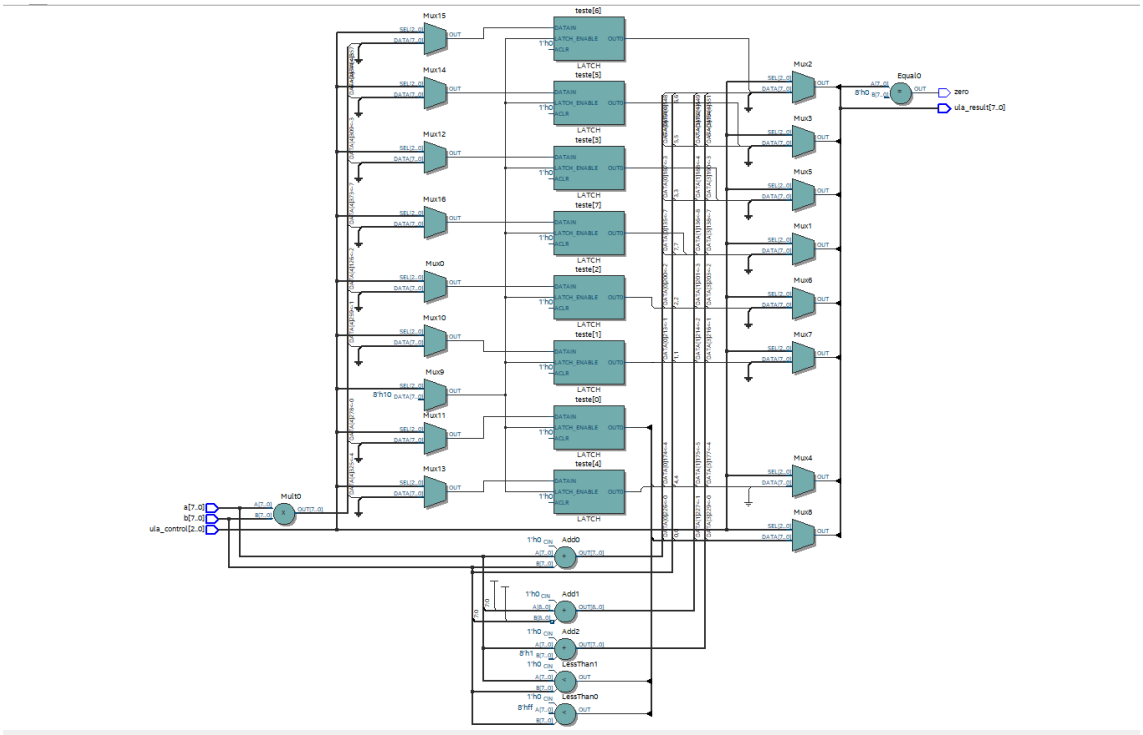
ula_controle - identificador da operação que será realizada de 3bits.

A ULA também possui duas saídas: zero - identificador de resultado (1bits) para comparações (1 se verdade e 0 caso contrário);

Figura 1- ULA (RTL View)

1.3.2 Banco de Registradores

O BDRegister guarda valores e/ou resultados das operações realizadas pela ULA.



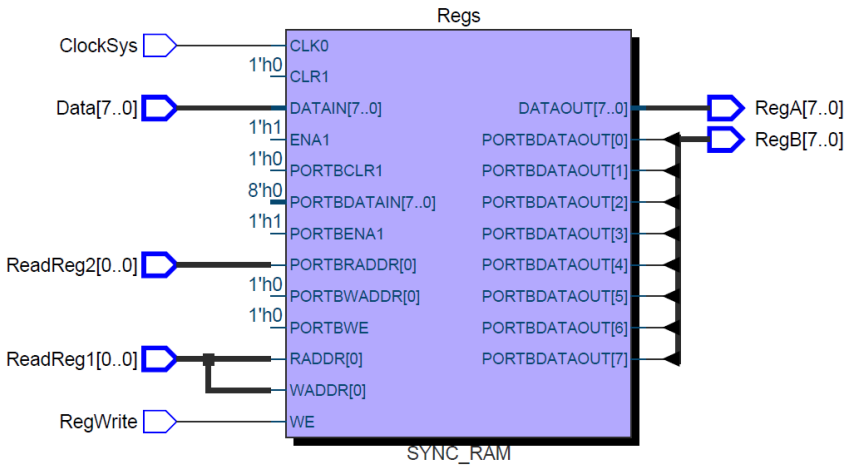


Figura 2 – Banco de Registradores

1.3.3 Clock

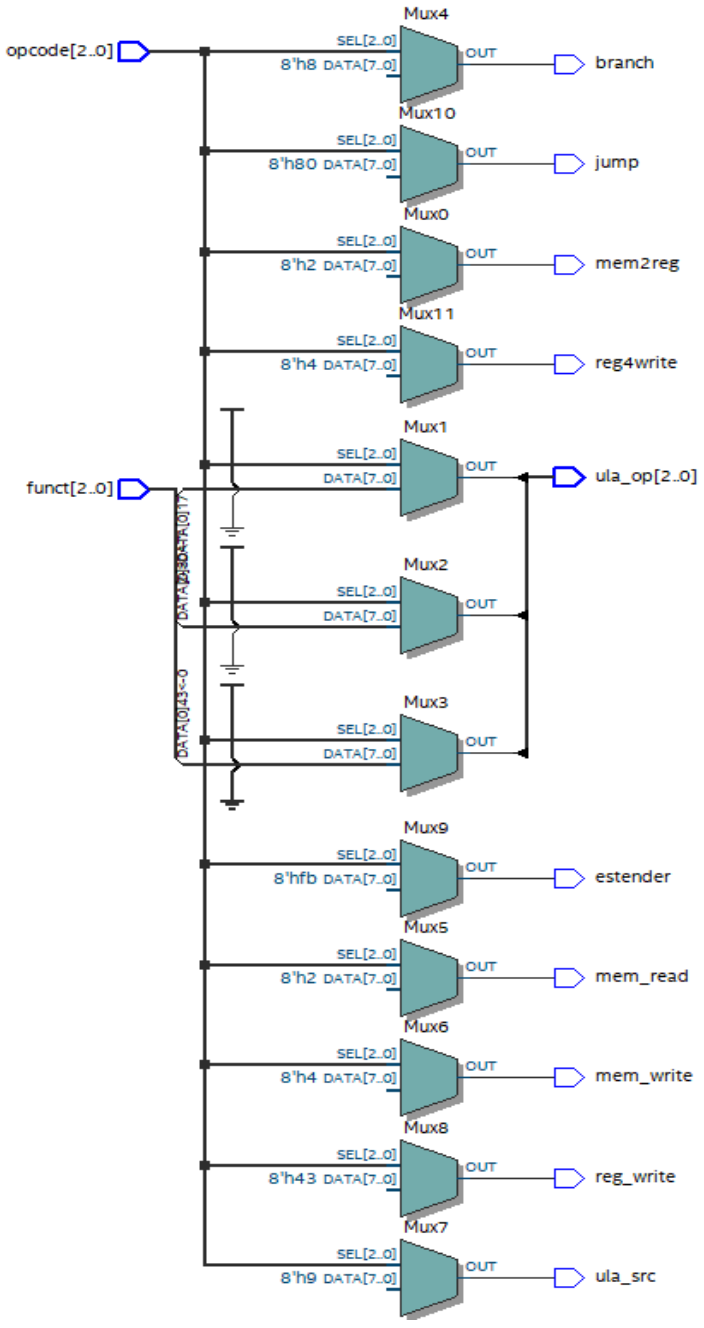
Serve para manter os componentes funcionando apenas quando estiver ligado, portanto se for True/1.

1.3.4 Unidade controle

O componente Control tem como objetivo realizar o controle de todos os componentes do processador de acordo com o opcode.

Figura 3 - Unidade de Controle

1.3.5 Memória de dados ou RAM



Armazena os valores na memória e os disponibiliza para serem salvos nos registradores. Tendo como entrada dado_entrada (que define o que vai ser salvo na memória); endereco - o endereço (onde o valor será salvo na memória); EscMem - flag que define

se algo vai ser escrito na memória ou não; LeMem - flag que define se algo vai ser lido da memória e passada para o registrador; clk - clock. E ele retorna com o que foi lido na memória - dado_saida.

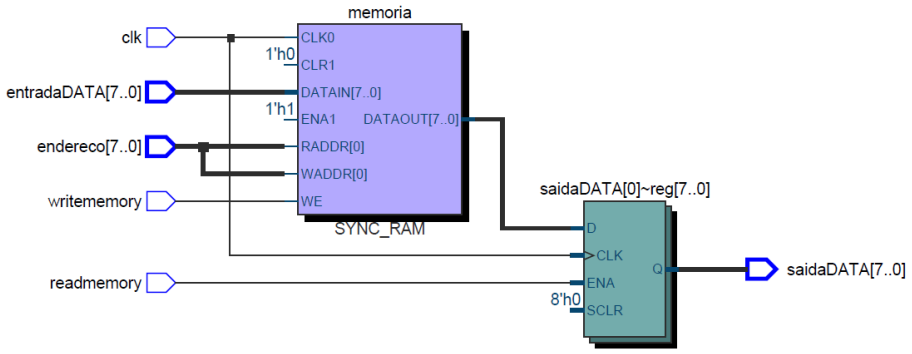


Figura 4 - RAM

1.3.6 Memória de Instruções ou ROM

Onde todas as instruções serão executadas e é a partir dela que o opcode vai para a unidade de controle e que sabemos quais registradores usar. A ROM recebe um endereço que vem do PC e executa o que está nesta instrução.

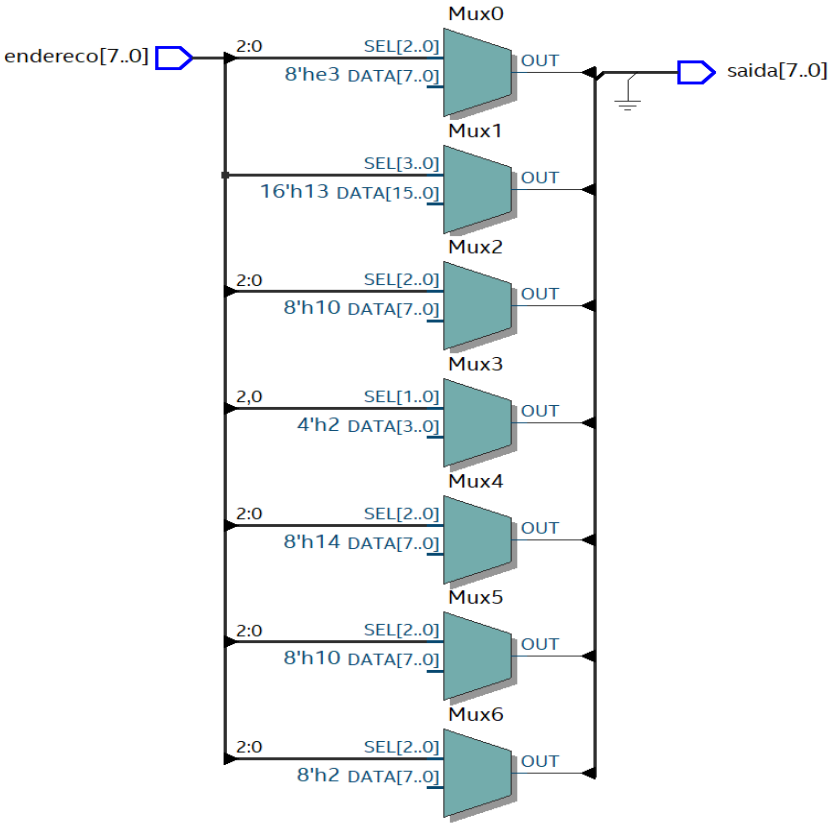


Figura 5 - ROM

1.3.7 Somador

Soma cada PC para que ele execute as instruções

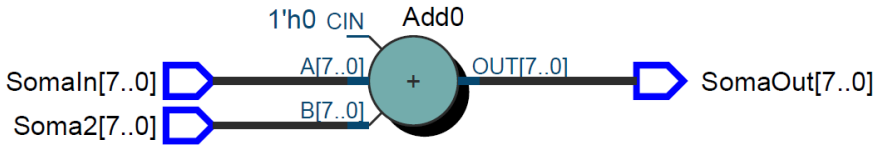


Figura 6 - Somador

1.3.8 And

Verifica se vai ou não ocorrer um branch

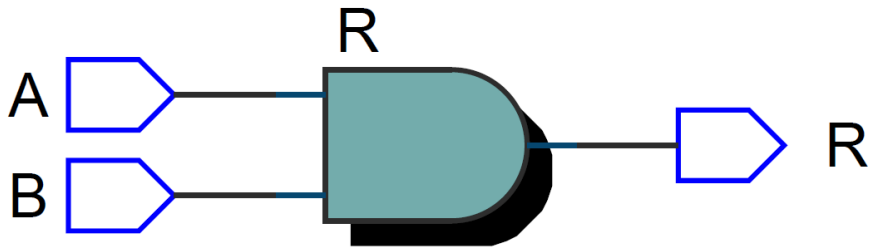


Figura 7 - AND

1.3.9 Multiplexador

O multiplexador auxilia a decidir que trilha será usada dependendo do valor da flag que está recebendo.

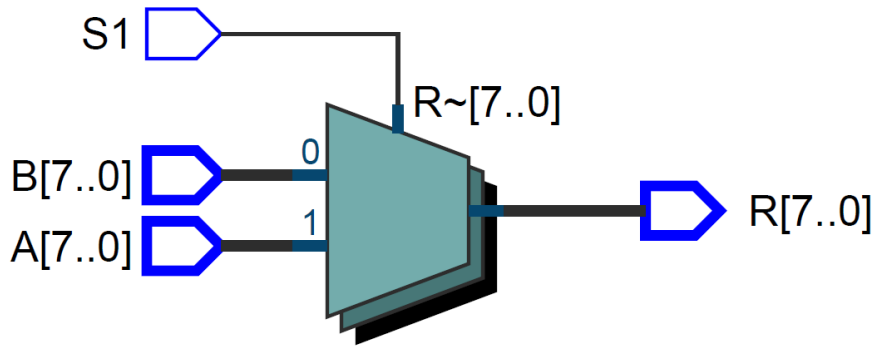


Figura 8 – Multiplexador

1.3.10 Program Counter

Responsável em mandar o endereço para a próxima instrução para a memória de instruções.

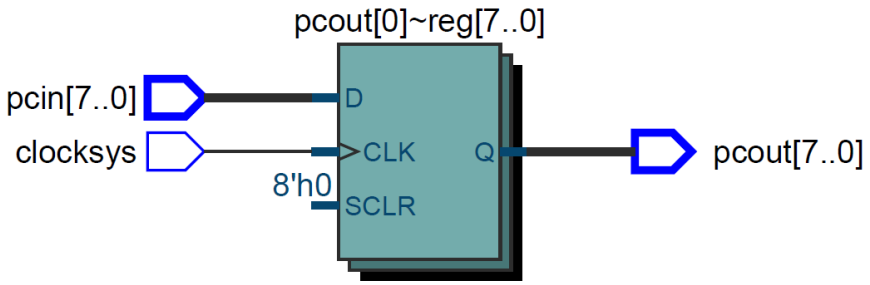


Figura 9 - PC

1.3.11 ZERO

Identificador de resultado (1bit) para comparações (1 se verdade e 0 se falso).

1.4. LeiDeBug8BitsProcessador. RTL View

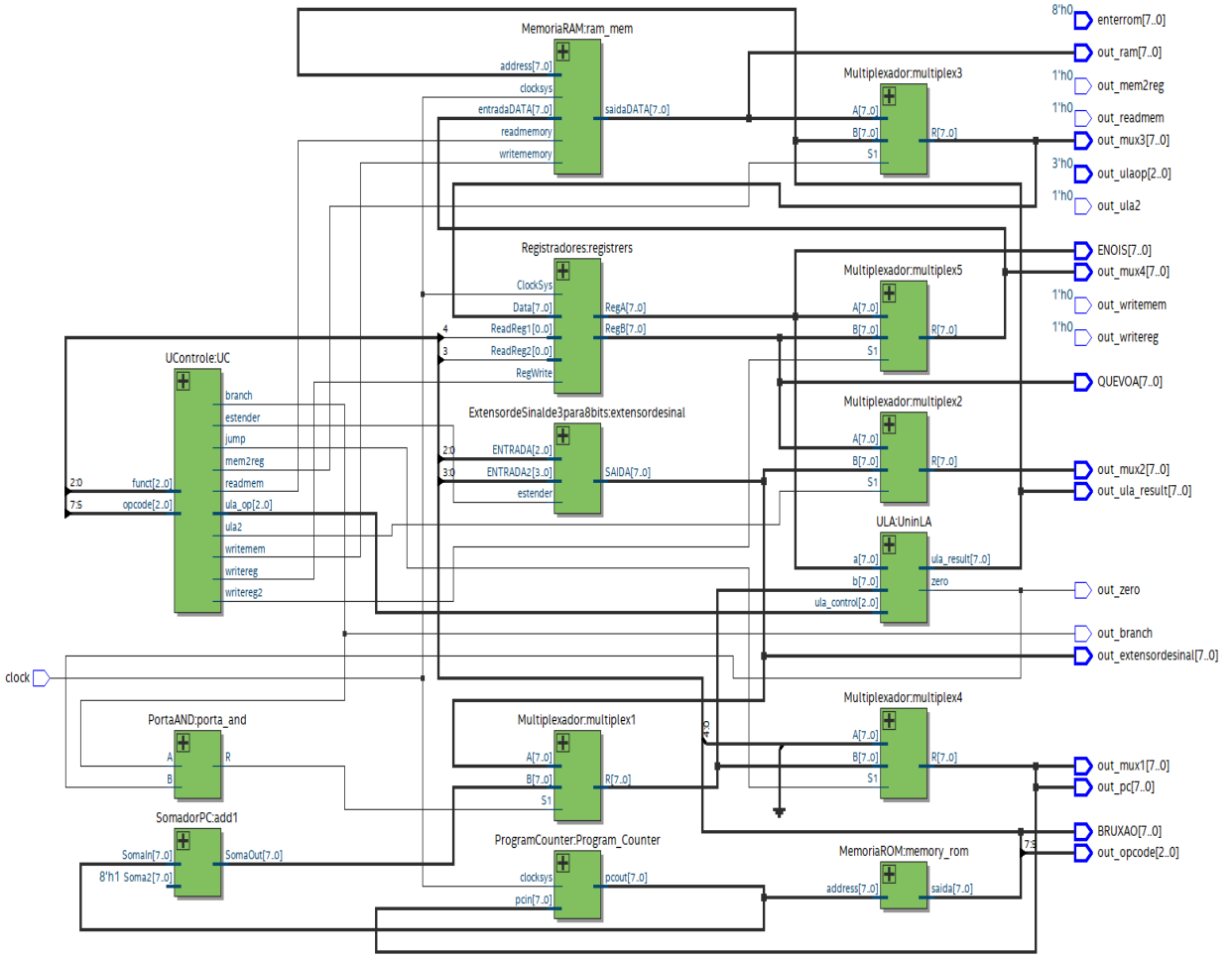


Figura 10 - LeiDeBug8BitsProcessador (RTL View)

1.5 Datapath

É a conexão entre as unidades funcionais formando um único caminho de dados e adicionando uma unidade de controle responsável pelo gerenciamento das ações que serão realizadas para diferentes classes de instruções.

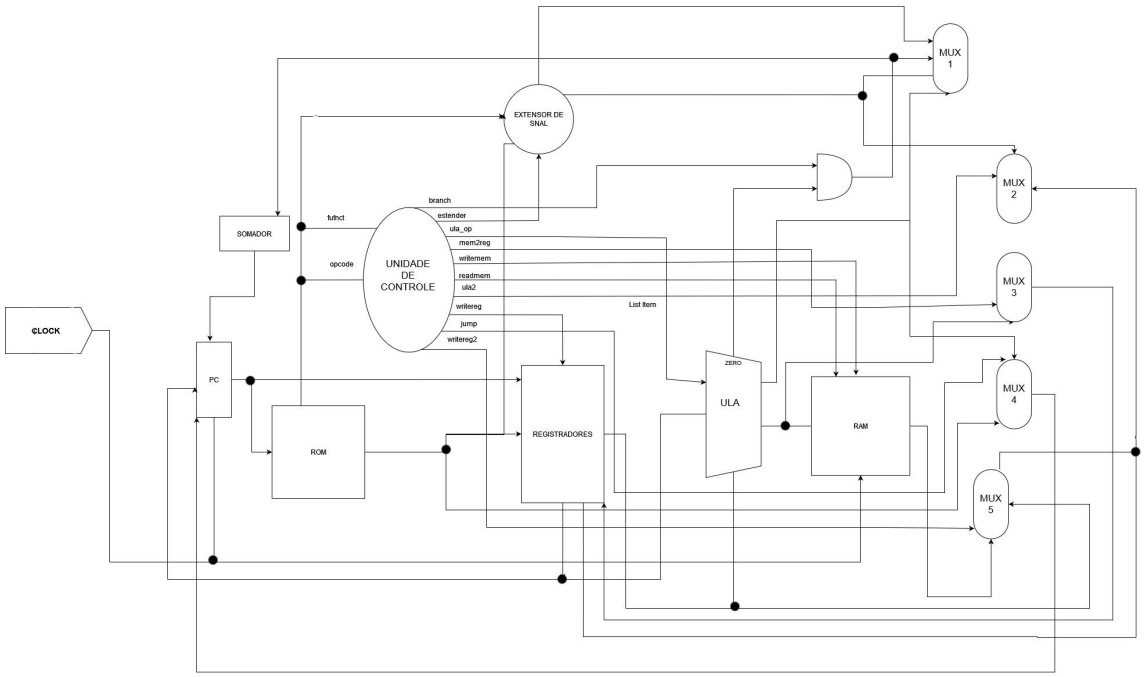


Figura 11 - DATAPATH

2. Simulações e Testes

Este trabalho apresentou o projeto e implementação do processador uniciclo de 8 bits referente ao projeto final de disciplina de Arquitetura e Organização de computadores. O processador contém todos os requisitos para que tenha essa alcunha pois contém operações aritméticas, memória de dados, condicionais, controle de dados e afins.

Figura12 - Teste 1 “Fibonacci”

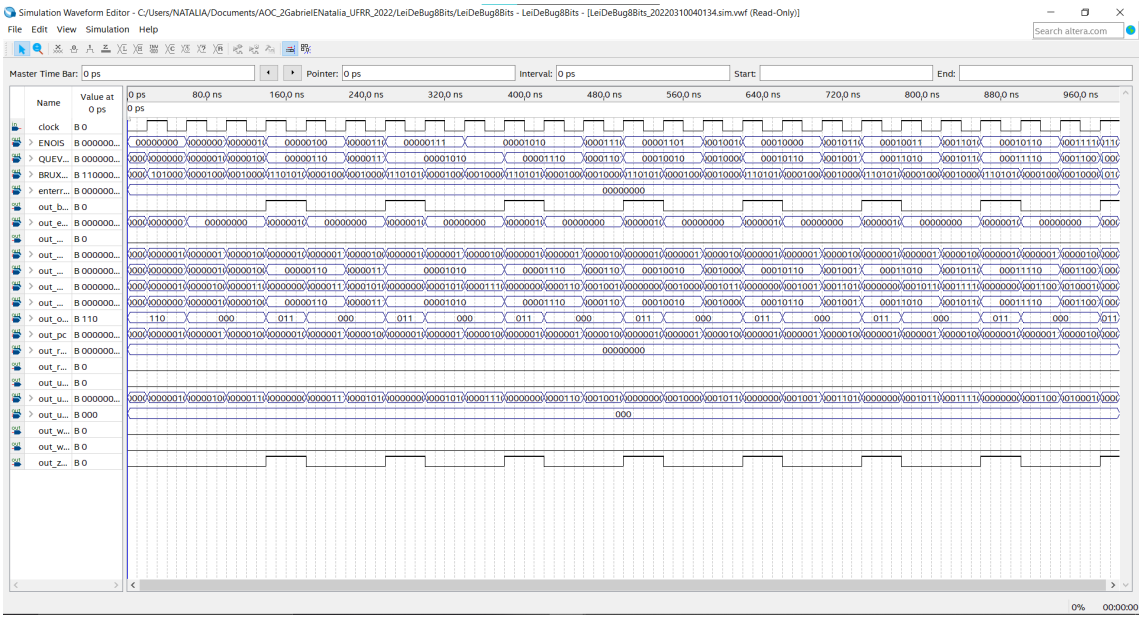
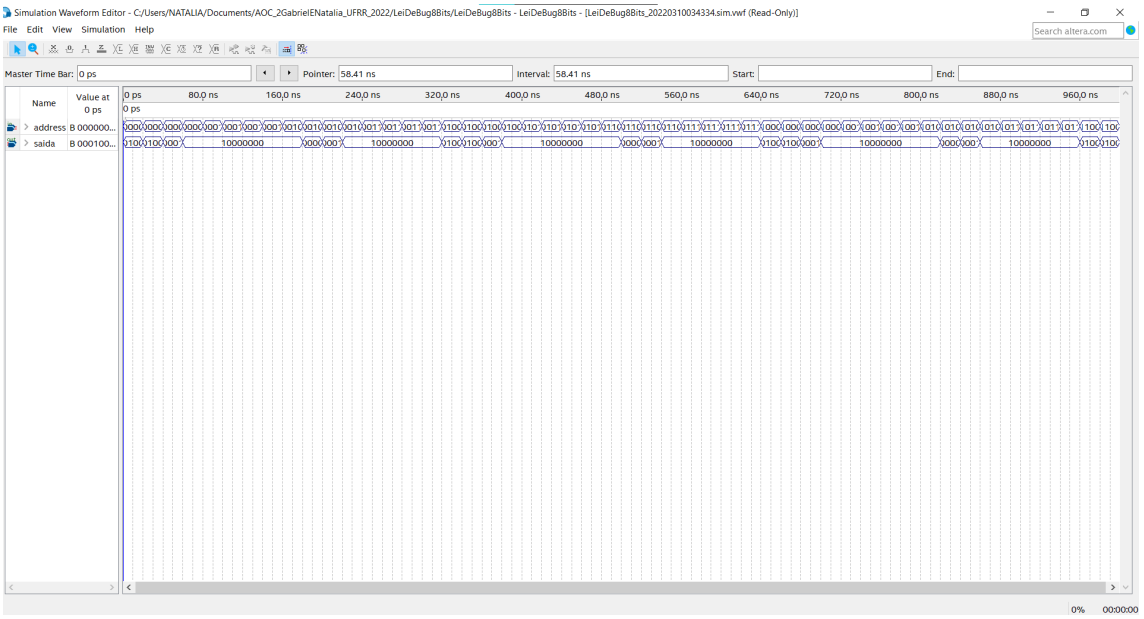


Figura 13 - Teste 2 add e addi na Memória ROM



3. Considerações finais

Este trabalho apresentou o projeto e implementação do processador uniciclo de 8 bits referente ao projeto final de disciplina de Arquitetura e Organização de computadores. O processador contém todos os requisitos para que tenha essa alcunha pois contém operações aritméticas, memória de dados, condicionais, controle de dados e afins.

4. Referências Bibliográficas

- https://www.fpga4student.com/2016/12/a-complete-8-bit-microcontroller-in-vhdl.html
- http://embeddedsystems.io/ahmes-um-processador-simples-de-8-bits-em-vhdl/
- https://github.com/jirawatsaesu/8-Bit-Simple-Processor
- http://web.archive.org/web/20040618011640/www.geocities.com/leon_heller/cpu.html
- https://youtu.be/jrtj3GKLKbo