



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

**ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES**

**RELATÓRIO DO PROJETO: AGAMOTTO 16 BITS**

**ALUNOS:**

**Bruno Cesar Da Silva Claudino - 1201424425**

**Francys Nutefe Tsigbey - 201514401**

**Novembro de 2018  
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: AGAMOTTO 16 BITS**

**Novembro de 2018  
Boa Vista/Roraima**

## Resumo

Este trabalho aborda o desenvolvimento de um processador de 16 bits para a disciplina de Arquitetura e Organização de Computadores. O processador foi implementado usando a linguagem de descrição de hardware VHDL. E os testes executados foram feitos utilizando-se waveforms geradas que simulam o comportamento do hardware descrito, estes mostram um pouco do funcionamento e performance do processador.

# Sumário

<b>1</b>	<b>Especificação</b>	<b>6</b>
1.1	Plataforma de Desenvolvimento . . . . .	6
1.2	Conjunto de instruções . . . . .	6
1.3	Descrição do hardware . . . . .	7
1.3.1	ULA X16 . . . . .	8
1.3.2	Banco de registradores . . . . .	8
1.3.3	Extensor de sinal 8x16 . . . . .	8
1.3.4	Unidade de Controle . . . . .	9
1.3.5	Memória RAM . . . . .	9
1.3.6	Memória ROM . . . . .	10
1.3.7	Multiplexador 2x1 . . . . .	11
1.3.8	Multiplexador 3x1 . . . . .	12
1.3.9	Multiplexador 4x1 . . . . .	13
1.3.10	PC(Program Counter) . . . . .	13
1.3.11	Registradores A e B . . . . .	14
1.3.12	Registrador de dados da memória . . . . .	16
1.3.13	Registrador de Instruções . . . . .	18
1.3.14	Registrador de saída da ULA . . . . .	19
1.4	Datapath . . . . .	21
<b>2</b>	<b>Simulações e Testes</b>	<b>21</b>
<b>3</b>	<b>Considerações Finais</b>	<b>21</b>

## Lista de Figuras

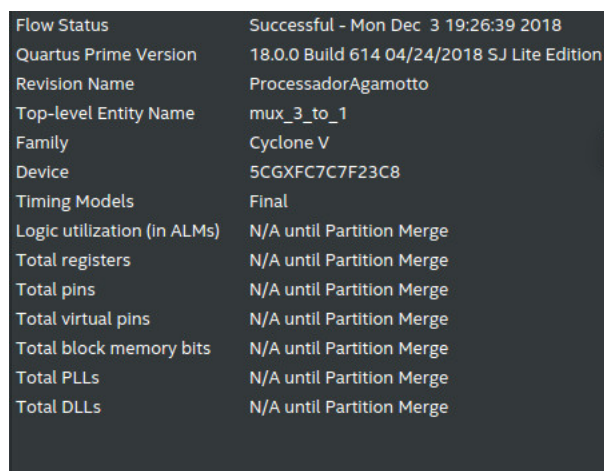
1	Plataforma de implementação. . . . .	6
2	Extensor de sinal de 8 bits para 16 bits. . . . .	9
3	Memória RAM . . . . .	10
4	Multiplexador 2x1 . . . . .	12
5	Multiplexador 3x1 . . . . .	13
6	Program Counter . . . . .	14
7	Registrador A. . . . .	15
8	Registrador B. . . . .	16
9	Registrador de dados de memória. . . . .	17
10	Waveform do Registrador de dados de memória. . . . .	18
11	Registrador instruções. . . . .	19
12	Registrador de saída da ULA. . . . .	20
13	Waveform do Registrador de saída da ULA. . . . .	21

# 1 Especificação

O processador Agamotto 16 bits foi desenvolvido para a disciplina de arquitetura e organização de computadores da Universidade Federal de Roraima no semestre 2018.2. O Processador é uniciclo, ou seja, executa uma única instrução por vez a cada ciclo de clock e as mesmas possuem 16 bits. Ele foi baseado na arquitetura MIPS, uma arquitetura de microprocessadores RISC desenvolvida pela MIPS Computer Systems.

## 1.1 Plataforma de Desenvolvimento

Para a implementação do processador Agamotto foi utilizado a IDE: Quartus Prime Versão 18.0 Lite Edition.



Flow Status	Successful - Mon Dec 3 19:26:39 2018
Quartus Prime Version	18.0.0 Build 614 04/24/2018 SJ Lite Edition
Revision Name	ProcessadorAgamotto
Top-level Entity Name	mux_3_to_1
Family	Cyclone V
Device	5CGXFC7C7F23C8
Timing Models	Final
Logic utilization (in ALMs)	N/A until Partition Merge
Total registers	N/A until Partition Merge
Total pins	N/A until Partition Merge
Total virtual pins	N/A until Partition Merge
Total block memory bits	N/A until Partition Merge
Total PLLs	N/A until Partition Merge
Total DLLs	N/A until Partition Merge

Figura 1: Plataforma de implementação.

## 1.2 Conjunto de instruções

O processador Agamotto possui 4 registradores: *Zero*, *s0*, *s1*, *s2*. Assim como 3 formatos de instruções de 16 bits cada. Instruções do tipo R, I, J seguem algumas considerações sobre as estruturas contidas nas instruções:

- **Opcode:** A operação básica a ser executada pelo processador, tradicionalmente chamado de código de operação;
- **Reg1:** O registrador contendo o primeiro operando fonte e adicionalmente para alguns tipos de instruções (ex. instruções do tipo R) é o registrador de destino;
- **Reg2:** O registrador contendo o segundo operando fonte;

Tipos de instruções:

- **Formato do tipo R:** Abrange instruções de operações aritméticas, tais como: soma, soma de imediatos, subtração e etc. E a divisão de bits é descrita a seguir:  
Formato para escrita de código:

Tipo de instrução	Reg1	Reg2
-------------------	------	------

Formato para escrita em código binário:

4 bits	6 bits	6 bits
15-12	11-6	5-0
Opcode	Reg1	Reg2

- **Formato do tipo I:** Abrange instruções de operações de carregamento e gravação de dados da memória, tais como: Load e Store. E a divisão de bits é descrita a seguir:

4 bits	4 bits	4 bits	4 bits
15-12 bits	11-8	7-4	3-0
Opcode	Reg1	Reg2	ADDRESS

- **Formato do tipo J:** Abrange instruções de operações de JUMP. E a divisão de bits é descrita a seguir:

4 bits	12 bits
15-12 bits	11-0
Opcode	Reg1

### Descrição geral das instruções do processador Agamotto:

O campo de Opcode de cada instrução é de 4 bits, logo temos 15 opcodes disponíveis para identificação de instruções na UC,  $2^4 - 1 = 15$ .

Opcode	Nome	Formato	Breve Descrição	Exemplo
0000	ADD	R	Soma	<b>add</b> \$S0,\$S1 , ou seja, \$S0 := \$S0+\$S1
0001	ADDI	R	Soma Imediata	<b>addi</b> \$S0,1 , ou seja, \$S0 := \$S0+1
0010	SUB	R	Subtração	<b>sub</b> \$S0,\$S1 , ou seja, \$S0 := \$S0-\$S1
0011	SUBI	R	Subtração Imediata	<b>sub</b> \$S0,1 , ou seja, \$S0 := \$S0-1
0100	MULT	R	Multiplicação	<b>mult</b> \$S0,\$S1 , ou seja, \$S0 := \$S0*\$S1
0101	BEQ	I	Branch Equal	<b>beq</b> \$S0,1
0110	BNE	I	Branch Not Equal	<b>bne</b> \$S0,1
0111	LW	I	Load	<b>lw</b> \$S0,\$S1
1000	SW	I	Store	<b>sw</b> \$S0,\$S1
1001	JUMP	J	Jump	<b>jump</b> L1

## 1.3 Descrição do hardware

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

### 1.3.1 ULA X16

O componente ULAX16 (Unidade Lógica Aritmética) tem como principal objetivo efetuar as principais operações aritméticas, dentre elas: soma, subtração, divisão e multiplicação. Adicionalmente o ULA X16 efetua operações de comparação de valor como maior ou igual, menor ou igual, somente maior, menor ou igual. O componente ULA X16 recebe como entrada três valores: A – dado de 16 bits para operação; B - dado de 16 bits para operação e UALOp – identificador da operação que será realizada de 4bits. A ULA X16 também possui três saídas: ZERO – identificador de resultado (1bit) para instruções de branch (1 se verdade e 0 caso contrário).

### 1.3.2 Banco de registradores

O componente Banco tem a função de ler e armazenar os valores em registradores sendo 4 no total. Sendo eles \$ZERO, \$S1, \$S2, \$S3, o \$ZERO tem um valor constante 0, usado para fazer comparações, e mover valores para outros registradores. Os registradores \$S1 ate \$s3 são para armazenamentos de valores calculados ou carregados da memória.

#### Sinais de entrada:

- **Clock:** Recebe o clock do sistema.
- **DadoEscrito(15..0):** Dado que sera escrito no registrador.
- **EscReg:**Flag de sinal que aciona a escrita no registrador.
- **RegEscrito1(1..0):**Recebe o endereço do registrador a ser escrito.
- **RegLido1(1..0):**Recebe o endereço do registrador a ser lido.
- **RegLido2(1..0):**Recebe o endereço do registrador a ser lido.

#### Sinais de saída:

- **RegLido1(15..0):**Recebe o valor lido para do primeiro registrador.
- **RegLido2(15..0):**Recebe o valor lido do segundo registrador.

### 1.3.3 Extensor de sinal 8x16

O componente extensor de sinal tem como funcao fazer a extensão de bits necessários completando os bits mais significados com 0's. No caso esse extensor é de 8 bits para 16 bits.

#### Sinais de entrada:

- **Entrada(7..0):** Recebe o valor a ser extendido o sinal.

#### Sinais de saída:

- **Saida(15..0):**Recebe o valor com sinal extendido de 8 para 16 bits.



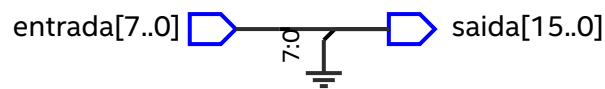


Figura 2: Extensor de sinal de 8 bits para 16 bits.

#### 1.3.4 Unidade de Controle

O componente (Ucontrol), tem como função o controle da execução de instruções através da ativação das flags necessárias em cada etapa de execução. É ele quem gerencia e controla todo o caminho de dados do Processador.

#### 1.3.5 Memória RAM

O componente Memória RAM é uma memória de Dados, onde está localizada todos os operandos que podem ser utilizados por instruções. A memória RAM pode ser utilizada por instruções de Load ou Store. **Sinais de entrada:**

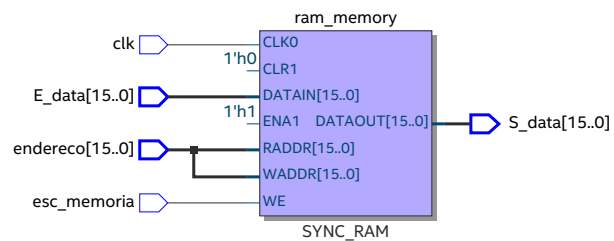
- **Clock:** Recebe o clock do sistema.
- **endereco(15..0):** Recebe o endereço onde encontra-se o operando.
- **Edata(15..0):** Recebe o dado a ser colocado na memória.
- **escmemoria:** Flag que ativa a escrita na memória.

Sinais de saída:

- **Sdata(15..0):** Recebe a saída com o valor operando buscado na memória.

Date: December 03, 2018

Project: ProcessadorAgamotto



Page 1 of 1

Revision: ProcessadorAgamotto

Figura 3: Memória RAM

### 1.3.6 Memória ROM

O componente Memória ROM é utilizada para armazenar as instruções a serem executadas.

**Sinais de entrada:**

- **endereco(15..0):** Recebe o endereço da instrução.

**Sinais de saída:**

- **data(15..0):** Saída com a instrução buscada em memória. Recebe a saída com o valor operando buscado na memória.

### 1.3.7 Multiplexador 2x1

O componente multiplexador serve como um IF ou seja, dependendo do seletor ele seleciona determinada operação a ser executada dentre as possíveis.

**Sinais de entrada:**

- **A(15..0):** Recebe a primeira opção que é selecionada caso o seletor seja 0.
- **B(15..0):** Recebe a segunda opção que é selecionada caso o seletor seja 1.
- **Seletor:** recebe da unidade de controle um valor para ser selecionado.

**Sinais de saída:**

- **Saida(15..0):** Recebe o valor da opção selecionada.

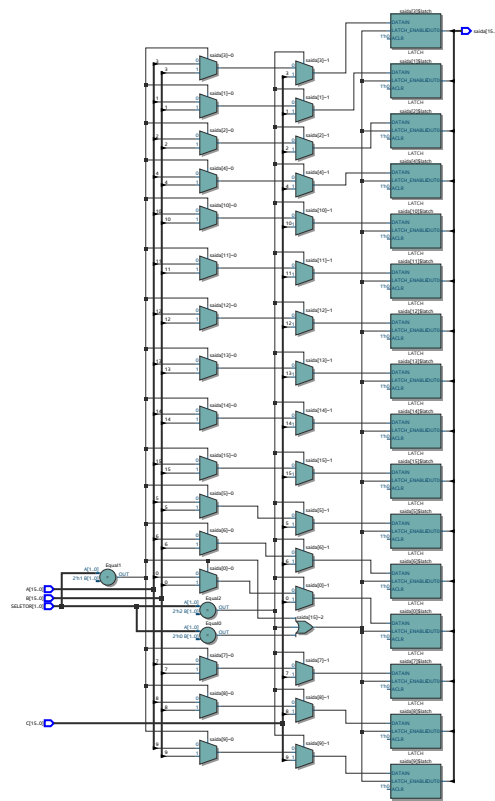


Figura 4: Multiplexador 2x1

### 1.3.8 Multiplexador 3x1

Sinais de entrada:

- **A(15..0):** Recebe a primeira opção que é selecionada caso o seletor seja 00.
- **B(15..0):** Recebe a segunda opção que é selecionada caso o seletor seja 01.
- **C(15..0):** Recebe a terceira opção que é selecionada caso o seletor seja 10.
- **Seletor(1..0):** recebe da unidade de controle um valor para ser selecionado.

Sinais de saída:

- **Saida(15..0):** Recebe o valor da opção selecionada.

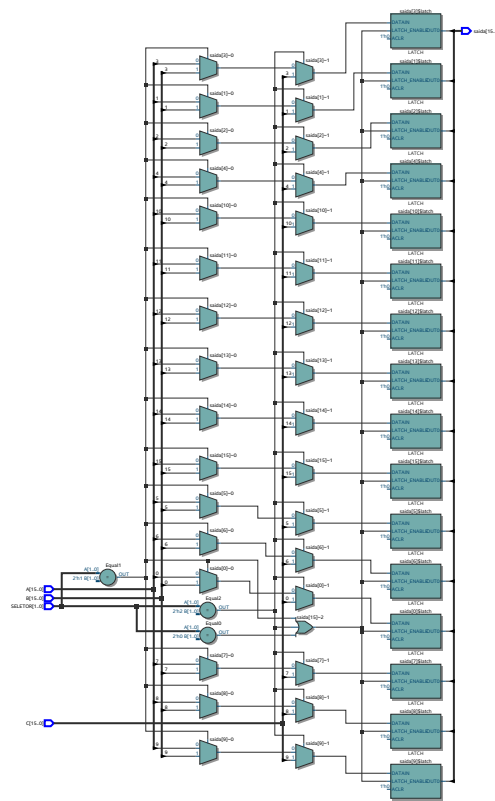


Figura 5: Multiplexador 3x1

### 1.3.9 Multiplexador 4x1

Sinais de entrada:

- **A(15..0):** Recebe a primeira opção que é selecionada caso o seletor seja 00.
- **B(15..0):** Recebe a segunda opção que é selecionada caso o seletor seja 01.
- **C(15..0):** Recebe a terceira opção que é selecionada caso o seletor seja 10.
- **D(15..0):** Recebe a terceira opção que é selecionada caso o seletor seja 11.
- **Seletor(1..0):** recebe da unidade de controle um valor para ser selecionado.

Sinais de saída:

- **Saida(15..0):** Recebe o valor da opção selecionada.

### 1.3.10 PC(Program Counter)

O componente PC é responsável por armazenar o endereço da instrução que está em execução. **Sinais de entrada:**

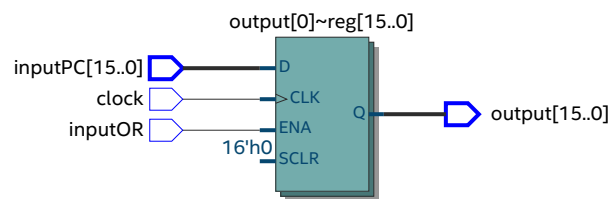
- **Clock:** Recebe o clock de sistema.
- **InputOR:** Flag de ativação do PC, sendo recebido da unidade de controle.
- **InputPC(15..0):** O endereço da próxima instrução a ser executada.

Sinais de saída:

- **Output(15..0):** Recebe o endereço instrução atual.

Date: November 29, 2018

Project: ProcessadorAgamotto



Page 1 of 1

Revision: ProcessadorAgamotto

Figura 6: Program Counter

### 1.3.11 Registradores A e B

O componente Registrador A e B servem para armazenar o valores que foram lidos do banco de registradores. E garantem que eles não irão se perder com a mudança do Clock.

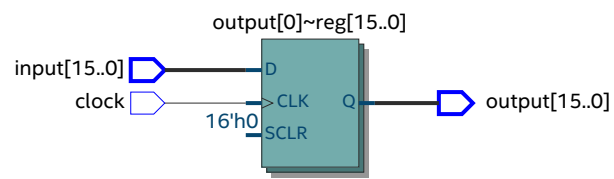


Figura 7: Registrador A.

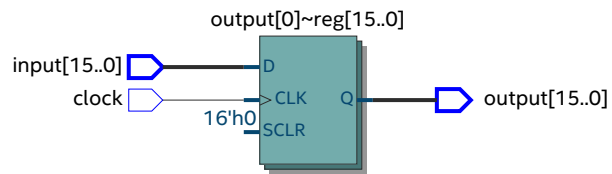


Figura 8: Registrador B.

### 1.3.12 Registrador de dados da memória

O registrador de dados da memória registra o valor do operando buscado em memória e garante que o mesmo não irá se perder na mudança de ciclo de clock. **Sinais de entrada:**

- **Clock:** Recebe o clock de sistema.
- **Input(15..0):** Recebe o dado que foi buscado na memória de dados.

**Sinais de saída:**

- **Output(15..0):** Saída do valor lido da memória.



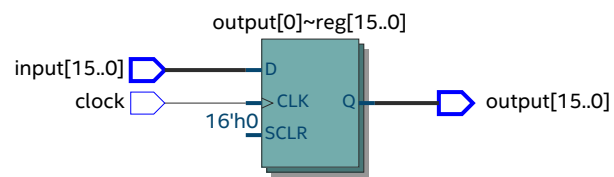


Figura 9: Registrador de dados de memória.

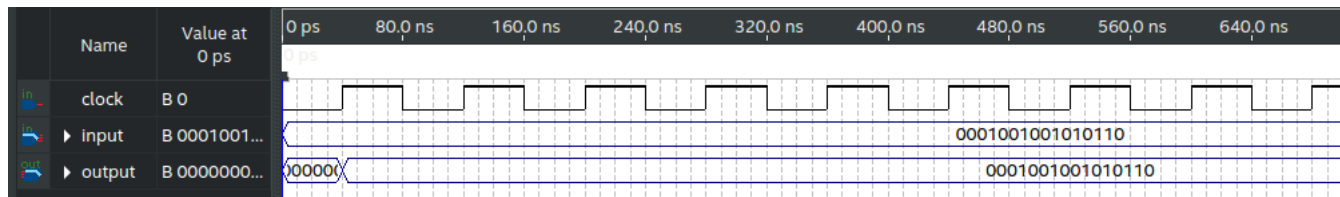


Figura 10: Waveform do Registrador de dados de memória.

### 1.3.13 Registrador de Instruções

O componente registrador de instruções tem como objetivo salvar a instrução e decodificar a mesma, em opcode, registrador1 e registrador2.

- **Clock:** Recebe o clock de sistema.
- **Input(15..0):** Recebe a instrução vinda da memória de instruções.
- **Sinal:** Flag que ativa a decodificação da instrução no registrador.

Sinais de saída:

- **output\_OPCODE(3..0):** Recebe o opcode para enviar a unidade de controle.
- **output\_r1:** Recebe o endereço do registrador1.
- **output\_r2:** Recebe o endereço do registrador2.

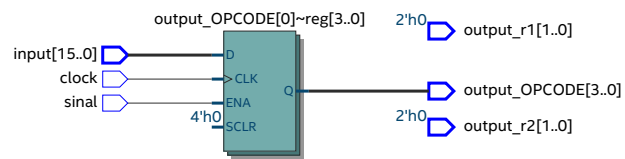


Figura 11: Registrador instruções.

#### 1.3.14 Registrador de saída da ULA

O componente registrador saída da ULA recebe o resultado da ULA.

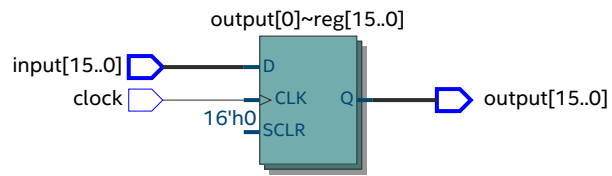


Figura 12: Registrador de saída da ULA.

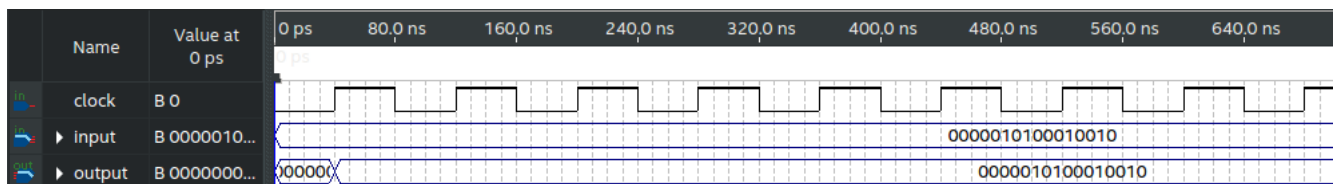


Figura 13: Waveform do Registrador de saída da ULA.

## 1.4 Datapath

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

## 2 Simulações e Testes

O processador Agamotto foi implementado e testado todos os componentes, porém a ultima fase de testes não pode ser realizada, portanto, não abordaremos o teste final do processador, ficando para um outro momento fazermos o teste final.

## 3 Considerações Finais

A execução deste projeto foi árdua, porém foi possível colocar em pratica os conhecimentos adquiridos ao longo das aulas lecionadas.