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

ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES

RELATÓRIO DO PROJETO: PROCESSADOR Qualcom-LI

ALUNOS:
IAN SANTOS NASCIMENTO - 2021000650
LUCAS ANDERSON LADISLAU AGUIAR - 2020005652

Dezembro 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 Qualcom-LI

Dezembro de 2022
Boa Vista/Roraima

Resumo

Este trabalho aborda o projeto e implementação de um processador de 8 bits, em arquitetura MIPS e RISC(Reduced Instruction Set Computer), denominado Qualcom-LI. O desenvolvimento do processador foi por meio da ferramenta Intel Quartus Prime Lite, para utilização dessa ferramenta foi necessária a programação de cada componente na linguagem VHDL(VeryHigh-Speed Integrated Circuit Hardware Description Language), os testes do processador foram realizadas por meio de WaveForms, permitindo a visibilidade das entradas e das saídas, e conhecimento de Assembly MIPS para serem gerados os testes corretamente.

Palavras-chave: MIPS. RISC. Quartus Prime Lite. VHDL.

Conteúdo

Lista de Figuras

Lista de Tabelas

1 Especificação

Nesta seção é apresentado o conjunto de itens para o desenvolvimento do processador Qualcom-LI, bem como a descrição detalhada de cada etapa da construção do processador. A divisão dos tópicos é feita para um melhor detalhamento do processador e seus componentes.

1.1 Plataforma de desenvolvimento

Para a implementação do processador Qualcom-LI foi utilizado a IDE: **Intel Quartus Prime Lite,** versão 20.1

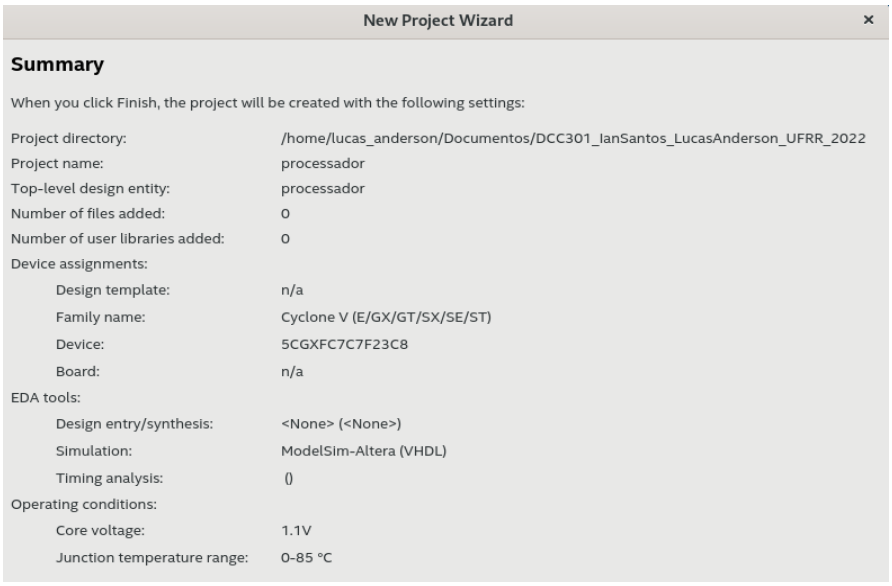

Figura 1 - Especificações do Projeto

Flow Status	Successful - Tue Dec 13 03:17:46 2022
Quartus Prime Version	20.1.0 Build 711 06/05/2020 SJ Lite Edition
Revision Name	processador
Top-level Entity Name	Qualcom_LI
Family	Cyclone V
Device	5CGXFC7C7F23C8
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	40
Total pins	69
Total virtual pins	0
Total block memory bits	32
Total DSP Blocks	1
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0

Figura 2 - Especificações da compilação Processador

1.2 Conjunto de instruções

O processador Qualcom-LI possui quatro(4) registradores: $S0(00), $S1(01), $S2(10) e $S3(11). Assim como três(3) formatos de instruções de 8 bits cada, instruções do **tipo J, R e I**, 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;

Tipo de Instruções:

- Formato do tipo R: Este formato aborda instruções baseadas em operações aritméticas.

Formato para escrita de código:

Tipo Instrução	Reg 1	Reg 2

Tabela 1 - Formatação de Instruções do Tipo R.

Formato para escrita em código binário:

OPCODE	REG 1	REG 2
4 bits	2 bits	2 bits
7 - 4	3 - 2	1 - 0

Tabela 1 - Formatação binária do Tipo R.

- Formato do tipo J: Este formato aborda instruções de salto em sua maioria.

Formato para escrita de código:

Tipo Instrução	Endereço

Tabela 2 - Formatação de Instruções do Tipo J.

Formato para escrita em código binário:

OPCODE	ENDEREÇO
4 bits	4 bits
7 - 4	3-0

Tabela 2 - Formatação binária do Tipo J.

- Formato do tipo I: Este formato aborda instruções que manipulam ou usam valores imediatos, como addi, subi…

Formato para escrita de código:

Tipo Instrução	Reg 1	Reg 2

Tabela 3 - Formatação de Instruções do Tipo I.

Formato para escrita em código binário:

OPCODE	REG 1	REG 2
4 bits	2 bits	2 bits
7 - 4	3 - 2	1 - 0

Tabela 3 - Formatação binária do Tipo I.

Visão geral das instruções do Processador Qualcom-LI:

O número de bits do campo **Opcode** das instruções é igual a quatro, sendo assim obtemos um total $(Bit(0e1)^4 \therefore 2^4 = 16)$ de 16 **Opcodes (0-15)** que são distribuídos entre as instruções, assim como é apresentado na Tabela 1.

Tabela 4 – Tabela que mostra a lista de Opcodes utilizadas pelo processador Qualcom-LI.

Nome	Formato	Instrução	Opcode
add	Tipo R	Soma	0000
addi	Tipo I	Soma com imediato	0001
sub	Tipo R	Subtração	0010
subi	Tipo I	Subtração imediato	0011
mult	Tipo R	Multiplicação	0100
lw	Tipo I	load word	0101
sw	Tipo I	save word	0110
move	Tipo R	move	0111
li	Tipo I	load imediato	1000

beq	Tipo J	Desvio condicional	1001
beq	Tipo J	Desvio condicional	1010
if_op	Tipo J	Condição para desvio	1011
J	Tipo J	Jump	1100

1.3 Descrição do Hardware

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

1.3.1 ALU ou ULA

A ULA (Unidade Lógica Aritmética) é um componente fundamental para a lógica de processamento, onde sua função é definir qual processo será realizado e as operações executadas. A ULA tem como principal objetivo efetuar as principais operações aritméticas, dentre elas: soma, subtração, divisão (considerando apenas resultados inteiros) e multiplicação. Adicionalmente a ALU efetua operações de comparação de valor como maior ou igual, menor ou igual, somente maior, menor ou igual. O componente ULA recebe como entrada três valores: entrada1 e entrada2 – dado de 8bits para operação e OP – identificador da operação que será realizada de 4bits. A ULA também possui três saídas: zero – identificador de resultado (2bit) para comparações (1 se verdade e 0 caso contrário); overflow – identificador de overflow caso a operação exceda os 8bits; e result – saída com o resultado das operações aritméticas.

São duas entradas de 8 bits, uma entrada de 4 bits que representa a operação que será ativada e uma saída que representa o resultado da operação.

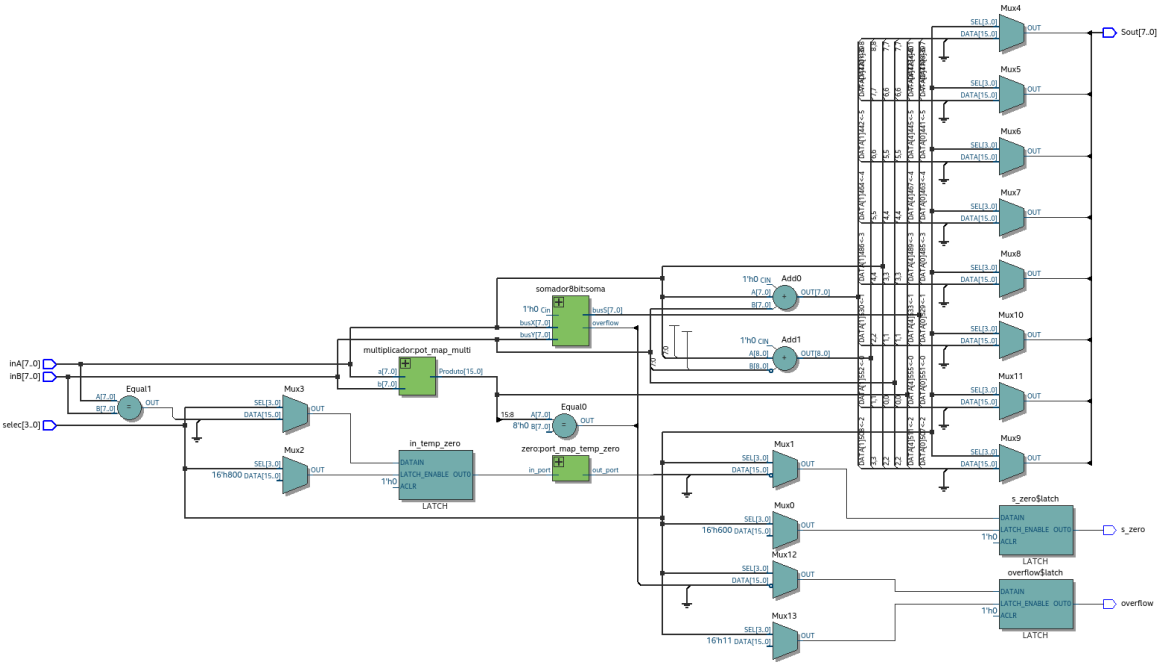

Figura 3 - Unidade de Lógica Aritmética(ULA)

1.3.2 Banco de Registradores

Banco de registradores é um componente que como o próprio nome sugere é formado por um conjunto de registradores que são acessados de forma organizada, os quais podem ser lidos e escritos nessa unidade.

No banco de registradores teremos 5 pinos entradas(1 para o clock, 1 para o dado que esta sendo inserido - escreveDado, 1 para saber se o dado deve ser escrito - escreveReg e 2 para o endereço do registrador) e 2 pinos de saída 1 que irá diretamente para a ULA e outro que será tratado posteriormente com um multiplexador para selecionar se ele será escolhido ou não.

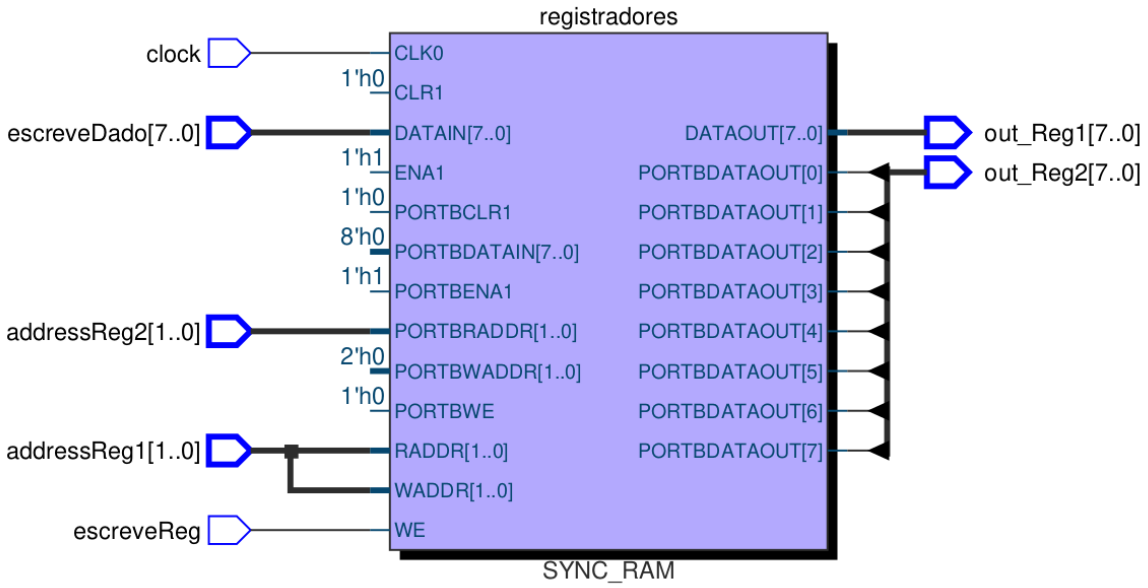

Figura 4 - Banco de Registradores

1.3.3 Unidade de Controle

A unidade de controle é o componente encarregado de alimentar o circuito com as flags das operações que serão executadas de acordo com o opcode inserido nela, controlando assim os sinais da CPU para o seu correto funcionamento.

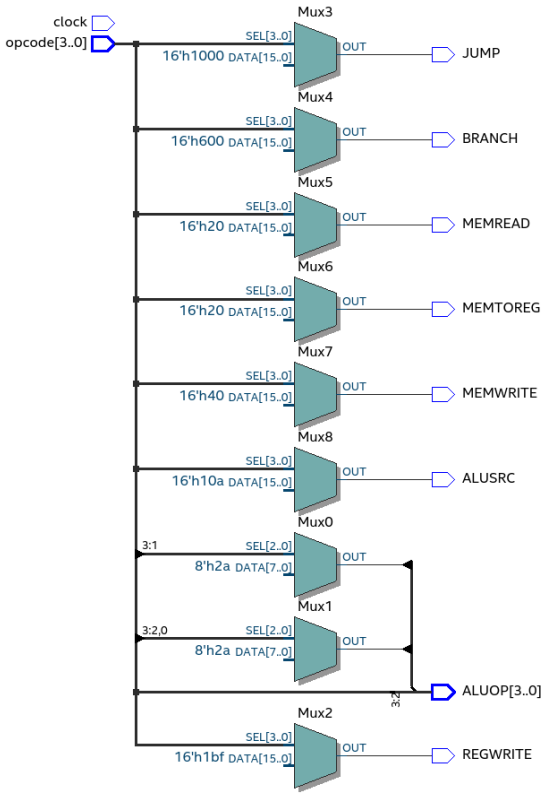

Figura 5 - Unidade de Controle

Essa alimentação é feito através das flags de saída abaixo:

- **Aluop**: 4 bits. (operação da ula)
- **RegWrite**: 1 bit (escrita do registrador)
- **Jump**: 1 bit (pulo incondional)
- **Branch**: 1 bit (Desvio e comparação)
- **MemRead**: 1 bit (Leitura na memória)
- **MemToReg**: 1 bit (Memória para registrador)
- **MemWrite:** 1 bit (escrita na memória)
- **AluSrc**: 1 bit (seletor para o multiplexador)

Abaixo segue a tabela, onde é feita a associação entre os opcodes e as flags de controle:

Tabela 5 - Detalhes das flags de controle do processador.

Comando	Aluop	RegWrite	Jump	Branch	MemRead	MemToReg	MemWrite	AlucSrc
add	0000	1	0	0	0	0	0	0
addi	0001	1	0	0	0	0	0	1
sub	0010	1	0	0	0	0	0	0
subi	0011	1	0	0	0	0	0	1
mult	0100	1	0	0	0	0	0	0
lw	0101	1	0	0	1	1	0	0
sw	0110	0	0	0	0	0	1	0
move	0111	1	0	0	0	0	0	0
li	1000	1	0	0	0	0	0	1
beq	1001	0	0	1	0	0	0	0
beq	1010	0	0	1	0	0	0	0
if_op	1011	0	0	0	0	0	0	0
J	1100	0	1	0	0	0	0	0

1.3.4 Memória de dados

Um espaço disponível para armazenamento de valores temporários, facilitando e agilizando o acesso para o processador, conhecida nos processadores como memória de dados.

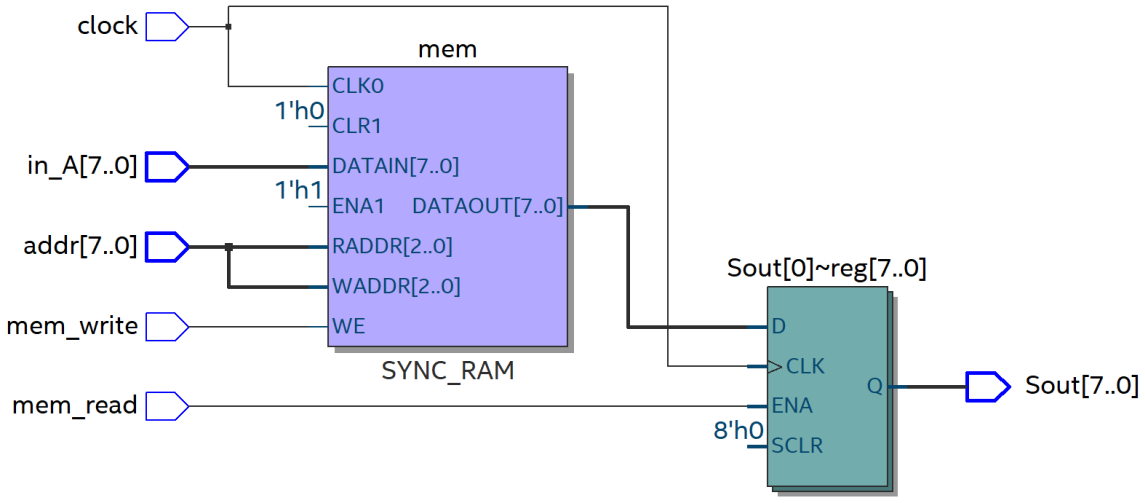

Figura 6 - Memória de Dados

1.3.5 Memória de Instruções

A memória rom é basicamente uma memória de leitura de dados que estão armazenados, na situação de componente do CI é nela que vão estar as instruções do processador, conhecida como memória instrução do datapath de um MIPS.

Na memória rom temos como pinos entrada de dados(o valor do endereço na memória) com um vetor de 8 bits, junto com o clock e a saída da instrução, também um vetor de 8 bits, que será executada.

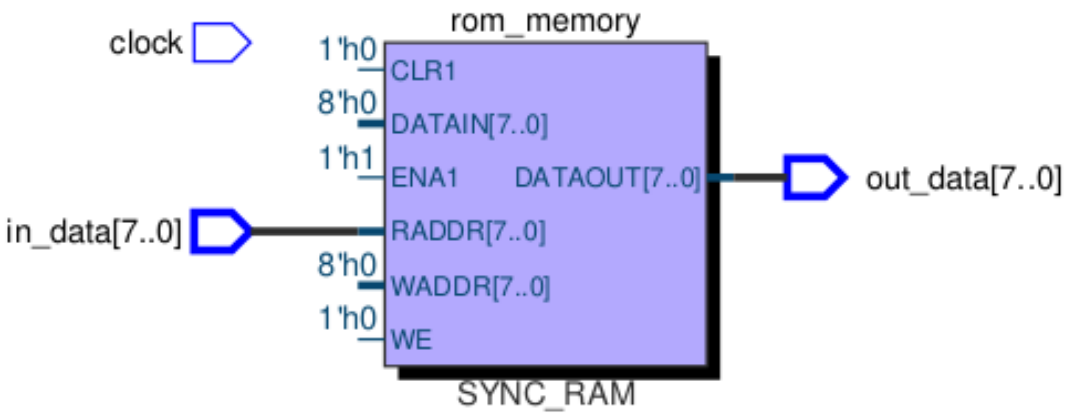

Figura 7 - Memória ROM

1.3.6 Somador

O somador de 1 bit é basicamente um componente que irá somar dois bits diferentes e resultará numa saída e num “vai um”, ou carryout.

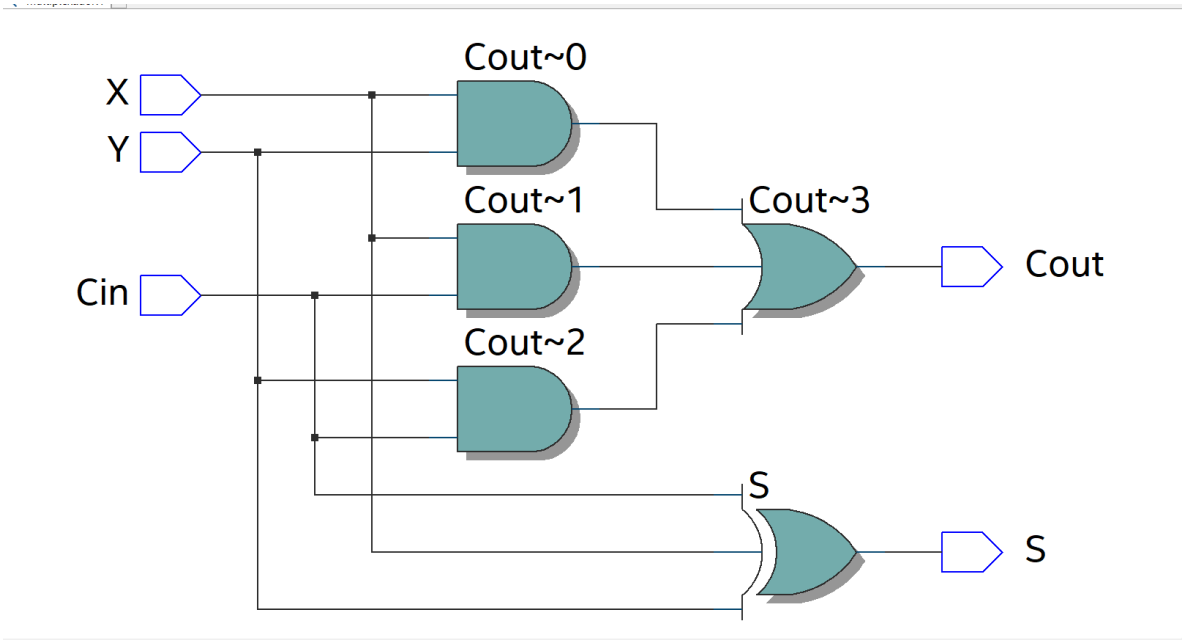

Figura 8- Célula do somador

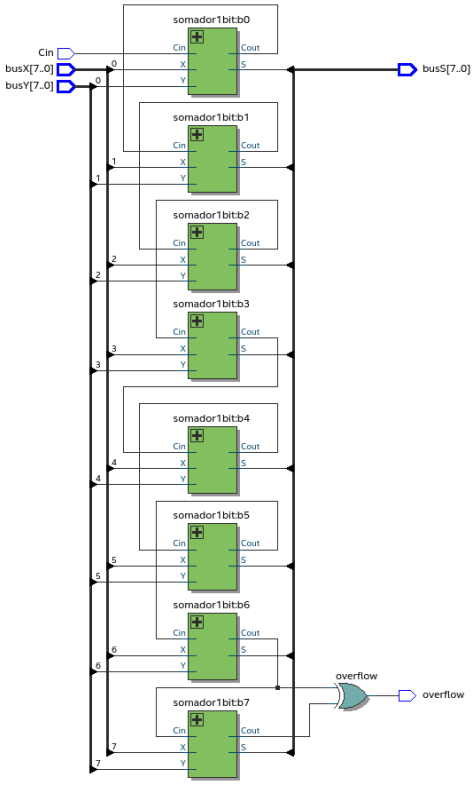

Figura 8 - Célula do somador

Circuito somador de 8 bits, utilizando a célula do somador de 1 bit, essa resulta num valor também de 8 bits e uma saída de overflow que indica se houve transbordamento da memória.

1.3.7 And

A porta AND é uma porta onde dois bits entram e a saída será “1” apenas quando ois dois bits de entrada estiverem com o valor “1”

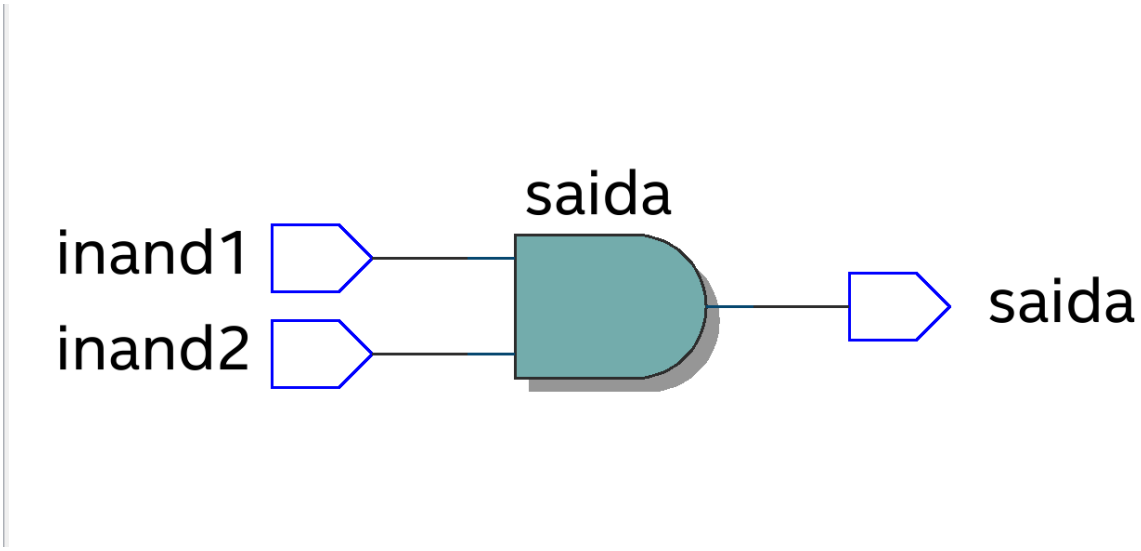

Figura 9- AND

1.3.8 Mux_2x1

Um multiplexador é basicamente um switch em que existem entradas de valores que serão escolhidas e uma entrada que irá definir qual valor será escolhido.

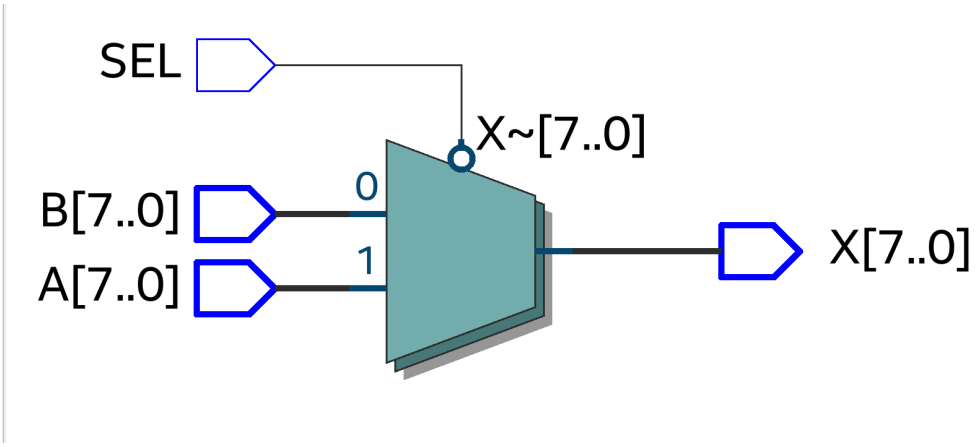

Figura 10 - Multiplexador

1.3.9 PC

Esse componente contém o endereço da instrução a ser executada, ele envia a informação para a memória de instrução que lê o endereço da instrução que se deseja e a retorna para o restante do circuito.

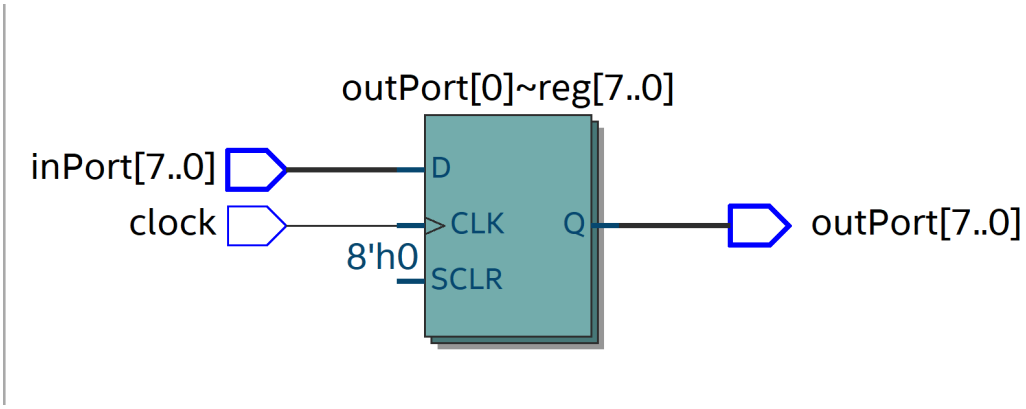

Figura 11 - PC

1.3.10 Extensor 2x8

Um extensor de sinal faz com que uma entrada de bits fique com uma largura maior que a anterior.

De acordo com a definição do extensor de sinal temos um pino de entrada, com um vetor de bits, e um pino de saída, com 8 bits. A extensão dos bits é feita com o acréscimo de bits menos significativos no vetor.

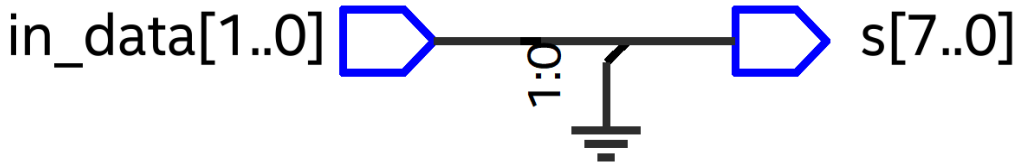

Figura 12 - Extensor 2x8

1.3.11 Contador Síncrono

Contador síncrono é um circuito digital formado por flip-flops em paralelos, tal que todas as entradas clocks estejam conectados na mesma fonte de clock

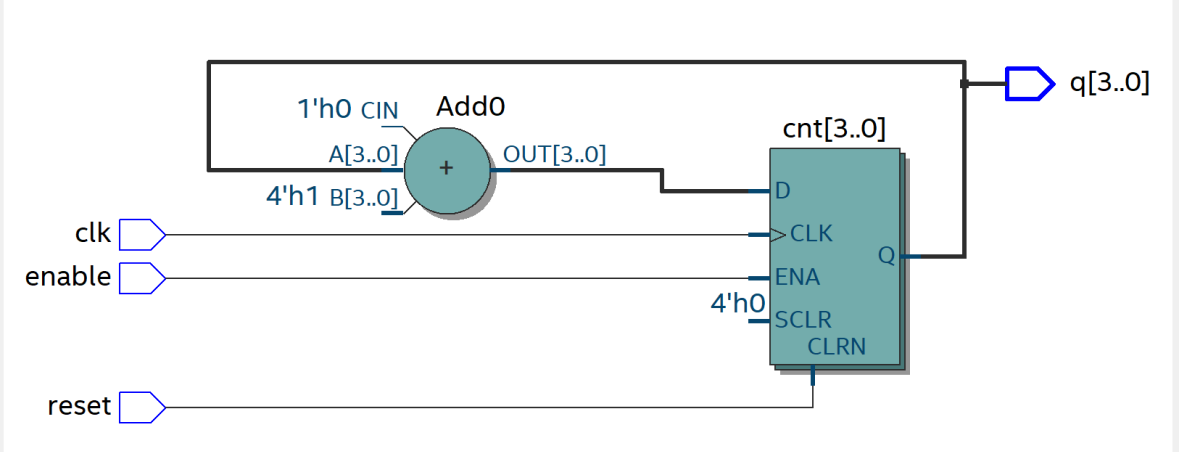

Figura 13 - Contador Síncrono

1.3.12 Extensor 4x8

De acordo com a definição do extensor de sinal temos um pino de entrada, com um vetor de 4 bits, e um pino de saída, com 8 bits. A extensão dos bits é feita com o acréscimo de bits menos significativos no vetor.

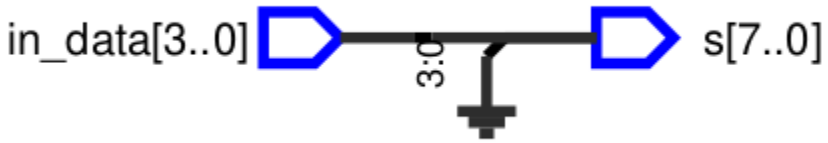

Figura 14- Extensor 4x8

1.4 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, essas interações através dos circuitos é que permite a correta comunicação dos componentes.

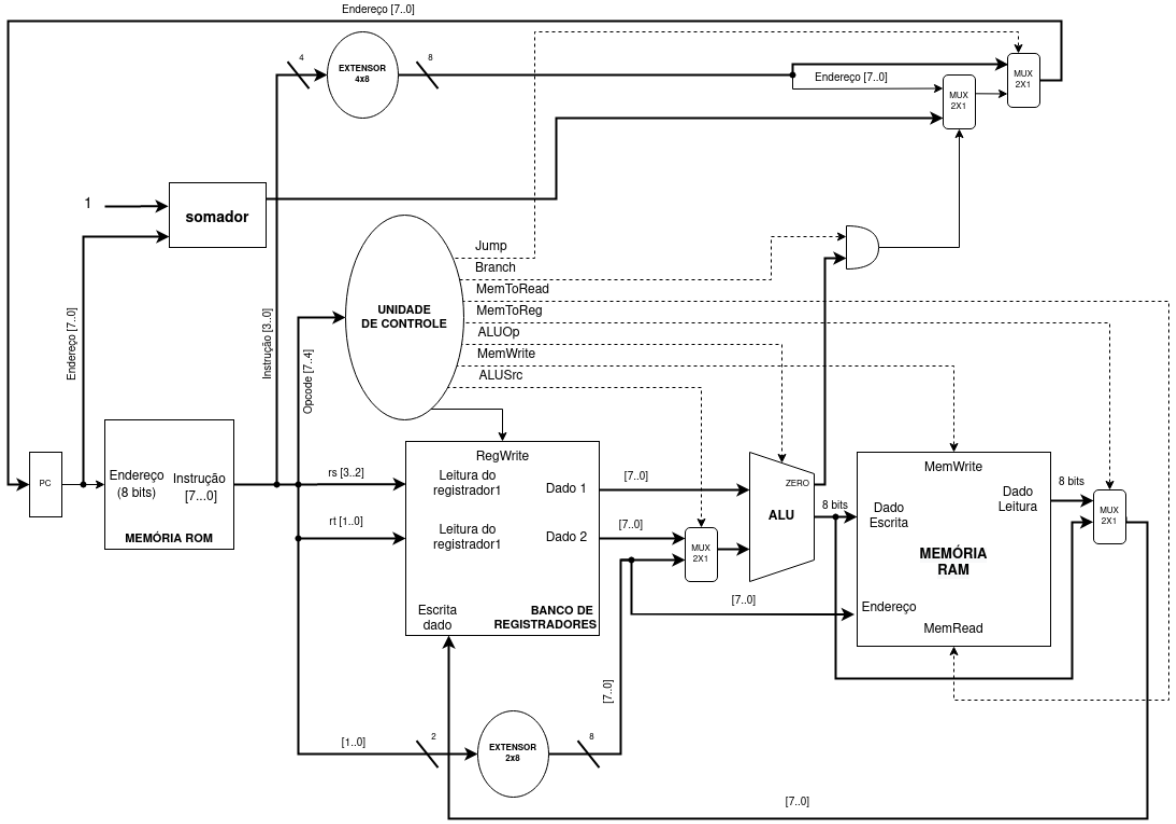

Figura 15- Diagrama do datapath

1.5 RTL_VIEWER

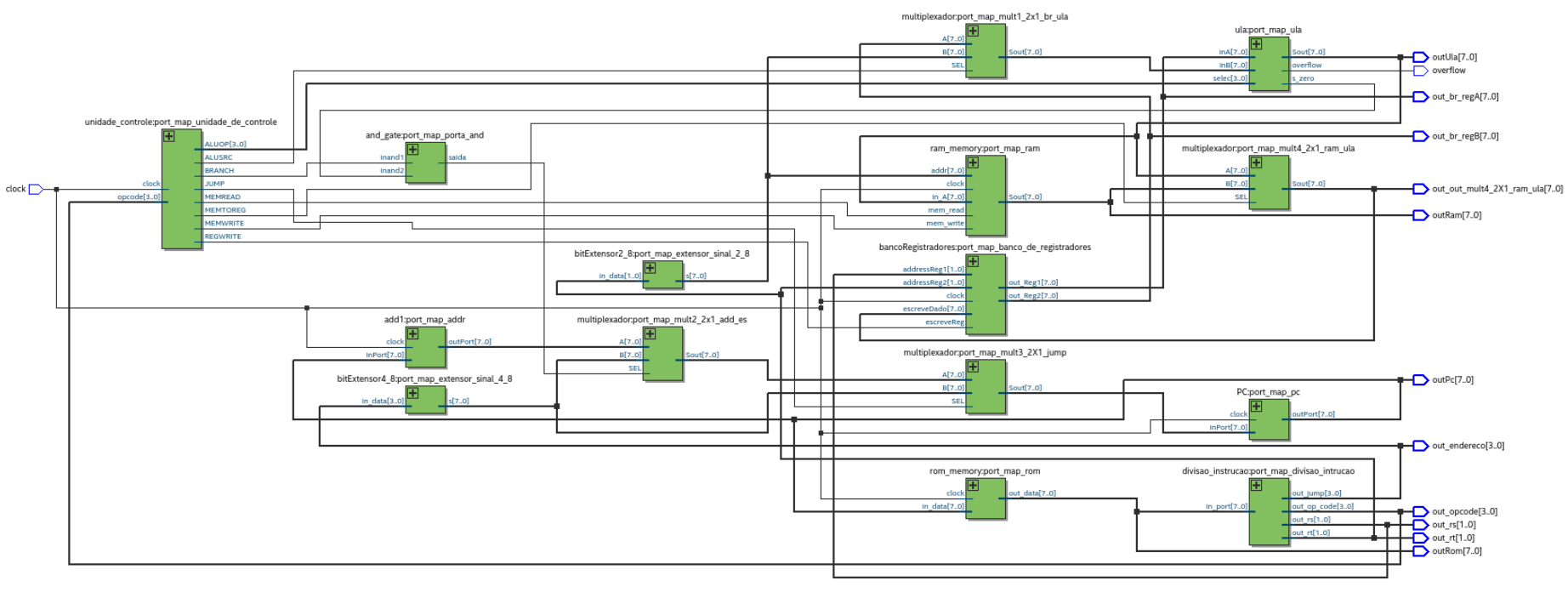

Figura 16- RTL viewer Qualcom-LI

Imagem do RTL_VIEWER de todos os componentes do processador integrador, onde fazemos uma conexão em cada unidade para ver o que se passa naquele componente a cada ciclo de clock.

2 Simulações e Testes

Objetivando analisar e verificar o funcionamento do processador, efetuamos alguns testes analisando cada componente do processador em específico, em seguida efetuamos testes das instruções que o processador implementa.

2.1 Teste de Fatorial

Para demonstrar o funcionamento do processador Qualcom-LI utilizaremos como exemplo o código para calcular o Fatorial.

Tabela 6 - Código Fatorial para o processador.

Endereço	Linguagem de Alto Nível	Binário		
		Opcode	Reg2	Reg1
			Endereço	
			Dado	
0	**LI** $S3, 3	1000	11	11
1	**ADDI** $S3, 3	0001	00	11
3	**SW** $S3, $S0	0110	00	11
4	**LI** $S2, 1	1000	00	01
6	**LI $**S0 1	1000	01	10
7	**MULTI** $S0, $S0	0100	00	11
9	**IF $S2 == $S3**	1011	00	00
10	**ADDI $S2 1**	0001	00	11
11	**BNE $S2,$S3, JUMP**	1010	10	11
12		0100		

Acima temos a instrução em Assembly MIPS e em binário para efetuar o teste de fatorial. Verificação dos resultados no relatório da simulação na Waveform: Após a compilação e execução da simulação, o seguinte relatório é exibido.

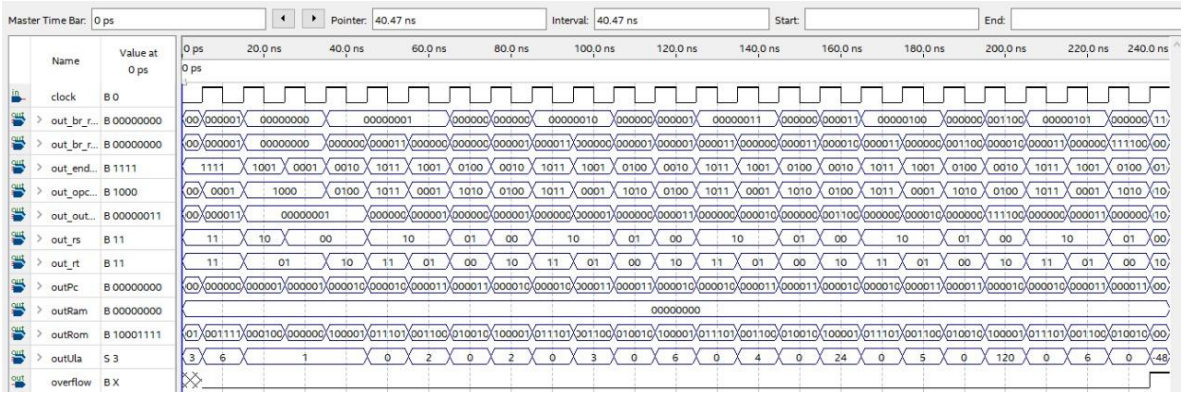

Figura 17 - Resultado na waveform fatorial.

Temos o teste com o fatorial, porém devido a limitação da arquitetura de 8 bits o fatorial máximo que pode ser gerado com êxito é até 5, gerando 120 como resultado, no

fatorial de 6 temos um overflow da memória, como podemos no penúltimo pino de saída gerando -48 como resultado, ou seja, a memória estourou, pois o fatorial de 6 gera um número binário que não é suportado em 8 bits, a flag de overflow então apresenta uma borda alta para indicar o overflow.

3 Considerações finais

Este trabalho apresentou o projeto e implementação do processador de 8 bits denominado de Qualcom-LI, os componentes estão integrados entre si, apesar da limitação da arquitetura de 8 bits, ainda foi possível efetuar o teste com o fatorial até o número 5. Pode-se entender um pouco o quão complexo é o processo de construção de processadores que são indispensáveis na modernidade, eles estão presentes em quase todos os meios eletrônicos, mas principalmente nos computadores. O quartus como ambiente de teste permitiu visualizar minuciosamente cada funcionamento desde as entradas até a saída, foi um projeto muito proveitoso de se desenvolver.