Especificação do Projeto

Processador LAPI DOpaCA LAMBA

Universidade Estadual de Feira de Santana

Build 2.1

Histórico de Revisões

Date	Descrição	Autor(s)
xx/xx/xxxx	<Descrição>	<Autor(es)>
08/12/2015	Teste	Filipe Boaventura
xx/xx/xxxx	• Exemplo de; • Revisões em lista;	<Autor(es)>

SUMÁRIO

1. Introducão

1.1. Propósito

O objetivo desse documento é definir as especificações para o processador **LAPI DOpaCA LAMBA**, incluindo o seu montador e simulador. Aqui são definidos os parametros de implementação do processador que incluem: modos de operação, arquitetura do conjunto de instruções (ISA), e a descrição dos registradoes e flags. Este documento está organizado da seguinte forma: Visão Geral na seção 2, Arquitetura do Conjunto de Instruções seção 3, Montador seção 4, Simulador seção 5.

1.2. Document Outline Description

Este documento está organizado da seguinte forma:

- Section 2: Apresenta uma visão geral do processador.
- Section 3: Especifica a arquitetura do conjunto de instruções.
- Section 4: Describes the general purpose registers.

1.3. Acronyms and Abbreviations

Acronym	Description
RISC	Reduced Instruction Set Computer
GPR	General Purpose Registers
FPGA	Field Gate Programmable Array
GPPU	General Purpose Processing Unit
SPRAM	Single-port RAM
HDL	Hardware Description Language
CPU	Central Processing Unit
ISA	Instruction Set Architecture
ALU	Arithmetic and Logic Unit
PC	Program Counter
RF	Flags Register
CST	Constant Value

2. Visão Geral

O LAPI DOpaCA LAMBA é um processador RISC que apresenta um conjunto de 42 instruções organizadas em X grupos funcionais e possiu três modos de endereçamento. O processador possui 16 registradores de propósito geral em seu banco de registradores. O LAPI DOpaCA LAMBA suporta operações lógicas e aritiméticas básicas, excerto multiplicação e divisão.

2.1. Perspectives

The MUSA 32-bit core release targets modern FPGA devices with embedded SPRAM memory, and is intended to be deployed as a GPPU soft IP-core. Its architecture is designed under MIPS original implementation, presenting a slightly reduced ISA.

2.2. Características Principais

- Arquitetura de 32 bits;
- 16 registradores de propósito geral;
- Arquitetura de conjunto de instruções (ISA) RISC;
- ISA composta por 42 instruções;
- Organização de dados Big-Endian;
- Três modos de endereçamento: immediato, por registrador, e base-deslocamento;

2.3. Non-functional Requirements

- The FPGA prototype should run in a Terasic ALTERA Cyclone III (EP3C25F324) Development platform;
- The design must be described using Verilog-HDL;
- The tests must be writen in both Verilog-HDL or SystemVerilog;
- A set of test programs must be provided in order to validate the implementation;

3. Conjunto de Instruções

O Processador LAPI DOpaCA LAMBA possui 42 instruções de 32 bits de largura que estão organizadas nos seguintes grupos funcionais:

- ALU;
- Constantes;
- Transferências de controle (jump);
- Memória (load e store);
- NOPs;
- Condição de término.

In MUSE load/store architecture, operations are performed through operands held into the register file (GPR Bank) and main memory is accessed only through load and store instructions. Signed and unsigned integers of 16-bits are supported by loads that either sign-extend or zero-extend the data loaded into the register.

3.1. CPU Load and Store Instructions

Mnemonic	Operands	Realization	Description
LW	$RD, RS1, I_{16}$	$RD \leftarrow MEM[RS1 + I_{16}]$	Load word from data memory.
SW	$RS1, RS2, I_{16}$	$MEM[RS1 + I_{16}] \leftarrow RS2$	Store word into data memory.

3.2. Computational Instructions

The MUSA provides 32-bit arithmetic operations. The computational instructions uses ALU two-operand instructions. These are signed versions of the following operations:

- Add
- Subtract
- Multiply
- Divide

Mnemonic	Operands	Realization	Description
ADD	$RD, RS1, RS2$	$RD \leftarrow RS1 + RS2$	Add two words.
SUB	$RD, RS1, RS2$	$RD \leftarrow RS1 - RS2$	Subtract two words.
MUL	$RD, RS1, RS2$	$RD \leftarrow RS1 * RS2$	Multiply two words.
DIV	$RD, RS1, RS2$	$RD \leftarrow RS1/RS2$	Divide two words.
AND	$RD, RS1, RS2$	$RD \leftarrow RS1 \odot RS2$	Logical AND.
OR	$RD, RS1$	$RD \leftarrow RS1 \oplus RS2$	Logical OR.
ADDI	$RD, RS1, I_{16}$	$RD \leftarrow RS1 + I_{16}$	Add a stored word and an immediate value.
SUBI	$RD, RS1, I_{16}$	$RD \leftarrow RS1 - I_{16}$	Subtract a stored word and an immediate value.
ANDI	$RD, RS1, I_{16}$	$RD \leftarrow RS1 \odot I_{16}$	Logical AND of a stored word and an immediate value
ORI	$RD, RS1, I_{16}$	$RD \leftarrow RS1 \oplus I_{16}$	Logical OR of a stored word and an immediate value.
CMP	$RD, RS1$	–	Compares RD and $RS1$ and set the flags.
NOT	RD	$RD \leftarrow \sim RD$	Logical NOT.

3.3. Jump/Branch Instructions

Mnemonic	Operands	Characteristic	Description
JR	R	Unconditional	Jump to destination.
JPC	I_{28}	Unconditional	Jump to destination PC-relative.
BRFL	RF, CST	Conditional	Jump to destination if $RF == CST$.
CALL	R	Unconditional	Subroutine call.
RET	–	Unconditional	Subroutine return

3.4. No Operation Instruction

The *No Operation* instruction (NOP) is used to control the instruction flow or to insert delays (stalls) into the datapath, such as when computing the result of a jump/branch instruction. When using a NOP instruction after a branch/jump instruction it is so named a **branch delay slot**.

3.5. End of Opperations

The HALT instruction (system stop) must be implemented as a `L: j L` (a unconditional branch to the current address)

4. Internal General Purpose Registers

The current MUSA architecture provides 32 fixed-point general purpose registers of 32 bits: R_0 to R_{31}. The R_0 have special use for the hardware always returning zero, no matter what software attempts to store to it.