ipPROCESS

SYSTEM SPECIFICATION

32-bit uDLX Core Processor

Universidade Federal da Bahia

Versão: 1.0

GNU LGPL License

Histórico de Revisões

Date	Description	Author(s)
04/14/2014	Conception	João Carlos Bittencourt

CONTENTS

1. Introduction

1.1. Purpose

The main purpose of this document is to define specifications of a uDLX implementation and to provide a full overview of the design. This specifications defines all implementation parameters that composes the general uDLX requirements and specification. This definitions include processor operation modes, instruction set (ISA) and internal registers characteristics. This document also include detailed information of pipeline stages architecture, buses and other supplemental units.

1.2. Document Outline Description

This document is outlined as follow:

- Section :
- Section :

1.3. Acronyms and Abbreviations

Along this and other documents part of this project, it will be recurrent the usage of some acronyms and abbreviations. In order to keep track of this elements the Table 1 presents a set of abbreviations used and its corresponding meaning.

Table 1: Acronym and descriptions of elements in this document.

Acronym	Description
RISC	Reduced Instruction Set Computer
GPR	General Purpose Registers
FPGA	Field Gate Programmable Array
GPPU	General Purpose Processing Unit
SDRAM	
HDL	Hardware Description Language
RAW	Read After Write
CPU	Central Processing Unit
ISA	Instruction Set Architecture
ALU	Arithmetic and Logic Unit
PC	Program Counter
RFlags	Flags Register
Const	Constant

1.4. Overview

The uDLX (said micro-DeLuX) is a 32-bit simple RISC-type architecture. It features a minimal instruction set, relatively few addressing modes, and a processor organization designed to simplify implementation. Its architecture was designed to be reusable and the implementation strategy was based on incremental improvements in order to produce several designs.

The uDLX hardware structure has a five-deep pipeline architecture, and was highly designed to cover mid-low complexity applications. The uDLX is a 32-bit word-oriented system. Its architecture has 16 GPR (General Purpose Registers) in a register file. Two of these registers are reserved for special purposes. Register 0 always contains zero. It can be used as a source operand whenever zero is needed, and stores to it have no effect. Register 31 is reserved for use by some uDLX instructions, as will be described shortly. uDLX also has a 32 bit program counter.

uDLX current state supports basic arithmetical and logical operations, including multiplication and division. Those operations and others are fully detailed in the following sections.

2. Design Overview

This Chapter presents an overview of the uDLX core, the main characteristics and applications. The following sections are outlined as follows: first it is elucidated the uDLX design perspectives; then it is described the uDLX main characteristics and limitations, such as instruction set and internal architecture organization.

2.1. Perspectives

The uDLX 32-bit core release is intended to target support FPGA applications and to be deployed as a GPPU soft core. It was designed under uDLX original architecture proposed by Hennessy & Patterson (1996) which is highly based in further MIPS architecture. The uDLX presents a slightly reduced ISA.

2.2. Main Characteristics

uDLX is a 32-bit soft core processor that has been designed for general embedded applications. The main uDLX features are highlighted below.

- Support for 32-bit word;
- 32-bit RAM address;
- Parallel memory interface modules for data (2 SDRAM 32Mx16) and instructions (1 SRAM 2Mx16);
- 16 general purpose registers;
- RISC-Like Architecture with a five-deep pipeline;
- Load-Store/Register-Register processor architecture;
- Support to six external interruption sources;
- uDLX Instruction Set Architecture (See Section 3);
- Hardware division and multiplication implementation;
- Three instructions functional groups: (1) load and store; (2) computational; and (3) jump and branch.
- Three addressing modes: (1) immediate; (2) base-shift; and (3) by register;
- Capability of handle three successive data hazards (RAW) without bubble insertion between functional pipeline stages;

2.3. Non-functional Requirements

Among the uDLX main characteristics (functional requirements), a list of non-functional requirements is given by the following:

- The FPGA prototype should run in a Terasic ALTERA DE2-115 platform;
- The design must be described using Verilog-HDL;
- A set of core processor testbenches must have be provided

3. 32-bit uDLX Core Specification

4. Instruction Set Architecture

uDLX is built under the perspective of RISC Load-Store/Register-Register processor architecture. CPU instructions are 32-bits long word and organized into the following functional groups:

- Load and store
- Computational
- Jump and branch

4.1. CPU Load and Store Instructions

uDLX based processors use a load/store architecture; all operations are performed on operands held in processor registers and main memory is accessed only through load and store instructions.

Signed and unsigned integers of different sizes are supported by loads that either sign-extend or zero-extend the data loaded into the register. Table 2 lists aligned CPU load and store instructions.

Table 2: Aligned uDLX CPU Load/Store Instructions.

Mnemonic	Description
LW	Load word from data memory.
SW	Store word in memory.

4.2. Computational Instructions

This section describes the following instruction groups:

- ALU Two-Operand Instructions.
- Multiply and Divide Instructions.

Two's complement arithmetic is performed on integers represented in Two's complement notation. These are signed versions of the following operations:

- Add
- Subtract
- Multiply
- Divide

The uDLX CPU provides 32-bit integers and 32-bit arithmetic. Table 3 lists those arithmetic and logical instructions computations.

Table 3: ALU arithmetic and logic instructions.

Mnemonic	Operands	Realization	Description
ADD	R_D, R_F	$R_D \leftarrow R_D + R_F$	Add two word.
SUB	R_D, R_F	$R_D \leftarrow R_D - R_F$	Subtract two words.
MUL	R_D, R_F	$R_D \leftarrow R_D * R_F$	Multiply two words.
DIV	R_D, R_F	$R_D \leftarrow R_D / R_F$	Divide two words.
AND	R_D, R_F	$R_D \leftarrow R_D \odot R_F$	Logical AND.
OR	R_D, R_F	$R_D \leftarrow R_D \oplus R_F$	Logical OR.
CMP	R_D, R_F	–	Compares R_D and R_F and set the flags.
NOT	R_D	$R_D \leftarrow \sim R_D$	Logical NOT.

4.3. Jump and Branch Instructions

This section describes the list of Jump and Branch Instructions. Table 4 lists the conditional and unconditional jump and branch instructions.

Table 4: ALU instructions with an immediate operand.

Mnemonic	Operands	Realization	Description
JR	R	Unconditional	Jump to destination.
JPC	I_{28}	Unconditional	Jump to destination PC-relative.
BRFL	$R, Const$	Conditional	Jump to destination if RFlags = Const.
CALL	R	Unconditional	Subroutine call (jump and link).
RET	–	Unconditional	Subroutine return

4.4. No Operation Instruction

The *No Operatio*n instruction (**NOP**) is to control the instruction flow or to insert breaks into the processor such as when computing the result of a jump/branch instruction. When using a NOP instruction after a branch/jump instruction it is called a **branch delay slot**.

5. uDLX Pipeline Architecture

A block diagram of the uDLX pipeline architecture is shown in Figure 1. The architecture components are organized by a five-deep pipeline architecture. This pipeline architecture implements a Forwarding Unit in order to avoid RAW data hazards. The transfer control hazards is solved by the insertion of a Branch Prediction unit within the first pipeline stage.

The uDLX core processor uses a five-deep parallel pipeline on its architecture. The current pipeline is divided into the following stages, also described in Figure 1:

1. Instruction Fetch
2. Instruction Decode
3. Arithmetic operation (Execution)
4. Memory access
5. Write back

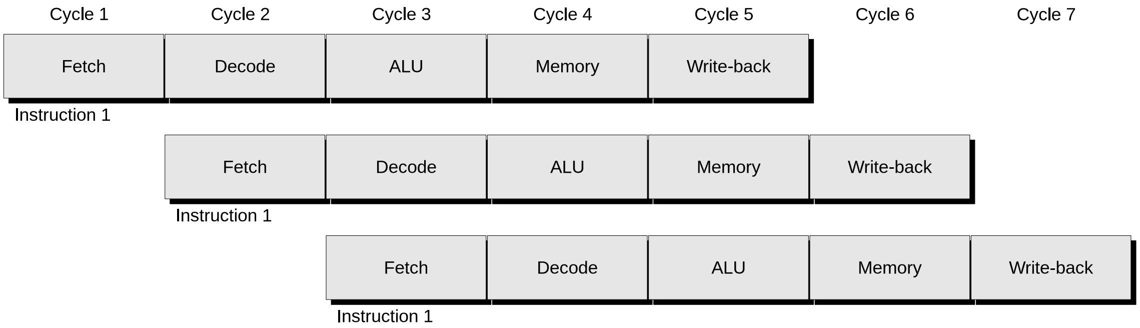

Figure 1: Five-Deep Single-Completion Pipeline.

6. Internal General Purpose Registers

The current uDLX architecture provide 32 fixed-point general purpose registers: R_0 to R_{15}. Two of these register have special use for the hardware. One R_0 always returns zero, no matter what software attempts to store to it. The other (R_{15}) is used by the normal subroutine-calling instruction (Jump and Link) for the return address.

7. Licenses

The uDLX soft core license of usage are free through provided LGPL v3. There is no external IP-core licenses related to uDLX core system development.