SYSTEM SPECIFICATION

32-bit μDLX Core Processor

Universidade Federal da Bahia

Version: 1.0

GNU LGPL License

History Review

Date	Description	Author(s)
04/14/2014	Conception	João Carlos Bittencourt
05/08/2014	Structure and description review	João Carlos Bittencourt

CONTENTS

1 Introduction **4**

1.1 Purpose 4

1.2 Document Outline Description 4

1.3 Acronyms and Abbreviations 4

2 Design Overview **5**

2.1 Perspectives 5

2.2 Main Characteristics 5

2.3 Non-functional Requirements 5

3 Instruction Set Architecture **6**

3.1 CPU Load and Store Instructions 6

3.2 Computational Instructions 6

3.3 Jump/Branch Instructions 7

3.4 No Operation Instruction 7

4 Pipeline Architecture **7**

5 Internal General Purpose Registers **8**

6 Licenses **9**

1. Introduction

1.1. Purpose

The main purpose of this document is to define the specifications of a μDLX implementation and to provide a full overview of the design. The following sections defines implementation parameters which composes the general μDLX requirements and specification. This requirements 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 2: This section presents the architecture design overview and requirements.
- Section 3: This section specifies the processor instruction set.
- Section 4: This section presents an overview of the proposed pipeline architecture.
- Section 5: This section describes the general purpose registers.
- Section 6: This section provides information about licenses of usage for this core.

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
SDRAM	Synchronous Dynamic Random Access Memory
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

2. Design Overview

The μDLX (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. The microprocessor hardware structure has a five-deep pipeline architecture, and was highly designed to cover low complexity applications. The μDLX is a 32-bit word-oriented system. Its architecture is composed by 16 GPR (General Purpose Registers) in one 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 16 is reserved for use by some μDLX instructions, as will be described shortly. μDLX also has a 32 bit program counter.

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

2.1. Perspectives

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

2.2. Main Characteristics

μDLX is a 32-bit soft core processor that has been designed for general embedded applications. The main μDLX 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;
- μDLX 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 NOP insertion between functional pipeline stages;

2.3. Non-functional Requirements

Among the μDLX main characteristics, 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. Instruction Set Architecture

μDLX 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

3.1. CPU Load and Store Instructions

μDLX-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 1 lists aligned CPU load and store instructions.

Table 1: Aligned μDLX CPU Load/Store Instructions.

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

3.2. Computational Instructions

The computational instructions uses ALU two-operand 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 μDLX provides 32-bit integers and 32-bit arithmetic. Table 2 lists those arithmetic and logical instructions computations.

Table 2: 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.

3.3. Jump/Branch Instructions

Table 3 lists the conditional and unconditional jump and branch instructions.

Table 3: Jump/Branch instruction set.

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

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. Pipeline Architecture

A block diagram of the μDLX pipeline architecture is shown in Figure 2. 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.

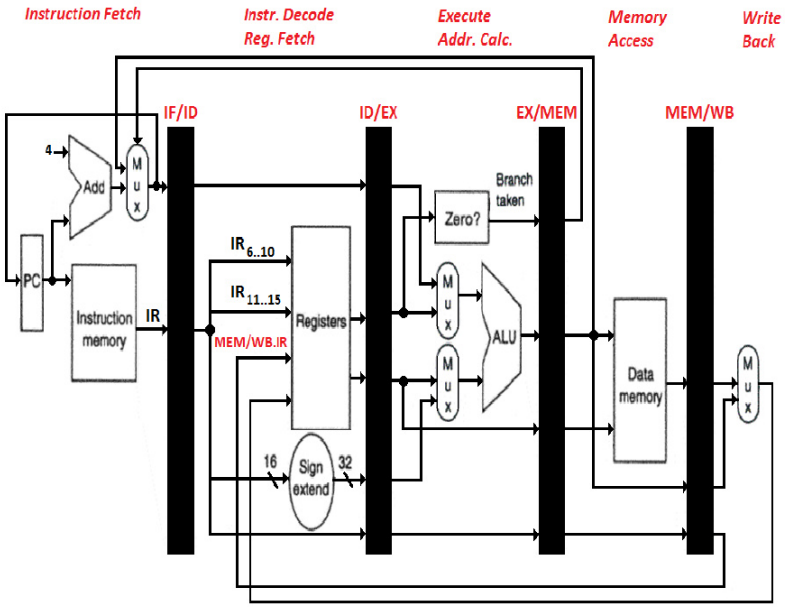

Figure 1: Pipeline datapath structure proposal.

The μDLX core processor uses a five-deep parallel pipeline on its architecture. The current pipeline is divided into the following stages, also described in Figure 2:

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

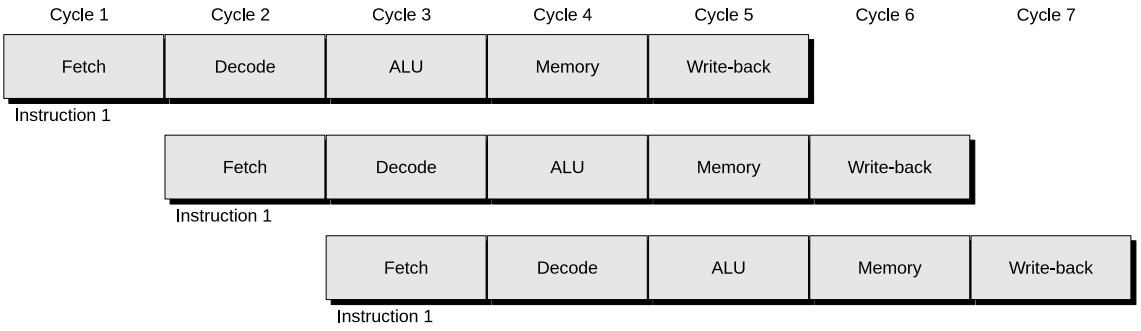

Figure 2: Five-Deep Single-Completion Pipeline.

5. Internal General Purpose Registers

The current μDLX architecture provides 16 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.

6. Licenses

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