POLITECNICO DI TORINO

Master Degree in Computer Engineering: Embedded Systems
MICROELECTRONICS SYSTEMS

Design and Development of a DLX Microprocessor

Professor: Mariagrazia Graziano

Autore: Alessandro Salvato 237771 (gr.45)

Data: September 13, 2018

Contents

Chapter 1

Introduction

This report aims at being a short documentation on the Microelectronic Systems course project at the first year of the master degree both in Computer and Electronics Engineering at Politecnico di Torino.
The target is the **design** and the **implementation** phase of a pipelined processor by VHDL, as is described in [1], followed by a simple **synthesis** and a **physical layout** definition. The main flow has passed through the following steps:

1. Design and implementation at RTL level
2. Simulation by assembly codes
3. Synthesis and gathering results about timing, area and power consuption
4. Realization of the physical layout

For 1 and 2, I used the *Xilinx ISE Design Suite 14.7*, although during the laboratory sessions the tool was *ModelSim*. I made this choice because I think it's better both from the management of files point of view and for the more intuitive interface of the simulation tool. The synthesis step has been performed by *Synopsys' DesignVision*, whose license has been granted to the Politecnico di Torino. In order to use it, I haved to use a virtual machine, running on *Linux*, provided by the Department of Electronics and Telecommunications servers. Same conditions for the last step, by *Cadence Innovus Implementation System*. Moreover, I liked exploiting *GitHub* functionalities to make easier the passage of data from my *Windows* laptop and the virtual machine; you can download it from `https://github.com/sandrosalvato94/MyDLX`

1.1 Basic vs Pro version

Specification are described in [2], you can find it among files in Documents directory. Summarizing, basics features for the Basic version are:

- **Pipeline**
- **Simple instruction set** (see below)
- **Simple datapath**
- **Basic synthesis**
- **Basic physical design**
- This **report**

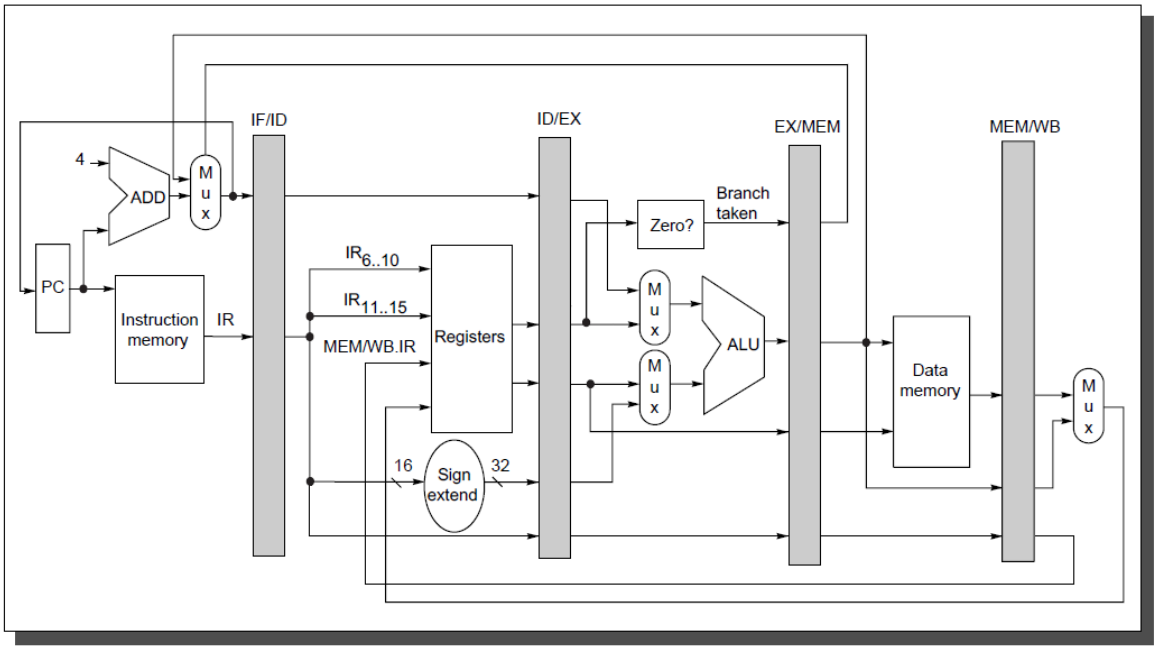

Figure 1.1: Basic DLX Datapath. Figure 3.4 from [1]

Among pro hints, these are reported:

- **Extended instruction set** (see below)
- **Extended datapath**
- **Windowing**
- **Control Hazard**
- **Optimization of the power consuption**
- **Caching**
- **Advanced synthesis**
- **Advanced physical design**
- **Whatever I wanted...**

I targeted the pro version, exploring and implementing the following features:

- **Extended instruction set**
  - **More** instructions
  - **Modified** instructions
  - **Totally new** instructions
- **Control hazard** management by a **Branch Target Buffer**
- **Data hazard** management by a **forwarding logic** (*just r-type and i-type instructions*)
- **Extended datapath** by new components

1.2 Instruction Set

Basics information on the DLX instruction structures can be found in [2].

Instruction	Type	Annotation
add addi and andi beqz bnez j jal lw nop or ori sge sgei sle slei sll slli sne snei srl srli sub subi sw xor xori	Basic DLX	n.27
addui subui lhi jr jalr srai seqi slti sgti lb lh lbu lhu sb sh sltui sgtui sleui sgeui sra addu subu seq slt sgt sltu sgtu sleu sgeu	Pro DLX	n.29
mult multu	Modified instructions	The binary format is changed. mult(u) r1, r2, r3 has been replaced by **mult(u) r2, r3** <– where the product is stored in two special registers, one for the 32 lowest bits and the other for the 32 highest.
mflo mfhi	New instructions	**mflo r1** <– loads the 32 lowest bits of a product in r1 **mfhi r1** <– loads the 32 highest bits in r1

Table 1.1: Instruction set

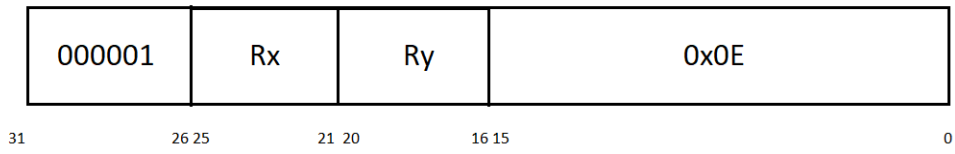

Figure 1.2: Mult format

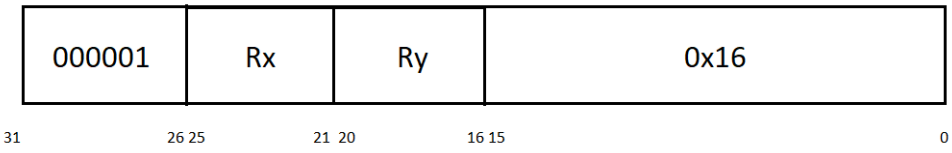

Figure 1.3: Multu format

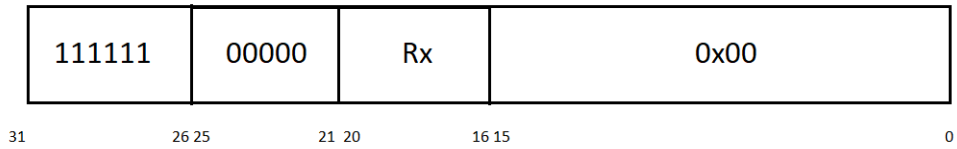

Figure 1.4: Mflo format

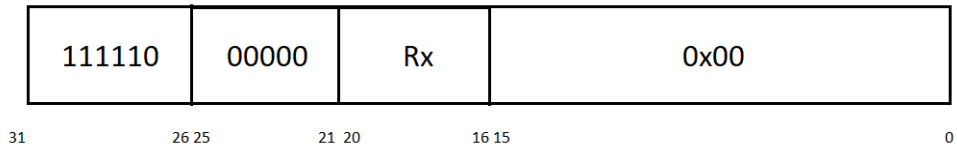

Figure 1.5: Mfhi format

More information on Basic and Pro DLX instruction in [2].

Chapter 2

Register Transfer Level

2.1 Core

This is the higher level of the design. It's interface with the external world and the data and the instruction memory, which are not reported in this document.

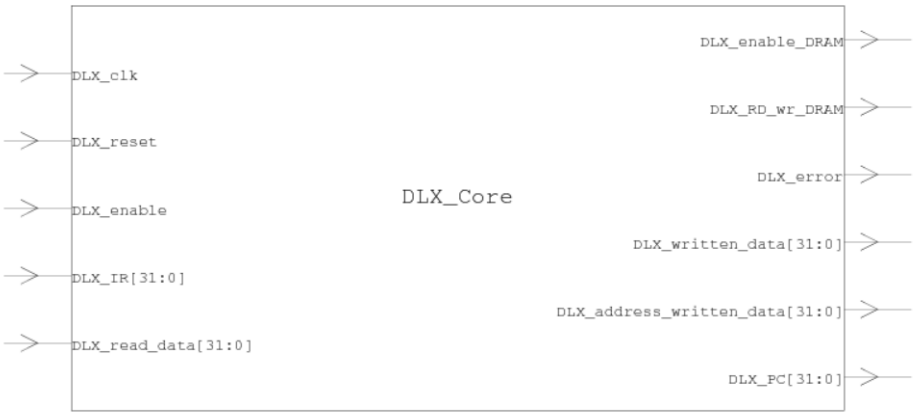

Figure 2.1: Core pinout

Just two informations about these signals. *Clk* and *enable* own their basic functionalities, *IR* is the data read from the instruction RAM at each clock cycle by fetching. *Read_data* comes from the DRAM; *enable_DRAM* allows read and write operation on it, where the choice is made by *RD_wr_RAM*. *Error* is just a single pin out used for debugging, so not consider it. *Written_data* and *address_written_data* are signals to the DRAM; in the end the *PC* to the IRAM.
The core is composed by 4 macroblocks:

- Datapath
- Control unit
- Branch target buffer
- Branch misprediction manager

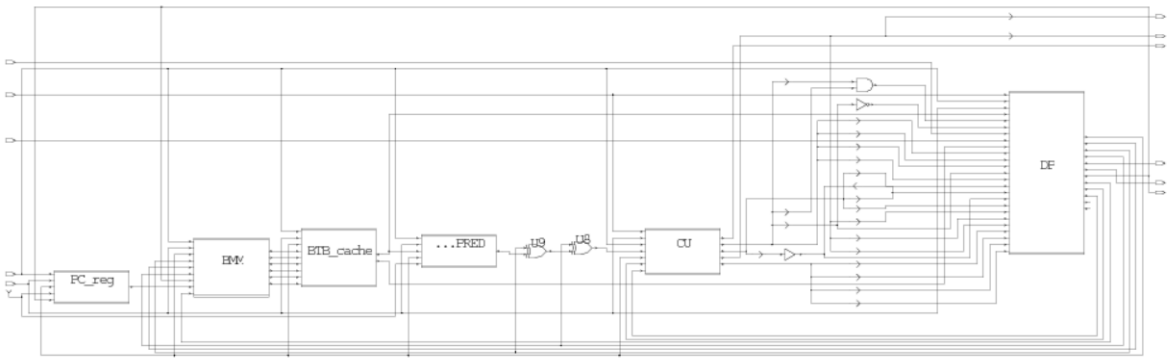

Figure 2.2: Core schematic - I

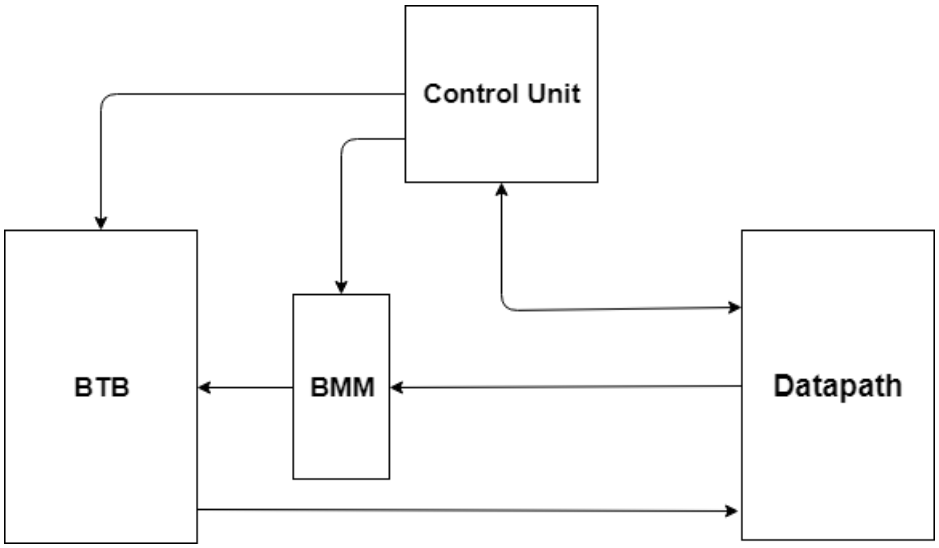

Figure 2.3: Core schematic - II

The datapath is the part of microprocessor computing instructions. In this design is the biggest component, where are defined the five pipeline stages. The branch target buffer is a sort of cache memory, used to manage control hazards; in fact branch addresses are stored in it. However almost all communications between datapath and BTB are managed by the Branch Misprediction Manager, which works when a wrong prediction occurs. Finally there is the hardwired Control Unit, that is the main brain of the DLX; as the datapath it's pipelined too.
Datapath sends to BMM, so to BTB, information about the current instruction and the previous one such as: the program counter and whether one instruction is a branch or not. On the other side the BTB responds sending the branch target and the boolean value of the prediction (taken or untaken). The control unit oversees all macroblocks. It receives the instruction register of the instruction getting in the decode stage and produces all control signals.

2.2 BTB

I guess talking about BTB before datapath should let you understand better the overall behaviour of this microprocessor.

Bibliography

[1] *Computer Architecture: A Quantitative Approach.* Morgan Kaufmann, 2011.

[2] Giovanna Turvani Mariagrazia Graziano, Giulia Santoro. Design and development of dlx microprocessors. Microelectronic Systems Final Project Specification.