MARISCO: A Multi-Core Platform

Laysson Oliveira Luz
Universidade Federal do Piauí
Departamento de Informática e Estatística
layssonoliveir4@gmail.com

Ivan Saraiva Silva
Universidade Federal do Piauí
Departamento de Informática e Estatística
ivan@ufpi.edu.br

Thiago R. B. da Silva Soares
Universidade Federal do Piauí
Departamento de Informática e Estatística
thiagorbss@gmail.com

ABSTRACT
Multi-core architecture is the natural result of industrial capability development. For decades enhancements in the performance of microprocessors were based on the reduction the transistors size and increasing the clock frequency. These technological evolutions enabled the integration of more and more transistors on a single chip as well as performance improvement without architectural modification. However, the industrial capability to continuously increase the clock frequency has reached its limits. In recent years with the inability to increase over again the clock frequency and the continuous increase in the integration capacity the industry was conducted to engage in the race for developing processors with multiple processing cores. This paper presents the design and implementation of a multi-core embedded processor platform through the integration of general-purpose processors using a crossbar as interconnection subsystems. All the hardware devices used in the multi-core embedded processor were described in VHDL language aiming an FPGA implementation.

Categories and Subject Descriptors
C.1.2 [**Processor Architecture**]: Multiple Data Stream Architectures (Multiprocessors) - *Array and vector processors, Associative processors, Connection machines, Interconnection architectures, Multiple-instruction-stream, multiple-data-stream processors (MIMD), Parallel processors, Pipeline processors.*

General Terms
Language, Microprocessors, Reconfigurable Architecture, Instructions.

Keywords
VHDL, RISCO, Microprocessors, Reconfigurable, Architecture.

1. INTRODUCTION
In the past decade was observed that the consumer electronic industry has been engaged in a race to increase the integration capability and the frequency of circuits. This race has emerged to meet the growing demand for consumer electronic devices: mobile phones, personal computers, among other high-tech gadgets. As a consequence devices were developed with enhanced functionality and design complexity. High level of integration, small size and low power consumption became the industry's technological goals. Furthermore significant impact as observed in the work designer's team, the time available for conclude a project reduces considerably [1].

The platform concept emerged with the need to develop new methods aiming to solve the problem of reduced time to market and the gap of technology. The technology gap is the difference between the productivity of development teams and integration capacity of the semiconductor industry.

The platform-based design methodology applies directly to the development of multiprocessor systems on chip (MP-SoCs). Such systems consist of an integrated set of cores with different processing capacity (general-purpose processors, dedicated cores, DSPs - Digital Signal Processors, memory, interconnection sub-system, among others). Multi-cores are more restrictive examples of MP-SoCs. It consists generally in the integration of processing cores, memory and interconnection sub-system.

A platform is frequently defined as an abstraction that hides or simplifies the design details of a system. A MP-SoC platform consist in a library cores, specified at some abstraction level, and delivered with rules of integration and information about performance functionalities.

This paper presents the design of a multi-core embedded processor platform. The design uses the RISCO [2] processor, a crossbar as interconnection subsystems and distributed memories. The paper is organized as follows: Section two presents the multi-core embedded platform; section three presents a discussion about integration and validation experiments and section four presents future works.

2. MULTI-CORE WORKS
Multi-core architectures have become mainstream in embedded systems design. Currently it is easy to find several papers in major journals and conferences on topics related to multi-core architectures. Many of them focus on topics related to software development: programming, operating systems, virtualization, and so one. Many other address topics related to hardware development, mainly interconnecting sub-systems, low power architectures, and memory hierarchy, among other topics.

This section will not present a comparative study between the architecture proposed in this paper and others presented in the literature, that is present architecture as the platform ARM cortex A9 (designed for mobile computation of geral purpose that can be custom before produced)[3] and the architecture Intel Core i7 (also of geral purpose, that use the concept of symmetric multithreading)[4]. The main reason is that it is a work with educational purpose that aims at a first moment to design a platform that after will be used to develop new hardware and software resources to multi-core processors.

3. THE MULTICORE PLATFORM

3.1 Risco Processor

RISCO is a RISC processor architecture 32-bit, developed for CMOS technology. Its data, instructions and addresses are 32-bit words. Its address unit is the word, not being addressed byte or half word. You can access, so about 4 Giga words (17 Gbytes).

Communication with the memory is made by only one bus, called the BSIS , multiplexing data and addresses through it and doing a search for instructions and memory access operations. There are 32 registers of 32 bits (R0 to R31), R0 being reserved for the constant 0, R1 reserved for status word and R31 set the program counter (PC).

RISC has a pipeline of three stages, reaching the peak of one instruction by machine cycle. Each machine cycle consists of three different phases.[2]

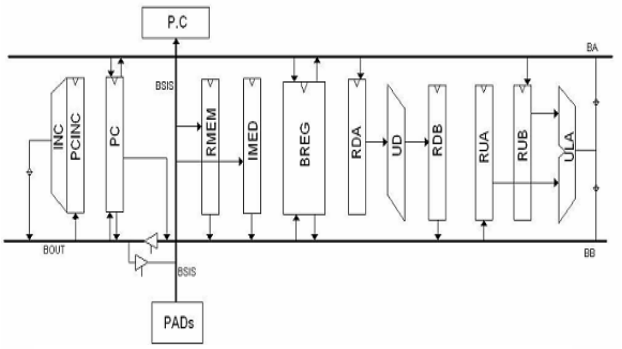

Figure 1.Image of microprocessor RISCO

3.2 The Crossbar Architecture

The crossbar is a quite simple NoC (Network on Chip), this is, a designing the communication subsystem between IP cores in a System-on-a-Chip (SoC). The crossbar is written in VHDL, which has gates for five cores, for transmitting both data and control signals.

The architecture does not have the cache or RAM, is described only with registers and signals, which ensures agility in processing such signals and data recovery also implies a low power, highly functional feature when dealing with processors because this integrated circuit should suit the needs of the five centers in real time, so any failure that may result in deadlocks, which is not acceptable.

In crossbar there is only crossing bit sequences the same maps all core in order to coordinate all of the cores transfer between them. The architecture maintains a register for each core and each of these with reference to the other core, so it is possible to address any data from one processor to any other, all in the shortest time possible.

3.3 MARISCO

By the universality of the crossbar architecture we envision a multi-core processors pipeline in order to evaluate the gain agility to execute tasks previously performed using multicycle core.

The platform developed in hardware description language (VHDL) is a homogeneous architecture as it is constituted by instances of the same microprocessor and logic has a simple and very functional, with the basic principle of communication core integrated to it.

Described in low-level language, the platform called MARISCO is basically a network of gates and wires that communicate among themselves RISC five processors, allowing create clusters of microprocessors.

The establishment of multi-core platform occur in first moment with a testing of each core in particular, after were tested two cores communicating among themselves, after this test having is approved, the others cores were integrated to platform crossbar, since the performance of the architecture consist to multicycle processor, the implementation MARISCO aims to evaluate the performance gain with pipeline RISC processors.

The microprocessor RISCO was modified so that the processing power of architecture to be tested for the experiments were inserted two instructions, data synchronization, processor, with which the cores were able to send and receive data between each other, this is, to RISCO was added to the two instructions of input and data output (execute_in and execute_out).

The integration of the five processors multi-core platform will occur by means of some control ports and data. In detail, each core has four input vectors to receive data from each of the other cores, and the control signals from the crossbar for writing and reading, and two registers are used respectively to send data and determine which core send the data, with these signals the crossbar sends sequences of bytes to their respective destinations.

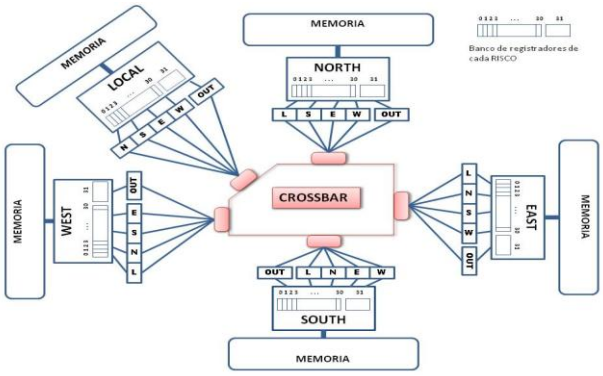

Figure 2. Schematic of multicore platform of processors RISCO (MaRISCO)

The whole architecture operating synchronously with the core causes instructions as the synchronization data to force cores involved in these instructions to "stop", this is, assuming two core , A and B, when the core A needs of a data core B, A must wait until the data has been read from the core A and written on your register corresponding to this core. This "stop" the flow of execution of the processor is required to prevent data recovery "garbage", this is, the core prevents receive undue data.

The MARISCO has the peculiarity of being a pipeline processor, which makes benchmarks programming for the same, this is, work for the programmer is complicated by the peculiarities of

the ISA (Instruction Set Architecture), due to having to avoid data conflicts, for example, request a register at the time that it is being updated or writing.

Therefore, the development of benchmarks for MARISCO is the need for attention to conflict does not occur, conflicts as loss of data on transmission between cores or even deadlocks that eventually would occur by any delay of a core in send data or for even not send the data.

4. MaRISCO's Especification

Having mounted the MARISCO, that is, completed the unification of the five instances RISCO and the crossbar architecture, the results of the compilation the MARISCO reveal some important features such as clock frequency and size (logic elements) of the architecture

Next to the compilation of the architecture, simulations were performed to verify the operation of the platform, evaluating it for safety and reliability, that is, verifying the integrity of the data after sending (out) and receiving (in) for each one of the cores.

The simulation is programmed as follows: each core has its program in memory, this program determines the initialization of registers, performs a simple arithmetic operation and then sends the resulting data to another core, and if this core need an external data, the program performs the operation *in* to receive this data.

The benchmark for validation of the architecture starts at the Local core, which initializes two registers and sends one to the North core and the other to the South, the North receives the data from the Local, save it in the register, use it as operating on an addition and sends the result to the East core. This receives the data from the North, stored in the register file, uses it in a division and sends the result to the Local core.

The South core receives the data delivered by the Local, uses it in a sum and sends the result to the West core, it receives it, store it in a register, use it in a multiplication and sends the result to the Local core.

Finally, the Local core, is waiting for data from the East and West cores, when they arrive, the Local receives, recording in register, then saves them in memory.

The simulations allowed to evaluate the performance of the architecture, despite the simplicity of the benchmark, all data transferred do not changed, and there was no loss of data, thus expressing the reliability of the architecture and the data transfer.

Described in VHDL, using the software of the Altera® Quartus II v9.1 Web Edition, the platform was synthesized on FPGA Ciclone III EP3C55F484C6, that has 55.856 logical elements available, these the architecture uses 26.544 combinational functions and 6.305 dedicated registers, so, the architecture occupies 48% of capacity of FPGA.

Each processor in specific has a frequency of 45,91 MHz, the platform, formed of five RISCO's unity that *crossbar* , has a frequency of 81,07 MHz, that features a execution very fast of multicycle instructions in pipeline.

5. FUTURE WORKS

The idea of increasing the processing capacity with more than two cores by chip is no longer sufficient to meet the needs of users. As increasingly decreased the size of transistors that would fit more transistors by chip, the tendency now is increasing the number of cores.

The multi-core platform of RISCO still being improved, so that have a greater storage capacity for both data as instructions, this is, for example the implementation of Harvard model, which has separate memories for different purposes.

Besides the constant need for greater storage capacity, is very importance that the memory gain does not imply in loss of speed or large increase in power consumption, therefore the platform is being worked in order to provide units of memory access, called DMA (Direct Memory access), to speed access to the memory.

The properties will MARISCO not limited physical layer of the architecture, the goal is virtualization and reprogramming, this is, and respectively the capacity of cores simulate other processors and capacity to reprogram the same at runtime.

In addition to developing the hardware level, there is a universal need for the compiler, that is, the development of the compiler RISCO not only to one processor, but for the five cores in order to program it already multi-core in order to support to reprogramming.

6. REFERENCES

[1] Bass, M. J.; Christensen, C. M.; The Futureof the Microprocessor Business. IEEE Spectrun. Abril 2002.

[2] Junqueira, A. A.; RISCO-Microprocessador RISC CMOS de 32 bits. Dissertação de Mestrado. Porto Alegre, RS. Setembro,

[3] "The arm cortex-a9 processors," tech.rep.,ARM Ltd., September 2007. Disponível em 11/06/2010 http://www.arm.com/pdfs/ARMCortexa-9Processors.pdf.

[4] Chiachia, Giovani; Arquiteturas Multicore*. Campinas, SP. Disponível em http://www.ic.unicamp.br/~ducatte/mo401/1s2010/T2/098362-t2.pdf