Advantages of FPGA-Based Multiprocessor Systems in Industrial Applications

Ralf Joost
Institute of Applied Microelectronics and Computer Engineering
University of Rostock
Rostock, 18051, Germany
ralf.joost@uni-rostock.de

Ralf Salomon
Institute of Applied Microelectronics and Computer Engineering
University of Rostock
Rostock, 18051, Germany
ralf.salomon@uni-rostock.de

***Abstract* – Today, *industrial* production machines must be highly flexible in order to competitively account for dynamic and unforeseen changes in the product demands. This paper shows that field-programmable gate arrays (FPGAs) are especially suited to fulfil these requirements; FPGAs are very powerful, relatively inexpensive, and adaptable, since their configuration is specified in an abstract hardware description language. In addition to the benefits resulting from using an FPGA, the case study presented here shows how FPGAs can be used for implementing a sophisticated multiprocessor architecture. In the course of presentation, this paper furthermore argues that due to the excellent tool support and the availability of reusable functional modules (also known as intellectual properties) this approach can be adopted by almost any industrial hardware designer.**

I. INTRODUCTION

Nowadays, manufactures face new challenges in the development of production lines. These production lines, i.e., the machines, have to acknowledge the demands of the products they produce. First of all, the products' life cycles are becoming shorter, and second, quality as well as reliability demands are significantly increasing. As a consequence of those market-driven trends, the production machines as well must allow for rapid and cost-efficient changes and modifications. In addition, both the amount of data to be processed during production and the requirements with respect to the processing speed itself are tremendously increasing.

Production machines (simply called machines in the remainder of this paper) generally consist of relatively fixed mechanical components, such as the machine's body and motors, and an electronic control system. Most control systems consist of an electronic hardware platform, which hosts various control programs. As opposed to the underlying hardware, control programs are highly flexible, since they are normally realized in software.

Traditionally, the designer has mainly the following three options to realize the machine's hardware control platform: digital signal processors[1] (DSPs), FPGAs, and ASICs. In the one extreme, ASICs provide highest performance, because their design can be optimally configured with respect to the application's requirements. However, the costs of ASIC-based solutions are the highest, partly due to the small number of units as the number of production machines is normally relatively small in comparison to the products it produces. On the other extreme, DSP-based solutions are cost effective, but do provide only suboptimal processing speed due to the execution of software programs.

Between these two extremes, FPGAs provide a compromise, which is suitable for many applications. Although FPGAs have been in use for many years and are well accepted by engineers and developers, they could not reach the propagation of application-specific microcontrollers. It is most likely that the price and well as the lack of available knowledge were the most important arguments.

During the last years, however, the development of FPGA-technology has advanced in terms of usable resources and number of gates, processing speed, energy consumption, and retail prices. State-of-the-art FPGAs are thus interesting alternatives to common microcontrollers and ASICs for the applications mentioned above [1, 2].

Current FPGA-based systems combine many advantages of DSPs and ASICs. This includes, rapid development cycles, high flexibility and reusability, moderate costs, easy upgrading (due to the usage of abstract hardware description languages (HDLs)), and feature extension (as long as the FPGA is not exhausted). Furthermore, current FPGAs allow for the integration of soft-core processors. That is, an FPGA might employ typical processor capabilities.

In addition to new developments, FPGA-based solutions can be advantageously used in existing, “naturally grown” systems. A “key feature” of such naturally grown systems is that over time, the original hardware platform has been augmented by various extension boards. As a consequence, such systems are both hard to maintain and hard to extend as well as sensible to component failures. A further problem arises from the distribution of cooperating functionalities, since the transfer of (large) data amounts is required. This increases the system's complexity further, which in turn decreases flexibility and reliability. Moreover, the oldest components restrict further modifications of such heterogeneous systems. This involves, for instance, limited I/O-capabilities, not-supported bus-types and protocols. Finally, heterogeneous systems depend on the support and mercy of a large number of *different* vendors.

[1] This also includes general-purpose CPUs, microcontroller, etc.

This paper argues and shows that FPGAs can at least relieve some of the design problems discussed above. To this end, Section II briefly surveys the possibilities and advantages of those FPGAs. A short introduction to the simple and easy use of soft-core processors will be given.

As Section III describes, FPGAs today have grown in size such that they can even realize one or even several custom-made processor cores.

In order to show the potential benefits, Section IV describes an industrial case study in the area of cutting-edge offset printing. Since the entire system has been grown during several years, it exhibits some limitations, which also hinder future developments. In order to expand to new frontiers, this section also proposes a new architecture, which is based on a multiprocessor platform.

Section V reports on a prototypical implementation, which can be seen as a proof-of-concept. The section also discusses the required resource consumption as well as other technical details. Finally Section V and VI finish this paper with a critical discussion and the conclusion respectively.

II. BACKGROUND ON FPGAS

First commercial FPGA's were released by XILINX in the middle of the 80's of the last century. At those times, devices were small, slow, and quite expensive. The first XILINX device XC2064 had 1200 gates. Today, FPGA's offer millions of system gates and operate at a speed of up to 300 MHz. Since the price of currently available FPGAs starts below 10 US$, many manufactures use them even for end products, such as mobile phones, network solutions, and multimedia devices.

Even factory automation applications might profit from the good performance-to-cost[2] ratio of modern FPGAs. Here, FPGAs are of particular interest, because a small number of production systems require significant development efforts.

Using FPGAs in the design process has the following three distict advantages. First, most FPGA-vendors support the design and development process by providing powerful and easy-to-use electronic design tools (EDA), excellent documentation, and personal support. Second, demonstration examples do not involve high manufacturing costs as would be usual with ASICs. Third, modifications and adjustments can be implemented at any stage of the design process and even "in the field". Advanced systems extent the latter point even further in that they allow the dynamic reconfiguration of the running hardware. Such systems are subject of current research efforts, also known as *dynamically reconfigurable hardware* [10]. This approach receives recent attention, because it allows for multiply re-using valuable resources.

FPGAs have grown in size that much that they allow for the soft-implementation of processor cores, which might also be a DSP-core. Thus, a design challenge is to decide, which parts of the system, i.e., functionalities, are to be implemented directly in hardware gates or processed by software, which runs inside the FPGAs soft core. State-of-the-art EDA-tools, such as ALTERA's system-on-a-programmable-chip (SOPC) Builder and XILINX' Embedded Development Kit (EDK), simplify this parallel hardware-software co-design process dramatically. This is accomplished by automatically recognizing all hardware components and providing software access methods for them. Also, the development tools solve the address and resource management automatically on the fly (see also Section IV.C).

Furthermore, the developer can resort to many ready-to-use components, also called intellectual properties (IP), such as network-Interfaces, video controllers, memory controllers, and so forth. The usage of IP-cores reduces the development efforts and thus may help reduce costs.

III. SOFT-CORES AND MULTIPROCESSOR SYSTEMS

As mentioned above, FPGAs offer the possibility of incorporating soft-cores. Soft-core processors can be considered as equivalents to a microcontroller or "computer on a chip". They combine a CPU, peripherals, and memory on a single chip. They also provide access beyond the actual FPGA chip through integrated standard or custom-made interfaces. Some available reduced instruction set computer (RISC) architectures are [3, 4, 5]:

1. NIOS and NIOS II CPUs from ALTERA ,
2. LEON2 and LEON3 CPUs from Gaisler Research,
3. and the Microblaze CPU from XILINX.

Some of the currently available FPGAs allow for the realization of several processors simultaneously. As is well known, multiprocessor systems are a proper method to increase system performance and to concentrate processing elements in one FPGA. Since multiprocessor systems are supported by the EDA tools, the system generation can be completed within a couple of days. Generally, most vendors support multiprocessor systems by providing dedicated hardware constructs, such as mutexes, to synchronized accesses to shared resources. For a general introduction to multiprocessor systems, the interested reader is referred to the pertinent literature [7].

IV. INDUSTRIAL CASE STUDY

This section presents an industrial case study. This includes a description of the existing system, the discussion of the problems and goals, and the presentation of a prototypical implementation.

A. The Existing System

In 1993, basysPrint Ltd. introduced a technology called Computer To conventional Plate (CTcP™) for the exposure

[2] In this context, "cost" refers not only to the actual FPGA but also to required tools, design efforts, maintenance, etc.

of printing plates for the offset printing process [12]. A distinct feature of the CTcP™ technology is that it allows for using ultraviolet-light-sensitive printing plates, which, in comparison to laser light exposition used by most other competitors, results in significant cost reductions because of lower acquisition and disposal charges. Fig. 1 shows an example of a CTcP™ solution available from basysPrint.

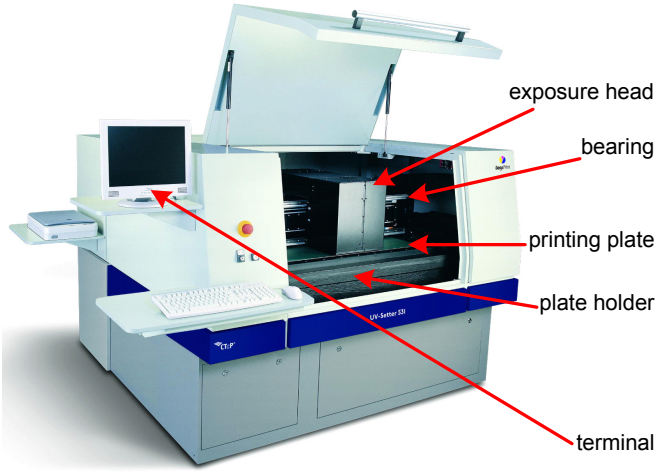

Fig. 1: A CTcP™ system, called UV-Setter

The very first version of the CTcP™ process exposed the printing plate in a so-called step-and-repeat mode. The image was divided into tiles, which were transferred and processed separately. A tile consists of up to 1280 by 1024 pixels. Each tile covers an area of approximately 4 cm². It might be important to note that the processing, i.e., exposing the printing plate, of each tile requires the exact compliance of pre-defined exposure times. After a tile is finished, the exposure head advances to the next position. During the movement of the exposure head, no information is transferred to the plate, thus increasing the time required to expose the whole plate. Due to this disadvantage, basysPrint developed a continuous exposing process, called scrolling mode. This process divides an image into adjacent stripes, with each consisting of an entire row of the aforementioned tiles. Furthermore, when processing a stripe, the exposure head is moving with constant speed.

Due to cost reasons, the new system architecture is realized by several modifications and add-ons of the old architecture. In other words, the current system architecture is the result of a rather evolutionary development process. Such an evolutionary development process is *the* regular case, but clearly results in some limitations, which are discussed in detail in Section IV.B.

Fig. 2 presents a simplified schematic of the electrical control system. All grey-shaded elements symbolize components, which are moving over the plate's surface during the image processing procedure. The two motors move the exposure head, also called image processing system, to the desired position. The two encoders feed the information about the current position back to the trigger-card, which is installed in the control PC. The trigger-card synchronizes the image updating with the exposure head's movement. The PC transfers the appropriate image data via a LAN-connection to the exposure head prior to the imaging of each stripe. In addition the control-PC also has to handle various other control signals. For the system's functionality, it is very important to ensure a precise coordination of head movements and data timings, because even a single malfunction spoils the entire plate.

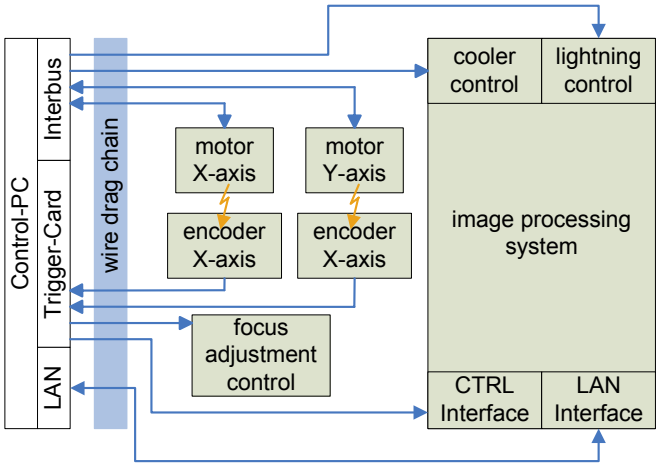

Fig. 2: Schematic of a printing plate processing system

B. Problems and Goals

The current system architecture suffers from the following four limitations. First of all, it employs a large number of electrical wires for transmitting all the required data, such as image data, motor positioning information, signals for focus adjustment, as well as various other control signals. These wires are embedded in a drag chain (connection between fixed body parts and the moving exposure head). From a mechanical engineering point of view, the large number of wires is a severe problem, because the drag chain requires a substantial amount of the machine's rare space. The usage of a second exposure head makes this situation even worse, since it requires an additional set of transfer and control wires.

The second issue is given by the spatial distribution of data sources and sinks. This requires rather long wires and, as a consequence, significant efforts with respect to signal stability and reliability, connectors, as well as testing and maintenance.

The third topic to address consists of the large variety of communication protocols. Since each connection has slightly different timing parameters, further system extensions are hard to realize, or nearly impossible.

The fourth limitation concerns the incorporation of a new processing scheme, which is currently under development at basysPrint. This new scheme tries to avoid mechanical interruptions and thus detrimental vibrations by a rather continuous exposing process.

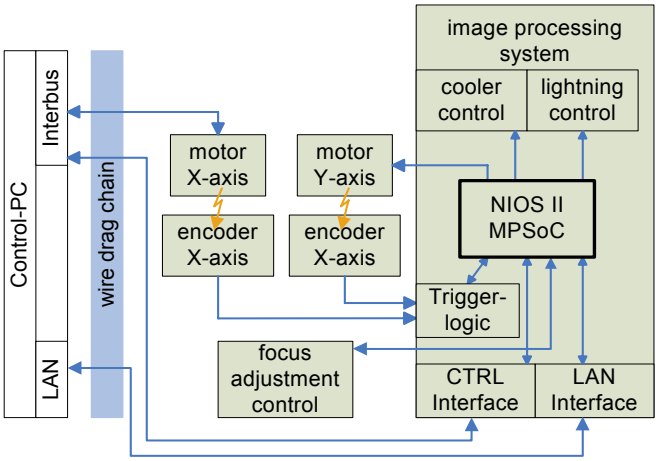

Fig. 3: Schematic of the modified system

As a consequence of the issues discussed above, basysPrint, in cooperation with the University of Rostock, is redesigning the entire electrical system[3]. In so doing, it pursues the following goals:

1. Significantly reducing the total number of all data and control wires.
2. Aiming at a compact system design, i.e., many components at only a few places, in order to reduce the number of cables, connectors, and potential sources of failures.
3. Concentration of functionalities locally in order to avoid permanent data-transfers and to reduce the number of required bus protocols.

The following subsection presents a system design that accounts for the problems and goals discussed above.

C. An FPGA-Based Multiprocessor Architecture

The new architecture, as shown in Fig. 3, is based on a programmable FPGA-device. The main part of the new system is a NIOS II-based multiprocessor architecture, which – among other advantages – allows for scalability on demand and loosely coupled system designs. For an in-depth discussion of the advantages and disadvantages of multiprocessor architectures, the interested reader is referred to [7]. The chosen FPGA contains the multiprocessor system on a chip (MPSoC) and the trigger-logic, formerly installed in the host-PC, as well. To ensure the system's proper timing behavior, i.e., the timely coordination of the motor movements, trigger events, and supply of image data, the MPSoC includes one processor-core only for managing the time-critical operations. This approach significantly relaxes the mutual dependencies, and thus limitations, between the exposure system and the host. Since the trigger-logic resides inside the FPGA, it also shortens the communication paths, and thus reducing the requirements and structures for signal propagation on the fly.

[3] In other cooperation with the University of Rostock, basysPrint is also redesigning other system parts.

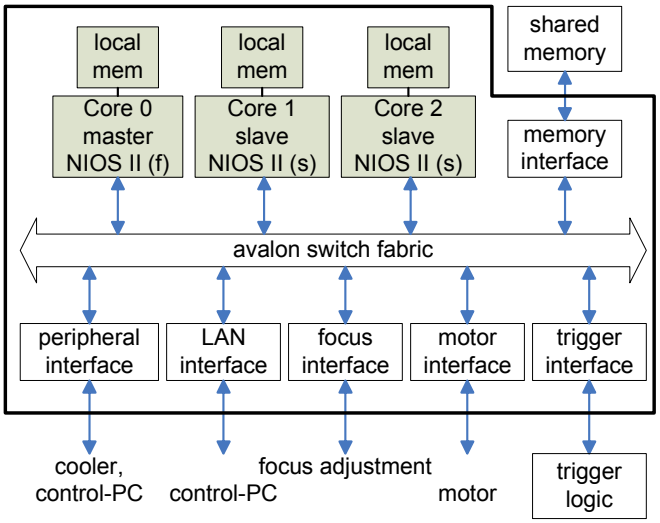

Fig. 4: Components of the NIOSII-MPSoC

In case basysPrint succeeds in the development of new exposing processes, the trigger-logic has to be properly adapted. The new logic will replace the old trigger-logic already implemented in the system.

This approach, as compared to the existing system, has the following distinct advantage: Any modification of the trigger-logic does not have any side-effects on the system-board, as long as the external interfaces of the trigger-logic remain fixed. Any new download of a hardware configuration file into the FPGA affects only the internal logic-elements of the FPGA.

The structure of the MPSoC proposed in this paper is an asymmetric shared memory architecture as all cores have access to the external memory device shown in Fig. 4. This enables communication between the various processor cores via globally accessible memory constructs. Since accesses to the shared memory are realized by simple load and store operations, it significantly simplifies communication efforts; i.e., this approach does not require any particular overhead, such as message queues, queue maintenance, or message processing.

The utilization of shared memory also allows dynamic changes in the executed software environment. For example, depending on the currently used printing plate type, core 0 can receive parameters or entire software segments by utilizing the LAN-Interface. After sending a notification, a different processor, e.g., core 1, can access the very same data without any further copying or processing operations. In that way, the flexibility for different purposes is increased as the system can adapt to various other plate types, which have varying characteristics and thus require slight modifications in the exposing process.

As can be seen in Fig. 4, all components are coordinated in the following way: Core 0, labelled "master", is responsible for maintaining status information and managing data transfer to and from the host-PC. It also controls execution of the two slave-cores, 1 and 2. The first slave processes all movement data generated by the attached trigger-logic. The

second slave-core operates as a peripheral controller supporting integrated system features, such as the focus adjustment, the cooler, and various other auxiliary components.

The MPSoC itself is created mostly by custom elements from ALTERA's system libraries. Only a few system-specific interfaces for focus control, motor-connection, and trigger-logic have to be designed by the developer. But even this work is supported by the ALTERA tools. Therefore, the design efforts for building interfaces are held at its minimum.

The system shown in Fig. 4 represents an expandable platform. Whenever the addition of further functionalities exceeds the current computational capabilities, the developer may solve this problem by adding another CPU. This way, the system performance can be adapted to any requirements. Especially time-restricted operations profit from this approach.

V. IMPLEMENTATION

A. Methods

The development discussed in this paper concentrates on the ALTERA NIOS development kit Cyclone™ Edition [11]. It features a Cyclone™ FPGA EP1C20 with approximately 20,000 programmable logic elements. The development board is equipped with a wide range of useful peripherals, such as 16MB SDRAM, 8MB flash-memory, an onboard Ethernet device, and several I/O-interfaces.

The Cyclone™ FPGA offers 300 configurable user I/O-pins, which accomplishes the application at hand. The price for such a device starts at approximately 100 US$ per device for a 24-device tray. Depending on the ordered volume, the price may be as low as 25 US$ per device.

As mentioned above, this research resorts on the configurable NIOS II CPU. "Configurable" in the context of the NIOS II CPU means that features, such as interfaces, on chip memories, watchdogs, and timers, can be freely added or removed. It is furthermore possible to augment the CPU's arithmetic logical unit (ALU) with dedicated hardware for special purposes, such as video- or audio-processing. This additional ALU-hardware is called "custom instruction". From the software-perspective, custom instructions appear as machine-generated assembly macros or C functions [3]. This approach yields significant performance improvements, since software functionalities can be transferred to physical hardware. A minimal configuration of a NIOS II processor consumes between 700 and 1800 logic elements and runs at a clock speed of up to 200 MHz.

ALTERA provides special hardware mutexes to simplify accesses to components shared by more than one processor. Such mutex performs an atomic test-and-set operation, in order to avoid concurrent memory access by two or more processors.

The creation of the processor system has been done with the help of ALTERA's SOPC-Builder. It provides a graphical user interface, which instantiates system features and handles both address and interrupt management. For more detailed information about ALTERAs SOPC-Builder, the interested reader is referred to the literature [6].

B. First Results and Critical Discussion

The first proof-of-concept implementation in the ALERA Cyclone FPGA takes about 9,000 logic elements, which is approximately 45% of the overall logic-resources. The clock generation is realized by the two onboard phase locked loops (PLL). In order to keep compilation time at a minimum, the core speed was set to 50 MHz and no optimization constraints were established during the development stage. When switching to the final release, the system will be optimized in order to gain higher clock rates, with an expectation of approximately 80 MHz.

The current development version already achieves the desired goals mentioned in Section IV; the aggregation of the trigger-logic and processing elements into an embedded system allows for shorter communication paths, reduced the number of separated system units, and made many of the wires connected to the host-PC dispensable. On a short term, this approach yields cost and maintenance improvements. On a long term, this modifiable adaptable platform is open to further improvements and also provides enough resources for additional features.

While fulfilling the given requirements, the prototypical implementation points to some issues, which are subject to current as well as future research. First of all, the central system bus, which handles all the interprocessor communications, might turn into a bottleneck. This potential problem should be appropriately addressed by both hardware and software designs. Current research is devoted to the development of simple and adaptable solutions [8, 9]. The prototype described in this paper uses the Avalon switch fabric for interprocessor communication purposes. In addition, the Avalon switch fabric improves the performance by its ability to support multiple bus masters. In the application at hand, it is possible to map software tasks onto separate CPUs, i.e., rather distributed than parallel processing, interprocessor communication is not a severe issue; other application profiles might require different communication mechanisms, such as message passing [7].

VI. FUTURE WORK

The current implementation is a prototype that represents a proof-of-concept. Future research will be devoted to the development of a complete platform, which will be converted to a production system by basysPrint. The remainder of this section sketches the most important hardware and software issues.

On the hardware side, the current VHDL code of the existing multiprocessor platform is to be enhanced by all the required interfaces. This includes a LAN-connection for

transferring image data, interfacing the peripherals, such as cooler, motors, and other auxiliary hardware, and connecting a CAN bus. Another major part consists of the integration of the existing trigger-logic functionality.

On the software side, the next steps are as follows: First of all, the existing software functionalities have to be ported to the new system. Then, the distribution of the data and communication objects, i.e., the memory map, should be optimized with respect to the system's runtime behavior.

After the completion of the steps mentioned above, a final field study has to prove both sufficient processing speed and fulfilment of the required timing behavior.

VII. CONCLUSION

This paper has presented the benefits of utilizing an FPGA to implement a multiprocessor platform for an industrial application in the field of offset printing. This paper has furthermore shown that FPGA-based solutions enable the usage of soft-core processors, which leads to flexible, scalable, and easy-to-maintain systems.

For the following reasons, the approach presented here might be of particular interests for industrial applications: It enables the efficient participation of future trends, such as dynamically reconfigurable hardware [10]; the support by powerful tools, the usage of an abstract hardware description language, and the option of employing IP-cores all result in significant time and money savings; since the particular system is not bound to a particular FPGA, i.e., the hardware can be ported to a different FPGA by a simple recompilation, it can be easily scaled if demands desire so.

Another point to consider, when working with FPGAs, is that hardware description, i.e., the VHDL code, remains at the development side and can furthermore not be recovered by third parties by reading the configuration file. That is, all system-related information can be kept confidential. This also reduces time and money costs during communication to third-party companies.

All the advantages described above are a strong argument for using FPGAs in industrial applications. Another equally important argument is that all the provided tools allow almost any developer to utilize FPGAs at very low personal and monetary costs.

VIII. ACKNOWLEDGMENTS

The authors gratefully thank Dr. Horst Steppat, managing director at basysPrint, and John Hedde, director of research and development at basysPrint, for providing the existing technology and the excellent cooperation. In addition the authors sincerely thank Jens Hildebrandt, former member of this research and now with Siemens VDO Automotive AG. This research was supported in part by the German federal ministry of education and research, grant number 03i4919B.

IX. REFERENCES

[1] Chris Wright and Mike Arens, "FPGA-based System-on-Module Approach Cuts Time to Market, Avoids Obsolescence", *FPGA and Programmable Logic Journal*, Volume VI, No. 6, Feb. 2005

[2] Salil Raje, "Catching the FPGA Productivity Wave", *Electronic Design*, Oct. 2004, Online ID 8931

[3] Altera Corp., *NIOS II Processor Reference Handbook*, Altera Document NII5V1-1.2, Altera Corp., San Jose, CA, Jan. 2005

[4] Gaisler Resarch, *LEON2 Processor User's Manual – XST Edition*, Gaisler Research AB, Goteborg, Sweden, Jan. 2005

[5] XILINX Inc., *MicroBlaze Processor Reference Guide*, XILINX Document UG081 (v5.0), XILINX Inc., San Jose, CA, Jan. 2005

[6] Altera *Corp., Quartus II Handbook Volume 4 - SOPC-Builder*, Altera Document QII5V4-1.0, Altera Corp., San Jose, CA, Feb. 2005

[7] David. E. Culler and Jaswinder Pal Singh, *Parallel Computer Architecture – A Hardware/Software Approach*, Morgan Kaufmann Publishers, Inc., San Francisco, California, 1999

[8] José Carlos Palma, Aline Vieira del Melo, Fernando Gehm Moraes, Ney Calanzans, "Core Communication Interface for FPGAs", *15th Symposium on Integrated Circuits and System Design*, Porto Allegre, Brazil, 2002

[9] Ross B. Ortega and Gaetano Borriello, "Communication Synthesis for Distributed Embedded Systems," in *Proceedings of the 1998 IEEE/ACM international conference on Computer-aided design*, pp. 437-444, 1998

[10] Reiner Hartenstein, "The Microprocessor is no more General Purpose: why Future Reconfigurable Platforms will win," in *Proceedings of the Second Annual IEEE International Conference on Innovative Systems in Silicon*, 1997

[11] Altera Corp., *Nios Development Board Reference Manual, Cyclone Edition*, Altera Document MNL-N2DEVLBDCYC-1.1, Altera Corp., San Jose, CA,

[12] basysPrint, *UV-Setters Series 7*, basysPrint Ltd., 2004