A Prototyping Platform for Dynamically Reconfigurable System on Chip Designs

Heiko Kalte, Mario Porrmann, Ulrich Rückert
Heinz Nixdorf Institute, University of Paderborn
Paderborn, Germany
E-mail: porrmann@hni.upb.de
WWW: http://wwwhni.upb.de/sct and http://www.RAPTOR2000.de

Abstract

This paper presents a flexible modular rapid prototyping environment for microelectronic system and circuit designs. The prototyping hardware consists of a PCI bus motherboard and up to six modules that can carry any kind of functional components (CPU, memory, FPGAs). Especially aspects of dynamic reconfiguration of today's high density FPGAs have been considered. Therefor a configuration manager has been implemented in hardware in order to provide minimum reconfiguration times during runtime. Additionally, a software environment for the rapid prototyping system and several applications that have been implemented utilizing this system are presented within the paper.

I. Introduction

Programmable logic is about to take over a central role in electronic systems. System-on-chip designs with a complexity of more than a million logic gates and several hundred kBytes of internal SRAM memory can be mapped on state-of-the-art Field Programmable Gate Arrays (FPGAs). Clock rates approach several hundred MHz boosting the chip-computational power above 10^4 MOPS (million operations per second) at a power consumption of a few watts. Today's FPGAs, shipped by the two leading programmable logic device companies Xilinx and Altera, provide 10 million system gates composed of memory, datapath elements and programmable logic gates. All functional units can be partially reconfigured during runtime which enables a new dimension of flexibility.

This system level complexity enables the integration of a whole system on one programmable device, also called System-on-Programmable-Chip (SoPC). SoPCs can include multiple processors, DSPs, high speed system busses, memory, peripheral components and a variety of application-specific standard products [1]. Xilinx lately announced the new 'Platform FPGA' series, the Virtex-II family, which targets board range applications formerly integrated as System-on-Chip (SoC). In contrast to SoC devices system-level FPGAs combine the advantage of large capacity and the flexibility enabled by the configurability. Previously, designers were relegated to use ASICs which involves fixed functionality, high non-recurring engineering as well as re-spin costs and strict requirements on minimum order quantities. There is a trend away from fixed system-on-chips towards highly flexible SoPC solutions with improved time-to-market.

In order to simplify the development of system-level designs and to utilize Platform FPGAs in a host computer environment, we developed the flexible rapid prototyping system RAPTOR2000. The system and one of its main features, the Configuration Manager that provides high performance dynamic reconfiguration of the system internal FPGAs, is described in the following (see also http://www.RAPTOR2000.de).

II. RAPTOR2000 - Prototyping Environment

The flexible concept of the RAPTOR2000 system consists of a motherboard and up to six application specific modules. Basically the motherboard provides the communication infrastructure for the functional components that are implemented on the application-specific modules (ASM). The communication infrastructure includes a PCI bus link to a host computer, that provides a wide range of facilities and permits the verification of prototypical implementations embedded in a host processor environment. Due to the easy extensibility by new modules, the system can be quickly adapted to almost any requirements. These aspects enable system level designs and reduce development costs and time to market.

II.1 Motherboard

The RAPTOR2000 motherboard forms the basis of the rapid prototyping system. It provides several interconnection and communication facilities within the RAPTOR2000 system and to the host computer. Additionally, management functions like bus arbitration, memory management and error detection services are integrated into two Complex Programmable Logic Devices (CPLD). The services are structured into Communication, Configuration, Management and Monitoring.

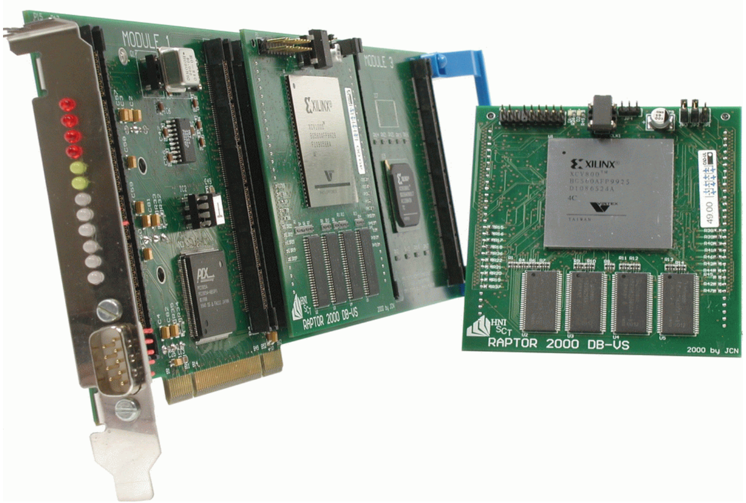


Figure 1: Photo of the RAPTOR2000 system and two FPGA modules

The block diagram of the motherboard in figure 2 depicts the various **Communication** options that can be utilized by the ASMs. Every ASM slot is connected to the *Local Bus* for internal communication with other devices or modules and for external communication with the host processor or other PCI bus devices. Therefore, a PCI bus bridge that enables single and burst transfers in master and slave mode as well as direct memory accesses (DMA) has been integrated into the RAPTOR2000 system. An additional *Broadcast Bus* can be used for simultaneous communication of any module to all other modules. Furthermore, these Broadcast Bus signals can be used for direct connections between the ASMs. Beyond this, the dual port SRAM can be accessed by all ASMs via the Broadcast Bus. The SRAM can be used e.g. as a buffer for fast direct memory access to the main memory of the host computer. Therefore, a module can transfer data via the Broadcast Bus to the SRAM memory and afterwards or concurrently the PCI bus bridge transfers the data via DMA transfer to the host computer. Theoretically, data transfer rates of up to 132MByte/s between the RAPTOR2000 system and the host computer are possible; measurements have proven average data transfer rates of more than 100MByte/s when using DMA transfers. Another communication channel is realizable by the application specific direct module interconnection wires between the ASMs. 128 direct connections are available from each ASM to its right and left neighbor. All six ASMs are arranged in a ring. If the Broadcast Bus signals are not used, these additional 75 signals lead to a maximum of more then two hundred direct interconnections between two ASMs. Assuming a rather low clock frequency of 50MHz leads to a data transfer rate beyond 1GByte/s between two ASMs. Another crucial aspect, especially concerning FPGA designs, is the **Configuration** of the programmable devices. Each ASM that carries an FPGA has to be configured by an application specific data stream that determines the function of the device. This procedure has to be executed before the first use (after power up) and every time the functionality should be changed. In order to utilize aspects of dynamic (during runtime) reconfiguration, it is necessary to minimize this reconfiguration time. The ASM that is presented in the next chapter embodies a Xilinx Virtex FPGA. Xilinx provides several serial and parallel configuration methods [2]. The fastest parallel mode and the serial JTAG configuration mode have been implemented into the RAPTOR2000 system. Especially the parallel configuration mode establishes some new facilities. The whole configuration algorithm has been implemented in Hardware. The concept is a further development of a former project that already implemented a Configuration Manager in hardware [6]. In order to reconfigure an FPGA located on one of the six modules, simply the base address of the configuration stream has to be determined. After transferring the reconfiguration command, a state machine starts to transfer the reconfiguration data via the Local Bus from the determined address to an internal register. Thereafter, the data is preprocessed and transferred to the desired device. The interconnection between the configuration state machine and the FPGA is independent of the Local Bus that allows reaching best performance levels. The implementation of the reconfigured algorithm does not distinguish if the data stream source is located within the RAPTOR2000 system or not. That means, configuration data that is stored e.g. in a system internal SRAM can be used as well as data that is located in the host computer's main memory. Furthermore, any device (Local Bus devices, host devices, PCI bus devices) can start the reconfiguration process for any module. This aspect enables entirely new possibilities; it is for example possible that an FPGA autonomously reconfigures itself by configuration data that is located anywhere in the system. One possible scenario is to have a basic configuration that modifies itself by dynamic partial reconfiguration dependent on the current application or requirements. This means no more external commands of the host processor are necessary to reconfigure the FPGA; the FPGA decides itself when to be reconfigured and which bitstream to be used.

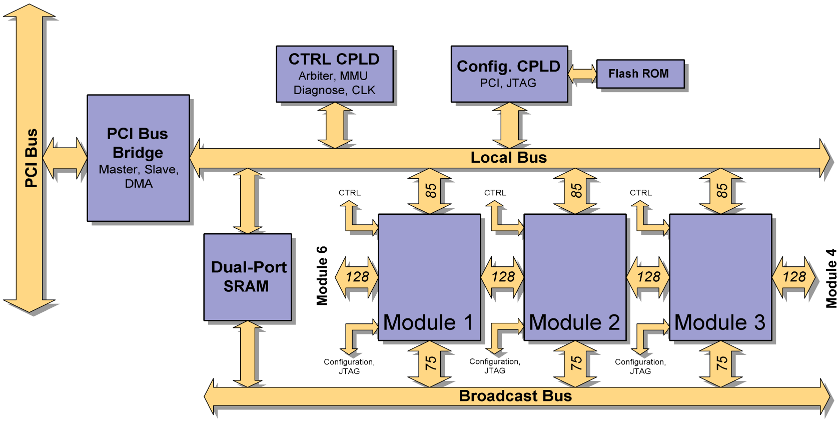


Figure 2: Architecture of the RAPTOR2000 rapid prototyping system

Additional monitoring and error detection methods permit extended diagnosis by a software tool during and after the reconfiguration process. Due to the hardware implementation of the reconfiguration algorithm a Xilinx Virtex 1000 FPGA can be reconfigured within about 15ms which is the minimum time permitted by the configuration interface of Xilinx. The RAPTOR2000 system being equipped with six Virtex 1000 ASMs would lead to a reconfiguration time of 90ms (about 4,6MByte configuration data). The configuration algorithm implemented in hardware also supports the partial reconfiguration of the system. This leads to much smaller reconfiguration times depending on the amount of logic that has to be reconfigured.

The access control for the intra-system resources is realized by a **Management** unit. It manages the Local Bus accesses of all master and slave devices of the RAPTOR2000 system. More precisely, the management unit controls the permission to initiate a Local Bus access for each master (bus arbiter) and assigns Local Bus addresses to the different devices and ASMs (memory management unit). In order to avoid system crashes caused e.g. by an access to an incorrect memory area, a **Monitoring** unit has been developed. This unit detects and corrects errors and allows diagnosis by a software tool that has been developed especially for error cases.

II.2 Modules

Various ASMs have been designed and integrated into the RAPTOR2000 system. As an example, for prototyping an SoC that integrates the functionality of an ethernet switch and an embedded RISC processor, three ASMs have been developed, that integrate the switching functionality, the physical layer and the RISC processor, respectively. All modules can be used for a wide range of different tasks and are not limited to the presented application.

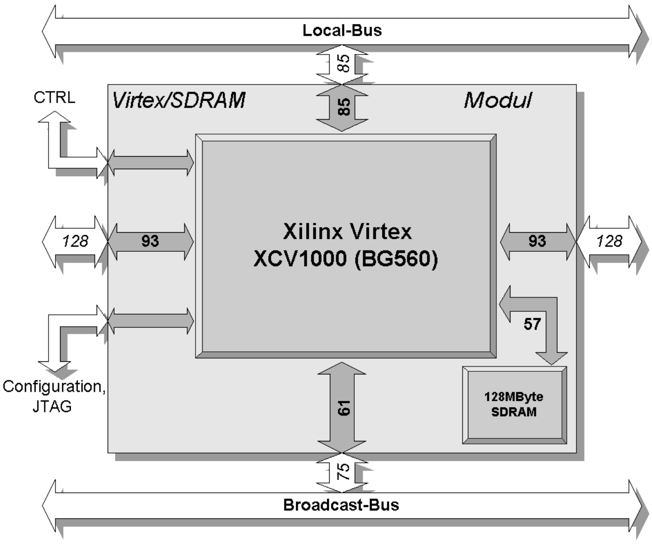


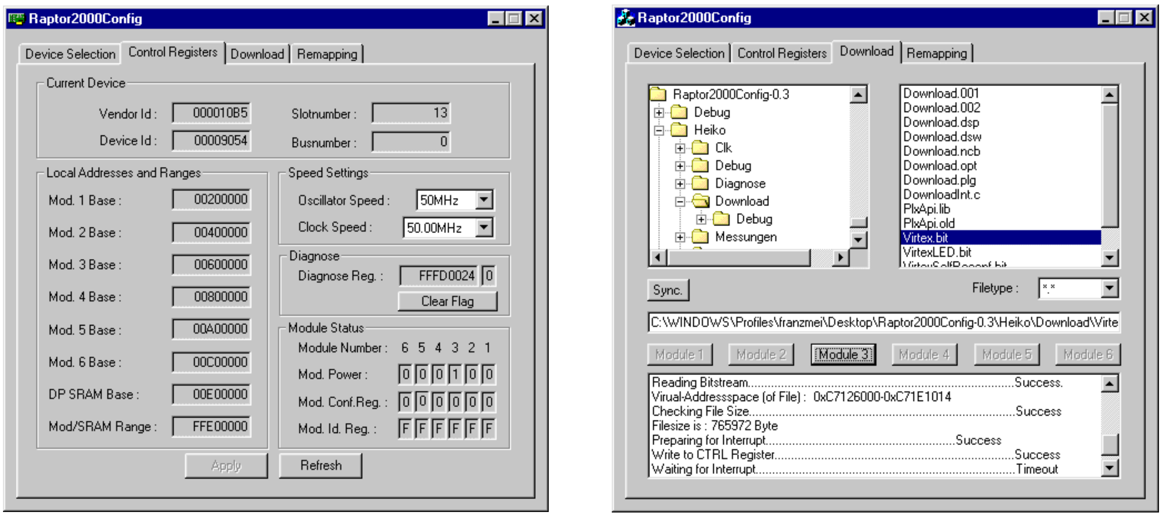
Figure 3: Architecture of the DB-VS FPGA module

As a general purpose ASM for application specific circuit designs, the module DB-VS has been developed for the RAPTOR2000 system. This ASM embodies an FPGA of the Xilinx Virtex (-E) series and optionally up to 128 MBytes of SDRAM. The gray colored block represents the daughterboard and the outer arrows represent the connection and communication infrastructure that are provided by one module socket of the motherboard. To enable the wide range of possible communication facilities, the FPGA is connected to the Local Bus and to the Broadcast Bus (top and bottom of figure 3) and to both neighbor modules (right and left side of figure 3). Due to the limited number of user IOs that are provided by the FPGA (400 Pins), not all available signals of the motherboard can be utilized by this ASM. Several additional control and configuration signals have been connected to the FPGA in order to enable the motherboard’s management and configuration services. Up to 128MByte of SDRAM are accessible by a controller. An appropriate controller core has been implemented into the Virtex device. The DB-VS can be equipped with various chips, that emulate circuits with a complexity of 400,000 up to 4 million system gates.

II.3 Software

To offer comfortable use of the whole RAPTOR2000 rapid prototyping system we developed a Graphical User Interface (GUI) that provides the administration of multiple RAPTOR2000 motherboards and ASMs. All important registers of the PCI bus bridge, the MMU, the Administration and the Configuration Manager can be visualized and modified (see figure 4). In order to download a bitstream simply a file and a module to be configured have to be selected. Afterwards all necessary settings are adjusted and finally the Configurations Manager is started. The right screenshot of figure 4 depicts the download area of the software.

For easy access to any application specific register implemented in an FPGA the software provides additional read/write operations to any module or the dual port SRAM. This feature enables e.g. the download of a program for a processor that is implemented in an FPGA.



Vendor Id : 000010B5
Device Id : 00009054
Slotnumber : 13
Busnumber : 0

Mod. 1 Base : 00200000
Mod. 2 Base : 00400000
Mod. 3 Base : 00600000
Mod. 4 Base : 00800000
Mod. 5 Base : 00A00000
Mod. 6 Base : 00C00000
DP SRAM Base : 00E00000
Mod/SRAM Range : FFE00000

Oscillator Speed : 50MHz
Clock Speed : 50.00MHz

Diagnose Reg. : FFFD0024 0

Module Number :	6	5	4	3	2	1
Mod. Power :	0	0	0	1	0	0
Mod. Conf.Reg. :	0	0	0	0	0	0
Mod. Id. Reg. :	F	F	F	F	F	F

Raptor2000Config-0.3
- Debug
- Heiko
  - Clk
  - Debug
  - Diagnose
  - Download
    - Debug
  - Messungen

Download.001
Download.002
Download.dsp
Download.dsw
Download.ncb
Download.opt
Download.plg
DownloadInt.c
PlxApi.lib
PlxApi.old
Virtex.bit
VirtexLED.bit

Filetype : *.*

C:\WINDOWS\Profiles\franzmei\Desktop\Raptor2000Config-0.3\Heiko\Download\Virte

```
Reading Bitstream.......................................................................Success.
Virual-Addressspace (of File) :  0xC7126000-0xC71E1014
Checking File Size......................................................................Success
Filesize is : 765972 Byte
Preparing for Interrupt.........................................................Success
Write to CTRL Register..................................................................Success
Waiting for Interrupt...................................................................Timeout
```

Figure 4: Two screenshots of the RAPTOR2000 Graphical User Interface (GUI)

Currently we are developing a Java applet that provides control of the RAPTOR2000 board via internet. The concept consists of Java network communication via Remote Method Invocation (RMI) between the Java applet and a server application that is running on the RAPTOR2000 host computer. In order to access the hardware we are using the Java Native Interface (JNI). This concept permits all circuit and systems designers to implement a design on the RAPTOR2000 system from a computer anywhere in the world.

II.3 Applications

Various neural network models have been implemented, utilizing the RAPTOR2000 system as a hardware accelerator. In particular, designs for radial basis function networks, binary associative memories and self-organizing feature maps have been developed [7,8]. For self-organizing feature maps, the dynamical reconfiguration capability that is provided by the RAPTOR2000 system can be used to adapt the size of the map (i.e. the number of neurons) that is simulated in order to achieve optimal performance [3,5]. Additionally, dynamic reconfiguration is used to adapt the precision of the processing elements. Starting with a low precision (e.g. 8 bit), a rough ordering of the map can be achieved. For fine tuning of the map the precision of the processing elements is increased (e.g. to 16 bit), by loading a new configuration file. Another case study focuses on the implementation of octree based 3D graphics [4]. To accelerate an octree based CAD tool, the most time consuming algorithms have been implemented in several hardware configurations. Depending on the algorithm, that is actually required by the user, the appropriate hardware configuration is downloaded to the RAPTOR2000 system during runtime.

III. Conclusion

The architecture of the flexible rapid prototyping system RAPTOR2000 has been presented. It was developed to meet the requirements of today's high density system and circuit design. The systems's motherboard provides six module sockets for functional components, a PCI bus link to a host computer, a powerful communication infrastructure and a hardware implementation of a configuration manager that is capable of configuring the system internal FPGAs in the minimum reconfiguration time. This allows to use the system not only as a prototyping platform, but also as a universal coprocessor board for reconfigurable computing. Additionally, the software environment of the RAPTOR2000 system and several applications have been presented.

Acknowledgement

This work has been partly supported by the Deutsche Forschungsgemeinschaft (German Research Council) DFG Graduate Center "Parallele Rechnernetzwerke in der Produktionstechnik".

References

[1] H. Kalte, D. Langen, E. Vonnahme, A. Brinkmann, U. Rückert: *Dynamically Reconfigurable System-on-Programmable-Chip*, Proceedings of the 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing (PDP 2002), January 9th-11th, 2002, Gran Canaria Island, Spain.

[2] Xilinx, Inc.: *Virtex-II Platform FPGA Handbook (v1.3)*, December, 2001.

[3] M. Porrmann, U. Witkowski, H. Kalte, U. Rückert: *Implementation of Artificial Neural Networks on a Reconfigurable Hardware Accelerator*, 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing (PDP 2002), 9.-11. Januar 2002, Gran Canaria Island, Spain.

[4] H. Kalte, M. Porrmann, U. Rückert: *Using a Dynamically Reconfigurable System to Accelerate Octree Based 3D Graphics*, PDPTA'2000, June 26-29, 2000 Monte Carlo Resort, Las Vegas, Nevada, USA, 2000.

[5] M. Porrmann, H. Kalte, U. Witkowski, J.-C. Niemann and U. Rückert: *A Dynamically Reconfigurable Hardware Accelerator for Self-Organizing Feature Maps*, In Proceedings of The 5th World Multi-Conference on Systemics, Cybernetics and Informatics, SCI 2001, Volume 3, Orlando, Florida, USA, July 2001 (Best Paper Award).

[6] H. Kalte, M. Porrmann, U. Rückert: *Rapid Prototyping System für dynamisch rekonfiguri-erbare Hardwarestrukturen*, AES2000, Karlsruhe, 18.-19. Jan. 2000.

[7] M. Porrmann, M. Franzmeier, H. Kalte, U. Witkowski and U. Rückert: *A reconfigurable SOM hardware accelerator In Proceedings of the 10th European Symposium on Artificial Neural Networks*, ESANN'2002, Bruges, Belgium, April 2002, to be published.

[8] M. Porrmann: *Leistungsbewertung eingebetteter Neurocomputersysteme*, PhD thesis, University of Paderborn, HNI Verlagsschriftenreihe Band 104, ISBN 3-935433-13-1, 2002.