

Exploitation of Run-Time Partial Reconfiguration for Dynamic Power Management in Xilinx Spartan III-based Systems

Katarina Paulsson, Michael Hübner, Salih Bayar, Jürgen Becker
Universität Karlsruhe (TH), Germany
<http://www.itiv.uni-karlsruhe.de/>
{paulsson, huebner, becker}@itiv.uni-karlsruhe.de
salih_bayar@yahoo.com

Abstract

Reconfigurable architectures such as FPGAs are a key technology for implementing self-adaptive and flexible systems, since the possibility for dynamic and partial hardware reconfiguration offers a higher degree of freedom in the resource allocation. However, for many applications, modern reconfigurable architectures cause too high power consumption. For this reason, FPGAs are typically used in highly computational applications where the possibility for performing parallel calculations can be exploited to achieve decreased power consumption as compared to a microcontroller based solution. In order to enable the usage of FPGAs even in low- power applications, techniques for adapting the power consumption according to the work load must be investigated. One novel approach would be to exploit the possibility for dynamic reconfiguration of the FPGA for this purpose. This paper presents the realization of dynamic clock scaling in a Xilinx Spartan 3 based system by performing dynamic and partial reconfiguration of the fixed Digital Clock Manager (DCM) core configuration.

Keywords: *Dynamic power management, low power applications, reconfigurable hardware, self-optimizing systems*

1. Introduction

Low power applications are traditionally implemented on power optimized microcontrollers in order to manage the strict power budget while maintaining the system requirements. Reconfigurable architectures such as FPGAs are normally not well suited for such applications due to higher power dissipation. For this reason FPGAs are often used in highly computational applications where the possibility for performing parallel processing on the FPGA architecture can be exploited to decrease the computation time and thereby enable decreased power consumption. The usage of FPGAs is in many cases an advantage because of the high flexibility and simplified

System-on-Chip design flow based on IP cores. The integrated tools for hardware/software descriptions enable combined simulation, verification and test of the different system components.

In addition to this, several FPGA architectures such as Xilinx Virtex 2, 4 and 5, now also offer the possibility to perform dynamic and partial hardware reconfiguration. By exploiting this feature, smaller chip sizes can be used which also leads to decreased static power consumption [2]. Still, in order to allow the usage of FPGAs even for low power applications, methods for optimizing the dynamic power consumption must be investigated.

Clock scaling is a common approach in ASIC designs or in microcontroller based systems for adjusting power consumption dynamically. On FPGAs however, this method is not as straightforward due to the internal clock network structure. Also, the clock managers that are available as fixed cores on several architectures (e.g. Virtex 2, 4 and Spartan 3) are not designed to allow scaling the clock frequency dynamically.

One approach to realize clock scaling on Xilinx FPGAs is to implement an additional frequency divider. This can easily be done by implementing a controller for tuning the frequency dynamically. The disadvantage with this approach however, is the increase in resource utilization caused by the frequency divider. In addition, a DCM core should be integrated anyway in order to generate a more accurate clock signal, which further increases the resource utilization.

An alternative approach to realize clock scaling on Xilinx FPGAs is to use the fixed DCMs only and perform the dynamic adaptation of the clock frequency by reconfiguring the configuration of the DCM. This way, the partial reconfiguration process would be exploited to control power consumption of the FPGA and no additional frequency divider is required. An on-chip configuration interface will be required, but in a system which exploits dynamic and partial reconfiguration for other purposes this is already included.

This paper describes the implementation of dynamic clock scaling by exploiting partial reconfiguration in a low- power level measurement system based on a Spartan 3 FPGA. The paper is organized the following way: section 2 describes the sources of power consumption in

FPGAs. Section 3 gives an overview of the capacity based level measurement system which was the target application for this work. In section 4, the realization of clock scaling by exploiting partial reconfiguration is presented and the results from the real measurements of the power consumption are given in section 5. Finally, the paper is concluded in section 6.

2. Power consumption in FPGAs

FPGAs normally cause higher power dissipation than ASICs due to the many programmable connections in the architecture [1]. For many data oriented applications however, FPGAs still offer the most power optimized solution, considering the implementation possibilities on a FPGA in comparison to a microcontroller.

According to Altera [9], the dynamic power consumption is a major part in typical high performance FPGA designs. The dynamic power consumption is caused by the switching of transistors in the FPGA implementation and therefore increases proportional with the clock frequency but also in dependence of the transistor switching activity and the capacity in the design. In [12], an approach for optimizing the routing of a design by considering the capacity and timing constrains of the signal lines is presented.

The static power consumption is caused by leakage currents in the transistors. As written above, in a typical high- performance FPGA- based design today, using 90-130 nm technologies, dynamic power consumption is usually the largest component. However, with decreasing technologies, the transistor leakage currents will increase and therefore also the static power consumption. For this reason the FPGA vendors and researchers [13] [14] are now focusing on methods for keeping the static power consumption low even for future technologies.

Some FPGA vendors also offer the possibility of dynamic and partial hardware reconfiguration, which is one way for the designer to decrease static power consumption. Exploiting reconfiguration can allow using a smaller chip size and this way achieve lower static power dissipation. One system which is based on this future is presented in [11]. Dynamic and partial reconfiguration of FPGAs also causes extra power dissipation during the reconfiguration. This is presented in [2], where the power consumption is measured during the run-time reconfiguration.

3. The Spartan III- based measurement application

In the work described in this paper, clock scaling was interated in an FPGA based low power level measurement application by exploiting dynamic and partial

reconfiguration. The target application is normally based on a low- power microcontroller, but has been implemented on FPGA using a MicroBlaze [7] soft-core processor for processing the data. The reason for investigating FPGAs for this kind of application was to see if dynamic and partial reconfiguration can be exploited in the future in the overall system for increasing the system flexibility and adaptivity to changing environmental requirements. This part of the work will however not be further described in this paper.

The basic functionality of the system can be described the following way: the FPGA system generates an 8- bit wide digital signal, which is converted into an analog sinus signal by an external DA converter. The signal is then applied to the tank in which the level measurement is performed. Every 100 ms the current is detected back from the tank and converted into digital data by an external AD-converter. The digital data is then processed by the MicroBlaze processor and the level of material in the tank is calculated. An overview of the complete system is given in Figure 1.

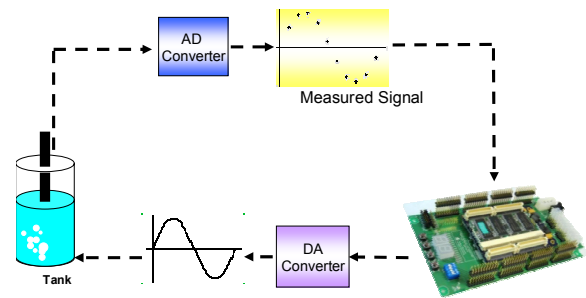


Figure 1. FPGA based level measurement system

The processing cycle is repeated every 100 ms with MicroBlaze reading the data and processing it. After the data processing has been completed and until the next cycle, MicroBlaze does not need to be active. This behavior can be exploited when implementing clock scaling in this system.

The scheme presented in Figure 2 was used to control the clock scaling for the specific level measurement application. This control flow was implemented as a small state machine (the ADC- controller) in VHDL, with focus on keeping the core as small as possible. The logic was then attached to a On- Chip Peripheral Bus (OPB) [5] slave interface. The Xilinx Embedded Development Tools Kit, EDK [3], and the Integrated Software Environment, ISE [6], were used for the entire system implementation.

The Spartan III FPGA [4] architecture, which was used for this application, includes 4 Digital Clock Managers implemented as fixed cores that can be used for adjusting the input clock frequency. These clock managers are not designed for scaling the output frequency during run-time.

The later part of this paper will describe how dynamic and partial reconfiguration can be used to scale the output frequency of these clock managers.

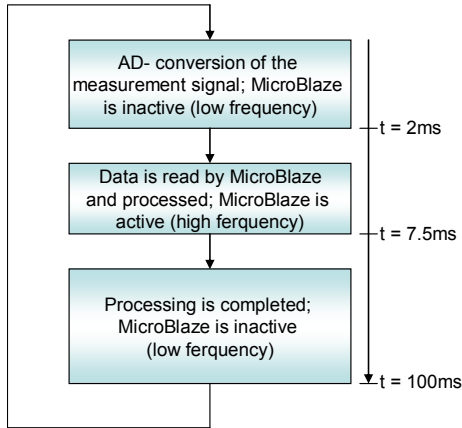


Figure 2. Measurement cycle

4. Run-time partial reconfiguration for dynamic power management

Some of Xilinx FPGAs, e.g. Virtex 2, 4 and 5, offer the possibility of dynamic and partial hardware reconfiguration. This means that parts of the hardware resources are dynamically reprogrammed in order for the system to adapt to changing system requirements.

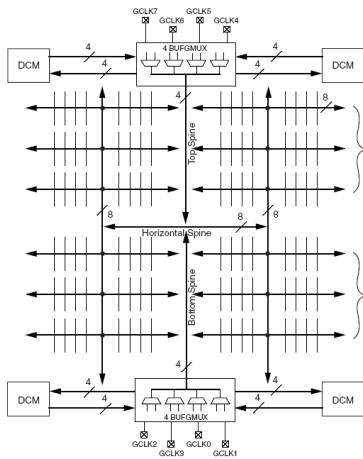


Figure 3. Spartan III clock network structure [4]

Typically this feature has been exploited for reconfiguring complete functions [11] whenever they are required by the system. Several constraints must be considered while performing dynamic and partial reconfiguration. For example, the FPGA resources must be partitioned into static and dynamic regions, where the static regions contains the components which are constantly required

throughout system operation and the dynamic regions include the components which can be dynamically reconfigured. Also, for the communication between the reconfigurable regions, so called bus macros [11] must be used. This is important since the communication lines between the reconfigurable regions must be on the exact same positions for the different reconfigurable modules, otherwise signal lines could be cut during reconfiguration.

However, not only the logic and the routing resources can be reconfigured dynamically. The content of the BRAMs as well as the configuration of the fixed DCM cores may also be reconfigured, which is exploited in this work to implement clock scaling.

Figure 3 shows the structure of the Spartan 3 clock network. The smaller Spartan 3 FPGAs only include 2 DCMs. In Figure 3, the FPGA architecture includes 4 DCMs, 1 located in each corner of the FPGA. The clock outputs of the DCMs can be connected to the FPGA clock tree over clock input buffers. The clock tree structure and the clock buffers help generating a more accurate clock signal and to prevent e.g. clock skew.

By reconfiguring a DCM in order to scale the frequency dynamically, all the advantages of the DCMs as well as the clock tree structure can be exploited without using an additional frequency divider.

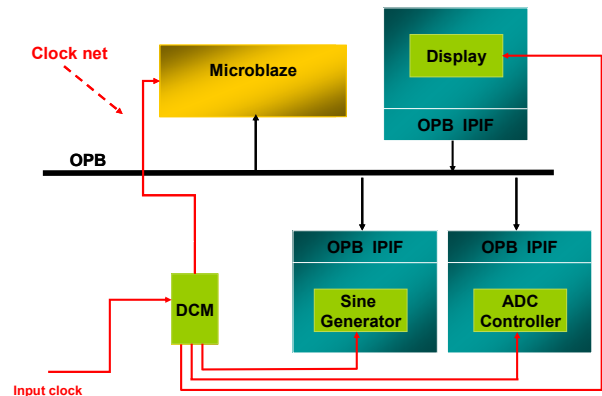


Figure 4. FPGA system for level measurement application

In order to implement clock scaling following this approach, the system must be designed in a specific way. First, 2 implementations of the same system with different DCM configurations must be generated. It is important that both implementations are identical except for the DCM configurations, in order to simplify the generation of the partial bitstreams for reconfiguration. Then the Xilinx tools for generation of partial bitstreams can be used to “cut” out the bits containing the DCM configuration and to generate a partial bitstream. Exactly how this is done will be described in the following parts of this paper.

An overview of the FPGA system is given in Figure 4.

The Sine Generator and the ADC controller were designed in VHDL and used as IP cores in the FPGA system. The DCM component takes the input clock signal and generates multiple output clock signals which are connected to the system components. MicroBlaze is the only component of which the frequency is scaled, the other components run with the same frequency throughout system operation. When the user-specific cores have been imported into the Xilinx Embedded Development Tools Kit, EDK, as IP cores, the complete system with its hardware and software components can be modeled. However, in order to make sure that the system implementation is identical for the systems with the different DCM configurations, the complete system must be exported into the Xilinx Integrated System Environment, ISE, for generation of the hardware bitfiles.

First, the system was implemented with the DCM core configured for generating a high clock frequency for MicroBlaze, 32 MHz. The system was then imported into ISE where a hardware bitfile was generated. This bitfile could then be imported back into EDK and updated with the system software in order to generate the complete bitstream for programming the FPGA. While generating the hardware bitfile, the Native Circuit Description, NCD file, which is generated by the ISE tools, was saved for later usage when implementing the system with the second DCM configuration.

The second system was modeled by simply adjusting the DCM configuration of the first system to generate a clock frequency of 4 MHz to the MicroBlaze. The clock frequencies to the other components stay the same. After exporting the second system into the ISE environment, the Place- and Route process was performed in "Exact" mode. This way, the earlier generated ncd file could be used as a model and the second implementation was generated to be as similar as possible to the first implementation. Then only the DCM configuration differs between the 2 implementations.

After performing the above design steps, 2 full bitstreams, one with MicroBlaze running at full speed and one with MicroBlaze running with a lower clock frequency, were generated. These files were then used to generate the partial bitstreams containing the DCM configurations.

Tools for generating partial bitstreams are documented for the Xilinx Virtex 2 / 2 Pro FPGAs [15]. There, the tool flows for Difference Based bitstream generation as well as for using the PartialMask tools are described. The Difference Based design flow takes two ncd- files as inputs and generates a partial bitstreams which only includes the difference between the 2 designs. This design flow is optimal when the changes in the design are minor and no reconfiguration of complete functions is to be performed. The second tool flow is the so called PartialMask. The PartialMask feature allows the user to

select which hardware resources shall be included in the generated partial bitstream. The different resources on the FPGA can thereby be addressed and "cut" out to generate the bitstreams. For example, the specific CLBs, IOBs and BRAMs can be specified.

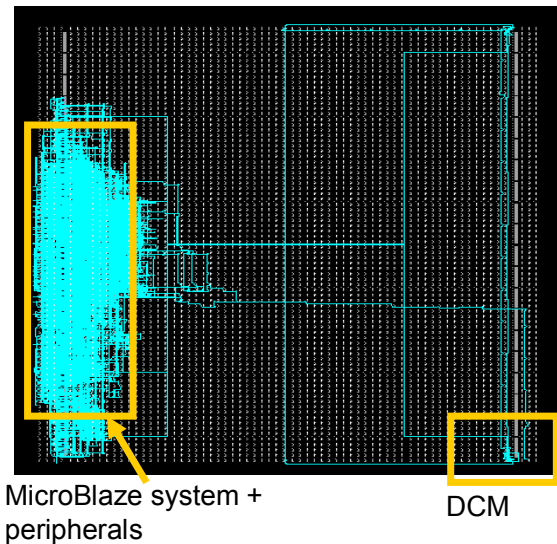


Figure 5. System implementation viewed in the FPGA Editor

In work presented here, the PartialMask tool was used for generating the partial bitstreams, but the Difference Based approach would be somewhat more suitable. In order to generate a partial bitstream with the DCM configuration by using the PartialMask tool, the complete configuration of the BRAM block placed above / underneath the specific DCM was addressed and included in the bitstream. This approach could lead to difficulties if the BRAMs are used to store program code or data, since the run-time content would be overwritten with the initial data during the reconfiguration which could lead to data loss. For this specific target application however, the BRAM block above the used DCM was empty so the reconfiguration did not cause any problems. This can be seen in Figure 5, which gives an overview of the system implementation in the FPGA Editor. In any case, the Difference Based tool would be more suitable to use since it would cut out the DCM configuration only which would lead to a smaller size of the partial bitstreams and decreased reconfiguration time.

5. Measurements and results

The system was implemented on a Xilinx Spartan III-2000 based RP board. The Spartan III 2000 FPGA is much too large for the target system, but since these measurements of power consumption were performed to

generate data for comparison, it was not required that the target FPGA size was used. Since the board does not provide the possibility of measuring power consumption of the FPGA itself, the power dissipation of the entire board was measured.

A voltage of 9 V was applied to the board, and power consumption was measured for 2 different frequencies for MicroBlaze, 4 and 32 MHz, which were dynamically generated by dynamically configuring the DCM core.

As mentioned in this paper, rewriting the configuration memory causes increased power consumption during the reconfiguration.

Table 1 Measured power consumption

	Static Power Consumption	Dynamic Power Consumption
Frequency: 32 MHz	1292 mW	461 mW
Frequency: 4 Mhz	1292 mW	311 mW
During reconfiguration, 32 MHz → 4 MHz:	1292 mW	507 mW

Therefore, in order to evaluate this approach for integrating clock scaling on FPGAs, power dissipation during the reconfiguration was also measured. Power consumption was measured with a digital oscilloscope for the 2 different frequencies and during reconfiguration. However, in order to evaluate the efficiency of this approach, it was necessary to determine the reconfiguration time as well. There is no internal configuration port available on the Spartan III FPGAs, which means that the reconfiguration had to be performed externally over the JTAG cable. By using the specified data rate for the Xilinx Parallel IV JTAG cable [16] the reconfiguration time could be estimated to approximately 34 ms.

The size of the partial bitstreams is 21 kByte and the reason for the very long configuration time was that the external JTAG configuration port was used. Using an internal reconfiguration port would be significantly faster. As can be seen in Table 1, power consumption increased to approximately 1799 mW during the reconfiguration of the DCM for high to low frequency. In Table 1 all results from the measurements of power consumption are presented. Please note that the measurements represent the power consumption of the entire board, not just the FPGA.

An expression for calculation the usefulness of this method for a certain application follows:

$$P_{cycle} = \frac{E_{high_freq} + E_{reconf_high_to_low} + E_{reconf_low_to_high} + E_{low_freq}}{t_{cycle}}$$

where

$$E_{high_freq} = t_{high_freq} * P_{high_freq}$$

$$E_{reconf_high_to_low} = t_{reconf_high_to_low} * P_{reconf_high_to_low}$$

$$E_{reconf_low_to_high} = t_{reconf_low_to_high} * P_{reconf_low_to_high}$$

$$E_{low_freq} = t_{low_freq} * P_{low_freq}$$

As can be seen in the above expression, the efficiency of this method is heavily depending on the reconfiguration time. In our example the reconfiguration time was very long, 34 milliseconds, for such a small bitstream. The reason for the long reconfiguration time is the usage of an external configuration port. By performing dynamic reconfiguration over the internal configuration port, the ICAP, the reconfiguration time could be dramatically decreased which leads to a reduced power consumption. The ICAP writes data to the configuration memory with a performance of 66 MBytes/s (for Virtex II FPGAs), which means that the reconfiguration of a bitstream with the size 21 kByte would take 0.32 milliseconds. As an example; assume that the cycle time is 100 ms, the active time is 7 ms and that the reconfiguration time is 0.32 milliseconds. Then the power consumption of one cycle would be (using the data from Table 1):

$$P_{cycle} = \frac{1753*7 + 1799*0.32 + 1649*0.32 + 1603*92.36}{100} \approx 1614mW$$

The power consumption for the same cycle time when not using clock scaling would be 1753 mW, which means that the clock scaling reduces power consumption with approximately 139 mW on the complete board. Of course this is just an imaginary example of how much power consumption could be decreased if an ICAP port was available on the Spartan 3 FPGA and does not consider all parameters which would influence power consumption. The Spartan 3A family includes an ICAP interface [10], but real measurements of power consumption must be performed specifically for this FPGA family in order to come to a realistic conclusion on the reduction of power consumption.

Naturally, the decrease in power consumption will be larger if a smaller bitstream is used for the partial reconfiguration, which is the reason for why the Difference Based feature for generating the partial bitstreams would lead to optimized results.

Also, it is important to remember that the results here were measured on a FPGA which is significantly larger than the system implementation requires. The static power consumption of the entire board was measured to 1292 mW but this component could be reduced if a smaller

FPGA was used. The reduction of dynamic power consumption by exploiting reconfiguration would thereby have a larger effect on complete power dissipation.

6. Conclusions

This paper presents one approach for integrating clock scaling for dynamic power management in Xilinx FPGA based systems by exploiting the dynamic and partial hardware reconfiguration. It makes sense that the available fixed Digital Clock Manager cores are used for dynamic clock scaling alone if it is possible, since resources are saved when it is not required to implement an additional controller for controlling the frequency dynamically. Naturally additional components (e.g. internal configuration interface and external memory) for performing the reconfiguration are required, but in a system which exploits dynamic and partial reconfiguration for other purposes, these components are already included.

Real measurements show that dynamic power consumption may be decreased with this approach; how much power is saved depends on the application and the target architecture. The presented results also show that dynamic and partial hardware reconfiguration can be exploited to fully increase the flexibility of an FPGA-based system and not only to load complete functions dynamically. The possibility for reconfiguring the clock networks could also be exploited for dynamically activating / deactivating parts of the FPGA clock network and decrease power consumption.

The results from the real measurements show that the reconfiguration time is very important for the efficiency of this method. Using an internal configuration port is for this reason highly recommended.

Future work will be to integrate clock scaling based on dynamic and partial reconfiguration by using an internal configuration port as well as investigate how dynamic reconfiguration can be exploited for fine grained optimization of the FPGA clock networks.

7. Acknowledgements

The work presented in this paper is done within the AETHER project, which is sponsored by the European Commission under the 6th Framework program.

8. References

- [1] V. Degalahal and T. Tuan, "Methodology for High Level Estimation of FPGA Power Consumption ", in the Proc. of ASP-DAC 2005 Conference, Shanghai, Jan 2005.
- [2] J. Becker, M. Huebner, M. Ullmann: "Real-Time Dynamically Run-Time Reconfiguration for Power-/Cost-optimized Virtex FPGA Realizations", Integrated Circuits and Systems Design, 2003. SBCCI 2003. Proceedings. 16th Symposium on 8-11 Sept. 2003 Page(s):283 - 288
- [3] Xilinx; "Embedded Systems Tools Reference Manual", UG111 (v6.0) June 2006
- [4] Xilinx; "Spartan III FPGA Family", DS099 April 2006
- [5] IBM; "On-Chip Peripheral Bus, Architecture Specifications", 2001
- [6] Xilinx; "ISE 7 Software Manuals", 2005
- [7] Xilinx; "MicroBlaze Processor Reference Guide", UG086 (v6.0) June 2006
- [8] Xilinx; „Platform Studio User Guide“,UG113 (v4.0) February 2005
- [9] Altera; "Power Optimization", QII52016-6.0.0 May 2006
- [10] Xilinx; "Spartan 3 Generation Configuration User Guide", UG332 (v1.1), February 2007
- [11] M. Ullmann, M. Huebner, B. Grimm, J. Becker, "An FPGA Run-Time System for Dynamical On-Demand Reconfiguration", Proc. of the 11th Reconfigurable Architectures Workshop (RAW/IPDPS), April 2004.
- [12] K. Paulsson, M. Huebner, J. Becker, "On-Line Optimization of FPGA Power-Dissipation by Exploiting Run-time Adaption of Communication Primitives", SBCCI, Sept. 2006
- [13] L. Cheng, P. Wong, F. Li; Y. Lin; L. He; „Device and architecture co-optimization for FPGA power reduction“, Design Automation Conference, June 2005
- [14] A. Kumar, M. Anis; „An Analytical State Dependent Leakage Power Model for FPGAs“, Design, Automation and Test in Europe, March 2006
- [15] Xilinx; „Virtex II Platform FPGA User Guide“, UG002 (v2.0) March 2005
- [16] Xilinx; "Xilinx Parallel Cable IV Product Specification" DS0097 (v2.3.1), November 2006