RUN-TIME FPGA RECONFIGURATION FOR POWER-/COST-OPTIMIZED REAL-TIME SYSTEMS

Jürgen Becker, Michael Hübner, Michael Ullmann
Institut für Technik der Informationsverarbeitung (ITIV)
Universität Karlsruhe (TH), Germany
http://www.itiv.uni-karlsruhe.de/
{becker, huebner, ullmann}@itiv.uni-karlsruhe.de

Abstract: The paper describes a new approach of a flexible run-time system for handling dynamic function reconfiguration in fine-grain Virtex FPGAs, whereas the fulfillment of given real-time constraints are central. Moreover, the detailed evaluation and measurement of the power consumption situation during this dynamic reconfiguration process is essential for realistically quantifying the power loss of fine-grain FPGAs during dynamic reconfiguration processes. This kind of real-time run-time systems and power analysis give the designer and user the possibility to compare FPGA implementation alternatives and to apply the required functionality reconfigurations during the selected application scenarios. Thus, a qualified decision can be done between fine-grain FPGAs of different sizes and different dynamic reconfiguration frequencies, e.g. using smaller and more cost- as well as power-efficient FPGAs by temporarily outsourcing suitable functionalities.

Key words: Virtex FPGA, power consumption,real-time run-time reconfiguration, function and data management

1. INTRODUCTION

Field programmable gate-arrays (FPGAs) are mainly used today for rapid-prototyping purposes. They can be reconfigured many times for different applications. Modern state-of-the-art FPGA devices like Xilinx Virtex FPGAs [11] additionally support a partial dynamic run-time

Please use the following format when citing this chapter:
Becker, Jürgen, Hübner, Michael, Ullmann, Michael, 2006, in IFIP International Federation for Information Processing, Volume 200, VLSI-SOC: From Systems to Chips, eds. Glesner, M., Reis, R., Indrusiak, L., Mooney, V., Eveking, H., (Boston: Springer), pp. 119-132.

reconfiguration which reveals new aspects for the designer who wants to develop future applications demanding adaptive and flexible hardware. Especially in the domain of mobile computing high-end mobile communication applications will benefit from the capabilities of the new generation of reconfigurable devices.

Actually there exist some recent new approaches deploying Virtex/Virtex II FPGAs in multimedia applications using their capabilities for a dynamic function-multiplex showing new ways for the efficient deployment of partial run-time reconfiguration [2] [4].

A new approach to create systems which are able to manage configuration are run-time systems. These systems use the flexibility of an FPGA by changing the configuration partially. Only the necessary functions are configured in the chip's memory. By demand a function can be substituted by another while used parts stay operative. To solve the problem of substitution and I/O management the configuration needs a main module controlling the tasks.

With such a system it is possible to save resources like output pins and energy because of outsourcing configuration data. The need of chip area becomes smaller and therefore the power consumption can be reduced. Nevertheless the power requirements of such applications will grow with increasing rate of configuration.

Creating such a system has two aspects: Reducing amount of chip area and reducing power consumption by designing a control system which manages the content of FPGAs configuration in an intelligent way to minimize reconfiguration rate. Additionally this management can control the on chip intercommunication bus to prevent an overhead of bus size.

One important aspect is the power consumption during the FPGA's reconfiguration phase. To solve this problem it is necessary to analyze the behavior of FPGA while reconfiguration.

There exist many approaches for the analysis and estimation of the run-time power consumption of designs on FPGAs and they have been able to derive metrics enabling the designer to estimate the design's power consumption at design time [8] [9] [10].

The results show that the most power is dissipated by the on-chip long-wires connecting different functional blocks. Especially the functional multiplex operational mode as mentioned above will demand information on the expected power consumption. In that case not only the operational phase is of interest. Because the frequency of reconfiguration may be increased in such systems it might be of interest how much power is spent during the reconfiguration phase.

Current application scenarios use run-time reconfiguration at a low rate (minute range) so that the power dissipation during reconfiguration can be neglected but that condition can change when the number of applications sharing the same resource FPGA is growing so that the functions' cycle time is reduced to milli-seconds. In that case the energy and time needed for reconfiguration will be of interest for the designer as well.

2. DYNAMICALLY FPGA RECONFIGURATION

Basically a Virtex FPGA consists of two layers and additional configuration and control logic which handles the configuration bitstream loading and the distribution of the configuration data on dedicated positions of the second layer [12]. The first layer contains the reconfigurable hardware like logic-blocks (CLBs), RAM-blocks, I/O-blocks and configurable wiring resources. All blocks of the same type (I/O-blocks excepted) are aligned into columns (see figure 2-1).

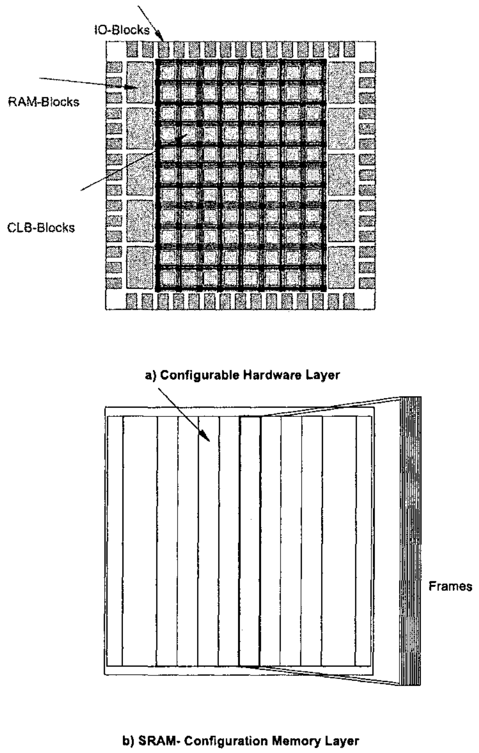
IO-Blocks
RAM-Blocks
CLB-Blocks
a) Configurable Hardware Layer
Frames
b) SRAM- Configuration Memory Layer

Figure 2-1 Layered FPGA structure (a,b)

The second layer which matches to the first layer is organized into columns as well. Each column whose width depends on the covered block-columns from the first layer consists of a set of one-bit sub-columns called frames.

A frame is the smallest piece of reconfiguration information that can be written on an FPGA and each frame contains fractions of the configuration information needed to configure the blocks assigned to the column. So if one block out of a column is to be reconfigured all other blocks in the same column have to be rewritten as well. This restriction demands that a partial reconfiguration of functions can only be done on groups of consecutive columns. It is possible to write the frames in a random manner out of order, but the configuration files are normally structured in an ordered way.

3. REAL-TIME RUN-TIME SYSTEM APPROACH

Figure 3-1 shows a possible schematic design of the physical FPGA-based part of a simple run-time system. The functions (for example A and B) are positioned on a fixed location. Several signals can be used for controlling and data exchange. The functions control their bus-driver with control signals. This is necessary to configure the drivers in the right way. The modules can be selected via an address bus. Next a comparison with the dedicated address is done and the enable signal becomes active.

By giving the module a default address at design time it is possible to assign a unique logic address to each module.

The design-time default module address is a value which is out of range of the normal address range during operation. So it is easy for the main control to recognize this new module and to assign a new valid logic address within the legal address range.

In operating state the bus arbiter calls every existing address. If a module does not want to send data, the busy line is active. This causes the arbiter to select the next address. All modules are called in a cyclic order without waiting if it's not necessary. When a module wants to send data, the specific request signal is active. Now the module is able to transfer its data via the data bus. The main control knows the functions data transfer time and allows to send data for a specific time interval. Another possibility is to send at first a time stamp with the information how long a data channel is needed. This opens a time window for the needed time. The advantage of this is that the modules can send information in different length without opening the data channel too long. In the other direction, if data from environment are dedicated for a module, the selected module gets the information to receive data via the Data-IN signal. As described in the text before, it is possible to send a time stamp at first to inform the module about the size of data. Should a function which is currently not configured on FPGA start working, a substitution of a not used module is initialized. In this case, the active state and data has to be saved to the local memory of the main control unit. If the function is needed later, the state and data can be re-transferred into this module and the function can start working with the same configuration as before. This context-save is done by using the data and state I/O signal lines.

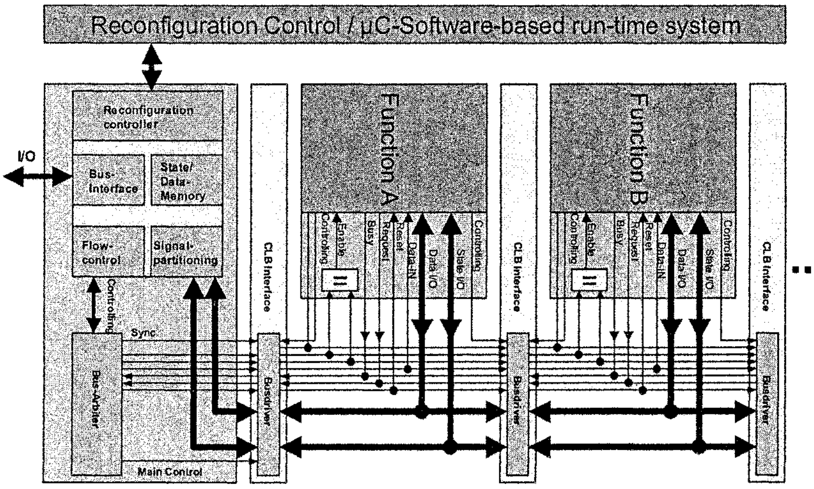

Figure 3-1 Simple run-time system with FPGA partial run-time reconfiguration support

Such a system is conceived to be implemented in future work to get an overview of the run-time and reconfiguration excess power.

We tested a simplified sample scenario using six different sample applications which implement electronic control units from the automotive domain (e.g. seat control, cabin compartment lighting where the results shown in table 1 have been calculated.

By outsourcing functions in this example, 12 I/O-pins were saved. Of course a logic is necessary to connect the shared external multiplexed signal lines in the right way. The saving of 467.6mW shows the capacity of this method in power saving. However, the power consumption of external memory storing the configuration data is not included in this scenario. But the amount of dissipated power of memory is smaller as the FPGAs.

The results were calculated by using real implemented functions on a Xilinx XCV2000E FPGA.

Table 1 Results of sample application

	All functions	Savings
Pins	108	12
Power consumption	1401.2mW	467.6mW

4. SCENARIO AND REAL TIME ANALYSIS

A real industrial scenario was used to analyze the advantage of using run-time systems for outsourcing configuration data.

The system is a motion control for DC-motors, which allows the movement in seven axis.

The overall system consists of seven explicit functions which needs an over-all amount of 1204 CLB blocks. The table in [12] shows, that the minimal FPGA which can be used for this application is a XCV300E with its 1536 CLB blocks. An estimation of dynamical power dissipation with Xilinx Power Estimator gives the value of 176mW.

Each of the seven functions needs 172 CLBs for implementation. By using a XCV200E, one function can be implemented in seven columns of the configuration memory. Additionally 4 CLB columns are necessary for the CLB interface. The specification for the system requires that three of seven functions are allowed to be used simultaneously. This means, that only three of seven functions have to be configured on FPGA. The amount of CLB columns for this case is 3x9 CLB columns for active functional blocks, 3x4 CLB columns for interface. 9 CLB columns for the run-time system (equals to 252 CLB blocks) are available for the control system. Each function causes a dynamic power dissipation of about 41mW on the XCV200E. With three active functions a difference of 53mW to the value for the XCV300E is left for the control system and the interfaces. It is realistic that the amount of power is less to the XCV300E implementation because the average toggle rate of the control system is not as high as the rate of the several functions and the static power dissipation is lower. However the power dissipation caused by reconfiguring has to be considered in this system. A value for this mode of operation is given in chapter 6.

Additionally the costs of XCV200E is less to the XCV300E and also a lower area on PCB is needed because of smaller packaging. An overall advantage by using a run-time system is shown in this small example.

Very important is to abide the real time demand. In the example the worst case situation is if all three functions have to be substituted by another. To reconfigure one functional block it is necessary to transfer seven plus four CLB columns into the configuration memory of the FPGA. The data size for these columns is about 40000 Bytes. Using the SelectMap port for reconfiguring, the amount of time for transferring the data to the FPGA is about 580µs on a transfer rate of 66Mbytes per second. To reconfigure all three functions, a time of 1.74ms is needed. The specification for these functions demands an answer time of <=100ms. 98.26ms are left for the control system managing the data transfer of context-save. With a frequency of 66MHz, 6485160 clock cycles are left for these operations.

The timing values have to be evaluated with the implemented system and can be shown exactly when the complete system is implemented on FPGA.

5. RECONFIGURATION BUS REALIZATION

In figure 3-1 the functions are separated with CLB interfaces. This is necessary to get a precise separation of routing recourses for each function. Bus interfaces are neccessary to seperate modules which can be substituted while dynamic and partial reconfiguration.

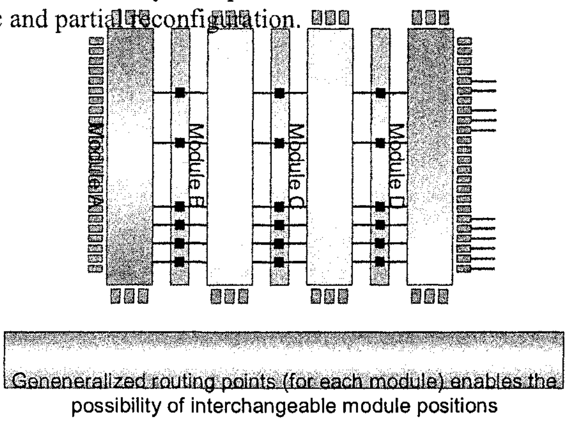
Module A
Module B
Module C
Module D
Genenerallized routing points (for each module) enables the possibility of interchangeable module positions

Figure 5-1 Generalized macros between modules

Figure 5-1 shows the schematic of a system with modules seperated by generalized bus macros. These macros enable the possibility to exchange e.g. module B and module C. Figure 5-2 shows a schematic view on the structure of such an interface. The FPGA's CLB (combined logic blocks) are used to connect the functional blocks. The dashed line is the border between the modules. Four horizontal aligned CLB blocks are used to build a 4 Bit interface.

The cause of using those called double CLB interfaces has its background on the routing in the design phase. To ensure that all connections of the modules are placed on the same position the CLB interfaces are positioned on a fixed place. Because of this, all modules can be implemented on every possible function column.

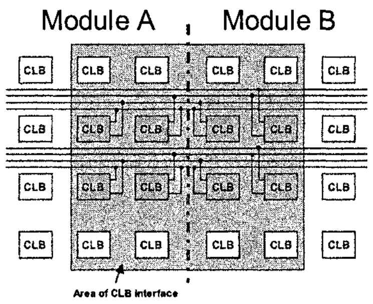

Figure 5-2 Structure of CLB interface

A more detailed schematic is shown in figure 5-3. To send a signal from left to right, for example the LE(0) input of the Tristate gate is on enable level, while RE(0) switches the second gate in Tristate mode. The signal of LI(0) now is connected to the Output(0) line which connects the two sides. In the other direction a communication is possible by changing the mode of the Tristate gates.

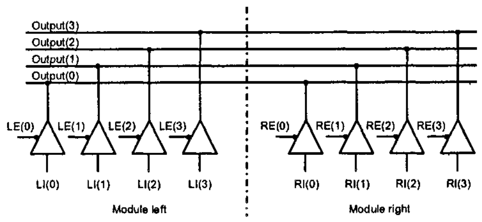

Figure 5-3 Tristate gates of CLB interface

Using Tristate gates for separating modules for dynamic and partial reconfiguration leads to problems while routing. Figure 5-3 shows that the names of the connection lines between the left and right module have the same name (e.g. Output(0)). While routing the autorouter might connect a signal line from the left module to the area of the right module. This causes an open signal line or even a short circuit if this modules ar substituted by another (dynamic and partial reconfiguration).

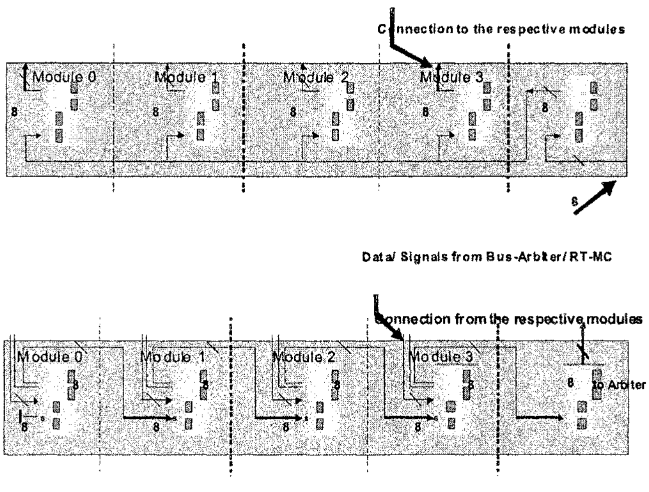
Connection to the respective modules
Module 0
Module 1
Module 2
Module 3
8
Data/ Signals from Bus-Arbiter/ RT-MC
Connection from the respective modules
to Arbiter

Figure 5-4 Schematic of Input/Output Macro

Figure 5-4 shows the schematic of macros using slices instead of TBUF elements. The slices of the input macro are programmed to route through the signals to the modules. Benefits are, that now the router has defined external pins to connect signals to the macros. This method allows a save and automatic generation of modules without time-consuming checking for signals crossing the modules border. More information about these macros can be found at [15].

6. POWER CONSUMPTION WHILE RECONFIGURATION

With a special measurement system, the amount of current for the FPGA while reconfiguration was measured. The used XCV2000E FPGA needs two power supplies. 1.8V for core and 3.3V supply for I/O elements. Figure 6-1 shows the measurement system. The measurement system consists of a PC with the control software, a Tektronix oscilloscope (Type TDS 220) connected to the PC's RS232 interface. The core and 3.3V supply current is measured with a shunt resistor connected to an opened jumper bridge on the Spyder-Virtex board.

By starting the measurement, the control software initializes the oscilloscope and pre-configures the FPGA. The next step is to start the measurement and transmit the second configuration to the board. The oscilloscope samples the voltage over the shunt resistor and stores the data into its memory.

Then the measure cycle is stopped and the data is read back from the oscilloscope memory into the PC and saved into a file. This file is the basis for further calculations with Matlab. This cycle is iterated n-times (n is given by the user). With the measurement system the core voltage and the 3.3V power supply were sampled to calculate the complete power dissipation.

The total power thus can be fragmented in P_{CORE} and P_{VCC}. The power was calculated in the following manner:

$$\overline{P_{CORE}} = \frac{\int_{T_{start}}^{T_{start}+T_{REC}} P_{CORE}(t)\cdot dt}{T_{REC}} \qquad (6.1)$$

$$\overline{P_{VCC}} = \frac{\int_{T_{start}}^{T_{start}+T_{REC}} P_{VCC}(t)\cdot dt}{T_{REC}} \qquad (6.2)$$

By using equation (6.1) and (6.2) a calculation of the median power of core and 3.3V supply between start and end of configuration is possible. T_{REC} is the total reconfiguration time.

Figure 6-2 and 6-3 show the measured power loss of core and 3.3V supply. To get a general result, these measurements were done with different kinds of implemented functions and repeated 10 times for each configuration.

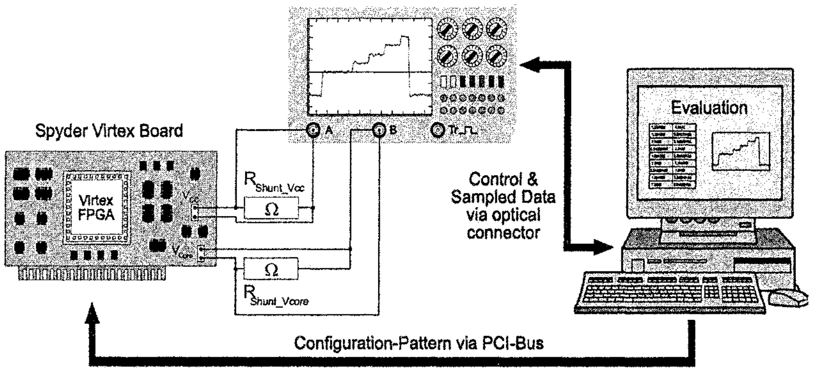

Figure 6-1 Power Measurement System

The rectangular shaped pulse of the 3.3V supply power caused by a partial reconfiguration Bitstream containing 30 CLB columns causes the additional power loss. The amount of core power dissipation stays on the same level. This level depends on the current configuration. We did some extended measurements on this topic. More detailed results were published in [1] showing the behavior of the FPGA while reconfiguration. Table 2 shows the result of the measurement. To get an overall view on the table the results of writing configuration data with a pre-reset deleting all preconfigured data and a value of power dissipation of the reseted FPGA are included. The last line of table 2 shows the values measured in dynamical partial reconfiguration mode. Table 2 and 3 show that the additional power loss may not be ignored. Systems with high reconfiguration rate, especially run-time systems have to be designed with the goal of saving power by managing the reconfiguration in an intelligent way.

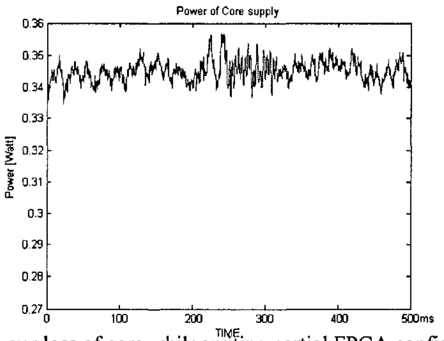

Figure 6-2 Power loss of core while writing partial FPGA configuration

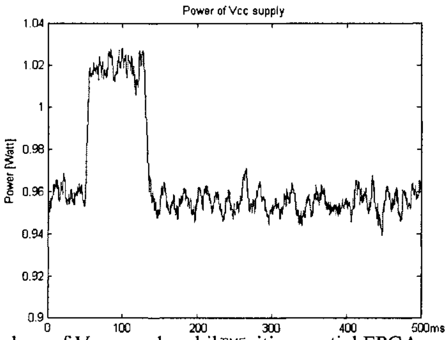

Figure 6-3 Power loss of Vcc supply while writing partial FPGA configuration

Table 2 Power dissipation while reconfiguring

	P_{core}	P_{vcc}	P_{tot}
Reseted empty FPGA (no configuration)	344.8mW	1024.64mW	1369,44mW
Writing configuration data after reset of configuration memory	394.5mW	1075.2mW	1496.7mW
Writing FPGA configuration data without previous reset of configuration memory	347.9mW (model dependent)	1017.2mW	1365.1mW (model dependent)

Table 3 Measured total power dissipation in normal mode

	Measured power dissipation
(1) Seat control unit	1165,1 mW
(2) Window lift unit	1138.4 mW

7. CONCLUSIONS

The paper described in detail a new approach of a flexible run-time system for handling dynamic function reconfiguration in fine-grain Virtex FPGAs. This included also the detailed description and implementation of the needed hardware infrastructure within today's commercial FPGA devices. The feasibility of the suggested approach and the fulfillment of given real-time constraints have been shown. The exemplaric real-time scenario examples provided are based on manual evaluations of the actual implementation of this run-time system. This contribution has evaluated and measured carefully the power consumption situation during the dynamic reconfiguration process. This first realistic quantification of the power consumption of fine-grain FPGAs gives the possibility of fair implementation comparisons, e.g. fine-grain FPGAs of different sizes by applying more or less dynamic reconfiguration steps, whereas cost and performance constraints also have to be taken into consideration. The use of such kind of real-time run-time and reconfiguration systems enables the designer to apply smaller and more cost- as well as power-efficient FPGAs, due to the possibility for temporarily outsourcing functions. Especially the reduction of power dissipation and pinout resources is of great advantage, as shown in the paper.

In addition, it is necessary to design intelligent control systems reducing reconfiguration rate by managing the constellation of modules for the needed scenarios. Thus, next steps will be to provide this kind of run-time-systems with the functionality of dynamically learning during application execution. If a function is more frequently needed as another one, the controller could decide to keep this function available within nearly located configuration memories by substituting a module which is not so frequently needed at the moment. The learned and evaluated scenario information, for example as table based data, has to be given to the main control as specification at design time. This will result in new techniques how real-time control systems manage the dynamic reconfiguration as well as data acquisition between periphery systems and internal modules, including the management of configuration data in internal and external memory topologies.

8. REFERENCES

[1] J. Becker, M. Hübner, M. Ullmann, "Power Estimation and Power Measurement of Xilinx Virtex FPGAs: Trade-offs and Limitations", SBCCI 2003, Brasil.

[2] Y. Ha, B. Mei, P. Schaumont, S.Vernalde, R. Lauwereins, H. De Man; "Development of a Design Framework for Platform- Independent Networked Reconfiguration of Software and Hardware"; Proc. 11th Int'l Conference on Field Programmable Logic and Applications, Belfast, Ireland, 2001.

[3] E.L. Horta, J.W. Lockwood, D.E. Taylor,D. Parlour, "Dynamic hardware plugins in an FPGA with partial run-time reconfiguration", Proceedings of 39th Design Automation Conference, 2002, Page(s): 343 -348

[4] J.-Y. Mignolet, S. Vernalde, D. Verkest, R. Lauwereins: "Enabling hardware-software multitasking on a reconfigurable computing platform for networked portable multimedia appliances"; Int'l. Conf. on Engineering of Reconfigurable Systems and Algorithms; June 25-27 2002, Las Vegas, USA

[5] F. Najm, "Transition density: a new measure of activity in digital circuits", IEEE Transactions on Computer-Aided Design, vol. 12, no. 2, pp. 310-323, February 1993

[6] J.C. Palma, A. Vieira de Melo, F. G. Moraes, N. Calazans, "Core Communication Interface for FPGAs", Proceedings of 15th Symposium on Integrated Circuits and Systems Design (SBCCI), 2002,Porto Alegre BRAZIL, Page(s): 183 –188

[7] http://www.arl.wustl.edu/arl/projects/fpx/parbit/

[8] K. Poon, A. Yan, S.J.E. Wilton, "A Flexible Power Model for FPGAs", 12th International Conference on Field-Programmable Logic and Applications, Sept 2002

[9] K. Poon, "Power Estimation for Field-Programmable Gate Arrays", Master of Applied Science Dissertation, University of British Columbia, 2002

[10] Li Shang, Alireza Kaviani and K. Bathala, "Dynamic Power Consumption in Virtex-II FPGA Family", International Symposium on Field-Programmable Gate Arrays (FPGA'2002), Monterey, CA, Feb. 2002, pp. 157-164.

[11] www.xilinx.com

[12] http://www.xilinx.com/xapp/xapp151.pdf

[13] http://www.xilinx.com/publications/products/software/xc_pdf/xc_xpower.pdf

[14] http://www.xilinx.com/support/techsup/powerest/virtex_power_estimator_v16.xls
[15] M. Huebner, T. Becker, J. Becker: "Real-Time LUT-Based Network Topologies for Dynamic and Partial FPGA Self-Reconfiguration", SBCCI04, Brasil