Impact of Task Distribution, Processor Configurations and Dynamic Clock Frequency Scaling on the Power Consumption of FPGA-based Multiprocessors

Diana Goehringer, Jonathan Obie
Fraunhofer IOSB
Ettlingen, Germany
{diana.goehringer, jonathan.obie}@iosb.fraunhofer.de

Michael Huebner, Juergen Becker
ITIV, Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
{michael.huebner, becker}@kit.edu

Abstract— **As only the currently required functionality on a dynamic reconfigurable FPGA-based system is active, a good performance per power ratio can be achieved. To find such a good performance per power ratio for a given application is a difficult task, as it requires not only knowledge of the behavior of the application, but also knowledge of the underlying hardware architecture and its influences on the performance and the static and dynamic power consumption. Is it for example better to use two processors running at half the clock frequency than a single processor? The main contributions of this paper are: the description of a tool flow to measure the power consumption for multiprocessor systems in Xilinx FPGAs, a novel runtime adaptive architecture for analyzing the performance per power tradeoff and for dynamic clock frequency scaling based-on the inter-processor communication. Furthermore, we use three different application scenarios to show the influence of the clock frequency, different processor configurations and different application partitions onto the static and dynamic power consumption as well as onto the overall system performance.**

Keywords- Power Consumption, Multiprocessor System-on-Chip (MPSoC), Dynamic Frequency Scaling, Task Distribution, Application Partitioning, Dynamic and Partial Reconfiguration, FPGA.

I. INTRODUCTION

Parameterizable function blocks used in FPGA-based system development, open a huge design space, which can only hardly be managed by the user. Examples for this are arithmetic blocks like divider, adder, soft IP-multiplier, which are adjustable in terms of bitwidth and parallelism. Additional to arithmetic blocks, also soft-IP processor cores provide a variety of parameters, which can be adapted to the requirements of the application to be realized with the system. Especially, Xilinx offers with the MicroBlaze Soft-IP RISC processor [1] a variety of options for characterizing the core individually. These options are amongst others the use of cache memory and its size, the use of an arithmetic unit, a memory management unit and the number of pipeline stages. Furthermore, the tools offer to deploy up to two processor cores as multiprocessor on one FPGA. Every option now can be adjusted to find an optimal parameterization of the processor core in relation to the target application. For example, a specific cache size can speed up the application tremendously, but also the optimal partition of functions onto the two cores has a strong impact on the speed and power consumption of the system. The examples show the huge design space, if only one parameter is used. It is obvious, that the usage of multiple parameters for system adjustment leads to a multidimensional optimization problem, which is not or at least very hardly manageable by the designer. In order to gain experience regarding the impact of processor parameterization in relation to specific application scenario, it is beneficial to evaluate e.g. the performance and power-consumption of an FPGA-based system and normalize the results to a standard design with a default set of parameter. The result of such an investigation is a first step for developing standard guidelines for designers and an approach for an abstraction of the design space in FPGA-based system design. This paper presents first results of a parameterizable multiprocessor system on a Xilinx Virtex-4 FPGA, where the parameterization of the processor is evaluated in terms of power consumption and performance. Moreover, the varying partition of the different application scenarios is evaluated in terms of power consumption for a fixed performance. For this purpose, a tool flow for analyzing the power consumption through generating the value change dump (VCD) file from the post place and route simulation will be introduced. The presented flow enables to generate the most accurate power consumption estimation from this level of abstraction. A further output of the presented work is an overview of the impact of parameterization to the power consumption. The results can be used as a basic guideline for designers, who want to optimize their system performance and power consumption.

The paper is organized in the following manner: In Section II related work is presented. Section III describes the power estimation tool flow used in this approach. The novel system architecture used for analyzing the performance and the power consumption of the different applications is presented in Section IV. The application scenarios are described in Section V. In Section VI the application integration and the results for performance and power consumption are given. Finally, the

paper is closed by presenting the conclusions and future work in Section VII.

II. Related Work

Optimization of the dynamic and static power consumption is very important, especially for embedded systems, because they often use batteries as a power source.

Therefore, many researchers like for example Meintanis et al [2] explored the power consumption of Xilinx Virtex-II Pro, Xilinx Spartan-3 and Altera Cyclone-II FPGAs. They estimated the power consumption at design-time using the commercial tools provided by Xilinx and Altera. They further explored the differences between the measured and estimated power consumption for these FPGAs. Becker et al. [3] explored the difference between measured and estimated power consumption for the Xilinx Virtex-2000E FPGA. Furthermore, they explored the behavior of the power consumption, when using dynamic reconfiguration to exchange the FPGA-system at runtime.

Other works focus on the development of own tools and models for efficient power estimation at design-time for FPGA-based systems. Poon et al. [4] present a power model to estimate the dynamic, short circuit and leakage power of island-style FPGA architectures. This power model has been integrated into the VPR CAD flow. It uses the transition density signal model [5] to determine signal activities within the FPGA. Weiss et al. [6] present an approach for design-time power estimation for the Xilinx Virtex FPGA. This estimation method works well for control flow oriented applications but not so well for combinatorial logic. Degalahal et al. [7] present a methodology to estimate dynamic power consumption for FPGA-based system. They applied this methodology to explore the power consumption of the Xilinx Spartan-3 device and to compare the estimated results with the measured power consumption.

All these approaches focus either on the proposal of a new estimation model or tool for estimating the power consumption at design-time or they compare their own or commercial estimation models and tools with the real measured power consumption. The focus of the investigations presented in this paper is to show the impact of parameterization of IP-cores, here specifically the MicroBlaze soft processor, which differs from the approaches mentioned above where the topic is more on tool development for power estimation.

The novelty of our approach is to focus on the requirements of the target application and to propose a design guideline for system developers of processor-based FPGA systems. This means, providing guidance in how to design a system to achieve a good tradeoff between performance and power consumption for a target application. To develop such a guideline the impact of the frequency, different processor configurations and the task distribution in a processor-based design is investigated in this paper for different application scenarios. To the best of our knowledge, similar work has not done before.

III. Tool Flow for Power Measurement

Xilinx provides two kinds of tools for power consumption estimation: Xilinx Power Estimator (XPE) [8] and Xilinx Power Analyzer (XPower) [9].

The XPE tool is based on an excel spreadsheet. It receives information about the number and types of used resources via the report generated by the mapping process (MAP) of the Xilinx tool flow. Alternatively, the user can manually set the values for the number and type of used resources. The frequencies used within the design have to be manually set by the user. The advantage of this method is that results are obtained very fast. The disadvantage is that the results are not very accurate, especially for the dynamic power consumption. This is, because the different toggling rates of the signals are not taking into account. Also, the results are not as accurate, because they are based on the MAP report, and not on the post place and route (PAR) report, which resembles the system used for generating the bitstream.

The XPower tool estimates the dynamic and static power consumption for submodules, different subcategories and the whole system based on the results of a post place and route (PAR) simulation. This makes the estimation results much more accurate compared to the XPE tool, because the final placed and routed system is considered for the power estimation. But even more important, due to the simulation of the PAR system with real input data, the toggling rates of the signals can be extracted and used within the power estimation. For estimating the power consumption with the XPower tool the following input files are required:

- Native Circuit Description (NCD) file, which specifies the design resources
- Physical Constraint File (PCF), which specifies the design constraints
- Value Change Dump (VCD) file, which specifies the simulated activity rates of the signals

The NCD and the PCF file are obtained after the PAR phase of the Xilinx implementation tool flow. The VCD file is generated by doing a simulation of the PAR design with the ModelSim simulator.

Due to the higher accuracy the XPower tool was used here. As we wanted to estimate the power consumption for systems with one or two MicroBlaze processors, the hardware and the software executables of the different system were designed within the Xilinx Platform Studio (XPS)[10]. Figure 1. shows the flow diagram for doing power estimation with XPower for an XPS system.

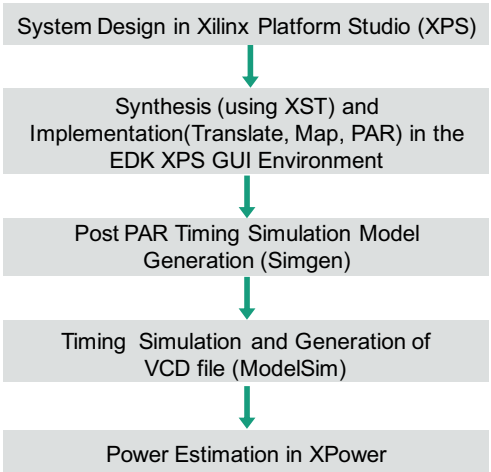

Figure 1. Flow Diagram of the EDK XPower Flow

After the system has been designed and implemented within the XPS environment, the Simgen [10] tool is used to generate the post PAR timing simulation model of the system. This simulation model is the used to simulate the behavior of the system with the ModelSim simulator and to generate the VCD file. In the last step XPower is used to read in the VCD, the NCD and the PCF files of the design and to estimate the dynamic and static power consumption.Care has to be taken, because in a normal Xilinx implementation flow the software executables are integrated into the memories of the processors after the bitstream has been generated. When using XPower and the post PAR simulation, the memories of the processor have to be initialized in an earlier step. This means, into the post PAR simulation model, otherwise the simulated system behavior and the VCD file would not be accurate.

IV. Novel System Architecture

The system structure of the dual-processor system is shown in Figure 2. Three new components have been designed and implemented: the Virtual-IO, the Bridge and the Reconfigurable Clock Unit. All three components have been integrated into a library for the XPS tool. Therefore, they can be inserted and parameterized using the graphical user interface (GUI) of the XPS tool, which makes them easy reusable within other XPS designs.

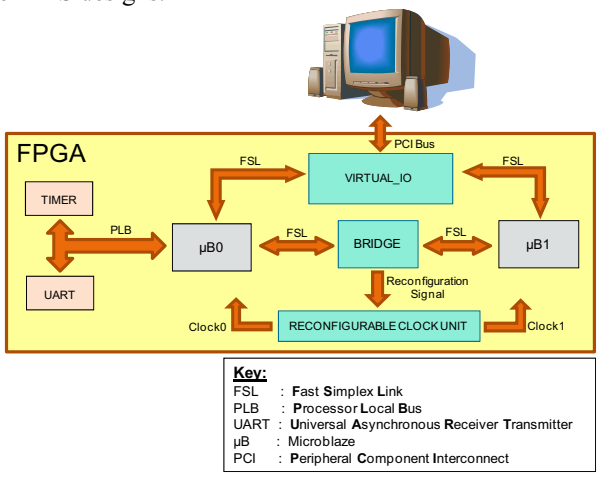

Figure 2. Dual processor design with three new components: Virtual-I/O, Bridge and Reconfigurable Clock Unit

The Virtual-IO receives data from the host PC and sends results back to the host PC via the PCI-bus. The Virtual-IO communicates via the Fast Simplex Links (FSLs) [11] with two MicroBlaze processors (µB0 and µB1). µB0 communicates with the user via the UART interface. It also has a timer, which is used to measure the performance of the overall system. The two processors communicate with each other via FSLs over the Bridge component. Depending on the fill level of the FIFOs within the Bridge reconfiguration signals are send to the Reconfigurable Clock Unit. The Reconfigurable Clock Unit reconfigures the clocks of the two processors based on the reconfiguration signals issued by the Bridge. For the uni-processor system, which is used for comparison, the Bridge, the Reconfigurable Clock Unit, µB1 and their connections were removed as shown in Figure 3.

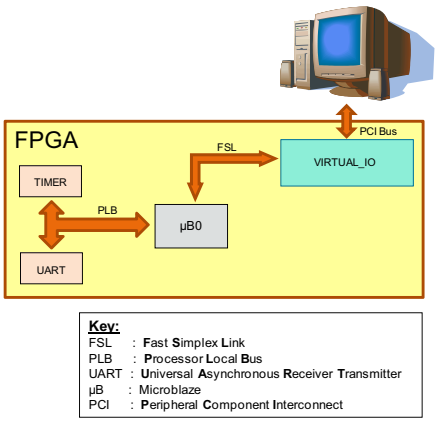

Figure 3. Uni-processor system

The following subsections explain the new components and their features more in detail.

A. *Virtual-IO*

The Virtual-IO component is used to communicate with the host PC via the PCI-bus. It provides an input and an output port to the PCI-bus and one input and one output port for each MicroBlaze processor. It consists of two FIFOs, one for the incoming and one for the outgoing data of the PCI-bus. Each FIFO is controlled via a Finite State Machine (FSM), as it is shown in Figure 4.

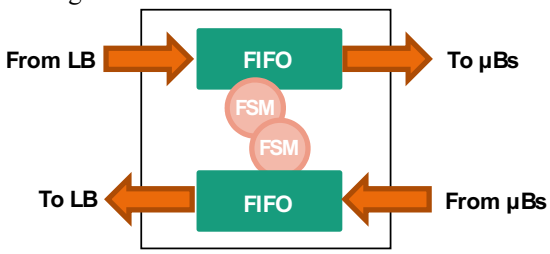

Figure 4. Virtual-IO component

The Virtual-IO is a wrapper around 6 different modules. The first module is Virtual-IO 1, which sends data first to µB0 and then to µB1. It then receives the calculated results in the same order. The second module is Virtual-IO 2, which sends data only to µB0. Results are only received over µB1. Therefore, µB0 sends its results to µB1, which then sends the results of µB0 together with its own results back to the Virtual-IO 2. The third module is Virtual-IO 3, which sends first data to µB0. Afterwards, it sends in parallel to both processors µB0 and µB1 the same data. Finally, it sends some data only to µB1. After the execution of the processors, first µB0 and then µB1 send their results back to the Virtual-IO 3. The fourth module is Virtual-IO 4, which is only connected to one of the processors, e.g. µB0. Due to this, this module is used in all uni-processor designs. For a dual-processor design it sends data to µB0, which then forwards parts of the data to µB1. After execution µB1 sends its results back to µB0, which forwards the results of the execution of the two processors to the Virtual-IO 4. The fifth module is Virtual-IO 5, which sends the same data to both processors in parallel, but receives the results only via µB0. The sixth module is Virtual-IO 6. It is very similar to

Virtual-IO 5. The only difference is that it receives the calculation results from µB1 instead of µB0.

The modules can be selected in the XPS GUI via the parameters of the Virtual-IO component. Other parameters that can be set by the user are: the number of input and output words for each processor separately, the number of common input words and the size of the image (only for image processing applications).

B. *Bridge*

The Bridge module is used for the inter-processor communication. It consists of two asynchronous FIFOs controlled by FSMs, to support a communication via the two different clock domains of the processors, as shown in Figure 5. This Bridge component controls the fill level of the two FIFOs. If one FIFO is to nearly full, it is assumed that the processor, which reads from this FIFO, is too slow. As a result, a reconfiguration signal to increase the clock rate of this processor is send to the Reconfiguration Clock Unit.

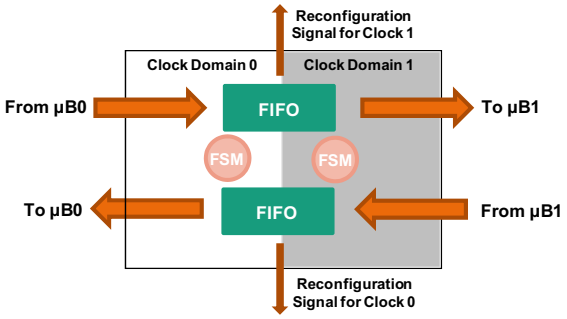

Figure 5. Internal structure of the Bridge

C. *Reconfigurable Clock Unit*

The internal structure of the Reconfigurable Clock Unit is shown in Figure 6. It consists of two Digital Clock Managers (DCMs) [12], two Clock Buffer Multiplexer primitives (BUFGMUXes) [13] and the Logic component, which controls the reconfiguration of the DCMs.

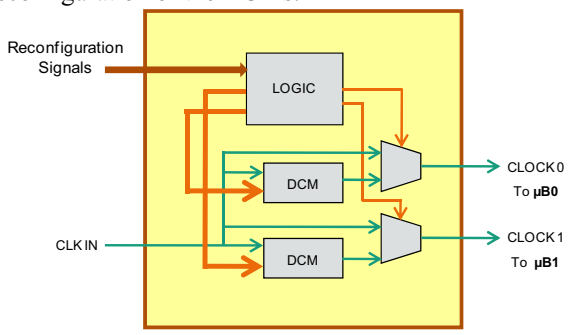

Figure 6. Internal structure of the Reconfigurable Clock Unit

The Logic component receives the reconfiguration signals of the Bridge component. It then starts the reconfiguration of the DCM primitive for the slower processor. For the reconfiguration the specific ports provided by Xilinx for dynamic reconfiguration of the Virtx-4 DCM primitive are used. During the reconfiguration process the DCM has to be kept in a reset state for a minimum of 200 ms. During this time interval the outputs of this DCM are not stable and cannot be used. Instead of stalling the corresponding processor, the BUFGMUX primitive is used to provide CLK_IN, the original input clock of the two DCM, to the processor, whose DCM is under reconfiguration. The BUFGMUX is a special clock multiplexer primitive, which assures, that no glitches occur, when switching to a different clock. After the configuration of the DCM is finished, the BUFGMUX is used to switch back to the DCM clock. An alternative would be to stall the processor, while its clock is being reconfigured. Because 200 ms are quite a long time, especially for image processing applications, where each 40 ms a new input frame is received from a camera; this would result in a loss of input data.

To prevent an oscillation, the controller logic will stop increasing the clock frequency, if 125 MHz for this MicroBlaze have been reached, which is the maximum frequency supported by the MicroBlaze and its peripherals, or if its clock frequency has been increased for three consecutive times. If the reconfiguration signal is still asserted meaning the processor is still too slow, then the DCM of the faster processor is reconfigured to provide a slower clock to the faster processor.

Alternatively, instead of dynamically reconfiguring the DCM, different output ports of a DCM could be used to generate different clocks. Using several BUFGMUXes the different clocks could be selected. The advantage is a faster switch between different clocks and the drawback is that not as many different clocks are possible as when dynamic reconfiguration is used. This will be investigated in future work.

V. Application Scenarios

Three different applications scenarios were used to explore the impact of the processor configurations, the task distribution and the dynamic clock frequency scaling on the power consumption of FPGA-based processor systems. The three different algorithms are described in detail in the next subsections. The first algorithm is the well known sorting algorithm called Quicksort [14]. It consists of a lot of branches and comparisons. The second algorithm is an image processing algorithm called Normalized Squared Correlation (NCC), which consists of many arithmetic operations, e.g. multiply and divide. The third algorithm is a variation of a bioinformatic algorithm called DIALIGN [15], which consists of many comparisons and additions and subtractions. These algorithms with their different algorithm requirements, e.g. branches, comparators, multiply & divide, add & subtract, were used to provide a user guideline of designing a system with a good performance per power tradeoff for a specific application. By comparing the algorithm requirements of new applications with the three example algorithms, the system configurations of the most similar example algorithm is chosen as a starting system. Such a guideline to limit the design space is very important to save time and achieve a higher time-to-market, because the simulation and the power estimation with XPower are very time-consuming. Also, the bitstream generation to measure the performance of the application on the target hardware architecture is time-consuming. These long design times can be shorten by starting with an appropriate design, e.g. the right processor configurations, a good task distribution and a well selected execution frequency.

A. *Sorting Algorithm: Quicksort*

Quicksort [14] is a well known sorting algorithm with a divide and conquer strategy. It sorts a list by recursively partitioning the list around a pivot and sorting the resulting sublists. It has an average complexity of Θ (n log n).

B. *Image Processing Algorithm: Normalized Squared Correlation*

2D Squared Normalized Correlation (NCC) is often used to identify an object within an image. The evaluated expression is shown in equation (1).

$$C(p) = \frac{\left(\sum_{i=0}^{n}\sum_{j=0}^{m}\left(\left(A_p(i,j) - \overline{A_p}\right) * \left(T(i,j) - \overline{T}\right)\right)\right)^2}{\left(\sum_{i=0}^{n}\sum_{j=0}^{m}\left(A_p(i,j) - \overline{A_p}\right)^2\right) * \left(\sum_{i=0}^{n}\sum_{j=0}^{m}\left(T(i,j) - \overline{T}\right)^2\right)}$$

T : Template image with n Rows and m Columns

A_p : Sub window of the search region wit h n Rows and m Columns

$\overline{T}$: Mean of T

$\overline{A_p}$: Mean of A_p (1)

This algorithm uses a template T of the object to be searched for and moves this template over the search region A of the image. A_p, the subwindow of the search region at point p with the same size as T, is then correlated with T. The result of this expression is stored at point p in the result image C. The more similar A_p and T are the higher is the result of the correlation. If they are equal, the result is 1. The object is then detected at the location with the highest value.

C. *Bioinformatic Algorithm: DIALIGN*

DIALIGN [15] is a bioinformatics algorithm, which is used for comparison of the alignment of two genomic sequences. It produces the alignment with the highest number of similar elements and therefore the highest score as shown in Figure 7.

Figure 7. Alignment of two sequences a and b with DIALIGN.

VI. INTEGRATION AND RESULTS

For the power consumption estimation and the performance measurement a Xilinx Virtex-4 FX 100 FPGA was used. The performance was measured on the corresponding FPGA Board from Alpha Data [16]. As measuring the exact power consumption of the FPGA on this board is not possible, it was estimated at design-time using the XPower tool flow as described in Section III. The impact of the clock frequency, the configuration of the processor and the task distribution onto the power consumption and the performance of the system has been explored and the results are presented in the following subsections. For each exploration some parameters had to be kept fixed to assure a fair comparison. For the exploration of the impact of the clock frequency, the algorithm and the processor configuration have been kept fixed. For the exploration of the impact of the configuration of the processor the clock frequency were kept fixed. Finally, for the exploration of the task distribution, the processor configuration and the performance were kept fixed to lower the overall system power consumption while maintaining the performance similar to the performance achieved with a reference uni-processor design running at 100 MHz, which is a standard frequency for Virtex-4 based MicroBlaze systems.

A. *Impact of the clock frequency*

First of all the impact of the variation of the clock frequency onto the power consumption was explored for a uni-processor system, which executes the NCC algorithm on one MicroBlaze. The MicroBlaze was configured to use a 5-stage pipeline and no arithmetic unit. The results for the dynamic and quiescent power consumption for the core and the other components as well as the total power consumption of the system are given in TABLE I. The quiescent power consumption is also called static power consumption in the following, because it represents the power consumption of the user configured FPGA without any switching activity.

The impact of the clock frequency onto the static - and the dynamic power consumption is presented in Figure 8. and Figure 9. respectively. As can be seen the static power consumption increases by around 0,24 mW / MHz, while the dynamic power consumption increases by around 3,26 mW / MHz.

Out of this results the impact onto the total power consumption, which is around 3,5 mW / MHz. The impact on the total power consumption as well as on the performance is shown in Figure 10.

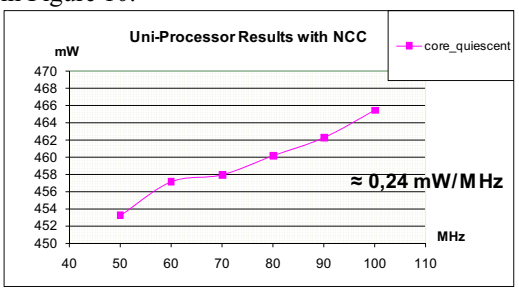

Figure 8. Impact of the clock frequency onto the static power consumption of a uni-processor design.

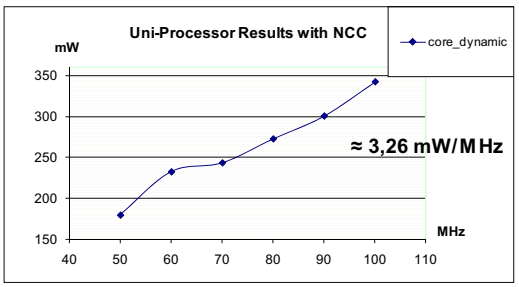

Figure 9. Impact of the clock frequency onto the dynamic power consumption of a uni-processor design.

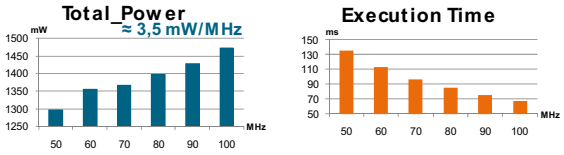

Figure 10. Impact of the clock frequency onto the total power consumption and onto the execution time of a uni-processor design executing the NCC algorithm.

TABLE I. IMPACT OF THE VARIATION OF THE CLOCK FREQUENCY ONTO THE POWER CONSUMPTION

Clk Freq. (MHz)	$P_{CoreDynamic}$(mW)	$P_{OthersDynamic}$(mW)	$P_{CoreQuiescent}$(mW)	$P_{OthersQuiescent}$(mW)	P_{Total}(mW)	P_{Total}(%)
50	180	26	453	641	1298	- 11,9
60	232	26	457	641	1355	- 8,0
70	243	26	458	641	1367	- 7,2
80	273	26	460	641	1398	- 5,1
90	301	26	462	641	1428	- 3,1
100	343	26	465	641	1473	NA

B. *Impact of the processor configurations*

For exploration, a uni-processor design consisting of a single MicroBlaze running at 100 MHz was used. The results were compared against a reference configuration, which was a MicroBlaze with a 5-stage pipeline and no arithmetic unit (integer divider and barrel shifter). The following configurations were explored:

i. adding an arithmetic unit (AU)
ii. reduction of the pipeline to 3-stages (RP)
iii. combination of i and iii (AU+RP)

The impact onto the power consumption and the performance was explored for all three algorithms. The impact is very different for the different applications, due to the different algorithm requirements, as mentioned in Section V and its subsections.

Figure 11. and TABLE II. show the impact of the different configurations for the Quicksort algorithm. Due to the multiple branches in the algorithm a reduction of the pipeline stages is very beneficial in terms of execution time and power consumption. The impact of the addition of the arithmetic unit only provides a minimal improvement in terms of performance, but with a stronger degradation of the power consumption. Depending on the performance and power consumption constraints, either the system with the AU + RP or the RP system would be chosen.

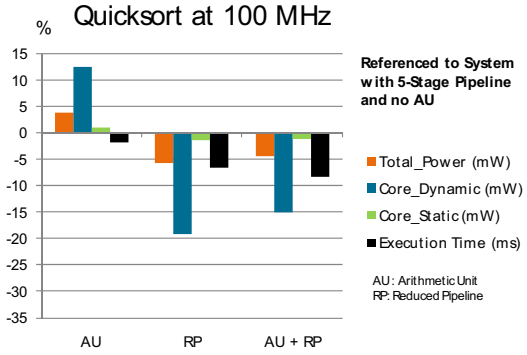

Figure 11. Impact of the MicroBlaze configurations for the Quicksort algorithm.

Figure 12. and TABLE III. show the impact of the different configurations for the NCC algorithm. As this algorithm requires many arithmetic operations, the addition of an AU improves the overall execution time, while the reduction of the pipeline stages results in a strong degradation (over 50%). This degradation is due to the reason that the execution of arithmetic operations take more clock cycles, if the pipeline is reduced. Therefore, for this and similar algorithms a system with an AU and a 5-stage pipeline would be optimal from a performance perspective. If the power consumption needs to be reduced and some performance degradation is acceptable, than the reference system or the AU+RP system would be a good choice.

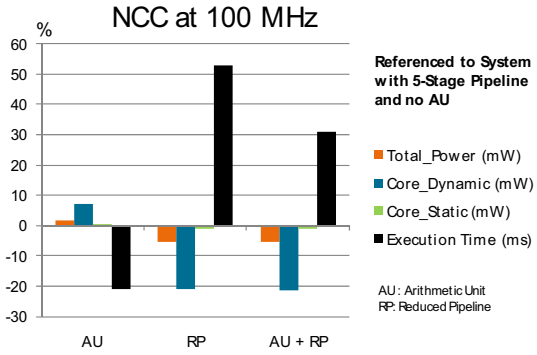

Figure 12. Impact of the MicroBlaze configurations for the NCC algorithm.

In Figure 13. and TABLE IV. the impact onto the performance and power consumption of the three different processor configurations compared to the reference system are presented for the DIALIGN algorithm. Adding an AU improves the execution time only a little bit, while increasing the overall power consumption compared to the reference design. The reduction of the pipeline to 3-stages improves the total power consumption by 6,8%, but worsening the execution time by 25%. The combination of AU+RP shows nearly the same impact as the RP system. Therefore, the reference system is the best choice, if performance is the most important factor. If on the other hand the power consumption is more important, than the RP system would be a good choice for these kinds of algorithms.

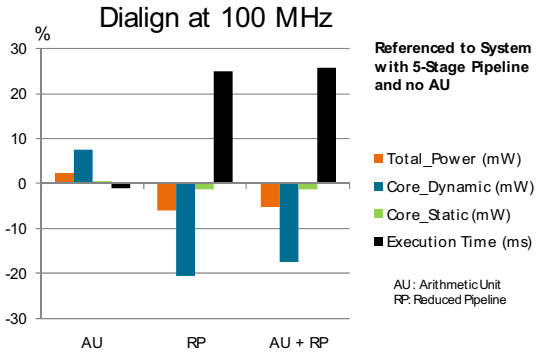

Figure 13. Impact of the MicroBlaze configurations for the DIALIGN algorithm

TABLE II. IMPACT OF THE MICROBLAZE CONFIGURATIONS FOR THE QUICKSORT ALGORITHM AT 100 MHZ

µB Parameter	$P_{CoreDynamic}$(mW)	$P_{OthersDynamic}$(mW)	$P_{CoreStatic}$(mW)	$P_{OthersStatic}$(mW)	P_{Total}(mW)	P_{Total}(%)	Time (ms)
Default	438,33	26,13	472,91	639,56	1576,93	NA	18,42
Arithmetic Unit	493,26	26,14	477,20	639,58	1636,18	+ 3,76	18,10
3-stage Pipeline	354,23	26,13	466,41	639,56	1486,33	- 5,75	17,21
Both	372,84	26,13	467,84	639,58	1506,39	- 4,47	16,89

TABLE III. IMPACT OF THE MICROBLAZE CONFIGURATIONS FOR THE NCC ALGORITHM AT 100 MHZ

µB Parameter	$P_{CoreDynamic}$(mW)	$P_{OthersDynamic}$(mW)	$P_{CoreStatic}$(mW)	$P_{OthersStatic}$(mW)	P_{Total}(mW)	P_{Total}(%)	Time (ms)
Default	341,68	26,09	465,44	639,57	1472,78	NA	67,74
Arithmetic Unit	366,28	26,13	467,33	639,57	1499,31	+ 1,80	53,62
3-stage Pipeline	269,63	26,10	459,97	639,57	1395,27	- 5,26	103,64
Both	269,40	26,12	459,95	639,58	1395,05	- 5,28	88,84

TABLE IV. IMPACT OF THE MICROBLAZE CONFIGURATIONS FOR THE DIALIGN ALGORITHM AT 100 MHZ

µB Parameter	$P_{CoreDynamic}$(mW)	$P_{OthersDynamic}$(mW)	$P_{CoreStatic}$(mW)	$P_{OthersStatic}$(mW)	P_{Total}(mW)	P_{Total}(%)	Time (µs)
Default	431,67	26,06	472,38	639,58	1569,69	NA	786,48
Arithmetic Unit	464,39	26,06	474,93	639,59	1604,97	+ 2,25	777,64
3-stage Pipeline	343,01	26,05	465,54	639,59	1474,19	- 6,08	982,85
Both	355,88	26,06	466,53	639,58	1488,05	- 5,20	988,05

TABLE V. QUICKSORT POWER CONSUMPTION

	Uni-Processor (100MHz)	Dual_2 (80/50 MHz)	Dual_5 (95 MHz)
Execution Time - ms	18,42	18,80	19,27
Core (dyn/stat)_Power - mW	438,33 / 472,91	295,89 / 461,95	384,34 / 468,72
Total Power - mW	1576,93	1475,56	1570,79
Total Power - %	NA	- 6,43	- 0,39

TABLE VI. NCC POWER CONSUMPTION

	Uni-Processor (100MHz)	Dual_3 (54 MHz)	Dual_2 (87,5/50 MHz)
Execution Time - ms	67,74	67,28	67,62
Core (dyn/stat)_Power - mW	341,68 / 465,44	297,39 / 462,07	322,32 / 463,96
Total Power - mW	1472,78	1477,20	1504,02
Total Power - %	NA	+ 0,30	+ 2,12

TABLE VII. DIALIGN POWER CONSUMPTION

	Uni-Processor (100MHz)	Dual_5 (50 MHz)	Dual_6 (50 MHz)
Execution Time - ms	30,21	30,16	30,16
Core (dyn/stat)_Power - mW	431,67 / 472,38	440,80 / 473,09	352,45 / 466,27
Total Power - mW	1569,69	1631,62	1536,44
Total Power - %	NA	+ 3,95	- 2,12

C. Impact of the task distribution and the frequency scaling

To measure the impact onto the power consumption the algorithms were partitioned onto two MicroBlaze processors. The frequency for the two processors was chosen in such a way, that the execution time of the dual-processor design was as similar as possible to the reference system consisting of a single MicroBlaze running at 100 MHz. For all systems the configurations of the processors were fixed to a 5-stage pipeline and no arithmetic unit.

TABLE V. shows the results for distributing the Quicksort algorithm on two processors instead of one. Two partitions were done. The first one is called Dual_2 (80/50 MHz), which means, that the Virtual-IO 2 was used and µB0 was running at 80 MHz while µB1 was running at 50 MHz. The algorithm was so partitioned that µB0 receives the whole data to be sorted. It then divides the data into two parts and sends the second part to µB1. Both then sort their partition. µB0 forwards its sorted part of the list to µB1, which sends the final combined sorted list via the Virtual-IO 2 to the host PC. With this partition the overall power consumption could be reduced by 6,43% compared to the single processor reference system.

The second partition called Dual_5 (95 MHz) uses the Virtual-IO 5 to send incoming data to both processors running at 95 MHz. µB0 searches the list for elements smaller and µB1 searches the list for elements bigger than the pivot. When one has found an element the position of this element is send to the other processor. Both processor then update their lists by swapping the own found element with the one the other processor has found. At the end both processors have as a

result a searched list. µB0 then sends its resulting list back to the host PC via the Virtual-IO 5. The power consumption of this version is nearly the same as the reference system, while the total execution time increases.

TABLE VI. shows the result for the partitioning of the NCC algorithm onto two processors. The first partitioning uses the Virtual-IO 3 to partition the incoming image into two overlapping tiles, one for each processor. The overlapping part is send to both processors simultaneously. As the NCC is a window-based image processing algorithm, the boarder pixels between the two tiles are needed by both processors. Each of the processors runs at 54 MHz, which results in a similar execution time, and also in a similar total power consumption as the reference design.

The second partition called Dual_2 (87,5 /50 MHz) uses Virtual-IO 2 to send the whole image to µB0. µB0 runs at 87, 5 MHz and calculates the complete numerator and the denominator. Then it forwards both to µB1, which does the division and sends the results back to the Virtual-IO 2. µB1 runs at 50 MHz. While the execution time is nearly the same, the overall power consumption is increased slightly by 2,12%.

TABLE VII. shows the result for executing the DIALIGN Algorithm with two processors. Two partitions were done. The first one is called Dual_5 (50 MHz) and uses Virtual-IO 5 to send the incoming sequences to both processors running at 50 MHz. Each processor calculates half of the resulting score matrix. µB0 calculates on a row-based fashion all values above the main diagonal. µB1 calculates on a column-based fashion all values below the main diagonal. The scores on the main diagonal are calculated by both processors. After µB0 has finished calculating one row and µB1 one column respectively, they exchange the first score nearest to the main diagonal, as this score is needed by both processors for calculating the next row/column respectively. While the execution time is nearly the same, the overall power consumption is increased by 3,95%.

The second partition is called Dual_6 (50 MHz). It uses the Virtual-IO 6 to send the sequences to the processors, which run both at 50 MHz. Here a systolic array approach is used for executing the DIALIGN algorithm. µB1 then sends the final alignment and the score back to the host PC. With this partition the overall power consumption could be reduced by 2,12% compared to the single processor reference system.

VII. Conclusions and Outlook

This paper reports the research and evaluation of different microprocessor parameterization, application and data partitioning on a dual-processor system. The results of the experiments show the impact of the different parameterization on the power consumption and performance in relation to a set of selected applications. Depending on the application type it can be seen that different parameter configurations, e.g. configuration of the processors and their frequencies, but also a good application partitioning, are essential for achieving an efficient tradeoff between performance and power constraints. The results can be used to guide developers what parameter set suits to a certain application scenario. The vision is that more application scenarios will be analyzed in order to provide a broad overview of the parameter impact. It is envisioned to extend existing hardware benchmarks from different application domains in terms of a parameterization guideline also for further FPGA series from Xilinx.

Furthermore, the paper provides a tutorial for the estimation of the power consumption on a high level of abstraction, but with a high accuracy through post place and route simulation. Therefore, other research in this area can be done and exchanged in the community.

Acknowledgment

The authors would like to thank Prof. Alba Cristina M. A. de Melo and Jan Mendonca Correa for providing us with their C code implementation of the DIALIGN algorithm.

References

[1] "Xilinx MicroBlaze Reference Guide", UG081 (v7.0), September 15, 2006, available at: http://www.xilinx.com.

[2] D. Meintanis, I. Papaefstathiou: "Power Consumption Estimations vs Measurements for FPGA-based Security Cores"; International Conference on Reconfigurable Computing and FPGAs 2008 (ReConFig 2008), Cancun, Mexico, December 2008.

[3] J. Becker, M. Huebner, M. Ullmann: "Power Estimation and Power Measurement of Xilinx Virtex FPGAs: Trade-offs and Limitations"; In Proc. of the 16th Symposium on Integrated Circuits and Systems Design (SBCCI'03), Sao Paulo, Brazil, September 2003.

[4] K. Poon, A. Yan, S.J.E. Wilton: "A Flexible Power Model for FPGAs"; In Proc. of 12th International Conference on Field-Programmable Logic and Applications (FPL 2002), September 2002.

[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] K. Weiss, C. Oetker, I. Katchan, T. Steckstor, W. Rosenstiel: "Power estimation approach for SRAM-based FPGAs"; In Proc. of International Symposium on Field Programmable Gate Arrays (FPGA'00), pp. 195-202, Monterey, CA, USA, 2000.

[7] V. Degalahal, T. Tuan; "Methodology for high level estimation of FPGA power consumption"; In Proc. of ASP–DAC 2005 Conference, Shanghai, January 2005.

[8] "Xilinx Power Estimator User Guide", UG440 (v3.0), June 24, 2009, available at: http://www.xilinx.com.

[9] "Development System Reference Guide", v9.2i, Chapter 10 XPower, available at: http://www.xilinx.com.

[10] "Embedded System Tools Reference Manual", Embedded Development Kit, EDK 9.2i, UG111 (v9.2i), September 05, 2007, Chapter 3, available at: http://www.xilinx.com.

[11] "Fast Simplex Link (FSL) Bus (v2.00a)"; DS449 Dec. 1, 2005, available at http://www.xilinx.com.

[12] "Virtex-4 FPGA Configuration User Guide", UG071 (v1.11), June 9, 2009, available at: http://www.xilinx.com.

[13] "Virtex-4 FPGA User Guide", UG070 (v2.6), December 1, 2008, available at: http://www.xilinx.com.

[14] C. A. R. Hoare: "Quicksort"; Computer Journal, vol. 5, 1, 10–15 (1962).

[15] A. Boukerche, J. M. Correa, A. C. M. A. de Melo, R. P. Jacobi: "A Hardware Accelerator for Fast Retrieval of DIALIGN Biological Sequence Alginments in Linear Space"; IEEE Transactions on Computers, vol. 59, no. 6, pp. 808-821, 2010.

[16] Alpha-Data: http://www.alpha-data.com