

# Plug-in of Power Models in the StepNP Exploration Platform: Analysis of Power/Performance Trade-offs

Giovanni Beltrame Gianluca Palermo Donatella Sciuto Cristina Silvano

Politecnico di Milano  
Dipartimento di Elettronica e Informazione  
Via Ponzio 34/5  
20133 Milano, Italy

{beltrame,gpalermo,sciuto,silvano}@elet.polimi.it

## ABSTRACT

In this paper, we propose a power/performance estimation layer designed for StepNP, a system-level architecture simulation and exploration platform for Network Processors and Multi-Processor Systems-on-Chip (MP-SoCs). The first goal of our work is to plug-in PIRATE, a parameterizable Network on-Chip in the StepNP platform, to support a fast exploration of on-chip interconnection networks. Up to now, StepNP does not provide any energy profiling, so our second goal is to dynamically plug-in power models of the different system components to provide power estimates quickly. The proposed power/performance exploration framework is based on a power characterization methodology and a system-level simulator to dynamically profile the given network application. This framework is intended to be used at different levels of the design, considering several levels of accuracy and taking full advantage of the StepNP performance profiling features. Experimental results are provided for the exploration of an ARM-based MP-SOC including a configurable NoC-IP executing an IPv4 forwarding application.

## Categories and Subject Descriptors

B.4.3 [Input/Output and Data Communications]: Interconnections—*Topology*; C.1.4 [Processor Architecture]: Parallel Architecture; C.4 [Performance of Systems]: Measurement techniques

## General Terms

Measurement, Performance, Design

## Keywords

Multiprocessor, Platform Based Design, Low-Power Design, Network on Chip

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CASES'04, September 22–25, 2004, Washington, DC, USA.  
Copyright 2004 ACM 1-58113-890-3/04/0009 ...\$5.00.

## 1. INTRODUCTION

The large diffusion of *Multi-Processor Systems-on-Chip (MP-SoC)* for embedded applications requires a flexible tuning framework to support the *Design Space Exploration-DSE* phase. The main goal of this phase is to find the optimal configuration of the parameterized MP-SoC platform in terms of both energy and performance requirements for the target application. The design space exploration of the *communication* infrastructure for MP-SoCs requires the definition of a design framework to support the analysis and comparison of different on-chip micro-network architectures from the power and performance combined perspective. In MP-SoC architectures, the communication infrastructure can connect up to tens of master and slave IP nodes through the network architecture. In many cases, the target MP-SoC architecture is based on the *Network-on-Chip (NoC)* approach, where on-chip communication represents the core of the overall system design. The main focus of the NoC approach consists of shifting from a deterministic *bus-based* communication among IP nodes towards a *communication-based* design methodology [1], exploiting data packetization and non-deterministic communication protocols used for general networks.

In this paper, we propose an approach for system-level power exploration based on the *StepNP platform*. StepNP is a system-level architecture simulation and exploration platform for Network Processors and MP-SOCs developed by STMicroelectronics [2], [3]. The main modelling language of the StepNP platform is SystemC 2.0 [4].

To support a fast and flexible exploration of several on-chip interconnection networks, the *first goal* of our research is to plug in the StepNP platform a configurable Network-on-Chip IP core (namely, *PIRATE* [5]). The PIRATE NoC-IP is composed of a set of parameterized modules, such as interconnection elements and switches, to generate different network topologies. In particular, the PIRATE NoC-IP is supported by a synthesizable RTL generator of the configurable NoC-IP core, an automatic power characterization flow, and a cycle-based NoC simulator for the dynamic profiling of power and performance at the system-level.

Up to now, StepNP does not provide any energy profiling, so the *second goal* of our research is to define a general power performance estimation framework designed for StepNP. The proposed framework is based on an automatic power characterization methodology and a system-level sim-

ulator to dynamically profile the given network application. A set of power models can be dynamically plugged-in to provide accurate and efficient power figures for the different architectures. The proposed framework is intended to be used at different levels of the design, considering various levels of accuracy and taking full advantage of the StepNP performance profiling features. The proposed methodology to plug-in power models in a high-level platform has been designed to be easily retargetable to other Hw/Sw co-design frameworks.

The proposed methodology has been experimented on the design exploration of an ARM-based multiprocessor system executing the IPV4 forwarding application. The target MP-SOC is composed of four ARM v7 core processors, the configurable memory sub-system, the Concurrency Engine (to support both thread and processor level parallelism), the Packet Engine (to support packet I/O). In the reported experiments, we simulate and compare three different network topologies connecting the system nodes through the NoC-IP: *Double-Ring*, *Mesh* and *Octagon*. For all topologies, the IPV4 packet forwarding application runs at 1 Gbps and it has been automatically mapped to the target MP-SoC by using the MultiFlex tools, supporting multi-processing by the DSOC (Distributed System Object Component). The long term goal of our research is the definition of a multi-objective design space exploration framework for network-centric MP-SoC architectures, considering both performance and energy consumption of the target network applications.

The paper is organized as follows. In Section 2, after a brief review of the most interesting approaches appeared in literature to support the exploration of MP-SoCs based on networks on-chip, the StepNP simulation and exploration framework is described. Section 3 shows our target network-centric MP-SoC architecture, while Section 4 describes the configurable Network-on-Chip IP core. The plug-in of the power models of the different system components in the StepNP platform is described in Section 5. Experimental results are reported in Section 6, while some concluding remarks and future evolutions of the present work are summarized in Section 7.

## 2. PREVIOUS WORK

On-chip micro-network architectures and protocols have been recently explored in literature, in some cases focusing on specific system configurations and application classes. The *SPIN Micronetwork* [6] represents an on-chip micro-network based on deterministic routing. The *Silicon Backplane Micronetworks* [7] based on a shared-medium bus and time-division multiplexing offers an example of transport layer issues in micro-network design. The *Octagon* on-chip communication architecture for OC-768 Network Processors has been proposed in [8].

The design of high-performance, reliable, and energy efficient interconnect-oriented MP-SoCs raises new challenges in terms of design methodologies ([1], [9]). Low-level power estimation tools (such as Synopsys Power Compiler) require synthesized RTL descriptions. Although these tools provide good levels of accuracy, they require long simulation time, becoming an unfeasible alternative for complex MP-SoCs. The exploration of MP-SoCs requires architectural-level power and performance simulators such as *SimpleScalar* [10], *Wattch* [11], and *Platone* [12] tools. A set of system-level power optimization tools are presented in [13].

The *Orion* approach [14] represents a power-performance interconnection network simulator to explore power performance trade-offs at the architectural level. The Orion framework can generate a simulator starting from a micro architectural specification of the interconnection network. A set of architectural level parameterized power equations have been defined to model the main modules of the interconnection network (such as FIFO buffers, crossbars, and arbiters) in terms of estimated switch capacitances to support dynamic simulation.

Power models for on-chip interconnection architectures have been recently proposed in literature. A survey of energy efficient on-chip communication techniques has been recently presented in [15]. Techniques operating at different levels of the communication design hierarchy have been surveyed, including circuit-level, architecture-level, system-level, and network-level. Patel et al. [16] focused on the need to model power and performance in interconnection networks for multiprocessor design, and proposed a power model of routers and links. An estimation framework to model the power consumption of switch fabrics in network routers has been proposed in [17]. The work introduces different modelling methodologies for node switches, internal buffers and interconnect wires for different switch fabric architectures under different traffic throughputs. The proposed modelling and simulation framework is limited only to switch fabrics. More recently, the same authors introduced in [18] a packetized on-chip communication power model for multiprocessors network design.

### 2.1 StepNP Simulation Platform

In this scenario, StepNP represents a system-level architecture simulation and exploration platform for Network Processors and MP-SoCs developed by STMicroelectronics [2], [3]. The main modelling language of the StepNP platform is SystemC 2.0.

StepNP consists of a Network Processor Unit (NPU) architecture simulation platform, to support a fully programmable architecture based on standard *RISC* processors and interconnect network. The basic platform consists of three main types of components:

- *Processor engines*: RISC-based architectures provided with configurable hardware multi-threading capability and a configurable  $n$ -stage pipeline;
- *NoC communication channel*: To support the high-level split-transaction channel model;
- *Specialized coprocessors* such as a concurrency engine.

The basic StepNP architecture platform includes the public-domain models of the ARM v7 and PowerPC instruction-set architectures and the Stanford DLX processor model. The StepNP architecture simulation platform is based on the encapsulation of functional instruction-set models into a SystemC wrapper. The encapsulation produces a *cycle-based* model implementing a configurable hardware multi-threading capability and a simple variable  $n$ -stage pipeline support. Each thread in the SystemC processor wrapper calls the ISS (Instruction Set Simulator) to implement the thread instructions. The ISS returns memory reference operations to the wrapper for implementation in SystemC. The wrapper communicates with the StepNP platform through the SystemC Open Core Protocol (SOCP) communication

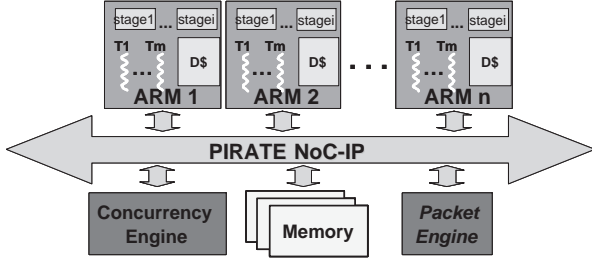


Figure 1: Target MP-SoC architecture.

channel interface. Users can integrate new coprocessor models into StepNP by using the SOCP interface. Currently, the platform includes a semaphore engine model to support concurrency management.

The interconnect model developed in the basic StepNP platform for the communication channel is a simple functional model supporting split-transaction communication, however one of the main goals of the platform is to support a standardized interface to plug-in IP components described at several abstraction levels without any assumption about the underlying interconnect architecture. In this paper, we exploited this capability of the StepNP platform by plugging-in our custom configurable NoC-IP module (namely PIRATE) composed of a set of parameterizable interconnection elements and switches connected by different topologies.

In StepNP, the communication channel interface is described by the SOCP channel interface model developed in SystemC 2.0 language. The model follows the high-level semantic of the Open Core Protocol (OCP) and the Virtual Component Interface (VCI). The StepNP platform supports an instrumentation methodology and an associated language, namely the SystemC Interface Definition Language (SIDL). The SIDL interface enables the external *control-and-view* components written in different languages to access a SystemC model object implementing this interface.

### 3. TARGET MP-SOC ARCHITECTURE

In this paper, we describe the application of the proposed power and performance exploration methodology to a domain-specific flexible MP-SOC architecture oriented towards networking applications derived from the StepNP platform. In particular, our target configurable MP-SOC architecture includes the following modules (see Fig. 1):

- Four to eight ARM v7 processor cores with a variable number of pipeline stages;
- Four to eight hardware threads per processor;
- The configurable memory sub-system;
- The Concurrency Engine to support both thread and processor level parallelism;
- The Packet Engine to support packet I/O.

The system modules have been connected through the parameterizable PIRATE NOC-IP described in the next section.

### 4. PIRATE NOC-IP ARCHITECTURE

In this section, we describe the plug-in of the Network-on-Chip IP core (namely, PIRATE [5]) in the StepNP simulation and exploration platform. PIRATE is composed of a set of parameterizable interconnection elements and switches connected by different topologies. An example of PIRATE using the Octagon topology and the switch architecture is reported in Figure 2. An input (output) queue is provided for each input (output) port of the switch. The length of each input (output) queue is configurable. An interconnection system is included into the switch to connect the input queues to the output queues. The internal interconnection system is based on a *crossbar topology*, where the  $N \times N$  network connects the  $N$  input queues with  $N$  output queues. Any of the  $N$  input queues can be connected to any of the  $N$  output queues by the corresponding cross-point of the crossbar switch. The input (output) queues and the crossbar of the switch are controlled by the Switch Controller, that includes the static routing table and the arbitration logic.

The PIRATE switch is a synchronous element requiring a one-cycle delay, when the queues are empty. The switch is based on wormhole routing and static routing defined by a routing table customized by the designer. To guarantee a high throughput, the switch architecture has been optimized. The one-cycle delay for each hop has been guaranteed by using a by-passing mechanism of the queues, when they are empty, and implementing a crossbar on the switch-interconnection between input and output ports.

The architecture of the PIRATE NoC-IP is parametric in terms of:

- *Switch Architecture*: Number of input and output ports, length of input and output queues and width of the connections;
- *Network Topology*: A set of parameters define the connection topology of the switches in the network. The explored standard network topologies are the Ring, Double Ring, Mesh, Cube, Binary-tree and Octagon. Other network topologies can be generated *ad hoc* to optimize the communication in the target architecture;
- *Information Flow in the NoC*: The routing mechanism in the switches is based on a static routing table. The designer can customize the routing table for each switch to balance and to optimize the traffic in the network.

The PIRATE NoC-IP simulatable model has been developed at system-level by using SystemC 2.0. The model has been developed at timed functional level, therefore providing accurate timing information and power estimates. The model is completely configurable in terms of all NoC micro-architectural parameters described above.

The design of the PIRATE NoC-IP is supported by a methodology to automatically generate the power model of each element of the interconnection system to enable the power profiling of the entire interconnection system. The power models are derived from [17]. For each component, an analytical model based on its design space and the traffic system level information can be automatically generated. More details about the automatic power characterization methodology can be found in [5].

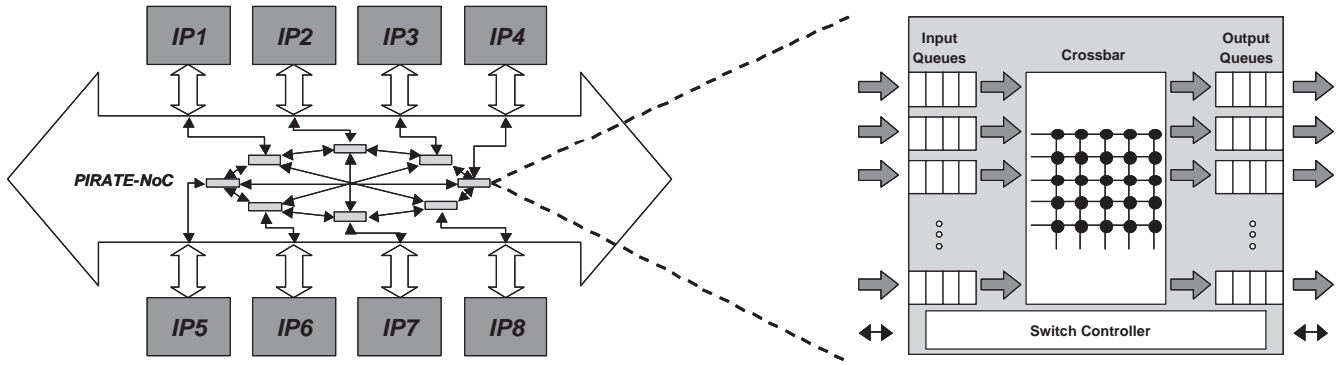


Figure 2: Example of the PIRATE NoC connecting 8 IPs by Octagon topology (left) and details of PIRATE switch architecture (right)

## 5. PLUG-IN OF POWER MODELS

At the time of writing, the StepNP simulation platform did not provide any energy profiling data. In this section, we describe the plug-in of system-level power models in the StepNP platform. In StepNP, all components are modelled as SystemC modules at different levels of abstraction starting from a functional description to the RT level.

We propose an approach to automatically integrate power models into the StepNP platform. The basic idea is to add a general and flexible energy framework that will be used for power estimation, model exploration and tuning. This framework represents a *power layer* between component models and power models. In this way, a power model can be plugged into the platform by attaching it to a given component. The main advantage of such an approach consists of the capability to use different models for the same component (to explore trade-offs between simulation speed and model accuracy).

The basic idea consists of using performance simulation data to compute power figures. To obtain power estimates, the models may require additional statistical parameters to bind simulation data to physical architectures. For this reason, each model requires to access a database of parameters, that will be used in conjunction with simulation data. These parameters are computed off-line and only once during the model tuning phase. This operation may be very time consuming and for this reason it has been automated by integrating the model tuning operations inside each power model. This means that each model has two operating modes: standard and tuning. In the standard mode, the model uses database and simulation data to compute power figures for a specific simulation run; in tuning mode, instead, data is collected from simulation and injected in the database. After the simulation is finished, data may be analyzed and model parameters estimated.

This approach implies the power estimation flow shown in Figure 3:

- *Model selection:* An *ad-hoc* model or a well known model is selected for the considered component or class of components;
- *Technology library building:* The model is tuned on a target technology or architecture. As an example, the processor model can be tuned on a specific instruction set architecture like ARM or microSPARC;

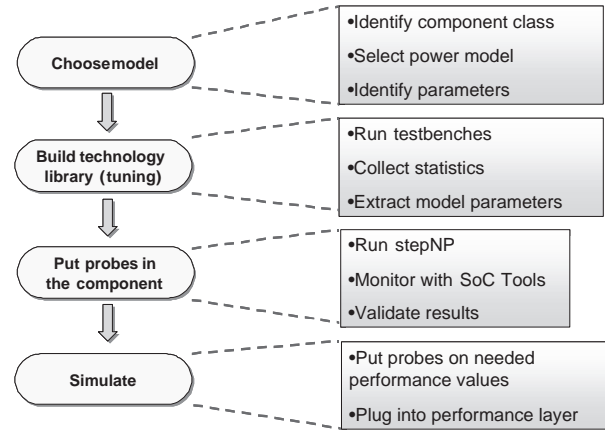


Figure 3: Power estimation flow.

- *Probe insertion:* Probes must be put into the component using StepNP probing facilities. A common name is assigned to each probe, so that it can be identified by models and external software;
- *Model validation:* Each model must be validated against physical devices to guarantee model accuracy. This is done via simulation and comparison.

It is worth noting that this flow is identical for each class of components, and it is indeed a standard flow for statistical analysis. The novelty of the approach is given by the flexibility introduced by the developed software platform: any combination of simulation accuracy may be obtained. Since each component can be specified at different levels of abstraction, a number of power models, with different accuracies, may be applied in several combinations to the platform with a reduced development effort.

The power estimation architecture is composed of three components as shown in Figure 4:

- *The ModelManager:* This component acts as the glue between platform components and power models. It extracts performance simulation data from a given *class* of components and it feeds them into the connected power models through probe streams;

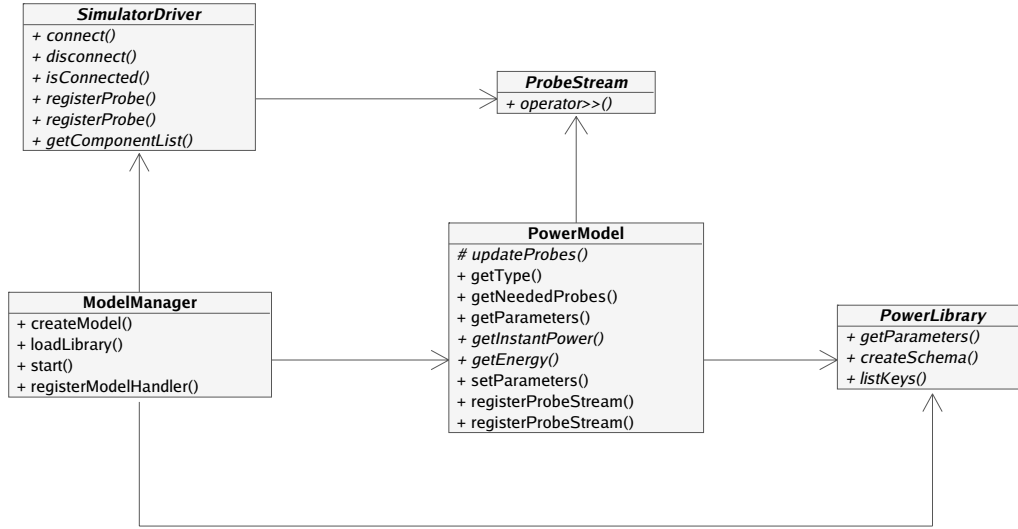


Figure 4: Power estimation framework.

- *The ProbeStream(s)*: These are abstractions of the data flow between the simulator and the power framework, providing probe values whenever these are produced;
- *The SimulatorDriver*: Provides common primitives to access a generic simulator, in our case the driver for the StepNP platform has been developed;
- *The PowerModel(s)*: Each class of components, or even each component if necessary, has a power estimation model, that uses simulation data to compute power figures.
- *The PowerLibrary*: This is simply an interface with a standard database that is used to read model parameters and to store tuning data.

To provide flexibility, power models and simulator drivers have been separated. The standard mode of simulation works as follows. First, data are extracted from the platform using the driver. Every time a model is connected to a component, it requests for a set of probes by common names. The ModelManager must connect such probes to the model and then exports (using the SIDL interface) methods for external tools like GUIs. In this way, at runtime, the models probe the component and technology database, and store the required data internally. Every time an external event (such as a GUI command) requests power figures, these are estimated by using the stored data and they are provided to the requester through the SIDL interface.

In tuning mode, the platform works mostly in the same way, but data is not stored internally by models. It is instead sent to the technology library component that will store it into the database for future statistical analysis. It is worth noting that not all technology libraries could be built in this way. According to model accuracy, additional data (such as data from hardware synthesis) could be needed.

## 5.1 Processor Power Model

The adopted power model is based on a functional decomposition of the activities carried out by a generic processor.

The problem of the identification of a functional model for the energy consumption at the instruction level has been investigated in [19] by considering the relation between the processor architecture and a set of functionalities. The resulting functional model exhibits general capabilities for a broad range of 32-bit microprocessor architectures.

Let  $m$  be the cardinality of the considered instruction set  $S$ . The energy associated with the SW program is expressed by:

$$E_{SW} = \sum_{s=1}^m \sum_{j=1}^k n_s \times e_{s,j} = \sum_{s=1}^m V_{dd} \times n_s \times i_s \times n_{ck,s} \times \tau \quad (1)$$

where  $e_{s,j}$  is the energy absorbed by the  $j$ -th functionality involved in the instruction  $s \in S$ ,  $k$  is the number of functionalities considered,  $n_s$  is the number of  $s$ -instructions executed in the application,  $V_{dd}$  is the supply voltage,  $n_{ck,s}$  is the total number of clock cycles of the instruction  $s$  and  $\tau$  is the clock period. The current  $i_s$  absorbed during the instruction  $s$  can be defined more in detail as:

$$i_s = \sum_{j=1}^k \delta_{j,s} i_j \quad (2)$$

where  $i_j$  is the current absorbed by the  $j$ -th functionality and  $\delta_{j,s}$  is a coefficient that is equal to one if the instruction  $s$  needs the  $j$ -th functionality, otherwise it is equal to zero.

## 5.2 Memory Power Model

For on-chip L1 caches, we used the analytical energy model developed in [20], that accounts for: (a) Technological parameters (such as capacitances and power supplies); (b) Architectural parameters (such as block size and cache size); (c) Switching activity parameters (such as number of bit line transitions). The switching activity parameters in the cache energy models have been computed by directly importing the actual values of hit/miss rates and bit transitions on cache components, that have been derived by the system-level simulation environment to account for the actual profiling information depending on the execution of embedded applications.

As energy model for the SRAM memory (different from caches), we used a model derived from [21]. In the proposed memory model, the energy dissipated by the SRAM memory explicitly distinguishes between read and write accesses and is given by:

$$E_{SRAM} = \sum_{i=1}^{nb} \left( \begin{array}{c} n_{i,read} \times E_{i,read} + \\ n_{i,write} \times E_{i,write} + \\ n_{i,idle} \times E_{i,idle} \end{array} \right) \quad (3)$$

where  $nb$  is the number of memory banks,  $n_{i,read}$  and  $n_{i,write}$  represent the number of reads and writes for the bank  $i$  respectively, while  $n_{i,idle}$  is the number of cycles with no accesses to the bank  $i$ ,  $E_{i,read}$ ,  $E_{i,write}$  and  $E_{i,idle}$  are the energy dissipated by a read access, a write access and idle cycle respectively.

### 5.3 NoC-IP Power Model

A methodology to automatically create the power models of the PIRATE NoC-IP has been defined and implemented to dynamically profile at system-level the power behavior of each module of the interconnection system. The methodology implemented is mainly composed of a power characterization of each parameterized module of the interconnection system (i.e. switch, encoder/decoder). For each module, at the end of the characterization flow, we automatically generate the corresponding analytical model based on its design space parameters and the traffic information derived from simulation.

We define the *design space* as the set of all feasible architectural implementations of a module. To exemplify our method, let us consider the power model of the switch. According to [17], the switch power model is dependent on the traffic and the design space parameters:

$$P_{SW}(a, TR) = B_0(a) + B_1(a) \times TR \quad (4)$$

where  $TR$  is the traffic factor,  $a$  is a generic point in the architectural design space  $A$  defined as:

$$A = S_{p_1} \times \dots \times S_{p_l} \dots \times S_{p_n} \quad (5)$$

where  $S_{p_l}$  is the ordered set of possible configurations for parameter  $p_l$  and " $\times$ " is the cartesian product. The  $B_0$  and  $B_1$  coefficients in Eq.(4) have been derived from gate-level power characterization by using STMicroelectronics 0.13 $\mu$ m technology.

## 6. EXPERIMENTAL RESULTS

The target MP-SoC platform described in Figure 1 has been configured with a set of fixed parameters and a set of variable parameters to explore, such as the number of threads per processor, interconnect and memory latency, NOC topology, etc.. In particular, the simulated target architecture includes the parameterizable PIRATE NOC-IP interconnecting the following 12 nodes:

- Four ARM v7 processor cores configured with 4 pipeline stages and a variable number of threads per processor;
- The configurable memory sub-system composed of 3 nodes: Global SRAM, Stacked Memory, Packet Memory;
- The Concurrency Engine to support both thread and processor level parallelism;

- The Packet Engine to support packet I/O;
- The Horba (Hardware Object Request Broker) Engine to coordinate objects communication;
- The Client Ware Master;
- DSOC (Distributed System Object Component) Engine.

Up to now, our framework supports the plug-in of the power models of ARM processors, on-chip L1 cache, global memory, stacked and packet memories and NOC-IP core. In the set of experiments reported in the paper, we simulate and compare three different network topologies connecting 12 nodes through the NoC-IP: *Double-Ring*, *Mesh* and *Octagon*. For all topologies, the IPv4 packet forwarding application has been automatically mapped to the target system architecture through the MultiFlex [22] tools, that support multi-processing by using the DSOC (Distributed System Object Component) model. The DSOC model is a message-passing model with a very simple CORBA-like interface definition language for distributed object modelling and inter-object communication. The platform implements in hardware both message passing and task scheduling. The target application runs in one million simulation cycles.

Main goal of the reported experimental results is to compare alternative architectural solutions for power and performance point of view. In that sense, for a fast exploration, relative accuracy is more important than absolute accuracy of the power and performance estimates. In particular, Figure 5 reports the packet throughput for the target MP-SOC architecture for an incoming packet rate of 1 Gbps. Three topologies (D=Double Ring, M=Mesh and O=Octagon) have been compared by varying the number of threads. From 4 to 16 threads, Octagon outperforms the Mesh and Double Ring topologies. From 32 to 128 threads, the maximum throughput of 1 Gbps is reached and the three topologies show the same saturation behavior.

For an incoming packet rate of 1 Gbps, Figure 6 reports the power break-down for the target MP-SOC architecture in terms of ARM cores, cache, SRAM memories (global SRAM, stacked and packet memories) and NOC. As before, the three topologies (D=Double Ring, M=Mesh and O=Octagon) have been compared by varying the number of threads. The power values are almost proportional to the corresponding throughput reported in Figure 5. The percentage of power dissipated in the different system modules is maintained almost constant by varying the number of threads.

A detailed power break-down of the percentage values is reported in Table 1 for 32 threads, when all the three topologies reach the target throughput of 1 Gbps. On average, the 31.02% of the total energy is consumed by the four ARM cores, the 41.12% by the L1 cache, 14.14% by the SRAM blocks (packet memory, stacked memory and global SRAM) and 13.76% by the NOC-IP. Because of the nature of the IPv4 application, the extensive use of caches and communication is reflected by their high percentage of power dissipated. Comparing the three NoC topologies, Octagon consumes less energy (12.96%) with respect to the other two topologies. In terms of total power, the three analyzed system configurations based on different topologies provide similar values.

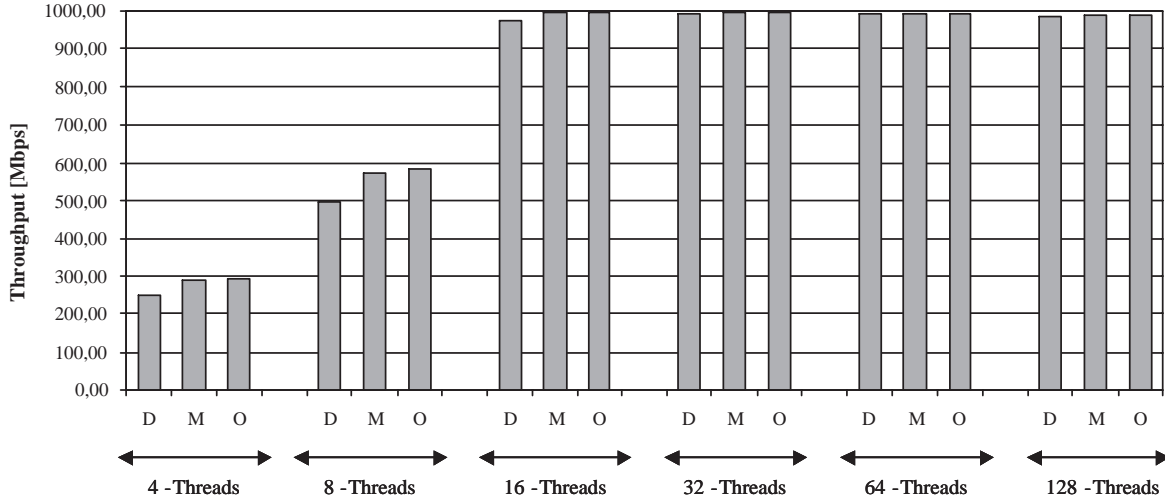


Figure 5: Packet throughput for the target MP-SOC architecture for different network topologies (D=Double Ring, M=Mesh and O=Octagon) by varying the number of threads for 1 Gbps incoming packet rate

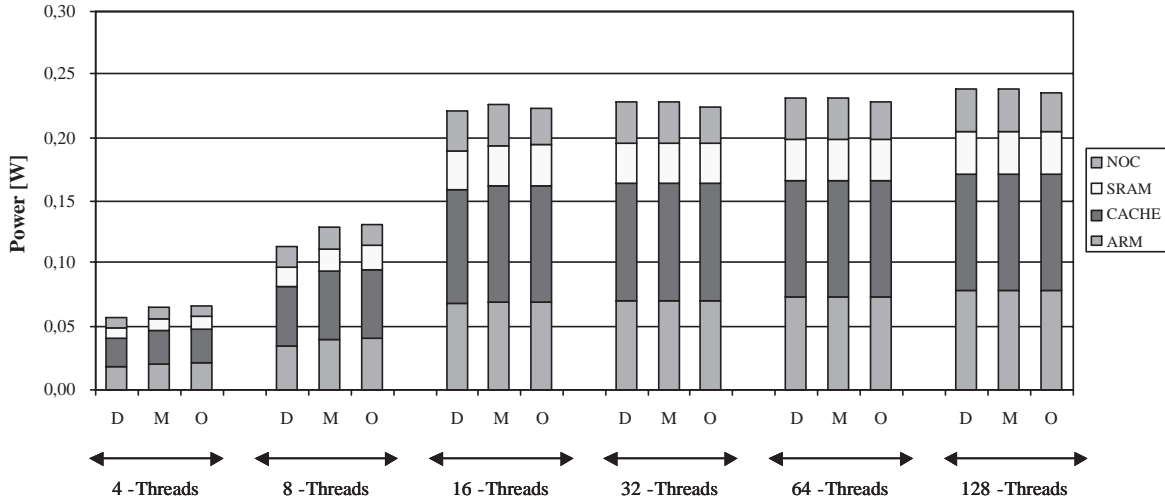


Figure 6: Break-down of power values for the target MP-SOC architecture for different network topologies (D=Double Ring, M=Mesh and O=Octagon) by varying the number of threads for 1 Gbps incoming packet rate

## 7. CONCLUSIONS AND FUTURE WORK

In this paper, an approach for power/performance system-level design space exploration based on the StepNP platform has been proposed. It consists of a configurable hardware multi-threaded multi-processor architecture, where hardware processing elements are connected through the PIRATE NoC. A power/performance estimation layer has been added to the StepNP framework, based on an automatic power characterization methodology and a system-level simulator to dynamically profile the given application. A set of power models can be dynamically plugged-in to provide accurate and efficient power figures for the different architectures. The methodology proposed for the plug-in of power models in the StepNP platform has been conceived in a gen-

eral way and therefore can be easily applied to other system level Hw/Sw co-design frameworks. In the paper, experimental results have been derived from mapping an Internet traffic management application running at 1 Gbps on the target MP-SoC. The experiments shown in the paper demonstrate how the plug-in of PIRATE NOC-IP and system-level power models into the StepNP HW/SW co-design framework enables the designer to explore and to derive communication-centric multi-processor solutions from the power performance combined perspective. Future work will focus on the integration of other parameterized models of system modules to increase the number of design choices and to provide a more reliable power and performance comparison for the target applications.

**Table 1: Power break-down for different network topologies in the target MP-SOC architecture with 32 threads executing the IPv4 forwarding application at 1 Gbps.**

Topology	4 ARM	4 L1 Caches	SRAM blocks	NOC-IP	Total Power [mW]
Double Ring	30,88%	40,93%	14,05%	14,14%	227,4
Mesh	30,90%	40,95%	14,08%	14,17%	227,3
Octagon	31,29%	41,47%	14,28%	12,96%	224,5
Average	31,02%	41,12%	14,14%	13,76%	226,4

## 8. ACKNOWLEDGMENTS

The authors would like to thank Pierre G. Paulin at STMicroelectronics laboratory in Ottawa who provided us with the StepNP simulation framework and Diego Vagni, Politecnico di Milano, for helpful discussions and feedbacks.

## 9. REFERENCES

- [1] L. Benini and G. De Micheli. Networks on Chip: A New SoC Paradigm. *IEEE Computer*, 35(1):70–78, January 2002.
- [2] P. G. Paulin, C. Pilkington, and E. Bensoudane. StepNP: A System-Level Exploration Platform for Network Processors. *IEEE Design and Test of Computers*, pages 2–11, November–December 2002.
- [3] P. G. Paulin, C. Pilkington, E. Bensoudane, M. Langevin, and D. Lyonard. Application of a Multi-Processor SoC Platform to High-Speed Packet Forwarding. In *DATE-04: Proceedings of Design, Automation and Test in Europe*, March, 2004.
- [4] SystemC 2.0 User’s Guide. *www.systemc.org*.
- [5] G. Palermo and C. Silvano. PIRATE: A Framework for Power/Performance Exploration of Network-On-Chip Architectures. In *PATMOS-04: Proceedings of International Workshop on Power and Timing Modeling, Optimization and Simulation*, September, 2004.
- [6] A. Adriahtenaina, H. Charlery, A. Greiner, L. Mortiez, and C. A. Zeferino. SPIN: A Scalable, Packet Switched, On-Chip Micro-Network. In *DATE-03: Proceedings of Design, Automation and Test in Europe*, March, 2003.
- [7] *www.sonicsinc.com*.
- [8] F. Karim, A. Nguyen, S. Dey, and R. Rao. On-Chip Communication Architecture for OC-768 Network Processors. In *DAC-38: Proceedings of the 38th Conference on Design Automation*, June, 2001.
- [9] A. Mihal K. Keutzer S. Malik J. Rabaey A. Sangiovanni-Vincentelli. M. Sgroi, M. Sheets. Addressing the System-on-a-Chip Interconnect Woes Through Communication-Based Design. In *DAC-38: Proceedings of the 38th Conference on Design Automation*, June, 2001.
- [10] D. Burger, T. M. Austin, and S. Bennett. Evaluating Future Microprocessors: The SimpleScalar Tool Set. Technical Report CS-TR-1996-1308, University of Wisconsin, 1996.
- [11] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: A framework for architectural-level power analysis and optimizations. In *ISCA-27: Proceedings of the 27th International Symposium on Computer Architecture*, pages 83–94, 2000.
- [12] T. D. Givargis and F. Vahid. Platune: A tuning framework for system-on-a-chip platforms. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 21(11):1317–1327, Nov. 2002.
- [13] L. Benini and G. De Micheli. System-level power optimization: Techniques and tools. *ACM Transactions on Design Automation of Electronic Systems.*, 5(2):115–192, 2000.
- [14] H. S. Wang, X. Zhu, L. S. Peh, and S. Malik. Orion: A Power-Performance Simulator for Interconnection Networks. In *MICRO-35: Proceedings of 35th International Symposium on Microarchitecture*, 2002.
- [15] M. B. Srivastava V. Raghunathan and R. K. Gupta. A Survey of Techniques for Energy Efficient On-Chip Communication. In *DAC-40: Proceedings of the 40th Conference on Design Automation*, pages 900–905, June, 2003.
- [16] C. Patel, S. Chai, S. Yalamanchili, and D. Schimmel. Power Constrained Design of Multiprocessor Interconnection Networks. In *ICCD-97: Proceedings of International Conference on Computer Design*, pages 408–416, October, 1997.
- [17] T. T. Ye, L. Benini, and G. De Micheli. Analysis of Power Consumption on Switch Fabrics in Network Routers. In *DAC-39: Proceedings of the 39th Conference on Design Automation*, pages 524–529, June, 2002.
- [18] T. T. Ye, L. Benini, and G. De Micheli. Packetized On-Chip Interconnect Communication Analysis for MPSoC. In *DATE-03: Proceedings of Design, Automation and Test in Europe*, pages 344–349, March, 2003.
- [19] C. Brandolese, F. Salice, W. Fornaciari, and D. Sciuto. Static Power Modeling of 32-bit Microprocessors. *IEEE Transactions on Computer-Aided Design*, 21(11):1306–1316, November 2002.
- [20] M. B. Kamble and K. Ghose. Analytical Energy Dissipation Models for Low Power Caches. In *ISLPED-97: ACM/IEEE International Symposium on Low Power Electronics and Design*, pages 143–148.
- [21] K. Itoh, K. Sasaki, and Y. Nakagome. Trends in Low-Power RAM Circuit Technologies. In *Proceedings of the IEEE*, volume 83, pages 524–543, 1995.
- [22] P. Magarshack and P. G. Paulin. How application/technology evolutions will shape classical EDA?: System-on-chip beyond the nanometer wall. In *DAC-40: Proceedings of the 40th Conference on Design Automation*, June, 2003.