

# Design Space Exploration and Prototyping for On-chip Multimedia Applications

Hyung Gyu Lee<sup>1</sup>, Umit Y. Ogras<sup>2</sup>, Radu Marculescu<sup>2</sup>, Naehyuck Chang<sup>1</sup>

<sup>1</sup>School of Computer Science and Engineering  
Seoul National University, Seoul, Korea  
{hglee,naehyuck}@cselab.snu.ac.kr

<sup>2</sup>Department of Electrical and Computer Engineering  
Carnegie Mellon University, Pittsburgh, PA, USA  
{uogras,radum}@ece.cmu.edu

## ABSTRACT

Traditionally, design space exploration for Systems-on-Chip (SoCs) has focused on the computational aspects of the problem at hand. However, as the number of components on a single chip and their performance continue to increase, a shift from computation-bound to communication-bound design becomes mandatory. Towards this end, this paper presents a comprehensive evaluation of two communication architectures targeting multimedia applications. Specifically, we compare and contrast the Network-on-Chip (NoC) and Point-to-Point (P2P) communication architectures in terms of power, performance, and area. As the main contribution, we present complete P2P and NoC-based implementations of a real multimedia application (MPEG-2 encoder), and provide direct measurements using a FPGA prototype and actual video clips, rather than simulation and synthetic workload. From an experimental standpoint, we show that the NoC architecture scales very well in terms of area, performance, power and design effort, while the P2P architecture scales poorly on all accounts except performance.

## Categories and Subject Descriptors

B.4.3[Interconnections (Subsystems)]: Topology (point-to-point, networks-on-chip)

## General Terms

Design, Measurement, Performance

## Keywords

Networks-on-chip, Point-to-point, System-on-chip, MPEG-2 encoder, FPGA prototype

## 1. INTRODUCTION

The increasing number of IP cores that can be integrated on a single chip enables implementation of complex applications using the SoC approach. Since these applications exhibit huge communication demands, scalable communication architectures are needed for efficient implementation of future systems.

Due to the lack of scalability, both in terms of power and performance, traditional bus-based communication architectures fail to satisfy the tight requirements of future applications. On the other

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, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

DAC 2006, July 24-28, 2006, San Francisco, California, USA.

Copyright 2006 ACM 1-59593-381-6/06/0007...\$5.00.

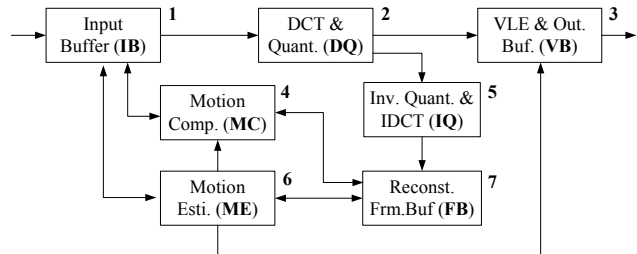


Figure 1. MPEG-2 encoder implementation using P2P communication architecture.

hand, the Point-to-Point (P2P) communication architectures can provide the utmost communication performance at the expense of dedicated channels among all the communicating IP pairs. In contrast to these methods, the Network-on-Chip (NoC) approach emerged as a promising solution to the on-chip communication problems [1,2,3].

## 1.1. Related Work

While it is intuitively accepted that NoCs can provide scalable communication with a small area overhead, this fact has not been justified to date by concrete NoC-based implementations of real applications. Theoretical studies rely mainly on simulation to justify such findings, so the network traffic used in such studies is either synthetic or approximating a real application via well-defined traffic generators [4,5]. On the other hand, several papers dealing with implementation issues in NoCs [6,7,8,9,14] do not consider the target application but assume synthetic traffic patterns instead. For instance, the authors in [6] present an MPEG-4 performance evaluation for a CDMA-based implementation of a NoC, but their design mimics the MPEG-4 traffic using a random traffic generator; this is clearly problematic given the complex nature of multimedia traffic [10].

## 1.2. Paper Contribution

To address the issues mentioned above, the contributions of this paper are twofold: First, we provide a complete MPEG-2 encoder implementation using both NoC and P2P communication architectures. Second, for several MPEG-2 implementations, we present extensive comparisons involving area, performance, and power consumption measurements using a FPGA prototype.

In terms of broader impact, both contributions are relevant to the larger class of embedded multimedia systems. Indeed, the MPEG-2 encoder has been selected as the driver application since it covers a rich class of multimedia applications where similar considerations apply from an implementation standpoint. For instance, JPEG, Motion-JPEG and MPEG-1 encoders, can all be implemented using similar architectures and set of IP cores.

We also note that, due to the small number of point-to-point connections in its architecture, the MPEG-2 encoder lends itself to a P2P implementation, as shown in Figure 1. Indeed, the link-to-node ratio in the MPEG-2 implementation shown in Figure 1 is only 1.5, while the same ratio is found to be 3.0 for a complete graph of same size even when all the links are unidirectional. It should be noted that for many applications where a subset of cores communicate with all the remaining nodes, the overhead incurred by the dedicated channels of the P2P architecture will be significant. As a result, the conclusions derived herein with respect to the benefits of the NoC architecture compared to the P2P architecture are rather conservative so they will shed light on a wide range of practical scenarios.

### 1.3. Overall Approach

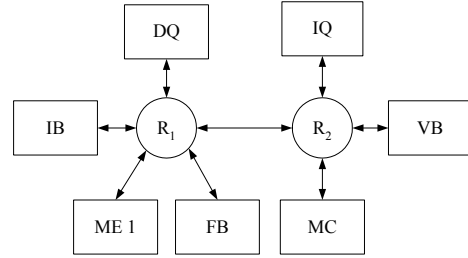
In the following, we first design an MPEG-2 encoder using a P2P communication architecture. After that, the same set of IP cores is used to implement the NoC-based version. We deliberately exclude the bus-based implementation from this analysis for several reasons. From an implementation standpoint, a bus-based design would clearly provide a much lower performance figure due to its limited bandwidth capabilities. Moreover, the large capacitive load for the bus drivers results in large delays and power consumption [15]; this makes the bus-based solution inappropriate for MPEG-2 encoder. Also, it is more interesting to see how the NoC solution compares to the P2P architecture, particularly since due to the high level of customization, the P2P approach is very difficult to beat in terms of performance and energy figures. Therefore, in what follows, our focus will remain on the P2P and NoC implementations. While NoCs gained recently a significant momentum in the research community, there are no complete NoC-based implementations of *real applications* reported to date. To remedy this situation, this paper presents a complete MPEG-2 encoder design using the NoC approach and compares it with a P2P architecture implementing the same application. As we will see later in the paper, our NoC-based MPEG-2 implementation achieves 47 *Frames/sec* encoding rate for a CIF frame of size 352×288, which is very close to the rate achievable by a P2P communication which is 48 *Frames/sec*. Although these values are close, the real benefits of using the NoC approach are observed when we analyze the scalability of these designs as a function of the number of cores. More specifically, we replicate the motion estimation module, which performs the most computationally expensive task, to exploiting the data parallelism available in MPEG-2. As the total number of modules in the design increases, the area occupied by the P2P implementation grows abruptly. On the other hand, the NoC implementation incurs only a modest area overhead, while keeping up with the performance increase achieved by the P2P implementation for a similar increase in the design size.

### 1.4. Paper Organization

The remaining part of this paper is organized as follows: Section 2 presents the details of the P2P and NoC-based implementations of the MPEG-2 encoder. Detailed area, performance and power consumption comparisons are provided in sections 3, 4 and 5, respectively. Finally, our conclusions appear in Section 6.

## 2. MPEG-2 ENCODER IMPLEMENTATION

The basic MPEG-2 implementation using the P2P approach is depicted in Figure 1. It consists of 7 modules: 1) *Input Buffer (IB)*, 2) *DCT & Quantization (DQ)*, 3) *Variable Length Encoder & Output Buffer (VB)*, 4) *Motion Compensation (MC)*, 5) *Inverse Quanti-*



**Figure 2. NoC-based implementation of MPEG-2 encoder with one ME module.**

*zation & Inverse DCT (IQ)*, 6) *Motion Estimation (ME)*, and 7) *Reconstructed Frame Buffer (FB)*. In our implementation, each intra (*I*) frame is followed by 3 predicted (*P*) frames. The P2P architecture obviously enables the fastest possible communication due to the existence of dedicated channels between all the communicating modules. On the other hand, the utilization of dedicated channels is low, since most of the time the links are idle. For instance, we measured an average utilization of the P2P links of only 4.9% using our MPEG-2 encoder prototype. In contrast, using a NoC communication architecture enables link sharing among the communicating cores. For instance, the NoC implemented for this encoder needs 8 links in the network as opposed 10 links needed for the P2P version, as shown in Figure 2. Each of link can be either bidirectional or unidirectional, depending on the functionality of the connected modules. Obviously, sharing links may cause extra communication delay compared to the P2P implementation, but performance degradation is negligible, as shown in Section 4.

### 2.1. Design of the Processing Elements

The area occupied by the individual cores in the design is summarized in Table 1. The second column (labeled “w/o wrapper”) shows the number of slices a core takes in a Xilinx XC2V4000 FPGA when implemented without any network interface<sup>1</sup> (wrapper). However, before using the core in a real design, we need to add a wrapper such that it can successfully communicate with the other nodes in the network. The wrapper for the P2P communication architecture has a simple flow control and I/O buffers so it takes only 116 slices to implement it. On the other hand, the NoC interface has to perform packetize/depacketize operations; this takes 189 slices. The third and forth columns in Table 1 show the area of the cores (in terms of number of slices and BlockRAMs) with all wrappers included. Although the basic wrapper for the P2P architecture is smaller than its NoC counterpart, the cores instantiated for the P2P architecture have actually a larger area. This is due to the fact that for the P2P architecture the modules have dedicated

**Table 1. Area comparison for individual cores in MPEG-2 (# of slices and BRAMs in a Xilinx Virtex2 4000 FPGA).**

Core	w/o wrapper	In P2P	In NoC
<i>IB</i>	74 slices (1BRAM)	394	263
<i>DQ</i>	2,527 (1)	2,868	2,716
<i>IQ</i>	3,873 (1)	4,082	4,062
<i>FB</i>	803 (75)	1,092	992
<i>ME</i>	956 (8)	1,346	1,145
<i>MC</i>	480 (19)	756	669
<i>VB</i>	961	1,196	1,150

1. Communication interfaces of the modules both in P2P and NoC implementations are referred as network interfaces (NI).

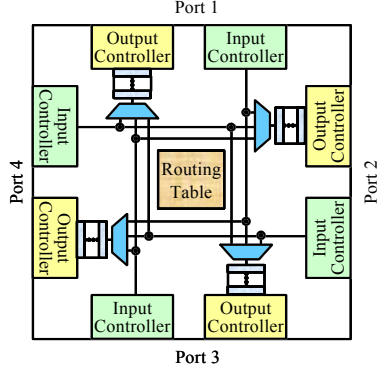


Figure 3. Block diagram of the router used in the NoC implementation.

interfaces for each connection, while for the NoC implementation each module has only one interface which connects it directly to the router.

## 2.2. Router Design

Due to moderate buffer requirements and good performance, our NoC uses wormhole routing. The block diagram of the router is shown in Figure 3. The network channel width and flit size is set to 16 bits. Each packet in the network contains one block in the current frame ( $8 \times 8 \times 16$  bits) divided into 64 flits. The router uses a priority-based scheduling. It takes only 4 cycles for the router to route the header flit. Then, the remaining flits follow the header in a pipelined fashion. Finally, the FIFO buffers implemented at the output ports of the router have a depth of 16 flits.

Table 2. Area taken by routers in a NoC implementation. Synthesis is performed for a Xilinx Virtex2 4000 FPGA.

Router type	# of slices	Device utilization
3-port router	219	1.4%
4-port router	304	1.8%
5-port router	397	2.2%
6-port router	503	2.8%

Based on the network topology, we use routers with 3, 4, 5 or 6 ports. The area occupied by the routers, as a function of the number of ports, is summarized in Table 2. Although none of the routers is optimized for area, their area overhead is smaller than the overhead incurred by the dedicated links and network interfaces of the P2P implementation; we discuss this in detail in Section 3.

## 3. EVALUATION OF DESIGN AREA

In this section, we compare the area occupied by the complete MPEG-2 encoder implemented with the P2P and NoC communication architectures. Besides the absolute value of area of the baseline designs in Figure 1 and Figure 2, we also analyze how the area scales up with the increasing number of cores. For this reason, we also implement a version of the MPEG-2 encoder which has 2 separate ME modules.

As shown with dotted lines in Figure 4(a), adding one more ME module to the P2P architecture requires 4 extra links and 8 network interfaces. In addition to this, the IB, MC, FB and VB modules have to be modified to allow the integration of the second ME module. On the other hand, the impact of this additional core on the NoC implementation is only local since we only need to add one more link from the newly inserted module to a router and one extra port inside the router (see Figure 4(b)). As a result, the effort of adding additional modules in order to increase the design parallel-

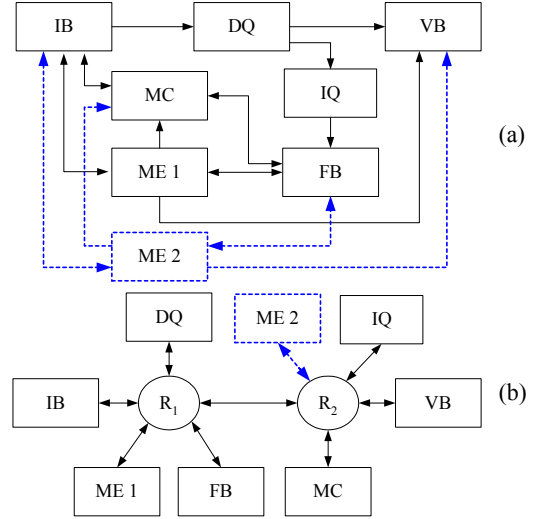


Figure 4. Implementation of MPEG-2 encoder with 2 MEs using (a) P2P and (b) NoC communication architectures.

ism and the penalty in area are both much smaller for the NoC design.

To be more concrete, the area of the P2P implementation on our FPGA prototype goes up from 11,587 to 13,501 slices resulting in a 16.5% increase. On the other hand, the area of the NoC design increases from 11,790 to 12,929 slices, which is about a 9.7% increase. Hence, the area for the NoC design with 2 ME modules is *smaller* than its P2P counterpart. It is also interesting to analyze the scaling effects for an arbitrary number of additional cores so we provide next an analytical framework to estimate it.

We analyze the area occupied by the logic components such as cores, network interfaces and routers, and the area occupied by the links, separately, because the interconnection mechanism of FPGAs is quite different from that in real silicon implementation due to reconfiguration facility of FPGAs.

### 3.1. Analytical Estimation of Area

The area occupied by the computation and communication resources in the design can be estimated by using the area of the individual modules, such as processing elements and all network interfaces. In the following,  $A(\cdot)$  denotes the area of its arguments,  $c_i$  ( $i=1, \dots, N_C$ ) denotes core  $i$ , while  $N_C$  is the total number of cores in the design. Finally,  $NI_{P2P}$  and  $NI_{NoC}$  stand for the network interfaces which correspond to the P2P and NoC designs, respectively. Using these notations, the area of the P2P implementation can be found as:

$$A_{P2P} = \sum_{i=1}^{N_C} A(c_i) + 2N_L \cdot A(NI_{P2P}) \quad (1)$$

where  $N_L$  is the number of links in the design. Similarly, the area of the NoC-based design is expressed as:

$$A_{NoC} = \sum_{i=1}^{N_C} A(c_i) + N_C \cdot A(NI_{NoC}) + \sum_{i=1}^{N_R} A(R_i) \quad (2)$$

where  $R_i$  ( $i=1, \dots, N_R$ ) denotes router  $i$ , while  $N_R$  is the total number of routers in the network.

Unlike the logic area, the interconnect area is difficult to estimate with simplified models, since it depends on the logic placement and space complexity. Hence, instead of using such models, we use an in-house communication-aware floorplanner

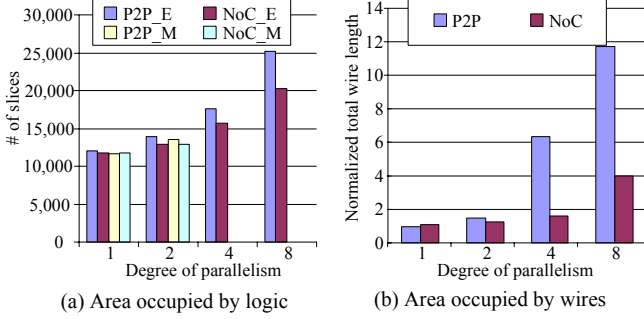


Figure 5. Area comparisons of the MPEG-2 encoder implemented using P2P and NoC architectures for increasing level of parallelism.

for both P2P and NoC implementations, and determine the total length of all wires [16].

### 3.2. Area Comparison of P2P and NoC Implementations

The area comparisons for the logic and wiring area of the P2P and NoC implementations are shown in Figure 5. The logic area estimates (*i.e.* the P2P\_E and NoC\_E bars) for the designs with one and two ME modules are within 4% of the measured values (P2P\_M and NoC\_M). We also note that the measured values are slightly smaller due to the optimization process during the synthesis of the complete design. For the MPEG-2 designs involving 4 and 8 ME modules, we use the estimated values as basis for comparison (*i.e.* P2P\_E and NoC\_E bars in Figure 5), since the total design area is larger than the capacity of the target FPGA.

As shown in Figure 5, starting with the designs involving 2 ME modules, both the logic and wiring area of NoC implementation is consistently smaller. More importantly, the difference in area increases as the number of cores becomes larger. In general, it has been shown that the P2P implementation scales as  $O(n^2\sqrt{n})$ , while NoC implementation scales as  $O(n)$  [11]. Hence, our experimental results are in agreement with the theoretical predictions.

## 4. PERFORMANCE EVALUATION

In this section, we develop an analytical model to estimate the throughput of the encoder. Then, we compare the performances of the P2P and NoC designs using both this model and measured data.

### 4.1. Analytical Estimation of Performance

Since both P2P and NoC implementations are pipelined, the encoder throughput is determined by the latency of the critical path on the data flow, as illustrated in Figure 6. More precisely, the critical path is given by:

$$T_{critical} = \max\{T_i\} \quad i \in S_C \quad (3)$$

where  $S_C$  is set of computational nodes and communication links on the critical path, and  $T_i$  is the latency of  $i^{th}$  node or link. From MPEG-2 functionality, it follows that for the I-frames, the critical path is given by  $S_C = \{T_{IB} \rightarrow T_{DQ} \rightarrow T_{VB}\}$ . Similarly, the critical path for the predicted (P) frames can be expressed as  $S_C = \{T_{IB} \rightarrow T_{ME} \rightarrow T_{MC} \rightarrow T_{FB} \rightarrow T_{MC} \rightarrow T_{IB} \rightarrow T_{DQ} \rightarrow T_{VB}\}$ . Since the critical path for the P-frames is significantly longer, we consider only the performance analysis of the P-frame encoding. Among the modules on the critical path for the P-frame encoding, the ME module requires the largest computational time. So, its latency value directly determines the overall performance of the system:

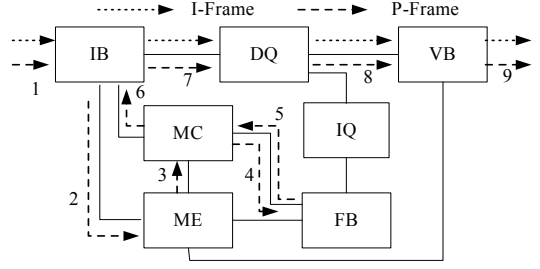


Figure 6. Critical paths for the I-Frame and P-Frame encoding are shown by the dotted and dashed lines, respectively.

$$Performance = \frac{1}{T_{ME} + L_{Data}} \quad (4)$$

where  $T_{ME}$  is the time required for motion estimation, and  $L_{Data}$  is the time for receiving data,  $N_{data}$  from the FB module. We note that  $T_{ME}$  is the same for both P2P and NoC designs, while  $L_{Data}$  is different. More precisely, for the P2P architecture,  $L_{Data}$  is the volume of data,  $N_{Data}$ , divided by the link bandwidth ( $W$ ):

$$L_{Data}^{P2P} = \left\lceil \frac{N_{Data}}{W} \right\rceil \quad (5)$$

For NoCs, on the other hand, we calculate the communication time using the latency formula for wormhole routing [12]:

$$L_{Data}^{NoC} = H \cdot L_R + \left\lceil \frac{N_{Data} - W}{W} \right\rceil \quad (6)$$

where  $H$  is the hop count between source and destination, and  $L_R$  is the time it takes to route the header flit. The first term in this equation gives the time it takes to route the header flit, while the second term gives the latency of the remaining flits, since they all follow the header flit in a pipelined manner.

### 4.2. Performance Comparison of P2P and NoC Implementations

The throughput of the P2P implementation is measured as 48 *Frames/sec* for a CIF frame of size 352×288. The corresponding NoC implementation achieves a throughput of 47 *Frames/sec*. As explained in the previous section, the bottleneck module in both designs is the ME module. For this reason, by duplicating this module, we expect a significant improvement in the throughput. The encoder implementation with two ME modules shows that this is indeed the case. Specifically, the P2P implementation with 2 ME modules has a throughput of 90 *Frames/sec*, which is about 87.9% improvement. Similarly, the throughput of the NoC design goes up to 86 *Frames/sec* showing a comparable improvement.

The accuracy of the performance estimates in Equation 5 and Equation 6 was validated against measured data on the FPGA prototype. As shown in Figure 7, the estimated values (*i.e.* the P2P\_E and NoC\_E bars) are in good agreement with the measured ones. Again, due to complexity reasons, for designs with higher degree of hardware redundancy (that is, the designs with 4 and 8 ME modules), we only provide the estimated values. We can clearly see that NoC-based implementations perform close to the P2P implementation. However, one should note that beyond a certain degree of parallelism, the communication performance becomes the bottleneck and the NoC performance settles down. It is possible to eventually stretch the performance beyond this point by customizing the network topology [5,13].

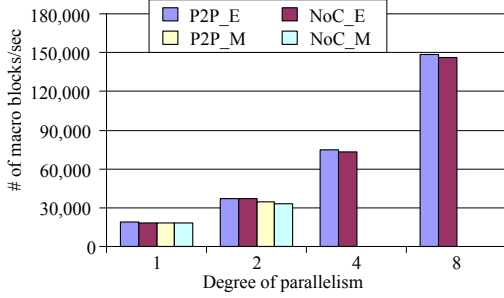


Figure 7. Performance comparison of the MPEG-2 encoder implementations. P2P\_E and NoC\_E show the analytical estimations, while P2P\_M and NoC\_M show the measurement.

## 5. ENERGY AND POWER CONSUMPTION EVALUATION

As battery-powered devices become popular, the energy consumption issues gain more importance. For this reason, this section evaluates energy and power consumptions for P2P and NoC communication architectures.

The activity-based model,  $P = \alpha CV^2 f$ , is the most commonly used model for power estimation at all levels of abstractions. However, as the complexity of systems increases, obtaining accurate  $C$  and  $\alpha$  values becomes more difficult. Therefore, we use a system-level approach to reduce the simulation time and the complexity of the work involved.

### 5.1. Analytical Estimation

The total system energy consumption can be divided into two components: the *computational* energy consumption,  $E_{COMP}$ , and the *communication* energy consumption,  $E_{COMM}$ . In turn, the computational energy can be expressed as:

$$E_{COMP} = \sum_{i=1}^{N_C} E(c_i) \quad (7)$$

where  $E(c_i)$  denotes the energy consumption of any component  $c_i$ , and  $N_C$  represents the total number of computational modules. The communication energy, on the other hand, consists of link ( $L$ ), network interface ( $NI$ ), and router ( $R$ ) energy consumption, namely:

$$E_{COMM} = \sum_{i=1}^{N_L} E(L_i) + \sum_{i=1}^{N_{NI}} E(NI_i) + \sum_{i=1}^{N_R} E(R_i) \quad (8)$$

where  $N_L$ ,  $N_{NI}$  and  $N_R$  represent the number of links, network interfaces and routers in the network, respectively.

Depending on the operating mode, the IP cores are characterized by different power consumption values. Therefore, we can analyze the energy consumption of any individual module  $E(c_i)$  as follows:

$$E(c_i) = \sum_{j=1}^{N_{LP}} (\pi_j P_j) \cdot t \quad (9)$$

where  $N_{LP}$  is the number of distinct power states,  $\pi_j$  is the probability that the module is in state  $j$  during execution time  $t$ , and  $P_j$  is the power consumption which characterizes the  $j^{\text{th}}$  power state<sup>1</sup>.

1. Here, the energy dissipation during transitions among different power modes is neglected since this represents a small fraction of the energy consumption in regular power states anyway.

In order to achieve accurate results, we measure the power consumption of each individual IP using a cycle accurate energy measurement tool based on the technique presented in [17]. Then, these individual power values are used to compute the power consumption of the entire system.

As seen from Equation 9, having power consumption estimates is critical for system energy characterization. Toward this end, we use two levels (*idle* and *active*) of power states to characterize the individual computational nodes, as summarized in Table 3. We note that the power consumption of the ME module dissipates less power than that of the DQ and IQ modules. However, the ME module consumes much more energy than these modules because its operation time is about ten times longer compared to that of the DQ and IQ modules.

For the communication components, we use a larger number of power states as summarized in Table 4. To accurately compute the link power consumption, we use an in-house communication-aware floorplanner for both P2P and NoC implementations, and determine the length of all links. Three typical values (which are classified as shortest, middle and longest) are shown in Table 4. After that, we measure the corresponding link power consumption in the FPGA prototype by varying the link length. These measured values are later used to estimate the link power consumption. Finally, we characterize the power consumption of the router, as shown in Table 4. The table also shows the router power consumption when the number of active ports varies.

Table 3. Power consumption (in mW@100MHz) for each computational node in the network

Node	Power in <i>idle</i> mode	Power in <i>active</i> mode
<i>IB</i>	60	109
<i>DQ</i>	353	1,279
<i>IQ</i>	420	1,755
<i>FB</i>	247	352
<i>ME</i>	133	411
<i>MC</i>	108	203
<i>VB</i>	210	626

Table 4. Power consumption (in mW@100MHz) over the communication channel

Resource	Mode	Power consumption	
		P2P	NoC
<i>Interface</i>	<i>Idle</i>	39	47
	<i>Receive</i>	74	69
	<i>Send</i>	78	69
	<i>Receive + Send</i>	92	101
<i>Link</i>	<i>Shortest</i>	12	
	<i>Middle</i>	16	
	<i>Longest</i>	19	
<i>Router</i>	<i>Idle</i>	NA	121
	<i>1 port</i>	NA	176
	<i>2 port</i>	NA	190
	<i>3 port</i>	NA	216
	<i>4 port</i>	NA	252

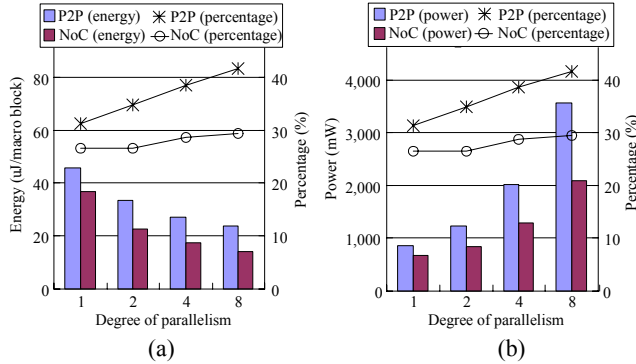


Figure 8. (a) Energy consumption and (b) Power consumption (@ 100MHz) as a function of the degree of parallelism for P2P and NoC implementations.

## 5.2. Energy Consumption Comparisons of P2P and NoC Implementations

We first obtain the utilization and operation modes of all the components in the design; this is done by dynamically profiling Verilog simulations using actual video clips. Then, we measure the energy consumption of each module separately; this is because the entire MPEG-2 encoder is too big to fit the memory space of the energy measurement tool we employ [17]. Finally, we use these values to estimate the energy and power consumption of the entire design, as explained in Section 5.1. Therefore, in the section we only provide estimated values for the energy and power consumption, unlike the area and performance comparisons reported before.

The power consumption of the computational modules is the same for both P2P and NoC implementations. Therefore, we focus next on the communication power consumption. Figure 8(a) shows that the communication energy consumption for the NoC implementation is consistently smaller than the P2P counterpart for different levels of parallelism. Likewise, we observe that the NoC design looks better in term of power consumption. Since the P2P implementation has more interfaces and links than the NoC counterpart, its power consumption is slightly larger than the power consumed by the NoC even for the baseline implementation. Furthermore, Figure 8(b) shows that the power consumption of the P2P architecture scales poorly as the degree of parallelism increases. Indeed, increasing the degree of parallelism makes the power consumption difference only bigger, since the P2P implementation requires a significantly larger number of additional links and network interfaces. Our experiments demonstrate that the NoC implementation consumes up to 42% less power compared to the P2P implementation for an implementation involving 8 ME modules. This corresponds to about 17% of the total power consumption which is quite important when optimizing portable systems.

Another issue of interest is the proportion of the communication energy consumption compared to the overall energy consumption. For this reason, we plot the percentage of the communication energy and power consumptions in Figure 8(a) and (b), respectively. For the baseline implementations, the communication power consumption of the P2P and NoC designs represents 31% and 26%, of the total power consumption, respectively. As we can see in Figure 8(b), the percentage of communication power consumption of the P2P implementation increases very fast with the increase in the degree of parallelism. On the other hand, the percentage of communication power in the NoC implementation increases much slower. This means that the NoC architecture has good scalability in terms of power as well as area and design complexity.

## 6. CONCLUSION

Integrating an increasingly large number of IP cores on the same chip makes the design of the communication architecture of future SoCs a challenging problem. As a result, design space exploration with emphasis on the communication aspects becomes crucial.

Towards this end, this paper presented a comprehensive evaluation of the P2P and NoC communication architectures targeting multimedia applications. Through analytical prediction and direct measurements on a FPGA-based prototype, it has been shown that the performance of the NoC implementation is very close to the P2P implementation of similar size. Moreover, the scalability of the P2P and NoC implementations was analyzed through duplicating the bottleneck module in the MPEG-2 design. It has been observed that NoC design scales very well in terms of area, performance, power and overall design effort, while the P2P architecture scales poorly on all accounts except performance.

## 7. ACKNOWLEDGEMENTS

This work was supported in part by the International Research Internship Program of the Korea Research Foundation (KRF) and in part by Marco GSRC.

## 8. REFERENCES

- [1] W. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," In *Proc. DAC*, June 2001.
- [2] L. Benin and G. De Micheli, "Networks on chips: A new SoC paradigm," *IEEE Computer*. 35(1), 2002.
- [3] A. Jantsch and H. Tenhunen (Eds.), *Networks on Chip*. Kluwer, 2003.
- [4] J. Hu and R. Marculescu, "Energy- and Performance-Aware Mapping for Regular NoC Architectures," In *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 24(4), April 2005.
- [5] K. Srinivasan, et. al., "Linear programming based techniques for synthesis of Network-on-Chip architectures," In *Proc. ICCD*, 2004.
- [6] M. Kim, D. Kim, and G. E. Sobelman, "MPEG-4 performance analysis for a CDMA network-on-chip," In *Proc. Intl. Conf. on Communications Circuits and Systems (ICCCS)*, 2005.
- [7] J. Liang, S. Swaminathan, and R. Tessier, "aSoC: a scalable, single-chip communications architecture," In *Proc. Intl. Conf. on PACT*, Oct. 2000.
- [8] A. Adriahtenaina and A. Greiner, "Micro-Network for SoC: Implementation of a 32-Port SPIN network," In *Proc DATE*, March 2003.
- [9] K. Lee, et. al., "A 51mW 1.6GHz On-Chip Network for Low-Power Heterogeneous SoC Platform," In *Proc. ISSCC*, San Francisco, Feb. 2004.
- [10] G. Varatkar and R. Marculescu, "On-chip traffic modeling and synthesis for MPEG-2 video application," *IEEE Tran. on VLSI*, 12(1), Jan. 2004.
- [11] E. Bolotin, et. al., "Cost considerations in network on chip," *Integration, the VLSI Journal*, 38(1), Oct. 2004.
- [12] J. Duato, et. al., *Interconnection Networks: an Engineering Approach*. Morgan Kaufmann, 2002.
- [13] U. Y. Ogras and R. Marculescu, "Energy- and performance- driven customized architecture synthesis using a decomposition approach," In *Proc. DATE*, 2005.
- [14] Jiang Xu, et. al., "A case study in networks-on-chip design for embedded video," In *Proc. DATE*, March 2004.
- [15] P.T. Wolkotte, et. al., "Energy model of networks-on-chip and bus," In *Proc. Intl. Symp. on System-on-Chip*, 2005.
- [16] J. Hu, Y. Deng, and R. Marculescu, "System-level point-to-point communication synthesis using floorplanning information," In *Proc. ASP-DAC*, Bangalore, India, Jan. 2002.
- [17] H. G. Lee, et. al., "Cycle-accurate Energy Measurement and Characterization of FPGAs," *Analog Integrated Circuits and Signal Processing*, 42(3), March 2005.