Approximate Wireless Networks-on-Chip

Giuseppe Ascia, Vincenzo Catania,
Salvatore Monteleone, Maurizio Palesi, and Davide Patti
Dept. of Electrical, Electronic and Computer Engineering
University of Catania
Catania, Italy
first.last@dieei.unict.it

John Jose
Dept. of Computer Science and Engineering
Indian Institute of Technology Guwahati
Guwahati, Assam, India
johnjose@iitg.ac.in

***Abstract*—Thanks to the forgiving nature of the emerging recognition, mining and synthesis applications, approximate computing (AC) has been recently rediscovered as a viable technique for improving the performance of computing systems. Although the application of AC techniques has, in several cases, an indirect positive effect on the performance of the on-chip communication sub-system, there are only few works aimed at proposing AC techniques specifically designed to improve the efficiency of the on-chip communication fabric. This paper introduces the concept of approximate communication in the context of wireless network-on-chip (WiNoC) architectures. This paper presents a technique through which the programmer can annotate those data structures of an application that, whenever affected by errors, do not impact the functionality of the application itself but only the quality of its outputs. Based on this annotation, the communications induced by the access to such data structures are realized with a reduced energy effort that, however, results to an increase of the probability for the data to be affected by errors. The underlying hardware mechanisms enabling the energy versus reliability trade-off are based on the dynamic link voltage swing and on the dynamic transmitting power tuning of the wired links and wireless transmissions, respectively. Both the hardware and software components needed for supporting the proposed technique are presented. The technique is assessed on a set of representative benchmarks and the energy saving vs. application output quality is discussed.**

***Index Terms*—Approximate communication, wireless network-on-chip, energy saving**

I. Introduction

In the emerging multi/manycore architectures the interconnection network plays a key role. In [1] it has been shown how the fraction of time spent in communication for several representative exascale parallel applications rapidly increases as the number of processing elements increases. As it can be observed from Figure 1 [2], the time spent in communication is as limited as 10% below 64 processors but it quickly increases over 50% for most of the considered applications as the number of computing cores reaches that in current manycore architectures in the market. If we relate such fraction of time spent in communication with Amdahl's law, it is evident how communication issues might represent a primary bottleneck for applications that target extreme parallelism. Similar observations might be done for what it concerns energy consumption issues. Indeed, the on-chip communication system accounts for a significant fraction of the total energy budget. The increasing gap between gates and wires for which, as technology geometries shrink, transistors becomes more and more efficient in terms of speed and energy whereas wires becomes more and more slow and power hungry [3], has paved the way through the Network-on-Chip (NoC) paradigm.

Recently, emerging communication technologies, including, wireless NoC (WiNoC) [4], have been proposed as a viable solution for the addressing scalability limitations, in terms of communication latency, due to the increasing network diameter [5]. A WiNoC augments the underlying wired NoC with a number of wireless interfaces enabling single-hop on-chip long range communications. Wireless on-chip data communication links with multi GigaHertz bandwidth in millimeter-wave bands have been fabricated and demonstrated [6]. Although WiNoC architectures open a new dimension in the design space allowing new optimization opportunities, communication energy related issues still represent an open issue. This paper focuses on the improvement of the communication energy efficiency in WiNoC architectures.

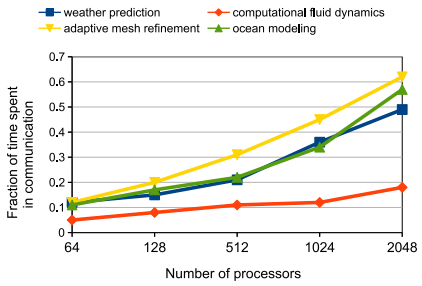

Fig. 1. Fraction of time spent in communication in multi-core systems with different processor count under different representative exascale parallel applications.

An ever more significant number of emerging applications can be classified under the umbrella of recognition, mining and synthesis (RMS) applications [7]. Such applications have been shown to possess a *forgiving nature* [8] towards the imprecision/errors affecting parts of data or computation during their execution. Indeed, errors affecting different data of an application have different impact on the application results. For instance, in a graphical application, imprecision/error affecting a variable storing the RGB components of a pixel will affect the quality of the image but not the functionality of the application. In many cases, the application developer is aware of the sensitivity of the results with respect to certain variables/data structures. Such knowledge can be used to annotate those data structures for which the application is resilient to errors/imprecision/approximation involving them. An approximation aware run-time system can thus trade off results accuracy versus various non-functional metrics, including, performance, energy, reliability, *etc.*. In a NoC based multi/many-core system, the communication system is responsible for the delivery of data exchanged by the nodes of the network (*e.g.*, computational cores, memory controllers). The idea behind this work is to make the communication system aware of the application error tolerance capability

This work was funded in part by Univ. of Catania, DIEEI, under DEDuCE project.

978-1-7281-0171-2/18/$31.00 ©2018 IEEE

with respect to the data involved in current communications. By exploiting such information, the communication channels are run-time configured for trading off communication energy saving and application results degradation. To do so, both the router and wireless interface architectures are augmented with a logic which allows to trading-off communication energy for communication reliability. Specifically, dynamic link voltage swing tuning is used as mechanism for saving the energy consumption of wired links whereas dynamic transmitting power variation is used as mechanism for saving energy consumption of wireless communications. The proposed HW/SW technique is assessed on a set of representative parallel benchmarks showing that up to 30% of communication energy can be saved without of (or with a negligible) impact on the application specific quality metrics.

II. Related Work

The majority of approximate computing techniques proposed in literature explore the trade-off between performance/energy and accuracy with reference to the solely computing subsystem. With the evermore increasing presence of manycore architectures based on the NoC design paradigm, the attention devoted to the on-chip communication system competes with the computational counterpart. The rationale behind the fact that an ever-increasing number of emerging applications possess a "forgiving nature" which make them little sensitive to the imprecision on data and computation can be extended even to communications.

Approximate communication can be considered as a subsection of the larger approximate computing field [2], [9]. Indeed, approximate computing techniques, when applied, generally have an impact on communication metrics. For instance, at software level, general techniques like loop perforation [10] and thread fusion [11] reduce the number of message transmitted. Other common techniques, including lossy data compression [11], precision scaling [12], and data sampling [13] reduce the size of communication messages. Further other techniques, including algorithm selection and parameter adjustment [14] indirectly determine a reduction of both the number and the size of communication messages. It is also worth mentioning approximate computing techniques operating at architectural level which indirectly have a positive impact on communication efficiency metrics. Among them, approximate value prediction [15] and fuzzy memoization [16], allow to reduce the number of messages whereas the use of neural accelerators [17] allows to reduce both the number and the size of messages.

Although the aforementioned techniques, that have been proposed in the context of approximate computing, have an indirect (positive) impact on the communication subsystem, specifically designed techniques aimed at improving the power/performance figures of the communication subsystem have been proposed in literature. Basically, we can classify approximate communication techniques taking into account the specific overhead they try minimizing, namely, synchronization and communication overhead. Techniques designed for tackling the performance and scalability limitation due to synchronization overhead are mainly based on relaxing synchronization. In [18] authors propose to relax a subset of the synchronization points and to exploit implicit noise tolerance of RMS applications. Techniques aimed at reducing the communication overhead can be further divided into compression and load value prediction based techniques. Compression based techniques [11] aim at removing redundant data effectively increasing the density of data to be communicated. Value prediction based techniques [15], based on the observation that many applications exhibit significant data value locality, seek to exploit this predictability for reducing memory traffic.

The aforementioned works are all based on the reduction of the network traffic (by means of dropping, compressing, estimating information) and thus require additional logic to recover, uncompress, and predict communication content. The technique proposed in this paper, instead, does not impact the characteristics of the traffic into the network but selectively configures at run-time the network resources to tailor the reliability requirements of the actual communication flow based on its error tolerance as specified by the application developer. Further, due to the fact the the proposed technique does not impact the network traffic, it is general and can be used in conjunction with several of the above discussed techniques.

III. Description of the Proposed Technique

The technique proposed in this paper to improve the energy efficiency of WiNoC architectures in the context of error tolerant applications [8] is based on the use of a communication system that allows to dynamically tuning the transmission energy based on the error tolerance characteristics of the current communication. Since the reliability level of a communication is a function of the transmission energy, a reduction in energy will result in a reduction from a nominal reliability level, to a lower reliability level.

We consider a programming paradigm similar to [19] in which the software developer specifies, by means of pragma annotation, those variables which if affected by errors do not impact the functionality of the application but just the quality of the results. For instance, a `#pragma resilient(w, rl)` indicates that the access to `w` can be performed with a *reliability level* `rl`. Such reliability level is measured in terms of the experienced bit error rate (BER) to access the data. Indeed, data accesses induce communication messages among processing cores and the memory controller cores. The reliability level is correlated to communication energy consumption, that is, higher the reliability requirement (*i.e.*, lower BER), higher will be the energy consumption for accessing the information.

The next sub-sections provide the details on how the system architecture is augmented to expose a reliability knob and on how such knob is tuned at run-time to provide the energy versus reliability trade-off in accessing the data structures as per developer specifications.

A. Dynamic Link Voltage Swing Circuitry

For the wired network, we propose the tuning of the links voltage swing as mechanism for trading off communication energy and communication reliability. Figure 2 shows the circuitry for implementing the dynamic link voltage swing. The circuitry is instanced for each output port of the router. As it can be observed, the bit-line is preceded by a chain formed by a demultiplexer, two tapered buffers as line drivers and two tristate buffers. The tristate buffers are based on transmission gate logic. With this solution, if the select input is high (low), the full (low) swing path is active and the low (high) swing path is disconnected by the high impedance state introduced by the tristate buffer. The level restorer circuit (similar to the sense amplifier in RAM memories) restores the signal at full swing if the signal on the line is set to low swing, or maintains the original swing if the signal is in full swing mode. By acting on the *Sel* input of the circuit, the link is configured to operate either at reliable mode or unreliable/low energy mode.

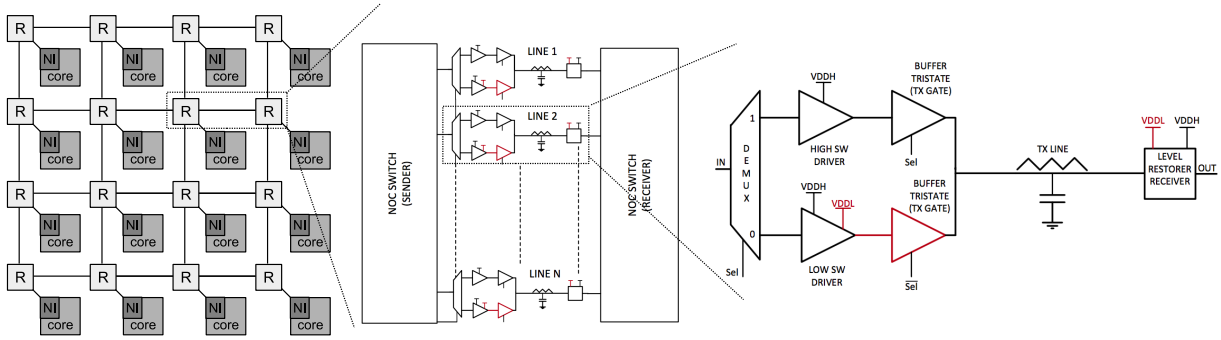

Fig. 2. Architecture of the dynamic link voltage swing circuitry.

TABLE I
HSPICE SIMULATION RESULTS FOR A BIT-LINE OF THE LINK.

	Conventional	Configurable	
	VDDH	VDDH	VDDL
Technology	1.1 V, 10 metal, 45 nm CMOS LVT		
Interconnect (Metal 7)	Width 0.4 μm, Space 0.32 μm, Length 2.8 mm Rwire 225 Ω, Cwire 946 fF		
Supply	1.1 V	1.1 V	0.6 V
Worst case total delay	214 ps	410 ps	
Avg. Energy/Transition	512 fJ	527 fJ	152 fJ
BER	1.3E-17	1.3E-17	3.8E-6

Table I summarizes the characteristics of the the dynamic link voltage swing circuitry. The design has been targeted to work at a clock frequency of 2 GHz (which is the target clock speed of our baseline router). The analysis has been carried out with HSPICE using a 45 nm CMOS LVT library from Nangate [20] which provides 10 metal layers. The parasitics extraction from layout has been made using Cadence Virtuoso. The table compares the conventional link using a single VDDH voltage swing, with the proposed configurable link supporting two voltage swing levels, VDDH and VDDL. We considered a conventional VDDH of 1.1 V and a VDDL of 0.6 V which determine a BER of 1.3E-17 and 3.8E-6, respectively. The worst case total delay of the proposed configurable link increases but it is still below the clock period of the baseline router. The energy per bit of the proposed configurable link increases less than 3% when it works at VDDH. This is due to the overhead introduced by the reconfiguration logic. However, when it works at VDDL the link energy saving is close to 70%.

B. Dynamic Transmitting Power Transceiver

In WiNoCs a Wireless Interface (WI) is used to enable a conventional router to connect the wireless medium. The reference architecture for a WI is shown in the top part Figure 3. It consists of three main parts, namely, antenna, analog, and digital modules. The digital domain includes the channel access token controller (used to implement the radio access control mechanism [21]) and the serializer/deserializer. The analog domain includes the Amplitude-Shift Keying or On-Off Keying (ASK-OOK) which is the most used modulation technique in mm-wave WiNoCs [5], [22], [23]. Although, for a given bit error rate, the ASKOOK modulation requires a higher transmitting power than that required by other modulation techniques (*e.g.*, the quadrature amplitude modulation (QAM) [24]), and has a poor spectral efficiency, its hardware implementation is simple (low area overhead as compared with QAM) and tailored to be applied in the on-chip context.

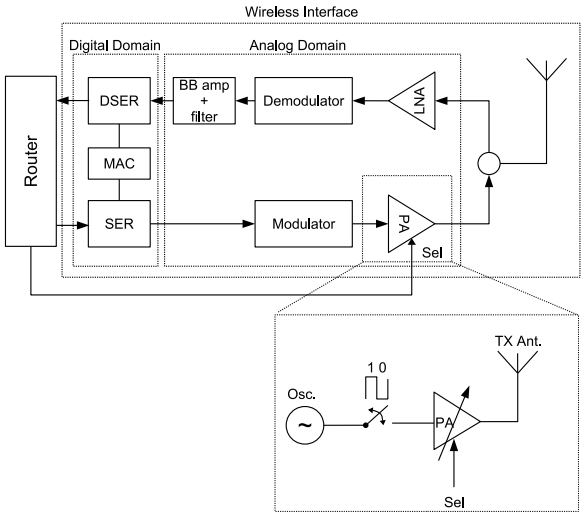

Fig. 3. Architecture of the dynamic transmitting power wireless interface.

The bottom part of Fig. 3 shows the dynamic transmitting power transceiver in which the conventional power amplifier is replaced with an variable power amplifier [25], [26] supporting two different transmitting power level, namely, conventional and reduced. We consider a zigzag antenna modeled and characterized with Ansoft HFSS [27] (High Frequency Structural Simulator) for obtaining the scattering parameters to compute the wireless medium attenuation. Based on this, the minimum transmitting powers for two considered BER levels, have been computed. Specifically, considering a data rate of 16 Gbps and a conventional reliability level corresponding to a BER of 10^{-12} we found that the average energy per bit is 1.47 pJ/bit. For the low-energy and unreliable wireless communication we found that for a BER of 10^{-6} the average energy per bit is 1.0 pJ/bit.

C. Control Mechanism

Both the dynamic link voltage swing and dynamic transmitting power WI have an input, *Sel* in Figs. 2 and 3, used for selecting the expected reliability level for transmitting the current information. As discussed at the beginning of the section, the software developer annotates the source code with pragma directives which inform the compiler that a particular data can be accessed with a certain reliability level. Access

TABLE II
CONFIGURATION PARAMETERS ADOPTED IN THE REFERENCE ARCHITECTURE

Parameter	Value
Number of cores	64
Number of memory controllers	4
Controllers positions	0, 7, 56, 63
L1I (size, bsize, assoc)	64K, 16, 4
L1D (size, bsize, assoc)	64K, 32, 4
L2 (size, bsize, assoc)	512K, 64, 8
Cache Protocol	Private L1/L2 DRAM directory MSI
Flit width	64 bit
Packet Size	64 bytes
Flits/Port Buffer	4

to data not annotated with any pragma directive is considered as reliable, whereas, access to data declared in a pragma is unreliable (with a certain reliability level) in the specified scope of the pragma. In the rest of the paper, we consider a single reliability level. Data involved in pragma declarations will be manipulated through load and store instructions that induce communication messages (among processing cores and memory controller cores) which require less energy effort but a higher BER.

To implement the mechanism, the header flit of the packet reserves one single bit (reliability bit) for marking the packet as either reliable (and energy-hungry) or unreliable (and energy-efficient). Thus, communications induced by load/store access to pragma annotated data simply by properly setting the reliability bit. Routers traversed by the packets use the reliability bit of the header flit to configure the output link or the wireless interface through which the packet will be routed. Indeed, the reliability bit is used as a driver for the *Sel* input port of the dynamic voltage swing link and dynamic transmitting power logic described in the previous subsection. Please note that, even if the packet is marked as unreliable, the header flit will be always transmitted in reliable mode as it contains control information which is not error tolerant.

IV. EXPERIMENTS

A. Simulation Setup and Evaluation Flow

In this section, in order to assess the effectiveness of the proposed approach on real-world scenarios, we take into consideration a set of four applications representative of different RMS workloads, such as financial analysis, computer graphic, data mining, and physics. We assume a 8×8 mesh-based NoC architecture which is simulated by using the Graphite Multicore Simulator [28]. The most relevant architectural parameters are reported in Table II. Each parallel thread of the application is mapped onto a single core, and a traditional shared-memory model is assumed, with the space of DRAM equally split among the 4 memory controllers placed at the corners of the mesh topology [29].

As first step, the application source code is annotated in order to explicitly specify the data structures suitable for approximate communication. Each annotation consists of an address, marking the begin of the annotated memory region, and a size, denoting the extension of the region. The purpose of this annotation process is twofold: first, the knowledge of region boundaries will allow a later detection of memory references candidate for approximation, and second, information about annotated data structure could be used to inject an artificial bit error rate to estimate the effects of approximate communication on the final outputs. Details of such annotations, along with the data input used, are provided in the next subsection when describing the application set. It should be pointed out that, in this work, each annotation has been done manually, by means of a static analysis of source code in order to determine the "role" of each data structure. A conservative approach has been adopted, annotating for approximation only those data structures clearly related to the input workload, following the main assumption of the forgiving nature of RMS application scenario. Nevertheless, such annotation process could be automatized and extended in future developments, further increasing the scope of applicability of approximate communications.

Once the application simulation is completed, the trace of the packet transmissions is obtained. In particular, we modified Graphite Multicore Simulator in order to gather the traces of the packets sent to the memory controllers and use the annotated regions to detect which data reference is candidate for approximate communication. Indeed, the annotation of each data structure contains data about the address and the extension of the region to be annotated. In this way, we are able to use the packet traces to detect which packets would carry data entirely falling in one of the annotated (*i.e.*, approximable) regions.

Finally, the traces are used to feed the Noxim [30] simulator to obtain the power/performance figures, as reported in the the next subsections. Four different NoC scenarios are considered: wired-only communication (NoC), wireless-enabled (WiNoC), and the approximate communication version of both, namely, Approx NoC and Approx WiNoC. With regard to the wireless-enabled NoCs, a natural choice was to assume that Wireless Interfaces are placed in the same locations of memory controllers, that is, in the four mesh corners. Also, a wireless usage policy was chosen so that when the distance (number of hops) to the destination is above a given threshold (five hops, in this case), the packet is redirected towards the nearest node that can provide a single-hop direct connection to the destination.

B. Application Set and Annotated Data Structures

A set of application representative of different RMS workloads has been annotated and simulated to generate the communication traces and annotation information. In particular, we considered the following applications [31]:

- `streamcluster`: The input workload used consists of $8,192$ points, with 64 dimensions per point, 1020 centers, and a maximum of 1,000 intermediate centers. The annotated data structure (*data*) is a multidimensional vector storing the coordinates of the points to be used as inputs. Each annotated region consist of 256 bytes required for storing the 64 dimensions of each point encoded as a floating point value of 4 bytes, for a total of 8192 regions. To evaluate simulation results the percentage of misplaced points (points placed in the wrong cluster) has been taken into account.
- `blackscholes`: A portfolio of 4096 financial derivative options has been used as input. Two data structures have been annotated: *optiondata* a 36 bytes floating point structure, and *prices* (4 bytes floating point), for a total of 147,456 bytes and a 16,384 bytes, respectively. Results are evaluated considering the average percentage change in prices.
- `canneal`: The input workload used consists of 10,000 swaps per temperature step, 2,000 start temperature, and 100,000 netlist elements. The annotation has been performed on the *netlist* element, inside the *getRandomElement()* function that each thread uses to pseudo-randomly pick one new netlist element per iteration, for a total of

160,000 instances of 64 bytes netlist elements. The metric evaluated is the percentage variation of the total routing cost.

- `radiosity`: The input used is the room model description included as standard workload in the SPLASH-2 benchmark suite. The annotated data structures consists of two main elements: *elemvertex_buf.col*, a data structure encoding the three RGB components as 4 bytes floating point values, and *elemvertex_buf.vertex*, a data structure encoding the 3-dimensional coordinates of each vertex of the polygons describing the 3D model of the scene. Each of these two structure occupies 12 bytes, for a total of 65,535 regions and 786,420 annotated bytes size each. Since this application produces an image as output, the evaluation is made by means of the average root mean square applied to corresponding pixels of the precise and approximate output images.

C. Results and Discussion

In Figure 4 are shown the normalized energy consumptions for the four applications being considered, obtained by feeding the Noxim simulator with the traffic traces obtained from Graphite simulations, assuming the same configuration parameters already shown in Table II when describing traces generation. Four different NoC scenarios are considered: wired communication only (NoC), wireless-enabled (WiNoC), and the approximate communication version of them (Approx NoC and Approx WiNoC).

A first observation is that enabling approximation results in energy savings that are not homogeneously distributed. Firstly, there is a different impact that seems to depend on the type (wired/wireless) of network considered: on average, NoC/Approx NoC pairs show a relative gain that is bigger than WiNoC/Approx WiNoC comparisons. In particular, wired NoCs gains range from a 10% of `streamcluster` application up to a 40% saving of `canneal`, while the impact of approximation results lower when comparing WiNoC/Approx WiNoC cases. Of course, this is an indirect consequence of using wireless-enabled networks, that, by reducing the number of hops required, also partially reduce the advantage of using approximate low power communications on longer paths. Nevetheless, fixed a given approximation policy (*e.g.*, enabled or not) and an application, wireless-enable networks still show better behaviors than their NoC counterparts.

Considering again the impact of enabling approximate communications, a second aspect that can be observed in Figure 4 is the not negligible variance of the results across the different applications. Considering , for example, the case of wireless-enable networks (red/green bars), the impact of approximate communication in terms of savings varies, being 7% for `streamcluster`, 22% for `blackscholes`, 11% in `radiosity`, and 30% in `canneal`. Indeed, the only knowledge of the details about the annotated regions suitable for approximate communication, as described in Subsection IV-A, is not sufficient to evaluate how effective will be the actual usage of approximate communication. In particular, assuming a given amount of annotated memory regions, it is the dynamic behavior of the application the dictates "how much" such regions will be referenced in memory requests, thus providing chances of applying approximate communications. Such measurement of the actual applicability of approximate communications on the whole memory references are shown as the rightmost bars (Perc Approx) for each benchmark of Figure 4. These values, ranging from a 20% of `streamcluster` to more than 50% for `canneal`, reflect the fraction of requests sent to the controllers referring to addresses that match annotated the annotated regions. While a detailed analysis of the dynamic behavior of memory reference is not within the scope of this work (interested reader can refer to [32]) an intuitive explanation of such variance can still be provided. In particular, remembering the shared memory/private cache architecture assumed in Table II, a primary impact can be surely attributed to the parallelization model of the application. For example, let us consider a large workload mapped in the main memory, accessed in its entire amount by every node performing the parallel computation. In this case, local caches could never be large enough to avoid main memory references, thus increasing request to the controllers and (possibly) the potential use of approximate communications. As a counterexample, the same large workload mapped in memory could be logically split in several slices, accessed separately and independently by each node: in this case, private local cache could be sufficient to avoid most of memory requests, thus reducing the packets sent to the memory controllers and thus the potentially related approximate communications. Finally, for sake of completeness, Table III summarizes in numerical form the results that have been already visually presented in Figure 4 along with other information, including, the percentage of approximated data structures, and the faction of energy spent for both wired and wireless communications.

Finally, Table IV-C reports, for each application, the error for the specific accuracy metric. As it can be observed, the impact on the accuracy metric is negligible for all the considered applications.

V. Conclusions

In this paper, we have presented an approximate communication technique for improving the energy efficiency of WiNoC architectures. We have proposed the use of dynamic link voltage swing for reducing the energy consumption of NoC links and dynamic transmitting power modulation for reducing the energy consumption of wireless communications. By means of a pragma based annotation of the application code, the load and store induced communications related to error tolerant data are recognized by the underlying communication fabric which selects the appropriate link voltage swing level and transmitting power of links and wireless interfaces involved in that communications. The proposed technique has been assessed on a set of representative benchmarks and the energy saving versus application accuracy trade-off has been discussed. Overall, up to 30% of total communication energy saving has been observed without any appreciable impact on the accuracy metrics.

References

[1] K. Bergman *et al.*, "Exascale computing study: Technology challenges in achieving exascale systems," Defense Advanced Research Projects Agency Information Processing Techniques Office (DARPA IPTO), Tech. Rep 15, Tech. Rep., 2008.

[2] F. Betzel, K. Khatamifard, H. Suresh, D. J. Lilja, J. Sartori, and U. Karpuzcu, "Approximate communication: Techniques for reducing communication bottlenecks in large-scale parallel systems," *ACM Comput. Surv.*, vol. 51, no. 1, pp. 1:1–1:32, Jan. 2018.

[3] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *ACM/IEEE Design Automation Conference*, Las Vegas, Nevada, USA, 2001, pp. 684–689.

[4] K. Chang, S. Deb, A. Ganguly, X. Yu, S. P. Sah, P. P. Pande, B. Belzer, and D. Heo, "Performance evaluation and design trade-offs for wireless network-on-chip architectures," *J. Emerg. Technol. Comput. Syst.*, vol. 8, no. 3, pp. 23:1–23:25, Aug. 2012.

[5] S. Deb, A. Ganguly, P. P. Pande, B. Belzer, and D. Heo, "Wireless NoC as interconnection backbone for multicore chips: Promises and challenges," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 2, no. 2, pp. 228–239, 2012.

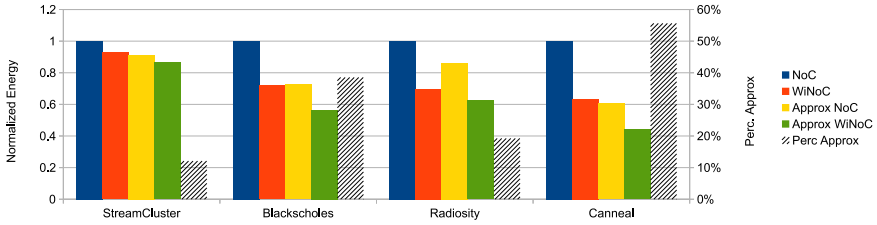

Fig. 4. Normalized communication energy and percentage of approximated data.

TABLE III
DETAILED SIMULATION RESULTS.

Application	Perc. Approx	Energy* NoC	Energy* Approx NoC	Avg Hops count NoC	Avg Hops count WiNoC	Energy* WiNoC Wired	Energy* WiNoC Wireless	Energy* WiNoC Total	Energy* Approx WiNoC Wired	Energy* Approx WiNoC Wireless	Energy* Approx WiNoC Total
streamcluster	12.1%	1.00	0.91	5.3	3.0	0.17	0.76	0.93	0.15	0.71	0.86
blackscholes	38.5%	1.00	0.73	6.9	3.1	0.15	0.57	0.72	0.11	0.45	0.56
radiosity	19.2%	1.00	0.86	6.9	3.0	0.14	0.56	0.70	0.12	0.50	0.62
canneal	55.6%	1.00	0.60	7.0	2.5	0.11	0.52	0.63	0.07	0.37	0.44

*All energy values are normalized with respect to the wired NoC energy consumption.

TABLE IV
PERFORMANCE METRICS.

Application	Metric	Value
streamcluster	% of misplaced points	0.04
blackscholes	Average % change in prices	$2.08 \cdot 10^{-5}$
radiosity	Average root mean square	$3.2 \cdot 10^{-4}$
canneal	% variation of the total routing cost	0.23

[6] J.-J. Lin, H.-T. Wu, Y. Su, L. Gao, A. Sugavanamand, J. E. Brewer, and K. K. O, "Communication using antennas fabricated in silicon integrated circuits," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 8, pp. 1678–1687, 2007.

[7] Y. K. Chen, J. Chhugani, P. Dubey, C. J. Hughes, D. Kim, S. Kumar, V. W. Lee, A. D. Nguyen, and M. Smelyanskiy, "Convergence of recognition, mining, and synthesis workloads and its implications," *Proceedings of the IEEE*, vol. 96, no. 5, pp. 790–807, May 2008.

[8] S. Venkataramani, S. T. Chakradhar, K. Roy, and A. Raghunathan, "Approximate computing and the quest for computing efficiency," in *Proceedings of the 52Nd Annual Design Automation Conference*, 2015, pp. 120:1–120:6.

[9] R. Boyapati, J. Huang, P. Majumder, K. H. Yum, and E. J. Kim, "Approx-noc: A data approximation framework for network-on-chip architectures," *SIGARCH Comput. Archit. News*, vol. 45, no. 2, pp. 666–677, Jun. 2017.

[10] S. Misailovic, S. Sidiroglou, H. Hoffmann, and M. Rinard, "Quality of service profiling," in *ACM/IEEE International Conference on Software Engineering*, 2010, pp. 25–34.

[11] M. Samadi, J. Lee, D. A. Jamshidi, A. Hormati, and S. Mahlke, "Sage: Self-tuning approximation for graphics engines," in *IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Dec 2013, pp. 13–24.

[12] M. A. Anam, P. N. Whatmough, and Y. Andreopoulos, "Precision-energy-throughput scaling of generic matrix multiplication and discrete convolution kernels via linear projections," in *IEEE Symposium on Embedded Systems for Real-time Multimedia*, Oct 2013, pp. 21–30.

[13] I. Goiri, R. Bianchini, S. Nagarakatte, and T. D. Nguyen, "Approx-hadoop: Bringing approximations to mapreduce frameworks," *SIGARCH Comput. Archit. News*, vol. 43, no. 1, pp. 383–397, Mar. 2015.

[14] H. Hoffmann, S. Sidiroglou, M. Carbin, S. Misailovic, A. Agarwal, and M. Rinard, "Dynamic knobs for responsive power-aware computing," *SIGPLAN Not.*, vol. 46, no. 3, pp. 199–212, Mar. 2011.

[15] J. S. Miguel, M. Badr, and N. E. Jerger, "Load value approximation," in *IEEE/ACM International Symposium on Microarchitecture*, Dec 2014, pp. 127–139.

[16] G. Keramidas, C. Kokkala, and I. Stamoulis, "Clumsy value cache: An approximate memoization technique for mobile gpu fragment shaders," in *Workshop on Approximate Computing*, 2015.

[17] B. Grigorian and G. Reinman, "Accelerating divergent applications on simd architectures using neural networks," *ACM Transactions Architecture Code Optimization*, vol. 12, no. 1, pp. 2:1–2:23, Mar. 2015.

[18] S. Misailovic, S. Sidiroglou, and M. C. Rinard, "Dancing with uncertainty," in *ACM Workshop on Relaxing Synchronization for Multicore and Manycore Scalability*, 2012, pp. 51–60.

[19] A. Sampson, W. Dietl, E. Fortuna, D. Gnanapragasam, L. Ceze, and D. Grossman, "Enerj: Approximate data types for safe and general low-power computation," *SIGPLAN Not.*, vol. 46, no. 6, pp. 164–174, Jun. 2011.

[20] "NanGate 45nm open cell library." [Online]. Available: http://www.nangate.com

[21] M. Palesi, M. Collotta, A. Mineo, and V. Catania, "An efficient radio access control mechanism for wireless network-on-chip architectures," *Journal of Low Power Electronics and Applications*, vol. 5, no. 2, pp. 38–56, 2015.

[22] D. DiTomaso, A. Kodi, S. Kaya, and D. Matolak, "iWISE: Inter-router wireless scalable express channels for network-on-chips (nocs) architecture," in *Annual Symposium on High Performance Interconnects*. Santa Clara, California, USA: IEEE Computer Society, 2011, pp. 11–18.

[23] S. Deb, K. Chang, M. Cosic, A. Ganguly, P. P. Pande, D. Heo, and B. Belzer, "Enhancing performance of network-on-chip architectures with millimeter-wave wireless interconnects," in *IEEE International Conference on Application-specific Systems Architectures and Processors*, 2010, pp. 73–80.

[24] L. Couch, *Digital and Analog Communication Systems*. Pearson/Prentice Hall, 2007.

[25] S. Kaushik, M. Agrawal, H. K. Mondal, S. H. Gade, and S. Deb, "Path loss-aware adaptive transmission power control scheme for energy-efficient wireless noc," in *International Midwest Symposium on Circuits and Systems (MWSCAS)*, Aug. 2017, pp. 132–135.

[26] A. Mineo, M. Palesi, G. Ascia, and V. Catania, "Exploiting antenna directivity in wireless noc architectures," *Microprocessors and Microsystems*, vol. 43, pp. 59–66, 2016.

[27] ANSYS. (2014, Jul.) Ansoft HFSS. [Online]. Available: http://www.ansys.com/"

[28] J. E. Miller, H. Kasture, G. Kurian, C. Gruenwald, N. Beckmann, C. Celio, J. Eastep, and A. Agarwal, "Graphite: A distributed parallel simulator for multicores," in *High Performance Computer Architecture (HPCA), 2010 IEEE 16th International Symposium on*. IEEE, 2010, pp. 1–12.

[29] W. Choi, K. Duraisamy, R. Kim, J. Doppa, P. Pande, D. Marculescu, and R. Marculescu, "On-chip communication network for efficient training of deep convolutional networks on heterogeneous manycore systems," *IEEE Transactions on Computers*, vol. 67, no. 5, pp. 672–686, 2018, cited By 0.

[30] V. Catania, A. Mineo, S. Monteleone, M. Palesi, and D. Patti, "Improving energy efficiency in wireless network-on-chip architectures," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 14, no. 1, 2017.

[31] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The splash-2 programs: Characterization and methodological considerations," in *Proceedings of the 22Nd Annual International Symposium on Computer Architecture*, ser. ISCA '95. New York, NY, USA: ACM, 1995, pp. 24–36. [Online]. Available: http://doi.acm.org/10.1145/223982.223990

[32] C. Bienia and K. Li, "Parsec 2.0: A new benchmark suite for chip-multiprocessors," in *Proceedings of the 5th Annual Workshop on Modeling, Benchmarking and Simulation*, June 2009.