Journal of Computing and Security

2020, Volume 7, Number 2 (pp. 95–101)

http://www.jcomsec.org

Signature Codes for Energy-Efficient Data Movement in On-chip Networks

Masoud Dehyadegari [a,*]

[a] *Computer Engineering Department, K. N. Toosi University of Technology, Tehran, Iran.*

ARTICLE INFO.

Article history:
Received: 29 December 2019
Revised: 22 July 2020
Accepted: 3 August 2020
Published Online: 24 September 2020

Keywords:
Interconnection Networks, Network on Chip, Data Movement, Coding

ABSTRACT

On-chip networks provide a scalable infrastructure for moving data among cores in many-core systems. In future technologies, significant amounts of dynamic energy are dissipated for data movement on on-chip links. This paper proposes Sig-NoC, a predictable approach via signature encoding and transition signaling to reduce switching activity on links. Also, we show that link energy dominates in future technologies. Sig-NoC makes the switching activity proportional to the number of 1s per data in the source by using transition signaling. We estimate the energy of each packet at the source of routing in NoC. Therefore, we reduce the number of 1s for high energy packets through signature coding in the source. Sig-NoC mechanism encodes the packets once at the source and decodes them at the destination only; therefore, it has virtually no impact on performance. Simulation results on NAS and Phoenix benchmark suits on 4X4 NoC indicate that Sig-NoC achieves an average of 28% reduction in the overall NoC energy.

1 Introduction

Multi-core and many-core systems are the main solutions for high-performance systems [1]. The cores are connected via an interconnection network to move data on the chip. A variety of networks on chip (NoC) such as wireless NoC, 3D NoC, and optical NoC have been employed to provide scalable communication in many-core systems [2, 3]. Technology projection indicates that data movement is a high portion of energy consumption in future many-core systems [4, 5]. Borkar *et al.* explains the energy consumption in moving data across links in NoC would be more than half the power budget of the processor in future technologies [6]. Hence, designing energy-efficient and high-performance NoCs is crucial to future many-core systems[7].

This paper examines an encoding mechanism to reduce the dynamic energy consumption of packets in NoCs used for a 16-core processor. We observed that the links used for interconnecting NoC routers contribute to significant amounts of energy consumption in modern many-core processors. Therefore, we present a signature-based encoding (Sig-NoC) along with transition signaling to reduce the dynamic link energy in the source. Due to the transition signaling, it is now possible to accurately estimate the amount of switching energy required for transferring each packet between every pair of source and destination nodes of the network.

2 Background and Motivation

On-chip networks consume energy due to static power in routers and links, as well as switching on the capac-

* Corresponding author.
Email address: dehyadegari@kntu.ac.ir (M. Dehyadegari)
https://dx.doi.org/10.22108/jcs.2020.120813.1041
ISSN: 2322-4460

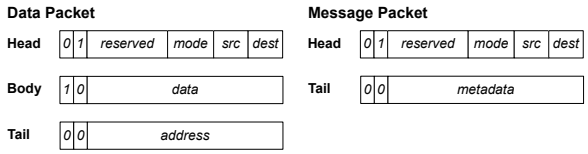

Figure 1. NoC Packets and Flits Used for a Many-Core System.

itive wires used for local and global data movement.

2.1 NoC Traffic in Many-Core Systems

In many-core systems, last level cache is usually shared and distributed among nodes. Main memory comprises off-chip DRAM, which is accessed through multiple DRAM controllers. All processor cores may read or write the shared physical memory through their private L1 cache. Therefore, a coherency protocol is necessary to keep the memory coherent. Directory-based coherence protocols are typically employed by many-core NoC systems. Multiple memory controllers, each of which tightly coupled to one DRAM controller, may be used in a large system to distribute the network traffic throughout the chip. Generally, two types of packets are transferred among the processor nodes and the directory controllers: *data* and *message*. Message packets are used to send a read request to the directory controllers or a status update for the local cache and directory records. The data packet is used to move cache blocks throughout the network. Each packet is composed of multiple floating units (flits), as shown in Figure 1. This paper models packets with 34-bit flits, which are classified as *head*, *body*, and *tail*. Two leftmost bits of each flit are dedicated to identify the flit type. All packets include a head and a tail flits; while, only the data packets include 16 body flits. Every flit is stored in an input buffer before being processed by the router. A virtual channel allocator is used to select a communication channel for every packet on the arrival of its head flit. A crossbar switch is then used to create a path from the buffers to an output port of the router for each flit. Communication links are then used to transfer the flit from one router to the next one. This process is continued until the flit arrives at a destination node.

2.2 Power Consumption in NoC

Dynamic power consumption ($P_{dyn} = \alpha CV^2 f$) is a function of the load capacitance in wires and logic gates (C), supply voltage (V), clock frequency (f), and the switching activity (α). The latter depends on the bit patterns transferred in the network, while all the other parameters are dictated by the underlying technology. Numerous architectural techniques have been proposed in the literature that lower the switching activity of the wires to reduce the data movement

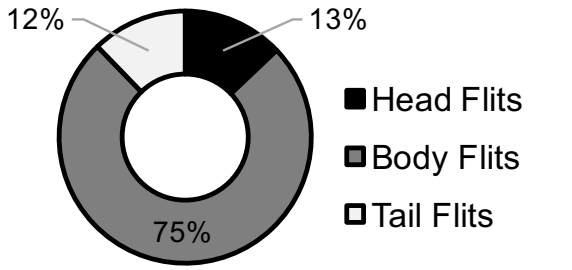

Figure 2. Energy Breakdown Across Various Flit Types in an Example 16-Core Multicore Processor.

energy. Figure 2 shows the relative amounts of energy consumed by different flit types in a 4×4 NoC that implements a directory-based distributed shared memory comprising 16 private L1s, one shared L2, and 4 DRAM channels. [1] The majority of the NoC dynamic energy (75%) is consumed for storing and moving data flits, which is significantly larger than the energy portions consumed by the head and tail flits—i.e., averages of 13% and 12%, respectively.

2.3 Power Optimization Techniques

Cai *et al.* [8] and Xiang *et al.* [9] propose buffer-less routers to consume less energy in routing algorithms and multi-cast traffics. Recent work has proposed to exploit emerging new memory technologies in STT-RAM and drowsy SRAM, to reduce power consumption in network buffers [10]. Muhammad *et al.* proposed to turn off idle virtual channels of the routers to reduce the overall dynamic power [11].

Data encoding has been widely adopted in both off-chip and on-chip communication to reduce power consumption by lowering the switching activity on wires [12]. Bus invert coding was first proposed by Stan *et al.* [13] to lower the dynamic power dissipation in data wires. The basic mechanism transfers either the true or complement of every data block that results in less switching activity, thereby reducing the peak dynamic power. Shen *et al.* propose a configurable NoC with four encoding approaches to provide reliability and power efficiency with minimal impacts on performance [14]. Jafarzadeh *et al.* present an encoding scheme that reduces power dissipation in NoC links by lowering the coupling transition activities on wires [15]. The work requires significant silicon overhead (15%) per router to achieve a 14% power reduction in real-world applications. During every packet transfer between a source and destination routers, data flits are repeatedly encoded and decoded based in the current state of the wires, which result in extra delay per link. In contrast, the proposed Sig-NoC mechanism en-

[1] Detailed explanation of the experimental setup is provided in Section 4.1

codes the packets once at the source and decode them at the destination only; therefore, it has virtually no impact on performance.

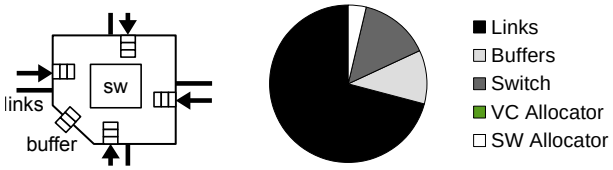

Figure 3. Consumed Energy in Various NoC Components for the Body Flits.

History-based techniques have been explored to exploit similarities between the current and old data blocks to reduce bus activity. For example, frequent value (FV) encoding [16] maintains the frequently transferred values to eliminate the need for transferring the actual data multiple times. VALVE and TUBE improved FV by detecting variable length frequent values [17]. Despite the effectiveness of these techniques in single-hop communication between two on-chip or off-chip nodes, they are ill-suited to networks-on-chip and multi-hop communication where the state of wires may change every cycle. This is mainly due to the significant hardware overheads required at every router to track the history of all possible communication paths. Since data movement in large caches leads to a significant energy consumptions , several researchers proposed transition level signaling in modern computer systems [18][19]. Also, Several approaches are presented to reduce the crosstalk effects by various encoding approaches in physical and register transfer level [20] [21]

3 Sig-NoC

Figure 3 shows a breakdown of the average dynamic energy consumed in the network at a CMOS 22nm technology node. The communication links, crossbar switch, and buffers consume 70%, 18%, and 10% of the NoC dynamic energy, respectively. Therefore, techniques for reducing the switching activity in the network can significantly decrease the dynamic energy. Sig-NoC adopts a signature-based encoding mechanism at the source of every packet transfer to reduce the switching energy in the communication links.

3.1 Transition Signaling for NoC Links

Switching energy is consumed due to charging/discharging capacitive wires during every voltage transition between `0` and `1`. Therefore, the current state of each wire determines if transferring a data bit results in further switching energy consumption. Bus invert coding [13] relies on the Hamming distance (HD) between the current state of wires and data to reduce switchings: if HD is greater than a half fo the bus width, the inverted data is transmitted; otherwise, the original data is sent.

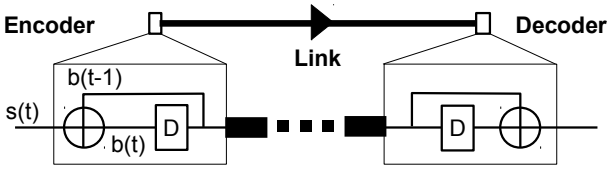

Figure 4. Encoder and Decoder for Transition Signaling.

As the wire state of all the links involved in a packet transfer within the network is not predicable at the source, data needs to be encoded and decoded on every link.

In contrast, transition signaling translates every `1` to a voltage transition between Vdd and Gnd on the wire, thereby making the switching energy predictable and proportional to the number of `1`s per data. Figure 4 shows the necessary circuits used for transition signaling on every NoC link. Every link is equipped with an encoder and a decoder. Consider transferring a data bit $s(t)$ that is modulated to $b(t)$. The value of $b(t)$ is computed as $s(t) \oplus b(t-1)$, which means the current value of the link is XORed with the data bit. On the other end, the decoder produces $s(t)$ by XORing $b(t)$ and $b(t-1)$.

3.2 Signature-based Encoding for Data Blocks

The main goal of signature-based data encoding in Sig-NoC to reduce the number of `1`s in data packets. Since the head flits include the source and destination addresses, data encoding is not applied to the head flits; otherwise, extra decoding overheads would be necessary before every switching routing. In contrast, body and tail flits consist of the cache block address and can be encoded with the signature-based codes. Sig-NoC employs the reserved bits within the head flits to send the signature of each packet, including the body and tail flits. Unlike, the bus invert coding technique that requires additional wires per every communication link between the routers, the proposed encoding technique does not require any additional wires.

Figure 5 shows an illustrative example of the signature-based encoding on four data nibbles (`1001`, `0110`, `1100`, and `0111`) to reduce the total number of transferred `1`s. The transmitter employs four up-counters to compute a 4-bit signature for the data. Every counter is employed to find the majority of logical bits (*i.e.*, `0` and `1`) in a bit position of the data nibbles. Every bit of the signature is determined based on a counter value: if the content of the counter is

greater than 2, the signature bit is set to 1; otherwise, it is set to 0.

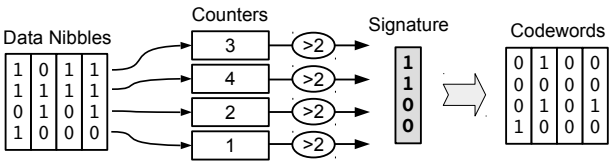

Figure 5. Illustrative Example of Signature-Based Encoding on Four Data Nibbles.

The proposed Sig-NoC in this paper employs 8-bit signatures that are generated for a total of 68 bytes per data packet—i.e., 64 data and 4 address bytes. The signature is once generated at the source of a data packet transfer and XORed with the original data to produce the codewords. The 8-bit signature is then placed in the head flit and sent to the destination node. By XORing the signature and the codewords at the destination node, the original data bytes are recovered.

4 Evaluation

In this section, we describe the experimental setup and the evaluation of Sig-NoC.

4.1 Experimental Setup

To evaluate the effectiveness of the proposed Sig-NoC, we simulate a set of 12 parallel applications from the NAS [22] and Phoenix [23] benchmark suites. We use the ESESC [24] simulator to model a 16-core multicore system, which is heavily modified to model a directory-based distributed shared memory system using a 4×4 mesh network. Detailed area, delay, and energy consumption of various components in the NoC are accurately modeled in CACTI 6.5 [25]. The architectural parameters of the proposed NoC are shown in Table 1, which correspond to an MSI directory-based cache coherency protocol. The directories are distributed at the four corners of the NoC, each of which is responsible for a quarter of the main memory space.

On a read miss in the last level cache, a `ReadShared` packet, including two flits, is sent to the corresponding directory. At the directory controller, if the data is available in other nodes, a `DirectoryRead` or a `DirectoryReadDemote` packet will be sent to one core based on the status data: *shared* or *modify*, respectively. The core will then send a reply to the directory using a `DirectoryReadReply` or a `DirectoryReadDemoteReply` packet. Finally, the directory forwards that cache block to the requesting core via a `DirectoryReply` packet. If the requested data is not in any nodes, the directory reads it from the main memory and forwards it to the requesting core.

Table 1. Architectural Parameters of the Evaluated System.

Parameter	Value
`Processor`	*16 out-of-order cores for* 4×4 *NoC and 64 out-of-order cores for* 8×8 *NoC*
`L1 Instruction Cache`	*32KB, 4-way, 64B blocks*
`L1 Data Cache`	*32KB, 4-way, 64B blocks*
`L2 Cache (shared)`	*16MB, MSI, 8-was, 64B blocks*
`Main Memory`	*tCAS:44, tRCD:44, tRP:44, tRAS:112, FCFS*

The same scenario is true for a write miss with the distinction that a `ReadModified` packet is sent to a directory node. If the status of that cache block is shared and it is on a chip, the directory sends a `DirectoryInvalidate` packet to the owner of data, and will receive the data via a `DirectoryInvalidateReply` packet. Finally, the directory sends a `DirectoryReply` containing the cache block to the requester.

Notice that a `Promotion` packet must be sent to the directory if a core wants to update (write in) a shared cache block. If the core is the only owner of data, a `PromotionReply` will be sent from the directory to the requesting core. If several cores have a copy of that data, a `DirecotyInvalidate` is first sent to all of the owners; then, after receiving `acknowledgment` packets, the directory sends a `PromotionReply` packet to the requesting core.

To evaluate the effect of the network level on Sig-NoC, we conduct some experiments on a 8×8 NoC. The simulation results in Figure 6 show the effectiveness of the Sig-NoC. The energy consumption of the NoC can be reduced by 28% compared with the base.

The area, delay, and power consumption for the Sig-NoC encoders and decoders are shown in the Table 2 based on hardware synthesis at CMOS 22nm. Encoding and decoding are only done in source and destination nodes respectively.

4.2 Simulation Results

Bus invert coding takes a local approach for reducing the dynamic energy of flits per every link; whereas,

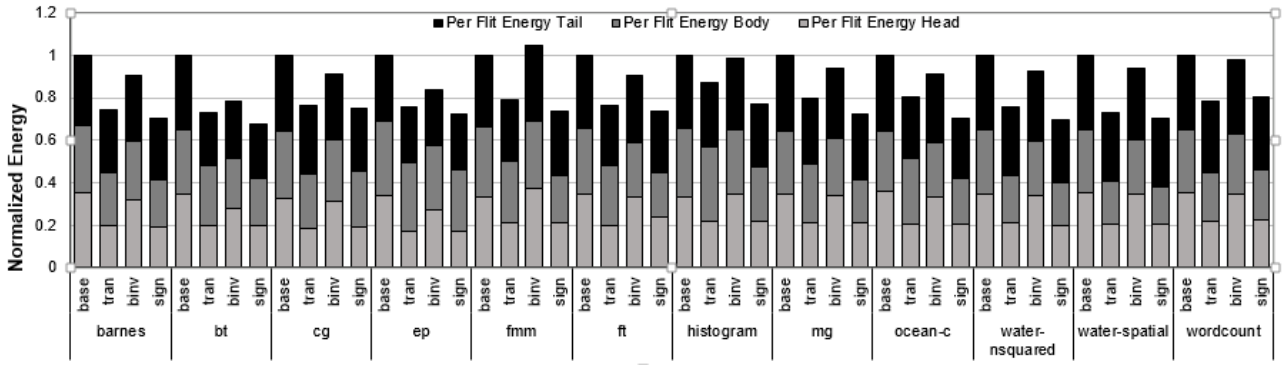

Figure 6. Normalized Per Flit Energy Consumption for the Baselines and Signature Coding Across All the Evaluated Applications In 8×8 NoC.

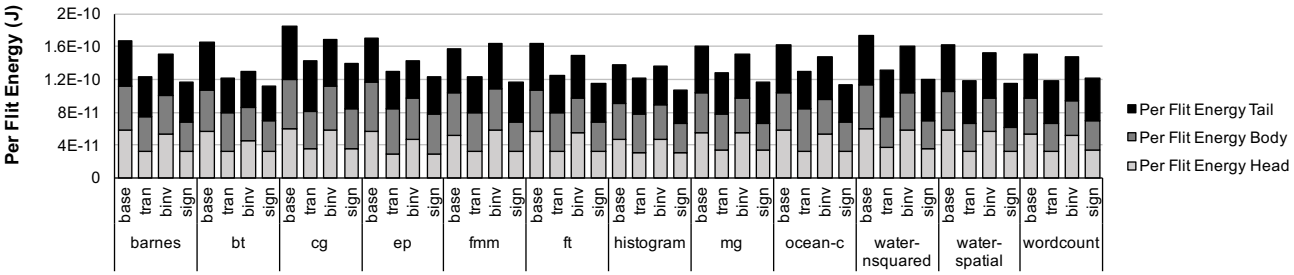

Figure 7. Per Flit Energy Consumption for the Baselines and Signature Coding Across All the Evaluated Applications In 4×4 NoC.

Table 2. Overheads of Sig-NoC Encoder and Decoders.

Encoder	$Area(\mu m^2)$	321
	$Delay(ns)$	0.168
	$Power(mW)$	0.93
Decoder	$Area(\mu m^2)$	85.12
	$Delay(ns)$	0.023
	$Power(mW)$	0.81

the proposed Sig-NoC employs a globally optimized approach relying on transition signaling and accurate energy estimation for each packet. Therefore, it achieves superior energy savings at less encoding/decoding costs and no need for extra wires. Figure 7 shows energy consumption per flit for NAS [22] and Phoenix [23] benchmark suites. `base`, `tran`, `binv`, and `sign` present the binary encoding with level signaling, binary encoding with transition signaling, bus invert coding with level signaling, and the proposed signature encoding, respectively. We observe that `sign`, `tran`, and `binv` reduce the NoC energy consumption as compared with `base` by averages of 28%, 23%, and 8%, respectively. Moreover, we notice that the highest energy reduction up to 33% is for `bt`, which is mainly due to finding the best signatures per each block. The least energy reduction (up to 19%) by Sig-NoC is achieved for `word-count` that processes text data in the ASCII format with low Hamming weights. As the probability of `1`s in input data decreases, the effectiveness of data encoding algorithms reduces.

To evaluate the effect of signature and bus invert encoding, various random data with different probabilities of `1`s are generated. The switching energy for these coding techniques on a single 32-bit data link is shown in Figure 8. The Horizontal axis represents the switching probability of every data bit; the vertical axis shows the relative energy normalized to the conventional binary encoding. As the switching probability increases, the relative energy consumption reduces.

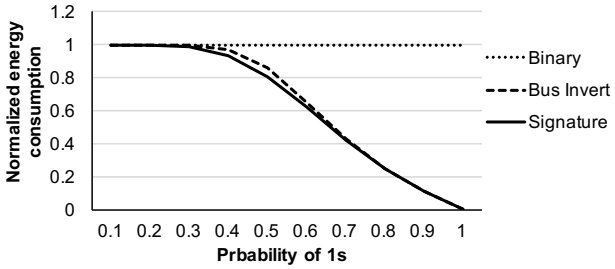

Figure 8. The Effect of Various Data Encoding Techniques on the Switching Energy as Switching Probability Increases.

5 Conclusions

In this paper, we present Sig-NoC to predict the energy consumption of each packet in source. Our main idea is to reduce the switching activity of data movement on links. Because link energy dominates in future technologies. We adopted signature-based encoding to reduce the number of 1s in NoC. We demonstrate the effectiveness of Sig-NoC by running NAS and Phoenix benchmark suits on 4×4 NoC and show that it can largely reduce dynamic energy consumption.

References

[1] T. Rondeau. Electronics Resurgence Initiative Architectures. https://www.darpa.mil/attachments/eri_architectures_DSSoC_Proposers_Day_Final.pdf, 2017. [Online; accessed 18-November-2017].

[2] T. Bjerregaard and S. Mahadevan. A survey of research and practices of network-on-chip. *ACM Computing Surveys (CSUR)*, 38(1), 2006. doi:10.1145/1132952.1132953.

[3] D. Bertozzi and L. Benini. Xpipes : a network-on-chip architecture for gigascale systems-on-chip. *IEEE Circuits and Systems Magazine*, 4(2): 18–31, 2004. doi:10.1109/mcas.2004.1330747. URL https://doi.org/10.1109/mcas.2004.1330747.

[4] X. Chen and N. K. Jha. Reducing wire and energy overheads of the SMART NoC using a setup request network. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 24(10):3013–3026, October 2016. doi:10.1109/tvlsi.2016.2538284. URL https://doi.org/10.1109/tvlsi.2016.2538284.

[5] J. Yin, P. Zhou, S. S. Sapatnekar, and A. Zhai. Energy-efficient time-division multiplexed hybrid-switched NoC for heterogeneous multicore systems. In *2014 IEEE 28th International Parallel and Distributed Processing Symposium*. IEEE, May 2014. doi:10.1109/ipdps.2014.40. URL https://doi.org/10.1109/ipdps.2014.40.

[6] S. Borkar and A. A. Chien. The future of microprocessors. *Communications of the ACM*, 54(5): 67–77, May 2011. doi:10.1145/1941487.1941507. URL https://doi.org/10.1145/1941487.1941507.

[7] W. J. Dally and B. P. Towles. *Principles and practices of interconnection networks*. Elsevier, 2004.

[8] Y. Cai, K. Mai, and O. Mutlu. Comparative evaluation of FPGA and ASIC implementations of bufferless and buffered routing algorithms for on-chip networks. In *Sixteenth International Symposium on Quality Electronic Design*. IEEE, March 2015. doi:10.1109/isqed.2015.7085472. URL https://doi.org/10.1109/isqed.2015.7085472.

[9] X. Xiang, W. Shi, S. Ghose, L. Peng, O. Mutlu, and N. Tzeng. Carpool: a bufferless on-chip network supporting adaptive multicast and hotspot alleviation. In *Proceedings of the International Conference on Supercomputing*. ACM Press, 2017. doi:10.1145/3079079.3079090. URL https://doi.org/10.1145/3079079.3079090.

[10] J. Zhan, J. Ouyang, F. Ge, J. Zhao, and Y. Xie. Hybrid drowsy SRAM and STT-RAM buffer designs for dark-silicon-aware NoC. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 24(10):3041–3054, October 2016. doi:10.1109/tvlsi.2016.2536747. URL https://doi.org/10.1109/tvlsi.2016.2536747.

[11] S. T. Muhammad, M. A. El-Moursy, A. A. El-Moursy, and A. M. Refaat. Optimization for traffic-based virtual channel activation low-power NoC. In *5th International Conference on Energy Aware Computing Systems & Applications*. IEEE, March 2015. doi:10.1109/iceac.2015.7352169. URL https://doi.org/10.1109/iceac.2015.7352169.

[12] G. Ascia, V. Catania, F. Fazzino, and M. Palesi. An encoding scheme to reduce power consumption in networks-on-chip. In *2009 International Conference on Computer Engineering & Systems*. IEEE, December 2009. doi:10.1109/icces.2009.5383319. URL https://doi.org/10.1109/icces.2009.5383319.

[13] M.R. Stan and W.P. Burleson. Bus-invert coding for low-power i/o. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 3(1):49–58, March 1995. doi:10.1109/92.365453. URL https://doi.org/10.1109/92.365453.

[14] J. Shen, P. Hsiung, and C. Huang. Learning-based adaptation to applications and environments in a reconfigurable network-on-chip for reducing crosstalk and dynamic power consumption. *Computers & Electrical Engineering*, 39(2):453–464, February 2013. doi:10.1016/j.compeleceng.2012.09.013. URL https://doi.org/10.1016/j.compeleceng.2012.09.013.

[15] N. Jafarzadeh, M. Palesi, A. Khademzadeh, and A. Afzali-Kusha. Data encoding techniques for reducing energy consumption in network-on-chip. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 22(3):675–685, March 2014. doi:10.1109/tvlsi.2013.2251020. URL https://doi.org/10.1109/tvlsi.2013.2251020.

[16] J. Yang, R. Gupta, and C. Zhang. Frequent value encoding for low power data buses. *ACM*

Transactions on Design Automation of Electronic Systems (TODAES), 9(3):354–384, July 2004. doi:10.1145/1013948.1013953. URL https://doi.org/10.1145/1013948.1013953.

[17] D. C. Suresh, B. Agrawal, J. Yang, and W. A. Najjar. Tunable and energy efficient bus encoding techniques. *IEEE Transactions on Computers*, 58(8):1049–1062, August 2009. doi:10.1109/tc.2009.39. URL https://doi.org/10.1109/tc.2009.39.

[18] P. Behnam and M. N. Bojnordi. STFL: Energy-Efficient Data Movement with Slow Transition Fast Level Signaling. In *Proceedings of the 56th Annual Design Automation Conference 2019*. ACM, June 2019. doi:10.1145/3316781.3317819. URL https://doi.org/10.1145/3316781.3317819.

[19] S. Mittal and S. Nag. A survey of encoding techniques for reducing data-movement energy. *Journal of Systems Architecture*, 97:373–396, August 2019. doi:10.1016/j.sysarc.2018.11.001. URL https://doi.org/10.1016/j.sysarc.2018.11.001.

[20] B. Subramaniam, S. Muthusamy, and G. Gengavel. Crosstalk minimization in network on chip (NoC) links with dual binary weighted code CODEC. *Journal of Ambient Intelligence and Humanized Computing*, March 2020. doi:10.1007/s12652-020-01842-1. URL https://doi.org/10.1007/s12652-020-01842-1.

[21] Z. Shirmohammadi and M. Asadinia. On-fly-TOD: an efficient mechanism for crosstalk fault reduction in WNoC. *The Journal of Supercomputing*, March 2020. doi:10.1007/s11227-020-03259-1. URL https://doi.org/10.1007/s11227-020-03259-1.

[22] D. H. Bailey, R. S. Schreiber, H. D. Simon, V. Venkatakrishnan, S. K. Weeratunga, E. Barszcz, J. T. Barton, D. S. Browning, R. L. Carter, L. Dagum, R. A. Fatoohi, P. O. Frederickson, and T. A. Lasinski. The NAS parallel benchmarks—summary and preliminary results. In *Supercomputing'91: Proceedings of the 1991 ACM/IEEE conference on Supercomputing*. ACM Press, 1991. doi:10.1145/125826.125925. URL https://doi.org/10.1145/125826.125925.

[23] R. M. Yoo, A. Romano, and C. Kozyrakis. Phoenix rebirth: Scalable MapReduce on a large-scale shared-memory system. In *2009 IEEE International Symposium on Workload Characterization (IISWC)*. IEEE, October 2009. doi:10.1109/iiswc.2009.5306783. URL https://doi.org/10.1109/iiswc.2009.5306783.

[24] E. K. Ardestani and J. Renau. ESESC: A fast multicore simulator using time-based sampling. In *2013 IEEE 19th International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, February 2013. doi:10.1109/hpca.2013.6522340. URL https://doi.org/10.1109/hpca.2013.6522340.

[25] N. Muralimanohar, R. Balasubramonian, and N. Jouppi. Optimizing NUCA organizations and wiring alternatives for large caches with CACTI 6.0. In *40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2007)*. IEEE, December 2007. doi:10.1109/micro.2007.33. URL https://doi.org/10.1109/micro.2007.33.

Masoud Dehyadegari received his Ph.D. degree from the University of Tehran, Tehran, IRAN, in 2013 in computer engineering. He is currently an Assistant Professor of school of computer engineering with the K. N. Toosi University of Technology, Tehran, IRAN. Form September 2011 until December 2012, he was a visiting scholar in University of Bologna, Italy. His research interests include Low-power system design, Network-on-chips, and Multi-Processor System-on-chip.