A Novel Asynchronous Network-On-Chip Based on Source Asynchronous Signaling

Venkata Nori, Baudouin Chauviere, Mackenzie Wibbels, and Kenneth S. Stevens
Department of Electrical and Computer Engineering
University of Utah
Salt Lake City, USA

***Abstract*— The communication subsystem of a system-on-chip (SoC) has been an exciting research area for more than three decades. With benefits ranging from increased power efficiency to better area utilization and scalability, networks-on-chip have seen an increased role in SoCs. Multi-core SoC architectures containing discrete functional units like GPUs and application-specific units are becoming increasingly popular, and in this, networks-on-chip play a critical role. Source Asynchronous Signaling (SAS) is a credit-based communication protocol that supports arbitrarily large but finite latency on communication links. The bandwidth and handshake frequency of this protocol are independent of wire latency. The work presented in this paper explores a new asynchronous network-on-chip (NoC) that utilizes SAS. First, we investigate the advantages of using SAS in NoCs. Then, we present a design method for building a deadlock-free NoC employing this protocol. Finally, we evaluate the power and performance numbers of this NoC using accurate simulation models in a torus architecture, and we compare it with a similar asynchronous NoC.**

***Index Terms*— Network-on-Chip, Source Asynchronous Signaling, Deadlock-Freedom**

I. Introduction

Performance has been the driving factor for a significant time in the history of digital circuits. Only in the past couple of decades has the problem of energy become important. The influence of circuits in our daily lives has grown to an extent where one without their presence cannot be imagined. With this increased presence has also come increased demand. Exponential curves can no longer characterize the demand for digital circuits in general. Market shares, manufacturability, and reduced manufacturing cost, paired with growing internet and AI penetration, mainly define future demand. In 2019 it was predicted that the semiconductor industry would see an increasing growth of over 13% from an already existing $440 billion market [1].

Contrary to the growing demand, the performance of the circuits seemed to have plateaued. This is primarily a result of the divergence of scaling from what Robert Denard predicted in 1974 [2]. With dwindling benefits from technology scaling, the burden of providing performance benefits is now on digital designers more than ever. Reduced scaling benefit for single-core processors has led to multi-core designs making more frequent appearances [3]. Generational performance improvements are attributed to the number of cores in a chip. While the benefit of multiple processing units in a similar die space cannot be overlooked, the complexity of communication between them also cannot be overlooked. Distance between the cores increases the latency of communication. Traditional bus-based architectures or even mesh-based systems, while being popular [4], fail to keep up with this demand for performance.

The difference between wire and transistor delays has been increasing even during the early scaling days [5]. The dependency on faster access to memory and even local memory being tagged with cores is an absolute sign that networks-on-chip (NoC) are undeniably the best option for the future of on-chip communication.

While efforts need to be directed at reducing the energy consumption of the circuits, making the circuits more efficient also has a similar effect. A two-pronged approach to making efficient low-powered circuits is perhaps the ideal way. Unlike traditional interconnect, NoC can be seen as an individual module on the die. This helps us design, characterize, and optimize NoCs more efficiently. This work focuses on designing, simulation, and evaluating one such efficient low-powered NoC.

Clockless circuits are power efficient, making them ideal for designing NoCs. Clockless NoCs have seen applications in various fields, including traditional design [6], digital signal processing [7], and neuromorphic architectures [8]. There have been several asynchronous NoC designs [9]–[13]. Designs like [10] take a hybrid approach; the rest are purely asynchronous.

Asynchronous approaches employ handshaking to ensure data validity in the design. While the concept of the request-acknowledge handshake is robust, implementing it in circuits can lead to many unforeseen implications. Deadlock freedom in NoCs is a well-studied topic [14]–[17]. At a fundamental level, the presence of cycles in the design, be it physically or algorithmically, can cause deadlocks. Handshaking implementations create a request-acknowledge cycle by definition, and this is an area that can potentially deadlock. One way to circumvent this is to provide additional hardware in the design that can ensure this cycle never gets activated. The NoC implementation in [18] uses this method by generating a negative-acknowledgment signal at the cost of signal and hardware overhead. This work proposes a novel solution to this problem that does not utilize additional signals for data control. Instead, the handshake itself is decoupled to remove the cyclic dependency.

A new clockless low-powered NoC that utilizes the source asynchronous signaling (SAS) [19] is designed in this work. Using timing as a design parameter for asynchronous NoC systems through the usage of relative timing is demonstrated. In order to simulate real-world usage, the NoC is put through a simulation model that generates bit-accurate traffic patterns. The energy and performance results are gate accurate. An 8×8 toroidal system with 64 networking nodes is designed. The system demonstrates improvements in energy and performance. Certain properties of the SAS protocol employed are verified. The Deadlock properties of the NoC are studied. This work utilizes the fundamental design principles in [18], [20] to prove deadlock-freedom.

II. Source Asynchronous Signaling

Traditional asynchronous handshake protocol ensures proper communication by sending data and its validity information (request) on the communication channel. The response to this is typically an acknowledgment signal that does the namesake on the receipt of data. Modern technology nodes, while giving jumps in performance, have resulted in increased issues in the interconnect department. Every new node has increased the physical parameters of the interconnect, particularly resistance. This increase results in a relative increase in communication delay on the channel employing the interconnect [5].

In an asynchronous handshaking channel, the cycle time is typically the sum of the time taken for the request signal to be sent, the delay of the logic elements present in the channel, and the time it takes for the acknowledgment signal to return. As delay increases, so do the latency numbers, which in turn, detrimentally affect the cycle time of the channel. If elements of the channel get placed far apart, this problem is exacerbated. There exist some solutions to this problem. Those include the use of two-cycle communication and placing pipeline stages together physically [21], [22].

Source asynchronous signaling (SAS) protocol is another elegant solution for this problem [19]. The SAS protocol is independent of wire latency. Fig. 1 shows a traditional communication pipeline compared to a SAS pipeline for the same sizeable end-to-end wire delay.

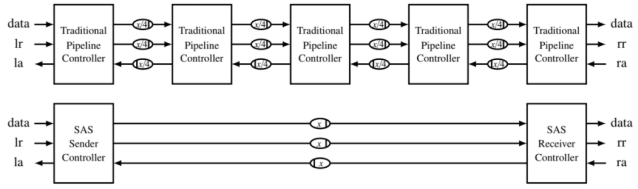


Fig. 1: SAS Pipeline

The request and acknowledgment signals are decoupled in a SAS communication channel. Multiple request or acknowledgment operations can happen without receiving an acknowledgment or a request. This kind of communication cannot be delay insensitive. The design generally comprises SAS receiver and SAS sender modules which ensure the communication's functional accuracy, which is held together by a necessary set of relative timing constraints.

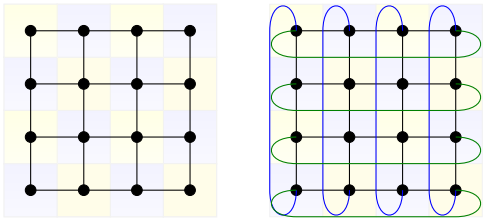
A. Advantages of SAS

Application of SAS communication channel delivers the following advantages to a design [19].

- The bandwidth of the SAS communication channel is wire latency independent. Employing this in an NoC provides the designer with a very high choice in terms of length, material, and placement of links.
- SAS channel provides more energy benefits at longer wire lengths and higher bandwidths.
- Area and latency improvements are achieved because no pipeline repeaters are required, and the data FIFO at the receiving end is smaller than the number of pipelined repeaters required for the same bandwidth.
- SAS is a credit-based protocol.

III. NoC Architecture

A two dimensional 4×4 mesh is shown in Fig. 2(a). Dimension-ordered routing provides a deadlock-free routing mechanism for such mesh topologies.



(a) Mesh (b) Wrapped Torus

Fig. 2: Network Topologies

Torus-based topologies, shown in Fig. 2(b), offer twice the channel bisection, half the channel load, and 24% lower hop count compared to a mesh [23]. However, torus configurations have cyclic dependencies on every edge. Therefore dimension order routing is insufficient to ensure deadlock freedom in a torus. Virtual channels can be employed to ensure deadlock freedom.

A. The Network

The architecture evaluated in this paper contains 64 routers in an 8×8 grid organized in a wrapped torus topology. The network uses dimension-ordered source routing using single flit packets. Two virtual channels are employed to guarantee deadlock freedom. Traffic coming from the core and on turns is placed on virtual channel 0 (VC0). Traffic crossing long wrap lines (the blue and green lines in Fig. 2(b)) is moved from VC0 to virtual channel 1 (VC1).

B. Packet Routing

Two address bits are used at each hop to address the router's four North, East, West, and South (NEWS) ports. The Core port is addressed with the same direction as the incoming port. Table I shows the values of these two bits for transitions. During the hop between routers, a rotating

mechanism is employed to replace the two most significant bits with the next two bits, which act as the address at the next router.

The design is source-routed using dimension-ordered routing, so the routers do not perform any address computation. Thus $2n$ routing bits are required for an $n \times n$ torus.

Address Bits	Addressed Port
00	East
01	North
10	South
11	West

TABLE I: Port addresses

C. Router Design

The entirety of this architecture is asynchronous. It uses Relative Timing (RT) to improve efficiency and meet the timing requirements. The network is built with five port routers at each of the nodes. These routers enable kn-type topologies with flexible usage of routing bits.

A parameterized data channel is used to send and receive data from each of these ports, but the control of this data is done using two virtual channels. These are represented using the 0 and 1 indexed requests and acknowledges in Fig. 3. The data transfer happens on handshaking between the controllers governed by bundled data protocols. Every port of the router contains a switch and a merge module. The modules include finite state machines, mutual exclusion (MUTEX) elements, and SAS pipelines using controllers and registers.

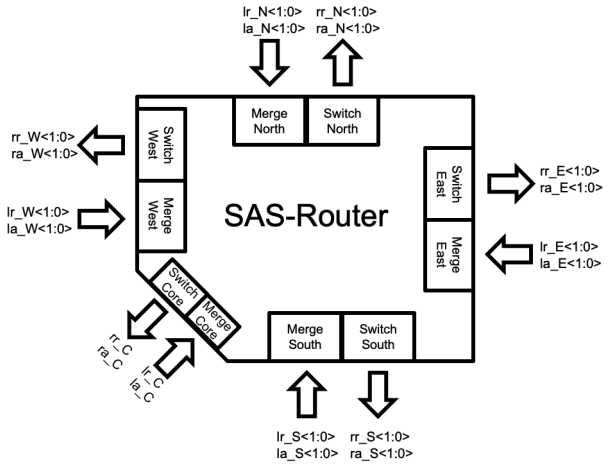


Fig. 3: 5-Port SAS Router

D. Switch Module

The function of the switch module is to accept data on each channel and send it to the destination port. The first two bits on the data stream determine which direction the data is headed. The switch module consists of two identical sub-parts, one per virtual channel. Inside each subpart are address-decoding demultiplexers; the red dotted box indicates this part in Fig. 4. The two subparts ensure no overlap exists in data addressing for each virtual channel.

A three-deep SAS FIFO is employed using the controller modules (LC). Left channel requests ($lr0$) are guaranteed to only occur when the FIFO is not full. Unlike traditional handshaking protocols, the FIFO's left channel ack ($la0$) is not the input channel acknowledgment. The acknowledgment is only returned when data is removed from the right side of the SAS FIFO. The acknowledgment decoupling is a SAS pipeline's fundamental behavior which implements its credit-based protocol. On the right side of the module, request signals are propagated as per the address ($rrx0, \ldots rrx3$). Four acknowledgment signals exist ($rax0, \ldots rax3$) per virtual channel combined using OR logic.

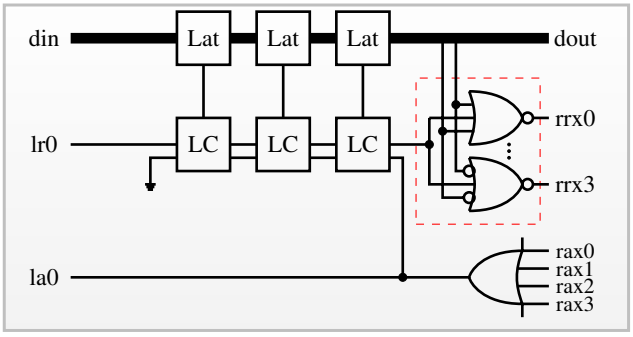


Fig. 4: Switch Module Design

E. Merge Module

The merge module has a two-level functionality, shown in Fig. 5. First-level requests are arbitrated from the switch channels. Seven total requests can happen internally inside the router – two per NEWS port, excluding the receiving port itself, and one from the core port. The Merge module segregates these requests into two groups. Virtual channel zero (VC0) consists of six such requests, and VC1 consists of one request. The arbitrated VC0 and VC1 signals are sent to the virtual merge circuit, which is the second logic level. MUTEX elements inside the Merge6 and Virtual Merge modules ensure that the arbitration process is unbiased.

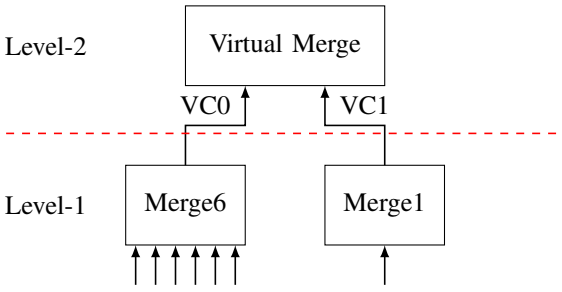


Fig. 5: Merge Module

The virtual merge module arbitrates requests from both virtual channels. The virtual merge module contains a three-deep SAS FIFO per virtual channel, shown in Fig. 6. The FIFOs track credits for each VC based on the buffering depth of the switch module. There is special circuitry forming a feedback loop indicated by the red dotted box in Fig. 6 that

acts as a full/empty indicator to the FIFO. The logic in this indicator blocks the request when the VC's SAS FIFO is full.

The incoming data is multiplexed to the output channel based on the virtual channel selected by the MUTEX logic. The timing in the circuit is designed such that the delay needed for the data to arrive at the switch module on the other end of the network channel is less than the delay for the request signals.

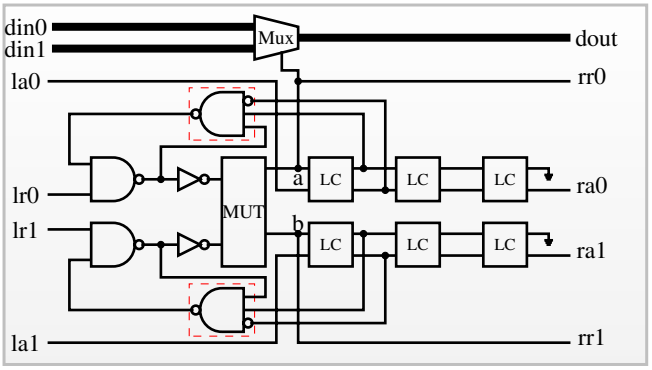


Fig. 6: Virtual Merge Module Design

F. Relative Timing and Hierarchy

Relative Timing refers to the act of using timing as a design parameter in building a circuit. Relative Timing is employed in the form of constraints defined below in Equation 1. *pod* refers to the point of divergence, and *poc* refers to the point of convergence. Assuming there are two points of convergence, a signal can take two distinct paths from the common point of divergence. Equation 1 says that the time taken for the signal to reach poc_0 is always less than the time taken for the signal to reach poc_1, along with some added margin. This can be employed in the design using Synopsys design constraints.

$$pod \mapsto poc_0 + margin \prec poc_1 \tag{1}$$

In SAS router design, the required design constraints and the aforementioned SAS channel constraints are employed at multiple stages. Firstly, in the pre-synthesis stage, they are applied to each individual subpart of a module that makes it up. The modules together compose a router. However, there is a possibility of timing violation at each junction where individually designed modules are met. Each of these links is checked individually, and required constraints are added. The design is then run through a constraint mapper to ensure that no constraints are missed.

After the routers are verified to have no negative slack on Synopsys PrimeTime, an initial simulation is run on the router individually to check the functional correctness. The routers are then individually synthesized using a 65nm library. The routers on the corners and edges of the network are pre-modified to eliminate parts that never get activated, saving design space equivalent to 3 entire routers in a 64-node network. Relative Timing constraints are applied one final time at the junctions where routers meet to ensure no timing violations exist, and the synthesized network is put through a simulator.

G. Deadlock Freedom

A network is deadlock-free if there exist no cycles in the channel dependency graph (CDG) [20]. A toroidal network can be split into two parts, as shown in Fig. 2, consisting of just the mesh section and the long hops. The black-colored nodes and lines form a 2D mesh. The routing used in this work is dimension ordered; at a fundamental level for a 2D network, this translates to XY routing, which by definition makes the mesh deadlock-free [23]. The complication comes from the usage of the single channel long hops, the blue and green colored lines in Fig. 2, which create cyclic dependencies.

Fig. 7 shows the representation of a hypothetical four-router loop. Each router in this loop ($R_{xy}, y = 0$) consist of a fork node (f_{x0}), an arbiter node (a_{x0}) and a three deep SAS buffer (b_{x0c}) for each of the two virtual channels. There is a single shared physical channel between the merge and fork modules.

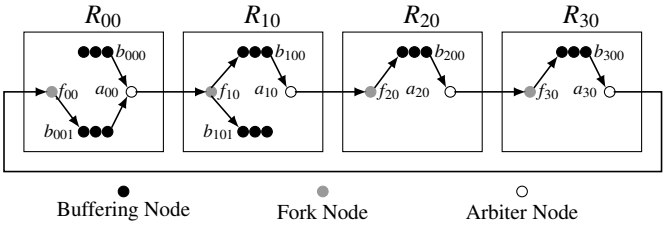


Fig. 7: Physical to Virtual Channel Implementation Mapping

The maximum hop distance in a single dimension of a torus is $n-2$. Flits in this architecture are always introduced into VC0 in any dimension. Therefore, the virtual channels create a cycle-free channel dependency graph (CDG), and there are no VC1 buffers in $R20$ or $R30$. The CDG is $b_{000} \rightarrow b_{100} \rightarrow b_{200} \rightarrow b_{300} \rightarrow b_{001} \rightarrow b_{101}$. Flits arriving at node b_{101} must be consumed by core 101.

A cycle-free CDG is insufficient to prove deadlock freedom in an asynchronous circuit because the handshake signals also create a cycle with the request and acknowledge signals on a communication channel [18]. Deadlock can occur even with a cycle-free CDG if a handshake can be stalled across the physical channel between the arbiter node and fork node. Since the SAS protocol implements credit-based communication, a request will never be sent across the physical channel unless a free buffer is ready to receive the data. Thus the physical channel can never deadlock under the SAS protocol. This allows data to always be moved to free buffers in a SAS channel regardless of the state of the other virtual channel. Thus the toroidal SAS NoC is deadlock-free.

For example, assume a fully loaded network without any bubbles in it. All SAS arbiters a_{x0} are idle, waiting for an

acknowledgment. The processor in node $R10$ will eventually consume data, introducing a bubble into buffer b_{101}. This bubble will propagate through $b_{101} \rightarrow f_{10} \rightarrow a_{00} \rightarrow b_{001}$. Through the same mechanism, this bubble can propagate back to b_{000}. This leads to a theorem that can be proved for networks employing SAS-based networks.

Theorem 1. *A toroidal network built with Source Asynchronous Signaling based routers is deadlock-free if these three conditions are met.*

- *Forward progress property is present in each dimension.*
- *All new traffic either injected into a dimension either through changing dimensions or, if newly injected, will be routed to buffers in virtual channel 0 (b_{xy0}).*
- *Fair arbitration is employed.*

Proof. Section III-G describes how a data flit in the network always propagates in the forward direction. This proves that the network obeys the forward progress property. Also, by definition, SAS protocol does not block a physical channel as long as data is accepted at the SAS receiver end. This ensures that no individual network loop enters a deadlock state. The MUTEX employed in this design internally has fair arbitration because of the absence of internal bias. The XY routing algorithm employed for the mesh part of the network is deadlock-free internally. Therefore, the overall network is deadlock-free. ■

IV. Evaluation

The 64-node SAS design is implemented and compared against a similar design, TECNO, that uses non-blocking arbitration across the physical channel to avoid deadlock [18]. This comparison makes sense because like the work in this paper, TECNO also tackles the problem of deadlock-freedom in asynchronous networks using relative-timed design. The most significant difference between the designs is the protocol used to avoid deadlock (non-blocking handshaking versus the SAS protocol) and buffering depth in the switch modules. The SAS design has three times the buffer depth compared to the non-blocking arbiter design. Work in TECNO evaluates three designs, the diffused wire (RC TECNO) design is most comparable to this design because of the type of wires used in it.

For a fair comparison, the network in this work is synthesized and timing closed in the same technology node, i.e., 65nm Artisan. Wire delays for each network link are pre-determined based on SPICE runs. Simulations are performed in ModelSim using sdf back annotated timing and the associated link delays under various offered loads. Each simulation run calculates latency for each flit and produces a vcd activity file for all nets in the router and network. Power is evaluated in primetime using the time-based vcd activity file from the simulation in conjunction with SPEF parasitics for each node. Power for the network links is calculated as the activity factor for each network wire multiplied by the switching energy for the wire as calculated by SPICE runs. This results in long evaluation run times but provides highly accurate latency, throughput, and power results.

The TECNO and SAS designs have been tested under uniform random test loads ranging from 8 Gb/s to 800 Gb/s bandwidth. The SAS protocol empowers a design to be wire delay independent [19]. Optimal SAS buffer depth allows the design to make all required data transactions at maximal router frequency. The RC TECNO design has a critical path that is dependent on network wire latency and the handshake control logic, which passes through non-blocking arbiters.

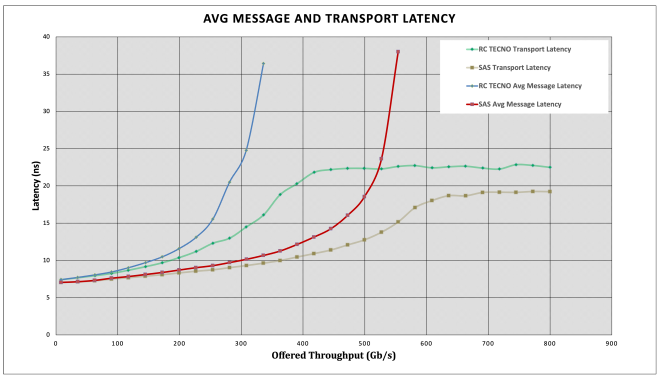


Fig. 8: Latency: Average Message and Transport Latency Comparison with RC TECNO

Fig. 8 compares the SAS design's average message and transport latency against the RC TECNO design. Message latency includes the delay of a message waiting to enter the NoC, whereas transport latency measures the delay from when a message enters the network to when it leaves the network. The SAS-based design has lesser transport and message latency. At 300 Gb/s of offered throughput, RC TECNO is already saturated, but the SAS-based design still easily handles that offered throughput. At this point, the transport latency values are 14ns and 9ns, respectively. This shows a 56% improvement in favor of the SAS-based design. The SAS-based design saturates much later than the RC TECNO design. When transport latency is compared at 20ns, where both designs are in saturation, the RC TECNO design offers a throughput of 280 Gb/s while the SAS-based design offers 505 Gb/s. This is an 80% improvement. At full saturation, the SAS design provides 32% lower transport latency.

The throughput and energy in this design show improvement as well. Fig. 9 graphs delivered versus offered throughput. Below 300 Gb/s, both the RC TECHNO and SAS designs deliver the offered throughput. The RC TECHNO design saturates at approximately 300 Gb/s, whereas the SAS design can deliver the offered load up to approximately 600 Gb/s, providing an improvement of 100% in delivered versus offered throughput. When both designs are fully saturated, the delivered throughput of the SAS and RC TECHNO designs are 609 Gb/s and 340 Gb/s, respectively. At full saturation, SAS delivers a 79% higher throughput.

The average energy per message is shown in Fig. 10 based on offered load. While the SAS and RC TECNO designs follow the same trend over the throughput range, the SAS-based design uses less energy per message. At saturated loads

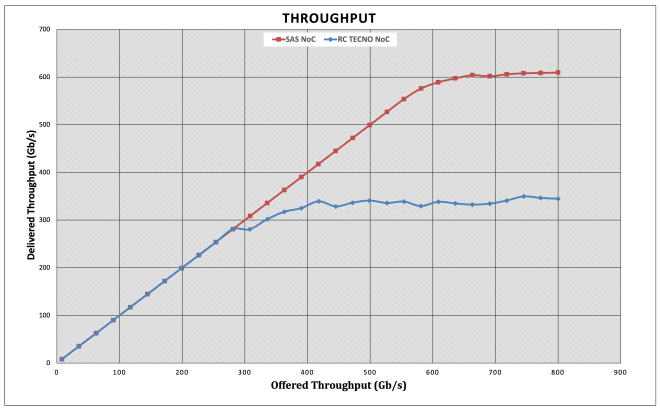


Fig. 9: Throughput Comparison with RC TECNO

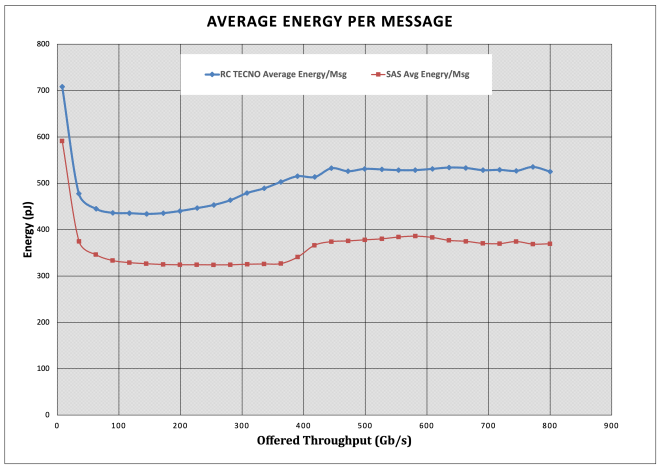


Fig. 10: Energy: Average Energy per Message Comparison with RC TECNO

of about 500 Gb/s, the SAS-based design consumes 380 pJ of energy per message, a 40% improvement over RC TECNO's 532 pJ.

The SAS NoC has a few fundamental advantages over the RC TECNO because of its architecture. Firstly, frequency of operation. The critical path in the TECNO NoC consists of two network wire latencies and 5 complex modules (two MUTEX, one C-Element, one 4-2 Converter, and one non-blocking arbiter). This, in particular, is important because this path decides the operation cycle time. In the SAS NoC, the critical path consists of two simple gates (One AND gate, One NAND gate) and one complex module (MUTEX). The cycle time for this combination is low, thus increasing the frequency of operation. This is responsible directly for the SAS-based design's lower latency and higher throughput. Nextly, the activity factor in any SAS design is lower because, by definition, no handshakes occur unless data is transferred. Compared to the TECNO design that employs negative acknowledgment, the signal is continuously asserted on a blocked channel. As the activity factor increases, the energy consumption increases, which is reflected in Fig. 10.

V. Conclusions

This work presents the design of a novel router using source asynchronous signaling (SAS) protocols. The SAS protocol decouples the request and acknowledge signals in a credit-based protocol that is dependent on network link signal integrity but not latency. The role of relative timing constraints in the design at low and high levels of hierarchy is explained. The very act of decoupling and implementation of robust timing parameters on the design ensure that the cycles that can be formed because of handshaking are avoided. Deadlock freedom using these routers is proven. The SAS design developed here is compared against a deadlock-free design that uses non-blocking handshake protocols. Both designs implement a 64-node torus network-on-chip. The 64-node SAS design is rigorously simulated at various load conditions. Compared to the non-blocking arbiter-based design, the SAS design offers an average of 40% less energy per message and a 100% improvement over saturation comparing delivered versus offered throughput. When both designs are fully saturated, the SAS design produces a 32% lower network latency with 79% higher throughput.

References

[1] E. Sicard and A. Boyer, "Impact of Technological Trends and Electromagnetic Compatibility of Integrated Circuits," in *EMC Compo 2019*, Haining, China, Oct. 2019. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02403882

[2] R. Dennard, F. Gaensslen, H.-N. Yu, V. Rideout, E. Bassous, and A. LeBlanc, "Design of Ion-Implanted MOSFET's With Very Small Physical Dimensions," *IEEE Journal of Solid-State Circuits*, vol. 9, no. 5, pp. 256–268, 1974.

[3] V. Agarwal, M. Hrishikesh, S. Keckler, and D. Burger, "Clock Rate Versus IPC: The End of The Road for Conventional Microarchitectures," 01 2000, pp. 248–259.

[4] S. Bell, B. Edwards, J. Amann, R. Conlin, K. Joyce, V. Leung, J. MacKay, M. Reif, L. Bao, J. Brown, M. Mattina, C.-C. Miao, C. Ramey, D. Wentzlaff, W. Anderson, E. Berger, N. Fairbanks, D. Khan, F. Montenegro, J. Stickney, and J. Zook, "TILE64 - Processor: A 64-Core SoC with Mesh Interconnect," in *2008 IEEE International Solid-State Circuits Conference - Digest of Technical Papers*, 2008, pp. 88–598.

[5] R. Ho, K. Mai, and M. Horowitz, "The Future of Wires," *Proceedings of the IEEE*, vol. 89, no. 4, pp. 490–504, 2001.

[6] G. Chen, M. A. Anders, H. Kaul, S. K. Satpathy, S. K. Mathew, S. K. Hsu, A. Agarwal, R. K. Krishnamurthy, S. Borkar, and V. De, "16.1 A 340mV-to-0.9V 20.2Tb/s Source-Synchronous Hybrid Packet/Circuit-Switched 16×16 Network-on-Chip in 22nm Tri-Gate CMOS," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, 2014, pp. 276–277.

[7] M. Stensgaard, T. Bjerregaard, J. Sparso, and J. Pedersen, "A Simple Clockless Network-on-Chip for a Commercial Audio DSP Chip," in *9th EUROMICRO Conference on Digital System Design (DSD'06)*, 2006, pp. 641–648.

[8] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam, B. Taba, M. Beakes, B. Brezzo, J. B. Kuang, R. Manohar, W. P. Risk, B. Jackson, and D. S. Modha, "TrueNorth: Design and Tool Flow of a 65 mW 1 Million Neuron Programmable Neurosynaptic Chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1537–1557, 2015.

[9] T. Bjerregaard and J. Sparsø, "Implementation of Guaranteed Services in The MANGO Clockless Network-on-Chip," in *IEEE Proceedings on Computers and Digital Techniques*, 2006.

[10] G. Campobello, M. Castano, C. Ciofi, and D. Mangano, "GALS Networks on Chip: A New Solution for Asynchronous Delay-Insensitive Links," *Proceedings of the Design Automation & Test in Europe Conference*, vol. 2, pp. 1–6, 2006.

[11] D. Rostislav, V. Vishnyakov, E. Friedman, and R. Ginosar, "An Asynchronous Router for Multiple Service Levels Networks on Chip," in *11th IEEE International Symposium on Asynchronous Circuits and Systems*, 2005, pp. 44–53.

[12] L. Xin and C.-S. Choy, "An Asynchronous Router with Multicast Support in NoC," ser. CSS '07. USA: ACTA Press, 2007, p. 59–63.

[13] W. Jiang, K. Bhardwaj, G. Lacourba, and S. M. Nowick, "A Lightweight Early Arbitration Method for Low-Latency Asynchronous 2D-mesh NoC's," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2015, pp. 1–6.

[14] A. Hansson, G. Kees, and R. Andrei, "Avoiding Message-Dependent Deadlock in Network-Based Systems on Chip," *VLSI Design*, vol. 2007, 04 2007.

[15] E. V. Castillo, W. J. Chau, G. Miorandi, and D. Bertozzi, "Dynamically Reconfigurable NoC using a deadlock-free flexible routing algorithm with a low hardware implementation cost," in *2015 IEEE 6th Latin American Symposium on Circuits Systems (LASCAS)*, 2015, pp. 1–4.

[16] K. Anjan and T. Pinkston, "An efficient, Fully Adaptive Deadlock Recovery Scheme: DISHA," in *Proceedings 22nd Annual International Symposium on Computer Architecture*, 1995, pp. 201–210.

[17] J. Duato, "A New Theory of Deadlock-Free Adaptive Routing in Wormhole Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 4, no. 12, pp. 1320–1331, 1993.

[18] M. J. Wibbels, S. Das, D. S. Takur, V. Nori, and K. S. Stevens, "A Transmission Line Enabled Deadlock Free Toroidal Network-on-Chip using Asynchronous Handshake Protocols," in *2019 25th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, 2019, pp. 36–45.

[19] S. Das, V. Vij, and K. S. Stevens, "SAS: Source Asynchronous Signaling Protocol for Asynchronous Handshake Communication Free from Wire Delay Overhead," in *2013 IEEE 19th International Symposium on Asynchronous Circuits and Systems*, 2013, pp. 107–114.

[20] W. J. Dally and C. L. Seitz, "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks," *IEEE Transactions on Computers*, vol. C-36, no. 5, pp. 547–553, May 1987.

[21] R. Ho, J. Gainsley, and R. Drost, "Long Wires and Asynchronous Control," in *10th International Symposium on Asynchronous Circuits and Systems, 2004. Proceedings.*, 2004, pp. 240–249.

[22] K. S. Stevens, P. Golani, and P. A. Beerel, "Energy and Performance Models for Synchronous and Asynchronous Communication," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 3, pp. 369–382, 2011.

[23] W. J. Dally and B. P. Towles, *Principles and Practices of Interconnection Networks*. Elsevier, 2004.