

# A Reusable Methodology for High-Performance Interconnects using a Standard-Cell based Asynchronous NoC Router

Chonghui Zhang, Yizhe Hu, Yi Kang

*University of Science and Technology of China, Hefei, China*

chonghui106@mail.ustc.edu.cn, huyz@ustc.edu.cn, ykang@ustc.edu.cn

**Abstract**—As multi-core and Chiplet systems increase in complexity, the bottlenecks of traditional synchronous network-on-chip (NoC) in clocking, power, and timing closure have become a critical barrier to performance scaling. To address this challenge, asynchronous circuits offer a compelling path forward, yet existing designs often struggle with the tradeoff between performance, which typically relies on customization, and EDA flow compatibility, which is often compromised. This paper presents and implements a fully asynchronous, standard-cell-based NoC router paradigm, aimed at translating the theoretical advantages of asynchrony into a practical, industrially viable solution. The paradigm synergizes an innovative Mix-Rail encoding strategy with an EDA-flow-friendly Click-style asynchronous handshake circuit, ensuring full compatibility with mainstream commercial EDA toolchains. This methodology not only achieves high performance but also, through its inherent modularity and generality, facilitates straightforward integration with more advanced NoC technologies. We rigorously validated this design. A test chip fabricated and measured in a 22nm CMOS process achieves a state-of-the-art average latency of 0.63 ns and a high energy efficiency of 0.16 pJ/bit at 0.85V, outperforming published SOTA asynchronous baselines. Furthermore, system-level simulations confirm the performance superiority of the asynchronous approach against a functionally equivalent synchronous baseline. This work provides a robust solution for energy-efficient interconnects in next-generation heterogeneous computing systems.

**Index Terms**—Network-on-chip(NoC), Asynchronous circuits, Multi-core system, GALS

## I. INTRODUCTION

With the end of Dennard Scaling and the slowdown of Moore’s Law, a consensus has emerged that sustaining performance growth in computing now hinges on System-on-Chip (SoC) and heterogeneous integration, particularly through Chiplets and 2.5D/3D packaging technologies [1], [2]. In such highly integrated multi-core and multi-chiplet systems, data communication—rather than computation itself—is rapidly emerging as the primary bottleneck for both system performance and power consumption [3]. As the core interconnect backbone, the design quality of the Network-on-Chip (NoC) directly dictates the entire system’s performance ceiling.

The NoC has been widely adopted as a flexible and efficient interconnect paradigm in multi-core and multi-chip systems [4], [5]. While extensive research has focused on optimizing NoC performance at the system-architecture level (e.g., topology, routing algorithms, and flow control [6]), the underlying physical implementation based on synchronous circuits is increasingly becoming a fundamental impediment to

realizing its full potential. First, constructing a global clock distribution network with low skew and jitter across a large die or multiple chiplets—potentially fabricated on different process nodes—becomes physically challenging, if not infeasible, making timing closure exceptionally difficult. Second, the vast clock tree network is itself a significant source of power consumption, consuming a non-trivial portion of the total power budget [7], [8]. Finally, to manage the multiple clock domains inherent in heterogeneous systems, the industry commonly adopts the Globally Asynchronous, Locally Synchronous (GALS) architecture [9], [10]. In GALS systems, however, the interfaces connecting these local synchronous modules have emerged as a new bottleneck. Conventional solutions rely on complex circuitry for cross-clock domain (CDC) transfers, which not only introduces significant latency and area overhead but also severely degrades overall communication performance.

With their event-driven nature and the absence of a global clock, asynchronous circuits offer a fundamental solution to the aforementioned synchronous bottlenecks [11]. The performance advantages of asynchronous circuits have been demonstrated in several domains [12]–[14], and numerous works have proposed asynchronous NoC designs [15]–[22]. Although the potential of asynchronous design has been academically validated, many of these works rely on custom cells, such as C-elements, which are incompatible with mainstream commercial EDA toolchains. This creates an insurmountable gap between academic prototypes and industrial products. Even in designs attempting to use standard cells, designers are often forced into difficult compromises: they either adopt conservative 4-phase handshake protocols or Quasi-Delay-Insensitive (QDI) designs, sacrificing significant latency and power to ensure robustness, or they opt for more aggressive schemes, only to face challenges in managing timing constraints and glitches, which compromises design reliability. Furthermore, the lack of a standardized and automated design flow for these asynchronous styles imposes a steep learning curve and significantly increases design effort, hindering their wider adoption [23], [24]. Consequently, there is a pressing need for an asynchronous NoC solution that can simultaneously achieve high performance and robustness while being fully compatible with standard EDA design flows.

To address this research gap, this paper proposes a low-latency, energy-efficient, fully asynchronous NoC router designed with 22nm standard cells. Our major contributions are

as follows:

- First, we introduce a reusable, general-purpose asynchronous paradigm based entirely on standard cells. Its modular architecture provides a flexible foundation of core building blocks, readily adaptable for advanced NoC technologies. Critically, this approach is fully compatible with conventional EDA toolchains, paving the way for its broad adoption in advanced computing systems.
- Second, we propose a synergistic co-design of a 2-phase handshake protocol and a Mix-Rail encoding scheme that resolves the classic trade-off between performance, area, and robustness. This combination provides both rapid on-chip signaling and reliable long-range communication without the overhead of traditional, more robust protocols.
- Finally, the design’s superiority is confirmed through comprehensive performance validation. High-level simulations first establish the advantages of our asynchronous approach over an equivalent synchronous baseline. These findings are then conclusively proven by a 22nm chip prototype, with physical measurements showing state-of-the-art results (0.63ns latency, 0.16 pJ/bit) that surpass prior art.

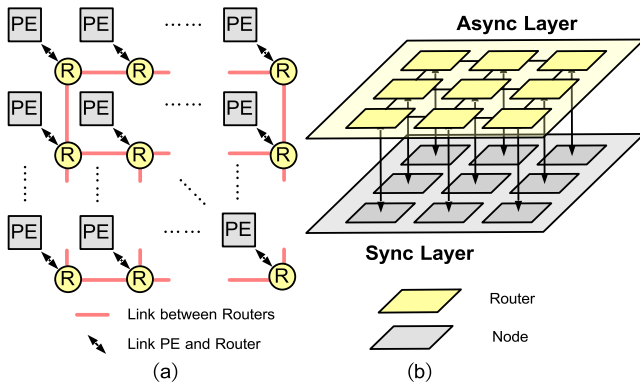


Fig. 1: (a) Traditional NoC with Router-Based Inter-Node Communication. (b) GALS System Comprising Async Layer and Sync Layer, where Inter-Node Traffic Is Predominantly Handled by the Async Layer.

## II. BACKGROUND AND MOTIVATION

### A. NoC and GALS

As on-chip core counts proliferate, traditional shared-bus architectures face severe scalability limitations and bandwidth contention. The NoC paradigm addresses this by applying macroscopic network principles to on-chip communication. An NoC consists of nodes, each connected to an adjacent router. The routers themselves are interconnected in a specific topology, such as a 2D-Mesh (Fig. 1a). Data, organized into packets, is forwarded hop-by-hop through the network according to routing algorithms implemented within the routers.

To overcome complex clocking challenges, the Globally Asynchronous, Locally Synchronous (GALS) paradigm partitions a system into two layers: a Synchronous Layer, where

nodes operate in their own local clock domains, and an Asynchronous Layer that handles inter-node communication (Fig. 1b). This approach eliminates the need for a large, power-hungry global clock network. Consequently, the modularity of GALS simplifies clock design, reduces power consumption, and enables flexible IP reuse and heterogeneous integration across different process technologies.

### B. Asynchronous Circuit Techniques

Asynchronous circuits achieve self-timed data transfer through fundamental techniques, the selection of which constitutes the core design trade-offs.

As shown in Fig. 2a, handshake protocols are the cornerstone of asynchronous communication, using Request (Req) and Acknowledge (Ack) signals to coordinate data exchange. The 4-phase (Return-to-Zero) protocol is simple but incurs significant latency and power overhead due to its four-step process. In contrast, the 2-phase (Non-Return-to-Zero) protocol treats every signal transition as a valid event. This two-step process can halve the handshake latency and boost throughput, albeit at the cost of more complex control logic.

As illustrated in Fig. 2b, data encoding defines how data and handshake signals are coordinated. Single-Rail (Bundled-Data) encoding uses  $N$  wires for  $N$  data bits, accompanied by a control signal for validity. While highly efficient in area and power, it relies on a strict timing assumption that data must arrive before the control signal—a requirement that is challenging to meet across PVT variations. Dual-Rail encoding, on the other hand, allocates two wires per bit, where different states on the wire pair represent the data value and embed timing information. This creates a robust, delay-insensitive channel without timing assumptions [25], but at a significant cost in wiring, area, and power. Both are instances of a broader class of M-of-N codes, which offer a spectrum of trade-offs between robustness and hardware cost.

Pipeline control styles manage the data flow between stages to enhance throughput. Fig. 2c illustrates various control circuits that implement several different pipeline styles. Following early designs based on C-elements [26], latch-based pipelines (e.g., MouseTrap [19]) emerged. While offering area efficiency, their level-sensitive transparency necessitates complex time-borrowing analysis during timing closure. In contrast, flip-flop-based pipelines, such as those in Click-style circuits [27], use standard edge-triggered flip-flops. Data is captured only on the active edge of a locally generated “Fire” signal, which inherently cuts timing paths to eliminate transparency-induced complexity. Critically, this approach aligns strictly with standard synchronous Static Timing Analysis (STA) flows, drastically lowering sign-off and verification complexity. Distinct from these styles that prioritize automation and robustness, others, such as GasP [28], pursue maximum speed through highly customized circuits that demand extensive manual design effort.

### C. Related Work

Existing asynchronous NoC designs illustrate a clear trade-off between practicality and performance. At one extreme, custom-designed NoCs like that of Vivet et al. [22] achieve

exceptional robustness but are impractical for modern systems due to high hardware costs and non-standard, custom-cell-based design flows. Conversely, attempts to address this by shifting to standard-cell designs, such as the latch-based MouseTrap pipeline by Bertozzi et al. [19], introduce new challenges. While EDA-compatible, this approach complicates timing analysis and its inherent structural limitations constrain performance and scalability by hindering the adoption of more advanced NoC technologies.

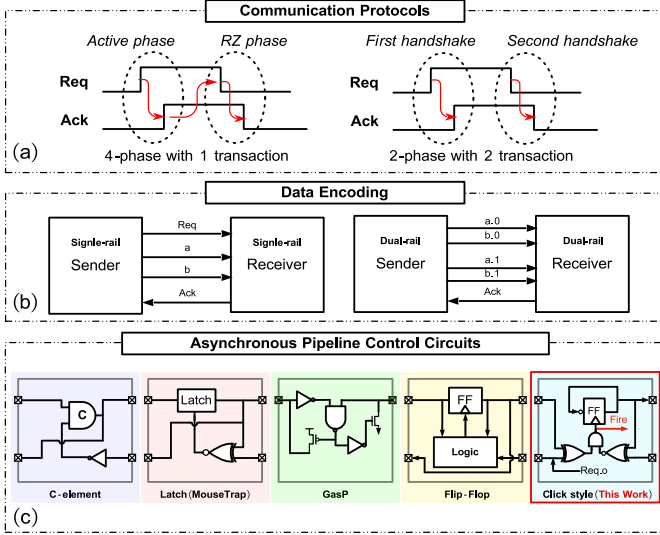


Fig. 2: (a) Two distinct handshake protocols. (b) Single-rail and dual-rail encodings. (c) Various asynchronous pipeline control circuits.

### III. PROPOSED ASYNCHRONOUS NOC

This section details the proposed asynchronous NoC design. We present the overall architecture, the core asynchronous pipeline and its integration into the router’s microarchitecture, the Mix-Rail encoding scheme, and the complete, EDA-compatible design methodology.

#### A. Overall Architecture

This work implements an NoC topology with nine routers on a test chip, as depicted in Fig. 3a. Each router features a five-port, full-duplex architecture, with four ports connecting to adjacent routers and one port connecting to a local Test Unit. The Test Unit is responsible for emulating the behavior of compute or storage nodes by injecting data packets into the network.

The core of the router is an interconnect composed of Input Port Modules (IPMs) and Output Port Modules (OPMs), as shown in Fig. 3b. The IPM of each port is connected to the OPMs of all other ports, forming a crossbar-like structure that ensures a data flit can be routed from any input to any output. Packets are segmented into multiple flits before injection. The lifecycle of a flit within the router is as follows: it first enters an ingress IPM, and after being buffered and routed, it requests a connection to the target OPM. Upon winning arbitration, the

flit is forwarded to the OPM and finally departs the current router for its next destination.

#### B. Asynchronous Pipeline Design

To balance design robustness with EDA flow compatibility, this work employs a flip-flop-based Click-style asynchronous pipeline. Its key advantage lies in the use of standard flip-flops as storage elements, which provides inherent immunity to glitches on input signals. As shown in Fig. 4a, a locally generated “Fire” signal acts as a “virtual clock” to drive the data flip-flops. The generation logic for this Fire signal directly defines the handshake protocol. Our design utilizes the following logic to implement an efficient 2-phase handshake:

$$Fire = (Req.i \oplus Ack.o) \wedge (Req.o \odot Ack.i) \quad (1)$$

This logic ensures that whenever an input request  $Req.i$  transitions (indicating new data arrival) while the output channel is free, the Fire signal generates a pulse to latch the data and update the output request  $Req.o$ . After the downstream acknowledge  $Ack.i$  is received, the Fire signal is reset, preparing for the next transfer. By customizing the Fire generation logic [27] in different modules and strategically inserting delay cells on critical paths, the handshake timing can be flexibly controlled, thereby simplifying front-end design and timing analysis.

#### C. Router Microarchitecture

The IPM is responsible for flit reception, route computation, and dispatch. When a head flit is read from the input FIFO, the Routing Logic computes the destination output port based on its target address (supporting both XY routing or Look-Up Tables). This computation occurs only once per packet, and the result is latched until a tail flit passes. The logic then generates a RouteActive signal to the target OPM to initiate arbitration. The Demultiplexer (DEMUX), shown in Fig. 4b, then precisely directs the handshake request to the corresponding OPM. The DEMUX utilizes dedicated flip-flops for each output channel to store the handshake state and gates the Fire signal with the routing result to activate the correct output handshake. This design completely decouples the input and output handshake domains, thereby avoiding the level-sensitivity and timing-matching challenges common in latch-based designs.

The core functions of the OPM are arbitration and forwarding. Lacking a global clock, this work employs a standard-cell mutex primitive, providing equivalent functionality at the cost of MTBF (Mean Time Between Failures). Each mutex comprises an SR latch for first-come-first-served arbitration and a filter to suppress glitches (Fig. 4c). By interconnecting six mutexes and four AND gates via specific wiring patterns (where Net I defines priority), a scalable four-input arbiter is implemented. Inputs must win three consecutive arbitrations to gain access. Based on the arbitration result, the MERGE module (Fig. 4d) selects the winning IPM’s request to generate a Fire signal and latches the corresponding data into the output Data Buffer. To ensure timely release of the arbiter, a tail-flit detection logic, aided by an extra inverter, uses the falling edge of the Fire signal to trigger a state update. This mechanism

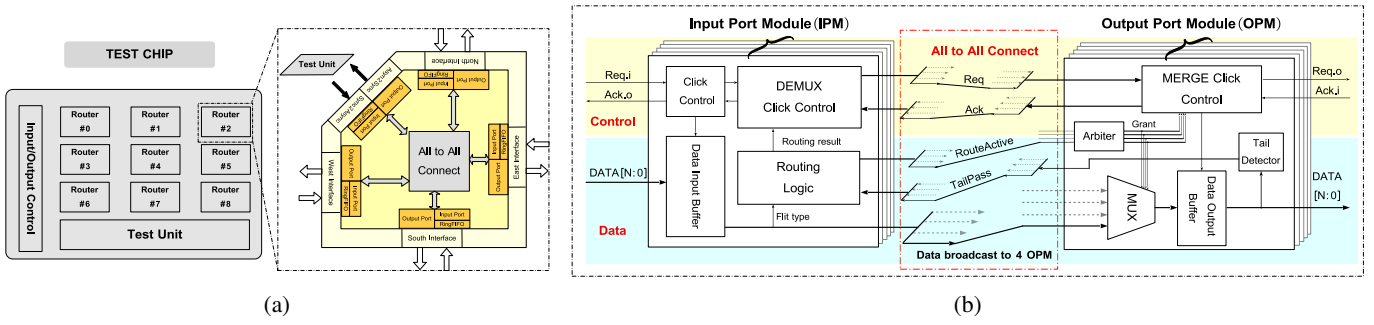


Fig. 3: (a) Test chip overall architecture. (b) Input Port Module (IPM) and Output Port Module (OPM). Every router contains 5 directional ports, each with 1 IPM and 1 OPM.

guarantees that routing resources are freed only after the flit has been fully accepted by the next stage, thereby ensuring data transfer integrity.

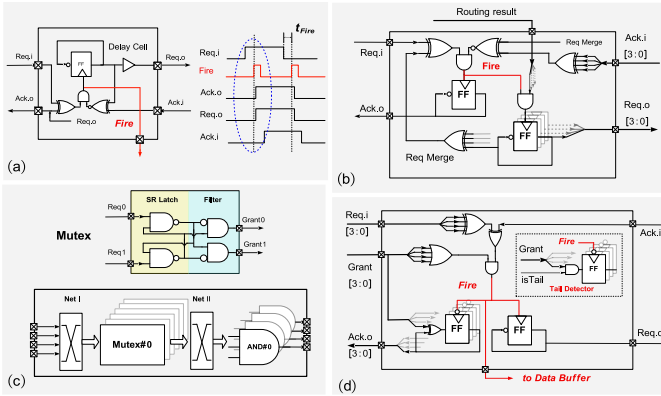


Fig. 4: (a) Click control circuits and waveform. (b) DEMUX module. (c) Mux and 4-way arbiter. (d) MERGE module.

To handle network congestion and decouple the upstream and downstream pipelines, each IPM is equipped with a Ring-FIFO at its input. Its read-write control module, also designed in the Click style, generates independent read and write pointers that circulate among a set of storage registers. This mechanism, in conjunction with a two-way handshake circuit, enables several key features: 1) read and write operations can occur concurrently and asynchronously; 2) an Ack signal is sent to the upstream module only when a free slot is available, enabling effective flow control; and 3) compared to a conventional asynchronous serial FIFO, its circular structure and parallel read-out mechanism avoid unnecessary latency, achieving fast dequeuing. The circuit details of the read-write control module are presented in Fig. 5b.

#### D. Mix-Rail Encoding and Link Design

To achieve an optimal balance among performance, area, and robustness, this work proposes a Mix-Rail encoding strategy. Within the timing-constrained intra-router environment, we employ an efficient Single-Rail encoding to minimize area and power. Conversely, for the long-distance inter-router links, which are susceptible to signal skew and noise, we use a robust

Dual-Rail encoding. For a 1-bit data item  $a$ , dual-rail encoding uses two wires,  $a.0$  and  $a.1$ : a transition on  $a.0$  represents a ‘0’, while a transition on  $a.1$  represents a ‘1’. Data validity is embedded within the signal transitions, eliminating the need for a separate Req signal.

Efficient encoders and decoders are required at the router ports to perform this conversion. While the Single-to-Dual encoding is straightforward and can be implemented with simple XOR/XNOR gates, the Dual-to-Single decoding, shown in Fig. 5a, is more elaborate. This circuit ingeniously integrates data buffering and decoding using three flip-flops and an XOR gate. Concurrently, a Click-style handshake circuit generates the corresponding internal single-rail Req signal upon detecting a transition on the dual-rail lines, which is then fed into the router core. The delay-insensitive property of this dual-rail encoding fundamentally eliminates the timing closure problem between data and control signals, greatly enhancing the robustness of long-distance data transmission and simplifying system-level timing verification.

#### E. Design Methodology and Flow

The core challenge in traditional asynchronous circuit design lies in managing timing constraints. Designs relying on C-elements or latches are difficult to support effectively with standard EDA tools, while the dependence of single-rail encoding on precise delay matching makes it vulnerable to PVT variations. As depicted in the abstract circuit diagram of Fig. 5c, a designer must ensure that the handshake signal arrives at the next-stage register later than the data. This is necessary both to guarantee data stability and to satisfy the flip-flop’s setup time constraint, which requires that the following inequality holds:

$$t_{delay} + t_{Fire} > t_{logic} + t_{setup} \quad (2)$$

Our Click-style design simplifies this by transforming global delay matching into manageable local setup constraints, focusing on combinational delay ( $t_{logic}$ ). Crucially, this timing relationship inherently masks glitches in combinational blocks (e.g., routing logic): by ensuring the Fire generation path exceeds logic settling time, signals are captured only after stability. In most modules, the inherent coupling between control and datapath obviates explicit delay matching, as the

control latency ( $t_{Fire}$ ) typically exceeds the flip-flop setup time ( $t_{setup}$ ) in standard cell libraries.

Furthermore, the Mix-Rail encoding strategy eliminates system-level timing dependencies on long links by employing delay-insensitive dual-rail encoding. Consequently, the timing closure effort is narrowed to a few critical internal paths, such as those within the Ring-FIFO's read-write control and the dual-to-single-rail conversion circuits. On these paths, timing constraints can be easily met by inserting an appropriate amount of delay.

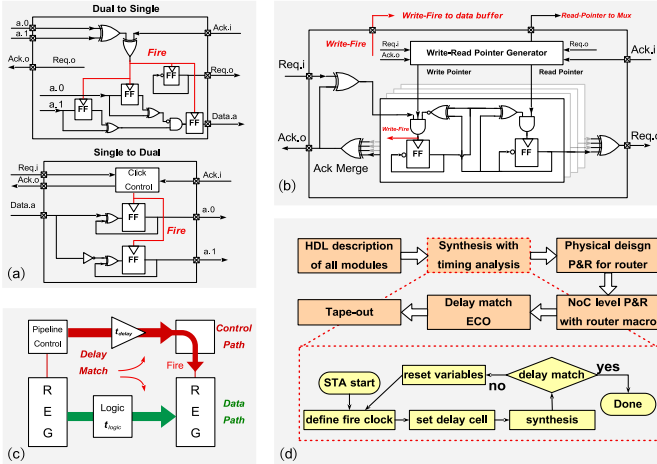


Fig. 5: (a) Mix-Rail Encoding module. (b) Write-Read control module in RING-FIFO. (c) Asynchronous pipeline setup time constraint. (d) Design flow.

Benefiting from this timing paradigm, this work implements a design flow, shown in Fig. 5d, that is highly similar to that of a synchronous circuit. All modules are described in HDL and can be implemented using commercial EDA tools. During logic synthesis, commands such as `set_generated_clock` for the Fire signal enable Static Timing Analysis (STA) tools to automatically check all critical paths. After Place & Route (P&R), STA is performed again with accurate parasitic parameters. A single verified router is then hardened into a Hard Macro, and the final NoC network is integrated according to the 2D-Mesh topology.

The proposed NoC design exhibits good extensibility. Its high degree of modularity and the locality of its asynchronous handshake make it feasible to incorporate other levels of optimization. For instance, Virtual Channels can be supported by instantiating multiple sets of buffers and arbitration logic in the IPMs and OPMs. Similarly, the routing computation unit, as a standalone combinational block, can be easily replaced with more complex logic to support advanced algorithms like Adaptive Routing without affecting the core asynchronous pipeline. This makes the work a solid foundation for application in advanced and complex SoC and Chiplet systems.

#### IV. EVALUATION AND RESULTS

This section evaluates the proposed asynchronous NoC router on two complementary fronts. First, we benchmark its physical performance against several state-of-the-art designs using

measurements from a fabricated test chip. Second, to isolate the fundamental benefits of our asynchronous paradigm, we compare it via simulation against a functionally equivalent synchronous baseline.

##### A. Test Chip Implementation

Following the design flow proposed in Section III-E, we fabricated a test chip in 22nm CMOS technology (Fig. 6). The  $0.37\text{mm} \times 0.12\text{mm}$  core integrates a 9-router NoC with associated test units and I/O controllers. Routers employ a 4-bit flit width due to test-chip area constraints. Notably, as asynchronous control overhead is constant, scaling to practical widths (e.g., 64-bit) would dilute this fixed cost, effectively enhancing amortized energy efficiency.

At a nominal voltage of 0.85V, the test chip exhibits excellent performance. Single-hop latency—defined as the interval from router entry to arrival at the next—measures 0.63 ns for 20-flit packets, covering the entire buffering and arbitration pipeline. Correspondingly, energy efficiency, calculated from average power during continuous transmission, is 0.16 pJ/bit.

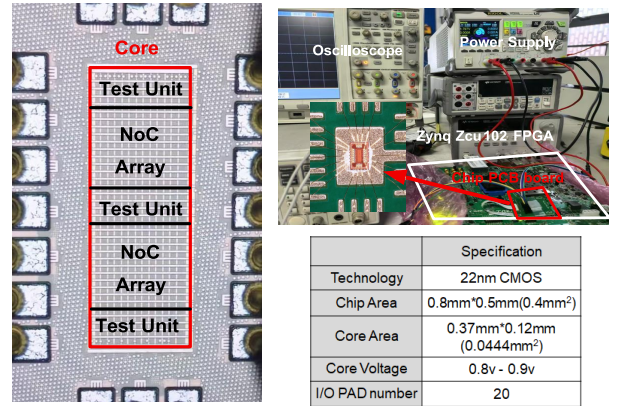


Fig. 6: Test chip die shot, test platform and chip summary table.

For an objective performance assessment, Table I provides a detailed comparison against recently published, silicon-proven, state-of-the-art (SOTA) asynchronous NoC designs. As the table shows, our latency demonstrates a significant advantage over all comparative designs. Regarding energy efficiency, the measured performance is on par with other SOTA works. It should be emphasized that our reported value is a comprehensive, system-level figure that includes the overhead of test units and input/output buffers. This demonstrates that our design maintains highly competitive energy efficiency even under a more realistic system-level assessment. These measured data provide compelling evidence that our proposed design paradigm successfully achieves an optimal balance between design practicality (i.e., full EDA compatibility) and high performance.

To perform an in-depth analysis of the intrinsic merits of the asynchronous paradigm, we designed and implemented a functionally equivalent synchronous NoC to serve as a baseline. To ensure a fair comparison, the synchronous router features a similar architecture and identical FIFO depth, while incorporating virtual channels [29] and the same XY routing algorithm. Both

TABLE I: Comparison of asynchronous NoC designs

	[22]	[19]	[20]	[16]	[17]	[13]	[12]	This work
Technology	65 nm	40 nm	65 nm	65 nm	28 nm	14 nm	28 nm	22 nm
Handshake Protocol	4-phase	2-phase	2-phase	2-phase	2-phase	2-phase	4-phase	2-phase
Data Encoding	4-rail	Single-rail	Single-rail	2-of-4	Dual-rail	Single-rail	Dual-rail	Mix-rail
Pipeline Structure	C-element	MouseTrap (Latch)	Flip-Flop	SAHB	C-element	Click	PCHB WCHB	Click
FIFO depth	0	4	8	0	0	N/A	N/A	4
Cell Type	Std. + custom	Std. + custom (Mutex)	Special lib.	Custom	Std. cell	N/A	N/A	Std. cell
Avg Latency (ns)	1.1	0.98	4.54	5.69	1.8	2.1 <sup>a</sup> 4.1/ 6.5 <sup>b</sup>	5.44Gbps <sup>e</sup>	0.63
Avg Energy (pJ/bit)	0.69	0.12	0.135	0.423	0.156	1.7 <sup>c</sup> (pJ/spike) 3.0/ 4.0 <sup>d</sup> (pJ/spike)	2.3 (pJ/spike)	0.16 <sup>f</sup>

<sup>a</sup> Within-tile spike latency. <sup>b</sup> Latency per tile hop(E-W/N-S). <sup>c</sup> Within-tile spike Energy. <sup>d</sup> Energy per tile hop(E-W/N-S)

<sup>e</sup> NoC Throughput. <sup>f</sup> Include routers, test unit and input/output control.

TABLE II: Comparison of Async and Sync Performance

	Headflit latency (ps)	Avg Flit latency (ps)	Routers Area ( $\mu\text{m}^2$ )	Avg Energy (pJ/flit) 16 flits	Avg Energy (pJ/flit) 2flits
Async	560	465	9902.4	4.43	5.65
Sync	1250	1250	10893.5	5.23	6.21
Overhead	123.2%	168.8%	10.0%	18.1%	9.9%

designs were implemented following the design flow detailed in the previous section. Furthermore, a simulation platform was built using Verilog [30], and network-level simulations were conducted for both designs under various injection rates and different packet lengths using a uniform random traffic pattern.

Table II summarizes the key post-synthesis metrics for a single asynchronous router and its synchronous counterpart, which operates at a clock frequency of 800 MHz. The results show that the asynchronous router has a significant advantage in transmission latency and is 10% smaller than the synchronous circuit. In a full physical implementation of the NoC, where the synchronous version would require a complex clock tree, the area advantage of the asynchronous circuit is expected to be even more pronounced. Furthermore, for both designs, shorter packet sizes lead to more frequent route computations and arbitrations, which increases the average energy consumption per flit.

In terms of network performance, the latency-versus-injection-rate curve in Fig. 7a indicates that the asynchronous NoC exhibits a fundamental latency advantage at low injection rates, as it is not constrained by clock cycle quantization. Both designs achieve similar saturation throughput, with longer packets causing the network to saturate earlier. At low injection rates, the average flit latency of the asynchronous design is reduced by 46% compared to the synchronous baseline.

Regarding power consumption, the power-versus-injection-rate curve in Fig. 7b illustrates that the synchronous router has a higher baseline power due to its constantly switching clock network. At low injection rates, the event-driven nature of the asynchronous NoC yields a distinct power advantage, reducing consumption by an average of 39% relative to its synchronous counterpart. As the injection rate increases, datapath switching

activity becomes the dominant contributor to total power, causing the power consumption of the two designs to converge.

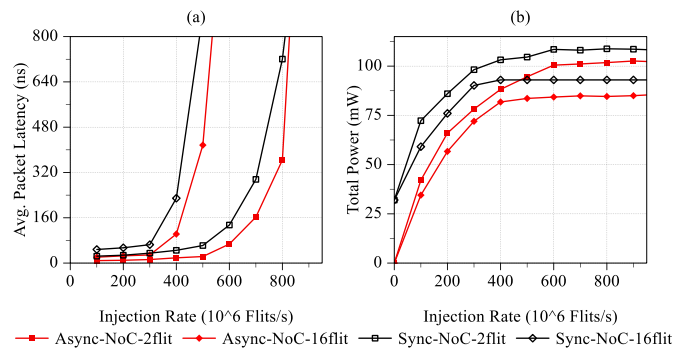


Fig. 7: Simulation results for a  $4 \times 4$  2D-Mesh with uniform random traffic. (a) Average packet latency vs. injection rate. (b) Total power consumption vs. injection rate.

## V. CONCLUSION

This paper presents a practical, standard-cell-based asynchronous NoC methodology that bridges the critical gap between the performance of custom designs and the practicality of EDA compatibility. By synergistically co-designing an EDA-friendly Click-style pipeline with efficient 2-phase and Mix-Rail protocols, our methodology achieves both high performance and robustness. This approach is validated by a 22nm test chip, whose measured performance—a 0.63 ns latency and 0.16 pJ/bit energy efficiency—demonstrates a clear advantage over SOTA designs and our synchronous baseline. These compelling results do more than confirm our solution’s competitiveness; they validate our entire design paradigm as a robust foundation for more advanced NoC technologies. Crucially, by delivering a clear and industrially-aligned workflow, this work provides a tangible path for the widespread adoption of high-performance asynchronous interconnects, enabling the next generation of complex computing systems built upon Chiplets and heterogeneous integration.

## REFERENCES

- [1] A. Smith, E. Chapman, C. Patel, R. Swaminathan, J. Wu, T. Huang, W. Jung, A. Kaganov, H. McIntyre, and R. Mangaser, "11.1 AMD Instinct™ MI300 Series Modular Chiplet Package – HPC and AI Accelerator for Exa-Class Systems," in *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 67, 2024, pp. 490–492.
- [2] M.-S. Lin, T.-C. Huang, C.-C. Tsai, K.-H. Tam, K. C.-H. Hsieh, C.-F. Chen, W.-H. Huang, C.-W. Hu, Y.-C. Chen, S. K. Goel, C.-M. Fu, S. Rusu, C.-C. Li, S.-Y. Yang, M. Wong, S.-C. Yang, and F. Lee, "A 7-nm 4-GHz Arm<sup>1</sup>-Core-Based CoWoS<sup>1</sup> Chiplet Design for High-Performance Computing," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 4, pp. 956–966, 2020.
- [3] G. H. Loh, N. E. Jerger, A. Kannan, and Y. Eckert, "Interconnect-Memory Challenges for Multi-chip, Silicon Interposer Systems," in *Proceedings of the 2015 International Symposium on Memory Systems*, ser. MEMSYS '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 3–10.
- [4] T. Jia, P. Mantovani, M. C. Dos Santos, D. Giri, J. Zuckerman, E. J. Loscalzo, M. Cochet, K. Swaminathan, G. Tombesi, J. J. Zhang, N. Chandramoorthy, J.-D. Wellman, K. Tien, L. Carloni, K. Shepard, D. Brooks, G.-Y. Wei, and P. Bose, "A 12nm Agile-Designed SoC for Swarm-Based Perception with Heterogeneous IP Blocks, a Reconfigurable Memory Hierarchy, and an 800MHz Multi-Plane NoC," in *ESSCIRC 2022- IEEE 48th European Solid State Circuits Conference (ESSCIRC)*, 2022, pp. 269–272.
- [5] N. E. Jerger, A. Kannan, Z. Li, and G. H. Loh, "NoC Architectures for Silicon Interposer Systems: Why Pay for more Wires when you Can Get them (from your interposer) for Free?" in *2014 47th Annual IEEE/ACM International Symposium on Microarchitecture*, 2014, pp. 458–470.
- [6] E. Kakoulli, "Latest Innovations in Intelligent Network-on-Chip Architectures: A Systematic Review," in *2024 17th IEEE/ACM International Workshop on Network on Chip Architectures (NoCArc)*, 2024, pp. 1–6.
- [7] L. Zhang, A. Carpenter, B. Ciftcioglu, A. Garg, M. Huang, and H. Wu, "Injection-Locked Clocking: A Low-Power Clock Distribution Scheme for High-Performance Microprocessors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 16, no. 9, pp. 1251–1256, 2008.
- [8] X. Wang, L. Feng, Y. Wang, T. Xu, Y. Han, and Z. Wang, "Resister: A Resilient Interposer Architecture for Chiplet to Mitigate Timing Side-Channel Attacks," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 30, no. 5, Aug. 2025.
- [9] P. Teehan, M. Greenstreet, and G. Lemieux, "A Survey and Taxonomy of GALS Design Styles," *IEEE Design & Test of Computers*, vol. 24, no. 5, pp. 418–428, 2007.
- [10] E. Painkras, L. A. Plana, J. Garside, S. Temple, F. Galluppi, C. Patterson, D. R. Lester, A. D. Brown, and S. B. Furber, "SpiNNaker: A 1-W 18-Core System-on-Chip for Massively-Parallel Neural Network Simulation," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 8, pp. 1943–1953, 2013.
- [11] M. Renaudin, "Asynchronous circuits and systems : a promising design alternative," *Microelectronic Engineering*, vol. 54, no. 1, pp. 133–149, 2000.
- [12] 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.
- [13] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain, Y. Liao, C.-K. Lin, A. Lines, R. Liu, D. Mathaikutty, S. McCoy, A. Paul, J. Tse, G. Venkataramanan, Y.-H. Weng, A. Wild, Y. Yang, and H. Wang, "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.
- [14] J. Zhang, D. Huo, J. Zhang, C. Qian, Q. Liu, L. Pan, Z. Wang, N. Qiao, K.-T. Tang, and H. Chen, "ANP-I: A 28-nm 1.5-pJ/SOP Asynchronous Spiking Neural Network Processor Enabling Sub-0.1-J/Sample On-Chip Learning for Edge-AI Applications," *IEEE Journal of Solid-State Circuits*, vol. 59, no. 8, pp. 2717–2729, 2024.
- [15] 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.
- [16] W.-G. Ho, K.-S. Chong, K. Z. L. Ne, B.-H. Gwee, and J. S. Chang, "Asynchronous-Logic QDI Quad-Rail Sense-Amplifier Half-Buffer Approach for NoC Router Design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 1, pp. 196–200, 2018.
- [17] Q. Zou, X. Cui, Y. Zhong, Z. Dai, and Y. Kuang, "A fully asynchronous QDI mesh router based on 28nm standard cells," in *2021 IEEE 14th International Conference on ASIC (ASICON)*, 2021, pp. 1–4.
- [18] T. Bjerregaard and J. Sparsø, "Implementation of guaranteed services in the MANGO clockless network-on-chip," *IEE Proceedings - Computers and Digital Techniques*, vol. 153, pp. 217–229, 2006.
- [19] D. Bertozzi, G. Miorandi, A. Ghiribaldi, W. Burleson, G. Sadowski, K. Bhardwaj, W. Jiang, and S. M. Nowick, "Cost-Effective and Flexible Asynchronous Interconnect Technology for GALS Systems," *IEEE Micro*, vol. 41, no. 1, pp. 69–81, 2021.
- [20] M. Gibiluka, M. T. Moreira, F. G. Moraes, and N. L. Vilar Calazans, "BAT-Hermes: A transition-signaling bundled-data NoC router," in *2015 IEEE 6th Latin American Symposium on Circuits & Systems (LASCAS)*, 2015, pp. 1–4.
- [21] 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.
- [22] P. Vivet, Y. Thonnart, R. Lemaire, C. Santos, E. Beigné, C. Bernard, F. Darve, D. Lattard, I. Miro-Panadès, D. Dutoit, F. Clermidy, S. Cheramy, A. Sheibanyrad, F. Pétrot, E. Flamand, J. Michailos, A. Arriordaz, L. Wang, and J. Schloeffel, "A 4 × 4 × 2 Homogeneous Scalable 3D Network-on-Chip Circuit With 326 MFlit/s 0.66 pJ/b Robust and Fault Tolerant Asynchronous 3D Links," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 1, pp. 33–49, 2017.
- [23] J. Cortadella, M. Kishinevsky, A. Kondratyev, L. Lavagno, and A. Yakovlev, "Petrify: A Tool for Manipulating Concurrent Specifications and Synthesis of Asynchronous Controllers," *IEICE Transactions on Information and Systems*, vol. E80-D, pp. 315–325, 03 1997.
- [24] G. Gimenez, A. Cherkaoui, G. Cogniard, and L. Fesquet, "Static Timing Analysis of Asynchronous Bundled-Data Circuits," in *2018 24th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, 2018, pp. 110–118.
- [25] W. Bainbridge, W. Toms, D. Edwards, and S. Furber, "Delay-insensitive, point-to-point interconnect using m-of-n codes," in *Ninth International Symposium on Asynchronous Circuits and Systems, 2003. Proceedings.*, 2003, pp. 132–140.
- [26] D. E. Muller and W. S. Bartky, "A theory of asynchronous circuits," *theory of switching*, 1959.
- [27] A. Peeters, F. te Beest, M. de Wit, and W. Mallon, "Click Elements: An Implementation Style for Data-Driven Compilation," in *2010 IEEE Symposium on Asynchronous Circuits and Systems*, 2010, pp. 3–14.
- [28] I. Sutherland and S. Fairbanks, "GasP: a minimal FIFO control," in *Proceedings Seventh International Symposium on Asynchronous Circuits and Systems. ASYNC 2001*, 2001, pp. 46–53.
- [29] G. Miorandi, A. Ghiribaldi, S. M. Nowick, and D. Bertozzi, "Crossbar replication vs. sharing for virtual channel flow control in asynchronous nocs: A comparative study," in *2014 22nd International Conference on Very Large Scale Integration (VLSI-SoC)*. IEEE, 2014, pp. 1–6.
- [30] D. U. Becker, *Efficient microarchitecture for network-on-chip routers*. Stanford University, 2012.