Journal of Electronic Testing (2024) 40:61–74
https://doi.org/10.1007/s10836-024-06104-y

Design and Verification of an Asynchronous NoC Router Architecture for GALS Systems

M. N. Saranya[1] · Rathnamala Rao[1]

Received: 23 July 2023 / Accepted: 17 January 2024 / Published online: 27 February 2024

Abstract
The increasing multi-core system complexity with technology scaling introduces new constraints and challenges to interconnection network design. Consequently, the research community has a converging trend toward an asynchronous design paradigm for Network-on-Chip (NoC) architecture as a promising solution to these challenges. This paper addresses the design and functional verification aspects of an asynchronous NoC router microarchitecture for a Globally Asynchronous Locally Synchronous (GALS) system. Firstly, the paper introduces a novel mixed-level abstract simulation approach for faster functional verification of the asynchronous architecture using the commercially available Spectre Analog and mixed-signal simulation (AMS) Designer tool. This simulation methodology intends to ensure the feasibility of the design and identify shortcomings, if any, before the subsequent implementation stages of the design. Also, the paper proposes a new baseline asynchronous router built on a domino logic pipeline template with a novel hybrid encoding scheme. The new hybrid encoding scheme facilitates simple architecture with no additional timing constraints. The proposed verification methodology evaluates the baseline asynchronous router's functional verification in Cadence's AMS designer tool. Preliminary simulation results conform to the objectives of the paper. Further, the same verification setup establishes the design validation in subsequent stages of the design implementation.

Keywords Network-on-Chip (NoC) · Asynchronous design · GALS system · Asynchronous pipeline · Domino logic

1 Introduction

The advancement in Very Large Scale Integration (VLSI) circuit technology has led to complex computing systems with thousands of cores integrated on a single chip paving the way for complex System on a Chip (SoC). The backbone of an SoC is the on-chip interconnect that connects all the cores and facilitates inter-core communication on an SoC. Contrary to traditional interconnects, Network-on-Chip (NoC) provides parallel communication and demonstrates higher scalability with the minimal area/power overheads [7]. Thus, NoC replaces the conventional interconnect in today's complex multi-core SoC. As a result, the practical implementation of the NoC design paradigm is continuously evolving to meet the performance constraints set by the SoC [5].

The conventional method for digital system design, including SoC and NoC, is synchronous logic, where systems communicate using a master clock. This greatly simplifies the design verification and implementation of the systems. However, with the increasing scale of integration, synchronous design approaches are facing numerous challenges: process variability, chip power and thermal challenges, electromagnetic interference, and scalability issues [3]. So, a significant academic and industrial research effort is spent exploring design techniques to overcome the growing limitations of synchronous design with technology scaling [5].

Asynchronous or clockless design has emerged as an alternative design paradigm to address the challenges of a fully synchronous system. Asynchronous designs are naturally low power; they consume dynamic power only when

Responsible Editor: B. Ghavami

✉ M. N. Saranya
mnsaranya.207ec005@nitk.edu.in

Rathnamala Rao
malarathna@nitk.edu.in

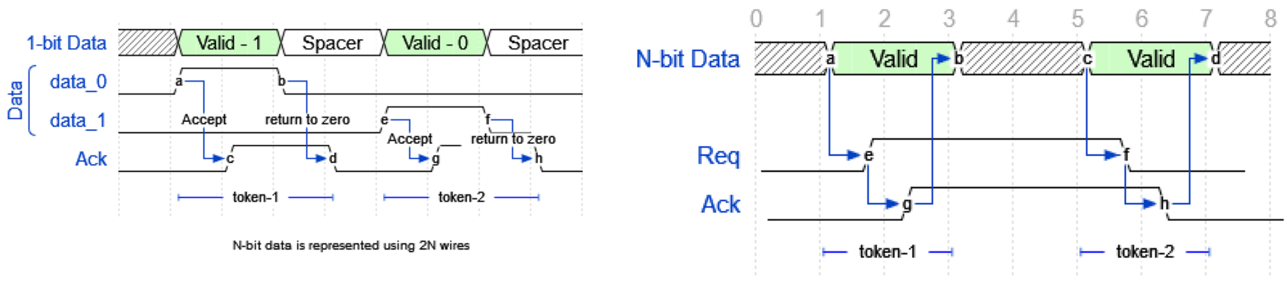
[1] Department of Electronics & Communication Engineering, National Institute of Technology Karnataka, Surathkal, Mangaluru 575025, Karnataka, India

active [24]. Typically, digital asynchronous logic computation falls under the approach of discrete value in continuous time. These systems do not require complex clock distribution and are robust to process and environmental variations. Although the asynchronous design approach exhibits many attractive attributes, this style is not widely accepted due to its design challenges. Hazard-free design requirements, non-availability of automated computer-aided design (CAD) tools, and difficulty in verifying and testing asynchronous designs are the few challenges that make asynchronous designs less popular.

In asynchronous design, the communication is through means of local handshaking channels. The choice of data encoding schemes and communication protocols define the handshaking channels or interface, affecting the circuit implementation characteristics [33]. The most widely used approach of communication is the 4-Phase or Return to Zero (RZ) protocol which involves two round-trip communications for a single token transaction, with Delay-Insensitive (DI) encoding schemes such as Dual-rail (DR) as illustrated in the Fig. 1a consuming more time and energy. However, it is extensively used for its simpler hardware design and suits well for dynamic logic [24]. Another well-known approach visualized in Fig. 1b is the 2-Phase, or Non-Return to Zero (NRZ) protocol (also known as transition signalling), with a single-rail (SR), bundled data encoding scheme. It leads to faster circuits at the cost of implementing complexity to respond to these signal transitions. Further, it imparts a one-sided timing constraint to ensure the correct operation; thus, it is less timing robust but exhibits high coding efficiency. More recently, a hybrid design approach for asynchronous systems exploiting the merits of dual-rail and single-rail bundled data encoding has been explored [35, 43]. These hybrid methods have proved to be achieving area-efficient and ultralow-power asynchronous circuits. It has been utilized to realize multiplier, adder circuits and FIR filter designs [36, 37]. Despite of their promising performance, these hybrid methods have not yet been used for complex modules.

An alternative design paradigm to bridge the gap between fully synchronous and asynchronous systems is a globally asynchronous and locally synchronous (GALS) system [1]. In a GALS system, the synchronous cores are connected using asynchronous communication networks. This approach eliminates the need for a global clock for the chip and facilitates functional block reuse. In recent years, research has been trending toward combining asynchronous design style with NoC [6, 8, 15, 21]. Inherently, the NoC approach separates the communication infrastructure and timing from the IP cores, making it the perfect choice for an asynchronous design style. Further asynchronous NoC is a promising solution for the problems faced by synchronous design, namely clock distribution, clock switching power, and ease of integration of heterogeneous processing core modules. The asynchronous NoC can support on-chip communication in fully asynchronous and GALS systems. However, the complexity involved in the circuitry and power of an asynchronous NoC plays a pivotal role in a GALS architecture.

Asynchronous systems can operate indefinitely, adding non-determinism and making it challenging to slow down or pause. Moreover, asynchronous designs often employ level-sensitive latches and combinational feedback for storage. These factors add complexity to testing. Also, traditional fault-tolerant methods employed in synchronous NoC are ineffective with asynchronous NoC [32]. Specific fault-tolerant strategies must be implemented exclusively for asynchronous interconnects [14, 38, 39]. In addition, Design for Testability (DfT) strategies face a challenge due to timing constraints between control and data paths, requiring concurrent testing for circuit accuracy in a bundled-data design [18]. At the same time, special fault-tolerance techniques are needed for delay-insensitive circuits [32, 39]. So, unlike synchronous systems, there is no standardized or generic methodology for DfT in asynchronous NoCs. Thus, the issue of testing in asynchronous NoCs remains a focal point in literature [24, 25, 32]. These metrics further exert the need for systematic solid CAD tool support in designing and



(a)

(b)

Fig. 1 Asynchronous communication channels **a** Dual-rail channel using 4-phase protocol **b** A bundled data channel using 2-phase protocol [3]

verifying asynchronous NoC routers, especially using commercially available CAD tools in the implementation flow.

The goal of this paper focuses on acknowledging the requirements of practical design and verification flow for asynchronous NoC switches using CAD tools. The paper is subdivided into five sections. Section II briefs and discusses the previous research and existing practice on designing and verifying asynchronous NoCs. Section III elaborates on the proposed design and modeling approach of the baseline asynchronous NoC architecture. Section IV summarizes the Spectre AMS Designer Suite simulation setup, functional verification flow and the verification results of the baseline asynchronous router. Finally, section V concludes the work.

2 Previous Work

A major predicament in asynchronous design is the unavailability of tools to support the design and verification. It is essential to validate the design with a high-level abstract model simulation before implementation, as it aids in identifying any bottlenecks in the design at an early stage. The primary concern with modeling is that the level of modeling must conform to system performance, and the models must be reusable to validate the hardware implementation. Many special asynchronous design frameworks have been proposed to aid in the design and simulation of asynchronous systems, such as *communicating sequential processes* (CSP) [20], *LARD*, *TANGRAM* [41], *Balsa* [2], System C based design [9] and SystemVerilog [45]. The primary concern with the usage of these languages is that they exhibit limited capability and support. Furthermore, most languages do not support circuit modeling at the transistor and logic levels. Languages like *TANGRAM* have evolved into *HASTE*, and CSP into *CAST*, which can be translated to Verilog for simulation and integration purposes.

Nevertheless, the exclusive nature of these languages restricts their usage by the designers. In addition, circuits generated by CSP and TANGRAM-based method demonstrate poor performance [3, 19]. Some frameworks can synthesize the design, while others need to be augmented with other tools, such as TAST, for implementation. To be more specific, some of these tools are custom in-house tools that are specialized for particular design styles and are not generally available to other researchers and designers [25].

Very High-Speed IC hardware description language (VHDL) and Verilog have been the most popular among hardware designers, so several approaches to using existing Hardware description language (HDL) for asynchronous systems have also been reported in literature [10, 26, 29, 30]. The behavioural simulation of asynchronous circuits using commercial tools can be tedious since there is no clock signal for a timing reference to the circuit. [29] proposed a method to convert CSP to VHDL for simulations. A System-C modeling of asynchronous circuits was also proposed [9], but timing modeling is not considered in this approach. However, achieving fine-grain concurrency in these approaches is difficult. Further, VHDL based approach requires additional wrappers to simulate propagation delays. Verilog-based modeling with its programming language interface (PLI) [30] and Verilog macros to model CSP without PLI known as VerilogCSP [10] are other methods proposed to exploit Verilog language. VerilogCSP and CSP are difficult to debug, and there is no concrete discussion about the implementation of designs specified based on these approaches.

Restricting the scope to asynchronous NoCs, many prominent research works have been carried out in the last decade. Some of the fully asynchronous NoC routers described in the literature are [6, 8, 15, 17, 21, 42]. In literature, only SystemC/ Transaction Level Modeling (TLM) [4] and *Balsa Framework* based [23] functional verification for asynchronous NoC architecture have been previously proposed. Most of the other asynchronous NoCs use existing HDLs [15–17, 27, 28, 31] or opt for a full-custom design approach [21] or an ad-hoc design style leading to the hard-macro designs. HDL-based modeling best suits the SR data path design-based routers, as it is similar to synchronous logic. However, VHDL requires careful annotations to meet its timing constraints and wrappers to simulate in a synchronous tool. So, in comparison to VHDL, the use of Verilog supports easy integration of asynchronous design in the existing design flow with its counterparts. If DR encoding is used, the modeling in HDLs is even more tedious and time-consuming, as reported in [23]. The DR encoding logic path may have domino logic that requires only a full-custom approach and hard-macro implementation.

The existing asynchronous NoC switch research has concentrated on a 4-phase protocol with delay insensitive or bundled data encoding. There are also a few reports of a 2-phase bundled data encoding scheme being used for asynchronous NoC [16]. However, designing complex circuitry such as NoC router switch architecture using DR and SR encoding schemes in the logic design approach remains unexplored. To the best of our knowledge, *ASPIN* NoC [31] is the only architecture that uses both bundled and dual-rail data encoding: the latter is used only for the intra-switch links leading to wiring overhead within the switch and increased switching activity while SR encoding is used for inter-switch links. The compatibility encoding logic between DR and SR at the boundaries and the completion detector for the DR data path lead to increased wiring and switching activity. As a result, area overhead and power consumption of the switch increase. Another critically acclaimed recent work is *TaBuLA* [8], which implements an asynchronous NoC switch in SR encoding with the 2-Phase protocol.

This communication scheme imposes stringent timing constraints on the switch design's data path and control logic to ensure functional correctness. To support these constraints, matched delay lines are added, which need to be carefully designed. Thus, the different approaches reported in the literature target various design facets of an asynchronous NoC switch with inherent performance trade-offs.

From this perspective, the paper addresses the design and verifying challenge of an asynchronous NoC switch with hybrid logic. To be more precise, the two contributions of this paper are as follows:

- A mixed circuit level modeling and verification flow is proposed in this work to address the shortcomings of using HDLs for asynchronous switches with mainstream CAD tools.
- A novel baseline asynchronous NoC router built on a high throughput pipeline template with a new hybrid encoding scheme.

The baseline asynchronous NoC router micro-architecture design is evaluated in the proposed flow by only restricting attention to its functional verification aspect. In this flow, the domino logic asynchronous pipeline stages at the transistor level are explicitly instantiated with the high-level behavioural/structural Verilog descriptions of the router specification and simulated on the same platform.

3 Proposed NoC Asynchronous Router Architecture

The baseline asynchronous NoC router under evaluation employs the proposed hybrid encoding scheme. The pipeline stages of the design use an asynchronous high-capacity hybrid logic pipeline template with a post-detection (PD-Hybrid) [35]. The other design aspects are chosen based on the widely adopted generic configurations of an NoC switch, as explained next.

3.1 Switch Architecture

Figure 2 shows the top-level switch instance. A general-purpose 2D-mesh topology is considered in this work to simplify the physical implementation of the NoC. The router presented is a 5-ported switch design. Each router switch module has four bidirectional ports connected to the network and one bidirectional port connected to the processing core through the network interface. The packets from the processing core module are injected into or ejected from the network through the port connected to the network interface. Each port has an Input Computation Module (ICM) and Output Computation Module (OCM). Point-to-point inter-switch links are considered for the baseline switch model, and buffers are not included at this primitive stage of the design. In Fig. 2, only the routing paths from the processing core input port to all other ports in the switch are indicated. Similar routing paths exist from each of the ICMs to the OCMs corresponding to other ports. A variable length packet organization is considered in this work to improve the performance and flexibility of the NoC. The basic switch architecture is stemmed from *ASPIN* [31] and baseline architecture of *TaBuLA* [8]. The main architectural difference between the proposed architecture and the base NoCs are in the encoding scheme, control logic blocks and pipeline stages, as explained in Sect. 3.2.

The proposed approach has two salient features. First, the proposed method facilitates the design of an asynchronous NoC router with low asynchronous control overheads. It does so by choice of hybrid encoding, where one bit is

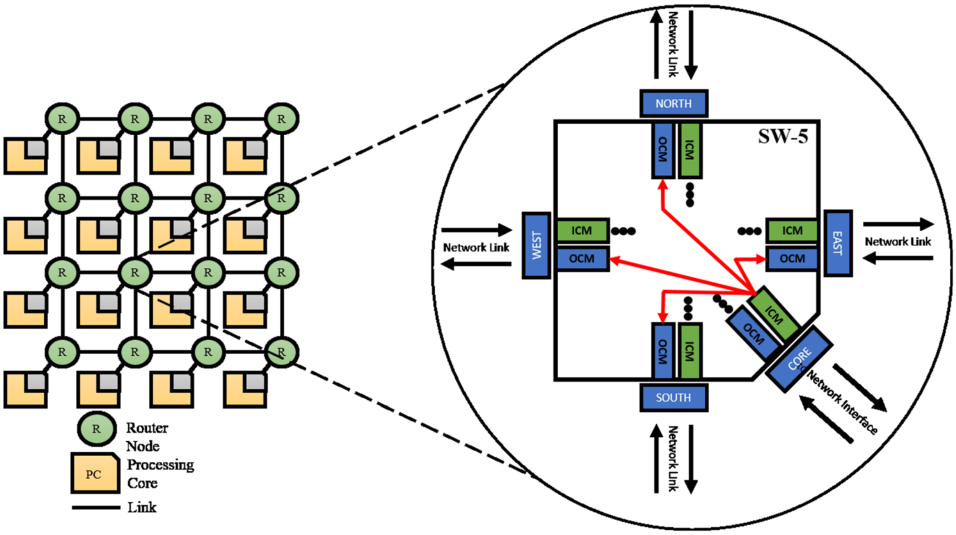


Fig. 2 A visual depiction of a switch instance in a 2D-mesh 4x4 Network-on-Chip (NoC) architecture [22]

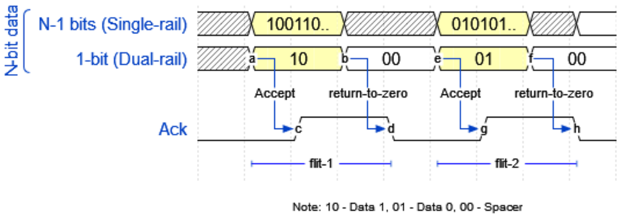


Fig. 3 An example of the proposed hybrid encoding scheme for the data channel

encoded in dual rail while the rest are encoded in SR. Figure 3 shows an example of a data channel using the proposed hybrid encoding scheme. In dual-rail encoding, a pair of wires is used to carry a single data bit. The complement data representation signifies both the value of data and the request signal. So, the dual-rail design uses a completion detector at the receiver to detect the presence of valid data on all data paths and produce acknowledgement to the sender. A complete DR encoding is very robust but results in wiring overhead and requires a tree of completion detector. At the same time, only SR bundled data encoding is simple to implement but requires a separate request wire and may impose many timing constraints on the design. So, the proposed encoding style aims to combine the robustness of DR and the simplicity of SR encoding schemes without adding overhead to the router's area footprint

The proposed encoding approach is very similar to single-rail bundled data; instead of a separate *Req* strobe, only one-bit data is encoded in DR encoding. It is done to ensure that the worst-case computation of SR is handled and that the receiver can explicitly identify the valid flit. Thus, in the proposed baseline router, only a single completion detector is required, and there is no necessity for a separate request line. In addition, there are no critical timing constraints to be considered for the correct synchronization of the switch, eliminating additional matching delay lines in the circuitry. Hence, this encoding style reduces the complexity of the switch circuitry while at the same time reaping the benefits of both DR and SR encoding schemes. The other salient feature of the proposed architecture is the simple and straightforward implementation of the data path and control logic.

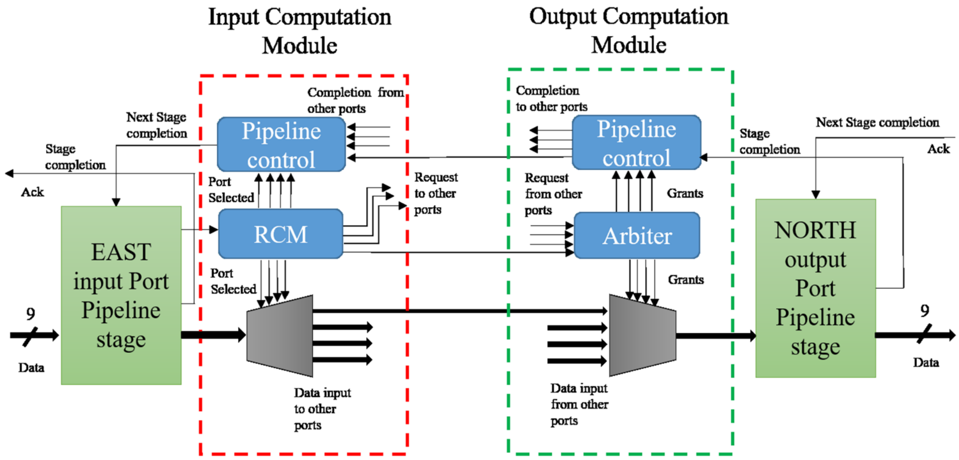
3.2 Data Flow Structure of the Proposed Switch Design

Each ICM has a set of independent data routing channels connected to all other OCM in the switch. Therefore, the router can support maximum concurrent data flow attributing to parallelism. The intra-switch data path is 9-bits wide with 1-bit in DR encoding, while the rest of the bits are in SR encoding. In this encoding scheme, instead of a separate request line, one bit of data is encoded in DR. So, the inter-switch data channel comprises the 9-bits data and acknowledgement signal. A conceptual schematic of the data path and control logic between a pair of ports, namely *EAST* and *NORTH*in the proposed asynchronous NoC switch, is shown in Fig. 4. The proposed switch architecture models wormhole routing [22], and packets are processed at the level of flits of width 8-bits. The routing channel is decided and reserved by processing head flit in the wormhole routing method. The body flits, and the tail flit of the packet follows the head flit through the reserved channel. The proposed switch model is parameterizable for flit size and the number of ports. XY algorithmic distributed routing [22] is chosen for the baseline switch model due to its simplicity.

3.2.1 Working

The arrival of the input flit from the upstream link to the pipeline stage at the ICM starts the processing. There is no separate request line. The ICM directs the packet to the destined OCM. The route decision is made in the *Route Computation Module* (RCM) present in ICM that realizes the

Fig. 4 An abstract representation of the data path and control logic between an input and an output port within the proposed router network



dimension order XY algorithm to identify the output port. It uses the 2-bit flit type field inserted in each flit to identify the flit type. Then the destination address extracted from the head flit is compared to the current switch address to identify the output port to which the packet has to be directed. The data is directed to the OCM at the output port selected. Concurrently, the RCM places a request to the *Arbiter* at the output port selected. The data path is established by granting the request between the input and an output port for the packet traversal. After receiving the input data, the output port sends the acknowledgement, i.e., *stage completion* signal of the pipeline stage at the output port. The handshake between the input and output port is governed by the *pipeline control block* (PCB). The PCB at the OCM ensures that the acknowledgement signal is properly channelled to the ICM transmitting the packet. Similarly, the PCB at the ICM accepts the acknowledgement signal only from the intended output port.

As long as the request from ICM remains high and the input port is granted the output port, all the flits from the packet are transmitted through the path delegated. At the end of the packet, when a tail flit is received at the OCM, the path established for packet traversal is terminated. The grant and request signals are de-asserted. The Arbiter at OCM in the proposed baseline switch model is a *four-way Arbiter* proposed by [16]. All the communication in the switch follows a 4-Phase protocol.

3.2.2 Pipeline Stage Architecture

The switch is built on the existing PD-hybrid asynchronous pipeline architecture that uses a hybrid data path [34]. Figure 5 illustrates the generic structure of PD-Hybrid pipeline architecture. The pipeline architecture is based on the simple and robust 4-Phase protocol. So, the control path of this pipeline is simple. Logic gates in each pipeline stage are constructed on domino logic style. This pipeline style comes with no stringent timing constraints and is suitable for high throughput applications, especially in the case of highly variable data path delays. Each pipeline stage contains a logic block, a stage controller and a completion detector. In steady-state operation, the logic block alternately generates valid data items and spacers for the next stage, and its completion detector specifies the completion of the stage's precharge or evaluation. The stage controller produces the precharge (pc) and evaluate (ev) signals to control the completion detector and the logic block.

The pipeline stage starts evaluation which is *evaluate phase* if it is ready and the preceding stage supplies valid data. After evaluating the input data, the stage enters into a *isolate phase* by itself and waits for the acknowledgement from its next stage. The completion detector generates an acknowledgement signal to the preceding stage. On receiving the acknowledgement from its next stage, it enters into *precharge phase*. After precharge, the stage proceeds with the next cycle on the arrival of new valid data inputs. There is only one synchronization between the stages; hence high throughput is achieved.

In the proposed baseline switch architecture, the pipeline stage's logic block comprises a bank of domino latches. The first bit of the flit is encoded in DR, while the rest of the bits in the flit are in SR encoded. Therefore, only the first latch in the bank of latches is realized in the Dual-rail domino synchronizing logic gate (DR-SLG) latch [44], while the rest are simple SR domino latches. The delay through the critical paths for all the applied data patterns is constant in SLG implementation. The SR and DR-SLG domino latch form the logic block of the proposed switch pipeline stages. Figure 6 shows the Domino Logic latch implementation for the pipeline stage logic block. The pipeline stage modeled in the proposed switch acts as a buffer to decouple the flit from the upstream link at the ICM and from the arbitration cycle to the downstream link at the OCM.

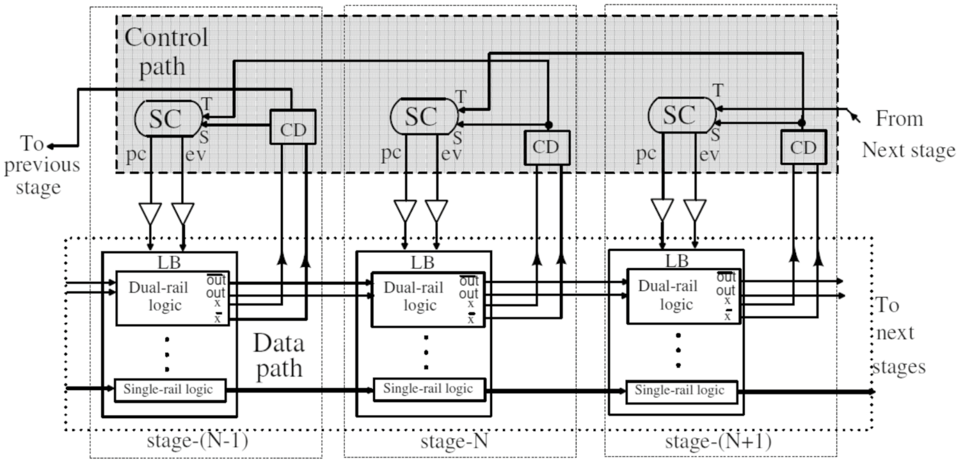


Fig. 5 Generic structure of PD-hybrid asynchronous pipeline [35]

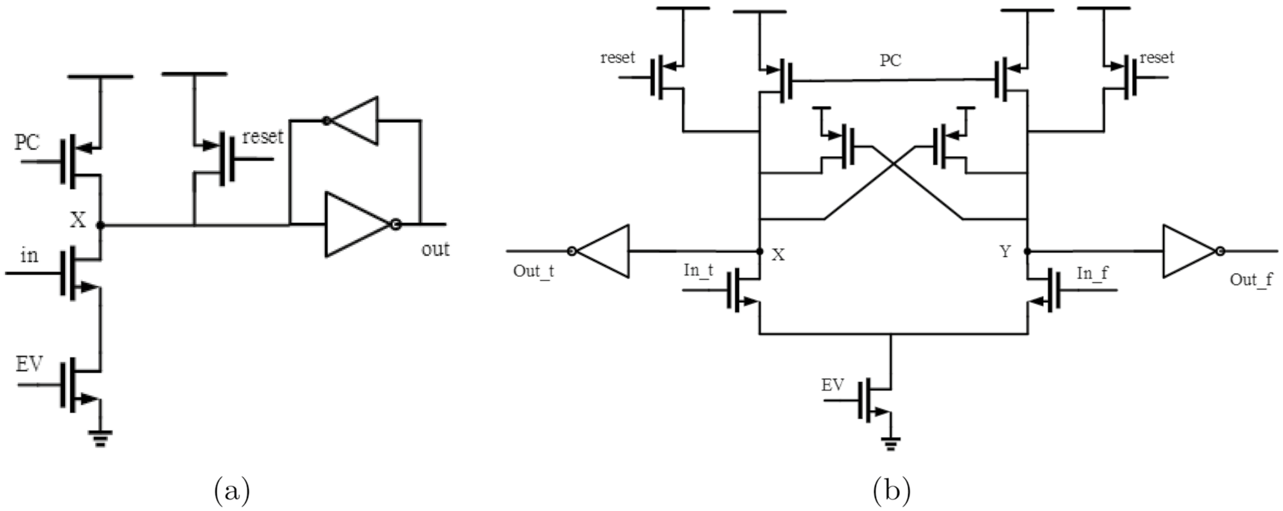


Fig. 6 Domino Logic latch implementation for the pipeline stage logic block **a** Single-rail latch **b** Dual-rail-synchronizing logic gate (DR-SLG) latch

4 Functional Verification

This section describes the simulation of an asynchronous circuit using a commercial simulation tool designed for SoCs. Specifically, it details the proposed functional verification of the baseline switch through a mixed transistor/gate-Register-transfer level (RTL)/behavioural circuit level modeling and simulation flow using Virtuoso Advanced mixed-signal simulation (AMS) environment and simulator in Cadence (Spectre AMS Designer) [11]. Simulation results are also presented.

4.1 AMS Simulator

Spectre AMS Designer extends the event-based simulator loops of Verilog/ System Verilog/VHDL to a continuous-time simulator, which solves the differential equations in analog domain. Both analog and digital domains are coupled: analog events can trigger digital actions and vice versa. It supports designs at three levels: transistor/gate, transistor/gate-RTL/behavioral, and mixed transistor/gate-RTL/ behavioral circuit levels. This allows designers to verify complex designs with ease. Further, digital asynchronous logic falls under discrete value but in continuous time computation. So, the AMS simulator is the best platform for verification of Asynchronous logic design.

This work proposes to perform functional verification of the proposed baseline switch by a mixed-level simulation on Cadence's AMS designer. The modules that encapsulate high-level behavioural/structural descriptions of the router switch and transistor-level domino logic PD-hybrid pipeline stages can be integrated on the same platform. Thus, using the Spectre AMS simulator, the design of the proposed baseline asynchronous NoC router switch is formalized, and the feasibility of the design is demonstrated through functional verification.

4.2 Verification flow and Simulation Setup

This section details the functional verification testbench. Figure 7 illustrates the verification flow using the AMS Virtuoso-based simulator and the required simulation setup. The flow is structured into two stages: Design entry and AMS Virtuoso-based verification flow. In the design entry stage, as stated in Fig. 7, the proposed baseline asynchronous NoC switch design is segregated based on router/switch logic and pipeline stages. The PD-Hybrid pipeline template used for the pipeline stage embeds Domino logic. Usually, a full custom design approach is opted for the domino logic family due to the unavailability of a standard cell library. In addition to this, the traditional automated design frameworks have not been able to support its noise and timing constraints [3]. So, the Domino logic PD-hybrid pipeline template is designed by the typical bottom-up approach at a transistor level.

In contrast, the router logic part, ICM and OCM sub-blocks are designed by the top-down approach as it takes too much time to design and simulate such complex blocks at the transistor level by the bottom-up approach. So, the critical challenge in asynchronous switch functional verification is instantiating and integrating all the domino pipeline stage, input and output computations in the same platform at an early stage of the design cycle. Thus, to address this challenge in verification, a mixed-level simulation can be utilized, as discussed in Fig. 7.

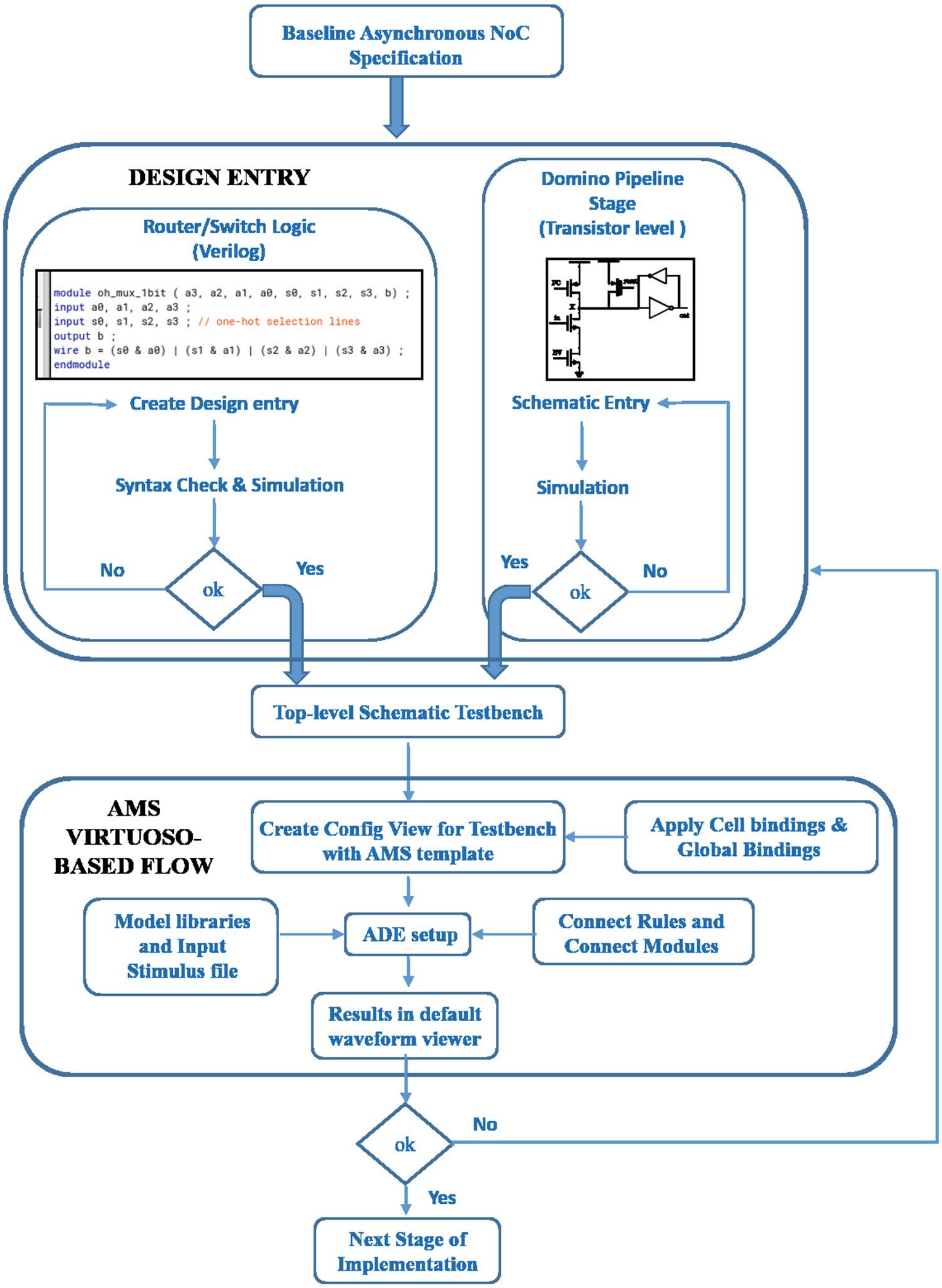


Fig. 7 The verification flow for the proposed baseline asynchronous NoC using AMS

Baseline Asynchronous NoC Specification
DESIGN ENTRY
Router/Switch Logic (Verilog)
Domino Pipeline Stage (Transistor level)
Create Design entry
Syntax Check & Simulation
No
ok
Yes
Schematic Entry
Simulation
Yes
ok
No
Top-level Schematic Testbench
AMS VIRTUOSO-BASED FLOW
Create Config View for Testbench with AMS template
Apply Cell bindings & Global Bindings
Model libraries and Input Stimulus file
ADE setup
Connect Rules and Connect Modules
Results in default waveform viewer
ok
No
Yes
Next Stage of Implementation

```
module oh_mux_1bit ( a3, a2, a1, a0, s0, s1, s2, s3, b) ;
input a0, a1, a2, a3 ;
input s0, s1, s2, s3 ; // one-hot selection lines
output b ;
wire b = (s0 & a0) | (s1 & a1) | (s2 & a2) | (s3 & a3) ;
endmodule
```

The main blocks of the switch are RCM, Arbiter, Multiplexer (MUX) and De-multiplexer (DEMUX). The RCM block of the ICM is described in high-level behavioural Verilog. In contrast, the asynchronous 4-way arbiter module, MUX and DEMUX are described at the structural level. The basic blocks of this arbiter module are C-element [6] and Mutex [40], constructed at the gate level from their standard cell design implementations. While the Domino logic PD-hybrid pipeline stages are realized at the transistor level in the UMC 65nm technology node at 1 V supply voltage. In the verification setup, as seen from Fig. 7, after the preliminary verification of the design entities separately, a top-level schematic test bench of the baseline switch with pipeline stages and router's ICM and OCM are instantiated and integrated as described in the Sect. 3. These switches are instantiated to form 4x4 NoC. Then Cadence AMS Virtuoso-based [12, 13] flow is followed for the simulation as shown in Fig. 7. A *configuration/config* view of the top-level schematic is created for the AMS simulator. The *config* view specifies the view to use for netlisting each block or module in the schematic.

The input packets for the simulation were generated using a script and included in the ADE as a stimulus file. The data packet is injected at a rate of 1.5 Gbits /s at the core

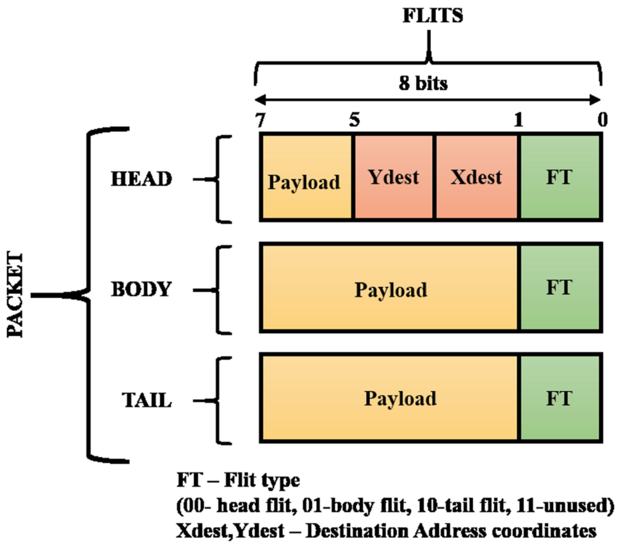

FLITS
8 bits
7
5
1
0
PACKET
HEAD
Payload
Ydest
Xdest
FT
BODY
Payload
FT
TAIL
Payload
FT
FT – Flit type
(00- head flit, 01-body flit, 10-tail flit, 11-unused)
Xdest,Ydest – Destination Address coordinates

Fig. 8 Flit structure within a packet

interface. This rate is determined based on the throughput of the pipeline stage designed. So, the switch architecture under test receives the test vectors, i.e., data packets from the injector block, and when the packet reaches the destination, it is out to the absorber block. The injector and absorber blocks are the same pipeline stage instances modified to inject and eject the test vectors. A data packet consists of a head flit containing the destination address, body flits with data /payload, and a tail flit to mark the end of a packet. For the system, we assume that we have a network consisting of 16 individual cores. Thus, we require 4 bits to represent the address of each core. In addition, we need 2 bits to represent the incoming flit type identifier: head, body, or tail. Each flit is of width 8-bits, and the flit type is identified using their 2-bit code. The structure of the flits considered for this work is as detailed in Fig. 8. The first 6 bits of a head flit are used to indicate the flit type and the address of the destination core represented as [5 0] in Fig. 8. The body and tail flit carries the payload along with flit type indicating bits. After the initial setup is completed, simulation runs are performed to verify the test bench in the ADE window. The results can be verified on waveform viewer.

4.3 Results and Discussion

A set of test cases were simulated using the Spectre AMS simulator to perform the functional verification of the proposed baseline asynchronous switch design. These test scenarios were brainstormed from the specification and architecture of the design. Further, the cases are drawn up to cover the critical features of the router design, such as the packet traversal, routing algorithm, request arbitration and pipeline control blocks. At this stage of design, the verification approach has no timing information.

4.3.1 Test Case 1: Packet Routing at the Switch Level

In the first test case, three packets of variable length are sent from the *EAST* port to *CORE* and *NORTH* ports of the same switch. One packet of 7 flits and two packets of 3 and 5 flits are sent to *NORTH* and *CORE* ports, respectively. This test case demonstrates packet flow in the router at the switch level. The simulation waveform is shown in Fig. 9. The head flit of the first packet sets the path to the *CORE* output port after the route selection at the *EAST* input port. The remaining flits of the packet flow freely on receiving the grant, which can be seen at *CORE* output port signal in the Fig. 9 exhibiting the wormhole switching. By the end of the first packet transmission, the second packet received at the input port is directed to *NORTH* output port. Similarly, the third packet traverses to its destination. From the simulation waveform, it can be observed that the valid output is present after a small computational latency, and every valid data item is separated by a spacer. A smooth and uninterrupted packet routing at the input port is evident from the simulation waveform.

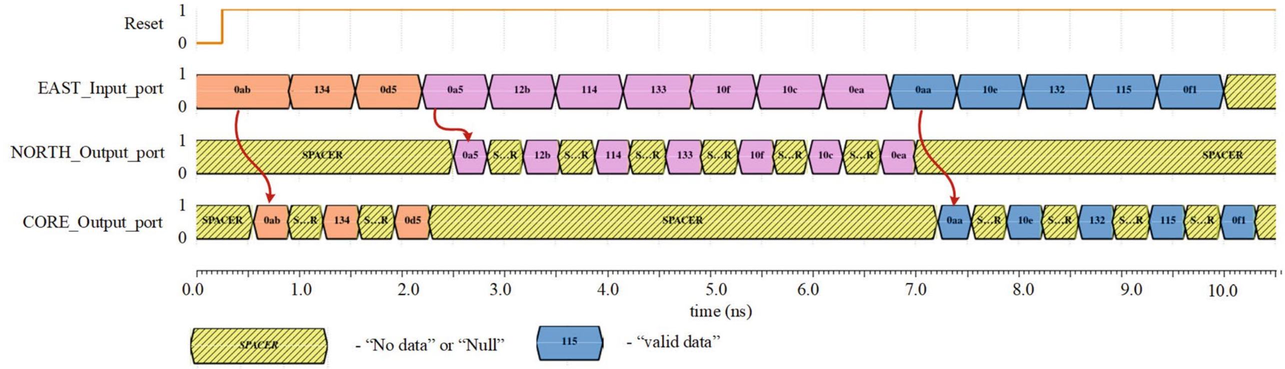


Fig. 9 Simulation result waveform of the packet routing at the input port

4.3.2 Test case 2: Packet Traversal at the Network Level

The next test case simulates a typical scenario of a message travelling from one processing core (Switch-5) to another (Switch-10) under no load conditions in the network. The test case is illustrated in Fig. 11. Based on the *XY routing algorithm*, the packet travels first along the X axis and then travels along the Y axis. The simulation waveform is shown in Fig. 10. For the sake of clarity, only input and output ports are displayed in the simulation waveform. In this case packet from SW5-CORE first reaches to *SW5-EAST* output port and moves to *SW6-WEST* as indicated in Fig. 11. Then the packet travels to *SW10-SOUTH* via the *SW6-NORTH* output port. Finally, the packet reaches its destination *SW10-CORE* output port travelling to NI of the processing core element. The flits of a packet being routed from *SW5-CORE* to *SW10-CORE* in Fig. 10 clearly shows the flit hopping from switch to switch illustrating the basic functional verification of the proposed baseline asynchronous NoC.

4.3.3 Test Case 3: Packet Routing in the Presence of Contention

This test case tests the router with the presence of contention. All the input ports send packets to the *NORTH* output port, except its input port itself. All the other input ports simultaneously compete for the *NORTH* output port. The arbitration is resolved, and a grant is asserted to the winning request. Figure 12 shows the resultant simulation waveform with contention at the *NORTH* output port. The waveform illustrates that the arbiter module functions as expected. It allows only one input port to access the output port at a given time. In this case, *SOUTH* input port request has won the grant to *NORTH* output port. The same can be confirmed by observing the output data at the *NORTH* output port in the simulation waveform. When the winning request port is de-asserted at the end of the packet, the following request port in the queue is channelized to the output port. Here, *CORE* input port request is next in line to win a grant to *NORTH* output port.

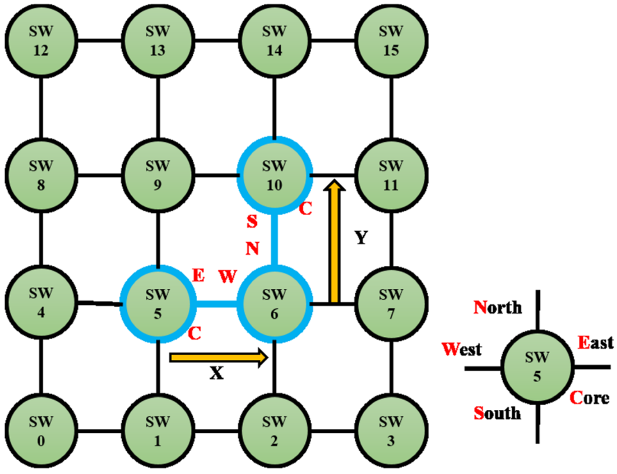


Fig. 11 Illustration of XY algorithmic routing from switch-5 to switch-10 in a mesh

4.3.4 Test Case 4: Concurrent Communication within the Switch

In this scenario, the router is simulated to its maximum throughput, i.e., all five ports communicate simultaneously. The test case is shown in Fig. 13. The communications happen as follows: i) from the *CORE* port to the *WEST* port; ii) from the *EAST* port to the *NORTH* port; iii) from the

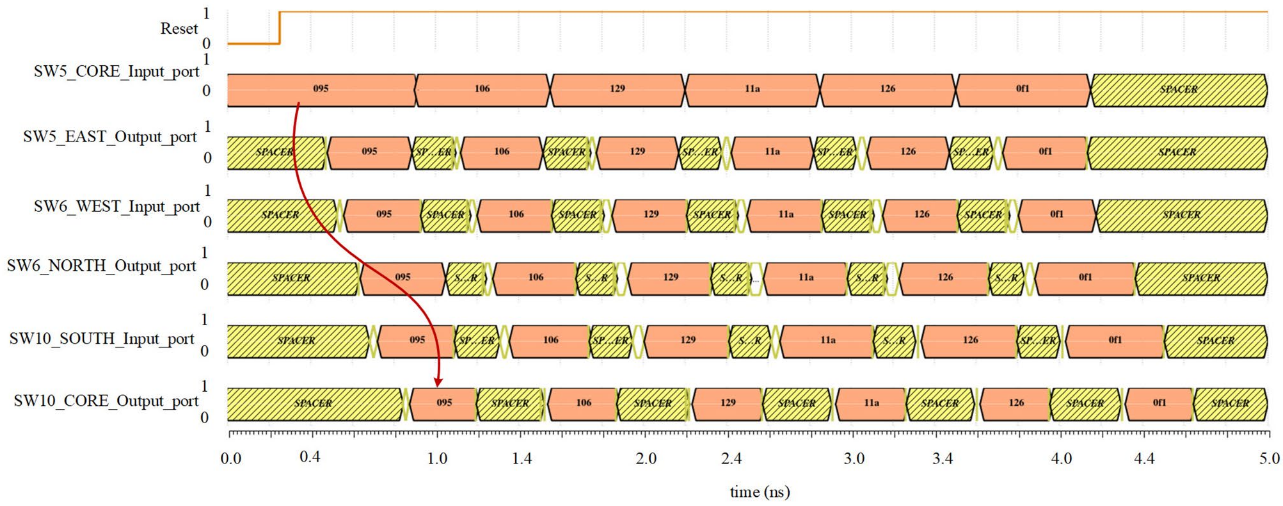


Fig. 10 Simulation result waveform demonstrating the packet traversal through the network

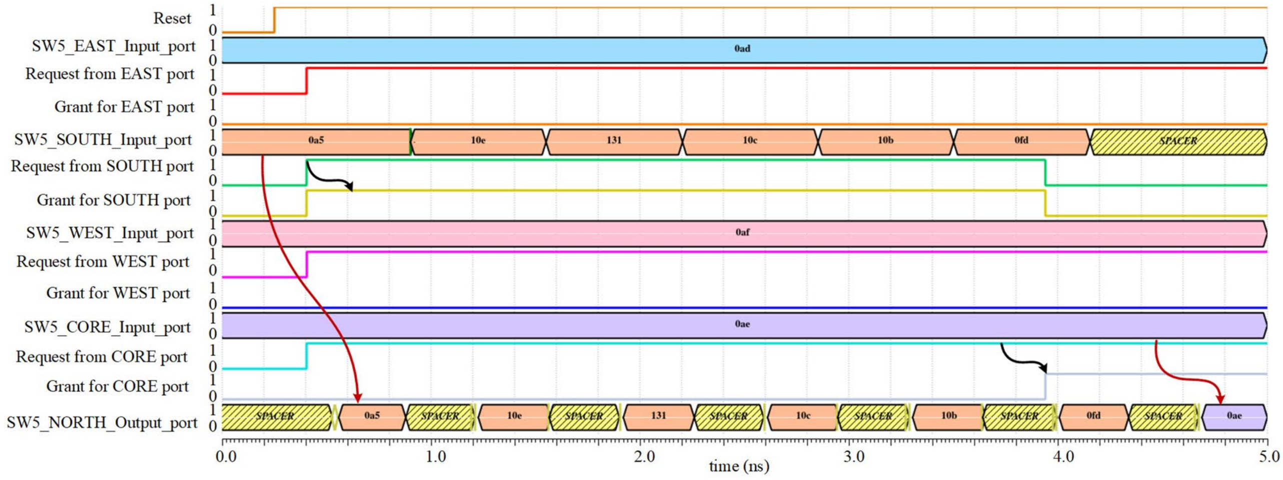


Fig. 12 Resultant simulation plot illustrating routing behaviour under contention

NORTH port to the *EAST* port; iv) from the *WEST* port to the *SOUTH* port; v) from the *SOUTH* port to the *CORE* port. The result of this test case simulation is displayed in Fig. 14. This example shows that packets can go across the same switch simultaneously using different or even the same port, provided they do not go in the same direction. Thus, the functionality of concurrent communication within the switch is validated through this test case.

The architectural design and functional verification of the proposed baseline router are completed using the AMS simulator. Further, the functional verification results demonstrate that the functionality of the proposed design adheres to the intended design specification. Thus, the proposed approach of functional verification is faster and requires less effort.

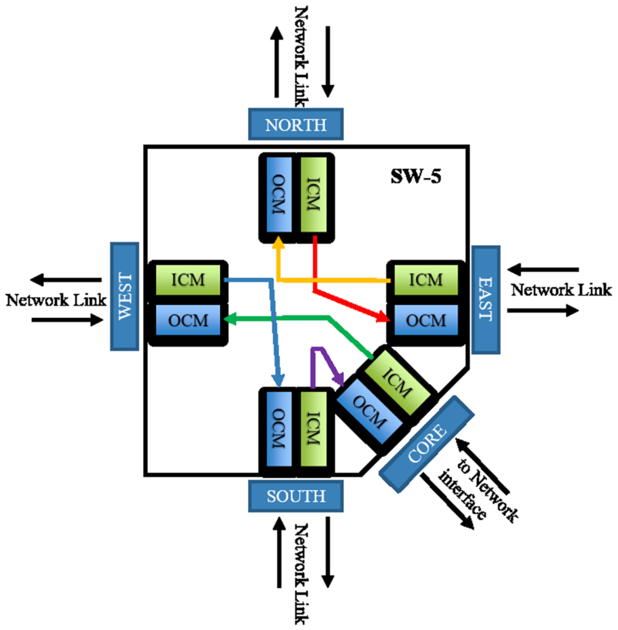


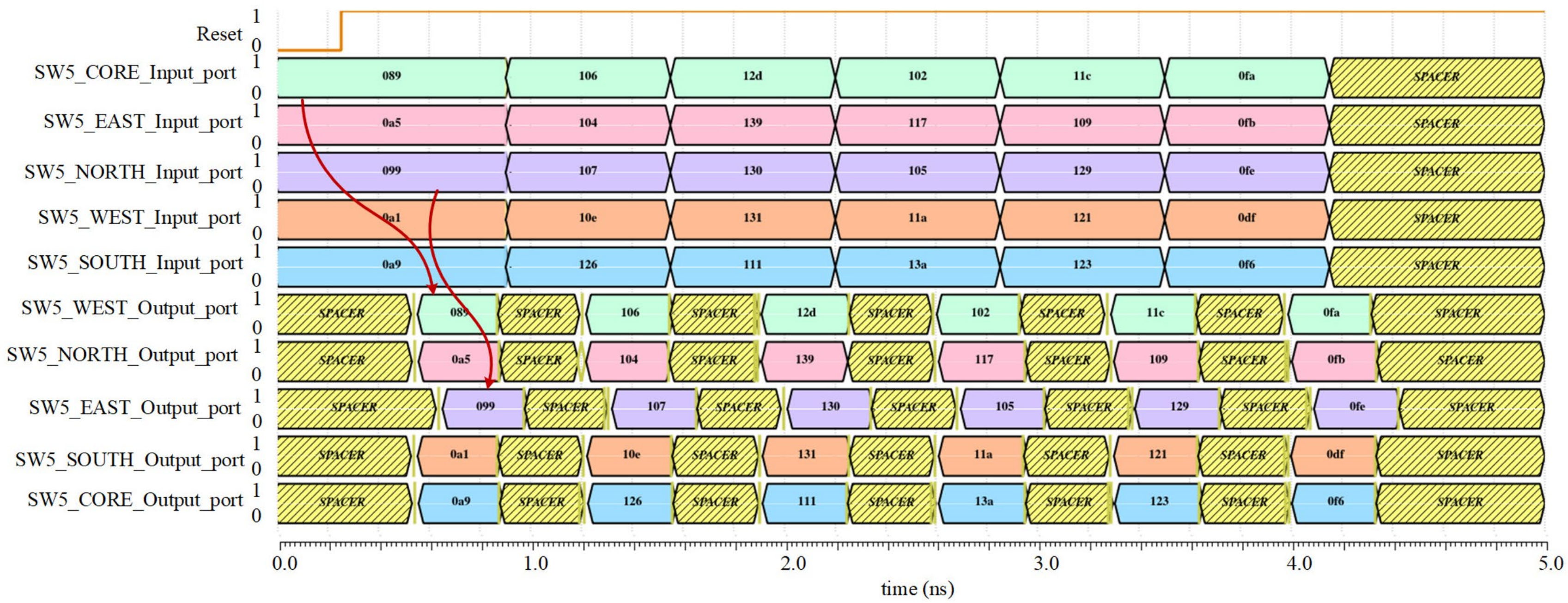
Fig. 13 An illustrated representation portraying concurrent communication among the switch's ports

5 Conclusion

A baseline asynchronous NoC router using the PD-Hybrid pipeline template has been proposed. A novel approach to modeling and simulating a mixed-level abstraction for functional verification of the proposed baseline asynchronous NoC router using Spectre AMS Designer is demonstrated. The test scenarios were carefully chosen to evaluate all the critical features of the design through simulation. The preliminary results confirm the same. These test benches can be utilised in the post-synthesis functional verification with back-annotated delays.

In this approach, there is no need to develop any asynchronous design framework-based language-specific models. It uses *Verilog*, a universally accepted hardware description language by the designers to model the switch functionality and control logic blocks. This work aims to achieve a faster and more efficient design verification for an asynchronous router design before starting the transistor-level implementation. This approach validates the feasibility of the proposed design and aims to address the inherent modeling challenge of asynchronous systems. Through the simulation, modules that are the bottleneck in switch performance can be identified at the preliminary stage of the design. Thus, the design effort can be appropriately channelised in the coming stages of implementation.

Fig. 14 Simulation outcome waveform portraying concurrent communication among the switch's ports



Author Contributions M N Saranya: Conceptualization, Methodology, Data curation, Software, Writing- Original draft preparation, Visualization, Investigation, Rathnamala Rao: Supervision.

Funding The authors did not receive support from any organization for the submitted work.

Availability of Data and Materials Our manuscript has no associated data.

Declarations

Conflict of Interest The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Amde M, Felicijan T, Efthymiou A, Edwards D, Lavagno L (2005) Asynchronous on-chip networks. IEE Proceedings-Computers and Digital Techniques 152(2):273–283
2. Bardsley A, Edwards D (1998) Balsa: An asynchronous circuit synthesis system. University of Manchester UK
3. Beerel PA, Ozdag RO, Ferretti M (2010) A designer's guide to asynchronous VLSI. Cambridge University Press
4. Beigné E, Clermidy F, Vivet P, Clouard A, Renaudin M (2005) An asynchronous NOC architecture providing low latency service and its multi-level design framework. In: Proc. 11th IEEE International Symposium on Asynchronous Circuits and Systems. IEEE. p. 54–63
5. Beigné E, Vivet P (2006) Design of on-chip and off-chip interfaces for a GALS NoC architecture. In: Proc. 12th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC'06). IEEE. p. 10 pp.–183
6. Bhardwaj K, Mantovani P, Carloni LP, Nowick SM (2019) Towards a complete methodology for synthesizing bundled-data asynchronous circuits on FPGAs. In: Proc. 2019 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED). IEEE. p. 1–6
7. Benini L, De Micheli G (2002) Networks on chips: A new SoC paradigm. Computer 35(1):70–78
8. Bertozzi D, Miorandi G, Ghiribaldi A, Burleson W, Sadowski G, Bhardwaj K et al (2020) Cost-effective and flexible asynchronous interconnect technology for GALS systems. IEEE Micro 41(1):69–81
9. Bjerregaard T, Mahadevan S, Sparsø J (2004) A channel library for asynchronous circuit design supporting mixed-mode modeling. In: Proc. International Workshop on Power and Timing Modeling, Optimization and Simulation. Springer. p. 301–310
10. Broenink J, Roebbers H, Sunter J, Welch P, Wood D (2005) High level modeling of channel-based asynchronous circuits using verilog. In: Communicating Process Architectures 2005: WoTUG-28: Proceedings of the 28th WoTUG Technical Meeting, 18-21 September 2005, Technische Universiteit Eindhoven, The Netherlands. vol. 63. Citeseer. p. 275
11. Cadence Design Systems I. Spectre AMS Designer. https://www.cadence.com/ko_KR/home/tools/custom-ic-analog-rf-design/circuit-simulation/spectre-ams-designer.html
12. Cadence Design Systems I. Spectre AMS Designer and Xcelium Simulator Mixed-Signal User Guide, Version 22.03
13. Cadence Design Systems I. Virtuoso ADE Explorer User Guide, Version IC6.1.8
14. David I, Ginosar R, Yoeli M (1995) Self-timed is self-checking. J Electron Test 6:219–228
15. Effiong C, Sassatelli G, Gamatie A (2017) Scalable and power-efficient implementation of an asynchronous router with buffer sharing. In: Proc. 2017 Euromicro Conference on Digital System Design (DSD). IEEE. p. 171–178
16. Ghiribaldi A, Bertozzi D, Nowick SM (2013) A transition-signaling bundled data NoC switch architecture for cost-effective GALS multicore systems. In: Proc. 2013 Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE. p. 332–337
17. Gibiluka M, Moreira MT, Moraes FG, Calazans NLV (2015) BAT-Hermes: a transition-signaling bundled-data NoC router. In: Proc. 2015 IEEE 6th Latin American Symposium on Circuits & Systems (LASCAS). IEEE. p. 1–4
18. Guazzelli RA, Fesquet L (2020) At-speed DfT Architecture for Bundled-data Design. In: Proc. 2020 IEEE International Test Conference (ITC). p. 1–9
19. Hansen J, Singh M (2008) Concurrency-enhancing transformations for asynchronous behavioral specifications: A data-driven approach. In: Proc. 2008 14th IEEE International Symposium on Asynchronous Circuits and Systems. IEEE. p. 15–25
20. Hoare CAR et al (1985) Communicating sequential processes. vol. 178. Prentice-hall Englewood Cliffs
21. Ho WG, Chong KS, Ne KZL, Gwee BH, Chang JS (2017) Asynchronous-Logic QDI quad-rail sense-amplifier half-buffer approach for NoC router design. IEEE Trans Very Large Scale Integr VLSI Syst 26(1):196–200
22. Jergerand N, Peh L (2009) On-Chip Networks: Synthesis Lectures on Computer Architecture. Mark Hill
23. Moreira MT, Magalhães FG, Gibiluka M, Hessel FP, Calazans NL (2013) BaBaNoC: an asynchronous network-on-chip described in Balsa. In: Proc. 2013 International Symposium on Rapid System Prototyping (RSP). IEEE. p. 37–43
24. Nowick SM, Singh M (2015) Asynchronous design-Part 1: Overview and recent advances. IEEE Des Test 32(3):5–18
25. Nowick SM, Singh M (2015) Asynchronous design-Part 2: Systems and methodologies. IEEE Des Test 32(3):19–28
26. Pletscher-Frankild S, Sparsø J (2000) Channel Abstraction and Statement Level Concurrency in VHDL++. In: Proceedings of the 4th Asynchronous Circuit Design Workshop
27. Pontes J, Moreira M, Moraes F, Calazans N (2010) Hermes-A–an asynchronous NoC router with distributed routing. In: Proc. International Workshop on Power and Timing Modeling, Optimization and Simulation. Springer. p. 150–159
28. Pontes JJ, Moreira MT, Moraes FG, Calazans NL (2010) Hermes-AA: A 65nm asynchronous NoC router with adaptive routing. In: Proc. 23rd IEEE International SOC Conference. IEEE. p. 493–498
29. Renaudin M, Vivet P, Robin F (1999) A design framework for asynchronous/synchronous circuits based on CHP to HDL translation. In: Proceedings. Fifth International Symposium on Advanced Research in Asynchronous Circuits and Systems. IEEE. p. 135–144
30. Saifhashemi A, Pedram H (2003) Verilog HDL, powered by PLI: a suitable framework for describing and modeling asynchronous circuits at all levels of abstraction. In: Proceedings 2003. Design Automation Conference (IEEE Cat. No. 03CH37451). IEEE. p. 330–333
31. Sheibanyrad A, Greiner A, Miro-Panades I (2008) Multisynchronous and fully asynchronous NoCs for GALS architectures. IEEE Des Test Comput 25(6):572–580
32. Song W, Zhang G (2022) Asynchronous On-chip Networks and Fault-tolerant Techniques. CRC Press
33. Sparsø J (2020) Introduction to Asynchronous Circuit Design. Technical University of Denmark, DTU Compute
34. Sravani K, Rao R (2017) High throughput and high capacity asynchronous pipeline using hybrid logic. In: Proc. 2017 International Conference on Innovations in Electronics, Signal Processing and Communication (IESC). IEEE. p. 11–15

35. Sravani K, Rao R (2020) Novel Asynchronous Pipeline Architectures for High-Throughput Applications. Arab J Sci Eng 45(8):6625–6638
36. Sravani K, Rao R (2020) Design of high throughput asynchronous FIR filter using gate level pipelined multipliers and adders. Int J Circuit Theory Appl 48(8):1363–1370
37. Sravani K, Rao R (2020) A high performance early acknowledged asynchronous pipeline using hybrid-logic encoding. Integration 71:134–143
38. Tran XT, Durupt J, Bertrand F, Beroulle V, Robach C (2006) A DFT Architecture for Asynchronous Networks-on-Chip. In: Proc. Eleventh IEEE European Test Symposium (ETS'06). p. 219–224
39. Tran XT, Thonnart Y, Durupt J, Beroulle V, Robach C (2008) A Design-for-Test Implementation of an Asynchronous Network-on-Chip Architecture and its Associated Test Pattern Generation and Application. In: Proc. Second ACM/IEEE International Symposium on Networks-on-Chip (nocs 2008). p. 149–158
40. van Berkel K, Huberts F, Peeters A (1995) Stretching quasi delay insensitivity by means of extended isochronic forks. In: Proceedings Second Working Conference on Asynchronous Design Methodologies. p. 99–106
41. van Berkel K, Kessels J, Roncken M, Saeijs R, Schalij F (1991) The VLSI-programming language Tangram and its translation into handshake circuits. In: Proceedings of the European Conference on Design Automation. IEEE. p. 384–389
42. Vivet P, Thonnart Y, Lemaire R, Santos C, Beigné E, Bernard C et al (2016) 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 J Solid State Circuits 52(1):33–49
43. Xia Z, Hariyama M, Kameyama M (2014) Asynchronous domino logic pipeline design based on constructed critical data path. IEEE Trans Very Large Scale Integr VLSI Syst 23(4):619–630
44. Xia Z, Ishihara S, Hariyama M, Kameyama M (2010) Synchronising logic gates for wave-pipelining design. Electron Lett 46(16):1116–1117
45. Yakovlev A, Vivet P, Renaudin M (2013) Advances in asynchronous logic: From principles to GALS & NoC, recent industry applications, and commercial CAD tools. In: Proc. 2013 Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE. p. 1715–1724

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

M.N. Saranya received her B.E. in Electronics and Electrical Engineering from Thiagarajar College of Engineering, Tamilnadu, India, in 2012 and subsequently joined as Associate System Engineer with IBM. She worked with IBM for five years and then pursued her M. Tech in VLSI Design from Vellore Institute of Technology, Vellore, India, in 2019. Saranya is currently working on her PhD, which involves designing of Asynchronous router for a Network on Chip in a GALS system at the National Institute of Technology, Karnataka. Her research interests include VLSI architectures and asynchronous circuits.

Rathnamala Rao received her B.E degree in Electronics and communication Engineering from Visvesvaraya Technological University, Karnataka, India in 2002. She received her Ph.D degree from the Indian Institute of Technology Madras (IITM), in 2011. She is currently an assistant professor in the Department of Electronics and communication Engineering at National Institute of Technology, Karnataka. Her research interests iclude VLSI design, Semiconductor device modeling, Micro actuators and control system design for micromachining applications.