A Study of Recent Contributions on Simulation Tools for Network-on-Chip (NoC)

Muthana Saleh Alalaki, Michael Opoku Agyeman
Department of Computing and Immersive Technologies
The University of Northampton, UK
Michael.OpokuAgyeman@northampton.ac.uk

Abstract

The growth in the number of Intellectual Properties (IPs) or the number of cores on the same chip becomes a critical issue in System-on-Chip (SoC) due to the intra-communication problem between the chip elements. As a result, Network-on-Chip (NoC) has emerged as a new system architecture to overcome intra-communication issues. New approaches and methodologies have been developed by many researchers to improve NoC. Also, many NoC simulation tools have been proposed and adopted by both academia and industry. This paper presents a study of recent contributions on simulation tools for NoC. Furthermore, an overview of NoC is covered as well as a comparison between some NoC simulators to help facilitate research in on-chip communication.

Keywords: Network-on-Chip (NoC), System-on-Chip (SoC), Embedded Systems, Computer Architecture.

I. INTRODUCTION

The significant improvement in scaling and nano scale technologies will make it possible to build a System-On-Chips (SoCs) with billions of transistors and hundreds of cores [1][2][3][4]. Thus, a significant new application development in the field of consumer electronics, telecommunications, and system platforms are expected[5]. On the other hand, these systems will need powerful communication architectures in order to achieve different purposes. Moreover, these architectures should provide templates reusability to reduce the cost involving in design productivity [6]. The current SoCs, where the sustainable communication templates used, primarily depend on the bus approach which includes either a hierarchy of buses or a single shared bus [7].

On the other hand, this approach has faced additional constraints which restrain the future of SoCs.

Firstly, the operating frequency of buses reduces the growth in the number of cores connected. Secondly, low-level scalability with the system size when using a hierarchy of buses of a single shared bus. Thirdly, the energy consumption increases when the wire length increases with increase in circuit size. Finally, only one communication is allowed at a time by a bus or even in a hierarchic bus architecture [8].

To achieve the major requirements of future SoC such as scalability, reusability, parallelism while dealing with clock distribution and power constraints, a new paradigm, which is called Network-on-Chip (NoCs), has emerged[9]. It is a switching network which has been proposed as an alternative to interconnected cores in SoCs[10][11]. This paper conducts a study of recent contributions of simulation tools for NoC.

The rest of the paper is structured as follows: Section II provides an overview of the NoC concepts. Section III discusses the recently proposed simulation tools for NoCs, and Section IV concludes the paper.

II. NETWORK-ON-CHIP (NOC)

Network-on-Chip (also known as packet-switched on-chip interconnection networks) is one of the agitated approaches in on-chip communication. NoCs are multi-hop interconnections networks that are integrated on one single chip [12][13][14][15]. Table 1 shows a qualitative comparison between NoCs and conventional Buses. NoC consists of a number of routers which are structurally interconnected by point-to-point channels as shown in figure 1. Each of these routers is connected to its neighbor as well as a processing core using a set of ports (known as local ports or terminals) [1].

| Table I: NoC and Buses Comparison [16] |
|-----------------|-----------------------------------|
| **NoCs**        | **Buses**                         |
| Upgrading is significant for designers to develop new concept. | Simple and understandable concept. |
| The bandwidth increases with the network size | Limitation of bandwidth and all elements are sharing it. |
| Distributed routing decisions. | Blockages due to delays generated by arbitrations at the bus especially when the number of masters is significant. |
| Good and fast test coverage. | Slow and problematic regarding testing. |
| Delay transitions can be used by the data transfer because it has point-to-point connections. | Managing the time is difficult. |
| Point-to-point interconnections of each element for all sizes of the network and the local performance is not degraded when scaling. | Parasitic capacitance will be added by each item which increases the electrical performance degradation. |

The model of communication in NoC is called message passing because each the processing cores connected to the network ports are communicating by exchanging messages and each of these messages consists of a header, payload, and trailer.

Figure 1: NoC Concepts [1]

NoC can be characterized by its topology as well as various strategies used for routing, switching, flow control, buffering and arbitration. The network topology describes how nodes and channels are arranged into a graph and the routing describes the process of choosing a path in the graph for each message. Flow control contends with the channel capacity, and switching is the technique of removing the data from the input channel to the output channel. The scheduling task of the use of channel is managed by the arbitration while the unscheduled messages are stored by the buffering task[17][18].

A. Topologies

The topologies of NoCs are significant when designing NoCs because the design of any router is based on it. Many topologies have been categorized into regular and irregular depending on the distribution process of routers within the network [9]. The most well-known topologies are mesh, torus, binary tree, ring, bus, butterfly, etc. (Figure 3)[19][20]:

a) Mesh: it consists of a number of columns and rows where routers are located in the intersection between to links and the computational resources are close to the routers.

b) Torus: it is a basic mesh network with some improvements where the heads and the tails of the columns are connected and the leftand right sides of the rows are connected as well. As a result, it requires more minimal routers and has better path diversity compared to themesh network.

c) Tree: in this topology, nodes represent the routers and leaves represent the computational resources.

d) Butterfly: it is a unidirectional (packets are routed from the left side which is the input to the right side which is the output) or bidirectional (all the inputs and outputs are located on the same side) and it uses a deterministic routing.

Figure 2: NoC Topologies

B. NoC Architecture

NoC normally consists of at least three fundamental elements [10][21]. Network adapters (known as Network Interface) are used to connect cores to the NoC and convert the bus protocol (used by Processing Elements) into network protocol (used by switches). Routing nodes are used to direct packets from source to destination based on the chosen protocol and they contain the routing strategies. Links which links the routing nodes and providing the bandwidth. They consist of one or more than one logical or physical channels for data transmission as well as connectivity between Network adapters and routing nodes and between the routing nodes themselves.

C. Routing

Routing is the transmission of data (packets) from source to destination using defined routing strategies. In literature, the routing schemes have been classified based on several criteria[22][23]. Source routing is a schema where the source node chooses the entire path before sending the packet to the destination. The main disadvantage of this approach is that the packet size will be increased because each one of the packets should carry the routing information through the transmission. Furthermore, the path cannot be updated or changed once the packet left the source. To solve this problem, Distributed routing is used where the routing table is accessed to determine the next hop(router) to forward the packet.

On the other hand, routing algorithms are also classified based on making the decision to choose certain paths among a set of available paths into deterministic, adaptive and oblivious. The source and destination address completely determine the path between which is the deterministic routing algorithm. The state of the network is important to make the routing decisions which is the adaptive routing algorithm. The better data flow can be achieved with this algorithm but it might generate complicated nodes[24][25]. In oblivious routing algorithm, the state of the network is not necessary to choose the route between the source and destination nodes. There are some other routing algorithms proposed in [26][27][28][29], but the main purpose of this paper is to present a study simulation tools, not the routing algorithms.
III. NOC SIMULATION TOOLS

The NoC design space is high dimensional and very large and includes topology optimization, congestion control methodologies, routing algorithms, the number of buffers, link capacities, and virtual channels per link, etc. In addition, the research area is expanding, which means more ideas, techniques as well as new architectures, are and will be proposed. Thus, the demand for simulation tools for evaluating different features of different NoCs proposals and designs increases. In this section, we have explained different simulation tools for NoC. Table II shows a comparison between these simulators and other well-known simulation tools.

A. BookSim

BookSim is a cycle-accurate and detailed simulator designed for NoCs as well as modeling interconnection networks for different systems. The first version of it is called BookSim1 which was a generic simulator and not designed for on-chip environment in specific. It has been used to study different network designs including routing, topology, flow control, quality-of-service, router microarchitecture as well as some new technologies like nanophotonic[30][31]. In terms of system simulation, it provides the required flexibility in high-level simulation tools. With regard to network design, all key elements of a network router detailed modeling are provided by BookSim. Furthermore, it is intended to facilitate the modifications and addition of brand new network features. On the other hand, some of the topologies and advanced features proposed for on-chip networks are not supported by original BookSim. Thus, BookSim2 is developed to break the limitations as well as provides models of inter-router channel delay, traffic models, router microarchitecture, highly accurate network model and actual hardware behavior[32].

B. DARSIM

DARSIM is a cycle-level, parallel and a highly configurable simulator for Network-on-Chip as well as supports different virtual channel (VC) allocation and various routing algorithms due to its extremely parametrized table-based design. It can be driven by application traces or synthetic patterns or a built-in MIPS-based multicore simulator. Moreover, it is a parallel (multithreaded) simulation engine that split the tasks equally between the available processor cores as well as cycle accurate precision. Furthermore, it offers periodic synchronization and permitting tradeoffs between high speed and perfect accuracy. In addition, most of the hardware parameters can be configured including bandwidth, geometry, pipeline depth and crossbar dimensions[33].

C. Gem5

Gem5 is one of the emerged simulators that overcome most of the limitations that faced the Computer architecture researchers. It is available for many researchers with a wide range of systems evaluation capabilities as well as clean interface and modularity commitment. In [34], its infrastructure is a combination of the best features of both GEMS [35] and M5 simulator[36]. That means it supports most of the features of these two simulators such as multiple ISAs, highly configurable, diverse CPU models from M5 and flexible, detailed system, including multiple cache coherence protocols as well as interconnect models from GEMS. In addition, it supports various commercials ISAs (ALPHA, MUPS, ARM, SPARK, X86, and Power), involving booting Linux on ALPHA, x86, and ARM. Moreover, high level of collaboration is one the aspects of this simulator provided by the gem5 community because it is a community-led project and open source. Furthermore, additional features such as Parallelization and a first-class power model which might be added soon, many versions of Gem5 simulator have emerged such as gem5-gpu[37] and Gem5v [38].

D. NOXIM

Noxim simulator is developed using a C++ system description library called SystemC. The reason behind choosing SystemC is the essential aim of Noxim project which includes scalable performance and allowing expansion easiness while the cycle-accurate simulation is still maintained[39]. It also supports several parameters of the configuration space which are the workload, topology and structure, simulation and dynamic behavior. In addition, it is an open source simulator with the capabilities of cycle accurate platform, performance analysis and power figures of both emerging Wireless NoC (WiNoC) and conventional wired based NoC[40]. Although it only supports mesh topology, it supports various traffic patterns, different routing algorithm, hotspot injection, as well as

<table>
<thead>
<tr>
<th>Simulator</th>
<th>Year</th>
<th>Benchmark</th>
<th>Framework</th>
<th>Open-source</th>
<th>Topologies</th>
<th>Heterogeneous support</th>
<th>GUI</th>
<th>Synchronous/asynchronous</th>
<th>Ref</th>
</tr>
</thead>
<tbody>
<tr>
<td>BookSim</td>
<td>2010</td>
<td>-</td>
<td>C++</td>
<td>+</td>
<td>Many</td>
<td>-</td>
<td>-</td>
<td>Synchronous</td>
<td>[32]</td>
</tr>
<tr>
<td>DARSIM</td>
<td>2010</td>
<td>+</td>
<td>C++</td>
<td>+</td>
<td>All</td>
<td>-</td>
<td>-</td>
<td>Synchronous</td>
<td>[33]</td>
</tr>
<tr>
<td>Gem5v</td>
<td>2015</td>
<td>+</td>
<td>C++</td>
<td>+</td>
<td>Many</td>
<td>-</td>
<td>-</td>
<td>Synchronous</td>
<td>[38]</td>
</tr>
<tr>
<td>NOXIM</td>
<td>2010</td>
<td>-</td>
<td>SystemC</td>
<td>+</td>
<td>Mesh</td>
<td>+</td>
<td>-</td>
<td>Synchronous</td>
<td>[40]</td>
</tr>
<tr>
<td>HNoCS</td>
<td>2013</td>
<td>-</td>
<td>OMNet++</td>
<td>+</td>
<td>All</td>
<td>+</td>
<td>+</td>
<td>Both</td>
<td>[41]</td>
</tr>
<tr>
<td>Gpnoesim</td>
<td>2007</td>
<td>-</td>
<td>Java</td>
<td>+</td>
<td>All</td>
<td>-</td>
<td>-</td>
<td>Synchronous</td>
<td>[42]</td>
</tr>
<tr>
<td>gem5-gpu</td>
<td>2015</td>
<td>+</td>
<td>C++</td>
<td>+</td>
<td>All</td>
<td>+</td>
<td>-</td>
<td>Synchronous</td>
<td>[37]</td>
</tr>
<tr>
<td>AdapNoC</td>
<td>2016</td>
<td>-</td>
<td>C++</td>
<td>-</td>
<td>Mesh Torus (virtualized)</td>
<td>-</td>
<td>-</td>
<td>Synchronous</td>
<td>[43]</td>
</tr>
<tr>
<td>ENoCS</td>
<td>2015</td>
<td>-</td>
<td>Java</td>
<td>+</td>
<td>Many</td>
<td>+</td>
<td>+</td>
<td>Synchronous</td>
<td>[44]</td>
</tr>
<tr>
<td>MPSoCSim</td>
<td>2015</td>
<td>+</td>
<td>SystemC</td>
<td>+</td>
<td>Mesh</td>
<td>+</td>
<td>+</td>
<td>Synchronous</td>
<td>[45]</td>
</tr>
<tr>
<td>DART</td>
<td>2014</td>
<td>+</td>
<td>SystemC</td>
<td>+</td>
<td>All</td>
<td>+</td>
<td>+</td>
<td>Synchronous</td>
<td>[46]</td>
</tr>
<tr>
<td>Sniper</td>
<td>2015</td>
<td>+</td>
<td>SystemC</td>
<td>+</td>
<td>Many</td>
<td>+</td>
<td>+</td>
<td>Synchronous</td>
<td>[47]</td>
</tr>
</tbody>
</table>

simple source code[48]. In [49], A new version of this simulator has proposed which supports heterogeneous wired/wireless NoC architectures. Furthermore, Access Noxi is a modified version of the original NoC that supports 3D NoC system, Beltway routing, adaptive routing (Proactive thermal budget and Thermal-aware buffer allocation)[49] [50].

E. HNOCs (Heterogeneous Network-on-Chip Simulator)

It is considered as the first introduced heterogeneous NoCs simulator to provide modeling of HNOCs with different link capacities as well as the number of virtual channels per unidirectional port and it is based on OMEv++ [32]. This simulator supports parallelism (by changing the link to unidirectional), different Quality-of-Service (QoS) mechanisms, various arbitrary technologies, many routing protocols as well as power estimation. It supports three types of routers which are asynchronous, synchronous and synchronous virtual output queue (VoQ). In addition, it is open source, fully parameterized, scalable, modular and extendable for NoC. Moreover, sufficient set of statistical measurements is provided at the packet level and the flit such as throughput, end-to-end delay, transfer latencies, VC acquisition latencies, etc.[41].

F. Sniper

Sniper is one of the multi-core simulators based on Graphite simulation infrastructure and the interval core model. It is considered to become the next generation high speed, parallel and accurate X86 simulator for different heterogeneous and homogeneous multi-core architectures. The primary feature of this simulator is the core model which is dependent on interval simulation. It is highly recommended for uncore as well as system-level studies which need more details compared to the traditional one-IPC models. SPLASH-2 benchmarks have been used to evaluate the performance and scalability of this simulator. As an added feature, it provides Python and SimAPI interfaces for controlling and monitoring its behavior at runtime[47].

IV. CONCLUSION

In this paper, we have presented an overview of NoC concepts which is significant in SoCs design. Furthermore, a study of the key but recent simulation tools dedicated for NoC is provided. Particularly, the paper highlights a comparison of the simulation tools to help researchers and developers to decide on the most suitable simulation tool for their research proposals and designs.

REFERENCES

on Computer Architecture and High Performance Computing Workshops (SBAC-PADW), 2016.


