*Covenant Journal of Informatics & Communication Technology. Vol. 7 No. 2, Dec. 2019*

**ISSN: p2354 – 3566 e 2354 – 3507 DOI:**

**An Open Access Journal Available Online**

# BANM: A Distributed Network Manager Framework for Software Defined Network-On-Chip (SDNoC)

**Imeh Jarlath Umoh, Gazal Oluwakemi Marufat, Yahaya Basira, Adekale Dare Abdulfatai & Momoh Omuya Muyideen**

Department of Computer Engineering,
Ahmadu Bello University, Zaria, Nigeria.
Corresponding authors {ime.umoh; momuyadeen}@gmail.com

**Received- May 5, 2019 Accepted- August 9, 2019**
**Date of Publication: December, 2019**

***Abstract***: In the SDNoC architecture; the performance of a centralized network manager (NM) decreases as the arrival of new requests increases. This paper presents a review of a Balance Network Manager (BANM) as a software implemented distributed network manager for SDNoC. BANM uses the principle of Software Defined Network (SDN). The SDNoC network is separated into control network and data network. BANM is executed on a dedicated core and separates the requests from the Network Interfaces (NIs) into local request and global request based on the distance between the source and destination. It uses Reqcheck to determine if a request is local or global, Reqreroute is used to reroute global requests. Local requests are handled by local BANMs, while the global BANM handles global requests. The BANM framework is expected to reduce control congestion on the NM in the SDNoC architecture, and increase performance.
***Keywords***:. Network-on-Chip, phits, MultProcessor System-on-Chip, Software Defined Network, Congestion Control, Network Manger

## 1. Introduction

The tremendous growth in technology has led to an invention of a single chip integrated circuits called System on-Chip (SoC) (Jarraya et al., 2014). This consists of components such as dedicated hardware's, programmable processors, and peripheral such as the on-chip communication architecture (for communication between components), input-output interfaces, and memories (Kreutz et al., 2015). A more complex chips that consist of multiple processor came to existence as a result of advancement in technology coupled with the demand for high computing devices (e.g. smartphones with cameras, GPS devices, Tablets) (Jarraya et al., 2014). These chip which consist of multiple processors and hundreds of thousands of additional components are often called Multiprocessor System on-Chip (MPSoC).

The design of a chip are characterized with four attributes: The memory, computation, communication, and I/O (Avasare et al., 2005) during program execution, MPSoC components need interact with each other. The on-chip communication architecture is responsible for components communication and accurate transfer of data from a source component to the destination component (Bjerregaard & Mahadevan 2006). In order to satisfy the application specific constraints, the communication architecture must provide low latency or guarantees bandwidth.

With an increasing demand for more on-chip modules, the bus architectures used for communication in MPSoC cannot scale up, hindering system performance as required by applications (Blial et al., 2016). Therefore, there is a need for a solution that can meet up with requirements such as latency, bandwidth and power consumption even intensive parallel communication systems. Hence, Network-on-Chip (NoC) is proposed. It utilizes the embedded switching network to interconnect the modules in SoCs. Networks-on-Chip (NoC) simplify and improve the design of MPSoC (Dally&Towles, 2001; Seitz, 1990) NoCs scale well and support modularity by decoupling communication from computation. To support high computations, there is a need for NoCs ensure high performance and reliability (e.g., packets cannot be lost) ( Konstantin et al., 2017). To provide effective and efficient NoC communication through manageability and programmability, the concept of Software Defined Network-on-Chip (SDNoC) was introduced (Seitz 1990).

Software Defined Networks (SDNs) aimed at decoupling both the control (management) and data planes, centralizes the control of information (to provide a global view) and provides network programmability (of components) (Feamster & Zegura, 2013).This concept of SDN is used in the design of SDNoC; to provide low cost, high performance communication architecture. The SDNoC scheme generally relies on the centralized network manager (NM). This NM can be implemented in software region and run on a dedicated core. The route for requests are computed using the global network view adaptively by the NM. In order to allocate routes by the NM, the configuration instruction are sent to the switches so as to employ virtual circuit switching (Konstantin et al., 2017). The efficient handling of local requests and non-local request is guaranteed by the centralized NM. In order to reduce the load on the NM, frequent events should be handled without modifying switches by processing requests local in scope (i.e. requests that process events from a single switch without using the network-wide state).

The remaining of this paper is structured as follows: Section 2 presents related research areas. Section 3 presents the concept of NoC, SDN and SDNoC. Section 4 describes the distributed Network Manager Framework and the conclusion is provided in section 5.

## 2. Related Research Work

A communication service, congestion controlled best-effort (CCBE) was presented by Brand et al., (2010) in order to address the effects of congestion on network performance in NoC. CCBE connections trade bandwidth for constant and reduced latency. The congestion measure that is used is link utilization. Measurement obtained by hardware analysis of link utilization are moved to a Model Predictive Control (MPC). The MPC decides the CCBE loads.

Avasare et al., (2005) presented a NoC communication management scheme. This scheme is based on a centralized end-to-end flow control mechanism which is implemented in software. Congestion control is the main gain of this scheme. The goal of Avasare's mechanism is to maximize communication throughput and minimize jitter (with respect to the user's requirements). The flow control mechanism provides a weak form of QoS at a low cost (i.e. good for NoC platform targeting multimedia applications).

Cong & Wang (2014) presented a software define network on chip (SDNoC) as a new on-chip network architecture. The SDNoC separates the network into control plane and data forwarding plane; based on the concept of software define networking (SDN). The ingress router acts as the controller. Each application is controlled by a controller. The application have the ability of configuring the NoC according to their own requirements, with their controllers (in the control plane).

Konstantin et al., (2017) presented the SDNoC hybrid hardware/software architecture. The SDNoC architecture sectionalizes the network. A network manager (NM) connects to the data network at a rate of one hop per clock cycle along link-disjoint routes. The bandwidth of the links is completely utilized by the route. As such each link is tied to at most one route. Hence, the critical path in the data network is a function of the length of its link and the delay of its endpoints (i.e., switch or NI). The SDNoC shows excellent performance where the is a slow rate of new requests (local or global).

## 3. Fundamental Concept

The Concept of NoC

NoC is an innovative approach for the design of system on chip (SoC) in order to overcome some of the constraints in bus-based data exchange in MPSoCs. NoC interconnects the components in MPSoCs. Therefore, the term NoC is normally used to refer to the communication framework of MPSoC which consists the hardware, the middleware, the software and the services. This is in addition to design tools that are used in mapping software applications in NoC's (Kreutz et al., 2015).

A. NoC Architectural Principles

NoC design is composed of various building principles. These principles are used in different stages in the NoC such as topology, routing and flow control.

(i) Topology: NoC topology determines the arrangement and interconnection of the NoC components physically. These components usually include network nodes and switches among others. Based on the arrangements, NoC topologies is classified into direct, indirect and irregular networks. Examples of topologies in NoCs include: mesh, torus, flat tree, butterfly, star and ring topology (Masoudin & Ghaffari, 2016).

(ii) Routing: Routing: This is done by routers (or switches). A routing algorithm is used to determine how data (packet) is routed from sender to receiver and also decide the most suitable path to route a message in order to get to the destination (receiver) with minimal cost. Based on the network topology; the main objective of any routing algorithm is to ensure that traffic of packets in the network are distributed uniformly as possible among the available paths. Thus, a good routing algorithm improve network latency and throughput. Some common classes of routing decision used in NoC are: centralized, distributed, source and multiphase routings. Due to these routing decisions, there are different routing algorithms that are used in NoC. These includes oblivious, deterministic and adaptive routing algorithms (Masoudi & Ghaffari, 2016).

(iii) Flow control: This specifies what and how various resources in the network such as buffer and channel capacities together with the control states are allotted to data packets that are passing through the network. Also, it determines how network resources are apportioned to different data packets within the NoC. Flow control can therefore, reduce latency and improve the network throughput. In order to classify flow control techniques, the granularity of the resource allocation is handled based on message, packet, or flit (Masoudi & Ghaffari, 2016).

B. NoC Architecture

The NoC architecture consists of three major blocks. They are explained as follows:

(i) Links: A link physically joins the network nodes and facilitate communication between them. Thus, the link is mainly made up of a group of electrical conductors which joins the network nodes, routers and switches. Thus, these links can also include some logical and physical connections. Therefore, it can be said that it include some logical and physical connections. Therefore, it can be said that based on aforementioned the in NoC data transfer from a node to another involves a serial or a parallel communication link (Avasare et al., 2015)

(ii) Routers: - A router consists basically of sets of input and output buffers, an interconnection matrix, and some control logic. A NoC router is composed of input ports where the node receives incoming packets, output ports where the node sends outgoing packets, a switching matrix connecting the input ports to other output ports, and a local port which is connected to the IP core. The input and output can be connected to shared-NoC channels. During design, the router implements a set of defined policies called protocols; which handles various situations that occur during packet transmission. These situations include routing, deadlock, congestion and livelock, etc. Some factors considered in the design of the router are: flow control policies, buffering policies, switching techniques and routing algorithms (Avasare et al., 2005; Jorg et al., 2004).

(iii) Network Interface (NIs): A network interface provides a logical link that connects the IP cores with the network components. Therefore, the NI is a vital part of the NoC that is used to take care of control of retransmission through marketization and packet re-ordering. It can also be considered as a protocol converter that maps the processing node (Avasare et al., 2015).

The Concept of SDN

Software Define Network (SDN) is a networking approach where the network is managed dynamically through efficient configuration so that better performance is achieved. This is made possible by the separation of the control plane which decides the routing of the traffic and the data plane which forwards the traffic of packets in accordance with the routing decision made by the control plane (Kreutz et al., 2015). The three main principle which SDN is based are (phemins et al., 2014).

1. Separation of software and hardware (physical) layers.
2. Centralization of control information.
3. Programmability of the network and its policies.

Therefore, the SDN architecture is vertically split into three main functional layers (Jarraya et al ., 2014):

1. The infrastructure layer (also called the data plane): These consists mainly of Forwarding Elements (FEs) such as the switches and routers.
2. The control layer (also called the control plane): This consists of a group of software based controllers which strengthens the control functionalities by means of open Application Programmable Interfaces (APIs).
3. Application Layer (also called Application Plane): This consists of business applications for end-users. It uses up the communication and network services of the SDN.

Figure 1 shows the router architecture of SDN (Yeganeh & Ganjali 2012). I/O protocol into the SDN is used within the NoC. The separation of computation and communication in the network is enabled by the NI. The NI allows the reuse of IP core and communication infrastructure such that both of them are used independently.

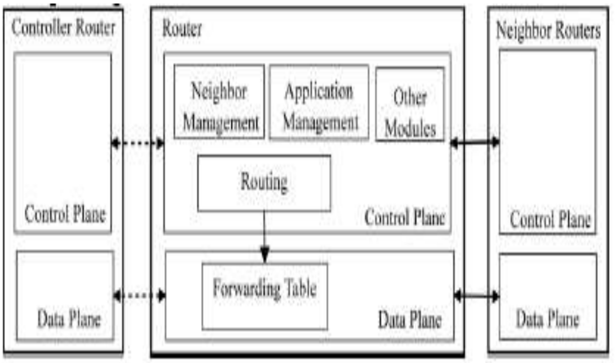


Figure 1: Router architecture on SDN

SDNoC Concept

This involves applying the concept of SDN to NoC to provide effective and efficient communication in MPSoCs. The SDNoC-scheme is depicted Figure 2.

It works based on a central Network Manager (NM) . The NM is implemented as a software and executed as a dedicated IP core. Thus, NM in the NoC provides certain advantages. Theses include the following (Konstantin et al., 2017).

1. The NM has a complete picture of the network. This facilitates effective and efficient apportionment of resources in the network.
2. Computation and apportionment of suitable packet routes less susceptible to small changes in network transit distances.
3. Curtailing overhead in the network: overhead is reduced in the network as the need for routing tables, buffers and routers is eliminated.
4. Curtailing overhead in data communication: The need for headers and control signals is eliminated.
5. Reduction of power consumption: This is made possible through on-demand computation of traffic. Figure 2 shows the SDNoC architecture (Yeganeh & Ganjali

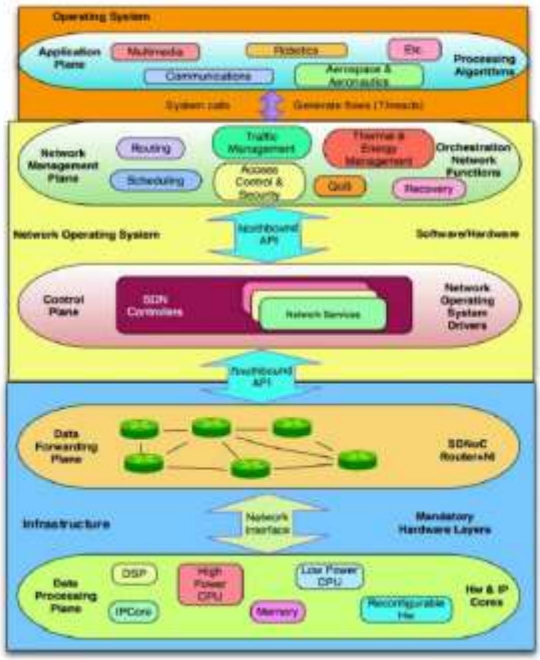


Figure 2: Software Defined Network on-Chip Architecture

SDNOC Architecture

The SDNoC architecture consists of: the physical network, network interfaces (NIs), switches and network manager (NM), the components and their relationship are as shown in Figure 2. In SDNoC; packets (partitioned messages by the cores) containing information such as source, destination and contents are transmitted between cores. Each of the packet is splitted into a number of flits (basic data units delivered over links in the data network) of the same length. The flit is also partitioned into phits (strings which are received along a link in one complete clock). The length of a phit is usually equals the number of data bits per link (Konstantin et al., 2017). The block diagram of the SDNoC is depicted in Figure 3.

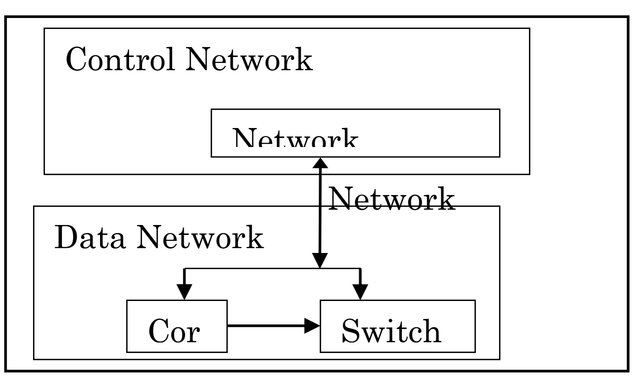


Figure 3: Block diagram of SDNoC

A. Physical Network

The SDNoC architecture separates the physical network into control and data network.

(i) Data Network: Data network comprises of two elements, the network interface (NI) and the switch. Packets are delivered via the link route in the network. The length of a link and the endpoint delay determines the critical path of the network.

(ii) Control Network: The control network provides the framework using a network manager in which all the network interfaces are linked. A centralized network manager is usually employed.

B. Network Interfaces (NI)

As shown in Figure 4, the network interface is links the core to the switch. The NI hands bi-directional traffic. An incoming packet is first stored in the NI's buffer before being forwarded to its core. A single port used between the NI and the switch prevents simultaneously sending and receiving packets.

C. Switches

The Switches (or routers) handle the transmission of packets. Switches contains several ports which can act as either an input port or output port but at the same time. The bidirectional links helps in connecting switches to its neighboring switches. At any point when each switch forward phits from its input to its output port, route is being created in the data network. Switches do not require buffers because it has the ability to store one phits per input port for one clock cycle. Configuration instructions are sent to the switches by the network manager (NM), defining active input and output ports.

D. Network Manager

The network manager (NM) resides on the control plane (control network). Its responsibility is to control the network. The NI sent a request for routing of packet to the NM. The NM has a global information of the entire network. It uses a FIFO queue to store all requests awaiting a route. Most recent request also join the queue. The NM stores the global information for routes allocation (Konstantin et al., 2017; Sandoval et al., 2015).

## 4. BANM: A Distributed Network Manager

Limiting the load on the NM is essential for realizing SDNoC, i.e. a system with congestion control. This involves limiting the overhead of frequent events (local events) in the network. The performance of the centralized NM is inversely proportional to packet arrival time. In accordance to the distance between the source and destination request (Konstantin et al., 2017) classified the request as local and global. We propose BANM; a distributed network manager that classifies request into local (frequent events) and global (rare events). BANM leverages the concept used in Kandoo and Orion (Yeganeh et al., 2012 & Fu et al., 2015). BANM is implemented using python programming language on a dedicated core, using python programming language. BANM creates a level two order for network managers. They are the local network managers which execute local requests even closer to switches, and a logically centralized global network manager that run global requests.

A. BANM

A BANM network consists of both local BANMs and global BANM that are logically centralized. The NMs combines to form BANM distributed network manager. BANM supports the OpenFlow specification of SDN. Each switch is connected to a local NM, and a local NM can control multiple switches. The global NM connects to all local NMs in the network. The BANM local NM will be deployed close to the switches. Figure 4 shows a BANM network with four BANMs: 3 local BANMs controlling switches and a global BANM.

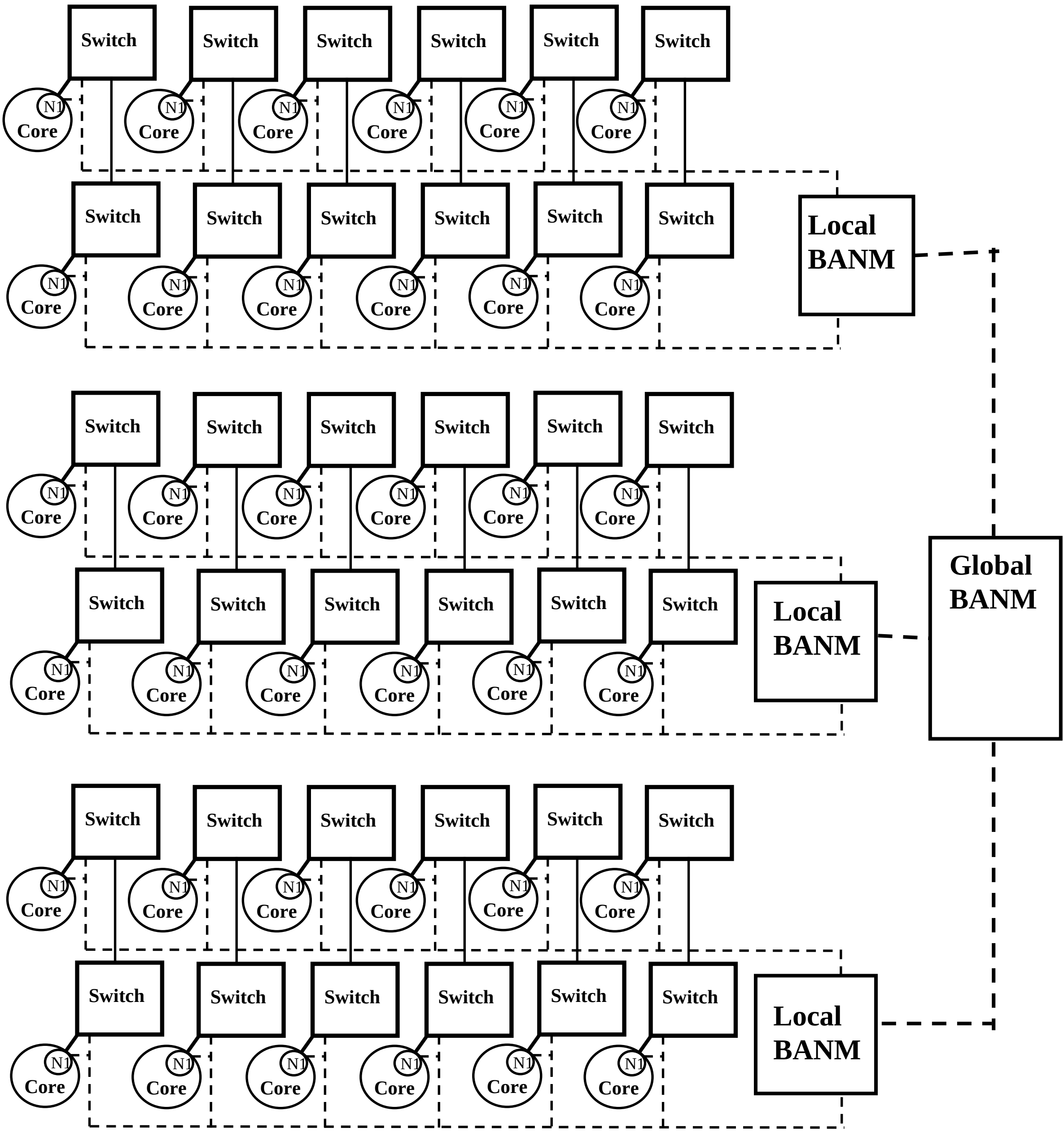


Figure 4: A BANM Network

B. How BANM works
BANM achieves classification of requests (local and global) using Request Detection Applications (Reqcheck and Reqreroute), located in each local BANM. The Request Detection Applications works at the NM. The Reqcheck is
activated at the NM, after the NM receives a request from the NI. The request is checked using the distance of the end to end (source and destination) i.e. if source and destination belong to the same cluster –cores located within the NM cluster. On detecting a request is local; the NM sends a start-reply. A start-reply is an instruction given to the NI to start sending/receiving packets. If request is global; the Reqcheck notifies the Reqreroute, which reroutes the request to the global BANM, which in turn sends a start-reply to the appropriate NI. Thus, Reqreroute requests can only be handled by the global BANM, while Reqcheck is local and can be run by all BANM.

## 5. Conclusion

This work proposes to improve the performance of a centralized Network Manager (NM) in SDNoC architecture. For this purpose, we suggest BANM, a distributed network manager framework that consists of local BANMs and a global BANM. The BANM is implemented using python programming language which run as a program on a dedicated core. Local BANMs are connected to the global BANM. BANM classifies requests into local requests and global requests. The classification is done using the span between source and destination. Local requests are run on the local BANM, while global requests are run on the global BANM. The BANM framework enables the NM provides flexibility (in introducing features such as security, QoS etc.) in conjunction with scalability to the SDNoC architecture.

## References

[1] Avasare, P., Nollet, V., Mignolet, J. Y., Verkest, D., & Corporaal, H. “Centralized end-to-end flow control in a best-effort network-onchip,” in Proceedings of the 5th ACM international conference on Embedded software. ACM, 2005, pp. 17–20.

[2] Blial O., Ben Mamoun M., & Benaini R. (2016). An Overview on SDN Architectures with Multiple Controllers. Journal of Computer Networks and Communications, Article ID 9396525, 8 pages,
http://dx.doi.org/10.1155/2016/9396525

[3] Bjerregaard, T., & Mahadevan S. (2006) A survey of research and practices of network-on-chip. ACM Comput Surv 38:1–51.

[4] Cong, L., Wen, W., & Zhiying, W. (2014, September). A configurable, programmable and software-defined network on chip. In 2014 IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA) (pp. 813-816).

[5] Dally, W. D. & Towles, B. (2001). “Route packets, not wires: On-chip interconnection networks,” in Design Automation

Conference, 2001. Proceedings. IEEE pp. 684–689.

[6] Feamster, N., Rexford, J., & Zegura E. (2013). The road to SDN. Queue, vol. 11, no. 12, pp. 20:20–20:40.

[7] Fu Y., Bi J., Chen Z., Gao K., Zhang B., Chen G., & Wu J. (2015). A Hybrid Hierarchical Control Plane for Flow-Based Large-Scale Software-Defined Networks. IEEE Transactions on Network and Service Management, VOL. 12, NO. 2.

[8] Jarraya Y., Madi T., & Debbabi M. (2014). A Survey and a Layered Taxonomy of Software-Defined Networking. Communications Surveys & Tutorials, IEEE, 16(4), 1955-1980. doi: 10.1109/COMST.2014.2320094

[9] Jorg Henkely, Wayne Wolfz & Srimat Chakradhar (2004). On-chip networks: A scalable, communication-centric embedded system design paradigm y. Proceedings of the 17th International Conference on VLSI Design (VLSID'04).

[10] Konstantin Berestizshevsky, Guy Even, Yaniv Fais, & Jonatan Ostrometzky (2017). SDNoC: Software Defined Network on a Chip. Microprocessors and Microsystems doi:10.1016/j.micpro.2017.03.005.

[11] Kreutz D., Ramos F. M. V., Esteves Verissimo P., Esteve Rothenberg C., Azodolmolky S., & Uhlig S. (2015). Software-Defined Networking: A Comprehensive Survey. Proceedings of the IEEE, 103(1), 14-76. doi: 10.1109/JPROC.2014.2371999

[12] Masoudi Rahim and Ghaffari Ali (2016). Software defined networks: A survey. Journal of Network and Computer Applications, http://dx.doi.org/10.1016/j.jnca.2016.03.016

[13] Phemius K., Bouet M., & Leguay J. (2014). Disco: Distributed Multi-Domain SDN Controllers. Network operations and management symposium (NOMS) IEEE; p.1–4.

[14] Sandoval-Arechiga, R., Vazquez-Avila, J. L., Parra-Michel, R., Flores-Troncoso, J., & Ibarra-Delgado, S. (2015, December). Shifting the network-on-chip paradigm towards a software defined network architecture. In 2015 International Conference on Computational Science and Computational Intelligence (CSCI) (pp. 869-870). IEEE

[15] Seitz, C. L. (1990). "Let's route packets instead of wires," in Proceedings of the sixth MIT conference on Advanced research in VLSI. MIT Press, pp. 133–138.

[16] Van den Brand, J. W., Ciordas, C., Goossens, K., & Basten, T. "Congestion-controlled best-effort communication for networks-on-chip," in Proceedings of the Conference on Design, Automation and Test in Europe, ser. DATE '07. San Jose, CA, USA: EDA Consortium.

[17] Yeganeh S. H. & Ganjali Y., (2012). Kandoo: A Framework

for Efficient and Scalable Offloading of Control Applications. Proc. HotSDN '12 Wksp., pp. 19–24