ISSN 2278-3091
Volume 9 No.2, March - April 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse203922020.pdf
https://doi.org/10.30534/ijatcse/2020/203922020

# Performance analysis of Synchronous Network on Chip (NoC) for High speed and Low Power Multiprocessor on FPGA

**Amaresh. C[1], Dr.Anand Jatti[2]**
[1]Research Scholar, R.V. College of Engineering, Bangalore, India, amareshapj@gmail.com
[2]Associate Professor, R.V. College of Engineering, Bangalore, India, anandjatti@rvce.edu.in

## ABSTRACT

Network-on-chip (NoC) is an emerging paradigm for handling communication in large system-on-chips. This project investigates the ability to prototype asynchronous NoCs on FPGAs. The implementation of asynchronous circuits on standard FPGAs is highly experimental; therefore the _rst part of the project has been to establish a design ow for the implementation of asynchronous circuits on FPGAs. In the project, an asynchronous best-e_ort NoC for an FPGA has been successfully developed. The NOC implementation consists of a router and network adapters and is implemented using a 4-phase bundled data handshake protocol. Cores connect to the network using an OCP interface. To demonstrate the NoC it has been implemented in a small multi-processor prototype using a mesh topology for the network. System-on-chip (NoC) is a rising worldview for taking care of the communication in huge system-on-chips. This researches work is main-ly investigation the capacity to model synchronous NoCs on FPGAs. The design and implementation of synchronous circuits on standard FPGAs is an exceptionally trial; in this manner, the first part of the proposed work has been to set up a design flow for the testing and validation of synchronous circuits on FPGAs. In the network applications a best-effort (BE) NoC for an FPGA has been effectively implemented. The NOC usage comprises of a switch and system connectors and is actualized utilizing a 4-stage packaged information handshake UART and Memory. Cores associate with the system utilizing an Open Core Protocol (OCP) interface. For real-time demonstrate, the designed NoC has been executed in a little multi-processor model utilizing a working topology for the system. The designed 3x3 NoC and UART protocol are successfully simulated using Verilog HDL and tested on Artix-7 FPGA using the Chipscope software tool. The Virtual Input/Output (VIO) and Integrated Controller (ICON) are interfaced with NoC design for data transfer from the source router to the destination router. To verify the effectiveness, Packet Delivery Ratio (PDR), latency and other hardware utilizations like slices, LUT, Flip Flop and area, the packets are generated using Traffic Pattern Generator. From obtained results, it is found there is a 12% improvement in LUT's, 7% in Flip Flop's, 23% in throughput and 26% in delay.

**Key words:** Asynchronous Network System**,** Network Chip Communication, NoC, FPGA, UART.

## 1. INTRODUCTION

Noc analysis is a development filed within the SoC analysis area. Various research teams have presented articles concerning the ideas and NoC systems. Bothe synchronous and Asynchronous NoC's have been discussed in existing works. In this research work three aschronous NoC applications are discussed. MANGO [1], QNoC [2], and Chain [3]. MANGO (Message-passing Asynchronous Network-on-Chip providing Guaranteed Services over OCP interfaces) is evolved at IMM, DTU. MANGO is an associate asynchronous NOC that furnishes bonded services. The core is associated with NOC employing the OCP interface. The NOC consists of both a BE network and a GS network. Connection-oriented GS services providing hard information measure and latency guarantees are enforced utilizing virtual channels. Packets are worm-hole routed within the BE network and source routing is employed [12].

In the projected work Open Cores Protocol (OCP) deployed Adapter for Network-on-Chip by Rasmus Grndahl Olsen [4] ) an enhanced model of a NA for the MANGO NoC is introduced. QNoC (Quality-Of-Service NoC) is initiated at the Israel Institute of Technology. It's asynchronous multi-service NoC. The QNoC router consists various input and output ports connected [14]. The routers are linked in a 3D mesh topology. Credit-based flow control is employed to reinforce output. Four categories of services are presented. This work introduces both a specific service level router and a multi-service level router. The NOC utilizes both source and wormhole routing. Chain (Chip space Interconnect) is a synchronous Best Effort (BE)NoC elaborated at the University of Manchester. Chain utilizes delay-insensitive 1-of-4 for information encoding and an individual wire is employed to signal the end of a packet. The packets in the network are source-routed [13].

The projected work primarily focuses on the following

- Initiating the Core and adaptive networks for effective communication
- Initiating the flit bits along the data bits for effective locating of the directions
- Presenting the traffic generator to avoid the obstructions and intrusion of the information. The credit-based flow will primarily cause the output to increase.

Tobias Bjerregaard. The MANGO Clock less Network-on-Chip: Concepts and Implementation. PhD thesis, Technical University of Denmark, 2005.The objective of the thesis is to execute an asynchronous NoC epitome on a customary FPGA. The earlier work concerning about the execution of asynchronous circuits on FPGAs is extremely confined therefore an important aspect of the project has been to advance an extensive design flow for the realization of asynchronous circuits on FPGAs. An elementary asynchronous best-effort NoC has been initiated. The NOC comprises of a router, a slave NA, and a master Na. The router is intended to perform in a mesh topology and make use of wormhole routing. Deadlock freedom is ensured by making use of xy-routing and supply routing is employed [17-18]. A packet can accommodate an unlimited number of its. To determine the start and stop of a packet, three categories' are used. The flit kind is encoded by adding two extra request signals to the handshake channel. The NAs implements AN OCP consolidates for the cores to connect to the network. Synchronization is managed by employing a straightforward two flip-flop synchronizer. The design used by the router is 660 LUTs and 475 Flip Flop's where 660% of the LUTs are employed by delay components. The overall output of the router has been calculated and found to be 43 MHz. A small multi-processor model using the asynchronous NoC has been executed. The paradigm consists of three CPUs and three peripheral modules that are connected by a 3x2 mesh topology. To guarantee that the system is free from message-dependent deadlocks an individual request and response web is employed. it is not viable to flit a substantial model on the FPGA. It has been corroborated hard to succeed in a high application of the logic resources on the FPGA. It is liable, due to the depletion of the routing resources. Because of these problems performance of the asynchronous NoC is uncertain. There seems to be a general drawback of embracing high utilization figures for the FPGA that has been utilized for the models. However, there is an explanation that the performance of the asynchronous NoC will increase this drawback. The basic issue for executing asynchronous circuits on FPGAs is the delay matching method. Also, undesirable optimizations by the planning tools are ambiguous [15].

To ideally delay match circuit the foregone conclusion of the delay component and also the delay of the information path should each be high. By using comparative placement constraints within the model of the delay component expected output has been achieved. To scale back the variations within the delay of the information path it has been tried to form macros with the locked placement of the basic design. Due to uncertain issues with the proposal tools, it is not been possible to form such macros. As a result, it is required to feature additional delay in the time of delay matching. It's unfeasible to show off the logical optimizations executed by the planning tools. They can be somewhat controlled by the employment of various settings and constraints. For precisely designed circuits like circuits synthesized by Petrify have the outcomes that it is required to do the LUT mapping and deployment manually. For alternative circuits, it has not substantiated to be an oversized issue.

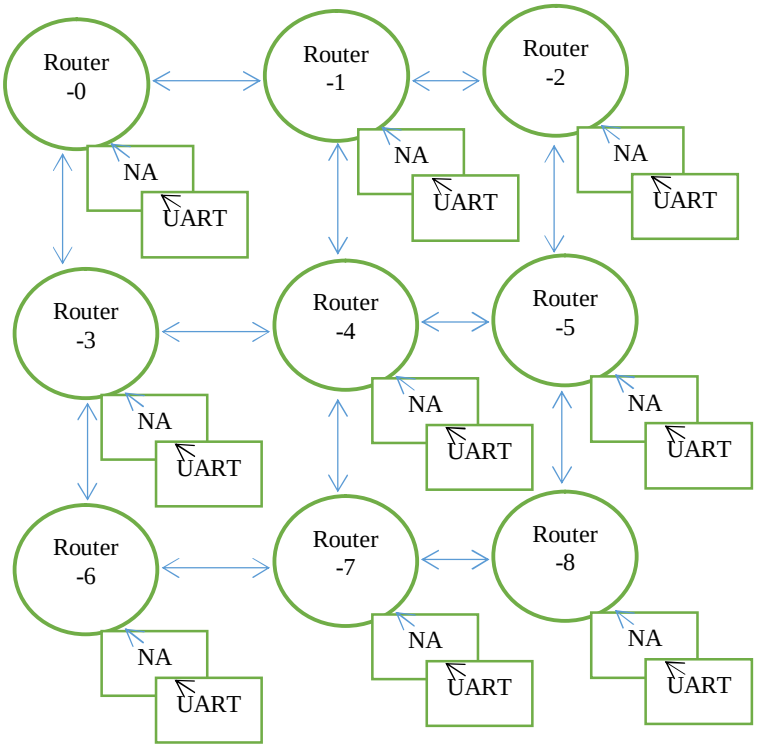


**Figure 1:** General Structure of NoC connected in a 3-by-3 mesh topology.

**A** NoC is composed of 4 elementary units: IP Cores, Network Adapters, Routers, and Links. **An** outline of every NoC module is shown below and is depicted in Fig.1. IP Cores: NoC aims to let the cores within the system to communicate with one another accurately. Thus, the cores do not seem to be a genuine part of the NoC. A core will starts **to requests** (a master core), or reply to **requests (**a slave core) or both. **Conventional** illustrations of master and slave cores are CPUs and memories respectively. Network Adapters (NAs): Allows interface for the core to convey with the NOC. **Usually**, the cores utilize a memory-mapped interface while the network relies on message-passing. The NA must interpret between the two forms of interfaces. NA must additionally manage synchronization problems between the cores and also the network.

**Routers:** are systematically linked to each other to form a network for the cores and to convey on. The routers route the

arriving packets to the desired destination depending on a routing rule. Links: are employed to link the routers to each other. The links comprise of control wires and information wire. The fundamental features of an NOC are the selection of topology, routing formula, shifting strategy, and flow control. The topology evaluates how the network modules are physically connected to every alternative. The routing algorithm actuates the routing information that follows through the network. The switching strategy shows how a message traverses the route. Flow control depicts when messages traverse the route. The topology defines how the routers are connected to each other's. The selection of topology influences several aspects of NoC, e.g. space, conduction, and power. Various topologies exist, e.g. meshes, tori, cubes, butteries, and trees. Topologies are often regular or irregular, while irregular topologies can be formed by combining various types of standard topologies. An instantaneous network has cores connected to all router nodes, whereas an indirect network solely has cores connected to a set of the router nodes. The network in Fig.1 is an illustration of an indirect network. A network is often packet-switched or circuit-switched. In a circuit-switched network, the route from supply to the destination is configured before the message is transmitted on the network and isn't taken down before the transmission is completed. in a very packet-switched network, the message is divided into various packets that are independently routed from source to destination. Every packet comprises advanced NoCs which uses packet switching [5-6]. The minimum quantity of information which will be sent between two nodes within the network is named as a flow control unit (it). A packet is composed of various it's that are transmitted serially. For packet-switched networks various packet switching methods exists**.** The three switching strategies are store-and-forward, wormhole, and virtual cut-through are explained as follows [11].

**Store-and-forward:** all their packets are accepted by the node before any of it is expressed to the successive node. If the successive node doesn't have an acceptable buffer area, the packet is interrupted.

**Wormhole:** once a node accepts it packet, then it is redirected to the successive node as early as possible. In the behind of the nodes, the tail of the packet is left. If the header is cut-off, all links extended by the packet are going to be intercepted.

**Virtual cut-through** is a trade-off between store-and-forward and worm-hole. Facilitating its work is as same approach for the wormhole strategy; However, the header it will be solely redirected to further node if it has adequate buffer area to store the entire packet. The wormhole approach has lower quiescence and lower buffer necessities when correlated with store-and-forward. The drawback is that the chance of deadlocks in the network will be more because a packet can extend various links. Virtual-cut-through will have an advantage from the low latency of the wormhole strategy under normal load. In case of high load, the virtual-cut through strategy can approach the function of a store-and-forward network, wherever it is collected within the front node. Thereby the inflated risk of deadlocks from wormhole is removed. However, the virtual cut-through still has the massive buffer needs of the store-and-forward strategy. As a result wormhole is preferred for switching strategy of NoCs [10].

Routing algorithms are organized into 2 categories: deterministic and adaptive. For a deterministic routing algorithmic program, the selected route relies on the source and destination, i.e. there's only one legal path between the pair of nodes. An adaptive algorithmic program permits more than one legal path between a pair of nodes and the route is decided on a per-hop basis. The selection of routing path is dynamically evaluated based on e.g. link congestion. As a result every router must be able to dynamically decide the routing direction; the quality of the router implementation will increase. The advantage of a routing algorithmic program is that it permits for higher link utilization, which may enhance the outcome considerably, particularly in case of high load. Flow control is a process used to verify as a message which moves along its route [7]. Flow control is primarily used to assure correct working of the network, however, it also enhances the usage of network resources and to offer an expected implementation. Assuring correct working of the network is primary and meanwhile keeping-away of deadlocks within the network is secondary. For various enhanced routing algorithms than XY-routing, deadlock issues will be resolved by initiating Virtual Channels (VCs).VCs are the fundamental approach used for ensuring deadlock flexibility in wormhole routed networks. A VC is formed by subdividing a physical channel into a group of logical independent channels by casting many buffers to the physical channel. To sort out, deadlocks VCs are accustomed to consistently break cyclic dependencies in the network. VCs may also be accustomed to enhance wire utilization, enhanced conduction and to produce Quality-of-Service (QoS). Two fundamental forms of QoS exist: best-effort services (BE) and warranted services (GS) [5]. in a BE NOC assures solely for correctness and execution of group action, whereas in a GS NOC performance assures like bandwidth and latency guarantees. The difficulty of a BE NOC is far apart than GS NOC.

## 2. METHODILOGY

Execution of synchronous systems on FPGAs is hypothetical therefore solely a simple BE NOC has been performed. In general, the major focus has been on accessibility, whereas execution has low priority. The accessible logic resources on the FPGA are restricted thus keeping the area low. The subsequent sections can justify the model selections created for the final NOC design. When selecting the topology, a topology that outlines well onto the architecture of the FPGA ought to be designated. The

entanglement of the topology ought to even below. A k-ary 2-cube mesh or torus with only one-directional links is well-being the successor. $A_k$-ary 2-cube network topology with two-way links is selected for the NoC. The primary reason for this is often that it's easier to ensure deadlock freedom in the network because of the huge range of links compared to a torus. Integrating with XY-routing deadlock freedom is assured without executing virtual channels. The 2-dimensional structure of the topology outlines well onto the architecture of the FPGA.A k-ary 2-cube network topology adds the subsequent needs to the interface of the router: there are four ports for network connections, one port for a core affiliation, and ports should be two-way.

**2.1 Routing**

When selecting the routing strategy significance has been given on accessibility and minimum space used. To cut back the buffer necessities wormhole routing is employed. Hence, every router is solely needed to buffer an individual t it. As result packets are allowed to contain the Associate range of its. Wormhole routing additionally has an advantage of reduced packet latency. To minimize the complications of the routers a source-routed strategy has been employed. Thus the issue of deciding successive hop is removed from the routers to the NAs. With source-routing, the routers can aid any routing rule in which the source is ready to work out the whole routing path to the receiver, but to make sure deadlock freedom it's prohibited to use the XY-routing algorithm. The network doesn't permit a packet to be routed back in the same direction as it arrives from. For the network topology that leaves an arriving packet at the router has four feasible routing directions. As a result solely 2 bits are needed to write the routing direction for every hop. The execution of virtual channels needs extra buffer capability and extra control circuitry and thereby enhances the complexness of the router. Casting virtual circuits to a BE NOC is completed for primarily 2 causes: deadlock avoidance and to enhance performance. Deadlock freedom is already assured by the selected routing rule and conduction doesn't have priority. Therefore virtual channels haven't been enforced.

**2.2. Handshake Channel**

The 4-phase bundled information handshaking protocol is employed. The handshaking protocol is extended with two further request signals that are used for encoding of its form. This can be explained thoroughly in the further section. All handshaking channels are push channels; therefore it is invariably the sender that initiates the transaction.

**2.2.1. Flit format**

A packet consists of many fits. To spot the start and therefore the finish of a packet 3 varieties of its are required:

- A header flit indicates the beginning of a packet and carries the routing information.
- The last flit is indicates, the last packet in the header forma
- Intermediate flits are all the flits in between the header it and also the finish it.

The header it detains the routing path and also the resultant its holds the packet data. Every packet comprises precisely one header it and one end flit. In between, zero or additional intermediate flits are allowed. Therefore a packet will contain any arbitrary range of its. The flit kind should be encoded into the flit. The traditional form of encoding the flit type is to encode the type as additional information bits. This is the approach used in MANGO [5] and QNoC [8]. In MANGO the BE router employs a single end-of-packet bit to point the flit form. The header it's recognized as the first it to approach after an end-of-packet it. In QNoC they require three it types similar to the categories explained previously and they are encoded in two bits. The flit sort is recognized by one of the three request forms: header request, intermediate request, and end request. By encryption in the form employing request signals the difficulties of the routers are reduced. The control circuits within the router are simplified, since the flit form does not have to be taken out from the information channel. The latency through the router is minimized as a result of the flit sort is known instantly, so it's not necessary to wait for the identification of it sort. The dimension of the acknowledgment channel isn't affected as because the additional request signal should have been routed as information signals anyway. Therefore, no further signals are required in the handshake channel. However the quality of the ordinary handshake elements will be high because of the extra request signals. Another drawback is that three delay components are required once a channel is delay matched [9].

**2.3 Core Interface**

The interface by which the cores hook up with the NA is termed as Core Interface (CI). Two distinct core interfaces are considered: Open Cores Protocol (OCP) [16] and wish bone [20]. The wishbone protocol is an open description provided by Open Cores [19]. Open Cores is an open source community for information processing cores. OCP is described by the OCP International Partnership that has members from a various electronics companies. The two descriptions are homogeneous and they both flit the aim of the project. The OCP protocol has been selected because of previous projects developed at IMM, DTU. This will permit utilization of elements developed in the earlier projects.

The following OCP commands laid out in the OCP specification should be supported by the AN:

- Easy write and skim transfer: scan and write requests sent by the master are accepted by the slave within the same clock cycle. The read response is distributed within the at once following clock cycle.
- Write with request handshake: The slave is allowed to delay the acceptance of the write requests by 'AN' discretion arrange of clock cycles.
- Read with request acknowledgment and separate response: The slave is allowed to delay the acceptance of the read request by a discretional range of clock cycles and therefore the slave is additionally allowed to delay the read response by an arbitrary number of clock cycles.

**Packet Format**

To support the specified OCP commands, there are three packet formats such as

1. Write Request packet.
2. Read Request packet.
3. Read Response packet.

The write request and read request, packets are transferred from the master cores to the slave cores and also the read response packet is distributed from the slave cores to the master cores. The flit size is ready to thirty-two bits since it is the dimension of the address and information signals of the OCP interface. If a smaller it size is chosen more flits are needed to send a packet and so the packet latency can be enhanced. However, the routing difficulty of the routers is minimized due to the narrower information signals. The routing data for the successive hop holds on two bits because of the two MSBs of the header it. Table one depicts the packet formats for the three packet forms as show in Table 1(a-c).

**Table 1 (a):** Format for write request packet

| Flit Name | Flit Type | Width | Description |
|---|---|---|---|
| header flit | header | 32 | Routing information |
| Control flit | immediate | 7 | MCmd and MByteEn |
| Addr flit | immediate | 32 | MAddr |
| Data flit | end | 32 | MData |

**Table 1 (b) :** Format for read request packet

| Flit Name | Flit Type | Width | Description |
|---|---|---|---|
| header flit | header | 32 | Routing information |
| Control flit | immediate | 7 | MCmd and MByteEn |
| Addr flit | immediate | 32 | MAddr |
| Data flit | end | 32 | Return routing path |

**Table 1 (c) :**Format for read response packet

| Flit Name | Flit Type | Width | Description |
|---|---|---|---|
| header flit | header | 32 | Routing information |
| Control flit | immediate | 2 | Sresp |
| Data flit | end | 32 | SData |

**2.3 Proposed Router Design**

The design of the router is extremely keen on the network within which it is used. The selection of topology, changing mechanisms, routing algorithm, and flow control mechanisms all affects the necessities of the router. The router should support a k-ary 2-cube mesh topology; consequently, it must offer five ports: four for connecting with different routers and one for connecting IP core. The links are duplex thus every port comprises of an input port and an output port. The router contains a non-blocking crossbar, i.e. each input port connected to any output port in any permutation. FIFO buffers are placed at the interface of each input ports and output ports.

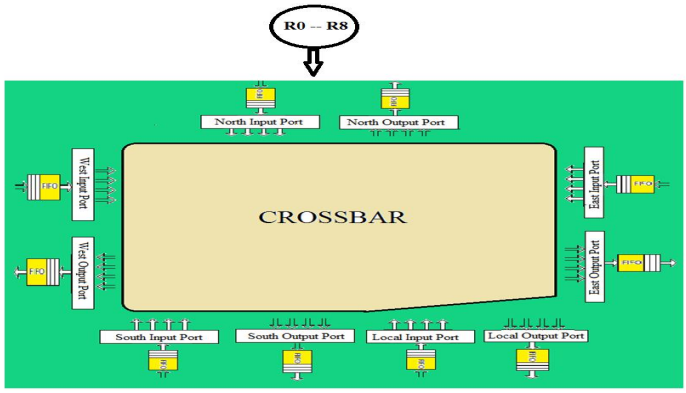


**Figure 2:** Proposed crossbar switch for NoC with five input ports and five output ports

The depths of the FIFO buffers are configurable. Fig. 2 depicts a diagram of the router. In the further units, the design of the input port, the output port, and also the FIFO buffers are conferred.

**2.4. Design of Input port**

The aim of the input port is to route the packet to the right output port. The input port has one input acknowledgment channel and 4 output handshake channels. The routing direction is controlled by a group of multiplexers. A diagram of the input port design is outlined in Fig .3. The header, intermediate, and end requests are represented as rh, ri, and re severally. The primary it arriving is the header it that contains the routing direction. The routing direction is held on in the two MSBs of the header it. The two routing direction bits are latched and employed as control inputs to the output multiplexers. The routing direction should be secured to a similar destination for all subsequent it's which belongs to the to the same packet. Therefore the latch is controlled by the header request signal such the latch is clear once rh is one. To assure that setup and hold times are not desecrated, the information validity theme for the input channel should be broad. Depending on the flit form the information signal should be treated distinctly. If it is an

intermediate or an end it, the information ought to be passed through untouched. If it is a header it, the information should be rotated two bits. The rotation is done by a rotate element and a multiplexer is employed to modify the two information signals. To retain broad information validity scheme the information multiplexer is controlled by a little feedback circuit comprising of a C-element and an OR gate, with the header request signal and the acknowledge signal are given as inputs. In the case of a header it the multiplexer chooses the rotated information signal and keeps the selection for the entire acknowledgment cycle. Delay components should be inserted on all three request channels. The delay components on the intermediate and the end request signal should match the delay of the information multiplexer deducted by the delay of the request de-multiplexer. Therefore the matched delay is small. The delay component on the header request signal should delay the request signal, till the control signal for the request de-multiplexer is stable. If the delay is not sufficiently large a flaw may appear on one among the header request output signals. The delay may be extensive enough for the rotation and multiplexing of the information signal. Consequently, the delay component for the header request signal must be larger than the other two.

### 2.4.1 Design of Output port

The output port consists of four input channels and one output channel. The output port should intercede between competitor inputs, such that one input channel is granted access to the output channel at a time. Once the associate input channel has gained access, it should keep exclusive access until the entire packet has been transmitted. The completion of a packet is specified by the receiving of end it. A combined part is employed to combine the four input channels onto the output channel. A diagram of the output port is outlined in Fig.4.The arbitration is handled by a group of access control circuits and a 4-input Mutex component. Because of the flit forms are encoded employing request signals the arbitration between competitor inputs is often drained a straightforward method. Each input channel has an associated access feedback circuit. When an access-control circuit receives a header request, it'll request the mutex for access to the output port. Once access is granted by the mutex the header it is passed through to the output. The mutex is not discharged before end it is received. Other competitive inputs will wait silently, with a declared header request signal, for the mutex to grant them access to the output channel. The access feedback circuit is given by the STG depicted in Fig .5. The header, intermediate, and end request signals are denoted rh, ri, and re respectively. The mutex request and grant signals are represented by m_req and m_grant. The fairness of the arbitration is decided by the mutex part. Delay components are inserted in the request signals on the output channel. The delay components should match the delay that the information signals expertise in the combined part deducted by the delay through the access feedback circuit.

The merge component has four input channels and one output channels. It relays handshakes from the input channels to the output channel. It is assumed that input requests are mutually absolute. The design of the combined element is shown in Fig.6.The model is predicated on the normal combined design bestowed in [22], however, it's been changed to support the three-request acknowledgment channel. For every input channel, the three request signals should be OR'ed . The output of the OR gate and also the output ack signal is employed with the input ack signal employing a C-element. The added overhead for the 4-input combined element to support the extra request signals is four 3-input OR gates and two 4-inputOR gates. To support broad information validity the request signals are OR'ed with the acknowledge signals. This ensures that the information multiplexer chooses the active input for the complete acknowledgment cycle.A 4-input mutex element is required for the output port design. A 4-input mutex may be formed by combining many 2-input mutex elements. QNoC [23] uses a 4-input mutex element and their design is additionally used in this project. The 4-input mutex element consists of six 2-input mutex components organized in three stages. In [23] an study of the fairness of the design is carried out. The proof that the mutex contains a delimited obstruction time and an request may be run by no more than two later requests. The proof assumes that the 2-input mutex elements are truthful. Even though the mutex won't preserve the first ordering in all cases, it is thought of to be truthful enough for the objective of this project, and assuring fairness is not a key issue.

### 2.5 Design of FIFO for efficient NoC router

FIFO buffers are inserted at each input and output port. The FIFO is designed, in the regular way, as a chain of handshake latches. This is shown in Fig.8.

Handshake latches for the 4-phase bundled data protocol can be designed in three distinct ways, depending on how strong the coupling is between the input channel and the output channel [24]: Un-decoupled, Semi-decoupled, and fully-decoupled. The choice between the different types is a tradeoff between complexity and performance. The un-decoupled latch controller is least complex. It does not permit latching of new data before the previous handshake cycle has finished entirely, i.e. it must wait for $Ack_{out}$ signal to become 0. In other words, it must wait for the superfluous return-to-zero phase of the handshake to complete. There exists a strict instruction between the handshaking on the input channel and the output channel: When $Req_{out}$ is one and is less than $ak_{in}$, the $ak_{in}$ also goes high and when $Req_{out}$ goes low and is less then $Ack_{in.}$ The $Ack_{in}$ also goes low. During the return-to-zero phase, the latch is transparent. Consequently, soley second latch in a FIFO will hold the correct data. It is said to have a Static spread of 2. Also, due to dependencies

with non-neighboring stages in the FIFO, it is unable to take favor of an asynchronous delay element and their request and acknowledge is depicted in Fig.8When designing the NA the placement of the crossing between the synchronous and asynchronous domain is very essential. The number of clock synchronization in a design should be kept very low for two main reasons: speed and reliability. Each time a signal is passed through a synchronizer and it takes two clock cycles (for a two-IP-op synchronizer) and thereby reduces the speed. The possibility of meta-stability can never be removed completely, thus the possibility of failure will always persist. The failure rate will enhance with the number of synchronizations executed. The only way to reduce the possibility of meta-stability is to enhance the latency by adding more IP-ops in the synchronizer. Therefore the NA is designed so it is only necessary to perform synchronization one time per packet transmission.

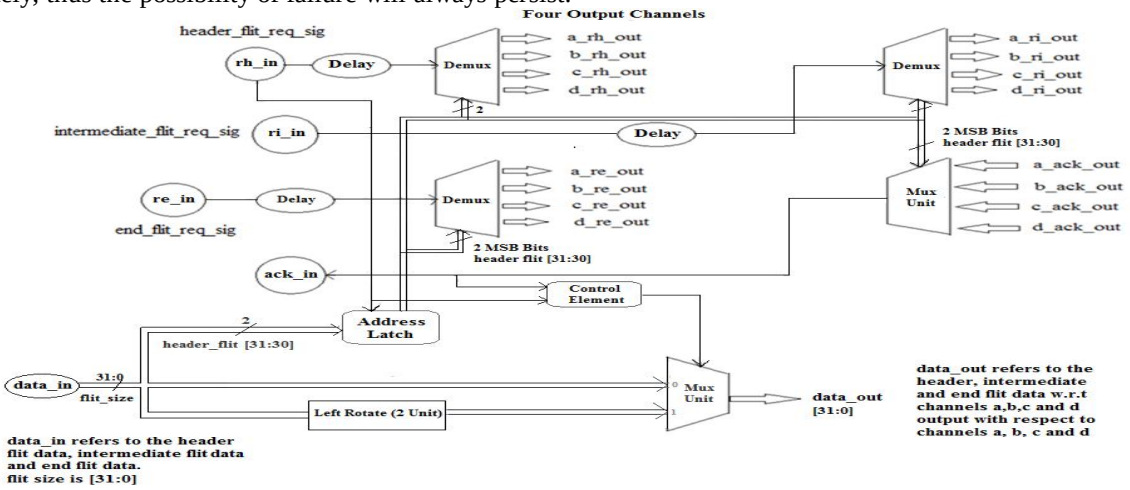


**Figure 3:** Proposed Input port design with delay, control signals and data transmission

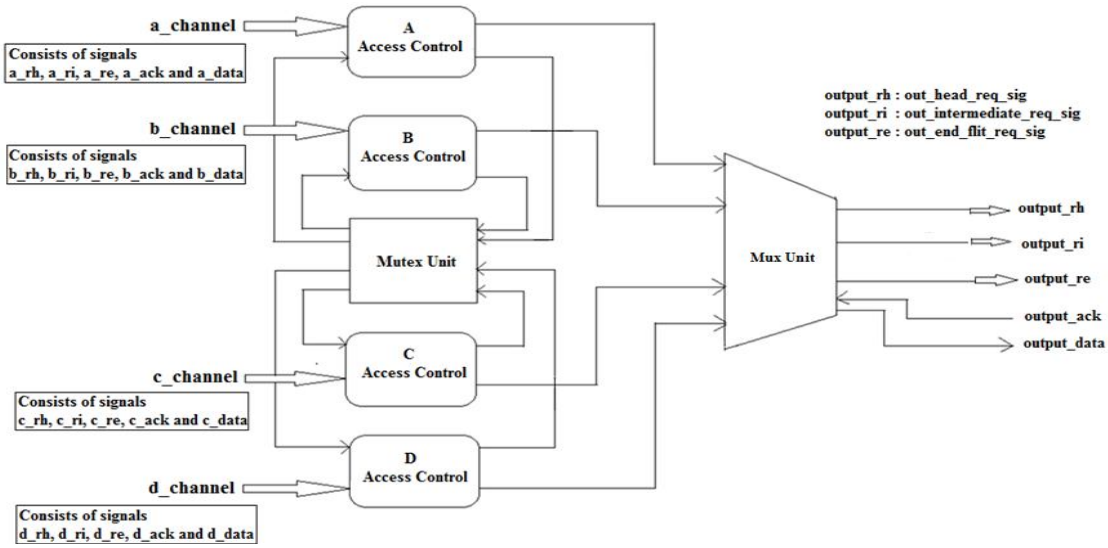


**Figure 4:** Proposed output port design with access control signals, Mutex and data transmission

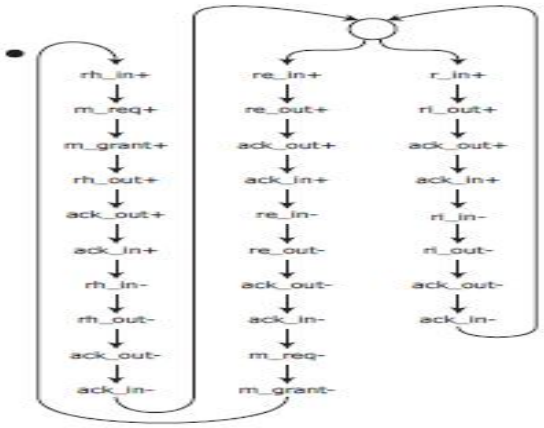


**Figure 5:** Specifications STG and its access control signals

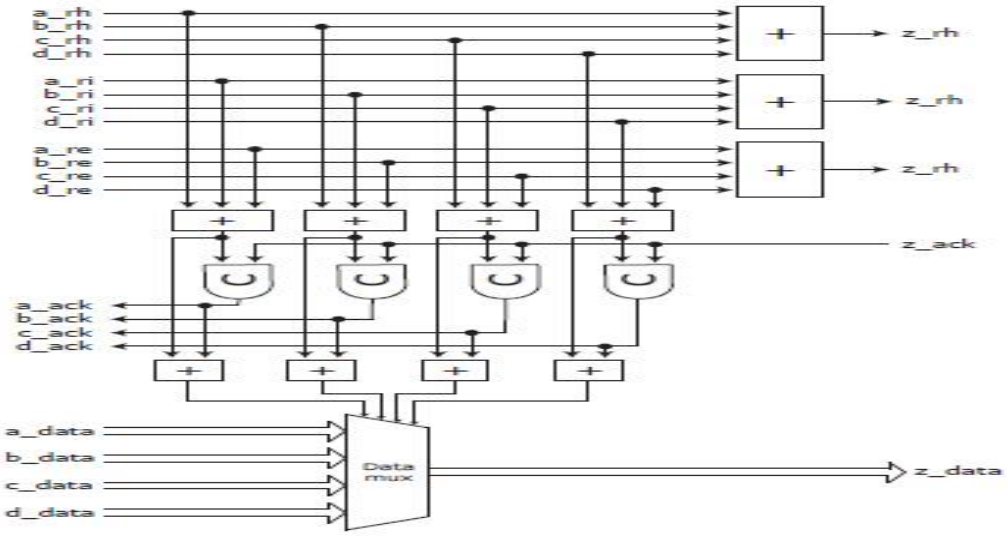


**Figure 6:** Logical diagram of the merge component and signal directions

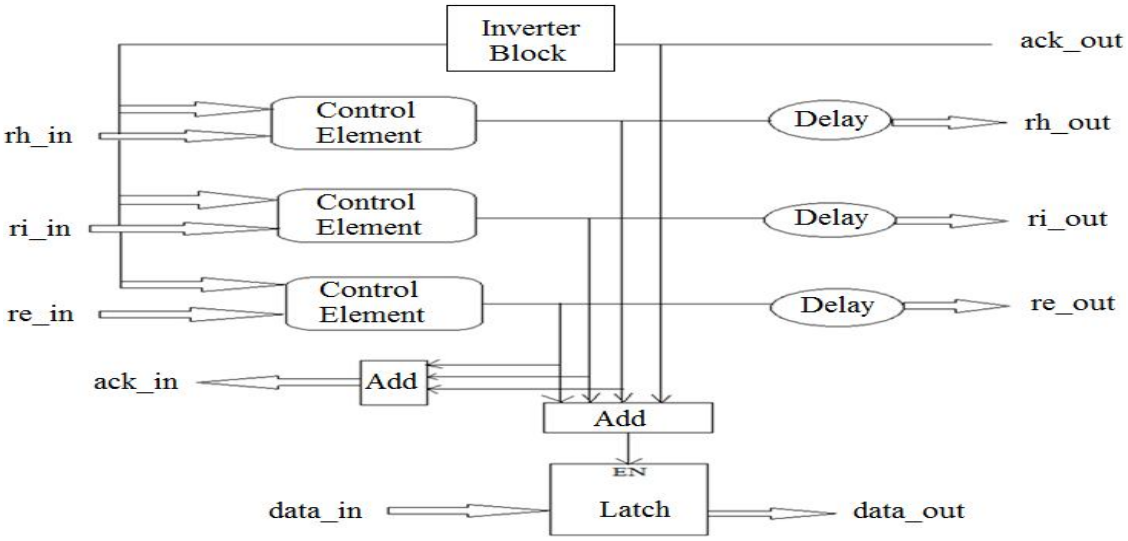


**Figure 7:** Design of the proposed FIFO along with control elements and delays

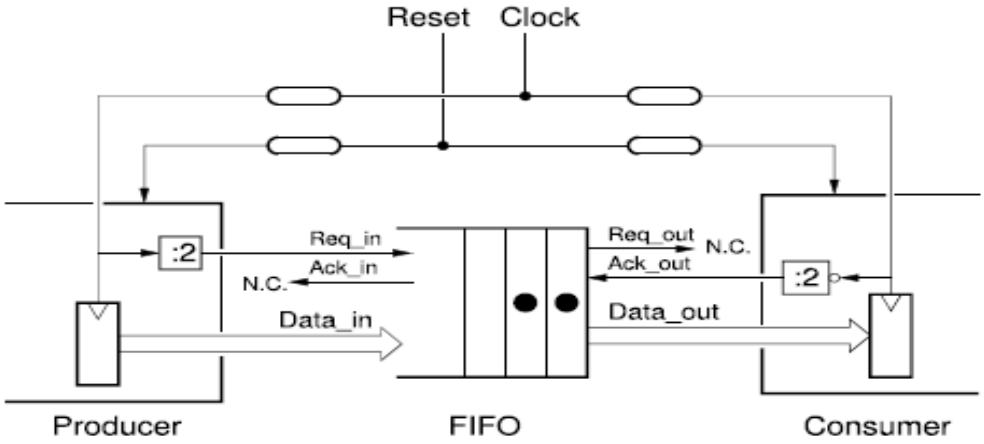


**Figure 8:** Proposed FIFO with Request & Acknowledge

FIFO buffers are inserted at each input and output port. The FIFO is designed, in a regular way, as a chain of handshake latches. This is shown in Fig.7. For testing functions, a straightforward traffic generator has been designed. The design comprises of a traffic supply and a traffic sink. The traffic source is able to trader a predefined set of packets from ROM. The traffic sink scans the received packets into ROM. it is then probable to match the input path with the output trail to verify correct performance. The representation of the traffic source is depicted in Fig.9. Every entry in the ROM comprises of it form and it information. The handshaking is not done by employing a simple repeater circuit that comprises of one NOR gate. Asynchronous counter clocked on the acknowledge signal is employed to increment the address input of the ROM. The Read Only Memory is clocked with the un-delayed request signal. For proper loading of the Read only Memory output, the clock input is gated with the reset signal. The flit form is employed to manage a de-multiplexer to yield the correct request type. The traffic sink must scan the received packets and store them in read-only memory. When the entire input path has been received the sink read-only memory information must be scanned out of the FPGA. This can be doable through the JTAG interface; However Xilinx does not offer any straightforward tools that may do this precisely. Manual communication with the JTAG interface is needed to extract the information. Fortunately, the ChipScope tool provided by Xilinx [25] can be utilized to scan the data into the read-only memory and extract it later. In other words,it can almost build the entire sink element. Chip Scope is advanced logic analyzer tool for Xilinx FPGAs

It provides cores that can keep-track of information and store information traces of any signal within the FPGA throughout the runtime. The ChipScope software package is employed to extract and show the information captured by the cores. The ChipScope software has Graphic user interface and is comparatively simple to perform. For the sink design the VIO (Virtual Input/Output) ChipScope core is employed to capture information. The core captures the information signal on the falling edge of the request signal. Once the interior storage of the VIO core is stuffed, the information is transmitted to the ChipScope software package. The design is outlined in Fig. 11. The VIO core is required for every signal that has to be monitored in the design

### 2.6. Traffic Generator Design

For testing functions, a straightforward traffic generator has been designed. The design consists of a traffic supply and a traffic sink. The traffic supply is in a position to transmit a predefined set of packets from fixed storage. The traffic sink reads the received packets into fixed storage. it's then potential to match the input path with the output trail to verify correct operation. The planning of the traffic supply is shown in Fig.9. Every entry in the ROM consists of its kind and it knowledge. The shake is not finished employing a simple repeater circuit that consists of one NOR gate. Asynchronous counter clocked on the acknowledge signal is employed to increment the address input of the fixed storage. The fixed storage is clocked with the un-delayed request signal. For correct low-level formatting of the fixed storage output, the clock input is gated with the reset signal. The flit type is used to control a de-multiplexer to output the correct request type. The traffic sink must read the received packets and store them in a ROM. When the complete input trail has been received the sink ROM data must be read out of the FPGA. This is possible through the JTAG interface; however, Xilinx does not provide any simple tools that can do that directly. Manual communication with the JTAG interface is required to extract the data. Fortunately, the ChipScope tool provided by Xilinx [26] can be used to read the data into the ROM and extract it afterward. In other words, it can almost build a complete sink component. ChipScope is a complex logic analyzer tool for Xilinx FPGAs. The flit sort is employed to manage a

de-multiplexer to output the correct request type. The traffic sink should browse the received packets and store them in exceedingly read-only memory. When the complete input path has been received the sink read-only memory knowledge should browse out of the FPGA. this can be doable through the JTAG interface, but Xilinx does not offer any straightforward tools that may do this directly. Manual communication with the JTAG interface is needed to extract the info. Fortunately, the ChipScope tool provided by Xilinx [26] will be wont to browse the data into the read-only memory and extract it later. In different words, it will almost build the whole sink part. ChipScope may be advanced logic analyzer tool for Xilinx FPGAs

It provides cores that can monitor and store data traces of any signal in the FPGA during runtime. The ChipScope software is used to extract and display the data captured by the cores. The ChipScope software has a GUI interface and is relatively easy to operate. For the sink design the VIO (Virtual Input/Output) ChipScope core is used to capture data. The core captures the data on the data signal on the falling edge of the request signal. When the internal storage of the VIO core is filled, the data is transmitted to the ChipScope software. The design is shown in Fig. 11. The VIO core is needed for each signal that must be monitored in the design. It provides cores that may monitor and store information traces of any signal within the FPGA throughout the runtime. The ChipScope software package is employed to extract and show the info captured by the cores. The ChipScope software package features a user interface and is comparatively simple to control. For the sink style the VIO (Virtual Input/Output) ChipScope core is used to capture information. The core captures the info on the data signal on the falling edge of the request signal. Once the interior storage of the VIO core is stuffed, the data is transmitted to the ChipScope software package. The look is shown in Fig. 8. The VIO core is required for every signal that has to be monitored in the design.

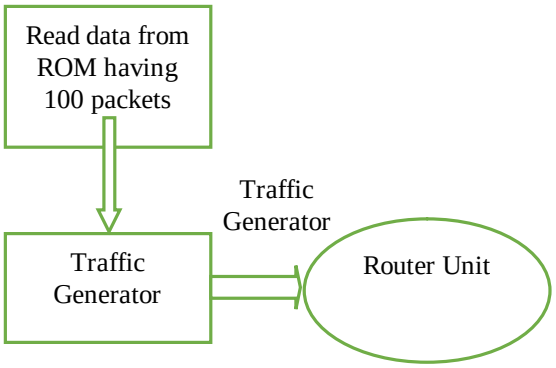


**Figure 9:** Block diagram for traffic generator for testing of 100 packets

The Chip Scope VIO core that is employed in the sink to gather the information and can be placed in the design in two ways. The ChipScope software provokes the VIO cores which may be instantiated by traditional approach. ChipScopecan additionally place the cores in the design by inserting them in the synthesized netlist mechanically. The last technique could be easier since no modifications are required in the Verilog HDL. However, the last technique cannot be employed in this case. The synthesizer can take away the information signals from the handshake channel because they do not seem to be connected to anything in the traffic sink, so they will not be in the synthesized netlist. AS a result the VIO cores are instantiated manually in Verilog HDL. Together with the VIO core, an ICON control core is also inserted in the model. The ICON core manages the communication with the ChipScope software over the JTAG interface. Also, unwanted optimizations by the planning tools have been problematic. To optimally delay match circuit the falling edge of the request signal is employed as the clock input for the VIO core. This signal must be routed on the allocated clock nets for the core to operate. This is done by placing a clock buffer (BUFG) on the signal. The Traffic Sink consists a delay component to delay the acknowledge signal. If this delay is not sufficiently large the VIO core will not conduct properly. A size of 10 is found to work. The ChipScope documentation says it supports frequencies of up to 100 MHz, thus a delay is not needed. It is expected that the VIO core fails because it predicts a real clock signal but it is clocked with the request signal that does not have a regular period. The ChipScope cores consists of a bug, hence they will not work with bus width larger than 16. The error only occurs with some designs and is not officially recognized by Xilinx. The bug results in a DRC error during the mapping process. In another post in the Xilinx Community Forums, the same drawback is reported [27]. It has not been possible to determine the exact source of the error.

## 3.RESULTS AND DISCUSSION

The implementation of the router is divided into 9 Verilog entities which follows the structure of the design and its direction is follows as per below encoding headers

The encoding of the routing direction used in the header it is the following:
North = "00"
East = "01"
South = "10"
West = "11"
As mentioned in the design, the local port is reached by routing it back in the same direction it came from. The area utilization of each component is listed in Table 1. The number of utilized LUTs is excluding delay elements. Thus the total area utilization of the router is 1295 LUTs and 330 latches. The percentage of the LUTs that are used for delay matching is 29%.

**Table 1:** List of Area utilization of each component

| Parameter | Proposed | Existing | Improvement |
|---|---|---|---|
| Slice Register | 390 | 560 | 17% |
| Slice LUT's | 660 | 1200 | 56% |
| LUT Flip-Flop Pairs used | 665 | 2301 | 62% |
| Delay | 0.871ns | 2.45ns | 47% |
| Power | 0.084Watts | 1.84Watts | 25% |
| Frequency | 574.229MHz | 342MHz | 21% |
| Throughput | 20.2GHz | 672MHz | 75% |
| Latency | 15*0.871ns= 13.06ns | 18.23ns | 12% |

Throughput $= \frac{\text{Frequency of operation}}{\text{delay multiplied by input size}} = \frac{574.229MHz}{0.871ns*32} = 20.2GHz$

Latency= No of clock cycles required to complete the operation and its delay = 13.06ns

The route lookup table in the Master NA is implemented as a ROM using the Block RAM resources available on the FPGA. Block RAM can be included in two ways: inferred by HDL or instantiated as an IP core generated by the Xilinx Core Generator. When the ROM is inferred by HDL it is much easier to change the content of the ROM.

## 4.CONCLUSION

The objective of the work has been to implement an asynchronous NoC paradigm on a customary FPGA. The previous work concerning about implementing asynchronous circuits on FPGAs is incredibly restricted therefore a significant part of the project has been to develop a general design flow for the implementation of asynchronous circuits on FPGAs. A simple asynchronous best-effort NoC has been developed. The NoC comprises of a router, a slave NA, and a master NA. The router is intended to design in a mesh topology and employs wormhole routing. Deadlock freedom is assured by using XY-routing and supply routing is employed. A packet will contain an infinite number of its. to spot the start and stop of a packet three it forms are employed. Then it type is encoded by addition of two extra request signals to the handshake channel. The NAS contributes an OCP interface for the cores to connect to the network. Synchronization is managed by employing a simple two IP-op synchronizer. the realm usage of the router is 1295 LUTs and 330 latches where9% of the LUTs are employed by delay components. The overall output of the router has been measured to be 43 MHz. Small multi-processor prototypes utilizing the asynchronous NOC have been developed. The prototype comprises of three CPUs and three peripheral units which are connected by a 3x2 mesh. To assure that the system is free from message dependent deadlocks a separate request and response web is employed. It has not been attainable to flit a larger design on the FPGA. it is demonstrated to be hard to achieve high utilization of the logic resources on the FPGA. It is suspected to ensure exhaustion of the routing resources. it is unclear if the implementation of the asynchronous NoC is because of these problems. There seems to be a general drawback of reaching high utilization Figures for the FPGA that has been used for the prototype. However, there are indications that the implementation of the asynchronous NoC will enhance this drawback. The basic drawbacks for implementing asynchronous circuits on FPGAs are the delay matching method. Also, uncertain optimizations by the design tools have been complicated. To optimally delay match circuit the certainty of the delay component and also the data path should each be high. By employing relative placement constraints in the design of the delay component satisfying predictability is achieved. To scale back the variations in the delay of the data path it is been tried to form macros with locked placement of the design primitives. Due to unresolved issues with the look tools it's not been possible to make such macros. As a consequence, it's required to feature further delay during delay matching. It's unattainable to show off the logical optimizations performed by the design tools. They can be somewhat controlled by the utilization of various settings and constraints. For rigorously designed circuits like the circuits synthesized by Petrify are having the consequence that it is necessary to do the LUT mapping and placement manually. For alternative circuits, it is not proven to be an oversized issue.

## REFERENCES

[1]. Mehdi Modarressi.et.al, "A Hybrid Packet-Circuit Switched On-Chip Network Based on SDM", 978-3-9810801-5-5/DATE09, 2009 EDAA.

[2]. Shashi Kuma.et.al, "A Network on Chip Architecture and Design Methodology", Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI'02), 0-7695-1486-3/02,2002 IEEE.

[3]. Pejman Lotfi-Kamran.et.al, "An Efficient Hybrid-Switched Network-on-Chip for Chip Multiprocessors", IEEE Transactions on Computers, vol. 65, no. 5, pp. 1656–1662, May 2016. https://doi.org/10.1109/TC.2015.2449846

[4]. Kang G. Shin.at.al, "Analysis and Implementation of Hybrid Switching", IEEE TRANSACTIONS ON COMPUTERS, VOL. 45, NO. 6, JUNE 1996. https://doi.org/10.1109/12.506424

[5]. Michael Opoku Agyeman.et.al, "Extending the Performance of Hybrid NoCs beyond the Limitations of Network Heterogeneity", Journal of Low Power Electronics and Applications, Appl. 2017, 7, 8;doi:10.3390/jlpea7020008.

[6]. Mohamad FallahRad.et.al, "2016 Euromicro Conference on Digital System Design", 978-1-5090-2817-7/16, 2016 IEEE, DOI 10.1109/DSD.2016.87.
[7]. Ashkan Aghdai.et.al, "Design of a Hybrid Modular Switch", arXiv:1705.09999v1 [cs.NI] 28 May 2017. https://doi.org/10.1109/NFV-SDN.2017.8169825
[8]. Arnab Kumar Biswas, "Efficient Timing Channel Protection for Hybrid (Packet/Circuit-Switched) Network-on-Chip", IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 0, NO. 0, OCTOBER 2017.
[9]. P.Ezhumalai.at.al, "High Performance Hybr id Two Layer Router Architecture for FPGAs Using Network-On- Chip", (IJCSIS) International Journal of Computer Science and Information Security, Vol. 7, No. 1, 2010
[10]. K. Paramasivam, "NETWORK ON-CHIP AND ITS RESEARCH CHALLENGES",ICTACT JOURNAL ON MICROELECTRONICS, JULY 2015, VOLUME: 01, ISSUE: 02 https://doi.org/10.21917/ijme.2015.0015
[11]. Wen-Chung Tsai.et.al, "Networks on Chips: Structure and DesignMethodologies", Hindawi Publishing Corporation, Journal of Electrical and Computer Engineering Volume 2012, Article ID 509465, 15 pages, doi:10.1155/2012/509465
[12]. Mikkel B.et.al, "ReNoC: A Network-on-Chip Architecture with Reconfigurable Topology", Second ACM/IEEE International Symposium on Networks-on-Chip, 978-0-7695-3098-7/08, 2008 IEEE, DOI 10.1109/NOCS.2008.13.
[13]. Angelo Kuti Lusala.et.al, "Combining SDM-Based Circuit Switching with Packet Switching in a Router for On-Chip Networks", Hindawi Publishing Corporation International Journal of Reconfigurable Computing, Volume 2012, Article ID 474765, 16 pages, doi:10.1155/2012/474765
[14]. William J, et.al, "Route Packets, Not Wires: On-Chip Interconnection Networks", DAC 2001, June 18-22, 2001, Las Vegas, Nevada, USA.Copyright 2001 ACM 1-58113-297-2/01/0006
[15]. Tue Strjer Lyster and Morten Briand Thomsen. Project in Asynchronous Systems. IMM/DTU, 2004.
[16]. OCP International Partnership. Open Core Protocol Specification. Release 2.0
[17]. Rasmus Grndahl Olsen. Ocp based adapter for network-on-chip. Master's thesis, Technical University of Denmark, 2005.
[18]. Open Verilog International. Standard Delay Format Specification, 3.0 edition, 1995.
[19]. Opencores. http://www.opencores.org/.
[20]. Jan M. Rabaey, Anantha Chandrakasan, and Borivoje Nikolic. Digital Integrated Circuits { A Design Perspective. Prentice Hall, 2nd edition, 2003
[21]. Morten Sleth Rasmussen, Christian Place Pedersen, and Matthias Bo Stuart. Asynchronous Circuits on FPGAs. IMM/DTU, 2005.
[22]. Morten Sleth Rasmussen, Christian Place Pedersen, and Matthias Bo Stuart. A noc-based soc executing a ray tracer, using synchronous multiprocessing, 2005. IMM, DTU. Polyteknisk Midtvejs Projekt
[23]. D. Rostislav, V. Vishnyakov, E. Friedman, and R. Ginosar. An asyn- chronous router for multiple service levels networks on chip. Asynchronous Circuits and Systems, 2005. ASYNC 2005. Proceedings. 11th IEEE International Symposium on, pages 44{53, 14-16 March 2005
[24]. Jens Spars. Asynchronous Circuit Design { A Tutorial. Techinical University of Denmark, 2006
[25]. Jens Spars. Course 02204 design of asynchronous circuits { lecture slides },2007. Lecture 8.
[26]. Xilinx. ChipScope Pro Software and Cores User Guide, 2007. Version 9.2.
[27]. Problems in using chipscope cdcle. http://forums.xilinx.com. Thread from the Xilinx Community Forums.