ENGINEERING IN ADVANCED RESEARCH SCIENCE AND TECHNOLOGY

ISSN 2278-2566
Vol.01, Issue.03
June -2019
Pages: -141-151

SECURED ADAPTIVE CONGESTION-AWARE ROUTING ALGORITHM FOR MESH NETWORK-ON-CHIP PLATFORM

1. D.NAGARAJU, 2. K.V.N.SWETHA
1. PG Scholar, Dept of ECE, ABR College of Engineering and Technology, Kanigiri, A.P
2. ASSISTANT PROFESSOR, Dept of ECE, ABR College of Engineering and Technology, Kanigiri, A.P

ABSTRACT:
Network-On-Chip (NoC) has surpassed the traditional bus based on-chip communication in offering better performance for data transfers among many processing, peripheral and other cores of high performance embedded systems. Adaptive routing provides an effective way of efficient on-chip communication among NoC cores. The message routing efficiency can further improve the performance of NoC based embedded systems on a chip. Congestion awareness has been applied to adaptive routing for achieving better data throughput and latency. This thesis presents a novel approach of analyzing congestion to improve NoC throughput by improving packet allocation in NoC routers. The routers would have the knowledge of the traffic conditions around themselves by utilizing the congestion information. We employ header flits to store the congestion information that does not require any additional communication links between the routers. By prioritizing data packets that are likely to suffer the worst congestion would improve overall NoC data transfer latency. Further, this project is enhanced by using more secured algorithm like Scalable Encryption technique.

INTRODUCTION: Embedded systems on chip in the recent decade have grown substantially utilizing many cores on a single chip known as embedded System-on-a-chip (SoC). It is important that data within the SoC, all cores have access to the desire resources while maintaining a data load balanced transmission links. Traditional bus-based interconnection has been able to allow high speed data transfers within a small SoC. However as the SoC scales up in size in today‟s most demanding applications, NoC is proven to provide better balance between traffic loads and data access for every core. [1] NoCs are expandable to allow communication for very large SoCs. It is not uncommon to see more than 256 cores in a NoC system, which is impossible to handle on a bus-based system. NoC based systems is a new strategy for data communication within the SoC. NoC performance depends on topology, data link width, traffic patterns, routing mechanisms and router arbitration. These parameters can be prioritized to improve performance, area of the design layout or power dissipation of the SoC. Depending on the chosen design requirement, the connections between routers and cores of a NoC can differ significantly. 2 NoC performance is important in high data throughput applications. It is ideal to reduce congestion and latency for all the data transfers within the NoC. Unfortunately, by increasing the number of cores within the NoC, congestion increases proportionally to the NoC size. Therefore, it is necessary to manage traffic effectively for reducing congestion within the NoC to achieve the best performance as the NoC grow in size. Design and scalability issues associated with increasing core counts on Chip Multi-Processors (CMPs) is a prominent research domain in computer architecture over the last decade. Communication among cores in these CMPs housing processors, caches and memory controllers is an important task that requires deeper exploration for better performance and throughput. Thus designing a scalable interconnect is critical for future energy efficient CMP designs. Interconnects like bus and crossbars are no longer scalable with these ever growing trend in CMPs. Hence, researchers have moved towards Network-onChip (NoC), a scalable, packet switched and distributed interconnect framework that offer much lower latency and higher bandwidth than their traditional

bus based counter parts. Most modern CMPs are arranged in 2D mesh topology due to its simple layout and short wires. Our approach here is to design a variable hardware router code by using Verilog and the same to be implemented for the SOC (System On Chip) level router. In this paper we are making a VLSI design for the implementation at the synthesizable level the same can be further enhanced to SOC level, but our main aim is limited to the NetList generation level which would give the result prediction and workable module vision. Our focus being in this is to make this router as much variable as we can which will give the robustness for the design to be called even as a Robust Router in which we can make the same router to not only go for N number of connections but also to detect all variety of packets and route the same. To do so we have to add the code with specific case"s for every type of packets we want to add to our router to route, with this paper of hardware code our approach is to get the basic packets routing with multiple protocols starting with the IPv4 and IPv6 The challenge of the verifying a large design is growing exponentially. There is a need to define new methods that makes functional verification easy. Several strategies in the recent years have been proposed to achieve good functional verification with less effort. Recent advancement towards this goal is methodologies. The methodology defines a skeleton over which one can add flesh and skin to their requirements to achieve functional verification. This project is aimed at building a reusable test bench for verifying Router Protocol Verilog. In this document the use Verilog to verify a design and to develop a reusable Test bench is explained in step by step as defined by verification principles and methodology. The Test Bench contains different components and each component is again composed of subcomponents, these components and subcomponents can be reused for the future projects as long as the interface is same. The Report is organized as two major portions; first part is brief introduction and history of the functional verification which tells about different technologies, strategies and methodologies used today for verification. Literature survey will contain an organized collection of data from different sources and significant changes that took place in the verification and design.. Second part is verification plan specifying the verification requirements and approaches to attack the problem, architecture of the test bench gives complete description about the components and sub components used to achieve the verification goals and also explains about improvements made in the design of the Router, test plan identifies all the test case required to meet the goals and finally results of the project. Four Port Network Router has a one input port from which the packet enters. It has four output ports where the packet is driven out. Packet contains 3 parts. They are Header, data and frame check sequence. Packet width is 8 bits and the length of the packet can be between 1 byte to 63 bytes. Packet header contains three fields DA and length. Destination address (DA) of the packet is of 8 bits. The switch drives the packet to respective ports based on this destination address of the packets. Each output port has 8-bit unique port address. If the destination address of the packet matches the port address, then switch drives the packet to the output port, Length of the data is of 8 bits and from 0 to 63. Length is measured in terms of bytes. Data should be in terms of bytes and can take anything. Frame check sequence contains the security check of the packet. It is calculated over the header and data. A data packet is typically passed from router to router through the networks of the Internet until it gets to its destination computer. Routers also perform other tasks such as translating the data transmission protocol of the packet to the appropriate protocol of the next network.

NOC BASICS: As the number of IP modules in Systems-on-Chip (SoCs) increases, bus-based interconnection architectures may prevent these systems to meet the performance required by many applications. For systems with intensive parallel communication requirements buses may not provide the required bandwidth, latency, and power consumption. A solution for such a communication bottleneck is the use of an embedded switching network, called Network-on-Chip (NoC), to interconnect the IP modules in SoCs. NoCs design space is considerably larger when compared to a bus-based solution, as different routing and arbitration strategies can be implemented as well as different organizations of the communication infrastructure. In addition, NoCs have an inherent redundancy that helps tolerate faults and deal with communication bottlenecks. This enables the SoC designer to find suitable solutions for different system characteristics and constraints. A network-on-chip is composed of three main building blocks. The first and most important one are the links that physically connect the nodes and actually implement the communication. The second block is the router, which implements the communication protocol (the descentralized logic behind the communication protocol). One can see the NoC as an evolution of the segmented busses where the router plays the role of a "much smarter buffer" (Bjerregaard and Mahadevan 2006). The router basically receives packets from the

shared links and, according to the address informed in each packet, it forwards the packet to the core attached to it or to another shared link. The protocol itself consists of a set of policies defined during the design (and implemented within the router) to handle common situations during the transmission of a packet, such as, having two or more packets arriving at the same time or disputing the same channel, avoiding deadlock and livelock situations, reducing the communication latency, increasing the throughput, etc. The last building block is the network adapter (NA) or network interface (NI). This block makes the logic connection between the IP cores and the network, since each IP may have a distinct interface protocol with respect to the network. Topology is the interconnection structure of the NoC that consists of routers, links and cores. There are many possibly topologies for NoC including mesh, torus, ring, star, cube and etc. [7,8]. NoC topology can expand into many dimensions. Each topology has its own advantages and disadvantages such as flexibility, reliability, number of links, routing latency and complexity. Performance would differ depending on topology since they are connected differently with various numbers of links [7]. A simple topology such as ring topology has very 7 simple routing mechanics but the number of routers is limited since latency will increase substantially if too many routers are added to the NoC. Mesh topology is generally chosen as it has many benefits such as multiple paths to every router, supports adaptive routing and fits many applications. Figure shows how a star topology is organized. This topology designates the centre router R0 as the main router. If router R0 is congested, it affects all router to router transfers but allows local cores within the slave routers R1-R5 to continue functioning. Each slave routers can be further expanded into branches but in the case where cores need to communicate with other branches, significant congestion would be created. The worst case scenario is when R0 fails as there is no backup link causing the system to total fail. Figure 2.2 illustrates a ring topology NoC. Unlike a start topology, the ring topology has two directions to route its data. This topology handles fault tolerance better than a star topology. The ring topology still suffers with the congestion issues. In the case where a router is congested, it creates a backlog of data holding up all data transfers behind it. The mesh topology resolves these issues with many alternative links between routers allowing multiple paths from a source to destination.

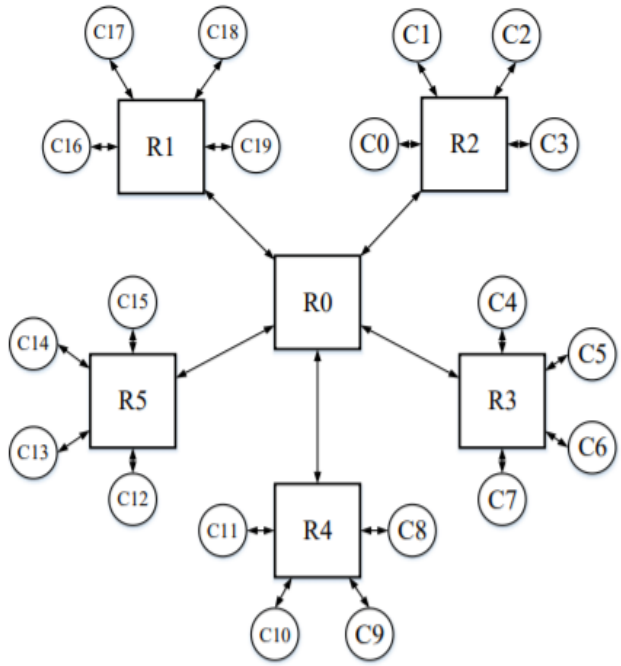

Figure - Star Topology NOC

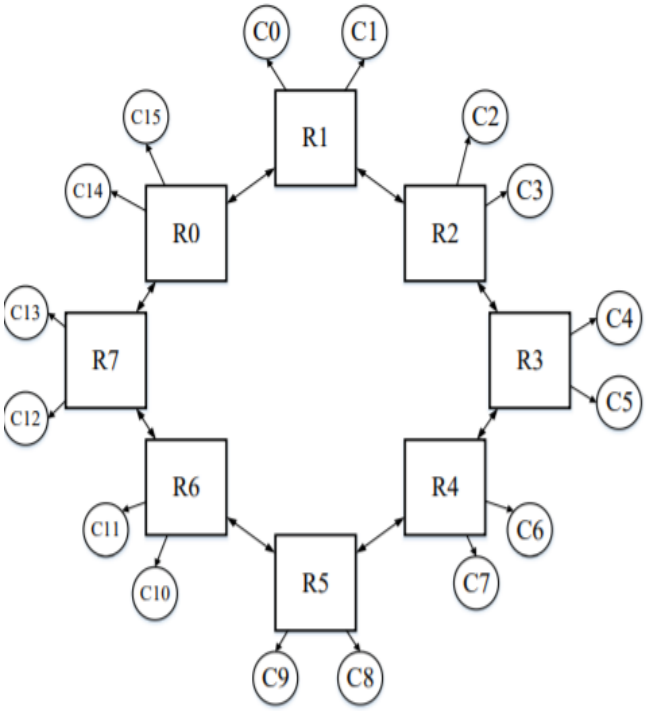

Figure- Ring Topology NoC

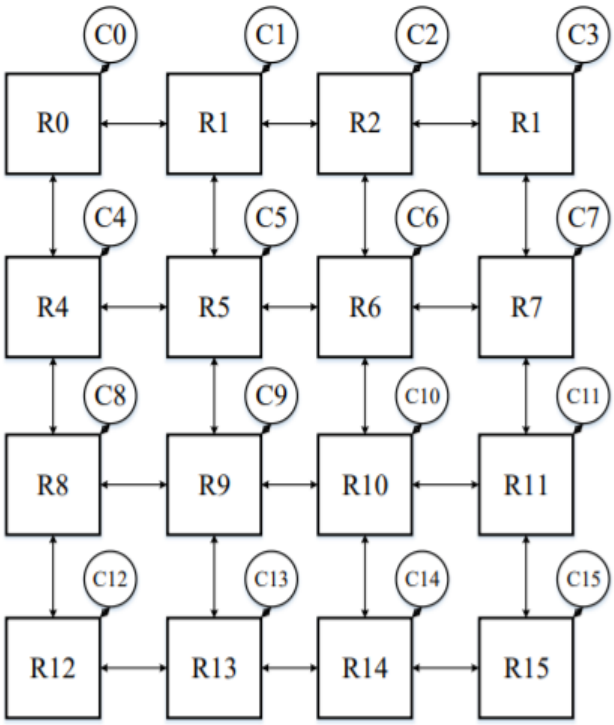

Figure - Mesh Topology NoC

Figure shows a typical 4x4 mesh topology NoC. Please note the difference between mesh and the previously illustrated topologies where each router only has a single connection to its core. This improves performance by reducing demand on each router and allowing multiple paths to a destination at the expense of more routers. Torus topology is an extension of this topology that also connects the routers on the edges among each other. The mesh topology can be stacked together to form higher dimension NoCs such as 3D cube topology. The mesh topology is a standard topology where routers connected in all four cardinal directions as shown in Figure. It is easily scalable and thus it has been used in many research works as well as physical applications [9]. Mesh topology is simple and adaptive to many NoC designs making it an attractive choice. Although the mesh topology offers a lot of flexible as a NoC topology, it has a limitation to transmit data diagonally or directly beyond the neighbouring routers. Qian et al. suggests that the mesh topology can be broken into regions of 4x4 and a hub router can be added to the NoC which can dynamically reconfigurate itself to connect to surrounding routers allowing packets to be transmitted diagonally to reduce latency [10]. Another implementation by Wang et al. modifies the mesh topology into a diagonal mesh (DMesh) which has additional diagonal links connecting each router on the NoC [11]. Both implementations show improvements over the traditional mesh topology at the cost of additional links and increase complexity of each router. Crossbars have additional contention from more input and output ports which can potentially reduce performance.

CONGESTION AWARENESS: Congestion awareness can be catorgorized in three types, locally adaptive, regionally adaptive and globally adaptive. The amount of congestion data received by each router depends on the type of congestion awareness chosen. Locally adaptive does not need traffic flow between routers as the amount of credits available from the credit value registers (CVRs) is sufficient while globally adaptive need a large amount of data transfer to hold all the congestion values to each downstream router. Congestion awareness generally sacrifices data transfer to determine a better path for a packet to route to its destination.

LOCALLY ADAPTIVE: Research on adaptive algorithms began with locally adaptive routing as shown in early implementations such as DyAD [25] and DyXY [24].These implementations rely solely on congestion data of its neighbours to select the next hop direction. Preferred Output Adaptive Routing [26] and DyXY have used the available credits as the congestion value to adaptively determine the direction for the next hop. DyAD uses an external 1-bit signal to indicate if there is any congestion or not. Due to the fact that locally adaptive algorithms are greedy in nature, the output port selected may not be the best direction. Hu and Marculescu proved [25] that DOR (XY) routing performed better than their DyAD routing for uniform traffic. This is also proven by Gratz et al. that locally adaptive routing performs worse than RCA due to its greediness [4]. Another technique, Neighbour-on-Path (NoP) utilizes a similar neighbouring monitor like DyXY [27]. NoP allows non-minimal path routing to avoid the congested NoC area. Although this improves latency, its non-minimal routing can lower the injection rate before its saturation. 23 DyAD switches between adaptive and deterministic routing depending on traffic conditions. Hu and Marculescu have chosen 60% as the threshold to switch between deterministic to adaptive routing. DyAD uses the OE routing algorithm, which prohibits certain turns depending on the current location of the packet, and when there is a choice to select one of the two output direction, DyAD will select the output port with no congestion. When both outputs are not congested or both are

congested, then the algorithm chooses one of them randomly. Preferred Output Adaptive Routing and DyXY are similar and would choose the direction with more credits available when choices are available. Preferred output routing makes use of route look-ahead while DyXY does not.

The OCP readily adapts to support new core capabilities while limiting test suite modifications for core upgrades.

Basically OCP has the address is of 13bits, data is of 8bits, control signal is of 3bits and burst is of integer type. The 8kbit memory (2^{13} = 8192bits = 8kbits) is used in the slav e side in order to verify the protocol functionality. The System will give the inputs to OCP Master during Write operation and receive signals from OCP Slave during Read operation.

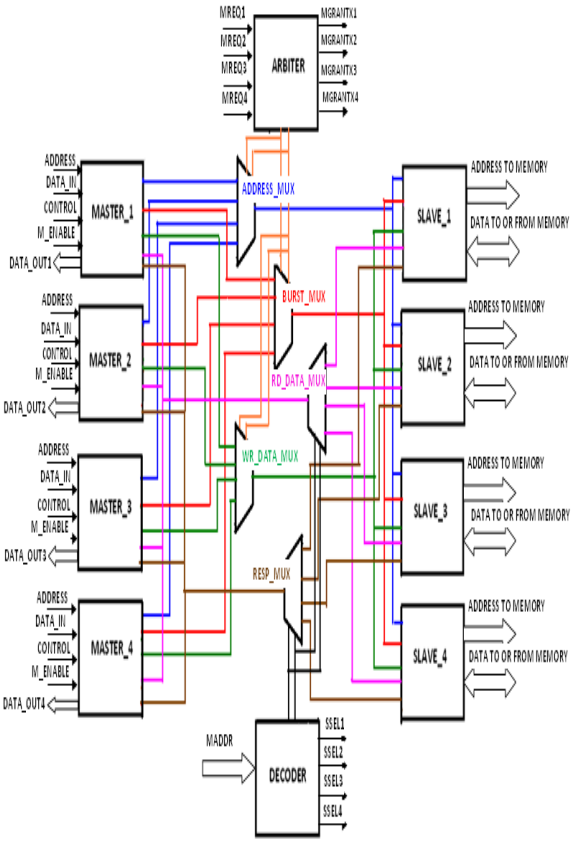

FIGURE : BLOCK DIAGRAM

PROPOSED TECHNIQUE:

SCALABLE ENCRYPTION ALGORITHM:

SEAn,b operates on various text, key, and word sizes. It is based on a Feistel structure with a variable number of rounds, and is defined with respect to the following parameters:

• n plaintext size, key size;

• b processor (or word) size;

• nb = n/2b number of words per Feistel branch;

• nr number of block cipher rounds. As an only constraint, it is required that n is a multiple of 6b (Because both the plain text are separated into 2 parts, and all the operation are done in 3 words). Example- using 8-bit processor, we can derive a 48-bit block ciphers, denoted as SEA48, 8. Let x be a n/2-bit vector. We consider the following two representations.

• Bit representation: x b = x ((n/2)-1)…… x(2) x(1) x(0).

• Word representation: x w = x nb-1 x nb-2 …… x2 x1 x0.

Due to its simplicity constraints, SEAn,b is based on a limited number of elementary operations (selected for their availability in any processing device) denoted as follows: 1) Bit wise XOR 2) Mod 2b addition 3) A 3-bit substitution box S: = [0, 5, 6, 7, 4, 3, 1, 2] that can be applied bit wise to any set of 3-bit words for efficiency purposes. In addition, we use the following rotation operations: 4) Word rotation R, defined on nb-word vectors R: x y = R (x) yi+1 = x i 0<=I<=nb-2 y0 = x n b -1 5) Bit rotation r, defined on nb-word vectors R: x y = r (x) y = x3i >>>1 y 3i+1 = x 3i+1 y 3i+2 = x 3i+2 <<<= i <=(nb/3) –1 and –1 and >>> and << respectively.

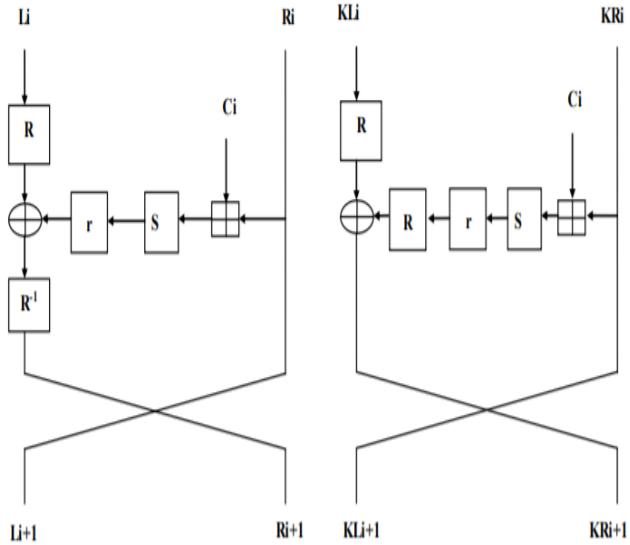

Fig: Encrypt/decrypt Round and key round

LOOP ARCHITECTURE OF SEA:

The structure of our loop architecture for SEA is depicted in Fig, with the round function on the left part and the key schedule on the right part. Resource-consuming blocks are the S boxes and the mod2b adder; the Word Rotate and Bit Rotate blocks are implemented by swapping wires.

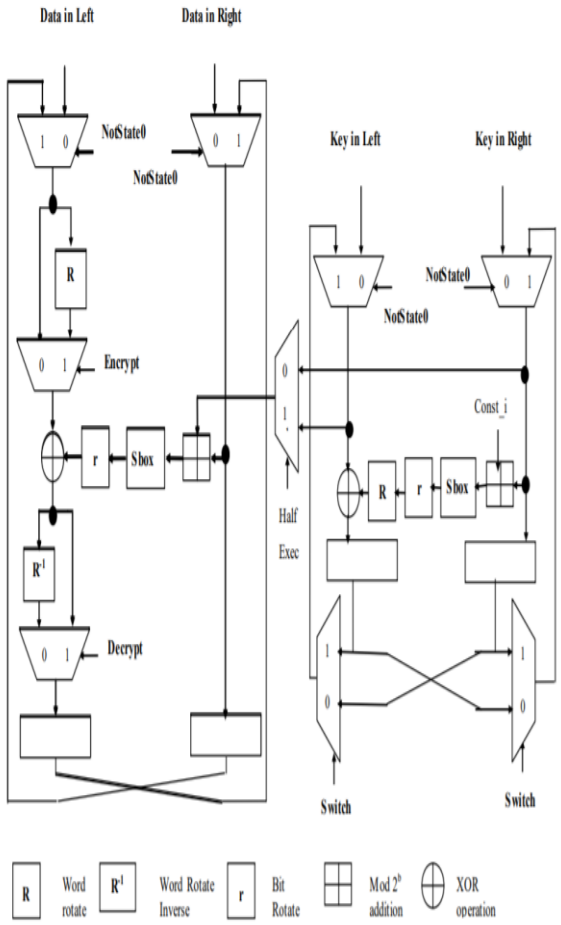

Fig . Loop architecture for SEA

According to the specifications, the key schedule contains two multiplexors allowing to switch the right and left part of the round key at half the execution of the algorithm using the appropriate command signal Switch. The multiplexor controlled by Half Exec provides the round function with the right part of the round key for the first half of the execution and transmits its left part instead after the switch. To support both encryption and decryption, finally added two multiplexors controlled by the Encrypt signal. Supplementary area consumption will be caused by the two routing paths. In the round function, the mod 2 adders are realized by using nb, b-bits adders working in parallel without carry propagation between them. In the key schedule, the signal Const_i (provided by the control part) can only take a value between 0 and nr/2.

RESULT:

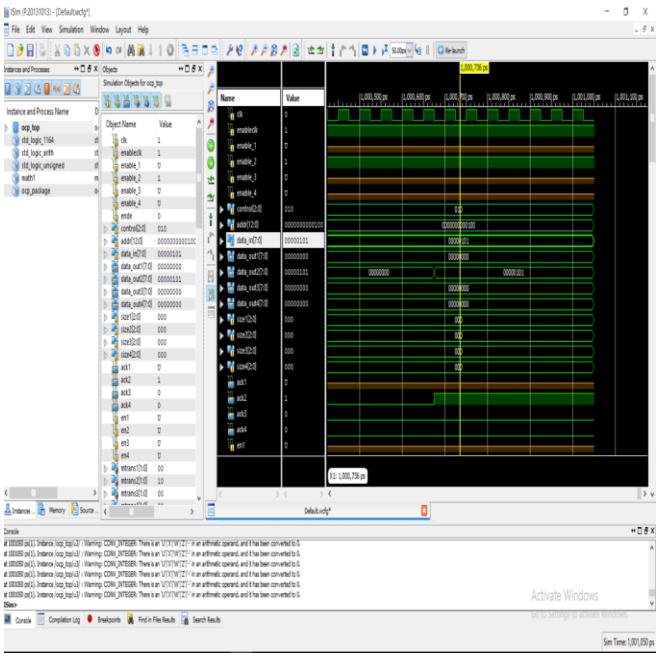

Fig: Existing technique

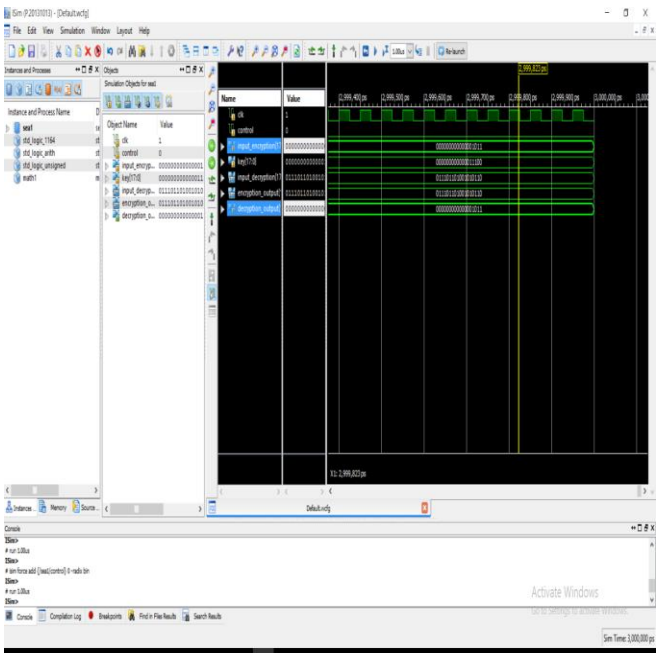

Fig: Proposed SEA algorithm

CONCLUSION:

This concept presents a novel approach that improves NoC throughput by packet prioritization. The objective is to increase the NoC throughput by using congestion aware information. Congestion awareness information has already been applied for on-chip communication to improve NoC routing. This project expanded adaptive routing by employing regional congestion data to latency for packet routing. Efficient enhancement described the methodology of expanding the regional congestion awareness data to improve packet selection for both VC and switch allocators. A new methodology of Congestion Aware Adaptive Routing (CAAR) is designed to prioritize the packet/flit allocation that suffers the most latency while travelling between NoC cores. Moreover, to improve on hardware usage and to avoid any additional links between routers, CAAR removes the sideband network to transfer congestion data and instead adds the congestion information into the header flit of a packet.

FUTURE SCOPE:

Programmable switching structure is an optimal way to implement the telecommunication system for extension lines using Network on chip (NoC) concept in FPGA chips, and make provision to co-control or cascade to the other FPGA for multiplexing the extension lines. The research work is not limited to cluster configuration, it can be made upto N = 32, 64, 128 or more based on the synthesis tools availability and designer's need. The effect of higher number of stages can also be investigated in order to increase the switching capacity. TACIT network security has the advantage that the block size and key size can be of 'N' bits and the integration of security with programmable structure will have best results, especially for secured data transmission in telecommunication network. In future, an additional work should be carried out with added features of network security with other algorithms for encryption and decryption of data transfer among inlets and outlets. The same concept of NoC and programmable switching structures can be integrated with other wireless technologies such as Wimax, WiFi, Bluetooth, and wireless sensor networks.

REFERENCES:

[1] A. J. Elbirt, W. Yip, B. Chetwynd, and C. Paar, "An FPGA implementation and performance evaluation of the AES block cipher candidate algorithm finalists," in Proc. AES Candidate Conf., 2000, pp. 13–12 .Oct 2005.

[2] K. Jarvinen, M. Tommiska, and J. Skytta, "Comparative survey of high-performance cryptographic algorithm implementations on FPGAs," IEE Proc. Inf. Security, vol. 152, pp. 3–12, Oct. 2005.

[3] F. Macé, F.-X. Standaert, and J.-J. Quisquater, "FPGA Implementation(s) of a Scalable Encryption Algorithm"IEEE Transactions on very large scale integration (VLSI) system .vol. 16, no. 2, FEB 2008.

[4] F.-X. Standaert, G. Piret, N. Gershenfeld, and J.-J. Quisquater, "Sea: A scalable encryption algorithm for small embedded applications," in Proc. CARDIS, 2006, pp. 222– 236.

[5] F.-X. Standaert, G. Piret, G. Rouvroy, and J.-J. Quisquater, "FPGA implementations of the ICEBERG block cipher," in Proc. ITCC, 2005, pp. 556–561.

[6] K.Wong, M.Wark and E.Dawson" A single-chip FPGA implementation of the data encryption standard (des) algorithm" Global Telecommunications Conference, 1998. GLOBECOM98. The Bridge to Global Integration. IEEE, 10.1109/ GLOCOM.1998.776849

[7] Advanced Encryption Standard, FIPS PUB 197, Nov. 2001. [8] Data Encryption Standard, FIPS PUB 46-3, Oct. 1999