CHAPTER 1

INTRODUCTION

1.1 NETWORK ON CHIP (NoC)

A typical NOC based system consists of processing elements (PE), network interfaces (NI), routers and channels. The router further contains switch, buffers and routing logic as shown in Figure 1. Buffers consume near about 64% of the total node (router + link) leakage power for all process technologies, which makes it the largest power consumer in any NOC system. All links in NOC can be simultaneously used for data transmission, which provides a high level of parallelism. It is an attractive solution to replace the conventional communication architectures such as shared buses or point-to-point dedicated links by NOC. NOC provides better scalability than on-chip buses because as more resources are introduced to a system, also more routers and links are introduced to connect them to the network. Buffers consume the largest fraction of dynamic and leakage power of the NOC node (router + link). Storing a packet in buffer consumes far more power as compared to its transmission. Thus, increasing the utilization of buffers and reduction in their number and size with efficient autonomic control reduces the area and power consumption. Also the Wormhole flow control has been proposed to reduce the buffer requirements and enhance the system throughput. However, one packet may be occupy or cover several intermediate switches at the same time. This introduces the problem of deadlocks and lovelocks. Thus to avoid this problem the use of virtual channel is introduced. Virtual channel flow control exploits an array of buffers at each input port. By allocating different packets to each of these buffers, flits from multiple packets may be sent in an interleaved manner over a single physical channel. This improves the throughput and reduces the average packet latency.

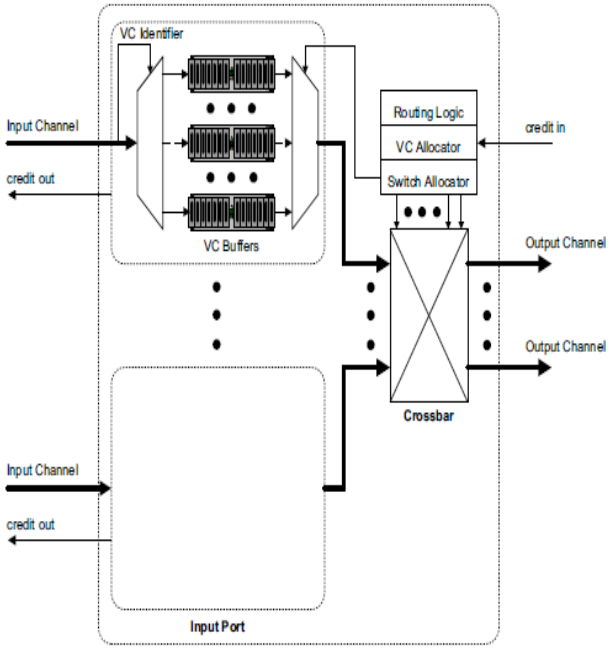

Fig 1.1 Conventional Virtual Channel Router Architecture

However, one packet may be occupy or cover several intermediate switches at the same time. This introduces the problem of deadlocks and lovelocks . Thus to avoid this problem the use of virtual channel is introduced. Virtual channel flow control exploits an array of buffers at each input port. By allocating different packets to each of these buffers, flits from multiple packets may be sent in an interleaved manner over a single physical channel. This improves the throughput and reduces the average packet latency.

1.2 SYSTEM ARCHITECTURE

To enhance the performance of typical VC architectures, new VC buffers should be inserted because a congested port cannot utilize the resources of neighbouring free port. VC utilization can be enhanced by sharing free buffers of a port with neighbouring overloaded input ports. The improved VC utilization directs to reduce the number of VC buffers and sustain the system performance. VC utilization can be further improved by sharing them among all the input ports. However, full sharing increases the control logic complexity and power

consumption due to a larger input crossbar. Thus a trade-off between resource utilization, design complexity and power consumption is needed.

Instead of sharing the VC buffers among all the input ports, PVS approach shares the VC buffers among few input ports according to the communication requirements. With this technique, the buffer utilization is increased and utilization level approaches close to the fully shared architecture without significant silicon area and power consumption overhead because of reduced input crossbar size.

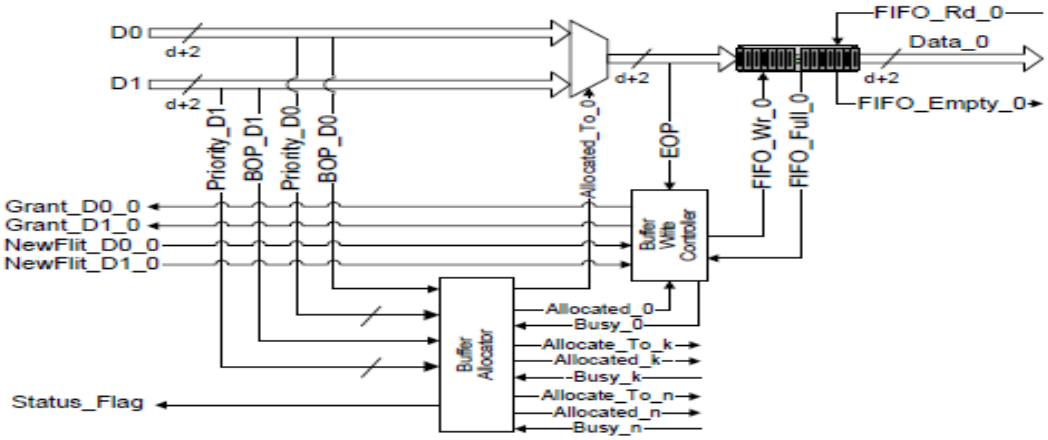

Fig 1.2. Proposed Input Part of Router Architecture

NI is a generic interface, which needs to be standardized. Main tasks of the NI are packetization and de-packetization of data. Different services can be introduced in NI architecture, such as multicasting and error monitoring. The queuing buffer for the PE can be considered as the part of NI on input port of the switch. Thus each PE has its own dedicated buffer, which simplifies the control logic and enhance the throughput without any area overhead. PI is the core specific interface, which acts as a clock synchronizer. In PVS approach, the input crossbars and control logic are responsible for buffer allocation and receiving the data packets. Input architecture uses the distributed routing logic whereas central VC allocator is used for the group of channels sharing the buffers. Instead of sharing the VC buffers among all the input ports, PVS approach shares the VC

buffers among a few input ports according to the communication requirements. With this technique, the buffer utilization is increased and approaches close to the utilization level of fully shared architecture without significant silicon area and power consumption overhead because of reduced input crossbar size. Overall, PVS approach is tradeoffs between system throughput, resource utilization and power consumption. The proposed architecture, sharing of buffers between neighbouring input ports, is shown in Fig1.2. The router architecture can be divided into two parts, the Input and the Output. The Input part is responsible for buffer allocation and receiving the packets from neighbouring routers. The Output part computes the route and transmits the packets accordingly. Within the router, the Input and the Output parts do not communicate at all. Both parts simply write and read from the buffers. Fig1.2 shows the signals of both parts to write and read from buffers and also the external interface of router. In PVS approach, the input crossbars and control logic are responsible for buffer allocation and receiving the data packets. Input architecture uses the distributed routing logic but central VC allocation for the group of channels sharing the buffers.

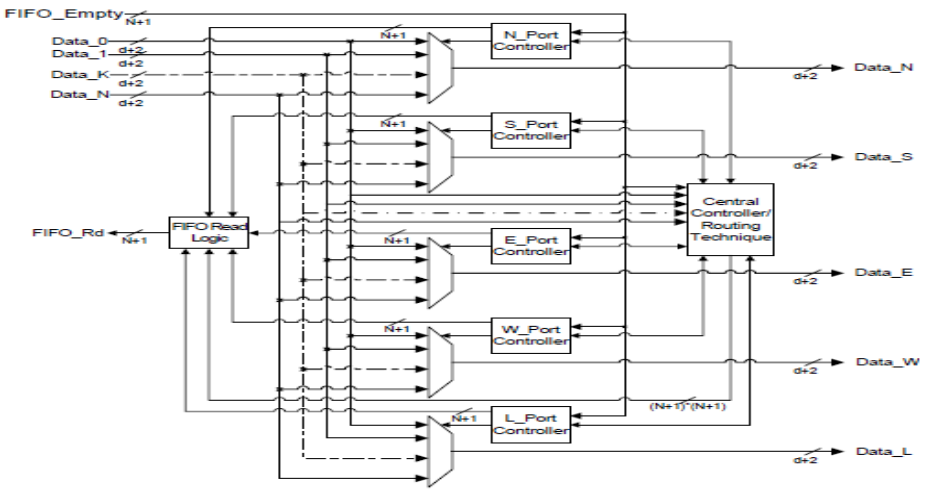

Fig1.3 System Output Part of Router Architecture

Thus the architecture is fully optimized for any number of I/O ports. The number of I/O ports can be increased to any number by simply duplicating the

hardware on input side and increasing the crossbar size on output side. At the same time, the buffer utilization is increased which significantly reduces the new buffer requirements. Thus decreasing the buffer size by 4 buffer slots (25%) leads to a power savings of 25.72% as compared to the baseline. Power reduces by 40.77% when the buffer size is reduced to 50% of the baseline. The proposed architecture can be integrated into the existing automated NOC design flow without requiring extra effort. An automated design flow can utilize the adoptability feature of this architecture to generate an optimized router module according to the application and design requirements. Therefore, the proposed architecture can play a significant role in maintaining the performance of NOC based systems regardless of the topology.

1.3 OVER VIEW OF THE PROJECT

The structure of this project is decomposed as chapters. In chapter2 describes about the related work of the related work of the multilayer diagnosis for single fault analysis in chapter 3 describes about conventional system in chapter 4 about the methodology involved in combination of various techniques like software based diagnosis protocol and problem statements in chapter 5 describes about the software description of our proposed system like Xilinx. In chapter 6 deals with results and discussion with comparison graph as well as table. Finally conclude the work in chapter 7and also say about some possible extension to make multi layer diagnosis for multi fault tolerant NOC system more efficient and describes about its future work.

CHAPTER 2

LITERATURE SURVEY

In this paper [1], presented Integration of large number of electronic components on a single chip has resulted in complete and complex systems on a single chip. The energy efficiency in the System-on-Chip (SoC) and its communication subset, the Network-on-Chip (NoC), is a key challenge, due to the fact that these systems are typically battery-powered. We present a survey that provides a broad picture of the state-of-the-art energy-efficient NoC architectures and techniques, such as the routing algorithms, buffered and bufferless router architectures, fault tolerance, switching techniques, voltage islands, and voltage-frequency scaling. The objective of the survey is to educate the readers with the latest design-improvements that are carried out in reducing the power consumption in the NoCs. In this paper [2], presented an uncover novel and imminent threat to an emerging computing paradigm: MPSoCs built with 3rd party IP NoCs. We demonstrate that a compromised NoC (C-NoC) can enable a range of security attacks with an accomplice software component. To counteract these threats, we propose Fort-NoCs, a series of techniques that work together to provide protection from a C-NoC in an MPSoC. Fort-NoCs's foolproof protection disables covert backdoor activation, and reduces the chance of a successful side-channel attack by "clouding" the information obtained by an attacker. Compared to recently proposed techniques, Fort-NoCs offers a substantially better protection with lower overheads. In this paper [3],presented Networks-on-Chip (NoCs) are vulnerable to security attacks, which can degrade the network performance, reduce its availability or even block the entire network. In this context, this work aimed at increasing the availability of a NoC by means of the implementation of hardware-based mechanisms that filter malicious packets injected into the network by an attacking core. The mechanisms discard packets

that affect the network availability, and regulate the injection rate of communication flows that attempt to consume a bandwidth higher than a limit specified by the system designer. The security mechanisms were described in VHDL and synthesized to an ASIC technology. Results demonstrate that they are effective in improving the network availability, with a reduced silicon overhead and low impact to the NoC performance. In this paper [4],presented In multiprocessor system-on-chip (MPSoC), a CPU can access physical resources, such as on-chip memory or I/O devices. Along with normal requests, malevolent ones, generated by malicious processes running in one or more CPUs, could occur. A protection mechanism is therefore required to prevent injection of malicious instructions or data across the system. We propose a self-contained Network-on-Chip (NoC) firewall at the network interface (NI) layer which, by checking the physical address against a set of rules, rejects untrusted CPU requests to the on-chip memory, thus protecting all legitimate processes running in a multicore SoC. To sustain high performance, we implement the firewall in hardware, with rule-checking performed at segment-level based on deny rules. Furthermore, to evaluate its impact, we develop a novel framework on top of gem5 simulation environment, coupling ARM technology and an instance of a commercial point-to-point interconnect from STMicroelectronics (STNoC). Simulation tests include scenarios in which legitimate and malicious processes, running in different CPUs, request access to shared memory. Our results indicate that a firewall implementation at the NI can have a positive effect on network performance by reducing both end-to-end network delay and power consumption. We also show that our coarse-grain firewall can prevent saturation of the on-chip network and performs better than fine-grain alternatives that perform rule checking at page-level. Simulation results are accompanied with field measurements performed on a Zedboard platform running Linux, whereas the NoC Firewall is implemented as a reconfigurable, memory-mapped device on top of AMBA AXI4 interconnect fabric. In this paper [5],presented Realization of an

application using Wireless Sensor Networks (WSNs) using Sensor Nodes (SNs) brings in profound advantage of ad-hoc and flexible network deployment. Implementation of these networks face immense challenges due to short wireless range; along with limited power, storage & computational capabilities of SNs. Also, due to the tiny physical attributes of the SNs in WSNs, they are prone to physical attacks. In the context of WSNs, the physical attacks may range from destroying, lifting, replacing and adding new SNs. The work in this paper addresses the threats induced due to physical attacks and, further proposes a methodology to mitigate it. The methodology incorporates the use of newly proposed secured and efficient key distribution technique based on the additional commodity hardware Trusted Platform Module (TPM). Further, the paper demonstrates the merits of the proposed methodology. With some additional economical cost for the hardware, the proposed technique can fulfill the security requirement of WSNs, like confidentiality, integrity, authenticity, resilience to attack, key connectivity and data freshness. In this paper [6], presented Network-on-Chips (NoCs) are becoming integral parts of modern microprocessors as the number of cores and modules integrated on a single chip continues to increase. Research and development of future NoC technology relies on accurate modeling and simulations to evaluate the performance impact and analyze the cost of novel NoC architectures. In this work, we present BookSim, a cycle-accurate simulator for NoCs. The simulator is designed for simulation flexibility and accurate modeling of network components. It features a modular design and offers a large set of configurable network parameters in terms of topology, routing algorithm, flow control, and router microarchitecture, including buffer management and allocation schemes. BookSim furthermore emphasizes detailed implementations of network components that accurately model the behavior of actual hardware. We have validated the accuracy of the simulator against RTL implementations of NoC routers. In this paper [9],presented As multicore processors find increasing adoption in domains such as aerospace and medical devices where failures have

the potential to be catastrophic, strong performance isolation and security become first-class design constraints. When cores are used to run separate pieces of the system, strong time and space partitioning can help provide such guarantees. However, as the number of partitions or the asymmetry in partition bandwidth allocations grows, the additional latency incurred by time multiplexing the network can significantly impact performance. In this paper [10], presented The security issues of the Internet of Things (IoT) are directly related to the wide application of its system. Beginning with introducing the architecture and features of IoT security, this paper expounds several security issues of IoT that exist in the three-layer system structure, and comes up with solutions to the issues above coupled with key technologies involved. Among these safety measures concerned, the ones about perception layer are particularly elaborated, including key management and algorithm, security routing protocol, data fusion technology, as well as authentication and access control, etc. In this paper [8], presented 3D-MPSoCs integrate cores of several vendors and support different applications on a single die, providing large performance and cost reduction. 3D-technology presents many security challenges and offers new opportunities to implement protection countermeasures. 3D-NoCs can be explored to assist the overall security of the system. In this work, we propose a 3D-NoC hardware architecture able to protect the 3D-MPSoCs against software attacks. Dynamic firewalls create elastic security zones by wrapping a set of components according to a trust policy, guaranteeing the protection and efficient execution of all applications. We compare our approach with several 3D-protection proposals and we show that our mechanism performs a fast detection of attacks, decreases the cost of security in terms of power and provides a high level of security. In this paper [7], presented A software-based method and system to provide a secure user interface on multiple and diverse electronic computing devices with a customized and secure dashboard feature. The systems and methods simultaneously integrate internally generated software utilities of an enterprise

with externally accessed software operating in a ‘cloud computing’ environment. The systems and methods can be used in management and operations that use computer based software, data management, creative processes and communication systems. The systems and methods reduce the requirement for additional programming to integrate or interchange equivalent and independently developed software for use within an enterprise. The systems and methods permit social network communications between members of an enterprise and an external community. The security features of the user interface portal permit collaborations between parties in an external community and enterprise members that can develop new processes that remain proprietary to the enterprise and parties of an external community.

CHAPTER 3

CONVENTIONAL SYSTEM

Networks-on-Chips (NoCs) have appeared in recent years as new strategy to connect and manage the communication among the variety of intellectual property blocks required in complex System-on-Chips (SoCs). However, the advantages introduced by the use of such a complex communication infrastructure may lead to new weaknesses in the system that should be subjected to careful studies and evaluations. While NoCs have been an emerging area of academic and research interest, security in such systems remains so far mainly unexplored. Therefore, focus on the security aspects related to the intercommunication infrastructure and an outline of possible techniques that could be applied to NoCs to contribute to the overall security of the system. As presented in security attacks to an embedded system can be classified in different ways. If consider a classification in terms of the agents used to perform the attack, they can be grouped as software attacks, physical or invasive attacks and side channel attacks. In the first group include all attacks launched through software agents such as viruses, worms and trojan horses, carried out, for instance, using pitfalls in code constructs, such as in the case of buffer overflow attacks . Physical attacks require physical intrusion in the embedded system at some level. They imply the use of sophisticated micro-probing techniques, involving de-packaging and reconstruction of the layout, in order to infer at various granularity the architectural structures and values on the buses and interfaces of the components. Side channel attacks are based on information gained from the physical implementation of the system, such as power consumption, timing information, or electromagnetic leaks. They exploit the correlation between the information measured and the execution flow generating it. The adoption in SoCs of a complex communication paradigm may introduce additional weaknesses in the

system. Systems based on NoCs can be subjected to attacks addressing their specific structure. In particular, three types of attacks can be identified . Denial of Service (DoS) attacks aim at lowering system performance in several ways. Among those, Bandwidth Reduction attacks aim at reducing the communication bandwidth through the transmission of frequent and useless packets, in order to cause high latency in the on-chip communications, up to the saturation of the network. Operation life of battery operated embedded systems is the target of Draining or Sleep Deprivation attacks. This particular type of attack can be performed through continuous sending of requests to the victim of the attack, in order to make it execute power-hungry tasks. Extraction of secret information aims at reading sensitive data, critical instructions or information kept in areas of the memory or in configuration registers in specific targets. It can be performed exploiting buffer overflow or similar techniques addressing software weaknesses. Hijacking attacks aim at altering the execution or configuration of the system in order to make it perform tasks set by the attacker in addition to its normal duties, such as in the case of the exploitation of buffer overflow to bypass Digital Right Management's protection in audio CODEC. 3 On-chip secure architectures: state of the art this section presents an overview of the most recent research work related to the security aspects of a NoC. A framework to secure the exchange of cryptographic keys within a NoC, addressing in particular the protection from power/EM attacks of a system containing non-secure cores as well as secure ones. The framework supports authentication, encryption, key exchange, new user's keys and public key storage, and similar procedures. No unencrypted key leaves the cores on the NoC and only secure IP cores running trusted software are supported. At the network level, security is based on symmetric key cryptography, where each secure core has its own security wrapper storing a private network key in a non-volatile memory.

CHAPTER 4

METHODOLOGY

Diagnosis techniques can be employed on various network layers to locate permanent faults in NoCs. Each layer has its own functionality, with which diagnosis must be integrated, and its specific information that is subject of diagnostic processes:On transport layer, a software-based diagnostic protocol (DP) can locate faulty network components by analyzing incoming data packets. On network and data link layer, functional diagnosis (FD) identifies functional misbehavior and remaining functionality of a network switch using dedicated test packets.

4.1 COMBINATION OF DIAGNOSTIC TECHNIQUES

The interaction between layer-specific diagnostic techniques is achieved by exchanging di-agnostic information between the layers and by using this in-formation to optimize the respective diagnostic processes.

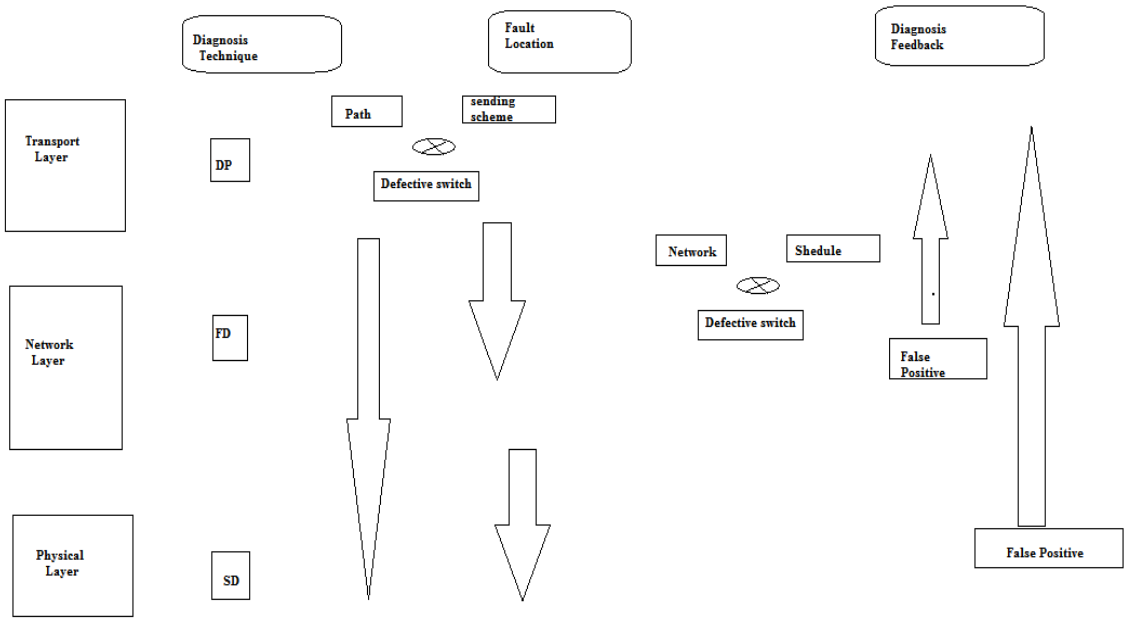

Fig 4.1 Interaction of Multi-Layer Technique

All possible combinations of the three layer-specific techniques will be considered. They are denoted as follows:

1)DP+FD,2) DP+SD,3) DP+FD+SD,4) FD+SD.

4.2 CROSS-LAYER INFORMATION FLOW

When a fault has only been detected, without further diag-nostic information, the set of potentially defective switches, , is the set of all switches of the NoC, S. A diagnosis technique t 2 fDP; F D; SDg determines the subset of faulty switches (including the connected links) out of this set. This is described with the diagnosis function d_t : S ! S. Details on the implementation of this function are described in subsequent chapters for each technique t. Here we show the multi-layer flow of diagnosis for DP, FD, and SD. This flow is modeled by the algorithm presented in Listing 1. This algorithm does not have a centralized implementation, but it describes the behavior that results from the distributed interaction of the layer-specific techniques. structural diagnosis (SD). If any of these techniques is not implemented in the system, it is skipped. Each diagnostic technique receives the set of potentially faulty switches, , as input and passes a potentially reduced set on to the lower layers. Assuming a deterministic routing, the set of switches that have to be diagnosed by DP can be reduced due to the knowledge about the path a packet has taken from sender to receiver. Thus, the diagnostic function is invoked only for the switches situated on that path, $_{path}$ (line 6). Diagnosis of FD and SD is carried out for set (line 12 and 18) received from the preceding technique.

4.3 SOFTWARE-BASED DIAGNOSIS PROTOCOL

The transport layer protocol, DP, is an enhanced version of our software-based end-to-end protocol .It has an increased localization granularity: besides faulty links, faulty crossbar connections of a switch can be diagnosed as well. The failure of a complete switch is diagnosed as the failure of all its links or crossbar

connections. Protocol DP consists of two parts: the base protocol and the diagnosis protocol. The base protocol is responsible for acknowledging the receipt of packets to the corresponding senders and for retransmitting packets in case they were received with errors or if they were lost. We employ end-to-end retransmission as switch-to-switch retransmission protocols have been found inferior unless unrealistically high error rates .The diagnosis protocol is used to locate the faulty network component in case of a permanent fault. It is activated for such packets that could not be successfully delivered to their destinations after a number of attempts. Beside the localization feature, the end-to-end protocol in-cludes a software-based routing mechanism for fault tolerance. If, due to a localized fault, a switch cannot forward a packet using the regular routing, the packet is consumed by the processing element (PE) connected to that switch. The PE calculates an intermediate destination that results in a path that bypasses the fault. It updates the packet header with this intermediate information and reinjects the packet into the network. Base protocol and diagnosis protocol are described in the following subsections.

4.3.1 BASE PROTOCOL

The base protocol uses the Selective Repeat ARQ mecha-nism, all packets received correctly at their destination are kept and only erroneous packets are retransmitted. To identify errors, each packet is equipped with a parity bit. Alternatively, any other error detection code (EDC) could be used. We fur-ther assume that on sender side, a retransmission buffer exists where a packet is stored until it is positively acknowledged. To reorder packets on receiver side, additional reorder buffers are required. Retransmission buffers and reorder buffers are assumed to be shared with the PE so that they can be managed in software. The base protocol handles packet corruption as well as packet loss. When a packet arrives at its destination it is checked for errors. If no errors are found, a positive acknowl-edgement (ACK) is transmitted back to the sender. Otherwise, negative

acknowledgement (NACK) is given. ACKs and NACKs are transmitted as single-flit protocol packets. When a sender receives an ACK, the corresponding packet is removed from the sender's retransmission buffer, and in case of a NACK, the packet is retransmitted. In case a packet is lost, neither ACK nor NACK are generated and thus the packet will neither be removed nor retransmitted. To handle this situation, the base protocol makes use of an adaptable timer . If no acknowledgement is received for a packet within time t , the packet is considered as lost and it is automatically retransmitted. This is also the case if a protocol packet with ACKs and NACKs was lost.

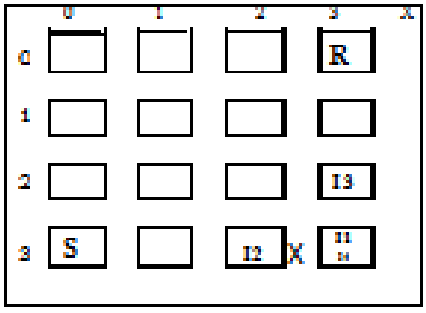

Fig 4.2 Sending Scheme

4.3.2 DIAGNOSIS PROTOCOL

On transport layer, fault localization relies on analysis of packet data. To be able to locate faulty network components, protocol DP must determine the path a packet has taken and thus requires knowledge about the routing. Moreover, fault localization relies on packets being retransmitted via the same path as the original packet. Therefore, the routing has to behave deterministically at least when the diagnosis protocol is activated. The routing may be reconfigured after diagnosis. The protocol finds the faulty network component by continuously narrowing down the fault position on the path. It can locate a single faulty network component at a time. The diagnosis protocol is activated when a permanent fault has been detected with a packet. That packet is no longer sent to the receiver directly but to a so called intermediate node determined by the sender.

Intermediate nodes are network nodes that are situated halfway on the path between sender and receiver. When a packet arrives at an intermediate node, it is consumed by that node and checked for faults. If the packet passes the check, i.e. no fault is found, this implies that the path from the original sender to the intermediate node is fault-free and the fault must reside in the second half of the path. In this case, the intermediate node takes over the role as sender for this packet, stores it in its retransmission buffer, and positively acknowledges the packet to the sender. On receipt of the acknowledgement, the sender removes the corresponding packet from its buffer. On the other hand, if the packet does not pass the check at the intermediate node, the fault is located between sender and intermediate. In this case, the intermediate node discards the packet and sends a negative acknowledgement to the sender. The sender now selects another intermediate node that is situated halfway to the old intermediate node.

4.4 FUNCTIONAL DIAGNOSIS

Functional diagnosis (FD) is performed on the network layer in order to identify more precisely the switch functionality affected by a fault. To this end, FD considers six functional failure classes as defined in

1) Misrouting: A packet is routed to a wrong output port.
2) Data corruption: Data within one or more flits is altered.
3) Packet loss: At least one packet is lost on its way from the input port to an output port of a switch.
4) Garbage packet: A new packet is generated and for-warded to an output port. This includes packet duplication.
5) Flit loss: At least one flit of a packet is lost on its way from the input port to an output port of a switch.
6) Garbage flit: A new flit is generated and forwarded to an output port.

CHAPTER 5

SOFTWARE ENVIRONMENT

5.1 SIMULATION SOFTWARE DESCRIPTION

The electronics industry has achieved a phenomenal growth over the last two decades, mainly due to the advent of VLSI. The number of applications of integrated circuits in high-performance computing, telecommunications and consumer electronics has been raising steadily and at a very fast pace. Typically, the required computational power (or, in other words, the intelligence) of these applications is the driving force for the fast development of this field. The current leading-edge technologies (such as low bit-rate video and cellular communications) already provide the end users a certain amount of processing power and portability. This trend is expected to be continued with very important implications of VLSI and systems design. As more and more complex functions are required in various data processing and telecommunications devices, the need to integrate these function in the small system /package is also increasing .The level of integration as measured by the number of logic gates in a monolithic chip has been steadily rising for almost three decades, mainly due to the rapid progress in processing technology and interconnect technology. Shows the evolution of logic complexity in integrated circuits over the last three decades, and marks the milestone of each era. Here, the numbers for circuit complexity should be interpreted only as representative examples to show the order of magnitude. A logic block can contain ten to hundred transistors depending upon the function.The important message here is that the logic complexity per chip has been increasing exponentially. The monolithic integration of a large number of functions on a single chip usually provides

- Less area / volume and therefore compactness.

- Less power consumption.
- Less testing requirements at system level.
- Higher reliability, mainly due to improved on-chip Interconnects.
- Higher speed, due to significantly reduced interconnection length.
- Significant cost saving.

5.1.1VLSI DESIGN FLOW

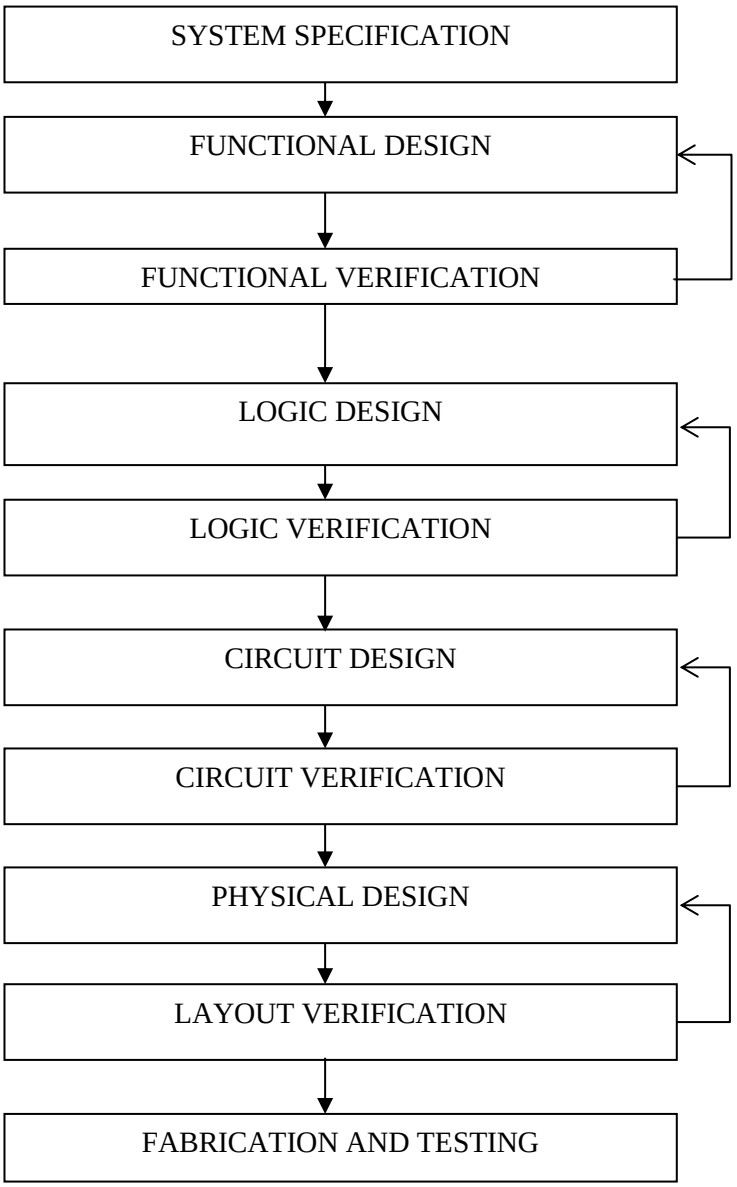

Fig 5.1.1 VLSI Design Flow

The design process, at various levels, is usually evolutionary in nature. It starts with a given set of requirements. Initial design is developed and tested against the requirements. When requirements are not met, the design has to be improved. If such improvement is either not possible or too costly, then the revision of requirements and its impacts analysis must be considered.

The VLSI design flow consists of three major domains, namely:

- Behavioral domain
- Structural domain
- Geometrical Layout domain.

The design flow starts from the algorithm that the behavior of the target chips. The corresponding architecture of the processor is first defined. It is mapped onto the chip surface by floor planning. The next design evolution in the behavioral domain defines finite state machines(FSMs), which are structurally implemented with functional modules such as registers and the arithmetic logic units (ALUs).These modules are then geometrically placed onto the chip surface using CAD tools for automatic module placement followed by routing, with a goal of minimizing the interconnects area and signal delays. The third evolution starts with behavioral modules are then implemented with leaf cells. At this stage the chip is described in terms of logic gates (leaf cells) which can be placed and interconnected by using a cell placement and routing program. The last evolution involves a detailed Boolean description of leaf cells and mask generation. In standard-cell based design, leaf cells are already pre-designed and stored in a library for logic design use.

5.1.2 DESIGN HIERARCHY

The use of hierarchy or “divide and conquer” technique involves dividing a module and then repeating this operation on the sub-modules until the complexityof the smaller parts becomes manageable. This approach is very

similar to the software case where large programs are split into smaller and smaller sections until simple sub-routines, with well defined functions and interfaces can be written. The design of VLSI chip can be represented in three domains. Correspondingly, a hierarchy structure can be described in each domain separately. However it is important for the simplicity of design that the hierarchies in different domains can be mapped into each other easily. In the physical domain, partitioning a complex system into its various functional blocks will provide a valuable guidance for the actual realization of these blocks on chip. Obviously, the approximate shape and size (area) of each sub-module should be estimated in order to provide a useful floor plan.

5.1.3 VLSI DESIGN STYLES

Several design styles can be considered for chip implementation of specified algorithms or logic functions. Each design style has its own merits and shortcomings, and thus a proper choice has to be made by designers in order to provide the functionality at low cost.

5.1.4 VLSI DESIGN PRESENT SCENARIO

Programmable logic device viz FPGA, CPLDs and their design tools have changed dramatically in the last 15 years. today with the advent of million gates, high performance, system level devices, the concept of SOC(system on chip) has arrived. Now you can create unique designs that were never possible before, get them to market longer, as a designer, your “windows of innovation” is practically unlimited. The FPGA and CPLD market is increasing and is about to grow more than the ASIC market, both terms of device and value. Companies like Xilinx, Altera, and Quick logic, Vantis, Lattice and Cypress are the major players in this market. million gate devices from them are already available. PLDs are one finding applications in areas, which formerly were the domain of ASICs only.

5.1.5 OVERVIEW OF HDL

Computer hardware and electronics industries, ever since their inception,have been working hand in hand. Technology development in the either of them helped and supported the other. For over a decade, computer researchers and programmers, and electronic component manufacturers have been on the lookout for a tool that can bridge the gap of exchanging data between designers and chip manufacturing companies. A Hardware Description Language (HDL) can be used to model and explain the behavior of any electronic component. The component can be as small as a gate or as complex as a complete multi-layered circuit board or a digital system. An HDL provides mechanisms to specify design specifications that are clear, unambiguous and simple-right from a small basic component till the complete system has been designed Today, we have many HDL 's, including the popular ones such as VERILOGHDL and Very High-Speed Integrated Circuit HDL (VHSIC-HDL) and others such asISP, UDLI, Abel, and HiLo.

5.2 VHDL OVERVIEW

5.2.1 INTRODUCTION

VHDL is a hardware description language. The word 'hardware' however is used in a wide variety of contexts, which range from complete systems like personal computers on one side to the small logical on their internal integrated circuits on the other side

5.2.2 HISTORY OF VHDL

The requirements for the language were first generated in 1981 under the VHSIC program. In this program, a number of U.S companies were involved in designing VHSIC chips for the Department of Defense (DoD). At that time most of the companies were using different languages to describe and develop their integrated circuits. As a result, a different vendors could not effectively exchange

designs with one another. A team of three companies IMB, Texas Instruments, and Intermetrics were first awarded the contract by the DoD to develop a version of the language in 1983. Version 7.2 of VHDL was developed and released to the public in 1985. After the release of version 7.2, there was an increasing need to make the language an industry-wide standard. Consequently, the language was transferred to the IEEE for standardization in 1987. According to IEEE rules, an IEEE standard has to be revaluated ever five years so that it may remain a standard. Consequently, the language was upgraded with new features, the syntax of many constructs was made more uniform, and many ambiguities present in the 1987 version of the language were resolved. This new version of the language is known as the IEEE std 1076-1993.

5.2.3 USES OF A VHDL

Since VHDL is a standard, the chip vendors can easily exchange their circuit designs without depending on their proprietary software. The designing process can be greatly simplified, as each component is designed individually and all such components are interconnected to form a full system- hierarchy and timing are always maintained. With simulators available, a circuit can be tested easily and any error found can be rectified without the expense of using a physical prototype, which means that design time and expenditure on this get slashed down. Programs written in either of the HDLS can be easily understood as they are similar to programs of C or Pascal.

5.2.4 FEATURES OF VHDL

VHDL provides five different types of primary constructs, called design units. They are

- Entity: It consists of a design's interface signals to the external circuitry
- Architecture: It describes a design's behavior and functionality.
- Package: It contains frequently used declarations, constants, functions, procedures, user data types and components.

- Configuration: It binds an entity to architecture when there are multiple architecture for a single entity
- Library: It consists of all the compiled design units like entities, Architectures, packages and configurations

5.2.5 RANGE OF USE:

The design process always starts with a specification phase. The component, which is to be designed, is defined with respect to function, size, interfaces, etc. Despite the complexity of the final product, mainly simple methods based on paper and pencil most of the time are being used. After that, self-contained modules have to be defined on the system level. Behavior models of standard components can be integrated into the system from libraries of commercial model developer's. The overall system can already be simulated. On the logic level, the models that have to be designed are described with all the synthesis aspects in view. As long as only a certain subset of VHDL constructs is used, commercial synthesis programs can derive the Boolean functions from this abstract model description and map them to the elements of an ASIC gate library or the configurable logic blocks of FPGAs. The result is a net list of the circuit or of the module on the gate level. Finally, the circuit layout for a specific ASIC technology can be created by means of other tools from the net list description. Every transition to a lower abstraction level must be proven by functional validation. For this purpose, the description is simulated in such a way that for all stimuli (=input signals for the simulation) the module's responses are compared. VHDL is suitable for the design phases from system level to gate level.

5.2.6 APPLICATION FIELD

VHDL is used mainly for the development of Application Specific Integrated Circuits (Asics). Tools for the automatic transformation of VHDL code into a gate level net list were developed already at an early point of time. This transformation is called synthesis and is an integral part of current design flows.

For the use with Field Programmable Gate Arrays (FPGAs) several problems exist. In the first step, Boolean equations are derived from the VHDL description, no matter, whether an ASIC or a FPGA is the target technology. But now, this Boolean code has to be partitioned into the configurable logic blocks (CLB) of the FPGA. This is more difficult than the mapping onto an ASIC library. Another big problem is the routing of the CLBs, as the available resources for interconnections are the bottleneck of current FPGAs

5.3. INTRODUCTION TO XILINX

Xilinx designs, develops and markets programmable logic products including integrated circuits (ICs), software design tools, predefined system functions delivered as intellectual property (IP) cores, design services, customer training, field engineering and technical support. Xilinx sells both FPGAs and CPLDs programmable logic devices for electronic equipment manufacturers in end markets such as communications, industrial, consumer, automotive and data processing. The ISE Design Suite is the central electronic design automation (EDA) product family sold by Xilinx. The ISE Design Suite features include design entry and synthesis supporting Verilog or VHDL, place-and-route (PAR), completed verification and debug using Chip Scope Pro tools, and creation of the bit files that are used to configure the chip. In this The Xilinx 12.1 ISE software version is used

5.3.1 XILINX 14.1 ISE SOFTWARE

The Xilinx 12.1 ISE software controls all aspects of the design flow. Through the Project Navigator interface, you can access all of the design entry and design implementation tools. You can also access the files and documents associated with your project. Project Navigator Interface By default, the Project Navigator interface is divided into four panel sub windows. On the top left are

the Start, Design, Files, and Libraries panels, which include display and access to the source files in the project as well as access to running processes for the currently selected source. The Start panel provides quick access to opening projects as well as frequently access reference material, documentation and tutorials. At the bottom of the Project Navigator are the Console, Errors, and Warnings panels, which display status messages, errors, and warnings. To the right is a multi document interface (MDI) window referred to as the Workspace. The Workspace enables you to view design reports, text files, schematics, and simulation waveforms. Each window can be resized, undocked from Project Navigator, moved to a new location within the main Project Navigator window, tiled, layered, or closed. You can use the View > Panels menu commands to open or close panels. You can use the Layout > Load Default Layout to restore the default window layout.

Design Panel: The Design panel provides access to the View, Hierarchy, and Processes panes.

View Panel: The View pane radio buttons enable you to view the source modules associated with the Implementation or Simulation Design View in the Hierarchy pane. If you select Simulation, you must select a simulation phase from the drop-down list.

Hierarchy Panel: The Hierarchy pane displays the project name, the target device, user documents, and design source files associated with the selected Design View. The View pane at the top of the Design panel allows you to view only those source files associated with the selected Design View, such as Implementation or Simulation. Each file in the Hierarchy pane has an associated icon. The icon indicates the file type (HDL file, schematic, core, or text file, for example). For a complete list of possible source types and their associated icons, see the “Source File Types” topic in the ISE Help. From Project Navigator, select Help > Help Topics to view the ISE Help. If a file contains lower levels of hierarchy, the icon has a plus symbol (+) to the left of the name. You can expand

the hierarchy by clicking the plus symbol (+). You can open a file for editing by double-clicking on the filename.

Processes Pane: The Processes pane is context sensitive, and it changes based upon the source type selected in the Sources pane and the top-level source in your project. From the Processes pane, you can run the functions necessary to define, run, and analyze your design.

The Processes pane provides access to the following functions:

Design Summary/Reports: Provides access to design reports, messages, and summary of results data. Message filtering can also be performed.

Design Utilities: Provides access to symbol generation, instantiation templates, viewing command line history, and simulation library compilation.

User Constraints: Provides access to editing location and timing constraints.

Synthesis: Provides access to Check Syntax, Synthesis, View RTL or Technology Schematic, and synthesis reports. Available processes vary depending on the synthesis tools you use.

Implement Design: Provides access to implementation tools and post-implementation analysis tools.

Generate Programming File: Provides access to bit stream generation.

Configure Target Device: Provides access to configuration tools for creating programming files and programming the device.

The Processes pane incorporates dependency management technology. The tools keep track of which processes have been run and which processes need to be run. Graphical status indicators display the state of the flow at any given time. When you select a process in the flow, the software automatically runs the processes necessary to get to the desired step. For example, when you run the Implement Design process, Project Navigator also runs the Synthesis process because implementation is dependent on up-to-date synthesis results. To view a running log of command line arguments used on the current project, expand Design Utilities and select View Command Line Log File.

Files Panel: The Files panel provides a flat, sort able list of all the source files in the project. Files can be sorted by any of the columns in the view. Properties for each file can be viewed and modified by right-clicking on the file and selecting Source Properties.

Libraries Panel: The Libraries panel enables you to manage HDL libraries and their associated HDL source files. You can create, view, and edit libraries and their associated sources.

Console Panel: The Console provides all standard output from processes run from Project Navigator. It displays errors, warnings, and information messages. Errors are signified by a red X next to the message; while warnings have a yellow exclamation mark (!).

Errors Panel: The Errors panel displays only error messages. Other console messages are filtered out.

Warnings Panel: The Warnings panel displays only warning messages. Other console messages are filtered out

Starting the ISE Software

To start the ISE software, double-click the ISE Project Navigator icon on your desktop, or

select Start >All Programs >Xilinx ISE Design Suite 12.1 > ISE Design Tools > Project Navigator.

Creating a New Project

To create a new project using the New Project Wizard, do the following:

1. From Project Navigator, select File > New Project. The New Project Wizard appears.

2. In the Location field, browse to c:\xilinx\12.1\ISE\ISEexamples or to the directory in which you installed the project.

3. In the Name field, enter wtut_vhd or wtut_ver.

4. Verify that HDL is selected as the Top-Level Source Type, and click Next.X-Ref Target

5. Select the following values in the New Project Wizard—Device Properties page:

ProductCategory,Family,Device,Package,Speed,SynthesisTool,Simulator,Preferred Language. This will determine the default language for all processes that generate HDL files.

6. Click Next, then Finish to complete the project creation.

After creating a project create file in that to type program using certain steps. Check syntax and Synthesis XST are runned to know whether error occurred in program. If not, Simulation procedure is followed to view Simulation output.

CHAPTER 6

RESULTS AND DISCUSSION

6.1 OUTPUT WAVEFORM FOR DDR DRAM

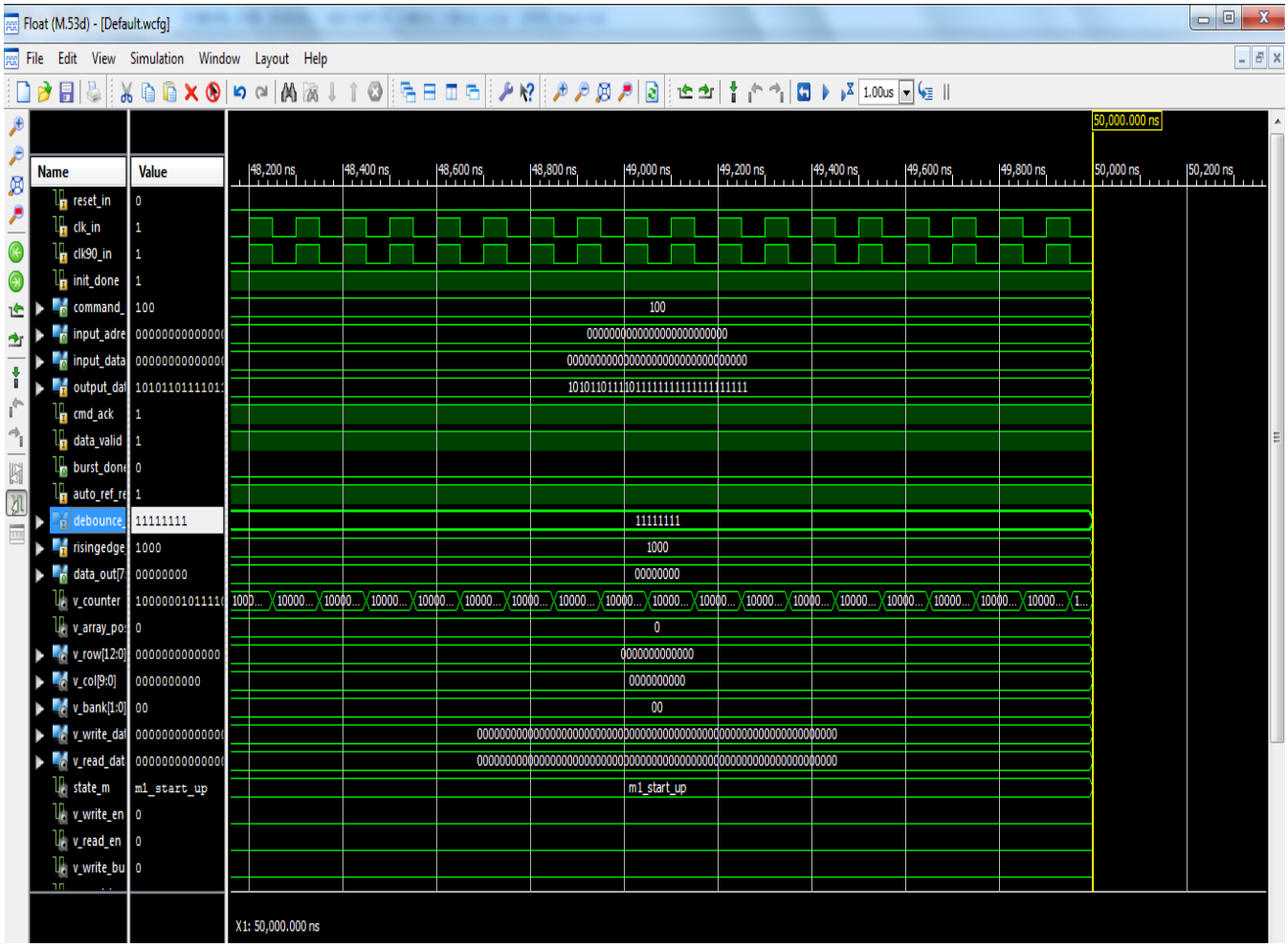

Figure 6.1DDR DRAM output waveform

6.2 OUTPUT WAVEFORM FOR UART TRANSMITTER

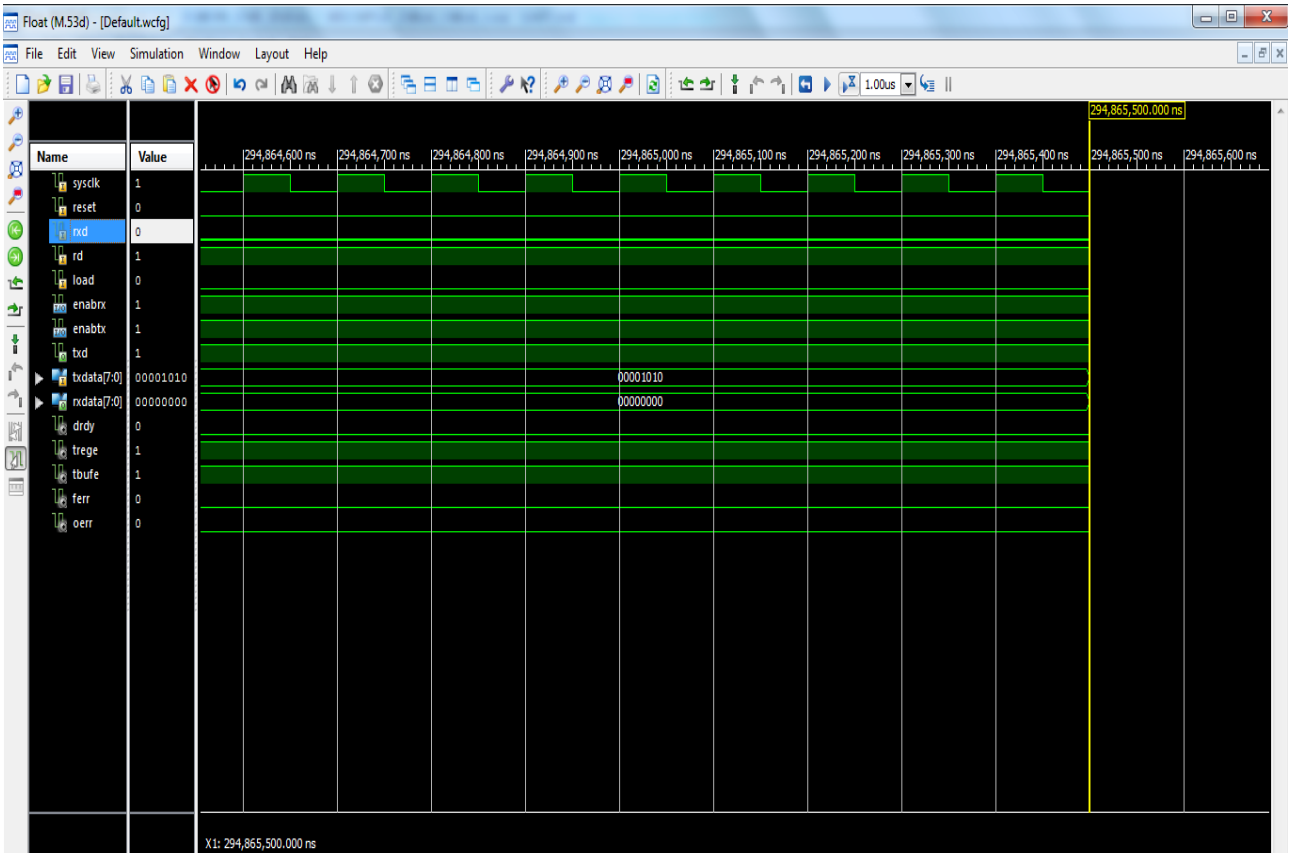

Fig 6.2 UART Transmitter output waveform

6.3 OUTPUT WAVEFORM FOR UART RECEIVER

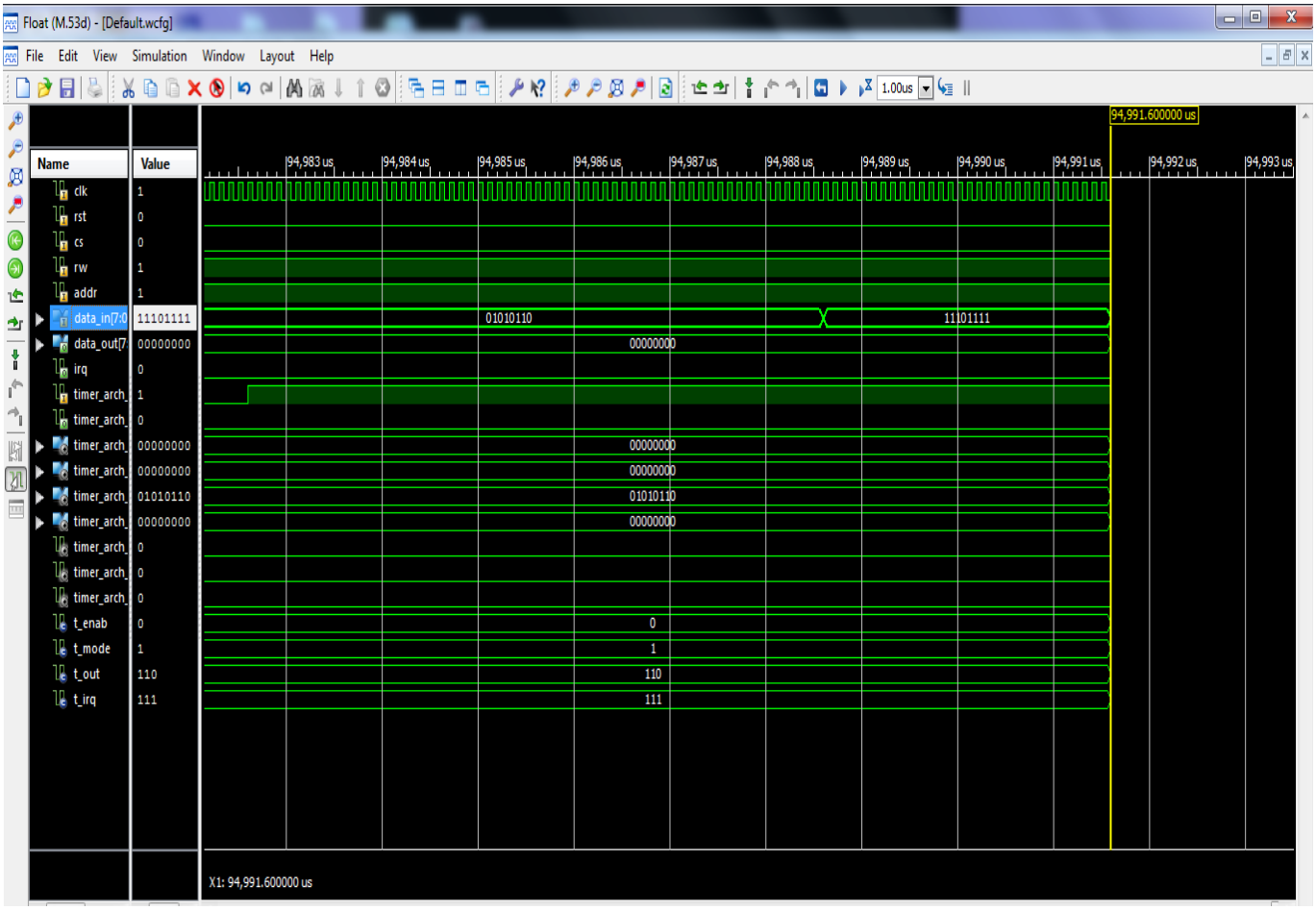

Fig 6.3 UART Receiver output waveform

6.4 OUTPUT WAVEFORM FOR MEMORY CIRCUITS

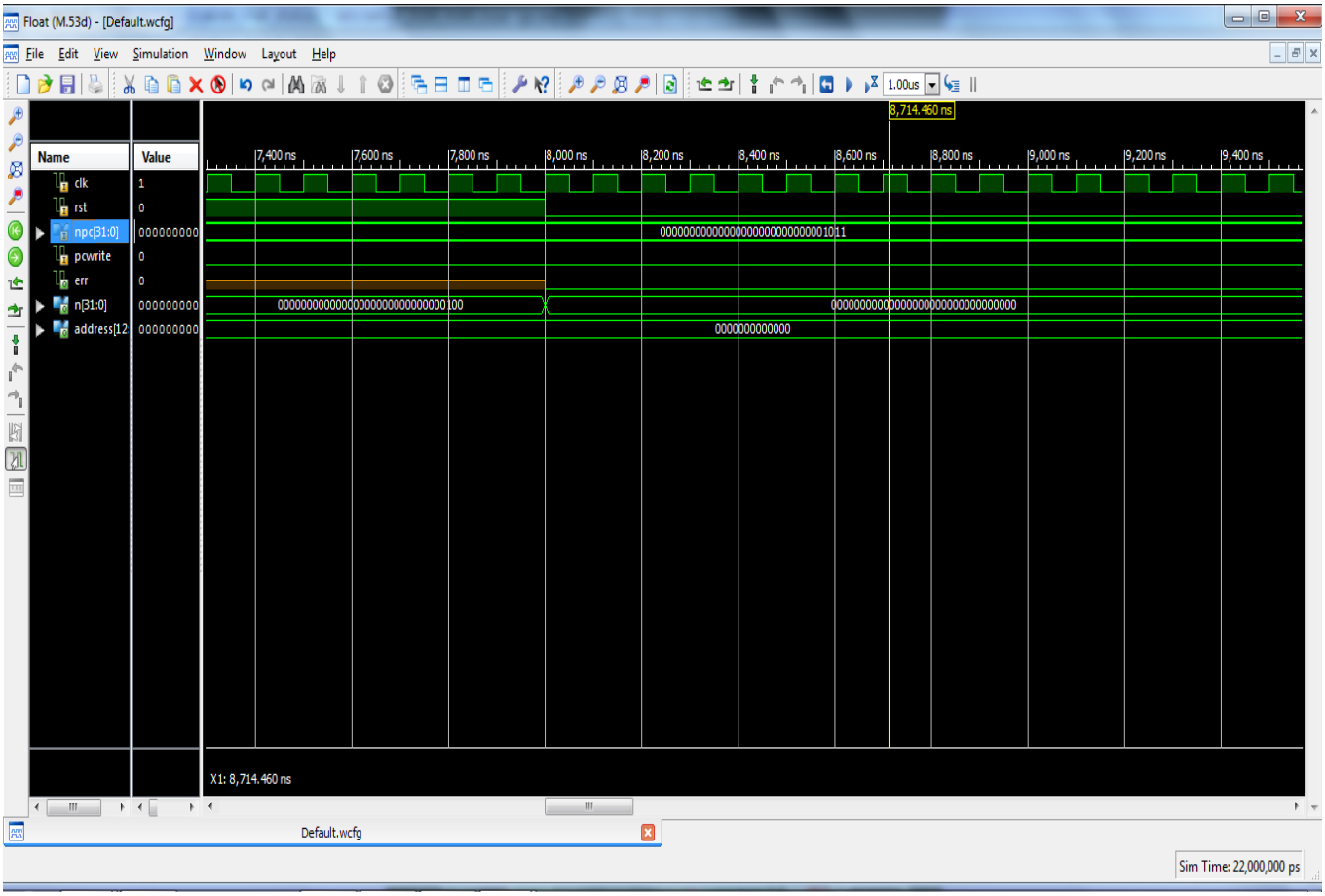

Fig 6.4 Memory Circuit output waveform

6.5 SIMULATED RESULT OF SF NOC SYSTEM

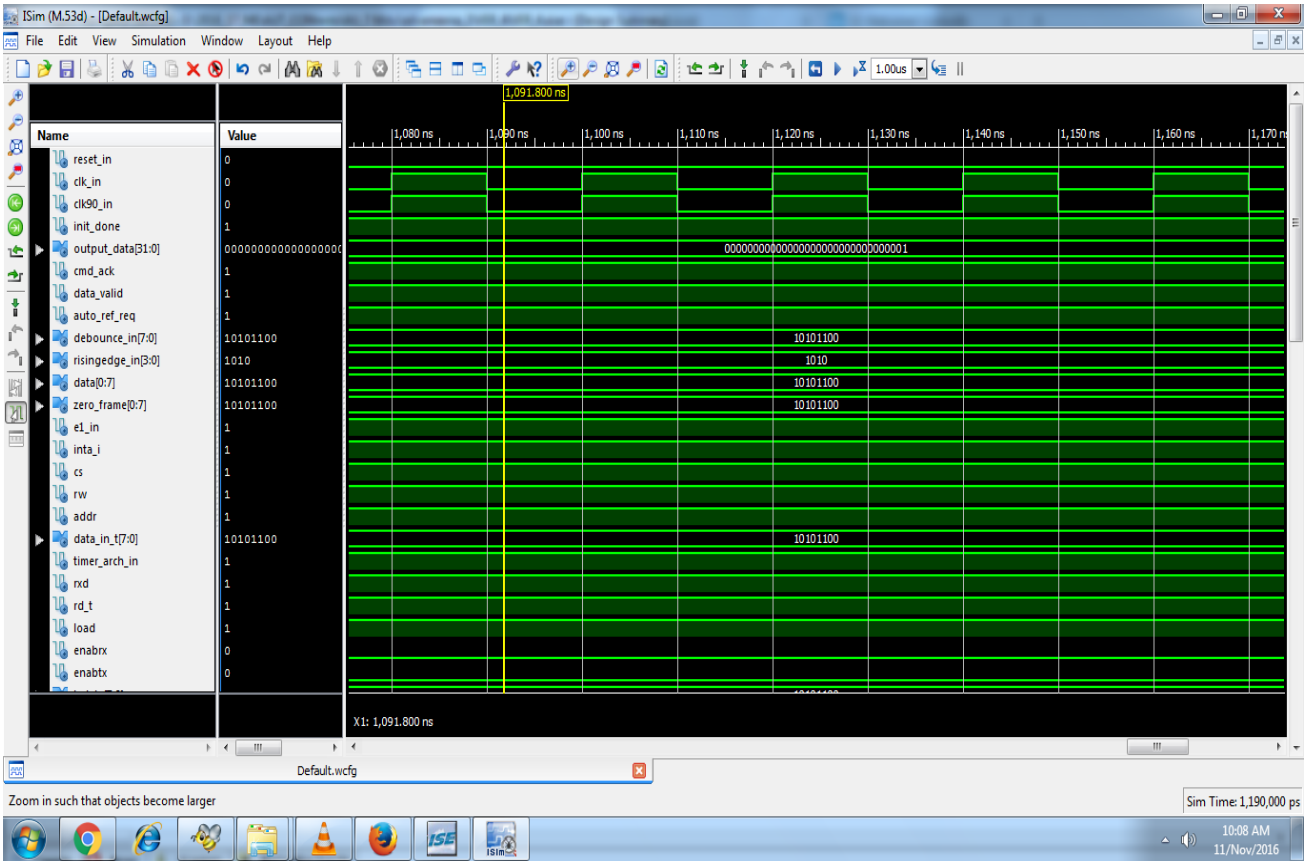

Fig 6.5 Simulated Result of SF NOC System

6.6 SYNTHESIS RESULT OF SF NOC SYSTEM

6.6.1 AREA ANALYSIS

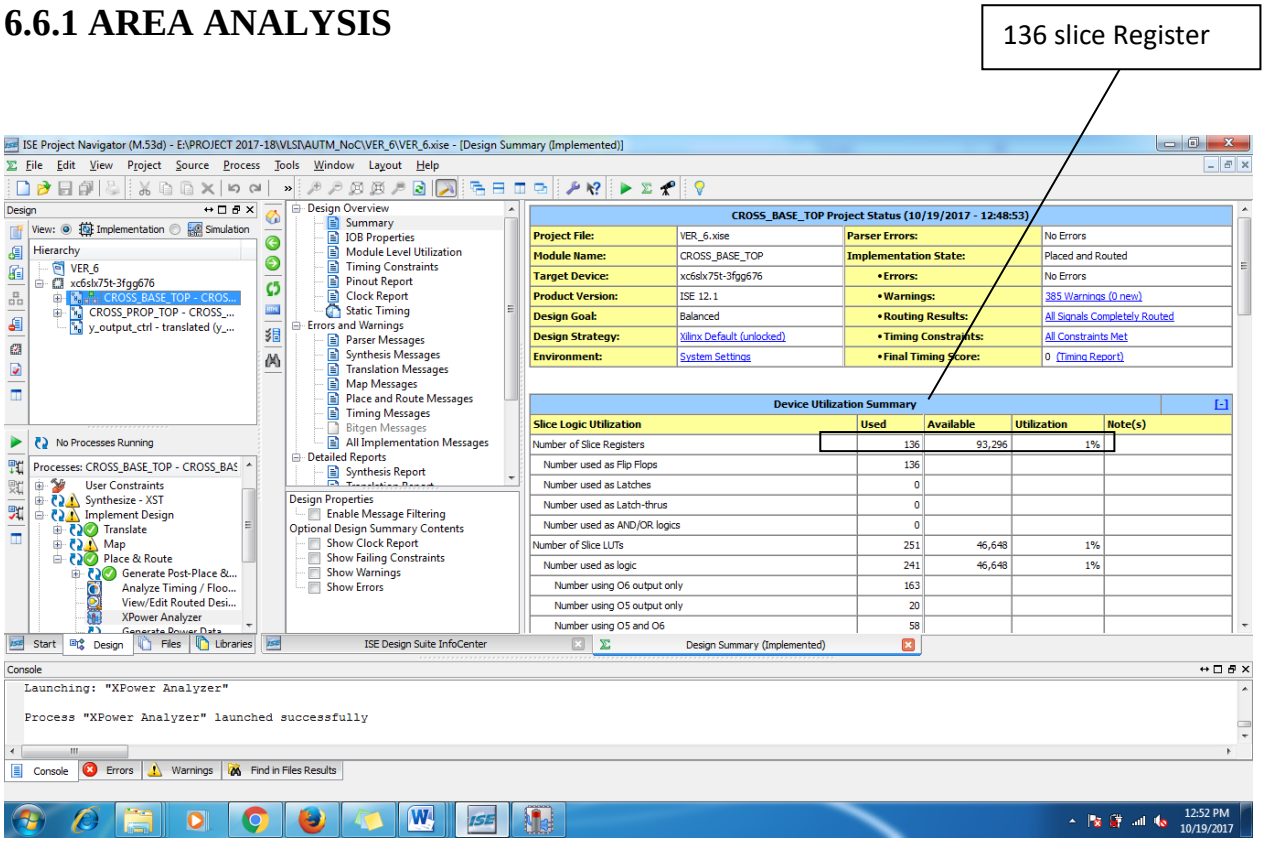

Fig 6.6.1 Area Analysis of SF NOC System

The Area analysis of SF NOC system is shown in above figure 6.6.1. This analysis mainly concentrated on the total number of slice registers and the slice LUTs in the entire system and the number of slice registers and the slice LUTs used to obtain the output. Hence, the available numbers of slice registers are 93,120 and the used numbers of slice registers are 136.

6.6.2 SPEED ANALYSIS

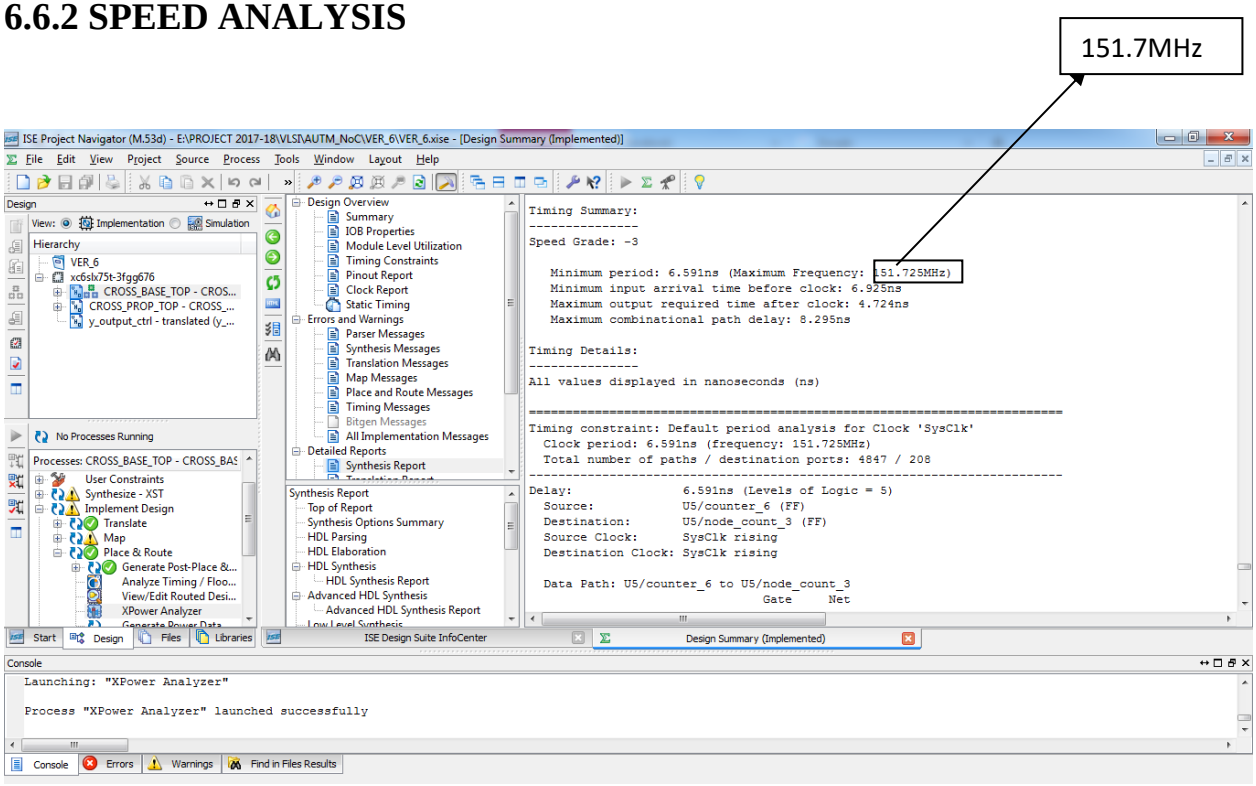

Fig 6.6.2 Speed Analysis of SF NOC System

The Speed analysis of SF NOC is shown in the above figure 6.6.2. This analysis depends upon the maximum range of frequency. Hence, the maximum range of frequency used to generate the output is 151.7MHz.

6.6.3 POWER ANALYSIS

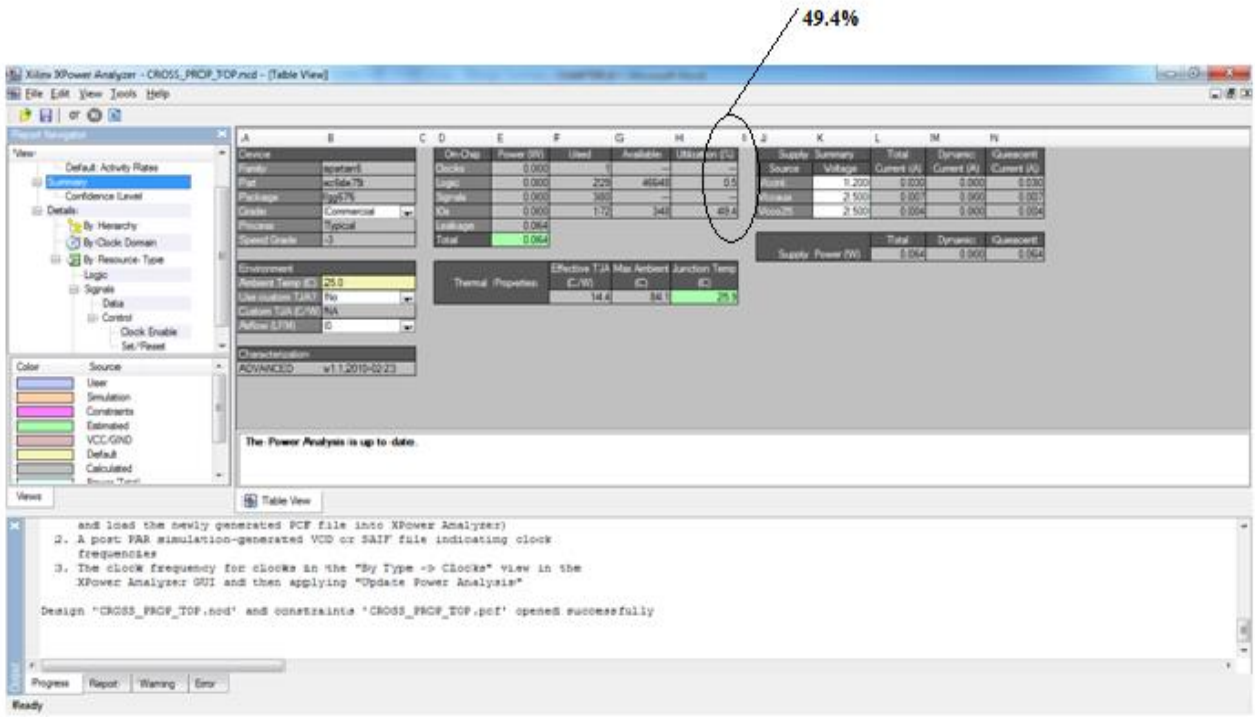

Fig 6.6.3 Power Analysis of SF NOC System

The Power analysis of SF NOC is shown in above figure 6.6.3. This analysis mainly concentrated on the number IOs in the entire system to obtain the output. In the entire system, the available numbers of IOs are 348 and the IOs used to obtain the output are 180. The utilization of power is 49.4 %.

6.6.4 DELAY ANALYSIS

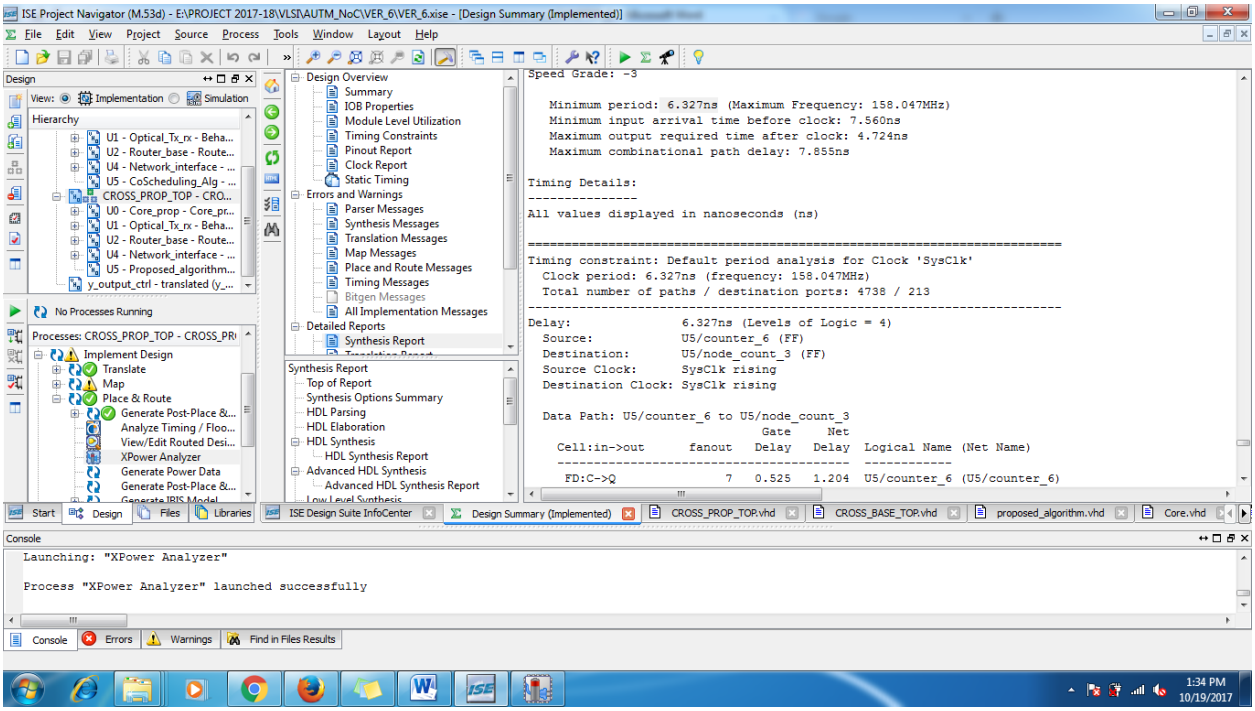

Fig 6.6.4 Delay Analysis of SF NoC System

The Delay analysis of SF NOC system is shown in above figure 6.6.4. The Main concentration of delay analysis is based on the time taken to obtain the output from the time of applying input. Hence, the total time required to XST the completion is 6.591ns.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 CONCLUSION

The combination of layer-specific diagnosis techniques into a multi-layer diagnosis approach is essential to find pareto-optimal solutions that offer a good tradeoff between sys-tem performance and diagnosis quality. The performance such as speed can be increased ,consumption of power is reduced and delay can be decreased by combining a lower layer technique with a second one on higher layer. In this setting, the technique on higher layer helps achieve good performance by limiting the performance impact of diagnosis thanks to a reduced fault localization effort. On the other hand, the technique on lower layer ensures a high diagnostic quality, enabled by the more detailed information available on the lower layer. The combination of a software based diagnosis protocol on transport layer with either functional diagnosis on network layer or structural diagnosis on the physical layer turn out to be the most promising multi-layer diagnosis techniques.

7.2 FUTURE WORK

In future multilayer faults can be identified in NoC and the speed can be further improved and the power can be decreased.

REFERENCES

1. Abbas, A., Ali, M., Fayyaz, A., Ghosh, A., Kalra, A., Khan, S.U., Khan, M.U.S., De Menezes,T., Pattanayak, S., Sanyal, A. and Usman, S., "A survey on energy-efficient methodologies and architectures of network-on-chip" in Proceedings International Symposium Computers & Electrical Engineering, 2014 pp.333-347.

2. Ancajas, D. M., Chakraborty, K., & Roy, S. "Fort-nocs: Mitigating the threat of a compromised noc". In *Proceedings of the 51st Annual Design Automation Conference* (pp. 1-6).ACM (2014, June).

3. Baron S, Wangham MS, Zeferino CA. "Security mechanisms to improve the availability of a Network-on-Chip.InElectronics, Circuits, and Systems (ICECS), 2013 IEEE 20th International Conference on 2013 Dec 8 (pp. 609-612). IEEE.

4. Grammatikakis,M.D.,Papadimitriou, K., Petrakis, P., Papagrigoriou, A., Kornaros, G., Christoforakis,I.,Tomoutzoglou, O., Tsamis, G. and Coppola, M., "Security in MPSoCs: a NoC firewall and an evaluation framework". IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2015,34(8), pp.1344-1357.

5. Kumar, N. and Nene, M.J., 2017, January.Chip-Based Key Distribution Technique for Security Enhancement in Hierarchical Wireless Sensors Networks. In Advance Computing Conference (IACC), 2017 IEEE 7th International (pp. 333-338). IEEE.

6. Jiang N, Balfour J, Becker DU, Towles B, Dally WJ, Michelogiannakis G, Kim J. "A detailed and flexible cycle-accurate network-on-chip simulator. InPerformance Analysis of Systems and Software (ISPASS)", 2013 IEEE International Symposium on 2013 Apr 21 (pp. 86-96). IEEE.

7. Pierce D, inventor; Mworks Worldwide, Inc., assignee. “Integrated enterprise software and social network system user interfaces utilizing cloud computing infrastructures and single secure portal access”. United States patent US 9,003,297. 2015 Apr 7.

8. Sepulveda J, Gogniat G, Flórez D, Diguet JP, Zeferino C, Strum M. Elastic “security zones for NoC-based 3D-MPSoCs. InElectronics, Circuits and Systems (ICECS)”, 21st IEEE International Conference on 2014 Dec 7 (pp. 506-509). IEEE.

9. Wassel HM, Gao Y, Oberg JK, Huffmire T, Kastner R, Chong FT, Sherwood T. “SurfNoC: a low latency and provably non-interfering approach to secure networks-on-chip”. InACM SIGARCH Computer Architecture News 2013 Jun 23 (Vol. 41, No. 3, pp. 583-594).ACM.

10. Zhao K, Ge L. “A survey on the internet of things security.InComputational Intelligence and Security (CIS)”, 2013 9th International Conference on 2013 Dec 14 (pp. 663-667). IEEE.