특정 용도를 갖는 네트워크 온 칩을 위한 트래픽 모니터링 시스템

Traffic Monitoring System for Application-Specific Networks-on-Chip

Traffic Monitoring System for Application-Specific Networks-on-Chip

Advisor : **Professor Hoi-Jun Yoo**

by

Kwanho Kim

Department of Electrical Engineering and Computer Science
Division of Electrical Engineering
Korea Advanced Institute of Science and Technology

A thesis submitted to the faculty of the Korea Advanced Institute of Science and Technology in partial fulfillment of requirements of the degree of Master of Engineering in the Department of Electrical Engineering and Computer Science, Division of Electrical Engineering

Daejeon, Republic of Korea
2005. 12. 21
Approved by

_____

Professor Yoo, Hoi Jun

특정 용도를 갖는 네트워크 온칩을 위한 트래픽 모니터링 시스템

김 관 호

위 논문은 한국과학기술원 석사학위논문으로 학위논문 심사위원회에서 심사 통과하였음.

2005년 12월 21일

심사위원장 유 회 준 (인)

심사위원 신 영 수 (인)

심사위원 정 송 (인)

MEE

20043049

김 관 호, Kim, Kwanho. Traffic Monitoring System for Application-Specific Networks-on-Chip. 특정한 용도를 갖는 네트워크 온칩을 위한 트래픽 모니터링 시스템. Department of Electrical Engineering and Computer Science, Division of Electrical Engineering. 2006. 47p Advisor Prof. Yoo, Hoi-Jun. **Text in English**

Abstract

A true application-specific Network-on-Chip (NoC) requires the exact knowledge of the internal traffic behavior during the NoC is fully running the target application. Therefore, an accurate traffic measuring method is necessary. In this work, a simple but powerful traffic monitoring system is presented for the accurate evaluation and refinement of an application-specific NoC. The traffic monitoring system measures various traffic parameters such as an end-to-end latency, a queuing buffer usage and a congestion level. A NoC-based portable multimedia system as a target system is implemented on multiple FPGAs to demonstrate the effectiveness of the proposed traffic monitoring system. Using the traffic monitoring system, the target system is diagnosed and refined in two different topologies; mesh and star topologies. For the application-specific NoC, three design refinement processes such as buffer size assignment, network frequency selection, and run-time routing path modification are performed in the target system. As a result of the refinement, 42% buffer reduction and 28% latency reduction are obtained in the mesh-connected application-specific NoC.

사랑하는 가족에게
이 논문을 드립니다.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

1.1 Motivation

As the complexity of systems-on-chips (SoCs) increases together with the interconnection issues of new IC technology generation, Networks on Chip (NoC) have been proposed as a new on-chip communication architecture to overcome the limitations of bus-based on chip interconnect [1-3]. Based on a packet-switching fabric structure, the NoC provides sufficient bandwidth and scalability for high performance SoCs.

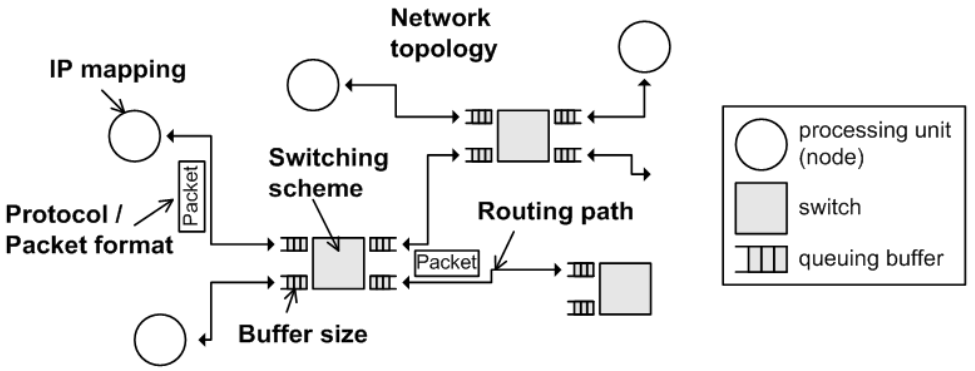

Fig. 1.1 NoC design parameters

NoC involves a complex design process such as a selection of a topology, protocol, and buffer size. Suitable selection of such design parameters as shown in Fig 1.1 is an integral part for the efficient design of an application specific NoC. For this purpose, observation of NoC traffic is required because the suitable selection of

design parameters is heavily affected by the on-chip traffic pattern. NoC design parameter selection can be achieved by two different approaches: static and dynamic method. As a static approach, [4] and [5] show the optimized NoC design by analytic analysis under the assumption of theoretical traffic pattern such as a uniform and Poisson distribution. However, this approach is not suitable for the design of a real NoC-based system because the traffic pattern of the real system cannot be described theoretically. On the contrary, a dynamic approach uses exact knowledge of NoC traffic and is also classified into two groups: a simulation-based and emulation-based approach. In a simulation-based approach, probing all nodes of NoC is possible, therefore, the internal traffic can be observed accurately [6-7]. However, it is unrealistic because it takes enormous time to obtain the details of the traffic with the real application software. Some previous work presented a NoC emulation environment implemented on a FPGA to evaluate various custom NoC solutions within a very fast execution time [8]. However, in this work, NoC is considered as a black box, thus, only end-to-end traffic pattern can be observed at the interfaces of NoC. Moreover, the emulation platform is constructed using traffic generator and receptor without real processing cores. As a result, a NoC traffic monitoring system which can measure accurate and detailed internal traffic behavior quickly is essential for application-specific NoC.

Recently, [9] presented the concepts of NoC monitoring service that offers communication observability and can be configured at run-time. However, this approach does not show detailed monitoring results about NoC traffic behavior, therefore, a design refinement for application-specific NoC is impossible.

In this work, a NoC traffic monitoring system is presented to observe the internal traffic behavior of NoC core. The monitoring system can get traffic information such as end-to-end latency and queuing buffer utilization using traffic probe attached to NoC components. In addition, the proposed monitoring system has very modular structure to be instantiated for any NoC design blocks such as a network-interface (NI) and switches. I

prove the effectiveness of the traffic monitoring system via NoC-based portable multimedia SoC implemented on multiple FPGAs. Through the analysis of the monitoring results, I can estimate the potential bottleneck of the NoC and also refine the design by cutting the overestimated NoC resources off. As a result, cost-efficient and higher-performance NoC design is possible thanks to the traffic monitoring system.

1.2 Advantage of Traffic Monitoring System

A traffic monitoring system enables the design of highly-optimized application-specific NoC because it: (a) evaluates and verifies the NoC accurately while the NoC is running the target application, (b) supports the design parameter selection based on a real traffic pattern, and (c) allows iterative design refinement processes by fast evaluation speed.

Fig. 1.2 shows an application-specific NoC design flow. In existing classic NoC design, design parameters such as topology, IP mapping, and routing are determined by static analysis based on application requirements. On the contrary, the NoC with a traffic monitoring system is able to tune the design parameter to a specific application more finely due to the advantages described above. As a result, a true application-specific NoC is generated with the help of the traffic monitoring system.

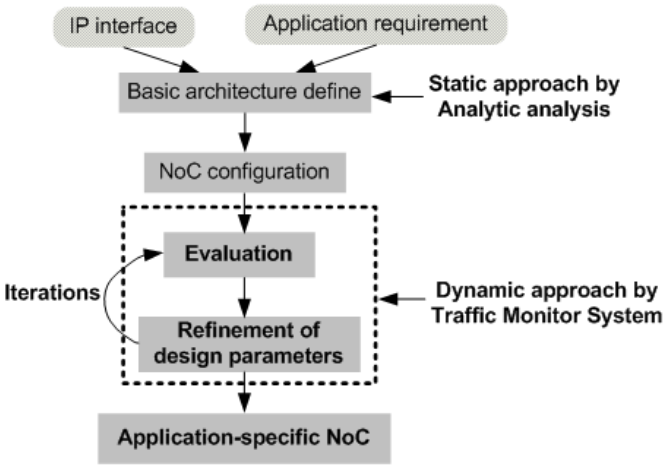

Fig. 1.2 Application-specific NoC Design Flow

1.3 Thesis Organization

This work is organized as follows: In Chapter 2, the NoC architecture and protocol are described. In Chapter 3, the proposed NoC traffic monitoring system is presented. In Chapter 4, the usefulness of the traffic monitoring system is demonstrated with various experimental results in a real NoC-based system. In Chapter 5, FPGA implementation is explained. Finally, conclusion will be made in Chapter 6.

CHAPTER 2

NoC Architecture and Protocol

2.1 On-chip Network Architecture

Fig. 2.1 shows the overall architecture of On-chip Network (OCN). OCN consists of 5 kinds of components: Master Network Interface (MNI), Slave Network Interface (SNI), Up_Sampler (UPS), Dn_Sampler (DNS), and Switch (SW). The MNI connects a master to the OCN. Using the UPS and DNS, the OCN serializes and deserializes packets. Such an on-chip serialization reduces the area of OCN significantly and lowers wiring complexity by reducing the number of link wires. Instead, it uses its own high frequency clock to sustain bandwidth of the network system. The non-blocking SW routes packets. The OCN has two separate networks, forward and backward. The forward network deals with a path from MNI to SNI, and the other a reverse path from SNI to MNI.

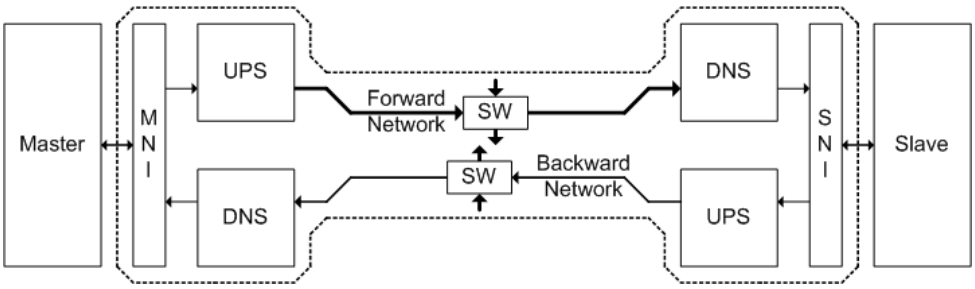

Fig. 2.1 Overall architecture of OCN

2.2 Packet Format

Fig 2.2 shows a packet format, which consists of a header, a timer and two payloads. The header is used for end-to-end network interfaces (NIs) and switches. 8b timing value is used as a time-stamp for traffic monitoring. The PL1 and PL2 carry 32b address and data.

The header supports various functionalities for packet transfer. A packet is transferred to a destination through a fixed routing path according to RI field in the header. An NI has an address map which defines a destination PU according to an address, and the RI field is generated based on the address map. BL field contains a burst length for burst transaction. 1-level packet priority and handshaking protocol are supported for quality-of-service (QoS) control and reliable transport, respectively. The packet header size is fixed to reduce latency and hardware complexity of header parsing logic.

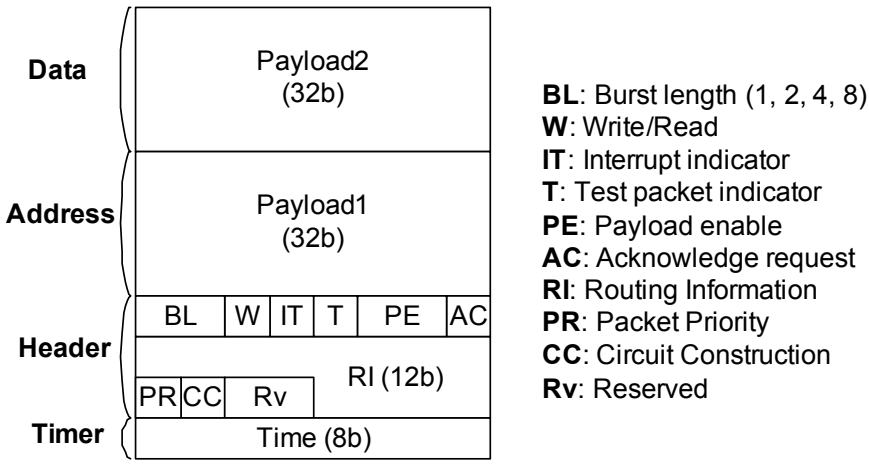

Fig. 2.2 Packet format

Whether packet length is fixed or variable affects complexity of packet parsing logic and serialization method. In this work, field-based serialization (FBS) as shown

in Fig.2.3 is used. In the FBS, which can be applied to fixed-length packet serialization, dedicated link wires are assigned to each packet field. Advantages of this scheme are as follows: 1. Packet parsing procedure is very simple, and 2. bitwidth of a field can be easily increased with additional link wires. A disadvantage of this scheme is that its link utilization is inefficient if some fields are disabled. For example, if a packet carries the payload2 (PL2) as it is empty, the link<21:14> remains idle but other packet transactions can not use the idle link wires.

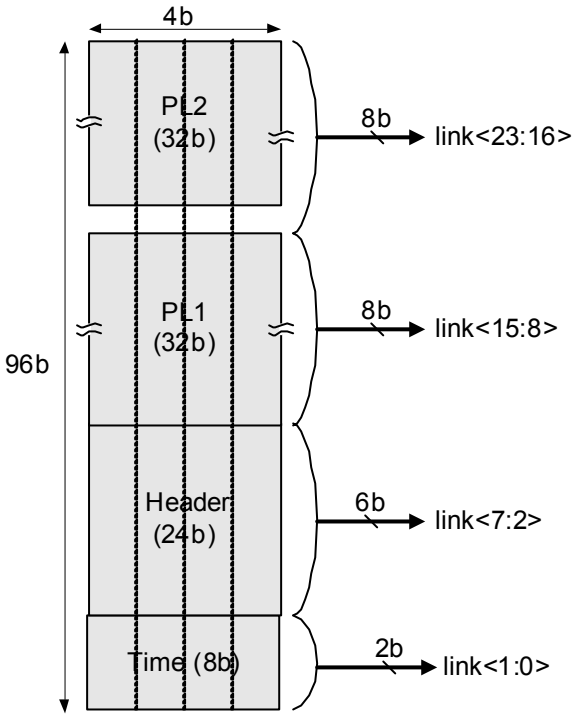

Fig. 2.3 Field-based serialization

2.3 NoC Protocol

Three special functions for end-to-end packet transaction are described. Multiple-outstanding-addressing, write with acknowledge, and burst packet transfer are defined to provide high-bandwidth energy-efficient, and reliable packet delivery.

Detail description of each function follows below.

2.3.1 Multiple-Outstanding Addressing

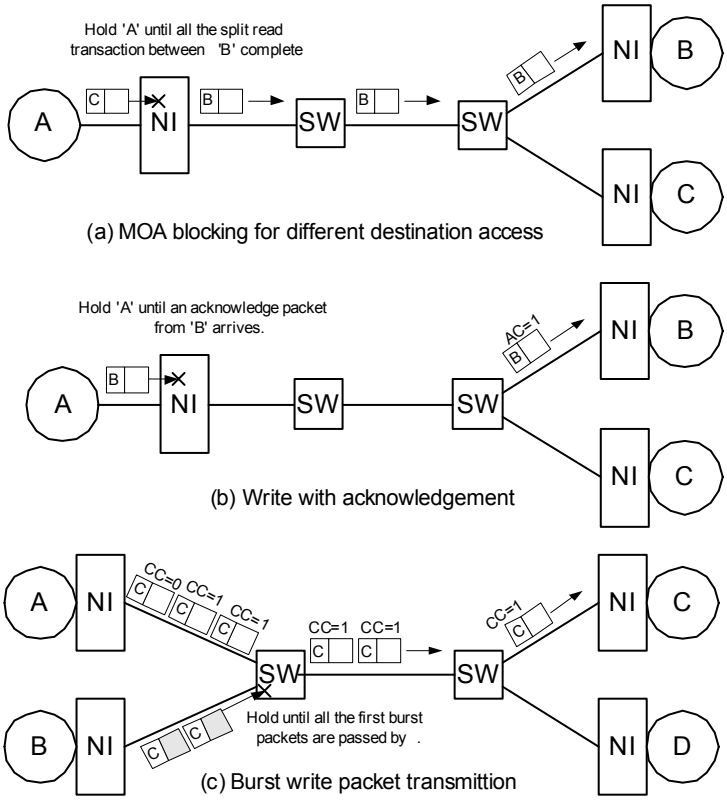

Fig. 2.4 NoC protocol with three end-to-end services

When a source PU issues a read command to a destination, multiple-outstanding-addressing (MOA) enables additional issues of read commands before the previous read transaction completes. The MOA scheme, which is well-known technique in advanced Bus architectures [33], hides the latency of a communication channel thus increases throughput. In order to use the MOA scheme, however, packet ordering issue must be resolved. In this work, NI controls it as shown in Fig. 2.4-(a): An NI inspects the destination of a read command packet, and

if successive read command packets target the same destination, MOA is admitted. Since the network provides a single fixed path for a single destination in this work, packet sequence is not modified in the network. However, if the destinations of two packets are different, their latency would be different so that the orders of packet issue and arrival may not be the same. Therefore, if a following read command packet targets a different destination, the NI holds the packet transmission of the source thus the MOA is blocked. The NI holds the transmission until all the previous packet transactions complete.

2.3.2 Write with Acknowledge

Since a network has a variable latency, a source PU does not know when its write packet has arrived to the destination. In some applications, a PU must perform certain operation after writing data thus it needs to know the timing the write packet arrives at the destination. For example, let us think about the situation that a microprocessor transfers data to a memory and issues a command packet to a dedicated hardware unit to process the data in the memory. In this case, the command packet should be transferred after the last data is delivered to the memory. The acknowledge request (AC) field in the packet header shown in Fig. 2.2 is used in such a situation. When the AC field is enabled, the destination NI returns an acknowledge packet to inform to the source NI that the packet has arrived at the destination. The source NI holds the packet transmission of the source PU when it transmits a packet with AC, and resumes transmission when it receives the acknowledge packet.

2.3.3 Burst operation

In order to support bulk data transfer for multimedia signal processing applications, burst read and write packet transfer modes must be supported. In a burst

read mode multiple sequential data are accessed using a single read address as a base address and incrementing pre-defined offset which is 4 typically. In a burst write mode multiple data are written in series with a single reference address and the pre-defined increment offset. For the burst mode packet transfer, an issue is that the flow of the series data must not be interrupted by other packet flow. The interruption means violation of burst mode protocol.

In this work, burst packet flow is protected from interrupt by holding arbiter operation. The CC header fields (See Fig. 2.2) of every packets except the last packet of the burst packet flow are enabled. When the CC field is enabled, the arbiter of a switch holds its grant output thus the output port assigned to the burst packet flow are not switched to other inputs until the arrival of the last packet.

In a specific situation, a burst read flow does not need to be protected. If PUs are categorized into master and slave groups, the burst read flow occurs only from slaves to masters. And, a master is expected to receive packets that it has initiated. In other words, when a master issues a burst read packet to a destination, packets come only from the destination. This implies that even if the burst packet flow from a slave to a master is interrupted by other flows, a master will receive burst data sequentially.

CHAPTER 3

NoC Traffic Monitoring System

3.1 Traffic parameters for On-chip Network

Three traffic parameters (1) an end-to-end latency of each transaction, (2) a backlog in each queuing buffer and (3) output-conflict rate at each switch output port are traced at runtime as the network performance indicators. (See Fig. 3.1) In addition, I can also measure a dynamic statistics such as a communication bandwidth between integrated Intellectual Properties (IPs), an evaluation time of the application and link/switch/buffer utilization.

Firstly the end-to-end latency represents one of the most important performance parameters of the NoC platform. The predictable and deterministic latency is a crucial requirement for real-time applications and also for the simplicity of software programming. Therefore, the latency and its variations should be carefully considered during the network topology design and IP mapping process. The proposed latency monitoring system provides the complete framework for the network optimization in the real application.

The second parameter is backlog i.e. the number of packets that are occupying an input queuing buffer of a switch. The queuing buffers occupy the majority of the on-chip-network and the amount of queuing buffers affects the network performance especially when the network becomes congested. Therefore, the buffer capacity must be determined as minimum without sacrificing network performance significantly. The monitoring of the queuing buffer backlog helps designers to find the optimal buffer size while the target application is running on the system.

The last parameter is output conflict status on a switch output port. The rate of output conflict represents the degree of traffic congestion on a specific path. By monitoring output conflict status, I can identify frequently congested path and consequently the detouring path between IP can be selected to reduce the traffic congestion at design-time or even run-time.

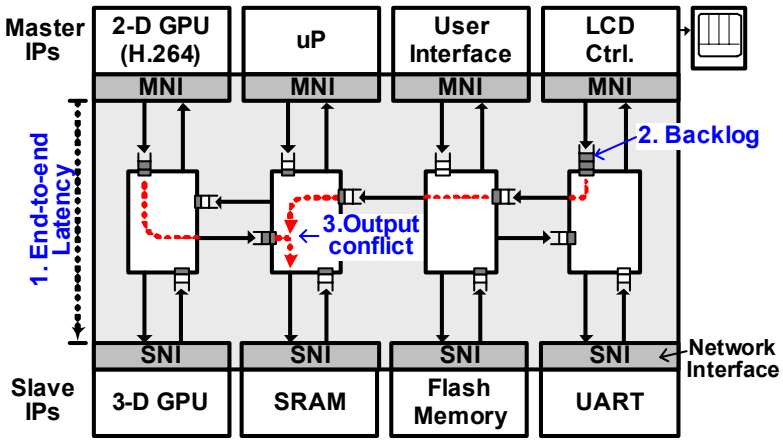

Fig. 3.1 Measured Traffic parameters

3.2 S/W-controlled Traffic Monitoring System

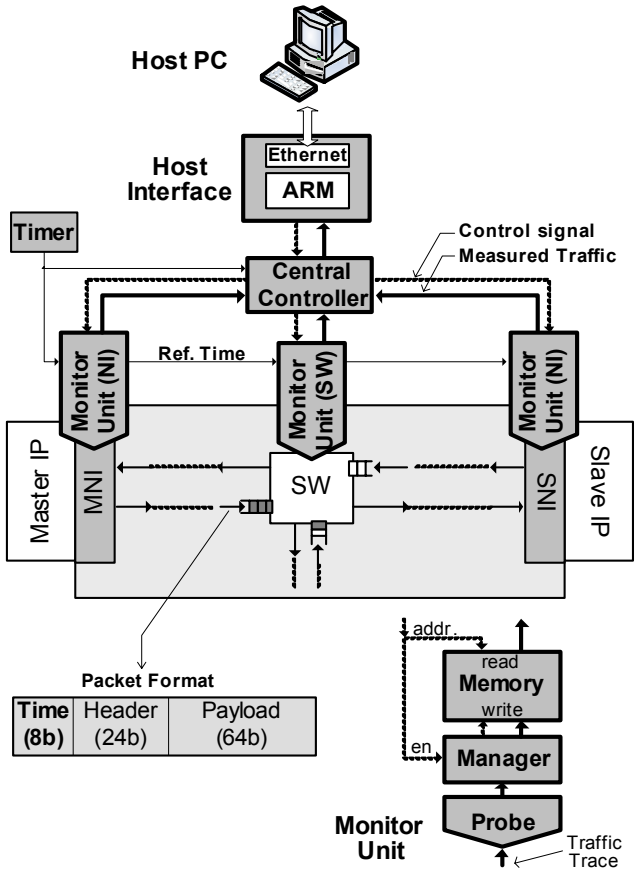

Fig. 3.2 Proposed Traffic Monitoring System

Fig. 3.2 shows the overall structure of the proposed traffic monitoring system. It consists of three sub-systems: Host Interface, central controller and Traffic Monitoring Units.

The Host Interface, a bridge between the central controller and a host PC, transfers traffic monitoring results to the host PC via Ethernet. The central controller enables/disables each monitoring unit based on the requested monitoring regional scope and time interval.

A monitor unit consists of a traffic probe, a traffic manager, and an on-chip memory. A traffic probe module is connected to a switch or a network interface in order to trace the real-time traffic parameters such as an end-to-end latency, a backlog and an output conflict

rate on a switch. Then, the traffic manager stores the traces in its local trace memory after attaching a time-stamp to each trace using a global timer connected to all monitor units. During the operation of the specific application, all monitoring results are accumulated in the corresponding local memory which is accessible by Host Interface's ARM via the central controller. Since all the monitoring processes do not have any influence on the NoC packet flow, non-intrusiveness probing is achieved.

The traffic monitoring system has modular architecture. Thus a monitor unit can be attached to any NoC components, which is a design-time choice. In the following subsection, I describe the detailed method for measuring the three traffic parameters.

3.3 Detailed Monitoring Method

Detailed monitoring method for three traffic parameters is shown in Fig. 3.3.

In order to measure end-to-end latency of an individual packet, 8-bit timing information is added to the original packet format as shown in Fig. 3.2. (Therefore the maximum measurable latency is 2^8 = 256-cycles) When a packet is injected into the network from a master, the lower 8-bit of the timer value is recorded in a master network interface (MNI) and the time-stamped packet is transferred to the slave IP. As a result, the end-to-end latency can be measured by the difference between the stamped time on the packet and the arrival time at slave network interface (SNI). The backlog, the number of packets stored in a switch input buffer, can be also measured by the difference between the FIFO read and write pointers. The output conflict status can be traced by detecting the duplicated request signals on the same output port arbiter.

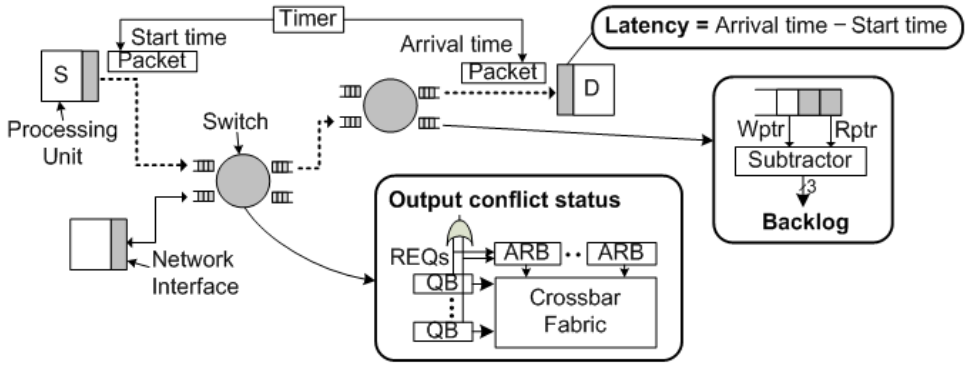

Fig. 3.3 Detailed Monitoring Method

The measured traffic parameters are stored in a local trace memory as regular formats shown in Fig. 3.4. To save the memory capacity, the trace parameters are recoded only when their values are changed. Speaking of the end-to-end latency, the consecutive same latencies are recoded only once with the number of the packets. The number of backlog of a queuing buffer is recoded only when the queue length has a transition. In this format, the maximum measurable number of input ports on a switch is five. When there is an output conflict event in a switch, the corresponding bit out of 8-bits in the form is recoded as 1 while others are 0s.

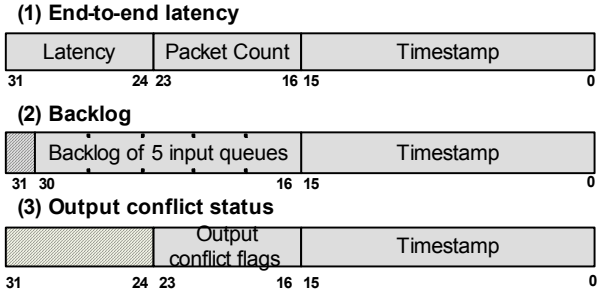

Fig. 3.4 Traffic trace format in trace memory

In case of long-term tracing, it is difficult to store all of the measured traces in a local

memory due to the limitation of on-chip memory capacity. For such a case, a traffic statistics is calculated by accumulating and counting of incoming measured traffic data without recording all the dynamic traces. As the traffic statistics, an average or variance of latency, backlog and output conflict rate can be obtained.

3.4 NoC Evaluation Flow

For application-specific NoC, accurate NoC evaluation and verification is required. Using a traffic monitoring system, the NoC evaluation is performed with concurrent H/W and S/W co-design flow as shown in Fig. 3.5.

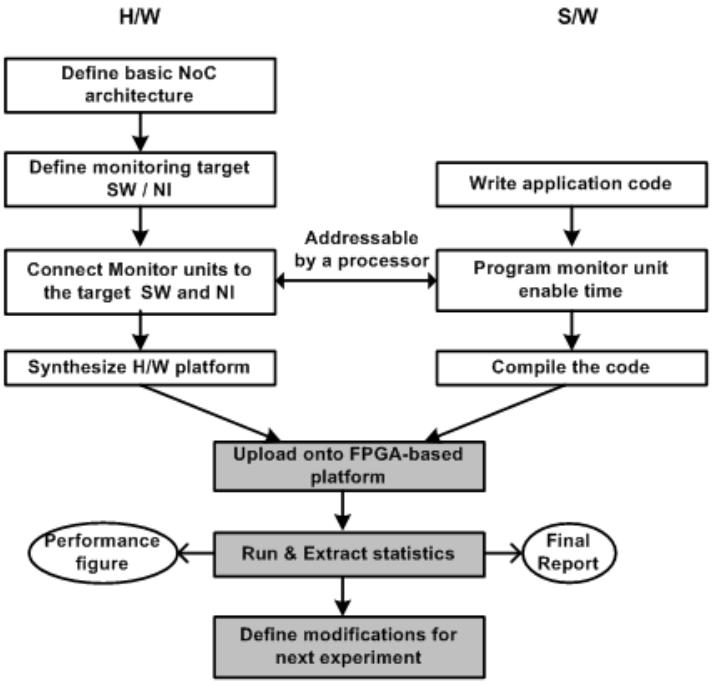

Fig. 3.5 NoC evaluation flow using a traffic monitoring system

In the H/W flow, the first phase is to define a basic NoC architecture by selection of NoC design parameters such as topology, protocol, and buffer size. In a selected topology,

IP-mapping is also performed in this phase. After that, the designer specifies which switches and network interfaces are the targets of traffic monitoring and connects Monitor Units in Fig.3.2 to the corresponding components. Finally, the NoC H/W platform with a traffic monitoring system is configured and synthesized with the standard tool (Quartus) provided by the FPGA supplier.

In the S/W flow, the first phase is to write application code of the final NoC-based system. Then, the designer programs enable or disable time of each target monitor unit which is addressable by Host Interface's ARM in the initialization section of the application code. After that, the program code is compiled and binaries are generated by compiler tool of RISC processor.

Finally, both the FPGA synthesis of the H/W platform and the compiled code of the application are uploaded onto the FPGA-based platform through an Ethernet of Host Interface. After the overall NoC platform is constructed, the system runs automatically while traffic statistics are extracted and sent to the host PC. In case designer wants to modify the executed application, no time-consuming H/W re-synthesis is required, therefore, various applications can be tested in few minutes using the traffic monitoring system.

Table. 3.1 Timing comparison between H/W monitoring and simulation

Application / Mode	Small application	Full application
H/W Monitoring	3.3sec	10.8sec
Simulation (Verilog-XL)	41hour 52min	136hour (Estimated)

* Small application : Simple loop program
  Full application : 3D graphics application program

Timing comparison results between H/W monitoring and simulation approaches are shown in Table 3.1. The NoC evaluation time by the H/W monitoring is about 45,000 times

faster compared to the cycle-accurate simulator. Therefore, the H/W monitoring framework is essential to evaluate a variety of NoC solutions for the target application on a real NoC-based system.

CHAPTER 4

Application to Portable Multimedia System

4.1 Target System

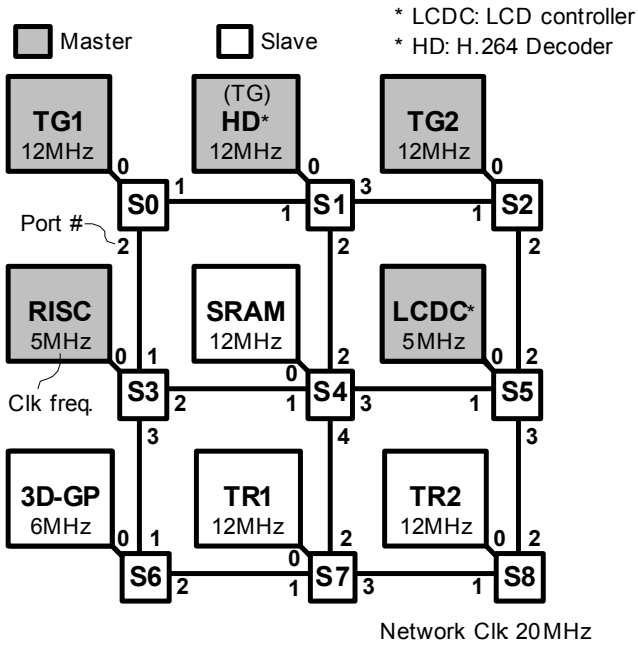

Fig. 4.1 Portable Multimedia System in Mesh topology

A portable multimedia system is implemented to demonstrate the effectiveness of the proposed traffic monitoring system. The implemented system includes various IPs; five masters (RISC CPU, LCD controller, H.264 decoder, and two traffic generators (TG)) and four slaves (3-D graphics processor [10], SRAM, and two traffic receptors (TR)) as shown in Fig. 4.1.

Table 4.1 Traffic flows of the target system

Guaranteed Throughput (GT) traffic

Master	Slave	Required bandwidth
RISC	SRAM	52Mbps
RISC	3D G.P	65Mbps
HD	SRAM	10Mbps (Decoding) 128Mbps (Frame Writing)
LCDC	SRAM	52Mbps

Best effort (BE) traffic

Master	Slave	Traffic characteristic
TG1	TR1	Uniform (Offered load 0.4)
TG2	SRAM	Burst (length 4)

The H.264 decoder (HD) is replaced with a traffic generator since the H.264 IP needs too huge logic-element resources to be implemented on FPGA board. However, the traffic generator produces the real traces assuming that it decodes CIF (352*288) H.264 baseline profile at level 2 with 30frames/sec. The SRAM is used as a display frame buffer for HD and the 3-D graphics processor (3D-GP). The LCD controller directly reads the frame data from the SRAM with burst operation (burst-length = 8) continuously. This transaction has a hard real-time requirement to keep the display frame rate (30frames/sec). Two TGs generate uniform traffic with offered load of 0.4 and burst packets with burst length of 4. Table 4.1 shows two different kinds of traffic flows in this application: four guaranteed throughput (GT) traffics for meeting the application requirements and two best effort (BE) traffics emulating other applications in the portable multimedia system.

The operation of this system is as follows. The 3D-GP is initialized by the RISC and then its instructions are fetched from the SRAM. After the rendering calculation of the 3D-GP, 3-D scenes are stored in the SRAM. All of the

transactions between the 3D-GP and the SRAM are conducted by the RISC because the 3D-GP is designed as a slave IP.

Meanwhile, HD decodes encoded video stream which is already downloaded in the internal memory. During the decoding process, the HD accesses the frame data in the SRAM with the bandwidth of 10Mbps. After the decoding process is completed, the HD writes the decoded 2-D scenes into the SRAM frame memory with 128Mbps bandwidth.

The LCD controller continuously reads the SRAM frame memory and displays the 3D scenes on a LCD screen.

In the following subsections, evaluation and refinement process of the portable multimedia system is performed based on traffic monitoring results with two different topologies; a mesh and a star topology.

4.2 Traffic Measurements and Diagnosis

Table 4.2 shows the overall traffic bandwidth measured by the traffic monitoring system. Fig.4.2 shows latency distribution of three selected flows. The first two traffic flows (HD → SRAM and TG1 → TR1) experience quite large latency and also its large variations. After the decoding process of the HD, the traffic from the HD to the SRAM increases abruptly. Thus the flow from the TG1 to the TR1 is also affected seriously because the two flows are sharing a link between SW1 and SW4. Meanwhile, the third traffic flow from the 3D-GP to the RISC shows smaller and more constant latency since the flow shares the network resource with no one.

Table. 4.2 Measured Traffic Bandwidth

Master	Slave	M → S	S → M
RISC	SRAM	0.52Mpkt/s	0.21Mpkt/s
RISC	3D G.P	0.67Mpkt/s	0.69Mpkt/s

HD	SRAM	1.92Mpkt/s	0.31Mpkt/s
LCD Ctrl	SRAM	1.35Mpkt/s	1.74Mpkt/s

(pkt/s = packet/sec)

Fig. 4.3(a) shows the backlog distribution on the three input queuing buffers. Bursty traffics from the HD and the LCD controller to the SRAM cause the SW4-port N queue to be in the full state. On the contrary, the backlog of SW3-port S queue is almost 0 or 1.

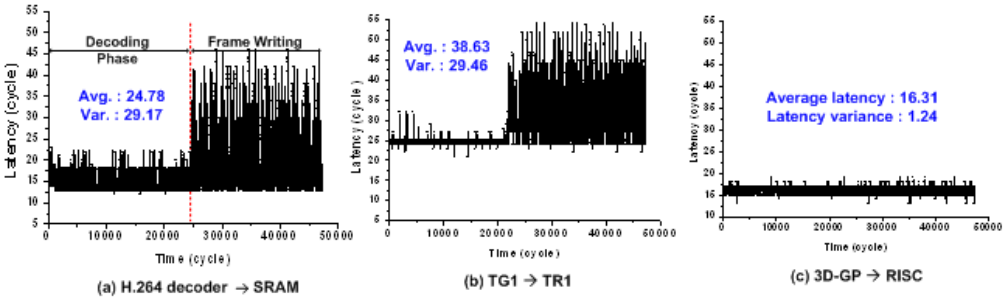

Fig. 4.2 Latency Distribution, Average Latency and its Variation

Fig. 4.3(b) presents the output conflict counts/1,000-cycles on the congested links at SW1 and SW4. After the decoding process is completed in the HD, the output conflict on the shared link increases rapidly (from the 25,000th cycle) and remains as highly congested around 90-conflicts/1,000-cycle.

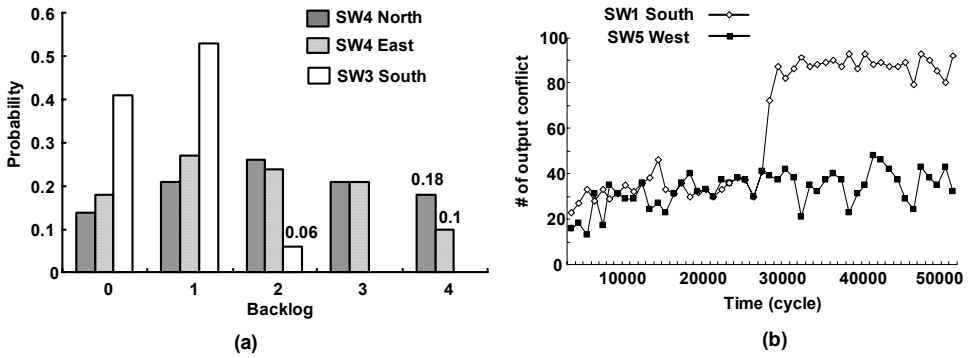

Fig. 4.3 (a) Backlog Distribution and (b) Output Conflict Status

4.3 NoC Design Refinement

In this section, the portable multimedia system is refined in three ways based on traffic monitoring results described above. In the following subsections, three design refinement processes will be presented in detail.

4.3.1 Buffer Size Assignment

The input queuing buffers in a switch take a significant portion of the chip area of the NoC, thus, their size should be minimized without significant performance degradation. The initial NoC design has all input buffers of 4-packet capacity uniformly. Although the uniform choice of the input buffer size is straightforward and widely used in current NoC designs, it may lead to excessive use of silicon area or poor performance. Based on the backlog monitoring results, the minimum buffer capacity of each input queue can be decided by a heuristic algorithm as shown in Fig. 4.4.

Given uniform buffer configuration, buffer size is initially assigned to the backlog value such that P(IB) is the maximum. That means the initial buffer is assigned to the necessary minimum size. In the next step, the buffer which has the largest P(F) is selected as the bottleneck buffer and the buffer size increase by one

packet. The above procedure is repeated until execution time of total application reaches that of uniform buffer configuration as shown in Fig. 4.5.

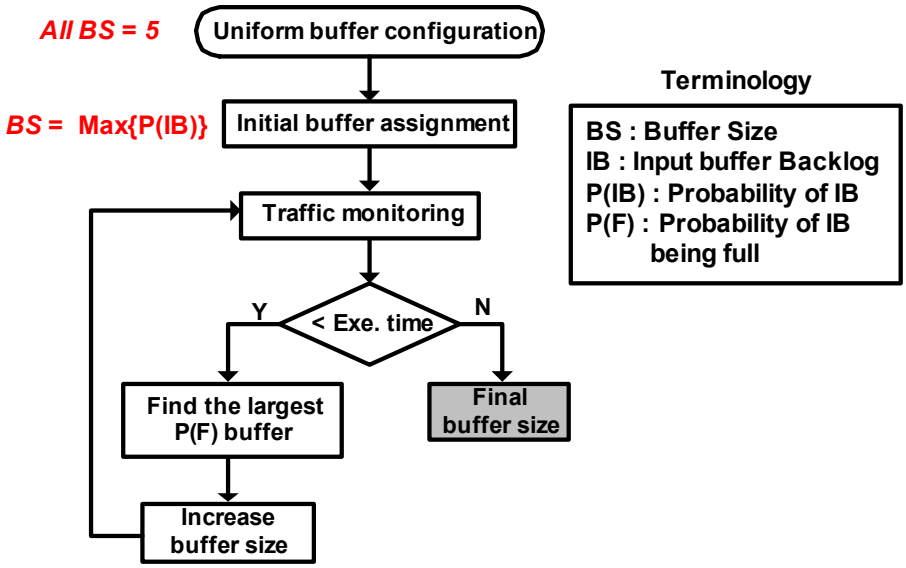

Fig. 4.4 Buffer Size Assignment Policy

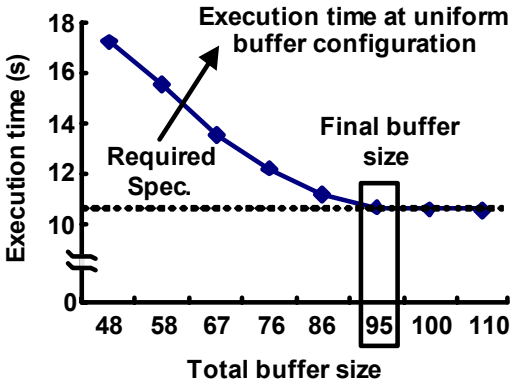

Fig. 4.5 Final Buffer Size Decision Process

Execution time of total application is used as an overall performance metric which is affected by queuing buffer size. If insufficient buffer size is assigned for target application, the probability of queuing buffer being full increases. As a result,

total execution time increases because 1b back-pressure flow control suppresses packet transmission when the queuing buffer is full.

Table 4.3 shows the buffer size assignment results. Buffer size of the congested link (SW4 port E) is increased while size of other buffers was reduced or remained the same. After the buffer size assignment, 42% total buffer size is reduced compared to the uniform buffer assignment.

Table. 4.3 Buffer Size Assignment Results

Port / SW	P	E	W	N	S	Total Buffer	Reduction
SW0	3	2			1	15 → 6	60%
SW1	4	2	4		4	20 → 12	40%
SW2	4		2		4	15 → 10	33%
SW3	4	3		2	3	20 → 12	40%
SW4	4	5	3	5	2	25 → 19	24%
SW5	4		4	5	2	20 → 15	25%
SW6	2	1		2		15 → 5	67%
SW7	2	2	2	4		20 → 10	50%
SW8	1		2	3		15 → 6	60%
Total buffer size						**165 → 95**	**42%**

4.3.2 Network Frequency Selection

Basically, operating frequency can be selected as a minimum frequency that meets the application performance requirements using bandwidth monitoring results according to the frequency. Fig. 4.6(a) shows all GT traffic flows meet the application's bandwidth requirements at minimum 40MHz. Also, I can know two

traffic flows (HD → SRAM and LCDC → SRAM) are the most performance-critical in this application because their bandwidth requirements are achieved at higher network frequency.

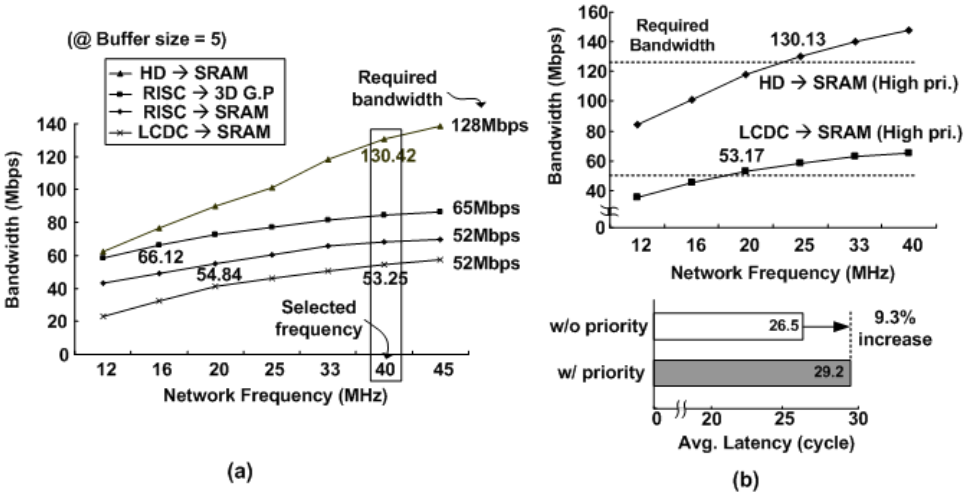

Fig. 4.6 Network Frequency Selection (a) w/o priority and (b) w/ priority

Each packet has a priority field in its header. When more than two packets are destined to the same output port in a switch, i.e. an output conflict occurs, a packet with higher priority gets a grant to the output port. Therefore, higher packet priority can be assigned to the performance-critical flow for guaranteeing the application performance requirements at lower frequency. In this application, I give a high priority to two critical traffics. Fig. 4.6(b) shows that the performance requirements of two traffic flows are met at 20MHz and 25MHz, respectively. The high priority flows does not affect other GT traffic flows because there are no shared links among all GT traffics (See Fig. 4.1). On the contrary, other BE traffic flows with lower priority (TG1 → TR1 and TG2 → SRAM) get longer latency. As a result, the overall average latency of all traffic flows increases by 9.3%.

From the results of the experiments, the minimum network frequency that meet

the application requirements varies according to packet priority assignment allowing the designer to make the performance-energy trade-offs.

4.3.3 Run-time Routing Path Modification

In NoCs, deterministic routing scheme such as source routing is widely used [2] because it is cost-effective scheme for network and transport layer design. In conventional source routing, a packet is transferred to a destination through a fixed routing path while the target application is running. In this work, a packet routing path is selected at run-time based on test packet latency. The routing path modification scheme is applied to a traffic flow (TG1 → TR1) in the target system. There are three candidate routing paths from TG1 to TR1 as shown in Fig. 4.7. Three test packets are generated periodically at source network interface and the latencies of test packets are measured at destination network interface. As a result, a routing path which has the smallest latency is selected at run-time and a routing table at source network interface is updated. Fig. 4.8 shows a hardware implementation for routing path modification. Candidate routing paths from a source node to a destination node and the period of test packet generation can be programmed at design time.

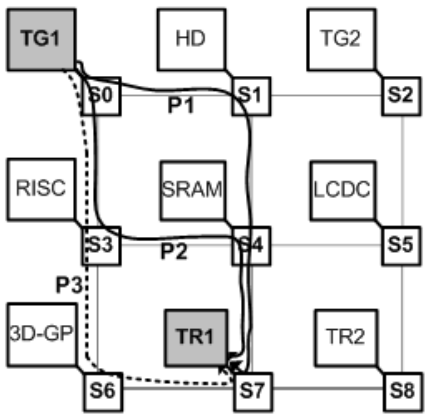

Fig. 4.7 Candidate routing paths from TG1 to TR1

Fig. 4.9 shows dynamic reconfiguration of a selected routing path at a period of 500 clock cycle. The probability of selecting routing path P1 increases abruptly according to time-varying traffic pattern (HD → SRAM). Fig 4.10 shows that 28% average latency reduction is obtained compared to a fixed source routing and its variation is also diminished significantly. As a result, the run-time routing path modification scheme reduces average latency and its variation of BE traffic affected by the GT traffic.

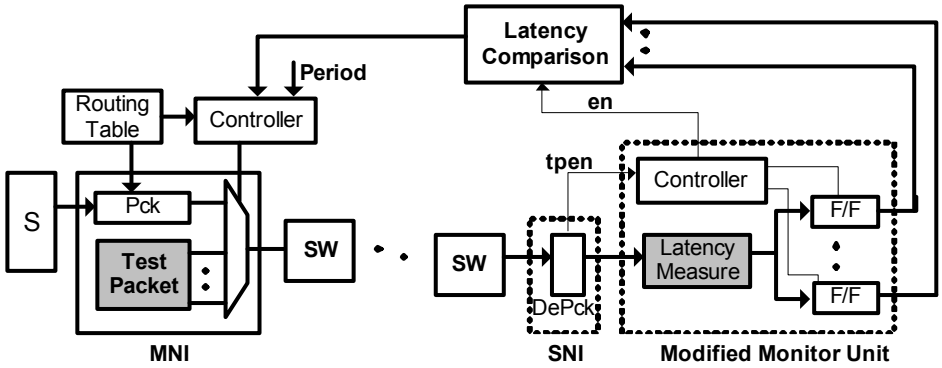

Fig. 4.8 Hardware implementation for routing path modification

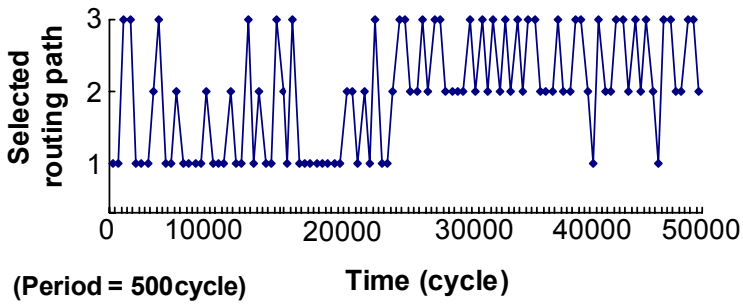

Fig. 4.9 Dynamic reconfiguration of a routing path

The experimental target system is somewhat straightforward and has only nine IPs. However, in more complex systems where it is very difficult to estimate the on-chip traffic patterns between IPs, the proposed monitoring system will be more effective to achieve highly-optimized application-specific NoC.

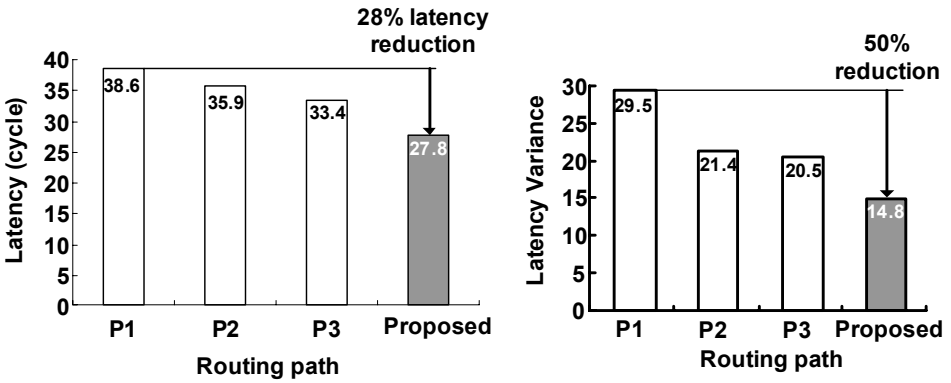

Fig. 4.10 Latency comparison from TG1 to TR1

4.4 Diagnosis and Refinement in Star topology

Recently, there was a report that star topology is more appropriate in on-chip situation if the number of PUs is limited to a few tens because there is no shared link in the star topology [2]. As a second case study, the same portable multimedia system is analyzed and refined in a star topology as shown in Fig. 4.11.

Fig. 4.12 shows the monitoring results of a traffic flow from the HD to the SRAM. The flow shows the largest average latency and largest backlog in the system. Compared to the mesh, 50% lower average latency and 60% lower latency-variance are observed in the star topology.

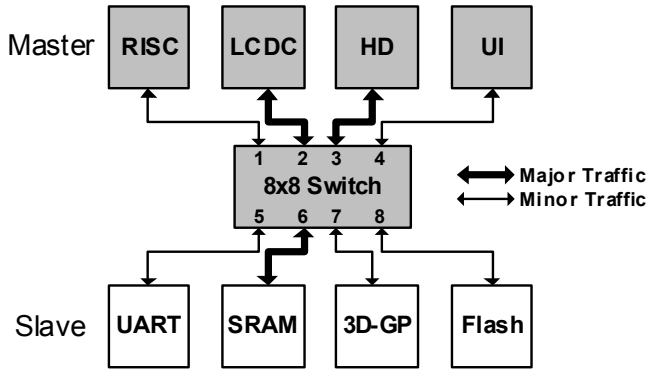

Fig. 4.11 Portable Multimedia System in Star topology

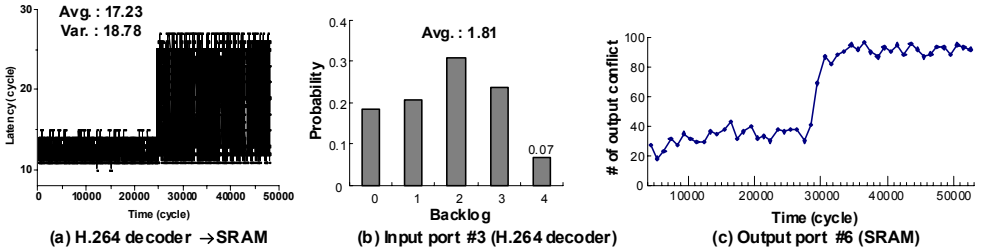

Fig. 4.12 Monitoring Results in Star topology

Buffer size assignment is carried out with the same policy described in chapter 4.3.1. As a result, total buffer size is reduced by 22% (from 32 to 25) while the system maintains the performance.

CHAPTER 5

FPGA Board Implementation

5.1 Overall System

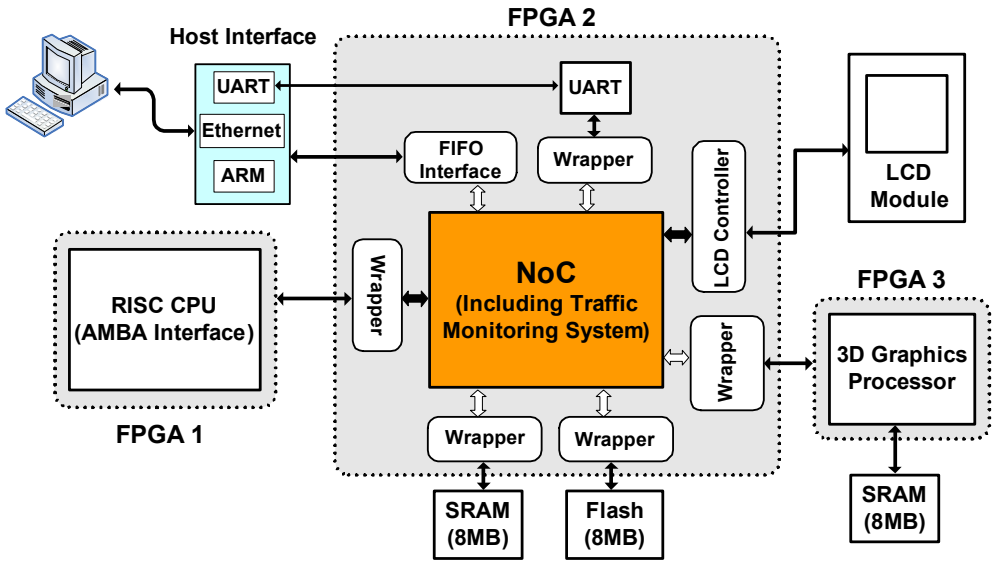

Fig. 5.1 Overall system block diagram

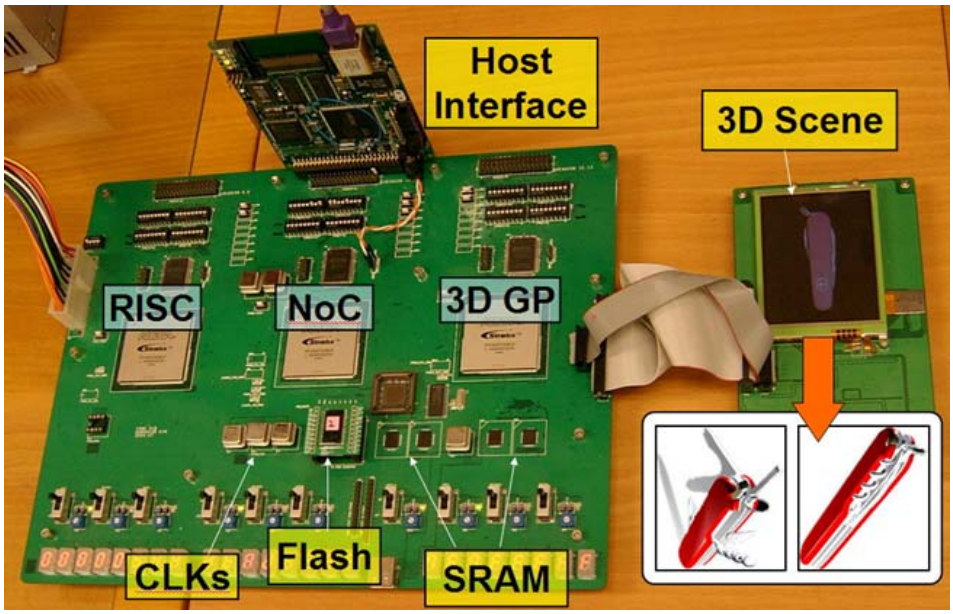

Fig. 5.2 Implemented NoC-based System on a FPGA board

Fig. 5.1 and Fig. 5.2 shows the overall system implemented on three Altera Stratix EP1S60 series FPGAs. The RISC and the 3D-GP are integrated on the left and right FPGA, respectively. The central FPGA integrates the total NoC with the proposed traffic monitoring system and other IPs such as the UART, the LCD controller, the H.264 traffic generator and the user interface logic. All implemented modules are designed in RTL level and 6 different clocks are used for the globally asynchronous locally synchronous (GALS) operation. Displayed 3D scenes on a LCD screen show that 3D graphics applications are successfully demonstrated on the NoC evaluation board, which shows feasibility of NoC in the implementation of a real system.

Table. 5.1 Specification of implemented system

Implemented IP	Features & Spec.
BONE	Maximum 51MHz with 3x3 mesh topology Maximum 46MHz with star topology (4x4 switch) Aggregated B/W : 1.36Gbps @ 20MHz
LCD Controller	Operate as DMA device 30fps @ 5MHz operation frequency
Memory Controller	Integrate 8MB SRAM @ 12MHz 8MB Flash memory @ 1MHz
Wrappers	Interface convert for RISC CPU (AMBA AHB), 3D GP, and UART (AMBA APB)

Table 5.1 shows the detailed specification of the implemented system. The network clock frequency is reported as maximum 58MHz on 4x2 mesh topology and 46MHz on star topology according to the FPGA compilation results. LCD controller operates 30frame/s at 5MHz. 8MB SRAM and 8MB flash memory operate at 12MHz and 1MHz, respectively. Various wrappers between IPs and network are designed for RISC CPU with AMBA AHB interface, 3D graphics processor, and UART with AMBA APB interface.

Table 5.2 shows the total logic element usage of the NoC core and monitoring modules. The proposed monitor system occupies a very little portion of the overall system, therefore, instantiation of a number of monitoring units for a larger NoC does not incur a problem.

Table. 5.2 Logic Elements Usage on FPGAs

Topology / Module	Mesh (LEs)	Star (LEs)
NoC Core (NoC)	45,506	20,089
Monitoring Modules (Mon)	2076	1126

Logic overhead (Mon/NoC)	4.6%	5.6%

CHAPTER 6

Conclusion

6.1 Conclusion

In this work, a NoC traffic monitoring system is proposed to probe the run-time internal traffic of the NoC cores for accurate performance evaluation and in-depth-refinement of application-specific NoCs. It provides dynamic network status such as a backlog, output conflict on a switch and an end-to-end communication latency for each packet flow. A portable multimedia system is implemented on FPGAs to demonstrate the effectiveness of the traffic monitoring system. Based on the monitoring diagnosis, the target system is refined in three ways; buffer size assignment, network frequency selection, and run-time routing path modification. As a result, buffering cost and latency reduction is obtained up to 42% and 28%, respectively in a mesh topology.

SUMMARY

특정 어플리케이션에 최적화된 네트워크 온 칩을 구현하기 위해서는 그 어플리케이션에 맞는 네트워크 온 칩 디자인 파라미터들을 결정하는 것이 중요하다. 이러한 파라미터들은 네트워크 온 칩 내부의 트래픽에 의해 많은 영향을 받기 때문에, 진정한 어플리케이션에 특화된 네트워크 온 칩을 위해서는 내부 트래픽을 관찰하는 것이 필수적이다.

본 연구에서는, 어플리케이션에 맞는 네트워크 온 칩의 정확한 평가와 개량을 위해 패킷 레이턴시나 버퍼 백로그와 같은 네트워크 온 칩 트래픽 파라미터들을 실시간으로 모니터링하는 트래픽 모니터링 시스템이 제안되었다. 이러한 트래픽 모니터링 시스템의 유용함을 증명하기 위해서 실제 네트워크 온 칩을 기반으로 하는 휴대용 멀티미디어 시스템이 3개의 FPGA를 이용해서 구현되었다. 트래픽 모니터링 시스템을 이용해서 타겟 시스템이 정확히 진단되고, 버퍼 사이즈 결정이나 네트워크 주파수 결정, 실시간으로 라우팅 경로를 바꾸는 것과 같은 디자인 개선 작업을 수행하였다. 그 결과 메시 형태로 구현된 시스템에서 42% 만큼 총 버퍼 사이즈가 감소하였고, 28% 만큼 레이턴시가 감소하였다.

REFERENCES

[1] L. Benini and G. De Micheli, "Networks on Chip : a new soc paradigm", *IEEE Computer*, vol. 35, pp. 70-78, 2002.

[2] S.Lee, et al., "An 800MHz Star-Connected On-Chip Network for Application to Systems on Chip", ISSCC Digest of Technical Papers, pp. 468-469, 2003.

[3] K.Lee, et al., "A 51mW 1.6GHz On-Chip Network for Low Power Heterogeneous SoC Platform", ISSCC Digest of Technical Papers, pp. 152-153, 2004.

[4] Tapani Ahonen, et al., "Topology optimization for application-specific networks-on-chip", International Workshop on System-Level Interconnect Prediction, pp. 53-60, 2004.

[5] Jingcao Hu and R. Marculescu, "Application-specific buffer space allocation for networks-on-chip router design", Proceedings of International Conference on Computer Aided Design, pp. 354-361, 2004.

[6] M. Coppola, et al., "Occn : a network-on-chip modeling and simulation framework", Proceedings of Design, Automation and Test in Europe, pp. 174-179, 2004.

[7] S. Pestana, et al., "Cost-performance trade-offs in networks on chip: a simulation-based approach", Proceedings of Design, Automation and Test in

Europe, pp. 764-769, 2004.

[8] N. Genko, et al., "A complete network-on-chip emulation framework", Proceedings of Design, Automation and Test in Europe, pp. 246-251, 2005.

[9] C. Ciordas, et al., "An event-based network-on-chip monitoring service", International High-Level Design Validation and Test Workshop, pp. 149-154, 2004.

[10] Ju-Ho Sohn, et al., "A 50 Mvertices/s graphics processor with fixed-point programmable vertex shader for mobile applications", ISSCC Digest of Technical Papers, pp. 192-193, 2005.

ACKNOWLEDGMENT

KAIST 학부과정 4년을 마치고, 반도체 시스템 연구실 (SSL) 에 들어온 지도 벌써 2년이 지났습니다. 많이 부족하고 이제부터 해야 할 일이 더 많겠지만 석사 과정 2년 동안의 노력의 결실로 한편의 논문을 완성하게 되어 뿌듯합니다. 이 논문을 위해 직간접적으로 도움을 주신 많은 분들께 감사의 글을 올리겠습니다.

우선 제 석사과정 연구를 함에 있어서 많은 가르침과 지도를 해주시어 논문이 잘 완성될 수 있도록 이끌어주신 유회준 교수님께 감사드립니다. 바쁜 와중에도 제 논문에 많은 조언을 해주시면서, 석사논문 심사를 맡아주신 신영수, 정송 교수님께도 감사드립니다.

2년 동안 랩에서 동고동락한 SSL 실험실원들에게도 감사의 말을 전합니다. 지금은 멀리 TI 로 가셨지만, 초기에 실험실에 대한 강한 인상을 남겨주신 람찬이형, OCN 팀 고문과 리더로써 저에게 많은 도움을 주신 세중이형, 강민이형, 이제 우리 실험실의 고참으로써 저에게 많은 충고와 조언을 해주신 병규형, 교민이형, 성대형, 주호형, 성준이형, 랩장으로서 책임감있게 랩을 이끌고 계시는 정호형, 1년선배로서 옆에서 많은 도움과 격려를 해주신 동현이형, 선영누나, 랩동기로써 2년동안 함께 동고동락하며 지낸 남준이형, 혜정이, 석1로서 궂은 일 도맡아 하면서 저에게 많은 도움을 준 담이형, 주영이, 이 모든 실험실원들에게 감사드립니다.

고등학교 때부터 지금까지 나와 가장 많은 시간을 같이하며, 서로 우정을 쌓아온 병호, 성화, 태영, 종준, 장훈, 종원, 창완, 상규 등에게도 감사의 말을 전

합니다.

마지막으로, 무뚝뚝한 동생을 위해 많은 것을 챙겨주었던 누나, 지금의 내가 있기까지 가장 헌신적으로 아끼고 도와주신 아버지, 어머니께 이 모든 결실을 바칩니다.

CURRICULUM VITAE

Name	Kim, Kwanho (김관호, 金官昊)
Date of Birth	18th August, 1982
Address	Dept. of E.E. KAIST, Guseong-dong, Yuseong-gu, Daejeon, 305-701, Korea (ROK)

Education

2004.3 - 2006.2	M.S. in Electrical Engineering, KAIST
2000.3 - 2004.2	B.S. in Electrical Engineering, KAIST
1998.3 - 2000.2	Hansung Science High School

Publication

[1] **Kwanho Kim**, Se-Joong Lee, Kangmin Lee, and Hoi-Jun Yoo, "An Arbitration Look-ahead Scheme for Reducing End-to-end Latency in Networks-on-Chip", in *IEEE Proc. Int. Symp. Circuits and Systems*, May 2005.