# SONA: An On-Chip Network for Scalable Interconnection of AMBA-Based IPs*

Eui Bong Jung[1], Han Wook Cho[1], Neungsoo Park[2], and Yong Ho Song[1]

[1] College of Information and Communications, Hanyang University, Seoul, Korea
{ebjung, hwcho, yhsong}@enc.hanyang.ac.kr
[2] Dept. of Computer Science and Engineering, Konkuk University, Seoul, Korea
neungsoo@konkuk.ac.kr

**Abstract.** Many recent SoCs use one or more busses to provide internal communication paths among integrated IP cores. As the number of cores in a SoC increases, however, the non-scalable communication bandwidth of bus tends to become a bottleneck to achieve high performance. In this paper, we present a scalable switch-based on-chip network, called SONA, which can be used to provide communication paths among existing AMBA-based IP cores. The network interfaces and routers for the on-chip network are modeled in register transfer level and simulated to measure the performance in latency. The simulation results indicate that the proposed on-chip network can be used to provide scalable communication infrastructure for AMBA-based IP cores with a reasonable cost.

## 1 Introduction

The recent improvement in semiconductor technology enables various modules with different functionality and complexity to be integrated to a single system-on-chip (SoC). A SoC often consisting of one or more processors, DSP cores, memories, I/Os and internal communication channels is used to build an embedded system such as mobile handsets, PDAs, etc. Considering that embedded systems often use battery as a power source, it is required that SoCs consume less energy for normal operation modes and nevertheless produce reasonable performance.

Traditionally one or more busses are used inside SoCs to implement communication channels among integrated components. It is because that the simple structure of this interconnection contributes to the reduction of design cost and effort. However, as a bus-based SoC integrates more and more components, the insufficiency in communication bandwidth often results in the degradation of the entire system. This problem becomes even worse when deep sub-micron technology is used for the implementation of SoCs. As the technology evolves, the relative length of global wires increases, which may make data transactions take more clock cycles and thus increase communication cost.

There have been many approaches to overcome the limitation in scalability of bus systems. One of them is to use a switch- (or router-) based network within a SoC,

* This work has been supported by a grant from Seoul R&BD Program.

V.N. Alexandrov et al. (Eds.): ICCS 2006, Part IV, LNCS 3994, pp. 244 – 251, 2006.

called *on-chip network* or *network-on-chip (NoC).* This network, an on-chip variation of high-performance system networks, effectively increases communication bandwidth and degree of operation concurrency. The communication links used in constituting on-chip networks are relatively short in length and arranged in a regular fashion, they often consume less energy for data transaction and overcome many electrical problems arisen from the use of deep sub-micron technologies. The provision of on-chip networks for SoCs effectively decouples computation from communication by the introduction of well-structured communication interfaces, which is becoming more important as the SoC density increases.

However, the change in communication layer inevitably necessitates the modification of communication interface of many existing IP (Intellectual Property) cores. In fact, the success of AMBA AHB [1] makes IP vendors develop hardware or software cores that can run in conjunction with the AMBA protocol. Considering that the reuse of IP cores plays a crucial role in reducing design cost and effort as well as preventing from taking unnecessary risk from making a new design, it is desirable to reuse the bus-based IPs in the implementation of a SoC based on a switch-based network.

This paper proposes an on-chip network, called SONA (Scalable On-chip Network for AMBA), as a scalable communication backbone which efficiently interconnects AMBA-based IPs. The network implements 2D mesh topology with a bidirectional link between a pair of switches. A network interface connects an AMBA IP to a SONA switch and converts communication protocols across the two different protocol domains. The network is modeled at register transfer level and simulated on MaxSim, a hardware/software co-simulator from ARM [2], to measure the communication performance.

## 2 Related Work

A variety of busses are used in SoCs to provide communication channels for IP cores within a chip. The example includes AMBA AHB from ARM [1], CoreConnect from IBM [5], MicroNetwork from Sonics [6], and Wishbone from Silicore [7]. This type of communication system provides many features for developers: simple to design and easy to develop software. However, as the amount of data to travel over bus systems increases, the insufficient bandwidth of the communication media inevitably results in long communication delay, which limits the use of bus systems only to small systems.

Switch-based networks have been long used as a communication infrastructure in the field of computer networks and parallel system networks. Such networks are brought on chip to solve the problem of insufficient communication bandwidth provided by traditional on-chip buses. Even though on-chip networks successfully inherit many useful features and techniques needed to boost communication performance, they still need to have solutions to many other constraints: buffer memories are expensive to implement, silicon budget is tight, and energy consumption needs to be kept low.

Recently there have been several attempts to design AMBA-compatible on-chip networks. However, the networks proposed in [8][9] employs a set of wide crossbar switches to simply forward AMBA signals between end nodes without the notion of

packetization, resulting in limited scalability and capability far less than those needed to implement high integration in future SoCs.

## 3 Network Architecture

The network performance is often measured in terms of latency (i.e. the time delay for delivering information from source to destination) and throughput (i.e. the amount of information delivered in a given time window). The goals in designing SONA are to achieve high level of scalability as well as to provide high performance. In addition, less energy consumption and reduced cost in protocol conversion are also pursued in this network architecture.

### 3.1 Packet Structure

In AMBA, a master drives necessary bus signals when it needs to start a new transaction. In order to deliver the bus signals over a switch-based on-chip network, a network interface at the master side packages the semantics of bus signals into a packet, and another at the slave side restores the signals from the packet.

Figure 1(a) illustrates the packet structure used in SONA. Each packet consists of a header and optionally a payload. A variable-sized packet is sliced into multiple flits each being 32 bits long. The header contains a target node number, a source node number, the packet size in flit, an associated command (or transaction type), and a 32-bit address. Depending upon its transaction type, the payload may grow up to 512 flits which is the case that the burst type (HBURST) is 16 and the size (HSIZE) is 32 words.

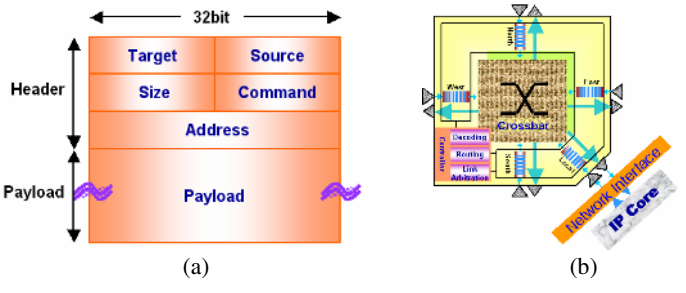


**Fig. 1.** (a) The packet structure and (b) the router architecture used in SONA

The target and source node number is encoded in 16 bits, which allows up to 65536 nodes to be integrated in SONA. Because the maximum length of payload is 512 words, at least 9 bits are needed to record the payload size in the header. For this reason, 16 bits are reserved for packet size in the header. The command field is used to encapsulate the AMBA control signals: HTRANS (2 bits), HBURST (3 bits), HSIZE (3 bits), and HWRITE (1 bit). The address field is used to deliver 32-bit HADDR signals and the payload field is used for transmitting 32-bit HWDATA or HRDATA.

A tail flit usually carries checksum information to verify the integrity of the packet at the receiver's side. In SONA, however, no tail flits are used in packets assuming

that no bit errors would occur during the transmission over communication links due to the short wire length. In order to detect the end of variable-sized packet, the receiver decodes the packet size from the header and counts the number of payload flits up to this size.

### 3.2 Router Architecture

Figure 1(b) shows a SONA router consisting of a central crossbar, four ports each providing a communication channel to its neighboring router, a local port through which an IP core accesses the network, and the control logic for implementing flow control and routing mechanisms. When a packet arrives at a port, it is buffered in an input queue awaiting a routing decision by the controller. Each queue in a port can hold up to 8 flits and packets are delivered in wormhole switching. The four inter-router ports are used to build a network with two-dimensional mesh topology. Each link is 32-bit wide so that a flit can move across the link in a clock cycle.

For simplicity, the on/off mechanism [10] is used for flow control over links. In order to implement this mechanism, each communication link provides a pair of control signals each for one direction. Flits can be sent over link only when the signal is set to ON. No virtual channels are implemented over each physical link. Routing deadlocks inside the network are avoid by the use of dimension-order routing [11].

### 3.3 Network Interface Architecture

The network interface bridging an AMBA core to the SONA network performs protocol conversion from AMBA AHB to SONA and vice versa. Depending upon the role in transaction, the AMBA core is connected to one of two network interfaces, master network interface (MNI) and slave network interface (SNI). For example, a RISC core and a memory module are connected to MNI and SNI, respectively.

Figure 2 shows the two SONA network interfaces, communicating over the network. Packets are delivered on 32 bit `flit_i`/`flit_o` channels. The presence of valid flits on these channels is indicated by accompanying `tx_o`/`rx_i` signals. Likewise, `on_off_i`/`on_off_o` signals are used for flow control.

Each network interface has two state machines, one for packet transmission (named MNI_Request and SNI_Resend) and the other for packet reception (named MNI_Response and SNI_Receive).

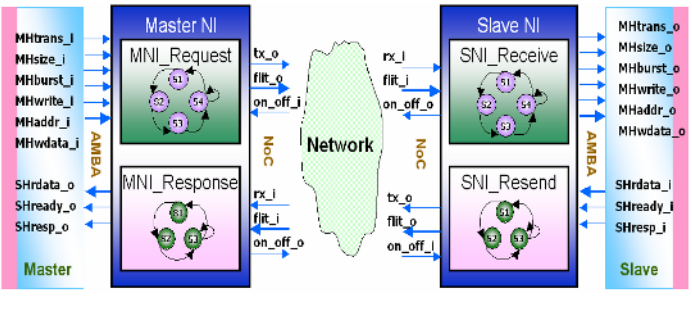


**Fig. 2.** Master and slave network interface for a master and a slave AMBA cores, respectively

The write operation by an AMBA bus protocol consists of three phases: arbitration phase, address phase, and data phase. During the arbitration phase, the network interface checks the buffer status of its neighboring router and reports it to the transaction-initiating local core by driving the SHREADY signal. Therefore, if there are no buffers available, the local core retries later on the reception of this signal. In the address phase, the state machine for packet transmission temporarily holds MHTRANS, MHSIZE, MHBURST, and MHWRITE into a packetization buffer and encodes them into the packet header. Likewise, when a network interface receives a packet from the network, the state machine for packet reception temporarily stores the packet into a de-packetization buffer until all the necessary signals are restored from the buffer.

In MNI, the packetization buffer is large enough to hold the AMBA signals from the local core necessary for generating a packet header. Note that the AMBA data is not stored into the buffer even when a burst write operation takes place. Instead, the MNI transmits the header flits from the buffer and payload flits from the HWDATA signal on the fly. The buffer in router becomes overflowed by the injection of long payload. There are two approaches to deal with this problem. The first is to increase the size of packetization buffer up to 512 flits to temporarily hold both the header and the data and retry the transmission later. It is not a good solution considering that memory resource within a chip is expensive. The second is to drive the SHREADY signal to low to indicate that the slave is not ready. In this case, the local core stops the generation of signals and retries the transaction later.

MNI uses a node number to identify packet destinations. It generates a destination node number by looking up a memory map which associates a node number with a memory address range. The AMBA address, MHADDR, is used for the map lookup.

SNI is responsible for delivering a request arriving from the network to the local core. It runs a de-packetization process to restore AMBA signals to the local core. For reads, it decodes a memory address from the header along with other AMBA signals such as MHTRANS, MHBURST, MHSIZE, and MHWRITE. Optionally, it generates a sequence of memory addresses needed to complete a burst type operation. When receiving a response from the core, SNI first checks if the SHREADY signal generated by the local core is set to AHB_READY. If this is the case, the data from the local memory is stored into the packetization buffer. For writes, SNI decode the payload into the MHWDATA signal. Optionally, for each word in a burst write, SNI can generate an address to drive the MHADDR signal along with MHWDATA.

## 4 Simulation

We have modeled a SONA network with 2x2 mesh topology network at synthesizable register transfer level using SystemC. The network model is used to build a virtual platform by adding transaction-level models for local cores including an ARM9 processor, a memory, an MPEG4 encoder and other cores necessary for mobile multimedia applications. The virtual platform is used to develop system/application software prior to building a hardware prototype. The MaxSim simulation tool from ARM is used for simulation with the traffic generated by the ARM9 transaction-level model running an application of an MPEG encoder for the 720x480 YCbCr 420 pixel format.

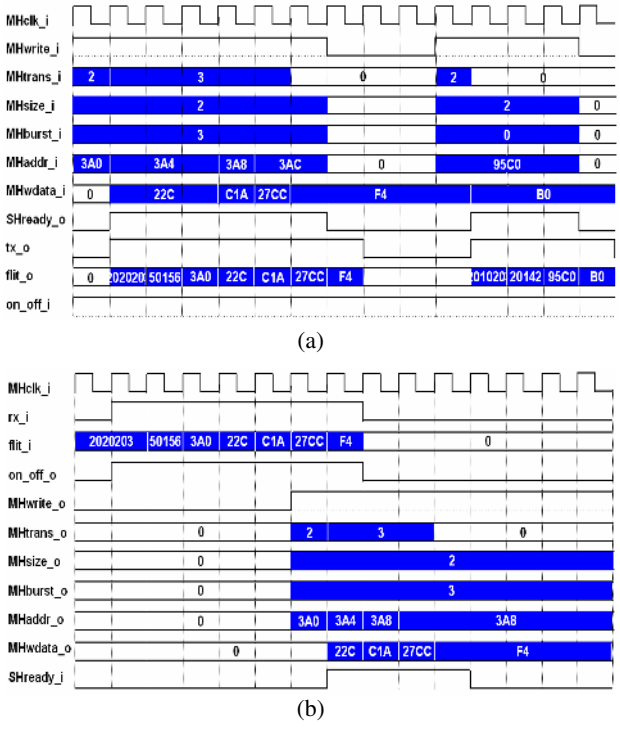


**Fig. 3.** The waveform indicating the operating of (a) master network interface and (b) slave network interface for a write operation

In order to evaluate the performance of SONA, we have measured only the latency for read and write operations because it is well studied that switch-based networks provide higher throughput than busses do. Figure 3(a) illustrates a burst write operation with 4 words. The first address generated at cycle 1, 3A0, is injected on `flit_o` at cycle 4, which indicates that it takes three clock cycles to inject a packet to its neighboring router at MNI upon the write request from a master core. For a burst operation, it takes an extra clock cycle for each payload flit.

Figure 3(b) shows the latency when a packet is delivered to SNI and further to the local core. The processing of a header takes three clock cycles and the payload delivery takes the same amount of delay in SNI.

Figure 4 illustrates the waveform for the signals of MNI for a read operation. It takes three clocks for MNI to inject a read request into the network as in MNI for a write (see Figure 4(a)). The transmission latency between MNI and SNI depends on the number of router hops that the packet traverses. As shown in Figure 4(b), it takes 6 clock cycles for SNI to de-packetize and deliver the request to the slave core and another 5 clock cycles to receive data from the core.

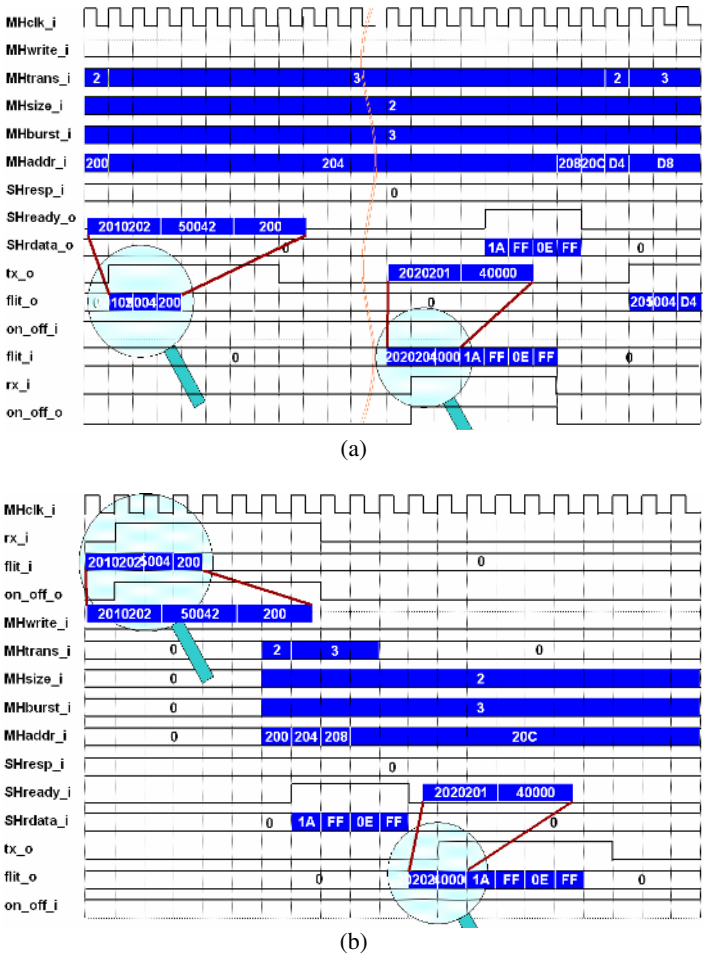


**Fig. 4.** The waveform indicating the operating of (a) master network interface and (b) slave network interface for a read operation

## 5 Conclusion

In this work, we present a scalable on-chip network for interconnecting AMBA-based IP cores. The network is modeled in SystemC to build a virtual platform for the development of software and to explore architectural space to enhance the performance. It is simulated along with other IP cores which are used to build a recent multimedia mobile SoC to observe the possibility of replacing a bandwidth-limiting on-chip AMBA with a scalable switch-based network.

Even though the use of switch-based networks brings about the increase in latency to complete a transaction, it enables IP cores to utilize increased bandwidth, making them experience less latency under high network loads. Considering that recent

mobile applications requires increasing bandwidth to provide high quality multimedia service, the poor scalability of on-chip bus may become a bottleneck for achieving high performance. The increased latency can be compensated by placing the IP cores closer which make high volume of traffic. Or the network can be used to bridge multiple AMBA buses in a scalable fashion.

## References

1. AMBA Bus Specification, http://www.arm.com
2. http://www.arm.com/products/DevTools/MaxSim.html
3. http://www.synopsys.com/products/logic/design_compiler.html
4. International Technology Roadmap for Semiconductors, http://public.itrs.net
5. CoreConnect Bus Architecture, http://www-03.ibm.com/chips/products/coreconnect/index.html
6. Sonics Integration Architecture, Sonics Inc., http://www.sonicsinc.com
7. W. Peterson, WISHBONE SoC Architecture Specification, Rev. B.3, Silicore Corp, 2002.
8. J. Lee, et al., SNP: A New Communication Protocol for SoC, International Conference on Communications, Circuits and Systems, Cheungdu, China, June 2004.
9. J. Jang, et al., Design of Switch Wrapper for SNA On-Chip Network, Joint Conference on Communications and Information, 2005.
10. William James Dally, Brian Towles, Principles and Practices of Interconnection Networks, Morgan Kaufmann Publishers, 2003
11. W. Dally and C. Seitz. Deadlock-free Message Routing in Multiprocessor Interconnection Networks, IEEE Transactions on Computers, 36(5):547–553, May 1987.
12. Magma Design Automation, A Complete Design Solution for Structured ASICs, white paper, http://www.magma-da.com
13. J. Liang, S. Swaminathan, R. Tessier, aSOC: A Scalable, Single-Chip Communications Architecture, Conference on Parallel Architectures and Compilation Techniques, 2000.
14. A. Radulescu, et al., An efficient on-chip network interface offering guaranteed services, shared-memory abstraction, and flexible network programming, IEEE Transactions on computer-aided design of integrated circuits and systems, January 2005.