Abstract

The future of on-chip communication is Networks-on-Chip (NoC), meaning that on-chip communication is done through packet based networks. Today System-on-Chips (SoC) uses shared buses as on-chip interconnection architecture. If old IP cores, that were designed to use shared buses as communication architectures, could be used on a NoC system without the need of changes, it would increase the speed of system development, since testing of the old IP cores is already done.

The purpose of this report is to just describe a NoC architecture using different topologies. Different communication issues in NoC and Nostrum concept for NoC .What are the different protocol layer used in Nostrum.

Contents

Chapter 1
Network-on-Chip

1.1 Introduction

Network-on-a-Chip (NoC) is a new approach to System-on-a-Chip (SoC) design. NoC-based systems can accommodate multiple asynchronous clocking that many of today's complex SoC designs use. The NoC solution brings a networking method to on-chip communication and brings notable improvements over conventional bus systems.

Integrated circuit technology develops fast according to Moore's law and will do so for almost another decade according to ITRS road map. In a few years from now this will lead to the possibility of having billions of transistors on one single chip. Using current methodologies as shared buses to design communication platforms for systems of that magnitude will introduce a number of issues to solve, both logical and physical. Some of the important ones are the utilization of on-chip communication infrastructures, the increase of wire delay, the controlling of power distribution and power dissipation and also the ability to guarantee signal integrity. To solve these issues there exist a proposed solution. It's called Network-on-Chip (NoC), which suggests that future on-chip communication should work as a network that is similar to today's computer network.

Another issue in future on-chip design is the level of reusability of Intellectual Property cores (IP cores). A poor reuse of IP cores leads to long development times and makes it impossible for the system designers to meet the market requirements of short time-to- market. At KTH there is a suggestion under development for how to design a platform for on-chip communication on a NoC. It is called the Nostrum concept. Nostrum suggests that a NoC should be arranged as a mesh of switches, where every switch is connected to one resource and four neighbouring switches. Nostrum also declares that the network protocols should work in different layers with different responsibilities, in a similar way as the OSI reference model.

NoC is a scalable architectural platform with a potential to handle growing complexity and can provide easy re-configurability. The basic idea of this way to design is future on-chip systems are proposed to use that processors are connected via a packet switched communication network on a single chip similar to the way computers are connected to internet. NoC is very useful when scalable systems with heterogeneous set of processors should be designed. NoCs modular organization with standardized interfaces

makes this architecture very suitable for designing products with already designed and verified cores. This architecture also provides a possibility to build powerful systems with several simple and slow computers , some of them highly specialized, on a single chip.

1.2 Reasons for using NoC

There are both physical and logical issues that prevent shared buses from being the on-chip interconnection solution of the future. Amongst the physical issues we find the increase of wire delay, the control of power distribution and cross- talk between wires. A logical issue is that if more subsystems use a shared bus the usage time of the bus for each subsystem will be less. That generates a low utilization level of IP cores that have to use the bus often.

Therefore it has been proposed that NoCs should be used as on-chip communication solutions instead of shared buses. Some scientists have proposed that hybrid interconnection solutions could also be used. A hybrid solution means that the interconnection solutions would be a mixture of NoC and shared buses. The main idea with NoCs, besides the solutions to the physical issues, is the possibility for more cores to communicate simultaneously, leading to larger on-chip bandwidths.

The adoption of NoC architecture is driven by several forces: from a physical design viewpoint, in nanometre CMOS technology interconnects dominate both performance and dynamic power dissipation, as signal propagation in wires across the chip requires multiple clock cycles. NoC links can reduce the complexity of designing wires for predictable speed, power, noise, reliability, etc., this is due to their regular, well controlled structure. From a system design viewpoint, with the advent of multi-core processor systems, a network is a natural architectural choice. An NoC can provide separation between computation and communication, support modularity and IP reuse via standard interfaces, handle synchronization issues, serve as a platform for system test, and, hence, increase engineering productivity.

1.3 Different NoC Topologies

There have been proposed a number of different NoC topologies. They all have in common that they connect resources to each other through networks and that information is sent as packets over the networks.

Network on Chip (NoC) has evolved as an important research topic during the last few years. The idea is that scalable switched networks are used for on-chip communication among processing units, in order to cope with design of continuously growing systems. The

main force behind NoC is the progress in semiconductor manufacturing technology and the fact that it is not followed by increased effectiveness in design automation tools. Design complexity promotes reuse of investment in earlier designs as well as purchase of outside intellectual property (IP), but due to larger designs, communication among components will become a bottleneck using traditional techniques like e.g. common buses. NoC is one solution to this where packet switched communication can provide higher flexibility, throughput and reusability.

To gain full advantage when using this concept in NoC architecture design, the size of resources should be similar and the communication facilities should be homogeneous. This can be in conflict with design goals like performance, power consumption or cost. In order to solve this it may be necessary to specialize parts of the network to create a feasible design. Tasks in specialization can involve selecting appropriate set of resources, like processors, dedicated hardware and programmable hardware. Another option for specialization is to modify the communication architecture itself for the requirements of certain applications.

1.3.1 Honey Comb Topology

The interconnect scheme is in many respects at the heart of the NOC architecture. Each resource, whether it is computational, storage or I/O, will have an address and will be interconnected by a network of switches. These resources, will communicate with each other by sending addressed packets of data, routed to their destination by the network of switches.

Though many topologies are possible, we present here one possibility for further discussion and analysis. The overall organisation is in the form of a honeycomb, as shown in Figure 1.The resources - computational, storage and I/O are organised as nodes of the hexagon with a local switch at the centre that interconnects these resources. Hexagons at the periphery would be primarily for I/O, whereas the ones in the core would have storage and computational resources.

Storage resources can be combined to create a larger virtual storage. This would be transparent to the applications; the refinement tool would be responsible for allocating and combining memories to create larger ones if necessary. Each resource, located on a hexagonal node being connected to three switches, can reach 12 resources with a single hop. To further improve the connectivity, switches are directly connected to their next nearest neighbours, as shown in Figure 1.1, allowing any resource to reach 27 additional

resource with two hops. As a last measure to further improve connectivity, every alternate switch is directly connected making each resource element reach a lot more elements with minimal number of hops.

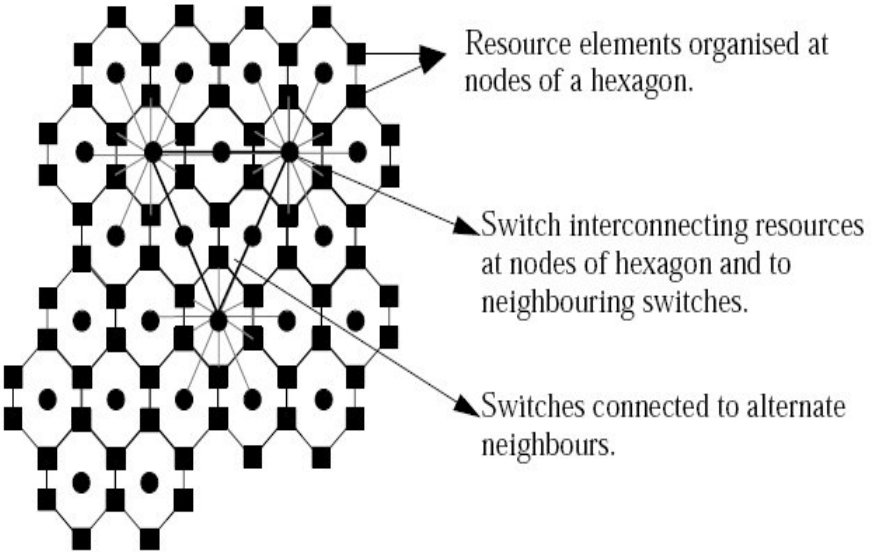

Figure 1.1 A Honey Comb Structure for NoC

1.3.2 Mesh Topology

NOC is a scalable packet switched communication platform for single chip systems. The NOC architecture consists of a mesh of switches together with some resources which are placed on slots formed by the switches. Figure 1.2 shows NoC architecture with 16 resources. Each switch is connected to four neighbouring switches and one resource. Re- sources are heterogeneous. A resource can be a processor core, a memory block, an FPGA, a custom hardware block or any other intellectual property (IP) block, which fits into the available slot and complies with the interface with the NOC switch. We assume switches in NOC have buffers to manage data traffic. Figure 1.3 shows the internal organization of a typical switch.

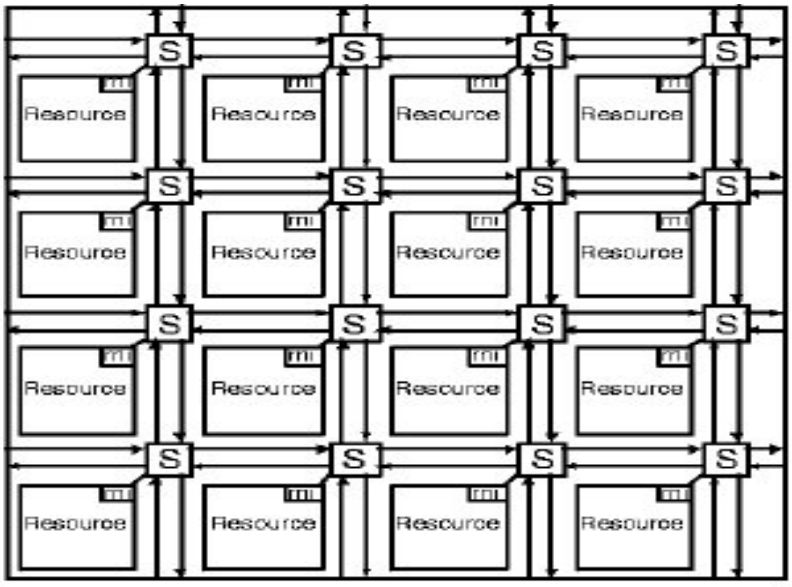

Figure 1.2 A Mesh Topology for NoC

NOC is a scalable packet switched communication platform for single chip systems. The NOC architecture consists of a mesh of switches together with some resources which are placed on slots formed by the switches. Figure 1.2 shows a NOC architecture with 16 resources. Each switch is connected to four neighbouring switches and one resource. Re- sources are heterogeneous. A resource can be a processor core, a memory block, an FPGA, a custom hardware block or any other intellectual property (IP) block, which fits into the available slot and complies with the interface with the NOC switch. We assume switches in NOC have buffers to manage data traffic. Figure 1.3 shows the internal organization of a typical switch.

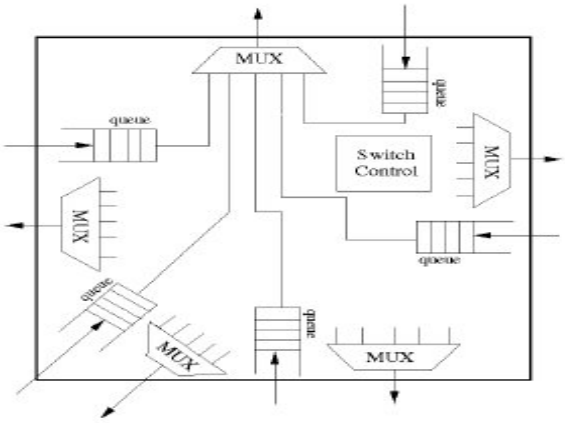

Figure 1.3 A typical NoC switch

1.3.3 Binary tree Topology

A novel interconnect template to integrate numerous IPs of an on-chip network based on binary tree architecture is shown in Figure 1.4. Because of the network topology impacts the complexity of the routing decisions, we choose a simple binary tree interconnection topology.

It has an area efficient layout on a two-dimensional surface. Furthermore, routing in a two-dimensional binary tree is easy resulting in potentially smaller switches, higher capacity, shorter clock cycle, and overall scalability. In our network, the IPs are placed at the leaves and switches placed at the node.

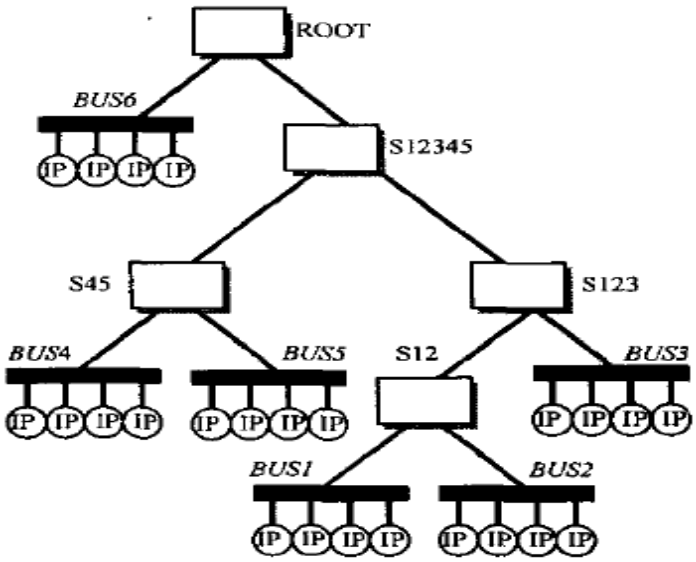

Figure1.4 A Binary tree Topology for NoC

In a bus system, users can group any number of components in a bus. All communications between cores are exclusive in time. Access requests are issued by using the bus arbiter to initiate a bus transaction. The arbiter decides priorities when there are conflicting requests. To transfer information from one side of the network to the other side, different buses are connected by switch.

1.4 Network Communication Issue

Changing from a shared bus communication platform that uses centralized bus arbitration to a network based on-chip interconnection platform leads to many new communication issues. Among these are: which switching technique should be chosen, how to guarantee Quality-of-Services? QoS? and how should the distributed arbitration work.

1.4.1 Communication Techniques

Circuit switched

A circuit switched network has dedicated paths between transmitters and receivers. When such path has been set up only the transmitter and receiver can send information over that path. A good example of a circuit switched communication network is a crossbar switched bus. In a crossbar switched bus dedicated circuits between different transmitters and receivers are set up before communication takes place. When the circuits have been set up, the information is sent as a stream through the circuits to the receivers. Afterwards all circuits are torn down and new ones can be set up.

Packet switched

Packet switched communication has no dedicated circuit for transmission. Instead it can be compared to ordinary mail delivery. You place your message into an envelope of proper size and attach a destination address and sometimes also a source address to it. Then you place it into a mailbox and it is routed between different post terminals before it ends up at the local post office, where it is brought out to the destination address. This way many different transmitters can send messages to the same receiver at the same time, but the receiver can only handle one message at a time when they arrive.

Cell switched

Cell switching is a form of packet switching with the difference that a fixed format for packets is defined. That means, if compared to the packet switching technique described above, that all you could send by mail would be for example postcards and nothing else. If your message can't t into a postcard you would have to fragment it into smaller parts and write the message down on several postcards. The receiver then has to sort the incoming postcards into the right order, since they might arrive out of order, before it can be able to retrieve the original message.

Message switched

Message switching is sometimes called store-and-forward switching because that's what it does. It works in the following way, a message is sent out to the network where the complete message is stored at the next switch, until a new routing decision is made. The message continues to jump to the next switch in the network until it reaches its destination.

This technique requires that all switches are equipped with buffers of sufficient size to be able to store complete messages.

Connection-oriented communication

In connection-oriented communication a path between transmitter and receiver has to be set up prior to transmission. For example circuit switching is always connection-oriented, because the communication path always has to be set up before any information exchange. Packet switched networks can also be connection-oriented, especially when large amounts of data is to be sent. In this case there will be no dedicated path for communication. Instead a virtual circuit (VC) is set up before transmission. The VC guarantees that a given bandwidth between the transmitter and receiver can be used.

Connection-less communication

Connection-less communication is on the other hand just what it sounds like, it's connection-less. That means that no establishment of a path has to be done prior to communication. A transmitter can send away its message and it will end its way to the receiver. Packet switched communication is often connectionless when small amounts of data is being sent, because the cost of setting up a path would be high compared to the amount of data. A virtual circuit (VC) is a path between two no des in a network. The path seems to be dedicated for the user of it but is often shared with other network traffic. A VC guarantees a certain bandwidth, which will be some fraction of the total bandwidth of the network path, for the user. The guaranteed bandwidth is negotiated during the set up of the VC.

1.4.2 Blocking and non-blocking communication

When processes communicate they can be blocking or non-blocking. For example if a blocking process performs a send or receive operation it will be blocked until an acknowledged is received or some other condition is true. If a non-blocking process performs a send or receive operation it can continue to run even if there hasn’t arrived an acknowledgement. The non-blocking process should however have a possibility to check if the transmission was successful or not later on.

1.4.3 Reliable and non-reliable communication

When two processes communicate with each other the transfer of information can be reliable or unreliable. A reliable data transfer means that the process gets an acknowledgement when the transfer is complete, so that it knows if the transfer was successful or not. In an unreliable data transfer the transmitting process just sends away the information with no control of the success or failure of the transmission.

In layered communication acknowledgement can be implemented in one or more layers. This leads to many different combinations of reliable and unreliable communication protocols. For example if acknowledgement is only implemented at the application layer this would lead to that a error would be detected very late in the communication hierarchy and lead to slow communication when errors or timeouts occur. If acknowledgement would be implemented at a lower level a retransmission decision could be made earlier.

Of course a designer has to consider the payoff between in which layers acknowledgement and retransmission should be implemented and the gain in system performance. The more acknowledgement implementations that exist the faster a retransmission can be made if an error occurs, the cost in that case will be more area needed in each layer where it is implemented. So a trade-off between area and system performance has to be made.

Chapter 2

Nostrum Concept

2.1 Introduction to Nostrum Concept

The Nostrum concept is a platform for communication between processes on NoCs. Nostrum is made up of a backbone and a NoC architecture. Its main purpose is to provide a stable and reliable platform for communication on NoCs. The backbone uses a general cell switched network which can be used by many different SoC designs.

2.2 Nostrum Terminologies

Some of the important definitions in Nostrum are explained in this section.

Resource

A resource is a unit connected to the on-chip network. It is made up of one or more processes.

Process

Nostrum defines that a process is a functional unit placed on a resource. For example it may be a memory, a processor or reconfigurable hardware. Furthermore one or more processes can be placed on one resource.

If processes on the same resource communicate it is called internal resource communication and if processes on different resources communicate it is called inter-resource communication.

Network Interface (NI)

Between every resource and network switch there exist a Network Interface (NI).The reason to have a NI is to get a homogeneous interface between resources and the network. The NI handles address decoding and buffering of incoming and outgoing packets.

Resource Network Interface (RNI)

A Resource Network Interface (RNI) is the custom hardware or software used to connect the Nostrum protocols with the protocols used by a resource. A RNI can thereby act as a DMA or bridge, depending on what IP cores that are placed on a resource and what communication interfaces they use. This means that a RNI handles resource specific functions as translation, packet ordering, message fragmentation, message concatenation and arbitration.

Process ID (PID)

Each process has its own identification number called Process identifier (PID).

Message Sequence Number (MSN)

Every message sent on the NoC has a message number (MSN).

Packet Sequence Number (PSN)

If a message doesn't t into one single packet it has to be fragmented into a series of packets. Each packet in a series of packets with the same MSN, receives a number called the Packet Sequence Number (PSN). The PSN tells the destination RNI in which order it must sort the incoming packets to retrieve the original message.

2.3 Nostrum protocol layers

Nostrum uses a lot of ideas from computer network theory, one is that Nostrum divides the transfer protocols into layers in the same way as the Open System Interconnection (OSI) model. The OSI model consists of seven different layers which are from the bottom to the top: physical, data link, network, transport, session, presentation and application layer. Each layer has different responsibilities in the communication process. The layers that are used in this NoC are presented later in this Report.

Nostrum doesn’t use the OSI model exactly as it’s used in ordinary computer networking. Instead it’s used more as an aid for the designer, since there are many differences in de- signing an on-chip network and an ordinary computer network. That’s also why Nostrum isn’t strict about keeping the layers separated in hardware. For example if a fusion of two or more layers in hardware results in an optimization of the design, so can be done.

Only the three lowest layers are compulsory in Nostrum, these are physical, data link and network layer. What these three layers do together, if seen as a black-box from the transport layers perspective, is simply to deliver packets from the node where they enter the network to the node pointed out by the destination address.

Nostrum suggests the following responsibilities for the four lowest protocol layers:

Physical Layer: This layer is concerned with physical characteristics of the physical medium used for connecting switches and resources with each other. In the context of SoC, it specifies voltage levels, length and width of wires, signal timings, number of wires connecting two units etc.

Data Link Layer: This layer has the responsibility of reliably transferring information across the physical link. The layer provides functionality of transferring one word of information from one node to a connected node without error. Since the two connected units may be working asynchronously, the data link layer takes care of hardware synchronization besides error detection and correction. Data link layer may also include data encoding or data rate management for controlling power consumption etc.

Network Layer: The network layer provides the service of communicating a packet from one resource to another using the network of switches. Its functionality includes buffering of packets and taking routing decisions in the intermediate switches.

Transport Layer: This layer has the responsibility of establishing end-to-end connection and delivery of messages using the lower layers. Therefore its functionality includes packetization of a message at the source and depacketization or assembly of packets into a message at the destination node.

Application Layer: For on chip communication, the relevant functionality of upper three layers of OSI reference model can be merged into this layer. Important services in this layer include message synchronization and management, conversion of data formats at receiver and application dependent functionality.

2.4 Benefits of Adopting NoCs

The adoption of NoC architecture is driven by several forces: from a physical design viewpoint, in nanometre CMOS technology, interconnects dominate both performance and dynamic power dissipation, as signal propagation in wires across the chip requires multiple clock cycles. NoC links can reduce the complexity of designing wires for predictable speed,

power, noise, reliability, etc., thanks to their regular, well controlled structure. From a system design viewpoint, with the advent of multi-core processor systems, a network is a natural architectural choice. An NoC can provide separation between computation and communication, support modularity and IP reuse via standard interfaces, handle synchronization issues, serve as a platform for system test, and, hence, increase engineering productivity.

Parallelism and Scalability

The wires in the links of the NoC are shared by many signals. A high level of parallelism is achieved, because all links in the NoC can operate simultaneously on different data packets. Therefore, as the complexity of integrated systems keeps growing, a NoC provides enhanced performance (such as throughput) and scalability in comparison with previous communication architectures (e.g., dedicated point-to-point signal wires, shared buses, or segmented buses with bridges). Of course, the algorithms must be designed in such a way that they offer large parallelism and can hence utilize the potential of NoC.

2.5 Discussion and Analysis

Till now we have discussed about various aspects of NOC, Now let us discuss the NOC's area and performance overhead compared to traditional architecture based on fixed interconnectivity.

Area overhead

This is hard to quantify, especially at this stage when the NOC is a concept. Moreover, NOC being a reprogrammable architecture like FPGAs, different designs will utilise the resources to a different extent. A more relevant question in this context would be the overhead of the switch based interconnect scheme. At this stage, we are not in a position to answer this question concretely. But given the fact that the proposed interconnect scheme allows any resource to reach any other resource, an overhead, we believe, is justified.

Performance overhead

A more concrete answer than the area overhead can be given here. Interconnect delay already dominate the performance equation. According to ITRP99 by the time we reach .1 micron, the delay in long wires would be 100x compared to the fastest switching gate and with clocks ticking at 10 GHZ, the interconnect delay for long wires would be 10s of cycles.

Taking these facts into account, we believe that switch based interconnect scheme will provide natural pipelining with the added benefit of interconnecting all resources.

Switches being control and memory intensive, the depth of logic in them would be significantly lower compared to that in the computational blocks. This observation leads to two conclusions.

• The first is, if the delay in wire is going to be 100x compared to a gate and if switches imply a small depth of gates, the switches would add an insignificant amount of delay.

• Secondly, maintaining global synchronous assumption is already proving to be difficult and designs involving multiple clock domains are becoming common (need to refer to our and other GALS work). Again based on the fact that the computational blocks would have much greater depth of logic compared to switches, we can think of clocking the switches at a much higher clock compared to computational blocks. In such a case, the latency introduced by the switches, as seen by the computational blocks would be attenuated by the ratio of the switch clock to computation clock.