NOC Architecture Design Methodology

Article · January 2006

4 authors, including:

Ankur Agarwal
Florida Atlantic University

Abu Asaduzzaman
Wichita State University

NOC 구조 설계 방법론

Ankur Agarwal* · A. S. Pandya* · Abu Asaduzzaman* · 노영욱**

NOC Architecture Design Methodology

Ankur Agarwal* · A. S. Pandya* · Abu Asaduzzaman* · Young-Uhg Lho**

요 약

다중처리기 SoC(System on Chip) 플랫폼은 SoC 설계를 위한 새로운 혁신적인 경향들을 가지고 있다. QoS 인수와 성능 매트릭스는 SoC을 위한 새로운 설계 방법론을 채택하도록 하였다. 이것은 NOC의 하부 통신 백본뿐만 아니라 전체 시스템 구조가 고도로 확장가능하고, 재사용가능하고, 예측가능하면서 가격과 에너지 측면에서 효율적인 플랫폼이 되도록 구체화할 것이다. 우리는 NOC의 통신 백본 구조가 계층화된 것처럼 NOC의 전체 시스템 구조가 자체적으로 7 계층이 되도록 제안한다. 이런 플랫폼은 동기화 문제를 가지는 병행성을 보다 효과적으로 모델화하는 영역에 특수한 문제들을 분리할 수 있다. 그러한 계층 구조에서 계산 모델은 어떤 응용에 자연스러운 병행성과 동기화 문제를 모형 할 수 있는 뼈대를 제공할 것이다. 그러므로 특정 NOC 영역에서 올바른 계산 모델을 사용하는 것은 아주 중요하다.

ABSTRACT

Multiprocessor system on chip (MPSoC) platforms has set a new innovative trend for the SoC design. Quality of service parameters and performance matrix are leading to the adoption of new design methodology for SoC, which will incorporate highly scalable, reusable, predictable, cost and energy efficient platform not only for underlying communication backbone but also for the entire system architecture of NOC. Like the layered architecture for the communication backbone of NOC, we have proposed the entire system architecture for NOC to be a seven layered architecture in itself. Such a platform can separate the domain specific issues which will model concurrency along with the synchronization issues more effectively. For such a layered architecture, model of computation will provide a framework to that can model concurrency and synchronization issues which are natural for any application. Therefore it becomes extremely important to use a right computation model in a specific NOC region.

키워드

Network on Chip(NOC), SoC, Layered Architecture, model of computation

Ⅰ. INTRODUCTION

The System Level Design era like NOC (Network on Chip) where creativity, innovative ideas, ingenuity and inspiration come to the fore is approaching and pushing technology at every turn. Moore's law predicts that a chip in 2010 will count more than four billion transistors operating in multi GHz range [1,2]. This is mainly due to linearly decrease in the transistor size enabling faster transistor switching times and more densely integrated circuits. Such computation

* Dept of Computer Science and Engineering, FAU
** 신라대학교 컴퓨터교육과

접수일자 : 2005. 8. 26

power has posed some challenges which include the disparity in transistor and wire speed, power dissipation leading to a decrease in the area of the chip which can be utilized with a single clock cycle [3,4]. Another dominant factor is to be able to design the system in an acceptable timeline know as time-to-market. As a result, system level designers are constantly looking for ways to provide a set of Quality of Service (QoS) parameters and performance matrix. Figures 1 and 2 depict the possible QoS and performance parameters for a system respectively.

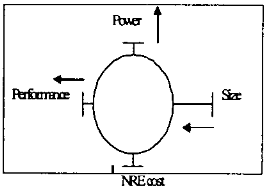

Fig. 1 QoS parameters

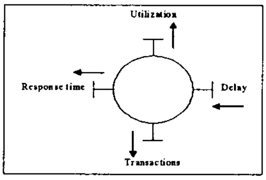

Fig. 2 Performance parameters

Technology scaling at the same time has unwanted side effects which include cross coupling, noise, and transient errors [5]. This has again led us to reuse the design blocks sometime referred as components which have been carefully designed by the expert designers. However it can never be guaranteed that those sub-micron effects will not pop-up again while reusing those components into a design of a sub-system or a system. Thus it can be concluded that components or the sub-system which perform as expected might not perform in the same way after the system integration [6]. This has resulted into a new domain of research work for system level integration and verification for the NOC architecture [7]. It can be seen that the manufacturing non recurring expense (NRE) of the chip with RTL design methodology would be enormous than the one incorporating the future improvements[8]. It is due to these improvements that the team of engineers and the managers were able to bring the cost of a product development to an affordable price for the customers with enhanced quality of service and customer support than the previous versions. However, if such innovations and future trends are not brought in the early stage of the product development cycle then the NRE cost of the product can shoot up-to an unaffordable amount of one billion dollars. Thus it is expected that the future systems will have increasing role of design automation, reusability, componentization increasing the market share for the electronic design automation (EDA) Industry. For such scenario, System-level modeling environment should be developed that essentially should support the middle-out design philosophy to exploit reuse to the maximum in order to reduce the design effort [9,10]. The high volume of reuse should cut down the overall system design cycle [11,12].

Ⅱ. Background Work

To be able to integrate a complex system comprising of hundreds and thousands of cores with adequate memory and communication backbone on a single silicon die is economical but at the same time highly complex to integrate[13]. Designing such systems on a chip (SOC) is a complex process, and is currently approached with little organized principles[14]. Researchers have contributed in various dimensions into this domain to make such an effort plausible. One such design discipline is platform based design [15,16]. Essential elements of platform-based design are the functional behavior of each core, along with the modeling of interaction among the cores. This orthogonalization of concerns is essential to the success of a reuse strategy as has been realized in recent years[15]. At the same time this platform is aimed at making highly scalable and configurable so that it can be adopted to the needs of different workloads, which maintaining the generality of application development methods and practices. As emphasized in the ITRS 2001 document [8], it is very important, especially at system level, to separate the computation from communication[13], as they are orthogonal issues which should remain separate whenever possible. From the communication side several researchers have suggested 2-D mesh architecture for NOC [14] which consist of resources and network. Figure 3 depicts such architecture. The network elements are switches, channels and resource-network interface. The resources for the NOC can be any core, memory, controller, FPGA, ASIC or other hardware block. This infrastructure will be ideal where QoS and performance parameters will be traded off based on the user requirement. New algorithms have been proposed in this domain to reduce the power and energy consumption as well for cost optimization. It has been a well established fact that such NOC architecture will be based on packet switched network which has lead to the new and efficient principles for the design of the routers for the NOC architecture.

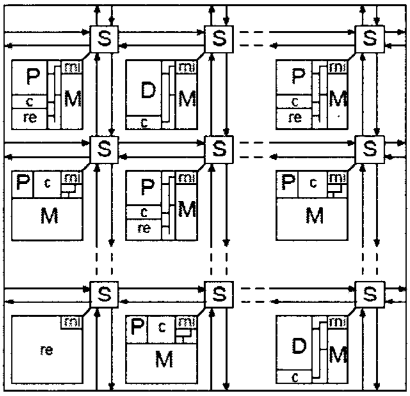

Fig. 3 NOC design with switches and resources

An energy efficient design for the router has been proposed as shown in Figure 4. These routers will be responsible for routing the entire traffic across and have to be interfaced with the switches and the resources in the NOC architecture.

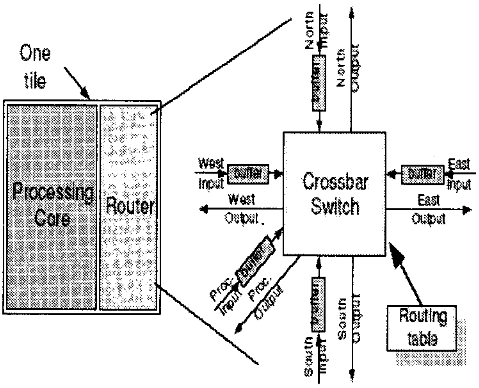

Fig. 4 Energy Efficient Design of a Router

The design of the resource network interface (RNI) should again be highly scalable and re-usable to be able to be integrated with different types of resources without the need for the change of the core RNI for different types of resources. With the concept of the routers for NOC the wide research has been done proposing the right data format needed for various layers in the protocol stack. One such design of a reusable switch which would effectively route the packet in the entire NOC with the capability of buffering the packets at the input and output is shown in Figure 5.

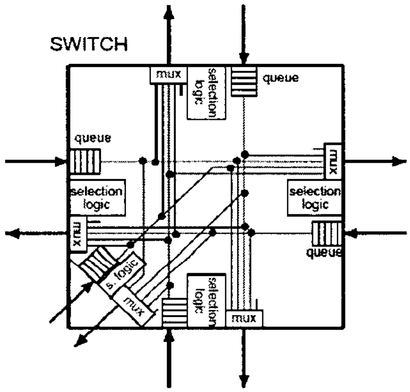

Fig. 5 Design of a switch

Once the design of the basic NOC architecture became a standard solution for NOC architecture, new techniques have evolved addressing the advanced issues such as the dynamic load balancing on to a node of NOC architecture, the shortest path for the data flow for NOC and the energy efficient NOC architecture design. At one end when the researchers have focused onto the communication architecture for NOC, few have focused on exploiting the computing capability of NOC. The computer architects are coming up with the innovative ideas to coup with the system complexity. One such novel way is to divide the system into smaller, local decoupled synchronous regions and then composing from the local solution. These synchronous regions would be easier to integrate, and verify. At the same time there will be an asynchronous way in which all the local synchronous regions will be managed at the top level referred as global solution. Thus these different synchronous regions need not to have a single clock domain to work. This approach would reduce the requirement for the clock tree designers as they needed to worry about the local synchronous regions which mainly consist of the components and the subsystem rather than the system as a whole. At the same time due to the flexibility in the reduction of the clock speed would reduce the amount of power consumption in a system.

Ⅲ. Novel Approach for NOC Architecture

The key challenge for the system level design with integration of thousands of cores is the modeling of concurrency together with synchronization issues, which in turn is to capture concurrent communication formally in a Model of Computation (MOC). This defines the two goals for the system level designers which are to model the system functionality well in advance of building the actual computing system in order to provide the level of flexibility in the system design, and to be able to manipulate the abstract set of design elements at the same time which can generate the different set of QoS parameters and performance matrix as per the user requirement. For designing such a system we first define NOC layered architecture and then try to integrate different MOC onto this layered architecture.

3.1 Layered Architecture for NOC Design

Layered architecture is an effective way of divide and conquer a problem. We have seen it working in an effective way as an example of open system interconnection (OSI) model. This defines a way of separation of concern of each domain providing an effective solution to a domain specific problem. It can also be argued that this approach will provide with a scalable solution in addressing the inter-domain specific issues. Thus we have defined seven-layer architecture for the NOC as shown in Figure 6.

Application
Algorithm
RTOS
Architecture
Communication protocol
Communication backbone
VLSI

Fig. 6 7 layered architecture for NOC design

3.1.1 Application

This will be the top level functionality of the NOC system where different applications are suppose to run. This layer in turn will contain the application software. As a NOC system should be able to support the wide range of the applications, needing a full support in form of the system libraries. These software applications should also be portable in nature so that we can be transferred from one domain into the other providing more reuse to the system level designers. Thus these applications should be well designed to have the proper interfaces with the system in order to exploit the reuse.

3.1.2 Algorithm

This will be a next lower level layer where the control and multimedia algorithm along with the optimization algorithm will run. These algorithms will be application and performance specific.

3.1.3 RTOS

The next lower layer will comprise of the real time operating system. This RTOS will be responsible for effective scheduling of the NOC resources to the application software. RTOS will be scheduling the application based on the priority which is turn will be defined by the QoS parameters and performance matrix. Along with the scheduling its other main functionality will be managing the concurrency and synchronization issues.

3.1.4 Architecture

Architecture exploration is done at this layer. Based on some simulation results, a set of optimal architectures is selected to run the target application(s). Hardware and software is traded off depending upon the performance matrix such as utilization, latency, and transactions. It will address the concerns related to the speeds of the processor, cache, bus, and memory and the cache parameters such as cache size, line size, associativity, and cache levels.

3.1.5 Communication Protocol

Packet switching has been proved to the right way to model this layer. This will define the size of the packet, the queuing discipline, routing strategies, and the issues related to the traffic density. For real time application this layer will forward the traffic on to a different reserved channel so that the timely response can be guaranteed in order to generate the required level of service. The algorithms related to dynamic load balancing can also be addressed as a part of this layer. This communication protocol layer in turn will have to be modeled and influence of a protocol stack design.

3.1.6 Communication Backbone

This layer below communication protocol will comprise of actual implementation of routers, switches, resource network interface along with buffers, busses, queues, and FIFO among other. This is the layer where actual transfer of data among the different resources actually takes place.

3.1.7 VLSI

This will be the final system implementation at the transistor level. It will convert each and every element of the system into the corresponding transistor level design. The VLSI related issues will be abstracted to the higher layers and will be addressed there thus easing the job of the designers.

3.2 Model of Computation

A MOC is a mathematical formalism that captures the communication among the concurrent systems. For such synchronous and asynchronous way of communication and management to coexist there should be more than one MOC needed for the integration of the system. In such a system each component or the subsystem of the system should be able to use any plausible MOC or the derivative of MOC and more importantly should retain its behavior after integration of the system. There can be another plausible case in which a subsystem lies in two or more domains of the NOC connected by the switch. For example if a digital camera can be considered as a component or a subsystem depending upon the complexity of the system, a communication handheld device (system) then it possible that its design would not fit into one local region. Under such scenario we should be able to the address the concurrency and synchronization issues across the domain.

Theoretical computer science treats any computational subject for which a good model can be created. Research on formal models of computation was initiated in the 1930s and 1940s by Turing, Post, Kleene, Church, and others. In the 1950s and 1960s programming languages, language translators, and operating systems were under development and therefore became both the subject and basis for a great deal of theoretical work. Based on our theoretical knowledge of MOC for Systems-on-Chips by Jo Ann Paul and Don Thomas from Carnegie Mellon university, we like to integrate various models of computation to model a complex computing system. Thus we have addresses the various MOC and exploited their behavior to address the domain specific issue for the NOC architecture.

3.2.1 NOC at System Level

At the system (top) level, NOC should be able to address the high level issue rather than addressing the lower level (RTL or sub-micron) level issues. If we think a digital camera as a NOC then the system level issues to be addresses related to this would be resolution, image size, power and cost, which we would refer as parameters. Thus we should be able to adjust one of these parameters which would in turn automatically adjust the lower level details by process of automation. Thus we can ague that at system level we would need some kind of manager to be help responsible for managing all the system level design parameters.

At the same time such a control (manager) should be some level of concurrency in time domain than it having a sequential behavior. This requirement rules our finite state machine (FSM) model to sit at the top as a manager. At the system level this manager should be able to observe and control only those signal which are changing their state or behavior rater than monitoring all the signals at every clock event. A discrete event simulation at the system level would suite such a requirement.

Another such possibility of asynchronous way of communication and management can be realized by process network (PN) domain. This domain was described by Kahn and MacQueeen. Two important properties of the PN domain which make such computation plausible are that processes communicate asynchronously and that the memory used in the communication is bounded whenever possible. The PN domain has the capability to model a system as a network of processes that communicate with each other by passing messages through unidirectional first-in-first-out channels.

3.2.2 NOC at Subsystem Level

The local region for NOC is again divided into two different domains as NOC at subsystem level and NOC at component level. At sub-system level we will have to address the issues related to a particular subsystem rather than the global issues. Such a sub-system will usually be some DSP sub-system, IP-core, field programmable gate arrays (FPGA), or application specific integrated circuit(ASIC). This subsystem in conjunction with the packet switching network would required to produce some fixed amount of data (tokens) at the input generating some fixed amount of token at output for being routed over the network. Synchronous data flow (SDF) is one such MOC which has the similar properties. At subsystem, there is also some need for a small control element. This control element would be required to address the synchronization issue at that subsystem level only making FSM as another plausible solution to MOC at subsystem level.

3.2.3 NOC at Component Level

This is another integrated part of the local region. These components together would constitute the higher subsystem level. At this level designer will not have to worry about addressing the system wide concurrency and synchronization issues and the design should be highly reusable to be able to utilized into other products and scalable to be mapped into the higher domains like subsystem and system. This component level would comprise of software component, computation part which in turn would be represented by electrical components and computer architecture component. For electrical components to be specified in such domain, we can use the mathematical equations to represent the properties of the model giving continuous time (CT) domain as a plausible solution. Hierarchical FSM can contribute to the synchronization among the software elements and the hardware components at component level where each state can be programmable in itself. Such a state machine is often referred at state charts where each state has trigger, action and guard condition. At the same time we would require to exploit the property of SDF domain to model the architectural issues.

Thus it can be realized that we need some platform to support which can integrate different models of computation to exploit the computation and capability of the future generation systems. At the same time such a platform should be able to model the software part, hardware part along with the communication model and the scheduler. Such a platform would not distinguish between hardware and software model but rather integrate both into the system in the same way. EDA software product development teams are constantly striving to balance by increasing the intelligence and functionality of the EDA tools by incorporating more and more sophisticated algorithms in the tool in order to improve the system architecture and meet the rapid changing requirement.

Ⅳ. Conclusion

Despite of many challenges the NOC offers we believe that layered architecture is one of the effective plausible paths for designing the system. The NOC design methodology seems to dominate the future design trends due to increasing system complexity which in turn is motivated by the need to support number applications on to the system. Under such a scenario the design must remain restricted to such a complexity level that it can be managed by a human. Under such a scenario it becomes extremely important to design a layered architecture so the design complexity can be easily managed and design issues can be addressed at different levels in the layered architecture.

REFERENCES

[1] L. Benini and G. De Micheli. "Networks on chip: a new SOC paradigm," IEEE Computer, January, 2002.

[2] Semiconductor Industry Association, The international Technology Roadmap for Semiconductors. 2001.

[3] Hemani, Axel Jantsch, Shashi Kumar Adam Postula, Johnny berg, Mikael Millberg, Dan Lindqvist, "Network on Chip: an architecture for billion transistor era," Proc. Of IEEE NorChip Conference, pp.8, 2000.

[4] Paul Wielage and Kees Goossens, "Network on silicon: Blessing or nightmare?," In Euromicro Symposium on Digital System Design, Dortmund, Genmany, Keynote Speech, September 2003.

[5] Tejasvi Das, Clyde Washburn, P. R. Mukund, Steve Howard, Ken Paradis, Jung-Geau Jang and Jan Kolnik, "Effects of Technology and Dimensional Scaling on Input Loss Prediction of RF MOSFETs," International Conference on VLSI Design, pp.295-300, 2005.

[6] Alexandre M. Amory, rika Cota, Marcelo Lubaszewski and Fernando G. Moraes, "Reducing test time with processor reuse in network-on-chip based systems," Proceedings of the 17th ACM symposium on Integrated circuits and system design, pp.111~116, 2004.

[7] Cota, E.; Carro, L.; Wagner F and Lubaszewski M, "Power-aware NoC reuse on the testing of core-based systems," Proceedings of ITC 2003 International, Vol. 1, pp.612~621, 2003.

[8] ITRS 2001. Available: http://public.itrs.net/ Files/ 2001ITRS/ Home.htm

[9] Edward A. Lee and Yuhing Xiong, "System Level Types for Component-Based Design," Workshop on Embedded Software, California, October 2001.

[10] Y. Xiong and E. A. Lee, "An Extensible Type System for Component-Based Design," The 6th Intl. Conf. on Tools and Algorithms for the Construction and Analysis of Systems, Berlin, Germany, April 2000.

[11] R. A. Bergamaschi, S. Bhattacharya, R. Wagner, C. Fellenz, M. Muhlada, F. White, W. R. Lee, and J.-M. Daveau. "Automating the Design of SOCs Using Cores," IEEE Design and Test of Computers, Vol.18 No.5, pp.3244, Sep. 2001.

[12] Cota, E., Kreutz, M., Zeferino, C.A., Carro, L., Lubaszewski, M., and Susin, A., "The impact of NoC reuse on the testing of core-based systems," Proc. of VLSI Test Symposium, pp.128~133, Apr. 2003.

[13] A. Jantsch and H. Tenhunen, Networks on Chip Kluwer Academic Publisher, 2003.

[14] S. Kumar, A. Jantsch, J-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, and A. Hemani. "A Network on Chip Architecture and Design Methodology," In IEEE Computer Society Annual symposium on VLSI, pp. 117-124, Apr. 2002.

[15] K. Keutzer, S. Malik, A. Richard Newton, Jan M. Rabaey and A. Sangiovanni-Vincentelli, "System Level Design: Orthogonalization of concerns and platform based design," IEEE Transaction on CAD of Integrated Circuits and Systems, Vol. 19 No. 12, pp. 1523-1543, 2000.

[16] A. Ferrari and A. Sangiovanni Vincentelli, "System Design: traditional concepts and new paradigms," In Intl. Conf. on Computer Design, pp.2~12, 1999.

저자소개

Ankur Agarwal

is a Visiting Instructor and a Ph.D. student at the Computer Science and Engineering Department, Florida Atlantic University. He pursued his MS in computer engineering from Florida Atlantic University in year 2003. He also holds two post graduate diplomas in VLSI design and real-time embedded system design. He has earned his bachelor of engineering from Pune University, India in year 2000.

※ Research Areas : concurrency modeling, system level design, network-on-chip, real-time-operating system and VLSI design.

A. S. Pandya

is a professor at the Computer Science and Engineering Department, Florida Atlantic University. He received his undergraduate education at the Indian Institute of Technology, Bombay. He earned his M.S. and Ph.D. in Computer Science from the Syracuse University, New York. He has worked as a visiting Professor in various countries including Japan, Korea, India, etc.

※ Research Areas : VLSI implementable algorithms, Applications of AI and Image analysis in Medicine, Financial Forecasting using Neural Networks.

Abu Asaduzzaman

is, currently, a Ph.D. candidate in the department of Computer Science and Engineering, Florida Atlantic University. He received his MS degree in computer engineering from FAU in 1997. He worked for ECI Telecom as a software engineer from 1998 to 2001. From 2001 to 2003, he worked for BlueCross and BlueShield of Florida and SunPass (FDoT) as an IT Consultant. Currently, he is working as a research assistant at CSE Dept, FAU.

※ Research Areas : cache optimization, architecture exploration, embedded system evaluation, and networks-on-a-chip.

노영욱(Young-Uhg Lho):교신저자

1985년 2월 부산대학교 전산통계학과 학사
1989년 2월 부산대학교 전산통계학과 석사
1995년 2월 부산대학교 전자계산학과 박사
1989년 ~ 1996년 한국전자통신연구소(ETRI) 연구원
1996년 ~ 현재 신라대학교 교수

※ 관심분야 : 내장형시스템, 멀티미디어 시스템, 병렬분산 시스템, 지능형시스템, 실시간운영체제, 컴퓨터교육