Feature Article

Projecting Network-on-Chip as a System-on-Chip platform for safe information

Alexander Pavlovich Galkin[1], R. H Al-Gaberi[2], H. M. Obadi[2] (Yemen), M. M.Amro[2] (Jordan)

Abstract
Different System-on-Chip(SoC) applications produce different traffic patterns and require different performances. For this reason, the Network-on-Chips (NoCs) should be scalable also for varying performance requirements in addition to the different system sizes.

Innovations occur where challenges are present. Network-on-Chip (NoC) was pro- posed in face of those challenges in the System-on-Chip (SoC) community [1].The GigaScale Research Center (GSRC) suggested NoC to address interconnection woes [2]. Researchers from the Philips Research presented a router architecture supporting both best-effort and guaranteed-throughput traffic for Network-on-Silicon with safe information [3].While network-on-chip is still in its infancy, concept has spread and been accepted in academia very rapidly.

Aimed to be a systematic approach, NoC proposes networks as a scalable, reusable and global communication architecture to address the SoC design challenges. It features a mesh structure composed of switches with each resource connected to exactly one switch, as shown in fig.1.

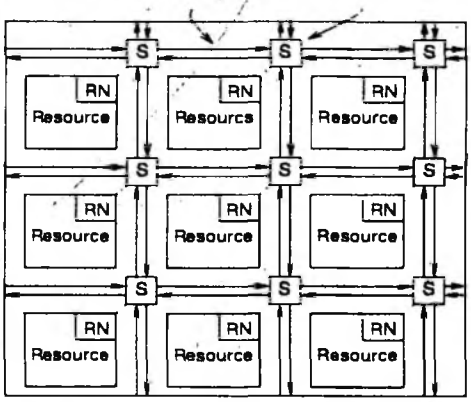

Fig. 1. A mesh NoC with 9 nodes

Aresource can be a processor, memory, ASIC, FPGA, IP block or a bus-based subsystem. The resources are placed on the slots formed by the switches. The maximal resource area is defined by the maximal synchronous region of a technology. The resources perform their own computational, storage and/or I/O processing functionalities, and are equipped with Resource-Network-Interfaces (RNIs) to communicate with each other by routing packets instead of driving dedicated wires. Communication network is a well-known concept developed in the context of telephony, computer communication as well as parallel machines. On-chip networks share many characteristics with these networks, but also have significant differences.

Cost is a major concern for on-chip networks since most SoCs target high-volume markets. The buffering in an on-chip network has very limited space and is expensive in comparison with board-level, local-area and wide-area networks. This means that a NoC allows a limited count of routing tables and virtual channel buffers in network nodes. Power consumption is important for all kinds of networks. However, on-chip networks are developed also for embedded applications with battery-driven devices. Such applications require extremely low power which is not comparable to large-scale networks. As we also mentioned, on-chip network designs are confronted by the Deep

1) Dr.Sci.Tech., the professor, The Vladimir State University: Russia, 600000, Vladimir, str.Gorkogo,87, phones:+7(4922) 47-97-95, +7(905) 1459898; E-mail: galkin@vlsu.ru
2) Post-graduate students

SubMicron (DSM) effects. Taming bad physical effects is as important as network design itself. Furthermore, many SoC networks are developed as a platform for multiple use cases, not only for a single use case. Therefore designing micro-networks also need to take reconfigurability into account.

As we view it, NoC is a revolutionary rather than evolutionary approach to address the SoC design crisis. It shifts our focus from computation to communication. It should take interconnect into early consideration in the design process, and might favor a meet-in-the middle (platform-based) design methodology against a top-down or bottom-up approach. NoC has the following features:

• **Interconnect-aware [2]:** As the technology scales, the reachable region in one clock cycle diminishes. Consequently, chip design is becoming communication-bound rather than capacity-bound. Since the size of a single module is limited by the reachable region in one cycle, to exploit the huge chip capacity, the entire chip has to be partitioned into multiple regions. A good partitioning should be regular, making it easier to manage the properties of long wires including middle-layer and top-layer wires.

Each module is situated in one partitioned region and maintains its own synchronous region. In this way, the reliance on global synchrony and use of global wires can be alleviated. To guarantee correct operation, registers may be used in wire segments to make the design latency-insensitive [3]. Besides, each IP may be attached to a switch. Switches are in turn connected with each other to route packets in the network. The signal and power integrity issues may be addressed at the physical, link and higher layers. For example, redundancy in time, space and information can be incorporated in transmission to achieve reliability. By physically structuring the communication and successfully suppressing the DSM effects, the design robustness and reliability can be improved.

• Communication-centric [1-3]: Networking distributed IP modules in a partitioned chip results in a naturally parallel communication infrastructure. As long as the chip capacity is not exceeded, the number of cores which can be integrated on a single chip is scalable. The inter-core communications share the total network bandwidth with a high degree of concurrency. The network can be dimensioned to suit the bandwidth need of the application under interest. The parallel architecture allows concurrent processing in computation and communication. This helps to leverage performance and reduce power in comparison with a sequential architecture permitting only sequential zed processing. A protocol stack is typically built to abstract the network-based communication. Each layer has well-defined functionalities, protocols and interfaces. The design space at each layer has to be sufficiently explored. The tradeoffs between performance and cost should be considered in the design, analysis and implementation of the communication architecture. Quality-of-Service (QoS) and system-wide performance analysis are central issues to address predictability.

• Platform-based [1]: Since the cost of design is the major obstacle for innovative and complex SoCs, developing a programmable, reconfigurable and extensible communication platform is essential for SoC designs. To this end, NoC shall serve as a communication and integration platform providing a hardware communication architecture, an associated interconnect interface, as well as a high-level interface for integrating hardware IPs, custom logic and for software programming. This enables the architecture level reuse. One challenge is to address the balance between generality and optimality. A platform must serve not only one application but also many applications within an application domain. On the other hand, customization to enhance performance and efficiency is needed to make designs competitive. Providing well-defined interfaces at least at the network level and the application level is important, because it enables IPs and functional blocks to be reusable. Interface standardization is one major concern to make IPs from different vendors exchangeable. It must be efficient and also addresses legacy IPs. The concept of interface-based design has been shown successful for IP plug-and-play in the history of software and hardware developments, for example, instruction sets and various interconnect buses or protocols such as Peripheral Component Interface (PCI) and Universal Serial Bus (USB). A NoC design methodology should also favor communication interfaces for the greatest possible IP reuse and integration [2,3]. Using validated components and architectures in a design flow shrinks verification effort, reduces time-to-market and guarantees product quality, thus enhancing design productivity.

As such, NoC research does not deal with only several aspects of SoC design but creates a new area [3]. The term NoC is used today mostly in a very broad meaning. It encompasses the hardware communication infra-structure, the middleware and operating system, application programming

interfaces [2], the design methodology and its associated tool chain. The challenges for NoC research have thus been distributed in all aspects of SoC design from architecture to performance analysis, from traffic characterization to application design.

NoC classification. Communication networks used in computer systems as well as the NoCs can be roughly classified to shared medium networks, direct networks, indirect networks, and hybrid networks according to how their resources, which are the network interfaces, switch nodes and communication channels, are connected to each other and how the communicating processor nodes are connected to the network. Fig. 2 depicts a classification of the NoCs which is based on the differences between the topologies and which is approximately similar to that of the computer networks presented in [2]. The topology determines many properties of the NoCs like, for example, the usable alternative routing algorithms, the scalability, the performance and the set of possible applications. The shared medium networks are basically on-chip system buses which are shared by multiple processors. The direct networks are composed of nodes which contain one processor and one switch which the processor uses for communicating with the other nodes. The number of switches and the processors are equal in direct networks. In indirect networks processors are connected only to edge switches, and the number of processors can be quite different from the number of switches. The hybrid networks may have features of more than one of the previously presented network topologies.

Shared on-chip buses on top of the classification depicted in fig. 2 include AMBA [2], and CoreConnect [3] which are supplied as reusable commercial IP blocks by different companies. Both of these on-chip bus architectures allow fast development and integration of reusable IP cores into SoCs. The problem of the shared on-chip buses is their bad scalability for large SoCs. This is because the wires become long and wire delays high as the number of blocks connected to the bus becomes high, which reduces the operation rate and throughput of the shared buses. AMBA is a hierarchical shared bus which is divided by a bridge into system bus (ASB) and peripheral (APB) bus. The processor and the main memory are connected to the ASB segment of the AMBA and the slower peripheral blocks to the APB segment. The ASB is also connected to an external bus interface and a test control interface, which makes it possible to test the SoCs and their IP blocks with external testers. CoreConnect is also a hierarchical bus which consists of different segments. The Processor Local Bus (PLB) can be connected to one or more processors and the On-Chip Peripheral Bus (OPB) connects slower peripheral blocks to the system.

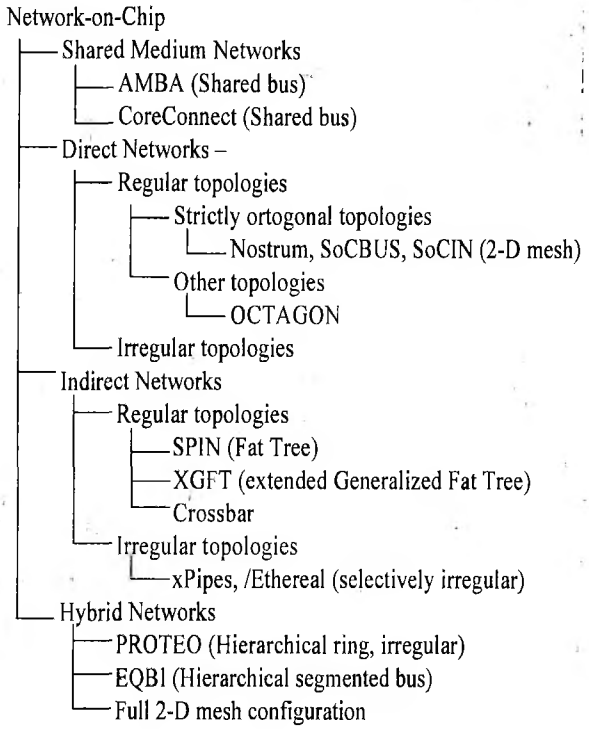

Fig. 2. Classification of Networks-on-Chip

The PLB can be implemented either as a simple shared bus or as a crossbar which the master devices like the processors can arbitrate independently.

Direct network topology is Octagon [1,2] where network nodes and bidirectional links between the adjacent nodes form a ring. In Octagon the nodes in the opposite sides of the ring are also connected to each other with bidirectional links. As the number of communicating nodes exceeds eight the nodes are arranged into two or more rings which have common nodes which operate as bridges. The advantage of the Octagon is a low diameter and its disadvantage is the high length of the longest links. The direct networks could also have an irregular topology optimized for some application, although such an example is not presented here.

Different SoC applications produce different traffic patterns and require different performances. For this reason, the NoCs should be scalable also for varying performance requirements in addition to the different system sizes. The topologies of the indirect networks can be more easily optimized for some particular application than those of the

direct networks, because the number of switch nodes can be chosen independently of the number of processors connected to the network. Therefore, they are more suitable for MPSoCs from this perspective. SPIN has a fat tree topology [2,3] which is a bidirectional multi-stage interconnection network (BMIN). It is a pure packet switch NOC where wormhole routing technique is used. It can be scaled for different system sizes which are equal to exponents of number two, and it routes packets with adaptive Turnaround routing [3].

Irregular indirect topologies where the degrees of switch nodes can be chosen independently of the degrees of the other switch nodes allow even more optimized NoC implementations. XPipes [3] is a library of soft components which can be used for implementing customized NoCs with irregular indirect topology for SoCs which contain general-purpose programmable processors, DSPs, hardware accelerators, memory blocks, I/O blocks etc. It consists of soft macros of network interfaces, switches and switch-to-switch links and it is targeted for interconnect centric design flows where pre-designed and pre-verified intellectual property (IP) blocks are integrated into an interconnect-centric system architecture. Since basically most of the SoCs integrate a heterogeneous set of specialized components, Xpipes allows the usage of irregular highly optimized network topology to be customized for every application individually. For example, three different network configurations for MPEG4 decoder implemented with XPipes are presented in [3]. Different network configurations consist of different number of switch nodes, and the degree of the switches may be different even in the same network configuration. The power consumptions, silicon areas, and routing latencies of the network configurations are also different, which illustrates how the networks can be optimized. Despite the irregularity of practical network topologies aimed at the SoCs to be composed of heterogeneous components, the Xpipes could also be used for generating, for example, 2,3-D mesh-networks depending on the application. The application can be presented as a core graph the vertices of which are the communicating components of the system. The edges of the core graph are annotated with the communication bandwidth between the components. The NoC design flow based on the usage of Xpipes Compiler reads among the other things the core graph as an input and generates a high-level SoC floorplan which includes the communicating blocks, the switch nodes and the communication links. The routing is source routing with look-up tables which are also generated by the Xpipes Compiler for every switch node separately.

Ethereal [3] is another NoC which can be flexibly configured for different applications and which can be used for implementing NoCs with irregular indirect topologies. It provides separate classes for best-effort (BE) and guaranteed services (GS) traffic. The BE (best-effort) traffic is routed through the NoC like in other networks with conventional wormhole routing, whereas the GS traffic is routed by pipelined time-division-multiplexed circuit switching. In one of the configurations every switch node also consists of two separate switches for switching GS and BE traffic and forms a combined GS-BE-switch. The GS switches have slot tables which map output ports to input ports for every time slot. All of the switch nodes use the same fixed-duration time slot and their operation must be synchronized. This can be done by using the same centralized synchronous clock in every switch node. Alternatively the synchronization can be based on tokens which output ports generate and send to input ports within the same switch node so as to indicate that they can receive more FLITs. The Ethereal is probably the most advanced NoC presented up till now, but it requires also quite complex design flow, which may be a disadvantage.

References:

1. V. Broido Computer systems, networks and telecommunications the textbook for high schools. – Sankt-Peterburg.: Peter, 2008,-768 p.
2. A.P. Galkin, etc. Improving fault tolerance computing networks for multihoming//News of institute of engineering physics. 2012.№3.P. 22-24.
3. A.P. Galkin, etc. Minimization of routers at maintenance of information protection in networks//News of institute of engineering physics. 2013.№1.P. 2-4.