ON-FPGA COMMUNICATION ARCHITECTURES AND DESIGN FACTORS

Terrence S. T. Mak *, *Pete Sedcole, Peter Y. K. Cheung*

Department of Electrical and Electronic Engineering
Imperial College London
email: {t.mak, pete.sedcole, p.cheung}@imperial.ac.uk

Wayne Luk

Department of Computing
Imperial College London
email: w.luk@imperial.ac.uk

ABSTRACT

The recent development of Platform-FPGA or Field-Programmable System-on-Chip architectures, with immersed coarse-grain processors, embedded memories and IP cores, offers the potential for immense computing power as well as opportunities for rapid system prototyping. These platforms require high-performance on-chip communication architectures for efficient and reliable inter-processor communication. However, as the number of embedded processors increases, communication bandwidth between embedded components becomes a limiting factor to overall system performance. In this paper, we survey the state-of-the-art on-FPGA communication architectures and methodologies. Salient factors, which include quantitative performance metrics and qualitative factors, relevant to design are identified and used to analyze and classify the on-FPGA communication architectures. This survey aims to facilitate innovation in and development of future on-FPGA communication architectures.

1. INTRODUCTION

On-FPGA communication is important to provide high-bandwidth and reliable data transfer between processing elements, and is therefore fundamental to overall FPGA-based system performance. In recently developed FPGA architectures, such as the so-called Platform-FPGA [1, 2] and Field-Programmable System-on-Chip [3], pre-fabricated coarse-grained modules including microprocessors, DSP units and memory modules are immersed into the fine-grain programmable fabric. These can provide significant improvements in speed, area as well as hardware configuration time [1, 4]. Computational intensive tasks such as video and image processing, biometric analysis and cryptographic analysis can gain speed in order-of-magnitude with judicious hardware implementation using Platform-FPGA [5]. This is also an effective solution to tackle the application and integration complexity challenges for product development with time-critical constraints. Furthermore, a communication architecture can be used as an interconnect backbone for different coarse-grain components, providing a plug-and-play style of modularity to facilitate system integration. However, as the number of embedded components increases, the communication bandwidth between embedded components becomes a fundamental factor to consider in overall system performance. As a result, there is a requirement to advance communication architectures for better performance and scalability.

However, technology scaling trends indicate that the performance and energy consumption characteristics of global interconnect are rapidly deteriorating, relative to logic components [6]. This will result in a significant performance gap between on-chip interconnects and logic gates, and the gap will continue to grow even with the help of new interconnect materials and aggressive interconnect optimization [7]. Programmable interconnect, such as local and global wire segments and programmable switches, are the most abundance resources in FPGAs and are fundamental to system performance and power dissipation. On-FPGA communication architectures, which are built on top of the programmable interconnect fabrics, need to tolerate and adapt to the technology scaling challenge.

There are many notable efforts devoted to addressing the on-FPGA communication issues, and a considerable number of different architectures have been proposed. In this paper we present a survey of on-FPGA communication architectures. Although general on-chip communication architectures will be presented as background information, our main focus is on the most recent FPGA-based communication architectures. Notable design factors are identified and used to evaluate the FPGA-based communication architectures. The contributions of this paper are as follows:

1. Providing an up-to-date survey on FPGA-based communication architectures. For example, FPGA-based network-on-chip and run-time reconfiguration bus architectures will be discussed.

2. Identifying notable design factors, such as quantitative performance metrics and qualitative design fac-

*Terrence Mak gratefully acknowledges the financial support provided by scholarship from The Croucher Foundation

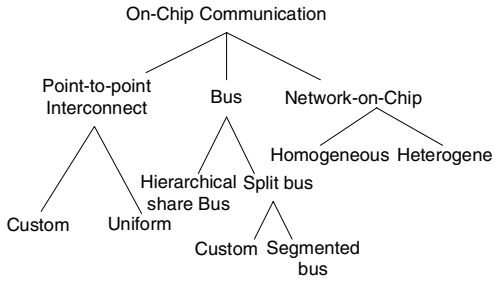

Fig. 1. Taxonomy of on-chip communication architectures

tors, and examining different FPGA-based communication architectures based on the design factors.

Note that on-FPGA communication architectures may be referred as physical-level interconnect architectures in some of the literature. In this paper, on-FPGA communication architectures refers to the FPGA-based architecture, which is built on top of the programmable interconnect fabrics. Programmable fabrics and interconnect architecture designs are beyond the scope of this survey.

In section 2, a taxonomy of on-chip communication architectures will be presented. Section 3 is devoted to the description of the most notable FPGA-based communication architectures. In section 4, seven design factors are presented and they will be used to analyze and classify the surveyed architectures. In section 5, the conclusion of the survey will be presented.

2. ON-CHIP COMMUNICATION ARCHITECTURE TAXONOMY

A taxonomy of on-chip communication architectures is shown in Fig. 1. In general, communication architectures can be categorized into three main classes. An architecture defines the structure of interconnection between processing elements, as well as protocols and interface design.

2.1. Point-to-Point Interconnect

In a point-to-point interconnect architecture, pairs of processing units communicate directly over dedicated physically wired connections Because of its simplicity, point-to-point interconnect has been widely adopted in many applications. Custom interconnect, sometimes referred as ad-hoc interconnect, is simply connecting processing elements by wires when there is a necessity. On the other hand, uniform interconnect often has well defined interconnect topology, which can be precisely specified by equations or graphs. Typical examples are systolic arrays and neural networks.

Simplicity is the major advantage of the point-to-point interconnect architecture. A communication channel between processing elements is simply a set of wires. As the channels are not shared by other processing elements, the latency and performance is deterministic. However, the most significant drawback is that the number of wires required grows rapidly as the number of channels increases. Routing of wires may become intractably difficult [7, 8]. Moreover, a point-to-point scheme suffers from low wire utilization for low bandwidth channels, and a high hardware overhead, as dedicated interfaces for each channel are required.

2.2. Bus Architectures

For bus architectures, long wires are grouped together to form a single physical communication channel, which is shared among different logical channels. An arbitration mechanism is used to control sharing of the bus. This architecture significantly reduces the total length of wires required and also reduces hardware area necessary for interfaces, communication and control. In addition, buses provides a generic infrastructure backbone for interconnection between processing elements.

A hierarchical shared bus defines a segmented bus architecture. Bus segments are connected via a bridge, which may buffer data. Protocols and structures can be varied in different bus segments and each segment may be dedicated to specific functions, such as providing high-performance or low-power communications. The partitioning can further allow optimization of local bus architecture and the communication performance. For example, AMBA (Advanced Micro-Controller Bus Architecture) is a bus architecture developed by ARM which is designed for efficient realization of complex system-on-a-chip (SoC) design [9]. Two system segmented buses are specified in AMBA: the Advanced High-performance Bus (AHB) is for high-performance communication and Advanced Peripheral Bus (APB) is for low-power application purposes. The overall bandwidth can be further improved by using multi-layer interconnect, or crossbar switch [10], in which multiple parallel channels are available to support parallel data transmission.

In a hierarchical bus architecture there is restricted scope to modify or customize the architecture structure and protocol. An alternative approach is to use a split-bus. This refers to a set of custom-design segmented buses. Bus segments can be interconnected in an ad-hoc manner, or based on a systematic approach. In addition, several emerging split-bus bus architectures, which provide multiple segments with adaptive topologies, have been proposed [11, 12, 13]. Enhancements in bandwidth and flexibility were reported.

Despite the advantages, shared bus architectures suffer from power and performance scalability limitations. Long bus wires are increasingly unfavorable in nanometer process technologies. In addition, handling high complexity design

with more IP cores, partitioning and distribution of cores to bus segments is often an ad-hoc task.

2.3. Network-on-Chip (NoC)

The Network-on-Chip (NoC) is an architecture inspired by data communication networks, such as LANs and WANs, with inter-processor communication supported by a packet-switched network [14, 15]. The basic concept of NoC is to communicate across the chip in the same way that messages are transmitted over the Internet today. Communication is achieved by sending message packets between blocks using an on-chip packet-switched network. It is believed the contributions and architectures from the Internet could be borrowed and used to resolve the on-chip communication challenges. It has been reported that the NoC architecture can overcome the long wire disadvantages from bus architectures, as on-chip switches are connected in a regular topology with point-to-point basis. Long wires can be eliminated from the architecture. Also, the architecture is decoupled into transaction and physical layers. Thus the layered architecture enables independent optimization on both sides.

There are two types of NoC: homogeneous and heterogeneous. This refers to whether the network topology is arranged in a regular network or an ad-hoc interconnection network. Different topological arrangements of switches and processing elements result in different performance. A typical example of a homogeneous network uses a mesh-based architecture [16, 17]. A similar structure was also adopted in other implementations [18, 19]. One concern is that functional components are usually of different sizes. Mapping these into regular blocks of fixed size can introduce area wastage or larger chip sizes. In [20, 21], a heterogeneous network topology was proposed. It was reported that the heterogeneous architecture was more area efficient.

3. ON-FPGA COMMUNICATION ARCHITECTURES

FPGA architectures provide a high density of pre-fabricated wires and programmable switches that can be used to realize different communication architectures. Note that the on-FPGA communication architectures discussed in this section inherit only the logical characteristics of the communication architectures in the previous section. Research into modifications to the programmable interconnect fabric of FPGAs is out of the scope of this survey.

3.1. Point-to-Point Interconnect

An early FPGA-based systolic array architecture, proposed by Dick [22], provides an example of this architecture with a structural and systematic topology. Processing elements are connected in a well defined structure with data being pumping through. In [22], the Discrete Fourier Transform is mapped to a systolic array of processing elements. The well-defined interconnection between the processing elements enables effective parallel computation and control. The design was implemented with a Xilinx XC4010 FPGA running at a clock frequency of 15.3 MHz. By using more recent FPGA the clock frequency of the design could be improved. In [23] the systolic array design for gene sequencing, running at a clock frequency of 202 MHz, was reported.

In [24], Shannon and Chow proposed a point-to-point interconnection architecture for rapid system development. The communication between processing elements is achieved by System Integrating Module with Predefined Physical Links (SIMPPL). Several modules are connected on a point-to-point basis to form a generic computing system. The mechanism for the physical transfer of data across a link is provided so that the designer can focus on the meaning of the data transfers, rather than how to connect the wires. The SIMPPL model greatly facilitates the speed and ease of hardware development. It was reported that hardware systems can be developed in a few hours time. In addition, asynchronous FIFOs were implemented as the interface between programmable switch and processing element, such that the communication architecture is more adaptable to IP cores with different operation frequency. The proposed methodology was exemplified with multimedia applications of video streaming and video camera snap shot. The design was implemented with Xilinx Virtex-II FPGA and the SIMPPL control was able to run at 50 MHz.

3.2. Island-Based Architectures

In [25], Cong et. al. proposed a regular distributed register microarchitecture, which offers regular point-to-point communication between islands of computation resources. Each island contains a Finite State Machine, register memory and arithmetic units. Algorithms or arithmetic operations can be partitioned and mapped to the islands, which are then directly connected using wires. More importantly, the proposed architecture provides direct support for multicycle communication. Since propagating and synchronizing a single clock signal to the whole chip is predicted to be a difficult challenge in future technology, multicycle communication takes into consideration the propagation delay of signal wires. A prototype of the architecture has been realized using an Altera Stratix FPGA. A 44% improvement on clock-period was reported for data flow intensive examples and a 28% improvement on clock-period for control flow examples. One concern of the multicycle architecture is the requirement of extra wiring to handle many simultaneous data transfers. The authors of [25] also suggested interconnect pipelining and sharing may be effective in reducing the wire overhead.

In [26], Royal and Cheung proposed an island-based

Globally Asynchronous Locally Synchronous (GALS) architecture. FPGA programmable fabrics are partitioned into islands, each of which is synchronised using a local clock. The global clock signal can be removed, thus avoiding the risk of clock skew and reducing clock power dissipation. However, communication between multiple clock islands may result in metastability, which may cause a functional failure of the system. To avoid metastability, a pausible-clock scheme is employed. A pausible-clock wrapper is introduced to each of the islands and the clock of the corresponding island can be stopped and restarted, so as to ensure data registering without metastability. The FPGA-based GALS architecture is effective in avoiding the clock skew problem and can potentially obtain higher computational speeds. More importantly, different coarse-grain modules can be integrated easily without the concerns of clocking problems in the GALS framework. In other words, the island-based GALS architecture provides high composibility for Platform-FPGAs [27].

3.3. Hierarchical Shared Bus

The CoreConnect Bus architecture [28], developed by IBM, has been adopted by Xilinx for both hard-core and soft-core processors in their Virtex-II Pro and Virtex-4 FPGAs. There is a close resemblance between CoreConnect and AMBA. To enhance the bus performance, CoreConnect defines two distinct bus segments, the Processor Local Bus (PLB) and On-chip Peripheral bus (OPB) which are connected through a bridge. The PLB is a high-speed and high-bandwidth bus. Usually, processors, on-chip RAM and other high bandwidth devices are attached to the PLB, such that it provides high performance communication between the processors. OPB is a low performance and low-power peripheral bus. It is used for asynchronous interfaces and general purpose peripheral components. As an alternative, a bus architecture developed by Altera [29], namely Avalon, provides a more simple solution for system component integration on reconfigurable logic devices.

In [30], IBM CoreConnect was used as the communication backbone between the embedded MicroBlaze soft-processor and custom FPGA-based hardware logic in a cryptography application. The bus architecture is regarded as the only communication channel between the processing elements. The system achieved 35-56 MHz operation frequency.

3.4. Bus Architectures with Run-Time Reconfiguration

In [31], Sedcole et. al. proposed a structured methodology for rapid development of FPGA-based systems. Systems comprise two types of buses: a global bus providing communication between all processing elements and a chain bus providing communication between adjacent processing elements. Extra logic is required for the arbitration of the global bus. Communication and computation tasks were separated by implementing a router at each processing element. The separation of communication and computation enhances the possibility of system architecture reuse for other applications. The computation modules can be reconfigured at run-time, so as to enhance the flexibility of system integration at a late stage of a system development cycle. The proposed methodology was exemplified with an architectural instance of Sonic-on-a-Chip and was realized using Xilinx Virtex-II Pro FPGA. It was reported that the system can be executed at 50MHz and with a throughput of 59.3 Mega-sample per second at its peak performance state.

In [32], a linear architecture for a System-on-Programmable-Chip was proposed. The system is initialized with an AMBA bus backbone and is physically floorplanned to be one dimensional. Processing elements can be attached to and removed from the AMBA bus at run-time. The 1-D placement has the advantages of predictable structure. Moreover, the modules in this approach are not required to be fixed in size. The design consists of run-time configurable Bridges, which can be use as bridging between modules. Further, it was found that the aspect ratio (ratio of the length and width of modules) has great impact on the operation frequency of the system.

3.5. GALS-Based Segmented Bus

In [11], Seceleanu suggested a segmented bus platform supporting communication using multiple clock frequencies. The bus was partitioned into segments of different operating frequency. A synchronizer was introduced into each bus segment. A central arbiter monitors the sharing of global buses. The proposed architecture was realized using an Altera FPGA. A high system frequency can be achieved: 124 MHz was reported. In addition, the number of clock cycles required was reduced from 3000 to 2880 at their experiment on data item transmission between processing elements.

3.6. Reconfigurable Crossbar Switch

In [33], a crossbar switch architecture was proposed to interconnect multiple processing elements. The proposed crossbar switch architecture incorporates an additional bus which can be scheduled to provide extra bandwidth dynamically. This hybrid architecture scheme is motivated by the fact that IP cores implement different functions with corresponding differences in required operating frequency and bandwidth. The additional bus can compensate the extra bandwidth requirement of specific IP cores. The authors of [33] showed that the architecture can provide the same bandwidth with 18% lowered operation frequency.

3.7. Adaptive Network-on-Chip

In line with the recently proposed Network-on-Chip (NoC) architecture, several FPGA-based NoC implementations have been reported [34, 35, 36, 37]. In [34], a dynamic network on chip (DyNoC) for coarse-grain programmable fabrics was proposed. It is an extension of an 1-D shared bus architecture to a 2-D network interconnect architecture. The authors pointed out that with dynamic reconfiguration, a newly reconfigured module may introduce an obstacle to block the packet transmission. A new routing algorithm was proposed in so that that the network-on-chip architecture can be realized with run-time reconfiguration. It should be noted that on-chip switches consume a large area of the design. For a Xilinx Virtex-II 1000 device, from 21% and 46% of logic resources were devoted to the on-chip switches of wordlength 32-bit and 64-bit respectively. The operation frequency of the system was around 70-77 MHz.

A topology adaptable communication network design was proposed in [35]. Network generation is supported by an operating system. The design of switches and routers are critical to the overall system performance and area. The paper reported that the number of slices required grows quadratically with the number of input signals. Further, significant hardware resources, including logic slices and memory, are required for solely network-based communication. Typically, for a 9-switch network, around 3000 slices are required, correspondingto around 14% of the overall resources of a Virtex-II Pro FPGA. The resources required is doubled in a 16-switch network. It appears that optimization of the area consumption is important for the effective realization of NoC architecture. Also, it was reported that the network can run at 50MHz with 16-bit wordlength, which results in a data rate of 100 Mega-byte per second.

A multiple layered network transmission protocol, as well as the design of routers and switches, were reported in [36]. However, a significant overhead on hardware slice consumption was reported. The overhead is attributed to the complicated switch design and routing algorithm implementation.

4. DESIGN FACTORS AND DISCUSSION

Communication architecture design is a complex multi-objective optimization problem that can be evaluated based on a number of criteria and evaluation metrics. The fundamental characteristics and classification of on-FPGA communication architectures are summarized in Table 1.

4.1. Quantitative Metrics

There are four important quantitative metrics. (i) The speed performance of the communication architecture can be quantified by the peak throughput, which is the amount of information transmission completed per unit of time. The throughput of the system is affected by the operation frequency and bandwidth of the communication channels. Alternatively, speed can be characterized as latency, which can be formulated as the sum of sender overhead, transport latency and receiver overhead. (ii) Area is an important criterion for circuit design. It can be defined as number of transistors, logics, memory and wire length. Especially, for FPGA devices, number of logic slices are commonly used as a area measurement. (iii) Interconnect utilization is the amount or percentage of time that the wire is carrying information. (iv) Power is the measurement of energy consumption in interconnect wires.

Table 1. Principal characteristics and classifications of on-FPGA communication architectures

Design	Scalability	Compatibility	Reusability	Routing Complexity	Design Space	Scheduling
P2P [21, 22, 23]	Logical	Interface	Ac-hoc	Complex	Ac-hoc/ systematic	Static
Island [22, 23]	Logical/ Physical	Interface /GALS	Ac-hoc	Complex /Modular	Ac-hoc	Static
Hierarchical [25, 34]	Logical	Interface	Protocol	Modular	Ac-hoc	Dynamic
RTR [28, 29]	Logical	Interface	Separation	Modular	Ac-hoc	Dynamic
Segment [8]	Logical/ Physical	GALS	Separation	Modular	Ac-hoc	Dynamic
Crossbar [30]	Logical/ Physical	Interface	Separation	Modular	Systematic	Dynamic
NoC [33, 34, 35, 36]	Logical/ Physical	Interface /GALS	Separation	Modular	Systematic	Static/ Dynamic

P2P: Point-to-point interconnects; RTR: Run-Time Reconfiguration; NoC: Network-on-Chip; GALS: Globally Asynchronous Locally Synchronous

Based on the published results, we have summarized the quantitative performance measurements for different architectures in Table 2. IBM CoreConnect provides a large range of throughput and its maximum throughput significantly outperforms other architectures. Although the operation frequency of the communication architecture is largely a function of the chosen FPGA technology, the frequency of the segmented-bus architecture is moderately higher than other architectures. On the other hand, the area consumption for the NoC architectures is substantial. However, there is marginal improvement on the performance metrics, such as the operation frequency which ranges from 40 to 64 MHz and the throughput which ranges between 39.8 and 310.4 MByte/s.

4.2. Scalability

A *logical scalable* architecture implies that as more processing elements are added to the communication fabrics, the

Table 2. Summary of the quantitative performance measurements for different architectures

Design	FPGA	Format (modules)	Throughput (MByte/s)	Frequency (MHz)	Bit-length (Bit)	Area (Slice)
CoreCnnnect [27]	Xilinx V2 Pro	n.p.	264-1600	50-100	32-128	1073
RTR-bus [30]	Xilinx V2 Pro	5-18	59.3	50	16	n.p.
Segmented-Bus [10]	Altera Quartus	2-4	43.06*	124	2-32	n.p.
RMBoC [33]	Xilinx V2	16	2.1-2.4*	85-96	16	1074-1759
Interconnect Network [18]	Xilinx Virtex	2x2	310.4	40	16	3227
NoC-OS [34]	Xilinx V2 Pro	3x3	100	50	16	5000
RASoC [35]	Altera Q-II	n.p.	n.p.	55.8-64	8, 16	n.p.
LiPaR [36]	Xilinx V2 Pro	3x3	39.84*	33.33	16	3934

*As results are not provided in the paper, these values are derived from equations given in the papers; n.p.: results are not provided in the paper.

communication bandwidth and peripheral parameters should be able to increases proportionally. In architectures such as a shared bus, the peak bandwidth is fixed. When more processing elements are added to the shred-bus, the overall communication performance will decrease accordingly [15]. Alternatively, Network-on-Chip architectures is logically scalable, as the total bandwidth of the network can be increased when new processing elements are added by increasing the number of network switches [15].

Physical scalability refers to communication architectures that are adaptive to technology scaling and new process technologies. Architectures comprised of many long wires will not adapt well to nanometer process technologies. This criterion usually refers to whether the design is tolerant to physical-level problems.

4.3. Compatibility

Pre-fabricated or pre-designed IP cores usually have specific operating conditions, such as maximum operating frequency. To effectively integrate IP cores, the interconnect backbone requires robust and versatile interfaces. Globally Asynchronous Locally Synchronous (GALS) wrappers provide a flexible interface for a wide range of operating frequencies that can greatly enhance the composibility of different IP cores. In contrast, for interconnect, which does not provide a GALS framework, it is necessary to design a specific interface for each of the IP cores. Therefore, a much longer development cycle is required. Point-to-point GALS architectures [26], segmented buses [11] and some of the NoC architectures implement the GALS framework.

4.4. Reusability

Increased productivity is achieved through system architecture and IP cores reuse. System-level timing is predetermined and one can effectively avoid the iterative design and verification cycles usually necessary to achieve timing closure [31]. The reusasibiltiy takes account of architectural modularization, abstraction and the separation of communication and computation. For architectures such as point-to-point interconnect, interface and interconnection topologies are devoted to a specific application. Thus, it is almost impossible to reuse the hardware. Using a structured design approach [31] explicitly exhibits the concern of separation of communication and computation among the surveyed architectures. Further, shared buses and network-on-chip architectures are also highly reusable, simply because their generic modularity and abstraction of communication.

4.5. Routing Complexity

The architectural and system design will be eventually realized into the FPGA physical device. FPGA routing is a critical step, which translates the higher level communication models into lower level physical interconnects by programming switches. As the FPGA physical interconnect is not an unlimited resource, it is intractably difficult to optimize the programmable switches and physical interconnect for a complex architecture. This may cause routing implementation to take long periods and result in unsatisfied routing constraints. It has been advocated that modular design would ease the layout tediousness for both microprocessor design and reconfigurable computing [7, 31]. For the surveyed architectures, shared buses and NoC architectures generally are designed with modularity abstraction. For the island-based architecture, the modularity and routing complexity depends on the granularity of the “islands”.

4.6. Design-Space Exploration

Communication architectures provide a large range of architectural parameters and options for the designer to consider. Due to the unique requirements of different applications, there exists a problem of searching for the optimal architecture from a huge design space. For example, the NoC topology has a significant effect on network latency, throughput, area, fault-tolerance and power consumption. Therefore, the design of NoC topology plays an important role in routing strategy and mapping the cores to the network nodes [38]. Although very little research focused on FPGA-based communication design space exploration is found in the literature, it is considered an important issue when dealing with communication architecture design. Choosing appropriate architectures and parameters is often performed ad-hoc. Especially for hierarchical buses, decisions for mapping tasks

and modules to bus segments is usually based on designer experiences. This ad-hoc design is limited when dealing with large scale problems. In particular, automating part of the communication architecture exploration task can be invaluable in finding the optimal or near optimal settings of design parameters.

4.7. Scheduling

When communication channels are shared by several processing units, arbitration is required to control and avoid conflicts of requests from processing units. Design of arbitration units has a great impact on the overall system performance. There are two types of scheduling: static routing and dynamic routing.

For *static routing*, the overall delay of communication has a great impact on the overall system performance. Static routing usually refers to the assignment of communication channels design time. In other words, the routing will not be changed according to the real-time traffic of the communication network. For example, the routing of wires in the multicycle architecture is defined at synthesis-time. The programmable interconnect switches in FPGA architectures are also a kind of static routing resources. There is an interesting example in network-on-chip scheduling. In [17, 39, 24], instruction memory and a programmable controller are embedded in the switches. Routing and arbitration are programmed and scheduled at compiled time by a software compiler, thus the overall latency of packet transmission is deterministic.

For *dynamic routing*, the routing can be modified based on the real-time traffic information. For example, in [21], the design of routing-table-based shortest path algorithms for minimizing the overall latency is proposed. Although there is an advantage of simplicity for the routing-table method, extra latency is attributed to the routing logic which may introduce an overhead in terms of power and speed to the communication architecture.

Dynamic arbitration mechanisms follow a set of arbitration policies. Typical policies, such as FIFO (First-In-First-Out), priority queue, round robin and simply random roulette-wheel approach are found in most of the existing bus architectures [9, 28]. Further, stochastic arbitration policies were proposed recently [40]. The bus arbitration is formulated as a Markov Decision Process, such that expectated power utilization and efficiency can be optimized from the perspective of stochastic processes.

5. CONCLUSION

On-FPGA communication architectures play a crucial role in determining the performance and energy consumption of platform-FPGAs containing embedded coarse-grain modules. In the survey, seven types of FPGA-based communication architectures are studied. The design of an efficient and reliable communication architecture is a challenging multi-objective optimization problem. To this end, we identified and presented seven notable design factors. The design factors are used to analyze and classify the architectures. The analysis leads to the following conclusions and suggestions.

Firstly, it is difficult to compare different communication architectures. Most of the published results are based on applying the communication architectures to specific applications. The performance of the architecture may be varied among different applications. Different protocols and variations of the architecture parameters can also generate different results. A more systematic approach for evaluating the communication architectures is required.

Secondly, Network-on-Chip (NoC) architectures has been advocated and is believed to be a promising solution for the on-chip communication challenges, especially in the Multi-Processor System-on-Chip (MP-SoC) design [15]. However, only marginal improvement was found for the FPGA implementation of NoCs. Although it is believed that the NoC could have better logical and physical scalability, the NoC architecture may not always be the best architecture for on-FPGA communication because of the significant area overhead.

Thirdly, there is little research on the on-FPGA communication architecture design space exploration. Most of the existing architectural mapping work relies on an ad-hoc approach.

Fourthly, the architectures with Globally Asynchronous Locally Synchronous (GALS) framework demonstrate a robust and versatile interconnect backbone. Innovative FPGA architectures, which provide GALS support, may provide an effective and reliable solution to the complexity and technology scaling challenge.

6. REFERENCES

[1] Xilinx, "Virtex-4 Data Sheets," 2005.

[2] ——, "Virtex-II Pro / Virtex-II Pro X Complete Data Sheet," 2005.

[3] Altera, "Stratix II Device Family Data Sheet," 2005.

[4] M. Beauchamp, K. S. Hemmert, K. Underwood, and S. Hauck, "Architectural Modifications to Improve Floating-Point Unit Efficiency in FPGAs," in *IEEE Symposium on Field-Programmable Custom Computing Machines*, 2006.

[5] K. Bondalapati and V. K. Prasanna, "Reconfigurable Computing Systems," *Proceedings of the IEEE*, vol. 90, no. 7, 2002.

[6] R. Ho, K. Mai, and M. Horowitz, "The Future of Wires," *Proceedings of the IEEE*, vol. 89, no. 4, pp. 490 – 504, 2001.

[7] J. Cong, "An Interconnect-Centric Design Flow for Nanometer Technologies," *Proceedings of the IEEE*, pp. 505–528, 2001.

[8] M. J. Alexander and G. Robins, "New Performance-Driven FPGA Routing Algorithms," in *Design Automation Conference*, 1995.

[9] ARM, "AMBA Specification v2.0," Tech. Rep., 1999.

[10] ——, "AMBA Multi-layer AHB overview," Tech. Rep., 2001.

[11] T. Seceleanu, "Communication on a Segmented Bus Platform," in *IEEE International SOC Conference*, 2004.

[12] K. K. Ryu and V. J. Mooney, "Automatic Bus Generation for Multiprocessor SoC Design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 11, pp. 1531–1549, 2004.

[13] K. K. Ryu, E. Shin, and V. J. Mooney, "A Comparison of Five Different Multiprocessor SoC Bus Architectures," in *Euromicro Symposium on Digital Systems Design*, 2001.

[14] W. Dally and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks," in *Design Automation Conference*, 2001.

[15] L. Benini and D. Bertozzi, "Network-on-chip Architectures and Design Methods," *IEE Proc.-Comput. Digit. Tech.*, vol. 152, pp. 261–272, 2005.

[16] S. Kumar, A. Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. berg, K. Tiensyrj, and A. Hemani, "A Network on Chip Architecture and Design Methodology," in *International Symposium in VLSI*, 2002.

[17] M. B. Taylor, "The Raw Microprocessor: A Computational Fabric For Software Circuits and General-Purpose Programs," *IEEE Micro*, 2002.

[18] R. Soares, I. S. silva, and A. Azevedo, "When Reconfigurable Architecture Meets Network-on-Chip," in *17th Symposium on Integrated Circuits and Systems Design*, 2004.

[19] T. Marescaux, A. Bartic, D. Verkest, S. Vernalde, and R. Lauwereins, "Interconnection Networks Enable Fine-Grain Dynamic Multi-Tasking on FPGA," in *Field Programmable Logic and Applications*, 2002.

[20] D. Bertozzi and L. Benini, "Xpipes: A Network-on-Chip Architecture for Gigascale Systems-on-Chip," *IEEE Circuits and Systems Magazine*, pp. 18–31, Second Quarter 2004.

[21] D. Bertozzi, A. Jalabert, S. Murali, R. Tamhankar, S. Stergiou, L. Benini, and G. D. Micheli, "NoC Synthesis Flow for Customized Domain Specific Multiprocessor Systems-on-Chip ," *IEEE Tranc on Parallel and Distributed Systems*, vol. 16, no. 2, pp. 113–129, 2005.

[22] C. Dick, "Computing the Discrete Fourier Transform on FPGA Based Systolic Arrays," in *ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 1996.

[23] C. Yu, K. Kwong, K. Lee, and P. Leong, "A Smith-Waterman Systolic Cell," in *Field Programmable Logic and Applications*, Lisbon, 2003.

[24] L. Shannon and P. Chow, "Simplifying the Integration of Processing Elements in Computing Systems using a Programmable Controller," in *IEEE Symposium on Field-Programmable Custom Computing Machines*, 2005.

[25] J. Cong, Y. Fan, G. Han, X. Yang, and Z. Zhang, "Architecture and Synthesis for On-Chip Multicycle Communication," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 4, pp. 550–564, 2005.

[26] A. Royal and P. Cheung, "Globally Asynchronous Locally Synchronous FPGA Architectures," in *Field Programmable Logic and Applications*, 2003.

[27] P. Molina, "The Design of a Delay-Insensitive Bus Architecture Using Handshake Circuits," Ph.D. dissertation, Imperial College of London, 1997.

[28] IBM, "CoreConnect(TM) Bus Architectre," *White paper*, 1999.

[29] Altera, "Avalon Bus Specification Reference Manual," 2003.

[30] R. Cheung, W. Luk, and P. Cheung, "Reconfigurable Elliptic Curve Crytosystems on a Chip," in *Design, Automation and Test in Europe Conference and Exhibition*, 2005.

[31] P. Sedcole, P. Y. K. Cheung, G. Constantinides, and W. Luk, "A Structured Methodology for System-on-an-FPGA Design," in *Field Programmable Logic and Applications*, 2004.

[32] H. Kalte, M. Porrmann, and U. Ruchert, "System-on-Programmable-Chip Approach Enabling Online Fine-Grained 1D-Placement," in *International Parallel and Distributed Processing Symposium*, 2004.

[33] D. Kim, K. Lee, S.-J. Lee, and H.-J. Yoo, "A Reconfigurable Crossbar Switch with Adaptive Bandwidth Control for Networks-on-Chip," 2005.

[34] A. Ahmadinia, C. Bobda, J. Ding, M. Majer, and J. Teich, "A Practical Approach for Circuit Routing on Dynamic Reconfigurable Devices," in *IEEE International Workshop on Rapid System Prototyping*, 2005.

[35] T. Bartic, J.-Y. Mignolet, V. Nollet, T. Marescaux, D. Verkest, S. Vernalde, and R. Lauwereins, "Topology Adaptive Network-on-Chip Design and Implementation," *IEE Proc.-Comput. Digit. Tech*, vol. 152, no. 4, 2005.

[36] C. Zeferino, M. E. Kreutz, and A. A. susin, "RASoC: A Router Soft-Core for Networks-on-Chip," in *Design, Automation and Test in Europe*, 2004.

[37] B. Sethuraman, P. Bhattacharya, K. Khan, and R. Vemuri, "LiPaR: A Light-Weight Parallel Router for FPGA-based Network-on-Chip," in *GLSVLSI*, 2005.

[38] U. Ogras, J.-C. Hu, and R. Marculescu, "Communication-Centric SoC Design for Nanoscale Domain," in *International Conference on Application-specific Systems, Architectures and Processors*, 2005.

[39] J. Liang, A. Laffely, S. Srinivasan, and R. Tessier, "An Architecture and Compiler for Scalable On-Chip Communication," *IEEE Transactions on VLSI Systems*, vol. 12, no. 7, pp. 711–726, 2004.

[40] N. Thepayasuwan, S. Kallakuri, A. Doboli, and S. Doboli, "Communication Subsystem Synthesis and Analysis Tool Using Bus Architecture Generation and Stochastic Arbitration Policies," in *International Symposium on Circuits and Systems*, 2005.