Communication Architectures for Dynamically Reconfigurable FPGA Designs

Thilo Pionteck[1], Carsten Albrecht[1], Roman Koch[1], Erik Maehle[1]
Michael Hübner[2], Jürgen Becker[2]

[1]University of Lübeck
Institute of Computer Engineering
23538 Lübeck, Germany
{pionteck, albrecht}@iti.uni-luebeck.de
{koch, maehle}@iti.uni-luebeck.de

[2]Universität Karlsruhe (TH)
Institut für Technik der
Informationsverarbeitung
76128 Karlsruhe, Germany
{huebner, becker}@itiv.uni-karlsruhe.de

Abstract

This paper gives a survey of communication architectures which allow for dynamically exchangeable hardware modules. Four different architectures are compared in terms of reconfiguration capabilities, performance, flexibility and hardware requirements. A set of parameters for the classification of the different communication architectures is presented and the pro and cons of each architecture are elaborated. The analysis takes a minimal communication system for connecting four hardware modules as a common basis for the comparison of the diverse data given in the papers on the different architectures.

1. Introduction

With the emergence of partially and dynamically reconfigurable FPGAs, System-on-Chip (SoC) architects are given a new degree of freedom in system level design. While conventional SoCs require the number, types and locations of hardware modules to be defined at design time, dynamically reconfigurable FPGAs allow these parameters to be adapted at runtime. This new degree of freedom, however, also raises new issues that have to be solved. Among these are the handling of the actual process of dynamic reconfiguration, problems related to online placement of hardware modules, and the question of suitable communication structures. There has been notable research effort addressing the first two issues while the interconnect between dynamically exchangeable hardware modules is the subject of only a few works. Nonetheless, the design of the communication network has a huge impact on total system performance. In a reconfigurable system different combinations of hardware modules may impose different demands on bandwidth and latency of the communication network. It is therefore advisable to apply dynamic reconfiguration not only to hardware modules, but to utilize this technique in order to adapt the communication resources to the actual communication pattern as well.

The present work gives an overview on adaptable communication architectures for dynamically reconfigurable FPGA designs. Compared to other works like [11, 13], this paper focuses exclusively on architectures which provide support for dynamic exchange of hardware modules and topology adaptation at runtime. In total four architectures are analyzed in terms of efficiency, hardware requirements and flexibility. The analysis is done with regard to the design and performance parameters defined within this work. A minimal communication system for connecting four hardware modules is assumed, so that a better comparison of the diverse data given in the papers on the different architectures could be achieved. Due to insufficient data for some architectures the technical details of the actual reconfiguration process are not covered in this paper.

The rest of this paper is organized as follows: Section 2 provides a brief overview on fundamental communication structures used in FPGA based designs and introduces the taxonomy to be used later in the discussion. Runtime adaptable communication architectures for dynamically reconfigurable FPGA designs are presented in Section 3, followed by a discussion and comparison in Section 4. Concluding remarks are given in Section 5.

2. Classification

Typical SoC designs for FPGAs usually utilize either bus-based or Network-on-Chip (NoC) based communica-

1-4244-0910-1/07/$20.00

tion schemes. The advantages and disadvantages inherent to these communication schemes are briefly reviewed in this section. To begin with, a set of parameters for the classification of the different communication schemes is given.

2.1. Parameters

For communication networks, the main performance parameters are latency and bandwidth. Performance numbers may relate either to real time or depend on the *maximum clock rate* of a synchronous system. For the purpose of this work, latency and bandwidth are defined as follows:

Latency $l_i \in \mathbb{N}_0$ of a network element i is the number of clock cycles by which a data transfer is delayed when passing through the network element. The *path latency* $l_P \in \mathbb{N}_0$ is the the number of clock cycles by which a data transfer is delayed by n network elements along the path from source to destination: $l_P = \sum_i^n l_i$.

Bandwidth $b_L \in \mathbb{R}_0^+$ is the amount of data a network link L can physically deliver per time unit.

As *bandwidth* depends on the clock frequency and bitwidth of a link, these basic parameters are mainly considered in the following. Besides latency and bandwidth, throughput is also commonly used for characterizing the performance of communication architectures:

Throughput $t \in \mathbb{R}_0^+$ of a networking is defined by $t = \sum_{i=1}^k b_{L_i}$, k being the number of links and b_{L_i} being the bandwidth of link L_i.

Throughput, however, is not an appropriate measure for the communication architectures considered in this paper. As the topology of these communication networks and thus the number of links may change during runtime, a fixed throughput value cannot be given. Rather than throughput the following measure will be referred to:

Parallelism $d_{max} \in \mathbb{N}$ supported by a network is the maximum number of independent data transfers that can occur simultaneously.

The parallelism of a communication network depends on the number of links k, the number of nodes n, and on the topology. A maximum parallelism of d_{max}does not guarantee that d_{max} data transfers can take place between *arbitrary* pairs of source and destination.

Besides the performance related parameters of networks, structural parameters play an important role when chosing an appropriate communication architecture for a given application. The main design parameters are: flexibility, scalability, extensibility and modularity.

Flexibility is the ability of a communication structure to support different communication patterns in a fixed design without loss of performance.

Scalability describes the ability of a communication structure to be modified to provide a fixed set of performance parameters independently from the system size and characteristics.

Normally, the term scalability would only refer to the design time of a system. As for dynamical reconfigurable FPGA designs structural modifications are not limited to design time, we extend the definition of scalability also to the runtime of a system.

Extensibility is the ability of an architecture to be extended to larger system designs.

Extensibility is closely related to scalability, but does not require the extended system to maintain the performance of the unextended system. Throughout this work only runtime extensibility is considered, i.e. the extension is done by means of dynamic reconfiguration.

Modularity The *modularity* of a communication architecture is defined by the extent up to which it can be divided into submodules.

The more an architecture can be decomposed into submodules, the easier is an adaptation of the architecture to a specific realization. In this context the term *reuseability* comes up as a modular architecture is one prerequisite to a successful design reuse.

2.2. Basic Communication Schemes

The most commonly used communication structures are buses. Their main advantages are their flexibility, extensibility and their low design costs. In general, buses show a low latency when the bandwidth demands are low. They are well suited for communication patterns where most of the communication takes place between two components, for instance between a processor and memory. As soon as the communication pattern involves more than two modules, the main disadvantage of buses become apparent: Their scalability is poor. Since all traffic shares a single path, time-overlapping bus requests have to be serialized. This leads to a sharing of the effective bandwidth as the number of components increases. One way to deal with this issue is using hierarchical or split buses. The AMBA [1] and CoreConnect [3] bus architectures are examples of hierarchical bus systems. They consist of a low-speed peripheral bus connected to a high-speed system bus through a bridge. There are also extensions where the buses are split into segments, thus allowing locality of communication to

be exploited. These extensions increase the scalability of buses at the cost of flexibility, as bridges may lead to bottlenecks between hardware modules on separated buses. Another disadvantage of buses are their long communication lines. In FPGA designs, those lines are costly to route, and in general lead to huge power consumption and a limited system clock speed.

The main advantage of NoCs is their ability to support concurrent communication among hardware modules. Compared to buses, NoCs show also a very good scalability. As, in general, for each new hardware module a new switch has to be added, the number of links and thus the possible number of parallel data transfers increases. NoCs can be designed to support much larger bandwidths than buses, yet they show a higher latency due to their multi-hop topology. NoCs are well suited for communication patterns where several modules communicate with each other in parallel. The modularity of NoCs eases design reuse and provides a good decoupling of the communication fabric from the processing elements. As a result of the segmented structure of NoCs, only local wires need to be used, resulting in less power consumption and higher clock frequencies compared to bus-based architectures. The main disadvantage of NoCs is the area overhead resulting from the existence of switches. Each switch comprises buffers, routing logic and arbitration logic, leading to a significant area overhead compared to buses which employ a central control logic.

3. Architectures

In this section, four communication architectures supporting the dynamic exchange of hardware modules are presented. The first two architectures, RMBoC [4, 5] (Reconfigurable Multiple Bus on Chip) and BUS-COM [9] base on bus-oriented communication schemes. DyNoC [5] (Dynamic Network on Chip), and CoNoChi [12] (Configurable Network on Chip) use a NoC-based approach. The design and implementation parameters for the architectures are summarized in table 1 and table 2, respectively. Note that for bus-based architectures the implementation parameters refer to the complete architecture while for NoC-based designs the implementation parameters refer to one switch.

3.1. Bus-based Systems

Bus-based communication architectures were the first ones developed for dynamically reconfigurable FPGA designs. They use a slot-based approach, i.e. only one hardware module can be configured in one column of the FPGA. This restriction was mainly motivated by the column-based configuration architecture of the Xilinx Virtex-II FPGA and the limited software support for dynamic reconfiguration.

RMBoC: Reconfigurable Multiple Bus on Chip

RMBoC [4, 5] bases on a modified version of Reconfigurable Multiple Bus (RMB) Networks [7] proposed for multi-processor systems. The RMBoC bus system consists of a so-called cross-point per processing element and multiple buses. The k buses are segmented in short parts s that link neighbouring cross-points. Communication channels between arbitrary processing elements are established by sending a request message to the destination via the bus system. A cross-point receiving the request tries to find a free bus in the next segment. If the destination is reached and the receiver accepts the connection a reply message is sent via the established channel. In case of completely occupied bus segments or the refusal of a connection request by the destination a cancel message is sent along the channel to release the reserved bus segments. An established channel is released after usage by sending a destroy message. So, the protocol is rather simple and demands the system application to deal fairly with the resources. RMBoC shows similarities to a circuit switched NoC. As RMBoC bases on the RMB architecture mentioned before, it is classified as a bus-based architecture.

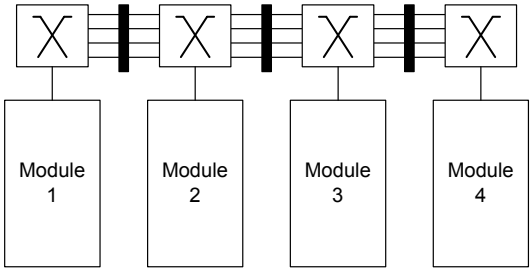

Figure 1. RMBoC architecture [4, 5]

The time to establish a connection from source to destination depends on the number of processing elements m and has an upper bound of $4 \cdot \left\lceil \frac{m^2+2m+4}{2} \right\rceil$ cycles. Further on, data is transferred within a single clock cycle. Table 2 gives both values for a scenario of four modules and four buses. A minimum of 8 clock cycles is required to set up a connection and after this data transfer can be done in one clock cycle.

An example of an RMBoC system with $k = 4$ parallel buses for the support of $m = 4$ exchangeable hardware modules of a fixed size is given in figure 1. The implementation parameters shown in tables 1 and 2 base on such a system prototyped on a Xilinx Virtex-II 6000 FPGA in [4]. For a proof of concept a small video application was used.

Depending on the bus width, a maximum clock frequency of about $100\ MHz \pm 6\%$ was reached with an area overhead in range between 4% and 15% of the FPGA area.

Architecture	Type	Topology	Module Size	Switching	Bit width	Overhead	max. Payload	Protocol Layers
RMBoC[4]	Bus	1D-Array	fixed	circuit	$1-32$	control msg.	*circuit switched*	1
BUS-COM[9]	Bus	1D-Array	fixed	time mult.	arbitrary	20 bit	256 $byte$	1
DyNoC[5, 6]	NoC	2D-Array	variable	packet	$8-32$	≥ 4 bit	n. p.	1
CoNoChi[12]	NoC	2D-Array	variable	packet	$8-32$	96 bit	1024 $bytes$	3

Table 1. Design Parameters

Architecture	Latency [cycles]	Frequency [MHz]	Area [Slices]	Device
RMBoC[4] ($c=4, m=4, \rightleftarrows 32bit$)	initial phase: 8 establ. connection: 1	94	5084	Virtex-II
BUS-COM[9] ($c=4, m=4, \leftarrow 32bit, \rightarrow 16bit$)	1	66	294	Virtex-II
DyNoC[5, 6] (Switch, $\rightleftarrows 32bit$)	1	75	370	Virtex-II
CoNoChi[12] (Switch, $\rightleftarrows 32bit$)	5	73	410	Virtex-II

c: number of buses, m: number of modules, $\rightleftarrows$: bidirectional bit width, $\leftarrow$: input bit width, $\rightarrow$: output bit width

Table 2. Implementation Parameters

Unfortunately, the behaviour of RMBoC during the reconfiguration of a slot at run time is not described. But, because of placing the cross-points in the columns of a slot, it is expected that cross-points are freezed during the reconfiguration process so that only established channels can be used. Note that any reconfiguration of the system does not alter RMBoC modules or its physical topology. The alteration of the communication infrastructure on application level is performed on the level of an overlay network with point-to-point channels.

BUS-COM

The second bus-based architecture is BUS-COM [9]. Contrary to RMBoC, this architectures uses unsegmented buses which are assigned to hardware modules by an arbiter. The hardware modules are connected with the bus system via interface modules called BUS-COM. All modules are physically connected to all buses, yet virtual network topologies can be formed by manipulating the time-slot based bus arbitration mechanism based on the FlexRay protocol [2]. For each bus 32 time slots exist. Each time slots is exclusively assigned to a certain BUS-COM module during design time. By means of dynamic reconfiguration the assignment can be changed. So the distribution of network resources and the virtual network topology can be adapted to the communication patterns of the hardware modules. The time slots for each bus are divided into static and dynamic slots. Static slots provide hardware modules guaranteed bus access time to send and receive data of a fixed payload size. Dynamic slots provide hardware modules additional bus access time according to their priority. The payload size for dynamic slots is variable but limited to 256 byte.

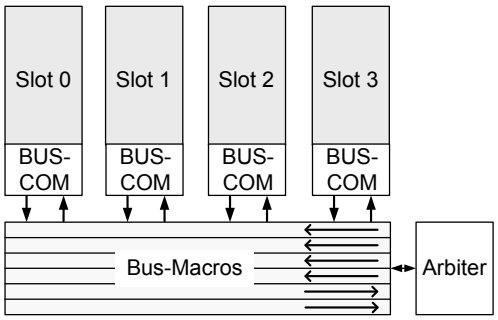

Figure 2. BUS-COM architecture [8]

In [8] a bus system for connecting four hardware modules as depicted in figure 2 was presented. The system consists of four buses with an input bit width of 32 bit and an output bit width of 16 bit. The system was designed for the usage in real time automotive applications and provides different functions for inner cabin applications on-demand. A Xilinx Virtex-II 3000 FPGA was used as hardware platform. The buses were realized using special bus macros. Each macro is able to transport 8 bits of data unidirectional, resulting in six macros per bus. Each macro uses 20 slices. Additional area overhead is introduced by the arbiter, and

the input- and output macros for one slot, so that in total 296 slices are required for the presented system. The maximum clock frequency of the BUS-COM architecture is 66 MHz.

In the presented version of BUS-COM only one hardware module can be configured into one slot. There exist an extended version where the height of a hardware module is arbitrary and thus multiple modules can be placed into one slot. The connection of hardware modules to the bus system is done online in separate slots. The applied techniques are currently refined so that arbitrary communication channels can be routed during runtime. As this work is in progress right now it will not considered for the rest of the paper.

3.2. Network-on-Chips

Network-on-Chip architectures provide a higher degree of freedom for the online placement of hardware modules. The placement is not restricted to slots of a fixed size anymore, instead an arbitrary 2D positioning of components of rectangular size is possible.

DyNoC: Dynamic Network on Chip

The first presented architecture utilizing a packet-based NoC approach for dynamically reconfigurable designs was DyNoC [5, 6]. DyNoC consists of a two dimensional array of processing elements and routers, in which each processing element is connected to one router. Hardware modules can be mapped onto multiple processing elements. Routers, which are physically between the processing elements used for one module, are removed from the network and can be used as additional hardware resources for the module they belong to. An example of a 5×5 DyNoC system is given in figure 3.

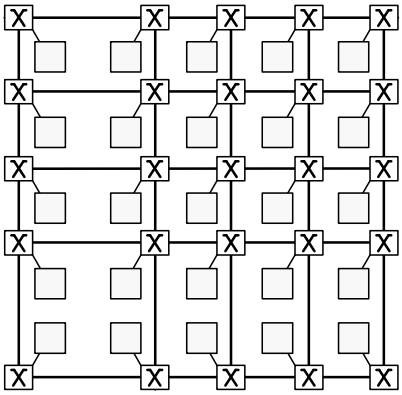

Figure 3. DyNoC architecture [5, 6]

The module placement is arranged so that a module is always completely surrounded by network routers. This guarantees that the network is always connected. Routing decision are made deterministic and locally using the S-XY-routing algorithm [5] which is an enhanced version of the XY-routing scheme. The S-XY-routing algorithms is capable of surrounding obstacles by exploiting the fact that a module is always surrounded by network routers. If a packet is blocked in its given direction by a placed component, the routers surrounding the component are informed in which direction a packet should be sent.

The feasability of the DyNoC architecture was proofed with a 4×4 DyNoC array in [6] and with a 3×3 DyNoC array in [5]. Both prototypes were realized using a Xilinx Virtex-II 6000 FPGA. As test applications, again a small video application utilizing a VGA controller was used.

CoNoChi: Configurable Network on Chip

The CoNoChi architecture [12] comprises virtual cut-through switches with four equal full-duplex links, lines connecting the switches, and modules attached to the network. An $i \times j$ grid of tiles $t_{i,j} \in \{0, S, H, V\}$ forms the basis of CoNoChi: An S type tile contains a switch while H and V type tiles contain horizontal and vertical communication lines, respectively. Modules are implemented in 0 type tiles. These tiles are not used by the infrastructure of the network. The network size and topology can be changed by replacing individual tiles with tiles of a different type. This allows a runtime adaptation of the topology of the network to the number and location of currently configured hardware modules. Depending on the topology, switches can be added or removed from the network by a global control unit without stalling the NoC. An example of CoNoChi is given in figure 4.

CoNoChi employs a three layer protocol header, supporting physical and logical addresses. Routing solely bases on the physical addresses, used for table lookups of locally stored routing information. Logical addresses are evaluated by special interface modules connecting the hardware module with a switch. Like the core modules, the interface components are configured into 0 type tiles. The support of logical addresses allows hardware modules and subordinate processing entities to be moved or combined within the NoC. A prototypical implementation of CoNoChi on a Virtex-II Pro 1000 was presented in [12]. The CoNoChi architecture focuses on streaming applications with high throughput demands such as network applications. An application example is given in [10].

4. Discussion

This section provides a comparison of the presented communication architectures with a focus on the dynamic reconfiguration capabilities. The discussion does not ac-

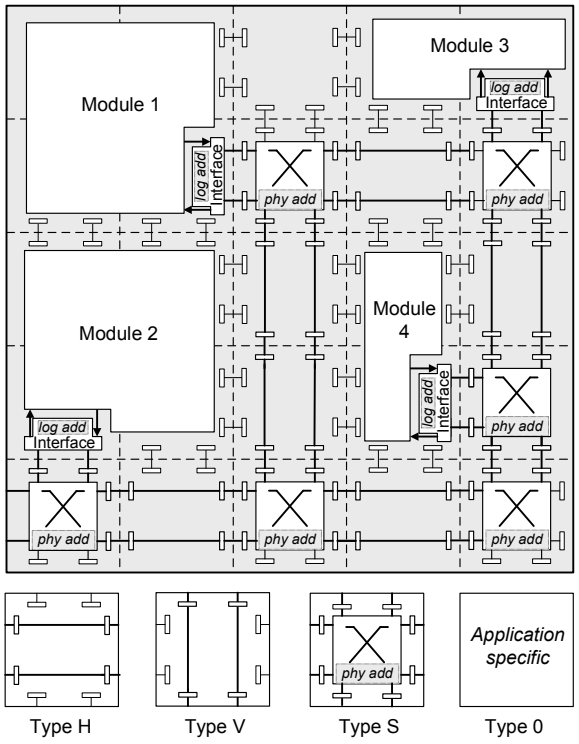

Figure 4. CoNoChi architecture

count the principle advantages and disadvantages of the basic communication schemes. As not for all architectures a hardware description or simulator are available, the comparison solely bases on the data given in the corresponding papers of the architectures.

In total, all architectures provide a sufficient support for dynamically reconfigurable FPGA designs. The NoC-based architectures show the best structural parameters but suffer from their huge area overhead. If area efficiency is the main design parameter, the bus-based systems are the first choice. Especially BUS-COM requires only few hardware resources. A fair comparison between BUS-COM and RMBoC is difficult, though as only RMBoC provides the area requirements of the complete system. Concerning flexibility, RMBoC is superior to BUS-COM due to the segmented buses. CoNoChi offers the best structural parameters and the best conceptional support for dynamic reconfiguration, but suffers from implementation difficulties on the Virtex-II platform. In contrast DyNoC is designed to get by with the limited reconfiguration capabilities of the Virtex-II platform.

4.1. Technical Aspects

Concerning system design, the most important questions are how well does a communication architecture support the communication patterns of runtime exchangeable hardware modules and what kind of restrictions does an architecture impose to the characteristics of hardware modules. Based on these questions, the presented architectures can be divided into two groups. The first group consists of the bus-based architectures RMBoC and BUS-COM. They both restrict the shape of modules to the height and width of the slots in which they can be configured in. The other group DyNoC and CoNoChi, both utilizing a NoC-based approach, support hardware modules of arbitrary rectangular shape. Thus, the NoC-based approaches offer the highest flexibility for the selection of hardware modules. This flexibility, however, does not come without costs.

RMBoC	BUS-COM	DyNoC	CoNoChi
5084	294	1480	1640

Table 3. Estimated minimum number of slices for connecting 4 modules with 32 bit links

The minimal area requirements of the presented interconnection architectures for connecting four modules with a link width of 32 bit are given in table 3. Note that for the comparison several assumptions had to be made. For DyNoC, the assumption is that each hardware module only occupies one processing unit of DyNoC. Thus, for four modules only four switches are required. The area values for CoNoChi and BUS-COM do not include the area requirements of the control units. For CoNoChi, this is the global control unit among others providing the routing tables and controlling the reconfiguration process, for BUS-COM this is the bus arbiter. When scaling the size of the communication network, the area requirements of these modules appear as a kind of offset. The only value which includes all hardware resources needed for operation is the one of the RMBoC architecture. Despite all these restrictions, the values in table 3 show a trend. The flexibility of the NoC-based architectures to support hardware modules of arbitrary rectangular shape comes at the expense of area efficiency. Especially if a larger number of modules have to be connected, the area requirements increase due to the increasing number of switches. The extent to which the area requirements for DyNoC and CoNoChi increases differ. Depending on the topology, for each new hardware module only one new switch is required for CoNoChi, while DyNoC may require several, depending on the size of the modules. Thus, for a larger number of modules and larger module sizes, the area overhead of CoNoChi will be less than for DyNoC. If a fixed module size is acceptable for a system design, the BUS-COM architecture seems to be a good solution, though with the restrictions of bus-based architectures in terms of flexibility and scalability.

Concerning the technical realizations, the most unprob-

lematic architectures are RMBoC and BUS-COM. Their slot based architecture is closely related to the reconfiguration capabilities of the Virtex-2 architecture and the (former) restrictions in the design flow. DyNoC avoids the reconfiguration restrictions by mapping its own reconfigurable architecture, composed of processing elements and router, on top of the FPGA. As pointed out in [12], the realization of CoNoChi on the Virtex-2 platform requires many workarounds. These are mainly caused by the limited reconfiguration capabilities of the architecture and the problem of relocating the content of tiles among each other. Therefore, a full exploitation of the architecture will require a change in the hardware platform to a Virtex-4 FPGA.

4.2. Performance Parameters

The maximum clock frequency of all architectures is in the range between 73 MHz and 94 MHz and thus in the same order of magnitude. Therefore, this parameter is not appropriate for ranking the different architectures. The same refers to the bandwidth b, as the bitwidth and depending on it the bandwidth of each architecture is adaptable at design time. Differences in the architectures can be found in the latency l. Considering an established connection, the lowest latency of $l_P = 1$ is achieved by the bus-based architectures. The path latency for the NoC-based architectures scales with the number of switches. However, as for the area requirements, the extent to which the path latency increases differs for DyNoC and CoNoChi. For CoNoChi, the number of switches in a system only depends on the number of hardware modules while the size of a DyNoC array and thus the number of switches also depends on the size of the hardware modules. For larger modules, the probability that for a communication path more switches have to be passed in DyNoC than in CoNoChi increases.

Another important performance parameter is the maximum achievable parallelism. Normally, bus-based architectures only provide $d_{max} = 1$. The bus systems described here break this limit by providing $k > 1$ buses which are used equivalently. While BUS-COM increases the degree of parallel communication by providing k buses RMBoC goes one step further by segmenting the buses into s segments. So, RMBoC supports a theoretical upper limit of $d_{max} = s \times k$ parallel communications, whereas BUS-COM only supports $d_{max} = k$ channels per time. BUS-COM is also limited in exploiting locality of communication, as the bus is not segmented. The NoCs described here perform packet-switching so that communication channels are not established exclusively. The degree of parallelism in communication is theoretically limited by the number of links but because of their minimal routing strategies links are not equally loaded.

For the efficiency of an architecture, the control overhead is also of importance. Except the CoNoChi protocol, the protocols are solely designed to address and route data packets, so, to establish connections on a minimal base. CoNoChi, however, includes features for reconfiguration such as packet redirection. Considering the protocols three sorts of overhead occur: firstly, an additional amount of data must be transmitted, secondly, the protocol components consume area and, thirdly, time. The packet-switched systems as well as BUS-COM need a header to address the communication partners. So, each data packet is limited in its size and contains control information which reduces the effective bandwidth of BUS-COM and CoNoChi to approximately 90%. Unfortunately, for DyNoC neither the effective bandwidth nor the maximum payload is given. The control overhead of RMBoC consists of two small packets, simplified reply and request, which are not related to a maximum amount of data. Thus, the protocol overhead becomes neglectable here.

4.3. Structural Parameters

The architectures presented in this paper follow different approaches to the problem of inter-module communication in runtime reconfigurable systems. These differences are reflected by the structural parameters introduced in Section 2, namely by *scalability* and *modularity*. Scalability, in turn, necessarily requires *extensibility* while *flexibility* as a parameter referring to a fixed design does not primarily relate to reconfigurability. Yet, a higher degree of flexibility can lower the need for frequent reconfiguration and thus have a positive effect on the performance of a reconfigurable system. The ranking of the architectures concerning the structural parameters is given in table 4.

With regard to the architectures considered in this paper, distributed routing tables and the packet redirection feature make CoNoChi the most flexible architecture for the price of an increased area overhead. It is followed by BUS-COM which provides support for virtually adaptive topologies and allows dynamically assigned time-slots for bus allocation. The structure of RMBoC in principle allows bandwidth adaptation by the use of a variable number of connections between two modules. Finally, DyNoC employs a light-weight routing scheme which does not support variable bandwidth or load adaptation. The Scalability of the presented architectures can be considered the same as for non-reconfigurable systems based on the same communication scheme. NoC-based architectures provide in general good scalability. Bus-based systems are limited in their scalability due to their fixed number of buses and thus in their number of parallel data transfers. Special effects in this general ranking introduced by the dynamic reconfiguration capability cannot be seen. Concerning runtime extensibility, the NoC-based architectures show the best charac-

Architecture	Flexibility	Scalability	Extensibility	Modularity
RMBoC[4]	high	medium	low	medium
BUS-COM[9]	medium	medium	medium	medium
DyNoC[5, 6]	low	high	high	high
CoNoChi[12]	high	high	high	high

Table 4. Characteristics of the communication architectures

teristics. New components can be added at each border of the system. BUS-COM in principle limits the extensibility to only one dimension due to the structure of the bus system, whereas for RMBoC no details about the extensibility of the bus structure are given. The considered approaches are also modular as each respective architecture provides a standard interface for any kind of module. Futhermore, no general restrictions apply to the location of any specific module. In the context of reconfigurable systems the term modularity can be narrowed by taking into account the granularity at which components can be replaced. Both DyNoC and CoNoChi are based on tiled grids and allow modules of arbitrary and varying rectangular size to be configured into the network. In contrast, BUS-COM and RMBoC provide module sites of fixed sizes. Thus, DyNoC and CoNoChi can be considered more modular than BUS-COM an RMBoC.

5. Summary

In this paper an overview was given on current approaches to the module interconnect problem in runtime reconfigurable systems. Focus was set on the specific properties of these interconnection architectures in the context of runtime reconfigurability. Using an unambiguous terminology, four particular architectures were presented and discussed. Albeit some shortcomings in comparable parameters, the fundamental advantages and disadvantages of different paradigms were set out as were the key properties of the respective architectures. Overall, this survey and analysis can serve as a guidance when a decision for one or the other interconnection architecture has to be made.

6. Acknowledgement

This work was funded in part by the German Research Foundation (DFG) within priority programme 1148 under grant reference Ma 1412/5.

References

[1] AMBA 2.0 specification. http://www.arm.com/products/-AMBA.

[2] FlexRay. http://www.flexray.de.

[3] IBM CoreConnect Bus Architecture. http://www.chips.ibm.com/products/coreconnect/.

[4] A. Ahmadinia, C. Bobda, J. Ding, M. Majer, J. Teich, and J. C. Fekete, Sandor P.and van der Veen. A Practical Approach for Circuit Routing on Dynamic Reconfigurable Devices. In *RSP '05: Proc. of the 16th IEEE Int. Workshop on Rapid System Prototyping (RSP'05)*, pages 84–90, Washington, DC, USA, 2005.

[5] C. Bobda and A. Ahmadinia. Dynamic Interconnection of Reconfigurable Modules on Reconfigurable Devices. *IEEE Design & Test of Computers*, 22(5):443–451, 2005.

[6] C. Bobda, A. Ahmadinia, M. Majer, J. Teich, S. P. Fekete, and J. van der Veen. DyNoC: A Dynamic Infrastructure for Communication in Dynamically Reconfigurable Devices. In *Proc. of the Int. Conference on Field-Programmable Logic and its Applications (FPL 2005)*, pages 153–158, 2005.

[7] H. ElGindy, A. K. Somani, H. Schroder, H. Schmeck, and A. Spray. Rmb – a reconfigurable multiple bus network. In *HPCA '96: Proc. of the 2nd IEEE Symposium on High-Performance Computer Architecture*, page 108, Washington, DC, USA, 1996.

[8] M. Huebner, T. Becker, and J. Becker. Real-time LUT-based network topologies for dynamic and partial FPGA self-reconfiguration. In *Proc. of the 17th Symposium on Integrated Circuits and System Design (SBCCI '04)*, pages 28–32, Pernambuco, Brazil, 2004.

[9] M. Huebner, M. Ullman, L. Braun, A. Klausmann, and J. Becker. Scalable Application-dependent Network on Chip Adaptivity for Dynamical Reconfigurable Real-Time Systems. In *Proc. of the Int. Conference on Field-Programmable Logic and its Applications (FPL 2004)*, pages 1037–1041, Antwerp, Belgium, August 2006.

[10] R. Koch, T. Pionteck, C. Albrecht, and E. Maehle. An Adaptive System-on-Chip for Network Applications. In *Proc. of 13th Workshop on Reconfigurable Architectures*, page 194, Apr. 2006.

[11] T. S. T. Mak, P. Sedcole, P. Y. K. Cheung, and W. Luk. On-FPGA Communication Architectures and Design Factors. In *Proc. of the Int. Conference on Field-Programmable Logic and its Applications (FPL 2006)*, pages 161–168, Madrid, Spain, August 2006.

[12] T. Pionteck, R. Koch, and C. Albrecht. Applying Partial Reconfiguration to Networks-on-Chip. In *Proc. of the Int. Conference on Field-Programmable Logic and its Applications (FPL 2006)*, pages 155–160, Madrid, Spain, August 2006.

[13] D. Puschini and F. Clermidy. A Comparison Between NoC and Bus Architectures Based on a Real-Application. In *Proc. of the 2nd Int. Workshop on Reconfigurable Communication-centric System-on-Chips (ReCoSoC)*, pages 161–168, Montepellier, France, July 2006.