ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 4, April 2014

Hybrid Routers for Wireless Network on Chip (Noc) Design

Selvakumar.V[1], Ajay.V.P[2]

PG Scholar, M.E.,VLSI DESIGN, KPR Institute Of Engineering And Technology, Coimbatore, India [1]

Assistant Professor, ECE Department, KPR Institute Of Engineering And Technology, Coimbatore, India [2]

Abstract: The design of more complex systems becomes an increasingly difficult task because of different issues related to latency, design reuse, throughput and cost that has to be considered while designing. In Real-time applications there are different communication needs among the cores. When NoCs (Networks on chip) are the means to interconnect the cores, use of some techniques to optimize the communication are indispensable. From the performance point of view, large buffer sizes ensure performance during different applications execution. But unfortunately, these same buffers are the main responsible for the router total power dissipation. Another aspect is that by sizing buffers for the worst case latency incurs in extra dissipation for the mean case, which is much more frequent. Reconfigurable router architecture for Noc is designed for processing elements communicate over a second communication level using direct-links between neighbouring elements. Several possibilities to use the router as additional resources to enhance complexity of modules are presented. The reconfigurable router is evaluated in terms of area, speed and latencies. With the reconfigurable router it was possible to reduce the congestion in the network, while at the same time reducing power dissipation and improving energy.

Keywords: Buffer, Latency, Network on chip, reconfigurable router, Throughput

I. INTRODUCTION

For Future MPSoC, a new communication design paradigm is needed. This communication medium should aim to provide high throughput, scalability, low power, reduced packet loss, utilize link efficiently, reduce contention and occupy less area on silicon. Chip density is increasing, allowing even larger systems to be implemented on a single chip. With increasing demands on flexibility and performance, these systems, known as Systems-on-Chips (SoCs), combine several types of processor cores, memories and custom modules of widely different sizes to form Multi-Processor Systems-on-Chips (MPSoCs). The bottleneck in such systems is shifting from computation to communication. The traditional way of using bus-based mechanisms for inter-module communication has two main limitations. Firstly, it does not scale well with increasing system complexity. Secondly, it couples computation and communication of the system leading to longer design times. Networks-on-Chip (NoCs) have been proposed as an efficient and scalable alternative to shared buses which allow systems to be designed modularly. From the above MPSoC examples one can see that, as it happens in the microprocessor market, each NoC used in an MPSoC can have different communication performance and costs, depending on the target application. Designing the same NoC router to cover the whole spectra of applications would mean an oversized and expensive router, in terms of area and power.At the same time, designing specific routers for different markets would mean that many important decisions would have to be taken at design time, hence precluding scalability and online optimizations targeting different behaviors with different application demands. This paper is organized as follows. Section II presents an analysis of the problem and identifies low efficiency in homogenous routers. The reconfigurable router is proposed in Section IV, where we describe the differences between the original and the new router architecture. In Section V, we present results of latency, buffer utilization, frequency, and power consumption. In Section VI, show some related works and a specific comparison of our proposal with the conclusions.

II. SYSTEM OVERVIEW

Communication between the hardware resources will occupy a very important place in future System-on-Chip designs. From the communication point of view it is convenient to use the OSI model to describe our platform. As Fig. 1 shows, at the highest-level the application designer uses the services offered by the layer beneath, the OS. For out platform the OS that manages the underlying communication infrastructure is called the operating system for reconfigurable systems .It is based on real-time Linux on top of which specific capabilities have been added. Part of the OS is implemented in hardware. The hardware part provides a standard interface to the IPs and the means to manage the data network. The transport level is also implemented by the interfaces. Router is the main building block of the Network. It serves two main functions, First it acts as the interface between the PEs and network. To provide re-configurability, wrappers are used to provide interface for non-compatible PEs. And secondly, it routes the data packets to the right path.

III. SOC ARCHITECTURE

System-on-Chip consists of a heterogeneous set of processors connected via a Network-on-Chip as depicted in Fig.2. The network consists of a set of

ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 4, April 2014

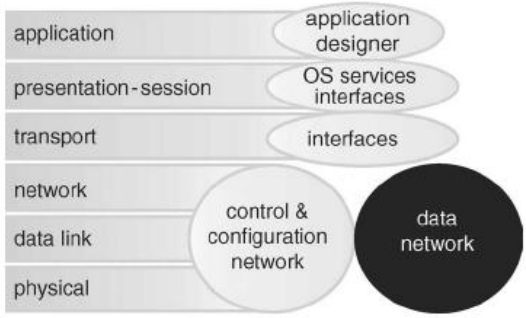

Fig.1. The Structure of our Platform from the communication point of view

interconnected by links. In this paper a regular two dimensional mesh topology of the routers. Every router is connected with its four neighboring routers via bidirectional point-to-point links and with a single processor tile via the tile interface. The SoC system is organized as a centralized system: one node, called Central Coordination Node (CCN), performs system coordination functions. The main task of the CCN is to manage the system resources. It performs run-time mapping of the newly arrived applications to suitable processing tiles and inter-processing communications to a concatenation of network links. It also tries to satisfy Quality of Service (QoS) requirements, to optimize the resources usage and to minimize the energy consumption.

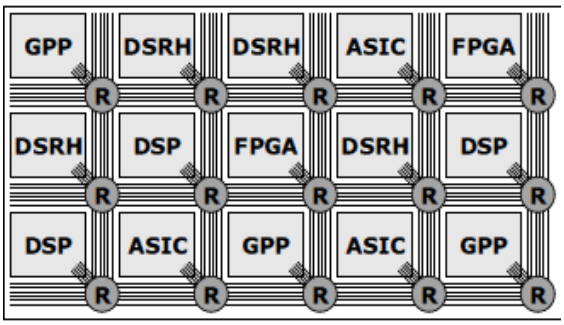

Fig.2. An example of a heterogeneous System-on-Chip (SoC) with a Network-on Chip (NoC).

The CCN does not perform run-time scheduling of individual processes and communications during execution. That is performed by the individual tiles and network routers. The CCN performs the feasibility analysis, spatial mapping, process allocation and configuration of the tiles and the NoC before the start of an application. It is expected that the on-chip communication networks of these future SoC will be one of the limiting factors for performance and possibly energy consumption. New architectural concept for on-chip communications using the communication characteristics of three wireless communication standards we propose a new reconfigurable circuit-switched on-chip network. This network benefits from the common characteristics of these wireless standards.

IV. PROPOSED ROUTER ARCHITECTURE

A. *Original Router Architecture*

The router architecture used in a Network on chip is a routing switch with up to five bi-directional ports (Local, North, South, West, and East), each port with two unidirectional channels and each router connected to four neighboring routers (North, South, West, and East). This router is a VHDL soft core, parameterized in three dimensions: communication channels width, input buffers depth, and routing information width. The architecture use the wormhole switching approach and a deterministic source based routing algorithm. The routing algorithm used is XY routing, capable of supporting dead-lock-free data transmission and the flow control is based on the handshake protocol.

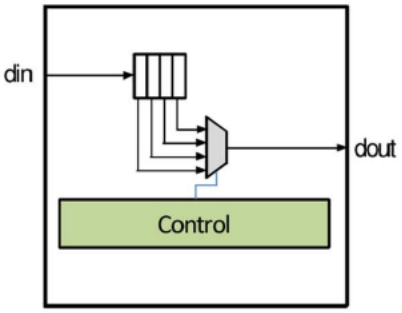

Fig.3. Input FIFO for Original Router

The wormhole strategy breaks a packet into multiple flow control units called flits, and they are sized as an integral multiple of the channel width. The first flit is a header with destination address followed by a set of payload flits and a tail flit. To indicate this information (header, payload, and tail flits) two bits of each flit are used. There is a round-robin arbiter at each output channel. The buffering is present only at the input channel. Each flit is stored in a FIFO buffer unit. The input channel is instantiated to all channels of the NoC, and thus all channels have the same buffer depth defined at design time.

B. *Reconfigurable Router Architecture*

The proposed architecture is able to sustain performance due to the fact that, statistically, not all buffers are used all the time. In our architecture it is possible to dynamically reconfigure different buffer depths for each channel. A channel can lend part or the whole of its buffer slots in accordance with the requirements of the neighbouring buffers. To reduce connection costs, each channel may only use the available buffer slots of its right and left neighbour channels. This way each channel may have up to three times more buffer slots than its original buffer with the size defined at design time.

Figure 4 presents the reconfigurable channel as an example. In this architecture it is possible to dynamically configure different buffer depths for the channels. In accordance with this figure, each channel has five multiplexers, and two of these multiplexers are responsible to control the input and output of data. These multiplexers present a fixed size, being independent of the buffer size. Other three multiplexers are necessary to control the read and write process of the FIFO. The size of the multiplexers

ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 4, April 2014

that control the buffer slots increases according to the depth of the buffer. These multiplexers are controlled by the FSM of the FIFO.

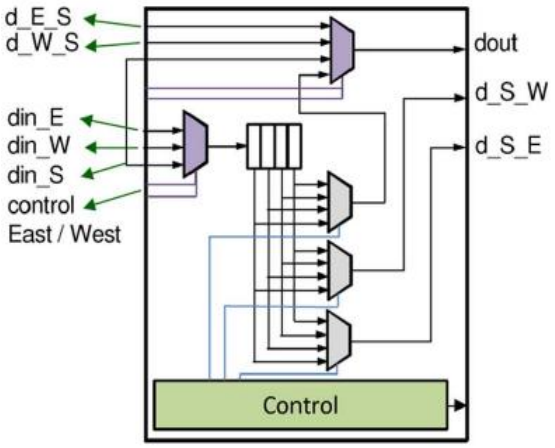

Fig.4. Input FIFO for Reconfigurable Router

In order to reduce routing and extra multiplexers, it adopted the strategy of changing the control part of each channel. Some rules were defined in order to enable the use of buffers from one channel by other adjacent channels. When a channel fills all its FIFO it can borrow more buffer words from its neighbours. First the channel asks for buffer words to the right neighbour, and if it still needs more buffers, it tries to borrow from the left neighbor FIFO. In this manner, some signals of each channel must be sent for the neighboring channels in order to control its stored flits.

Each channel needs to know how many buffer words it uses of its own channel and of the neighboring channels, and also how much the neighbor channels occupy of its own buffer set. A control block informs this number. Then, based on this information, each channel controls the storage of its flits. These flits can be stored on its buffer slots or in the neighbor channel buffer slots. Each input port has a control to store the flits and this control is based in pointers. Each input channel needs six pointers to control the read and writing process: two pointers to control its own buffer slots, two pointers to control the left neighbor buffer slots, and two more pointers to control the right neighbor buffer slots. In this design, it's not considering the possibility of the Local Channel using neighboring buffers, only the South, North, West, and East Channel of a router can make the use of their adjacent neighbors.

C. *Reliable router architectures*

On-chip networks have a tight area and power budget, necessitating simple router structures. Architectural approaches to reliable router architectures include triple modular redundancy (TMR) based approaches, such as the Bullet Proof router. However, in general, *N* modular redundancy approaches are expensive, as they require at least *N* times the silicon area. Another strategy explores the trade-offs of various levels of redundancy. Other work investigates the reliability of single components, for example a reliability-enhanced crossbar. Reconfiguration is approached by for pipelines, by for link failures, and by with modular design. Protection against transient errors has been explored.

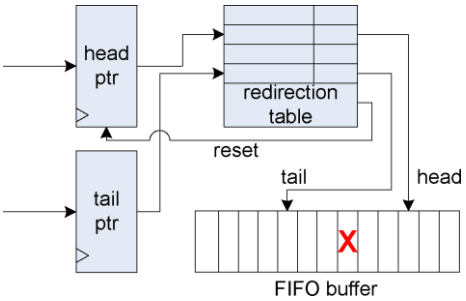

Fig.5. Flexible FIFO buffer logic

The reconfiguration process begins when one router broadcasts an error status bit through the network, although not necessarily its location, via an extra wire in each link. The initialized BIST unit then performs a distributed synchronization algorithm with other routers BIST units, ensuring that each BIST in the network runs all remaining routines in lock-step. After synchronization, each component of each router is diagnosed for faults. The diagnosis step does not rely on information from previous diagnostic phases, or from the detection mechanism, thus all permanent faults are diagnosed (or re-diagnosed), regardless of whether they are responsible for triggering this reconfiguration event, or not. Once all components and routers have been tested, faulty components are disabled and normal operation resumes. Since the BIST units are operational only during reconfiguration, they are power-gated off during normal operation for wear out protection.

D. *Input Port Swapping*

During the initial evaluation often a few faults would disable multiple network links or disconnect important processor nodes. To prevent this, we developed input port swapping to consolidate several faults into a single link failure, and to provide additional priority for maintaining connected processors. In order to safely route through the network, the routing algorithm requires functional bidirectional links. Each link is comprised of two input ports and two output ports, all four of which must be fully functional for the link to be operational. If one of these ports fails, port swapping may be used to maximize functional bidirectional links. Each input port is comprised of a FIFO buffer and a decode unit, identical for each direction of traffic input port swapper is used to modify which physical links are connected to each input port. For instance, Fig. 6 illustrates an example where a fault on the South port and a second fault in the adapter FIFO are consolidated, allowing the adapter link to use the former South FIFO. While it would be possible to include an additional output port swapper at the output ports, their small area and consequent low probability of faults did not warrant the area overhead of an additional swapper. On the other hand, the input ports constitute the majority of

ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 4, April 2014

the total router area and therefore are most susceptible to faults. Thus, adding input port swappers provides with the ability to consolidate the impact of several faults into one, or a few, links.

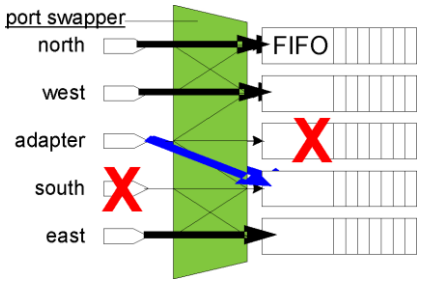

Fig.6. Port swapping unit

V. NETWORK SIMULATION

Figure 7 shows the channel of figure 4 organized to constitute the reconfigurable router. Each channel can receive three data inputs. Let us consider the south channel as an example, having the following inputs: the own input (din_s), the right neighbor input (din_e), and the left neighbor input (din_w). For illustration purposes, let us assume we are using a router with buffer depth equal to 4, and there is a router that needs to be configured as follows: south channel with buffer depth equal to 9, east channel with buffer depth equal to 2, west channel with buffer depth equal to 1, and north channel with buffer depth equal to 4. In such case, the south channel needs to borrow buffer slots from its neighbors. As the east channel occupies two of its four slots, this channel can lend two slots to its neighbor, but even then, the south channel still needs more three buffer slots. As the west channel occupies only one slot, the three missing slots can be lent to the south channel. When the south channel has a flit stored in the east channel, and this flit must be sent to the output, it is passed from the east channel to the south channel (d_e_s), and so the flit is directly sent to the output of the south channel (dout_s) by a multiplexer. The south channel has the following outputs: the own output (dout_s) and two more outputs (d_s_e and d_s_w) to send the flits stored in its channel but belonging to neighbor channels. The choice to resend the flits stored in a neighbor channels to its own channel before sending them to the output was preferred in order to avoid changes in others mechanisms of the architecture.

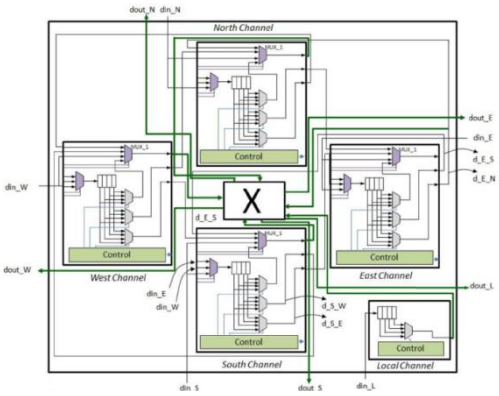

Fig.7. Proposed Router Architecture

Figure 7 shows an example of the reconfiguration in a router according to a needed bandwidth in each channel. First, a buffer depth for all channels is decided at design time in this case, it defined the buffer size equal to 4, as illustrated in Figure7.After this, the traffic in each channel is verified and a control defines the buffer depth needed in each link to attend to this flow, as shown in Figure 7.The distribution of the buffer words among the neighbor channels is realized as shown in Figure 7.Meanwhile, the buffer physical disposition in each channel correspondent the FIFO depth initially defined, as shown in Figure 7. But the allocation of buffer slots among the channels can be changed at run time, as exemplified in Figure 7.

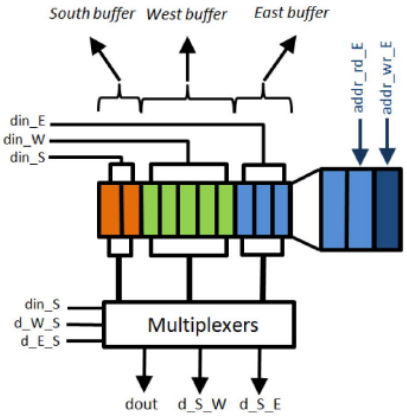

Fig 8 South Input Channel with the Buffers Partitioned for Three Channels

In this example considering again the South channel. Let us assume that the South channel divides part of its buffer with the neighboring channels (West and East channels). In this case, South channel uses only two buffer slots, five buffer slots are used by West channel, and three buffer slots are used by East channel. The number of buffers of a channel is partitioned according to the need for loans among the channels. In this way, each buffer slot is allocated in a mutually independent way. Pointers to each buffer partition are used in order to control the flit storage Figure.8 South input channel with the buffers partitioned for three channels: South, West, and East. Process (read and writes). Each slice of partitioned buffer in a channel has two pointers, one to control the read and another to control the write in the buffer (for example, addr_rd_E and addr_wr_E in Figure 8).

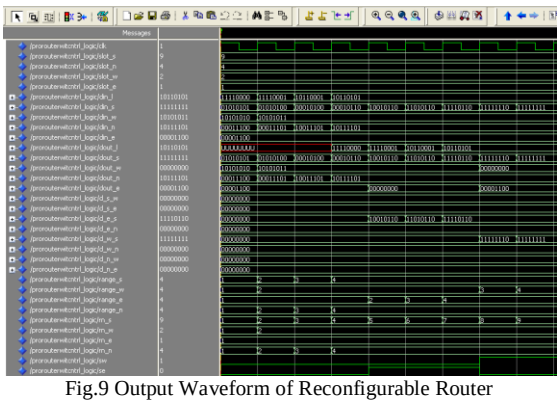

Fig.9 Output Waveform of Reconfigurable Router

Besides the pointers, there are other control signals that are needed, as the signal that indicates when the partitioned buffers are empty and full. With these signals, each channel allows neighbor channels to allocate buffer slots of this port and to guarantee that the flits are not mixed among the channels. The information about how many slots of buffers are used for each channel can been used to dynamically adjust their usage, consequently improving the efficiency. With this, one can monitor the noc traffic flow and analyze how the resources are being used. This information can be used to increase the efficiency of the noc design. In this proposal consists of reconfiguring the channel according to the availability of buffers in the channels. If a new channel depth is required, the buffer depth is updated slot by slot, and this change is made whenever a buffer slot is free.

VI. RESULT ANALYSIS

The proposed router was described in VHDL and used the ModelSim tool to simulate the code. Analysed the average power consumption, area, and frequency results to a standard cell library using the Design Compiler tool. The design operates at a supply voltage of 1.8 V, and the power results were obtained with a 200 MHz clock frequency in both architectures. The performance comparison of Routers is shown in Table 6.1.

With the proposed router it is possible to have one single NoC connecting different applications that might change their communicating patterns at run time. In the same way, this architecture allows application updates without compromising the performance of the system. Meanwhile, if a homogeneous router had been used in these situations, design modifications at design time would have had to be made to achieve the optimum case. In such case, one would need to redesign the homogeneous NoC to set buffer sizes and position of the cores in the network. The technique here proposed avoids costly redesigns and new manufacturing.

TABLE VI
COMPARISON OF ROUTERS AND PARAMETERS

Parameters	Area(um^2)	Delay(ns)	Function Block Inputs Used
Original Router	233.815	46.4	59%
Reliable Router	260.5	45.4	95%
Reconfigurable Router	340.861	40.4	89%

VII. CONCLUSIONS

This paper has described the architecture of a new dynamically reconfigurable NoC with intelligent nodes that changes the communication parameters for high data throughput and less timing delays. Our simulation results have demonstrated the effective use of network resources, making it a suitable alternative to traditional NoC. The downside of this network in the form of complexity of its node is compensated by the increased data throughput and low silicon cost. This kind of NoC can be ideal for reconfigurable MPSoCs with multimedia capabilities or for safety critical systems where the timing constraints are tight.

Future work is to include reconfiguration with fault tolerance for the same modified system. A built-in self-test at each router diagnoses the number and locations of hard faults. Architecture features, including crossbar bypass bus and port swapping, are then to be deployed to work around the faults for better performance.

VIII. REFERENCES

[1]. Ahmad and T.Arslan, "Dynamically reconfigurable NOC with bus based interface for ease of integration and reduced designed time,"2008.
[2]. Ahonen and J.Nurmi, "Hierarchically heterogeneous network-onchip,"2007.
[3]. Al Faruque and J. Henkel, "Configurable links for runtime adaptive on-chip communication,"2009.
[4]. Azimi, N.Cherukuri, "Integration challenges and tradeoff fortera-scale architectures," Aug 2007.
[5]. Benini and De Micheli, "Analysis of power consumption on switch fabrics in network routers,"2002.
[6]. Bertozzi, A.Jalabert, "NoC synthesis flow for customized domain specific multiprocessor systems-on-chip," Feb 2005.
[7]. Bouhraoua and E.Elrabaa, "Addressing heterogeneous bandwidth requirements in modified fat-tree network-on-chip," 2008.
[8]. Jingcao R.Marculescu, "System-level buffer allocation for application-specific networks-on-chip router design," Dec 2006.
[9]. LanandJ. Chen, "BiNoC: A bidirectional NoC architecture with dynamic self-reconfigurable channel," 2009.
[10]. Lee and N.Bagherzadeh, "Increasing the throughput of an adaptive router in network-on-chip (NoC),"2006.
[11]. Leeand H. Yoo, "Low-power network-on-chip for high performance SoC design," Feb 2006.
[12]. Manferdelli and C.Crall, "challenges and opportunities in many-core computing," May 2008.