International Journal of Scientific Engineering and Research (IJSER)
ISSN (Online): 2347-3878
Index Copernicus Value (2015): 56.67 | Impact Factor (2017): 5.156

Low Power and High Performance Reconfigurable Routers

Jyoti Sangogi[1], Rupali Patil[2]

[1]Sinhgad College of Engineering Department, Pune University, Pune, Maharashtra, India
[2]Sinhgad College of Engineering, Pune University, Pune, Maharashtra, India

Abstract: ***An energy-efficient network-on-chip (NoC) is presented for possible application to high-performance system-onchip (SoC) design. Network-on-chip (NoC) designs are based on a latency, power dissipation or energy and the balance is defined at design time. Setting all parameters, such as buffer size, at design time can use either excessive power dissipation by router under utilization or a higher latency. It is today's requirement. Network-on-chip architecture is proposed to enhance the performance of on-chip communication. The condition whenever the application changes its communication pattern, e.g. a portable phone downloads a new service. Large buffer sizes can ensure performance during the execution of different applications, but these same buffers are mainly responsible for the router total power dissipation. Another condition is that by sizing buffers for the worst case latency incurs extra dissipation for the mean case, which is much more frequent. Router delivers better performance and requires smaller buffer size than that of a conventional network-on-chip (NoC). NoC architecture proposed in [1]–[3]has gained high popularity due to its simplicity and flexibility.Propose the use of a reconfigurable router, where the buffer slots are dynamically allocated to increase router efficiency in an NoC, even under rather different communication loads. In the proposed architecture, the depth of each buffer word used in the input channels of the routers can be reconfigured at run time. The reconfigurable router allows up to 52% power savings, while maintaining the same performance as that of a homogeneous router, but using a 64% smaller buffer size.***

Keywords: NoC Network-on-chip, system-on chip SOC

1. Introduction

Moreover, experiments compare favourably with other dynamic topologies like virtual channels. An analysis of the problem and identifies low efficiency in homogenous routers. The reconfigurable router is proposed in chapter Three, where It describe the differences between the original and the new router architecture, present results of latency, buffer utilization, frequency, area, and power consumption. Some related works and a specific comparison of our proposal with virtual channels.

Multiprocessor System on chip (MPSoCs) are one of the technologies providing a way to support the growing design complexity of embedded systems, since they provide processor architectures adapted to selected problem classes, allied to programming flexibility. To ensure flexibility and performance, future MPSoCs will combine several types of processor cores and data memory units of widely different sizes, leading to a very heterogeneous architecture. The increasing interconnection complexity and the known scalability deficiency of buses require another model of interconnection. The communication among cores of an MPSoC having reusable and scalable interconnections is being provided by networks-on-chip (NoCs) .

While System-on-chip (SoC) designs provide integrated solutions to challenging design problems in the telecommunications, multimedia, and consumer electronics domains. Much of the progress in these fields hinges on the designers' ability to conceive complex electronic engines under strong time-to-market pressure. Success will rely on using appropriate design and process technologies, as well as on the ability to interconnect existing components including processors, controllers, and memory arrays reliably, in a plug-and-play fashion.

On-chip networks relate closely to interconnection networks for high-performance parallel computers with multiple processors, in which each processor is an individual chip. Like multiprocessor interconnection networks, nodes are physically close to each other and have high link reliability. Further, developers have traditionally designed multiprocessor interconnections bandwidth and latency constraints to support effective parallelization. Similar constraints will drive micro network design [1]. NoCs have been proposed to integrate several Intellectual Property (IP) cores, providing high communication bandwidth and parallelism.

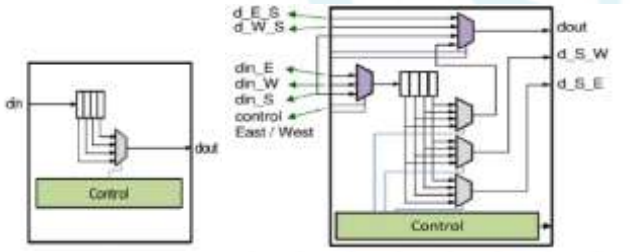
Over the last few years, dual-core processors have become mainstream in desktop, mobile, and server platforms due to their ability to deliver higher system. The trend towards higher core counts is continuing strong with quad-core processors establishing an increasing presence across all market segments. Industry experience with small-scale shared memory multiprocessors enabled a relatively effortless integration of a small number of processors into a single die. Moving beyond a small number to tens or hundreds of processor cores at the same time as other platform ingredients such as memory controllers, I/O bridges, and graphics engines find their way to the processor die, introduces significant challenges to the infrastructure that ties all these together.

This infrastructure includes the on-die interconnect, the cache hierarchy, the memory, the I/O, and system interfaces. The term uncore to collectively refer to all the elements in the processor die that are not computing engines. The tera-scale architecture uncore must be capable of satisfying the communication requirements of a large number of cores, fixed function computing engines, and the external memory and I/O system. In order to scale effectively, the uncore must find ways to keep the off-die bandwidth manageable and

within the constraints of cost, power, and high-speed signaling technology. The uncore must be able to offer significant flexibility to assign computing resources to concurrently solve different problems. It must include mechanisms to enable high-volume manufacturing by enhancing reliability in the presence of increasing architectural complexity and its functions within a constrained power envelope.

The proposed router architecture was embedded in the SoCIN NoC. SoCIN has a regular 2-D-mesh topology and parametric router architecture. The router architecture used is RaSoC, which 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 uses the wormhole switching approach and a deterministic source-based routing algorithm. The routing algorithm used is -routing, capable of supporting deadlock-free data transmission, and the flow control is based on the handshake protocol. 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

2. Reconfigurable Router Architecture



(a) (b)

Figure 8: Input FIFO (a) original and (b) proposed router. [12]

If an NoC's router has a larger FIFO buffer, the throughput will be larger and the latency in the network smaller, since it will have fewer flits stagnant on the network [. Nevertheless, there is a limit on the increase of the FIFO depth. Since each communication will have its peculiarities, sizing the FIFO for the worst case communication scenario will compromise not only the routing area, but power as well [6]. However, if the router has a small FIFO depth, the latency will be larger, and quality of service (QoS) can be compromised. The proposed solution is to have a heterogeneous router, in which each channel can have a different buffer size. In this situation, if a channel has a communication rate smaller than its neighbor, it may lend some of its buffer slots that are not being used. In a different communication pattern, the roles may be reversed or changed at run time, without a redesign step.

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 neighboring buffers. To reduce connection costs, each channel may only use the available buffer slots of its right and left neighbor 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.

Fig.7.shows the original and proposed input FIFO. Comparing the two architectures, the new proposal uses more multiplexers to allow the reconfiguration process. Fig. 9. presents the South 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 that control the buffer slots increases according to the depth of the buffer. These multiplexers are controlled by the FSM of the FIFO. In order to reduce routing and extra multiplexers, we 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 neighbors. First the channel asks for buffer words to the right neighbor, 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.

In result, 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 each case, one pointer to the read operation and one pointer to write operation).

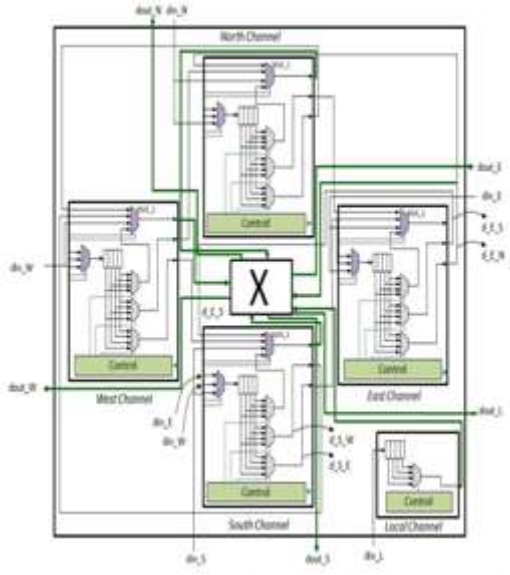


Figure 9: Proposed router architecture [12]

In this design, we are 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. As mentioned before, the loan granularity used in this proposal is a buffer slot. The area results of the reconfigurable router would not present a significant change if loan granularity was increased. This is due to the fact that the control overhead is defined mainly by the FIFO's control circuit. As the buffers are implemented using circular FIFOs, the FIFO pointers are incremented to each new slot, and this control will be the same whatever the used loan granularity. If we increase the loan granularity to more than one slot, then the loss in performance could be large, and the reduction in area or power would be minimal.

In addition, we are considering sharing of the buffer slots only among adjacent channels. This decision is based on the costs of interconnections, multiplexers, and logic to control the combination of all loans among all input channels. Consequently, the area and power consumption would be much larger if we consider the last case, and the gains in performance would not be large enough to compensate this extra cost. Fig.10 shows the channel of 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.

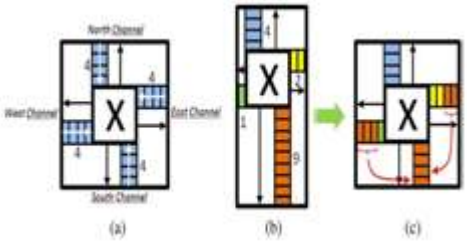


Figure 10: (a) Router designed with FIFO depth 4; (b) One example of need of configuration of the router; (c)

Reconfiguration of the buffers to attend the need.[7]

In this manner changes cannot be done in the routing algorithm, avoiding the possibility of data deadlock, since the NoC continues using routing, which is intrinsically deadlock free. With this definition, the complexity of the implementation to obtain the correct function of the router was reduced in this aspect. Each flit stored in a neighbor channel returns to the respective channel when it needs to be sent to an output channel. In this case, when an input channel is connected to an output channel, the flits are sent one-by-one, and the pointers are updated as each flit is sent.

As each channel knows how many buffer slots it has allocated, when the pointers present an address belonging to a neighbor buffer slot, the control of the first multiplexer of Fig.10 allows the sending of the respective flits to the output of its channel. As we do not change the routing policy, there is no possibility of entering a deadlock situation. Of course, one could be concerned about one channel asking buffers from another channel which is also asking for buffers. Since only the neighbors are asked about lending/borrowing, no cycle can be made, and hence at the circuit level there is also no possibility of deadlock.

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, the buffer size is defined equal to 4. 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.

The distribution of the buffer words among the neighbor channels is realized as shown in Fig. 10. Meanwhile, the buffer physical disposition in each channel correspondent the FIFO depth initially defined, as shown in Fig. 11. but the allocation of buffer slots among the channels can be changed at run time, as exemplified in Fig. 11(C).

Now, let us consider an input buffer with ten slots, as shown in Fig. 11. In this example we are 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, eachbuffer slot is allocated in a mutually independent way. Pointers to each buffer partition are used in order to control the flit storage Fig. 12. South input channel with the buffers partitioned for three channels: South, West, and East. Process (read and write). 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 .

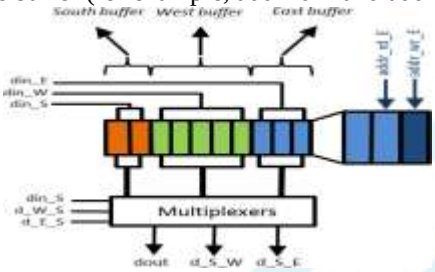


Figure 11: South input channel with the buffers partitioned for three channels [7]

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.

It 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 his change is made whenever a buffer slot is free. For the set of benchmarks used in this work, and as reported in many related works, whenever the application is changed, a different bandwidth is required among the channels. The reconfigurable router can change its depth in only few cycles, which means a very small performance overhead. Moreover, as each core send spackets at a different rate, the reconfiguration of the router was implemented considering that in some possible interval among packets there would be a time-slack. As the traffic is composed of packets, the buffers are not used 100% of the time in all parts of the network

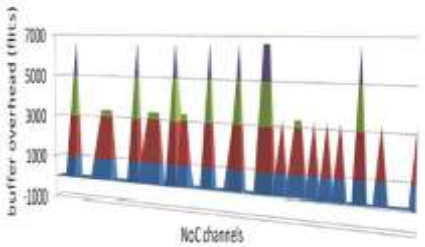


Figure 12: The original architecture[12]

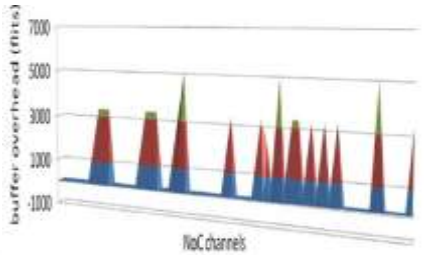


Figure 13: The reconfigurable router[12]

Fig. 13. Flits need to wait the buffer availability to be sent to the next router for the VOPD application for buffer depth equal to 4 for (a) the original architecture and for (b) the reconfigurable router.

3. Conclusion

In this topic, the advantage of the use of an NoC with reconfigurable routers instead of homogeneous ones has been presented.

Using reconfiguration, one can dynamically change the buffer depth to each channel, in accordance to the necessity of the application, increasing the power efficiency of the system for the same performance level and verified that to reach the same performance obtained with the reconfigurable router, the original architecture needs many more buffers.

The new router, while reaching the same performance than the original architecture, obtained a reduction of approximately 25% of power consumption in the worst case, and of 52% for the best case analyzed. Besides, when compared with the ViChaR architecture, other proposal obtains 78% of power reduction for the same configuration. Moreover, the reconfigurable router obtains the same performance of the homogeneous router with a buffer depth 64% smaller.

Moreover, with the new architecture it is possible to reconfigure the router in accordance with the application, obtaining similar performances even when the application radically changes

References

[1] "Network on chips: A new SoC paradigm," L. Benini and G. De Micheli, IEEE Computer, vol. 35, no. 1, pp. 70–78, Jan. 2002.
[2] "Integration challenges and tradeoff for tera-scale architectures", M. Azimi, N. Cherukuri, D. N. Jayasimha, A. Kumar, P. Kundu, S. Park, I. Schoinas,

and A. S. Vaidya, Intel Technol. J., vol. 11, no. 3, Aug. 2007.
[3] "Challenges and opportunities in many-core computing," L. Manferdelli, N. K. Govindaraju, and C. Crall, Proc. IEEE, vol. 96, no. 5, pp. 808–815, May 2008.
[4] Xbox 360 system architecture," J. Andrews and N. Baker, IEEE Micro, vol. 26, no. 2, pp. 25–37, Mar./Apr. 2006.
[5] Area and performance optimization of ageneric network-on-chip architecture," M. Vestias and H. Neto, "in Proc. Symp. Integr. Circuits Syst. Des. (SBCCI), 2006, pp. 68–73.
[6] "Leakage power modeling and optimization in interconnection networks," C. Xuning and L. Peh, in Proc. Int. Symp. Low Power Electron. Des. (ISLPED), 2003, pp. 90–95.
[7] "Analysis of power consumption on switch fabrics in network routers," T. Benini and G. De Micheli, in Proc. 39th Des. Autom. Conf. (DAC), 2002, pp. 524–529.
[8] "Increasing the throughput of an adaptive router in network-on-chip (NoC)," S. E. Lee and N. Bagherzadeh, in Proc. Int. Conf. Hardw./Softw. Codes. Syst. Synth., 2006, pp. 82–87.2008, pp. 309–314.
[9] Survey of Network on Chip (NoC) Architectures & Contributions" Ankur Agarwal, Florida Atlantic University, Volume 3, Issue 1, 2009
[10] Arslan, "Dynamically reconfigurable NOC with bus based interface for ease of integration and reduced designed time," Ahmad, A. Ahmadinia, and T. Arslanin Proc. NASA/ESA Conf. Adapt. Hardw. Syst. (A
[11] "Hierarchically heterogeneous network-onchip," T. Ahonen and J. Nurmi in Proc. Int. Conf. Comput. As a Tool (EUROCON), 2007, pp. 2580–2586.
[12] "Reconfigurable Routers for Low Power and High Performance", Débora Matos, Student Member, IEEE, Caroline Concatto, VOL. 19, NO. 11, NOVEMBER 2011