Journal of Computer Science 8 (6): 987-993, 2012
ISSN 1549-3636


# Buffer Pocketing and Pre-Checking on Buffer Utilization

[1]Saravanan, K. and [2]R.M. Suresh
[1]Department of Information Technology,
Velammal Engineering College, Chennai, Tamilnadu, 600 080, India
[2]Department of Computer Science,
R.M.D. Engineering College, Chennai, Tamilnadu, India

**Abstract: Problem statement:** In this study, the paper proposes the development of Effective Buffer Utilization on Adaptive Router with Buffer Pocketing and Pre-Buffer Checking Technique. Network-on-Chip could be prepared more capable by tricky faster routers. By using improved buffers, superior number of ports and channels, adaptive routing, all of which acquire key of overheads in hardware costs. **Approach:** This technique will improve communication efficiency without increasing the buffer size with support of input buffer space feedback controller in an input channel. A Buffer-Pocketing system enables the input channels to use the unused buffer from another channel at runtime that have not enough buffer space to utilize as per the input flits. The current buffer status in a router could be updated on each cycle of data flits transmitted to buffer space controller with the help of Router Monitor Sensor (RMS). **Results:** Implementation results of the proposed design for a 64-bit 4 input-buffer router show a reduction of the average packet transmission latency and an increase of the average transmission flits. The results show that the proposed design can reduce the cycles required for transmitting a fixed number of packets, when compared to that without buffer stealing. **Conclusion:** The study confirmed that the pre-buffer checking and feedback collecting from the router design take the place of the original design in terms of both throughput and latency. Thus, BS is more robust in handling hardware overhead ratio.

**Key words:** Buffer pocketing, adaptive router, buffer space, buffer pool management, RMS

## INTRODUCTION

A basic NoC architecture is composed of routers, communication links between routers and a Network-Interface Component (NIC) between each pair of router and processing element. NoC allows much higher bandwidth through parallel communication. Each router can accept at the same time the flits arriving from all of the input channels by storing them in input buffers. The input buffers in a router are used to provisionally store arriving flits that cannot be forwarded directly to required output channels. The flits in the buffers are then transmitted through the output channels. All the above three cases motivate the need for the effective use of router buffers such that the communication efficiency of inter link networks can be elevated. Buffers into a router at design time, buffer stealing enables the input channels that have insufficient free buffer space to utilize at runtime the free input buffers from other input channels. Multiprocessor Systems-On-Chips (MPSoCs) provide a huge design space exploration for applications with high computational demands. MPSoCs are used in applications such as networking, signal processing and multimedia. Raising its programmability makes them more flexible, allowing its use in a wide range of digital systems. In this way, the MPSoC lifetime increases, reducing the price for the final consumer. Since the platform computational power is distributed in several Processing Elements (PEs), its organization and message passing have a crucial role in the system performance. As the number of PEs trends to increase to dozens in a near future, a scalable interconnection architecture, such as traditional busses, is not recommended to be used in such systems. Networks-on-Chip (NoCs) supports the communication requirements of modern MPSoCs, due to features as scalability, QoS support, parallel transactions and higher aggregated throughput.

## MATERIALS AND METHODS

**Buffer stealing designs:**

**Thief buffer*:*** A thief buffer is a buffer that steals the buffer space of other channels when its free buffer space is not enough to store incoming flights. For proof

**Corresponding Author:** Saravanan, K., Department of Information Technology, Velammal Engineering College, Anna University, Chennai, Tamilnadu, 600 080, India Tel: + 91 09976667047

of concept, in this study the buffers in the north and south Input channels are designed as thief buffers.

**Victim buffer:** A wounded buffer is a buffer whose free space can be stolen by a thief buffer. Huang and Hwang (2006) here it needs to record that a stored flit is from its own input channel or from that of a thief buffer. For proof of concept, in this work the buffers in the west and east input channels are designed as wounded buffers.

**Buffer storage unit:** A buffer storage unit is the basic amount of memory space to store a flit. In this study, here assume the size of a buffer storage unit is 8 bits and the size of the whole input buffer is 64 bits (8 bits × 8 buffer storage units).

**Adaptive physical channel regulation schemes:** In this study primarily elucidate the three regulatory schemes used in an APCR Router1.

**Monopolizing:** Similar to a common router, monopolizing allows only one Virtual Channels (VC) to use the total bandwidth of the output channel every cycle. In a generic router design, the fleet size is usually the same as the pH its size. A VC can fully use the whole bandwidth of the output channel. However, in an APCR router design the flit size is smaller than the pH its size. In other words, potentially multiple flits can be in the same channel on currently. Considering this characteristic, an APCR router allows a virtual channel to transmit multiple flits in the same cycle. There is one restriction on this situation. Wormhole flow control allows different packets stored in a VC without interleaving and the basic routing unit is a packet not a flit. Therefore, a virtual channel is not allowed to transmit as many flits as it has.

**Fair-sharing:** Considering the incompetent channel utilization scenario described above, the paper proposes fair-sharing. VCs rather share the output channel resources. To achieve fair-sharing, a wide physical channel needs to be separated into quite a few small parts, called sub-channels. This study reserves a different sub-channel for VCs of the same input port. VCs of different input ports share sub-channels. Assuming that a physical channel is divided into four sub-channels and each input port have four VCs, then each VC of the same input port can have one sub-channel of its own.

**Buffer space updating:** The design of buffer status in a router might update for each cycle in data flits for transmission in a router by a buffer stealing. From those kinds of buffer status, input channel will check the buffer status then decide to end amount of flits to route of transmission.

**Channel-stealing:** To further improve the operation of wide channels, the paper proposes channel-stealing, which is built upon fair sharing. Different from fair-sharing, if a Virtual Channel (VC) finally has no flit to be sent, its sub-channel will be stolen by other VCs. Here the stealing occurs in two ways. One is stealing from VCs belonging to the same input port and the other is stealing from VCs of different input ports which have the same output direction. Channel-stealing explores the channel income carefully. It optimizes the arbitration of output channels by using the buffer occupancy information from each VC and finally increases the network throughput. By consideration of VC2 and VC3 have no flits to be sent. VC0 and VC1 can steal the sub-channel assigned to VC2 and VC3 and send more than one flit. There are two options: Either VC0 and VC1 send two flits each or VC0 sends one flit while VC1 sends three flits.

**Related works:** The design methodology and key research problems of NoCs. In exact, a large buffer size reduces the average packet latency in NoC; however, larger buffer size also increases the overall NoC area. To increase buffer use, some buffer sharing methods (Liu and Delgado-Frias, 2007; Lai *et al*., 2008; Hashimoto *et al*., 2005) were proposed for Virtual Channels (VC) in a router design. The performance improvement achieved by the buffer sharing methods in VC routers is limited, because the control complexity of VC design incurs huge overheads in terms of additional hardware resources and power encumbrances. For example, virtual channel buffers require up to nearly 50% of area and account for 64% of leakage power in a router implemented under the 70 nm CMOS technology.

Chen and Peh (2003) proposed the shifting in the interconnection architecture from busses to NoCs, modern MPSoCs need to jointly manage computation and communication resources to ensure QoS to specific flows. The abstraction of the communication or the computation architectures to higher abstraction levels (e.g., Through an API), hides the hardware complexity, allowing the system programmer to explore the design space in an efficient way. The Tilera MPSoC consists of an 8×8 grid of tiles connected by five overlapped 2D mesh NoCs (iMesh). To take advantage of the whole bandwidth afforded by the on-chip integration of multiple mesh networks, Tilera provides a C-based user-level API library called iLib. There are two broad categories of communication in iLib: socket-like channels for streaming algorithms and an MPI-like

message passing for ad hoc messaging. ILib provides several channel APIs, each optimized for a different communication needs such as low latency and high throughput. Through several communication primitives, it lets the programmer to use the best communication interface for the application being developed.

Several previous NOC designs have been proposed to explore the abundant channel resources. Work by (Hausman *et al*., 1990) looks at multiple flits sharing a channel. In their network, there are two kinds of flits, short and long. The sharing condition is simple: if two short flits are routed to the same output port, they can simultaneously traverse the crossbar and output channel. For long flits, no sharing is applied. One concern about this design is that to support two kinds of flit sizes in the same network, the flow control can be challenging. Since a flit is the basic flow control unit, providing two kinds of flits will also make the credit management complicated. Hoskote *et al*. (2007) Introduces the concept, Spatial Division Multiplexing (SDM), into the NOC design. The results show that SDM is a more interesting approach than Time Division Multiplexing (TDM), due to the high complexity and power needed by buffers to store the TDM configuration for each clock cycle.

Howard *et al*. (2010) here the study discusses several prior works studying directory protocol optimizations. Mukherjee and Hill propose using prediction to accelerate directory coherence protocols using a predictor based on a Pap-style branch predictor. They predict the upcoming coherence actions based on the recent history of coherence requests. The Memory Sharing Predictor improves on the accuracy of coherence predictors by limiting predictions to memory requests (reads, stores and upgrades) rather than all coherence messages (acknowledgment and invalidations are eliminated). Gratz *et al*. (2006; 2008) here they proposed the remote access latency can be reduced by having the directory initiate these coherence requests speculatively. Circuit-switched coherence focuses on predicting who will source the data for a given request to accelerate that transfer via a circuit-switched connection.

**Proposed scheme:** The proposed buffer pocketing design was implemented at the cycle-accurate level. This study analyzed several situations to illustrate the advantages and overheads of the proposed buffer-pocket design. The paper uses burst traffic patterns which represent different traffic loads to compare the proposed design with the original buffer design. Buffer management will maintain the buffer level of router and require buffer size for next data during the packet transmission time. Here, boost traffic refers to a periodic data transmission that exhibits a very high data signaling rate for very short transmission durations, which are interrupted by fixed idle time intervals.

**Router monitoring sensor:** RMS is included in our proposed method, for monitoring the each process handling by the router. This sensor has an information about the buffer usage in an ongoing process then produce a status of buffer in a router for the next packet transmission. This will update the buffer status to the input buffer control for managing the input buffer to router on each cycle.

**Input buffer control:** Input buffer control used in the input session of router to manage the flow of input packets to the router. This control will reduce the traffic flow in a router and will increase router efficiency. Buffer control placed in between a router and router input channel for easy flow management process. It gets updated information from the RMS about the buffer status of the router on ongoing process.

**Buffer steals:** Adding extra buffers into a router at design time, buffer stealing enables the input Channels that have unsatisfactory free buffer space to make use of at runtime the free input buffers from other input channels. Buffer steal process will take place while any one of the input channels require an extra buffer to manage the data flow in the router.

Implementation algorithm:

Step 1: Input to the router channels sufficient storage size
Step2: Set input channel value to buffer control
If (buffer required > buffer empty)
Stop the process
Else
Send buffer to router input channel
Step 3: Router has to get an input channel for
If input channel not having enough buffer
Search a used buffer and steal buffer
Else
Pass packet to output channel
Repeat step3 until packet send to the destination
Step 4: Buffer sensing have get update Buffer in Router
Step 5: Update buffer control buffer variable with buffer status
Step 6: Repeat step 1-5 until packet send to Destination

Liu and Delgado-Frias (2007) describe a hierarchical QoS model for managing multimedia applications running on an MPSoC. The target application is a MPEG-4 shape-texture decoder that is fully object based, using arbitrary object shapes. The work considers a class of QoS systems that relies on predicting the execution times of the application at run-

time, while also taking into account the data dependencies. The architecture of the proposed QoS concept is based on two negotiating managers, instead of a conventional single resource manager (Fig. 1). The buffer stealing is processed with the unused memory during the runtime for the better usage to get transmission efficiency if the buffer size exceeds the limit the buffer fragment will triggered for the buffer usage of additional part. Router Monitor Sensor (RMS) will get updated information about the router buffer at runtime with the time interval and then this information could be updated in the input buffer controller for preventing the buffer traffic in router.

## RESULTS

The Central Buffer (CB) design outperforms the Buffer Stealing (BS) design. From these two buffer usage , one might conclude that the CB design is better in reducing buffer congestion and thus enhancing the router throughput However, the CB design suffers from serious hardware resource overhead and performance overhead. Table 1 shows the synthesis result of various buffer designs corresponding to frequency and hardware overhead. Table 1: Various buffer designs with Frequency and hardware overhead.

Results for different buffer designs, shows that the hardware overhead of the CB design is very large (almost an additional amount of 215% resources required) than the conventional buffer design. Thus the CB design is not a cost-efficient implementation. However, the proposed buffer pocketing method only incurs a hardware resource overhead of 22% compared to the conventional buffer design.

To compare the benefit to overhead ratio for different buffer designs, the paper computes the throughput to hardware overhead ratio (flit (#)/hardware overhead (%)), as shown in and the ratio of latency to hardware overhead ((1/latency (# of cycles))/hardware overhead (%)), this study observe that with buffer pocketing a router exhibits an enhancement of maximum 35% in throughput to hardware overhead ratio than the original buffer design. However, with the faster output period of the buffer, the buffer congestion does not occur frequently and thus the Buffer Stealing (BS) mechanisms do not need to steal the free buffer space of other input channel.

Table 1: Synthesis result of various buffers

| Buffer design | Frequency | Hardware overhead (%) |
|---|---|---|
| Buffer pocketing | 203.442 | 22.00 |
| Auxiliary buffer | 174.000 | 43.00 |
| Central buffer | 142.511 | 215.48 |

As shown in Fig. 2, the study finds that the average increase in the number of flits output from the North input buffer is almost the same, irrespective of whether the traffic loads on the East and West buffers are as heavy as or lighter than that on the North and South buffers or even negligible. Due to buffer stealing, the maximum throughput increase is 50% and average throughput increase is 29% compared to that of the extended buffer. This maximum average throughput is achieved when I: O = 3:7, which shows that buffer stealing is more effective when there is an appreciable difference in input and output ratio (medium congestion). It becomes less effective when there is little difference in the I/O ratio or when there is very heavy congestion.

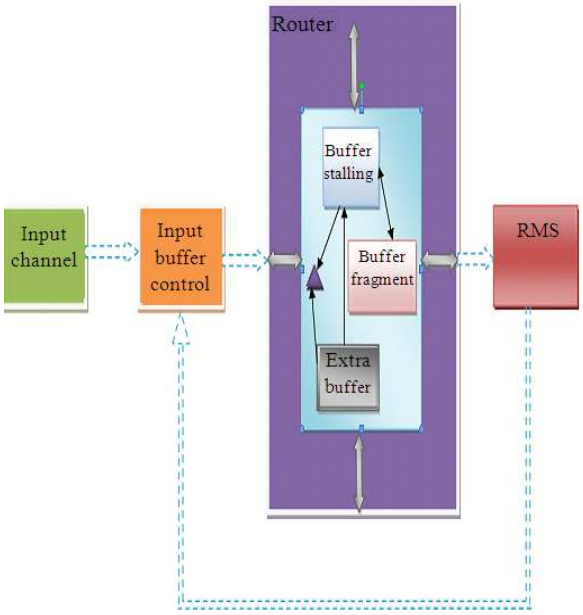


Fig. 1: Architecture of the proposed method

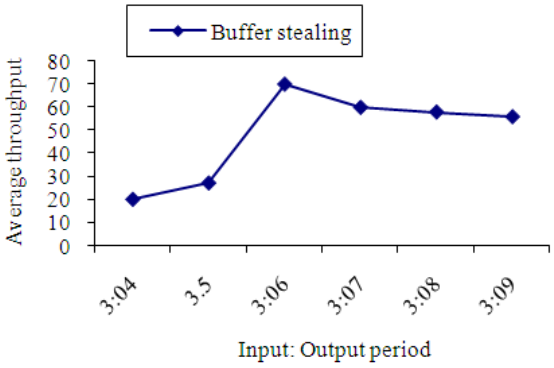


Fig. 2: Average increment of flits

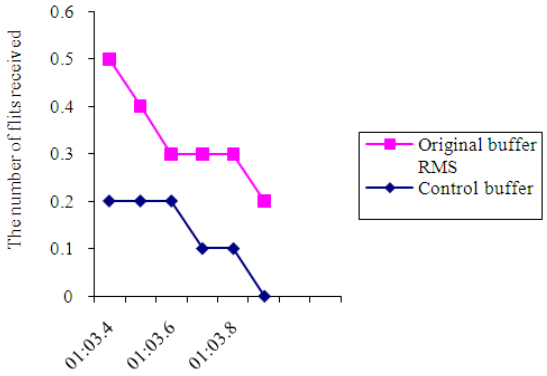


Fig. 3: Congestion tolerance rate in throughput reduction

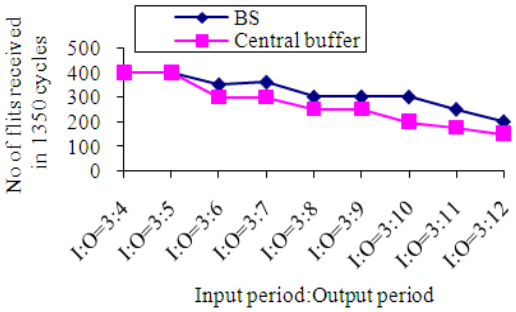


Fig. 4: Average increment of flits in the 1350 cycle's period

Figure 3 illustrates the growth in latency for transmitting flits via a router for an increasing output period. This study observes that with buffer stealing a router exhibits an average growth of 30% in latency, while the extended buffer design shows an average growth of 35%. Also note that the trend of latent growth in BS design is tardy than that for the extended buffer design. A slower latency growth is achieved by buffer stealing because of the reduced average waiting time for each flit. Also note how the reduced growth in latency is achieved by buffer stealing irrespective of the traffic load on the East and West buffers.

Figure 4 The number of flits that can be received by a router in a fixed duration of 1350 cycles. For the fairness of comparison, the Central Buffer (CB) allows the sharing among three entire buffers (200 bits) since the thief buffer in BS design has its local buffer and two victim buffers to be used. From Fig. 7, it shows that the central buffer is able to receive more flits than our BS design.

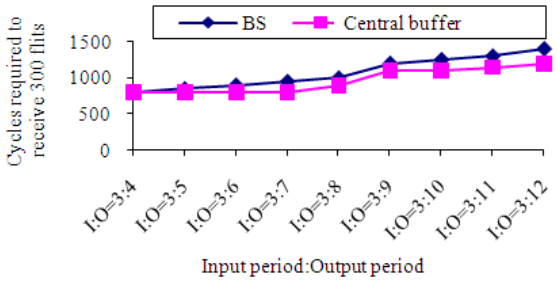


Fig. 5: Cycles reduction for receiving 30 flits

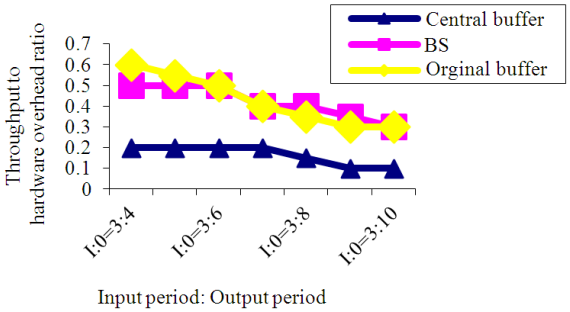


Fig. 6: Throughput to hardware overhead ratio

Because the thief buffer in BS can share the free space in victim buffers; however, the victim buffers cannot share the free spaces in the other victim buffer and thief buffer.

Figure 5 shows the number of cycles required to receive a fixed number of 300 flits by the BS design and the CB design. The CB design outperforms the BS design. From the above two experiments, one might conclude that the CB design is better in reducing buffer congestion and thus enhancing the router throughput and reducing the flit waiting time. However, the CB design suffers from serious hardware resource overhead and performance overhead. Table 1 shows the synthesis results for different buffer designs, where this study proposes that the hardware overhead of the CB design is very large (almost an additional amount of 220% resources required) than the conventional buffer design. Thus the CB design is not a cost-efficient implementation. However, the proposed buffer stealing method only incurs a hardware resource overhead of 25% compared to the conventional buffer design.

This study Fig. 6 observe that with buffer stealing a router exhibits an enhancement of maximum 32% in throughput to hardware overhead ratio than the original buffer design. However, with the faster output period of the buffer. The buffer congestion does not occur frequently and thus the BS mechanism does not need to steal the free buffer space of other input channel.

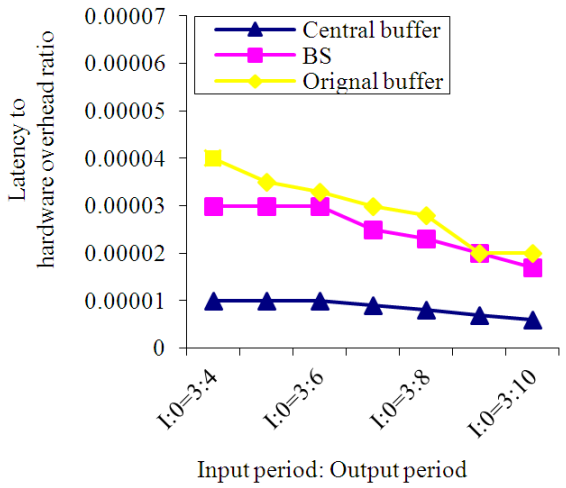


Fig. 7: Latency to hardware overhead ratio

Similar situations can be found in the ratio of latency to hardware overhead shown in Fig. 7. Based on the above experimental results, the paper can conclude that the proposed buffer stealing method can enhance router performance at smaller hardware overhead.

## DISCUSSION

The Central Buffer (CB) design outperforms the Buffer Stealing (BS) design. From these two buffer usage, one might conclude that the CB design is better in reducing buffer congestion and thus enhancing the router throughput. This study observes that with buffer stealing a router exhibits an enhancement of maximum 32% in throughput to hardware overhead ratio than the original buffer design. However, with the faster output period of the buffer. The buffer congestion does not occur frequently and thus the BS mechanism does not need to steal the free buffer space of other input channel.

## CONCLUSION

In this study the paper discussed a buffer pocketing method with RMS that can steal the free buffers in the low-load channels to support the channels which require more buffers than others for storing large amount of packets during the runtime. Our practical verified with different cyclic data pocket transferring through input channel and input buffer controller. Our results show that the proposed design can reduce the cycles required for transmitting a fixed number of packets, when compared to that without buffer stealing. The experiments show that the pre-buffer checking and feedback collecting from the router design take the place of the original design in terms of both throughput and latency. Thus, BS is more robust in handling hardware overhead ratio. Future work will consist of the support for dynamically reconfigurable system.

## REFERENCES

Chen, X. And L.S. Peh, 2003. Leakage power modeling and optimization in interconnection networks. Proceedings of the International Symposium on Low Power Electronics and Design, Aug. 25-27, ACM, Korea, pp: 90-95. DOI: 10.1145/871506.871531

Gratz, P., B. Grot and S.W. Keckler, 2008. Regional congestion awareness of load balance in networks-on-chip. Proceedings of the 14th International Symposium on High-Performance Computer Architecture, Feb. 16-20, IEEE Xplore Press, Salt Lake City, pp: 203-214. DOI: 10.1109/HPCA.2008.4658640

Gratz, P., C. Kim, R. McDonald, S.W. Keckler and D. Burger, 2006. Implementation and evaluation of on-chip network architectures. Proceedings of the International Conference on Computer Design, Oct. 1-4, IEEE Xplore Press, San Jose, pp: 477-484. DOI: 10.1109/ICCD.2006.4380859

Hashimoto, M., T. Yamamoto and H. Onodera, 2005. Statistical analysis of clock skew variation in H-tree structure. Proceedings of the 6th International Symposium on Quality of Electronic Design, Mar. 21-23, IEEE Xplore Press, DOI: 10.1109/ISQED.2005.114

Hausman, K., G. Gaudenzi, J. Mosley and S. Tempest, 1990. US patent 4978927-programmable voltage controlled ring oscillator.

Hoskote, Y., S. Vangal, A. Singh, N. Barker and S. Barker, 2007. A 5-GHz mesh interconnect for a teraflops processor. IEEE Micro, 27: 51-61. DOI: 10.1109/MM.2007.4378783

Howard, J., S. Dighe, Y. Hoskote, S. Vangal and D. Finan *et al*., 2010. A 48-Core IA-32 message-passing processor with DVFS in 45nm CMOS. Proceedings of the International Solid State Circuits Conference IEEE, Feb. 7-11, IEEE Xplore Press, San Francisco, pp: 108-109. DOI: 10.1109/ISSCC.2010.5434077

Huang, P.T. and W. Hwang, 2006. 2-level FIFO architecture design for switch fabrics in network-on-chip. Proceedings of the International Symposium on Circuits and Systems, May 21-24, IEEE Xplore Press, Island of Kos, pp: 4863-4866. DOI: 10.1109/ISCAS.2006.1693720

Lai, M., Z. Wang, L. Gao, H. Lu and K. Dai, 2008. Dynamically-allocated virtual channel architecture with congestion awareness for on-chip routers. Proceedings of the 45th Annual Design Automation Conference, Jun. 08-13, ACM, USA., pp: 630-633. DOI: 10.1145/1391469.1391630

Liu, J. And J.G. Delgado-Frias, 2007. A DAMQ shared buffer scheme for network-on-chip. Proceedings of the 5th IASTED International Conference on Circuits, Signals and Systems, Jul. 2-4, ACTA Press, Canada, pp: 53-58.