An Interconnect Channel Design Methodology for High Performance Integrated Circuits

Vikas Chandra, Anthony Xu, Herman Schmit and Larry Pileggi
Department of Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
{vchandra, acx, herman, pileggi}@ece.cmu.edu

Abstract

On-chip communication is becoming a bottleneck for high performance designs. Conventional interconnect design methodology does not account for architectures and/or communication schemes that require storage buffers (First-In-First-Out queues or FIFOs) in the interconnect channel. For example, FIFOs and flow-control are needed for Network-on-Chip, high performance ASICs and multiple clock domain designs. These IC implementation architectures require an efficient methodology to determine the size of the FIFOs in the channel since the FIFO sizes affect system performance. In this work we devised a methodology to size the FIFOs in an interconnect channel containing one or more FIFOs connected in series. We show that the sizing of the FIFOs in the channel is a function of system parameters such as data production rate and consumption rate, data burstiness, number of channel stages etc. and we also quantify their effect on performance. For a single clock design, we have developed an efficient algorithm which reduces the search space for the optimal sizing of the FIFOs in the channel.

1 Introduction

Modern large-scale chip design is faced with a number of profound problems caused by the scale and complexity of the designs. First, the required on-chip communication bandwidth is growing beyond that provided by standard on-chip busses. Second, the interconnect delay across the chip, even when appropriately buffered, exceeds the average clock period of the IP blocks. The ratio of global interconnect delay to average clock period will continue to grow.

In this paper we show that designing an interconnect channel for an architecture which requires storage buffers (First-In-First-Out queues or FIFOs) is not a trivial task. Significant research has been done in the area of optimizing wires for performance, but the challenges are different when the channel contains FIFOs. In this work we show that the interconnect channel design is a function of various system parameters. Among these parameters are the stalling behavior of the synchronous blocks, the data burstiness, amount of buffering etc. These system parameters need to be taken into account for an efficient channel design.

Network-on-Chip (NoC) has been proposed as a solution to the scalability problems of a bus-based system-on-chip designs [3, 9]. Briefly, NoCs can provide scalable interconnect bandwidth for a large number of IP blocks. Networks also provide structured pipelining and re-buffering of chip-scale interconnects. Networks also require compliance to a network interface, which increases re-usability and verification of the components. Global interconnects in an NoC require *buffering* and *flow-control*. Flow-control is a protocol to manage the data-flow in an interconnect which requires FIFOs. Sets of signals (like full and empty) are used to enable flow-control. To our knowledge, there has been no prior work in developing a design methodology for global interconnections in an NoC.

Buffering and flow-control are required even for high performance ASICs if two IP blocks communicate in bursts using transaction-based latency-insensitive protocols [6]. Although an NoC based system-on-chip differs from an ASIC in many ways, the challenges in designing an interconnect channel are very similar for both of them (if there is a need for buffering and flow-control in the ASIC design). In this work, we have developed an approach for an efficient channel design which is applicable to NoC interconnects as well as ASIC interconnects which require buffering and flow-control.

The rest of the paper is organized as follows. Section 2 explains the terms used in this paper and describes the problem with respect to an NoC as well as an ASIC perspective. Section 3 formulates the problem and discusses the theoretical and practical issues involved with the channel design. Section 4 explains the various system parameters which affect the channel design. Section 5 discusses our design methodology, experiments and our observations. Section 6 summarizes the work and concludes.

2 Background

Definitions of the terms commonly used in this paper.

- **FIFO:** A FIFO is a First-In-First-Out queue. The FIFO architecture used in this paper is a low-latency FIFO implemented synchronously [7]. Synchronous FIFOs are also referred as *elastic-buffers*.
- **FIFO size:** This refers to the maximum number of data elements a FIFO can store. This is sometimes referred as the *capacity* of the FIFO or the storage space available in the FIFO.
- **Channel:** A channel consists of one or more FIFOs connected in series.
- **Stage:** Each of the FIFOs in an interconnect channel is referred as a stage. For example, in a channel containing 3 FIFOs, there will be 3 channel stages.
- **Atomic FIFO:** When there is only one FIFO in the channel between two synchronous blocks, the FIFO is referred as an atomic FIFO.
- **Distributed FIFO:** A series connected (or tandem) network of FIFOs is referred as a distributed FIFO.

1530-1591/04 $20.00 (c) 2004 IEEE

2.1 Network-on-Chip Interconnects

Figure 1 shows a communication link in an NoC. The packets from the IP block **A** are sent to the IP block **B** through the link shown in the figure. The link goes through multiple routers and multiple FIFOs. FIFOs (or queues) are needed to support contention of routing resources. We have abstracted the connection between block A and block B as shown in Figure 1. In this work, our goal is to efficiently size each of the FIFOs in the channel between two synchronous blocks A and B, given a set of system parameters, such that the channel meets a given throughput requirement, ignoring traffic and congestion.

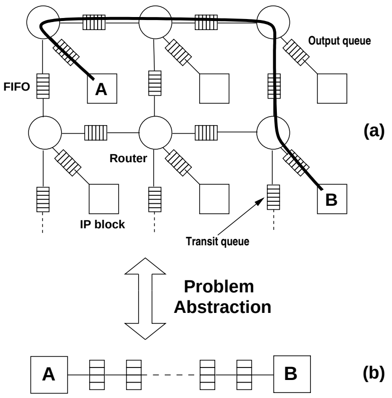


Figure 1. An abstraction of a communication link in an NoC

Some work has been done in the area of optimizing memory in a network switch fabric [21]. In [21], it was shown that there is a balance needed between transit queue memory (FIFOs between two router nodes) and output queue memory (FIFOs between IP block and router node) for better average performance for the whole chip. [21] considered traffic and congestion for variable length routes. This paper considers sizing of the FIFOs in a series connected network of FIFOs to optimize throughput between two fixed points in the network.

The routing technique is assumed to be minimal [14]. All the interconnections are optimized by efficiently sizing the FIFOs in the channel. If multiple connections share the same channel then the size of the FIFOs in that channel are chosen such that the throughput requirement of all the connections are satisfied. We will show that the performance of an interconnect channel is just dependent on the total size of the FIFOs in the channel and it is independent of how the sizes are distributed in the transit FIFOs and the output FIFOs. Hence, a FIFO size re-distribution can make sure that all the point-to-point connections in an NoC meet the throughput.

2.2 ASIC Interconnects

On chip communication is becoming a key performance bottleneck for high performance ASIC designs [4, 13, 20]. One solution to this design issue is to optimize the wires by inserting repeaters [2, 10], as shown in Figure 2. In [16], it was shown that conventional *delay-driven*, *wire-by-wire* planning paradigm cannot guarantee the reliability or feasibility of synthesized communication links. In [16], the authors propose a throughput-driven on-chip communication fabric synthesis methodology for system level interconnects and communication planning of SoCs.

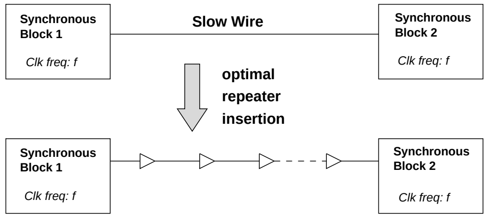


Figure 2. Optimal repeater insertion in wires

Even with the repeater insertion, the delay in the wires can exceed one clock period and multiple clock periods are needed to transfer the data from one synchronous block to another. There has been some work done to optimally insert latches in the wire to improve the throughput [8, 12, 17]. This scenario is referred as *wire-pipelining* and is shown in Figure 3.

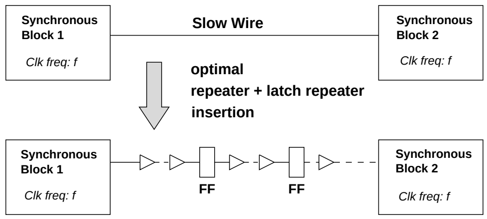


Figure 3. Optimal latch repeater insertion in wires

A pipelined wire will not be sufficient for a *bursty* ASIC interconnect. A bursty interconnect is characterized by an intermittent bursts of data production by the producer or an intermittent bursts of data consumption by the consumer. We need to have flow-control and elastic buffers. To avoid extra latency, the latches in a latch repeater inserted wire can be replaced by FIFOs to enable flow-control and provide storage, as shown in Figure 4. Also, extra wires (with appropriate buffering) need to be added for the flow-control signals, as shown in Figure 4.

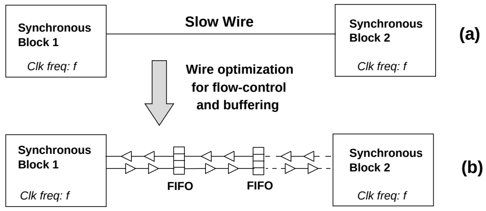


Figure 4. Wire optimization for an ASIC interconnect

This paper explores the design shown in Figures 1(b) and 4(b) in terms of an efficient sizing of the FIFOs in order to meet a throughput requirement. Although the scenarios shown in Figures 1(b) and 4(b) are in different context, they have the same design issues.

2.3 Our Contribution

In this paper, we analyze the design of an interconnect channel containing multiple channel stages. The work applies directly to NoC global interconnects as well as ASIC interconnects where there is a need to have FIFOs between IP blocks due to the bursty behavior of one or more blocks. There are three important research issues involved here:

1. How many stages of FIFOs are needed?
2. How to insert repeaters between the channel stages?
3. How to size each of the FIFOs in the channel?

Questions 1 and 2 have different answers depending on whether we are looking at NoC interconnects or ASIC interconnects. For an NoC interconnect, the number of channel stages will be an outcome of the minimal routing as shown in Figure 1. For an ASIC interconnect, the number of channel stages will be the same as the number of latch repeaters in the channel. Significant research has been done in the area of optimal repeater and latch repeater insertion for ASIC interconnects [2, 8, 10, 12, 17]. As mentioned earlier, for an ASIC interconnect, we replace the latch repeaters by FIFOs to avoid extra latency in the channel.

In this work, we have answered Question 3. The *size* of a FIFO refers to the number of storage elements in it. In this paper, we mainly focus on the sizing methodology of the FIFOs in the channel. It will be shown that the throughput of the interconnect channel is a function of how each of the FIFOs in the channel is sized. In this work we have explored the effect of various system design parameters on the sizing of FIFOs in a series connected network of FIFOs.

3 Problem Formulation

In this work, we look at the FIFO sizing methodology for an interconnect channel between two synchronous blocks. The interconnect channel could be from an NoC (Figure 1) or an ASIC (Figure 4).

Problem statement

Given a specified stalling behavior of the producer and the consumer, bursty behavior of the data and a number of channel stages, what are the sizes of each of the FIFOs in the channel connecting two synchronous blocks such that the channel meets the given throughput requirement?

3.1 Assumptions

Our motivation for this work is multiple clock domains. But the FIFO sizing problem has not been solved even for a single clock system. Hence, in this work we considered a single clock synchronous system. All the blocks in the system (the producer, consumer and intermediate FIFOs) run on a single clock. Results from this work will help us analyze a multiple clock domain system more efficiently. In this work all the events in the system occur at the positive edge of the clock making the system discrete-time. Also, in case a FIFO is full, the data in the previous stage waits until the next FIFO has enough space to take the data.

3.2 Queueing Network

The FIFO sizing problem can also be formulated as a queueing network problem. Queue sizing is a very old problem in the area of queueing network. A queueing network consists of an arbitrary connection of queues[1]. Figure 5 shows an example of a queueing network which is also called a *tandem* queueing network. A tandem queueing network is a linear array of queues. In Figure 5, **P** denotes the producer and **C** denotes the consumer.

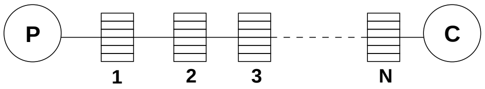


Figure 5. A tandem queueing network

Analyzing a queueing network is trivial if each of the FIFOs has an infinite size. Jackson's theorem [11, 15] specifies an easy way to analyze an arbitrary network of queues; the only condition being that all the queues have infinite-storage (i.e. each of the queues are M/M/1/∞ queues). In our case, each of the FIFOs in the channel has a fixed-storage (i.e. M/M/1/K queues). Analyzing a network of fixed-storage queues is an extremely difficult problem and has no closed form solution. There exists many computational methods to analyze network of fixed-storage queues [5, 11, 15, 19].

The queueing theory analysis is further complicated by the concept of *blocking*. Blocking is defined as an action taken when one or more queues in the network are full. In our case, if a FIFO is full, the data in the previous FIFO will wait until there is a space in the next FIFO. This is an example of *transfer-blocking* [1, 18]. Taking blocking into account makes the analysis of M/M/1/K queue networks even more complicated and computationally expensive. Further, the queueing theory approach applies to a continuous-time system. The problem we are trying to solve is in discrete-time, since all the events occur at the positive edge of clock. Applying queueing theory to a discrete-time system is not trivial. Hence, in this work we have developed a simulation based approach to efficiently size each of the FIFOs in a series network of FIFOs in the channel.

3.3 Issues in FIFO Sizing

In this work, our FIFO model is similar to what was proposed in [7]. As mentioned earlier, the size of a FIFO refers to the storage space available in it. The following factors affect the sizing of the FIFOs in the channel:

- Average data production and consumption rates
- Data bursts in the channel
- Throughput requirement
- Number of stages in the channel

Lemma 1 *If there are N stages of FIFO and M storage spaces are to be distributed among them, then the throughput of the system is independent of the way the M storage spaces are distributed among N FIFOs as long as each FIFO has at least 2 storage spaces.*

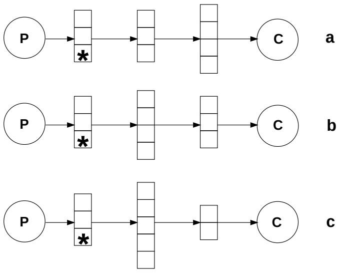


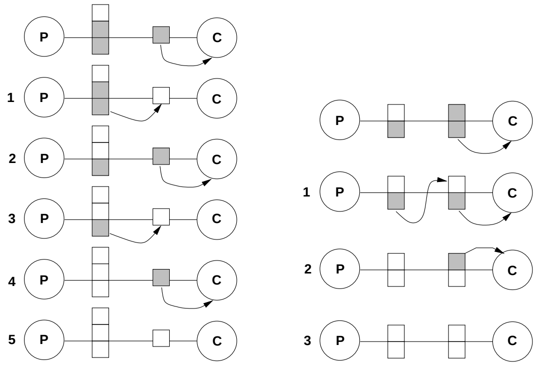
Figure 6. Effect of FIFO size distribution among stages

At a time *t*, let us say that there are *k* data elements in the channel. Since each of the FIFO element has more than one element, the i_{th} element will move ahead every clock cycle (assuming that the consumer accepts data every clock cycle). It will take the same amount of time for the i_{th} data element to reach the consumer no matter how the different FIFOs are sized as long as the total storage space is same and each FIFO at least has a capacity of 2. In other words, the time taken by an i_{th} data element in the queue to reach the consumer depends only upon the total number of storage elements between the i_{th} element and the consumer and not on the distribution of the storage elements among different stages. This is due to the fact that the channel is synchronous and the FIFOs are elastic [7]. Figure 6 shows an example of a 3 stage channel. For each of the three cases shown, the number

[1] Queueing discipline in our case is First-In-First-Out (FIFO).

of clock cycles needed for the $*$ element to reach the consumer will be the same. The reason being that for each of the cases in Figure 6, the total number of storage spaces between the $*$ element and the consumer is same (7) although the distribution of storage spaces between the second and third stage is different (3-4 for (a), 4-3 for (b) and 5-2 for (c)).

To understand the channel behavior when any of the FIFOs in the channel has just one storage space, refer to Figure 7. The total number of data elements in the channel is same in both Figures 7(a) and 7(b), but in Figure 7(a), the second stage has a size 1 FIFO, whereas in Figure 7(b) all the FIFOs are of size 2. It is obvious that it takes more number of clock cycles in Figure 7(a) compared to Figure 7(b) to read the same amount of data from the channel (since the buffers are elastic, the physical location of the entry to be read in the FIFO does not matter). We can note that even if both the FIFOs are of size 3, the behavior will be similar to that of Figure 7(b). In a nutshell, a FIFO of size 1 is expensive in terms of throughput because of the *bubble* propagation back and forth. Also, the location of a 1 size FIFO matters in the channel if there is a read-burst or a write-burst. Hence the lemma does not hold true for a channel where there is at least one FIFO of size 1.



(a) (b)

Figure 7. (a) Behavior of the channel with one 1 size FIFO (b) Behavior of the channel with all FIFO sizes more than 1

Lemma 1 assumes that the number of stages (*N*) is fixed. A total FIFO size of *M* can be distributed in many ways among the *N* FIFOs. Lemma 1 helps us analyze the throughput of the channel for each of the distributions.

Lemma 2 *If there is only one FIFO in the channel and M storage spaces are required to meet a certain throughput, then the total storage spaces needed for a distributed FIFO (for the same channel) to meet the same throughput are M or more.*

For a given throughput, an atomic FIFO is the most efficient. The reason being that both the producer and the consumer have direct access to all the elements of the FIFO, hence it leads to less overflow or underflow. In the case of a distributed FIFO, each of the stages is smaller and hence the channel performance goes down due to increased blocking. To meet the same performance as of an atomic FIFO, a distributed FIFO has to have same or more storage spaces. The need for extra storage spaces will depend on system parameters like data bursts, producer/consumer data rates etc.

In the next section, we will take an example of an interconnect channel and we will explore the dependence of FIFO sizing on different system parameters mentioned earlier in this section.

4 Analysis

In this work, we designed an interconnect channel using an ASIC methodology. We specified a maximum-frequency requirement from the channel. The length of the channel is given by the physical design information. Given the length of the channel and the maximum-frequency requirement, we optimized the channel using the methodology mentioned in [17]. For the channel design, we need to specify a power constraint also. Using the methodology mentioned in [17], we got a channel with 3 stages of latch repeaters. The design flow in [17] uses a SPICE engine to optimize the channel given a maximum-frequency requirement and power constraint. We need a power constraint to make sure that the interconnect synthesis tool in [17] does not pipeline the wire more than it is required. We replaced the latch repeaters in the channel with FIFOs to enable data bursts and flow-control in the channel as shown in Figure 8. The repeaters between the stages are not shown in Figure 8 since they do not affect the analysis as long as their delay is within one clock cycle. Although we considered a 3 stage channel in this work, we believe that the results from this work will generalize to an interconnect channel containing arbitrary number of stages. The FIFOs need to be efficiently sized to meet the throughput requirement from the channel. The next few sections explain the various system parameters which have an effect on the FIFO sizing.

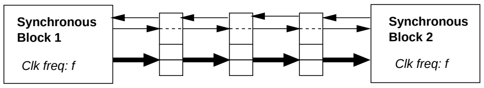


Figure 8. An example of a 3 stage channel

4.1 Production & consumption rates

We need buffering in the channel because the producer or the consumer can stall for an arbitrary number of clock cycles even though they work at the same clock frequency. The behavior of the queue will depend upon the relative data production and consumption rates. λ denotes the expected data arrival rate and μ denotes the expected data departure rate. Our definitions of λ and μ are different from the classical queueing theory definitions. λ of 0.5 means that at every positive edge of the clock there is a 50% chance that a new data will be generated (if there is no blocking). Similar analogy holds for the consumer as well.

4.2 Data bursts

A producer (consumer) can either write (read) data one data item at a time or it can generate a burst to write (read) *n* data items. The sizes of the FIFO in the channel are a function of the bursty behavior of the producer and the consumer.

4.3 Throughput requirement

Throughput is defined as the number of data items read per time unit . We defined throughput as the performance metric for the channel. For an interconnect channel, the system designer will specify a throughput requirement. Given a throughput requirement, we first analyze the FIFO size requirement assuming that there is just one FIFO (atomic FIFO) in the channel. Combining the atomic FIFO size needed to meet the throughput with the observations due to *Lemma 1* and *Lemma 2* helps to prune the search space for the actual channel with distributed FIFOs.

Figure 9 shows the relation between throughput and size of an atomic FIFO assuming that λ is 0.5 and μ is 0.5. The throughput is plotted for different FIFO sizes and as is expected the curve flattens out beyond a certain FIFO size. The throughput of a channel is dependent on the FIFO sizes, but as the FIFO sizes increase the throughput saturates. A throughput threshold is chosen which gives the region of feasible FIFO size in case of an atomic FIFO. In the realistic case (distributed FIFO) such as what is shown in Figure 8, we try to meet the same throughput requirement as was for an atomic case. We will also show that the total FIFO size required for a distributed FIFO is more than the FIFO size required for an atomic FIFO (for the same throughput) and the requirement of additional storage spaces is a function of system parameters described in this section.

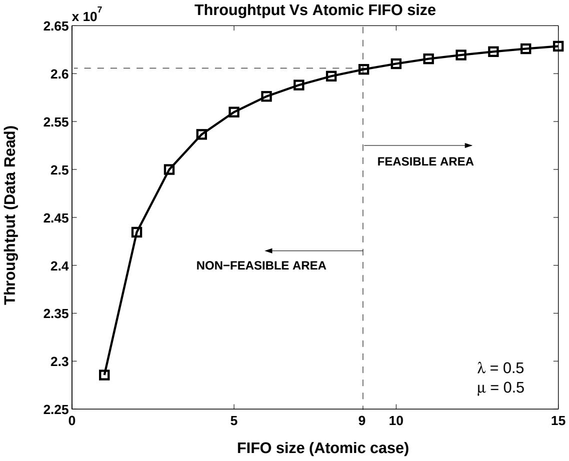


Figure 9. Throughput of an atomic FIFO

5 Experiments and Results

We used a software simulator written in C to explore the sizing of the FIFOs in the channel. Our simulator is capable of simulating any number of channel stages as well as it can generate parameterizable data-bursts with a given probability (for both the producer and the consumer). The algorithm to size a distributed FIFO channel is as follows:

1. *Given the system parameters, analyze the channel to find out the atomic FIFO size (M) which meets the throughput requirement.*
2. *For the distributed FIFO channel, explore the total FIFO size from M to M+δ (δ will vary with the system parameters).*
3. *Find out the minimum total FIFO size which meets the throughput requirement.*

Lemma 2 decreases the search space by limiting the search space from M to M+δ. The value of δ will vary with system parameters and can be tuned for a particular channel. From our experiments, we found that an approximate value of δ for a 3 stage channel is given by:

$$\delta \approx \frac{M}{2} + log_2(\text{size of data bursts}) \qquad (1)$$

Using *Lemma 1*, we can distribute the total FIFO size in any manner between the FIFOs in the channel, as long as each of the FIFOs has at least 2 storage spaces. *Lemma 1* further reduces the search space which can otherwise become huge.

We have done experiments with two distinct sets of system. One is a system where λ and μ are balanced, and for the other system the λ and μ values are skewed with respect to each other. The results are distinctly different for both the systems.

5.1 Balanced λ and μ

For this example, we chose λ as 0.5 and μ as 0.5. Our base line case is given by the curve shown in Figure 9. The atomic FIFO size needed to meet the throughput was 9. The data bursts size was limited to 1 and δ was chosen to be 5 using Equation 1 (M = 9 and size of data bursts = 1). Since we chose δ as 5, the FIFO size exploration (as mentioned in the algorithm earlier) for the distributed case will be from 9 to 14. Figure 10 compares the throughput of an atomic FIFO of size 9 versus a distributed FIFO with varying total sizes. The optimal solution in this case is 10.7 but the smallest feasible total FIFO size is 11. To meet the throughput of an atomic FIFO of size 9, the total size of the distributed FIFO has to be 11 for this example. This is in accordance with *Lemma 2*.

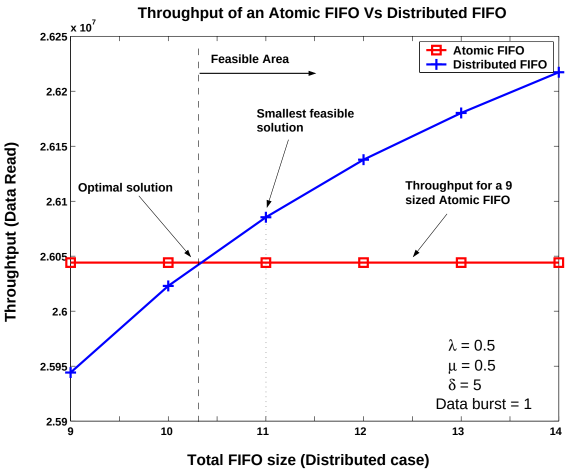


Figure 10. Throughput of an atomic FIFO Vs distributed FIFO

We characterized the same system (λ=0.5, μ=0.5) with varying size of data bursts. The results are shown in Figure 11. For each size of data bursts (1, 4, 8), the atomic FIFO size was 9 and we found out the optimal total size needed for a distributed FIFO to meet the performance for each case. It is interesting to note that the optimal size needed for the distributed FIFO goes up linearly with the size of data bursts. The value of δ also goes up as the data bursts size

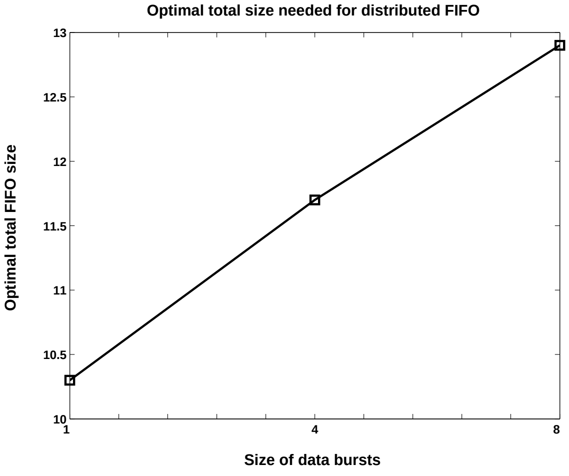


Figure 11. Optimal total size for distributed FIFO wrt size of data bursts

increases. This is in accordance with the Equation 1. As the size of data bursts increases, the total FIFO size for the channel needs to be increased to maintain the same throughput. The reason being that blocking (Section 3.2) increases with increasing data bursts size and hence the FIFO sizes need to be increased to compensate for the loss of throughput due to blocking.

5.2 Skewed λ and μ

We also considered systems where the production rate and the consumption rate are not balanced. The configurations shown in Table 1 were considered. When the production rate and the consumption rate are skewed, the optimal total size for a distributed FIFO is same as that of an atomic FIFO. For example, the throughput of a distributed FIFO with a total size of 9 is same as that of an atomic FIFO of size 9. This behavior is independent of the size of data bursts. The reason for this is that when the λ's and μ's are so skewed, the sizes of the FIFOs in the channel does not matter that much. All that is needed for such a channel is flow-control with minimal storage space.

Experiment	λ	μ	Data burst size
1	1	0.2	1
2	0.2	1	1
3	1	0.2	8
4	0.2	1	8

Table 1. Experiments with skewed λ's and μ's

6 Conclusions and Future Work

In this paper we devised a methodology to size the FIFOs in an interconnect channel containing series connected network of FIFOs. This problem is applicable to Network-on-Chip interconnects as well as high performance ASIC interconnects. The FIFO sizing is a function of various system parameters. We developed an algorithm to efficiently size the FIFOs in the channel. These are the conclusions from this work:

- A FIFO size of 1 is inherently bad for throughput because of the delay involved in the bubble propagation back and forth.
- If each of the FIFOs in a distributed FIFO channel has size more than 1, it does not matter how one distributes the total FIFO size among different FIFOs in the channel.
- For the same channel, the total FIFO size required for a distributed FIFO is equal to or more than the size required for an atomic FIFO to have the same throughput.
- When the values of λ and μ are similar, the overhead (in terms of extra storage spaces) for the distributed FIFO case increases with the size of data bursts. The value of δ also need to be increased with increasing size of data bursts.
- When the values of λ and μ are skewed, there is no overhead (in terms of extra storage spaces) for the distributed FIFO when compared to the atomic FIFO (for the same throughput).

Using the observations mentioned above, our FIFO sizing algorithm efficiently sizes the FIFOs in the channel to meet the throughput requirement. *Lemma 1* and *Lemma 2* help reduce the search space for optimal sizing.

In this work, all the blocks work on a single clock. For a frequency islands based design, the trade-offs in the distributed FIFO sizing are going to be different. In our future work, we plan to extend this work to a multiple clock system where the producer and the consumer operate at different clocks. Few of the issues which need to be considered for frequency islands based designs are: choice of intermediate clocks in the channel and the distribution of total FIFO sizes between each of the FIFOs in the channel. Since each of the FIFOs could be operating at different clocks, the sizing of the FIFOs in the channel will be of key importance from performance and power standpoint. Further, we are also looking at designing asynchronous pipelines for an asynchronous system and specifically we are looking at asynchronous FIFO sizing issues. Our future work will address these issues.

References

[1] I. F. Akyildiz, ”Mean Value Analysis for Blocking Queueing Networks,” *IEEE Transactions on Software Engineering*, 1988.

[2] K. Banerjee and A. Mehrotra ”A power-optimal repeater insertion methodology for global interconnects in nanometer designs,” *IEEE Transactions on Electron Devices*, Nov. 2002.

[3] L. Benini, G. De Micheli, ”Network on Chips: A New SoC Paradigm,” *IEEE Computer*, 2002.

[4] M. T. Bohr, ”Interconnect Scaling - the real limiter to high performance ULSI,” *Proceedings of IEDM*, 1995.

[5] C. Buyukkoc, ”An Approximate Method for Feedforward Queueing Networks with Finite Buffers: A Manufacturing Perspective,” *IEEE Intl. Conference on Robotics and Automation*, 1986.

[6] L. P. Carloni, K. L. McMillan, A. Saldanha and A. L. Sangiovanni-Vincentelli, ”A Methodology for Correct-by-Construction Latency Insensitive Design,” *ICCAD*, 1999.

[7] T. Chelcea and S. Nowick, ”Robust Interfaces for Mixed-Timing Systems with Application to Latency-Insensitive Protocols,” *IEEE Design Automation Conference*, 2001.

[8] P. Cocchini, ”Concurrent Flip-Flop and Repeater Insertion for High Performance Integrated Circuits,” *ICCAD*, 2002.

[9] W. J. Dally and B. Towles, ”Route Packets, not wires: On-chip interconnection networks,” *Proc. of 38th Design Automation Conference*, June 2001.

[10] J. A. Davis et.al, ”Gigascale Integration (GSI) Interconnect Limits and N-Tier Multilevel Interconnect Architectural Solutions,” *Proc. SLIP*, 2001.

[11] D. Gross and C. Harris, ”Fundamentals of Queueing Theory,” *John Wiley & Sons*,1998.

[12] S. Hassoun and C. J. Alpert, ”Optimal Buffered Routing Path Constructions for Single and Single and Multiple Clock Domain Systems,” *IEEE Transactions on Computer-Aided Design*, 2003.

[13] R. Ho, K. W. Mai and M. Horowitz, ”The future of wires,” *Proceedings of the IEEE*, April 2001.

[14] J. Hu and R. Marculescu, ”Exploiting the Routing Flexibility for Energy/Performance Aware Mapping of Regular NoC Architectures” *Design Automation and Test in Europe*, 2003.

[15] L. Kleinrock, ”Queuing Systems - Volume 1: Theory,” *John Wiley & Sons*, 1975.

[16] T. Lin and L. Pileggi, ”Throughput-Driven IC Communication Fabric Synthesis,” *ICCAD*, 2002.

[17] T. Lin, ”Ph.D. work in Progress,” *Dept. of ECE, Carnegie Mellon University, USA*.

[18] D. Manjunath et. al., ”QNAT: A Graphical Tool for the Analysis of Queueing Networks,” *IEEE TENCON Intl. Conference*, 1998.

[19] J. M. Smith and N. Chikhale, ”Buffer Allocation for a Class of Nonlinear Stochastic Knapsack Problems,” *Technical Report, Dept. of Industrial Engg. and Operations Research, Univ. of Massachusetts, Amherst*,1995.

[20] D. Sylvester and K. Keutzer, ”Getting to the bottom of deep submicron,” *International Conference on Computer Aided Design (ICCAD)*, 1998.

[21] D. Whelihan and H. Schmit, ”Memory Optimization in Single Chip Network Switch Fabrics,” *Design Automation Conference*, 2002.