0.0in

Tuning and Verification of Simulation Models for High Speed Interconnect Fabrics

M. Manzke, S. Kenny and B. Coghlan
Computer Science Department
Trinity College Dublin
Ireland

O. Lysne
Department of Informatics
University of Oslo
Blindern, N-0316 Oslo
Norway

Abstract ***In this paper we demonstrate how the non-intrusive acquisition of interconnect trace data and a subsequent data analysis can be used to accurately tune the definition of interconnect loads and the parameterisation of interconnect simulation models. High speed interconnects or system area networks are the principal components of a compute cluster that transform stand alone computers into a cluster. The design of such interconnect fabrics may be assisted through performance prediction. This prediction is accomplished through simulation if the model's parameterisation reflects the physical fabric and realistic load descriptions are provided. This paper describes the software and hardware tools necessary to accurately tune a model that predicts the performance of an IEEE-1596 Scalable Coherent Interface (SCI) interconnect network. The simulation model is implemented in a parameterised SCI node model in OPNET Modeler. SCI is one of the enabling interconnect technologies for high performance computing on desktop and server Clusters. This implementation proves the concept and demonstrates its suitability for other switched fabric interconnects such as InfiniBand^{TM}.***

Keywords: Simulation Tuning, Clusters Interconnects, SCI Simulation model, Non-Intrusive Tracing, Trace Analysis, Trace-database

1 Introduction

A detailed interconnect simulation can aid in the design of complex fabrics by predicting the performance of the proposed design. A number of fabric configurations may be evaluated without the need to actually build the systems. A simulation may be the preferred option if the system is too large, making it too expensive to build just for test purposes, or because the hardware cycle time is too high for the results of the evaluation to arrive in time for the design.

A simulation is only as accurate as its simulation model. A simulation model must be tuned and verified in order to guarantee that the model reflects the real physical system behavior. The verification and tuning requires information about the true temporal behavior of the physical interconnect. This information can be extracted from non-invasively measured interconnect traffic trace data. The trace data must be acquired non-invasively for it to be accurate. Such data may be stored in a trace database and a particular subsets can then be statisticaly analysed.

It is the non-intrusive character of the measurement that allows us to analyse the true temporal behavior of the system under test. The analysis process requires the extraction of particular trace data subsets from a large set of non-intrusively sampled trace data. This filter process can only be performed if trace information has been decoded to allow the identification of particular attributes. The decoding step converts a large set of raw trace data into a set of structured trace data that makes all the attributes, for a subsequent filter and analysis process, explicit. These data structures are suitable for storage in relational databases. A relational database engine provides the mechanism needed to implement the trace data filter process through SQL-Queries [1].

The non-intrusive measurement of interconnect traffic is achieved with two stages of trace instrumentation. The following hardware options represent the first stage of instrumentation: a *SCITRAC* [2] link tracer, a *SCIview* [3] field programmable tracer instrument or an adapter card that observes interface traffic through snooping on the *Blink*^{TM} [4], Dolphin's implementation of the IEEE standard SCI transfer cloud [5].

The second stage collects trace data from this instrumentation. There are two options available for this purpose: an instrument developed at Trinity College Dublin [1, 6, 7, 8] or a standard commercially available logic analyser.

Once the traces have been collected by the instrumentation, the raw trace data can be decoded to structured information that is stored into a relational database. The trace database and the trace decoding, which was designed and implemented at Trinity College, is specifically intended as a means for a fine grained analysis of large quantities of trace data.

Figure 1 shows the components of the simulation, measurement and analysis system. The individual components are based on previous work on non-intrusive trace acquisition [1, 2, 3] and simulation of high-speed-interconnect traffic [9] at the Department of Physics and the Department of Informatics at the University of Oslo and at Trinity College Dublin.

The paper is organized as follows: In Section 2 we present the parameterized SCI node model developed at the University of Oslo. This model has been used to evaluate SCI-topologies consisting of 20 ringlets, and up to 96 nodes. The model development was done within the framework provided by the *OPNET Modeler* [10]. A standard distribution of OPNET contains models of most standard communication technologies like Ethernet, FDDI, ATM, etc. This facilitates easy integration of our node with models of other technologies, enabling simulation of heterogeneous systems with very limited additional development.

In Section 3 we present mechanisms that allow the extraction of statistical information from a trace-database for the generation of realistic system loads. These loads are used in the simulation model described in Section 2.

2 SCI Simulation Model

We describe a model of an SCI-node that is implemented in *OPNET-modeler*. OPNET was originally developed at MIT, and was introduced in 1987 as the first commercial network simulator. The node model is designed to simulate SCI at the packet level for increased simulation efficiency. Still, we achieve symbol level accuracy, by exploiting the fact that the time interval between the arrival of the first and the last symbol of a packet is a function of its length. The model is object oriented in the sense that there are separate modules (objects) for units like the input and output buffers, the bypass FIFO, the stripper, the multiplexer at the output end, etc.. It contains a full implementation of the go-bit based flow control, as well as the A-B aging scheme of the retry protocol. The formation of ringlets or bigger topologies consisting of both rings and switches can to a large extent be done through drag and drop functionality.

The usefulness of a simulation model is dependent on the statistics it assumes for the physical system. This is connected to the statistics it is able to generate when it runs. In our SCI model we have implemented a comprehensive set of "points of measurements". Figure 3 shows where these points are located in the logical model. The points of measurements are:

1. The number of packets held in the output buffer. A new sample is generated at each

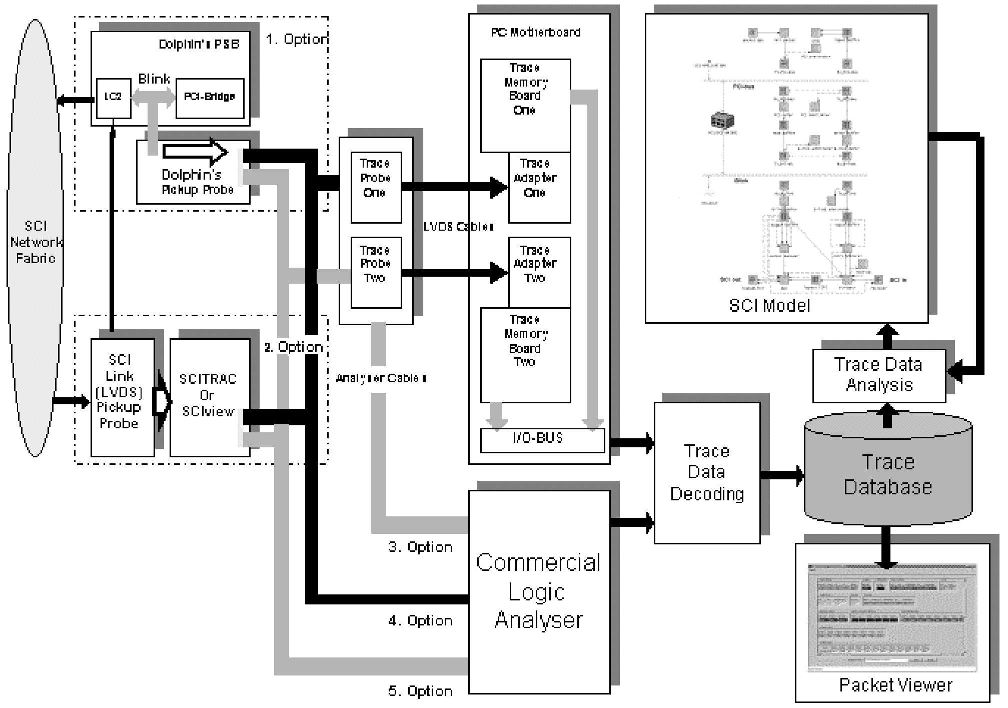

Figure 1: Simulation, Measurement and Analysis System Overview

packet insertion or removal from the output buffer.

2. The number of packets held in the active buffer. A new sample is generated at each packet insertion or removal from the active buffer.

3. Output throughput, which is measured in *symbol per clock cycle*. The output throughput is collected whenever a new packet arrives at the output buffer.

4. Bypass throughput, which is measured in *symbol per clock cycle*. The bypass throughput is collected whenever a new packet arrives at the output buffer.

5. Node throughput, which is measured in *symbol per clock cycle*; *Node throughput = Output throughput + Bypass throughput*.

6. Idles between arriving packets. This measures the time between arriving packets at the node's input link, which indicates the link's load. The link is run at maximum capacity if there is only one idle between arriving packets. In this case we say that the **load factor** equals 1.

7. Number of send packets held in the input buffer. A new sample is generated at each packet insertion or removal from the input buffer.

The SCI interface simulation generates SCI packets in the packet_gen module. Figure 2 shows the individual simulation components. The packet_gen module is shown above the PCI-bus. The Model assigns the packet type randomly and the inter-arrival time of the packets is dictated by a Probability Density Function (PDF).

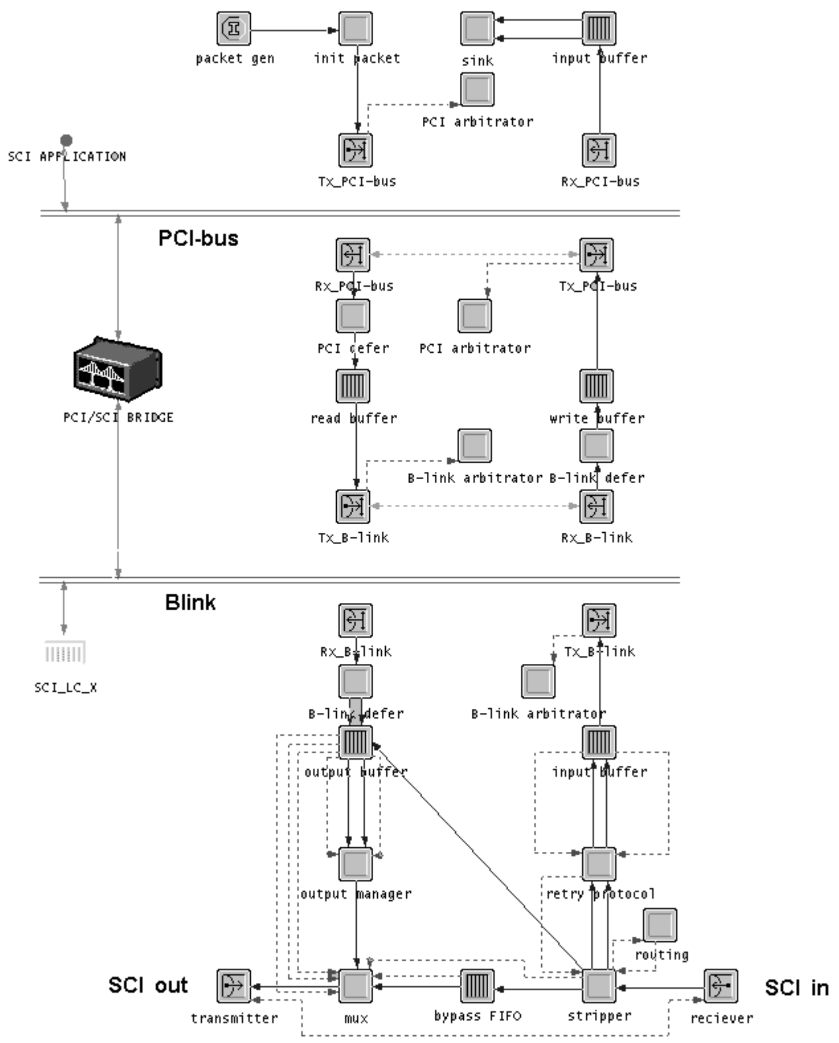

Figure 2: SCI node OPNET model including PCI-bridge and Blink

3 SCI Simulation Model Tuning

The relational trace database stores SCI trace data including timestamps. These timestamps are associated with individual packets. Timing details are appended to the packet during the non-intrusive acquisition of trace data. The trace database makes these absolute timestamps available for further analysis. Consequently trace data provides accurate information about the temporal behavior of the system under test. Time stamped packets are essential for a statistical description of the system's behavior.

The system provides the means to monitor interconnect traffic at three different locations: snooping on the link cable SCI IN and/or SCI OUT and/or the *Blink*^{TM}. Figure 2 shows the three snoop targets in an OPNET representation of the SCI interface. Traces acquired from these three locations complement each other by providing different subsets of the full set of ringlet and *Blink*^{TM} traffic that passes through the link controller. The link controller receives ringlet packets on the SCI IN port. These packets may be addressed to this node and routed into the link controller or forwarded to the SCI OUT port if addressed to a different node. The link controller processes the packet

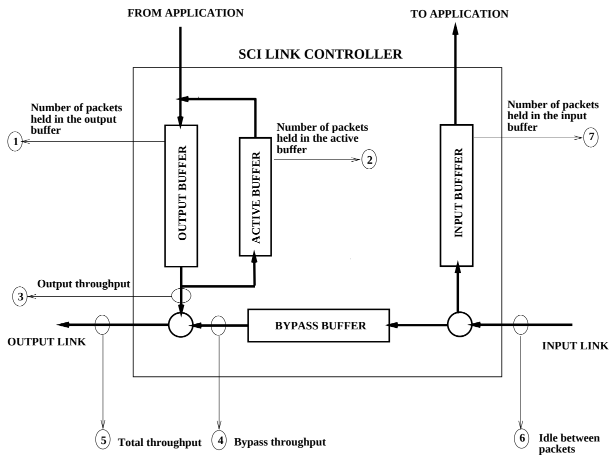

Figure 3: The points of measurement where the statistic is collected for an SCI node. For convenience the source and the destination device for the input and the output buffer are called 'TO APPLICATION' and 'FROM APPLICATION'. In a real SCI node the buffers are physically connected to a bridge.

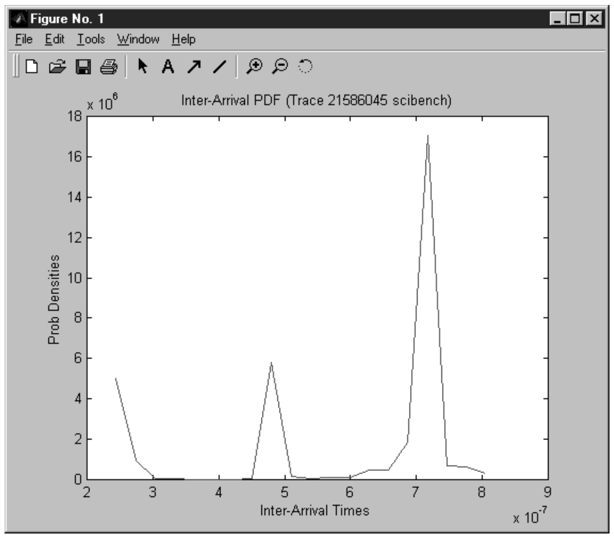

Figure 4: Propability density functions

further by decoding the packet type and processing it accordingly, e.g. a echo packet acknowledges a previous transaction and will be absorbed in the link controller but send-request or send-response packets are encapsulated and forwarded onto the *Blink*^{TM} because they require the assistance of the PCI/SCI Bridge. Send-request or send-response also require the generation of echo packets to acknowledge the transaction to the source node. This echo packet is transmitted on the SCI OUT port.

The link controller also receives encapsulated send-requests or send-responses on its *Blink*^{TM} port from the PCI/SCI Bridge and transmits them on the SCI OUT port. This description of link controller transactions is by no means exhaustive but should demonstrate that monitoring on the *Blink*^{TM} is restricted to the subset of send-request and send-response packets whereas snooping on the link provides information about link level related transactions. See [5] for full details. Due to the nature of unidirectional link traffic, monitoring on a single link cable limits the visibility of node transactions, e.g. snooping on the SCI OUT cable enables the observation of a send-request packet but we will neither see the associated echo packet nor a send-response packet from the responding node. We will see the echo packet that acknowledges the reception of a send-response packet. This packet also completes this transaction. On the other hand monitoring the *Blink*^{TM} would allow the observation of both the send-request and the send-request packets but not the echo packets nor any passing traffic.

The above implies that the most comprehensive tracing would require the use of three trace data acquisition channels and a synchronized trigger mechanism. The trace database actually allows one to relate 2 + n traces that are monitored at different locations whether at the same node or remote notes in the SCI fabric. Queries for particular trace data attributes, e.g. transaction ID in conjunction with time constraints, can extract packets related to a specific transaction.

Let us now discuss a trace data analysis that takes advantage of the completeness of the information held within the trace database and the relational operations that can be performed upon it. We extract specific subsets of trace data in order to generate probability density functions that can be used in the SCI simula-

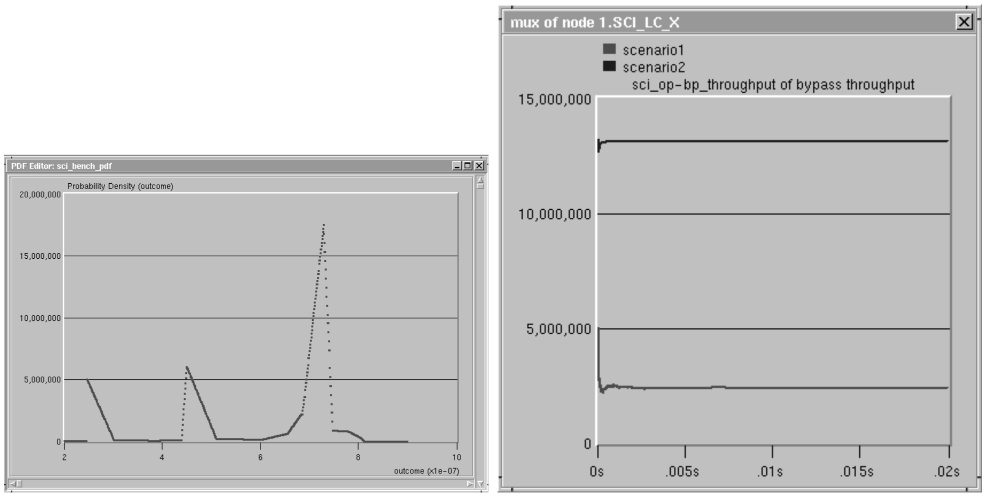

Figure 5: left: Load definition, right: model output

tion model described in section 2. The SCI interface simulation generates SCI packets in the packet_gen module. Figure 2 shows the individual simulation components. The packet_gen module is shown above the PCI-bus. As previously indicated the model assigns the packet type randomly and the inter-arrival time of the packets is dictated by a Probability Density Function (PDF). Currently the SCI model uses PDFs provided by the OPNET package but the software allows the definition of PDFs by the user. For this purpose we extract through a SQL query a relevant subset from the trace database and compute the PDFs from timestamps that are associated with each packet. The default setup uses a uniform PDF with limits set to 6.0E-7 and 6.0E-6 seconds. Our analyses show a strong deviation from this assumption, as shown in Figure 4. The result of our SQL query provides the input to a statistics module that will calculate the Probability Density Function for this subset of trace information. This function represents the probability density of packets generated by real system loads. When we replace the default PDF with our PDF we therefore stimulate the model with realistic system loads. The left hand graph of Figure 5 shows our PDF derived from measurements shown in Figure 4.

In a further step we have stimulated the SCI model with both a realistic system load and the default uniform PDF. The right hand graph of Figure 5 compares the resulting system throughput predicted by the SCI model. The significance of this is yet to be established.

4 Conclusion

The non-invasive acquisition of interconnect traffic allows analyses of the true temporal behavior of compute clusters. The advantage of offline query-based filtering of traces as opposed to real-time filtering within the trace acquisition instrument, is the ability to analyse numerous different aspects of the same traces without the need to re-perform the trace acquisition. Bear in mind that repeated acquisitions may not lead to similar results, thereby creating uncertainty about the correctness of the analysis. The method described here generates realistic system load statistics for the SCI model. Furthermore, it enables the tuning and the verification of the SCI model's parameterization. We are actively enhancing this framework, and have begun evaluating its extension to support similar interconnects, in particular *InfiniBand*^{TM} [11, 12].

Acknowledgments

The development has been supported by the European Commission through the project P25257 SCIEurope. The overall framework described here was not a planned outcome of this project, but includes individual results of the project, and has been a direct result of the very friendly collaborations that evolved. We thank everyone who has been involved in this most successful project.

References

[1] Michael Manzke and Brain Coghlan. Non-intrusive deep tracing of sci interconnect traffic. In *Conference Proceedings of SCI Europe '99*, 1999.

[2] Bernhard Skaali, Inge Birkeli, Baard Nossum, and David Wormald. Scitrac - an lsa preprocessor for sci link tracing. In *Scaleble Coherent Interface: Technology and Application*, 1998.

[3] Bernhard Skaali, Baard Nossum, Inge Birkeli, and David Wormald. Sciview-sci test, verification and monitoring instrument. In *Conference Proceedings of SCI Europe '99*, 1999.

[4] Dolphin. *A Backside Link (Blink) for Scalable Coherent Interface (SCI) nodes.*

[5] The Institute of Electrical and Inc. Electronics Engineers. *IEEE Standard for Scalable Coherent Interface (SCI).* The Institute of Electrical and Electronics Engineers, Inc., 1993.

[6] Brain Coghlan, Michael Manzke, Erich Barnstedt, Ronon Cunniffe, and Jonathan Dukes. Deep trace dt200.1, prototype tracer. Technical report, Trinity College Dublin, 1998.

[7] Brain Coghlan and Michael Manzke. Prototype trace probe and probe adapter. Technical report, Trinity College Dublin, 1999.

[8] Michael Manzke and Brain Coghlan. Prototype trace software. Technical report, Trinity College Dublin, 1999.

[9] Gunnar Rønneberg and Olav Lysne. An opnet-based simulation model of sci-nodes. In *Conference Proceedings of SCI Europe '99*, 1999.

[10] OPNET Modeler. *OPNET Modeler manuals.* MIL 3, Inc. 3400 International Drive NW, Washington DC 20008 USA, 1989-1997.

[11] InfiniBand$^{\text{TM}}$. *InfiniBand^{TM}Architecture Specification - General Specification.* InfiniBand Trade Association, 2000.

[12] InfiniBand$^{\text{TM}}$. *InfiniBand^{TM}Architecture Specification - Physical Specification.* InfiniBand Trade Association, 2000.