May 11, 2007

Effects of Adaptive Wormhole Routing in Event Builder Networks

R. Moser

CERN, Geneva, Switzerland

Vienna University of Technology, Vienna, Austria

CMS Collaboration

Abstract

The data acquisition system of the CMS experiment at the Large Hadron Collider features a two-stage event builder, which combines data from about 500 sources into full events at an aggregate throughput of 100 GByte/s. To meet the requirements, several architectures and interconnect technologies have been quantitatively evaluated. Both Gigabit Ethernet and Myrinet networks will be employed during the first run. Nearly full bi-section throughput can be obtained using a custom software driver for Myrinet based on barrel shifter traffic shaping.

This paper discusses the use of Myrinet dual-port network interface cards supporting channel bonding to achieve virtual 5GBit/s links with adaptive routing to alleviate the throughput limitations associated with wormhole routing. Adaptive routing is not expected to be suitable for high-throughput event builder applications in high-energy physics. To corroborate this claim, results from the CMS event builder preseries installation at CERN are presented and the problems of wormhole routing networks are discussed.

Presented at 15th IEEE NPSS Real Time Conference 2007, Batavia, Illinois, USA, April 29 – May 4, 2007

Submitted to IEEE NPSS Real Time Conference

Submitted to IEEE Transactions on Nuclear Science

Effects of Adaptive Wormhole Routing in Event Builder Networks

G. Bauer[9], V. Boyer[5], J. Branson[6], A. Brett[5], E. Cano[5], A. Carboni[5], M. Ciganek[5], S. Cittolin[5], S. Erhan[5,7], D. Gigi[5], F. Glege[5], R. Gomez-Reino[5], M. Gulmini[2,5], E. Gutierrez Mlot[5], J. Gutleber[5], C. Jacobs[5], J.C. Kim[4], M. Klute[9], E. Lipeles[6], J.A. Lopez Perez[5], G. Maron[2], F. Meijers[5], E. Meschi[5], R. Moser[1,5], S. Murray[8], A. Oh[5], L. Orsini[5], C. Paus[9], A. Petrucci[2], M. Pieri[6], L. Pollet[5], A. Racz[5], H. Sakulin[5], M. Sani[6], P. Schieferdecker[5], C. Schwick[5], K. Sumorok[9], I. Suzuki[8], D. Tsirigkas[5], J. Varela[3,5]

[1]Vienna University of Technology, Vienna, Austria
[2]INFN - Laboratori Nazionali di Legnaro, Legnaro, Italy
[3]LIP, Lisbon, Portugal
[4]Kyungpook National University, Daegu, Kyungpook, South Korea
[5]CERN, Geneva, Switzerland
[6]University of California San Diego, La Jolla, California, USA
[7]University of California, Los Angeles, California, USA
[8]FNAL, Batavia, Illinois, USA
[9]Massachusetts Institute of Technology, Cambridge, Massachusetts, USA

Presented by Roland Moser (roland.moser@cern.ch)

***Abstract*–The data acquisition system of the CMS experiment at the Large Hadron Collider features a two-stage event builder, which combines data from about 500 sources into full events at an aggregate throughput of 100 GByte/s. To meet the requirements, several architectures and interconnect technologies have been quantitatively evaluated. Both Gigabit Ethernet and Myrinet networks will be employed during the first run. Nearly full bi-section throughput can be obtained using a custom software driver for Myrinet based on barrel shifter traffic shaping.**

This paper discusses the use of Myrinet dual-port network interface cards supporting channel bonding to achieve virtual 5GBit/s links with adaptive routing to alleviate the throughput limitations associated with wormhole routing. Adaptive routing is not expected to be suitable for high-throughput event builder applications in high-energy physics. To corroborate this claim, results from the CMS event builder preseries installation at CERN are presented and the problems of wormhole routing networks are discussed.

I. Introduction

The architecture blueprint of the data acquisition (DAQ) system for the CMS experiment [1]-[3] at the CERN LHC pp collider is sketched out in Fig. 1. A high performance network connects 512 *readout units* (RUs), via a switch fabric, to 512 *builder units* (BUs). The RUs receive event data fragments from detector elements at a first level trigger peak rate of 100 kHz and buffer the fragments for up to one second. The expected average event size is 1 MByte (log-normal distributed), corresponding to event fragment sizes of 2 kBytes. With events of this size, the 512 × 512 event building network requires an effective aggregate throughput of 100 GByte/s. A software trigger running in *filter units* (FUs) connected to the BUs further reduces the rate to about 100 Hz. The remaining accepted events are forwarded to permanent storage for off-line analysis. The *event manager* controls the event flow. It broadcasts the first level trigger information to all RUs and assigns the destination BU to each event. This message flow is routed through the event-building network.

Event building traffic is highly systematic as multiple sources compete for the same destination. Depending on the switching technology, the result may be reduced throughput, increased latency and/or loss of data. An appropriate destination assignment algorithm and traffic shaping can reduce these effects [4]. Traffic shaping controls the outgoing messages before their submission to the network. One such technique is the *barrel shifter* [5], [6]. In this scheme, sources are synchronized to emit fragments in time slots in such a way that no two sources send to the same destination during the same time slot and all sources regularly send to all destinations in a cycle. This approach is efficient for fixed size data blocks.

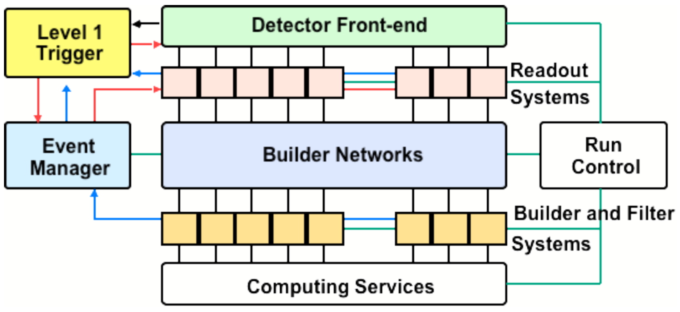

Fig. 1. The CMS data acquisition architecture.

Up to now Gigabit Ethernet [7], Myrinet [8], [9] and a custom *barrel shifter* implementation based on Myrinet [5], [6] have been analyzed. This paper presents an evaluation of the next generation Myrinet traffic shaping technology found in the *Myrinet Express* [10,11] message-passing library. It implements 2-port channel bonding and adaptive wormhole routing. To determine properties and limitations of this communication subsystem we carried out studies with standalone test programs and *event builder* applications.

II. Myrinet

Myrinet is a low-latency, wormhole routing based supercomputer interconnect [11]-[13]. The signal rate of a single Myrinet link is 2.5 GBit/s and corresponds to a maximum throughput of 2 GBit/s due to 8B10B coding [14].

Myrinet network interface cards (NIC) feature a programmable RISC CPU that makes this technology attractive for customization in environments constrained by high-performance communication requirements. In addition, the card features three controllable DMA engines. The CPU is programmed in the C language using a host to NIC cross compiler. The compiled program called MCP (Myrinet Control Program) is a real-time application that is uploaded to the NIC through the host PCI interface.

Myrinet Express (MX) is a protocol on top of Myrinet hardware. MX consists of two parts, an MCP firmware residing in the network interface cards processor and a host based user space library. The protocol implements channel bonding to improve throughput and to reduce the effects of *Head of Line (HoL) Blocking* [15].

Connecting each input port to each output port would result in N^2 connections. To limit the number of physical links, Myrinet uses cost and performance effective Clos switches [16]. They have a minimum bisection of N for N senders and receivers. Hence, this switch technology permits achieving full bisectional throughput. In addition, this type of network can tolerate failing internal links and switches [17].

Myrinet is a source-routed network. Each connected host needs a complete map of the network to be able to route the packets to the correct destination. A daemon process running on each computer called *mx_mapper* scouts out the network at regular intervals [18]. It performs leader election based on MAC addresses. The election winner continues to map the network using breadth-first search. Once fully mapped, a binary lookup tree is generated and distributed to all computers in the network.

Based on this information hosts generate routes using Dijkstra's breadth-first search algorithm [19] that finds the routes with the lowest number of hops. An improved algorithm is implemented that chooses the successor randomly to generate different routes on different hosts for further reducing the number of potentially conflicting routes.

Myrinet Express uses *adaptive dispersive routing* [20]. Based on the assumption that traffic follows repetitive patterns, a route is changing only when blocking is detected. This aims at reducing the Clos internal link contention. The algorithm allows finding non-blocking routes in Clos-based networks in case of rarely changing communication patterns.

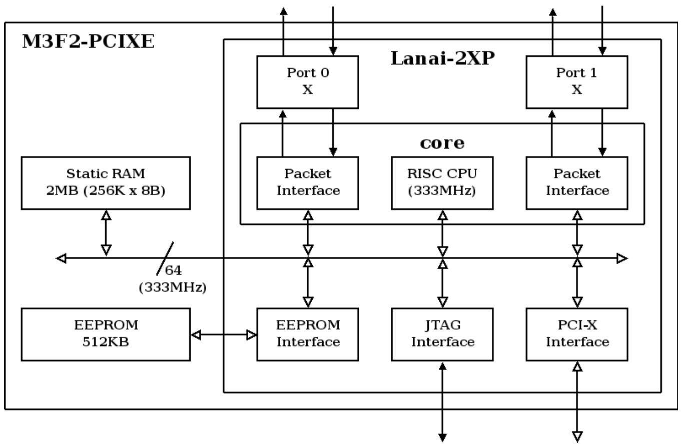

Fig. 2. Block diagram of the Myrinet network interface card.

III. Materials and Methods

Our testing environment is the *tdrdemo* development cluster of the CMS experiment at CERN. This cluster comprises 15 computers, each one running *Scientific Linux CERN Release 3.0.6 (SL),* using a Linux 2.4.21 kernel and consisting of the components listed in table 1.

The Myrinet kernel driver has been configured to use *Programmed Input/Output* (PIO) for message sizes ranging from 0 to 127 bytes. To transfer larger packets the NIC utilizes *Direct Memory Access* (DMA).

For testing and benchmarking *Myrinet Express*, a number of different test programs have been used to determine throughput in different scenarios [21]. To highlight differences in message routing, each test program was run with both ports active (channel bonding) and with only one port active. Each test is repeated 10 times for obtaining representative mean values and standard deviations. For each test run 100.000 messages (about 1.6 GBytes of event data) are sent from each source to each destination node.

A ***standalone program*** measures the raw performance of MX without additional application layer overhead. The test measures unidirectional message transmission. Buffers are allocated during initialization and reused afterwards.

A ***roundtrip*** application is implemented with XDAQ the CMS experiment's on-line software framework [22]-[24]. It induces additional application layer processing overheads in the order of 4 microseconds per incoming message. XDAQ decouples distributed applications from the underlying network technologies. *Peer transports* provide this abstraction and present a standard communication interface to application developers. Thus, the transport layer can be switched from

TABLE I
Computer Components of the Test Environment

CPU	2× Intel® Xeon™, 2.40GHz, 512kB cache, 533MHz FSB
Motherboard	Supermicro, Inc. X5DL8-GG, 533MHz FSB
Memory	2× 256MB, DDR-SDRAM, PC2700, CL2.5, ECC Registered
Myrinet NIC	1× Myrinet M3F2-PCIXE in 64bit 100MHz PCI-X slot

Ethernet to Myrinet without modifying the application code concerned.

By design, the roundtrip test mimics the UNIX ping command. The test program can pipeline multiple messages as shown in Fig. 3. For measuring two-way bandwidth, 2000 messages are allowed to be in the network at the same time.

Transmission bottlenecks may not only stem from limits of the network link but may also be dominated by the memory bandwidth of the interface between the computer's memory and the Myrinet card [25]. Therefore we also measured this vital parameter using a tool (*mx_dmabench)* provided by Myricom.

MStreamIO is a test application implemented with XDAQ to test full $N \times N$ communication. Fig. 4 lays out the protocol between server and client using only one client and one server. For measuring unidirectional throughput of N sources and N sinks, additional control messages are needed. Once a client has received "start" message from every server application it begins sending data messages to each server.

The ***event builder*** is a production XDAQ application for the CMS data acquisition system. In test mode the *readout unit* (RU) generates test data and the *builder unit* (BU) discards the data as soon as all event fragments have been assembled [26].

The first ports of the *event manager* (EVM) and seven *readout units* (RUs) are connected to one line card as shown in Fig. 5. Their second ports are linked to a second line card. Another seven computers, called *builder units* (BUs), are connected to a third and fourth line card. A Myrinet chassis implementing a 3-layered Clos switching fabric (Clos-128) hosts these four line cards.

The event building protocol [27] is shown in Fig. 6. The BUs sends event requests to the *event manager*. It assigns a BU destination to each event and replies with an event identifier to the requesting BU. The BU initiates the data transfer by requesting event fragments from each RU. After all fragments making up a full event have been received by the BU, it sends a message to the EVM to clear the event identifier. The communication pattern between RUs and BUs is $N \times N$, whereas the traffic between EVM and BUs it is $N \times 1$.

All communication, data and control, are carried out using Myrinet. This affects the results of pure N×N data message exchange only slightly and is negligible since multiple control commands are gathered in single messages.

IV. Results

Using the applications described in the previous section, unidirectional and bidirectional throughputs have been measured. Additional tests quantify throughput and scalability parameters using N senders and N receivers concurrently.

One-way throughput data obtained with a standalone program and *MStreamIO* are compared in table 2. The framework based application reaches 95 percent of the standalone performance corresponding to an application induced overhead of 11 MB in single port mode and 24 MB in channel bonding mode. Fig. 7 visualizes throughput over message size.

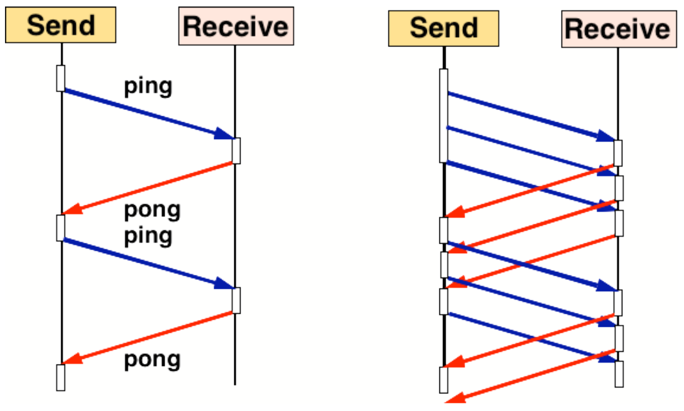

Fig. 3. The r*oundtrip* protocol. One message in the network (left). Maximum of three messages in the network (right).

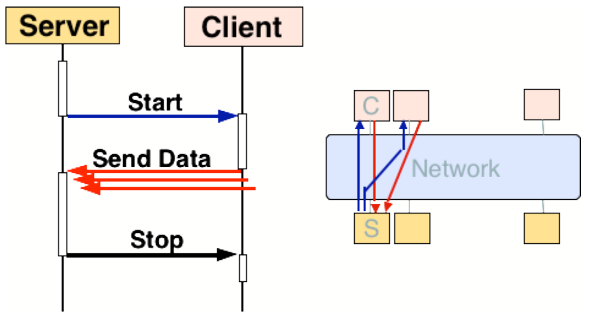

Fig. 4. The *MStreamIO* protocol for measuring unidirectional throughput. 1×1 communication (left), N×N communication (right).

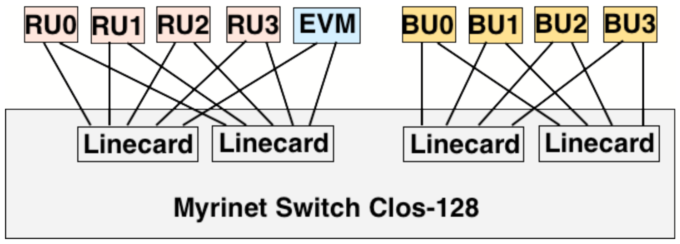

Fig. 5. 4x4 Event Builder setup

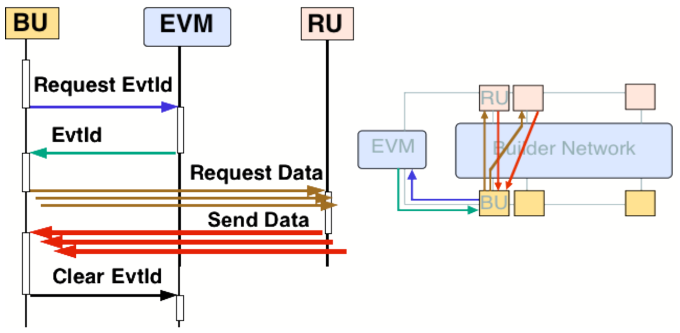

Fig. 6. The *event building* protocol.

Two-way throughput was determined using the *roundtrip* framework-based application. In single port mode the card was able to saturate the output link at 93 percent of its theoretical limit. Using both ports, however, a maximum throughput of only 65 percent of the sum of the theoretical limit per link was achieved.

Table 4 lists the throughput limits between host memory and network interface card. It should be noted that the bus capacity is 800MB/s (100MHz and 64 bit). Measuring an upper bound of combined read and write was not possible due to vendor software limitations, but has been calculated assuming equal size for each read and write request, resulting in 684.85 MB/s. Hence, the achieved two-way throughput is 94.9 percent of the raw host memory to NIC bandwidth.

The throughput behavior for unidirectional throughput from multiple source nodes to multiple destination nodes has been determined using *MStreamIO*. Table 5 shows the average throughput per node for 16KB packets. With growing system size the throughput drops to 40 percent of the wire-throughput at 7 senders and 7 receivers (see fig. 10).

Finally the *event builder* results are compiled in table 6. The application has been run in test mode with auto-generated data and 16kB fixed sized fragments. The aggregate throughput of the *builder units* is shown in fig. 11 for a system scaling from 1RU/1BU to 7RUs/BUs. Adding additional senders and receivers did not further increase the aggregated throughput of the *event builder*.

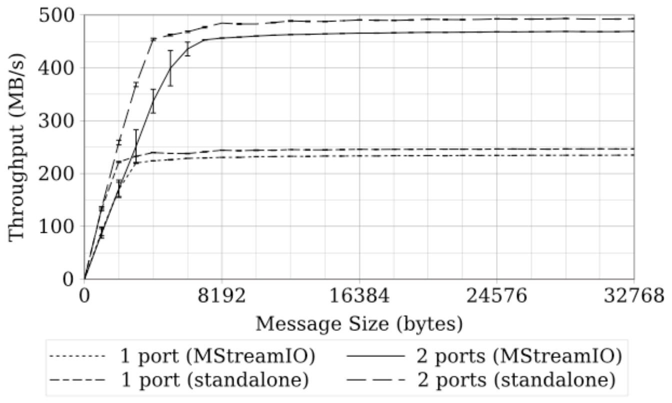

Fig. 7. One Way throughput plotted over packet size.

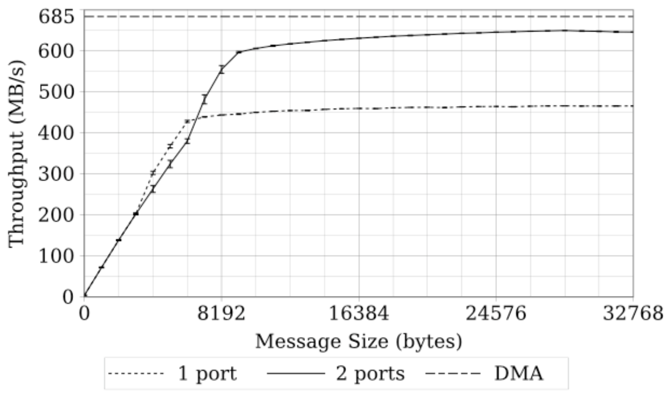

Fig. 8. *Two Way Throughput.*

TABLE II
MAXIMUM ONE WAY THROUGHPUT

Program	Number of used ports	Maximum throughput (MB/s)	Saturation (%)
Standalone	1	246.71	98.68
	2	493.41	98.68
MStreamIO	1	235.01	94.00
	2	469.14	93.83

TABLE III
MAXIMUM TWO WAY THROUGHPUT

	Maximum throughput (MB/s)	Saturation (%)
1 port	465.98	93.20
2 ports	649.90	64.99

TABLE IV
MEMORY THROUGHPUT WITH DMA

Mode	Maximum throughput (MB/s)
Read	630.41±2.36
Write	749.59±0.31
Combined (calculated)	684.85

TABLE V
MSTREAMIO N×N THROUGHPUT PER NODE

	1 port		2 ports	
N	Throughput (MB/s)	Saturation (%)	Throughput (MB/s)	Saturation (%)
1	233.46±0.02	93.38	465.66±0.65	93.13
2	192.16±0.36	76.87	346.06±0.14	69.21
3	157.05±4.41	62.82	289.47±0.39	57.89
4	137.23±0.30	54.89	263.11±0.33	52.62
5	123.33±0.95	49.33	224.63±3.49	44.93
6	110.33±1.45	44.13	209.64±2.80	41.93
7	102.00±1.28	40.80	199.18±0.79	39.83

TABLE VI
EVENT BUILDER N×N THROUGHPUT PER NODE

	1 port		2 ports	
N	Throughput (MB/s)	Saturation (%)	Throughput (MB/s)	Saturation (%)
1	242.16±0.09	96.87	484.63±1.03	96.93
2	191.96±0.86	76.78	338.03±0.49	67.61
3	154.08±0.88	61.63	277.19±0.63	55.44
4	129.28±0.42	51.71	231.83±1.08	46.37
5	109.74±1.94	43.90	191.36±3.52	38.27
6	93.71±1.89	37.48	164.29±3.52	32.86
7	79.04±2.23	31.62	141.60±4.34	28.32

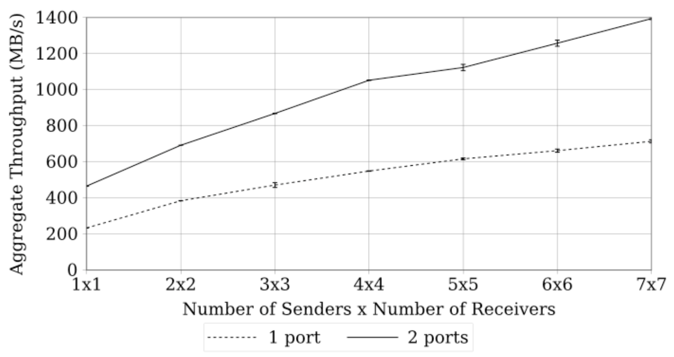

Fig. 9. Aggregate throughput for *MStreamIO* in N×N configuration.

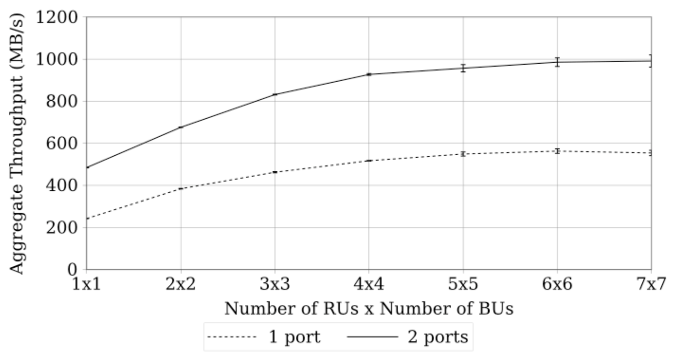

Fig. 11. Aggregate throughput for event building in N×N configuration.

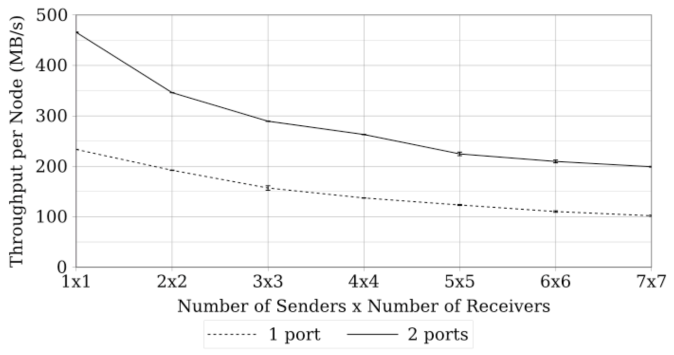

Fig. 10. Throughput per node for *MStreamIO* in N×N configuration.

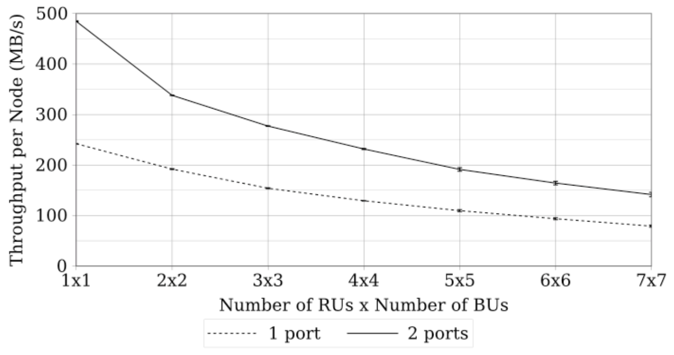

Fig. 12. Throughput per node for event building in N×N configuration.

V. Discussion

The effective performance of high-speed wormhole routing networks depends on a number of factors, among them the application level communication patterns [21], [28], [29]. While these effects have been studied extensively for supercomputing computation-bound applications [30]-[32], experience for communication-bound applications like high-energy physics data acquisition is still scarce. For small packets as found in scientific parallel computing, throughput is affected by the application-processing overhead. This overhead becomes largely negligible for messages larger than 8KB with today's network latencies. This is the case for our event-building application that operates with 16KB messages. For fully bidirectional traffic an additional bottleneck between the host's memory and the network interface card was revealed. Such effect is technology independent and applies to all high-speed networks. One aspect inherent to adaptive wormhole routing becomes observable in the presence of unidirectional N×N traffic, as it is the case for event building. Due to *Head of Line* and switching fabric internal blocking, systems become meta-stable and show evidence of sub-linear scaling. Interrupts cause no problems as the *Myrinet Express* API is working in polling mode.

In our tests, unidirectional traffic (*MStreamIO)* reaches 95 percent throughput of a standalone program. Using the asynchronous API we enqueue multiple send and receive requests, which are handled concurrently by the MCP on the network card. Handling requests on this layer is more efficient due to better scheduling granularity compared to application level and due to offloading processing power from the host CPU. Hence, the additional processing of the framework is mostly shadowed for messages larger than 8KB.

In addition, throughput in the presence of small packets is independent on the number of physical links. The gap between two consecutive messages is constant and largely application dependent. Thus channel bonding has no impact and the throughput remains the same for single-port and dual-port mode. Full throughput can be reached when the transmission time is larger or equal to the gap. In this case the application overhead is mostly shadowed by the transmission time.

When sending in both directions simultaneously over both ports the theoretic upper throughput limit is 1 GB/s. This is at the same time the limit of the PCI-X bus. Accordingly the highest achievable throughput may be lower depending on the components used for bridges and DMA controllers. For dual-port mode the highest achieved throughput is about 650 MB/s. We have seen that the throughput is limited by the interface between the host's memory and NIC to about 685 MB/s (neglecting read/write overhead). Using better hardware components throughput can be increased to over 900 MB/s. In one-port operation a single link can be almost saturated, because its wire-speed is well below the DMA engines limits.

In contrast, *Head of Line* and internal blocking in the switch fabric limit event building and unidirectional N×N traffic. The measured results obtained with *Myrinet Express* drop to 40 percent in case of unidirectional traffic and to about 30 percent for event building. Randomization at application level has been exercised, but has shown to be ineffective.

TABLE VII
THEORETICAL THROUGHPUT LIMIT (QUEUEING THEORY)

N	Throughput (%)
1	100.0
2	75.00
3	68.25
4	65.53
5	63.99
6	63.02
7	62.34
8	61.84
∞	58.58

Assuming an internally non-blocking switching fabric an upper throughput limit can be analytically obtained by applying *Queueing Theory* [33]. Table 7 presents these numbers per single switch input and output node for random traffic. For an infinite number of inputs and outputs the throughput levels-off at 58.58 percent of the wire bandwidth per link. Contrary to our initial assumptions, blocking may well occur in multi-stage Clos switching fabrics if used with source routing. Internal blocking is suggested to be a cause for the difference between *Queueing Theory* predicted and the experimentally obtained results. Two or more data sources may independently choose the same intermediate path for a route resulting in degraded throughput due to the waiting condition.

The high degree of complexity resulting from the interactions among the switching stages does not yet permit taking these effects into consideration when performing analytic performance predictions. However, simulations can provide better predictions as in [3]. In addition efforts on improving simulation environments that may eventually include such effects are under way [34], [35].

Adaptive routing aims at getting around this situation by changing routes in case of blocking. To find a non-blocking routing configuration the routes must remain stable for O(1000) consecutive messages. For event building traffic, destinations of consecutive messages are always diverse. Hence, the algorithm cannot determine a stable non-blocking routing configuration. This leads to significant lower throughput compared to the theoretical limits based on Poisson distributed random traffic shown in table 7.

Generally spoken, traffic shaping is shown to be effective if performed close to the wire, operating at time scales similar to the one of the networking technology (microseconds in case of Myrinet) [5], [6]. One example is a *peer transport* implementation for the CMS on-line software infrastructure based on a *barrel shifter* algorithm implemented as MCP [5]. It provides higher throughput than the theoretical limit for random traffic for a specific traffic pattern. An *event builder* deploying this communication subsystem benefits from linear scalability.

The *barrel shifter* implementation builds upon the idea to have one queue for each destination at the source. It takes advantage of the backpressure for synchronizing the sending operations among all data sources. After initialization each source node cycles through all destination queues in the same order scatter/gathers pending messages into fixed size buckets and sends them. To keep synchronization every source has to send a packet in each timeslot and thus must send empty packets if not data are pending for output [6].

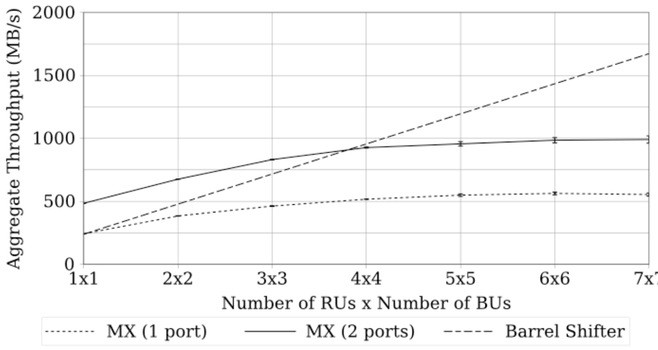

Fig. 13. Aggregate throughput for event building in N×N configuration.

The *barrel shifter* based implementation used the previous LANai 9 based NIC with a single link and was shown to provide over 95 percent of the wire-throughput for event-building traffic for up to 32 *readout* and *builder units* [36]. Fig. 13 presents the aggregated throughput results for event building for the 2 modes of *Myrinet Express* and an extrapolation of the *barrel shifter* based MCP. For the CMS experiment 200 MB/s throughput per node are required for event building [6]. This cannot be achieved by the *MX* based *peer transport* for more than 3 RUs and 3 BUs.

It remains to be elucidated, however, that a barrel-shifter system is stable under continuous, long-lasting operation as barrel-shifter synchronization is entirely based on the backpressure signals from the Clos fabric's input-layer switch ports. A disadvantage of the *barrel shifter* is also the constraint of the quadratic system configuration (N×N), thus avoiding mixed configurations with a variable number of readout and builder units. A trapezoid setup in which readout units send event data fragments directly to the high-level trigger processing nodes (*filter units*) is therefore not achievable with the Myrinet based *barrel shifter* implementation.

VI. SUMMARY

We have implemented a *Myrinet Express* based *peer transport* for the CMS on-line software framework (XDAQ) and we have determined the values for various event building traffic related parameters. We have identified and discussed several throughput and scalability limiting factors associated with adaptive wormhole routing networks.

Due to the high transfer rates, network throughput in channel bonding mode is at first order limited by the bandwidth between host memory and network interface card. In dual-port operation with full bi-directional communication, 65 percent of the wire throughput was reached in our test environment.

Using *N*×*N* random traffic the throughput drops to 40 percent of the theoretical maximum. In case of event building traffic, the throughput decreases further to 30 percent of the wire-throughput with seven sources and seven destinations. Primary causes for throughput degradation are *Head of Line* and multistage switching fabric internal blocking. Sources choose their routing paths independent from the others and may select the same intermediate path. This results in (internal) blocking of all packets except one at a time for this sub path. Adaptive routing alternates blocking routes to find a non-blocking configuration. For proper functioning of the algorithm, routes must remain stable for a large number of consecutive messages. This is not the case for event-building traffic, where consecutive messages have different destinations. Hence, the algorithm fails to adapt to this situation.

From our quantitative evaluation we conclude that adaptive wormhole routing is not well suited for event building traffic. *Myrinet Express* is targeted at scenarios where multiple consecutive messages are sent to the same destination, allowing adapting routes to find a non-blocking configuration. This applies to calculation bound parallel computing, clearly demonstrating the difference of data bound cluster computing such as high-energy physics applications.

Acknowledgments

We acknowledge support from the DOE and NSF (USA), KRF (Korea), Marie Curie Program and Eva Kühn (Vienna University of Technology, Austria).

References

[1] The CMS Collaboration, *The Compact Muon Solenoid*, CERN, Technical proposal, No. 7, LHCC 94-38, 1995.
[2] J. Gutleber, "High Performance Distributed Objects in Large Hadron Collider Experiments," 1999.
[3] P. Sphicas et al. Cms, *The TriDAS Project - Technical Design Report, Volume 2: Data Acquisition and High-Level Trigger*, 2002.
[4] E. Barsotti, A. Booth, M. Bowden, "Effects of Various Event Building Techniques on Data Acquisition System Architectures," *Computing in high energy physics*, 1990.
[5] G. Antchev et al., "Evaluation of Myrinet for the Event Builder of the CMS Experiment," *First Myrinet User Group Conference*, Lyon, France, 2000.
[6] V. Brigljevic et al., "The CMS Event Builder," *Proceedings of 2003 Conference for Computing in High Energy and Nuclear Physics (CHEP03)*, La Jolla, California, pp. WEPT003, 2003.
[7] M. Bellato et al., "The CMS Event Builder Demonstrator based on GigaEthernet Switched Network," *Computing in High Energy and Nuclear Physics (CHEP2000)*, Padova, Italy, 1999.
[8] G. Antchev et al., "The CMS Event Builder Demonstrator based on Myrinet," *Proceedings of the 14th IEEE NPSS Real Time Conference*, Santa Fe, NM, 1999.
[9] Antchev, G., et al. "The CMS Event Builder Demonstrator and Results with Myrinet," *Computing Physics Communications*, vol. 140, pp. 130–138, 2001.
[10] P. Geoffray, "Myrinet eXpress (MX): Is your Interconnect Smart?" *Seventh International Conference on High Performance Computing and Grid in Asia Pacific Region*, 2004.
[11] Myrinet.com Inc., "Myrinet Express (mx): A High-Performance, Low-Level, Message-Passing Interface for Myrinet," 2005.
[12] M. Gerla, P. Palnati, S. Walton, "Multicasting Protocols for High-speed, Wormhole-Routing Local Area Networks," *Conference proceedings on Applications, technologies, architectures, andprotocols for computer communications*, pp. 184-193, 1996.
[13] N.J. Boden, D. Cohen, R.E. Felderman, A.E. Kulawik, C.L. Seitz, J.N. Seizovic, Wen-King Su, "Myrinet – A Gigabit-per-Second Local Area Network", *IEEE Micro*, vol. 15(11), pp. 29-36, 1995.
[14] A.X. Widmer and P.A. Franaszek, "A DC-Balanced, Partitioned-Block, 8B/10B Transmission Code," *IBM J. Res. Develop*, vol. 27(5), pp. 440-451, 1983.
[15] T. Nachiondo, J. Flich and J. Duato, "Destination-Based HoL Blocking Elimination," *12th International Conference on Parallel and Distributed Systems*, ICPADS 2006, pp. 10, 2006.
[16] Myricom.com Inc., "Guide to Myrinet-2000 Switches and Switch Networks," 2001.
[17] R. Kettimuthu and S. Muthukrishnan, "A Performance Study of parallel FFT in Clos and Mesh Networks," *Proceedings of the 2005 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2005)*, 2005
[18] Myrinet.com Inc., "The GM Message Passing System," 2000.
[19] M.A. Anwar, T. Yoshida, "OORF: an object-oriented route finder," *Proceedings of the 2000 ACM symposium on Applied computing*, pp. 301-306, vol. 1, 2000.
[20] Private communication with Myrinet.com, Inc.
[21] S.C. Woo, M. Ohara, E. Torrie, J.P. Singh, A. Gupta, "The SPLASH-2 Programs: Characterization and Methodological Considerations," *Proceedings of the 22nd annual international symposium on Computer architecture*, pp. 24 - 36, 1995.
[22] J. Gutleber, S. Murray, and L. Orsini, "Towards a homogeneous architecture for high-energy physics data acquisition systems," *Computing Physics Communications*, vol. 153, pp. 155–163, 2003.
[23] J. Gutleber and L. Orsini, "Software Architecture for Processing Clusters Based on I2O," *Cluster Computing*, vol. 5(1), pp. 55-64, 2002.
[24] G. Lo Presti. *Peer-to-peer Architectures in Distributed Data Management Systems for Large Hadron Collider Experiments*, 2004. Available: http://lopresti.home.cern.ch/lopresti/phd.html
[25] C.A. Thekkath, H.M. Levy, "Limits to Low-Latency Communication on High-Speed Networks," *ACM Transactions on Computer Systems (TOCS)*, vol. 11(2), pp. 179-203, 1993.
[26] S. Murray, "RU Builder User Manual," Version 3.9.
[27] G. Antchev, L. Berti, E. Cano, et al, "The CMS Event Builder Demonstrator and Results with Ethernet and Myrinet Switch Technologies," *Computing in High Energy and Nuclear Physics (CHEP01)*, 2001.
[28] J.P. Singh, J.L. Hennessy, A.Gupta, "Implications of Hierarchical N-body Methods for Multiprocessor Architectures," ACM Transactions on Computer Systems (TOCS), vol. 13(2), pp. 141-202, 1995.
[29] R. Fatoohi, K. Kardys, S. Koshy, S. Sivaramakrishnan, J.S. Vetter, "Performance Evaluation of High-Speed Interconnects using Dense Communication Patterns," *International Conference Workshops on Parallel Processing (ICPP 2005 Workshops)*, pp. 554-561, 2005.
[30] J. J. Dongarra , "The LINPACK Benchmark: An Explanation," *Proceedings of the 1st International Conference on Supercomputing*, Springer-Verlag New York , pp. 456-474, 1998.
[31] J.P. Singh, W. Weber, A. Gupta, "SPLASH: Stanford Parallel Applications for Shared-Memory," *Technical Report: CSL-TR-91-469*, Stanford University, 1991.
[32] J.P. Singh, W. Weber, A. Gupta, "SPLASH: Stanford Parallel Applications for Shared-Memory," *Technical Report: CSL-TR-92-526*, Stanford University, 1992.
[33] M.G. Hluchyj and M.J. Karol, "Queueing in High-Performance Packet Switching," *IEEE Journal on Selected Areas in Communications*, vol. 6(9), pp. 1587-1597, 1998.
[34] D. Tutsch, M. Brenner, "MINSimulate - A Multistage Interconnection Network Simulator," *17th European Simulation Multiconference: Foundations for successful Modelling Simulation (ESM'03)*, Nottingham, UK., pp. 211–216, 2003.
[35] T.L. Wilmarth, G. Zheng, E.J. Bohm, Y. Mahta, N. Choudhury, P. Jagadishprasad, L.V. Kale, "Performance Prediction using Simulation of Large-Scale Interconnection Networks in POSE," *Proceedings of the Workshop on Principles of Advanced and Distributed Simulation (PADS'05)*, 2005.
[36] F. Meijers, D. Samyn, "EVB Demonstrators. Status and Plans – New Results on Myrinet RU-Builder Plans," 2002.