Testing and Modeling Ethernet Switches and Networks for Use in ATLAS High-level Triggers

R. W. Dobinson[1], S. Haas[1], K. Korcyl[1], M. J. LeVine[2,1],
J. Lokier[1,3], B. Martin[1], C. Meirosu[1,4], F. Saka[1,5], K. Vella[6]

[1]CERN, 1211 Geneva 23, Switzerland
[2]Brookhaven National Laboratory, Upton, NY 11973-5000, USA
[3]University of Liverpool, Liverpool, UK
[4]"Politehnica" University of Bucharest, Bucharest, Rumania
[5]Royal Holloway College, University of London, Egham, UK
[6]University of Malta, Msida MSD06, Malta

***Abstract*-- The ATLAS second level trigger will use a multi-layered LAN network to transfer 5 Gbyte/s detector data from ~1500 buffers to a few hundred processors. A model of the network has been constructed to evaluate its performance. A key component of the network model is a model of an individual switch, reproducing the behavior measured in real devices. A small number of measurable parameters are used to model a variety of commercial Ethernet switches. Using parameters measured on real devices, the impact on the overall network performance is modeled.**

In the Atlas context, both 100 Mbit and Gigabit Ethernet links are required. A system is described which is capable of characterizing the behavior of commercial switches with the required number of nodes under traffic conditions resembling those to be encountered in the Atlas experiment. Fast Ethernet traffic is provided by a high density, custom built tester based on FPGAs, programmed in Handel-C and VHDL, while the Gigabit Ethernet traffic is generated using Alteon NICs with custom firmware. The system is currently being deployed with 32 100Mbit ports and 16 Gigabit ports, and will be expanded to ~256 nodes of 100 Mbit and ~50 GBE nodes.

I. INTRODUCTION

ATLAS [1], [2] is one of four experiments which will be located at the Large Hadron Collider now being constructed at CERN. Proton-proton bunch crossings occur in ATLAS at 40 MHz. Crossings which contain events of interest are selected in a series of trigger decisions: level 1 (LVL1) accepts events at rates up to 100 kHz. ATLAS is composed of several detectors, which deliver data to a set of approximately 1560 readout buffers (ROB) over optical links in response to a LVL1 accept. Based on the type of LVL1 trigger, a subset of data is selected, corresponding to regions of interest (ROI) in each detector, to be passed to the next trigger level. The readout and subsequent analysis of these ROI data is assigned to a member of the level 2 (LVL2) processor farm, a set of 500-600 commodity CPUs. The designated processor sends readout requests to the appropriate ROBs (about 5 per cent are selected for each LVL1 trigger) and receives the data sent in response to these requests.

The requirements for the network interconnecting the ROBs and the LVL2 system are set by the above parameters. Each ROB sees approximately 11K requests per second, corresponding to a worst-case data rate of ~4MB/s; each LVL2 CPU generates requests and receives responses at the rate of 14 kHz, with a corresponding average data rate of 6.6 MB/s. The aggregate data rate flowing between ROBs and processor nodes is about 2.3 GB/s.

LVL2, which must reach a decision in a few msec, accepts events at rates on the order of 1-5 kHz. Accepted events are assembled by the Event Filter, where another level of event selection is carried out on the complete events.

The Ethernet network described here as a candidate for the ATLAS LVL2 application is shown in Fig. 1. The configuration assumed for this network is a two-level system where the nodes (ROBs and CPUs) are connected to 100 Mbit/s ports in concentrating switches which in turn are connected to a central switch via 1 Gigabit/s ports.

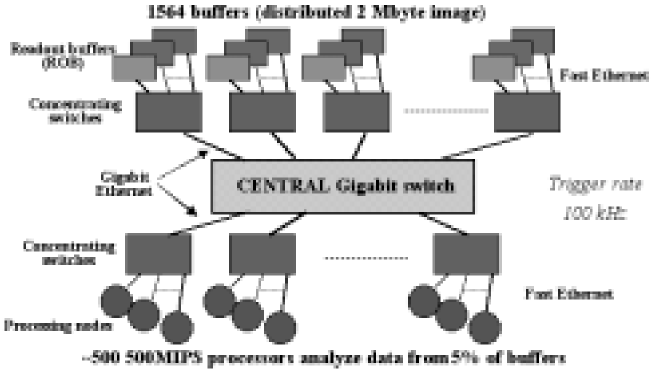


Figure 1. The network interconnecting ATLAS ROBs and LVL2 trigger system.

The required transaction rate of 14 kHz at the CPUs is well in excess of that which can be supported by present CPUs running TCP/IP [see section III]. For that reason MESH [3], a scheduler which makes use of a lightweight protocol running on raw Ethernet packets is used.

II. MODELING

The paradigm followed here was to use an object-oriented model of the switches that constitute the network. The

Manuscript received Nov 1, 2000 This research was supported in part by the U.S. Department of Energy under Contract No. DE-AC02-98CH10886.

model, written in the C++ language, is interfaced to an event-driven simulation framework. Both OPNET [4], a commercial product which is designed specifically to model computing networks, and Ptolemy [5], a general modelling framework developed at the University of California, Berkeley, have been used for the framework in the studies to be described.

A. The switch model

At the heart of the network model is the parameterized model [6] of the Ethernet switches. Such a parameterized model was developed for the concentrating switches (Fig. 1), assuming a hierarchical architecture as shown in Fig. 2. The switches are also assumed to utilize a store-and-forward mechanism, where frames are completely buffered at the input port before being sent to the output port. The store-and-forward model was chosen because it is the dominant mechanism used in commercially available switches.

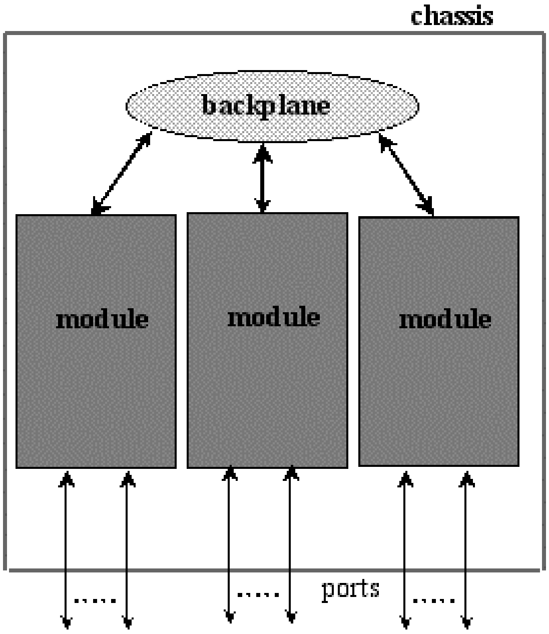
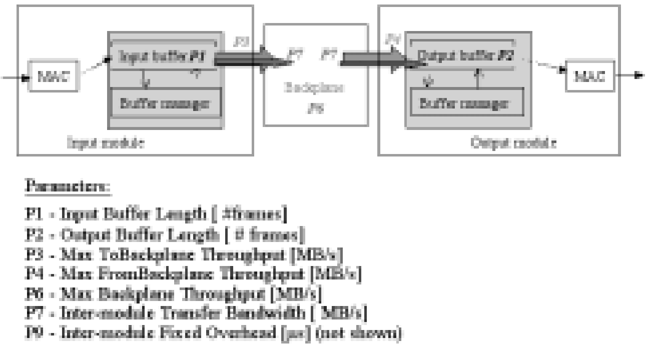


Figure 2. The hierarchical switch model

Parameters characterizing the switch behavior are chosen for both intra- and inter-module traffic. The parameters were chosen to represent a minimal set which describes the behavior of various switches, and which is amenable to direct measurement to the largest extent possible. The set of parameters used for the inter-module traffic is shown in Fig. 3.



Parameters:

P1 - Input Buffer Length [#frames]
P2 - Output Buffer Length [# frames]
P3 - Max ToBackplane Throughput [MB/s]
P4 - Max FromBackplane Throughput [MB/s]
P6 - Max Backplane Throughput [MB/s]
P7 - Inter-module Transfer Bandwidth [MB/s]
P9 - Inter-module Fixed Overhead [μs] (not shown)

Figure 3. The parameter set used to characterize the inter-module switch behavior.

B. Switch measurements

While a few of the parameters (P1, P2 in Fig. 3) are determined from the manufacturer's specifications, most of them are determined by a set of measurements. These consist of measurements of latency, and of streaming throughput, both for packets of different sizes, and with the nodes connected either directly, through the same switch module, or through different switch modules. All measurements are carried out using PCs running MESH (raw Ethernet). Latency was determined as one-half the round-trip time in the ping pong mode, where messages at the receiving end are returned to the sender. The switch parameters were determined for a variety of commercially available switches, with consistent results for the switches tested. The switches tested included two different fast Ethernet (FE) switches and a switch with up to 32 FE ports and up to 4 Gigabit Ethernet (GBE) ports.

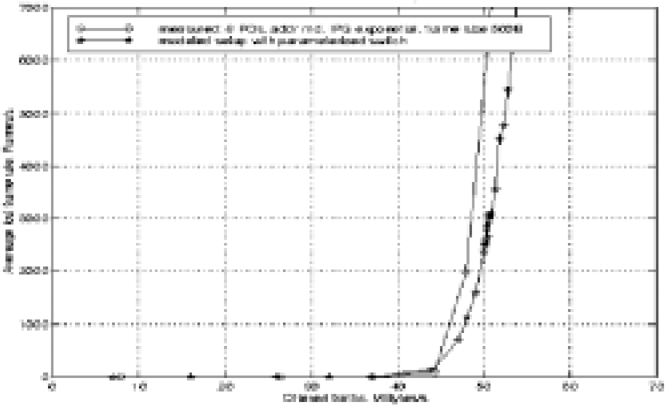


Figure 4. Comparison of the switch model with measurement. Packet loss is plotted as a function of traffic offered.

Fig. 4 shows packet loss, measured and predicted by the parameterized switch model, as a function of offered traffic. The model correctly predicts the onset of packet loss.

C. The network model calculations

The parameterized switch model was used to model the ATLAS LVL2 system shown in Fig. 1. The models of the ROBs and the processor nodes were very simple. The processors generated requests of 100 bytes to randomly chosen ROBs; the ROBs generated responses whose size corresponded to the part of the ATLAS detector it was serving. There were 1532 ROBS and from 500 to 600 CPUs in the model calculations. To model the concentrating switches, the parameters determined for the Fast Ethernet/Gigabit Ethernet switch were used. The concentrating switches were modeled as comprising 3 modules with 8 FE ports each, and one module with a single GBE port which was the connection to the central switch The central GBE switch was modeled as fully non-blocking; this model will be replaced when real GBE switches can be characterized.

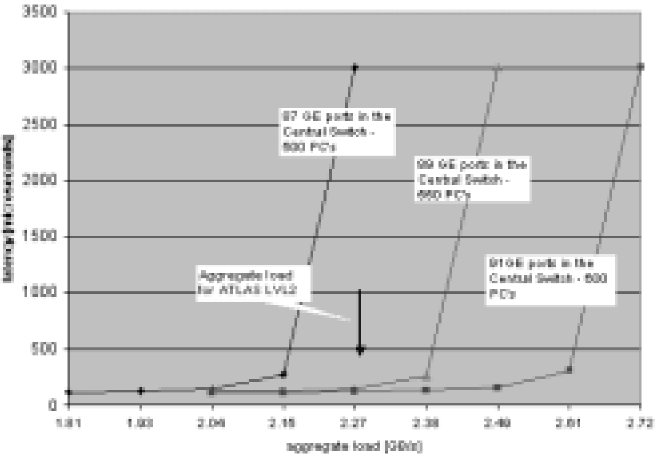


Figure 5. Latency vs aggregate load for the modeled ATLAS LVL2 network.

A result of the model calculations is shown in Fig. 5. The latency is entirely due to the switches, since there is no processing delay in the model calculations. The results show that the network becomes unstable due to congestion in the ports in the central switch when there are fewer than 89 ports in this switch (550 CPUs in the LVL2 farm), for the load expected for this system (vertical arrow in Fig. 5). The calculation simulating 2 seconds of activity for the ATLAS LVL2 system required almost 3 hours of running on a 400 MHz PC; the same calculation using a generic model of the switch required almost 6 hours.

D. Model limitations

The model calculations are based on switch parameters determined over only a limited range of the operational space of the switch. The PC-based measurements with FE were not able to generate traffic at full line speed for the smallest packet sizes, and the standard driver for the GBE adapter could only run at 35 per-cent of line speed. In addition, the assumptions in the model itself are not well tested by measuring a system composed of a single switch. In order to have greater confidence in the model predictions for a system the size of ATLAS LVL2, it would be necessary to measure the characteristics of a significantly larger system.

III. Further PC-based measurements

Measurements of CPU utilization vs. packet rate have been performed for minimum and maximum length Ethernet frames, with both MESH and TCP/IP, for FE as well as for GBE. These measurements were carried out with a 400 MHz Pentium III PC, running Linux. The results in Fig. 6 show that MESH greatly reduces the fixed per message overhead and also eliminates memory-to-memory copies. In this case the CPU load for a given frame rate is virtually independent of the frame length and therefore the lines for the short and long frames overlap.
The vertical bar shows the required frame rate for the LVL2 CPUs in the ATLAS LVL2 system. It is clear from these measurements that TCP/IP with existing PCs cannot meet this requirement.

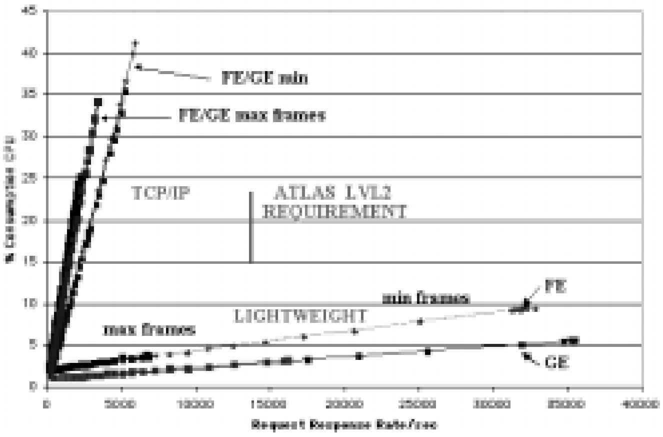


Figure 6. CPU utilization for TCP/IP and MESH for minimum and maximum length frames, measured for FE and GBE.

IV. Gigabit Ethernet switch testing

Efforts to extract maximum performance from GBE NICs using standard drivers for TCP/IP frames were disappointing, achieving 35 per-cent of line speed at best. Two bottlenecks were identified in generating Gigabit Ethernet traffic using standard PC-based systems: the protocol stack and the transfer speed of the PCI bus. In order to overcome the first of these bottlenecks raw Ethernet frames were used; the second is avoided by generating the frames directly on the network card, following a traffic pattern that is downloaded into the NIC before the start of the test.
In this study ALTEON [7] ACENIC network adapters were used. These NICs are based on the TIGON chip, which contains two MIPS R4000-style processors, each with private scratchpad memory, two DMA engines, a bridge to the PCI bus, and access to a shared memory. Since all the packet processing is done by the NIC itself, the standard network card driver is no longer needed. Instead we have developed a driver that accesses the NIC's memory as a UNIX character device and provides the means for communicating with the onboard processors via a shared memory area. This driver is also used to download the firmware into the card, starting, stopping and resetting the onboard CPU. It was developed for the Linux operating system and can be used with kernel versions starting from 2.0 up to 2.4test.
For development of code to run on the MIPS-like CPUs, the standard GNU MIPS compiler was rebuilt to run on Linux. Since these processors lack a hardware implementation for multiplication, division and floating point, a software implementation is used and linked at compile time. The linker was modified to relocate the code intended to reside in the fast scratchpad memory of the processors, which made the coding more convenient and the resulting code clearer.
The firmware code is written in the C language. We modified the original firmware, made available by the NIC's manufacturer, in order to adapt it to our specific demands and to increase performance. The performance improvement has two sources: modified send and receive procedures and reducing all other processing to the absolute minimum. In order to achieve line speed rate for almost all size of packets, the send code makes sure that the MAC always has at least one packet in the transmit queue. The packets to be sent are preloaded to the NIC in an array of packet descriptors. The source and destination addresses

are contained in the packet descriptors, so at the actual moment of sending the NIC CPU has only to compute a timestamp (if required); no further processing is done on the packet. The CRC for the packet is automatically added by the hardware. The receiver code is in fact a tight loop polling the MAC registers for the arrival of a packet. When this event occurs, a timestamp is computed (if required) and the packet is consumed without further processing. Full advantage was taken of the two processors embedded in the NIC controller. The first processor is used for receiving packets, since this is a time critical job, especially when latency is to be measured; the second processor is used for sending packets. With a send and receive code very close to that of the vendor and using only one processor for both sending and receiving packets, we were only able to achieve line rate for frames bigger than 500 bytes. By assigning the send task to one processor and the receive task to the other processor, we were able to reduce this size to about 300 bytes. With our current optimized code, line rate is achieve for frames starting with 50 bytes data payload.

The packets are time stamped with a “virtual global” clock value.   A local clock is used, whose value is corrected using a table updated periodically to compensate for drift with respect to an external [global] clock which is located on another PCI board within the same computer and provides a 66 MHz clock value. Several of the global cards can be used, in different computer chassis; the synchronization of these global clock cards is maintained by direct electrical means. The accuracy of packet time measurements obtained by this heuristic is +/- 225 ns (15 global clock ticks).

Eight of these Alteon NICs, running firmware as described above, have been installed in the backplane of an industrial PC, and is being used to test a BATM Titan 5, 8-port Gigabit Ethernet switch. Latency was measured for raw Ethernet packets traversing the switch as a function of aggregate throughput in the switch, using uniform size packets in a systematic traffic pattern (Fig. 7). This measurement shows that, for maximum size frames, the latency is constant until the full switch bandwidth is utilized.

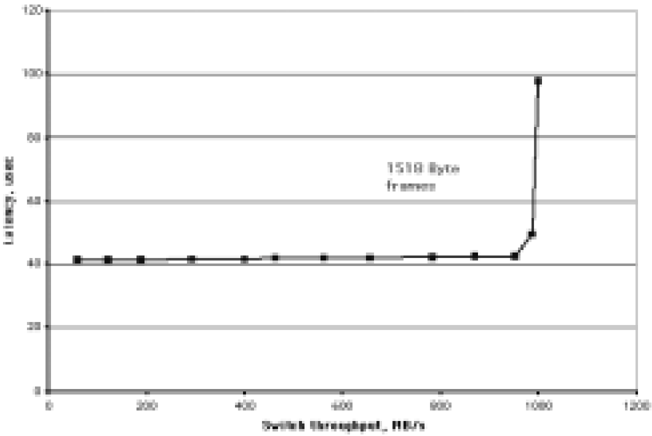


Figure 7. Latency *vs* aggregate throughput for an 8-port switch measured with the customized Alteon NICs described in the text.

V. FAST ETHERNET TESTER

In order to facilitate testing a larger number of FE ports while exploring the full operational space of FE, a custom-built FE tester board was designed and built. The board is equipped with 32 full-duplex FE ports, a high-speed parallel port connected to an intelligent host, and two FPGAs which generate packet descriptors (TxMan) according to a pre-programmed traffic pattern, as well as filling histograms (RxMan) of packet latencies and packet losses. Latencies are calculated using time stamps embedded in each frame, with the aid of an externally generated hardware clock.

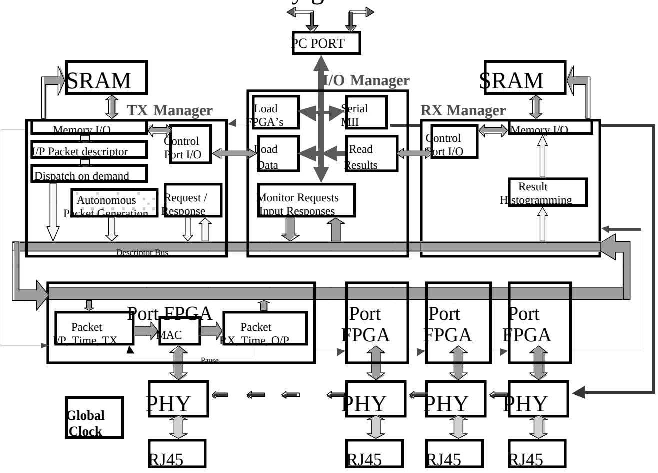


Figure 8. Block diagram of the FE tester. There are 32 FE ports, a parallel port connection to the host, a receiver manager and transmitter manager, all interconnected by a wide, synchronous bus.

The MAC functionality on the board is implemented in Altera FLEX 10K50S [8] FPGAs, programmed largely in Handel-C [9], a high-level language which implements a subset of the C language, with extensions appropriate to FPGA targets. The language also provides an underlying state machine implementation that is accessed using constructs similar to those of Occam [10]. A few time-critical tasks are programmed in VHDL. The MACs programmed in this way are able to handle full-duplex raw Ethernet traffic at FE line speed.

A wide, synchronous bus connects the 32 FE ports with a receiver manager and transmitter manager. This bus carries information to each MAC in the form of a packet descriptor. Packet descriptors which dictate packet length, destination, delay, and priority of outgoing packets are carried from TxMan to each MAC, while information about incoming packets is carried from each MAC to RxMan, which manages the histogramming.

Both TxMan and RxMan are equipped with 4 MB of fast RAM, to store the traffic patterns and histograms.

Traffic patterns are downloaded from the host via the parallel port, and histograms can be uploaded using the same mechanism.

The board has been constructed and tested. The FPGA code is largely complete.

VI. ROB EMULATOR

The FE tester was designed to generate arbitrary Ethernet traffic patterns and to record patterns of latency, throughput, and packet loss under varying conditions. It became obvious that this board could be utilized to generate traffic specific to the ATLAS LVL2 system. In other words, the ports on the tester could be programmed to emulate ROB behavior. Each port would respond to a ROB data request from a LVL2 processor by generating a response whose size and delay could be programmed in the MAC FPGA. The response would copy the time stamp of the request packet into the body of the response. The contents of the packet would not have any relation to the ROB’s data, however.

Because of the request-response nature of the LVL2 traffic, the round-trip time is calculated by the sending node (the LVL2 processor), which time stamps the outgoing packet, as well as time stamping the incoming response packet. Subtracting the two times give the round-trip time with no need for clock synchronization, since both time stamps were generated by the same CPU.

Using the tester board in this way will allow us to create a large-scale test bed for the LVL2 system which will carry traffic which is characteristic of the real ATLAS traffic. Eight boards will be constructed in order to implement 256 emulated ROB ports, and to interconnect these with 64 CPUs, using a two-level switch configuration as described in section II.

VII. Further modeling

Detailed studies of the large-scale test bed will be continued following the improvements to the parameterized models of the switches made possible by the improved testing capability described here, and the results will be compared with measurements made using the test bed. Following this, the full-scale LVL2 system will be modeled.

VIII. References

[1] Atlas High-Level Triggers, DAQ and DCS; Technical Proposal. CERN/LHCC/2000-17 March 2000.

[2] J. Bystricky and J.C. Vermeulen, “Paper Modeling of the ATLAS LVL2 Trigger System,” ATLAS Internal Note ATL-DAQ-2000-030, April 2000.

[3] M.Boosten et al, “High bandwidth concurrent processing on commodity platforms,” IEEE Real-Time 99, Santa Fe, USA, June 1999.

[4] OPNET Modeler Environment – MIL3, Inc., 3400 International Drive NW, Washington DC 20008, USA. Available: http:/www.mil3.com.

[5] P. Clarke, G. Crone, M. Dobson, R. Huges-Jones, K. Korcyl, S. Wheeler, “Ptolemy simulation of the ATLAS level-2 trigger,” ATLAS internal note, ATL-DAQ-2000-039, CERN, 25.04.2000

[6] K. Korcyl, B. Dobinson, F. Saka, “Modeling large Ethernet networks using parameterized switches,” OPNETWORK 2000 conference, Washington DC, 28 Aug - 1 Sep 2000. Available: http://nicewww.cern.ch/korcyl/opnetwork2000/op2k_paper.pdf

[7] Alteon WebSystems, 50 Great Oaks Blvd., San Jose, CA 95119, USA. Available: http://www.alteonwebsystems.com.

[8] [Altera Corporation, 101 Innovation Drive, San Jose, CA 95134, USA. Available: http:/www.altera.com

[9] [Celoxica Limited (formerly Embedded Solutions Limited), 7 - 8 Milton Park, Abingdon, Oxfordshire, OX14 4RT UK, Available: http://www.celoxica.com.

[10] [Programming in Occam 2, Geraint Jones and MichaelGoldsmith, ISBN 0-13-730334-3, 1988.