Modeling Ethernet networks for the ATLAS Level-2 trigger.

K.Korcyl[1,2], *F.Saka*[1,3], *R.W.Dobinson*[1]

[1] CERN, Geneva, Switzerland;
[2] Henryk Niewodniczanski Institute of Nuclear Physics, Cracow, Poland;
[3] Royal Holloway, University of London, U.K.

Abstract

In the ATLAS LVL2 trigger a switching network is required to transfer ~5 GByte/s detector data from ~1500 read-out buffers to ~500 processors which execute the selection algorithms. Because of the size of the network it has to be built as a multi-layered structure using many individual switches. In the first part of this note we present an approach whereby only a small number of measurable parameters are used to model a variety of commercial Ethernet switches. In the second part we present results from modeling the full ATLAS second level trigger network. Our first model of the network uses idealized switches, where only queuing is of consequence i.e. the performance of the network is limited by the Ethernet link rates. Modeling the network using parameterized models of the switches based on real measurements will follow.

1.0 Introduction

The second level trigger of LHC's ATLAS experiment has to perform real-time analysis on detector data at rate of ~5 GByte/s. A switching network is required to connect ~1500 read-out buffers to ~500 processors which execute the trigger algorithms [1].

As the technology choice for the network we have focused on Ethernet (IEEE 802.3). This offers a number of advantages:

- Ethernet is a very well established international standard for 10Mbps, 100Mbps, and 1 Gbps rates. There is a new 10 Gbps standard under rapid development;
- Ethernet has 80% of the LAN market. Competition between numerous manufacturers, its popularity and ubiquity as a COTS (Commercial Off The Shelf) product makes it the cheapest LAN technology;
- New standard modifications look very useful for our needs:
  - Trunking (use of multiple up-links between switches);
  - Priority tagging for frames;
  - VLANs.

The size of the ATLAS LVL2 trigger will not allow the construction of a full size prototype of the network required in the final system. The maximum rates the system can handle, the system's throughput, latency, scalability and other issues can only be estimated by modeling. As the market for Ethernet switches grows rapidly we are interested in a generalization and parameterization of the switch operation. Such approach would allow us to model large networks using models of the switches currently on the market as well as new developments. Simplified models of the switches execute faster which is an important issue when modeling large networks.

This note consists of two parts. In the first we present an approach which led us to produce a parameterized model of the switch. The second part is a snap shot of ongoing work aimed at using the parameterized model of the Ethernet switches to model the full LVL2 system network.

1.1 Objectives of the project

The main objective of this work was to understand behavior of the switches and to try to identify parameters relevant to their performance. The objective was not to produce parameterized models with a large set of parameters, but rather to limit the number of parameters to those required to reproduce the behavior observed in real measurements. We sought parameters that could later be used to describe the behavior of a whole class of switches (store-and-forward switches with hierarchical architecture) and not just one product from a particular vendor.

Switch performance – throughput, latency and frame loss - depends on how the frame routing policies are implemented. The high priority tagged frames will have different latencies from low priority frames. The performance of the switch also depends on routing policy implemented for different types of network transfers: unicast transfers may be treated differently from multicasts and broadcasts. Therefore the model of a switch should be flexible enough to easily implement the varying approaches adopted in different vendors' products.

Additional Ethernet features such as trunking or Quality-Of-Service may also have significant impact on performance and latencies. Thus the objective here was to produce a model flexible enough to accommodate changes and development of the IEEE 802.3 standard.

We approached the objectives in the following steps:

- Select a representative switch.
  As there are many Ethernet switches from different manufacturers we selected for this study one of the products available on the market. At the start of the project it was not clear whether a suitable switch parameterization could be achieved and how far we would have to understand switch internals. We therefore sought and achieved a close collaboration with the switch vendor.
- Make set of measurements to find out how the switch works.
- Construct a detailed model of the switch (with collaboration from the switch vendor) to better understand it's operation.
- Use the detailed model with various number of settings to isolate parameters crucial for the switch performance
- Having successfully identified a suitable set of parameters construct model of a class of Ethernet switches independent of vendor specific internal details.
- Construct a model of the multi-layered network of the ATLAS LVL2 system.
- Model the network performance using idealized models of the switches. This would allow us to predict the performance of the network in the best case scenario.
- Model the network performance using the parameterized model of the switches to see the impact introduced by today's switches (the parameterized model could be tuned to a particular switch with parameters measured on and collected from the real device).
- Try to assess parameters of the switches which would satisfy the best the ATLAS LVL2 system requirements.

In the following sections of this paper we present the steps in more detail.

2.0 The COTS Ethernet switches

Most of the Ethernet switches on the market today have a hierarchical architecture of ports, modules and backplane. A schematic of the hierarchical switch organization is shown in the Figure 1. A number of ports are housed in a module. For 10Mpbs and 100 Mbps Fast Ethernet this number is significantly greater than one. For a Gigabit Ethernet module the number of ports is typically in the single digits range (less than 10). Communication between ports on the same module is handled internally by the module. Communication between modules requires the backplane. The chassis houses the physical structure.

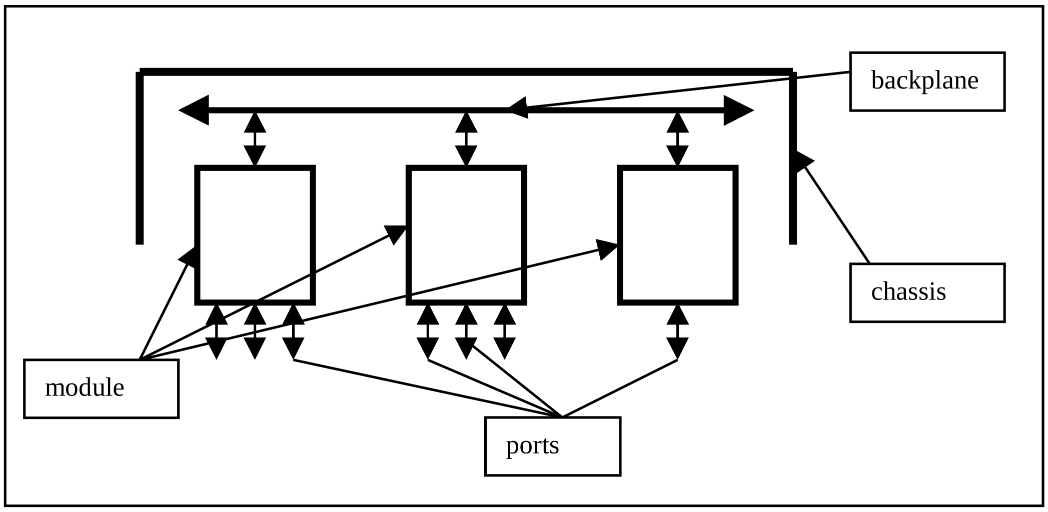


Figure 1. Hierarchical switch architecture

One of the switching options is the way incoming frames are forwarded. Switches using a store-and-forward mode of operation store a frame completely before forwarding it to the appropriate destination port. In cut-through switches, a routing decision is taken as soon as the destination address of the frame has been decoded. The rest of the frame can be received at the input port while at the same time the beginning of the frame is being transmitted at the output port. We used the store-and-forward mode in our models as it has become more and more dominant in current products. There are a number of reasons for this.

- Store-and-forward allows the transfer of frames between different speed ports (for example between Fast and Gigabit Ethernet);
- Buffering in switches improves network performance (which is important in congested networks);
- Switches can discard corrupted frames before forwarding them to the destination port, this limits unnecessary transfers of corrupted frames and associated network occupation.

2.1 Basic communication measurements suggest internal switch structure

One of the important characteristics of the store-and-forward switches is the number of stages where the frame is fully buffered before it reaches an output port. This number can be deduced from basic communications measurements. The measurements can also reveal limitations when handling heavy loads. The measurements were carried out using PCs running the Linux operating system with optimized Ethernet drivers, details of which can be found in [2].

The basic communication measurements are:

1. The Ping-pong or COMMS1 measurements: one PC sends request to another PC and awaits for response of the same size. In this measurements there is only one frame at a time in the system. The latency of the frame's round trip time is measured.
   The latency measurements can be used to calculate the transfer bandwidth. If the latency measurements can be expressed in form: $L = m * x + c$, then the inverse of m corresponds to the transfer bandwidth (speed of the transfer) [3], [4].
2. Streaming measurements: one or more PCs generate multiple frames continuously with a specified inter-frame time. The total number of generated frames is compared with the number of accepted frames to check whether there are any packets lost. Packet loss can be due to saturated switch or overflowing the receiving PC. The streaming measurements are done with disabled flow control on the switch for varying inter-frame times. The reason for disabling the flow control on the switch is to investigate the switch limitations rather than provoking reduction in the transmit PC's rate by the switch.
   To measure limitations in the switch a number of PCs generating and absorbing traffic may be required.

Interpretation of results from the streaming measurements is much simpler if the number of places where frames may queue is limited. Therefore the streaming measurements are performed with systematic traffic: i.e.

- only pairs of PCs should communicate (to avoid queuing when two PCs send frames to the same destination PC);
- inter-frame time should be constant (to avoid queuing inside the PC in case the inter-frame time might be shorter than the time to send out the previous frame).

To reveal the internal structure of modular switches the basic communication measurements are repeated in different configurations:

- PCs communicating via a common module;
- PCs communicating via backplane.

We made the following measurements:

Ping-pong measurements:

1. Direct connection between two PCs (no switch): to obtain a base for the different sorts of software overheads introduced by the PC (operating system, communication with Network Interface Card etc.)
2. Intra-module communication: to measure the frame latencies for the intra-module communications as a function of the frame size.
3. Inter-module communication: to measure the frame latencies for the inter-module transfers as function of the frame size.

Streaming measurements:

1. Direct connection: These measurements are designed to obtain the maximum rates handled by the PC-NIC combination as a function of the frame size;
2. Intra-module communications: These measurements are designed to obtain maximum rates for the intra-module communications as a function of the frame size.
3. Inter-module communications: These measurements are designed to obtain maximum rates for the inter-module communications. Limitations in the inter-module communications may arise from either the backplane bandwidth itself or from limited

access from a module to the backplane. The module's access rates to and from the backplane may be asymmetric.

Setups and aims of the measurements are summarized in the Table 1.

Setup of the measurement		What is measured	What can be learnt from measurements
Ping-pong (COMMS1)	Direct connection	Latency L	The PCs software overhead
	Intra-module	Latency L	Constant overhead c_1 and transfer bandwidth $1/m_1$ from equation $L = m_1x + c_1$ (where x – frame size)
	Inter-module	Latency L	Constant overhead c_2 and transfer bandwidth $1/m_2$ from equation $L = m_2x + c_2$ (where x – frame size)
Streaming	Direct connection	Achieved rate	Maximum throughput (rate * frame size) the PCs can generate/absorb
	Intra-module	Achieved rate	Maximum intra-module throughput (rate * frame size)
	Inter-module	Achieved rate	Maximum inter-module throughput (rate * frame size)

Table 1: Basic switch measurements.

2.2 Results from measurements help to reverse engineer internal switch structure.

Results from the ping-pong measurements and from the streaming measurements taken on the COTS Fast Ethernet switch are presented in Figure 2 and in Figure 3 respectively .

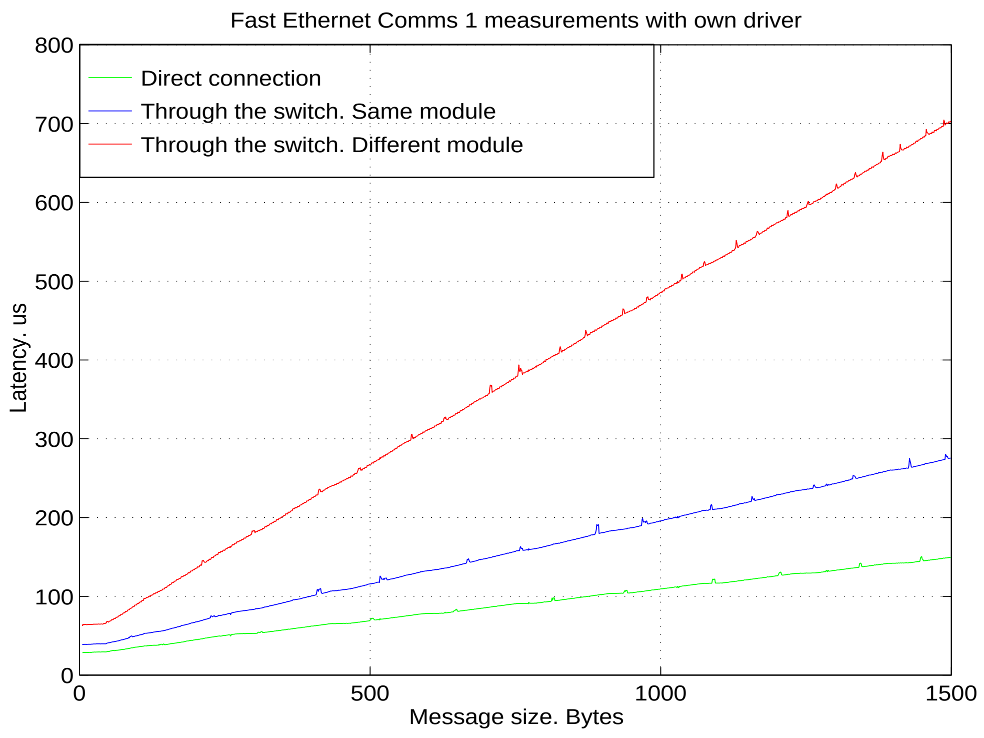


Figure 2: Ping-pong (COMMS1) measurements on a Fast Ethernet switch

In both cases we examined three different setups: the direct connection, the intra-module communication and the inter-module communication over the backplane. In these setups we took measurements for different data sizes. Results from the streaming measurements for the direct connection and for the intra-module communications are not shown in Figure 3. For 500 Bytes and 1500 Bytes frames linespeed was achieved.

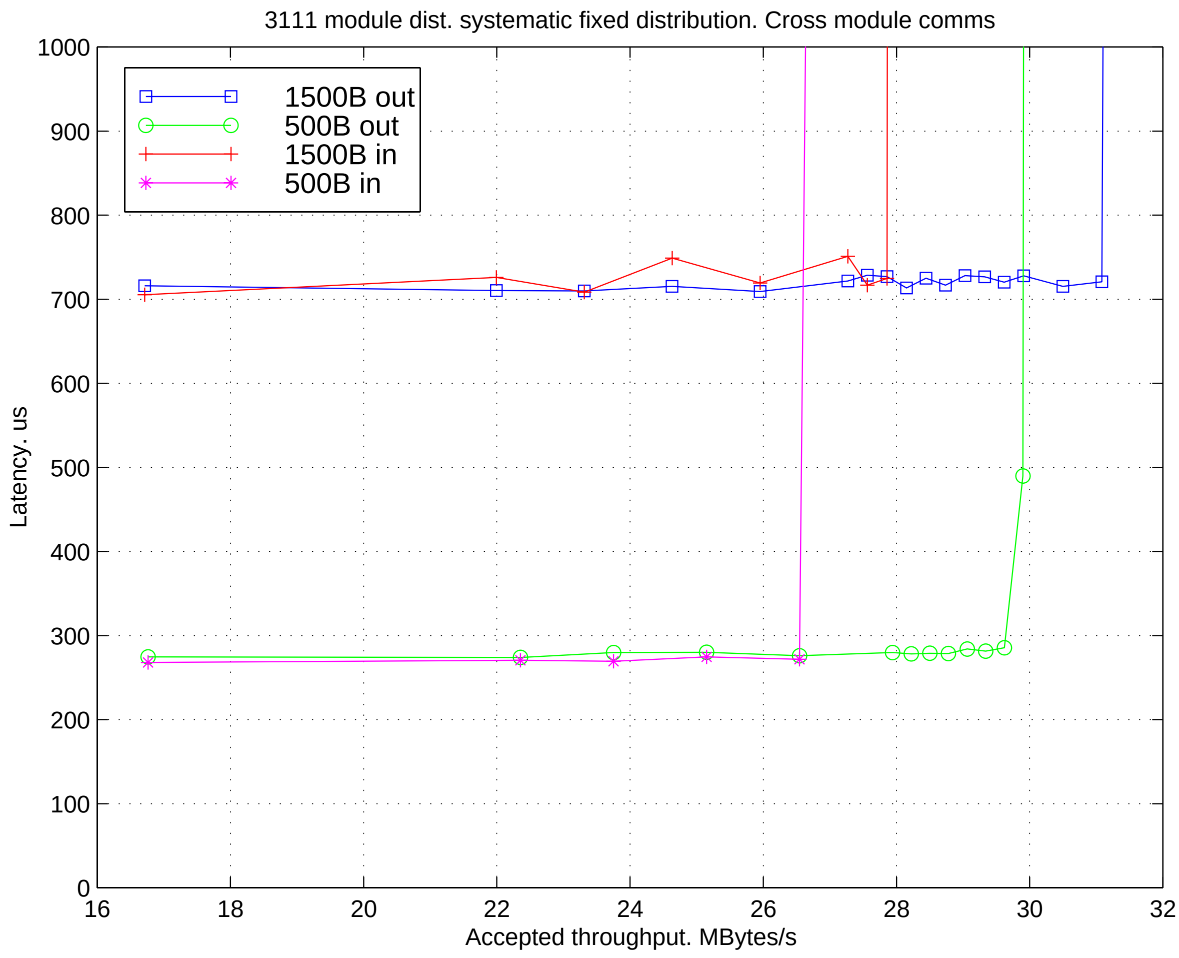


Figure 3: Streaming measurements for inter-module communications on the Fast Ethernet switch.

Results from the ping-pong and the streaming measurements are presented in the Table 2. The latencies from the ping-pong measurements represent half of the round trip time. From the latencies we subtracted the time the frame is transferred on the Ethernet links. In the intra-module and inter-module setups we subtracted also the software overhead measured in the direct connection setup. The constant part of latency is presented in time units. The data size dependent part (in the inter-module transfers) is presented as a transfer bandwidth.

In the streaming measurements we multiplied recorded rates by the frames sizes to get throughput.

Setup of the measurement		What we learnt
Ping-pong (COMMS1)	Direct connection [1]	Fixed overhead: 28 µs
	Intra-module[2]	Fixed overhead: 4 µs
	Inter-module[3]	Fixed overhead: 12 µs; Bandwidth <4 MB/s
Streaming	Direct connection	Fast Ethernet linespeed achieved
	Intra-module	Fast Ethernet linespeed achieved
	Inter-module[4]	500Bytes: IN 26.5 MB/s, OUT 29.5 MB/s 1500 Bytes: IN 28.0 MB/s, OUT 31.5 MB/s

Table 2: Results from the ping-pong and streaming measurements on the Fast Ethernet switch.

1. The PC software overhead is independent of the frame sizes. The overhead is introduced by the operating system on the PC (Linux) and communication with low-level driver of the NIC (MESH). The measured value is the sum of overhead introduced by the sending and receiving nodes.

2. The switch overhead for intra-module transfers is constant for different frame sizes. Therefore the module introduces a single store-and-forward and is using shared memory to transfer data from input port to the output port. The measured constant overhead can be attributed to the operation of the switch when making the routing decision.

3. The switch overhead for inter-module transfers consists of the constant part and the data dependent part. Therefore the switch makes at least two store-and-forwards for the inter-module communications. The first part can be attributed to the operation of the switch when making the routing decision. The data dependent part represents the bandwidth required by the frame for the inter-module transfers (the transfer speed over the backplane between source and destination modules). We were expecting that the inter-module measurements would confirm the manufacturer’s information that the transfer speed over the backplane was 4 MBytes/s. Assumption of this value and of two store-and-forward (in source and destination modules) did not agree with our measurements. After consultations with manufacturer, the discrepancies were cleared by introducing an additional store-and-forward. The measurements then confirmed the transfer speed over the backplane at 4 MBytes/s.

4. The streaming measurements show limitations for the inter-module transfers. We measured limits with two data sizes: 500 Bytes and 1500 Bytes. In the measuring setup we used 6 PCs: 3 were connected to the same module and remaining 3 were connected to a separate modules (we called this setup 3111). The setup is shown in the Figure 4. The PCs were communicating in pairs: Pca<-> PC1, PCb<->PC2, PCc<->PC3. The PCs were generating frames with fixed inter frame gap. There was no queuing neither in the PCs nor in the switch. We were expecting to record constant latency up to the moment when the offered load hits the limits in the switch. By shortening the gap we were increasing the load generated by the PC’s. Aggregated throughput accepted by the PCa, PCb and PCc shows a limitation in access from the backplane to the module (marked as ‘in’ in the Figure 3 legend). Aggregated throughput accepted by the PC1, PC2 and PC3 shows a limitation in access from the module to the backplane (marked as ‘out’ in the Figure 3 legend).

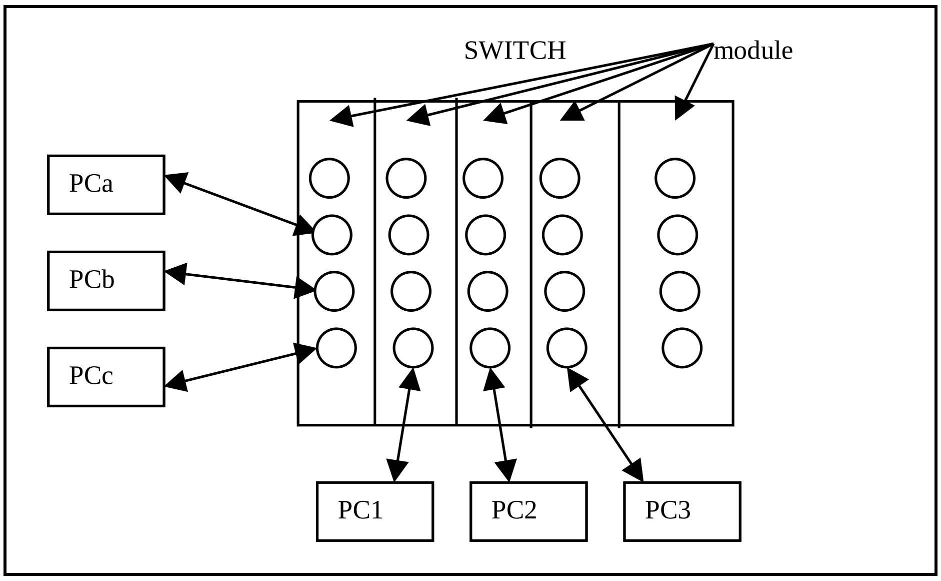


Figure 4: Setup 3111 for streaming measurements

2.3 Detailed model

Measurements taken on the switch together with consultations with manufacturer allowed us construct a fairly detailed model. We modeled the internals of the switch that may contribute to the throughput and the frame's latency. The structure of the model is shown in the Figure 5.

For the simplicity, only communication from MODULE A to the backplane and communication from the backplane to the MODULE B is presented. Arrows show the data flow direction when the frame is transferred from the MODULE A to the MODULE B.
Each module houses 4 ports. Data arriving on the ports are collected in the MAC (Media Access Controler) and stored in the SHARED BUFFER. For intra-module transfers the frames are fetched from the SHARED BUFFER and sent out through the corresponding MACs on the same module. For inter-module transfers frames are fetched from the SHARED BUFFER and stored in the OUT LINK. The backplane crossbar provides connection to the IN LINK of the destination module. Once the whole frame is collected in the IN LINK of the destination module, it is then transferred to the SHARED BUFFER. From the SHARED BUFFER the frame is sent out through the corresponding MAC. The data transfers to and from the SHARED BUFFER are provided by the bus. The data is transferred in small fragments and the throughput of the bus is big enough to provide uninterrupted data flow. The OUT FIFO and the MACs have internal FIFOs. They can start sending data as soon as they receive the very first fragment from the SHARED BUFFER. In such organization, the data transfers to and from the SHARED BUFFER are overlapped and pipelined and they do not contribute to the frame's latencies. Therefore we do not model operation of the bus.

When the frame is to be transferred to the other module, the very first fragment of data is fetched from the SHARED BUFFER and via the OUT FIFO is loaded to the OUT LINK. The transfer over the crossbar backplane it then started. If there is another frame ready for the inter-module transfer it then can be fetched from the SHARED BUFFER and loaded via the OUT FIFO to another OUT LINK. Such organization allows that at any time instant there might be more than one transfer from the module to the matrix.

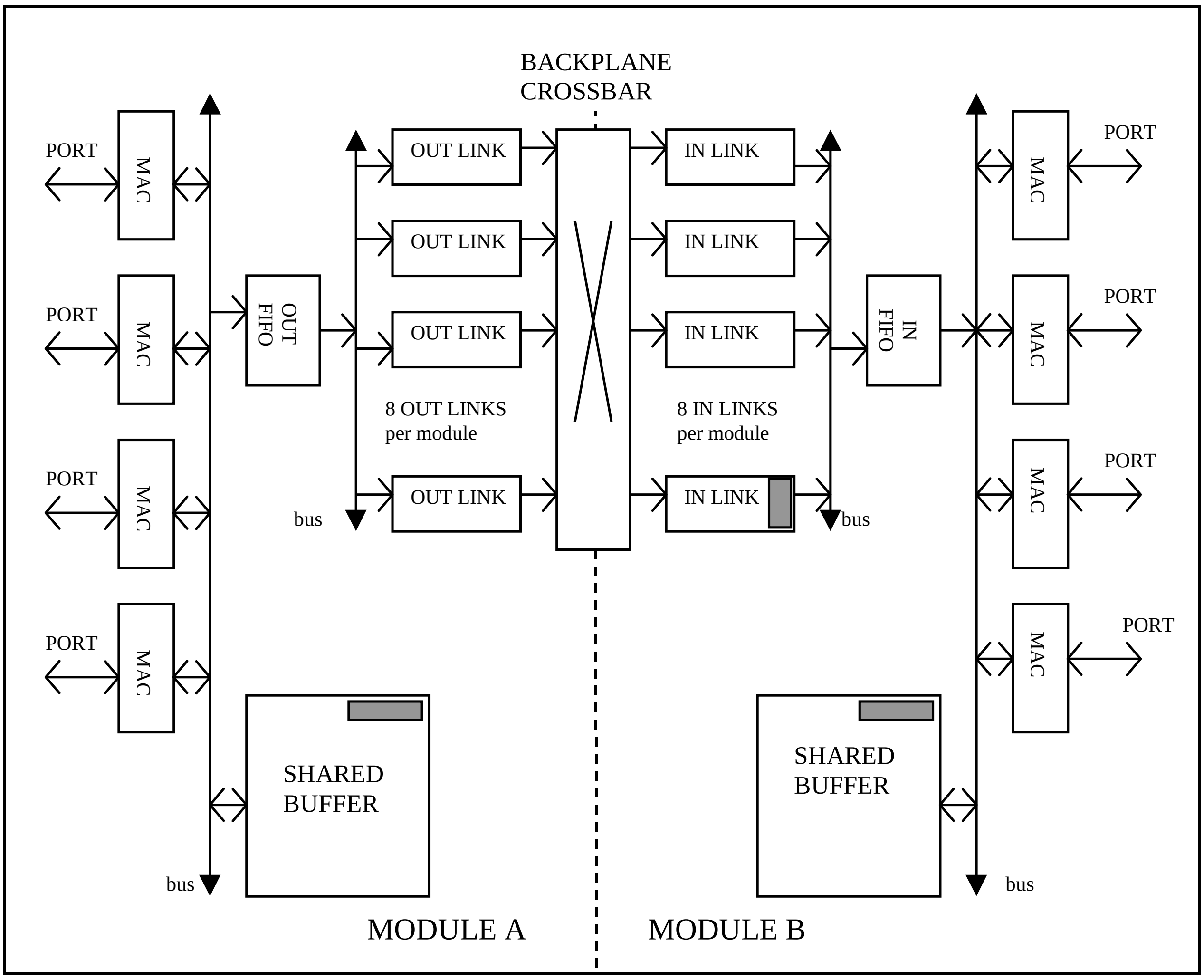


Figure 5: Organization of the detailed model. Shows 3 places where the frame is stored.

The same applies to transferring data from the IN LINK to the to the SHARED BUFFER. More than one IN LINK can receive data from the crossbar backplane, and when the full frame is collected in the IN LINK, it is then transferred in fragments to the SHARED BUFFER of the destination module. The parallelism in the inter-module transfers somewhat compensates for the relatively low transmission bandwidth for the single transfer (measured 4 MBytes/s). It is possible to have up to 8 simultaneous transfers active to and from the module.

There are three store-and-forward for the inter-module transfers in the model (and in the real switch). The first place where the frame is fully buffered is in the source module (SHARED BUFFER in module A). The second place is in one of the IN LINKs of the destination module. The frame in the inter-module transfer has to be fully received in the IN LINK before it will be moved into fragments to the SHARED BUFFER. The third store-and- forward is localized in the SHARED BUFFER of the destination module.

The operation of the model is based on the concept of resources. Frames arriving to the switch require resources to continue transfer. Components of the switch are modeled as resources which frames may acquire and hold for the time needed to carry out the transfer.

When the frame’s transfer finishes resources are released and can be allocated to another frame. The OUT FIFO, OUT LINK, IN LINK and IN FIFO have been modeled as resources. If resource is occupied and another frame wants to use it, the new frame has to wait until resource will be freed and will become available.

A frame arriving at one of the ports of the Module A is received by the MAC. The model of MAC implements FIFO and the data from the FIFO is moved to the SHARED BUFFER in small fragments while the frame is being collected from the port. When the frame’s transfer finishes the frame is fully buffered in the SHARED BUFFER. The occupancy of the SHARED BUFFER then increments. The frame waits in the buffer for 4 μs to model the fixed overhead introduced by switch for intra-module transfers. If the destination node is attached to the port in the same module the frame is then queued for that port. If the queue is empty the frame is then sent immediately to the MAC of the destination port and later transferred out of the switch. When the last bit of the frame is transferred out by the MAC the occupancy of the SHARED BUFFER is decremented.

If the destination port is located in the different module frame has to wait an additional time to model the switch fixed overhead for the inter-module transfers (measured as 12 μs). When the time elapses a number of checks are made to verify whether there are enough resources in the switch to start the transfer. The resources checked are the OUT FIFO and the OUT LINK in the source module and the IN LINK in the destination module. If the OUT FIFO is free and there is at least one OUT LINK free and at least one IN LINK free the frame is considered as being transferred. It occupies the OUT FIFO for the time necessary to transfer the whole frame with the bandwidth of 40 MBytes/s (the speed of the OUT FIFO we quoted by the switch manufacturer). The frame being transfered occupies the OUT LINK and the IN LINK resources for the time corresponding to its size and the transfer speed (4 MBytes/s). When the time elapses the frame is fully stored in the IN LINK of the destination module. To complete the transfer to the SHARED BUFFER of the destination module the IN FIFO resource must be obtained. If the resource is available, the frame is then moved to the SHARED BUFFER holding the IN FIFO for the same time as for the OUT FIFO (transfer speed of 40 MBytes/s) . If IN FIFO is allocated to another transfer - frame has to wait in the IN LINK occupying it for a longer time. When the IN FIFO resource becomes available the frame arrives at the SHARED BUFFER of the destination module. At that moment the frame releases allocated resources: OUT LINK, IN LINK and IN FIFO. Also the capacity of the SHARED BUFFER at the source module is increased.

The remaining transfer to the destination port is an intra-module transfer. In case of frame arriving to the module from the IN LINK (as opposed to from the local port), there is no need to model 4 μs fixed intra-module overhead as a fixed overhead has been already taken into account.

The model of the SHARED BUFFER has limited capacity. If the number of frames occupying place in the buffer reaches it’s capacity then the new frames are dropped. We did not model the flow control.

2.4 Comparison with measurements

Modeling results together with the measurements are presented in the Figure 6. The model has been verified with measurements in the setup with 4 PCs communicating between themselves at random (the destination PC was selected at random). The two PC were connected to the same module and other two were connected to the other module (see the title of the picture in the Figure 6 – ‘4 nodes. 2/2 nodes/module’). In such setup we have the mixture of inter and intra-module transfers. The PCs were set to generate frames at times from the exponential distribution. We conducted tests for two frame sizes: 500 Bytes and 1500 Bytes. We were changing offered load by decreasing the mean value of the exponential distribution.

In the Figure 6 we present latency of frames recorded at the destination PCs. The latency is presented as a function of accepted load. When there is no packet loss the accepted load is equal to the offered load. The results from modeling show very good agreement with measurements, especially for the lower loads. The model of the switch properly treats the inter and intra-module traffic and assigns correct latencies. For the higher loads, when the frames start to queue because of lack of resources, the results from modeling correspond to the measurements with a few percent of accuracy. The model properly identifies the load when the switch becomes saturated and the latencies start to grow significantly. At this point switch starts loosing frames in absence of the flow control. We accept the achieved accuracy as we do not plan to run the network and switches with loads close to the saturation.

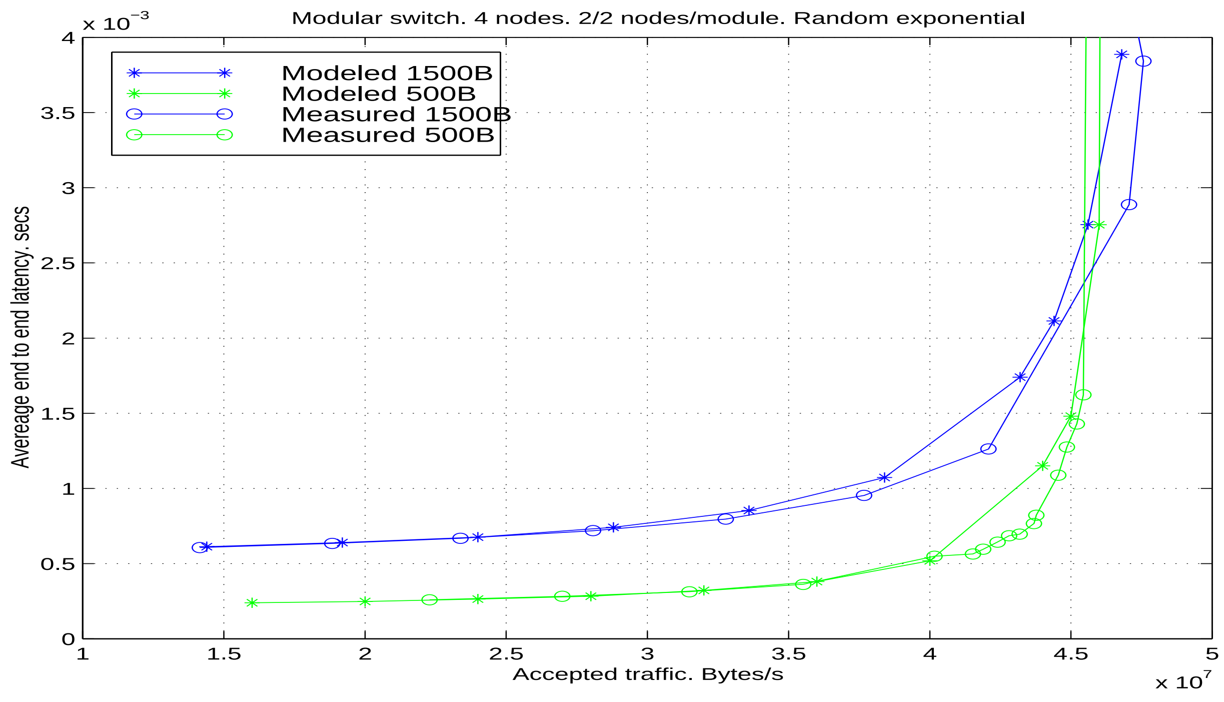


Figure 6. Average end-to-end latency measured in 4 nodes setup compared with results from modeling with the detailed model. Comparison was made with two data sizes: 500 and 1500 Bytes.

1.0 The parameterized model of the switch

The detailed model discussed previously helped us understand the operation of the Ethernet switch. In the model we could change different parameters and check their impact on the switch performance. We changed the bandwidth of a single inter-module transfer, the number of links from the module to the backplane and from backplane to the module, the number of ports in a module, the size of the buffer etc – this is impossible to do on the real device. This led us to identify a set of parameters having the biggest impact on the switch performance. The structure of the resulting parameterized model is shown in Figures 7 and 8.

The parameterized model reflects the hierarchical architecture of the switch. The Ethernet ports receive and transfer frames via MACs. The ports are grouped together and form modules. The intra-module communication is handled internally by the module. Transfer between modules requires the backplane.

To provide the transfer of a frame from the input buffer to the destination's port output buffer certain switch resources have to be allocated (buffering, transfer bandwidth). Parameters quantify resources in the parameterized switch. The operation of the model of the switch is characterized by calculations with limited number of parameters. In the following section we describe parameters in more detail. A description of how a single frame traverses the parameterized switch is given in section 3.2.

3.1 Description of the parameters

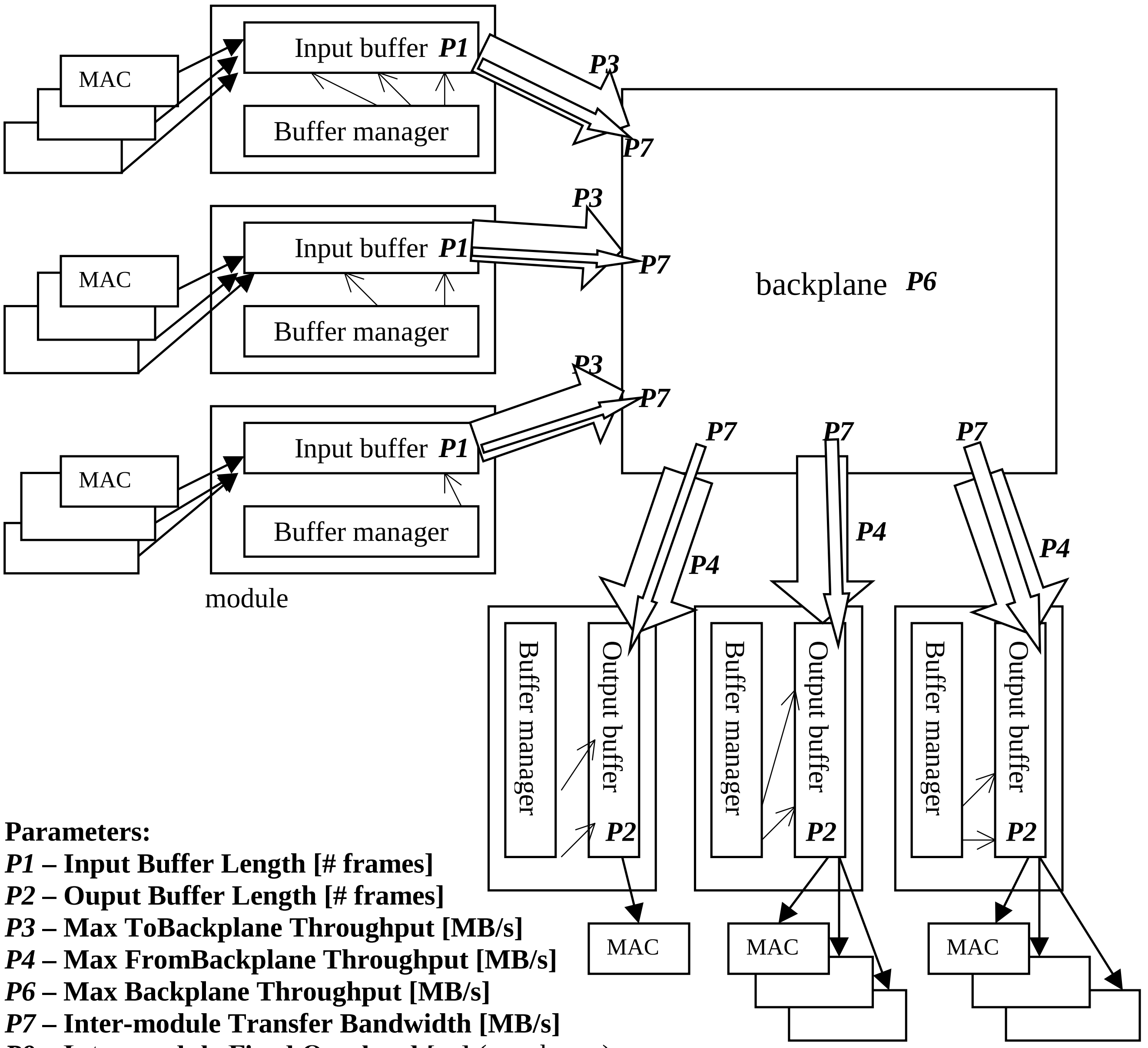
The following set of parameters has been identified as contributing most to frame latencies:

1. ***Parameter P1***: The length of the input buffer in the module. The length is expressed in number of frames. This parameter represents ability of the switch to buffer frames at the input. Frames are buffered in the input buffer for a time needed to make routing decision. Frames continue to occupy the input buffer in cases where there is not enough transfer resources in the switch to move frame from the input module buffer to the output module buffer. To avoid the head-of-line blocking the input buffer is managed by the buffer manager. The buffer manager may implement different policies when deciding which of the waiting frames will be transferred next.

2. ***Parameter P2***: The length of the output buffer in the module. The length is expressed in number of frames. This parameter represents ability of the switch to buffer frames at the output. After the frame reaches the destination module it is buffered in the output buffer. If the destination port is free the frame is send out via the attached MAC. The buffer is controlled by the buffer manager. The buffer manager may implement different policies when deciding which of the frames waiting for the particular port will be transferred next (for example it can organize buffer into high and low priority queues).

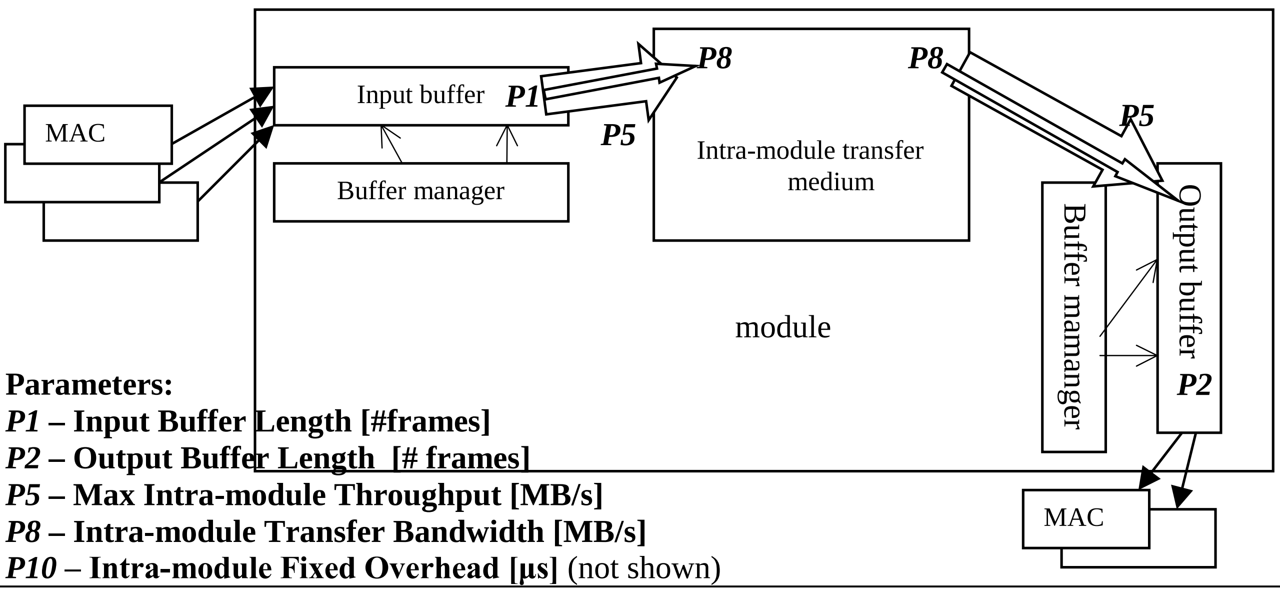
   Note – very often switches implement a shared buffer for both input and output. This results in a cheaper hardware design and more flexibility in the module. In such cases demarcation between input and output buffers changes dynamically. This however does not affect the concept of providing buffering resources at the input and output. The size of the buffer turns out to not be a critical parameter. In an overloaded network in the longer run, the buffering will be exhausted anyway.

Figure 7. Model of the parameterized switch for inter-module communication



Parameters:
***P1* – Input Buffer Length [# frames]**
***P2* – Ouput 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 8. Model of the parameterized switch for intra-module communication



Parameters:
***P1* – Input Buffer Length [#frames]**
***P2* – Output Buffer Length [# frames]**
***P5* – Max Intra-module Throughput [MB/s]**
***P8* – Intra-module Transfer Bandwidth [MB/s]**
***P10* – Intra-module Fixed Overhead [µs]** (not shown)

3. ***Parameter P3***: The maximum throughput for the traffic passing from the module to the backplane in the inter-module transfers. It is expressed in MBytes/s. This represents resource the module offers to the frames to get from the input buffer to the backplane. When a frame needs to be transferred from the input buffer it requests certain amount of bandwidth (see parameter P7). If such request, together with other requests from other frames currently being transferred, does not exceed the maximum throughput P3, the frame can start transfer. Setting the parameter P3 equal to the amount of bandwidth requested by the single frame models a single link between the module and the backplane. Setting it bigger, allows more than one simultaneous transfer from the module to the backplane.

4. ***Parameter P4***: The maximum throughput for the traffic from the backplane to the module in the inter-module transfers. It is expressed in MBytes/s. See the explanation of the parameter P3, but with the traffic flow from the backplane to the module.

   Note - In most switches parameters P3 and P4 will have equal values. However, there might be cases (like in the switch which we tested – internals shown in Figure 3), where these values may differ. In the switch from the Figure 3 transfer from the SHARED BUFFER to the OUT LINK in the module A was overlapped with data movement over the backplane (as soon as the first fragment of frame's data arrived to the OUT LINK it could start transfer over the backplane, and during that transfer collect remaining data from the SHARED BUFFER). However, the transfer from the IN LINK to the SHARED BUFFER in the module B could happen only when the whole frame has been collected in the IN LINK. Access to the IN FIFO could be assigned to only one IN LINK, whereas other frames in other IN LINKs would have to wait. This asymmetry resulted in different maximum throughputs measured for the ToBackplane and FromBackplane traffics.

5. ***Parameter P5***: The maximum throughput for the intra-module traffic. It is expressed in MBytes/s. See explanation of the parameter P3, but the traffic concerned is between the input buffer and the output buffer on the same module.

   Note - This parameter should be set to some high value in the switches where the input and output buffers share the same memory, and where the intra-module communication does not create bottle-necks (memory bandwidth is big enough). This parameter could be used in modeling switches without the hierarchical architecture – for example single switch unit with number of ports sharing a bus. The parameterized model can then be simplified to have only one module, housing all ports of the modeled switch, and model only intra-module transfers. In such case the parameter P5 would represent the maximum throughput of the bus.

6. ***Parameter P6:*** The maximum throughput of the backplane. The parameter P6 represents limitation in total number of simultaneous inter-module transfers. In the real switches not all transfers which could pass the limits represented by P3 and P4 will be able to start because of the limitations in the backplane throughput.

7. ***Parameter P7***: The bandwidth required for the single frame transfer in the inter-module communications. It is expressed in MBytes/s. It represent amount of bandwidth that has to be allocated in the switch resources for a transfer of a single frame from the input buffer in the source module to the output buffer in the destination module. This parameter is used together with the frame size in latency calculations for inter-module transfers.

8. ***Parameter P8:*** The bandwidth required for the single frame transfer in the intra-module communications. It is expressed in MBytes/ s. It represents amount of bandwidth that has to be allocated in the switch resources for a transfer of a single frame from the input buffer to the output buffer in the same module. This parameter is used together with the frame size in latency calculations for intra-module transfers

9. ***Parameter P9:*** Fixed overhead in frame latency introduced by the switch for the inter-module transfer. It is expressed in microseconds. It represents time spent by the switch when making the routing decision for the inter-module transfer.

10. ***Parameter P10****:* Fixed overhead in frame latency introduced by the switch for the intra-module transfer. It is expressed in microseconds. It represents time spent by the switch when making the routing decision for the intra-module transfer.

The parameters P3 and P4 represent the limitation in usable bandwidth for the traffic from the module to the backplane and from the backplane to the module respectively. The usable bandwidth may either be a bandwidth of a single link from the module to the backplane or an aggregate bandwidth of a number of such links. In the ideal switch the parameters P3, P4, P6 and P7 should form the equation: $M*(n*P7) = M*P3 = M*P4 = P6$; where M is the number of ports, n is an integer, the P7 represents a bandwidth of the single transfer and the P6 is the maximum throughput of the backplane. For the $n = 1$ we have a single link between a module and the backplane and it's usable bandwidth does not limit the single transfer. The $n \geq 2$ represents a number of links between a module and the backplane providing simultaneous transfer of n frames. In the real switches the measurements of the parameters P3 and P4 may show that the available bandwidth is sometimes lower than indicated from the link speed and that the limits are function of the frame size. Many switches transfer the frame in small cells adding some control information for every transferred cell. Using the link bandwidth for cell control information affects all frames independent of their sizes. Before the inter-module transfer starts the modules may negotiate the transfer using the same link and hence using some of its bandwidth. Taking some of the link bandwidth for negotiation is frame size dependent and limits more the available bandwidth for the smaller frames.

3.2 Principles of operation of the parameterized model of the switch

The operation of the parameterized model is based on calculations using parameters representing buffering and transfer resources in the switch. The operation also assumes that any limitations of the transfer resources in the switch can be modeled as input queuing. When the frame arrives to the switch a check is made to see whether there are enough resources to buffer the frame (if the current count of frames buffered in the buffer does not exceed the parameter P1). If the check is negative the frame is dropped. Once the frame is buffered in the input buffer the current count of buffered frames in the source module is increased and the routing decision is made. Depending whether it is intra or inter-module transfer the corresponding parameter: P9 or P10 is used to model the fixed overhead time for taking the routing decision. Currently there are 4 types of transfers: inter-module unicast, inter-module multicast, intra-module unicast and intra-module multicast. The type of transfer defines which resources will be necessary to start the transfer. In case of unicasts the resources for a single frame transfer from the input buffer of the source module to the output buffer of the destination module will be necessary. In case of multicasts resources for multiple transfers between and inside the modules will be necessary. The routing decision is followed by the calculations made on P3, P4 and P6 or P5 depending on the transfer type. The frame transfer is seen as a request to provide a certain amount of bandwidth needed to commence the transfer: in the inter-module transfers the requested bandwidth is represented by the parameter P7 and for the intra-module transfers the requested bandwidth is represented by the parameter P8.

Frames currently being transferred occupy some part of the throughput represented by parameters P3, P4 and P6 for the inter-module transfers and P5 for the intra-module transfers. Together with evaluation of the transfer resources another check is made to verify if there is enough buffering capacity in the output buffer of the destination module. If the available throughput is larger or equal to the requested bandwidth and there is buffering available the frame can start transfer. Newly inserted frames reduce the available throughput by a fraction corresponding to the parameter P7 or P8 depending whether they are inter or intra-module transfers. Also, the current count of buffered frames in the output buffer is incremented. Once the resources have been allocated calculations are made to get the occupancy time. The resource occupancy times are calculated based on parameters P7 and P8 and the frame size. The larger the frame size the longer the resource will be allocated.

When the available throughput is less than that requested the frame has to wait. It waits until the necessary resources become available (usually when another frame leaves the switch). If there are more frames waiting for resources it is up to the buffer manager to decide which frame will be transferred next. The buffer manager may implement different policies to take decision: the frame waiting the longest time, the high priority frame etc.

When the frame arrives to the output buffer of the destination module it frees the allocated transfer and buffering resources in the input buffer of the source module. It is then up to the output buffer manager to decide which frame from the output buffer will be sent out next. Similarly to the operation of the input buffer manager, the output buffer manager can implement different policies when making it’s decision When the frame finally leaves the switch via the MAC, the current count of buffered frames in the output buffer is decremented.

The allocation of resources for the multicast and broadcast might be different from the single frame transfer. The policy of handling the multicasts and broadcast is strongly bound to the switch and we have not found any generalization there. Currently the model creates a copy of the multicast (broadcast) frame for each remote module housing at least one destination port.

3.3 Implementation of the switch parameterized model

The model has been written in the C++ language using OO techniques in order to satisfy the project’s objectives for flexibility in adapting to different routing policies and future modifications to the Ethernet standard. The switch which holds the routing tables and defines the type of transfer, its components – models of modules and resources - are represented as objects.

Another class of objects used to model a frames' transfer through the switch are wrappers. These object are created any time the new frame arrives to the switch. There are four different types of wrappers depending of the type of transfer (inter-module unicast, intra-module unicast, inter-module multicast and intra-module multicasts). Currently broadcasts are handled as multicasts. When the type of transfer is determined the appropriate wrapper is created and the frame is attached to that wrapper. Then the wrapper object 'guides' the attached frame through internals of the switch. The wrappers 'know' what resource objects they should contact to provide the requested type of transfer. The wrappers also 'know' what objects should be contacted (the buffer managers) to let them know that the frame has to wait in case there was not enough resources to start the transfer immediately. Objects modeling the buffer managers may implement different priority policies on decisions concerning which frame should be the next to go when resources become available. Thanks to the OO technique the object implementing one policy may be very easy exchanged by another object with different policy. Such substitution does not require any change to the rest of software of other objects.

The easy substitution of objects applies also to objects representing modules and resources. They can be exchanged with other objects modeling modules but implementing different internal organizations. For example a module dropping frames in case of overflow can be exchanged by the module implementing flow-control (IEEE 802.3x) or by a module implementing trunking (IEEE 802.ad) or by the module which implements both. Thanks to the OO techniques we can easily (without any changes to the rest of software) extend behavior of our model of the switch by replacing modules with new modules implementing recent changes and upgrades to the Ethernet standard.

3.4 Comparison of parameterized model with measurements

The parameterized model has been verified on a number of switches. We measured parameters for 2 Fast Ethernet switches (FEA and FEB) and for 2 Gigabit Ethernet switches (GEA and GEB) from different vendors. The GEA switch was capable of housing modules with either single Gigabit Ethernet port or 8 Fast Ethernet ports. Measured parameters are presented in the Table 3.

We present here comparison results of the parameterized model running with parameters collected on the switch with the lowest bandwidth (FEB). This switch was the easiest to saturate having limited number of PCs. Further tests will continue and we will update this section with new results.

In the test setup we run 8 PCs communicating at random between themselves via the Fast Ethernet (FEB) switch. The latency measurements together with results from modeling are presented in the figure 9. In the parameterized model we used parameters from the Table 3. The PCs were set to generate frames of 500 Bytes and the inter-packet gap was generated according to the exponential distribution. Figure 9 shows average latency as a function of accepted throughput aggregated on all PCs. The good agreement has been reached up to 42 MB/s of accepted throughput. We speculate the reason for the measurements curve turning back on itself is that at the accepted load of 45 MBytes/s the switch starts loosing frames and accepted load decreases. The Figure 10 shows the measurements and results from modeling of the frame losses as a function of the offered load. The shows that the model and the real switch start loosing frames around 45 MB/s of offered traffic.

	FEA		FEB	GEA		GEB
# ports per module	4 Fast		8 Fast	1 Gigabit	8 Fast	8 Gigabit
# of modules in chassis	15		Single unit switch	4		Single unit switch
P1 [frames]	64[V]		Not known	1350[V]	672[V]	Not known
P2 [frames]	64[V]		Not known	1350[V]	672[V]	Not known
P3 [Mbytes/s]	31.5[1,S]	29.5[1,S]	NA	123.38[S]	Line speed	NA
P4 [Mbytes/s]	28[1,S]	26.5[1,S]	NA	123.38[S]	Line speed	NA
P5 [Mbytes/s]	Line speed		50[S]	NA	Line speed	?[2]
P6 [Mbytes/s]	400[V]		NA	1200[V]		?[2]
P7 [Mbytes/s]	4[V,P]		NA	109.9[P]	151.3[P]	?[2]
P8 [Mbytes/s]	-		12.5[P]	NA	Line speed	185[P]
P9 [µs]	12[P]		5[P]	5.4[P]	8.4[P]	NA
P10 [µs]	4[P]		NA	NA	4.9[P]	5.2[P]

Table 3. Parameters for parameterized model measured on two Fast (FEA, FEB) and two Gigabit Ethernet (GEA, GEB) switches.

Notes:

1. The max bandwidth for 1500 Bytes and 500 Bytes of user data are given.
2. We were not able to make measurements due to limited number of PCs or due to limits in the PCs

P. Ping-pong (COMMs1) measurements,
S. Streaming measurements
V. Information collected from the vendor

3.5 Conclusions

Analyzing results from the set of basic communication measurements we were able to identify internal structure of the COTS Ethernet switch. With the help from vendor we constructed detailed model of the switch. It helped us to identify parameters contributing the most significantly to the frame latency and the throughput when traversing the switch. We constructed a simplified model of the switch based on the identified parameters – the parameterized model. This shows good agreement with measurements. For the parameterized model we demonstrated it’s applicability to the wide range of switches with different internal and hierarchical architecture. The parameterized model applies to the class of switches characterized as: modular architecture (switch is composed of modules communicating by backplane) and is type of store-and-forward (with two stages of buffering frames: in the source and in the destination modules). The model has been originally developed with OPNET [5] modeling environment. It was written in C but later converted to C++ to provide better flexibility in extension of the model for new developments and standard modifications . After conversion the model is still used with the OPNET environment. It also was used for modeling the full scale ATLAS LVL2 system using the PTOLEMY [5] [6] modeling environment.

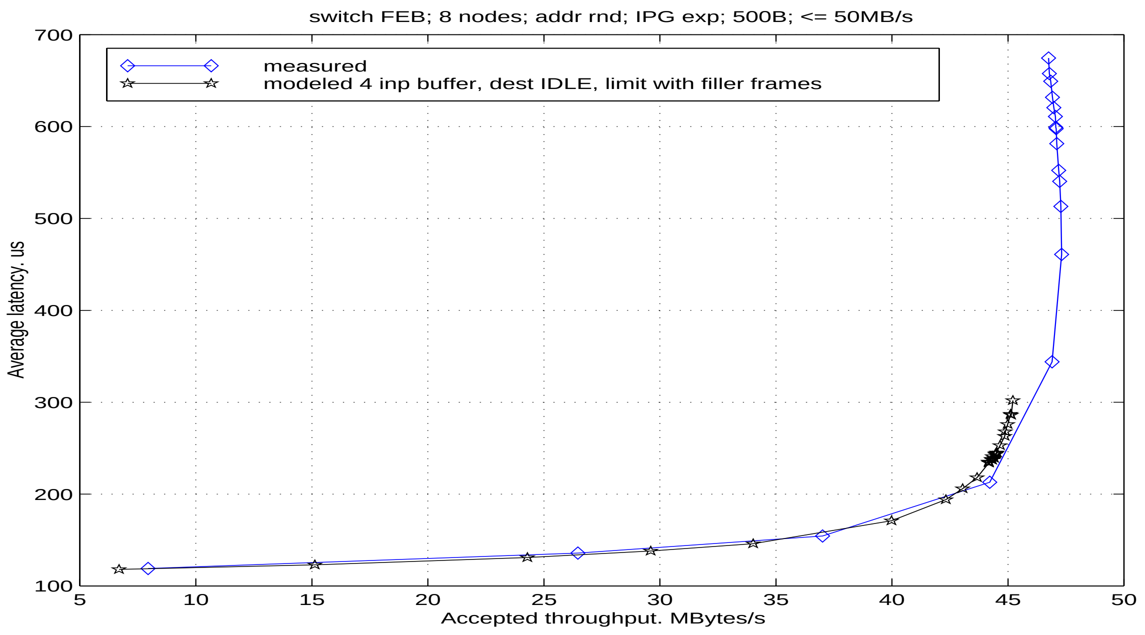


Figure 9. Average latency as a function of accepted throughput from measurements and from modeling with parameterized model. Tests were performed on Fast Ethernet single unit switch FEB (see table 3 for parameters). In the setup there were 8 PCs generating random destination address and using exponential distribution for Inter Packet Gap (IPG). The tests were performed for the 500 Bytes frames**.**

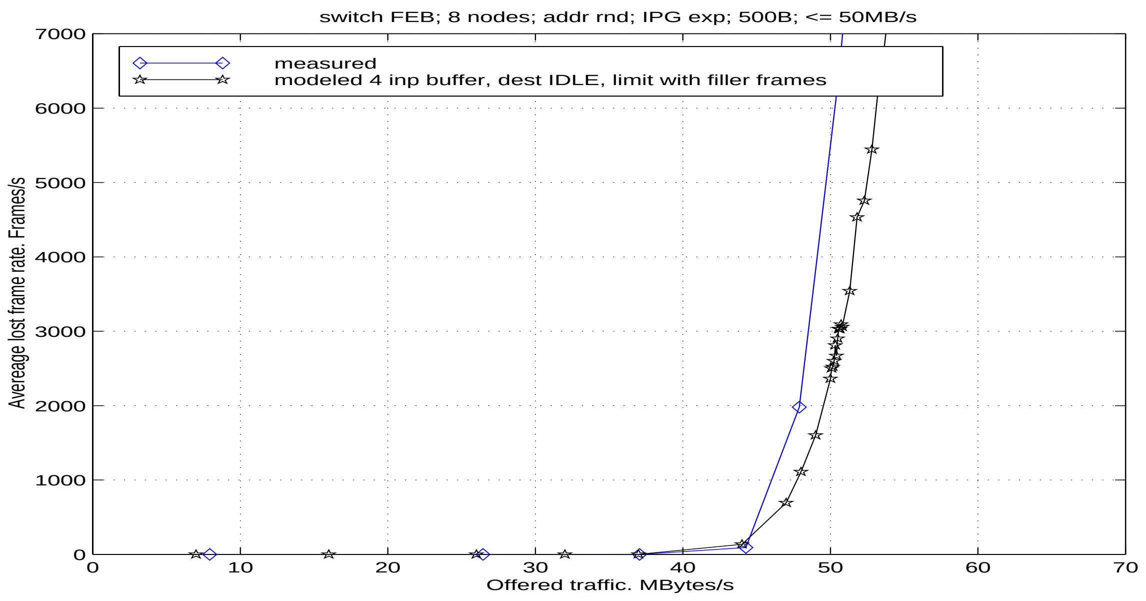


Figure 10. Average lost frame rate as function of offered load from measurements and from modeling with parameterized model. Tests were performed on Fast Ethernet single unit switch FEB (see table 3 for parameters). In the setup there were 8 PCs generating random destination address and using exponential distribution for Inter Packet Gap (IPG). The tests were performed for the 500 Bytes frames.

4.0 Modeling the ATLAS LVL2 system network using multi layered architecture of the Ethernet switches.

As a possible architecture for the ATLAS LVL2 system we investigated the ‘Central switch’ architecture. The architecture is presented in the figure 11.

Buffers with data from detectors (ROBs) are grouped in partitions. The partition is a group of ROBs serving for the same sub-detector. All ROBs from a common partition generate data with the same frequency and the same data size. The ROBs are connected to the concentrating switches by the Fast Ethernet links. From a single concentrating switch there is one Gigabit Ethernet up-link connected to the central Gigabit Ethernet switch. At the bottom of the picture there is another layer of concentrating switches for connecting the processing nodes. There is only one partition of processing nodes.

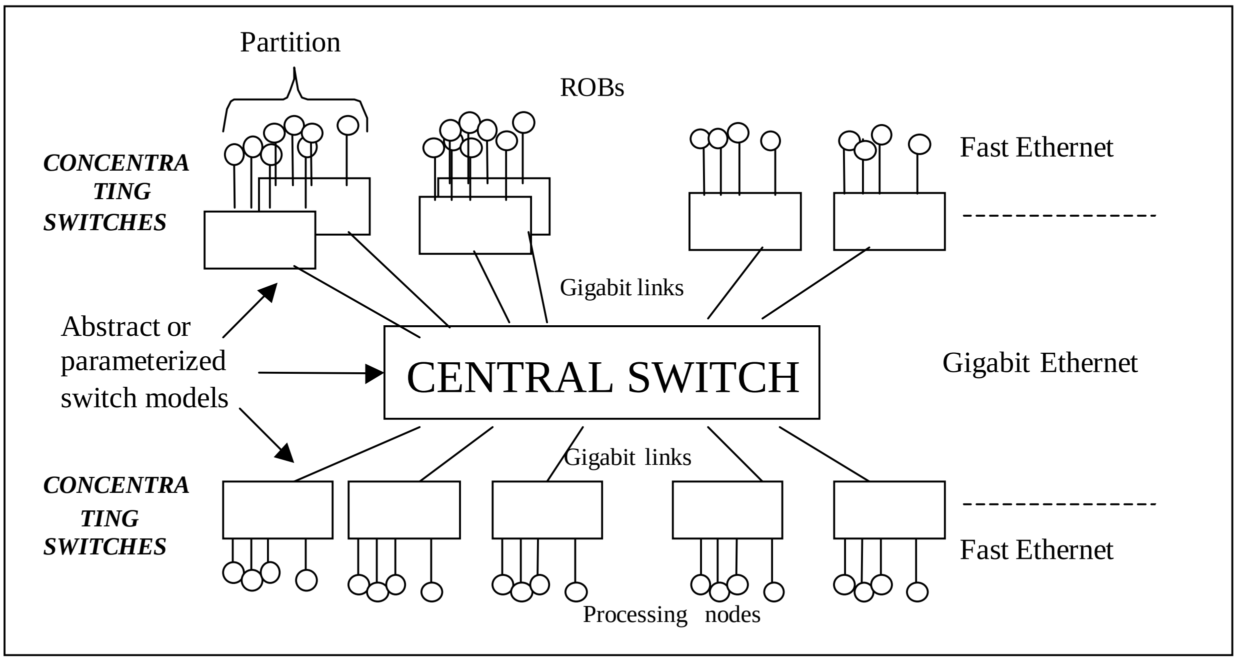


Figure 11. Central switch architecture

4.1 ‘Central switch’ architecture

As a first stage evaluation we modeled the ‘central switch’ architecture using Fast and Gigabit Ethernet links and only abstract models of switches. This abstract model of switch provides an immediate transfer of the frame from the input port to the output queue (single store-and-forward). There is no modular structure and only the output queues are modeled. Thus the abstract model of the switch presents a non-blocking operation with an infinitely fast backplane. As we watch the switch technology progress there are more and more non-blocking switches on the market and the backplane throughput increases into Gigabit/s performance. Using the abstract model of switches will allow us to show the limitations on the performance of the architecture introduced by the Ethernet links and queuing. We refer to these results as to the best case scenario (when using plain IEEE 802.3 standard without any extensions).

Models of ROBs, and processors were very simple. Processors generated requests to randomly chosen ROBs. A ROB responded to the requesting processor by generating a frame of a size assigned to its partition.

The numbers of ROBs and processors constituting the architecture were taken from the ATLAS paper model studies [1]. The average amount of data generated by each ROB or accepted by a single processor were also taken from the paper model studies for different physics modes: low and high luminosity.

4.2 Network components

The low luminosity mode with B-physics creates the biggest demand for bandwidth in the ATLAS LVL2 system to transfer data from the ROBs to the processing nodes. In the absence of primary ROI guidance the full TRT scan has to be performed to identify secondary ROIs. To process data from the TRT scan without increasing LVL2 average latency, a few hundreds of processors is required.

In our studies we use these conditions to determine details of our architecture. The average volume of ROI data per ROB (the average request frequency per ROB and the average data size fragments) were taken from the paper model studies [1] for low luminosity with B-scan. From the same source we took the number of processors and the average amount of data requested by the processor. These numbers were used to calculate number of concentrating switches as well as the number of ports in switches. The average load allowed on a single Gigabit up-link is used as a parameter in this calculations. We studied different levels of concentration by changing the parameter from 80 MBytes/s up to the throughput when the switches' queues occupancies start to grow rapidly.

The model deals only with single frame messages. For the calorimeters, the average data size is bigger than the maximum size of the Ethernet frame. For this study, the maximum Ethernet size was assumed for the calorimeters, but the request frequency was adjusted accordingly (increased) to retain average data volume per ROB from calorimeters.

The collection of the parameters used in determining the network (taken from the paper model studies) is presented in the top part of the Table 4. The average frequency of requests generated by the processing nodes is based on the average frequencies of the ROBs taken from the paper model and total number of processing nodes. The processing nodes generate requests at random.

In the bottom part of the Table 4 there are results from the components calculations. Four sets of configuration data (in terms of number of switches and numbers of ports per switch) were calculated depending on the average load put on the Gigabit Ethernet up-links (from 70 MB/s to 100 MB/s – column 1 in the bottom part of Table 4). For every partition there is a number of switches needed to accommodate traffic from the partition with number of ports per switch (columns 2-6 in the bottom part of Table 4). Number of ports in the switch includes a single Gigabit Ethernet up-link. In the brackets there is a number of ports used in the last switch (number of ports which were left over after all previous switches were fully connected – the number also includes the Gigabit Ethernet port). In the last column the number of Gigabit Ethernet ports needed in the central switch is presented.

In the Figure 12 we present distribution of latency for frames traveling the network from ROBs to the processors. The peaks in the distribution correspond to the latencies of frames arriving from partitions with the highest rates. For low values of the Gigabit up-link load the peaks are sharp what represents queuing in the partition's concentrating switches.

As the allowed load on the Gigabit up-link increases the peaks become less sharp signifying queuing at the processing nodes.

	pixel	**sct**	**Trt**	**em cal**	**had cal**	**μ-trigger**	**μ-prec**	**processors**	
Number	84	92	256	760	98	48	192	768	
Load [MB/s]	1.23	3.0	8.08	1.11	4.0	0.26	0.25	8.0	
Frequency [kHz]	11.0	10.63	10.33	0.74	1.33	0.64	0.60	7.01	
Frame size [bytes]	110	280	780	1500	1500	410	833	100	
Ethernet Gigabit up-link average load	in the cells below: number of switches * number of ports in the brackets: number of ports connected in the last switch of the sub-detector								nb of Gigabit ports in central switch
80 MB/s	2 * 66 (20)	4 * 27 (15)	29 * 10 (5)	11 * 73 (41)	5 * 21 (19)	1 * 49	1 * 193	77 * 11	130
90 MB/s	2 * 74 (12)	4 * 31 (3)	24 * 12 (4)	10 * 82 (32)	5 * 23 (11)	1 * 49	1 * 193	70 * 12	117
100 MB/s	2 * 82 (4)	3 * 34 (27)	22 * 13 (5)	9 * 91 (41)	4 * 26 (24)	1 * 49	1 * 193	64*13	106
110 MB/s	network becomes unstable (queues length grow infinite)								

Table 4. Composition of the central switch architecture for low luminosity

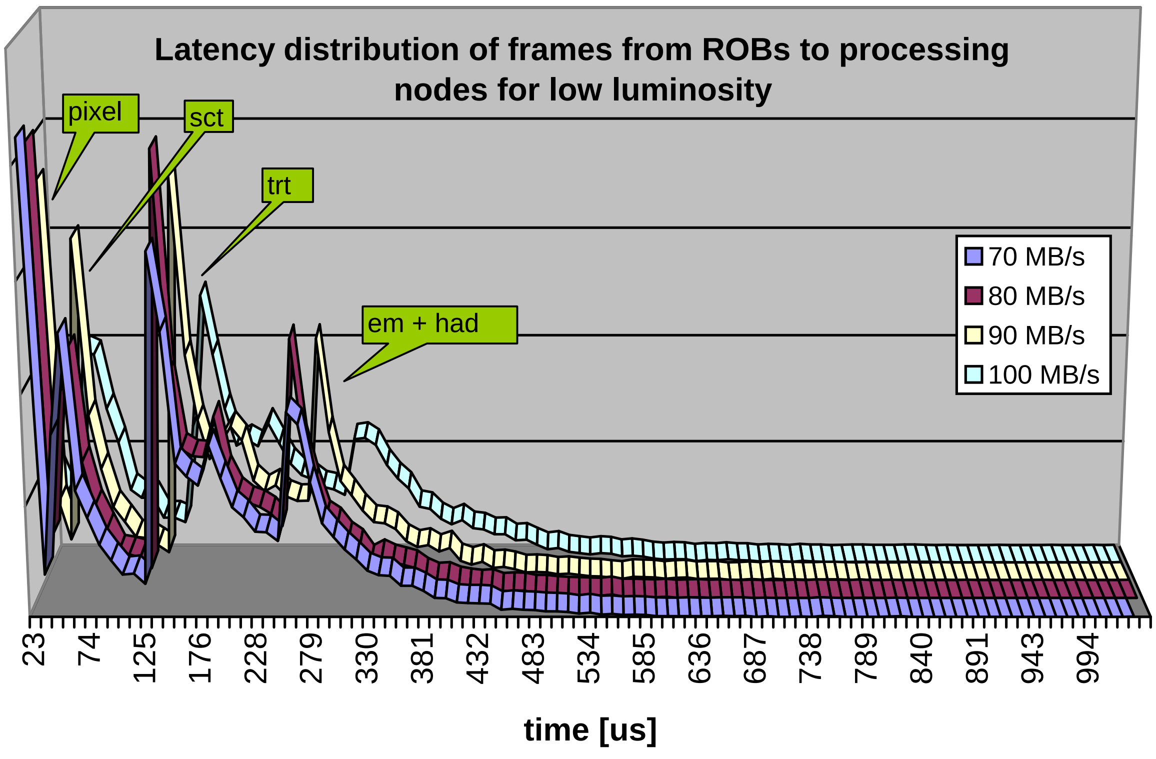


Figure 12. Latency distribution for low luminosity as function of allowable load on Gigabit up-links from concentrating switches

Before the LVL2 system accepts or rejects an event it has to collect information from a number of ROBs corresponding to a ROI region (it may be more than one ROI per event). As an example of the ROI collection we modeled a process of gathering data from four ROBs. To evaluate latency distribution of the ROI collection one of the processing nodes was selected to generate specially marked requests. The node was generating four frames directed to randomly chosen four ROBs from the same partition (mimics the ROI collection process from the sub-detector). Frames were queued in the processing node (there was no broadcast) and later delivered to the ROBs. The ROBs responded by sending back frames to the requesting node. When the frame from the last ROB (out of the four requested ones) arrived to the processing node the time of the whole process was registered. The latency distribution of the ROI collection for the electromagnetic calorimeter and the low luminosity are shown in the Figure 13.

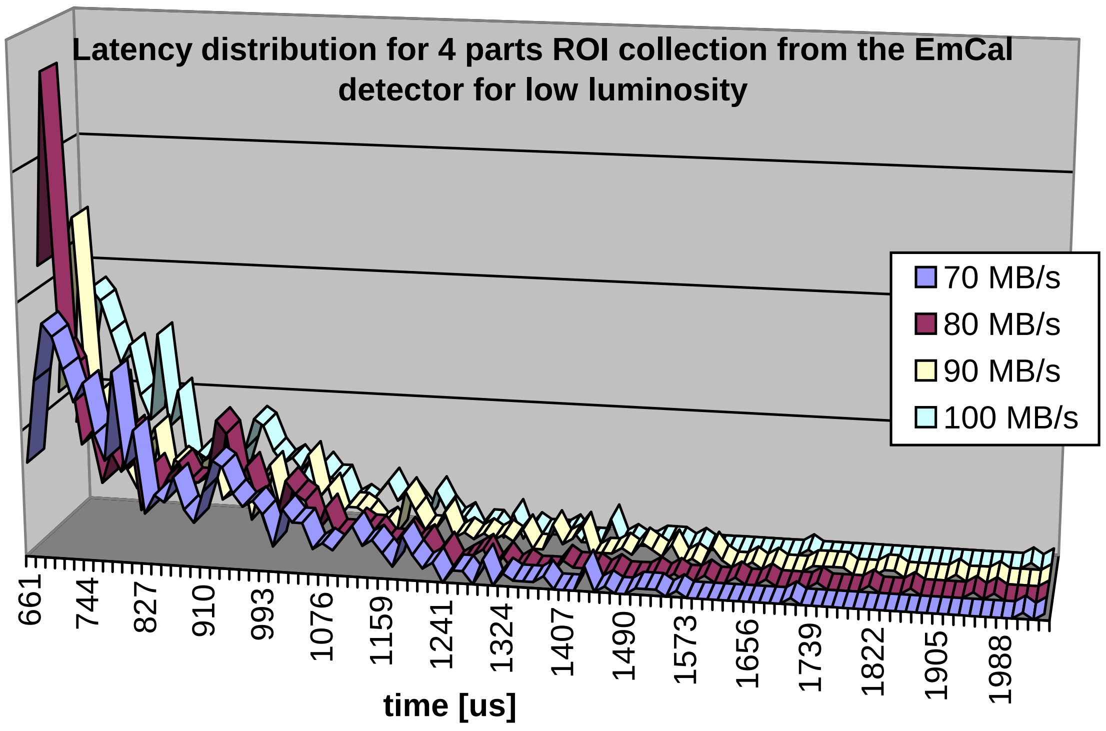


Figure 13. Latency distribution for 4-parts ROI collection process from electromagnetic calorimeter in low luminosity physics conditions.

It should be noted here, that the most demanding data collection is the TRT-SCAN. The data from the whole TRT detector has to be collected in the single processor. Back-of-the envelope calculations show that collecting data from 256 ROBs in the 'central switch' architecture takes roughly at least 19 ms for the average ROB data size of 760 Bytes. This is mainly due to serial transfer of all the data through the port of the central switch connected to the concentrating switch to the processor requesting the scan.

4.3 Conclusions

We constructed the model of an example of the architecture for the ATLAS LVL2 system. We were using the simplified models of switches presenting non-blocking operation of the switches. The aim was to show limitations imposed by the architecture (central switch) and the links (Fast and Gigabit Ethernet). The presented plots show the 'best scenario' (section 4.1) the selected architecture and the selected technology can achieve. As the project is ongoing we plan to use the parameterized models of switches currently on the market.

References:

1 “Paper modeling of the ATLAS LVL2 trigger system”, J.Bystricky, J.C.Vermeulen, ATLAS Internal Note, ATL-COM-DAQ-2000-022,

2 “MESH: Messaging and ScHeduling for Fine-Grain Parallel Processing on Commodity Platforms”, M.Boosten et al, presented by M.Boosten at the International Conference on Parallel and Distributed Processing Techniques and Applications: PDPTA99, Las Vegas, USA: 28 June – 1 July 1999, http://home.cern.ch/mboosten/

3 “The measurement software and clock synchronization, F.Saka http://fsaka.home.cern.ch/~fsaka

4 “The measurement procedure”, F.Saka http://fsaka.home.cern.ch/~fsaka

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

6 “The PTOLEMY Project”, Department of EECS, UC Berkeley, USA http://ptolemy.berkeley.edu

7 “Ptolemy simulation of the ATLAS level-2 trigger”, P.Clarke et al, ATLAS Internal Note, ATL-COM-DAQ-2000-020