# Real-Time Diagnostics at ASDEX Upgrade - Architecture and Operation

K. Behler, H. Blank, H. Eixenberger, A. Lohs, K. Lüddecke[1)], R. Merkel, G. Raupp,
G. Schramm, W. Treutterer, M. Zilker, and ASDEX Upgrade Team

*Max-Planck-Institut für Plasmaphysik, EURATOM Association, Boltzmannstr. 2, 85748 Garching, Germany*

*[1]UCS GmbH, Seeshaupter Straße 15, 82393 Iffeldorf, Germany*

## Abstract

Diagnostics at ASDEX Upgrade have available a very large number of highly developed measuring channels. The prospect of making this wealth of information usable for plasma optimization led to the implementation of a number of diagnostics running data acquisition in real-time (RT). Ultimately, this development aims to achieve a network of intelligent diagnostics delivering analysed data for high-level plasma performance control such as profile shaping and NTM stabilization.

The new RT diagnostics consist of standard industrial 19" servers organized in clusters and running a standard UNIX multiprocessor RT-capable operating system (RT OS). Built-to-purpose computer interface cards deliver data (e.g. via serial links) from the data acquisition (DAQ) front-ends directly into the main memory of the DAQ servers. An RT data analysis task immediately following the running direct memory access (DMA) data transfers processes the data and delivers the results to follow-up systems in the control chain.

Whereas the first systems were implemented in a simple JBOC (just a bunch of computers) configuration being operated as a number of single diagnostics, newer systems are integrated into diagnostic clusters using parallel computing techniques such as message passing interface (MPI).

The paper describes the hardware (ADC front-ends, serial I/O, selection criteria and performance of the involved computer busses and systems) and software (DAQ, DA, RT OS, MPI) architecture of the assembled systems. Benchmark results for DAQ and MPI bandwidth and latencies as well as for the behaviour of the RT OS will be given.

*Keywords*

Real-Time, Data Acquisition, Data Analysis, serial I/O, Hot-Link, RT Solaris

## Introduction

Diagnostics at ASDEX Upgrade (AUG) are implemented using a distributed architecture of autonomous data

acquisition systems [1]. The modular concept of using autonomous stations for each diagnostic was meant to address modular set-up and configuration, independent but coordinated operation, and distributed data acquisition and analysis, avoiding bottlenecks in single central systems. This underlying philosophy along with the fundamental software concepts remained essentially unchanged over 17 years, in spite of the fact that nearly all of the implementing components (front-end hardware, computer systems, networking components) have been changed. Due to common configuration and access methods [2], diagnostics control schemes, and data archive structures [3], which were persistently maintained throughout all changes, it was never a problem to operate old and new diagnostics side by side or to access data of all diagnostics in the same way.

The RT methods were initially introduced as the best solution for acquiring large amounts of data from the Mirnov Probe and Soft-X-Ray diagnostics [4]. The possibility of including RT data analysis for purposes such as MHD mode activity recognition, temperature and density profile evaluation, etc. was envisioned as a long-term prospect, but initially seemed too difficult to accomplish. With a plasma discharge time of only 10 seconds we were concentrating on providing access "in time" between shots instead of "real-time" during a shot.

The overhaul of the AUG Control System completed in 2006 [5], opened the possibility of feeding externally acquired and preprocessed data as RT inputs to performance optimization processes [6]. This led to a demand for delivery of results from RT DAQ and RT Analysis as input into Control. As prerequisites for this coupling of DAQ and Control the following topics have been addressed: 1st the need to develop an RT compatible communication method between DAQ and Control, 2nd the necessity to fully promote the RT qualities of the RT DAQ systems and to develop them to a standard conforming to a reasonable responsiveness and reliability as required for by Control, and 3rd the specification of a high performance parallel computing method to achieve a single result stream pace-keeping with real time from the distributed computer nodes of a multi-node RT diagnostic.

## Real-Time DAQ and Analysis Architecture at AUG

The term RT-DAQ is used at AUG when the DAQ system is able to transfer the samples from a number of analogue input channels directly into the main memory of a computer in a "pace-keeping" way, and to allow accessing this just measured data shortly afterwards in real time (i.e. with a defined short latency) by some analysis processes. "Pace-keeping" means that the propagation of data through the computers I/O-devices and -busses into the main memory keeps pace with the creation of samples in the front-end input channels[1] so that no samples are

1 An additional requirement would be to keep pace also transferring data further to long term storage. This would enable long pulse or steady state RT DAQ. However, this is not crucial for ASDEX Upgrade since with short (10 s) pulses one can keep all data in memory and has time to cleanup buffers and write shot files to the archive after the RT phase is over and the experiment is finished.

lost. The term “latency” is defined as the maximum time delay between sampling an input signal and its value appearing at a certain main memory location ready for processing[2].

An ASDEX Upgrade diagnostic implementing this kind of pace-keeping high-speed DAQ was first implemented in 1999 [4]. Further similar systems were developed in the following years. These systems were based on the following major RT-capable building blocks: the analogue front-end devices, the computer interface devices, the computer platform (OS and hardware), and the RT software layers for data transportation and data analysis. Figure 1 gives an overview of the principal hardware elements of AUG RT diagnostic systems. It is structured in three layers: 1st the analogue front-end layer with one or more crates, 2nd the computer I/O layer represented by the cPCI or cPCIe (cPCI(e)) interface cards, and 3rd the computer system. The layers are interconnected by standardized links. The connection between analogue front-end and the I/O devices is based on the HOTLink II architecture by Cypress [7]. Here bidirectional fibre optic links connect the “yellow card” in the front-end crates to serial I/O-cards (SIO-cards, see below) in a cPCI or cPCIexpress (cPCI(e)) crate representing the computer I/O layer. The connection between the external cPCI(e) crate and the computer host is a bus extension based on commercial PCI(e) to cPCI(e) bridge design, e.g. [8].

This architecture in layers introduces a modularity and makes the whole set-up of diagnostics - being very specific in the input channels - completely independent from the fast developing commercial computer world. The 2nd cPCI(e) layer mainly utilizes commercial components. With the cPCI form factor it implements a robust mechanical and electrical platform and an expected long-term standard. As extender of the internal computer bus it allows the physical separation of the I/O-cards and offers an augmented number of slots. A configuration with a one unit high 19” powerful server, as for instance shown in Figure 1, would allow up to 4 SIO-cards in the cPCI(e) crate, thus enabling the connection of up to 16 front-end crates with up to 64 channels adding up to a theoretical limit of 1024 input channels per 19” computer. Having RT requirements in mind and this flexible cPCI(e) architecture at hand, a family of cPCI(e) I/O interfaces was built [9]. The SIO-card is its newest member.

### *Analogue Front-End Configuration*

The requirements for analogue input channels in fusion research are often highly specialized and require an individual design of front-ends. To address this need a solution providing high flexibility and simplicity for custom built input devices was developed. This consists of an active backplane for analogue or digital input cards and a

2 Since latency is mostly determined by the (dis-)continuity of data transport caused by hardware arbitration, packet size, depth of intercostal buffers, and system reaction (PCI-bus, DMA logic, OS response), a short latency requires immediate start transfers (DMA, no buffering) and short interruptions (< 100 µs, IRQ response times) of data transport on the way from the input socket to main memory.

generic 16 bit parallel interface. The backplane serves as a mechanical and electrical support for the modules; additionally it is an easy to configure fast data multiplexer for a flexible number of input modules and channels. The backplane is built as a macroscopic 16 bit parallel shift register or "pipeline" with 20 insert positions. Each insert point (socket) on the pipeline features a tri-state 16 bit buffer, which can be switched to take its input either from the input card in the socket or from its next neighbour left on the backplane. The two phases "load" and "shift" of the backplane are illustrated in Figure 2. The logic to switch the tri-state inputs and to clock the 16 bit sample propagation from buffer to buffer (i.e. the pipeline control) is situated at the end of the backplane in a so called "yellow card".

As already mentioned the communication between the "yellow card" and the SIO is bidirectional. Using the downlink the "yellow card", the pipeline, and eventual internal functions of the analogue input cards are controllable via the SIO and all the clocks and timing in the front-end can synchronously be derived from the centrally synchronized experiment clock[3] available on the SIO because of its on-board TDC (time to digital converter). The "yellow card" with its FPGA decodes commands and data from the SIO, synchronously triggers the sampling of the input cards upon sample requests recieved from the SIO, and hosts the serializer and the fast up-link logic to send the acquired samples from the pipeline to the SIO. Also the timing for the various clock lines on the backplane is generated by this FPGA. Having the backplanes timing completely derived from the SIOs TDC clock provides additional RT-relevant features: deterministic control and exact measurment of the sampling time and rate by the computers timing hardware (UTDC or TDC, see below), and synchronized sampling of signals throughout distributed computers and diagnostics (all UTDCs behaving synchronously).

Figure 3 shows how the backplane can be populated with input and "yellow" cards. Additional up-link bandwidth for sampling rates from 500 kHz up to 2 MHz can be achieved by configuring additional yellow cards.

This overall data flow from analogue input to the computer satisfies strictest real-time requirements: no accumulation of samples in the front-ends, no unavoidable delays by buffering and multiplexing, and an easy to guarantee deterministic behaviour.

### *The SIO-card – a generalized serial I/O-interface with TDC*

The central link in the RT chain from the measuring sources to the computer memory is a serial HOTLink II

3 The central experiment clock is a 50 MHz timing signal distributed from a central timer via a star-topology fibre optics network to each system that has a UTDC or TDC timing module. The central experiment time is a 64-bit count (counting nanoseconds since start) which is distributed over the experiment clock and available as a synchronized internal multipurpose time base in the UTDC or TDC timing modules.

interface which makes the connection between the ADC front-ends and the computer's I/O-bus logic.[4]

The most recent FPGA/PCI design combines the development lines of the UTDC timing board and the HOTLink interface, into the so called SIO-board, which allows a centrally synchronized timing of serially attached analogue front-end devices.

Functional blocks on a SIO-board are: 1. the central FPGA, 2. four bidirectional 500 Mb optical links together with a quad HOTLink II serializer / deserializer, 4. the fibre optics timing clock receiver, and 5. a pluggable cPCI- or cPCIe-controller. All the logic blocks of the SIO-board are implemented in the FPGA and can cooperate in a configurable but deterministic way. The TDC block acts as synchronized time counter, derives the sampling clock from the central experiment time, and acquires time stamps for each sampling event. Figure 4 schematically shows how the data management logic merges the data streams from the serial inputs with a stream of time stamps generated by the TDC block to produce a combined stream of "data frames". Each "data frame" consists of a 64bit time stamp from the TDC together with the 16bit samples of all channels belonging to this particular point in time. The maximum configurable number of 16bit samples in a time slice is 4x16 or 4x64 if additional multiplexing in the analogue front-end cards is enabled. The various interface cards exist in different layouts to match the popular PCI-bus formats.

### *Hardware Platform and OS*

The standard platform for AUG DAQ systems[5]is Solaris on SPARC workstations, ranging from cheap uniprocessor desktop machines to 19" standard industrial server configurations. The latter systems have a number of high performance features such as two or four processors, multiple high-speed PCI-busses (PCI-X, PCIe), an optimised memory configuration providing high bandwidth, and a powerful interconnect between the system components avoiding bottlenecks in intra-machine data traffic [13].

There exist only a small number of specialized real-time operating systems (RTOS) (such as VxWorks), which satisfy the RT requirements posed by control systems. For RT DAQ, on the other hand, the requirement is not to

4 The present development emerged from the first HOTLink serial I/O-cards developed in 1999 [10] for the Mirnov and Soft-Xray diagnostics in combination with the ideas developed for the Universal Time-to-Digital Converter (UTDC) [11]. The UTDC was built for the renewal of the ASDEX Upgrade control system and to satisfy the need for a new central experiment timing system. Follow-up developments featuring an FPGA based PCI-card as a flexible digital I/O solution led to the creation of the DIO-board [9], and the Transputer-Link to PCI-bus interface [12].

5 Solaris had been our standard OS environment for over a decade and so there was no tendency to diversify in any of the LINUX, Windows, or VxWorks directions without necessity.

react rapidly to new situations but to keep pace with the incoming data streams and if possible to perform a subsequent pace-keeping analysis with low latency. Sun Microsystems has claimed for many years that its Solaris operating system exhibits reasonable RT behaviour. With this prospect and after other explorations of Solaris for RT DAQ [10][9], we decided to keep with Solaris also for the current RT DAQ and Analysis project.

## Software Architecture for RT Analysis

The implementation of diagnostic data acquisition and analysis software at ASDEX Upgrade follows the standardized specification for ASDEX Upgrade diagnostics as established in the original design [1]. This scheme specifies the description of hardware devices, data objects, and relations between them. It also provides for placing the previously defined object in lists which serve as directives to conduct the operation of a diagnostic through the three DAQ phases of a plasma discharge described below. New objects and features have been defined extending the AUG description standard to implement elements needed for RT DAQ. Providing the appropriate software drivers, RT devices can be members of the object lists in the shot file header (SFH) like other DAQ devices and data objects. RT device capabilities and relations to other shot file objects are configured as usual, and other SFH objects such as signals, mapping functions, and parameter sets can be used for the corresponding RT purposes as well.

### *The Real-Time DAQ phase*

The operation of diagnostics throughout a plasma discharge mainly takes place in three phases: 1. a “prepare” step which is initiated by distribution of the next shot number, 2. an “acquire” step following the prepare just before the plasma discharge, and 3. a final “archive & restart” step, which collects the acquired data buffers and writes them transformed into a shot file into the archive. Finally the diagnostic restarts itself and waits for the next experiment cycle to be started by distribution of the next shot number.

During the “prepare” step a diagnostic typically initiates the devices. RT diagnostics additionally negotiate with plasma control about the configuration parameters for the next shot, set the operation modes of the DAQ devices and the RT Analysis tasks, allocate and lock buffers for the subsequent DMA transfers, prepare timing devices (UTDCs) to generate clock impulses upon trigger events, and last but not least, start the RT DAQ and Analysis threads with RT priority.

During the “acquire” phase a legacy diagnostic simply waits for the acquisition in the external devices (CAMAC modules, data recorders, etc.) to finish. A RT diagnostic in contrast starts its core activity which typically is driven by the incoming data and the interrupts from finishing DMA transfers. Thus a RT thread stays active during the acquire phase and follows DMA activity. Depending on the particular implementation it may be necessary

regularly to restart DMA transfers upon interrupt. For continuously running sampling streams this is usually done in the interrupt handler without returning control to the user level. In burst mode cases it may be desirable to return control to the user's program between DMA transfers to potentially modify parameters before the next burst (e.g. Thomson scattering being clocked by the laser shots has reasonable time gaps between DMA transfers). Other supervising tasks might include watching the DMA activity filling memory and keeping track of parallel analysis tasks running immediately behind the leading edge of incoming data which is indicated by the increasing DMA pointer.

The RT criteria for DAQ will not be discussed in depth since they are obvious: No samples may be lost at the given highest sampling rate.

### *RT Analysis attached to RT DAQ*

Once started, the DMA data transfer activity is controlled by the DMA logic of the peripheral device and places little demand on the CPU resources of the host computer. Therefore, a second thread or process - depending on the details of the implementation - can make use of free computing capacity for RT data analysis. Using advanced OS features to share the DMA buffer for reading by other programs it is possible to give the analysis process access to the incoming data stream while the DMA is still active. Figure 5 illustrates this kind of RT shared memory usage. Positioning of the read pointer to address the most recent transferred section of samples in real time can be done in at least two ways:

1. Reading the DMA pointer from the external device and synchronizing the CPU with the running DMA one can expect to find the data transferred at the moment of synchronization.
2. Assuming the start time and time scale of DAQ are exactly known[6] and the DMA transfer proceeds in real time with low latency, one can calculate the buffer position where at a certain time the awaited data frame will appear in memory. Having the buffer initialized with zeros at the beginning one can poll this buffer position until the expected time value appears. This then is an indicator for both that the DMA has passed this memory location and the time one has waited for is over.

After both methods the analysis process will positively know that the next calculation step is pending and the required data is ready to be accessed. Thus the RT algorithm can proceed step-wise along the acquired data.

6 Both presumptions can be stated as satisfied, since the SIO-card containing a TDC logic allows exact determination of time on an absolute experiment-wide synchronized scale. Each sample clock pulse leading to the creation of a SIO data frame automatically stores the correct time into this frame.

*RT algorithm considerations*

Despite all the measures described above, the algorithm will always be only close to real time and the RT Analysis will always have a preliminary character with respect to what a diagnostic may deliver in a fully fledged after-shot Analysis. The numerical approach has to be pragmatic and effective, and the algorithm has to find an optimum between near real time performance, accuracy, completeness, and robustness of operation. In order to determine which RT criteria apply for Analysis following considerations should be made:

- Do the time scales of the physics subject and the analysis algorithm match?
- What is the maximum latency for results to stay relevant for subsequent algorithms?
- Is a reduced sampling rate and number of channels sufficient for meaningful results?
- How much load on the computing platform is tolerable?
- How often is a violation of the RT criteria allowed?

The answers to these questions have major impact on how much effort is required and what can be achieved.

*Fast calculation methods for raw data*

As shown in Figure 5 the shared read from the DMA buffer is the first Analysis activity. It accesses raw data in the format created by the hardware front-end. This means no mapping of samples to signals is done, no time base is extracted, no offset correction or calibration nor characteristic curve is applied. Often the 12 or 14 bit ADC samples are padded to 16 bits and sometimes these padding bits contain additional status information thus making a direct interpretation of the samples as numerical values impossible. These circumstances require an appropriate transformation of the raw data streams - potentially in multiple steps – before numerical calculations can begin.
To make transformations and subsequent calculations as fast as possible we decided to explore methods for processing of audio and multimedia data which could be appropriate for acquired plasma signals as well. For this kind of task, modern processors are equipped with special instructions allowing SIMD (single instruction multiple data) operations on packed integer arrays. With respect to our chosen SPARC processor platform this is the VIS instruction set [14]. On the Intel processor line the MMX instruction set provides similar features. However, using an assembler instruction set for probably often changing algorithms which have to be written and managed by "normal" computer users seemed not to be a good choice. Fortunately, with the mediaLib [15], a powerful and easy to use C library implementation of VIS / MMX based subroutines is available and offers an extensive collection of vector and array operations and functions directly applicable to sampled data streams.
As a first example for the usage of mediaLib an algorithm originating from MHD mode detection exploiting Mirnov probe measurements as proposed by Zohm [16] and Maraschek [17] has been implemented by Janzer et

al.[7]. This algorithm contains as numerical kernel a scalar product to project the signals from the various probe positions onto the eigenvector of a particular MHD mode. Instead of doing this time point by time point a few mediaLib calls can handle whole time intervals in a single call, thus achieving a reasonable speed-up. Another mediaLib advantage is the direct treatment of the input data streams as Q15 fractional integers (i.e. raw ADC samples, no conversion into floating point), leading to an even higher processing performance. Running this algorithm over 26 signal buffers of length 512, reducing the amount of raw data by a factor of 26 into a final result signal, takes 40 µs execution time. Here 512 samples correspond to a time interval of 256 µs in real time. To keep pace with the data acquisition the rest of the calculation has to be at least fast enough to allow further subsequent characterization of the mode signal and forwarding of the condensed results to plasma control, for example as input for MHD stabilization purposes (c.f. [6]).

### *Parallelizing Analysis with MPI*

The above Mirnov diagnostic example raises a further problem not yet mentioned: This is the number of channels (currently 112) and the production of data at a sampling rate of 2 MHz (4 MB/s/channel). When Mirnov was set up using the old HOTLink DAQ interfaces in 2001 [10] this huge data flow was addressed by dividing it into subsystems implementing only two HOTLink cards per system, restricting the number of channels to only sixteen and the data flow to 64 MB/s on each subsystem. Today seven of these Mirnov diagnostics are distributed on seven SunFire V240 computers. It is not easy to build a scalar product over data from all these seven machines as illustrated above: the application of distributed parallel computing methods is required.

An MPI [18] implementation which is well integrated with Solaris and its development tools was chosen [19] to transform the Mirnov cluster into a parallel computer. Although still in the development stage, first tests of a distributed scalar product algorithm have been made with the Soft X-Ray diagnostic cluster. The MPI algorithm to synchronize the calculation and to reduce the partial results of all ranks into one final result is a simple MPI_BCAST followed by an MPI_REDUCE. This means the cluster master broadcasts a start signal (e.g. the TDC start time) to the other nodes thus synchronizing the activity. The REDUCE subroutine implements a distributed algorithm to most efficiently communicate and condense the partial results to a final result on the master node. Figure 6 shows that the reduction of eight distributed result vectors into a single vector on the master node with MPI can be achieved in a reasonable time of 750 µs to 1750 µs depending on the vector length. This benchmark was conducted using a private TCP/IP network with a low-end 1 Gb Ethernet switch.

7 M. A. Janzer, K. Behler, A. Buhler, M. Maraschek: work in progress. C-code example illustrating mediaLib calls for raw data analysis may be obtained on E-mail request from the corresponding author of this paper.

*The RT Behaviour of Solaris*

The MPI benchmarks displayed in Figure 6 were achieved under RT priority; i.e. the involved computation processes were placed into the "real time" scheduling class, reducing the danger of these RT processes being interrupted by low priority "time sharing" activities. In fact, the test systems were not cleared of other users before the benchmark test; nonetheless, the timing results did not scatter more than 10% from the average, demonstrating how precisely the Solaris scheduler performs for RT requirements.

However, this is only a weak evidence if stronger RT conditions must be handled by Solaris as well. An important RT criterium in DAQ is, for instance, that a system always reacts on an interrupt from an I/O interface fast enough to prevent an overflow of the FIFO buffers. With the new SIO-cards this was a mandatory test upon delivery of the prototype. Since the SIO-cards only have small 32 KB buffers implemented in the FPGA and are intended to operate at a data rate of up to 64 MB/s in the cPCI version and up to 200 MB/s in the cPCIe version, only 500 µs and 160 µs respectively remain to handle device interrupts without losing data. Exploiting the interrupt generation feature of the built-in TDC the Solaris interrupt response time was measured 10 million times overnight under RT priority. Figure 7 displays the result as a histogram of the incidence of all particular interrupt response times with a resolution of 1 µs. The narrow distribution of 97.5% of all response times in the range 18-25 µs gives proof of the nearly deterministic behaviour of Solaris under RT priority. With respect to our stated requirements, Solaris seems still to be an appropriate choice.

*Summary of RT DAQ and Analysis*

Summing up the plans and findings from above one can arrive at the following conclusions about the feasibility of RT Analysis along the described concept:

- The analogue front-ends and computer interfaces (ADCs, buffers, transmission lines) work with a minimum latency of only a few sample clock intervals (Hot-Link II with SIO-board) up to a maximum latency of 256 µs (old Hot-Link interface board).
- The DMA transfer into memory is able to keep pace with DAQ. If the analysis process (running on a second CPU) is polling for valid data appearing in the DMA buffers, a small additional latency has to be taken into account for emptying the buffers along the computer's internal I/O path and to synchronize the CPU's caches with the main memory. Only a short latency is required to reschedule processes in reaction to data availability interrupts (see measurements of the Solaris IRQ response times below).
- A considerable time has to be taken into account for the numerical calculations. Here, dedicated algorithms using the DSP features of modern processors achieve a reasonable speed-up. In the given example, a group of transient signals (512 samples) could be multiplied by weight coefficients and

reduced to one signal in 40 µs.

- If data from distributed diagnostics have to be combined to a common result, the subsystems involved can be integrated into an MPI cluster environment so that high performance communication and calculation methods can be used. The combination of signal vectors of length 8192 from eight contributing systems took on average less than 2 ms.

Considering that 8192 samples at a rate of 2 MHz represent 4 ms in real time and at the end of all these RT pipelines the production of a condensed result seems to be possible in roughly half of this time, it looks promising to follow this direction to finally deliver analysed data from diagnostics to Control.

## Summary and Outlook

A flexible and extendible hardware architecture which assembles and transports time stamped data frames into computer memory with low latency has been developed. Synchronous time stamped DAQ and appropriate Solaris features allow for pace-keeping RT Analysis algorithms. High performance signal processing & parallel computing concepts are in place to build fast algorithms on diagnostic clusters. With these concepts and building blocks it seems feasible to do pace-keeping RT Analysis in time steps of 5-10 ms. This is fast enough to stay relevant for Control.

The described components of the new RT diagnostic system for ASDEX Upgrade have been set up and tested separately. The next step will bring them together with Control to achieve a plasma performance controller, e.g. for MHD stabilization, which will base its controlling algorithms on the condensed results of a huge amount of data from one or more combined diagnostics.

## References


[1] K. Behler, H. Blank, A. Buhler, R. Drube, H. Friedrich, K. Förster, K. Hallatschek, P. Heimann, F. Hertweck, J. Maier, R. Merkel, M. -G. Pacco-Düchs, G. Raupp, H. Reuter, U. Schneider-Maxon, R. Tisma, M. Zilker, Review of the ASDEX Upgrade data acquisition environment - present operation and future requirements, Fusion Engineering and Design, **43** (1999) 247-258

[2] The Storage System of ASDEX Upgrade and the Shotfile Access Library, https://www.aug.ipp.mpg.de/aug/manuals/ddww/ddww.html

[3] H. Reuter, MR-AFS: a global hierarchical file-system, Fusion Engineering and Design, **48** (2000) 199-204

[4] M. Zilker, K. Hallatschek, P. Heimann, F. Hertweck, ASDEX Upgrade Team, Multiprocessor systems for real-time data acquisition on the ASDEX Upgrade and future plasma experiments, Fusion Engineering and Design, **43** (1999) 417-423

[5] G. Raupp, K. Behler, R. Cole, K. Engelhardt, A. Lohs, K. Lüddecke, W. Treutterer, G. Neu, T. Vijverberg, D. Zasche, Th. Zehetbauer, ASDEX Upgrade Team, Real-Time Control and Data Acquisition with ASDEX Upgrade's New Time System, Fusion Engineering and Design, **81** (2005) 1747-1751

[6] W. Treutterer, K. Behler, L. Giannone, N. Hicks, A. Manini, M. Maraschek, G. Raupp, M. Reich, A.C.C. Sips, J. Stober, W. Suttrop, ASDEX Upgrade Team, Real-Time Diagnostics at ASDEX Upgrade-Integration with MHD Feedback Control, Fusion Engineering and Design, **to be published** (2007)

[7] CYV15G0402DXB Datasheet, http://www.cypress.com

[8] PCI to cPCI bus bridges, http://www.ni.com

[9] A. Lohs, K. Behler, K. Lüddecke, G. Raupp, ASDEX Upgrade Team, The ASDEX Upgrade UTDC and DIO cards - a family of PCI/cPCI devices for rea, Fusion Engineering and Design, **81** (2006) 1859-1862

[10] M. Zilker, P. Heimann, High-speed data acquisition with Solaris and Linux operating systems, Fusion Engineering and Design, **48** (2000) 193-197

[11] G. Raupp, R. Cole, K. Behler, M. Fitzek, P. Heimann, A. Lohs, K. Lüddecke, G. Neu, J. Schacht, W. Treutterer, D. Zasche, Th. Zehetbauer, M. Zilker, A "Universal Time" system for ASDEX Upgrade, Fusion Engineering and Design, **66-68** (2003) 947-951

[12] G. Raupp, K. Behler, R. Cole, K. Engelhardt, A. Lohs, K. Lüddecke, G. Neu, W. Treutterer, Th. Vijverberg, D. Zasche, Th. Zehetbauer and ASDEX Upgrade Team, Replacement strategy for ASDEX Upgrade's new control and data acquisition, Fusion Engineering and Design, **71** (2004) 41-45

[13] Sun Fire V210 and Sun Fire V240 Server Architecture,

http://www.sun.com/servers/entry/v210/arch_wp.pdf

[14] VIS Instruction Set, http://www.sun.com/processors/vis/index.html

[15] mediaLib Downloads and Documentation, http://www.sun.com/processors/vis/mlibfiles.html

[16] H. Zohm, J. Greene, L. Lao, E. Strait, Mirnov coil analysis in the DIII-D tokamak using the singuar value decomposition method, Research Report, **GA-A20886** (1992)

[17] M. Maraschek, J. C. Fuchs, K. F. Mast, V. Mertens, H. Zohm, Real-Time Determination of Total Radiated Power by Bolometric Cameras with Statistical Methods, Rev. Sci. Inst., **69** (1998) 109-115

[18] William Gropp, Ewing Lusk, Anthony Skjellum, Using MPI, 2nd Edition , MIT Press (1999)

[19] Sun HPC ClusterTools 5 Documentation, http://docs.sun.com/app/docs/coll/hpc-clustertools5

## Figure captions

Figure 1: The overall hardware architecture of a modern RT diagnostic. The fusion specific frontends, the interface I/O cards and the commercial computer platform are physically separated in three layers. This modular concept provides high flexibility, extandability and maintainability.

Figure 2: Schematic of the backplane showing the pipeline in its two main operation modes:

1. loading the tri-state buffers from the corresponding input cards, and after changing the state of the buffers on the pipeline

2.-n. stepping the 16bit samples through all n-1 shift buffers of the pipeline until the last one has arrived in the “yellow card”. The coordinated switching and clocking of the input and shift buffers is centrally handled through a few dedicated lines by this “yellow card” in the last slot of the backplane which also contains the HOTLink II serializer and sender.

Figure 3: Analogue front-end crate with ADC channels inserted “yellow cards” and SIO-card as computer interface. The SIO-card featuring an internal TDC thus allows centrally synchronized timing and time stamped data acquisition.

Figure 4: Functional blocks in a SIO-card illustrating the aggregation of data frames from time stamps and input streams

Figure 5: RT interaction between DAQ and Analysis using shared memory data exchange

Figure 6: Calculation times of the partial algorithms running on the nodes of an MPI cluster of eight computers (SunFire V240). MPI operation was a short BCAST and a REDUCE ADD operation on a short integer

array of a length from 128 to 8192 samples. Given is the averaged elapsed time measured in five successive runs under RT priority. The average deviation of the measurements is about 10%. While for the shorter arrays the scattering of the time measurements mostly is determined by the scattering of the "background" time consumption. The higher vector lengths show a clear tendency: four nodes are fast just doing the basic algorithm and sending the result vector to their next neighbours, the remaining four nodes take a little longer to do a first ADD of result vectors, the next two nodes take even longer doing a second ADD, and the master node (0) finally does the last ADD.

Figure 7: Histogram of Solaris IRQ response times accumulating 10 million measurements. The average IRQ response is faster than 20 µs. The distribution has a very thin tail with less than 1 in 1000 events above 50 µs, less than 1 in a million above 150 µs, and no event above 300 µs. It is expected that the very rare outlying response times do not sum up to a final violation of the overall real-time criterion for the DAQ and RT Analysis process. So this behaviour turns out as sufficient, if only enough "headroom" for the resulting response time fluctuations is foreseen in DAQ hardware buffers and time allocation to RT algorithms.

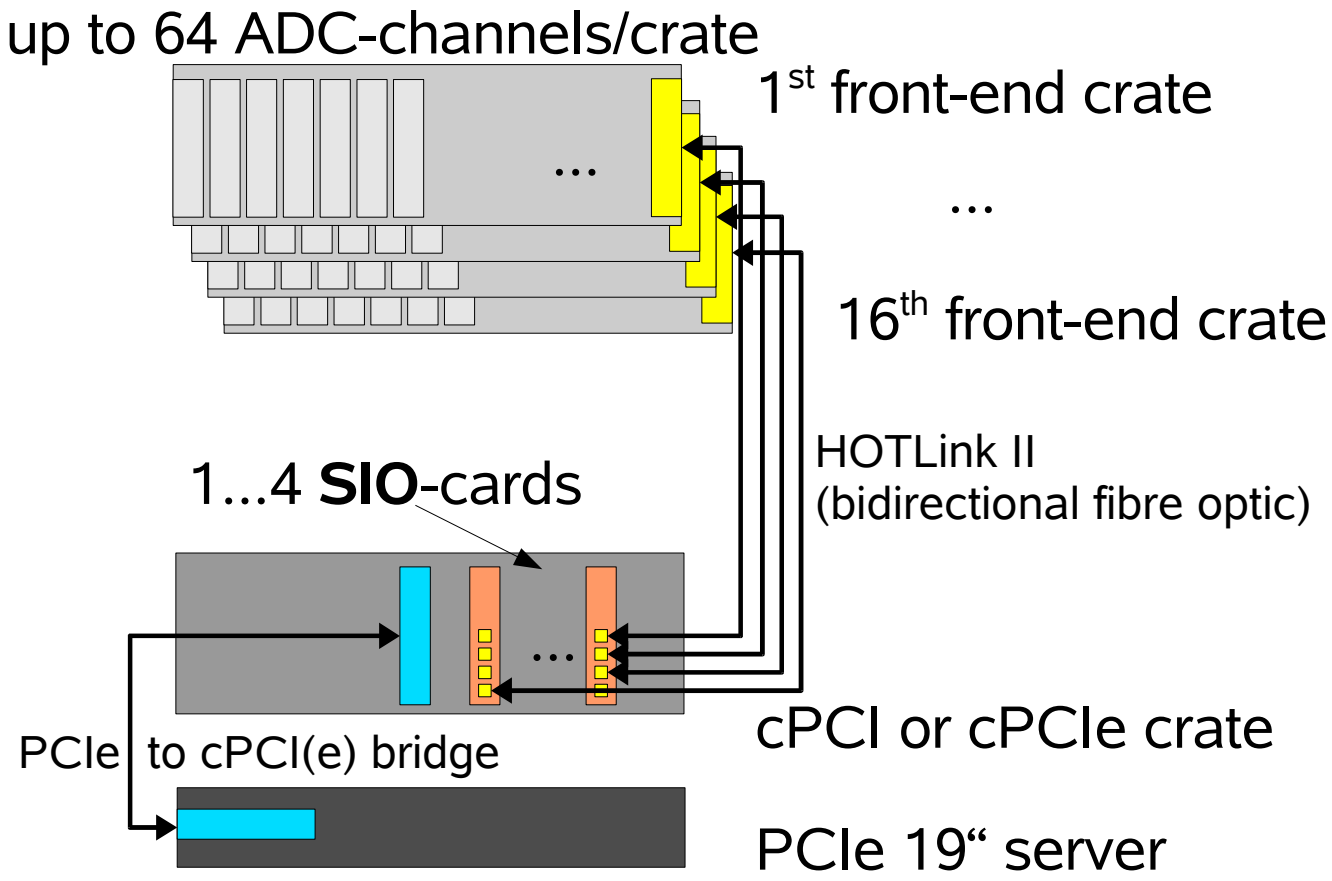
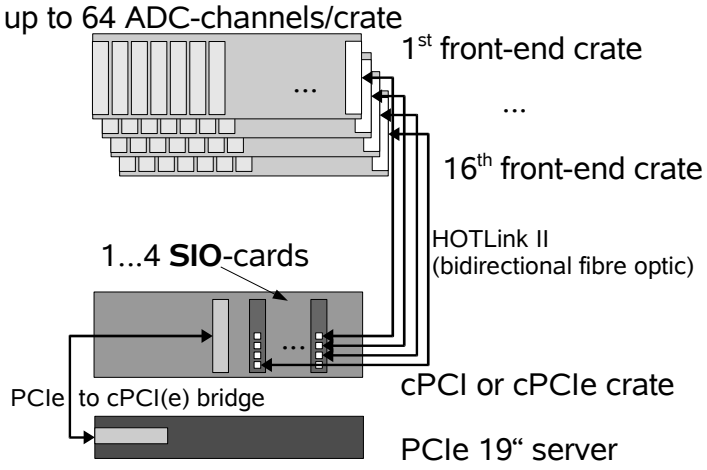



*Figure 1: The overall hardware architecture of a modern RT diagnostic. The fusion specific frontends, the interface I/O cards and the commercial computer platform are physically separated in three layers. This modular concept provides high flexibility, extandability and maintainability.*



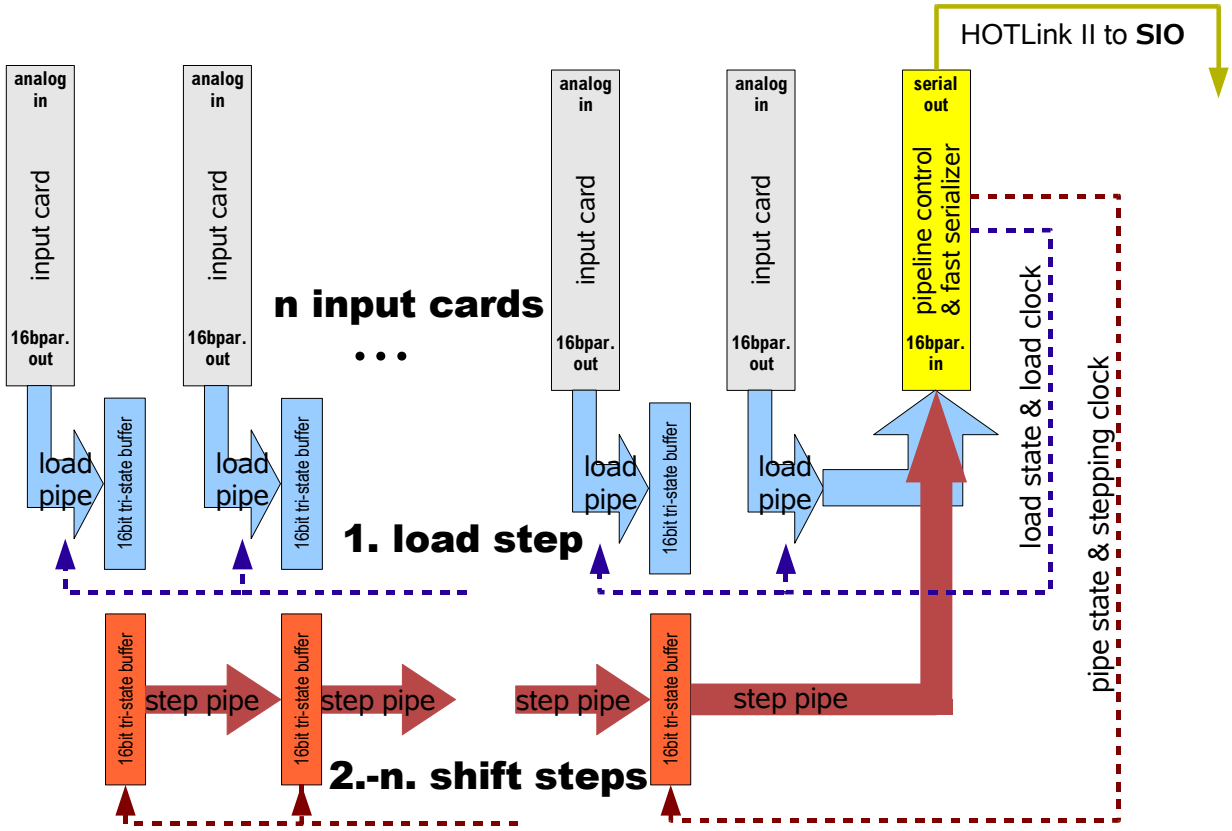



*Figure 2: Schematic of the backplane showing the pipeline in its two main operation modes:*
*1. loading the tri-state buffers from the corresponding input cards, and after changing the state of the buffers on the pipeline*
*2.-n. stepping the 16bit samples through all n-1 shift buffers of the pipeline until the last one has arrived in the "yellow card". The coordinated switching and clocking of the input and shift buffers is centrally handled through a few dedicated lines by this "yellow card" in the last slot of the backplane which also contains the HOTLink II serializer and sender.*

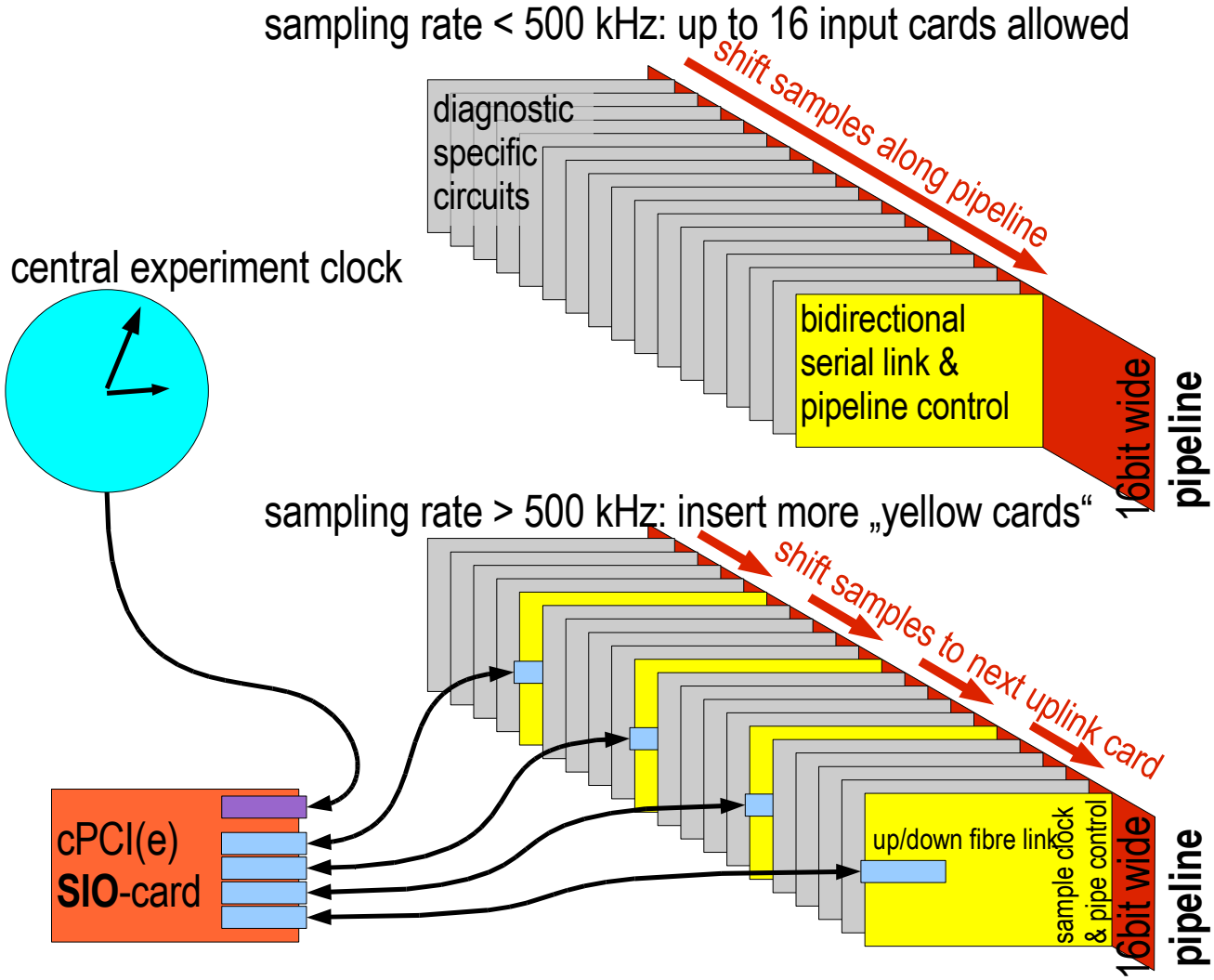



*Figure 3: Analogue front-end crate with ADC channels inserted "yellow cards" and SIO-card as computer interface. The SIO-card featuring an internal TDC thus allows centrally synchronized timing and time stamped data acquisition.*

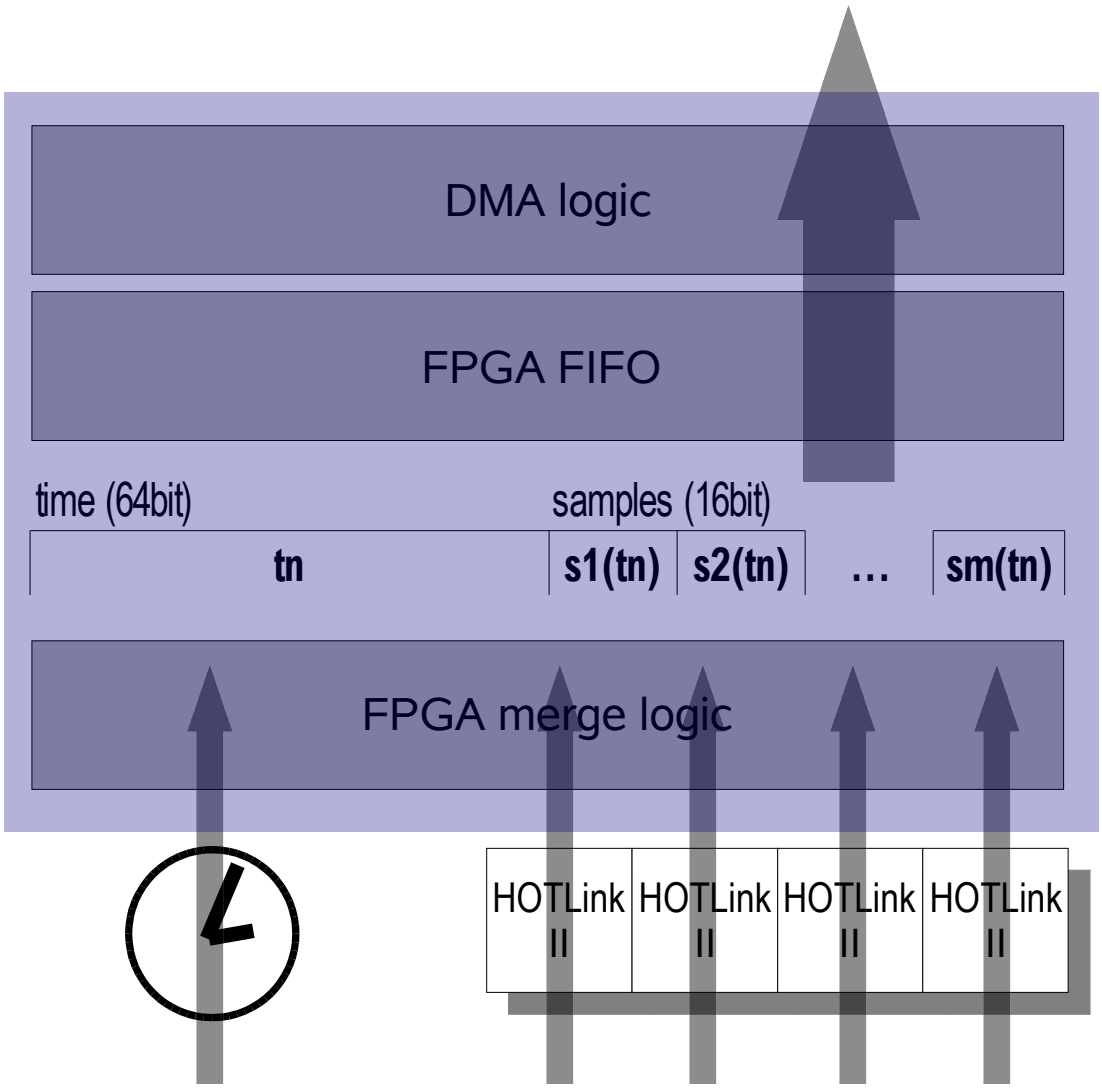



*Figure 4: Functional blocks in a SIO-card illustrating the aggregation of data frames from time stamps and input streams*

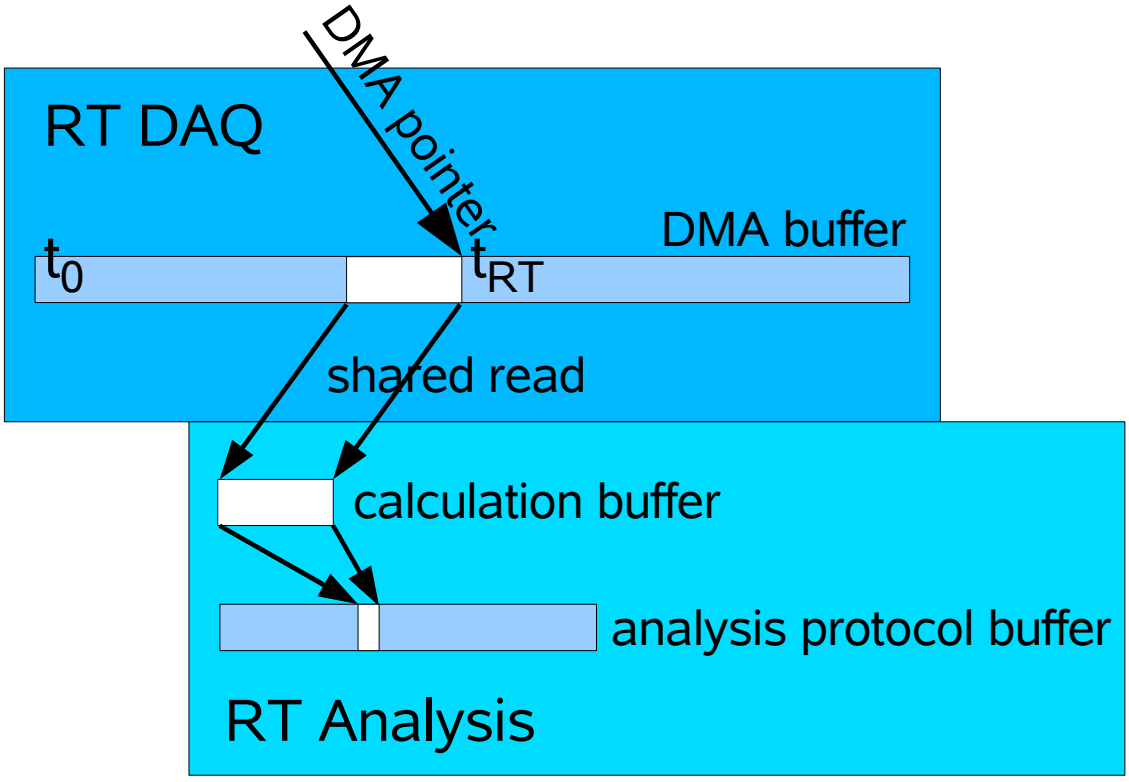



*Figure 5: RT interaction between DAQ and Analysis using shared memory data exchange*

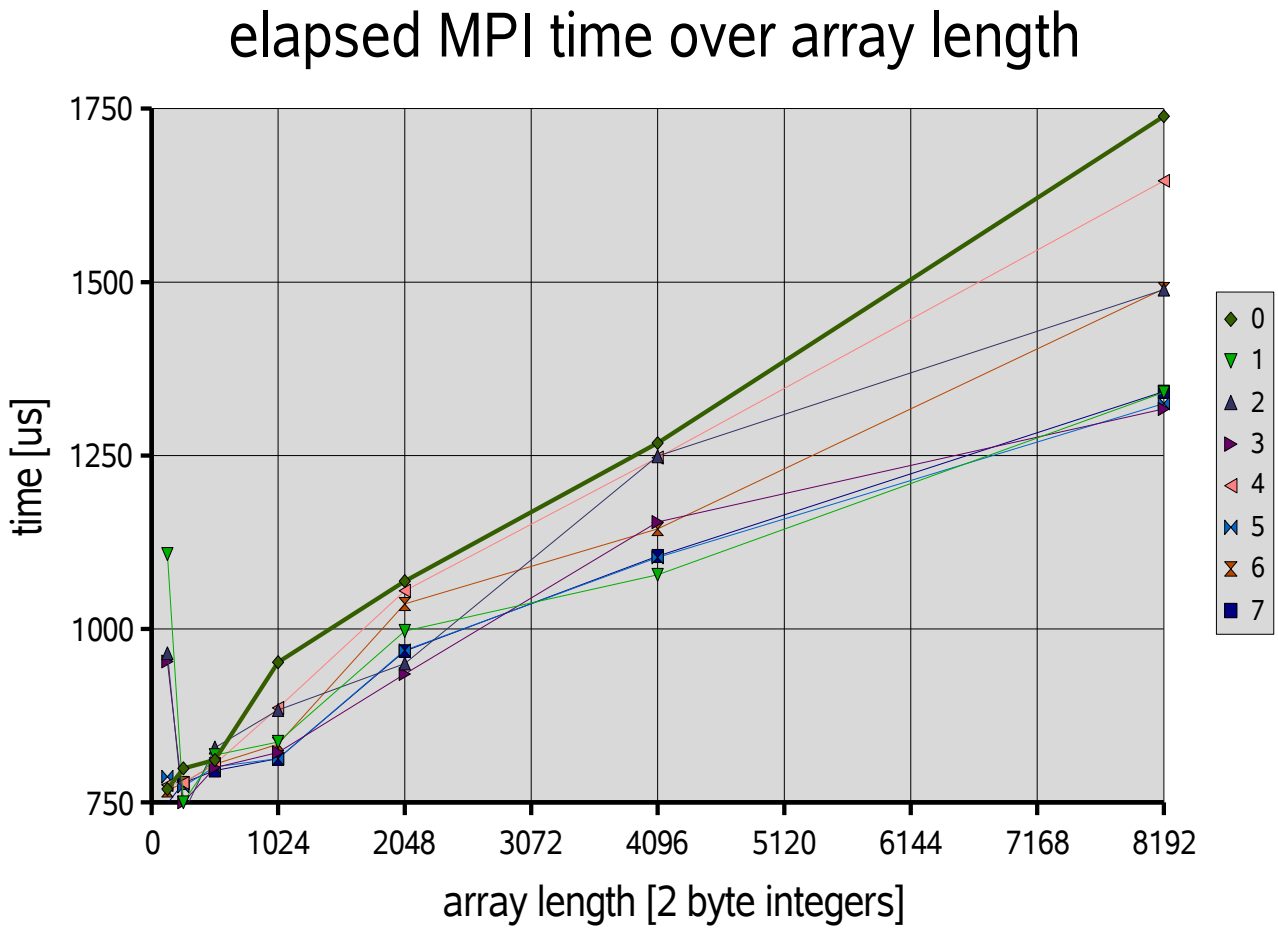



*Figure 6: Calculation times of the partial algorithms running on the eight nodes of an MPI cluster (8x SunFire V240). MPI operation was a short BCAST and a REDUCE ADD operation on a short integer array of a length from 128 to 8192 samples. Given is the averaged elapsed time measured in five successive runs under RT priority. The average deviation of the measurements is about 10%. While for the shorter arrays the scattering of the time measurements mostly is determined by the scattering of the "background" time consumption. The higher vector lengths show a clear tendency: four nodes are fast just doing the basic algorithm and sending the result vector to their next neighbours, the remaining four nodes take a little longer to do a first ADD of result vectors, the next two nodes take even longer doing a second ADD, and the master node (0) finally does the last ADD.*

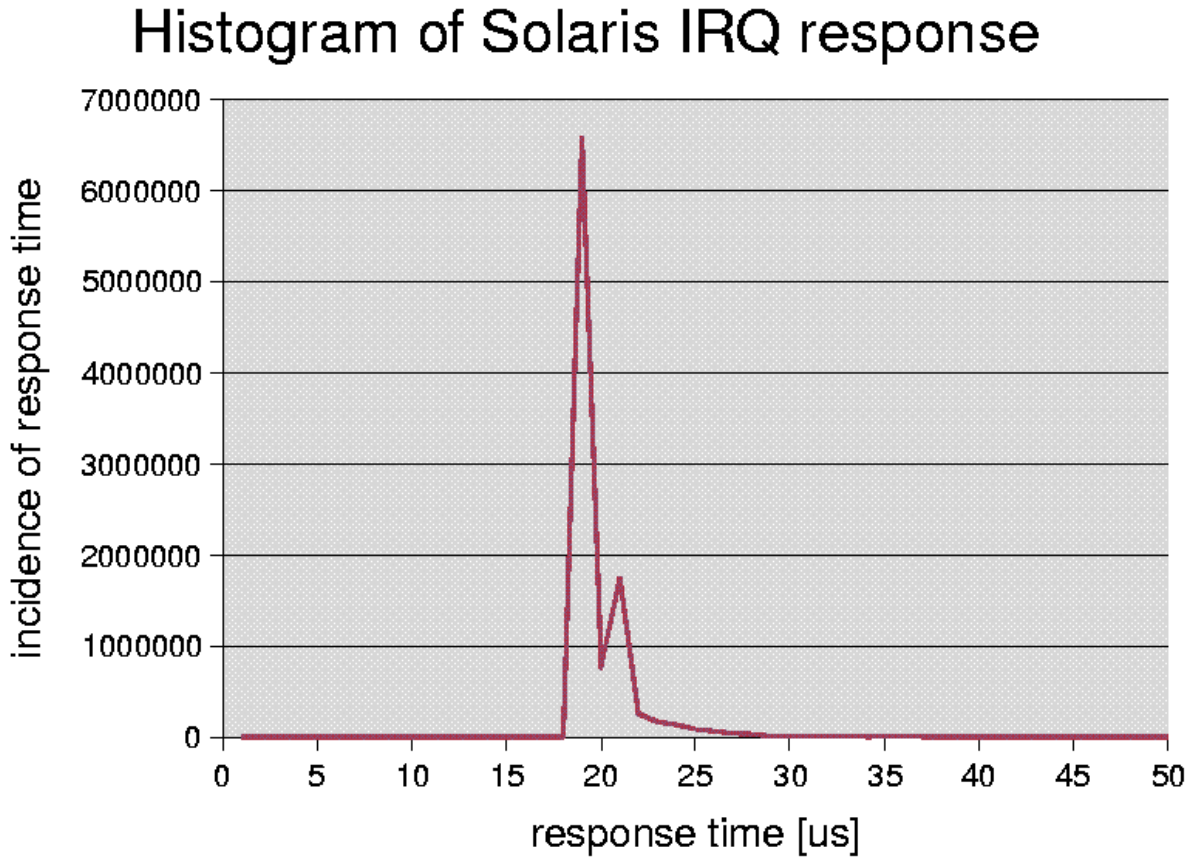



*Figure 7: Histogram of Solaris IRQ response times accumulating 10 million measurements. The average IRQ response is faster than 20 µs. The distribution has a very thin tail with less than 1 in 1000 events above 50 µs, less than 1 in a million above 150 µs, and no event above 300 µs. It is expected that the very rare outlying response times do not sum up to a final violation of the overall real-time criterion for the DAQ and RT Analysis process. So this behaviour turns out as sufficient, if only enough "headroom" for the resulting response time fluctuations is foreseen in DAQ hardware buffers and time allocation to RT algorithms.*