Data reduction in the ITMS system through a data acquisition model with self-adaptive sampling rate

M. Ruiz , JM. López , G. de Arcas , E. Barrera , R. Melendez , J. Vega

Grupo de Investigación en Instrumentación y Acústica Aplicada, Universidad Politécnica de Madrid (UPM),

Departamento de Sistemas Electrónicos y de Control, Universidad Politécnica de Madrid (UPM),

Asociación EURATOM/CIEMAT para Fusión, Madrid, Spain

Abstract

Long pulse or steady state operation of fusion experiments require data acquisition and processing systems that reduce the volume of data involved. The availability of self-adaptive sampling rate systems and the use of real-time lossless data compression techniques can help solve these problems. The former is important for continuous adaptation of sampling frequency for experimental requirements. The latter allows the maintenance of continuous digitization under limited memory conditions. This can be achieved by permanent transmission of compressed data to other systems. The compacted transfer ensures the use of minimum bandwidth. This paper presents an implementation based on intelligent test and measurement system (ITMS), a data acquisition system architecture with multiprocessing capabilities that permits it to adapt the system's sampling frequency throughout the experiment. The sampling rate can be controlled depending on the experiment's specific requirements by using an external dc voltage signal or by defining user events through software. The system takes advantage of the high processing capabilities of the ITMS platform to implement a data reduction mechanism based in lossless data compression algorithms which are themselves based in periodic deltas.

Keywords: Real-time data acquisition and processing; Adaptive sampling; Lossless compression; Advanced intelligent instrumentation

1. Introduction

One goal in data acquisition systems used in long pulse or steady state fusion diagnostics is to acquire capabilities to adapt the sampling rate depending on the experiment's time evolution. Another is to compress the acquired information in order to reduce the high amount of data to be processed, and therefore reduce the required network bandwidth and storage requirements. In order to achieve these goals, flexible data acquisition system architecture that can be easily customized to specific experiment requirements within a short development time is needed The ITMS platform is an example of such system architecture. The final goal of ITMS is to offer a technology, consisting of a set of hardware and software tools that allows the development of advanced intelligent instruments (AII) with fast data acquisition and processing capabilities that can be reconfigured through software. The system architecture and the use of fourth generation programming languages, such as LabVIEW, allow the reduction of development time substantially.

This paper presents an application of this technology to implement a data acquisition and processing system that has adaptive sampling capabilities, and can also compress the acquired data before sending it to a database server or to other processing systems through a network connection. Both features reduce the amount of data acquired by the system, while maintaining the same amount of information, allowing a reduction of the required transmission bandwidth and storage requirements of the overall system. Adaptive sampling capabilities means that the sampling rate of data acquisition cards that are used can be changed dynamically, e.g. during the experiment duration, according to specific requirements. In order to keep track of the sampling frequency used to acquire each sample, this system uses an analogue dc voltage applied to an input of each data acquisition card, which is related to the sampling rate being used. There is a direct relation between the dc voltage value of that input channel and the sampling frequency being used to acquire the rest of the channels (see Section 3). The system also includes a data reduction mechanism that is implemented based in data compression algorithms developed in LabVIEW

2. ITMS architecture

Fig. 1 shows the hardware architecture of the ITMS platform adapted to this application. The hardware consists of the following elements:

- One standard PXI chassis, with an embedded system CPU (SCPU). SCPU is used to configure data acquisition cards (DAQ), to acquire data from DAQ channels, to distribute the acquired data among system's CPUs and to process the data of desired channels.
- Several DAQ cards.
- Several peripheral CPU cards (PCPU) allocated in peripheral PXI slots (CC8-BLUES Pentium 3 from EKF System or Inova ICP-PM-4). PCPUs are used to process the data acquired from desired channels. PCPUs will increase the computing capacity of the PXI platform as compared to typical PXI systems based only in the system CPU controller

ITMS software architecture is based in three software modules that run both in the SCPU and in the PCPUs (Fig. 2): Setup&DAQ, dynamic data processing system (DDPS) and event detection and sample rate modification (EDSRM). These modules were developed using Linux kernel modules (Fedora Core 1) based on RTAI (Version 3.3), the COMEDI data acquisition driver and LabVIEW (Version 8.2).

The Setup&DAQ module deals with the data acquisition and distribution configuration and the acquisition and data distribution among processing elements of the system (PCPUs and SCPU). The user interface was developed in LabVIEW, while the data acquisition and distribution part was coded using Linux kernel modules. The part that deals with the continuous data collection in the final user applications was developed in C and integrated in LabVIEW code by using code interface nodes (CINs).

The DDPS module is in charge of running the data processing routines defined by the user. These routines were developed in LabVIEW. The results obtained by DDPS are sent to another computer using the network connection.

The EDSRM module is a client-server application that runs in each CPU (SCPU or PCPUs). The client, which usually runs in the PCPUs, detects the events generated by the DDPS module and sends them to the server module as messages through TCP/IP. The server, which always runs in the SCPU, receives the messages from the clients, or any other external systems, and updates the sampling rate accordingly. Sampling rate changes can also be caused by external systems by applying an external dc voltage signal to one of the system's inputs (external sampling rate control signal). The relation between the sampling frequency and the external dc voltage value is defined during the system configuration process. The EDSRM server also checks for changes in this voltage to detect sampling rate modification requests.

3. Implementation of the adaptive sampling rate model in the ITMS

Sampling rate in commercial data acquisition cards is controlled through an internal or external clock signal. The raw data acquired from the card does not include any information about the sampling rate that has been used to acquire it. The sampling rate information is added by the data acquisition driver when it collects the data, according to the data acquisition configuration information. Fig. 3 shows how the ITMS hardware architecture was modified to implement adaptive sampling rate. The DAQ's external clock input is used to control the acquisition process. This clock is generated using a counter of an additional DAQ card (DAQ0), whose input can be chosen between an external master clock signal connected to the clock's source input, or the internal DAQ timebase clock. In addition, one analog input channel of the DAQ card is used to read the value of the external sampling rate control signal. Finally, an analog output voltage channel of the same card is used to generate the mark voltage

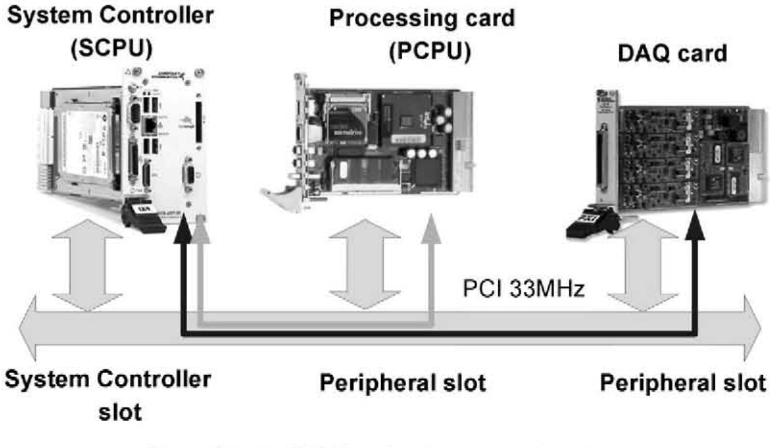

Fig. 1. Standard ITMS platform hardware architecture.

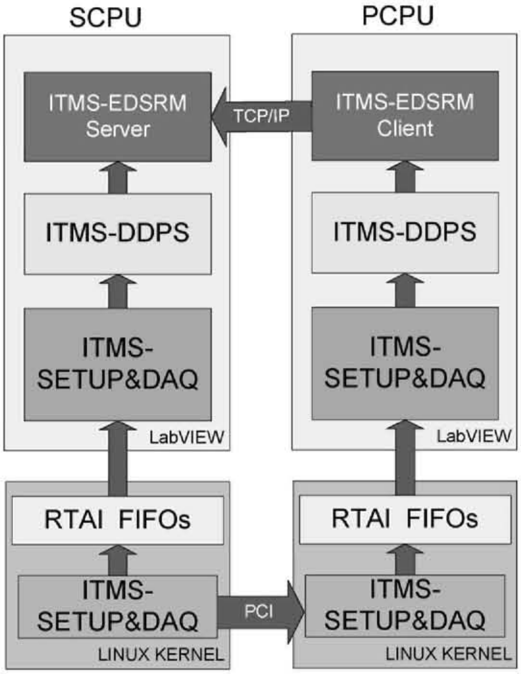

Fig. 2. Diagram of software applications, at LabVIEW and Linux kernel level, running in system CPU (SCPU) and peripheral CPUs (PCPUs).

channel. This voltage is related to the sampling rate used for a predefined set of values, and is connected to the analog input channel 0 of each DAQ card. This channel is named mark voltage channel (V_{MC}). When the sampling frequency is changed during an acquisition, the analog dc voltage generated is also updated, and therefore when new samples are acquired, the sampling rate used can be determined by looking for a change in the voltage level of the first input channel of that DAQ card. The hardware must guarantee that the changes in the sampling rate clock and the generated voltages are produced simultaneously. It must be noted that in this implementation model it is not necessary to stop and start the data acquisition driver in order to change the sampling rate.

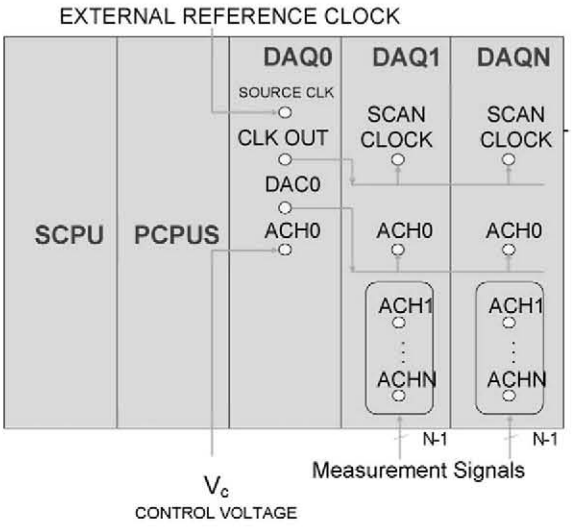

Fig. 3. Modified ITMS hardware architecture for self-adaptive sampling rate.

The Setup&DAQ module running in the SCPU analyzes the mark channel in the data blocks in order to determine if a change of the sampling rate has been produced at any time, and assigns the correct sampling information to each data block.

This mechanism ensures that any external system can change the sampling rate as needed by changing the analog dc voltage. An example of such an external system could be a specific hardware of the experimental environment used to generate the temporal profile of the sampling rate. Also, another device of the same PXI system could generate this control voltage.

Fig. 4 shows the steps in the ITMS during a change in the sampling rate for a particular case in which it is necessary to increase the sampling rate if one of the acquired signals has a significant amount of energy in the 2 kHz band (being this is a user defined event). The goal is that the system changes automatically the sampling rate being used from 100 kS/s to 200 kS/s if it detects the above mentioned event.

The system behaves as follows (see Fig. 4):

Step 1: The acquired data is collected by the Setup&DAQ module running in the SCPU. This module analyzes the mark channel information to determine if there has been any change during that block acquisition in the sampling rate. Then it sends the data to be processed to the PCPU using a frame that contains the following elements: the initial sampling rate used in the buffer; the final sampling rate used in the buffer; the index within the buffer of the sample where the sampling rate was changed, or -1 if there had been no change; and the data block size in bytes.

Step 2: The frame is received by the Setup&DAQ module running in the PCPU and passed to the DDPS.

Step 3: The frame is processed according to the desired event detection algorithm developed by the user in LabVIEW. In this case, as an example, a floating point FFT is performed to determine if the input signal has energy in the 2 kHz frequency band. As this is not the case in this step nothing else happens and the cycle repeats.

Step 4: Block N is sent to PCPU1.

Step 5: Block N is received by PCPU1.

Step 6: If the DDPS module detects energy in the 2 kHz band, it generates automatically an event and sends it to the client EDSRM module running in the PCPU using LabVIEW notifiers.

Step 7: The EDSRM client sends a message to the server EDSRM module running in the SCPU for it to change the sampling rate.

Step 8: When the server module receives this message it changes the sampling clock and the mark channel dc voltage accordingly. These tasks can only be done in the SCPU.

Step 9: The change in the sampling rate will be detected by the Setup&DAQ module running in the SCPU when a new block arrives containing a change in the value of the voltage of the mark channel.

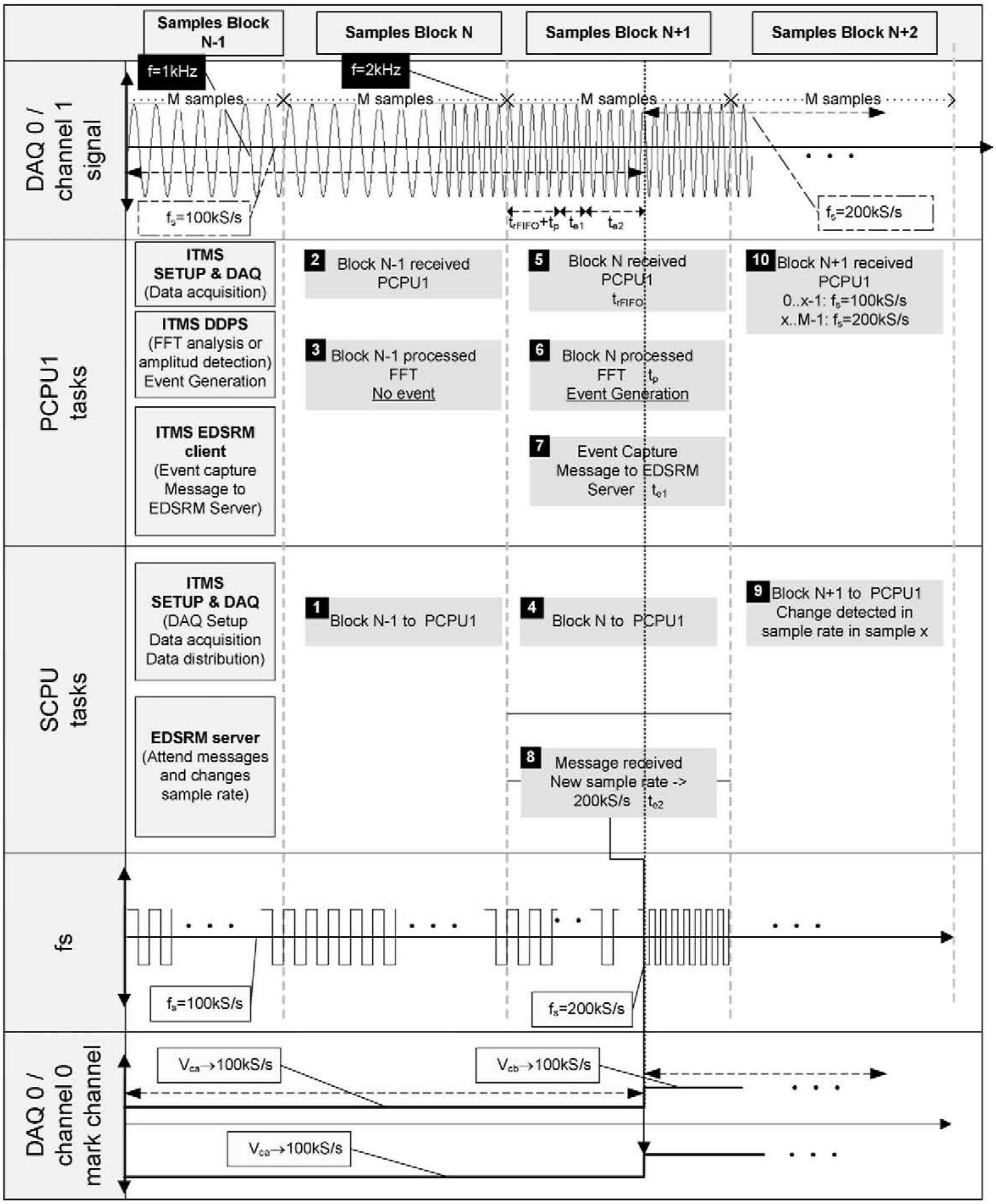
Samples Block N-1
Samples Block N
Samples Block N+1
Samples Block N+2
DAQ 0 / channel 1 signal
f=1kHz
f=2kHz
M samples
M samples
M samples
M samples
f$_s$=100kS/s
$t_{rFIFO}+t_p$
t_{e1}
t_{e2}
f$_s$=200kS/s
PCPU1 tasks
ITMS SETUP & DAQ (Data acquisition)
ITMS DDPS (FFT analysis or amplitud detection) Event Generation
ITMS EDSRM client (Event capture Message to EDSRM Server)
2 Block N-1 received PCPU1
3 Block N-1 processed FFT No event
5 Block N received PCPU1 t_{rFIFO}
6 Block N processed FFT t_p Event Generation
7 Event Capture Message to EDSRM Server t_{e1}
10 Block N+1 received PCPU1 0..x-1: f$_s$=100kS/s x..M-1: f$_s$=200kS/s
SCPU tasks
ITMS SETUP & DAQ (DAQ Setup Data acquisition Data distribution)
EDSRM server (Attend messages and changes sample rate)
1 Block N-1 to PCPU1
4 Block N to PCPU1
9 Block N+1 to PCPU1 Change detected in sample rate in sample x
8 Message received New sample rate -> 200kS/s t_{e2}
fs
f$_s$=100kS/s
f$_s$=200kS/s
DAQ 0 / channel 0 mark channel
V_{ca}→100kS/s
V_{cb}→100kS/s
V_{ca}→100kS/s

Fig. 4. Steps (marked with squares) in the operation of ITMS to adapt experiment's sampling rate when user defined events have been defined.

Experimental results show that if three input signals are acquired using the conditions of Fig. 4, a sampling rate change appears, as an average, in the sample index 800 of the next block. For the sampling rate used in the example, this means a latency time of 8 ms. This latency is measured by analyzing in which block's sample the change in the mark channel appears. This latency is due to the internal processes of the system, which mainly consist of: reading the data samples from the RTAI FIFO memory ($t_{\rm rFIFO}$), process the data block and generate the event to the EDSRM client ($t_{\rm p}$), generate and receive the event message to finally update the sampling rate and mark channel voltage ($t_{\rm e1}$). Therefore, the latency time depends on the number of samples to acquire, the event detection algorithm implemented and the client-to-server latency time of the EDSRM modules. In this example the communication protocol between the EDSRM modules was implemented using TCP/IP trough a network connection, but the next step is to implement it using the PCI bus to reduce its latency. In this example the DDPS module also sends the acquired data to a database storage server through a network connection.

One of the inconveniences of this approach is the impact of the time needed to reprogram the counter of the DAQ card used to generate the sampling rate clock. With the hardware used it was detected that while the counter is being reprogrammed its output remains at a low level, and thus no samples are acquired during that interval (more o less 500 μs) causing a small loss of data each time the sampling rate is changed.

4. Data reduction implementation in ITMS

The data reduction mechanism that was implemented in this application consists of using the lossless data compression algorithms based in periodic deltas These algorithms are applied to data blocks in the DDPS module before sending them to the network. The algorithms, developed in C, were integrated in LabVIEW and coded using CINs. It must be noted that this means the ITMS can send three different data frame types through the network depending on the data format: binary, compressed binary and floating point. In this particular application the compressed binary format frame includes information about the sampling rate, the compression method, the initial sample and the delta values.

5. Discussion

Data acquisition systems with adaptive sampling rate capabilities can be very interesting, if not crucial, for long pulse or steady state experiments. The possibility of having scalable DAQ systems with local multi-processing capabilities allows the development of highly efficient and complex data acquisition applications oriented to data reduction.

The model presented here, based on the ITMS platform, in which the sample rate is controlled by a dc voltage, allows finding solutions to complex experiment requirements.

Hardware solutions must be proposed in order to solve the problems related to the loss of information during the short time needed to update the sampling rate clock.

The solution proposed is the first step in the definition of a model for the development of AII. These can adapt their behaviour to the experiment temporal evolution.

References

E. Barrera, M. Ruiz, S. López, D. Machon, J. Vega, PXI-based architecture for real time data acquisition and distributed dynamic data processing, IEEE TNS 53 (3) (2006) 923–926.

M. Ruiz, E. Barrera, S. López, D. Machón, J. Vega, E. Sánchez, Distributed real time data processing architecture for the TJ-II data acquisition system, Rev. Sci. Instrum. 75 (10) (2004) 4261–4264.

J. Vega, M. Ruiz, E. Sánchez, A. Pereira, A. Portas, E. Barrera, Real-time lossless data compression techniques for long-pulse operation, Fusion Eng. Des. 82 (2007) 1301–1307.