Available online at www.sciencedirect.com
SCIENCE DIRECT®
Fusion Engineering and Design 81 (2006) 1835–1840
Fusion Engineering and Design
www.elsevier.com/locate/fusengdes

Integration of MDSplus in real-time systems

A. Luchetta, G. Manduchi *, C. Taliercio

Consorzio RFX, Euratom-ENEA Association, corso Stati Uniti 4, 35127 Padova, Italy

Available online 6 May 2006

Abstract

RFX-mod makes extensive usage of real-time systems for feedback control and uses MDSplus to interface them to the main Data Acquisition system. For this purpose, the core of MDSplus has been ported to VxWorks, the operating system used for real-time control in RFX. Using this approach, it is possible to integrate real-time systems, but MDSplus is used only for non-real-time tasks, i.e. those tasks which are executed before and after the pulse and whose performance does not affect the system time constraints.

More extensive use of MDSplus in real-time systems is foreseen, and a real-time layer for MDSplus is under development, which will provide access to memory-mapped pulse files, shared by the tasks running on the same CPU. Real-time communication will also be integrated in the MDSplus core to provide support for distributed memory-mapped pulse files.

Keywords: Real-time systems; Data acquisition; Distributed systems

1. Introduction

The MDSplus data acquisition package is currently used in several nuclear fusion devices, such as RFX [1], CMOD [2], NSTX [3]. In these experiments MDSplus is used pervasively, i.e. for data acquisition, data access and presentation, graphical user interface and sequence coordination. Several other experiments employ custom data acquisition systems, but use MDSplus for data presentation, i.e. to provide a MDSplus interface to stored experiment data so that data can be accessed by data analysis programs using a standard data access library.

The architecture of MDSplus is based on the concept of shot (or pulse) and therefore MDSplus handles pulsed experiments in which plasma discharge is achieved for a short period, during which the data acquisition system typically does not work, data being acquired by transient recorders which fill their local memory during the pulse. MDSplus comes in action before the shot, to download configuration parameters, and after the shot, to upload data stored in transient recorders and other devices and to store data into the pulse files. This organization fulfils the requirements for most current fusion devices, but cannot be used in the next generation of fusion devices, for which the plasma discharge is quasi-continuous. For such devices, it is not acceptable to restrict the usage of the data acquisition system to before and after the shot, and the system must carry out data acquisition and control

* Corresponding author. Tel.: +39 049 8295039; fax: +30 049 8700718.
E-mail address: gabriele.manduchi@igi.cnr.it (G. Manduchi).

0920-3796/$ – see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.fusengdes.2006.04.035

during the pulse itself. In this case real-time constraints must be considered, while off-pulse computation is typically not subject to such constraints. Real-time constraints do not represent however the only issues to be considered for long lasting discharges, but several other features, not currently supported by MDSplus, need to be implemented in this context, such as the possibility of tagging data with time and the possibility of incremental data update.

In this paper we present the ideas we developed in using MDSplus to integrate the real-time control components of RFX-mod into the MDSplus framework. It is in fact our belief that the requirements for MDSplus in real-time applications are very similar to those for long lasting discharge experiments.

2. Current implementation

RFX-mod started operation in December 2004 after a 5 year shutdown during which major modifications have been done to the machine assembly. The most important upgrade has been the substitution of the previous thick shell, with 400 ms time constant for radial field penetration, with a thin copper shell whose time constant is 50 ms. This is shorter than the expected duration of the discharge in RFX-mod, so active control is required during the pulse. A thinner shell allows a more rapid penetration by externally applied electromagnetic fields, providing a variety of possible control scenarios, not possible in the previous RFX assembly.

The architecture of the real-time control system of RFX-mod is described in detail in [4], and we shall provide here only a brief overview. The system is composed of seven VME control units communicating in real time via Ethernet and UDP in an isolated 100 Mbit/s network. Each control unit hosts a MVME5100 PPC processor board and may host Analog to Digital (ADC) and Digital to Analog (DAC) devices. During control, data are read from the ADC modules and two kinds of elaboration are defined in the architecture: a Pre-Elaboration procedure which takes locally acquired data and produces a data buffer to be sent via UDP to other control units, and an Elaboration procedure taking data both locally acquired and received over the network to produce the outputs of the unit, i.e. data which is then sent to the DAC devices to produce the reference signal for the actuators.

The former computation is used to derive plasma parameters from raw data, such as the position of the plasma column and the spatial Fourier components of the radial field along toroidal and poloidal positions. The latter computation carries out the control algorithms, normally based on the results of the pre-elaboration routine in some other control unit.

The kernel of the real-time control system handles a real-time database in memory, containing the signals acquired and the produced references, and does not make use of MDSplus at all. Before the shot, the system receives a set of configuration parameters, originally stored in the MDSplus experiment model for RFX-mod, and provides the off-line readout of the signals stored in memory during the pulse, representing the evolution of the inputs and of the outputs together with the intermediate results of the control computation. These signals are read and stored in the MDSplus pulse file after the shot. The set of parameters for each control unit forms a MDSplus device, i.e. a subtree of the experiment model with an associated graphical user interface. The INIT method for such device, called by the MDSplus framework before the pulse, downloads configuration parameters, and the STORE method, called after the pulse, uploads the waveforms stored by the control unit in local memory. For upload and download, MDSplus is still used, and several components of MDSplus have been therefore ported to VxWorks, the operating system used in the control units. In particular, a mdsip server runs on each control unit (mdsip is the TCP/IP-based protocol defined in MDSplus for network data exchange) so that it is possible to evaluate expressions specified in TDI, the scripting language of MDSplus. In particular, a TDI expression may specify the activation of a user routine belonging to a specified library. Using this approach, the integration of the control units into the MDSplus framework required only writing a generic parameter setting function which takes as arguments the name of the parameter and its content, and another function for reading a waveform stored in memory. No code handling TCP/IP communication has been required, since data transport is already handled by the mdsip layer.