THE ELETTRA CONTROL AND DATA ACQUISITION SYSTEM

P. Michelini

Sincrotrone Trieste, Padriciano 99, 34012 Trieste, Italy

Abstract

Sincrotrone Trieste has been founded to design and construct a radiation source from ultraviolet to soft X-ray (ELETTRA). It will be an insertion device dominated facility with a large number of straight sections for wiggler and undulators, with high flux of radiation from wiggler and high brilliance from undulators in the photon energy range from 10 eV to 1-2 KeV. These design goals will be achieved by a low emittance storage ring with beam energies between 1.5 and 2 GeV.

The ELETTRA control system relies on a fully distributed architecture, hierarchically divided in three layers of computers and two levels of networks providing the communication facilities between the adjacent layers. The first computer layer consists of different machines whose main tasks are in terms of graphical user interface and physics simulations. On the other hand the low level network and the two lowest computer layers are those involved in the execution of the real-time tasks which perform data acquisition and closed-loop controls.

This paper will describe the software architecture and the communication facilities of the data acquisition system outlining the extensive use of standards for both hardware and software.

Overview

The control and data acquisition subsystem, part of the more general Elettra control system [1], is made of two layers of computers, the so called Local Process Computers (LPC) and the Equipment Interface Units (EIU). Between these two layers there is an exchanged of data and information via the Low Level Network (LLN) which interconnects one LPC to up to thirty EIUs (fig. 1).

The LPC layer acts as a sort of gateway between the different branches of its connected EIUs and the Control Room computers. Its main tasks are to provide a standard access to the controlled equipment, supervise the EIU operations and the alarm managment of the front-end electronics.

The EIU layer implements the physical interface to the controlled devices, and is in charge of performing data acquisition, closed loop controls and monitoring of the hardware equipments.

The relationship between LPCs and EIUs is strictly hierarchical, in fact all the communications taking place between the EIUs and the LPCs are in direct consequence of an LPC's request. The only exception is in case of abnormal events such as faults or alarms when the EIU has to be able to send an alarm message on its own initiative to its LPC.

This last feature has greatly influenced the choice of the LLN and the adopted protocol where a simple serial time multiplexed command/response communication, called Multidrop Bus, is being subsequently used as data acquisition network.

Hardware Configuration

The LPC and EIU computers are microprocessor assemblies based on the VMEbus [2] and the Motorola 680x0 microprocessor family (x=2,3 for LPCs, x=0 for EIUs). Moreover they are equipped with the same Microware OS-9 [3] real time operating system. It is a multi-user, multi-tasking UNIX-like [4] operating system with a versatile re-entrant, position independent memory module design.

The multidrop bus used for the Elettra LLN is based on MIL-STD-1553-B standard [5], originally developed for military aircrafts. It is a master/slave system where one master, called Bus Controller (BC), continuosly polls its slaves on the bus (Remote Terminal os RT). The data transmission rate reaches 1 Mbps for a distance less than 300 meters whereas it decreases to 125 kbps for a length up to 2 km. For a data bus longer than 2 km, electronic repeaters can be used.

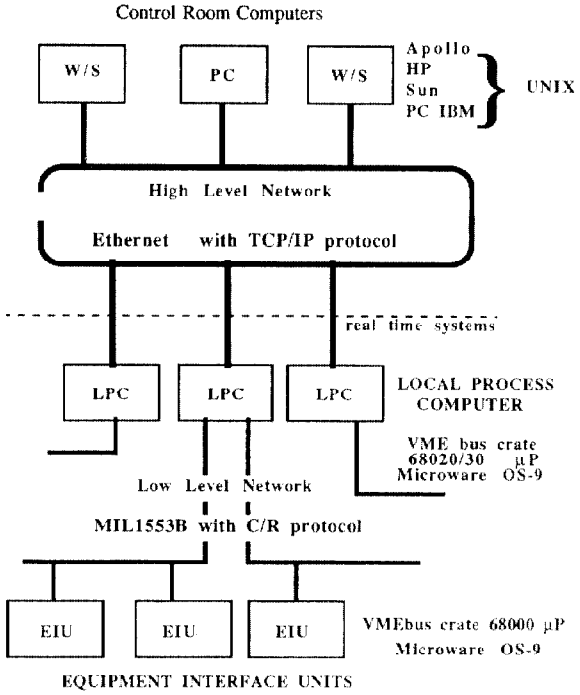

Fig. 1 ELETTRA control and data acquisition system. The dashed line defines the real time environment.

Software Design

Given the hardware scenario we can now deal with the software architecture carrying out the desired features,widely described in [1],[6],[7],[8].

The advantage of using the same operating system for the two computer layers ensures a good uniformity, a sort of simmetry in the software design which results in drastic reduction of the application and system software development time. On the other hand, a system with front-end computers equipped with a complete operating system could be oversized for some applications at present, but this choice will allow an easy system riconfiguration and the possibility to face future developments.

Although LPC and EIU main tasks will differ from one another, the control and data acquisition software can be basically divided in the following families:

communication software
equipment access software
utility programs
application or user programs;
alarm analysis and surveillance programs.

The communication software allows the message exchanges between the LPC and EIU layer. Since the relationship between these two computer layers is hierachical with one master (LPC) and various slaves (EIUs), a simple command-response protocol is mainly used. All communication mechanism and implementation of the I/O hardware interface must be hidden to the users.

The equipment access software deals with the translation between the physical address of an I/O point and its associated logic name [9]. In such a way the users will not have to worry about hardware detail referring to every piece of device in a symbolic fashion.

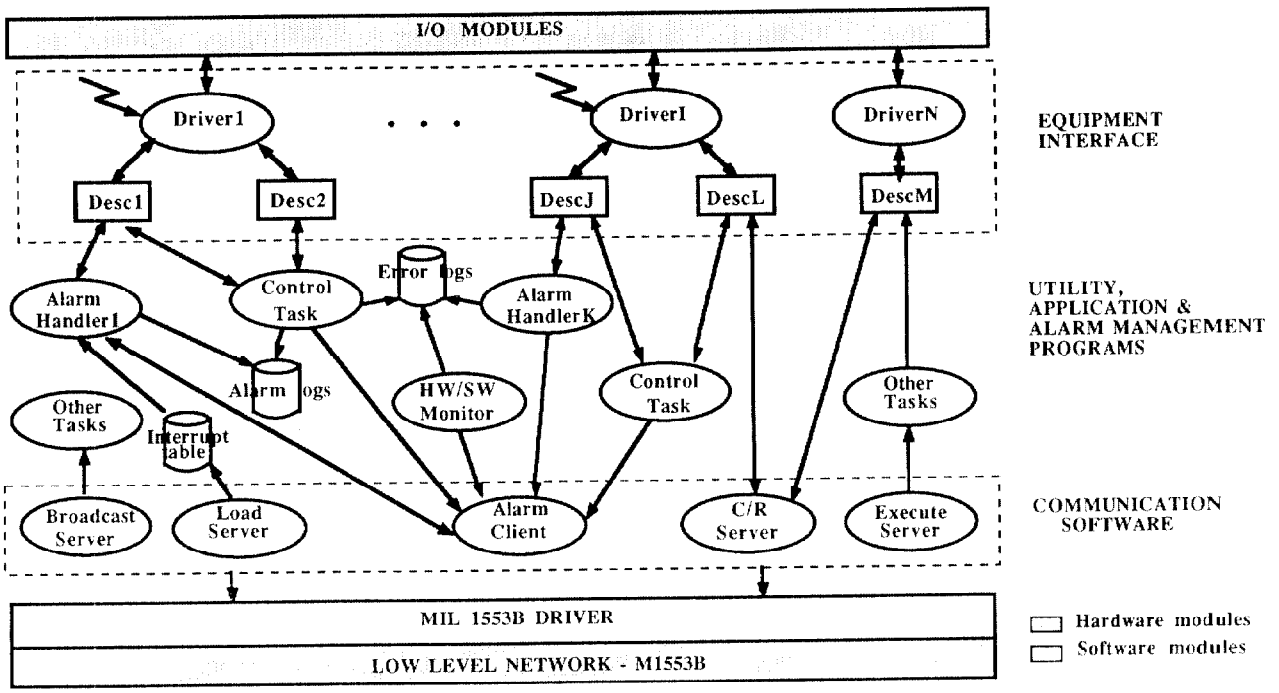

Fig. 2 EIU software architecture; the symbols close to the first two drivers represent an interrupt signal.

The third family is composed of a set of programs which can be considered as a natural appendix to the native operating system.

Application and user programs actually perform data acquisition, monitoring and all operations related to the synchrotron radiation facility.

The last program family has a fundamental role in the data acquisition system. First of all it has to be able to detect any error, fault condition or alarm message coming from the front-end electronics. Then, after a preliminary analysis of this data and the eventual request for test or survey program execution, it has to transmit the incoming alarm to the central Alarm Server in the control room.

We shall now analyze more in detail the implementation of this software exploiting, as best as possible, the OS-9 operating system characteristics: then, using its terminology we shall point out the EIU software architecture outlining the main differences with the LPC design.

EIU Software Architecture

Communication Software

Two software layers are in charge of managing the communication going through the LLN: the MIL 1553B device driver, originally developed at CERN [10] for the LEP control, and the MIL 1553B interface.

The MIL 1553B device driver supplies a set of low level routines providing a standard interface to the media while hiding physical packet management and transmission mechanism. It is responsible for queueing the incoming requests and delivering them to the right processes. The EIU's and LPC's low level network interfaces necessarily differ due to the choice of a master slave protocol which has to accomplish completely different tasks.

The MIL 1553B interface is built on top of this driver; it provides a simple, reliable and robust protocol implementing a set of communication services between the LPC and EIU layer. At the EIU level this interface basically consists of the following processes:

Command/Response server
Execute server
Load server
Broadcast server
Alarm client.

All the servers generally wait for a service request, execute it and finally send a reply message to the calling process; the process taking the initiative in requesting the service is usually called client process.Therefore every server process at EIU level will have its corresponding client process at LPC level and viceversa.

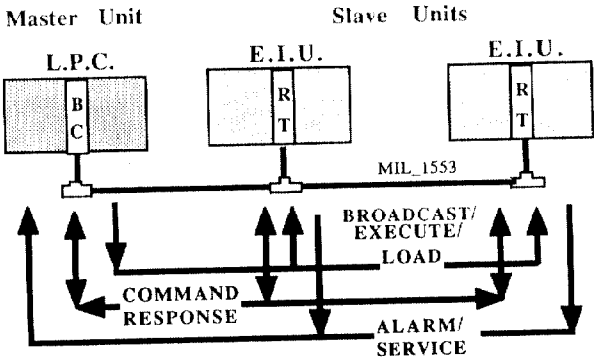

Fig. 3 Direction of the main data flow.

Command/response server provides a direct access to the equipments allowing the users to read/write data in a symbolic way from/to any I/O point.

Execute server allows the remote execution of an OS-9 command (load, kill, execute processes etc.).

Load server transfers files through the network.

Broadcast server allows the EIU to receive a broadcast message coming from its LPC.

The Alarm process is the only client process at the EIU layer: it must supply a uniform interface to the alarm forwarding system and ensure a much shorter response time in delivering the message than the other processes. At present a higher execution priority fulfils our requirements very well nevertheless, in the future, a dedicated and much simpler alarm protocol will be implemented.

Equipment Access Software

This software has to provide a standard interface to the different machine components dealing with format conversion, scale factors, settings and I/O logical names-hardware addresses translation. This information must be available at any I/O access and its management has a fundamental role in terms of global time response.

The first solution could be to create a local data base containing the logical names and a name server to manage it. In such a way all the applications interfacing the front-end electronics via the device drivers should query these tables to obtain all the needed information.

The second solution exploits some of the characteristics of the OS9 operating system and fits perfectly into its unified I/O model [3]. For any device driver a set of device descriptors are created, one for every I/O point; the names of these not executable modules match the I/O logical names and will contain all the information on the physical point. This tecnique, while preserving the use of a database, allows the access to the equipments by logical name down to the lowest possible level and it assures a good ease to reconfigure the interface modules. Although this choice increases the complexity of the device drivers on the other hand it has the great advantage to increase the system response time by reducing the number of software layers to access a single I/O point.

The device descriptor modules can easily be built by a configuration program at the startup time and then downloaded from the central database of the control room: therefore the reconfiguration of all control system parameters can occurr by simply modifying the central equipment database.

Utility Programs

They are identified as "Other Tasks" and represent all those processes, usually at a low priority, dealing with offline tests, benchmarks, calibrations, etc.

Application or User Programs

The functions of these tasks are essentially the closed-loop control and the monitoring of the I/O points so that we generally call them control tasks. These processes have different priorities according to the kind of equipment they have to check and they can deliver an alarm message to the upper layer on an error or a fault condition. The control tasks receive the closed loop parameters when they start and after that they can receive data only from the equipment interface: therefore any parameter reconfiguration implies to stop the process and restart it with the new values.

Alarm Analysis and Surveillance Program

An alarm condition is usually carried out in two ways: during a closed loop control when some parameter exceeds its normal range or due to an interrupt signal from the front end electronics. While in the first case the periodically running control tasks reveal the error condition and generate an alarm message, in the second case the control system has to detect and manage an asynchrounous signal. For this purpose we created a so called "alarm handlers" for every device driver that accepts the equipment interrupts.

At the moment these processes only deliver the alarm messages to the upper layer where they are analyzed and the correct sequence of operations is undertaken. In the future we may decide to give an autonomous action to these tasks or to associate them to special applications.

Acknowledgements

The author would like to thank R. Mazzurco and M. Mignacco for the stimulating discussions and their contribution to the definition of the system design.

References

[1] D. Bulfone, M. Mignacco, "The Conceptual Design of the ELETTRA Control System", Sincrotrone Trieste, ST/M-89/2, February 1989.

[2] VMEbus Specification, IEEE 1014-87 oc IEC 821.

[3] Microware Systems Corporation, "OS-9 Operating System Technical Manual".

[4] UNIX SYSTEM V by AT&T

[5] MIL-STD-1553B, 21 September 1978, U.S. Department of Defence.

[6] M.C. Crowley-Milling, "Layered Software for Control System", Sincrotrone Trieste, ST/M-88/6, April 1988.

[7] M.C. Crowley-Milling, "Preliminary List of Servers and Applications Programs for ELETTRA", Sincrotrone Trieste, ST/M-TN-89/13, May 1989.

[8] M. Mignacco, "Remote Procedure Call in the ELETTRA Control System", Sincrotrone Trieste, ST/M-89/4, March 89.

[9] M.C. Crowley-Milling, "Naming Conventions for ELETTRA Components", Sincrotrone Trieste, ST/M-TN-88/13, September 1988.

[10] M.C. Crowley-Milling, "The LEP/SPS Access to Equipment", LEP control Note 54 Rev.