# FIRST APPROACH TO THE CONTROL SYSTEM FOR THE LSB[*]

J. A. Perlas, J. Campmany
Laboratori Sincrotró de Barcelona - IFAE
Edifici Cn, Universitat Autònoma de Barcelona, E-08193 Bellaterra, Catalonia (Spain)

## *Abstract*

A new synchrotron light source is planned to be built in Barcelona, formed by a storage ring, a booster synchrotron and a pre-injector. Here we present the main trends of the control system for this machine (LSB). It is based on the so-called “standard model” of control systems for accelerators. In LSB, the middle-level local controllers will be distributed for functional as well as geographical reasons. We also propose to fully exploit the growing up of commercial standard products both in hardware and in software. VME/VXI products will be considered in the equipment and local controller levels. Software standardization, EPICS and Object-Oriented technologies will also be taken into account in the design of the whole control system software, in order to fulfill the manpower and budget requirements.

## 1 INTRODUCTION

In the actual stage of conception of accelerator control systems some trends are common to many machines, giving rise to the so-called “standard model” (SM) [1]. Its main features are:

- Distributed system
- Three-level hardware architecture based on three levels of hierarchical processing: high level processors (HLPs), with global control functions, medium level processors (MLPs), with functional control features in real time, and low level processors (LLPs), controlling all the I/O to the equipment
- Three networks/buses types communicating the levels
- Automatic alarm management
- Layered software.

Nevertheless, this SM has some non-specified items, specially those related with the implementation of the above mentioned hardware and software. In this paper we will underline some open aspects of the SM implementation, giving the present existing alternatives and selecting those that fulfill the LSB requirements.

## 2 OPEN ITEMS WITHIN THE “STANDARD MODEL”

### *2.1 Hardware implementation*

The MLPs are widely implemented using VME / VXI standards. LLPs are mainly VME, but other options like CAMAC or Fastbus are still used in certain designs. UNIX workstations or PCs are used as HLPs.

With respect to the buses, the LAN must be standard in the sense that must be able to be connected to equipment coming from several vendors. It must support an intense flux of information, but not necessarily in real time. This leads to three options: Ethernet, FDDI, and the recently developed ATM. The bus communicating the MLPs with LLPs is usually the MIL-1553B bus although other options are also used (PDV-Bus...). Finally, the bus communicating the LLPs and the machine equipment is not standardized. Several options are used: RS232, RS485, GPIB, direct I/O, Fieldbus, Allen-Bradley 1771, Bitbus...

Another open issue related to hardware is the failure-safe feature: some functions of the accelerator must have double control lines and failure-safe configurations, to increase their robustness.

### *2.2 Software implementation*

An open item in this chapter is the real-time versus shared time processing at each level: while it is clear that a real-time environment is not necessary in the high level, it is a must in the other two lower levels. The operating system (OS) in each level must be chosen according to this fact. In general, UNIX is the high level OS and VxWorks, LynxOS or OS-9 are used in the other levels.

EPICS [2] is usually used in the software design within the “standard model” frame. Nevertheless, the benefits of extending EPICS using object-oriented (OO) techniques is still under discussion [3].

Concerning the alarm management, the treatment of alarm signals must have priority to arrive to superior processors. But the reactions in case of alarms must be done automatically in real time by MLPs or LLPs. The operators (HLPs) must be informed about them and the status of the machine once the lower layers have reacted.

Another open item is the implementation of an Expert System to handle the routinely accelerator operations and the most frequent procedures for error recovery. Artificial Intelligence methods are proved to be very useful in this field. Although we are not considering to build an Expert System in the first implementation of the control system, we think that future upgrades will have to take it into account.

[*] Work supported by CIRIT and CICYT.

# 3 DISCUSSION ON THE IMPLEMENTATION

## 3.1 Hardware architecture

At the high level, the choice between PCs or workstations is the biggest concern. Windows NT PCs have some advantages over UNIX workstations, the cost being the main one (PCs are half the price of workstations). But there are also clear advantages in software development tools, like the existence of robust compilers, graphic tools, inexpensive commercial software and, of course, user-friendly environments. In addition, network tools developed for Windows NT are good enough for accelerator control systems. However, up to now, the main software applications related to accelerators are written for UNIX workstations (e.g. EPICS). This is an important constraint for small groups developing an accelerator control system.

At medium and low levels, the best election is VME/VXI, because of the real-time requirements and the existence of widely available commercial products. However, there is the open issue of a VXI versus VME implementation: up to where VME is feasible? In order to control instruments by MLPs, it seems that VXI is better than a VME + GPIB combination or something equivalent. In addition, there are some disadvantages concerning VME: EM noise, only register-based communications, and functions not implemented (e.g. signal generators). On the other hand, VME is cheaper and more widely used. A solution can be the use of VXI in the medium level, to which the instruments will be attached, whereas the low level is only made of VME crates, although this alternative is more expensive than having VME in both levels.

About the LAN selection several possibilities exist: we consider Ethernet, FDDI or ATM.

The main advantages of Ethernet are its low price and that commercial VME interfaces are widely available. Its disadvantage is that it can saturate, depending on the number of MLPs and network traffic.

Concerning FDDI, the main advantage is that it is ten times faster than Ethernet. Commercial VME interfaces are also available. Its disadvantage is that it is more expensive than Ethernet.

Among the advantages of using ATM we see that it has been designed for bursty communications[1]. The interface with VME is also available. Its disadvantages are that it is quite new (which implies inexperience and lack of commercial products) and the price is high.

At the equipment level, is it possible to establish any standardization criteria with respect to the bus? At present it doesn't seem to be the case. However, the development of the Fieldbus, specially in the industrial applications, makes it very attractive as a good candidate for the LSB low level bus implementation. Nevertheless, the use of this as the unique bus seems very difficult and we will have to remain open in this subject.

## 3.2 Software architecture

It is clear that the hardware architecture imposes a three-level distributed software. In addition, it has many benefits to design each level using multi-layered software. A possible solution to these requirements is the use of EPICS. It has many advantages, such as the existence of a consolidated support via its working collaboration and that it is the standard de-facto in the accelerator control system world. But it also has some disadvantages: it has little tradition in Europe, and might constrain some hardware choices (e.g. UNIX workstations at the high level).

Another field that we want to explore is the Object-Oriented technologies. They give a maximum of modularity and reusability of the code, being therefore very easy to customize and upgrade, and thus provide a significant reduction in programming time.

CORBA technologies may be suitable to implement objects distributed in several levels, but there is a lack of standardization in the definition of accelerator objects (in which CERN and ESRF are working together), and in the application programming interfaces (in which CDEV [4] can be considered an step towards the confluence of object based control systems and EPICS).

At the low level, software standardization is difficult due to the diversity of the equipment used. Nevertheless, the use of proper drivers and libraries to encapsulate the hardware details can always help.

# 4 PROPOSAL FOR THE LSB

An schematic diagram of the LSB control system hardware architecture as we understand it up to now is shown in Figure 1. What follows is a description of the system that we propose.

## 4.1 Control system hardware

The HLPs consoles will be implemented as UNIX workstations. The usual servers (printers...) and a powerful workstation for database management, simulation codes, etc., will also be connected to a local network that we have chosen to be an FDDI ring. We will have more than one console connected to this network, in order to inter-substitute them in case of failure.

Despite of this choice, the PC option remains the second alternative and will even be the solution adopted if the adaptation of the control software to this platform is considered to be feasible after an evaluation.

[1] Although this is not necessarily the case of accelerator control.

At the middle level we have chosen as first option the VME standard. These crates will be distributed around the ring and will have, in addition to connections to the high and low levels, a data storage capable of containing all the data taken in one full running day. The MLPs will be powerful embedded processors running real-time UNIX or similar.

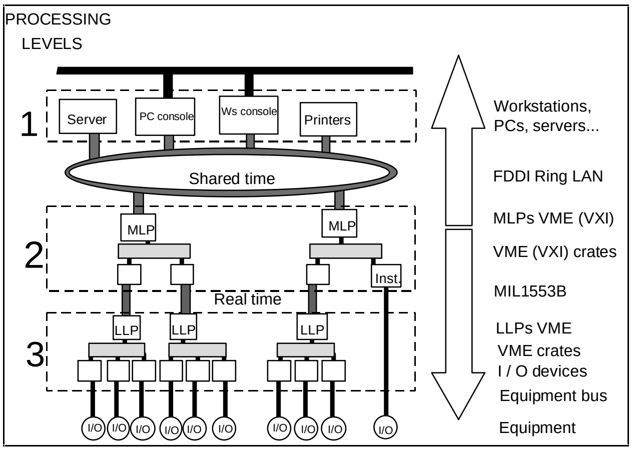

Figure 1. Hardware architecture proposed for the LSB.

The VXI option at this level is not yet discarded and will actually be the choice if we find severe constraints with our initial architecture. Nevertheless, VME will, with no doubt, fit better within our budget envelope.

At the low level, VME is certainly suitable. A typical configuration contains ADCs, DACs, digital outputs, and a processor. We are considering the use of MIL-1553B for the connection with the MLPs, whereas the connection with the equipment is not yet decided and is of course dependent of the equipment selection.

With this proposal, we try to reduce as much as possible the diversification of the number of standards required with the corresponding benefits that this implies in the reduction in time and difficulty of programming, and cost. Nevertheless, we are aware that in practice this goal is very difficult to achieve.

### *4.2 Control system software*

Due to the benefits in both robustness and cost, we have decided to choose EPICS as our main software environment. However, we are aware of the enormous benefits that the OO technologies can give us and for this reason we plan to make system upgrades in that direction, using CORBA or other approaches.

In any of the choices, the software proposed is of a multi-layer type: each of them using the layers below it and serving the above ones. The independence of the layers allows, on one hand, to work with software in different places with minimum interaction and to update it in an easier way; on the other hand, this gives great modularity allowing us to smoothly include procedures implemented using OO techniques.

The proposed software layers are mainly:

- Management of the I/O devices that connect to the equipment
- Communication between the three hardware levels
- Concertation of actions on several equipment
- Database and alarm management
- User interface.

## 5 CONCLUSIONS

We have described the currently open items of the SM of accelerator control systems, and we have discussed the present alternatives for its implementation.

On the basis of this discussion, a first proposal for the LSB distributed control system has been made. The relevant elements from the hardware point of view are to use UNIX workstations as HLPs but with big expectations of using the new state-of-the art PCs running Windows NT. Our aim is to use PCs as soon as we prove that EPICS can be adapted to this platform.

At the medium and low levels we have chosen the VME standard real-time environments but we are still open to use VXI for the medium level depending on the budget constraints.

Concerning the buses we have chosen to use FDDI to make the connection between the HLPs and the MLPs and we plan to use MIL-1553B as the connection between the MLPs and the LLPs.

About the software architecture we think that EPICS is a good candidate but we are very interested in its extension using modern OO technologies, according to the benefits that one can obtain using them. In any case we will implement a multi-layer and modular design since we are convinced of the several advantages that this approach presents.

We are planning to evaluate all the above components in a prototype test laboratory and to make the final choices according to this experience.

## REFERENCES

[1] Pak, C.O., S. Kurokawa, T. Katoh, Eds. of ICALEPCS Proceedings, Tsukuba, Japan, 11-15 Nov. 1991, KEK Proc. 92-15.

[2] Dalesio, L.R., et al. “The Experimental Physics and Industrial Control System Architecture: past, present, and future”, ICALEPCS 93 Proceedings, Nucl. Instr. and Meth. in Phys. Res. A, **352** (1994) 1-5.

[3] Di Maio, F. et al., “Towards a Common Object Model and API for Accelerator Controls”, ICALEPCS Proceedings, Chicago, IL, USA/30 Oct. - 3 Nov. 1995, *in press.*

[4] Watson, C., et al., “CDEV User’s Guide”, CEBAF, USA, July 1995.