LNLS CONTROL SYSTEM

P.F. Tavares, A.R.D. Rodrigues*, R.H.A. Farias, G.S. Franco, L.C. Jahnel, G. Monteiro and J.R. Piton, LNLS, Campinas, SP 13081-970 BRAZIL

The control system of the LNLS Synchrotron Light Source facility is presented. The system is composed of a number of local controllers based on the Zilog Z80 CPU, which are connected through a 2 Mbps serial line (running an in-house developed communication protocol) to concentrators which provide the interface to microcomputers, which are responsible for the high level user interface. The high level software is developed using object-oriented techniques and visual design tools, whereas the low level software is mostly assembly language.

1. INTRODUCTION

LNLS is a materials science research facility built around a synchrotron radiation source based on a 1.2 GeV electron storage ring currently under commissioning. The storage ring and the injection system (a 120 MeV Linac) are described elsewhere[1].

In this paper we describe the low and high level software and hardware comprising the machine control system. The system has been in operation since December 1995, when commissioning of the injector started with the preliminary version of the high level software. System performance and reliability have improved steadily since then and have now reached the design specifications, demonstrating that the initial conceptual design was basically sound.

2. Low-Level Hardware and Software

The Control System is divided in three levels (Fig. 1). The equipment level is the lowest one, in direct contact with the equipments being controlled. It is formed by home-made Local Controllers (LOCO), based on the Z80 microprocessor. A LOCO crate is composed of a serial and a CPU cards in addition to control cards attached directly to the equipments. The intermediate level, called distribution level, is formed by a Concentrator, which is composed of various serial cards. Each of these serial cards is connected to a number of LOCOs forming a control system network segment. Each LOCO is called a node of the control system network. All data coming from the local controllers are condensed in a Dual-Port Memory device (DPM) in the serial card.

Finally, the user interface level is represented by a low cost PC connected to the concentrator. The communication between the concentrator and the PC is carried out via a PCLOCO card which is installed in one of the ISA bus slots of the PC and is directly connected to the DPM of the various nets connected to the concentrator rack.

Local Controller

The interface between the Control System and the equipments in the facility is managed by the local controllers. The control interfaces in their lowest level —at the equipment level— have also been developed at LNLS, using TTL digital signals and analog signals in the ranges of 0 V to +10 V and -10 V to +10 V, with 12 or 16 bits resolution.

The software for the local controller is written in assembly language and is standard for all controllers, regardless of the specific equipments controlled.. Thus, all controllers run strictly the same software, which makes the general maintenance simpler. This

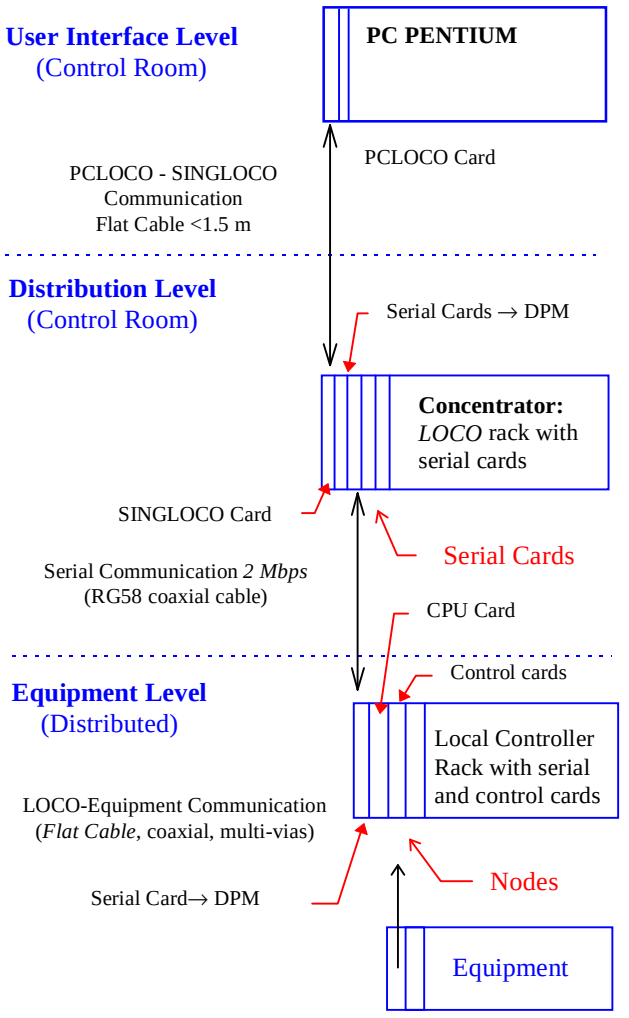

Figure 1 - Schematic layout of the Control System Structure

standardization required the implementation of algorithms that allow the Local Controller's CPU to automatically recognize the cards being controlled and to follow the appropriate management procedures. Distributed databases are thereupon not necessary.

The card recognition procedure is carried out during the boot operation by means of CPU calls to preestablished input addresses, which return an indication of whether or not a card exists as well as its type. The CPU then allocates the memory necessary for the card and calls the specific management routines. The allocated memory is mirrored in the serial communication card responsible for the final data exchange with the higher levels of the control system

Once the boot is completed, the CPU starts the data transmission from the equipment control cards to the serial card, and periodically checks the serial card for the existence of fresh data to be sent to the equipments. It must be remarked that the local controllers do not have any information about the equipments which are being controlled but only about the cards which are being used. This increases the flexibility as regards modifications which may be eventually necessary in the control system, since it requires only updating a database in the control computer. On the other hand, the accidental insertion or removal of a card from a Local Controller would cause data transfer to the higher level control software to fail. Algorithms for the detection of such faults are currently being implemented.

Communications

A 2 Mbps serial card is responsible for autonomously transmitting data between the Local Controllers (equipment level) and the Concentrators (distribution level). The CPU processing time is then employed exclusively in the control of the equipments. Data transfers within the Local Controller use a Dual-Port memory device. The serial card collects the data to be sent to the higher levels of the control system and makes those received from the higher levels available to the CPU. Data are periodically collected by the concentrator, which requests data from each node in the control system network individually, thus preventing collision events in the transfer process. In the concentrator, data from each node are stored at preestablished addresses of a DPM and can then be accessed by the PCLOCO card. From the high level software point of view acting upon an equipment means reading or writing a byte at a specific I/O position of the PC port.

3. High Level Software

The high level software layer implements the user interface and machine physics algorithms, and manages static and dynamic machine parameter databases.

The user interface is implemented on Intel PCs running Windows95® with the Pascal-like object-oriented visual-design environment DELPHI®. The choice of the development environment was dictated by time constraints regarding both performance requirements and programming ease, allied to the user-friendly interface provided by Windows applications.

The user interface comprises a series of modules (Pascal units defining a window or *Form*) which allow the operator to access every single equipment in the facility. All these modules share a common dynamic database (implemented as two vectors of bytes in memory - one for the values sent to the local controllers and the other for values read from the local controllers), which contain all information on the status of the machine (Fig. 2). This dynamic database is created, initialised and maintained by an independent application (the so called SERVER) which is permanently running in the background and which communicates with the various user-interface modules via DDE (Dynamic Data Exchange) links. Thus, several independent interface

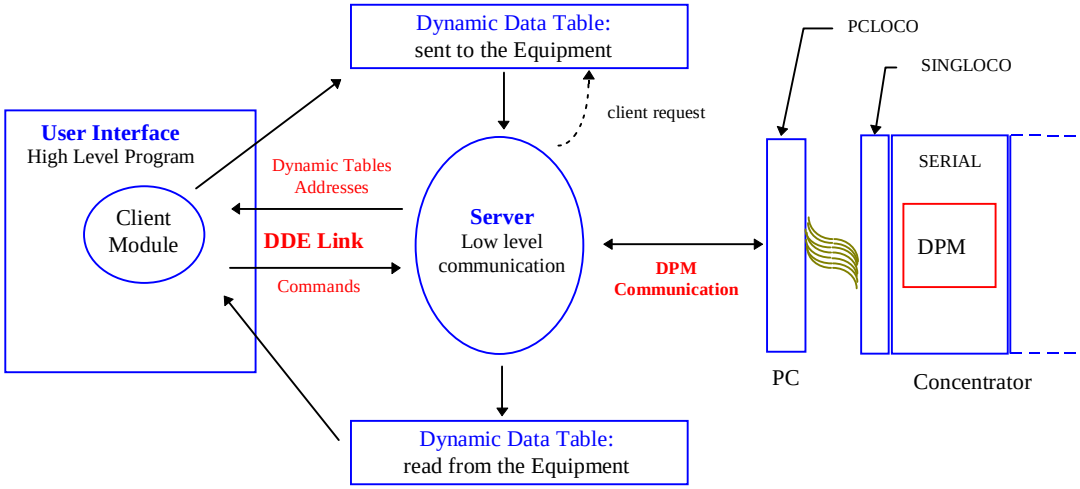

Figure 2 - Schematic layout of the high level software configuration.

modules (even modules belonging to different applications, perhaps running in different machines) can access the same dynamic database via the server. It is the server's task to interact with the lower levels of the control system software and hardware: it permanently reads the status of all equipments in the system and answers to requests from its clients to send new settings to the local controllers or to prepare special communication protocols (such as in preparing do dump a ramping curve before ramping the machine to high energy).

The DDE Server is a fairly short application written with performance in mind and contains no knowledge of the inner workings of the various equipments connected to the local controllers. In contrast, the client modules must access various static databases describing detailed properties of the controlled equipments such as conversion constants, calibration curves, etc. The strategy of dividing the pure user-interface tasks from the actual communications and control tasks into two independent concurrent applications proved very useful because it allowed the many client modules to be divided into various applications, thus minimising memory limitations inherent to the Windows environment.

The client modules are divided into 9 applications. The 3 main applications (LINAC, Transport_Line and Storage_Ring) contain most of the control windows for individual equipments in the three subsystems. The remaining six applications perform the following tasks: (a) continuously monitor the interlock system, (b) condition (cycle) the magnets, (c) routine checking of all control system network nodes and equipments prior to machine operation, (d) on-line detection of equipment faults, (e) preparation and management of the ramping process and (f) management (saving, loading, printing) of machine configurations.

Some client modules act on groups of equipments performing simultaneous coupled changes in various machine settings, according to predefined beam dynamic's procedures. It is possible, for example, to use a pair of correctors to adjust independently position and angle of the beam at the injection point. Pairs of quadrupoles in the transport line can also be adjusted in order to empirically match the vertical dispersion function of the line.

All client modules make heavy use of static databases which are implemented in the form of Paradox 5.0 tables, directly accessible from DELPHI programs. Although this implies a certain performance penalty since table searches must be performed at run-time to set up equipment parameters for the interface modules, the ability to edit these tables on-line (without the need to recompile the code) has proved to be extremely useful during commissioning of the system, when some of these parameters are still subject to frequent changes.

Beam Optics Modules

Standard beam optics simulation tools, such as the machine optics design code MAD[2] as well as in-house developed orbit correction codes are installed in a remote UNIX workstation. These beam optics applications can be accessed from the control room PC's via front-end modules developed in DELPHI. The main capabilities of these front-end modules are: the direct interface with the main modules of the control system, which allows on-line acquisition of machine parameters to be fed to the simulation programs; the user-friendly interface which eases the manipulation of input files; post-processing tools which permit fast analysis of the outputs as well as sending new machine settings to the control system. The connection between the PC-based codes and the workstation is accomplished via a LAN running the TCP-IP protocol.

4. CONCLUSIONS

The control system of the LNLS synchrotron radiation source has been in operation for six months. Its reliability and performance have met the design parameters. Enough flexibility was built into the system design so as to allow several software tools to be quickly added to the system during the commissioning stage in response to demands from the various technical groups involved in the machine operation.

The present configuration of the Control System includes a 90MHz Pentium PC and a concentrator which is connected to 7 network segments making up 305 control boards. There are at present about 3200 control points dealt with by the control system software.

Most of our network communication problems up to now have been minimised by redesigning the topology of the network, minimising the length and number of nodes of each network (while keeping the number of nodes constant). Apart from these hardware solutions to such difficulties, software corrections (in the form of more rigorous transmission checking routines) are also being implemented in the serial communication protocol.

5. REFERENCES

* Also IFQSC, Univ. of São Paulo, São Carlos, Brasil

[1] A.R.D.Rodrigues et al, LNLS Commissioning and Operation, these proceedings.

[2] F.C.Iselin, The MAD Program, CERN/LEP-TH/88-38.