PSI PROTON ACCELERATOR CONTROL SYSTEM UPGRADE

D. Anicic, M. Gasche, H. Lutz, A.C. Mezger
Paul Scherrer Institute, Villigen, Switzerland

Abstract

The I/O Computers (IOC) in our Control System are, since over ten years, VME based. The data acquisition and control equipment was primarily based on bit serial CAMAC, interfaced through VME to CAMAC modules. The interfaces to PLC or other kinds of equipment were always implemented as CAMAC modules. In the future we will use VME modules whenever applicable, and communicate with PLC-s directly over TCP/IP. The same PLC communication will be used to interface the industrial PLC based control system for the new PROSCAN project. Support for centralized write (set value) logging was implemented, too. This is highly useful in debugging distributed applications. Concurrently we are also replacing the existing HP rt743 IOC-s running under HP-RT operating system with the new MVME51xx running under Lynx OS.

IOC REPLACEMENT

Motivation

The motivation for the IOC upgrade and replacement program is twofold. Firstly, the present HP rt743 single board VME real-time computers, under the HP-RT operating system, are over ten years old and will not be supported for a much longer time anymore. Secondly, we have new projects based on VME modules. The upgrade has already started. We use MVME 51xx VME computers with the Lynx OS operating system. This has been already announced at [1].

Old IOC

The old IOC computers (formerly FEC, Front-End Computers) are VME based HP rt743 with HP-RT OS. They are now already aged (over ten years), and running out of support. The controlled equipment was CAMAC with the addition of the home-developed orthogonal bus, ROAD-C, interfaced through a CAMAC module. PLC devices are connected through ROAD-C based RS232 interface. Physical interface to CAMAC is the CERN developed "CAMAC Serial Highway Driver in VME" (L. Antonov, V. Dimitrov, L.Heinze). The IOC software (called Services) implementation was made with mostly only CAMAC in mind, but it's design [2] and architecture has allowed for, the now welcome, extensions. We will continue to use these IOC computers, since we did not have a single failure in all this years of any of our 14 IOC-s.

New IOC

The CAMAC equipment is becoming more and more outdated, harder to obtain and to repair. On the other hand, there are many industry VME products, and our own hardware developments are driving in a VME direction, too. The existing IOC-s are VME based, but it was just that VME equipment was in any case not used before. The new IOC-s were chosen to fit the existing VME crates in case of the HP rt743 failure and to provide for the coming needs. The Motorola MVME 51xx PowerPC single board VME computers running Lynx OS satisfy these needs. They provide two Ethernet interfaces, one for the Controls network and the other one for the IOC's private PLC network. The same IOC was chosen in collaboration with the SLS (Swiss Light Source) division of PSI. The IOC software was extended to support VME modules and TCP/IP communication to PLC-s. Since then, we have three new IOC in operation. The Figure 1 shows the new IOC hardware configuration.

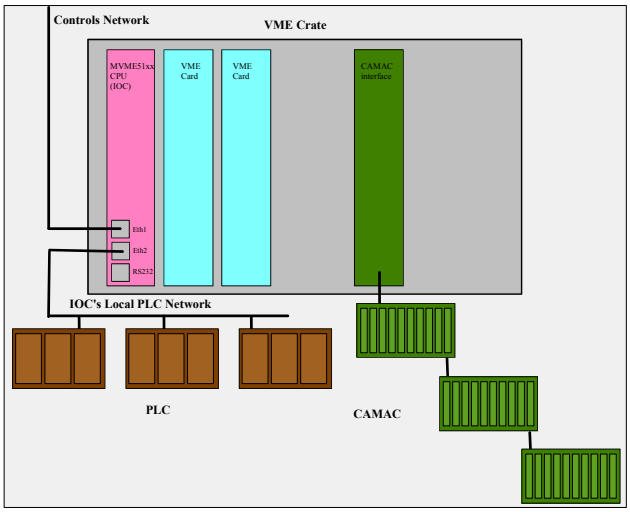

Figure 1: The new IOC hardware configuration

VME INTEGRATION

For the VME (and PLC) integration, the structure of the IOC software had to be slightly modified. The device object class, **DEVICE_OB** (Fig 2.), was using CAMAC addresses only. The create method was modified to accept either CAMAC address or the VME/PLC configuration (*cdriver* and *configStr* strings). According to the *cdriver* the appropriate **C_DRIVER's** *init_HW* (create) method is called with corresponding *configStr*. It parses the configuration string, detects the hardware and does the necessary initialization. Further on, the **DEVICE_OB's** methods (*read, write*, …) are adjusted to use **C_DRIVER's** methods. The **C_DRIVER** class (Fig 3.) is implemented in C language, to reduce the number of classes. The original software is written mostly in Sather OO (Object Oriented) language. The initial design required one new class for almost any new device functionality. The **C_DRIVER** for the VME IP modules

uses the **CARRIER_BOARD** class (Fig 3.) which does the carrier board initialization part. The IP modules are addressed as A,B,C and D, and a special address L denoting the carrier board's internal registers, or additional dual ported memory in a case of the DSP equipped carrier board. Both classes implement the *isGreen* method, which is used to signal false database configuration (needed HW module is not in that VME or IP slot). It could be used to handle hot-swap (detected through VME bus errors) if we find it necessary in the future.

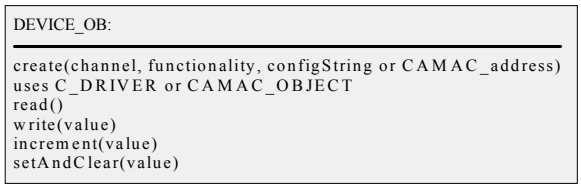

Figure 2: simplified DEVICE_OB class

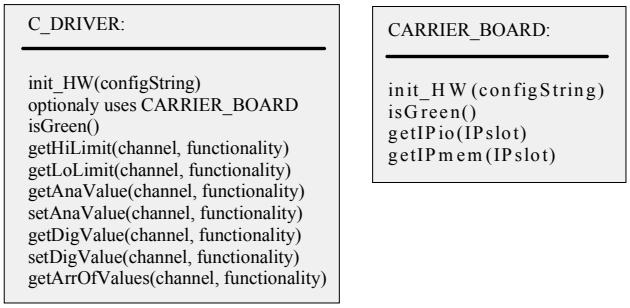

Figure 3: simplified C_DRIVER and CARRIER_BOARD class

PLC INTEGRATION

From the IOC software point of view there is no difference between VME and PLC equipment down to the C_DRIVER interface. Each C_DRIVER class then implements it's own communication with the connected "hardware". For PLC equipment, the client (the IOC) communicates with the PLC "hardware" on TCP/IP, connection mode. To be able to detect if corresponding PLC is connected, we have defined the PLC message format (Fig 4.).

The message header provides byte order and string byte order detection, message size (in 16 bit words), message type and detection of stale PLC. The object descriptors contain the device names, which must be the same as official control system device names, type of the object and offset to its data in Payload Data space. Object Types are tightly coupled to Message Type (in a PLC C_DRIVER implementation) and define the structure of the object's data. For any new PLC message type the PLC programmer has to provide the structure of the used object types which then has to be implemented in new C_DRIVER class. On connect, the client (IOC) checks the message size and format for validity, checks the used device names and hooks them to the control system device objects. Any inconsistency/error results in a break of communication, and repeats the connect procedure. After successful connect only the Payload Data section and Alive counter in a Message Header are allowed to change. Any other change results in a break of the communication and restarts with a new connect.

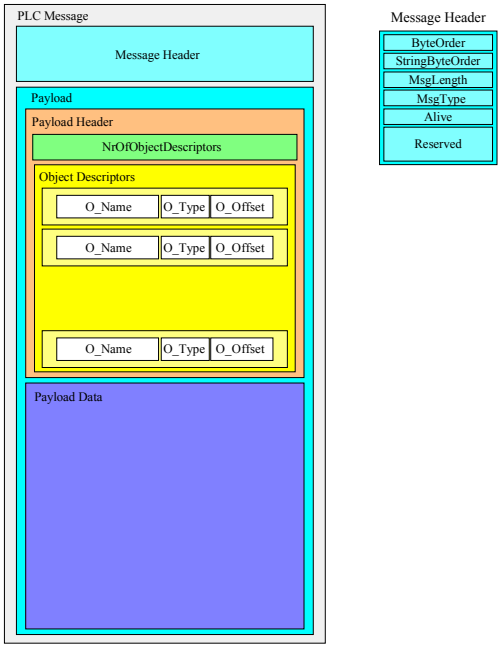

Figure 4: PLC message format

SET VALUE LOGGING

The applications in our control system run on operator workstations and on dedicated server workstations. In some cases an application may consist of several stand-alone programs. In such situations some sort of inter-process communication and synchronization is needed. This is in turn sometimes complicated to debug. To be able to debug such applications more easily, we have modified the IOC software to report all messages containing any non-read operations to the dedicated "Set Value Logging" server.

It stores the data for later retrieval and it's companion retrieval/display application can show the incoming data in historical or real time mode. This gives us the possibility to locate which application, from which operator or server workstation, and under what circumstances has set which value. The time related dependences from different applications can also be detected, helping debug distributed applications. The stored data is kept for approximately one week, which is sufficient for the purpose. The amount of stored data is approximately 2 gigabytes per day, which is more than 60 million values. About one third of it, 20 million, are non-read values (approx. 250 per second). The most of these set values come from feedback applications for orbit centering, ion source regulation and automated filed corrections.

The retrieval/display application (Fig 5.) is written in Java. It can display the graph trends for clients and IOC-s, either a total or for each separately. The historical or the real-time values can be displayed in a compact or in a full

form. The filtering functions can be used to reduce the amount of displayed data, or to show the data for the particular workstation or IOC or application or Device, … The filter is a logical expression (>, <, ==, !=, and, or, not, bit operations, string operations, …) on all of the message and device parameters.

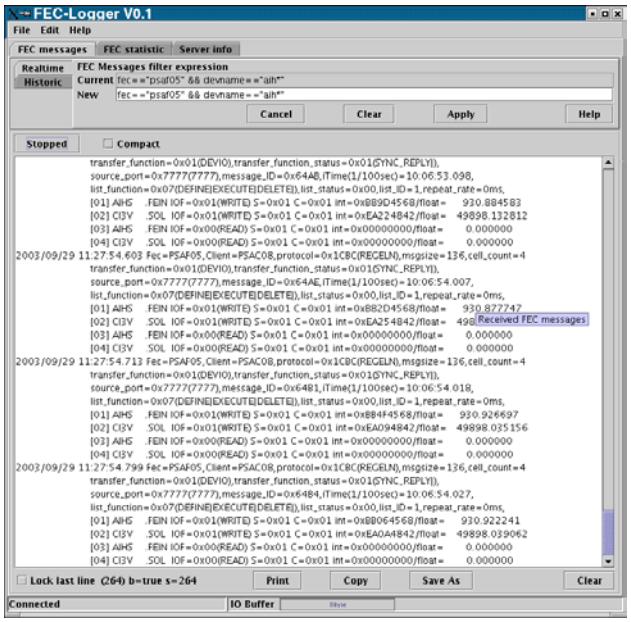

Figure 5: Set Value Logging retrieval application

FINAL WORD

The planned control system upgrade is on schedule. The framework to be able to use VME modules and to communicate with PLC-s over TCP/IP has been implemented. Adding new VME modules or new PLC message formats requires just one, quite simple, C file (see C_DRIVER above). Currently implemented is PLC message format from our vacuum system group. The supported VME IP modules are: Hytec 8401 ADC, Hytec 8402 DAC, Hytec 8501 digital I/O, PSI RPM (Run Permit Module). The supported carrier board types are: Hytec 8001, Hytec 8002, Hytec 8003 (with DSP). VME modules: OMS VME58 eight channel motor controller. With DSP equipped Hytec 8003 carrier board, we plan to implement various "intelligent" (called KOMBI) controllers. They hide the actual IP modules used, presenting their own register/memory map towards the control system (C_DRIVER). Currently implemented are two versions for digital power supply controllers connected by optical link. A very important job was done in a migration of our configuration database from VMS to Linux, and extensions to cover VME and PLC requirements [3].

REFERENCES

[1] D. Anicic, "Replacement of Magnet Power Supplies, Control and Field-bus for the PSI Cyclotron Accelerators", ICALEPCS'01, San Jose, California, 2001.

[2] Z. Sostaric, "Modern Design of a Fast Front-end Computer", ICALEPCS'93, Berlin, Germany, 1993.

[3] H. Lutz, "Migration of the Configuration Database for PSI Cyclotron Accelerators", ICALEPCS'03, Gyeongju, Korea, 2003.