ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S07IC04

A Front-End System for Industrial Type Controls at the SSC

D. R. Haenni
Superconducting Super Collider Laboratory*
2550 Beckleymeade, Dallas, Texas 75237 USA

Abstract

The SSC control system is tasked with coordinating the operation of many different accelerator subsystems, a number of which use industrial type process controls. The design of a high-performance control system front end is presented which serves both as a data concentrator and a distributed process controller. In addition it provides strong support for a centralized control system architecture, allows for regional control systems, and simplifies the construction of inter-subsystem controls. An implementation of this design will be discussed which uses STD-Bus for accelerator hardware interfacing, a time domain multiplexing (TDM) communications transport system, and a modified reflective memory interface to the rest of the control system.

I. INTRODUCTION

The design of a control system for the SSC faces significant challenges arising from its immense physical size, plethora of control points, and wide range of time scales for controls. At the slow end of the time scale there are accelerator subsystems like cryogenics, vacuum, low-conductivity water, and industrial cooling water which use industrial process controls. For a proper perspective consider the vacuum controls. A recent estimate indicates that there will be 50,000 to 60,000 points divided among 200 niches around the ring. This number, which does not include vacuum controls for the injector accelerators or beam lines, is roughly comparable to the total FNAL control system. The cryogenic system is 2-3 times larger than the vacuum system. The controls for these SSC accelerator subsystems totaling more than 250,000 points dwarf most large industrial and high-energy physics control systems. The present paper describes a possible front-end system for process controls at the SSC. An earlier discussion of this system can be found in Ref. [1].

II. DESIGN REQUIREMENTS

The controls for these subsystems at the SSC must support process controls and smoothly integrate into the rest of the SSC control system. A general list of requirements appropriate for this discussion are as follows:

1. Provide industrial process controls.
2. Minimize hardware in hostile or radiation areas.
3. Use open, widely available industrial standards.
4. Prefer commercial products to custom hardware.
5. Set no limits on the number of control points.
6. Execute control functions at different system levels.
7. Support a centralized control system architecture in which the entire complex is operated from a single location.
8. Provide regional level controls to support commissioning and maintenance of large accelerator components.
9. Eliminate accelerator hardware data hiding in the control system architecture.
10. Facilitate near real time inter subsystem information sharing and implementing geographically distributed or inter subsystem control loops.
11. Contain costs.
12. Maximize reliability.
13. Follow the overall SSC control system architecture.

While meeting many of the process control requirements for individual subsystems, commercial process control systems have problems supporting either the large number of control points or extended geographical area or desired level of integration.

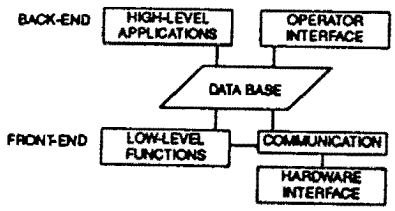

Figure 1. Front-end system components in relationship to the rest of the control system.

III. FRONT-END SYSTEM DESIGN

In Fig. 1 the basic components of the front-end system are show in relationship to the rest of the system. The front end consists of the interface hardware, communications, and processors to execute low-level control functions. It supports data I/O and functions such as loops, alarm monitoring, and interlocks. The back end is commercial and SSC developed software providing operator interface and high-level control applications such as complex sequencing, expert systems for alarm analysis, data logging, and process simulation. Effectively the high-speed reflex-like control activities are

*Operated by the Universities Research Association, Inc., for the U.S. Department of Energy under Contract No. DE-AC02-89ER40486.

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S07IC04

assigned to the front end while the slower more complex processes are placed in the back end. This organization naturally allows the high-speed controls to be distributed near the I/O interface when needed. The interface between the front and back ends is shown as a database. This will likely consist of several different types of databases coupled with data access and communications management software. In operation the front end continuously updates the database at a rate which would maintain a near real-time snapshot of the control points. The back end has high-speed random read access to the data as needed by the different applications. Commands to change output settings normally must pass various sanity and permission tests before being committed to the hardware. They are best processed as messages.

A three level architecture is suggested for the front end. The global level is the central SSC control room. All data from the accelerator complex is available at this location. The global level supports a back end for all process subsystems and other higher-speed SSC controls. Low-level functions executed at this level are able to span the entire accelerator complex. The regional level would cover a part of the accelerator or a major component. A regional system would have access to the regions data and would support back ends but only for hardware covered by the region. Regionally supported operator interfaces are intended for commissioning, maintenance, and emergency backup. It is envisioned that most of the low-level control functions would be executed at the regional level. The bottom level contains the hardware interfaces. Each system can see only the I/O interfaced directly to it. The primary function of these systems is to act as a concentrator for control points. They will support minimal operator interfaces for maintenance. When needed low-level control functions can be added to an interface level system. There is 1 global level system, less than 100 regional level systems, and 2000-3000 interface level systems.

A basic feature of the SSC control system will be the use of telephone technology based communications in lieu of LAN based communications. The idea is to provide a large number of independent point to point communications paths with dedicated band width over a TDM system. There is no degradation of performance even if all paths operate at 100% capacity. The telephone industry continues to push TDM technology to higher data rates thus giving rise to the notion that communications is not a limiting factor in the design of a front-end system. For a LAN based system this is not the case. Since performance decreases with increasing load one must design a system which minimizes communications usage.

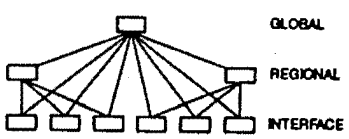

Figure 2. Front-end system communications links.

A possible set of communications links between the three front-end system levels is shown in Fig. 2. Each interface level system is connected to a regional and the global systems. Regional systems also connect to the global system. Thus the interface level delivers the same communications performance to both the regional and global levels. This would not be the case if the data had to flow through the regional system on its way to the global system. Depending on the TDM configuration this scheme allows for redundant communications paths. The role of a failed regional processor could be picked up by the global system thus providing backup processing without having to supply a duplicate of each regional level system.

The communications protocol over the TDM links may be chosen to minimize data and processor overhead. It has been decided to implement communications using reflective memory. Reflective memory is similar to a shared memory except that instead of one physical memory being shared by two processors there are two memories connected by a hardware implemented communications link. When a value is changed in one memory there is a small delay before it is reflected in the other. Hardware implemented reflective memory reduces the communications overhead on the processors to memory access. Such a link is proposed for the connection between the global and regional systems. This would allow the global system to monitor and change the parameters of the regional low-level function processing and provide a consistent basis for switching from regional to global back-up control. Note that this communications link is only for front-end operation. The regional and global systems are also connected by LANs for other types of communication.

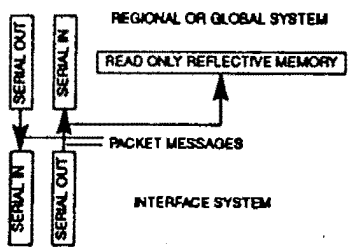

Figure 3. Communications model for the global/regional to interface level links.

Communications with the interface level systems uses a modified form of reflective memory. This is outlined in Fig. 3. Effectively one combines packet (serial) communication with read-only reflective memory at the upper level end of the link. The high-end communications interface interprets some messages as reflective memory updates and others as serial communications. The link to the interface system uses only packets. This scheme takes advantage of the above mentioned message nature of commands and the fact that control points are more often read than written. Thus a single link can serve both reflective memory and serial communications. This provides for program down loading,

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S07IC04

remote debugging, and hardware interface testing. Finally the communications interface is simplified for the front-end level which has the most systems.

The global and many of the regional systems will integrate controls from several process subsystems and/or higher-speed accelerator subsystems. Exact details of the hardware and software for these systems are beyond the scope of this paper. Of importance to the process controls front end is the need for a database and software to manage the configuration of and access to the reflective memory data. A second database would provide a repository for system run time parameters like set points or alarm limits. The former database is a constant of the hardware configuration while the latter could change dynamically during accelerator operation. These databases are extracted from a central configuration database and must be maintained consistent between all systems. This will be accomplished using communications links and software outside of the front-end system.

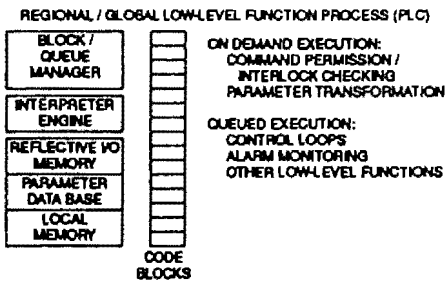

Figure 4. Components of the global/regional level server process for low-level control functions.

The global and regional systems support the front-end system process which provides a server for the low-level control functions. A schematic idea for this process is shown in Fig. 4. It would contain a large number of more or less independent code segments each of which carries out some specific control function. The code in these blocks (likely tokenized) could be executed by an interpreter which supported several specific languages for different types of blocks (e.g. ladder logic, control loops, alarm logic, data transformation, etc.). These blocks would have access to reflective memory data, parameter data, and local data which would be reflected between regional and global systems. A block manager would be able to dynamically add and delete blocks from the list, schedule their execution, and monitor block memory access. It should be possible to safely modify control strategies for various devices without shutting down the whole system to make the changes. Blocks could be scheduled for repeated execution in various queues thus forming a kind of PLC. Other blocks could be run on demand to provide interlock/permission checking or transformations from raw values to engineering units. This process should be maximized for efficient block execution by perhaps running on dedicated hardware and closely integrating with reflective memory and parameter access software to minimize execution overhead.

The hardware and software at the interface level would be entirely within the front-end system domain. It is envisioned that interface systems would be based on an open, industrial standard bus supported by a number of vendors supplying inexpensive industrial interface cards. An interface crate would contain a TDM communications card and a minimal I/O processor, IOP, whose task was to manage communications. When needed a second processor could be added which provided PLC functions and had its own communications capability. The IOP would provide data for the PLC at the same rate as the regional and global levels. Fig. 5 shows a schematic of systems with and without the optional PLC. Interface level systems would allow direct terminal attachment for bench testing and/or emergency low-level interface maintenance. Part of the general control system architecture is an operator station network that connects the global and all regional level systems with operator stations. This network will extend to all technical areas. With a portable operator console it will be possible to test the entire control system linkage to a given device while being at the device location.

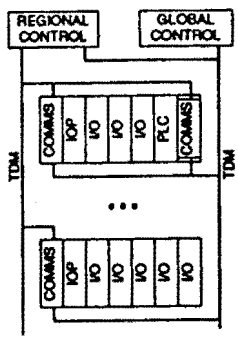

Figure 5. Schematic of interface level systems with and without an optional PLC. Connections to upper level systems are also shown.

The proposed front-end system is a combination of both hardware and software. Care must be taken in designing the software for several reasons. First there are a large number of I/O processors, regional/global level code blocks, and perhaps interface level PLC systems. Secondly, this software is very dependent on the configuration of the interface hardware. Next the system levels are linked through the layout of the reflective memory. Finally, considering the wide variation in controls at the SSC it is unlikely that there will be the uniformity between systems which would allow programs to be used in a large number of systems with only minor modifications.

For these reasons it is impractical to consider programming each system separately in traditional ways. The software generation must be automated to a large extent. One way to accomplish this is shown in Fig. 6. Here the main technical database which holds the configuration of the

ISBN: 978-3-95450-254-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS1991-S07IC04

accelerator and control system forms a data source to automate the program generation. This database provides the configuration of the reflective memory to the regional/global systems and provides crate configuration, I/O attachment, and reflective memory update message format information to an intermediate compiler which generates the IOP code. Code blocks and PLC programs could also be captured in the database through macro statements again with the code being regenerated through appropriate translators. As suggested in the figure, users would not directly modify programs but rather change the database which in turn modifies the front-end software. Such a scheme can effectively enhance code reuse, simplify programming, and provide a single data source for describing the system. Some of these ideas have been successfully used in an earlier cyclotron control system[2].

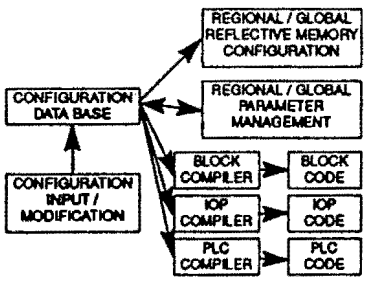

Figure 6. Possible method of automating front-end system software development through the use of a configuration data base

IV. TEST IMPLEMENTATION

A prototype of an interface level of this system is under development at the SSC. The initial system will be based on STD-Bus, IEEE-961, which is an inexpensive, open, industrial standard bus that has been used extensively for control applications in many areas. It is a well established bus supported by a large number of vendors. While initially an 8 bit bus, recent extensions to 16 and now 32 bit protocols give indications that the bus will not quickly fade or be replaced in the time scale of the SSC control system. The bus specifies a straight forward interface to the backplane thus greatly simplifying the construction of custom interface cards when needed. The bus normally uses processor cards having INTEL 80xxx or equivalent cpu chips. This gives direct access to the large amount of excellent and inexpensive development software available for the PC platform.

A dual channel STD-Bus communications card which can extract a 64 kbit/sec data stream for a T1 communications line has been designed, constructed, and is undergoing tests. This will allow up to 24 STD-Busses to be connected to a single T1 line.

An estimate of the ability of an STD-Bus system to provide communications has been made. Assuming a 64 kbit/sec communications band width and a crate with 128 analog input signals attached to standard 32 channel multiplexed 12 bit ADC cards, an 8 Mhz 8088 processor could update 256 bytes of reflective memory in both a regional and global system at 20 Hz using only 34% of the cpu. The output link is saturated to about 70%. This example was chosen to show worst case throughput since the cpu must manage the ADC multiplexing. Quantity one pricing for this system consisting of 4 ADC cards, cpu card, communications card, crate, and power supply is on the order of $4000.

Several STD-bus manufacturers produce slave processors which have shared memory on the STD-Bus but cannot initiate backplane data transfers. These systems have ISBX I/O connectors and the communications board can be easily modified to become an ISBX daughter board. This hardware configuration could support the local PLC shown in Fig. 4. PLC software which is portable to any INTEL 80xxx embedded system is available commercially[3].

The initial regional/global system TDM interface will be a commercial communications processor which directly interfaces with a T1 communications line. Enhanced communications interface cards capable of connecting to higher rate TDM lines will be developed as part of the general SSC controls system .

Initially the I/O processor will be directly programmed to produce an embedded system without an operating system or real time kernel. One option is to embed a FORTH kernel in the processor and program it over the communications line. Other options include more traditional embedded code development using assembler, C, or C++.

V. CONCLUSION

A scheme for providing highly integrated process controls for the SSC has been described. It makes use of a different style of communications based on telephone technology, supports a high-performance centralized architecture while allowing regional systems to exist for commissioning and maintenance. System prototypes are being implemented.

VI. REFERENCES

[1] D.R. Haenni *et al.*,"The SSC Field Bus: A High-Performance System Front End for "Slow" Accelerator Controls", SSCL-391, 1991.

[2] D.R.Haenni *et al.*,"Personal Computer Control System for the TAMU K500 Cyclotron", in B. Martin and K. Ziegler eds., *Twelfth International Conference in Cyclotrons and Their Applications*, World Scientific, 1991, pp 264-267.

[3] CPI Industrial Software, P.O. Box 1046, Bettendorf, Ia. 52722, USA.