SLAC-PUB-3331
April 1984
(A)

TRENDS IN ACCELERATOR CONTROL SYSTEMS*

M. C. CROWLEY-MILLING
Stanford Linear Accelerator Center
Stanford University, Stanford, California 94305

Nowadays, it is hard to think of a control system for an accelerator that does not incorporate computers, but it is less than twenty years ago that the first accelerator was designed to use computers as an integral part of the control system: the LAMPF at Los Alamos.[1] Even after that, some people were doubtful if there was a net advantage in having computers.

Over the years since then, we have seen a revolution in control systems that has followed the ever decreasing cost of computer power and memory. It started with the data gathering, when people distrusted the computer to perform control actions correctly, through the stage of using a computer system to provide a convenient remote "look and adjust" facility, to the present day, when more and more emphasis is being placed on using a computer system to simulate or model all or parts of the accelerator, feed in the required performance and calling for the computers to set the various parameters and then measure the actual performance, with iteration if necessary.

In this review I shall not limit myself to Linacs, even though this is a Linac conference, since most of the requirements are common to all types of accelerators, and many Linacs these days are being designed to have associated damping or stretching rings, but I will choose Linac systems for my examples where possible.

It is often asked why we don't go and buy a commercial process control system for our accelerators and avoid all the design and development work. One answer is that the requirements are rather different. Unlike a process control system for an industrial plant, the requirements for accelerator control can rarely be specified with any precision before it is built, as most new accelerators aim to reach beyond the limits of previous experience in at least one respect, and accelerators are constantly improved and modified during their lives. An extreme example of this is the SPS at CERN, which was designed as a pulsed proton accelerator for fixed target experiments, without the least idea that it would later be called to act as a proton/antiproton storage ring for collider physics, and is now being modified to act as injector to LEP, to accelerate electrons and positrons interleaved with proton acceleration. Such evolutions require very great flexibility in a control system, with the ability to modify programs as required in a simple way. This is in direct contrast with a control system for an industrial process. Taking an extreme example, the programs for running a nuclear power station, once a "fail-safe" procedure has been established, must be "cast in concrete" and only changed after a full review of all the possible consequences, and approval from the licensing authority.

With this introduction, I will review the progress that has been made in the fields of architecture, communications, computers, interface, software design and operator interface.

Architecture

Originally, when computers and memory were expensive, control systems were usually designed about a single computer, as large as one could afford, with as much as possible of the accelerator equipment interfaced to it, directly or through a hardware multiplexing system. The LAMPF system was of this type.

The next stage, with the advent of the relatively cheap mini-computers, was to have a two level hierarchy: a central master computer controlling a number of slave mini-computers distributed along the accelerator. These slaves carried out the data acquisition and control actions on the equipment, and acted as remote multiplexors, relieving the central computer of some of its load. Figure 1 shows an example of this type of architecture which has been used for the control of SLAC since 1974.[2] This architecture has been very popular and used for many different accelerators, with the slaves gradually taking on more sophisticated processes. The latest example is again

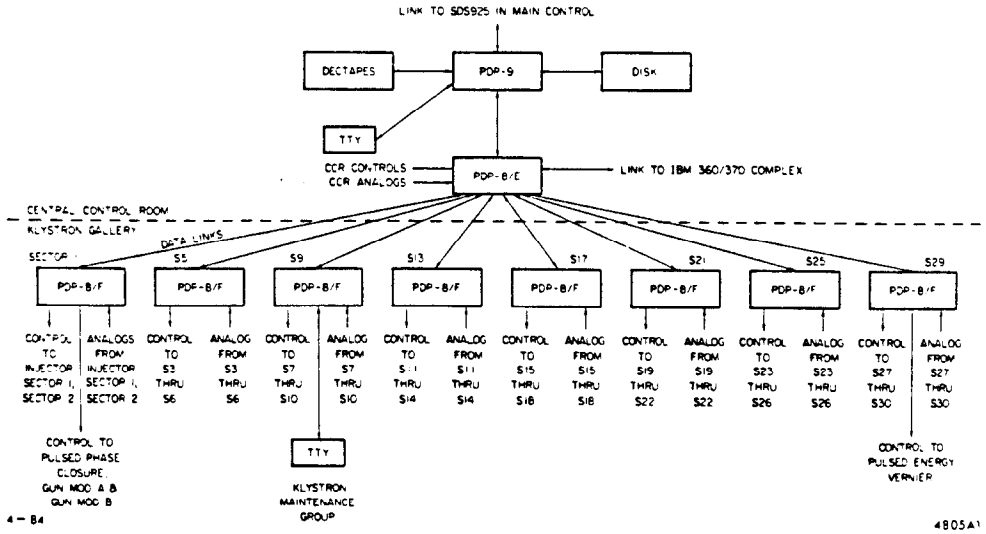

Fig. 1. Master/slave architecture at SLAC 1974.

* Work supported by the Department of Energy, contract DE-AC03-76SF00515

Invited paper presented at the 1984 Linear Accelerator Conference,
Seeheim/Darmstadt, West Germany, May 7-11, 1984.

from SLAC, where the linac control system is being redesigned for the collider program (SLC).[3] In this case, the master is a VAX 11-780, and the slaves are micro-computers which, such is progress, can outperform the best mini-computer of a few years ago.

One of the disadvantages of this type of single-master/ multiple-slave architecture is that the single master computer can prove to be a bottle-neck unless it is very powerful, as it is involved in practically all the processes.

An alternative architecture is to have all the computers at the same hierarchical level, with a communications system that allows any two computers to cooperate in a process without involving any of the others. This type of architecture was pioneered by the SPS,[4] where advantage was taken of the single level to divide the processes carried out by a central computer into separate processors as shown in Fig. 2, so eliminating a potential bottle-neck and allowing parts of the system to be developed independently. However, this single level hierarchy, where any computer can become temporary master of the system, needs a greater discipline in the organization of programs, since the possibilities for creating chaos are greater than in a single master system.

In describing these as one and two level systems, I am ignoring the micro-processors that are being used in the interface equipment and discussed below.

Communications

Any multi-computer system requires some means of communication between the computers, the equipment and the operators control desks. In the absence of suitable systems from the computer manufacturers in the past, most laboratories have had to develop their own , and these have usually been star networks involving one cable per link. As accelerators have become larger, the cable costs have become a more and more important consideration, and the possibilities of using a single cable for more than one connection, by multiplexing, have been explored. Such a system has been put into operation at SLAC, using frequency-division multiplexing.[5] A single co-axial cable is used to carry not only all inter-computer communication but also video signals, timing pattern signals, terminal traffic and fast feedback signals, using cable television techniques and equipment. Each of these services transmits on a different frequency in the 5 to 120 MHz band to one end of the cable, where an up-converter changes all frequencies by 156 MHz and transmits them back along the cable in the 160 to 280 MHz band. This enables a single cable to be used for two-way transmission.

Another type of multiplexing is called time-division multiplexing, as it involves allocation of the use of the full bandwidth of the transmission medium successively between users. Most of the Local Area Networks (LANs) one hears so much about these days use time division, but have different schemes for the controlling of access to the medium.[6] Two types are being investigated for machine control purposes. The first is usually known as the Ethernet type, in which all stations are connected to a single highway. A station can start transmitting any time the highway is free, and if two stations start transmitting at the same time, they must both stop and wait random lengths of time before trying again. The other type is the "Token Ring", so called because the stations are connected together in a ring and a special bit pattern called a token is passed from station to station. A station can only send a message when it is in possession of the token, so only one can transmit a message at a time. Either system could be used in a control system as long as the maximum load is only a fraction of the capability of the LAN, but if the loading is heavy, the token ring seems preferable for a real-time application, because the response time can be very long for a few unlucky messages in a heavily loaded Ethernet. A token ring, using optical fibers as the transmission medium, is being used for the TRISTAN control system.[7]

Another type of time division multiplexing that is being investigated for use in accelerator control is that developed for the telephone system trunk lines.[8] In this, a number of digital signals are interleaved to produce a single signal at a higher bit-rate, which can be transmitted on a co-axial cable or optical fibre link. Bit-rates up to 140 Mbits/s are in routine use, and 560 Mbits/s are used experimentally over distances of tens of kilometres between repeaters on the best optical fibres. Experiments at CERN have shown that a channel in such a system can be used as an extended data link in a token-passing ring LAN. The advantage of using such a scheme is that all forms of data communication, the telephone system and digitized video signals could be transmitted over the same transmission medium.

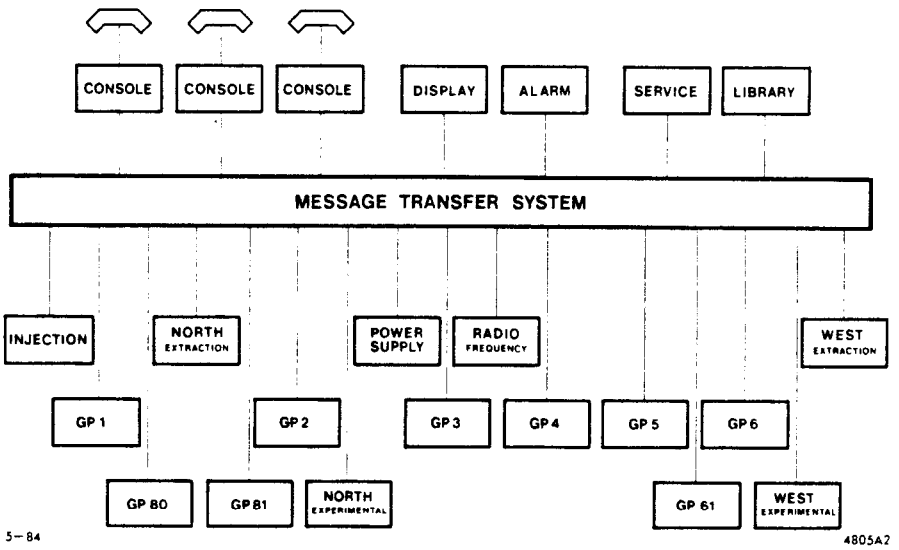

Fig. 2. Single layer architecture at the SPS 1976.

Interface

In the case of the earlier computer control systems, few computer manufacturers supplied the I/O interfaces needed to provide the digital and analogue input and output suitable for control and most systems were designed by the users. Later when the CAMAC standards were established for nuclear instrumentation modules, they were adopted for the control interface system for many accelerators. At first the restriction on the maximum length of the parallel bus was a serious limitation for large systems, but this was overcome by the introduction of the serial bus CAMAC system in the mid-1970s.[9] CAMAC has some serious disadvantages which make it relatively expensive to use for large systems, but the availability of a large number of different types of standard modules from many manufacturers, and controllers to suit many computers, make it the present day automatic choice for small and medium sized systems where the development of special interface systems is not justified. Cheaper specialized interface systems have been developed at some laboratories (MPX at CERN,[10] SEDAC at DESY,[11] etc.) but others have used CAMAC for large machines, developing special modules to make more efficient use of the system (SLAC, FERMILAB, CERN, etc.).

A recent development in interface systems is the provision of computing power at this level. It started with the incorporation of a micro-processor and memory in a CAMAC module, to make a unit usually known as an Autonomous Crate Controller (ACC), which can perform a string of CAMAC commands independently of the computer controlling the crate, and so unload the computer of some repetitive tasks.[12] This has been followed by more and more CAMAC modules containing microprocessors to do specific jobs, such as display controllers and "smart" ADCs that carry out ranging, conversion and are self standardizing.

In the future, I expect that crate and bus systems designed specifically for micro-processors will take over from CAMAC for control systems when the battle between the different standards is resolved and enough different types of input/output and other modules become available.

Already, this migration of computing power down into the interface system is going even further, into the equipment itself. When each major piece of equipment has a micro-processor in it, many possibilities are opened. Take the example of a power supply feeding a magnet. The processor can take over some of the duties normally carried out by hardware logic, such as start-up sequencing, closed loop control, etc., and provide for ramping, hysteresis correction and self testing and diagnosis. It also means that communication with the equipment can be by means of exchanging messages, rather than the succession of coded commands and responses needed for a conventional interface. If these messages are in printable ASCII characters, then the equipment can be fully tested using a simple terminal. This is already happening with some of the more sophisticated test equipment incorporating micro-processors, which are usually interfaced to the GP-IB (IEEE488) standard, which was first produced by Hewlett-Packard. This standard for passing messages has restrictions on the number of equipments that can be connected and the length of the connections that tend to limit its use, as a general purpose interface, to the smaller control systems. One would like the properties of the type of LAN described above, but at a lower connection cost, for joining large numbers of "smart" equipments to a computer, to pass the messages. A system originally developed for aircraft control, the MIL-STD/1553B,[13] has been chosen for this purpose at LEP and TRISTAN.

One disadvantage of incorporating processors in the equipment is an increase in the response time for urgent actions, as the processor has to be interrupted from whatever routine task it was carrying out, which usually takes longer than the transmission of a few CAMAC type commands in a conventional system. However, one command to a processor can produce a whole sequence of actions that would require a very large number of CAMAC commands. Most control systems that have requirements for very fast response or exact timing use a separate timing signal distribution which can provide trigger pulses directly to the hardware.

Computers

Only a few years ago, a multi-computer system would usually consist of a small main-frame as central computer with mini-computers as satellites. Now we have 32-bit mini-computers many times more powerful than quite large main-frames of the past, and the satellites are becoming micro-computers. In some cases, powerful single-board micro-computers with a multi-tasking operating system are being used to carry out all the tasks of the satellite, as in the new control system at SLAC for the SLC:[3] in others, a number of micro-computers are used to share the tasks, using a simpler operating system, as at LBL for the Super HILAC[14] and other projects there, and the systems now being developed for LEP,[15] for HERA,[16] and by a Philips/NIKHEV Collaboration.

In both cases it is necessary to have a crate and bus system to interconnect the processor(s) and the modules providing the connections to the computer system and to the equipment, etc., and there are a number of different systems offered by various manufacturers for this purpose. The most popular at the moment is Intel's Multibus, but Motorola's VME is rapidly gaining popularity. Present control systems use 16 bit micro-processors, but 32 bit versions are becoming available and the bus systems are being upgraded to suit.[17] An attempt has been made by the IEEE P896 committee to define a manufacturer independent bus system, the "Futurebus", but progress is slow, and it seems unlikely to find sufficient support to be used widely.

Software

The design of control systems can be carried out in many different ways, and there are no generally agreed "best" ways of doing things. Nowhere is this more apparent than in the software. The ideas in this field can be divided into two opposing camps. On one hand we have those that maintain that software is the province of the software professionals; the users should specify exactly what is required and then the experts design the programs, code them, debug them and then hand them over to the user. Such a system can lead to well designed, well documented programs, especially if the aids to programming provided by the better modern systems are used, but the lead time for a new program can be long, modifications are resisted, and, unless a large number of programmers are available, the running-in of the machine can be painful when requirements are found that were not in the program specifications.

On the other hand we have those that maintain that programming the system should be made very easy so that the programs can be written by those who have to use them, the hardware specialists, the machine physicists and the operators.

The primary role of the software specialists is then to provide the tools; the system, the procedures and functions needed, etc. This approach requires the use of an interpretive system to allow the interactive development and debugging of programs in a relatively safe environment. Its advantages are that the programs are developed by the user, who can incorporate changes as he goes along, when he finds the original concept needs to be modified, with immediate feedback s to the consequences of the change. The disadvantages are that the programs may not be well designed, or make optimum use of the facilities, and are likely to be poorly documented, leading to difficulties when the authors have moved on to other work. In addition, the interpretive approach, where linkages between functions have to be made at the time of execution, runs slower than pre-compiled programs. This can be partly overcome if provision is made so that, once the algorithms have been worked out and tested in the interpretive fashion, large parts of the program can be compiled into modules than can be linked by the interpreter on-line. The interpretive types of system have been used at CERN, DESY, KEK, Rutherford, NIKHEV and JET, and was proposed for ISABELLE.

Another area of controversy is the language(s) to be used for programming the system. Assembly language is now normally only used for special time-critical applications, and the majority of systems use a high level language for most of the applications programs. FORTRAN, although it has some deficiencies for a real-time system, is very popular, because of its widespread use in scientific computing, but there are a number of others which have been used for accelerator control. These can be classified into two types: Those designed for real-time systems, such as RTL2 and CORAL, and those designed for structured programming, PASCAL and its derivatives. Most of the systems using the interpretive approach have used NODAL, developed at CERN,[18] as a basis, with the exception of NIKHEV, who developed their own language, and some small systems that have used BASIC.

Extra facilities, not normally available with the older languages, are required for programming a multi-computer system, such as those to synchronize programs running in different computers, and new languages are being developed for this purpose. The best known of these is ADA, designed to a specification from the American Department of Defense, for which compilers are now becoming available, and MODULA-2, which is being used for systems programming in LEP. An even newer language, OCCAM,[19] has been designed expressly for multi-computer process control. Whatever other language is used, it seems essential to make provision for running FORTRAN programs in the system, since the simulation and modelling programs will almost certainly be written in this language.

The programming of micro-processors in the lower level of control systems presents particular problems, since they are usually limited in their capabilities. The task can be made much easier if there is software available in the main computer(s) for the development and debugging of the programs for the micro-processors, with facilities for cross-compiling, emulation, etc. Such software is now available for a large number of hosts and targets.

Databases

Most of the earlier systems used a central data base containing the various parameters needed for the control of the accelerator by the applications programs. With the multi-computer systems, parts of this data base are normally moved to the lower level satellite computers, so that they can carry out some of their repetitive jobs without constant reference to the central data base. There are different philosophies about updating - in some cases the central data base is automatically updated from the satellites, and all programs use the central data base, while in others the data is obtained from the satellites only when required by a program.

In addition to the data needed to run the accelerator, there are many other data needed for the design, installation and maintenance of the machine, which in the past have been kept mainly in paper form. With large machines this data can be very extensive, and the trend is to keep this information on a commercial data base management system (DBMS). Such a system on a large computer can store an enormous amount of data but, because of all the searching that has to be done to find a given item, it would be too slow to be used for the operational data base of a control system. To overcome this difficulty, a program can be written to extract the control information from a DBMS and form it into a series of linked files which can then be loaded into the control computers for rapid access. One advantage of this linking of data bases is that if, for example, the operator receives a message that some component is not operating properly, he can obtain, directly at his console, information as to the location of the component, its maintenance record, what to do about it or who to call and how to get hold of him. Such a system is being implemented at LAMPF[20] and will be used for LEP, where the design and installation information are being recorded in the DBMS during construction.

Operator Interface

As beauty is reported to be only skin deep, so, to the user, the beauty of a control system is largely determined by the quality of the operator interface. However elegant the internal workings of the system, if it requires what the user considers to be unnatural actions on his part, it will not be accepted as successful. Unfortunately, people do not agree on what is natural!

We have come far from the early days when the natural way of interacting with a computer was the teletype, and now we have a whole range of devices. The most popular output device is still the cathode ray tube, on which character or graphics displays can be shown, now usually using TV raster techniques. Reductions in the price of memory have led to the widespread use of multicolour displays, refreshed from a local store, without involving the computer system.

Input from the operator is frequently by means of touch-sensitive screens over an array of "soft" button images for the selection of actions to be performed, supplemented by the positioning of a cursor on a display by means of tracker-ball, joystick or "mouse". Knobs can be assigned to variables that may be settings of a component of the accelerator or abstract parameters that require the coordinated setting of a number of components. Although a keyboard is usually provided, its use should not be required for normal operation of the machine.

The choice of devices and their arrangement on a control desk are a matter of personal taste, and the most successful systems have taken into account both ergonomics and psychology. An example of the latter is what I call comfort displays. Even with the most sophisticated systems, which can give the operator full information when anything goes wrong, he usually likes to have a display in a fixed position, permanently there

without having to push a button, showing some parameter of the machine that tells him that all is well.

A trend that is increasing, especially for large machines, is the provision of portable control consoles which can be plugged in at various positions, for local control, testing and maintenance. Depending on the architecture, these can act as terminals, calling for programs to be run in the central system and the results sent down, or they can have sufficient power to run programs themselves, enabling testing of parts of the accelerator when the central system is unavailable.

Conclusion

As can be seen from the above review, in the last few years we have moved from a situation where the dominant cost in a computer control system was that of the computers and their memory to one where it is that of the interface, cabling and software. This means that we should be lavish with computing power if we can make a saving in software. However, the demands on the software constantly increase as the accelerators become more and more sophisticated, and require to be modelled in the control system rather than "flown by the seat of the pants" as was often the case earlier.

References

1. J. M. Putnam *et al.*, "The Application of a Digital Computer to the Control and Monitoring of a Proton Linear Accelerator," IEEE Trans. Nucl. Sci. NS – 12, 3 (June 1965).
2. K. B. Mallory, "Initial Experience with a Multiprocessor Control System," Proc. 9th Int. Conf. on High Energy Accelerators, Stanford, May 1974.
3. R. E. Melen, "A New Generation Control System at SLAC," IEEE Trans. Nucl. Sci. NS – 28, 3 (June 1981).
4. J. T. Hyman, "The Computer Control for the SPS," Proc. 9th Int. Conf. on High Energy Accelerators, Stanford, May 1974.
5. W. Struven, "Wide-Band Cable Systems at SLAC," IEEE Trans. Nucl. Sci. NS – 30, 4 (August 1983).
6. R. Parker and S. F. Shapiro, "Untangling Local Area Networks," Computer Design, March 1983.
7. H. Ikeda *et al.*, "Design of the Control System of TRISTAN," IEEE Trans. Nucl. Sci. NS – 29, 3 (June 1981).
8. P. Bylanski and D. G. Ingram, "Digital Transmission Systems," Peter Penegrinus (London) 1980.
9. "A CAMAC Serial Highway System for Process Control of the VICKSI Accelerators," Proc. 2nd Int. Sym. CAMAC in Computer Applications, EVR 5485, March 1976.
10. M. C. Crowley-Milling, "Experience with the Control System for the SPS," CERN 78-09 (December 1978).
11. H. Feese and G. Hochweller, "The Serial Data Acquisition System at PETRA," IEEE Trans. Nucl. Sci. NS – 26, 3 (June 1979).
12. J. Bobbitt, "Applications of Microprocessors in Accelerators," IEEE Trans. Nucl. Sci. NS – 22, 3 (June 1975).
13. "Serial Digital Bus Heads for Industrial Systems", Electronic Design, Vol. 28, No. 19, September 13, 1980.
14. S. Maggary *et al.*, "Operating Experience with a New Accelerator Control System Based on Microprocessors," IEEE Trans. Nucl. Sci. NS – 28, 3 (June 1981).
15. J. Altaber *et al.*, "Replacing Mini-Computers by Multi-micro-processors for the LEP Control System," IEEE Trans. Nucl. Sci. NS – 30, 4 (August 1983).
16. G. Hochweller *et al.*, "PADAC Multi-Microcomputers; Basic Building Blocks for Future Accelerators," IEEE Trans. Nucl. Sci. NS – 28, 3 (June 1981).
17. "Multi-Processing 32-Bit Busses are Starting to Blossom," Electronics, March 22, 1984.
18. M. C. Crowley-Milling and G. Shering, "The NODAL System for SPS," CERN 78-07 (September 1978).
19. R. Taylor and P. Wilson, "Process Oriented Language Meets Demands of Distributed Processing," Electronics, November 1982.
20. S. K. Brown, "The Use of a Commercial Data Base Management System in the LAMPF Control System," IEEE Trans. Nucl. Sci. NS – 30, 4 (August 1983).