IEEE Transactions on Nuclear Science, Vol. NS-26, No. 3, June 1979

THE ADDITION OF LOCAL INTELLIGENCE TO THE ISR CONTROL SYSTEM

C.R.C.B. Parker

Abstract

The control system of the CERN Intersecting Storage Rings (ISR) has been gradually modified over the last few years to become almost entirely based on CAMAC, giving the computer independence and flexibility which that system provides. It has recently become desirable to add local intelligence to various parts of the system, i.e. a processing capability closely associated with the equipment being controlled or monitored. This local intelligence has been provided by processors which are able to run autonomously and are transparent: they simply give the main control center the illusion that the equipment is more elaborate than it really is. It has proved possible to incorporate two types of such processor quite elegantly into the CAMAC system even though CAMAC was not designed with these applications in mind. This paper describes the design philosophy of these two types of processor, and the areas of application in which they have so far been used. The development aids at all levels which are available are also described. Finally, extensions to the CAMAC specifications are suggested which would allow it to be used more easily for such projects.

1. Introduction

Until recently, the ISR computer control system was centrally organized with a single controlling computer (a Ferranti Argus 500). Equipment interfacing was done centrally (mostly by CAMAC[1]); several crates were provided in the central area, and cables were brought from the remote equipment to these crates.

2. The Evolution of the Control System

In about 1975, the control system became saturated. The single Argus computer was being over-utilized. It did not have sufficient disk file storage, memory, or computational capacity for all the tasks it was required to perform. Moreover, the number of tasks which could be run simultaneously was too low. The foreseen limit for the maximum number of CAMAC crates (7) had been reached, and these crates were virtually full. To have added more crates would have required considerable development effort.

It was becoming increasingly evident that the control system had to be re-structured to allow for expansion. A hierarchical system was adopted. CAMAC crates were installed in the remote buildings so that the interconnection length between the crates and the equipment would be short. These crates were interconnected using the CAMAC Serial Highway system and driven by parent computers which were added in the central area. The parent computers were dedicated to particular application areas (magnet power supply control, vacuum status monitoring) and were attached to a data network allowing communication between themselves and to the Argus.

It was at the time of this re-structuring that it became feasible, due to the availability of low-cost processing and memory chips, to incorporate intelligence at lower level of the control system than the top. The three levels at which intelligence could be incorporated are the equipment level, the CAMAC module level, and the CAMAC crate level. This is illustrated graphically in Fig. 1, with an indication of the "area of control" of devices at each level. Incorporation of local intelligence at any level reduces the dependence on higher levels, i.e. the data rate to higher levels is lowered and the system is more tolerant of failures at higher levels.

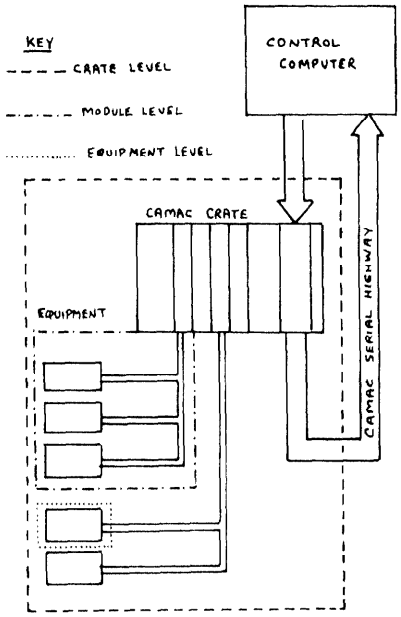

Fig. 1 Area of Control of Intelligent Devices

The present ISR control system contains examples of intelligence at all three levels, but this paper will not deal with the equipment level, since the way in which intelligence is added at this level, and the benefits of doing so, depend so much on the particular equipment. Nevertheless, it is the author's view that it is the equipment level that offers the most possibilities for the addition of local intelligence.

3. Local Intelligence at the Module Level

As part of the process of rationalizing and expanding the control system, it was necessary to interface much old instrumentation to CAMAC. It was desired to retain as much as possible of this old instrumentation and its cabling; this meant that the peculiarities of each instrument would have to be "mapped" onto the CAMAC command structure.

Rather than develop special modules for each instrument, a general-purpose module, known as the Master Sequencing Unit, was designed and constructed. A block diagram of this unit is shown in Fig. 2. It contains a CAMAC interface, a Signetics 2650 micro-processor, 1kbyte of Random Access Memory, and up to 2kbytes of PROM. In effect, the module simply provides translation between the CAMAC Dataway and the micro-computer bus. The interface to the CAMAC Dataway is via two registers, an 8-bit operation register and a 16-bit data register. This allows complete programmability of the CAMAC functions which can be accepted by the module. A connector on the front panel of the module carries a buffered version of the micro-computer bus; this provides communication with a second module containing circuits specific to the particular application. The intention is that most of the complexity of the interface (timing, synchronization, sequencing, simple data manipulations, etc.) are handled by the micro-computer program, so the second module is trivial.

There is provision in the module for the CAMAC Look-At-Me (to the parent computer) to be set by pro-

ISR Division, CERN, CH-1211 Geneva- 23

0018-9499/79/0600-3392$00.75 © 1979 IEEE

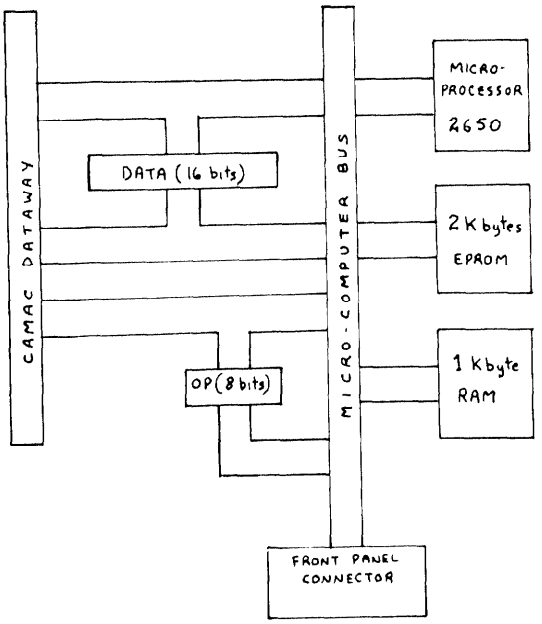

Fig. 2 Block Diagram of MSU

gram, and two hard-wired functions are implemented for testing and clearing this LAM. A third hard-wired CAMAC function causes a master reset. All other CAMAC functions must be interpreted by the micro-computer program. Provided the micro-computer is ready (indicated by a programmable status bit), then the first CAMAC "write" or "control" function to the module can be accepted (acceptance is signalled to the CAMAC system by giving the Q = 1 response; this technique precludes the use of Q for any other purpose). Acceptance of further commands is then inhibited whilst interpretation of the current command proceeds. At an appropriate time, the micro-computer indicates its readiness to accept another CAMAC command by setting the status bit once more. This technique overcomes the difficulties caused by the disparity between micro-computer instruction times and CAMAC data rates.

CAMAC "read" commands are dealt with in a slightly different way. The micro-computer sets up in advance the code for the next expected command and the appropriate read data, and then sets the "ready" bit. The Q = 1 response is only given if the micro-computer was ready AND the actual command was the one it was expecting. In any event, the micro-computer is informed which CAMAC command actually took place.

The module has three limitations, which are not serious in practical situations. Firstly, the interpretive process is necessarily slow - typically 50 µs per CAMAC instruction - although the host computer's operating system often imposes greater overheads than this time. Secondly, the efficient operation of the system depends on the CAMAC commands following a predefined sequence, which again is not a significant drawback in practice, as such a sequence will generally be defined in the design stage of any interface system. Thirdly, the "Q" response cannot be used for status indication, but this is not serious because "read status" commands can easily be implemented.

The advantages of the module are due to its programmability. A large number of CAMAC commands can be implemented, so the functions obeyed by the module can be defined logically rather than being limited by the idiosyncrasies of the hardware. Debugging of a system is very simple since small sections of the hardware may be tested independently, using special test programs. This also means that testing can be more thorough. Finally, modification of the system is much simpler than for purpose-built units.

Nearly 30 Master Sequency Units have been installed at the ISR for four different projects. A typical example is its application to an analog data acquisition system monitoring many inputs at low speed. The MSU is connected to an adaption unit (via the front panel micro-computer bus) which drives a dual channel 160 way relay scanner and two digital voltmeters. It is programmed to accept CAMAC requests for blocks of up to 160 (double) readings starting at a particular address. The readings are then taken, converted from BCD to two's complement form, and the controlling computer interrupted. The results can then be read out as a block.

In general then, intelligence has been added at the module level to deal with the peculiarities of particular instrumentation, presenting the user (i.e. the control center) with a more straightforward, rational interface. The programs at this level are closely related to the hardware; once written they change rarely - the requirement is well-defined and static.

4. Local Intelligence at the Crate Level

Intelligent devices at this level, able to control the whole of the instrumentation attached to a crate, are much closer in function to the main control computers. They effectively run applications programs, and since the requirements change frequently, so do the programs. They can benefit from the same facilities that the main control computers use for flexibility and convenience; multi-programming, multi-processing, real-time operating systems, and even comprehensive filing systems all have their place here.

To satisfy an ever-growing requirement for intelligent processors at this level, the ISR have developed an Auxiliary Crate Controller for CAMAC, known as the ACC100. This is a single-board controller capable of generating full 24-bit CAMAC commands. It contains a Ferranti F100-L 16-bit microprocessor, together with a Signetics microsequencer, a fast arithmetic-logical unit, and a scratchpad register file. A block diagram of the ACC100 is given in Fig. 3. It uses the CAMAC Dataway for all access to its associated memory. To

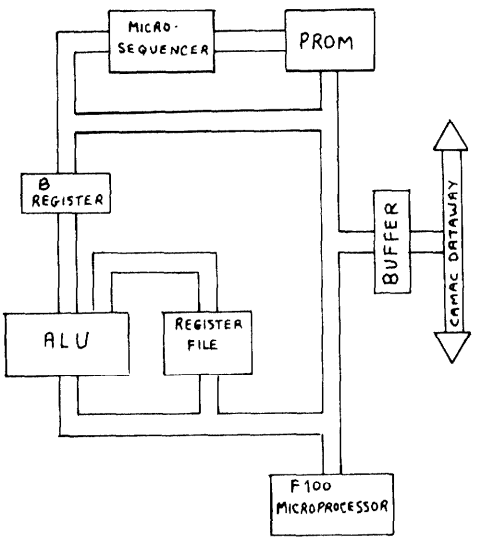

Fig. 3 Block Diagram of ACC100

make more efficient use of the Dataway for this application, there is provision for non-standard cycles with handshake timing and in which the Read and Write lines are used simultaneously for memory address and memory data. Special function codes are used to distinguish between standard and non-standard cycles. Using this method, a wide variety of memory modules may be accommodated, either as standard CAMAC modules or as special modules.

The ACC100 can be used in conjunction with the type L2 serial crate controller, or with controllers using the Auxiliary Controller Bus recently defined by NIM and ESONE[2]. It can also be used in stand-alone mode. Hardware bus arbitration is provided which allows several ACC100's to share a CAMAC dataway and thus the memory in the crate.

The microsequencer "snoops" on all memory accesses from the F100, adds an offset (datum) to all memory addresses and checks whether such addresses exceed a preset limit. This arrangement makes the job of loading several tasks into a single memory space easier, and reduces the possibility of accidental interference between tasks. The microsequencer also traps a certain range of instruction op-codes which are not used by the F100 chip and either interprets them directly (as in the case of a user request for a CAMAC operation) or calls the executive kernel to handle them.

The executive kernel deals with the scheduling of tasks and with the communication between them. It arranges that the pool of available ACC100's will run the tasks required; there is not necessarily a direct relation between the number of processors and the number of tasks. It deals with the suspension of active tasks and the activation of suspended tasks.

Overall control of the crate is provided, for non-stand-alone systems, by the host supervisor. This runs on the parent machine and interacts with the local processors in the crate by means of its shared access (via the CAMAC serial highway) with the memory. Thus, it communicates with the executive kernel through various tables in defined areas of memory. It also deals with the loading of tasks and keeps track of store allocation.

The ACC100 has been used already in several projects at the ISR. These include driving a local panel controlling a FFT analyser and frequency synthesizer, and monitoring a terminal switching exchange. Later this year, several crates containing ACC100's will be installed to act as nodes in a high-speed data network.

5. Development Aids

The programming languages used at the ISR are almost exclusively high-level. The block-structured language CORAL-66 has been used on the Argus computers for almost ten years. The same language was adopted for the Nord-10 computers and more recently for the ACC100 projects. It has been found that debugging aids are not often required when programs are written in high-level languages. For example, although we have a simulator for the ACC100, it has never yet been used.

When the Master Sequencing Unit project was started, no high-level language compiler was available for the particular micro-computer chosen, and so the manufacturer's development system was purchased. This proved to be especially useful in the first phases of the project, when "in-circuit emulation" was used to check out the hardware. However, a high-level language is now available and will be installed on one of our local mini-computers; it is expected that the use of the development system will then be much reduced.

In general, therefore, it has been found that low-level development aids are not necessary when high-level languages are used.

6. Possible Improvements

Most of the difficulties encountered during the addition of local intelligence at both levels was due to inflexibility in the interfacing standard (CAMAC). This can be overcome in two ways - by adopting a different, more flexible standard or by modifying the existing one. The latter approach is the only feasible one for those organizations with a heavy investment in the existing standard.

In an attempt to resolve these problems for CAMAC in a defined way, ESONE have set up a study group to suggest extensions to the CAMAC specifications which would allow a more flexible timing and an extended addressing capability, while retaining complete compatibility with existing equipment.

References

1. CAMAC - A modular instrumentation system for data handling, NIM Specification TID-25875, ESONE Specification EUR-4100e, IEC Recommendations 482, 516.

2. Multiple controllers in a CAMAC crate, NIM Specification DOE/E-0007, ESONE Specification EUR-6500e.