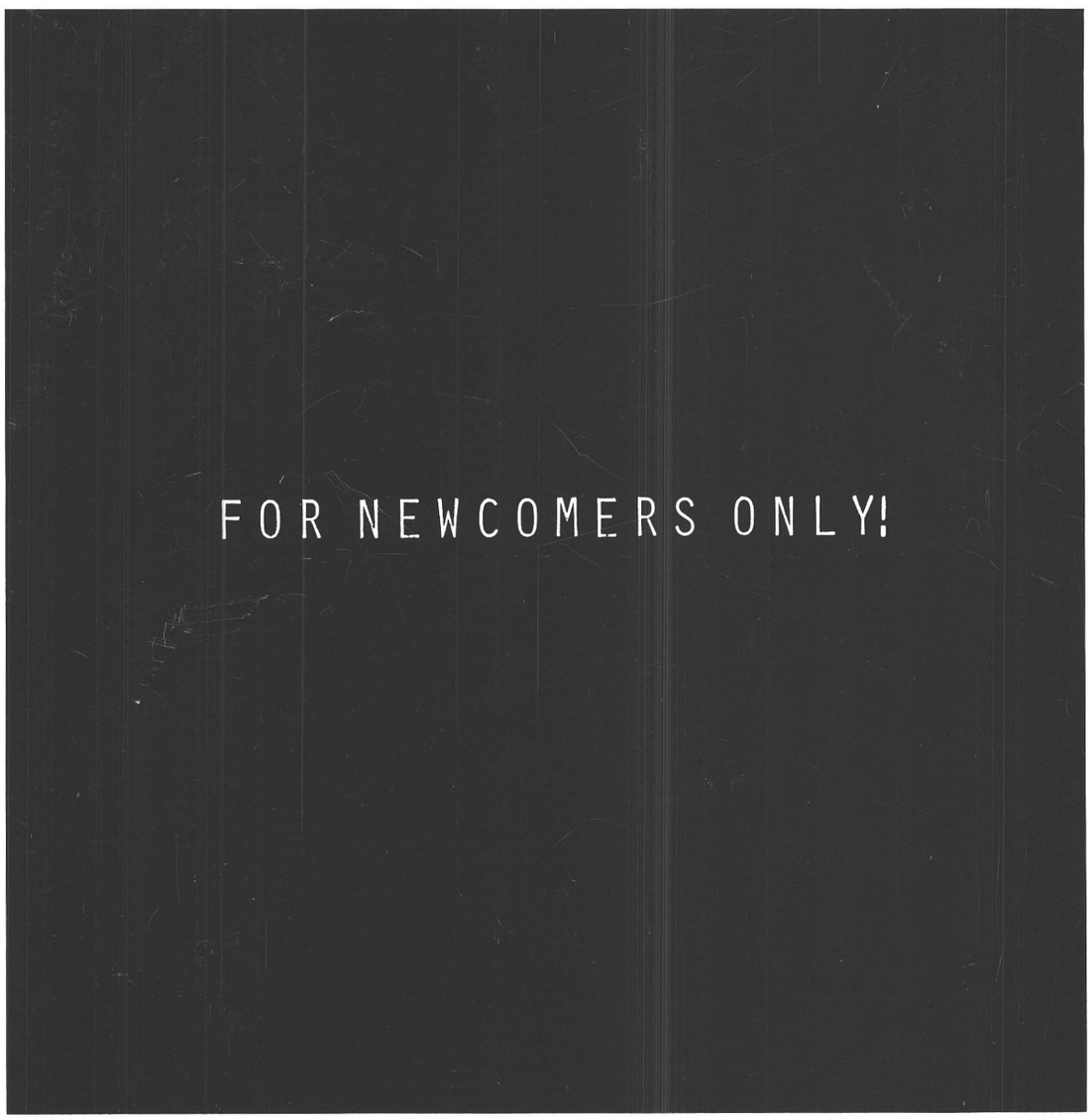
CAMAC bulletin

A publication
of the
ESONE Committee

ISSUE No. 13
September 1975
Supplement A



Distributed by :
Commission des Communautés Européennes
29, rue Aldringen
Luxembourg

C A M A C

FOR

NEWCOMERS

BY

HANS-JOACHIM STUCKENBERG
DEUTSCHES ELEKTRONEN-SYNCHROTRON DESY, HAMBURG

MARCH 1975

Acknowledgement

The author is pleased to acknowledge his indebtedness to his colleagues

R.C.M. Barnes, AERE Harwell, U.K.
I. N. Hooton, AERE Harwell, U.K.
H. Meyer, BCMN Geel, Belgium
K. Zander, HMI Berlin, Germany
H. Klessmann, HMI Berlin, Germany
B. A. Brandt, University Marburg, Germany

for useful discussions.

The author expresses his particular appreciation to Mrs. G. Andersson for typing the manuscript.

Quod capite, tot sensus.
Horaz

CONTENT

1. CAMAC, what's that?

1.1 Introduction

CAMAC is the description of the hardware and software in a system for exchanging data and control information between a computer and its on-line data inputs and outputs. This system, formerly designed by nuclear research centers to transfer the high data rates produced in nuclear physics experiments, has precisely fixed parameters where these are needed, but gives a great deal of freedom where this can be tolerated, in order to make the user very happy in solving his problems. So, CAMAC becomes very attractive for applications in non-nuclear fields, e.g. in

- Industrial Process Control,
- Medicine and Health Service,
- Public Utilities,
- Traffic Control,
- Data Communication and
- Laboratory Automation.

The basic idea of such a system, which is known as the CAMAC Interface, is due to the fact that, in the long term, it should be more economical to use systems which are internationally standardized and produced by many firms than to follow the special local standard of one manufacturer.

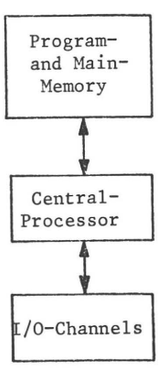
Today CAMAC is used in more than 20 countries all over the world, and it is manufactured by more than 60 firms.

The aim of this paper is to show newcomers the way to use CAMAC in their application, with the aid of examples from the process control field.

1.2 What about the Connection between the Process and the Computer?

Process computers act as the central part in process control systems. In typical applications their job ranges from data acquisition with off-line computing of the parameters, to on-line realtime control in industrial process automation.

The architecture of process computers is similar to those of other computers (Fig.1.1). They consist of:



1.1

- the central processing unit (CPU), whose main parts are the arithmetic and logic unit (ALU), the instruction decoder, the program counter, the general registers, timing and interrupt handling facilities,
- the memory, which stores the data and the program,
- the I/O-Ports, which handle the incoming and outgoing instructions and data and act as the coupling between the computer and the interface.

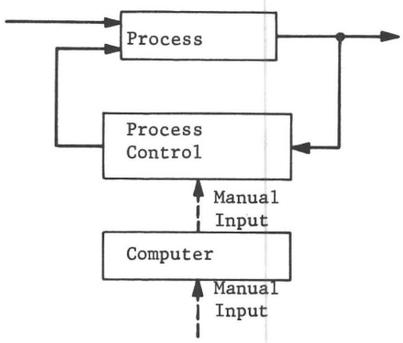
Process computers are often designed in a modular form, so that their configuration is flexible and can change to suit the application.

Process control means that relevant process parameters are measured and compared with set points and, by a program, they are converted to time-variable functions which control the process via the final control elements.

There are three typical operation modes:

- off-line,
- on-line, open loop,
- on-line, closed loop.

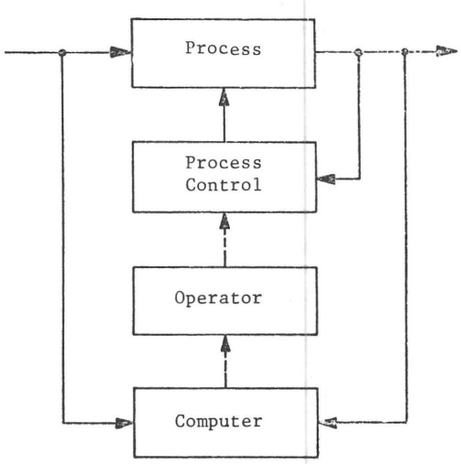
In the off-line mode neither the incoming nor the outcoming process data are connected directly to the computer (Fig.1.2). The process



1.2

parameters are read by the operators and stored in the computer, which handles the data. The results are use to control the process manually.

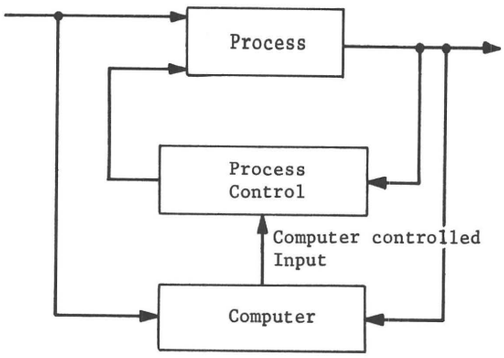
In the on-line, open-loop mode, the computer accesses the process parameters directly. After the parameters are manipulated, the process is controlled manually. This is the first phase of automation (Fig.1.3).



1.3

In the on-line, closed-loop mode, the computer is directly connected to the input and the output of the process. The automatic cycle of data acquisition, data handling and process control is under full control of the computer (Fig.1.4).

Data and control and status information of the process are stored in registers of the controlling devices. Seen from the computer



1.4

they are peripheral devices or I/O-devices. The interface, which connects the computer with the I/O-devices, is normally a bidirectional bus-system, i.e. a system of parallel wires, on which addresses, data and control information are transferred between the computer and the I/O-devices. The interface acts as a highway for information. The register is addressed via the address lines, during timing strobe signals the data is gated to or from the data lines, and the control information follows on separate lines.

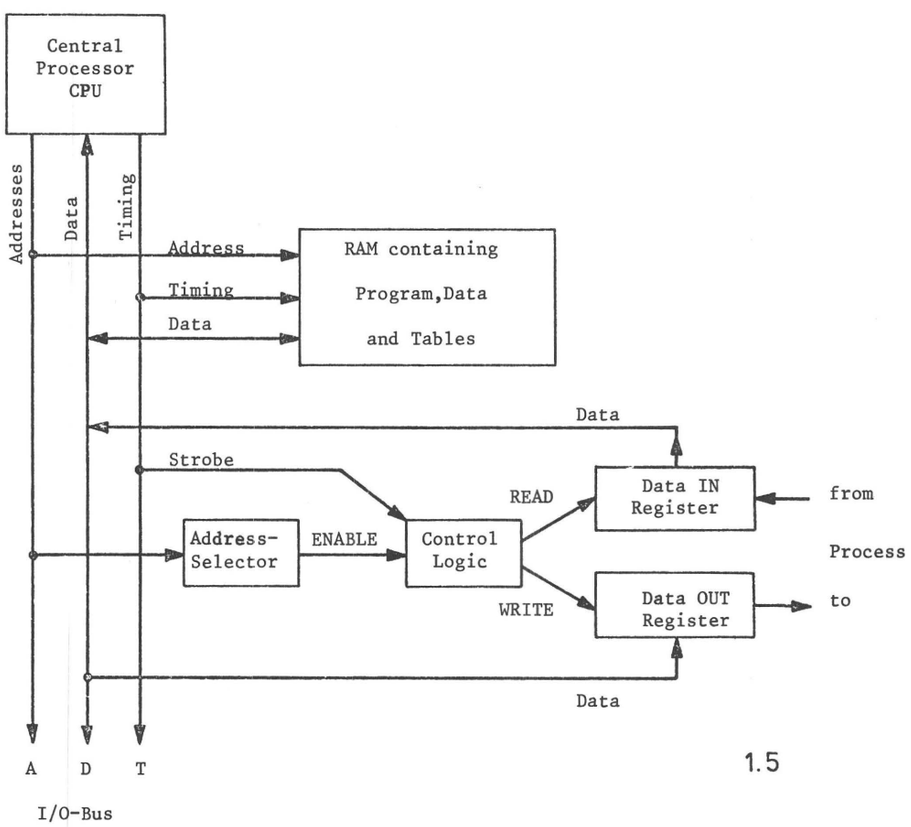
Modern interfaces are typically bus-highways between the I/O-ports of the computer and the registers of the I/O-devices. From this definition we can see that it is not very difficult to standardize interfaces.

1.3 What does a Typical I/O-Bus System Look Like?

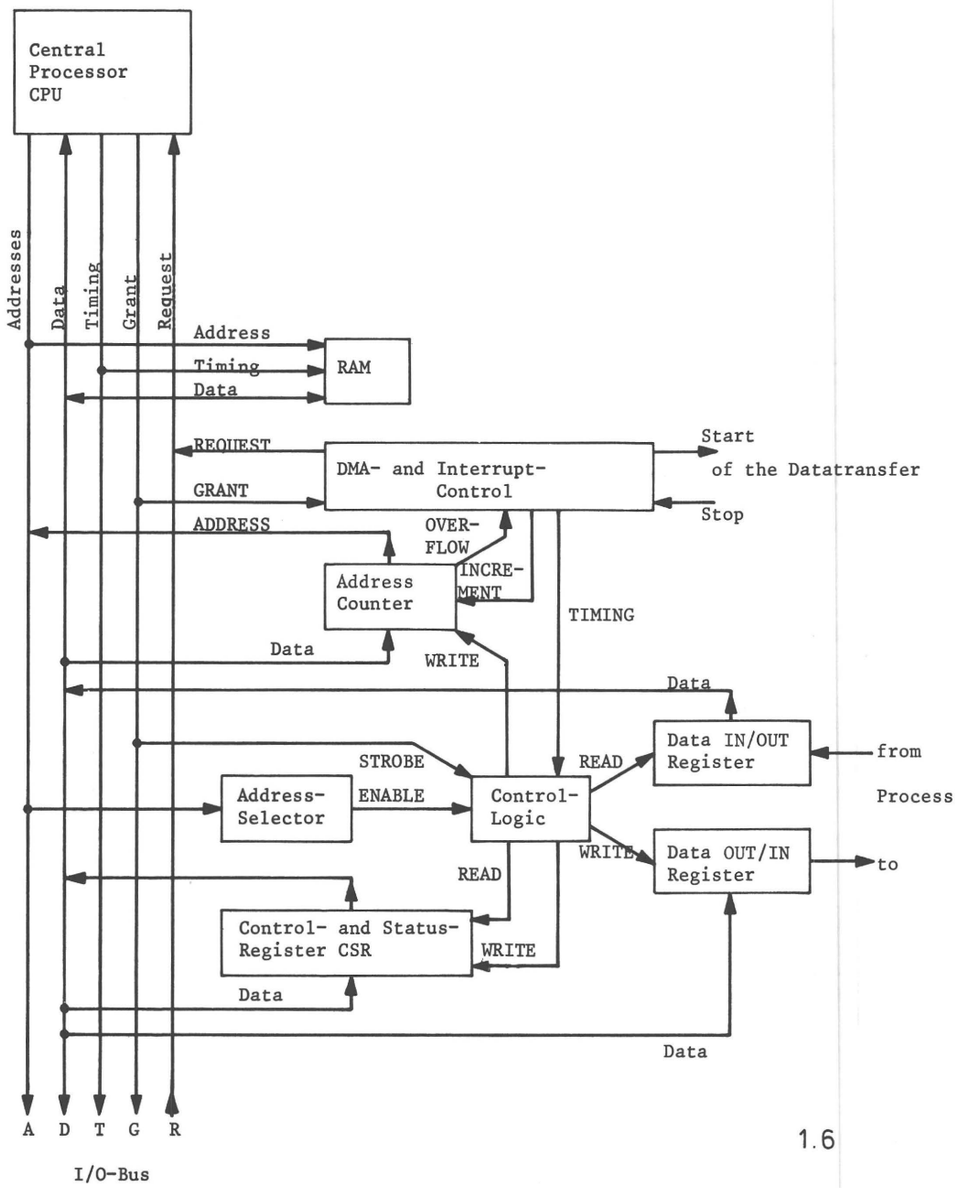
The function of the I/O-bus system is to transfer the data and control information between the computer and its I/O-devices. This can be done in two different modes, which are called:
- programmed transfer and
- direct memory access (DMA-transfer).

The programmed transfer mode is normally a word-by-word operation mode, in which the processor controls the transfer completely. In the DMA-mode, which has typically the highest priority on the bus, the processor releases bus control and then, when the bus is free and there is no other priority restriction, a block transfer of data between I/O-registers and the main memory can start. When the transfer stops, the I/O-device can request a processor interrupt in order to start a subroutine which will handle the data block.

The bus system, which is needed for programmed transfers, contains wires for addresses, data, and control and status signals (Fig.1.5). The processor sends an instruction word, the first part of which addresses the register, and the second part tells the register what it is to do. The function (READ or WRITE) is performed during the timing strobe signal. The interface must have some additional circuits to handle also DMA-transfers, e.g. control logic which sends a request to the processor to release the bus after the current cycle (Fig.1.6).



1.5



1.6

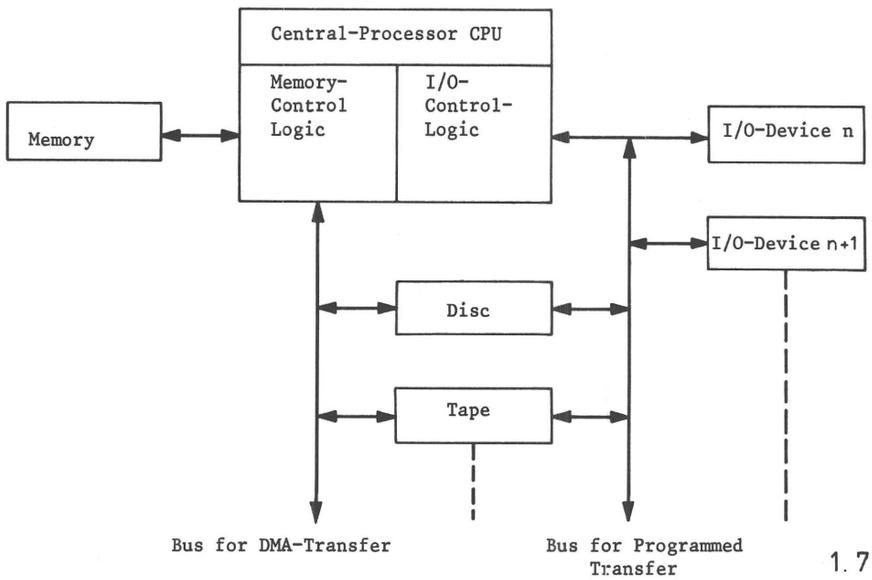
The processor checks the priority of the request and, if it is okay, sends a grant signal. The program counter of the I/O-device holds the address in memory, at which the transfer starts. After each word is transferred, the program counter is incremented to have the right address for the next word. After transferring the whole block, the control logic will set in its control and status register an interrupt bit with a vector-address to show that an interrupt is requested.

This complicated procedure demonstrates the amount of circuitry needed for the DMA-mode. You can see that it is very expensive, to provide this control logic in each I/O-device which may need to handle DMA-transfers.

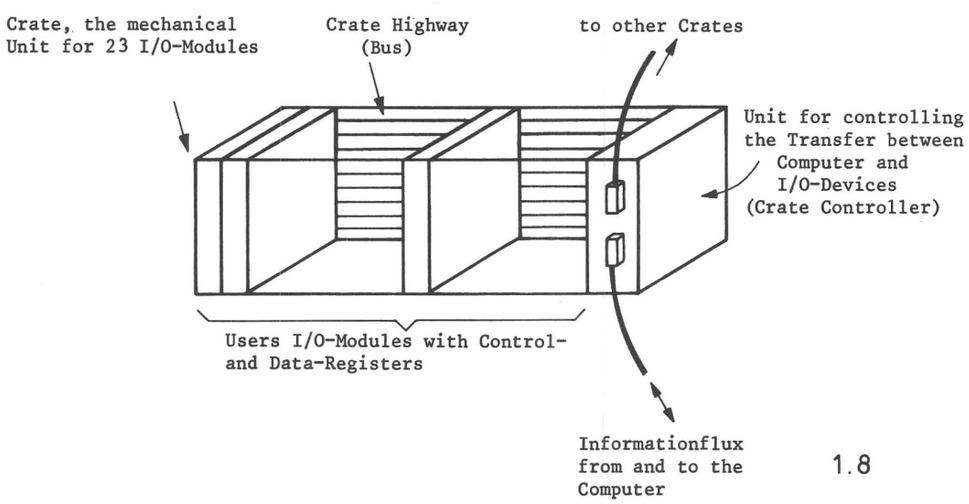
Some designers of computer hardware divide the I/O-bus system into two parts with respect to the transfer mode. So they need two different control units, which does not simplify the operation (Fig.1.7).

A very general solution of this problem, which is transparent, logical and economical, combines the I/O-devices and their registers into groups, housed mechanically in 19"-crates, together with a controller, which is a dialog-station between the I/O-modules and the computer, and handles programmed transfer, DMA-transfer and the interrupt organisation for its I/O-modules, Fig.1.8 shows this. This means that the processor looking into the interface via the I/O-ports, sees only the controller with its dedicated registers. A standardized bus system can be connected between the processor and the controller. The bus can be designed as a parallel or serial highway for the data and control information. A second bus system, which is of fixed length, is connected between the controller and its I/O-modules in the crate.

This system, which has standardized highways, specified hardware controllers and a de-



1.7



1.8

fined language to describe the hardware operations is called CAMAC, which means 'computer aided measurements and control'. It is an economical and logically transparent solution for many different applications in various fields. In the following sections we will demonstrate the basic features together with some typical applications. Maybe you will get some ideas for solving your own problems with CAMAC.

1.4 Do you Prefer a Parallel or a Serial Bus between Computer and Controller?

A crate is a mechanical system of fixed dimensions, so it is a suitable solution to wire the crate highway between the I/O-modules and the crate controller in a parallel form.

But it is another question, whether the highway between the computer and the crate controllers should be a parallel or a serial bus. The serial bus is much cheaper, but the transfer rate may be slower and one has to decide whether this type of interface will fulfil the desired conditions of the special application. We can discuss the pros and cons of both solutions from the user's point of view.

In parallel bus systems the transfer rate of information is very high, e.g. the CAMAC Branch Highway can transport about 1/2 million 24-bit words per second. Such high rates are typical in high-energy physics experiments with pulsed accelerators where, within the time of a particle beam, hundreds of data words coming from the nuclear detectors must be stored. All control and timing signals are immediately and simultaneously available in parallel systems.

The strongest disadvantages of parallel systems are the very expensive cables and multipole connectors and the relatively high failure rate of components which are connected in parallel. If balanced signals are not used, a maximum of up to 50 meters of bus-length may be economical, but, if you want to use CAMAC in industrial plants, or in hospitals where a central computer is on one floor and the CAMAC-peripherals are on another, then without any doubt the use of parallel highways will become highly uneconomical.

But CAMAC is ideally suited for such applications because, especially in hospitals, there are no technical staff well-trained in computer problems. Every home-standard system, delivered by a special manufacturer, will involve additional training for the technicians. It would be much better if they were involved with only one system, which is an international one, where the rules are fixed for a long time.

But fortunately in this application the data rates are not very high, so you can see the point of the CAMAC serial highway which can handle distances of kilometers.

In medical or industrial fields there is a need for very safe transfers of information, so one must make parity checks of various kinds. The messages which are used for communication in the CAMAC serial highway are checked for longitudinal and transverse parity.

Not only the data rates in the CAMAC serial system are slower, but also the response time of an addressed register is longer because the CAMAC serial highway is a loop, so all information, commands, replies and also demands, travel along the bus in the same direction, going out from the output port of the serial driver, through the loop and the addressed crate and coming in at the driver's input port.

All these features:
- an economical solution,
- a loop system with distances between the crate of kilometers,
- bit or byte serial transfer mode,
- two-dimensional parity check in the messages

make the serial highway very attractive in many typical applications outside the nuclear field where the CAMAC system was born.

1.5 The CAMAC Highways

In the previous chapter we have learned that CAMAC is specified with three different bus systems, two of them are parallel types, the third is a serial one. We will now describe these three systems, which we call the CAMAC highways, in a very short form. A detailed specification can be found in (H1,H2,H3). Some useful descriptions of additional CAMAC-features are explained in (H7,H8,H9,H10,H11).

1.5.1 The Parallel Crate Highway

The crate highway is a parallel bus system which connects the I/O-modules with the crate controller, which is the dialog station between the modules and the computer.

Each instruction set up by the computer contains an address and operand field: The address field points to the register in which the function, coded in the operand, will be executed.

The address field consists of two parts, the station number of the module and the subaddress of the register within the module.

If an I/O-module is to execute a CAMAC instruction, the computer must first call the crate controller, then specify the number of the module in the crate, and the subaddress of the register in which a coded function is to be performed. So the CAMAC instruction passed from the controller to the addressed register includes:
- the station number N, together with the subaddress A, in which module and register are specified to execute the
- function F.

N is decoded in the controller, A and F in the module.

Up to 25 stations are provided in the crate, up to 23 of which are used for the I/O-modules, and 2 for the controller. The stations are addressed via individual lines from the controller. Up to 16 registers are assigned for use in a module, so the subaddress is coded into 4 bits. The address field of the CAMAC instruction on the crate highway is then 5 bits wide (N = 1, A = 4), the operand field also 5 bits, because up to 32 different functions (dataless control functions and various read/write functions) are used in the CAMAC system.

Two timing signals, which control the execution sequence on the crate highway, are produced by a clock generator in the controller. The first is a pulse which strobes the data from the highway into the modules, the second performs possible clear operations within the register. All modules are able to respond to a 1 µsec dataway cycle.

The read and write lines are unidirectional, so that there are 24 read-lines and 24 write-lines in the crate highway, also called the dataway.

Four additional lines are necessary to transfer important status information:
- the Look at me (LAM) signal,
- the Busy signal B, which accompanies each dataway cycle,
- the Command-Accepted Signal X and
- the Response signal Q.

The LAM-signal, which is a demand-signal, is produced by those modules which need computer service. The LAM, which can be enabled or disabled by hardware or software, can be graded in the controller, i.e. selected, rearranged or combined, in conjunction with other internal or external demands. After that, the demand is transmitted to the computer, which starts to analyze the demand pattern so that it can handle the demand by the desired service routine.

If a module gets a valid CAMAC command, it must set an indicator signal, called X. Modules which can't execute the required CAMAC function, or those which are dead or have power failures, can't give a positive X-signal, so the operation stops.

The Q-response is a most important signal in block transfers, if it is true the block transfer can continue, if it is false the block transfer is interrupted (see also Chapter 3.2.2.2). In response to test operations the Q-signal indicates the status of internal features of the modules, e.g. the demand status.

1.5.2 The Parallel Branch Highway

Up to 7 crates can be addressed by the computer via a bus system. They are connected as a chain, rather than in a star configuration. Each crate controller is equipped with 2 internally linked multipole connectors on its front panel, called the highway ports. The bus lines come in from the computer or the previous crate controller, and they go out to the next

controller. Each controller receives the information dedicated to its crate and transmits information between the addressed I/O-registers and the processor or memory.

The branch is a bus connection for 65 signals and their individual return lines, with defined signal conventions and contact allocations at connectors. The lines are terminated at the end of the branch and also in the branch driver, which connects the branch highway with the I/O-channel of the computer used in this application.

The basic mode of branch operation is the command mode, in which the computer sets a CAMAC instruction onto the bus and the addressed I/O-module executes the desired function. The command can start a single or a multi-crate operation.

Therefore a CAMAC command on the branch consists of signals on one or more crate address lines, a bit pattern on 5 N-lines (station numbers), and other patterns on the 4 A and 5 F lines to specify the subaddress of the register and the function. Fig. 1.9 shows the CAMAC instruction word. The data lines in the branch for read or write operations are bidirectional. Fig. 1.10 shows the configuration of the buslines coming from the computer via its I/O-bus, the branch driver, the branch highway and crate controller to the I/O-modules. Equivalent signals in the branch are distinguished from corresponding lines in the crate highway by the prefix B.

The timing of branch operations is controlled by 2 interlocking signals, because of the different physical length of the branch in different applications. Operations are handled in four phases:

Phase 1: The branch driver sets the address, control lines, data in write operations, and the first timing signal BTA.
Phase 2: The addressed controller accepts the command and generates the crate dataway cycle together with the second timing signal BTB.
Phase 3: The branch driver accepts the status bit X and Q, and the data in read operations, and clears the BTA signal.
Phase 4: After finishing the crate dataway cycle the controller clears the BTB signal and the branch driver removes the command (and write data).

Fig. 1.11 shows the timing sequence of the 'handshake'-mode.

Demand operations are initiated by the LAM-signal from an I/O-module. The crate controller generates a branch demand signal (BD). The branch driver then responds to BD by sending a graded L Request. Each controller answers by transmitting a word containing up to 24 graded -L bits via the data lines. This word is an arrangement of the LAM-signals in the crate. The branch driver may generate interrupts in response to some graded-L bits or DMA requests in response to other bits.

During each CAMAC operation the I/O-module sets the X and Q signals and these are received as BX and BQ signals by the branch driver.

There is also a common signal, Branch-Initialize (BZ), which is transmitted initially through the branch and sets all flip-flops and registers to the start position.

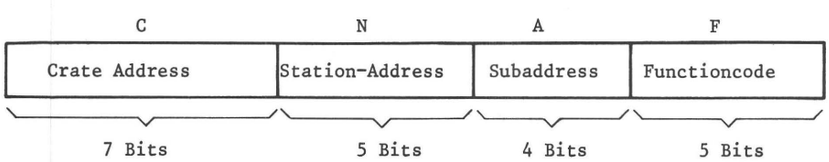
1.5.3 What is the Job of the Crate Controller in the Branch Highway?

The crate controller serves as a dialog station between the branch driver and the I/O-modules. The controller accepts the CAMAC instruction from the driver, waits for the timing signal BTA and then generates the crate dataway cycle. The CAMAC operation is synchronized by the signals Busy (B), Strobe S1, Timing BTB and Strobe S2 (see also Fig.1.12).

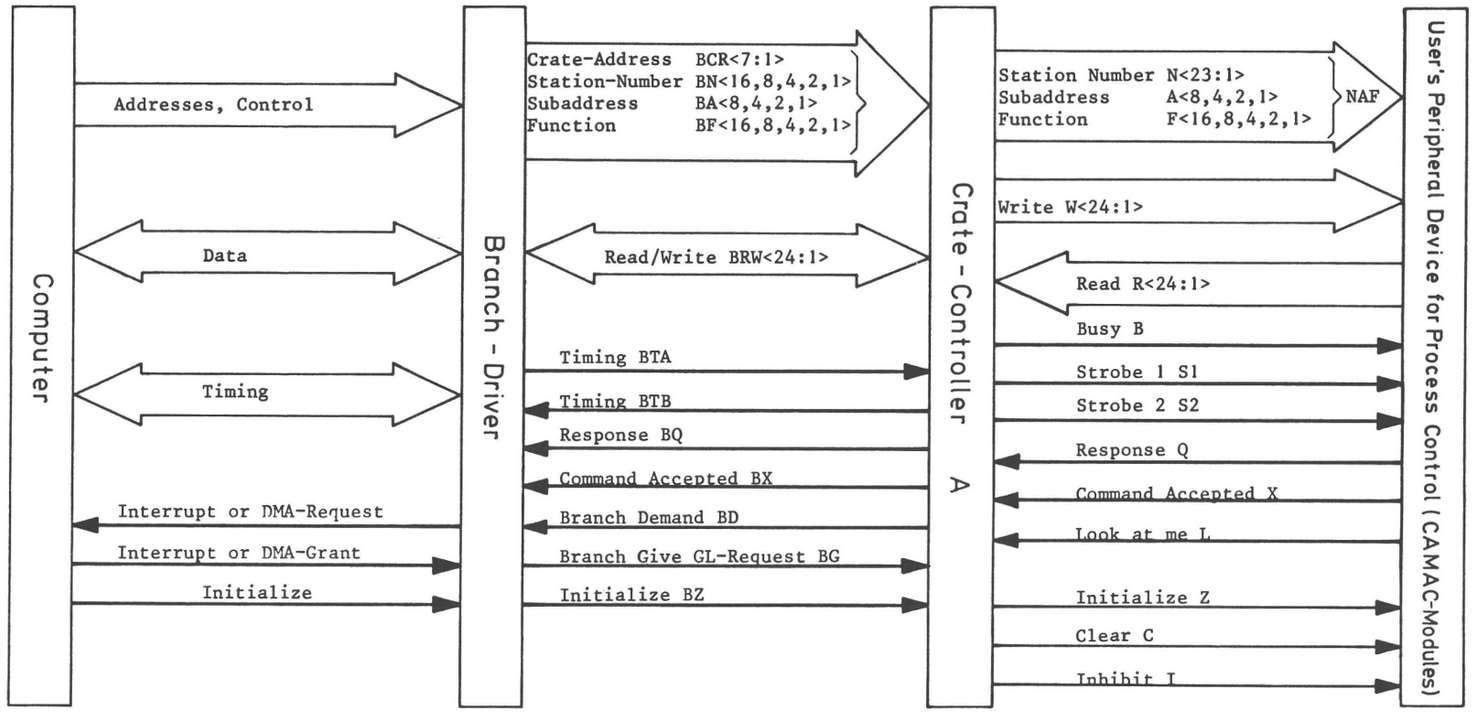
Some commands are implemented within the crate controller, e.g. to enable or disable branch demand signals. Therefore there are some flip-flops or registers in the controller, and these can be addressed by the redundant station numbers N24 to N31, together with subaddresses and function codes.

If an I/O-module needs processor service, it will set its LAM-signal. Each module can have more than one LAM-source, but certainly there must be an upper limit so that the system cannot be hung up by too many interrupts. Hence each crate should have a LAM-grader. This is often a special module in which the individual LAM-signals are combined with other internal or external LAM-sources by hardware wiring and to form a demand signal. The controller sends BD, the branch driver responds with the Branch-Give Alarm Pattern (BG) signal to get the graded-L word from the controller.

The controller which has these features is called a Type A Controller and is used as a standard controller in branch highway systems. But the branch highway is not the best and most economical solution in all applications, where a parallel bus system is needed, e.g., if only one crate is used in the system.



1.9

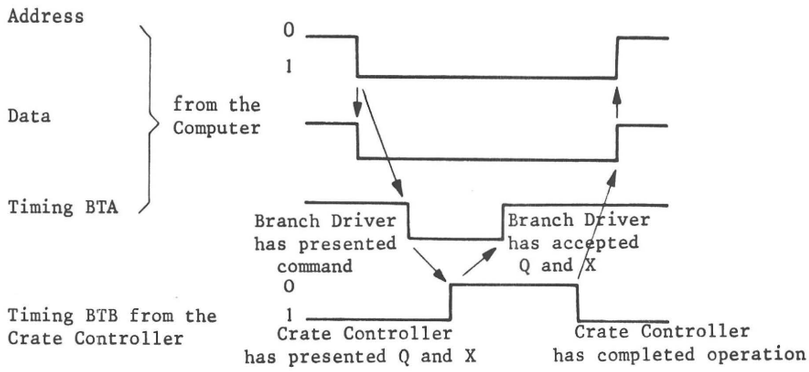


1.10

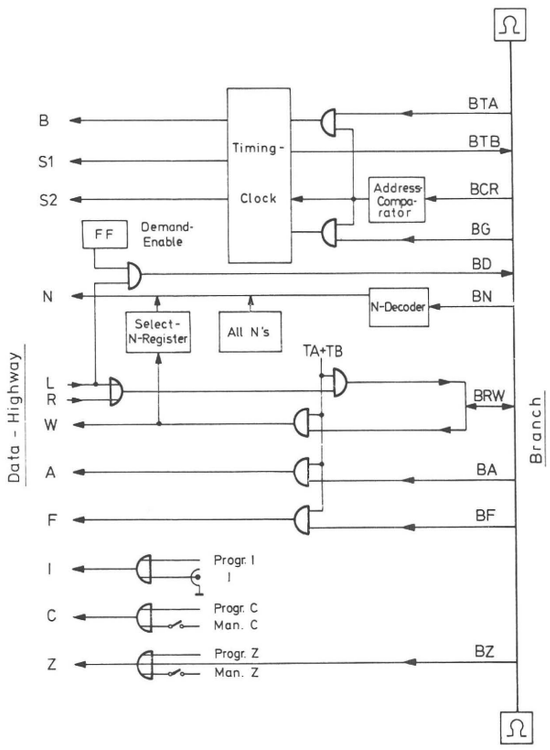
Computer - I/O - Bus

CAMAC - Branch - Highway

CAMAC - Dataway



1.11



Crate-Controller-Block-Diagram

1.12

For this reason many manufacturers offer special crate controllers, which translate directly from the I/O-bus of the computer to the crate highway. They are called dedicated crate controllers 'Type U' and are available for several computers, e.g.

- PDP-8, PDP-11,
- NOVA, SUPERNOVA,
- HP 2100, HP 2114, HP 2115, HP 2116,
- IBM/1130,
- H112, H316/DDP 516,
- SIGMA 2, SIGMA 5, C90-40,
- Siemens S320, S330, 404.

A dedicated controller normally contains a control and status register, data registers for the high- and low order bytes (if the computer has a word length of less than 24 bits) and also registers for the individual LAM's.

All controllers generate crate common signals like clear, inhibit and initialize, although the latter is normally set up by the computer rather than by the controller.

The detailed specification for Crate Controllers Type A, all of which are fully interchangeable, are presented together with branch highway rules in (H2). The operating conditions of the dedicated controllers are available from the manufacturers.

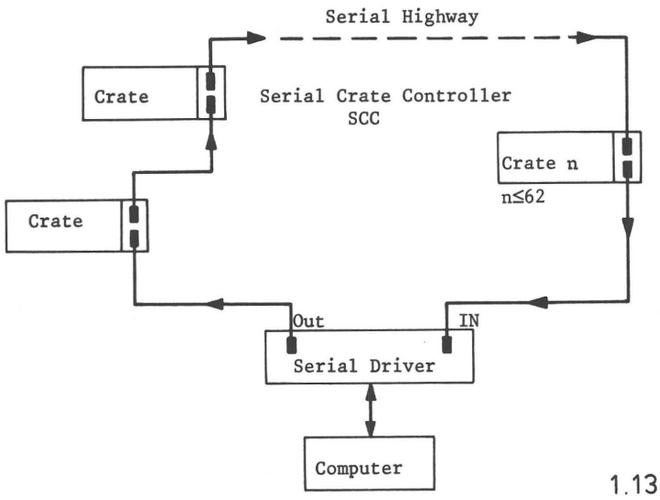
1.5.4 The Serial Highway

In contrast to the branch highway, the seri-

al highway is formed as a unidirectional closed loop including the I/O-port of the serial driver (SD) which connects the highway to the computer. Along the highway up to 62 CAMAC crates are placed, each of which contains a serial crate controller connected to the highway through I/O-ports on its front panel. Messages coming into the I-port are checked to see whether they are assigned to the crate, otherwise they are sent out through the O-port. Fig. 1.13 shows the closed loop configuration. The information flow is organized in a byte or bit serial mode, so that there are 8 or 1 signal lines (or pairs of lines) in the bus system

- demand messages, transmitted by a crate controller, if a module in its crate has set a LAM-signal.

The serial driver initiates a command message which contains 5 bytes in read or control commands or 9 bytes in write commands (when a 24-bit word is included). Fig. 1.14 demonstrates the format of a byte group. The first byte identifies the addressed crate. SC = 0 is used to address the driver and SC = 63 is not an allowed address, therefore 62 crate-addresses are available. When a controller receives a command message addressed to it, it checks



1.13

The maximum bus length and the type of communication lines are related to the overall design considerations. They will be selected on the basis of factors such as cost, required speed and distances. Because of the different types of communication lines and modes, only the signals and the lines at the I/O-ports are defined. Balanced receivers and transmitters are used, in connection with converters if necessary. The signal standards for the defind ports are based on CCITT recommendations.

In both the byte and bit serial modes, the unit of data is the 8-bit byte. In the bit serial mode, data bytes are transmitted in a frame of either 10 or 11 bits; the first is a 0-Start bit, followed by the data byte. The last bit or bits are 1-Stop bits. The transfer of bits or bytes are synchronized by a continuously running clock, whose frequency covers the range from 0 to 5 MHz.

The output of the serial driver generates a sequence of bytes, called the byte stream, which travel along the serial highway, passing through all crate controllers with timing dictated by the clock. If controllers want to transmit bytes to the serial driver they must do this in synchronism with the clock. The byte stream can contain message bytes or empty bytes, called WAIT-bytes.

Three different types of message are exchanged between the driver and the controllers:

- command messages, an instruction set up by the computer to an addressed crate in the case of a write command, the instruction plus the data,
- reply messages, control and status information sent by the addressed controller and, in the case of a read command, the control bits plus the data,

the message for error (longitudinal and transverse parity checks) and, if there is no failure, executes the accepted command. Bits M2 and M1, located in the second byte, are used by the serial driver to distinguish between the three types of messages. After the SUM-byte, the serial driver sends some SPACE-bytes to allow the addressed controller to insert the reply message within these bytes. A command-reply sequence is finished by an END-byte.

The reply from the addressed crate consists of 3 bytes in write or control commands or 7 bytes in read commands. The reply message is repeated by each downstream crate controller until it reaches the serial driver. Fig. 1.15 demonstrates the format of a reply message. The first byte specifies the address of the transmitting crate. The second byte includes the error (ERR) bit, set by the check logic in the controller if there was an error in the command message. If there was a failure in the previous operation the delayed error (DERR) bit is set. The ENDSUM byte contains the parity bits and completes the reply message.

Demand messages are constructed of 3 bytes, the first specifies the address of the demanding crate, the second the M2-bit together with the 5 graded-L bits, which are a logical priority arrangement of the LAM-bits or any combination of these bits. The first and the second byte together may provide a direct pointer to the LAM-source within the crate. Fig.1.16 shows the group of demand bytes.

Any crate controller which is capable of generating demand messages must have a 3-byte delay which is switched into or out of the highway between its I- and O-ports. This is used to delay messages passing through the controller

MSB 8	7	6	5	4	3	2	LSB 1	Byte-Nr.	
P1	0	SC32	SC16	SC8	SC4	SC2	SC1	Crate Address(D)	1
P2	0	M2=0	M1=0	SA8	SA4	SA2	SA1	Subaddress	2
P3	0	1	SF16	SF8	SF4	SF2	SF1	Function	3
P4	0	1	SN16	SN8	SN4	SN2	SN1	Station-No.	4
P5	0	SW24	SW23	SW22	SW21	SW20	SW19	for	5
P6	0	SW18	SW17	SW16	SW15	SW14	SW13	WRITE	6
P7	0	SW12	SW11	SW10	SW9	SW8	SW7	operations	7
P8	0	SW6	SW5	SW4	SW3	SW2	SW1	only	8
PΣ	0	Σ6	Σ5	Σ4	Σ3	Σ2	Σ1	SUM-Byte	9
1	0	1	1	1	1	1	1	SPACE-Byte	
1	1	1	0	0	0	0	0	END-Byte	

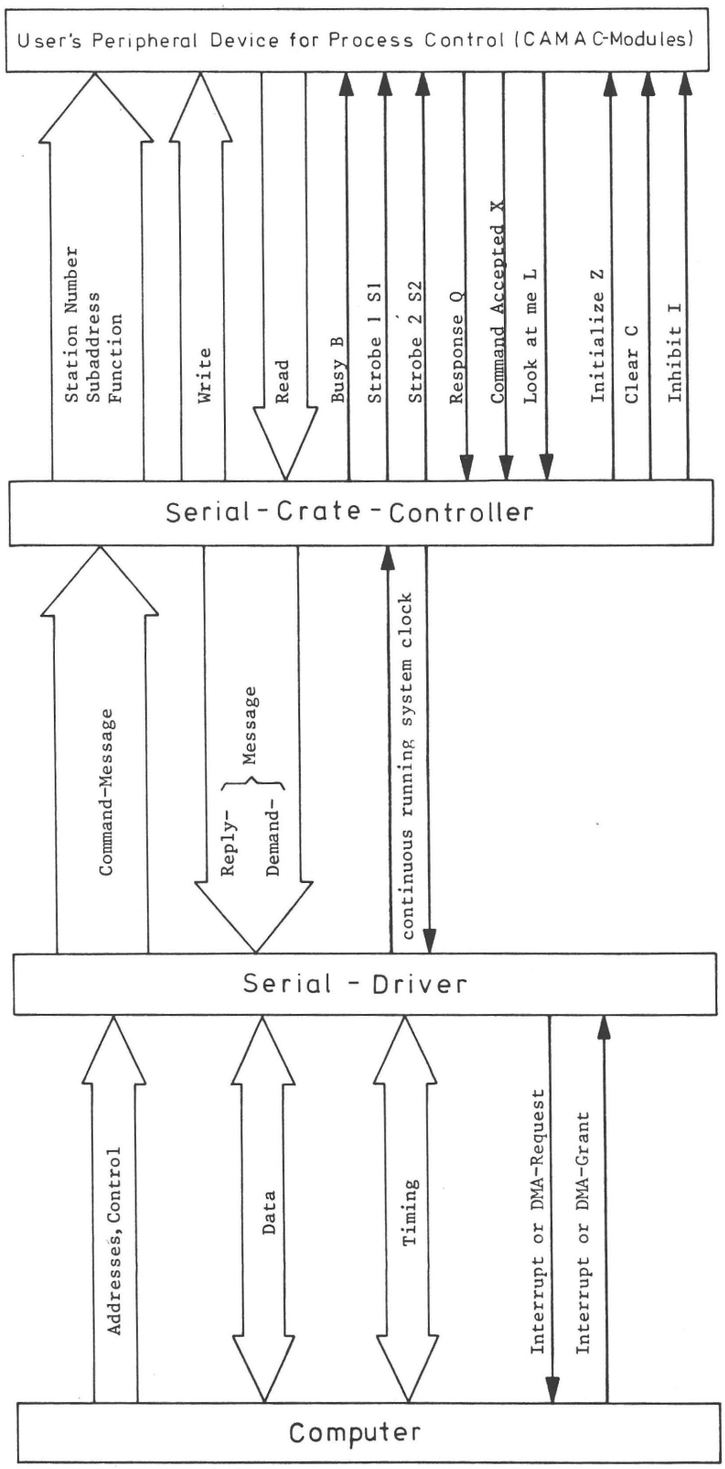
1.14

MSB 8	7	6	5	4	3	2	LSB 1	Byte Nr.	
P1	0	SC32	SC16	SC8	SC4	SC2	SC1	Crate Address (S)	1
P2	0	M2=0	M1=1	DERR	SQ	SX	ERR	Status Byte	2
P3	0	SR24	SR23	SR22	SR21	SR20	SR19	for	3
P4	0	SR18	SR17	SR16	SR15	SR14	SR13	READ	4
P5	0	SR12	SR11	SR10	SR9	SR8	SR7	operations	5
P6	0	SR6	SR5	SR4	SR3	SR2	SR1	only	6
PΣ	1	Σ6	Σ5	Σ4	Σ3	Σ2	Σ1	ENDSUM-Byte	7

1.15

8	7	6	5	4	3	2	1	Byte-Nr.	
P1	0	SC32	SC16	SC8	SC4	SC2	SC1	Crate Address (S)	1
P2	0	M2=1	SGL5	SGL4	SGL3	SGL2	SGL1	SGL-Byte	2
PΣ	1	Σ6	Σ5	Σ4	Σ3	Σ2	Σ1	ENDSUM-Byte	3

1.16


User's Peripheral Device for Process Control (CAMAC-Modules)
Station Number
Subaddress
Function
Write
Read
Busy B
Strobe 1 S1
Strobe 2 S2
Response Q
Command Accepted X
Look at me L
Initialize Z
Clear C
Inhibit I
Serial - Crate - Controller
Command-Message
Reply-
Demand-
Message
continuous running system clock
Serial - Driver
Addresses, Control
Data
Timing
Interrupt or DMA-Request
Interrupt or DMA-Grant
Computer
CAMAC - Dataway
CAMAC - Serial - Highway
Computer - I/O - Bus

1.17

and permits it to transfer demand messages without disturbing other messages passing through. The switching of the 3-byte delay must be synchronized with the system clock.

Fig. 1.17 is similar to Fig. 1.10 and demonstrates the direction of messages and the lines used in the serial highway. For simplification only one crate is shown.

1.5.5 What is the Job of the Crate Controller in the Serial Highway?

Serial crate controllers must accept the CAMAC commands set up by the computer and execute the desired CAMAC function. In contrast to the controllers in the branch highway the format of the reply and demand messages are different, also the parity-check operation is used only in the serial controllers. Responding to each executable command the controller must set a SX-bit and, if appropriate, also the SQ-bit. After each operation is performed the 3 bits ERR, SX and SQ will be written as DERR, DSX and DSQ into the status register, from where they can be read once more by the computer.

1.6 Typical CAMAC-Configurations

The configuration of the CAMAC-crates is adapted to suit the different applications in various fields.

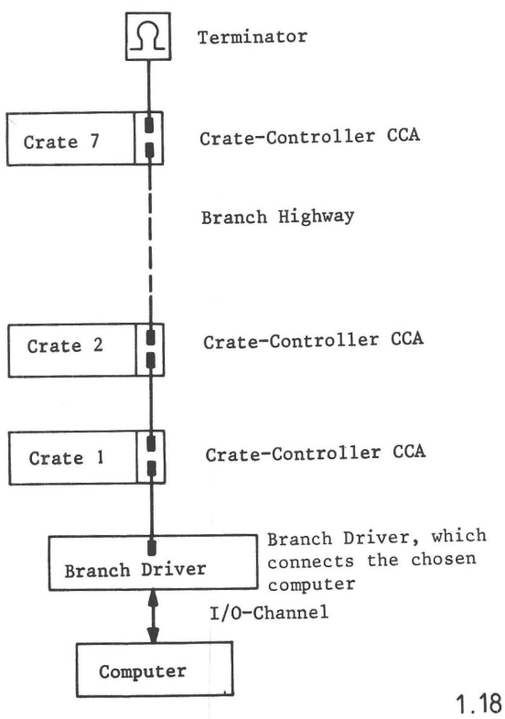
1.6.1 Parallel System with the Branch Highway

In typical CAMAC applications in the nuclear field the parallel bus with branch and Crate Controller A is used (see Chapter 1.5.2 and 1.5.3). Up to 7 crates are connected to the branch, which serves as a bidirectional data bus. It is terminated at both ends.

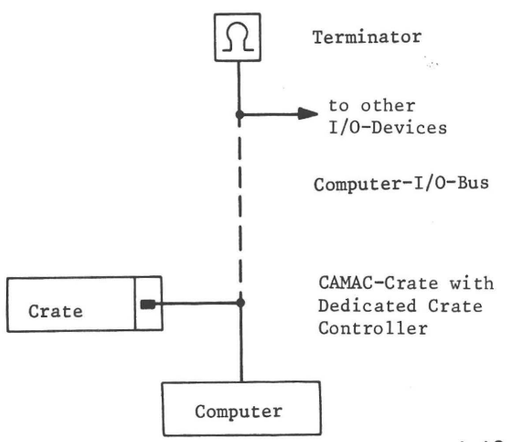
Connection to the computer is accomplished by the branch driver, which translates the branch highway to the I/O-bus. Fig.1.18 shows this configuration.

1.6.2 Parallel System without the Branch Highway

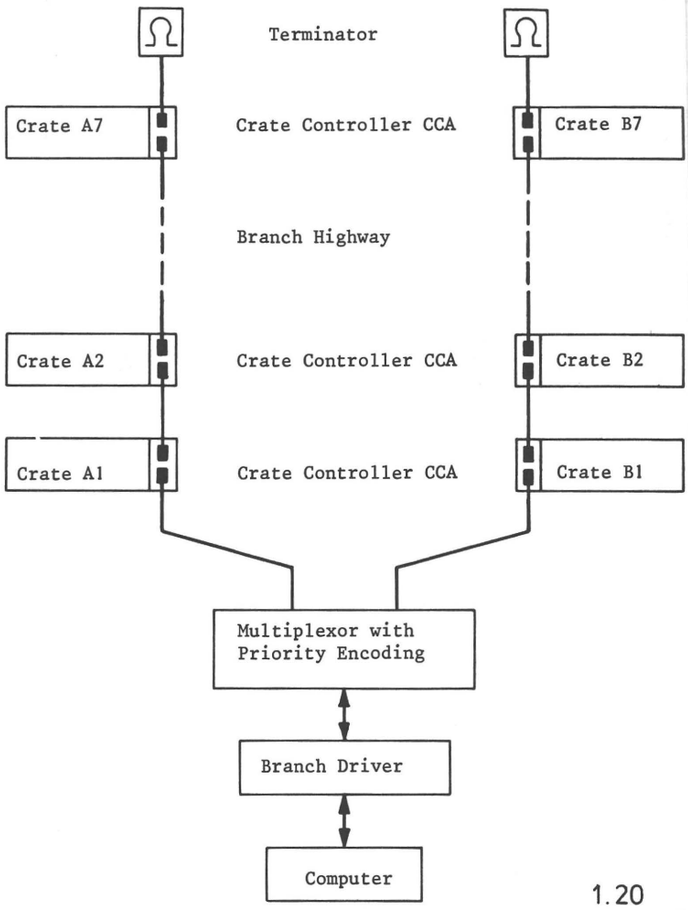
If the user has an existing computer system including some standard peripherals and he needs only one or two CAMAC crates, then an economical solution may be to use a dedicated crate controller (see Chapter 1.5.3). This controller translates directly between the crate highway (dataway) and the I/O-bus of the computer. Fig.1.19 demonstrates this configuration.



1.18



1.19



1.20

1.6.3 Parallel System with more than one Branch

If there are applications with parallel bus systems, where the number of stations included in 7 crates is not enough, then the use of several branches is recommended, Fig.1.20 shows this type of configuration. Each branch can be controlled by the computer via a multiplexer system.

1.6.4 Serial Bus System

Reasons for the use of a serial bus system are discussed in Chapters 1.4 and 1.5.4. Serial systems have lower cable costs but also, in general, lower speed. The CAMAC serial highway is suitable for up to 62 crates, which are connected in a unidirectional closed loop. Fig.1.13 in Chapter 1.5.4 shows the basic arrangement.

1.6.5 Serial Bus System with MODEMS

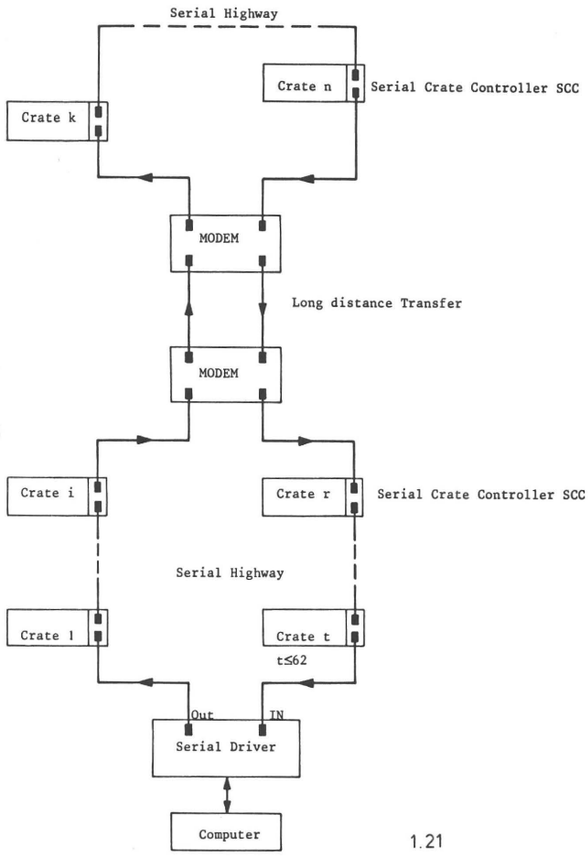
In a CAMAC serial highway operating at slow bit-rates the distances between the crates can be of the order of kilometers and the communication lines may be telephone lines. So one converts the signals from the source crate into the proper signal standard with a MODEM, transfers the data along the phone lines and converts them at the destination crate into the highway signal standard. Fig. 1.21 shows this application.

1.6.6 Serial Bus System with Opto-Couplers

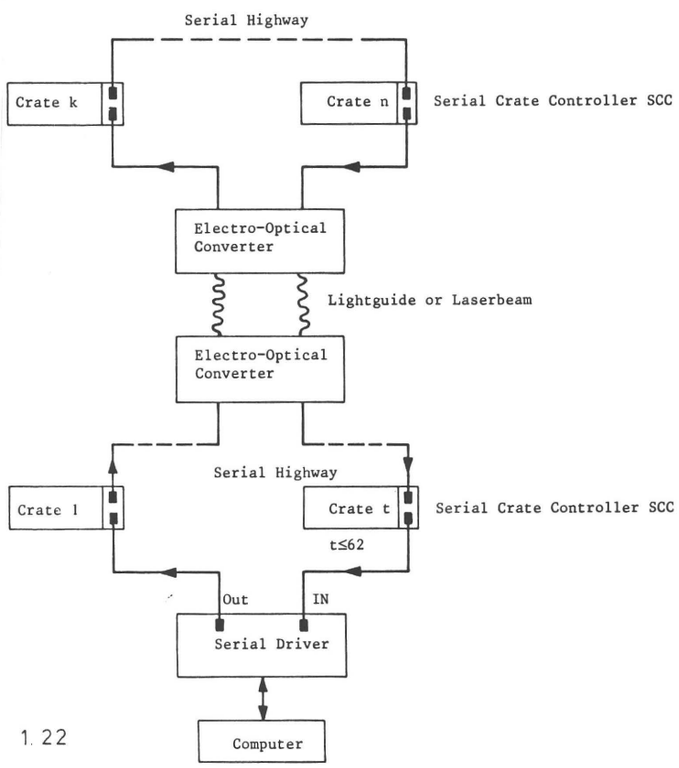
If data transfer is carried out between systems having high potential differences in signal amplitudes, or different earth potentials, then isolating opto-couplers can be used. Also laser coupling should be taken into account. Fig. 1.22 shows the fundamental arrangement.

1.6.7 Autonomous CAMAC Systems

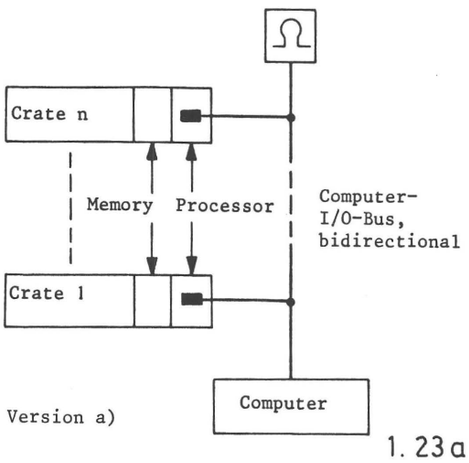
Normally a process is controlled by a main computer through one or more crates containing the process modules. But there are some applications, e.g. in medical and health service (see Chapter 2.2 and 2.2.1) or in laboratory automation (see Chapter 2.4) where the use of autonomous systems is of high interest. These configurations can be arranged by a mini or microcomputer which is now the dialog station between the peripherals and the main computer. These autonomous processors can accept information from the process, manipulate it with some arithmetic, which in case of the microprocessor is very limited, and store the results in a buffer memory. After this prepro-


Serial Highway
Crate n
Serial Crate Controller SCC
Crate k
MODEM
Long distance Transfer
MODEM
Crate i
Crate r
Serial Crate Controller SCC
Serial Highway
Crate l
Crate t
t≤62
Out
IN
Serial Driver
Computer

1.21


Serial Highway
Crate k
Crate n
Serial Crate Controller SCC
Electro-Optical Converter
Lightguide or Laserbeam
Electro-Optical Converter
Serial Highway
Crate l
Crate t
Serial Crate Controller SCC
t≤62
Out
IN
Serial Driver
Computer

1.22



Version a)

1.23a

cessing the manipulated data are sent by a block transfer to the main computer. Also the realtime facilities could be much better with autonomous controllers. The coupling between the controllers and the main computer can be done in several ways, Fig.1.23 shows two typical arrangements.

1.7 Is the Use of a Computer-Independent Interface Reasonable?

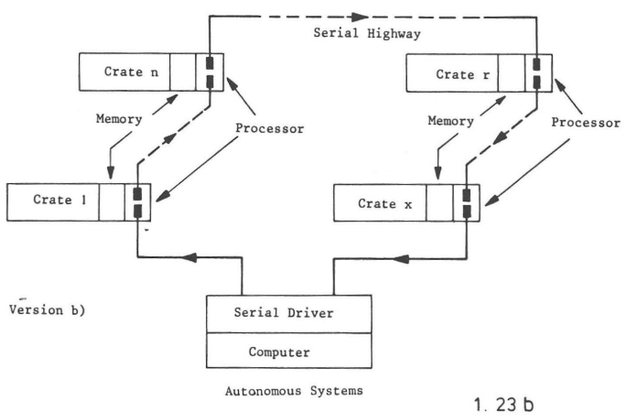
In the previous chapters we have learned something about the specifications and rules of standardized highways for transferring information between computers and peripheral devices controlling processes. So we can summarize the features of standardized interfaces.

A standardized system must not limit too much the designer of the hardware and software for data handling systems to serve the desired application. From this point of view, the CAMAC specifications fix only the essential rules. So the designers have sufficient possibilities to adapt CAMAC to suit their problems.

If there are applications using one or more computers with one or more interfaces, either correlated to each other or working separately, than the question is whether to buy computers with interfaces to a local standard from one manufacturer, or to use an international standardized interface and to buy any suitable computer from any manufacturer. The interface know-how of the well-chosen manufacturer may be excellent at the time, but is that a guarantee for the future? Each user should know that if he buys a manufacturer's local standard, it is possible that he may have to throw away all the equipment some years later when the manufacturer changes his standard.

Standardized systems are in the course of time always cheaper than others because of training the operators only once, and not every two years with new rules belonging to new systems.

In time, when the industry becomes more and more international, and when rationalization is a most important factor in designing new systems, one must look ahead for a data handling organisation which can fulfil most demands and ideas of the users in various applications. CAMAC is the only existing standard interface in world-wide use today.



Version b)

Autonomous Systems

1.23b

1.8 Glossary of Hardware Terms

Hardware — means all the mechanical, electrical or magnetical components which are included in computers and peripheral devices.

Software — implies the internal programs and subroutines, which specify the operation of a computer. Software includes assemblers, compilers, simulators, libraries of subroutines, operating systems and users' programs.

On-line — systems are those, in which the operation of the peripheral devices is directly controlled by the processor.

Off-line	means that the processor does not control the peripheral directly.
Realtime	operation involves the performance of a computation during the actual time set by the related physical process.
Interface	is a common boundary or a junction between two systems or two devices.
Bus system	is a highway to transfer information from various sources to various destinations.
Parallel transfer	means that all bits in the instruction or data word are transferred simultaneously via a corresponding number of lines.
Serial transfer	stands for a transfer mode in which either a bit or a byte at a time is transmitted via one line or a corresponding number of lines.
Bit	is the abbrevation of 'binary digit' which is one character of a binary number or one pulse of a group.
Byte	is a group of bits, normally 8, which are handled together.
Word	is a set of bits, e.g. the content of a memory location, which is handled as a unit by the computer. Word-length refers to number of bits, word organisation to the arrangement of bits within the word. The control unit of the computer handles instruction words, the arithmetic unit data words.
Parity	is an error detecting feature consisting of an additional bit added to a group of bits which makes the total number of 1's even or odd (even parity, odd parity).
Flip-Flop (FF)	is a circuit having two stable states. It can store one bit of information.
Set	is the true signal to one of the two FF-inputs, which sets the Q-output (this stands for the content of the FF) to 1.
Clear	is the true signal to the other input of the FF, which resets the Q-output to 0.
Register	is a set of FF's, in which a computer word or parts of it can be stored, e.g. to perform arithmetical, logical or transfer operations.
Increment	means the addition of a fixed value to a number, e.g. the addition of one to the content of a program counter to address the next memory location.
Decrement	is the inverse of increment, it means the subtraction of a fixed value from a number.
Strobe pulse	is a signal which opens and closes gates simultaneously to let information pass through the gates at a fixed time.
Processor	is the central unit of a computer. It accesses data, manipulates them arithmetically according to a program, and stores the result in the memory. The processor includes the arithmetic logic unit, the instruction register with decoder, general register for use as accumulators, index registers and pointers, the priority logic for bus control or interrupts and the timing.

Interrupt

is a request signal, coming from a peripheral device, to stop the current program handled by the processor. If the processor grants the signal, because of the higher priority of the requesting device, the processor starts a subroutine which is called by the peripheral.

I/O-device

is a general term for a peripheral device, which communicates with the computer.

I/O-channel

is that part of a computer which receives or transmits information from or to the I/O-devices.

MODEM

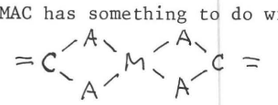
is the abbreviation of the term MOdulator-DEModulator unit. A MODEM converts signals to or from a form suitable for transmission via a communication channel.

Opto-couplers

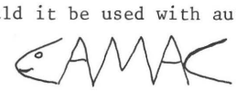
are used, if signals are exchanged between systems which have different earth potentials. They consist of light emitting diodes and light sensitive transistors. Isolation between these components is due to optical material, e.g. plastics, lightguides etc.

CAMAC

is a puzzle word, which has many translations, e.g. a serious one:
Computer Aided Measurements And Control
or with a smile:
Confuse All Measurements in Any Case
or
Cooking Aid for Many Applications with Computers.
Maybe, CAMAC has something to do with chemistry?



Or should it be used with automatic fishing?



We are interested in your ideas!

1.9 Hardware-Literature

(H1) CAMAC - A Modular Instrumentation System for Data Handling; Revised Description and Specification, Commission of the European Communities, Report EUR 4100e, 1972.

(H2) CAMAC - Organisation of Multi-Crate Systems; Specification of the Branch Highway and CAMAC Crate Controller Type A; Commission of the European Communities, Report EUR 4600e, 1972.

(H3) CAMAC - Serial System Organisation - A Description, ESONE/SH/01, ESONE-Secretariat, Dr. H. Meyer, CBNM Euratom, Steenweg naar Retie, B-2440, Geel, Belgium.

(H4) CAMAC-Bulletin; A publication of the ESONE Committee, Commission of the European Communities, Luxemburg, P.O.Box 1003.

(H5) CAMAC Bibliography, CAMAC Bulletin No.8, Suppl., November 1973.

(H6) A CAMAC Glossary, CAMAC Bulletin No.7, Suppl., July 1973.

(H7) L. Costrell, NBS, Highways for CAMAC-Systems, A Brief Introduction, IEEE Trans. on Nucl. Sci., Vol.NS-21, No.1, Feb. 1974, p.1.

(H8) F. A. Kirsten; CAMAC-Specifications, IEEE Trans. on Nucl. Sci., Vol.NS-20, No.1, Feb. 1973, p. 562.

(H9) R. S. Larsen; CAMAC Dataway and Branch Highway Signal Standards, IEEE Trans. on Nucl. Sci., Vol.NS-20, No.2, April 1973, p. 28.

(H10) S. Dhawan; CAMAC Crate Controller Type A, IEEE Trans. on Nucl. Sci., Vol.NS-20, No.2, April 1973, p.35.

(H11) R. C. M. Barnes; The CAMAC Serial Highway - A Preview, CAMAC Bulletin No.8, November 1973, p.5.

(H12) D. R. Machen; The CAMAC Serial Systems Description for Long Line, Multicrate Applications, IEEE Trans. on Nucl. Sci., Vol.NS-21, No.1, p. 876.

2. How is CAMAC Used?

2.1 Introduction

CAMAC, described in part 1, is a very versatile interface for data handling systems. In this part we will pick out some characteristic applications in the following fields:
- medicine and health services,
- industrial process control,
- laboratory automation and
- data communication

and give some basic remarks. Additional reports on successful use of CAMAC in these and other fields are given in the CAMAC Bulletin, the IEEE Transactions on Nuclear Science and other relevant periodicals.

The detectors used in measuring the relevant on-line data produce analog signals in the range from micro volts to volts. These signals must be amplified, shaped and converted into digital numbers before they can be used on the CAMAC highways. Some conditions for these signals are specified in (A1), additional specifications are being prepared by the ESONE Analog Working Group.

2.2 CAMAC in Medicine

CAMAC has a good chance of being introduced in medicine because it is a modular and flexible system. Applications up to now are located in
- intensive care stations,
- nuclear medicine,
- acquisition of physiological data (ECG, EEG etc.)
- clinical laboratories.

In intensive care stations, where the condition of a patient is highly unstable when breathing or heartbeat stops, some critical values must be controlled permanently, values such as
- blood pressure,
- frequency of breathing,
- frequency of heart,
- temperature.

From the measured values trends are calculated to avoid false alarm situations. There is no need to store the accessed data over a long period because they are used mainly by doctors and nurses watching the displays.

The detectors are connected to modules in a crate dedicated to the patient's bed. The analog signals are converted to digital numbers which are manipulated with other parameters (preprocessing) and transferred to a computer via a standardized interface.

In the nuclear medicine field CAMAC is often applied to diagnose the function of organs, using the scintigraphy method. Radioactive materials are employed as tracers in
- long term measurements, in which the values are taken within periods of hours or days
- short term measurements, in which the periods are in the range of milliseconds, because the CRT-displays giving the time distribution of the tracers in the organs should be available for the doctors immediately after acquisition to permit interactive diagnosis. These applications require relatively high data rates to measure up to 1000 events per second.

The detectors used in scintigraphical methods are often gamma-cameras, which convert γ-radiation into pulse signals by means of scintillation counters, and calculate the coordinates of the γ-quanta. The spatial variation of the measured channel gives the distribution in the organ on the whole. Typical examples are the discovery of brain tumours, tests on the thyroid gland, blood volume evaluations etc. In these applications CAMAC is used as an on-line data acquisition system with preprocessing of the measured values, together with some interactive system facilities.

Acquisition of physiological data means measuring the electrical activities of heart or brain. The results are used to determine the function of the organs.

The electrical charge distribution of the heart with respect to time can be measured by electrodes mounted on the patient's body. The result is the ECG (electrocardiagram). Analog signals in the microvolt range are amplified, shaped and converted to a digital form, before they are stored on tape. Data acquisition is on-line, analysis of the ECG is off-line.

Measuring the electrical signals from the brain is done by electrodes mounted on the head of the patient. The signals are sorted in 4 frequency channels (from 0 to 30 Hz). Data acquisition and handling is similar to that in the ECG case.

In clinical laboratories computer-aided data handling systems are used to acquire the patient's history and clinical data. The latter are typically derived from the examination of liquids like blood, urine or plasma. These data are taken by a mixture of physical, chemical and electronic devices from various manufacturers, with varying degrees of automation. After some preprocessing the measured values are combined with dedicated data identifying the patient. This data has to be checked for plausibility and is then transferred to the computer, where it is stored and displayed. The doctor can therefore use it immediately to make a diagnosis.

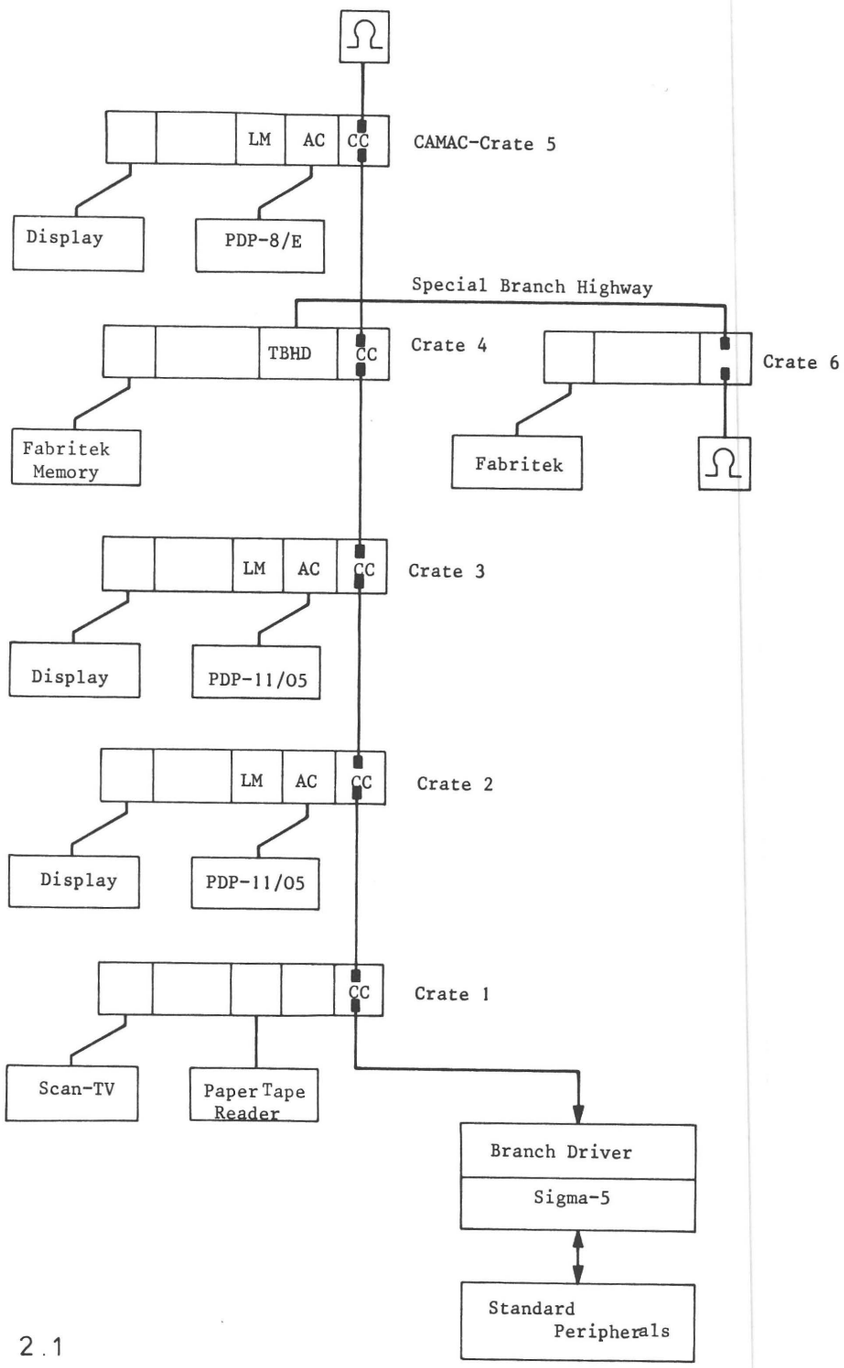
The measuring devices we have mentioned are of different types, not only dif-

ferent in function but also in the electrical lay-out and read-out circuitry. Therefore only a standardized interface is an economical solution to the problem of connecting these different devices to a central computer.

The computer normally has an operating system which can handle more than one task simultaneously, so the peripheral devices are connected in a satellite configuration. But there are reasons why a central computer servicing non-intelligent peripherals is not the best solution. If, for example, there are failures in the processor all the I/O-devices are out of action.

Also changes of configuration within such a small I/O-system could be performed much better if each satellite is more independent of the main computer. Therefore we see that in some cases it is better to design autonomous systems each handling only one task and all connected to a main computer through a standardized interface bus.

Such autonomous systems we have discussed basically in Chapter 1.6.7. The heart of this intelligent subsystem is a mini- or microprocessor with a memory for program and data. Microprocessors may have not a very extended instruction set for arithmetic and have also a



2.1

very limited memory capacity but they can handle many preprocessing features. They are freely programmable processors (some of them are also microprogrammable), so they are powerful autonomous crate controllers.

The connection to the interface can be made in a different manner from that shown in Fig.1.23 in Chapter 1.6.7. Normally the CAMAC serial highway configuration with byte-organized data transfer should be the best fit and the most economical solution. Typical microprocessors of the second generation are designed around a bidirectional 8-bit data bus, therefore the byte is the right data format in this case. But in some applications with high data rates, e.g. in scintigraphy, it may be better to use 16- or 24-bit miniprocessors connected to a parallel branch.

2.2.1 A CAMAC-Application within a Clinical Laboratory

Fig.2.1 shows the arrangement described in (A2). A SIGMA-5 computer is used as a central computer, with a 128 k-byte memory, a 6 M-byte disc and additional peripherals. Six crates are connected via a branch driver, the sixth crate being driven through a special driver module housed in Crate 4, because the distance is more than 300 m. The SIGMA-5 can handle the CAMAC crates in realtime operation. Interactive mode is also possible because of teletypes and displays which are dedicated to the crates.

To improve the realtime facilities some crates have not only the standardized crate controller (CC) but also an autonomous controller (AC) linked to a minicomputer PDP-8 or PDP-11. This autonomous controller is at a normal station in the crate and connected to the crate highway (dataway) in the same way as other modules. In the arrangement described here there is no arbitration unit assigning priority to one of the two controllers. The first controller to start the CAMAC cycle completes its operation before the second controller can start a cycle. Because two different minicomputers are used there must also be two different autonomous controllers driving the bus lines (OMNIBUS and UNIBUS). During the realtime phase of the operation the collected data are stored in the main memory of the PDP from where they are transferred to the SIGMA-5 computer via the PDP-bus, the branch and the I/O-channel of the main computer.

Communication between the autonomous controller and the crate controller (and hence between the local minicomputer and the SIGMA-5) is carried out via a link module (LM) which is a CAMAC module addressable from both controllers. Information from the SIGMA-5 can be written into the link module, the processor of the mini then starts a read operation that accesses the written data, and vice versa.

The 6 crates are used in the clinical laboratory in the following manner:

- Crate 1 is located in the same room as the SIGMA-5. It is equipped with various modules like ADC, DAC, Display Driver and Paper Tape Reader which can drive different displays, e.g. graphics terminals, X-Y plotter, TV monitor via the Scan-converter, or line printers. Analog data from magnetic tape or digital data from paper tape can be analyzed by the SIGMA-5. Crate 1 is also used for testing modules and programs.
- Crate 2 is used in a laboratory for neurophysiological research. Modules are ADC, DAC, Counter, Display Driver to a 611 Tektronix scope, etc. The crate is controlled by the CC-controller and a PDP-11/05 processor as an autonomous controller.
- Crate 3 is also controlled by an additional autonomous unit connected to a PDP-11/05 processor. The crate is used to control neurophysiological experiments, therefore it has similar modules to the previous crate.
- Crate 4 is in a laboratory for pharmacological research. One module drives a Fabritek 1k memory unit with a hardwired program to execute analog to digital conversion. Another module is the special driver for the branch to Crate 6.
- Crate 5 is used in a laboratory for otolaryngological research. This crate is controlled by the CC and a PDP-8 autonomous controller.
- Crate 6, which is 300 m away from the SIGMA-5, is used for research in biochemistry.

From this arrangement one can see that in some applications where the data rate is high or the response time in realtime must be very short, the use of relatively fast autonomous mini- or microcomputers is most efficient. Therefore users in this field of application need such configurations.

2.3 CAMAC for Industrial Process Control

Ten years ago the control of industrial processes with computers was invented. Since this time there have been many applications in different fields. In very simple systems the computer is used only as a data logging device, an application which can sometimes be done better with hardwired logic or now with microprocessors. In complicated systems e.g. controlling chemical processes or the operation of steel plants, there is a manifold need for realtime computer configurations, sometimes organized hierarchically.

In the industrial field inventors of computer and interface systems have had to contend with difficult operational conditions like

- changing composition of the room atmosphere,
- mechanical shocks,
- electromagnetical interference signals, from nearby cranes etc.

After extended investigations of performance in large industrial systems especially in the United States (A3, A4) one can say that CAMAC is suited at least as well as systems used previously.

Detectors used in industrial process applications also normally deliver analog signals in the microvolt to millivolt range, which must be amplified, shaped and converted in an ADC. Signals may also come from detectors the output of which is a current source, therefore the current has to be converted into a proportional voltage before the signals are digitized. Other signals from synchrodetectors give information proportional to an angular position. They may be also phase modulated. Such types of signals can be digitized in a synchro-digital converter.

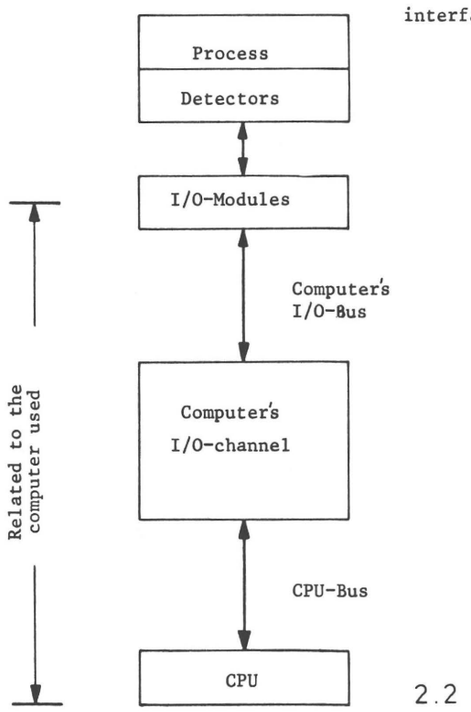
But engineers designing I/O-modules for process control have to take into account not only these signal conditions, they must also solve problems about the connection to the computer's I/O-bus. If the system is designed directly in combination with the I/O-channel of the desired computer (see Fig.2.2) then the engineers will have to change the I/O-modules if at any time a better or bigger computer is used, because many details of the I/O-bus are implemented in the modules.

Standardization of the data highways between process and computer is therefore very useful because in such systems the I/O-modules are not affected by the use of other types of computers. This can be seen in Fig.2.3 which demonstrates a CAMAC configuration for a typical process control system. Changing the computer type requires only a new branch or serial driver, nothing else.

The great advantages of choosing CAMAC in industrial plants are the following:

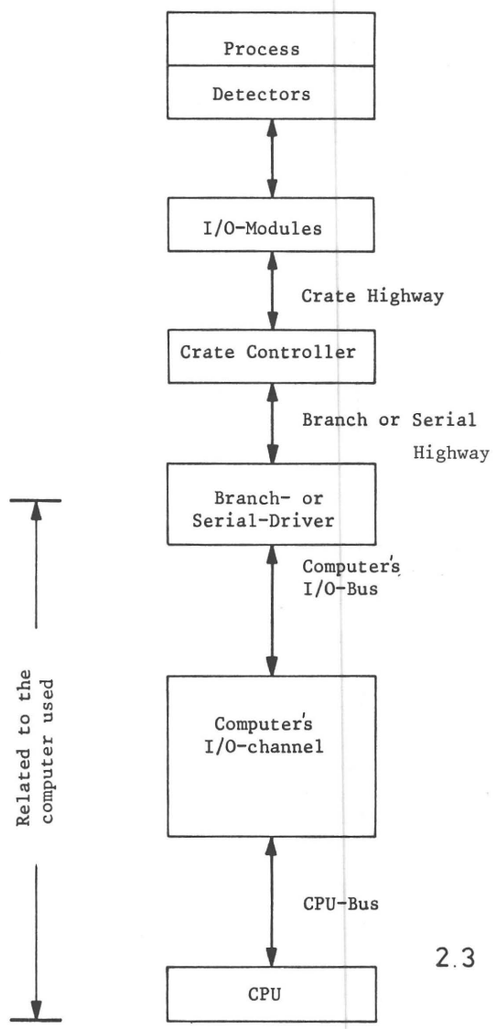
- many manufacturers are delivering CAMAC modules (multi-sourcing),

Arrangement of a controlled process using a computer with home standard interface



2.2

Typical arrangement of a CAMAC controlled process



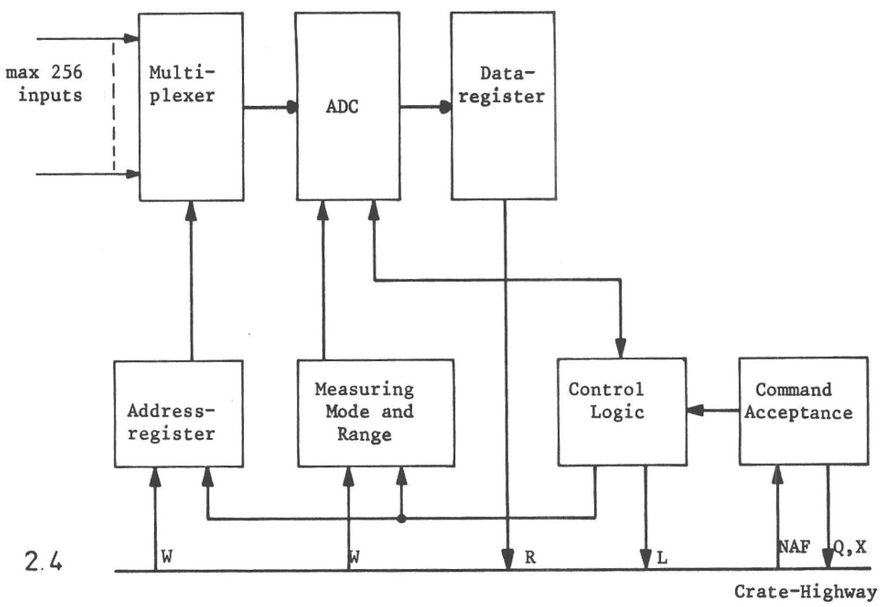
2.3

- the system is a modular one, therefore the user can add new modules at a later time if he needs them,
- the user has to have in stock only crates, modules and bus lines of one system. This is true also if he is using different types of computers.
- the technical staff of the user have to be trained in only one system, which may be a great advantage.

2.3.1 A CAMAC-Controlled ADC System for Process Control

Many industrial processes are described by a great number of values which can be measured and used to control the process itself. The accuracy of measurement of analog signals from the detectors must often be of the order of 10^{-3} or better. Because of the price of an ADC with a resolution of 12 bits or more, it may be better to design a device having one high-accuracy ADC and many input lines switched by a multiplexer. Such a solution is certainly less

expensive than having one ADC of medium resolution and medium price in each I/O-module, but the access and conversion time is much higher because the signal handling for the various inputs is done in sequence. In control systems in which time is not the most important point, ADC-modules of complex design like the following described in (A5) are very useful. Fig.2.4 shows the basic arrangement.



2.4

The ADC is coupled to 256 input lines through a relay multiplexer which has an ultra low ON-resistance. All input lines are symmetrical pairs, twisted or parallel for high common mode rejection. Each of them is switched with 2 relay contacts.

Because of the 256 different inputs, the multiplexer is addressed by an 8-bit word in the address register. The multiplexer can be operated in several modes:

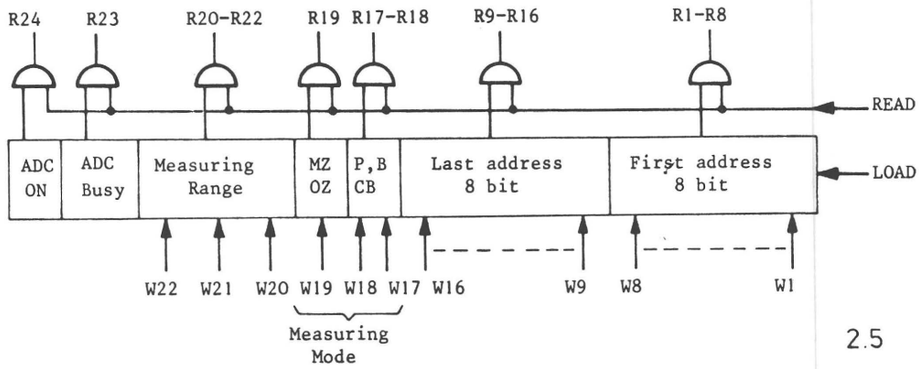
- the programmed transfer mode, in which only one input line is addressed; the input signal is digitized and transferred through the bus to the computer,
- the block transfer mode, in which a set of input lines, defined by the first and the last address, is switched in sequence to the ADC-input. Each digitized signal is stored in a buffer memory, and the whole set is transferred to the computer after converting the last signal.
- cyclic block transfer mode, in which all input lines within the block of addresses are switched periodically to the ADC. After converting the last signal in each block, the data set is transferred to the computer.

Additional power lines connected to a very stable voltage or current source can be switched to passive detectors (which do not have their own power supply) some hundred milliseconds before addressing the detector.

Fig.2.5 shows the bit configuration of the control and status register. In the programmed transfer mode only the first address is loaded through the CAMAC-write lines W1-W8, in the other modes also the last address within the block through W9-W16. Bits W17-W19 specify one of the three transfer modes, W19 defines whether the detector needs an additional power supply or not, W20-W22 choose the voltage range from 10 mV to 1000 V full scale. The last 2 bits are set if the ADC is ON and if it is busy because it is converting a signal from an addressed input line. The control and status register can be read by the computer.

To execute the desired CAMAC functions, the instructions coming from the computer via the highways are checked for validity. During the conversion time the content of the data register is not fixed, therefore it should not be read by the computer. That means a read instruction during that time is acceptable (the ADC responds with X = 1) but not executable (the Q-response is Q = 0). After the conversion the Q-line is set (Q = 1) and the read instruction can be performed. Other instructions like loading or reading the control and status registers are handled in the same manner.

The control logic produces the signals to switch the ADC into the ON or OFF state to change the measuring ranges, and to address



2.5

Control and Status Register

the multiplexers. In the block transfer mode the range cannot be altered, in the programmed transfer mode it can be changed between two addresses.

Such a complex system is not only useful in industrial process control but also in the field of laboratory automation, where CAMAC is already a well-known interface configuration.

2.4 CAMAC and the Automation of Laboratories

Modern laboratories are overcrowded with measuring instruments of different types, each of them designed with circuits to read out the measured value. There is a growing trend to analyse these data by computers, since minicomputers are on the market with very powerful instruction sets and timesharing possibilities at a medium price.

If we distinguish the instruments with respect to their data rate, we see that only a few generate high rates, in the range of 10^5 words/s, e.g. mass spectrometers with high resolution. Most of them transmit data at a medium rate between $5 \cdot 10^2$ words/s and $5 \cdot 10^3$ words/s, e.g. mass spectrometers with medium resolution, nuclear resonance spectrometers. Gaschromatographs are typical instruments with very low rates in the range of 1 word/s.

Measuring instruments in laboratories usually deliver analog voltages between millivolts and volts which are amplified, shaped and digitized in the manner described in Chapter 2.3.1. Then the data are sent to the computer through the interface.

High data rates could need many memory bits, but this is a very expensive solution. Instead of using a computer with a megabyte memory one should consider having intelligent preprocessing, resulting in a large data reduction. This is a very useful and economical solution. This job can be done sometimes by hardwired logic, but there is more scope if it is carried out by programmable microprocessors. This configuration leads us to the application of autonomous systems we have introduced in Chapter 1.6.7 and discussed in Chapter 2.2. We see that this type of processor-processor link will be of more and more interest. Because microprocessors are able to make simple arithmetical operations, some instruments can also use these processors in realtime phases of the process.

If the interface highways are standardized, many different types of instruments can be connected to the bus systems. In that way CAMAC has a good chance to be the most widely-used system in laboratory automation in the future.

2.4.1 LABCOM, a CAMAC-Controlled Laboratory Instrument System.

The possibility discussed in the previous chapter, of using several autonomous systems, was not applied to this configuration of laboratory automation at Harwell, which is described in (A6). The central computer is a PDP-8/I with disc and tape units. Its operating system is able to handle several tasks simultaneously. Data acquisition is done by measuring instruments connected to CAMAC modules. Each CAMAC crate services a group of instruments, and is connected to the computer in a time-sharing mode of operation. Interactive data exchange is possible between teletypes and displays.

Fig.2.6 shows the crate configuration. Each crate includes the I/O-modules for the connected instruments, which are for measurements in UV and IR spectroscopy, and electron spin resonance.

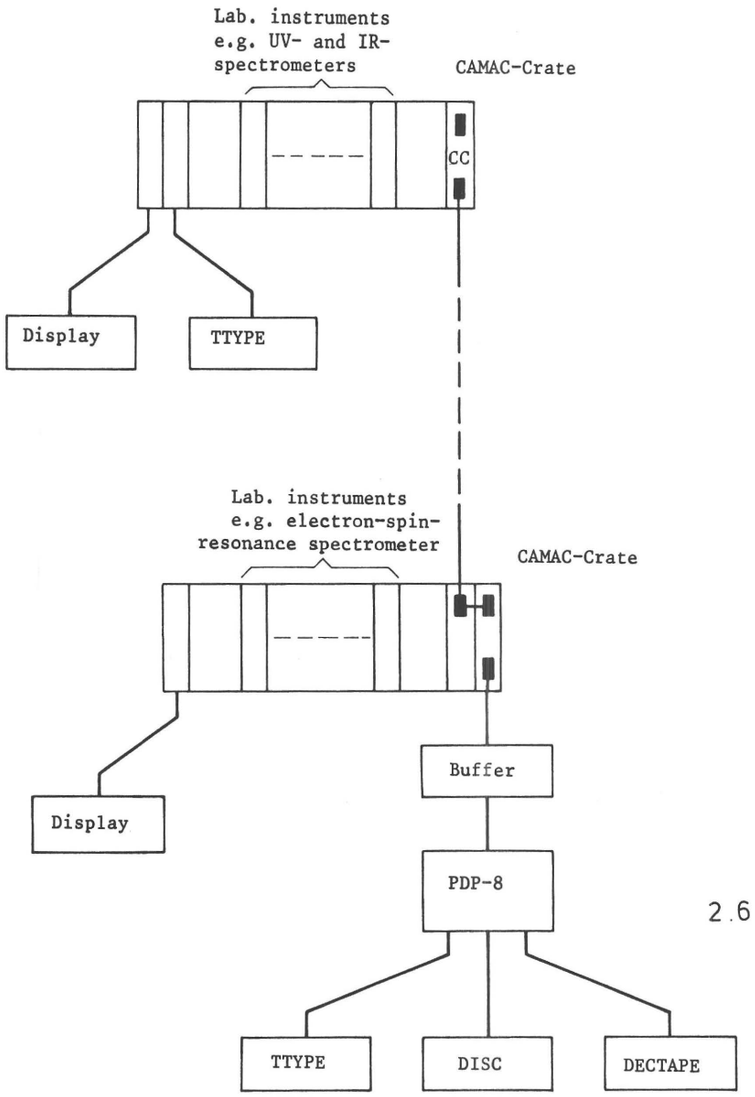
The time-sharing mode, which means handling tasks simultaneously, is controlled by a supervising program. This real-time supervisor deals with a hierarchy of interrupts. It maintains a priority-ordered list of programs awaiting attention, organizes links between programs, and organizes transfers to or from the disc. The main programs are written in such a way that it is easy to include new instruments in the system.

The CAMAC hardware used in this laboratory automation configuration has provided a flexible means of interfacing different instruments to a central computer.

A CAMAC setup similar to that described in Chapter 2.3.1 is published in (A7). It serves an analog-digital-converter system with 768 analog inputs addressed by a multiplexer at a maximum rate of 10 channels per second. Hardware and software problems are considered there in detail.

2.5 CAMAC in Data Networks

Many applications in the fields described make use of data exchange between computers. Thus, users can take advantage of the features of other computers connected through the networks. Therefore their own data handling



2.6

problems can be solved much better because of the speed or capacity of these computers.

It is very useful to have a standardized interface between the computers. We shall discuss in this chapter some modules and configurations by which CAMAC is used to exchange data within computer networks.

2.5.1 CAMAC-CAMAC Links

Links between CAMAC systems can be realized in various ways. Examples will be given of parallel, and bit or byte serial transfer modes and also a microwave data link.

The first example shows a bidirectional station described at (A8). Circuits are very similar in the receiver and the transmitter module except for a number of lines for control and status signals.

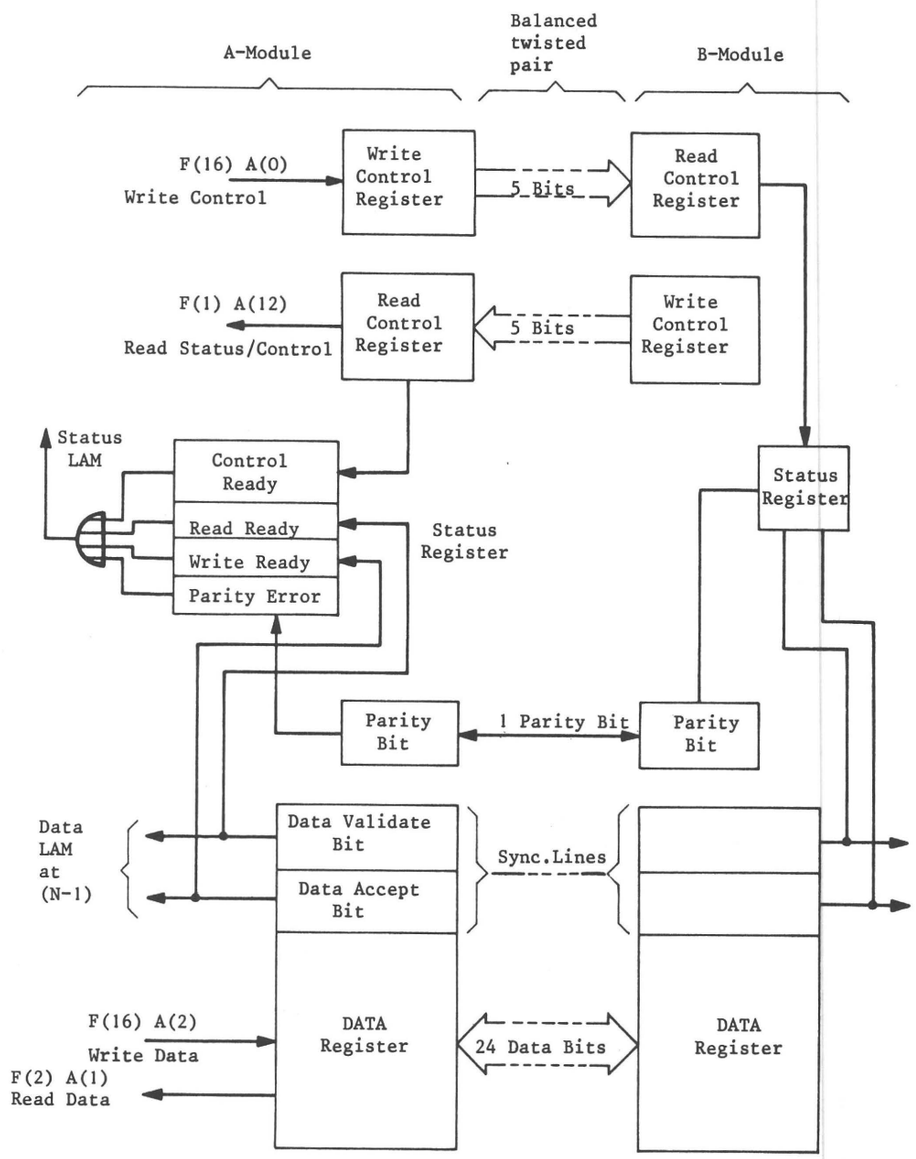
The bidirectional data link handles 24-bit CAMAC data words. Additionally, there are two groups of 5-bit paths transferring control words in each direction. These words, together with the data, generate 4 status bits which are ORed to a LAM signal that can be enabled or disabled by a mask. A data parity bit is generated at the transmitting and at the receiving end, and is transferred together with the data (see Fig.2.7).

Normally the transfer of data and control words are made in the handshake mode. Synchronization can be achieved by transmitting two additional bits confirming the acceptance and validity of the data. These two bits generate a LAM-signal in station N (the module is of double-width), due to a parity error or the arrival of a control word. The data transfer can be operated with or without interrupts.

The generation of the data-validate bits is started when the Q-signal is true, therefore one can write the flow charts of Fig.2.8.

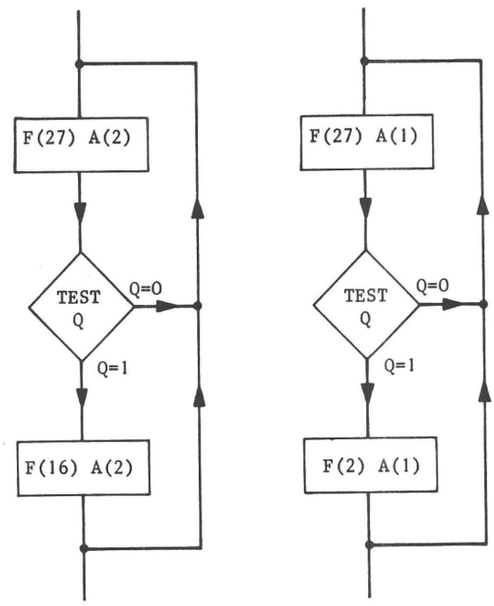
The speed of the data transfer is limited only by the CAMAC cycle time, e.g. 1-1.5 µs for each 24-bit word. The link consists of twisted pair cables terminated at each end.

The second example describes the linking of computers via a transmission loop. Auto-



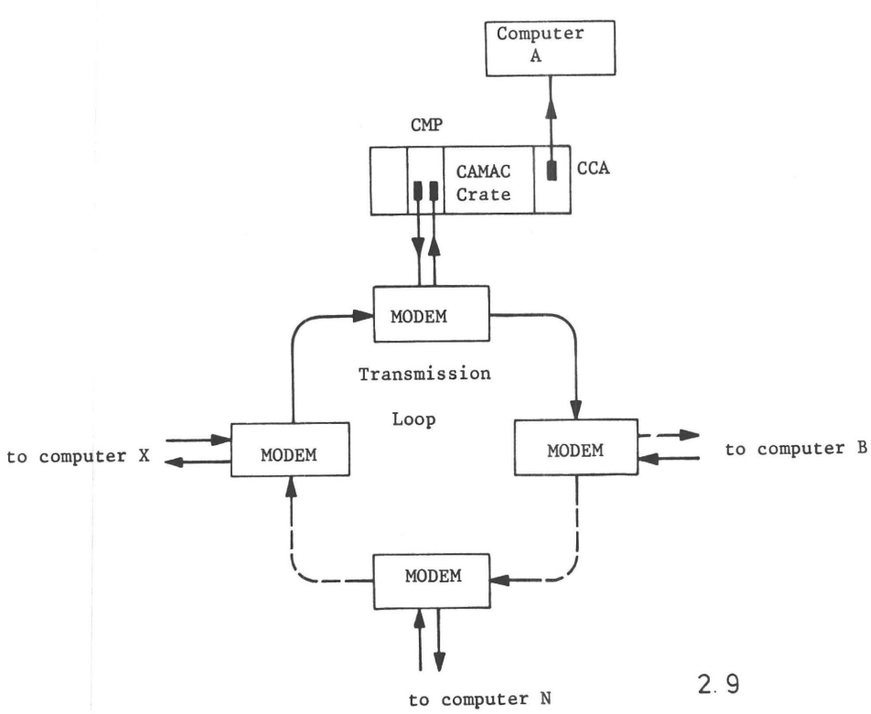
Block diagram of CAMAC-CAMAC link

2.7

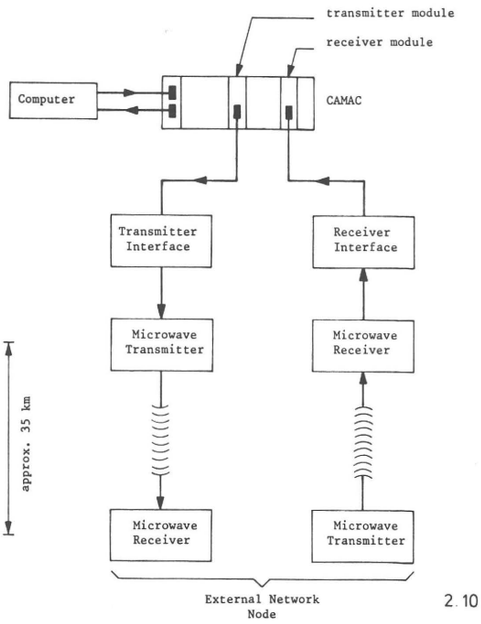


2.8

Simple command sequence for data transfers.



2.9



DNPL microwave data link

nomous CAMAC systems, containing CAMAC I/O-modules and the associated computer, are connected to a MODEM through a CAMAC message processor, a double-width unit in the crate. The message processor (see Fig.2.9) provides intermediate storage within a buffer memory (16 words, 24 bits) code conversion and error checking. The transmission loop consists of a data pair and a control pair.

If computer A, for example, wants to transmit a message, it must get control of the bus by becoming bus master. This can be done by opening the control loop, which prevents other processors from getting control. After finishing the data transfer, the control loop is closed and the next processor is able to transmit data. A pulse traveling along the control pair reaches the ports sequentially its trailing edge allows the connected processor to open the loop if its computer has something to say, if not the pulse travels to the next port. The circulating pulse is stored if the loop is opened.

The message to be sent out from a computer is stored in the buffer of the message processor. It is a group of 48 bytes. The first byte contains the destination address. After accepting the message, the processor seizes the first opportunity to get bus control. The next time the pulse comes into its port the messsage processor switches a duplex link to the called processor. It checks the identity of the correspondent and asks if it is ready to receive a message. If all the checks are positive the message is sent out. The receiving processor carries out the parity check, stores the group of 48 bytes and acknowledges their acceptance. The transmitting processor announces to its computer the correct transfer or, if not, what are the reasons for the error. Then the connection is broken and the bus is free for any other computers which want to send messages.

The processor which has received the message sets up a LAM-signal to its computer to cause an interrupt, after which the new data can be handled.

The maximum speed on the transmission loop is 500k Baud, serial asynchronous. The maximum number of ports is 256.

The third example of CAMAC in data communication is shown in Fig.2.10, and is described in (A10). Information is transferred over a distance of up to 35 km using standard Post Office video microwave transmitters and receivers. The data and control signals are pulse modulated. The carrier is at 6.8 GHz, and the band-width 5 MHz. The transmission rate is 10^7 bits/s, and the error rate is less than 1 bit in 10^{10}. Extended tests were made by performing interactive graphical tasks.

The last example is related to a demonstration of the possibilities of the CAMAC Serial Highway, at Los Alamos in Summer 1974 (A11). Two systems were shown, in the first case the distance between the serial driver and a crate was 5000 km, therefore the data rates were low; in the second case the rates were high, but the distance was only a few meters.

A serial crate controller at Los Alamos, New Mexico, was connected via MODEMS and telephone lines to a serial driver which was located in Lockport, Illinois. LAM-signals generated in Los Alamos were interpreted appropriately and executed by the associated computer. The computer then transmitted command messages to generate a display on a TV monitor in Los Alamos. The transfer rate was 300 bits/s.

The other system was completely housed in one room. Data were transferred successfully at a rate of 2.5 M bit/s between a serial driver and a CAMAC crate. The serial highway will be the most interesting CAMAC connection in the future not only over long distances but also in laboratory applications.

2.6 Glossary of Application Terms

Autonomous systems	have one or more crates in which operations can proceed independently of the program in the system computer. Such crates may be controlled by a local mini- or microprocessor. Several autonomous systems can be connected via an interface to a main computer.
Process control system	is a system whose main use is to automate a continuously running process.
Transducers	are detectors converting variables such as temperature, pressure or mechanical movement into equivalent electrical variables which can be measured easily.
Analog-digital-converters	convert an analog voltage into a binary coded digital word whose content is proportional to the measured voltage. There are different conversion methods, the resolution is typically between 8 and 17 bits with an error of 10^{-3} to 10^{-5} of the fullscale. The conversion time varies from micro- to milliseconds.
Digital-analog-converters	handle the inverse function, and convert the content of a digital word into a proportional analog voltage. Resolution ranges from 8 to 16 bit with an error of linearity from 10^{-3} to 10^{-5}. The conversion is performed in the time range of 40 ns to 250 µs.
Data logging	is the term for automatically recording process data within a desired time interval.
Preprocessing	is normally a hard-wired sequence of operations to manipulate and reduce the accepted data. The processing is done by arithmetical and logical operations.
Scintigraphy	is a nuclear medicine method to investigate the function and shape of organs. With direction sensitive scintillation counters the distribution of radio nuclides in the body tissue is measured and displayed on a monitor. From that the doctors come to conclusions about structual changes e.g. in the brain, in the thyroid gland or in the liver.
Electrocardiagrams (ECG)	are records of the electrical activities of the heart. There are charges between excited and unexcited muscles, the time distribution of which is measured.
Electro-Encephalograms (EEG)	are plots of the electrical signals of the brain, which has its own rhythmical activity when awake or asleep.
Neurophysiology	is the investigation of the functions of the nervous system.
Otolaryngology	is the branch of medical science combining the fields of laryngology and otology.
Spectrometry	is a method of measuring interactions between electromagnetic radiation and matter. Different energy transfers caused by absorption or emission are found in different ranges of frequency. Nuclear reactions are due to high energy γ-rays, electronic transitions in the inner shells or between valence electrons are related to the low energy γ's. If energy is lower, spectra from UV- and visible light are used to identify elements. Changes in the vibrational and rotational energy caused by absorption are observed in the IR-range. In the microwave range there are nuclear resonance

absorption phenomena in strong magnetic and rf-fields.
Spectrometrical methods are used to investigate the structure of fluids, crystals and metals.

Gas-Chromatography

is a method of analysing gases and vapors. The components are separated by different solubility coefficients, and they are identified by measuring the thermal conductivity or the gas density.

2.7 Applications Literature

(A1) CAMAC: Specification of Amplitude Analogue Signals within a 50Ω System, Commission of the European Communities, Report EUR 5100 e, Luxembourg 1974.

(A2) W.K.B. Sie, J.N.T. Potvin; MEDAC-CAMAC - A CAMAC System for Medical Data Acquisition and Control, CAMAC Bulletin No.7, July 1973, p.8.

(A3) W.T. Lyon; Let's Standardize our Computer Problems, presented at IEEE Industrial Applications Society Annual Meeting, Milwaukee, Wisconsin, Oct.10, 1973.

(A4) F.G. Willard; Interfacing Standardization in the Large Control System, presented at IEEE Industrial Applications Society, Annual Meeting, Pittsburgh, Pa. Oct.8, 1974.

(A5) T. Friedle, W. Heep, W. Stiefel; CAMAC-Meßwerterfassungssystem, Kernforschungszentrum Karlsruhe, KFK 1813, Juni 1973.

(A6) A.M. Deane, C. Kenward, A.J. Tench; A CAMAC-Based Data Processing System: LABCOM, CAMAC Bulletin No.5, Nov. 1972, p.11.

(A7) E.G. Kingham, R.E. Martin; A Central Data Acquisition and Processing System, Proc. of the First Intern. Symp. on CAMAC in Real-Time Computer Appl., Luxembourg, Dec.4-6, 1973, CAMAC Bulletin No.9, Suppl., April 1974, p.129.

(A8) B. Zacharov; CAMAC as a Processor Interface in Computer Networks, Proc. of the First Intern. Symp. on CAMAC in Real-Time Computer Appl., Luxembourg, Dec.4-6, 1973, CAMAC Bulletin No.9, Suppl., April 1974, p.179.

(A9) H. Verelst; A Multi-User Data Network for Communication Between Computers, CAMAC Bulletin No.10, July 1974, p.7.

(A10) A.C. Peatfield, H.J. Sherman, A.J. White, B. Zacharov; Microwave Data Link for Computer Communication, Proc. IEE, 120 (1973) 186.

(A11) CAMAC Serial System GO, News Item in CAMAC Bulletin No.11, Nov. 1974, p.20.

3. How to Program CAMAC?

3.1 Introduction

In order to solve a problem by computer it is usual to write programs in a higher problem-oriented language (source language) and to translate these programs into the machine language of the computer (object-language). This is done by a special translation program stored in the computer memory.

A program consists of a finite series of commands. The translation program generates a finite series of machine commands from each command written in the higher language.

There are three types of translating program,
- compilers, translating problem-oriented programs into machine programs,
- assemblers, translating machine-oriented programs into machine programs,
- interpreters, translating commands written in the source language into machine language instructions at run-time.

Many features of CAMAC software were fixed when the CAMAC hardware was defined. A CAMAC language must take into account the CAMAC-concept which contains essentially I/O-functions. But if one is generating an I/O-language for typical CAMAC applications one would not deal with all aspects of the possible real time features, because for this the language needs also data handling elements.

To define such a complicated language would take a long time. Therefore it was decided to generate a language at a lower level, matched to the CAMAC hardware and therefore similar to an assembler language.

This language is called IML. It is implemented as a parasite language in the environment of a host-language. There are some critical links in this two level concept but normally the communication paths between the host and the parasite language are well defined.

IML will be described in a short form in the next chapter, after that we will discuss some other possibilities for programming CAMAC.

3.2 Short Description of CAMAC-IML (S1)

Operations of the CAMAC hardware are defined by commands which start, control and stop the desired operations. Certainly only a few CAMAC users are able to describe all details of the logical functions of the crate controllers or system controllers used in the system in order to generate their own series of commands. Therefore we must define CAMAC-statements which are independent of the special apparatus, so that the users are able to program CAMAC activities without knowledge of the details of the controllers. These statements are the essential content of the CAMAC-IML software. The examples given here are expressed in the Macro syntax IML-M1, which is only one of many possible syntaxes for IML.

The statements may be divided into two general classes:
- declaration statements and
- action statements.

3.2.1 Declaration Statements

Declarations of this type can be given about devices, demands, memory references, and computer channels.

3.2.1.1 Device Declarations LOCD

A register implemented in CAMAC is identified by its address consisting of the branch, crate, station number, and the subaddress. If the register is located in branch 1, crate 2, station 1 and subaddress 0 and if it is called REG, then we can describe it as

REG = B(1)C(2)N(1)A(0)

This can be declared as

LOCD REG, H, 1, 2, 1, 0

H means a direct hardware address (address constant). It is also possible to have an indirect (pointer) addressing mode (address variable) which has the advantage that one can change the CAMAC address during the run.

Devices may also be declared as an element of a hardware array. Three types of arrays are allowed
- a random distribution
- a regular distribution and
- an address scan array.

Arrays can consist of scalers whose contents can be read in the address scan mode, or there may be a buffer the data of which should be read in the stop mode (see Chapter 3.2.2.2).

3.2.1.2 Demand Declarations LOCL

The sub-address of a LAM-source which is generating a demand within a module can be detected in two different ways (see Chapter 1.5.1 to 1.5.3), either by scanning the sub-addresses with the appropriate test function and looking for Q = 1, or by reading the content of the LAM-status register in the module. With two forms of declaration we are

able to have identical statements for the access to the demand. The LAM-grader module in the crate (see Chapter 1.5.2) generates a graded-L word from the individual LAM-signals.

If, for example, ALPHA is the name of a demand connected to the graded-L number GL12 coming from a LAM source associated with a subaddress, this can be declared in IML as:

```
LOCL ALPHA, SUB, 1, 2, 3, 4, GL12
```

The source of demand ALPHA is accessed at subaddress 4 at station 3 of the crate 2 which is in the branch 1.

If, for example the GL-number is 14 associated with the demand BETA, and the source is accessed through bit 7 of the status register in the same station as above, the local declaration would be:

```
LOCL BETA, BIT, 7, 1, 2, 3, GL14
```

3.2.1.3 Memory Reference Declaration LOCM

Memory references must be declared in IML too, and must conform to the host programming system. IML needs to declare constants, variables, arrays, buffers and the wordlength required to hold the data.

A local memory reference of a variable of 24 bits into the memory cell VALUE is:

```
LOCM VALUE, V, 24.
```

3.2.2 Action Statements

There are three general types of action statements:
- single action statements,
- uni-device block transfer action statements,
- multi device action statements.

Single actions execute one CAMAC cycle and transfer one data word if the CAMAC function code is READ or WRITE. They do not transfer CAMAC words if the function code is in the OPERATE- or READ STATUS group, because these are dataless control or test functions.

An uni-device block transfer moves a block of data between a single CAMAC address and a buffer area or array in computer memory, or vice versa. This can be done via an autonomous hardware channel or by single actions under program control (software channel). The transfer is used with functions READ and WRITE and is normally terminated if the repeat count limit is reached or if the module sends Q = 0.

Multidevice actions are a series of single actions executed at different CAMAC addresses, using the same function code. Transfers are done with functions READ or WRITE between CAMAC and computer memory, with one data word from or to each CAMAC address.

3.2.2.1 Single Action Statements SA

They carry out one CAMAC cycle and can use any function code. If one wants to read the content of a 16-bit register REG into the memory location VALUE, then the associated statement is

```
SA RD1 REG VALUE
```

If the register REG is located in branch 1, crate 4, station 2 and subaddress 0, it will have been declared by

```
LOCD REG, H, 1, 4, 2, 0
```

If VALUE is a 16-bit variable it will have been declared by

```
LOCM VALUE, V, 16
```

The mnemonic RD1 means READ Group 1 Register (Function code 0). If the same register is to be cleared, the statement is

```
SA CL1, REG
```

where CL1 means Clear Group 1 Register (Function code 9). Single actions which are followed by a program branch depending on the state of Q are provided by two statements called SJQ (single, jump on Q) and SJNQ (single, jump on not Q). They are actions in which the program branches to the given statement label if the CAMAC-operation returns Q = 1 or Q = 0 respectively. They can be used with functions of the type READ, WRITE or OPERATE.

3.2.2.2 Uni-Device Block Transfer Action Statements UB

In CAMAC systems block transfers can be controlled by using the Q-response from the module (see Chapter 1.5.1) in the following three modes:
- the address scan mode, in which the registers within a module are arranged in such a manner that sequential access is possible. The scan starts within a module at subaddress A(0) and stops at A(n). The response Q = 1 means that the addressed register exists, Q = 0 means that the previous address was the last in the module, and that the scan should resume at A(0) in the next station.
- the repeat mode, in which the registers are not necessarily ready for data transfer at any time. If there is a read command when the register is not able to transfer, it will send Q = 0. That means the controller should repeat its command at a later time. It may require several attempts before the register sends Q = 1. This mode is attractive if the ready signal is expected to come within a few computer cycles, otherwise too much time is lost. In this mode Q = 1 means that the register is able to transfer, Q = 0 means it is not possible at that time.
- the stop mode, in which the data transfer is terminated by a block-end signal from the module. Q = 1 means the data word is within the block, Q = 0 means the end of the block.

Also, with respect to synchronizing the transfer we can distinguish three modes:
- the UBR-mode (repeat while not Q) is synchronized by the use of Q in the repeat mode,
- the UBC-mode (controller synchronized), in which the transfers are synchronized by the system controller. This can be used when the transferrate available in the module is faster than that demanded by the controller,
- the UBL-mode (LAM synchronized) in which the module sets a LAM-signal to request each individual transfer.

If for example a sequential transfer operation has to be set up to read characters from a teletype into a memory array called BUFFER, we must declare how many words should be transferred and what is the channel to be used. As an example we assume that 48 words (repeat count) should be read via channel 1 every time a LAM-signal indicates a demand called GAMMA. The channel is declared as

```
LOCC CH1, REP48, TALLY1, STATUS, 1
```

where CH1 is the channel name
REP48 is the repeat count
TALLY is a variable giving the number of transfers actually achieved
STATUS is a variable giving the reason for termination
1 is the physical channel to be used.

The action statement is then written as

UBL RD1, TTYPE, BUFFER, GAMMA, CH1

3.2.2.3 Multi Device Action Statements

There are three statements used in multi device actions, which handle a series of single actions related to the same function code:
- the basic MA-mode which is used together with the functions of the type READ, WRITE and OPERATE. As with the block transfer modes we need to define the number of words which should be transferred (repeat count) and the channel which is to be used.
- the MNQ-mode (repeat while not Q) which is used in multi device test actions. After the first operation which returns Q = 1 the action stops. This mode is associated only with dataless control functions, typically to search for the source of a demand.
- the MAD-mode which implements the multi address scan (see Chapter 3.2.2.2) used with READ or WRITE functions.

3.2.3 LAM Action Statements

LAM actions are always single actions, by which one can enable, disable, test and clear the LAMs. This is done by translation into the specific function codes defined in the CAMAC specifications (H1), appropriate to the SUB or BIT declaration of the LAM.

3.2.4 System Action Statements

These statements define general functions in crate controllers, branch or serial drivers and system controllers. They are related to operations like INITIALIZE, CLEAR, INHIBIT, ENABLE or DISABLE DEMANDS and some test functions. The actions are always single actions.

3.3 Other Possibilities for Programming CAMAC

IML is a language which must be embedded in a host language to handle data processing. The result is a host-parasite language which is translated in a compiler or assembler (S2). But IML is not the only possibility.

3.3.1 FORTRAN-Subroutines

Another way to program CAMAC systems is to describe CAMAC operations as standard subroutines in a high level problem-oriented language like FORTRAN. This language is the best-known and most-used language for solving scientific and technical problems. The advantages in using standard subroutines are that the users are familiar with the language, that there are compilers for nearly all computers, that the subroutines are core-efficient because they require only one copy of a subroutine, however many times it us used in a program, and that the communication paths between the subroutine and the main program are well defined.

But there are disadvantages too. Programs written in a high-level language are slower than those written in assembler language. Therefore subroutines for CAMAC operations should be as standardized as possible to shorten the execution time. Most subroutines or procedures can be written in a time-saving way, but nevertheless they can be very transparent, e.g. to allow the block transfer modes (address scan-, repeat- and stop-mode).

A suitable set of standard subroutines are specified by the US-NIM-Software Working Group (S3, S4, see also some associated aspects in S5). The subroutines should be used in FORTRAN or FORTRAN-compatible languages.

Certainly most subroutines must contain some values characteristical of the system in use, e.g. how many computer words are needed to transfer a 24-bit CAMAC word, how many modules are housed in the crate and how many crates are in the branch. For that, all subroutines have a common block CAMAC-COMMON (CMCCOM) in which these parameters are defined.

LAM signals can be used for starting interrupt service routines in three different ways:
- null-mode, in which no LAM signal is used in the subroutines, with the exception of LAM test commands,
- event-mode, in which the parameters to execute the routine are present, but only the arrival of the LAM signal starts the routine,
- step-mode, which is similar to the event-mode with the exception that before execution of each command the status of the LAM is tested and the next step will be carried out only if the LAM signal is true.

Standard subroutines contain the describing parameters:
- function F,
- branch B,
- crate C,
- station N,
- subaddress AD,
- number of words LN transferred in a block,
- data block DATA,
- value of the Q-signal,
- frequency NEX of executing the function at Q = 1,
- items about possible timing- or address errors ERRORA,
- content of the graded-L word GLNO, giving the address of the LAM source (see Chapter 5.2).

Fundamental subroutines are
- CALL CMCINT (BL,GLNO) which stops the task until the LAM status in the branch is set. After that the return from the subroutine follows.
- CALL CMCBSC (F,B,C,N,AD,LN,DATA,ERRORA)
  CALL CMCBCE (F,B,C,N,AD,LN,DATA,ERRORA,BL, GLNO)
- CALL CMCBCS (F,B,C,N,AD,LN,DATA,ERRORA,BL, GLNO)

These three subroutines execute a single CAMAC operation related to a single address once or several times depending on the LN-value. The three subroutines are distinguished by the LAM-modes described above.
- CALL CMCSEQ (F,BA,CA,NA,ADA,LN,DATA,Q, ERRORA)

This subroutine which has also three versions with respect to the LAM-mode, describes a sequential operation in a array A which is the same at all addresses within A, e.g. READ or WRITE in the address n (BA(n)CA(n),NA(n) and ADA(n). The number of elements in the array is

defined by the LN-value.

In addition to these fundamental routines there are some which start general functions, e.g.:

- CALL CMC BZ(B) which starts the branch-INITIALIZE (BZ)-signal in branch B (see Chapter 1.5.2)
- CALL CMC ENL (BL,GLNO)
- CALL CMC DL (BL, GLNO)

which enable or disable the individual LAM signals referred to the graded-L word GLNO in the branch B.

Subroutines which carry out CAMAC block transfer operations in the address scan, repeat and stop modes (see Chapter 3.2.2.2) are described, e.g. an operation in the repeat-mode:

- CALL CMC RPT (F,B,C,N,AD,LN,DATA,ERRORA,BL, GLNO)

which is the form in the event-mode, started by a LAM signal coming from branch B with the graded-L word GLNO.

The run time of the program can be utilized more economically if one splits the subroutines in two parts. The first part combines and checks the parameters by declaration procedures; the second executes the procedures. This method should be applied especially in the set-up and test phase because the user-system dialogues running at that time are very useful but need time which should be reserved for the real time phase of the experiment.

3.3.2 BASIC-Subroutines

Subroutines can be written also in BASIC and FOCAL because they are easy to learn and have some similarity to FORTRAN.

BASIC as a host language together with PAL-11 as a assembler language is often used in the PDP-11 computers, today's wide-spread process computer family, because there are not only BASIC compilers but also a powerful set of arithmetical instructions to handle real time applications. BASIC is really not a language for real-time, but the PDP-11 version can call the external function routine EXF which is used as a link between BASIC and PAL-11. The general form of this EXF-routine is:

LET U = EXF (parameters P_i)

The parameters are transmitted from the BASIC statements to the assembly routine; their meaning is given by the user.

An example (S6) assumes that the first parameter describes the type of action, e.g. if it should be a dataless control operation or a read or write operation. The other parameters are the values of branch (B), crate (C), station (N), subaddress (A) and function code (F) which are contained in the CAMAC command. But, similar to the other languages used for CAMAC and mentioned above (IML and FORTRAN), it is necessary to state the length of the dataword, the block transfer mode and the number of words to be transferred.

A typical flow chart of a routine is shown in Fig. 3.1. If the first parameter is T = 0, then it will be a dataless control operation, if T = 1 a directly addressed data transfer, T = 2 an indirect addressed data transfer, T ≥ 3 is reserved for special functions, T = 13 means the return from the subroutine.

The next parameters are the hardware addresses and the function code. If a data operation is required, the parameters L, M and K are needed to define the length of the data word, the block transfer mode and the word count, before the subroutine READ or WRITE is started.

For all types of operation a BASIC variable Q is automatically defined and given the value of the CAMAC Q response.

After execution of the CAMAC-EXF routine there is a return to the BASIC main program.

Additional examples of this type are given in (S7,S8,S9).

In CAMAC applications, interrupts are handled mostly in assembly language routines. Before the interrupt starts the routine, the contents of the program counter and the processor status register are automatically saved in a memory stack from where they will be recovered after the return from the routine.

If BASIC is used as a host language, one should notice the different froms of the interpreter, e.g. the CALL-statements are different and not implemented at all. Several software groups are writing interpreters in a preprocessor form, and also implementing interrupts in BASIC.

Routines similar to those we have mentioned above, but written in FOCAL, are described in (S10,S11,S12). They are used mostly with the 12-bit minicomputers of the PDP-8 family.

3.3.3 CASIC Macros

I/O-routines can be written not only as subroutines but also as macros, which have the advantage of time saving at program run-time. In both cases the routine uses the same instructions each time it is used, but the operands are normally different. Thus special parameters must be inserted in the subroutine or in the macro command. When using subroutines this is done during the run time of the program, but with macros it is done during the compilation.

Subroutines are core efficient with respect to the users program but they need additional memory cycles to hand over the parameters from the main program to the subroutine. The use of macros saves program run-time because the additional cycles are not needed, but it wastes more memory bits. If the users program contains often-used sequences which are relatively long, then it is better to use subroutines rather than macros.

The Schlumberger Software Group (S13) has developed a BASIC extension for use with CAMAC. This language, called CASIC, provides all the standard statements of BASIC plus a set of statements for CAMAC purposes. It is designed to work with a PDP-11 computer. Because it is written as a macro instruction generator the execution speed is improved by a factor of ten, compared with conventional BASIC. CASIC is also able to handle interrupts.

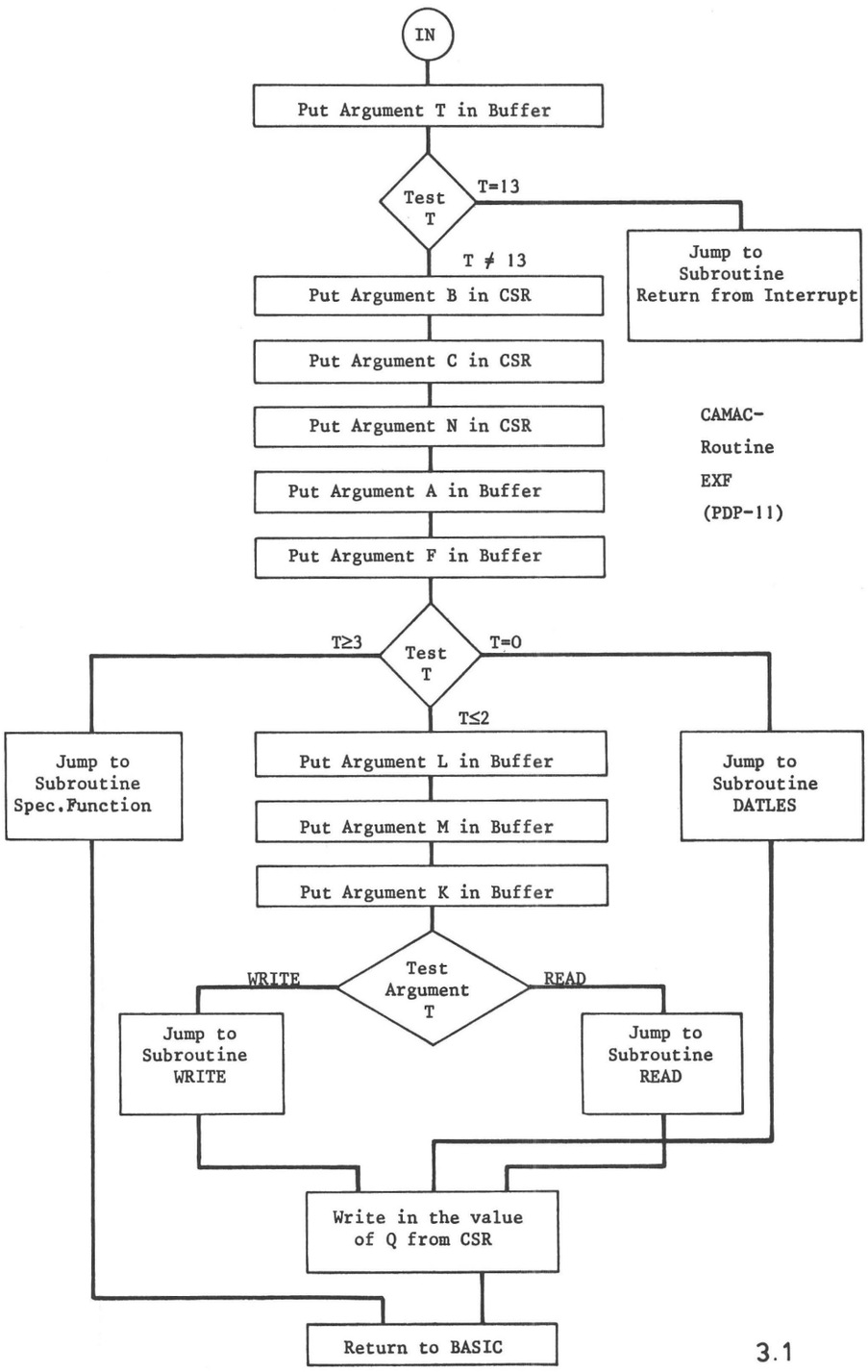
CASIC is described by naming statements, CAMAC operation statements and LAM-handling. We will give here a short survey of CASIC.

If CAMAC registers are declared by a name, the LET-statement can be written in the form:

LET name = STA CAMAC address

The name may consist of 1 to 3 alpha-numeric characters which are preceded by the % character. The CAMAC address contains the branch, crate, station number and the subaddress. In an example we can declare the register named R5 which is addressed by A=4, N=17, C=3 and B=2, in the following way

LET % R5 = STA(2,3,17,4).


IN
Put Argument T in Buffer
Test T
T=13
Jump to Subroutine Return from Interrupt
T ≠ 13
Put Argument B in CSR
Put Argument C in CSR
Put Argument N in CSR
Put Argument A in Buffer
Put Argument F in Buffer
CAMAC-
Routine
EXF
(PDP-11)
Test T
T≥3
T=0
T≤2
Jump to Subroutine Spec.Function
Put Argument L in Buffer
Put Argument M in Buffer
Put Argument K in Buffer
Jump to Subroutine DATLES
Test Argument T
WRITE
READ
Jump to Subroutine WRITE
Jump to Subroutine READ
Write in the value of Q from CSR
Return to BASIC

3.1

In a CAMAC data operation the action statement is given by the following general LET-statement:

```
LET dest = CAM (source, function code).
```

In the case of write operations 'dest' means the name of the register to be written into. In the case of read operations 'dest' is a variable which is given the value of the result of the read operation.

'Source' is the name of the register to be read or, in write operations, a variable or constant of the program.

'Function code' is related to the CAMAC specification, i.e. read operations are connected with F(0) to F(7), write operations with F(16) to F(23).

Two typical statements are:

```
LET % R5 = CAM(32,WRITE)

LET % R0 = CAM(%R5,READ).
```

In the first example, the number 32 is written into the register R5, in the second the content of register R5 is read by register into R0.

Dataless operations are defined by the LET-statement:

```
LET CAM (source, function code)
```

where 'source' is the controlled register. The function code for control operation is F(8) to F(15), and F(24) to F(31) (see definition of the CAMAC function codes in (H1)). To test the status of a LAM source in register R5, the function code is F(8) and the statement is written as:

```
LET CAM(%R5,8)
```

LAM handling is done by two different statements, one for starting the service routine, and another one to test if the LAM from a register has been received and processed or to wait for it if this has not occurred. The two statements are:

```
ON LAM CAMAC station THEN GO TO xxx

WAIT CAMAC register
```

The service routine started by the interrupt must be completed with a Return from Interrupt (RTI) statement.

Another macro instruction generator has been developed at the Daresbury Laboratory, U.K. for use with Honeywell 316 and 516 computers. It is described in (S14).

3.3.4 CAMAC-Monitor

A monitor written by EGG/ORTEC, Oak Ridge, Tenn., USA for use with the PDP-11 is called COMP11. It accepts and executes CAMAC-like commands from the teletype and is written in PAL-11 (S15). With COMP11 the user should be able to send a single CAMAC command via branch driver and crate controller CCA into an addressed module where it is executed and the user can observe the result. After each executed command, COMP11 returns control to the user.

COMP11 assumes only 4k of core and no peripherals other than the teletypewriters and the branch driver. The communication between the main program and the CAMAC drivers makes use of the TRAP instructions which are elements of the PDP-11 instruction set. This enables various subprograms like:

- forming a CAMAC instruction and loading it into the branch driver,
- checking error bits in the control and status register and printing-out the errors found,
- loading crate and word count registers,
- loading and printing the content of data registers,
- checking which crate controllers are on-line,
- forming a list of checked functions which can be executed a number of times.

Monitor programs like COMP11 are easy to learn, they are very useful in the test and set-up phase of process control systems. They can also be used to demonstrate CAMAC modules.

The list of software possibilities and successful applications is much longer than those described in this paper. Here we can only give you a limited survey to trigger your ideas and start CAMAC actions.

3.4 Glossary of Software Terms

A program	is a sequence of step-by-step instructions which declares to the computer precisely, how the problem should be solved.
An instruction	is a coded program step which gives the computer the details of the next operation. The instruction contains one or more addresses to specify the source of operands, the destination for results, and the source of the next instruction.
Machine languages	define instructions by reference to the individual internal structure of the computer. Machine instructions are in binary code but few users could write instructions in this form.
Assembly languages	represent the address and operand part of the instruction word by easy-to-learn alphanumerical symbols. The symbolic addresses are converted to absolute and relative addresses by a machine-specific translation program, the assembler. Typically each symbolic instruction corresponds to one machine instruction, this is called an 1:1 relation. Assembler programs are normally provided by the computer manufacturer.
Problem-oriented languages	are the so-called high level program languages. They are defined as much as possible with respect to the problems which are to be solved. Their symbols are taken from elements of mathematical formulas and from the colloquial language. Computer manufacturers usually provide translation programs, called compilers, for the popular problem-oriented languages, e.g. for FORTRAN, ALGOL, PL/1, COBOL, BASIC.
A compiler	translates the instructions written in the high level source language into instructions written in the object language. If the object language is identical with the machine language, one needs a direct compiler. If the object language is the assembly language of the computer, then one needs another translation before executing the instruction.
An assembler	translates the source language into the machine language in an 1:1 relation ship.
An interpreter	translates the source language, or a special stored version of it, into machine language, instruction-by-instruction at run-time.
Subroutines or procedures	are used to define operations which are used frequently in the same main program, but which are formulated only once. These subprograms are generally called procedures in high level languages, subroutines in lower level languages. They are given a name by which they can be called at any time from the main program. One has to distinguish: - open subprograms, which are programmed only once but can be used in other program sequences at any time - closed subprograms, which are stored in the memory only once and can be called by any program as much as needed. Procedures and subroutines are called by a special jump instruction at that line number of the main program where they should be appended. After execution of the routine there is a jump back to the line number from where the routine has been started and the main program will be continued.

Macro instructions

are single source-language instructions used in the main program to call a group of computer instructions. Macro instructions therefore are translating in an 1:n-relation rather than the 1:1 relation used in assemblers. During the translation process the assembler replaces each occurrence of a macro instruction by a separate copy of the object code instructions.

An implementation

is a realization or execution of a hardware and/or software system. A particular system may implement various ideas, principles and standards.

3.5 Software Literature

(S1) CAMAC, the Definition of IML, a Language for Use in CAMAC Systems, ESONE/IML/01, October 1974, Distributed by ESONE Secretariat, Dr. H. Meyer, CBNM-CRC, Steenweg naar Retie, B-2440 Geel, Belgium.

(S2) W. Kneis; Implementation of CAMAC Intermediate Language (IML) in an Assembler Language Environment, CAMAC Bulletin, No.10, July 1974, p.28.

(S3) R. F. Thomas Jr.; Some Aspects of CAMAC Software, IEEE Trans. on Nucl. Sci., Vol.NS-20, No.2, April 1973, p.50.

(S4) R. F. Thomas Jr.; Specifications for Standard CAMAC Subroutines, CAMAC Bulletin, No.6, March 1973, p.23.

(S5) I. Michelson, H. Halling; Procedure Calls - A Pragmatic Approach; Proc. of the First Intern. Symposium on CAMAC in Real-Time Computer Appl., Luxembourg, Dec. 4-6, 1973, CAMAC Bulletin, No.9, Suppl. April 1974, p.63.

(S6) I. Bals, E. de Agostino; An Extended BASIC Language for CAMAC-Programming, CAMAC Bulletin, No.7, July 1973, p.25.

(S7) H. Halling, K. Zwoll, W. John; CAMAC - Overlay for Single-User BASIC-Modification of 8-User BASIC for the PDP-11, CAMAC Bulletin, No. 6, March 1973, p.15.

(S8) B. Becks, H. Halling; Translating CAMAC Test Programs from Interpretive to Assembled BASIC, CAMAC Bulletin, No.10, July 1974, p.27.

(S9) E. M. Rimmer; CAMAC and Interactive Programming; Proc. of the First Intern. Symp. on CAMAC in Real-Time Computer Appl., Luxembourg, Dec. 4-6, 1973, CAMAC Bulletin, No.9, Suppl., April 1974, p.69.

(S10) F. May, W. Marschik, H. Halling; A FOCAL Interrupt Handler for CAMAC, CAMAC Bulletin, No.6, March 1973, p.21.

(S11) K. Zwoll, E. Pofahl, H. Halling; PDP-8 Operating System for Non-Time-Critical CAMAC Experiments, CAMAC Bulletin, No.5, November 1972, p.28.

(S12) F. May, H. Halling, K. Petreczek; FOCAL Overlay for CAMAC Data and Command Handling, CAMAC Bulletin, No.1, June 1971, p.18.

(S13) I. M. Servent; CASIC, A CAMAC Extended BASIC Language, Proc. of the First Intern. Symp. on CAMAC in Real-Time Computer Appl., Luxembourg, Dec.4-6, 1973, CAMAC Bulletin, No.9, Suppl., April 1974, p. 97.

(S14) F. R. Goulding, A. C. Peatfield, K. Spurling; CAMACRO - An Aid to CAMAC Interface Programming, CAMAC Bulletin, No.5, November 1972, p.29.

(S15) R. M. Keyer; COMP11, A CAMAC Oriented Monitor for the PDP-11, CAMAC Bulletin, No.7, July 1973, p.27.