GENERAL-PURPOSE CAMAC SOFTWARE
FOR HP 2100 SERIES COMPUTERS

GENEVA
1978
GENERAL-PURPOSE CAMAC SOFTWARE FOR HP 2100 SERIES COMPUTERS

E.M. RIMMER
DD Division
CERN, 1211 Geneva 23, Switzerland

ABSTRACT

FORTRAN/BAMBI callable software has been written to drive CAMAC using an HP 2100 series computer. (BAMBI is a CERN-written extension of BASIC.) Two operating environments are catered for, the preliminary test and debug phase, when CAMAC functions are executed in-line in the main user program, and the data acquisition phase, when data are accumulated into buffers via DMA under CAMAC interrupt control and are available in the main program as events for monitoring and analysis.

1. INTRODUCTION

The On-Line Computing Group of the CERN Data Handling Division presently owns about 50 HP 2100 series computers, all having 32K minimum memory but only 10 with discs. These HPs are in an equipment pool and are loaned to users for varying periods of time to test hardware, perform experiments and, in some cases, to analyse data tapes.

The aims of the On-Line Group are:

a) to give modular software support with packages which can run in all of our HPs; minimal hardware configurations use a sub-set of the facilities available on larger installations;

b) to use HP-written software whenever possible and to concentrate effort on packages not provided by the manufacturer (CERN-written routines all have names beginning with Z to reduce conflicts);

c) to allow the user to program for his particular application in FORTRAN or BAMBI (a CERN-written version of BASIC which has FORTRAN-compatible subroutine calling sequences, amongst other extended features); the software modularity provides sufficient flexibility for use in widely differing applications;

d) to keep as much compatibility as possible with utility libraries available on the CDC and IBM machines at the CERN Computer Centre;

e) to employ only five or six programmers to provide this support.

To achieve this, we have packages for histogramming, graphics, data tape handling, etc.\textsuperscript{1}, which have FORTRAN calling sequences compatible with similar packages at the Computer Centre, for example HBOOK. On the hardware side we have taken advantage of the widespread use of CAMAC to provide standard I/O facilities suitable for most requirements.

In this talk I will describe the CAMAC software available to HP users. We have two CAMAC software packages; one for performing individual CAMAC operations, CAMAC calls, and one for fast data acquisition. The packages are supported for three hardware interfaces in a manner which is more or less transparent to the user program:
HPC C 172 (BORER 1531) Single Crate Controller  
BORER 2201 Branch Driver  
CERN 7529 Branch Driver.

Work is currently in progress to implement the software also for the GEC-Elliott System Crate Interface, the PTI-21.

I will assume that you are familiar with CAMAC and with the differences between the execution of a compiled FORTRAN program and of an interpreted, interactive BASIC (RAMBI) program.

2. **HP BASIC CONTROL SYSTEM, BCS**

The CAMAC software runs in the HP Basic Control System, BCS. This is a minimum I/O system and is therefore relatively fast and occupies very little memory. It has the disadvantage that it provides no program development facilities. However, it is a simple and static system, giving us a good chance to locate (user) hardware errors in actual applications and avoiding the problems of incompatibility with system updates. Also we can run the same software on ALL installations, small or large, ensuring that it is eventually debugged and minimizing the manpower requirements for support.

3. **HP REAL-TIME EXECUTIVE, RTE**

RTE is a typical disc-based, real-time, multi-tasking system which we use for program development. We have a CERN-written relocating cross-loader, ZXL, which prepares absolute (core image) programs for the target BCS machines from RTE relocatable binary disc files. The absolute programs can be stored as disc, magtape, cassette or paper-tape files.

At installations with a disc, the RTE and BCS machines are one and the same, and we have an RTE/BCS swopping program which uses disc files. For non-disc installations, users work at our central program development machine and either transport BCS programs to their local machine on magtape, for example, or down-load the program via an HP serial link, using HP software, from the RTE central machine into their BCS satellite. We are moving towards having mini-centres located in the big CERN experimental areas, each with a substantial RTE installation linked to several BCS satellites serving different particle beam-lines; however lack of money and manpower are the main obstacles to this development at present!

For complex data acquisition systems, we use two computers, an RTE machine linked to a BCS machine. The fast data acquisition and tape-writing take place in the BCS machine as usual and the RTE machine, in addition to providing program development and down-loading facilities, is used also for running user monitor programs which obtain data samples from the BCS machine via the link. This arrangement has the advantages that correct operation of the user monitor program is decoupled from that of the data acquisition program, which can stay minimum and stable, and also that the data throughput to tape is not influenced by changes in size or speed of the user's "on-line" program.
4. **BAMBI VERSUS FORTRAN**

The CAMAC software consists of subroutines identically callable from a user main program in BAMBI or FORTRAN.

**BAMBI** is used when interactive programming is needed, for example, for small short-lived test programs. Execution speed is irrelevant to these applications and programming flexibility is important.

**FORTRAN** is used when high-speed, memory-efficient program execution is needed, for example, for data acquisition, storage and monitoring.

Many users start by writing their application program in BAMBI even if it is eventually to be in FORTRAN, because of the ease of the debugging program logic in an interactive environment. We provide a **BAMBI - FORTRAN** translator for the final program. The memory released by removing the BAMBI interpreter (approx. 8 kwords) can then be simply absorbed as data or histogramming buffers, or used for more FORTRAN code.

5. **THE 'CAMAC CALLS' PACKAGE**

These subroutines operate in a non-interrupt environment and are intended primarily for hardware tests and debugging. They occupy approximately 1.2 kwords of memory.

The subroutines generate (B)CAMAF commands and transfer data, Q-responses, fault diagnostics, etc. CAMAC interrupts are not directly handled and must be monitored by program loops containing F8 (Test LAM) functions.

The simple form of a call is, for example:

```fortran
CALL ZCRPI(C,N,A,F,D,Q)
```

- **C** = crate number
- **N** = station number
- **A** = module sub-address
- **F** = function code
- **D** = data, if any
- **Q** = returned Q-response.

The data, **D**, can be real, integer, or double precision (indicated by the 3rd character of the routine name, **R**, **I** or **D**). It can be treated, at the module level, as a binary word, an array of bits for pattern units, or as a BCD-coded word (indicated by the 5th character, **1**, **2** or **3**).

Branch selection is made separately:

```fortran
CALL ZCSBS(B) where B = the branch number.
```

Similar routines implement the standard CAMAC block modes, with either program controlled or DMA data transfer, for example:

```fortran
CALL ZCRPE(C,N,A,F,D(i),Q,W)
```

- Q-end mode block transfer
- Program controlled data transfer
- Real data
D(i) = real data array
W = given as maximum word count (the dimension of D)
    = returned as the actual number of words transferred.

CALL ZCRDE(C,N,A,F,D(i),Q,W)

As ZCRPE but with DMA data transfer.

Here is an example of a BMBI program to histogram and display an ADC reading, which illustrates the ease of producing application programs; you will notice that it consists mainly of remarks!

ICOM C,N,A,F,D(Q),IHIST(520),IE
5 REM : THE ICOM STATEMENT DECLARES INTEGER VARIABLES
10 REM : INITIALIZE CAMAC, BRANCH 2 ET CETERA.
15 ZCSBKZ
20 C=1
21 N=1
25 A=0
30 REM : REMOVE DATAWAY INHIBIT
35 ZCIP1(C,30,9,24,D(Q)

40 REM : INITIALIZE HISTOGRAMMING
45 ZHRES(IHIST,520,1,IE)
50 ZHBL1(1,1,"ADC1",512,0,1,IE)
55 REM : INITIALIZE GRAPHICS
60 ZHFM1(1,1,1)
65 ZGLUN(2)

70 REM : CHECK FOR EVENT
75 ZCIP1(C,N,A,8,D(Q)
80 IF Q=1 THEN 200
85 REM : CLEAR GRAPHICS SCREEN IF Hp FRONT PANEL SWITCH 1 IS SET
90 IF ISSW(2) THEN NEWP
95 REM : DISPLAY HISTOGRAM IF SWITCH 0 SET
100 IF ISSW(0)=1 THEN ZHFL1(1,2,IE)
105 GOTO 70

200 REM : EVENT INPUT (LAM SET)
215 REM : READ DATA, CLEAR ADC AND LAM
220 ZCIP1(C,N,A,0,D(Q)
225 ZCIP1(C,N,A,9,D(Q)
230 REM : HISTOGRAM DATA
235 ZHFL1(1,D)
240 GOTO 85
999 END
6. THE "CAMAC DATA ACQUISITION" PACKAGE

These subroutines and device handlers\textsuperscript{,1,2} operate in a CAMAC interrupt environment and are intended primarily for high-speed data acquisition, buffering and storage. They occupy between 3.5 and 5 kwords of memory approximately, depending on the facilities used.

The BCS program contains only two modules which are application-dependent, the user main program and the so-called Experimental Descriptor.

The main program is written by the user in FORTRAN or BAMI. Calls are made to subroutines for run control and data retrieval. A minimum of three calls is sufficient to control the system (a complete list of available calls is given in Appendix A).

\begin{verbatim}
CALL ZINIT to initialize hardware and software
CALL ZRUN(IH,IM) to start a data-taking run
CALL ZEVNT(IARR,ISTAT) to obtain a copy of the latest event from the input buffers and also the current status of the run.
\end{verbatim}

In the run start routine ZRUN, the user passes an array IH to be part of each magtape record header and receives a run mode flag IM indicating the data source, CAMAC or magtape. The run mode and other run options are set during a dialogue in ZRUN, which is conducted via the keyboard/display terminal. Alternatively run options can be set by calls to a subroutine (ZOPT) before calling ZRUN. A complete list of available options is given in Appendix B.

A run can be ended either by the occurrence of a particular CAMAC Graded IAM, GL, or after a fixed number of events, records or write-parity errors. The user is notified of the end-of-run in the status flag of the ZEVNT call.

In CAMAC input run mode, the CAMAC data handling tasks are table-driven at the CAMAC interrupt level and data are read into a series of buffers, used cyclically. The set of data words read when a GL is serviced is called an event. Buffers filled with events can be written as records to magtape under interrupt control. The magtape can be labelled or unlabelled. Events can be copied into a local array under user control, by ZEVNT, for monitoring and analysis. Hardware GLs can be simulated by a subroutine (ZLAM) with the GL number given as a parameter. These are called software GLs.

In replay mode, data are read into the buffers from a previously written magtape. This does not affect the logic of event retrieval for the user, because there are no specific I/O operations within the user program.

If a disc is available, it can be used in the BCS environment for tasks such as disc histogramming, program overlays, etc.

Here is an example of a FORTRAN program which can be used for data input and monitoring with either CAMAC or magtape as the data source.
C
PROGRAM FMINT
DIMENSION IH(2),IE(101)
C INITIALIZE SYSTEM
CALL ZINIT
IH(1)=2
IE(1)=101
C ZERO THE RUN NUMBER
IH(2)=0
C START A NEW RUN, INCREMENT RUN NUMBER
10  IH(2)=IH(2)+1
    CALL ZRUN(IH,IP)
C GET A COPY OF EVENT
300  CALL ZEVENT(IE,IS)
C CHECK EVENT AND RUN STATUS
    GO TO (100,200,300),IS
C COPY O.K., DISPLAY MEANINGFUL DATA WORDS
100  L=IE(2)+1
    IF (L.GT.IE(1)) L=IE(1)
    WRITE(2,1000) (IE(I),I=2,L)
1000  FORMAT("EVENT":/5110)
    GO TO 300
C RUN HAS ENDED (EXTERNALLY)
200  WRITE(2,2000) IH(2)
2000  FORMAT("END OF RUN":,I4)
    GO TO 10
END

6.1 The Experimental Descriptor

The Descriptor contains parameters, tables, etc., which describe the particular experimental environment; it is in HP Assembler code. It includes: the GL table of pointers to lists of CAMAC jobs which are performed when the corresponding GL is set either by hardware or software; the joblists themselves: the number of data buffers used, with a description and space reservation for each buffer. If the memory exceeds 32K, the excess can be simply used for data buffers, by declaring a set of mapping tables in the Descriptor. At present we do not support program mapping in the BCS environment. A disadvantage of the Descriptor is that it is in Assembler code and must be modified by the HP Editor, followed by an Assembly and cross-load. This is inconvenient for users who do not have a local disc system. We are working on software to generate and modify Descriptors in a more flexible and interactive way.
6.2 The GL Table and CAMAC Joblists

When a CAMAC interrupt occurs, the CAMAC handler reads the hardware GL pattern and merges the software GLs with it, if any. It then scans the GL table from the top (high GL number) down, providing simple priority, and interprets those joblists with pointers in table positions which correspond to bits set in the GL pattern. Each joblist contains absolute CAMAC commands, subroutine entry points, etc. When the scan reaches the end of the table, the hardware/software GL pattern is refreshed and the scan is repeated, unless and until the pattern is empty, when return is made to the mainline program at the point of interruption.

During interpretation of a joblist, data enter the buffers whenever CAMAC READ functions are executed with the DMA channel open for input. The event thus produced has a header appended giving its length and GL number. When a buffer becomes full, a DMA interrupt is produced and the DMA handler switches input to the "next" buffer, cyclically. If the next buffer is still being used for output, e.g. to magtape, the CAMAC job is suspended and control returns to the mainline program until the buffer is freed, when the job is continued.

The conventions for the contents of words in a joblist are:

The 1-word
DEC 0 is the end-of-list

The 2-word sequence
DEC -1
DEF SUBAD is JSB SUBAD, execute a subroutine

The 3-word sequence
DEC -1
DEC -1
DEC N is JMP **N in the list, where * represents the list position of the word "DEC N" and N can be +ve or -ve.

Any other value is interpreted as a valid CAMAC command and it is output to the CAMAC hardware, followed by a software wait loop (timed-out) until a hardware READY status is returned, when the next word in the list is interpreted.

Here is an example of a joblist attached to GL12, showing both the GL table entry and the use of readable mnemonics for the CAMAC operations. In this example, GL13 and GL11 are not expected and are therefore trapped by an ERROR joblist.

GL13 DEF ERR
GL12 DEF EV12
GL11 DEF ERR

EV12 DEC -1
DEF DMON.
DEF C2+N1+A0+F0 read register C2,N1,A0 into buffer via DMA
ABS C4+N6+A1+F2 read (& clear) register C4,N6,A1 into buffer
ABS C2+N10+A0+F10 reset LAM at C2,N10 (which caused GL12)
DEC 0 end-of-joblist (closes DMA)
7. TIMING

Approximate program execution times are:

CAMAC Calls: \( \sim 0.5-1 \) msec for a single-cycle CAMAC operation.
Data Acquisition: \( \sim 0.5 \) msec to handle a CAMAC interrupt,
\( \sim 35 \) usec to interpret one joblist (CAMAC) word.

8. CONCLUSIONS

With these two CAMAC software packages, callable from either Bambi or FORTRAN, we have succeeded in satisfying the needs of a wide range of users for a vast variety of applications during the past few years. The modularity and simplicity of the software as seen by the user has lead to great ease of evolution and modification of data acquisition systems, freeing the user to concentrate effort on the particular application. Also, innovations resulting from experience gained in any one operating environment have been to the ultimate benefit of all users.

* * *

REFERENCES

The address of all authors in these references is:

DD Division
CERN
1211 Geneva 23
Switzerland

1) B. Hopson, HP Program Library Catalogue, June 1978.
2) S. Cittolin, Bambi, September 1977.
4) E.M. Rimmer, ZDAQ, Short write-up, April 1978.
APPENDIX A

DATA ACQUISITION SUBROUTINES

A list is given here of the FORTRAN/BAMBI callable subroutines available for control and monitoring of data acquisition.

**Over-all control**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZINIT</td>
<td>Initialize the system</td>
</tr>
<tr>
<td>ZOPT</td>
<td>Set run options</td>
</tr>
<tr>
<td>ZRUN</td>
<td>Start a data-taking run</td>
</tr>
<tr>
<td>ZRFDG</td>
<td>Read one of the system flags</td>
</tr>
</tbody>
</table>

**Event retrieval for data sampling (with various copy conventions)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZEVNT</td>
<td>Copy data from the buffers into a user array</td>
</tr>
<tr>
<td>ZEVST</td>
<td>Copy data from the buffers into a user array</td>
</tr>
<tr>
<td>ZEVNX</td>
<td>Copy data from the buffers into a user array</td>
</tr>
</tbody>
</table>

**Analysis priority for data sampling**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZHIP</td>
<td>Set analysis priority to high, 100% sample</td>
</tr>
<tr>
<td>ZLOP</td>
<td>Set analysis priority to low, input-rate dependent sample</td>
</tr>
<tr>
<td>ZPCP</td>
<td>Set priority to obtain a minimum % valid data sample</td>
</tr>
</tbody>
</table>

**GLs and GL masks**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZLAM</td>
<td>Request a software GL to be generated</td>
</tr>
<tr>
<td>ZLMSK</td>
<td>Update the GL mask</td>
</tr>
</tbody>
</table>

**Magtape handling**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZHEAD</td>
<td>Write a special header in the next MT record</td>
</tr>
<tr>
<td>ZMKT</td>
<td>Write a special (user) MT record</td>
</tr>
<tr>
<td>ZMRED</td>
<td>Read a MT record</td>
</tr>
<tr>
<td>ZMBSR</td>
<td>Backspace 1 record</td>
</tr>
<tr>
<td>ZMFSR</td>
<td>Forward space 1 record</td>
</tr>
<tr>
<td>ZMBSF</td>
<td>Backspace 1 file</td>
</tr>
<tr>
<td>ZMF_SF</td>
<td>Forward space 1 file</td>
</tr>
<tr>
<td>ZMREW</td>
<td>Rewind MT, leave on-line</td>
</tr>
<tr>
<td>ZMRWD</td>
<td>Rewind MT, leave off-line</td>
</tr>
<tr>
<td>ZMSTA</td>
<td>Get status after last MT operation</td>
</tr>
</tbody>
</table>
APPENDIX B

RUN OPTIONS

Available options are the same whether set during the ZRUN dialogue or by calls to ZOPT. They remain unchanged between runs unless specifically modified by the user. All options consist of two characters and some have a decimal integer value in addition, which is input after the option.

The options available are:

**With or without MT in the system**

- **HELP** (or HE) Print a list of the options
- **RU** Start data taking
- **TO** Set the CAMAC operation time-out counters
  - Option value = loop count
  - A loop count of 1 = approx. 40 milliseconds
- **HP** Set to high priority analysis, i.e. 100% user monitoring sample
- **LP** Set to low priority analysis, i.e. rate-dependent monitoring sample
- **PP** Set to percentage priority
  - Option value = minimum % sample for monitoring
- **NR** Reset the run number
  - Option value = new run number
- **LM** Set (open) or reset (close) a hit in the CAMAC GL mask
  - Option value = bit number, 1 to 16 or 24
  - +ve = set ; -ve = reset
  - 100 = all set; 0 = all reset
- **DF** Restore default option settings of all run parameters
- **FE** Set event counter for automatic run end
  - Option value = number of events
- **FR** Set record counter for automatic run end
  - Option value = number of records
- **OP** Set logical unit number for system output messages
  - Option value = logical unit number
  - 0 = no messages
- **RS** Restart system
- **BY** Exit from BCS system and return to RTE disc system; or HALT if no disc
  - EX = BY
  - /E = BY

**With MT in the system**

- **PF** Position tape NF files from present position
  - Option value = number of files to be moved, NF
  - +ve = forward space; -ve = backward space
- **VL** Read and print-out the VOL1 label on MT
- **PL** Write a pre-label on MT
UM
  * RU with user MT records only
  1*EOF mark left after previous run (file) on the tape

UX
  * RU with user MT records only
  No EOF left after previous run (file) on the tape
  Only for unlabelled tape; defaults to UM for labelled tape

RX
  * RU with no EOF left after previous run (file)
  Only for unlabelled tape; defaults to RU for labelled tape
  Write end-of-data marks on MT at present position
  2*EOF marks for unlabelled tape
  End-file marks and "empty" end-file labels for labelled tape

RP
  Replay data from MT
  Option value = run number for replay
  = 0, replay starts from present MT position

XM
  Exclude MT writing

IM
  Include MT writing

FF
  Space MT forward until after 2*EOF marks

RW
  Rewind MT, leave on-line

PE
  Set parity error counter for automatic run end
  Option value = maximum number of records having write parity errors allowed
  before run is ended