MPS/CO Computer 71-11
8 December, 1971

PROPOSAL FOR THE INJECTION CONSOLE SOFTWARE EMPLOYING THE IMLAC PDS1 DISPLAY

B. Carpenter

Introduction

The injection console[1] is expected to be in provisional operation by the middle of 1972. It is therefore necessary that the software behind the console should be prepared by April 1972. This document is a proposal for the general structure of the software (named ISAAC - Interpretive System for Automated Accelerator Control). Details of individual measurements and controls will be decided as each part is written.

The decision to adopt the "maxi-console" approach[2] means that the software will operate mainly by use of the PDS1 display and keyboard. A fundamental constraint is that the PDS1 and the 4 K partition in the IBM 1800 are shared with ejection and general PS programs; careful co-operation, at least with ejection, is therefore desirable. The proposed system will occupy the partition and the PDS1 100% of the time when it is in use, although it will be computing for only about 25% of the time because of multi-programming.

It was originally hoped to provide a complete on-line high-level language, in the same way as at RHEL[3]. The existing injection interpreter program, derived from that proposed in ref. 4, was written as a pilot project for such a language. However, because of the shortage of manpower and because of problems with using an interpreter in a

multi-programmed computer, it has been decided not to provide a full language in the short term. Instead it is proposed to expand the existing injection interpreter to give the operator on-line subroutine and program calling facilities, grouped controls, and the possibility of using shaft encoders. The subroutines or programs called will be pre-defined, and only their sequence and parameters will be changed on-line. A data-bank (in core or on disc) and a few named variables (A, B, C, X, Y, Z) will be used for communication between the subroutines and programs.

Simple or very frequent measurements, displays or controls will be provided by core-resident subroutines called from the keyboard by certain parts of a syntax. More complex or less frequent operations will be performed by LOCAL (load on call) routines or by separate linked coreloads, both called by another part of the syntax. This distinction is necessary both because of the disc access time and because of the fact that only one LOCAL area is allowed by MPX.

The system must provide "SAVE AND BACK" and related facilities to agree with the general computer control philosophy. Disc file storage is necessary for this and desirable for STAR addresses and other variable information about the hardware. Unlike the case of midi-consoles, grouped controls play an important part in operation, so "SAVE AND BACK" etc., are only meaningful if carried out on groups of parameters *. For this reason the midi-console disc file format[5] is not suitable, and a different one must be adopted.

While considering grouped controls, we may define three terms:

sub-set of parameters - all the parameters involved in a particular grouped control.

---

* If it were possible to analyse the values of parameters to give values of the grouped control (e.g. Fourier analysis to give harmonics), then values of grouped controls themselves could be used for "SAVE AND BACK". However, this seems impracticable in a 4 K partition.

set of parameters - a set of parameters which are considered together with "BACK AND SAVE" etc.

super-set of parameters - all the parameters handled in parallel by the maxi-console at one time.

Note that the set is greater than the sub-set only if grouped controls overlap. This should be avoided if possible. The super-set is limited by the size of the partition. If it does not cover all injection parameters, a facility must be provided to change from one super-set to another by linking to another coreload. The number of parameters in a super-set is very difficult to predict without actually writing the program. The case where a set is larger than the super-set must not occur, and here the computer may impose limitations on the operation.

It might be desirable to force the operator to take the decision SAVE AND BACK/SET BUFFER/AUTO before changing from one set to another, and it will be essential before linking to another coreload. The number of parameters in a super-set is very difficult to predict without actually writing the program. The case where a set is larger than the super-set must not occur and here the computer may impose limitations on the operation.

It might be desirable to force the operator to take the decision SAVE AND BACK/SET BUFFER/AUTO before changing from one set to another, and it will be essential before linking to another coreload.

Syntax

The following is a definition of the proposed syntax available to the operator through the PDS1 keyboard.

Resident acquisition and display

{ TYPE EXPR (abbreviated T EXPR)
{ DISPLAY EXPR (abbreviated D EXPR).

These put on the display screen a value or graph (depending on EXPR). TYPE is executed once and forgotten. DISPLAY is executed repetitively every PS cycle.

EXPR can be

a) ± 5 digit decimal number
b) a recognized acquisition variable (e.g. IP, K1)
c) a recognized display name (e.g. ORBIT)
d) a recognized software variable (A, B, C, X, Y, Z)
e) ± decimal number multiplied by b) or d) (e.g. -23 * K1).

Resident control

{ SET CNAME OP EXPR
{ HOOK CNAME TO EXPR

These change the value of the variable CNAME according to the operator OP and the value of EXPR. SET is executed once and forgotten. HOOK is executed repetitively every PS cycle, and is intended for shaft encoders.

CNAME can be

d) a software variable
f) a recognized single or grouped control (e.g. HDIP89 , harmonic).

OP can be =
UP
DOWN
OFF
TO.

All operators can be abbreviated. TO is the same as UP. There is no EXPR when the operator is OFF. (If an "ON" control exists, it is implied by the other operators.)

For SET and HOOK, EXPR cannot be type c).

Non-resident, more complex instructions

```
CALL      ROUT
REPEAT    ROUT
C   ROUT   (EXPR1, EXPR2, ...)
R   ROUT   (EXPR1, EXPR2, ...).
```

These call, once or repetitively, a routine which may have up to 6 parameters. While in a REPEAT no other CALL or REPEAT is allowed as there is only one LOCAL area. CALL may also call a linked coreload with parameters (which will be transferred in a common). Routines and coreloads may modify A, B, C, X, Y, Z but other parameters are read-only.

EXPR as a parameter cannot normally be type c). However, if necessary, a name may be passed in character form to the routine or coreload for acceptance or rejection (e.g. CALL SANDB (HDIP) to carry out the SAVE AND BACK operation on the horizontal dipoles).

Delete repetitive instructions

U NAME

This undisplays, unhooks or unrepeats everything called NAME which has been requested by DISPLAY, HOOK or REPEAT.

It will be noticed that CALL is identical to a FORTRAN CALL. It is intended that a user will write and test a FORTRAN or FORTRAN-compatible subroutine, and that it can then be incorporated in the interpreter coreload as a LOCAL. After this it can be called on-line with parameters determined on the console, e.g.

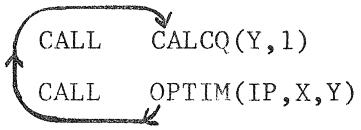
```
SET   X = 100
HOOK  X TO K1
REPEAT  OPTIM  (IP, X).
```

This would cause routine OPTIM to be called every PS cycle with the proton intensity IP and a value controlled by a shaft encoder as its

parameters. The operation would be stopped by:

```
U   OPTIM
U   X
```

Note that we cannot do a loop like the following:



If an application is too complex to be written as a LOCAL routine it can be written instead as a coreload, also called by CALL (but not by REPEAT). After completion, this coreload can link back into the interpreter which will continue normally (although of course some cycles will have been lost).

Thus CALL and REPEAT allow independently written routines and coreloads to be incorporated easily in the system. The only constraints are the need for compatible use of the PDS1 and respect for a common area. (In contrast, subroutines called by T, D, S, H statements must be written according to strict rules.) STAR control required by LOCALs will be done via the interpreter, not by a direct call. Coreloads using STARC must update the appropriate tables on disc.

Implementation

The computer facilities available are the following:

a) 500 words in the PDS1, plus the resident program, full time
b) partition of almost 4 K, priority 8, about 25% of the time
c) V-core, 6.5 K, priority 9, for the shortest possible time
d) N*320 words of disc file.

The software will be organized as an interpreter, i.e. a list-driven subroutine caller, because this is the most economical organization and is easy to modify. Since list-searching takes a significant time, the number of interpretive operations each PS cycle must be fairly small. The 500 words in the PDS1 will be used to check

the syntax of the commands typed in and to transmit them to the IBM 1800. With the syntax defined above, all but about 20 words of the PDS1's 4 K will be occupied.

The program in the IBM partition will perform the following functions every PS cycle:

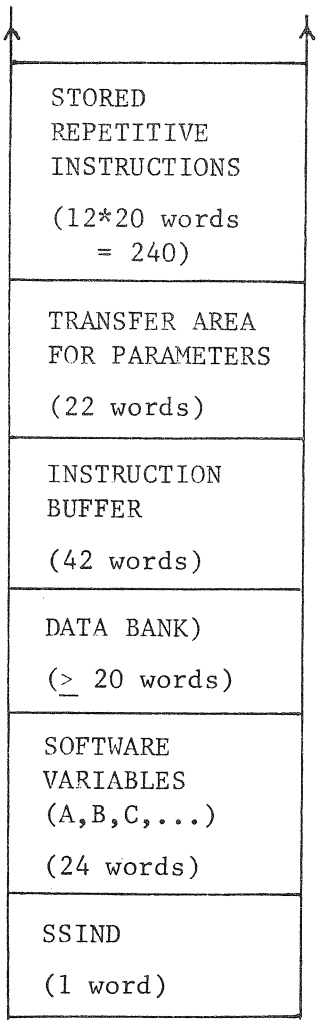
a) STAR acquisition for all parameters in the super-set
b) read in command from PDS1 } if necessary
c) interpret command } if necessary
d) interpret repetitive commands
e) STAR control required by c) and d)
f) transmissions to PDS1 required by c) and d).

Certain initialization operations, e.g. drawing axes, will be carried out by an interrupt coreload. "SAVE AND BACK", etc., will be initiated by a CALL and could also be done by an interrupt coreload if space is short (compare the present ICL for the back-leg windings). There will be a communication table whose address will be in a skeleton flag, and also a COMMON area in the partition. Figure 1 gives a provisional layout for the common.

To respect this common, a coreload linked by a CALL command must have been assembled with a *COMMON 350 card. It will exit simply by a CALL LINK back into the interpreter. If the coreload cannot permit such a long common, or if it destroys the display, it can use *COMMON 110, but must set SSIND to zero before the link. SSIND is the super-set indicator, telling the interpreter which super-set corresponds to the stored instructions. If SSIND does not agree with the super-set indicator of the interpreter in core, the stored instructions will be deleted as meaningless.

An ad hoc format will be chosen for the core data-bank, the disc data-bank and the disc storage of STAR information. The space required depends on operational needs but will be approximately:

Figure 1 COMMON



1 or 2 words per data-bank item

7 words per parameter
(2 STAR addresses, straight section number, 3 values, spare)

plus 20% miscellaneous.

Acknowledgement

This proposal has benefitted greatly from discussions with many people, notably H. van der Beken, M. Bouthéon, P. Lefèvre and G. Shering.

B. Carpenter

Distribution

CO Computer Section
IBM programmers
E. Asséo
O. Barbalat
C. Bovet
D. Dekkers
L. Henny
J.T. Hyman
P. Lefèvre
J.H.B. Madsen
G. Rosset
P.H. Standley
C. Steinbach

References

1. La console injection, M. Bouthéon, MPS/CO Note 71-52, 14 November, 1971

2. Minutes of meeting "Controls for PS injection systems", J.H.B. Madsen, 8 February, 1971

3. Visit to Rutherford Laboratory, B. Carpenter, P. Heymans, G. Shering, MPS-SI/Note CO/71-21, 22 November, 1971

4. Proposal for the control of the PS low field corrections using the PDS1 display and interpretive instructions, M. Bouthéon, B. Carpenter, G. Shering, MPS/CO Note 71-21, 6 April, 1971

5. a) Système midi-console, E. Asséo, L. Burnod, A. Daneels, E. Sigaud, CERN/MPS-SI/CO 71-3, 27 September, 1971

   b) A. Daneels, private communication