technical memorandum Daresbury Laboratory

DL/SCI/TM19A

A PROCESS CONTROL SOFTWARE PACKAGE FOR THE SRS

by

V. R. ATKINS, D. E. POOLE and W. R. RAWLINSON,
Daresbury Laboratory.

MARCH 1980

Science Research Council

Daresbury Laboratory

A PROCESS CONTROL SOFTWARE PACKAGE FOR THE S.R.S.

V. R. Atkins, D. E. Poole and W. R. Rawlinson

1. INTRODUCTION

The computer control system of the Daresbury Synchrotron Radiation Source employs a network of Interdata computers coupled as shown in Fig. 1. The site central computer, an IBM 370/165, is also included in the control system network, being used for programme preparation, bulk data storage and applications requiring larger processing power than the Interdata machines can handle.

The operating system software used in the control system computers was supplied with the machines. The model 7/32 runs under OS/32-MT and the model 7/16's run under OS/16-MT. Both operating systems have been considerably enhanced at Daresbury, in order to add features essential to the production of an integrated control system. A major part of this work was the production of the network intercommunication software to conform with the Daresbury network protocol, for both machine types. Another large development was the software to give high level access from application programmes for monitoring and control of the accelerator plant on a network-wide basis. This latter package which includes the design and implementation of the control system database, a special supervisor call and 'executive' type task handling all process input/output services for the 7/32, and process control 'device driver' software for the 7/16's forms the subject of this report.

2. SYSTEM ORGANISATION

A block diagram showing the organisation of the process control software is shown in Fig 2. The system database is defined by a source file containing a complete definition of every parameter to be controlled or monitored. This is used as input data to programme 'DATABASE' which builds four working files and constructs pointers for cross-referencing between them. These four files reside on a 7/32 disc pack, and it is necessary to produce a subset of the main database for each of the 7/16 minicomputers, which must be capable of operating with somewhat reduced facilities in stand-alone mode. All the 7/16 software is core-resident, and two means of establishing their database are provided. Programme 'MINIBASE' can produce either an assembler source version which, after assembly, can be linked into an OS/16-MT system as part of the normal system generation process, or a core image which can be loaded into a working 7/16 via the network communication system.

On completion of the database generation, the executive task 'PROCESIO' is loaded and started in the 7/32. This establishes calls to all 7/16's in the system via the network and assigns the database files. All application tasks in the 7/32 make their process control requests via the special supervisor call SVC8 to PROCESIO, which carries out database accesses and network operations on their behalf.

At the 7/16's, process control requests are received from the network by task 'REMPIO' which makes I/O requests via the normal OS/16-MT supervisor call SVC1 to three special device drivers, one handling all analogue input, the second all analogue output and the third all plant status control and monitoring. These drivers all access the plant via

CAMAC modules in a CAMAC serial highway system, obtaining the appropriate crate, station and subaddress codes from the 7/16 database subset.

The database subset may be accessed directly by application tasks within the 7/16's, and process control operations may be performed via SVC1 calls, although the repertoire of process control functions is more restricted than those available in the 7/32.

3. DATABASE

The structure of the control system database was determined by the method chosen for grouping and naming the machine parameters. Each parameter is defined by a three part name, the parts being termed the section name, group name and item number. The section name consists of two alphabetic characters, the first defining the plant area, and the second the system within that area. The present allocations for these are :-

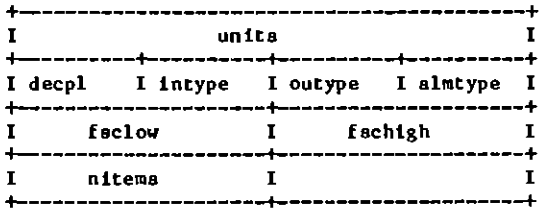
L - Linear Accelerator	V - Vacuum
F - Flight path to booster	G - Electron gun
B - Booster Synchrotron	R - R.F.
T - Transfer path to storage ring	C - Control System
S - Storage Ring	M - Magnets
P - Beam exit port	I - Injection
U - User beam lines	E - Extraction
	B - Beam

The group name consists of four characters, the first must be alphabetic, the remainder alphanumeric, and defines the name of a group of identical plant items, for example BV.IONP defines Booster Vacuum Ion Pumps. The item number, in the range 01 to 99 is appended to define uniquely a single parameter. There are eight ion pumps on the booster, identified as BV.IONP.01 to BV.IONP.08 , the order for allocating the item numbers being that in which the beam passes them during its transport through the machine complex.

The database structure is shown in Fig 3. A directory to the database is built up as the source data is processed by programme 'DATABASE' and saved in a contiguous file on disc. This contains an entry for each section name, a pointer to the first group entry for that section, and the number of groups within that section. It also contains a list of all the group names.

At system initialisation, the directory is read by programme 'PROCESIO' and retained in core to ensure fast look-up time. Given a parameter name, the section directory is searched until the section name is found, then the appropriate group file pointer and number of groups are used to search only the relevant portion of the group name table, and the index at which the group name is found is used as the random address to access the group data disc file which contains all the data common to that group plus a pointer and range for access to the data for all individual items contained in the item file.

The structures and definitions for the group and item file records are as follows :-



Group record - 16 bytes

where :

Units - Units of measurement, 4 ascii characters

Decpl - Number of significant decimal places

Intype - Used to define calibration algorithm for converting digital input to engineering units.

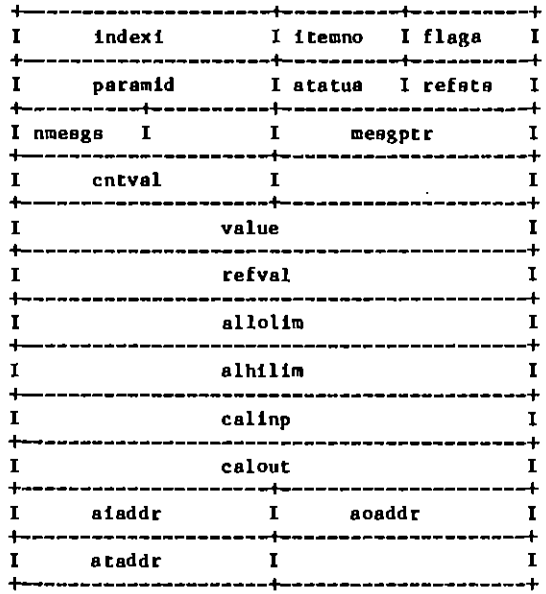
Outype - Used to define algorithm for converting engineering units to digital output.

Almtype - used to select type of alarm check to be applied to this group by data logging software (0 = no alarm check)

Fsclow - minimum allowable value for digital output

Fschigh - maximum allowable value for digital output

Nitems - number of plant items in this group.



Item record - 48 bytes

Indexi - random address of this record in item file, used when writing it back to disc.

Itemno - plant item number

Flags - low order hex digit describes attributes of parameter, high order hex digit used for access flags as follows:-

```
x... .... parameter is offline
.x.. .... control reserved for 7/16 system via
          portable console
.... .x.. parameter has status control
.... ..x. parameter has analogue control
.... ...x parameter has analogue monitoring
```

Paramid - parameter identifier, allocated at database generation. Hexadecimal quantity, numbers starting at 1000 allocated to parameters in Linac 7/16 system, numbers starting at 2000

allocated to parameters in Booster 7/16 system, numbers starting at 3000 allocated to parameters in Storage Ring 7/16 system, numbers starting at 4000 allocated to parameters in the beam lines system, numbers from 5000 upwards reserved for R & D system, and numbers with the sign bit set allocated to 'virtual' parameters which provide control via groups of other parameters.

Status - current status of parameter

Refsts - reference status, i.e. expected status for normal accelerator operation.

Nmesgs - the number of status interlock messages relevant to this parameter.

Mesgptr - random address of the first message in the message file. This is always the parameter title. Any further messages relate to interlock contacts connected via the status control and monitoring system.

Cntval - current control value applied to digital output.

Value - current value in engineering units.

Refval - Reference value, normally contains a 'standard setting' for that parameter.

Allolim - alarm low limit value.

Alhilim - alarm high limit value.

Calinp - calibration factor for analogue input.

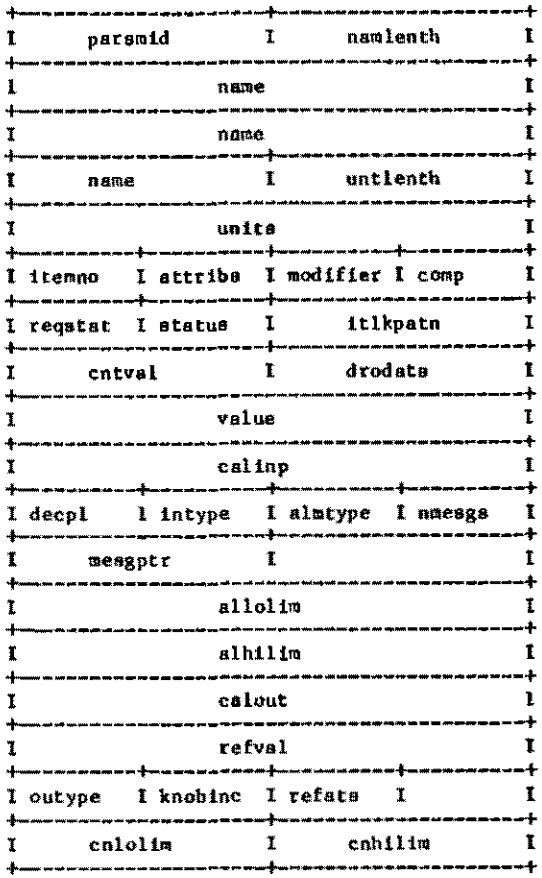
Calout - calibration factor for analogue output.

Aiaddr - encoded analogue input address.

Aoaddr - encoded analogue output address.

Staddr - encoded status address.

Application tasks use the process I/O procedures to access the database via task 'PROCESIO' and data from both group and item files is combined into a third record structure (mode PARAM in RTL2) which all the application level procedures use. In addition to database information, the structure contains space for volatile information about the plant items to be held. The structure is :



Param record - 72 bytes

Variables with names appearing in the group and item records above are copied from the database to the user parameter record when procedure dbopen is called. Definitions of new variables are:

Name - the parameter name in an RTL2 type array.

Units - units are now in an RTL2 type array.

Attribs - the contents of the flags byte in the database.

Modifier - used for enable/disable of control. Normally set up by task 'PROCESIO'.

Comp - the individual completion code set by any process I/O call. Procedure DBOPEN sets it to zero normally and hex 40 if that parameter has already been opened for control by another task in the 7/32. All other calls return codes set by the drivers in the 7/16a, see chapter 6.

Reqstat - requested status to be set on next call to SETSTS.

Status - plant status returned from last call to READSTS.

Itlkpatn - state of interlocks returned from last call to READITLK.

Cntval - control setting for analogue output to be set on next call to SETVAL.

Drodata - digital reading from last call to READVAL.

When DBCLOSE is called, only Cntval, Value and Status are copied back to the database files.

4. THE 7/32 PROCESS I/O SYSTEM

Process i/o to S.R.S plant is available to the INTERDATA 7/32 programmer by means of a supervisor call,SVC 8. The facility consists of

the SVC 8 handler and an executive task, PROCESIO. The process i/o task currently occupies 10.5 k bytes of core and handles database and network operations in response to user requests. An overhead of approximately 1k bytes is incurred for each task allowed to perform concurrent process i/o operations. The present implementation allows 3 tasks to be concurrently using PROCESIO.

The SVC 8 handlea user requests by making use of the OS32MT task handled trap facility. Requests to PROCESIO are added to its task queue.

A set of interface routines have been provided for the RTL/2 user. These routines construct the SVC 8 parameter block and issue SVC 8.

A user task in the 7/32 has access to any plant item via the supervisor call, SVC 8. He may issue SVC 8 to OPEN a table of parameters which he intends to read-only(i.e. monitor) or control by setting appropriate bits in the SVC 8 command byte. Operations can be performed on an i/o and wait or i/o and proceed basis. Having OPENed a table of plant items for process i/o the user may request that the table be UNPACKed into network buffers in readiness for the execution of various process i/o operations. The EXECUTE/PACK facility allows the user to specify a number of process i/o operations such as read value, set value, read status etc. by setting appropriate bits in the SVC 8 function code byte.When the network transactions for the EXECUTE/PACK function have completed, the user buffer is PACKed with the returned values in the network buffers. The UNPACK-EXECUTE/PACK mechanism operates on the whole of the user buffer. If the user intends to perform a control operation on only one of his tabled parameters he may use the ENABLE/DISABLE function of SVC 8 to selectively

disable control on those parameters for which the operation is not to be performed. This facility works at the user level only, the actual control or monitor status for all parameters having been established at OPEN time.Control operations on parameters which have been disabled at the user level result in the return to the user of current values held in the 7/16 database. This provides a means of setting up the 7/32 database from the 7/16. When the user no longer requires control or monitoring access to one or more of the parameters in his buffer, he may CLOSE those parameters by issuing an SVC 8 with the appropriate bit set in the SVC 8 command byte. The CLOSE function copies the control value, current value, status and reference status from the user's buffer to the database item file if the user had control access to the parameter.

The following describes the individual functions available in SVC 8 in more detail:

4.1 Functions Supported

Open - Gives the user access to one or more database parameters. A request to open can be made for monitoring only or for control and monitoring access. The open function completes entries in the user provided buffer for each opened parameter and in the case of controlled access sets bits in a bit map to indicate control of parameter(s) by this task.

Close - parameters previously opened by the user may be freed for use by other tasks by a call to CLOSE. The close function indicates a parameter is closed to the user by zero-ing the Paramid for that entry in the user's buffer. In the case of a

parameter which has been assigned for controlled access the relevant bit(s) in the bit maps are reset to indicate to PROCESIO that the parameter is no longer under control.

Unpack - following an OPEN operation, the user may issue an UNPACK request if he intends to monitor or control the opened parameters. The unpack function searches the user's buffer for non-zero Paramids. For entries with non-zero Paramids an entry is made in the relevant network buffer(linac, booster or storage ring).

The UNPACK function is performed in two passes. In the first pass entries are searched for parameters for which controlled access has been granted. The second pass searches for parameters which have monitoring access only. The two-pass approach allows ordering of the network buffers with controlled parameters placed first. Any subsequent execute operation, which is allowed for controlled parameters only, can be carried out with no re-arrangement of the network buffers.

Execute and pack - the Execute and Pack function carries out a plant i/o operation on behalf of the user for either monitor or control operations. This function performs network operations to the relevant mini computer(s) and returns the results to the user's buffer.

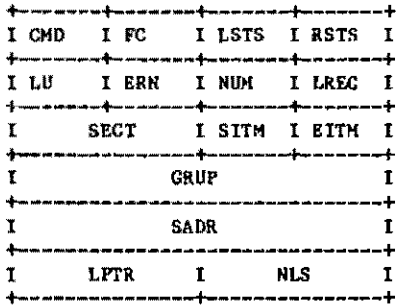
Closeall - This function is not accessible to the user. It is provided for use by the svc 7 executor when a task has gone to

'end-of-task'. Parameters which the user has failed to close are closed.

Enable/Disable - this facility is provided to give the user the ability to enable or disable control of parameter(s) AT HIS LEVEL. It does NOT alter control or monitoring status at the procesio task level.

4.2 SVC 8 Parameter Block

The svc 8 parameter block is 24 bytes long and has the following form:



where:

CMD - Command byte -bits have following meaning:

```
x... ....  open
.x.. ....  unpack
..x. ....  execute and pack
...x ....  enable/disable control
.... x...  close
.... .x..  'wait only'
.... ..x.  test i/o complete
.... ...x  not used
```

FC - Function Code-bits have following meaning:

```
x... ....  i/o and wait
.x.. ....  database group boundary crossing
```

```
                enable
..x. ....       database section boundary crossing
                enable
...x ....       control of parameter(s) required
.... xxxx       encoded part - used in execute and pack
                only:-
               0=nop
               1=read analogue binary
               2=read analogue calibrated
               3=read status
               4=read interlocks
               5=write analogue binary
               6=write analogue calibrated
               7=write status
               8=write reference value
               9=write reference ststus
               10=write increment
```

LSTS, RSTS - Local and remote status respectively. Lsts is set to indicate an error detected during processing of a requested process i/o operation. Rsts is set only by network read operations in the execute function. The status returned on a network read is logically 'OR'ed into rsts.

LU - Logical unit number of process i/o 'device'. In order to make use of the event co-ordination structure of OS/32-MT, process i/o operations are treated like i/o operations to a device.a DCB exists for the 'device' and a task wishing to perform a process i/o operation connects to the process i/o 'device' leaf.

ERN - Error number. This number is set by the process i/o task to indicate the last successful process i/o operation accomplished in a multi-function command. The value of this number is the number of the command byte bit specifying the function performed, i.e 1=open, 2=unpack etc.

NUM - The number of parameters in the user buffer.Also referred to as the 'number of lines' in the user buffer.

LREC - The record length of the user buffer. The value of Lrec is fixed presently at 72 bytes.

SECT - Section name-refers to the database section name for use in the OPEN function, e.g LV linac vacuum.

SITM, EITM - Start and end item numbers for parameters in the database.Used in OPEN function only.

GRUP - Database group name-used in OPEN function, e.g IONP

SADR - Address of user buffer.

LPTR - Line pointer-points to the parameter in the user's buffer at which the specified process i/o operation(s) is to begin.

NLS - Number of lines. Specifies the number of parameters in the user's buffer for which the requested process i/o operation(s) is to be performed.

4.3 SVC 8 Executor

The SVC 8 executor resembles an OS/32-MT device driver having initialisation,interrupt service and termination phases. The functions of the three phases are as follows:

4.3.1 Initialisation phase

This phase is entered in RS state, and the following checks are made:

.Presence of PROCESIO task

.Command byte non-zero

.Section name non-zero for OPEN request

.Start and end addresses of user's buffer relocatable and in the same logical segment

.Validity of specified logical unit number

.Validity of DCB address

For a valid request, the following actions are taken:-

.For 'wait only' and test i/o complete a check is made to see if the requesting task is connected to the process i/o 'device' leaf. If it is not, the condition code is set to zero and a return is made.

.For test i/o complete, the condition code is set to x'f' before returning to the user.For 'wait only', the i/o wait pending bit is set prior to exit.

.connection of the task to the process i/o 'device' leaf is attempted via EVQCON

.For i/o and wait the i/o wait pending bit is set in the TCB

.The entry point of the interrupt service phase is put into ISPTAB

.The DCB address is added to the timeout chain

.The DCB address is added to the process i/o task's queue

In addition an entry point SVC.CLA exists for the CLOSE ALL function which is called from the svc7 executor when the task has gone to 'end-of-task'.

4.3.2 Interrupt service phase

This phase is entered in IS state, and the following actions are taken:-

.The ISPTAB is set to ignore further interrupts

.A check is made for DCB time-out. If the DCB has timed out, an exit is made from the interrupt service phase since the termination phase will automatically have been scheduled

.Normal termination is indicated and the leaf address is added to the system queue

4.3.3 Termination phase

This phase is entered in ES state, and the following actions are made:-

.A check is made for DCB time-out. If the DCB has timed out then an error is indicated in the local status (sv8.lsts)

.A check is made for i/o and wait or i/o and proceed. For i/o and wait the i/o wait pending bit in the TCB is reset. For i/o and proceed the unrelocated parameter block address is added to the task's queue

.The dcb address is removed from the time-out chain

.The task is disconnected from the process i/o 'device' leaf.

4.4 PROCESIO Task

When PROCESIO is first started, it goes through an initialisation procedure which consists of the following:-

.Reading the database directory from disc into core

.Assigning the database files:

DATA:DATABASE.GRP group file

DATA:DATABASE.ITM item file

.Initialisation of PCB's, process control blocks (see later)

.Setting up calls to the plant mini computers. These calls remain permsnently set up

.Logging a message to the system console

.Setting up a 5 second timer interval

Following the initialisation procedure, PROCESIO goes into trap wait and becomes a completely trap driven task. At this point its task status word(TSW) specifies the following state:-

.Task queue service is enabled

.The i/o completion trap is enabled

.The timeout completion trap is enabled

.Svc 8 traps are enabled

On the occurence of a trap, the task status word specifies task queue service disabled with all the above traps enabled so that further traps occurring during trap servicing are queued. Up to 10 traps may be queued.

On completion of every trap service routine, a test is made to see if there is an entry on the task's queue. If there is an entry, the appropriate service routine is called. If no entry exists, the task enters trap wait.

The i/o completion trap is present to cater for the possibility of asynchronous network i/o operations being added at a later date.

4.5 Timeout Completion Trap

Every 5 seconds, a time interval expiry causes a trap which is serviced by the timeout completion trap service routine.

A check is made of flags within the task to see if any of the calls to the plant mini computers has gone down. An attempt is made to set up any calls which have gone down.

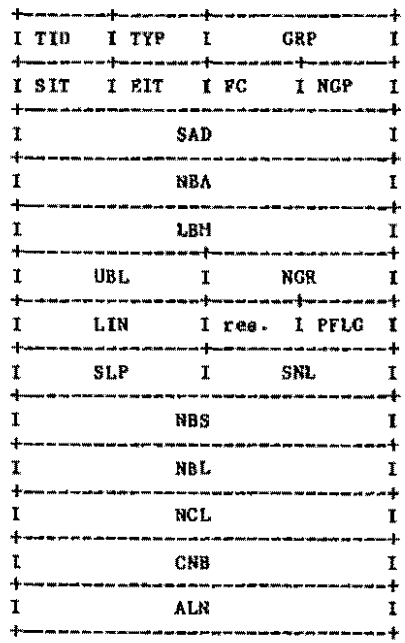
A new 5 second time interval is set up prior to exiting the service routine.

4.6 SVC8 Trap

The main body of the process i/o task is concerned with servicing the svc 8 trap.

The number of tasks which may concurrently be engaged in process i/o operations is determined by the number of process control blocks(PCB) which have been set up in PROCESIO. At present there are 3 PCB's,so 3 tasks can be involved in concurrent operations.

The PCB is 52 bytes long and has the following form:-



where:

TID - Task id. of task allocated to this pcb

TYP - Used in OPEN function. It specifies the type of OPEN request made.

GRP - Used in OPEN processing. It is a pointer to the group file pointers array held in the database directory.

SIT - Used in OPEN processing. It is the start item number requested by the user.

EIT - Used in OPEN processing. It is the end item number requested by the user.

FC - The function code specified by the user in his svc8 parameter block.

NGP - Used in OPEN processing. The pointer to the number of groups array in the database directory.

SAD - The relocated start address of the user's parameter buffer.

NBA - The network buffer area address.Each PCB is assigned three network buffers, each 256 bytes long, when PROCESIO is initialised. The network buffers are filled by the UNPACK function and transmitted to the 7/16 computers in the EXECUTE/PACK function. A network buffer is allocated to a specific 7/16 during the UNPACK function. The allocation is purely dependent on the plant items in the parameter buffer.

LBM - The local bit map. Each PCB is allocated a local bit map of 100 bytes, during PROCESIO initialisation. Each bit in this area represents a plant item in the database. In the present implementation 1200 parameters can be thus represented. A set bit

indicates that the task currently assigned to this PCB has controlled access to the represented parameter. A global bit map of the same size exists to indicate to PROCESIO which parameters are currently assigned for controlled access.

UBL - The user parameter buffer length, i.e. the number of parameters in the user's buffer as specified in the NUM field of the SVC 8 parameter block.

NGR - Used in OPEN processing. Specifies the number of the group record in the database.

LIN - The user specified line pointer from the LPTR field of the SVC 8 parameter block. This is saved in the PCB so that it can be restored in multiple function commands.

PFLG - Used in UNPACK function processing. It is the pass flag used to specify whether the monitor or control pass is being executed.

SLP - Used in UNPACK function processing to store the user's line pointer over the monitor pass.

SNL - Used in UNPACK function processing to store the user specified number of lines over the monitor pass.

NBS - Used in UNPACK and EXECUTE/PACK function processing. Consisting of four bytes, the most significant three bytes describe the status of each of the three network buffers. Note is made here of whether the buffer has been unpacked and executed and to which 7/16 computer the buffer has been allocated.

NBL - Used in UNPACK and EXECUTE/PACK function processing. Consisting of four bytes, the most significant three bytes are used to indicate the length of, i.e.the number of parameters unpacked into each of the three network buffers.

NCL - Used in UNPACK and EXECUTE/PACK function processing. Consisting of four bytes, the most significant three bytes are used to indicate the number of parameters, granted control access by OPEN, unpacked into each of the three network buffers.

CNB - Used in UNPACK and EXECUTE/PACK functions. Consisting of four bytes, the most significant three bytes are used to indicate the intended destination of each of the three network buffers.

ALN - Used in UNPACK and EXECUTE/PACK functions. Consisting of four bytes, the most significant three bytes are used to indicate the actual length of data in each of the three network buffers for transmission over the network.

4.7 Global Camac

A facility has been implemented by which a user task in the Interdata 7/32 can access the CAMAC systems of all the S.R.S. computers. It does not really belong to the process control package, but it is briefly described here for completeness, since it shares the network with the process i/o system.

The method adopted for user tasks to access CAMAC is via another Daresbury written supervisor call SVC0. The CAMAC branch numbers (hex) used within the 7/32 are as follows:

Branch 00 - 7/32 System crate

Branch 01 - 7/32 Control room Parallel branch
Branch 10 - Linac 7/16 System crate
Branch 11 - Linac 7/16 Serial branch
Branch 20 - Booster 7/16 System crate
Branch 21 - Booster 7/16 Serial branch
Branch 30 - Storage Ring 7/16 System crate
Branch 31 - Storage Ring 7/16 Serial branch
Branch 40 - Beam Lines 8/16 System crate
Branch 41 - Beam Lines 8/16 Serial branch

The SVCO executor carries out directly any camac cycles for which the high order hex digit of the branch code is zero, and routes all other requests via a trap to task PROCESIO. The high order hex digit is now used to select the appropriate 7/16 and the request is sent via the same network call as process i/o requests, and the CAMAC cycle is finally performed by 7/16 task REMPIO (see chapter 5) making an SVCO call.

5. THE 7/16 PROCESS I/O SYSTEM

5.1 Database Subset

The 7/16 computers have no disc storage, and need a database of their own in order to deal efficiently with requests from the 7/32, and to be able to operate in stand alone mode with the SRS portable control consoles. This database is core-resident and must be as compact as possible in order to minimise storage requirements. Only part of the data held in the 7/32 is repeated in the 7/16 database. The alarm limit data is omitted, and the output calibration data is omitted, so that parameters cannot be set to a value given in engineering units, but can be incremented by modifying the control value. The portable consoles are therefore restricted to this mode of operation.

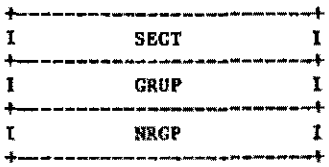
An additional feature is the provision of reverse pointers within the item and group records to allow backward referencing to the group and section data from the item data. Thus any call from the 7/32 need only identify parameters by their Paramid, which is effectively a displacement into the item data table, and the 7/16 can access the relevant group and item data using these reverse pointers.

The group name codes and the units of measurement are not held as Ascii characters strings, but as single bytes containing the displacements into tables of Ascii strings, so that duplication of the data in Ascii is avoided.

Three special process I/O drivers were produced which work directly into the database. This reduces the amount of data movement necessary and gives automatic maintenance of the variable, data within the database. A standard table format for all process I/O operations was adopted. This same table format is used as the data packet format on the network and thus avoids the need for table translations.

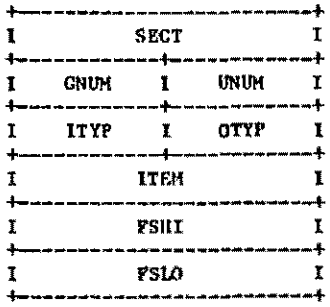
Process I/O addresses specified in the database contain a field which identifies the control station to which they belong and thus the CAMAC addresses involved may be obtained from the look-up tables. At first the simple approach of operating on each process I/O address in turn was adopted with no attempt to overlap operations in different control stations. If speed subsequently becomes an embarrassment then overlap can be accomplished by maintaing control station pointers for the I/O tables. These being driven down the I/O table in parallel to control operations in their respective control stations. The I/O terminating when all pointers reach the bottom of the table.

5.1.1. Structures for 7/16 database



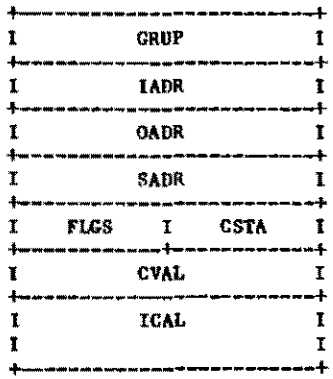
Section Record (6 bytes)

Where SECT = Two ASCII character section name
GRUP = Group file record pointer
NGRP = Number of groups in this section



Group Record (14 bytes)

Where SECT = Section record reverse pointer
GNUM = Group number (Button Number)
UNUM = Number of unit in unit array
ITYP = Input type number
OTYP = Output type number
STYP = Status type number
NITM = Number of items in this group
ITEM = Item file record pointer
FSHI = High full scale limit
FSLO = Low full scale limit



Item Record (16 bytes)

Where GRUP = Group record reverse pointer
IADR = Analogue input address
OADR = Analogue output address
SADR = Status address
FLGS = Flags
CSTA = Current status (Esp. stepping motors)
CVAL = Current control value
ICAL = Input calibration factor

5.2 Process I/O Addresses

Each parameter item has in general three I/O addresses associated with it, viz analogue monitor address, analogue control address and status address. These addresses occupy 16 bits each in the database and are of the following format.

DDDS SSMM MMMM MMMM

where

D = Device Code
S = Control Station Number
M = Multiplexer Address (format dependent on device code.)

Input, output and status addresses are distinguished by context.

Device codes which have been allocated so far are:

Status

Device code 0 - SRS Status Control System - Normal module
Device code 1 - SRS Status Control System - Ion pump module
Device code 2 - SRS Status Control System - Valve module
Device code 3 - SRS Status Control System - Extra interlocks module
Device code 7 - Stepping Motors

Analogue Input

Device code 0 - CAMAC input using F(0)
Device code 1 - CAMAC input using f(1)
Device code 2 - Solartron Analogue Scanner
Device code 7 - Read database control value (no I/O)

Analogue Output

Device code 0 - CAMAC output using F(16)
Device code 1 - CAMAC output using f(17)
Device code 2 - Stepping Motor
Device code 7 - Set database control value only (no I/O)

There is one and optionally two CAMAC crates in each control station. The mandatory CAMAC crate has the same crate number as the number of the control station which houses it and is called primary CAMAC. The other CAMAC crate is called secondary CAMAC, and has a crate number obtained by adding 8 to the control station number. Thus control station 1 has crates numbered 1 and 9, control station 2 crates numbered 2 and 10 etc. All potential LAM sources are in primary CAMAC. CAMAC device

drivers will, whenever possible be housed in the same CAMAC station in all Control stations, and the station numbers then become system constants.

Multiplexer address formats which have already been defined are listed below.

Status

Device code 0,1,2,3 - SRS Status Control system

system

XCCCCNNNNN

X = UNUSED
C = Status crate number
N = Status station number

Device Code 7 - Stepping Motor

Mutiplexor address unused

Analogue input

Device code 0,1 - CAMAC using F(0),f(1)

CNNNNNAAAA

C = 0 - Primary CAMAC
C = 1 - Secondary CAMAC
N = CAMAC station

A = CAMAC subaddress

Device code 2 - Solartron Analogue Scanner

CCNNNNAAAA

C = Scanner crate number
N = Scanner station number
A = Scanner point number

Analogue output

Device code 0,1 - CAMAC using F(16),f(17)

CNNNNNAAAA

C = 0 Primary CAMAC
= 1 Secondary CAMAC
N = CAMAC Station
A = CAMAC subaddress

Device Code 2 - Stepper Motor

00000nnnnn

N = CAMAC station

5.3 REMPIO task

Remote process i/o is handled by this task in the 7/16. The task waits to be called from the 7/32 via the network, and, having established its call, can receive a request from the 7/32 to read or set a list of up to 31 parameters, whose Paramids are contained in the data packet received from the 7/32. The header contains the logical unit number for REMPIO to use in a normal SVC1 call to one of the three process i/o drivers detailed below. The driver updates the table with any data required, and with the completion code for each parameter, and the task fills in the overall completion code and returns the table to the 7/32. The logical unit assignments used by the task are:

Logical unit 3 - Device C0 - Analogue Input
Logical unit 6 - Device 71 - Analogue Output
Logical unit 5 - Device 68 - Status

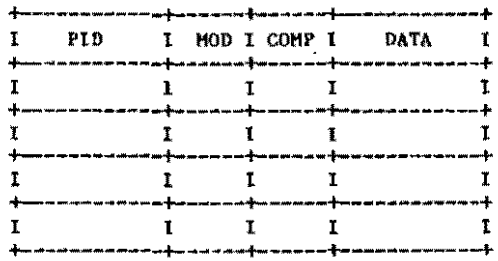
REMPIO tests for a process i/o packet by checking that all bits in the status and device number fields are set. If not, the packet is assumed to be a global camac request (see 4.7) and is used as the parameter block for an SVC0 call.

6. THE 7/16 PROCESS I/O DRIVERS

6.1 Table Formats

Various table formats are used - different for internal and external process I/O calls. All these tables are of horizontal format, where parameter ID's and data areas are interleaved. The various table formats can be presented in a uniform way to the driver by supplying the start address of the parameter I/O table, the start address of the data table, the data table increment and the number of entries.

The format of process I/O data tables which are transmitted along the network is



PID = Parameter I/D
MOD = Function modifier
COMP = Completion code (individual)
DATA = Data area - varies with function and device

A call for this table format would therefore look like

```
PTABLE   DB   function, logical unit
         DB   status, device number
         DC   TABADR
         DC   TABADR + 4
         DB   8, number of entries
         DC   ECB
```

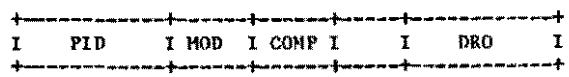
an overall completion code is provided which is the logical OR of the individual completion codes.

Two bits are reserved in the MOD byte which allow the user to inhibit control operations on one or more of the I/O table entries to a control driver (one bit for analogue, one bit for status). This proves useful if a table of parameters is being monitored by a programme but only one is being controlled. Correct manipulation of the control inhibit bits allows this to be done with no modifications to the parameter table between reading and setting. The control drivers check these control inhibit bits as the final check before carrying out the requested control operation, so that all other checks are made and errors reported even if control inhibit is set.

6.2 Analogue Input Driver

Functions: READ

Table format:



Addressing

Within the Linac and Booster systems is seems likely that only one analogue input system will be used, whereas in the Storage Ring system two or more may be present. Therefore, the drivers will be different within these systems.

Solartron Relay multiplexer - Linac and Booster systems

Address to multiplexer

This has capacity of 1000 points addresses with three BCD digits 12 bits. Three BCD digits correspond to crate, station subaddress. We limit our address space to 10 bits giving capacity for 4 crates (i.e. 400 points) per control station.

000000CCNNNNAAAA

Address from database

001SSSCCNNNNAAAA

S - Control station

C - Crate number

N - Station

A - Subaddress

6.3 Analogue Output Driver

Fuoctions - SET
INCREMENT

PID	MOD	COMP	VAL

PID	MOD	COMP	INC

Three types of analogue output modules are in use, viz. DAC modules (DAC 1081, DAC 1016V, ec418) and stepping motor drivers (1161, 1162). All DAC modules fit the definition of device code 0, that is CAMAC addressed by f(16). Addresses are therefore of the CAMAC format.

0009 SSCN NNNN AAAA

where:-

S = Control Station Number

C = 0 - Primary Camac

1 - Secondary Camac

N = CAMAC Station

A = CAMAC Subaddress

6.3.1 Stepping motors

Stepping Motor driver addresses take the form of CAMAC station numbers but as the modules are LAM sources they must be in the primary crate. In order to avoid delays within the control system, the driving of stepping motors is carried out asynchronously to the controlling programme. The controlling task, by its calls to the analogue output driver, simply manipulates the required destination value within the item record for the stepping motor. Control of the stepping motor is then carried out by interrupt service routines which run asynchronously to any task within the system. On a read from a stepping motor the position information is derived from the control value and therefore keeps pace with the control programme and not with the motor. The async. routines manipulate the status of the stepping motor so that an operator may know when the motor has reached the required set point. Thus when the motor has been started the read status will indicate MOVING until the motor stops when BLANK status will be read. As the position reading is provided by this 'dead reckoning' process, at system startup the datum point has to be established. This is accomplished by

driving the motor until the home limit switch is hit. During this process the read status will indicate RESETTING. When the home limit switch has been hit the motor is then driven out to the current set point as indicated by the control value in the database. The control value may be changed during the resetting process by the set or increment function. The RESET function of the status driver, if directed at a stepping motor, also causes this seek home-reset position cycle to be executed. This is useful if it is suspected that the 'dead reckoning' position may be incorrect. If a stepping motor is already resetting when a RESET command is received the command is ignored.

Additional data required for stepping motor control is held in motor control blocks. These reside in the OS-16/MT configuration module and the number of motors for which space is reserved is a system generation option.

PID	DN	FLGS	DEST	BCN

Motor Control Block (8 bytes)

where:

Pid - Parameter identifier as already defined

Dn - Device number for vectoring interrupts from CAMAC. Set up when system is initialised.

Flgs - Flags byte allocated as follows:

```
x... .... Motor is resetting
.x.. .... Motor is moving
..x. .... External fault
```

Dest - The total step count from home at the end of the current movement.

Bcn - The CAMAC branch, crate, and station of the stepping motor control module. Set up when system is initialised.

For both the SET and INCREMENT functions of the analogue output driver the updated control value is returned to the user in the value field of the I/O table. This provides for bumpless transfer of control between the 7/16 and 7/32, and also for automatic maintenance of the control value as contained on the 7/32 disc based database. It is also possible by this mechanism to move the current control value from the 7/32 database to the 7/16 database and vice versa. Thus enabling control settings to be preserved through a database re-generation (providing no items have been added or deleted).

6.4 Status Driver

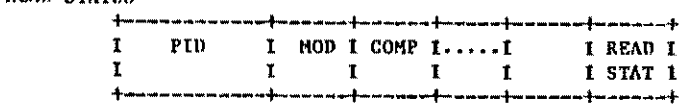
Functions: SET/READ STATUS
READ STATUS
READ INTERLOCKS

Table formats

SET/READ STATUS

PID	MOD	COMP	REQ STAT	READ STAT

READ STATUS



READ INTERLOCKS

Status address and command word to module.

ICCCCNNNNNXSTRNF

I - SET = READ INTERLOCK
RESET = DO COMMANDS

C = CRATE
N = STATION
X = UNUSED
S = START
T = TEST CONTINUITY
R = RESET
N = SWITCH ON
F = SWITCH OFF

Status address from database

DDDSSSOCCCCNNNNN

D = DEVICE CODE
S = CONTROL STATION (=CAMAC CRATE ADDRESS)
C = STATUS CRATE ADDRESS
N = STATION

Status codes returned to caller are of the following format:

XDDD AAAA

Where X = Unused
D = Device Code
A = Staus Code

Status Codes are:-

0 - No status
1 - Offline
2 - Reserved
3 - Invalid
4 - Unplugged

5 - Off,fault
6 - Off
7 - Off,ready
8 - On
9 - On,timing
A - Not used
B - Not used
C - Not used
D - Not used
E - Not used
F - Not used

Status requests from caller

		Command Hex
0	- Set Offline	
1	- Set Online	
2	- Reserve	
3	- Reset	0C
4	- Switch On	0A
5	- Switch Off	09
6	- Start	18
7	- Reset and On	0E

Status from module

6 bits XSCFRO

	Set	Reset
X	Unused	
S	- Start	
C	- Connected	Disconnected
F	- Fault	No Fault
R	- Ready	Not Ready
O	- On	Off

6.5 Network Considerations

Each task within the 7/16 system is given the attribute of local or remote. A remote task is one which carries out operations on behalf of some process external to the 7/16. In the present system only task REMPIO is given the attribute remote. The flags field of the item descriptor (ITM.FLGS) contains a bit indicator which is reset if that item is to be controlled by a remote process only and set if it is to be

controlled by a local process only. The two control drivers DACDVR and STADVR check if the calling task has the correct attribute to control the requested item. If it has not then the requested control operation is not carried out and the reserved bit is set in the completion code. The analogue read driver ADCDVR also makes this check and sets the reserved bit if control is not possible so that advanced warning is given to the calling process that control is not possible, although the requested read operation is completed.

Under normal circumstances the only control source internal to the 7/16 system is from a portable control console and external control sources are in the 7/32. This mechanism therefore prevents control conflicts between an operator using a portable console and a control programme in the 7/32 (this includes an operator at a main console).

7. FUTURE DEVELOPMENT - VIRTUAL PARAMETERS

An important enhancement to the system is planned, in order to provide a method of including in the control system parameters which are calculated from groups of other parameters, and which may only be controlled by simultaneously varying several parameters. We define these as virtual parameters, and a typical example is the energy of the booster synchrotron, and the field gradient at injection. These can be calculated from the a.c and d.c. magnet currents (BM.ACPI.01 & BM.DCPI.01), and the injection field level (BM.BINJ.01). The procedure for monitoring and controlling virtual parameters will be as follows.

On receiving an open request for a virtual parameter, the new PROCESIO task will access the database as normal and fill the appropriate line in the user's buffer, but, noting by its negative Paramid that it is a virtual, will also read in from a file a virtual parameter descriptor (VPD) containing code for calculating the value of the virtual parameter from its component parameters, code for calculating the settings of the component parameters for a required value of the virtual parameter, and parameter buffer space for the component parameters. On receiving a read request from the user task, PROCESIO will carry out a two stage operation, first reading the real parameters into its VPD buffer, and then using the algorithm in the VPD to calculate the value of the virtual parameter, which will be written into the user task's buffer. A reverse procedure will operate to set virtual parameters.

The virtual parameter enhancement is designed so that the user tasks do not need to know when virtual parameters are being processed, thus making a very general purpose facility.

Probably the most important set of virtual parameters will be those controlling the storage ring Multipole Magnets. By varying the excitation pattern of 12 individual windings, the Multipole is capable of producing any correction field from dipole to octupole. In order to make sensible correction field control available, the operator will be presented with a set of 'virtual' dipoles, quadrupoles, sextupoles etc which

will act upon the real winding currents via the virtual parameter software.

Thus it will be possible to introduce into the system control of any of the physics parameters of the accelerators whose algorithms in terms of real parameter values can be defined , and quantities like beam energy, q values, and chromaticities will be directly controllable under the hand of the operator.

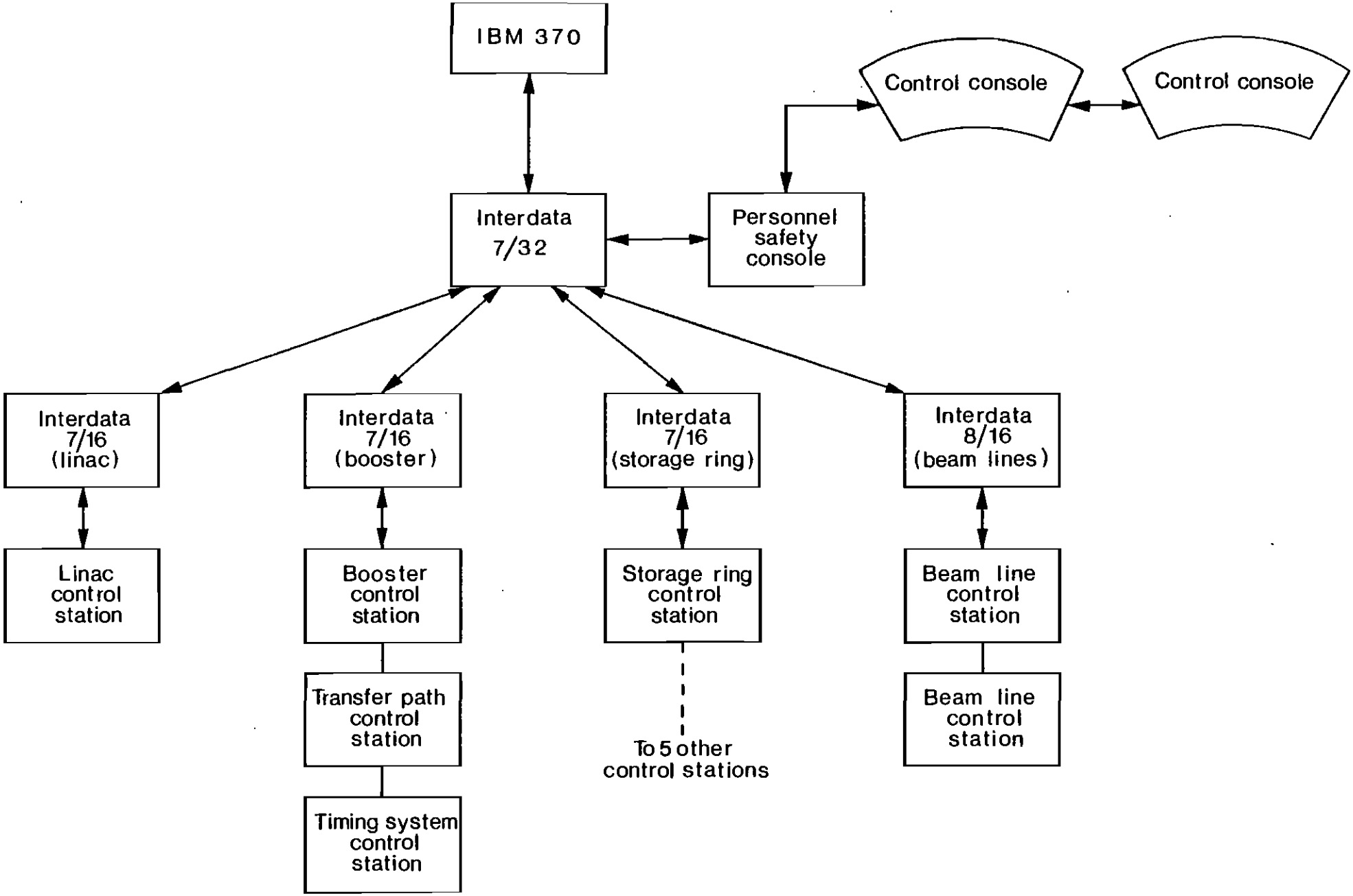

IBM 370
Control console
Control console
Interdata 7/32
Personnel safety console
Interdata 7/16 (linac)
Interdata 7/16 (booster)
Interdata 7/16 (storage ring)
Interdata 8/16 (beam lines)
Linac control station
Booster control station
Storage ring control station
Beam line control station
Transfer path control station
Beam line control station
To 5 other control stations
Timing system control station

Fig.1

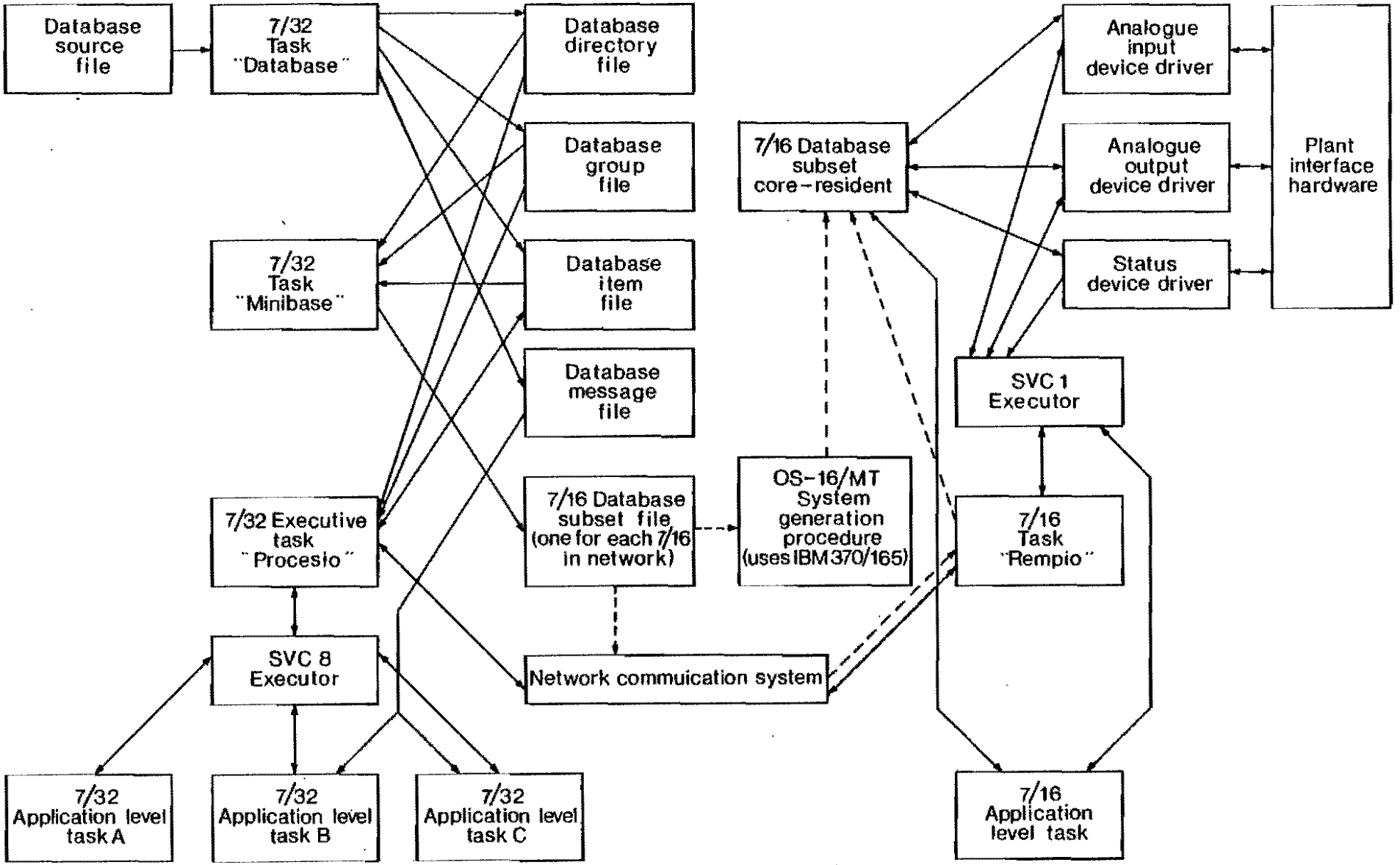

Database source file
7/32 Task "Database"
Database directory file
Database group file
Database item file
Database message file
7/16 Database subset file (one for each 7/16 in network)
7/16 Database subset core-resident
OS-16/MT System generation procedure (uses IBM 370/165)
Analogue input device driver
Analogue output device driver
Status device driver
Plant interface hardware
SVC 1 Executor
7/16 Task "Rempio"
7/16 Application level task
7/32 Task "Minibase"
7/32 Executive task "Procesio"
SVC 8 Executor
Network commuication system
7/32 Application level task A
7/32 Application level task B
7/32 Application level task C

Fig. 2

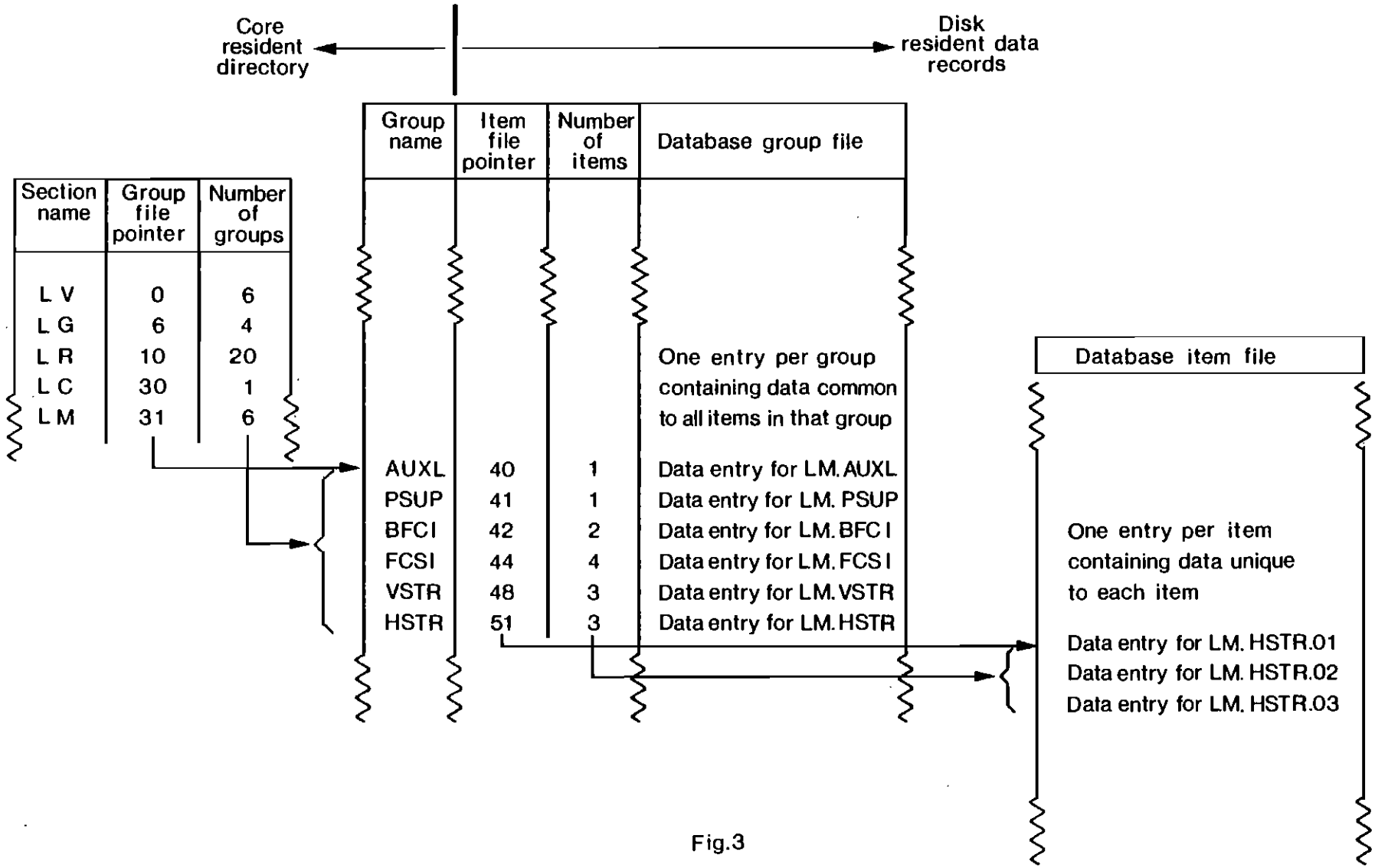

Core resident directory
Disk resident data records
Section name
Group file pointer
Number of groups
L V 0 6
L G 6 4
L R 10 20
L C 30 1
L M 31 6
Group name
Item file pointer
Number of items
Database group file
One entry per group containing data common to all items in that group
AUXL 40 1 Data entry for LM.AUXL
PSUP 41 1 Data entry for LM. PSUP
BFCI 42 2 Data entry for LM.BFC I
FCSI 44 4 Data entry for LM.FCS I
VSTR 48 3 Data entry for LM.VSTR
HSTR 51 3 Data entry for LM.HSTR
Database item file
One entry per item containing data unique to each item
Data entry for LM. HSTR.01
Data entry for LM. HSTR.02
Data entry for LM. HSTR.03

Fig.3