© 1983 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

IEEE Transactions on Nuclear Science, Vol. NS-30, No. 4, August 1983

THE CONTROL SYSTEM FOR THE DEDICATED SYNCHROTRON RADIATION STORAGE RING BESSY

G.v. Egan-Krieger, W.-D. Klotz and R. Maier
Berliner Elektronenspeicherring-Gesellschaft für Synchrotronstrahlung m.b.H. (BESSY)
1000 Berlin (West) 33, Lentzeallee 100

I. Introduction

BESSY[1] is a 800 MeV electron-storage ring dedicated to research with synchrotron radiation in the VUV and soft X-ray region. The storage ring is fed by a 800 MeV separated function synchrotron with a classical 20 MeV microtron as preinjector. The machines are remotely controlled by four minicomputers which are linked by a local network. The computers are connected to a commercially available electronic interface system[2]. The total amount of controlled I/O-channels is given in tab. 1. The general philosophy for the system design was determined by boundary conditions like:

i) reliable processor hardware
ii) reliable and powerful multiuser multiprogramming real-time operating systems
iii) well developed software tools
iv) a ready to use distributed network with high throughput
v) compatability of the electronic interface system
vi) locally present service work shops
vii) if possible, avoid assembly coding

II. The Hardware System

II.1 The electronic interface system

A 12 bit parallel data multiplexing system, called DEV 12, is used. The system is connected via full duplex interfaces to the computers. It uses 1 byte for data and addresses, 4 bits for control. Each I/O-card is addressed by a slot (0-15) and a crate (1-15) number. Up to 8 crates (DEV 12.1) can be connected to one distributor (DEV 12.4). The physical slot positions define the interrupt priority. All crates and distributors are optically isolated from each other. Cables of 25 symmetrically driven twisted pairs are used as interconnections. The system is completely program controlled by the computers.

II.2 Standard I/O-cards

The physical card dimensions are approximately of eurocard format (100x225 mm^2). Six different types of cards (tab. 2) were developed. 3/4 of all equipments are interfaced by a single card. The rest uses combinations of several cards. For all register outputs differential line drivers with clamping diodes are used. All register inputs are equipped with optically coupled isolators and schmitt-triggers. The pulse outputs are driven by optically isolated darlington transistors with a maximum steady drive current of 0.5 A. For reading analog signals with a resolution up to 12 bits, voltage to frequency conversion is used. Up to 17 of these signals can be connected to a single multichannel counter card. For signals with resolution up to 16 bits a FLUKE 8502A digital voltmeter with analog scanner and HP-IB bus control is in operation.

III. The computers and the network

The computers are four idential Hewlett-Packard 1000F 16 bit processors. They possess hardware floating point arithmetic and microcoded fast FORTRAN, scientific and vector instruction sets. Their maximum semiconductor memory capacity is 2 Mbyte. The computers are linked by a DS/1000 local network. This network is a communication package for HP-1000 computers consisting of network software-firmware, serial interfaces and cables. Its hardware transfer rate is 1 Mbyte/s. Its effective transfer rate for program to program communication is in the order of 15 Kbyte/s. The processor overhead, defined as percent of otherwise available CPU's time, is typically 40% for a single hardware link. A complete layout of the network's and control system's hardware is shown in fig. 1.

IV. The software system

The software system is organized in a hierarchy of different layers. With the exception of the producer's software the largest part of the programming (about 90%) was done in FORTRAN. Mainly for reasons of time optimization the rest was written in ASSEMBLER.

IV.1 The message transfer system

From the very start it was clear that a large number of programs, each of them performing a different task within the system, has to communicate with each other. Therefore, as one of the most basic building blocks of the system, a general purpose message handling processor was developed. This message handling processor, called IPCCM (interprocessor communication & control monitor), is written in FORTRAN, uses 12 Kbyte for code and 4 Kbyte of dynamic buffer memory. It runs with high priority in real-time foreground and is locked into semiconductor memory in order to prevent disc accesses for message transactions. Each node of the network contains one IPCCM program. All IPCCMs are permanently available to all programs, they are initialized on system start up. Each IPCCM performs the following task:

i) it queues messages from sending programs with a specified timeout,
ii) it queues requests for reception of a message. That means, it puts receiver programs in a special wait state, if no pending messages are available at the time the program performs the request. This request can also be given a timeout, so that after the specified interval the requesting program will continue running without getting a message,
iii) it controls flow of messages through the computer network based on a store-and-forward mechanism. Say a program A sends a message to program B on a different node. The message will be automatically passed through the network by the IPCCMs. Sending and receiving messages is completely transparent to the programmer. To achieve this, the IPCCMs are able, to communicate directly with each other via remote EXECUTIVE calls (fig. 2). The path of the message through the network, once the sender's and receiver's nodes are known, is found by a network description table, which is common to all IPCCMs.
iv) it gives actual status information of the message transfer system within each node. This is a very essential feature for software development and debugging in an environment where about 40 active programs on different computers communicate by message exchange.

IV.2 The peripheral interface modules

A common problem in controlsystem design is the large number of different equipments and subsystems which

0018-9499/83/0800-2273$01.00©1983 IEEE

have to be handled by the software system. All peripherals are grouped into classes of equal or very similar equipments. For instance: 4 groups of power supplies, each containing about 20 to 40 equipments, 2 groups of power supplies for ion getter pumps, 1 group of vacuum valves etc. Access (I/O and control) to equipments of one group is performed by peripheral interface programs (PIM). In the system there is no other path to equipments than via PIMs and therefore only these programs contain device dependent code. The PIMs communicate with programs, performing control algorithms, through the message transfer system by exchange of messages in a standardized format (fig. 3). For each equipment a PIM owns a device description table (DDT). All DDTs of a group of equipments form the DATA TABLE (DT). The DT can be visualized as a 2-dimensional sequential list. The first index points to the equipment, the second to entries in the DDT. For instance:

```
DT ( 1, 5)  = I/O-address of power supply # 1
DT (16,21) = status of power supply # 15
DT (20,16) = actual current of power supply # 20 etc.
```

After initialization of all PIMs, all DTs in the different nodes represent a distributed data base. Access to this data base is exclusively done by the PIMs (fig. 3). A user program can activate a PIM by sending a message to it, specifying the equipment # and an operations code (OP-code) in the message. The user specifies the OP-code by a 4 character string, which is internally converted into a definite 16 bit integer:

```
#CUR    =   read current
#STA    =   read status
#SET    =   set current
#ON     =   switch on
#OFF    =   switch off    etc. ...
```

After performing the requested task, the PIM sends an acknowledgement to the caller and goes into the receive state.

IV.3 The data base

Besides the actual data base of the running system represented by the DTs in the PIMs, there is need for a disc resident data base for the following reasons:

i) in order to initialize the system, the PIMs have to read their DTs from a back up file,
ii) in order to update permanent data in the data base, the DTs have to be resident on disc,
iii) there must exist a directory of equipments in the system, so that programs which want to perform a control operation on an equipment can get an identifier, specifying the equipment # and the PIM which is responsible.

Each equipment in the system is definitely defined by a 6 character string. A program which wants to perform an I/O operation has to do the following steps:

i) to get its own program identifier
ii) to ask the data-base-directory monitor (DBDM) to convert the equipment name into an identifier
iii) to send a message to the corresponding PIM
iv) to issue a receive request to get an acknowledgement from the PIM.

In the initializing phase the PIM asks the DBDM for the complete DT. The DBDM then reads the DTs from the data base file and puts all equipment names into its memory resident directory. After initialization the DBDM does not have to perform further disc accesses. All entries of the directory are accesses by a hash algorithm performed on the strings of the PIMs and the equipments names. Filling data into the data base file is done by utility programs like a data-base-editor, a data-base-generator etc.

IV.4 The system library

In order to free the programmer from taking care about internal details, libraries of system calls have been established. For example the FORTRAN statement

```
CALL CUR (6HDIPPR ,X)
```

reads the current of the dipole power supply named "DIPPR" and returns its value in X.

```
CALL OFF (6HDIPPR )
```

switches off the same power supply and

```
CALL SET (6HDIPPR , 900.)
```

sets the current of this power supply to 900 amperes All library calls are available in BASIC, FORTRAN and PASCAL.

IV.5 The producer software

The efficiency of planning, coding and debugging control system software is strongly dependent on the versatility and performance of the producer software. The HP computers, equipped with mass storage devices, are running RTE-IVB, a disc based real-time multiuser multiprogramming executive. A smaller version RTE-MIII, is running on the satellite processors, which are not equipped with a hard disc. The executive calls are identical for both operating systems. Therefore all programms can be developed and tested on the disc based systems using full advantage of their file management. The programs are finally crossloaded on the central nodes and then downloaded into the satellites. Essential features of the DS/1000 network software are

i) remote command processing
ii) program to program data exchange (PTOP)
iii) remote file access
iv) remote executive call
v) automatic store-and-forward

In order to stay compatible with future improvements of the producer's software, no attempt at all was made to change any part of this category of software to adapt it to specific needs.

V. The operator interface

Three different devices are the operator's interface to the control system.

V.1 Alphanumeric terminals

The operator can enter commands interactively from terminals distributed on the site. He simply has to push programmable keys on the keyboard to step down a tree-structured menu until he reaches a task like setting a power supply's current or reading an equipment status. He also has the opportunity to write simple procedure files in an unstructured command language, like switching on 20 power supplies one after the other. In order to perform control algorithms in a structured language, he has the possibility to schedule a real-time BASIC interpreter to which the library of system calls was added.

An example:

```
10 LET A$="DIPPR"
20 FOR I=600 TO 700 STEP 0.5
30 CALL SET (A$,I): FAIL GOTO <ERROR>
40 WAIT (1000)
50 NEXT I
```

Within the loop the power supply named DIPPR will be ramped from 600 A to 700 A in steps of 0.5 A, one step per second.

V.2 Colored TV-raster scan monitors with interactive cursors

Independently from the terminials the operator has the possibility to get status information and to control the machines by stepping through a menu by means of interactive tracker-ball units connected to two colored raster scan monitors. The refresh memory of the raster scan monitors has a resolution of 256x512 pixels with 8 bit depth per pixel.

V.3 Computer controlled knobs

The most important devices for adjustment of machine parameters are six computer controlled knobs. The knobs are assignable to any controllable variable of the machines, so that the operator can for instance change power supply's current in a quasi analogeous manner. The actual and the demanded values of the controlled variable are displayed on-line on a small TV-monitor. The knob consists of an incremental angle encoder with 500 counts per revolution. The TV-monitor is connected to a character generator which produces an alphanumeric display of 8 lines with 16 characters per line. When the incremental encoder is turned, the first pulse of the pulse train triggers a gate circuit which opens an up-down counter input for 100 ms. When the gate circuit closes again, an interrupt to the computer is generated. After reading out the counter the computer enables the gating circuit again. With one knob in action a total turn around time, starting from the interrupt, sending a new value to the equipment through the message transfer system and receiving the actual value on the display, of about 100 ms is achieved. The knobs have already successfully been used for multiparameter control tasks, like producing bumps on the closed orbit, where at least three power supplies have to be variied at the same time with definite proportions.

digital register outputs	173
dital register inputs	438
output pulsed (commands)	132

tab. 1: total number of I/O-channels in units of bytes

typ	function	name
DOR	16 bit output register with handshake or interrupt	digital output register
IIR	16 bit input register with handshake or interrupt	isolated input register
SPO	16 bit input register with handshake or interrupt 8 pulse outputs	status input/ pulse output
PSC	12 bit output register 8 bit input register 5 pulse outputs	power supply controller
MCC	17x16 bit counters with RAM-sequencer control and FIFO I/O-buffer	multichannel counter
HP-IB	16 bit open collector I/O	HP-IB interface

tab. 2: standard I/O-cards

References

1) D. Einfeld, W.-D. Klotz, G. Mülhaupt, T. Mueller, R. Richter IEEE NS-26 (1979)3801
2) FREY Analogtechnik, Marathonallee 33, D-1000 Berlin 19, FRG
3) Donald E. KNUTH, The art of computer programming, ADDISON-WESLEY, PUB.COMP., INC.(1973) 2nd Ed.

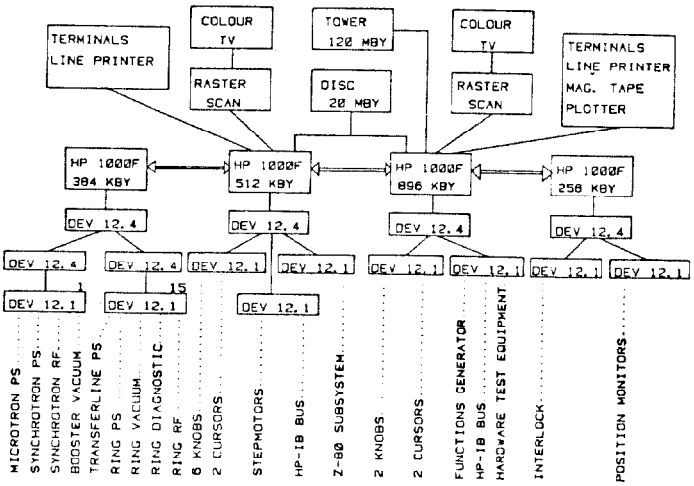

fig. 1: Control system network

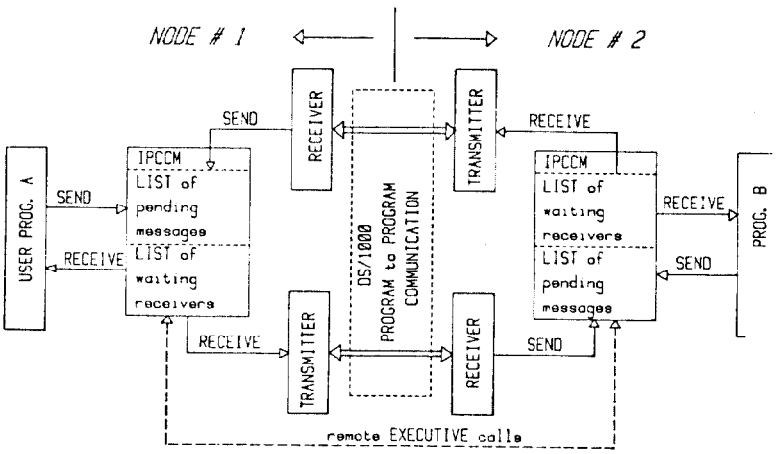

fig. 2: Data paths in the message transfer system

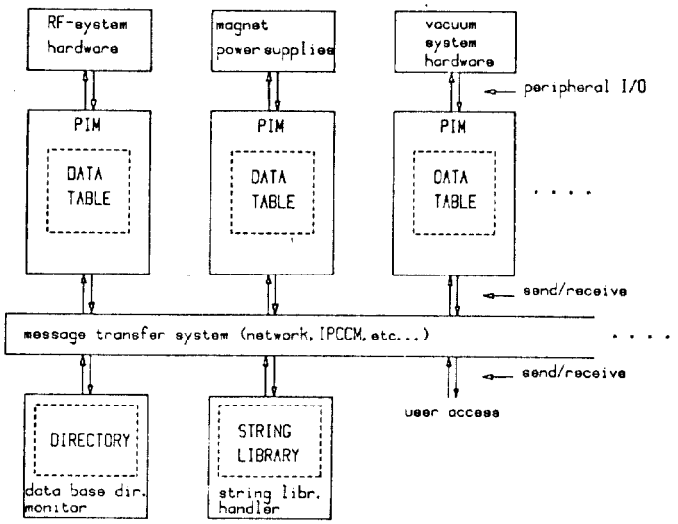

fig. 3: The distributed data base