THE COMMUNICATION PACKAGE FOR THE EQUIPMENT NETWORK OF THE LEP AND SPS ACCELERATORS

A. Bland

European Organization for Nuclear Research (CERN),
CH - 1211 Geneva 23, Switzerland

Abstract

The CERN Large Electron Positron (LEP) collider ring under construction and the existing Super Proton Synchrotron (SPS) accelerator use the same protocol for communication over the Equipment Network. Communication with equipment is performed via the MIL-1553-B bus from Bus Controllers in the VME Process Control Assemblies (PCA). All information is transmitted over the 1553 Bus and the VME Bus using datagrams and various services have been implemented on top of this basic communication mechanism. Programs in the PCA and in Equipment can send and receive Command/Response messages, using a Name Server to provide physical addressing. Equipment can broadcast on their local bus, perform programmable broadcasts globally, send Alarms and request the running of a program in the PCA. In addition an unstructured transparent mode of communication allows industrial equipment to communicate with each other without modification to their software. The overall goals have been to keep the software required in the equipment to a minimum (<10k bytes) as many are 8 bit systems, to provide reasonable speeds for large data transfers (10-20k bytes/s) and fast response times (up to 250 local broadcasts per second).

Introduction

The Equipment Network of the LEP and SPS Accelerators is used to control a very large amount of equipment in the LEP and SPS tunnels and service buildings. Access to the Equipment Network from the main LEP and SPS control room is via the Token-Ring Network [1]. In addition an accelerator synchronisation network permits real-time equipment triggering for control and data acquisition.

The basic aim of the communication package for the LEP and SPS has been to have a uniform application software interface that is small and fast, despite the differences in system hardware and software between various implementations.

Communication with the LEP and SPS Equipment Control Assemblies (ECAs) is from a Process Control Assembly (PCA). The MIL-1553-B bus is used as the communication medium. There may be up to 8 bus controllers (BCs) in a PCA and each Bus Controller may be connected to up to 30 Equipment Control Assemblies.

PCA Hardware Overview

In the SPS, systems typically consist of a Norsk Data NORD 100 minicomputer connected via a special NORD 100 - VME interface card to a VME chassis called the MINI-PCA (Mini Process Control Assembly). The MINI-PCA chassis contains a number of 68010 processors. One of these is the Equipment Directory Unit (EDU) which performs the Name Server function looking up the physical address of an equipment from the name used by application software to access equipment. The other 68010 processors are MIL-1553-B Bus Controllers (BCs).

Current systems for LEP consist of an IBM PC/AT running PC-DOS or UNIX connected via a 9600 baud serial line to a MINI-PCA chassis identical to that described above. The serial line will soon be replaced by a 1553 connection between the IBM PC/AT and one of the Bus Controllers in the MINI-PCA, which will improve performance.

Future LEP systems will be based on DLX Process Control Assemblies made by the French company CIMSA-SINTRA. These are 68010 VME bus multiprocessors systems running the ELECTRE real time executive. In addition to providing the Name Server function and having 1553 Bus Controllers as in the MINI-PCA these systems will be connected to the LEP Token Ring Network and be capable of running application programs in the PCA itself.

ECA Hardware Overview

Equipment Control Assemblies of many types are supported. The Remote Terminal Interface (RTI) cards to connect an ECA to the 1553 bus are available for various busses.

1) G64 bus RTI cards for Zilog Z80, Motorola 6809, 68000 and 68010 based Microprocessor systems.

2) VME bus RTI cards for 68000 and 68010 systems.

3) MICEN E bus RTI cards for Texas 99000 systems.

4) IBM PC bus 1553 adapter card incorporating a 68010 processor and implementing both Bus Controller and Remote Terminal Interface protocols.

PCA Software Overview

Access to ECAs from NORD 100s is from the interpreted language NODAL. NODAL is similar to BASIC in that it is interactive but it incorporates many extra features to allow multi-tasking and program execution on remote computers via a network. The SPS environment consists of many NORD 100s communicating via the TLTN network. From any NORD 100 on this network a NODAL program or NODAL line may be executed on any other NORD 100 also on the network.

IBM PC/ATs running PC-DOS provide access to ECAs from NODAL and the compiled language Modula-2. As these are essentially single-tasking systems they are run standalone, without access to other networks apart from the Equipment network. They are
used mainly for Equipment Access program development but some operational systems for the LEP are installed which use PC-DOS.

IBM PC/ATs running the Unix-like operating system XENIX are connected to the Token Ring network via TCP/IP protocols. Application programs to access equipment may be written in NODAL, C or Modula-2.

The Sinstra DLX Process Control Assemblies communicate over the Token Ring network using TCP/IP protocols. Programs running in the DLX can access equipment from NODAL, Modula-2, C and Fortran.

ECA Software Overview

Various software interfaces between Equipment Control Assemblies and the Equipment Network have been developed by the LEP/SFS controls group.

1) A simple 'polling' interface for Motorola 6809 microprocessor systems using the G64 bus. This interface is written in the language PASCAL. These systems often run the FLEX operating systems and the interface is compatible with FLEX although they may also be used standalone without an operating system.

2) A PASCAL interrupt driven interface for standalone G64 6809 systems or G64 6809 systems running the FLEX operating system.

3) A PASCAL multi-tasking interface using the ANX real-time executive for standalone G64 6809 systems or G64 6809 systems running the FLEX operating system.

4) A multi-tasking interface written in Modula-2 for 68000 and 68010 systems. This can be used from Modula-2, PASCAL and C.

In addition LEP and SPS Equipment designers have written interfaces for:

1) 6809 and 68000 systems running the OS9 operating system. This is implemented as an OS9 device driver so it can be used from C, BASIC and assembly language.

2) Zilog 280 systems running CP/M using the G64 bus. The interface is written in Turbo PASCAL.

3) NODAL running in Texas 99000 systems using the MICEW bus. The interface itself is written in assembly language.

Communication in the Equipment Network

Communication in the equipment network is performed by the transmission of packets (i.e. datagrams). Packets have the same format on the 1553 Bus, VME bus and IBM PC/AT serial line. Each packet has a 16 byte packet header followed by up to 240 bytes of data, giving a maximum packet size of 256 bytes.

The BYTECOUNT gives the total packet length in bytes including the header.

The VERSION is used to check if the sender and receiver of a packet are running the same version of the communication software and allows extra features to be added in later versions while maintaining compatibility with older versions.

The DESTINATION ADDRESS is used for routing a packet to its destination and the SOURCE ADDRESS allows the receiving system to reply to the sender by simply swapping the source and destination addresses. The high byte of an address specifies the VME module number of a processor in the Process Control Assembly. Generally VME module numbers are between 1 and 31. The low byte specifies the sub address with respect to the VME module. Equipment Control Assemblies connected via a 1553 RTI card have addresses between 0 and 31 while by convention 255 is used to address the VME module itself. In this way every element of a Process Control Assembly and every ECA can be uniquely addressed.

The PACKET TYPE is used to differentiate the various services provided. The most important packet types are:

OpenEquip - Asks EDU for ECA address (Command/Response)
ReturnOpen - EDU returns address of ECA with this
WriteData - Data for Command/Response service
CloseEquip - Informs EDU that Command-/Response finished
OpenTransparent - Asks EDU for ECA address (Transparent Service)

TransparentData - Data for Transparent Service
CloseTransparent - Informs EDU that Transparent mode finished
Alarm - For sending alarms
BroadcastData - Data for Broadcast Service
MulticastData - Request a multicast service
ServiceRequest - Service Request
ModuleInfo - Received at startup - supplies server addresses
Bounce - Low level test packet
Error - Returned to sender of a packet which failed

The SEQUENCE indicates if this is the FIRST PACKET, one of the MIDDLE PACKETS or the LAST PACKET of a multi-packet message. SINGLE PACKET messages are used for the simpler services. In this way the amount of data in a message can vary between 0 and an infinite number of bytes (i.e. a stream of data).

The SOURCE TRANSPORT number and the DESTINATION TRANSPORT number are used to uniquely identify the sender and receiver processes. In this way for any particular address (i.e. VME module or ECA) there can be a number of processes communicating.
independently, which permits multi-tasking in VME modules and ECAs.

The SESSION ERROR byte is used only when a communication error occurs and contains information about the type of fault which occurred.

**Services Provided**

All services are implemented as a layer of software above the basic packet protocol [2].

**Command/Response Service**

This is the principal service used by an application program to communicate with ECAs connected to the 1553 bus and with VME modules. All addressing is via a name for the ECA and not via the physical address of the ECA. In this way changes of hardware configuration do not require changes to be made to application software. The Equipment Directory Unit contains the address of each named ECA. These names and addresses are defined at startup of the PCA or dynamically later on (via a special form of the Command/Response Service which implicitly goes to the Equipment Directory Unit).

A name is in fact composed of a FAMILY and a MEMBER, each of which is a string of up to 6 characters. For example the ECA with family MAGNET and member MAIN might have the physical address 00 5A 22. Many names may have the same physical address.

The first step in a Command/Response transaction with an ECA is to interrogate the EDU for the physical address of the ECA. This is performed by sending a packet of type OpenEquip(ment) to the EDU with the data containing the FAMILY and MEMBER. If receiving this packet the EDU does the following:

1) It looks up the physical address of the ECA with FAMILY, MEMBER.

2) Checks that the ECA is connected.

3) Checks that the ECA is not already busy with another Command/Response transaction. Simple ECA software can only deal with one transaction at a time. Sophisticated ECA software can perform multiple transactions so this check can be overridden.

4) Lock the ECA to stop other transactions.

5) Send a packet of type ReturnOpen back to the sender containing the physical address of the ECA or if any of the checks failed, an error message.

The next step is to send the command and any data to the ECA. A strict format has been defined for this:

**FAMILY,MEMBER,MODE,SERIAL,CAPABILITY!data**

ACTION is used to tell the ECA what to do with the data. MODE indicates the direction of data transfer (write or read), the type of the data (real, integer or string) and whether the data is transferred as binary data or as ASCII text.

The basic modes supported are:

- **RS/WS** - read/write string
- **RAI/WAI** - read/write ASCII integers
- **RBI/WBI** - read/write binary integers

**BAR/WAR** - read/write ASCII reals
**EBR/WBR** - read/write binary reals

Binary integers are transferred as 2's complement integers and binary reals are transferred as 32 bit IEEE floating point reals [3]. Binary transfer is efficient but when transferring reals the floating point formats used must be the same for sender and receiver. Numbers sent in ASCII format are separated by commas (,).

**SECTION and CAPABILITY** are used by the ECA to check that the application program has permission to perform the action requested.

The exclamation mark (!) marks the start of any data.

When the ECA receives the command it reads the data if it was a write command. It then forms a reply returning any data if it was a read command.

**!DATA**

The R indicates this is a reply and the exclamation mark marks the start of any data. If the ECA wants to return an error then it would send:

**!ERROR CODE,ERROR STRING**

The E indicates this is an error message and the exclamation mark marks the start of the error code. The comma marks the start of the explanatory error string.

The final step in a command/response transaction is to tell the EDU to unlock the ECA. This is done with a packet of type CloseEquip(ment). The data of the packet indicates which ECA to unlock.

Simple procedure calls from C, Modula-2 or NODAL perform all the necessary low level operations described above for a command/response transaction and take as parameters the family and member of the equipment, the action to perform, the data to send to or from the equipment, the data size and the mode of the transfer (read/write, binary/-ascii, real/integer/string etc.).

**Examples**

To set the current (CUR) of the ECA with family MAGNET and member MAIN to 3 amps the application code would be:

```c
NODAL
SET MAGNET(‘MAIN’,’CUR’)=3
```

Note that the mode, section and capability are implicit in the syntax of NODAL.

**Modula-2**

```c
PROCEDURE Example;(* slightly simplified *)
VAR
info:INFORM;(* a record which contains family,
    member, action and mode *)
tmp:REAL;
error:ARRAY[0..79] OF CHAR;
BEGIN
info.family:=‘MAGNET’;
info.member:=‘MAIN’;
info.action:=‘CUR’;
info.mode := ‘WAR’;  (* Write ASCII Real *)
tmp:=3.0;
(* EQUIP performs a command/response transaction *)
```
And TEST 1, TEST 2 and TEST 5 will receive a multicast data packet containing:

HELLO! magnet main has started

Transmission of Packets over the VME bus

Each processor in a VME chassis has a certain amount of Dual Port VME memory which can be accessed by the other processors. In addition each processor has a 16 word First In First Out (FIFO) buffer which generates a processor interrupt whenever a word is written to the FIFO.

The sender of a packet puts the packet in its dual port memory, writes a word in the receiver's FIFO to tell the receiver that there is a packet for it and the address that it can be found. The receiver then reads the packet via the VME bus, writing a word into the sender's FIFO to indicate when it has finished with the packet so that the sender can liberate the memory used.

Transmission of Packets over the 1553 bus

The 1553 Bus is a 2 wire serial multidrop [4]. The data bits are encoded in Manchester II biphasic code. For distances up to 400 metres it can run at 1 Megabit per second, whereas for distances up to 2 km it can be run at 125 kilobits per second. With the aid of repeaters even greater distances can be covered.

A Bus Controller manages all communication on the bus and up to 30 remote terminal interfaces may be connected. Normally there is one RTI card in each Equipment Control Assembly. Only the Bus Controller can start a data transfer.

Basic 1553 Data Transfers

When writing data to an RTI the Bus Controller sends a 1553 command word followed by up to 32 16 bit words of parity checked data. The 1553 command contains an RTI address, word count, direction indicator (read or write) and a 5 bit field to indicate where the data should go in the RTI memory. Only the addressed RTI receives the 1553 command word and data. The RTI hardware checks that the 1553 command word is legal, that the correct number of words were received and that there were no parity errors. In addition the hardware checks that the bits of each word were correctly Manchester II coded. The hardware puts the data in the correct RTI memory locations and returns a 1553 status word to the Bus Controller indicating the success of the transfer.

To read data from an RTI the Bus Controller sends a 1553 command word specifying the number of words, that it is a read and the memory area to read. The RTI hardware checks the command and sends the data followed by the 1553 status word. The Bus Controller hardware checks the parity and Manchester II coding of the data it receives.

1553 Buffers

The RTI cards used in the LEP/SPS control system have additional hardware to the basic minimum required to implement the 1553 bus protocol. There are two 256 byte or larger First In First Out Buffers and a Control Status Register (CSR). One buffer is used as a packet receive buffer and the other as packet transmit buffer. The RTI Control Status Register can be written and read both by the Bus Controller and the ECA processor. The bits in the CSR allow the BC and ECA to:
1) Indicate receive buffer full/empty
2) Indicate transmit buffer full/empty
3) Reset transmit/receive buffer pointers
4) Interrupt the ECA processor
5) Modify and monitor the state of the RTI card

To send a packet to the ECA the Bus Controller checks that the receive buffer is empty, clears
the receive buffer pointer, transfers the data in
up to 4 32 word transfers, i.e. up to 256 bytes,
sets the receive buffer full bit and the ECA
processor interrupt bit. The ECA gets the inter-
rupt, sees the receive buffer is full and reads
the data from the receive FIFO.

For a Bus Controller to receive packets from the
ECAs it must continually 'poll' all the ECAs on
the 1553 bus, checking if any have the transmit
buffer full bit set. This task occupies almost
100% of the Bus Controllers processing time.
Normally the Bus Controller will only poll the
ECAs it knows are connected but periodically it
will check all the possible ECA addresses (1 to
30). If a new ECA is found then it will be added
to the list of ECAs to be polled frequently. The
Bus Controller sends an alarm to the Alarm server
if an ECA is disconnected or reconnected. In addi-
tion various other states are monitored.

All packets from an ECA pass by the Bus Control-
ner, even if their destination is another ECA on
the same 1553 bus.

Performance

The time for command/response transaction to send
one real from a MODAL application program running
on a NORD 100 to a G64 6809 ECA is 60ms. 10 kilo-
bytes of data can be sent at 20 kilobytes per
second.

A 10 kilobyte command/response transaction from
one G64 6809 ECA to another similar ECA on the
same 1553 bus is performed at 24 kilobytes per
second.

A G64 6809 ECA can broadcast 1 byte of data to all
the other ECAs on its 1553 bus 250 times per
second.

The interface library for a G64 6809 ECA providing
the command/response, transparent, alarm and
broadcast services is smaller than 10 kilobytes.
This is important in a 6809 system because they
can only address 64 kilobytes of memory.

Conclusion

The equipment network of the LEP/SPS control
system provides a variety of services enabling
many types of Equipment Control Assemblies to be
accessed from application programs. Application
programs may run in different environments but
they all have the same software interface towards
the equipment network allowing portability between
current and future configurations of the LEP/SPS
control system.

The speed of the equipment network, despite the
long distances and number of Equipment Control
Assemblies involved should ensure that bottlenecks
do not occur. The interface software is relatively
compact so that small, inexpensive microprocessor
systems can be interfaced.

References

[1] R. Eausch, Real Time Control Networks for the
LEP and SPS Accelerators, Europhysics
Conference on Control Systems for Experimental
Physics, Villars-sur-Ollon, Switzerland, 28
September 1987.

LEP/SPS Controls Group, LEP Controls Note
No. 54, CERN.

[3] A Proposed Standard for Binary Floating-Point
Arithmetic, Draft 8.0 of IEEE Task P754, 1981,
IEEE.

equipements LEP et SPS, Manuel de l'utilisateur
(User Manual of the MIL-1553B Multidrop Bus
for the Control of LEP and SPS Equipment),
D. Francart, R. Eausch, J.M. Sainson et
E. Wilhelm, LEP Controls Note No. 56 Rev., CERN.