PERFORMANCE OF THE TRISTAN COMPUTER CONTROL NETWORK

H. Koiso, A. Akiyama, T. Katoh, E. Kikutani, S. Kurokawa, and K. Oide
National Laboratory for High Energy Physics (KEK)
Oho-machi, Tsukuba-gun, Ibaraki-ken, 305 Japan

M. Shinomoto, N. Kurihara, and K. Abe
Omika Works, Hitachi Ltd.
Hitachi-shi, Ibaraki-ken, 319-12 Japan

Abstract

An N-to-N token ring network of twenty-four minicomputers controls the TRISTAN accelerator complex. The computers are linked by optical fiber cables with 10 Mbps transmission speed. The software system is based on the NODAL, a multi-computer interpreter language developed at CERN SPS. Typical messages exchanged between computers are NODAL programs and NODAL variables transmitted by the EXEC and the REMIT commands. These messages are exchanged as a cluster of packets whose maximum size is 512 bytes. At present, eleven minicomputers are connected to the network and the total length of the ring is 1.5 km. In this condition, the maximum attainable throughput is 980 kbytes/s. The response of a pair of an EXEC and a REMIT transactions which transmit a NODAL array A and one line of program 'REMIT A' and immediately remit the A is measured to be $95+0.039x$ ms, where x is the array size in byte. In ordinary accelerator operations, the maximum channel utilization is 2%, the average packet length is 96 bytes and the transmission rate is 10 kbytes/s.

Introduction

An e^+e^- colliding beam facility, TRISTAN[1], now under construction at KEK, is controlled by a highly-computerized control system[2]. Twenty-four 16-bit minicomputers (Hitachi HIDIC 80's) are distributed around the accelerators. These computers are linked together by optical fiber cables to form an N-to-N token ring network (see Fig. 1). In this paper, we explain the structure of the network system and discuss its performance.

NODAL system

The software system of the TRISTAN control is based on the NODAL[3] interpreter originally developed at the CERN SPS. NODAL has the multi-computer facility: the syntax of NODAL enables us to make a program composed of subtasks which can be executed on other computers. An example below illustrates how the multi-computer facility works:

```
1.1 DIM A(10)
1.2 EXEC<MGO> 2 A ; WAIT<MGO>
1.3 FOR I=1,10 ; TYPE A(I)!
1.4 END

2.1 FOR I=1,10 ; SET A(I)=MAG(I,'CUR')
2.2 REMIT A
```

When a computer, for example OPO, interprets this program and encounters the EXEC command in the line 1.2, OPO sends the program lines 2.1 and 2.2 and the array A to the computer MGO, which interprets these lines and sends back the array A to OPO by the REMIT command. Then OPO types out the value of A on its terminal.

The KEK version of NODAL[2,4] has the multi-computer file system, under which we can uniformly access any NODAL program files, data files, and I/O devices attached to different computers. For example, we can print out the program file TEST stored in the disk of computer OPO to the line printer of computer DVO as follows:

```
OLD OPO/TEST
OPEN(11,'W','DVO/LP:')
LIST<11>
CLOSE(11)
```

TRISTAN computer network

The high-level services offered to the users in the TRISTAN computer network are: (1) remote execution by one of the EXEC, EXEC-P or IMEX commands and its answer by the REMIT command, and (2) uniform file access throughout the system.

The network system has a four-layered structure: (1) DFW (data-free way) as the lowest, (2) DPCS, (3) VCAM, and (4) MTS as the highest layer (see Fig. 2).

DFW

DFW is a high-speed token ring network for distributed industrial control made by Hitachi, Ltd. It connects stations by optical fiber cables, on which the

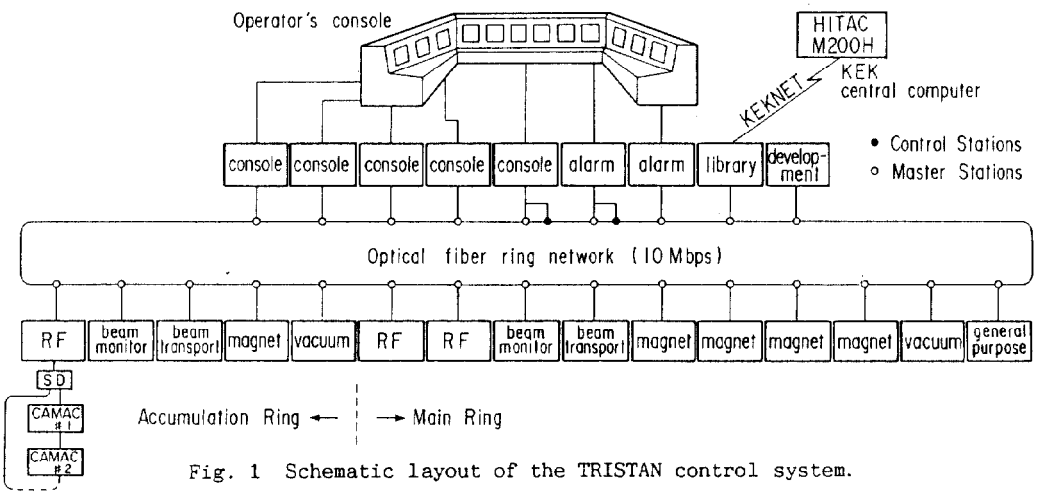

Fig. 1 Schematic layout of the TRISTAN control system.

0018-9499/85/1000-2068$01.00© 1985 IEEE

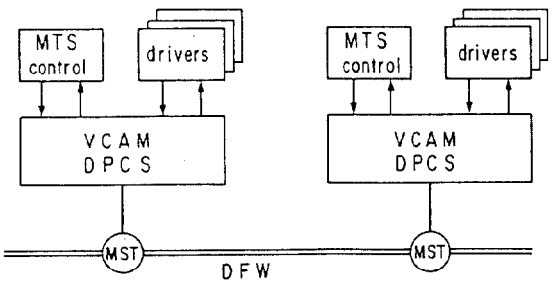

Fig. 2 Structure of the TRISTAN network.

transmission speed is 10 Mbps. Two types of stations are used in the TRISTAN network: CST (control station) which controls message flow on DFW, and MST (master station) to which HIDIC 80's are connected.

Data-link procedures of DFW are as follows:

(1) In usual case, a set of patterns GA (go-ahead) and NPOL (normal-polling) which is sent as a free token by CST circulates on the ring.

(2) If some source MST wants to send a message to the ring, it first catches GA and NPOL; then the MST changes NPOL to the pattern RSV (reserve) and sends an I-frame (information frame) which contains destination address and data to be transmitted; the set of GA and RSV forms a busy token. If more than one MST's claim data transmission simultaneously, the upstream MST has the priority.

(3) Every MST receives the I-frame and checks the destination address. If the address matches its station address, the MST gets the contents of the I-frame; otherwise it passes the I-frame to the next MST.

(4) The destination MST checks if there are any errors in the I-frame; then it sends the reply information as a RESP-frame (response-frame) followed by GA and RPOL (retry-polling). The RESP-frame contains the error status of the I-frame.

(5) The source MST receives and checks the RESP-frame. If no errors are detected, it destroys the RESP-frame and rewrites the RPOL to NPOL and frees the channel; otherwise, it changes the RPOL to RSV and retries the message transmission from the procedure (2).

The format of an I-frame is shown in Fig. 3, where LC is the field for loop commands which instruct the destination MST how to deal with the information contained in the I-field. The loop commands used frequently in the TRISTAN network are:

TR --- transfer message,
RPL --- request to write the memory on the computer connected to the station,
RPLA --- answer of the RPL,
RRB --- request to read the memory on the computer connected to the station,
RRBA --- answer of the RRB.

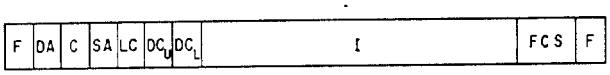

Fig. 3 Format of an I-frame. F: flag indicating the start and stop of a frame, DA: destination address, C: command used to identify an I-frame, SA: source address, LC: loop command, DCU and DCL: data count, I: information, and FCS: frame check sequence.

The destination MST which has received the loop command RRB reads the contents of the memory by DMA and sends them to the source MST as an RRBA message. This process has the small overhead because it is transacted by micro-programming software on the destination MST.

DPCS

DPCS is a control software embedded in PMS, a real-time multitasking operating system of HIDIC 80 series computer. The DPCS supports data communication between tasks on different computers. Tasks use macro commands RCOM/WCOM (read/write communication) to receive/transmit data. Maximum size of data is limited to 512 bytes. One call of RCOM/WCOM corresponds to one loop command TR of DFW.

Another function of DPCS is remote read and write of memories on different computers. Tasks use macro commands RFIL/WFIL (read/write file). One call of RFIL/WFIL corresponds several pairs of loop commands RRB/RPL and RRBA/RPLA of DFW.

VCAM

Under DPCS, the transmitting task cannot check whether data has been successfully received by the receiving task. Under VCAM, the task which has received data sends an acknowledging message to inform the transmitting task of data acceptance; therefore one transmission of data corresponds to a pair of loop commands TR's of DFW.

MTS

The main functions of MTS are as follows: (1) Using VCAM, it controls the data transmission on the network. (2) For the transmission of large amount of data such as NODAL programs and variables for remote execution, MTS directly uses DPCS in order to minimize the overhead. (3) MTS facilitates the access to files in the system uniformly using logical file numbers.

As shown in Fig. 2, MST consists of an MTS control task and various driver tasks: for example, the program driver for NODAL EXEC and REMIT commands, the NODAL driver for load and save of NODAL programs, the device drivers such as the line printer driver, etc. The messages are exchanged between the MTS control task and a driver task on different computers via DPCS and VCAM.

Protocol of the NODAL EXEC command

To show how MTS works, the procedure of the NODAL EXEC command from computer A to computer B is explained below:

(1) When the NODAL interpreter task encounters an EXEC command, it calls the MTS subroutines; then the control is transferred to MTS.

(2) The MTS control task on A sends a message to the program driver task on B using VCAM; the message contains the address and size of the area where programs and variables to be transmitted are stored.

(3) The program driver task on B reads the contents of the area on A using the RFIL macro of DPCS.

(4) After the completion of RFIL, the program driver on B sends the message to the MTS control task on A using VCAM to inform that it has completed the memory read action.

Performance of the network

The CST traces the following information on 4-

kbyte RAM: (1) the number of turns of NPOL, and (2) the source and the destination addresses, the loop command and the message length of I-frame. We can read the RAM with NODAL user functions. The contents of the n-th block of the RAM are set in the array A as:

```
CALL DFWRAM('OPO','CST')
CALL DFWBUF('OPO','CST',n,A).
```

This on-line tracing facility is a powerful tool for diagnosis of the network.

The throughput is determined by three parameters: the message length (M), the loop length of the ring (L) and the number of stations (N). A message of longer than 512 bytes is divided into some I-frames. The time for transmission of one I-frame, T_t, is given by $82.6+5.0L(\text{km})+0.80M(\text{byte})+1.8N$ μs; then the throughput is given by M/T_t Mbytes/s. Figure 4 shows the throughput as a function of M in the present network ,where L is 1.5 km and N is 13 (two CST's and eleven MST's).

The channel utilization of the network is calculated by a ratio, mT_t / (mT_t + nT_n), where m is the number of transmitted I-frames, n is the number of turns of NPOL and T_n is the time for one turn of NPOL, which is given by $15.0+5L+1.8N$ μs. Figure 5 shows the variation of this ratio in ordinary accelerator operations, where the maximum utilization was 2.0%. At that time, the average message length was 96 bytes and the transmission rate was 10 kbytes/s. We can estimate the channel utilization of the complete system to be 10-13%, assuming that L is 10 km, N is 26, and transmission rate is 50 kbyte/s.

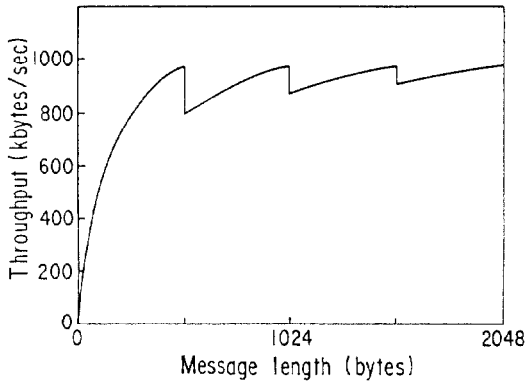

Fig. 4 Throughput of the present network.

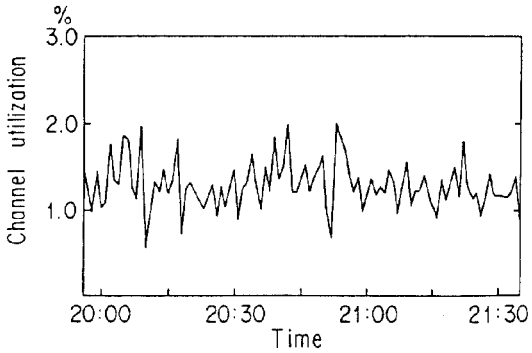

Fig. 5 Example of the channel utilization in ordinary accelerator operations.

The response of EXEC commands was measured in two typical cases: (1) transmitts only the END line and (2) transmitts an array of 2048 bytes and remitts the array immediately as:

```
(1)  1.1 EXEC<LBO> 2 ; WAIT<LBO>
     2.1 END

(2)  1.1 DIM-I A(1024)
     1.2 EXEC<LBO> 2 A ; WAIT<LBO>
     2.1 REMIT A
```

The results were 92 ms for (1) and 175 ms for (2). We also obtained the response of $95+0.039x$ ms for (2), where x is the array size in byte. Since most of EXEC transactions transmit data of less than 2 kbytes (moreover, three-fourths of them transmit data of less than 500 bytes), the EXEC response of the present system is typically 100-150 ms.

Figure 6 shows exchange of I-frames observed in the EXEC and REMIT transactions of (2). The width of an arrow is proportional to time for transmission of an I-frame. Because there remains enough unoccupied time between I-frames, the network maintains the good response even if some pairs of computers execute EXEC and REMIT transactions at the same time. We measured also 175 ms response, when three simultaneous EXEC transactions of the type (2) between three pairs of computers were processed. In this case the utilization of the network was 12.5%.

Acknowledgements

We wish to thank Professors Y. Kimura and G. Horikoshi for their encouragement and support during this work. We also thank the members of the TRISTAN control group and the TRISTAN operation group for their useful discussion and advice.

References

[1] T. Nishikawa and G. Horikoshi: "Status of KEK TRISTAN Project", IEEE Trans. Nucl. Sci. 30 (1983) 1983.

[2] A. Akiyama, et al.: "Computer Control System of TRISTAN", in Proceedings of the Europhysics Conference, Computing in Accelerator Design and Operation, Berlin, September 1983, in Lecture Note in Physics 215, Springer-Verlag 1984, pp.367-371.

[3] M.C. Crowley-Milling and G.C. Shering: "The NODAL System for the SPS", CERN 78-07 (1978).

[4] S. Kurokawa et al.: "KEK NODAL System", contributed paper to this conference.

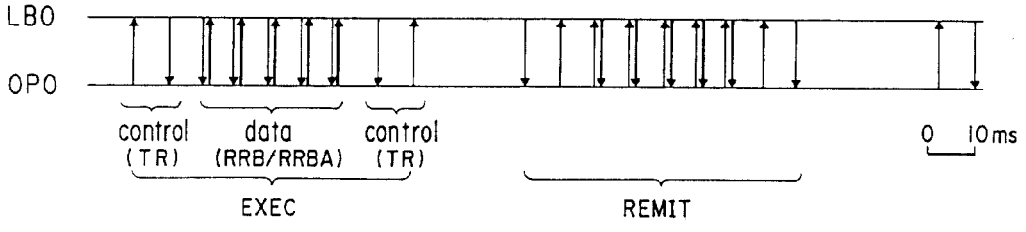

Fig. 6 Exchange of I-frames on the EXEC and REMIT transactions.