Journal of the Korean Nuclear Society
Volume 30, Number 4, pp.298~307, August 1998

A Communication Method Between Distributed Control System and Function Test Facility Using TCP/IP and Shared Memory

Jung-Soo Kim, Chul-Hwan Jung, Jung-Taek Kim, and Dong-Young Lee, Chang-Sik Ham

Korea Atomic Energy Research Institute
150 Dukjin-dong, Yusong-gu, Taejon 305-353, Korea

(Received October 1, 1997)

Abstract

In order to design mutual communication between a distributed control system and a function test facility, we used the Inter-Process Communication(IPC) in two systems and Transmission Control Protocol/Internet Protocol(TCP/IP) protocol. The data from the function test facility are put in the shared memory using an IPC, which is then accessed by the distributed control system through an Application Program Interface(API). The server in the function test facility includes two processes(one for sending and one for receiving), which are generated by the fork function from the client signal. The client in the distributed control system includes two separate programs(one for receiving and one for sending).

1. Introduction

Generally, we have used the TCP/IP[1,2] for communication among workstations. We can also use the socket function for designing the TCP/IP.

The function test facility[3] includes a software-code which simulates the Nuclear Power Plant. This facility will be utilized for testing or for providing data for the auto-startup system or the alarm system.

The auto-startup system[4] consists of a Distributed Control System(DCS) and a G2 which is a real time expert s/w tool. The auto-startup system is capable of automatically operating the nuclear power plant at the startup stage. Figure 1 shows the hardware configuration of this system. For its development, we designed the communication method between DCS and FTF. This system is designed to automatically shift a plant' s parameters to function as a general operation guide in the startup operation mode.

The startup operation mode consists of four operational modes: the operation mode from cold shutdown to hot shutdown, the operation mode from hot shutdown to hot standby, the operation mode of secondary system' s heatup/startup, the operation mode from hot standby to 2%. Mostly, the operations at these startup modes in existing nuclear power plants

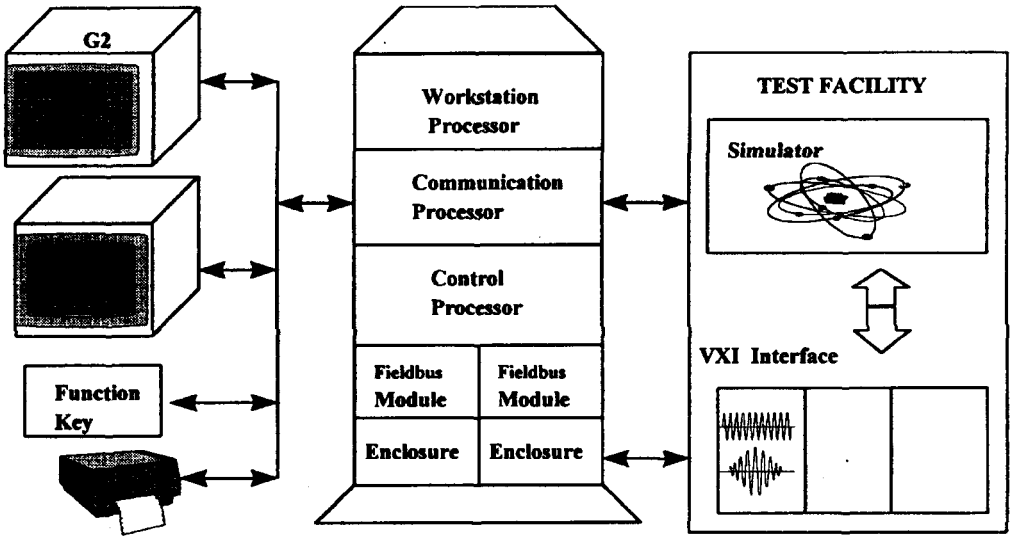

Fig. 1. Hardware Configuration of Auto-Startup System

have been manually controlled by the operator. The manual operation of this stage increases an operator' s burden, thereby causing operator mistakes to occur.

The communication method developed in this paper is constructed to receive the necessary data during startup operation mode from the function test facility to process this data in the auto-startup system and to send the distributed control system' s control data to the function test facility. To develope the communication method, we used a shared memory for receiving/sending the data between the two systems. This shared memory links the DataBase(DB) at the two systems. The shared memory is applied to put in the necessary data from the DB. In order to use the shared memory from the distributed control system, we utilized an API[5] software(s/w) tool. The function of the API is similar to that of the IPC in other workstations. This tool is provided by the Distributed Control System(DCS). We modified the API S/W to be used properly. Also, in the case of the function test facility, we used the IPC' s[6] function to be provided at the HP Workstation. The function of the IPC is to approach the shared memory in the function test facility.

This communication method is designed for two-server processes and two client programs. The server in the function test facility includes two processes : one is the sending process, the other is the receiving process. These two processes are generated by the fork function from the client signal. The client in the distributed control system includes two separate programs; a receiving program and sending program. We designed individual two-server processes and two-client programs in order to send or receive needed data at two systems simultaneously, the scan rates of the two systems being different(The distributed control system' s scan rate is 0.5 seconds while the function test facility' s scan rate is 0.2 seconds). This scan rate is unrelated to the number of parameters. Figure 2 shows the data flow between the distributed control system and the function test facility.

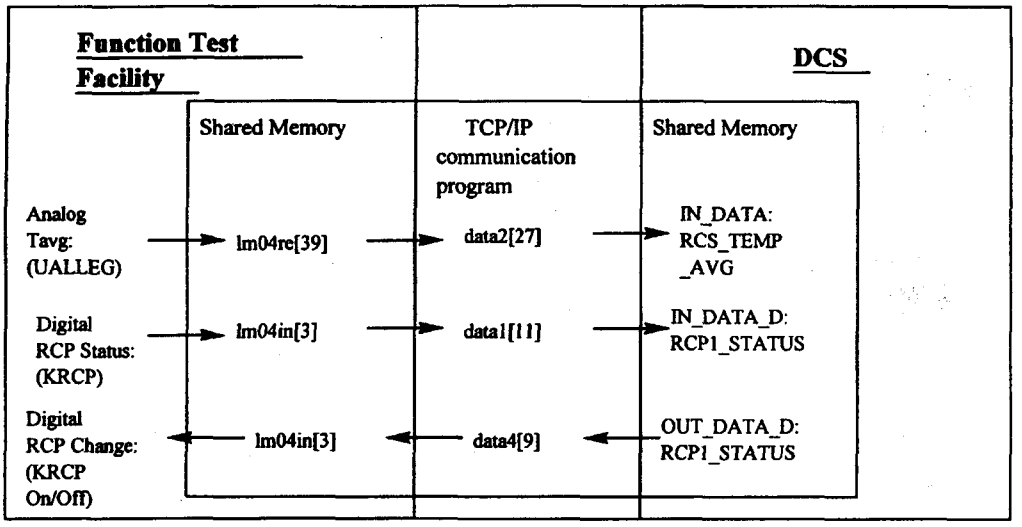

Fig. 2. Data Flow Between the Distributed Control System and Function Test Facility

2. TCP/IP Communication Programming[1-2]

We needed three communication procedures : connect-initialization, data-transfer and stop-connection, for the data-exchange. In the case of TCP/IP communication, we have made a distinction between the connection-oriented protocol and the connectionless protocol. If we use the connection-oriented protocol, it is necessary to connect the communication layer before the data-exchange. Figure 3 shows the general connection-oriented TCP/IP programming structure. The role of functions from Figure 3 are as follows: ① socket( ) : this function is applied to both the server and client. The role of this function is to determine the domain of communication(for example, Internet domain or Unix domain) and the type of communication(TCP or User Datagram Protocol(UDP)). ② bind( ) : this function is used only by the server side. The role of this function is to decide the socket address and its size. ③ listen( ) : this function is used only by the server side. The role of this function is to inform the client of the end-time point of the connection-ready. ④ accept ( ) : this function is utilized only by the server side. It has three roles: to identify the address of the client and its size from the connect( ) function; to return the socket descriptor; and server waits until the other client requests a connection. ⑤ connect( ) : this function is used only by the client side. Its role is to correspond with the accept( ) function of the server side, to determine this server' s address and size.

3. The Development of Communication Method Between Distributed Control System and Function Test Facility

3.1 The Communication Method of Function Test Facility(Server)

3.1.1. The Method of Achieving Shared Memory at Function Test Facility

Before the data exchange, we need to know how to access the shared memory and get/give

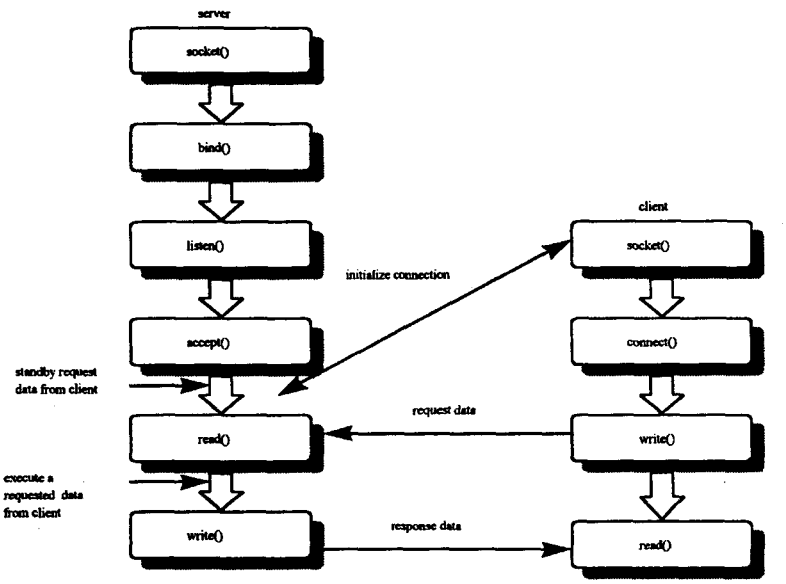

Fig. 3. The General Connection-oriented TCP/IP Programming Structure

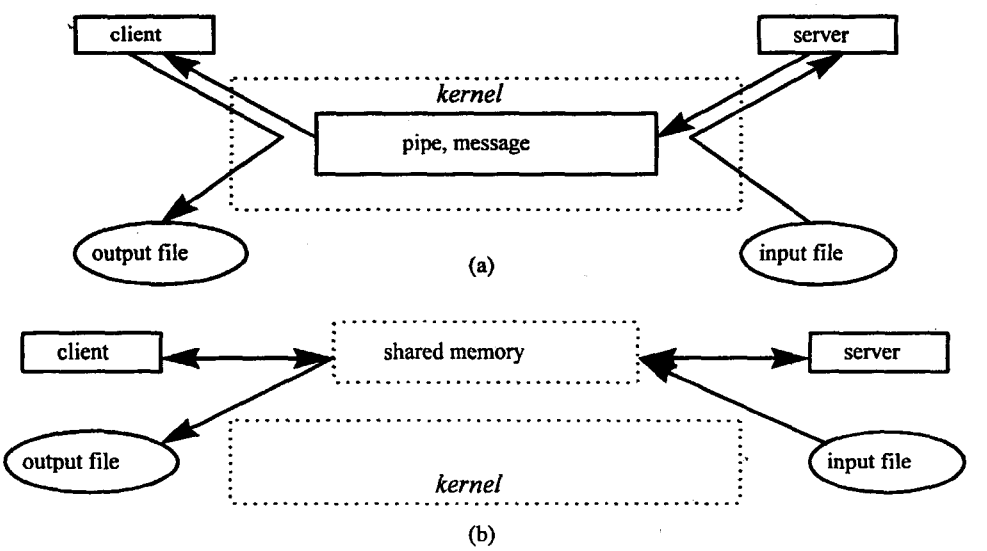

Fig. 4. (a) Data Exchange Using Kernel (b) Data Exchange Using Shared Memory

the data from/to the shared memory. The reason we use a shared memory is to separate the main program in the function test facility from the communication program and easily extract the needed data from the DB of the main program using an IPC' s function without going through the kernel. The merit of using the shared memory is that it lessens the communication load on the inside of the workstation.

We used the IPC function provided by the HP Workstation. The functions utilized for accessing the shared memory are as follows : shmget( ),

shmat( ), shmdt( ) and shmctl( ). The detailed role of these functions are described in the reference[6]. Figure 4 compares the communi cation method using the kernel with that using a shared memory.

3.1.2. The Construction of Communication Method

After accessing the shared memory, we started the data-exchange. We designed the communication program to two slave servers behind a master because we separated the sender process from the receiver process. Two processes are generated by a fork function. The fork function is executed when a signal to connect is requested by a client. The general server-client communication from Figure 3 sends the data in compliance with the request of the server. However, if we applied this method to our system, there are some problems because the scan rates of the two systems are different. If we applied the general method to our system, the client sends the data to the server without the data-change of the client by request of the server. We then send the data from the client to the server unconditionally, without the request of the server, when the data of the client are changed. The merit of this method is that the server immediately knows the time when the data of the client is changed. It is possible to design the real-time control system if we use this proposed method. Figure 5 shows the total construction of the server. The communication execution is as follows: connect initialization, execute the fork function, recognize connection, take/save the data and get/send the data. The initialization of connection starts when the client sends the connection-signal to the server. The server then gets the signal and executes the fork function by the process itself. The fork function generates two processes. The recognization of the connection is executed so that two processes open two-separate communication ports. The slave servers share the FTF' s shared memory for taking/saving or getting/sending the data of the server and sending it to or receiving it from the client using an IPC' s function uniformly.

3.2. The Communication Method of Distributed Control System(Client)

3.2.1. The Method of Accessing Shared Memory at Distributed Control System

We utilize the API s/w tool for accessing the

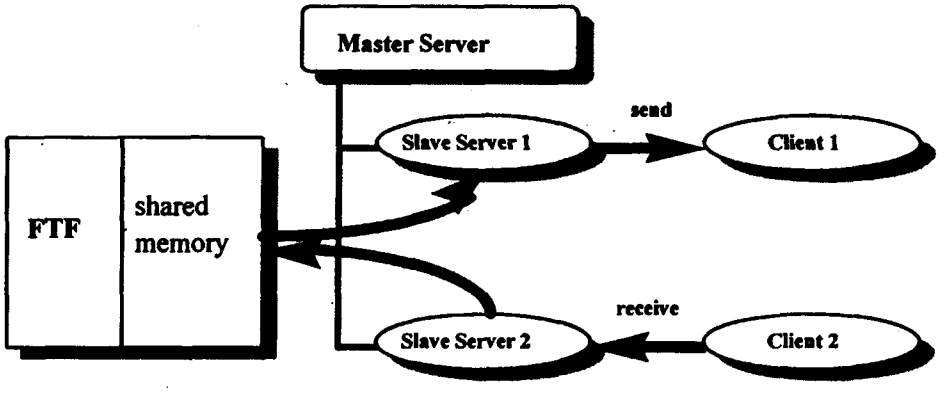

Fig. 5. Total Construction of Server

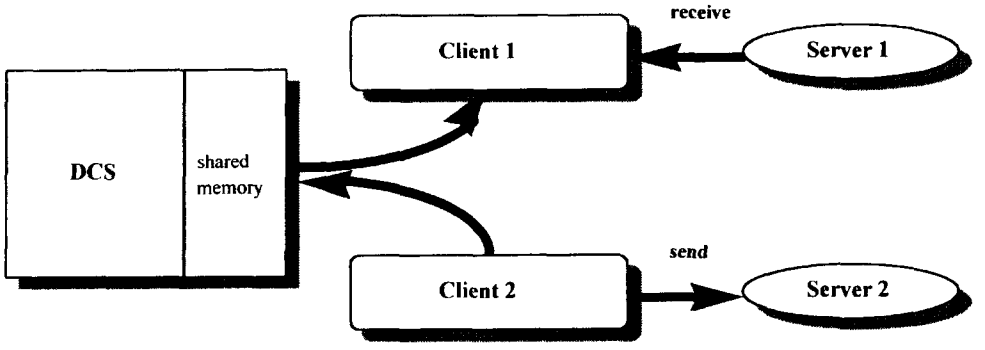

Fig. 6. Total Construction of Client

shared memory at the DCS. The API s/w tool is linked between the shared memory and the DCS' s DataBase. The DB of the DCS is called by the Object Manager(OM). The OM manages all the data of the DCS. We use the API s/w tool in the next procedure. First we generate the data file, which includes the needed data name. For example, steam generator level value (= compound name: block name. MEAS or OUT or SPT)=IN_DATA:Steam_Gen.MEAS and construct the API environment. We assign the shared memory and open the generated data file by API' s open procedure. Then the data of the DB accesses the shared memory. We use the read/write function(readval( ), wrtval( )) for connecting the communication program to the shared memory and this communication method is linked and compiled by the DCS' s library function. It is then completed for accessing the shared memory and is ready for communication.

3.2.2. The Construction of Communication Method

After accessing the shared memory of the DCS, we start the data-exchange. We design the client to two clients correspondent to two servers. The difference between client and server is that the client does not execute the fork function but does execute two individual programs, i.e. receiving program and sending program, due to the characteristics of the client system. Each client program shares the shared memory at DCS. Figure 6 shows the total construction of the client. The communication procedure is the same as that of the server, but is different in that the client sends the start-signal to the server for receiving/sending at the connect-initialization' s phase. The client does not execute the fork function.

4. Experiment and Analysis of a Developed Communication Method

We analyzed the normal operating procedure[7] for developing the distributed control system. As a result of the normal operating procedure analysis, the distributed control system received about 300 variables(the sum of the analog and digital parameters) from the FTF and sent it about 60 variables. We designed the construction of the communication method using the following procedure. First, we placed 200 variables in the FTF' s shared memory among the data of the FTF(1200 variables). Second, we wrote the needed data file

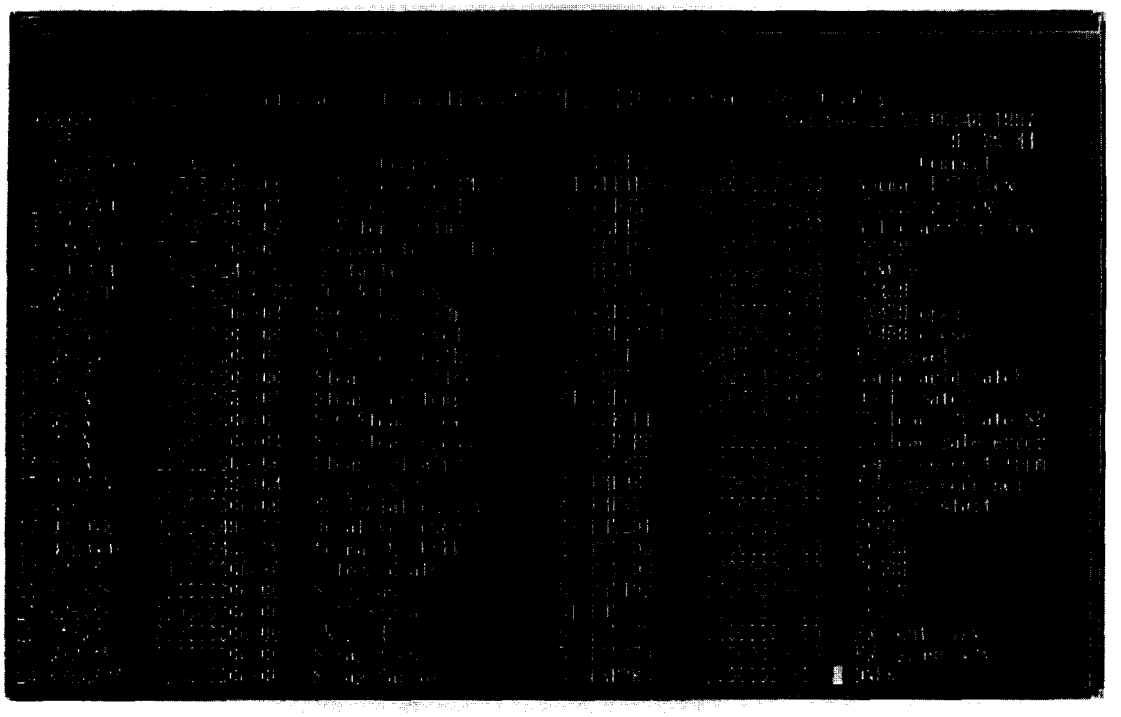

Fig. 7. Executing the FTF at Startup Mode

with the client and put the written data in the shared memory using API S/W. Third, we executed the communication program to send/receive data from the shared memory at two systems. We experimented all data(300 variables) from the FTF. These data are read by the communication program from the FTF' s shared memory, one by one, and the communication program sends these data to the distributed control system The communication program in the distributed control system receives these data from FTF and puts in the DCS' s shared memory one after another. The reverse is the same procedure. For example, in the case of RCS Tavg data(analog), the data name of FTF is UALLEG and the shared memory' s name is lm04re[39], correspondent to UALLEG. The communication program reads lm04re[39]' s data. The communication program changes lm04re[39] to data2[27] for communication. We receive data2[27] from the communication program and the communication program put in the distributed control system' s shared memory, which changes the name of the data2[27] to IN_DATA: RCS_TEMP_AVG.MEAS. The reason the name is changed at each stage is because we separate the name of the main program from the name of communication program and it is different for the method to give a name at two systems. Figure 7 shows that the FTF is executed at the startup mode. This figure displays more than 200 variables, however, we put only 200 variables needed at the distributed control system into the shared memory. For reference, the FTF can execute about 1200 variables during normal operation. Figure 8 shows the system overview developed at the distributed control system. This figure' s data are

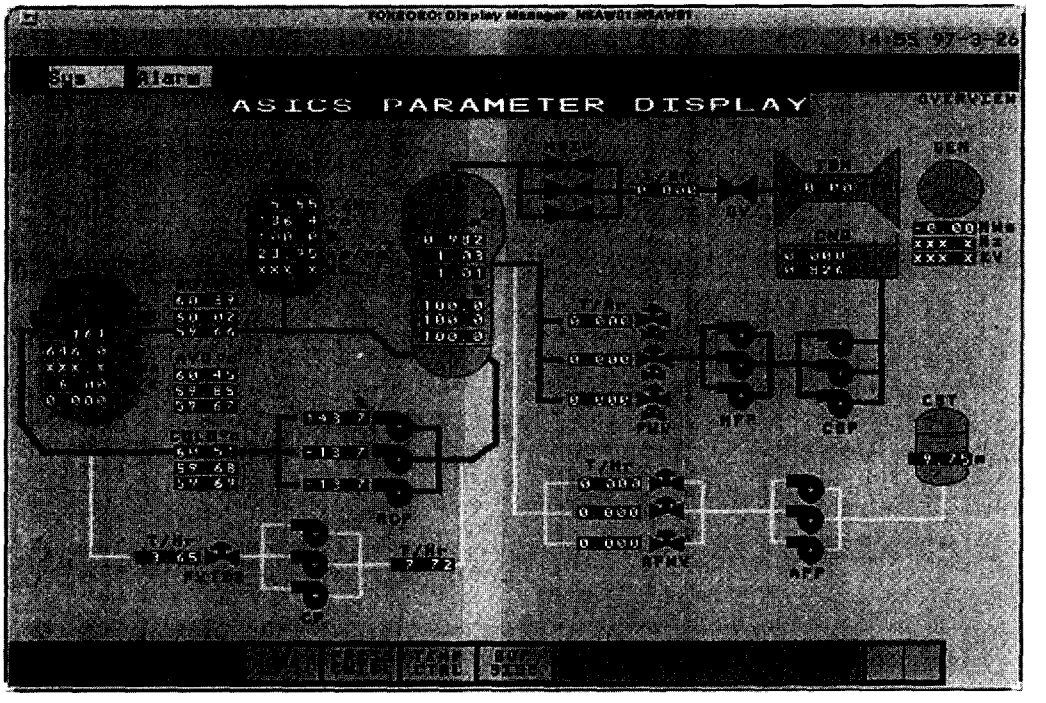

Fig. 8. Auto-startup Overview Display

received from the FTF by the communication program. The analog data in this figure display the digit value and the digital data display the color. For example, the Temperature is ***.* C and the RCP pump status is red(green : off, red : on). These data sent by the FTF are put in the distributed control system' s shared memory and saved in the distributed control system' s DB. This DB is connected to the distributed control system system' s graphic displayer. From Figure 8, we know that these data are correctly received from FTF to distributed control system within the setting up time. Figure 9 represents the Mode I status. Mode I refer to the stage from cold shutdown to hot shutdown. Figure 9 display Mode I' s plant parameter. From Figure 9, the temperature and pressure parameters must follow a Pressure-Temperature. We designed the Proportional Integral and Derivative(PID) controller in the distributed control system which was designed to follow a P-T automatically.

5. Conclusions

A distributed control system needs plant data to design the controller and we get this data from the FTF. In this paper, we developed a communication method using TCP/IP protocol and shared memory. The methods of sending/receiving the plant data in the shared memory at two systems utilize the IPC function in FTF and API s/w tool in the distributed control system, respectively. The merit of the developed method is that its lessens the communication load at the workstation because it uses the shared memory. Furthermore, this program easily accesses the DB. We modified the general TCP/IP protocol because the scan rates of the two systems are

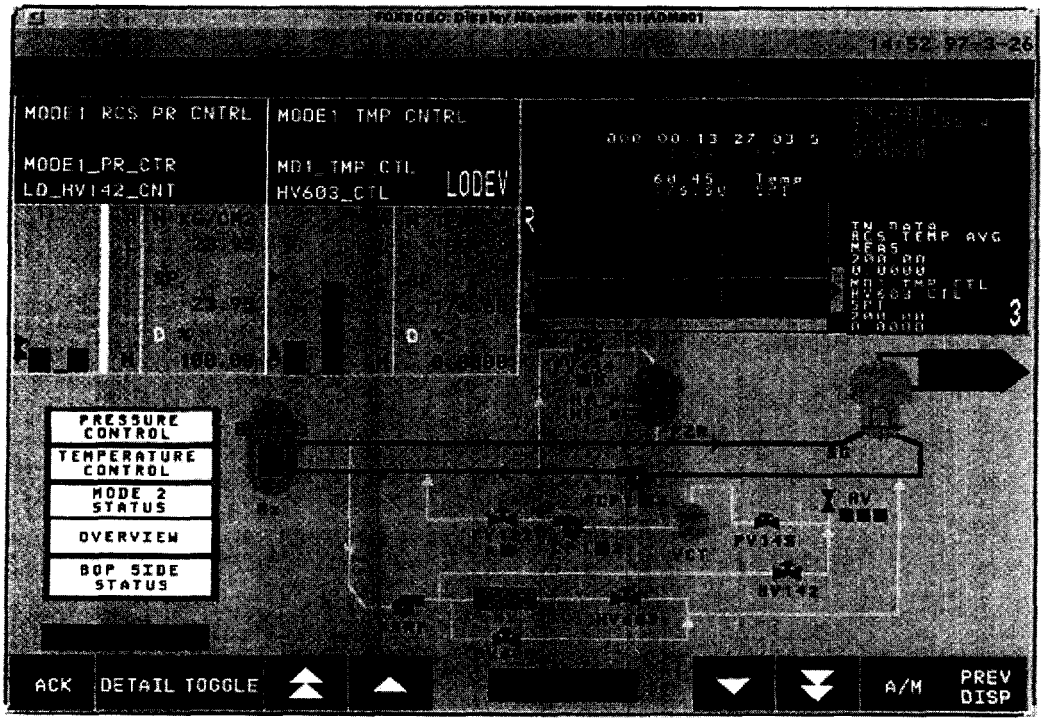

Fig. 9. Mode I Status Display

different. We designed a way to send data from the auto-startup system to the FTF without the request of the FTF unconditionally when the data of the distributed control system changes. Also, using this method, the relevant data of the FTF changes immediately when the data of the distributed control system are changes.

If we use this proposed method, it is possible to design the real-time control system. If we apply this method to a different system and reform the method of accessing the shared memory used in a different DCS, this method can be used with ease because of the general protocol and method of handling the shared memory. Also, if a Nuclear Power Plant is fully-digitalized in the future or the automatic startup system is applied to the next generation Nuclear Power Plant, this system can be used.

References

1. Sang-Yol You, TCP/IP Internet(theory, programming, the useful method), Song-An Dang, (1994).
2. Stevens, UNIX Networking Programming, Prentice-Hall, (1991).
3. KAERI/RR-1504/94, The Development of Verification and Validation Technology for Instrumentation and Control in NPP' s, July. (1995).
4. KAERI/RR-1503/94, The Development of Basic Technology for Instrumentation and Control, July. (1995).
5. Foxboro Manual, Application Interface Software Programmer' s Guide for 50 Series System, Foxboro, (1993).

6. HP-UX. Manual, HP-UX. Reference Volume 2, Hewlett-Packard, (1991).
7. KORI 3 & 4, General Operating Procedure, (1984).
8. Jung Soo Kim, Chul Hwan Jung, and Chang Sik Ham, "The development of communication program between FTF and auto-startup system", Proceedings of the Korean Nuclear Society Spring Meeting, pp. 499 - 504, March. (1996).