A Development of DCS Binding Delay Analysis System based on PC/Ethernet and Realtime Database

Kwi-Yil Gwak*, Sung-Woo Lee*, Yong-Hun Lim*, Beom-Seok Lee*, Duck-Hwa Hyun*

*Department Korea Electric Power Research Institute, Daejeon, Korea
(Tel : +82-42-865-5972; E-mail: psalmist@kepri.re.kr)
(Tel : +82-42-865-5973; E-mail: swlee@kepri.re.kr)
(Tel : +82-42-865-5972; E-mail: adsac@kepri.re.kr)
(Tel : +82-42-865-5972; E-mail: leebs@kepco.co.kr)
(Tel : +82-42-865-5972; E-mail: hyundh@kepri.re.kr)

Abstract: DCS has many processing components and various communication elements. And its communication delay characteristic is affected diverse operating situation and context. Especially, binding signal which traversed from one control-node to another control-node undergo all sort of delay conditions. So its delay value has large deviation with the lapse of time, and the measurement of delay statistics during long time is very difficult by using general oscilloscope or other normal instruments. This thesis introduces the design and implementation of PC-based BDAS(Binding Delay Analysis System) System developed to overcomes these hardships. The system has signal-generator, IO-card, data-acquisition module, delay-calculation and analyzer module, those are implemented on industrial standard PC/Ethernet hardware and Windows/Linux platforms. This system can detect accurate whole-system-wide delay time including io, control processing and network delay, in the resolution of msec unit, and can analyze each channel's delay-historic data which is maintained by realtime database. So, this system has strong points of open system architecture, for example, user-friendly environment, low cost, high compatibility, simplicity of maintenance and high extension ability. Of all things, the measuring capability of long-time delay-statistics obtained through historic-DB make the system more valuable and useful, which function is essential to analyze accurate delay performance of DCS system. Using this system, the verification of delay performance of DCS for nuclear power plants is succeeded in KNICS(Korea Nuclear Instrumentation & Control System) projects

Keywords: DCS, Delay, Performance, Binding, PC, Ethernet, Linux, Windows

1. Introduction

In order to satisfy the requirement of realtime system, the signal processing delay, transmission delay, and response time of the DCS(Digital Distributed Control System) or any other real time control systems must be guaranteed to be less than a limited value. And the measurements and analyses of these delayed performances becomes a necessary required process for verification of realtime system.. However, in the DCS, or in any other control system which has distributed nodes, the synthetic delay time of the control signals(binding signal) which traverse several nodes undergo a multiple processing components constituted of the IO handling process, control process, and communications process. So this delay performance value has a very variable form and show a wide range of fluctuation width.

Although the measuring of any momentary delay value of the signals being transmitted through the communications network is a complex work itself, the gathering of these delay statistics is more complex job, because diverse range of delay fluctuation makes it worse. So this task especially requires steady observation during a certain period, and thus, the presumption, measurement, and analysis of the delay performance of realtime system becomes even more difficult. Normally, the delay time between certain control points are measured by utilizing the oscilloscope or general measuring equipments etc. However, by these traditional devices, it is difficult to gain the statistics of the delay value which shows huge fluctuation, and the measurement on tens and hundreds of multi-channels is also impossible.

In this paper, the design and implementation of DCS delay analyzing system, BDAS(Binding Delay Analyzing System), based on the open system, is introduced in order to solve above problems. To make precise measurement and analysis of the delay time, the DAS(Data Acquisition System), signal generator, signal history recording device, and signal comparing and analyzing device must be prepared efficiently. This BDAS system developed all these functions modules through the open industrial standard PC/Ethernet hardware and the Linux/Windows Platform, which are of great accessibility and low expense. The developed system calculates the precise delay time of the control signals transmitted through the communications network by the unit of 1msec. And unitizing the realtime database, it can shows delay value statistics during some times within multi-channels.

Further on, the chapter 2 presents the background of development and overall composition of this BDAS, chapter 3 and 4 discusses of the distributed IO node and analysis node which are responsible of the handling of inputs and outputs, and chapter 5 and 6 shows the measured examples of delay performances by this system and the conclusion.

2. Background and System Components

2.1 Concepts of Binding Delay and requirements of BDAS

The DCS generally has multi-level communications networks in its structure as field-network, control-network, and information-network. And even when there is only a single-level communications network, the several control nodes on the communications network are connected to each other, and they send and receive mutually the input/output signals or realtime control signals when necessary. The following Fig-1, an example of these network, displays the components of DCS for nuclear power plant, developed by a KNICS project(The information network is excluded from the figure). In the figure, the communications network nodes called PCU(Process Control Unit) are connected through the ring-type control network, and the field control nodes called

FCU(Field Control Unit) are connected through the field network. Like this style, If DCS or other control systems is constituted of communication networks and multiple node's connections, the synthetic time of processing delay and transmission/queuing delay are also composed by a mixture of these elements, and so, the whole-system-wide delay time generating an even more various and complex form.

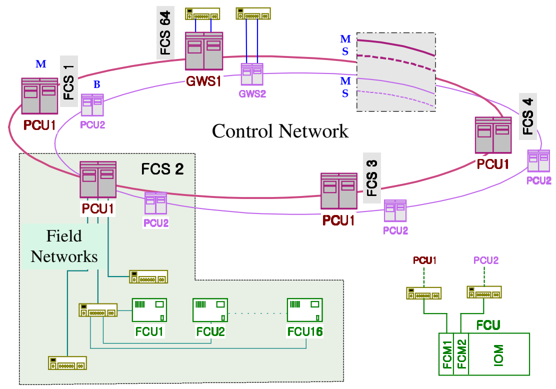

Fig. 1. An example of DCS Networks and Nodes

In the example of the figure above, an distributed FCU has the filed control algorithm and IO cards. And the process of a signal flow, having been input in a certain FCU, being transmitted through a control communications network and output by other FCU, is called 'binding'. The total binding delay of DCS is affected by the binding path, for instance input set FCU --> upstream field network --> control network--> down stream field network-->output set FCU. And its delay value is varied by various factors such as the input/output scan cycle, the network transmission cycle, the communication speed, and the handling speed of the control nodes. So delay time has fluctuation with large depth and changes from time to time and communication circumstances.

Therefore, in order to verify the realtime characteristics of the realtime control system, this binding time must be precisely measured, and it should be confirmable that the whole processing and transmission is finished in a limited time in various circumstances. This BDAS system was designed and implemented under this purpose and requirements considered are as the following.

○ function of handling and distinguishing the 1 msec unit
  - μs time-stamping of input/output process
  - measurement(calculation) of the delay time in msec
○ function for signal generation by auto or manual mode
○ function for real time signals acquisition
○ function for judgment of delay signal and calculation of its value within multi channels
○ function for storing of each signal's time-stamp and delay history
○ function for preparing of statistics, including the average, maximum, minimum value of delay time and its variation

Among above, the 1msec unit requirement came from DCS requirement, which needs to distinguish the SOE(Sequence of Event) by the unit of 1msec.

2.2 Components of BDAS and connection Structure

The overall BDAS system accommodates a PC environment, and it is consisted of the master node(analysis node) which is responsible for generate the input/output commands of the overall system , gathering and analyzing of delay statistics, and sub-node(IO handling node) which is responsible for input/output signal acquisition from/to DCS by according to the commands of the master node.

The sub-node(IO node) is industrial PC operated by Linux OS, and it has PCI-bus IO card. The master node(Analysis nodes) is a single Pentium PC operated by Window 2000. The links between the IO node and the analysis node is the Ethernet.

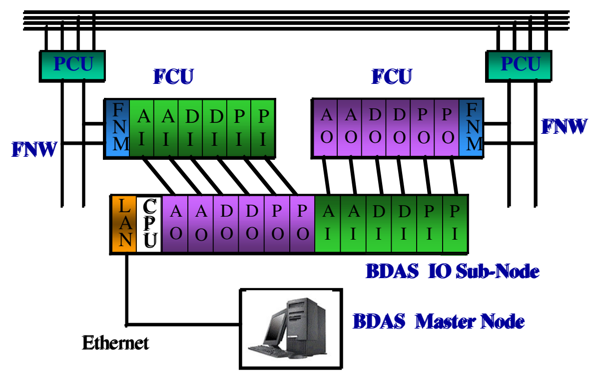

Fig 2. Components of BDAS & its Connection with DCS

The overall measurement process is as the following. ① A command of output signal sequence is created by master-node in real time mode, or a signal creation scenario(batch commands) is downloaded from master-node to sub-node to create output signals in batch mode, that batch commands started by master node ② Then the IO node outputs the actual signals to the output card according to this command and feedbacks its time-stamp and output value to the analysis node. ③ When the output signal traversed through DCS components is arrived to the input card of IO node, the arriving time and values of this signal is transmitted to the analysis node again. ④ The analysis node compares the output signal with the input value, the delay time is calculated, and this information is recorded on the realtime DB. Thus, real time trend of each control point and their statistic value during a certain period could be evaluated through this recording-data from the analysis node.

3. Design & Implementation of the IO Node

3.1 The main compositions and functions of the IO node

The IO node is an industrial PC based on Linux, which accommodates the Ethernet card and IO cards of the PCI-bus. While it is impossible for the Windows OS to precisely control the time by the unit of msec because of its original Event-Drive handling method. But the Linux, even though it is not RTOS, offers PCI IO handling within a certain number of μs, and sufficiently grants the time control or processing by the unit of msec. Therefore, all time-stamping and time control is charged to the IO node in this system. The main modules and functions of the IO node are as the following.

○ communication control module:
module for receiving all traffics descending from the master node(down-stream, i.e., control order, environment file and information of the output scenario, output value etc.) and transmitting the ascending traffics(up-stream, i.e., value of the scan input-signal, output feedback etc.).

○ IO environment & node-state control module:
- according to the received IO environment information (IO card spec, and the number of IO card, etc), it

automatically sets the communications frame structure while considering the received information(environment files) on the IO card size and quantity.

- control the various states(refer to clause 3.3) of the IO node according to the received control codes.

○ IO Processing module :

according to the received real time output signal or batch output scenario, performs the precise input and output by a time unit less than 1msec on the IO card, and sends its input/output value and time-stamp to the master-node via communication control module..

On the other hand, in the Linux system, the controlling the precise time by the unit of 1ms with the general sleep() or usleep(μs-sleep) is impossible. This is because the sleep or usleep function call is managed not by the actual sec/μs unit, but the 1-tick, which is the general minimum time unit generated by the PC board, and is thus dealt by the interval of 55ms(18.2/sec). Therefore, a separate timer-hardware must be adopted, or an additional function to check the precise timing must be developed and used. On this study, the latter method was chosen.

3.2 IO card size & communications protocol

All IO cards equipped in the system are commercial PCI-bus card with card driver developed on the Linux-Kernel-2.4.7. The delegate four types of cards are as the following.

- DI : 64ch, Opto-Isolated, 5μs Max, 12 ~ 24V
- DO : 64ch, Opto-Isolated, 5μs Max, 12 ~ 24V
- AI : 16bit/16ch, A/D conversion 10μs, 0 ~ ±10V
- AO : 16bit/4ch, A/D conversion 10μs, 0 ~ ±10V,

The communications between the IO node and the analysis node is high-speed cyclic with the 1024 Byte fixed-length UDP/IP message packet on the Ethernet. The packet consists of the source and destination address of the distributed nodes, the time-stamp, the control codes, and the actual data. The master node controls the status of the IO node through the control codes here.

3.2 Status transition and tasks of IO node

The IO node can perform various status transition and respective work through the control code as the following figure.

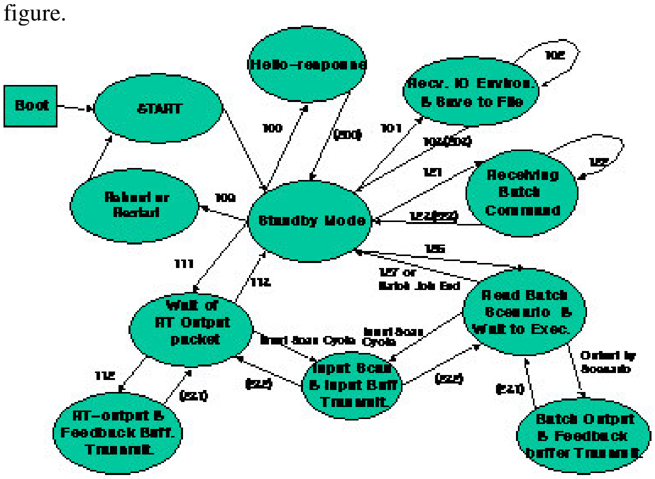

Fig 3. The status transition diagram of IO node

The IO node, which selected its communications packet structure according to the information of the IO card composition at the starting point of it, can inform its alive(Hello-Node) through the control code of the received packet, or can be restarted by receiving the information of the IO card composition again. When the order of the real time output(RT-Output) is received, the real time signal received from the master is instantly made into output, and its output time is printed and re-sent(feedback) to the analysis(master) node. Concurrently with output process, an new additional process(a forked process) observes the input signals and sends changed values to the analysis node instantly when the value is changed, and when there is no change, the present value is periodically transmitted to the analysis node.

The Batch-Output mode is that Output process is created by Linux itself in an orderly form according to the "batch scenario" received from the master. Of course the input process is operated in the same method with above. The batch scenario usually simulates the SOE(Sequence of Event) signals, and is usually utilized to verify whether the SOE signals are handled properly in the DCS system. That is, output signals can not be constantly generated in each 1ms interval from the master node operated on Windows. Therefore, master node creates an output scenario and transmits to Linux instead, and real output of 1ms unit is operated in sequence from Linux according to this scenario. And time-stamp of input/output time is sent to the analysis node so as to verify handling ability and delay time of the DCS. Ⅳ. Design Creation of the Analysis Node

4. Design & Implementation of the Analysis Node

4.1 Main compositions and functions of the analysis node

The analysis node is a general PC operated by Windows OS, and its main functions are generating real time output signals (or creating batch scenarios), saving the input and output signals, and analyzing the delayed performances. Its main function module is composed as the following.

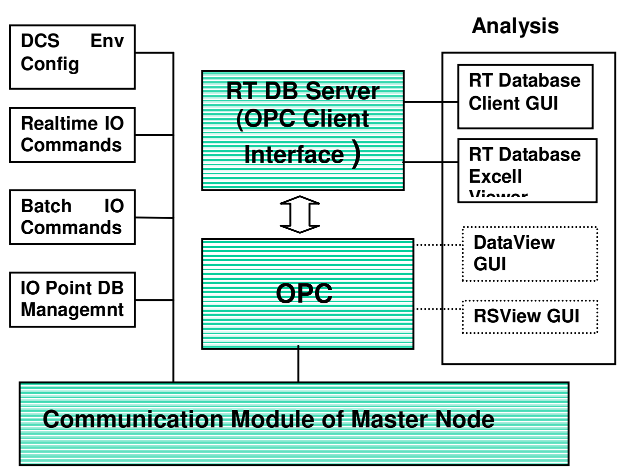

Fig 4. Main Function Modules of the Analysis Node

○ IO Environment Setting Module: creates the IO card information (channels, number of cards etc.) of each IO node and decides the structure of the communications packet. The IO node recreates its packet structure according to this, and is rebooted.

○ Real-time IO Commanding Module: periodically generates certain pattern output signals of single or multiple IO channels. commands to the IO node to output, and stores the timestamps ascended from the IO node.

○ Batch Scenario Commanding Module: creates a series of output orders by the unit of 1ms, sends this to the IO node for actual practice, and stores the signals and timestamps ascended from the IO node.

○ Communications Control Module & OPC Server Module: deals with all communication exchanged with the IO node. The analysis node itself provides the basic OPC server functions. So its real time trend could be observed in an outside network with a certain GUI(Data-View, RS-View, etc).

○ Real Time DB Module: All record data ascended from the IO node are stored in the real time DB, and this data can be retrieved latter and can be analyzed through various utilities afterwards. The PI(OSI Co., U.S.A) complying with the Windows standard is utilized for the RTDB in this system.

○ Analyzing Module: Do retrieval and observation from the real time DB, and make statistical analyses. Each analysis modules were not developed in this system, but the real time trend and statistic analyzing value was provided though the application of RTDB utility and Excel etc. The trend can also be observed in outside network by connecting other GUI tools to the OPC server.

○ Other Function Modules which handles information of IO control points(Tag, max/min value).

4.2 Implementation of analysis node & GUI

All programs of the analysis node are created with C++/MFC. The main program, UDP/IP communications control modules and the OPC server module is a separate thread module. An example of a GUI screen of master node is as the following.

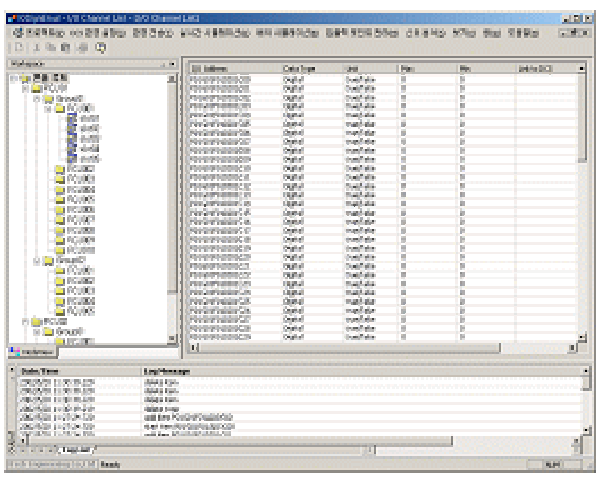

Fig. 5. An example of GUI screen of master node

5. An Example of Performance Test using BDAS

With BDAS, the delay performance test of new DCS product (named KDCS), developed for nuclear power in KNICS project, is done successfully. In this chapter, the evaluation and verification for first trial-product among that of various trial products are briefly discussed.

5.1 The components of KDCS and testing methods.

The communications network and components structure of the KDCS is shown in figure 1. There are three levels of communication networks, i.e., the information network, the control network shaped ring topology with special Ethernet, and field network linked with RS-485 to the control network. The control communications network uses a special equipment called the ring accelerator which separates the failure node form the other control node to provide fault tolerance.

The testing and verification method is to evaluate about whole DCS system, as described above, including IO processing, control logic processing in CPU, binding process through network, to verify performance parameters(response time including binding delay, zitter, time precision, etc.).

The sample node components subjected to the 1st trial product test are as the following.

○ Number and Type of Stations
- 3 PCU (including Gateway)
- 2 FCU (allocating 1 PCU for each)
- IO module (1 DI, DO, AI, AO cards respectively)

○ Test Data capacity on the Control Network: 128 Kbyte
- Gateway: no data, manages only tokens
- 2 PCU: 64Kb/Node × 2 = 128Kbyte
- The total amount of traffic does not change because KDCS uses the Status-base communications, which transmits all data by schedule, and not event-driven.

○Station Binding Path(test examples)
DO(AO) card --> FCU1 --> PCU1 --> control network(Ring) --> PCU2 -->FCU2 --> DI(AI) card

The maximum delay value, obtained from overall binding path here would be the total sum of the maximum values of ① the DI(AI) scan cycle and handling delay, ② the communications delay between the upstream field network and the control network, ③ the delay time of the control communications network(maximum TRT-twice the time of the token rotation), ④ the communications delay between the control network and the downstream field network 2, and ⑤ the DO(AO) output cycle and handling delay etc.

5.2. Measuring the delay time and analyzing the trend

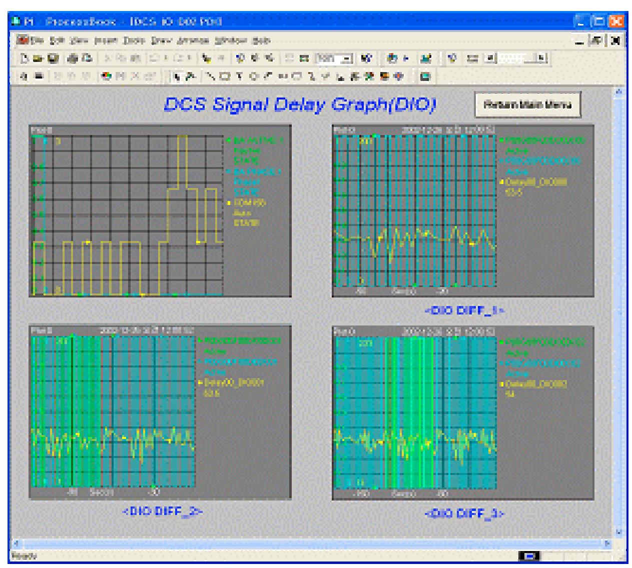

Fig 6. Trend of Delay between the Input & Output Points

The Fig-6 is a trend graph of the delay time for the digital input and output signals. Here, the DIO DIFF_1 displays the delay value between BDAS digital output channel number 0(=DCS digital input channel 0) and BDAS digital input channel number 0(= DCS digital output channel 0) after passing the DCS communications network and Station Binding. Among the trend line, the angled Delay00_DIO00 (marked yellow) displays the trend of the delay time. The other DIO DIFF_2,3 graph also shows the trend of its mutually connected input and output value, which undergoes station binding, along with its delay time.

From the above figure, it could be found that the binding delay time accommodates considerable changes, and because each channel is served simultaneously in KDCS, there is almost no difference in delay time among the channels. Currently, the binding delay time is marked as approximately 54ms. This 54ms changes depending to the amount of data of each station, the processing speed, and the delay of communications network(field network, control network included),

5.3 Average/maximum delay and Zitter

The following Excel figure is created as a report for the changes of signal values, and binding delay between certain control points. Which value is stored in the real time data base(RTDB). All types of statistic value can be induced from DB through Excel sheets. Although not displayed on the screen, the statistic value of this report is marked on the right side of this report.

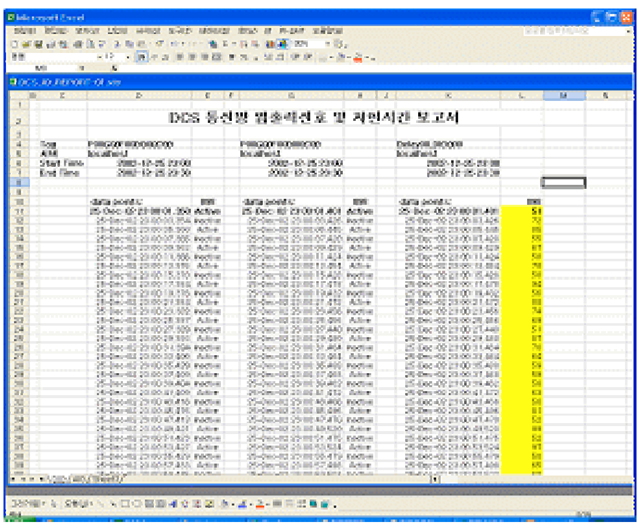

Fig. 7 A Statistic report about binding delay

The statistic values of the delay and zitter of the 1st trial products, analyzed through Excel above, are as the following.

○ Delay(Response) Time
- Max/Min/Avg : 101/28/64.8 ms

○ Delay Zitter
- Avg : 34.3 msec

As a result of statistical analyzing about multiple control points, all binding delay of all the most point in the KDCS shows same statistics value as the shown above. It is because that KDCS provides deterministic communication service to the all IO Point through the fairness scheduling. So this result is induced at all control points.

5.4 Evaluation and Verification of Binding Delay

The KDCS system for nuclear power plant possess a stronger distributional character than the ovation or any other foreign DCS. The distributed control is performed through FCUs scattered nearby field instruments on the site. Therefore, an action of control in unit time of ms and μs is possible in these distributed control node or the unit controller. However, in case of that signal binding is needed between the remote controllers inevitably, the binding delay must be minimized according to the dynamic characteristics of the power plant.

Generally the control signal's binding delay, for stable control on the power plant, considering the dynamic characteristics of the plant, should be guaranteed under 250msec(example of PID). It could be recognized that the binding delay value, with an average of 65ms and a maximum of 101ms, is sufficiently satisfying this required value. (In case of the Ovation, it is presumed to be about 200ms, considering its communications cycle.). On the other hand, pure communication delay on the control network(TRT – Token Rotation Time, in case of this ring style control network) can be detected by network analyzer. So the sum of control process delay, IO Processing delay, and field network delay can be induced from above overall binding delay. As an example, the TRT time of the 1st trial product of KDCS has a fixed value of 36msec, and thus, it could be acknowledged that an average of 29ms was consumed by the remaining field network, control process and the input/output handling. Thus, these results verify that the delay performance of the 1st trial products satisfies the required terms. But it should be noticed that this binding value is variable depending to the data capacity and the number of communications network nodes. So a separate test is necessary according to the each network composition in the actual site case by case.

6. Conclusion

This BDAS system makes the efficient measurement and testing of the delay performance such as the DCS binding time in the Linux/Windows based system and general PCs, which are not systems based on an expensive integration of special measuring devices or a high-priced special real time operation system. This system easily and naturally produces/gets the SOE test signals by the unit of 1msec, which is difficult to produce/measure from the general Windows system, through the batch scenario performed in the Linux sub-node. In additionthe respective delay value of the a lot of channels are simultaneously measured and their trend can be displayed, that is difficult with the oscilloscope or general measuring devices.

Especially, this BDAS provides the long-term statistic values as to average/maximum/minimum/variance value etc, from history data stored in realtime database, measured in the irregularly changing control circumstance. Thus, this strong point offers large benefit to evaluate, verify or validate the overall performance of DCS or an control system.

Along with these merits, the system's open system architecture gives compatibility, extensibility, easiness of maintenance, and low cost to the user. Therefore, the BDAS is expected to be efficiently applicable to measuring and testing not only for DCS, but also for other various control systems that need to analyze. the delay/response performance.

REFERENCES

[1] S. G. Kim; S. I. Song; E.S. Oh, K. Y. Gwak, S. M. Park, "Real Time Data Interconnection of the Distributed Control System for Nuclear Power ", *Proceedings of 3rd Measurement & Control Technique Workshop*, Joint Workshop of Nuclear Society-KIEE. Nov. 2003

[2] J. Y. Cho, K.Y. Gwak, J. K. Lee "A Design of IO Control System for Power Plant Simulations based on PCs", *Proceedings of Summer Conference* , KIEE, 2001

[3] S. H. Byun, T. I. Jang, J. Y. Cho, K. Y. Gwak, "An

Implementation of I/O Interface for Power Plant Simulator", *Proceedings of Summer Conference IEEK 99* , p.p 773-776, 99. 6
[4] T. I. Jang, K. Y. Gwak, S. H. Byun, J. Y. Cho, "Design and Implementation of a PC/Ethernet-based Communication Protocol for Control Systems" Proceedings *Proceedings of Summer Conference IEEK,* 1999. 6
[5] K.Y. Gwak, et. al.(3), "A Development of Fieldbus Network for Power Plant Simulator", *Proceedings of ICEE,* vol 2, p.p 421-424, 1998
[6] S. H. Hong, "A Study on the Implementation of Fieldbus Networks in the Distributed Control of Power Systems", *J. of KIEE,* Vol. 46, No. 4, pp.593-602, 1997
[7] S.H. Hong, K.A. Kim, J.Y. Kim, S.J. Ko, "Distributed Control, Automation System and Fieldbus", ICASE Magazine, Vol.2, No.4, pp.19-29, 1996
[8] S. Dhawan, *Networking Device Driver*, Van Nostrand Reinhold, ITP Inc. New York, USA, 1995, Chap5, pp.197-231