DCS Emulator Development for the ACR-1000® Simulator

Y. Nakashima[1], R. Trueman[2] and K. Ishii[3]
[1] Hitachi Canada Ltd., Power & Industry Division, Ontario, Canada
[2] Atomic Energy of Canada Ltd., Ontario, Canada
[3] Hitachi Ltd., Information & Control Systems Company, Ibaraki, Japan

Abstract

Nuclear Power Plant (NPP) simulators are the main means for operator training and as such are a crucial part of the NPP operation life-cycle. Hitachi, Ltd., Information & Control Systems Company (henceforth "Hitachi") is the preferred DCS and DCS emulator developer and supplier for the ACR-1000® NPP control system. Hitachi's concept for the DCS (distributed control system) portion of the ACR-1000® simulator is "total emulation of the DCS" by software. This paper will review the current status of the technical development and the major project milestones.

1. Introduction

As the ACR-1000® preferred DCS supplier, Hitachi has worked closely with AECL to apply the G-HIACS νSAFE (pronounced "nu-safe"), the latest version of Hitachi's successful "HIACS" series of DCS, to the ACR-1000® computer controls. In parallel, Hitachi has been developing the νSAFE DCS emulator with the fundamental goal of "total emulation by software". Hitachi's extensive experience with DCS emulators and BWR/ABWR simulators will support not only its internal development strategy but also AECL's state-of-the-art ACR-1000® full-scope simulator (FSS) development. As a follow-up to Hitachi's 2009 CNS DCS emulator paper [1], this paper will focus on the steps for development integration of the νSAFE DCS emulator component of the overall FSS architecture (DCS emulator, plant model and PDS).

2. Hitachi's DCS Emulator vs. Simulator Experience

Hitachi's first BWR NPP simulator was designed and built for the Shimane-2 (NS-2) BWR (commercial operation February 1989). Hitachi was directly involved in the establishment of the "BWR Training Center" (BTC) located near the Kashiwazaki-Kariwa NPP site. Hitachi's latest NPP simulator contract is for the turn-key supply of an FSS for the Shimane-3 (NS-3) ABWR currently under construction. Hitachi is also the NS-3 turnkey EPC contractor. NS-3 is scheduled for fuel load to begin in December 2010 and commercial operation in December 2011; however, the simulator has been in operation since August 2009. The NS-3 simulator will be the 6th NPP simulator for which Hitachi has full or partial scope, and it will be its first ABWR simulator. The NS-3 simulator is noteworthy also because it will be the first to integrate DCS emulator technology. Note that the BTC simulator (BTC-3) for the Kashiwazaki-4 BWR was recently retrofitted with DCS emulator technology.

Advance delivery of the plant FSS is essential for the timely validation and implementation of the operator training programs; however, from an engineering and commercial point of view, it creates some additional challenges. In the NS-3 case, the initial NPP application programming was completed in September 2008; this version was the first used in the NS-3 FSS. The FSS team started its internal testing program using this initial version. The NS-3 NPP application program testing was completed in December 2008. This "released from factory" version of the application program was downloaded to the FSS DCS emulator; FSS testing continued with this updated version. Following final on-site commissioning, testing and start-up, the final "as built" application program was downloaded to the FSS DCS emulator, thus synchronizing the FSS to the operational NPP application. Hitachi's experience to date indicates that a similar evolution can be expected in all NPP new build projects including any ACR-1000® new builds.

For information and comparison purposes, Figure 1 shows the NS-3 FSS major project milestones vs. overall NPP project milestones.

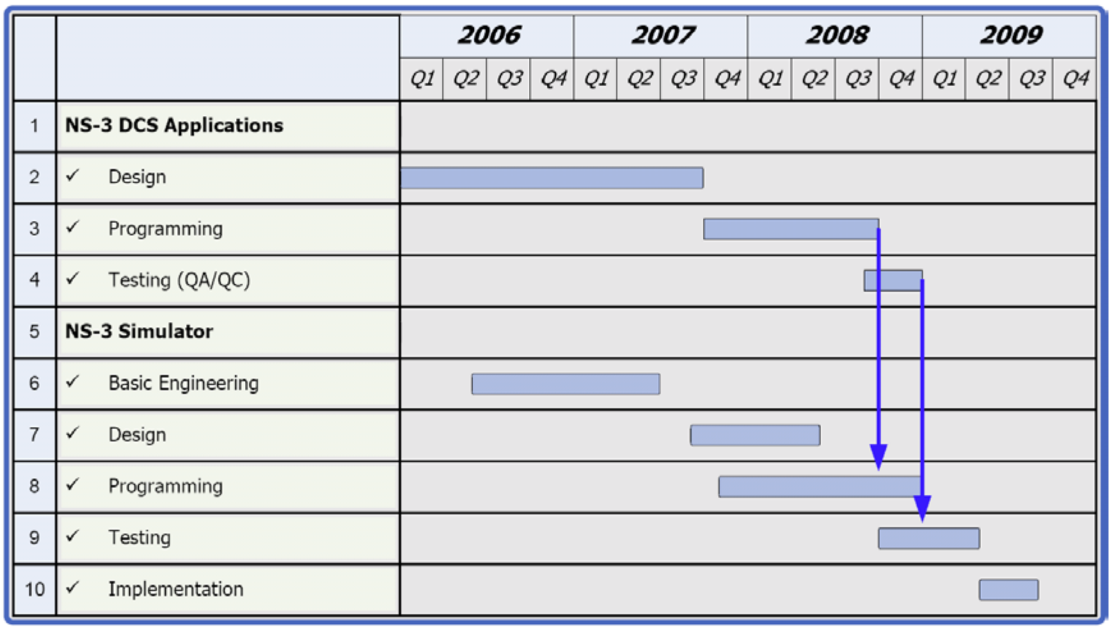

Figure 1 NS-3 NPP and FSS Project Tasks, High Level (2006-2009)

3. Project Milestones – DCS Emulator vs. Overall ACR-1000® Simulator

The primary objective of this effort is to provide an FSS for ACR-1000® NPP operator training and to have this available well in advance of the first fuel load. The FSS is divided into 3 main modules: (1) the plant model, (2) the DCS emulator, and (3) the application software programs (PDS, DCS). A major portion of the simulator is a complete emulation of the vSAFE DCS. The development of module (2) is key to achieving the secondary objective of this FSS development effort: a training and simulator environment that eliminates the need for (while maintaining the capability to use) DCS "hardware-in-the-loop". The following sub-sections describe the main activities required for the development of the vSAFE DCS emulator and integration within the overall ACR-1000® FSS.

FSS Software

1. Reactor/Plant Model
    - Design Data (modelling of plant systems)
    - Implementation Data – (reproducing system design parameters, heat balance, and instrumentation / control systems)
    - Performance and Transient Data (modelling of proper responses to reactor events)
    - Reactor Core Data (modelling of the reactor core)
2. DCS Emulator
    - DCS Emulator – Testing and checkout of prototype DCS emulator.
3. Plant Control Application Software
    - Baseline DCS and Plant Display System (PDS) software applications

Equipment Deliveries

- DCS equipment and DCS Emulator (for both simulation modes, i.e., (1) with DCS hardware in-the-loop and (2) with full DCS emulation)
- PDS Equipment, Hard Panel and I/O (replicas of the MCR control panels and operator work stations)
- Simulator Computer Equipment (modelling computers, instructor computers and connectivity equipment)

3.1 DCS Emulator Development

Currently, we expect that vSAFE DCS emulator development will progress in 3 major “steps”, as described in the following sub-sections.

3.1.1 DCS Prototype Emulator Application

This step has already been implemented. The prototype vSAFE DCS emulator has been developed to work in conjunction with a simple plant model (implemented as an emulated controller) and a simplified HMI (virtual PDS). AECL has developed a prototype emulator sample application program to test the overall system concepts (multiple POU, task assignments and controller to controller communication) as well as the capabilities of the DCS emulator. This prototype DCS emulator is described in more detail in section 5.

3.1.2 DCS Emulator with External Plant Model and PDS

The preliminary prototype DCS emulator will be expanded to allow for connection to a PC-based plant model; the plant model and DCS emulator will reside either on the same PC or 2 separate PCs. A connection path to the actual PDS hardware and software system will be established. This will demonstrate the required functionality for use in the mock-up FSS and final FSS systems. Figure 2 provides an overview of this architecture.

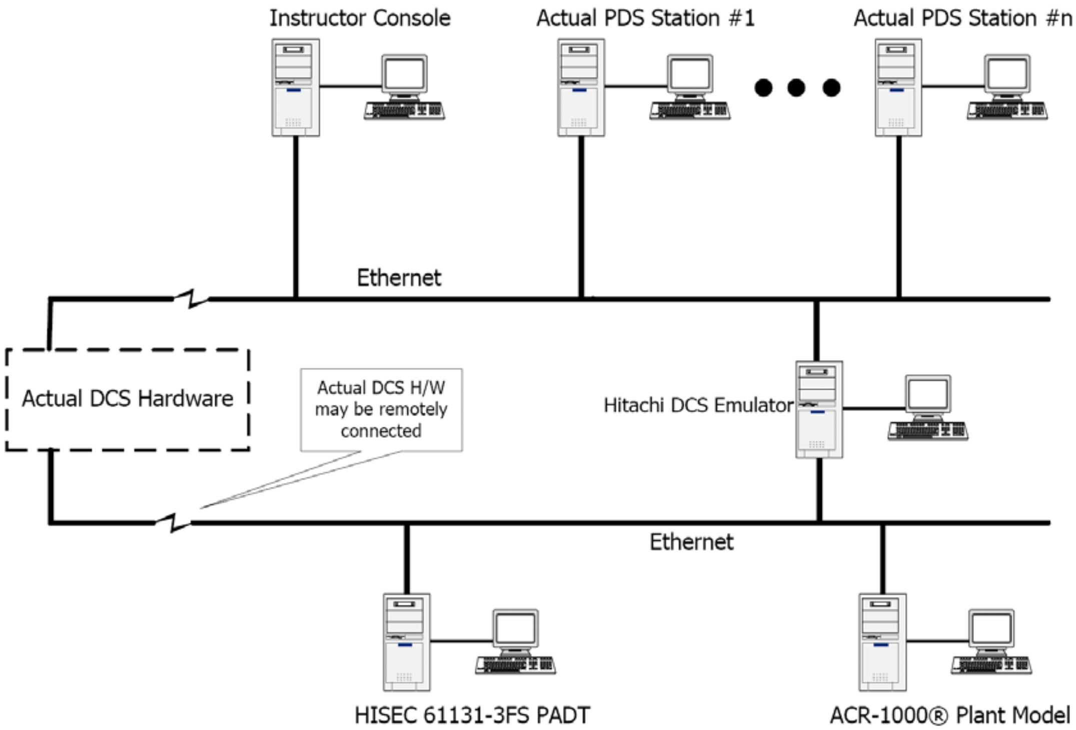

Figure 2 DCS Simulator Architecture

The DCS emulator has been developed to allow different options for connection with external plant models. The API (Application Program Interface) supporting this function enables data exchange between the 2 applications using a well-defined/specified protocol. One option being developed by AECL is the ACRSim product, which is based on the Matlab/SimuLink environment. Developed to Tier 2 coding standards, this model can be used for control analysis in addition to mock-up FSS purposes. ACRSim includes thermal-hydraulic, reactor physics and control models and also simulates the dynamics of the ACR reactor, coolant systems, and instrumentation and control, under a wide variety of conditions and scenarios.

This version will fully support the following key features and functionality:

1. Allow the actual DCS application code and ACR-1000® multi-controller configuration to run in the plant training simulators without the need for any DCS hardware (i.e., control applications can be virtually downloaded to the emulated controllers).
2. Support TCP/IP based communications channels and requisite interfacing equipment thus allowing the Plant Model to communicate with DCS application logic running either in actual DCS hardware (via an I/O emulator) or in the DCS emulator. See section 4.4.
3. Switch between a DCS emulation environment and a stimulated environment (using actual DCS hardware and an I/O emulator) on a partition-by-partition basis.
4. Execute the same application software running in the real NPP (with requisite simulation "hooks") and be able to load and execute new revisions of the NPP application software with minimum modifications.
5. Respond to DCS malfunctions in a manner nearly identical to that of the real DCS.

In addition, the following key actions by the simulator will be supported by the emulator:

1. Freeze / Run
2. Store / Restore
3. Backtrack / Restore Backtrack
4. Snapshot
5. Replay / Fly-out
6. Speed Up / Slow Down
7. Synchronization

3.1.3 ACR-1000® Simulators

The primary purpose of the ACR-1000® simulators is to provide (to the utility/owner) a dedicated facility for operator training and re-qualification for safe and efficient NPP operation. The vSAFE DCS Emulator described in section 3.1.2 will be designed to support the three simulator architectures defined below:

Full Scope Simulators

FSSs will look and respond like real ACR Main Control Rooms (MCRs) (see Figure 3).

Figure 3 Full Scope Simulator Main Control Room

The FSS main components are the plant model, DCS emulator, Instructor station(s) and a PDS/SSMC. FSSs will be controlled by Host Computer Systems that model a single unit plant. FSS activities will be monitored and supervised from instructor stations, i.e., separate computer systems residing in the simulator MCR. See Figure 4 for an FSS architecture example.

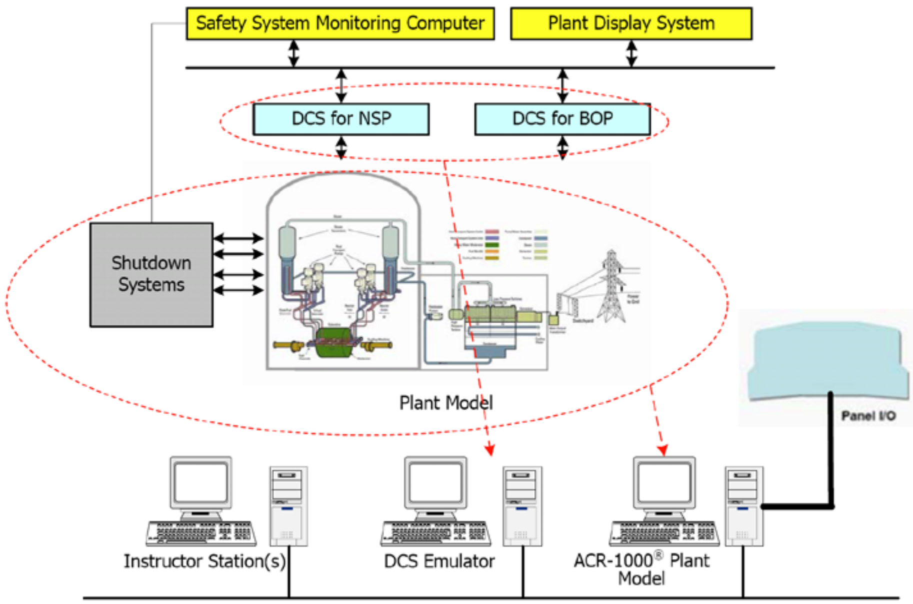

Figure 4 Full Scope Simulator Main Components, Example

Classroom Simulators

Classroom Simulators (CRSs) are Training Simulators that only contain a limited number of soft panel displays residing in training classrooms. These facilities will be used to train station staff and can provide overflow support for certain training scenarios should the FSS be fully booked.

Desktop Simulators ("Everything on a Single PC")

Desktop Simulators (DTSs) will operate on (single) local PCs. Each DTS will have the computing and memory capacity to execute the Host Computer System functionality, the Instructor Station functionality, and the DCS emulator functionality. Each dedicated DTS will be able to support up to four displays. Each DTS will be able to be operated with a single display. See Figure 5.

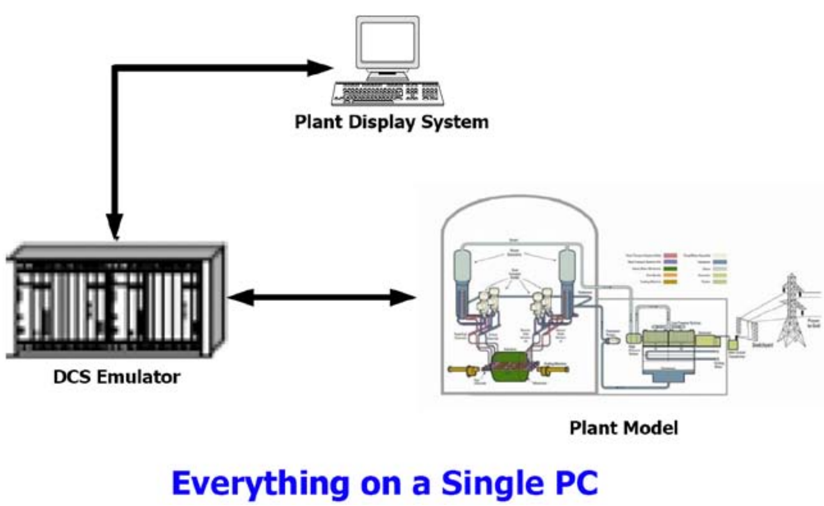

Figure 5 Desktop Simulator Concept

An integral part of all three ACR-1000® Simulators described above will be total emulation of the DCS control software.

4. Concept of G-HIACS νSAFE DCS Emulator

Hitachi is currently in the final stages of certification and major development of the G-HIACS νSAFE, an IEC61508 SIL2 Functional Safety DCS platform with an IEC61131-3-compliant programming and diagnostic tool interface (HISEC61131-3FS PADT). The PADT package includes a logic and DCS emulator. This state-of-the-art DCS emulator will be incorporated into the ACR-1000® FSS. This section provides an updated overview of this latest Hitachi DCS technology product.

4.1 Target machine

For easy simulator implementation, the νSAFE DCS emulator allows the actual DCS application software to be executed on a single PC without the need for any DCS hardware. Hitachi recommends the following minimum PC specifications to support emulator use:

- CPU : Intel® Core™2 Duo 2.4GHz
- Memory : 2 GByte
- OS : Microsoft® Windows XP SP3

The target PC OS choice [1] has been changed from Linux to Windows™. While Hitachi considers a Linux machine to be better from an engineering and development (i.e., internal) point of view, a Windows™-based machine offers important marketability and usability advantages:

- performance simplification
- simplification of architecture (“everything on one PC” strategy now possible)
- widespread use, acceptance and availability and support of Windows™ based machines

4.2 Specifications

Given the minimum target PC specified in section 4.1, the νSAFE DCS emulator will be able to support the emulation of up to 64 controllers. While one expects that improvements in PC and OS technology would allow us to increase this number, we do not currently anticipate the need to count on such improvements to completely emulate the ACR-1000®. The decision to support a maximum of 64 is technically reasonable on the basis of the following ACR-1000® DCS controller estimated requirements:

- NSP PCSS Device X/Y, Group : 11 (22 duplex controllers)
- NSP PCSS Gateway CTL : 3 (6 duplex controllers)
- NSP ECSS Single CTL : 6 (single controllers)
- NSP ECSS Gateway CTL : 12 (single controllers)
- BOP DCS : Estimated 24 (on the basis of latest NS-3 BOP)

The estimated ACR-1000® emulation requirement is a total of 56 controllers. Since this is within the capabilities of the Windows™-based option, the marketing and usability advantages of using a "regular" PC were considered decisive.

4.3 Software architecture

Despite the change in OS, the fundamental software architecture has not changed [1]; however, the emulator task is now executed by using an API rather than the internal PC task scheduler. The emulator task functionality has remained unchanged, i.e., calling each emulated controller one-by-one and emulating one scan cycle of their respective application programs. The simulator calls the API in order to keep synchronization between the emulator and the plant model.

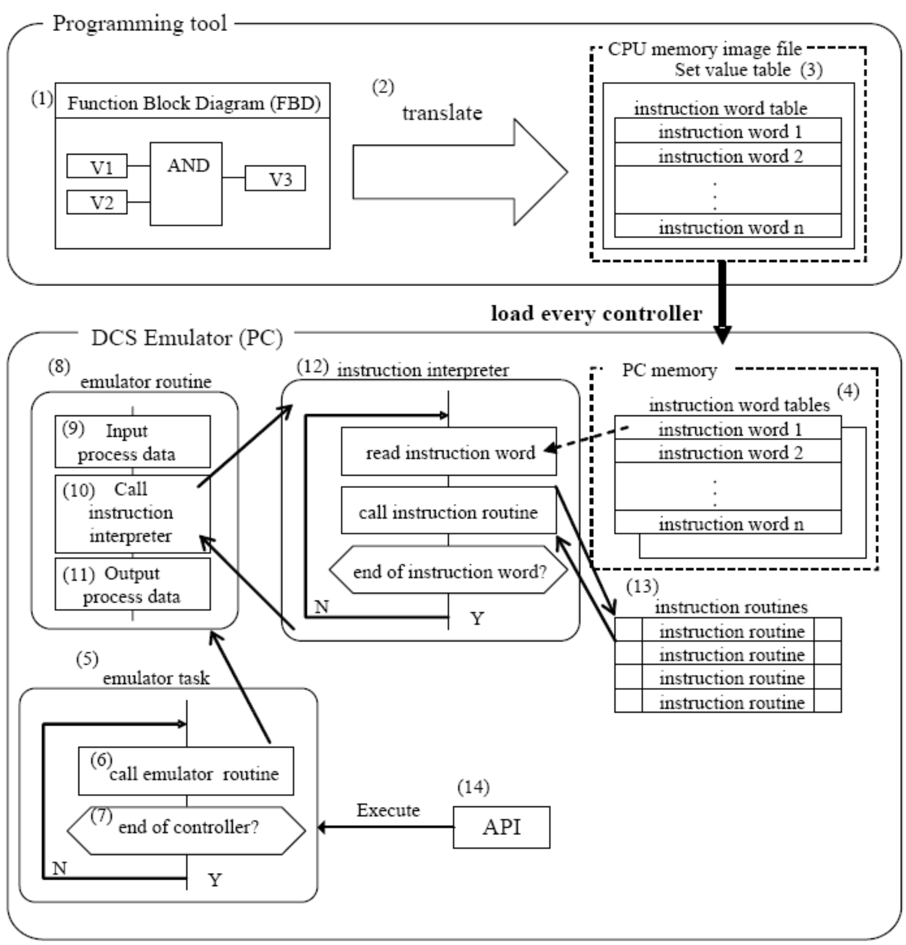

Figure 6 DCS Emulator Software Architecture

Figure 6 above shows the software architecture of the vSAFE series DCS Emulator. The compiled application control software can run on a PC as if it were on a real DCS. Application programming by the user is currently limited to using Function Block Diagram (FBD); ST (and SFC) will eventually be included as well. Programming with FBD (1) takes place in the PADT environment; the PADT (i.e., "programming tool") translates (2) the user-created FBD (1) into the Instruction Word Table (3), which is the same as that used in the real DCS. The PADT then loads the Instruction Word Table for every controller into Function Block (FB) instruction tables (4) in the DCS emulator. In the DCS emulator, the emulator task (5) is activated by the API (14) and calls the emulator routine (8). The emulator routine then takes input data from the plant model (9), calls the FB instruction interpreter (10), and then sets the outputs to the plant model (11). When called, the FB instruction interpreter (12) takes the FB instruction words out of the FB instruction tables (4), and then calls the corresponding FB instruction routine (13) which is the same as that used in the real DCS. For a multi-controller emulation, the emulator task (5) executes each controller's software in sequence (7).

4.4 Communications

Communication modes described in #1-3 are unchanged [1]. Due to the OS specification change for the host PC, communication mode #4 is being revised.

1. <u>Inter-Controller Communication</u>

The νSAFE DCS emulator can support the emulation of multiple controllers (up to 64). The data exchanges between emulated controllers are implemented by writing to or reading from a shared common memory resident on the emulating PC.

2. <u>Communication with Physical Controller(s)</u>

Since it may be necessary to exchange data between the emulated controller(s) and actual physical controllers, the νSAFE DCS emulator supports a communications protocol that is identical to that used in the actual plant. Refer to section 3.1.2.

3. <u>Communication with Programming and Diagnostic Tool (PADT)</u>

The νSAFE DCS emulator allows users to use the PADT "as is" to control directly emulator functions such as run, freeze, data forcing and monitoring.

4. <u>Communication with Applications Developed by Users</u>

APIs will be supplied for communicating with other applications (e.g., see the functions described in section 4.5). Open OPC is a useful capability for developing applications for verification and validation strategies; therefore, Open OPC, which provides a universally accepted standard interface, will be used for communication with external systems. Specifically, Open OPC functions will be used to call the various APIs.

4.5 Functions

The DCS emulator now provides the three basic API functions described below. These basic APIs will allow the simulator to access the key actions described in section 3.1.2.

1. <u>Execute</u>

This function executes one emulator scan cycle. The simulator's "Run" command is executed by repeated, sequential calling of "Execute". The simulator "Freeze" command stops this repeated execution of "Run". The sequential execution of the emulator and the plant model causes "Synchronization". "Speed Up / Speed Down" is activated by changing the cycle time of the simulator calling of the "Execute" command.

2. Input / Output (to/from Emulator)

The emulator has virtual I/O data which are allocated space in the PC memory. The emulated controllers read from / write to the virtual I/O. This memory area is common to the simulator program and the emulator; therefore, the simulator also can read from/ write to the same virtual I/O. In this case, "I/O" includes commands to and from the Emulator (e.g., from/to the PDS).

All these emulator I/O data (signals, commands, operator actions, and so on) can be recorded and can be "Replayed". If a user wants to retry a recorded scenario from a specific point, the user can stop the "Replay" and "Fly-out" to the point at which he wants to re-enter and retry new commands.

3. Store / Restore

Executing the "Store" command causes the emulator to store its internal state so that it can be recalled / restored when the "Restore" command is executed. The user can use this function to implement the simulator "Store / Restore", "Backtrack / Restore Backtrack" and "Snapshot" commands.

5. Prototype G-HIACS vSAFE DCS Emulator

For feasibility assessments, Hitachi has developed the Prototype G-HIACS vSAFE DCS Emulator. The example configuration and logic will consist of a typical ACR DCS partition using three (3) G-HIACS vSAFE R800FS controllers with the following functionality:

1. Group (GRP) Controller
    - Obtain current values from the DevX and DevY controllers
    - Determine controller health, set flags accordingly
    - Execute Moderator Temperature Control (MTC) PID control
    - Generate small and large valve lifts
    - Send lift values to DevX and DevY

2. DevX (A/B) Controller
    - Obtain current values from the Group controller
    - Execute device control (small and large valve outputs to the field)
    - Handle manual valve requests from a "Virtual" PDS.
    - Send values back to the Group controller

3. DevY Controller
    - See DevX (A/B)

A fourth controller will also be included in the prototype DCS Emulator. This controller will be programmed with a simplified model of the ACR-1000® moderator in order to provide realistic responses to the control logic running in the Group, DevX and DevY controllers. The Model controller will be linked to the Emulated DCS controllers via "virtual" I/O connections. This controller will have the following functionality:

4. <u>MDL Controller</u>

- Read small and large valve lift signals (AOs) from DevX and DevY
- Convert valve positions to flow
- Determine total flow into Heat Exchanger
- Calculate temperature drop across Heat Exchanger
- Implement temperature lags and add bias
- Make temperatures available as input signals (AIs) to DevX and DevY

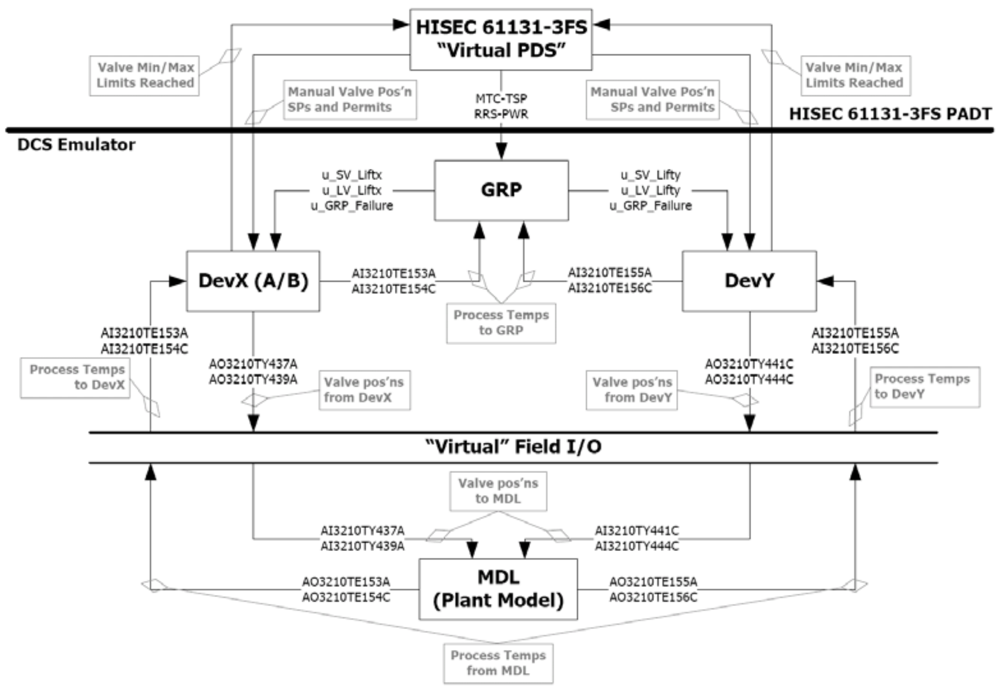

Figure 7 Emulator Sample Code Data Flow Diagram

Figure 7 provides the high level data flow between the Emulated controllers as well as between the "Virtual PDS" and the plant model. Transmission of data to/from controllers will follow safety data transmission protocols defined by AECL and Hitachi. The PADT will be able to access and manipulate the "virtual" PDS signals in order to simulate PDS interactions with the controller logic.

Figure 8 shows the dynamic response of valve lift and process flow temperature for a part of the current vSAFE DCS emulator.

- PA_SPIN = temperature set point
- uSV_Liftx = small control valve lift signal

- uLV_Liftx = large control valve lift signal
- AE3210TExxxx = process temperatures

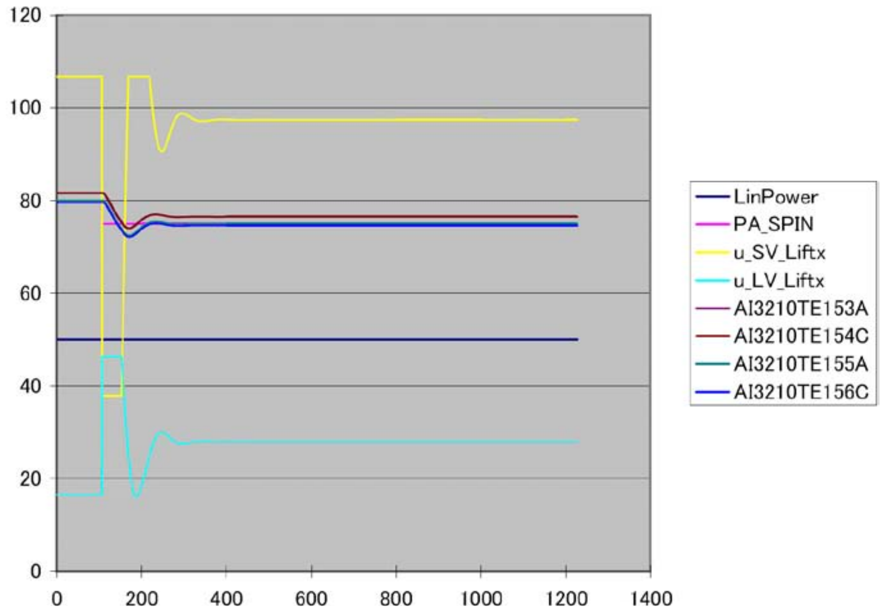

Figure 8 Valve Dynamic Response, vSAFE Emulator

6. Conclusions

As of March 2010, Hitachi has reached the first step in its vSAFE DCS emulator development; this first version is currently interacting with AECL's simple plant model and simplified PDS. This achievement has enabled AECL to begin its preliminary software application tests as well as its "proof-of-concept" for the architecture of both the ACR-1000® control system and the FSS. Preliminary verification and validation is also underway. The next emulator development stage ("DCS Emulator with External Plant Model and PDS") will have the capability to support the entire DCS architecture and have the capability of communication with an external plant model and improved PDS. AECL Mock-up FSS development, software testing and expanded verification and validation will be supported. The final stage of DCS emulator will be integrated into and support the development of the ACR-1000® FSS, enabling total system testing as well as operator training. In all stages, the goal of "total emulation of the DCS by software" will be maintained and achieved.

7. References

[1] Y. Nakashima, K. Ishii and D. Chiba, "DCS Emulator Development for Nuclear Power Plants", 30th Annual Conference of the Canadian Nuclear Society, Calgary, Alberta, Canada, June 3, 2009.

[2] Y. Nakashima, S. Masunaga and Y. Maruyama, "Hitachi Power Plant DCS Simulator Development", 15th Annual Conference of the Institute of Electrical Engineers of Japan, Hitachi-shi, Ibaraki-ken, Japan, December 1, 2007.