INTERNATIONAL ATOMIC ENERGY AGENCY



# Reactor Simulator Development

## Workshop Material



TRAINING COURSE SERIES 12

VIENNA 2001

PLEASE BE AWARE THAT
ALL OF THE MISSING PAGES IN THIS DOCUMENT
WERE ORIGINALLY BLANK

TRAINING COURSE SERIES No. 12

# Reactor simulator development

*Workshop material*

INTERNATIONAL ATOMIC ENERGY AGENCY, 2001

The originating Sections of this publication in the IAEA were:

Nuclear Power Technology Section
and
Nuclear Power Economic Planning Section
International Atomic Energy Agency
Wagramer Strasse 5
P.O. Box 100
A-1400 Vienna, Austria

REACTOR SIMULATOR DEVELOPMENT
IAEA, VIENNA, 2001
IAEA-TCS-12



Printed by the IAEA in Austria
April 2001

## FOREWORD

The International Atomic Energy Agency (IAEA) has established a programme in nuclear reactor simulation computer programs to assist its Member States in education and training. The objective is to provide, for a variety of advanced reactor types, insight and practice in reactor operational characteristics and their response to perturbations and accident situations. To achieve this, the IAEA arranges for the supply or development of simulation programs and training material, sponsors training courses and workshops, and distributes documentation and computer programs.

This publication consists of course material for workshops on development of such reactor simulators. Participants in the workshops are provided with instruction and practice in the development of reactor simulation computer codes using a model development system that assembles integrated codes from a selection of pre-programmed and tested sub-components. This provides insight and understanding into the construction and assumptions of the codes that model the design and operational characteristics of various power reactor systems.

The main objective is to demonstrate simple nuclear reactor dynamics with hands-on simulation experience. Using one of the modular development systems, CASSIM™, a simple point kinetic reactor model is developed, followed by a model that simulates the Xenon/Iodine concentration on changes in reactor power. Lastly, an absorber and adjuster control rod, and a liquid zone model are developed to control reactivity.

The built model is used to demonstrate reactor behavior in sub-critical, critical and supercritical states, and to observe the impact of malfunctions of various reactivity control mechanisms on reactor dynamics. Using a PHWR simulator, participants practice typical procedures for a reactor startup and approach to criticality.

This workshop material consists of an introduction to systems used for developing reactor simulators, an overview of the dynamic simulation process, including numerical methods, and an introduction to the CASSIM™ development system. Techniques and practice are presented, with exercises, for development and application of reactor simulators.

The workshop material was prepared by W.K. Lam, Canada, for the IAEA. The IAEA officer responsible for this publication was R. Lyon from the Division of Nuclear Power.

## *EDITORIAL NOTE*

*The use of particular designations of countries or territories does not imply any judgement by the publisher, the IAEA, as to the legal status of such countries or territories, of their authorities and institutions or of the delimitation of their boundaries.*

*The mention of names of specific companies or products (whether or not indicated as registered) does not imply any intention to infringe proprietary rights, nor should it be construed as an endorsement or recommendation on the part of the IAEA.*

## CONTENTS

# 1.0 Introduction to Modular Modeling Systems

In 1968, General Electric equipped its training center with the first full-scope training simulator. That was followed by all the major manufacturers of light water reactors in the U.S., and subsequently, by several utilities and training institutes, both in the U.S. and overseas. Nowadays, power plant simulators have become necessities among nuclear utilities worldwide and have become much more than a training tool. In addition to imparting to the operator an extensive "hands-on" experience in startups, power maneuvering and shutdown operations, they give them the opportunity to face more emergency and abnormal situations. They also constitute a valuable tool during licensing examination and in the development of operating procedures. As well, they have been used in plant design validation as engineering simulators.

Over the span of the last 30 years, the simulator technologies have changed significantly due to the rapid advances in computer hardware and software. 15 years ago, computers that were used to simulate a complete nuclear power plant down to the last electrical fuse occupied a full room and cost millions of dollars. Nowadays, the same full scope simulation can be accommodated by a high-end multi-Pentium chips PC sitting on your desktop, and it only costs few thousands of dollars.

Despite these advances and dramatic price/performance improvement in computer hardware and software, the essential development cost of a power plant simulator remains to be the knowledge intensive mental labor that is required in mathematical modeling — representing the physical systems by algebraic and differential equations, and subsequently transforming these equations into computer codes, followed by endless process of model tuning and enhancements.

In the past, mathematical modeling was a very labor intensive process because any minor change in the equations, or some adjustment in numeric constants, necessitated repeated program re-compiling and debugging. Therefore, the development of complex models simulating large scale systems was a very difficult and time consuming task. Then in the mid 80's, the use of a modular approach for the model development of large scale systems began to appear and evolved rapidly. Nowadays, almost every simulator development will involve some kind of a Modular Modeling System (MDS). However, the ease of use and the capabilities of the individual MDS often vary with different vendors.

## 1.1 Modular Model Generation Process

In general, the modular model generation processes include:

1. Based on inspection of the whole plant, the **basic units of component** like a valve, pump, heat exchanger etc. will be identified. Individual mathematical model for each of these components will be developed to

cover various operating conditions. It is important that the model will be generic and flexible enough to facilitate easy customization of the component for specific application. For example, to change a valve model from a first order linear valve to a quick-opening non-linear valve only requires changing certain parameters in the model. Some vendors called these basic units "generic algorithms"; some call them "standard model handlers"; others call them "model objects" (a more modern terminology). In essence, they all amount to the same thing — that is, they are all compiled into object files and stored in a library that is readily called upon by the main simulation program.

2. The next step is **modularization** of the whole plant into subsystem modules — reactor, heat transport system, boiler, feedwater system etc. Each subsystem model is further broken down to process model, electrical power distribution model and control logic model. This is a critical process in which modularization guidelines will be established in sub-dividing the system into manageable units. When these criteria are combined in the definition of the borders of the system's modules, the result will be a set of well designed easily manageable modules with a low level of coupling with them.

3. The third step is to develop **specialized algorithms** specific to the requirements of the particular modules — for example, specialized two phase flow modules for reactor thermalhydraulics to cater for loss of coolant accident (LOCA) simulation, or special 3D spatial kinetic reactor model etc.

4. The fourth step is to **build subsystem models** using basic blocks of generic algorithms, as well as the specialized algorithms. Generally speaking, the simulation model is described as a sequence of interconnecting blocks where each block has defined inputs and outputs, whose relationships are characterized by a set of differential-algebraic equations — mathematical representation of a basic component. Specific component block can be created by specifying parameters to the generic algorithm. The output(s) of one block feeds into one or more subsequent block(s) downstream. When the simulation runs, each block is executed in sequence, producing output(s) which is passed onto the next connected block(s). This next block in turn executes the simulation based on its inputs and coefficients (parameters), and produce outputs for subsequent blocks connected to its outputs. When all the blocks in the model have been executed once in the sequence, the simulation has completed one iteration.

5. The fifth step is **testing** to ensure that the subsystem model satisfies the modeling requirements specification. Such requirements will include: range of operating conditions; model fidelity ( % deviation from operating data); model performance on selected malfunctions etc.

6. As the several interrelated subsystem models are completed, they will be **integrated** to form a large size system model, followed by testing and model tuning etc. leading to the final plant model.

The process for building generic algorithm library and the large scale model is illustrated in Figure 1.

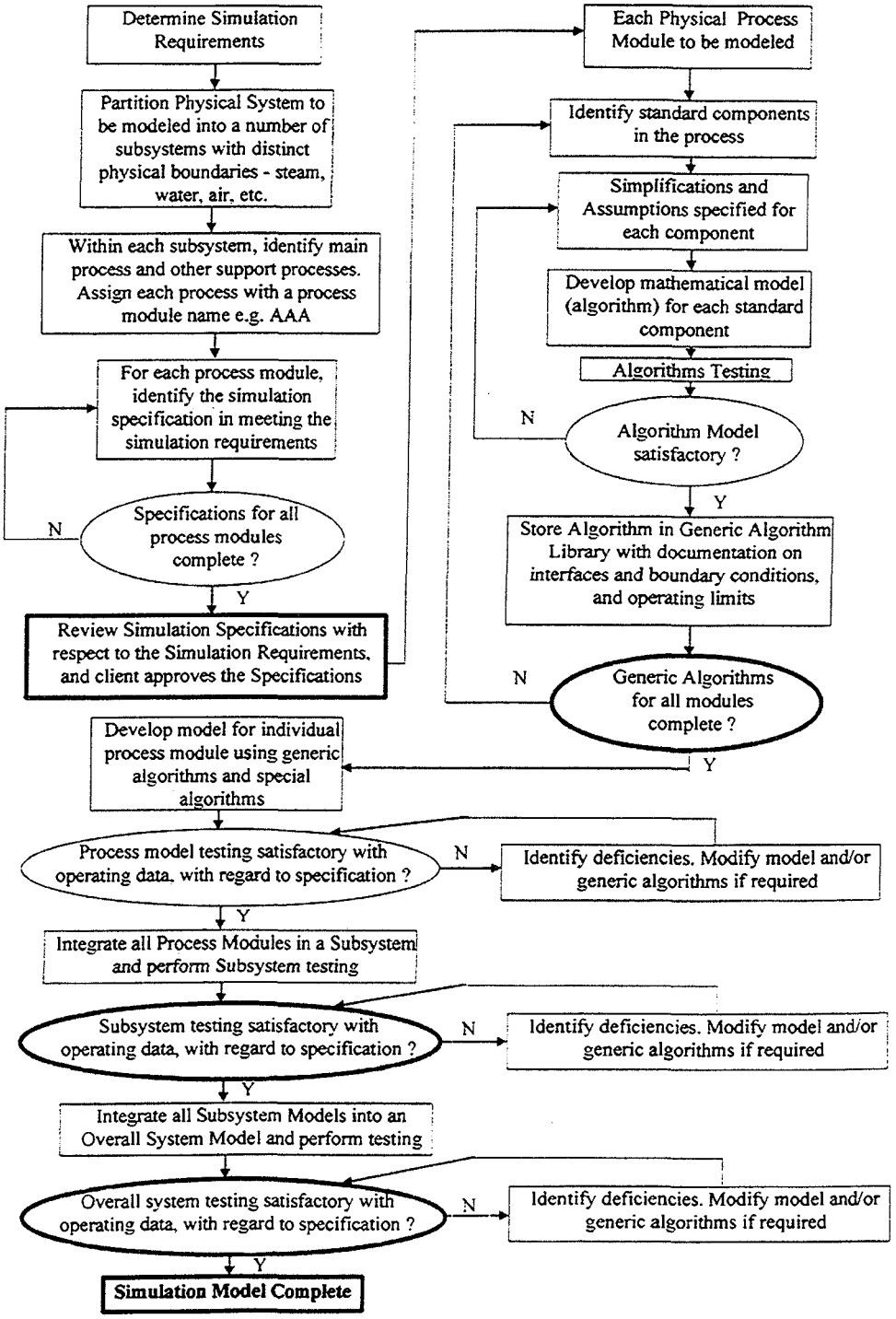



*Figure 1.*

## 1.2 Integral Components of a Modular Modeling System

Basing on the above described process of modular model development, a typical MDS typically consists of the following integral components:

(a) **Generic Model Algorithm Library Management** — create generic algorithms and store them into libraries.

(b) **Model Development System** — to create blocks, arrange block sequence, make block connections, extract or merge portion of the model with another model.

(c) **Model Testing and Debugging Environment** — to change block connections; change model parameters; change block sequences.

(d) **Thermal Hydraulic Flow Network Solver** — to create a network topology consisting nodes and branches representation of a hydraulic flow circuit in the plant, where pressures are calculated in control volumes known as nodes; and the flow between these nodes can be obtained from the application of momentum equations in the branches between adjacent nodes. By specifying nodes and branches, a system of linearized algebraic equations will be formulated and solved by fast matrix methods.

In the sections below, we will briefly describe the Simulation Development System from various vendors, ranging from Full Scope Simulator Development System to PC based Desktop Simulator Development System. As there are many simulation vendors worldwide, the ones selected below are only considered as representative samples.

## 1.3 Simulation Development System for Control Room Replica Full Scope Simulators

(a) ***ROSE (Real time Object oriented Simulation Environment ) from Canadian Aviation Electronics (CAE), Montreal, Canada.***

ROSE from CAE is an object oriented, icon-based design and simulation development system and has been used for full scope training simulator applications for nuclear power plant and fossil fuel power plant. The library objects, the lowest level building blocks in the simulation, are developed using the **Object Editor**. A ROSE Object consists of a graphical icon and corresponding object definition information. In the ROSE environment, models are represented graphically by **Schematics**. Schematics are drawn by dragging objects from the Library Window onto the Schematic Window, then by connecting objects to one another. Simulation configurations may be loaded and run through the runtime executive. A suite of simulation and monitoring tools allows users to open schematics and display, plots or update any simulation variable. The **Model Builder** groups the schematics and passes them to the appropriate code generators. The **Code Generators** are a set of tools used to parse the

schematics and generate simulation model codes in C and FORTRAN. Specialized Code Generators include: (a) Hydraulic and Electrical Network Solvers (2) Relay Logic Model Generator (3) Sequential Logic Generator

(b) ***CETRAN from Asea Brown Boveri (ABB), Connecticut, USA***

ABB's CETRAN has been used for real time full scope simulators application for nuclear and fossil power plants. CETRAN software is block structured. In block structured approach, the model development engineer creates blocks from an algorithm library, enters physical and thermodynamic coefficients to make the model match the physical plant, and then links the blocks together interactively and on-line. A CETRAN Model is created by interconnecting pre-programmed blocks (e.g. heat exchanger, time delay etc.). The **CETRAN Executive** controls all of the simulation software modules and performs the following simulation controls : (a) creates the necessary initialization before simulation runs (b) controls the timing based on the real time clock (c) Passes necessary information to other modules (d) activates/deactivates the other modules. The CETRAN Graphics Package generates a variety of displays for the instructor and operator stations. This software allows the user to build tabular, graphic, or trend displays, and to make the displays interact with any simulated variables of the model. CETRAN Network Solvers are provided for Thermal hydraulic (THLF) Network Models and Electrical (ELNET) Network Models.

### 1.4 Simulation Development System for PC Based Desktop Simulators

(a) ***The Modular Modeling System (MMS) from Framatome Technologies, Lynchburg, Virginia, USA***

Modular Modeling System (MMS) is Simulation Model Development tool targeted for Desktop Simulators applications. It includes a **Component Modeling Library** which contains generic pant components. The Model source code generation is aided by the use of a **Graphical Model Builder** — models are built by placing graphical icons on the worksheet representing a module from the Component Library. Once the components are placed, connections are made between ports on the modules to define the flow of fluids, control signals etc. Each module has an interactive input form that allows the entry of component data (i.e. pipe length and diameter). MMS makes use of the general purpose simulation language — Advanced Continuous Simulation Language (ACSL) from Mitchell and Gauthier Associates (MGA). The model source code generated by the Graphical Model Builder is an ASCL source code macro, which requires ASCL macro translator and FORTRAN compiler to convert them to object codes. Therefore, the ASCL software and the FORTRAN Compiler are prerequisites for running MMS.

(b) ***The APROS — APMS (Advanced PROcess Simulator — Advanced Paper and Pulp Mill Simulator) from Technical Research Institute of Finland***

The APROS — APMS concept can be explained by comparing it to a spreadsheet program, where the configuration of the calculations corresponds to the configured process model in APROS — APMS, and the figures input into the calculation compares to the dynamic variables of the PROCESS model. The simulation system can be defined by using the APROS — APMS graphic client interface. Also, model definitions can be created and manipulated using the APROS command interpreter from ASCII command files. The simulated process is described by means of modules. A module represents a well defined part or component in a physical process. The modules and their mutual connections are mapped onto a computational network and the process model can be simulated and modified without any program recompilations. APROS — APMS simulation environment consists of an intuitive user interface program, the APROS simulation server program with toolkits, simulation software libraries and application model specific files.

(c) ***CASSIM (CASsiopeia SIMulation) Simulation Development System from CTI (Cassiopeia Technologies Inc.), Toronto, Canada***

The CASSIM™ based simulation development software (www.cti-simulation.com) is made up of a **Dynamic Linked Library (DLL)** of generic algorithms, consisting of supporting FORTRAN subroutines that model various types of power plants using CASSIM™ block oriented modeling methodology. Generic software algorithms were developed to correspond to physical plant components such as a process, a logical unit, or equipment (e.g. valve). The block oriented modeling technique simulates a large complex process system as a collection of these smaller components, each representing a subset of the larger system, and each is connected to one another to represent the detailed interactions between the subsystems.

CASSIM™ is a simulation development system based on three common core technologies: (1) ***CASSBASE***: the database engine which is used to manage the library of generic algorithms, and to manage the *interconnections* between the blocks in a model. In addition, it is also used to manage the ***Hydraulic Flow Networks.*** More importantly, as the model is being developed into a number of blocks, it is used for *debugging* so that the model developer can view each block's input and output values during each simulation iteration. Furthermore, the developer can dynamically modify these values, or even block interconnections on-line *without recompiling* the simulation code. In support of the model maintenance and upgrade, CASSBASE can be used for model "merging" and "extraction". (2)***CASSENG*** is the real time simulation run-time engine. It is used to run the model data file produced by *CASSBASE*, and to control the simulation such as "freeze", "iterate", "run" in response to user's command. To enable

inter-task communication within Microsoft Windows operating system, *CASSENG* supports Dynamic Data Exchange (DDE), so that *CASSBASE* can view block's details during debugging, and LabView can represent the plant's simulated data via its graphical user interface. (3) ***LABVIEW:*** *LABVIEW for Windows* is the *user interface development environment* from National Instruments of Texas, U.S.A., with many value-added features provided by C.T.I. for the purpose of developing user-friendly graphical interfaces for simulator applications. Graphical screens with buttons, icon symbols, pop-up dialogs, etc. are developed using LabView for Windows. User interface symbols, and special-purpose modules are provided in the form of LabView libraries. CTI has the custom LabView libraries which provide power plant simulation related symbols, modules for pop-up's, for performing Microsoft Windows Dynamic Data Exchange (DDE) and TCP/IP communication with CASSENG, and more. Through either DDE or TCP/IP, user interface screens request data from the simulation running in CASSENG for display and monitoring. Likewise, user input interactions such as specific control actions and setpoint changes can be transmitted via button status changes sent to the running simulation in CASSENG.

### 1.5 Technology Trends for Modular Modeling System

1. **Object Orientation** — the current prevailing MDS technology is predominantly block oriented; each block is defined by a set of input-output equations. With the advent of object oriented software design, object oriented modeling languages begin to appear. Object orientation allows for a definition of classes which group components with common properties. These classes can then be extended to form more refined components by simply adding to the properties which are inherited from the parent class, thus maximizing the reuse of existing models and minimizing the amount of programming.

2. **Distributed Computing** — standalone workstations delivering several tens of millions per second are commonplace, and continuing increases in power are predicted. When these computer systems are interconnected by an appropriate high speed network, their combined computational power can be applied to solve a variety of computationally intensive applications. Indeed, network computing may even provide supercomputer-level computational power. Further, under the right circumstances, the network based approach can be effective in coupling several similar multiprocessors, resulting in a configuration that might be economically and technically difficult to achieve with supercomputer hardware.

## 1.6 Simulation Fundamentals

### 1.6.1 Dynamic Simulation versus Steady–State Simulation

Process simulation is the use of mathematical models to develop a realistic representation of the real process behavior, as measured by the dynamic responses of the monitored process variables such as levels, temperature, pressures, and materials compositions etc. Process simulation can be divided into two categories: Steady–State and Dynamic.

- Steady–State simulation usually consists of groups of algebraic equations solved iteratively to account for the steady–state heat and material balance calculations of the process under various steady–state conditions. In all these steady state conditions, mass flow and energy flow equilibrium are always assumed. That is, the rates of change of mass and energy hold up in the process, known as accumulation terms, are set to zero. Steady–state simulation programs are commonly used in the process industry such as sizing a boiler, turbine equipment for a specific plant's heat rate requirements.

- Dynamic simulation is not only concerned with the steady–state heat and material balance calculations, but also the transient conditions of process evolution — for example, the dynamic response of boiler level due to sudden loss of turbine generator in a power plant. This is accomplished by solving a large set of coupled, non-linear differential equations in real time, that represent the dynamic material and energy balances for the process being simulated. The mass and energy accumulation rates are computed continuously and integrated over small time intervals, to produce realistic dynamic responses imitating process levels, temperatures, pressures, and materials compositions etc. Because dynamic simulation can provide dynamic responses of process over time, it is widely used in:

  1. The development of training simulators (for operator training).

  2. Safety analysis for nuclear power plant.

  3. Control design studies for process controls.

### 1.6.2 Mathematical Modeling

Mathematical Models of process are obtained by applying to each process component the laws of conservation of mass, energy and momentum, and the laws of physics, as well as the appropriate semi-empirical correlation of flow and heat transfers etc. This results in a set of coupled Differential, Algebraic and Boolean equations that are solved by digital computer in time series manner.

Formulating real time simulations of physical process are, to a large extent, very much situation-dependent. There is no one "all-encompassing" model which covers every possible situation of the process conditions. For

example, the technique which applies to a single phase heat exchanger differs considerably from those which apply to a two phase heat exchanger. Hence, correct simulation problem definition is the first and the most critical step in simulation. This is commonly known as "**Model Specification**".

The simulation analyst must identify all of the "situations" which are to be "simulated" by his/her model over the entire operating range, including "situations" mitigated by component malfunctions (valves, pumps etc.). As well, it is important to identify simulated conditions (single phase; two phases) in an equipment such as heat exchanger so that the model will give correct dynamic responses. However, as a caution, one must be able to balance the needs for accurate models versus expensive model development cost, and *NOT* to "over-specify" the requirements for the model, resulting in paying for an expensive model feature for insignificant training (or design) value.

Therefore development of physical process models utilizes intuition and good judgment which comes with experience; as well as classical scientific method of observation, classification, hypothesis, and test.

- Intuition is required to established the basic assumptions, major relationship between key variables, and the initial approaches to establishing the physical process model.

- Judgment is required to preserve a balance between accuracy and completeness versus complexity and cost.

The generic modeling techniques discussed in this lesson cover only basic process simulation of flow and heat transfer. It should be noted that some complex systems and important plant processes would require more complex modeling techniques to simulate specific phenomena such as two phase flow and steam bubbles formation. Examples for these systems are found in a nuclear power plant (e.g. reactor coolant system; steam generator; pressurizer; etc.); as well as in a fossil-fired plant (e.g. waterwalls; furnace boiler system etc.)

### 1.6.3 Model Specification

Prior to developing the physical process model, as stated previously, the purpose and objectives of the model should be clearly stated. This is known as "Model Specification". In general, the following parameters should be kept in mind for the development of the physical process model:

- Scope or boundary of the process under study — practically speaking, scope and boundary definition would require "marking" up Process & Instruments Devices (P & ID) Diagram (diagram detailing the process flow, pipework arrangement, equipments and instruments associated with a process subsystem.)

- Depth of the details — specify whether every process, every equipment (pump, valve, etc.), every instrument (pressure transmitter, temperature transmitter etc.) shall be simulated. If simulation is required, should it be a detailed simulation (down to the single electrical fuse); or functional simulation is adequate?

- Physical and Safety constraints — In case the process model is required to simulate the extreme physical and safety constraints such as pipe break; two phase phenomenon; reverse flow; boiler implosion/explosion, how long should the simulation last under such extreme constraints?

- Steady–State and Dynamic responses — these responses should be obtained from actual process operational data.

- Accuracy required — also called model fidelity. How accurate should the simulation be, when compared with the actual process transients?

- Need and method for updating model

- State variables and available control variables — State variables of a dynamic system are the smallest set of variables which determine the state of the dynamic system. For instance, if at least N variables (X1(t), X2(t), ... Xn(t) — e.g. pressure, flow etc.) are needed to completely describe the behavior of a dynamic system, then such N variables X1(t), X2(t), ...Xn(t) are a set of state variables for the dynamic process. State variables need to be identified in the model.

  A simple input-output Block Diagram is shown in Figure 2 for a typical control system associated with a plant process. The controller produces control signals (**Control Variables**) based on the discrepancies between the reference **Input Variables** (setpoints) and the **Manipulated Variables** (e.g. tank level, a subset of the state variables) from the plant process. In practical situations, there are always some **disturbances** acting on the plant. These may be external or internal origin, and they may random or predictable. The controller must take into consideration any disturbances which will affect the **Output Variables**.

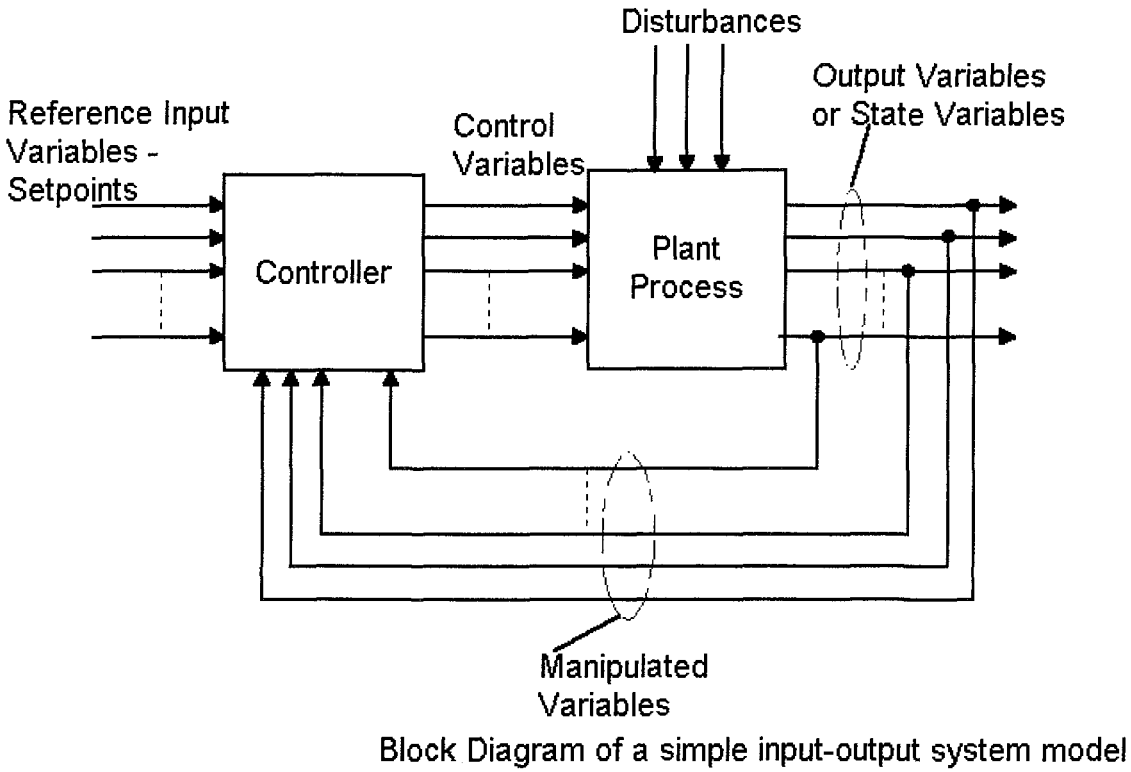



Figure 2 — Typical Block Diagram of a control system associated with a plant process

- Disturbances — all disturbances to the process need to be identified.

We have only presented here a general introduction of Model Specification. Detailed discussion is a course by itself and is not within the scope of this course.

### 1.6.4 Basic Physical Laws

The foundation for the development of physical process models consists of the basic physical laws of the following types of systems: pneumatic, hydraulic, thermal, mechanical, and electrical.

**(A) Unified Approach of System Analysis**

Despite the fact that there are distinct types of systems (pneumatic, hydraulic, thermal, mechanical, and electrical), a Unified Approach of System Analysis can be applied to these systems so that the same basic laws of system analysis can still apply to them, regardless of the type of system it belongs.

The following notions and definitions should be introduced before the Unified Approach of System Analysis is explained:

- **Control Volume** - also known as "element", or "block" — Control Volumes are physical objects whose behavior may be defined by measurements performed with respect to two points in space. They are the unit building blocks of a system. Control Volume may store,

transmit, convert, and dissipate energy. Typical control volumes are valves, pipes, generators, motors, vessels, and heat exchangers.

- **Junctions** — Junctions are points which join together one or more Control Volumes together. No energy, storage, transmission, conversion, or dissipation of energy takes place in junctions. Typical junctions are electrical busses, and connectors of pipes to vessels.

- **System** — a collection of Control Volumes where the net sum of energy is zero. Or energy generated is equal to energy converted and dissipated.

- **Quantity Variable** — a **State Variable** which describes the quantity contained in a Control Volume. A Quantity Variable is a "one-point" variable; i.e. it can be defined by one point in space. Example of quantity variables are *volume, enthalpy, momentum, and electrical charge* etc. Junctions cannot contain quantity variables, since they only join elements together.

- **Flow Variable** — the rate of change of Quantity Variable with respect to time. It is also a "one-point" variable. Examples are flow rate, enthalpy-flow, force, and electrical current, etc.

- **Potential Variable** — a State Variable which measures across the two ends of a Control Volume, i.e. it is a "two-point" variable in space. Examples of Potential Variable are *pressure, temperature, velocity, and voltage*, etc. In many cases, one of the point in this "two-point" variable is a reference point, e.g., melting point of ice, ground potential, and atmospheric pressure.

Figure 3 illustrates the definition of Quantity, Flow, Potential Variables of a Control Volume. Figure 4 illustrates Junctions as points which join together one or more Control Volumes.

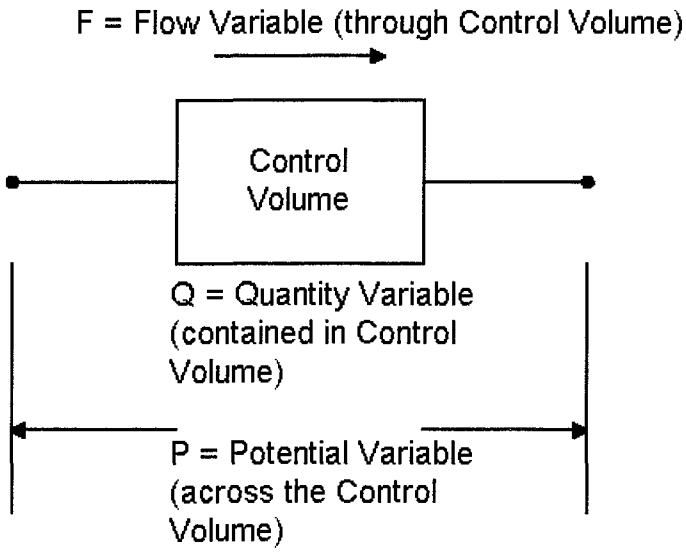



Figure 3 Quantity, Flow and Potential Variable of a Control Volume

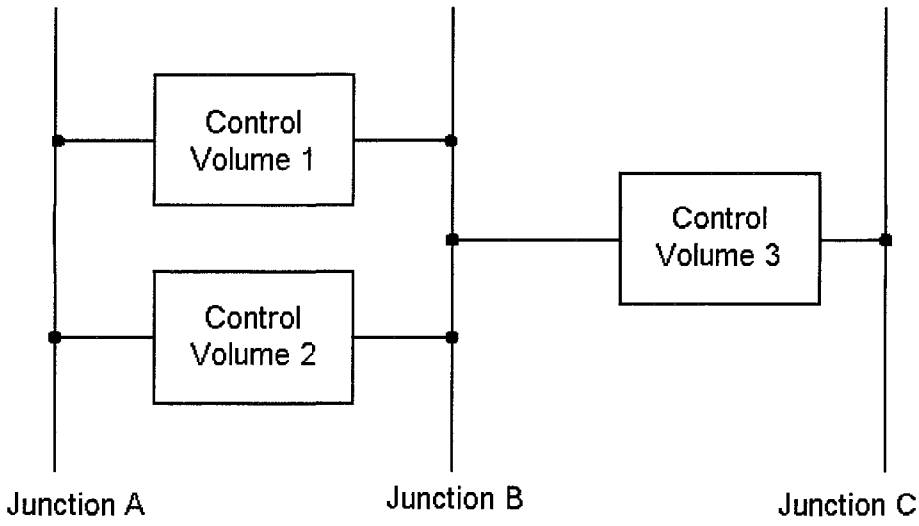



Figure 4 - Junctions defined as points which join together one or more Control Volumes

The following Table 1 summarizes the Quantity Variables, Flow Variables, and Potential Variables used for pneumatic, hydraulic, thermal, mechanical, and electrical systems. The units for these variables are listed below:

| **Type of System** | **Quantity Variable** | **Flow Variable** | **Potential Variable** |
|---|---|---|---|
| Pneumatic | Weight (lb) | Flow Rate (lb/sec) | Pressure (Psi) |
| Hydraulic | Volume (ft**3) | Flow Rate (ft**3/sec) | Head (ft) |
| Thermal | Enthalpy (BTU/lb) | Enthalpy Flow (BTU/lb-sec) | Temperature (Degrees) |
| Mechanical | Momentum (lb-sec) | Force (lb) | Velocity (ft/sec) |
| Electrical | Charge (Coulomb) | Current (ampere) | Voltage (volt) |

**(B) Two Basic Laws For System Analysis**

There are two basic laws for system analysis which are based on the laws of "**Conservation of Energy**", and the "**Continuity of Materials**".

1. The sum of all flow variables for any Junction is equal zero.
2. The sum of all potential variables for any closed loop in the system is equal to zero.

As shown in Figure 5, these two laws can be expressed:

For Junction n,

$$\sum_i F_{ni} = 0 \qquad \text{.....(1.1)}$$

where Fni is the Flow Variable from ith Control Volume going into (or away) from Junction n.

For the mth Loop,

$$\sum_j P_{mj} = 0 \qquad \text{.....(1.2)}$$

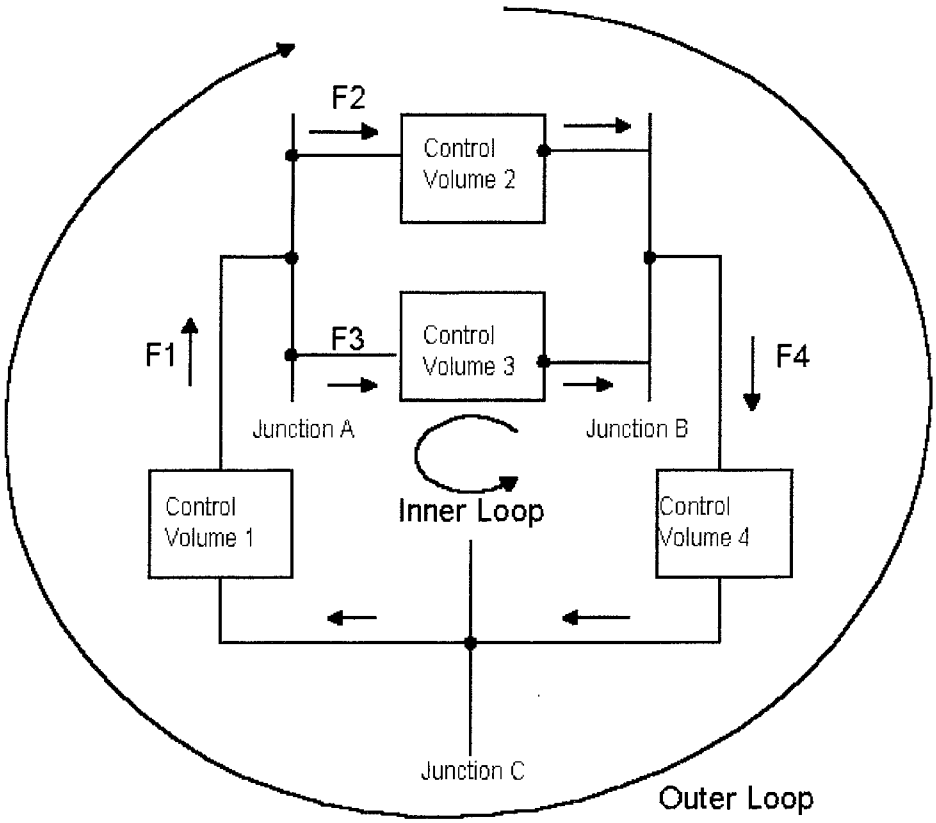



Figure 5 Typical System used to illustrate the two laws of system analysis

where Pmj is the Potential Variable for jth Control Volume

Applying the First Law for Junctions A, B, C, respectively,

**F1 - F2 - F3 = 0**

**F2 + F3 - F4 = 0**

**F4 - F1 = 0**

Applying the Second Law for Inner and Outer Loops, respectively,

**P1 + P2 + P4 = 0**

**P1 + P3 + P4 = 0**

Note that the direction of flow should be taken into account in the flow equations. In the potential equations, the potential variable is assumed to be measured in the direction of the flow variable. Thus P1 is the potential variable measured across junctions C to A.

Therefore applying the Two Basic Laws of System Analysis to the different types of system, we have:

| Type of System | First Law: Flow Variable connected to a Junction | Second Law: Potential Variable around a loop |
|---|---|---|
| Pneumatic | $\Sigma$ (flow rates ) = 0 | $\Sigma$ (pressures) = 0 |
| Hydraulic | $\Sigma$ (flow rates) = 0 | $\Sigma$ (heads) =0 |
| Thermal | $\Sigma$ (enthalpy flows)= 0 | $\Sigma$ (temperatures)=0 |
| Mechanical | $\Sigma$ (forces) = 0 | $\Sigma$ (velocities)=0 |
| Electrical | $\Sigma$ (currents) =0 | $\Sigma$ (voltages) =0 |

### 1.6.5 Physical Process Characteristics of a Control Volume

This section will discuss the physical process characteristics of a Control Volume, particularly:

- the relationship between the Quantity Variable and the Potential Variable
- the relationship between the Flow Variable and the Potential Variable
- the concept of Resistance and Capacitance

First the general case will be developed, then the characteristics of typical pneumatic, hydraulic, thermal, mechanical, and electrical systems will be presented.

**(1) General Case**

Let the following symbols be assigned:

Q = Quantity Variable

F = Flow Variable

P = Potential Variable

R = Resistance

C = Capacitance

The Resistance of a Process Control Volume is defined as the rate of change of the Potential Variable with respect to the Flow Variable, and is given by:

$$R = \frac{dP}{dF} \quad \text{.....(1.3)}$$

The Capacitance of a Process Control Volume is defined as the rate of change of the Quantity Variable with respect to the Potential Variable, and is given by:

$$C = \frac{dQ}{dP} \quad \text{.....(1.4)}$$

If a Control Volume has both Resistance and Capacitance, it is further assumed that this Control Volume is divided into two elements, with one Resistance element connected in series to the other Capacitance element. Thus the Change in Potential Variable of this Control Volume can be expressed as :

$$dP = RdF + \frac{1}{C}dQ \quad \text{.....(1.5)}$$

The Resistance and Capacitance for the various physical systems are introduced below as illustration of the concepts, without the detailed derivation of the appropriate equations.

**(2) Pneumatic Systems**

Gas flow through small orifice can often be considered as turbulent, whereas gas flow at certain velocities through larger pipes is often considered as laminar. Elements which often produce turbulent flow are orifices, valves, and small pipes. Elements which subsequently produce laminar flow are circular tubes and pipes. The Quantity variable for pneumatic systems is Weight, the Flow Variable is Flow Rate, and the Potential Variable is Pressure.

Thus the Resistance to laminar gas flow in an adiabatic condition is

$$R = \frac{dP}{dF} = \frac{128\mu \cdot L}{\pi \cdot \gamma \cdot D^4} \quad \text{.....(1.6)}$$

Where R = Pneumatic resistance (sec per square foot)

P = Pressure across the Control Volume, in (Psi)

F = Flow rate through the Control Volume (lb/sec)

L = Length of the pipe (ft)

D = Inside Diameter of pipe (ft)

g = gravitational constant, (ft/sec**2)

K = flow coefficient, dimensionless

A = area of restriction, (ft**2)

Y = rational expansion factor (lb/ft**3)

$\gamma$ = Gas density (lb/ft**3)

$\mu$ = Absolute viscosity, (lb-sec/ft**2)

The resistance to turbulent gas flow in an adiabatic condition is

$$R = \frac{dP}{dF} = \frac{\sqrt{2}}{KAY}\left(\frac{\gamma \cdot V}{g}\right)^{0.5} \quad \text{.....(1.7)}$$

and $$K = \sqrt{\frac{fD}{L}} \quad \text{.....(1.8)}$$

Where V is the head; f is the friction factor; D is the inside diameter of the orifice, and L is the equivalent length.

The Capacitance of a pneumatic vessel is given by:

$$C = \frac{dQ}{dP} = \frac{V}{nGT} \quad \text{.....(1.9)}$$

Where C = Pneumatic capacitance, (ft**2)

Q = Weight of gas in vessel (lb)

V = Volume of vessel (ft**3)

G = Gas constant for specific gas (ft/deg. C)

T = Temperature in (Degrees C)

n = polytropic constant = ratio of specific heats for adiabatic expansion

**(3) Hydraulic System**

Hydraulic systems also have two types of flow: turbulent flow where the Reynolds number ( a function of geometry, flow rate and viscosity) is greater than 4000; and laminar flow if Reynolds number is less than 2000. The Quantity Variable of Flow Rate; Potential Variable is pressure.

The Resistance of laminar flow hydraulic flow is:

$$R = \frac{dP}{dF} = \frac{128\mu \cdot L}{\pi \cdot \gamma \cdot D^4} \quad \text{.....(1.10)}$$

The Resistance of turbulent hydraulic flow is:

$$R = \frac{dP}{dF} = \frac{F}{gK^2A^2} \quad \text{.....(1.11)}$$

and $$K = \frac{1}{KA}\sqrt{\frac{2P}{g}} \quad \text{.....(1.12)}$$

Where R = Hydraulic resistance (sec per square foot)

C = Hydraulic capacitance (ft**2)

P = Head, in (ft)

F = Flow rate through the Control Volume (ft**3/sec)

Q = Volume (ft**3)

L = Length of the pipe (ft)

D = Inside Diameter of pipe (ft)

g = gravitational constant, (ft/sec**2)

K = flow coefficient, dimensionless

A = area of restriction, (ft**2)

$\gamma$ = fluid density (lb/ft**3)

$\mu$ = Absolute viscosity, (lb-sec/ft**2)

The hydraulic capacitance for a uniform cross-section tank is:

$$C = \frac{dQ}{dP} = A \quad \text{.....(1.13)}$$

where A is the cross section of the tank at the liquid surface.

**(4) Thermal Systems**

There are three types of heat transfer: conduction, convection and radiation. Conduction takes place when heat is transferred through a solid object. Convection takes place when heat is transferred through the movement of liquid or gas in a vessel. Radiation is the transfer of heat from the surface of one object to another object at a distance. The Quantity

Variable is Enthalpy, or commonly known as heat; the Flow Variable is Enthalpy-Flow, or Heat Flow; the Potential Variable is Temperature.

The thermal resistance for conduction through a homogeneous material is

$$R = \frac{dP}{dF} = \frac{X}{KA} \quad \text{.....(1.14)}$$

The thermal resistance for forced convection is

$$R = \frac{dP}{dF} = \frac{1}{HA} \quad \text{.....(1.15)}$$

Where F = heat flow through the Control Volume (BTU/sec)

P = temperature across the Control Volume (deg.)

R = Thermal resistance (deg-sec/BTU)

C = Thermal capacitance (BTU/deg)

K = Thermal conductivity (BTU/ft-sec-deg)

H = Convection coefficient of the Control Volume (BTU/ft**2-sec-deg)

A = Cross-sectional area of the Control Volume (ft**2)

X = Thickness of the Control Volume (ft)

The thermal capacitance of a block is

$$C = \frac{dQ}{dP} = W \cdot S \quad \text{.....(1.16)}$$

Where Q = heat stored (BTU)

W = Weight of block (lb)

S = Specific heat at constant pressure (BTU/deg-lb)

**(5) Mechanical Systems**

There are two kinds of mechanical systems: translational and rotational. Mechanical systems have an additional characteristic that differentiate form pneumatic, hydraulic and thermal systems. In addition to resistance and capacitance, mechanical systems have the characteristics of mass, known as Moment of Inertia.

Only translational mechanical system is presented here. Consider a mechanical spring system. The Quantity Variable is Momentum; Flow

Variable is Force; Potential Variable is Velocity. According to Hooke's Law, the spring constant is the inverse of the derivative of displacement with respect to force.

$$K = \frac{1}{dD/dF} = \frac{dD}{dF} = \frac{dD/dt}{dF/dt} = \frac{Velocity}{Momentum} \quad \text{.....(1.17)}$$

where D = displacement (ft)

F = Force (lb)

$$K = \frac{1}{dD/dF} = \frac{dD}{dF} = \frac{dD/dt}{dF/dt} = \frac{Velocity}{Momentum} \quad \text{.....(1.18)}$$

Recall the definition of Capacitance = dQ/dF = Momentum/Force. Hence the spring constant is analogous to Capacitance.

Recall that the definition of Resistance = dP/dF = Velocity/Force. This is analogous to damping in a hydraulic piston system, where viscous damping is the result of the force resisting the rate of change piston displacement.

**(6) Electrical Systems**

The Quantity Variable of an Electrical System is Charge; Flow Variable is Current; Potential Variable is Voltage. The three characteristics of an electrical system are: resistance, capacitance, and inductance.

The Resistance is equal to the derivative of the voltage across an element with respect to current

R = dP/dF = dV/dI, .....(1.19)

where V = Voltage (volts)

I = Current ( Amp)

Capacitance of a electrical element is

C = dQ/dP = d(Charge)/dV, .....(1.20)

The Inductance is equal to

L = dV/dI.....(1.21)

**(7) Summary**

Table 2 summarizes the physical characteristics for various physical systems:

| **Type of System** | **Quantity Variable** | **Flow Variable** | **Potential Variable** | **Resistance** | **Capacitance** |
|---|---|---|---|---|---|
| Pneumatic | Weight (lb) | Flow rate (lb/sec) | Pressure (psi) | Laminar flow: $R = \frac{dP}{dF} = \frac{128\mu \cdot L}{\pi \cdot \gamma \cdot D^4}$ Turbulent flow $R = \frac{dP}{dF} = \frac{F}{gK^2A^2}$ | $C = \frac{dQ}{dP} = \frac{V}{nGT}$ |
| Hydraulic | Volume (ft**3) | Flow rate (ft**3/sec) | Head (ft) | Laminar flow: $R = \frac{dP}{dF} = \frac{128\mu \cdot L}{\pi \cdot \gamma \cdot D^4}$ Turbulent flow $R = \frac{dP}{dF} = \frac{F}{gK^2A^2}$ | $C = \frac{dQ}{dP} = A$ |
| Thermal | Enthalpy (BTU/lb) | Enthalpy flow (BTU/lb/ sec) | Temperatures (degrees) | Conduction $R = \frac{dP}{dF} = \frac{X}{KA}$ Forced Convection $R = \frac{dP}{dF} = \frac{1}{HA}$ | $C = \frac{dQ}{dP} = W \cdot S$ |
| Mechanical | Momentum (lb-sec) | Force (lb) | Velocity (ft/sec) | Viscous Damping Constant | Spring Constant |
| Electrical | Charge (coulomb) | Current (ampere) | Voltage (volt) | V/I | Q/V |

# 2.0 Overview of Dynamic Simulation for Power Plant Process

## 2.1 Introduction

Formulating real time simulations of physical processes is, in some respects, more difficult than formulating other quantitative models. This is so because as of to date, there is no cogently expressed situation-independent theory of real time simulation. Over the years, various modeling techniques have evolved; the more successful survived and the less successful have not. What is presented here then is a compendium of techniques which are of necessity situation-oriented.

The fundamental mathematical modeling approach to the simulation of power plant processes is based on the systematic application of first principles to all principle plant systems. We have discussed the concept of "Control Volume" in Chapter 1. Thus:

- Each system is segmented in a number of "Control Volumes" or "Blocks" in which volume averaged properties (e.g. enthalpy, pressure, flow) are obtained from the application of conservation of mass, energy and momentum. The resulting equations are obtained in the transient form, unless the time constants associated with the variables they represent cannot be perceived by the operator.

- State of the art engineering equations and/or correlations are used in order to describe accurately energy and mass transfers from one "Control Volume" to another, over the entire operational range to be covered by the models.

- Equations of the State are approximated by the ideal gas law or by accurate curve-fits to Steam Tables over the entire operational range of the simulated plant.

- Local distributions of a variable within a Control Volume are obtained from the volume averaged quantities and an appropriate correlation taking into consideration local effects such as temperature distribution in containment atmosphere in a nuclear plant.

It should be emphasized that the fundamental properties of the simulated systems are represented, whether they are directly observable from the control room instrumentation or not. However, depending on the purpose of the simulation, typically, when there is limited instrumentation connected to those systems, their simulation may be simplified and limited to the extent that they can be operated from the main control room. For example.

- Alarms associated with auxiliary tank levels may be logically simulated.

- Indications associated with water chemistry, which are not influenced by operator actions may be made parameters adjustable by instructor inputs.
- Loads of small components (e.g. valves) may be lumped.

### 2.2 Basic Equations

In general, most process algorithms are based on the principles of:

- Overall Mass Balance
- Component Mass Balance
- Heat (energy) Balance
- Momentum Balance

The general unsteady–state equation that applies in all cases is:

**Accumulation = Input — Output**

Where accumulation term is usually time derivative of :

- Mass for liquid mass balance
- Pressure for vapor or gas mass balance
- Temperature or enthalpy for heat balance
- Mass flow rate for momentum balance

### 2.3 Mass and Component Balance Equations

Consider overall mass balance is applied to a "Control Volume" containing a liquid. It has i inlet streams and j outlet streams:

Rate of Mass Accumulation = Rate of Mass flow in — Rate of Mass flow out

$$\frac{dM}{dt} = \sum_{i}^{n} W_{in}^{(i)} - \sum_{j}^{m} W_{out}^{(j)} \quad \text{.....(2.1)}$$

Where:

M = Mass of liquid in the "Control Volume" at any given time.

$W_{in}^{(i)}$ = Flow rate of inlet stream i into the "Control Volume".

$W_{out}^{(j)}$ = Flow rate of outlet stream j from the "Control Volume".

Normally, the level of liquid in the tank is the desired variable to be calculated:

$$L = \frac{M}{\rho \cdot A} \quad \text{.....(2.2)}$$

Where

L = level

$\rho$ = liquid density

A = tank cross-sectional area (assuming constant cross-sectional area)

Substituting equation (2.2) into (2.1), assuming constant density,

$$\frac{dL}{dt} = \frac{1}{\rho \cdot A}\left(\sum_i W_{in}^{(i)} - \sum_j W_{out}^{(j)}\right) \quad \text{.....(2.3)}$$

If concentrations of substances which are tracked by instrumentation (e.g. chemical composition), the component balance can be applied to the "Control Volume", assuming perfect mixing. Consider a Control Volume, with i inlet streams going into the Control Volume. Each inlet flow stream contains concentration of a given component xi. The question is to find the resulting of concentration X of the given component in the Control Volume, after mixing i inlet streams.

$$\frac{d}{dt}(M.X) = \sum_i (W_{in}^{(i)}.x_{in}^{(i)}) - (\sum_j W_{out}^{(j)}).X - S \quad \text{.....(2.4)}$$

where

$x_{in}^{(i)}$ = concentration of given component in inlet stream i

X = concentration of given component in the Control Volume

S = rate of depletion (or creation) of component by chemical reaction, if applicable.

It is also assumed that under perfect mixing, the concentration of any stream leaving the "Control Volume" is the same as that of the "Control Volume".

Expanding the left hand side of equation 2.4, and substituting equation 2.1. After re-arranging, we have:

$$\frac{dX}{dt} = \frac{1}{M}\left(\sum_i (W_{in}^{(i)}.(x_{in}^{(i)} - X)) - S\right) \quad \text{.....(2.5)}$$

## 2.4 Rate of Change of Pressure in a Control Volume

Equation (2.1) can be re-written as:

$$V.\frac{d\rho}{dt} + \rho\frac{dV}{dt} = \sum_i W_{in}^{(i)} - \sum_j W_{out}^{(j)} \quad \text{.....(2.6)}$$

Assuming the fluid is incompressible, i.e. constant volume, (2.6) becomes

$$V.\frac{d\rho}{dt} = \sum_i W_{in}^{(i)} - \sum_j W_{out}^{(j)} \quad \text{.....(2.7)}$$

In general, density is a function of pressure and enthalpy (energy), h, expanding the time derivative of density, we have:

$$\frac{d\rho}{dt} = \frac{\partial\rho}{\partial P}\cdot\frac{\partial P}{\partial t} + \frac{\partial\rho}{\partial h}\cdot\frac{\partial h}{\partial t} \quad \text{.....(2.8)}$$

Generally speaking, enthalpy changes very slowly in a Control Volume, when compared to pressure changes. That is

$$\frac{\partial h}{\partial t} << \frac{\partial P}{\partial t}$$

Hence dropping the second term in (2.8) :

$$\frac{d\rho}{dt} = \frac{\partial\rho}{\partial P}\cdot\frac{\partial P}{\partial t} \quad \text{.....(2.9)}$$

Substituting (2.9) into (2.7),

$$V.\frac{\partial\rho}{\partial P}\cdot\frac{\partial P}{\partial t} = \sum_i W_{in}^{(i)} - \sum_j W_{out}^{(j)} \quad \text{.....(2.10)}$$

or

$$C.\frac{dP}{dt} = \sum_i W_{in}^{(i)} - \sum_j W_{out}^{(j)} \quad \text{.....(2.11)}$$

Where C is defined as the Capacitance of the Control Volume

$$C = V.\frac{\partial\rho}{\partial P} \quad \text{.....(2.12)}$$

Note that $\frac{\partial \rho}{\partial P}$ is a measure of fluid compressibility. For instance, for incompressible fluid such as water, $\frac{\partial \rho}{\partial P}$ has the value of 4.43E-4 kg/m**3/KPa.

## 2.5 Momentum Equation for Links Joining Control Volumes

Consider two Control Volumes 1 and 2 are connected by a pipe. There is fluid flow from Control Volume 1 to Control Volume 2. In "hydraulic flow network" terminology, this pipe is known as a "link". Fluid flow in the link between two Control Volumes depends upon the physical nature of the flow path, as well as the flow conditions: laminar flow or turbulent flow, etc. For passive paths such as a pipe with a valve, the relationship for turbulent flow and constant density can be expressed by (Bernoulli momentum equation):

$$W = C\sqrt{P_1 - P_2} \ldots\ldots for P_1 > P_2 \ldots\ldots\ldots\ldots\ldots\ldots (2.13)$$

Where:

W = Mass flow rate

C = Flow conductance including valve position

P1 = link's upstream pressure (i.e. Control Volume 1 pressure)

P2 = link's downstream pressure (i.e. Control Volume 2 pressure)

If the pipe has a "check valve" ( i.e. only allows flow in one direction), the flow rate will become zero whenever downstream pressure is greater than upstream pressure. On the other hand, if flow reversal is possible for a particular situation, equation (2.13) becomes:

$$W = -C\sqrt{P_2 - P_1} \ldots\ldots if .. P_2 > P_1 \ldots\ldots\ldots\ldots\ldots\ldots (2.14)$$

If the fluid is gas, or vapor, the flow in the link connecting the two Control Volumes also depends on density. Hence the following equation may be used:

$$W = C \cdot \sqrt{\rho} \cdot \sqrt{P_1 - P_2} \ldots\ldots P_1 > P_2 \ldots\ldots\ldots\ldots\ldots\ldots (2.15)$$

Where C is flow Conductance including valve position.

## 2.6 Linearized Momentum Equation

For the ease of computation, it is necessary to have the momentum equation in linearized form. Thus equation (2.13) can be written as:

$$W = A.(P_1 - P_2) \quad \text{.....(2.16)}$$

Where A is defined as Admittance and is expressed as:

$$A = \frac{C}{\sqrt{P_1 - P_2}} \quad \text{.....(2.17)}$$

Note that as the pressure difference goes to zero in equation (2.17), the Admittance, A, will go to infinity. This can be prevented by limiting the value of A to some maximum value when the pressure difference goes below a predetermined valve.

### 2.7 Pressure Drop Calculation

If the flow in a link is known, but the pressure drop (difference) across the flow path is required, this can be easily done by re-arranging equation (2.13):

$$\Delta P = \left(\frac{W}{C}\right)^2 \quad \text{.....(2.18)}$$

### 2.8 Pump Flow Equation

The pressure/flow relationship for a typical centrifugal pump inside a link connecting two Control Volumes 1 and 2 having pressure P1 and P2 respectively is expressed as (assuming incompressible flow):

$$W_{1,2} = C_p \cdot \sqrt{\Delta P_{max} \cdot S_p^2 - \Delta P} \quad \text{.....(2.19)}$$

Where:

W1,2 = flow from Control Volume 1 to Control Volume 2

Cp = Constant representing pump flow conductance

Sp = Normalized pump speed

$\Delta P_{max}$ = Maximum possible pressure rise across the pump. This also commonly known as pump's **Design Static Head.**

$\Delta P$ = Pressure rise across pump due to pumping action. This is also known as pump's **Dynamic Head.**

= Pump Discharge Pressure — Pump Suction Pressure

= (P2 - P1)

Note that the pump's actual static head is related to the Design Static head as follows:

Actual Static Head = Design Static Head * (Pump Speed)$^2$ *

(Fluid Density/Fluid Design Density) * (1 - degradation factor)

..... (2.20)

Thus at 100 % rated pump speed, if there is no fluid density change, and if the pump runs at 100 % efficiency (i.e. no degradation), then the Actual Static Head is equal to Pump Design Static Head.

Again converting equation (2.19) into linearized form,

$$W_{1,2} = A_p \cdot (\Delta P_{max} \cdot S_p^2 - \Delta P) \quad \text{.....(2.21)}$$

Where the pump Admittance Ap is defined as:

$$A_p = \frac{C_p}{\sqrt{\Delta P_{max} \cdot S_p^2 - \Delta P}} \quad \text{.....(2.22)}$$

As pressure rise increases to maximum pressure rise, Ap will approach infinity. This is prevented by limiting the value of Ap to some maximum value when the pressure rise increases above a predetermined value. The resulting pump model for pump flow as a function of pressure rise can be seen in Fig. 6.

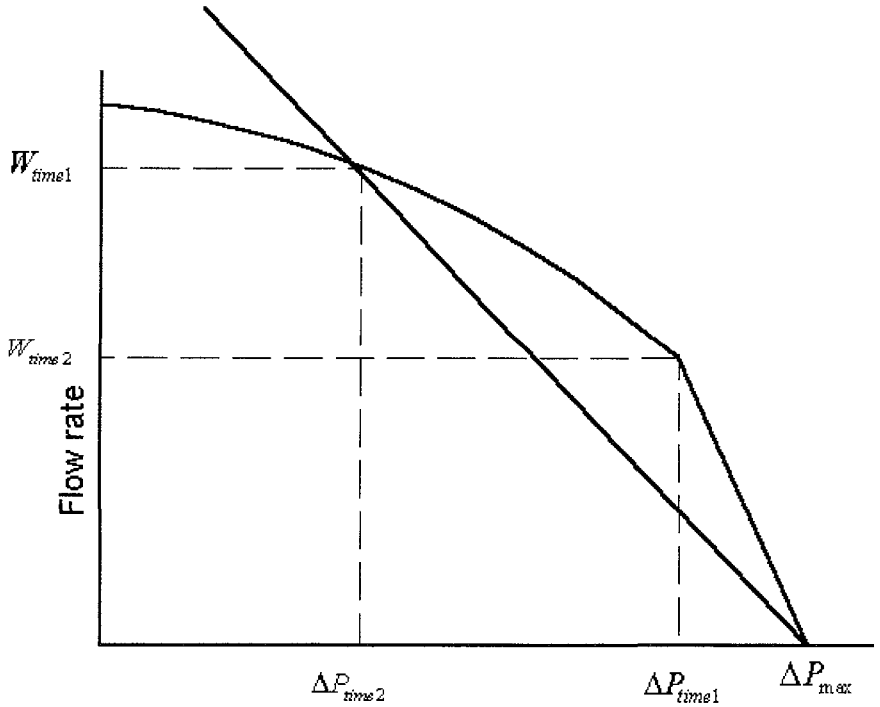



Figure 6 Pressure Rise (dynamic head)

With this model, it can be seen that as the pump is started, the dynamic head = (discharge pressure — suction pressure) is at its maximum. As pump flow is increased, the dynamic head decreases accordingly.

## 2.9 Flow Equation in Link with Pump and Valve

Consider a "link" connecting two Control Volumes 1 & 2 has a pump and a valve in series, the flow equation for this link can be expressed as:

$$W_{1,2} = C_{1,2} \cdot V_A \sqrt{P_1 + P_{DYH} - P2} \quad \text{.....(2.23)}$$

Where:

C1,2 = Link conductance

VA = Valve port area

P1 = Pressure at Control Volume 1

P2 = Pressure at Control Volume 2

PDYH = Pump Dynamic Head

## 2.10 Pump Cavitation Factor

In simulation, it often necessary to find out when pump will cavitate. The Pump Cavitation Factor (normalized) is given by:

$$Cavitation\ Factor = \frac{(Suction\ \text{Pr}essure - Saturation\ \text{Pr}essure)}{N.P.S.H} \quad \text{.....(2.24)}$$

Where N.P.S.H = Net Positive Suction Head required for the pump to function.

## 2.11 Flow Equation to Match Specific Pump Curve & Incompressible Flow

In certain simulation situations, it is necessary to model pump behavior to match manufacturer's pump performance curve, rather than the standard pump equation as described in equation (2.13). As well, the incompressible flow equations should be modified to reflect compressible flow characteristics for steam flow, gas flow etc. These are advanced topics which will be discussed in a more advanced course.

## 2.12 Energy Balance Equations

For a "Control Volume" containing a liquid, or vapor, or gas, the energy balance is generally expressed in terms of enthalpy. It is assumed that all the entering streams are perfectly mixed with the contents of the "Control Volume" and thus the volume averaged value of enthalpy is calculated as a function of time. As well, all outlet streams from the "Control Volume" are assumed to have the same volume averaged enthalpy. Hence,

*The Rate of Energy Accumulation in Control Volume =*

*Rate of energy inflow from the incoming streams — Rate of energy outflow from the outgoing streams + (or - ) Rate of external energy gain (or loss)*

Mathematically,

$$\frac{d(M \cdot H)}{dt} = \sum_i (W_{in}^{(i)} \cdot H_{in}^{(i)}) - \sum_j (W_{out}^{(j)} \cdot H_{out}^{(j)}) + (or-)\ Q \quad \text{.....(2.25)}$$

Where:

M = Mass of fluid in Control Volume

H = Volume averaged enthalpy in control volume

Win = Mass flow rate for incoming stream

Wout = Mass flow rate for outgoing stream

Hin = Enthalpy of incoming stream

Hout = Enthalpy of outgoing stream

Q = rate of external energy gain (or loss)

Expanding the left side and substituting for dM/dt from equation (2.1), after re-arranging, equation (2.25) will become:

$$\frac{dH}{dt} = \frac{1}{M}\left(\sum_i W_{in}^{(i)} \cdot (H_{in}^{(i)} - H) + (or-)Q\right) \quad \text{.....(2.26)}$$

After integration of (2.26), the temperature can be calculated by means of a thermodynamic functions which relates temperature to enthalpy and pressure: T = f(H, P).

### 2.13 Rate of change of fluid Temperature in Control Volume

In some cases, if the fluid specific heat can be considered as constant over the operational range, then

$$\Delta H = C_p \cdot \Delta T \quad \text{.....(2.27)}$$

Equation (2.26) will become

$$\frac{dT}{dt} = \frac{1}{M \cdot C_p}\left(\sum_i W_{in}^{(i)} \cdot C_p^{(i)} \cdot (T_{in}^{(i)} - T) + (or-)Q\right) \quad \text{.....(2.28)}$$

Where Cp = Volume average specific heat of fluid in control volume

In the derivation of equation (2.28), it is assumed that the specific heat of inlet and outlet streams are equal.

## 2.14 External Heat Transfer Rate in Control Volume

The external heat transfer, Q, is normally calculated from:

$$Q = UA \cdot (T - T_s) \quad \text{.....(2.29)}$$

Where UA = Heat transfer coefficient * Heat Transfer Area

Ts = External Source temperature

## 2.15 Equations of State

Power plant processes mainly consist of heating water to make steam, passing steam through turbine, and then condensing the exhaust steam into water. This means thermodynamic properties of water and steam must be evaluated inside various equipments throughout the power plant. The approach usually taken is as follows:

- Material mass balance gives pressure, P (equation 2.11).
- Heat balance in control volume gives enthalpy, H (equation 2.26)
- All other thermodynamic properties can be calculated using established curve-fit of the steam table functions, using pressure and enthalpy as inputs:

for example:

Saturated Water Entropy = f (P)

Saturated Steam Entropy = f (P)

Saturated Steam Temperature = f(P)

Density of water, or steam, or saturated water/steam mixture = f(P, H)

Subcooled water temperature = f(H, P)

Superheated steam = f (P,H)

Subcooled water density = f(P,H)

Superheated steam temperature = f(P.H)

## 2.16 Two Phase Systems (Simple Equilibrium Model)

The following two-phase system model can be used in all vessels where two phase conditions exist under thermodynamic equilibrium. Examples of such applications include most vessels in the plant such as feedwater heaters. Exceptions are the specialized equipments such as steam generator, and the primary heat transport system in a nuclear plant. The vessel in this model has n inlet streams for steam flow. Steam condenses inside the vessel, due to heat transfer to external source (feedwater). At the

same time, the condensate is drained out from the vessel at a certain rate. If the drainage rate is less than the steam condensation rate, condensate will accumulate inside the vessel to form level L.

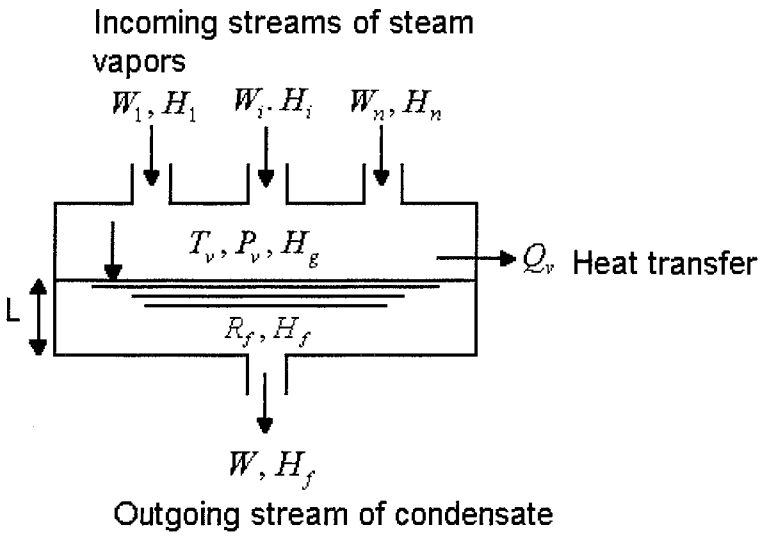



Figure 7 · An example of simple two phase system model

Symbols Definition:

$W_i$ = Steam flow rate for inlet stream i, i = 1...n

$H_i$ = Enthalpy of inlet stream i, i = 1....n

$T_v$ = Saturated steam temperature in vapor space of vessel

$P_v$ = Saturated steam pressure in vapor space of vessel

$H_g$ = Saturated steam enthalpy

$R_f$ = Density of condensate

$H_f$ = Saturated liquid enthalpy

W = Condensate outflow rate

$Q_v$ = Heat transfer from vessel vapor space

L = Level of Condensate

Mass Balance in the steam space —

$$\frac{dP_v}{dt} = \frac{1}{C} \cdot (\sum_{i=1}^{n} W_i - W_c) \ldots\ldots\ldots\ldots\ldots (2.30)$$

Where C is the Capacitance of the Vessel Steam Space. Wc is the steam condensation rate.

It can also be shown that by using Ideal Gas Law, assuming constant volume and constant temperature (recall Chapter #1 - on pneumatic system)

$$C = \frac{M.V_v}{Z.R.T_v}$$

Where Z = Compressibility Factor

R = Ideal Gas Law constant

M = mass of steam vapor

Vv = Volume of steam space

Wc = steam condensation rate

- Mass Balance of liquid space gives:

$$\frac{dL}{dt} = \frac{1}{R_f.A}(W_c - W) \ldots\ldots\ldots\ldots\ldots\ldots (2.31)$$

Where A is the cross sectional area of the vessel, assuming constant

- Overall Energy Balance to give condensation rate Wc -

$$W_c.(H_g - H_f) = \sum_{i=1}^{n} W_i \cdot (H_g - H_i) + Q_v \ldots\ldots\ldots\ldots (2.32)$$

Compute $T_v, H_f, H_g, R_f$ using saturated steam table curve fit, using Pv as input.

## 3.1 Introduction

The real time solution of a set of ordinary differential equations (ODE's) in a digital computer based simulator requires the application of appropriate numerical integration techniques to provide numerically stable solutions, given a specific integration time step.

In general for real time computation, classical Euler's first order integration techniques are employed in most simulation applications, because of simplicity of its integration scheme, stability control, and fixed time step.

Two Euler techniques are presented here:

- Explicit Euler Integration Technique
- Implicit Euler Integration Technique

The selection of Euler techniques depends on the analysis of the numerical stability conditions for the equation set to be solved. It is important to note that the selection of the proper technique ensures numerical stability of the algorithms, and does not require any artificial treatment of the equation set to mask local stability problems.

Other important modeling considerations that will be discussed include:

- Range of Validity
- Realism
- Reliability
- Real time Simulation

## 3.2 Explicit Euler Integration

Consider the following ODE:

$$\frac{dX}{dt} = f(X) \quad \text{.....(3.1)}$$

Where f can be either a linear or non-linear function of the dependent variable X.

The solution of X(t) of equation (3.1) is defined in the interval $t > t_0$ if an initial value of X is known for $t = t_0$ , that is

$$X(t_0) = X_0$$

A numerical solution of equation (3.1) can be obtained by approximating the time derivative of X by a "backward difference" over the time interval $\Delta t$:

$$\frac{dX}{dt} = \frac{X - X^*}{\Delta t} = f(X^*) \quad \text{.....(3.2)}$$

where:

X= X(t)

$X^* = X(t - \Delta t)$

With this Explicit Euler Integration Technique, the solution X at the instant t can be found as a function of the solution at the previous time step $(t - \Delta t)$ from (3.2):

$$X = X^* + \Delta t \cdot f(X^*) \quad \text{.....(3.3)}$$

with initial value $X(t_0) = X_0$

This integration technique is computationally most efficient. As for every first order integration algorithm, it can be shown that its accuracy is of the order of $\Delta t$ . Therefore, the error on the transient solution is proportional to the time step. Hence, the smaller the integration time step, the more accurate solution will be.

The drawback of the Explicit Euler integration method is that it is conditionally stable, that is, a stable solution will be obtained only under certain conditions.

The stability condition can be studied as follows. Consider at the instant t, the calculated solution is equal to the exact solution, plus an error term $\varepsilon$:

$$X_c = X + \varepsilon \quad \text{.....(3.4)}$$

Thus , at each time step, the calculated $X_c$ is obtained from (3.3) by substituting (3.4):

$$X + \varepsilon = X^* + \varepsilon^* + \Delta t \cdot f(X^* + \varepsilon^*) \quad \text{.....(3.5)}$$

Replacing $f(X^* + \varepsilon^*)$ by its Taylor expansion, limited to first order terms:

$$f(X^* + \varepsilon^*) = f(X^*) + \varepsilon^* \cdot \frac{\partial f}{\partial X} \quad \text{.....(3.6)}$$

Substituting (3.6) into (3.5), we obtain:

$$X + \varepsilon = X^* + \varepsilon^* + \Delta t \cdot f(X^*) + \Delta t \cdot \varepsilon^* \cdot \frac{\partial f}{\partial X} \quad \text{.....(3.7)}$$

Again substituting (3.3) into (3.7), after re-arranging, we have obtained a relationship between the error term at the current time and that at the past time step:

$$\varepsilon = \varepsilon^* \cdot (1 + \Delta t \cdot \frac{\partial f}{\partial X}) \quad \text{.....(3.8)}$$

Re-arranging to obtain ratio of error terms, and taking absolute value on both side of the equation:

$$\left|\frac{\varepsilon}{\varepsilon^*}\right| = \left|1 + \Delta t \cdot \frac{\partial f}{\partial X}\right| \quad \text{.....(3.9)}$$

In order for the solution to be stable, $\left|\frac{\varepsilon}{\varepsilon^*}\right| < 1$ That is,

$$\left|1 + \Delta t \cdot \frac{\partial f}{\partial X}\right| < 1 \quad \text{.....(3.10)}$$

Thus, to obtain a numerically stable solution, the following conditions must be satisfied:

(a) $$\frac{\partial f}{\partial X} < 0 \quad \text{.....(3.11)}$$

(b) $$\Delta t \cdot \left|\frac{\partial f}{\partial X}\right| < 2 \quad \text{.....(3.12)}$$

Condition (a) is a measure of the equation's natural time constant and is always satisfied if the equation is properly formulated.

Condition (b) depends on the value of partial derivative of f, as the time step $\Delta t$ is fixed.

Examples where this technique is applied are:

- Valve actuators
- Thermocouples

Typically the differential equation describing these devices is :

$$\frac{dX}{dt} = \frac{X_0 - X}{\tau} \quad \text{.....(3.13)}$$

Where X0 = initial value of X

$\tau$ = time constant

Clearly, this integration technique works well with sufficiently large time constant. (Why ? )

### 3.3 Implicit Euler Integration

A numerical solution of equation (3.1) can also be obtained by approximating the time derivative by a forward difference over the time interval $\Delta t$:

$$\frac{dX}{dt} = \frac{X - X^*}{\Delta t} = f(X) \quad \text{.....(3.14)}$$

and $$X = X^* + \Delta t \cdot f(X) \quad \text{.....(3.15)}$$

Applying the same error analysis as in Explicit Euler Integration, we obtain

$$X + \varepsilon = X^* + \varepsilon^* + \Delta t \cdot f(X + \varepsilon) \quad \text{.....(3.16)}$$

With Taylor's expansion,

$$X + \varepsilon = X^* + \varepsilon^* + \Delta t \cdot (f(X) + \varepsilon \cdot \frac{\partial f}{\partial X}) \quad \text{.....(3.17)}$$

Substituting (3.15), we obtain:

$$\frac{\varepsilon}{\varepsilon^*} = \frac{1}{1 - \Delta t \cdot \frac{\partial f}{\partial X}} \quad \text{.....(3.18)}$$

Hence a numerically stable solution will be obtained if

$$\left| \frac{\varepsilon}{\varepsilon^*} \right| < 1 \quad \text{.....(3.19)}$$

or, $$\left| 1 - \Delta t \cdot \frac{\partial f}{\partial X} \right| > 1 \quad \text{.....(3.20)}$$

which requires the following condition to be met:

$$\frac{\partial f}{\partial X} < 0 \quad \text{.....(3.21)}$$

Therefore, the Implicit Euler integration technique is unconditionally stable, as long as the partial derivative of f is negative, which is always met, as long as the equation is formulated properly.

One drawback of the Implicit Euler integration technique is that f(X) is evaluated at the current time t, which yields an implicit algebraic equation to solve for X.

For linear equations, the implicit integration can be applied easily, for example, consider the following differential equation:

$$\frac{dX}{dt} = -aX + b \quad \text{.....(3.21)}$$

Applying Implicit Euler technique:

$$X = X^* + \Delta t \cdot (-aX + b) \quad \text{.....(3.22)}$$

re-arranging:

$$X = \frac{X^* + b \cdot \Delta t}{1 + a \cdot \Delta t} \quad \text{.....(3.23)}$$

For non-linear equation, it is possible to eliminate the implicit character of the equation by linearizing the non-linear function f on the right hand side by Taylor's expansion.

For example, if the function f in (3.15) is non-linear

$$X = X^* + \Delta t \cdot f(X) \quad \text{.....(3.24)}$$

With Taylor's expansion:

$$X = X^* + \Delta t \cdot \left(f(X^*) + (X - X^*) \cdot \frac{\partial f}{\partial X}\Big|_{X^*}\right) \quad \text{.....(3.25)}$$

Let

$$a = -\frac{\partial f}{\partial X}\Big|_{X^*} \quad \text{.....(3.26)}$$

$$b = f(X^*) - X^* \cdot \frac{\partial f}{\partial X}\Big|_{X^*} \quad \text{.....(3.27)}$$

then (3.25) becomes:

$$X = X^* - a \cdot \Delta t \cdot X + \Delta t \cdot b \quad \text{.....(3.28)}$$

or,

$$X = \frac{X^* + b \cdot \Delta t}{1 + a \cdot \Delta t} \quad \text{.....(3.29)}$$

It can also be shown that the linearization of f(X) does not affect the stability conditions.

Let us apply the same reasoning as before:

$$X + \varepsilon = X^* + \varepsilon^* + \Delta t \cdot (f(X^* + \varepsilon^*) + (X + \varepsilon - X^* - \varepsilon^*) \cdot \frac{\partial f}{\partial X}) \quad \text{.....(3.30)}$$

After rearranging and substituting, we arrive at the same stability condition as (3.18):

$$\frac{\varepsilon}{\varepsilon^*} = \frac{1}{1 - \Delta t \cdot \frac{\partial f}{\partial X}} \quad \text{.....(3.31)}$$

### 3.4 System of Equations

Real simulation applications involve solving a large sets of ordinary differential equations and algebraic equations, which can be written in matrix form:

$$\frac{d\vec{X}}{dt} = \vec{F}(\vec{X}, t) + \vec{G} \quad \text{.....(3.32)}$$

Where:

$$\vec{X} = \begin{pmatrix} X_1 \\ \cdot \\ \cdot \\ X_n \end{pmatrix}$$

$$\vec{F}(\vec{X}, t) = \begin{pmatrix} F_{11} & \cdot & \cdot & F_{1n} \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & F_{ij} & \cdot \\ F_{m1} & \cdot & \cdot & F_{mn} \end{pmatrix} \begin{pmatrix} X_1 \\ \cdot \\ \cdot \\ X_n \end{pmatrix}$$

$$\vec{G} = \begin{pmatrix} G_1 \\ . \\ . \\ G_n \end{pmatrix}$$

The same techniques for Explicit Euler and Implicit Euler can be applied to the solution of the system of ODE's.

In the linear case, or in the non linear case after linearization, a typical equation can be written as:

$$\frac{dX_i}{dt} = \sum_{j=1}^{n} F_{ij} \cdot X_j + G_j \qquad (3.33)$$

or,

$$X_i = X_i^* + \Delta t \cdot F_{ii} \cdot X_i + (\sum_{\substack{j=1 \\ j \neq i}}^{n} F_{ij} \cdot X_j + G_i) \cdot \Delta t \qquad (3.34)$$

$X_i$ in the right side of equation (3.34) can be evaluated either at current time (Implicit Euler) or at past time (Explicit Euler), as shown before.

If $\Delta t \cdot \left| \frac{\partial F_{ij}}{\partial X} \right| < 2$ (3.35), that means the system is loosely coupled. Explicit Euler integration will give stable solution, and each equation can be solved individually.

However, if the system is strongly coupled, implicit Euler integration must be used. The equation (3.34) will become:

$$(1 - \Delta t \cdot F_{ii}) \cdot X_i - \Delta t \cdot (\sum_{\substack{j=1 \\ j \neq i}}^{n} F_{ij} \cdot X_j) = X^* + G_i \cdot \Delta t \qquad (3.36)$$

Therefore, the system to be solved can be written in matrix form:

$$\vec{A} \cdot \vec{X} = \vec{X}^* + \Delta t \cdot \vec{G} \qquad (3.37)$$

Where:

$$\vec{A} = \begin{pmatrix} 1 - \Delta t \cdot F_{11} & - \Delta t \cdot F_{12} & . & - \Delta t \cdot F_{1n} \\ - \Delta t \cdot F_{21} & 1 - \Delta t \cdot F_{22} & & \\ . & & 1 - \Delta t \cdot F_{ii} & \\ - \Delta t \cdot F_{m1} & & & - \Delta t \cdot F_{nn} \end{pmatrix}$$

The matrix solution to equation (3.37) becomes:

$$\vec{X} = \vec{A}^{-1} \cdot (\vec{X}^* + \Delta t \cdot \vec{G}) \quad \text{.....(3.38)}$$

It should noted that if the size of matrix A is large, and if it contains very large and very small values, (such as the case for solving mass and momentum conservation equations (pressure and flow have small time constants) and thermodynamics (temperature has large time constant)), the matrix A is called "stiff" matrix.

Finding the inverse of a "stiff" matrix A is computationally "expensive" and may not be obtained in real time usually PC platform. This advanced topic will be dealt with in more details in an advanced course.

## 3.5 Other Important Model Considerations

There are a number of important considerations in simulation modeling, and they are introduced in the followings:

### 3.5.1 Range of Validity

In contrast to most engineering applications which are designed to meet very specific design situations, dynamic simulation models have to be valid in the whole range of plant operations, for example, cold start, plant warmup, loading, full load, unloading, cooldown, shutdown. In addition, the models may have to cover also the abnormal or emergency situations resulting from operator actions or malfunctions. In some instance, some models function normally; in other instances, the model gives wrong results.

Consider an example of a heat exchanger between a one phase fluid and a two phase fluid, such as the heat exchange between reactor coolant and water/steam mixture in a nuclear steam generator, the heat balance on the primary side can be written as ( recall equation 2.8):

$$\frac{dT_o}{dt} = \frac{1}{M.C_v}\left[W.C_p(T_i - T_o) - UA.(\frac{(T_i + T_0}{2} - T_s)\right] \quad \text{.....(3.39)}$$

Where:

$T_i, T_o$ = inlet and outlet coolant temperatures respectively.

$T_s$ = the temperature of secondary mixture at saturation temperature

W = Coolant mass flow rate

$C_p$, $C_v$ = specific heats at constant pressure and at constant volume

M = mass of coolant

UA = overall heat transfer coefficient, multiplied by the average heat transfer area

For a constant temperature in the secondary, corresponding to a constant pressure, the steady state solution to (3.38) is given by

$$\frac{T_0 - T_s}{T_i - T_s} = \frac{2m-1}{2m+1} \quad \text{.....(3.40)}$$

Where

m is a dimensionless factor, defined by:

$$m = \frac{C_p.W}{UA} \quad \text{.....(3.41)}$$

This shows that for low flow, m could be small and less than 1/2. In such case, the right hand side of equation (3.39) is negative, which means the ratio of temperature differences is negative. This implies either T0 is less than Ts, or Ti is less than Ts. In either case, this is physically impossible.

Therefore the point illustrated here is that model should be validated under all possible operating conditions, and in the case of the above heat exchanger model, the use of another method such as NTUs ( number of transfer units) method, also known as Effectiveness method will avoid such problem, but the detailed discussion of this topic is beyond the scope of this introductory course.

(B) Realism

Realism is one of the key ingredients in real time simulation for effective training applications. It requires that the control room hardware and indications of the instrumentation look almost identical to those in real control room. The fidelity of the instrument indications requires that the results provided by the model must satisfy some performance criteria. For example, there is an ANSI Standard for Nuclear Plant Training Simulator (ANSI/ANS-3.5-1981), which stipulates that steady state computed values for the principal mass and energy balances and for critical parameters shall be within + or - 2%.

### 3.5.2 Reliability

The effectiveness of a simulator can be greatly reduced, if it often interrupted by a failure of the simulator, either in the hardware, or in the software model. An example of model failure is the case of division - a number divided by another number, which suddenly becomes zero, causing computer to "crash". To obtain a highly reliable software model, unconditional stability of the numerical techniques may have to used to solve ODE's, and the implicit algebraic equations. As well, model may

have to undergo "stress testing" to ensure proper behavior outside expected range covered by the model. For example, in addition to testing a boiler model under all operating conditions, it may have to be tested under extreme conditions such as all water inventory evaporated, and see whether the model can survive and still gives reasonable results.

### 3.5.3 Real time Simulation

As discussed above, all ODE's and algebraic equations representing power plant process, as well as Boolean equations representing control system logic are solved in time marching process. Here are some important points that should be presented before we understand the meaning of real time:

- When all ODE's have been integrated once with a fixed time step $\Delta t$ ; as well all other equations — algebraic and logic, are solved once, this process is then called one **ITERATION** of the model equations. Digital computer via a special "simulation executive" will execute the model equations iteration by iteration continuously in time, unless stopped by human intervention (in this case, we call it simulation "freeze").

- The time interval between the start of two consecutive model iterations is call "**Iteration Interval** ".

Usually the simulation executive allocates a fixed Iteration Interval for simulation, say $\Gamma$ sec.

If the current iteration is completed in $\delta$ sec, which is less than $\Gamma$ sec, the simulation executive will not start the next iteration until $\Gamma$ sec. has elapsed in real time.

Should the current iteration take longer than $\Gamma$ sec. to complete, i.e. $\delta > \Gamma$ then the simulation executive will wait until the current iteration is finished and will immediately execute the next iteration.

Now we can discuss the meaning of real time and other terms:

- **REAL TIME:**

1. If we choose an Iteration Interval such that $\Gamma = \Delta t$ (that is, the Iteration Interval is **equal** to the integration time step), *AND*

2. If $\delta \leq \Gamma$ (that is, model iteration time is less than or equal to Iteration Interval)

If these conditions are achieved, real time simulation is achieved.

- **FASTER THAN REAL TIME:**

1. If $\Gamma < \Delta t$ (that is Iteration Interval is **less than** the integration time step)

*AND*

2. If $\delta \leq \Gamma$ (that is, model iteration time is less than or equal to Iteration Interval)

- **SLOWER THAN REAL TIME:**

If $\Gamma > \Delta t$, regardless whether $\delta$ is less than or longer than $\Gamma$.

OR

Despite that fact that we set the Iteration Interval to be equal to the integration time step (i.e. $\Gamma = \Delta t$), but $\delta > \Gamma$.

It should be noted that there are limitations (such as numerical stability) that would limit the size of the integration time step to be used. Hence, once a certain time step is chosen, the Iteration Interval will be set to be equal to that time step size accordingly. This would then impose a upper bound for the time to complete one complete model iteration, in order to meet real time criterion. For an exceeding large model, and depending on the computer to be used, this is often a challenge.

## 4.1 Introduction to CASSIM™ Modeling

- The objective of this lesson is to introduce you to block oriented dynamic modeling using the CASSIM™ Simulation Development System.

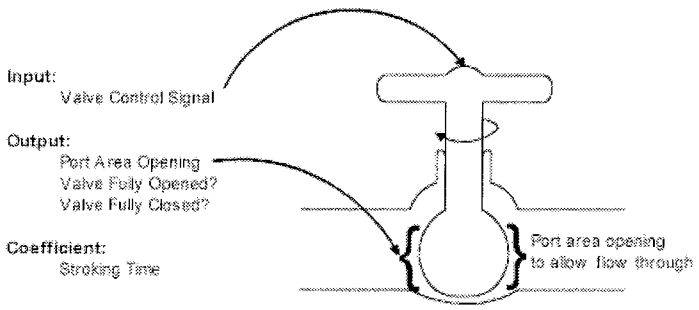
- You will be given the assignment of building a simulation model of a process system with two tanks. Detail instructions will be given to guide you step by step to build the simulation model for this system. When you are done, you will be able to run the simulated system, operate all the equipment and see the effects of such a system in action. Along the way, you will be using many of the powerful simulation modeling tools in CASSIM.

- One final note, the completed solution to the simulation model you are about to build is included to help you as a reference guide. It is installed in the directory called 2TANKS.SOL. Please feel free to reference this model solution in the CASSIM™ CD.

## 4.2 Block oriented Modeling Concept

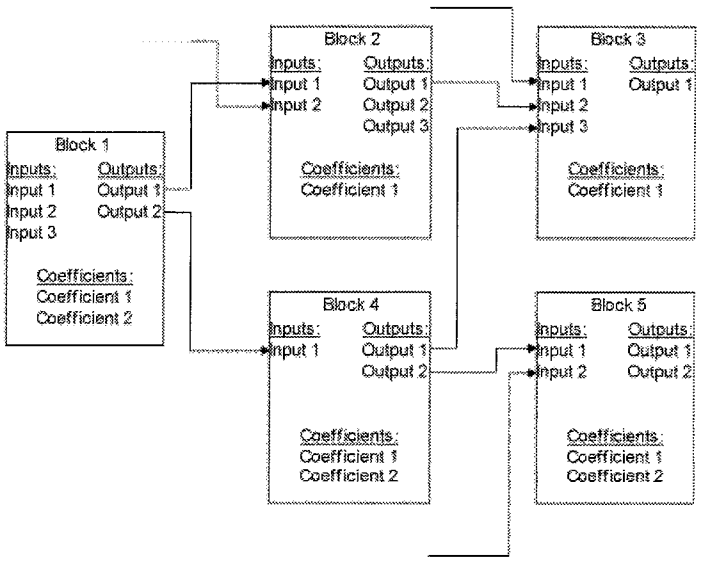
The block oriented modeling technique simulates a process system as a collection of small components connected to one another to perform as a system. A component may be a process, a logical unit, or a piece of equipment like a pump or a valve. Each component can be described by one or more attributes. Some attributes can be manipulated by external forces (e.g., moving the plug in a valve), some attributes uniquely characterize a specific component (e.g., the stroking time of a particular valve), and some attributes change over time as the component operate (e.g., the port area opening of the valve), taking into account the effects of the other attributes.

In simulation terms, a component is called a block. The attributes of a component are categorized as follows: attributes which can be manipulated by external forces are inputs to the block, attributes which uniquely characterize a specific component are called coefficients and attributes which change over time as the component operate are block outputs. An example of a component block is a valve:




Block Representation of a Valve Valve

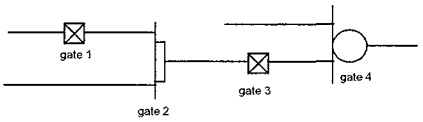
In block oriented modeling, a simulation model is described as a sequence of interconnecting blocks, where each block represents a component in the simulation. The output(s) of one block feeds into one or more subsequent block(s) downstream. When the simulation runs, each block is executed in sequence, producing output which is passed onto the next connected block(s). This next block in turn operates based on its inputs (and coefficients), and produce outputs for subsequent blocks connected to its outputs. When all the blocks in the model has executed once in sequence, the simulation has completed one iteration.

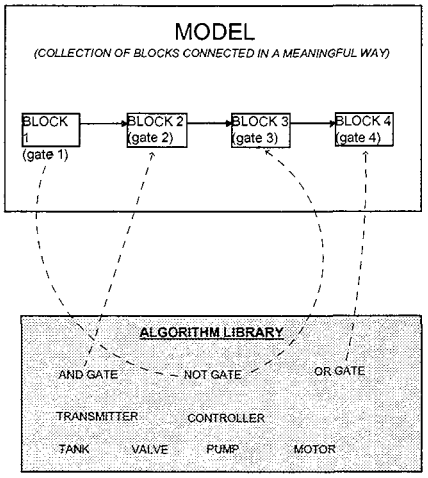



CASSIM™ models are real time dynamic models which are based on systematic application of first principles of mathematical modeling of plant process systems.

## LOGIC MODEL

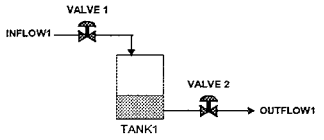
Given:

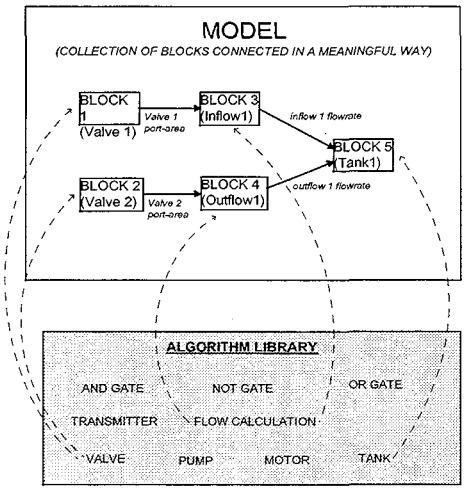


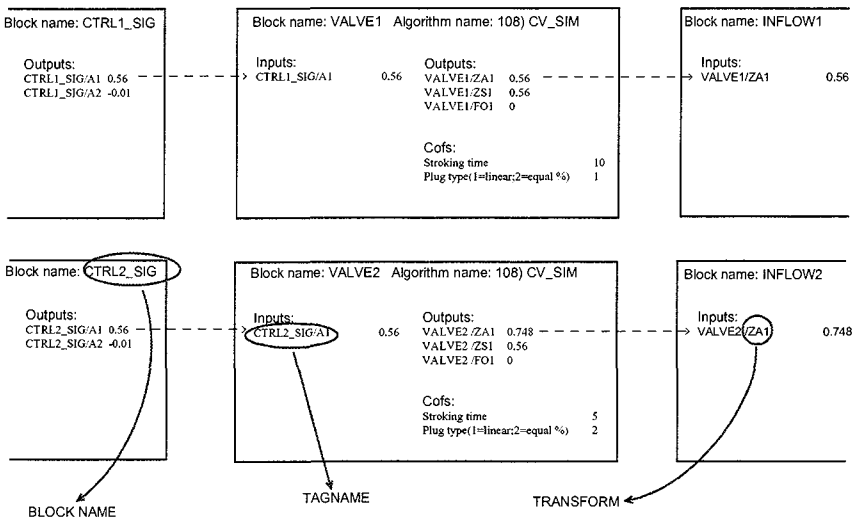

## PROCESS MODEL

Given:



***INPUTS, OUTPUTS & COEFFICIENTS***




Each generic component is called an "algorithm". Other special purpose algorithms can be added to a library of generic algorithms. In any block structured model, a "block" is a *customized* version of one of the algorithms from the library. For example, a customized valve block can be created if the generic valve algorithm is customized with a stroking time parameter of, say, 5 seconds, while another valve can be created using the same generic valve algorithm, but with different stroking time parameter. The stroking time parameter in the valve algorithm is called a "coefficient". The relationship between model, algorithm, and block is illustrated in Figures 3, 4; whereas the relationship of Inputs, Outputs and Coefficients is illustrated in Figure 5.

## Quick Quiz

1) What are the three categories of attributes of a block?

2) What attributes of one block connect to what attributes of other blocks downstream?

3) Define iteration.

## 4.3 CASSIM™ Simulation Development System

CASSIM™ is a simulation development system based on three common core technologies:

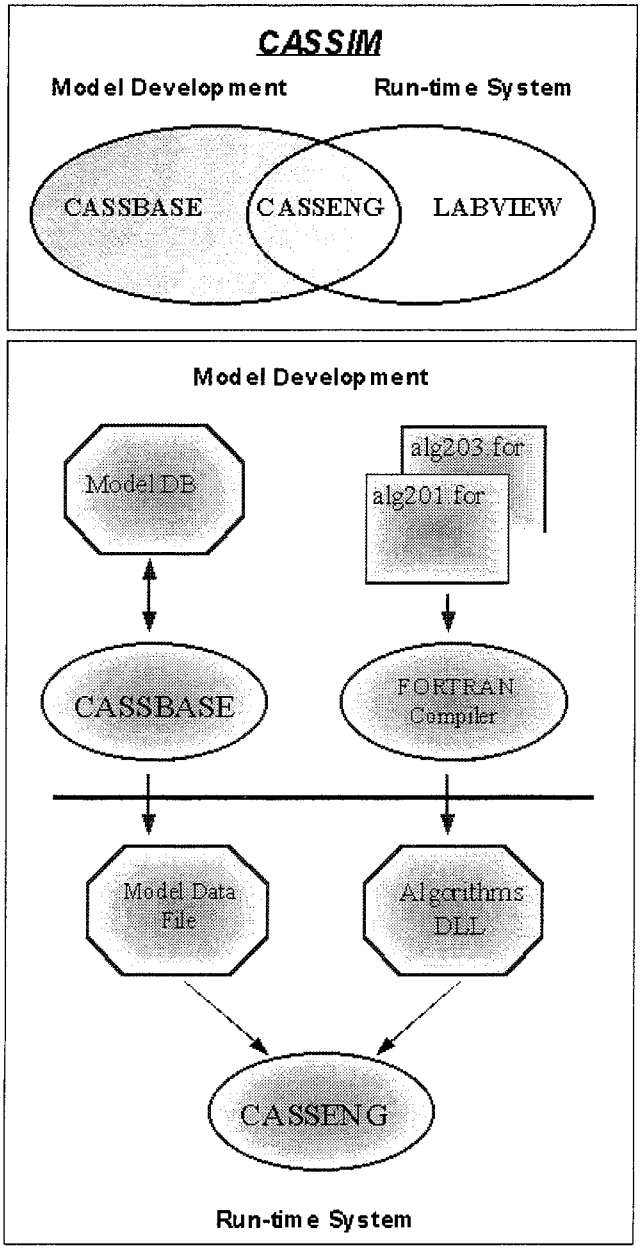
- CASSBASE: the block oriented simulation modeling tool.
- CASSENG: the simulation run time engine
- LABVIEW® for Windows: the graphical user interface from National Instruments.

The seamless integration of these three tools provides a powerful Microsoft Windows-based dynamic real time model development environment, and a user interface development environment for producing high fidelity, user-friendly simulation applications. The CASSIM development system is also capable of producing stand-alone simulation run time applications.

CASSBASE and CASSENG together form a robust model development environment. CASSBASE is the block oriented simulation modeling tool used for building, debugging and maintaining simulation models. CASSENG is the simulation run time engine. Models developed using CASSBASE are run in real time using CASSENG. CASSIM™ provides an on-line, interactive debugging facility which allows a developer to debug a simulation while it is running. Furthermore, the developer can dynamically modify simulation values on-line without recompiling the simulation code.

LABVIEW for Windows is the user interface development environment from National Instruments, with many value-added features provided by CTI for the purpose of developing user-friendly graphical interfaces for simulator applications. Graphical screens with buttons, iconic symbols, pop-up dialogs, etc. are developed using LabVIEW for Windows. User interface symbols, and special-purpose modules are provided in the form of LabVIEW libraries. CTI has custom LabVIEW libraries which provide power-plant simulation related symbols, modules for pop-up's, for performing Microsoft Windows Dynamic Data Exchange (DDE) and TCP/IP communication with CASSENG, and more. Through either DDE or TCP/IP, user interface screens request data from the simulation running in CASSENG for display and monitoring. Likewise, user inputs such as specific control actions, button presses, and setpoint changes are transmitted to the simulation running in CASSENG.

LabVIEW Application Builder for Windows is a LabVIEW for Windows add-on utility for compiling a collection of LabVIEW user interface screens into a run time executable. The resulting executable is a stand-alone application that runs without the LabVIEW development system. A run time simulation application would consist of the LabVIEW executable, and CASSENG, running the corresponding simulation model. The following diagram illustrates where each component of CASSIM fits into the simulation development life cycle:




In summary:

- CASSBASE is used for managing block oriented modeling, such as sequencing, and interconnecting blocks, and generating thermalhydraulic flow networks.

- CASSENG enhances the model development facility by providing the ability to run the simulation model dynamically while debugging the model in CASSBASE.

- LabVIEW is used for user interface development. The user interface screens communicate with the simulation model running in CASSENG to allow user interactions with the simulation. LabVIEW Application Builder is used to produce a stand-alone executable application consisting of the user interface screens.

The remainder of this Chapter is a series of step by step exercises which guides you through the process of building a simulation model for the two tanks system described earlier. You will be using CASSBASE and CASSENG to develop and run the simulation model. After successful testing, you will be running a LabVIEW runtime screen interact with the simulation.

### 4.4 Simulation Modeling using CASSBASE

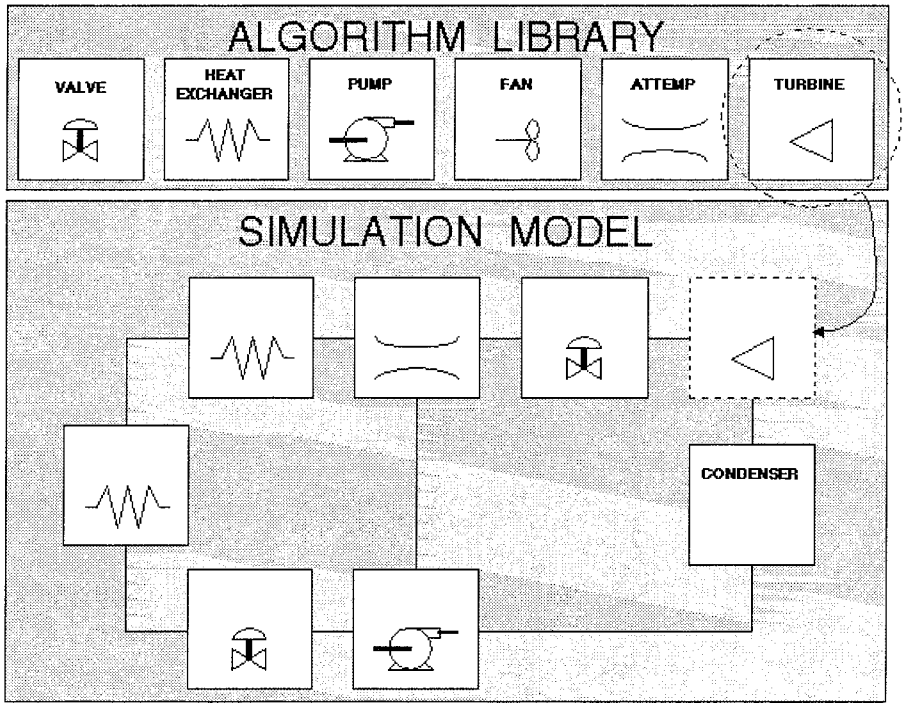
A CASSBASE model is composed of a sequence of interconnected blocks. Each block represents a component in the process or control system. For example, a valve is represented as a block in a simulation model of a plant.

In the same plant, there may be many other valves similar to it. From a simulation development standpoint, it would be cumbersome to model each and every individual valve separately in the plant when they all have similar characteristic. It would be more efficient to model a generic valve only once, with some configurable parameters (or attributes as described earlier), and store it as a template in a library. Subsequently, every valve created will be based on the generic template stored in the library, with the configurable parameters filled in to describe the particular instance of the valve being modeled.

An example of a configurable parameter for a valve is the stroking time. One valve may have a stroking time of five seconds while it may be one second for another valve.

In CASSBASE, each generic template is called an algorithm. CTI includes with the CASSIM product a library of over 200 generic algorithms. Other special purpose algorithms can be added to this library. In any model, each and every block defined is a customized version of one of the algorithms from the library.

The relationship between model, algorithm, and block is illustrated in the following diagram.




A CASSBASE simulation model consists of blocks which are customized versions of generic algorithms from the algorithm library. We are going to illustrate CASSIM™ based modeling using a simple two tank process. Before we start, we should first describe the process.

### 4.5 Opening the Two Tanks Model

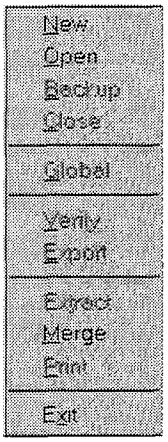
- Let's open up CASSBASE, the simulation modeling tool. In your Program Manager, double-click on the icon that looks like a baseball diamond.

- The first screen displayed is a product introduction panel. Click on the picture to make it disappear (or admire it quickly for the next five seconds after which it will disappear all by itself). The top menu bar of CASSBASE looks like this:


- Each menu item will open up a drop-down menu. The first three drop-down menu: Model, Algorithm, Block are used for managing models, algorithms, and blocks respectively. The rest of the drop-down menus are productivity tools to assist you in simulation modeling. The drop-down menu items will be presented in more detail later. You will also be using quite a few of these items to build the two tanks system.

- Let's open up a simulation model and begin building the two tanks system. A model has already been started for you. From the top menu

bar, select the menu item Model. The drop-down menu list looks like this:




- Select Open (hold down the left mouse button, drag the mouse pointer onto Open, and let go of the left mouse button). This will open up the file dialog for accessing a model.

- A model is a collection of files stored under a Windows directory. The name of the model is the directory name.

- The model you want to open now is called: C:\CASSDEMO\2TANKS

- Keep in mind that you will be building the 2TANKS model. A completed solution is also available in the model C:\CASSDEMO\2TANKS.SOL. You can reference the finished model in 2TANKS.SOL to help you out in case you are stuck.

- Using the directory list on the right hand side of the file dialog, position yourself into the C:\CASSDEMO\2TANKS directory. You should see a list of files ending with .dbf in the file list on the left hand side of the file dialog.

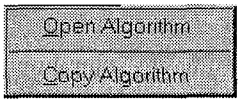
- Click on any one of these files, for instance, blk.dbf, and press the OK button in the top right hand corner of the file dialog.

- The file dialog will disappear. A few windows will flash by and you should see the message:

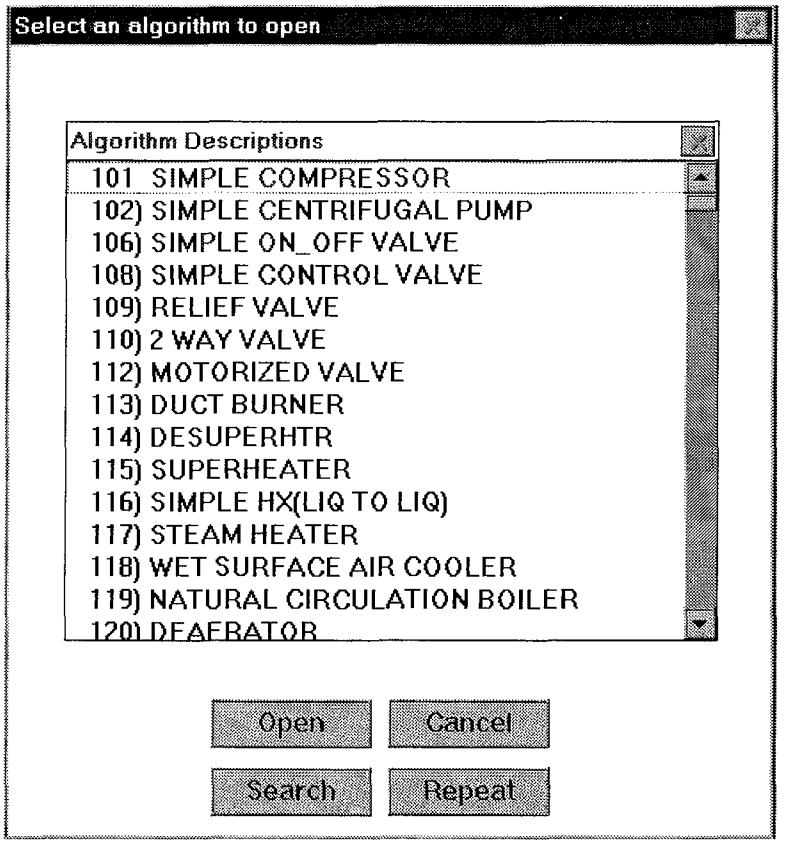
**Model C:\CASSDEMO\2TANKS opened.**

### 4.6 Browsing the Algorithms Library

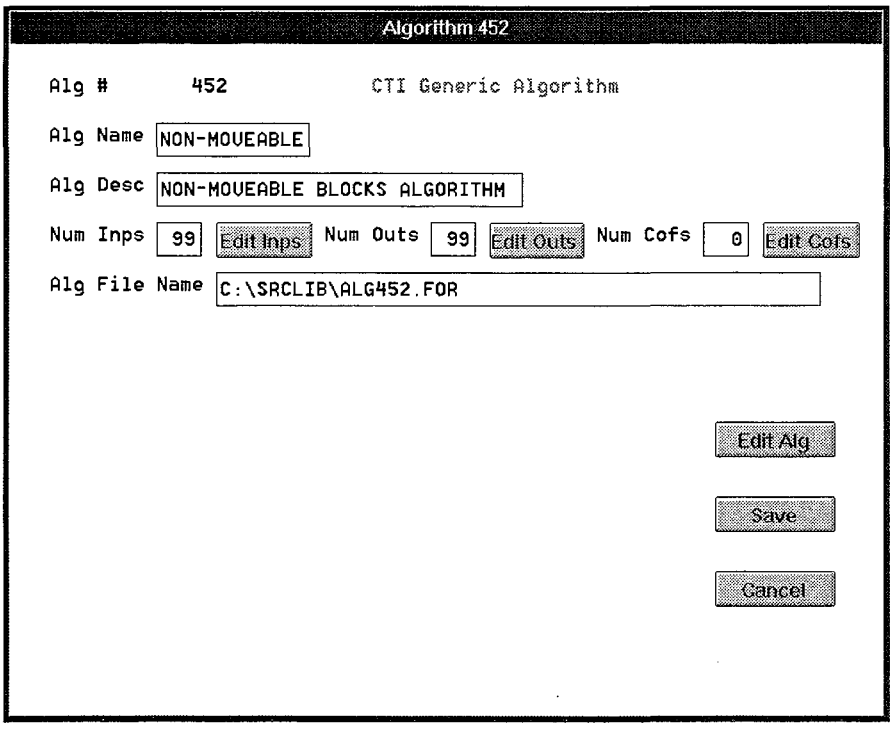
- The first thing to do when building a simulation model in CASSBASE is to look at the algorithm templates available in the algorithm library. It is from these templates that model blocks are created. From the top menu bar, select the menu item Algorithm. The drop-down menu list looks like this:




- Select Open Algorithm (hold down the left mouse button, drag the mouse pointer onto Open Algorithm and let go of the left mouse button). This will open up a list of algorithms. The dialog looks like the following:




- This list allows you to choose an algorithm template to look at. By using scroll bar on the right, click the "down arrow" until you see algorithm #452 in the Algorithm List Window. Let's look at algorithm #452 called NON-MOVEABLE BLOCKS ALGORITHM. To open this algorithm's template definition, click on this line with the mouse pointer. The line will be highlighted. Now move the mouse onto the Open button and click. This will open an algorithm template window like the following.




### 4.6.1 The NON-MOVEABLE BLOCKS Algorithm

This algorithm is a CTI Generic Algorithm that comes with the full CASSIM Simulation Development System. It is a very simple algorithm. Its purpose is to bring together outputs (signals) from specific blocks in the model to this one block for displaying the simulation values. You will see later on when debugging the simulation that this block is very useful for monitoring the running simulation values.

The algorithm is called non-moveable because once a block is defined based on this algorithm template; its location in the model should be fixed for convenient future access.

This simple algorithm works as follows: There are 99 inputs and 99 outputs. When the simulation runs, each value coming into the non-moveable block via an input is transferred unchanged to the corresponding output. Thus, if you keep you eyes on the 99 outputs while the simulation runs, you will see the values changing as the corresponding input values change.

#### (a) The Inputs

Click on the Edit Inps button to open the table for editing the input descriptions. In the demo version of CASSBASE, you can not change the contents of the table. In the first column (labeled PINNUM) are the input numbers. In the second column (labeled DESC) are the descriptions for each input. (When we developed this algorithm, there really wasn't much to say about each input. Note that we simply use the input number as the description). The last column is the data type of each input. The letter A means analog, which is the same for all the inputs of this algorithm. An

analog value is a real numeric value. The data type can be digital, in which case, the letter D is used. A digital value can have two possible values: TRUE or FALSE, represented by 1.0, and 0.0 respectively.

Let's close this table by pressing Close Inps.

**(b) The Outputs**

Next press Edit Outs to open the table for browsing the output descriptions. The columns are similar except there is one more column called TRANSFORM. This column is where you would give short names to the outputs. You will see later that these short names are used when connecting blocks. Press Close Outs to close the table.

There are no coefficients in a non-moveable block algorithm.

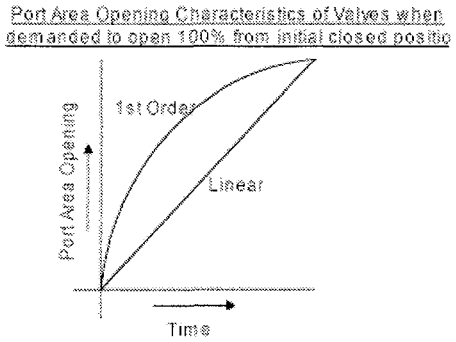
The Alg File Name field specifies where the FORTRAN source code for this algorithm is stored. Note: If the Algorithm is CTI's generic algorithm, source code file is not included. For this reason, if you click Edit Alg button, you will not be able to see the source file.

Press Cancel to close the algorithm template window.

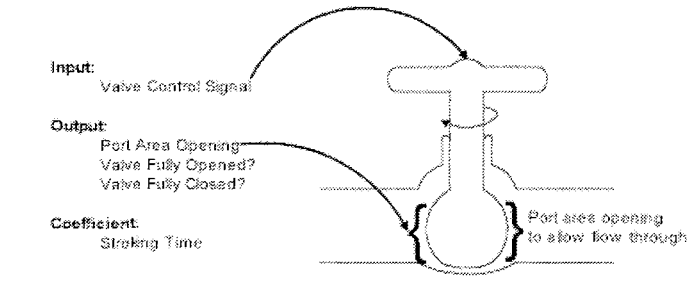
Let's move on to look at another algorithm. In the Two Tanks problem, you will be needing some valves. So let's look at a valve algorithm. Select Open Algorithm from the main menu. Select algorithm # 857) DEMO 1ST ORDER VALVE and click Open.

### 4.6.2 The DEMO 1ST ORDER VALVE Algorithm

This algorithm models a valve which operates based on a 1st order system.




Given an opening demand, the valve port area will open quickly and will slow down when the opening approaches the demand. This behavior is different than that simulated by algorithm #856 DEMO LINEAR VALVE. The linear valve algorithm simulates a valve whose port area opens linearly based on the demand signal.




Block Representation of a Valve Valve

The DEMO 1ST ORDER VALVE algorithm has one input, three outputs, and one coefficient.

- Press Edit Inps to open the input description table. The one input is Port Area Opening Demand. It is an analog value specified as a percentage (decimal value between 0 and 1) opening.

- Press Edit Outs to close the input description table and open the output description table. There are three outputs: Effective Port Area, Valve Fully Opened, Valve Fully Closed.

    1. Effective Port Area is the actual port area opening of the valve. Although the demanded port area opening may be a certain value, it takes some time before the valve drives to the desired opening. This output simulates over time the action of driving the valve.

    2. Valve Fully Opened is a digital value indicating whether the valve is fully opened. A 1.0 (or TRUE) means fully opened. A 0.0 (or FALSE) means not fully opened.

    3. Valve Fully Closed is a digital value indicating the opposite of the Valve Fully Opened output.

- Press Edit Cofs to close the output description and open the coefficient description table. There is only one coefficient: Stroking Time. This is use to set the stroking time characteristic of this particular valve. The stroking time is defined as the amount of time it takes to drive the valve from fully closed to 90% opened.

- Press Cancel to close the algorithm template.

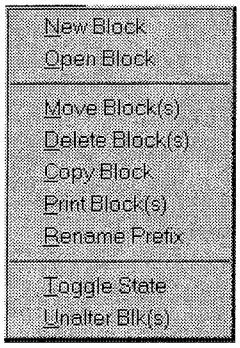
The CTI Generic algorithm #108 Simple Control Valve incorporates both of these valve port area opening types in addition to an Equal Percentage opening type. The selection of which valve type to simulate is done by specifying one of the coefficients in that algorithm. Feel free to examine this and other algorithms in the algorithm template library.

Let's have look at how the two algorithm templates: 1) 452 NON-MOVEABLE BLOCKS ALGORITHM, and 2) DEMO 1ST ORDER VALVE are used as blocks in a model.

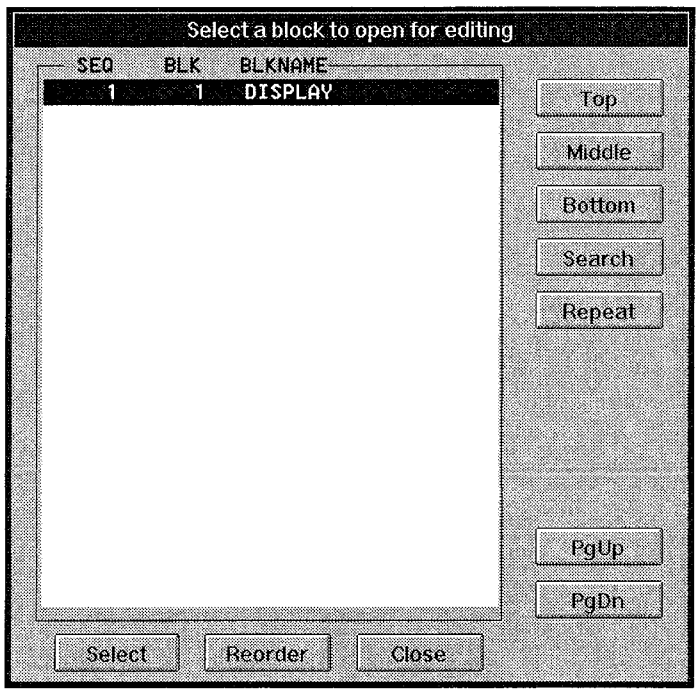
## 4.7 Opening a Block in the Model

One block of the two tanks system has been created for you.

- To open it and see its content, use the Block menu. From the top menu bar, select the menu item Block. The drop-down menu list appears as shown below:




- Select Open Block. This will open up a list of blocks defined in the model in a block selection window.
- The block selection window appears as shown below:



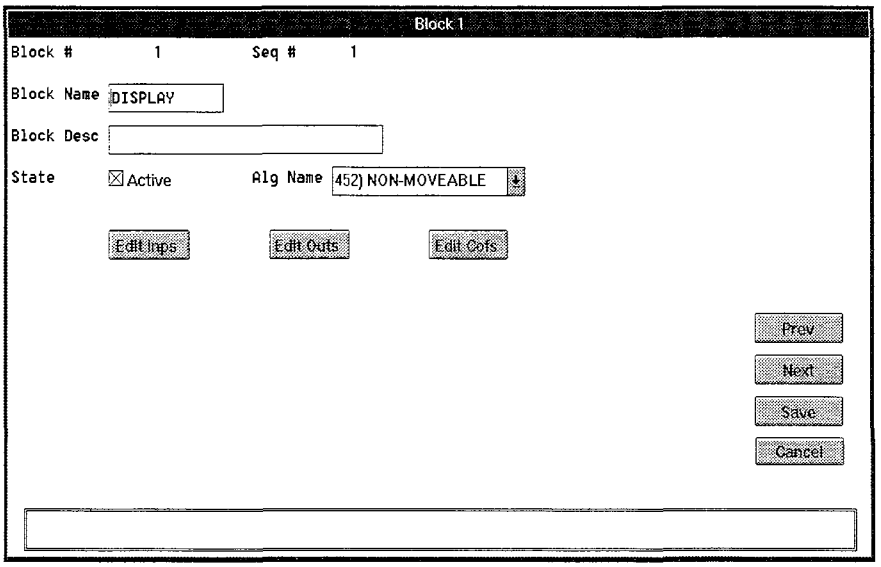

- This window is used for browsing the list of blocks in the model. Scroll up and down the list by pressing the PgUp and PgDn buttons. Use the Top, Middle, Bottom buttons to jump to the respective section of the model. The Search and Repeat buttons are used to locate to a block beginning with a specific string. We will not be needing the Search and Repeat functions.
- The blocks in the list are listed in execution sequence order (i.e., ordered by the numbers under the SEQ column). To change the display order, press the Reorder button to display the blocks in order by the

numbers under the BLK column. Usually, the sequence numbers and the block numbers are the same. After moving and deleting blocks in the model, the numbers in the two columns may no longer match. The simulation always execute in the order by sequence numbers.

- Since there is only one block in the model, the cursor is on this block. Press Select to open this block in a block edit window for browsing.

### 4.7.1 Browsing Block #1

Block 1 in the block edit window looks like the one below:



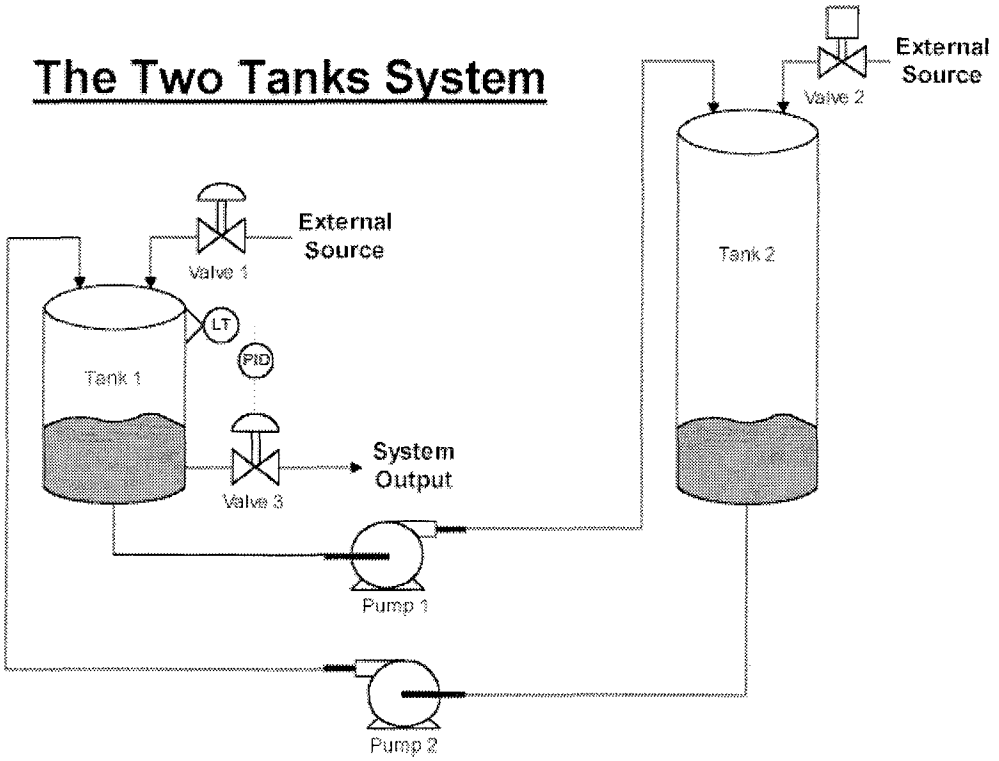

- The block number and the sequence number are both 1. The name of the block is DISPLAY. There is currently no block description text for this block. The block is active. Inactive blocks will not be executed by the simulation engine.
- The next four items concern the algorithm template which this block is based on. The box following the title Alg Name is used for selecting which algorithm template to base this block on.
- This block is currently based on algorithm #452 NON-MOVEABLE BLOCKS ALGORITHM. The block inherits the characteristics of algorithm 452 and thus has: 99 inputs, 99 outputs, and no coefficients.
- Press Edit Inps to see the input connection table. Many of the entries have been filled in. Use the scroll bar to the right of the table to scroll up and down to see the various connections. There are connections which connect to other blocks' outputs that currently do not exist in the model. You will be building them shortly.
- You can also move the highlighted cell within the table. Note that as the highlighted cell moves from one row to another, the gray area below the input table shows the algorithm description of the input that the highlighted cell is on. Recall that the algorithm input descriptions for the algorithm that this block is based on is just the input number.

- Press Edit Outs to open up the output table for browsing. Scroll up and down the table to see other outputs.
- Press the Cancel button to close this block. When prompted whether to save the changes or not, press No.
- You are now back at the block selection window. Press Close to close this window.

### 4.8 Two Tanks System Modeling Analysis and Design

The best approach to block oriented modeling is to first analyze the problem and design a block representation of the solution before using CASSBASE to build the model.

#### 4.8.1 Description of The Two Tanks System




- The process system to be simulated has two tanks, each receiving liquid from external sources. The objective is to combine these two sources of liquid and deliver the liquid through the output of the smaller tank.
- In reality, the flow from the external sources are available only intermittently. Therefore, the tanks are used as reservoirs to hold liquid from these sources for use in the system regardless of whether there is flow or not from the external sources.
- For the purpose of simulation, the flows from the external sources are available at all times. This facilitates testing of the rest of the system. A possible future enhancement to the simulation would be to simulate the pattern of availability of liquid from external sources.

- A line is provided for pumping liquid from the larger tank to the smaller tank. Likewise, a similar line is available for transferring liquid from the smaller tank to the larger tank. This is to facilitate the storing of as much reserve liquid as possible utilizing the larger tank to meet the demands of when the external sources are not available.
- The delivery of the liquid out of the smaller tank is controlled by a level controller monitoring the tank level. The level controller adjusts the valve on the outlet pipe to let liquid out so as to maintain tank level at or below the controller set point.
- The following are some technical specifications of the system described above. These specifications will be referenced later as you build the simulation model of the system.

| **Tanks** | **Tank 1** | **Tank2** |
|---|---|---|
| Tank height | 10 m | 20 m |
| Tank Diameter | 0.5 m | 0.5 m |
| Height of outlet pipe from bottom of tank | 0.5 m | 0.5 m |

| **Pumps** | **Pump 1** | **Pump 2** |
|---|---|---|
| Maximum Flow Capacity | 5 Kg/s | 15 Kg/s |
| Run Up/Rundown Time | 5 Seconds | 5 seconds |

| **Valves** | **Valve 1** | **Valve 2** | **Valve 3** |
|---|---|---|---|
| Valve Type | Linear Plug Valve | Linear Plug Valve | |
| Valve Control | Port area opening is proportional to control signal (0 - 100 %) | either ON or OFF | Port area opening is proportional to control signal (0 - 100 %) |
| Port Area Type | First order system | Constant speed motorized valve | First order system |
| Stroking Time | 5 seconds | 5 seconds | 5 seconds |

| Physical Properties | | | |
|---|---|---|---|
| Liquid density | | 1000 $Kg/m^3$ | |
| **Flow Rates** | Flow into Valve 1 | Flow into Valve 2 | Flow into Valve 3 |
| Flow Rate | 5 Kg/s | 10 Kg/s | 20 Kg/s |

*Control Specifications*

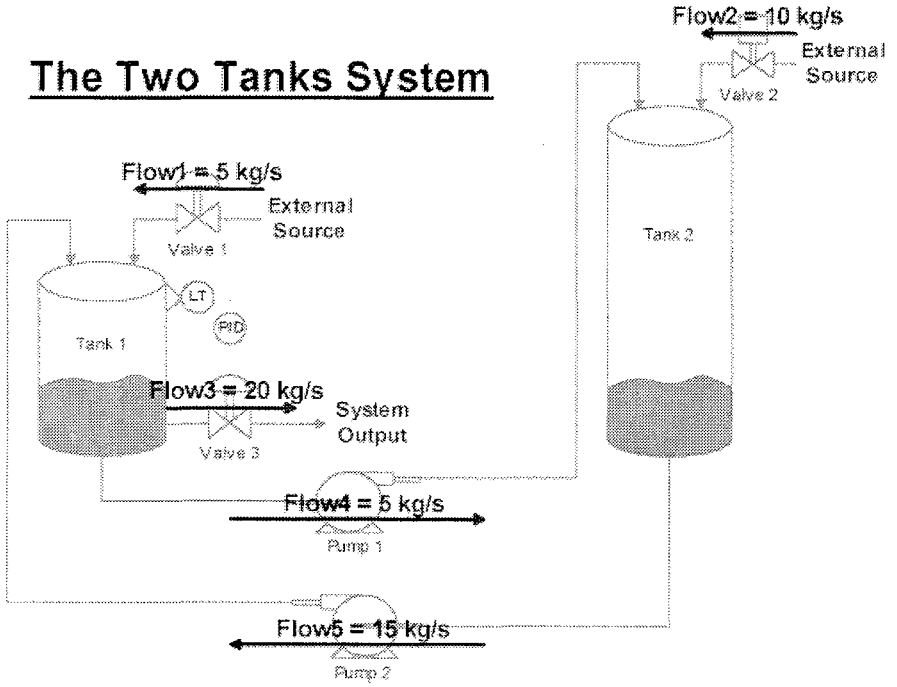
- Valve 1's port area restricts the flow through the cross-section area in the pipe and can be adjusted by controlling the stem position of the valve to the desired position of opening. This control action is achieved by manual control.
- Valve 2 is an on-off valve and can be opened or closed under manual control.
- A simple PID loop is used to control the outflow from tank 1 through valve 3 by opening or closing valve 3 to maintain a tank level of maximum 8 m. The PID loop can be put into manual mode to allow for manual adjustment of the valve 3 port area opening.
- Both pumps 1 & 2 can be started or stopped under manual control.

### 4.8.2 Algorithms Selection

Based on the P&ID diagram plus the data given in the description of the two tanks system, most of the components can readily be modeled as blocks.

| Equipment | Represented by Algorithm |
|---|---|
| Pumps | 855 |
| Valves | 856 or 857 |
| Tanks | 858 |
| PID Level Controllers | 859 |

One concept to emphasize is that flows are modeled as blocks although they are not usually drawn on the P&ID diagram. A revised diagram showing the flows is given below:




### 4.8.3 Initial Conditions

- Tank 1 initial level = 5 m.
- Tank 2 initially empty.
- Valve 1, 2, 3 closed.
- Pump 1 & 2 stopped.

### 4.8.4 Flow Calculation Method

In this lesson, a flow is represented as a block using algorithm #860. In fact, it should be computed as proportional to the square root of the pressure drop between two points — part of the Hydraulic Network Solution schema provided by CASSIM™. Since it will involve details beyond the scope of this lesson, we are going to take a simpler approach here. However, future Lesson will involve detailed Hydraulic Network Solutions.

Therefore, in this lesson here, each flow block represents a flow path which can carry a specified maximum flow rate. The same flow block also allows the inclusion of one valve and one pump on the flow path to affect the effective flow rate through the path. As depicted above, there are five flow paths in the two tanks system.

Algorithm #860 calculates the effective flow in a path using the following equation:

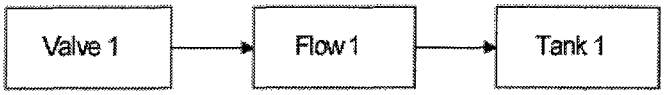
**Effective Flow = Maximum Flow * Valve Opening * Pump Power**

where

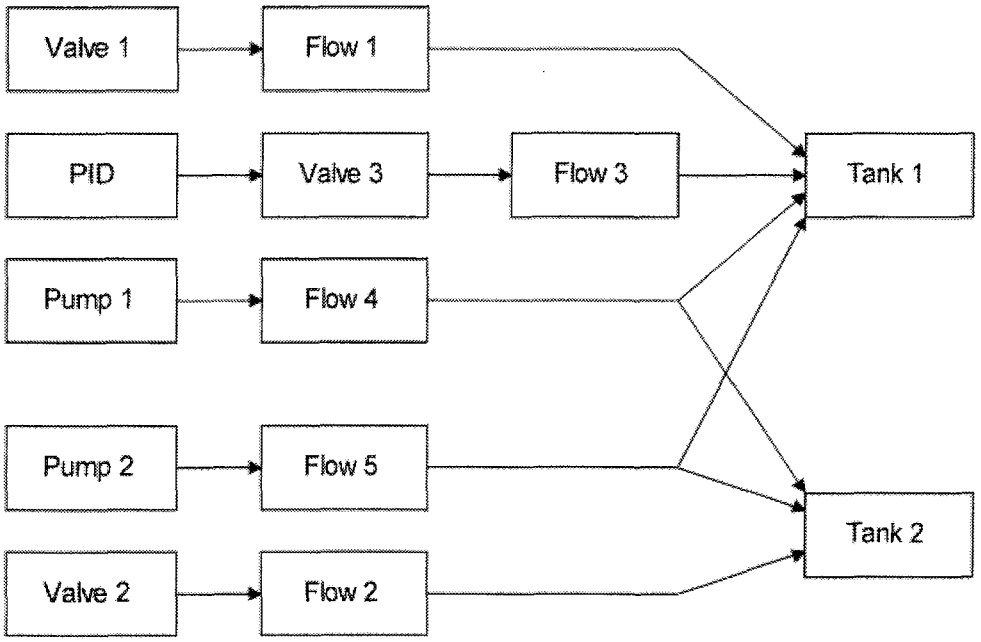
- Maximum Flow is in kg/s

- Valve Opening is % port area opening of the valve (in the range 0.0 to 1.0)
- Pump Power is % of full power (in the range 0.0 to 1.0)

Suppose we use algorithm #860 in a flow block to represent flow 1, the flow from an external source through valve 1 into tank 1. An input into this flow block is valve 1's port area opening. The effective flow rate output of the flow block is input to tank 1. Depicted as a block diagram illustrating data flow:




In fact, if we look at the entire two tanks system, a suitable block diagram illustrating data flow could be:



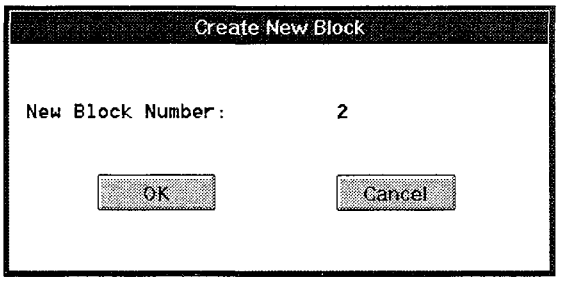

You can follow this block diagram design to build a CASSBASE model representing the two tanks system.

Let's begin by creating a new block to represent valve 1.

## 4.9 Two Tank System Modeling

### 4.9.1 Creating Block #2 — Valve 1

- From the top menu bar, select the menu item Block. From the drop-down menu list, select New Block. You will get a dialog like the following:




Press OK to create the new block.

- You are now looking at a new block in a window similar to the one you used when browsing block 1. The difference is, this one is empty. You will fill in the block entries, select an algorithm to base this block on, and fill in the inputs, outputs, and coefficients.

- In the Block Name entry box, type in VALVE1 and press Enter.

- In the Block Desc entry box, type in a description describing the purpose of this block. For example "VALVE #1". After you have typed in a description, press Enter to go to the next item.

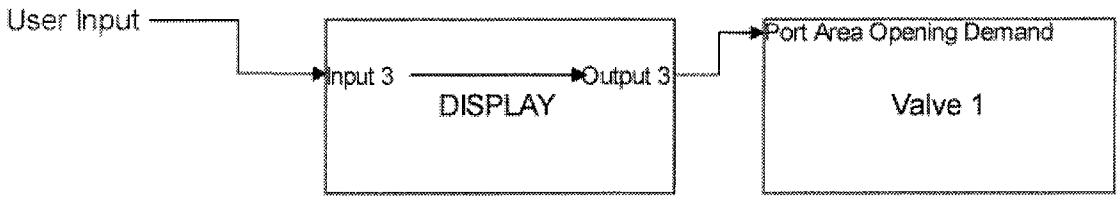
- The next item is the Active item. This specifies whether or not this block is executed when the simulation runs. By default, it is marked with an "X" to indicate active. You can toggle the active/inactive selection by clicking in the box.

- The next item is the algorithm selection for this block. By default, the first algorithm in the list is selected. For this block, you want the block to be based on algorithm #857 DEMO 1ST ORDER VALVE. To select algorithm #857, click on the downward pointing arrow to drop-down a list of algorithms. Scroll down and select the entry 857) DEMO_VALVE.

- Press Edit Inps to open the input table for editing. There is only one input for this block, the port area opening demand.

- For this simulation, the input comes from the user interface, where an operator would control the valve by typing in a port area opening demand. This demand value will be passed to the running simulation.

- The block where the input is passed to is the DISPLAY block. If you had noticed before, input #3 of the DISPLAY block was reserved for taking the port area opening demand for valve 1. Knowing that the DISPLAY block transfers the value coming into input #3 to output #3, VALVE1 block's port area opening demand input should be connected to output #3 of the DISPLAY block.




- In general, an input can be specified in two different ways:

  1) input can be a constant value, or

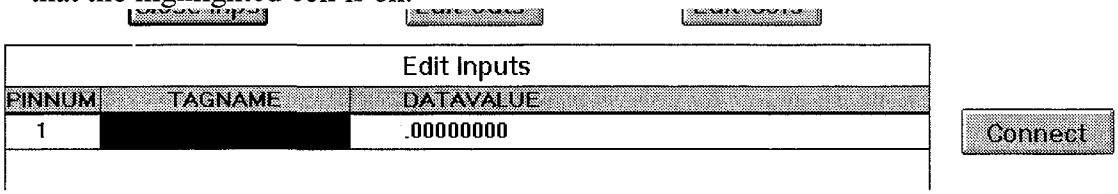
  2) input can come from the output of another block.

- To specify a constant value, enter a constant tag name in the TAGNAME column, and enter the constant value in the DATAVALUE column. A constant tag denotes that the value for input is stored in the DATAVALUE column. A constant tag name begins with a '$' and can be up to 17 characters consisting of letters, numbers, and underscore. Examples of constant tag names are: $VALVE_ON, $PUMP_OFF, etc.

- To specify an input from the output of another block, enter in the TAGNAME column the output tag name to connect to. An output tag name consists of the output block's name followed by a '/', followed by the transform of the particular output of the block to connect to. Recall that transforms are names associated with the outputs of an algorithm; one unique transform per output.

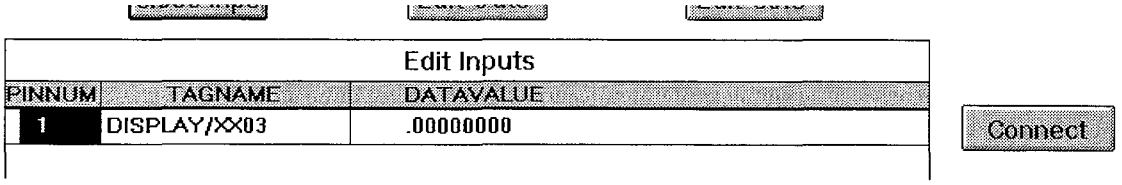
- Instead of typing in the output tag name to connect to, use the Connect button to the right of the input table. For the VALVE1 block's port area opening demand (input 1), move the highlighted cell inside the input table to the cell to the TAGNAME column corresponding to the row with pinnum = 1.

- Note that as the highlighted cell moves within the table, the gray area below the input edit table show the algorithm description of the input that the highlighted cell is on.

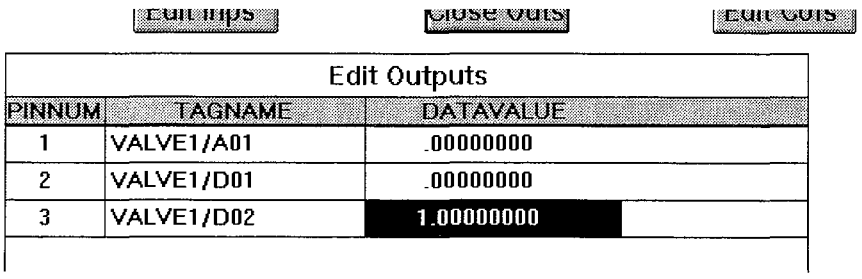



- Click the Connect button to the right of the input table. A block selection window opens to allow for selection of the block to connect to. Select block #1, the DISPLAY block and click the Select button. This opens up a Select Transform dialog listing the output transforms for the DISPLAY block. Select output #3 (XX03) and click the Select button.

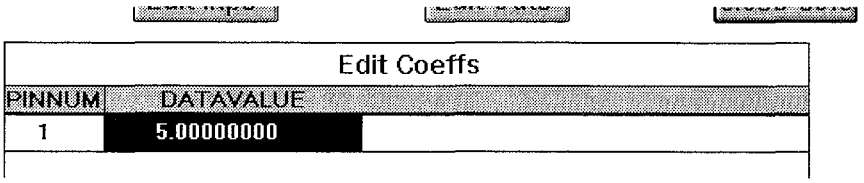
- Because this input gets its value from another block's output, there is no need to enter any value into the DATAVALUE column.




- Press Edit Outs to open the output table. Values can be entered into the DATAVALUE column to initialize the outputs produced by this block before the simulation begins to run. In this case, the valve is initially closed, therefore you have to configure the VALVE FULLY CLOSED output to be TRUE (1.0).




- Press Edit Cofs to open the coefficient table. Position the highlighted cell to enter the DATAVALUE for VALVE1's Stroking Time. According to the specification given for VALVE1, the stroking time is 5 seconds.




- Press Close Cofs to close the coefficient table.

- Press Save to save the editing you have done.

- You are now presented with the Create New Block dialog. Press OK to create block #3.

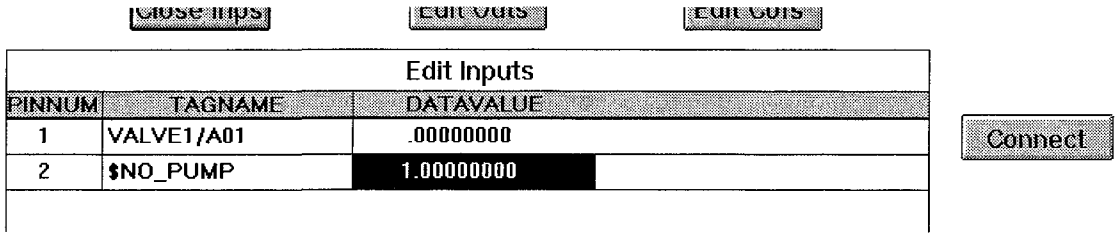
### 4.9.2 Creating Block #3 — FLOW1

- The output of the VALVE1 block is the valve's port area opening and it is used in the calculation of the flow path called FLOW1. Make block #3 this flow calculation. Enter FLOW1 as the block name. Enter a block description. The block should be active. The algorithm this block is based on is algorithm #860 DEMO FLOW CALCULATION.

- Press Edit Inps to open the input table. There are two inputs.

- For input #1, the port area opening of the valve on the flow path, the input comes from output #1 of the VALVE1 block. Position the highlighted cell on the TAGNAME column for output #1. Press the Connect button to the right of the input table. Select the VALVE1 block from the block selection window. Select the Effective Port Area output (A01) transform.

- For input #2, the pump power of the pump on the flow path, there is no pump on this flow path. To make the flow calculation work, enter a constant value of 1.0 for this input. To do this, enter a constant tag name for input #2 in the TAGNAME column. Type in $NO_PUMP. Also enter 1.0 into the DATAVALUE column for input #2.




- Press Edit Outs to open the output table. Initially, this calculation should show no flow. Therefore, leave the ACTUAL FLOW output's DATAVALUE as 0.0.

- Press Edit Cofs to open the coefficient table. According to the specifications given, the flow from an external source coming into VALVE1 is 5 kg/s. This means that if VALVE1 is completely opened, the maximum flow in the flow path is 5 kg/s. Enter 5.0 in the DATAVALUE column for coefficient #1.

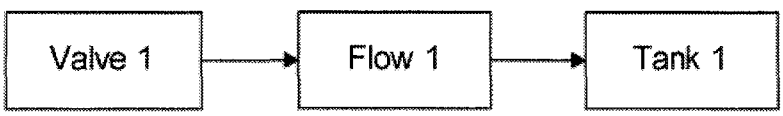
- Press Close Cofs to close the coefficient table.

- Press Save to save the editing you have done.

- You are now presented with the Create New Block dialog. Press OK to create block #4.

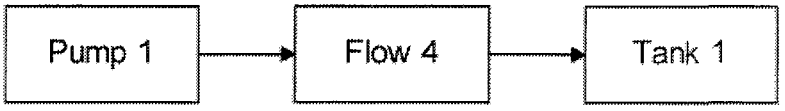
### 4.9.3 Creating Block #4 — Tank1

- You have just finished created the first two blocks in the following flow path:



- If you refer to the block diagram earlier, you will notice that before we can create a block for tank 1, there are other flow paths that are input to tank 1.

- Let's chose another flow path to model.




- Make block #4 represent pump 1. In the empty block edit window, type in PUMP1 as the block name. Enter a block description. The block should be active. The algorithm that this block is based on is algorithm #855 DEMO PUMP.

- Press Edit Inps to open the input table.

- There is only one input to PUMP1, either request it to turn on, or request it to turn off. This input comes from the user interface and is transferred through input #1 and output #1 of the DISPLAY block. Connect this input to output #1 of the display block. Press Connect, select block #1, the DISPLAY block, and select output #1 (XX01) transform.

- Press Edit Outs to open the output table.

- There are two outputs. By default, PUMP1 is not energized (i.e., no power) and is off. Therefore, the default DATAVALUES of 0.0 for both output is fine.

- Press Edit Cofs to open the coefficient table.

- There is only one coefficient for PUMP1: the pump run up time. Once the pump is turned on, it takes some time for it to run up to full power. The pump run up time is reflected in the pump power output. According to the specifications given, PUMP1 runs up in 5 seconds. Enter 5.0 in the DATAVALUE column.

- Press Close Cofs to close the coefficient table.

- Press Save to save the editing you have done.

- You are now presented with the Create New Block dialog. Press OK to create block #5.

### 4.9.4 Creating Block #5— FLOW4

- The pump power output of the PUMP1 block is used in the calculation of the flow path called FLOW4. Make block #5 this flow calculation. Enter FLOW4 as the block name. Enter a block description. The block should be active. The algorithm this block is based on is algorithm #860 DEMO FLOW CALCULATION.

- Press Edit Inps to open the input table. There are two inputs.

- For input #1, the port area opening of the valve on the flow path, there is no valve on this flow path. To make the flow calculation work, enter a constant value of 1.0 for this input. To do this, enter a constant tag name for input #2 in the TAGNAME column. Type in $NO_VALVE. Also enter 1.0 into the DATAVALUE column for input #1.

- For input #2, the pump power of the pump on the flow path, the input comes from output #1 of the PUMP1 block. Position the highlighted cell on the TAGNAME column for output #1. Press the Connect button to the right of the input table. Select the PUMP1 block from the block selection window. Select the Pump Power output (A01) transform.

- Press Edit Outs to open the output table. Initially, this calculation should show no flow. Therefore, leave the ACTUAL FLOW output's DATAVALUE as 0.0.

- Press Edit Cofs to open the coefficient table. According to the specifications given, the flow capacity of PUMP1 is 5 kg/s. This means that if PUMP1 is running at full power, the maximum flow in the flow path is 5 kg/s. Enter 5.0 in the DATAVALUE column for coefficient #1.

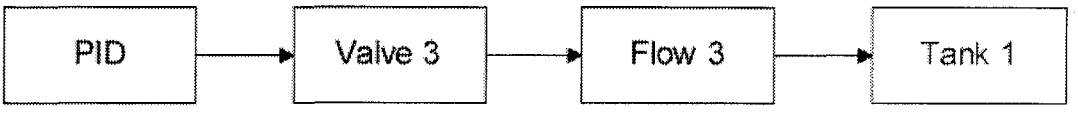
- Press Close Cofs to close the coefficient table.

- Press Save to save the editing you have done.

- You are now presented with the Create New Block dialog. Press OK to create block #6.

### 4.9.5 Creating Block #6 — PID Controller

- Let's choose another flow path that leads into tank 1 to model.




- Make block #6 the PID level controller. A PID controller has four inputs:

1. Controller setpoint
2. Present value of the process variable being controlled
3. AUTO/MANUAL mode request
4. Manual control signal

- In AUTO mode, the controller automatically adjusts its output control signal to drive the process variable (level) being controlled to maintain its value at the controller's setpoint. This is done by comparing the current value of level vs. the controller setpoint and adjust output control signal accordingly.

- In MANUAL mode, the manually entered control signal is directly used as the controller's output control signal to drive the process variable.

- In the two tanks system, the PID controller is used as a tank level controller to maintain the level of the liquid in tank 1 such that it does not exceed a specific height. The specified height is used as the controller setpoint. The PID level controller's output control signal drives valve #3 which will open to relief some liquid inside tank 1 if the level rises above the height setpoint.

- In the empty block edit window, enter PID as the block name, Enter a block description. The block should be active. The algorithm this block is based on is algorithm #859 DEMO PID CONTROLLER.

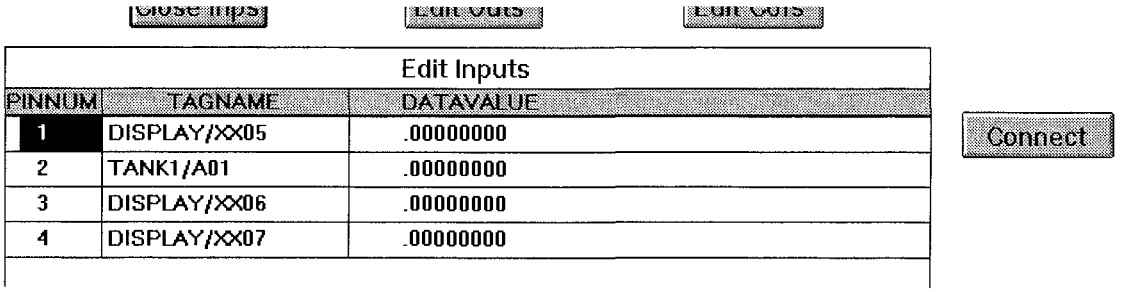
- Press Edit Inps to open the input table.

- The controller setpoint input (#1), AUTO/MANUAL mode request input (#3), and manual control signal input (#4) all come from the user interface and goes through input/output #5, #6, and #7 of the DISPLAY block respectively.

- Input #2, the present value of the process variable being controlled, is the tank level output of tank 1. The Connect button can not be used for specifying this connection because the block representing tank 1 has not been defined yet. However, you can still specify the connection if you know what the tag name of tank 1's level output is. Since you did the analysis and design, a likely name for the tank 1 block is TANK1. And knowing that the transform for the tank level output is A01, you can enter into the input table's TAGNAME column TANK1/A01 for input #2.

- Another approach to handling this situation of having to connect to a block not yet defined is to enter in an arbitrary tag name and later use other tools to identify the arbitrary connections and correct it.

- The input table should be as shown below:



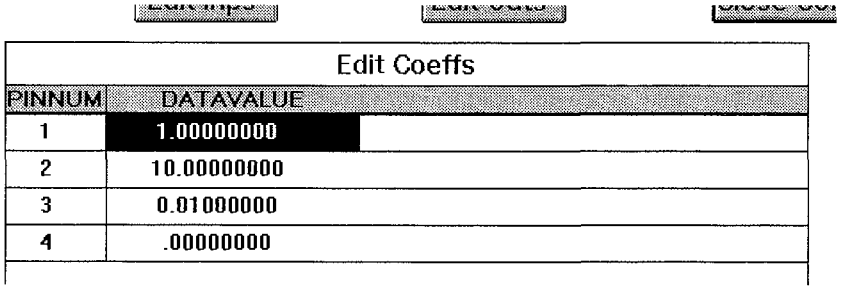
Edit Inputs

| PINNUM | TAGNAME | DATAVALUE |
|---|---|---|
| 1 | DISPLAY/XX05 | .00000000 |
| 2 | TANK1/A01 | .00000000 |
| 3 | DISPLAY/XX06 | .00000000 |
| 4 | DISPLAY/XX07 | .00000000 |

Connect

- Press Edit Outs to open the output table. The default values of 0.0 for all of these outputs are just fine. Therefore, leave them as they are.

- Press Edit Cofs to open the coefficient table. There are four coefficients. These are values for tuning the controller so that it makes its automatic adjustments smoothly. Enter the values as shown below:



Edit Coeffs

| PINNUM | DATAVALUE |
|---|---|
| 1 | 1.00000000 |
| 2 | 10.00000000 |
| 3 | 0.01000000 |
| 4 | .00000000 |

- Press Close Cofs to close the coefficient table.

- Press Save to save the editing you have done.

- You are now presented with the Create New Block dialog. Press OK to create block #7.

### 4.9.6 Creating Block #7 — VALVE3

- The output control signal from the PID level controller controls valve 3. Make VALVE3 the name of block #7. Enter in a block description. The block should be active. The algorithm that this block is based on is algorithm #857 DEMO 1ST ORDER VALVE.

- Press Edit Inps to open the input table.

- VALVE 3's port area opening demand signal comes from the output control signal of the PID block. Press Connect and make the appropriate selections to form the tag name PID/A01 for input 1.

- Press Edit Outs to open the output table. VALVE3 is initially closed, therefore you have to configure the VALVE FULLY CLOSED output to be TRUE (1.0). Leave the other output DATAVALUES.

- Press Edit Cofs to open the coefficient table. According to the specification given for VALVE3, the stroking time is 5 seconds. Enter this into the DATAVALUE column.

- Press Close Cofs to close the coefficient table.

- Press Save to save the editing you have done.

- You are now presented with the Create New Block dialog. Press OK to create block #8.

### 4.9.7 Creating Block #8 — FLOW3

- Block #8 is the block representing FLOW3.

- Enter FLOW3 as the block name. Enter a block description. The block should be active. The algorithm this block is based on is algorithm #860 DEMO FLOW CALCULATION.

- The inputs are: VALVE3/A01 for input 1, the port area opening of the valve on the flow path, and $NO_PUMP with a DATAVALUE of 1.0 for input 2, the power of the pump on the flow path.

- The output is just fine. Press Edit Outs to check the table and press Close Outs to close it.

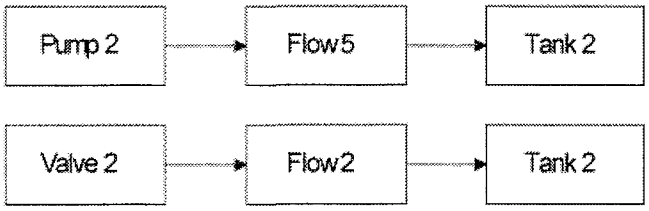
- The stroking time coefficient according to the given specification is 5 seconds.

- Press Save to save the editing you have done.

- You are now presented with the Create New Block dialog. Press OK to create block #9.

### 4.9.8 Creating Block #9, 10, 11, and 12

- You create the blocks #9, 10, 11, and 12 to model the flow paths shown below:




- Make block #9 VALVE2, block #10 FLOW2, block #11 PUMP2, block #12 FLOW5. These blocks are very similar to blocks #2, 3, 4, and 5 respectively. Use the specifications given earlier.

- Note that VALVE2 is a LINEAR valve. Choose algorithm #856 DEMO LINEAR VALVE. The inputs, outputs, and coefficients are the same as a 1ST ORDER valve. The difference is the behavior of the effective port area opening.
- For each new block, make sure you press the buttons: Edit Inps, Edit Outs, and Edit Cofs at least once. CASSBASE checks this to try and ensure you accept the contents of the tables before saving.
- The following tables list the values you should have in the blocks, use these tables as reference in the unlikely event that you get stuck.

| **Block #9** | **Algorithm** | **Pin** | **Data Value or Connection** |
|---|---|---|---|
| VALVE2 | Alg #856 DEMO_VALVE1 | Input 1 | DISPLAY/XX04 |
| | | Output 1 | 0.0 |
| | | Output 2 | 0.0 |
| | | Output 3 | 1.0 |
| | | Coefficient 1 | 5.0 |

| **Block #10** | **Algorithm** | **Pin** | **Data Value or Connection** |
|---|---|---|---|
| FLOW2 | Alg #860 FLOW_CALC | Input 1 | VALVE2/A01 |
| | | Input 2 | $NO_PUMP, 1.0 |
| | | Output 1 | 0.0 |
| | | Coefficient 1 | 10.0 |

| **Block #11** | **Algorithm** | **Pin** | **Data Value or Connection** |
|---|---|---|---|
| PUMP2 | Alg #855 DEMO_PUMP | Input 1 | DISPLAY/XX02 |
| | | Output 1 | 0.0 |
| | | Output 2 | 1.0 |
| | | Coefficient 1 | 5.0 |

| **Block #12** | **Algorithm** | **Pin** | **Data Value Connection** |
|---|---|---|---|
| FLOW5 | Alg #860 FLOW_CALC | Input 1 | $NO_VALVE, 1.0 |
| | | Input 2 | PUMP2/A01 |
| | | Output 1 | 0.0 |
| | | Coefficient 1 | 15.0 |

- When you are done saving block 12, you are presented with the Create New Block dialog. Press OK to create block #13.

### 4.9.9 Creating Block #13

- You have now created all the flow paths. It is time to create the tanks. Let's make block #13 TANK1. The algorithm that this block is based on is algorithm #858 DEMO TANK.

- Press Edit Inps to open the input table.

- The TANK1 block has four inputs. The tank in this demo has two inlets and two outlets. The algorithm takes into account the flow rates entering and leaving the inlets and outlets and adjusts the tank level accordingly.

- Based on the P&ID diagram with the flow rates given, the flow rates which are input to TANK1 are FLOW5/A01 and FLOW1/A01. Likewise, the flow rates which deplete from TANK1 are FLOW4/A01 and FLOW3/A01. Make these connections. Connect FLOW5/A01 to input #1, FLOW1/A01 to input #2, FLOW3/A01 to input #3, and FLOW4/A01 to input #4.

- Press Edit Outs to open the output table.

- TANK1 has two outputs: tank level and tank overflow indicator. At the start of the simulation, TANK1 should be set to a level of 5 meters. At a level of 5 meters, there is no overflow. Therefore, the tank overflow indicator having a 0.0 (FALSE) value is fine.

- Press Edit Cofs to open the coefficient table. The four coefficients for TANK1 are: tank height (10 m), tank radius (0.25 m), liquid density (1000 kg/m3), and height of outlet from the bottom of the tank (0.5 m). Enter these values into the DATAVALUE column.

- Press Close Cofs to close the coefficient table.

- Press Save to save the editing you have done.

- You are now presented with the Create New Block dialog. Press OK to create block #14.

### 4.9.10 Creating Block #14

- Tank 2 is very much like tank 1. You fill in the details.

- Set the initial TANK2 level to 0.0. Also note that there is only one outlet in TANK2.

- The following tables list the values you should have in the TANK2 block, use this table as reference.

| Block #14 | Algorithm | Pin | Data Value or Connection |
|---|---|---|---|
| TANK2 | Alg #858 DEMO_TANK | Input 1 | FLOW4/A01 |
| | | Input 2 | FLOW2/A01 |
| | | Input 3 | FLOW5/A01 |
| | | Input 4 | $NO_FLOW, 0.0 |
| | | Output 1 | 0.0 |
| | | Output 2 | 0.0 |
| | | Coefficient 1 | 20.0 |
| | | Coefficient 2 | 0.25 |
| | | Coefficient 3 | 1000.0 |
| | | Coefficient 4 | 0.5 |

### 4.10 Hands-on Testing and Debugging the Simple Two Tanks Process

#### 4.10.1 Verifying The Model

On the Model drop-down menu, you will find a Verify menu item. Select this now to verify that there are no errors within the model. The Verify process performs three checks. Examples of potential errors are: connections not specified, or an input connection specifies a block or output connection that does not exist. If you have followed the steps presented in this manual, you should not have any errors. A successful Model | Verify will produce the following messages.

```
(14:10:36) Model Verify begin ...
(14:10:37)  Performing sequence numbers check
(14:10:37)  Performing inputs, outputs, coefficients check
  Processing block   14
(14:10:38)  Performing hanging tags check
(14:10:38) Model Verify end.
```

If you are unfortunate enough to run into some errors, the listing of errors are captured in a text file. The Verify process will display a message notifying you that there are errors and will ask you if you want to see the error listing text file. Press OK to open this text file using MS Write. The first MS Write dialog will prompt you whether you want to convert the text file to Write format. Press No Conversion.

The most common error are hanging tag errors. A Hanging Tag is an erroneous connection (hanging connection) specified under the

TAGNAME column in the input table. That is, the TAGNAME entry for the input is suppose to be the output tag name of a block output to connect to, but instead is a tag name which does not identify any output tag name in the model. This usually occurs as a result of typing errors. If you used the Connect button to specify connections, then you would not have hanging tag problems.

Once you have corrected the errors (if any) ... YOU ARE DONE! All the detailed work is done. Now comes the fun part of running the simulation!

### 4.10.2 Exporting Model Data File for CASSIM™ Simulation Engine — CASSENG

The model you have finished building is now ready for testing. To test the simulation, you need the Simulation Engine — CASSENG and the Debugger.

- First, the simulation engine requires a compact data file from CASSBASE to run the simulation.

- To produce this file, select the menu item Export from the Model drop-down menu. You are immediately presented with a file dialog to enter the name of the compact data file to export to. Using the directory selection list on the right hand side of the dialog, position yourself to the \CASSDEMO\2TANKS directory. Then, in the entry field under the File name:, type in the filename TANKS.DAT. Press OK.

- CASSBASE will create the compact data file \CASSDEMO\2TANKS\TANKS.DAT.

- Let's start up CASSENG and open the compact data file. First minimize CASSBASE. In your Program Manager, double-click on the icon that looks like a train engine.

- CASSENG opens up a small window in the lower right hand corner of the screen. Once the CASSENG window is opened, select the File menu, scroll down and select the Open menu item. You are presented with a file dialog for selecting the data file to open. Select the TANKS.DAT file in the \CASSDEMO\2TANKS directory.

- The simulation model runs inside CASSENG. There are some simulation controls under the Control menu. You can FREEZE (stop), and RUN the simulation. In addition, the simulation can be stepped through by ITERATION, or by BLOCK. When the simulation is running, the  icon is displayed. Select the RUN option and see what happen !

- These controls are not very exciting since you cannot see the model values change as the simulation runs. There is a debug tool in CASSIM™ that allows you to watch the inputs, outputs, and coefficients of any block while the simulation is running.

The above only demonstrates that CASSENG and the model data file can run by itself, but you do not see much going on. In order to visualize the simulation data, you need to invoke the CASSIM Debugger:

- Quit CASSENG by selecting File menu in CASSENG and scroll down to select EXIT.

- Exit CASSBASE by selecting the menu item Model | Exit. Some prompt are displayed:

  Close C:\CASSDEMO\2TANKS model? Select OK.

  Do you want to make a backup? Select OK

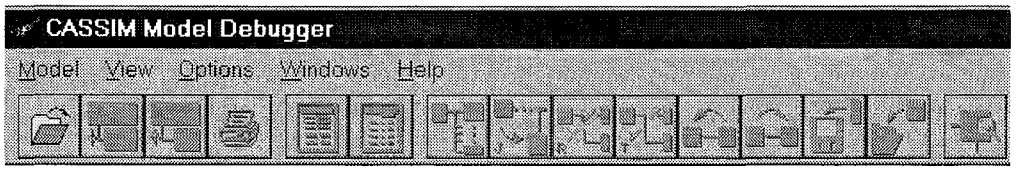
  Exit CASSBASE? Select OK.

### 4.10.3 Testing the Simulation Using CASSIM™ Debugger

**(A) Block Debugging**

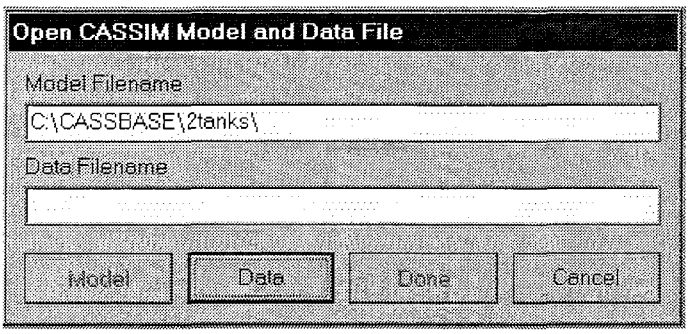
- Go to Window Program Manager and invoke CASSIM Debugger with this icon


- The following CASSIM Debugger Menu will appear.

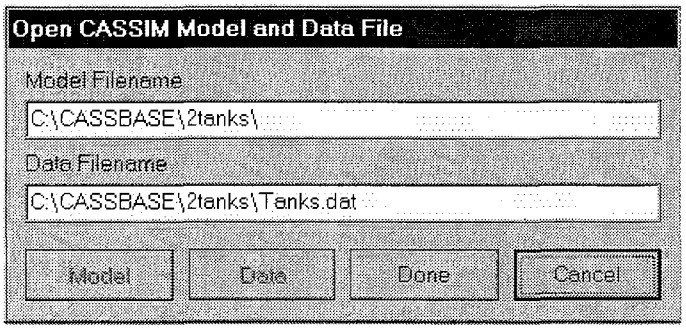



- Click on the Model folder (yellow color). A "Model Open" Dialog box shows up.

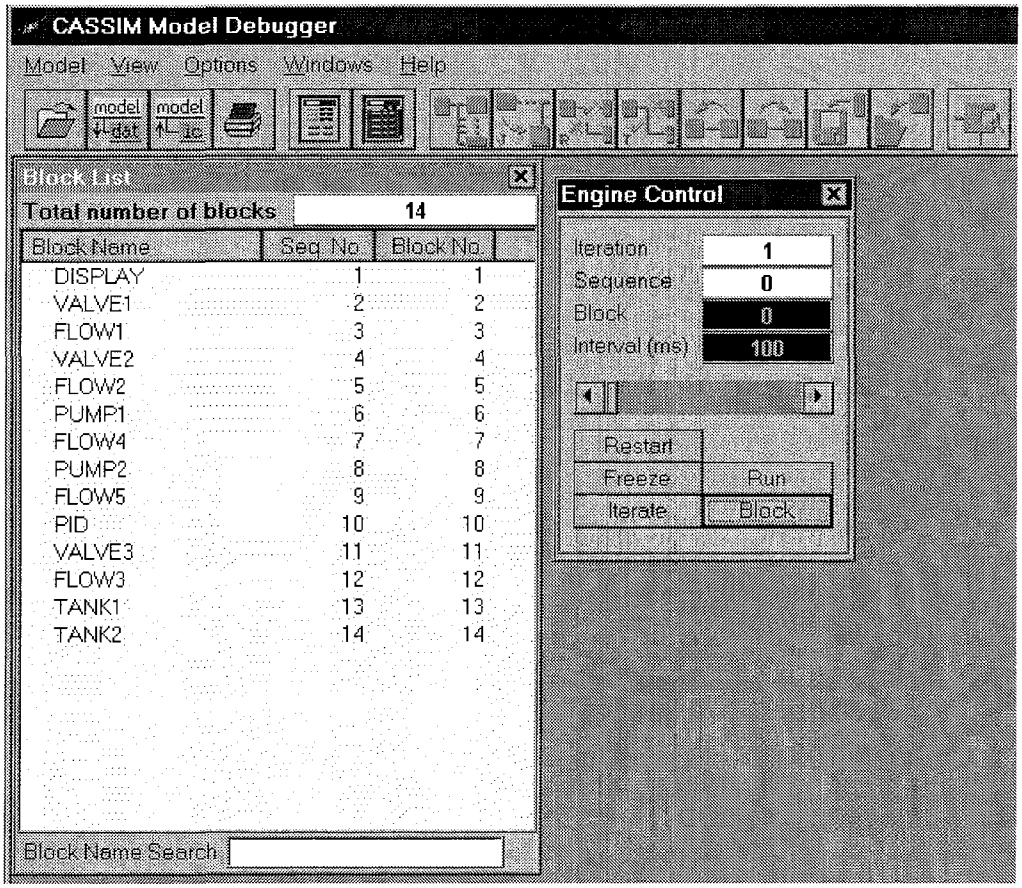
- Press "Model" button. A dialog box will appears, requesting your input for the directory where the Model resides. Select CASSBASE\CASSDEMO\2TANKS directory, and click on any .dbf file in the directory. The model will be loaded.



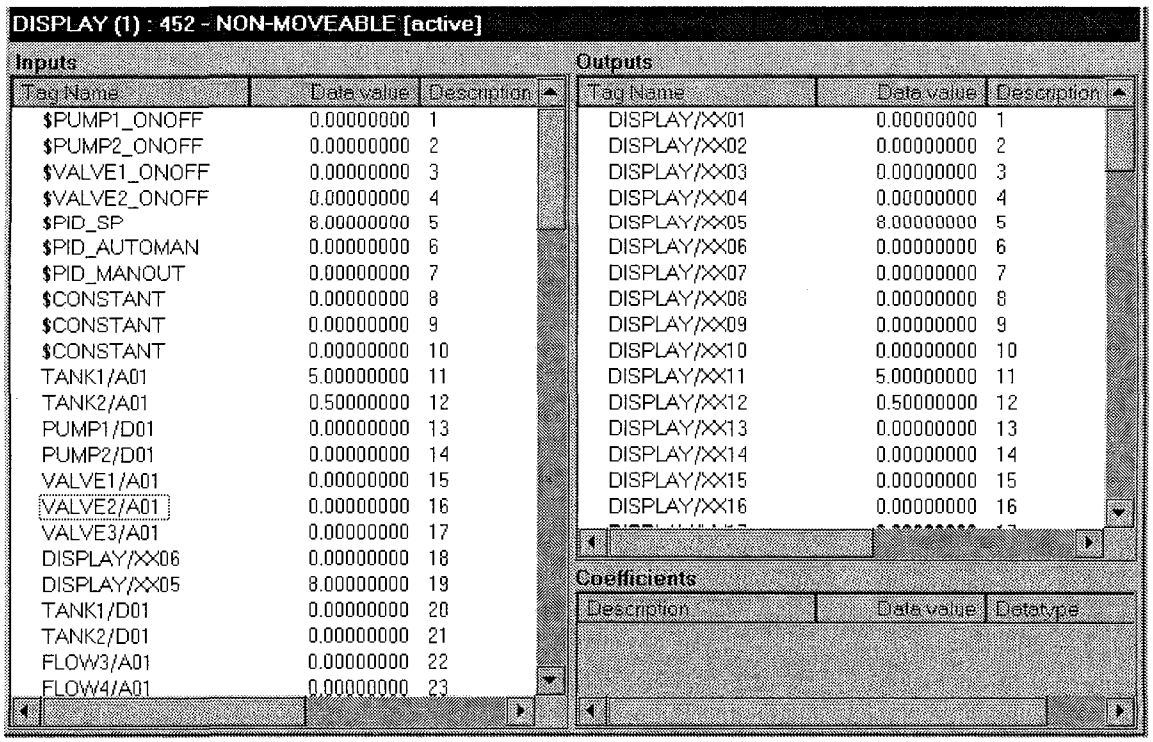

- Press "Data" Button. A dialog box will appear, requesting your input for model data file. Click "Tanks.dat" to select the model data file.




- Press "Done" Button. You will see the Debugger is loading up the Model.
- Use the mouse to click on "Engine Control" and "Block List" Menu items. (Note if you do not know where they are, move the mouse cursor to any of the menu items, you will see a "tail-tips" behind it). You will see the following dialog boxes.




- Double-click on the first block on the list "DISPLAY". The DISPLAY block will be presented.



- The debug block window communicates with CASSENG via Microsoft Window's Dynamic Data Exchange (DDE) facility. The debug block window requests data from CASSENG and displays the returned data. Likewise, input from the user is sent by the debug block window to CASSENG and these inputs are integrated into the running simulation by CASSENG.

If you wish to know more about debugging operations, press "HELP" menu item to read more details.

**(B) Controlling the Simulation**

- The "Engine Controls" can be used to Freeze or Run the simulation.
- Press Iterate to step the simulation through one iteration of all the blocks in the model.
- Press Block to step the simulation one block at a time.
- The Iteration count shows how many iterations the simulation has run.
- The Sequence count shows the sequence number of the block that has just completed execution. The Block value shows the block number corresponding the sequence count shown. When the simulation is running, the Iteration counter will be counting.

*Exercises:*

If the simulation is frozen (i.e., counter not counting), press Run now. The Iteration count will count up. Press Freeze to stop the simulation. Press Step Iteration once and notice the iteration count go up by one.

**(C) Jumping between Connections**

- The inputs of the current block may be connected to outputs of other blocks. Likewise, the outputs of the current block may be connected to inputs of other blocks.
- Use the "left" mouse button to click on the first block output DISPLAY/XX01. It will be highlighted in "blue".
- With DISPLAY/XX01" highlighted, click the "right" mouse button to display a "drop-down" menu for "Connection List" or "jump". Select Jump to trace the connections from one block to another.
- If the selection is connected to another block, that connected block will be displayed in a separate debug block window. In this case, PUMP1 block will be displayed, because its input is connected to output: DISPLAY/XX01. Note: if there is no connection to the selected input or output, a message is displayed " No Jump Point".
- For outputs, it is possible that one output is connected to many inputs. If this is true, a connection list is shown after you select the "Connection List" in the drop-down menu. Select one of the blocks in the connection list to open the debug window for that block.

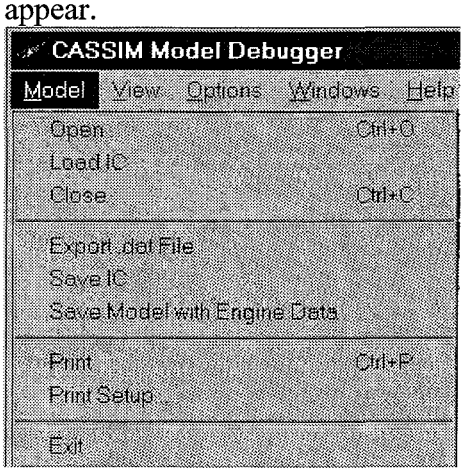
*Exercise:*

- Single-click on the 11th input line, it looks like:

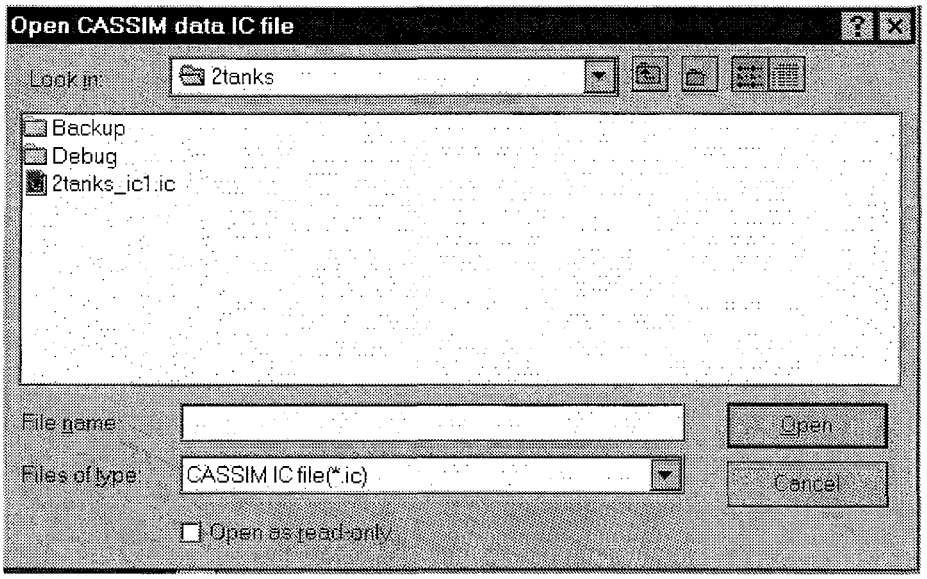
TANK1/A01 .....

- This selects the input and it will be highlighted in "blue". This input is connected to the block TANK1's A01 output. Try to display the debug window for block TANK1. Loading Initial Conditions

- Single-click on Debugger Menu item "Model", a drop down menu will appear.




- Select "Load IC", meaning to load an Initial Condition. The following dialog box will appear.




- Note that the 2 tanks process that you are simulating can be at different process conditions — that is, some pumps may be turned on and some valves may be opened. So during debugging, you can store and reload a particular process condition of interest. These initial conditions are stored as IC files with *.ic file extension.

- Select 2 tanks_ic1.ic. This is a pre-stored IC for a normal condition for the process.
- You will see the IC file is loaded via the "progress indicator".

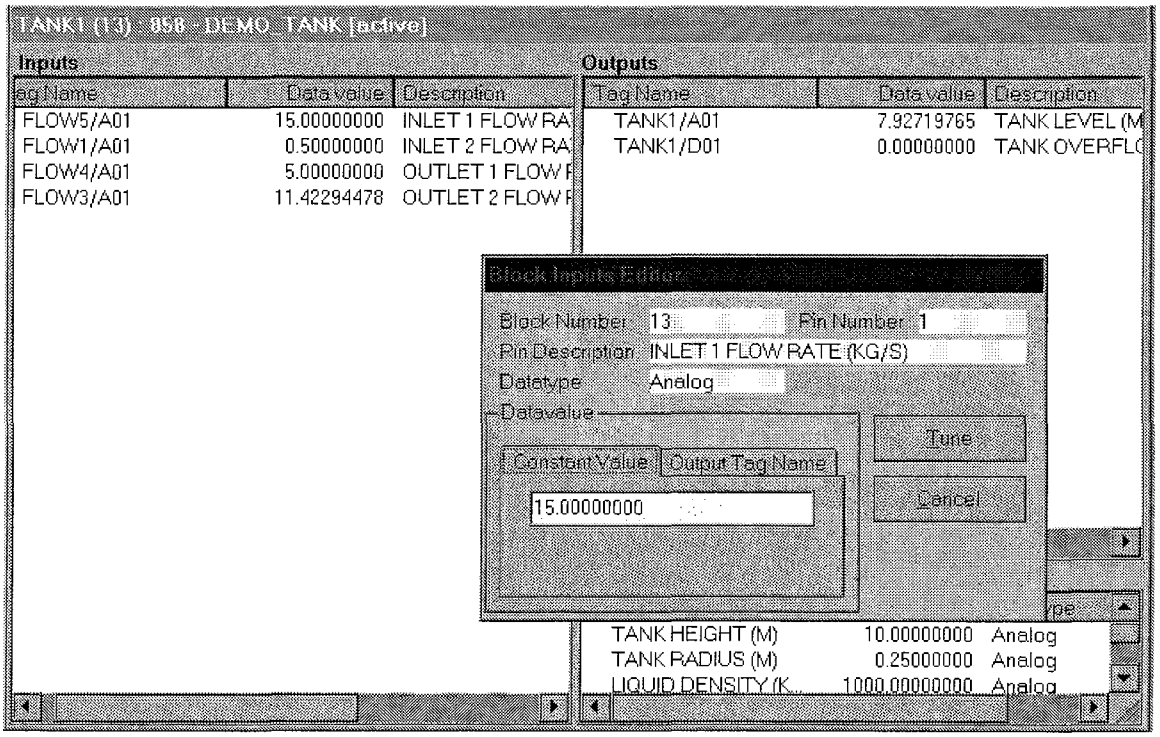
Now, we can do some on-line tuning.

**(D) On-line Tuning of Simulation**

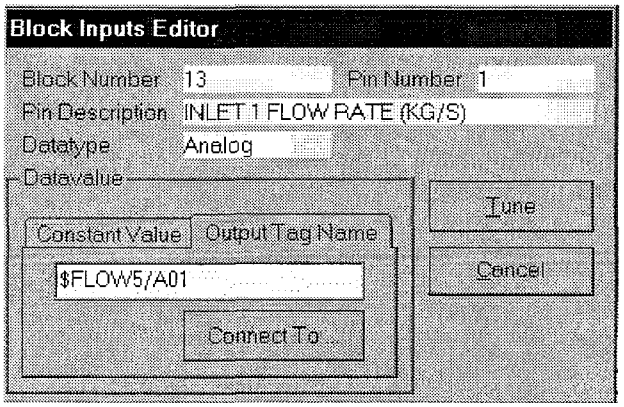
- Double-click on any Input, Output, or Coefficient line of any block to bring up the Tune dialog for manually changing that value on-line.
- The output and coefficient tune dialog are virtually the same, displaying all information about the selected item and allowing input of a new Data Value.
- After modifying the data value, press Tune to send this updated value to CASSENG to include in the simulation. The input tune dialog optionally allows for modifying a constant data value (in this case, tag name would start with a $), or modifying a tag name to change the output to which this input is connected to.
- If the current input is defined as a constant value, either change the value or press Output Tag Name and enter a valid tag name in the entry box. If the current input is defined as a output tag name, either change to another tag name using the Connect button, or press Constant Value and enter a data value in the entry box. Press Tune to send this change to CASSENG to include in the simulation.

*Exercise:*

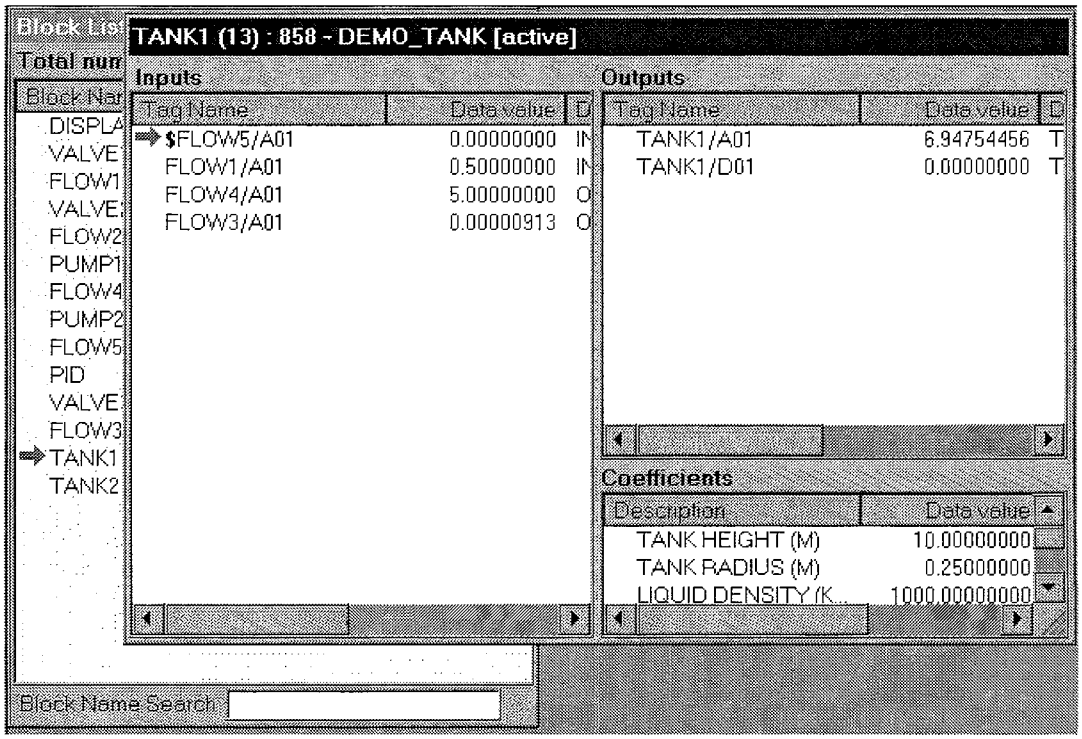
- Let's tune some of the inputs of TANK1. First bring up the debug window for TANK1.
- First run the simulation by pressing Run on "Engine Control". Note that there is an inflow of 15 kg/s (input 1 connected to FLOW5/A01) and an outflow of 5 kg/s (input 3 connected to FLOW4/A01). The tank level output of the tank (TANK1/A01) is increasing because there is more flow coming in than going out.
- Let's turn off the inflow at input 1. Double-click on the input 1 line. This will bring up a Block Inputs Editor dialog as follows:




- Press Output Tag Name, you will see the Input Tagname FLOW5/A01.
- Put a $ sign in front of FLOW5/A01. In other words, you have disconnected the output connection from Block FLow5 output A01, and in its place, you have put a constant tag name called $FLOW5/A01 as input.




- Press "Constant Value" in the Block Input Editor dialog box. This will switch the Data Value area to allow entering a constant value. Enter 0.0 in the entry box, and press Tune. The input is now tuned to have a constant inflow of 0 kg/s.




- You will see "red" arrow pointing input $FLOW5/A01 in TANK1 Block. As well, you will see "red arrow" pointing the block TANK1, indicating tuning changes are made to this block.
- Now press Run and notice that the tank level output is decreasing because there is no inflow and still an outflow of 5 kg/s.
- Repeat the steps similar to above to tune the outflow input currently connected to FLOW4/A01 to have a constant outflow of 0 kg/s. Run the simulation and notice that the tank level output is constant because there is no inflow and no outflow.

### (E) Restoring Original Model Values and Saving Tuned Values

- Note that tuning modifies the run time values used by CASSENG in the running simulation. The tuned values are not stored in the model database.
- For any tuned input or coefficient, the original model value in the model database can be restored to replace the tuned value in the simulation.
- Conversely, the tuned value can be saved to the model database to replace the original model value.
- For any input or coefficient that has been tuned, it is possible to discard the tuned value and restore the running simulation to use the original model value defined in the model database.
- To do this, single click the tuned input $FLOW5/A01. It will be highlighted in "blue", then press the "right" mouse button and select "Restore".
- Once the value has been restored to that defined in the model database, the line will no longer have a red arrow pointer.

- All tuned values are used in the running simulation only and are not stored in the model database if not saved. To save the tuned values into the model database, single click anywhere in a block debug window (but not on any input, output or coefficient line); then click the "right" mouse button. A drop-down menu will appear, then select "save block".

- **Caution for Saving Block Tuning Changes**

1. Before you save the block changes into the model, make sure the changes are final and not temporary ones, because after you have "saved" the block changes, you have to re-export the model data file TANKS.DAT.
2. If the change is temporary, it is recommended to save the block tuning changes in a new IC file. You can do that by clicking on the Debugger Menu "Model" and select "Save IC" in the drop-down menu. You will be asked to enter a new IC file name. As debugging progresses, you can continue to make incremental "save" of the tuning changes into this IC file, or if you wish, you can save into another new file, as the situation requires. If there are numerous IC files, it is recommended that you should label the IC files succinctly so that you may easily recall what tuning changes those files have had, by looking at their names.
3. After numerous testing, and if you are satisfied that the tuning changes are final, you can select to update the model database with your IC file. The way to do it is to first load the IC file as you have done previously in "Loading Initial Conditions". After the simulation engine is loaded with IC data, you select Debugger menu item " Save Model with engine data", the tuning changes will be saved into the model database. You have to re-export the model data file TANKS.DAT.

*Exercise:*

- Open TANK1 block and tune the inlet flow to be 10 Kg/s. As well, tune the outlet flow to be 5 Kg/s.
- Save an IC file for this condition, and call this file tanks_ic2.ic
- Exit the Debugger, and make sure to exit the CASSENG as well.
- Now re-invoke the Debugger, load the original Model Data file TANKS.DAT.
- Load the IC file tanks_ic.ic
- Restore all the tuning changes that you have made previously.

You should now have an idea of the debug capabilities of CASSBASE in providing dynamic debugging at simulation run time, and you are now ready to run the graphical user interface to interact with the simulation you built using CASSBASE.

## 4.11 Running the Simulation Model with LabVIEW® Screen

### 4.11.1 Running the Two Tanks Simulation

A graphical interface program is distributed with the CASSIM demonstration software to allow you to interact with the CASSENG simulation engine. This program is a Windows executable program called 2TANKS.EXE installed in the C:\CASSDEMO directory.

2TANKS.EXE is a runtime program developed using LabVIEW for Windows. LabVIEW for Windows is included in the CASSIM product. Normally, you would use LabVIEW to develop the screen display program for interacting with the simulation. But using LabVIEW can be a course by itself and is beyond the scope of this Lesson. However, if you have LabVIEW, it is recommended to go over the LabVIEW tutorials.

LabVIEW uses a graphical programming environment for developing data acquisition and display programs. The user displays are built by dragging and dropping graphical components like buttons, digital readouts, graphs, etc. onto a front panel. After completing the front panel, the program which operate with the front panel is built by assembling and connecting icons in a diagram, instead of writing text-based programs. A LabVIEW diagram resembles a traditional engineering flowchart with computational icons, programming structure icons, communication icons, etc. The icons are connected together with lines, or wires, to direct the flow of data within the program.

A running LabVIEW program communicates through DDE with CASSENG in the same way the debug block window of CASSBASE works with CASSENG. A LabVIEW program can request data from CASSENG for display. A LabVIEW program can also send user input to CASSENG for integrating into the running simulation. CTI includes with CASSIM a customized library of LabVIEW components specifically designed for building simulation user interface programs.

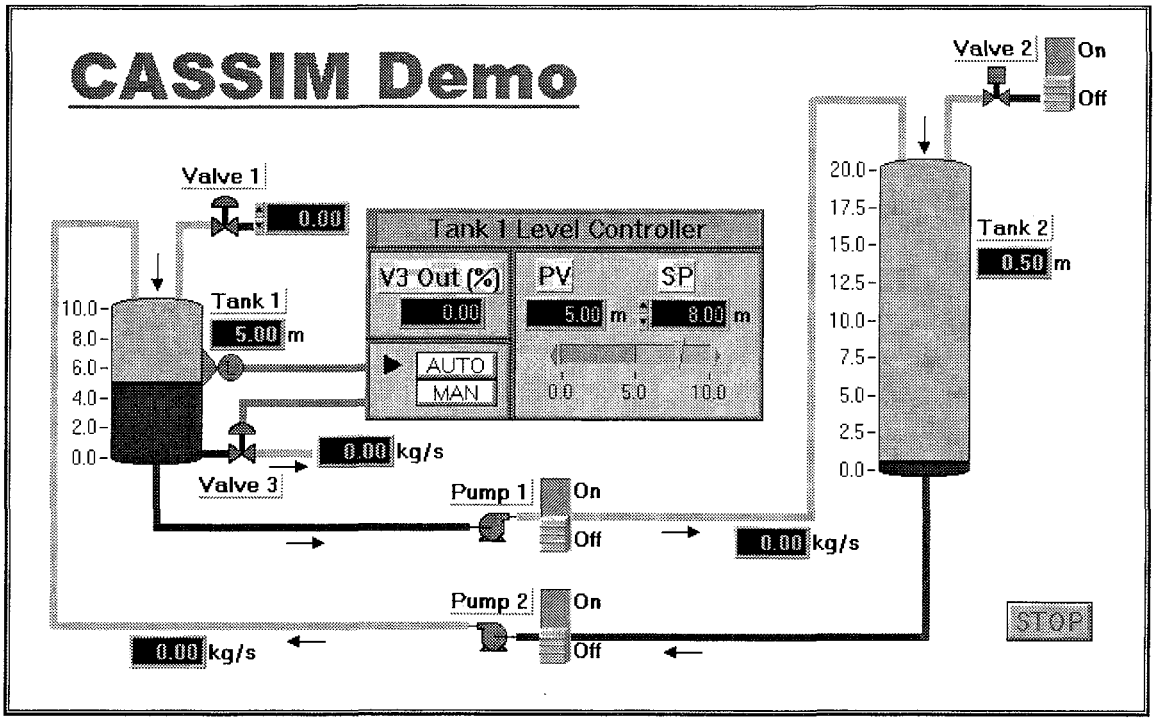
- Before running 2TANKS.EXE, start CASSIM Debugger, and load the 2 tanks model, as well as the Model data file.

  C:\CASSBASE\CASSDEMO\2TANKS\TANKS.DAT.

- After the Debugger is loaded, minimize it.

- Open Program Manager and double-click on the icon that has two tanks:


- This program sets the CASSENG simulation into run mode. The screen appears as shown on the next page.



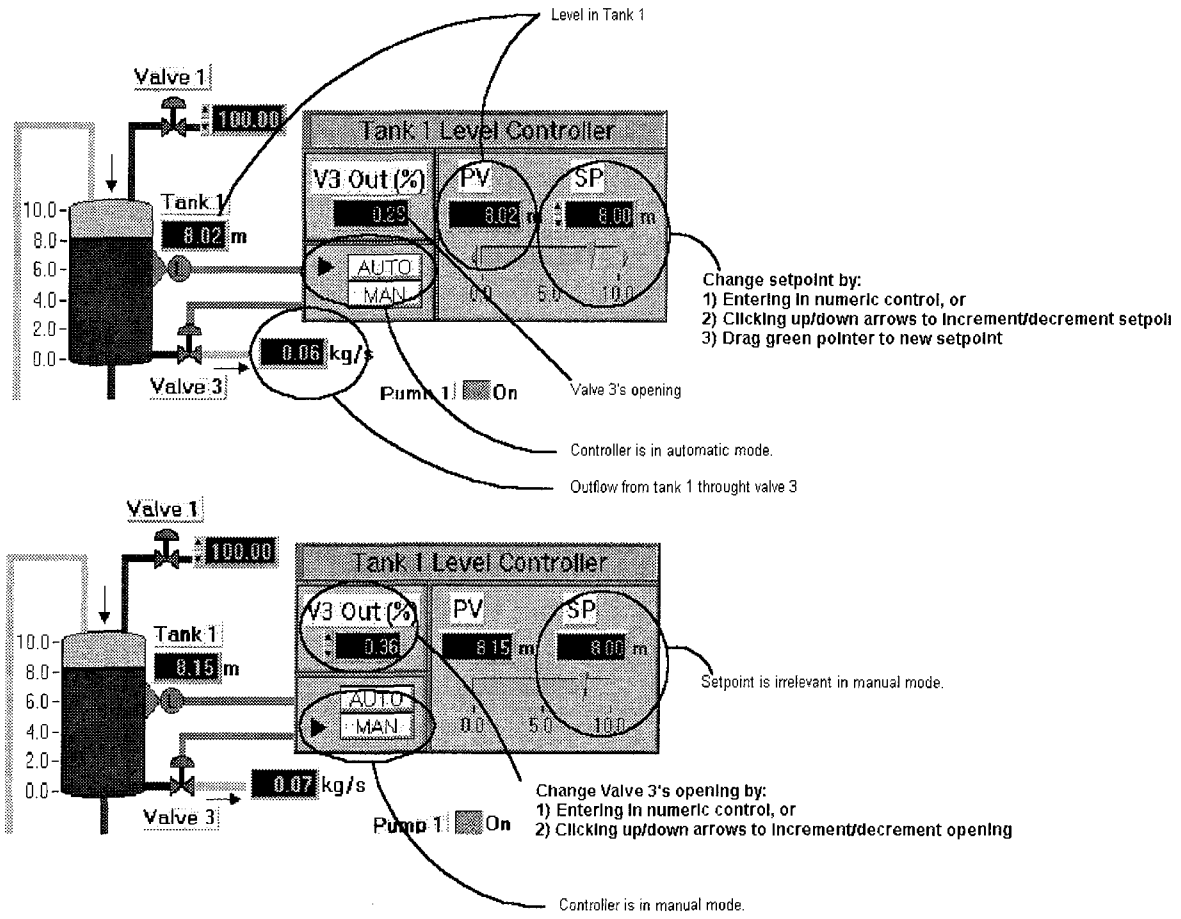

- At this point, you will notice there are THREE tasks running in Windows: (1) CASSENG (2) Debugger (3) LabVIEW graphics executable.
- The equipment that you modeled in CASSBASE are presented graphically in the program window. There are two tanks, with pumps on the pipes connecting them. Valves 1 and 2 control the inlets to tanks 1 and 2 respectively. There is a level controller on tank 1 controlling the outflow through valve 3.
- Valves are green in color when they are fully closed, and red when they are fully opened.
- Turn on valve 2 now by clicking the On/Off switch besides the valve. Notice the pipe flowing into tank 2 fill up and the level of tank 2 rises. The level of tank 2 is shown along the scale to the left of the tank and as a numeric display to the left of the tank.
- Recall that tank 2 is the reservoir tank. Let's try to fill this tank's level up to 10 m. To speed things along, it may help to use tank 1 to fill up tank 2. First open the flow from tank 1 to tank 2 by turning on pump 1. Turn on pump 1 now by clicking the On/Off switch besides the pump. Notice the pipe connecting pump 1 to tank 2 fill up. This flow into tank 2 will speed up the fill of tank 2.
- At the moment there is only outflow from tank 1 and no inflow. Eventually, this will deplete the liquid from tank 1. To add some inflow, turn on valve 1 controlling the flow from an external source. Turn on valve 1 now to fully opened by typing in 100.0 in the numeric control besides valve 1 and press Enter. Recall that valve 1 can be controlled to be partially opened.

- **A couple of things to take note of here**

1. Notice that although you typed in 100.0% opening, the valve still takes time to stroke. In fact, the valve was modeled to take 5 seconds to stroke for 0 to 90% opened. The 100.0% entered is the demand opening only. There is no numeric display shown of the actual valve opening. The color of the valve helps in this case. When the valve was fully closed, the color was all green. When the valve is stroking, the color is half red and half green. When the valve reaches 100.0% opened, the valve color changes to all red.
2. Also notice that once the valve has completely opened to 100.0%, the total inflow rate (with pump 2 off) is 5 kg/s as modeled. The total outflow rate (with valve 3 off) is also 5 kg/s. This effectively makes the level in tank 1 remain constant.
3. Is tank 2's level filled up to 10 m yet? Once the level gets close to 10 m, turn off first pump 1, then valve 2. This shuts off all flow into tank 2.
4. Time to squeeze some juice out of valve 3! That is, let's use the level controller to operate valve 3 to let out the liquid in tank 1.
5. All along, valve 3 is being controlled by the level controller which is in automatic mode. The setpoint is at 8 m. That means as long as the level in tank 1 is below 8 m, then the controller will keep valve 3 close. Once the level goes above 8 m, the controller will open up valve 3 accordingly to let out some flow from tank 1.
6. The following diagrams show how to use the level controller in automatic mode and in manual mode.

- If the level in tank 1 has not reached 8 m yet, wait for it. If you are impatient, you can lower the setpoint to a lower level. Try typing 7.0 into the setpoint (SP) numeric control. The controller will adjust its control on valve 3 accordingly.
- Once the tank level has reached the setpoint, valve 3 opens. Turn off valve 1 now to shut all flow into tank 1. Watch the controller slowly close valve 3 as the level in tank 1 comes below the setpoint.
- Another method of letting some liquid out of tank 1 is to manually operate valve 3. To do this, switch the controller to manual mode. Switch to manual mode now by pressing on the MAN button. The pointer will switch to point to MAN. You can now control valve 3 by entering the desired opening in the numeric control under the label V3 Out (%). Open valve 3 completely now to 100.0% by typing in the numeric control. Notice the flow out of valve 3 increase up to the modeled maximum flow of 20 kg/s.

- At this point, you have interacted with most of the equipment modeled using CASSBASE. Feel free to continue to run the simulation, trying out different operating scenarios.




- NOW maximize Debugger, practice some tuning changes and observe that the changes can be seen in the screen — e.g. change the tank diameter.

## 4.12 Merging Models

It is highly desirable in simulation modeling that some sub-system modules can be extracted and recycled for different simulation projects. For example, the feedwater model for power plant can be used for another plant if similar feedwater design is used. CASSIM™ has a "model merging" feature which facilitates easy model copying and merging of the model. The merging of models will be demonstrated.

## 4.13 Review of Thermalhydraulic Network Modeling

Solutions to differential equations representing the thermalhydraulics of a power plant process contain a wide spread of time constants, i.e. some components of the solutions have a fast response, while the remaining components are slow varying. The eigenvalues caused by the mass and

momentum conservation equations (flow and pressure) are generally very large while the eigenvalues caused by thermodynamics (temperature) are small. Therefore a system of coupled differential equations describing mass, momentum and energy dynamics of a large process could contain wide spread of eigenvalues, and thus is usually referred to as "stiff" set of equations. Stiff equations are usually solved by implicit methods due to their inherent stability. However, standard implicit methods usually require finding the Jacobian for the equation set during each iteration (Newton's method), as well as solving the resulting set of linear algebraic equations. These solution techniques can provide solutions with high degree of accuracy and stability; however, they cannot be solved "economically" in real time.

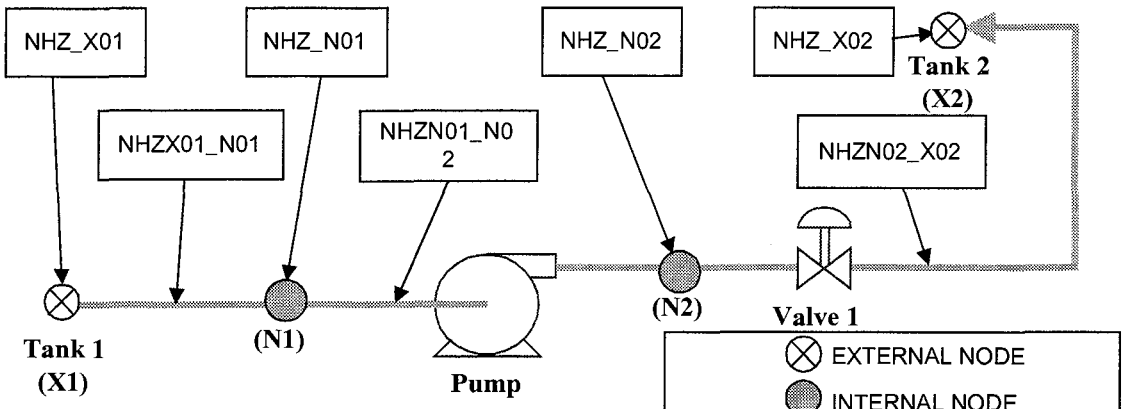
CASSIM™ employs a real time solution technique for thermalhydraulic modeling, which involves identifying control volumes within the simulated system, where pressures are calculated. These control volumes are known as nodes. The flows between these nodes can be obtained from momentum equations, and are known as links. Each node pressure can be obtained from mass balance around the node. A linearized solution method is employed for the nodal representation of the flow network. This method involves linearizing the momentum equations, and solving the resultant set of linear algebraic equations implicitly by matrix methods.

Figure 8 illustrates the mass and momentum equations set for a typical flow path containing equipments such as pumps and valves, and demonstrates how these equations can be linearized to be solved by matrix methods. By specifying nodes and links, the flow network problem can be defined using modular block structured modeling, that is, each node block and link block can be created by using generic node and link algorithm respectively.

Hydraulic properties of the nodes and links blocks (e.g. node size, valve size, pump dynamic head etc.) are either treated as coefficients or inputs to these blocks. CASSBASE, the database engine of CASSIM™, has a computerized network generation program which will automatically create matrix elements for the network defined by the connected nodes and links blocks.

As the flow network problem generally creates "sparse" type of algebraic systems, the network generation program can reduce the computation time required to obtain a solution by rearranging the algebraic equation set. This computerized network generation program produces a data file which can be run during each simulation iteration. If the topology governing the nodes and links has been changed, the network generation program has to be used to re-generate the network again.

# *HYDRAULIC NETWORK FLOW DIAGRAM*




Assuming incompressible flow

$$W_{X1N1} = K_{X1N1}(P_{X1} - P_{N1})^{1/2}$$
$$W_{N1N2} = K_{N1N2}(P_{N1} + P_{DYH} - P_{N2})^{1/2}$$
$$W_{N2X2} = K_{N2X2}V_A(P_{N2} - P_{ELV} - P_{X2})^{1/2}$$

Linearizing equation by defining admittance

$$A_{X1N1} = W_{X1N1} / (P_{X1} - P_{N1})$$
$$A_{N1N2} = W_{N1N2} / (P_{N1} + P_{DYH} - P_{N2})$$
$$A_{N2X2} = W_{N2X2} / (P_{N2} - P_{ELV} - P_{X2})$$

Performing mass balance on N1 and N2; assuming constant density

$$C_{N1}(dP_{N1}/dt) = W_{X1N1} - W_{N1N2}$$
$$C_{N2}(dP_{N2}/dt) = W_{N1N2} - W_{N2X2}$$

**Legend**

| | |
|---|---|
| $W_{ij}$ | = flow from node i to j |
| $P_i$ | = pressure in node i |
| $V_A$ | = valve port area |
| $P_{DYH}$ | = pump dynamic head |
| $P_{ELV}$ | = elevation head |
| $A_{ij}$ | = admittance between node i and j |
| $K_{ij}$ | = conductance between node i and j |
| $C_i$ | = capacitance of node i |
| $\Delta t$ | = time increment |
| $P'_i$ | = Pi at previous iteration |
| $M_{ij}$ | = matrix element ij |
| $S_i$ | = source element i |

Using backward Euler; and substitute admittance

$$C_{N1}((P_{N1} - P'_{N1})/\Delta t) = A_{X1N1}(P_{X1} - P_{N1}) - A_{N1N2}(P_{N1} + P_{DYH} - P_{N2})$$
$$C_{N2}((P_{N2} - P'_{N2})/\Delta t) = A_{N1N2}(P_{N1} + P_{DYH} - P_{N2}) - A_{N2X2}(P_{N2} - P_{ELV} - P_{X2})$$

Rearranging to matrix form

$$\begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix} \bullet \begin{pmatrix} P_{N1} \\ P_{N2} \end{pmatrix} = \begin{pmatrix} S_1 \\ S_2 \end{pmatrix}$$

where

$$M_{11} = (C_{N1}/\Delta t) + A_{X1N1} + A_{N1N2} \qquad M_{12} = A_{N1N2}$$
$$M_{21} = -A_{N1N2} \qquad M_{22} = (C_{N2}/\Delta t) + A_{N1N2} + A_{N2X2}$$

$$S_1 = (C_{N1}/\Delta t)P'_{N1} + A_{X1N1}P_{X1} - A_{N1N2}P_{DYH}$$
$$S_2 = (C_{N2}/\Delta t)P'_{N2} + A_{N1N2}P_{DYH} - A_{N2X2}(P_{ELV} + P_{X2})$$

Procedure for solving pressure solution:
1) Calculate admittance from flow, $P'_{N1}$, and $P'_{N2}$
2) Solve $P_{N1}$, $P_{N2}$ from matrix
3) Calculate flow from $P_{N1}$, $P_{N2}$

*Figure 8 — CASSIM™ Hydraulic Flow Network Numerical Method Technique.*

The above diagram illustrates a simple hydraulic circuit which involves a flow path from an open tank#1, through a suction header N1, a pump P1, a discharge header N2, a valve#1, and lastly another open tank#2.

In hydraulic network terminology, we call open tank#1, #2 as "external node" X1, X2 respectively; whereas for the suction and discharge headers, we call them "internal node" N1 and N2 respectively. According to

CASSIM™ Network naming convention, we give this network a 3 letter pre-fix NHZ. Hence, all nodes and branches block name will have this pre-fix identifier. For instance, N1 will be labeled as NHZ_N1; the branch connecting Tank #1 and Node N1, is labeled as NHZX1_N1.

The equation sets in the above diagram can be summarized as follows:

(a) The flow in each link is computed using momentum balance for each link.

(b) The pressure in each node is computed using mass balance equation for each node.

The calculation scheme is as follows:

(1) Users create node blocks and link blocks. The node block output contains node pressure values; whereas the link block output contains link flow values.

(2) After users have constructed all the node and link blocks with the equipment (pumps and valves), and with proper initializations, users "generate" the network using CASSIM.

(3) After "network generation", the CASSIM utility has already built the matrix equation similar to that shown in the above diagram.

(4) In the first iteration, for each link, given the initial flow, initial pressure in upstream node i and downstream node j respectively, admittance is calculated in each link block (see equation above); and hence all the M's and S's in the matrix elements can be calculated. When these calculations are all done, the Network Solver inside CASSIM solves the matrix equation in real time to give the pressure values for all the nodes. Once the pressures of the nodes are solved, they update all the node blocks' pressure output.

(5) After the pressure update in the node blocks is completed, flow can then be calculated from executing the link flow block algorithm (see equation above).

(6) Now it is ready for the next iteration with exactly the same iteration scheme as described in (4) above.

Note:

(1) The equations formulated below for the hydraulic network pressure and flow calculations are only good for incompressible flow. For compressible flow (steam, air systems etc.), users have to consider the use of "variable" conductance in the link, in order to use the same set of incompressible flow equations. See explanation below.

(2) The hydraulic flow/pressure calculation is decoupled from the energy transfer calculations. The energy transfer in the link of a network (e.g.

heat exchanger) is done outside the hydraulic network blocks boundary. It can be achieved by using "mixing node" algorithm, in which the output temperature or enthalpy of the mixing node is calculated from the inlet streams' temperatures or enthalpies, assuming perfect mixing. The effect of temperature change can lead to a change in the flow in the link due to the change in the fluid density (input # 7 of the link algorithm).

### 4.13.1 CASSIM™ Thermalhydraulic Modeling Simplifications

- It should be mentioned that, for computational efficiency, there are two important simplifications taken by the CASSIM™ thermalhydraulic modeling approach:

- The hydraulic flow/pressure calculation is decoupled from the energy transfer calculations in the flow network. For example, the heat transfer in the link of a network (e.g. heat exchanger) is done outside the hydraulic network solution schema. It can be achieved by using "mixing node" algorithm, in which the output temperature or enthalpy of the mixing node is calculated from the inlet streams' temperatures or enthalpies, assuming perfect mixing. The effect of temperature change can lead to a change in flow in the link due to the change in the fluid density.

- The numerical solution technique for solving the network pressures and flows, as formulated in Figure 8, generally applies for ***incompressible flow***. For ***compressible*** flow such as steam, gas systems, etc., the following momentum equations *should have been* used instead, assuming constant temperature:

For non-choked flow:

$$Flow = K_c \cdot V_1 \cdot \left(P_{up}{}^2 - P_{down}{}^2\right)^{0.5}$$

For choked flow:

$$Flow = K_c \cdot V_1 \cdot P_{up}$$

where $K_c$ = link conductance; $V_1$ = Valve port area in the link between the upstream and downstream nodes; $P_{up}, P_{down}$ = upstream and downstream node pressure respectively.

- However, in CASSIM™, a "*variable conductance*" technique is developed, by which the same set of *incompressible* flow equations could be used for *compressible* flow systems. In this technique, when links flows are calculated by using the *incompressible* flow equations, they are substituted back into the above compressible momentum equations using the initial pressure conditions, to back calculate the link conductance $K_c$. The calculated link conductance, a block output,

is then connected to the link block as input. In other words, this technique "*allows*" the link momentum equation to be an "*incompressible*" flow type; however in this instance, the link conductance should not be fixed, but rather a "*variable*" one and is calculated in each iteration from the compressible momentum equations using back-substitution. The application of this technique allows the same numerical solution methodology formulated for "incompressible" systems (Fig. 8) to be applicable for "compressible" flow systems, with the additional computation of "*variable link conductance*".

### 4.13.2 How to Create a Hydraulic Network System

**REMEMBER** in CASSIM Hydraulic Network,

- Internal Nodes = Control volumes whose pressures will vary with time.
- External Node = Control volumes of constant pressures which do not vary with time.
- Branches/Inks =Flow between each node.

*The procedure for creating a hydraulic network system is:*

1) Create a Nodal Diagram
    - Identify location of the equipment and headers with respect to the nodal diagram.

2) Calculate/estimate the pressure drop between each node.

    (A) Pressure drop can be calculated using Darcy's Law of the exact piping configuration is known.

    (B) An estimate of the pressure drop can be made if:

    - The overall pressure drop of the system is known.
    - The number of valves involved is known.
    - Sharing the overall pressure drop between the branches and the valves.

3) Create the external nodes, internal nodes and branches/links blocks. Note it is very important to *label the node blocks in increasing order*, i.e. NH?_N01, NH?_N02, ..... NH?_N10 etc., as you traverse the Nodal Diagram from the "starting" node to the "ending" node in the direction of flow. Note that *the order 1-5-4-2-3-6 will not work.*

4) The following block naming convention for branches should be followed: NH?N01_N02 is the branch connecting node NH?_N01 to NH?_N02. See more details in CASSIM™ in the User Manual.

4) Create Network Control Block with Blockname ***NH?_CTRL*** and move it ahead of the network marker block ***NH?_END***

5) Run the Network "Generate" Utility.

## 4.14 Algorithm Source Code Development

### 4.14.1 FORTRAN Algorithm Coding Standard

In creating new FORTRAN Algorithm for CASSIM, the following FORTRAN coding standards should be adhere to when creating algorithms. These standards help ensure that double precision calculations are performed with as much accuracy as possible.

1. *Declare all variables.* By default, FORTRAN will give undeclared variables starting with particular letters default data types. The default data type may not be what you want.

   *Why?*

   If a variable is not declared, it may default to a wrong type. For example, according to the FORTRAN 77 language specification, any variable not explicitly declared and beginning with a letter other than I, J, K, L, M, and N will be of type REAL by default. Let's say you assigned a value to a variable called SPEED, and you did not explicitly declared SPEED as DOUBLE PRECISION. The value may lose precision because REAL (single precision) can only hold six to seven digits of precision.

   *Hint*

   Microsoft FORTRAN 5.1: Use the /4Yd option when compiling to generate warning messages when undeclared variables are encountered.

   Watcom FORTRAN-77 10.5: Turn on the Require symbol declaration (option exp) when compiling to generate warning messages when undeclared variables are encountered.

2. *Assign a value to a variable before referencing it in an expression or using it on the right hand side of an assignment statement.*

   Why?

   If a variable has not been assigned a value, the memory space occupied by it contains garbage data. When you reference this undefined variable, the garbage data will be interpreted as legitimate data and will cause unexpected results.

   This problem can be hidden within code. For example, an IF-THEN-ELSE structure assigns a value to a variable only when the condition is

TRUE but does nothing if the condition is FALSE. A reference to that variable just after the IF-THEN-ELSE structure will return the assigned value if the condition was TRUE and will return garbage data if the condition was FALSE.

3. Explicitly declare all floating point constants as double precision by using a D to indicate the exponent.

   *Examples:*

   123.764 Incorrect

   .4352344 Incorrect

   1423.34E12 Incorrect

   +345.E-4 Incorrect

   -.4565788E3 Incorrect

   2E6 Incorrect

   1234. Incorrect

   The above constants are all interpreted as single precision floating point numbers. The following constants are double precision floating point numbers.

   123.764D0 Correct

   1423.34D12 Correct

   +345.D-4 Correct

   -.4565788D3 Correct

   2D6 Correct

   1234.D0 Correct

4. When assigning a value to a double precision variable, do not assign an integer, or a single precision floating point number.

   *Examples:*

   TLOAD = 0 Incorrect
   TLOAD = 0.0 Incorrect
   TLOAD = 0.0E0 Incorrect

TLOAD = 0.0D0  Correct

5. When writing a mathematical equation, Use the same data type in all parts of the calculation to ensure that precision is not lost.

### 4.14.2 Algorithm Source Code Standard

See details in CASSIM™ User Manual Section 2.3.2.

### 4.14.3 Simple Algorithm Exercise

1) Develop an FORTRAN Algorithm for the simple ordinary differential equation:

$dY/dt = AY + C$

where A and C are constants and treated as coefficients.

The initial condition is: $Y(0) = 0$.

2) Use Euler integration method for solving the ODE numerically.

3) After the algorithm is developed, create a block using this algorithm. Use Debugger to change values of A = 2.0, -2.0, 0.000001, and values of C = 10.

## 5.1 Quick Review of Nuclear Reactor Neutron Physics

### 5.1.1 Rates of Neutron Reactions - Reactor Power

- *Reaction Rate:*

Consider **n** neutrons per unit volume move with velocity **v** towards a thin target. From experiment with nuclear physics, it is found that the rate R at which a particular neutron-nucleus reaction occurs is proportional to the neutron flux $\phi$ . That is,

$$R = \phi \bullet \Sigma \quad \text{..... (5.1)}$$

where flux $\phi$ = **n** • **v** = number of neutrons crossing 1 $cm^2$ per sec.

$\Sigma$ = proportionality constant, or defined as the macroscopic cross section for the reaction.

It is the probability of the reaction occurring in 1 cm. of travel of 1 neutron through the reacting medium. It has units - reactions/(neutron.cm).

- *Macroscopic Cross-section:*

The macroscopic cross-section $\Sigma$ is also proportional to nuclei density of the medium; hence

$$\Sigma = N \bullet \sigma \quad \text{.....(5.2)}$$

where N = number of nuclei per cm

$\sigma$ = microscopic cross-section for the medium in $cm^2$.

- *Mean free Path of neutron:*

If $\lambda$ is the average distance a neutron travels before undergoing a certain reaction, then the average number of reactions per sec is $\mathbf{v} / \lambda$, since in 1 second, the neutron would have traveled **v** cm.

With neutron density n, it follows that the rate of neutron reactions

$$R = n \bullet \mathbf{v} / \lambda = \phi / \lambda \quad \text{.....(5.3)}$$

Equating (5.1) and (5.3), we have

$$\lambda = 1 / \Sigma \quad \text{.....(5.4)}$$

Hence $\lambda$ is also known as the mean free path of neutron.

- *Fission Cross-Section:*

Since different reactions occur with different probabilities, they will be associated with different cross-sections:

(a) fission cross-section $\Sigma_f$;

(b) absorption cross-section $\Sigma_a$;

(c) elastic scattering cross-section $\Sigma_s$;

(d) inelastic scattering cross-section $\Sigma_i$.

Since we are primarily dealing with nuclear fission which causes nuclear chain reaction; hence the number of nuclear fissions per unit volume per sec is $\phi \bullet \Sigma_f$.

- *Reactor Power:*
  Given a reactor with fissile material with volume V $cm^3$, fission cross section $\Sigma_f$ (fission reactions/(neutron.cm), neutron flux $\phi$ (neutrons/(cm2.sec)), the number of fissions occurring per sec in such reactor is $V \bullet \Sigma_f \bullet \phi$.

  Note: the average energy released per fission is 200 MeV. Note 1 eV = $1.6 \times 10^{-19}$ watt sec (Joule), so 200 MeV = $3.2 \times 10^{-17}$ watt sec.

  The reactor power P in Megawatts is then

$$3.2 \times 10^{-17} \bullet V \bullet \Sigma_f \bullet \phi \quad \text{.....(5.5)}$$

### 5.1.2 Multiplication Factors for Thermal Reactors

In a critical reactor, one neutron from each fission causes exactly one further fission. It goes through the typical cycle shown below:

Fast neutron born at fission ⟶ Fast neutron is slowed in moderator to become thermalized ⟶ thermal neutron is captured in U-235 to cause fission

Of the (typically) 2.5 neutrons produced at fission, only one goes through this cycle to maintain chain reaction. The other 0.5 are lost by capture or leakage. We start with N thermal neutrons in the reactor, and go round the cycle as follows:

1. A fraction **f** of the thermal neutrons is assumed to be absorbed by the fuel, and the remaining fraction **(1 - f)** is then lost by absorption in material other than fuel. So **f N** neutrons are absorbed by the fuel. The *quantity f is called the thermal utilization factor*. It can be define as the ratio of the number of neutrons absorbed by the fuel to the total number of neutrons absorbed in the reactor.

2. Not all the neutrons absorbed by the fuel will cause fission, because some will be lost by radiative capture. Let a fraction "a" cause fission in U-235, the remaining fraction (1 - a) being lost by radiative capture. Therefore, we get **afN** fissions.

3. Each fission will produce $\nu$ fast neutrons (note for U-235, $\nu = 2.43$). Therefore, **a $\nu$fN** fast neutrons are produced by the original N thermal

neutrons. The product **a** $\nu$ is usually replaced by one symbol $\eta$ so that $\eta$ **fN** fast neutrons are produced by U-235 fissions. *The factor* $\eta$ *is called the thermal fission factor.* It represents the number of fast neutrons produced from each thermal neutron captured in the fuel; whereas $\nu$ represents the number of neutrons produced from each thermal neuron actually causing fission.

4. A small fraction of the fast neutrons produced by these fissions will cause fast fission of U-238 before they are slowed down. The total fast neutrons population will therefore be a slightly greater than $\eta$ **fN**. This additional contribution (typically 2–3 %) is allowed for by increasing the fast neutron population by a factor $\varepsilon$, *commonly known as the fast fission factor.* The total number of fast neutrons produced by both U - 235 and U- 238 fissions is therefore $\varepsilon\ \eta$ **fN.**

5. These fast neutrons (~ 10 MeV) now have to be thermalized (~ 0.025 eV). Some of them escape from the reactor due to leakage before they reach the resonance energies. Let $P_f$ *be the fast neutron non-leakage probability*, then the number of fast neutrons which do not leak from reactor while slowing down is $\varepsilon\ \eta$ **f** $P_f$ **N.**

6. Some of the non-leaked fast neutrons will be lost due to resonance absorption in the moderator while slowing down. Let *p denotes the resonance escape probability*, then the number of fast which have thermalized in the reactor is $\varepsilon\ \eta$ **f** $P_f$ **p N.**

7. Some thermal neutrons will be leaked out from the reactor. Let $P_{th}$ *be the thermal neutrons non-leakage probability*, then the number of thermal neutrons which do not leak from the reactor is

   $$\varepsilon\ \eta\ \mathbf{f}\ P_f\ \mathbf{p}\ P_{th}\ \mathbf{N}$$

   Now these non-leaked thermal neutrons will be absorbed in the fuel again. Each thermal neutron has a probability to produce new generation of fast neutrons again. So we have complete the complete neutron cycle.

- If we define the effective multiplication factor $K_{eff} = \varepsilon\ \eta\ \mathbf{f}\ P_f\ \mathbf{p}\ P_{th}$. The effective multiplication factor $K_{eff}$ can be defined as number of neutrons produced in one generation divided by that of the previous generation.

- For very large reactors, leakage is very small, hence $P_f \sim 1$ and $P_{th} \sim 1$, we get

  $$K_\infty = \varepsilon\ \eta\ \mathbf{f}\ \mathbf{p} \quad \text{.....(5.6)}$$

$K_\infty$ is called the infinite multiplication factor, meaning that it is the factor that would apply if the core in question were made so large that the neutron leakage became negligibly small. Hence, this called the "four-factor formula"

- For $K_{eff} \geq 1, = 1, < 1$, the neutron population increases, is constant, and decreases, respectively; and the reactor is said to be "*Supercritical*", "*Critical*", "*Subcritical*". Note $K_{eff}$ is also called $K_e$ in some literature.

### 5.1.3 Steady State Neutron Balance

We are now in a position to write the neutron balance equation. In a steady state reactor, the rate of change of neutron density is zero. That means the production rate of neutrons is exactly equal to the loss rate of neutrons. In a critical reactor, the production is only from fissions in the fuel, while in a subcritical reactor, one has to take into account any external neutron sources intentionally introduced. The loss of neutrons is primarily due to leakage out of the reactor, as well as absorption in reactor materials such as fuel, moderator, coolant, control rod, etc. Thus mathematically, one can write:

- Time rate of change of neutron density = 0 = neutron production Rate - neutron loss rate.
- Neutron Production Rate = $\phi \bullet \Sigma_f \bullet K_{eff}$ neutrons/(cm$^3$ sec)
- External source = S neutrons/(cm$^3$ sec)
- Loss rate due to leakage: Fick's law states that assuming that scattering is isotropic; neutron flux is a slowly varying function of position; no source in the medium, then neutrons flow from regions of higher densities to lower densities, i.e. the neutron current (number of neutrons crossing unit area in unit time in the normal direction is related to the neutron flux -

$$\bar{J} = -D \cdot grad\phi = -D \cdot \nabla^2 \phi \quad .....(5.7)$$

where D is the diffusion coefficient constant.

- The loss rate due to absorption = $\phi \bullet \Sigma_a$
- Thus the steady state diffusion equation with source is:

$$\frac{dn}{dt} = 0 = K_{eff} \cdot \phi \cdot \Sigma_f + S - (-D \cdot \nabla^2 \phi + \phi \cdot \Sigma_a), \text{ or}$$

$$D \cdot \nabla^2 \varphi - \phi \cdot \Sigma_a + K_{eff} \cdot \phi \cdot \Sigma_f + S = 0 \quad .....(5.8)$$

- The diffusion equation is the basic equation which will be used by reactor engineers in designing the reactor. However, since diffusion theory makes the basic assumption that all neutrons have the same thermal energy, the multiplication factor or critical size calculations are therefore not very accurate. Application of the slowing down theory improves the result somewhat since the neutrons are classified into epithermal (slowing down region) and thermal (diffusion region). Even this two energy group representation might not be adequate for actual design of power plant. In order to circumvent this problem, multigroup diffusion theory equations have been developed. In practice the total neutron energy spread from 10 MeV to thermal energy (0.025 eV) may be divided into 100 groups. To account for leakage and other spatial effects, one has to divide the reactor into some spatial mesh points at which the diffusion equations are solved by applying the boundary conditions. For example, if there are 100 spatial points in each direction, then there are 106 points in the reactor at which one has to solve the diffusion equations. Since we are primarily interested in the dynamic behavior of the reactor, we conclude the steady state design topic discussion here.

### 5.1.4 Mean Neutron Lifetime & Mean Effective Neutron Lifetime

- In an infinite reactor, there is no leakage, so the mean free path $\lambda$ is equal that for absorption, i.e. $\lambda_a = \frac{1}{\Sigma_a}$. Since neutrons travel at v cm per sec., therefore the mean neutron lifetime for infinite reactor is

$$l_0 = \frac{\lambda_a}{v} = \frac{1}{v\Sigma_a} \quad \text{.....(5.9)}$$

- For a finite reactor, the mean neutron lifetime $l = l_0 \cdot P_{th}$, where $P_{th}$ is the thermal neutrons non-leakage probability.

- Since in time $l$, N neutrons are absorbed, and $K_{eff}$ • N neutrons are born, therefore, let us define $\Lambda$ as the mean effective neutron lifetime for each neutron to be born, hence:

$$l \cdot N = \Lambda \cdot K_{eff} \cdot N$$

or $$\Lambda = \frac{l}{K_{eff}} \quad \text{.....(5.10)}$$

- We can see that $l$ is based on the death of neutrons; whereas $\Lambda$ is based on their generation

### 5.1.5 Reactivity Units

- Define Excess Multiplication factor $K_{ex} = K_e - 1$ .....(5.11)

- The reactivity, denoted by $\Delta K$ or $\rho$, is defined in some literature by the equation

$$\rho = \Delta K = \frac{K_e - 1}{K_e} = \frac{K_{ex}}{K_e} \quad \text{.....(5.12)}$$

- Reactivity is normally measured in milli-k, or 0.001k. Example, if k = 1.002, $\Delta K$ = 2 milli-k ( 2 mk). The reactor is then said to be 2 mk supercritical or to have 2 mk positive reactivity. If k =0.995, the reactor is 5 mk subcritical or you can say that it has 5 mk negative reactivity.

### 5.1.6 Changes in Neutron Flux Following a Reactivity Change in Reactor with Prompt Neutrons Only

Consider a reactor to be operating in the steady state with $K_e$ = 1, when reactivity is suddenly increased by an amount $\Delta K$ and then held fixed. This known as step change. Assume that just before the reactivity change was made, the number of neutrons in the reactor was N. They will now multiply themselves from one generation to the next generation with a multiplication constant - the gain in neutrons will N.$(K_e - 1)$, therefore the rate in neutron population will be given by:

$$\frac{dN}{dt} = \frac{(K_e - 1) \cdot N}{l} = \frac{N \cdot \Delta K}{\Lambda} \quad \text{..... (5.13)}$$

The solution to this ordinary differential equation is

$$N = N_0 \cdot e^{\left(\frac{\Delta K}{\Lambda} \cdot t\right)} \quad \text{.....(5.14)}$$

Since neutron flux and reactor power are both proportional to neutron density, and they will have the similar exponential behavior as the neutron population growth equation above.; that is

- Neutron density : $n_t = n_0 \cdot e^{\left(\frac{\Delta K}{\Lambda} \cdot t\right)}$ .....(5.15)
- Neutron flux: $\Phi_t = \Phi_0 \cdot e^{\left(\frac{\Delta K}{\Lambda} \cdot t\right)}$ .....(5.16)
- Neutron power : $P_t = P_0 \cdot e^{\left(\frac{\Delta K}{\Lambda} \cdot t\right)}$ .....(5.17)

### 5.1.7 The Reactor Period for Reactor with Prompt Neutrons Only

If we define the reactor period T as the length of time required for the power (or flux or neutron density) to change by a factor e. That is:

$$T = \frac{\Lambda}{\Delta K} \quad \text{.....(5.18)}$$

The previous equations now become:

$$n_t = n_0 \cdot e^{\left(\frac{t}{T}\right)}$$

$$\Phi_t = \Phi_0 \cdot e^{\left(\frac{t}{T}\right)} \quad \text{.....(5.19)}$$

$$P_t = P_0 \cdot e^{\left(\frac{t}{T}\right)}$$

You can see that the shorter the reactor period, the faster the power changes will be. Conversely, should the reactor period be infinite, the reactor would be critical.

### 5.1.8 Power Changes With Prompt Neutrons Only

- If all the neutrons in the reactor were prompt neutrons, the neutron lifetime, $\Lambda$, would be about 0.001 seconds. The neutron lifetime is largely dictated by the diffusion time in the moderator. Hence the reactor period resulting from a step reactivity change from 0 to +0.5 mk will be given by:

$$T = \frac{\Lambda}{\Delta K} = \frac{0.001}{0.0005} = 2\,\text{sec}$$

In one second, reactor power would increase by a factor of $e^{0.5}$=1.65.

- For a reactivity change of + 5mk (i.e. 10 times increase in reactivity);

$$T = \frac{0.001}{0.005} = 0.2\,\text{sec.}$$

The reactor power increase per second is $e^5$, or 148 times.

- These two examples show how rapid the power increases would be - even for small reactivity changes of a order of a mk - if all the neutrons were prompt neutrons.

### 5.1.9 The Effect of Delayed Neutrons on Power Changes

Fortunately, delayed neutrons are generated by the decay of unstable fission products precursors, of which 6 groups are usually identified. These delayed neutrons, which only represent 0.65 % ($\beta$) of the neutrons generated by fission, prolong the reactor period considerably, and as a result, reactors can be controlled quite easily. If it were not for the presence of these delayed neutrons, controlled nuclear power from fission would not be possible.

The weighted average lifetime of all neutrons, prompt and delayed, is 0.085 seconds. It represents the average length of time for all neutrons generated by fission to be "born". The lifetime of a generation, $\tau$, is then

this weighted average lifetime plus the slowing-down time and the diffusion time in the moderator. The latter is typically 0.001 sec., and so we arrive at $\tau$ = 0.085 + 0.001 = 0.086. Hence, when compared with lifetime of prompt neutrons of 0.001 sec., delayed neutrons increase the average lifetime of all neutrons by 86 times.

Consider the previous example of reactivity increase by 5 mk, the reactor period is

$$T = \frac{\tau}{\Delta K} = \frac{0.086}{0.005} = 17.2$$

The relative power increase per second is $\frac{P_t}{P_0} = e^{\frac{1}{17.2}} = 1.05986$; that is 5.9 % per sec. corresponding to +5 mk increase in reactivity. Therefore, the power increase is much less rapid than with prompt neutrons alone.

Table 3 showing Delayed Neutrons Groups for a Typical HWR.

| **Delayed Neutrons** | **Group** | **Mean Lifetime** | **Yield (%)** |
|---|---|---|---|
| | 6 | 0.33 sec | 0.025 |
| | 5 | 0.88 sec | 0.075 |
| | 4 | 3.32 sec | 0.257 |
| | 3 | 8.98 sec | 0.127 |
| | 2 | 32.78 sec | 0.142 |
| | 1 | 80.40 sec | 0.0215 |
| | | | Total = 0.65 % |

## 5.2 Point Kinetic Model — Delayed Neutrons

Given total m groups of fission precursors, the decay of the $i^{th}$ group precursor is described by the following differential equation:

$$\frac{dC_i}{dt} = \beta_i \cdot K_e \cdot \frac{n}{l} - \lambda_i \cdot C_i \quad i = 1 \ldots\ldots m \quad \ldots\ldots(5.20)$$

where $C_i$ = Concentration of i-th group precursor

$\beta_i$ = Fraction of neutrons in i-th delayed group. Note $\beta = \sum_{i=1}^{m} \beta_i$

$\lambda_i$ = Decay constant of i-th group precursor

$\beta_i K_e \frac{n}{l}$ = Neutrons born to the i-th group precursor

$\lambda_i C_i$ = Rate of decay of the i-th group precursor concentration, or it can also be interpreted as the number of delayed neutrons born per sec due to decay of i-th group precursor.

The neutron differential equation with delayed neutrons is then:

$\frac{dN}{dt}$ = Rate of birth of Prompt Neutrons - Rate of loss of Neutrons + Rate of birth of Delayed Neutrons .....(5.21)

Rate of birth of Prompt Neutrons = $K_e(1-\beta)\frac{n}{l}$ .....(5.22)

Rate of loss of Neutrons = $\frac{N}{l}$ .....(5.23)

Rate of birth of Delayed Neutrons = $\sum_{i=1}^{m} \lambda_i \cdot C_i$ .....(5.24)

Therefore neutron differential equation becomes:

$$\frac{dn}{dt} = (K_e(1-\beta)-1)\cdot\frac{n}{l} + \sum_{i=1}^{m} \lambda_i \cdot C_i \quad \ldots\ldots\ldots(5.25)$$

or $$\frac{dn}{dt} = (K_{ex} - K_e\beta)\cdot\frac{n}{l} + \sum_{i=1}^{m} \lambda_i \cdot C_i \quad \ldots\ldots\ldots(5.26)$$

or $$\frac{dn}{dt} = \frac{\Delta K - \beta}{\Lambda}\cdot n + \sum_{i=1}^{m} \lambda_i \cdot C_i \quad \ldots\ldots\ldots\ldots\ldots(5.27)$$

In summary, the Point Kinetic Model Equations for m groups of delayed precursors are:

$$\frac{dn}{dt} = \frac{\Delta K - \beta}{\Lambda} \cdot n + \sum_{i=1}^{m} \lambda_i \cdot C_i \quad \text{.....(5.28)}$$

$$\frac{dC_i}{dt} = \beta_i \cdot \frac{n}{\Lambda} - \lambda_i \cdot C_i \quad \text{i = 1..... m .....(5.29)}$$

### 5.2.1 Analytical Solution to Point Kinetic Reactor Model Equations

In order to study the dynamics of Point Kinetic Reactor Model, we solve the single group model equation using Laplace Transform, for a step change in reactivity $\Delta K$, from an initial equilibrium of $N_o$, $C_o$ for $t < 0$ ($\Delta K = 0$ for $t < 0$).

Single delayed group model:

$$\frac{dn}{dt} = \frac{\Delta K - \beta}{\Lambda} n + \lambda \cdot C \quad \text{.....(5.30)}$$

$$\frac{dC}{dt} = \frac{\beta}{\Lambda} n - \lambda \cdot C \quad \text{.....(5.31)}$$

Applying Laplace Transform,

$$(s - \frac{\Delta K - \beta}{\Lambda}) \cdot N(s) = \lambda \cdot C(s) + N_0 \quad \text{.....(5.32)}$$

$$(s + \lambda) \cdot C(s) = \frac{\beta}{\Lambda} \cdot N(s) + C_0 \quad \text{.....(5.33)}$$

Rearranging the second equation and substituting expression for C(s) into the first equation, we have:

$$(s - \frac{\Delta K - \beta}{\Lambda} - \frac{\lambda \cdot \beta}{\Lambda(s + \lambda)}) \cdot N(s) = N_0 + \frac{\lambda \cdot C_0}{s + \lambda} = N_0 (1 + \frac{\beta / \Lambda}{s + \lambda}) \quad \text{....(5.34)}$$

because $\frac{dC}{dt} = 0$ for $t < 0$, hence $C_0 = \frac{\beta}{\lambda \cdot \Lambda} N_0$ .....(5.35)

Hence $N(s) = N_0 \dfrac{s + \lambda + \frac{\beta}{\Lambda}}{s^2 + (\lambda - \frac{\Delta K - \beta}{\Lambda})s - \frac{\lambda \cdot \Delta K}{\Lambda}}$ .....(5.36)

The characteristic equation is: $s^2 + (\frac{\lambda \cdot \Lambda - \Delta K + \beta}{\Lambda})s - \frac{\lambda \cdot \Delta K}{\Lambda} = 0$ .....(5.37)

If $\lambda \cdot \Lambda << \beta$, the characteristic equation for single precursor model reduces to:

$$s^2 + (\frac{\beta - \Delta K}{\Lambda})s - \frac{\lambda \cdot \Delta K}{\Lambda} = 0 \quad \text{.....(5.38)}$$

The roots are :

$$s_{1,2} = \frac{1}{2}\frac{(\beta - \Delta K)}{\Lambda}\left[-1 \pm \sqrt{1 + \frac{4\Lambda\lambda\Delta K}{(\beta - \Delta K)^2}}\right] \quad \text{.....(5.39)}$$

and noting that the series approximation:

$$\sqrt{(1+x)} = 1 + \frac{1}{2}x - \frac{1}{2\cdot 4}x^2 + \frac{1\cdot 3}{2\cdot 4\cdot 6}x^3 + \text{.....}$$

We can keep the first two terms as approximation if x <<1. For small $\Delta K$ and the fact that $\lambda \cdot \Lambda << \beta$, this approximation is valid. Hence, the roots equation can be approximated as:

$$s_{1,2} = \frac{1}{2}\frac{(\beta - \Delta K)}{\Lambda}\left[-1 \pm (1 + \frac{2\Lambda\lambda\Delta K}{(\beta - \Delta K)^2})\right] \quad \text{.....(5.40)}$$

that is $s_1 = \dfrac{\lambda\Delta K}{(\beta - \Delta K)}$; .....(5.41)

$$s_2 = -\frac{1}{\Lambda}\left[\frac{(\beta - \Delta K)^2 + \Lambda\lambda\Delta K}{(\beta - \Delta K)}\right] \quad \text{.....(5.42)}$$

$$\approx -\frac{(\beta - \Delta K)}{\Lambda} \text{ since } \lambda \cdot \Lambda << \beta \quad \text{.....(5.43)}$$

By partial fraction expansion, one finds that the approximate solution to the one delayed group precursor neutron differential equation:

$$N = N_0\left[\frac{\beta}{\beta - \Delta K}e^{\frac{\lambda\Delta K}{(\beta - \Delta K)}t} - \frac{\Delta K}{\beta - \Delta K}e^{-\frac{(\beta - \Delta K)}{\Lambda}t}\right] \quad \text{.....(5.44)}$$

The roots $s_1, s_2$ depend on $\Delta K$. For $\Delta K$ to be positive, one root is positive (growing exponentially); the other root negative; whereas for $\Delta K$ to be negative, both roots become negative. This has found to be true for multiple precursor case; for $\Delta K$ positive, one root is positive and all other roots negative; for $\Delta K$ negative, all roots negative, and after a while, the slowest delayed neutron group dominates the response.

Returning the subject matter of "Reactor Period", for prompt neutron model, the reactor period T is

$$T_P = \frac{\Lambda}{\Delta K} \quad \text{.....(5.45)}$$

For the delayed neutron model, for positive $\Delta K$,

Case (1) if $\Delta K << \beta$, the reactor period is dominated by first exponential term :

$$T = \frac{(\beta - \Delta K)}{\lambda \Delta K} \approx \frac{1}{\Delta K} \frac{\beta}{\lambda} \quad \text{.....(5.46)}$$

Plugging in some numbers: $\Delta K$ =0.003; $\beta$ = 0.0065; $\lambda$ =0.077

T = 28 sec.

This is intuitive because in our approximation, we assume $\lambda \cdot \Lambda << \beta$. Dividing both side of this inequality by $\lambda \Delta K$, we have

$$\frac{\Lambda}{\Delta K} << \frac{1}{\Delta K} \frac{\beta}{\lambda} \quad \text{.....(5.47)}$$

That is $T_p << T$, as we would expect.

For multiple precursors model, the reactor period is given by:

$$T = \frac{1}{\Delta K} \sum_{i=1}^{m} \frac{\beta_i}{\lambda_i} \quad \text{.....(5.48)}$$

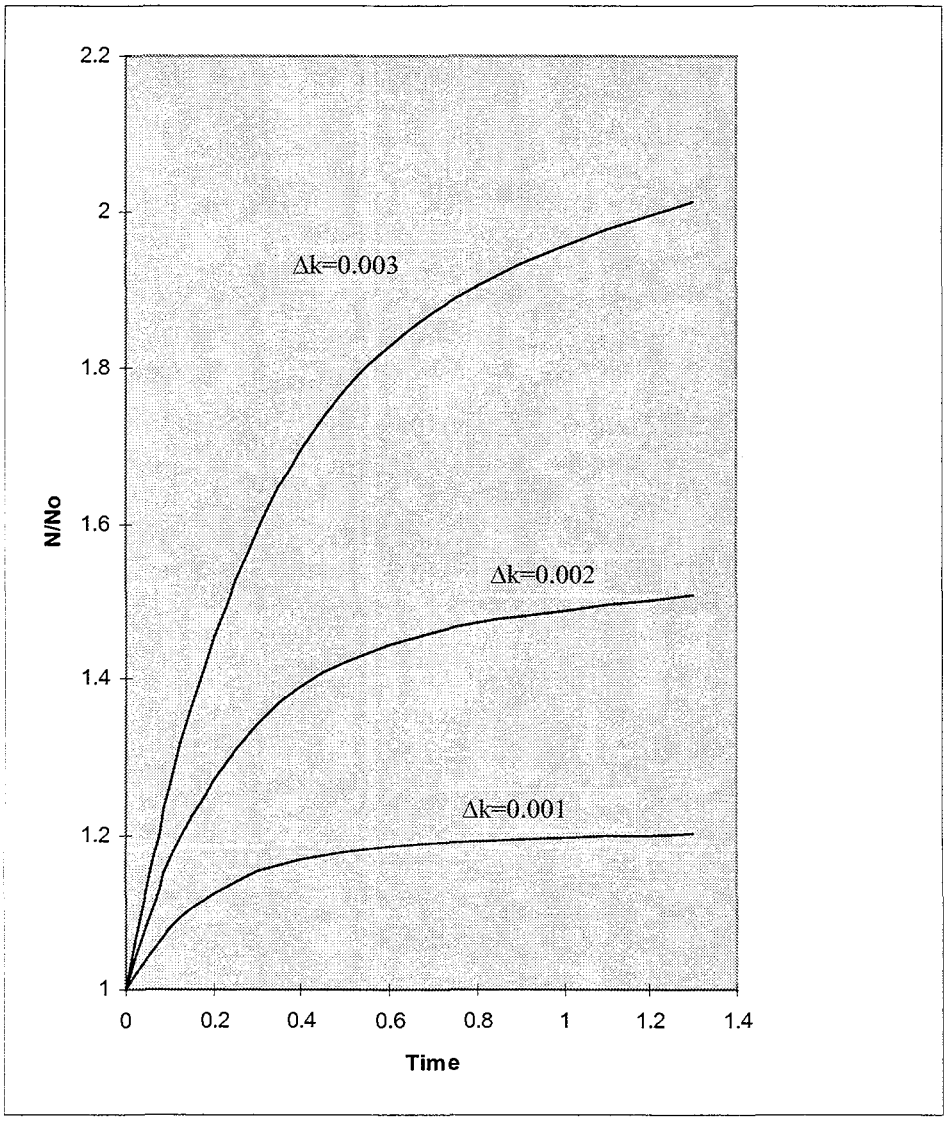
Case (2) if $\Delta K >> \beta$, the reactor period is dominated by second exponential term:

$$T = \frac{\Lambda}{(\beta - \Delta K)} \approx \frac{\Lambda}{\Delta K} \quad \text{.....(5.49)}$$

This is the same as the prompt neutron model. This implies that if $\Delta K >> \beta$, the beneficial effect of the delayed neutrons has been lost, and prompt neutrons dominate the response.

Some single precursor model responses for different constant values of $\Delta K$ are shown in figure below by substituting value of $\beta$ = 0.0064; $\lambda$ =0.076, $\Lambda$ =0.001 and for $\Delta K$ = 0.001, 0.002, 0.003 into equation:

$$N = N_0 \left[ \frac{\beta}{\beta - \Delta K} e^{\frac{\lambda \Delta K}{(\beta - \Delta K)} t} - \frac{\Delta K}{\beta - \Delta K} e^{-\frac{(\beta - \Delta K)}{\Lambda} t} \right]$$




The behavior of the single precursor model can be explained as follows:

The number of delayed neutrons contributed to the chain reaction at any instant is proportional to the thermal flux at some *previous* time, but the number of precursors being produced by fission (i.e. the number of neutrons being delayed) is proportional to the flux at the *given* instant. As soon as reactivity is increased above unity, the fission rate will increase, and prompt neutrons will immediately be generated from the prompt *neutrons and delayed neutrons already there.* Hence, for a very short time, the reactor behaves as if all the neutrons were prompt, and the flux increases very rapidly. This is known as "**Prompt Rise**". When the very short lived neutron precursors formed *before the reactivity change* have all decayed (in this case 0.5 seconds), this delayed neutron deficiency must be compensated by the neutrons delayed *after the reactivity change.* Therefore, the rate of flux increase gradually falls off as it has to wait for these delayed neutrons, and ultimately the rate of flux growth will be governed entirely by the new delayed neutrons. The net result is that after a step change in reactivity, there will be a sudden fast prompt rise followed by the more leisurely growth in neutron flux (or reactor power).

**Dollars**: one dollar is defined as the magnitude of $\Delta K$ which is equal to $\beta$. Thus for Uranium 235 fueled reactor, one dollar is equal to 0.0065,or +6.5 mK. When the reactivity is one dollar, the reactor is critical on prompt neutrons alone (since dn/dt = 0; if $\Delta K = \beta$). Normally reactor operates near delayed critical with $\Delta K$ near zero.

**In-Hours**: One in-hour corresponds to that magnitude of $\Delta K$ which produces a reactor period of one hour. For example, $\beta$ = 0.0065; $\lambda$ =0.077, for reactor period T = 3600 seconds,

$$\Delta K \approx \frac{1}{T}\frac{\beta}{\lambda} = \frac{0.0065}{3600 \cdot 0.077} = 0.0000234 = 0.023mK$$

Note: for inherent safety of the reactor, if the period is smaller than 10 seconds, the reactor control is designed to automatically shutdown the reactor.

## 5.3 Modeling Exercise - Simple Point Kinetic Model

### 5.3.1 Problem Description:

Create a simple point kinetic model for a PHWR type reactor core.

The following equations described the reactor kinetics assuming a point kinetic model:

$$\frac{dN_{FLUX}}{dt} = \frac{(\Delta k - \sum_{i=1}^{6} \beta_i) * N_{FLUX}}{N_{LIFE}} + \sum_{i=1}^{6} \lambda_i * C_i$$

$$\frac{dC_i}{dt} = \frac{\beta_i * N_{FLUX}}{N_{LIFE}} - \lambda_i * C_i$$

**Data:**

NLIFE = 0.0008 sec

b1 = 0.000329

b2 = 0.001003

b3 = 0.000856

b4 = 0.002319

b5 = 0.000587

b6 = 0.000206

l1 = 0.01261 $sec^{-1}$

l2 = 0.03051 $sec^{-1}$

l3 = 0.12 $sec^{-1}$

l4 = 0.3209 $sec^{-1}$

l5 = 1.262 $sec^{-1}$

l6 = 3.239 $sec^{-1}$

NFLUX = neutron Flux(normalized from 0 to 1)

Ci = concentration of delayed neutron groups(normalized)(i=1,6)

Dk = overall reactivity change (K)

bi = delay neutron fraction of delay group i(normalized)(i=1,6)

NLIFE = mean neutron lifetime(sec)

li = decay constant of delayed neutron groups(/sec)

### 5.3.2 Point Kinetic Reactor Simulation Model using CASSIM

1. Create a custom FORTRAN algorithm using the above equations with overall reactivity change Dk as input, and neutron flux $N_{FLUX}$ and Delayed neutron group precursor, $C_i$ as outputs.

2. Remember to follow the CASSIM release notes(document #1) guidelines for calculations involving double precision numbers.

3. Use Euler's backward difference(implicit method) to model the above equations. Try to reduce the two equations to one equation by eliminating the time dependent variable Ci. Why is it best to use the implicit Euler's method?

4. Identify the necessary outputs, and coefficients

5. Use CASSBASE to overwrite ALG802, and replace it with your algorithm. Remember to define your inputs and outputs for the template in CASSBASE, and save your FORTRAN algorithm as ALG802.FOR.

6. Exit CASSBASE, and go to SRCLIB directory, and recreate CASSIM.DLL with the new ALG802 using the ADDTOLIB batch file.

7. Copy the new CASSIM.DLL to the CASSENG directory.

8. Use CASSBASE to create a new model and call it IAEA1.

9. Create a NMB block(ALG452) for communication with LABVIEW.

10. Create a block for calculating the reactor neutron flux using the custom algorithm ALG802. Remember to initialize the neutron concentrations by depositing their values in the output section.

11. A LABVIEW executable, 'TRENDS.EXE is provided for you to trend 8 variables that is connected as input 1 to 8 to block NMB. Trend the neutron flux as calculated from the reactor block.

12. Try to maneuver the reactor power to 80% by depositing the liquid zone reactivity change to no more that -0.1mK.

13. Shutdown the reactor with large negative reactivity, and observe the reactor power decay.

### 5.3.3 Simple Point Kinetic Model Algorithm Formulation (Solution)

The following equations described the reactor kinetics assuming a point kinetic model:

$$\frac{dN_{FLUX}}{dt} = \frac{(\Delta k - \sum_{i=1}^{6} \beta_i) * N_{FLUX}}{N_{LIFE}} + \sum_{i=1}^{6} \lambda_i * C_i$$

$$\frac{dC_i}{dt} = \frac{\beta_i * N_{FLUX}}{N_{LIFE}} - \lambda_i * C_i$$

NFLUX = neutron Flux(normalized from 0 to 1)

Ci = concentration of delayed neutron groups(normalized)(i=1,6)

Dk = overall reactivity change (K)

bi = neutron flux(normalized from 0 to 1)

NLIFE = mean neutron lifetime(sec)

li = decay constant of delayed neutron groups(/sec)

Use Backward Euler's to approximate the above equations:

1) delay neutron concentration

$$\frac{dC_i}{dt} = \frac{\beta_i * N_{FLUX}}{N_{LIFE}} - \lambda_i * C_i$$

$$\frac{C_i - \overline{C_i}}{\Delta t} = \frac{\beta_i * N_{FLUX}}{N_{LIFE}} - \lambda_i * C_i$$

re - arranging $C_i$

$$C_i * (1 + \lambda_i * \Delta t) = \frac{\beta_i * N_{FLUX} * \Delta t}{N_{LIFE}} + \overline{C_i}$$

$$Ci = \frac{\frac{\beta_i * N_{FLUX} * \Delta t}{N_{LIFE}} + \overline{C_i}}{(1 + \lambda_i * \Delta t)}$$

2) neutron flux calculation

$$\frac{dN_{FLUX}}{dt} = \frac{(\Delta k - \sum_{i=1}^{6} \beta_i) * N_{FLUX}}{N_{LIFE}} + \sum_{i=1}^{6} \lambda_i * C_i$$

$$\frac{N_{FLUX} - \overline{N}_{FLUX}}{\Delta t} = \frac{(\Delta K - \Sigma\beta_i) \cdot N_{FLUX}}{N_{LIFE}} + \Sigma\lambda_i \cdot C_i$$

Substituting for Ci,

$$\frac{N_{FLUX} - \overline{N}_{FLUX}}{\Delta t} = \frac{(\Delta K - \sum_{i=1}^{6} \beta_i) \cdot N_{FLUX}}{N_{LIFE}} + \Sigma\lambda_i \cdot \left[ \frac{\frac{\beta_i \cdot N_{FLUX} \cdot \Delta t}{N_{LIFE}} + \overline{C_i}}{(1 + \lambda_i \cdot \Delta t)} \right]$$

Multiply both sides by $\Delta t$,

$$N_{FLUX} - \overline{N}_{FLUX} = \frac{(\Delta K - \sum_{i=1}^{6} \beta_i) \cdot N_{FLUX} \cdot \Delta t}{N_{LIFE}} + \Sigma \lambda_{i=1}^{6} \cdot \Delta t \cdot \left[ \frac{\frac{\beta_i \cdot N_{FLUX} \cdot \Delta t}{N_{LIFE}} + \overline{C_i}}{(1 + \lambda_i \cdot \Delta t)} \right]$$

defining Ai and Bi

$$Ai = \frac{\lambda_i * \Delta t * \overline{Ci}}{(1 + \lambda_i * \Delta t)}$$

$$Bi = \frac{\lambda_i * \beta_i * \Delta t}{(1 + \lambda_i * \Delta t) * N_{LIFE}}$$

The equation reduces to:

$$N_{FLUX} - \overline{N_{FLUX}} = \frac{(\Delta k - \sum_{i=1}^{6} \beta_i) * N_{FLUX} * \Delta t}{N_{LIFE}} + \sum_{i=1}^{6} (Bi * N_{FLUX} * \Delta t) + Ai$$

re – arranging for $N_{FLUX}$:

$$N_{FLUX} * 1 - \frac{(\Delta k - \sum_{i=1}^{6} \beta_i) * \Delta t}{N_{LIFE}} - \sum_{i=1}^{6} (Bi * \Delta t) = \overline{N_{FLUX}} + \sum_{i=1}^{6} Ai$$

$$N_{FLUX} = \frac{\overline{N_{FLUX}} + \sum_{i=1}^{6} Ai}{1 - \Delta t * \frac{(\Delta k - \sum_{i=1}^{6} \beta_i)}{N_{LIFE}} + \sum_{i=1}^{6} (Bi)}$$

### 5.3.4 Simple Point Kinetic Reactor Model FORTRAN Program (Solution)

```
      SUBROUTINE ALG802(INP,OUT,COF,DT)
C
C ALGORITHM NUMBER: 802
C
```

```
C
C ALGORITHM NAME: SIMPLE POINT REACTOR KINETICS &
C REACTIVITY
C
C     AUTHOR       DATE
C
C-------------------------------------------------------
C
C THIS ALGORITHM CALCULATES CANDU REACTOR KINETICS
&
C REACTIVITY
C
C-------------------------------------------------------
C
C INPUTS: 2
C
C INP(1) = DKXE, reactivity change due to xenon(K)
C INP(2) = DKLZ, reactivity change due to liquid zone(K)
C-------------------------------------------------------
C
C OUTPUTS: 9
C OUT(1) = NFLUX, neutron flux (normalized)
C OUT(2) = DK, overall reactivity change (K)
C OUT(3) = BETA1, Total delayed neutron fraction
C OUT(4) = C(1), neutron concentration, group 1(normalized)
C OUT(5) = C(2), neutron concentration, group 2(normalized)
C OUT(6) = C(3), neutron concentration, group 3(normalized)
C OUT(7) = C(4), neutron concentration, group 4(normalized)
C OUT(8) = C(5), neutron concentration, group 5(normalized)
C OUT(9) = C(6), neutron concentration, group 6(normalized)
C-------------------------------------------------------
C
C COEFFICIENTS: 13
C COF(1) = TNEUTRON, mean neutron lifetime(sec)
C COF(2) = BETA(1), delayed neutron fraction, group 1(normalized)
C COF(3) = BETA(2), delayed neutron fraction, group 2(normalized)
C COF(4) = BETA(3), delayed neutron fraction, group 3(normalized)
C COF(5) = BETA(4), delayed neutron fraction, group 4(normalized)
C COF(6) = BETA(5), delayed neutron fraction, group 5(normalized)
C COF(7) = BETA(6), delayed neutron fraction, group 6(normalized)
C COF(8) = LAMDA(1), delayed neutron decay constant, group 1 (1/sec)
C COF(9) = LAMDA(2), delayed neutron decay constant, group 2 (1/sec)
C COF(10) = LAMDA(3), delayed neutron decay constant, group 3 (1/sec)
C COF(11) = LAMDA(4), delayed neutron decay constant, group 4 (1/sec)
C COF(12) = LAMDA(5), delayed neutron decay constant, group 5 (1/sec)
C COF(13) = LAMDA(6), delayed neutron decay constant, group 6 (1/sec)
C-------------------------------------------------------
C
C PROGRAM DECLARATIONS
```

```
C
      IMPLICIT NONE
C
      INTEGER I
      DOUBLE PRECISION INP(*), OUT(*), COF(*), DT
C
      DOUBLE PRECISION DKXE,DKLZ
      DOUBLE PRECISION NFLUX,DK,C(6), BETA1
      DOUBLE PRECISION BETA(6),LAMDA(6), TNEUTRON
      DOUBLE PRECISION A(6), B(6)
C
C INPUTS, OUTPUTS, AND COFS SETUP
C
      DKXE  = INP(1)
      DKLZ = INP(2)

C
      NFLUX = OUT(1)
      DK   = OUT(2)
      BETA1 = OUT(3)
      C(1)  = OUT(4)
      C(2)  = OUT(5)
      C(3)  = OUT(6)
      C(4)  = OUT(7)
      C(5)  = OUT(8)
      C(6)  = OUT(9)
C
      TNEUTRON = COF(1)
      BETA(1) = COF(2)
      BETA(2) = COF(3)
      BETA(3) = COF(4)
      BETA(4) = COF(5)
      BETA(5) = COF(6)
      BETA(6) = COF(7)
      LAMDA(1)= COF(8)
      LAMDA(2)= COF(9)
      LAMDA(3)= COF(10)
      LAMDA(4)= COF(11)
      LAMDA(5)= COF(12)
      LAMDA(6)= COF(13)

C
C-------- TOTAL REACTIVITY CHANGE (K)
C
       DK = DKLZ + DKXE
C
C
C-------- TOTAL DELAYED NEUTRON FRACTION
C
```

```
      BETA1 = BETA(1) + BETA(2) + BETA(3) +
     &    BETA(4) + BETA(5) + BETA(6)
C
C--------DEFINE Ai,
C
      DO 1 I = 1, 6
      A(I) = LAMDA(I) * C(I) * DT / ( 1D0 + DT * LAMDA(I))
 1    CONTINUE
C
C--------DEFINE Bi
C
      DO 2 I = 1, 6
      B(I) = LAMDA(I) * DT * BETA(I) /
     &     (TNEUTRON * ( 1D0 + DT * LAMDA(I)))
 2    CONTINUE
C
C-------- NEUTRON FLUX CALCULATION (NORMALIZED)
C
     NFLUX =DMAX1(((NFLUX + A(1) + A(2) + A(3) + A(4) +
     &    A(5) + A(6))   /
     &    (1D0 - DT * ((DK -BETA1)/TNEUTRON +
     &    B(1) + B(2) + B(3) + B(4) + B(5) + B(6)))),0.0D0)
C
C--------DELAYED GROUPS CONCENTRATION
C
     DO 3 I = 1, 6
     C(I) = (C(I) + (DT * BETA(I) * NFLUX/TNEUTRON)) /
     &     ( 1D0 + DT * LAMDA(I))
 3   CONTINUE
C

     OUT(1) = NFLUX
     OUT(2) = DK
     OUT(3) = BETA1
     OUT(4) = C(1)
     OUT(5) = C(2)
     OUT(6) = C(3)
     OUT(7) = C(4)
     OUT(8) = C(5)
     OUT(9) = C(6)
C
      RETURN
      END
```

## 5.4 The Effects of Neutron Sources on Reactor Kinetics

The point kinetic reactor model equations in the previous chapter are all based on the assumption that there are no sources of neutrons in the reactor other than the (induced) fission. This assumption is not entirely true. In this chapter, we shall consider the effects that neutrons sources other than fission have on power changes.

### 5.4.1 The Neutron Sources

- In general there will always be a source of neutrons present in the reactor, if only because of the spontaneous fissions which take place. The spontaneous decay rate of 1 gram of natural uranium is about 7 x $10^{-3}$ fissions per second. This is very small, but significant.

- During reactor startup, it may expedient to introduce an artificial sources of neutrons into the reactor for the initial startup. This will be discussed later.

- Finally, photoneutrons may be produced by the absorption of fission product gamma photons in reactor materials. This is particularly relevant in reactors containing heavy water, where gamma reactions with deuterium are possible for gamma energies in excess of 2.21 MeV ( i.e. the binding energy of the deuterium nucleus); namely,

$$ {}_1H^2 + \gamma = {}_1H^1 + {}_0n^1 \quad \text{..... (5.50)} $$

This photoneutrons source will persist after the reactor has been shutdown and the prompt and delayed neutrons have died down.

### 5.4.2 The Effect of Neutron Sources on the Total Neutron Population

- In a critical reactor with no neutron sources other than induced fission, the neutron population in the reactor remains constant from one generation to the next. In other words, neutron losses due to absorption and leakage exactly take care of the excess neutrons generated by fission that are not required to keep the chain reaction going. Now imagine a neutron emitting source $S_o$ neutrons per second to be inserted into the reactor. These neutrons will be multiplied by the factor $K_e$ from one generation to the next. Therefore at a time t seconds after the insertion, neutrons of various generations up to the $\frac{t}{\Lambda}$ will be present.

- Let us iterate the neutrons population growth. The number of source neutrons at the end of the first generation will be $S_0 \cdot \Lambda$, since $S_0$ are produced per second. At the end of the second generation, these will have $S_0 \cdot \Lambda \cdot K_e$ and another $S_0 \cdot \Lambda$ new one will have been added by the source. At the end of the third generation, there will be a total of $S_3$, given by

$$S_3 = (S_0\Lambda K_e + S_0\Lambda)K_e + S_0\Lambda = S_0\Lambda(1 + K_e + K_e^2) \ldots\ldots\ldots (5.51)$$

Hence after $m^{th}$ generation, the total neutron population due to the source will be:

$$S_m = S_0\Lambda(1 + K_e + K_e^2 + \ldots\ldots\ldots + K_e^{m-1}) \ldots\ldots\ldots\ldots (5.52)$$

or, $$S_m = \frac{S_0\Lambda(1 - K_e^m)}{(1 - K_e)} \ldots\ldots\ldots\ldots\ldots (5.53)$$

since $m = \frac{t}{\Lambda}$, therefore the number of source neutrons at time t is given by:

$$S_t = \frac{S_0\Lambda(1 - K_e^{\frac{t}{\Lambda}})}{(1 - K_e)} \ldots\ldots\ldots\ldots\ldots (5.54)$$

- Let us now look at three cases:

Case (1) $K_e < 1$: As t approaches infinity, $S_t$ will reach an equilibrium value given by :

$$S_t = \frac{S_0\Lambda}{(1 - K_e)} \ldots\ldots\ldots\ldots\ldots (5.55)$$

Case (2) $K_e = 1$: In this case, $S_0\Lambda$ neutrons will be added to the neutron population in every generation, so that after a time t equivalent to $m = \frac{t}{\Lambda}$ generations, the number of source neutrons will be:

$$S_t = S_0\Lambda\frac{t}{\Lambda} = S_0 t \ldots\ldots\ldots\ldots (5.56)$$

Case (3) $K_e > 1$: For a supercritical reactor, the neutron population will increase very rapidly. The fission neutrons already there will increase at rate governed by the reactor period and the source neutrons will multiply as given by equation (5.54)

### 5.4.3 The Effect of Neutrons Sources on a Reactor at Power

Let us consider the implications of the neutron sources by using a typical heavy water reactor as an example.

- When the reactor is at power, photoneutrons will be produced by $(\gamma, n)$ reactions with deuterium nuclei. The gamma source is made up of fission gammas, capture gammas and fission product and activation product decay gammas. The photoneutrons flux will typically be of the order of $10^{-6}$ of the thermal flux. If reactor power is held constant, the

total flux will be constant, so will the neutron density. Suppose there are n neutrons in any generation, they will related by :

$$nK_e + 10^{-6}n = n \quad \text{.....(5.55)}$$

- That means $K_e = 1 - 10^{-6} = 0.999999$. It should be interesting to note that when $K_e$ is exactly unity, we should expect the neutron population to grow by a factor of $10^{-6}$ per generation. *Hence the effect of the photoneutron source on critical reactor operation at power is therefore seen to be quite negligible.*

### 5.4.4 The Effect of Neutron Source on Reactor Shut-Down

Let us now consider what happens when the reactor, which has been running at power, is suddenly made subcritical. In the *absence* of neutron source, the reactor power would diminish to zero at a rate ultimately governed by the longest lived delayed neutron group. However, with a neutron source present, the neutron population will never decrease below the equilibrium value

$$S_t = \frac{S_0 \Lambda}{(1 - K_e)} \quad \text{.....(5.55)}$$

We can therefore say that the final power level, $P_\infty$ , will be given by:

$$P_\infty = \frac{P_s}{1 - K_e} \quad \text{.....(5.56)}$$

where $P_s$ is the neutron source strength, expressed in watts. It is the fission power produced in a reactor by a source of $S_0$ neutrons per second. It should be noted that the final power level of the reactor depends on the amount of negative reactivity inserted. For example, if $P_s$ = 30 W, and $\Delta K$=-30 mk, then $P_\infty = \frac{30}{0.03} = 1000W$

### 5.4.5 Photoneutrons Sources in Heavy Water Reactor

There are 9 photoneutron groups, analogous to the 6 delayed neutron groups, categorized according to their respective half-lives. They are listed in Table 4. The photoneutrons constitute about 4 % of the delayed neutron production, and that shortly after shutdown, the photoneutrons yield is decreasing rapidly. In fact after a couple of days or so, the photoneutron source will be down by the order of a thousand or so, and only the last group of photoneutrons will be effective.

They will decay with a half life of 18 days and will therefore set limit on the length of time during which the reactor may still be restarted without special instrumentation.

Table 4 - Photoneutron Groups for a typical HWR.

| **Photoneutrons** | 7 | 3.6 sec | $19 \times 10^{-3}$ |
|---|---|---|---|
| | 8 | 57 sec | $5.7 \times 10^{-3}$ |
| | 9 | 3 minutes | $1.3 \times 10^{-3}$ |
| | 10 | 18 minutes | $1.11 \times 10^{-3}$ |
| | 11 | 40 minutes | $0.61 \times 10^{-3}$ |
| | 12 | 2.5 hours | $0.64 \times 10^{-3}$ |
| | 13 | 6.3 hours | $0.08 \times 10^{-3}$ |
| | 14 | 2.2 days | $0.02 \times 10^{-3}$ |
| | 15 | 18.5 days | $0.02 \times 10^{-3}$ |
| | | | Total=$28 \times 10^{-3}$ % |

**5.5 Modeling Exercise - Point Kinetic Reactor Model with Photoneutrons.**

The modeling equations should be changed to:

$$\frac{dn}{dt} = \frac{\Delta K - \sum_{i=1}^{15} \beta_i}{\Lambda} \cdot n + \sum_{i=1}^{15} \lambda_i \cdot C_i$$

$$\frac{dC_i}{dt} = \beta_i \cdot \frac{n}{\Lambda} - \lambda_i \cdot C_i \quad \text{for i = 1, ... 15}$$

Note that i = 1 to 6 belong to delayed neutron groups; whereas i = 7 to 15 belong to photoneutron groups.

**Data:**

NLIFE = 0.0008 sec

b1 = 0.000329 Delayed Neutron Group Fraction

b2 = 0.001003 "

b3 = 0.000856 "

b4 = 0.002319 "

b5 = 0.000587 "

b6 = 0.000206 "

b7 = 0.00019 Photoneutron Group Fraction

b8 = 0.000057 "

b9 = 0.000013 “

b10 = 0.0000111 “

b11 = 0.0000061 “

b12 = 0.0000016 “

b13 = 0.0000008 “

b14 = 0.0000002 “

b15 = 0.0000002 “

l1 = 0.01261 $sec^{-1}$ Delayed Neutron Group Decay Constant

l2 = 0.03051 $sec^{-1}$ “

l3 = 0.12 $sec^{-1}$ “

l4 = 0.3209 $sec^{-1}$ “

l5 = 1.262 $sec^{-1}$ “

l6 = 3.239 $sec^{-1}$ “

l7 = 0.2778 $sec^{-1}$ Photoneutron Group Decay Constant

l8 = 0.01754 $sec^{-1}$ “

l9 = 0.00556 $sec^{-1}$ “

l10 = 0.000926 $sec^{-1}$ “

l11 = 0.0004167 $sec^{-1}$ “

l12 = 0.0001111 $sec^{-1}$ “

l13 = 0.0000441 $sec^{-1}$ “

l14 = 0.00000526 $sec^{-1}$ “

l15 = 0.00000063 $sec^{-1}$ “

NFLUX = neutron Flux(normalized from 0 to 1)

Ci = concentration of neutron groups(normalized)( delayed neutron group i=1,6 ; Photoneutron Group i = 7, 15)

Dk = overall reactivity change (K)

bi = neutron fraction (normalized) of delay group (i =1,6);
photoneutron group (i= 7, 15)

NLIFE = mean neutron lifetime(sec)

li = decay constant of neutron groups(/sec); Delayed group i = 1 to 6 ;
Photoneutron group i = 7 to15

1. Modify the Algorithm # 802 to incorporate these changes. Be sure to include additional inputs, outputs and coefficients

2. Shutdown reactor power with insertion of large negative reactivity. Observe reactor power decay transient.

### 5.5.1 Point Kinetic Reactor Model with Photoneutrons Sources (Solution)

To be included in the class

## 5.6 Reactivity Feedback Effects

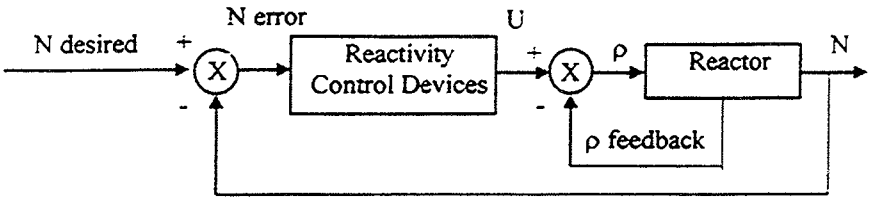



Figure 9 - control block diagram for reactor reactivity control

In a reactor control system such as in Figure 9, the power error ( neutron density error) is used to adjust the control reactivity U by adjustments of the reactivity control devices. The actual reactivity $\rho\ (=\Delta K)$ is the algebraic sum of U an the inherent reactivity feedback effects due to the changes of the system variables affecting reactivity. These reactivity feedback effects can be classified into three categories:

1. Long term effects which consist of fuel burn up, conversion of fertile material into fuel material (build up), and the accumulation of fission products. Time scale is in months.

2. Intermediate effects where the time scale is in the order of several hours consist of reactivity changes due to xenon and samarium poisons.

3. Short term effects which arise due to temperature changes in fuel, heat transport fluid (void) and moderators, The time scale for these short term effects are of the order of minutes.

Let us discus each of these effects briefly.

### 5.6.1 Long Term Reactivity Effects - Fuel Burnup & Buildup

- The composition of the fuel will change quite significantly during its life in the reactor. There are two predominant effects - the burn up of fissile U-235 and the conversion of non-fissile U-238 into fissile Plutonium.

- The rate at which this occurs depends on the neutron flux, because the rate dC/dt at which U-235 is destroyed by neutron capture is given by :

$$\frac{dC}{dt} = -C\sigma_a\phi \quad \text{.....(5.57)}$$

where C = the concentration of U-235

$\sigma_a$ = absorption cross section for U-235

$\phi$ = neutron flux

- This depletion of U-235 is offset by the conversion of U-238 into Pu-239:

$$U^{238} + n \rightarrow U^{239} + \gamma$$

$$\downarrow$$

$$_{93}Np^{239}$$

$$\downarrow$$

$$_{94}Pu^{239}$$

- The conversion ratio is the number of Pu-239 nuclei formed for each U-235 nuclei consumed. In typical CANDU Heavy Water reactor, it is about 0.8.

- The Pu-239 that is produced will eventually build up to equilibrium when its rate of production will be equal to its rate of removal. The Pu-240 formed by neutron capture has properties very similar to U-238, but if it captures another neutron, it will form fissile Pu-241. Therefore after a long period of reactor operation, power will be produced by U-235, Pu-239 and Pu 241.

- The rate of change in the concentration Ci of the ith isotope in the fuel in a flux of $\phi$ is in general given by:

$$\frac{dC_i}{dt} = C_{i-1}\sigma_{i-1}\alpha_{i-1}\phi + C_j\lambda_j - C_i\sigma_i\phi - C_i\lambda_i \quad \text{.....(5.58)}$$

where $\sigma_i$ is the absorption cross-section,

$\lambda_i$ is the radioactive decay constant

$\alpha_i$ is the probability of an absorbed neutrons producing a radiative capture $(n,\gamma)$ event.

The first term $C_{i-1}\sigma_{i-1}\alpha_{i-1}\phi$ is the production of the $i^{th}$ isotope by neutron capture in the $(i-1)^{th}$ isotope.

The second term $C_j\lambda_j$ is the production of i$^{th}$ isotope from radioactive decay of a parent j$^{th}$ nuclide.

The last two terms are the losses due to neutron absorption and radioactive decay of the ith isotope respectively.

- The composition of the reactor fuel after a given time t can be obtained by solving the set of simultaneous equations (5.58) and inserting as a boundary condition the initial fuel composition at t = 0. This change in fuel composition will affect both thermal fission factor $\eta$ , and the thermal utilization factor, $f$ , and hence $\eta f$ . This will be given by:

$$\eta f = \frac{C_5 \cdot \sigma \cdot f_5 \cdot \nu_5 + C_9 \cdot \sigma \cdot f_9 \cdot \nu_9 + C_1 \cdot \sigma \cdot f_1 \cdot \nu_1}{C_5\sigma_5 + C_8\sigma_8 + C_9\sigma_9 + C_0\sigma_0 + C_1\sigma_1 + \frac{\bar{\phi}_m}{\bar{\phi}_f}(C_m\sigma_m)} \quad \text{.....(5.59)}$$

Here the subscript 5 and 8 refer to U-235 and U-238, and subscript 9, 0 and 1 refer to Pu-239, Pu-240 and Pu-241. The absorption in materials with concentration Cm in the fuel other than fuel isotopes is included in the last term of the denominator.

- In general, it has been found that the changes in the composition of the fuel will not change the value of the fast fission factor $\varepsilon$, or the resonance capture probability p, since both of these are dependent on the U-238 concentration in the fuel, and this will not change significantly with time.

- Thus the change in reactivity after a specified burnup can be determined by solving equations (5.58) and plugging the answers into equation (5.59) to obtain new value of $\eta f$, and hence the new value of reactivity. Numerous computer programs exist to carry out such burnup calculations. But general observations of the burnup are as follows:

  1. Initially the burnup of U-235 and its replacement by a smaller number of Pu-239 (conversion factor 0.8) leads to an increase in reactivity. This is due to higher cross section of Pu-239 relative to U-235, leading to higher value of f.

  2. At higher irradiations, the U-235 is still being removed, but the buildup of Pu-239 becomes less rapid since it will eventually reach an equilibrium level when the production of Pu-239 will equal the removal due to absorption. Consequently, at higher irradiations the reduction in the number of fissile nuclei causes a reduction on reactivity.

- Previously, we saw that the small fraction of delayed neutrons substantially increases the reactor period for a given reactivity. The delayed neutrons yields from U-235 are such that the generation

lifetimes are increased by a factor of 85. However, when fuel are burnt, Pu-239 are formed. Plutonium-239 has lower delayed neutron yields, and the effect is to reduce the mean lifetime and so make reactor behavior more dynamic. For example, for equilibrium fuel with roughly equal U-235 and U-239 content, the neutron generation lifetime will be around 0.06 seconds. A + 1mk insertion will therefore cause a more rapid power increase for equilibrium fuel than for fresh fuel. Therefore, in the design of the reactor control system, an allowance must be made for the decrease in reactor period as the plutonium concentration increases.

- The burnup of the fuel will produce change in the neutron flux distribution across the reactor. This occurs because the burnup at any point in the core is proportional to the flux at that point. Since the flux is higher in the central region of the reactor, the burnup proceeds at a greater rate there. The gradual reduction of reactivity in these regions will cause the flux to be depressed there relative to what it was before. This changing flux shape across the core affects refuelling scheme to be used if maximum burnup is to be obtained from the fuel.

### 5.6.2 Intermediate Term Reactivity Effects - Xenon Poison

Some of the fission products and the products into which they decay have very high absorption cross-sections for thermal neutrons. They capture many neutrons and hence they are called "poison". Samarium is one of these "poisons", but the by far the most important one is xenon, Xe-135, which has in fact the largest known absorption cross section for thermal neutrons. It forms and decays as follows with half-lives shown:

$$Tellurium^{135} \rightarrow Iodine^{135} \rightarrow Xenon^{135} \rightarrow Cesium^{135}$$

Tellurium is formed in about 5.6 % of the fissions, and Xe-135 is also formed directly in about 0.3 % of the fissions. During normal operation, Xe-135 is changed to harmless Xe-136 by absorption of neutrons, but in absence of neutrons, it disappears only by the slow natural decay as indicated above.

Because Tellurium decays very fast into idoine I-135, we may assume that iodine is produced directly from fission. Hence the Iodine and xenon concentrations are given by:

$$\frac{dX}{dt} = \gamma_X \Sigma_f \phi + \lambda_I I - \lambda_X X - \sigma_X \phi X \quad \text{.....(5.60)}$$

$$\frac{dI}{dt} = \gamma_I \Sigma_f \phi - \lambda_I I \quad \text{.....(5.61)}$$

where X, I = xenon, Iodine concentrations, nuclei/cm$^3$

$\phi$ = neutron flux in neutrons/cm$^2$.sec

$\Sigma_f$ = macroscopic fission cross section

$\gamma_X$, $\gamma_I$ = fractional yield of xenon and Iodine ($\gamma_X$=0.003; $\gamma_I$=0.056)

$\lambda_X, \lambda_I$ = decay constants of xenon and iodine ($\lambda_X$=2.1 × $10^{-5}$; $\lambda_I$=2.9 × $10^{-5}$ sec$^{-1}$)

The terms on the right hand side of the ODEs can be described as follows:

- $\gamma_X \Sigma_f \phi$ and $\gamma_I \Sigma_f \phi$ are production rates of X and I, directly from fission.
- $\lambda_I I, \lambda_X X$ are loss rates of Iodine, xenon due to natural decay.
- $\sigma_X \phi X$ is the loss of Xe-135 due to conversion to Xe-136 by neutron capture. ($\sigma_X$ = 3.5 × $10^{-18}$ cm$^2$ is the microscopic capture cross-section of xenon for thermal neutrons.)

Under steady state condition, (dX/dt = 0; dI/dt = 0) for a constant neutron flux $\phi_0$, the solutions are:

$$I_0 = \frac{\gamma_I \Sigma_f \phi_0}{\lambda_I} \quad \text{.....(5.62)}$$

$$X_0 = \frac{(\lambda_I I_0 + \gamma_X \Sigma_f \phi_0)}{(\lambda_X + \sigma_X \phi_0)} \quad \text{.....(5.63)}$$

Substituting (5.62) into (5.63), we have

$$X_0 = \frac{(\gamma_I + \gamma_X)\ \Sigma_f \phi_0}{\lambda_X + \sigma_X \phi_0} \quad \text{.....(5.64)}$$

For small flux $\phi_0$,

$$X_0 \approx \frac{(\gamma_I + \gamma_X)\ \Sigma_f \phi_o}{\lambda_X} \text{ therefore proportional to flux .....(5.65)}$$

For large flux $\phi_0$,

$$X_0 \approx \frac{(\gamma_I + \gamma_X)\ \Sigma_f}{\sigma_X}, \text{ therefore constant.....(5.66)}$$

*So for high power reactor, the steady state xenon concentration is independent of power.*

xenon Poison Reactivity Feedback:

A reactor which is critical with xenon will certainly be supercritical without its xenon. The *reduction in reactivity caused by the xenon* is called "xenon Load", and is expressed in mk. The xenon load is due to the change in thermal utilization factor f.

Let $f = \dfrac{\Sigma_a}{\Sigma_a + \Sigma_m}$ without xenon, and $f' = \dfrac{\Sigma_a}{\Sigma_a + \Sigma_m + \Sigma_X}$ with xenon

Note that $\Sigma_m$ is the absorption cross section of all materials in the reactor that is not fuel or xenon.

Now

$$\frac{K_e' - K_e}{K_e'} = \frac{f' - f}{f'} = 1 - \frac{f}{f'} = 1 - \frac{\Sigma_a + \Sigma_m + \Sigma_X}{\Sigma_a + \Sigma_m} = -\frac{\Sigma_X}{\Sigma_a + \Sigma_m}$$
$$= -\frac{\Sigma_X}{\Sigma_a} \cdot \frac{\Sigma_a}{\Sigma_a + \Sigma_m} = -R \cdot f \qquad \text{.....(5.67)}$$

where R is called *poisoning factor*. It is a ratio of absorption by Xe-135 to absorption by the fuel $\dfrac{\Sigma_X}{\Sigma_a}$.

At equilibrium, $\dfrac{\Sigma_X}{\Sigma_a} = \dfrac{X_0 \cdot \sigma_X}{\Sigma_a}$ .....(5.68)

Substituting equation (5.64) into (5.68), we have

$$R = \frac{\Sigma_X}{\Sigma_a} = \frac{\sigma_X \cdot (\gamma_I + \gamma_X)\ \Sigma_f \phi_0}{(\lambda_X + \sigma_X \phi_0) \cdot \Sigma_a} \qquad \text{.....(5.69)}$$

When the reactor is critical with its xenon load (i.e. $K_e' = 1$), and - Rf represents the reactivity loss due to the xenon. In other words, the xenon load is the product of R and f.

For example; a reactor with man fuel flux of $\phi_0 = 3 \times 10^{13}$ n.cm$^{-3}$.s$^{-1}$ ; $\sigma_X = 3.5 \times 10^{-18}$ cm$^2$ ; $\gamma_X = 0.003$; $\gamma_I = 0.056$; $\lambda_X = 2.1 \times 10^{-5}$; $\dfrac{\Sigma_f}{\Sigma_a} = 0.54$ (for natural uranium), substituting these numbers into (5.69), we have R = 0.028. Given f = 0.9, the xenon load R will be 0.0252. This means that this reactor would suffer a reactivity loss of 25.2 mk caused by equilibrium xenon poisoning.

The presence of an equilibrium xenon load means that enough excess reactivity must be built into the reactor so that the regulating system can compensate for the loss in reactivity as the xenon builds up.

Let us consider xenon transients after sudden reactor shutdown or following power reductions.

Let at t=0 the reactor be shutdown suddenly from a steady state flux $\phi$ with corresponding xenon and iodine concentrations $X_{0,}I_0$. Putting $\phi = 0$ in equation (5.60), (5.61), we have:

$$\frac{dX}{dt} = -\lambda_X X + \lambda_I I$$
$$\frac{dI}{dt} = -\lambda_I I \qquad \text{.....(5.70)}$$

The solution is:

$$I(t) = I_0 e^{-\lambda_I t}$$
$$X(t) = X_0 e^{-\lambda_X t} + \frac{\lambda_I}{\lambda_X - \lambda_I} I_0 (e^{-\lambda_I t} - e^{-\lambda_X t}) \qquad \text{.....(5.71)}$$

It can be seen that for initial steady flux of $\phi = 10^{14}$ n.cm$^{-2}$.sec$^{-1}$, the xenon reactivity load rises from an equilibrium value of - 28 mk at t = 0 to a maximum value of -132 mk somewhere between 7 and 12 hours after shutdown. Note that the peak xenon load level depends proportionally on the reactor flux level before shutdown. Physically, after shutdown, there are no neutrons to convert the xenon, so it disappears only by slow natural decay. But xenon is being produced at a faster rate by the decay of iodine, so its level increases. Due to the decay of iodine, a maximum is reached, and the xenon decays thereafter.

To find the peak xenon load, we differentiate (5.71) with time and put dX/dt = 0, we have

$$\frac{dX}{dt} = \frac{\lambda_I I_0}{\lambda_X - \lambda_I}(\lambda_X e^{-\lambda_X t} - \lambda_I e^{-\lambda_I t}) - \lambda_X X_0 e^{-\lambda_X t} = 0 \qquad \text{.....(5.72)}$$

The time where peak xenon load occurs is:

$$t = \frac{1}{\lambda_X - \lambda_I} \ln\left(\frac{\lambda_I \lambda_X (I_0 + X_0) - \lambda_X^2 X_0}{\lambda_I^2 I_0}\right) \qquad \text{.....(5.73)}$$

Substituting this into X(t) gives:

$$X_{\max} = (I_0 + AX_0)(B + AB\frac{X_0}{I_0})^C \qquad \text{.....(5.74)}$$

where

$$A = 1 - \frac{\lambda_X}{\lambda_I}; B = \frac{\lambda_X}{\lambda_I}; C = \frac{\lambda_X}{\lambda_I - \lambda_X} \quad \text{.....(5.75)}$$

The consequences for peaking xenon load are far reaching - it is not economical to provide the extra fuel required to override such large negative reactivities. The net result is that after a shutdown, *the reactor can only be restarted within about 40 minutes, before the xenon has grown too much.* If this is not possible, it is necessary to wait about *40 hours* for the xenon to decay naturally to a sufficiently low level.

Xenon effects also accompany both power reductions and increases. For a power reduction, the effect is essentially as above, with the maximum xenon reactivity load depending on the magnitude and the rate of the power reduction. With a power increase, the effect is reversed, and a strong dip occurs in the xenon concentration. Its depth again depends on the size and the rate of the increase. To prevent a sharp increase of reactor power due to this dip, it is necessary to introduce neutron absorbing reactivity control devices to compensate for it.

Spatial oscillations can be present in large reactors, due to the xenon. Suppose that for some reason, power rises slightly in one half of the reactor. If the total power is kept constant, this will cause a decrease in the other half. Since in the first half, more neutrons are available. More are absorbed by xenon so that xenon concentration in this half reduces, and therefore, the power in this half rises even more because fewer neutrons are absorbed by xenon. This further rise absorbs more xenon, increasing power further. The increased power produces more iodine, the decay of which into xenon counteracts the power increase. A maximum power is reached, and continued iodine decay then causes a power decrease, which in turn raises xenon, reducing power further etc. The period of these oscillations is ~ 1 day.

### 5.6.3 Short Term Reactivity Effects

(A) Power Reactivity Feedback

With temperature changes, the nuclear cross sections will change due to changes in mean energies, and density variations will affect the neutron mean free path and the leakage probabilities; as well, there is the broadening of the resonance cross section due to an increase in the temperature in the fuel, known as the "Doppler effect".

Therefore, the overall effect of temperature changes in the reactor will affect the reactor reactivity and is given by

$$\Delta K_T = \alpha_T \cdot (T_{FPSS} - T_{ZERO}) \quad \text{.....(5.76)}$$

where $\alpha_T$ is the temperature coefficient ( -15.06 × $10^{-6}$ /deg. C for fresh fuel)

$T_{FPSS}$ is the fuel temperature at reactor hot full power condition.

$T_{ZERO}$ is the fuel temperature at reactor zero power cold condition

For example, in order to bring the reactor power from cold 25 deg. C to hot condition at 271 deg, C, the reactivity needed to compensate for reactivity loss due to temperature effects is

$$\Delta K_T = -15.06 \times 10^{-6} \times (271 - 25) = -0.3705 \times 10^{-2} = -3.705 \text{ mk}$$

Reactor Coolant Void Reactivity Feedback

It has been shown that reactor core voiding (due to boiling or in extreme case, loss of coolant accident) will lead to positive reactivity excursion. Is generally expressed as:

$$\Delta K_V = \Delta K_{V\max} \cdot F \cdot (\chi - \chi_{FPSS}) \quad \text{.....(5.77)}$$

where $\Delta K_{V\max}$ =the maximum reactivity from coolant voiding ( typically 2 mk)

F = Flux shape weighting factor (typically 0.68)

$\chi$ = Coolant quality

$\chi_{FPSS}$ = Coolant quality at reactor full power (for CANDU, it is typically 2 to 4 %)

## 5.7 Modeling Exercise - Poison Reactivity Effects

### 5.7.1 Problem Description

1. Create a model which will predict the xenon concentration on a step change of neutron power.
2. Use the model to predict the following:
   - poison-out period for a reactor trip from 100% full power
   - the poison-prevent power level.

The following equations described the formation of xenon and iodine in a CANDU-type nuclear reactor.

$$\frac{dC_I}{dt} = 9.445E-3 * N - 2.8717E-5 * C_I$$

$$\frac{dC_{XE}}{dt} = 9.167E-4 * N + 2.8717E-5 * C_I - (2.1E-5 + 3.5E-4 * N) * C_{XE}$$

$$\Delta K_{XE} = (27.93 - C_{XE}) * 1E-3$$

CI = concentration of iodine(mK)

CXE = concentration of xenon(mK)

DKXE = reactivity change due to xenon(K)

N = neutron flux(normalized from 0 to 1)

### 5.7.2 Modeling Steps

1) Create a custom FORTRAN algorithm using the above equations, with neutron flux as an input to the algorithm, and concentration of iodine, concentration of xenon and the reactivity change due to xenon as outputs.

2) Remember to follow the CASSIM Technical Notes: algorithm development guidelines for calculations involving double precision numbers.

3) Use Euler's forward difference (explicit method) to model the above equations, and introduce an extra input, for varying the integration time-step, and call it the speed-up factor.

4) Use CASSBASE to overwrite ALG801, and replace it with your algorithm. Remember to define your inputs and outputs for the template in CASSBASE, and save your FORTRAN algorithm as ALG801.FOR.

5) Exit CASSBASE, and go to SRCLIB directory, and recreate CASSIM.DLL with the new ALG801 using the ADDTOLIB batch file.

6) Copy the new CASSIM.DLL to the CASSENG directory.

7) Use CASSBASE to create a new model and call it IAEA2.

8) Create a NMB block(ALG452) for communication with LABVIEW.

9) Create a block for calculating the xenon/iodine concentration transient using the custom algorithm ALG801. Remember to initialize the xenon and iodine concentration by depositing their values in the output section.

10) USE CASSBASE COPY BLOCK function to create 3 more blocks using ALG801.

11) A LABVIEW executable, CURVES.EXE is provided to display the results. This LABVIEW custom trends executable requires an additional block calls TIMEKEEPER(ALG451) which calculates the iteration count. The LABVIEW executable will convert the count to simulation time. Connect the OUPUT(1) of this block to INPUT(1) of this block, and OUTPUT(2) of this block to INPUT(2) of this block.

12) Since all 4 xenon/iodine transient calculation block is required to use the speedup factor, is there a quicker way than to change the inputs for all four blocks before running the model?

13) LABVIEW executable, CURVES.EXE expects the display variables in the following order:

    INPUTS TO BLOCK NMB

    1) TIMEKEEPER/IT1

    2) XENON_1/A2 (xenon conc)

    3) XENON_2/A2 (xenon conc)

    4) XENON_3/A2 (xenon conc)

    5) XENON_4/A2 (xenon conc)

    note: NMB must be block 1.

14) Use the above model and create transients for the cases of a step change of:

    a) 100% power reduction

    b) 80% power reduction

    c) 60% power reduction

    d) 40% power reduction.

15) Use a speedup factor of 600. Why is it better to use a smaller speedup factor, i.e. a smaller integration time-step?

16) Using the model and with the help of the LABVIEW executable, CURVES.EXE, can you find the following:

a) If a reactor core is tripped from full power, what will be the poison-out period? When must the reactor be re-started before a 'poison-out' occurs? And if a 'poison-out' did occur, how long before the reactor can be re-started?

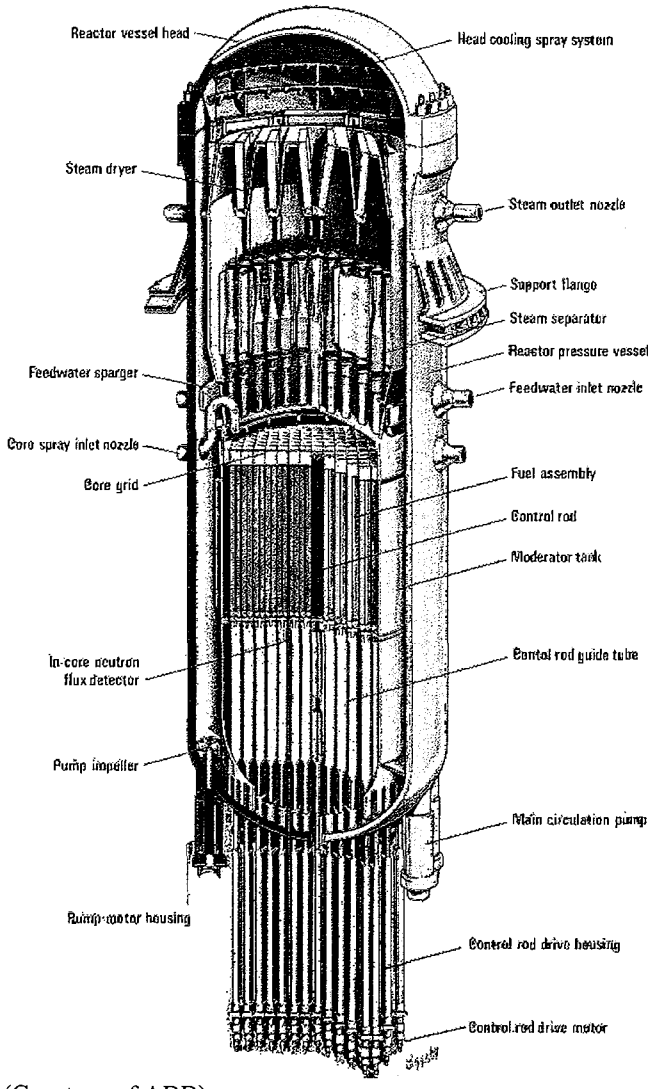
b) What *maximum* step-change in power will never give rise to a 'poison-out' reactor?

Note: the LABVIEW executable, CURVES.EXE can be found in directory CAS_TRN. Double-clicking the file from file manager will start the executable. It is best to start CASSENG before CURVES.EXE is started.

17) Try repeating the exercise with a different speedup factor.

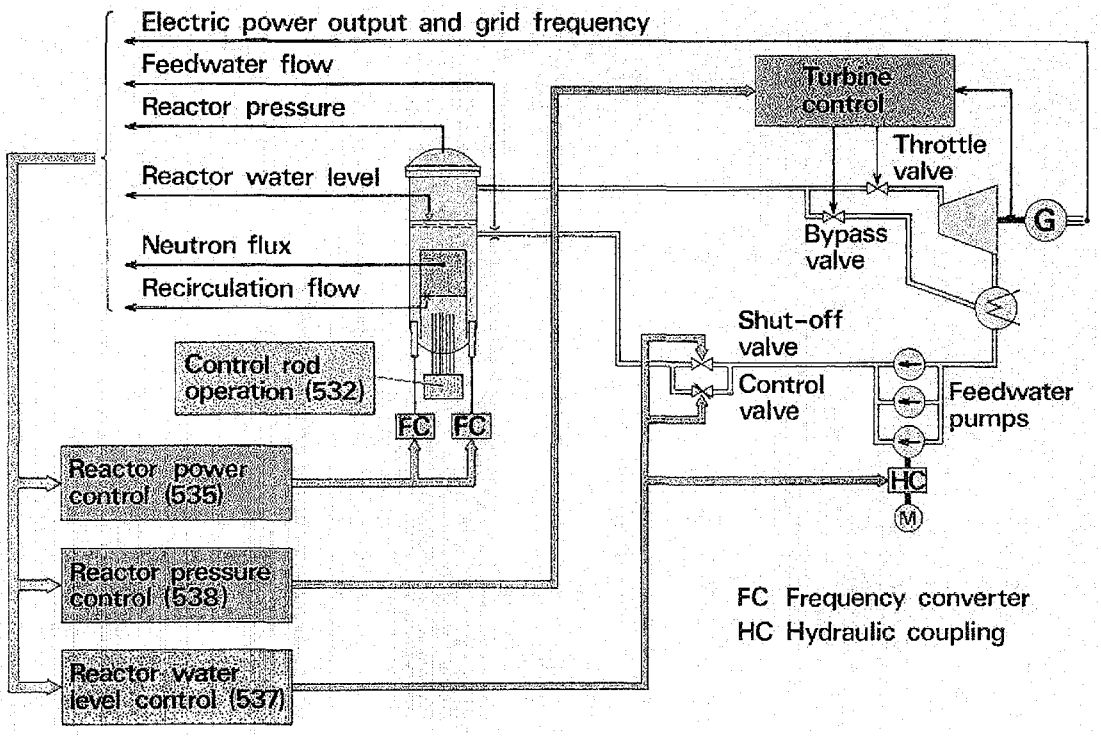
## 6.0 Reactor Control Simulation Model Development

### 6.1 Brief Review of BWR Reactor Power Control




(Courtesy of ABB)

BWR is the abbreviation for the Boiling Water Reactor. These reactors were originally designed by Allis-Chambers and General Electric (GE). The General Electric design has survived, whereas all Allis-Chambers units are now shutdown. The first GE US commercial plant was at Humboldt Bay (near Eureka) in California. Other suppliers of the BWR design world-wide include - ASEA-Atom, Kraftwerk Union, Hitachi. Commercial BWR reactors may be found in Finland, Germany, India, Japan, Mexico, Netherlands, Spain, Sweden, Switzerland, and Taiwan.




(Courtesy of ABB)

In the figure above, water is circulated through the Reactor Core picking up heat as the water moves past the fuel assemblies. The water eventually is heated enough to convert to steam. Steam separators in the upper part of the reactor remove water from the steam.

The steam then passes through the Main Steam Lines to the Turbine-Generators. The steam typically goes first to a smaller High Pressure (HP) Turbine, then passes to Moisture Separators (not shown), then to the 2 or 3 larger Low Pressure (LP) Turbines. In the sketch above there are 3 low pressure turbines, as is common for 1000 MW(e) plant. The turbines are connected to each other and to the Generator by a long shaft (not one piece).

The Generator produces the electricity, typically at about 20 000 volts AC. This electrical power is then distributed to a Generator Transformer, which steps up the voltage to either 230 000 or 345 000 volts. Then the power is distributed to a switchyard or substation where the power is then sent offsite.

The steam, after passing through the turbines, then condenses in the Condenser, which is at a vacuum and is cooled by ocean, sea, lake, or river water. The condensed steam then is pumped to Low Pressure Feedwater Heaters (shown but not identified). The water then passes to the Feedwater Pumps which in turn, pump the water to the reactor and start the cycle all over again.

*The BWR is unique in that the Control Rods, used to shutdown the reactor and maintain an uniform power distribution across the reactor,*

*are inserted from the bottom by a high pressure hydraulically operated system.* The BWR also has a Torus or a Suppression Pool. The torus or suppression pool is used to remove heat released if an event occurs in which large quantities of steam are released from the reactor or the Reactor Recirculation System, used to circulate water through the reactor.

The BWR reactor typically allows bulk boiling of the water in the reactor. The operating temperature of the reactor is approximately 570F producing steam at a pressure of about 1000 pounds per square inch.

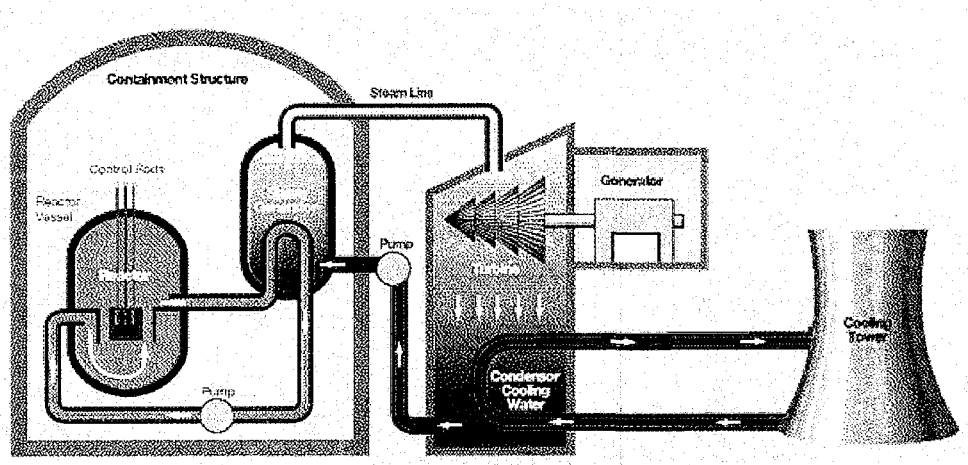
The reactor power output control system in BWR plant consists of a control rod made of neutron-absorbing materials, rod drive system and recirculation flow control system. The control rod and its drive system maintain a constant desired power level by adjusting the position of the rod inside the core. The recirculation flow control system also controls the reactor power level by changing the recirculation flow to alter the density of the water, a moderating material use to slow down the fission neutrons. This recirculation flow control is capable of changing the reactor power output rapidly over a wide range while keeping the power distribution in the core constant.

A BWR plant has numerous control rods installed inside the core. Each control rod is provided with a drive system and hydraulic control system. Rod position is adjusted by drawing out or insertion by means of the hydraulic pressure that is remotely controlled in the control room. Only one rod is operative at a time.

When the power output is changed by using a control rod, the changing rate is approximately 2 % power per minute. In the event of emergency shutdown of the reactor, all control rods are promptly inserted at once (scram) by the reactor trip system.

The coolant recirculation flow is controlled by a recirculation pump. The pump speed changes according to the change of the power frequency of the induction motor that drives the pump. Thus recirculation flow is regulated via a flow control system which consists of a recirculation pump motor generator set with a variable frequency generator and drive motor between which a hydraulic coupling is provided. The Flow Master Controller controls the speed of the motor generator set, thus providing the capability to change the reactor power output at a rate of 30 % per minute.

## 6.2 Brief Review of PWR Reactor Power Control




PWR is the abbreviation for the Pressurized Water Reactor. These reactors were originally designed by Westinghouse Bettis Atomic Power Laboratory for military ship applications, then by the Westinghouse Nuclear Power Division for commercial applications. The first commercial PWR plant in the United States was Shippingport, which was located near Pittsburgh, Pennsylvania.

In addition to Westinghouse, Asea Brown Boveri-Combustion Engineering (ABB-CE), Framatome, Kraftwerk Union, Siemens, and Mitsubishi have typically built this type of reactor throughout the world. Babcock & Wilcox (B&W) built a PWR design power plant but used vertical once-through steam generators, rather than the U-tube design used by the rest of the suppliers. Refuelings must be done with the plant shutdown.

The Pressurized Water Reactor (PWR) has 3 separate cooling systems. Only 1 is expected to have radioactivity - the Reactor Coolant System. The Reactor Coolant System, shown inside the Containment, consists of 2, 3, or 4 Cooling "Loops" connected to the Reactor, each containing a Reactor Coolant Pump, and Steam Generator. The Reactor heats the water that passes upward past the fuel assemblies from a temperature of about 530F to a temperature of about 590F. Boiling, other than minor bubbles called nucleate boiling, is not allowed to occur. Pressure is maintained by a Pressurizer (not shown) connected to the Reactor Coolant System. Pressure is maintained at approximately 2250 pounds per square inch through a heater and spray system in the pressurizer. The water from the Reactor is pumped to the steam generator and passes through tubes. The Reactor Cooling System is expected to be the only one with radioactive materials in it. Typically PWRs have 2, 3, or 4 reactor cooling system loops inside the containment.

In a Secondary Cooling System (which include the Main Steam System and the Condensate-Feedwater Systems), cooler water is pumped from the

Feedwater System and passes on the outside of those steam generator tubes, is heated and converted to steam. The steam then passes through the a Main Steam Line to the Turbine, which is connected to and turns the Generator.

The steam from the Turbine condenses in a Condenser. The condensed water is then pumped by Condensate Pumps through Low Pressure Feedwater Heaters, then to the Feedwater Pumps, then to High Pressure Feedwater Heaters, then to the Steam Generators. The diagram above simplifies the process by only showing the condenser, a pump, and the steam generator.

The condenser is maintained at a vacuum using either vacuum pumps or air ejectors. Cooling of the steam is provided by Condenser Cooling Water pumped through the condenser by Circulating Water Pumps, which take a suction from water supplied from the ocean, sea, lake, river, or Cooling Tower (shown). A discussion of cooling towers is provided in the photo section.

Light water PWR plants have four main control systems: control rod control (reactor power control), steam generator level control, pressurizer pressure control and pressurizer level control.

*The control rod control is used for reactor power control (Figure 10). The control rods are moved up and down when a deviation between "primary power", $P_{PRI}$, and the "reference power", $P_{REF}$, formed by the turbine first-stage pressure (secondary power) exceeds the predetermined setpoint.*

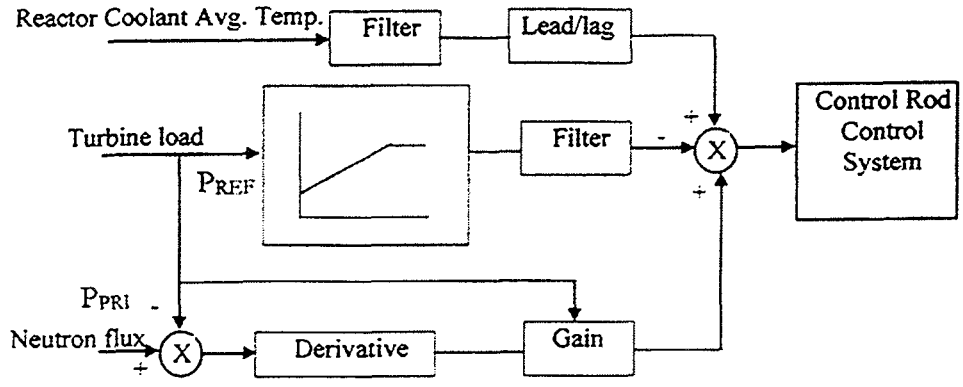
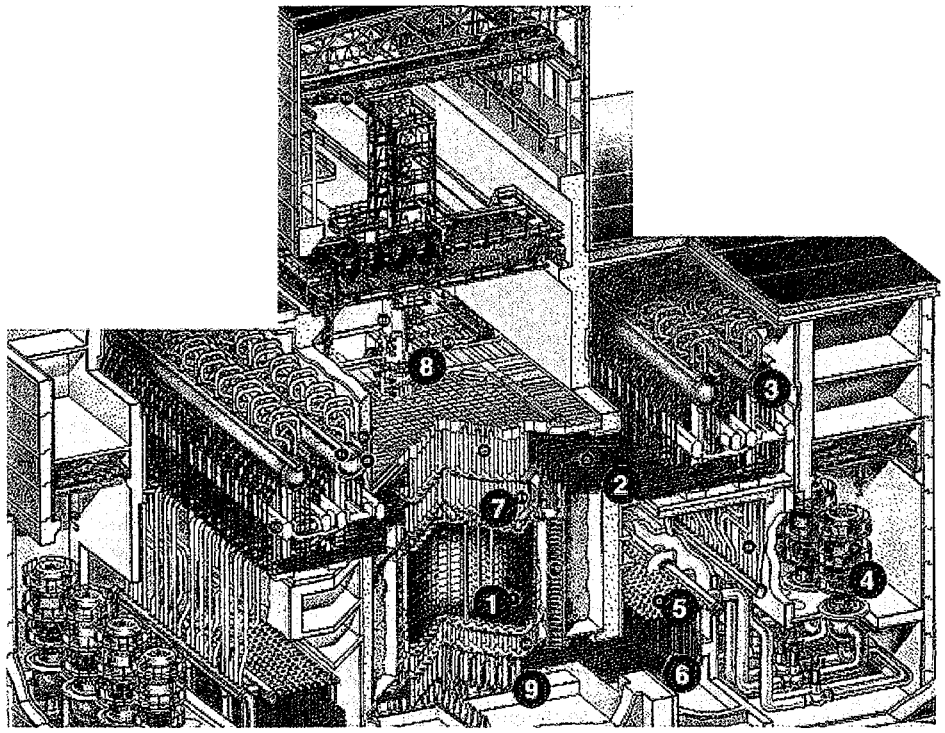



Figure 10 - Control Block Diagram for Reactor Control for a typical PWR. (Courtesy of IAEA)

Reactor power change due to $P_{PRI}$ change is compensated for by introducing the differentiated deviation signal between neutron flux and turbine 1stage pressure (power demand) into this system.

The boron concentration control system (which requires manual action) is used for the relatively long-term and slow core reactivity control, whereas the reactivity control (including reactor trip) responsive to load changes during normal operation is performed by controlling the rod positions

### 6.3 Brief Overview of RBMK Reactor Power Control

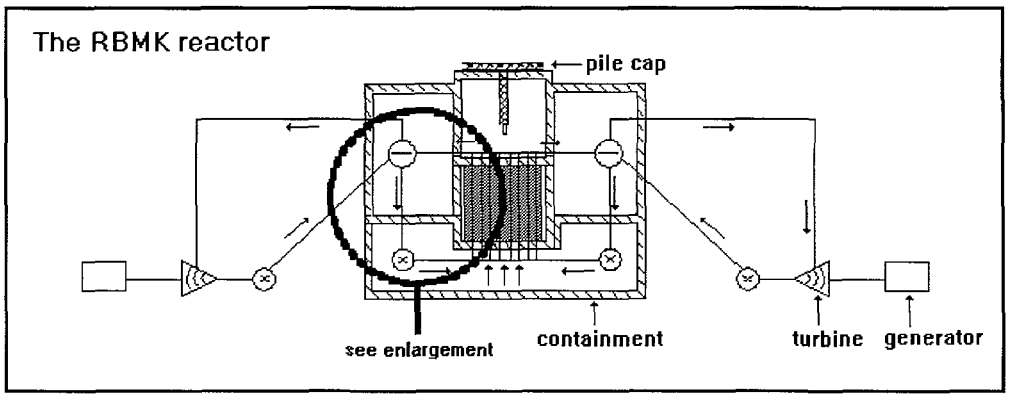


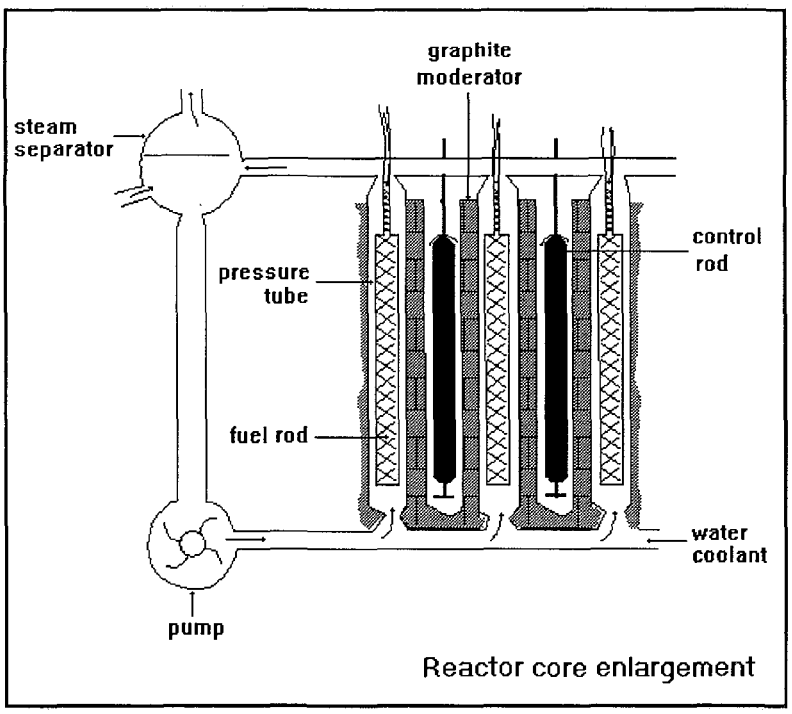

(Courtesy of Argonne National Lab.)

1. Reactor core - produces heat for converting water to steam
2. Steam-to-water pipes - carry steam to drum separator for moisture separation of steam
3. Drum Separator - used to separate water from steam
4. Main circulation pumps - used to circulate water through reactor for cooling
5. Group dispensing headers - headers coming from main circulating pumps
6. Water pipelines - headers supplying each channel in the reactor
7. Upper biological shield - protect operators and other personnel from radiation while working above reactor, e.g. during daily refueling
8. Refueling machine - used to place new fuel assemblies into the core and remove spent fuel assemblies
9. Lower biological shield - used to protect personnel from radiation while working in lower areas adjacent to reactor

The Soviet designed RBMK (a Russian acronym meaning Reactor High-Power Boiling Channel Type) is a pressurized water reactor with individual fuel channels and using ordinary water as its coolant and

graphite as its moderator. It is very different from most other power reactor designs as it was intended and used for both plutonium and power production. The combination of graphite moderator and water coolant is found in no other power reactors. *The design characteristics of the reactor were shown, in the Chernobyl accident, to cause instability when low power. This was due primarily to control rod design and a positive void coefficient. A number of significant design changes have now been made to address these problems.*



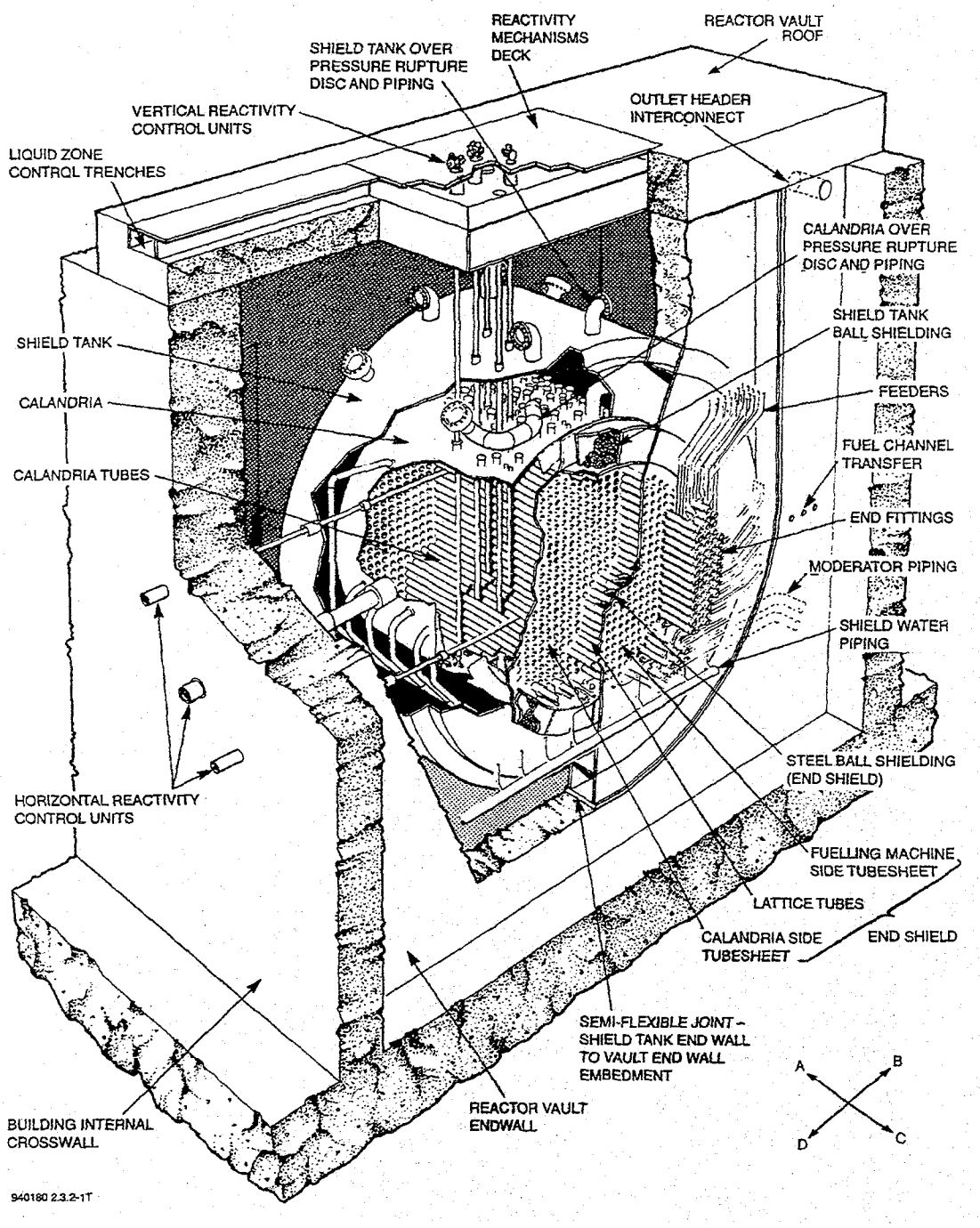

(courtesy of Uranium Institute)

1. Fuel rods - Pellets of enriched uranium oxide are enclosed in a zircaloy tube 3.65m long, forming a fuel rod. Two sets of 18 fuel rods are arranged cylindrically in a carriage to form a fuel assembly of about 10 m length. These fuel assemblies can be lifted into and out of the reactor mechanically, allowing fuel replenishment while the reactor is in operation.

2. Pressure tubes -within the reactor each fuel assembly is positioned in its own pressure tube or channel. Each channel is individually cooled by pressurized water.
3. Graphite moderator - a series of graphite blocks surround, and hence separate, the pressure tubes. They act as a moderator to slow down the neutrons released during fission. This is necessary for continuous fission to be maintained. Conductance of heat between the blocks is enhanced by a mixture of helium and nitrogen gas.
4. *Control rods - boron carbide control rods absorb neutrons to control the rate of fission. A few short rods, inserted upwards from the bottom of the core, even the distribution of power across the reactor. The main control rods are inserted from the top down and provide automatic, manual or emergency control. The automatic rods are regulated by feedback from in-core detectors. If there is a deviation from normal operating parameters (e.g. increased reactor power level), the rods can be dropped into the core to reduce or stop reactor activity. A number of rods normally remain in the core during operation.*
5. Coolant - two separate water coolant systems each with four pumps circulate water through the pressure tubes. Ninety-five percent of the heat from fission is transferred to the coolant. There is also an emergency core cooling system which will come into operation if either coolant circuit is interrupted.
6. Steam separator - in the RBMK design, boiling occurs. The steam produced passes to the Steam Separator which separates water from the steam. The steam then passes to the Turbine as in the Boiling Water Reactor design. Similar to the BWR case, the steam is radioactive, however, the steam separator introduces a delay time so radiation levels near the turbine may not be as high as in the BWR case.
7. Containment - the reactor core is located in a concrete lined cavity that acts as a radiation shield. The upper shield or pile cap above the core, is made of steel and supports the fuel assemblies. The steam separators of the coolant systems are housed in their own concrete shields.

In the RBMK design, water coolant slows the reaction down, meaning that the water coolant absorbs the neutrons, a necessary part of the reaction. In other words, *the water coolant additionally serves to control the reaction. The lack of water increases the reaction in the RBMK design, due to positive void reactivity feedback.* This is one chief difference between the power plants in the United States, known as the LWR (Light Water Reactors), and the RBMK type. In the LWR the lack of water kills the reaction, opposite of a RBMK.

## 6.4 Pressurized Heavy Water Reactor




(Courtesy of AECL)

The CANDU-PHW (CANada Deuterium Uranium, Pressurized Heavy Water) nuclear reactor is a heavy-water-moderated, heavy-water-cooled, natural uranium-fueled reactor which utilizes the pressure tube concept. The pressure tubes containing the fuel run horizontally through the reactor core. Pressurized heavy water carries the heat from the fuel to the steam generators.

Each pressure tube is isolated and insulated from the heavy water moderator by a concentric calandria tube and a gas annulus. Consequently the moderator system is operated at low temperature and pressure. The reactivity control and shutdown mechanism reside in the low-pressure moderator, thus simplifying their design, construction and maintenance and eliminating the possibility of their ejection in an accident situation. As well, this cool moderator can act as a heat sink under certain accident conditions.

The use of natural uranium fuel in an optimized lattice, and heavy water as moderator and coolant, combined with the capability to refuel the reactor while at full power, gives the CANDU reactor its good economy and low excess reactivity. This results in a power with very low fuel costs.

*The only reactivity feedback effects that are of any consequence are the coolant density (or void), fuel temperature and xenon effects. Since the moderator temperature is controlled independently of the primary coolant, it does not contribute any significant reactivity feedback. The fuel temperature reactivity coefficient is negative and therefore stabilizing whereas the coolant void reactivity coefficient is positive and therefore destabilizing. The sum of these, and lesser reactivity feedback effects, is a power reactivity coefficient that is near zero under normal operating conditions.*

### 6.4.1 Pressurized Heavy Reactor Control System General Description

A simplified diagram of a generic CANDU nuclear power plant and its digital computer control system is shown in Figure 11. Circulating pumps, also known as primary heat transport pumps maintain the flow of heavy water coolant around the "primary heat transport circuit" which transports fission energy from the reactor to the boiler. Inside the reactor, this coolant passes through fuel channels which contain cylindrical bundles of fuel. Heavy water "moderator" fills the space between the channels.

A surge tank, or pressurizer tank is used to absorb shrinkage and swell due to coolant temperature changes and to help maintain steady pressure in the circuit. The controls needed to control this circuit are basically:

(A) Heat Transport Pressure - it is controlled by electric heaters and steam bleed valves on the surge tank.

(B) Surge Tank Level - it is controlled by the feed and bleed valves shown on the circuit.

In the boiler, the energy is transferred to ordinary light water steam, and the steam drives a conventional turbine-generator set. The feedwater valves, pumps and preheaters complete the chain from the condenser back to the boiler. Steam can also be discharged directly to the condenser via the discharge valves with a capacity of 68 % of full power steam flow, or to the atmosphere via atmospheric steam discharge valves with a capacity of

10 %. The need for these valves is related to xenon poison effects explained earlier.

The boiler level control system (not indicated on the diagram) control the light water level in the boiler drum via feedwater valves.

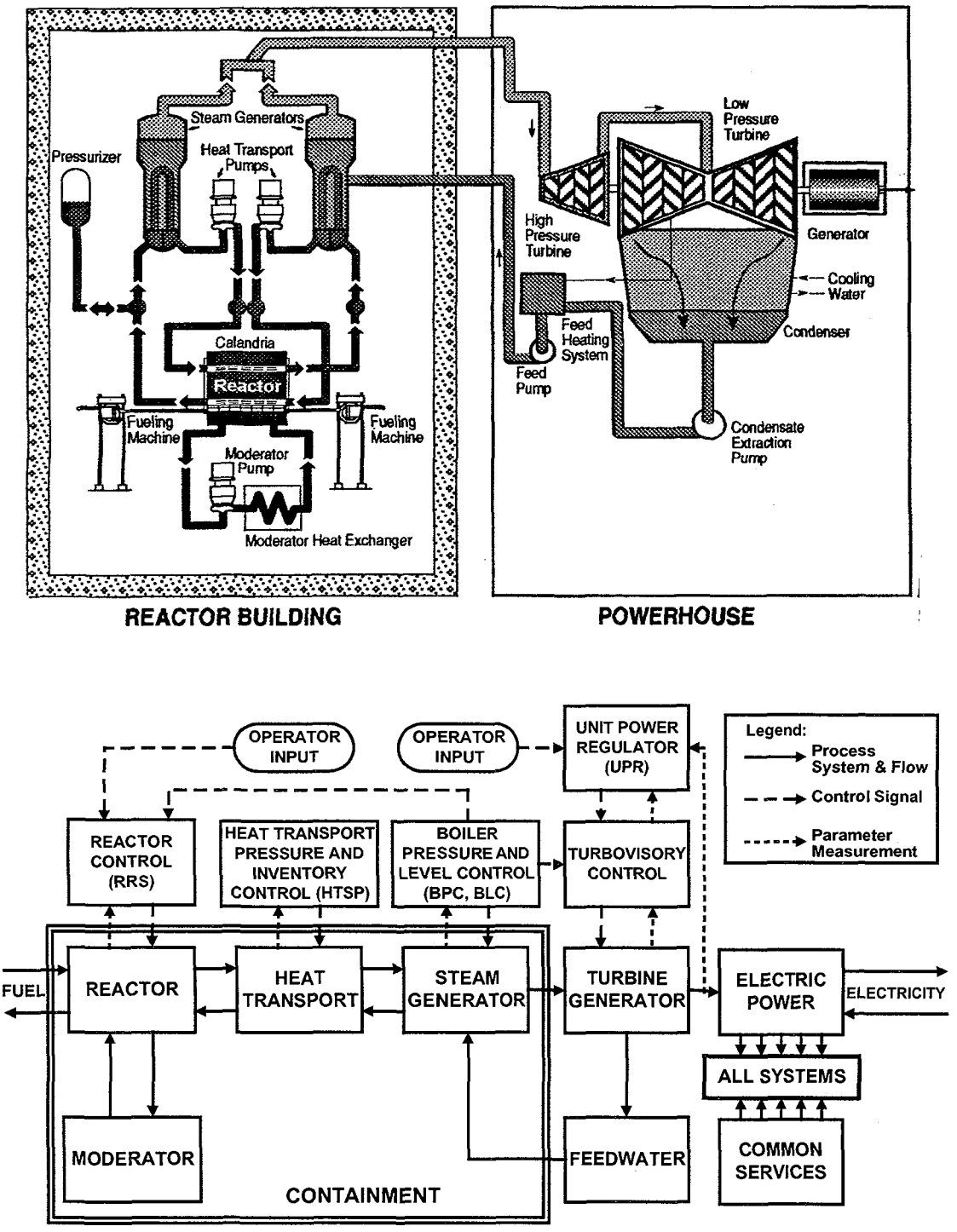



*Figure 11- A simplified diagram of a generic CANDU nuclear power plant and its digital computer control system (Courtesy of AECL)*

The three main control programs are:

- Unit Power Regulator (UPR) - it controls the generator output to the desired setpoint, which is set locally by the operator or remotely by the central load dispatch system, by adjusting the turbine load setpoint via the speeder gear of the turbine governor.

- Boiler Pressure Controller (BPC) - in the normal mode of operation, boiler pressure is controlled by adjusting the reactor power setpoint. It can also be controlled by adjusting the governor valves, and by opening the atmospheric or condenser steam discharge valves.

- Reactor Regulating System (RRS) - this is the control system which controls reactor power. It attempts to make the actual reactor power (determined by a Power Measurement Program), equal to the demanded reactor power (determined by Demanded Power Routine) by adjustment of the reactivity control devices via the Reactivity Mechanism Control Program. A "Flux Tilt" control program is used to prevent non-uniform flux distributions throughout the reactor by differential adjustment of reactivity control devices in different part of the reactor.

Three types of reactivity control devices are shown :

1. 14 light water zone control absorbers - these are vertical cylindrical compartments, distributed strategically through the core, which can be filled to any desired level with light water, a neutron absorber. Differential adjustment of levels in individual compartments is used for spatial (zonal) control. Platinum in-core flux detectors provide the neutron flux feedback signals required by the digital control algorithms for the regulation of the both the bulk and spatial flux.

2. 8 banks of Adjuster rods (total # of rods ~ 21)- these are neutron absorbers which are situated in various locations in the core and are normally fully inserted in the core. Being neutron absorbers, they will increase reactivity in the reactor by pulling themselves away from the core. They are withdrawn in symmetrical banks, under the control of the digital control computer, to provide positive reactivity for shimming the zonal light water control absorbers as well as for xenon override following a shutdown. The total reactivity worth of all fully pull-out adjuster rods is approximately + 17 mk. Each adjuster bank is assumed to have the same reactivity worth. In the event of a shutdown of reactor from full power, the adjuster rods have enough reactivity to restart the reactor within half an hour, before poisoning out.

3. 2 banks of mechanical Absorber rods (total # of rods = 4) - these Absorber rods decrease the reactivity inside the reactor by inserting themselves into the core. Being neutron absorbers, insertion into the core means more neutron absorption, thereby reducing the reactivity and the neutron power of the reactor. The total reactivity worth of both fully inserted absorber rods is approximately - 9mk. Each absorber bank is assumed to have the same reactivity worth.

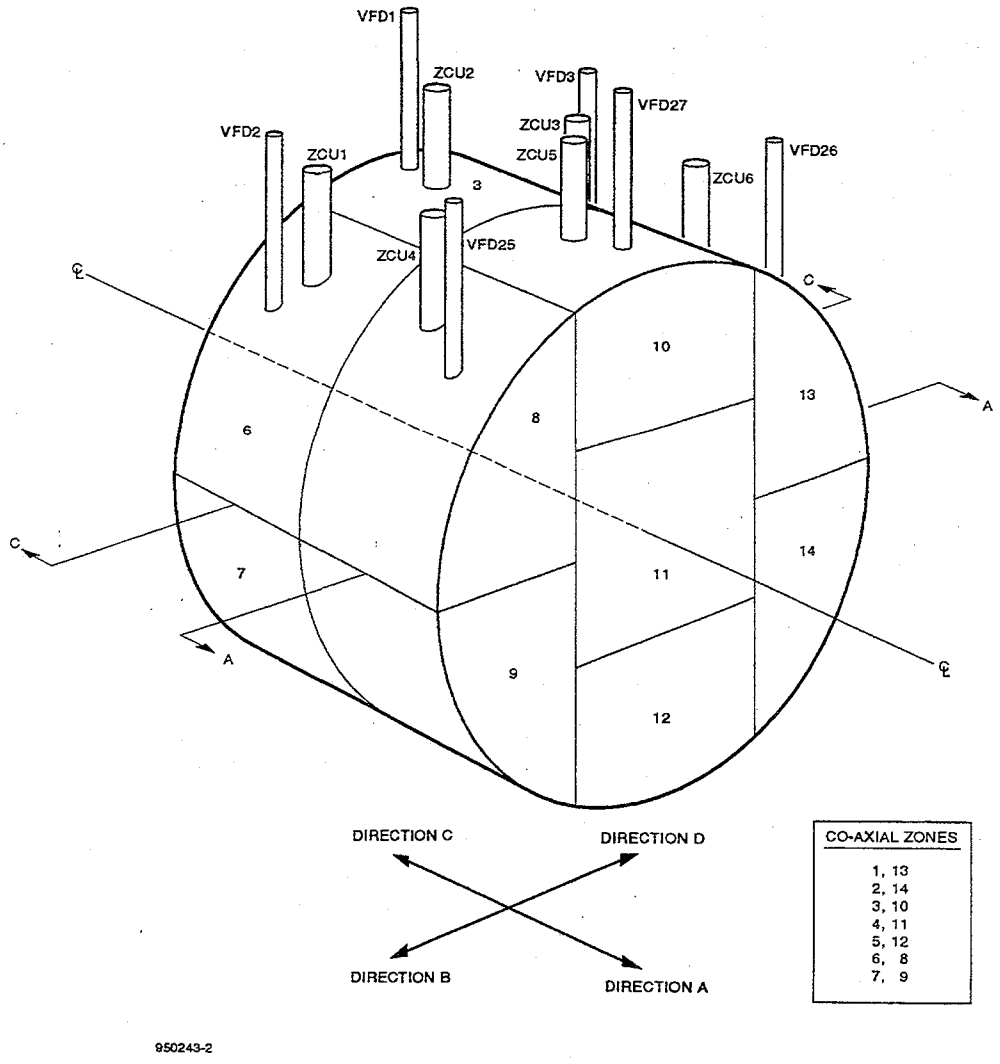



Figure 12–14 Liquid Zone Configurations in a typical CANDU Reactor

(Courtesy of AECL)

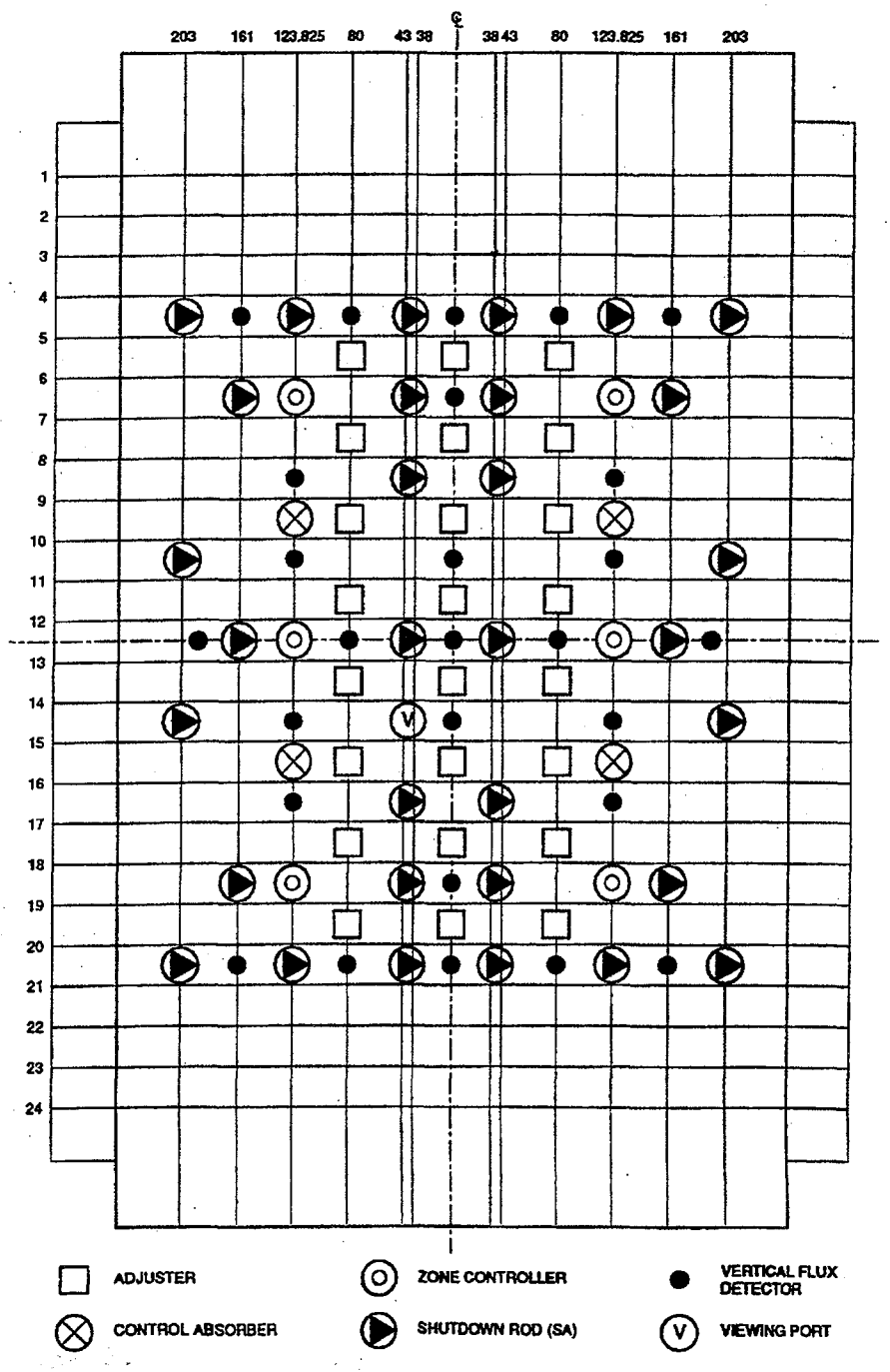



Figure 13 - Reactivity Mechanism Desk - showing the arrangement of Adjuster rods, Control Absorbers, Shutdown rods, and Zone Controllers.

(Courtesy of AECL)

### 6.4.2 Moderator Liquid Poison Addition and Removal System

Here the "poisons" refer to boron or gadolinium, which have large neutron capture cross section for thermal neutrons. They are added to the moderator to provide large negative reactivities under certain conditions:

- For instance, a typical CANDU has available excess reactivities of + 110mk at an initial startup with fresh fuel. For the reactor to be critical, reactivity usage must equal to available reactivity. Therefore, to compensate for fresh fuel, the equivalent of -60 mk is added to the moderator in the form of boron. As the fuel burns down, the boron must be gradually removed.

- Poison is also used at reactor startup after a long shutdown. The equivalent of - 30 mk of gadolinium must then be added to compensate for the absence of xenon, and gradually removed as xenon build up.

- Poison is also used to compensate for transient dip in xenon load during reactor power increase

Poison is removed by the ion exchange purification system. But the time constant of the removal is of the order of hours. Operation of poison addition and removal is controlled manually.

### 6.4.3 Reactor Trip System

Two safety shutdown system exist for CANDU; each with its own separate trip system. One is a high capacity system for direct poison injection into the moderator. Gadolinium, through more corrosive, is used as poison. It has the advantage of being much more easily removable by ion exchange, facilitating removal of the large quantity of poison in less time.

The second system consists of a number of shut-off rods, which can be dropped into the core in 2 seconds by de-energizing clutches in the drives. They are essentially the same as Absorber rods and are made of a stainless steel cadnium.

Conditions which cause reactor trips are:

- High neutron power > 115 % full power.

- High rate of power increase > 10 % per sec.

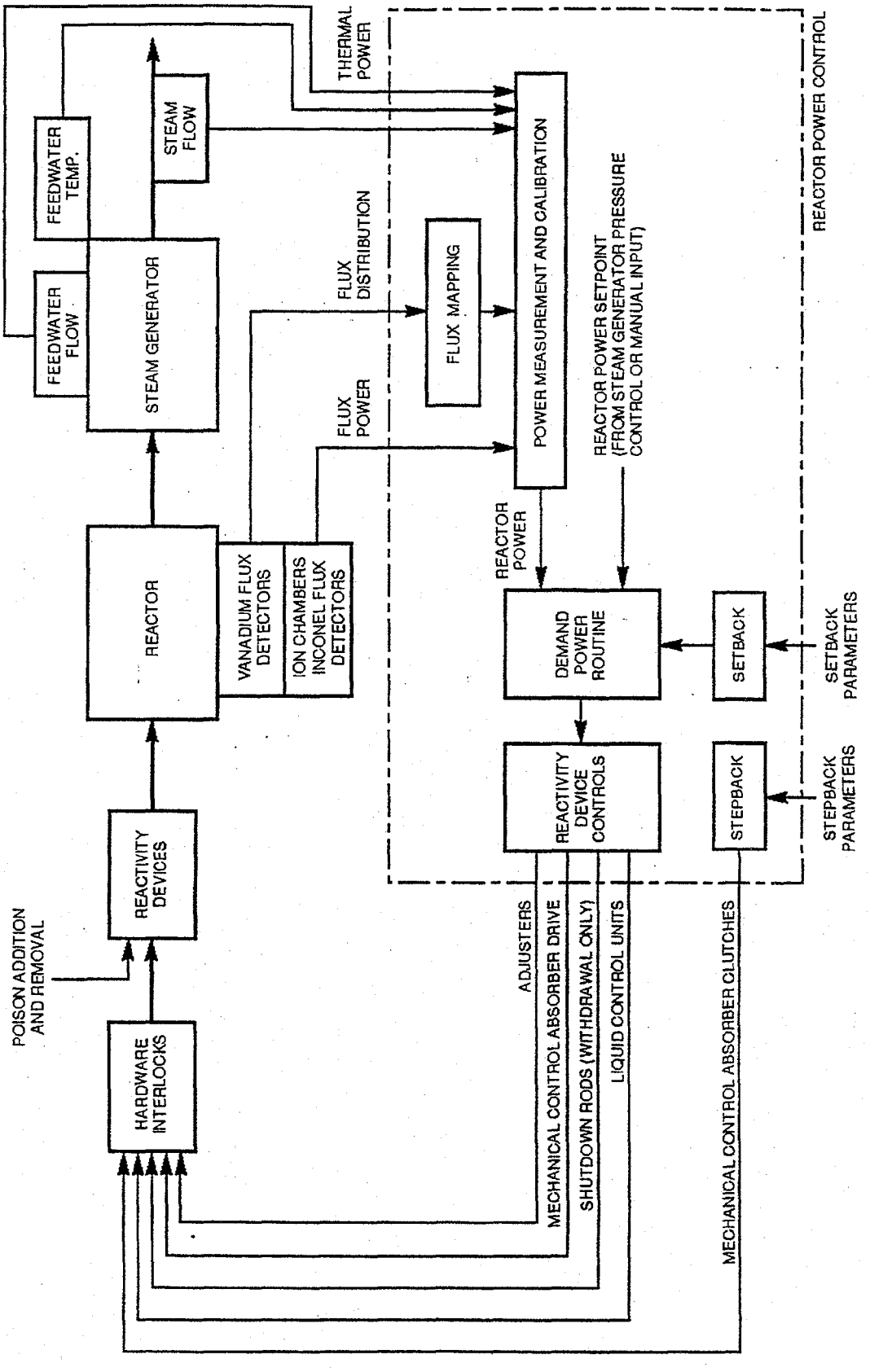
- Dangerous pressures or flows in certain parts of the system.

Triplicated instrumentation channels, with 2 out of 3 to be actuated to complete a trip, are used to avoid unnecessary trips due to spurious operation of one channel.

## 6.5 PHW Reactor Regulating System Brief Overview

The overview of the PHW Reactor Regulating System is shown in the following block diagram (figure 14):

(Courtesy of AECL)



- The *Power Measurement & Calibration* Program determines average reactor power and its distribution through the reactor.

- The *Demand Power Routine* sets the desired reactor power which depends on the mode of reactor operation. In the "Normal" mode of operation, reactor power is set to *follow* turbine generator power such that boiler pressure is kept constant under all conditions. In this mode, the setpoint demand signal comes from the Boiler Pressure Controller (BPC). In the upset conditions, the mode of plant operation is switched to "Alternate" Mode, in which case the reactor setpoint is manually adjusted or adjusted via a *Reactor Power Setback Routine.*

- The *Reactor Setback Routine* monitors a number of essential plant parameters and ramps the reactor power down if any one parameter is out of permissible limits. One of these parameters is provided by *the Power Mapping Routine*, which measures neutron flux in 54 locations, and uses these and other measurements to determine whether there are individual fuel bundles in certain locations whose power ratings has been exceeded.

- The Reactor Stepback Routine monitors a critical plant parameters and initiates a very fast power reduction if any one of the parameters is out of limits.

- Three types of Reactivity Control Mechanisms are available to implement desired power changes. Under normal circumstances, power is controlled only via the 14 liquid zonal absorbers. The Power Measurement Program provides the power in each zone. The zone controllers are adjusted in unison to control overall bulk reactor power with the use of the Flux Tilt Control, which is used to equalize the powers in the zones in order to prevent "flux tilt".

The reactivity mechanism control logic is summarized in Figure 15.

1. The primary method of short-term reactivity control is by varying the liquid level in the zone controllers. Normally, the adjusters are fully inserted , the control absorbers are fully withdrawn and the average liquid zone control compartment level is between 30 % and 50 %. The total control signal to the zone control valves consists of the bulk power control term plus a differential component proportional to that zone's power error.

2. In case of a shortage of negative reactivity, indicated by high zone controller level or a large positive power error, the mechanical control absorbers are driven in, one bank at a time.

3. In case of a shortage of positive reactivity, indicated by a low zone controller level or a large negative power error, the adjusters are driven out in a specific sequence.

4. The adjusters and mechanical control absorbers are driven at a speed proportional to power error to minimize, at low power errors, the shim reactivity rate which must be canceled by zone controllers.

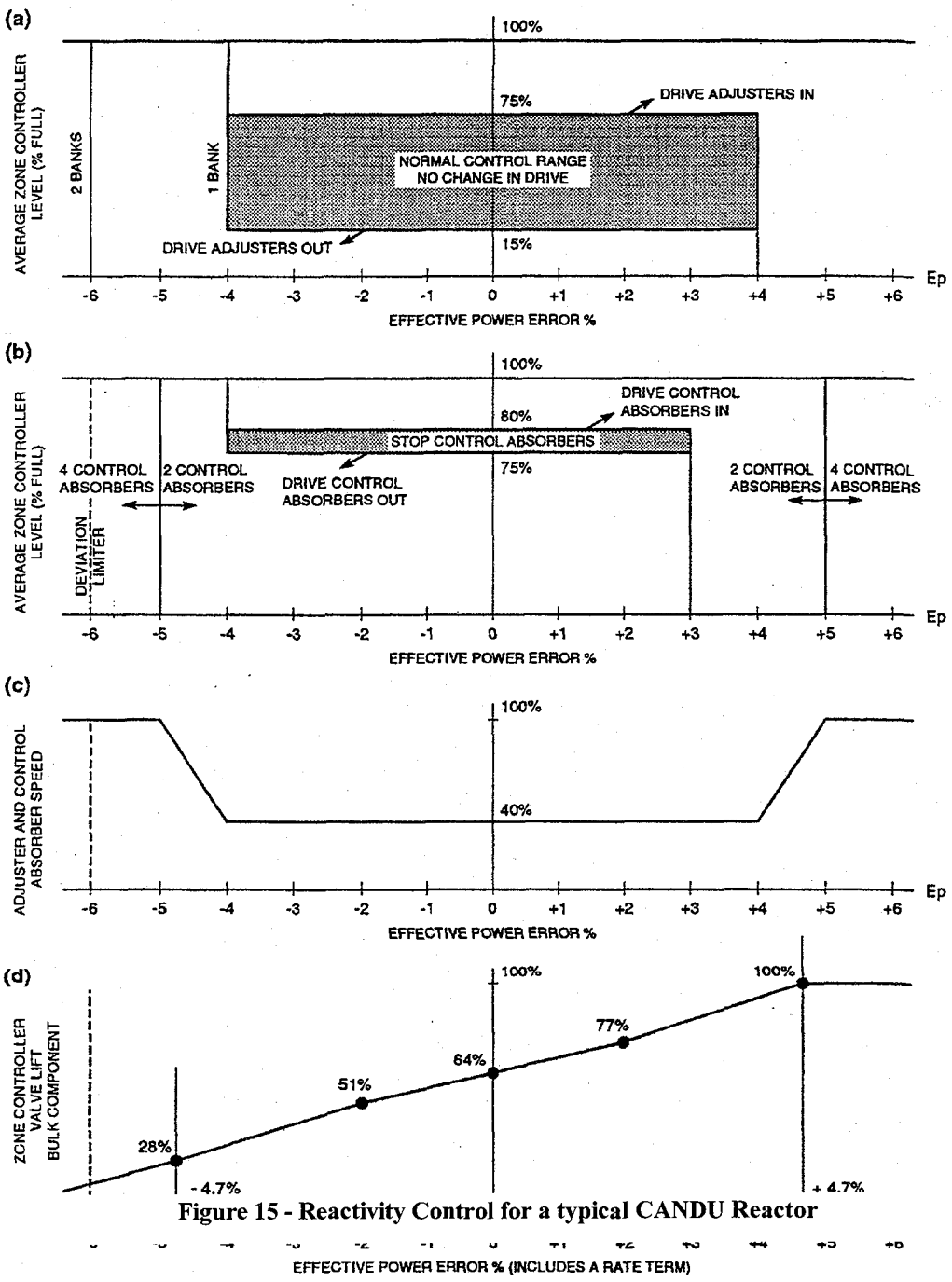



**Figure 15 - Reactivity Control for a typical CANDU Reactor**

**(Courtesy of AECL)**

## 6.6 Generic CANDU Simulator - RRS Familiarization

Summary of Simulator Features.

| SYSTEM | SIMULATION SCOPE | DISPLAY PAGES | OPERATOR CONTROLS | MALFUNCTIONS |
|---|---|---|---|---|
| REACTOR | • neutron flux levels over a range of 0.001 to 110% full power, 6 delayed neutron groups<br>• decay heat (3 groups)<br>• all reactivity control devices<br>• xenon and boron poison<br>• reactor regulating system<br>• reactor shutdown system | • reactivity control devices<br>• shutdown rods<br>• reactor regulating system | • reactor power and rate of change (input to control computer)<br>• manual control of reactivity devices<br>• reactor trip<br>• reactor setback<br>• reactor stepback | • reactor setback and stepback fail<br>• one bank of control rods drop into the reactor |
| HEAT TRANSPORT | • two phase main circuit loop with four pumps, four steam generators, four equivalent reactor coolant channels<br>• pressure and inventory control (pressurizer, degasser condenser, feed & bleed control, pressure relief)<br>• operating range is zero power hot to full power | • main circuit<br>• pressure control<br>• pressurizer control<br>• feed and bleed control<br>• inventory control<br>• degasser condenser control | • circulating pumps<br>• pressurizing pumps<br>• pressurizer pressure<br>• pressurizer level<br>• degas cond. Pressure<br>• degas cond. Level<br>• feed & bleed bias<br>• isolation valves for: pressurizer, degasser cond., feed and bleed | • main circuit relief valve fails open<br>• pressurizer relief valve fails open<br>• pressurizer isolation valve fails closed<br>• feed valve fails open<br>• bleed valve fails open<br>• reactor header break |
| STEAM & FEEDWATER | • boiler dynamics, including shrink and swell effects<br>• steam supply to turbine and reheater<br>• turbine by-pass to condenser<br>• steam relief to atmosphere<br>• extraction steam to feed heating<br>• steam generator pressure control<br>• steam generator level control<br>• boiler feed system | • steam generator feed pumps<br>• steam generator level control<br>• steam generator level trends<br>• steam generator pressure control<br>• extraction steam | • level controller mode: computer or manual<br>• manual level control gain & reset time<br>• level control valve selection<br>• level control isolation valve opening<br>• extraction steam valves<br>• feed pump operation | • all level control isolation valves fail closed<br>• one level control valve fails open<br>• one level control valve fails closed<br>• all feed pumps trip<br>• all safety valves open<br>• steam header break<br>• flow transmitter fails |
| TURBINE-GENERATOR | • very simple turbine model<br>• mechanical power and generator output are proportional to steam flow<br>• speeder gear and governor valve allow synchronized and non-synchronized operation | • turbine-generator | • turbine trip<br>• turbine run-back<br>• turbine run-up and synchronization<br>• atmospheric and condenser steam discharge valves | • turbine spurious trip<br>• turbine spurious run-back |
| OVERALL UNIT | • fully dynamic interaction between all simulated systems<br>• unit power regulator<br>• unit annunciation<br>• computer control of all major system functions | • overall unit<br>• unit power regulator | | |

### 6.6.1 Familiarization with CANDU Generic Simulator

(1) *First load the CANDU generic Simulator up as per instructions given in the class.*

The generic CANDU Compact Simulator is made up of numerous interactive display screens or pages. All of these screens have the same information at the top and bottom of the displays, as follows:

- top of the screen contains 21 plant alarms and annunciations; these indicate important status changes in plant parameters that require operator actions;
- top right hand corner shows the simulator status:
- the window under 'LabVIEW' (this is the proprietary software that generates the screen displays) has a counter that is incrementing when LabVIEW is running; if LabVIEW is frozen (i.e. the displays cannot be changed) the counter will not be incrementing;
- the window displaying 'CASSIM' will be green and the counter under it will not be incrementing when the simulator is frozen (i.e. the model programs are not executing), and will turn red and the counter will increment when the simulator is running;
- to stop (freeze) LabVIEW click once on the 'STOP' sign at the top left hand corner; to restart 'LabVIEW' click on the $\Rightarrow$ symbol at the top left hand corner;
- to start the simulation click on 'Run' at the bottom right hand corner; to 'Stop' the simulation click on 'Freeze' at the bottom right hand corner;
- the bottom of the screen shows the values of the following major plant parameters:
    1. Reactor Neutron Power (%)
    2. Reactor Thermal Power (%)
    3. Turbine Power (%)
    4. Main Steam Header Pressure (kPa)
    5. Steam Generator Level (m)
    6. UPR Mode ('Normal' or 'Alternate')
- the bottom left hand corner allows the initiation of two major plant events:
    1. 'Reactor Trip'

2. 'Turbine Trip'

these correspond to hardwired push buttons in the actual control room;

- the box above the Trip buttons shows the display currently selected (i.e. 'Plant Overview'); by clicking and holding on the arrow in this box the titles of the other displays will be shown, and a new one can be selected by highlighting it;

- the remaining buttons in the bottom right hand corner allow control of the simulation one iteration at a time ('Iterate'); the selection of initialization points ('IC'); insertion of malfunctions ('Malf'); and calling up the 'Help' screen.

(2) *Explore the SHUTDOWN RODS PAGE.*

- The screen shows the status of Shutdown System (SDS) #1, as well as the reactivity contributions of each device and physical phenomenon that is relevant to reactor operations.

- The positions of each of the two SDS1 SHUTDOWN ROD banks are shown relative to their normal (fully withdrawn) position.

- REACTOR TRIP status is shown as NO (green) or YES (yellow), the trip can be reset here (as well as on the RRS / DPR page); note that SDS1 RESET must also be activated before RRS will begin withdrawing the Shutdown Rods.

- The REACTIVITY CHANGE of each device and parameter from the initial 100% full power steady state is shown, as well as the range of its potential value.

- Note that reactivity is a computed not a measured parameter, it can be displayed on a simulator but is not directly available at an actual plant.

- Note also that when the reactor is critical the Total Reactivity must be zero.

(3) *Explore the REACTIVITY CONTROL PAGE*

- This screen shows the Limit Control Diagram, and the status of the three reactivity control devices that are under the control of the Reactor Regulating System (RRS).

- The Limit Control Diagram displays the Operating Point in terms of Power Error and Average Liquid Zone level.

  POWER ERROR = ACTUAL POWER - DEMANDED POWER*

  * magnitude as well as rate differences are computed and multiplied by appropriate constants.

- If power error is negative, more (positive) reactivity is needed, hence liquid zone level will decrease and if this is insufficient, absorber rods and adjuster rods will be withdrawn from the reactor.

- If power error is positive, negative reactivity is needed, hence liquid zone level will increase and if this is insufficient, absorber rods and adjuster rods will be driven into the reactor.

- The ABSORBERS are moved in two banks, and are normally outside the core. They can be moved by RRS if AUTO is selected, or manually if they are placed in MANUAL mode.

- The ADJUSTERS are moved in eight banks, and are normally fully inserted into the core. They can be moved by RRS if AUTO is selected, or manually if they are placed in MANUAL mode.

- A single controller representation of the 14 liquid zone controllers is used in the simulation: the in-and out-flow rates, average zone level, inlet flow control valve position are shown, and it is possible to place the liquid zone controller on MANUAL and directly control the position of the Zone Level Control Valve.

- The speed of the Absorbers and Adjusters is displayed but cannot be controlled from this page.

(4) *Explore the RRS / DPR PAGE.*

- This screen permits control of reactor power setpoint and its rate of change while under Reactor Regulating System (RRS) control, i.e. in 'alternate' mode. Several of the parameters key to RRS operation are displayed on this page.

- The status of reactor control is indicated by the four blocks marked MODE, SETBACK, STEPBACK AND TRIP. They are normally green but will turn yellow when in the abnormal state.

- MODE will indicate whether the reactor is under NORMAL to ALTERNATE control, this status can also be changed here.

- SETBACK status is indicated by YES or NO; Setback is initiated automatically under the prescribed conditions by RRS, but at times the operator needs to initiate a manual Setback, which is done from this page on the Simulator: the Target value (%) and Rate (%/sec) need to be input.

- STEPBACK status is indicated by YES or NO; Stepback is initiated automatically under the prescribed conditions by RRS, but at times the operator needs to initiate a manual Stepback, which is done from this page on the Simulator: the Target value (%) need to be input.

- TRIP status is indicated by YES or NO; trip is initiated by the Shutdown Systems, if the condition clears, it can be reset from here. Note however, that the tripped SDS#1 must also be reset before RRS will pull out the shutdown rods, this must be done on the Shutdown Rods Page

- Key components of RRS and DPR control algorithm are also shown on this screen.

- The ACTUAL SETPOINT is set equal to the NORMAL SETPOINT under UPR control ('normal mode'), the upper and lower limits on this setpoint can be specified here.

- The ACTUAL SETPOINT is set equal to the ALTERNATE SETPOINT under RRS control ('alternate mode'); the value of ALTERNATE SETPOINT is input on this page.

- HOLD POWER 'On' will select 'alternate mode' and stops any requested changes in DEMANDED POWER SETPOINT.

- It is important to note that Reactor Power Setpoint Target and Rate are specified on the Simulator in terms of %FP and %FP/sec, i.e. as linear measurements, instead of the logarithmic values used in practice. The requested rate of change should be no greater than 7% of actual power per second in order to avoid a reactor LOG RATE trip. This is readily achieved in the 'at-power' range (above 15%FP), but only very small rates should be used at low reactor power levels (below 1%FP), such as encountered after a reactor trip.

(5) *Explore the UPR PAGE.*

- This screen permits control of station load setpoint and its rate of change while under Unit Power Regulator (UPR) control, i.e. 'normal' mode. Control of the Main Steam Header Pressure is also through this screen, but this is not usually changed under normal operating conditions.

- OUC (overall Unit Control) MODE can be changed from NORMAL to ALTERNATE.

- TARGET LOAD - on selection Station Load (%) and Rate of Change (%/sec) can be specified; change becomes effective when 'Accept' is selected.

- The OPERATOR INP TARGET is the desired setpoint inserted by the operator; the CURRENT TARGET will be changed at a POWER RATE specified by the operator.

- Note that the RANGE is only an advisory comment, numbers outside the indicated range of values may be input on the Simulator.

- MAIN STEAM HEADER PRESSURE SETPOINT (MPa) - alters the setpoint of the Steam Generator Pressure Controller, which is rarely done during power operation. Caution must be exercised when using this feature on the Simulator, since the requested change takes place in a step fashion as soon as the change is made; changes should be made in increments of 0.1 MPa.

### 6.6.2 Simulator Exercise related to PHW Reactor Control

(A) **POWER MANEUVER**: 10% Power Reduction and Return to Full Power

- initialize Simulator to 100% full power
- verify that all parameters are consistent with full power operation.
- select the UPR page, and change the scale on the "Reactor Pwr & Thermal Pwr" and "Current Target Load & Turbine Pwr" graphs to be between 80 and 110 percent, the "Main Steam Hdr Pressure & SP" to 4500 and 5000 kPa, "Boiler Level" to 13 and 15 meters, and set "Resolution" to "Max Out".
- reduce unit power in the 'normal' mode, i.e.
  - ⇒ using the UPR display
  - ⇒ select 'TARGET LOAD (%)' pop-up menu
  - ⇒ in pop-up menu lower 'target' to 90.00% at a 'Rate' of 1.0 %/sec
  - ⇒ 'Accept' and 'Return'
- observe the response of the displayed parameters until the transients in Reactor Power and Steam Pressure are completed (approximately 4 minutes and full time scale on the graph) without freezing the Simulator and/or stopping LabVIEW, and explain the main changes
- continuing the above operation, raise "UNIT POWER" to 100% at a rate of 1.0%FP/sec.

**(B) REACTOR REGULATING SYSTEM EXERCISES**

(1) *UNIT RESPONSE TO POWER MANEUVER IN 'ALTERNATE' MODE*

- initialize the Simulator to 100%FP, record parameters
- reduce power using RRS to 50% at 0.5%/sec, observe parameter changes during transient, and freeze as soon as Reactor Neutron Power reaches 50% and record parameter values
- unfreeze and let parameters stabilize, record parameter values

- return reactor power to 100%FP at 0.5%/sec, freeze as soon as Reactor Neutron Power reaches 100% and record parameter values
- unfreeze and let parameters stabilize, record parameter values

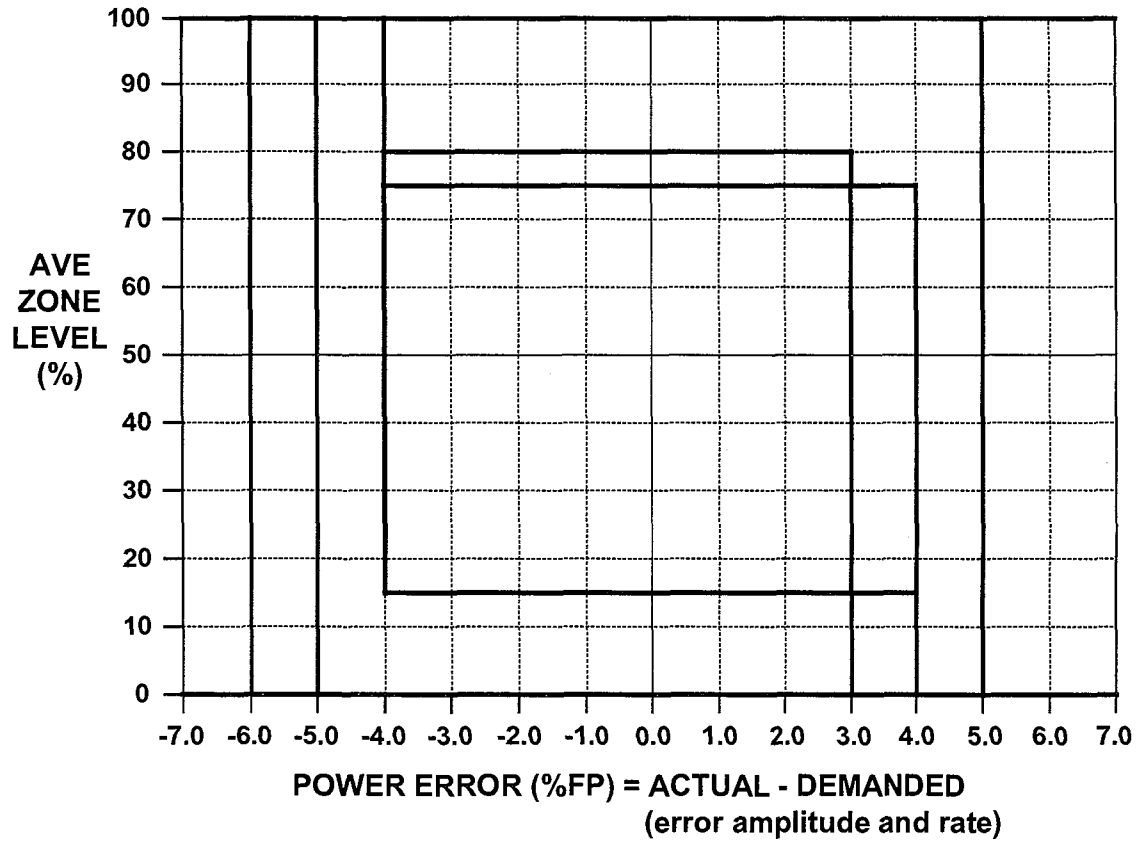
(2) *RESPONSE OF RRS CONTROL ALGORITHM TO POWER MANEUVER*

- initialize the Simulator to 100%FP and from the Reactivity Control page note the position of the operating point on the attached diagram (confirm the value of Average Zone Level on the Plant Overview page):

- insert a power reduction request using RRS to 70%FP at 0.8%/sec and freeze the simulator

- go to the Reactivity Control page, unfreeze, and note the path of the operating point on the attached diagram, until power error has stabilized at or near zero (about 3 - 4 minutes)

- confirm the value of average zone level on the Plant Overview page




Question - Why is the final zone level higher than the original zone level?

(3) *REACTOR AND RRS RESPONSE TO POWER MANEUVER*

- initialize the Simulator to 100%FP and from the Reactivity Control page note the position of the operating point on the attached diagram
- insert a power reduction request using RRS to 10%FP at 0.8%/sec and freeze the simulator
- go to the Reactivity Control page, unfreeze, and note the path of the operating point on the attached diagram, until at least one Adjuster Rod bank is out of the reactor (about 20 minutes) - once the first Adjuster Bank is more than 50% withdrawn, place Absorbers on Manual and drive them fully OUT.
- compare the response to previous case and explain the main differences, particularly the 'end' state

(4) *RESPONSE TO POWER MANEUVER UNDER MANUAL CONTROL*

- initialize the Simulator to the ZONESMAN initialization point (note if you do not have this initialization point, put all liquid zone on manual and create an initialization point accordingly - as instructed by instructor.)
- on the Reactivity Control page place the Absorbers and Adjusters to Manual
- using Absorber and Adjuster drives on Manual, maneuver reactor power to keep generator power between 80+1%FP
- note the time taken from the start of lowering reactor power until steady operation within the specified error limits is achieved

(5) *RESPONSE TO MANUAL WITHDRAWAL OF ADJUSTER RODS*

- initialize the Simulator to the ZONESMAN initialization point (note that all liquid zone controllers are on Manual - they are not to be used during this exercise)
- on the Reactivity Control page place the Absorbers and Adjusters onto Manual
- manually withdraw the Adjuster rods
- describe and explain the response of the system

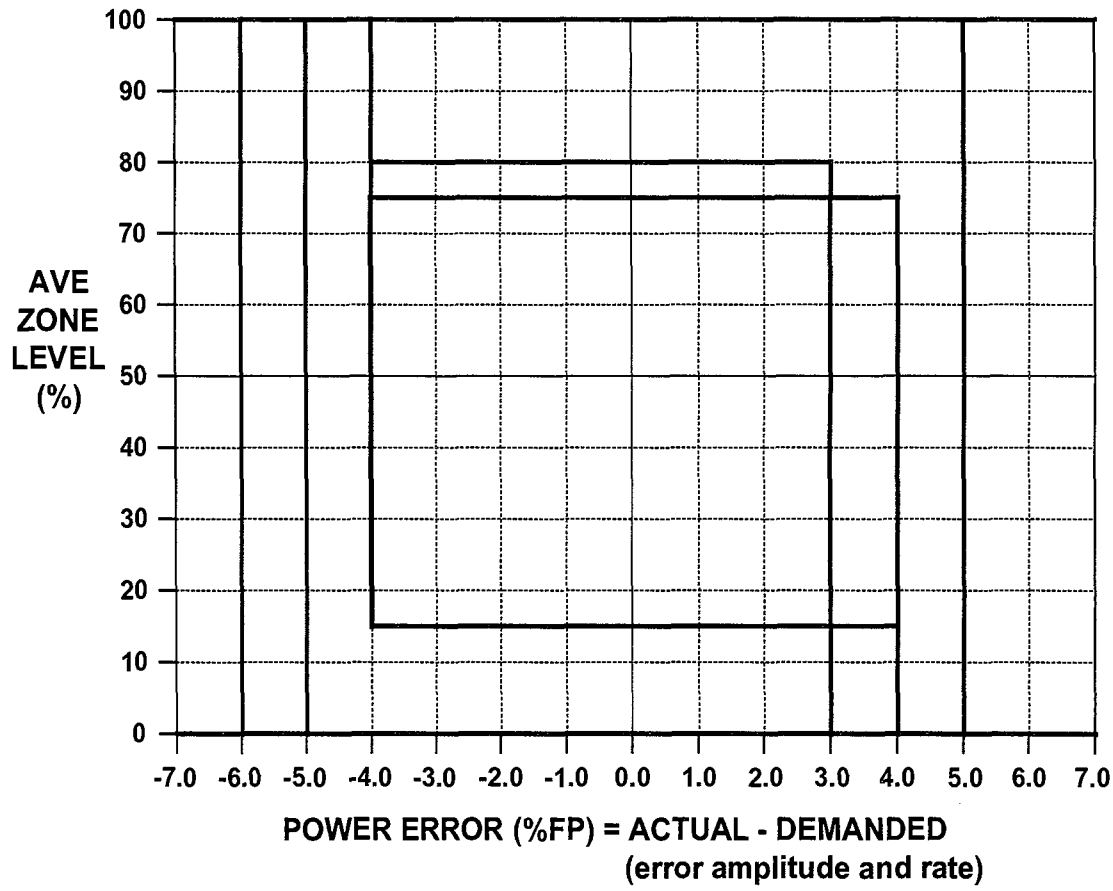
(6) *RESPONSE TO MANUAL WITHDRAWAL OF ADJUSTER RODS*

- initialize the Simulator to the ZONESMAN initialization point (note that all liquid zone controllers are on Manual - they are not to be used during this exercise)

- on the Reactivity Control page place the Absorbers and Adjusters onto Manual
- manually drive both banks of Absorbers into the core, and by the simultaneous withdraw of one or more the Adjuster banks keep reactor neutron power within 2% of 100%FP
- once all the Absorbers are fully in the core, attempt to drive all remaining Adjusters simultaneously out of the core
- describe and explain the response of the system

(7) *RESPONSE TO MALFUNCTION "ONE BANK OF ABSORBER RODS DROP"*

- initialize the Simulator to 100%FP and from the Reactivity Control page note the position of the operating point on the attached diagram
- insert the Malfunction "One Bank of Absorber Rods Drop" (use a five second time delay)
- observe system response on the Reactivity Control page and note the path of the operating point on the attached diagram



- note OUC mode and reactor power
- clear the malfunction
- once the Absorbers have fully withdrawn from the reactor, raise reactor power to a level dependent on the number of Absorber banks out of the reactor: for each bank partially or fully out, reactor power is limited by 5% (i.e. one bank - 95%FP, two banks - 90%FP, etc.)

  Question - what is the maximum power level that can be achieved?

(8) *RESPONSE TO SDS#1 REACTOR TRIP AND RECOVERY*

- initialize the Simulator to 100%FP
- manually trip the reactor
- observe the response of the overall unit
- wait until Generator power is zero and reactor neuron power less than 0.1%
- reset Reactor Trip and SDS#1
- record the time (using the display under the chart recorders) needed to withdraw all shutdown rods
- raise reactor power to 60%FP
- observe the response of the reactor regulating system and the reactivity changes that take place

(9) *RESPONSE TO SDS#1 REACTOR TRIP AND POISON-OUT*

- initialize the Simulator to 100%FP
- manually trip the reactor
- observe the response of the overall unit
- wait one hour before resetting Reactor Trip and SDS#1 (use initialization point given by instructor).
- after the shutdown rods have been withdrawn observe the status of the reactivity control devices
- attempt to raise reactor power - why is it not possible?
- note the reactivity changes that have taken place, in particular note the magnitude and estimate the rate of change of Xenon reactivity buildup

(10) *RESPONSE TO SDS#1 REACTOR TRIP AND POISON OVERRIDE*

- using the data from the previous two exercises, estimate the time available to the operator from the initiation of the reactor trip until the trip must be reset to avoid a poison outage: the desired end state of this exercise is reactor power at 60%FP and less than one bank of adjuster rods left in the core (i.e. the last bank is partially withdrawn)

- initialize the Simulator to 100%FP

- manually trip the reactor

- wait until the above calculated time has expired

- reset Reactor Trip and SDS#1

- raise reactor power to 60%FP

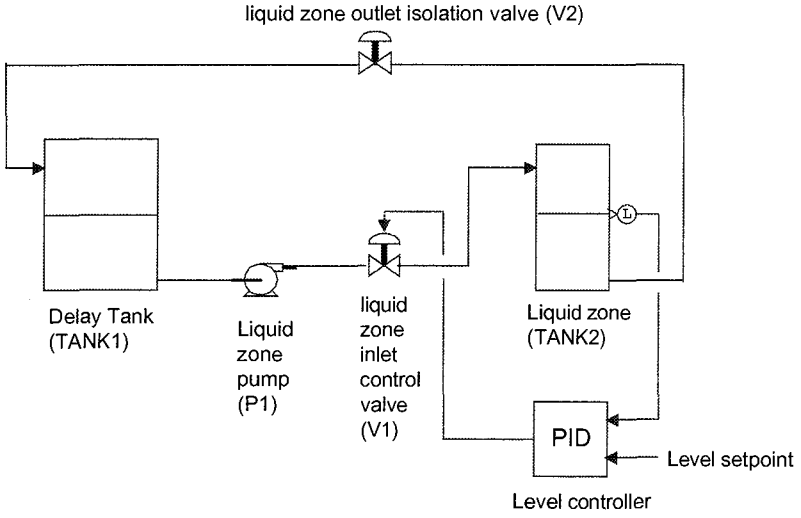
- note the final state of the adjuster rods and Average Liquid Zone level

## 6.7 Modeling Exercise - Reactivity Control Mechanism: Liquid Zone

### 6.7.1 Liquid Zone Hydraulic Modeling & Approach

***Liquid Zone Hydraulic Model***

Problem Description:
Create a process model as presented in the figure below.




1) Delay tank: Design Pressure = 135 KPaa
Design temperature = 30 deg C
Dimensions: radius = 0.5m
height = 4m
Full power design level = 2m
Elevation head = 5m

2) Liquid zone: Design Pressure = 500 KPaa
Design temperature = 30 deg C
Dimensions: radius = 0.224m
height = 4m
Full power design level = 2m

3) Liquid zone pump: Design flow = 12.6 Kg/sec
No load maximum static head = 1177 KPaa
Dynamic head = 1001 KPaa
Run-up/down time = 10 secs
Power requirements = 4.16 KV

4) Liquid zone isolation valve: stroking time = 2 seconds
plug type = linear
design flow = 6.3 Kg/sec

5) Liquid zone inlet control valve: stroking time = 2 secs
plug type = equal %
design flow = 12.6 Kg/sec

6) Properties of water: density = 1000 kg/$m^3$
heat capacity = 4.16 KJ/Kg/deg C

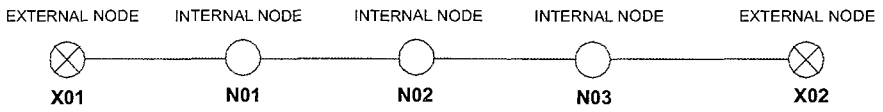
7) Temperature of water in the system can assumed to be constant.

1. Identify the process equipment which will be required to be modeled, and find the corresponding generic algorithms to match the equipment.

2. Identify which method of flow calculation will be used for each flow paths in the system.

3. Create a nodal diagram to represent the hydraulic network circuit.

***Liquid Zone Process***
***Network Nodal Diagram***




**Equipment locations:**
NHAX01_N01: ELEV HEAD OF 5 metres
NHAN01_N02: PUMP P1
NHAN02_N03: VALVE V1

**Full power design value:**
NHA_X01: 135 KPAA
NHA_X02: 500 KPAA
NHA_N01: 134 KPAA
NHA_N02: 1100 KPAA
NHA_N03: 900 KPAA

**FLOW:** 12.6 KG/S

4. Use CASSBASE to create a new model and call it IAEA3

5. The system ID chosen is AAA.

6. The flow between TANK1 and TANK2 will be calculated using the network method, and the return flow from TANK2 to TANK1 will be done outside of the network.

7. The process/non-network flow blocks required to be created are as follows:

| Equipment/function | **Block name** | **Generic algorithm used** | **Connection comments** |
|---|---|---|---|
| Pump 1 Breaker | AAA_P1_S | ALG133, BREAKER | The inputs to control the breaker will be constant tags. |
| Pump 1 Motor | AAA_P1_MTR | ALG134, MOTOR | The inputs to control the motor will be from the breaker status from the breaker block. |
| Pump 1 Hydraulics | AAA_P1 | ALG102, PUMP_SIM | The main inputs will be the motor speed from the motor block, and the flow from the network block |
| Valve 1 | AAA_V1 | ALG108, CV_SIM | The inputs to the valve position will be from the controller block. |
| Valve 2 | AAA_V2 | ALG106, VLV_ONOFF_SIM | The inputs to control the valve position will be a constant tag. |
| Conductance of TANK2 return flow | AAA_V2_COND | ALG301, 2_MULTI | Valve 2 normalized port-area is multiplied together will TANK2 level factor, such that if TANK2 runs dry, it will stop the return flow. |
| TANK 2 return flow calculation | AAA_RETURN | ALG140, CONDUCT_FI | The input will the TANK2 return flow conductance |
| TANK1 total inlet flow | AAA_RET_TOT | ALG_311, 2_SUM | TANK1 total inlet flow is sum of the return flow from TANK2 and the overflow from TANK2 |
| TANK1 | AAA_TANK1 | ALG128, TANK | The inlet flow is calculated from TANK1 total inlet flow block and the outflow from the network |
| TANK2 | AAA_TANK2 | ALG128, TANK | The inlet flow is calculated from the network, and the outlet flow from the TANK2 return flow calculation |

8. Create a NMB block(ALG452) for communication with LABVIEW.

9. Use CASSBASE and create the blocks for modeling the process equipment, and non-network type flow calculations.

   - Remember to create marker blocks for enclosing the process equipment blocks.

   - Input connections can be done after all the blocks are created.

10. Use CASSBASE and create the blocks related to the hydraulic network. Remember to create marker blocks for enclosing the network related blocks.

    - An initial node capacitance of 0.001 can be used..

    - Tank level factor must be set to one if no tank is connected directly to the link.

    - Remember to create a network control block and call it ***NHA_CTRL***, after you have created the last link block.

    - Remember to create marker blocks for enclosing the network related blocks. The NETWORK GENERATE routine requires it.

11. Use CASSBASE and run NETWORK GENERATE to produce the auxiliary network blocks.

12. Create a PID controller to control TANK2 level by controlling valve V1.

13. Use CASSBASE to EDIT the blocks and make the connections to the block inputs.

14. Make sure the connections between the blocks are made correctly, use the VERIFY command to locate any hanging tags, and clear these hanging tags by reconnecting the correct output tags to the inputs in question.

15. Connect the variables you would like to trend to block NMB (up to 8 is available).

    - Start LABVIEW executable TRENDS.EXE.

    - Observe the transient when you try to set the level setpoint of TANK2 to 1.5m.

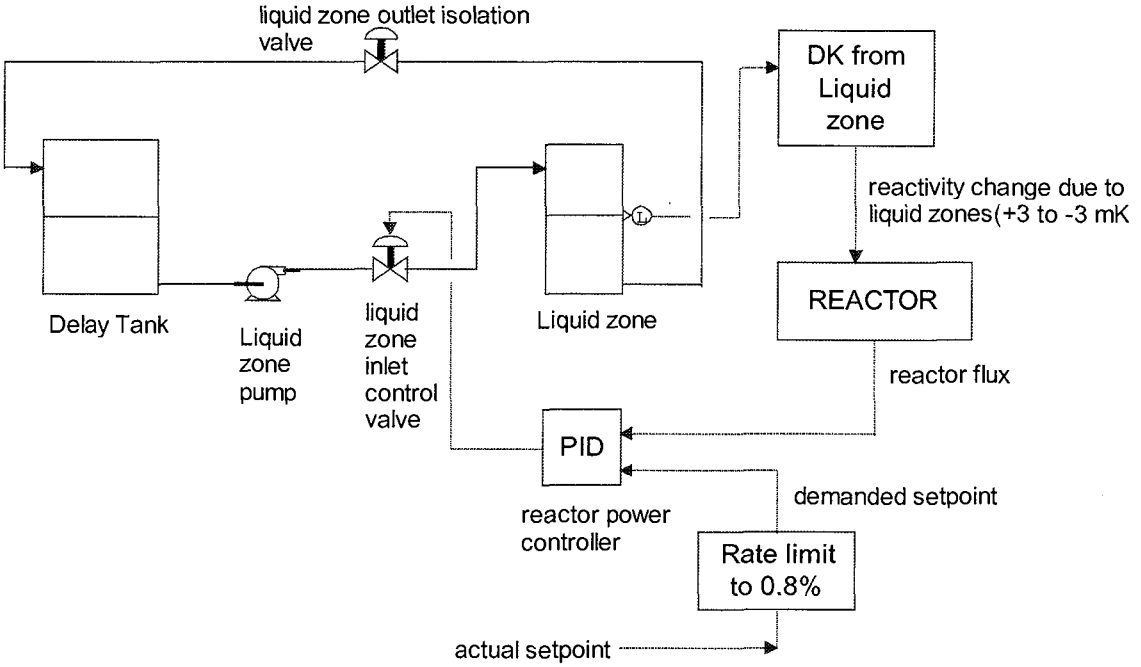
    - Use CASSBASE DEBUG functions to change controller constants to find optimum settings.

    - Remember to save the controller constants by pressing SAVE on the DEBUG screen in CASSBASE.

16. Operate set the level setpoint back to 2m.

- When steady-state is reached, save an Initial Condition(IC)
- Also save an IC file data into CASSBASE using the CASSIM Debugger.
- Exit CASSENG, and load up CASSBASE, check that the initial condition will now be transferred into the database. Now export a new Data File.

### 6.7.2 Liquid Zone Reactivity Control Modeling




**(A) PROBLEM DEFINITION**

Create a model to control the reactor power using the liquid zone.

- Include the effects of xenon poisoning.
- The reactor power can be controlled using a simple PID loop, though the reactor demanded setpoint should ramped towards the actual setpoint at 0.8% per second.

**(B) MODELING APPROACH**

1. Use CASSBASE and merge 3 model databases to form a combined Model IAEA4 :
   - the Point Kinetic Reactor Model with photoneutrons, IAEA1.
   - the xenon/iodine poison model, IAEA2.

- the liquid zone hydraulic model, IAEA3.

2. Delete the unnecessary NMB block using the DELETE BLOCK function.

3. Connect the reactivity change "output" from xenon/iodine poison block to the designated reactivity change "input" in the reactor block.

4. Create a block converts the TANK2 level to reactivity change, such that when TANK2 level is 4m, the reactivity change is -3mK, and when TANK2 level is 0m, the reactivity change is +3mK. (hint: use linear function Y = mX + c, and use generic algorithm ALG#311).

5. Connect the reactivity change due to liquid zone level to the reactor block.

6. Convert the level controller to a reactor power reactor controller, by reconnecting the process variable as the reactor flux.

7. The reactor setpoint should be limited to a rate of change of 0.8% per second. Create a block to calculate the demanded power setpoint by increasing it to the actual setpoint at the rate of 0.8% per second.(Hint: try ALG326, VR_LIMITER)

8. Connect the demanded reactor setpoint to the reactor power controller. Tuning constants may need to be changed.(Optimum Gain and reset time will be different)

9. Select the parameters you wish to trend (e.g. reactor flux, TANK2 level.) and connect it to the NMB block. Use the LABVIEW executable TRENDS.EXE to trend the variables.

10. Try lowering the reactor setpoint to 2%.

    - Can you control reactor power at 2%? If not, why?

    - If you are successful, can you raise the reactor power back to 100%? If not why?

    Any suggestions on what we might need ?

11. Use CASSBASE to extract the whole model out, and call it IAEA5.

12. Perform NETWORK GENERATE, and EXPORT on the new model. Have you notice any difference to the model?

### 6.8 Solution to Modeling Exercise - Reactivity Control Mechanism: Liquid Zone

To be included later in the class.

## 7.0 Development of Adjuster, Absorber, Shutdown Rods Model

### 7.1 Development of Adjuster and Absorber Rods Model

(A) *PROBLEM DEFINITION*:

- The overall reactivity change for a reactor can be expressed as:

$$\Delta K = \Delta K_C + \Delta K_P + \Delta K_V + \Delta K_{XE} + \Delta K_{SDS} + \Delta K_{FUEL} \quad .....(7.1)$$

where

$\Delta K$ = overall neutron reactivity change (K)

$\Delta K_C$ = neutron reactivity change due to control devices (K)

$\Delta K_P$ = overall neutron reactivity change due to power changes (K)

$\Delta K_V$ = overall neutron reactivity change due to channel voiding (K)

$\Delta K_{XE}$ = overall neutron reactivity change due to xenon poisoning (K)

$\Delta K_{SDS}$ = overall neutron reactivity change due to safety shutdown systems (K)

$\Delta K_{FUEL}$ = overall neutron reactivity change due to fuel burnup (K)

- The reactivity change due to control devices consists of reactivity change due to adjusters, absorbers and liquid zone.

$$\Delta K_C = \Delta K_{ADJ} + \Delta K_{CA} + \Delta K_Z \quad .....(7.2)$$

where

$\Delta K_C$ = neutron reactivity change due to control devices (K)

$\Delta K_{ADJ}$ = neutron reactivity change due to adjuster banks (K)

$\Delta K_{CA}$ = neutron reactivity change due to absorber banks (K)

$\Delta K_Z$ = neutron reactivity change due to liquid zone (K)

- We have already developed model for liquid zone. Now we will develop models for reactivity changes due to adjuster rods and absorbers rods.

- The reactivity change worth of the adjuster banks and absorbers are assumed to be:

| Normalized length of **Adjuster** rod inside core | Adjuster reactivity change (per unit rod worth) |
|---|---|
| 1.0 | 0.0 |
| 0.9 | 0.0 |
| 0.8 | 0.02 |
| 0.7 | 0.09 |
| 0.6 | 0.2 |
| 0.5 | 0.34 |
| 0.4 | 0.5 |
| 0.3 | 0.68 |
| 0.2 | 0.86 |
| 0.1 | 0.97 |
| 0.0 | 1.0 |

| Normalized length of **Absorber** rod inside core | Absorber reactivity change (per unit rod worth) |
|---|---|
| 0.0 | 0.0 |
| 0.1 | -0.00617 |
| 0.2 | -0.0544 |
| 0.3 | -0.1532 |
| 0.4 | -0.2525 |
| 0.5 | -0.3552 |
| 0.6 | -0.4809 |
| 0.7 | -0.6336 |
| 0.8 | -0.7907 |
| 0.9 | -0.9321 |
| 1.0 | -1.0 |

- We have already correlated the rod's position versus reactivity change :

$$\Delta K_{CA} = \sum_{i=1}^{2} ABSR_{iWORTH} * (MAX(-1, MIN(0, (-0.001267 + 0.13904 * S_{CAi} - 2.32103 * S_{CAi}^2 + 1.16576 * S_{CAi}^3)))$$

$$\Delta K_{ADJ} = \sum_{i=1}^{8} ADJ_{WORTH} * (MAX(0, MIN(1, (1.02293 - 0.484729 * S_{Ai} - 2.8621 * S_{Ai}^2 + 2.34474 * S_{Ai}^3)))$$

Where

$\Delta K_{CA}$ = neutron reactivity change due to absorber banks (K)

$\Delta K_{ADJ}$ = neutron reactivity change due to adjuster banks (K)

$ABSR_{iWORTH}$ = total reactivity change worth of absorber bank i(K)(i=1,2)

$ADJ_{WORTH}$ = total reactivity change worth of 1 adjuster bank (K)

$S_{CAi}$ = absorber bank i position (norm)(i=1,2); $S_{CAi}$ = 0 means absorber completely out of core; $S_{CAi}$ = 1 means absorber completely in core;

$S_{Ai}$ = adjuster bank i position (norm)(i=1,8); $S_{Ai}$ = 0 means adjuster completely pulled out of core; $S_{Ai}$ = 1 means adjuster completely in core.

- The total reactivity worth for the 2 absorber banks in core is -9 mk; the total reactivity worth for all 8 banks of adjuster out-of-core is + 17 mk.
- At full speed, an adjuster rod takes 60 seconds for full travel; at full speed, an absorber rod takes 150 seconds for full travel.

(B) *MODELING APPROACH*

- Develop simulation model "IAEA5" for absorbers and adjusters rods using generic algorithms. It is expected you will use ALG # 241, #301, #312, #322, #323.

### 7.2 Development of Shutdown Rods Model

(A) *PROBLEM DEFINITION*

- The reactivity change due to safety-shutdown rods are assumed to be:

$$\Delta K_{SDS} = \sum_{i=1}^{2} \frac{\Delta K_{SDSMAX}}{2} * (MAX(0, MIN(1, (SDS_{iPOS}))))$$

$\Delta K_{SDS}$ = neutron reactivity change due to shutdown rods (K)

$\Delta K_{SDSMAX}$ = reactivity change due to shutdown rods when fully inserted(K)

- The total reactivity worth for the 2 shutdown rods is - 68 mk, when fully inserted in the core. The maximum time for full rod drop should be less than 2 sec.

(B) *MODELING APPROACH*

- Develop the Shutdown Rod model "IAEA6" using generic algorithms. By now, you probably know which algorithms to use.
- Design a reactor tripping logic. Make this trip is a latch in once set.

trip the reactor if d(log n)/dt > 10 % per sec.

Can you model this using generic algorithm?

- Integrate IAEA4, IAEA5 and IAEA6 to form one combined model IAEA7.
- Make the necessary connections and create blocks, if necessary to add the individual reactivity contributions together.
- Now test the model by manually tripping the reactor at full power. Monitor xenon-iodine poison buildup. Try to speed up the xenon poison buildup, if you can.
- Now remove the shutdown rods manually by resetting the trip "latch-in".
- Use the liquid zone to raise reactor power. If liquid zone cannot recover full power, use adjuster rod to help. Do it slowly, and make IC points on the way.

## 8.0 Reactor Startup and Approach to Criticality

The commissioning of PHW reactors usually takes place in four distinct phases:

- Phase A - Fresh fuel is loaded into the reactor; heat transport system is filled with heavy water. Moderator is overpoisoned — Guaranteed Shutdown State (GSS) (> - 475 mk).
- Phase B - Approach to first criticality, check-out of reactivity mechanism.
- Phase C - Gradual runup to full power.
- Phase D - Reactor at full power and establish normal operation.

The approach to critical is a procedure that is *undertaken with a great deal of respect*, because during this time, the reactor is in the most "dangerous" condition. The reasons are as follows:

1. The available reactivity is at its maximum, since the fuel has suffered no burnup and fission product poisons do not exist yet.
2. The regulating system will be inoperative because it is "off-scale" at the low count rates encountered here. The approach to criticality is therefore manual, rather than an automatic one.
3. The response of the neutron instrumentation depends on the flux density in the sense that the speed of response is slow at low count rates. At very low power, this could result in high power increases occurring without them being detected quickly enough.
4. The critical value of the variable that is used for the startup is unknown, For instance, if the approach is used by gradual removal of poison, then how much of poison removal is enough for criticality is unknown.

So in this section, we are going to experience a simulated reactor startup situations using the CANDU simulator, which will illustrate the *important points* as mentioned above.

### 8.1 Subcritical Reactor Behavior

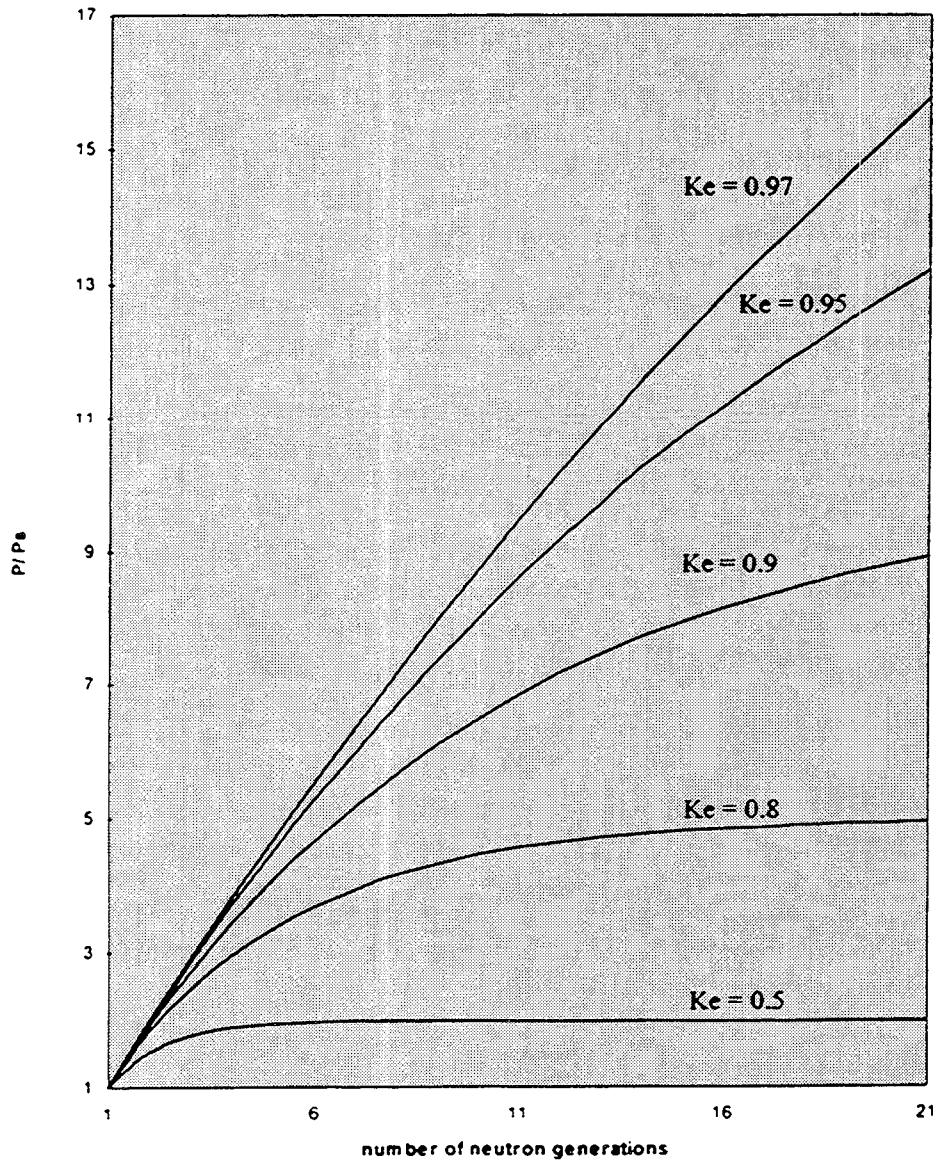
During the approach to critical, the reactor will be subcritical (Ke <1). In previous chapter and modeling exercise, we have discussed the behavior of a subcritical reactor. Let us review the essential concepts here again.

In earlier chapter, we discuss the effect of neutron sources on the total neutron population in a reactor:

$$S_t = \frac{S_0 \Lambda (1 - K_e^m)}{(1 - K_e)} \quad \text{.....(8.1)}$$

where $S_t$ is the neutron population ; $S_0$ is the neutron source; m is the number of neutron generations ( m = t/$\Lambda$, i.e. the time duration after source is introduced divided by the neutron generation time).

Equating $S_t$ and $S_0$ to neutron power P, Ps respectively, we have $S_t/(S_0\ \Lambda)$ = P/Ps. If we plot P/Ps versus m, with different value for Ke (i.e. Ke = 0.5, 0.8, 0.9, 0.95, 0.97), we have:



For a subcritical reactor, as Ke is increased, the power levels off more slowly. As Ke is increased further step by step, a point will finally be reached when the power will increase steadily without leveling off at all, and this is the point where criticality (sustained nuclear fission chain reaction) has been reached i.e. Ke =1. This is the basic method we use for approaching criticality, by gradually increasing Ke. First let us review the reactor startup instrumentation which makes this startup possible.

### 8.2 Reactor Instrumentation

Three instrumentation systems are provided to measure reactor thermal neutron flux over the full power range of the reactor.

- Startup instrumentation covers the eight-decade range from $10^{-14}$ to $10^{-6}$ of full power;
- The ion chamber system extends from $10^{-7}$ to 1.5 of full power.
- The in-core flux detector system provides accurate spatial measurement in the uppermost decade of power (10 percent to 120 percent of full power).
- The fuel channel temperature monitoring system is provided for channel flow verification and for power mapping validation.

Figure 16 illustrates the range of sensitivity of nuclear instrumentation for reactor power measurement.

#### 8.2.1 Startup Instrumentation

This system is used on initial plant startup or after a prolonged shutdown, and performs a dual regulating and protective role over the lowest power range when the normal flux measuring instrumentation of the regulating and shutdown systems is off scale. This occurs on the initial reactor startup and during a shutdown of 40 to 70 days or more when the photo-neutron source decays below the sensitivity of the ion chamber system.

Two sets of neutron detectors ($BF_3$ counters) are used covering the range from $10^{-14}$ of full power up to $10^{-6}$ of full power. One set, *located in-core*, covers the range from spontaneous source level ($10^{-14}$ full power) to $10^{-9}$ full power. The in-core detectors are installed via the vertical viewing port penetration from the reactivity deck. *This instrumentation is removed after the low power commissioning period.*

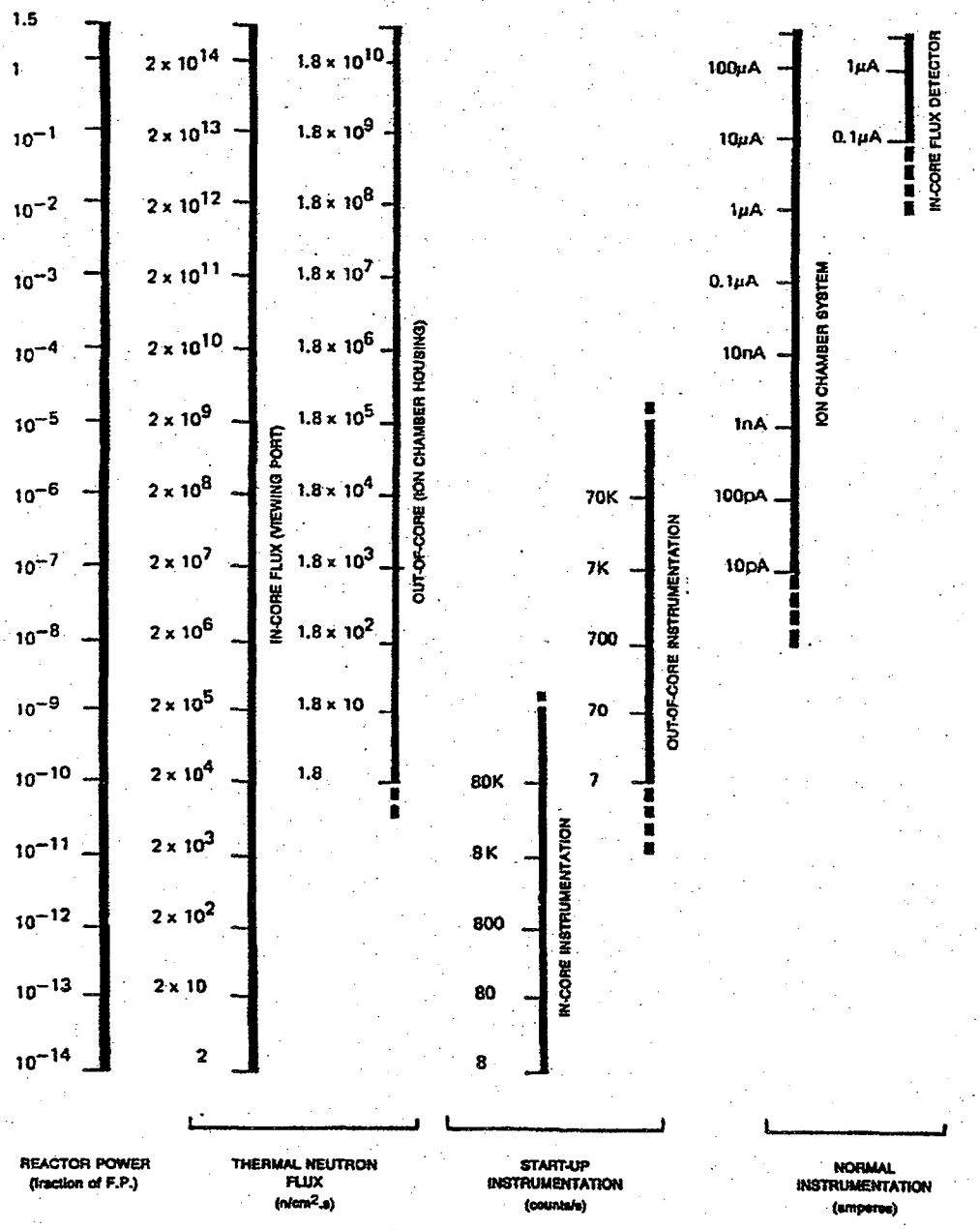



*Figure 16 - Range of sensitivity for different reactor instrumentation.*

The other set of neutron detectors is *located out of core* in the three spare cavities of the shutdown system number 2 ion chamber housings.

The detector outputs are connected to startup monitoring instrumentation located temporarily in the main control room. The signals are processed to give log and linear count rate indications and rate of change information. This information is used for protection and monitoring during the approach to criticality.

In this range, the reactor is controlled manually by the operator from the control room. The operator monitors and records the output of the neutron counting instrumentation which is mounted in a mobile cabinet. This instrumentation is divided into three redundant channels and includes trip comparators which are connected into the D, E and F channels of the trip logic of the shutdown system number 1. Shutdown rods will drop into the core under any one of the following three conditions:

- The count rate exceeds the setpoint,
- The rate of change is excessive,
- Two or more channels of the instrumentation fail.

On restart after an extended shutdown, the system is normally on-scale with the out of core detectors; in-core counters are used only on the initial startup. The arrangement of the system is shown in the following Figure 17.

The in-core detectors (or sometimes called counters, or ratemeters), have the flexibility of being moved further out from the core to prevent them from saturating due to increased neutron density during reactor startup. Typically when counts in the detectors at a particular core position exceed a certain counts per sec., it is required to move the detectors further out to another core position in order to keep its counting rate within the useable range.

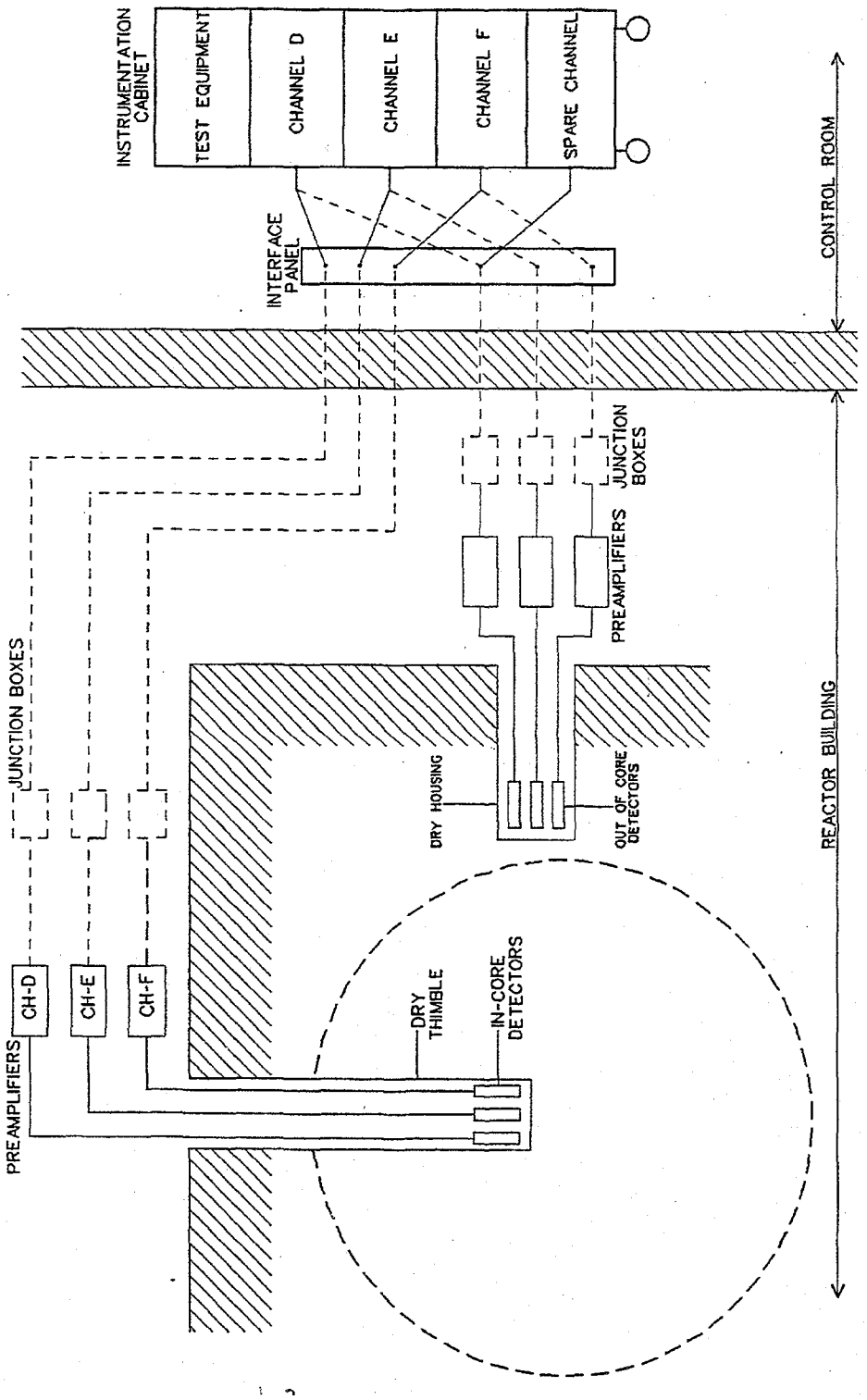



Figure 17 - Arrangement of Startup Instrumentation.

### 8.2.2 Ion-Chamber Instrumentation

The ion chamber system is a part of the on-line power measurement equipment of the reactor control system. This instrumentation is on-scale during normal operation, and remains on-scale during a normal shutdown. If the shutdown lasts more than 40 to 70 days (depending on reactor operating history and ion chamber neutron and gamma sensitivity), the flux decays below $10^{-7}$ of the full power value and startup neutron detectors are needed.

The ion chamber units consist of lead-shielded housings mounted on the outside of the calandria shell, in which ion chamber instruments and calibration shutters are installed. The lead attenuates gamma radiation so that the ion chambers measure neutron flux, primarily. The output current from separate ion chamber instruments is amplified to generate independent inputs for the reactor control system and each shutdown system, as follows:

a. log neutron power, $10^{-7}$ to 1.5 full power,

b. linear neutron power, 0 to 1.5 full power,

c. rate of change of log power, -15 percent to +15 percent of present power per second.

Ion chambers measure only the average flux, as seen at the side of the core, but are sensitive to very low flux levels, for monitoring sub-critical as well as full power behavior. Because they are outside the calandria, the neutron flux they detect has been attenuated by the moderator and any poison dissolved in the moderator.

### 8.2.3 In-Core Flux Detectors

In-core flux detectors are used at high power levels (above 10 percent of full power) because they provide spatial information needed, at high power, to control xenon-induced flux tilts and to achieve the optimum flux distribution for maximum power output.

- *Vertical Flux Detector Assemblies*

The control system flux detectors are of two types. One type has an Inconel emitter and is used for the zone control system. The other type has a vanadium emitter, and is used for the flux mapping system.

Both types of control system detectors, along with the Inconel detectors used for the shutdown system number 1 high neutron power trip, are contained in vertical flux detector assemblies located throughout the core. These assemblies extend from the reactivity mechanisms deck to the bottom of the reactor.

- *Zone Control Flux Detectors (Inconel)*

The self-powered in-core flux detectors are installed in flux detector assemblies to measure local flux over two decades, $10^{-1}$ full power to 1.2 full power, in the regions associated with the liquid zone controllers. At each location there are two detectors for redundancy.

- *Flux Mapping Detectors (Vanadium)*

The flux mapping system (described in previous Section) uses vanadium detectors distributed throughout the core to provide point measurements of the flux. These detectors are almost totally neutron-sensitive; however, their response is delayed by 5.4 minutes due to the half-life of the beta emission from vanadium-52. The signals, approximately 3 μA at full power, are fed to amplifiers in a stand alone distributed control system station inside the reactor building, where they are multiplexed and sent to the plant display system computer.

### 8.3 Methods for Approaching Criticality

For a PHW reactor such as CANDU, there are two methods for approaching criticality:

1. Remember $K_{eff} = \varepsilon\ \eta\ \mathbf{f}\ P_f\ \mathbf{p}\ P_{th}$. For large reactor, leakage is very small, hence the fast neutrons non-leakage probability $P_f \sim 1$ and the thermal neutrons non-leakage probability $P_{th} \sim 1$. So the idea for approaching critically is to maintain moderator level at some low level initially to incur large neutron leakage; then keeping the four factors unchanged, gradually reduce leakage by raising moderator level until enough fuel is covered to sustain a chain reaction.

2. An alternate method is start at a certain moderator level (nominally near full calandria) with a high enough boric concentration in the moderator to ensure that criticality cannot be possible. The boron "poison" is then gradually removed until criticality is reached. In this case, the leakage is held constant, and the value of f, the thermal utilization factor is increased until Ke becomes equal to 1.

The first method was used in earlier generation of CANDUs - NPD, Douglas Point and Pickering Units 1 and 2. Later generations of CANDUs such as Bruce, Darlington, which do not have moderator level control; thus they use the second method for approaching criticality.

*Note that these two approaches to critical do not of course have to be repeated every time the reactors are shutdown.* Under normal conditions, the photoneutron source provided by the fission product gamma reactions with deuterium is sufficiently large for the reactor to be started using automatic regulation. Only when a reactor has been shutdown for a period of 3 or 4 months, by which time this photoneutron source will have decayed, will a manual startup be necessary again.

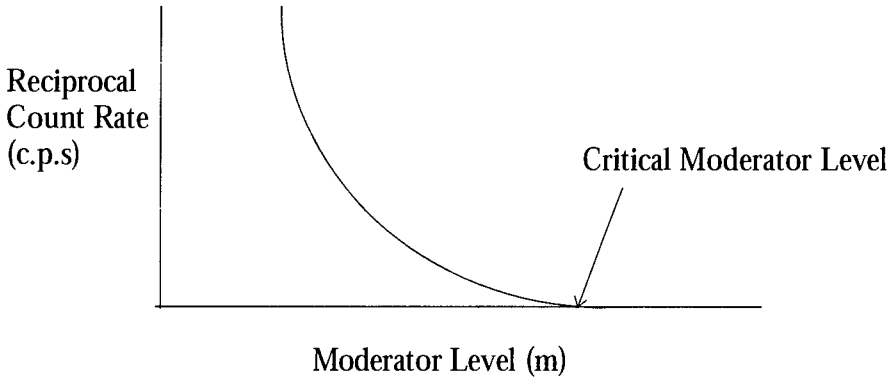
Reactors which do not have a natural photoneutron sources like CANDUs are usually equipped with an artificial source, in the sense that some Beryllium is included in the core which undergoes (γ, n) reactions in much the same way as deuterium.

### 8.3.1 Approach Reactor Criticality by Raising Moderator Level

In the first method, the approach to criticality is monitored by devising an approximate linear plot which can readily be extrapolated to predict the critical moderator level. From equation 8.1, you will see that

$$\frac{1}{count\ \ rate} = \text{constant x ( 1 - Ke)} \quad \text{.....(8.2)}$$

As the moderator level is raised in small steps, the count rates will increase, and the reciprocal count rates are plotted against moderator level. A plot of inverse count rate versus moderator level will look like this:




### 8.3.2 Approach Reactor Criticality by Removing Boron Absorber

In the second method, the approach to criticality is monitored by devising an approximate linear plot which can readily be extrapolated to predict the critical Boron concentration. It was discussed previously that :

$$\frac{1}{count\ \ rate} = \text{constant x ( 1 - Ke)}$$

As you recall the definition for $\Delta K$, this becomes

$$\frac{1}{count\ \ rate} = A \cdot \frac{\Delta K}{K_e}$$

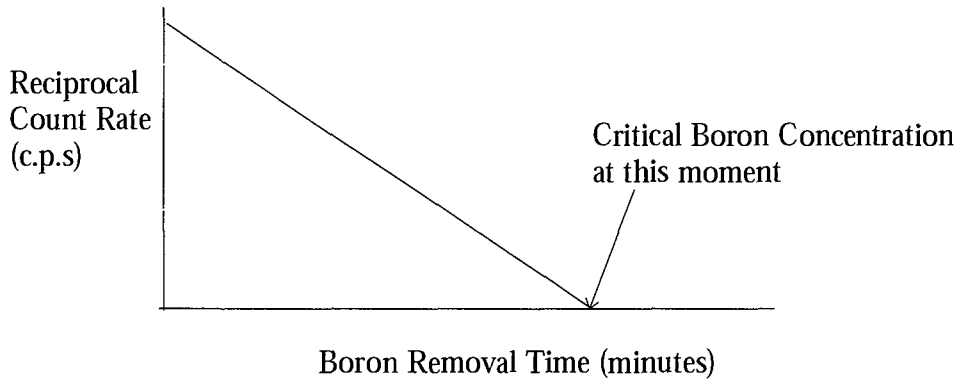
$$= A \cdot B \cdot (C - C_{critical}) \quad \text{.....(8.3)}$$

where A, B are proportionality constants.

C - Boron concentration at time t.

$C_{CRITICAL}$ - Boron concentration at criticality.

The procedures used are as follows: the count rate is measured for the initial boron concentration, and the moderator purification circuit is then valves in. Boron is removed for an interval of time ~ 20 minutes, and the count rate were recorded again. This procedure is repeated a number of times and the inverse count rate is plotted against total boron removal time.




In conjunction with the measurement of inverse count rate vs. Boron removal time as an indication of criticality, liquid zone control response is also used by some CANDU station (e.g. Darlington GS) for indication of criticality.

The theory is simple: if all the adjuster rods are in "Manual", and the liquid zone control is in "Auto". As Ke gradually approaches 1 by slowly removing Boron, liquid zone controllers always control water level at setpoint, thus making zero reactivity contribution from liquid zone. As Ke is very close to 1, and if Boron removal is stopped at that time, and one makes a small step change in reactor power increase by RRS, the average liquid zone level will drops (for power increase), *if in fact Ke =1, after the desired power level is reached, the liquid zone level will eventually return to its original level setpoint after a few minutes*. However, if in fact Ke is less than 1, as we explained earlier, the neutron power after initial increase will *level off quickly*. Because reactor power cannot reach setpoint, liquid zone level cannot recover quickly to setpoint, and in fact it will continue to drop in order to contribute reactivity for power increase. This is the indication of a subcritical reactor, so Boron removal is resumed for some time, and then the small step change in reactor power is repeated and the liquid zone level response will be assessed again to determine criticality. In the simulated reactor startup that we are going to practice, we will use the liquid zone response technique to determine reactor criticality.

## 8.4 Practice Reactor Startup and Approach to Criticality Using Simulator

1. First load the simulator up and initialize the simulator with the IC = GSS_SD.ic (that is, Reactor in Guaranteed Shutdown State with moderator overpoisoned).

2. In the Plant Overview screen, you will see the overall plant condition from the annunciators: Turbine trip; Primary Heat Transport System is depressurized at low pressure < 300 KPa, at temperature 54 deg. C; PHT pumps shutdown.

3. Go to another screen called Reactor SU/SD (startup/shutdown) controls. Here the display represents the reactor startup instrumentations in combination with certain controls for RRS and Liquid Zone. The details are described as follows :

   (a) Special Liquid Zone Controls under the box titled "LIQUID ZONE CONTROL MODE". A control push-button is provided for switching the Liquid Zone Control between "Standard" and "Special Shutdown" Mode. Some explanations are necessary for these two modes - under normal operating conditions, that is when reactor power is > 0.1 %, the liquid zone controller is in "standard" mode, that is, the zone level controller signal is proportional to the combination of reactor power errors and flux tilt errors components. However, when reactor power is less than 0.1 % and at such low power, flux tilt control is not necessary and operator will put the zone controller in " Special Shutdown" Mode, meaning that the zone level controller now responds to local level setpoint as inputted by the operator. The zone level controller is now proportional to the level errors only. As you will see later, as we approach criticality, the liquid zone controller will be "kicked" out of "Special Shutdown Mode" into "Standard" mode, when power level is > 0.1 % (-3 decades) and there is a positive power error present.

   (b) The reactor startup instrumentations under the box titled "POWER LEVEL READINGS". The instrumentation simulated here are :

   - the in-core detectors (counters), which have 3 ranges : Long (long distance from the core); Medium (middle position in the core); Short (short distance from the core). The IC indicates it is now in "Long" range.

   - as well, the Ion Chambers and Flux Detectors are simulated, but they are all off-scale, meaning the reactor power is below -6 decades.

   (c) The reactor power maneuvering rates - PWR LOG rate (DEC/sec); and power error (DEC) are also indicated here for monitoring

purposes. The Xenon Load (mk) normally would not be displayed in real situation, but is displayed here for our study and analysis.

(d) SDS1 Reactor trip Setpoint can be set here under titled box "SDS1 Trip Setpoint". For safety concern, you always set trip setpoint just slightly above the current the reactor power level, to prevent the unlikely situation of reactor power running away. You can set the setpoint in % full power or in decades. Currently the SDS1 setpoint is set at -2 decades. Remember to raise the SDS1 trip setpoint as the reactor power level reaches -2 decades. In real situation, there will be a warning message informing a "pending" reactor NOP (neutron overpower) trip, when the trip setpoint is approaching.

(e) Specific RRS controls are included to provide:

- Alternate Mode reactor power setpoint input. This setpoint input will be processed by RRS to indicate an allowable "Demanded Power Setpoint", which is a slowly ramped setpoint calculated by the RRS.

- RRS Startup Flag Control - ON/OFF. Note RRS Startup Flag "On" means that the RRS will use a minimum reactor power of -5.9 decades in its calculation.

- Hold Power control - to hold reactor at any time.

(f) Poison Controls are included here for adding (pointer "ADDITION"), removing (pointer "REMOVAL"), or stopping (pointer "STOP") the boron process. For the boron removal, there is a button that you can use to speedup the boron removal process, but *use it with care*! This speedup is not applicable in real life situation. The Boron poison load is also indicated for our study and analysis. It is normally not displayed in real situation. Right now, the boron load is approximately - 60 mk.

(g) The graphical trends are provided here for count rate (counts per sec); power log rate (dec/s); ion-chamber detectors (dec); boron poison load (mk); power error (dec); average zone level (%).

Let us follow the procedures outlined in the table below to perform our simulated reactor startup and approach to criticality. The sequence of steps to be performed should follow the numerical order of the steps indicated. The symbol $\Rightarrow$ only gives you the highlights on the statuses of the subsystem conditions.

**Note:**

(1) For our purpose, we can ignore the first step regarding operations in the PHT regarding the warming up of the pressurizer liquid, and go to Step #2 right away.

(2) The currently simulated startup counters will saturate at 6000 cps at the "Long" position. One *should* take the approach of plotting the inverse counts rate versus total boron removal time in order to arrive at the extrapolated "critical" boron concentration. But to save time for our exercise, we are using the method of *liquid zone controller response* as an approximate indication of "criticality".

| Operating Phases | Unit Load | Reactor RRS | PHT | SDC | SGPC | Check/Monitor |
|---|---|---|---|---|---|---|
| **1.** Initial State: De-pressurized cooldown state: HTS temperature < 54°C and pressure < 300 KPa. Reactor has been down for 4 weeks. | ⇒ Reactor power level depends on how long the reactor has been 'down' for. The last group of photo-neutrons modeled has a half life of the order of days. | ⇒ Liquid zones control in Special S/D mode with a level setpoint of 40%.<br>⇒ Adjusters control in manual<br>⇒ Absorber control in auto.<br>⇒ Ion chamber will be off-scale if the reactor power < -6.25 Decades | ⇒ HTS in a cold de-pressurized state:<br>⇒ HTS temperature < 54°C<br>⇒ HTS pressure < 300 KPa<br>⇒ HTS inventory control is in 'SOLID' mode as both the pressurizer and the bleed condenser are isolated<br>⇒ Bleed flow is controlled by CV15, the bleed condenser level control valves and the pressure letdown valve MV10 which is opened at 10%. | ⇒ Both side of the SDC is in service under 'SDC pump' mode.<br>⇒ For SDC #2, the flow path is MV23, MV45, SDC P2, MV31 (bypass MV18 is closed), Bleed cooler, MV20, MV28, MV29 to RIH #1 & #3.<br>⇒ Some bleed to MV10 (10% open) & CV15(Bleed condenser LCV) to Purification. The tempering valve MV19 should be closed. | ⇒ In 'Hold' mode with a SG pressure setpoint of 461 KPa | |
| **1.** Continued: Initial State: De-pressurized cooldown state: HTS temperature < 54°C and pressure < 300 KPa. Reactor has been down for 4 weeks. | | | | ⇒ For SDC#1, the flow path is MV12, MV46, SDC P1, MV2, SDC HX, MV8, MV9 to RIH #2 & #4. The bypass line valve MV4 is closed and the Tempering line MV5 is closed. | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2. Reactor Approaching Criticality - Phase1 | | 2. **Set RRS Startup flag to 'ON'.** *(from screen Reactor SU/SD Controls)*<br><br>⇒ Startup flag will ensure that the RRS use a minimum reactor power value of -5.9 decades in its calc.<br><br>3. **Remove Poison by IX columns until Ion Chambers are on scale and flux detector's response to poison removal is significant (power changes within 1-2 minutes).** *(from screen Reactor SU/SD Controls)* | 1. **Turn all pressurizer heaters to 'auto' and set the solid mode pressurizer pressure setpoint to 7.3 MPa** *(from screen PHT Pressure Control)*<br><br>⇒ The pressurizer may take a long time to warmup depending on its initial condition. (warmup time can take up to 9 hours)<br><br>⇒ Speedup control is available to artificially speedup warmup process | ⇒ Circuit #1 & 2 both in service. | ⇒ In 'Hold' mode with a SG pressure setpoint of 461 KPa | ⇒ SDS1 poised with NOP set at LOW POWER TRIP= 1 %<br><br>⇒ Moderator Poison addition available.<br><br>⇒ Note Power Level and Zone Level |
| 2. Continued: Reactor Approaching Criticality - Phase1 | | 4. **If Ion Chamber is less than -5.9 decades, input demand power of -5.9 decades.** *(from screen Reactor SU/SD Controls)*<br><br>5. **Place all adjuster rods on Manual and fully insert them into the core** *(from screen Reactivity Control)*<br><br>⇒ LZCS remains in Special SD with a level setpoint of 40% | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| 3. Reactor Approaching Criticality- Phase 2<br><br>3. Continued: Reactor Approaching Criticality- Phase 2 | | **6. Stop poison removal when Ion Chamber indicates reactor power > -6 decades.** *(from screen Reactor SU/SD Controls)*<br><br>**7. Raise power by inputting a reactor power setpoint which is 0.3 decade above the present power.** *(from screen Reactor SU/SD Controls)*<br><br>**8. Monitor the average zone level. If the average level should drop below 25 %, press the 'HOLD POWER' button.** *(from screen Reactor SU/SD Controls)*<br><br>⇒ Note: the average zone level is available as one of the 6 trends on the screen 'Reactor SU/SD Controls'<br><br>**9. Resume poison removal so that the zones must rise to hold power. Monitor the zone level rises and the reactor power. As** | | | | ⇒ Monitor time to double the count, which gives some indication of proximity to Criticality. Adjust range if count saturates<br><br>⇒ LZC will be "kickout" of Special S/D Mode on positive power error. |

| | | | | | | |
|---|---|---|---|---|---|---|
| 3. Continued: Reactor Approaching Criticality- Phase 2 | | **the zones rises to 70 %, stop poison removal. Note reactor power and zone level.** *(from screen Reactor SU/SD Controls)*<br>**10. Repeat Steps 7, 8, 9 until after a reactor power SP increase, the average zone level drops, and returns to within 5% of the original level after a few minutes.** *(from screen Reactor SU/SD Controls)*<br>**11. Stop Poison removal. Enter a slight power SP decrease. If the zone rises and power drops and then the zones return to the original** | | | | |
| | | **level (+/- 5 %), then Reactor is declared CRITICAL.** *(from screen Reactor SU/SD Controls)*<br>**12. Resume Poison removal until the average zone level is around 60 %, and power holding.** *(from screen Reactor SU/SD Controls)* | | | | |

## 9.1 Spatial Kinetic Model for PHW Reactor

Now that we have developed a simple point kinetic reactor model with 6 fission products groups and 9 photoneutron groups, it would be a logical evolution at this point to consider a more sophisticated model such as the spatial kinetic reactor model. In the following sections, we will be discussing the techniques used in building such model. Because the development of a spatial kinetic reactor model will involve certain amount of time, so we might just discuss the underlying theories and the techniques and it is hoped that the materials that we cover here will provide enough background materials for the students to try out the simulation development by themselves.

## 9.2 Nodal Approach for Simulating Space-Time Reactor Kinetics

Traditionally, two different approaches have been proposed in the real-time simulation of space-time kinetics. One Approach is based on modal kinetic formulation (1). The other approach is based on Avery's coupled region kinetics theory (2).

In a typical coupled nuclear reactor model, the reactor core is divided into a number of nodes axially and radially. The usual considerations for the choice of the nodes are the core symmetry and the accuracy required in the description of neutron flux distributions, and the execution time of the nodal kinetic model. For a reactor core divided into 14 zones, the temporal nodal fluxes are computed by the following nodal kinetic equations using Avery formulation (3).

For Zone i,

$$l_i \frac{dN_i}{dt} = (1-\beta)\sum_{j=1}^{14} K_{ij}N_j - N_i + \sum_{j=1}^{14} K_{ij}\sum_{m=1}^{6} \lambda_m C_{mj} \ldots\ldots\ldots (9.1)$$

$$i = 1,2,3.....14$$

$$\frac{dC_{mj}}{dt} = \beta_m N_j - \lambda_m C_{mj} \ldots\ldots\ldots\ldots\ldots (9.2)$$

$$j = 1,2,.....14$$

$$m = 1,2,.....6$$

where:

$N_i$ = Neutronic fluxes in Zone i, respectively (nodal fluxes)

$l_m$ = Decay constants of the mth delayed neutron group

$b$ = Total delayed neutron fraction

$\beta_m$ = Delayed neutron fraction of the mth group

Kij = "Coupling Coefficient" determining the probability of a neutron born in zone j producing a fission neutron in zone i in the next generation.

$\lambda mCmj$ = Partial power of zone j contributed from the mth delayed neutron group.

Cmj = Concentration of delayed neutron group m in zone j

$l_i$ = Mean neutron life time

Equation (9.1) can be rewritten by regrouping the coupling coefficients for zone i,

$$\frac{dN_i}{dt} = \left\{(1-\beta)K_{ii} - 1\right\}\frac{N_i}{l_i} + \frac{K_{ii}}{l_i}\sum_{m=1}^{6}\lambda_m C_{mi} +$$

$$\frac{(1-\beta)}{l_i}\sum_{j=1,j\neq i}^{14}K_{ij}N_j + \frac{1}{l_i}\sum_{j=1,j\neq i}^{14}K_{ij}\sum_{m=1}^{6}\lambda_m C_{mj}$$

$$i = 1,2,3.....14 \quad \text{.....(9.3)}$$

The above respective terms represent the various contributions of neutronic flux changes in zone i from the following sources:

(a)

$$\left\{(1-\beta)K_{ii} - 1\right\}\frac{N_i}{l_i}$$

is the rate of neutronic flux changes in zone i due to the its zone reactivity.

(b)

$$\frac{K_{ii}}{l_i}\sum_{m=1}^{6}\lambda_m C_{mi}$$

is the rate of neutronic flux changes in zone i due to its concentration of delayed neutron groups.

(c)

$$\frac{(1-\beta)}{l_i}\sum_{j=1,j\neq i}^{14}K_{ij}N_j$$

is the rate of neutronic flux changes in zone i due to the coupling effects of the neutronic fluxes in the other 13 zones.

(d)

$$\frac{1}{l_i}\sum_{j=1,j\neq i}^{14} K_{ij}\sum_{m=1}^{6}\lambda_m C_{mj}$$

is the rate of neutronic flux changes in zone i due to coupling effects from the concentration of delayed neutron groups in the other 13 zones.

By introducing the definition of reactivity ΔKi = (Kii -1)/Kii, equation (9.3) can be written as:

$$\frac{dN_i}{dt} = \frac{(\Delta K_i - \beta)}{\Lambda_i} N_i + \sum_{m=1}^{6}\lambda_m^* C_{mi} + \alpha_i \sum_{j=1,j\neq i}^{14} K_{ij} N_j \; + \; \frac{1}{l_i}\sum_{j=1,j\neq i}^{14} K_{ij}\sum_{m=1}^{6}\lambda_m C_{mj} \quad .....(9.4)$$

Where

$$\Lambda_i = \frac{l_i}{K_{ii}}$$

$$\alpha_i = \frac{(1-\beta)}{l_i}$$

$$\lambda_m^* = \frac{\lambda_m}{l_i}$$

Equation (9.4) is almost identical to point kinetic model for reactor zone i, with the exception of an extra zone coupling source terms:

$$\alpha_i \sum_{j=1,j\neq i}^{14} K_{ij} N_j \quad \text{and} \quad \frac{1}{l_i}\sum_{j=1,j\neq i}^{14} K_{ij}\sum_{m=1}^{6}\lambda_m C_{mj}$$

If we absorb the "zone coupling effects" into "the zone i reactivity term Δρii" and rewrite the point kinetic equations for zone i:

$$\frac{dN_i}{dt} = \frac{(\Delta\rho_{ii} + \sum_{j=1,j\neq i}^{14}\Delta\rho_{ij} - \beta)}{\Lambda_i} N_i + \sum_{m=1}^{6}\lambda_m^* C_{mi} \quad .....(9.5)$$

Where

Δρii is the zone i reactivity change.

Δρij is the reactivity change for zone i due to coupling effects in zone j

Equation (9.4) and (9.5) will be identical if

For zone 1 & 2, $$\frac{\Delta\rho_{12}\cdot N_1}{\Lambda_1}=\alpha_1 K_{12}N_2+\frac{1}{l_1}\left(K_{12}\sum_{\substack{ZONE2\\m=1}}^{6}\lambda_m C_m\right)$$

For zone 1 & 3, $$\frac{\Delta\rho_{13}\cdot N_1}{\Lambda_1}=\alpha_1 K_{13}N_3+\frac{1}{l_1}\left(K_{13}\sum_{\substack{ZONE3\\m=1}}^{6}\lambda_m C_m\right)$$

For zone 1 & 4, $$\frac{\Delta\rho_{14}\cdot N_1}{\Lambda_1}=\alpha_1 K_{14}N_4+\frac{1}{l_1}\left(K_{14}\sum_{\substack{ZONE4\\m=1}}^{6}\lambda_m C_m\right)$$

•

•

For zone 1 & 14, $$\frac{\Delta\rho_{1,14}\cdot N_1}{\Lambda_1}=\alpha_1 K_{1,14}N_{14}+\frac{1}{l_1}\left(K_{1,14}\sum_{\substack{ZONE14\\m=1}}^{6}\lambda_m C_m\right)$$

*Repeat these similar equations for all other 13 zones*

Therefore the equation for Δρij, the reactivity change for zone i due to coupling effects in zone j is:

$$\Delta\rho_{ij}=\Lambda_i K_{ij}\left(\alpha_i\frac{N_j}{N_i}+\frac{1}{l_i}\frac{\sum_{\substack{ZONEj\\m=1}}^{6}\lambda_m C_m}{N_i}\right) \quad \text{.....(9.6)}$$

It can be seen that equation (9.6) involves the calculation of coupling coefficient Kij. These coefficients Kij define the probability of a neutron born in node j producing a fission neutron in node i in the next generation.

In Avery's formulation(2), the coupling coefficients for the two energy groups of neutrons are given by the following equations:

$$kij=\frac{\int_i \nu\sum_f \phi_{th}(r)\,dr}{\int_j \nu\sum_f \phi_{th}(r)\,dr}*\frac{\int_i \phi^*_f(r)\,\nu\sum_f \phi_{jth}(r)\,dr}{\int_i \phi^*_f(r)\,\nu\sum_f \phi_{th}(r)\,dr} \quad \text{.....(9.7)}$$

where:

$\sum_f$ =fission neutron production cross section

$\phi_{th}(r)$ = real thermal flux at position r

$\phi^*_f(r)$ = adjoint fast flux at position r

$\phi_{jth}(r)$ = contribution to $\phi th(r)$ from fission neutrons produced in node

j in the previous generation with distribution $\Sigma_f \phi_{th}(r)$

Kij involves the computation of the distribution of the real thermal flux and adjoint fast fluxes which usually involves a lot of CPU time. It would be impractical to compute these fluxes in real-time environment. To deal with this problem, an approximate method (3) is implemented to compute the coupling coefficients in real time using a perturbation form (relative to the equilibrium values):

$$K_{ij} = K_{ij}^0 + g_{ij} \quad \text{.....(9.8)}$$

where $K_{ij}^0$ = nominal value of Kij (nominal coupling coefficients)

$g_{ij}$ = change in coupling coefficients in node i due to perturbation

in node j

$g_{ij}$ can further be defined as

$$g_{ij} = \sum_{k=1}^{14} P_k K_{ijk} \quad \text{.....(9.9)}$$

where $P_k$ = net reactivity perturbation in zone k (mk).

$K_{ijk}$ = change in coupling coefficient in node i due to node j with +1.0 mk change in reactivity in node k (normalized perturbation gradient).

Rewriting equation (9.8),

$$K_{ij} = K_{ij}^0 + \sum_{k=1}^{14} P_k \cdot K_{ijk} \quad \text{.....(9.10)}$$

- It can be seen that the implementation of this approximation method involves two major components - the nominal coupling coefficients, $K_{ij}^0$, and the normalized perturbation gradients, $K_{ijk}$.

- $K_{ij}^0$ can be computed off-line for a given (nominal) core condition and configuration by using equation (9.7), for all i, j.

- The core is perturbed from nominal core condition by changing reactivity in one zone, say zone #1 and equation (9.7) is used again to

calculate the new coupling coefficients Kij off-line, for all i, j. With the use of equation (9.10), $K_{ij1}$ can be obtained for all i, j. Repeat the same for other zone to obtain $K_{ijk}$, for k= 2, ...14.

- Store $K_{ij}^{0}$ and $K_{ijk}$ as coefficients in the computation for $\Delta\rho_{ij}$ in equation (9.6).

Equation (9.6) is computed every simulation iteration, and $\Delta\rho_{ij}$ is obtained for all i, j and they provides inputs to a summer $\sum_{j=1, j\neq i}^{14}\rho_{ij}$, which in turn inputs into the reactivity change "input" of the affected reactor zone.

### 9.3 Summary of Model Formulation for PHW Reactor

Here is the summary of the essential modeling details for the spatial kinetic model:

- 14 point kinetic models are used respectively to simulate the 14 reactor zones inside the core.
- Each point kinetic model will calculate the neutron power based on 6 different neutron delay groups and 9 photoneutron groups, and the overall change in reactivity.
- The change in reactivity will be a function of control devices (e.g. adjusters), concentration of xenon, voiding in channels, power changes, reactor zones reactivity coupling effects and safety shutdown devices. ( See Fig 18, and Fig. 19).
- The decay heat calculation assumes that 3 separate decay product groups exist, each with a different decay time constant.
- An arithmetic average flux from the 14 reactor zones is used in the decay heat calculation.
- The reactor fuel elements are divided into 4 lumped channels. The total heat generated from the fuel as calculated using the average reactor flux is assumed to be divided equally among the 4 lumped channels.
- Each channel is assumed to have its own coolant flow, and its own lumped fuel element. The temperature of the lumped fuel element in each channel is calculated, and the lumped fuel sheath temperature will be used in the coolant heat transfer calculation.
- In the spatial reactor, the reactivity change includes:

    1. 8 independent lumped adjuster banks, there is an equal reactivity split between 14 zones

2. 2 independent lumped absorber banks, there is an equal reactivity split between 14 zones.

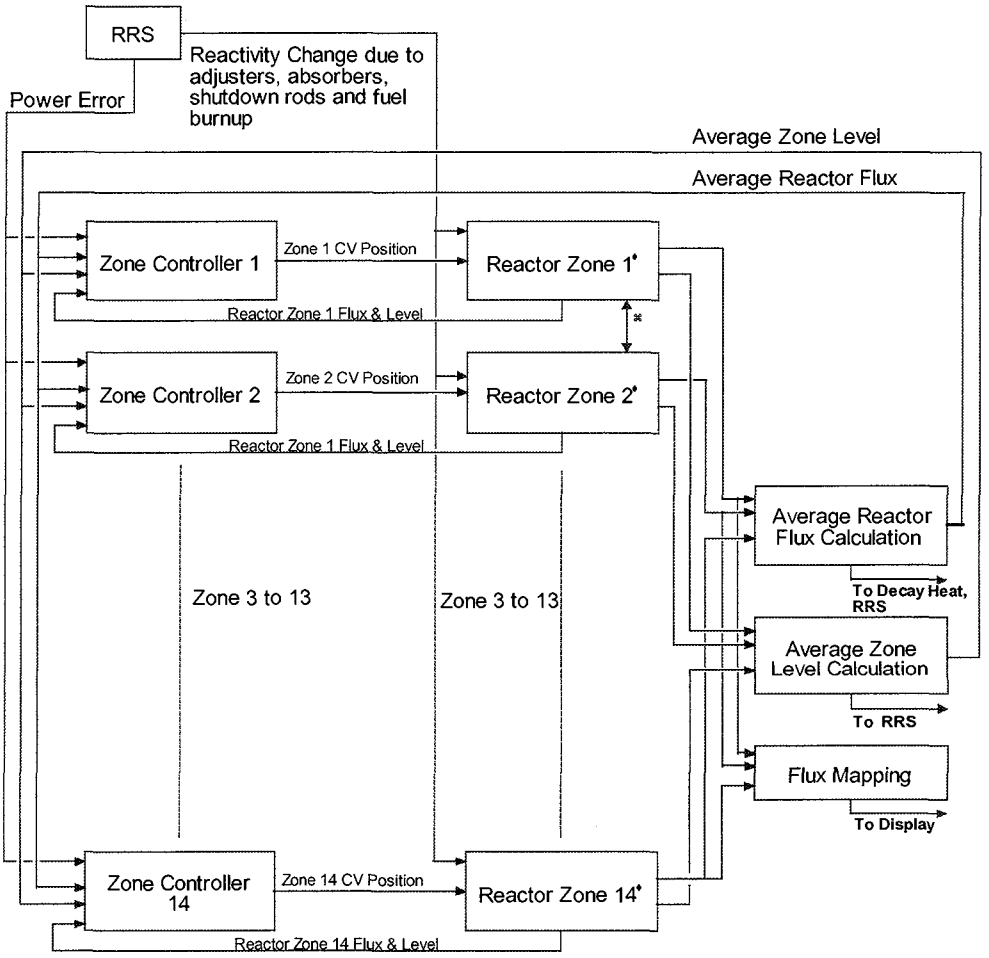
3. liquid zones; the individual zone level will affect the respective reactivity change in its designated zone reactor.

4. reactivity changes due to temperature changes, which is assumed to be proportional to the power changes.

5. reactivity change due to voiding.

6. reactivity due to burnup in fuel.

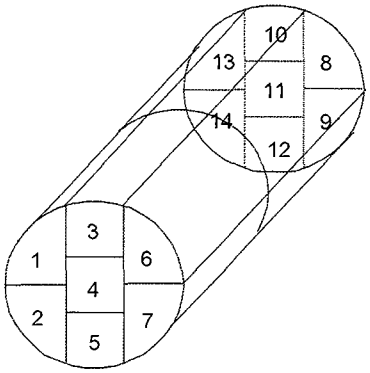
***Fig. 18 Spatial Kinetic Reactor Model***




* **reactivity changes due to temperature change, xenon poisoning and voiding are within each reactor zone**

* **coupling is modelled between each neighbouring zones according to prescribed formula**

***Fig.19 Reactor Model Zones Configuration***




(A) For each Zone Reactor, the point kinetic equations and the governing equations for various reactivity feedbacks are:

1. The total delayed neutron fraction is the summation of the neutron fraction of the 6 neutron groups, 9 photoneutron groups

$$\beta_1 = \sum_{i=1}^{15} \beta_i \quad \text{.....(9.3-1)}$$

$\beta_1$ = total delayed neutron fraction

$\beta_i$ = group i delayed neutron fraction (i=1,2,3,4,5,6);

for i = 7, ...15, $\beta_i$ is the photoneutron delayed group faction.

2. The delayed neutron groups concentration can be expressed as:

$$\frac{dC_i}{dt} = \frac{\beta_i * N_{FLUX}}{T_{NEUTRON}} - \lambda_i C_i \qquad i = 1,....15 \quad \text{.....(9.3-2)}$$

$C_i$ = concentration of 6 delayed neutron groups,

and 9 photoneutron groups.

$\lambda_i$ = decay constant of delayed neutron groups.

$\beta_i$ = group i delayed neutron fraction (i=1,2,3,4,5,6)

( i = 7, ....15 for photoneutrons delayed fraction)

$N_{FLUX}$ = total neutron flux in zone(norm)

$T_{NEUTRON}$ = mean neutron lifetime(sec)

3. The rate of change of neutron flux in a point kinetic model can be expressed as:

$$\frac{dN_{FLUX}}{dt} = \frac{(\Delta K - \beta_1) * N_{FLUX}}{T_{NEUTRON}} + \Sigma_{i=1}^{15} \lambda_i C_i \quad \text{.....(9.3-3)}$$

$N_{FLUX}$ = total neutron flux in zone(norm)

$T_{NEUTRON}$ = mean neutron lifetime(sec)

$C_i$ = concentration of delayed neutron groups.

$\lambda_i$ = decay constant of delayed neutron groups.

$\beta i$ = group i delayed neutron fraction (i=1,2,3,4,5,6)

i = 7,...15 belongs to photoneutron delay groups fraction

$\Delta K$ = overall neutron reactivity change (K)

4. $N_{FLUX}$ can be calculated by solving above equations, using backward Euler's expansion.

$$N_{FLUX} = \frac{N_{FLUX'} + \Sigma_{i=1}^{15} A_i}{1 - \Delta t * (\frac{\Delta K - \beta_1}{T_{NEUTRON}} + \Sigma_{i=1}^{15} B_i)}$$

*where*:

$$Ai = \frac{\lambda i * Ci * \Delta t}{1 + \lambda i * \Delta t}$$

$$Bi = \frac{\lambda i * \beta i * \Delta t}{(1 + \lambda i * \Delta t) * T_{NEUTRON}} \quad \text{.....(9.3-4)}$$

$N_{FLUX'}$ = total neutron zonal flux from previous iteration(norm)

5. The overall reactivity change is expressed as:

$$\Delta K = \Delta KC + \Delta KP + \Delta KV + \Delta KXE + \Delta KSDS + \Delta KFUEL + \Delta KDIFF$$

.....(9.3-5)

$\Delta K$ = overall neutron reactivity change (K)

$\Delta K_C$ = neutron reactivity change due to control devices (K)

$\Delta K_P$ = overall neutron reactivity change due to power changes (K)

$\Delta K_V$ = overall neutron reactivity change due to channel voiding (K)

$\Delta K_{XE}$ = overall neutron reactivity change due to xenon poisoning (K)

$\Delta K_{SDS}$ = overall neutron reactivity change due to safety shutdown systems (K)

$\Delta K_{FUEL}$ = overall neutron reactivity change due to fuel burnup (K)

$\Delta K_{DIFF}$ = overall neutron reactivity change due to coupling between zones (K)

6. The reactivity change due to control devices consist of reactivity change due to adjusters, absorbers and liquid zone.

$$\Delta K_C = \Delta K_{ADJ} + \Delta K_{CA} + \Delta K_Z \quad \text{.....(9.3-6)}$$

$\Delta K_C$ = neutron reactivity change due to control devices (K)

$\Delta K_{ADJ}$ = neutron reactivity change due to adjuster banks (K)

$\Delta K_{CA}$ = neutron reactivity change due to absorber banks (K)

$\Delta K_Z$ = neutron reactivity change due to liquid zone (K)

7. The reactivity change due to liquid zone is calculated in this algorithm:

$$\Delta KZ = MIN\ (MAX(\Delta KZONE / 2 - \Delta KZONE * LZ), -\Delta KZONE / 2),$$

$\Delta$KZONE /2)) .....(9.3-7)

$\Delta K_Z$ = neutron reactivity change due to liquid zone (K)

$\Delta K_{ZONE}$ = Zone controller reactivity worth from completely drained to 100% full (K)

$L_Z$ = liquid zone level(norm)

8. The reactivity change due to power changes is assumed to be:

$$\Delta K_P = \alpha_{TP} * (1 - Q_R) \quad \text{.....(9.3-8)}$$

$\Delta K_P$ = overall neutron reactivity change due to power changes (K)

$\alpha_{TP}$ = power feedback reactivity at zero power (K)

$Q_R$ = heat transfer to coolant(norm)

9. Assuming there are I coolant channels in the reactor zone, the reactivity change due to channel voiding is assumed to be:

$$\Delta K_V = \Delta K_{VO} * N * \frac{(\Sigma_{i=1}^{I} \beta_{Li})}{I}$$

*where*

$$\beta_{Li} = XE_i - XE_{FP} \qquad \text{.....(9.3-9)}$$

$\Delta K_V$ = overall neutron reactivity change due to channel voiding (K)

$\Delta K_{VO}$ = maximum neutron reactivity change due to channel voiding (K)

N = neutron flux squared weighting factor for channels

$\beta_{Li}$ = effective boiling length in channel i(i=channel 1,...I)

XEi = effective exit quality in channel i(i=channel 1,...I)(norm)

$XE_{FP}$= effective exit quality in channel at full power(norm)

10. The reactivity change due to xenon poisoning is assumed to be:

$$\Delta KXE = 0.001*(27.93 - CXE)/14 \text{ .....(9.3-10)}$$

$\Delta K_{XE}$ = overall neutron reactivity change due to xenon poisoning (K)

$C_{XE}$ = xenon concentration

11. The formation of xenon is assumed to be from the decay of iodine as well as from the initial fission products. The concentration of xenon can be found using the following rate equations.

$$\frac{dC_I}{dt} = 9.445E-3 * N_{FLUX} - 2.8717E-5 * C_I$$

$$\frac{dC_{XE}}{dt} = 9.167E-4 * N_{FLUX} + 2.8717E-5 * C_I - (2.1E-5 + 3.5E-4 * N_{FLUX}) * C_{XE} \qquad \text{.....(9.3-11)}$$

$C_I$ = iodine concentration

$C_{XE}$ = xenon concentration

$N_{FLUX}$ = total neutron flux in zone(norm)

12. The reactivity change due to Zone Coupling effects:

$$\Delta\rho_{ij} = \Lambda_i K_{ij} \left( \alpha_i \frac{N_j}{N_i} + \frac{1}{l_i} \frac{\sum_{m=1}^{6}{}_{ZONEj} \lambda_m C_m}{N_i} \right) \quad \text{.....(9.3-12)}$$

Where

$$K_{ij} = K_{ij}^0 + \sum_{k=1}^{14} P_k \cdot K_{ijk} \quad \text{.....(9.3-13)}$$

$$\Lambda_i = \frac{l_i}{K_{ii}}$$

$$\alpha_i = \frac{(1-\beta)}{l_i}$$

$$\lambda_m^* = \frac{\lambda_m}{l_i}$$

(see previous section 9.2 for definition of symbols).

(B) The Overall Reactor

- Referring to figure 18 above. Reactivity changes ΔK due to temperature change, xenon poison buildup, voiding, are *modeled within each point kinetic reactor model.*

- The reactivity changes due to zone coupling effects are calculated separately for each zone. *The zone coupling effects for a particular zone due to all other zones are then summed up and entered as one of the reactivity change inputs for that reactor zone.*

- The reactivity change due to fuel burnup are calculated for one whole reactor, and split up among the 14 zones according to certain criteria - e.g. zones near the center of the reactor would have more contribution than the outer zones.

- The reactivity change due to 2 banks of absorbers (total reactivity worth - 9 mk) are calculated for one whole reactor, and split up among the 14 zones according to certain criteria - e.g. some weighting factors related to the proximity of the absorber location relative to the zones

- The reactivity change due to 8 banks of adjusters (total reactivity worth +17 mk) are calculated for one whole reactor, and split among all the 14 zones according to certain criteria - e.g. some weighting factors related to the proximity of the adjuster location relative to the zones

- The total neutron power from the 14 zones reactor (each normalized) will be summed up and then divided by 14 to get the normalized overall reactor power. As well, each zonal power will provide into to Flux Mapping routine for display.

- All the zone levels will be summed up and divided by 14 to get the average zone level to RRS.

- The power error inside the reactor is defined as:

$$P_{ERR} = \frac{N_{FLUX} - P_{DEM}}{N_{FLUX}} + K_{RATE} * (\frac{N_{FLUX} - N_{FLUX}'}{\Delta t * N_{FLUX}} - R_D)$$

However, under reactor trip, stepback, or if the shutdown rods are fully inserted, the power error is always set to 0.05.

$P_{ERR}$ = power error(norm)

$N_{FLUX}$ = average corrected neutron flux(norm)

$N_{FLUX}'$ = average corrected neutron flux at the previous iteration (norm)

$P_{DEM}$ = demanded reactor power(norm)

$K_{RATE}$ = power error control gain

$R_D$ = demanded power rate of change(/sec)

- The absorber banks drive logic is as follows:

1. If the absorber banks control is set to auto mode, then the absorber banks will move according to the power error versus zone level plot below.

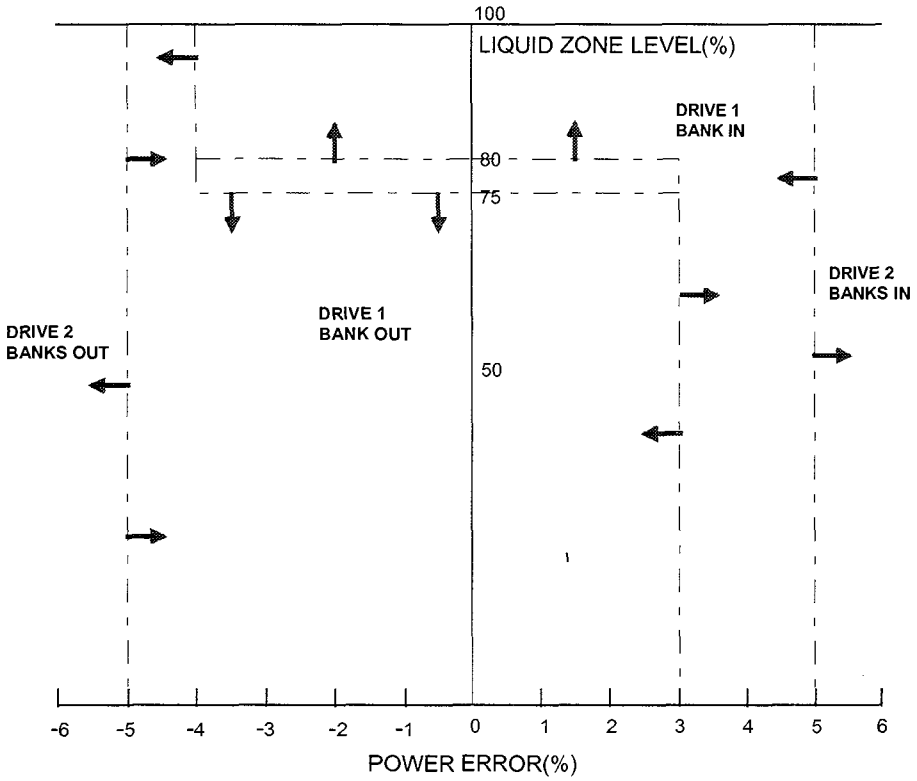
2. In the 'DRIVE 1 BANK IN' region, the absorber bank 1 will be driven in first, and bank 2 will start to drive in when bank 1 is completely driven in.

3. In the 'DRIVE 2 BANKS IN' region, both banks will be driven in simultaneously.

4. Similarly, in the 'DRIVE 1 BANK OUT' region, the absorber bank 2 will be driven out first, and bank 1 will start to drive out when bank 2 is completely driven out.

5. In the 'DRIVE 2 BANKS OUT' region, both banks will be driven out simultaneously.

**ABSORBER BANKS AUTO MODE CONTROL**




- The adjuster rods banks drive logic is as follows:

  1. If the adjuster banks control is set to auto mode, then the adjuster banks will move according to the power error versus zone level plot below.

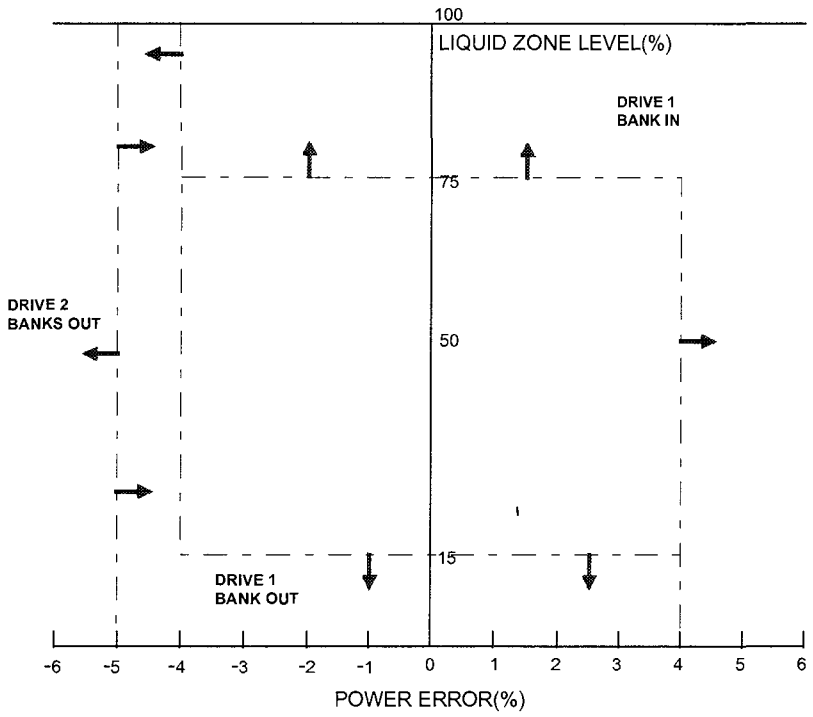
  2. In the 'DRIVE 1 BANK IN' region, the adjuster bank will be driven in an 'ascending bank number' manner, i.e. only one bank will be traveling at a time, and the next bank will start to drive in when the last bank is completely driven in, and the rest of the banks will be driven in this sequential order.

  3. In the 'DRIVE 2 BANKS IN' region, both banks will be driven in simultaneously.

  4. Similarly, in the 'DRIVE 1 BANK OUT' region, one adjuster bank will be driven out in the 'last in first out' manner, and next bank will start to drive out when the traveling bank is completely driven out.

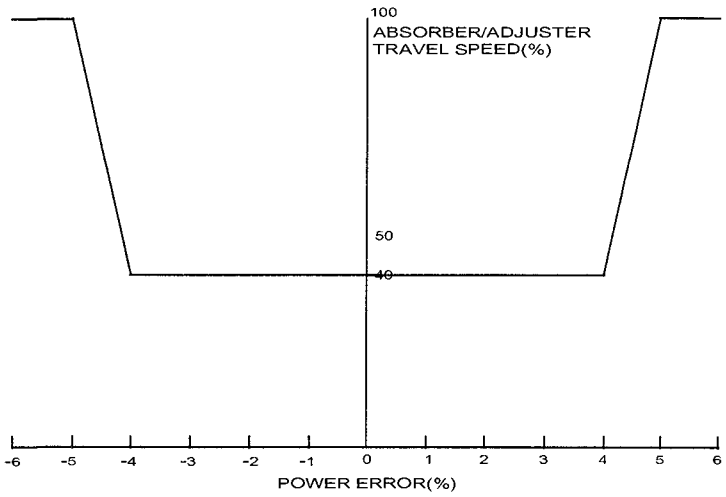
  5. In the 'DRIVE 2 BANKS OUT' region, 2 banks will be driven out simultaneously.

**ADJUSTER BANKS AUTO MODE CONTROL**




- The absorber and adjuster traveling speed is calculated based on power error and is described below. Under reactor stepback conditions, the absorber will take 3.5 seconds to completely dropped into the core.

**ADJUSTER/ABSORBER RODS SPEED PLOT**




- The actual absorber/adjuster position will be calculated from the previous absorber/adjuster position and its speed, if electrical power is available.

- The reactivity change worth of the adjuster and the absorber banks are assumed to be:

$$\Delta K_{CA} = \sum_{i=1}^{2} ABSR_{iWORTH} * (MAX(-1, MIN(0,$$
$$(-0.001267 + 0.13904 * S_{CAi} + 2.32103 * S_{CAi}^{2} + 1.16576 * S_{CAi}^{3})))$$

$$\Delta K_{ADJ} = \sum_{i=1}^{8} ADJ_{WORTH} * (MAX(0, MIN(1,$$
$$(1.02293 - 0.484729 * S_{Ai} - 2.8621 * S_{Ai}^{2} + 2.34474 * S_{Ai}^{3})))$$

$\Delta K_{CA}$ = neutron reactivity change due to absorber banks(K)

$\Delta K_{ADJ}$ = neutron reactivity change due to adjuster banks (K)

$ABSR_{iWORTH}$ = total reactivity change worth of absorber bank i(K)(i=1,2) = total -9 mk for 2 banks

$ADJ_{WORTH}$ = total reactivity change worth of 1 adjuster bank (K) = total +17 mk for 8 banks

$S_{CAi}$ = absorber bank i position (norm)(i=1,2)

$S_{Ai}$ = adjuster bank i position (norm)(i=1,8)

- 2 banks of shutdown rods of equivalent worth are modeled. When reactor trips, the banks of shutdown rods will drop in together. The rods will remain "in" until the shutdown system is "reset". (This occurs after the reactor trip button has been reset). Once the shutdown system is "reset", shutdown rods bank 2 will be pulled out first. After Shutdown rod Bank 2 is completely pulled out, Shutdown rod bank 1 begins to withdraw. Note the rods insertion speed is different than that of the rods 'pull-out' speed.

- The reactivity change due to safety-shutdown rods are assumed to be:

$$\Delta K_{SDS} = \sum_{i=1}^{2} \frac{\Delta K_{SDSMAX}}{2} * (MAX(0, MIN(1, (SDS_{iPOS}))))$$

$\Delta K_{SDS}$ = neutron reactivity change due to shutdown rods (K)

$\Delta K_{SDSMAX}$ = reactivity change due to shutdown rods when fully inserted(K) = total -68 mk for 2 shutdown rods

## 9.4 Coupled Reactor Kinetics Reference


1. Luxat, J.C., " The Potential of a Generalized Modal Analysis of Postulated Loss-of-Coolant Accidents in CANDU Reactors", Nuclear Technology, Vol. 46, Mid-Dec 1979.

2. Avery, R., "Theory of Coupled Reactors", Proc. 2nd Int. Conf. On Peaceful Uses of Atomic Energy, Geneva, September 1958, 8/1958, p.182-191.

3. Chou, Q.B., et al, "Development of a Low-Cost Spatial CANDU Reactor Simulation Program for Power Plant Control Studies", Proceedings of 1975 Summer Computer Simulation Conference, San Francisco, California, July 21-23, 1975.

## Typical Workshop Agenda

**Session 1 -**

**1. Introduction to Various Modular Modeling Systems**

1.1 Integral Components of a Modular Modeling System

1.2 Simulation Development System for Control Room Replica Full Scope Simulator (ROSE, CETRAN)

1.3 Simulation Development System for PC based Full Scope, Part Scope, or Principle Simulator (MMS, APROS, CASSIM)

1.4 Technology Trends for Modular Modeling System

1.5 Introductory Hands-On Exposure to CASSIM™

**2. Dynamic Simulation Fundamental**

2.1 Introduction to Mathematical Modeling and Physical Laws

2.2 Overview of Dynamic Simulation for Power Plant Process

2.3 Numerical Methods & Other Important Model Considerations

2.4 Simple Modeling Example (2 Tanks) with CASSIM™

**3. CASSIM™ Simulation Fundamentals**

3.1 Merging Models with CASSIM™

3.2 Thermalhydraulic Modeling with CASSIM™ Hydraulic Flow Network (HFN)

3.3 Creating a simple CASSIM Algorithm

**4. Reactor Model Development**

3.4 Quick Overview of Nuclear Reactor Kinetics

3.5 Simple Point Kinetic Reactor Model - Basic Equations & Analytical Solution

3.6 Development of the Point Kinetic Reactor Algorithm

**Session 2 -**

**1. Wrap Up Point Kinetic Reactor Model Development**

**2. Effects of Neutron Sources on Reactor Kinetics**

2.1 Effects of neutron sources on Reactor at Power and Shutdown

2.2 Photoneutron Sources in Heavy Water Reactors

2.3 Modify Point Kinetic Reactor Algorithm with Photoneutron Sources

2.4 Discussion

**3. Reactivity Feedback Effects**

3.1 Overview of Reactivity Feedback Effects: Fuel Burnup and Buildup; Fission Products Poisoning; Coolant Void Reactivity Feedback; Temperature Feedback Effects

3.2 Xenon/Iodine Poison - Basic Equations and Dynamics

3.3 Developing Xenon/Iodine Model

3.4 Discussion

**Session 3 -**

**1. Overview of Reactor Control for Different Reactors**

1.1 Brief Overview of BWR, PWR, RBMK Reactor Controls

1.2 Overview of PHWR Reactivity Control Mechanisms & Reactor Regulating System (RRS)

**2. PHW Reactor Regulating System (RRS) Familiarization Using Simulator**

2.1 Familiarization with Generic CANDU Simulator

2.2 Simulator Exercises related to PHW Reactor Control

**3. Simulator Exercises continued for PHW Reactor Control**

**4. Reactivity Control Mechanism Modeling - Liquid Zone**

4.1 Liquid Zone Hydraulic Modeling

4.2 Liquid Zone Reactivity Control Modeling

**Session 4 -**

**1. Reactivity Control Mechanism Modeling Continued**

1.1 Develop Adjuster Rods Model

1.2 Develop Absorber Rods Model

**2. Reactor Shutdown System Modeling**

2.1 Develop Reactor Shutdown Rods Model

**3. Model Integration for Simple Reactor**

3.1 Integrate Reactor Model + Xenon-Iodine Poison Model +Adjuster & Absorber & Shutdown Rods Model + Liquid Zone Model

3.2 Testing & Discussion

**Session 5 -**

**1. Practice Reactor Startup and Approach to Criticality using PHWR Simulator**

2.1 Subcritical Reactor Behavior

2.2 Reactor Instrumentation

2.3 Methods for Approaching Criticality

2.4 Practice Reactor Startup and Approach to Criticality Using Simulator

**2. From Simple Model to Higher Fidelity Nuclear Reactor Models**

2.1 PHWR Model with Spatial Kinetics & Demonstrations

2.2 Nodal Approach for Simulating Space-Time Reactor Kinetics

2.3 Summary of Model Development for PHW Reactor

2.4 Model Development Roadmap & Discussion

APPENDIX

Selected CASSIM™ Algorithms Descriptions

**Algorithm # 102: PUMP_SIM, Simple Centrifugal Pump**

***Purpose & Descriptions:***

Note: the pump manufacturer normally provides pump performance diagrams for users -
(a) Pump power (at certain specific gravity and speed) versus mass flow rate.
(b) Total dynamic head (at certain specific gravity and speed) versus mass flow rate and
Net Positive Suction Head (N.P.S.H.) required.

The pump model is a simple centrifugal pump dynamic model which utilizes the pump's manufacturer performance data for the following computation:

1. The pump static head is calculated as proportional to the product of the **design static head** and the **square of normalized pump speed**. Other factors such as pump degradation and fluid density changes relative to the design density are taken into account in the calculation. That means as the pump is starting up, the pump static head will be increasing and will be on a quadratic rise to the design static head as the speed increases.
2. The pump power is computed as quadratic function of pump mass flow, using curve-fit technique on the pump performance curve to obtain respective constants (KP1, KP2, KP3) for the quadratic function.
3. Heat dissipated by the pump frictional loss is calculated as a fraction of pump power
4. Cavitation factor is calculated as (suction pressure - saturation pressure) / N.P.S.H.

Note that the CASSIM hydraulic network solution scheme is represented by a nodal diagram which involves identifying control volumes (known as nodes) within the simulated hydraulic flow circuit. The flow between these nodes, called link flow, can be obtained from the linearized momentum equation governing the pressure differentials across these nodes. Each node pressure can be obtained from mass balance around the nodes. In the present network solution scheme, the pump static head is assigned to the link in which the momentum equation is expressed as :
link mass flow is proportional to the square root of {pressure at suction node of the pump + the static head of the pump - discharge node pressure}. The pump dynamic head is calculated in the link block in which the pump is located. For details on additional hydraulic flow network solution techniques, refer to CTI's document "CASSIM™ Thermohydraulic Flow Network Solutions - Techniques & Examples"

***Scope of Application:***

**Physical effects simulated:**

- Static head calculation as function of speed, fluid density changes and pump efficiency.
- Pump power as a function of mass flow.

- Heat dissipation as pump frictional loss.
- Cavitation factor is calculated.

**Limitations or effects not simulated:**
- Pump rundown and flywheel effects are not simulated.

**Validity check:**
- Cavitation factor should be checked if it is near or equal to 1.

**Range of Operation:**
- Full range of operation with the exception of pump cavitation.

**Special Requirements & Assumptions:**
- None.

**Input Specifications (ALG102):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Pump motor speed | A | Normalized | Connect to the speed output tag of the motor block for the pump. |
| 2. Pump mass flow | A | Kg/sec | Connect to the link flow output tag of the link block in which the pump is located |
| 3. Not used | | | |
| 4. Fluid density | A | $Kg/m^3$ | This can be inputted from a table look-up function or just a constant tagname ($Constant) if it is a constant |
| 5. Pump suction Pressure | A | KPa | Connect to pressure output tag of the suction pressure node block |
| 6. Fluid saturation pressure | A | KPa | This can be inputted from a table look-up function or just a constant tagname ($Constant) if it is a constant |
| 7. Pump degradation factor | A | Normalized | 0 - no degradation; 1 - 100 % degraded. This can be inputted from a table look-up function or just a constant tagname ($Constant) if it is a constant. Usually it is obtained from the pump efficiency specification. |

**Output Specifications (ALG102):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Pump static head (G1) | A | KPa | Connect to pump static head input tag of the link block in which the pump is located. |
| 2. Cavitation factor (G2) | A | Normalized | For indication |
| 3. Pump power (PM1) | A | KW | For indication and electrical motor current calculation |
| 4. Heat loss (Q1) | A | KW | For calculation of pump heat to the pumping fluid. |

**Coefficient Specifications (ALG102):**

| Coefficient Descriptions | Analog (A)/ Digital (D) | Engineering Units | Calculation Details |
|---|---|---|---|
| 1. Design static head | A | KPa | This is normally obtained from pump specification |
| 2. Fluid Design Density | A | $Kg/M^3$ | This is normally obtained from pump specification |
| 3. Required NPSH | A | KPa | This is normally obtained from pump specification |
| 4. Pump power factor KP1 | A | KW | This is obtained from curve-fit to pump performance curve |
| 5. Pump power factor KP2 | A | KW/(Kg/sec) | This is obtained from curve-fit to pump performance curve |
| 6. Pump power factor KP3 | A | KW/(Kg/sec) 2 | This is obtained from curve-fit to pump performance curve |

**Algorithm # 106: VLV_ONOFF_SM, Simple ON-OFF Valve**

***Purpose & Descriptions:***

1. This simple valve model allows user to specify the type of valve to be used: linear; equal percentage; quick opening.
2. The model checks for motive force for the valve. Usually for pneumatic control valve, the motive force is instrument air with a constant pressure as specified by the valve manufacturer. On total loss of motive force, the valve is either failed open, or failed closed. On partial loss of motive force, the valve will fail in mid-position on a graduated scale between fully open to fully closed position, being proportional to the value of the degraded motive force signal at the time.
3. The valve model accepts the solenoid valve status and drives the valve stem position to the fully opened position when the solenoid is energized and fully closed when the solenoid is de-energized. The stem position will travel in a first order lag fashion. The time constant for the lag is a function of the stroking time constant for the valve.
4. The stroking time for the valve is defined as the time required for the valve stem to travel from the fully closed position to the stem position at maximum $C_V$ on a step-change in control signal from fully closed requested to fully opened requested.
5. The normalized valve port area is calculated based on the type of valve specified by the user. For linear valve, the normalized port area is equal to the stem position; for equal percentage valve, the normalized port area is a quadratic function of valve stem position; for quick opening valve, the port area is proportional the cube of stem position.

***Scope of Application:***

**Physical effects simulated:**

- Motive force (e.g. instrument air signal) for the valve.
- 3 types of valve can be specified: linear, equal %, quick opening
- Valve can fail open, closed, or failed to desired position.

**Limitations or effects not simulated:**

- Valve cavitation is not calculated in the model
- If the valve characteristics is known, and it does not fit the linear, equal % or the quick opening, an extra block(ALG152-CURVEFIT) can be created to calculate the effective port-area, using the stem position output(OUT(2)) from this block as an input to the CURVEFIT block.

**Validity check:**

- Valve will not operate if motive force is not provided.

**Range of Operation:**

- Full range of valve operation, provided motive force is kept constant and is greater than the motive force required to fully open and close the valve.

**Special Requirements & Assumptions:**

- None

**Input Specifications(ALG106):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Valve solenoid signal | D | nil | Connect to the solenoid valve status. A datavalue of 1= fully open, and 0= fully closed |
| 2. Valve Malfunction - **"Fail to position - On"**: | D | nil | Connect to Non-Moveable block which gets inputs from graphical user interface for inserting malfunction. 1 means malfunction active; 0 means malfunction inactive |
| 3. Valve Malfunction - **"Fail to position-magnitude "**: | A | Normalized | Connect to Non-Moveable block which gets inputs from graphical user interface for inserting malfunction. The datavalue of this input represents the stem position the valve is failed to.(e.g.:1=open, 0 =close, 0.5=half open) |
| 4. Motive Force | A | Depends on motive force medium : Pneumatic (KPa); electronic (mA, or mV) | This can be connected to instrument air source, or power supply source, depending on the motive force medium. If these sources are not modelled, just enter a constant with appropriate value ($Constant). For the valve to stroke the full range, the datavalue of this input must be greater than the data-value of COF(3)(motive force required to fully open) and the data-value of COF(4)(motive force required to fully closed) |

**Output Specifications(ALG106):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Valve port area (ZA1) | A | Normalized | Connect to the valve area input tag of the link block in which the valve is located. |
| 2. Valve stem position (ZS1) | A | Normalized | For indication |
| 3. Valve fully opened (FO1) | D | Nil | Connect to logic blocks for control or indication. This fully opened limit switch is set at stem position > 98% |
| 4. Valve fully closed (FC1) | D | Nil | Connect to logic blocks for control or indication. This fully closed limit switch is set at stem position < 2% |
| 5. Valve stem position in % (AS1) | A | % | Connect to logic blocks for control or indication |

**Coefficient Specifications(ALG106):**

| Coefficient Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Valve stroking time | A | sec | This is normally obtained from the valve specification. This stroking time is defined as the time required to open the valve from fully closed to a stem position at maximum $C_V$ |
| 2. Type of Valve | A | Nil | 1 = Linear; 2= Equal %; 3= Quick Open |
| 3. Motive force required to fully open | A | Depends on motive force medium : Pneumatic (KPa); electronic (mA, or mV) | This is normally obtained from the valve specification |
| 4. Motive force required to fully close | A | Depends on motive force medium : Pneumatic (KPa); electronic (mA, or mV) | This is normally obtained from the valve specification |
| 5.Stem position at maximum $C_V$ | A | Normalized | This is normally obtained from the valve specification. A typical value of 90% can be assumed. |

**Algorithm # 108: CV_SIM, Simple Control Valve**

***Purpose & Descriptions:***

1. This simple valve model allows user to specify the type of valve to be used: linear; equal percentage; quick opening.
2. The model checks for motive force for the valve. Usually for pneumatic control valve, the motive force is instrument air with a constant pressure as specified by the valve manufacturer. On total loss of motive force, the valve is either failed open, or failed closed. On partial loss of motive force, the valve will fail in mid-position on a graduated scale between fully open to fully closed position, being proportional to the value of the degraded motive force signal at the time.
3. The valve model accepts control signal (normalized) from controller and drives the valve stem position to follow the control signal command in a first order lag fashion. The time constant for the lag is a function of the stroking time constant for the valve.
4. The stroking time for the valve is defined as the time required for the valve stem to travel from the fully closed position to the stem position at maximum $C_V$ on a step-change in control signal from fully closed requested to fully opened requested.
5. The normalized valve port area is calculated based on the type of valve specified by the user. For linear valve, the normalized port area is equal to the stem position; for equal percentage valve, the normalized port area is a quadratic function of valve stem position; for quick opening valve, the port area is proportional the cube of stem position.

***Scope of Application:***

**Physical effects simulated:**

- Motive force (e.g. instrument air signal) for the valve.
- 3 types of valve can be specified: linear, equal %, quick opening
- Valve can fail open, closed, or failed to desired position.

**Limitations or effects not simulated:**

- Valve cavitation is not calculated in the model
- If the valve characteristics is known, and it does not fit the linear, equal % or the quick opening, an extra block(ALG152-CURVEFIT) can be created to calculate the effective port-area, using the stem position output(OUT(2)) from this block as an input to the CURVEFIT block.

**Validity check:**

- Valve will not operate if motive force is not provided.

**Range of Operation:**

- Full range of valve operation, provided motive force is kept constant and is greater than the motive force required to fully open and close the valve.

**Special Requirements & Assumptions:**

- None

**Input Specifications(ALG108):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Control Signal | A | Normalized | Connect to control signal output tag of PID controller block |
| 2. Valve Malfunction - **"Fail to position - On"**: | D | nil | Connect to Non-Moveable block which gets inputs from graphical user interface for inserting malfunction. 1 means malfunction active; 0 means malfunction inactive |
| 3. Valve Malfunction - **"Fail to position-magnitude "**: | A | Normalized | Connect to Non-Moveable block which gets inputs from graphical user interface for inserting malfunction. The datavalue of this input represents the stem position the valve is failed to.(e.g.:1=open, 0 =close, 0.5=half open) |
| 4. Motive Force | A | Depends on motive force medium: Pneumatic (KPa); electronic (mA, or mV) | This can be connected to instrument air source, or power supply source, depending on the motive force medium. If these sources are not modelled, just enter a constant with appropriate value ($Constant). For the valve to stroke the full range, the datavalue of this input must be greater than the data-value of COF(3)(motive force required to fully open) and the data-value of COF(4)(motive force required to fully closed) |

**Output Specifications(ALG108):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Valve port area (ZA1) | A | Normalized | Connect to the valve area input tag of the link block in which the valve is located. |
| 2. Valve stem position (ZS1) | A | Normalized | For indication |
| 3. Valve fully opened (FO1) | D | Nil | Connect to logic blocks for control or indication. This fully opened limit switch is set at stem position > 98% |
| 4. Valve fully closed (FC1) | D | Nil | Connect to logic blocks for control or indication. This fully closed limit switch is set at stem position < 2% |
| 5. Valve stem position in % (AS1) | A | % | Connect to logic blocks for control or indication |

**Coefficient Specifications(ALG108):**

| Coefficient Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Valve stroking time | A | sec | This is normally obtained from the valve specification. This stroking time is defined as the time required to open the valve from fully closed to a stem position at maximum $C_V$ |
| 2. Type of Valve | A | Nil | 1 = Linear; 2= Equal %; 3= Quick Open |
| 3. Motive force required to fully open | A | Depends on motive force medium : Pneumatic (KPa); electronic (mA, or mV) | This is normally obtained from the valve specification |
| 4. Motive force required to fully close | A | Depends on motive force medium : Pneumatic (KPa); electronic (mA, or mV) | This is normally obtained from the valve specification |
| 5.Stem position at maximum $C_V$ | A | Normalized | This is normally obtained from the valve specification. A typical value of 90% can be assumed. |

**Algorithm #128: TANK, Simple Storage Tank with Heating**

***Purpose & Descriptions:***

1. An overall mass balance is performed by this algorithm to calculate the mass accumulation in the tank based on the inlet and outlet flowrate. The level in the tank can also be calculated based on the mass accumulation in the tank, the fluid density, and the tank's orientation.

2. An energy balance is also used to calculate the temperature of the fluid in the tank. The temperature is calculated assuming a constant heat capacity. Enthalpy can also be used in place of temperature, as long as the units for INP(3), and OUT(3) are consistent. However, if OUT(3) is chosen to represent enthalpy, COF(6) the Cp of the fluid must be set to 1.

3. If the level in the tank is greater than the overflow height, overflow of the liquid will result. A simple first order lag is used to model the overflow.

4. When the level in the tank falls below the elevation at which the outlet pipe is connected to the tank(KEMPTY), the outlet flow should decrease. The outlet flow is assumed to be decreased to zero, when the level falls to 20% of KEMPTY. The tank level factor is used as a measure to reduce the outlet flow when the level in the tank falls below KEMPTY.

***Scope of Application:***

**Physical effects simulated:**

- Overall mass balance to calculate the liquid level inside the tank
- Overall energy balance to calculate the liquid temperature inside the tank
- Liquid overflow will occur, when the liquid level inside the tank exceeds the overflow limit
- Tank level factor reflecting the required reduction in outflow when the tank liquid level falls below the elevation at which the outlet pipe is connected to the tank.

**Limitations or effects not simulated:**

- Only fluid in single phase can be considered.
- Reverse flows at the input of the tank is not allowed.
- No pressure calculation inside the tank. The pressure is assumed to be constant.
- The density of liquid in the tank is assumed to be constant

**Validity check:**

- If reverse flows are simulated, additional blocks must be used to calculate overall input and output flowrates with respect to the tank.
- Check for two phase fluid.

**Range of Operation:**

**Special Requirements & Assumptions:**

- All fluids are in single phase, no boiling or condensation takes place in the tank.
- The properties $C_p$ of both fluids are assumed constant within the range of their temperature range.
- Only 1 overall inlet flow and 1 overall outlet flow is modelled. If more than one inlet flow or outlet flow exists, additional blocks will be required to calculate the overall inlet and outlet flows.
- The energy balance for the tank assumed perfect mixing for the tank.

**Input Specifications(ALG128):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Inlet flowrate | A | Kg/sec | Connect to the link flow, output tag of the link block in which the fluid stream 1 is located. Or if more than one inlet flow exists, use a mixing node block(ALG151) to calculate the combined inlet flowrate. |
| 2. Outlet flowrate | A | Kg/sec | Connect to the link flow, output tag of the link block in which the fluid stream 2 is located. Or if more than one inlet flow exists, use a summer block(ALG312) to calculate the combined outlet flowrate. |
| 3. Inlet temperature | A | $^oC$ | Connect to the block which calculate the temperature of the inlet stream. Or if more than one inlet flow exists, use a mixing node block(ALG151) to calculate the combined inlet temperature. If enthalpy is used instead of temperature, COF(6) must be set to one. |
| 4. Heat input | A | KJ/S | Connect to the block which calculate the external heat into the tank. Any heat loss should be combined to this value. |

**Output Specifications(ALG128):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Tank Level(L1) | A | M | Connect to equipment requiring the tank level. This output is required to be initialized with the design data at the full power steady state point. |
| 2. Mass in Tank(M1) | A | Kg | This output is required to be initialized with the design data at the full power steady state point. |
| 3. Temperature in Tank | A | $^oC$ | Connect to equipment or heat balances requiring the temperature of the liquid in the tank or leaving the tank. This output is required to be initialized with the design data at the full power steady state point. |
| 4. Tank Overflow(FM1) | A | Kg/sec | This can be connected to the hydraulic flow network as an external flow. |
| 5. Tank level Factor(G1) | A | Normalized | Connect to the hydraulic flow link which calculate the outlet flow from the tank. If the tank level is greater than level at which outlet flow begins to reduce, the value of this output should be one. |

**Coefficient Specifications(ALG128):**

| Coefficient Descriptions | Analog (A)/ Digital (D) | Engineering Units | Calculation Details |
|---|---|---|---|
| 1. Orientation flag | D | Nil | 0 = horizontal; 1 = vertical |
| 2. Tank radius | A | M | This value should be obtained from the equipment manufacturer |
| 3. Tank height/length | A | M | This value should be obtained from the equipment manufacturer |
| 4. Overflow height | A | M | This value should be obtained from the equipment manufacturer |
| 5 Maximum outflow | A | Kg/sec | Can be estimated from the normal outlet flows by multiplying that value by 130% |
| 6: Cp of liquid | A | KJ/Kg/°C | For water Cp = 4.16 KJ/Kg/°C |
| 7: Density of liquid | A | $Kg/M^3$ | For water at 40 °C density = 992.3 $Kg/M^3$ |
| 8: Overflow lag | A | sec | A default value of 2 seconds can be assumed |
| 9: level at which outlet flow begins to reduce | A | M | This value should be obtained from the equipment manufacturer |

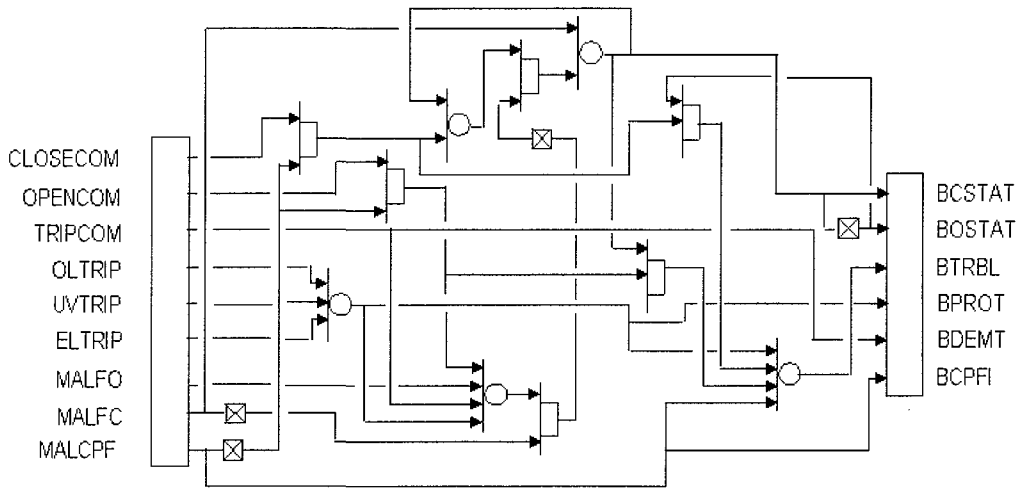
**Algorithm # 133: BREAKER, Breaker**

***Purpose & Descriptions:***

1. This algorithm simulates a typical breaker, with breaker close and breaker open status as outputs. Breaker alarms indications are also modelled.

2. The inputs for the breaker algorithm includes close and open command, trip command, trip relays, and malfunctions.

3. The operation of the breaker can be summarized by the following logic diagram.




| | |
|---|---|
| CLOSECOM | = command to close breaker |
| OPENCOM | = command to open breaker |
| TRIPCOM | = command to trip breaker |
| OLTRIP | = overcurrent trip relay 51 |
| UVTRIP | = undervoltage trip relay 27 |
| ELTRIP | = ground protection trip relay 64 |
| MALFO | = failed open malfunction |
| MALFC | = failed close malfunction |
| MALCPF | = control power failure relay 8 |
| BCSTAT | = breaker closed status |
| BOSTAT | = breaker open status |
| BTRBL | = breaker trouble alarm |
| BPROT | = breaker protection trip operated |
| BDEMT | = breaker demanded to trip |
| BCPFI | = breaker control power failure indication |

***Scope of Application:***

**Physical effects simulated:**

- Breaker status
- Breaker alarm indications

**Limitations or effects not simulated:**

**Validity check:**

**Range of Operation:**
**Special Requirements & Assumptions:**

**Input Specifications (ALG133):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Command to close | D | Nil | Connect to the block where the command to close the breaker is calculated. Typical a control macro block is used, and the On/start command should be connected to this input. |
| 2. Command to open | D | Nil | Connect to the block where the command to open the breaker is calculated. Typical a control macro block is used, and the Off/stop command should be connected to this input. |
| 3. Command to trip | D | Nil | Connect to the block where the command to trip the breaker is calculated. Typical a control macro block is used, and the trip open command should be connected to this input. |
| 4. Overcurrent trip | D | Nil | Connect to the block where the overcurrent trip is calculated |
| 5. Undervoltage trip | D | Nil | Connect to the block where the undervoltage trip is calculated |
| 6. Ground protection trip | D | Nil | Connect to the block where the ground protection trip is calculated |
| 7. Failed open malfunction | D | Nil | Connect to the non-movable block which modelled this malfunction |
| 8. Failed closed malfunction | D | Nil | Connect to the non-movable block which modelled this malfunction |
| 9. Control power failure | D | Nil | Connect to the non-movable block which modelled this malfunction |

**Output Specifications (ALG133):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Breaker close status (BC1) | D | Nil | This output should be initialized. This breaker output can be connected to motor blocks as a command to start. |
| 2. Breaker open status (BO1) | D | Nil | This output is for indication only. This should always be the inverse of OUT(1) |
| 3. Breaker trouble alarm (D1) | D | Nil | |
| 4. Breaker protection trip operated(D2) | D | Nil | |
| 5. Breaker demanded to trip(D3) | | Nil | |
| 6. Control power failure indication(D4) | D | Nil | If the electrical circuit is modelled, then this output should be connected to the electrical circuit blocks. |

**Coefficient Specifications (ALG133):** None

**Algorithm # 134: MOTOR, Simple motor model**

***Purpose & Descriptions:***

1. This algorithm calculates the electrical current and the normalized speed of the motor.

2. A constant acceleration and deceleration are assumed. The user must specify the runup time, as the time required for the motor to reach a normalized speed of 1 from zero, on a start command, and the shutdown time, as the time required , for the motor to reach a normalized speed of 0 from 1 on a stop command. The normalized speed is calculated based on the constant acceleration and deceleration term.

3. The total current can be calculated based on the motor start-up current and the motor running current.

4. The motor running current is calculated based on the mechanical power used by the equipment, e.g. the pump/fan power, the power factor, the voltage of the bus, and the motor efficiency.

5. The motor start-up current is calculated based on the speed of the motor, and the initial start-up current when the speed is zero.

   $STARTI = MSTARTI*(1 - NSPEED^{EXPF})$

   STARTI = Start-up current in amps
   MSTARTI = Start-up current when motor speed is zero
   NSPEED = normalized motor speed
   EXPF = Exponent of Speed

***Scope of Application:***

**Physical effects simulated:**

- Motor speed
- Current used by the motor

**Limitations or effects not simulated:**

- Braking effects on motor

**Validity check:**

**Range of Operation:**

**Special Requirements & Assumptions:**

- Constant acceleration and de-celeration are assumed

**Input Specifications(ALG134):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Start/stop command | A | Nil | Connect to the block where the command to start/stop the motor is calculated. Typical a breaker block is used, and the breaker close status is used to drive this input. |
| 2. Pump/fan power | A | KJ/sec | Connect to the block where the pump/fan power is calculated. For centrifugal pumps, the pump power is calculated in the pump dynamics blocks(ALG102, PUMP_SIM) |
| 3. Bus voltage | A | KV | Connect to the block where the bus voltage is calculated |

**Output Specifications(ALG134):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Motor Speed | A | Normalized | This value should be connected to a pump dynamics block(ALG102 PUMP_SIM) as an indication of motor speed. It can be connected to any block which required normalized motor speed. |
| 2. Acceleration | A | norm/sec | |
| 3. Start-up current | A | amperes | |
| 4. Running current | A | amperes | |
| 5. Total current | A | amperes | If the electrical circuit is modelled, then this output should be connected to the electrical circuit blocks. |

**Coefficient Specifications(ALG134):**

| Coefficient Descriptions | Analog (A)/ Digital (D) | Engineering Units | Calculation Details |
|---|---|---|---|
| 1. Run-up time to full speed | A | sec | This value should be obtained from the equipment manufacturer. |
| 2. Run-dwon time to zero speed | A | sec | This value should be obtained from the equipment manufacturer. |
| 3. Current calculation flag | A | Nil | If this coefficient is set to zero, then the current calculation is by-passed. |
| 4. Power conversion factor | A | | This value should be set to 1 if the bus voltage and the pump power are in the units KV, and KW. |
| 5. Reference power factor | A | Nil | A typical value of 0.85 can be assumed |
| 6. Motor efficiency | A | Normalized | This value should be obtained from the equipment manufacturer. A typical value of 0.9 can be assumed |
| 7. Motor start-up current | A | amperes | This value should be obtained from the equipment manufacturer. A typical value of 5*the running current at design conditions can be assumed. |
| 8. Exponent of speed in current calculation | A | Nil | A typical value of 0.25 can be assumed |

**Algorithm # 140: CONDUCT_FI, incompressible flow conductance**

***Purpose & Descriptions:***

1. This algorithm predict the flow in a link using incompressible flow momentum equation.

   $W = K*VAL*(P_{up}-P_{down})^{0.5}$

2. Check valve is modelled for this link.

***Scope of Application:***

**Physical effects simulated:**

- The flow through the link as predicted by incompressible flow equation

**Limitations or effects not simulated:**

- Static head from pump or elevation difference are not considered. The user has to modelled static heads in a separate block and account for them in the upstream pressure input term.
- 2-phase flow effects are not modelled.

**Validity check:**

**Range of Operation:**

**Special Requirements & Assumptions:**

- If more than 1 valve exists on the same link, the effective valve port-area can be found by assuming the overall effective port-area = the multiplication of all the port-areas on the same link.

**Input Specifications(ALG140):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Inlet Pressure | A | KPaa | Connect to node pressure output tag of the node block in which the inlet pressure is located. |
| 2. Outlet Pressure | A | KPaa | Connect to node pressure output tag of the node block in which the outlet pressure is located. |
| 3. Valve position | A | Normalized | Connect to the block where the overall valve port-area is calculated. |

**Output Specifications(ALG140):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Total outflow(FM1) | A | Kg/sec | May be connected to the hydraulic network or tank block to preserve mass balance or use as indication. |

**Coefficient Specifications(ALG140):**

| Coefficient Descriptions | Analog (A)/ Digital (D) | Engineering Units | Calculation Details |
|---|---|---|---|
| 1. Rated flow conductance | A | $Kg/KPa^{0.5}$ | This value is calculated by dividing the mass flow by the square root of the pressure difference of the upstream and downstream node at design conditions. |
| 2. Check valve status | D | Nil | Enter 1 if check valve is present else enter 0 |

**Algorithm #223: RSFF, Reset — Set (RS) Flip Flop**

***Purpose & Descriptions:***

This algorithm simulates a logical R-S Flip flop element:

- Input #1 designates the "Set" signal; whereas " Input #2 designates the "Reset" signal; Coefficient #1 dictates whether "Set" Overrides "Reset" ( if value of the coefficient #1 = 1), or "Reset" overrides "Set" ( if value of the coefficient #1 = 0).
- Assuming that Coefficient #1 = 0, (i.e. Reset overrides Set), when the Set signal is equal to 1 (logical true) **and** Reset signal is equal to 0 (logical false), then Output 1 is equal to 1 and "latched in". Output 2 is the inversion of Output 1, therefore equal to 0.
- In the event that Reset signal is equal ! (logical true) when the Set signal is equal 1, Output 1 is always equal to 0 (logical false); Output 2 is always equal to 1 (logical true), due to the Reset override.
- The output 1 latched-in state will be reset to 0 (logical false), as soon as the Reset signal becomes 1 (logical true).
- The truth table representation for the R-S Flip Flop is as follows:
  Note that Output 1 & 2 in each row represent logical states corresponding to different logic states of inputs and coefficients.

| Set Input State | Reset Input State | Coefficient State | Output 1 State | Output 2 State |
|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 |

The truth table below represents a typical sequence of events for the output logic states evolution corresponding to Set Signal & Reset Signal with Coefficient = 0

| Event Sequence | Set Input State | Reset Input State | Coefficient State | Output 1 State | Output 2 State |
|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 0 |
| 2 | 0 | 0 | 0 | 1 | 0 |
| 3 | 0 | 1 | 0 | 0 | 1 |
| 4 | 0 | 0 | 0 | 0 | 1 |

***Scope of Application:***

**Physical effects simulated:** Not applicable.

**Limitations or effects not simulated:** Not applicable.

**Validity check:** Not applicable.

**Range of Operation:**

- Input values are either 0 (logical false) or 1 (logical true). No intermediate values between 0 and 1 are allowed.

**Special Requirements:**

- None.

**Input Specifications (ALG #223):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Set Input | D | Nil | |
| 2. Reset Input | D | Nil | |

**Output Specifications(ALG #223):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Output 1 | D | Nil | |
| 2. Output 2 | D | Nil | |

**Coefficient Specifications (ALG #223):**

| Coefficient Descriptions | Analog (A)/ Digital (D) | Engineering Units | Calculation Details |
|---|---|---|---|
| 1. Coefficient 1 | D | Nil | 0 = Reset overrides Set; 1 = Set overrides Reset |

**Algorithm #232: Comparator, Single Analog Comparator with Deadband**

***Purpose & Descriptions:***

This algorithm simulates a single analog comparator (e.g. level switch, temperature switch, pressure switch, etc.) with the following functions::

1. Using the coefficient #1, the user specifies whether the comparator is to check if the input is "less than" the comparator setpoint (coefficient #1 = 0), or "greater than" the comparator setpoint (coefficient #1 = 1).
2. The comparator setpoint is entered by the user in coefficient # 2.
3. If deadband for the comparator setpoint is required, it will be entered by the user in coefficient #3
4. Output #1 provides status indication on whether analog input is "less than" or "greater than" the comparator setpoint (see (1) above).
5. Output #2 provides comparator setpoint as specified by the user in coefficient #2.

***Scope of Application:***

**Physical effects simulated:**

- The effects of a comparator switch (e.g. level switch etc.)

**Limitations or effects not simulated:** Not applicable

**Validity check:** Not applicable

**Range of Operation:**

- Input values can assume any values from -99999999.99999999 to +99999999.99999999

**Special Requirements:**

- None

**Input Specifications (ALG # 232):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Analog Input | A | Nil | Connect to process variable (e.g. flow, temperature, pressure, etc.) |

**Output Specifications (ALG #232):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Digital Output (D1) (status indication on whether analog input is less than or greater than setpoint.) | D | Nil | If the analog input is less than or greater than the setpoint (depending on coefficient #1); this output will be "1"; else, it will be "0". This output is usually connected to alarm and control logic etc. |
| 2. Comparator setpoint (A1) (equated to coefficient #2) | A | Nil | mainly for announcing the comparator setpoint value to other blocks. |

**Coefficient Specifications (ALG #232):**

| Coefficient Descriptions | Analog (A)/ Digital (D) | Engineering Units | Calculation Details |
|---|---|---|---|
| 1. Comparison specification: "Less than" or Greater than" | D | Nil | 0 = less than setpoint<br>1 = greater than setpoint |
| 2. Comparator Setpoint | A | The engineering unit is the same as that for analog input | |
| 3. Comparator deadband | A | The engineering unit is the same as that for analog input. | If no deadband is required, enter 0 |

**Algorithm #236: DELTA_INP, Δ Input = (Current Input — Previous Input)**

***Purpose & Descriptions:***

This algorithm performs the following functions:

1. It stores the input value from the previous iteration, then it takes the difference between the current input value (in current iteration) and the previous input value ( in previous iteration).
2. The difference ( or called Δ input) is equated to output #1.
3. Output #2 provides the absolute value of the Δ input.
4. Output #3 stores the input value for the previous iteration.
5. Output #4 is a flag to indicate whether the algorithm has executed at least once.

***Scope of Application:***

**Physical effects simulated:** Not applicable

**Limitations or effects not simulated:** Not applicable

**Validity check:** Not applicable

**Range of Operation:**

- Input values can assume any values from -99999999.99999999 to +99999999.99999999

**Special Requirements:**

- None

**Input Specifications (ALG #236):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Analog Input | A | Nil | |

**Output Specifications (ALG #236):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. (Current Input - Previous Input ) (A1) | A | Nil | |
| 2. Absolute value of (Current Input - Previous Input ) (A2) | A | Nil | |
| 3. Previous Input Value (A3) | A | Nil | Mainly for storing previous input value |
| 4. First time through flag (D1) | D | Nil | Mainly for storing flag status |

**Coefficient Specifications (ALG # 236):** None.

**Algorithm #241: SPEEDER_GEAR, Steam Turbine speed/load changer gear**

***Purpose & Descriptions:***

This algorithm simulates a simple speed/load changer gear for a rotational equipment such as a steam turbine, or gas turbine etc. Usually the position of the speeder gear controls the position of the speed/load governor:

1. Users can specify the rate of the speeder gear movement from 0 % governor position to 100 % governor position (or vice versa). The rate can be expressed respectively as Fast Time or Slow Time in the coefficient specifications:
2. Fast Time (coefficient #1) is the time in seconds specified by the user for the speeder gear to move from 0 % to 100% position, if the gear is selected (via Input #4 = 1) to move at the fast rate.
3. Slow Time (coefficient #2) is the time in seconds specified by the user for the speeder gear to move from 0 % to 100% position, if the gear is selected (via Input #5 = 1) to move at the slow rate.
4. Control commands to raise or lower the speed/load changer gear are received at Input #1 and #2 respectively. If there is no speeder gear failure signal (at Input #3 (Fail to Position) = -1), and there is no Manual control signal (at Input # 6 = 0), the speeder gear will move to a new position with each increment of DT, the rate of gear movement is determined by whether the fast or slow rate is selected (Input #4 = 1 or Input #5 = 1).
5. The **normalized** speeder gear position is continuously updated in Output # 1.
6. When the speeder gear has reached its maximum limit, Output #2 will be equal to 1.
7. When the speeder gear has reached its minimum limit, Output #3 will be equal to 1.
8. Malfunction signal of the speeder gear can be sent to Input #3. Any value other than "- 1.0" will be treated as the speeder gear "fail-to-position" value.
9. If manual control of the speeder gear is required, send a digital signal to Input #6 which will switch the gear to manual control. The analog value specified in Input #7 will indicate the desired manual controlled position of the gear.

***Scope of Application:***

**Physical effects simulated:**

- Simple turbine speed/load changer gear.

**Limitations or effects not simulated:**

- Effects of governor hunting are not simulated

**Validity check:**

- None

**Range of Operation:**

- All digital input signals assume value either "0" or "1".
- All analog input signals assume normalized value.

**Special Requirements:**

- None

**Input Specifications (ALG # 241):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Raise Speed/Load Gear | D | Nil | Connect to control signal |
| 2. Lower Speed/Load Gear | D | Nil | Connect to control signal |
| 3. Speeder Gear "Fail-to-position" value | A | Normalized | Connect to malfunction block output. If there is no malfunction, enter $ constant with value = -1. |
| 4. Fast Rate Selected | D | Nil | Connect to governor control logic |
| 5. Slow Rate Selected | D | Nil | Connect to governor control logic |
| 6. Manual Control Selected | D | Nil | Connect to manual control logic |
| 7. Manual Control Signal | A | Normalized | Connect to manual control logic |

**Output Specifications (ALG #241):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Speed/Load Changer Gear Position (Z1) | A | Normalized | Connect to speed relay, servomotor and governor control valve |
| 2. Gear at maximum position (HL) | D | Nil | For status indication |
| 3. Gear at minimum position (LL) | D | Nil | For status indication |

**Coefficient Specifications (ALG # 241):**

| Coefficient Descriptions | Analog (A)/ Digital (D) | Engineering Units | Calculation Details |
|---|---|---|---|
| 1. Fast Time | A | seconds | Nil |
| 2. Slow Time | A | seconds | Nil |

**Algorithm #312: 5_SUM, Sum of 5 Inputs**

***Purpose & Descriptions:***

This algorithm add 5 Analog Inputs together and produce the sum at the Output:

**Output #1 = Input #1 * Coefficient #1 + Input #2 * Coefficient #2 + Input #3 * Coefficient #3 + Input #4 * Coefficient #4 + Input #5 * Coefficient #5**

1. Coefficient #1 to #5 are multiplying factors to be used for individual Analog Input, if it is necessary to scale respective inputs for the addition.

***Scope of Application:***

**Physical effects simulated:** Not applicable

**Limitations or effects not simulated:** Not applicable

**Validity check:** Not applicable

**Range of Operation:** Not applicable

**Special Requirements:** Not applicable

**Input Specifications (ALG # 312):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Analog Input #1 | A | Nil | |
| 2. Analog Input #2 | A | Nil | |
| 3. Analog Input #3 | A | Nil | |
| 4. Analog Input #4 | A | Nil | |
| 5. Analog Input #5 | A | Nil | |

**Output Specifications (ALG # 312):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Output (A1) | A | Nil | |

**Coefficient Specifications (ALG # 312):**

| Coefficient Descriptions | Analog (A)/ Digital (D) | Engineering Units | Calculation Details |
|---|---|---|---|
| 1. Multiplying Factor for Input #1 | A | Nil | |
| 2. Multiplying Factor for Input #2 | A | Nil | |
| 3. Multiplying Factor for Input #3 | A | Nil | |
| 4. Multiplying Factor for Input #4 | A | Nil | |
| 5. Multiplying Factor for Input #5 | A | Nil | |

**Algorithm #332: PID_MALF, Proportional + Integral + Derivative Controller with Controller Malfunction**

***Purpose & Descriptions:***

This algorithm is exactly the same as ALG # 330, except that controller malfunction is added for its output to fail to any demanded position.

1. This algorithm represents a proportional + Integral + Derivative Controller.

2. The mathematical formulation for the simple controller is as follows:
   Output = Proportional Term + Integral Term + Derivative Term
   The **normalized** controller output is continuously updated at Output #1.; whereas the controller output signal in % is displayed at Output #15
   Output #2 displays Error signal = Process Variable (PV) - Setpoint (SP).
   Output #3 displays the Proportional Term calculated in the algorithm.
   Output #4 displays the Integral Term calculated in the algorithm.
   Output #5 displays the Derivative term calculated in the algorithm.
3. The controller algorithm computes the error signal at Output #2 based on a "forward-acting" action: i.e. **Forward-Acting:** Error = Process Variable (PV) - Setpoint (SP). However, most controller action is **Reverse-Acting:** Error = Setpoint (SP) - Process Variable (PV). Hence Coefficient #4 can be used to determine what the controller action should be
   ( Coefficient #4: 0 means **Forward Acting**; 1 means **Reverse Acting**).
4. The controller has 4 modes of operations:
   (a) **Auto** Mode - In Auto Mode, the Setpoint (SP) is provided by local user input. The
   controller output is determined by the P + I + D control action taken by the controller.
   Auto mode can be requested from two sources :
   (i) Man-Machine Interface (MMI) - Input #2
   (ii) Internal logic from Model - Input #3
   When Auto mode request is accepted, Output #6 will indicate that the controller is in Auto Mode.
   In Auto Mode, the Setpoint signal comes from Man-Machine Interface (MMI) - Input #16.
   (b) **Manual** Mode - In Manual Mode, automatic controller action is disabled. User manually
   controls the controller's output.
   Manual Mode request is received at Input #4 through MMI request and when it is accepted, Output #7 will indicate that the controller is in Manual Mode.
   In Manual Mode, the Manual control signal comes from Input #20 via MMI input.
   (c) **Remote** Setpoint Mode (RSP) - In RSP Mode, the Setpoint (SP) is provided NOT by the
   user, but by the internal logic.
   RSP Mode can be requested from two sources:
   (i) Man-Machine Interface (MMI) - Input #6
   (ii) Internal logic from Model - Input # 7

When RSP mode requested is accepted, Output #8 will indicate that the controller is in RSP Mode.

In RSP Mode, the Setpoint signal comes from the Model via Input #17.

(d) **Tripped-To-Manual** Mode - When the controller is in any of the above control modes, if Input #5 is "true" (= 1), the controller will be "tripped to Manual". Output #7 will indicate that the controller is in Manual; whereas Output #27 will indicate that the controller has been "tripped to manual". Logic Model may send a Manual Control signal at Input #21, in order to set the controller output to a desired value, should the controller be "tripped to Manual".

The " Tripped to Manual" mode will be reset when Input #27 is "true". When that happens, Output #27 will be reset to "0", and the controller will remain in "Manual" mode ((b) above).

5. There is a provision to "track" a controller when Input #14 (Track Command) is "true" ( = 1). When that happens, regardless of what mode the controller is in, the controller output #1 will follow the signal provided at Input #15 (Track Value). Output #9 will indicate that the controller is in "TRACK".
6. The controller can be locked in a particular mode:
   Input #11 = Lock Auto
   Input #12 = Lock Manual
   Input #13 = Lock RSP
   It is important for the user to make sure that the"lock" signals for the 3 modes (Input #11, #12, #13) cannot be applied simultaneously.
   When the mode "lock" signal is accepted, Output #10 will indicate that the controller is in "lock".
7. The controller can be inhibited from "Auto", "Manual" or "RSP" mode via respective Input #8, Input #9 and Input #10. The "Inhibit" function will prevent the inhibited mode from being set. When a mode inhibit function occurs, Output #12, #13, #14 will display the mode inhibited function respectively.
8. The controller accepts "feedforward" bias via Input #18 for MMI, or via Input #19 for Model input.
9. Usually the controller tuning constants: Proportional Gain, Reset Time, Set Time are set by the respective coefficient #1 , #2, #3. However, this algorithm provides a tuning provision with which the user can tune the controller "on-line" by sending a "controller tune" request at Input #22. When that happens, Proportional Gain can be provided at Input #23; Reset Time at Input #24; Set Time at Input #25.
10. The controller output can have a low and high limits specified by the user respectively at the Coefficient #5 and #6.
11. Note that for the purpose of interfacing to Labview GUI, a special algorithm #457 has been designed to interface with this algorithm. See details in ALG # 457.
12. This algorithm accepts controller malfunction request via Input #28 (failed controller flag); and via Input #29 (failed controller output position) If the controller has failed, then the controller output will be set to equal Input #29.

*Scope of Application:*

**Physical effects simulated:** Not applicable

**Limitations or effects not simulated:** Not applicable

**Validity check:** Not applicable

**Range of Operation:** Not applicable

**Special Requirements:** Not applicable

**Input Specifications (ALG # 332):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Process Variable (PV) | A | to follow input | |
| 2. MMI Auto Mode Request | D | Nil | See 4(a) above |
| 3. Model Logic Auto Mode Request | D | Nil | See 4(a) above |
| 4. MMI Manual Mode Request | D | Nil | See 4(b) above |
| 5. Model Trip to Manual Request | D | Nil | See 4(d) above |
| 6. MMI RSP Model Request | D | Nil | See 4(c) above |
| 7. Model Logic RSP Request | D | Nil | See 4(c) above |
| 8. Inhibit Auto Mode | D | Nil | See 7 above |
| 9. Inhibit Manual Mode | D | Nil | See 7 above |
| 10. Inhibit RSP Mode | D | Nil | See 7 above |
| 11. Lock Auto Mode Request | D | Nil | See 6 above |
| 12. Lock Manual Mode Request | D | Nil | See 6 above |
| 13. Lock RSP Mode Request | D | Nil | See 6 above |
| 14. Controller Track Command | D | Nil | See 5 above |
| 15. Controller Track Value | A | Normalized | See 5 above |
| 16. MMI Setpoint Signal | A | to follow input | See 4(a) above |
| 17. Model Logic Setpoint Value | A | to follow input | See 4(c) above |
| 18. MMI Feedforward bias | A | Normalized | See 8 above |
| 19. Model Logic Feedforward bias | A | Normalized | See 8 above |
| 20. MMI Manual Control signal | A | Normalized | See 4(b) above |
| 21. Model Logic Manual Control Signal | A | Normalized | See 4(d) above |
| 22. MMI Controller Tuning Request | D | Nil | See 9 Above |
| 23. MMI Proportional Gain | A | Units of Prop Term/Units of Error | See 9 above |
| 24. MMI Reset Time | A | sec | See 9 above |
| 25. MMI Set Time | A | sec | See 9 above |
| 26. MMI to set Controller Action: Forward Acting or Reverse Acting. | D | Nil | See 9 above.<br>0 = Forward Acting; 1 = Reverse Acting |

| | | | |
|---|---|---|---|
| 27. MMI to Reset "Tripped to Manual" | D | Nil | See 4(d) above |
| 28. Controller Failed Flag | D | Nil | See 12 Above |
| 29. Controller Output Failed Position | A | Normalized | See 12 Above |

**Output Specifications (ALG # 332):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Controller Output Signal (Z1) | A | Normalized | Connect to control device |
| 2. Error Signal (ER1) = (PV - SP) | A | to follow PV & SP | For algorithm internal use |
| 3. Proportional Term (A1) | A | Normalized | For algorithm internal use |
| 4. Integral Term (A2) | A | Normalized | For algorithm internal use |
| 5. Derivative Term (A3) | A | Normalized | For algorithm internal use |
| 6. Controller in Auto Mode (D1) | D | Nil | For Labview pop-up interface, see connection details in ALG # 457 |
| 7. Controller in Manual (D2) | D | Nil | For Labview pop-up interface, see connection details in ALG # 457 |
| 8. Controller in RSP (D6) | D | Nil | For Labview pop-up interface, see connection details in ALG # 457 |
| 9. Controller in Track (D3) | D | Nil | For Labview pop-up interface, see connection details in ALG # 457 |
| 10. Controller in Lock (D4) | D | Nil | For display. |
| 11. Controller Mode (A13) 0 = Auto; 1= Manual; 2 = RSP | A | Nil | For Labview pop-up interface, see connection details in ALG # 457 |
| 12. Auto Mode Inhibited (D7) | D | Nil | For Labview pop-up interface, see connection details in ALG # 457 |
| 13. Manual Mode Inhibited (D8) | D | Nil | For Labview pop-up interface, see connection details in ALG # 457 |
| 14. RSP Mode Inhibited (D9) | D | Nil | For Labview pop-up interface, see connection details in ALG # 457 |
| 15. Controller Output Signal (A4) | A | % | |
| 16. Controller Tune Flag (D5) | D | Nil | For Labview pop-up interface, see connection details in ALG # 457 |
| 17. Proportional Gain registered (A5) | A | Units of Prop. Term/Units of Error | For Labview pop-up interface, see connection details in ALG # 457 |
| 18. Reset Time Registered (A6) | A | sec. | For Labview pop-up interface, see connection details in ALG # 457 |
| 19. Set Time Registered (A7) | A | sec. | For Labview pop-up interface, see connection details in ALG # 457 |
| 20. Controller Action Registered: Forward Acting or Reverse Acting (A8) | A | Nil | For Labview pop-up interface, see connection details in ALG # 457 |

| | | | |
|---|---|---|---|
| 21.Controller Output Low Limit (A9) | A | Normalized | For algorithm internal use |
| 22.Controller Output High Limit (A10) | A | Normalized | For algorithm internal use |
| 23.Controller Setpoint Registered (A11) | A | to follow SP | For Labview pop-up interface, see connection details in ALG # 457 |
| 24.Controller Feedforward Bias Registered (A12) | A | Normalized | For Labview pop-up interface, see connection details in ALG # 457 |
| 25. MMI Manual Control Signal Update Request (D10) | D | Nil | For Labview pop-up interface, see connection details in ALG # 457 |
| 26. MMI Setpoint Update Request (D11) | D | Nil | For Labview pop-up interface, see connection details in ALG # 457 |
| 27. Controller Tripped to Manual (D12) | D | Nil | For Labview pop-up interface, see connection details in ALG # 457 |

**Coefficient Specifications (ALG # 332):**

| Coefficient Descriptions | Analog (A)/ Digital (D) | Engineering Units | Calculation Details |
|---|---|---|---|
| 1. Proportional Gain | A | Units of Prop Term/Units Errors | |
| 2. Reset Time for Integral Term | A | sec. | |
| 3. Set Time for Derivative Term | A | Sec. | |
| 4. Controller Action: Forward Acting or Reverse Acting | D | Nil | 0 = means Forward Acting<br>1 = means Reverse Acting |
| 5. Controller Output Low Limit | A | Normalized | |
| 6. Controller Output High Limit | A | Normalized | |

**Algorithm #398: MARKER, Dummy Marker for Model Identification**

***Purpose & Descriptions:***

This algorithm performs no operation at all. It is intended to be used only as a Model Identifier.

***Scope of Application:***

**Physical effects simulated:** Not applicable

**Limitations or effects not simulated:** Not applicable

**Validity check:** Not applicable

**Range of Operation:** Not applicable

**Special Requirements:** Not applicable

**Input Specifications (ALG # 398):**
No Inputs.

**Output Specifications (ALG # 398):**
No Outputs.

**Coefficient Specifications (ALG # 398):**
No Coefficients.

**Algorithm #399: DUMMY_DISPLAY, Dummy Display of 20 Output Tags**

*Purpose & Descriptions:*

1. This algorithm is designed for debugging the model, where the user wants to display a group of output tags, along with their values updated continuously from the model. These output tags may come from various outputs from different blocks in the model.
2. Insofar as the algorithm is concerned, there is no computation at all inside this algorithm.
3. User may delete the block using this algorithm after model debugging.

*Scope of Application:*

**Physical effects simulated:** Not applicable

**Limitations or effects not simulated:** Not applicable

**Validity check:** Not applicable

**Range of Operation:** Not applicable

**Special Requirements:** Not applicable

**Input Specifications (ALG # 399):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Analog Input #1 | A | Nil | Connect to the Output tag of a specific block's output |
| 2. Analog Input #2 | A | Nil | Connect to the Output tag of a specific block's output |
| 3. Analog Input #3 | A | Nil | Connect to the Output tag of a specific block's output |
| . | " | " | " |
| . | " | " | " |
| . | " | " | " |
| 18. Analog Input #18 | A | Nil | Connect to the Output tag of a specific block's output |
| 19. Analog Input #19 | A | Nil | Connect to the Output tag of a specific block's output |
| 20. Analog Input #20 | A | Nil | Connect to the Output tag of a specific block's output |

**Output Specifications (ALG # 399):**

No Outputs.

**Coefficient Specifications (ALG # 399):**

No Coefficients.

**Algorithm #401: H_NET_INT, Internal Node**

***Purpose & Descriptions:***

1. This algorithm is used in conjunction with H_NET_ID(network ID block), H_NET_EXT(external node block), and H_NET_LINK(network link block).

2. The pressure and flow calculation is performed by reducing the momentum equation to a linear form, and from the mass balance on each of the node, a system of linear algebraic equations can be formed. The solution of the algebraic system will give the pressure for each of the node.

3. Please see *CASSIM USER MANUAL* **section 2.3.4** and *CASSIM TUTORIAL* **section 2.3** on hydraulic flow networks for further explanations.

4. Each node can be considered as a control volume with a constant pressure within the control volume.

5. Any flows to and from the node that are not related to the network link can be inputted to INP(1) from the node.

***Scope of Application:***

**Physical effects simulated:**

- Pressure in the node will respond to a mismatch between the inflow and outflow of the node.

**Limitations or effects not simulated:**

- The node pressure calculation due to boiling is not modelled

**Validity check: None**

**Range of Operation:**

- network flows remain in a single phase, for 2 phase systems, a 2 phase friction factor has to be calculated outside the network blocks.

**Special Requirements & Assumptions:**

- This block must be used in conjunction with other blocks to form the network system.

**Input Specifications(ALG # 401):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Sum of external flows | A | Kg/sec | Connect to link flow output tag of the link block in which the inlet stream 1 is located. |

**Output Specifications(ALG # 401):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Node pressure(P1) | A | KPaa | May be used as indication, control or other network calculation. This output is required to be initialized at the design node pressure. |
| 2. Capacity/dt(G1) | A | | This output is only required by the network solver |
| 3. Node pressure* capacity/dt(G2) | A | | This output is only required by the network solver |
| 4. Sum of external flows (FM1) | A | Kg/sec | This output is the same as INP(1) |
| 5. Not used | A | | |

**Coefficient Specifications(ALG # 401):**

| Coefficient Descriptions | Analog (A)/ Digital (D) | Engineering Units | Calculation Details |
|---|---|---|---|
| 1. Node capacity | A | KPaa/Kg | The capacity of the node can be derived from the volume of the control volume(e.g. a mixing header) * the change of density with pressure for the fluid in the node. See *CASSIM TUTORIAL* **section 2.3.4.** |

**Algorithm #402: H_NET_EXT, External Node**

***Purpose & Descriptions:***

1. This algorithm is used in conjunction with H_NET_ID(network ID block), H_NET_INT(internal node block), and H_NET_LINK(network link block).

2. The pressure and flow calculation is performed by reducing the momentum equation to a linear form, and from the mass balance on each of the node, a system of linear algebraic equations can be formed. The solution of the algebraic system will give the pressure for each of the node.

3. Please see *CASSIM USER MANUAL* **section 2.3.4** and *CASSIM TUTORIAL* **section 2.3** on hydraulic flow networks for further explanations.

4. This block can be seen as the interface block to represent the existence of an external node.

5. The external pressure is usually calculated outside of this block.

***Scope of Application:***

**Physical effects simulated:** None

**Limitations or effects not simulated:** None

**Validity check:** None

**Range of Operation:**

- See *CASSIM USER MANUAL* **section 2.3.4** and *CASSIM TUTORIAL* **section 2.3** on hydraulic flow networks for further explanations.

**Special Requirements & Assumptions:**

- This block must be used in conjunction with other blocks to form the network system.

**Input Specifications(ALG # 402):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. External Pressure | A | KPaa | Connect to the block which calculate the external node pressure. It can be a constant tag, e.g. atmospheric pressure<br>Or the pressure may be from an internal node block from another network circuit. |

**Output Specifications(ALG # 402):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Node pressure(P1) | A | KPaa | May be used as indication, control or other network calculation. |

**Coefficient Specifications(ALG # 402):** None

**Algorithm #403: H_NET_LINK, Hydraulic Network Link**

***Purpose & Descriptions:***

1. This algorithm is used in conjunction with H_NET_ID(network ID block), H_NET_EXT(external node block), and H_NET_INT(internal node block).

2. The pressure and flow calculation is performed by reducing the momentum equation to a linear form, and from the mass balance on each of the node, a system of linear algebraic equations can be formed. The solution of the algebraic system will give the pressure for each of the node.

3. With the pressure difference between the two connected nodes, the link flow can be calculated using a linearized form of the momentum equation.

4. Please see *CASSIM USER MANUAL* **section 2.3.4** and *CASSIM TUTORIAL* **section 2.3** on hydraulic flow networks for further explanations.

4. The link flow as per the momentum equation is calculated in this algorithm.

5. The momentum equation used is assuming **incompressible** flow, therefore the momentum equation can be reduced to the form:

   $$W = KCOND*TANK\_LVL\_FAC*VAL1*VAL2*(P_{up}+P_{elev}+P_{static}-P_{down})^{0.5}$$

   where:

   **KCOND**, the link conductance is calculated based on the link mass flow and the link pressure drop at design conditions. ***For cases, where the momentum equation for the link does not follow the above form, e.g. compressible flow, or pump head/flow curve, an external conductance has to be calculated external to this algorithm, and connected to INP(9) of this algorithm. COF(8) has to be set to one, if external conductance is used.***

   **TANK_LVL_FAC** is the tank level factor (see ALG128 and input specifications below for more details)

   **VAL1, VAL2**, is the normalized valve port-area for the valves in the link

   $P_{up}$ is the pressure at the upstream node

   $P_{down}$ is the pressure at the downstream node

$\mathbf{P_{elev}}$ is the elevation head of the piping arrangement in the link connecting the two nodes.

$\mathbf{P_{static}}$ is the static head for the pump located in the link

6. If there are valves located on this link, the normalized valve port-areas must be connected.(INP(3), INP(4)). These valve port-areas will reduce the conductance of the link thereby reducing the link flow.

7. Any elevation head related to this link must be connected to INP(6).

8. Non-return valves can be modelled in the link, such that if downstream pressure is greater than the upstream pressure, the flow is equalled to zero. (Set COF(1) to 1 if a non-return valve exists in the link.)

*Scope of Application:*

**Physical effects simulated:**

- Flow calculation based on incompressible flow
- Reverse flows can be simulated.

**Limitations or effects not simulated:**

- 2-phase flow not modelled. If it is required, then calculate flow external to this block, obtain an equivalent conductance, and connected to the external conductance input for the link block.
- Link algorithm only allows 2 valve in parallel in the link. If more than 2 valve exists in the link. The overall valve port-area must be calculated in a separate block, by multiplying all the effective normalized valve port-area together, before connecting to this link block.
- Link algorithm assumes the link's momentum equation is of the incompressible form. For compressible flows or a link flow which has to match a pump head curve, the suitable momentum equation has to be calculated outside the network blocks, and has to be connected to the link block using the external conductance input. See *CASSIM TUTORIAL* **section 2.3.3.3 & section 2.3.3.4.**

**Validity check: None**

**Range of Operation:**

- network flows remain in a single phase, for 2 phase systems, a 2 phase friction factor has to be calculated outside the network blocks.

**Special Requirements & Assumptions:**

- This block must be used in conjunction with other blocks to form the network system.
- Non-return valves are assumed to be perfect.

**Input Specifications(ALG # 403):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Upstream pressure | A | KPaa | Connect to the network node block which calculate the upstream node pressure |
| 2. Downstream pressure | A | KPaa | Connect to the network node block which calculate the downstream node pressure |
| 3. Valve area 1 | A | Normalized | Connect to the valve block port-area output. If no valve exists on this link, then a constant value of 1 must be inputted. |
| 4. Valve area 2 | A | Normalized | Connect to the valve block port-area output. If no valve exists on this link, then a constant value of 1 must be inputted. |
| 5. Pump static head | A | KPaa | Connect to the pump dynamics block(ALG102), static head output. If no pump exists for this link, set this term to zero. |
| 6. Elevation head | A | M | Connect to the a block which calculate the elevation head. For static elevation head, a constant tag with a value representing the elevation should be used.(note: the elevation head is defined as (the elevation at the upstream node - the elevation at the downstream node.)) |
| 7. Density of fluid | A | $Kg/M^3$ | Connect to the a block which calculate the density of fluid for the link. Density can be made a function of temperature using a curve-fit block. |
| 8. Tank level factor | A | Normalized | If the upstream node of the link is an external node representing an accumulation vessel.(e.g. a tank, or drum), connect this input to the block output simulating the vessel's tank level factor. Otherwise, set this input as a constant tag with a value of 1. (see ALG128 for explanation of tank level factor. Note: some algorithm requires extra block to calculate tank level factor. e.g. deaerator) |
| 9. External conductance | A | $Kg/KPaa^{0.5}$ | Connect to link conductance output tag of the link block where the flow is calculated using a different form of the momentum equation. |

**Output Specifications(ALG # 403):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Link flow(FM1) | A | Kg/sec | May be used as indication, control or other network calculation. |
| 2. Link admittance (AA1) | A | | This output is only required by the network solver |
| 3. Link admittance* head (G1) | A | | This output is only required by the network solver |
| 4. Conductance (CN1) | A | $Kg/KPaa^{0.5}$ | This output is only required by the network solver |
| 5. Pump dynamic head (G2) | A | KPaa | |

**Coefficient Specifications(ALG # 403):**

| Coefficient Descriptions | Analog (A)/ Digital (D) | Engineering Units | Calculation Details |
|---|---|---|---|
| 1. Check valve status | D | Nil | If a non-return valve is present in the link, set this coefficient to 1, else set it to 0 |
| 2. Design flow | A | Kg/sec | Refer to *CASSIM TUTORIAL* **section 2.3.4.** |
| 3. Design delta pressure | A | KPaa | Refer to *CASSIM TUTORIAL* **section 2.3.4.** |
| 4. Gravitational constant | A | $KPa*M^2/Kg$ | The value is 0.00981 |
| 5. Design valve opening | A | Normalized | Refer to *CASSIM TUTORIAL* **section 2.3.4.** |
| 6. Design pump static head | A | KPaa | This value should be obtained from the equipment manufacturer. This value is the no load static head and should match with the value specify in the pump dynamics block. |
| 7. Design pump dynamic head | A | KPaa | This value should be obtained from the equipment manufacturer |
| 8. Use external conductance as input | D | KPaa/Kg | If conductance of the link is variable, and is calculated outside of this algorithm, set this coefficient to 1, else set it to 0. |

**Algorithm #451: TIMEKEEPER**

***Purpose & Descriptions:***
This algorithm outputs current simulation time and current simulation iteration count.

- Simulation Time
  The simulation time is represented as an eight-digit double precision number. Starting from the left, digits 1 and 2 represent the hour, digits 3 and 4 represent the minute, digits 5 and 6 represent the second, and digits 7 and 8 represent half second.

  There are two ways to use the simulation time with TIMEKEEPER:
  1) Use the PC's current clock time as the initial simulation start time
     Code the data value at output 1 as zero.
  2) Specify a particular initial simulation start time
     Code a non-zero time as the data value at output 1. Use the eight-digit format as described above.

  Each iteration increments the simulation time by dt (the iteration time step specified in the model).

- Simulation Iteration Count
  There are two ways to use the simulation iteration count with TIMEKEEPER:
  1) Start the simulation count initially at zero
     Code the data value at output 2 as zero.
  2) Specify a particular initial simulation count
     Code a non-zero count as the iteration count at output 2

  Each subsequent simulation iteration will increment the iteration by 1.

- Output 1 must be hooked up to input 1. Output 2 must be hooked up to input 2.

***Scope of Application:***

**Physical effects simulated:** Not Applicable.

**Limitations or effects not simulated:** Not Applicable.

**Validity check:** Not Applicable.

**Range of Operation:** Not Applicable.

**Special Requirements:** Not Applicable.

**Input Specifications (ALG # 451):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Simulation Time | A | nil | Connect to simulation time output of this same block. |
| 2. Simulation Iteration | A | nil | Connect to simulation iteration output of this same block. |

**Output Specifications (ALG # 451):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Simulation Time (TM1) | A | nil | Connect to simulation time input of this same block. |
| 2. Simulation Iteration (IT1) | A | nil | Connect to simulation iteration input of this same block. |

**Coefficient Specifications (ALG # 451):**
None.

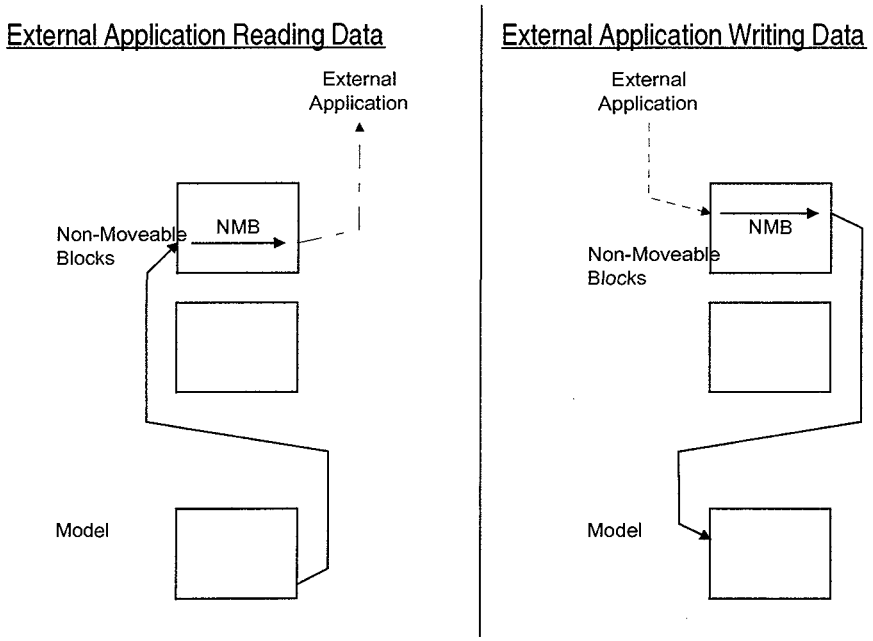
**Algorithm #452: NON-MOVEABLE, Non-moveable Block NMB**

***Purpose & Descriptions:***
A non-moveable block is to facilitate external applications reading and writing simulation data.

This algorithm copies each of the 99 inputs to the corresponding output.

- To facilitate external applications reading data from the model
  Simulation data is made available to external applications by connecting the data to the inputs of this block. The algorithm will copy the input data to the outputs. When an external application wants to read simulation data, read from the outputs.

- To facilitate external applications writing data to the model
  To pass input data from the external application to the simulation, write to the inputs. The algorithm will copy the input data to the outputs. The outputs are connected to the blocks which will accept and process the inputs. Note that the input data is a steady signal (value) which remains at the input once written. The output will continue to show the same value until the value is changed in a subsequent write.




***Scope of Application:***

**Physical effects simulated:** Not Applicable.

**Limitations or effects not simulated:** Not Applicable.

**Validity check:** Not Applicable.

**Range of Operation:** Not Applicable.

**Special Requirements:** None.

**Input Specifications (ALG # 452):**

| Input Descriptions | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Input 1 | A | nil | If used for outputting a value to an external application, connect to the block output where the value originates. The value will be copied to this block's corresponding output for reading by external application.<br>If used for accepting input from external application, configure as a CONSTANT tag. External applicationwill be writing to this input. |
| . | . | . | " |
| . | . | . | " |
| . | . | . | " |
| 99. Input 99 | A | nil | " |

**Output Specifications (ALG # 452):**

| Output Descriptions/ Transform | Analog (A)/ Digital (D) | Engineering Units | Connection Details |
|---|---|---|---|
| 1. Output 1 (XX01) | A | nil | If used for accepting input from external application, the external application will be writing to the corresponding input. The value will be passed to this output. Connect this output to the block which will accept the input value. |
| . | . | . | " |
| . | . | . | " |
| . | . | . | " |
| 99. Output 99 (XX99) | A | nil | " |

**Coefficient Specifications (ALG # 452):**
None.