DIGITAL COMPUTER SUPERVISION
OF AN ANALOG
NUCLEAR POWER PLANT
SIMULATION

The author is indebted to Prof. J. R. Jenkins, Jr., of the
Chemical Engineering Department, Prof. John C. Courtney, Dr. Frank A.
Jenkins and Dr. Edward N. Lande of the Nuclear Science Center,
and Dr. Myron H. Young of Mechanical Engineering, and Dr. Thomas
Minton of the Electrical Engineering Department at Louisiana State
University for their guidance and encouragement in this work.

A Thesis

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Master of Science

in

The Department of Nuclear Engineering

by

Ernest Ivry Hamilton, Jr.
B.S., Louisiana State University, 1970
May, 1972
ACKNOWLEDGEMENT

The author is indebted to Dr. Adrian E. Johnson, Jr., of the Chemical Engineering Department, Dr. John C. Courtney, Dr. Frank A. Iddings and Dr. Edward N. Lambremont of the Nuclear Science Center, to Dr. Myron H. Young of Mechanical Engineering, and Mr. Thomas Linton of the Electrical Engineering Department at Louisiana State University for their guidance and encouragement in this work.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGEMENT</td>
<td>ii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>v</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>vi</td>
</tr>
<tr>
<td>CHAPTER</td>
<td></td>
</tr>
<tr>
<td>I. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>History, Background and Data Supporting the Reasons for this Work</td>
<td>1</td>
</tr>
<tr>
<td>The Problem</td>
<td>5</td>
</tr>
<tr>
<td>Hybrid Computer Simulation</td>
<td>6</td>
</tr>
<tr>
<td>II. NUCLEAR REACTOR PLANT SIMULATION</td>
<td>8</td>
</tr>
<tr>
<td>Simulation of Neutron Kinetics</td>
<td>8</td>
</tr>
<tr>
<td>The Analog Implementation</td>
<td>10</td>
</tr>
<tr>
<td>III. CONTROL COMPUTER CONCEPTS</td>
<td>22</td>
</tr>
<tr>
<td>Information Flow</td>
<td>22</td>
</tr>
<tr>
<td>Methods of Computer Implementation</td>
<td>22</td>
</tr>
<tr>
<td>Software Requirements</td>
<td>23</td>
</tr>
<tr>
<td>Interrupt Servicing</td>
<td>25</td>
</tr>
<tr>
<td>IV. DIGITAL DATA ACQUISITION AND CONTROL PACKAGE</td>
<td>29</td>
</tr>
<tr>
<td>The Function of the Package</td>
<td>29</td>
</tr>
<tr>
<td>The Description of the Package</td>
<td>29</td>
</tr>
<tr>
<td>Timing Module</td>
<td>31</td>
</tr>
<tr>
<td>Process-Computer Interface Module</td>
<td>32</td>
</tr>
<tr>
<td>Control Module</td>
<td>33</td>
</tr>
<tr>
<td>Human-Computer Interface Module</td>
<td>34</td>
</tr>
<tr>
<td>Intra-Structure Communication Module</td>
<td>36</td>
</tr>
<tr>
<td>The Use of the Package</td>
<td>36</td>
</tr>
<tr>
<td>V. ACTION OF A DIGITAL SUPERVISORY CONTROL PROGRAM</td>
<td>42</td>
</tr>
<tr>
<td>Digital Control Program</td>
<td>42</td>
</tr>
<tr>
<td>Response of Process to Digital Supervisory Control</td>
<td>43</td>
</tr>
</tbody>
</table>
VI. CONCLUSIONS AND RECOMMENDED FUTURE TOPICS .......................... 49
  Conclusions ........................................................................... 49
  Recommended Future Topics ................................................. 49
LITERATURE CITED ..................................................................... 51
LIST OF SYMBOLS ..................................................................... 52
APPENDIX A ............................................................................. 56
  Analog Patching Schematics ................................................... 56
APPENDIX B ............................................................................. 75
  Digital Process Control Computer Software .......................... 75
VITA ......................................................................................... 99
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>Schematic of PWR Primary Loop</td>
<td>12</td>
</tr>
<tr>
<td>2-2.1</td>
<td>Heat Generation and Coolant Transfer Loop Equations</td>
<td>15</td>
</tr>
<tr>
<td>2-2.2</td>
<td>Heat Generation and Coolant Transfer Loop Equations</td>
<td>16</td>
</tr>
<tr>
<td>2-3</td>
<td>Control Systems Equations</td>
<td>17</td>
</tr>
<tr>
<td>2-4.1</td>
<td>Analog Magnitude Scaled Heat Generation and Coolant Transfer Loop Equations</td>
<td>18</td>
</tr>
<tr>
<td>2-4.2</td>
<td>Analog Magnitude Scaled Heat Generation and Coolant Transfer Loop Equations</td>
<td>19</td>
</tr>
<tr>
<td>2-5</td>
<td>Analog Magnitude Scaled Control Systems Equations</td>
<td>20</td>
</tr>
<tr>
<td>2-6</td>
<td>Reactor Scram Response</td>
<td>21</td>
</tr>
<tr>
<td>3-1</td>
<td>Interrupt Operation</td>
<td>26</td>
</tr>
<tr>
<td>4-1</td>
<td>Structure of Hybrid Package</td>
<td>30</td>
</tr>
<tr>
<td>4-2</td>
<td>Operators Console Functions</td>
<td>38</td>
</tr>
<tr>
<td>4-3.1</td>
<td>Control Table</td>
<td>39</td>
</tr>
<tr>
<td>4-3.2</td>
<td>Control Table</td>
<td>40</td>
</tr>
<tr>
<td>4-4</td>
<td>Variable Table</td>
<td>41</td>
</tr>
<tr>
<td>5-1</td>
<td>System Response to Step Change in Steam Valve Opening with no Supervisory Control</td>
<td>45</td>
</tr>
<tr>
<td>5-2</td>
<td>System Response to Step Change in Steam Valve Opening with only Proportional Control</td>
<td>46</td>
</tr>
<tr>
<td>5-3</td>
<td>System Response to Step Change in Steam Valve Opening with Proportional and Integral Control</td>
<td>47</td>
</tr>
<tr>
<td>5-4</td>
<td>System Response to Step Change in Steam Valve Opening with Proportional and Integral Control</td>
<td>48</td>
</tr>
</tbody>
</table>
ABSTRACT

The results of the development of a prototype hybrid computer package to study various proposed digitally controlled systems are presented. The completed package, consisting of a digital control system linked to an analog-simulated process, is the nucleus from which other more elaborate control studies can easily be performed.

The specific physical system implemented in this work to demonstrate the functioning of the package deals with digital computer supervisory control of an analog computer simulated pressurized water nuclear reactor power plant. The prototype was developed utilizing the facilities and equipment of the Louisiana State University Chemical Engineering Hybrid Simulation Laboratory.

The prototype is not limited to study of simulated processes only. By using the analog computer as an interface almost any process with measuring lines generating electrical signals may be studied. Should control be desired there must be controllers that are operable by electrical signals. This means that control studies can be performed with pilot plants (processes) located within communication range of the hybrid interface.
CHAPTER I

INTRODUCTION

History, Background and Data Supporting the Reasons for this Work

A hybrid computer, which has features common to both digital and analog computers, is a very versatile research instrument. It can be used to study many diverse aspects of both similar and dissimilar topics. In the field of nuclear power plant development, for example, the hybrid computer can explore and evaluate such things as control, safety, and plant design, and can be used to train operators by simulating various situations expected in plant operations and management.

Experience has shown that persons trained on a nuclear power plant simulator are better prepared to pass the Operator Licensing Test given by the Atomic Energy Commission. In designing nuclear power plants, the computer can be used to analyze all systems which can be expected to influence reactor behavior. Reactor kinetics equations and auxiliary mathematical expressions of reactor dynamics can be conveniently and rapidly analyzed.

Although the installation of an on-line computer in conventional steam-electric generating plants is becoming quite common today, computer applications in a nuclear plant presents a somewhat different approach. There has been no concerted effort toward direct digital control in nuclear plants because of the extensive precautions necessary in reactor operations.

In the United States, the trend is to use the computer of a nuclear plant only to accumulate data, perform calculations of internal
reactor parameters or core performance, and as an operational aid. When a plant's computer is out-of-service, the plant can continue to operate at full power or slightly less using the remaining instrumentation. Operation at less than full power may be required when the plant computer is out-of-service if the computer is being used to determine operating limits.

After a sufficient history of reliable computer operations has been accumulated, control and safety systems will become part of the plant's computer functions. After these functions are assumed, the need for redundancy in control systems in nuclear plants will probably result in dual plant computers. A dual plant computer system would be designed so that failure of one plant computer would automatically allow a second computer to take over the functions of the inoperative computer.

The potential applications of a computer in a nuclear power plant include, in general, the following:

1) Data logging (Scan, Convert, and Alarm)
2) Calculations (Plant efficiencies)
3) Operator Aids (Plant conditions)
4) Safety System (Area radiation monitoring, and systems to reduce the possibility of human error)
5) Supervisory Control

The great advantage of on-line computation is that it can provide the plant operator with almost continuous information which permits operating the plant closer to limits set by engineering, maintenance, safety, or procedural considerations.
A study of digital computer supervision of a nuclear power plant revolves around the very real possibility of optimization of an objective function, which is usually net profit in dollars. The study presents a problem common to many endeavors, namely evaluating the proposed system. In evaluating proposed systems or designs one can generally select either of two options: 1) an experimental evaluation, or 2) an analytical evaluation. An experimental program is usually characterized by a minimum of analysis, the construction of a prototype of the system, and considerable trial and error work with the prototype. The cost and time consumed in an experimental evaluation are normally much greater than that required in an analytical evaluation of the same scope. In the analytical approach, the first task is to derive a set of equations (a mathematical model) whose solution will describe the behavior of the variables of the system. Then these equations instead of the experimental prototype are manipulated to generate the desired results.

Since the derivations of mathematical models nearly always require some degree of approximation, some experimentation usually is required for the verification of the model. However, prototypes designed from analytical investigations hopefully portray reality. Then the only experimental results required are those which validate the mathematical model. Once the model is proven valid, additional data can be generated analytically for various operating conditions, which may result in a considerable cost reduction compared to the experimental approach. The analytical approach is not necessarily always better than the experimental. Both are different forms of analysis. The results of experimentation are required to generate the mathematical relations of
parameters (mathematical models) for analytical evaluation, and to determine boundary conditions in some cases.

Electronic computation methods for the solution of mathematical models utilize the digital computer or the analog computer or some combination of both the digital computer and the analog computer called a hybrid computer. In analog computers, the solutions of the mathematical models depend on the analogy between the physical quantities and the mathematical numbers or manipulations. For example, consider the analogy between electrical, mechanical and thermal equations:

\[ i = \frac{dE}{dt} \quad \text{(current flow through a capacitor)} \]

\[ F = \frac{W}{g} \frac{dV}{dt} \quad \text{(Force acting on a mass)} \]

\[ Q = wc \frac{dT}{dt} \quad \text{(Heat flow in a solid)} \]

The form of the differential equation is the same in each case, with the only difference being the constants and the physical meaning of the variables. The variables are represented by scaled voltages in an analog computer. The term "scaled voltages" (Magnitude Scaling) in analog computers means that the voltage output of each amplifier is proportional to the represented problem variable. This proportionality constant is chosen so that the problem variable (temperature, pressure, etc.) will be at maximum or minimum when the analog computer variable (voltage) is maximum or minimum. (Usually this maximum is +10 volts or +100 volts and the minimum is -10 volts or -100 volts, depending on the machine reference voltage.) If a number of events take place at the same time in the real world they will also take place at the same time
in the analog computer simulation. This occurrence at the same time in the real world is termed parallel operation. On an analog computer it is called parallel solution.

Digital computers perform all calculations serially, not in parallel, and therefore, unlike the analog computer, require more and more time as the problem becomes even more complex. In reality the machine performs only Boolean Operations, which are used to build an adder which adds numbers. To subtract, the machine adds the complement of the number to itself the number of times equal to the multiplier. Division has a similar algorithm. All operations that can be performed on a digital computer have algorithms that, in their elementary form, depend only on a certain structure of Boolean Logic. Therefore, with the basic set of operations of Boolean Logic (and, or, not) the computer can perform many operations. (4)

The Problem

The objective of the work to be described in this thesis was to develop a prototype hybrid computer package to study various proposed systems. The package is the nucleus about which other more elaborate studies can easily be made by other persons by using more detailed analog models and by adding more control functions to the digital section. This package has been developed in a modular form, allowing virtually any problem that can be simulated on the analog to be studied. The specific physical system implemented in this work (to demonstrate that the package functions correctly) deals with digital
computer supervision of a simulated nuclear power plant. The prototype was devised utilizing the facilities and equipment of the Louisiana State University Chemical Engineering Simulation Laboratory.

**Hybrid Computer Simulation**

The simulation of a nuclear power plant and its associated plant digital computer is logically suited to hybrid techniques because a nuclear power plant can be conveniently simulated on the analog computer while the digital computer performs the functions of the digital supervisor. In general, analog computers have the following advantages over the digital computer for simulation of physical systems:

1) The speed of the solution is independent of the problem complexity, and can be chosen to be faster or slower than the physical system being simulated.

2) The analogy between the computer simulation variable and the problem variable is straightforward.

3) The values of parameters and input variables can be easily changed during operation, and the results of the change can be observed at the rate of the time scale of the analog simulation.

Digital computers, on the other hand, have quite different advantages compared to the analog computers:

1) They are more precise, and solutions can be made as accurate as the solution time or the mathematical model will allow.

2) They handle logical operations much better than analogs.

3) They can store and manipulate huge quantities of data.
4) With floating point arithmetic, magnitude scaling is no problem.

To enlarge slightly on the advantages of analog and digital computers, consider the concept of information flow in either continuous or discrete states. Continuous information flow is in the realm of the analog computer. An analog computer is made up of electronic components which function basically as operators in the mathematical sense. Information flow in the discrete form is in the realm of the digital computer. In digital computers, structures depending on Boolean Logic serve as mathematical operators.
CHAPTER II

NUCLEAR REACTOR PLANT SIMULATION

Simulation of Neutron Kinetics

The kinetics of a nuclear fission reactor can be approximated by the time and space dependent diffusion equation:

\[ \nabla^2 \Phi + B^2 \Phi = \frac{\partial n}{\partial t} \]

where \( \Phi \) is the neutron flux in \( \text{n/cm}^2\)-sec

\( B^2 \) is the buckling in \( \text{cm}^2 \)

\( n \) is the neutron density in \( \text{n/cm}^3 \)

By considering a point in a reactor away from source sink and boundary that has no spatial variation of neutron flux, the diffusion equation can be simplified to give:

\[ \frac{dn}{dt} = \frac{\delta k - \beta}{1^*} n + \sum_{i=1}^{m} \lambda_i C_i \]

\[ \frac{dC_i}{dt} = \frac{\beta_i}{1^*} n - \lambda_i C_i \]

where \( k \) is the effective multiplication factor of the reactor

\( \delta k \) is \((k-1)/k\), the reactivity

\( \beta \) is the delayed neutron yield

\( 1^* \) is the mean effective lifetime of prompt neutrons in the system

\( m \) is the number of delayed neutron groups

Due \( \lambda_i \) is the decay constant of the \( i \)th group of delayed neutron precursors.

\( C_i \) is the concentration of the \( i \)th group of delayed neutron precursors.
With these simplifications the time and space dependent diffusion equation becomes a single-node kinetics simulator. This single-node simulation has thus neglected the space dependence leaving only the time dependent diffusion equation. This time dependent diffusion equation is an ordinary differential equation, which is easily programmed on an analog computer.

There are three basic types of single-node kinetics simulators:

1) One amplifier per group of delayed neutrons connected to a single amplifier.

2) A single amplifier with a complex input impedance.

3) A single amplifier using a complex feedback impedance.

Each of these three types has its place in nuclear reactor studies. The circuit with one amplifier per group of delayed neutrons is used when amplifier availability is not a problem or when the delayed groups are to have initial conditions, such as in the simulation of an old core. The circuit using a single amplifier with complex input impedance is used when there is a shortage of amplifiers but the variable $dn/dt$ must be available, such as in a multinode reactor core simulation. A multinode reactor core simulation uses a number of coupled space and time dependent diffusion equations, thus giving spatial variation and the variation of neutron flux. The circuit using the single amplifier with complex feedback impedance is used when there is a need to conserve amplifiers and the variable $dn/dt$ is not explicitly needed in the simulation. This is accomplished with no loss of accuracy.

Due to the large range of the power level in a reactor, as much as a factor of $10^{14}$ from source level to peak power level, the simulation
of a nuclear reactor on an analog computer presents a difficult magnitude scaling problem. Since it is impractical to cover more than two or three decades in a single run on an analog computer, there have been several alternate methods developed to treat the problem. One of these methods depends on the nature of logarithms. By a substitution of variables, the time dependent kinetics equations can be solved for the logarithm of the power level instead of the power level. This allows the power level itself to vary over a hundred decade range remaining within the magnitude limitations of the analog computer. This advantage is not gained without added problems. These problems stem from the large number of multipliers that must be used (one multiplier for each group of delayed neutrons) and the high accuracy needed in the multipliers to obtain satisfactory results in the simulation. Another method that produces adequate precision depends on the manual rescaling of the problem or the subdividing of the original problem into a number of problems each covering two or three decades of the original problem. (5)

In this study the method implemented involves normalizing the neutron flux which is equivalent to simulating the operation of the nuclear reactor from three or four decades below maximum power level up to maximum power level.

**The Analog Implementation**

This simulation was adapted from a study performed by EAI (Electronic Associates, Inc., Red Bank, N.J.) on the primary loop of a nuclear power plant. (6) In the EAI study the reactor neutron kinetics was simulated by a passive feedback network. The transport delay of the coolant flow was simulated by a piece of electronic equipment called
a capacitor wheel. The scram system consisted simply of a manual switch which was required by the need for a human operator to perform the function of quickly shutting down the simulation of a nuclear reactor in operation. The adaptation for this work consisted of transforming the analog program from dual EAI Model TR-10's to a model EAI-680 along with transforming the passive network for simulation of the nuclear reactor kinetics, the transport delay of the primary coolant loop, and the scram system, all to the equivalent conventional analog patching.

The simulator will reproduce the behavior of the primary loop of a large pressurized water nuclear reactor (PWR) shown in Figure 2-1. In this adaptation, the primary loop is considered to be operating initially under steady-state conditions at one-half of its maximum power. Typical responses studied in this simulation are the response of the reactor to a step change in reactivity; the response of the control system to power demand changes; the reactor response to control system failure during power demand changes; and the response of the reactor when scram rods (large negative reactivity) are inserted into the core.

There are a number of interacting physical systems represented by the simulation:

1) The reactor uses $^{235}\text{U}$ fuel elements as the source of energy (heat) from the fission process.

2) Heat transfer within the fuel element is caused by the temperature difference within the element.
Figure 2-1
Schematic of PWR Primary Loop

1. Outlet Plenum
2. Primary Coolant Pump
3. Heat Exchanger
4. Inlet Plenum
5. Reactor Core
6. Control Rods
7. Control Rod Drive Unit
8. Cascade Controller
9. Analog Error Signal
10. Average Temperature Across Heat Exchange
11. Valve to Maintain Constant Pressure to Turbine
12. Turbine
13. Steam Temperature
14. Digital Computer
3) The fuel element to coolant heat transfer is due to temperature differences between them. This results in the partial extraction of fission energy from the core with later application in the generation of electricity.

4) The pressurized fluid (coolant) leaves the core through the outlet mixing chamber and is transported to the primary heat exchanger (steam generator).

5) The energy is transferred from the primary loop to the secondary loop by temperature differences in the steam generator.

6) Although not simulated, flow through the secondary loop would drive a turbine which in turn would drive a generator.

7) The coolant in the primary loop after exit from the steam generator is pumped back to the inlet mixing chamber.

8) The coolant reenters the core from the mixing chamber and flows over the fuel elements removing the heat and thus completing the coolant loop.

9) The error signal feeding the control system is generated by the difference between the average temperature of the primary loop coolant existing in the steam generator and the setpoint temperature (the temperature at which the steam generator should be operating).

10) The error signal is operated on by a cascade controller to generate the control rod position which in turn determines the reactivity which through the reactor kinetics equation dictates the neutron level. The cascade controller takes the error signal and operates on it with a PI (proportional,
Reactor Kinetics:
\[
\frac{dn(t)}{dt} = \frac{\beta n(t)}{1^*} - \frac{\beta n(t)}{1^*} + \sum_{i=1}^{2} \lambda_i C_i
\]

\[
\frac{dC_i}{dt} = \frac{\beta_i n(t)}{1^*} - \lambda_i C_i
\]

Fuel Element Heat Transfer:
\[
\frac{dT_f}{dt} = \frac{n \Delta H_f}{M_f c_f} - \frac{\omega U_A}{M_f c_f} T_f + \frac{U_A}{M_f c_f} T_c
\]

Fuel Element to Coolant Heat Transfer:
\[
\frac{dT_c}{dt} = \frac{U_A}{M_c c_c} (T_f - T_c) - \frac{2W_c}{M_c} (T_c - T_{ic})
\]

Out of Reactor Core:
\[
T_{oc} = 2T_c - T_{ic}
\]

Outlet Plenum:
\[
\frac{dT_o}{dt} = \frac{W_c}{M_o} (T_{oc} - T_o)
\]

Piping Delay to Steam Generator:
\[
T_{ix} = T_o (t-D)
\]

Steam Generator:
\[
\frac{dT_x}{dt} = \frac{W_c}{M_x} (T_{ix} - T_{ox}) - \frac{U_A}{M_x c_x} (T_{ix} - T_s)
\]

Out of Steam Generator:
\[
T_{ox} = 2T_x - T_{ix}
\]

Figure 2-2.1
Heat Generation and Coolant Transfer Loop Equations
Piping Delay to Inlet Plenum

\[ T_{ix} = T_{ox} (t-D) \]

Inlet Plenum

\[ \frac{dT_{ic}}{dt} = \frac{W_c}{M_i} (T_i - T_{ic}) \]

---

Figure 2-2.2

Heat Generation and Coolant Transfer Loop Equations
Average Temperature:

\[ T_{\text{avg}} = 0.5(T_{\text{ox}} + T_{\text{ix}}) \]

Error Signal:

\[ \epsilon(t) = T_{\text{ref}} - T_{\text{avg}} \]

PI Controller:

\[ \frac{d^2 u(t)}{dt^2} = \frac{1}{\tau_m} \frac{du(t)}{dt} = \frac{K_m}{\tau_c} \left( \frac{n_o - n}{n} \right) \]

Reactivity:

\[ \delta k = \delta k_p + \delta k_f + \delta k_c + \delta k_t \]

\[ \delta k_t = \alpha(T_f - T_o) \]

Figure 2-3
Control Systems Equations
Reactor Kinetics:

\[ \frac{d}{dt} \left[ 10n^* \right] = (0.5) [20000 \text{sk}] \left[ 10n^* \right] - 10(0.64) [10n^*] + 10 \sum_{i=1}^{2} \lambda_i c_i^* \]

\[ \frac{d}{dt} \left[ \frac{c_i}{10} \right] = \frac{1}{10^2} \left[ \frac{\beta_i}{1} \right] [10n^*] - \lambda_i \left[ \frac{c_i}{10} \right] \]

Fuel Element Heat Transfer:

\[ \frac{d}{dt} \left[ \frac{T_f}{200} \right] = 0.1 \left( \frac{\Delta H_f}{200 M_f c_f} \right) [10n^*] - \left( \frac{UA}{M_f c_f} \right) \left[ \frac{T_f}{200} \right] + 0.1 \left( \frac{5UA}{M_f c_f} \right) \left[ \frac{T_c}{100} \right] \]

Fuel Element to Coolant Heat Transfer:

\[ \frac{d}{dt} \left[ \frac{T_c}{100} \right] = \left( \frac{2UA}{M_f c_c} \right) \left[ \frac{T_f}{200} \right] - \left( \frac{UA + 2W c_c}{M_f c_c} \right) \left[ \frac{T_c}{100} \right] + \left( \frac{2W c_c}{M_f c_c} \right) T_{ic} \]

Out of Reactor Core:

\[ [\frac{T_{oc}}{100}] = 10(0.2) [\frac{T_c}{100}] - [\frac{T_{ic}}{100}] \]

Outlet Plenum:

\[ \frac{d}{dt} \left[ \frac{T_o}{100} \right] = \left( \frac{W}{M_o} \right) \left[ \frac{T_{oc}}{100} \right] - \left( \frac{W}{M_o} \right) \left[ \frac{T_o}{100} \right] \]

Piping Delay to Steam Generator:

\[ [\frac{T_{ix}}{100}] = \left[ \frac{T_o}{100} \right] (t-D) \]

Steam Generator:

\[ \frac{d}{dt} \left[ \frac{T_x}{100} \right] = \left( \frac{2W}{M_x} \right) \left[ \frac{T_{ix}}{100} \right] + \left( \frac{UA x}{M_x c_c} \right) \left[ \frac{T_s}{100} \right] - \left( \frac{UA x}{M_x c_c} \right) \left[ \frac{T_x}{100} \right] - \left( \frac{2W}{M_x} \right) \left[ \frac{T_x}{100} \right] \]

**Figure 2-4:1**

Analog Magnitude Scaled Heat Generation and Coolant Transfer Loop Equations
Out of Steam Generator:

\[
\frac{T_{100}^{ox}}{100} = 10(0.2) \left[ \frac{T_{100}^x}{100} \right] - \left[ \frac{T_{100}^{ix}}{100} \right]
\]

Piping Delay to Inlet Plenum:

\[
\left[ \frac{T_{100}^{ix}}{100} \right] = \left[ \frac{T_{100}^{ox}}{100} \right] (t-D)
\]

Inlet Plenum:

\[
\frac{d}{dt} \left[ \frac{T_{100}^{ic}}{100} \right] = \frac{W_c}{M_1} \left[ \frac{T_{100}^{i}}{100} \right] - \frac{W_c}{M_1} \left[ \frac{T_{100}^{ic}}{100} \right].
\]

---

**Figure 2-4.2**

Analog Magnitude Scaled Heat Generation and Coolant Transfer Loop Equations
Average Temperature:
\[
\left[ \frac{T_{\text{avg}}}{100} \right] = 0.5 \left( \left[ \frac{T_{\text{ox}}}{100} \right] + \left[ \frac{T_{\text{ix}}}{100} \right] \right)
\]

Error Signal:
\[
\left[ \frac{\varepsilon}{100} \right] = [10] \left( \frac{T_{\text{ref}}}{1000} \right) - \left[ \frac{T_{\text{avg}}}{100} \right]
\]

Cascade Controller:
\[
\left[ 2(n_0-n^*) \right] = (2 \times 10^3 K_c) \int_0^t \frac{1}{10} \left[ \frac{\varepsilon}{100} \right] dt + (2000 K_c \tau_c) \left[ \frac{\varepsilon}{100} \right]
\]
\[-0.200[10^*] + 2n_o(0)
\]

Control Rod Drive Unit:
\[
\frac{d^2}{dt^2} \left[ 2000\mu \right] = \left( \frac{1}{\tau_m} \right) \frac{d}{dt} \left[ 2000\mu \right] = \left( \frac{K_M 10^3}{\tau_m} \right) \frac{[20(n_0-n^*)]}{[10^*]}
\]

Reactivity:
\[
\left[ 2000\delta k \right] = 10(20k)[10] - 10(4 \times 10^4|\alpha|) \left[ \frac{T_f}{200} \right] + \left[ 2000\mu \right]
\]
\[k = \delta k_f + \delta k_c(0) + \delta k_p - \alpha T_o\]

Figure 2-5
Analog Magnitude Scaled Control Systems Equations
Figure 2-6

Reactor Scram Response
CHAPTER III

CONTROL COMPUTER CONCEPTS

Information Flow

Three types of flow can be considered in process analysis: flow of material, flow of energy, and flow of information. The flow of information is enhanced by the incorporation of control computers in the scheme. Information flow is essential to control for without information in its many forms the control function (the ability to accomplish a desired end) is effectively lost. A digital computer control system is a tool which may be applied in the area of information flow. The digital computer has the ability to quickly acquire, assimilate, analyze, and disseminate large amounts of information with great speed, accuracy, and flexibility.

Methods of Computer Implementation

The methods of computer implementation of control which are of interest are: 1) off-line, and 2) on-line. Each can be further divided into: a) open-loop, and b) closed-loop. The terms off-line and on-line refer to the method by which process data is entered into the computer. The terms open-loop and closed-loop refer to the method by which the feedback signal that is calculated by the computer is passed to the process.

The off-line computer receives information about the process from a human intermediary and the resulting calculated control actions are applied to the process through the human operator (open-loop mode). In
the closed-loop mode, the computer applies the control action to the process through an electronic interface that converts the signals into settings for the controlled elements in the process.

As the intricacies of the process unfold, the potential of information flow in the process is enhanced by on-line computers. The term "on-line" refers to the ability of the computer to accept signals directly from process instruments and to convert them into a form suitable for computer processing. The on-line ability greatly reduces the data accumulation time. An on-line computer could operate in the open-loop mode, again referring to the fact that the resultant calculated actions are applied to the process through a human operator. In on-line, closed-loop computer control, the computer receives information strictly from the process and its calculated control actions are applied directly to the process through suitable instruments. In this last mode the faster information flow results in the least delay time from the time something happens in the process to the time the process receives a resulting control action. (7)

Software Requirements

In the past in computer control implementation, there have been many instances of underestimation of the cost of user-written software. There are several reasons for this:

1) Since every process is unique there will be a considerable amount of effort expended initially in developing the custom software required by each process. Cost for the effort depends upon many things, one of which is the previous experience of the personnel on the project.
2) The software was often not designed to be easily expandable. The once-and-for-all concept was used with little or no thought for future expansion.

The development of the philosophy and general operating characteristics of a complete set of real-time programs is an extremely important task. The success of the entire project may hinge upon successful software design and development. To some degree the structure of the software system differs between DDC (Direct Digital Control) and Supervisory Control. In DDC, the tasks to be performed are usually simple, but must be performed at frequent intervals. These tasks are usually very similar. In contrast, in Supervisory Control the tasks are usually longer and more complex, but the system is usually composed of fewer tasks. Supervisory Control can be thought of as setpoint control. The computer outputs the setpoint to the controlled device (analog controller) which in turn maintains the controlled variable at setpoint conditions by manipulating a control element. In DDC the computer outputs (position, etc.) directly to the control element, thereby replacing the function performed by the analog controller should the controlled variable experience a disturbance. While the demands on the software packages differ between DDC and Supervisory Control, both must operate in real time.

In either Supervisory Control or DDC, the monitor or executive system can be divided into three parts:

1) Interrupt Servicing,
2) Cyclic Program Servicing,
3) Free-time Servicing.
The basis for this structure revolves about the idea that the accomplishment of some tasks are more important than the accomplishment of others. Therefore the tasks that are more important are assigned a higher priority than the less important tasks. Depending upon this pre-assigned priority, the tasks are assigned to initiators or interrupt levels within the computer.

**Interrupt Servicing**

A process control computer may have any number of levels of interrupts. Each level has associated with it a priority. When the computer is operating and an interrupt request occurs that has a higher priority than the current task which the computer is executing, the current task is interrupted with the contents of all registers saved and the higher priority task enters execution. When the higher priority task has been completed the interrupted task registers are restored and execution continues from the point of interruption. If an interrupt request occurs that has lower priority than the current task which the computer is executing, there is no interruption of the higher priority task to service the lower priority interrupt request. Instead when all higher priority requests have been serviced, it will be serviced. An interrupt request of the same priority level of a current task will not interrupt the current task but will be serviced later.

To illustrate the operation of interrupts refer to Figure 3-1. The computer is in the idle state at the start of this example, which is graphically represented at the top of the time line. As time passes, one moves down the time line until Interrupt 1 occurs. The computer responds by servicing this interrupt since none of higher priority are
<table>
<thead>
<tr>
<th>Occurrence of Events</th>
<th>Time Computer Line</th>
<th>Computer Interrupt Servicing</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Idle</td>
<td>#3</td>
</tr>
<tr>
<td>Interrupt 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Begin Task</td>
</tr>
<tr>
<td>Interrupt 2</td>
<td></td>
<td>End Task</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Begin Task</td>
</tr>
<tr>
<td>Interrupt 3</td>
<td></td>
<td>End Task on 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Begin Task on 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>End Task</td>
</tr>
<tr>
<td>Interrupt 2</td>
<td></td>
<td>Begin Task on 2</td>
</tr>
<tr>
<td>Interrupt 1</td>
<td></td>
<td>Halt Task on 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Begin Task on 1</td>
</tr>
<tr>
<td>Interrupt 3</td>
<td></td>
<td>End Task on 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Return to Task on 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Finish Task on 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Begin Task on 3</td>
</tr>
<tr>
<td>Interrupt 1</td>
<td></td>
<td>Halt Task on 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Begin Task on 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>End Task on 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Return to Task on 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>End Task</td>
</tr>
</tbody>
</table>

Figure 3-1
Interrupt Operation
utilizing the computer. When this servicing is completed, since no other interrupts of lower priority require servicing, the computer is returned to the idle state. As time passes, Interrupt 2 occurs. Since no interrupts of higher priority are active, the computer begins servicing Interrupt 2. Before the computer has finished servicing Interrupt 2, Interrupt 3 occurs. Interrupt 3 does not receive control of the computer now since an interrupt of higher priority has control. When Interrupt 2 is completed the computer gives control to level 3 since no interrupt of higher priority is active and Interrupt 3 is waiting to be serviced. Since no other interrupts occur during the servicing of Interrupt 3, the level is not interrupted and when service is completed, the computer is returned to the idle state. Later Interrupt 2 occurs again. Since the computer is in the idle state, there are no high priority interrupts active and the computer gives control to Interrupt 2. Before level 2 can finish, Interrupt 1 occurs. The computer saves status on level 2 and begins servicing Interrupt 1 since it is the highest priority active in the computer. Interrupt 3 occurs during the servicing of Interrupt 1, but since level 3 is lower than level 1 or level 2, it will have to wait until the completion of the servicing of levels 1 and 2. Interrupt 1 has now been completed and the computer proceeds to give control to Interrupt 2, since it is waiting and now has the highest priority. When the computer finishes servicing level 2 it will find that level 3 now has the highest priority and gives control to Interrupt 3. During the servicing of level 3, Interrupt 1 occurs and the computer gives control to level 1 putting level 3 in the wait state. When level 1 is completed, level 3 is restored and continues servicing from the point at which Interrupt 1 had occurred. When level 3 is
completed there are no interrupts active and the computer again returns to the idle state. (8)

In general, the assignment of interrupts to their associated tasks is not a simple matter. In most cases, monitor interrupts have the highest priority, level 1. The next level, level 2, may be assigned to disastrous plant alarms. The groups that would be attached at this level are equipment shutdown tasks. This level could also include some type of logging function which after an emergency condition could, upon request, produce the record of events during the emergency condition. With this information the operator should be able to decide what caused the condition. On level 3, there could be some type of timing function or clock task which would be the scheduler for programs that need to run on a cyclic base. The different control programs would be attached to different interrupts within level 3. When the clock task detects that it is time for one of the tasks to run, the clock task turns on the interrupt associated with the tasks that need servicing. On level 4, there could be a task that allows the human operator to communicate with the control computer. The usual name for this task is operators console. This interface relationship between the human and the computer is very important. The function of operators console is to allow the operator to use the computer to expand the control capability.
CHAPTER IV

DIGITAL DATA ACQUISITION AND CONTROL PACKAGE

The Function of the Package

This package performs the necessary digital data acquisition and data manipulations required to accomplish a set of control tasks. The package is connected to an interface through which process variables are communicated. The process in this study was simulated on an analog computer. The process need not be simulated should the real process be accessible to the computer. The interface communicates signals between -10 and +10 volts. The analog computer operates with signals in the range of -10 to +10 volts. Should the real process not have signals initially in this range, the signals can be transformed into the required voltage. The real process variables are then connected into the interface instead of the simulated process variables. Thus the researcher can study a simulated process or a real process with equal ease.

The Description of the Package

The package was designed with further expansion in mind. This capability for further expansion stems from its modular structure, which was designed at the outset of the project. The structure consists of several modules each composed of groups of related tasks. The basic modules required are: 1) time module, 2) process-computer interface module, 3) control module, 4) human-computer interface module and 5) intra-structure communication module; refer to Figure 4-1.
Timing Module

The timing module consists of a clock to keep track of elapsed time and a flag indicating the start of the next second. It has been determined that the highest external routine to be executed is the one that was last executed. The timing module keeps track of this last executed routine and executes it first. The necessary registers are set to a default value before the last executed routine is executed.

Intra-Structure Communication Module

Process-Computer Interface Module

Control Module

Human-Computer Interface Module

Figure 4-1
Structure of Hybrid Package
Timing Module

The timing module consists of a clock function for counting elapsed time and a triggering function for initiating all tasks that run on a cyclic or time initiated base. At memory location X'5A' (hexadecimal) in the Sigma 5 is a hardware feature called "counter 3 interrupt if zero". The integer number stored in this location is automatically decremented by 1 every one five-hundredth of a second. Depending upon the time frame desired, this location is initially set at a certain value (e.g. 500). After 1 second has passed the location will have counted down to zero and the internal interrupt from "counter 3 interrupt if zero" will occur. Because this is the highest priority interrupt in the computer, it will be serviced immediately. To service this interrupt, the computer first stops executing whatever program it is currently working on and stores into location X'5A' the same value as before so that the automatic count-down to zero will occur again the next second. Then the computer triggers interrupt level X'60', the highest external interrupt which is connected to the timing module routine. Therefore, instead of returning to the program which it may have been executing when the counter 3 interrupt occurred, the computer next begins executing the timing module routine after first saving the necessary registers to permit it to return to the interrupted program later. The timing module updates the time-of-day, a record it is keeping, and decrements by 1 second all the contents of locations associated with the time-to-run (ITRUN) table, which keeps track of when each task that runs on a cyclic base should be initiated. The timing module then checks to see if any of these tasks have a time-to-run equal to zero. If so the interrupt to which the task is connected is triggered by the
timing module and the task run interval is stored in the time-to-run table. None of these triggered tasks start immediately because the timing module routine is running at the highest priority external interrupt level. When it is finished with the time-to-run table, the timing module interrogates the intra-structure communication module to determine if any control programs need servicing. These control programs could need servicing because of either unusual conditions in the process detected by the process-computer interface or by request from other modules. If any need servicing, the interrupt levels associated with them are triggered. Therefore, the control programs may run on a cyclic time base and also on requests. This describes the present design of the timing module structure. The timing module is easy to add to or modify.

Process-Computer Interface Module

The process-computer interface module performs four functions and is connected to the second highest external interrupt level X'61'. These four functions are: 1) receive, 2) filter, 3) send, and 4) convert. In receive, the computer interrogates the process to determine the current values of the variables in the process which have been chosen as process inputs and therefore are patched into the Analog Digital Converters (ADC units). The digital values which the computer receives from the ADC units are normalized quantities (between -1 and +1). These normalized values must be multiplied by a scaling constant for each individual variable to obtain the value in engineering units. These scaling constants are read into a table as data when the package is first loaded into the computer. Thus the values in engineering
units, E.U., have dimensions, whereas the values initially read from
the process as normalized variables are undimensioned. The E.U.
values next could be digitally filtered to reduce the effect of any
random error or "noise" in the readings from the process. This filtering
function was not implemented in this study since no need for it
was detected. Later applications involving readings from actual
instruments on pilot plant equipment will likely require filtering.
The position to insert filtering has been denoted in the program list-
ing should the need develop. The filtered E.U. values are stored in a
table in the intra-structure communication module for access by other
modules. Thus there exists in memory at all times while the package
is in operation a table of the most recent E.U. values of the process
inputs. The process-computer interface next checks to see if any other
modules have entered any values to be sent to the process (setpoints,
etc.). Should there be outputs to be sent to the process, the process-
computer interface normalizes the E.U. values of the outputs then sends
them to the process via the Digital to Analog Converters (DAC units).
At this point the module has completed its function for this cycle.

Control Module

The control module has slots for eight control programs, one of
which was implemented in this study. Expansion of the number of control
slots beyond eight could be easily implemented. Each control program
has associated with it a run interval since control programs normally
run on a cyclic base. If the control program has been previously turned
on, the time-to-run for the control program (ITRUNC) is being decre-
mented each cycle by the clock module. The control module merely
interrogates the time-to-run of each control program and executes those that have zero entries. Thus the control module itself runs on the shortest cycle when initiated by the timing module, and each control program is called at its appropriate time by the control module. All run interval values are stored in the intra-structure communication module and they can be changed with the human-computer interface module. When a control program needs to output information back to the process, it stores the value in the intra-structure communication module and sets a flag that indicates the value is new and awaiting transmission to the process. The next time the process-computer interface module runs, the set flag is detected and the value is sent to the process. The control module is connected to both external interrupt level X'62' and X'63'. Should a control program be requested to run by another control program or by any module, interrupt level X'62' will be activated. If the control is requested by the timing module, interrupt X'63' will be activated. This allows control programs with a higher need to run than the normal cyclic operation to cut in line. Control programs also have built-in priorities with respect to each other depending upon the order in which they are interrogated by the control module.

**Human-Computer Interface Module**

The human-computer interface is located at the lowest priority in the interrupt structure because of the relatively long response time of a human operator. The computer can respond to an operator request via the human-computer interface module and still perform all the higher priority tasks described above, normally without the human operator detecting a lull in the transmissions. Were this module given higher
priority, the human operator-computer communication would not increase
noticeably, but the computer would spend excessive time in the idle
state since it cannot service lower level interrupts until higher levels
are completed. Consequently, many of its tasks would simply not be
executed. Instead, the priority structure assigning the human-computer
interface module the lowest priority results in the best overall per-
formance. The human-computer module is termed the "operators console
program" by most users in industry. The operators console program
should be broad enough in scope so that the human operators abilities to
interpret and run the plant are improved, not hindered. To accomplish
this task effectively, the module performs two basic functions. The
first gives the human operator the capability of changing values stored
in memory locations with the computer, and the second gives him the
capability of looking at listings of values stored in the computer. The
human operator would be required to remember the locations of the values
and exactly what the value had to be to accomplish the desired action if
these two module functions were not expanded. In the expanded form
implemented in this study, instead of only one general purpose memory
change function and one general purpose memory printout function, there
are a number of specific change and print functions designed to change
or print out specific types of quantities. Each function is requested
by the operator by entering three codes into the computer. The first
code, IFUNCOD (Function Number), identifies the function. The second
code, IDPT (ID point), identifies a sub-element within the specific
function and the third code, VALUE, when applicable designates the new
value to be entered into memory. When a human operator attempts to
change any value by more than five percent, the operator console program
requires a verification from him before the change is affected. The operators console functions implemented in this work are listed in Figure 4-2. The values that the operators console program can change or print out are stored in the intra-structure communication module.

**Intra-Structure Communication Module**

A Fortran programmer would recognize the intra-structure communication module as "Common". Common is a storage area that can be used by the different modules, but it is not located within the modules. In this study, values that are parameters are stored in common in the CT (control table). These parameters can be changed or printed out by an operators console function. The values that are variables, i.e., values that are either read from the process or generated by a module, are stored in common in the VT (variable table). These values cannot be changed directly by the human operator, but they can be printed out. Typical CT and VT constants are listed in Figure 4-3 and 4-4 along with the meaning of each element.

**The Use of the Package**

The process if simulated is patched on the analog board. The inputs and outputs to the simulated or real pilot plant are selected and patched into the interface section on the analog board (ADC units and DAC units). With knowledge of the inputs and outputs, the parameter cards are punched and added to the program deck. The program is then loaded into the computer. The setup of the deck will not be discussed here since appropriate comment cards are included within the listing of the program deck in the appendix. The parameter cards which follow the
program deck are also shown in the listing in the appendix. They are described by comment cards in the listing and their formats are defined by format statements in the program.
<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>IDPT</th>
<th>VALUE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Idpt</td>
<td>Value</td>
<td>Update of Constants in Control Table (Idpt) to (Value)</td>
</tr>
<tr>
<td>02</td>
<td>Idpt</td>
<td>Value</td>
<td>Update of Task Run Interval (Idpt) to (Value)</td>
</tr>
<tr>
<td>03</td>
<td>Idpt</td>
<td>Value</td>
<td>Update of Alarm Lower Limit (Idpt) to (Value)</td>
</tr>
<tr>
<td>04</td>
<td>Idpt</td>
<td>Value</td>
<td>Update of Alarm Upper Limit (Idpt) to (Value)</td>
</tr>
<tr>
<td>05</td>
<td>Void</td>
<td>Void</td>
<td>List on Logging Device Alarm Limits</td>
</tr>
<tr>
<td>06</td>
<td>Idpt</td>
<td>Void</td>
<td>Change Name of Analog Input (Idpt)</td>
</tr>
<tr>
<td>07</td>
<td>Void</td>
<td>Void</td>
<td>List on Logging Device Names of Analog Inputs</td>
</tr>
<tr>
<td>08</td>
<td>Idpt</td>
<td>Void</td>
<td>Start Trending Log, Idpt = Number of Variables to be Trended</td>
</tr>
<tr>
<td>09</td>
<td>Void</td>
<td>Void</td>
<td>Stop Trending Log</td>
</tr>
<tr>
<td>10</td>
<td>Void</td>
<td>Void</td>
<td>Add One Variable to Trend Log (Max.9)</td>
</tr>
<tr>
<td>11</td>
<td>Idpt</td>
<td>Void</td>
<td>Delete One Variable From the Trending Log, Idpt = the Number of the Variable Deleted</td>
</tr>
<tr>
<td>12</td>
<td>Void</td>
<td>Void</td>
<td>To Give Clock Correct Time of Day</td>
</tr>
<tr>
<td>13</td>
<td>Idpt</td>
<td>Void</td>
<td>To Turn on a Control Program (Idpt)</td>
</tr>
<tr>
<td>14</td>
<td>Idpt</td>
<td>Void</td>
<td>To Turn Off a Control Program (Idpt)</td>
</tr>
<tr>
<td>15</td>
<td>Void</td>
<td>Void</td>
<td>To List on Logging Device Status of Control Programs</td>
</tr>
<tr>
<td>16</td>
<td>Idpt</td>
<td>Void</td>
<td>To Allow Control Program to Print Messages</td>
</tr>
<tr>
<td>17</td>
<td>Idpt</td>
<td>Void</td>
<td>To Turn Off Messages From Control Program</td>
</tr>
</tbody>
</table>

Figure 4-2

Operator Console Functions
## CONTROL TABLE

<table>
<thead>
<tr>
<th>IDPT</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>41</td>
<td>1.0</td>
</tr>
<tr>
<td>42</td>
<td>20000.0</td>
</tr>
<tr>
<td>43</td>
<td>1000.0</td>
</tr>
<tr>
<td>44</td>
<td>1000.0</td>
</tr>
<tr>
<td>45</td>
<td>1000.0</td>
</tr>
<tr>
<td>46</td>
<td>1000.0</td>
</tr>
<tr>
<td>47</td>
<td>1000.0</td>
</tr>
<tr>
<td>48</td>
<td>1000.0</td>
</tr>
<tr>
<td>49</td>
<td>1000.0</td>
</tr>
<tr>
<td>50</td>
<td>1000.0</td>
</tr>
<tr>
<td>51</td>
<td>1.0</td>
</tr>
<tr>
<td>52</td>
<td>10000.0</td>
</tr>
<tr>
<td>53</td>
<td>1000.0</td>
</tr>
<tr>
<td>54</td>
<td>1000.0</td>
</tr>
<tr>
<td>55</td>
<td>1.0</td>
</tr>
<tr>
<td>56</td>
<td>1.0</td>
</tr>
<tr>
<td>57</td>
<td>0.005</td>
</tr>
<tr>
<td>58</td>
<td>1000.0</td>
</tr>
<tr>
<td>59</td>
<td>1.0</td>
</tr>
<tr>
<td>60</td>
<td>0.0</td>
</tr>
<tr>
<td>61</td>
<td>0.0</td>
</tr>
<tr>
<td>62</td>
<td>0.0</td>
</tr>
<tr>
<td>63</td>
<td>0.0</td>
</tr>
<tr>
<td>64</td>
<td>0.0</td>
</tr>
<tr>
<td>65</td>
<td>0.0</td>
</tr>
<tr>
<td>66</td>
<td>0.0</td>
</tr>
<tr>
<td>67</td>
<td>0.0</td>
</tr>
<tr>
<td>68</td>
<td>0.0</td>
</tr>
<tr>
<td>69</td>
<td>0.0</td>
</tr>
<tr>
<td>70</td>
<td>0.0</td>
</tr>
<tr>
<td>71</td>
<td>400.0</td>
</tr>
<tr>
<td>72</td>
<td>0.1</td>
</tr>
<tr>
<td>73</td>
<td>0.05</td>
</tr>
<tr>
<td>74</td>
<td>0.0</td>
</tr>
</tbody>
</table>

### DESCRIPTION

- **Maximum Value Neutron Flux**
- **Avg. Fuel Temp. F**
- **Temp. Out of Core F**
- **Mixing F**
- **in Exch. - F**
- **Avg. Temp. Exch. F**
- **Cool in Core F**
- **Temp. Out Exch. - F**
- **Inlet Mixing F**
- **T Inlet Core F**
- **Control Rod Position**
- **Steam Temp. - F**
- **Temp. Avg. Compar. F**
- **Error Temp. F**
- **Del K**
- **Temp. Ref. F**
- **Setpoint Controller F**
- **Digital Load Change**

Position of Valve on Heat Exchanger

Temp. Steam Target F
KP For CONTR1
KI For CONTR1

Figure 4-3.1

Control Table
## CONTROL TABLE

<table>
<thead>
<tr>
<th>IDPT</th>
<th>VALUE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>41</td>
<td>60.0</td>
<td>CONTR1 Run Interval Seconds</td>
</tr>
<tr>
<td>42</td>
<td>60.0</td>
<td>CONTR2 &quot; &quot; &quot; &quot;</td>
</tr>
<tr>
<td>43</td>
<td>60.0</td>
<td>CONTR3 &quot; &quot; &quot; &quot;</td>
</tr>
<tr>
<td>44</td>
<td>60.0</td>
<td>CONTR4 &quot; &quot; &quot; &quot;</td>
</tr>
<tr>
<td>45</td>
<td>300.0</td>
<td>CONTR5 &quot; &quot; &quot; &quot;</td>
</tr>
<tr>
<td>46</td>
<td>300.0</td>
<td>CONTR6 &quot; &quot; &quot; &quot;</td>
</tr>
<tr>
<td>47</td>
<td>300.0</td>
<td>CONTR7 &quot; &quot; &quot; &quot;</td>
</tr>
<tr>
<td>48</td>
<td>300.0</td>
<td>CONTR8 &quot; &quot; &quot; &quot;</td>
</tr>
<tr>
<td>49</td>
<td>0.0</td>
<td>Flag to Print Cont. Prog. 1 Run</td>
</tr>
<tr>
<td>50</td>
<td>0.0</td>
<td>&quot; &quot; &quot; &quot; &quot; &quot; 2 &quot;</td>
</tr>
<tr>
<td>51</td>
<td>1.0</td>
<td>&quot; &quot; &quot; &quot; &quot; &quot; 3 &quot;</td>
</tr>
<tr>
<td>52</td>
<td>1.0</td>
<td>&quot; &quot; &quot; &quot; &quot; &quot; 4 &quot;</td>
</tr>
<tr>
<td>53</td>
<td>1.0</td>
<td>&quot; &quot; &quot; &quot; &quot; &quot; 5 &quot;</td>
</tr>
<tr>
<td>54</td>
<td>1.0</td>
<td>&quot; &quot; &quot; &quot; &quot; &quot; 6 &quot;</td>
</tr>
<tr>
<td>55</td>
<td>1.0</td>
<td>&quot; &quot; &quot; &quot; &quot; &quot; 7 &quot;</td>
</tr>
<tr>
<td>56</td>
<td>1.0</td>
<td>&quot; &quot; &quot; &quot; &quot; &quot; 8 &quot;</td>
</tr>
<tr>
<td>57</td>
<td>1.0</td>
<td>&quot; &quot; &quot; &quot; &quot; &quot; 9 &quot;</td>
</tr>
<tr>
<td>58</td>
<td>1.0</td>
<td>&quot; &quot; &quot; &quot; &quot; &quot; 10 &quot;</td>
</tr>
<tr>
<td>59</td>
<td>0.0</td>
<td>&quot; &quot; &quot; &quot; &quot; &quot; 11 &quot;</td>
</tr>
<tr>
<td>60</td>
<td>400.0</td>
<td>Temp. Setpoint F</td>
</tr>
</tbody>
</table>

**Figure 4-3.2**

Control Table
<table>
<thead>
<tr>
<th>IDPT</th>
<th>VALUE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Current Values</td>
<td>Current Value(EU) of Variable Described CT(IDPT)</td>
</tr>
<tr>
<td>1</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>2</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>3</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>4</td>
<td>The capacity of the heat exchanger is expressed in terms of its surface</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>6</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>7</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>8</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>9</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>10</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>11</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>12</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>13</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>14</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>15</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>16</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>17</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>18</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>19</td>
<td>Available for future use</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>21</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>22</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>23</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>24</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>25</td>
<td>&quot; &quot;</td>
<td>&quot; &quot;</td>
</tr>
<tr>
<td>26</td>
<td>Current Values</td>
<td>Position of Valve on Heat Exchanger</td>
</tr>
<tr>
<td>27</td>
<td>Available for future use</td>
<td></td>
</tr>
</tbody>
</table>

Figure 4-4

Variable Table
CHAPTER V

ACTION OF A DIGITAL SUPERVISORY CONTROL PROGRAM

Digital Control Program

The digital control program implemented in this study demonstrates the utility of the hybrid package for making studies of various proposed digitally controlled systems. The cascaded control system included in the analog simulation of the PWR is not capable of maintaining constant steam temperature when a load disturbance occurs in the system. With the addition of a supervisory digital control program to maintain constant steam temperature the total system adjusts automatically to load disturbances and returns the steam temperature to its desired value. The objective of maintaining constant steam temperature stems from the characteristics of turbines. Turbines perform better if they are operated at constant pressure. Since the steam in this model is considered to be saturated, constant pressure means constant temperature. Therefore the objective of better performance of the steam turbines is met by maintaining a constant temperature steam flow leaving the heat exchanger.

Each time it runs, the digital control program uses the velocity form of the PI control algorithm to compute an adjustment to the manipulated variable (temperature setpoint for the analog cascaded PI controller) based on the error in the controlled variable (desired steam temperature — actual steam temperature). The velocity algorithm
can be expressed as:
\[ M = K_p(EN - E_0) + K_i(EN) \]

where
- \( M \) = Change in the manipulated variable
- \( K_p \) = Proportional constant
- \( K_i \) = Integral constant
- \( EN \) = Error now
- \( E_0 \) = Error at previous time increment

There was no attempt made in this study to develop optimal values for the two digital controller settings, the proportional constant and the integral constant.

Response of Process to Digital Supervisory Control

The above described digital supervisory control program was written and inserted into the digital package to illustrate the application of the digital control package to control an analog simulated process. As explained above, the digital control program, when turned on by the human operator through an operator console function, adjusts the setpoint of the analog cascade controller to maintain a constant temperature of the steam generated in the boiler. The graphs in Figure 5-1 to 5-4 represent the responses produced by the hybrid computer package to a ten percent load change. The load was varied by adjusting a parameter in the analog simulation representing the opening of a valve in the steam line to the turbine. The plotted information was obtained while the simulated process was running via the operators console function. Different responses were obtained by changing the digital controller proportional and integral constants. Values of the controller constants are listed on the individual graphs.
Figure 5-1 illustrates the response without the digital control program in operation. The steam temperature does not return to its original position before load change. Figure 5-2 illustrates the response with only proportional action. In this case the steam temperature does not have as much steady-state off-set from the set-point as in the case without any control. Figure 5-3 and 5-4 illustrate the response with both proportional and integral action. There is no permanent off-set if integral action is employed. Different values of integral action produce differences in the periodic time constant and the stability of the system. Increased integral action decreases the response time of the system, but at the same time decreases the stability of the system.
Figure 5-1
System Response
to Step Change in Steam Valve Opening
with no Supervisory Control
Figure 5-2
System Response
to Step Change in Steam Valve Opening
with only Proportional Control
Steam Temperature (°F)

Figure 5-3
System Response

to Step Change in Steam Valve Opening

with Proportional and Integral Control
CHAPTER VI

CONCLUSIONS AND RECOMMENDED FURTHER WORK

A hybrid system can be used as an excellent model for the analysis of many aspects of reaction time. The time delay can be studied by applying
an appropriate input function to the system and observing the
response. The system can be represented by a set of differential
equations that can be solved numerically.

For example, if the system is simulated on a digital computer, it can be
implemented on a suitable physical system. The use of the
computer can lead to more accurate and reliable results.

The design of such systems can be facilitated by the availability of
components such as microprocessors. The use of digital control systems
including steam temperature controllers, temperature sensors, and
digital controllers can be particularly effective.

Steam Temperature (°F)

Figure 5-4

System Response
to Step Change in Steam Valve Opening
with Proportional and Integral Control
CHAPTER VI

CONCLUSIONS AND RECOMMENDED FUTURE TOPICS

Conclusions

A hybrid computer package has been developed allowing the study
of many aspects of various physical time varying systems. The variety
of physical systems that can be studied are limited by only two con-
straints. The system must either be communicable in a mathematical
sense such that a mathematical model of it (simulation) can be patched
on the analog portion of the hybrid, or be actual physical processes
that can communicate with the hybrid.

For demonstration of the package, a nuclear power reactor was
simulated on the analog section with a digital process control computer
implemented on the digital section of the hybrid computer. The digital
control objective was to maintain a certain output (steam temperature)
of the analog simulation at a specific value.

The demonstration system was operated successfully, and all
components of the hybrid package were shown to function as designed,
including Scan, Alarm, Trend Logging, Operators Console Functions, and
Digital Control Programs.

Recommended Future Topics

The hybrid package could be expanded in various ways to yield an
even more powerful research tool.

The capability of maintaining in the computer's auxiliary memory a
historical record recallable on demand (e.g. 24 hours) of all the data
being generated would produce for the researcher a more complete
picture of what was occurring.

At present when the package is running both the analog and digital machines are dedicated to the hybrid package. The digital section of the hybrid package could be further developed so as to time-share the digital computer with other digital programs. The use of the hybrid computer to control either simulated or analog processes would not interfere greatly with normal batch-type use of the digital computer.

For any specific study, additional control programs could be developed relying on more advanced control techniques. Two such techniques are Feedforward Control and Adaptive Control. Feedforward control generates control actions based upon measurement of inputs to the process instead of the controlled variable. Should the inputs change, a feedforward control program computes the corrective action needed to maintain the controlled variable at the desired value.\(^{(11)}\)

Adaptive control is defined as a system which is provided with a means of continuously monitoring its own performance in relation to a given index of performance and modifying its own parameters by closed loop action so as to approach optimal performance.\(^{(12)}\)

Should the control programs exceed the available core storage, the structure could be modified to allow the control programs to reside in auxiliary storage. The programs could then be brought into core when needed (e.g. overlay structure).
LITERATURE CITED


(9) Xerox Data Systems, 701 South Aviation Boulevard, El Segundo, California, 90245, Sigma 5 (Model).


<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>MEANING</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Heat transfer area in the reactor</td>
<td>ft²</td>
</tr>
<tr>
<td>Aₓ</td>
<td>Heat transfer area in the steam generator</td>
<td>ft²</td>
</tr>
<tr>
<td>A'</td>
<td>Cross sectional flow area</td>
<td>ft²</td>
</tr>
<tr>
<td>Cₖ</td>
<td>Specific heat of the coolant</td>
<td>Btu/lb-°F</td>
</tr>
<tr>
<td>Cₜ</td>
<td>Specific heat of the fuel, moderator, etc.</td>
<td>Btu/lb-°F</td>
</tr>
<tr>
<td>Cₗ</td>
<td>Concentration of neutrons from delayed neutron group &quot;j&quot;</td>
<td>Neu/sec</td>
</tr>
<tr>
<td>Ċ</td>
<td>Fluid specific heat</td>
<td>Btu/lb-°F</td>
</tr>
<tr>
<td>k</td>
<td>Reactivity constant</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>K</td>
<td>Control rod drive unit gain</td>
<td>ft/sec</td>
</tr>
<tr>
<td>Kₖ</td>
<td>Controller gain</td>
<td>neutrons°F-sec-cm²</td>
</tr>
<tr>
<td>Mₖ</td>
<td>Mass of coolant in the reactor</td>
<td>lbs.</td>
</tr>
<tr>
<td>Mₖᵢ</td>
<td>Mass of fuel, moderator, etc.</td>
<td>lbs.</td>
</tr>
<tr>
<td>Mᵢ</td>
<td>Mass of coolant in the inlet plenum chamber</td>
<td>lbs.</td>
</tr>
<tr>
<td>M₀</td>
<td>Mass of coolant in the outlet plenum chamber</td>
<td>lbs.</td>
</tr>
<tr>
<td>Mₙ</td>
<td>Mass of coolant in the steam generator</td>
<td>lbs.</td>
</tr>
<tr>
<td>S</td>
<td>Operator</td>
<td>Seconds⁻¹</td>
</tr>
<tr>
<td>Tₐ</td>
<td>Ambient temperature</td>
<td>°F</td>
</tr>
<tr>
<td>Tₖ</td>
<td>Average coolant temperature in the reactor</td>
<td>°F</td>
</tr>
<tr>
<td>Tₖᵢ</td>
<td>Average fuel temperature</td>
<td>°F</td>
</tr>
<tr>
<td>Tᵢ</td>
<td>Coolant temperature at the input to the reactor inlet plenum chamber</td>
<td>°F</td>
</tr>
<tr>
<td>T₀</td>
<td>Coolant temperature at the outlet plenum chamber, or temperature coefficient of reactivity reference temperature (temperature of which temperature contribution is zero).</td>
<td>°F</td>
</tr>
<tr>
<td>SYMBOL</td>
<td>MEANING</td>
<td>UNITS</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>$T_s$</td>
<td>Average temperature of secondary fluid in steam generator</td>
<td>°F</td>
</tr>
<tr>
<td>$T_x$</td>
<td>Average coolant temperature in the steam generator</td>
<td>°F</td>
</tr>
<tr>
<td>$T_{ic}$</td>
<td>Temperature of coolant entering the reactor</td>
<td>°F</td>
</tr>
<tr>
<td>$T_{ix}$</td>
<td>Temperature of coolant entering the steam generator</td>
<td>°F</td>
</tr>
<tr>
<td>$T_{oc}$</td>
<td>Temperature of coolant leaving the reactor</td>
<td>°F</td>
</tr>
<tr>
<td>$T_{ox}$</td>
<td>Temperature of coolant leaving the steam generator</td>
<td>°F</td>
</tr>
<tr>
<td>$T_{ref.}$</td>
<td>Reference temperature</td>
<td>°F</td>
</tr>
<tr>
<td>$T_{ave.}$</td>
<td>Average system temperature</td>
<td>°F</td>
</tr>
<tr>
<td>$T$</td>
<td>Fluid temperature</td>
<td>°F</td>
</tr>
<tr>
<td>$U$</td>
<td>Overall coefficient of heat transfer in the reactor core, or</td>
<td>$\frac{\text{Btu}}{\text{sec-ft}^2}$ Dimensionless</td>
</tr>
<tr>
<td>$U_{x}$</td>
<td>Control rod reactivity variable</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>$U'$</td>
<td>Overall coefficient of heat transfer in the steam generator</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>$V$</td>
<td>Overall coefficient of heat transfer</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>$W_c$</td>
<td>Mean fluid velocity</td>
<td>ft/sec</td>
</tr>
<tr>
<td>$W$</td>
<td>Mass flow rate of coolant</td>
<td>lbs/sec</td>
</tr>
<tr>
<td>$X$</td>
<td>VA, mass rate of flow of the fluid</td>
<td>lbs/sec</td>
</tr>
<tr>
<td>$X_a$</td>
<td>Control rod position</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>$X_i$</td>
<td>Heat transfer area per unit length of conduit</td>
<td>ft.</td>
</tr>
<tr>
<td>$i$</td>
<td>Index for delayed neutron groups</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>$l_i$</td>
<td>Effective length of inlet piping system</td>
<td>ft.</td>
</tr>
<tr>
<td>$l_o$</td>
<td>Effective length of outlet piping system</td>
<td>ft.</td>
</tr>
<tr>
<td>$l_1$</td>
<td>Effective neutron lifetime</td>
<td>seconds</td>
</tr>
<tr>
<td>SYMBOL</td>
<td>MEANING</td>
<td>UNITS</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>n</td>
<td>Neutron density</td>
<td>Neutron/cu.cm.</td>
</tr>
<tr>
<td>n*</td>
<td>Reduced neutron density</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>n_m</td>
<td>Maximum practical neutron density</td>
<td>Neutron/cu.cm.</td>
</tr>
<tr>
<td>n_o</td>
<td>Demand-power level neutron density</td>
<td>Neutron/cu.cm.</td>
</tr>
<tr>
<td>t</td>
<td>Time</td>
<td>seconds</td>
</tr>
<tr>
<td>v_o</td>
<td>Mean velocity of coolant in outlet piping system</td>
<td>ft/sec.</td>
</tr>
<tr>
<td>v_i</td>
<td>Mean velocity of coolant in inlet piping system</td>
<td>ft/sec.</td>
</tr>
<tr>
<td>x</td>
<td>Position along conduit</td>
<td>ft.</td>
</tr>
<tr>
<td>α</td>
<td>Temperature coefficient of reactivity</td>
<td>°F⁻¹</td>
</tr>
<tr>
<td>k_j</td>
<td>Fraction of prompt neutrons appearing in delayed neutron group &quot;j&quot;</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>δ</td>
<td>Reactivity</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>δk_c</td>
<td>Reactivity contribution of control rod positions</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>δk_c(0)</td>
<td>Initial reactivity contribution of control rods</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>δk_f</td>
<td>Built-in reactivity of fuel</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>δk_p</td>
<td>Reactivity contribution of reactor poisons</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>δk_t</td>
<td>Reactivity contribution due to the fuel temperature</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>ΔH_f</td>
<td>Heat of fission</td>
<td>Btu-cu.cm. sec. Neu°F</td>
</tr>
<tr>
<td>σ(t)</td>
<td>Error signal</td>
<td></td>
</tr>
<tr>
<td>λ_j</td>
<td>Decay constant associated with group &quot;j&quot;</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>μ(t)</td>
<td>Departure of control rod reactivity from its initial value</td>
<td>Dimensionless</td>
</tr>
<tr>
<td>τ_o</td>
<td>Outlet piping system delay time</td>
<td>seconds</td>
</tr>
<tr>
<td>SYMBOL</td>
<td>MEANING</td>
<td>UNITS</td>
</tr>
<tr>
<td>--------</td>
<td>---------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>$\tau_i$</td>
<td>Inlet piping system delay time</td>
<td>seconds</td>
</tr>
<tr>
<td>$\tau_m$</td>
<td>Control rod drive unit time constant</td>
<td>seconds</td>
</tr>
<tr>
<td>$\rho$</td>
<td>Fluid density</td>
<td>seconds</td>
</tr>
</tbody>
</table>
MODULE

Reactor Kinetics
Delayed Neutron Groups
Fuel Element Heat Transfer
Fuel Element to Coolant Heat Transfer
Out of Reactor Core
Outlet Plenum
Piping Delay to Steam Generator
Steam Generator
Out of Steam Generator
Piping Delay to Inlet Plenum
Inlet Plenum
Average Temperature
Comparator
PI Controller
Control Rods
Reactivity
Automatic Scram
Boiler-Power Demand

OFF PAGE CONNECTORS

A B C D
A D
B E F G H
E F G H
F G I
G I J
J K
K L M
K L N
L M
N O
O P
P Q
Q R
R S
S T
T U
U V
V W
W X
X Y
Y Z
Z A
Reactor Kinetics:

\[
\frac{d}{dt} [10n^*] = (0.5)[2000 \delta k][10n^*] - 10(0.64)[10n^*] + 10 \sum_{i=1}^{2} \lambda_i C_i^*
\]

<table>
<thead>
<tr>
<th>Pot</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>05</td>
<td>0.5000</td>
</tr>
<tr>
<td>07</td>
<td>0.6400</td>
</tr>
<tr>
<td>31</td>
<td>0.5000</td>
</tr>
</tbody>
</table>
Delayed Neutron Groups:

\[ \frac{d}{dt} \left[ \frac{C_i^*}{10} \right] = \frac{1}{10^2} \beta_i \left[ 10n^* \right] - \lambda_i \left[ \frac{C_i^*}{10} \right] \]
Fuel Element Heat Transfer:

\[
\frac{dT_f}{dt} \left[ \frac{T_f}{200} \right] = 0.1 \left( \frac{\Delta H_f}{200 M_f C_f} \right) [10n^*] - \left( \frac{UA}{M_f C_f} \right) \left[ \frac{T_f}{200} \right] + 0.1 \left( \frac{5UA}{M_f C_f} \right) \left[ \frac{T_c}{100} \right]
\]
Fuel Element to Coolant Heat Transfer:

\[
\frac{d}{dt} \left[ \frac{T_c}{100} \right] = \left( \frac{2UA}{M_c C_c} \right) \left[ \frac{T_f}{200} \right] - \left( \frac{UA + 2 W c C_c}{M_c C_c} \right) \left[ \frac{T_c}{100} \right] + \\
\left( \frac{2 W c C_c}{M_c C_c} \right) \left[ \frac{T_{ic}}{100} \right]
\]
Out of Reactor Core:

\[
\left[ \frac{T_{oc}}{100} \right] = 10(0.2) \left[ \frac{T_c}{100} \right] - \left[ \frac{T_{ic}}{100} \right]
\]
Outlet Plenum

\[
\frac{d}{dt} \left[ \frac{T_o}{100} \right] = \left( \frac{W_c}{M_o} \right) \left[ \frac{T_{oc}}{100} \right] - \left( \frac{W_c}{M_o} \right) \left[ \frac{T_c}{100} \right]
\]
Piping Delay to Steam Generator:

\[
\frac{T_{ix}}{100} = \frac{T}{100} (t-D)
\]
Steam Generator:

\[
\frac{dT}{dt} \left[ \frac{T_x}{100} \right] = \frac{2W_c}{M_x} \left[ \frac{T_{ix}}{100} \right] + \frac{UA_x}{M_x c} \left[ \frac{T_s}{100} \right] - \frac{UA_x}{M_x c} \left[ \frac{T_x}{100} \right]
\]

\[
\frac{2W_c}{M_x} \left[ \frac{T_x}{100} \right]
\]

<table>
<thead>
<tr>
<th>Pot</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>0.5000</td>
</tr>
<tr>
<td>50</td>
<td>0.6000</td>
</tr>
<tr>
<td>57</td>
<td>0.9999</td>
</tr>
</tbody>
</table>
Out of Steam Generator:

\[
\frac{T_{ox}}{100} = 10(0.2) \left\{ \frac{T_x}{100} \right\} - \left\{ \frac{T_{lx}}{100} \right\}
\]
Piping Delay to Inlet Plenum:

\[
\frac{T_{ix}}{100} = \frac{T_{ox}}{100} (t-D)
\]
\[ \frac{dT_{ic}}{dt} = \frac{W}{M_i} \left( \frac{T_i}{100} \right) - \frac{W}{M_i} \left( \frac{T_{ic}}{100} \right) \]
Average Temperature:

\[
\frac{T_{\text{avg.}}}{100} = 0.5\left(\frac{T_{\text{ox}}}{100} + \frac{T_{\text{ix}}}{100}\right)
\]

<table>
<thead>
<tr>
<th>Pot</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>91</td>
<td>0.5000</td>
</tr>
<tr>
<td>92</td>
<td>0.5000</td>
</tr>
</tbody>
</table>
Comparator:

\[
\left[ \frac{\epsilon}{100} \right] = \left[ 10 \right] \left( \frac{T_{ref}}{1000} \right) - \left[ \frac{T_{avg}}{100Q} \right]
\]

Pot Setting
Q07 0.4000
**PI Controller:**

\[
[2(n_o - n^*)] = (2 \times 10^3 k_c) \int_0^t \frac{1}{10} \left[ -\frac{\epsilon}{100} \right] dt + (2000 k_c \tau_c) \left[ -\frac{\epsilon}{100} \right] - 0.200[10n^*] + 2 n_o^*(0)
\]
Control Rods:

\[
\frac{d^2}{dt^2} [2000\mu] + \left( \frac{1}{\tau_m} \right) \frac{d}{dt} [2000\mu] = \frac{K \times 10^3 [20(n_o - n*)]}{(\frac{m}{\tau_m})[10n*]}
\]
Reactivity:

\[ [2000\text{sk}] = 10(20k) \cdot 10 - 10(0.8) \cdot \frac{T_f}{200} + 2000\mu \]
Automatic Scram:

\[
T = \frac{N}{100} = \frac{T}{100}
\]

Note: A = Threshold

Pot

Setting

36

0.9500
Boiler - Power Demand:

\[ \frac{dT}{dt} \left[ \frac{S}{100} \right] = K_1 \left( \frac{T_x}{100} - \left[ \frac{S}{100} \right] \right) - K_2 A \]

Where A = Throttle Opening

A is the fraction of full open throttle.
PROGRAMMER: ERNEST IVRY HAMILTON, JR.

MACHINE: LOUISIANA STATE UNIVERSITY

DATE: 24 APRIL 1972

PURPOSE: FORTRAN IV-H, ASSEMBLE DIGITAL PROCESS CONTROL COMPUTER SOFTWARE

DOUBLE PRECISION NADC
DOUBLE PRECISION N2ADC
COMMON /ADC/ ADC(24),OADC(24)
COMMON /NADC/ NADC(24),N2ADC(24)
COMMON /TIME/ IHR,MIN,ISEC
COMMON /CONTAB/ CT(70),IDAMSET, IDS(12)
COMMON /RUN1/,IRUN1(4),TRUN,ITRUN(4)
COMMON /ALARML, ALARML(24),ALARMU, ALARMU(24)
COMMON /IDEM/,IDEM(13),VARTAB,VT(70)
COMMON /FLAG/,IFLAG(8),TRUNC,ITRUNC(8)
CONNECT (2Z60,CLOCK)
CONNECT (2Z61,SUB1)
CONNECT (2Z62,SUB2)
CONNECT (2Z63,SUB3)
CONNECT (2Z64,SUB4)
CONNECT (2Z65,SUB5)
READ IN THE RUN INTERVALS
READ(5,10)(IRUNI(I),I=1,4)
FORMAT(4I5)

READ IN LOWER ALARM LIMITS
READ(5,40)(ALARML(I),I=1,8)
READ(5,40)(ALARML(I),I=9,16)
READ(5,40)(ALARML(I),I=17,24)
FORMAT(8F10.5)

READ IN UPPER ALARM LIMITS
READ(5,40)(ALARMU(I),I=1,8)
READ(5,40)(ALARMU(I),I=9,16)
READ(5,40)(ALARMU(I),I=17,24)
IHR=0
MIN=0
ISEC=0

READ IN NAMES OF VARIABLES
1ST LINE OF NAME IS 8 COL. WIDE
AND STARTS IN COL. THAT IS 1
PLUS MULTIPLE OF 10
READ(5,50)(NADC(I),I=1,8)
READ(5,50)(NADC(I),I=9,16)
READ(5,50)(NADC(I),I=17,24)
FORMAT(8(A8,2X))
2ND LINE OF NAME IS 8 COL. WIDE  
AND STARTS IN COL. THAT IS 1 
PLUS MULTIPLE OF 10 

READ(5,50)(N2ADC(I),I=1,8)  
READ(5,50)(N2ADC(I),I=9,16)  
READ(5,50)(N2ADC(I),I=17,24) 

READ IN THE MAXIMUM VALUES  
OF THE ANALOG VARIABLES  
WITH ADC NUMGER PLUS 1 EQUAL  
TO ADC NUMBER IN DIGITAL 

READ(5,60)(CT(I),I=1,8)  
READ(5,60)(CT(I),I=9,16)  
READ(5,60)(CT(I),I=17,24) 

READ IN THE MAXIMUM VALUES OF DAMS--  
(DIGITAL ANALOG MULTIPLIERS)  
AND DAMSET NUMBER PLUS 24 EQUAL  
TO 1 GREATER THAN DAC NUMBER IN  
THE DIGITAL. 

READ(5,60)(CT(I),I=25,32)  
READ(5,60)(CT(I),I=33,39) 
60  
FORMAT(8E10.0)  

READ IN CONTROL PROGRAM RUNI  
C  

READ(5,150)(CT(I),I=41,48)  
150  
FORMAT(8F10.2)
DO 160 I=1,J
J=J+40
160 ITRUNC(I)=CT(J)
C TO ZERO CONTROL RUN INTERVAL FLAGS
C
DO 200 I=1,8
200 IFLAG(I)=3
DO 100 I=1,12
100 IDS(I)=0
IDEM(13)=0
DO 20 I=1,4
20 ITRUN(I)=IRUNI(I)
C TO SET INDICATORS TO NOT ALLOW MESSAGES TO BE
C PRINTED FROM CONTROL PROGRAMS
C
DO 300 I=51,58
300 CT(I)=0.0
CALL LTDA(1.0,400)
VT(25)=400.0
CT(26)=0.700
CALL SSRM(64)
CALL CONEC
STOP
END

SUBROUTINE CLOCK
COMMON /CONTAB/ CT(70) /IDAMSET/ IDS(12)
COMMON /TIME/ MIN, ISEC
THE TIME FOR THE REAL TIME CLOCK

ISEC=ISEC+1
IF(ISEC*GE*60)GOTO1
3       IF(MIN*EQ*60)GOTO2
        GOTO4
1       ISEC=0
        MIN=MIN+1
        GOTO3
2       MIN=0
        IHR=IHR+1
        IF(IHR*LE*24)GOTO4
        IHR=0
4       CONTINUE

TO DOWN COUNT TRUN

DO 20 I=1,4
     ITRUN(I)=ITRUN(I)-1
20

TO TRIGGER THE DIFFERENT INTERRUPTS

IF(ITRUN(1)*GT*0)GOTO 30
S       LI,1    X’0061’
S       CAL1,5  1
        ITRUN(1)=IRUNI(1)
30       IF(ITRUN(2)*GT*0)GOTO 40
S       LI,1    X’0062’
S     CAL1.5  1
ITRUN(2)=IRUNI(2)
40    IF(ITRUN(3) GT 0) GOTO 50
S     LI,1  X'0063'
S     CAL1.5  1
ITRUN(3)=IRUNI(3)
50    IF(IDEM(13) EQ 0) GOTO60
IF(ITRUN(4) GT 0) GOTO 60
S     LI,1  X'0064'
S     CAL1.5  1
ITRUN(4)=IRUNI(4)
60    CONTINUE
1000  CONTINUE
DO 70 I=1,8
IF(IFLAG(I) GE 1) GOTO69
ITRUNC(I)=ITRUNC(I)-1
IF(ITRUNC(I) GT 0) GOTO70
65    IFLAG(I)=2
J=I+40
ITRUNC(I)=CT(J)
69    CONTINUE
S     LI,1  X'0063'
S     CAL1.5  1
ITRUN(3)=IRUNI(3)
70    CONTINUE
END

SUBROUTINE SUB1
COMMON /CONTAB/ CT(70)/IDAMSET/ IDS(12)
COMMON /ADC/ ADC(24),OADC(24)/VARTAB/ VT(70)
DO 5 I=1,24
C
to save the last value in engineering units
C by storing into OADC(I)
C
5
OADC(I)=ADC(I)
C
to scan process
C
I=0
N=0
10
N=N+1
CALL CRAC (I,ADC(N))
I=I+1
IF(I.LT.24)GOTO 10
C
to convert to engineering units
C
DO 30 I=1,24
ADC(I)=ADC(I)*CT(I)
30
C
to filter
C and store in vartab--VT(I)
C at present there is no filtering
C
DO 40 I=1,24
VT(I)=ADC(I)
VT(26)=CT(26)
C
used to set dams--(digital to analog
C multipliers)
SUBROUTINE SUB3
COMMON /FLAG/ IFLAG(8)
IF(IOFLAG(1) .EQ. 2) CALL CONTR1
IF(IOFLAG(2) .EQ. 2) CALL CONTR2
IF(IOFLAG(3) .EQ. 2) CALL CONTR3
IF(IOFLAG(4) .EQ. 2) CALL CONTR4
IF(IOFLAG(5) .EQ. 2) CALL CONTR5
IF(IOFLAG(6) .EQ. 2) CALL CONTR6
IF(IOFLAG(7) .EQ. 2) CALL CONTR7
IF(IOFLAG(8) .EQ. 2) CALL CONTR8
END

SUBROUTINE CONTR1
COMMON /TIME/ IHR MIN ISEC
COMMON /COLST/ ICOLST(8)
COMMON /CONTAB/ CT(70) IDAMSET IDS(12)
COMMON /FLAG/ IFLAG(8) VARTAB VT(70)
IF(IOCOLST(1) .EQ. 0) GOTO 40
VT(12) IS ACTUALLY A NEGATIVE NUMBER.
ERRNOW = CT(37) + VT(12)
ACTKP = CT(38) * (ERRNOW - ERROLD)
ACTKI = CT(39) * ERRNOW
DELM = ACTKP + ACTKI
ERROLD = ERRNOW
CT(60) = VT(25) + DELM
IDS(1) = 1
GOTO 50
ERROLD = 0
CT(37) = - VT(12)
ICOLST(1) = 1
50 CONTINUE
IF(CT(51).NE.1.0)GOTO51
WRITE(7,9999) IHR,MN,ISEC,ERRNOW,DELM,CT(38),
*CT(39),ACTKP,ACTKI,CT(37),CT(60)
9999 FORMAT(*TIME *,3(I2,1X),' CONTROL PROGRAM 1 RAN '*,
** ERROR NOW = ',F10.2, ' DEG F',/, *
**DELM = ',F10.2, ' KP = ',F10.2, ' KI = ',F10.2,/
**PROP ACTION = ',F10.2, ' INT ACTION = ',F10.2, *
** TARGET = ',F10.2, 'MANIPULATED VAR = ',F10.2)
51 CONTINUE
IFLAG(1)=0
END

SUBROUTINE CONTR2
COMMON /FLAG/ IFLAG(8)/TRUNC/ ITRUNC(8)
WRITE(7,9999)
9999 FORMAT(*CONTROL PROGRAM 2 RAN*)
IFLAG(2)=0
END

SUBROUTINE CONTR3
COMMON /FLAG/ IFLAG(8)/TRUNC/ ITRUNC(8)
WRITE(7,9999)
9999 FORMAT(*CONTROL PROGRAM 3 RAN*)
IFLAG(3)=0
END
END IF (FD ECP PRECISION TEMP1)
DOUBLE PRECISION TEMP2
COMMON /COSTT/ ICOAST(8)
COMMON /TIME/ IHR, MIN, ISEC
SUBROUTINE CONTR8
COMMON /FLAG/ IFLAG(8), TRUNC, ITRUNC(8)
WRITE(7, 9999)
9999 FORMAT(9******CONTROL PROGRAM 8 RAN*)
IFLAG(8) = 0
END

SUBROUTINE SUB4
COMMON /TIME/ IHR, MIN, ISEC, VARTAB, VT(70)
COMMON /CONTAB/ CT(70), IDAMSET, IDS(12)
COMMON /ADC/ ADC(24), DADC(24)
COMMON /IDEM/ IDEM(13)
C TO PRINT THE VALUES THAT ARE IN VARTAB
C FOR THE TREND LOG
C
IDPT = IDEM(13)
WRITE(6, 10) IHR, MIN, ISEC, (VT(IDEM(I)), I = 1, IDPT)
10 FORMAT(1X, I2, 2X, I2, 2X, I2, 12(1X, F10.3, 1X))
END

SUBROUTINE SUB5
DOUBLE PRECISION NADC
DOUBLE PRECISION N2ADC
DOUBLE PRECISION TEMP1
DOUBLE PRECISION TEMP2
COMMON /COLDST/ ICOLST(8)
COMMON /TIME/ IHR,MIN,ISEC
COMMON /FLAG/ IFLAG(8), TRUNC, ITRUNC(8)
COMMON /CONTAB/ CT(70), IDAMSET, IDS(12)
COMMON /RUNI/ IRUNI(4), TRUN, ITRUN(4)
COMMON /ADC/ ADC(24), OADC(24)
COMMON /ALARML/ ALARML(24), ALARMU, ALARMU(24)
COMMON /NADC/ NADC(24), N2ADC(24)
COMMON /IDEM/ IDEM(13), VARTAB, VT(70)

C
TO COMMUNICATE WITH THE HUMAN OPERATOR

C
WRITE(7,10)
10 FORMAT('IFUNCODE IDPT VALUE')
   CONTINUE
   WRITE(7,11)
   FORMAT('IIII IFUNCODE EXAMPLE TYPE-INO*/
   '*02 04 10,00000000 FOLLOWED BY NL THEN EOM*/
   READ(7,12)IFNCOD, IDPT, VALUE
12 FORMAT(12,1X,12,F10.5)
   GOTO(21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
   *36, 37, 38), IFNCOD
   GOTO41

C
UPDATE OF CONSTANTS IN CONTROL TABLE(IDPT) TO VALUE.

C
21 OVALUE=CT(IDPT)
   WRITE(7,100)IDPT, VALUE
100 FORMAT('**** UPDATE OF CONSTANTS ***************/
   ** IN CONTROL TABLE(*.I2,*) TO *.F10.5)
IF (IDPT GT 50) GOTO 1110

1100 CALL QUEST(OVALUE, VALUE, IXX)
IF (IXX EQ 2) GOTO 40
CT (IDPT) = VALUE
GOTO 40

1110 IF (IDPT GT 58) GOTO 1100
J = IDPT - 51
IDS (J) = 1
GOTO 40

C
C UPDATE RUN INTERVAL(IDPT) TO VALUE.
C
22 OVALUE = IRUNI (IDPT)
WRITE (7, 101) IDPT, VALUE

101 FORMAT ('***** UPDATE OF CONSTANTS ***************')
** IN TASK RUN INTERVAL (', I2, ' ) TO ', F10.5)
CALL QUEST (OVALUE, VALUE, IXX)
IF (IXX EQ 2) GOTO 40
IRUNI (IDPT) = VALUE
GOTO 40

C
C UPDATE ALARM LOWER LIMIT (IDPT) TO VALUE.
C
23 OVALUE = ALARML (IDPT)
WRITE (7, 102) IDPT, VALUE

102 FORMAT ('***** UPDATE OF CONSTANTS ***************')
** IN LOWER ALARM LIMIT (', I2, ' ) TO ', F10.5)
CALL QUEST (OVALUE, VALUE, IXX)
IF (IXX EQ 2) GOTO 40
ALARML (IDPT) = VALUE
GOTO 40

C
C UPDATE ALARM UPPER LIMIT (IDPT) TO VALUE.

24   OVALUE=ALARMU(IDPT)
25   WRITE(7,103)IDPT,OVALUE
103   FORMAT('**** UPDATE OF CONSTANTS ******************'/
* IN UPPER ALARM LIMIT(*,I2,*') TO *,F10.5)
   CALL QUEST(OVALUE,VALUE,IXX)
   IF(IXX.EQ.2)GOTO40
   ALARMU(IDPT)=VALUE
   GOTO40

C LIST ON LOGGING DEVICE ALARM LIMITS.

C

25   DO 200 N=1,8
    M=N+8
    L=N+16
    WRITE(6,200)N,ALARML(N),N,ALARMU(N),
    *M,ALARML(M),M,ALARMU(M),
    *L,ALARML(L),L,ALARMU(L)
200   FORMAT(1X,*ALARML(*,I1,*')*F10.2,
*2X,*ALARML(*,I2,*')*F10.2,
*2X,*ALARMU(*,I2,*')*F10.2,
*2X,*ALARML(*,I2,*')*F10.2,
*2X,*ALARMU(*,I2,*')*F10.2)
   GOTO40

C CHANGE NAME OF ANALOG INPUT (IDPT).

C

26   WRITE(7,1000)IDPT
1000  FORMAT('**** CHANGE NAME ***********************'/
* OF ANALOG VARIABLE(*,I2,*'))
WRITE(7,201)
201 FORMAT(‘TYPE IN 16 CHARs NAME’)
READ(7,202)TEMP1,TEMP2
202 FORMAT(A8,A8)
   WRITE(7,1001)NADC(IDPT),N2ADC(IDPT),TEMP1,TEMP2
1001 FORMAT(‘* DO YOU WANT TO CHANGE NAME’/
**FROM ’A8,A8,’ TO ’,A8,A8/
**TYPE IN 1 IF CORRECT OR 2 IF NOT CORRECT’)
READ(7,1002)IEC
1002 FORMAT(I1)
   IF(IEC.EQ.2)GOTO1003
   NADC(IDPT)=TEMP1
   N2ADC(IDPT)=TEMP2
   WRITE(7,1004)NADC(IDPT),N2ADC(IDPT)
1004 FORMAT(‘* THE NAME IS CHANGED TO ’,A8,A8)
   GOTO40
1003 WRITE(7,1005)NADC(IDPT),N2ADC(IDPT)
1005 FORMAT(‘* THE NAME REMAINS ’,A8,A8)
   GOTO40
C
C  LIST ON LOGGING DEVICE NAMES OF ANALOG INPUTS
C
27 DO 205 J=1,8
   M=J+8
   N=J+16
205 WRITE(6,206)J,NADC(J),N2ADC(J),M,NADC(M),
   *N2ADC(M),N,NADC(N),N2ADC(N)
206 FORMAT(1X,’NADC(‘,I1,’),’A8,A8,4X,’NADC(‘,I2,’),’A8,A8)
   *GOTO40
C
C  START TRENDING LOG, IDPT = NUMBER OF VARIABLES
C TRENDED MORE THAN 10 VARIABLES
C
28    CONTINUE
C TO IF,IDEM(13),GE,10,GOTO230
C DO 209 J=1,IDPT
C 232    WRITE(7,208)
C 208    FORMAT(‘TYPE IN ID OF ANALOG INPUT, FORMAT I2’)!
C 209    READ(7,210)IDEM(J)
C 210    FORMAT(I2)
C 219    WRITE(6,220)(NADC(IDEM(I)),IDEM(I),I=1,IDPT)
C 220    FORMAT(1X,’HR MIN SEC’,10(1X,10,A8,1X,I2))
C 221    WRITE(6,221)(N2ADC(IDEM(I)),IDEM(I),I=1,IDPT)
C 221    FORMAT(11X,10(1X,A8,1X,I2))
C     IDEM(13)=IDPT
C     GOTO40
C C STOP TREND LOGGING
C
29    IDEM(13)=0000
C     GOTO40
C C ADD ONE VARIABLE TO THE TREND LOG
C C IDEM(13) EQUALS NUMBER OF VARIABLES ON TREND
C C LOG THE MAXIMUM IS 10
C C
30    IF,IDEM(13),GE,10,GOTO230
C     IDEM(13)=IDEM(13)+1
C     J=IDEM(13)
C     IDPT=J
C     GOTO 232
C 230    WRITE(7,231)
C 231    FORMAT(‘YOU CANNOT ADD TO THE TREND ’,
*LOG MORE THAN 10 VARIABLES*)
GOTO 40
C
C TO DELETE ONE VARIABLE FROM THE TREND LOG
C
31  MMM=IDEM(13)
    DO 240 JJ=1,MMM
    IF(IDPT.EQ.IDEM(JJ))GOTO 250
240  CONTINUE
    WRITE(7,241) IDPT
241  FORMAT('YOU ARE NOT TRENDING IDEM(',I2,')')
    GOTO40
250  M=IDEM(13)-JJ
    IDEM(13)=IDEM(13)-1
    DO 251 KK=1,M
    MM=JJ+1
    IDEM(JJ)=IDEM(MM)
251  JJ=JJ+1
    J=IDEM(13)
    IDPT=J
    GOTO219
C
C TO GIVE THE CLOCK THE CORRECT TIME OF DAY
C
32  WRITE(7,60)
60  FORMAT('MY CLOCK SEEMS TO HAVE ',',
    '*LOST THE TIME OF DAY COULD YOU',',',
    '*TELL ME THE TIME IF IT WERE ',',
    '*TO BE 15 SECONDS AFTER 0830 ',',
    '*YOU WOULD TYPE THAT IN AS 083015 ',',',
    '*NOTE I LIKE MILITARY TIME ALSO ',')
    READ(7,61)IHR,MIN,ISEC
00004870
00004880
00004890
00004900
00004910
00004920
00004930
00004940
00004950
00004960
00004970
00004980
00004990
00005000
00005010
00005020
00005030
00005040
00005050
00005060
00005070
00005080
00005090
00005100
00005110
00005120
00005130
00005140
00005150
00005160
00005170
00005180
61 FORMAT(3I2)
62 WRITE(7,62)IHR,MIN,ISEC
62 FORMAT('THANK YOU FOR THE TIME',/,'** THE TIME IS NOW',1X,3I2)
   GOTO40

C TO TRIGGER CONTROL PROGRAMS
C
33 IF(IFLAG(IDPT)*EQ.0)GOTO350
   WRITE(7,300)IDPT
300 FORMAT('CONTROL PROGRAM','I2','** ALREADY WAITING TO RUN')
350 IFLAG(IDPT)=1
   WRITE(7,360)IDPT
360 FORMAT('CONTROL PROGRAM','I2',' TRIGGERED')
   GOTO40
C TO TURN OFF A CONTROL PROGRAM
C
34 IF(IFLAG(IDPT)*EQ.3)GOTO370
   IFLAG(IDPT)=3
   J=50+IDPT
   CT(J)=0.0
   ICOLST(IDPT)=0
   WRITE(7,374)IDPT
374 FORMAT('CONTROL PROGRAM','I2',' MASKED')
   GOTO40
370 WRITE(7,375)IDPT
375 FORMAT('CONTROL PROGRAM','I2',' ALREADY MASKED')
   GOTO40
C TO LIST STATUS OF CONTROL PROGRAMS
35 DD 390 I=1*8
36 WRITE(7,380)I,IFLAG(I)
380 FORMAT(' CONTROL PROGRAM ',I2,' FLAG SET ',I2)
390 CONTINUE
400 WRITE(7,400)
   FORMAT(1'A 3 MEANS MASKED ',I2,' 2 MEANS TIMED OUT',I2,
         ' 1 MEANS TRIGGERED ',I2,' 0 MEANS NOT MASKED')
GOTO40

C TO ALLOW CONTROL PROGRAM(IDPT) TO PRINT MESSAGES
C TO OPERATOR

C 36 IF(IDPT.LT.1)GOTO410
37 IF(IDPT.GT.8)GOTO410
38 J=IDPT+50
39 CT(J)=1*0
40 WRITE(7,430)IDPT
410 FORMAT(' CONTROL PROGRAM ',I2,' WILL BE ALLOWED',I2,
         ' TO PRINT MESSAGES')
420 GOTO40

C TO STOP A CONTROL PROGRAM(IDPT) FROM
C PRINTING MESSAGES TO OPERATOR
C

C 37 IF(IDPT.LT.1)GOTO510
38 IF(IDPT.GT.8)GOTO510
39 J=IDPT+50
40 CT(J)=0*0
WRITE(7,530)IDPT
530 FORMAT(’CONTROL PROGRAM ’*I2’,
*’ WILL STOP PRINTING MESSAGES’)
GOTO40
510 WRITE(7,520)IDPT
520 FORMAT(’THERE IS NOT A CONTROL ’*,
*’PROGRAM ’*I2)
GOTO40
C C TO CHANGE LOAD ON HEAT EXCHANGER
C C 38 WRITE(7,550)
550 FORMAT(’CHANGE IN LOAD ON HEAT EXCHANGER’)
   OVALUE=CT(26)
   CALL QVEST(OVALUE,VALUE,IXX)
   IF(IXX.EQ.2)GOTO40
   CT(26)=VALUE
   CALL LTDA(2,CT(26))
   CALL SSRM(9)
   GOTO40
C C TO PRINT THAT ILLEGAL FUNCTION CODE
C WAS ENTERED
C C 41 WRITE(7,42)IFNCOD
42 FORMAT(1X,’YOU HAVE ENTERED AN ILLEGAL ’*,
*’FUNCTION ’*I2)
40 CONTINUE
END
SUBROUTINE QUEST(VALO,VALN,Ixx)
  Ixx=0
  PERCEN=ABS(((VALN-VALO)/VALO)*100.0)
  IF(PERCEN.LT.5.0)GOTO20
  WRITE(7,10)VALO,VALN
10  FORMAT(**** DO YOU WANT TO CHANGE ?
       *'THE VALUE FROM ','F10.2,' TO ','F10.2,/
       ** TYPE IN 1 IF CORRECT OR 2 IF NOT CORRECT')
  READ(7,11)Ixx
11  FORMAT(I1)
  IF(Ixx.EQ.1)GOTO20
  IF(Ixx.EQ.2)GOTO100
  FORMAT(YOU TYPED IN A ',I1)
  GOTO1
100  WRITE(7,13)
13  FORMAT(**** THERE WILL BE NO CHANGE ?
       **IN THE VALUE')
  GOTO25
20  WRITE(7,14)VALO,VALN
14  FORMAT(****** THE VALUE WILL BE ?
       *'CHANGED FROM ','F10.2,' TO ','F10.2)
25  CONTINUE
RETURN
END

SYMBOL GO,SI,LO

DEF CONEC
<table>
<thead>
<tr>
<th>REDUCED</th>
<th>AVG FUEL</th>
<th>TEMP OUT</th>
<th>TEMP OUT</th>
<th>T* INLET</th>
<th>AVG TEMP</th>
<th>COOL IN</th>
<th>TEMP OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>T INLET</td>
<td>T INLET</td>
<td>CNTL RD</td>
<td>STEAM</td>
<td>TEMP AVG</td>
<td>ERROR</td>
<td>5555555</td>
<td>66666666</td>
</tr>
<tr>
<td>77777777</td>
<td>TEMP*</td>
<td>LOAD ON</td>
<td>00000000</td>
<td>11111111</td>
<td>22222222</td>
<td>33333333</td>
<td>44444444</td>
</tr>
<tr>
<td>N DEN*</td>
<td>TEMP*</td>
<td>F CORE F</td>
<td>MIXING F</td>
<td>EXCH -F</td>
<td>EXCH F</td>
<td>CORE F</td>
<td>EXCH -F</td>
</tr>
<tr>
<td>77777777</td>
<td>SET PT*</td>
<td>EXCH*</td>
<td>00000000</td>
<td>11111111</td>
<td>22222222</td>
<td>33333333</td>
<td>44444444</td>
</tr>
<tr>
<td>1.0</td>
<td>2000.0</td>
<td>1000.0</td>
<td>1000.0</td>
<td>1000.0</td>
<td>1000.0</td>
<td>1000.0</td>
<td>1000.0</td>
</tr>
<tr>
<td>1000.0</td>
<td>1000.0</td>
<td>1000.0</td>
<td>1000.0</td>
<td>1000.0</td>
<td>1000.0</td>
<td>1000.0</td>
<td>1000.0</td>
</tr>
<tr>
<td>0.005</td>
<td>1000.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1000.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.1</td>
<td>0.01</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>10.0</td>
<td>60.0</td>
<td>60.0</td>
<td>60.0</td>
<td>300.0</td>
<td>300.0</td>
<td>300.0</td>
<td>300.0</td>
</tr>
</tbody>
</table>
VITA

Ernest Ivry Hamilton, Jr. was born in Lake Charles, Louisiana, 17 July, 1947.

Secondary education was obtained in Hackberry at Hackberry High School, from which he graduated in 1965. In June, 1965, he entered Louisiana State University at Baton Rouge, where he received a Bachelor of Science in Chemical Engineering in May, 1970.

In September, 1970, he enrolled in the Graduate School of Louisiana State University. At present he is a candidate for a degree of Master of Science in the Department of Nuclear Engineering.